

IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers

Sponsor

Microprocessor and Microcomputer Standards Committee
of the
IEEE Computer Society

Approved 21 September 2000

Reaffirmed 7 December 2011

IEEE-SA Standards Board

Abstract: A signaling method for asynchronous, fully interlocked, bidirectional parallel communications between hosts and printers or other peripherals is defined. A functional subset of the signaling method may be implemented on personal computers (PCs) or equivalent parallel port hardware with new software. New electrical interfaces, cabling, and interface hardware that provides improved performance while retaining backward compatibility with this subset is detailed.

Keywords: bidirectional parallel communications, computers, interfaces, PCs, personal computers, printers

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2000 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 24 October 2000. Printed in the United States of America.

Print: ISBN 0-7381-2615-2 SH94880
PDF: ISBN 0-7381-2616-0 SS94880

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

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

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

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

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

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

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

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying 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 is the sole entity that may authorize the use of certification marks, trademarks, or other designations to indicate compliance with the materials set forth herein.

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; (978) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

(This introduction is not a part of IEEE Std 1284-2000, IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers.)

This standard was formally started as an IEEE effort in January 1992, but without the advance work done by a loose alliance of printer manufacturers and printer software developers called the Network Printing Alliance, this standard would not be possible.

The following lists the key contributors to the revision of this standard:

Forrest D. Wright, *Chair*

Larry Stein, *Secretary*

Lance Spaulding, *Editor*

Darrell Cox
Lee Farrell
Robert Gross
Laurie Lasslo

Hideki Morozumi
Bill Myntti
Fumio Nagasaka
David Roach
Bill Russell

Greg Shue
Jerry Thrasher
Rick Vander Wegen
Craig Whittle

The following members of the balloting committee voted on this standard:

Steven R. Bard
Lon Canaday
Keith Chow
James R. Davis
Sourav K. Dutta
Edwin Vivian El-Kareh
Lee Farrell
Thomas M. Kurihara

Joseph R. Marshall
Gene E. Milligan
Robert Mortonson
Klaus-Dieter Mueller
Atsushi Nakamura
Roman Orzol
Gary S. Robinson
David Rockwell

Jaideep Roy
Thomas J. Schaal
Akihiro Shimura
Larry Stein
Joseph Tardo
Michael G. Thompson
Forrest D. Wright
Oren Yuen

When the IEEE-SA Standards Board approved this standard on 21 September 2000, it had the following membership:

Donald N. Heirman, *Chair*

James T. Carlo, *Vice Chair*

Judith Gorman, *Secretary*

Satish K. Aggarwal
Mark D. Bowman
Gary R. Engmann
Harold E. Epstein
H. Landis Floyd
Jay Forster*
Howard M. Frazier
Ruben D. Garzon

James H. Gurney
Richard J. Holleman
Lowell G. Johnson
Robert J. Kennelly
Joseph L. Koepfinger*
Peter H. Lips
L. Bruce McClung
Daleep C. Mohla

James W. Moore
Robert F. Munzner
Ronald C. Petersen
Gerald H. Peterson
John B. Posey
Gary S. Robinson
Akio Tojo
Donald W. Zipse

*Member Emeritus

Also included is the following nonvoting IEEE-SA Standards Board liaison:

Alan Cookson, *NIST Representative*

Donald R. Volzka, *TAB Representative*

Greg Kohn

IEEE Standards Project Editor

Contents

1.	Overview.....	1
1.1	Scope.....	1
1.2	Purpose.....	1
2.	References.....	2
3.	Definitions.....	2
3.1	General terminology	2
3.2	Communication modes	4
3.3	Operating phases.....	4
4.	Features and compliance.....	7
4.1	Interface overview	7
4.2	Features	7
4.3	Device compatibility criteria.....	7
4.4	Device compliance criteria	8
4.5	Cable compliance criteria	9
5.	Interface signals	10
5.1	HostClk/nWrite (nStrobe): host driven.....	10
5.2	AD1...AD8 (Data1...Data8)	10
5.3	PtrClk/PeriphClk/Intr (nAck): peripheral driven	10
5.4	PtrBusy/PeriphAck/nWait (busy): peripheral driven.....	11
5.5	AckDataReq/nAckReverse (PError): peripheral driven	11
5.6	Xflag (Select): peripheral driven	11
5.7	HostBusy/HostAck/nDStrb (nAutoFd): host driven.....	12
5.8	Peripheral Logic High: peripheral driven	12
5.9	nReverseRequest (nInit): host driven.....	12
5.10	nDataAvail/nPeriphRequest (nFault): peripheral driven	13
5.11	1284 Active/nAStrb (nSelectIn): host driven	13
5.12	Host Logic High: host driven.....	13
6.	Interface concepts	14
6.1	Link level and data level separation.....	14
6.2	IEEE 1284 communication options	14
6.3	Nibble Mode/Byte Mode transfer	15
6.4	Host-initiated transfers.....	15
6.5	Peripheral-initiated transfers.....	15
6.6	Multiple byte transfers	15
6.7	Interface errors.....	15
6.8	Peripheral error resolution	16
6.9	ECP Mode command/data	16
6.10	EPP Mode addressing	17
6.11	Device identification.....	18

7.	Interface operation	19
7.1	Power-On	19
7.2	Initialization	19
7.3	Compatibility Mode	19
7.4	Negotiation.....	20
7.5	Peripheral-to-host transfer modes	28
7.6	Device ID	52
7.7	Termination.....	53
7.8	Collisions	53
8.	Mechanical and electrical interface	56
8.1	General considerations.....	56
8.2	Mechanical characteristics	56
8.3	Electrical characteristics	61
9.	Software support	74
9.1	General considerations.....	74
9.2	Application level compatibility.....	75
9.3	MS-DOS IEEE 1284 driver	75
9.4	Windows 1284 driver.....	75
9.5	Reverse channel data.....	75
9.6	Link performance.....	75
	Annex A (normative) Timing specifications	76
	Annex B (normative) Signal transition events.....	78
	Annex C (informative) Centronics and PC-compatible parallel interfaces	81
	Annex D (informative) Bibliography.....	98
	Annex E (informative) Reducing data loss in ECP reverse termination	99

IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers

1. Overview

1.1 Scope

This standard defines a signaling method for asynchronous, fully interlocked, bidirectional parallel communications between hosts and printers or other peripherals. A functional subset of the signaling method may be implemented on personal computers (PCs) or equivalent parallel port hardware with new software. This standard recommends new electrical interfaces, cabling, and interface hardware that provides improved performance while retaining backward compatibility with this subset.

This standard also specifies a format for a peripheral identification string and a method of returning this string to the host outside of the bidirectional data stream.

1.2 Purpose

This standard was created because there existed no defined standard for bidirectional parallel communications between personal computers and printing peripherals. Pre-existing methods utilized a wide variety of hardware and software products, each with unique and incompatible signaling schemes. This standard was developed to provide an open path for communications between computers and more intelligent printers and peripherals. The availability of a standard bidirectional protocol will encourage the development of new peripherals that return significant data, as well as basic status, to the host.

2. References

This standard shall be used in conjunction with the following publication. If the following publication is superseded by an approved revision, the revision shall apply.

IEC 60950 (1999-04), Safety of information technology equipment.¹

3. Definitions

3.1 General terminology

The following terms are used in this standard and/or may be used in conjunction with the signaling methods defined in this standard. The definitions are not intended to be absolute, but they do reflect the use of the terms in this standard.

3.1.1 bidirectional operation: When the peripheral and host communicate using both forward and reverse data channels. As defined in this standard, Nibble and Byte Modes provide reverse channel communication and are used in conjunction with Compatibility Mode to provide bidirectional operation. Extended Capabilities Port (ECP) and Enhanced Parallel Port (EPP) Modes support bidirectional communication.

3.1.2 Centronics: The popular name for the parallel printer port used as the parallel interface for most printers and supported by most “MS-DOS compatible” PCs. The name is derived from the printer manufacturer that introduced this interface, Centronics Data Computer Corporation. Despite a basic similarity, many variations of this interface have been implemented in different peripherals and hosts. This specification describes the more prevalent variations of the “Centronics” interface and defines a family of signaling methods that are backward compatible with the typical “Centronics” interface.

3.1.3 Centronics connector: The popular name for the 36-pin ribbon contact type connector commonly used for the parallel port on printers. This connector is referred to as the “IEEE 1284-B” connector in this standard.

3.1.4 command set: A field in the Device ID message identifying the type of data expected by the peripheral. For example, a printer might use this field to report which page description language(s) it supports.

3.1.5 compatible device: A device that supports any of a specified range of popular variants of the “Centronics” interface. Compatible devices will interoperate with compliant devices in Compatibility Mode only.

3.1.6 compliant device: A device that supports either the Level 1 or Level 2 electrical interface, plus Compatible and Nibble Mode operation, as well as the negotiation phases necessary to transition between these two modes.

3.1.7 device driver: A program that runs on the host and manages the sending and receiving of information from the peripheral. The driver utilizes the link level interface defined in this standard to communicate data between the application program and the peripheral personality.

3.1.8 Device ID: A structured, variable length ASCII message identifying the manufacturer, command set, and model of the peripheral. Additional information may also be included. The message is provided by the

¹IEC publications are available from the Sales Department of the International Electrotechnical Commission, Case Postale 131, 3, rue de Varembe, CH-1211, Genve 20, Switzerland/Suisse (<http://www.iec.ch/>). IEC publications are also available in the United States from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

peripheral in response to a request issued by the host during the negotiation phase. Provided that the peripheral supports the bidirectional mode requested by the host, this message is provided in the requested mode. The Device ID is intended to assist the host in selecting the device and/or peripheral driver appropriate to the peripheral.

3.1.9 forward channel: Data path from the host to the peripheral.

3.1.10 host: A device, typically a personal computer, that will control the communications with attached peripherals.

3.1.11 IEEE 1284-compatible device: *See:* **compatible device.**

3.1.12 IEEE 1284-compliant device: *See:* **compliant device.**

3.1.13 IEEE 1284-A connector: A plug or receptacle 25-pin subminiature D-shell connector, as specified in 8.2 and shown in Figure 24 of IEEE Std 1284-2000. This is the type of connector used on the MS-DOS compatible PC printer port adapter.

3.1.14 IEEE 1284-B connector: A plug or receptacle 36-pin ribbon type connector, as specified in 8.2 and shown in Figure 25 of IEEE Std 1284-2000. This type of connector is also known as a “Centronics Connector.”

3.1.15 IEEE 1284-C connector: A miniature plug or receptacle 36-pin ribbon type connector, as specified in 8.2 and shown in Figure 26 of IEEE Std 1284-2000.

3.1.16 Level 1 device: A device that supports the Level 1 electrical interface.

3.1.17 Level 2 device: A device that supports the Level 2 electrical interface.

3.1.18 link: The physical connection and electronic hardware by which data is transferred between the host and the peripheral. This standard is concerned only with the link layer; the only information transfer defined or required as part of this standard is that necessary to control and synchronize the communication of peripheral data. The defined interface is transparent to the peripheral data communicated at data or higher layers.

3.1.19 n: When preceding a capitalized signal name, denotes a signal having negative true logic (e.g., nAck).

3.1.20 PC parallel port: The parallel printer port used as the parallel interface for most printers and supported by most (personal computers) PCs. This interface has been variously defined by different PC and peripheral manufacturers. This standard describes the more prevalent variations of the interface and defines a family of signaling methods that are backward compatible with the typical PC parallel port.

3.1.21 peripheral: A device, attached to a host via a communication link.

3.1.22 peripheral personality: The characteristics of a language processor or operating environment that a peripheral runs to interpret commands and data being sent.

3.1.23 reverse channel: Data path from the peripheral to the host.

3.1.24 solicited status: Information generated by the peripheral in response to a command from the host.

3.1.25 status: A term used generally to describe data generated by the peripheral that reflects the current operating state of the peripheral.

3.1.26 status lines: Unidirectional signals from the peripheral to the host, defined in Compatibility Mode to handshake data and to report error conditions. In other IEEE 1284 interface modes defined in this standard (IEEE Std 1284-2000), these lines are used for control, data, and/or status.

3.1.27 unidirectional operation: When the peripheral and host communicate data in one direction only. Compatibility Mode is unidirectional in the forward direction; Byte and Nibble Modes are unidirectional in the reverse direction.

3.1.28 unsolicited status: Information generated by the peripheral that has not been asked for by the host, yet is important enough that the peripheral desires to send it to the host.

3.2 Communication modes

NOTE—The signaling method described herein provides for several modes of host-peripheral communication. The interface is initially in Compatibility Mode. Other modes provide additional features and/or improved performance. Compliant devices can determine and switch to the most effective mode supported by both devices. All compliant devices are required to support Compatibility Mode, Nibble Mode, and the negotiation necessary to switch between communication modes.

3.2.1 Compatibility Mode: An asynchronous, byte-wide forward (host-to-peripheral) channel with data and status lines used according to their original definitions. Compatibility Mode is backward compatible with many existing devices, including the PC parallel port, and is the base mode common to all compliant interfaces.

3.2.2 Nibble Mode: An asynchronous, reverse (peripheral-to-host) channel, under control of the host. Data bytes are transmitted as two sequential 4-bit nibbles using four peripheral-to-host status lines. Nibble Mode is used with Compatibility Mode to implement a bidirectional channel. These two modes cannot be active simultaneously.

3.2.3 Byte Mode: An asynchronous, byte-wide reverse (peripheral-to-host) channel using the eight data lines of the interface for data and the control/status lines for handshaking. Byte Mode is used with Compatibility Mode to implement a bidirectional channel, with transfer direction controlled by the host when the host and peripheral both support bidirectional use of the data lines. The two modes cannot be active simultaneously.

3.2.4 Extended Capabilities Port (ECP) Mode: An asynchronous, byte-wide, bidirectional channel. An interlocked handshake replaces the minimum timing requirements of Compatibility Mode. A control line is provided to distinguish between command and data transfers. A command may optionally be used to indicate single-byte data compression or channel address.

3.2.5 Enhanced Parallel Port (EPP) Mode: An asynchronous, byte-wide, bidirectional channel controlled by the host device. This mode provides separate address and data cycles over the eight data lines of the interface.

3.3 Operating phases

NOTE—Interface operation is divided into a number of interface phases. Each communication mode consists of one or more phases. Additional phases are defined to cover initialization and transitions between communication modes. The names and interpretations of the interface signals may vary between phases, but are consistent within each phase.

3.3.1 Compatibility Mode phases

3.3.1.1 Compatibility Mode forward data transfer phase: Begins when the host asserts nStrobe and ends following data hold time and nStrobe de-assertion. (Note that the host is not free to send the next data byte

until the peripheral acknowledges the transfer using nAck.) The host may not initiate negotiation to a new operating mode until the interface returns to Compatibility Mode forward idle phase.

3.3.1.2 Compatibility Mode forward idle phase: When the interface is in Compatibility Mode with no data transfer in progress. The host may initiate a data transfer in Compatibility Mode or may initiate negotiation to a new operating mode.

3.3.2 Nibble and Byte Mode phases

3.3.2.1 reverse data transfer phase: When data transfers from the peripheral to the host.

3.3.2.2 reverse host busy data available phase: When the peripheral has data to transmit.

3.3.2.3 reverse host busy data not available phase: When the peripheral has no more data to transmit.

3.3.2.4 reverse idle phase: When no data transfer is in progress and the host is waiting for peripheral data. When data are available, the peripheral will cause the interface to go to the reverse interrupt phase.

3.3.2.5 reverse interrupt phase: A phase that provides the mechanism for the peripheral to alert the host that it has data to transfer. While in this phase, the host may cause the interface to go to the termination phase.

3.3.3 ECP Mode phases

3.3.3.1 ECP setup phase: A phase that sets the interface signals to the correct state for the ECP forward transfer phase. It immediately follows the negotiation phase.

3.3.3.2 ECP forward transfer phase: When the host transfers data or commands to the peripheral using HostClk. The peripheral may indicate its desire to send data to the host by asserting nPeriphRequest.

3.3.3.3 ECP forward idle phase: When no data transfer is in progress. The peripheral may indicate its desire to communicate with the host by asserting nPeriphRequest. The host may cause the interface to go to the forward to reverse phase, forward transfer phase, or the termination phase.

3.3.3.4 ECP forward to reverse phase: When the interface is changing from the forward direction to the reverse direction.

3.3.3.5 ECP reverse transfer phase: When the peripheral transfers data or commands to the host. The host may interrupt the peripheral, causing the interface to go to the reverse to forward phase.

3.3.3.6 ECP reverse idle phase: When no data transfer is in progress. The host may interrupt the peripheral, causing the interface to go to the reverse to forward phase. The peripheral may cause the interface to go to the reverse transfer phase or reverse to forward phase.

3.3.3.7 ECP reverse to forward phase: When the interface is changing from the reverse direction to the forward direction.

3.3.4 EPP mode phases

3.3.4.1 EPP address read phase: When the host is performing an address read transfer operation.

3.3.4.2 EPP address write phase: When the host is performing an address write transfer operation.

3.3.4.3 EPP data read phase: When the host is performing a data read transfer operation.

3.3.4.4 EPP data write phase: When the host is performing a data write transfer operation.

3.3.4.5 EPP initial idle phase: A transition phase from negotiation phase to EPP Mode. After a transfer operation (address or data read or write), the interface will enter the idle phase.

3.3.4.6 EPP idle phase: When no address or data transfer is in progress.

3.3.4.7 EPP termination phase: A host-initiated transition phase in which the interface is changed from EPP Mode to Compatibility Mode.

3.3.5 Additional phases

3.3.5.1 initialization phase: A phase that includes both power-on initialization and host-driven interface reset.

3.3.5.2 negotiation phase: Signal handshaking to change the signaling method from Compatibility Mode to Nibble, Byte, ECP, or EPP Mode.

3.3.5.3 power-on phase: A phase that includes power-on initialization for both devices.

3.3.5.4 termination phase: A host-initiated transition phase in which the interface is changed from Nibble, Byte, or ECP Mode to Compatibility Mode.

4. Features and compliance

4.1 Interface overview

This interface is a bidirectional extension of the existing PC parallel interface (commonly known as the Centronics interface). The bidirectional communication occurs using the signals in the existing interface. The interface supports a number of distinct communication modes, and the interpretation of interface signals depends on the current mode. Compatibility Mode provides host-to-peripheral communication in a manner compatible with the traditional unidirectional interface. Nibble and Byte Modes provide peripheral-to-host communication and may be combined with Compatibility Mode to provide bidirectional operation. ECP Mode provides symmetric bidirectional communications without the overhead of changing communication modes. EPP Mode provides asymmetric bidirectional data transfer driven by the host device. Not all devices support all communication modes. A mechanism is provided for the host and peripheral to negotiate the mode to be used for data transfers and to renegotiate as needed.

In Compatibility Mode, the host sends 8 bits to the peripheral over the interface data signals D1 through D8. In Nibble Mode, peripheral-to-host data bytes are sent as two sequential nibbles on the parallel port status lines. In Byte Mode, ECP Mode, or EPP Mode, peripheral-to-host data bytes are sent on the parallel port data signals. Each of these arrangements gives the interface at least two separate channels, one in each direction. Having two channels allows peripheral-to-host data to be transferred even when the host-to-peripheral data channel is blocked.

In the Nibble or Byte Mode, the host requests a reverse channel transfer, and the peripheral responds by indicating whether there is data to be sent. If no data is ready to be sent, the host can enter the reverse idle phase, in which the peripheral will indicate to the host when data becomes available. In the ECP or EPP Modes, the host may read or write address or data information from or to the peripheral. The host may return the link to the Compatibility Mode whenever the interface is in a “valid” state, which is uniquely specified for each mode.

4.2 Features

This interface provides several capabilities to the user. These capabilities satisfy the following list of user objectives:

- a) This interface provides the capability to send data from the host computer to the peripheral at a higher speed than previously done. This is accomplished by shortening the signal timing values where possible.
- b) This interface provides a path for data to be sent from the peripheral to the host using existing and future parallel port hardware.
- c) The reverse channel capability reduces the amount of user interaction needed to operate the peripheral. For example, application programs can ask a printer for a list of fonts currently available. The application can then decide if new fonts must be downloaded, rather than ask the user.

4.3 Device compatibility criteria

Devices that meet the mechanical interface requirements of 8.2, the electrical requirements of 8.3.2, and the protocol compatibility requirements of 7.3, but neither the compliance requirements of that subclause nor the additional protocols of 7.4 et al., are referred to as “IEEE 1284-compatible” devices. *IEEE 1284-compatible devices are not IEEE 1284 compliant*, but they meet the minimum requirements for operation in Compatibility Mode with any IEEE 1284-compliant device.

The popular parallel port implementations vary widely in the timing of the nStrobe, nAck, and Busy signals. For instance, many existing hosts ignore the nAck signal entirely, relying on Busy to control the flow of data to the peripheral. The IEEE 1284 protocol compatibility requirements are sufficient to ensure operation of an IEEE 1284-compatible device with an IEEE 1284-compliant device, but not to ensure interoperation of two IEEE 1284-compatible devices. Conversely, the IEEE 1284 protocol compliance requirements ensure operation of IEEE 1284-compliant devices with both IEEE 1284-compliant and IEEE 1284-compatible devices.

Historically, the published specifications of the popular variants of the “Centronics-compatible” interface (see Annex C) specified a single “logic high” and a single “logic low” level for devices on both ends of the interface, leaving no operational margin for noise or losses in the connecting cable. In practical applications, the actual operating characteristics often exceeded these minimum requirements.

4.4 Device compliance criteria

This standard defines two levels of device compliance, as follows:

- a) An IEEE 1284-I compliant device shall have a mechanical interface using IEEE 1284-A and IEEE 1284-B connectors, have a Level 1 electrical interface, and abide with the protocol compliance criteria, all of which are listed in 4.4.3.
- b) An IEEE 1284-II compliant device shall have a mechanical interface using IEEE 1284-C connectors, have a Level 2 electrical interface, and abide with the protocol compliance criteria, all of which are listed in 4.4.3.

4.4.1 Mechanical

IEEE 1284 devices shall use IEEE 1284-A, IEEE 1284-B, or IEEE 1284-C connectors in one of the configurations depicted in Clause 8. The IEEE 1284-C connectors are recommended for all new designs.

4.4.2 Electrical

Level 1 electrical requirements provide sufficient margin to assure operation with any compatible or Level 1 compliant device. See 8.3.2 for a complete list of requirements.

Level 2 requirements ensure that all IEEE 1284 Level 2 compliant devices will interoperate with any IEEE 1284 Level 2 compliant device and in general with IEEE 1284 Level 1 compliant devices. See 8.3.3 for a complete list of requirements.

It is recommended that all new designs implement Level 2 drivers and receivers.

4.4.3 Protocol

Compliant devices shall implement Compatibility and Nibble Modes as defined within this standard, as well as the negotiation phases necessary to transition between these two modes. Implementation of Byte, ECP, and/or EPP Modes (also defined within this standard) is recommended but not required for IEEE 1284 compliance.

Peripherals may, but are not required to, implement the Device ID string as specified in 7.6. Peripherals that implement the Device ID string shall be able to return the ID string to the host in the Nibble Mode. It is recommended that peripherals be able to return the ID string to the host using any of the reverse channel modes implemented by the device with the exception of EPP.

4.5 Cable compliance criteria

Interconnection cables intended for use with IEEE 1284 devices shall meet the mechanical and electrical requirements of Clause 8. Cable assemblies that meet these requirements shall be clearly and permanently labeled “IEEE Std 1284-2000 compliant” to distinguish them from cables having the same type connectors but different electrical characteristics.

5. Interface signals

NOTE—IEEE Std 1284-2000 mode signal names are shown with their Compatibility Mode names in parentheses. Usage of mode-specific signal names imply the interface is operating in that mode.

5.1 HostClk/nWrite (nStrobe): host driven

Compatibility Mode: Set active low to transfer data into the input latch of the peripheral. Data is valid while nStrobe is low.

negotiation phase: Set active low to transfer the extensibility request value into the input latch of the peripheral. Data is valid on the leading (falling) edge of HostClk.

reverse data transfer phase: Nibble Mode: Set high during transfers to avoid latching data into the peripheral. Byte Mode: Pulsed low during transfers to acknowledge transfer of data from the peripheral. The peripheral shall ensure that this pulse does not transfer a new data byte into the input latch of the peripheral.

ECP Mode: Used in a closed-loop handshake with PeriphAck(Busy) to transfer data or address information from the host to the peripheral.

EPP Mode: Set low to denote an address or data write operation to the peripheral. Set high to denote an address or data read operation from the peripheral.

5.2 AD1...AD8 (Data1...Data8)

Driven by the host in Compatibility Mode and the negotiation phase, not used in Nibble Mode, and bidirectional in all other modes.

All modes: Data 1 is the least significant bit (bit 0).
Data 8 is the most significant bit (bit 7).

Compatibility Mode: Forward channel data.

negotiation phase: Extensibility request value.

reverse data transfer phase: Nibble Mode: Not used (host may still drive bus). Byte Mode: Reverse channel data.

ECP Mode: Host-to-peripheral or peripheral-to-host address or data.

EPP Mode: Host-to-peripheral or peripheral-to-host address or data.

5.3 PtrClk/PeriphClk/Intr (nAck): peripheral driven

Compatibility Mode: Pulsed low by the peripheral to acknowledge transfer of a data byte from the host.

negotiation phase: Set low to acknowledge IEEE 1284 support, then set high to indicate that the Xflag(Select) and data available flags may be read.

reverse data transfer phase: Nibble and Byte Modes: Used to qualify data being sent to the host.

reverse idle phase: Set low, then high by the peripheral to cause an interrupt indicating to the host that data is available.

ECP Mode: Used in a closed-loop handshake with HostAck(nAutoFd) to transfer data from the peripheral to the host.

EPP Mode: Used by the peripheral to interrupt the host. This signal is pulsed low for a minimum of 500 ns and must be raised for a minimum of 1 μ s before reasserting.

5.4 PtrBusy/PeriphAck/nWait (busy): peripheral driven

Compatibility Mode: Driven high to indicate that the peripheral is not ready to receive data.

negotiation phase: Reflects the present state of the forward channel of the peripheral.

reverse data transfer phase: Nibble Mode: Data bit 3 then 7, then forward channel busy status. Byte Mode: forward channel busy status.

reverse idle phase: Forward channel busy status.

ECP Mode: The peripheral uses this signal for flow control in the forward direction. PeriphAck also provides a ninth data bit that is used to determine whether command or data information is present on the data signals in the reverse direction.

EPP Mode: This signal should be driven inactive as a positive acknowledgment from the peripheral that transfer of data or address is completed. The signal is active low. It should be driven active as an indication that the device is ready for the next address or data transfer.

5.5 AckDataReq/nAckReverse (PError): peripheral driven

Compatibility Mode: Driven high to indicate that the peripheral has encountered an error in its paper path. The meaning of this signal varies from peripheral to peripheral. Peripherals shall set nFault low whenever PError is set high.

negotiation phase: Set high to indicate IEEE 1284 support, then follows nDataAvail(nFault).

reverse data transfer phase: Nibble Mode: Data bit 2 then 6. Byte Mode: Same as nDataAvail(nFault).

reverse idle phase: Set high until the host requests a data transfer, then follows nDataAvail(nFault).

ECP Mode: The peripheral drives this signal low to acknowledge nReverseRequest. The host relies upon nAckReverse to determine when it is permitted to drive the data signals.

EPP Mode: (User Defined 1) This signal is manufacturer specific and beyond the scope of this standard.

5.6 Xflag (Select): peripheral driven

Compatibility Mode: Set high to indicate that the peripheral is online.

negotiation phase: The name Xflag refers to extensibility flag. It is used by the peripheral to reply to the requested extensibility byte sent by the host during the negotiation phase. The signal level used to indicate an affirmative response for each respective extensibility byte is shown in Table 4.

reverse data transfer phase: Nibble Mode: Data bit 1 then 5. Byte Mode: Same as negotiation phase.

reverse idle phase: Same as negotiation phase.

ECP Mode: Same as negotiation phase.

EPP Mode: (User Defined 3) This signal is manufacturer specific and beyond the scope of this standard.

5.7 HostBusy/HostAck/nDStrb (nAutoFd): host driven

Compatibility Mode: The interpretation of this signal varies from peripheral to peripheral. Set low by host to put some printers into auto-line feed mode. May also be used as a ninth data, parity, or command/data control bit.

negotiation phase: Set low in conjunction with IEEE 1284 Active(nSelectIn) being set high to request a IEEE 1284 mode. Then set high after the peripheral sets PtrClk(nAck) low.

reverse data transfer phase: Nibble Mode: Set low to indicate that the host can receive peripheral-to-host data, then set high to acknowledge receipt of that nibble. Byte Mode: Same as Nibble Mode to request and acknowledge bytes. Following a reverse channel transfer, the interface transitions to idle phase when Host-Busy(nAutoFd) is set low and the peripheral has no data available.

reverse idle phase: Set high in response to a PtrClk(nAck) low pulse to re-enter reverse data transfer phase. If set high with IEEE 1284 Active(nSelectIn) set low, the IEEE 1284 idle phase is aborted, and the interface returns to Compatibility Mode.

ECP Mode: The host drives this signal for flow control in the reverse direction. It is used in an interlocked handshake with PeriphClk(nAck). HostAck also provides a ninth data bit that is used to determine whether command or data information is present on the data signals in the forward direction.

EPP Mode: This signal is active low. It is used to denote a data cycle.

5.8 Peripheral Logic High: peripheral driven

Set high to indicate (subject to the provisions of 8.3.5) that all other signals sourced by the peripheral are in a valid state. Set low to indicate that the peripheral power is off or that peripheral-driven interface signals are otherwise in an invalid state.

Peripheral manufacturers may, but are not required to, use this signal to provide 5 V of power to an attached device. In any case, the peripheral shall limit the short-circuit current to a maximum of 1.0 A and shall provide circuitry to ensure a valid logic low level (as defined in 8.3.5) on this signal when the peripheral power is off.

5.9 nReverseRequest (nInit): host driven

Compatibility Mode: Pulsed low in conjunction with IEEE 1284 Active low to reset the interface and force a return to Compatibility Mode idle phase.

negotiation phase: Set high.

reverse data transfer phase: Set high.

ECP Mode: This signal is driven low to place the channel in the reverse direction. While in the ECP Mode, the peripheral is only allowed to drive the bidirectional data signals when nReverseRequest is low and IEEE 1284 Active is high.

EPP Mode: This signal is active low. When driven active (low), this signal initiates a termination cycle that results in the interface returning to Compatibility Mode.

5.10 nDataAvail/nPeriphRequest (nFault): peripheral driven

Compatibility Mode: Set low by the peripheral to indicate that an error has occurred. The meaning of this signal varies from peripheral to peripheral.

negotiation phase: Set high to acknowledge IEEE 1284 compatibility. In Nibble or Byte Mode, it is then set low to indicate peripheral-to-host data is available following the host setting HostBusy(nAutoFd) high.

reverse data transfer phase: Nibble Mode: Set low to indicate that the peripheral has the data ready to send to the host. Then used to send data bit 0 then 4. Byte Mode: Used to indicate that data is available.

reverse idle phase: Used to indicate that data is available.

ECP Mode: During ECP Mode, the peripheral shall drive this pin low to request communications with the host. This signal provides a mechanism for peer-to-peer communication. This signal would be typically used to generate an interrupt to the host. This signal is valid in the forward and reverse directions.

EPP Mode: (User Defined 2) This signal is manufacturer specific and beyond the scope of this standard.

5.11 1284 Active/nAStb (nSelectIn): host driven

Compatibility Mode: Set low by host to select peripheral.

negotiation phase: Set high in conjunction with Host Busy set low to request an IEEE 1284 mode.

reverse data transfer phase: Set high to indicate that the bus direction is peripheral to host. Set low to terminate IEEE 1284 mode and to set bus direction to host to peripheral.

reverse idle phase: Same as reverse data transfer phase.

ECP Mode: Driven high by the host while in ECP Mode. Set low by the host to terminate ECP Mode and return the link to the Compatibility Mode.

EPP Mode: This signal is used to denote an address cycle. It is active low.

5.12 Host Logic High: host driven

Set high to indicate (subject to the provisions of 8.3.5) that all other signals sourced by the host are in a valid state. Set low to indicate the host power is off or host-driven interface signals are otherwise in an invalid state.

Host manufacturers may, but are not required to, use this signal to provide 5 V of power to an attached device. In any case, the host shall limit the short-circuit current to a maximum of 1.0 A and shall provide circuitry to ensure a valid logic low level (as defined in 8.3.5) on this signal when the host power is off.

6. Interface concepts

6.1 Link level and data level separation

The concepts of physical, link, and data levels originate from the open systems interconnect (OSI) model developed under the auspices of the International Organization for Standardization (ISO) and promulgated as ISO 7498:1984 [B13].² The IEEE 1284 interface separates the operation of the link between the host and peripheral from the data communicated over that link. This allows link layer functions, such as IEEE 1284 communication mode negotiation, to be localized in an interface driver that transports application data without having to interpret the data stream (see Figure 1).

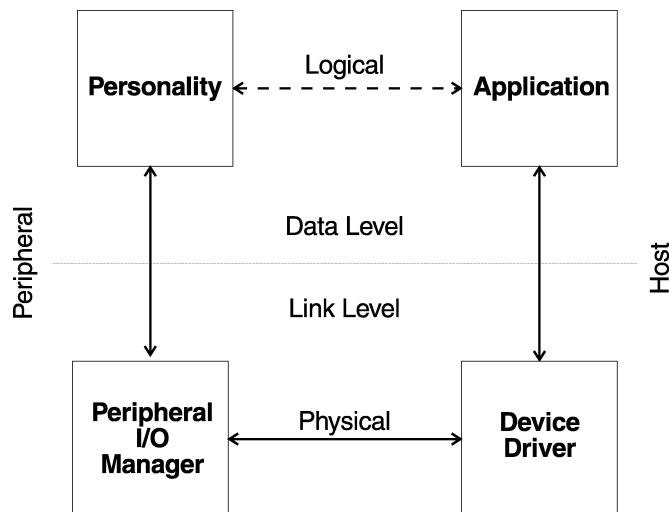


Figure 1—Separation of data level and link level

Ideally, the application and the peripheral personality should have an open dialogue that allows them to exchange information as needed. The IEEE 1284 interface provides a means by which this dialogue may be achieved. There are differences in the way the application and the personality communicate, depending on the device hardware and on the direction information is flowing. The interface drivers in the host and the peripheral deal with the details of selecting an appropriate communications mode or modes, hiding the interface dependencies from the application and peripheral personalities.

6.2 IEEE 1284 communication options

The IEEE 1284 negotiation phase allows the host and peripheral to select a mutually supported communication mode. During the negotiation phase, the host indicates which communication mode and options it would like to use via the Extensibility Request Value (see Table 4). There is one bit in the Extensibility Request Value for each IEEE 1284 mode, plus additional bits for communications options such as device identification (see 7.6) and ECP Mode Run Length Encoding (RLE).³ The peripheral indicates that it supports all requested options by setting the Extensibility Flag low (for Nibble Mode without options) or high (for any other IEEE 1284 mode or options) as shown in Table 4. If the peripheral does not support the

²The numbers in brackets correspond to those of the bibliography in Annex D.

³The host shall only request one communication mode option at a time, i.e., Byte, ECP, etc. The Device ID may be requested using Nibble, Byte, or ECP Modes. RLE may be requested when using the ECP Mode.

requested mode or options, it sets the Extensibility Flag low,⁴ and the interface returns to Compatibility Mode.

6.3 Nibble Mode/Byte Mode transfer

Many personal computers do not have the hardware necessary to receive data over the eight data lines. Nibble Mode allows peripheral-to-host data to be transferred over the parallel port status lines 4 bits at a time. Byte Mode allows the peripheral to send data to the host on the eight data lines. The host indicates its Nibble Mode/Byte Mode capability during the negotiation phase. It is required that all IEEE 1284 devices support Nibble Mode to guarantee that any two IEEE 1284 devices can communicate.

6.4 Host-initiated transfers

The host initiates all transfers in the Compatibility, Nibble, Byte, and EPP Modes and host-to-peripheral transfers in the ECP Mode except as noted in 6.5. Arbitration between ECP forward and reverse phases is controlled by the host.

6.5 Peripheral-initiated transfers

The peripheral initiates peripheral-to-host data transfers in the Nibble and Byte Modes by pulsing the PtrClk(nAck) line. The peripheral is only permitted to initiate a reverse channel transfer during the reverse idle phase in order to prevent non-IEEE 1284 hosts from interpreting the PtrClk(nAck) pulse as a request for more data from the host.

The peripheral requests use of the data bus in ECP Mode by asserting the nPeriphRequest (nFault) signal. After the host grants the request, the peripheral initiates data transfers in ECP Mode reverse phase.

The host initiates all transfers in EPP Mode.

6.6 Multiple byte transfers

The IEEE 1284 interface allows multiple bytes to be transferred from the peripheral to the host without having to renegotiate to an IEEE 1284 mode. For the Nibble and Byte Modes, the nDataAvail(nFault) signal indicates whether there is a subsequent byte ready to transfer. In the ECP Mode, the peripheral may transfer as many bytes as needed to the host, provided that the host does not issue a request to return the channel to the forward direction.

6.7 Interface errors

Errors can result from timeouts, invalid states, and invalid transitions between states. These errors can be detected by either the host computer or the peripheral. When either the host or peripheral detects an error, it shall terminate the current transfer.

When the host or peripheral detects an error, it should immediately abort (with no termination phase) and resume Compatibility Mode operation.

In particular, to protect against bus fight conditions on the bidirectional data signals, the peripheral shall stop driving the data bus within 1 μ s in the event of a protocol exception. The peripheral should carefully moni-

⁴This sequence of operations was chosen so that compliant peripherals (which support Nibble Mode, but no other IEEE 1284 modes or options) can set the Extensibility Flag low without having to examine the Extensibility Request Value.

tor both IEEE 1284 Active (nSelectIn) and nReverseRequest (nInit) to detect when the host has aborted to Compatibility Mode.

NOTE—Immediate termination upon error conditions is not supported in EPP Mode.

6.8 Peripheral error resolution

Some peripherals are subject to errors or exception conditions that require operator intervention before they can accept any more data from the host (for example, a printer might run out of paper). Peripherals may also be subject to internal processing delays of indefinite length. The IEEE 1284 interface allows the host to request an IEEE 1284 mode reverse transfer in order to resolve whatever problem exists on the peripheral. If an error message has been generated by the peripheral, the host may retrieve it and report the condition to the application and/or user.

6.9 ECP Mode command/data

ECP Mode supports two advanced features to improve the effectiveness of the protocol for some applications. The features are implemented by allowing the transfer of 8 bit data or 8 bit commands.

When in the forward direction, data is transferred when HostAck is high, and an 8 bit command is transferred when HostAck is low. The most significant bit of the command indicates whether it is a run-length count (for compression) or a channel address.

Table 1—Forward channel commands (when HostAck is low)

Bit 7	Bits 6..0
0	Run-length count (0–127) (mode 0011 0X00 only)
1	Channel address (0–127)

When in the reverse direction, data is transferred when PeriphAck is high, and an 8 bit command is transferred when PeriphAck is low.

Table 2—Reverse channel commands (when PeriphAck is low)

Bit 7	Bits 6..0
0	Run-length count (0–127) (mode 0011 0X00 only)
1	Channel address (0–127)

Devices indicate that they support decompression via the negotiation sequence. Devices that negotiate into ECP RLE mode (data 0011 0X00) shall support decompression of RLE data and may optionally compress it. Devices that negotiate into the non-RLE ECP Mode *shall not* transfer ECP Mode RLE compressed data.

6.9.1 ECP Mode data compression

Single-byte run-length-encoding is supported, which compresses strings of identical bytes. The compression is particularly useful on raster imaging devices. This simple compression does not preclude the use of other data compression schemes at a higher (data stream or packet) level.

When a run-length count is received, the subsequent data byte is replicated the specified number of times. A run-length count of zero specifies that only 1 byte of data is represented by the next data byte, whereas a run-length count of 127 indicates that the next byte should be expanded to 128 bytes. To prevent data expansion, however, run-length counts of zero should be avoided.

6.9.2 ECP Mode channel addressing

A channel addressing scheme is used in ECP Mode, providing 128 forward and reverse channel addresses. The channel addresses may be dynamically changed while in ECP Mode. Channel address definitions are device specific. The host and peripheral channel addresses default to zero after each negotiation from Compatibility Mode into ECP Mode. After a channel address command is issued from the host, that address becomes the current channel address for both host-to-peripheral and peripheral-to-host communications until another channel address command is issued or until termination.

The channel addressing capability is based on a master-slave relationship with the host being the master. The host can either use the current channel address or choose a new current channel address. In either case, the host sends data intended for the current forward channel address.

If the peripheral has requested attention and wishes to return data on the current channel address (current means the same as the last host-selected channel address), then the peripheral may send the data intended for the current channel address without first selecting a reverse channel address.

However, if the peripheral has requested attention and wishes to return data on an address different from the current channel address, then the peripheral shall first select the appropriate reverse channel address before sending the data. If the reverse channel address has been changed and the peripheral wishes to return data intended for the current channel address (the last host-selected address), then the peripheral shall first select the current channel address prior to sending data.

A channel address command issued from the peripheral only affects the peripheral-to-host communication channel. (The address used for host-to-peripheral communication remains unchanged.) That address remains in effect indefinitely for peripheral-to-host communications until another channel address command is issued from either the host or the peripheral or until termination.

6.10 EPP Mode addressing

EPP Mode incorporates a signaling technique that is microprocessor bus oriented. In the case of the typical bus system that has a time-multiplexed address/data bus, all cycles are initiated by the host processor. First, an address is generated on the bus and is latched by external circuitry when the processor generates an address strobe. A separate strobe is generated to perform the actual data transfer. Cycles are terminated when the addressed device signals “ready” back to the processor.

EPP Mode provides an interface that is functionally equivalent to such a system. All cycles are initiated by the host with a means of generating address and data strobes. Also analogous to a microprocessor bus, cycles are terminated and cycles can be extended by the device.

6.11 Device identification

Prior to the first peripheral-to-host transfer, the host does not know the type of device to which it is attached, or how to communicate with it. The device identification option allows the host to request ID information from the peripheral using one of the IEEE 1284 reverse data transfer modes (Nibble, Byte, or ECP). The peripheral identifies itself by sending a sequence of bytes to the host indicating its device type, device family, and language capabilities. Refer to 7.6 for details of the Device ID response format.

Nibble Mode: All IEEE 1284-compliant devices are required to support Nibble Mode, and all IEEE 1284-compliant devices that support Device ID are required to support Device ID in Nibble Mode. This ensures that host devices will be able to read the Device ID of the peripheral, if present, without extended IEEE 1284 mode negotiation.

ECP Mode: Channel addressing is not used during Device ID. RLE data compression may be used during Device ID, if requested by the host Extensibility Request Value and acknowledged by the peripheral Extensibility Flag. Forward channel data is not sent during Device ID mode. To transfer data, the host shall first terminate from the Device ID mode and renegotiate with the Device ID Request bit de-asserted in the Extensibility Request Value.

7. Interface operation

The interface is always initialized to the Compatibility Mode, a conventional, unidirectional host-to-peripheral interface. From the Compatibility Mode the host may

- a) Transmit data to the peripheral using the Compatibility Mode, or
- b) Direct the interface into another mode.

Refer to Annex A for signal timing definitions and to Annex B for signal event definitions.

A peripheral-to-host data transfer is initiated by the host negotiating with the peripheral for a mutually supported communication mode. At the discretion of the host, the interface can be returned to the Compatibility Mode at any time. Phase transition diagrams showing the interface as it moves between phases are shown in Figure 6, Figure 12, and Figure 16 for the Nibble and Byte, ECP, and EPP Modes, respectively.

The remainder of this clause describes the negotiation, currently defined peripheral-to-host data transfer protocols, and the return to Compatibility Mode procedures.

7.1 Power-On

Hosts and peripherals indicate their readiness to communicate by asserting Host Logic High and Peripheral Logic High respectively. All hosts with IEEE 1284-C connectors shall provide Host Logic High. All peripherals with IEEE 1284-C connectors shall provide Peripheral Logic High. The interface becomes active 500 ms after both Host Logic High and Peripheral Logic High exceed 3.0 V, at which point all control, data, and status signals are required to be valid (see 8.3.5). Since devices with IEEE 1284-A or IEEE 1284-B connectors may or may not support Host Logic High or Peripheral Logic High, there is no reliable way to initiate a transfer. If the other device is not present when a transfer begins, a timeout may occur or the interface will hang.

7.2 Initialization

Hosts may reinitialize the interface at any time by asserting $nInit$ low in conjunction with $nSelectIn$ low. The interface returns to the Compatibility Mode idle phase after initialization. Resetting the interface may also reset the peripheral.

7.3 Compatibility Mode

The Compatibility Mode follows pre-existing industry conventions with eight parallel bits, (Data 1–Data 8) qualified by a low-going pulse on $HostClk(nStrobe)$. The peripheral responds by setting $PtrBusy(Busy)$ high to hold off the next byte until the current one has been removed from the input latch. When ready for the next byte, the peripheral pulses $PtrClk(nAck)$ low, then sets $PtrBusy(Busy)$ low. Timing requirements on peripheral Busy ensure that compliant peripherals will interoperate with existing hosts that monitor Busy but not $nAck$.

Compatibility Mode timing values are specified in Figure 2 and Table 3.

After receiving Busy low from the peripheral, hosts shall not set $nStrobe$ low until T_{ready} . The peripheral can reactivate Busy asynchronously due to some condition other than the reception of a data byte. Due to this race condition, the host may at that instant be asserting $nStrobe$ low to send the next data byte. The peripheral shall accept at least one byte from the host after entering a Busy state in this situation.

Hosts shall provide T_{setup} from $D1...D8$ and $nAutoFd$ stable to the falling edge of $nStrobe$.

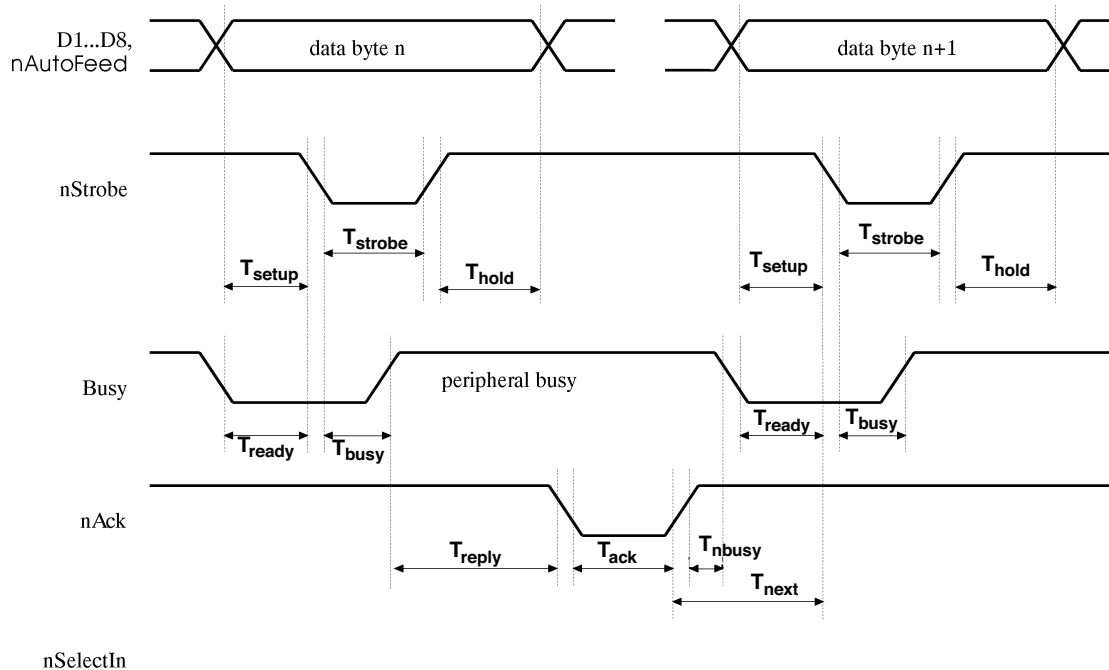


Figure 2—Compatibility Mode handshake

Hosts shall pulse $nStrobe$ low for T_{strobe} .

Hosts shall provide T_{hold} from the rising edge of $nStrobe$ to any change in $D1...D8$ or $nAutoFd$.

Compliant peripherals shall set $Busy$ high before setting either $nFault$ low, $PError$ high, or $Select$ low. $Busy$ has to be held high until all these signals return to their normal state ($nFault$ high, $PError$ low, and $Select$ high).

Normally, every byte sent from the host to the peripheral is acknowledged by the peripheral pulsing $nAck$ low while the $Busy$ line is held high. This may not be the case when a host negotiation causes the interface to change from Compatibility Mode to another IEEE 1284 interface mode after the host asserts $nStrobe$ low but before the peripheral pulses $nAck$. In this event, when the interface is returned to Compatibility Mode the host shall rely on the $Busy$ signal to determine when the peripheral is ready for the next byte.

7.4 Negotiation

The IEEE 1284-compliant host negotiates with the peripheral to determine whether or not the peripheral is IEEE 1284 compliant. (Annex B defines the signal transition events noted in the following subclauses.) The host verifies that the device is IEEE 1284 compliant when, at event 2 (see Figure 3), the peripheral, as a result of the host stimulus (event 1), sets its status lines to include $PError$ and $nFault$ both at logic high levels. This will not happen on a non-IEEE 1284 compliant peripheral. If the peripheral does not indicate that it is an IEEE 1284-compliant device (event 2 never happens), the host should remove the stimulus (event 1) and conclude that the device is not IEEE 1284 compliant.

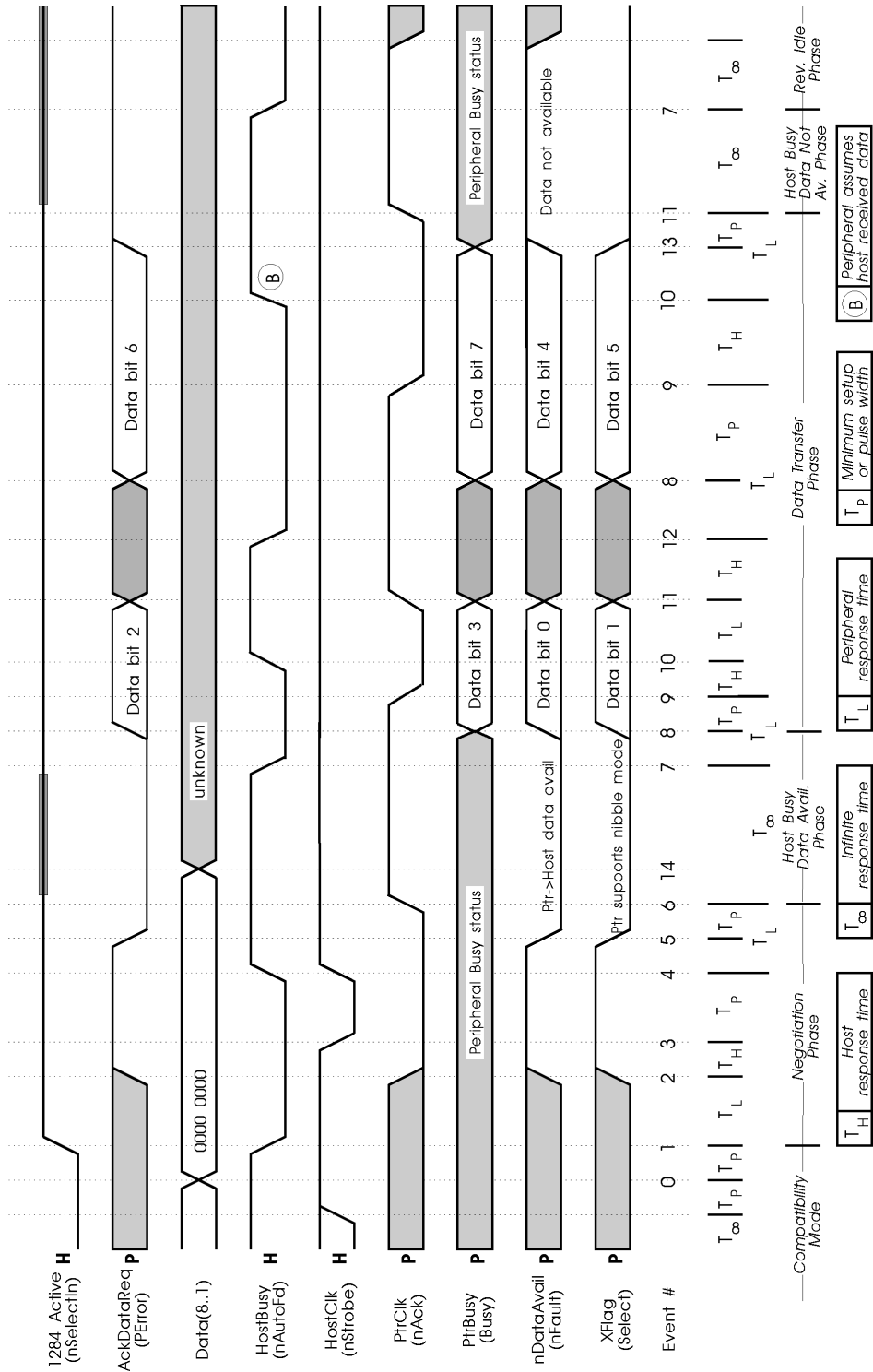


Figure 3—Nibble Mode negotiation and transfer

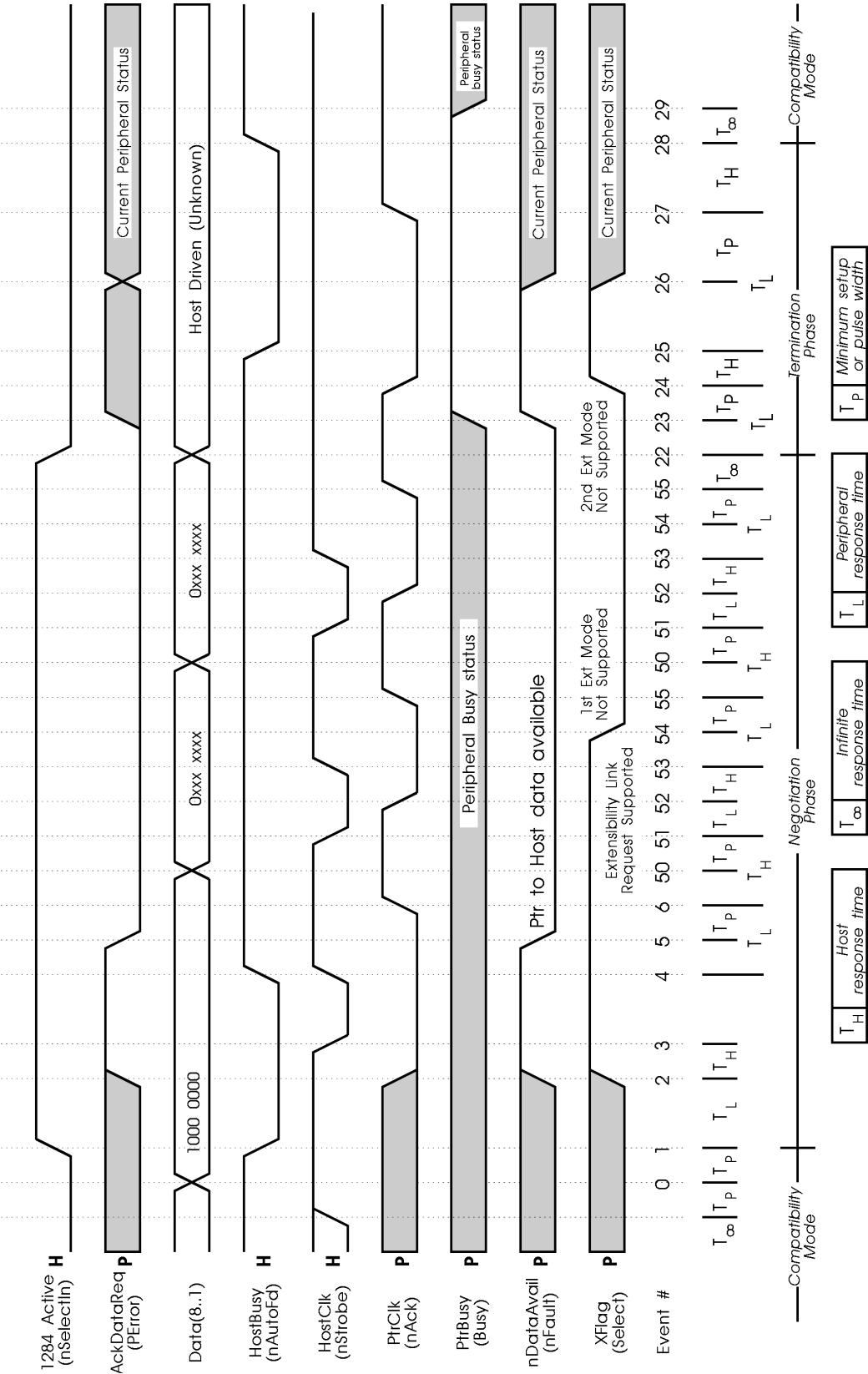


Figure 4—Negotiation with extensibility link

Table 3—Compatibility Mode handshake timing values

Parameter	Measured at	Measured from	Measured to	Value (minimum/maximum)	Compliance
T_{ready}	Host output	$\text{Busy} \leq V_{\text{IL}}$	$\text{nStrobe} \leq V_{\text{OH}}$	0 min.	Compatible hosts
$T_{\text{setup (host)}}$	Host output	Data stable	$\text{nStrobe} \leq V_{\text{OH}}$	750 ns min.	Compatible hosts
$T_{\text{setup (peripheral)}}$	Peripheral input	Data stable	$\text{nStrobe} \leq V_{\text{IH}}$	500 ns max. ^a	Compatible peripherals
$T_{\text{strobe (host)}}$	Host output	$\text{nStrobe} \leq V_{\text{OL}}$	$\text{nStrobe} \geq V_{\text{OL}}$	750 ns min., 500 ns max.	Compatible hosts
$T_{\text{strobe (peripheral)}}$	Peripheral input	$\text{nStrobe} \leq V_{\text{IL}}$	$\text{nStrobe} \geq V_{\text{IL}}$	500 ns max.	Compatible peripherals
$T_{\text{hold (host)}}$	Host output	$\text{nStrobe} \geq V_{\text{OH}}$	Data or nAutoFd change	750 ns min.	Compatible hosts
$T_{\text{hold (peripheral)}}$	Peripheral input	$\text{nStrobe} \geq V_{\text{IL}}$	Data or nAutoFd change	500 ns max.	Compatible peripherals
T_{busy}	Peripheral output	$\text{nStrobe} \leq V_{\text{IL}}$	$\text{Busy} \geq V_{\text{OH}}$	500 ns max.	Compliant peripherals
T_{reply}	Peripheral output	$\text{Busy} \geq V_{\text{IH}}$	$\text{nAck} \leq V_{\text{OH}}$	0 min.	Compatible peripherals
T_{ack}	Peripheral output	$\text{nAck} \leq V_{\text{OL}}$	$\text{nAck} \geq V_{\text{OL}}$	500 ns min., 10 μs max.	Compatible peripherals
T_{nbusy}	Peripheral output	$\text{nAck} \geq V_{\text{OH}}$	$\text{Busy} \leq V_{\text{OH}}$	0 min. ^b	Compliant peripherals
T_{next}	Host output	$\text{nAck} \geq V_{\text{IL}}$	$\text{nStrobe} \leq V_{\text{OH}}$	0 min.	Compliant hosts
NOTES 1—For more information on the history of Centronics Standard Parallel and PC-Compatible Parallel Interfaces, see Annex C and, in particular, C.6.2 for Busy-to-Ack timing variations. 2— V_{IL} is the low-level voltage input V_{OL} is the low-level voltage output V_{OH} is the high-level voltage output V_{IH} is the high-level voltage input					

^aThe maximum value stated for peripherals in this table are referenced to the peripheral. For example, the peripheral cannot require more than 500 ns data setup time.

^bRecognize that complimentary signal changes may have overlapping signal transitions. The zero minimum value cannot be guaranteed.

Upon verification that the peripheral is IEEE 1284 compliant, the host will request a communication mode for the peripheral. The IEEE 1284-compliant device will acknowledge the communication mode request based on its capabilities and execute or reject the request as appropriate.

The IEEE 1284 interface protocol has been designed to allow the host to request that the peripheral use various communication modes. This is accomplished by the host placing an extensibility request value on the data bus during the negotiation phase (event 0) (see Figure 3). The value 00h requests peripheral-to-host data

exchange using Nibble Mode reverse channel transfers. The Extensibility Request Value is used to request a communication mode. Table 4 shows the current assignments of these values.

Table 4—IEEE 1284 Extensibility Request Values—Bit assignments

Definition	Extensibility values	Xflag affirmative response
Request Extensibility Link	1000 0000	High
Request EPP Mode	0100 0000	High
Request ECP Mode with RLE	0011 0000	High
Request ECP Mode	0001 0000	High
Reserved	0000 1000	High
Request Device ID; return data using Nibble Mode Reverse Channel Transfer	0000 0100	High
Byte Mode Reverse Channel Transfer	0000 0101	High
ECP Mode Transfer without RLE	0001 0100	High
ECP Mode Transfer with RLE	0011 0100	High
Reserved	0000 0010	High
Byte Mode Reverse Channel Transfer	0000 0001	High
Nibble Mode Reverse Channel Transfer	0000 0000	Low

NOTE—IEEE Std 1284.3-2000 [B8] defines other extensibility values. Other future specifications may also define new values.

Bit seven, the “extensibility link request” bit, is a special bit. It is used to add a second extensibility request byte during the negotiation phase. This allows more extensions to be added to the interface in the future. A multiple byte negotiation is shown in Figure 4.

In Table 4, the column labeled “XFlag affirmative response” denotes the level of the XFlag signal that the host expects to see, as an indication of a positive response, when it provides the specified Extensibility Request byte to the peripheral during the negotiation phase. All requests are honored by a positive response with the exception of the Nibble Mode reverse channel transfer request. This allows a “dumb peripheral” to support this specification using only the lowest peripheral-to-host transfer protocol and without having to examine the data signals on every host negotiation. For example, if the “dumb peripheral” always provides a low-level response to every host negotiation request, then when the host requests to use a Nibble Mode protocol, it will receive a positive response from the peripheral. The peripheral has already acknowledged that it is IEEE 1284 compliant by its responses to the beginning sequence of the negotiation phase (event 2).

The default IEEE 1284 communication mode for all hosts and peripherals is Compatibility Mode. This mode is maintained until the host has successfully verified that it is connected to an IEEE 1284-compliant device. To begin the negotiation phase, the host places the Extensibility Request Value on the data bus (event 0), then sets IEEE 1284 Active(nSelectIn) high and HostBusy(nAutoFd) low (event 1). (Figure 3 shows the

negotiation events along with a reverse channel data transfer for the Nibble Mode. These negotiation events through event 4 are the same regardless of the communication mode being requested.) The peripheral responds by setting $\text{PtrClk}(\text{nAck})$ low, $\text{nDataAvail}(\text{nFault})$ high, $\text{Xflag}(\text{Select})$ high, and $\text{AckDataReq}(\text{PError})$ high (event 2). The host then sets $\text{HostClk}(\text{nStrobe})$ low (event 3), strobing the Extensibility Value into the input data latch of the peripheral. The host then sets $\text{HostClk}(\text{nStrobe})$ and $\text{HostBusy}(\text{nAutoFd})$ high (event 4), acknowledging that it has recognized an IEEE 1284-compatible peripheral. The peripheral then sets $\text{Xflag}(\text{Select})$ to its appropriate value corresponding to the extensibility feature requested (event 5). It also sets $\text{AckDataReq}(\text{PError})$ and $\text{nDataAvail}(\text{nFault})$ low if peripheral-to-host data is available. Otherwise they are set high. The peripheral then sets $\text{Ptr-Clk}(\text{nAck})$ high (event 6), indicating that the other status lines may be read. If peripheral-to-host data is available, the interface enters the host busy, data available phase. If not, the interface enters the host busy, data not available phase. At event 5 and beyond, the state of the interface is based on the mode selected.

Table 5 defines the timing values at the driver (the receiver has to make allowances for any cable effects) associated with Figure 3 through Figure 23.

Table 5—IEEE 1284 mode handshake timing values

Time	Minimum	Maximum	Description
T_H	0	1.0 s	Host response time
T_∞	0	Infinite	Infinite response time
T_L	0	35 ms	Peripheral response time
T_R	0	N/A	Peripheral response time (ECP Mode only)
T_S	35 ms	N/A	Host recovery time (ECP Mode only)
T_P	0.5 μs	N/A	Minimum setup or pulse width
T_D	0		Minimum data setup time (ECP/EPP Modes only)
T_{ES}	0	125 ns	Short response time (EPP Mode only)
T_{EL}	0	10 μs	Long response time (EPP Mode only)
T_{ER}	50 μs	Infinite	Termination/nInit pulse width
T_n	100 ns	N/A	Peripheral to host data setup time
NOTE— T_R state can be ended by the host when its T_S has expired and a recovery process is started. T_S is a minimum of 35 ms and a maximum of infinity.			

These subclauses are accompanied by full-page diagrams showing the handshake sequences. Along the bottom of each diagram are numbers corresponding to signal transition events. Each of these events is listed in Annex A for reference. The event numbers are also shown in parentheses in the following subclauses to improve readability of the diagrams.

Note that these diagrams are examples of correct operation (“typical” or “nominal”) and are not necessarily requirements.

Valid termination states (see 7.7) are indicated in the diagrams by IEEE 1284 Active(nSelectIn) shown as a heavy line.

7.4.1 Successful negotiation

There are several possible outcomes of the negotiation sequence, depending on the mode requested, the response of the peripheral, and subsequent host action.

If Nibble or Byte Mode was requested and

- a) The peripheral responds correctly and peripheral-to-host data is available, the host may
 - 1) Proceed with a reverse channel data transfer using the mode requested by the host,
 - 2) Remain in the host busy, data available phase, or
 - 3) Terminate and return to Compatibility Mode.
- b) The peripheral responds correctly, but no peripheral-to-host data is available, the host may
 - 1) Set HostBusy(nAutoFd) low, which puts the interface into the idle phase, or
 - 2) Terminate and return to Compatibility Mode.

If ECP Mode was requested and the peripheral responds correctly, the link, following a setup phase, will enter the ECP forward idle phase. From this state

- The host can request to send data to the peripheral,
- The peripheral can request to send data to the host, or
- The host can terminate the ECP Mode and return to Compatibility Mode.

If EPP Mode was requested and the peripheral responds correctly, the link will enter the EPP Mode idle phase. From this phase

- The host can write data to the peripheral.
- The host can read data from the peripheral.
- The host can write an address value to the peripheral.
- The host can read an address value from the peripheral.
- The host can terminate EPP Mode and return to Compatibility Mode.

NOTE—After successfully negotiating to the EPP Mode, low-to-high transitions of Intr(nAck) shall not be treated by the host interface as an EPP interrupt request until after the first EPP cycle. In other words, Intr(nAck) is not recognized as an EPP interrupt until after the first EPP cycle is complete.

When requesting the peripheral to provide Device ID information, the extensibility byte includes a mode to be used for the peripheral-to-host transfer. Assuming that the peripheral responds correctly, the peripheral will execute the request and transfer the data. After the host has received the data for the Device ID (the number of bytes is defined by the first two bytes received from the peripheral), the host is required to return the link to the Compatibility Mode.

7.4.2 Unsuccessful negotiation

For a transfer mode request, if the peripheral does not support the requested mode it will set Xflag(Select) low if Byte, ECP, or EPP Mode was requested (event 5). [Responding with Xflag(Select) high for a request for Nibble Mode is not allowed since all IEEE 1284-compliant devices shall support Nibble Mode.] When this occurs, the host shall return the link to the Compatibility Mode and renegotiate for another transfer mode. This sequence is shown in Figure 5.