

IEEE Standard for Interface and Protocol Extensions to IEEE 1284-Compliant Peripherals and Host Adapters

Sponsor

Microprocessor and Microcomputer Standards Committee
of the
IEEE Computer Society

Approved 21 September 2000

IEEE-SA Standards Board

Abstract: System extensions consistent with the implementation and functionality of IEEE Std 1284-2000 are covered. Multiport expansion architectures, daisy chains, an application and device driver programming interface architecture, and data link layer services are explored.

Keywords: daisy chain, data link layer services, device driver programming interface architecture, IEEE 1284, multiport expansion architecture, port sharing

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

Copyright © 2001 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 12 February 2001. Printed in the United States of America.

Print: ISBN 0-7381-2617-9 SH94881
PDF: ISBN 0-7381-1618-7 SS94881

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

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

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

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.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

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

Comments for revision of 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. 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 part of IEEE Std 1284.3-2000, IEEE Standard for Interface and Protocol Extensions to IEEE 1284-Compliant Peripherals and Host Adapters.)

This standard was formally started as an IEEE effort in March 1994. Members of the IEEE P1284.3 Working Group would like to thank the Enhanced Parallel Port Committee for their efforts in establishing the foundation for this standard and allowing this committee to include their work as the basis for much of this standard.

At the time this standard was completed, key contributors to the IEEE P1284.3 Working Group were as follows:

Larry A. Stein, *Chair*
Bill Stanley, *Vice-Chair*
Robert Gross, *Secretary*
R. Randall McBride, *Editor*
Grant Deardon, *Editor*

Balaji Baktha
Brian Batchelder
Motti Beck
Eric Byer
Dale Cronau
Lee Farrell
John Fobel
Bill Hawkins

Kenneth Hilliard
Reed Hinkel
Monte Johnson
Dave Jolley
Ken Konechy
Peter Leunig
Mark Myran
Darryn McDade
Mike Moldovan

Jon Newman
Ron Norton
Rick Pennington
Ron Proesel
Walt Scheiderich
Randy Turner
John Vitek
Forrest D. Wright

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

Malcolm J. Airst
Scott Akers
Harry A. Andreas
Keith D. Anthony
Bill Baker
Charles Brill
Vivian Cancio
Robert S. Crowder
Dante Del Corso
Stephen L. Diamond

Julio Gonzalez-Sanz
Lawrence Lamers
Joseph R. Marshall
R. Randall McBride
Robert McComiskie
Edward McCreight
Mike Moldovan
Roman Orzol
Granville Ott
Maia Pindar
Jeff Rackowitz

Thomas J. Schaal
Jim Soriano
Robert K. Southard
Larry A. Stein
Robert G. Stewart
Paul Walker
James Wolffe
Forrest D. Wright
Oren Yuen
Janusz Zalewski

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

The following information is given for the convenience of users of this standard and does not constitute an endorsement by the IEEE of these products.

Centronics is a registered trademark of the Genicom Corporation.

Intel is a registered trademark of Intel Corporation.

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	Acronyms.....	4
3.3	Communication modes	4
4.	Features and compliance.....	4
4.1	Overview	4
4.2	Multiport (MP).....	5
4.3	Service provider interface (SPI) features	6
4.4	Data link layer features	6
4.5	Physical interface	7
4.6	Compliance	7
5.	Port-sharing protocols	8
5.1	Overview	8
5.2	Command packet protocol (CPP)	8
5.3	DC architecture	14
5.4	Multiplexor architecture	29
5.5	Global reset.....	44
6.	Service provider interface (SPI)	44
6.1	Overview	44
6.2	SPI Client interface definition	45
6.3	SPI hardware interface definition	51
7.	Data link layer.....	53
7.1	Overview	53
7.2	Data link SPI.....	53
7.3	Wire protocol	55
8.	Additional IEEE 1284 modes	57
8.1	Bounded extended capabilities port (BECPP) mode	57
8.2	Channelized nibble mode.....	57
Annex A (informative)	Enhanced parallel port (EPP) BIOS	59
Annex B (informative)	SPI usage examples	64
Annex C (informative)	Bibliography	68
Annex D (normative)	Signal transition events.....	69

IEEE Standard Interface and Protocol Extensions to IEEE 1284-Compliant Peripherals and Host Adapters

1. Overview

1.1 Scope

IEEE Std 1284.3-2000 defines system extensions consistent with the implementation and functionality of IEEE Std 1284-2000¹.

These functions include the following:

- Multiport expansion architectures
 - Multiplexor
 - Daisy chain
- Application and device driver programming interface architecture that can be supported across various operating systems.
- Data link layer services for supporting IEEE 1284 parallel ports.

1.2 Purpose

IEEE Std 1284-2000 specifies all the required mechanical, electrical, and timing characteristics for a high-speed, bidirectional parallel port for printers and other peripherals. While these parameters define the physical environment and protocol requirements for an operational IEEE 1284 port, there is a need to standardize a number of extensions and operating system interfaces to help facilitate operation across various system platforms and operating environments.

¹Information on references can be found in Clause 2.

2. References

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

IEEE Std 1284-2000, IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers.²

3. Definitions

3.1 General terminology

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

3.1.1 bidirectional operation: The peripheral and host communicate using forward and reverse data channels. As defined in this document, 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 Basic Input Output System (BIOS): A set of functions and services that drivers may use to implement system functions independent of the platform implementation.

NOTE—See Annex A.

3.1.3 Centronics®: The popular name for the parallel printer port used as the parallel interface for most printers and supported by most personal computers. The name is derived from Centronics Data Computer Corporation, the printer manufacturer that introduced this interface. This interface has never been published as a standard. Despite a basic similarity, many variations have been implemented in different peripherals and hosts.

3.1.4 control lines: Unidirectional signal lines from host to peripheral(s) used to implement handshaking and protocol. These are the nStrobe, nAutoFeed, nInit, and nSelectIn lines.

3.1.5 daisy chain (DC): An interconnection signaling protocol defined in this standard by which multiple devices are attached to a single host or multiplexor port in “chained” fashion (i.e., one input, one output). In this architecture, peripheral 1 is attached to the host, or Mux port. Peripheral 2 is attached to peripheral 1, peripheral 3 is attached to peripheral 2, etc. Up to four DC-capable devices may be chained together, with one additional non-DC-capable device added to the end.

3.1.6 data lines: Bidirectional signal lines that carry binary data during forward and reverse channel data transfer cycles. These are lines defined by IEEE Std 1284-2000 as AD1 through AD8 for forward channel data transfer and 8-bit reverse channel data transfer. In Nibble Mode, reverse data is transferred using the status lines.

3.1.7 data link layer: A group of functions that provide data transfer services to a high-layer client.

NOTE—See 4.4 and Clause 7.

²IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA (<http://standards.ieee.org/>).

3.1.8 Dongles: External device with pass-through connections commonly used as software security keys.

3.1.9 forward direction: Data communication whose direction is from host to peripheral.

3.1.10 host: The device that controls communications to the attached peripheral devices.

3.1.11 IEEE 1284 compatible device: A device that supports any of the common variants of the Centronics interface. Compatible devices shall interoperate with compliant devices in Compatibility Mode.

3.1.12 IEEE 1284-compliant device: A device that supports either IEEE 1284 Level 1 or Level 2 electrical interface, plus Compatibility and Nibble Mode operation, as well as the negotiation phases necessary to transition between the two modes.

NOTE—See Clause 4 of IEEE Std 1284-2000.

3.1.13 IEEE 1284 Type A connector: A plug or receptacle 25-pin sub-miniature D-shell connector as specified in IEEE Std 1284-2000. This is the type of connector used on most personal computer (PC) printer port adapters. It is commonly referred to as the DB25 connector.

3.1.14 IEEE 1284 Type B connector: A plug or receptacle 36-pin ribbon type connector as specified in IEEE Std 1284-2000. This type of connector is also known as a Centronics connector.

3.1.15 IEEE 1284 Type C connector: A miniature plug or receptacle 36-pin ribbon type connector as specified in IEEE Std 1284-2000.

3.1.16 multiport device (MP device or MPD): A device designed to operate in a multiplexor or daisy chain environment, as defined in IEEE Std 1284.3-2000.

3.1.17 multiplexor: A device that attaches to the host parallel port and provides two to four device ports that are logically connected to the host port.

NOTE—See 4.2.1.

3.1.18 n: When preceding a capitalized signal name, is used to denote a signal having negative true logic.

3.1.19 non-multiport device (non-MP device): A device that does not meet the compliance criteria of IEEE Std 1284.3-2000.

NOTE—See 4.6.1 or 4.6.2.

3.1.20 parallel port (IEEE 1284 parallel port): The physical interface as defined by IEEE Std 1284-2000.

3.1.21 peripheral: A device controlled by a host.

3.1.22 plug and play (PnP): A hardware and software framework to allow for autodetection and configuration of peripheral devices and add-in cards attached to a host.

3.1.23 reverse direction: Data communication whose direction is from peripheral to host.

3.1.24 service provider interface (SPI): The operating system interface to the IEEE 1284.3 physical layer.

NOTE—See 4.3 and Clause 6.

3.1.25 status lines: Unidirectional signals lines from peripheral(s) to host used to implement handshaking and protocol. These are the Busy, nAck, Select, PError, and nFault lines.

3.2 Acronyms

DC	Daisy chain (see 3.1.13)
MP	Multiport (see 3.1.14)
Mux	Multiplexor (see 3.1.15)
EPP	Enhanced parallel port (see 3.3)
ECP	Extended capabilities port (see 3.3)

3.3 Communication modes

The Communication Modes as defined in IEEE Std 1284-2000 provide different levels of communication capabilities and allow a peripheral device to implement the most appropriate method. A mechanism is provided to synchronize the host and peripheral modes.

3.3.1 Byte Mode: An asynchronous, byte-wide reverse channel in which data is transmitted one byte at a time using the data lines.

3.3.2 Compatibility Mode: An asynchronous, byte-wide forward channel with data and status lines as used in the Centronics interface. Compatibility Mode is backward compatible with many existing devices, including the personal computer (PC) parallel port, and is a mode common to all IEEE 1284-compliant devices.

3.3.3 Extended Capabilities Port Mode (ECP Mode): An asynchronous, interlocked, byte-wide, bidirectional channel. A control line is provided to distinguish between command and data transfers. A command may optionally be used to indicate RLE count for data compression or channel address.

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

3.3.5 Negotiation: A signaling protocol used by a compliant host adapter to determine the capabilities and set the mode of an IEEE 1284-compliant peripheral. Both host and peripheral enter Compatibility Mode following reset. The host may then initiate the negotiation sequence. A compliant peripheral will respond correctly, while a noncompliant peripheral will not respond. At the end of the negotiation sequence, a compliant peripheral and host interface may enter one of the advanced modes. If a negotiation sequence fails, then both interfaces remain in Compatibility Mode.

3.3.6 Nibble Mode: An asynchronous reverse channel. Data bytes are transmitted as two sequential nibbles using the status lines.

4. Features and compliance

4.1 Overview

With the adoption of IEEE Std 1284-2000, there has been a tremendous increase in the types of peripherals that are being attached to the parallel port. This, combined with the increasing use of portable platforms, has led to the desire to share the parallel port with other peripherals. This standard defines two multiport configurations:

- Multiplexing
- Daisy chaining

The additional features of IEEE 1284 have created a requirement for a definition of the data link layer to support this interface, as well as a method for the software to manage port contention and sharing. This requirement is satisfied in the clauses titled

- Service provider interface (SPI) (Clause 6)
- Data link services (Clause 7)

Subclause 4.5 deals with electrical issues resulting from an architecture where multiple devices are sharing one port. This subclause makes changes to the interface recommendations of IEEE Std 1284-2000, but stays within the specification for compliance.

4.2 Multiport (MP)

This standard defines methods by which the host's parallel port can be connected to more than one peripheral device. Two types of multiport configurations are supported: multiplexor and daisy chain. Peripheral device ports are accessed on a time-shared basis and only one device may be selected at a time.

In both multiport configurations (multiplexor and daisy chain), interrupt requests are considered to be 'service requests'. Interrupt requests for an unselected device will not be passed through to the host interface. The peripheral shall maintain a local interrupt status. The host shall recognize this status when a 'Query Interrupt' cycle is performed. There is no guaranteed minimum latency with respect to port selection. Peripheral devices shall provide sufficient buffering to cope with real-time considerations.

If both a multiplexor and daisy chain are present, the multiplexor shall be directly attached to the host's IEEE 1284 parallel port. Daisy chains are then attached to the multiplexor's device ports. There can only be one Multiplexor attached to a host port.

4.2.1 Multiplexor

A multiplexor (Mux) is an *external device* interposed between the host port and two or more peripherals, permitting multiple parallel port devices to share a single host port. The Mux has a single port that connects to the host, and up to four device ports to which single peripherals or daisy chained devices may be attached. The Mux provides a mechanism by which the host may make a connection between its host port and any of the device ports. The Mux provides a *transparent* IEEE 1284 signaling path between the selected device port and host's port. When selected, the peripheral device appears to be directly connected to the host's port. Only one device port may be selected at any given time. Figure 1 shows a Mux with its device port numbering and attached peripherals. These peripherals can be non-MP devices or MP devices.

4.2.2 Daisy chaining

A daisy chain (DC) peripheral device has two ports: host and pass-through. The device's host port connects either to the host's parallel port, a device port of a Mux, or the pass-through port of another upstream DC device. The pass-through port is used to connect to the next downstream peripheral device in the daisy chain, if any. The last device may be a non-MP device. A daisy chain may be composed of up to 4 DC-compliant devices and an optional fifth non-MP device.

Figure 2 shows DC device numbering. The non-MP device at the end of the chain is referred to as the end-of-chain (EOC) device.

The protocol used to address, select, and query a DC device is called the command packet protocol (CPP). The CPP is a method of selection that does not interfere with any of the operational modes of IEEE Std 1284-2000 and is completely transparent to an IEEE 1284 device (see 5.2).

DC devices or non-MP devices

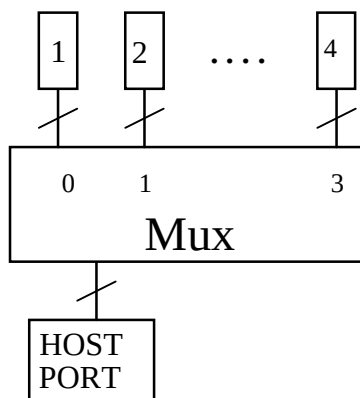


Figure 1—IEEE 1284.3 Mux

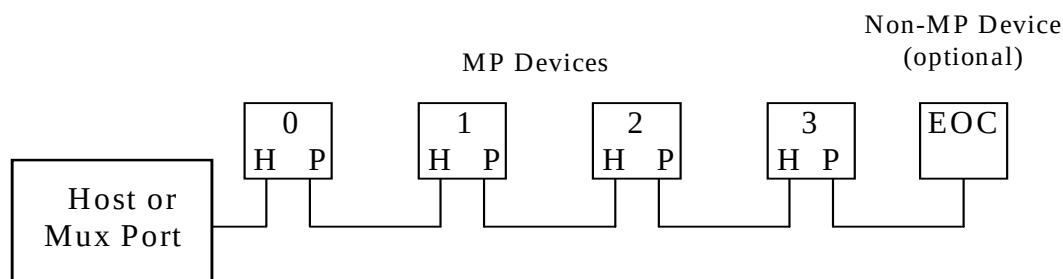


Figure 2—IEEE 1284.3 daisy chain (DC)

4.3 Service provider interface (SPI) features

The SPI layer is the software component that directly interfaces between other software components and the IEEE 1284 physical interface. This layer is responsible for providing the following:

- Data transfer
- Device enumeration of the IEEE 1284.3 topology
- Host adapter information
- Port contention

A complete description of the SPI can be found in Clause 6.

4.4 Data link layer features

The IEEE 1284.3 data link layer extends the services provided by the SPI by exchanging packets of data between one or more software components on the host and their peer components on the peripheral.

The IEEE 1284.3 data link layer provides the following:

- Unacknowledged connectionless packet data transfer services.
- Routing between peer software components.

A complete description of the data link layer services can be found in Clause 7.

4.5 Physical interface

DC devices shall meet all the electrical specifications of IEEE Std 1284-2000. However, for these devices, the potential for multiple pull-up termination resistors on non re-driven data lines requires the use of no less than 10 k Ω rather than the 1.2 k Ω values recommended in IEEE Std 1284-2000.

Figure 3 shows the electrical connections for the data lines of a system with a host, two DC devices, and a non-MP device at the end. All control and status lines shall be re-driven by each DC device. The maximum combined cable length between the host port and any peripheral shall not exceed 10 m. All cables shall be IEEE 1284-compliant.

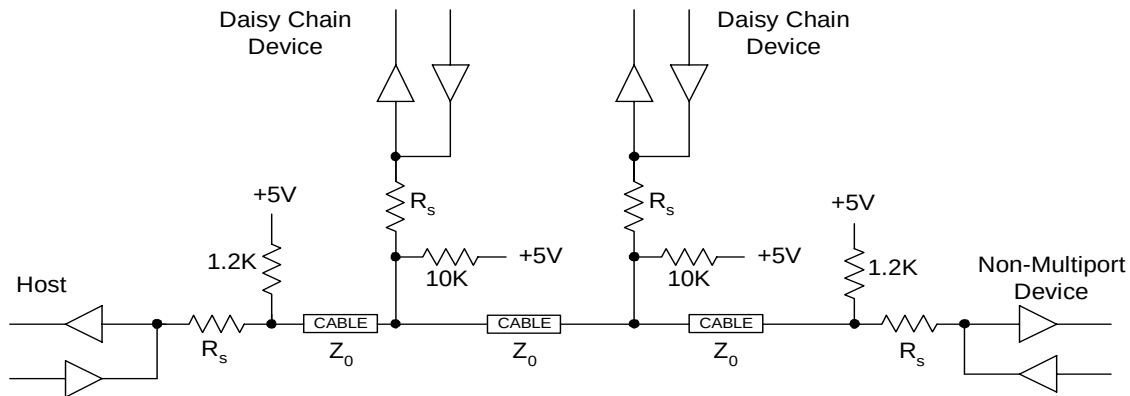


Figure 3—IEEE 1284.3-compliant data line circuitry example

4.6 Compliance

Devices that meet the mechanical interface requirements of IEEE Std 1284-2000, the electrical requirements of 4.5, and the protocol compatibility requirements of Clause 5 are referred to as “IEEE 1284.3-compliant” devices.

4.6.1 DC device

An IEEE 1284.3-compliant DC device shall meet the electrical requirements of 4.5 and the protocol requirements of 5.2 and 5.3.5. Additionally, it shall meet the compliance requirements in 4.4 of IEEE Std 1284-2000 and implement the Device ID string as specified in 7.6 of IEEE Std 1284-2000.

4.6.2 Mux device

An IEEE 1284.3-compliant Mux shall meet the electrical requirements of IEEE Std 1284-2000 and the protocol requirements of 5.2 and 5.4 of this standard.

4.6.3 SPI

Implementations of the IEEE 1284.3 SPI shall implement the functionality specified in 6.2 and 6.3.

4.6.4 Data link

Implementations of the IEEE 1284.3 data link shall comply with 7.3 and shall implement the functionality specified in 7.2.

5. Port-sharing protocols

5.1 Overview

When communicating with MP devices, the host uses a protocol that does not interfere with any IEEE 1284 protocols. This clause defines the protocol, which is transparent to non-MP Devices. It is first described in general terms, then in detail for each specific type of MP device.

5.2 Command packet protocol (CPP)

The host uses the CPP to select the peripheral device. This protocol is transparent to all IEEE 1284 communication modes. CPP defines a packet consisting of a preamble, a function ID, a command, and a terminator. The format of the initial portion of the CPP sequence is such that the data is self-qualifying; i.e., no strobes are required for the preamble or function ID. The initial portion of the CPP is transmitted as a sequence of data bytes onto the parallel port data lines without any host control line transitions. The data pattern provides synchronization and validation.

The MP device will have a means to detect, decode, and validate the CPP. Packet detection begins when stable data corresponding to the first byte of the preamble is detected. Detection aborts if any control line transitions are detected, or if stable data is found that does not match the next expected preamble byte. After the preamble and function ID, a command byte (or bytes) is sent. Depending on the type of MP device, each command byte may be qualified by nStrobe. An 0xFF on the data lines terminates the packet. Data is stable when it has remained unchanged for at least T_W . The host must maintain data on the parallel port for a minimum of T_H (see Table 4).

5.2.1 CPP format

The general format of the all CPP packets is shown in Table 1.

Table 1—General CPP format

Preamble	Function ID	Command Code	Terminator
0xAA 0x55	XX $\overline{XX}$ YY $\overline{YY}$	ZZ...	0xFF

5.2.1.1 Preamble

The first two bytes of the CPP shall be 0xAA followed by 0x55.

5.2.1.2 Function ID

Function ID's provide a mechanism to communicate with specific MP devices without causing interference with other devices. There are two types of Function ID's: top level and second level (see Table 2 and Table 3). Top-level ID's are associated with IEEE 1284.3 components that are required to implement the MP topology. The SPI shall support these types of function ID's. Top-level devices are as follows:

- Multiplexor
- Daisy chain

Second-level Function ID's are defined options to provide a mechanism for other types of peripherals to communicate without causing interference. The operational details of these functions are beyond the scope of this standard. These are essentially CPP escape codes. Examples of second-level devices are as follows:

- Dongles
- Vendor unique
- Extension key (reserved for future IEEE 1284.3 extensions)

The SPI, as defined in this standard, does not support second-level codes.

Table 2—Defined top-level Function ID's

Function ID codes	Type of device or action
0x00 0xFF 0x87 0x78	Daisy chain
0xF0 0x0F 0x52 0xAD	Multiplexor

Table 3—Defined second-level Function ID's

Function ID codes	Type of device or action
0xFA 0x05 0xBC 0x43	Dongle (security) devices
0xFA 0x05 0x61 0x9E	Extension key...indicates two additional extension bytes follow
0xFA 0x05 0x27 0xD8	Vendor unique sequence follows

5.2.1.3 Command codes

The command code field of the CPP contains one or more bytes, which are specific to the function ID. The control lines may be used to qualify these command bytes. Valid command codes are from 0x00–0xEF. Values from 0xF0–0xFF are invalid.

5.2.1.4 Termination

The termination field follows the command sequence of the CPP. Termination is defined by a value of 0xFF on the data lines for T_W . During the preamble and function ID times packet, detection shall be terminated by any transition of a host control line.

5.2.2 General CPP timing

Figure 4 is an example of CPP timing for a DC device. At event 1, the CPP sequence begins at the IEEE 1284-compatible Idle state. At event 2, the host drives the first byte of the preamble onto the data lines. The host shall maintain the data for a minimum of T_H . The MP device waits T_W time (event 3) before recognizing the byte as stable data. Each of the preamble and function ID bytes is driven in a similar fashion according to the timing shown. In this example, the command is qualified by nStrobe falling low at event 15 and then being driven high at event 16. The terminator byte is driven at event 17 and recognized by the device at event 18.

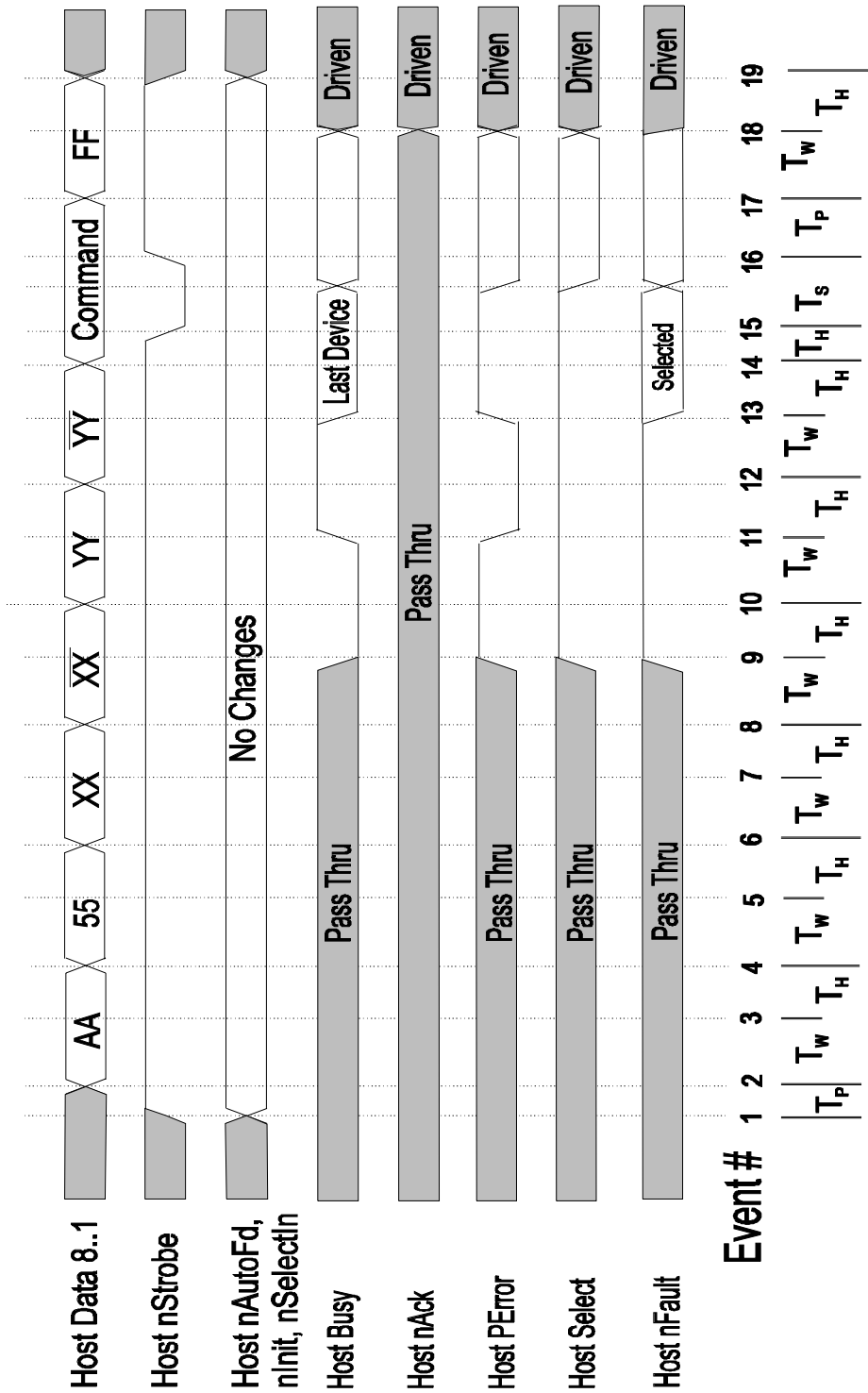


Figure 4—Daisy chain CPP timing example

Table 4 defines CPP timing values.

Table 4—CPP timing values

Time	Minimum	Maximum	Description
T_H	1800 ns	Infinite	Host data width
T_W	750 ns	1200 ns	Peripheral validation time
T_S	900 ns	2000 ns	Command strobe pulse width
T_P	900 ns	Infinite	Setup or hold time
T_A	0	100 ns	Host to pass-through data propagation delay
T_T	10 ns	250 ns	Signal transition time
T_{PR}	1000 ns	1450 ns	Peripheral response time
T_{HR}	1700 ns	Infinite	Host response time
T_{PSR1}	1700 ns	Infinite	Peripheral status response 1
T_{PSR2}	0	750 ns	Peripheral status response 2
T_{HRSEL}	2000 ns	Infinite	Selected status propagation delay
T_{CR}	0	750 ns	Peripheral command response time
T_{HSR}	1050 ns	Infinite	Host command response time

5.2.2.1 Preamble and function ID timing

T_H is the time the host shall maintain stable values on the data lines. T_W is the time in which the MP device shall detect stable data (see Figure 5).

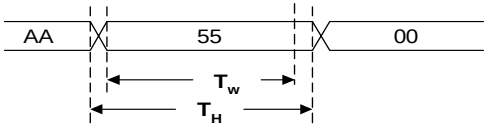


Figure 5—Preamble and function ID timing

5.2.2.2 MP device response time

This timing applies to the periods in the CPP cycle where the host is waiting for a response from the MP device (see Figure 6).

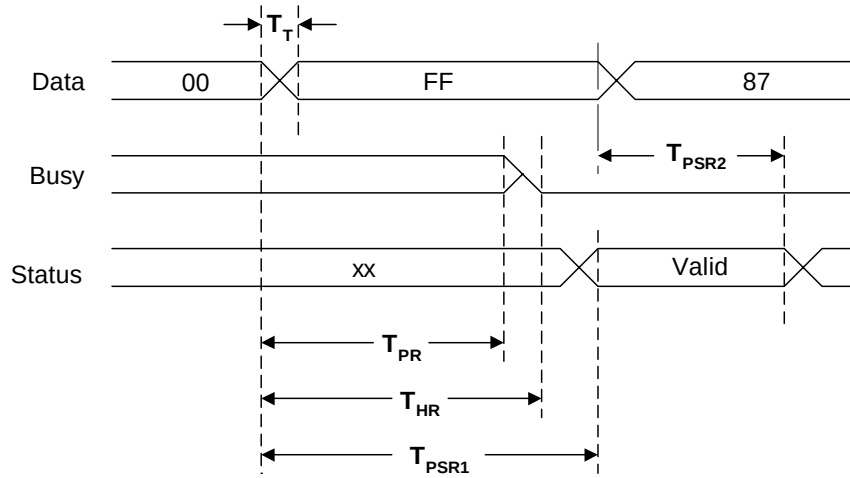


Figure 6—CPP response time example

$T_T = 250 \text{ ns max.}$ (see 8.3 of IEEE Std 1284-2000)

Maximum time for data to change and for that change to propagate down the cable.

$$T_{PR} (\text{min.}) = T_T + T_W (\text{min.}) = 1.0 \mu\text{s}$$

$$T_{PR} (\text{max.}) = T_T + T_W (\text{max.}) = 1.45 \mu\text{s}$$

Minimum time for MP device response to CPP function ID.

For example, in the DC CPP sequence, the DC device must respond with Busy and PE within T_{PR} of the transition from 0x00 to 0xFF on the data lines.

$$T_{HR} = 1700 \text{ ns min.}$$

Minimum time the host shall wait to read valid response from MP device.

A special case occurs in the DC Select command. The host shall wait T_{HRSEL} after issuing the Select command and before asserting nStrobe. This is to allow the Select status to propagate from the last device on the DC to the host.

$$T_{HRSEL} = T_{PR} (\text{max.}) + T_T + 3T_A = 1.45 \mu\text{s} + 0.250 \mu\text{s} + 3(0.100) = 2.0 \mu\text{s}$$

$$T_{PSR1} = T_W (\text{max.}) + 2T_T = 1.7 \mu\text{s}$$

Minimum time the host must wait until reading the pass through status lines.

$$T_{PSR2} = T_W (\text{min.}) = 750 \text{ ns}$$

Maximum time the host has to read the pass through status after changing the CPP symbol.

5.2.2.3 Command strobe

Figure 7 is used to indicate timing constraints of MP device response when command strobe is utilized to validate commands.

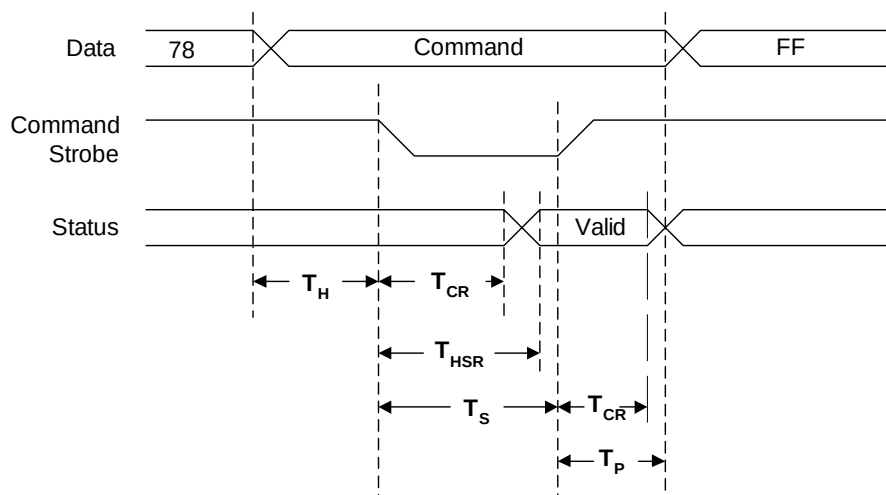


Figure 7—CPP common strobe timing

$$T_{CR} (\text{min.}) = 0$$

$$T_{CR} (\text{max.}) = 750 \text{ ns}$$

Time for MP device to respond with status in response to the host assertion of the command strobe signal.

$$T_{HSR} = 1050 \text{ ns min.}$$

Time the host shall wait for a valid response from the MP device in response to a command strobe.

$$T_S = 900 \text{ ns min.}$$

Command strobe pulse width.

$$T_P = 900 \text{ ns min.}$$

Various setup and hold times.

5.3 DC architecture

5.3.1 Interconnection

This subclause describes the physical configuration of the daisy chain. It consists of one or more devices having both host and pass-through connectors for the parallel port as defined in IEEE Std 1284-2000 (Type A, B, or C). These devices may be used together by connecting the host connector on one device to the computer's parallel port and the pass-through connector to the host connector of the next device. Up to four devices may be connected in this fashion, as shown in Figure 2. Any parallel port device, which is a non-MP device (i.e., a standard printer), may be connected to the pass-through connector of the last device for a total of five devices on the chain.

The daisy chain device host port connector shall be one of the following:

- IEEE 1284 Type A plug
- IEEE 1284 Type B receptacle
- IEEE 1284 Type C receptacle

The DC device pass-through port connector shall be one of the following:

- IEEE 1284 Type A receptacle
- IEEE 1284 Type C receptacle

5.3.2 Implementation

The parallel port contains three groups of signals: Status, Control, and Data.

The status lines (Busy, nAck, PError, Select, and nFault) are inputs to the host. The DC device shall be able to read these inputs from the pass-through port and re-drive them to the host port. It also shall be able to block the pass-through status to the host port.

The control lines (nSelectIn, nInit, nAutoFd, and nStrobe) are outputs from the host. The DC device shall be able to receive host port control signals and re-drive them to the pass-through port. The daisy chain device shall be able to latch the state of the host port control signals driven to the pass-through port.

The Data lines (Data8–Data1) are bidirectional, and shall not interfere with the bidirectional data flow between the host and pass-through ports.

With the DC implementation the Status and Control lines are point to point and shall be governed by the driver and termination requirements as outlined in IEEE Std 1284-2000. The interface to the data lines shall be governed by this document (see 4.5).

5.3.3 Operation

DC devices default to transparent mode until addressed and selected. Upon selection, the peripheral and host port are in the Compatibility Mode. Once selected, the peripheral may be negotiated into another mode. The interface shall always terminate to Compatibility Mode prior to de-selection. IEEE Std 1284-2000 details the method which the host and peripheral may use to negotiate a communications mode. See Table 5 for an example of DC device state changes. In this example, DC device 1 is attached to the host and DC device 2 is attached to the pass-through port of DC device 1.

Once a device is addressed, that address remains in effect until a new Assign Address command is completed.

In the un-selected state, the DC device is considered transparent and shall pass through and not alter Data, Control or Status lines. In selected state, the selected DC peripheral shall be the only device actively communicating with the parallel port host. The state of the output control lines is latched at the time of selection. While selected, the pass-through port of the DC device shall maintain the value of the output control lines in this latched state.

When selected, the host port status lines of the DC device reflect the state of the DC device, not that of the pass-through port.

When not selected, the host status lines reflect the status of the DC device pass-through port.

Table 5—Daisy chain device state table

Action	DC device 1 state	DC device 2 state	Host: IEEE 1284 Mode
Power Up	Un-Addressed, Un-selected	Un-Addressed, Un-selected	Compatibility Mode
Assign Address	Addressed 0, Un-selected	Addressed 1, Un-selected	Compatibility Mode
Select Device 1	Selected	Un-selected	Compatibility Mode
Negotiate	Selected	Un-selected	New Mode
Data Transfer	Selected	Un-selected	New Mode
Terminate	Selected	Un-selected	Compatibility Mode
De-Select Device	Addressed, Un-selected	Addressed, Un-selected	Compatibility Mode
Select Device 2	Un-selected	Selected	Compatibility Mode
Negotiate	Un-selected	Selected	New Mode
Data Transfer	Un-selected	Selected	New Mode
Terminate	Un-selected	Selected	Compatibility Mode
De-Select Device	Addressed, Un-selected	Addressed, Un-selected	Compatibility Mode

When a CPP preamble has been detected the host status lines reflect the appropriate status response for the DC CPP protocol (see Figure 8).

5.3.4 General DC CPP

This subclause details the CPP for managing the DC devices. See Figure 9 for logic flow information.

Figure 10 is a timing diagram showing a DC CPP transaction. At event 1 the CPP sequence is initiated. The host control signals $nAutoFd$, $nInit$, and $nSelectIn$ are not allowed to transition until event 27. Because the CPP is initiated from IEEE 1284 Compatible Idle state, the signal $nStrobe$ is high at event 1 and not allowed to transition until event 22. Pass-through control signals shall be stable at event 2. DC devices are allowed T_A maximum time for propagation of the control signals from the host port to the pass-through port. Similarly, they are allowed T_A time to propagate status lines from the pass-through port to the host port. The pass-through port status lines are driven to the host status lines.

At event 3 the host places the first preamble value on the data lines. The host shall hold this data for a minimum of T_H and shall be accepted by the DC device as stable CPP sequence data after time T_W , event 4. The host follows the first preamble value with the remaining values as shown at events 5, 7, and 9. After the DC device detects a stable 0xFF in the function ID, it responds by taking over the host status lines and setting $Busy$ low, and $PError$, $Select$, and $nFault$ high (event 10). Status lines are received by the DC device on the pass-through port at events 11–13 (as early as event 11 and as late as event 13), events 15–17, and events 19–21.

The mechanism used by a DC device to determine if it is the last device in the chain is the status information. After a stable 0x87 is recognized, the device will set $PError$ low, and $Busy$, $Select$ and $nFault$

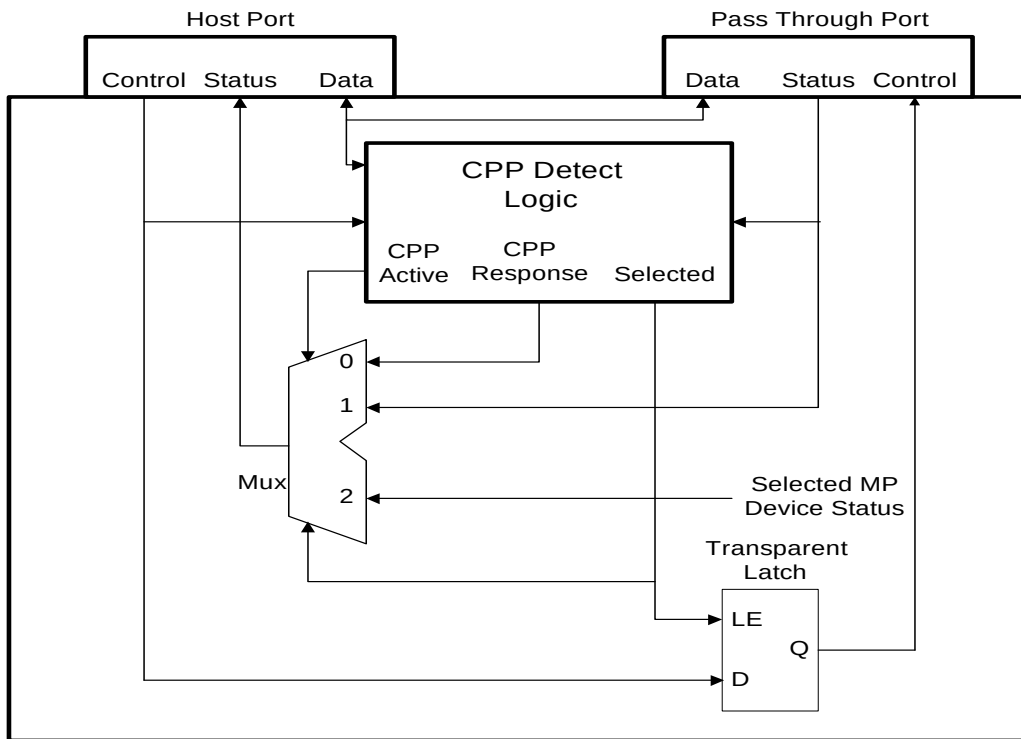


Figure 8—DC device signal flow example

high, event 14. Each device is responsible for monitoring its pass-through port to determine if a downstream DC device is attached. The device does this by comparing its status lines to the pass-through status lines. A DC device detects a downstream device attached if its host and pass-through status lines are in identical states between events 11-13, and between events 15-17. If all the above conditions are not met, then the DC device is the last DC device on the chain.

After a stable 0x78 is detected, the DC device sets PError and Select high, event 18. Busy is set low if other DC devices were detected on the pass-through port, or set high if it determines that it is the last DC device. If the device is the last device, then nFault is set high if the device is currently selected. If the device is not the last device, then nFault is set high if the device is selected or the pass-through nFault is high. The device will maintain these values on the status lines until the falling edge of nStrobe, or until it aborts the packet due to an unexpected control line transition. This handshake allows the host to determine both if a device is present before finishing the communication (so that it will not pulse nStrobe to a device that would misinterpret the data), and if any device is currently selected. The handshake also allows a device to determine that it is the last DC device so that it can block nStrobe to any non-MP device connected to its pass-through port.

The host places the DC Command on the data lines at event 20. The host signal nStrobe is driven low at event 22. The DC Device uses the falling edge of nStrobe to accept the command (event 23) and stop driving the host status line values for last device and selected. The status lines from the pass-through port are driven back to the host beginning at event 26. The signal nStrobe is driven high by the host at event 24.

If a device finds that it is the last DC device in the chain it shall assume that a non-MP Device follows. The signal nStrobe is latched as early as state 18 and as late as state 21, and driven high to the pass-through port

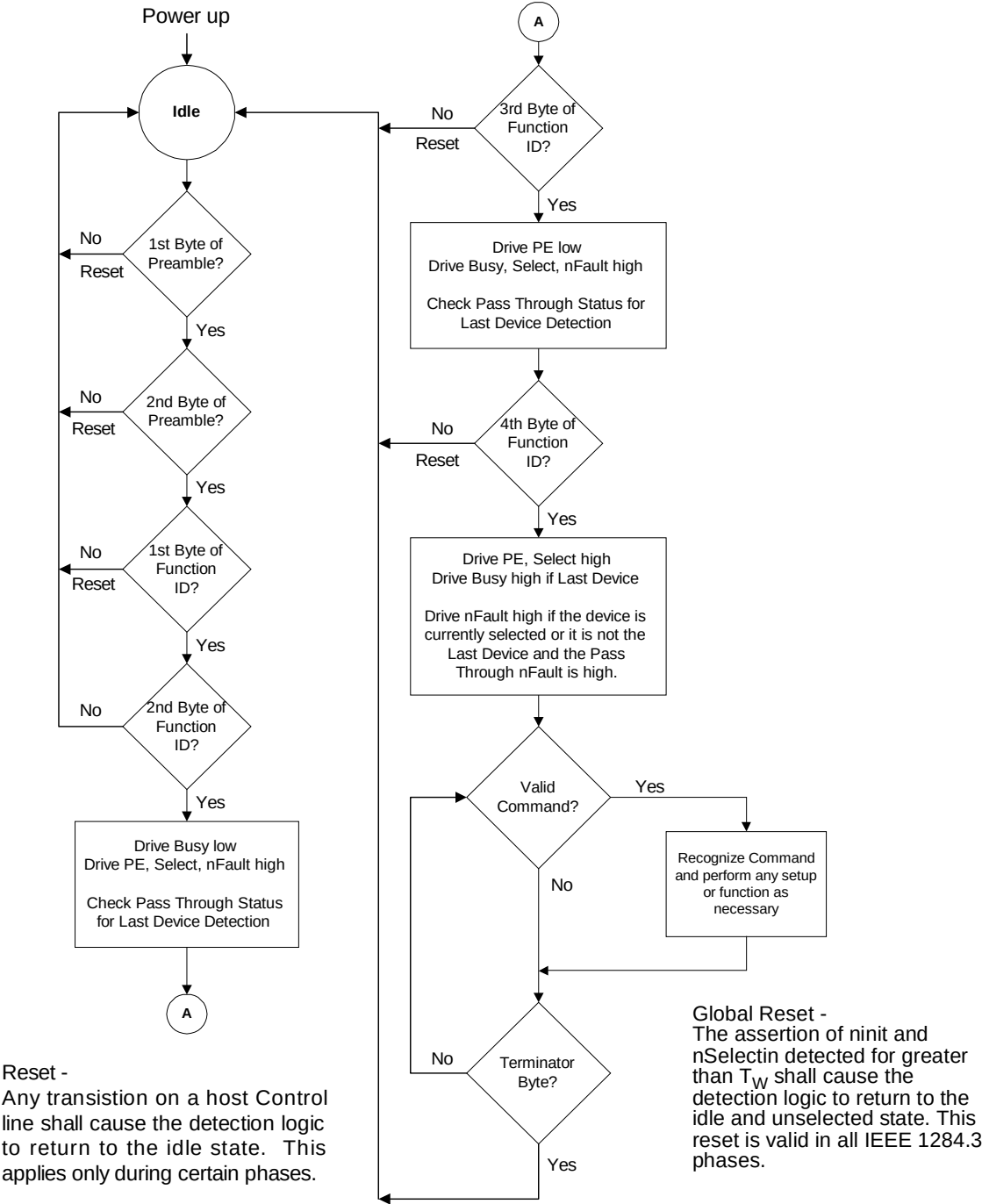


Figure 9—DC CPP command flowchart