

1532™

**IEEE Standard for In-System
Configuration of Programmable
Devices**

IEEE Computer Society

Sponsored by the
Test Technology Technical Council



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

10 January 2003

Print: SH95071
PDF: SS95071

IEEE Standard for In-System Configuration of Programmable Devices

Sponsor

Test Technology Technical Council
of the
IEEE Computer Society

Approved 11 December 2002

IEEE-SA Standards Board

Abstract: The communication protocol described by IEEE Std 1149.1™-2001 (Standard Test Access Port and Boundary-Scan Architecture) has been adopted by this standard for providing standardized programming access and methodology for programmable integrated circuit devices. Devices that implement this standard will first be compliant with IEEE Std 1149.1-2001, which is used for testing purposes. A device, or set of devices, implementing this standard can be programmed (written), read back, erased, and verified, singly or concurrently, with a standardized set of resources. Sample implementation and application details (which are not part of this standard) are included for illustrative purposes.

Keywords: complex programmable logic device (CPLD), erasable programmable read-only memory (EPROM), field programmable gate array (FPGA), Flash RAM, in-system (or in situ) configuration (ISC), programmable device

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

Copyright © 2003 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 10 January 2003. Printed in the United States of America.

Print: ISBN 0-7381-3507-0 SH95071
PDF: ISBN 0-7381-3508-9 SS95071

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.

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

Introduction

(This introduction is not part of IEEE Std 1532-2002, IEEE Standard for In-System Configuration of Programmable Devices.)

The development of this standard began with a preliminary meeting held 19 April 1996. There, a group of Programmable Device vendors and users discussed the possibility of standardizing the programming process for these devices. Subsequent meetings were held regularly and agreement was developed for basing the programming process upon the IEEE Std 1149.1 communication protocol, because many of these devices were expected to support IEEE Std 1149.1 for testing purposes. This group adopted as its mission:

To define, document, and promote the use of a standardized process and methodology for implementing programming capabilities within programmable integrated circuit devices, utilizing (and compatible with) the IEEE Std 1149.1 communication protocol. This standard would allow the programming of one or more compliant devices concurrently, while mounted on a board or embedded in a system, known as “In-System Configuration.” Concurrent programming may often result in significant programming time efficiencies. The In-System feature would address the need to configure or reconfigure, read back, verify or erase programmable devices after they have been installed by a manufacturing process. This eliminates handling damage and the need for manufacturing steps and inventory management related to preprogrammed devices.

This work received the support of the IEEE Std 1149.1 Working Group which ultimately urged that this effort become its own standard within the Test Technology Technical Committee of the IEEE Computer Society, in July of 1998. The initially approved version of this standard described the necessary hardware elements for compliance. By making this available quickly and early, the development of compliant silicon was facilitated and its availability accelerated. The next revision of this standard described the software elements necessary for full implementation of this standard. The software elements described include extensions to the BSDL file that describe the programming algorithm and hardware implementation of this standard. Additionally a new data file format has been described. This revision further extends the standard by adding support for devices that require adaptive programming algorithms along with some slight modifications to the initial standard that will help to better meet user’s needs.

Following is a list of participants in the In-System Configuration Working Group of the Test Technology Technical Council. Voting members at the time of publication are marked with an asterisk (*).

Neil G. Jacobson, *Chair*

David Bonnet, *Vice-Chair*

Dennis Lia, *Secretary*

Mark Moyer, *Editor*

Kenneth P. Parker, *Editor Emeritus*

Turgut Abacioglu
John Andrews
Don Arnold
Bill Aronson
Marco Barahmand
Jason Baumbach*
Glenn Baxter
Menachem Blasberg
Terry Borroz
Stephen Brown
Ian Bryant
Steven Butkovich
Tapan Chakraborty
Sung Chung*
C. J. Clark*
Ray Dellecker *
Yadvindra Dhani
Wim Driessen*
Alon Drory
Ted Eaton*
Rob Eccles
Peter van den Eijnden
Bernardo Elayda

P. D. Felen
Judy Chen Feng
Dan Gravitz
Jack Griffin*
Kurt Guntheroth
Peter Hansen
Steve Harrison
Bernie Haverkamp
Alan Herrmann*
Brad Ishihara*
John Joy
Raj Kamdar
Doug Kay
Arthur Khu
Peter Larsen
Bertrand Leigh
Frank Lien
Keith Loftstrom
Patrick McHugh
G. Meijerink
Matthew Murphy
Gary O'Brien

David Paul
Bruce Peterson
Etienne Racine
Bapcha Ramamurthy
Doug Raymond
Jacky Renaux
Robert Russell
David Lee Rutledge
Bobbie Saville
Sanjeev Sharma*
Steve Stark
Chung Sun
Jerry Talen
Howard Tang*
Elke Tiemeyer
Douglas Way*
Thomas Wenzel
Mike Westermeier
Kevin Wible
Bill Wilkie
Carmy Yellin
Dan Young
Piotr Zbysinski

The following members of the balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Morris Balamut
Roger Bennetts
David Bonnett
Terry Borroz
Tapan Chakraborty
Keith Chow
C. J. Clark
Luis E. Cordova

Frans De Jong
Heiko Ehrenberg
Peter van den Eijnden
Kazumi Hatayama
Neil G. Jacobson
Adam Ley
Dennis Lia

Yinghua Min
Mark Moyer
Benoit Nadeau-Dostie
Bruce Peterson
Mike Ricchetti
Gordon Robinson
Robert Russell
Douglas Way

When the IEEE-SA Standards Board approved this standard on 11 December 2002, it had the following membership:

James T. Carlo, *Chair*
James H. Gurney, *Vice Chair*
Judith Gorman, *Secretary*

Sid Bennett
H. Stephen Berger
Clyde R. Camp
Richard DeBlasio
Harold E. Epstein
Julian Forster*
Howard M. Frazier

Toshio Fukuda
Arnold M. Greenspan
Raymond Hapeman
Donald M. Heirman
Richard H. Hulett
Lowell G. Johnson
Joseph L. Koepfinger*
Peter H. Lips

Nader Mehravari
Daleep C. Mohla
William J. Moylan
Malcolm V. Thaden
Geoffrey O. Thompson
Howard L. Wolfman
Don Wright

*Member Emeritus

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

Alan Cookson, *NIST Representative*
Satish K. Aggarwal, *NRC Representative*

Noelle D. Humenick
IEEE Standards Project Editor

Contents

1.	Overview	1
1.1	Scope	1
1.2	Organization of the standard	1
1.3	Context	2
1.4	Objectives	2
2.	References	3
3.	Definitions and acronyms	3
3.1	Definitions	3
3.2	Acronyms	9
4.	System modal states	9
4.1	Definition of system modal states	10
4.2	Description of system modal state transitions	12
4.3	System modal state transition timing	15
4.4	System I/O pin behavior	18
5.	ISC architecture	22
5.1	Overview	22
5.2	Specification	35
5.3	Description	38
6.	ISC Instructions	39
6.1	General properties of ISC instructions	39
6.2	Mandatory ISC instructions	46
6.3	Optional ISC programming instructions	52
6.4	Optional ISC program control/security instructions	59
6.5	Optional address/data access instructions	63
6.6	Requirements for IEEE Std 1149.1-2001 masters and ISC programming algorithms	68
6.7	Adding new instructions for ISC programming support	70
7.	Data registers	70
7.1	Device identification register	71
7.2	Bypass register	72
7.3	Boundary scan register	72
7.4	ISC_Default	72
7.5	ISC_Config	73
7.6	ISC_PData	74
7.7	ISC_RData	74
7.8	ISC_Sector	74
7.9	ISC_Info	74
7.10	ISC_Data	75
7.11	ISC_Address	75
7.12	ISC_Inc	75

8.	Conformance and documentation requirements	75
8.1	Conformance	75
8.2	Documentation	76
8.3	BSDL extension for ISC description (STD_1532_2002)	78
8.4	ISC extension structure	79
8.5	ISC BSDL attribute definitions	81
9.	Data file format	110
9.1	Data file syntax specification	110
9.2	Description	113
9.3	Mapping data onto a flow	114
9.4	Merging flows for concurrent programming	115
9.5	Data file transmission CRC calculation	117
Annex A	(informative) Use of device features	119
Annex B	(informative) ISC programming flows	123
Annex C	(informative) Example ISC BSDL/data file	134

IEEE Standard for In-System Configuration of Programmable Devices

1. Overview

1.1 Scope

This standard describes a series of mandatory and optional programming instructions and associated data registers that define a standard methodology for accessing and configuring programmable devices. These additional registers and instructions extend the capabilities of devices that comply with IEEE Std 1149.1™-2001 such that the IEEE Std 1149.1-2001 Test Access Port may be used for configuration activities. A data description format and extensions to Boundary-Scan Description Language (BSDL) are also specified that provide for the development of standardized automation tools for device programming.

1.2 Organization of the standard

This standard is divided into nine clauses and three annexes, as follows:

- Clause 1 describes the scope, context, objectives, and organization of this standard.
- Clause 2 provides references to other IEEE documentation needed to understand this document.
- Clause 3 defines terms and acronyms used in this standard.
- Clause 4 defines the variations of system state that are unique to programmable logic devices.
- Clause 5 describes the general architecture of an in-system configurable (ISC) device.
- Clause 6 describes the ISC instruction set.
- Clause 7 describes the ISC register set and the operation of each register.
- Clause 8 describes conformance and documentation requirements.
- Clause 9 provides a data file format to be used to describe data to be downloaded to an ISC device.
- Annex A provides a device designer with a rationale for implementing various features of this standard.
- Annex B describes programming flows for combinations of devices compliant with this standard.
- Annex C gives a coordinated example of a BSDL file and data file for a hypothetical ISC device.

1.3 Context

Figure 1 shows a printed circuit board containing many types of devices. Of these, some could be compliant with IEEE Std 1149.1-2001 for the support of testing activities. Three devices labeled “ISC” are in-system configurable, meaning they are programmable devices that may not have their programming states loaded into them before the board has been assembled. These devices may be programmed after they have been attached to the board, via a physical and logical protocol. The physical protocol for programming utilizes the IEEE Std 1149.1-2001 Test Access Port (TAP) pins, which allows a multiplicity of devices (both ISC and conventional) to be accessed serially with four or five wires. The logical protocol is based upon new instructions added to the IEEE Std 1149.1-2001 instruction set and is the subject of this standard document.

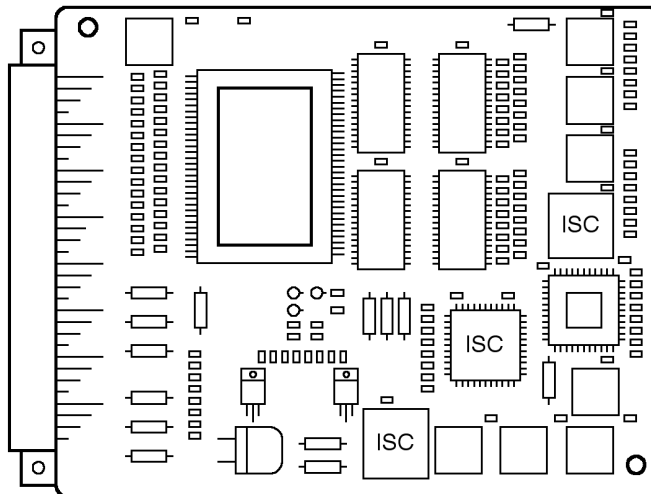


Figure 1—A printed circuit board containing a variety of components, including three ISC integrated circuits

The ISC devices also contain testability circuitry compliant with IEEE Std 1149.1-2001. This allows them to participate in manufacturing tests that detect and diagnose faults such as open solder joints and shorts either before or after they are programmed.

1.4 Objectives

The objective of this standard is to enhance user access to IEEE Std 1149.1-2001-based programmable devices. Since such devices are built on the foundation of the IEEE Std 1149.1-2001 Test Access Port and TAP Controller state machine, they shall already comply with that standard.

Devices that adhere to this standard will realize significant savings in software, product development, and manufacturing costs related to ISC. New tools will not need to be developed to support new vendor parts with ISC features as they arrive in the marketplace. These parts can be integrated immediately into existing systems and tools.

In addition, tools based on this standard may be developed upon the existing IEEE Std 1149.1-2001-based tool sets and knowledge base, rapidly incorporating the features of this standard.

This standard also allows for multiple devices from independent vendors to be programmed concurrently. Development time and manufacturing costs (in terms of programming time) can thereby be significantly reduced, even in the multivendor environment. In the past, one family of devices from a single vendor may have been incompatible with another family from that same vendor, with respect to the programming process.

The standard also serves board and system designers by providing a consistent framework for system initialization, stabilization, and event initiation, which is important for robust system design.

The standard will realize the following additional benefits:

- It will serve as the development model for new entrants to the ISC industry and guarantee their immediate access to all the latest tools, development systems, and test and manufacturing flows and processes.
- It will facilitate innovations potentially including applications in reconfigurable computing, emulation, diagnostics, and field upgrading that would be usable across vendors.
- It will encourage tool vendors targeting the programmable device marketplace that there will be sufficient return on their investment to warrant tool development.

2. References

This standard shall be used in conjunction with the following standards. When the following standards are superseded by an approved revision, the revision shall apply.

IEEE Std 1149.1-2001, IEEE Standard Test Access Port and Boundary-Scan Architecture.^{1, 2}

3. Definitions and acronyms

3.1 Definitions

For the purposes of this standard, the following terms and definitions apply. See also IEEE Std 1149.1-2001 for definitions of terms used in this document.

3.1.1 adaptive programming (algorithm): This term may be used when the minimum burn time may vary among identical devices. To optimize the overall programming time, the device can be queried for the status of the programming operation, which will be given by the optional “Operation Success Indication.” If the programming operation has been successfully completed, then the next step in the programming process can commence. If the device has not completed its programming operation to satisfaction, then it will continue until completion or until the maximum time has been met. This process will require the looping of one or more instructions for successful completion. *Contrast: deterministic programming.*

NOTE See 5.1.10.

3.1.2 addressing: Programmable devices usually contain one or several arrays of memory locations. Each memory location has an address, which must be provided when reading or programming memory content.

3.1.3 asymmetrically writeable device: A device with program memory that can only be written to one state. For example, a “1” may be written into a location containing a “0”, but not vice-versa. Such devices must be erased to the blank state before new program data can be written. *Contrast: symmetrically writeable device.*

¹IEEE 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/>).

²The IEEE standards or products referred to in Clause 2 are trademarks owned by the Institute of Electrical and Electronics Engineers, Incorporated.

3.1.4 blank: Programmable devices usually contain one or several arrays of memory locations. The data placed in these locations constitute the device program. Many devices are asymmetrical with respect to writing data into these locations. For example, a “1” may be written into a location containing a “0”, but not vice-versa. Thus, a “blank” device is filled with a field of overwriteable states. For symmetrically writeable devices, the “blank” state is usually defined as a field of states that implement no programmable functionality. *See also:* **device program**.

3.1.5 burn time: An interval of time during which a set (or subset) of device programming bits is presented to on-chip programming functions, which become the definition of the logic functions the device will subsequently implement. In some older technologies, tiny on-chip fuses are vaporized (hence the term *burn*), which requires a minimum time to achieve. Burn time is technology dependent and may be nearly instantaneous in some devices or require many milliseconds in others. Synonyms include bake time, cook time, and soak time.

3.1.6 chain: Two or more IEEE Std 1149.1-2001-compliant devices that share common test clock (TCK) and test mode select (TMS) signals and are linked together with the test data output (TDO) signal of one device connected to the test data input (TDI) signal of a subsequent device. The overall chain may be accessed by a 4 or 5-wire test access port (TAP). *See also:* **test access port (TAP)**.

3.1.7 checksum: This integer is created by a mathematical process that operates on a stream of data. The integer for this data is then published so that it can be compared to subsequently recalculated integers for that same data. If the data have not been changed or disturbed, the published integer should match a newly calculated integer. The mathematical process is often based upon binary addition (to some modulus) or logical Exclusive OR of bytes or words. There are well-known flaws with such processes that can allow damaged data streams to masquerade as good by having the same checksum. *See also:* **CRC cyclic redundant codes**.

3.1.8 completion: In the context of an ISC instruction, this process occurs when the sequence of activities performed during execution has finished. However, the requirements for execution may not be met, or error conditions may be encountered such that successful completion is not possible. These events constitute unsuccessful completion. *See also:* **Execute**.

3.1.9 complex programmable logic device (CPLD): A high-capacity programmable logic device (PLD) usually provided in packages with 40 or more pins, essentially an array of programmable logic blocks interconnected by a routing matrix.

3.1.10 concurrentization: The process of making two or more devices work concurrently.

3.1.11 concurrent operation: The ability to conduct programmable device operations in two or more devices in the same IEEE Std 1149.1-2001 chain concurrently in time. During this process, all devices in a chain operate continuously in the *Run-Test/Idle* TAP Controller state. In particular, the burn times of the devices can be made to elapse concurrently.

3.1.12 CRC cyclic redundant codes: This code can be used as an improved checksum process. Such a code (properly constructed) is much less prone to aliasing a bad data stream with a good stream because it encodes the positional information of 1s and 0s in a stream as well as their frequency. (*See also:* **checksum**.)

3.1.13 deterministic programming (algorithm): A device implements this algorithm when the minimum required burn time to satisfy programming operations is documented in the BSDL file and does not change for like devices. *Contrast:* **adaptive programming**.

3.1.14 device program: A sequence of bits that, when loaded into a programmable device via a protocol, subsequently define the functions it will implement. *See also:* **volatile program** and **nonvolatile program**.

3.1.15 displaced: An instruction in the IEEE Std 1149.1-2001 instruction register is displaced when another instruction is shifted in and then a falling edge of TCK in the *Update-IR* TAP Controller state occurs. Note that any instruction besides IDCODE (the default instruction jammed into the IEEE Std 1149.1-2001 instruction register) will be displaced immediately if the TAP Controller enters the *Test-Logic-Reset* TAP Controller state.

3.1.16 duration: In the context of an ISC instruction, a period of time that starts when a TAP Controller first enters the *Run-Test/Idle* TAP Controller state. This period of time is given for each ISC instruction, which specifies how much time (typically a minimum) the action(s) performed by that instruction will take to complete.

3.1.17 dynamic concurrentization: This process merges the programming algorithms of multiple devices at run time and is based on the intermediate results of the programming operations. With this process, decisions regarding when information is shifted into or out of a device or when operations are executed are made immediately before the action takes place. *Contrast:* **static concurrentization**.

3.1.18 electrically erasable programmable read-only memory (EEPROM): Programmable read-only memories have evolved from mask-programmable devices to devices that were erasable by exposure to ultraviolet light (EPROM) to devices that can be electrically erased.

3.1.19 enabled protection mechanism: ISC devices may be implemented with the ability to protect the memory content from erasure, reading or writing, in various combinations. When any of these mechanisms is available, it can be enabled, thus preventing the action it can protect against. *See also:* **protect**.

NOTE See 5.1.11.

3.1.20 erasable programmable logic device (EPLD): At first an EPROM-based device, the term now includes flash and EEPROM-based devices as well.

3.1.21 erase: An operation that removes any previously installed device program by changing programmed bits to their unprogrammed state. Erasure may be implemented in several ways. For example, there may be a “bulk erase” function that erases the entire device program in one action. There may be selective, or addressed erasure that only erases a specified “word” of the program. Between these two extremes, there may be a “sector” erase feature that erases a group of words. This operation may be inhibited if the device contains a security feature enabled by a Protect operation.

3.1.22 execute: In the context of an ISC instruction, execution occurs after the TAP Controller has proceeded to the *Run-Test/Idle* TAP Controller state. In this state, the instruction performs a specific function that may contain multiple steps or a set of activities. After a minimum amount of time (or clocking of TCK) has occurred, the instruction completes execution. Additional time (or clocking of TCK) beyond this minimum requirement does not affect the results of the instruction’s execution.

3.1.23 explicit addressing: Address information is provided by an external algorithm when operating on memory content.

3.1.24 field-programmable gate array (FPGA): A programmable alternative to a masked gate array using a channeled programmable interconnect matrix to join blocks of configurable logic. Physical characteristics such as line length and number of fuses are not known until routing. This may lead to some timing unpredictability. The advantage comes in logic density and flexibility.

3.1.25 fixed system pin: A system pin with an I/O function that is independent of configuration information programmed into the device and has not been designated as an ISC system pin by the device designer.

3.1.26 flash random access memory (Flash RAM): A memory technology similar to EEPROM but implemented with a single transistor rather than two.

3.1.27 flow: A logical sequence of ISC instructions that performs a basic programming function such as erasing programmable memory, verifying it, or programming it.

3.1.28 generic array logic (GAL): An earlier programmable logic technology still widely used, featuring a fixed OR array and programmable AND array.

3.1.29 implicit addressing: Address information is provided by an internally managed process when operating on memory content. For example, successive memory locations may be sequentially accessed by an “auto-increment” addressing process supported by the device such that an external algorithm does not have to be aware of how memory locations are organized.

3.1.30 in-system configuration (ISC): The ability to load, read back, or erase programming bits in a programmable logic device after it has been attached to a printed circuit board. *See also:* **device program**.

3.1.31 ISC_Disable_Erase signal: An optional signal used to indicate that program data erasure is not allowed.

NOTE See 5.1.11.3.

3.1.32 ISC_Disable_Program signal: An optional signal used to indicate that program data writing is not allowed.

NOTE See 5.1.11.2.

3.1.33 ISC_Disable_Read signal: An optional signal used to indicate that program data reading is not allowed.

NOTE See 5.1.11.1.

3.1.34 ISC_Done signal: A signal that is used to indicate that a device program has been successfully written to the satisfaction of an external algorithm. When this signal is cleared, the device is considered to be unprogrammed, even if physically, the device does contain programmed bits. When this signal is set, the device is considered to be programmed and ready for operation. In volatile devices, this signal is automatically cleared at power-up. In nonvolatile devices, this signal maintains its state during nonpowered intervals. This signal can be set by programming operations described in this standard, and it may be protected by a security function implemented within the device.

3.1.35 ISC_Enabled signal: A signal that is used to indicate that a device has been successfully conditioned for programming activity. This signal is cleared during power-up for both volatile and nonvolatile devices.

NOTE See Clause 4.

3.1.36 ISC system pin: A system pin with an I/O function that is defined by configuration information programmed into the device, or has been designated as an ISC system pin by the device designer.

3.1.37 load: In the context of an instruction, an instruction is loaded when opcode bits present in the IEEE Std 1149.1-2001 instruction shift register are updated into the parallel-hold portion of the register by passing through the *Update-IR* TAP Controller state. At the time an instruction is loaded, its data register is selected for data shifting. *Contrast:* **execute**.

3.1.38 mission logic: *See:* **system logic**.

3.1.39 mission pin: *See: system pin.*

3.1.40 noninvasive instruction: Instructions from IEEE Std 1149.1-2001 (including BYPASS, SAMPLE, PRELOAD, IDCODE, and USERCODE) that maintain the device in system mode rather than in test mode. Thus, these instructions do not interfere with the system behavior of the device. *Contrast: pin-permission instruction.*

3.1.41 nontest instruction: Instructions that bring the device out of test mode and put it into unprogrammed, operational, or ISC accessed mode. These instructions can be, but are not limited to, any of the noninvasive instructions or any of the instructions defined in this standard. *Contrast: test instruction.*

3.1.42 nonvolatile program: Device programming that is retained by a device even when power is removed and later restored.

3.1.43 nullified: Proper ISC instruction operation is predicated on the device being in the ISC accessed system modal state while it is being executed. If this condition is not met, the instruction's operation is nullified. A nullified ISC instruction will still place a defined register in the shift path and shift bits through it as required by IEEE Std 1149.1-2001, but the ISC actions defined for the instruction by this standard will not be completed and the result of its execution is undefined.

NOTE See Clause 4.

3.1.44 optimize: ISC programming activities can be optimized for time efficiency. There are two forms of optimization. First, if a device has a specific instruction loaded, it may be redundant to reload this instruction. This is called "IR optimization." Second, when a register is targeted between TDI and TDO, "DR optimization" can be employed by shifting out captured data while new update data are being shifted in, rather than first shifting out captured data and then shifting in new update data. When a chain of devices is operated, optimization can be performed, but each device in the chain must be considered. For example, if two of three devices in subsequent activities will be executing the same instruction but the third will need a new instruction loaded, then IR optimization is not possible because loading the third device's instruction forces the reload of the first two.

3.1.45 pin-permission instruction: Instructions from IEEE Std 1149.1-2001 (including EXTEST, INTEST, CLAMP, HIGHZ, and RUNBIST) and instructions from this standard that place the device in test or programming modes. These instructions take over control of the system I/O pins. *Contrast: noninvasive instruction.*

3.1.46 program: An operation that installs externally supplied device programming bits (or a portion thereof) within a programmable device. This operation may be inhibited if the device contains a security feature enabled by a Protect operation.

3.1.47 programmable array logic (PAL): An earlier programmable logic technology still widely used, featuring a fixed OR array and programmable AND array.

3.1.48 programmable logic devices: This family of devices includes complex programmable logic device (CPLD), electrically erasable programmable read-only memory (EEPROM), erasable programmable logic device (EPLD), field-programmable gate array (FPGA), flash random access memory, and programmable array logic.

3.1.49 protect: Traditionally, an operation that prevents a subsequent read operation from being able to read out a device program. This is a security feature used to prevent copying or deciphering of a device program. In some implementations, writing one or more "security" bits located somewhere in the device program enables protection. Once set, these bits block subsequent read operations. Security bits are usually only

cleared by erasing the device program. More recently, this operation has been expanded to include the prevention of unwanted programming or erasure operations.

3.1.50 read: An operation that allows a device program to be read back out from a programmed device. This operation may be inhibited if the device contains a security feature enabled by a protect operation.

3.1.51 signal: A named signal is used to denote a binary condition used to choose between two outcomes. It is deliberately “implementation-vague” so as not to constrain a designer’s freedom. For example, a signal may be implemented as a bit in a register, or derived as a function of several bits, or as a one-hot decode from an instruction decoder, as fits the designer’s needs. This should not be confused with I/O signals on device pins that may be high, low, or disabled.

3.1.52 simple programmable logic device (SPLD): A family of programmable logic devices, including programmable array logic (PAL), generic array logic (GAL), and the erasable programmable logic device (EPLD).

3.1.53 static concurrentization: This process merges the programming algorithms prior to any operation being executed on the chain of devices. All decisions regarding optimization of the programming algorithms have been made before run time so that the run-time for the flow is completely predictable. *Contrast:* **dynamic concurrentization.**

3.1.54 symmetrically writeable device: A device with program memory that can be written to either state. These devices are often Static RAM-based. *Contrast:* **asymmetrically writeable device.**

3.1.55 system logic: Any item of logic that is dedicated to realizing the nontest function of the component or is at the time of interest configured to achieve some aspect of the nontest function. In the case of programmable devices, the system logic may not be defined until after the device is programmed. *See also:* **device program.** *Syn.:* **mission logic.**

3.1.56 system modal states: IEEE Std 1149.1-2001 defines two modes, “system” mode and “test” mode. Devices that conform to this standard have a system mode that is divided into four modal states. This reflects the fact that programmable devices, at various times in their operation, may (A) be in “Operational” mode containing system logic, may (B) be in an “Unprogrammed” mode with blank logic configuration memory, may (C) be in “ISC Accessed” mode for programming or accessing logic configuration memory, or may (D) be in “ISC Complete” mode when accessing configuration memory is complete before returning to the operational mode.

NOTE See Clause 4.

3.1.57 system pin: A component pin that feeds, or is fed from, the on-chip system logic. In the case of programmable devices, the system logic may not be defined until after the device is programmed. In devices that contain conventional logic in addition to programmable logic, system pins may be classified as ISC system pins and fixed system pins. *Syn.:* **mission pin.** *See also:* **device program.**

NOTE See 3.1.25 and 3.1.36.

3.1.58 TAP Controller: A 16-state finite state machine used to control the activities of the test access port and the test logic within the device. *See also:* **Test Access Port (TAP).**

3.1.59 test access port (TAP): Defined by IEEE Std 1149.1-2001. This consists of four (optionally five) device pins dedicated for use by IEEE Std 1149.1-2001 and programming usage defined by this standard. TAP pins are not system pins.

3.1.60 test instruction: Instructions that include any pin-permission instruction that puts the device into test mode but not into any of the programming modes.

3.1.61 verify: An operation similar to reading that compares an installed device program (or a portion thereof) to a program supplied externally. If a discrepancy is found, an error indication is given. This operation may be inhibited if the device contains a security feature enabled by a protect operation.

3.1.62 volatile program: Device programming that is lost when a device has power removed from it.

3.1.63 write: *See: program.*

3.2 Acronyms

The following acronyms are used within this standard.

BSDL	Boundary-Scan Description Language (see IEEE Std 1149.1-2001)
BSR	Boundary-Scan Register
CPLD	Complex Programmable Logic Device
EEPROM	Electrically Erasable Programmable Read-Only Memory
EPLD	Electrically Programmable Logic Device
EPROM	Erasable Programmable Read-Only Memory
FPGA	Field-Programmable Gate Array
GAL	Generic Array Logic
IC	Integrated Circuit
I/O	Input/Output
IOM	Input/Output Module
ISC	In-System Configuration
OST	Operation Success Test
PAL	Programmable Array Logic
PROM	Programmable Read-Only Memory
SRAM	Static Random Access Memory
TAP	IEEE Std 1149.1-2001 Test Access Port
TCK	The Test Clock input pin contained in the IEEE Std 1149.1-2001 TAP
TDI	The Test Data Input pin contained in the IEEE Std 1149.1-2001 TAP
TDO	The Test Data Output pin contained in the IEEE Std 1149.1-2001 TAP
TLR	Test-Logic-Reset, the reset state of the IEEE Std 1149.1-2001 TAP Controller state diagram
TMS	The Test Mode Select input pin contained in the IEEE Std 1149.1-2001 TAP
TRST*	The Test Reset input pin contained in the IEEE Std 1149.1-2001 TAP (this pin is optional)

NOTE Following the convention used in IEEE Std 1149.1-2001, pins or signals with a trailing asterisk (“*”) in their names are active low true signals.

4. System modal states

A conventional (i.e., nonprogrammable) device compliant with IEEE Std 1149.1-2001 has a “system” mode and a “test” mode. In system mode, the device performs its intended mission even while executing “noninvasive” nontest instructions. In test mode, test instructions control the device’s system I/O pins for testing purposes. A compliant IEEE Std 1149.1-2001 device passes from system mode to test mode when certain IEEE Std 1149.1-2001 instructions become active, as shown in Figure 2.

Devices that are compliant with IEEE Std 1149.1-2001 power-up in system mode. Noninvasive nontest IEEE Std 1149.1-2001 instructions include BYPASS, PRELOAD, SAMPLE, IDCODE, and USERCODE.

Annex D

(informative)

Bibliography

IEEE 100™, *The Authoritative Dictionary of IEEE Standards Terms*, Seventh Edition.⁵

⁵The IEEE standards or products referred to in Annex D are trademarks owned by the Institute of Electrical and Electronics Engineers, Incorporated.