ANSI/IEEE Std 488.1-1987
(Revision of ANSI/IEEE Std.488-1978)

An American National Standard

IEEE Standard Digital Interface for Programmable Instrumentation

Sponsor
Automated Instrumentation Technical Committee
of the
IEEE Instrumentation and Measurement Society

Approved June 11, 1987
IEEE Standards Board

Approved February 2, 1988
American National Standards Institute

ISBN 471-62222-2

IEEE Standards documents are developed within the Technical Committees of 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 which 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 once 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 technical 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
345 East 47th Street
New York, NY 10017
USA

Foreword

(This Foreword is not a part of ANSI/IEEE Std 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation.)

IEEE Std 488 has enjoyed continuous and widespread use since its initial publication in 1975. The first revision occurred in 1978 as a result of practical experience and recognition that certain clauses needed clarification to improve compatibility among independently designed products. No major changes were made in 1978; many changes were pure editorial, however, twenty clauses had textual changes with technical implications, though none contradicted the concepts as defined in the original publication. Supplement A was introduced in 1980 to correct one minor deficiency in the controller function related to “take control synchronously.”

A systematic review has been undertaken as a result of both the normal 5-year review cycle and related work on IEEE 488 device-dependent message syntax structures. In addition, there was a strong desire on the part of both IEEE participants and our IEC colleagues to bring equivalent standards (IEC 625-1) into closer alignment. ANSI/IEEE Std 488.1-1987 represents the culmination of this review cycle. Again, no major technical changes have been made, and care has been exercised to preserve compatibility with earlier versions of IEEE Std 488.

The use of “488.1” was deemed appropriate to identify the close relationship to a companion document, ANSI/IEEE Std 488.2-1987, IEEE Standard Codes, Formats, Protocols and Common Commands. Three of the clauses in which changes and clarifications have been made in IEEE Std 488.1 are attributable directly to the needs of IEEE Std 488.2. The changes in IEEE Std 488.1 that could have a bearing on technical compatibility among independently designed products are as follows:

* Clauses 1.4.2, 1.4.3, 2.8.1, 2.8.3, 2.8.5; RL Function, more flexible interpretation and use

Clause 2.3.3.3; SH FCTN, minimize TACS when no LACS

Clause 2.4.2; AH FCTN., remove text obsoleted by Supplement A

Clauses 2.5.1, 2.6.1; T & L FCTN's; clarify primary/secondary address use

Clauses 2.5.5, 2.6.5; T & L FCTN's; use of TON and LON when no C FCTN

* Clause 2.7.5; SR FCTN., preclude redundant RQS messages

* Clause 2.10.5; DC FCTN., minimize inadvertent loss of information

Clause 2.13.5; Remote Message Coding, clarify notes in Table

Clause 4; Mechanical Spec., align with current EMC practice and IEC revisions 625-1

Clause 5.7; Unimplemented Interface Message Handling, new clause to clarify

Appendix C; clarify notation use for driver types E1/E2 in Sect. 3.3

Appendix H; Description of Interface Parameters on Data Sheets, additional, parameter description guidelines

Appendix I; Address Switch Labels & Interface Status Indicators, additional, guidelines

Appendix J; Recommended Methods for Reducing the Effects of Radiated and Conducted Interference for Devices Specified in this Standard, additional, EMC performance, cables and devices

NOTES:

1 — * In direct support of IEEE Std 488.2

2 — Appendix H and IAppendix J provide further alignment with pending IEC 625-1 revisions

The ANSI/IEEE Std 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation, deals with systems that use a byte-serial, bit-parallel means to transfer digital data among a group of instruments and system components. The interface system described herein is optimized as an inter-device interface for system components in relatively close proximity able to communicate over a contiguous party-line bus system.

This document contains seven sections as follows.

Section 1 contains the scope, the object, basic definitions, and summary description of the interface.

Section 2 deals with functional concepts and specifications of the interface system described in this standard. One or more interface functions contained within a device are each able to process messages and change states to maintain an orderly flow of information among a set of interconnected devices.

Section 3 deals with the electrical realization of the interface in order to transfer messages among a set of interconnected devices.

Section 4 deals with the mechanical realization of the interface in order to implement the electrical aspects of the interface system.

Section 5 deals with system considerations that must be given to the design of an individual device in order to make it compatible with other devices of a measurement system.

Section 6 deals with system considerations that must be recognized by the user of devices designed in accordance with this standard.

Appendixes deal with explanatory matter and examples.

In order to interconnect and program equipment designed in accordance with this standard, the user should have knowledge of Sections 1 and 6. If the coding and transfer of messages is not done automatically by the apparatus to be programmed, it will be necessary that the user have knowledge of Section 2. General familiarity with the other sections is recommended. The user must also be familiar with device-dependent characteristics of apparatus that may be used in a system, but that are beyond the scope of this standard.

This standard defines an interface with the objective to assure that messages may be accurately communicated between two or more devices in a system, but does not guarantee that each device will interpret properly all possible messages sent to it or will properly generate all necessary messages. A wide latitude of interface capability is permitted within the scope of this standard which may permit operational incompatibility among interconnected devices.

A device designer must have sufficient awareness of the characteristics of systems which might include his device in order to select correctly among the options provided in this standard. Likewise, a system configurator must have sufficient awareness of the options included in each of the devices in his system in order to ensure that the correct communication techniques are used.

This standard does not specify the device-dependent or operational characteristics required for complete system compatibility. Therefore, following the rules and procedures of this standard alone will not guarantee unconditional compatibility.

The interface specified by this standard includes patented matter which pertains only to the three-wire handshake as defined throughout 2.3 and 2.4 and summarized in Appendix B.

The IEEE Standards Office calls attention to the fact that it is claimed that the three-wire handshake referenced in 2.3 and 2.4 is the subject of one US patent and corresponding patents of foreign countries owned by the Hewlett-Packard Company. Although these patents appear to cover the subject of 2.3 and 2.4 in this standard, the IEEE takes no position with respect to patent validity. The Hewlett-Packard Company has assured the IEEE that it is willing to grant a license

under these patents on reasonable and nondiscriminatory terms and conditions to anyone wishing to obtain such a license. The Hewlett-Packard Company's undertakings in this respect are on file with the IEEE Standards Office and the license details may be obtained from the legal department of the Hewlett-Packard Company whose address is 3000 Hanover Street, Palo Alto, CA 94304.

This standard is based on work initiated by the International Electrotechnical Commission (IEC) within Technical Committee 65, Subcommittee 65C, Working Group 3 (formerly TC66/WG3), and follows the general concepts of a standard prepared by the IEC. This standard influenced, and was influenced by, working documents in the IEC.

The two IEEE Committees responsible for the preparation and evaluation of this standard within the US were the Instrumentation and Measurements Society Subcommittee on Instrument/computer Interfaces (which also serves as the US Advisory Committee to US representatives on IEC SC 65/WG3) of the Instrumentation and Measurements Society Technical Committee on Automated Instrumentation.

The “helpful note” on metric threads found in previous editions has been deleted since metric thread use is common IEEE 488 practice. Consequently, the recommendation to coat such parts in black material to call attention to metric threads is also considered unnecessary. Electrical conductivity on the surface of these parts is, however, still considered essential.

Readers of Std ANSI/IEEE488.1-1987 are encouraged to read also the companion ANSI/IEEE Std 488.2-1987 . Further, it is to be noted that full compatibility with IEEE Std 488.2 requires implementation of all * revisions indicated previously in this Foreword.

[Product/implementations based solely on ANSI/IEEE Std 488.1-1987 may not necessarily provide full IEEE Std 488.2 compatibility/functionality.]

Participants on the Instrument/Computer Interfaces Subcommittee were:

Don Loughry *, *Chair*

Bruce Choyce*	Jeffrey Kodosky	Dana Trout*
Bob Cram*	William Maciejewski*	Don Ware*
Stephen Greer	Tom Leedy	
Damon Hart	Glen Meldrum*	

*Voting members at time this revision was approved.

The following persons were on the balloting committee that approved this document for submission to the IEEE Standards Board:

David Ahlgren	Larry Gross	Jerry Merritt
John Barker	Carl Hagerling	L F. Moebus
Steve Barryte	Chris Hancock	Charles Osborn
Richard Day	Faisal Imdad	Larry Ross
Ron Doss	B. Kowaluk	Eric Sacher
Richard Drews	Robert Kurkjian	Milton Slade
Gary Gallagher	Thomas Leedy	J. R. Weger
Bernard Gollomp	Fred Liguri	Jim Weitenhagen
Arnie Greenspan	Don Loughry	D. Williamson
Bill Gustafson	John McGlaughlin	

When the IEEE Standards Board approved this standard on June 11, 1987, it had the following membership:

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

James H. Beall
Dennis Bodson
Marshall L. Cain
James M. Daly
Stephen R. Dillon
Eugene P. Fogarty
Jay Forster
Kenneth D. Hendrix
Irvin N. Howell
Leslie R. Kerr
Jack Kinn
Irving Kolodny
Joseph L. Koepfinger*
Edward Lohse
John May
Lawrence V. McCall
L. Bruce McClung
Donald T. Michael*
L. John Rankine
John P. Riganati
Gary S. Robinson
Frank L. Rose
Robert E. Rountree
Sava I. Sherr*
William R. Tackaberry
William B. Wilkens
Helen M. Wood

*Member emeritus

CLAUSE PAGE

CLAUSE PAGE

An American National Standard
IEEE Standard Digital Interface for Programmable Instrumentation

1. General

1.1 Scope

This standard applies to interface systems used to interconnect both programmable and nonprogrammable electronic measuring apparatus with other apparatus and accessories necessary to assemble instrumentation systems. It applies to the interface of instrumentation systems, or portions of them, in which the:

1) Data exchanged among the interconnected apparatus is digital (as distinct from analog)
2) Number of devices that may be interconnected by one contiguous bus does not exceed 15
3) Total transmission path lengths over the interconnecting cables does not exceed 20 m
4) Data rate across the interface on any signal line does not exceed 1 Mb/s.

The basic functional specifications of this standard may be used in digital interface applications which require longer distances, more devices, increased noise immunity, or combinations of these. Different electrical and mechanical specifications may be required (for example, symmetrical circuit configurations, high threshold logic, special connectors, or cable configurations) for these extended applications.

This standard may also be applicable to other instrumentation system elements such as processors, stimulus, display, or storage devices, and terminal units found useful in instrumentation systems. It applies generally to laboratory and production test environments which are both electrically quiet and restricted as to physical dimensions (distances between the system components).

This standard deals only with the interface characteristics of instrumentation systems to the exclusion of design specifications consideration of radio-interface regulations, performance requirements, and safety requirements of apparatus.

NOTE — For the latter two items, reference is made to IEC Publication 348 (1978) Safety Requirements for Electronic Measuring Apparatus [3] and IEC Publication 359 (1971) Expression of the Functional Performance of Electronic Measuring Equipment. [4] [1]

[1]Number in brackets correspond to those of the references in 1.5 standard.

A primary focus of this standard is to set forth an interface system to interconnect self-contained apparatus to other apparatus by external means. This same standard may be applied to interconnecting the internal subsections within a self-contained equipment.

1.2 Object

This standard is intended:

1) To define a general-purpose system for use in limited distance applications
2) To specify the device-independent mechanical, electrical, and functional interface requirements that the apparatus shall meet in order to be interconnected and communicate unambiguously via the system.
3) To specify the terminology and definitions related to the system
4) To enable the interconnection of independently manufactured apparatus into a single functional system
5) To permit apparatus with a wide range of capability—from the simple to the complex—to be interconnected to the system simultaneouly
6) To permit direct communication between the apparatus without requiring all messages to be routed to a control or intermediate unit
7) To define a system with a minimum of restrictions on the performance characteristics of the apparatus connected to the system
8) To define a system that permits asynchronous communication over a wide range of data rates
9) To define a system that, of itself, may be relatively low cost and permits the interconnection of low cost devices
10) To define a system that is easy to use

1.3 Definitions

The following definitions apply for the purpose of this standard.

This section contains only general definitions. Detailed definitions are given in further sections as appropriate.

1.3.1 General System Terms

compatibility.: The degree to which devices may be interconnected and used, without modification, when designed as defined throughout this standard (for example, mechanical, electrical, or functional).

handshake cycle: The process whereby digital signals effect the transfer of each data byte across the interface by means of an interlocked sequence of status and control signals. Interlocked denotes a fixed sequence of events in which one event in the sequence must occur before the next event may occur.

interface: A common boundary between a considered system and another system, or between parts of a system, through which information is conveyed.

interface system: The device-independent mechanical, electrical, and functional elements of an interface necessary to effect communication among a set of devices. Cables, connector, driver and receiver circuits, signal line descriptions, timing and control conventions, and functional logic circuits are typical interface system elements.

local control: A method whereby a device is programmable by means of its local (front or rear panel) controls in order to enable the device to perform different tasks. (Also referred to as manual control.)

programmable: That characteristic of a device that makes it capable of accepting data to alter the state of its internal circuitry to perform a specific task(s).

remote control: A method whereby a device is programmable via its electrical interface connection in order to enable the device to perform different tasks.

system: A set of interconnected elements constituted to achieve a given objective by performing a specified function.

1.3.2 Units Connected Via the Interface System.

programmable measuring apparatus: A measuring apparatus that performs specified operations on command from the system and may transmit the results of the measurement(s) to the system.

terminal unit: An apparatus that terminates the considered interface system and by means which a connection (and translation, if required) is made between the considered interface system and another external interface system.

1.3.3 Signals and Paths

bidirectional bus: A bus used by any individual device for two-way transmission of messages, that is, both input and output.

bit-parallel: Refers to a set of concurrent data bits present on a like number of signal lines used to carry information. Bit-parallel data bits may be acted upon concurrently as a group (byte) or independently as individual data bits.

bus: A signal line or a set of signal lines used by an interface system to which a number of devices are connected and over which messages are carried.

byte: A group of adjacent binary digits operated on as a unit and usually shorter than a computer word (frequently connotes a group of eight bits)

byte serial: A sequence of bit-parallel data bytes used to carry information over a common bus.

high state: The relatively more positive signal level used to assert a specific message content associated with one of two binary logic states.

low state: The relatively less positive signal level used to assert a specific message content associated with one of two binary logic states.

signal: The physical representation of information.

NOTE — For the purpose of this standard, this is a restricted definition of what is often called “signal” in more general terms, and is hereinafter referred to digital electrical signals only.

signal level: The magnitude of signal compared to an arbitary reference magnitude (voltage in the case of this standard).

signal line: One of a set of signal conductors in an interface system used to transfer messages among interconnected devices.

signal parameter: That parameter of an electrical quantity whose values or sequence of values convey information.

unidirectional bus: A bus used by any individual device for one-way transmission of messages only, that is, either input only or output only.

1.4 Interface System Overview

1.4.1 Interface System Objective

The overall purpose of an interface system is to provide an effective communication link over which messages are carried in an unambiguous way among a group of interconnected devices.

Messages (quantities of information) carried by an interface system belong to either of two broad categories:

1) Messages used to manage the interface system itself, hereinafter called interface messages
2) Messages used by the devices interconnected via the interface system that are carried by, but not used or processed by the interface system directly, hereinafter called device dependent messages

NOTE — The detailed specification of device dependent messages is beyond the scope of this standard.

1.4.2 Fundamental Communication Capabilities

An effective communication link requires three basic functional elements to organize and manage the flow of information to be exchanged among devices:

1) A device acting as a listener
2) A device acting as a talker
3) A device acting as a controller

In the context of the interface system described by this standard:

1) A device with the capability to listen can be addressed by an interface message to receive device dependent messages from another device connected to the interface system.
2) A device with the capability to talk can be addressed by an interface message to send device dependent messages to another device connected to the interface system.
3) A device with the capability to control can address other devices to listen or to talk. In addition, this device can send interface messages to command specified actions within other devices. A device with only this capability neither sends nor receives device dependent messages.

NOTE — The use of the word controller throughout this standard applies strictly to the management (control) of the interface system and does not imply the broad capabilities typically associated with the word in the data processing context. Further classification of the controller will be made in Section 2 to distinguish between different types of controller capabilities related to the interface system.

Listener, talker, and controller capabilities occur individually or in any combination in devices interconnected via the interface system as shown in Fig 1.

In addition to the basic listener, talker, and controller functions, the system provides interface messages to accomplish the following operations:

1) A serial poll sequence may be initiated when a device (with talker function) requires some action by the controller, by transmitting the service request message. The controller will then obtain the status byte of all possible devices in sequence to ascertain which required service.
2) The Parallel Poll function provides a device with the ability to transmit on the controller's demand one bit of status information (request service) simultaneously with several other devices. The assignment of a.data line to a particular device for the response to a parallel poll may be accomplished through interface messages.
3) The Device Clear and Device Trigger functions provide a device with the ability to be initialized or triggered, on command from the controller. This may occur simultaneously with other selected or all devices in a system.
4) The remote/local function provides a device with the ability to accept program data from the bus, local data (for example, front panel controls), or both.

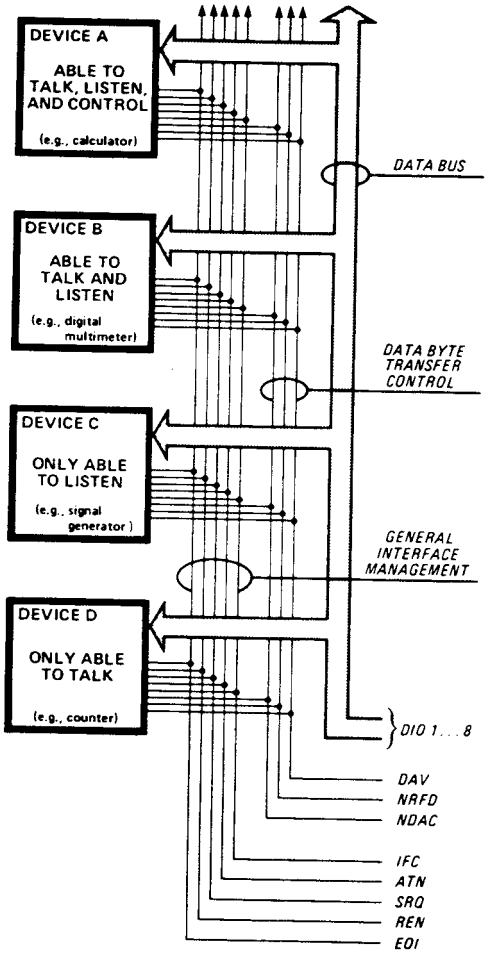


Figure 1—Interface Capabilities and Bus Structure

1.4.3 Message Paths and Bus Structure

The interface system contains a set of sixteen signal lines used to carry all information, interface messages, and device dependent messages among interconnected devices.

Messages may be coded on one or a set of signal lines as determined by the particular message content and its relationship to the interface system.

The bus structure is organized into three sets of signal lines:

1) Data bus, 8 signal lines
2) Data byte transfer control bus, 3 signal lines
3) General interface management bus, 5 paths

Figure 1 illustrates the basic communication paths.

A set of eight interface signal lines carries all 7 bit interface messages and the device dependent messages:

1) DIO1 (data input output 1)
   . .
   . .
   . .
8) DIO8 (data input output 8)

Message bytes are carried on the DIO signal lines in a bit-parallel byte-serial form, asynchronously, and generally in bidirectional manner.

NOTE — A message may be carried on an individual DIO signal line when required.

A set of three interface signal lines is used to effect the transfer of each byte of data on the DIO signal lines from a talker or controller to one or more listeners:

1) Data Valid (DAV) is used to indicate the condition (availability and validity) of information on the DIO signal lines
2) Not Ready For Data (NRFD) is used to indicate the condition of readiness of device(s) to accept data
3) Not Data Accepted (NDAC) is used to indicate the condition of acceptance of data by device(s)

The DAV, NRFD, and NDAC signal lines operate in what is called a three-wire (interlocked) handshake process to transfer each data byte across the interface.

Five interface signal lines are used to manage an orderly flow of information across the interface:

1) Attention (ATN) is used (by a controller) to specify how data on the DIO signal lines are to be interpreted and which devices must respond to the data
2) Interface Clear (IFC) is used (by a controller) to place the interface system, portions of which are contained in all interconnected devices, in a known quiescent state
3) Service Request (SRQ) is used by a device to indicate the need for attention and to request an interruption of the current sequence of events
4) Remote Enable (REN) is used (by a controller) in conjunction with other messages, to enable or disable one or more local controls that have corresponding remote controls.
5) End or Identify (EOI) is used (by a talker) to indicate the end of a multiple byte transfer sequence or, in conjunction with ATN (by a controller), to execute a polling sequence

1.4.4 Interface System Elements

The primary elements of this interface system are:

1) Functional elements
2) Electrical elements
3) Mechanical elements

Each is described in a following section.

1.5 References

This standard shall be used in conjunction with the following publications:

[1] ANSI X3.4-1986, American National Standard Code for Information Interchange Coded Character Set — 7-Bit.[2]

[2]ANSI publications can be obtained from the Sales Department, American National Standards Institute. 1430 Broadway, New York, NY 10018

[2] IEC Publication 68, Basic Environmental Testing Procedures, Part 2, Test, (1982 ed).[3]

[3] IEC Publication 348 (1978), Safety Requirements for Electronic Measuring Apparatus.

[4] IEC Publication 359 (1971), Expression of the Functional Performance of Electronic Measuring Equipment

[5] MIL STD 202F (1986), Test Method for Electronic and Electrical Component Parts.[4]

2. Functional Specifications

2.1 Functional Partition

A device is a physical entity designed for a particular application. It may be partitioned conceptually into three major functional areas each containing unique capabilities:

1) Device functions (definition is application dependent)
2) Interface functions (definition is application independent)
3) Message coding logic

All communication to or from interface functions is defined in terms of messages and state linkages (see 2.1.3).

All messages carried on the signal lines are coded according to the coding logic defined in 2.13.

2.1.1 Device Functions

The scope, purpose, size, content, and organization of the device function area (for example, analog signal measurement capability, range, modes of operation, etc) are beyond the scope of this standard. Figure 2 illustrates the device function area B for which the designer has complete freedom to define device related capability and the interface function area A for which the designer has no freedom to define new capability beyond that specified in this standard.

2.1.2 Interface Function Concepts

2.1.2.1 Interface Functions

An interface function is the system element which provides the basic operational facility through which a device can receive, process, and send messages. A number of interface functions, each of which acts in accordance with specific protocol, are defined throughout this section of the standard. Each specific interface function may only send or receive a limited set of messages within particular classes of messages.

2.1.2.2 Interface Function State

Each of the interface functions is defined in terms of one or more groups of interconnected, mutually exclusive states.

One and only one state shall be active at any one time within a single group of interconnected, mutually exclusive states.

[3]IEC publications are available in the US from the Sales Department. American National Standards Institute, 1430 Broadway, New York, NY 10018, USA. The IEC publications are also available from the International Electrotechnical Commission, rue de Varembé, Case postale [3], [2] 1 Geneve 20, Switzerland/Suisse.
[4]MIL publications can be obtained from the Naval Publications Forms Center, 5801 Tabor Avenue, Philadelphia, PA 19120-5099.

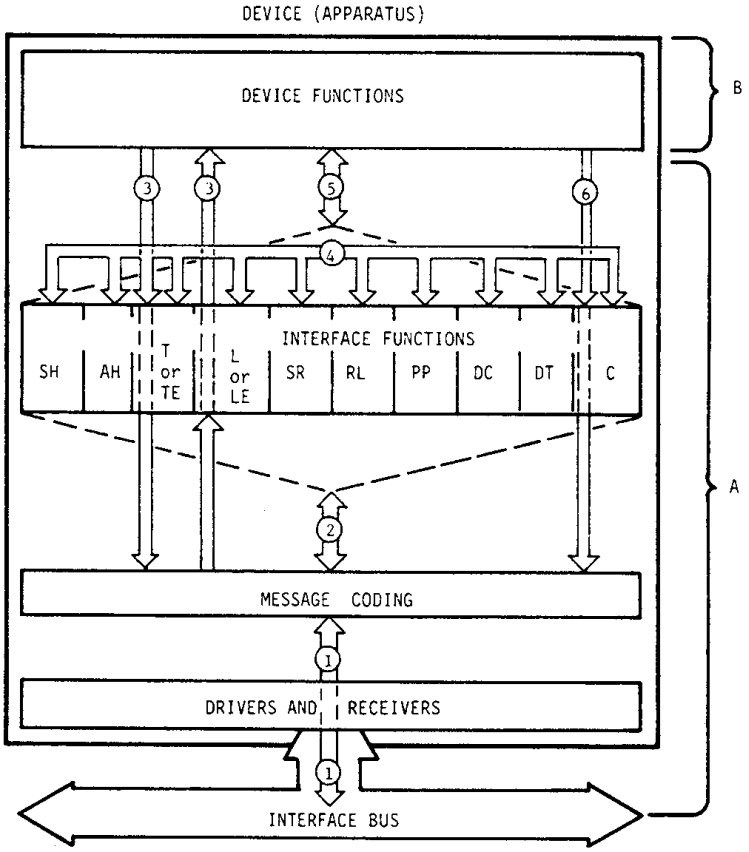
For each state of an interface function definitions are given for the following:

1) Messages that may or must be sent over the interface while that state is active
2) Conditions under which the function must leave that state and enter one of the other states in its group

These messages and conditions define the processing capability of the state.

2.1.2.3 Interface Function Repertoire

The designer is given the choice to select the particular set of interface functions necessary to fit the particular device application area. Figure 2 and Table 1 identify the available interface functions.



A = Capability defined by this standard
B = Capability defined by the designer
1 = Interface bus signal lines
2 = Remote interface messages to and from interface functions
3 = Device dependent messages to and from device functions
4 = State linkages between interface functions
5 = Local messages between device functions and interface functions (messages to interface functions are defined, messages from interface functions exist according to the designer's choice)
6 = Remote interface messages sent by device functions within a controller

Figure 2—Functional Partition Within a Device

Table 1—Interface Function Repertoire

Interface Function	Symbol	Relevant Message Paths
Source handshake	SH	1, 2, 4, 5
Acceptor handshake	AH	1, 2, 4, 5
Talker or extended talker	T or TE	1, 2, 3, 4, 5
Listener or extended listner	L or LE	1, 2, 3, 4,5
Service request	SR	1, 2, 4, 5
Remote local	RL	1, 2, 4, 5
Parallel poll	PP	1, 2, 4, 5
Device clear	DC	1, 2, 4, 5
Device trigger	DT	1, 2, 4, 5
Controller	C	1, 2, 4, 5, 6

The total processing capability of a set of interface functions (designer selected set included in a specific device) at any moment is the logic conjunction of the processing capabilities of all those states (within each individual interface function) that are active at that moment.

2.1.2.4 Interface Function Assumptions and Perspective

The state diagrams used to define the interface functions do not indicate, explicitly, or implicitly, the intended existence of specific circuits elements to achieve the logical and physical implementation of a function. For example, not all states necessarily imply the existence of a latched flip-flop or other memory element.

The state diagrams used to define the interface functions are intended to permit the use of a wide variety of logic circuit implementations (for example, random logic, sequential logic, etc).

The designer is free to combine and implement two or more interface functions with one logic design provided all the conditions for each state of each interface function as defined in this section are met.

Throughout this section of the standard, the state diagrams, written descriptions, requirements, and guidelines are written for and should be interpreted from the device perspective. Sections 5.1 and 6 will describe the interaction among devices from the system perspective.

An interface function must ignore (not respond to) any message coding not specifically defined.

A function may stay in any state for any amount of time (including zero) after exit conditions are met if this is not in conflict with specified constraints.

2.1.3 Message Concepts

2.1.3.1 Message

Each message represents a quantity of information and will be received either true or false at any specific time. All communications between an interface function and its environment is accomplished through messages sent or received.

2.1.3.2 Local Message Route and Content

Messages sent between a device function and an interface function are called local messages.

Local messages flow between device functions and interface functions; see Fig 2, message route 5.

NOTE — Certain local messages are conveyed as remote messages and vice versa.

The designer is not allowed to introduce new local messages to interface functions.

The designer is allowed to introduce a local message derived from any state of any interface function to device function(s).

Local messages sent by device functions must exist for enough time to cause the required state transitions.

2.1.3.3 Remote Message Route and Content

Messages sent via the interface between interface functions of different devices are called remote messages.

Each remote message is either an interface message or a device dependent message.

Each interface message is sent to cause a state transition within another interface function. An interface message will not be passed along to the device when received by an interface function as shown in Fig 2, message route 2.

Device dependent messages are passed between the device functions and the message coding logic via specified interface functions. These will cause no state transitions within the interface functions. Examples of device dependent messages include device programming data, device measurement data, and device status data as shown in Fig 2, message route 3.

2.1.3.4 State Linkage Route and Content

A state linkage is the logical interconnection of two interface functions where the transition to an active state of one interface function is dependent on the existence of a specified active state of another interface function as indicated in Fig 2, message route 4.

2.1.3.5 Message Coding

Message coding is the act of translating remote messages to or from interface signal line values. A message sent over a single line is called a uniline message. Two or more of these messages can be sent concurrently. A message that shares a group of signal lines with other messages, in some mutually exclusive set, is called a multiline message. Only one multiline message (message byte) can be sent at one time.

2.1.3.6 Classification of Multiline Messages

Multiline messages are interpreted as interface messages when the ATN message is true. Multiline messages are interpreted as device dependent messages when the ATN message is false. The ATN message, when true, enables the accepting and processing of these specific classes of multiline messages:

1) Universal commands (all devices)
2) Addressed commands (all devices addressed to listen)
3) Addresses (all devices)
4) Secondary addresses or commands (all devices enabled by a primary address or command)

For a list of specific multiline commands see Table 36.

2.1.3.7 Message Transfer Conventions

2.1.3.7.1 Remote Message Transfer Conventions

1) The value (true or false) of all remote false messages capable of being sent by a device shall at all times be as dictated by active states of its interface functions.
2) The interface signal line(s) used to send a message value shall be set to the levels specified by Table , Remote Message Coding.
3) Since normal interface operation allows two devices to simultaneously send opposite values of the remote messages, a technique shall be provided for resolving these conflicts. This is accomplished by implementing two types of message transfer over the interface, active transfer and passive transfer. The interface is structured so that in all conflicts between two message values, one of them will be active and the other passive. Messages shall be transferred in such a way that the active value overrides the passive value in every conflict that arises.
4) A remote message can be transferred in one of four ways:
   a) An active true value being sent is guaranteed to be the value received and the device need not allow it to be overridden
   b) A passive true value being sent is not guaranteed to be the value received, and the device must allow it to be overridden
   c) An active false value being sent is guaranteed to be the value received, and the device need not allow it to be overridden
   d) A passive false value being sent is not guaranteed to be the value received and the device must allow it to be overridden
5) Throughout the text, the terms true and false if not qualified are assumed to mean active true and active false during all discussions of remote message values sent by an interface function.
6) For two specific remote messages, DAC and RFD, only false values are defined to be sent actively. Thus, an AND operation can be considered to be performed on the interface signal lines (see 5.4).
7) For one remote message, SRQ, only true values are defined to be sent actively. Thus, an OR operation can be considered to be performed on the interface signal lines (see 5.4).
8) Only the multiline message(s) to be sent true will be specified for an interface function state since multiline messages (sent via the DIO lines) are by their nature mutually exclusive. It should be understood that all unspecified multiline messages are sent passive false while the state is active.

2.1.3.7.2 Local Message Transfer Conventions

1) The coding of local messages is beyond the scope of this standard and is left to the discretion of the device designer.
2) It is recommended that local messages qualifying transitions within any group of mutually exclusive states of an interface function be themselves mutually exclusive.

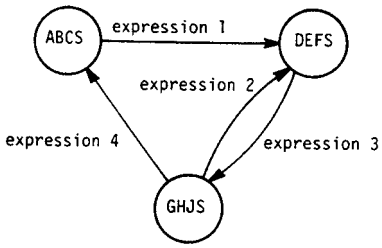
2.2 Notation Used to Specify Interface Functions

2.2.1 State Diagram Notation

Each state that an interface function can assume is represented graphically as a circle. A four-letter upper-case mnemonic always ending in an S, is used within the circle to identify the state:

All permissible transitions between states of an interface function are represented graphically by arrows between them.

Each transition is qualified by an expression whose value is either true or false. The interface function shall remain in its current state if all expressions which qualify transitions leading to other states are false. The interface function shall enter the state pointed to if, and only if, one if these expressions becomes true. The new state may be entered at any time after the expression(s) become(s) true, unless a time value is specified.



An expression consists of one or more local messages, remote messages, state linkages, or minimum time limits used in conjunction with the operators AND, OR, or NOT.

A local message to an interface function is represented by a three-letter mnemonic written in lower case; for example, rdy.

A remote message (received via the interface) is represented by a three-letter mnemonic written in upper case; for example, ATN.

A linkage from another state diagram is represented by a four-letter mnemonic enclosed in an oval; for example, (LACS). A state linkage is true if the enclosed state is currently active; otherwise, it is false.

A minimum time limit is represented by the symbol T_n. This symbol achieves a true value only after the interface has been in the state originating the corresponding transition for the time value specified. It will remain true until the state is exited. The values for these time limits are contained in Table 39.

The AND operator is represented by the symbol $\wedge$.

The OR operator is represented by the symbol $\vee$.

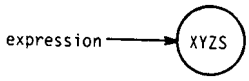
The AND operator takes precedence over the OR operator within an expression unless otherwise specified by parentheses.

The NOT operator is represented by a horizontal bar placed over the portion of the expression to be negated. The resulting negated expression has a true value if and only if the value of the expression under the bar is false.

If a transition is further qualified by a maximum time limit (within t_n), then the state pointed to shall be entered within the specified amount of time after the expression becomes true. The values for these time limits are contained in Table 39.

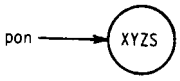
If a portion of an expression is optional in that its true value is not required for the complete expression to be true (at the designer's choice), then it is enclosed within square brackets […].

If a specific expression causes a transition to a state from all other states of the diagram, shorthand notation is used instead of all the individual transitions being drawn. An arrow without a state at its origin is used to represent this condition, and is assumed to originate in all states (for example, IFC or REN):

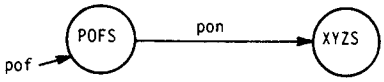


Although power-off (POFS) is a valid state of most interface functions and should normally be shown on all diagrams with a transition leading to the state to be entered at power-on time, a shorthand form is used showing the pon pseudomessage originating a transition to the first state to be entered when power is turned on:

1) Abbreviated notation used on state diagram:



2) Complete representation implied by preceding symbol:



2.2.2 Message Output Notation

The message output table included with each interface function state diagram summarizes only the remote messages allowed to be sent during each of the states of the function.

Rows of the table are used to indicate states of the interface function.

Columns of the table are used to indicate remote messages allowed to be sent during at least one state of the interface function.

Each table entry indicates the value of a message that shall be sent while a specified state is active:

1) T indicates active true
2) F indicates active false
3) (T) indicates passive true
4) (F) indicates passive false

One column in each table is allocated, if required, to the group of multiline remote messages allowed to be sent. The multiline message to be sent true during each state is placed in its corresponding table entry. False values are not shown since multiline messages are mutually exclusive. Parentheses around a multiline message name specify that it shall be sent passive rather than active true.

A separate column for device function interaction summarizes the corresponding types of messages (or resultant action) device function (s) are allowed to send or receive. Local message(s), beyond the scope of this standard, from the interface function to the device functions may be used to coordinate the appropriate action at the choice of the designer.

2.3 Source Handshake (SH) Interface Function

2.3.1 General Description

The SH interface function provides a device with the capability to guarantee the proper transfer of multiline messages. An interlocked handshake sequence between the SH function and one or more acceptor handshake (AH) functions

(each contained within separate devices) guarantees asynchronous transfer of each multiline message. The SH interface function controls the initiation of, and termination of, the transfer of a multiline message byte. This function utilizes the DAV, ready for data (RFD), and data accepted (DAC) messages to effect each message byte transfer.

2.3.2 SH Function State Diagram

The SH function shall be implemented so as to perform according to the state diagram given in Fig 3 and the state descriptions given throughout 2.3. Table 2 specifies the set of messages and states required to effect transition from one active state to another. Table 3 specifies set of messages and states required to effect transition from one active state to another. Table 3 specifies the messages that shall be sent and the device function interaction required while each state is active.

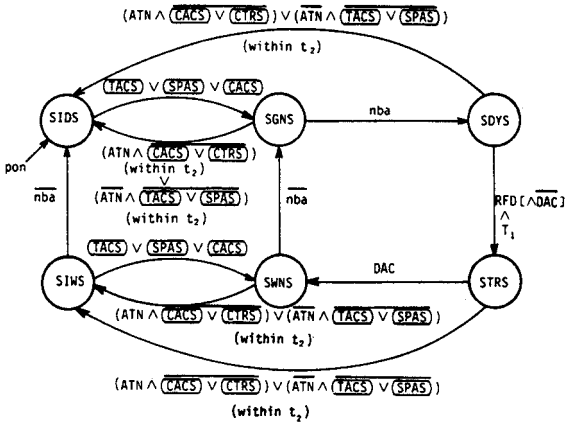


Figure 3—SH State Diagram

Table 2—SH Mnemonics

Messages		Interface States	
pon	= power on	SIDS	= source idle state
nba	= new byte available	SGNS	= source generate state
ATN	= attention	SDYS	= source delay state
RFD	= ready for data	STRS	= source transfer state
DAC	= data accepted	SWNS	= source wait for new cycle state
		SIWS	= source idle wait state
		(TACS)	= talker active state (T function)
		(SPAS)	= serial poll active state (T function)
		(CACS)	= controller active state (C function)
		(CTRS)	= controller transfer state (C function)

Table 3—SH Message Outputs

SH State	Remote Message Sent DAV	Device Function (DF) Interaction
SIDS	(F)	DF can change remote multiline messages
SGNS	F	DF can change remote multiline messages
SDYS	F	DAB, EOS multiline, and END messages shall not change
STRS	T	DAB, EOS multiline, and END messages shall not change
SWNS	T or F	DF requested to change multiline messages
SIWS	(F)	DF requested to change multiline messages

2.3.3 SH Function State Descriptions

2.3.3.1 Source Idle State (SIDS)

In SIDS the SH interface function is not engaged in the handshake cycle and does not have a new message byte available. The SH function powers on in SIDS.

In SIDS the DAV message shall be sent passive false.

The SH function shall exit SIDS and enter the source generate state (SGNS) if:

1) The talker active state (TACS) is active
2) Or the serial poll active state (SPAS) is active
3) Or the controller active state (CACS) is active

2.3.3.2 Source Generate State (SGNS)

In SGNS the device is generating a new message byte and the function is waiting for the new byte to become available

In SGNS the SH function shall send the DAV message false. In this state the device may change the multiline message being sent via the talker or controller interface function while in TACS or CACS or SPAS.

The SH function shall exit SGNS and enter:

1) The source delay state (SDYS) if the new byte available (nba) message is true
2) The SIDS within t_2 if either:
   a) The ATN message is true and neither CACS nor CTRS is active
   b) Or the ATN message is false and neither TACS nor SPAS is active

2.3.3.3 Source Delay State (SDYS)

In SDYS the SH function is waiting for a message byte to settle on the interface signal lines after the change during SGNS and for all the acceptor functions to indicate their readiness to accept the message byte. In SDYS the SH

function shall send the DAV message false. In this state the device shall not change the multiline message being sent. The SH function shall exit SDYS and enter:

1) The source transfer state (STRS) only after T_1, if the RFD message is true and if optionally, the DAC message is false.
2) The SIDS within t_2 if either:
   a) The ATN message is true and neither CACS nor CTRS is active
   b) Or the ATN message is false and neither TACS nor SPAS is active

2.3.3.4 Source Transfer State (STRS)

In STRS the SH function indicates to the AH function that it is continuously sending a valid message byte.

In STRS the SH function shall send the DAV message true. In this state the device shall not change either the multiline message or the END message (if used) being sent.

The SH function shall exit STRS and enter:

1) The source idle wait state (SIWS) within t_2 if either:
   a) The ATN message is true and neither CACS or CTRS is active
   b) Or the ATN message is false and neither TACS nor SPAS is active.
      NOTE — This implies an asynchronous interrupt (see 2.1.2.5).
2) The source wait for new cycle state (SWNS) if the DAC message is true

2.3.3.5 Source Wait for New Cycle State (SWNS)

In SWNS the SH function is waiting for the device to start a new message generation cycle.

In SWNS the SH function may send the DAV message true or false. In this state the device may change the multiline message being sent.

The SH function shall exit SWNS and enter:

1) The SGNS if the nba message is false
2) The SIWS within t_2 if either:
   a) The ATN message is true and neither CACS nor CTRS is active
   b) Or the ATN message is false and neither TACS nor SPAS is active.

2.3.3.6 Source Idle Wait State (SIWS)

In SIWS the SH function is not active in the external message byte transfer process but is active in the internal process of waiting for the device to start a new message generation cycle. This SIWS allows a sequence of message byte transfers to be interrupted without loss of data over the interface while at the same time the device may continue to prepare for the new (next) message byte generation cycle.

In SIWS the DAV message shall be sent passive false.

The SH function shall exist SIWS and enter:

1) The SIDS if the nba message is false
2) The SWNS if either:
   a) The TACS is active
   b) Or the SPAS is active
   c) Or the CACS is active

2.3.4 SH Function Allowable Subsets

The only allowable subsets to the SH function shall be those listed in Table 4.

Table 4—Allowable Subsets to SH Function

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
SH0	no capability	all	none	none
SH1	complete capability	none	none	T1-T8, TE1-TE8, or C5-C28

2.3.5 Additional SH Function Requirements and Guidelines

The nba true message indicates the device has generated a (new) message byte and made it available on the interface signal lines.

The nba message shall become true only in SIDS or SGNS. The nba message may become false in any other SH states.

The expression for interruption $(ATN \wedge \overline{(CACS)} \vee \overline{(CTRS)} \vee (\overline{ATN} \wedge \overline{(TACS)} \vee \overline{(SPAS)})$ may be substituted by $\overline{(TACS)} \wedge \overline{(SPAS)} \wedge \overline{(CACS)} \wedge \overline{(CTRS)}$ if the transition of the latter expression can be effected within t_2 after the change of ATN.

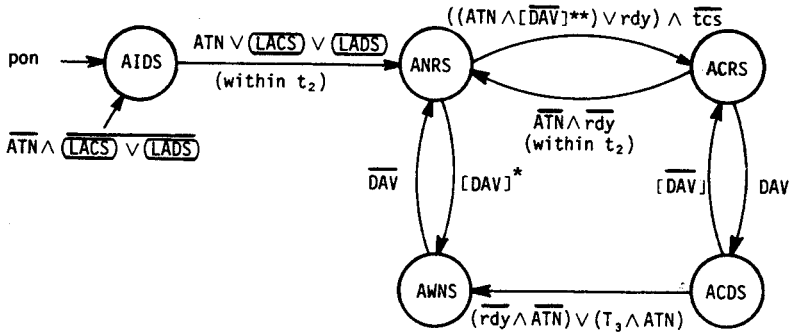
2.4 Acceptor Handshake (AH) Interface Function

2.4.1 General Description

The AH function provides a device with the capability to guarantee proper reception of remote multiline messages. An interlocked handshake sequence between an SH function and one or more AH functions (each contained within separate devices) guarantees asynchronous transfer of each message byte. An AH function may delay either the initiation of, or termination of a multiline message transfer until prepared to continue with the transfer process. The All function utilizes the DAV, RFD, and DAC messages to effect each message byte transfer.

2.4.2 AH Function State Diagram

The AH interface function shall be implemented so as to perform according to the state diagram given in Fig 4 and the state descriptions given throughout 2.4. Table 5 specifies the set of messages and states required to effect transition from one active state to another. Table 6 specifies the messages that shall be sent and the device function interaction required while each state is active.



*This transition will never occur under normal interface operation; however, it may be implemented to simplify the interface function design.

**The need for the optional DAV term arises only for devices exiting AIDS and then only under take control synchronous (tcs) conditions (see 2.12.3.7). A minor revision to the controller function is under consideration that eliminates the need for the [DAV] term. This term is retained for historical and compatibility reasons.

The addition of [DAV] prevents a data byte from being misinterpreted as an interface message (due to momentary coincidence of DAV and ATN) when ANRS is entered from AIDS. If ANRS has been active for more than 1.5 μs, it can be assumed DAV is false.

Figure 4—AH State Diagram

Table 5—AH Mnemonics

Messages		Interface States	
pon	= power on	AIDS	= acceptor idle state
rdy	= ready for next message	ANRS	= acceptor not ready state
tcs	= take control synchronously*	ACRS	= acceptor ready state
ATN	= attention	ACDS	= accept data state
DAV	= data valid	AWNS	= acceptor wait for new cycle state
		(LADS)	= listener addressed state (L function)
		(LACS)	= listener active state (L function)

*See the first paragraph of Section 2.12.3.7

Table 6—AH Message Outputs

	Remote Message Sent		
AH State	RFD	DAC	Device Function (DF) Interaction
AIDS	(T)	(T)	DF cannot receive remote multiline or END messages
ANRS	F	F	DF cannot receive remote multiline or END messages
ACRS	(T)	F	DF cannot receive remote multiline or END messages
AWNS	F	(T)	DF cannot receive remote multiline or END messages
ACDS	F	F	DF can receive remote multiline or END messages if LACS is active

2.4.3 AH Function State Descriptions

2.4.3.1 Acceptor Idle State (AIDS)

In AIDS the AH function is inactive and not engaged in the handshake cycle. The AH function powers on in AIDS.

In AIDS the RFD and DAC messages shall be sent passive true.

The AH function shall exit AIDS and enter the acceptor not ready state (ANRS) within t_2 if either:

1) The ATN message is true
2) Or LACS is active
3) Or LADS is active

2.4.3.2 Acceptor Not Ready State (ANRS)

In ANRS the AH function indicates to the interface it has not yet prepared internally to continue with the handshake cycle.

In ANRS the RFD and DAC messages shall be sent false.

The AH Function shall exit ANRS and enter:

1) The ACRS if the take control synchronously (tcs) message is false (see the first paragraph of 2.12.3.7) and either:
   a) The ATN message is true and the DAV message is false
   b) Or the ready for next message (rdy) message is true
   NOTE — Use of the DAV message is optional.
2) The AIDS if the ATN message is false and neither:
   a) The LADS is active
   b) Nor LACS is active
3) The AWNS if, optionally, the DAV message is true (note that this transition will never occur under normal interface operation)

2.4.3.3 Acceptor Ready State (ACRS)

In ACRS the AH function indicates to the interface that it is prepared to receive multiline messages.

In ACRS the DAC message shah be sent false and the RFD message shall be sent passive true.

The AH function shall exit ACRS and enter:

1) The accept data state (ACDS) if the DAV message is true
2) The AIDS if the ATN message if false and neither:
   a) The LADS is active
   b) Nor LACS is active
3) The ANRS within t_2 if both the ATN and the rdy messages are false

2.4.3.4 Accept Data State (ACDS)

In ACDS the AH function indicates to the SH function that it shall maintain a valid message byte. This is the only, state in which multiline messages on the DIO signal lines are valid. The ACDS indicates to the interface functions that an interface message is present and valid if the ATN message is true. The ACDS indicates to the device functions that a device dependent message is present and valid if LACS is active.

In ACDS the DAC and RFD messages shall be sent false.

The AH function shall exit the ACDS and messages enter:

1) The acceptor wait for new cycle State (AWNS) if either:
   a) The ATN message is true and a period of T_3 has elapsed
   b) Or the ATN and rdy messages are both false
2) The AIDS if the ATN message is false and neither:
   a) The LADS is active
   b) Nor LACS is active
3) The ACRS if, optionally the DAV message is false (note that this transition can occur only when the controller takes control asynchronously)

2.4.3.5 Acceptor Wait for New Cycle State (AWNS)

In AWNS the AH function indicates that it has received a multiline message byte.

In AWNS the RFD message shall be sent false and the DAC message shall be sent passive true.

The AH function shall exit the AWNS and enter:

1) The ANRS if DAV is false
2) The AIDS if the ATN message is false and neither:
   a) The LADS is active
   b) Nor LACS is active

2.4.4 Acceptor Function Allowable Subsets

The only allowable subsets to the AH function shall be those listed in Table 7.

Table 7—Allowable Subsets to AH Function

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
AH0	no capability	all	none	none
AH1	complete capability	none	none	none

2.4.5 Additional AH Function Requirements and Guidelines

The local message rdy shah not become false during the ACRS. The transition from ACRS to ANRS shall occur only at the time ATN becomes false.

The RFD message received by an SH function is the logical AND of all the RFD messages sent by all the active AH functions. Similarly, the DAC message received by an SH function is the logical AND of all the DAC messages sent by all the AH functions. The way in which the composite effects of multiple AH functions interact with an SH function to perform the logical AND function via the use of the NRFD and NDAC signal lines is explained further in 5.4.

Since interface functions need be designed only so as to perform according to the state diagrams specified, it is not required that exactly the states specified are the ones which exist in an implementation. One consequence of this statement is that interface function state transitions which are qualified by interface messages can occur after the message has been received as long as the RFD message is held false until they occur. The resulting performance cannot be distinguished from the performance of the specified diagrams in which the transitions shall occur while the interface message is being received. If this type of implementation is chosen, then the AH function should remain in

ANRS even though the exit condition is true in order to hold the RFD message false (this is allowed by the last paragraph in 2.1.2.4).

In noisy environments, filter action on the incoming DAV message within a device can minimize false transitions to the ACDS state.

2.5 Talker (T) Interface Function (Includes Serial Poll Capabilities)

2.5.1 General Description

The T interface function provides a device with the capability to send device dependent data (including status data during a serial poll sequence) over the interface to other devices. This capability exists only when the T interface function is addressed to talk.

There are two alternative versions of the function: one with and one without address extension. The normal T function uses a 1 byte address, the primary talk address. The T interface function with address extension [hereinafter called a TE (extended talker) function] uses a 2 byte address, the primary and secondary talk addresses. In all other respects, the capabilities of both versions are the same.

Only one of the two alternative T functions need be implemented in a specific device.

NOTE — Both the T function and TE function are described concurrently throughout 2.5 due to the extensive similarity between these two functions.

2.5.2 T Function State Diagrams

The T function shall be implemented so as to perform according to the state diagrams given in Fig 5 and the state descriptions given throughout 2.5. Table 8 specifies the set of messages and states required to effect transition from one active state to another. Table 9 specifies the messages that shall be sent and the device function interaction required while each state is active.

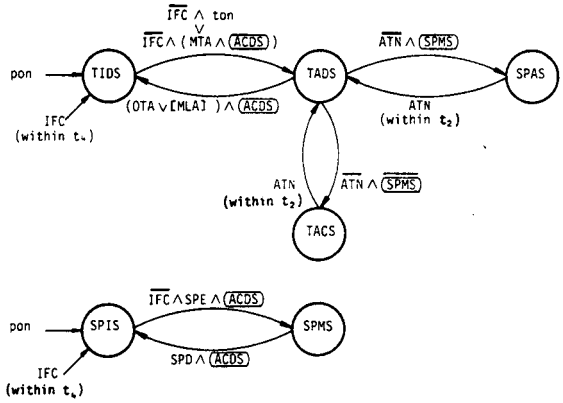


Figure 5—T State Diagram

Table 8—T Mnemonics

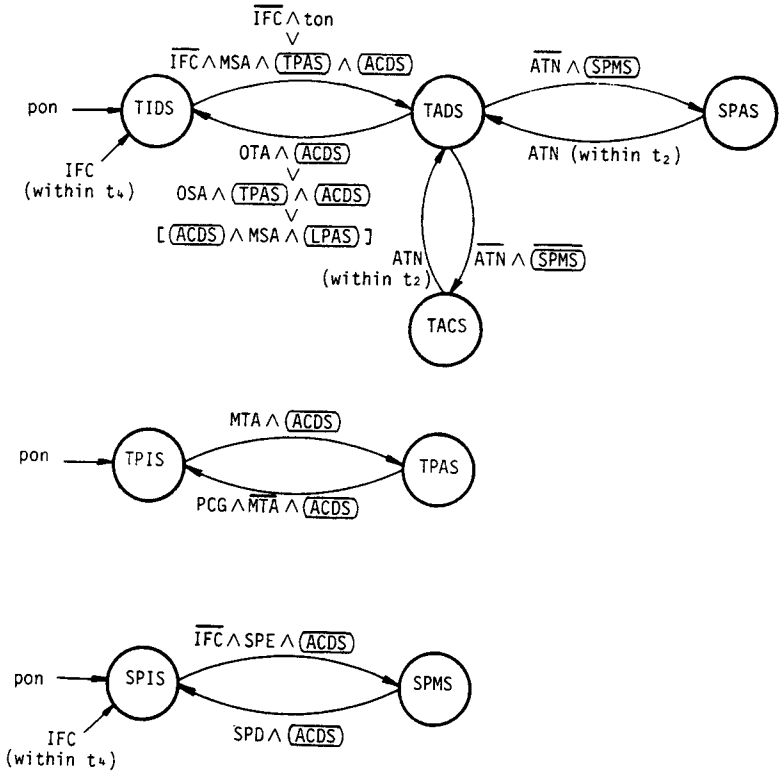
Messages		Interface States	
pon	= power on	TIDS	= talker idle state
ton	= talk only	TADS	= talker addressed state
IFC	= interface clear	TACS	= talker active state
ATN	= attention	SPAS	= serial poll active state
MTA	= my talk address	SPIS	= serial poll idle state
SPE	= serial poll enable	SPMS	= serial poll mode state
SPD	= serial poll disable	(ACDS)	= accept data state (AH function)
OTA	= other talk address		
MLA	= my listen address		

Table 9—T or TE Message Outputs

		Remote Messages Sent*			
T State	**Qualifier**	**Multiline**	**END**	**RQS†**	**Device Function (DF) Interaction**
TIDS		(NUL)	(F)	(F)	DF not allowed to send messages
TADS		(NUL)	(F)	(F)	DF not allowed to send messages
TACS		DAB‡ or EOS	T or F	(F)	DF can send DAB, EOS, or END message (if used) concurrent with DAB§
SPAS	APRS inactive	STB	F or T	F	DF can send one STB message
SPAS	APRS active	STB	F or T	T	DF can send one STB messages

*See Table , Section 2.13.
†See 2.5.3.4.
‡Messages enabled by the T function originating within the device functions
§Under SH control.

The TE function shall be implemented so as to perform according to the state diagrams given in Fig 6 and the state descriptions given throughout 2.5. Table 10 specifies the set of messages and states required to effect transition from one active state to another. Table 9 specifies the messages that shall be sent and the device function interaction required while each state is active.



NOTE: If the TE function is used together with the L function instead of the LE function, then [MSA ∧ (ACDS) ∧ (LPAS)] shall be replaced by [MLA ∧ (ACDS)].

Figure 6—TE State Diagram

Table 10—TE Mnemonics

Messages		Interface States	
pon	= power on	TIDS	= talker idle state
ton	= talk only	TADS	= talker addressed state
IFC	= interface clear	TACS	= talker active state
ATN	= attention	SPAS	= serial poll active state
MTA	= my talk address	TPIS	= talker primary idle state
OTA	= other talk address	TPAS	= talker primary addressed state
OSA	= other secondary address	SPIS	= serial poll idle state
PCG	= primary command group	SPMS	= serial poll mode state
SPE	= serial poll enable	(ACDS)	= accept data state (AH function)
SPD	= serial poll disable	(LPAS)	= listener primary addressed state (L function)
MSA	= my secondary address		

2.5.3 T Function State Descriptions

2.5.3.1 Talker Idle State (TIDS)

In TIDS neither the T function nor the TE function is engaged in sending data or status bytes. The T function or the TE function powers on in TIDS.

In TIDS the END and request service (RQS) messages shall be sent passive false and the NUL message shall be sent passive true.

The T function shall exit the TIDS when the IFC message is false and enter the talker addressed state (TADS) if either:

1) The my talk address (MTA) message is true and ACDS is active
2) Or the talk only (ton) message is true (see the last paragraph of 2.5.5)

The TE function shall exit TIDS and enter TADS if the IFC message is false and either:

1) The my secondary address (MSA) message is true, and ACDS is active, and the talker primary address state (TPAS) is active
2) Or the ton message is true

2.5.3.2 Talker Addressed State (TADS)

In TADS the T function has received its talk address and is prepared for, but not engaged in, sending data or status bytes. In TADS the TE function has received both its primary and secondary talk addresses and is prepared for, but not engaged in, sending data or status bytes.

In TADS the END and RQS messages shall be sent passive false and the NUL message shall be sent passive true.

The T function shall exit TADS and enter:

1) The talker active state (TACS) if the ATN message is false and the serial poll mode state (SPMS) is inactive
2) The serial poll active state (SPAS) if the ATN message is false and SPMS is active
3) The TIDS if either:
   a) The other talk address (OTA) message is true and ACDS is active
   b) Or the MLA message is true and ACDS is active
   c) Or within t_4 if the IFC message is true

NOTE — Use of the expression containing the MLA message is optional.

The TE function shall exit TADS and enter:

1) The TACS if the ATN message is false and SPMS is inactive
2) The SPAS if the ATN message is false and SPMS is active
3) The TIDS if either:
   a) The OTA message is true and ACDS is active.
   b) Or the other secondary address (OSA) message is true and TPAS and ACDS are active
   c) Or the MSA message is true and both the listener primary addressed state (LPAS) and ACDS are active
   d) Or within t_4 if the IFC message is true

NOTE — Use of the expression containing the MSA message is optional.

2.5.3.3 Talker Active State (TACS)

In TACS the T function, or the TE function, enables the transfer of the data byte (DAB) message and END, if used, from the device function to the interface signal lines. The message content is determined solely by the device function(s). The SH function determines when the device function(s) may change the message content of DAB (and END if used).

During TACS the DAB or end of string (EOS) and END messages may be sent by the device functions. The RQS message shall be sent passive false.

NOTE — The coding and format of the data is, in general, device dependent and beyond the scope of this standard.

The T function or the TE function shall exit TACS and enter:

1) The TADS within t_2 if the ATN message is true
2) The TIDS within t_4 if the IFC message is true

2.5.3.4 Serial Poll Active State (SPAS)

In SPAS the T function or the TE function enables the transfer of a single status message from the device function to the interface signal lines using the SH function to control the transfer of the status byte that contains both the RQS and STB messages.

Although a controller needs only one byte for the STB and RQS messages from a device it is allowable for the device to repeat this combined message byte if the controller does not assert ATN after the first transfer. In this case the content of the STB message may change between subsequent transfers although the RQS message is held unaltered by the SR function.

During SPAS, whether APRS state is active or inactive, the END message shall be sent either true or false. The RQS message shall be sent true if APRS is active, or false if APRS is inactive. In addition, the STB message shall be sent by the device function(s).

NOTE — The APRS is contained in the SR interface function.

The T function or the TE function shall exit SPAS and enter:

1) The TADS within t_2 if the ATN message is true
2) The TIDS within t_4 if the IFC message is true

2.5.3.5 Serial Poll Idle State (SPIS)

In SPIS the T function or the TE function is not enabled to participate in a serial poll. The T or TE function powers on in SPIS.

The SPIS does not provide a remote message sending capability.

The T function or the TE function shall exit SPIS and enter SPMS if the serial poll enable (SPE) message is true and ACDS is active and the IFC message is false.

2.5.3.6 Serial Poll Mode State (SPMS)

In SPMS the T function or the TE function is enabled to participate in a serial poll.

The SPMS does not provide a remote message sending capability.

The T function or the TE function shall exit SPMS and enter SPIS if either:

1) The serial poll disable (SPD) message is true and the ACDS is active
2) Or within t_4 if the IFC message is true

2.5.3.7 Talker Primary Idle State (TPIS)

In TPIS the TE function is able to recognize its primary address and not able to respond to its secondary address. The TE function powers on in TPIS.

The TPIS does not provide a remote message sending capability.

The TE function shall exit TPIS and enter TPAS if the MTA message is true and ACDS is active.

2.5.3.8 Talker Primary Addressed State (TPAS)

In TPAS the TE function is able to recognize and respond to its secondary address.

The TPAS does not provide a remote message sending capability.

The TE function shall exit TPAS and enter TPIS if the primary command group (PCG) message is true, the MTA message is false, and ACDS is active.

2.5.4 T Function and TE Function Allowable Subsets

The only allowable subsets to the T and TE interface functions shall be those listed in Tables 11 and 12.

2.5.5 Additional T and TE Interface Function Requirements and Guidelines

Each device which includes a T function or TE function shall provide a means by which the talk address (or secondary address) which it recognizes as MTA (or MSA) can be changed in the field by the user of the device.

The interruption of device sending data by transitions in and out of TACS should not adversely affect the format of the output data. It is recommended that a device returning to TACS should continue with the output data string at the point of interruption.

Each device that includes the ton message shall be provided with a local means to generate the talk only function. It is intended that the ton message be used in a system with no C interface function capability.

2.6 Listener (L) Interface Function

2.6.1 General Description

The L interface function provides a device with the capability to receive device dependent data (including status data) over the interface from other devices. This capability exists only when the function is addressed to listen.

There are two alternative versions of the function; one with and one without address extension. The normal L function uses a 1 byte address, the primary listen address. The L function with address extension [hereinafter called an extended listener (LE) function] uses a 2 byte address, the primary and secondary listen addresses. In all other respects, the capabilities of both versions are the same.

Table 11—Allowable Subsets to T Interface Function

Identification	Description				States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities						
	Basic Talker	Serial Poll	Talk Only Mode	Unaddress If MLA			
T0	N	N	N	N	all	none	none
T1	Y	Y	Y	N	none	omit [MLA ∧ (ACDS)]	SH1 and AH1
T2	Y	Y	N	N	none	omit [MLA ∧ (ACDS)] ton always false	SH1 and AH1
T3	Y	N	Y	N	SPIS, SPMS, SPAS	omit [MLA ∧ (ACDS)]	SH1 and AH1
T4	Y	N	N	N	SPIS, SPMS, SPAS	omit [MLA ∧ (ACDS)] ton always false	SH1 and AH1
T5	Y	Y	Y	Y	none	include [MLA ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
T6	Y	Y	N	Y	none	include [MLA ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4
T7	Y	N	Y	Y	SPIS, SPMS, SPAS	include [MLA ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
T8	Y	N	N	Y	SPIS, SPMS, SPAS	include [MLA ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4

Table 12—Allowable Subsets to TE Interface Function

Identification	Description				States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities						
	Basic Extended Talker	Serial Poll	Talk Only Mode	Unaddress If MSA ∧ (LPAS)			
TE0	N	N	N	N	all	none	none
TE1	Y	Y	Y	N	none	omit [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and AH1
TE2	Y	Y	N	N	none	omit [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and AH1
TE3	Y	N	Y	N	SPIS, SPMS, SPAS	omit [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and AH1
TE4	Y	N	N	N	SPIS, SPMS, SPAS	omit [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and AH1
TE5	Y	Y	Y	Y	none	include [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
TE6	Y	Y	N	Y	none	include [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4
TE7	Y	N	Y	Y	SPIS, SPMS, SPAS	include [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
TE8	Y	N	N	Y	SPIS, SPMS, SPAS	include [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4

Only one of the two alternative L functions need be implemented in a specific device.

NOTE — Both the L function and LE function are described concurrently throughout 2.6 due to the extensive similarity between these two functions.

2.6.2 L Function State Diagram

The L function shall be implemented so as to perform according to the state diagram given in Fig 7 and the state descriptions given throughout 2.6. Table 13 specifies the set of messages and states required to effect transition from one active state to another. Table 14 describes the device function interaction required while each state is active.

The LE function shall be implemented so as to perform according to the state diagram in Fig 8 and the state descriptions given throughout 2.6. Table 15 specifies the set of messages and states required to effect transition from one active state to another. Table 14 describes the device function interaction required while each state is active.

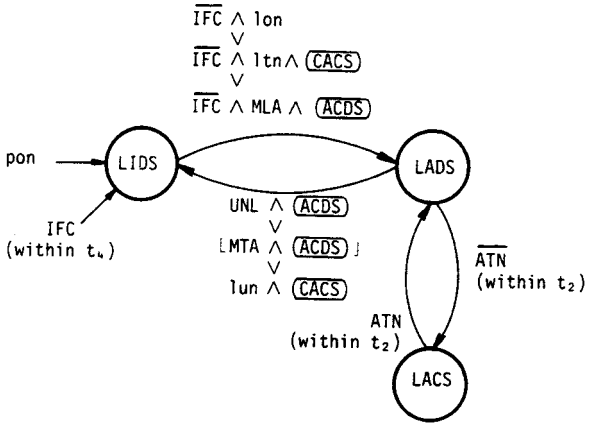


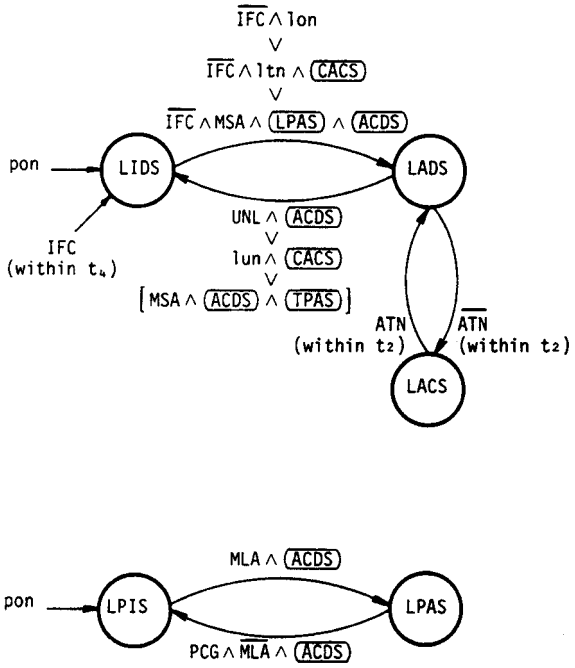
Figure 7—L State Diagram

Table 13—L Mnemonics

Messages		Interface States	
pon	= power on	LIDS	= listener idle state
ltn	= listen	LADS	= listener addressed state
lun	= local unlisten	LACS	= listener active state
lon	= listen only		
IFC	= interface clear	(ACDS)	= accept data state (AH function)
ATN	= attention	(CACS)	= controller active state (C function)
UNL	= unlisten		
MLA	= my listen address		
MTA	= my talk address		

Table 14—L or LE Message Outputs

L or LE State	Remote Messages Sent	Device Function (DF) Interaction
LIDS	none	DF not allowed to receive messages
LADS	none	DF not allowed to receive messages
LACS	none	DF can receive one device dependent message byte each time ACDS is active



NOTE: If the LE function is used together with the T function, then [MSA ∧ (ACDS) ∧ (TPAS)] shall be replaced by [MTA ∧ (ACDS)].

Figure 8—LE State Diagram

Table 15—LE Mnemonics

Messages		Interface States	
pon	= power on	LIDS	= listener idle state
ltn	= listen	LACS	= listener active state
lun	= local unlisten	LADS	= listener addressed state
lon	= listen only	LPIS	= listener primary idle state
IFC	= interface clear	LPAS	= listener primary addressed state
ATN	= attention	(ACDS)	= accept data state (AH function)
UNL	= unlisten	(CACS)	= controller active state (C function)
MLA	= my listen address	(TPAS)	= talker primary addressed state (T function)
PCG	= primary command group		
MSA	= my secondary address		

2.6.3 L Function State Descriptions

2.6.3.1 Listener Idle State (LIDS)

LIDS neither the L function nor the LE function is engaged in the transfer of device dependent messages.

The L or LE function powers on in the LIDS state.The LIDS does not provide a remote message sending capability.

The L function shall exit LIDS and enter the listener addressed state (LADS) if the IFC message is false and either:

1) The my listen address (MLA) message is true and ACDS is active
2) Or the listen only (lon) message is true (see 2.6.5)
3) Or the listen (ltn) message is true and CACS is active

The LE function shall exit LIDS and enter LADS if the IFC message is false and either:

1) The my secondary address (MSA) message is true and the ACDS state is active, and the listener primary addressed state (LPAS) is active
2) Or the Ion message is true
3) Or the ltn message is true and CACS is active

2.6.3.2 Listener Addressed State (LADS)

In LADS the L function has received its listen address and is prepared for, but not engaged in, the transfer of device dependent messages. In LADS the LE function has received both its primary and secondary listen addresses and is prepared for, but not engaged in, the transfer of device dependent messages.

The LADS does not provide a remote message sending capability.

The L function shall exit LADS and enter

1) The listener active state (LACS) within t_2 if the ATN message is false
2) The LIDS if either:
   a) The unlisten (UNL) message is true and ACDS is active
   b) Or the local unlisten (lun) message is true and CACS is active
   c) Or the MTA message is true and ACDS is active
   d) Or within t_4 if the IFC message is true

NOTE — Use of the expression containing the MTA message is optional.

The LE function shall exit LADS and enter:

1) The LACS within t_2 if the ATN message is false
2) The LIDS if either:
   a) The UNL message is true and ACDS is active
   b) Or the lun message is true and CACS is active
   c) Or the MSA message is true and TPAS and ACDS are active
   d) Or within t_4 if the IFC message is true

NOTE — Use of the expression containing the MSA message is optional.

2.6.3.3 Listener Active State (LACS)

In LACS the L function, or the LE function, is enabled to transfer any device dependent message (DAB, EOS, STB, END, or RQS) to the device functions as received via the interface signal lines. The AH function is used by the the device function(s) to control the message transfer.

NOTE — The coding and format of the data is, in general, device dependent and beyond the scope of this standard.

The LACS does not provide a remote message sending capability.

The L function or the LE function shall exit LACS and enter:

1) The LADS within t_2 if the ATN message is true
2) The LIDS within t_4 if the IFC message is true

2.6.3.4 Listener Primary Idle State (LPIS)

In LPIS the LE function is able to recognize its primary address and not able to respond to its secondary address. The LE function powers on in LPIS.

The LPIS does not provide a remote message sending capability.

The LE function shall exit LPIS and enter LPAS if the MLA message is true and ACDS is active.

2.6.3.5 Listener Primary Addressed State (LPAS)

In LPAS the LE function is able to recognize and respond to its secondary address.

The LPAS does not provide a remote message sending capability.

The LE function shall exit LPAS and enter LPIS if the primary command group (PCG) message is true, the MLA message is false and ACDS is active.

2.6.4 L Function and LE Function Allowable Subsets

The only allowable subsets to the L and LE interface functions shall be those listed in Tables 16 and 17.

2.6.5 Additional L or LE Requirements and Guidelines

Each device that includes an L function, (or LE function), shall provide a means by which the listen address (or secondary address), that it recognizes as MLA (or MSA), can be changed in the field by the user of the device.

Table 16—Allowable Subsets to L Interface Function

Identification	Description			States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities					
	Basic Listener	Listen Only Mode	Unaddress If MTA			
L0	N	N	N	all	none	none
L1	Y	Y	N	none	omit [MTA ∧ (ACDS)]	AH1
L2	Y	N	N	none	omit [MTA ∧ (ACDS)] lon always false	AH1
L3	Y	Y	Y	none	include [MTA ∧ (ACDS)]	AH1 and T1-T8 or TE1-TE8
L4	Y	N	Y	none	include [MTA ∧ (ACDS)] lon always false	AH1 and T1-T8 or TE1-TE8

Table 17—Allowable Subsets to LE Interface Function

Identification	Description			States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities					
	Basic Extended Listener	Listen Only Mode	Unaddress If (MSA ∧ (TPAS))*			
LE0	N	N	N	all	none	none
LE1	Y	Y	N	none	omit [MSA ∧ (TPAS) ∧ (ACDS)]	AH1
LE2	Y	N	N	none	omit [MSA ∧ (TPAS) ∧ (ACDS)] Ion always false	AH1
LE3	Y	Y	Y	none	include [MSA ∧ (TPAS) ∧ (ACDS)]	AH1 and T1-T8 or TE1-TE8
LE4	Y	N	Y	none	include [MSA ∧ (TPAS) ∧ (ACDS)] Ion always false	AH1 and T1-T8 or TE1-TE8

The interruption of a device receiving data by transitions in and out of LACS should not adversely affect the future receipt of input data It is recommended that a device returning to LACS should continue with the input data string at the point of interruption.

Each device that includes the Ion message shall be provided with a local means to generate the listen only condition. It is intended that the Ion message be used in a system with no C interface function capability.

2.7 Service Request (SR) Interface Function

2.7.1 General Description

The SR interface function provides a device with the capability to request service asynchronously from the controller in charge of the interface.

It also synchronizes the content of the RQS message of the composite status byte present during a serial poll so that the SRQ message can be removed from the interface once the message is received true by the controller in charge (see 2.12.1).

2.7.2 SR Interface Function State Diagrams

The SR interface function shall be implemented so as to perform according to the state diagram given in Fig 9 and the state descriptions given throughout 2.7. Table 18 specifies the set of messages and states required to effect transition from one active state to another. Table 19 specifies the messages that shall be sent and the device function interaction required while each state is active.

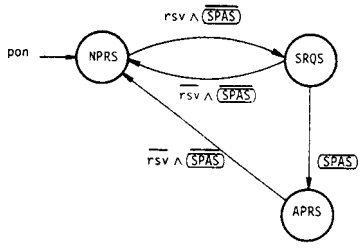


Figure 9—SR State Diagram

Table 18—SR Mnemonics

Messages		Interface States
pon = power on	NPRS	= negative poll response state
rsv = request service	SRQS	= service request state
	APRS	= affirmative poll response state
	(SPAS)	= serial poll active state (T function)

Table 19—SR Message Outputs

	Remote Message Sent	
SRQ State	SRQ	Device Function Interaction
NPRS	(F)	none
SRQS	T	none
APRS	(F)	none

2.7.3 SR State Description

2.7.3.1 Negative Poll Response State (NPRS)

In NPRS the SR function is not requesting service. The SR function powers on in NPRS.

In NPRS the SRQ message shall be sent passive false.

NOTE — The RQS message will be sent false when SPAS is active (see 2.5.3.4) and NPRS is active.

The SR function shall exit NPRS and enter SRQS at any time the request service (rsv) message is true and SPAS is not active.

2.7.3.2 Service Request State (SQRS)

In SRQS the SR function continuously indicates over the interface that it is requesting service.

In SRQS the SRQ message shall be sent true.

The SR function shall exit SRQS and enter:

1) The NPRS if the rsv message is false and SPAS is not active.
2) The affirmative poll response state (APRS) if SPAS is active

2.7.3.3 Affirmative Poll Response State (APRS)

In APRS the SR function requires service, but is not actually requesting it over the interface.

In APRS the SRQ message shall be sent passive false.

NOTE — The RQS message will be sent true by the talker when SPAS is active (see 2.5.3.4) and APRS is active.

The SR function shall exit APRS and enter NPRS at any time the rsv message is false and SPAS is not active.

2.7.4 SR Interface Function Allowable Subsets

The only allowable subsets to the SR interface function shall be those listed in Table 20.

Table 20—Allowable Subsets to SR Interface Function

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
SR0	no capability	all	none	none
SR1	complete capability	none	none	T1, T2, T5, T6, TE1, TE2, TE5 or TE6

2.7.5 Additional SR Interface Function Requirements and Guidelines

The SR function is required for each unique reason for requesting service.

If more than one reason exists, within a device, to request service then a separate SR function and corresponding rsv message shall be used for each separate reason.

Preferred practice is to logically OR multiple conditions within a device to generate a single reason for requesting service for a single SR function. If multiple SR functions are used, a single SRQ true message should be sent when requested by any of the SR functions within a device.

While the T function is in the SPAS, the RQS message shall be sent true if any of the SR functions, within a device, is in the APRS. When the SR function exits SRQS, the SRQ message is not sent again until either the rsv message goes false and reoccurs, or a different SR function in the same device enters SRQS.

The SRQ message received, via the controller (c) function, is the logical OR of the SRQ messages sent by all SR functions. The way this is performed via the use of the SRQ signal line is explained in 5.4.2.

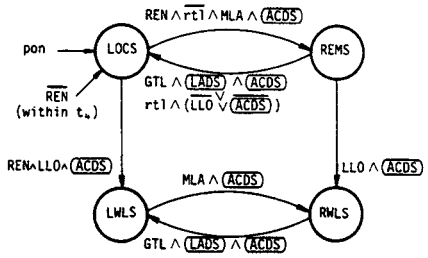
2.8 Remote Local (RL) Interface Function

2.8.1 General Description

The RL interface function provides a device with the capability to enable and disable its local controls.

2.8.2 RL Function State Diagram

The RL interface function shall be implemented so as to perform according to the state diagram given in Fig 10 and the state descriptions given throughout 2.8. Table 21 specifies the set of messages and states required to effect transition from one active state to another. Table 22 specifies the device function interaction required while each state is active.



NOTE: If the RL function is used together with the LE function, then the term MLA shall be replaced by the term (MSA ∧ (LPAS)).

Figure 10—RL State Diagram

Table 21—RL Mnemonics

Messages		Interface States	
pon	= power on	LOCS	= local state
rtl	= return to local	LWLS	= local with lockout state
REN	= remote enable	REMS	= remote state
LLO	= local lockout	RWLS	= remote with lockout state
GTL	= go to local	(ACDS)	= accept data state (AH)
MLA	= my listen address	(LADS)	= listener addressed state (L)

Table 22—RL Message Outputs

RL State	Remote Messages Sent	Device Function Interaction
LOCS	none	device is in “local control” mode
LWLS	none	device is in “local control” mode
REMS	none	device is in “remote control” mode
RWLS	none	device is in “remote control” mode

2.8.3 RL State Descriptions

2.8.3.1 Local State (LOCS)

In LOCS all local controls of the associated device functions are operative and the device may respond to corresponding device dependent messages from the interface. The RL function powers on in LOCS.

The LOCS does not provide a remote message sending capability.

The RL function shall exit LOCS if the REN message is true and enter.

1) The remote state (REMS) if the return to local (rtl) message is false and the MLA message is true and ACDS is active
2) The local with lockout state (LWLS) if the universal coded command local lockout (LLO) is true and ACDS is active

2.8.3.2 Local with Lockout State (LWLS)

In LWLS all local controls of the associated device functions are operative, and the device may respond to corresponding device dependent messages from the interface. The rtl message is ignored.

The LWLS does not provide a remote message sending capability.

The RL function shall exit LWLS and enter:

1) The remote with lockout state (RWLS) when MLA is true and ACDS is active
2) The LOCS within t_4 the REN message is false

2.8.3.3 Remote State (REMS)

In REMS some or all of the local controls (of the associated device functions) that have corresponding remote controls except those controls which send local messages to interface functions, may be inoperative.

The REMS does not provide a remote message sending capability.

The RL function shall exit REMS and enter:

1) The RWLS is the LLO message is true and ACDS is active
2) The LOCS:
   a) Within t_4 if the REN message is false
   b) Or the go to local (GTL) message is true and ACDS and LADS are active
   c) Or the rtl message is true and either the LLO message is false or ACDS is inactive

2.8.3.4 Remote with Lockout State (RWLS)

In RWLS some or all of the local controls (of the associated device functions) that have corresponding remote controls, except those controls which send local messages to interface functions, may be inoperative. The rtl message is ignored.

The RWLS does not provide a remote message sending capability,

The RL function shall exit RWLS and enter:

1) The LOCS within t_4 if the REN message is false
2) The LWLS if the GTL message is true and LADS and ACDS are active

2.8.4 RL Function Allowable Subsets

The only allowable subsets to the Remote Local Interface Function shall be those listed in Table 23.

Table 23—Allowable Subsets to RL Interface Function

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
RL0	no capability	all	none	none
RL1	complete capability	none	none	L1–L4 or LE1–LE4
RL2	no local lock out	LWLS and RWLS	rtl always false	L1–L4 or LE1–LE4

2.8.5 Additional RL Interface Function Requirements and Guidelines

The ability of a device either to send device dependent messages over the interface or to receive and utilize device dependent messages not in conflict with locally available data is independent of the state which is active within the RL function.

When either REMS or RWLS is active, the associated device shall become responsive to all subsequent input data received via the interface. Local controls shall be ignored unless specifically enabled by device dependent messages sent after entering REMS or RWLS.

It is recommended that the device not alter its state (including local controls) as a result of a transition from LOCS to REMS or from LWLS to RWLS.

Conversely, when either LOCS or LWLS becomes active, the associated device shall become responsive to future use of local controls.

After a transition from REMS or RWLS to LOCS or LWLS, it is recommended that devices, whose indicators (mechanical, positional, etc) cannot be changed by remote control, alter their local controls (and device state variables) as necessary for their front panel indicators and device state to agree.

After a transition from REMS or RWLS to LOCS or LWLS, it is recommended that devices, whose front panel indicators can be changed by remote control, alter their indicators as necessary for their front panel indication and device state to agree.

It is required that the rtl message shall not be generated permanently.

Applications that require absolute local control of a device by a local programming source (for example, a human operator) are beyond the scope of this standard.

2.9 Parallel Poll (PP) Interface Function

2.9.1 General Description

The PP interface function provides a device with the capability to present a PPR message to the controller in charge without being previously addressed to talk.

The signal lines DIO1 through DI08 are used to convey the device status bits during the parallel poll. In order for a device to respond with a PPR message, a device shall have been assigned (configured) to a single DIO line either by the controller or by a local message. This allows up to eight devices with a one-line-per-device assignment although any number of devices can be handled through sharing of DIO lines.

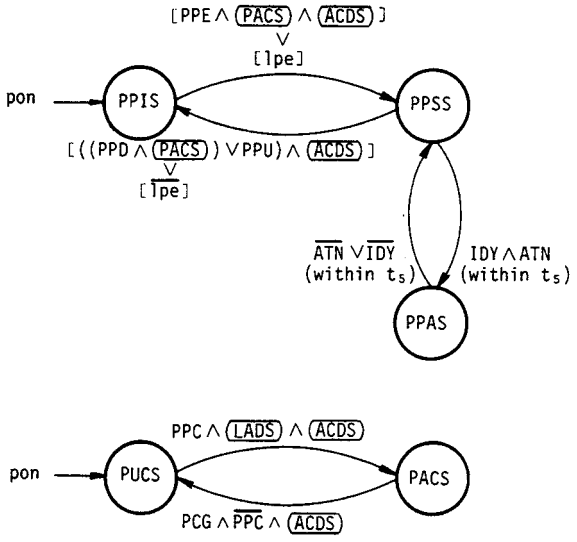
The use of the parallel poll facility within a system requires a commitment of the current interface controller to conduct a parallel poll, as required.

The parallel poll facility can be used to indicate a request for service. This capability differs from use of the SRQ message in the following ways:

1) A controller initiates a parallel poll sequence whereas any device requests the initiation of a serial poll sequence.
2) A parallel poll enables the transfer of status data from multiple devices concurrently whereas a serial poll sequentially collects status data from each device.

2.9.2 PP Function State Diagram

The PP interface function shall be implemented so as to perform according to the state diagram given in Fig 11 and the state descriptions given throughout 2.9. Table 24 specifies the set of messages and states required to effect transition from one active state to another. Table 25 specifies the messages that should be sent and the device function interaction required by the function while each state is active.



NOTE: See Table 27 for restrictions on use of the optional transitions.

Figure 11—PP State Diagram

Table 24—PP Mnemonics

Messages		Interface States	
pon	= power on	PPIS	= parallel poll idle state
ist	= individual status (Table 25)	PPSS	= parallel poll standby state
lpe	= local poll enabled	PPAS	= parallel poll active state
ATN	= attention	PUCS	= parallel poll unaddressed to configure state
IDY	= identify	PACS	= parallel poll addressed to configure state
PPE	= parallel poll enable	(ACDS)	= accept data state (AH function)
PPD	= parallel poll disable	(LADS)	= listener addressed state (L function)
PPC	= parallel poll configure		
PCG	= primary command group		
PPU	= parallel poll unconfigure		

Table 25—PP Message Outputs

PP State	Qualifier	Remote Message Sent	Device Function Interaction
		PPRn*†	
PPIS		(F)	none
PPSS		(F)	none
PPAS	ist ≡ S	T	none
PPAS	ist ≡ S	(F)	none

*See 2.9.3.3, second paragraph.
†This column refers only to the specific message assigned by the device.

2.9.3 PP State Descriptions

2.9.3.1 Parallel Poll Idle State (PPIS)

In PPIS the PP function is unable to respond to a parallel poll issued by the interface controller.

The PP function powers on in PPIS.

In PPIS all parallel poll response (PPR) messages shall be sent passive false.

The PP function shall exit PPIS and enter the parallel poll standby state (PPSS) ff either:

1) The parallel poll enable (PPE) message is true and PACS and ACDS are active
2) Or the local poll enabled (lpe) message is true

NOTE — Both the lpe and PPE transitions are optional; only one shall be used at any given time.

2.9.3.2 Parallel Poll Standby State (PPSS)

In PPSS the PP function is able to respond to parallel polls issued by the device controller whenever they occur.

In PPSS all PR messages shall be sent passive false.

The PP function shall exit the PPSS and enter:

1) The parallel poll active state (PPAS) within t_5 if the identify (IDY) and ATN messages are true (a parallel poll is in progress)
2) The PPIS if:
   a) The lpe message is false
   b) Or the parallel poll disable (PPD) message is true and PACS and ACDS are active
   c) Or the parallel poll unconfigure (PPU) message is true and ACDS is active

NOTE — Both the lpe and PPD transitions are optional; only one shall be used at any given time.

2.9.3.3 Parallel Poll Active State (PPAS)

In PPAS the PP function is responding to the parallel poll currently being conducted by the interface controller.

In PPAS one of the PPR messages shall be sent true if, and only if, the value of the individual status (ist) message is equal to the value of the sense (S) bit received as part of the most recently received PPE command. The PPR message

to be sent shall be the one specified by the three bits P1 through P3 received as part of the most recently received PPE command. Table 26 lists the PPR message specified by each of the combinations of values of P1 through P3 (see 2.9.5). All other PPR messages should be sent passive false.

Table 26—PPR Message Specified by Each of the Combination of Values PI Through P3

Bits Received with Most Recent PPE Command			PPR Message Specified
P3	P2	P1	
0	0	0	PPR1
0	0	1	PPR2
0	1	0	PPR3
0	1	1	PPR4
1	0	0	PPR5
1	0	1	PPR6
1	1	0	PPR7
1	1	1	PPR8

The PP interface function shall exit PPAS and enter PPSS within t_5 if either the IDY or ATN message is false (the parallel poll is over).

2.9.3.4 Parallel Poll Unaddressed to Configure State (PUCS)

In PUCS the PP function shall ignore any PPE or PPD messages which might be received over the interface. The PP function powers on in PUCS.

The PUCS does not provide a remote message sending capability.

The PP function shall exit PUCS and enter the parallel poll addressed to configure state (PACS) ff the PPC message is true, and LADS and ACDS are active.

2.9.3.5 Parallel Poll Addressed to Configure State (PACS)

In PACS the PP function is able to act upon PPE or PPD messages received over the interface. If a PPE message is received, the attendant bits S, P1, P2, and P3 should be saved by the function.

The PACS does not provide a remote message sending capability.

The PP function shall exit PACS and enter PUCS when the PCG message is true, the parallel poll configure (PPC) message is false and ACDS is active

2.9.4 PP Interface Function Allowable Subsets

The only allowable subsets to the parallel poll interface function shall be those listed in Table 27.

Table 27—Allowable Subsets to PP Interface Function

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
PP0	no capability	all	none	none
PP1	remote configuration	none	include [((PPD ∧ (PACS)) ∨ PPU) ∧ (ACDS)] include [PPE ∧ (PACS) ∧ (ACDS)] exclude lpe	L1-L4 or LE1-LE4
PP2	local configuration	PUCS, PACS	include lpe exclude [((PPD ∧ (PACS)) ∨ PPU) ∧ (ACDS)] exclude [PPE ∧ (PACS) ∧ (ACDS)] local messages shall be substituted for S, P1, P2, P3	none

2.9.5 Additional PP Interface Function Requirements and Guidelines

If subset PP2 is taken, field-settable local messages shall substitute for the PPE command to specify PPR message and the message sense to be used during a parallel poll.

2.10 Device Clear (DC) Interface Function

2.10.1 General Description

The DC interface function provides the device with the capability to be cleared (initialized) either individually or as part of a group of devices. The group may be either a subset or all addressed devices in one system.

2.10.2 DC Function State Diagram

The DC interface function shall be implemented so as to perform according to the state diagram given in Fig 12 and the state descriptions given throughout 2.10.

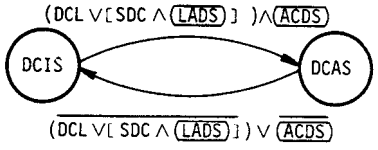


Figure 12—DC State Diagram

Table 28 specifies the set of messages and states required to effect transition from one active state to another. Table 29 specifies the device function interaction required while each state is active.

Table 28—DC Mnemonics

Messages	Interface States
DCL = device clear SDC = selected device clear	DCIS = device clear idle state DCAS = device clear active state (ACDS) = accept data state (AH function) (LADS) = listener addressed state (L function)

Table 29—DC Message Outputs

DC State	Remote Message Sent	Device Function (DF) Interaction
DCIS	none	normal device function operation
DCAS	none	DF should return to a known fixed state

2.10.3 DC Function State Descriptions

2.10.3.1 Device Clear Idle State (DCIS)

In DCIS the DC function is inactive.

The DCIS does not provide a remote message sending capability. The DC function shall exit DCIS and enter the device clear active state (DCAS) if ACDS is active, and either:

1) The device clear (DCL) message is true
2) Or the selected device clear (SDC) message is true and LADS is active

NOTE — Use of the expression containing the SDC message is optional.

2.10.3.2 Device Clear Active State (DCAS)

In DCAS the DC function sends an internal message to the device function(s) causing it (them) to be cleared. The DCAS does not provide a remote message sending capacity.

The DC function shall exit DCAS and enter the device clear idle state (DCIS) if either ACDS is inactive or neither:

1) The DCL message is true
2) Nor the SDC message is true and LADS is active

NOTE — Use of the expression containing the SDC message is optional.

2.10.4 DC Interface Function Allowable Subsets

The only allowable subsets to the DC interface function shall be those listed in Table 30.

Table 30—Allowable Subsets to DC Interface Function

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
DCØ	no capability	all	none	none
DC1	complete capability	none	none	L1-L4 or LE1-LE4
DC2	omit selective device clear	none	omit [SDC ∧ (LADS)]	AH1

2.10.5 Additional DC Function Requirements and Guidelines

The DCAS affects only device functions and does not affect other interface functions (cleared by IFC)

A device may use the DC function for any purpose consistent with its operation. Normally, use of the DC function should allow resumption of device dependent message flow to and from device functions. However, this function may be used to put any subset of the device's functions to a defined state deemed appropriate by the designer, which state the designer shall then specify.

2.11 Device Trigger (DT) Interface Function

2.11.1 General Description

The DT interface function provides the device with the capability to have its basic operation started either individually or as part of a group of devices. The group may be either a subset or all addressed devices in one system.

2.11.2 DT Function State Diagram

The DT interface function shall be implemented so as to perform according to the state descriptions given Fig 13 and the state descriptions given throughout 2.11. Table 31 specifies the set of messages and states required to effect transition from one active state to another. Table 32 specifies the device function interaction required while each state is active.

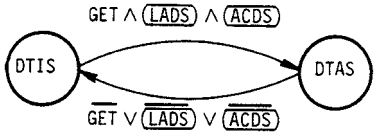


Figure 13—DT State Diagram

Table 31—DT Mnemonics

Messages	Interface States	
GET = group execute trigger	DTIS	= device trigger idle state
	DTAS	= device trigger active state
	(ACDS)	= accept data state (AH function)
	(LADS)	= listener addressed state (L function)

Table 32—DT Message Outputs

DT State	Remote Messages Sent	Device Function (DF) Interaction
DTIS	none	normal DF operation
DTAS	none	DF should start performing triggered operation

2.11.3 DT Function State Descriptions

2.11.3.1 Device Trigger Idle State (DTIS)

In DTIS the DT function is inactive. The DTIS does not provide a remote message sending capability.

The DT function shall exit DTIS and enter the device trigger active state (DTAS) if:

1) The group execute trigger (GET) message is true
2) And LADS and ACDS are active

2.11.3.2 Device Trigger Active State (DTAS)

In DTAS the DT function sends an internal message to the device function causing it to start performing its basic operation.

The DTAS does not provide a remote message sending capability.

The DT function shall exit DTAS and enter DTIS if either:

1) The GET message is false
2) Or LADS is inactive
3) Or ACDS is inactive

2.11.4 DT Interface Allowable Subsets

The only allowable subsets to the DT interface function shall be those listed in Table 33.

Table 33—Allowable Subsets to DT Interface Function

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
DT0	no capability	all	none	none
DT1	complete capability	none	none	L1-L4 or LE1-LE4

2.11.5 Additional DT Function Requirements and Guidelines

The DTAS indicates that the device (or defined portions of the device) is to start performing its designated operation.

It is recommended that the device should begin the operation immediately after DTAS becomes active.

Once a device operation has been started, it shall not respond to subsequent state transitions until the operation is complete. Only after completion of the first operation can the device start a new operation in response to the next DTAS active condition.

2.12 Controller (C) Interface Function

2.12.1 General Description

The C interface function provides a device with the capability to send device addresses, universal commands and addressed commands to other devices over the interface. It also provides the capability to conduct parallel polls to determine which devices require service.

A C interface function can exercise its capabilities only when it is sending the ATN message over the interface.

If more than one device on the interface has a C interface function, then all but one of them shall be in the controller idle state (CIDS) at any given time. The device containing the C interface function which is not in the CIDS is called the controller-in-charge (of the interface system). Protocol is provided within this standard to allow devices with a C interface function to take turns as the controller-in-charge of the interface.

The C interface function in one of the devices connected to an interface (but no more than one) can exist in the system control active state (SACS). It shall remain in this state throughout operation of the interface and so possesses the capability to send the IFC and REN messages over the interface at any time whether or not it is the controller-in-charge. This device is called the system controller (of the interface system).

2.12.2 C Function State Diagram

The C interface function shall be implemented so as to perform according to the state diagram given in Fig 14 and the state descriptions given throughout 2.12.1. Table 34 specifies the set of messages and states required to effect transition from one active state to another. Table 35 specifies the messages that shall be sent and the device function interaction required while each state is active.

2.12.3 C State Descriptions

2.12.3.1 Controller Idle State (CIDS)

In CIDS the C function relinquishes all of its interface control capabilities. The C function powers on in CIDS.

In CIDS the ATN and IDY messages shall be sent passive false and the NUL message shall be sent passive true.

The C function shall exit CIDS and enter controller addressed state (CADS) when either:

1) The take control (TCT) message (sent by the controller-in-charge) is true, the TADS and ACDS are active, and the IFC message is false
2) Or the system control interface clear active state (SIAS) is active

NOTE — The expression containing the TCT messages is optional.

2.12.3.2 Controller Addressed State (CADS)

In CADS the C function is in the process of becoming the controller-in-charge of the interface but is waiting until the current controller stops sending the ATN message

In CADS the ATN and IDY messages shall be sent passive false and the NUL message shall be sent passive true.

The C function shall exit CADS and enter:

1) The controller active state (CACS) if the ATN message is false
2) The CIDS within t_4 if the IFC message is true and SACS is not active

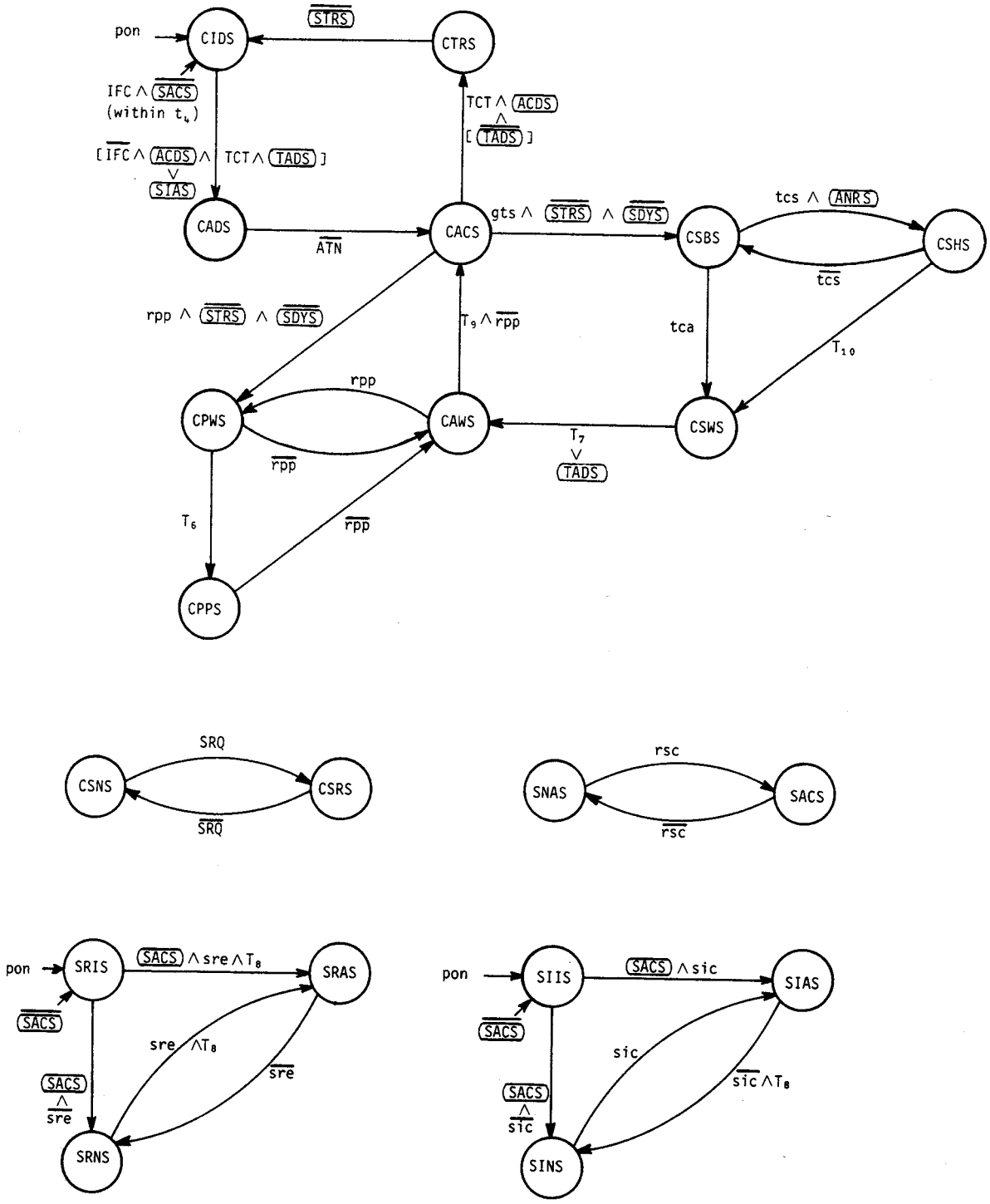


Figure 14—C State Diagram

Table 34—C Mnemonics

Messages		Interface States	
pon	= power on	CIDS	= controller idle state
rsc	= request system control	CADS	= controller addressed state
rpp	= request parallel poll	CTRS	= controller transfer state
gts	= go to standby	CACS	= controller active state
tca	= take control asynchronously	CPWS	= controller parallel poll wait state
tcs	= take control synchronously	CPPS	= controller parallel poll state
sic	= send interface clear	CSBS	= controller standby state
sre	= send remote enable	CSHS	= controller standby hold state
IFC	= interface clear	CAWS	= controller active wait state
ATN	= attention	CSWS	= controller synchronous wait state
TCT	= take control	CSRS	= controller service requested state
		CSNS	= controller service not requested state
		SNAS	= system control not active state
		SACS	= system control active state
		SRIS	= system control remote enable idle state
		SRNS	= system control remote enable not active state
		SRAS	= system control remote enable active state
		SIIS	= system control interface clear idle state
		SINS	= system control interface clear not active state
		SIAS	= system control interface clear active state
		(ACDS)	= accept data state (AH function)
		(ANRS)	= acceptor not ready state (AH function)
		(SDYS)	= source delay state (SH function)
		(STRS)	= source transfer state (SH function)
		(TADS)	= talker addressed state (T function)

Table 35—C Message Outputs[5]

	Remote Messages Sent			
C State	**ATN**	**IDY**	**Multiline**	**Device Function (DF) Interaction**
CIDS	(F)	(F)	(NUL)	DF shall not send interface messages
CADS	(F)	(F)	(NUL)	DF shall not send interface messages
CACS	T	F	*	DF can send interface messages
CPWS	T	T	(NUL)	DF shall not send interface messages
CPPS	T	T	(NUL)	DF can receive PPR messages
CSBS	F	(F)	(NUL)	DF shall not send interface messages
CSHS	F	(F)	(NUL)	DF shall not send interface messages
CSWS	T	F or (F)	(NUL)	DF shall not send interface messages
CAWS	T	F	(NUL)	DF shall not send interface messages
CTRS	T	F	TCT	DF shall finish sending TCT message

	Remote Messages Sent	
C State	IFC	Device Function (DF) Interaction
SIIS	(F)	none
SINS	F	none
SIAS	T	none

	Remote Messages Sent	
C State	REN	Device Function (DF) Interaction
SRIS	(F)	none
SRNS	F	none
SRAS	T	none

C State	Remote Messages Sent	Device Function (DF) Interaction
CSNS	none	no service requests exist
CSRS	none	DF notified of request for service

*Any coded interface message listed in Table . Although enabled by the C function, these messages originate within the device functions.

2.12.3.3 Controller Active State (CACS)

In CACS the C function enables the transfer of multiline interface messages from the device function(s) to the interface signal lines. These messages include device addresses, universal commands, or addressed commands. The SH function determines when the device function(s) may change the message content of the multiline messages being sent. However, message content is determined solely by the device function(s).

The ATN message shall be sent continuously true and the IDY message shall be sent continuously false, while CACS is active, under which conditions any of the multiline messages in Table 36 may be sent by the device functions.

[5]Message values sent are shown opposite only those states which affect them. Each major section of the table corresponds to a group of mutually exclusive states within the controller function.

The C function shall exit CACS and enter:

1) The controller transfer state (CTRS) if the TCT message is true, TADS is (optionally) inactive, and ACDS is active
2) The controller parallel poll wait state (CPWS) ff the request parallel poll (rpp) message is true, and neither SDYS nor STRS is active
3) The CIDS within t_4 if the IFC message is true and SACS is not active
4) The controller standby state (CSBS) if the gts (go to standby) message is true, and neither STRS nor SDYS is active

2.12.3.4 Controller Parallel Poll Wait State (CPWS)

In CPWS the C function is conducting aparallel poll over the interface but waiting for the DIO lines to settle.

In CPWS the ATN and IDY messages shall be sent true and the NUL message shall be sent passive true.

The C function shall exit CPWS and enter:

1) The controller parallel poll state (CPPS) after a period of T_6 has elapsed
2) The CIDS within t_4 if the IFC message is true and SACS is not active
3) The CAWS state ff the rpp message is false

Table 36—Multiline Messages

Universal Commands (multiline)	Addresses
LLO	(LAD)*
DCL	(TAD)†
SPE	UNL
SPD	
PPU	
Addressed Commands	**Secondary Commands**
GET	(SAD)‡
GTL	PPD
PPC	PPE
SDC	
TCT	

*Represents a listen address of a specific device (received as MLA).
†Represents a talk address of a specific device (received as MTA or OTA).
‡ Represents a secondary address of a specific device (received as MSA or OSA).

2.12.3.5 Controller Parallel Poll State (CPPS)

In CPPS the C function is conducting a parallel poll and actively transferring PPR message values to the device function(s) as received via the interface signal lines.

In CPPS the ATN and IDY messages shall be sent true and the NUL message shall be sent passive true.

The C function shall exit CPPS and enter:

1) The CAWS if the rrp message is false
2) The CIDS within t_4 if the IFC message is true and SACS is not active

2.12.3.6 Controller Standby State (CSBS)

In CSBS the C function is allowing two or more devices to transfer device dependent messages over the interface.

In CSBS the ATN message shall be sent false, the IDY message shall be sent passive false, and the NUL message shall be sent passive true.

The C function shall exit CSBS and enter:

1) The controller standby hold state (CSHS) if the take control synchronously (tcs) message is true and ANRS is active
2) The controller synchronous wait state (CSWS) if the take control asynchronously (tca) message is true
3) The CIDS within t_4 if the IFC message is true and SACS is not active

2.12.3.7 Controller Synchronous Wait State (CSWS)

In CSWS the C function is in the process of entering the controller active wait state (CAWS) but is waiting for a specified time T_7 to make sure that the current active talker recognizes the ATN message being sent over the interface. If this state were entered via the tcs message, the device function(s) shall continue to send it true during this state.

This causes the AH interface function to continue sending the RFD message false over the interface, holding off transfer of the next data byte.

In CSWS the ATN message shall be sent true, the IDY messages shall be sent active or passive false, and the NUL message shall be sent passive true.

The C function shall exit CSWS and enter:

1) The CAWS after a period of T_7 has elapsed or if TADS is active
2) The CIDS within t_4 if the IFC message is true and SACS is not active

2.12.3.8 Controller Active Wait State (CAWS)

In CAWS the C function is waiting for a period of T_9 before entering CACS. This wait shall occur in order to guarantee that the EOI line has settled to its proper value and that no device is responding erroneously to what appears to be a parallel poll.

In CAWS the ATN message shall be sent true, the IDY message shall be sent false and the NUL message shall be sent passive true.

The C function shall exit CAWS and enter:

1) The CACS if the rpp message is false and a period of T_9 has elapsed
2) The CPWS if the rpp message is true
3) The CIDS within t_4 if the IFC message is true and SACS is not active

2.12.3.9 Controller Transfer State (CTRS)

In CTRS the C function is sending the TCT addressed command to another device and is thus in the process of becoming idle.

In CTRS the ATN message shall be sent true, the IDY message shall be sent false, and the TCT message shall continue to be sent true.

The C function shall exit CTRS and enter CIDS when either:

1) The STRS becomes inactive
2) Or within t_4 if the IFC message is true and SACS is not active

2.12.3.10 Controller Service Requested State (CSRS)

In CSRS the C function is notifying the device function(s) via a local message that at lea.st one device on the interface is requesting service.

The CSRS does not provide a remote message sending capability.

The C function shall exit CSRS and enter the controller service not requested state (CSNS) if the SRQ message is false.

2.12.3.11 Controller Service Not Requested State (CSNS)

In CSNS the C function is notifying the device function(s) via a local message that no device on the interface is requesting service.

The CSNS does not provide a remote message sending capability.

The C function shall exit CSNS and enter CSRS if the SRQ message is true.

2.12.3.12 System Control Not Active State (SNAS)

In SNAS the C function relinquishes all of its system control capabilities. The SNAS does not provide a remote message sending capability.

The C function shall exit SNAS and enter SACS if the request system control (rsc) message is true.

2.12.3.13 System Control Active State (SACS)

In SACS the C function is allowed to exercise its system control capabilities. The SACS does not provide a remote message sending capability.

The C function shall exit SACS and enter SNAS if the rsc message is false.

2.12.3.14 System Control Interface Clear Idle State (SIIS)

In SIIS the C function has no capability to clear the interface. The C interface function powers on in SIIS.

In SIIS the IFC message shall be sent passive false.

The C function shall exit SIIS if SACS is active and enter.

1) The system control interface clear not active state (SINS) if the send interface clear (sic) message is false
2) The system control interface clear active state (SIAS) if the sic message is true

2.12.3.15 System Control Interface Clear Not Active State (SINS)

In SINS the C function is not engaged in clearing the interface.

In SINS the IFC message shall continuously be sent false.

The C function shall exit SINS and enter:

1) The SIAS if the local sic message is true
2) The SIIS if SACS is not active

2.12.3.16 System Control Interface Clear Active State (SIAS)

In SIAS the C function is engaged in clearing the interface.

All interface functions connected to the system shall respond to the IFC true message and will transfer to a known initial state.

In SIAS the IFC message shall be sent true.

The C function shall exit SIAS and enter:

1) The SINS if the sic message is false and SIAS has been active for at least a period of T_8
2) The SIIS if SACS not active

2.12.3.17 System Control Remote Enable Idle State (SRIS)

In SRIS the C function has no remote enable capability. All implementations of the C function should remain in the SRIS continuously except when used in a device capable of system controller performance. The C function powers on in the SRIS state.

In SRIS the REN message shall be sent passive false.

The C function shall exit SRIS and enter:

1) The system control remote enable not active state (SRNS) if the send remote enable (sre) message is false and SACS is active
2) The system control remote enable state (SRAS) if the sre message is true, SACS is active and SRIS has been active for at least a period of T_8

2.12.3.18 System Control Remote Enable Not Active State (SRNS)

In SRNS the C function is not engaged in enabling remote operation of other devices over the interface.

In SRNS the REN message shall be sent passive false.

The C function shall exit SRNS and enter:

1) The SRAS if the sre message is true for at least a period of T_8
2) The SRIS if SACS is not active

2.12.3.19 System Control Remote Enable Active State (SRAS)

In SRAS the C function is actively engaged in enabling remote operations of other devices over the interface.

In SRAS the REN message shall be continuously sent true.

The C function shall exit SRAS and enter:

1) The SRNS if the sre message is false
2) The SRIS if SACS is not active

2.12.3.20 Controller Standby Hold State (CSHS)

In CSHS the C function is in a hold state until such time as the DAV message is false as presented to all devices connected to the system. The CSHS state prevents false coincidence of the ATN and DAV messages, as observed by idle devices, during a tcs sequence.

In CSHS the ATN message shall be sent false, the IDY message shall be sent passive false, and the NUL message should be sent passive true.

The C function shall exit CSHS and enter.

1) The Controller Standby Wait State (CSWS) if a period of T_{10} has elapsed
2) The CSBS if the tcs message is false
3) The CIDS within t_4 if the IFC message is true and SACS is not active

2.12.4 C Interface Function Allowable Subsets

The only allowable subsets to the C interface function shall be those listed in Table 37.

2.12.5 Additional C Function Requirements and Guidelines

> **WARNING:** *Use tca with caution.*

Restriction on the use of tea: The designer shall not assume that valid data will be transferred across the interface if the tca message becomes true while a device dependent message is true.

Background: Asynchronous interruption of an active talker by a controller through the use of tea may occur at any time when a device dependent message is true. If a device dependent message is true and ATN becomes true the interrupted byte could be misinterpreted by other devices as an interface message (for example, command or address) and produce unintended state transitions.

The tcs message, if used, may change from false to true only during CSBS. It may change from true to false only during the CAWS. These restrictions guarantee that RFD is held false for the proper amount of time during a synchronous take-control operation.

A device with system controller capability should be provided with a manual means to interrupt the local rsc message if that device is to be used in multiple controller environments.

2.13 Remote Message Coding and Transfer

2.13.1 Remote Message Coding

Each remote message is sent by an interface function or received by an interface function via one or more interface signal lines. This section defines the complete set of remote messages and how they are coded and transferred on the signal lines. The coding of all remote messages sent or received by the various interface functions is specified in Table 38, Remote Message Coding.

Table 37—Allowable Subsets to Controller Interface Function

	System Controller	Send IFC and Take Charge	Send REN	Respond to SRQ	Send I.F. Messages	Receive Control	Pass Control	Pass Control to Self	Parallel Poll	Take Control Synchronously	SNAS, SACS	SHS, SIAS, SINS	SRIS, SRAS, SRNS	CSNS, CSRS	CACS, CSBS, CSHS, CSWS, CAWS	CADS	CIDS	CTRS	CPWS, CPPS	[TCT ∧ ACDS ∧ TADS]†	[TADS]‡	*tcs* not always false	C1	C2	AH1, L1-L4, or LE1-LE4	SH1	T1-T8, TE1-TE8
C0	N	N	N	N	N	N	N	N	N	N	O	O	O	O	O	O	O	O	O	O	O	O	O	–	–	–	–
C1	Y	–	–	–	–	–	–	–	–	–	R	R	–	–	–	–	–	–	–	–	–	–	R	–	–	–	–
C2	–	Y	–	–	–	–	–	–	–	–	(1)	–	–	–	–	–	–	–	–	–	–	–	–	R	–	–	–
C3	–	–	Y	–	–	–	–	–	–	–	(1),(6)	–	R	–	–	–	–	–	–	–	–	–	–	R	–	–	–
C4	–	–	–	Y	–	–	–	–	–	–	(1)	–	–	R	–	–	–	–	–	–	–	–	–	–	–	–	–
C5	–	–	–	–	Y	Y	Y	Y	Y	Y	(2),(3)	–	–	–	R	R	R	R	R	R	R	R	–	–	R	R	R
C6	–	–	–	–	Y	Y	Y	Y	Y	N	(2),(3)	–	–	–	R	R	R	R	R	R	R	O	–	–	R	R	R
C7	–	–	–	–	Y	Y	Y	Y	N	Y	(2),(3)	–	–	–	R	R	R	R	O	R	O	R	–	–	R	R	R
C8	–	–	–	–	Y	Y	Y	Y	N	N	(2),(3)	–	–	–	R	R	R	R	O	R	O	O	–	–	R	R	R
C9	–	–	–	–	Y	Y	Y	N	Y	Y	(2),(3)	–	–	–	R	R	R	R	R	O	R	R	–	–	R	R	R
C10	–	–	–	–	Y	Y	Y	N	Y	N	(2),(3)	–	–	–	R	R	R	R	O	O	R	O	–	–	R	R	R
C11	–	–	–	–	Y	Y	Y	N	N	Y	(2),(3)	–	–	–	R	R	R	R	R	R	O	R	–	–	R	R	R
C12	–	–	–	–	Y	Y	Y	N	N	N	(2),(3)	–	–	–	R	R	R	R	O	O	O	O	–	–	R	R	R
C13	–	–	–	–	Y	Y	N	Y	Y	Y	(2)	–	–	–	R	R	R	O	R	O	R	R	–	–	R	R	R
C14	–	–	–	–	Y	Y	N	Y	Y	N	(2)	–	–	–	R	R	R	O	R	O	R	O	–	–	R	R	R
C15	–	–	–	–	Y	Y	N	N	N	Y	(2)	–	–	–	R	R	R	O	O	O	R	R	–	–	R	R	R
C16	–	–	–	–	Y	Y	N	N	N	N	(2)	–	–	–	R	R	R	O	O	O	O	O	–	–	R	R	R
C17	–	–	–	–	Y	N	Y	Y	Y	Y	(2),(3),(4)	–	–	–	R	O	R	R	R	R	R	R	–	–	R	R	R
C18	–	–	–	–	Y	N	Y	Y	Y	N	(2),(3),(4)	–	–	–	R	O	R	R	R	R	R	O	–	–	R	R	R
C19	–	–	–	–	Y	N	Y	Y	N	Y	(2),(3),(4)	–	–	–	R	O	R	R	O	R	R	R	–	–	R	R	R
C20	–	–	–	–	Y	N	Y	Y	N	N	(2),(3),(4)	–	–	–	R	O	R	R	O	R	R	O	–	–	R	R	R
C21	–	–	–	–	Y	N	Y	N	Y	Y	(2),(3),(4)	–	–	–	R	O	R	R	R	R	O	R	–	–	R	R	R
C22	–	–	–	–	Y	N	Y	N	Y	N	(2),(3),(4)	–	–	–	R	O	R	R	R	R	O	O	–	–	R	R	R
C23	–	–	–	–	Y	N	Y	N	N	Y	(2),(3),(4)	–	–	–	R	O	R	R	O	R	O	R	–	–	R	R	R
C24	–	–	–	–	Y	N	Y	N	N	N	(2),(3),(4)	–	–	–	R	O	R	R	O	R	O	O	–	–	R	R	R
C25	–	–	–	–	Y	N	N	N	Y	Y	(2),(5)	–	–	–	R	O	O	O	R	O	O	R	–	–	R	R	R
C26	–	–	–	–	Y	N	N	N	Y	N	(2),(5)	–	–	–	R	O	O	O	R	O	O	O	–	–	R	R	R
C27	–	–	–	–	Y	N	N	N	N	Y	(2),(5)	–	–	–	R	O	O	O	O	O	R	O	–	–	R	R	R
C28	–	–	–	–	Y	N	N	N	N	N	(2),(5)	–	–	–	R	O	O	O	O	O	O	O	–	–	R	R	R

*Typical notation to describe a controller consists of the letter C followed by one or more of the numbers indicating the subsets selected. For example; C1, 2, 3, 4, 8.

†This is part of the CIDS to CADS transitional expression.

‡This is part of the CACS to CTRS transitional expression.

NOTES:
(1) One or more of subsets C1 through C4 may be chosen in any combination with any one of C5 through C28.
(2) Only one subset may be chosen from C5 through C28.
(3) The CTRS state must be included in devices which are to be operated in multicontroller systems.
(4) These subsets are not allowed unless C2 is included.
(5) These subsets are intended to be used in devices and systems where no control passage is possible.
(6) When a system controller asserts IFC during the time another physical device is operating as controller-in-charge, the system controller should refrain from active assertion of the source handshake and ATN until the removal of the IFC message to preclude multiple controller contention.
O = omit, R = required, hyphen = not applicable or not required, Y = yes, N = no.

2.13.2 Remote Message Coding Concepts

Messages may be coded into the logical state of one or more signal lines.

For this standard, a message derived from or sent as the logical state of only one signal line is referred to as a uniline message (for example, ATN).

For this standard, a message derived from or sent as a combination of logical states of two or more signal lines is referred to as a multiline message (for example, DCL).

A message may be defined as a logical combination (AND, OR, or NOT) of other messages (for example, OTA).

The coding of a message sent and received is the same.

2.13.3 Remote Message Transfer

A message is sent by driving one or more specified signal lines to a logical 1 or a logical 0. Lines not specified as part of the message coding shall not be driven.

A message is received by sensing one or more specified bus signal lines to determine the logical value of each signal line as either 1 or 0. Lines not specified as part of the message coding are ignored.

A uniline message value is considered valid as soon as its corresponding logic state is detected. (See Tables 3, 6, 9, 19, 25, 35 for times at which messages may be sent.)

A multiline message is valid only within the context of the SH and AH functions. A transmitted multiline message is valid while the SH function is in the source transfer state (STRS). A received multiline message is valid while the AH function is in the accept data state (ACDS).

All passive message values are transferred as 0 signal line slates. This requires only the logic OR of signal line states to be performed on the interface.

2.13.4 Remote Message Coding Table Organization and Conventions

All messages capable of being sent or received by an interface function are listed by name and mnemonic in Table .

The table correlates the message value (true or false) to the bus signal line logical value (1 or 0) and vice versa.

Each remote message entry in the table specifies both the encoding required to send the messages and the decoding required to recieved the messages.

The true value of a uniline message is specified by the assignment of a specific logical state to a signal line.

The true value of a multiline message is specified by the assignment of a unique set of logical states (1 or 0) to the corresponding set of signal lines that contain the message.

The false value of a message is any combination of logic states (1 or 0) other than the unique set that specifies the true value.

Each message entry in the table is identified by type: either uniline U or multiline M. Each message is further identified by class (1 of 7) according to the function it performs within the interface function or device function.

The logical state a bus signal line may have is specified in the table as a 0, 1, Y, or X. These represent the logic states as follows:

0 = logical zero
1 = logical one
X = don't care (for the coding of a received message)
X = shall not drive unless directed by another message (for the coding of a transmitted message)
Y = don't care (for the coding of a transmitted message)
Y = don't care (recommended for the coding of a received message)

2.13.5 Remote Message Coding Table Perspective

Table is constructed to reflect each remote message sent (or received) by each the interface function. In actual use, two or more messages as defined in the table may be sent concurrently (for example, DAB true and ATN false) by different interface functions. See Notes 9 and 10 to Table and Appendix C.13.

2.13.6 Summary Notes and Symbols for Remote Message Coding Table

Level Assignment:

0 = High state signal level
1 = Low state signal level

The coding of Table may be translated to equivalent electrical signal levels as specified in 3.2.

Symbols:

Type	U	= Uniline message
	M	= Multiline message
Class	AC	= Addressed command
	AD	= Address (talk or listen)
	DD	= Device dependent
	HS	= Handshake
	UC	= Universal command
	SE	= Secondary
	ST	= Status

2.13.7 ISO Code Representation: Message Coding Guidelines

Many devices use the ISO-7 bit code (or the equivalent code in ANSIX3.4-1986 [1] , American National Standard Code for Information Interchange) because it is convenient to both generate and interpret this code. The relationships between the ISO code and the messages (binary bit patterns) defined and described in this standard are identified in this clause.

2.13.7.1 Interface Messages

The interface system utilizes message coding as defined in Table to carry interface messages among devices when the ATN message is true. This coding may be correlated to the ISO code by relating DIO1 through DIO7 to bits 1 through 7, respectively. The ISO code does not contain the equivalent of the dedicated ATN message (bit or line).

When the interface system defined in this standard is interconnected, via a terminal unit, to other environments, then protocol beyond the scope of this standard shall be used to enable proper communication and avoid possible contradictions with other assigned meanings for the ISO code.

Table 38—Remote Message Coding

						Bus Signal Line(s) and Coding That Asserts the True Value of the Message													
			Type	Class	DIO	8	7	6	5	4	3	2	1	DDINNDRDAFAVDC	ATN	EOI	SRQ	IFC	REN
Mnemonic	Message Name																		
ACG	addressed command group		M	AC		Y	0	0	0	X	X	X	X	XXX	1	X	X	X	X
ATN	attention		U	UC		X	X	X	X	X	X	X	X	XXX	1	X	X	X	X
DAB	data byte	(Notes 1, 9)	M	DD		D	D	D	D	D	D	D	D	XXX	0	X	X	X	X
						8	7	6	5	4	3	2	1						
DAC	data accepted		U	HS		X	X	X	X	X	X	X	X	XX0	X	X	X	X	X
DAV	data valid		U	HS		X	X	X	X	X	X	X	X	1XX	X	X	X	X	X
DCL	device clear		M	UC		Y	0	0	1	0	1	0	0	XXX	1	X	X	X	X
END	end	(Note 9)	U	ST		X	X	X	X	X	X	X	X	XXX	0	1	X	X	X
EOS	end of string	(Notes 2, 9)	M	DD		E	E	E	E	E	E	E	E	XXX	0	X	X	X	X
						8	7	6	5	4	3	2	1						
GET	group execute trigger		M	AC		Y	0	0	0	1	0	0	0	XXX	1	X	X	X	X
GTL	go to local		M	AC		Y	0	0	0	0	0	0	1	XXX	1	X	X	X	X
IDY	identify		U	UC		X	X	X	X	X	X	X	X	XXX	X	1	X	X	X
IFC	interface clear		U	UC		X	X	X	X	X	X	X	X	XXX	X	X	X	1	X
LAG	listen address group		M	AD		Y	0	1	X	X	X	X	X	XXX	1	X	X	X	X
LLO	local lock out		M	UC		Y	0	0	1	0	0	0	1	XXX	1	X	X	X	X
MLA	my listen address	(Note 3)	M	AD		Y	0	1	L	L	L	L	L	XXX	1	X	X	X	X
									5	4	3	2	1						
MTA	my talk address	(Note 4)	M	AD		Y	1	0	T	T	T	T	T	XXX	1	X	X	X	X
									5	4	3	2	1						
MSA	secondary address	(Note 5)	M	SE		Y	1	1	S	S	S	S	S	XXX	1	X	X	X	X
									5	4	3	2	1						

Table 38—Remote Message Coding (Continued)

Mnemonic	Message Name		Type	Class	Bus Signal Line(s) and Coding That Asserts the True Value of the Message: DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	NDAC NRFD VDC	ATN	EOI	SRQ	IFC	REN
NUL	null byte		M	DD	0	0	0	0	0	0	0	0	XXX	X	X	X	X	X
OSA	other secondary address		M	SE	(OSA = SCG ∧ MSA)													
OTA	other talk address		M	AD	(OTA = TAG ∧ MTA)													
PCG	primary command group		M	—	(PCG = ACG ∨ UCG ∨ LAG ∨ TAG)													
PPC	parallel poll configure		M	AC	Y	0	0	0	0	1	0	1	XXX	1	X	X	X	X
PPE	parallel poll enable	(Note 6)	M	SE	Y	1	1	0	S	P3	P2	P1	XXX	1	X	X	X	X
PPD	parallel poll disable	(Note 7)	M	SE	Y	1	1	1	D4	D3	D2	D1	XXX	1	X	X	X	X
PPR 1	parallel poll response 1	(Note 10)	U	ST	X	X	X	X	X	X	X	1	XXX	1	1	X	X	X
PPR 2	parallel poll response 2	(Note 10)	U	ST	X	X	X	X	X	X	1	X	XXX	1	1	X	X	X
PPR 3	parallel poll response 3	(Note 10)	U	ST	X	X	X	X	X	1	X	X	XXX	1	1	X	X	X
PPR 4	parallel poll response 4	(Note 10)	U	ST	X	X	X	X	1	X	X	X	XXX	1	1	X	X	X
PPR 5	parallel poll response 5	(Note 10)	U	ST	X	X	X	1	X	X	X	X	XXX	1	1	X	X	X
PPR 6	parallel poll response 6	(Note 10)	U	ST	X	X	1	X	X	X	X	X	XXX	1	1	X	X	X
PPR 7	parallel poll response 7	(Note 10)	U	ST	X	1	X	X	X	X	X	X	XXX	1	1	X	X	X
PPR 8	parallel poll response 8	(Note 10)	U	ST	1	X	X	X	X	X	X	X	XXX	1	1	X	X	X
PPU	parallel poll unconfigure		M	UC	Y	0	0	1	0	1	0	1	XXX	1	X	X	X	X
REN	remote enable		U	UC	X	X	X	X	X	X	X	X	XXX	X	X	X	X	1
RFD	ready for data		U	HS	X	X	X	X	X	X	X	X	X0X	X	X	X	X	X
RQS	request service	(Notes 8, 9)	U	ST	X	1	X	X	X	X	X	X	XXX	0	X	X	X	X
SCG	secondary command group		M	SE	Y	1	1	X	X	X	X	X	XXX	1	X	X	X	X
SDC	selected device clear		M	AC	Y	0	0	0	0	1	0	0	XXX	1	X	X	X	X
SPD	serial poll disable		M	UC	Y	0	0	1	1	0	0	1	XXX	1	X	X	X	X

Table 38—Remote Message Coding (Continued)

					Bus Signal Line(s) and Coding That Asserts the True Value of the Message													
			Type	Class	DIO8	7	6	5	4	3	2	1	DIO VDC	NDAC	NRFD	ATN	EOI	SRQ	IFC	REN
Mnemonic	Message Name																			
SPE	serial poll enable		M	UC	Y	0	0	1	1	0	0	0	XXX	1	X	X	X	X		
SRQ	service request		U	ST	X	X	X	X	X	X	X	X	XXX	X	X	1	X	X		
STB	status byte	(Notes 8,9)	M	ST	S	X	S	S	S	S	S	S	XXX	0	X	X	X	X		
					8		6	5	4	3	2	1								
TCT	take control		M	AC	Y	0	0	0	1	0	0	1	XXX	1	X	X	X	X		
TAG	talk address group		M	AD	Y	1	0	X	X	X	X	X	XXX	1	X	X	X	X		
UCG	universal command group		M	UC	Y	0	0	1	X	X	X	X	XXX	1	X	X	X	X		
UNL	unlisten		M	AD	Y	0	1	1	1	1	1	1	XXX	1	X	X	X	X		
UNT	untalk	(Note 11)	M	AD	Y	1	0	1	1	1	1	1	XXX	1	X	X	X	X		

The 1/0 coding on ATN when sent concurrent with multiline messages has been added to this revision for interpretive convenience.

NOTES:

1 — D1-D8 specify the device dependent data bits.

2 — E1-E8 specify the device dependent code used to indicate the EOS message.

3 — L1-L5 specify the device dependent bits of the device's listen address.

4 — T1-T5 specify the device dependent bits of the device's talk address.

5 — S1-S5 specify the device dependent bits of the device's secondary address.

6 — S specifies the sense of the PPR.

S	Response
0	0
1	1

P1-P3 specify the PPR message to be sent when a parallel poll is executed.

P3	P2	P1	PPR Message
0	0	0	PPR1
.	.	.	.
.	.	.	.
.	.	.	.
1	1	1	PPR8

7 — D1-D4 specify don't-care bits that shall not be decoded by the receiving device. It is recommended that all zeroes be sent.

8 — S1-S6, S8 specify the device dependent status. (DIO7 is used for the RQS message.)

9 — The source of the message on the ATN line is always the C function, whereas the messages on the DIO and EOI lines are enabled by the T function.

10 — The source of the messages on the ATN and EOI lines is always the C function, whereas the source of the messages on the DIO lines is always the PP function.

11 — This code is provided for system use, see 6.3.

2.13.7.2 Device Dependent Messages

The specific coding of device dependent messages is beyond the scope of this standard. After a talker and listener(s) have been addressed via interface messages, any commonly understood binary, BCD, or alphanumeric code may be used when the ATN message is false.

1) The alphanumeric codes (dense subset of the ISO code, columns 2 through 5) are preferred for communication of the device dependent messages wherever possible. Bit 1 through bit 7 of the ISO code corresponds to DIO1-DIO7.
2) When other codes are used (for example, binary) the most significant bit should be placed on the DIO line that has the highest number (for example, DI08 for bit 8).

The ISO code is further illustrated in Appendix E as it correlates with the codes of this standard.

2.13.8 State Transition Timing Values

The T_x and t_y values listed in Section 2 throughout the interface function descriptions and state diagrams are defined in 3.8.

3. Electrical Specifications

3.1 Application

This section defines the electrical specifications for interface systems to be used in environments where:

1) Physical distance between devices is short
2) Electrical noise is relatively low

All electrical specifications for the driver and receiver circuits are based on the use of transistor transistor logic (TTL) technology.

NOTES:

1 — Interface function circuitry connected to the drivers or receivers may be implemented in other device technologies at the designer's choice.

2 — Driver and receiver devices need only be used on those signal lines required for the interface functions implemented (see 3.5.1 for termination requirements).

3 — Either open collector of three-state drivers may be used as determined by data rate considerations of 3.3 and 5.2.

3.2 Logical and Electrical State Relationships

The relationship between the logical states defined in Table , Remote Message Coding, and the electrical state levels present on the signal lines is as follows:

Coding Logical State	Electrical Signal Levels
0	corresponds to ≥ +2.0 V, called high state
1	corresponds to ≤ + 0.8 V, called low state

The high and low states are based on standard TTL levels for which the power source does not exceed +5.25 V dc and is referenced to logic ground.

This section indicates current flow into a node with a positive sign and current flow out of a node with a negative sign.

3.3 Driver Requirements

Messages may be sent in either an active or passive manner over the interface (see 2.1.3). All passive true message transfer occurs the high state and shall be carried on a signal line using open collector drivers.

3.3.1 Driver Types

Open collector drivers shall be used to drive the SRQ, NRFD, and NDAC signal lines.

Open collector drivers or three-state drivers may be used to drive DIO 1-8, DAV, IFC, ATN, REN, and EOI signal lines with this exception: DIO1-8 shall use open collector drivers for parallel polling applications (see 2.9.3.3).

NOTE — Three-state drivers are useful for systems where higher speed operation is required

It is recommended that a three-state driver be used within a controller to drive the ATN signal line if the controller is intended to be used in a system in which other devices are implemented with three-state drivers on the DIO, DAV, and EOI signal lines.

3.3.2 Driver Specifications

The specifications for drivers shall be as follows:

Low state: Output voltage (three-state or open collector drivers) < +0.5 V at +48 mA sink current

The driver shall be capable of sinking 48 mA continuously.

High state: Output voltage (three-state) ≥ + 2.4 V at –5.2 mA

Output voltage (open collector) (see 3.5)

The indicated voltage values are measured at the device connector between the signal line and logic ground.

See 3.5.3 for additional requirements which may apply to the driver.

3.4 Receiver Requirements

3.4.1 Receiver Specifications, Allowed

The specification for receivers with nominal noise immunity shall be as follows:

Low state: Input voltage ≤ + 0.8 V

High state: Input voltage ≥ + 2.0 V

See 3.5.3 for additional requirements which may apply to the receiver.

3.4.2 Receiver Specifications, Preferred

To provide added noise immunity, the use of Schmitt-type receiver circuits (or equivalent) for all signal lines is recommended. The specifications for these receivers shall be as follows:

Hysteresis: V_t pos - V_t neg ≥ +0.4 V

Low state: Negative going threshold voltage V_t neg ≥ +0.8 V

High state:Positive going threshold voltage +2.0 ≥ V_t pos

3.5 Composite Device Load Requirements

3.5.1 Resistive Termination

Each signal line (whether or not it is connected to a driver or receiver) shall be terminated within the device by a resistive load whose major purpose is to establish a steady-state voltage when all drivers on a line are in the high-impedance state. This load is also used to maintain a uniform device impedance on the line and improve noise immunity. For specific requirements see the last paragraph of 3.5.3, and for typical resistive values see 3.5.5.

3.5.2 Negative Voltage Clamping

Each signal line to which a receiver is connected shall be provided with means to limit the negative voltage excursions. Typically this circuit element is a diode clamp contained within the receiver component.

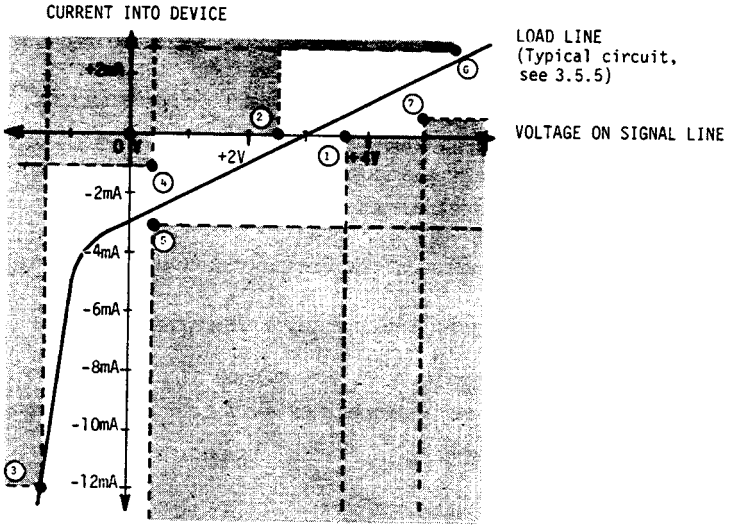
3.5.3 DC Load Requirements

The dc load characteristics of a device are affected by the driver and receiver circuits as well as the resistive termination and voltage clamping circuits; therefore they are specified for the composite device interface circuits not for the individual components. This section, however, provides complete specifications for the resistive termination and voltage clamping circuits.

Load measurement conditions assume that the receiver, driver, and resistive termination circuits are connected together within the device with the driver in the high-impedance state.

Each signal line interface within a device shall have the following dc load characteristics and shall fall within the unshaded area of Fig 15.

1) If $I \leq$ mA, V shall be < 3.7V
2) If $I \geq 0$ mA, V shall be > 2.5V
3) If $I \geq 12.0$ mA, V shall be > −1.5 V (only if receiver exists)
4) If $V \leq 0.4$ V, I shall be < + 1.3 mA
5) If $V \geq 0.4$ V, I shall be > −3.2 mA
6) If $V \leq 5.5$ V, I shall be < 2.5mA
7) If $V \geq 5.0$ V, I shall be > 0.7 mA or the small-signal Z shall be < 2 kΩ at 1 MHz



NOTE: The slope of the dc load line should, in general, correspond to a resistance not in excess of 3 kΩ.

Figure 15—DC Load Boundary Specification

3.5.4 Capacity Load Limit

The internal capacitance load on each signal line shall not exceed 100 pF within each device (see 5.2).

NOTE — The effect of device capacitance on bus operations is most critical at low voltages. Since the design of driver and receiver circuits may contribute capacitive loads that vary with voltage, the capacitance should be measured at several voltage levels, all below 2 V, with the device powered on.

3.5.5 Typical Circuit Configuration

Figure 16 shows a typical circuit configuration for signal line input-output circuits for which readily available component exist. This basic circuit is compatible with both TTL micro-circuit and discrete element devices. The specifications for this typical configuration are as follows:

R_{L1}: 3 kΩ ± 5%, (to V_{cc})
R_{L2}: 6.2 kΩ ± 5% (to ground)

Driver: Output leakage current (open collector driver) +0.25 mA max at V_0 = +5.25 V Output leakage current (three-state driver) ± 40 μA max at V_0 = +2.4 V

Receiver: Input current

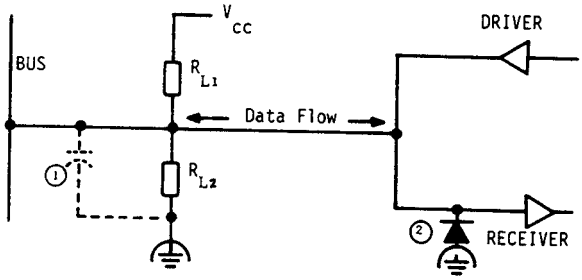
–1.6 mA max at V_o = + 0.4 V

Input leakage current

+40 μA max at V_o = +2.4 V

+ 1.0 mA max at V_o = 5.25 V

V_{cc}: +5 V ± 5%



1 = Internal capacitance (cabling and transceivers) as defined by 3.5.4 and measured according to 3.7.2 for voltage values from 0 to 5 V.
2 = Typically contained within receiver component

Figure 16—Typical Signal Line Input-Output Circuit

Only a single driver and receiver may be connected to each signal line in the typical configuration of Fig 16. Other configurations may exist in which this restriction does not hold provided the composite device load specifications of 3.5.3 are met.

3.6 Ground Requirements

The overall shield of the interconnecting cable shall be connected through one contact of the connector to frame (safety earth) to minimize susceptibility to and generation of external noise.

> **WARNING:** *Devices should not be operated at significantly different frame potentials. The interface connection system may not be capable of handling excessive ground currents.*

It is recommended that the ground returns of the individual control and status signal lines be connected to logic ground at the logic circuit driver or receiver to minimize cross-talk interference transients.

3.7 Cable Characteristics

3.7.1 Conductor Requirements

The maximum resistance for the cable conductors shall be, per meter of length:

1) Each signal line (for example, DIOI, ATN) 0.14 Ω
2) Each individual signal line ground return 0.14 Ω
3) Common logic ground return 0.085 Ω
4) Overall shield 0.0085 Ω

3.7.2 Cable Construction

The cable should contain at least 24 conductors of which 16 shall be used for signal lines and the balance used for logic ground returns and overall shield.

The maximum capacitance measured (at 1 kHZ) between any signal line and all other lines (signals, grounds, and shield) connected to ground shall be 150 pF per meter.

The shield shall contain a braid of 36 AWG wire or equivalent with at least 85% coverage.

The cable shall be constructed to minimize the effects of cross talk between signal lines, the susceptibility of the signal lines to external noise, and the transmission of interface signals to the external environment.

1) Each of the signal lines DAV, NRFD, NDAC, IFC, ATN, EOI, REN, and SRQ shall be twisted with one of the logic ground wires or isolated using an equivalent scheme to minimize cross talk
2) The cable shall contain an overall shield carried through the cable assembly and connectors at both ends to be returned to earth ground.
3) A cable construction in which twisted pairs contained in the core of the cable and the individual DIO lines contained around the periphery of this core has been found satisfactory as has been the use of twisted pair conductors for all 16 signal lines where each signal line is twisted with an earth conductor.
4) Alternately, any other internal cable construction which yields the same results may be used.

3.8 State Transition Timing Values

To ensure maximum possible compatibility among interconnected devices, Table 39 states the mandatory time relationships between critical signal inputs and outputs to a specific device.

The T_1, $T_6 - T_9$ time values allow for the normal propagation delays of the transmission path and the circuit delays within other devices.

They are measured from the time the source output driver is seen to start its transition as viewed from its associated connector. It is further recommended that, for the minimum values of T_1, T_6–T_9; the high-state driver voltage not be degraded, cable resistance and capacitance be kept as low as possible, and cross talk be kept at a minimum value.

Table 39—Time Values

Time Value Identifier*	Function (applies to)	Description	Value
T_1	SH	settling time for multiline messages	≥ 2 μs†
t_2	SH,AH,T,L,LE,TE	response to ATN	≤ 200 ns
T_3	AH	interface message accept time‡	>0 § §
t_4	T,TE,L,LE,C,RL	response to IFC or REN false	<100 μs
t_5	PP	response to ATN ∧ EOI	≤ 200 ns
T_6	C	parallel poll execution time	≥ 2 μs
T_7	C	controller delay to allow current talker to see ATN message	≥ 500 ns
T_8	C	length of IFC or REN false	> 100 μs
T_9	C	delay for EOI**	≥ 1.5 μs††
T_{10}	C	delay for DAV	≥ 1.5 μs

*Time values specified by a lower case *t* indicate the maximum time allowed to make a state transition. Time values specified by an upper case *T* indicate the minimum time that a function shall remain in a state before exiting

†If three-state drivers are used on the DIO, DAV, and EOI lines, T_1 may be:

1) ≥ 1100 ns
2) Or ≥ 700 ns ff it is known that within the controller ATN is driven by a three-state driver however this value is not recommended.
3) Or ≥ 500 ns for all subsequent bytes following the first sent after each false transition of ATN (the first byte shall be sent in accordance with (1) or (2))
4) Or ≥ 350 ns for all subsequent bytes following the first sent after each false transition of ATN under conditions specified in 5.2 and warning note.

‡Time required for interface functions to accept, not necessarily respond to, interface messages.

§Implementation dependent

**Delay required for EOI, NDAC, and NRFD signal lines to indicate valid states.

††≥ 600 ns for three-state drivers.

4. Mechanical Specifications

4.1 Application

This section defines the mechanical specification for interface systems to be used in environments where:

1) Physical distances between devices are limited
2) Star or linear bus interconnection networks are useful
3) Connector mounting space is limited

4.2 Connector Type

A quality connector of the rack and panel type with proven performance shall be used which has these minimum characteristics.

4.2.1 Electrical Considerations

Voltage rating; 200 V

Current rating; (at T = 25 °C) 5 A per contact

Contact resistance; (at 10 mA) < 20 mΩ

Insulation resistance; > 1 GΩ

Test Voltage; (1 min, 20 °C) 500 V

Capacitance; (between contacts at 1 kHz) < 1.5 pF

Endurance; (with 1 A and 70 °C) > 1000 h

4.2.2 Mechanical Considerations

Number of contacts; 24

Contact surface; (self-wiping) 2.16 mm

Polarization; (shell shape) trapezoidal

Shell material; corrosion resistant, conductive

Retention force per contact; > 0.15 N

Typical insertion and withdrawal force (F); 8 N<F<89 N

Endurance; (for specified contact resistance) > 500 insertions

Clearance between adjacent contacts; > 0.5 mm

Solderability (if applicable); nominal 235 °C, 2 s

Typical external dimensions (see 4.4 for additional dimensions);

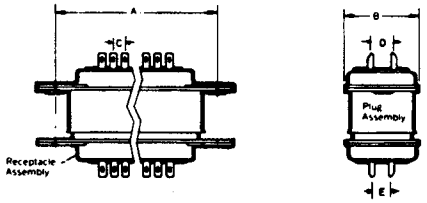
A 46.8 mm

B 15.5 mm

C 2.16 mm

D 4.29 mm

E 3.33 mm



4.2.3 Environmental Considerations

Basic environmental performance relative to temperature, humidity and vibration criteria should be determined in accordance with IEC Publication 68-2 (1982) [2] for climatic category 25/070/21 or MIL STD 202F (1986) [5], where appropriate.

4.3 Connector Contact Assignments

A contact assignment of the cable connector and the device connector shall be as shown below:

Contact	Signal Line	Contact	Signal lines
1	DIO1	13	DIO5
2	DIO2	14	DIO6
3	DIO3	15	DIO7
4	DIO4	16	DIO8
5	EOI (24)	17	REN(24)
6	DAV	18	Gnd. (6)
7	NRFD	19	Gnd. (7)
8	NDAC	20	Gnd. (8)
9	IFC	21	Gnd. (9)
10	SRQ	22	Gnd. (10)
11	ATN	23	Gnd. (11)
12	SHIELD	24	Gnd. LOGIC

NOTE — Gnd, (*n*) refers to the signal ground return of the referenced contact. EOI and REN return on contact 24.

4.4 Device Connector Mounting

Each device shall be provided with a receptacle type connector having the typical dimensions shown in Fig 17. The two rows of twelve contacts each are centered within the trapezoidal shell. The connector mounting shall make provisions to accept the locking screws of the cable assembly.

The preferred orientation of the connector, as mounted on a device and viewed from the rear of the device in its normal operating position, is with contact 1 in the upper right-hand corner. The connector location should allow for sufficient cable clearance as shown in Fig 21.

The connector may be mounted on either the outside or inside of the panel for which the typical panel cutout dimensions are given in Fig 18.

The connector shall be attached to the device with one of the stud mount standoffs shown in Fig 19 as determined by the panel mounting method used.

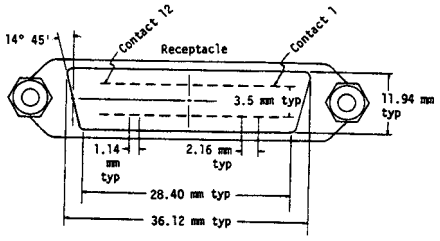


Figure 17—Receptacle Dimensions

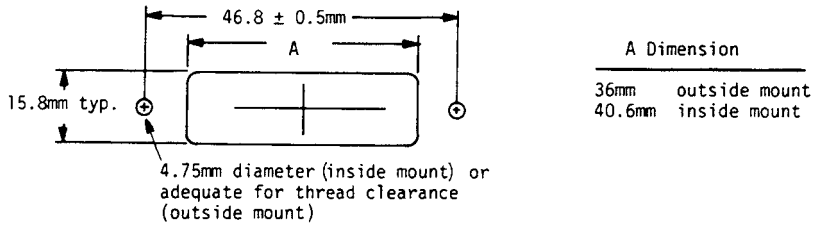


Figure 18—Connector Panel Cutout

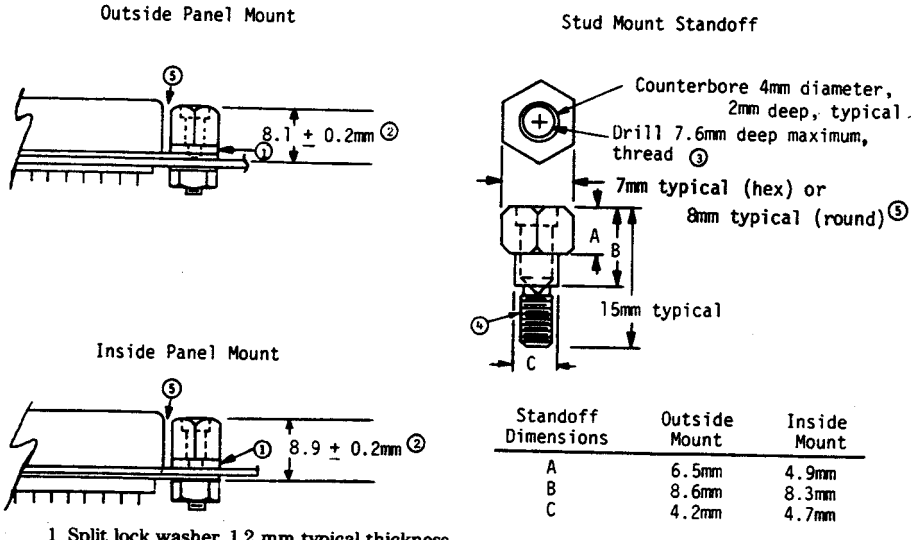


Figure 19—Mounting Dimensions

4.5 Cable Assembly

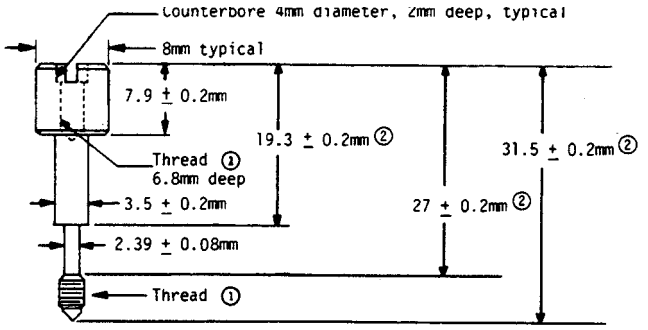
The cable assembly shall be provided with both a plug and a receptacle connector type at each end of the cable. The preferred method of assembling the stacked connectors contains a rigid structure (assures a reliable and positive connection of multiple cable assemblies) as shown in Fig 21.

Each connector assembly shall be fitted with a pair of captive locking screws. Each lock screw shall conform to the mechanical dimension shown in Fig 20. A retaining ring, or equivalent, shall be used to retain the lock screw as a captive element.

It is recommended that each pair of connectors, assembled according to the first paragraph of 4.5, be partially enclosed within a suitable housing as shown in Fig 21.

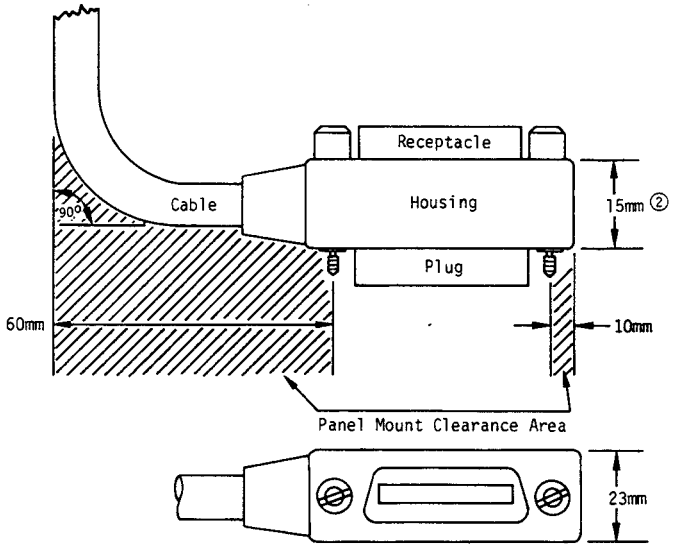
Individual cable assemblies may be of any length up to 4 m.

The housing may be plastic or metallic material, the latter is preferred for superior EMC performance; see Appendix I.4 for additional information on appropriate means for screening the complete cable assembly.



1 = ISO metric thread M3.5×0.6 or equivalent O.M.F.S. thread 3.5PO.6
2 Length is function of housing dimension

Figure 20—Lock Screw



NOTES: (1) All measurements are typical.
(2) Length of lock screw is a function of this dimension.

Figure 21—Cable Connector Housing

5. System Applications and Guidelines for the Designer

5.1 System Compatibility

This interface system offers a wide range of capability from which to choose the appropriate interface functions to fit different applications. Within most interface functions a number of options are available. In addition the designer has freedom to select all the device dependent capabilities contained within the device functions.

It is the responsibility of the designer to define the complete capability of a device (interface system choices and related device dependent interactions) so that the end user of the device can efficiently interface and program the device for appropriate system applications.

Selection of a minimum set of interface functions from Section 2 leads to the following minimum set of signal lines in order to be system compatible:

1) DIO 1–7
2) DAV, NRFD, NDAC
3) IFC and ATN (unnecessary in systems without a controller)

In order to provide system compatibility a designer shah not introduce new interface functions beyond those designed in Section 2.

5.2 Data Rate Consideration

Designers of devices intended to communicate over the interface system bus are advised to consider the relationships between various levels of system performance and the specific device circuits used to provide these different levels of performance. The following statements are intended as guidance.

5.2.1

The interface bus will operate at distances up to 20 m, at a maximum of 250 000 bytes per second, with an equivalent standard load for each 2 m of cable using 48 mA open collector drivers.

5.2.2

The interface bus will also operate at distances up to 20 m at a maximum of 500 000 bytes per second, with an equivalent standard load for each 2 m of cable using 48 mA three-state drivers.

5.2.3

Higher Speed Operation. To achieve the maximum possible data transfer ratio (nominally up to 1 000 000 bytes per second) within a system, the following guidelines should be observed:

1) All devices expected to talk at; higher rates should use a minimum T_1 value of 350 ns
2) All devices expected to operate at higher rates should use 48 mA three-state drivers
3) The device capacitance on each lead (REN and IFC excepted) should be less than 50 pF per device. In a system configuration the total device capacitance should be no more than 50 pF for each equivalent resistive load in the system.
4) All the devices in the system should be powered on

5) Interconnecting cable links should be as short as possible up to a maximum of 15 m total length per system with at least one equivalent load for each meter of cable

> **WARNING:** *Any time a device following condition (1) is placed in a system,* ***even if higher speed operation is not intended****, there may be data transfer errors if conditions (2) through (5) are not met for that system.*

NOTES:

1 — Devices with a $T_{(1)}$ value of less than 500 ns, device capacitance of 50 pF, or having multiple resistive loads shall be so marked, as acceptable variants. Multiple resistive powered loads, beyond one per signal line per device, can be added up to a maximum of 15 loads per signal line per system. Multiple loads can be used, with caution, to improve the device to cable-length ratio (up to 15 m max).

2 — Actual maximum data rates may be more a function of device dependent time delays than the cable characteristics or timing values of 2.13.8.

3 — See 3.5.4 for warning of adverse effects of variable capacitive loading

4 — Use of data byte buffer store within the device may be advantageous.

5.3 Device Capabilities

5.3.1 Busy Function

In system operation it is useful to either program a device or initiate some operation within a device and then proceed to communicate with other devices (while the first device is busy carrying out the required task). The busy (operation being completed) function is a device state and not an interface state. In order to permit interface bus communication independent of the busy condition of a device, three possible methods are available:

1) SRQ and serial poll
2) Parallel poll
3) NFRD hold

Both the serial poll and parallel poll methods are described in Section 2.

5.3.2 NRFD Hold

The NRFD signal line may be gated to incorporate the busy function.

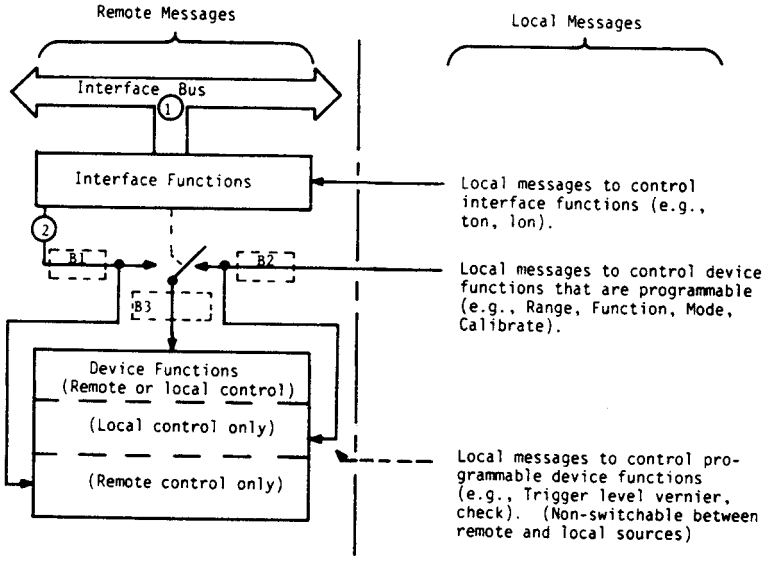
In so doing, the NRFD signal (or RFD message) changes its definition to include more than the normal “ready for the next data byte” meaning. The internal busy signal is gated to the NRFD signal line through the AH function. In this manner the device may be unaddressed as listener during a “busy cycle” and the interface bus may be used for other purposes. When readdressed as a listener, the device will indicate its internal busy status to the interface. The device indicates “busy” be setting NRFD to 1 and indicates “operation complete” by setting NRFD to 0.

> **CAUTION —** :*If NRFD hold is used for the busy function where a device may not recover or may reach the nonbusy condition, then another listen address (always accessible) should be available to clear the potential hang-up condition.*

5.3.3 RL Applications

The designer is free to implement within a device, whatever programmable device functions are appropriate for a particular device application(s). The designer is not free to remotely program the local control functions which interact directly with the interface functions as specified throughout Section 2.

To implement a programmable device capable of being either remotely or locally controlled may require the switching of some or all of the typical controls illustrated in Fig 22. This figure is not meant to imply a comprehensive set of switching techniques, switching locations, or switched message contents.



1 Remote messages (for example, ATN, MLA interface messages, DAB device messages)
2 Remote control of programmable device functions (for example, range, function) and the execution of device functions (for example, clearing, triggering)
B Possible locations of buffer storage elements. Normally B1 and B2 are used in combination or B3 is used alone.

Figure 22—Remote—Local Message Paths

5.4 AND and OR Logic Operations

The message sent by one interface function is not necessarily the same message received by another interface function, (irrespective of time differences due to transmission characteristics of the signal lines) in case of three messages as used in the SH, AH, and SR interface functions:

1) The RFD (or DAC) message received (by an SH function) shall be the logic AND of all RFD (or DAC) messages sent (by all AH functions)
2) The SRQ message received (by a C) shall be the logic OR of all SRQ messages sent (by the SR functions)

NOTE — The DAV message received (by all AH functions) shah be the DAV message sent (by one and only one SH function).

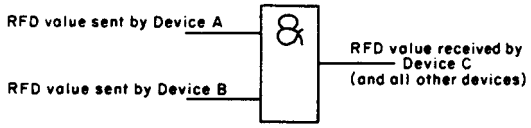
5.4.1 RFD and DAC Messages

The RFD (or DAC) message sent true (or false) by an AH function is performed by setting the NRFD (or- NDAC) signal line to 0 (high) or driving the NRFD (or NDAC) signal line to I (low), respectively.

The RFD (of DAC) message received by an SH function is received true when the state of the signal line is 0 (high) which means that all RFD (or DAC) messages sent are sent passive true.

The RFD (or DAC) message received by an SH function is received false when the state of the signal line is 1 (low) which means that one or more RFD (or DAC) messages sent are sent bi-state false.

The logical equivalent of these conditions is illustrated below.



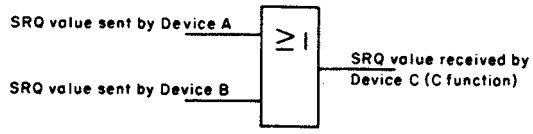
5.4.2 SRQ Message

The SRQ message sent true or false, by an SR function is performed by driving the SRQ signal line to 1 (low) or setting the SRQ signal line to 0 (high), respectively.

The SRQ message received by the C function is received true when the state of the bus signal line is 1 (low) which means that one or more SRQ functions have sent the SRQ message true.

The SRQ message received by a controller function is received false when the state of the bus signal line is 0 (high) which means that all SRQ functions have sent the SRQ message passive false.

The logical equivalent of these conditions is illustrated below.



5.4.3 Circuit Implementations

A typical circuit configuration with which the AND and OR functions on the respective bus signal lines can be performed is that represented in one 3.5.5, Fig 16. The driver element shall be a bi-state (open collector) driver as represented in Fig 23.

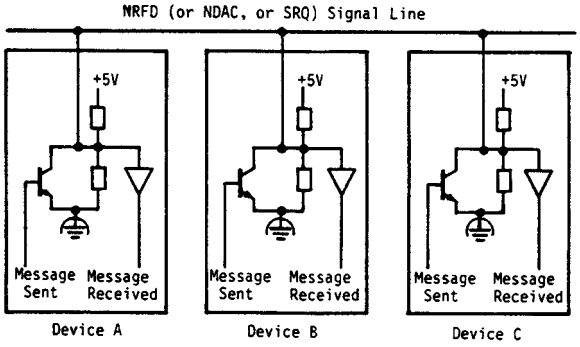


Figure 23—Bi-State Signal Line Logic (Open Collector Drivers)

NOTE — Whether or not invertors are used to convert the internal representation of the RFD (or DAC) message into the actual message sent on the bus signal lines depends on the internal assertion definition for true and false with respect to the high or low voltage levels used internal to the device. This matter is left to the designer.

Typical signals presented to the NRFD (or SRQ) interface bus signal lines by devices *A* and *B* as described in 5.4.1 and 6.4.2 may be represented as shown in Fig 24. Only the composite signal line waveform as received at device *C* exists on the bus. The signal levels shown for devices *A* and *B* exist only within the devices' drivers and not on the bus signal line.

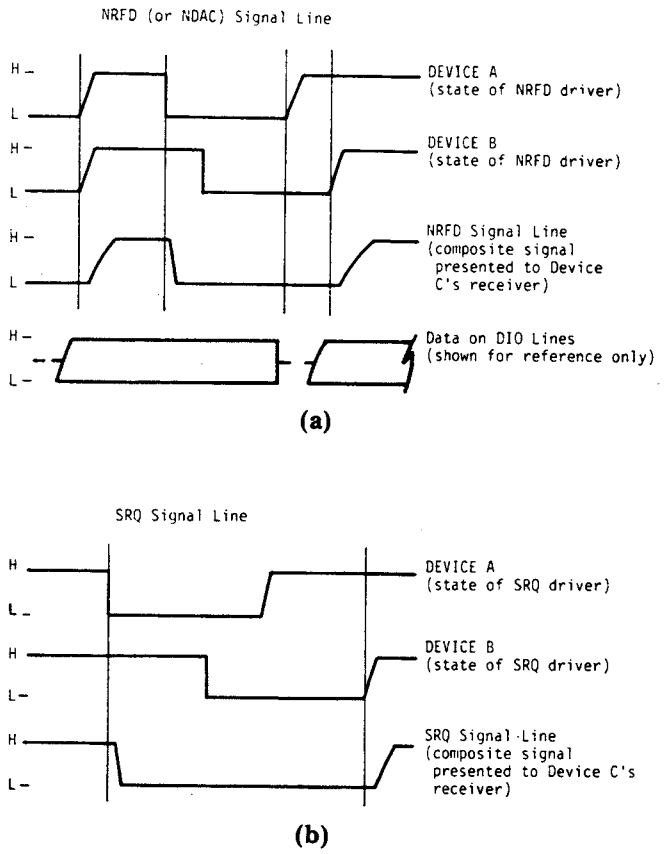


Figure 24—Signal Line Logic and Timing Relationships—(a) NRFD (or NDAC) Signal Line; (b) SRQ Signal Line

5.5 Address Assignment

Normally, a device will be assigned a single talk and single listen address to perform the essential tasks. It may be useful to design a device with multiple talk (or listen) addresses to facilitate system requirements. A device could be assigned two talk addresses (for example, one to output raw data the other to output processed data). Care should be given to minimize the use of such multiple addresses as later system configurations may be restricted due to excessive use of primary addressing capability.

5.6 Typical Combinations of Interface Functions

The designer is free to select the particular interface functions required to meet specific device applications. The selection of certain interface functions requires the inclusion of other interface functions as defined throughout the allowable subset clauses of Section 2

Device	Typical Interface Functions Used
Signal generator (Only able to listen)	AH, L, RL, DT
Tape Reader (Only able to talk)	SH, AH, T
Digital voltmeter (Able to talk and listen)	SH, AH, T, L, SR RL, PP,DC, DT
Calculator (Able to talk, listen, and control)	SH, AH, T, L, C

The list above represents typical combinations of interface functions and does not imply that these are the only combinations possible or useful.

5.7 Unimplemented Interface Message Handling

When the ATN message is true, a device should ignore all multiline messages that are inappropriate given the current states of the implemented interface functions A device shall handshake the inappropriate multiline message, but should not take further action including recording an error, requesting service, or interrupting the exchange of remote messages. Subsequent messages should be processed in the normal manner.

Examples:

1) If a listening device implements the DT0 subset, the group execute trigger (GET) remote message should be ignored by the device.
2) If the LE interface function is in LPIS and the PP interface function is in PUCS, the my secondary address (MSA) remote message should be ignored by the device.
3) If a universal command group remote message that is not GTL, SDC, PPC, GET, or TCT or an addressed command group remote message that is not LLO, DCL, PPU, SPE, or SPD is received, the device should ignore the message.

6. System Requirements and Guidelines for the User

6.1 System Compatibility

Devices designed to this interface system may have a wide range of capability relative to their ability to communicate over the interface. This standard does not cover the operational characteristics of devices, only the mechanical, electrical, and functional capabilities of the interface system.

The burden of responsibility for system compatibility at the operational level is on the user. The user shall be familiar with all device characteristics interacting with the interface system (for example, device dependent program codes, output data format and codes, etc).

6.2 System Installation Requirements

This includes system configuration restrictions.

6.2.1 Maximum Number of Devices

The maximum number of devices that can be connected together to form one interface system is 15.

6.2.2 Minimum System Configurations

An interface system shall contain one or more devices containing at least one T function, one L function, and one C function.

If all the T functions include the use of the ton message (talker types T1, T3, T5, T7, TE1, TE3, TE5, or TE7), and all the L functions include the Ion message (listener types L1, L3, LE1 or LE3), a system may be operated without a C function when the ton and Ion messages are true. The lon and ton messages are normally provided by local switches.

6.2.3 System Controllers

All system configurations containing more than one controller shall satisfy the following conditions:

1) There shall not be more than one C function in a system that is in the system control active state (SACS)
2) Every controller in the system shall be able to pass and receive control of the interface

6.2.4 Devices Powered Off and On

A system will operate without adversely affecting normal data transfer with at least two thirds of the devices powered on. A system will operate correctly with any number of devices powered off provided all those devices powered off do not degrade the specified high state condition, that is, that the voltage on each signal line with all its output drivers passive false should exceed +2.5V with respect to the logic ground at each device.

Unless special precautions are taken (that is, use of special driver circuits beyond the scope of this standard) powering a device to on while the system is running may cause faulty operation.

6.3 Address Assignment

6.3.1 Primary Talk Addresses

A device that contains a T function or a TE function may be assigned any value for bits T1 through T5 of its my talk address (MTA) message code other than:

T5	T4	T3	T2	T1
1	1	1	1	1

This code, defined as UNT, is provided as a systems convenience, for the controller, to return all devices to the talker idle state.

Two or more T functions (whether within the same or separate devices) shall not be assigned the same value for bits T1 through T5 of their MTA codes.

A device that contains both a T and L function may be assigned a talk address such that T1 through T5 of its MTA code equals L1 through L5 of its MLA code.

ATE interface function shall not be assigned the same value for bits T1 through T5 of its MTA code as that assigned to a T function.

6.3.2 Primary Listen Addresses

A device that contains an L function or an LE function may be assigned any value for bits L1 through L5 of its my listen address (MLA) code other than:

L5	L4	L3	L2	L1
1	1	1	1	1

Two or more L functions (usually within separate devices) may be assigned the same value for bits L1 through L5 of their MLA codes.

A device that contains both an L and T function may be assigned a listen address such that LI through L5 of its MLA code equals T1 through T5 of its MTA code.

6.3.3 Secondary Addresses

A device that contains a TE function or LE function may be assigned any value for bits S 1 through S5 of its my secondary address (MSA) code other than:

S5	S4	S3	S2	S1
1	1	1	1	1

Two or more TE functions (whether within the same or separate devices) shall not be assigned the same value for bits T1 through T5 of their MTA codes and bits S1 through S5 of their MSA codes.

Two or more LE functions (usually within separate devices) may be assigned the same value for both bits L1 through L5 of their MLA codes and bits S1 through S5 of their MSA codes.

A device that contains both a TE and LE function may be assigned a listen address such that L1 through 1.5 of its MLA code equals T1 through T5 of its MTA code and both functions may utilize the same secondary address.

6.4 Cabling Restrictions

6.4.1 Maximum Cable Length

The maximum length of cable that shall be used to connect together a group of devices within one bus system is:

1) 2 m times the number of devices
2) Or 20 m, whichever is less

6.4.2 Distribution of Maximum Cable Lengths

The maximum length of cable as defined in 6.4.1 may be distributed among the devices in a system in any manner deemed suitable by the user. Caution should be taken if individual cable length exceeds 4 m.

6.4.3 Cabling Configurations

Cables may be interconnected in any manner deemed suitable by the user (that is, star, linear, or combinations thereof).

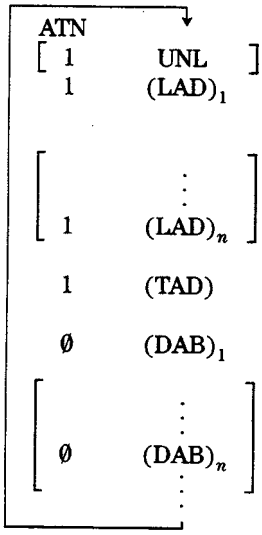
Devices should not be operated at significantly different frame potentials as the system may not be capable of handling excessive ground currents.

6.5 Operational Sequence Guidelines

Most interface communication tasks require a sequence of coded messages to be sent over the interface. Although the specification of operational sequences is beyond the scope of this standard, several sequences are recommended for typical tasks. Many other sequences might be found useful.

NOTE — Caution should be observed by the system user to assure that exit conditions from a given sequence leaves devices in an acceptable state. Adequate device documentation facilitates this process.

6.5.1 Data Transfer



Inhibits all current listeners (can be omitted if not required)
Each address sent enables a specific device to receive future data bytes.
More than one address may be sent if multiple listeners are desired

The address sent enables a specific device to send data as soon as ATN becomes 0

Sent by the currently enabled talker to all currently enabled listeners

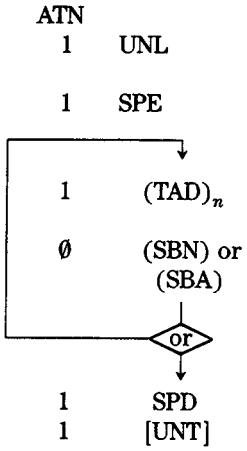
Bytes may be sent until the controller again sets ATN to 1 to repeat the 0 (DAB)(n) sequence. If the talker is sending a specific-length record, it may operationally set EOI = 1 concurrent with the last byte

NOTE — (LAD) represents a listen address of a specific device.

(TAD) represents a talk address of a specific device. (DAB) represents any data byte.
(TAD) represents a talk address of a specific device
(DAB) represents any data byte
[ ] indicates optional segments of a sequence.
( ) indicates messages not uniquely defined in this standard.

6.5.2 Serial Poll

(issued by controller usually whenever SRQ= 1 on the interface)



Prevents other devices from listening to status sent (controller continues to listen without being addressed)

Puts interface into serial poll mode during which all devices send status instead of data when enabled

Enables a specific device to send status. Within this loop, devices should be sequentially enabled

Status byte sent by enabled device. If SBN was sent, loop should be (SBA) repeated. If SBA was sent, the enabled device is identified as having or sent SRQ over the interface and will automatically remove it

Removes the interface from serial poll mode

Disables last talker from sending data if ATN is set 0

NOTES:

1 — (TAD) represents a talk address of a specific device.

2 — (SBN) represents a status byte sent by a device in which a request for service is not indicated (bit 7 =0) (SBN = STB $\wedge$ $\overline{\text{RQS}}$).

3 — (SBA) represents a status byte sent by a device in which a request for service is indicated (bit 7 = l) (SBA = STB $\wedge$ RQS).

6.5.3 Control Passing

ATN		
1	(TAD)	The address sent should be that of the device to which control is being passed
1	TCT	Notifies addressed device to take over control of the interface
1		New controller-in-charge at this time

NOTE — (TAD) represents a talk address of a specific device.

6.5.4 Parallel Poll

6.5.4.1 Parallel Poll Configure

ATN		
1	(LAD)	Addresses a particular device for which a parallel response coding is to be assigned
1	PPC	Enables the addressed listener to be configured
1	PPE	Bit 4 specifies the sense of the poll response, bits 1 … 3 specify, in binary code, the DIO line on which the poll response is to be given
1	UNL	End of the configuration routine

NOTES:
1 — (LAD) represents a listen address of a specific device.

2 — The PPE command can be cleared by a PPD command.

3 — The configuration can be cleared by a PPU command.

6.5.4.2 Parallel Poll Response

ATN	IDY	
1	1	Whenever the bus is in this state, predetermined devices will each place their requests on a specific DIO line. If more than one device is sharing a DIO line, the line value can indicate either an ORing or an ANDing of requests depending on commands previously sent to the devices in- structing them to use the 0 or 1 value to request service.

6.5.5 Placing Devices in Forced Remote Control

ATN	REN		
1	1	LLO	Disables all devices' rtl message
I	1	$(LAD)_1$	Each address sent places the addressed device into remote state, disabling all local controls
1	1	.	
1	1	.	
1	1	$(LAD)_n$	

NOTES:
1 — (LAD) represents a listen address of a specific device. (Devices will all revert back to local state as a group at any time a Φ value of REN is placed on the interface.)
2 — Selected local controls may be re-enabled by sending device-dependent remote messages.

6.5.6 Sending Interface Clear

While the IFC message is being sent only the DCL, LLO, PPU, and REN universal commands will be recognized.

Annex A Typical Instrument System

(Informative)

(These Appendixes are not a part of ANSI/IEEE Std 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation, but are included solely for the purposes of information.)

The typical system shown in Fig A.1 illustrates the capability of the interface system to handle a variety of instrumentation system needs. Two possible event sequences, to accomplish specific measurement tasks using the interface system, are included as examples.

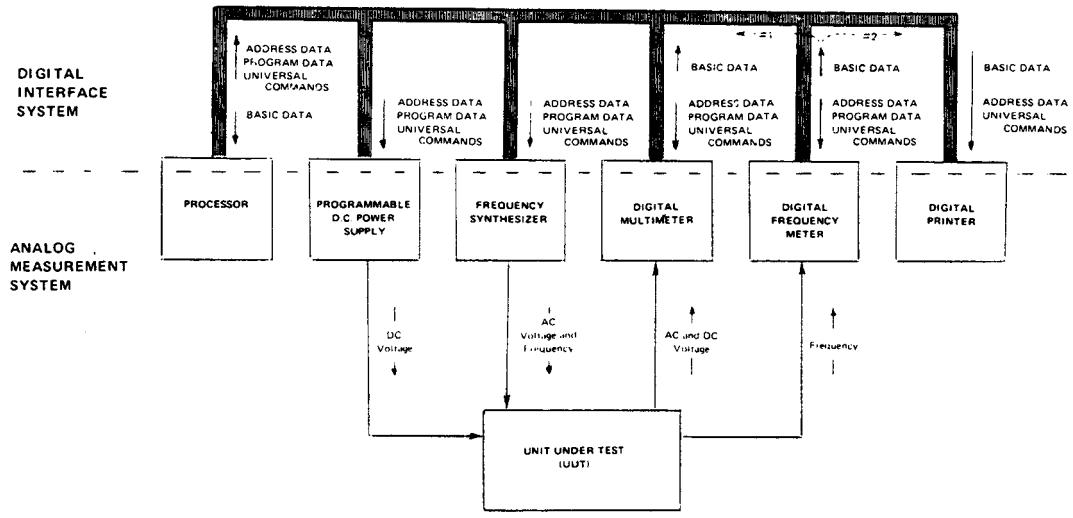


Figure A.1—Typical System Showing Capability of Interface System to Handle Variety of Instrumentation System Needs

A.1 Event Sequence 1

(Device Dependent Data Returned to Processor)

Processor programs instruments and initiates measurements; resulting basic data is returned to processor.

1) Processor initializes the interface system by sending the IFC message true.
2) Processor causes all devices to set their internal conditions to be a predefined state by sending the DCL message true.
3) Processor sends the listen address of the dc power supply followed by program data for that device.
4) Processor sends the unlisten command, then the listen address for the next device, followed by program data for it.
5) Event (4) is repeated until each device of interest for this specific test has been addressed and programmed, then the unlisten command is sent.
6) Processor sends listen address of selected measurement device (for example, the digital frequency meter), then that program code required to initiate a measurement.
7) Processor sends unlisten command, addresses itself to listen, then sends talk address of the measurement device.
8) Upon completion of its internal measurement cycle, the digital frequency meter sends (talks) its measurement results (device dependent data) to the addressed listener, the processor.

A.2 Event Sequence 2 (Device Dependent Data Directed to Digital Printer)

Processor programs instruments and initiates measurements; resulting device dependent data is returned to another device.

(1)—(6) Identical to Event Sequence 1.

(7) Processor sends unlisten command, then the listen address of the digital recorder, followed by the talk address of the measurement device.

(8) Upon completion of its measurement, the measurement device again sends its resulting device dependent data to the addressed listener, the digital recorder.

NOTE — If the processor were to address both the digital recorder and itself, the resulting device dependent data would be accepted by both devices, even though the two may have vastly different rates at which data can be accepted.

Annex B Handshake Process Timing Sequence

(Informative)

B.1 General Comments

Each data byte transferred by the interface system uses the handshake process to exchange data between source and acceptor. Typically, the source is a talker and the acceptor a listener.

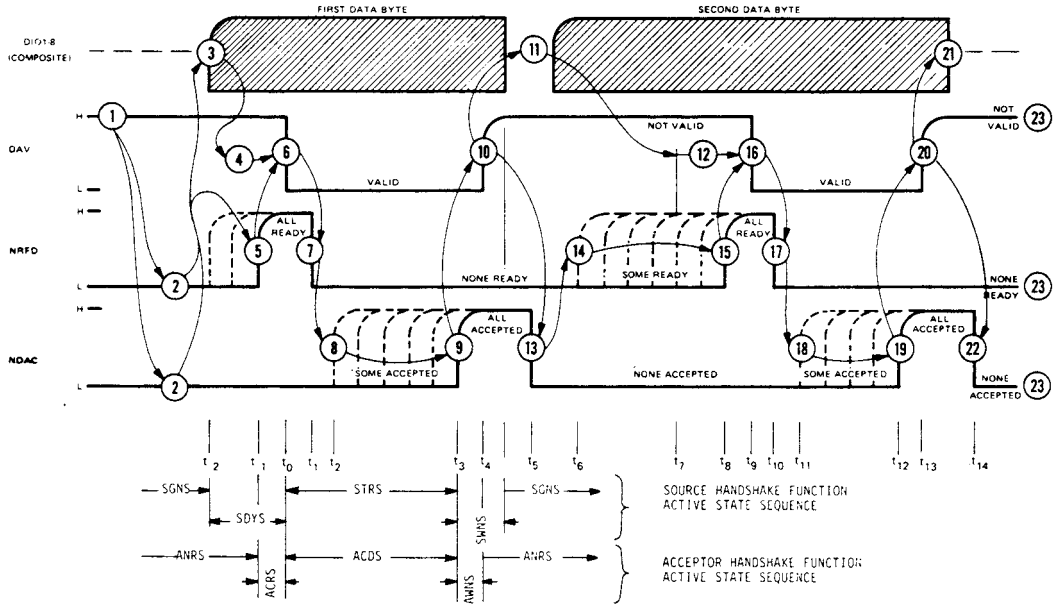
Figure B.1 illustrates the handshake process by indicating the actual waveforms on the DAV, NRFD, and NDAC signal lines. The NRRD and NDAC signals each represent composite waveforms resulting from two or more listeners accepting the same data byte at slightly different times due to variations in the transmission path length and different response rates (delays) to accept and process the data byte.

Figure B.2 represents the same sequence of events, in flow chart form, to transfer a data byte between source and acceptor. The annotation numbers on the flow chart and the timing sequence diagram refer to the same event on the list of events.

B.2 List of Events for Handshake Process

The ( ) refers to sequential events in Fig B.1.

1) — Source initializes DAV to high (H) (data not valid).
2) — Acceptors initialize NRFD to low (L) (none are ready for data), and set NDAC to low (L) (none have accepted the data).
3) t_{-2} Source checks for error condition (both NRFD and NDAC high), then sets data byte on DIO lines.
4) t_{-2}—t_o Source (delays to allow data to settle on DIO lines.
5) t_{-1} Acceptors have all indicated readiness to accept first data byte; NRFD lines goes high.
6) t_o Source, upon sensing NRFD high, sets DAV low to indicate that data on DIO lines is settled and valid.
7) t_1 First acceptor sets NRFD low to indicate that it is no longer ready, then accepts the data. Other acceptors follow at their own rates.
8) t_2 First acceptor sets NDAC high to indicate that it has accepted the data. (NDAC remains low due to other acceptors driving NDAC low).
9) t_3 Last acceptor sets NDAC high to indicate that it has accepted the data; all have now accepted and the NDAC line goes high.
10) t_4 Source, having sensed that NDAC is high, sets DAV high. This indicates to the acceptors that data on the DIO lines must now be considered not valid.
11) t_4—t_7 Source changes data on the DIO lines.
12) t_7—t_9 Source delays to allow data to settle on DIO lines.
13) t_5 Acceptors, upon sensing DAV high (at 10) set NDAC low in preparation for next cycle. NDAC line goes low as the first acceptor sets the line low.
14) t_6 First acceptor indicates that it is ready for the next data byte by setting NRFD high. (NRFD remains low due to other acceptors driving NRFD low).
15) t_8 Last acceptor indicates that it is ready for the next data byte by setting NRFD high; NRFD signal line goes high.
16) t_9 Source, upon sensing NRFD high, sets DAV low to indicate that data on DIO lines is settled and valid.
17) t_{10} First acceptor sets NRFD low to indicate that it is no longer ready, then accepts the data.
18) t_{11} First acceptor sets NDAC high to indicate that it has accepted the data [as in (8)].
19) t_{12} Last acceptor sets NDAC high to indicate that it has accepted the data [as in (9)].
20) t_{13} Source, having sensed that NDAC is high, sets DAV high [as in (10)].
21) — Source removes data byte from DIO signal lines after setting DAV high.
22) t_{14} Acceptors, upon sensing DAV high, set NDAC low in preparation for next cycle.
23) — Note that all three handshake lines are at their initialized states [as in (1) and (2)].



*(See Fig B2 and List of Events) H ≥ +2.0 V; L ≤ +0.8 V

Figure B.1—Signal Line Timing Sequence for One Talker and Multiple Listeners Using Handshake Process[6]

[6](See Fig B.2 and List of Events) H ≥ +2.0 V; L ≤ +0.8 V

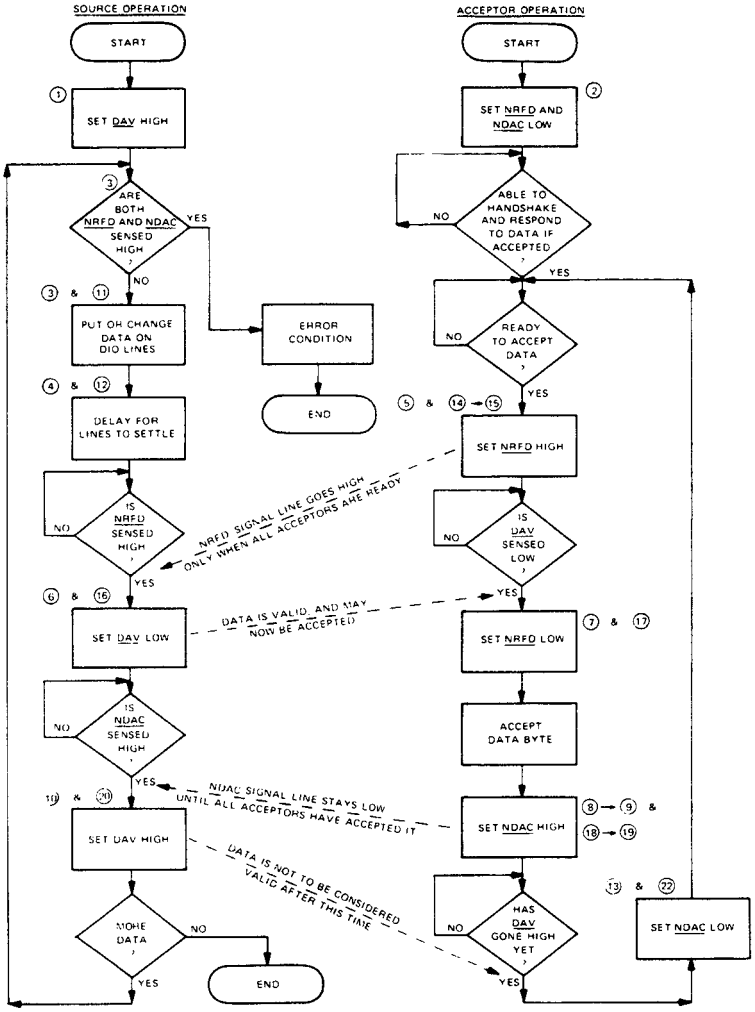


Figure B.2—Logical Flow of Events for Source and Acceptor When Transferring Data Using Handshake Process[7]

[7](See List of Events) (This flow diagram is not intended to represent the only method of implementing an acceptor handshake. See 2.4.5, paragraph three.)

Annex C Interface Function Allowable Subsets

(Informative)

C.1 General Comments

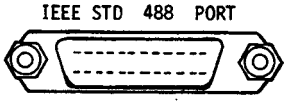
Sections 5 and 6 of this standard point out device designer responsibility to identify and device user responsibility to be familiar with the interface capabilities of each device containing ANSI/IEEE Std 488.1-1987 interface functions. It is therefore recommended that each device be marked with an explicit code which, in effect, identifies those interface functions and subsets implemented within that device.

The reader is further reminded that full operational systems require detailed knowledge of the device dependent characteristics of each device in a system. These specifications are beyond the scope of this standard.

C.2 Capability Identification Codes

It is recommended that an ANSI/IEEE Std 488.1-1987 capability code be placed below or near the interface connector on each device to identify the complete set of interface functions contained within that device. In this standard each interface function and allowable subset thereof has an equivalent alphanumeric code to identify that particular capability. All such interface function capability codes may be expressed in a concise alphanumeric string and marked on the exterior of the device to facilitate user system assembly.

For example, a device with the basic talker function, the ability to send status bytes, the basic listener function, a listen only mode switch, service request capability, remote local capability without local lockout, manual configuration of the parallel poll capability, complete device clear capability, no capability for device trigger, and no controller capability would be identified with the following code:



SH1, AH1, T2, L1, SR1, RL2, PP2, DC1, DT0, C0, E1

The code identifies the eight specific interface functions implemented In addition, the type of electrical interface contained within the device is specified. The notation E1 is used to identify that open collector drivers are used (everywhere there is a choice) and the notation E2 is used to identify that three-state drivers are used (everywhere there is a choice). See 3.3.1.

The device designer can place any further device capability information, useful for system configuration, at the appropriate places on the physical equipment and in the relevant documentation for that equipment.

C.3 SH Function Allowable Subsets

Identfication	Description	States Omitted	Other Requirements	Other Function Subsets Required
SH0	no capability	all	none	none
SH1	complete capability	none	none	T1-T8, TE1-TE8, or C5-C28

C.4 AH Function Allowable Subsets

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
AH0	no capability	all	none	none
AH1	no capability	none	none	none

C.5 T Function Allowable Subsets

Identi-fication	Description				States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities						
	Basic Talker	Serial Poll	Talk Only Mode	Unaddress If MLA			
TØ	N	N	N	N	all	none	none
T1	Y	Y	Y	N	none	omit [MLA ∧ (ACDS)]	SH1 and AH1
T2	Y	Y	N	N	none	omit [MLA ∧ (ACDS)] ton always false	SH1 and AH1
T3	Y	N	Y	N	SPIS, SPMS, SPAS	omit [MLA ∧ (ACDS)]	SH1 and AH1
T4	Y	N	N	N	SPIS, SPMS, SPAS	omit [MLA ∧ (ACDS)]	SH1 and AH1
T5	Y	Y	Y	Y	none	include [MLA ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
T6	Y	Y	N	Y	none	include [MLA ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4
T7	Y	N	Y	Y	SPIS, SPMS, SPAS	include [MLA ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
T8	Y	N	N	Y	SPIS, SPMS, SPAS	include [MLA ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4

C.6 T Function (with Address Extension) Allowable Subsets

Identification	Description				States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities						
	Basic Extended Talker	Serial Poll	Talk Only Mode	Unaddress If MSA ∧ (LPAS)			
TE0	N	N	N	N	all	none	none
TE1	Y	Y	Y	N	none	omit [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and AH1
TE2	Y	Y	N	N	none	omit [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and AH1
TE3	Y	N	Y	N	SPIS, SPMS, SPAS	omit [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and AH1
TE4	Y	N	N	N	SPIS, SPMS, SPAS	omit [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and AH1
TE5	Y	Y	Y	Y	none	include [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
TE6	Y	Y	N	Y	none	include [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4
TE7	Y	N	Y	Y	SPIS, SPMS, SPAS	include [MSA ∧ (LPAS) ∧ (ACDS)]	SH1 and L1-L4 or LE1-LE4
TE8	Y	N	N	Y	SPIS, SPMS, SPAS	include [MSA ∧ (LPAS) ∧ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4

C.7 L Function Allowable Subsets

Identification	Description			States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities					
	Basic Listener	Listen Only Mode	Unaddress If MTA			
LØ	N	N	N	all	none	none
L1	Y	Y	N	none	omit [MTA ∧ (ACDS)]	AH1
L2	Y	N	N	none	omit [MTA ∧ (ACDS)] lon always false	AH1
L3	Y	Y	Y	none	include [MTA ∧ (ACDS)]	AH1 and T1-T8 or TE1-TE8
L4	Y	N	Y	none	include [MTA ∧ (ACDS)] lon always false	AH1 and T1-T8 or TE1-TE8

C.8 L Function (with Address Extension) Allowable Subsets

Identification	Description			States Omitted	Other Requirements	Other Function Subsets Required
	Capabilities					
	Basic Extended Listener	Listen Only Mode	Unaddress If MSA ∧ (TPAS)*			
LEØ	N	N	N	all	none	none
LE1	Y	Y	N	none	omit [MSA ∧ (TPAS) ∧ (ACDS)]	AH1
LE2	Y	N	N	none	omit [MSA ∧ (TPAS) ∧ (ACDS)] lon always false	AH1
LE3	Y	Y	Y	none	include [MSA ∧ (TPAS)] ∧ (ACDS)]	AH1 and T1-T8 or TE1-TE8
LE4	Y	N	Y	none	include [MSA ∧ (TPAS) ∧ (ACDS)] lon always false	AH1 and T1-T8 or TE1-TE8

*Replaced by MTA when used together with the T function.

C.9 SR Function Allowable Subsets

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
SR0	no capability	all	none	none
SR1	complete capability	none	none	T1, T2, T5, T6, TE1, TE2, TE5 or TE6

C.10 RL Function Allowable Subsets

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
RL0	no capability	all	none	none
RL1	complete capability	none	none	L1-L4 or LE1-LE4
RL2	no local lock out	LWLS and RWLS	rtl always false	L1-L4 or LE1-LE4

C.11 PP Function Allowable Subsets

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
PPØ	no capability	all	none	none
PP1	remote configuration	none	include [((PPD ∧ (PACS) ∨ PPU) ∧ (ACDS)] include [PPE ∧ (PACS) ∧ (ACDS)] exclude lpe	L1-L4 or LE1-LE4
PP2	local configuration	PUCS, PACS	include lpe exclude [((PPD ∧ (PACS)) ∨ PPU) ∧ (ACDS)] exclude [PPE ∧ (PACS) ∧ (ACDS)] local messages shall be substituted for S, P1, P2, P3	none

C.12 DC Function Allowable Subsets

Identi-fication	Description	States Omitted	Other Requirements	Other Function Subsets Required
DC0	no capability	all	none	none
DC1	complete capability	none	none	L1-L4 or LE1-LE4
DC2	omit selective device clear	none	omit [SDC ∧ (LADS)]	AH1

C.13 DT Function Allowable Subsets

Identification	Description	States Omitted	Other Requirements	Other Function Subsets Required
DT0	no capability	all	none	none
DT1	complete capability	none	none	L1-L4 or LE1-LE4

C.14 Function Allowable Subsets

Identification*	Capabilities										Notes	States Required									Other Requirements			Other Function Subsets Required			
	System Controller	Send IFC and Take Charge	Send REN	Respond to SRQ	Send I.F. Messages	Receive Control	Pass Control	Pass Control to Self	Parallel Poll	Take Control Synchronously		SNAS, SACS	SIIS, SIAS, SINS	SRIS, SRAS, SRNS	CSNS, CSRS	CACS, CSBS, CSHS, CSWS, CAWS	CADS	CIDS	CTRS	CPWS, CPPS	[TCT ∧ (ACDS) ∧ (TADS)]†	[(TADS)]‡	tcs not always false	C1	C2	AH1, L1-L4, or LE1-LE4	SH1	T1-T8, TE1-TE8
C0	N	N	N	N	N	N	N	N	N	N		O	O	O	O	O	O	O	O	O	O	O	O	O	—	—	—	—
C1	Y	—	—	—	—	—	—	—	—	—	(1)	R	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
C2	—	Y	—	—	—	—	—	—	—	—	(1),(6)	—	R	—	—	—	—	—	—	—	—	—	—	R	—	—	—	—
C3	—	—	Y	—	—	—	—	—	—	—	(1)	—	—	R	—	—	—	—	—	—	—	—	—	R	—	—	—	—
C4	—	—	—	Y	—	—	—	—	—	—	(1)	—	—	—	R	—	—	—	—	—	—	—	—	—	—	—	—	—
C5	—	—	—	—	Y	Y	Y	Y	Y	Y	(2),(3)	—	—	—	—	R	R	R	R	R	R	R	R	—	—	R	R	R
C6	—	—	—	—	Y	Y	Y	Y	Y	N	(2),(3)	—	—	—	—	R	R	R	R	R	R	R	O	—	—	—	R	R
C7	—	—	—	—	Y	Y	Y	Y	N	Y	(2),(3)	—	—	—	—	R	R	R	R	O	R	R	R	—	—	R	R	R
C8	—	—	—	—	Y	Y	Y	Y	N	N	(2),(3)	—	—	—	—	R	R	R	R	O	R	R	O	—	—	—	R	R
C9	—	—	—	—	Y	Y	Y	N	Y	Y	(2),(3)	—	—	—	—	R	R	R	R	R	R	O	R	—	—	R	R	R
C10	—	—	—	—	Y	Y	Y	N	Y	N	(2),(3)	—	—	—	—	R	R	R	R	R	R	O	O	—	—	—	R	R
C11	—	—	—	—	Y	Y	Y	N	N	Y	(2),(3)	—	—	—	—	R	R	R	R	O	R	O	R	—	—	R	R	R
C12	—	—	—	—	Y	Y	Y	N	N	N	(2),(3)	—	—	—	—	R	R	R	R	O	R	O	O	—	—	—	R	R
C13	—	—	—	—	Y	Y	N	N	Y	Y	(2)	—	—	—	—	R	R	R	O	R	O	O	R	—	R	R	R	—
C14	—	—	—	—	Y	Y	N	N	Y	N	(2)	—	—	—	—	R	R	R	O	R	O	O	O	—	R	—	R	—
C15	—	—	—	—	Y	Y	N	N	N	Y	(2)	—	—	—	—	R	R	R	O	O	O	O	O	—	R	R	R	—
C16	—	—	—	—	Y	Y	N	N	N	N	(2)	—	—	—	—	R	R	R	O	O	O	O	O	—	R	—	R	—
C17	—	—	—	—	Y	N	Y	Y	Y	Y	(2),(3),(4)	—	—	—	—	R	O	R	R	R	R	R	R	—	—	R	R	R
C18	—	—	—	—	Y	N	Y	Y	Y	N	(2),(3),(4)	—	—	—	—	R	O	R	R	R	R	R	O	—	—	—	R	R
C19	—	—	—	—	Y	N	Y	Y	N	Y	(2),(3),(4)	—	—	—	—	R	O	R	R	O	R	R	R	—	—	R	R	R
C20	—	—	—	—	Y	N	Y	Y	N	N	(2),(3),(4)	—	—	—	—	R	O	R	R	O	R	R	O	—	—	—	R	R
C21	—	—	—	—	Y	N	Y	N	Y	Y	(2),(3),(4)	—	—	—	—	R	O	R	R	R	R	O	R	—	—	R	R	R
C22	—	—	—	—	Y	N	Y	N	Y	N	(2),(3),(4)	—	—	—	—	R	O	R	R	R	R	O	O	—	—	—	R	R
C23	—	—	—	—	Y	N	Y	N	N	Y	(2),(3),(4)	—	—	—	—	R	O	R	R	O	R	O	R	—	—	R	R	R
C24	—	—	—	—	Y	N	Y	N	N	N	(2),(3),(4)	—	—	—	—	R	O	R	R	O	R	O	O	—	—	—	R	R
C25	—	—	—	—	Y	N	N	N	Y	Y	(2),(5)	—	—	—	—	R	O	O	O	R	O	O	R	—	—	R	R	—
C26	—	—	—	—	Y	N	N	N	Y	N	(2),(5)	—	—	—	—	R	O	O	O	R	O	O	O	—	—	—	R	R
C27	—	—	—	—	Y	N	N	N	N	Y	(2),(5)	—	—	—	—	R	O	O	O	O	O	O	R	—	—	R	R	—
C28	—	—	—	—	Y	N	N	N	N	N	(2),(5)	—	—	—	—	R	O	O	O	O	O	O	O	—	—	—	R	—

*Typical notation to describe a controller consists of the letter C followed by one or more of the numbers indicating the subsets selected. For example; C1, 2, 3, 4, 8.

†This is part of the CIDS to CADS transitional expression.

‡This is part of the CACS to CTRS transitional expression.

NOTES:

(1) One or more of subsets C1 through C4 may be chosen in any combination with any one of C5 through C28.

(2) Only one subset may be chosen from C5 through C28.

(3) The CTRS state must be included in devices which are to be operated in multicontroller systems.

(4) These subsets are not allowed unless C2 is included.

(5) These subsets are intended to be used in devices and systems where no control passage is possible.

(6) When a system controller asserts IFC during the time another physical device is operating as controller-in-charge, the system controller should refrain from active assertion of the source handshake and ATN until the removal of the IFC message to preclude multiple controller contention.

O = omit, R = required, hyphen = not applicable or not required, Y = yes, N = no.

Annex D Interface Message Reference List

(Informative)

Mnemonic	Message	Interface Function(s)
LOCAL MESSAGES RECEIVED (by interface functions)		
gts	go to standby	C
ist	individual status qualifier	PP
Ion	listen only	L, LE
[lpe]	local poll enable	PP
ltn	listen	L, LE
lun	local unlisten	L, LE
nba	new byte available	SH
pon	power on	SH, AH, T, TE, L, LE, SR, RL, PP, C
rdy	ready	AH
rpp	request parallel poll	C
rsc	request system control	C
rsv	request service	SR
rtl	return to local	RL
sic	send interface clear	C
sre	send remote enable	C
tca	take control asynchronously	C
tcs	take control synchronously	AH, C
ton	talk only	T, TE
LOCAL MESSAGES SENT (to device functions)		
	None defined; see Message Output tables in Section 2 for description of Device Function Interaction which provides guidelines as to the appropriate states from which local messages may be sent to the device functions.	
REMOTE MESSAGES RECEIVED		
ATN	attention	SH, AH, T, TE, L, LE, PP, C
DAB	data byte	(via L, LE)
DAC	data accepted	SH
DAV	data valid	AH
DCL	device clear	DC

Mnemonic	Message	Interface Function(s)
END	end	(via L, LE)
GET	group execute trigger	DT
GTL	go to local	RL
IDY	identify	L, LE, PP
IFC	interface clear	T, TE, L, LE, C
LLO	local lockout	RL
MLA	my listen address	L, LE, RL
[MLA]	my listen address	T
MSA or [MSA]	my secondary address	TE, LE
MTA	my talk address	T, TE
[MTA]	my talk address	L
OSA	other secondary address	TE
OTA	other talk address	T, TE
PCG	primary command group	TE, LE, PP
PPC	parallel poll configure	PP
[PPD]	parallel poll disable	PP
[PPE]	parallel poll enable	PP
PPRn	parallel poll response n	(via C)
PPU	parallel poll unconfigure	PP
REN	remote enable	RL
RFD	ready for data	SH
RQS	request service	(via L, LE)
[SDC]	selected device Clear	DC
SPD	serial poll disable	T, TE
SPE	serial poll enable	T, TE
SRQ	service request	(via C)
STB	status byte	(via L, LE)
TCT or [TCT]	take control	C
UNL	unlisten	L, LE
REMOTE MESSAGES SENT		
ATN	attention	C
DAB	data byte	(via T, TE)
DAC	data accepted	AH
DAV	data valid	SH
DCL	device clear	(via C)

Mnemonic	Message	Interface Function(s)
END	end	(via T)
GET	group execute trigger	(via C)
GTL	go to local	(via C)
IDY	identify	C
IFC	interface clear	C
LLO	local lockout	(via C)
MLA or [MLA]	my listen address	(via C)
MSA or [MSA]	my secondary address	(via C)
MTA or [MTA]	my talk address	(via C)
OSA	other secondary address	(via C)
OTA	other talk address	(via C)
PCG	primary command group	(via C)
PPC	parallel poll configure	(via C)
[PPD]	parallel poll disable	(via C)
[PPE]	parallel poll enable	(via C)
PPRn	parallel poll response n	PP
PPU	parallel poll unconfigure	(via C)
REN	remote enable	C
RFD	ready for data	AH
RQS	request service	T, TE
[SDC]	selected device clear	(via C)
SPD	serial poll disable	(via C)
SPE	serial poll enable	(via C)
SRQ	service request	SR
STB	status byte	(via T, TE)
TCT	take control	(via C)
UNL	unlisten	(via C)
UNT	untalk	(via C)

Annex E Multiline Interface Messages: ISO Code Representation

(Informative)

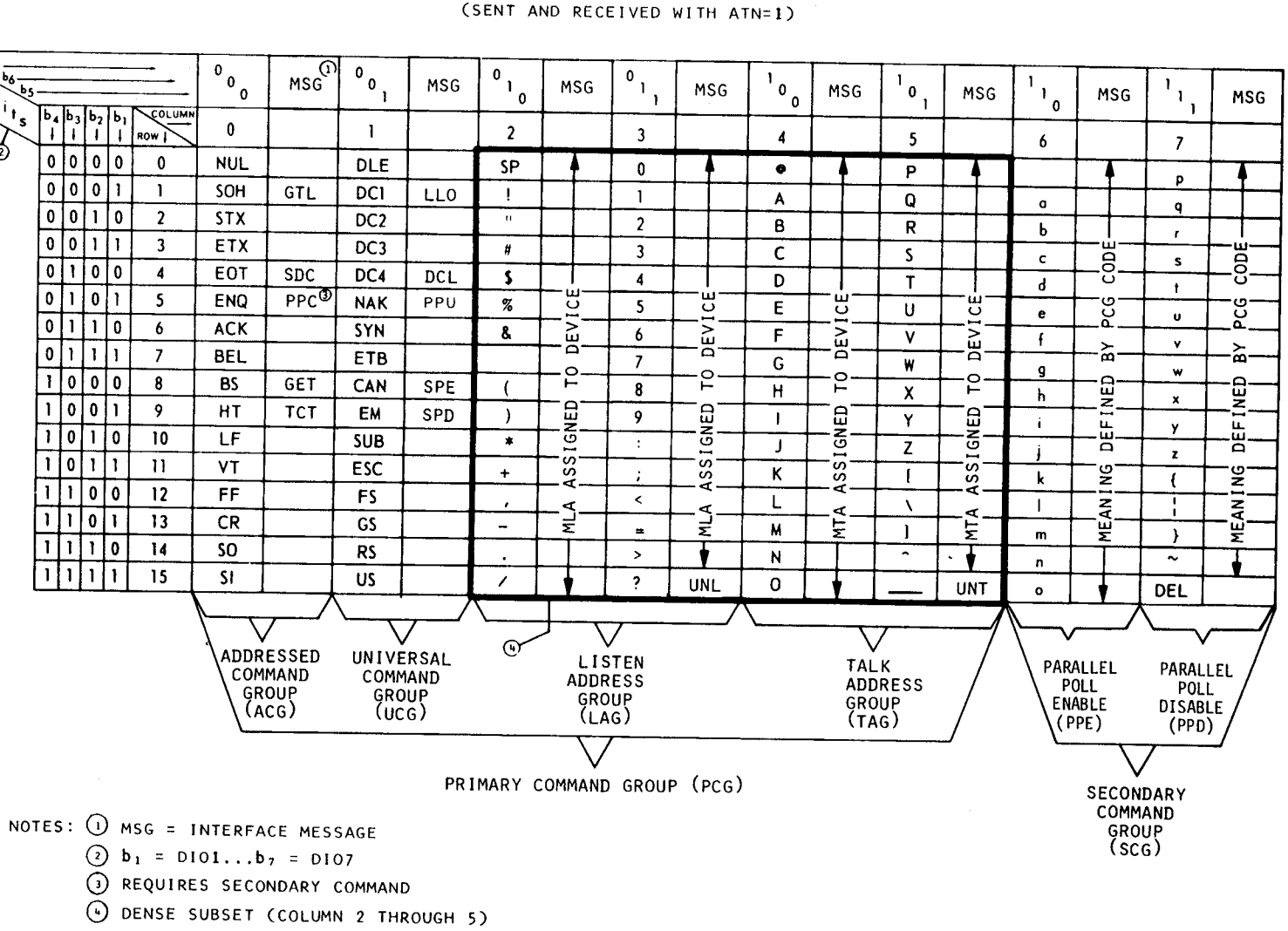


Figure E.1—Multiline Interface Messages: ISO 7 -Bit Code Representation

Annex F Logic Circuit Implementation

(Informative)

To assist the designer in the interpretation of the state diagrams, possible circuit implementations are given for situations occurring within the interface functions. It must be understood that the logic diagrams given in this appendix do not show the only implementations possible nor do they even represent recommended implementations. They are for educational purposes only.

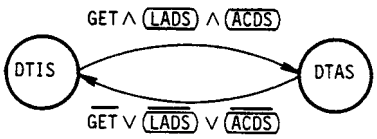
State diagrams are used to represent two concepts:

1) They allow differentiation between the different responses an interface function might produce and identify each with one or more unique states of the interface function.
2) They identify those situations where an interface function is required to remember past events in order to produce the correct response.

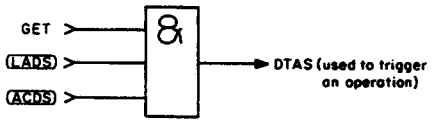
Each state in any of the diagrams serves either or both of these purposes. For example, the LADS of the L interface function has no unique response associated with it and cannot be distinguished from LIDS. Its purpose, however, is to remember that the device has received a listen address over the bus and is therefore able to enter LACS when the ATN message is received false (see Fig 7). Conversely, the LACS is an example of one which has no memory with it but which exists as a distinct state only to show a specific response capability which does not exist during LADS. The only internal difference between the two states is the value of the ATN message, and no memory is required since this message value is continuously available.

F.1 Implementing States Which Require No Memory

The DT interface function is an example of a complete interface function which requires no memory. Its state diagram (identical to Fig 13) is the following:

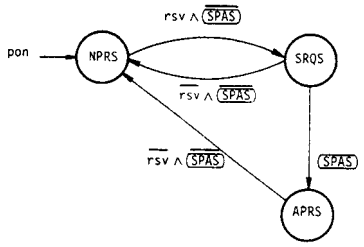


Since no memory is required, this interface function can be implemented with a single AND gate:



F.2 Implementing States Which Require Memory

The SR interface function is an example of one requiring memory for its states. Its state diagram (identical to Fig 9) is the following:



The two top states taken by themselves represent a circuit whose internal state follows the value of the rsv message but only if SPAS is false. Figure F.1 is a standard DC Flip-Flop.

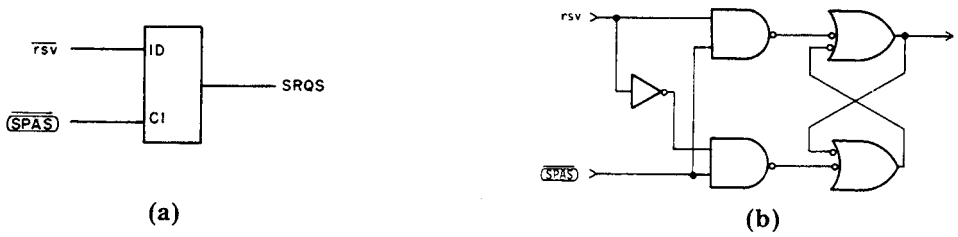


Figure F.1—Partial SR Function — (a) Composite Logic; (b) Gate Level Logic

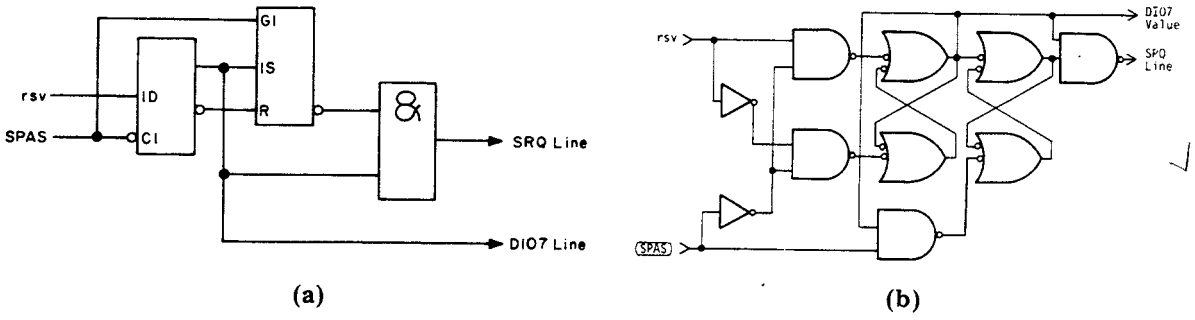


Figure F.2—Full SR Function — (a) Composite Logic; (b) Gate Level Logic

To complete the circuit all that is needed is a memory that (SPAS) has occured after the latch has turned on. This circuit can be built around a standard RS flip-flop and added to the latch to produce Fig F2.

In this circuit, the RS flip-flop output stage is forced clear whenever the value of the latched rsv message is false. When the latched rsv message becomes true, it remains cleared until the first time SPAS becomes active at which time it sets, remembering that an RQS message has been sent and SRQ no longer needs to be held true.

Annex G Parallel Polling Sequence

(Informative)

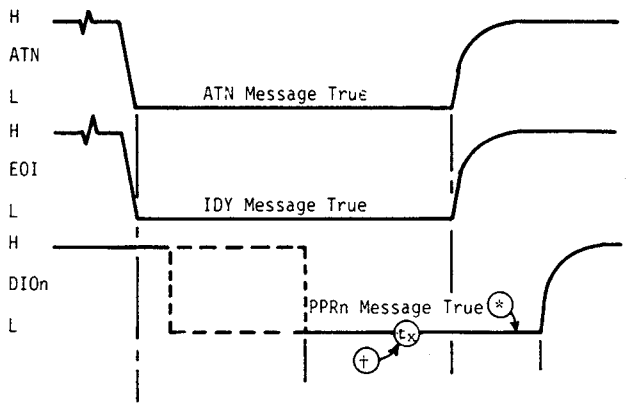


Figure G.1—Parallel Poll Sequence: Signal Line Waveforms

*PPRn message true shown in one of two alternate states as determined by the PPE message.

†Strobe of DIOn lines occurs internal to the controller at any time during the CPPS state by a designer defined method (the transfer of status data bits during parallel poll does not utilize the handshake process).

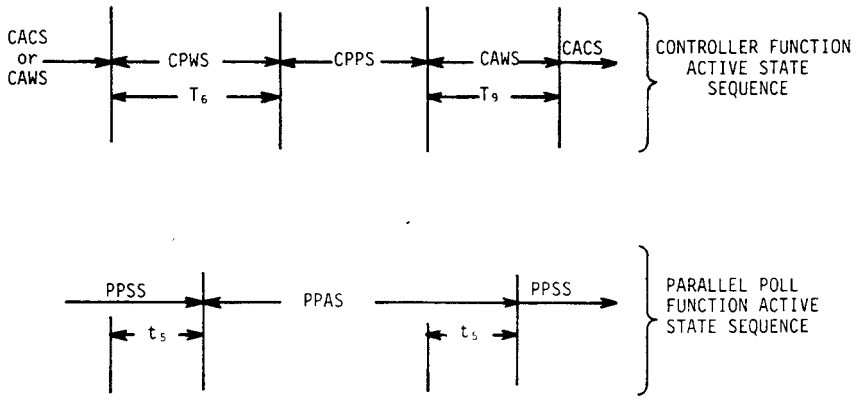


Figure G.2—Interface Function Active States During Parallel Poll

Annex H Description of Interface Parameters on Data Sheets

(Informative)

This set of guidelines is intended to facilitate the preparation of documentation, particularly data sheets, related to products meeting the requirements of this standard. A product meeting the requirements of this standard may also have additional device function capability; reference to such capabilities which are accessible via the interface port of this standard may be useful to instrumentation systems users.

This appendix forms the basis of a CHECKLIST for both designer and user to describe important interface related parameters, hence not all parameters listed are expected to be used in every product. The parameters listed are intended as a general guideline and should not be regarded as exhaustive. The amount of material that can be provided on any data sheet depends on space available and the major purpose for which it is intended (for example, detailed technical data sheet, general descriptive brochure, abbreviated listing in operating manuals).

H.1 General

It is recommended that the data sheet for an instrument or device meeting the requirements of this standard contain information which will enable the user of that device to analyze its general capability, programmability, and system performance (relative to the interface). Similarly, it is recommended that the manual supplied with the instrument provide a more detailed description of the interface related capabilities to facilitate the configuration of the instrumentation systems. Sections 5.1 and 6 of this standard require the device designer to identify, and the device user to be familiar with, the interface capabilities.

It should be noted that for full operational systems it is necessary to have detailed knowledge of the device-dependent characteristics of each device in a system (such detailed knowledge may require information additional to that contained in this standard) therefore, compatibility, in the total sense, is not necessarily obtained even though the requirements of this standard have been met. In addition to the following recommendations, the designer should include, with the user information, details about the programmable device-dependent functions of the instrument.

H.2 Description of Interface Function Capabilities

It is recommended that data sheets indicate, in symbolic form, the set of interface functions provided by the subject device. Short descriptive phrases may be useful where space permits. Not all interface functions need be included in a product, in which case “no capability” may be expressed by the function mnemonic and the number 0 (for example, C0).

		Reference	
Function	**Symbol**	**Section**	**Table**
Source Handshake	SH	2.3	4
Acceptor Handshake	AH	2.4	7
Talker	T	2.5	11
Extended Talker	TE	2.5	12
Listener	L	2.6	16
Extended Listener	LE	2.6	17
Service Request	SR	2.7	20
Remote Local	RL	2.8	23
Parallel Poll	PP	2.9	27
Device Clear	DC	2.10	30
Device Trigger	DT	2.11	33
Controller	C	2.12	37

H.3 Electrical Driver/Receiver Capabilities

H.3.1

Signal lines with open collector drivers (everywhere there is a choice) use E1 (specify applicable DIO lines and Constraints).

H.3.2

Signal lines with three-state drivers (everywhere there is a choice) use E2 (specify applicable DIO lines and constraints).

H.4 Additional Information of Value to Systems Users and Designers

H.4.1 Functional Specifications

Any operating modes that deviate from the 488 standards should be explicitly stated and explained.

The [IEEE-488 documents to which the device conforms should be listed full title with information on how to obtain such standards.

The means of generating and using the R/L function and the rtl local message should be documented.

The response to Group Execute Trigger should be documented.

H.4.2 Controls, Connectors and Indicators

All IEEE-488 related switches should be illustrated or their locations described. In addition, the following information should be documented:

1) How to set primary and secondary (if any) addresses.
2) How to read addresses from the front panel (if possible).
3) Effects of other IEEE-488 alterable modes (such as ton, Ion, etc).

H.4.3 Power Up/Down Sequences and Default Values

1) Describe pon serf-test functionality and abnormal condition reporting.
2) Describe the effect of power-interrupt.
3) Describe nonvolatile memory features.
4) Describe default device state at pon.

H.4.4 Programmable Device Functions

List which device functions are bus controllable. Describe I/O buffering (if any), for example:

1) Buffer sizes
2) Maximum lines accepted,
3) Maximum number of digits.

List limits of numerical parameters, for example:

1) Mantissa and exponent limits,
2) Internal precision for rounding and rounding rules,
3) Limits on parameters and effect of going beyond.

H.4.5 Status Handling Information

Identify if any rsv local messages may be inhibited. List all conditions that may cause the device to set rsv TRUE.

H.5 Description of Typical Time Related Values

The description of time related values in this section is highly dependent upon the total system configuration (that is, the nature of the talker, listener, and controller devices). Actual time values may be highly dependent upon the actual measurement conditions, the device-dependent nature of the instrument being specified, and possibly the operating system software located in either instrument or controller. Therefore, it is difficult and possibly irrelevant to specify precise values.

H.5.1 Data rates for DAB messages

1) Identify data input rate (when addressed to listen). For example, N kilobytes per second, and relevant conditions (for example, configuration, data type, functional operation).
2) Identify data output rate (when addressed to talk). For example, N kilobytes per second and relevant conditions (for example, configuration, data type, functional operation).

H.5.2 Identify other time values relating to system performance

For Example:

1) Interface handshake delays (for example, time out, hold),
2) Time to respond to device commands,
3) Time to respond to interface messages.

Annex I Address Switch Labeling and Interface Status Indicators

(Informative)

I.1 General Comments

To assist a device designer, a recommended positioning and labeling for the talk and listen address switch is given. A recommendation is also given for the labeling of interface status indicators or the interface status messages to be used on displays.

I.2 Talk and Listen Addresses

A device should have a nonvolatile local means of preselecting the values assigned to its my talk address (MTA) and its my listen-address (MLA). The value for bits T1 through T5 assigned to MTA or bits L1 through L5 assigned to MLA should be capable of being altered by the device operator. If the MTA and MLA are identical, (bits 1-5, as in 6.3.1 and 6.3.2), then a single common storage element may be used. This discussion will use this single address value but does not preclude the use of separate addresses for MTA and MLA.

I.3 DIP Switch

When the common MTA and MLA, (bits 1-5), address value is selected by a DIP switch, the DIP switch needs 5 bits (combination of T1 through T5 and L1 through L5) or switch poles to set the address value. The 5 switch poles should be physically adjacent to each other as indicated in Fig Il or Fig I.2. If the local messages ton, or lon, or both are implemented as poles on the same DIP switch, they should be placed as shown in Fig I.1 or Fig I.2. In the case that ton is not implemented but Ion is, the Ion switch should be immediately to the left or above the address switches.

Where feasible, the DIP switch should be externally accessible and labeled similar to Fig I.1 or Fig I.2. The description of the switch and its location should be clearly visible in the device's operation manual.

Labeling associated with the DIP switch is to include the binary weighting applied to each DIP switch position to facilitate determination of the address value. An illustration of this is shown in Fig I.1 and Fig I.2. An indication of which switches are used to set the address should also be provided. Diagrams in the instruction manual should be provided to clearly indicate which physical position of the individual DIP switches provides an “0/I” condition. The “1” position for the switches is to present a logical TRUE state on the IEEE Std 488 interface bus while the “0” position is to present a logical FALSE state on the bus. The labeled “X” positions show a sample selection of the switches.

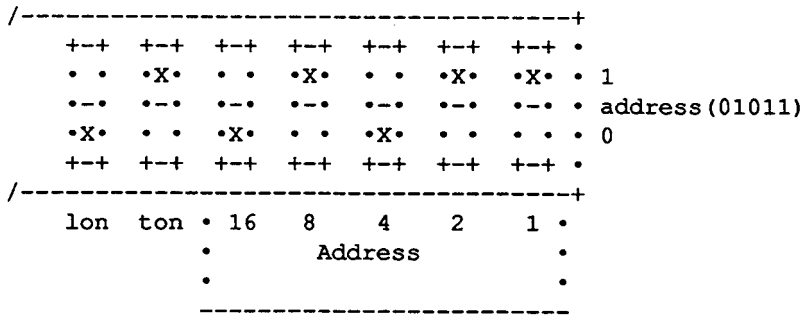


Figure I.1—Labeling Diagram for DIP Switch In Horizontal Orientation

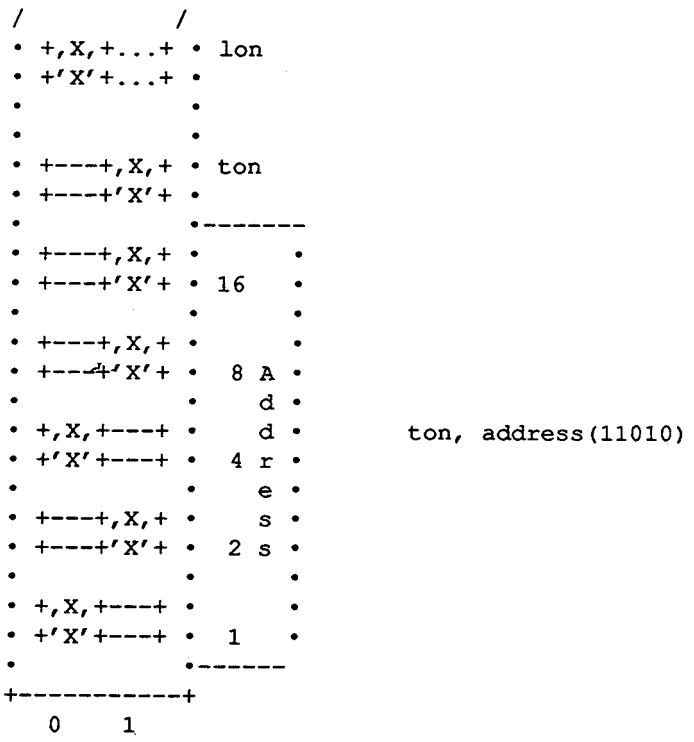


Figure I.2—Labeling Diagram for DIP Switch In Vertical Orientation

I.4 Alternate Implementations

Other methods besides DIP Switches can be used to set the nonvolatile value of the MTA and MLA address or change the state of ton and on local messages. This alternative method, however, should provide the ability of displaying and modifying the address values. When ton and Ion are implemented, this alternative method should also allow their state to be examined and modified.

I.5 Device Status Indicators

A device may optionally contain indicators, or displays, or both showing the current state (or operating mode) of the device interface. When these interface indicators or displays are provided, they should be labeled with or use the abbreviation or name given in the following table. The NDAC, NRFD, and SRQ indicators are not to represent the state on the interface bus, but the state of the device's remote message. An ADDR indicator may be used in place of the TALK and LSTN indicators.

Abbreviation	Name	Current State
ADDR	Addressed to Talk or Listen	Device is in TADS, TACS, LADS, or LACS
LOCK	Front Panel Lockout	Device is in RWLS
LSTN	Addressed to Listen	Device is in LADS or LACS
MA	My Address	Current Address Value
NDAC	Not Data Accepted	NDAC Asserted
NRFD	Not Ready For Data	NRFD Asserted
REM	Remote	Device is in REMS or RWLS
SRQ	Service Request	SRQ Asserted
TALK	Addressed to Talk	Device is in TADS or TACS
CACT	Controller Active	Device is in CTRS, CACS, CSBS, CSHS, CSWS, CAWS, CPWS or CPPS

Annex J Recommended Methods for Reducing the Effects of Radiated and Conducted Interference for Devices Specified in this Standard (Informative)

This Standard specifies a connector and cable assembly that provides acceptable ElectroMagnetic Compatibility (EMC) performance under many circumstances. In some applications, however, it is either advisable or required to further reduce the effects of EM interference. The recommended methods provide general guidance on precautions that can be taken to minimize the effects of conducted or radiated EM interference on instrumentation systems to levels below those which would otherwise be obtained by complying with the specifications given elsewhere in this standard.

It is generally accepted that the cable assembly (that is, the interconnecting cable and the associated connectors) and the connector/cable interface immediately inside the instrument will, to a large extent, determine the EMC performance of the overall instrumentation system.

The following guidelines describe how to connect instruments fitted with connectors specified in this standard to obtain improved EMC performance. It should be noted that it may be necessary to reduce the effects of four basic types of interference:

1) Leakage from cables and connectors that affects other RF systems.
2) External interference affecting cables and connectors, and consequently data transfer as described in this standard.
3) Injection of signals between any signal ground return, logic ground and shield.
4) External interference which may be injected between any signal ground return, logic ground and shield.

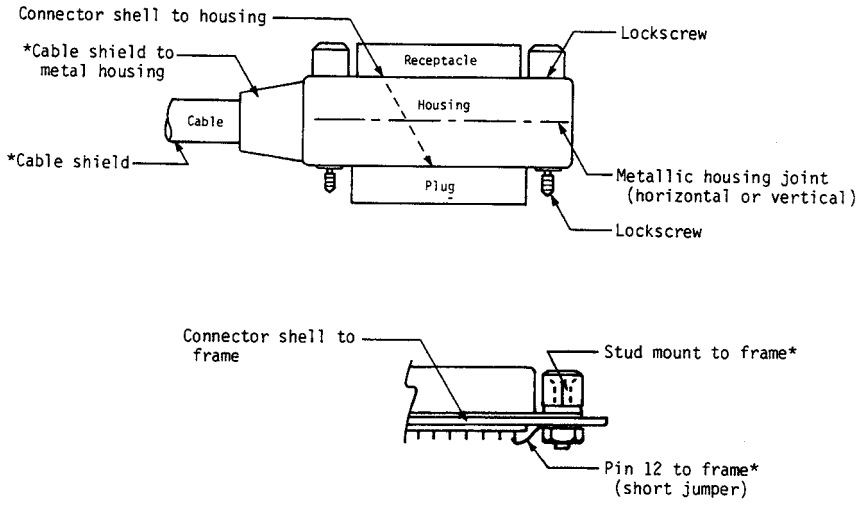
J.1 Reducing the Effect of Radiated Interference

These goals can be accomplished by using cabling techniques with minimum leakage.

The basic recommendations detailed below and illustrated in Fig J.1 may be of assistance in meeting the screening requirements applicable to instrumentation systems which might be imposed in some countries by their National Regulations.

J.1.1 Screened Cables

A minimum coverage of 85% is essential, but a 90% coverage would provide a further improvement. A combination of both braid and metallized mylar or foil provides still further improvement in cable screening and is recommended.



NOTES: (1) All surfaces pointed to are metallic.
(2) Good quality electrical contact at all points recommended.
(3) Most critical conductive points, (*) where good RF practice is essential.

Figure J.1—RF Contact (Ground) Points

J.1.2 RF Connection Techniques

It is insufficient to rely only on ground connection between the cable screen and the connector on the equipment through one or even several connector contacts. Instead, an adequate RF connection to chassis, for example through the metallic connector housing, is required. Care should be observed to provide backward compatibility (mechanical and electrical) with the already existing shapes of connector housing.

J.1.3 Connector Housing

The connector housing should be metallic. Since the housing is to be used for chassis connection, the parts of the housing that provide the electrical contact should be designed and finished accordingly (for example, metallic and conductive housing, metal shelled connectors or equivalent, good RF connection between cable screen and metallic housing, etc).

J.1.4 EM Radiation Screening

EM radiation screening can be further improved by providing a metallic cover over the ends of the “piggyback” connectors. This measure will, however, only be necessary in exceptional cases. The contribution to RF leakage by the connector contacts is small.

J.2 Connectors Conforming to this Standard Used On Equipment

J.2.1

The IEEE-488 connector port should have a metal shell and ensure adequate RF contact with the cable connector housing (in accordance with J1.3).

J.2.2

The electrical connection between the “shield” contact of the connector, the metal housing and shell of the chassis mounted connector, and chassis frame should be made via as short a route as possible. Noticeable performance degradation may occur with ground lengths in excess of about 50mm. Therefore, when using a flat cable to make the internal connections to the IEEE-488 connector port, the shield should not be connected to chassis via this cable.

J.3 Reducing the Effects of Conducted Interference

Signals which exist on the signal ground lines can cause conducted electromagnetic interference inside a device when ground loops are formed as the signal ground lines are connected together inside the device.

Such ground loops can also create Electromagnetic Interference, EMI, as signals are generated by the effect of RF fields on such ground loops.

J.3.1 Generation of Injected Signals on Signal Ground Lines

It is recommended that a device minimize generation of conducted interference among the logic ground and earth conductors, (that is, 12, 18, 19, 20, 21, 22, 23, 24), by using low impedance rf ground returns as close to the connector port as possible within the device.

J.3.2 Protection Against Conducted EMI

It is recommended that a device minimize its susceptibility to conducted interference among the logic ground and earth conductors, (that is, 12, 18, 19, 20, 21, 22, 23, 24), by using low impedance rf ground returns as close to the connector port as possible within the device.