INTERNATIONAL STANDARD

# IEC 60488-1

First edition
2004-07

# IEEE 488.1™

## Higher performance protocol for the standard digital interface for programmable instrumentation –

## Part 1: General



Reference number
IEC 60488-1(E):2004
IEEE Std. 488.1(E):2003

## Publication numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series. For example, IEC 34-1 is now referred to as IEC 60034-1.

## Consolidated editions

The IEC is now publishing consolidated versions of its publications. For example, edition numbers 1.0, 1.1 and 1.2 refer, respectively, to the base publication, the base publication incorporating amendment 1 and the base publication incorporating amendments 1 and 2.

## Further information on IEC publications

The technical content of IEC publications is kept under constant review by the IEC, thus ensuring that the content reflects current technology. Information relating to this publication, including its validity, is available in the IEC Catalogue of publications (see below) in addition to new editions, amendments and corrigenda. Information on the subjects under consideration and work in progress undertaken by the technical committee which has prepared this publication, as well as the list of publications issued, is also available from the following:

- **IEC Web Site (www.iec.ch)**

- **Catalogue of IEC publications**

  The on-line catalogue on the IEC web site (www.iec.ch/searchpub) enables you to search by a variety of criteria including text searches, technical committees and date of publication. On-line information is also available on recently issued publications, withdrawn and replaced publications, as well as corrigenda.

- **IEC Just Published**

  This summary of recently issued publications (www.iec.ch/online_news/ justpub) is also available by email. Please contact the Customer Service Centre (see below) for further information.

- **Customer Service Centre**

  If you have any questions regarding this publication or need further assistance, please contact the Customer Service Centre:

  Email: custserv@iec.ch
  Tel: +41 22 919 02 11
  Fax: +41 22 919 03 00

# INTERNATIONAL STANDARD

# IEC 60488-1

First edition
2004-07

# IEEE 488.1™



## Higher performance protocol for the standard digital interface for programmable instrumentation –

## Part 1: General



International Electrotechnical Commission, 3, rue de Varembé, PO Box 131, CH-1211 Geneva 20, Switzerland
Telephone: +41 22 919 02 11 Telefax: +41 22 919 03 00 E-mail: inmail@iec.ch Web: www.iec.ch

The Institute of Electrical and Electronics Engineers, Inc, 3 Park Avenue, New York, NY 10016-5997, USA
Telephone: +1 732 562 3800 Telefax: +1 732 562 1571 E-mail: stds-info@ieee.org Web: www.standards.ieee.org

Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

# CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

# HIGHER PERFORMANCE PROTOCOLE FOR THE STANDARD DIGITAL INTERFACE FOR PROGRAMMABLE INSTRUMENTATION –

## Part 1: General

## FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
6) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC/IEEE 60488-1 has been processed through subcommittee 65C: Digital communications, of IEC technical committee 65: Industrial-process measurement and control.

This standard cancels and replaces the second edition of IEC 60625-1 (1993).

At times in this standard, specific reference is made to IEEE Std 488.1-1987, which constituted an earlier version of IEEE Std 488.1-2003, the IEEE edition upon which this standard is based. Where specific dated references were made to the 1987 edition, these references have been maintained.

Furthermore, it is to be noted that full compatibility of this standard with IEC/IEEE 60488-2:2004 requires implementation of all revisions indicated previously in the IEEE Introduction. Therefore, readers of this standard are encouraged to read also the companion standard IEEE Std 488.2-1987, which constitutes an earlier edition of IEC 60488-2:2004.

The text of this standard is based on the following documents:

| IEEE Std | FDIS | Report on voting |
|---|---|---|
| 488.1 (2003) | 65C/319A/FDIS | 65C/343/RVD |

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives.

The committee has decided that the contents of this publication will remain unchanged until 2009.

IEC/IEEE 60488 consists of the following publications:

- Higher performance protocol for the standard digital interface for programmable instrumentation – Part 1: General (60488-1).
- Standard digital interface for programmable instrumentation – Part 2: Codes, formats, protocols and common commands (60488-2).

## IEC/IEEE Dual Logo International Standards

This Dual Logo International Standard is the result of an agreement between the IEC and the Institute of Electrical and Electronics Engineers, Inc. (IEEE). The original IEEE Standard was submitted to the IEC for consideration under the agreement, and the resulting IEC/IEEE Dual Logo International Standard has been published in accordance with the ISO/IEC Directives.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEC/IEEE Dual Logo International Standard is wholly voluntary. The IEC and IEEE disclaim liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEC or IEEE Standard document.

The IEC and IEEE do not warrant or represent the accuracy or content of the material contained herein, and expressly disclaim any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEC/IEEE Dual Logo International Standards documents are supplied “AS IS”.

The existence of an IEC/IEEE Dual Logo International 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 IEC/IEEE Dual Logo International Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEC and IEEE are not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Neither the IEC nor IEEE is undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEC/IEEE Dual Logo International Standards or IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

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 concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments for revision of IEC/IEEE Dual Logo International Standards are welcome from any interested party, regardless of membership affiliation with the IEC or IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA and/or General Secretary, IEC, 3, rue de Varembé, PO Box 131, 1211 Geneva 20, Switzerland.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

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

# IEEE Standard for Higher Performance Protocol for the Standard Digital Interface for Programmable Instrumentation

Sponsor

**Technical Committee on Automated Test Systems and Instrumentation (TC-8)**
of the
**IEEE Instrumentation and Measurement Society**

Approved 10 October 2003
**American National Standard Institute**

Approved 12 June 2003
**IEEE-SA Standards Board**



**Abstract:** 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

a) Data exchanged among the interconnected apparatus is digital (as distinct from analog)

b) Number of devices that may be interconnected by one contiguous bus does not exceed 15

c) Total transmission path lengths over the interconnecting cables does not exceed 20 m

d) Data rate among devices does not exceed 8 000 000 B/s

The basic functional specifications of this standard may be used in digital interface applications that 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.

**Keywords:** GPIB, HPIB, HS488, non-interlocked handshake, three-wire handshake

## IEEE Introduction

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, 20 clauses had textual changes with technical implications, although 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."

In 1987, a systematic review was 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. IEEE Std 488.1-1987 represents the culmination of this review cycle. Again, no major technical changes were made, and care was exercised to preserve compatibility with earlier versions of IEEE Std 488.

The IEEE Std 488.1-2003 specification adds new interface functions to allow designers the option of implementing noninterlocked handshake transfers.

In preparing this specification, several stylistic changes to IEEE Std 488.1-1987 were necessary to bring the document up to IEEE standards. The following changes were made:

— Clause 1, which previously consisted of an overview, definitions, and references, was divided into three separate sections. Clause 1 now contains the overview, Clause 2 contains references, and Clause 3 contains definitions.
— The first level of ordered lists was previously numbered (1, 2, 3...). To comply with the IEEE standards, the first level of ordered lists is now lettered (a, b, c...), and the second level of ordered lists is numbered (1, 2, 3...).
— In the earlier version of this specification, tables had no grids. Grids have been added to the tables in this version.

The IEEE 488.1-2003, IEEE Standard for 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 interdevice 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:

— Clause 1 contains the scope, the object, and summary description of the interface.
— Clause 2 contains helpful references.
— Clause 3 contains basic definitions.
— Clause 4 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.
— Clause 5 deals with the electrical realization of the interface in order to transfer messages among a set of interconnected devices.
— Clause 6 deals with the mechanical realization of the interface in order to implement the electrical aspects of the interface system.
— Clause 7 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.

— Clause 8 deals with system considerations that must be recognized by the user of devices designed in accordance with this standard.
— Annexes 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 Clause 1 and Clause 8. 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 Clause 4. 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 it 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.

Device designers must have sufficient awareness of the characteristics of systems, which might include their devices, in order to select correctly among the options provided in this standard. Likewise, system configurators must have sufficient awareness of the options included in each of the devices in their systems 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 1987 version of this standard was based on work initiated by the International Electrotechnical Commission (IEC) within Technical Committee 65, Subcommittee 65C, Working Group 3 (formerly TC66/WG3), and it follows the general concepts of a standard prepared by the IEC. In 1992, the IEEE technical working group enhanced the 1987 standard to improve performance over IEEE Std 488.1-1987.

The “helpful note” on metric threads found in previous editions has been deleted because metric thread use is common IEEE Std 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.



## Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention. A patent holder has filed a statement of assurance that it will grant licenses under these rights without compensation or under reasonable rates and nondiscriminatory, reasonable terms and conditions to all applicants desiring to obtain such licenses. The IEEE makes no representation as to the reasonableness of rates and/or terms and conditions of the license agreements offered by patent holders. Further information may be obtained from the IEEE Standards Department.

### Interpretations and errata

Interpretations and errata associated with this standard may be found at one of the following Internet locations:

— http://standards.ieee.org/reading/ieee/interp/
— http://standards.ieee.org/reading/ieee/updates/errata/

# HIGHER PERFORMANCE PROTOCOL FOR THE STANDARD DIGITAL INTERFACE FOR PROGRAMMABLE INSTRUMENTATION –

## Part 1: General



## 1. Overview

### 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

a) Data exchanged among the interconnected apparatus is digital (as distinct from analog)

b) Number of devices that may be interconnected by one contiguous bus does not exceed 15

c) Total transmission path lengths over the interconnecting cables does not exceed 20 m

d) Data rate among devices does not exceed 8 000 000 B/s

The basic functional specifications of this standard may be used in digital interface applications that 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 that 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 61010-1: 2001, and IEC 60359:2001.[1]

[1]For information on references, see Clause 2.

Throughout this standard, and insofar as further distinction is not necessary, the term “system” denotes the bit-parallel byte-serial interface system that, in general, includes all circuits, cables, connections, message repertoire, and control protocol to effect unambiguous data transfer between devices; and the term “device” or “apparatus” denotes any programmable measurement device or other product connected to the interface system that communicates information via, and conforms to, the interface system definition.

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

a) To define a general-purpose system for use in limited-distance applications
b) To specify the device-independent mechanical, electrical, and functional interface requirements that the apparatus shall meet in order to be interconnected and to communicate unambiguously via the system
c) To specify the terminology and definitions related to the system
d) To enable the interconnection of independently manufactured apparatus into a single functional system
e) To permit devices with a wide range of capability—from the simple to the complex—to be interconnected to the system simultaneously
f) To permit direct communication among the devices without requiring all messages to be routed to a control or intermediate unit
g) To define a system with a minimum of restrictions on the performance characteristics of the devices connected to the system
h) To define a system that permits asynchronous communication over a wide range of data rates
i) To define a system that, of itself, may be relatively low cost and permits the interconnection of low-cost devices
j) To define a system that is easy to use

## 1.3 Interface system overview

### 1.3.1 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:

a) Messages used to manage the interface system itself, hereinafter called interface messages
b) 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.

### 1.3.2 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:

a) A device acting as a listener
b) A device acting as a talker
c) A device acting as a controller

In the context of the interface system described by this standard

a) 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.
b) 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.
c) 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 Clause 4 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 Figure 1.

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

a) 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.
b) 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.
c) The Device Clear and Device Trigger functions provide a device with the ability to be initialized or triggered, respectively, on command from the controller. This may occur simultaneously with other selected or all devices in a system.
d) 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.

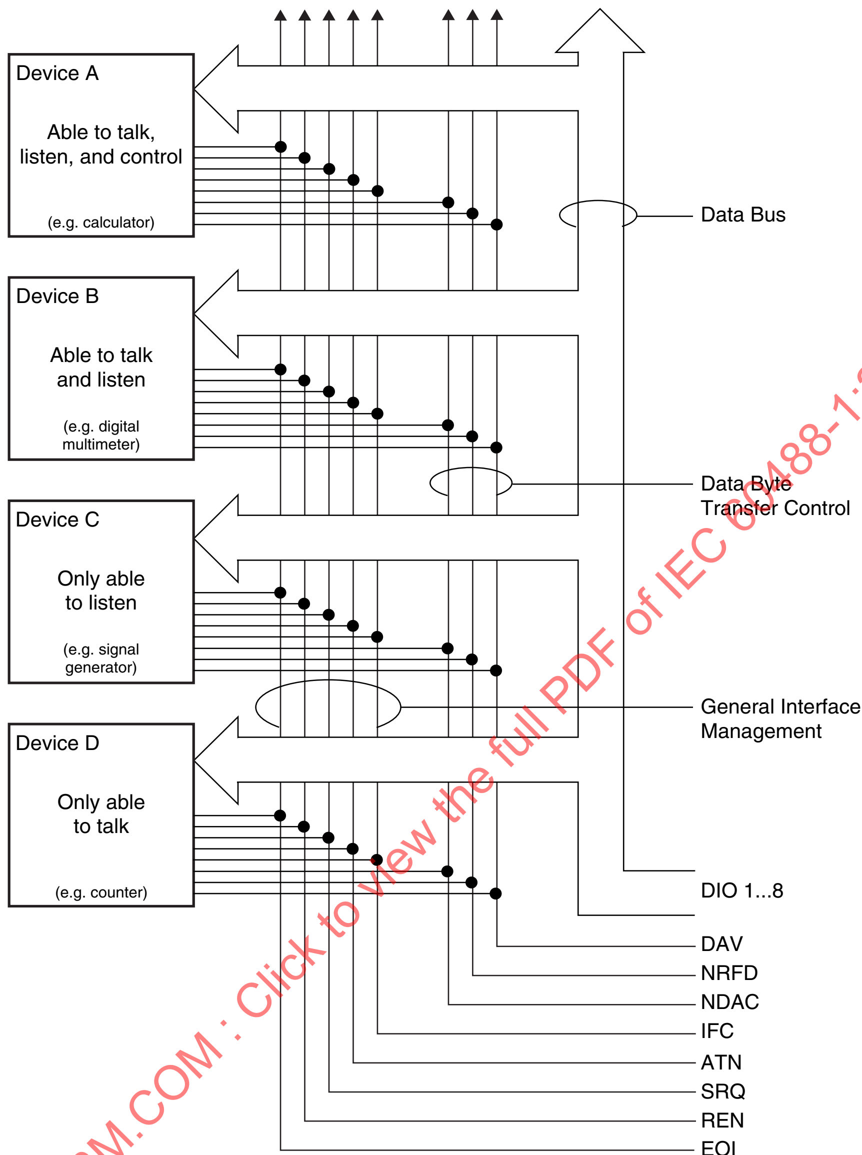



**Figure 1—Interface capabilities and bus structure**



### 1.3.3 Message paths and bus structure

The interface system contains a set of 16 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:

a) Data bus, eight signal lines
b) Data byte transfer control bus, three signal lines
c) General interface management bus, five 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

DIO1 (data input output 1)
.                    .
.                    .
.                    .
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 a 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:

a) Data Valid (DAV) is used to indicate the condition (availability and validity) of information on the DIO signal lines.
b) Not Ready For Data (NRFD) is used to indicate the condition of readiness of device(s) to accept data or (by a source) to indicate to all acceptors that it is capable of supporting noninterlocked handshake cycles.
c) 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 or a noninterlocked 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:

a) 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.
b) 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.
c) 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.
d) 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.
e) 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.

### 1.3.4 Interface system elements

The primary elements of this interface system are as follows:

a) Functional elements
b) Electrical elements
c) Mechanical elements

Each is described in a following clause.

## 2. References

This standard shall be used in conjunction with the following standards. For this standard, all references have been updated to reflect the most recent editions. When these references have been superseded by an approved revision, the revision shall apply. For undated references, the latest edition of the referenced document (including any amendments) applies.

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

IEC 60068-2 (all parts) Environmental Testing – Part 2: Tests.[3]

IEC 61010-1:2001, Safety Requirements for Electrical Equipment for Measurement, Control, and Laboratory Use – Part 1: General requirements.

IEC 60359:2001: Electrical and Electronic Measuring Equipment — Expression of Performance.

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



## 3. Definitions

For the purposes of this recommended practice, the following terms and definitions apply. *The Authoritative Dictionary of IEEE Standards Terms,* Seventh Edition, should be referenced for terms not defined in this clause.

### 3.1 General system terms

**3.1.1 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).

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

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

**3.1.4 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.

**3.1.5 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.)

**3.1.6 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).

**3.1.7 remote control:** A method whereby a device is programmable via its electrical interface connection, enabling it to perform different tasks.

---

[2]ANSI publications can be obtained from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.
[3]IEC publications also are available in the US from the Sales Department. American National Standards Institute, 11 West 42nd Street, 113 Floor, New York, NY 10036, USA.
[4]MIL publications are available from Customer Service, Defense Printing Service, 700 Robbins Ave., Bldg. 4D, Philadelphia, PA 19111-5094.

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

## 3.2 Units connected via the interface system

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

**3.2.2 terminal unit:** A device 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.

## 3.3 Signals and paths

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

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

**3.3.3 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.

**3.3.4 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).

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

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

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

**3.3.8 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 it is hereinafter referred to digital electrical signals only.

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

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

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

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

# 4. Functional specifications

## 4.1 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:

a) Device functions (definition is application dependent)
b) Interface functions (definition is application independent)
c) Message coding logic

All communication to or from interface functions is defined in terms of messages and state linkages (see 4.1.4).

All messages carried on the signal lines are coded according to the coding logic defined in 4.13.

### 4.1.1 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.

### 4.1.2 Interface function concepts

#### 4.1.2.1 Interface functions

An interface function is the system element that 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.

### 4.1.3 Interface function state

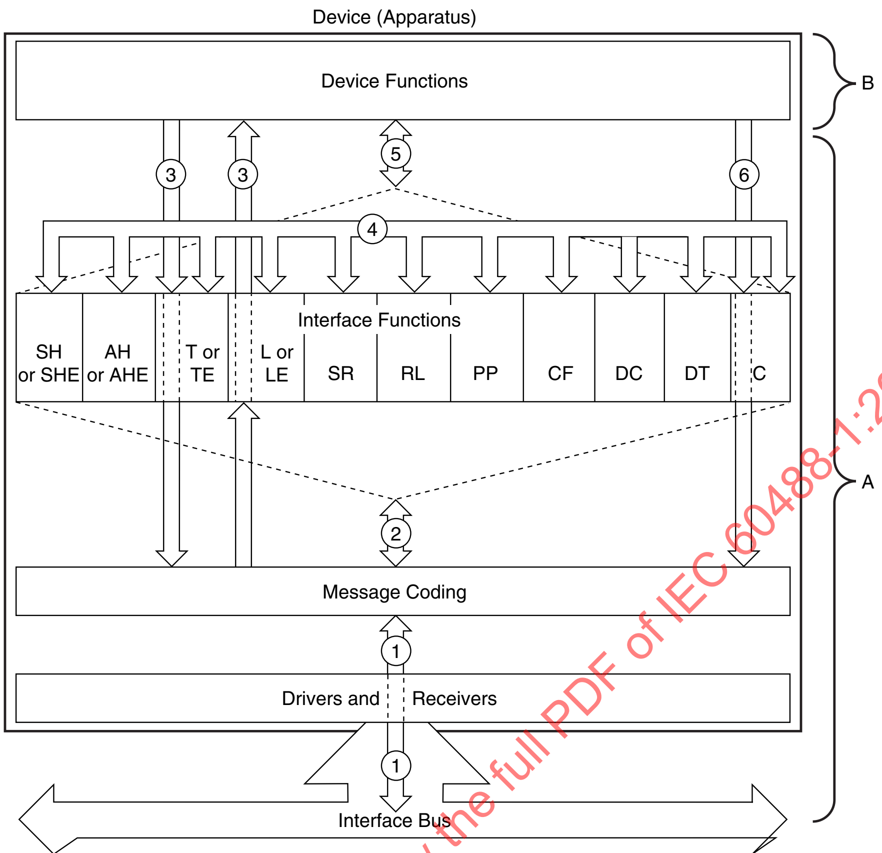
Each of the interface functions is defined in terms of one or more groups of interconnected, mutually exclusive states.

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

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

a) Messages that may or must be sent over the interface while that state is active
b) 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.




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**



#### 4.1.3.1 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.

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.

#### 4.1.3.2 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 elements.

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. Clause 5 and Clause 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.

**Table 1—Interface function repertoire**

| Interface Function | Symbol | Relevant Message Paths |
|---|---|---|
| Source handshake or extended source handshake | SH or SHE | 1, 2, 4, 5 |
| Acceptor handshake or extended acceptor handshake | AH or AHE | 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 |
| Configuration | CF | 1, 2, 4, 5 |



### 4.1.4 Message concepts

#### 4.1.4.1 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 are accomplished through messages sent or received.

#### 4.1.4.2 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 Figure 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.

#### 4.1.4.3 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 Figure 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 Figure 2, message route 3.

#### 4.1.4.4 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 Figure 2, message route 4.

#### 4.1.4.5 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.

#### 4.1.4.6 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:

a) Universal commands (all devices)
b) Addressed commands (all devices addressed to listen)
c) Addresses (all devices)
d) Secondary addresses or commands (all devices enabled by a primary address or command)

For a list of specific multiline commands, see Table 42.

#### 4.1.4.7 Message transfer conventions

##### 4.1.4.7.1 Remote message transfer conventions

a) 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.
b) The interface signal line(s) used to send a message value shall be set to the levels specified by Table 44.
c) As 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 will be passive. Messages shall be transferred in such a way that the active value overrides the passive value in every conflict that arises.
d) A remote message can be transferred in one of four ways:
   1) An active true value being sent is guaranteed to be the value received, and the device need not allow it to be overridden.
   2) A passive true value being sent is not guaranteed to be the value received, and the device must allow it to be overridden.
   3) An active false value being sent is guaranteed to be the value received, and the device need not allow it to be overridden.
   4) A passive false value being sent is not guaranteed to be the value received, and the device must allow it to be overridden.
e) 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.
f) 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 7.4).
g) 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 7.4).
h) Only the multiline message(s) to be sent true will be specified for an interface function state because 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.

##### 4.1.4.7.2 Local message transfer conventions

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

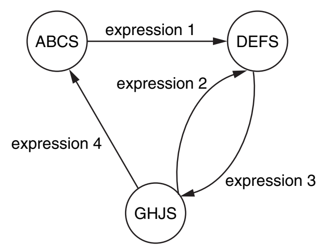
## 4.2 Notation used to specify interface functions

### 4.2.1 State diagram notation

Each state that an interface function can assume is represented graphically as a circle. A four-character upper-case alphanumeric 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 that 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. The representation may be appended by an integer; for example, PPR8.

A linkage from another state diagram is represented by a four-letter, bold, italicized mnemonic, 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 48.

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

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

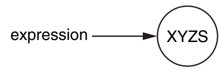
The AND operator takes precedence over the OR operator within an expression unless otherwise specified by parentheses.

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 48.

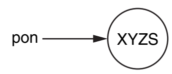
If a portion of an expression is optional in that its true value is not required for the complete expression to be true (at the designer's choice), then it is enclosed within square brackets [. . .].

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 it is assumed to originate in all states (for example, IFC or REN):

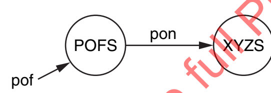



Although power-off (POFS) is a valid state of most interface functions and should normally be shown on all diagrams with a transition leading to the state to be entered at power-on time (pon), a shorthand form is used showing the pon pseudomessage originating a transition to the first state to be entered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

a) Abbreviated notation used on state diagram:




b) Complete representation implied by preceding symbol:




### 4.2.2 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:

a) T indicates active true
b) F indicates active false
c) (T) indicates passive true
d) (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 because 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) that 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.

## 4.3 Source handshake (SH) interface function

### 4.3.1 General description

The SH interface function provides a device with the capability to guarantee the proper transmission of multiline messages. Two alternative versions of the function exist: the SH function and the extended source handshake (SHE) function. The SHE interface function is a superset of the SH interface function. Only one of the two alternative functions shall be implemented in a specific device.

The SH function, or SHE function, controls the initiation of, and termination of, the transmission of a multiline message byte. This function utilizes the data available (DAV), ready for data (RFD), and data accepted (DAC) messages to effect message byte transfers.

Transfers between an SHE function and an extended acceptor handshake (AHE) function may use noninterlocked handshake cycles. Other transfers use interlocked handshake cycles.

NOTE—Both the SH function and the SHE function are described concurrently throughout 4.3 due to the extensive similarity between these two functions.

### 4.3.2 SH function state diagram

The SH function shall be implemented to perform according to the state diagram given in Figure 3 and the state descriptions given throughout 4.3. Table 2 specifies the set of messages and states required to effect transition from one active state to another. Table 4 specifies the messages that shall be sent and the device function interaction required while each state is active.

The SHE function shall be implemented to perform according to the state diagram given in Figure 4 and the state descriptions given throughout 4.3. Table 3 specifies the set of messages and states required to effect transition from one active state to another. Table 5 specifies the messages that shall be sent and the device function interaction required while each state is active.

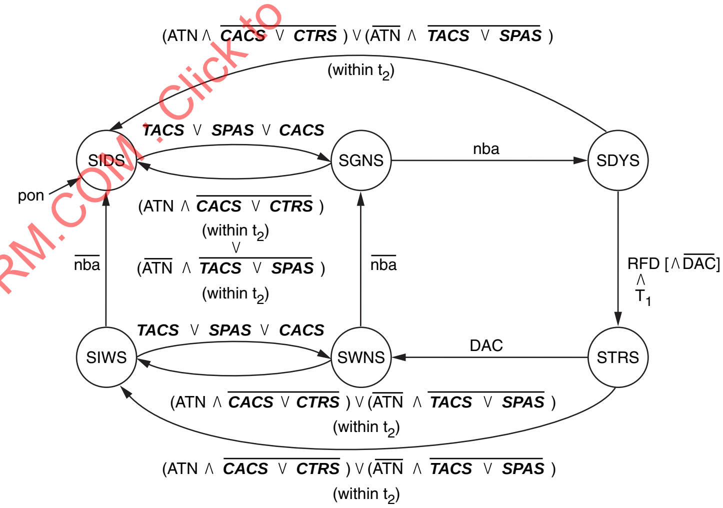



**Figure 3—SH state diagram**

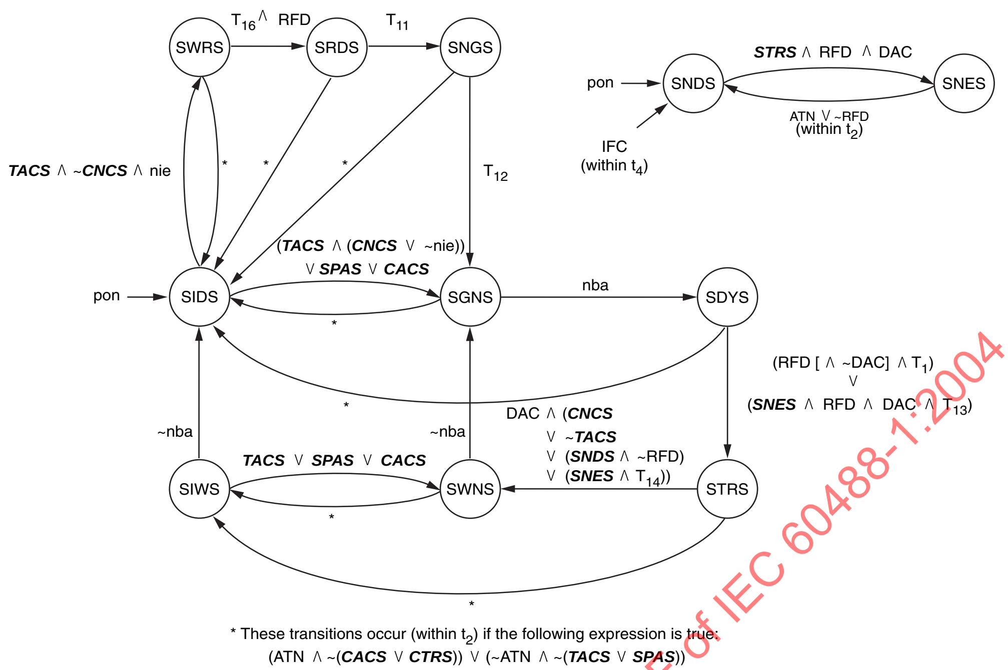



**Figure 4—SHE state diagram**

**Table 2—SH mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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—SHE mnemonics

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| pon | power on | SIDS | source idle state |
| nba | new byte available | SGNS | source generate state |
| nie | noninterlocked enable | SDYS | source delay state |
| | | STRS | source transfer state |
| IFC | interface clear | SWNS | source wait for new cycle state |
| ATN | attention | SIWS | source idle wait state |
| RFD | ready for data | SNDS | source noninterlocked disable state |
| DAC | data accepted | SNES | source noninterlocked enable state |
| | | SWRS | source wait for RFD state |
| | | SRDS | source RFD delay state |
| | | *CNCS* | configure not configured state (CF function) |
| | | *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 4—SH message outputs

| SH State | Remote Message Sent | Device Function (DF) Interaction |
|---|---|---|
| | **DAV** | |
| 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 |

**Table 5—SHE message outputs**

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



### 4.3.3 SH function state descriptions

#### 4.3.3.1 Source idle state (SIDS)

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

In SIDS, the SH function shall send the DAV message passively false. In SIDS, the SHE function shall send the DAV and noninterlocked capable (NIC) messages passively false.

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

a) The talker active state (TACS) is active
b) Or the serial poll active state (SPAS) is active
c) Or the controller active state (CACS) is active

The SHE function shall exit SIDS and enter

a) The SGNS if either
   1) The SPAS is active
   2) Or the CACS is active
   3) Or the TACS is active and either the CNCS is active or the nie message is false
b) The SWRS if the TACS is active and the CNCS is not active and the nie message is true

#### 4.3.3.2 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 SGNS, the SHE function shall send the DAV message false and send the noninterlocked capable (NIC) message passively 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, or the SHE function, shall exit SGNS and enter

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

#### 4.3.3.3 Source delay state (SDYS)

In SDYS, the SH function, or the SHE function, is waiting for a message byte to settle on the interface signal lines after the change during SGNS. In SDYS the SH function, or the SHE function (if it is using interlocked handshaking), is also waiting for all of the acceptor functions to indicate their readiness to accept the message byte.

In SDYS, the SH function shall send the DAV message false. In SDYS, the SHE function shall send the DAV message false and send the NIC messages passively false. In this state, the device shall not change the multiline message being sent.

The SH function shall exit SDYS and enter

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

The SHE function shall exit SDYS and enter

a) The STRS if either
   1) The RFD message is true, only after $T_1$, and if optionally, the DAC message is false
   2) Or the SNES is active and the RFD and DAC messages are true (only after $T_{13}$)
b) The SIDS within $t_2$ if either
   1) The ATN message is true and neither CACS nor CTRS is active
   2) Or the ATN message is false and neither TACS nor SPAS is active

#### 4.3.3.4 Source transfer state (STRS)

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

In STRS, the SH function shall send the DAV message true. In STRS, the SHE function shall send the DAV message true and send the NIC messages passively false. 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

a) The source idle wait state (SIWS) within $t_2$ if either
   1) The ATN message is true and neither CACS nor CTRS is active
   2) Or the ATN message is false and neither TACS nor SPAS is active
b) The source wait for new cycle state (SWNS) if the DAC message is true

The SHE function shall exit STRS and enter

a) The source idle wait state (SIWS) within $t_2$ if either
   1) The ATN message is true and neither CACS nor CTRS is active
   2) Or the ATN message is false and neither TACS nor SPAS is active
b) The SWNS if the DAC message is true and either
   1) The CNCS is active
   2) Or the TACS is not active
   3) Or the SNDS is active and RFD message is false
   4) Or the SNES is active after $T_{14}$

#### 4.3.3.5 Source wait for new cycle state (SWNS)

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

In SWNS, the SH function shall send the DAV message true or false. In SWNS, the SHE function shall send the DAV true or false and the NIC messages passively false.

The SH function, or the SHE function, shall exit SWNS and enter

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

#### 4.3.3.6 Source idle wait state (SIWS)

In SIWS, the SH function, or the SHE 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 the device may continue to prepare for the new (next) message byte generation cycle.

In SIWS, the SH function shall send the DAV message passively false. In SIWS, the SHE function shall send the DAV and NIC messages passively false.

The SH function, or the SHE function, shall exit SIWS and enter

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

#### 4.3.3.7 Source wait for RFD state (SWRS)

In SWRS, the SHE function is waiting for all acceptor functions to indicate their readiness to accept the first DAB since the most recent false transition of ATN.

NOTE—The SHE will enter SWRS to initiate a noninterlocked mode of data transfer. SWRS can only be entered if CNCS is false. CNCS can only be false if the controller explicitly issues a CFGn command. It is a REQUIREMENT that all noninterlocked handshake mode features default (power-on) to disabled until an explicit CFGn command is issued.

In SWRS, the SHE function shall send the DAV false and send the NIC messages passively false.

The SHE function shall exit SWRS and enter

a) The SRDS if the RFD message is true (only after $T_{16}$)
b) Or the SIDS within $t_2$ if either
   1) The ATN message is true and neither CACS nor CTRS is active
   2) Or the ATN message is false and neither TACS nor SPAS is active

#### 4.3.3.8 Source RFD delay state (SRDS)

In SRDS, the SHE is waiting for all acceptors to see the RFD message true before issuing the NIC message. All acceptors must observe the RFD message true before the NIC message is issued to distinguish between the RFD message of a slower acceptor and the source's NIC message.

In SRDS, the SHE function shall send the DAV message false and the NIC message passively false.

The SHE function shall exit SRDS and enter either

a) The SNGS only after $T_{11}$
b) Or the SIDS within $t_2$ if either
   1) The ATN message is true and neither CACS nor CTRS is active
   2) Or the ATN message is false and neither TACS nor SPAS is active

#### 4.3.3.9 Source NIC generate state (SNGS)

In SNGS, the SHE function indicates to all acceptor functions that it is capable of sourcing bytes using non-interlocked handshake cycles.

In SNGS, the SHE function shall send the DAV message false and the NIC message true.

The SHE function shall exit SNGS and enter

a) The SGNS only after $T_{12}$
b) The SIDS within $t_2$ if either
   1) The ATN message is true and neither CACS nor CTRS is active
   2) Or the ATN message is false and neither TACS nor SPAS is active

#### 4.3.3.10 Source noninterlocked disable state (SNDS).

In SNDS, the SHE function is not capable of sourcing multiline message bytes using noninterlocked handshake cycles. The SHE function powers on in SNDS.

The SHE function shall exit SNDS and enter SNES if

a) The STRS is active
b) And the DAC message is true
c) And the RFD message is true
d) And the IFC message is false

#### 4.3.3.11 Source noninterlocked enable state (SNES)

In SNES, the SHE function is capable of sourcing multiline message byte using noninterlocked handshake cycles.

The SHE function shall exit SNES and enter SNDS if either

a) the ATN message is true (within $t_2$)
b) Or the RFD message is false (within $t_2$)
c) Or the IFC message is true (within $t_4$)

### 4.3.4 SH function and SHE function allowable subsets

The only allowable subsets to the SH and SHE interface functions shall be those listed in Tables 6 and 7.

**Table 6—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 |

**Table 7—Allowable subsets to SHE function**

| Identification | Description | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|
| SHE0 | no capability | all | none | none |
| SHE1 | complete capability | none | none | CF1 and T1-T8, TE1-TE8, or C5-C28 |

### 4.3.5 Additional SH and SHE 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, SWRS, SRDS, SNGS, or SGNS. The nba message may become false in any other SH or SHE states.

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

## 4.4 Acceptor handshake (AH) and extended acceptor handshake (AHE) interface functions

### 4.4.1 General description

The AH function provides a device with the capability to guarantee proper reception of remote multiline messages. Two alternative versions of the function exist: the AH function and the extended acceptor handshake (AHE) function. The AHE interface function is a superset of the AH interface function. Only one of the two alternative functions shall be implemented in a specific device.

The AH function may delay either the initiation of, or termination of, a multiline message transfer until prepared to continue with the transfer process. The AH function utilizes the DAV, RFD, and DAC messages to effect each message byte transfer. When a SHE function is transferring data to one or more AHE functions using noninterlocked handshake cycles, an AHE function may either delay the initiation of multiline message bytes or force the SHE to use interlocked handshake cycles.

Transfers between an SHE function and an AHE function may use noninterlocked handshake cycles. Other transfers use interlocked handshake cycles.

NOTE—Both the AH function and the AHE function are described concurrently throughout 4.4 due to the extensive similarity between these two functions.

### 4.4.2 AH function state diagram

The AH interface function shall be implemented so as to perform according to the state diagram given in Figure 5 and the state descriptions given throughout 4.4. Table 8 specifies the set of messages and states required to effect transition from one active state to another. Table 10 specifies the messages that shall be sent and the device function interaction required while each state is active.

The AHE interface function shall be implemented so as to perform according to the state diagram given in Figure 6 and the state descriptions given throughout 4.4. Table 9 specifies the set of messages and states required to effect transition from one active state to another. Table 11 specifies the messages that shall be sent and the device function interaction required while each state is active.

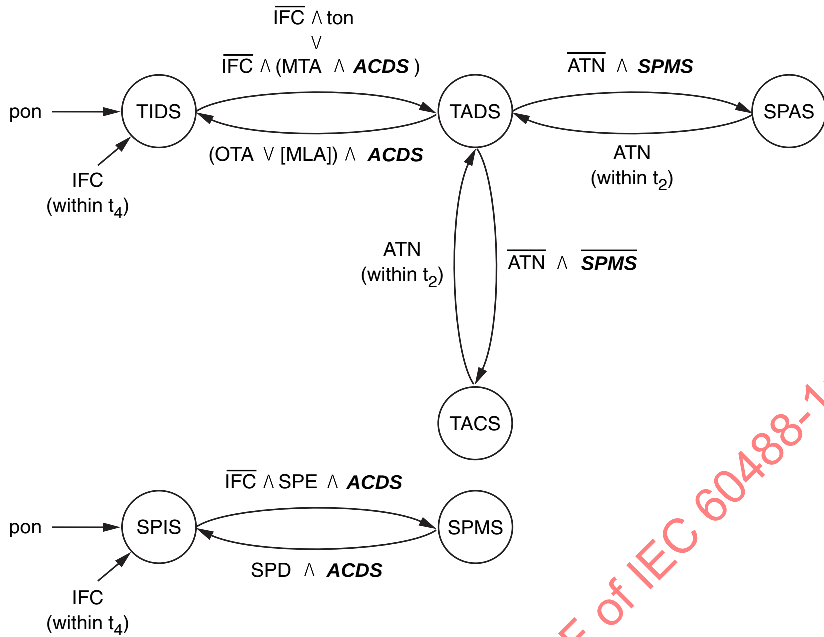



**Figure 5—AH state diagram**

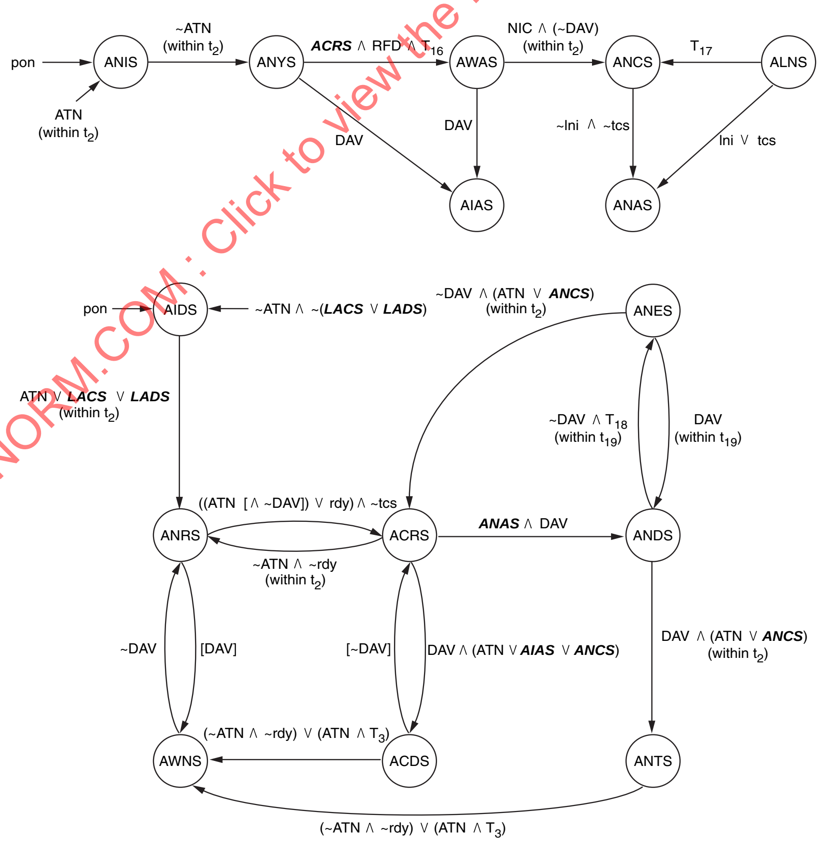



**Figure 6—AH state diagram**

**Table 8—AH mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| pon | power on | AIDS | acceptor idle state |
| rdy | ready for next message | ANRS | acceptor not ready state |
| tcs | take control synchronously[a] | ACRS | accept 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) |

[a]See the first paragraph of 4.12.3.7.

**Table 9—AHE mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| nba | new byte available | AIDS | acceptor idle state |
| pon | power on | ANRS | acceptor not ready state |
| rdy | ready for next message | ACRS | accept ready state |
| tcs | take control synchronously | ACDS | accept data state |
| lni | leave noninterlocked | AWNS | acceptor wait for new cycle state |
| rft | ready for three | ANDS | accept noninterlocked ready state |
| | | ANES | accept noninterlocked not ready state |
| ATN | attention | ANTS | accept noninterlocked terminate state |
| DAV | data valid | ANIS | accept noninterlocked inactive state |
| RFD | Ready For Data | ANYS | accept noninterlocked delay state |
| NIC | noninterlocked Capable | AWAS | accept wait for noninterlocked capable state |
| | | AIAS | accept interlocked always state |
| | | ANCS | accept noninterlocked configured state |
| | | ANAS | accept noninterlocked active state |

**Table 9—AHE mnemonics *(continued)***

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| | | ALNS | accept leave noninterlocked state |
| | | ***LADS*** | listener addressed state (L function) |
| | | ***LACS*** | listener active state (L function) |

**Table 10—AH message outputs**

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



**Table 11—AHE message outputs**

| AHE State | Qualifier | Remote Message Sent | | Device Function (DF) Interaction |
|---|---|---|---|---|
| | | RFD | DAC | |
| AIDS | | (T) | (T) | DF cannot receive multiline or END messages |
| ANRS | | F | F | DF cannot receive multiline or END messages |
| ACRS | | (T) | F | DF cannot receive multiline or END messages |
| AWNS | | F | (T) | DF cannot receive multiline or END messages |
| ACDS | | F | F | DF can receive multiline or END messages if LACS is active |
| ANDS | ANAS $\wedge$ rft | (T) | (T) | DF can receive multiline or END messages (using noninterlocked handshaking) if LACS is active |
| ANDS | (ANAS $\wedge$ rft) | (T) | F | DF can receive multiline or END messages (using noninterlocked handshaking) if LACS is active |

**Table 11—AHE message outputs** ***(continued)***

| AHE State | Qualifier | Remote Message Sent | | Device Function (DF) Interaction |
|---|---|---|---|---|
| | | RFD | DAC | |
| ANES | ANAS ∧ rft | (T) | (T) | DF cannot receive multiline or END messages |
| ANES | (ANAS ∧ rft) | (T) | F | DF cannot receive multiline or END messages |
| ANTS | | F | F | DF can receive multiline or END messages (using noninterlocked handshaking) if LACS is active |

### 4.4.3 AH function state descriptions

#### 4.4.3.1 Acceptor idle state (AIDS)

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

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

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

a) The ATN message is true
b) Or LACS is active
c) Or LADS is active

#### 4.4.3.2 Acceptor not ready state (ANRS)

In ANRS, the AH function, or AHE 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, or AHE function, shall exit ANRS and enter

a) The ACRS if the take control synchronously (tcs) message is false (see the first paragraph of 4.12.3.7) and either
   1) The ATN message is true and the DAV message is false
   2) Or the ready for next message (rdy) message is true

NOTE—Use of the DAV message is optional.

b) The AIDS if the ATN message is false and neither
   1) The LADS is active
   2) Nor LACS is active
c) The AWNS if, optionally, the DAV message is true (note that this transition will never occur under normal interface operation)

#### 4.4.3.3 Acceptor ready state (ACRS)

In ACRS, the AH function, or AHE function, indicates to the interface that it is prepared to receive multiline messages using interlocked handshaking.

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

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

The AHE function shall exit ACRS and enter

a) The accept data state (ACDS) if the DAV message is true and either
   1) The ATN message is true
   2) Or the AIAS is active
   3) Or the ANCS is active
b) The ANDS if the DAV message is true and the ANAS is active
c) The AIDS if the ATN message is false and neither
   1) The LADS is active
   2) Nor LACS is active
d) The ANRS within $t_2$ if both the ATN and the rdy messages are false

#### 4.4.3.4 Accept data state (ACDS)

In ACDS, the AH function, or AHE function, indicates to the SH function that it shall maintain a valid message byte. In this state, the 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, or AHE function, shall exit the ACDS and messages enter

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

#### 4.4.3.5 Acceptor wait for new cycle state (AWNS)

In AWNS, the AH function, or AHE 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

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

#### 4.4.3.6 Accept noninterlocked ready state (ANDS)

In ANDS, the AHE function has accepted a data byte using noninterlocked handshaking. The AHE function shall accept the data byte upon entry to ANDS.

In ANDS, the RFD messages shall be sent passively true. The DAC message shall be sent passively true if the ANAS is active and the ready for three (rft) local message is true. The DAC message shall be sent false if the ANAS is inactive or the rft local message is false.

The ready for three bytes (rft) local message indicates the buffer the device uses to accept incoming multi-line message bytes has room for at least three more bytes and must become false before entry into ANDS that accepts the byte, making the device two from full. The rft local message can be used to stop the transfer for other reasons, which may be asynchronous to the transfer; however, in this case, more than three bytes may be received before the transfer is stopped.

The AHE function shall exit ANDS and enter

a) The AIDS if the ATN message is false and neither
   1) The LADS is active
   2) Nor LACS is active
b) The ANES, after $T_{18}$ but within $t_{19}$, if the DAV message is false
c) The ANTS within $t_2$ if the DAV message is true and either the ATN message is true or the ANCS is active

#### 4.4.3.7 Accept noninterlocked not ready state (ANES)

In ANES, the AHE function is prepared to receive multiline messages using noninterlocked handshaking.

In ANES, the RFD messages shall be sent passively true. The DAC message shall be sent passively true if the ANAS is active and the ready for three (rft) local message is true. The DAC message shall be sent false if the ANAS is inactive or if the rft local message is false.

The AHE function shall exit ANES and enter

a) The AIDS if the ATN message is false and neither
   1) The LADS is active
   2) Nor LACS is active
b) The ANDS within $t_{19}$ if the DAV message is true
c) The ACRS within $t_2$ if the DAV message is false and either the ATN message is true or the ANCS is active

#### 4.4.3.8 Accept noninterlocked terminate state (ANTS)

In ANTS, the AHE function indicates that it is resuming interlocked handshaking. In ANTS, the RFD and DAC messages shall be sent false.

The AHE function shall exit ANTS and enter

a) The AIDS if the ATN message is false and neither
   1) The LADS is active
   2) Nor LACS is active
b) The AWNS if either
   1) The ATN message is true and a period of $T_3$ has elapsed
   2) Or the ATN and rdy messages are both false

#### 4.4.3.9 Accept noninterlocked inactive state (ANIS)

In ANIS, the AHE function is not capable of using noninterlocked handshaking. The AHE function powers on in ANIS.

The AHE function shall exit ANIS and enter ANYS (within $t_2$) if the ATN message is false.

#### 4.4.3.10 Accept noninterlocked delay state (ANYS)

In ANYS, the AHE function is waiting for all acceptors to enter either ACRS or AIDS after a true to false transition of ATN. All acceptors must have ACRS or AIDS active before the sourcing device may send the NIC message true.

The AHE function shall exit ANYS and enter either

a) The ANIS, within $t_2$, if the ATN message is true
b) Or the AWAS, after $T_{16}$, if the ACRS is active and the RFD message is true
c) Or the AIAS if the DAV message is true

#### 4.4.3.11 Accept wait for noninterlocked capable state (AWAS)

In AWAS, the AHE function is waiting for either:

a) The sourcing device to send the DAV message true—indicating that the sourcing device will transmit multiline messages using interlocked handshake cycles.
b) The sourcing device to send the NIC message true—indicating that the sourcing device is capable of sending multiline messages using noninterlocked handshake cycles.

The AHE function shall exit AWAS and enter either

a) The ANIS, within $t_2$, if the ATN message is true
b) Or the AIAS if the DAV message is true
c) Or the ANCS if the NIC message is true and the DAV message is false

#### 4.4.3.12 Accept interlocked always state (AIAS)

In AIAS, the AHE function has detected that the sourcing device has sent the first data byte without sending the NIC message. In AIAS, the AHE function will accept data bytes using the interlocked handshake.

The AHE function shall exit AIAS and enter the ANIS, within $t_2$, if the ATN message is true.

#### 4.4.3.13 Accept noninterlocked configured state (ANCS)

In ANCS, the AHE function has detected that the sourcing device has sent the NIC message, but the AHE function is not ready to accept data bytes using noninterlocked handshake cycles.

If the leave noninterlocked (lni) local message is true, the device will always perform interlock handshaking on all multiline message bytes. If lni local message is false and the source issues the NIC message, the device will perform noninterlock handshaking on all multiline message bytes.

NOTE—The AHE will enter ANCS to initiate a noninterlocked mode of data transfer. ANCS can only be entered if NIC is true, indicating the SHE is in SNGS. The SHE can only be entered if previously CNCS was false. CNCS can only be false if the controller explicitly issues a CFGn command. It is a REQUIREMENT that all noninterlocked handshake mode features default (power-on) to disabled until an explicit CFGn command is issued.

The AHE function shall exit ANCS and enter either

a) The ANIS, within $t_2$, if the ATN message is true
b) Or the ANAS if the leave noninterlocked (lni) local message is false and the take control synchronously (tcs) local message is false

#### 4.4.3.14 Accept noninterlocked active state (ANAS)

In ANAS, the AHE function is capable of accepting data bytes using noninterlocked handshaking.

The AHE function shall exit ANAS and enter either

a) The ANIS, within $t_2$, if the ATN message is true
b) Or the ALNS if the lni local message is true or the tcs local message is true

#### 4.4.3.15 Accept leave noninterlocked state (ALNS)

In ALNS, the AHE function is preparing to stop accepting data bytes using noninterlocked handshake cycles. By leaving ANAS, the DAC message will be sent false while the ANES or ANDS are active (see 4.4.3.6 and 4.4.3.7). In ALNS, the AHE function is waiting for the SH function to detect the DAC message false and stop sending multiline message bytes.

The AHE function shall exit ALNS and enter either

a) The ANIS, within $t_2$, if the ATN message is true
b) Or the ANCS after $T_{17}$

### 4.4.4 AH function and AHE function allowable subsets

The only allowable subsets to the AH and AHE interface functions shall be those listed in Tables 12 and 13, respectively.

### 4.4.5 Additional AH and AHE 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 7.4.

As 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 that exist in an implementation. One consequence

**Table 12—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 |

**Table 13—Allowable subsets to AHE function**

| Identification | Description | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|
| AHE0 | no capability | all | none | none |
| AHE1 | complete capability | none | none | CF1 |



of this statement is that interface function state transitions that 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 4.1.3.2).

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

The AHE function may resume interlocked handshaking at any point during a transfer. A noninterlock-capable listener may choose not to use noninterlocked handshake cycles for any reason. For example, if a noninterlocked-capable listener is only planning to accept a few bytes, it may accept them all using interlocked handshake cycles.

The AHE function requires a buffer of at least 3 bytes for receiving multiline messages.

If a device makes the tcs message true to cause a transition from the CSBS to the CSHS, the device normally should also make the local lni message true. lni forces the AHE function to return to interlocked handshaking, which is required for taking control synchronously.

## 4.5 Talker (T) interface function (Includes serial poll capabilities)

### 4.5.1 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 needs to be implemented in a specific device.

NOTE—Both the T function and the TE function are described concurrently throughout 4.5 due to the extensive similarity between these two functions.

### 4.5.2 T function state diagrams

The T function shall be implemented so as to perform according to the state diagrams given in Figure 7 and the state descriptions given throughout 4.5. Table 14 specifies the set of messages and states required to effect transition from one active state to another. Table 15 specifies the messages that shall be sent and the device function interaction required while each state is active.

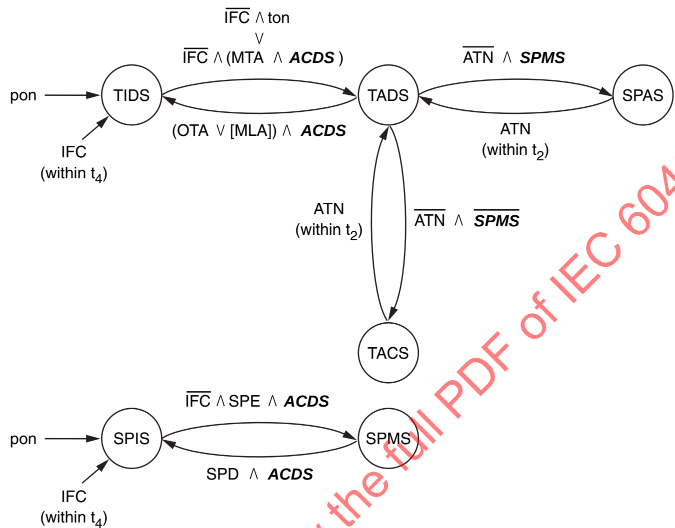



**Figure 7—T state diagram**

The TE function shall be implemented so as to perform according to the state diagrams given in Figure 8 and the state descriptions given throughout 4.5. Table 16 specifies the set of messages and states required to effect transition from one active state to another. Table 15 specifies the messages that shall be sent and the device function interaction required while each state is active.

**Table 14—T mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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 15—T or TE message outputs**

| T State | Qualifier | Remote Messages Sent[a] | | | Device Function (DF) Interaction |
|---|---|---|---|---|---|
| | | **Multiline** | **END** | **RQS[b]** | |
| TIDS | | (NUL) | (F) | (F) | DF not allowed to send messages |
| TADS | | (NUL) | (F) | (F) | DF not allowed to send messages |
| TACS | | DAB[c] or EOS[c] | T or F[c] | (F) | DF can send DAB, EOS, or END message (if used) concurrent with DAB[d] |
| SPAS | APRS inactive | STB[c] | F or T | F | DF can send one STB message[d] |
| SPAS | APRS active | STB[c] | F or T | T | DF can send one STB message[d] |

[a]See Table 44, 4.13.
[b]See 4.5.3.4.
[c]Messages enabled by the T function originating within the device functions.
[d]Under SH control.

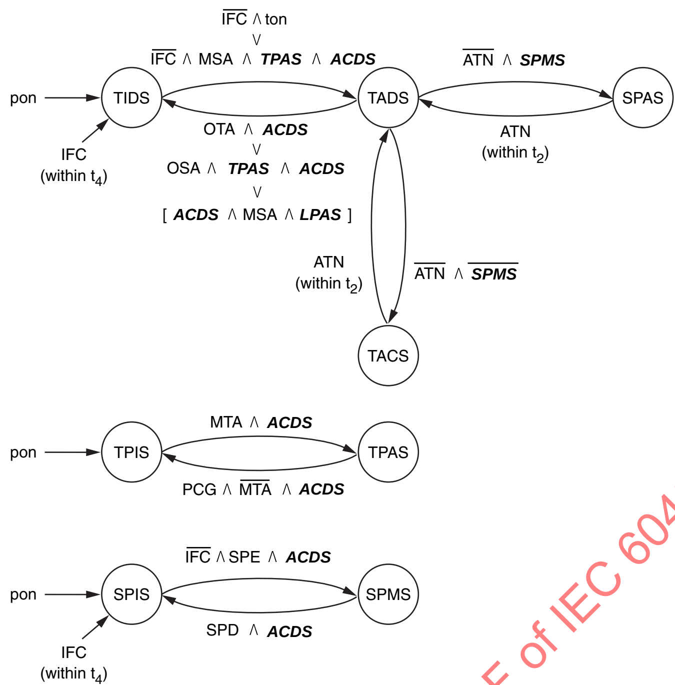



**Figure 8—TE state diagram**

**Table 16—TE mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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 | | |

### 4.5.3 T function state descriptions

#### 4.5.3.1 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

a) The my talk address (MTA) message is true and ACDS is active
b) Or the talk only (ton) message is true (see the last paragraph of 4.5.5)

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

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

#### 4.5.3.2 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

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

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

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

#### 4.5.3.3 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 are, in general, device-dependent and beyond the scope of this standard.

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

a) The TADS within $t_2$ if the ATN message is true
b) The TIDS within $t_4$ if the IFC message is true

#### 4.5.3.4 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 or SHE function to control the transfer of the status byte that contains both the RQS and the STB messages.

Although a controller needs only 1 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

a) The TADS within $t_2$ if the ATN message is true
b) The TIDS within $t_4$ if the IFC message is true

#### 4.5.3.5 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.

#### 4.5.3.6 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

a) The serial poll disable (SPD) message is true and the ACDS is active
b) Or within $t_4$ if the IFC message is true

#### 4.5.3.7 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.

#### 4.5.3.8 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.

### 4.5.4 T function- and TE function-allowable subsets

The only allowable subsets to the T and TE interface functions shall be those listed in Tables 17 and 18.

**Table 17—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 or SHE1 and AH1 or AHE1 |
| T2 | Y | Y | N | N | none | omit [MLA ∧ ***ACDS***] ton always false | SH1 or SHE1 and AH1 or AHE1 |
| T3 | Y | N | Y | N | SPIS, SPMS, SPAS | omit [MLA ∧ ***ACDS***] | SH1 or SHE1 and AH1 or AHE1 |
| T4 | Y | N | N | N | SPIS, SPMS, SPAS | omit [MLA ∧ ***ACDS***] ton always false | SH1 or SHE1 and AH1 or AHE1 |
| T5 | Y | Y | Y | Y | none | include [MLA ∧ ***ACDS***] | SH1 or SHE1 and L1-L4 or LE1-LE4 |

**Table 17—Allowable subsets to T interface function *(continued)***

| Identification | Description: Capabilities | | | | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|---|---|---|
| | Basic Talker | Serial Poll | Talk Only Mode | Unaddress If MLA | | | |
| T6 | Y | Y | N | Y | none | include [MLA ∧ ***ACDS***] ton always false | SH1 or SHE1 and L1-L4 or LE1-LE4 |
| T7 | Y | N | Y | Y | SPIS, SPMS, SPAS | include [MLA ∧ ***ACDS***] | SH1 or SHE1 and L1-L4 or LE1-LE4 |
| T8 | Y | N | N | Y | SPIS, SPMS, SPAS | include [MLA ∧ ***ACDS***] ton always false | SH1 or SHE1 and L1-L4 or LE1-LE4 |

**Table 18—Allowable subsets to TE interface function**

| Identification | Description: Capabilities | | | | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|---|---|---|
| | 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 [MLA ∧ ***LPAS*** ∧ ***ACDS***] | SH1 or SHE1 and AH1 or AHE1 |
| TE2 | Y | Y | N | N | none | omit [MLA ∧ ***LPAS*** ∧ ***ACDS***] ton always false | SH1 or SHE1 and AH1 or AHE1 |
| TE3 | Y | N | Y | N | SPIS, SPMS, SPAS | omit [MLA ∧ ***LPAS*** ∧ ***ACDS***] | SH1 or SHE1 and AH1 or AHE1 |
| TE4 | Y | N | N | N | SPIS, SPMS, SPAS | omit [MLA ∧ ***LPAS*** ∧ ***ACDS***] ton always false | SH1 or SHE1 and AH1 or AHE1 |
| TE5 | Y | Y | Y | Y | none | include [MLA ∧ ***LPAS*** ∧ ***ACDS***] | SH1 or SHE1 and L1-L4 or LE1-LE4 |

**Table 18—Allowable subsets to TE interface function *(continued)***

| Identification | Description: Capabilities: Basic Extended Talker | Serial Poll | Talk Only Mode | Unaddress If MSA ∧ *LPAS* | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|---|---|---|
| TE6 | Y | Y | N | Y | none | include [MLA ∧ ***LPAS*** ∧ ***ACDS***] ton always false | SH1 or SHE1 and L1-L4 or LE1-LE4 |
| TE7 | Y | N | Y | Y | SPIS, SPMS, SPAS | include [MLA ∧ ***LPAS*** ∧ ***ACDS***] | SH1 or SHE1 and L1-L4 or LE1-LE4 |
| TE8 | Y | N | N | Y | SPIS, SPMS, SPAS | include [MLA ∧ ***LPAS*** ∧ ***ACDS***] ton always false | SH1 or SHE1 and L1-L4 or LE1-LE4 |

### 4.5.5 Additional T and TE interface function requirements and guidelines

Each device that includes a T function or TE function shall provide a means by which the talk address (or secondary address) that 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.

## 4.6 Listener (L) interface function

### 4.6.1 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.

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

NOTE—Both the L function and the LE function are described concurrently throughout 4.6 due to the extensive similarity between these two functions.

### 4.6.2 L function state diagram

The L function shall be implemented so as to perform according to the state diagram given in Figure 9 and the state descriptions given throughout 4.6. Table 19 specifies the set of messages and states required to effect transition from one active state to another. Table 20 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 Figure 10 and the state descriptions given throughout 4.6. Table 21 specifies the set of messages and states required to effect transition from one active state to another. Table 20 describes the device function interaction required while each state is active.

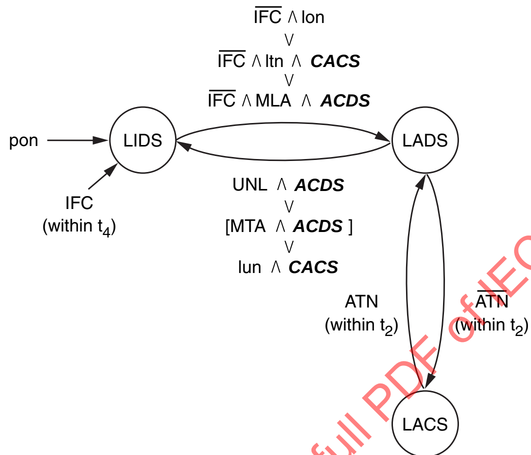


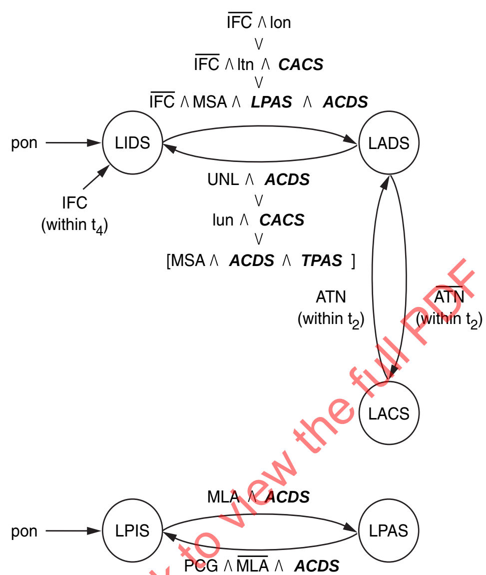

**Figure 9—L state diagram**

**Table 19—L mnemonics**

| **Messages** | | **Interface States** | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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 20—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 10—LE state diagram**



### 4.6.3 L function state descriptions

#### 4.6.3.1 Listener idle state (LIDS)

In 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

a) The my listen address (MLA) message is true and ACDS is active
b) Or the listen only (lon) message is true (see 4.6.5)
c) Or the listen (ltn) message is true and CACS is active

**Table 21—LE mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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 | | |

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

a) The my secondary address (MSA) message is true and the ACDS state is active, and the listener primary addressed state (LPAS) is active
b) Or the lon message is true
c) Or the ltn message is true and CACS is active

### 4.6.3.2 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

a) The listener active state (LACS) within $t_2$ if the ATN message is false
b) The LIDS if either
   1) The unlisten (UNL) message is true and ACDS is active
   2) Or the local unlisten (lun) message is true and CACS is active
   3) Or the MTA message is true and ACDS is active
   4) 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

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

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

#### 4.6.3.3 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 or AHE function is used by the device function(s) to control the message transfer.

NOTE—The coding and format of the data are, 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

a) The LADS within $t_2$ if the ATN message is true
b) The LIDS within $t_4$ if the IFC message is true

#### 4.6.3.4 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.

#### 4.6.3.5 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.

### 4.6.4 L function and LE function allowable subsets

The only allowable subsets to the L and LE interface functions shall be those listed in Tables 22 and 23.

### 4.6.5 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), which it recognizes as MLA (or MSA), can be changed in the field by the user of the device.

**Table 22—Allowable subsets to L interface function**

| Identification | Description: Capabilities: Basic Listener | Listen Only Mode | Unaddress If MTA | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|---|---|
| L0 | N | N | N | all | none | none |
| L1 | Y | Y | N | none | omit [MTA ∧ ***ACDS***] | AH1 or AHE1 |
| L2 | Y | N | N | none | omit [MTA ∧ ***ACDS***]<br>lon always false | AH1 or AHE1 |
| L3 | Y | Y | Y | none | omit [MTA ∧ ***ACDS***] | AH1 or AHE1 and T1-T8 or TE1-TE8 |
| L4 | Y | N | Y | none | omit [MTA ∧ ***ACDS***]<br>lon always false | AH1 or AHE1 and T1-T8 or TE1-TE8 |

**Table 23—Allowable subsets to LE interface function**

| Identification | Description: Capabilities: Basic Extended Listener | Listen Only Mode | Unaddress If MSA ∧ ***TPAS*** | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|---|---|
| LE0 | N | N | N | all | none | none |
| LE1 | Y | Y | N | none | omit [MSA ∧ ***TPAS*** ∧ ***ACDS***] | AH1 or AHE1 |
| LE2 | Y | N | N | none | omit [MSA ∧ ***TPAS*** ∧ ***ACDS***]<br>lon always false | AH1 or AHE1 |
| LE3 | Y | Y | Y | none | omit [MSA ∧ ***TPAS*** ∧ ***ACDS***] | AH1 or AHE1 and T1-T8 or TE1-TE8 |
| LE4 | Y | N | Y | none | omit [MSA ∧ ***TPAS*** ∧ ***ACDS***]<br>lon always false | AH1 or AHE1 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 lon message shall be provided with a local means to generate the listen-only condition. It is intended that the lon message be used in a system with no C interface function capability.

## 4.7 Service request (SR) interface function

### 4.7.1 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 4.12.1).

### 4.7.2 SR interface function state diagrams

The SR interface function shall be implemented so as to perform according to the state diagram given in Figure 11 and the state descriptions given throughout 4.7. 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 shall be sent and the device function interaction required while each state is active.

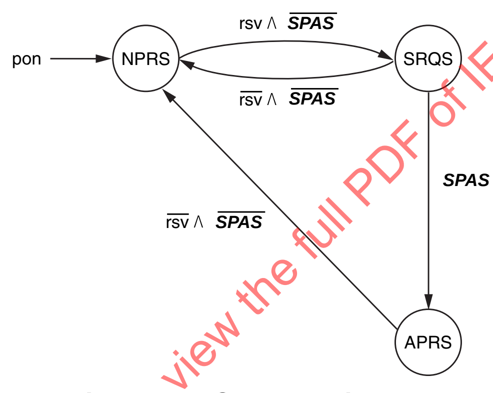



**Figure 11—SR state diagram**



**Table 24—SR mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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) |

### 4.7.3 SR state description

#### 4.7.3.1 Negative poll response state (NPRS)

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

**Table 25—SR message outputs**

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

In NPRS, the SRQ message shall be sent passive false.

NOTE—The RQS message will be sent false when SPAS is active (see 4.5.3.4) 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.

#### 4.7.3.2 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

a) The NPRS if the rsv message is false and SPAS is not active
b) The affirmative poll response state (APRS) if SPAS is active

#### 4.7.3.3 Affirmative poll response state (APRS)

In APRS, the SR function requires service, but it 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 4.5.3.4) 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.

### 4.7.4 SR interface function allowable subsets

The only allowable subsets to the SR interface function shall be those listed in Table 26.

### 4.7.5 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.

**Table 26—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, T6, TE1, TE2, TE5, or TE6 |

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.

Although 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 7.4.2.

## 4.8 Remote local (RL) interface function

### 4.8.1 General description

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

### 4.8.2 RL function state diagram

The RL interface function shall be implemented so as to perform according to the state diagram given in Figure 12 and the state descriptions given throughout 4.8. Table 27 specifies the set of messages and states required to effect transition from one active state to another. Table 28 specifies the device function interaction required while each state is active.

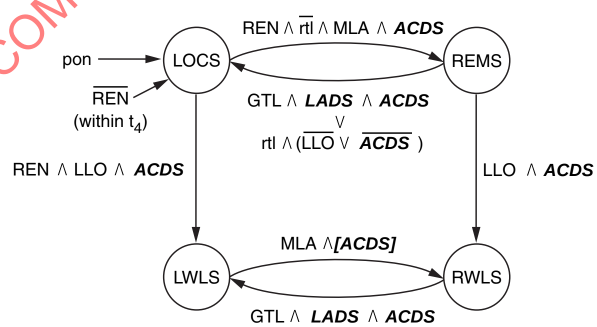



**Figure 12—RL state diagram**

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

**Table 27—RL mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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 function) |
| MLA | my listen address | ***LADS*** | listener addressed state (L function) |

**Table 28—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 |



### 4.8.3 RL state descriptions

#### 4.8.3.1 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

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

#### 4.8.3.2 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

a) The remote with lockout state (RWLS) when MLA is true and ACDS is active
b) The LOCS within $t_4$ if the REN message is false

#### 4.8.3.3 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 that 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

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

#### 4.8.3.4 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 that 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

a) The LOCS within $t_4$ if the REN message is false
b) The LWLS if the GTL message is true and LADS and ACDS are active

### 4.8.4 RL function-allowable subsets

The only allowable subsets to the Remote Local Interface Function shall be those listed in Table 29.



**Table 29—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 lockout | LWLS and RWLS | rtl always false | L1-L4 or LE1-LE4 |

### 4.8.5 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 use device-dependent messages not in conflict with locally available data is independent of the state that 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.

## 4.9 Parallel poll (PP) interface function

### 4.9.1 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 DIO8 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.

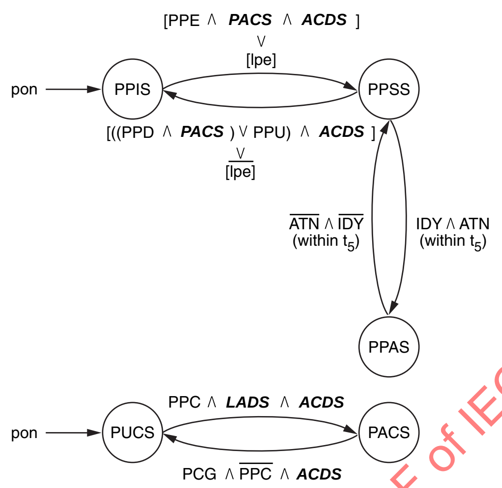
The use of the parallel poll facility within a system requires a commitment of the current interface controller to conduct a parallel poll, as required.

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:

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

### 4.9.2 PP function state diagram

The PP interface function shall be implemented so as to perform according to the state diagram given in Figure 13 and the state descriptions given throughout 4.9. Table 30 specifies the set of messages and states required to effect transition from one active state to another. Table 31 specifies the messages that should be sent and the device function interaction required by the function while each state is active.




Note—See Table 33 for restrictions on use of the optional transitions.

**Figure 13—PP state diagram**



**Table 30—PP mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| pon | power on | PPIS | parallel poll idle state |
| ist | individual status (Table 31) | 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) |
| PPD | parallel poll disable | ***LADS*** | listener addressed state (L) |
| PPC | parallel poll configure | | |
| PCG | primary command group | | |
| PPU | parallel poll unconfigure | | |

**Table 31—PP message outputs**

| PP State | Qualifier | Remote Message Sent | Device Function Interaction |
|---|---|---|---|
| | | PPRn[a, b] | |
| PPIS | | (F) | none |
| PPSS | | (F) | none |
| PPAS | ist = S [b] | T | none |
| PPAS | ist ≠ S [b] | (F) | none |

[a]This column refers only to the specific message assigned by the device.
[b]See 4.9.3.3, second paragraph.

### 4.9.3 PP state descriptions

#### 4.9.3.1 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) if either

a) The parallel poll enable (PPE) message is true and PACS and ACDS are active
b) 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.

#### 4.9.3.2 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

a) The parallel poll active state (PPAS) within $t_5$ if the identify (IDY) and ATN messages are true (a parallel poll is in progress)
b) The PPIS if
   1) The lpe message is false
   2) Or the parallel poll disable (PPD) message is true and PACS and ACDS are active
   3) 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.

#### 4.9.3.3 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 32 lists the PPR message specified by each of the combinations of values of P1 through P3 (see 4.9.5). All other PPR messages should be sent passive false.

**Table 32—PPR message specified by values P1 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).

#### 4.9.3.4 Parallel poll unaddressed to configure state (PUCS)

In PUCS, the PP function shall ignore any PPE or PPD messages that 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) if the PPC message is true and if LADS and ACDS are active.

#### 4.9.3.5 Parallel poll addressed to configure state (PACS)

In PACS, the PP function is able to act on 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.

### 4.9.4 PP interface function-allowable subsets

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

**Table 33—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***]<br>include [PPE ∧ ***PACS*** ∧ ***ACDS***]<br>exclude lpe | L1-L4 or LE1-LE4 |
| PP2 | local configuration | PUCS, PACS | include lpe<br>exclude [((PPD ∧ ***PACS***) ∨ PPU) ∧ ***ACDS***]<br>include [PPE ∧ ***PACS*** ∧ ***ACDS***]<br>local messages shall be substituted for S, P1, P2, P3 | none |

### 4.9.5 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.

## 4.10 Device clear (DC) interface function

### 4.10.1 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.

### 4.10.2 DC function state diagram

The DC interface function shall be implemented so as to perform according to the state diagram given in Figure 14 and the state descriptions given throughout 4.10.


**Figure 14—DC state diagram**

Table 34 specifies the set of messages and states required to effect transition from one active state to another. Table 35 specifies the device function interaction required while each state is active.

**Table 34—DC mnemonics**

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

**Table 35—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 |

### 4.10.3 DC function state descriptions

#### 4.10.3.1 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

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

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

#### 4.10.3.2 Device clear active state (DCAS)

In DCAS, the DC function sends an internal message to the device function(s), which causes 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

a) The DCL message is true
b) Nor the SDC message is true and LADS is active

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

### 4.10.4 DC interface function-allowable subsets

The only allowable subsets to the DC interface function shall be those listed in Table 36.

**Table 36—Allowable subsets to DC interface function**

| Identification | 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 $\wedge$ ***LADS***] | AH1 or AHE1 |



### 4.10.5 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.

## 4.11 Device trigger (DT) interface function

### 4.11.1 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.

### 4.11.2 DT function state diagram

The DT interface function shall be implemented so as to perform according to the state descriptions given in Figure 15 and the state descriptions given throughout 4.11. Table 37 specifies the set of messages and states required to effect transition from one active state to another. Table 38 specifies the device function interaction required while each state is active.

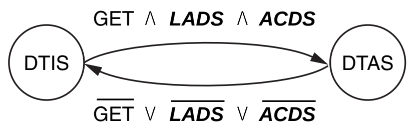



**Figure 15—DT state diagram**

**Table 37—DT mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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 38—DT message outputs**

| DT State | Remote Message Sent | Device Function (DF) Interaction |
|---|---|---|
| DTIS | none | normal device function operation |
| DTAS | none | DF should start performing triggered operation |

### 4.11.3 DT function state descriptions

#### 4.11.3.1 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

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

#### 4.11.3.2 Device trigger active state (DTAS)

In DTAS, the DT function sends an internal message to the device function, which causes 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

a) The GET message is false
b) Or LADS is inactive
c) Or ACDS is inactive

### 4.11.4 DT interface-allowable subsets

The only allowable subsets to the DT interface function shall be those listed in Table 39.

**Table 39—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 |

### 4.11.5 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.

## 4.12 Controller (C) interface function

### 4.12.1 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 that 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 it is the controller-in-charge. This device is called the system controller (of the interface system).

### 4.12.2 C function state diagram

The C interface function shall be implemented so as to perform according to the state diagram given in Figure 16 and the state descriptions given throughout 4.12.1. Table 40 specifies the set of messages and states required to effect transition from one active state to another. Table 41 specifies the messages that shall be sent and the device function interaction required while each state is active.

**Table 40—C mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| 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 |

**Table 40—C mnemonics *(continued)***

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| | | 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 41—C message outputs[a]

| C State | Remote Messages Sent | | | | | Device Function (DF) Interaction |
|---|---|---|---|---|---|---|
| | ATN | IDY | Multiline | IFC | REN | |
| CIDS | (F) | (F) | (NUL) | | | DF shall not send interface messages |
| CADS | (F) | (F) | (NUL) | | | DF shall not send interface messages |
| CACS | T | F | [b] | | | 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 |
| SIIS | | | | (F) | | none |
| SINS | | | | F | | none |
| SIAS | | | | T | | none |
| SRIS | | | | | (F) | none |
| SRNS | | | | | F | none |
| SRAS | | | | | T | none |
| CSNS | none | | | | | no service requests exist |
| CSRS | none | | | | | DF notified of request for service |

[a]Message values sent are shown opposite only those states which affect them. Each major section (denoted by heavier row dividers) of the table corresponds to a group of mutually exclusive states within the controller function.
[b]Any coded interface message listed in Table 44. Although enabled by the C function, these messages originate within the device functions.

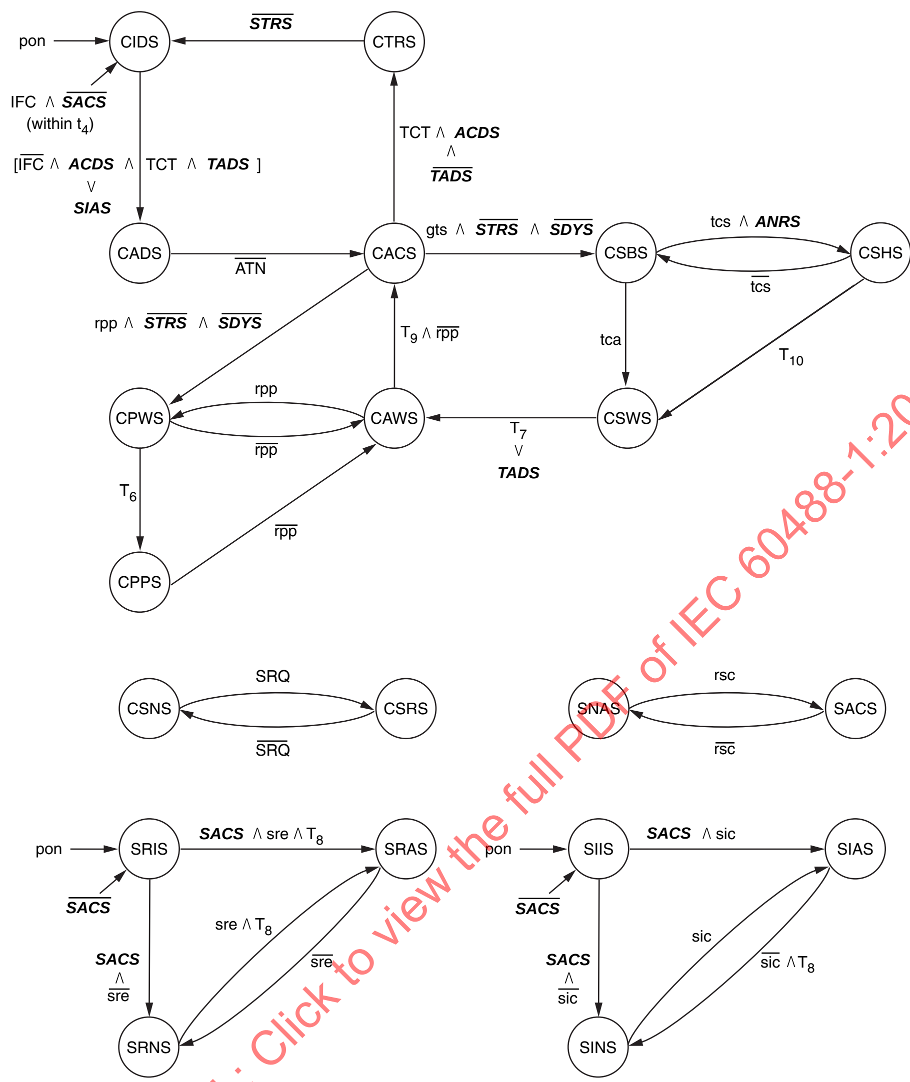



**Figure 16—C state diagram**

## 4.12.3 C state descriptions

### 4.12.3.1 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

a) 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
b) Or the system control interface clear active state (SIAS) is active

NOTE—The expression containing the TCT messages is optional.

#### 4.12.3.2 Controller addressed state (CADS)

In CADS, the C function is in the process of becoming the controller-in-charge of the interface, but it 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

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

#### 4.12.3.3 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 or SHE 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 42 may be sent by the device functions.

**Table 42—Multiline messages**

| Universal Commands (multiline) | Addresses | Addressed Commands | Secondary Commands |
|---|---|---|---|
| LLO | (LAD)[a] | GET | (SAD)[b] |
| DCL | (TAD)[c] | GTL | PPD |
| SPE | UNL | PPC | PPE |
| SPD | | SDC | |
| PPU | | TCT | |

[a]Represents a listen address of a specific device (received as MLA).
[b]Represents a secondary address of a specific device (received as MSA or OSA).
[c]Represents a talk address of a specific device (received as MTA or OTA).

The C function shall exit CACS and enter

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

#### 4.12.3.4 Controller parallel poll wait state (CPWS)

In CPWS, the C function is conducting a parallel 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

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

#### 4.12.3.5 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

a) The CAWS if the rrp message is false
b) The CIDS within $t_4$ if the IFC message is true and SACS is not active

#### 4.12.3.6 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

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

#### 4.12.3.7 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 was entered via the tcs message, the device function(s) shall continue to send it true during this state.

This causes the AH or AHE 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

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

#### 4.12.3.8 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

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

#### 4.12.3.9 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

a) The STRS becomes inactive
b) Or within $t_4$ if the IFC message is true and SACS is not active

#### 4.12.3.10 Controller service requested state (CSRS)

In CSRS, the C function is notifying the device function(s) via a local message that at least 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.

#### 4.12.3.11 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.

#### 4.12.3.12 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.

#### 4.12.3.13 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.

#### 4.12.3.14 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

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

#### 4.12.3.15 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

a) The SIAS if the local sic message is true
b) The SIIS if SACS is not active

#### 4.12.3.16 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

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

#### 4.12.3.17 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

a) The system control remote enable not active state (SRNS) if the send remote enable (sre) message is false and SACS is active
b) 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 $T_8$

#### 4.12.3.18 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

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

#### 4.12.3.19 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

a) The SRNS if the sre message is false
b) The SRIS if SACS is not active

#### 4.12.3.20 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

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

### 4.12.4 C interface function allowable subsets

The only allowable subsets to the C interface function shall be those listed in Table 43.

**Table 43—Allowable subsets to controller interface function**

| Identification[a] | 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*][b] | [*TADS*][c] | *tcs* not always false | C1 | C2 | AH1 or AHE1, L1-L4, or LE1-LE4 | SH1 or SHE1 | 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 | R | — | 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 |

**Table 43—Allowable subsets to controller interface function *(continued)***

| Identification[a] | 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*][b] | [*TADS*][c] | *tcs* not always false | C1 | C2 | AH1 or AHE1, L1-L4, or LE1-LE4 | SH1 or SHE1 | T1-T8, TE1-TE8 |
| 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 | — |
| 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 | — |

[a] 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.
[b] This is part of the CIDS to CADS transitional expression.
[c] 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 shall 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, — = not applicable or not required, Y = yes, N = no.

### 4.12.5 Additional C function requirements and guidelines

WARNING—Use tca with caution.

Restriction on the use of tca: 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 tca 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.

## 4.13 Remote message coding and transfer

### 4.13.1 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 44.

### 4.13.2 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.

### 4.13.3 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 Table 4, Table 10, Table 15, Table 25, Table 31, and Table 41 for times at which messages may be sent.)

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

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

### 4.13.4 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 44.

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 receive 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)



### 4.13.5 Remote message coding table perspective

Table 44 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 footnotes b and k to Table 44 and Annex D.

### 4.13.6 Summary notes and symbols for remote message coding Table 44

Level Assignment:

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

The coding of Table 44 may be translated to equivalent electrical signal levels as specified in 5.2.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

**Table 44—Remote message coding**

| Mnemonic | Message Name | Type | Class | Bus Signal Line(s) and Coding That Asserts the True Value of the Message | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | DIO8 | 7 | 6 | 5 | 4 | 3 | 2 | DIO1 | DAV | NRFD | NDAC | ATN | EOI | SRQ | IFC | REN |
| ACG | addressed command group | M | AC | Y | 0 | 0 | 0 | X | X | X | X | X | X | X | 1 | X | X | X | X |
| ATN | attention | U | UC | X | X | X | X | X | X | X | X | X | X | X | 1 | X | X | X | X |
| DAB | data byte[a b] | M | DD | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | X | X | X | 0 | X | X | X | X |
| DAC | data accepted | U | HS | X | X | X | X | X | X | X | X | X | X | 0 | X | X | X | X | X |
| DAV | data valid | U | HS | X | X | X | X | X | X | X | X | 1 | X | X | X | X | X | X | X |
| DCL | device clear | M | UC | Y | 0 | 0 | 1 | 0 | 1 | 0 | 0 | X | X | X | 1 | X | X | X | X |
| END | end[g] | U | ST | X | X | X | X | X | X | X | X | X | X | X | 0 | 1 | X | X | X |
| EOS | end of string[g c] | M | DD | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | X | X | X | 0 | X | X | X | X |
| GET | group execute trigger | M | AC | Y | 0 | 0 | 0 | 1 | 0 | 0 | 0 | X | X | X | 1 | X | X | X | X |
| GTL | go to local | M | AC | Y | 0 | 0 | 0 | 0 | 0 | 0 | 1 | X | X | X | 1 | X | X | X | X |
| IDY | identify | U | UC | X | X | X | X | X | X | X | X | X | X | X | X | 1 | X | X | X |
| IFC | interface clear | U | UC | X | X | X | X | X | X | X | X | X | X | X | X | X | X | 1 | X |
| LAG | listen address group | M | AD | Y | 0 | 1 | X | X | X | X | X | X | X | X | 1 | X | X | X | X |
| LLO | local lock out | M | UC | Y | 0 | 0 | 1 | 0 | 0 | 0 | 1 | X | X | X | 1 | X | X | X | X |

**Table 44—Remote message coding *(continued)***

| Mnemonic | Message Name | Type | Class | Bus Signal Line(s) and Coding That Asserts the True Value of the Message | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | DIO8 | 7 | 6 | 5 | 4 | 3 | 2 | DIO1 | DAV | NRFD | NDAC | ATN | EOI | SRQ | IFC | REN |
| MLA | my listen address[d] | M | AD | Y | 0 | 1 | L5 | L4 | L3 | L2 | L1 | X | X | X | 1 | X | X | X | X |
| MTA | my talk address[e] | M | AD | Y | 1 | 0 | T5 | T4 | T3 | T2 | T1 | X | X | X | 1 | X | X | X | X |
| MSA | my secondary address[f] | M | SE | Y | 1 | 1 | S5 | S4 | S3 | S2 | S1 | X | X | X | 1 | X | X | X | X |
| NUL | null byte | M | DD | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | X | X | X | X | X | X | X | X |
| OSA | other secondary address | M | SE | (OSA = SCG ∧ $\overline{\text{MSA}}$) | | | | | | | | | | | | | | | |
| OTA | other talk address | M | AD | (OTA = TAG ∧ $\overline{\text{MTA}}$) | | | | | | | | | | | | | | | |
| PCG | primary command group | M | - | (PCG = ACG ∨ UCG ∨ LAG ∨ TAG) | | | | | | | | | | | | | | | |
| PPC | parallel poll configure | M | AC | Y | 0 | 0 | 0 | 0 | 1 | 0 | 1 | X | X | X | 1 | X | X | X | X |
| PPE | parallel poll enable[g] | M | SE | Y | 1 | 1 | 0 | S | P3 | P2 | P1 | X | X | X | 1 | X | X | X | X |
| PPD | parallel poll disable[h] | M | SE | Y | 1 | 1 | 1 | D4 | D3 | D2 | D1 | X | X | X | 1 | X | X | X | X |
| PPR1 | parallel poll response 1[i] | U | ST | X | X | X | X | X | X | X | 1 | X | X | X | 1 | 1 | X | X | X |
| PPR2 | parallel poll response 2[i] | U | ST | X | X | X | X | X | X | 1 | X | X | X | X | 1 | 1 | X | X | X |
| PPR3 | parallel poll response 3[i] | U | ST | X | X | X | X | X | 1 | X | X | X | X | X | 1 | 1 | X | X | X |
| PPR4 | parallel poll response 4[i] | U | ST | X | X | X | X | 1 | X | X | X | X | X | X | 1 | 1 | X | X | X |
| PPR5 | parallel poll response 5[i] | U | ST | X | X | X | 1 | X | X | X | X | X | X | X | 1 | 1 | X | X | X |
| PPR6 | parallel poll response 6[i] | U | ST | X | X | 1 | X | X | X | X | X | X | X | X | 1 | 1 | X | X | X |

**Table 44—Remote message coding** ***(continued)***

| Mnemonic | Message Name | Type | Class | Bus Signal Line(s) and Coding That Asserts the True Value of the Message | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | DIO8 | 7 | 6 | 5 | 4 | 3 | 2 | DIO1 | DAV | NRFD | NDAC | ATN | EOI | SRQ | IFC | REN |
| PPR7 | parallel poll response 7[i] | U | ST | X | 1 | X | X | X | X | X | X | X | X | X | 1 | 1 | X | X | X |
| PPR8 | parallel poll response 8[i] | U | ST | 1 | X | X | X | X | X | X | X | X | X | X | 1 | 1 | X | X | X |
| PPU | parallel poll unconfigure | M | UC | Y | 0 | 0 | 1 | 0 | 1 | 0 | 1 | X | X | X | 1 | X | X | X | X |
| REN | remote enable | U | UC | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | 1 |
| RFD | ready for data | U | HS | X | X | X | X | X | X | X | X | X | 0 | X | X | X | X | X | X |
| RQS | request service[g j] | U | ST | X | 1 | X | X | X | X | X | X | X | X | X | 0 | X | X | X | X |
| SCG | secondary command group | M | SE | Y | 1 | 1 | X | X | X | X | X | X | X | X | 1 | X | X | X | X |
| SDC | selected device clear | M | AC | Y | 0 | 0 | 0 | 0 | 1 | 0 | 0 | X | X | X | 1 | X | X | X | X |
| SPD | serial poll disable | M | UC | Y | 0 | 0 | 1 | 1 | 0 | 0 | 1 | X | X | X | 1 | X | X | X | X |
| SPE | serial poll enable | M | UC | Y | 0 | 0 | 1 | 1 | 0 | 0 | 0 | X | X | X | 1 | X | X | X | X |
| SRQ | service request | U | ST | X | X | X | X | X | X | X | X | X | X | X | X | X | 1 | X | X |
| STB | status byte[g j] | M | ST | S 8 | X | S 6 | S 5 | S 4 | S 3 | S 2 | S 1 | X | X | X | 0 | X | X | X | X |
| TCT | take control | M | AC | Y | 0 | 0 | 0 | 1 | 0 | 0 | 1 | X | X | X | 1 | X | X | X | X |
| TAG | talk address group | M | AD | Y | 1 | 0 | X | X | X | X | X | X | X | X | 1 | X | X | X | X |
| UCG | universal command group | M | UC | Y | 0 | 0 | 1 | X | X | X | X | X | X | X | 1 | X | X | X | X |
| UNL | unlisten | M | AD | Y | 0 | 1 | 1 | 1 | 1 | 1 | 1 | X | X | X | 1 | X | X | X | X |
| UNT | untalk[k] | M | AD | Y | 1 | 0 | 1 | 1 | 1 | 1 | 1 | X | X | X | 1 | X | X | X | X |
| NIC | noninterlocked capable[l] | U | HS | X | X | X | X | X | X | X | X | X | 1 | X | X | X | X | X | X |

**Table 44—Remote message coding *(continued)***

| Mnemonic | Message Name | Type | Class | Bus Signal Line(s) and Coding That Asserts the True Value of the Message | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | DIO8 | 7 | 6 | 5 | 4 | 3 | 2 | DIO1 | DAV | NRFD | NDAC | ATN | EOI | SRQ | IFC | REN |
| CFE | configure enable | M | UC | Y | 0 | 0 | 1 | 1 | 1 | 1 | 1 | X | X | X | 1 | X | X | X | X |
| CFGn | configure n meters[m] | M | SE | Y | 1 | 1 | 0 | N4 | N3 | N2 | N1 | X | X | X | 1 | X | X | X | X |

TABLE NOTES:

[a]D1-D8 specify the device-dependent data bits.
[b]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.
[c]E1-E8 specify the device-dependent code used to indicate the EOS message.
[d]L1-L5 specify the device-dependent bits of the device's listen address.
[e]T1-T5 specify the device-dependent bits of the device's talk address.
[f]S1-S5 specify the device-dependent bits of the device's secondary address.
[g]S specifices 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 |

[h]D1-D4 specify don't-care bits that shall not be decoded by the receiving device. It is recommended that all zeroes be sent.
[i]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.
[j]S1-S6, S8 specify the device-dependent status. (DIO7 is used for the RQS message.)
[k]This code is provided for system use; see 8.3.
[l]The NIC message uses the same Remote Message Coding at the RFD message being sent false. The NIC message is sent by the SHE function, and the RFD message is sent by the AH or AHE function.
[m]N4-N1 specify the particular CFGn message (CFG1, CFG2, . . . CFG15). For example, Y1101001 corresponds to CFG9.



### 4.13.7 ISO code representation: message coding guidelines

Many devices use the ISO-7 bit code (or the equivalent code in ANSI X3.4-1986, 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.

#### 4.13.7.1 Interface messages

The interface system uses message coding as defined in Table 44 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.

#### 4.13.7.2 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.

a) 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.
b) 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, DIO8 for bit 8).

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

### 4.13.8 State transition timing values

The $T_x$ and $t_y$ values listed in Clause 4 throughout the interface function descriptions and state diagrams are defined in 5.8.

## 4.14 Configuration (CF) interface function

### 4.14.1 General description

The Configuration Interface Function provides a device with the capability to record system configuration information sent by the controller.

### 4.14.2 CF function state diagrams

The CF function shall be implemented so as to perform according to the state diagram given in Figure 17 and the state descriptions given throughout 4.14. Table 45 specifies the set of messages and states required to effect transition from one active state to another.


**Figure 17—CF state diagram**



### 4.14.3 CF function state descriptions

NOTE—In the following descriptions, CFGn refers to any one of the following remote messages: CFG1, CFG2, CFG3, CFG4, CFG5, CFG6, CFG7, CFG8, CFG9, CFG10, CFG11, CFG12, CFG13, CFG14, or CFG15. CnS refers to any one of the following states: C01S, C02S, C03S, C04S, C05S, C06S, C07S, C08S, C09S, C10S, C11S, C12S, C13S, C14S, C15S.

**Table 45—CF mnemonics**

| Messages | | Interface States | |
|---|---|---|---|
| **Mnemonic** | **Definition** | **Mnemonic** | **Definition** |
| pon | power on | NCIS | noninterlocked configuration idle state |
| | | NCAS | noninterlocked configuration active state |
| CFE | configure enable | CNCS | configure not configured state |
| PCG | primary command group | C01S | configure active state 1 |
| CFG1 | configure 1 meter | C02S | configure active state 2 |
| CFG2 | configure 2 meters | C03S | configure active state 3 |
| CFG3 | configure 3 meters | C04S | configure active state 4 |
| CFG4 | configure 4 meters | C05S | configure active state 5 |
| CFG5 | configure 5 meters | C06S | configure active state 6 |
| CFG6 | configure 6 meters | C07S | configure active state 7 |
| CFG7 | configure 7 meters | C08S | configure active state 8 |
| CFG8 | configure 8 meters | C09S | configure active state 9 |
| CFG9 | configure 9 meters | C10S | configure active state 10 |
| CFG10 | configure 10 meters | C11S | configure active state 11 |
| CFG11 | configure 11 meters | C12S | configure active state 12 |
| CFG12 | configure 12 meters | C13S | configure active state 13 |
| CFG13 | configure 13 meters | C14S | configure active state 14 |
| CFG14 | configure 14 meters | C15S | configure active state 15 |
| CFG15 | configure 15 meters | ***ACDS*** | accept data state (AH function) |



#### 4.14.3.1 Noninterlocked configuration idle state (NCIS)

In NCIS, the CF function shall ignore any CFGn message received over the interface. The CF function powers on in NCIS.

The NCIS does not provide a remote message-sending capability.

The CF function shall exit NCIS and enter NCAS if the configuration enable (CFE) message is true and the ACDS is active.

#### 4.14.3.2 Noninterlocked configuration active state (NCAS)

In NCAS, the CF function is capable of recording system configuration information sent by the controller. If a CFGn message is received, the appropriate CnS becomes active. For example, if the CFG15 message is received, the C15S becomes active.

The NCAS does not provide a remote message-sending capability.

The CF function shall exit NCAS and enter NCIS if a PCG message is true, the CFE message is false, and the ACDS is active.

#### 4.14.3.3 Configure not configured state (CNCS)

In CNCS, the device is not configured to participate in noninterlocked handshake cycles.

The CNCS does not provide a remote message-sending capability.

The CF function powers on in CNCS.

The CF function shall exit CNCS and enter CnS if the configure n meters (CFGn) message is true, NCAS is active, and ACDS is active. NOTE: The CF function exits CNCS only if the controller sends an explicit CFGn message. It is a REQUIREMENT that all noninterlocked handshake mode features default (power-on) to disabled until an explicit CFGn command is issued.

#### 4.14.3.4 Configure active state 1 (C01S)

In C01S, the controller has communicated to the CF function that the system contains no more than 1 meter of cable.

The C01S does not provide a remote message-sending capability.

The CF function shall exit C01S and enter

a) The CNCS if the ACDS is active and the CFE message is true, or
b) The C02S if
   1) The NCAS is active
   2) And the ACDS is active
   3) And the CFG2 message is true
c) The CnS (when $3 \le n \le 15$) if
   1) The NCAS is active
   2) And the ACDS is active
   3) And the CFGn message is true

#### 4.14.3.5 Configure active state n (CnS), $2 \le n \le 15$

NOTE—The C02S, C03S, C04S, C05S, C06S, C07S, C08S, C09S, C10S, C11S, C12S, C13S, C14S, and C15S states are described concurrently in this section due to extensive similarities among them.

In CnS, the controller has communicated to the CF function that the system contains no more than n meters of cable.

The CnS does not provide a remote message-sending capability.

The CF function shall exit CnS and enter

a) The CNCS if the ACDS is active and the CFE message is true, or
b) The C01S if
   1) The NCAS is active
   2) And the ACDS is active
   3) And the CFG1 message is true
c) The CmS (when $2 \leq m \leq 15$ and $n \neq m$) if
   1) The NCAS is active
   2) And the ACDS is active
   3) And the CFGm message is true

### 4.14.4 CF interface function-allowable subsets

The only allowable subsets to the CF interface function shall be those listed in Table 46.

**Table 46—Allowable subsets to the CF interface function**

| Identification | Description | States Omitted | Other Requirements | Other Function Subsets Required |
|---|---|---|---|---|
| CF0 | no capability | all | none | none |
| CF1 | complete capability | none | none | AHE1, SHE1 |

# 5. Electrical specifications

## 5.1 Application

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

a) Physical distance between devices is short
b) 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—Interface function circuitry connected to the drivers or receivers may be implemented in other device technologies at the designer's choice.

Driver and receiver devices need only be used on those signal lines required for the interface functions implemented (see 5.5.1 for termination requirements).

Either open collector of three-state drivers may be used as determined by data rate considerations of 5.3 and 7.2.

## 5.2 Logical and electrical state relationships

The relationship between the logical states defined in Table 47, Remote Message Coding, and the electrical state levels present on the signal lines is as follows.

**Table 47—Logical and electrical state relationships**

| Coding Logical State | Electrical Signal Levels |
|---|---|
| 0 | corresponds to ≥ + 2.0V, 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.

## 5.3 Driver requirements

Messages may be sent in either an active or passive manner over the interface (see 4.1.4). All passive true message transfer occurs in the high state and shall be carried on a signal line using open collector drivers.

### 5.3.1 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 4.9.3.3).

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.

### 5.3.2 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 5.5)

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

See 5.5.3 for additional requirements that may apply to the driver.

## 5.4 Receiver requirements

### 5.4.1 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 5.5.3 for additional requirements that may apply to the receiver.

### 5.4.2 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: $V_{t+} - V_{t-} \geq +0.4$ V
Low state: Negative going threshold voltage Vt neg ≥ + 0.8 V
High state: Positive going threshold voltage + 2.0 ≥ Vt pos

Devices implementing the SHE or AHE functions are required to use Schmitt-type receiver circuits on all signals.

## 5.5 Composite device load requirements

### 5.5.1 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 5.5.3, and for typical resistive values, see 5.5.5.

### 5.5.2 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.

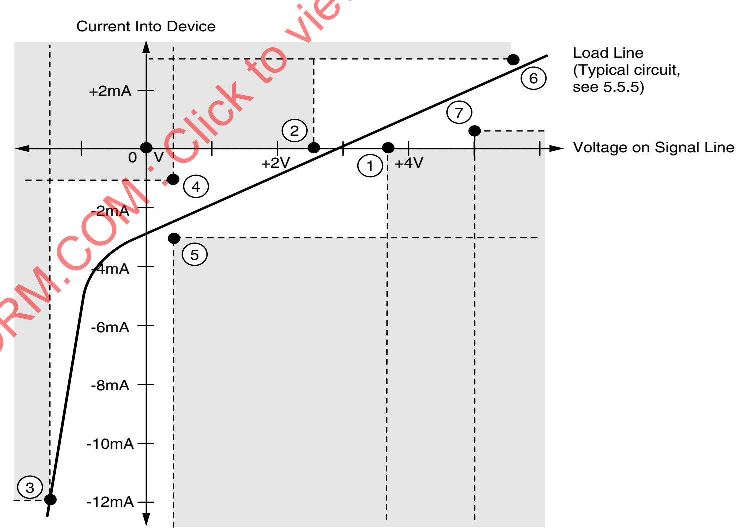
### 5.5.3 DC load requirements

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 and 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 Figure 18.

1) If $I \leq 0$ mA, $V$ shall be $<$ 3.7 V.
2) If $I \geq$ 0 mA, $V$ shall be $>$ 2.5 V.
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 $<$ 2.5 mA.
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 18—DC load boundary specification**

### 5.5.4 Capacity load limit

The internal capacitance load on each signal line shall not exceed 100 pF within each device (see 7.2).

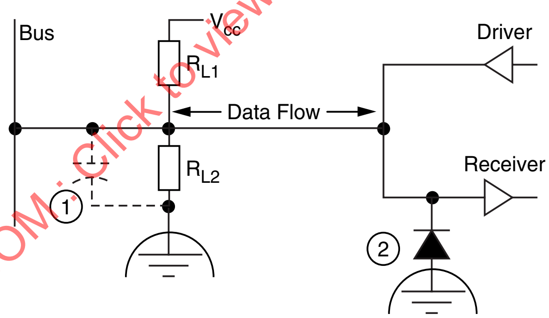
NOTE—The effect of device capacitance on bus operations is most critical at low voltages. As the design of driver and receiver circuits may contribute capacitive loads that vary with voltage, the capacitance should be measured at several voltage levels, all below 2 V, with the device powered on.

### 5.5.5 Typical circuit configuration

Figure 19 shows a typical circuit configuration for signal line input-output circuits for which readily available components exist. This basic circuit is compatible with both TTL microcircuit 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 $V_o$ = +5.25 V Output leakage current (three-state driver) ± 40 µA max at $V_o$ = +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%

Only a single driver and receiver may be connected to each signal line in the typical configuration of Figure 19. Other configurations may exist in which this restriction does not hold, provided the composite device load specifications of 5.5.3 are met.




1 = Internal capacitance (cabling and transceivers) as defined by 5.5.4 and measured according to 5.7.2 for voltage values from 0 to 5 V.
2 = Typically contained within receiver component.

**Figure 19—Typical signal line input-output circuit**

## 5.6 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.

## 5.7 Cable characteristics

### 5.7.1 Conductor requirements

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

a) Each signal line (for example, DIO1, ATN) 0.14 Ω
b) Each individual signal line ground return 0.14 Ω
c) Common logic ground return 0.085 Ω
d) Overall shield 0.0085 Ω

### 5.7.2 Cable construction

The cable should contain at least 24 conductors, of which 16 shall be used for signal lines and the balance shall be 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.

a) 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.
b) The cable shall contain an overall shield carried through the cable assembly and connectors at both ends to be returned to earth ground.
c) 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.
d) Alternatively, any other internal cable construction that yields the same results may be used.

## 5.8 State transition timing values

To ensure maximum possible compatibility among interconnected devices, Table 48 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 48—Time values**

| Time Value Identifier[a] | Function (applies to) | Description | Value |
|---|---|---|---|
| $T_1$ | SH, SHE | setting time for multiline messages | ≥ 2 µs[b] |
| $t_2$ | SH, SHE, AH, AHE, T, L, LE, TE | response to interface messages or state transitions | ≤ 200 ns |
| $T_3$ | AH, AHE | interface message accept time[c] | > 0[d] |
| $t_4$ | 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[e] | ≥ 1.5 µs[f] |
| $T_{10}$ | C | Delay for $\overline{\text{DAV}}$ | ≥ 1.5 µs |
| $T_{11}$ | SHE | settling time for assertion of RFD messages | ≥ 750 ns |
| $T_{12}$ | SHE | hold time for noninterlocked capable multiline messages | ≥ 500 ns |
| $T_{13}$ | SHE | settling time for multiline messages | [g] |
| $T_{14}$ | SHE | hold time for multiline messages | [h] [i] |
| $T_{16}$ | SHE, AHE | response of NRFD to ATN unasserting | ≥ 1 µs |
| $T_{17}$ | AHE | waiting time for the source or extended source function to stop non-interlock handshake cycles | ≥ 750 ns |
| $T_{18}$ | AHE | DAV settling time | [j] |
| $t_{19}$ | AHE | DAV response time | [k] |

[a]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.

[b]If three-state drivers are sued on the DIO, DAV, and EOI lines, T1 may be:

(1) ≥ 1100 ns

(2) Or ≥ 700 ns if it is known that within the controller ATN is driver 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 7.2.3 and warning note.

[c]Time required for interface function to accept, not necessarily respond to, interface messages.

[d]Implementation dependent.

[e]Delay required for EOI, NDAC, and NRFD signal lines to indicate valid stares.

[f]≥ 600 ns for three-state drivers.

[g]The source must guarantee that all acceptors will see the DIO lines stable and valid for at least 10 ns before the assertion edge of DAV. After entering STRS, the SHE function shall not reenter STRS to source another byte for at least 125 ns. See Table 49 for $T_{13}$ times for drivers meeting the requirements of 5.3.2.

[h]The source must guarantee that all acceptors will see the DIO lines stable and valid for at least 10 ns after the assertion edge of DAV. See Table 49 for $T_{14}$ times for drivers meeting the requirements of 5.3.2.

[i]If either the END or EOS message is true, $T_{14}$ shall be ≥ 750 ns. This requirement allows the listener(s) to accept END or an EOS byte using the interlocked handshake.

[j]$T_{18}$ and $t_{19}$ depend on the total length of cable in the system. Refer to Table 50.

[k]See footnote i.

**Table 49—T13 and T14 delay times**

| Total Cable Length (*M*) | T13 | T14 |
|---|---|---|
| $M \le 1$ m | 80 ns | 33 ns |
| 1 m < $M \le 2$ m | 120 ns | 50 ns |
| 2 m < $M \le 3$ m | 151 ns | 69 ns |
| 3 m < $M \le 5$ m | 211 ns | 105 ns |
| 5 m < $M \le 10$ m | 294 ns | 216 ns |
| 10 m < $M \le 15$ m | 344 ns | 336 ns |

**Table 50—T18 and t19 delay times**

| Total Cable Length (*M*) | T18 | t19 |
|---|---|---|
| $M \le 3$ m | 10 ns | 25 ns |
| 3 m < $M \le 7$ m | 25 ns | 40 ns |
| 7 m < $M \le 15$ m | 40 ns | 75 ns |

# 6. Mechanical specifications

## 6.1 Application

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

a) Physical distances between devices are limited
b) Star or linear bus interconnection networks are useful
c) Connector mounting space is limited

## 6.2 Connector type

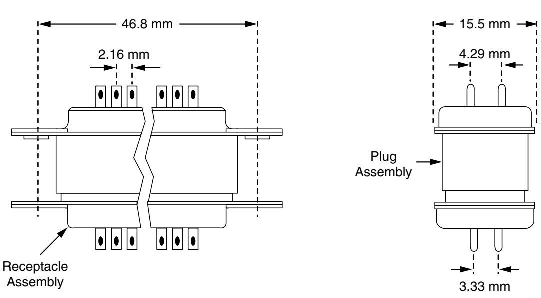
A quality connector of the rack and panel type with proven performance shall be used that has these minimum characteristics.

### 6.2.1 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

### 6.2.2 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 6.4 for additional dimensions):
A 46.8 mm
B 15.5 mm
C 2.16 mm
D 4.29 mm
E 3.33 mm



### 6.2.3 Environmental considerations

Basic environmental performance relative to temperature, humidity, and vibration criteria should be determined in accordance with IEC 60068-2 for climatic category 25/070/21 or MIL STD 202F (1986), where appropriate.

## 6.3 Connector contact assignments

A contact assignment of the cable connector and the device connector shall be as follows in Table 51:

**Table 51—Connector contact assignments**

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

**Table 51—Connector contact assignments** ***(continued)***

| Contact | Signal Line |
|---|---|
| 22 | Gnd. (10) |
| 23 | Gnd. (11) |
| 24 | Gnd. LOGIC |

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



### 6.4 Device connector mounting

Each device shall be provided with a receptacle type connector having the typical dimensions shown in Figure 20. The two rows of 12 contacts each are centered within the trapezoidal shell. The connector mounting shall make provisions to accept the locking screws of the cable assembly.

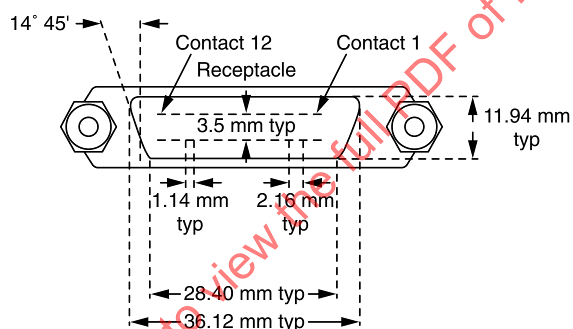



**Figure 20—Receptacle dimensions**

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 Figure 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 Figure 21.

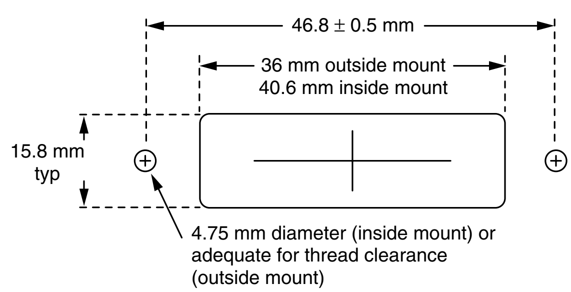
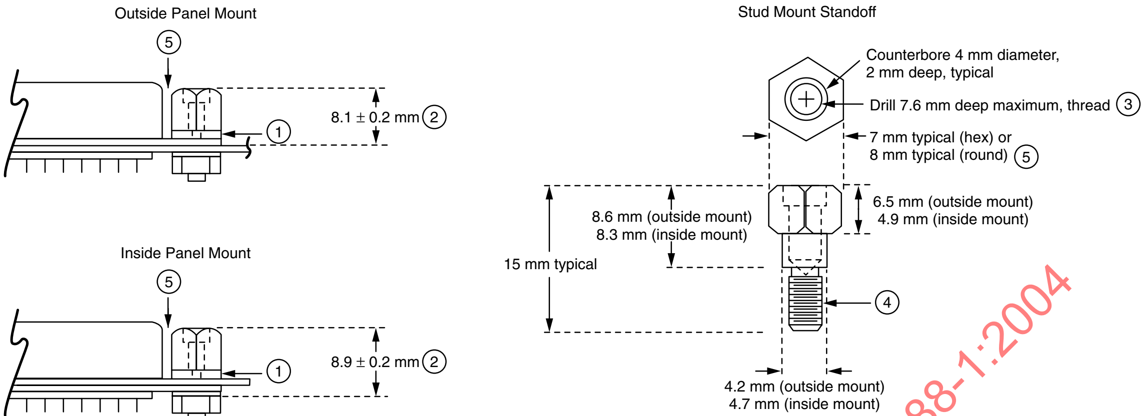



**Figure 21—Connector panel cutout**

The connector shall be attached to the device with one of the stud mount standoffs shown in Figure 22, as determined by the panel mounting method used.



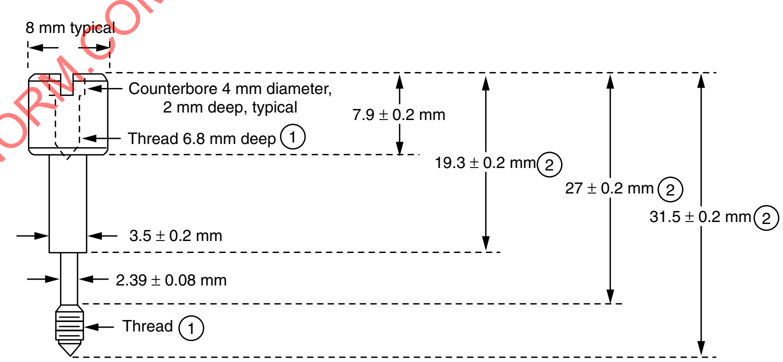

1 Split lock washer, 1.2 mm typical thickness
2 Typical dimension, determined by panel and lock washer thickness, connector housing
3 ISO metric thread M3.5 x 0.6 or equivalent Optimum Metric Fastener System (OMFS) thread 3.5PO.6
4 Thread size designers choice (does not affect compatibility)
5 Care must be taken to allow clearance for mating shell connector

**Figure 22—Mounting dimensions**

## 6.5 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 Figure 23.

Each connector assembly shall be fitted with a pair of captive locking screws. Each lock screw shall conform to the mechanical dimension shown in Figure 23. A retaining ring, or equivalent, shall be used to retain the lock screw as a captive element.




1 ISO metric thread M3.5x0.6 or equivalent O.M.FS thread 3.5PO.6
2 Length is function of housing dimension

**Figure 23—Lock screw**

It is recommended that each pair of connectors, assembled according to the first paragraph of 6.5, be partially enclosed within a suitable housing as shown in Figure 24.

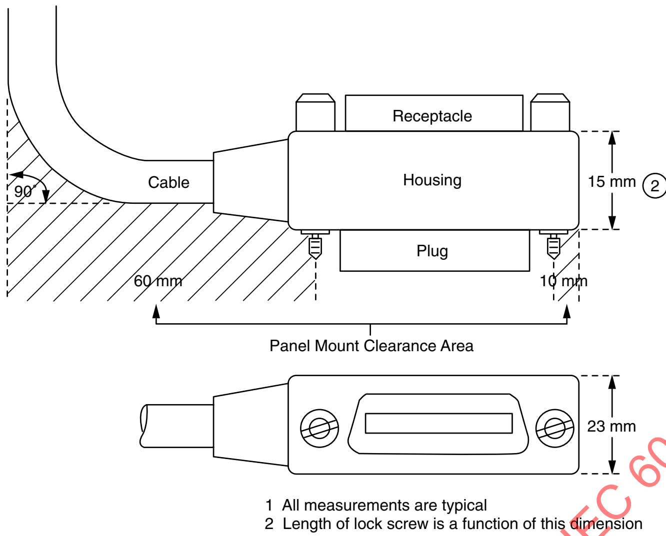



**Figure 24—Cable connector housing**

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

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

# 7. System applications and guidelines for the designer

## 7.1 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 of 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 Clause 4 leads to the following minimum set of signal lines in order to be system compatible:

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

In order to provide system compatibility, a designer shall not introduce new interface functions beyond those designed in Clause 4.

## 7.2 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.

### 7.2.1 Open collector drivers data rates

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.

### 7.2.2 Three-state drivers data rates

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.

### 7.2.3 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:

a) All devices expected to talk at higher rates should use a minimum $T_1$ value of 350 ns.
b) All devices expected to operate at higher rates should use 48 mA three-state drivers.
c) 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.
d) All the devices in the system should be powered on.
e) 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 a) is placed in a system, even if higher speed operation is not intended, there may be data transfer errors if conditions b) through e) are not met for that system.

NOTE—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 maximum).

Actual maximum data rates may be more a function of device-dependent time delays than the cable characteristics or timing values of 5.8.

See 5.5.4 for warning of adverse effects of variable capacitive loading.

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

### 7.2.4 Data rate considerations

If the SHE and AHE functions are used to transfer data with noninterlocked handshake cycles, all guidelines of 7.2.3 shall be followed.

## 7.3 Device capabilities

### 7.3.1 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:

a) SRQ and serial poll
b) Parallel poll
c) NFRD hold

Both the serial poll and parallel poll methods are described in Clause 4.

### 7.3.2 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 or AHE 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 never reach the nonbusy condition, then another listen address (always accessible) should be available to clear the potential hang-up condition.

### 7.3.3 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 that interact directly with the interface functions, as specified throughout Clause 4.

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 Figure 25. This figure is not meant to imply a comprehensive set of switching techniques, switching locations, or switched message contents.




**Figure 25—Remote—local message paths**

## 7.4 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, SHE, AH, AHE, and SR interface functions:

a) 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 or AHE functions).
b) The SRQ message received (by a controller) shall be the logic OR of all SRQ messages sent (by the SR functions).

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

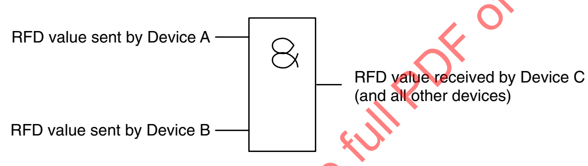
### 7.4.1 RFD and DAC messages

The RFD (or DAC) message sent true (or false) by an AH or AHE function is performed by setting the NRFD (or NDAC) signal line to 0 (high) or driving the NRFD (or NDAC) signal line to 1 (low), respectively.

The RFD (or 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 bistate false.

The logical equivalent of these conditions is illustrated as follows.



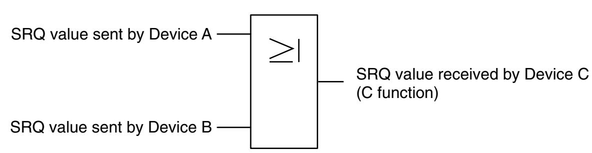

### 7.4.2 SRQ message

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 as follows.




### 7.4.3 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 5.5.5, Figure 19. The driver element shall be a bistate (open collector) driver as represented in Figure 26.

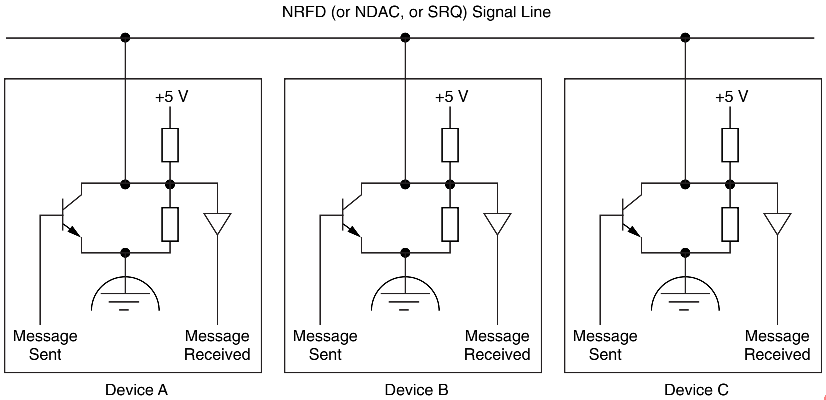



**Figure 26—Bi-state signal line logic (open collector drivers)**



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. The convention is determined by the designer.

Typical signals presented to the NRFD (or SRQ) interface bus signal lines by devices A and B as described in 7.4.1 and 8.4.2 may be represented as shown in Figure 27. 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.

## 7.5 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 and 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.

## 7.6 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 subclauses of Clause 4.

Table 52 represents typical combinations of interface functions and does not imply that these are the only combinations possible or useful.

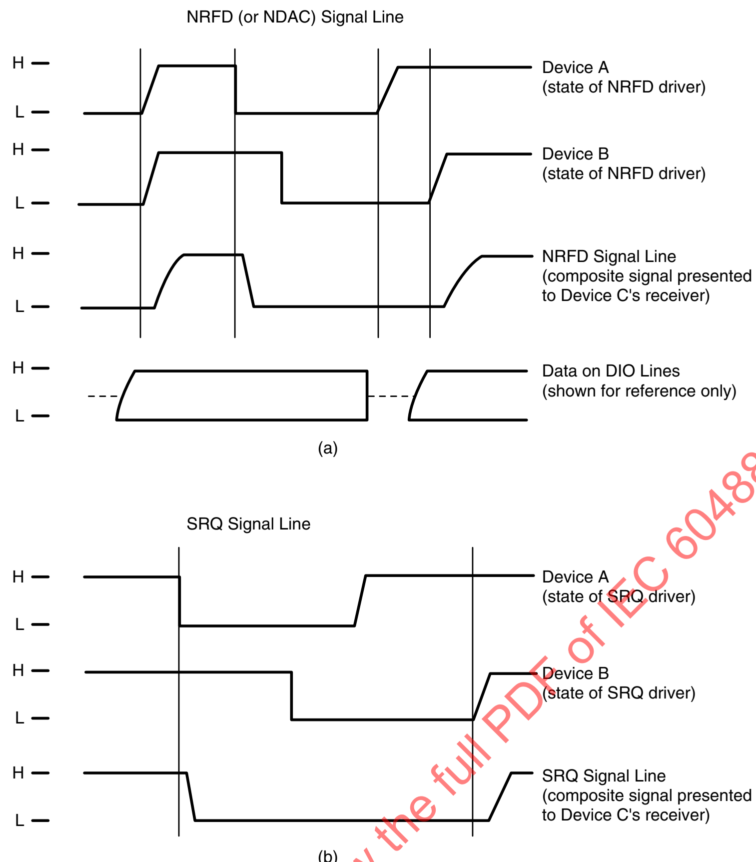



**Figure 27—Signal line logic and timing relationships—(a) NRFD (or NDAC) signal line; (b) SRQ signal line**



## Table 52—Typical combinations of interface functions

| Device | Typical Interface Functions Used |
|---|---|
| Signal generator (only able to listen) | AH or AHE, L, RL, DT |
| Tape Reader (only able to talk) | SH or SHE, AH or AHE, T |
| Digital voltmeter (able to talk and listen) | SH or SHE, AH or AHE, T, L, SR, RL, PP,DC, DT |
| Calculator (able to talk, listen, and control) | SH or SHE, AH or AHE, T, L, C |

### 7.7 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. For example

a) If a listening device implements the DT0 subset, the group execute trigger (GET) remote message should be ignored by the device.
b) 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.
c) 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.

# 8. System requirements and guidelines for the user

## 8.1 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.).

## 8.2 System installation requirements

This includes system configuration restrictions.

### 8.2.1 Maximum number of devices

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

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

### 8.2.3 System controllers

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

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

### 8.2.4 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 of 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 +2.5 V 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.

## 8.3 Address assignment

### 8.3.1 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 an 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.

A TE 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.

### 8.3.2 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 a T function may be assigned a listen address such that L1 through L5 of its MLA code equals T1 through T5 of its MTA code.

### 8.3.3 Secondary addresses

A device that contains a TE function or an LE function may be assigned any value for bits S1 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 an LE function may be assigned a listen address such that L1 through L5 of its MLA code equals T1 through T5 of its MTA code, and both functions may utilize the same secondary address.

## 8.4 Cabling restrictions

### 8.4.1 Maximum cable length

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

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

### 8.4.2 Distribution of maximum cable lengths

The maximum length of cable as defined in 8.4.1 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.

### 8.4.3 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.

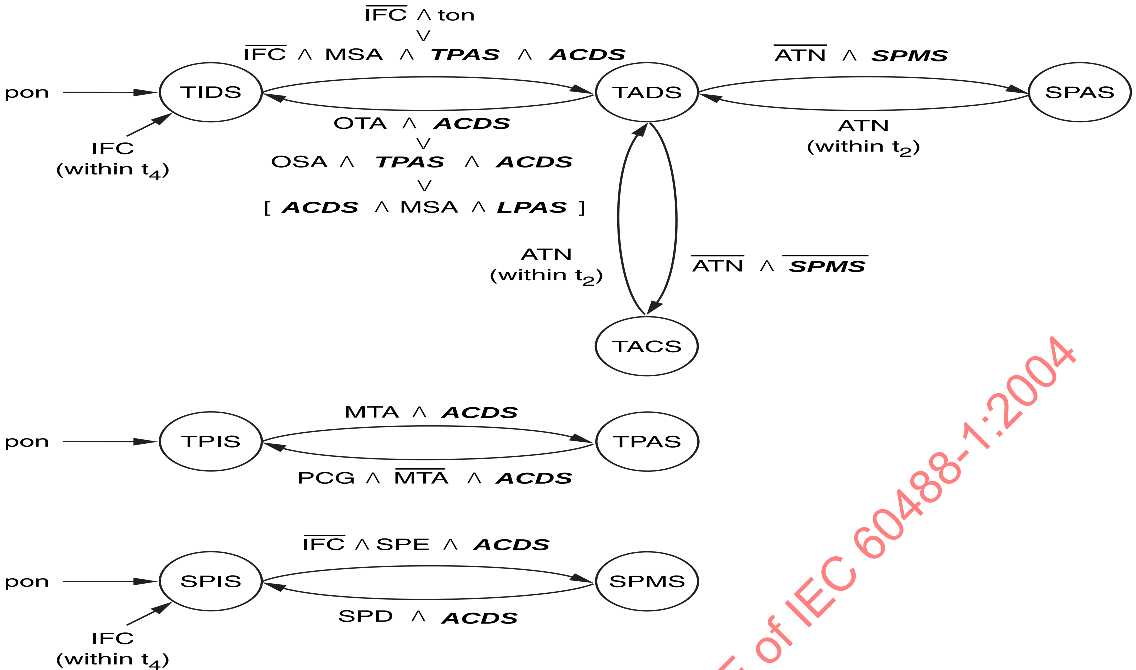
It is recommended that system communications be verified after cable layout changes.

## 8.5 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 leave devices in an acceptable state. Adequate device documentation facilitates this process.

### 8.5.1 Data transfer




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

### 8.5.2 Serial poll (issued by controller usually whenever SRQ = 1 on the interface)

| ATN | | |
|---|---|---|
| 1 | UNL | Prevents other devices from listening to status sent (controller continues to listen without being addressed) |
| 1 | SPE | Puts interface into serial poll mode during which all devices send status instead of data when enabled |
| 1 | $(TAD)_n$ | Enables a specific device to send status. Within this loop, devices should be sequentially enabled |
| 0 | (SBN) or (SBA) | 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 sent SRQ over the interface and will automatically remove it |
| | or | |
| 1 | SPD | Removes the interface from serial poll mode |
| 1 | [UNT] | Disables last talker from sending data if ATN is set 0 |

NOTE—(TAD) represents a talk address of a specific device.
(SBN) represents a status byte sent by a device in which a request for service is not indicated (bit 7=0) (SBN=STB ∧ RQS).
(SBA) represents a status byte sent by a device in which a request for service is indicated (bit 7=1) (SBA=STB ∧ RQS).

### 8.5.3 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.

### 8.5.4 Parallel poll

#### 8.5.4.1 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, and 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—(LAD) represents a listen address of a specific device.

The PPE command can be cleared by a PPD command.

The configuration can be cleared by a PPU command.

#### 8.5.4.2 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 instructing them to use the 0 or 1 value to request service. |

### 8.5.5 Placing devices in forced remote control

| ATN | REN | | |
|---|---|---|---|
| 1 | 1 | LLO | Disables all devices' rtl message |
| 1 | 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—(LAD) represents a listen address of a specific device. (Devices will all revert back to local state as a group at any time a 0 value of REN is placed on the interface.)

Selected local controls may be re-enabled by sending device-dependent remote messages.

### 8.5.6 Sending interface clear

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

# Annex A

(informative)

# Typical instrument system

The typical system shown in Figure 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.

## A.1 Event sequence 1 (device-dependent data returned to processor)

Processor programs instruments and initiates measurements; resulting basic data are returned to processor:

a) Processor initializes the interface system by sending the IFC message true.
b) Processor causes all devices to set their internal conditions to be a predefined state by sending the DCL message true.
c) Processor sends the listen address of the dc power supply followed by program data for that device.
d) Processor sends the unlisten command, then the listen address for the next device, followed by program data for it.

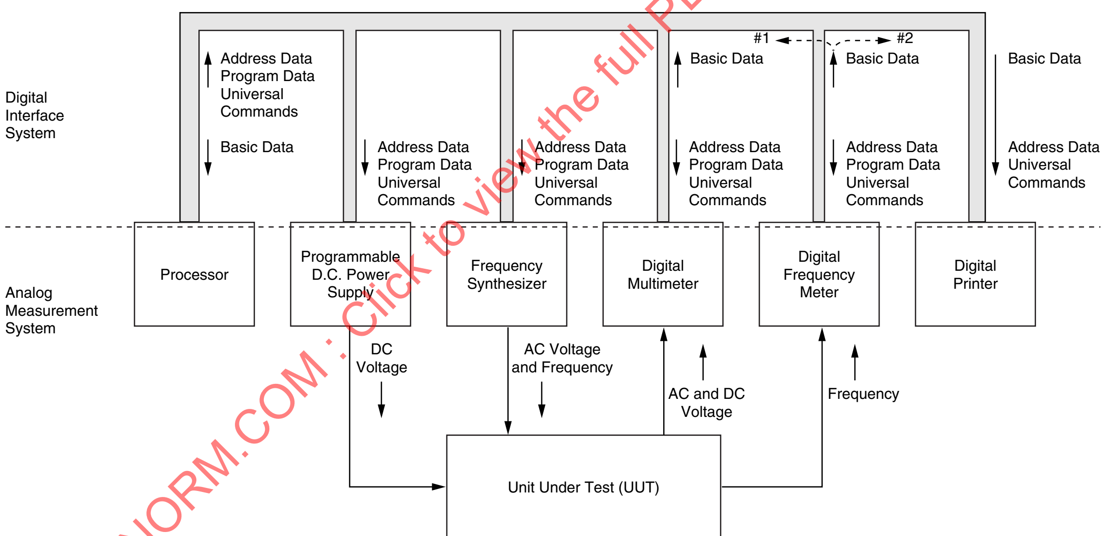



**Figure A.1—Typical system showing capability of interface system to handle variety of instrumentation system needs**

e) Event (d) is repeated until each device of interest for this specific test has been addressed and programmed, and then the unlisten command is sent.
f) Processor sends listen address of selected measurement device (for example, the digital frequency meter), and then that program code required to initiate a measurement.
g) Processor sends unlisten command, addresses itself to listen, and then sends talk address of the measurement device.
h) 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 are returned to another device.

a) Identical to Event Sequence 1.
b) Identical to Event Sequence 1.
c) Identical to Event Sequence 1.
d) Identical to Event Sequence 1.
e) Identical to Event Sequence 1.
f) Identical to Event Sequence 1.
g) Processor sends unlisten command, then the listen address of the digital recorder, followed by the talk address of the measurement device.
h) 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 addressed 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

(informative)

# Handshake process timing sequence

## B.1 General comments

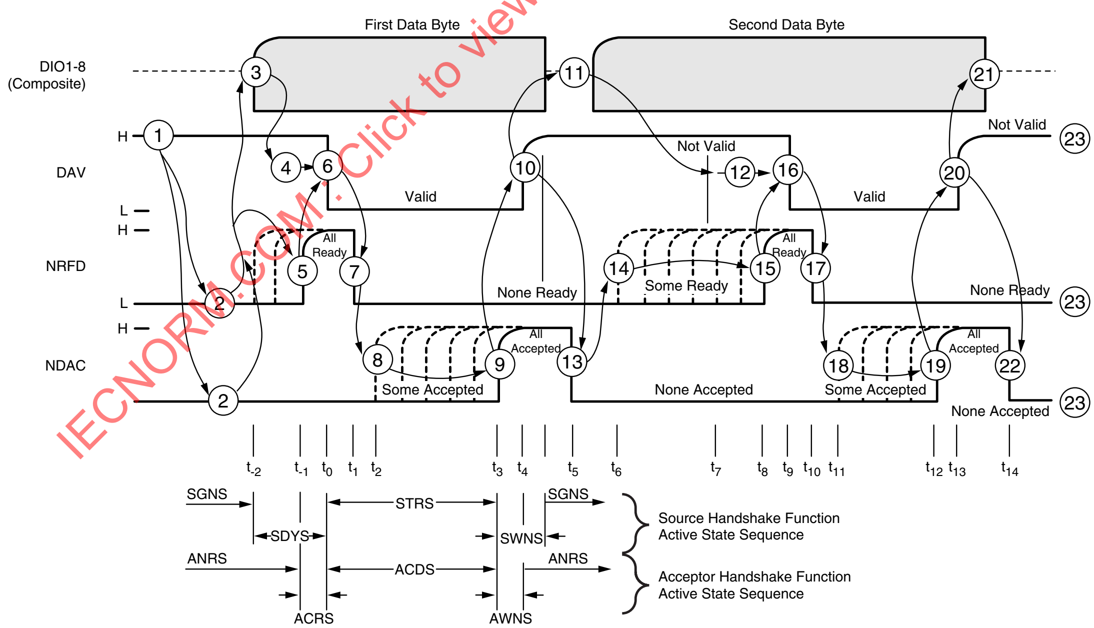
Each data byte transferred by the interface system uses either the interlocked handshake process or the non-interlocked handshake process to exchange data between source and acceptor. Typically, the source is a talker and the acceptor is a listener.

Figure B.1 illustrates the interlocked handshake process by indicating the actual waveforms on the DAV, NRFD, and NDAC signal lines. The NRFD 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.

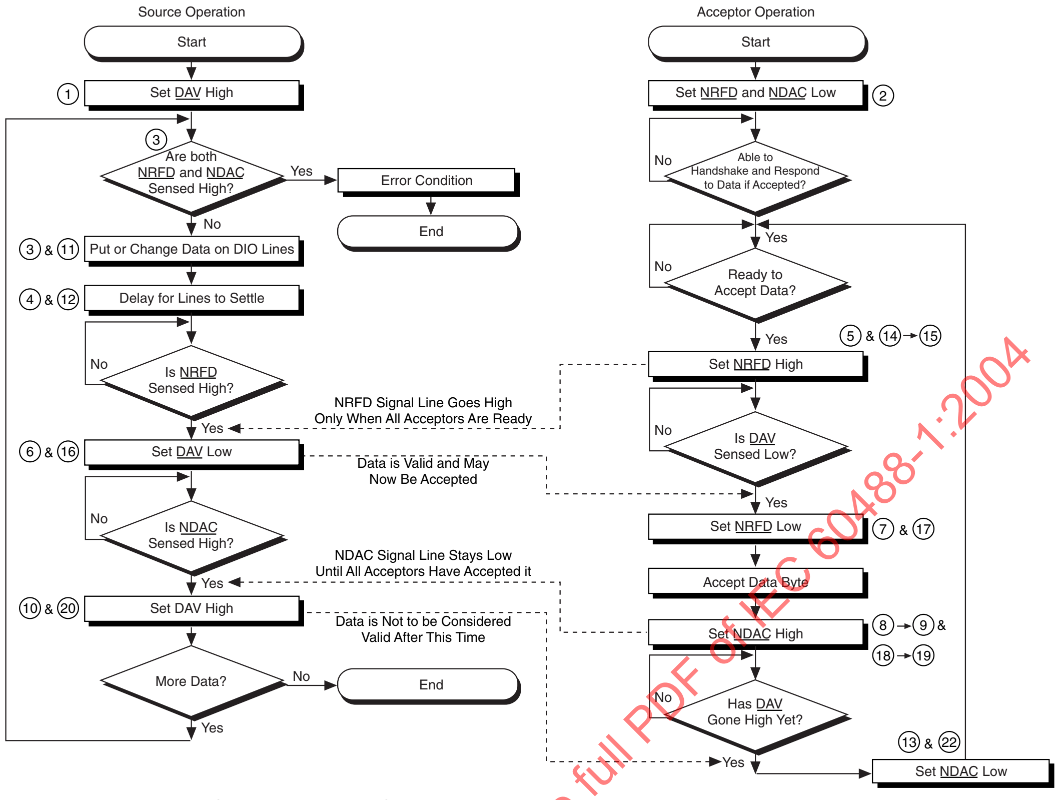
Figure B.3 illustrates the entry into the noninterlocked handshake process by indicating the actual waveforms on the DAV, NRFD, and NDAC signal lines.

Figure B.4 illustrates the transition from a noninterlocked handshake process to an interlocked handshake process due to a holdoff condition.




NOTE—(See Figure B.2 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 interlocked handshake process**




NOTE—(See List of Events) (This flow diagram is not intended to represent the only method of implementing an acceptor handshake. See 4.4.5, paragraph three.)

**Figure B.2—Logical flow of events for source and acceptor when transferring data using interlocked handshake process**

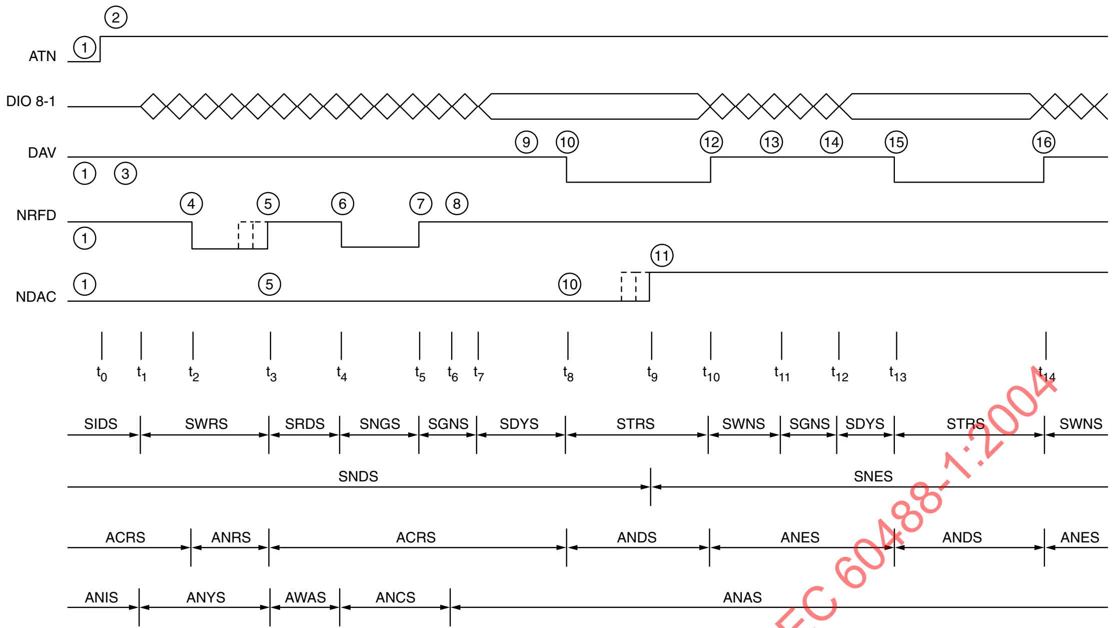



**Figure B.3—Signal line timing sequence for one talker and one or more listeners using noninterlocked handshake cycles**

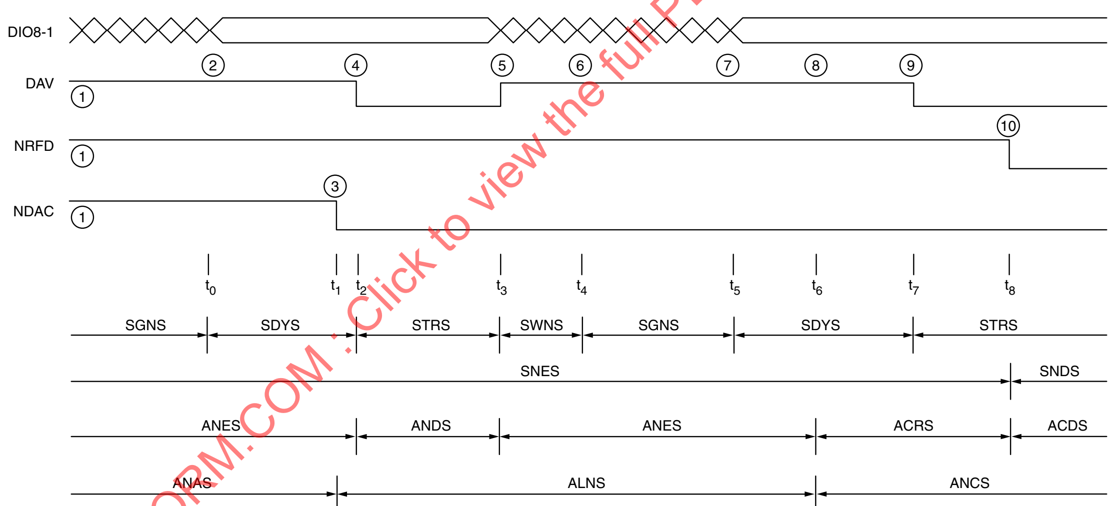



**Figure B.4—Signal line timing sequence for holdoff case**



## B.2 List of events for interlocked handshake process

The numbers in Table B.1 refer to sequential events in Figure B.1 and Figure B.2.

## B.3 List of events for noninterlocked handshake process (Figure B.3)

The noninterlocked Talker transmits bytes to one or more noninterlocked Listeners, as shown in Figure B.3 and described in Table B.2. The ( ) refers to sequential events in Figure B.4.

**Table B.1—Events for interlocked handshake process (Figure B.1 and Figure B.2)**

| Sequential Event | Timing Sequence | Description |
|---|---|---|
| 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), and then sets data byte on DIO lines. |
| 4 | $t_{-2} - t_0$ | 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 go high. |
| 6 | $t_0$ | Source, upon sensing NRFD high, sets DAV low to indicate that data on DIO lines are settled and valid. |
| 7 | $t_1$ | First acceptor sets NRFD low to indicate that it is no longer ready, and 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, and 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. |

### Table B.1—Events for interlocked handshake process (Figure B.1 and Figure B.2)

| Sequential Event | Timing Sequence | Description |
|---|---|---|
| 22 | $t_{14}$ | Acceptors, upon sensing DAV high, set NDAC low in preparation for next cycle. |
| 23 | — | Note that all three interlocked handshake lines are at their initialized states (as in 1 and 2). |

### Table B.1—Events for noninterlocked handshake process

| Sequential Event | Timing Sequence | Description |
|---|---|---|
| 1 | — | Controller in CACS asserting ATN; source in SIDS; acceptor(s) are asserting NDAC in ACRS ready to accept another command byte. |
| 2 | $t_0$ | Controller transitions to CSBS and unasserts ATN. |
| 3 | $t_1$ | Source transitions to SWRS in response to TACS; the acceptor(s), upon sensing ATN false, transition from ANIS to ANYS. |
| 4 | $t_2$ | Acceptor(s) transition to ANRS (if rdy is false) and asserts NDAC and NRFD. |
| 5 | $t_3$ | Acceptor(s) transition to ACRS when rdy becomes true - NRFD unasserts when all acceptors transition to ACRS, upon sensing NRFD false after $T_{16}$, transitions into AWAS; source, upon sensing NRFD false after $T_{16}$, transitions to SRDS. |
| 6 | $t_4$ | Source transitions to SNGS and asserts NIC (NRFD) after $T_{11}$; the acceptor(s) transition to ANCS upon detection of NIC (NRFD). |
| 7 | $t_5$ | Source transitions to SGNS and unasserts NIC (NRFD) after $T_{12}$. |
| 8 | $t_6$ | Acceptor(s) transition to ANAS when lni is false. |
| 9 | $t_7$ | Source transitions to SDYS and sets data byte on DIO lines when nba is true. |
| 10 | $t_8$ | Source, upon sensing NRFD false and NDAC true, sets DAV true to indicate that data on DIO lines is settled and valid; acceptor(s), upon sensing DAV true, transitions to ANDS and unasserts NDAC. |
| 11 | $t_9$ | Source, upon sensing NDAC and NRFD false, enters SNES. |
| 12 | $t_{10}$ | Source, upon sensing NRFD and NDAC false after $T_{14}$, transition to SWNS unasserting DAV. |
| 13 | $t_{11}$ | Source enters SGNS upon detecting nba false. |
| 14 | $t_{12}$ | Source transitions from SGNS to SDYS and sets data byte on DIO lines when nba is true. |
| 15 | $t_{13}$ | Source, upon sensing NRFD and NDAC false after $T_{13}$, sets DAV true to indicate that data on DIO lines is settled and valid. |
| 16 | $t_{14}$ | Source, upon sensing NRFD and NDAC false after $T_{14}$, transition to SWNS unasserting DAV. |