

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

© Copyright 1988 by

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

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

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

IEEE Standards documents are adopted by the Institute of Electrical and Electronics Engineers without regard to whether their adoption may involve patents on articles, materials, or processes. Such adoption does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the standards documents.

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 I Appendix 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*
Bob Cram*
Stephen Greer
Damon Hart

Jeffrey Kodosky
William Maciejewski*
Tom Leedy
Glen Meldrum*

Dana Trout*
Don Ware*

*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
John Barker
Steve Barryte
Richard Day
Ron Doss
Richard Drews
Gary Gallagher
Bernard Gollomp
Arnie Greenspan
Bill Gustafson

Larry Gross
Carl Hagerling
Chris Hancock
Faisal Imdad
B. Kowaluk
Robert Kurkjian
Thomas Leedy
Fred Liguri
Don Loughry
John McGlaughlin

Jerry Merritt
L F. Moebus
Charles Osborn
Larry Ross
Eric Sacher
Milton Slade
J. R. Weger
Jim Weitenhagen
D. Williamson

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
1. General	1
1.1 Scope	1
1.2 Object	2
1.3 Definitions	2
1.4 Interface System Overview	3
1.5 References	6
2. Functional Specifications	7
2.1 Functional Partition	7
2.2 Notation Used to Specify Interface Functions	11
2.3 Source Handshake (SH) Interface Function	13
2.4 Acceptor Handshake (AH) Interface Function	17
2.5 Talker (T) Interface Function (Includes Serial Poll Capabilities)	21
2.6 Listener (L) Interface Function	26
2.7 Service Request (SR) Interface Function	32
2.8 Remote Local (RL) Interface Function	34
2.9 Parallel Poll (PP) Interface Function	37
2.10 Device Clear (DC) Interface Function	41
2.11 Device Trigger (DT) Interface Function	43
2.12 Controller (C) Interface Function	45
2.13 Remote Message Coding and Transfer	54
3. Electrical Specifications	61
3.1 Application	61
3.2 Logical and Electrical State Relationships	62
3.3 Driver Requirements	62
3.4 Receiver Requirements	63
3.5 Composite Device Load Requirements	63
3.6 Ground Requirements	65
3.7 Cable Characteristics	65
3.8 State Transition Timing Values	66
4. Mechanical Specifications	67
4.1 Application	67
4.2 Connector Type	67
4.3 Connector Contact Assignments	69
4.4 Device Connector Mounting	69
4.5 Cable Assembly	70
5. System Applications and Guidelines for the Designer	72
5.1 System Compatibility	72
5.2 Data Rate Consideration	72
5.3 Device Capabilities	73
5.4 AND and OR Logic Operations	74
5.5 Address Assignment	76
5.6 Typical Combinations of Interface Functions	77
5.7 Unimplemented Interface Message Handling	77

CLAUSE	PAGE
6. System Requirements and Guidelines for the User.....	77
6.1 System Compatibility.....	77
6.2 System Installation Requirements.....	77
6.3 Address Assignment	78
6.4 Cabling Restrictions	79
6.5 Operational Sequence Guidelines	80
Annex A Typical Instrument System (Informative)	83
Annex B Handshake Process Timing Sequence (Informative).....	85
Annex C Interface Function Allowable Subsets (Informative).....	88
Annex D Interface Message Reference List (Informative).....	95
Annex E Multiline Interface Messages: ISO Code Representation	98
Annex F Logic Circuit Implementation (Informative).....	99
Annex G Parallel Polling Sequence (Informative).....	101
Annex H Description of Interface Parameters on Data Sheets (Informative).....	102
Annex I Address Switch Labeling and Interface Status Indicators (Informative).....	106
Annex J Recommended Methods for Reducing the Effects of Radiated and Conducted Interference for Devices Specified in this Standard (Informative)	109

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

¹Number 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 simultaneously
- 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 arbitrary 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.

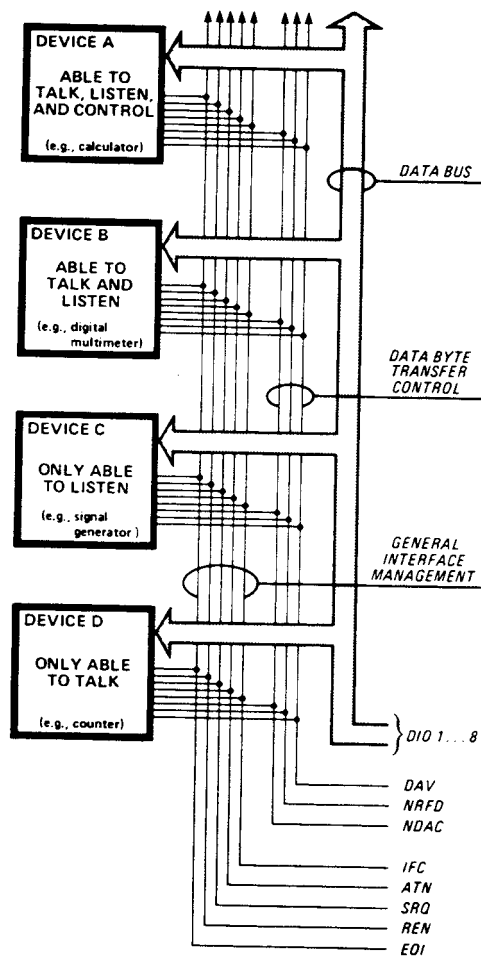


Figure 1—Interface Capabilities and Bus Structure

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

²ANSI 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.

³IEC 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 Varembe, Case postale [3], [2] 1 Geneve 20, Switzerland/Suisse.

⁴MIL publications can be obtained from the Naval Publications Forms Center, 5801 Tabor Avenue, Philadelphia, PA 19120-5099.

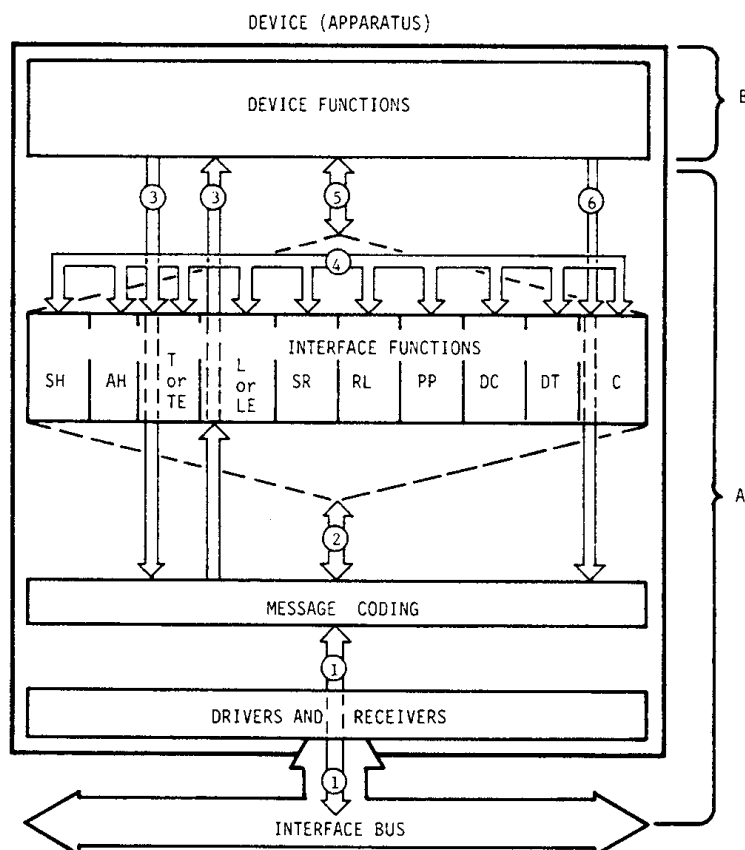
For each state of an interface function definitions are given for the following:

- 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 listener	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 circuit 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

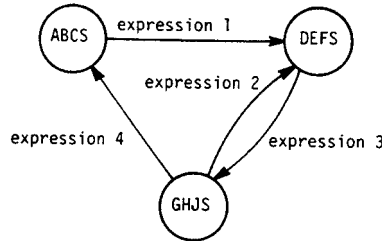
2.2.1 State Diagram Notation

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 of 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 $\vee$.

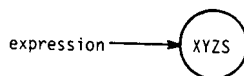
The AND operator takes precedence over the OR operator within an expression unless otherwise specified by parentheses.

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.

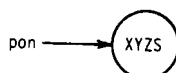
If a portion of an expression is optional in that its true value is not required for the complete expression to be true (at the designer's choice), then it is enclosed within square brackets [...].

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):

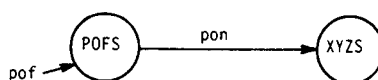


Although power-off (POFS) is a valid state of most interface functions and should normally be shown on all diagrams with a transition leading to the state to be entered at power-on time, a shorthand form is used showing the pon pseudomessage originating a transition to the first state to be entered when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

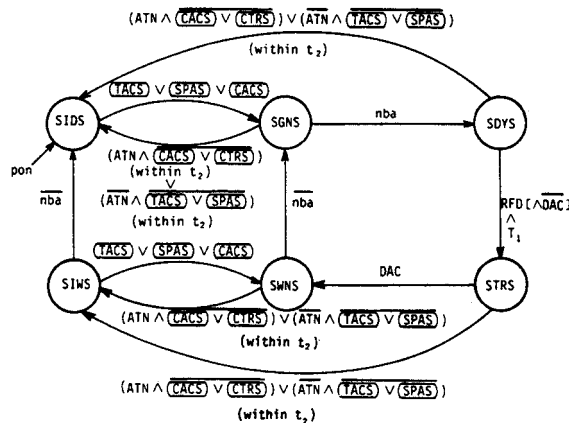


Figure 3—SH State Diagram

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
		<u>TACS</u>	= talker active state (T function)
		<u>SPAS</u>	= serial poll active state (T function)
		<u>CACS</u>	= controller active state (C function)
		<u>CTRS</u>	= 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.
- 2) The source wait for new cycle state (SWNS) if the DAC message is true

NOTE — This implies an asynchronous interrupt (see 2.1.2.5).

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 $(\overline{ATN} \wedge (\overline{CACS} \vee \overline{CTRS}) \vee (\overline{ATN} \wedge (\overline{TACS} \vee \overline{SPAS}))$ 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.

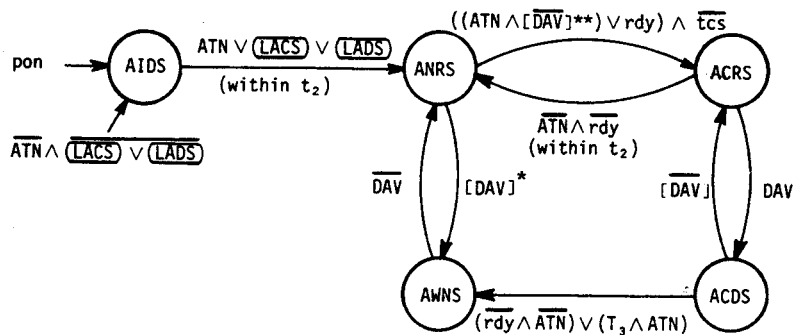
2.4 Acceptor Handshake (AH) Interface Function

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 μ s, 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 shall 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 is 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 shall 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.

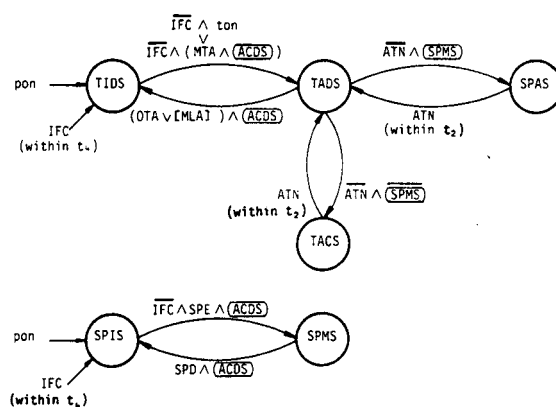


Figure 5—T State Diagram

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	

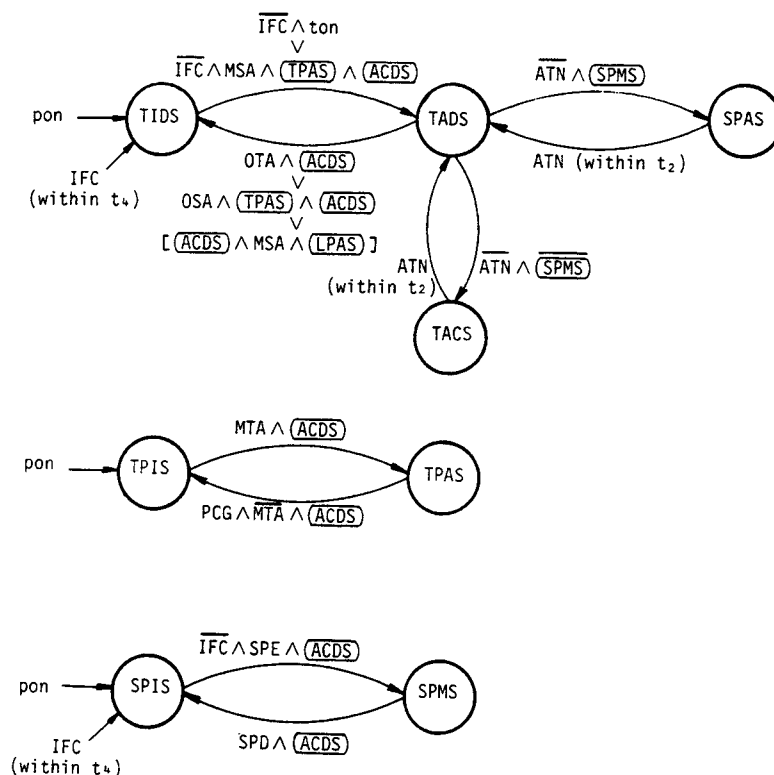
Table 9—T or TE Message Outputs

T State	Qualifier	Remote Messages Sent*			Device Function (DF) Interaction
		Multiline	END	RQS [†]	
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 $[MSA \wedge (ACDS) \wedge (LPAS)]$ shall be replaced by $[MLA \wedge (ACDS)]$.

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 $\wedge$ (ACDS)]	SH1 and AH1
T2	Y	Y	N	N	none	omit [MLA $\wedge$ (ACDS)] ton always false	SH1 and AH1
T3	Y	N	Y	N	SPIS, SPMS, SPAS	omit [MLA $\wedge$ (ACDS)]	SH1 and AH1
T4	Y	N	N	N	SPIS, SPMS, SPAS	omit [MLA $\wedge$ (ACDS)] ton always false	SH1 and AH1
T5	Y	Y	Y	Y	none	include [MLA $\wedge$ (ACDS)]	SH1 and L1-L4 or LE1-LE4
T6	Y	Y	N	Y	none	include [MLA $\wedge$ (ACDS)] ton always false	SH1 and L1-L4 or LE1-LE4
T7	Y	N	Y	Y	SPIS, SPMS, SPAS	include [MLA $\wedge$ (ACDS)]	SH1 and L1-L4 or LE1-LE4
T8	Y	N	N	Y	SPIS, SPMS, SPAS	include [MLA $\wedge$ (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)]	SH1 and AH1
TE3	Y	N	Y	N	SPIS, SPMS, SPAS	ton always false omit [MSA ^ (LPAS) ^ (ACDS)]	SH1 and AH1
TE4	Y	N	N	N	SPIS, SPMS, SPAS	omit [MSA ^ (LPAS) ^ (ACDS)]	SH1 and AH1
TE5	Y	Y	Y	Y	none	ton always false include [MSA ^ (LPAS) ^ (ACDS)]	SH1 and L1-L4 or LE1-LE4
TE6	Y	Y	N	Y	none	include [MSA ^ (LPAS) ^ (ACDS)]	SH1 and L1-L4 or LE1-LE4
TE7	Y	N	Y	Y	SPIS, SPMS, SPAS	ton always false 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.

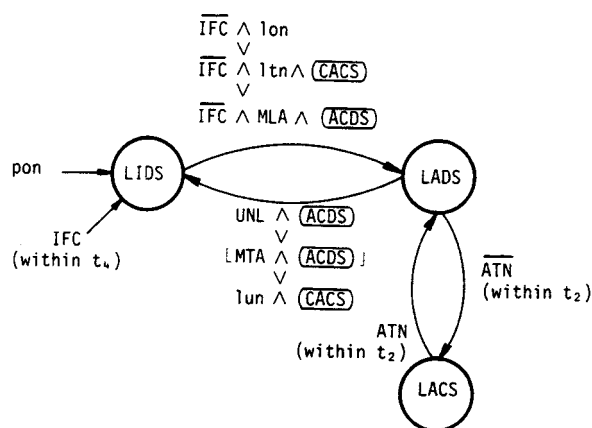


Figure 7—L State Diagram

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	(ACDS) = accept data state (AH function)
IFC = interface clear	(CACS) = controller active state (C function)
ATN = attention	
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