
An interface system for programmable measuring instruments (byte serial, bit parallel)

Part 1. Specification for functional, electrical and mechanical requirements, system applications and requirements for designer and user

[IEC title : An interface system for programmable measuring instruments (byte serial, bit parallel) -

Part 1 : Functional specifications, electrical specifications, mechanical specifications, system applications and requirements for the designer and user]

Un système d'interface pour instruments de mesurage programmables
(bits parallèles, octets série)

Partie 1. Spécifications fonctionnelles, spécifications électriques, spécifications mécaniques, application du système et règles pour constructeur et utilisateur

Ein Schnittstellensystem für programmierbare Meßgeräte (byteseriell, bitparallel)

Teil 1. Festlegungen für funktionelle, elektrische und mechanische Anforderungen, Anwendungendes Systems und Richtlinien für Entwickler und Anwender

National foreword

This Part of this British Standard, published under the direction of the Electronic Equipment Standards Committee, is identical with IEC Publication 625-1 'An interface system for programmable measuring instruments (byte serial, bit parallel)' Part 1 'Functional specifications, electrical specifications, mechanical specifications, system applications and requirements for the designer and user' published in 1979 by the International Electrotechnical Commission (IEC).

The interface system described in this British Standard is, in practice, identical in its functional and electrical specifications with those described in IEEE Standard* 488-1978 'Digital interface for programmable instrumentation'. The mechanical connector system specified in section four of this Part differs from that described in IEEE 488-1978, in which a 24-pin connector is specified. Products complying with the requirements of either the mechanical specification of this Part or of IEEE-488 may be interconnected into the same instrumentation system by the use of a simple mechanical adapter assembly.

Part 2 of this British Standard, which is published simultaneously with this Part, deals with code and format conventions.

Terminology and conventions. The text of the international standard has been approved as suitable for publication as a British Standard without deviation. Some terminology and certain conventions are not identical with those used in British Standards.

References to page numbers in the text refer to IEC page numbers, which are given in brackets at the bottom of each page.

Cross-references

International standard	Corresponding British Standard
IEC 68-1 : 1978	BS 2011 Basic environmental testing procedures Part 1.1 : 1980 General (Identical)
IEC 348 : 1978	BS 4743 : 1979 Specification for safety requirements for electronic measuring apparatus (Identical)

The technical committee has reviewed the provisions of IEC 50 (351), referred to in 2.1.1, and has decided that they are acceptable for use in conjunction with this standard.

IEC 359 : 1971, which is referred to in 1.2.1, is related to BS 4889 : 1973 'Method for specifying the performance of electronic measuring equipment'.

ISO 646-1973, which is referred to in 16.6 and appendix E, is related to BS 4730 : 1974 'Specification for the United Kingdom 7-bit data code'.

IEC 50(351), IEC 359 and ISO 646 are referred to for information purposes only.

CONTENTS

	Page
National foreword	Inside front cover
Cooperating organizations	Back cover

SECTION ONE – GENERAL

Clause		Page
1. Scope and object		4
2. Definitions		5
3. Interface system overview		8

SECTION TWO – FUNCTIONAL SPECIFICATIONS

4. Functional partition	12
5. Notation used to specify interface functions	17
6. Source handshake interface (SH) function	20
7. Acceptor handshake interface (AH) function	24
8. Talker interface (T) function (includes serial poll capabilities)	28
9. Listener interface (L) function	34
10. Service request interface (SR) function	40
11. Remote/local interface (RL) function	42
12. Parallel poll interface (PP) function	46
13. Device clear interface (DC) function	50
14. Device trigger interface (DT) function	52
15. Controller interface (C) function	54
16. Remote message coding and transfer	65

SECTION THREE – ELECTRICAL SPECIFICATIONS

17. Application	69
18. Logical and electrical state relationships	69
19. Driver requirements	69
20. Receiver requirements	70
21. Composite device load requirements	70
22. Ground requirements	73
23. Cable characteristics	73
24. State transition time values	74

SECTION FOUR – MECHANICAL SPECIFICATIONS

25. Application	75
26. Connector type	75
27. Contact assignments	76
28. Device connector mounting	77
29. Cable assembly	77

SECTION FIVE – SYSTEM APPLICATIONS AND GUIDELINES FOR THE DESIGNER

30. System compatibility	80
31. Data rate consideration	81
32. Device capabilities	81
33. AND and OR logic operations	83
34. Address assignment	85
35. Typical combinations of interface functions	85

SECTION SIX – SYSTEM REQUIREMENTS AND GUIDELINES FOR THE USER

36. System compatibility	87
37. System installation requirements	87
38. Address assignment	88
39. Cabling restrictions	89
40. Operational sequences	89

	Pages
APPENDIX A – Typical instrument system	92
APPENDIX B – Handshake process timing sequence	94
APPENDIX C – Interface function allowable subset reference list	98
APPENDIX D – Interface message reference list	103
APPENDIX E – Multiline interface messages: ISO code representation	106
APPENDIX F – Logic circuit implementation	107
APPENDIX G – Parallel poll sequence	109

Notice to the device designer:

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

A device designer shall have sufficient awareness of the characteristics of systems which might include his device in order to select correctly among the options provided in this standard. Likewise, a system configurator shall have sufficient awareness of the options included in each of the devices in his system in order to ensure that 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. Device designers and system users must adhere to the guidelines and requirements specified in Sections Five and Six respectively and the unique device-dependent characteristics of a system to achieve a complete operational system.

2. The interface concepts of this standard have been found to work well under all typical conditions where independently designed equipment is interconnected in a common system. To give added assurance to the user, designers should pay particular attention to these points by allowing adequate design margin: design of cable systems, transient susceptibility on control and management bus signal lines, handling of management and control signals when the source of bus messages changes. Additional guidelines to the designer, on these subjects, are under consideration.

General information on licensing

3. The interface specified by this standard includes patented matter which pertains only to the three-wire handshake as defined throughout Clauses 6 and 7 and summarized in Appendix B.

The International Electrotechnical Commission calls attention to the fact that it is claimed that the three-wire handshake defined throughout Clauses 6 and 7 is the subject of one United States patent and corresponding patents of foreign countries owned by the Hewlett-Packard Company. Although these patents appear to cover the subject of Clauses 6 and 7 in this IEC publication, the IEC takes no position with respect to patent validity. The Hewlett-Packard Company has assured the IEC that it is willing to grant a licence under these patents on reasonable and non-discriminatory terms and conditions to anyone wishing to obtain such a licence.

The Hewlett-Packard Company's undertakings (policy letter on licencing, the licence offer, and the form of licence) in this respect are on file with the International Electrotechnical Commission and available for inspection by all interested parties at the IEC Central Office.

The licence details may be obtained from the Legal Department of the Hewlett-Packard Company whose address is:

1501 Page Mill Road,
Palo Alto, California, 94304,
USA.

British Standard

An interface system for programmable measuring instruments (byte serial, bit parallel)

Part 1. Specification for functional, electrical, and mechanical requirements, system applications and requirements for designer and user

SECTION ONE – GENERAL

1. Scope and object

1.1 *Scope*

1.1.1 This standard is applicable to interface systems used to interconnect both programmable and non-programmable electronic measuring apparatus with other apparatus and accessories necessary to assemble instrumentation systems.

1.1.2 This standard is applicable to the interface of instrumentation systems, or portions of them, in which:

- the data exchanged among the interconnected apparatus is digital (as distinct from analogue);
- the number of devices that may be interconnected by one contiguous bus does not exceed 15;
- the total transmission path length over the interconnecting cables does not exceed 20 m;
- the data rate across the interface on any signal line does not exceed 1 megabit per second.

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

1.1.3 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.

1.1.4 This standard applies generally to laboratory and production test environments which are both electrically quiet and restricted as to physical dimensions (distances between the system components).

- 1.1.5 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 standard may be also applied to interconnect the internal subsections within a self-contained equipment.

1.2 Object

This standard is intended:

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

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

Note. – For the latter two items, reference is made to IEC Publication 348: Safety Requirements for Electronic Measuring Apparatus, and IEC Publication 359: Expression of the Functional Performance of Electronic Measuring Equipment.

2. Definitions

Throughout this standard, and in so far as further distinction is not necessary:

- the term “system” denotes the byte-serial, bit-parallel interface system that, in general, includes all circuits, cables, connectors, message repertoire, and control protocol to effect unambiguous data transfer between devices;
- 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.

The following definitions apply for the purpose of this standard.

This clause contains only general definitions. Detailed definitions are given under the headings of further clauses and sub-clauses, as appropriate.

2.1 General system terms

2.1.1 System (from International Electrotechnical Vocabulary (I.E.V.) 351-01-01)

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

Note. – The system is considered to be separated from the environment and other external systems by an imaginary surface which cuts the links between them and the considered system. Through these links, the system is affected by the environment, is acted upon by external systems, or acts itself on the environment or the external systems.

2.1.2 Interface

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

2.1.3 Interface system

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

2.1.4 Programmable

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

2.1.5 Remote control

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

2.1.6 Local control

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

2.1.7 Compatibility

The degree to which devices may be interconnected and used, without modification, when designed as defined throughout this standard (e.g. mechanical, electrical, functional).

2.1.8 Handshake cycle

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

2.1.9 Programmable measuring apparatus

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

2.1.10 *Terminal unit*

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

2.2 *Signals and paths*

2.2.1 *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 the general sense, and hereinafter refers to digital electrical signals only.

2.2.2 *Signal parameter*

That parameter of an electrical quantity whose value or sequence of values conveys information.

2.2.3 *Signal level*

The magnitude of a signal compared to an arbitrary reference magnitude (voltage in the case of this standard).

2.2.3.1 *High state*

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

2.2.3.2 *Low state*

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

2.2.4 *Signal line*

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

2.2.5 *Byte*

A group of concurrent binary digits operated on as a unit and usually shorter than a computer word. (Frequently connotes a group of eight bits.)

2.2.6 *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.

2.2.6.1 *Unidirectional bus*

A bus used by any individual device for one-way transmission of messages only, i.e. either input only or output only.

2.2.6.2 *Bidirectional bus*

A bus used by any individual device for two-way transmission of messages, i.e. both input and output.

2.2.7 *Byte serial*

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

2.2.8 *Bit parallel*

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

3. Interface system overview

3.1 *Interface system objective*

3.1.1 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.

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

- messages used to manage the interface system itself, hereinafter called *interface messages*;
- 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.

3.2 *Fundamental communication capabilities*

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

- a device acting as a *listener*;
- a device acting as a *talker*;
- a device acting as a *controller*.

Note. – The terms *talker* and *listener* have specialized meanings in this standard to better describe the role of devices. These terms are used in order to avoid confusion with other technical terms such as transmitter and receiver.

3.2.2 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 Section Two to distinguish between different types of controller capabilities used in the interface system.

3.2.3 Listener, talker, and controller capabilities occur individually or in any combination in devices interconnected via the interface system as shown in Figure 1, page 25.

3.2.4 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 requires 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, on command from the controller. This may occur simultaneously with either selected devices or all devices in a system.
- d) The remote/local function provides a device with the ability to select between programme data from the bus and local data (e.g. front panel controls).

3.3 *Message paths and bus structure*

3.3.1 The interface system contains a set of 16 signal lines used to carry all information, interface messages and device-dependent messages, among interconnected devices.

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

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

- data bus, eight signal lines;
- data byte transfer control bus, three signal lines;
- interface management bus, five signal lines.

Figure 1 illustrates the basic communication paths.

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

a) DATA INPUT OUTPUT 1 (DIO1);

h) DATA INPUT OUTPUT 8 (DIO8).

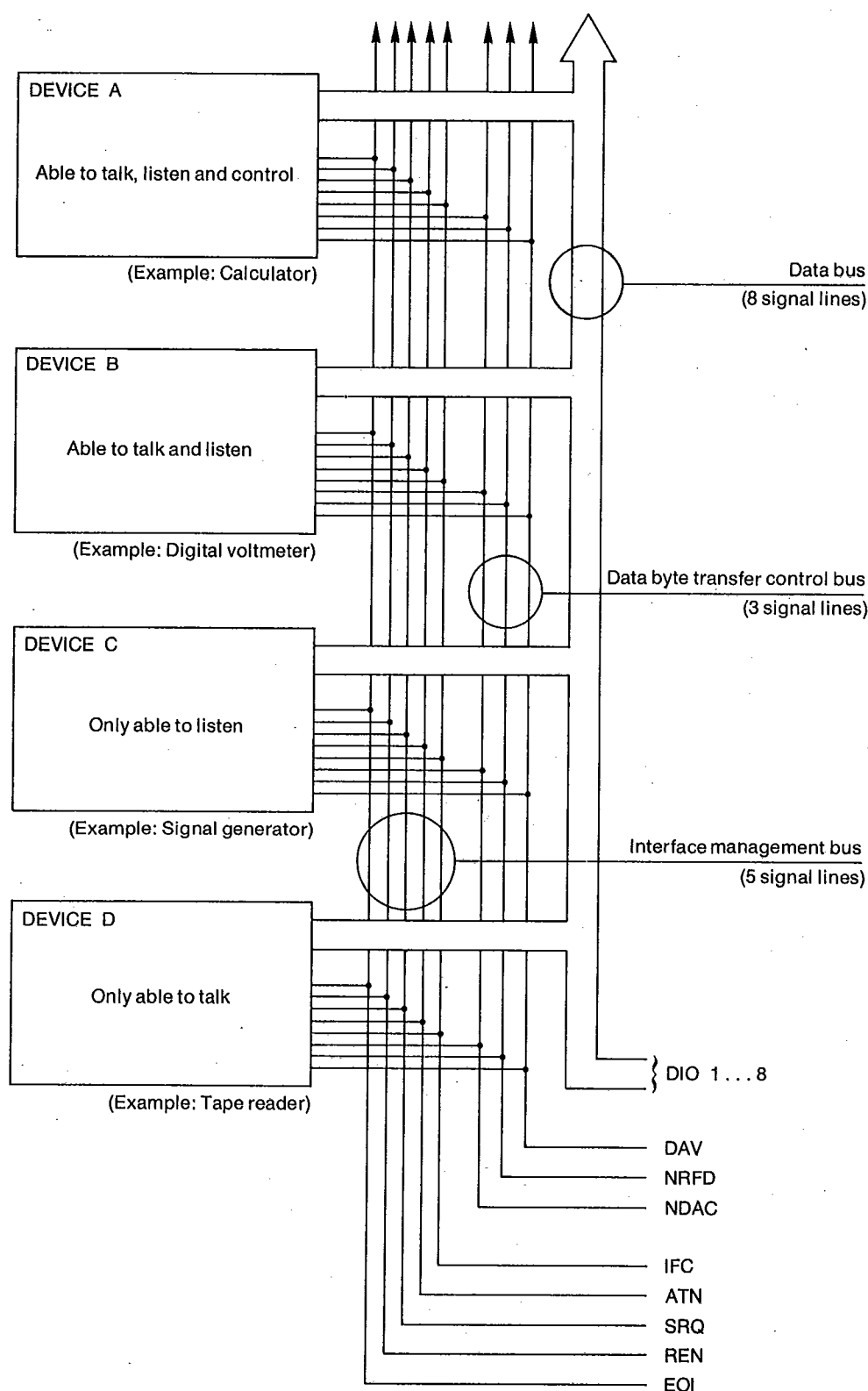


FIG. 1. – Interface capabilities and bus structure.

Message bytes are carried on the DIO signal lines:

- in a bit-parallel byte-serial form;
- asynchronously;
- bidirectionally.

Note. – A message may be carried on an individual DIO signal line when required.

3.3.3.2 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.
- 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 process to transfer each data byte across the interface.

3.3.3.3 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 select between two alternate sources of device programming data.
- 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 parallel poll.

3.4 *Interface system elements*

The primary elements of this interface system are:

- functional elements;
- electrical elements;
- mechanical elements.

Each is described in a following section.

SECTION TWO – FUNCTIONAL SPECIFICATIONS

4. 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:

- device functions (definition is application-dependent);
- interface functions (definition is application-independent);
- message coding logic.

All communication to or from interface functions is defined in terms of messages and state linkages (see Sub-clause 4.3).

All messages carried on the signal lines are coded according to the coding logic defined in Clause 16.

4.1 Device functions

The scope, purpose, size, content, and organization of the device function area (e.g. analogue signal measurement capability, range, modes of operation, etc.) are beyond the scope of this standard. Figure 2, page 31, 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.

4.2 Interface function concepts

4.2.1 Interface function

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 each particular class of messages.

4.2.2 Interface function state

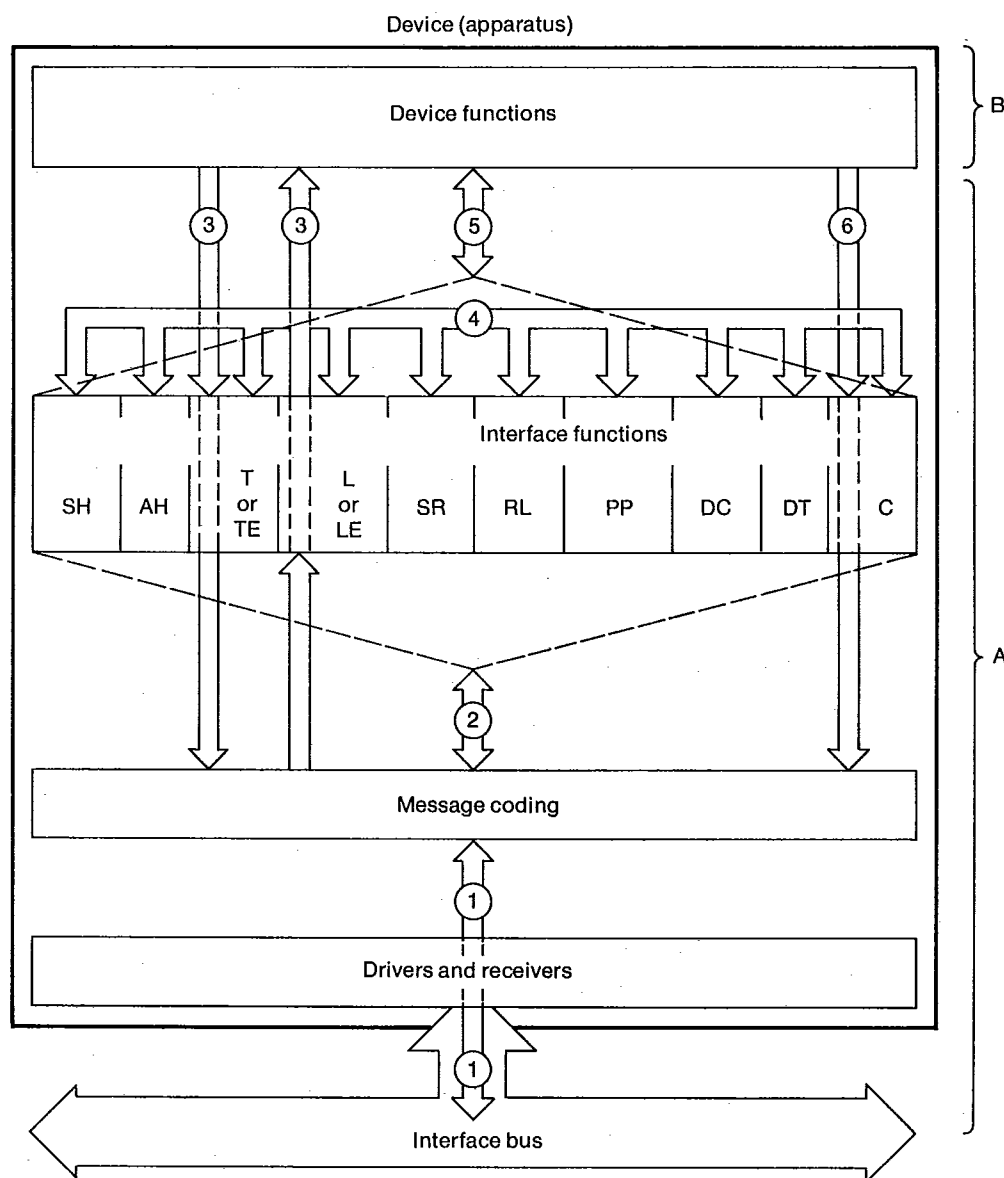
4.2.2.1 Each of the interface functions is defined in terms of one or more groups of interconnected, mutually exclusive *states*.

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

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

- messages which may or must be sent over the interface while that state is active, and
- 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 in this standard; messages from interface functions are specified by the designer)
 6 = remote interface messages sent by device functions within a controller

FIG. 2. — Partition of functions within a device.

4.2.3 Interface function repertoire

4.2.3.1 The designer is given the choice to select the particular set of interface functions necessary to fit the particular device application area. Figure 2, page 31, and Table I identify the available interface functions.

TABLE I

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

4.2.3.2 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.

4.2.4 Interface function assumptions and perspective

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

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

4.2.4.3 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.

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

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

4.2.4.6 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.

4.3 *Message concepts*

4.3.1 *Message*

Each message represents a quantity of information and will be received either true or false at any specific time.

All communication between an interface function and its environment is accomplished through messages sent or received.

4.3.2 *Local message route and content*

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

4.3.2.2 Local messages flow between device functions and interface functions; see Figure 2, page 31, message route 5.

Note. – Certain local messages are conveyed as remote messages and vice versa.

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

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

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

4.3.3 *Remote message route and content*

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

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

4.3.3.3 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.

4.3.3.4 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.

4.3.4 *State linkage route and content*

4.3.4.1 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.

4.3.5 *Message coding*

4.3.5.1 Message coding is the act of translating remote messages to or from interface signal line values.

4.3.5.2 A message sent over a single signal line is called a *uniline message*. Two or more of these messages can be sent concurrently.

4.3.5.3 A message which 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.

4.3.6 *Classification of multiline messages*

4.3.6.1 Multiline messages are interpreted as interface messages when the ATN message is true.

4.3.6.2 Multiline messages are interpreted as device-dependent messages when the ATN message is false.

4.3.6.3 Each device, when the ATN message is true, may accept and process any or all of these specific classes of multiline messages as determined both by the extent of the interface function capability contained within the device and the current set of active interface function states:

- universal commands (all devices independent of interface function states);
- addressed commands (all devices addressed to listen);
- addresses (all devices);
- secondary addresses or commands (all devices enabled by a primary address or command).

For a list of specific commands, see Table XXXVI, page 121.

4.3.6.4 Each device, when the ATN message is false, may accept and process its own device-dependent message when addressed to listen.

4.3.7 *Message transfer conventions*

4.3.7.1 *Remote message transfer conventions*

- a) The value (true or false) of all remote 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 the remote message coding Table XXXVIII, page 139.
- c) Since normal interface operation allows two or more devices to simultaneously send opposite values of the remote messages, a technique must be provided for resolving these conflicts. This is accomplished by implementing two types of message transfer over the interface, *active* transfer and *passive* transfer. The interface is structured so that in all conflicts between two message values, one of them will be active and the other passive. Messages must 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:
- An *active true* value being sent is guaranteed to be the value received (the device need not allow it to be overridden).
 - A *passive true* value being sent is not guaranteed to be the value received and the device must allow it to be overridden.
 - An *active false* value being sent is guaranteed to be the value received (the device need not allow it to be overridden).
 - 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 Clause 33).
- 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 line (see Clause 33).
- h) Only the multiline message(s) to be sent true will be specified for an interface function state since multiline messages (sent via the DIO lines) are by their nature mutually exclusive. It should be understood that all unspecified multiline messages are sent passive false while that state is active.

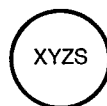
4.3.7.2 Local message transfer conventions

- 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.

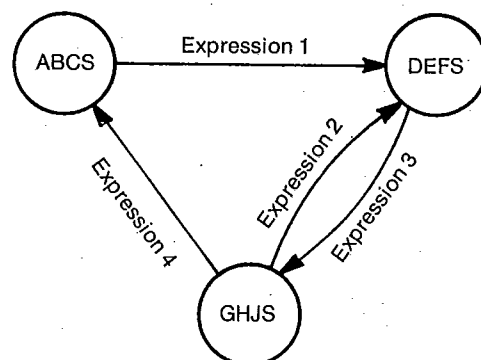
5. Notation used to specify interface functions

5.1 State diagram notation

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

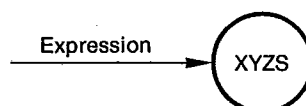


- 5.1.2 All permissible *transitions* between states of an interface function are represented graphically by arrows between them.
- 5.1.3 Each transition is qualified by an *expression* whose value is either true or false. The interface function must remain in its current state if all expressions which qualify transitions leading to other states are false. The interface function must 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.



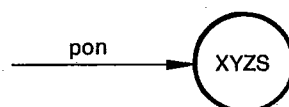
- 5.1.3.1 An expression consists of one or more local messages, remote messages, state linkages, or minimum time limits used in conjunction with the operators AND, OR, or NOT.
- 5.1.3.2 A local message to an interface function is represented by a three-letter mnemonic written in lower case; for example, *rdy*. (Within the text, local message mnemonics are written in italics for clarity; e.g. *rdy*.)
- 5.1.3.3 A remote message (received via the interface) is represented by a three-letter mnemonic written in upper case; for example ATN.
- 5.1.3.4 A linkage from another state diagram is represented by a four-letter mnemonic enclosed in an oval; for example (LACS). A state linkage is true if the enclosed state is currently active; otherwise, it is false.
- 5.1.3.5 A minimum time-limit is represented by the symbol "*Tn*". 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 XXXIX, page 155.
- 5.1.3.6 The "AND" operator is represented by the symbol " $\wedge$ ".
- 5.1.3.7 The "OR" operator is represented by the symbol " $\vee$ ".
- 5.1.3.8 The "AND" operator takes precedence over the "OR" operator within an expression unless otherwise specified by parentheses.
- 5.1.3.9 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.
- 5.1.4 If a transition is further qualified by a maximum time-limit "(within *tn*)", then the state pointed to must be entered within the specified amount of time after the expression becomes true. The values for these time-limits are contained in Table XXXIX.
- 5.1.5 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 "[...]".

- 5.1.6 If a specific expression causes a transition to a state from all other states of the diagram, a shorthand notation is used instead of all the individual transitions being drawn. An arrow without a state at its origin is used to represent this condition, and is assumed to originate in all states (i.e. $\overline{IFC}$ or $\overline{REN}$). It is further assumed that these same expressions are false (i.e. $\overline{IFC}$ or $\overline{REN}$) to permit all other transitions in the diagram, and therefore they are omitted from the diagrams.

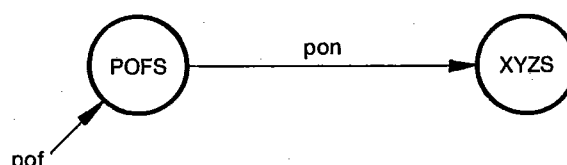


- 5.1.7 Although "power-off" (POFS) is a valid state of most interface functions and should normally be shown on all diagrams with a transition leading to the state to be entered at "power-on" time, a shorthand form is used showing the "pon" pseudo-message originating a transition to the first state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

- abbreviated notation used on state diagram:



- complete representation implied by above symbol:



5.2 Message output notation

- 5.2.1 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.
- 5.2.2 Rows of the table are used to indicate states of the interface function.
- 5.2.3 Columns of the table are used to indicate remote messages allowed to be sent during at least one state of the interface function.
- 5.2.4 Each table entry indicates the value of a message that shall be sent while a specified state is active:
- "T" indicates active true.
 - "F" indicates active false.
 - "(T)" indicates passive true.
 - "(F)" indicates passive false.

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

5.2.6 A separate column for device function interaction summarizes the corresponding types of messages (or resultant action) device functions are allowed to send or receive. Local messages 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.

6. Source handshake interface (SH) function

6.1 General description

The SH function provides a device with the capability to guarantee the proper transfer of multiline messages. An interlocked handshake sequence between the SH function and one or more Acceptor Handshake Interface Functions (each contained within separate devices) guarantees asynchronous transfer of each multiline message. The SH function controls the initiation of, and termination of, the transfer of a multiline message byte. This function utilizes the DAV, RFD, and DAC messages to effect each message byte transfer.

6.2 Source handshake function state diagram

The SH function shall be implemented so as to perform according to the state diagram given in Figure 3, page 47, and the state descriptions given throughout Clause 6. Table II specifies the set of messages and states required to effect transition from one active state to another. Table III specifies the messages that shall be sent and the device function interaction required while each state is active.

6.3 Source handshake function state descriptions

6.3.1 SOURCE IDLE STATE (SIDS)

6.3.1.1 In the SIDS state the SH function is not engaged in the handshake cycle and does not have a new message byte available.

The SH function powers on in the SIDS state.

6.3.1.2 In the SIDS state the DAV message shall be sent passive false.

6.3.1.3 The SH function shall exit the SIDS state and enter the SOURCE GENERATE STATE (SGNS) if:

- the TALKER ACTIVE STATE (TACS) is active;
- or the SERIAL POLL ACTIVE STATE (SPAS) is active;
- or the CONTROLLER ACTIVE STATE (CACS) is active.

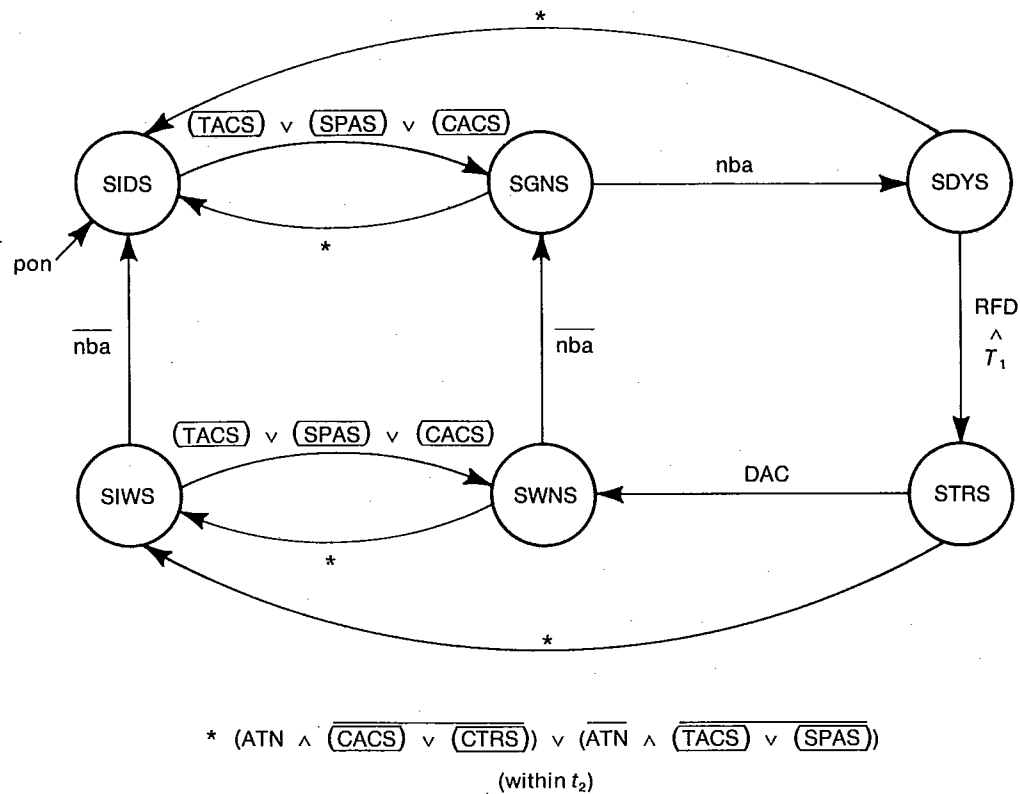


FIG. 3. – Source handshake state diagram.

TABLE II

SH mnemonics

Messages:	Interface states:
pon – power on	SDYS – SOURCE DELAY STATE
nba – new byte available	STRS – SOURCE TRANSFER STATE
ATN – ATTENTION	SWNS – SOURCE WAIT FOR NEW CYCLE STATE
RFD – READY FOR DATA	SIWS – SOURCE IDLE WAIT STATE
DAC – DATA ACCEPTED	$(\overline{\text{TACS}})$ – TALKER ACTIVE STATE (T FUNCTION)
	$(\overline{\text{SPAS}})$ – SERIAL POLL ACTIVE STATE (T FUNCTION)
	$(\overline{\text{CACS}})$ – CONTROLLER ACTIVE STATE (C FUNCTION)
	$(\overline{\text{CTRS}})$ – CONTROLLER TRANSFER STATE (C FUNCTION)
Interface states:	
SIDS – SOURCE IDLE STATE	
SGNS – SOURCE GENERATE STATE	

TABLE III

SH message outputs

SH state	Remote message sent	Device function interaction
	DAV	
SIDS	(F)	Device function can change remote multiline messages
SGNS	F	Device function 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	Device function requested to change multiline messages
SIWS	(F)	Device function requested to change multiline messages

6.3.2 SOURCE GENERATE STATE (SGNS)

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

6.3.2.2 In the SGNS state the SH function shall send the DATA VALID (DAV) message false. In this state the device may change the multiline message being sent via the Talker or Controller Interface Function while in the TACS or CACS states.

6.3.2.3 The SH function shall exit the SGNS state and enter:

- a) the SOURCE DELAY STATE (SDYS) if the *new byte available (nba)* message is true;
- b) the SIDS state within t_2 if either:
 - the ATTENTION (ATN) message is true and neither the CACS state nor the CTRS state is active;
 - or the ATN message is false and neither the TACS state nor the SPAS state is active.

6.3.3 SOURCE DELAY STATE (SDYS)

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

6.3.3.2 In the SDYS state the SH function shall send the DAV message false. In this state the device shall not change the multiline message being sent.

6.3.3.3 The SH function shall exit the SDYS state and enter:

- a) the SOURCE TRANSFER STATE (STRS) only after T_1 if the READY FOR DATA (RFD) message is true;
- b) the SIDS state within t_2 if either:
 - the ATTENTION (ATN) message is true and neither the CACS state nor the CTRS state is active;
 - or the ATN message is false and neither the TACS state nor the SPAS state is active.

6.3.4 SOURCE TRANSFER STATE (STRS)

6.3.4.1 In the STRS state the SH function indicates to the acceptor handshake function that it is continuously sending a valid message byte.

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

6.3.4.3 The SH function shall exit the STRS state and enter:

- a) the SOURCE IDLE WAIT STATE (SIWS) within t_2 if either:
 - the ATTENTION (ATN) message is true and neither the CACS state nor the CTRS state is active;
 - or the ATN message is false and neither the TACS state nor the SPAS state is active;

Note. – This implies an asynchronous interrupt (see Sub-clause 15.5).

- b) the SOURCE WAIT FOR NEW CYCLE STATE (SWNS) if the DATA ACCEPTED (DAC) message is true.

6.3.5 SOURCE WAIT FOR NEW CYCLE STATE (SWNS)

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

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

6.3.5.3 The SH function shall exit the SWNS state and enter:

- a) the SGNS state if the *nba* message is false;
- b) the SIWS state within t_2 if either:
 - the ATTENTION (ATN) message is true and neither the CACS state nor the CTRS state is active;
 - or the ATN message is false and neither the TACS state nor the SPAS state is active.

6.3.6 SOURCE IDLE WAIT STATE (SIWS)

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

6.3.6.2 In the SIWS state the DAV message shall be sent passive false.

6.3.6.3 The SH function shall exit the SIWS state and enter:

- a) the SIDS state if the *nba* message is false;
- b) the SWNS state if either:
 - the TACS state is active;
 - or the SPAS state is active;
 - or the CACS state is active.

6.4 Source handshake function allowable subsets

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

TABLE IV

Identification	Description	States omitted	Other requirements	Other function subsets required
SH0 SH1	No capability Complete capability	All None	None None	None T1-T8 or TE1-TE8 or C5-C28

6.5 Additional SH function requirements and guidelines

6.5.1 Interpretation of the local nba messages

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

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

6.5.2 Alternate transition to idle states

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

7. Acceptor handshake interface (AH) function

7.1 General description

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

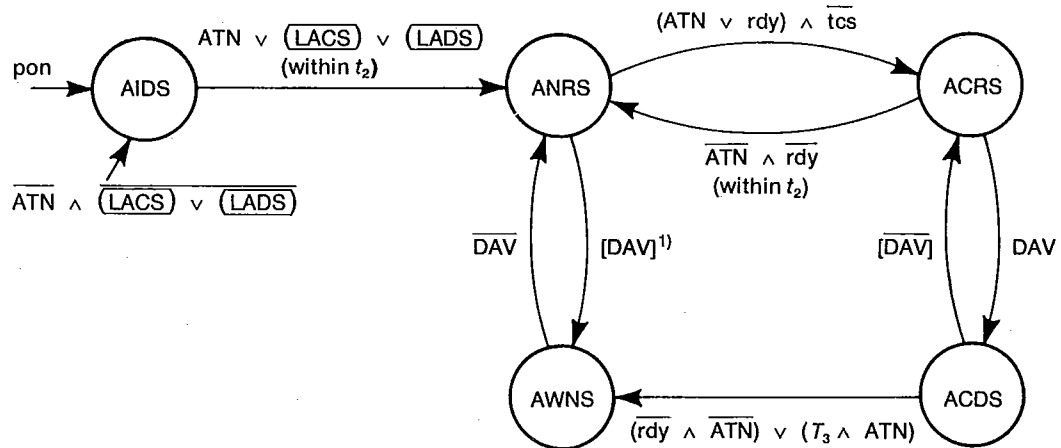
7.2 Acceptor handshake function state diagram

The AH function shall be implemented so as to perform according to the state diagram given in Figure 4, page 55, and the state descriptions given throughout Clause 7. Table V specifies the set of messages and states required to effect transition from one active state to another. Table VI specifies the messages that shall be sent and the device function interaction required while each state is active.

7.3 Acceptor handshake function state descriptions

7.3.1 ACCEPTOR IDLE STATE (AIDS)

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



¹⁾ This transition will never occur under normal interface operation; however it may be implemented to simplify the interface function design.

FIG. 4. – Acceptor handshake state diagram.

TABLE V

AH mnemonics

Messages:	Interface states:
pon – power on	AIDS – ACCEPTOR IDLE STATE
rdy – ready for next message	ANRS – ACCEPTOR NOT READY STATE
tcs – take control synchronously ¹⁾	ACRS – ACCEPTOR READY STATE
ATN – ATTENTION	ACDS – ACCEPT DATA STATE
DAV – DATA VALID	AWNS – ACCEPTOR WAIT FOR NEW CYCLE STATE
	(LADS) – LISTENER ADDRESSED STATE (L FUNCTION)
	(LACS) – LISTENER ACTIVE STATE (L FUNCTION)

¹⁾ See Sub-clause 15.3.7.1.

TABLE VI

AH message outputs

AH state	Remote message sent		Device function interaction
	RFD	DAC	
AIDS	(T)	(T)	Device function can receive neither remote multiline nor END messages
ANRS	F	F	Device function can receive neither remote multiline nor END messages
ACRS	(T)	F	Device function can receive neither remote multiline nor END messages
AWNS	F	(T)	Device function can receive neither remote multiline nor END messages
ACDS	F	F	Device function can receive remote multiline messages if LACS is active

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

7.3.1.3 The AH function shall exit the AIDS state and enter the ACCEPTOR NOT READY STATE (ANRS) within t_2 if either:

- the ATN message is true;
- or the LACS state is active;
- or the LADS state is active.

7.3.2 ACCEPTOR NOT READY STATE (ANRS)

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

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

7.3.2.3 The AH function shall exit the ANRS state and enter:

- a) the ACRS state if the *take control synchronously (tcs)* message is false (see Sub-clause 15.3.7.1) and either:
 - the ATN message is true;
 - or the *rdy* message is true;
- b) the AIDS state if the ATN message is false and neither:
 - the LADS state is active;
 - nor the LACS state is active;
- c) the AWNS state if the DAV message is true (optional, note that this transition will never occur under normal interface operation).

7.3.3 ACCEPTOR READY STATE (ACRS)

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

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

7.3.3.3 The AH function shall exit the ACRS state and enter:

- a) the ACCEPT DATA STATE (ACDS) if the DAV message is true;
- b) the AIDS state if the ATN message is false and neither:
 - the LADS state is active;
 - nor the LACS state is active;
- c) the ANRS state within t_2 if both the ATN and the *ready for next message (rdy)* messages are false.

7.3.4 ACCEPT DATA STATE (ACDS)

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

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

7.3.4.3 The AH function shall exit the ACDS state and enter:

- a) the ACCEPTOR WAIT FOR NEW CYCLE STATE (AWNS) if either:
 - the ATN message is true and a period of T_3 has elapsed;
 - or the ATN and *rdy* messages are both false;
- b) the AIDS state if the ATN message is false and neither:
 - the LADS state is active;
 - nor the LACS state is active;
- c) the ACRS state if the DAV message is false (optionally, this transition can occur only when the controller takes control asynchronously).

7.3.5 ACCEPTOR WAIT FOR NEW CYCLE STATE (AWNS)

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

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

7.3.5.3 The AH function shall exit the AWNS state and enter:

- a) the ACCEPTOR NOT READY STATE (ANRS) if DAV is false;
- b) the AIDS state if the ATN message is false and neither:
 - the LADS state is active;
 - nor the LACS state is active.

7.4 Acceptor function allowable subsets

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

TABLE VII

Identification	Description	States omitted	Other requirements	Other function subsets required
AH0	No capability	All	None	None
AH1	Complete capability	None	None	None

7.5 Additional AH function requirements and guidelines

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

7.5.2 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 Clause 33.

7.5.3 Since interface functions need be designed only "so as to perform according to" the state diagrams specified, it is not required that exactly the states specified are the ones which exist in an implementation. One consequence of this statement is that interface function state transitions which are qualified by interface messages can occur after the message has been received as long as the RFD message is held false until they occur. The resulting performance cannot be distinguished from the performance of the specified diagrams in which the transitions must occur while the interface message is being received. If this type of implementation is chosen, then the AH function should remain in the ANRS state even though the exit condition is true in order to hold the RFD message false (this is allowed by Sub-clause 4.2.4.6).

8. Talker interface (T) function (includes serial poll capabilities)

8.1 General description

8.1.1 The T 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. The capability exists only when the T function is addressed to talk.

8.1.2 There are two alternative versions of the functions: one with and one without address extension. The normal T function uses a one-byte address. The T function with address extension (hereinafter called an Extended Talker (TE) function) uses a two-byte address. In all other respects, the capabilities of both versions are the same.

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

Note.— Both the T function and TE function are described concurrently throughout Clause 8 due to the extensive similarity between these two functions.

8.2 Talker function state diagrams

8.2.1 The T function shall be implemented so as to perform according to the state diagrams given in Figure 5, page 63, and the state descriptions given throughout Clause 8. Table VIII specifies the set of messages and states required to effect transition from one active state to another. Table IX specifies the messages that shall be sent and the device function interaction required while each state is active.

8.2.2 The TE function shall be implemented so as to perform according to the state diagrams given in Figure 6, page 65, and the state descriptions given throughout Clause 8. Table X specifies the set of messages and states required to effect transition from one active state to another. Table IX specifies the messages that shall be sent and the device function interaction required while each state is active.

8.3 Talker function state descriptions

8.3.1 TALKER IDLE STATE (TIDS)

8.3.1.1 In the TIDS state either the T function or the TE function is not engaged in sending data or status bytes. The T function or the TE function powers on in the TIDS state.

8.3.1.2 In the TIDS state the END and RQS messages shall be sent passive false and the NUL message shall be sent passive true.

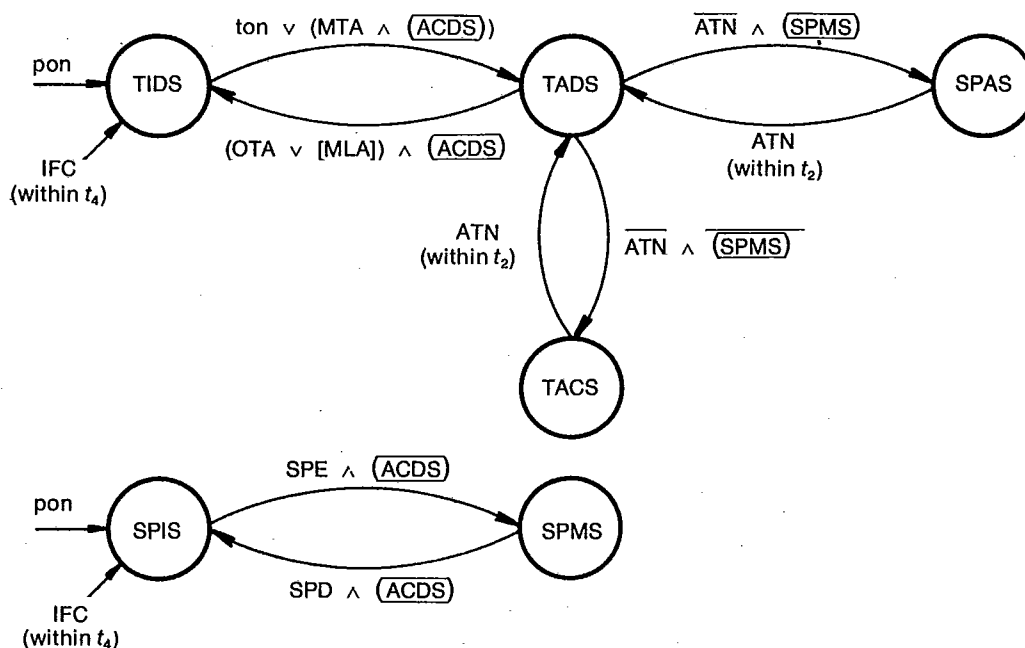


FIG. 5. – Talker state diagram.

TABLE VIII

T mnemonics

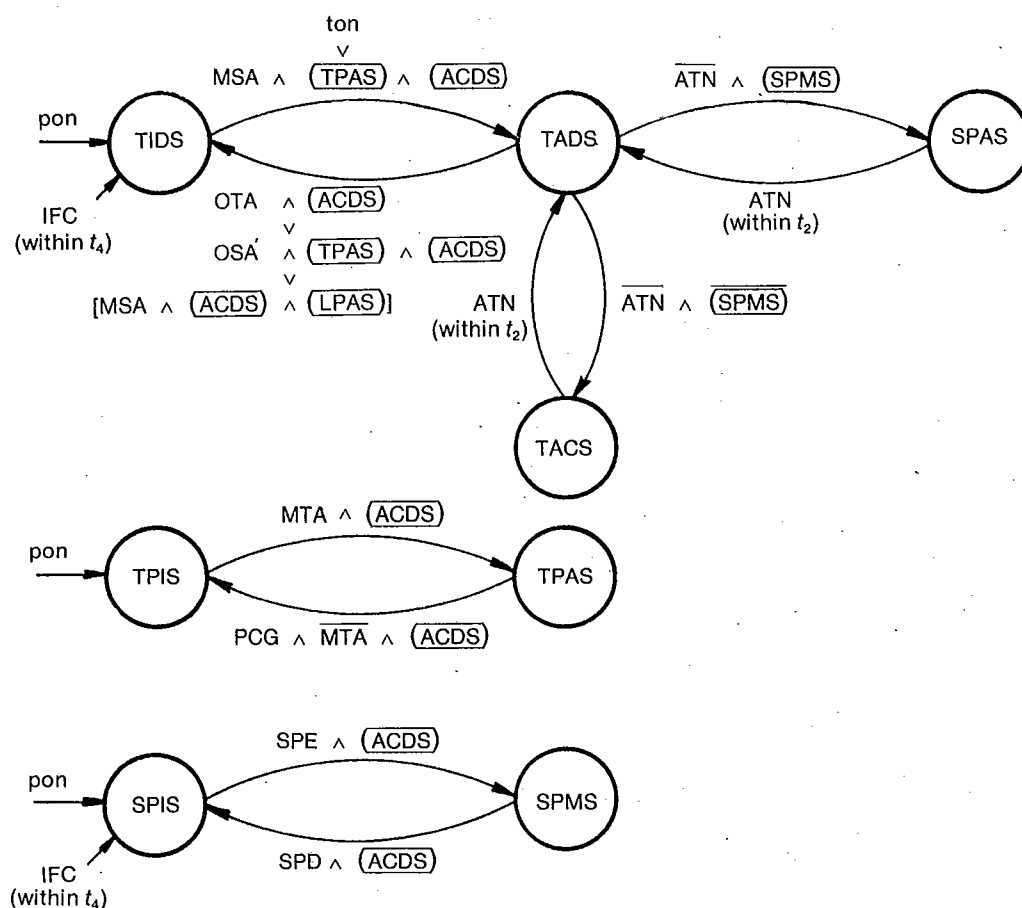
Messages:	Interface states:
pon – power on	TIDS – TALKER IDLE STATE
ton – talk only	TADS – TALKER ADDRESSED STATE
IFC – INTERFACE CLEAR	TACS – TALKER ACTIVE STATE
ATN – ATTENTION	SPAS – SERIAL POLL ACTIVE STATE
MTA – MY TALK ADDRESS	SPIS – SERIAL POLL IDLE STATE
SPE – SERIAL POLL ENABLE	SPMS – SERIAL POLL MODE STATE
SPD – SERIAL POLL DISABLE	(ACDS) – ACCEPT DATA STATE (AH FUNCTION)
OTA – OTHER TALK ADDRESS	
MLA – MY LISTEN ADDRESS	

TABLE IX

T or TE message outputs

T State	Qualifier	Remote message sent ²⁾			Device function interaction (DF)
		Multiline	END	RQS ⁴⁾	
TIDS		(NUL)	(F)	(F)	Device function not allowed to send messages
TADS		(NUL)	(F)	(F)	Device function not allowed to send messages
TACS		DAB ¹⁾ or EOS ¹⁾	T or F ¹⁾	(F)	Device function can send DAB, EOS, or END message(s) concurrent with DAB ³⁾
SPAS	APRS ⁵⁾ inactive	STB ¹⁾	F or T	F	Device function can send one STB message ³⁾
SPAS	APRS ⁵⁾ active	STB ¹⁾	F or T	T	Device function can send one STB message ³⁾

¹⁾ Messages enabled by the T function originating within the device functions.²⁾ See Table XXXVIII (Clause 16).³⁾ Under SH control.⁴⁾ See Sub-clause 8.3.4.1.⁵⁾ See Sub-clause 10.3.3.



Note. – If the TE function is used together with the L function then $[MSA \wedge \overline{ACDS} \wedge \overline{LPAS}]$ shall be replaced by $[MLA \wedge \overline{ACDS}]$.

FIG. 6. – Extended talker state diagram.

TABLE X
TE mnemonics

Messages:	Interface states:
pon – power on	TIDS – TALKER IDLE STATE
ton – talk only	TADS – TALKER ADDRESSED STATE
IFC – INTERFACE CLEAR	TACS – TALKER ACTIVE STATE
ATN – ATTENTION	SPAS – SERIAL POLL ACTIVE STATE
MTA – MY TALK ADDRESS	TPIS – TALKER PRIMARY IDLE STATE
OTA – OTHER TALK ADDRESS	TPAS – TALKER PRIMARY ADDRESSED STATE
OSA – OTHER SECONDARY ADDRESS	SPIS – SERIAL POLL IDLE STATE
PCG – PRIMARY COMMAND GROUP	SPMS – SERIAL POLL MODE STATE
SPE – SERIAL POLL ENABLE	$\overline{ACDS}$ – ACCEPT DATA STATE (AH FUNCTION)
SPD – SERIAL POLL DISABLE	$\overline{LPAS}$ – LISTENER PRIMARY ADDRESSED STATE (L FUNCTION)
MSA – MY SECONDARY ADDRESS	

T or TE message outputs (see Table IX)

8.3.1.3 The T function shall exit the TIDS state and enter the TALKER ADDRESSED STATE (TADS) if either:

- the MY TALK ADDRESS (MTA) message is true and the ACDS state is active;
- or the *talk only* (*ton*) message is true (see Sub-clause 8.5.3).

8.3.1.4 The TE function shall exit the TIDS state and enter the TADS state if either:

- the MY SECONDARY ADDRESS (MSA) message is true, and the ACDS state is active, and the TALKER PRIMARY ADDRESS STATE (TPAS) is active;
- or the *ton* message is true.

8.3.2 TALKER ADDRESSED STATE (TADS)

8.3.2.1 In the TADS state the T function has received its talk address and is prepared for, but not engaged in, sending data or status bytes. In the TADS state 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.

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

8.3.2.3 The T function shall exit the TADS state 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 the SPMS state is active;
- c) the TIDS state if either:
 - the OTHER TALK ADDRESS (OTA) message is true and the ACDS state is active;
 - or the MLA message is true and the ACDS state is active;
 - or within t_4 if the IFC message is true.

Note. – Use of the MLA message is optional.

8.3.2.4 The TE function shall exit the TADS state and enter:

- a) the TACS state if the ATN message is false and the SPMS state is inactive;
- b) the SPAS state if the ATN message is false and the SPMS state is active;
- c) the TIDS state if either:
 - the OTA message is true and the ACDS state is active;
 - or the OTHER SECONDARY ADDRESS (OSA) message is true and the TPAS and ACDS states are active;
 - or the MSA message is true and both the LISTENER PRIMARY ADDRESSED STATE (LPAS) and ACDS states are active;
 - or within t_4 if the INTERFACE CLEAR (IFC) message is true.

Note. – Use of the expression containing the MSA message is optional. See also note under Figure 6, page 65.

8.3.3 TALKER ACTIVE STATE (TACS)

8.3.3.1 In the TACS state the T function, or the TE function, enables the transfer of the 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).

8.3.3.2 During the TACS state the DATA BYTE (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.

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

- a) the TADS state within t_2 if the ATN message is true;
- b) the TIDS state within t_4 if the IFC message is true.

8.3.4 SERIAL POLL ACTIVE STATE (SPAS)

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

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

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

Note. – The APRS state is contained in the Service Request Interface Function.

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

- a) the TADS state within t_2 if the ATN message is true;
- b) the TIDS state within t_4 if the IFC message is true.

8.3.5 SERIAL POLL IDLE STATE (SPIS)

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

8.3.5.2 The SPIS state does not provide a remote message sending capability.

8.3.5.3 The T function or the TE function shall exit the SPIS state and enter the SPMS state if the SERIAL POLL ENABLE (SPE) message is true and the ACDS state is active.

8.3.6 SERIAL POLL MODE STATE (SPMS)

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

8.3.6.2 The SPMS state does not provide a remote message-sending capability.