

# IEEE Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture

IEEE Computer Society

| **STANDARDS**

Developed by the  
Test Technology Standards Committee

**IEEE Std 1149.7™-2022**  
(Revision of IEEE Std 1149.7™-2009)

# **IEEE Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture**

Developed by the

**Test Technology Standards Committee  
of the  
IEEE Computer Society**

Approved 16 June 2022

**IEEE SA Standards Board**

**Abstract:** Circuitry that may be added to an integrated circuit to provide access to on-chip Test Access Ports (TAPs) specified by IEEE Std 1149.1™ is described in this standard. The circuitry uses IEEE Std 1149.1 as its foundation, providing complete backward compatibility, while aggressively adding features to support test and applications debug. It defines six classes of IEEE 1149.7 Test Access Ports (TAP.7s), T0 to T5, with each class providing incremental capability, building on that of the lower level classes. Class T0 provides the behavior specified by 1149.1 from startup when there are multiple on-chip TAPs. Class T1 adds common debug functions and features to minimize power consumption. Class T2 adds operating modes that maximize scan performance. It also provides an optional hot-connection capability to prevent system corruption when a connection is made to a powered system. Class T3 supports operation in either a four-wire Series or Star Scan Topology. Class T4 provides for communication with either a two-pin or four-pin interface. The two-pin operation serializes IEEE 1149.1 transactions and provides for higher Test Clock rates. Class T5 adds the ability to perform data transfers concurrently with scan, supports utilization of functions other than scan, and provides control of TAP.7 pins to custom debug technologies in a manner that ensures current and future interoperability.

**Keywords:** 2-pin, 2-wire, 4-pin, 4-wire, Advanced Protocol, Advanced Protocol Unit, APU, Background Data Transfer, background data transport, BDX, boundary scan, BSDL, BSDL.1, BSDL.7, BYPASS, Capture-IR, CDX, Chip-Level TAP Controller, CID, Class T0, Class T1, Class T2, Class T3, Class T4, Class T5, CLTAPC, compact JTAG, compliant behavior, compliant operation, control level, controller address, Controller ID, Controller Identification Number, CP, Custom Data Transfer, custom data transport, Data Register, debug interface, debug logic, debug and test interface, DOT1, DOT7, DTI, DTS, DTT, Debug Test System, debug test target, Escape, EOT, EPU, extended operation, Extended Protocol, EXTEST, four-pin, four-wire, HSDL, HSDL.7, IDCODE, IEEE 1149.1™, IEEE 1149.7™, Instruction Register, JScan, JScan0, JScan1, JScan2, JScan3, JTAG, MScan, MTCP, Multi-TAP Control Path, narrow Star Scan Topology, nTRST, nTRST\_PD, optimized scan, OScan, OScan0, OScan1, OScan2, OScan3, OScan4, OScan5, OScan6, OScan7, Pause-DR, Pause-IR, PC0, PC1, RSU, Reset and selection unit, RTI, Run-Test/Idle, scan, scan DR, scan format, scan IR, Scan Packet, scan path, scan performance, scan protocol, scan topology, series, Series Branch, Series Scan, Series Scan Topology, Series-Equivalent Scan, Series Topology, Shift-DR, Shift-IR, SiP, Star Scan, Star Scan Topology, Star Topology, Star-2, Star-2 Branch, Star-2 Scan, Star-2, Scan Topology, Star-4, Star-4 Branch, Star-4 Scan, Star-4 Scan Topology, SP, SScan, SScan0, SScan1, SScan2, SScan3, stall, SSD, Scan Selection Directive, Standard Protocol, star scan, STL, System Test Logic, TAP, TAP controller, TAP controller address, TAP selection, TAP.1, TAP.7, TAPC, TCA, TCK, TCKC, TDI, TDIC, TDOC, TDOE, Test Access Port, test and debug, Test-Logic-Reset, TLR, TMSC, Transport Packet, T0, T0 TAP.7, T1, T1 TAP.7, T2, T2 TAP.7, T3, T3 TAP.7, T4, T4 TAP.7, T4(N), T4(N) TAP.7, T4(W), T4(W) TAP.7, T5, T5 TAP.7, T5(N), T5(N) TAP.7, T5(W), T5(W) TAP.7, TP, two-pin, two-wire, Update-DR, Update-IR, ZBS, zero bit scan

---

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

Copyright © 2022 by The Institute of Electrical and Electronics Engineers, Inc.  
All rights reserved. Published 14 October 2022. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-1-5044-8875-4 STD25536  
Print: ISBN 978-1-5044-8876-1 STDPD25536

*IEEE prohibits discrimination, harassment, and bullying.*  
For more information, visit <https://www.ieee.org/about/corporate/governance/p9-26.html>.  
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.

## **Important Notices and Disclaimers Concerning IEEE Standards Documents**

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>), appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

### **Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents**

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning this standard, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

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.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

### **Translations**

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE is the approved IEEE standard.

## Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter's views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s).

## Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents.**

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and subcommittees of the IEEE SA Board of Governors are not able to provide an instant response to comments, or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or in revisions to an IEEE standard is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#).<sup>a</sup> An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.<sup>b</sup>

## Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

## Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

## Copyrights

IEEE draft and approved standards are copyrighted by IEEE under US and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and

---

<sup>a</sup> Available at: <https://development.standards.ieee.org/myproject-web/public/view.html#landing>.

<sup>b</sup> Available at: <https://standards.ieee.org/content/ieee-standards/en/about/contact/index.html>.

adoption by public authorities and private users, neither IEEE nor its licensors waive any rights in copyright to the documents.

## Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

## Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, 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 order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit [IEEE Xplore](#) or [contact IEEE](#).<sup>c</sup> For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

## Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#).<sup>d</sup> Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

## Patents

IEEE Standards are developed in compliance with the [IEEE SA Patent Policy](#).<sup>e</sup>

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 by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

---

<sup>c</sup> Available at: <https://ieeexplore.ieee.org/browse/standards/collection/ieee>.

<sup>d</sup> Available at: <https://standards.ieee.org/standard/index.html>.

<sup>e</sup> Available at: <https://standards.ieee.org/about/sasb/patcom/materials.html>.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

## **IMPORTANT NOTICE**

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. IEEE Standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

## Participants

At the time this IEEE standard was completed, the P1149.7 Working Group had the following membership:

**Jason L. Peck, Chair**  
**Rolf Kühnis, Vice Chair**  
**Gary A. Cooper \*, Editor**  
**Ashwini Athalye, Secretary**

John Horley

Adam W. Ley

Bradford G. Van Treuren

\*Member Emeritus

The P1149.7 Working Group would like to acknowledge the work of the original Working Group responsible for IEEE Std 1149.7™-2009:

**Robert Oshana, Chair**  
**Bart Vermeulen, Vice Chair**  
**Gary Swoboda, Technical Architect and Principal Author**

ARM Ltd.  
ASSET InterTech, Inc.  
Freescale Semiconductor

Intel Corporation  
Lauterbach Datentechnik GmbH

Nokia Corporation  
NXP Semiconductors  
Texas Instruments

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

Ashwini Athalye  
Adam Cron  
Heiko Ehrenberg  
Peter Harrod  
Werner Hoelzl

Rolf Kühnis  
Adam W. Ley  
Ian McIntosh  
Nick S. A. Nikjoo  
Jim O'Reilly  
Jason L. Peck

Saghir Shaikh  
Walter Struppner  
David Thompson  
Lisa Ward  
Oren Yuen

When the IEEE SA Standards Board approved this standard on 16 June 2022, it had the following membership:

**David J. Law, Chair**  
**Ted Burse, Vice Chair**  
**Gary Hoffman, Past Chair**  
**Konstantinos Karachalios, Secretary**

Edward A. Addy  
Ramy Ahmed Fathy  
J. Travis Griffith  
Guido R. Hertz  
Yousef Kimiagar  
Joseph L. Koepfinger\*  
Thomas Koshy  
John D. Kulick

Johnny Daozhuang Lin  
Kevin Lu  
Daleep C. Mohla  
Andrew Myles  
Damir Novosel  
Annette D. Reilly  
Robby Robson  
Jon Walter Rosdahl

Mark Siira  
Dorothy V. Stanley  
Lei Wang  
F. Keith Waters  
Karl Weber  
Sha Wei  
Philip B. Winston  
Daidi Zhong

\*Member Emeritus

## Introduction

This introduction is not part of IEEE Std 1149.7-2022, IEEE Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture.

This standard defines a debug and Test Access Port that provides both compatibility with IEEE Std 1149.1 and operation with as few as two pins. It complements IEEE Std 1149.1 and is not intended to replace it. The facilities defined by the standard seek to provide a cost-effective debug and test interface solution supporting the debug and test of applications. The standard supports the needs for application debug of products with complex digital circuitry, applications software, and one or more central processing units (CPUs). The interface also provides a means to instrument these applications concurrently with scan transactions associated with the underlying IEEE 1149.1 foundation.

## History of the development of this standard

The process of developing this standard began in 2004 when the Mobile Industry Processor Interface (MIPI) Alliance Test and Debug Working Group was formed. During 2004, a standard that addressed the needs of both test and debug was described and requirements were gathered. The emphasis was on creating an interface that allowed standardization of access to debug and test capabilities on-chip while addressing the needs of applications developers sharing an integrated circuit's Test Access Port with test uses. This interface also targeted making chips exhibit the behavior that is specified by IEEE Std 1149.1 when multiple Test Access Ports are deployed on a single chip. It was thought that applications development tools vendors would be a big beneficiary of this interface as common connectors providing access to the debug and test interface would be recommended. It was felt that all of this would best be accomplished with an interface that was compatible with chip intellectual property already utilizing IEEE Std 1149.1.

Once the requirements gathering process was completed in 2004, the Test and Debug Working Group turned its attention to proposals for addressing the above requirements. Both competing and complementary proposals were submitted. The proposals were debated, and one was selected as a basis to move forward in April 2005. A preliminary specification replaced the proposal and was reviewed inside the MIPI Test and Debug Working Group and by the NEXUS standards body in late 2005. Since both bodies felt the applicability of the specification was well beyond their own objectives, they both recommended using the on-chip portions of those specifications as the basis for an IEEE standard in a joint meeting in December 2005.

Multiple drafts of the specification were completed and reviewed from 2005 until the standard's approval, with the capability provided by the standard expanded along the way. Revisions in the presentation and number of capabilities took place along the way to the standard's unanimous approval in 2009.

As the standard was scheduled to become inactive in 2019, some of the original Working Group members were queried and there was a general consensus that the effort should be made for revision per maintenance policy for IEEE standards. A new PAR was created, and a new Working Group was formed.

No major additions or changes to the standard were targeted as part of the renewal effort. As such, this is not a major update and the revised version is fully backward compatible with the original. However, the Working Group did agree to evaluate feedback from customers of the original version in an effort to clarify certain issues and fix potential errors.

## **Changes introduced by this revision**

After reviewing the feedback against the original version, the Working Group agreed to make the following two changes to help clarify the technical content of the standard:

- Subclause 23.7 was updated to include an explicit Specification section (23.7.2). The normative material contained here existed in the original standard but was not called out using the formalized method outlined in 1.12.2.
- Subclause F.6 was added to Annex F in an effort to clarify the design issues associated with how the K-bias requirements on TMS(C) can impact system performance. Several system design considerations are outlined that can help minimize the impact of this problem.

In addition, several updates were made to address editorial and presentation issues. The major ones include the following:

- Added 1.3.
- Updated several figures to address issues related to appearance, rendering, and avoiding the use of color to convey meaning.
- Eliminated the use of dated references to IEEE standards.
- Updated BSDL usage within document to IEEE Std 1149.1-2013.
- Introduced initiator/responder as a culturally neutral alternative to describe entity relationships previously presented in disfavored terms.

## Contents

1. Overview .....	56
1.1 Scope .....	56
1.2 Purpose .....	56
1.3 Word usage .....	56
1.4 Contrasting IEEE Std 1149.1 and this standard .....	57
1.5 Challenges .....	58
1.6 Important considerations.....	59
1.7 Nomenclature.....	59
1.7.1 References to technology and standards .....	59
1.7.2 Describing Test Access Port behaviors.....	59
1.7.3 Describing TAPs and TAP controllers.....	60
1.7.4 Describing scan exchanges .....	61
1.7.5 Describing TAP signals .....	61
1.8 Ensuring transparency to IEEE 1149.1 intellectual property .....	62
1.8.1 IEEE 1149.1 constraints .....	62
1.8.2 IEEE 1149.1 constraints/requirements balancing.....	62
1.8.3 IEEE 1149.1 upgrade path.....	62
1.9 Maximizing compatibility with 1149.1 IP .....	63
1.9.1 IEEE 1149.1 infrastructure .....	63
1.9.1.1 IEEE 1149.1 instructions.....	63
1.9.1.2 IEEE 1149.1 Scan Paths.....	63
1.9.1.3 IEEE 1149.1 boundary-scan capability .....	63
1.9.2 Test Clock treatment .....	63
1.9.2.1 Test Clock source .....	63
1.9.2.2 Test Clock shared or dedicated use .....	64
1.9.2.3 Unorthodox use of Test Clock.....	64
1.10 Scalability .....	65
1.10.1 Types of operation/capability classes.....	65
1.10.2 Signaling .....	67
1.11 Flexibility.....	67
1.11.1 Supporting various scan topologies .....	67
1.11.2 Operation with more than one scan topology/other technologies .....	69
1.12 Document content .....	69
1.12.1 Descriptive material .....	70
1.12.2 Specification material .....	70
1.13 Document organization.....	70
1.13.1 Partitioning.....	70
1.13.1.1 Foundation.....	71
1.13.1.2 TAP.7 Classes .....	72
1.13.1.3 Test description languages .....	72
1.13.1.4 Annexes.....	72
1.13.2 Pictorial view .....	73
1.14 Using the standard .....	74
1.14.1 User background knowledge.....	74
1.14.2 Types of users .....	74
1.14.2.1 Chip designer .....	74
1.14.2.2 Programmer.....	74
1.14.3 Use summary .....	74
1.15 Conventions .....	75
2. Normative references.....	81
3. Definitions, acronyms, and abbreviations .....	82

3.1 Definitions .....	82
3.2 Acronyms and abbreviations .....	86
4. TAP.7 concepts and architecture .....	91
4.1 Introduction .....	91
4.2 Concepts supporting system architecture.....	91
4.2.1 Maximizing compatibility with IEEE Std 1149.1 .....	91
4.2.1.1 Hardware components.....	92
4.2.1.2 Software components .....	92
4.2.2 TAPC hierarchy .....	92
4.2.2.1 Overview .....	92
4.2.2.2 Hierarchy levels .....	93
4.2.3 Parking the TAPC state.....	94
4.2.3.1 Overview .....	94
4.2.3.2 Parking-state relationships .....	95
4.2.3.3 Parking-state terminology .....	96
4.2.3.3.1 ADTAPC selection states .....	96
4.2.3.3.2 CLTAPC selection states.....	96
4.2.3.3.3 EMTAPC selection states.....	97
4.2.3.3.4 Selection state summary .....	98
4.2.4 Choice of parking states .....	98
4.2.5 Parking methods.....	98
4.2.6 Operation of the TAP.7 Controller.....	98
4.2.6.1 TAP.7 Controller management.....	98
4.2.6.2 System and Control Paths .....	99
4.2.6.3 Power management .....	100
4.2.7 Scan topologies .....	101
4.2.7.1 Series and Star Scan Topologies .....	101
4.2.7.2 Scan Topology Training.....	102
4.2.7.3 Sharing of signaling with other technologies .....	102
4.2.7.4 Direct addressability for Star Scan Topologies .....	102
4.2.7.5 Series-Equivalent Scans for Star Scan Topologies.....	103
4.2.7.6 Output-drive characteristics .....	104
4.2.7.7 Scan formats.....	105
4.2.7.8 Interoperability .....	106
4.3 Concepts supporting pin efficiency .....	106
4.3.1 Signaling methods.....	106
4.3.2 Protocols .....	107
4.3.2.1 Protocol types.....	107
4.3.2.2 Standard Protocol .....	108
4.3.2.3 Advanced Protocol .....	108
4.3.2.3.1 Interleaving of scan and non-scan information.....	109
4.3.2.3.2 Serialization of scan information.....	110
4.3.2.4 Control Protocol .....	111
4.3.3 Advanced and Control Protocol characteristics .....	111
4.3.3.1 Packets, bit-frames, and bits.....	111
4.3.3.2 Data and control information .....	112
4.3.4 Performance .....	113
4.3.4.1 TAP operation .....	113
4.3.4.1.1 Signal timing .....	113
4.3.4.1.2 Registering of signals .....	114
4.3.4.2 Scan transfer efficiency .....	114
4.4 Concepts supporting capability.....	114
4.4.1 Concepts already described.....	114
4.4.2 Resets .....	115
4.4.3 Private commands and registers.....	115
4.5 IEEE 1149.7 architecture.....	115

4.5.1 Components .....	115
4.5.2 Reset types .....	118
4.5.3 Start-up options .....	119
4.6 Operating models.....	120
4.6.1 Model types.....	120
4.6.2 Standard models.....	120
4.6.3 Advanced models.....	122
5. T0–T3 TAP.7 operational overview .....	123
5.1 Introduction .....	123
5.2 T0 TAP.7 .....	123
5.2.1 Overview.....	123
5.2.2 Operating modes and capabilities .....	124
5.2.3 Operation .....	125
5.2.3.1 Multi-TAPC architecture.....	125
5.2.3.2 Selection and deselection of EMTAPCs .....	125
5.2.4 T0 TAP.7 high-level block diagram.....	126
5.3 T1 TAP.7 .....	127
5.3.1 Overview.....	127
5.3.2 Operating modes and capabilities .....	128
5.3.3 Operation .....	128
5.3.3.1 Adding TAP.7 functionality to the <i>BYPASS</i> and <i>IDCODE</i> instructions .....	128
5.3.3.2 Zero-bit DR scans.....	128
5.3.3.3 Utilizing ZBSs for TAP.7 Controller functionality .....	129
5.3.3.3.1 Locking the ZBS count.....	129
5.3.3.3.2 Control levels .....	129
5.3.3.3.3 Exiting a control level .....	130
5.3.3.3.4 Zeroing the ZBS count .....	130
5.3.3.4 EPU Operating States.....	130
5.3.3.5 Commands.....	130
5.3.3.6 Registers.....	133
5.3.3.7 Using the System and Control Paths .....	135
5.3.3.7.1 System Path .....	136
5.3.3.7.2 Control Path.....	136
5.3.3.7.3 TDO Drive Policy.....	137
5.3.3.8 EPU groups .....	137
5.3.4 T1 TAP.7 high-level block diagram.....	137
5.4 T2 TAP.7 .....	137
5.4.1 Overview.....	137
5.4.2 Operating modes and capabilities .....	139
5.4.3 Operation .....	140
5.4.3.1 Overview .....	140
5.4.3.2 Using the System and Control Paths .....	141
5.4.3.3 TDO Drive Policy .....	142
5.4.3.4 STL groups.....	142
5.4.3.4.1 STL group types .....	142
5.4.3.4.2 STL Group Membership changes.....	143
5.4.4 T2 TAP.7 high-level block diagram.....	143
5.5 T3 TAP.7 .....	144
5.5.1 Overview.....	144
5.5.2 Operating modes and capabilities .....	145
5.5.3 Operation .....	147
5.5.3.1 Within series and star scan topologies.....	147
5.5.3.2 T3 TAP.7 Controller addressability in a Star-4 Scan Topology .....	147
5.5.3.3 Pause-IR and Pause-DR STL groups .....	147
5.5.3.4 Series/star scan equivalency .....	148
5.5.3.4.1 Defining scan equivalency.....	148

5.5.3.4.2 Creating a Series-Equivalent Scan within a Star Scan Topology .....	148
5.5.3.5 Scan Selection Directives.....	149
5.5.3.5.1 Enabling the use of SSDs .....	149
5.5.3.5.2 Types of SSDs .....	149
5.5.3.5.3 SSD execution .....	150
5.5.3.5.4 SSD State Machine.....	150
5.5.3.6 Series-Equivalent Scan creation.....	151
5.5.3.6.1 Exclusivity of SSD and TAP.7 Controller commands.....	152
5.5.3.7 Using the System and Control Paths .....	153
5.5.3.8 TDO Drive Policy .....	154
5.5.4 T3 TAP.7 high-level block diagram.....	154
6. T4–T5 TAP.7 operational overview .....	156
6.1 Introduction .....	156
6.2 T4 TAP.7 .....	157
6.2.1 Operating modes and capabilities .....	157
6.2.2 Operation .....	158
6.2.2.1 Signal behaviors.....	158
6.2.2.2 Rising and falling TMSC input sampling.....	158
6.2.2.3 Controller addressability in a Star-2 Scan Topology.....	159
6.2.2.4 RSU and APU functions .....	159
6.2.2.4.1 Bypass ( <i>BPA</i> ) .....	161
6.2.2.4.2 Check Process Active ( <i>CPA</i> ).....	161
6.2.2.4.3 Scan Packet Active ( <i>SPA</i> ).....	164
6.2.3 T4 TAP.7 high-level block diagram.....	165
6.3 T5 TAP.7 .....	166
6.3.1 Overview.....	166
6.3.2 Operating modes and capabilities .....	168
6.3.2.1 Transport source/destinations.....	168
6.3.2.1.1 Single-client operation.....	168
6.3.2.1.2 Multi-client operation .....	169
6.3.2.1.3 Client-to-client operation.....	170
6.3.2.2 Transfer characteristics .....	171
6.3.3 Operation .....	172
6.3.3.1 Transport Control Function.....	172
6.3.3.2 <i>TPA</i> .....	173
6.3.4 T5 TAP.7 high-level block diagram.....	174
6.4 TAP.7 feature summary.....	175
7. System concepts .....	177
7.1 Introduction .....	177
7.2 Key system attributes.....	177
7.3 DTS/TS connectivity with a mix of technologies.....	177
7.3.1 Technology mixes .....	177
7.3.2 Technology branches .....	178
7.4 TAP.7 deployment scenarios .....	179
7.4.1 TAP.1 Series Branches .....	179
7.4.2 TAP.7 Series, Star-4, and Star-2 Branches .....	179
7.5 Chip TAPC hierarchy .....	180
7.6 Combined view of TAP connectivity and TAPC hierarchy.....	181
7.7 Chips, components, and boards .....	182
8. TAPC hierarchy.....	184
8.1 Introduction .....	184
8.2 Selection/deselection with the TAPC hierarchy .....	184
8.2.1 Selection choices.....	184
8.2.2 Selection/deselection/class relationships.....	185
8.2.3 TAPC parent/child relationships .....	185
8.3 TAPC selection/deselection characteristics .....	185

8.3.1 TAPC and scan path behavior.....	185
8.3.2 Selection/deselection mechanisms .....	186
8.3.3 Parking states and resynchronization .....	186
8.4 ADTAPC selection/deselection .....	187
8.4.1 Parking use cases .....	187
8.4.2 DTS/ADTAPC relationship .....	187
8.4.3 ADTAPC operation .....	188
8.5 CLTAPC selection/deselection .....	189
8.5.1 Parking use cases .....	189
8.5.2 ADTAPC/CLTAPC relationship .....	189
8.5.3 CLTAPC operation .....	189
8.6 EMTAPC selection/deselection .....	191
8.6.1 Parking use cases .....	191
8.6.2 CLTAPC/EMTAPC relationship .....	191
8.6.3 EMTAPC operation .....	191
8.7 Using a common selection/deselection protocol across technologies.....	192
8.8 RSU deployment.....	192
8.8.1 Use with new or existing IP .....	192
8.8.2 Using TAP pins for multiple functions .....	193
8.9 Using the TAPC hierarchy.....	193
8.9.1 Selection considerations.....	193
8.9.2 Start-up considerations.....	194
8.10 Test/debug applications and the TAPC hierarchy.....	194
8.10.1 Debug use of the TAPC hierarchy .....	194
8.10.2 Test use of the TAPC hierarchy .....	195
9. Registers, commands, and scan paths.....	197
9.1 Introduction .....	197
9.2 Command basics.....	197
9.3 Register portfolio .....	199
9.3.1 Description .....	199
9.3.1.1 Global and Local Registers .....	199
9.3.1.2 Register loads .....	199
9.3.1.3 Register reset values.....	200
9.3.2 Specifications.....	200
9.4 Command portfolio.....	202
9.4.1 Description .....	202
9.4.1.1 Command types.....	202
9.4.1.2 Store commands .....	202
9.4.1.3 Select commands.....	203
9.4.1.4 Scan commands.....	203
9.4.1.5 Enumerate commands .....	203
9.4.1.6 Private commands .....	203
9.4.1.7 Effects a TAP.7 Controller reset .....	203
9.4.2 Specifications.....	204
9.5 Representation of commands in examples.....	209
9.6 Global and Local Register programming with commands .....	209
9.7 Scan paths .....	210
9.7.1 Conceptual and physical views .....	210
9.7.1.1 Description .....	210
9.7.1.1.1 Path characteristics .....	210
9.7.1.1.2 Conceptual path selection .....	210
9.7.1.1.3 Physical path selection.....	211
9.7.1.2 Specifications .....	212
9.7.2 EPU Scan Paths and their selection .....	214
9.7.2.1 Description .....	214
9.7.2.2 Specifications .....	214

9.7.3 EPU Scan Path characteristics .....	215
9.7.3.1 Description .....	215
9.7.3.1.1 Scan-path continuity .....	215
9.7.3.1.2 EPU Bypass-Path characteristics .....	216
9.7.3.1.3 EPU Bit-Path characteristics .....	216
9.7.3.1.4 EPU String-Path characteristics.....	216
9.7.3.1.5 Enumerate-Path characteristics.....	219
9.7.3.1.6 Auxiliary Path.....	219
9.7.3.2 Specifications .....	219
9.8 Two-part commands .....	221
9.9 Three-part commands .....	221
9.9.1 SCNB Command characteristics.....	221
9.9.2 SCNS Command characteristics .....	222
9.9.3 CIDA Command characteristics .....	223
9.10 RDBACKx and CNFGx Registers.....	225
9.10.1 RDBACKx Registers .....	226
9.10.1.1 Description .....	226
9.10.1.2 Specifications .....	226
9.10.2 CNFGx Registers .....	227
9.10.2.1 Description .....	227
9.10.2.1.1 Overview .....	227
9.10.2.1.2 CNFG0 mandatory configuration information .....	228
9.10.2.1.3 CNFG0 optional configuration information .....	228
9.10.2.1.4 CNFG1 optional configuration information .....	229
9.10.2.1.5 CNFG2 and CNFG3 Registers .....	229
9.10.2.1.6 Determining the TAP type and class using configuration information.....	229
9.10.2.2 Specifications .....	229
9.11 An approach to implementing command processing and scan paths.....	231
10. RSU ancillary services .....	235
10.1 Introduction .....	235
10.2 Resets.....	235
10.2.1 Description .....	235
10.2.1.1 Overview .....	235
10.2.1.2 Reset considerations.....	236
10.2.1.3 Reset effects .....	236
10.2.1.3.1 Type-5 Reset.....	237
10.2.1.3.2 Type-4 Reset.....	237
10.2.1.3.3 Type-3 Reset.....	237
10.2.1.3.4 Type-2 Reset.....	238
10.2.1.3.5 Type-1 Reset.....	238
10.2.1.3.6 Type-0 Reset.....	238
10.2.1.3.7 Type-0 versus a Type-2 Reset .....	238
10.2.1.4 TAP.7 Controller operation is ensured after power-up.....	238
10.2.1.5 Other effects of a TAP.7 Controller reset.....	238
10.2.1.6 An approach to implementing TAP.7 Controller resets .....	239
10.2.2 Specifications .....	240
10.3 Start-up options.....	242
10.3.1 Description .....	242
10.3.1.1 Overview .....	242
10.3.1.2 1149.1-compliant behavior start-up option .....	243
10.3.1.3 1149.1-Compatible Start-up option .....	243
10.3.1.4 IEEE 1149.1-Protocol Compatible .....	244
10.3.1.5 Offline-at-Start-up option.....	244
10.3.1.6 Start-up behavior .....	245
10.3.2 Specifications .....	246
10.4 Escape Detection .....	250

10.4.1 Description .....	250
10.4.1.1 Overview .....	250
10.4.1.2 Detection .....	251
10.4.1.2.1 Custom Escape .....	252
10.4.1.2.2 Selection and Deselection Escapes .....	252
10.4.1.3 Reset Escape.....	253
10.4.1.4 Timing considerations .....	253
10.4.1.5 An approach to implementing Escape Detection .....	253
10.4.2 Specifications.....	255
10.5 Selection Alert .....	256
10.5.1 Description .....	256
10.5.1.1 Overview .....	256
10.5.1.2 Selection Alert Bit Sequence.....	257
10.5.1.3 Selection Alert detection .....	258
10.5.1.4 An approach to implementing Selection Alerts.....	258
10.5.2 Specifications .....	260
10.6 Deselection Alert .....	262
10.6.1 Description .....	262
10.6.2 Specifications .....	262
10.7 Programming considerations .....	263
10.7.1 Resets .....	263
10.7.2 Escapes.....	263
10.7.3 Selection Alerts.....	263
10.7.4 Test and debug .....	263
10.7.5 Concurrent use of a Selection Escape and Selection Alert.....	263
10.8 ADTAPC State Machine .....	264
10.8.1 Need.....	264
10.8.2 An approach to implementing the ADTAPC .....	264
11. RSU Online/Offline capability .....	266
11.1 Introduction .....	266
11.2 Managing Online/Offline operation.....	266
11.3 Online/Offline operating principles .....	267
11.3.1 Conceptual view of Online/Offline operation.....	267
11.3.2 Events affecting Online/Offline operation .....	268
11.3.3 Summary of responses to selection/deselection events .....	269
11.3.4 Interoperability with other technologies .....	269
11.4 Initiating Offline operation .....	271
11.4.1 Description .....	271
11.4.1.1 Events initiating Offline operation .....	271
11.4.1.2 Escapes .....	271
11.4.1.3 Alerts .....	272
11.4.1.4 Use of an unsupported feature.....	272
11.4.1.5 Offline-at-Start-up .....	272
11.4.2 Specifications .....	272
11.5 Initiating Online operation.....	272
11.5.1 Description .....	272
11.5.2 Specifications .....	273
11.6 Context-sensitive response to Selection and Deselection Escapes .....	273
11.6.1 Description .....	273
11.6.1.1 Escape qualification criteria during Online operation.....	274
11.6.1.2 Escape qualification criteria during Offline-at-Start-up operation.....	274
11.6.1.3 Selection Alert during Offline-at-Start-up operation.....	275
11.6.2 Specifications .....	275
11.7 Selection Sequence .....	276
11.7.1 Initiation.....	276
11.7.2 Format .....	276

11.7.3 Technology-independent portion .....	276
11.7.4 Technology-dependent portion .....	277
11.7.5 Forms of Selection Sequence .....	277
11.7.6 Online Activation Code .....	278
11.7.6.1 Description .....	278
11.7.6.2 Specifications .....	279
11.7.7 TAP.7 Extension Code.....	281
11.7.7.1 Description .....	281
11.7.7.2 Specifications .....	282
11.7.8 Global Register load.....	283
11.7.8.1 Description .....	283
11.7.8.2 Specifications .....	284
11.7.9 Check Packet.....	284
11.7.9.1 Description .....	284
11.7.9.1.1 Format .....	284
11.7.9.1.2 Function.....	285
11.7.9.1.3 Directives.....	286
11.7.9.2 Specifications .....	286
11.8 Parking-state considerations .....	287
11.8.1 Description .....	287
11.8.2 Specifications.....	287
11.9 Control State Machine .....	290
11.9.1 Mandatory and optional behaviors.....	290
11.9.1.1 Description .....	290
11.9.1.2 Specifications .....	291
11.9.2 Standard state ( <i>STD</i> ) .....	292
11.9.2.1 Description .....	292
11.9.2.2 Specifications .....	292
11.9.3 Advanced state ( <i>ADV</i> ).....	293
11.9.3.1 Description .....	293
11.9.3.2 Specifications .....	293
11.9.4 Offline waiting state ( <i>OLW</i> ).....	294
11.9.4.1 Description .....	294
11.9.4.2 Specifications .....	294
11.9.5 Test state ( <i>TEST</i> ).....	295
11.9.5.1 Description .....	295
11.9.5.1.1 Selection test.....	295
11.9.5.1.2 Factors requiring a Global Register load for placement Online .....	296
11.9.5.1.3 ADTAPC resynchronization.....	296
11.9.5.1.4 Priority of conditions causing state changes.....	297
11.9.5.1.5 Test state function.....	297
11.9.5.1.6 Selection Sequences requiring a state load.....	298
11.9.5.2 Specifications .....	299
11.9.6 Check Packet state ( <i>CHK</i> ).....	302
11.9.6.1 Description .....	302
11.9.6.1.1 <i>CHK</i> substates .....	302
11.9.6.1.2 CP examples .....	303
11.9.6.2 Specifications .....	305
11.9.7 Offline-at-Start-up state ( <i>OLS</i> ).....	308
11.9.7.1 Description .....	308
11.9.7.1.1 Placement Online.....	308
11.9.7.1.2 CLTAPC state initialization .....	308
11.9.7.1.3 Selection Escape qualification in the <i>OLS</i> state.....	309
11.9.7.1.4 Example of exiting the <i>OLS</i> state .....	309
11.9.7.2 Specifications .....	310
11.9.7.3 An approach to implementing the CSM .....	311

11.10 Programming considerations .....	313
11.10.1 Escapes.....	313
11.10.1.1 Selection Escape.....	313
11.10.1.2 Deselection Escape.....	313
11.10.1.3 Reset Escape.....	314
11.10.2 Alerts.....	314
11.10.2.1 Deselection Alert.....	314
11.10.2.2 Selection Alert.....	314
11.10.2.3 Offline-at-Start-up .....	314
11.10.3 Selection Sequences .....	315
11.10.3.1 DTS/TAP.7 Controller synchronization.....	315
11.10.3.2 Short Form .....	315
11.10.3.3 Long Form.....	315
11.10.4 Hang caused by a programming error.....	316
12. TAP signals .....	317
12.1 Introduction .....	317
12.2 TAP.7 Class/signal relationships .....	317
12.2.1 Description.....	317
12.2.2 Specifications.....	318
12.3 Signal function and bias.....	319
12.3.1 Description.....	319
12.3.2 Specifications.....	320
12.4 Test Reset (nTRST and nTRST_PD) signals.....	321
12.4.1 Description.....	321
12.4.2 Specifications.....	322
12.5 TAP.7 signal functions with corresponding IEEE 1149.1 names .....	322
12.5.1 Description.....	322
12.5.2 Specifications.....	322
12.6 Test Clock (TCK) .....	322
12.6.1 Description.....	322
12.6.2 Specifications.....	322
12.7 Test Mode Select (TMS/TMSC).....	323
12.7.1 Description.....	323
12.7.1.1 Online start-up.....	323
12.7.1.2 Offline start-up .....	324
12.7.1.3 Combined view of Online and Offline-at-Start-up TMS(C) signal behaviors .....	326
12.7.2 Specifications.....	329
12.8 Test Data Input (TDI/TDIC).....	330
12.8.1 Description.....	330
12.8.2 Specifications.....	331
12.9 Test Data Output (TDO/TDOC) .....	333
12.9.1 Description.....	333
12.9.2 Specifications.....	334
12.10 Offline-at-Start-up behavior.....	335
12.10.1 Description.....	335
12.10.2 Specifications .....	336
12.11 TAP connections.....	336
12.11.1 Description.....	336
12.11.2 Specifications .....	336
12.12 Applicability of this standard.....	337
12.12.1 Description.....	337
12.12.2 Specifications .....	337
12.13 Recommendations for interoperability.....	338
12.13.1 Overview.....	338
12.13.2 Power-up behavior .....	338
12.13.3 IEEE 1149.7-Non-disruptive behavior.....	338

12.13.3.1 Description .....	338
12.13.3.2 Specifications .....	339
12.13.4 IEEE 1149.7-Other Behavior .....	339
12.13.4.1 Description .....	339
12.13.4.2 Specifications .....	339
13. TDO(C) Signal Drive Policy .....	341
13.1 Introduction .....	341
13.2 TDO(C) Signal Drive Types .....	341
13.2.1 TDO(C) Signal Drive Types .....	341
13.2.1.1 Single Drive .....	342
13.2.1.2 Joint Drive .....	342
13.2.1.3 Voting Drive .....	342
13.2.1.4 Inhibited Drive .....	342
13.2.2 Wire-ANDed TDOC data created with a combination of drives .....	342
13.3 Factors affecting the TDO(C) Drive Policy .....	343
13.4 TDO(C) Drive Policy template .....	344
13.4.1 General characteristics .....	344
13.4.2 TDOC drive enables .....	344
13.4.3 TDO(C) Drive Policy components .....	344
13.4.4 TAP.7 Class/TDO(C) Drive Policy component applicability .....	345
13.4.5 Dormant TDO(C) Drive Policy .....	345
13.4.6 Transition TDO(C) Drive Policy .....	345
13.4.7 Series TDO(C) Drive Policy components .....	345
13.4.7.1 Series System .....	345
13.4.7.2 Series Command .....	346
13.4.7.3 Series Control Level .....	346
13.4.8 Star-4 TDO(C) Drive Policies .....	346
13.4.8.1 Star-4 System .....	346
13.4.8.2 Star-4 Command .....	346
13.4.8.3 Star-4 control level .....	347
13.4.9 Hierarchical and flat views of the TDO(C) Drive Policy .....	347
13.4.10 Conceptual diagram of the TDO(C) Drive Policy .....	350
13.5 T0 TAP.7 TDOC Drive Policy .....	351
13.5.1 Description .....	351
13.5.2 Specifications .....	351
13.6 T1 and T2 TAP.7 TDOC Drive Policy .....	352
13.6.1 Description .....	352
13.6.2 Specifications .....	353
13.7 T3 and above TAP.7 TDOC Drive Policy .....	354
13.7.1 Description .....	354
13.7.2 Specifications .....	355
13.8 STL Group Membership .....	357
13.8.1 Tracking the Group Membership of STLs .....	357
13.8.2 STL Group Membership changes .....	357
13.8.2.1 Group membership changes with the <i>Test-Logic-Reset</i> state .....	357
13.8.2.2 Group membership changes with the <i>Run-Test/Idle</i> state .....	357
13.8.2.3 Group membership changes with the <i>Pause-IR</i> state .....	358
13.8.2.4 Group membership changes with the <i>Pause-DR</i> state .....	358
13.8.3 Commands/SSDs affecting group Scan Group Candidacy and Membership .....	358
13.8.3.1 Commands affecting Scan Group Candidacy .....	358
13.8.3.2 SSDs affecting Scan Group Candidacy and Membership .....	359
13.8.3.2.1 SSDs associated with the <i>Run-Test/Idle</i> state .....	359
13.8.3.2.2 SSDs associated with the <i>Pause-IR</i> state .....	359
13.8.3.2.3 SSDs associated with the <i>Pause-DR</i> state .....	359
13.8.4 Only Scan Group Member determination .....	360
13.8.4.1 Criteria .....	360

13.8.4.2 Method and information used to make determination .....	360
13.8.4.2.1 Idle Group Membership and membership counts .....	360
13.8.4.2.2 Pause-xR Group Membership and membership counts .....	361
13.8.4.2.3 Scan Group Membership and membership counts .....	361
13.8.4.2.4 Information recorded .....	362
13.8.5 STL group candidate and membership counts .....	362
13.8.5.1 Description .....	362
13.8.5.1.1 Scan Group Candidate Count (SGCC) .....	362
13.8.5.1.2 Potential Scan Group Membership Count Last .....	364
13.8.5.1.3 Factors creating SGCC and PSGMCL ambiguity .....	365
13.8.5.2 Specifications .....	365
13.8.6 Scan Group Membership Count Last determination .....	368
13.8.6.1 Description .....	368
13.8.6.2 Specification .....	368
13.8.7 Only Scan Group Member Last determination .....	369
13.8.7.1 Description .....	369
13.8.7.2 Specification .....	369
13.9 EPU Group Membership .....	371
13.9.1 Description .....	371
13.9.1.1 Tracking the EPU's Group Membership .....	371
13.9.1.2 Conditional Group Member Count .....	371
13.9.1.3 Only Conditional Group Member determination .....	372
13.9.2 Specifications .....	373
13.10 Drive Policy summary .....	375
13.11 An approach to implementing TDOC Drive Policy .....	376
13.11.1 Policy generation .....	376
13.11.2 Potential Scan Group Member Last .....	377
13.11.3 The SGCC and PSGMCL functions .....	377
13.11.4 Determining Scan Group Only Member Last/Membership Count Last .....	378
13.11.5 The CGMC function .....	379
13.12 Programming considerations .....	379
14. TMS(C) Signal Drive Policy .....	380
14.1 Introduction .....	380
14.2 TMS(C) output bit types .....	380
14.2.1 Scan Packet content .....	380
14.2.2 Transport Packet content .....	381
14.2.3 Drive relationship with TCKC .....	382
14.3 Drive policy by output bit type .....	383
14.4 TMSC Signal Drive Types .....	384
14.4.1 TMSC Signal Drive Types .....	384
14.4.1.1 Single Drive .....	385
14.4.1.2 Joint Drive .....	385
14.4.1.3 Voting Drive .....	385
14.4.1.4 Inhibited Drive .....	385
14.4.2 Wire-ANDed TMSC signal values .....	385
14.5 Dormant Bit Drive Policy .....	386
14.5.1 Description .....	386
14.5.2 Specifications .....	386
14.6 Precharge Bit Drive Policy .....	386
14.6.1 Description .....	386
14.6.2 Specifications .....	386
14.7 RDY Bit Drive Policy .....	387
14.7.1 Description .....	387
14.7.1.1 Characteristics .....	387
14.7.1.2 Policy details .....	387
14.7.1.3 Relationship to CLTAPC selection state changes .....	388

14.7.2 Specifications .....	388
14.8 TDO Bit Drive Policy .....	390
14.8.1 Description .....	390
14.8.1.1 Characteristics .....	390
14.8.1.1.1 Policy details for System Path .....	390
14.8.1.1.2 Policy details for Control Path.....	391
14.8.1.2 Correlation to TDO(C) Drive Policy .....	391
14.8.1.3 Combined TDO bit drive summary .....	392
14.8.2 Specifications .....	393
14.9 Transport Bit Drive Policy.....	394
14.9.1 Description .....	394
14.9.2 Specifications .....	395
14.10 An approach to implementing TMSC Drive Policy.....	395
14.11 Programming considerations .....	399
15. IEEE 1149.1-compliance concepts.....	401
15.1 Introduction .....	401
15.2 Background.....	401
15.3 Test and debug views of a system of interest.....	402
15.4 An approach to implementing EMTAPC selection/deselection .....	403
16. T0 TAP <sub>7</sub> .....	404
16.1 Introduction .....	404
16.2 Deployment .....	404
16.3 Capabilities .....	405
16.4 Configurations .....	405
16.4.1 Description .....	405
16.4.2 Specifications .....	405
16.5 Start-up behavior .....	406
16.5.1 Description.....	406
16.5.2 Specifications .....	406
16.6 Supporting multiple on-chip TAPCs .....	406
16.7 Controlling the selection state of EMTAPCs.....	407
16.7.1 Description .....	407
16.7.2 Specifications .....	408
16.8 Control via the CLTAPC Instruction Register.....	410
16.8.1 Description .....	410
16.8.1.1 Exclusion of TAPCs.....	410
16.8.1.2 Isolation of TAPCs.....	412
16.8.1.3 CLTAPC output registering of MTCP and MTDP control .....	413
16.8.2 Specifications .....	414
16.9 Control via one or more CLTAPC Data Registers.....	414
16.9.1 Description .....	414
16.9.2 Specifications .....	415
16.10 Control via internal or external <i>tapc_select</i> signals .....	416
16.10.1 Description.....	416
16.10.2 Specifications .....	417
16.11 Example use cases .....	418
16.11.1 IR control method with exclusions of TAPCs.....	419
16.11.2 IR control method with isolation of TAPCs.....	419
16.11.3 DR control method with exclusion of TAPCs.....	420
16.11.4 DR control method with isolation of TAPCs .....	420
16.12 Identification of on-chip TAP controller(s) .....	421
16.12.1 Description .....	421
16.12.2 Specifications .....	422
16.13 Multiple dies in one package .....	422
16.13.1 Description.....	422
16.13.1.1 Exposing an IEEE 1149.1 DR for the <i>BYPASS</i> and <i>IDCODE</i> instructions.....	423

16.13.1.2 Exposing the complete boundary-scan chain for IEEE 1149.1 instructions.....	423
16.13.1.3 BSDL documentation.....	423
16.13.1.4 Packaging dies.....	424
16.13.1.5 SiP-TAP POR* functionality .....	425
16.13.1.6 DR-wire bypass .....	425
16.13.2 Specifications.....	426
16.14 Managing STL Group Membership.....	427
16.15 RSU operation .....	427
16.15.1 Description.....	427
16.15.2 Specifications.....	427
16.16 Programming considerations .....	428
17. Extended concepts .....	429
17.1 Introduction .....	429
17.2 Suitability of <i>BYPASS</i> and <i>IDCODE</i> instructions for extended control .....	429
17.3 ZBS detection .....	429
17.3.1 Description.....	429
17.3.2 Specifications.....	430
17.4 Incrementing, locking, and clearing the ZBS count.....	430
17.4.1 Description.....	430
17.4.2 Specifications.....	431
17.5 Shared use of ZBSs by the EPU and STL.....	433
17.5.1 Description.....	433
17.5.1.1 EPU Operating States.....	433
17.5.1.2 EPU Operating State characteristics.....	434
17.5.1.3 ZBS use that is compatible with EPU Operating States .....	435
17.5.1.4 An approach to implementing EPU Operating States .....	436
17.5.2 Specifications.....	438
17.6 EPU functionality associated with the ZBS count .....	439
17.6.1 Description.....	439
17.6.2 Specifications .....	439
17.7 Programming considerations .....	440
18. T1 TAP.7 .....	441
18.1 Introduction .....	441
18.2 Deployment .....	442
18.3 Capabilities .....	442
18.3.1 Inherited .....	442
18.3.2 New .....	443
18.4 Register and command portfolio.....	443
18.4.1 Description.....	443
18.4.1.1 General information .....	443
18.4.1.2 Register acronyms .....	444
18.4.1.3 Global Registers .....	445
18.4.1.4 Registers already described .....	445
18.4.1.5 New register descriptions .....	445
18.4.2 Specifications.....	445
18.5 Configurations .....	447
18.5.1 Description .....	447
18.5.2 Specifications .....	447
18.6 Start-up behavior .....	448
18.6.1 Description.....	448
18.6.2 Specifications .....	448
18.7 Conditional Group Membership .....	448
18.8 Test Reset .....	449
18.8.1 Description .....	449
18.8.2 Specifications .....	450
18.9 Functional reset.....	451

18.9.1 Description .....	451
18.9.2 Specifications .....	453
18.10 Power control .....	455
18.10.1 Description .....	455
18.10.1.1 Overview .....	455
18.10.1.1.1 Use cases .....	456
18.10.1.1.2 Power-control options .....	456
18.10.1.1.3 Power management within a typical system .....	457
18.10.1.1.4 Power-control topics .....	458
18.10.1.2 Power-Control Model .....	458
18.10.1.2.1 TAP.7 Controller power-management states .....	458
18.10.1.2.2 Key model attributes .....	459
18.10.1.3 The chip-level power manager's role in power control .....	460
18.10.1.3.1 Responsibilities .....	460
18.10.1.3.2 Interaction with TAP.7 Controller .....	460
18.10.1.3.3 Periods when a Type-0 Reset is asserted .....	461
18.10.1.3.4 Handling of power-up and power-down requests .....	461
18.10.1.3.5 Chip-Level power-up and power-down enables .....	462
18.10.1.3.6 The default Power-Control Mode .....	462
18.10.1.4 The DTS' role in power control .....	463
18.10.1.4.1 Responsibilities .....	463
18.10.1.4.2 Directed power-up .....	463
18.10.1.4.3 Detected power-up .....	464
18.10.1.5 The TAP.7 Controller's role in power control .....	465
18.10.1.5.1 Responsibilities .....	465
18.10.1.5.2 Operation .....	465
18.10.1.5.3 Test periods .....	466
18.10.1.5.4 Power-up confirmation test .....	467
18.10.1.5.5 Power-down initiation test .....	467
18.10.1.5.6 Power-down request summary .....	468
18.10.1.5.7 Awaiting power-down with the TAP.7 Controller operation shutdown .....	468
18.10.1.6 Example power-down sequences .....	469
18.10.1.7 An approach to implementing power control .....	471
18.10.2 Specifications .....	471
18.11 RSU operation .....	476
18.11.1 Description .....	476
18.11.2 Specifications .....	477
18.12 Programming considerations .....	477
19. T2 TAP.7 .....	478
19.1 Introduction .....	478
19.2 Deployment .....	480
19.3 Capabilities .....	480
19.3.1 Inherited .....	480
19.3.2 New .....	480
19.4 Register and command portfolio .....	481
19.4.1 Description .....	481
19.4.1.1 General information .....	481
19.4.1.2 Register acronyms .....	481
19.4.1.3 Effects of a Long-Form Selection Sequence .....	481
19.4.1.4 New register descriptions .....	481
19.4.2 Specifications .....	482
19.5 Configurations .....	483
19.5.1 Description .....	483
19.5.2 Specifications .....	483
19.6 Start-up behavior .....	483
19.6.1 Description .....	483

19.6.2 Specifications .....	483
19.7 Scan formats .....	484
19.7.1 Description .....	484
19.7.2 Specifications .....	484
19.8 STL Group Membership .....	484
19.8.1 Description .....	484
19.8.1.1 Factors affecting group membership .....	484
19.8.1.2 Reset effects .....	485
19.8.1.3 TAPC state effects .....	485
19.8.1.4 Control Path effects .....	485
19.8.1.5 Parked state/selection relationships .....	487
19.8.1.6 Concurrent CLTAPC and EMTAPC selection changes .....	488
19.8.1.7 An approach to implementing CLTAPC selection with the T2 Class .....	489
19.8.2 Specifications .....	491
19.9 RSU operation .....	493
19.9.1 Description .....	493
19.9.2 Specifications .....	493
19.10 Programming considerations .....	494
20. T3 TAP.7 .....	495
20.1 Introduction .....	495
20.2 Deployment .....	497
20.3 Capabilities .....	498
20.3.1 Inherited .....	498
20.3.2 New .....	498
20.4 Register and command portfolio .....	498
20.4.1 Description .....	498
20.4.1.1 General information .....	498
20.4.1.2 Register acronyms .....	499
20.4.1.3 Effect of a Long-Form Selection Sequence .....	499
20.4.2 Specifications .....	499
20.5 Configurations .....	500
20.5.1 Description .....	500
20.5.2 Specifications .....	501
20.6 Start-up behavior .....	501
20.6.1 Description .....	501
20.6.2 Specifications .....	501
20.7 Scan formats .....	501
20.7.1 Description .....	501
20.7.2 Specifications .....	502
20.8 TAP.7 Controller Address (TCA) .....	502
20.8.1 Description .....	502
20.8.2 Specifications .....	503
20.9 Aliasing the TCA to a Controller ID .....	504
20.9.1 Description .....	504
20.9.1.1 CID Allocate Command (CIDA) .....	504
20.9.1.1.1 CID-allocation criteria .....	505
20.9.1.1.2 CID-allocation candidates .....	505
20.9.1.1.3 CID-allocation process .....	505
20.9.1.1.4 Directed CID Allocation .....	506
20.9.1.1.5 Undirected CID Allocation .....	506
20.9.1.2 External AT generation with the JScan3 Scan Format .....	506
20.9.1.3 CID-allocation examples .....	507
20.9.1.4 An approach to implementing CID allocation .....	508
20.9.2 Specifications .....	509
20.10 Scan Selection Directives .....	512
20.10.1 Description .....	512

20.10.1.1 Overview .....	512
20.10.1.2 SSD format.....	513
20.10.1.3 SSD effects on STL Group Membership.....	513
20.10.1.4 Enabling SSD processing .....	514
20.10.1.5 SSD processing .....	514
20.10.1.6 Conditional SSD execution .....	517
20.10.1.7 SSD interaction with other controller functions .....	517
20.10.1.8 SSD State Machine.....	517
20.10.1.9 SSD states allowing Scan Group Membership.....	519
20.10.1.9.1 Using SSDs to create Series and Star-Equivalent Scans.....	520
20.10.1.9.2 Examples of SSD use .....	520
20.10.1.10 An approach to implementing the SSD function.....	525
20.10.2 Specifications .....	527
20.11 Scan Topology Training Sequence .....	531
20.11.1 Description.....	531
20.11.1.1 Overview .....	531
20.11.1.2 Topology Register Function.....	531
20.11.1.3 Use cases .....	532
20.11.1.3.1 Operation with a single TAP.7 Branch.....	532
20.11.1.3.2 Operation with more than one TAP.7 Branch .....	533
20.11.1.4 Scan-path characteristics used to determine the scan topology.....	533
20.11.1.5 Scan Topology Training Command Sequence .....	534
20.11.1.5.1 Connectivity test.....	535
20.11.1.5.2 Continuity test .....	535
20.11.2 Specifications .....	536
20.12 Managing STL Group Membership.....	536
20.12.1 Description.....	536
20.12.1.1 Factors affecting group membership .....	536
20.12.1.2 An approach to implementing CLTAPC selection with T3 and above classes .....	536
20.12.2 Specifications .....	538
20.13 RSU operation .....	540
20.13.1 Description.....	540
20.13.2 Specifications.....	541
20.14 Programming considerations .....	541
21. Advanced concepts .....	542
21.1 Architecture .....	542
21.2 Advanced capabilities .....	543
21.2.1 Mandatory and optional capabilities .....	543
21.2.2 Online and Offline operation .....	544
21.2.3 Interoperability with T0–T3 TAP.7s.....	544
21.2.4 Interoperability with T4 and above TAP.7s .....	544
21.3 Comparing the Standard and Advanced Protocols.....	545
21.4 APU functions .....	545
21.4.1 Conceptual view.....	545
21.4.2 Bypass Function.....	546
21.4.3 Scan Function.....	547
21.4.4 Transport Function.....	548
21.4.5 Bypass/Scan/Control Function interactions .....	550
21.5 APU interfaces.....	550
21.5.1 TAP.....	550
21.5.2 EPU.....	551
21.5.3 DCC .....	551
21.6 APU function/Operating State relationships.....	553
21.6.1 Operating State/function relationships .....	553
21.6.2 APU Operating State changes.....	554
21.6.3 Example operating state sequences .....	555

21.7 TAPC state/packet relationships .....	557
21.7.1 TAPC state and packet sequence relationships .....	557
21.7.2 Constructing packet sequences .....	559
21.7.3 Packet combinations/TAPC state relationships.....	560
21.7.4 Scheduling of packets .....	560
21.8 User's and implementer's views of the Advanced Protocol .....	562
21.8.1 User's view .....	562
21.8.2 Implementer's view.....	563
21.9 An approach to implementing APU Operating State scheduling.....	563
21.10 Structure of the clauses describing T4 and above TAP.7s.....	565
22. APU Scan Packets .....	567
22.1 CPs.....	567
22.2 SPs .....	567
22.2.1 Conceptual view of an SP .....	567
22.2.2 SP format .....	568
22.2.3 SP content .....	569
22.2.3.1 Header Element content .....	569
22.2.3.2 Payload Element content.....	569
22.2.3.3 Delay Element content .....	570
22.2.4 Types of output bit-frame transactions.....	571
22.3 SPs that advance the TAPC state .....	572
22.4 TPs.....	573
22.4.1 Conceptual view of a TP .....	573
22.4.2 TP format .....	573
22.4.3 Content.....	575
22.5 APU state diagram .....	575
22.6 An approach to implementing packet scheduling .....	577
22.6.1 Pipelining and its effects .....	577
22.6.2 SP Element scheduling.....	577
22.6.3 TP Element scheduling .....	578
23. T4 TAP.7 .....	579
23.1 Introduction .....	579
23.2 Deployment .....	579
23.3 Capabilities .....	580
23.3.1 Inherited .....	580
23.3.2 New.....	580
23.4 Register and command portfolio.....	581
23.4.1 Description .....	581
23.4.1.1 General information .....	581
23.4.1.2 Register acronyms .....	581
23.4.1.3 Effect of a Long-Form Selection Sequence.....	581
23.4.1.4 New register descriptions .....	582
23.4.1.4.1 Auxiliary Pin Function Control (APFC) Register .....	582
23.4.1.4.2 Delay Control (DLYC) Register.....	582
23.4.1.4.3 Ready Control (RDYC) Register.....	582
23.4.1.4.4 Sample Using Rising Edge (SREdge) Register.....	582
23.4.1.4.5 Scan Format (SCNFMT) Register .....	582
23.4.1.4.6 System Test Clock Duty Cycle (STCKDC) Register .....	582
23.4.2 Specifications .....	583
23.5 Configurations .....	585
23.5.1 Description .....	585
23.5.2 Specifications .....	585
23.6 Start-up behavior .....	586
23.6.1 Description .....	586
23.6.2 Specifications .....	586
23.7 Scan formats .....	587

23.7.1 Description .....	587
23.7.1.1 Overview .....	587
23.7.1.2 Addition of optional scan formats .....	589
23.7.1.3 Influences on scan format definition .....	589
23.7.1.4 Performance and flexibility tradeoffs .....	589
23.7.1.5 Comparing the scan formats .....	590
23.7.1.5.1 MScan .....	590
23.7.1.5.2 OScan .....	590
23.7.1.5.3 SScan .....	591
23.7.2 Specification .....	592
23.8 Configuration Faults .....	592
23.8.1 Description .....	592
23.8.2 Specifications .....	592
23.9 Increasing STL performance .....	593
23.9.1 Description .....	593
23.9.2 Specifications .....	594
23.10 Auxiliary Pin Function Control .....	595
23.10.1 Description .....	595
23.10.2 Specifications .....	596
23.11 Sample Using Rising Edge .....	596
23.11.1 Description .....	596
23.11.2 Specifications .....	597
23.12 System and EPU TMS signal values .....	597
23.12.1 Description .....	597
23.12.2 Specifications .....	598
23.13 System and EPU TDI signal values .....	599
23.13.1 Description .....	599
23.13.2 Specifications .....	600
23.14 RDY bit values .....	601
23.14.1 Description .....	601
23.14.2 Specifications .....	602
23.15 TDO bit values .....	603
23.15.1 Description .....	603
23.15.2 Specifications .....	604
23.16 Advanced Protocol effects on the EPU/CLTAPC relationship .....	604
23.16.1 Description .....	604
23.16.2 Specifications .....	604
23.17 SSD detection .....	604
23.17.1 Description .....	604
23.17.2 Specifications .....	604
23.18 Programming considerations .....	605
23.19 An approach to implementing a TAP.7 Controller with maximum performance .....	605
24. MScan Scan Format .....	607
24.1 Capabilities .....	607
24.1.1 Primary purpose .....	607
24.1.2 Application types supported .....	607
24.1.3 Important characteristics .....	608
24.2 High-level operation .....	608
24.3 Scan Packet content .....	609
24.3.1 Description .....	609
24.3.2 Specifications .....	609
24.4 Payload Element .....	609
24.4.1 Description .....	609
24.4.1.1 Format .....	609
24.4.1.2 Relationship to EPU signals .....	610
24.4.1.3 RDY bits .....	611

24.4.2 Specification .....	613
24.5 Delay Element .....	613
24.5.1 Description .....	613
24.5.1.1 Format .....	613
24.5.1.2 Delay Element Directives.....	614
24.5.1.3 Uses.....	614
24.5.2 Specifications .....	615
24.6 Advancing the TAPC state .....	616
24.6.1 Description .....	616
24.6.2 Specifications .....	617
24.7 CID allocation.....	617
24.7.1 Description .....	617
24.7.2 Specifications .....	619
24.8 Increasing STL performance with the MScan Scan Format .....	619
24.9 An approach to implementing the MScan Scan Format .....	619
24.9.1 Payload State Machine.....	619
24.9.2 Ready State Machine .....	621
24.9.3 Delay State Machine .....	622
24.10 Where to find examples .....	623
25. OScan Scan Formats.....	624
25.1 Capabilities .....	624
25.1.1 Primary purpose .....	624
25.1.2 Application types supported.....	624
25.1.3 Important characteristics .....	625
25.2 High-level operation .....	625
25.3 Scan Packet content .....	626
25.3.1 Description .....	626
25.3.2 Specifications.....	626
25.4 Payload Element .....	627
25.4.1 Description .....	627
25.4.1.1 Format .....	627
25.4.1.2 Optimizations .....	628
25.4.1.3 Relationship to EPU signals .....	629
25.4.1.4 Input bit-frame .....	633
25.4.1.5 Drive types .....	633
25.4.1.6 RDY bits.....	634
25.4.2 Specifications.....	635
25.5 Delay Element .....	636
25.6 Advancing the TAPC state .....	637
25.6.1 Description .....	637
25.6.2 Specifications .....	639
25.7 CID allocation.....	640
25.7.1 Description .....	640
25.7.2 Specifications .....	640
25.8 Increasing STL performance with OScan Scan Formats .....	640
25.9 An approach to implementing OScan Scan Formats .....	641
25.9.1 Payload State Machine.....	641
25.9.2 Shift Progress flag.....	643
25.9.3 CSM and SSM activation.....	644
25.9.4 RDY State Machine .....	644
25.9.5 TAP advance.....	645
25.10 Where to find examples .....	645
26. SScan Scan Formats .....	646
26.1 Capabilities .....	646
26.1.1 Primary purpose .....	646
26.1.2 Application types supported.....	647

26.1.3 Control Segments.....	648
26.1.4 Data Segments .....	649
26.1.5 Stall profiles.....	649
26.1.6 Important characteristics .....	650
26.2 High-level operation .....	650
26.2.1 Overview.....	650
26.2.2 Segments and their use.....	651
26.2.2.1 Description .....	651
26.2.2.1.1 Use with a TAP.1-like component.....	651
26.2.2.1.2 Use with a DMA or FIFO component .....	652
26.2.2.1.3 Use with a direct-access data-rate-dependent component .....	653
26.2.2.1.4 Use with a buffered TDI/TMS component .....	653
26.2.2.1.5 Utilizing the same scan format for two applications types .....	654
26.2.2.2 Specifications .....	655
26.3 Scan Packet content .....	655
26.3.1 Description .....	655
26.3.2 Specifications.....	656
26.4 Header Element .....	657
26.4.1 Description.....	657
26.4.2 Specifications.....	657
26.5 Payload Element .....	658
26.5.1 Description.....	658
26.5.1.1 Factors determining payload content.....	658
26.5.1.2 Optimizations .....	660
26.5.1.3 Input bit-frame .....	663
26.5.1.4 Output bit-frame.....	666
26.5.1.4.1 Content .....	666
26.5.1.4.2 SScan0/1 output-only segments.....	666
26.5.1.4.3 SScan2/3 output-only segments.....	667
26.5.2 Specifications .....	671
26.6 Delay Element .....	673
26.7 Packet sequences and factors influencing them .....	674
26.7.1 Description.....	674
26.7.2 Specifications .....	676
26.8 Advancing the TAPC state .....	677
26.8.1 Description.....	677
26.8.1.1 SP followed by a CP.....	677
26.8.1.2 Control Segments .....	677
26.8.1.3 Data Segments.....	677
26.8.2 Specifications .....	681
26.9 CID allocation.....	681
26.9.1 Description .....	681
26.9.2 Specifications .....	681
26.10 Increasing STL performance with SScan Scan Formats .....	682
26.11 An approach to implementing SScan Scan Formats .....	682
26.11.1 Payload State Machine.....	682
26.11.2 Escape Detection State Machine.....	683
26.11.3 Shift Progress flag.....	684
26.11.4 TAP advance.....	685
26.11.5 Additional entry point loads for output-only segments .....	685
26.11.6 Header Register.....	686
26.11.7 Timing diagrams .....	687
26.12 Where to find examples .....	689
27. T5 TAP.7 .....	690
27.1 Introduction .....	690
27.2 Deployment .....	691

27.3 Capabilities .....	692
27.3.1 Inherited .....	692
27.3.2 New.....	692
27.4 Register and command portfolio.....	692
27.4.1 Description.....	692
27.4.1.1 General information .....	692
27.4.1.2 Register acronyms .....	693
27.4.1.3 Effect of a Long-Form Selection Sequence.....	693
27.4.1.4 Transport protocol revision (TPPREV) register.....	693
27.4.1.5 Transport states (TPST) .....	694
27.4.1.6 Data Element length (TP_DELN) .....	694
27.4.1.7 Physical Data Channel to Logical Data Channel association (PDCx_LCA).....	694
27.4.1.8 Physical Data Channel selected (PDCx_SEL) .....	694
27.4.1.9 Physical Data Channel DCC selection (PDCx_DCC).....	694
27.4.1.10 Physical Data Channel DCC Control Registers (PDCx_DCCy_CRz).....	694
27.4.2 Specifications .....	695
27.5 Configurations .....	699
27.5.1 Description.....	699
27.5.2 Specifications.....	700
27.6 Start-up behavior .....	701
27.6.1 Description.....	701
27.6.2 Specifications.....	701
27.7 Configuration Faults .....	701
27.7.1 Description.....	701
27.7.2 Specifications.....	701
27.8 Enabling transport.....	702
27.8.1 Description.....	702
27.8.2 Specifications.....	702
27.9 Transport Packet composition .....	703
27.9.1 Operations.....	703
27.9.2 Directive, Register, and Data Elements .....	704
27.10 Directive Elements.....	704
27.10.1 Description.....	704
27.10.1.1 Overview .....	704
27.10.1.2 Directive Element acronyms .....	705
27.10.1.3 Surrounding context of directives .....	705
27.10.1.4 Directive/transport building block relationships .....	706
27.10.1.5 Directive encoding .....	707
27.10.1.6 Directive types.....	708
27.10.1.6.1 Unconditional Directives.....	709
27.10.1.6.2 Reset Directives.....	709
27.10.1.6.3 Selection Directives.....	709
27.10.1.6.4 Conditional Directives.....	710
27.10.1.6.5 Transfer Directives .....	711
27.10.2 Specifications .....	711
27.11 Register Elements .....	715
27.11.1 Description.....	715
27.11.1.1 Overview .....	715
27.11.1.2 Register Element length .....	715
27.11.1.3 Transfer direction .....	715
27.11.1.4 Summary of Register Element characteristics .....	715
27.11.2 Specifications .....	716
27.12 Data Elements .....	716
27.12.1 Description.....	716
27.12.1.1 Overview .....	716
27.12.1.2 Data Element length .....	716

27.12.1.3 Transfer direction .....	716
27.12.1.4 Utilization and generation of data .....	717
27.12.1.5 Operation with single and multiple clients.....	717
27.12.1.6 Summary of Data Element characteristics.....	718
27.12.2 Specifications.....	718
27.13 Selection of control and data targets.....	719
27.14 Data Channel Client functions.....	720
27.14.1 Description.....	720
27.14.1.1 Initializing a Data Channel Client .....	720
27.14.1.2 Orderly shutdown of a transfer.....	720
27.14.1.3 Effects of Online/Offline operation on the Transport Function .....	721
27.14.2 Specifications.....	722
27.15 Partitioning of the Transport Control Function.....	722
27.15.1 Building blocks .....	722
27.15.2 Directive/register/operational relationships .....	724
27.16 Programming considerations .....	725
27.16.1 Managing transport with the DTS.....	725
27.16.2 Single and multi-client data exchanges .....	725
27.16.3 Bandwidth allocation .....	725
27.16.4 Common and uncommon operations performed with directives.....	725
27.16.5 Dynamic source/destination changes .....	726
27.16.6 Transfer alignment characteristics .....	727
27.17 Aspects of transport not covered by this specification.....	727
28. Transport operation and interfaces .....	728
28.1 Introduction .....	728
28.2 TAP interface .....	728
28.2.1 TPA state flow.....	728
28.2.2 Directive Element characteristics.....	729
28.2.2.1 Description .....	729
28.2.2.2 Specifications .....	729
28.2.3 Register Element characteristics .....	731
28.2.3.1 Description .....	731
28.2.3.1.1 Format, timing, and TMSC signal drive characteristics .....	731
28.2.3.1.2 Register Element length.....	731
28.2.3.1.3 Register Element content.....	732
28.2.3.2 Specifications .....	733
28.2.4 Data Element characteristics .....	734
28.2.4.1 Description .....	734
28.2.4.1.1 Format, timing, and TMSC signal drive characteristics .....	734
28.2.4.1.2 Data Element length .....	735
28.2.4.1.3 Data Element content .....	736
28.2.4.1.4 Drive characteristics .....	737
28.2.4.1.5 Multi-client data transfers.....	737
28.2.4.1.6 Data Element alignment with the data that is transported .....	738
28.2.4.2 Specifications .....	738
28.3 Transport State Machine.....	739
28.3.1 Description .....	739
28.3.1.1 Conceptual view .....	739
28.3.1.2 TSM operation .....	741
28.3.1.3 Scheduling a TP .....	742
28.3.1.4 Starting/restarting directive processing .....	744
28.3.1.5 Completing a TP.....	744
28.3.2 Specifications .....	745
28.4 PDCx/DCC interface .....	746
28.4.1 Description .....	746
28.4.1.1 Signal functions.....	747

28.4.1.2 Signal descriptions .....	747
28.4.1.3 Signal use .....	749
28.4.2 Specifications.....	753
28.5 Five-bit directives .....	753
28.5.1 Description.....	753
28.5.2 Specifications.....	753
28.6 Eight-bit directives.....	755
28.6.1 Description.....	755
28.6.2 Specifications.....	756
28.7 12-bit directives .....	757
28.7.1 Description.....	757
28.7.2 Specifications.....	757
28.8 DCC interface operation .....	760
28.8.1 Basic capability.....	760
28.8.2 Pipelining register <i>dcc_rdo</i> signaling.....	761
28.8.3 Pipelining <i>dcc_ddo</i> and <i>dcc_cor</i> signaling.....	761
28.9 An approach to implementing the Transport Function .....	762
28.9.1 Overview.....	762
28.9.2 The Transport State Machine .....	763
28.9.3 Multi-use register bits .....	764
28.9.3.1 Input configurations .....	764
28.9.3.2 Transport Packet processing.....	765
28.9.3.2.1 Directive processing .....	765
28.9.3.2.2 Data Element processing .....	765
28.9.3.2.3 12-bit directive processing (other than TP_CRR and TP_CRW) .....	766
28.9.3.2.4 TP_CRR and TP_CRW Directive processing .....	766
28.9.3.3 TSM state/register value relationships .....	767
28.9.3.4 Transport output and DCR input selection.....	770
28.9.3.5 TP activity examples .....	772
29. Test concepts .....	777
29.1 Introduction .....	777
29.2 Interoperability .....	777
29.3 Construction of the unit under test.....	778
29.4 Background (IEEE 1149.1 paradigm).....	778
29.4.1 Topology .....	779
29.4.2 Scan-state sequencing .....	779
29.5 Implications for test applications arising from this standard .....	780
29.5.1 Divergences versus IEEE Std 1149.1.....	780
29.5.2 Accommodation/resolution of divergences versus IEEE Std 1149.1.....	781
29.6 Test example—a narrative .....	781
29.7 Describing the unit under test .....	782
29.8 Documentation model.....	783
29.9 Considerations for large-system applications .....	784
30. Documenting IEEE 1149.7 test endpoints (BSDL.7).....	786
30.1 Introduction .....	786
30.2 Conventions .....	787
30.3 Purpose of BSDL.7 .....	787
30.4 Scope of BSDL.7 .....	787
30.5 Expectations of a BSDL.7 parser.....	788
30.6 Relationship of BSDL.7 to BSDL.1 .....	788
30.6.1 Description.....	788
30.6.2 Specifications .....	789
30.7 Lexical elements of BSDL.7 .....	789
30.7.1 Description .....	789
30.7.2 Specifications .....	789
30.8 BSDL.7 reserved words .....	789

30.8.1 Description .....	789
30.8.2 Specifications .....	789
30.9 Components of a BSDL.7 description .....	790
30.9.1 Description .....	790
30.9.2 Specifications .....	790
30.10 The entity description (BSDL.7).....	790
30.10.1 Overall structure of the entity description (BSDL.7).....	790
30.10.1.1 Syntax and content .....	790
30.10.1.1.1 Description .....	790
30.10.1.1.2 Specifications .....	791
30.10.1.2 Semantic checks .....	792
30.10.1.2.1 Description .....	792
30.10.1.2.2 Specifications .....	792
30.10.2 Standard use statement (BSDL.7).....	792
30.10.2.1 Syntax and content .....	792
30.10.2.1.1 Description .....	792
30.10.2.1.2 Specifications .....	793
30.10.2.2 Examples .....	793
30.10.3 Version control.....	794
30.10.3.1 Syntax and content .....	794
30.10.3.1.1 Description .....	794
30.10.3.1.2 Specifications .....	794
30.10.4 Component conformance statement (BSDL.7).....	794
30.10.4.1 Syntax and content .....	794
30.10.4.1.1 Description .....	794
30.10.4.1.2 Specifications .....	795
30.10.4.2 Examples .....	795
30.10.4.3 Semantic checks .....	795
30.10.5 Scan port identification (BSDL.7) .....	795
30.10.5.1 Syntax and content .....	795
30.10.5.1.1 Description .....	795
30.10.5.1.2 Specifications .....	796
30.10.5.2 Examples .....	796
30.10.5.3 Semantic checks .....	798
30.10.5.3.1 Description .....	798
30.10.5.3.2 Specifications .....	798
30.10.6 Compliance enable description (BSDL.7) .....	799
30.10.6.1 Syntax and content .....	799
30.10.6.1.1 Description .....	799
30.10.6.1.2 Specifications .....	799
30.10.6.2 Examples .....	799
30.10.6.3 Semantic checks .....	800
30.10.6.3.1 Description .....	800
30.10.6.3.2 Specifications .....	800
30.10.7 Device identification register description (BSDL.7).....	800
30.10.7.1 Syntax and content .....	800
30.10.7.1.1 Description .....	800
30.10.7.1.2 Specifications .....	800
30.10.7.2 Examples .....	800
30.10.7.3 Semantic checks .....	801
30.10.7.3.1 Description .....	801
30.10.7.3.2 Specifications .....	801
30.10.8 Configuration register description (BSDL.7).....	801
30.10.8.1 Syntax and content .....	801
30.10.8.1.1 Description .....	801
30.10.8.1.2 Specifications .....	801

30.10.8.2 Examples .....	801
30.10.8.3 Semantic checks .....	802
30.10.8.3.1 Description .....	802
30.10.8.3.2 Specifications .....	802
30.11 The Standard BSDL.7 Package STD_1149_7_2009 .....	803
30.11.1 Description.....	803
30.11.2 Specifications .....	803
30.12 A typical application of BSDL.7 .....	803
31. Documenting IEEE 1149.7 test modules (HSDL.7).....	806
31.1 Introduction .....	806
31.2 Conventions .....	806
31.3 Purpose of HSDL.7.....	807
31.4 Scope of HSDL.7.....	807
31.5 Expectations of an HSDL.7 parser.....	808
31.6 Relationship of HSDL.7 to BSDL.7 (and BSDL.1).....	808
31.6.1 Description.....	808
31.6.2 Specifications .....	809
31.7 Lexical elements of HSDL.7 .....	809
31.7.1 Description.....	809
31.7.2 Specifications.....	809
31.8 HSDL.7 reserved words.....	809
31.8.1 Description.....	809
31.8.2 Specifications .....	809
31.9 Components of an HSDL.7 description .....	809
31.9.1 Description.....	809
31.9.2 Specifications.....	810
31.10 The entity description (HSDL.7) .....	810
31.10.1 Overall structure of the entity description (HSDL.7).....	810
31.10.1.1 Syntax and content .....	810
31.10.1.1.1 Description .....	810
31.10.1.1.2 Specifications .....	810
31.10.1.2 Semantic checks .....	811
31.10.1.2.1 Description .....	811
31.10.1.2.2 Specifications .....	811
31.10.2 Module standard use statement (HSDL.7) .....	811
31.10.2.1 Syntax and content .....	811
31.10.2.1.1 Description .....	811
31.10.2.1.2 Specifications .....	812
31.10.2.2 Examples .....	813
31.10.3 Version control.....	813
31.10.3.1 Syntax and content .....	813
31.10.3.1.1 Description .....	813
31.10.3.1.2 Specifications .....	813
31.10.4 Module component conformance statement (HSDL.7).....	814
31.10.4.1 Syntax and content .....	814
31.10.4.1.1 Description .....	814
31.10.4.1.2 Specifications .....	814
31.10.4.2 Examples .....	814
31.10.4.3 Semantic checks .....	814
31.10.5 Module package pin mappings (HSDL.7).....	815
31.10.5.1 Syntax and content .....	815
31.10.5.1.1 Description .....	815
31.10.5.1.2 Specifications .....	815
31.10.5.2 Examples .....	815
31.10.5.3 Semantic checks .....	815
31.10.5.3.1 Description .....	815

31.10.5.3.2 Specifications .....	815
31.10.6 Module scan port identification (HSDL.7) .....	815
31.10.6.1 Syntax and content .....	815
31.10.6.1.1 Description .....	815
31.10.6.1.2 Specifications .....	816
31.10.6.2 Examples .....	816
31.10.6.3 Semantic checks .....	817
31.10.6.3.1 Description .....	817
31.10.6.3.2 Specifications .....	817
31.10.7 Module members declaration (HSDL.7) .....	818
31.10.7.1 Syntax and content .....	818
31.10.7.1.1 Description .....	818
31.10.7.1.2 Specifications .....	819
31.10.7.2 Examples .....	820
31.10.7.3 Semantic checks .....	821
31.10.7.3.1 Description .....	821
31.10.7.3.2 Specifications .....	821
31.11 The Standard HSDL.7 Package STD_1149_7_2009_module .....	822
31.11.1 Description .....	822
31.11.2 Specifications .....	822
31.12 Applications of HSDL.7 .....	822
31.12.1 HSDL.7 for IEEE 1149.1 serial connection using one TMS signal .....	822
31.12.2 HSDL.7 for IEEE 1149.1 connection in two paralleled serial chains .....	824
31.12.3 HSDL.7 for a basic Star Scan Topology .....	825
31.12.4 HSDL.7 for a basic hierarchical topology .....	826
31.12.5 A typical application of HSDL.7 .....	828
Annex A (informative) IEEE 1149.1 reference material .....	830
Annex B (informative) Scan examples in timing diagram form .....	834
B.1 MScan and OScan SP types .....	834
B.2 MScan and OScan transactions .....	835
B.2.1 MScan transaction .....	835
B.2.2 OScan0 transaction .....	837
B.2.3 OScan1 transaction .....	838
B.2.4 OScan2 transaction .....	839
B.2.5 OScan3 transaction .....	840
B.2.6 OScan4 transaction .....	841
B.2.7 OScan5 transaction .....	842
B.2.8 OScan6 transaction .....	843
B.2.9 OScan7 transaction .....	845
B.3 SScan transactions .....	846
B.3.1 SScan0 transaction .....	847
B.3.2 SScan1 transaction .....	850
B.3.3 SScan2 transaction .....	853
B.3.4 SScan3 transaction .....	856
B.4 BDX and CDX Transport Packet examples .....	859
Annex C (informative) Scan examples in tabular form .....	861
C.1 Overview .....	861
C.2 MScan and OScan SP types .....	861
C.3 SScan SP types .....	876
Annex D (informative) Programming considerations .....	903
D.1 Overview .....	903
D.2 DTS' view of TAPs .....	903
D.2.1 Technology branches .....	903
D.2.2 Power-management RSU combinations .....	904
D.2.3 TAP.7 Controller deselection and selection .....	906
D.2.4 Start-up versus steady-state operation .....	907

D.2.5 Start-up .....	908
D.2.6 Initialization requirements .....	909
D.2.6.1 Initial state.....	909
D.2.6.2 Factors influencing initialization .....	909
D.2.6.3 Initialization function.....	909
D.2.6.4 Reset Escape .....	910
D.2.6.5 Reset TAP.7 Controllers without an RSU.....	911
D.2.6.6 Power-up TAP.7 Controllers that are un-powered.....	911
D.2.6.7 Placing a TAP.7 Controller Online that is Offline-at-Start-up.....	911
D.2.7 Scan Topology Training .....	915
D.3 Scan topology interrogation.....	916
D.3.1 Series Branch interrogation.....	917
D.3.1.1 Series characteristics used for interrogation .....	917
D.3.1.2 Interrogation process.....	918
D.3.1.3 Configuration Register reads .....	920
D.3.1.4 Determining whether a Series Branch is selectable .....	921
D.3.1.5 Quick determination of TAP types in a Series Branch .....	921
D.3.2 Star-4 Branch interrogation.....	922
D.3.2.1 Information provided .....	922
D.3.2.2 Interrogation process.....	922
D.3.3 Star-2 Branch interrogation.....	926
D.3.3.1 Information provided .....	926
D.3.3.2 Interrogation process.....	926
D.4 Establishing TAP.7 operating conditions.....	929
D.5 CID management .....	929
D.6 Managing simultaneous debug actions .....	930
D.7 Operations to avoid with a Star-4 Scan Topology .....	930
D.7.1 The use of the JScan0–JScan2 Scan Formats .....	930
D.7.2 The selection of CLTAPCs of TAP.7 Controllers allocated the same CID .....	930
D.8 Using a T5 TAP.7 .....	931
D.8.1 Setup and use of the Transport Function.....	931
D.8.2 Sharing the use of a Logical Data Channel.....	931
D.8.3 Transferring data with Transport Packet Data Payloads .....	931
Annex E (informative) Recommended electrical characteristics.....	932
Annex F (informative) Connectivity/electrical recommendations.....	933
F.1 Overview .....	933
F.1.1 General information.....	933
F.1.2 Factors affecting reliable operation .....	933
F.1.3 Factors affecting link performance.....	934
F.2 Physical connection topologies considered .....	935
F.3 Termination schemes considered.....	936
F.4 Chip considerations .....	938
F.4.1 SOC input conditioning .....	938
F.4.2 Signal driver rise and fall times .....	939
F.4.3 Clock-to-output delays .....	941
F.4.4 Supply and buffer impedances.....	941
F.4.5 Additional chip considerations related to multi-TAPC topologies .....	941
F.5 Board considerations .....	941
F.5.1 Signaling requiring a termination scheme .....	941
F.5.1.1 Lumped load .....	942
F.5.1.2 Distributed load .....	942
F.5.1.3 Classifying a load .....	942
F.5.1.4 Transmission-line impedance .....	943
F.5.2 Impedance discontinuities/symmetry .....	944
F.5.3 Transmission-line construction.....	947
F.5.3.1 Uniformity of transmission-line impedance .....	947

F.5.3.2 Proper construction.....	947
F.5.3.3 Improper construction.....	949
F.5.3.4 Connector treatment .....	949
F.5.4 Choosing a connection configuration .....	949
F.5.5 Termination schemes with simple configurations .....	950
F.5.5.1 Series termination.....	950
F.5.5.1.1 Overview .....	950
F.5.5.1.2 Challenges in series termination.....	951
F.5.5.1.3 Benefits of using series termination.....	951
F.5.5.2 Parallel termination.....	951
F.5.5.2.1 Overview .....	951
F.5.5.2.2 Challenges in using parallel termination .....	953
F.5.5.2.3 Benefits of using parallel termination.....	954
F.5.5.3 Parallel AC termination .....	954
F.5.5.3.1 Overview .....	954
F.5.5.3.2 Challenges in using parallel AC termination .....	956
F.5.5.3.3 Benefits of using parallel AC termination .....	957
F.5.6 Effects of transmission-line length on operating frequency .....	957
F.5.7 Capacitive load isolation and input pin low-pass filtering.....	958
F.5.8 DTS and chip models .....	958
F.5.9 Additional board considerations related to multi-TAPC topologies.....	959
F.6 System considerations for supporting Keeper bias (K bias) in multi-TAPC topologies.....	959
F.6.1 K bias considerations at the chip and board level.....	959
F.6.2 K bias considerations for embedded TAPC topologies .....	961
F.7 DTS considerations .....	963
F.7.1 Source impedance.....	963
F.7.2 Input capacitance isolation .....	964
F.7.2.1 DTS output levels with parallel terminated target topologies.....	964
F.7.2.2 DTS output edge rates .....	964
F.7.2.3 High-speed, high-current, low-voltage configurable edge-rate drivers .....	965
F.7.2.4 Low-voltage, high-current fixed slow edge-rate drive translator.....	965
F.7.2.5 In-system DTS considerations.....	967
F.7.2.5.1 Output buffer impedance .....	967
F.7.2.5.2 Power considerations .....	967
F.7.2.5.3 Signal edge rates.....	967
F.8 Point-to-Point configuration .....	967
F.8.1 Model.....	967
F.8.2 Unidirectional signaling .....	968
F.8.2.1 Series termination.....	968
F.8.2.2 Parallel AC termination .....	968
F.8.3 Bidirectional signaling.....	969
F.9 Line configuration of a transmission line .....	971
F.9.1 Model.....	971
F.9.2 Unidirectional signaling .....	973
F.9.2.1 Parallel AC termination .....	978
F.9.2.2 Parallel termination.....	980
F.9.2.3 Summary .....	980
F.9.3 Bidirectional signaling.....	980
F.9.3.1 Four loads .....	983
F.9.3.2 Eight loads .....	986
F.10 T configuration of a transmission line .....	989
F.10.1 Model.....	989
F.10.2 Unidirectional signaling .....	992
F.10.3 Bidirectional signaling.....	993
F.11 X configuration of a transmission line.....	997
F.11.1 Model.....	997

F.11.2 Unidirectional signaling .....	999
F.11.3 Bidirectional signaling.....	1002
F.12 XT configuration of a transmission line .....	1008
F.12.1 Model.....	1008
F.12.2 Unidirectional signaling .....	1009
F.12.3 Bidirectional signaling.....	1011
F.13 Recommendations .....	1017
F.13.1 Summary .....	1017
F.13.2 Connectivity/termination scheme .....	1018
F.13.3 Termination scheme/capacitive load isolation/signal relationships.....	1020
F.13.4 Utilizing board design tools and simulation .....	1022
Annex G (informative) Utilizing SScan Scan Formats.....	1023
Annex H (informative) The RTCK signal .....	1027
Annex I (informative) Bibliography.....	1035
Index.....	1036

## Figures

Figure 1-1 — IEEE 1149.7 impact areas.....	57
Figure 1-2 — IEEE 1149.7/IEEE 1149.1 upgrade path.....	62
Figure 1-3 — Systems where the DTS and TS source the Test Clock .....	63
Figure 1-4 — Dedicated and Shared Test Clock signal.....	64
Figure 1-5 — Systems with a Return Test Clock .....	64
Figure 1-6 — Types of TAP.7 operation.....	65
Figure 1-7 — TAP.7 capability classes and their layered capability .....	66
Figure 1-8 — Utilization of TAP.7 Classes within scan topologies .....	68
Figure 1-9 — A system with TCK(C) and TMS(C) shared between TAP.7 and technologies .....	69
Figure 1-10 — IEEE 1149.7 standard organization .....	71
Figure 1-11 — Document organization.....	73
Figure 1-12 — Clause subject areas .....	75
Figure 1-13 — Annex subject areas .....	75
Figure 1-14 - Arc priority .....	78
Figure 1-15 - Flowchart symbol expansion .....	79
Figure 1-16 - MSB/LSB conventions .....	80
Figure 4-1 — Scan architectures utilizing IEEE Std 1149.1 .....	93
Figure 4-2 — TAP.7 TAPC hierarchy.....	94
Figure 4-3 — Relationships of parking the TAPC state to its associated resources.....	95
Figure 4-4 — TAP.7 Controller management .....	99
Figure 4-5 — System and Control Path relationship.....	100
Figure 4-6 — Components involved in TAP.7 Controller power management .....	101
Figure 4-7 — TAP.7 Protocols and Escapes .....	108
Figure 4-8 — Control/Standard/Advanced Protocol relationships .....	110
Figure 4-9 — Expanded view of an SP sequence with the OScan1 Scan Format .....	110
Figure 4-10 — Control Protocol/Standard and Advanced Protocol relationships.....	111
Figure 4-11 — Packets, bit-frames, and bits .....	112
Figure 4-12 — TAP.7 architecture .....	116
Figure 4-13 — Conceptual view of the interface bridge .....	117
Figure 4-14 — Operating model types .....	120
Figure 4-15 — Conceptual view of basic T0–T2 TAP.7 operation without RSU .....	121
Figure 4-16 — Conceptual view of T0–T3 TAP.7 operation with an RSU .....	121
Figure 4-17 — TMSC pin utilization flow diagram for a T5 TAP.7.....	122
Figure 5-1 — T0 TAP.7 chip block diagram.....	123
Figure 5-2 — T0 TAP.7 test and debug views of the chip .....	124
Figure 5-3 — System-on-Chip (SoC) with IEEE 1149.7 multiple TAPC architecture .....	125
Figure 5-4 — High-level T0 TAP.7 functional block diagram .....	127
Figure 5-5 — T1 TAP.7 chip block diagram.....	127
Figure 5-6 — T1 TAP.7 features.....	128
Figure 5-7 — Command generation .....	131
Figure 5-8 — DR-Scan sequence used for command creation .....	132
Figure 5-9 — High-level view of TAP.7 Controller registers .....	133
Figure 5-10 — High-level view of TAP.7 Controller Global/Local Registers .....	134
Figure 5-11 — T1 TAP.7 physical and conceptual path relationships .....	135
Figure 5-12 — T1 TAP.7 Control and System Paths .....	136
Figure 5-13 — High-level T1 TAP.7 functional block diagram .....	138
Figure 5-14 — T2 TAP.7 chip block diagram.....	138
Figure 5-15 — T2 TAP.7 scan-path length reduction .....	140
Figure 5-16 — T2 TAP.7 physical and conceptual path relationships .....	141
Figure 5-17 — T2 TAP.7 Control and System Paths .....	142
Figure 5-18 — T2 TAP.7 group membership determination .....	143
Figure 5-19 — High-level T2 TAP.7 functional block diagram .....	144

Figure 5-20 — T3 TAP.7 chip block diagram.....	145
Figure 5-21 — T3 TAP.7 scan-path length reduction .....	146
Figure 5-22 — T3 TAP.7 group membership related to the use of the System Path.....	148
Figure 5-23 — Series-Equivalent Scan using SSDs in a Star Scan Topology.....	152
Figure 5-24 — T2 TAP.7 physical and conceptual path relationships .....	153
Figure 5-25 — T2 TAP.7 Control and System Paths .....	154
Figure 5-26 — High-level T3 TAP.7 functional block diagram .....	155
Figure 6-1 — T4/T5 TAP.7 chip block diagram .....	156
Figure 6-2 — Conceptual block diagram of the functions of a T4 TAP.7.....	159
Figure 6-3 — TMSC pin utilization flow diagram for a T4 TAP.7 .....	160
Figure 6-4 — CPA state entry from the <i>BPA</i> state with subsequent Online operation.....	161
Figure 6-5 — CPA state entry from the <i>SPA</i> state with subsequent Online operation .....	162
Figure 6-6 — CPA state entry from the <i>BPA</i> state with subsequent Offline operation .....	162
Figure 6-7 — CPA state entry from the <i>SPA</i> state with subsequent Offline operation.....	163
Figure 6-8 — Reset initiated CPA state entry/awaiting a Selection Sequence for an exit.....	163
Figure 6-9 — SP and TAPC state relationships .....	164
Figure 6-10 — Expanded view of an SP sequence.....	164
Figure 6-11 — Serialization and de-serialization of SP information.....	165
Figure 6-12 — High-level T4 TAP.7 functional block diagram .....	166
Figure 6-13 — T5 TAP.7 chip block diagram.....	167
Figure 6-14 — Logical channel configurations.....	168
Figure 6-15 — DTS data exchanges with one client .....	169
Figure 6-16 — DTS data exchanges with more than one client .....	170
Figure 6-17 — DTS data exchange between multiple clients .....	171
Figure 6-18 — Conceptual block diagram of the functions of T5 TAP.7 .....	172
Figure 6-19 — TMSC pin utilization flow diagram for a T5 TAP.7 .....	173
Figure 6-20 — TP placement and content.....	174
Figure 6-21 — High-level T5 TAP.7 functional block diagram .....	175
Figure 7-1 — System connectivity with mixed use of technologies and scan topologies .....	178
Figure 7-2 — TAP.1 Series Branches .....	179
Figure 7-3 — TAP.7 Series, Star-4, and Star-2 Scan Topologies .....	180
Figure 7-4 — TAP.7 TAPC hierarchy.....	181
Figure 7-5 — System connectivity supported by the TAP.7 architecture .....	182
Figure 7-6 — Contrasting IEEE 1149.1 and IEEE 1149.7 test views .....	183
Figure 8-1 — Parking the ADTAPC state .....	189
Figure 8-2 — Parking the Chip-Level TAPC state.....	190
Figure 8-3 — Parking the Chip-Level TAPC state.....	192
Figure 8-4 — Selection within a chip or system .....	193
Figure 8-5 — Example of debug use of the RSU .....	194
Figure 8-6 — Using Series/Star-4/and Star-2 Scan Topologies concurrently .....	195
Figure 9-1 — DR Scan sequence used for command creation .....	198
Figure 9-2 — Command completion timing .....	204
Figure 9-3 — Command construction and execution .....	205
Figure 9-4 — Conceptual to physical path mapping paths.....	211
Figure 9-5 — Conceptual view of the T1 and above TAP.7 Scan Paths .....	212
Figure 9-6 — Saved Scan Group Candidacy/ZBS-count and scan path selection relationships .....	213
Figure 9-7 — Conceptual view of a String Scan Path that supports reads and writes.....	217
Figure 9-8 — Example Series Scan Path with a String Path selected in a chip of interest.....	218
Figure 9-9 — Example of DTS-generated string-write data for three trailing bits.....	218
Figure 9-10 — String Scan Path characteristics in a Series Scan Topology .....	219
Figure 9-11 — Conceptual view of the Enumerate Scan Path .....	219
Figure 9-12 — Enumerate Scan Path characteristics in a Star-4 Scan Topology .....	225
Figure 9-13 — Conceptual view of command processing.....	232
Figure 9-14 — Conceptual view of command processor .....	233
Figure 9-15 — Conceptual view of EPU Scan Paths .....	234
Figure 10-1 — Conceptual view of the EPU reset logic and state machines .....	235

Figure 10-2 — Reset State Machine.....	239
Figure 10-3 — Conceptual view of reset function .....	240
Figure 10-4 — Test Reset setup time .....	242
Figure 10-5 — A Type-0 through Type-2 Reset invoking the use of the Standard Protocol .....	247
Figure 10-6 — A Type-3 Reset invoking the use of the Standard Protocol .....	248
Figure 10-7 — A Type-4 Reset invoking the use of the Standard Protocol .....	249
Figure 10-8 — Type-0 through Type-2 Resets invoking Offline-at-Start-up operation.....	249
Figure 10-9 — A Type-3 Reset invoking Offline-at-Start-up operation .....	250
Figure 10-10 — Escape Detection.....	252
Figure 10-11 — Selection Escape followed by a Selection Sequence .....	253
Figure 10-12 — Conceptual view of the Escape Detection function .....	254
Figure 10-13 — Selection-Alert Bit Sequence .....	258
Figure 10-14 — Selection Alert LFSR for bit sequence generation.....	258
Figure 10-15 — TMSC delay options .....	259
Figure 10-16 — Selection Alert support logic.....	260
Figure 10-17 — Deselection Alert operation .....	262
Figure 10-18 — Selection Alert with embedded Selection Escape .....	264
Figure 10-19 — Falling-edge ADTAPC State Machine.....	265
Figure 11-1 — Conceptual view of technology selection mechanism .....	267
Figure 11-2 — Online/Offline operation with a TAP.7 Controller .....	268
Figure 11-3 — Online/Offline operation with another technology/RSU with Escapes.....	270
Figure 11-4 — Online/Offline operation with another technology/RSU with Alerts.....	270
Figure 11-5 — Offline initiation events .....	271
Figure 11-6 — Online initiation events .....	273
Figure 11-7 — TAP.7 Selection Sequence.....	276
Figure 11-8 — Conceptual view of TAP.7 Controller Selection Sequence .....	277
Figure 11-9 — Selection Escape/subsequent data timing relationship.....	280
Figure 11-10 — Selection Alert/subsequent data timing relationship .....	281
Figure 11-11 — Serial load and initialization of Global Registers.....	283
Figure 11-12 — CP format.....	285
Figure 11-13 — TAPC parking-state creation with a deselection Escape/Standard Protocol .....	288
Figure 11-14 — TAPC parking-state creation with a Selection Escape/Standard Protocol .....	288
Figure 11-15 — TAPC parking state/Selection Escape/the Advanced Protocol/relationship .....	289
Figure 11-16 — TAPC parking state/Deselection Escape/the Advanced Protocol/relationship .....	289
Figure 11-17 — Conceptual view of the Control State Machine .....	291
Figure 11-18 — CSM TEST state function.....	298
Figure 11-19 — Substates of the CSM CHK state .....	303
Figure 11-20 — A CP_NOP Directive initiated extension of the CP Body Element.....	303
Figure 11-21 — A CP_END Directive followed by Offline operation .....	304
Figure 11-22 — A CP_END Directive followed by the use of the Standard Protocol.....	304
Figure 11-23 — A CP_END Directive followed by the use of the Advanced Protocol.....	305
Figure 11-24 — A CP_RSO Directive initiating a Type 3 TAP.7 Controller reset.....	305
Figure 11-25 — CP template.....	306
Figure 11-26 — OLS Selection Escape qualification sequence.....	309
Figure 11-27 — Simultaneous Selection Escape qualification and TAPC state initialization .....	310
Figure 11-28 — Conceptual view of the CSM with all sub-states exposed .....	312
Figure 11-29 — Counter supporting function of the OLS state .....	313
Figure 12-1 — Conceptual view of TAP.7 signal functions .....	319
Figure 12-2 — Conceptual view of bias for TAP.7 input and input/output signal functions .....	320
Figure 12-3 — Online-at-Start-up Test Mode Select/signal bias relationships .....	324
Figure 12-4 — Offline-at-Start-up Test Mode Select/signal bias relationships .....	325
Figure 12-5 — A Type-0-Type-2 Reset initiating Offline-at-Start-up operation.....	326
Figure 12-6 — Test Mode Select signal bias changes.....	329
Figure 12-7 — Test Data Input/signal bias relationships for legacy operation .....	331
Figure 12-8 — Test Data Output/signal bias relationships.....	334
Figure 13-1 — TDO(C) Drive Policy/factor relationships .....	343

Figure 13-2 — Hierarchical view of the TDO(C) Drive Policy .....	348
Figure 13-3 — Flattened view of the TDO(C) Drive Policy .....	349
Figure 13-4 — Conceptual view of TDOC Drive Policy .....	350
Figure 13-5 — T0 TAP.7 TDO Drive Policy .....	351
Figure 13-6 — T1-T2 TAP.7 TDO Drive Policy.....	353
Figure 13-7 — T3, T4(W), and T5(W) TAP.7 TDOC Drive Policies.....	355
Figure 13-8 — Conceptual view of the Scan Group Candidate Count operation.....	364
Figure 13-9 — SGCC and PSGMCL timing relationships .....	367
Figure 13-10 — SSD State/Delayed SSD State/timing relationships.....	369
Figure 13-11 — Conceptual view of the Conditional Group Membership Count operation.....	372
Figure 13-12 — Only Conditional Group Member determination .....	373
Figure 13-13 — Example of TDOC Drive Policy for a T3 and above TAP.7 .....	376
Figure 13-14 — Conceptual view of SGCC and PSGMCL State Machines.....	378
Figure 13-15 — Determining Scan Group Only Member Last/Membership Count Last .....	378
Figure 13-16 — Conceptual view of CGMC state development.....	379
Figure 14-1 — SP content preview .....	380
Figure 14-2 — SP payload template.....	381
Figure 14-3 — Transport Packet content preview.....	382
Figure 14-4 — TMSC Advanced Protocol drive characteristics .....	383
Figure 14-5 — The TMSC Signal's Distributed Drive Policy .....	384
Figure 14-6 — Precharge Bit Drive Policy .....	386
Figure 14-7 — RDY Bit Drive Policy.....	388
Figure 14-8 — CLTAPC selection state/STL sourced output value relationships .....	390
Figure 14-9 — Correlation of TDO(C) and TDO Bit Drive Policies .....	392
Figure 14-10 — TDO Bit Drive Policy .....	393
Figure 14-11 — Transport Bit Drive Policy.....	395
Figure 14-12 — Conceptual view of TMSC signal output and miscellaneous support logic.....	397
Figure 14-13 — Conceptual view of TMSC data and drive enables .....	398
Figure 15-1 — Inclusion and exclusion of the STL from the scan path.....	402
Figure 15-2 — Conceptual view of the operation of an EMTAPC .....	403
Figure 16-1 — Deployment of the T0 TAP.7.....	405
Figure 16-2 — SoC with IEEE 1149.7 multiple TAPC architecture.....	406
Figure 16-3 — Operating mode changes/ <i>Run-Test/Idle</i> state relationships.....	409
Figure 16-4 — Example IEEE 1149.7 multi-TAP architecture with IR-controlled TAPC exclusion .....	410
Figure 16-5 — IR Scan Path after the CLTAPC has exited the <i>Test-Logic-Reset</i> state and entered the <i>Run-Test/Idle</i> state.....	411
Figure 16-6 — DR Scan Path with the <i>IDCODE</i> instruction loaded in the CLTAPC IR .....	411
Figure 16-7 — DR selection with the <i>TAPC_SELECT_NNN</i> instruction loaded in the CLTAPC IR.....	412
Figure 16-8 — Example IEEE 1149.7 multi-TAP architecture to allow isolating of individual TAPCs under CLTAPC IR control .....	413
Figure 16-9 — Example IEEE 1149.7 multi-TAP architecture with DR-controlled TAPC exclusion or isolation .....	415
Figure 16-10 — Example IEEE 1149.7 multi-TAP architecture to allow control over the operating mode of individual TAPCs by using <i>tapc_select</i> chip signals.....	416
Figure 16-11 — Conceptual view of the <i>TAP_QUERY</i> instruction .....	417
Figure 16-12 — Example system .....	418
Figure 16-13 — Example of the active <i>BYPASS</i> instruction .....	423
Figure 16-14 — Package-level connectivity of the POR* and nTRST signals .....	425
Figure 16-15 — The DR-wire solution in the SiP-TAP approach.....	426
Figure 17-1 — Conceptual view of zero bit DR Scan detection .....	430
Figure 17-2 — Conceptual view of ZBS count locking .....	431
Figure 17-3 — TAPC-state/control-state and ZBS-count relationships .....	433
Figure 17-4 — EPU Operating States .....	433
Figure 17-5 — ZBS use state and ZBS-count control flow.....	436
Figure 17-6 — Conceptual view of the ZBS use states with both TAP.7 and other ZBS use .....	438
Figure 18-1 — Conceptual view of a T1 TAP.7.....	441

Figure 18-2 — Deployment of the T1 TAP.7.....	442
Figure 18-3 — Conceptual view of <i>nsys_trst</i> and <i>sys_tms</i> generation .....	449
Figure 18-4 — Conceptual view of functional reset request generation .....	452
Figure 18-5 — Conceptual view of function reset request timing/system-initiated clear .....	452
Figure 18-6 — Conceptual view of function reset request timing with DTS-initiated clear .....	453
Figure 18-7 — Components involved in TAP.7 Controller power management .....	457
Figure 18-8 — TAP.7 Controller power-management state progression .....	459
Figure 18-9 — TAP.7 power-manager operation .....	461
Figure 18-10 — Conceptual view of the TAP.7 power manager .....	462
Figure 18-11 — Typical power-mode interface .....	463
Figure 18-12 — Directed power-up of a TAP.7 Controller.....	464
Figure 18-13 — Detected power-up of a TAP.7 Controller .....	465
Figure 18-14 — Conceptual operation of TAP.7 Controller power control .....	466
Figure 18-15 — TAP.7 Controller power-management responsibilities .....	466
Figure 18-16 — Power-down criteria selection.....	467
Figure 18-17 — Mode 2 – TAP.7 Controller and STL shutdown .....	469
Figure 18-18 — Mode 0 – with a TCK TIMEOUT initiating power-down .....	470
Figure 18-19 — Mode 1 – with the <i>Test-Logic-Reset</i> state and TCK TIMEOUT initiating power-down ..	470
Figure 18-20 — Mode 2—with the <i>Test-Logic-Reset</i> state initiating power-down.....	470
Figure 18-21 — Power-down logic example.....	471
Figure 18-22 — Power-down request generation flowchart.....	473
Figure 18-23 — Power-down criteria selection.....	474
Figure 18-24 — Default Power-Control Mode.....	474
Figure 18-25 — A PD_STOP initiating a Type-3 TAP.7 Controller reset.....	476
Figure 19-1 — Typical T2 TAP.7 .....	479
Figure 19-2 — Deployment of the T2 TAP.7 .....	480
Figure 19-3 — <i>Run-Test/Idle</i> /Control Path/Scan Group Candidacy relationships .....	486
Figure 19-4 — Skewed <i>Run-Test/Idle</i> /Control Path/Scan Group Candidacy relationships.....	487
Figure 19-5 — TAP.7 Controller reset creating <i>Test-Logic-Reset</i> parking state .....	488
Figure 19-6 — Re-synchronization of ADTAPC and CLTAPC states following assertion of <i>nsys_trst</i> .....	489
Figure 19-7 — Conceptual view of T2 TAP.7 Scan Group Membership management .....	490
Figure 19-8 — Conceptual view of T2 TAP.7 current Scan Group Candidacy development.....	490
Figure 19-9 — CLTAPC selection state change/ <i>Run-Test-Idle_f</i> /register relationships .....	492
Figure 19-10 — CLTAPC selection state change/ <i>Test-Logic-Reset_f</i> .....	492
Figure 20-1 — T3 TAP.7 .....	496
Figure 20-2 — Deployment of the T3 TAP.7 .....	497
Figure 20-3 — TDOC output with the JScan3 Scan Format within the CR Scan .....	507
Figure 20-4 — CID allocation with a DTS-sourced AT .....	508
Figure 20-5 — CID allocation with a TS-sourced AT .....	508
Figure 20-6 — Conceptual view of CID and CIDI Registers.....	509
Figure 20-7 — Conceptual view of an SSD .....	513
Figure 20-8 — SSDs detection and completion regions.....	515
Figure 20-9 — Conceptual view of SSD Processing State Machine .....	516
Figure 20-10 — SSD Interlocking State Machine.....	518
Figure 20-11 — SSD detection and completion window .....	521
Figure 20-12 — SSDs changing the Scan Group Membership in the <i>RTI</i> state .....	522
Figure 20-13 — SSD_SA ignored because of early exit from the <i>RTI</i> state .....	523
Figure 20-14 — SSDs changing the Pause Selection state .....	524
Figure 20-15 — SSD_SA ignored in the <i>Pause-xR</i> state .....	525
Figure 20-16 — Conceptual view of SSD processing .....	526
Figure 20-17 — Conceptual view of SSD state and SSD state delayed .....	526
Figure 20-18 — Conceptual view of TCA and CID comparison .....	527
Figure 20-19 — TMS and reset value required for SSD to perform its designated function .....	529
Figure 20-20 — SSD effects .....	531
Figure 20-21 — Scan topology characteristics used in Scan Topology Training.....	534

Figure 20-22 — Conceptual view of current Scan Group Candidacy development—T3 and above TAP.7s .....	537
Figure 20-23 — Current Scan Group Candidacy development T3 and above TAP.7s .....	537
Figure 20-24 — Group membership operation.....	539
Figure 20-25 — Saved Scan Group Candidacy operation.....	540
Figure 20-26 — <i>sys_tck</i> gating/Scan Group Membership change relationships .....	540
Figure 21-1 — T4 and above TAP.7 chip block diagram.....	542
Figure 21-2 — Conceptual view of fully deployed APU functionality .....	543
Figure 21-3 — Conceptual block diagram of the APU and RSU functions .....	546
Figure 21-4 — APU/EPU interface.....	551
Figure 21-5 — APU/DCC(s) interface.....	552
Figure 21-6 — Conceptual view of TAP.7 use with a T5 TAP.7 Controller .....	553
Figure 21-7 — APU functions/APU Operating State relationships .....	554
Figure 21-8 — Operating state sequence examples.....	556
Figure 21-9 — Typical DTS-initiated packet sequences and APU state transitions .....	558
Figure 21-10 — Packet combinations used to advance the TAPC state.....	560
Figure 21-11 — Conceptual view of packet scheduling.....	563
Figure 21-12 — Example of scheduling and sequencing of state machine activity .....	565
Figure 21-13 — Advanced operation function/description relationships .....	566
Figure 22-1 — Conceptual view of a Scan Packet .....	568
Figure 22-2 — Scan exchange types .....	571
Figure 22-3 — Voting bit transaction.....	572
Figure 22-4 — Initiator/Responder bit transaction.....	572
Figure 22-5 — Scripted scan exchange .....	572
Figure 22-6 — Conceptual view of a TP .....	574
Figure 22-7 — TP placement and content .....	575
Figure 22-8 — APU state diagram .....	576
Figure 22-9 — Use of the virtual TAPC state .....	577
Figure 22-10 — Conceptual view of a Scan Packet Element scheduling.....	578
Figure 23-1 — Deployment of the T4 TAP.7.....	580
Figure 23-2 — Advanced Scan Format overview .....	588
Figure 23-3 — TAP.7 data path with positive- and negative-edge clocking.....	593
Figure 23-4 — The <i>sys_tck</i> signal .....	594
Figure 23-5 — Example of the MScan Scan Format with <i>sys_tck</i> signal duty-cycle options .....	594
Figure 23-6 — <i>sys_tck</i> signal logic 1 time .....	595
Figure 23-7 — Conceptual view of APFC operation .....	596
Figure 23-8 — SREdge/tmsc_mux/tmsc_r relationships .....	597
Figure 23-9 — Conceptual view of <i>sys_tms</i> generation for a T4 and above TAP.7.....	597
Figure 23-10 — Conceptual view of TMS generation with the Advanced Protocol.....	598
Figure 23-11 — Conceptual view of <i>sys_tdi</i> signal generation for a T4 and above TAP.7.....	600
Figure 23-12 — Conceptual view of System Ready treatment in its simplest form.....	601
Figure 23-13 — Conceptual <i>sys_rdy</i> generation for more complex use cases .....	602
Figure 23-14 — Conceptual view of TDO data generation for output via the TMSC signal .....	603
Figure 23-15 — Using the TMSC value in next state equations .....	606
Figure 24-1 — Application types supported by the MScan Scan Format .....	608
Figure 24-2 — High-level operation of the MScan Scan Format.....	608
Figure 24-3 — Scan Packet Elements with the MScan Scan Format .....	609
Figure 24-4 — MScan payload template.....	610
Figure 24-5 — MScan payload/EPU input and output relationships.....	610
Figure 24-6 — MScan output bit-frame .....	611
Figure 24-7 — MScan output bit-frame with stalled completion.....	611
Figure 24-8 — Typical TCKC and TMSC signal system delays .....	612
Figure 24-9 — Use of [RDY] bit(s) to accommodate propagation delays .....	612
Figure 24-10 — Delay Element format .....	614
Figure 24-11 — MScan SPs with Delay Elements.....	615
Figure 24-12 — Type-3 Reset generated by a variable-length Delay Element .....	616

Figure 24-13 — MScan SP payload/TAPC state advance relationship.....	617
Figure 24-14 — CID allocation with MScan SPs.....	618
Figure 24-15 — Conceptual view of MScan SP Payload Element behavior.....	620
Figure 24-16 — Conceptual view of the Payload State Machine for the MScan Scan Format.....	621
Figure 24-17 — Conceptual view of Ready State Machine MScan operation.....	622
Figure 24-18 — Conceptual Delay Element operation.....	622
Figure 24-19 — Conceptual view of the Delay State Machine .....	623
Figure 25-1 — Application types supported by OScan Scan Formats .....	625
Figure 25-2 — High-level operation of the OScan Scan Formats .....	626
Figure 25-3 — Scan Packet Elements with the OScan Scan Format.....	626
Figure 25-4 — OScan payload template .....	627
Figure 25-5 — Payloads for <i>Shift-xR</i> and non- <i>Shift-xR</i> TAPC states.....	628
Figure 25-6 — EPU input/output relationships with an nTDI, TMS, RDY, and TDO SP payload .....	630
Figure 25-7 — EPU input/output relationships with a TMS SP payload .....	630
Figure 25-8 — EPU input/output relationships with an nTDI, RDY, and TDO SP payload.....	631
Figure 25-9 — EPU input/output relationships with an nTDI SP payload.....	631
Figure 25-10 — EPU input/output relationships with an nTDI/TDO SP payload .....	632
Figure 25-11 — TMS generation with an EOT Escape.....	632
Figure 25-12 — OScan6 PAD bit and its function.....	633
Figure 25-13 — OScan output bit-frame.....	634
Figure 25-14 — OScan/SScan bit-frame with stalled completion.....	635
Figure 25-15 — Using delays with the OScan Scan Formats .....	637
Figure 25-16 — OScan6 SP payload/TAPC state advance relationship .....	638
Figure 25-17 — Virtual TAPC state with OScan6 Scan Format.....	639
Figure 25-18 — Example with the <i>sys_tck</i> signal duty-cycle options with OScan1 .....	641
Figure 25-19 — Conceptual view of OScan SP Payload Element behavior .....	642
Figure 25-20 — Conceptual view of Payload State Machine entry point generation.....	643
Figure 25-21 — Shift Progress flag for OScan6 support.....	643
Figure 25-22 — CSM and SSM activation.....	644
Figure 25-23 — Conceptual view of Ready State Machine OScan/SScan operation.....	644
Figure 25-24 — Conceptual view of TAP advance with MScan/OScan Scan Formats .....	645
Figure 26-1 — Application types supported by SScan Scan Formats.....	647
Figure 26-2 — Stall profile/application relationships .....	650
Figure 26-3 — SScan segment use cases .....	651
Figure 26-4 — SScan transactions with a TAP.1-like component .....	652
Figure 26-5 — SScan transactions with a DMA or FIFO .....	652
Figure 26-6 — SScan transactions with an FIFO .....	653
Figure 26-7 — SScan transactions with rate-dependent components.....	653
Figure 26-8 — SScan transactions with a buffered rate-dependent component.....	654
Figure 26-9 — Stall profile combinations .....	654
Figure 26-10 — Scan Packet Elements with the SScan Scan Formats.....	656
Figure 26-11 — Header Element of an SP .....	657
Figure 26-12 — SP payloads for <i>Shift-xR</i> and non- <i>Shift-xR</i> TAPC states.....	659
Figure 26-13 — Segment composition.....	663
Figure 26-14 — SScan input-only and input/output Data Segment processing .....	664
Figure 26-15 — SScan0/SScan1 output-only Data Segment processing .....	667
Figure 26-16 — SScan2/3 output-only Data Segment processing .....	668
Figure 26-17 — TDO pipelining with the SScan2/SScan3 Scan Formats.....	669
Figure 26-18 — CLTAP TDO pipelining with SScan2/SScan3 Scan Formats/with stalls .....	670
Figure 26-19 — Using delays with the SScan Scan Formats .....	674
Figure 26-20 — Packets following an SP associated with the <i>Update-DR</i> and other states .....	675
Figure 26-21 — Packets following an SP associated with the <i>Capture-xR/Exit2-xR</i> states .....	676
Figure 26-22 — SScan0/1 TAPC state advance examples .....	678
Figure 26-23 — SScan2/3 TAPC state advance examples for I/O and input-only segments .....	679
Figure 26-24 — SScan2/3 TAPC state advance examples for output-only segments .....	680
Figure 26-25 — Conceptual view of the Payload State Machine with Header handling .....	682

Figure 26-26 — Conceptual view of the Escape Detection State Machine .....	684
Figure 26-27 — SScan Shift Progress flag .....	685
Figure 26-28 — Conceptual view of TAP advance with MScan/OScan/SScan Scan Formats .....	685
Figure 26-29 — SScan entry point reload additions for output only .....	686
Figure 26-30 — Conceptual view of Header Register .....	686
Figure 26-31 — Conceptual view of Payload State Machine entry point with SScan support .....	687
Figure 26-32 — Joint operation of the state machines .....	688
Figure 26-33 — Joint operation of state machines with Command Part Two <i>Update-DR</i> .....	688
Figure 27-1 — Conceptual view of T5 TAP.7 Transport Function .....	690
Figure 27-2 — Deployment of a T5 TAP.7 Controller .....	691
Figure 27-3 — Conceptual view of T5 TAP.7 Controller configurations .....	699
Figure 27-4 — TP placement and content .....	703
Figure 27-5 — Transport Packet Elements .....	704
Figure 27-6 — TP Directive Elements and their surrounding context .....	706
Figure 27-7 — Directive/transport building block relationships .....	707
Figure 27-8 — Summary of Register Element characteristics .....	715
Figure 27-9 — Summary of Data Element characteristics .....	718
Figure 27-10 — Selection mechanisms for data and register transfers .....	720
Figure 27-11 — Operation of the DCCy_ENAR and DCCy_ENAA register bits .....	720
Figure 27-12 — Operation of the DCCy_DCNR and DCCy_DRNA register bits .....	721
Figure 27-13 — Conceptual view of T5 TAP.7 transport control .....	723
Figure 27-14 — Conceptual view of transport control facilitating a data exchange .....	724
Figure 28-1 — <i>TPA</i> state behaviors .....	728
Figure 28-2 — <i>TPA</i> state flow .....	729
Figure 28-3 — TP Directive Element format, timing, and TMSC signal drive characteristics .....	730
Figure 28-4 — A Type-3 Reset generated by the TP_RSZ and TP_RSO Directives .....	730
Figure 28-5 — TP Register Element format, timing, and TMSC signal drive characteristics .....	731
Figure 28-6 — Register Element with surrounding TP Directives .....	732
Figure 28-7 — TP Data Element format, timing, and TMSC signal drive characteristics .....	735
Figure 28-8 — Fixed-length Data Payload with surrounding TP Directives .....	735
Figure 28-9 — Variable-length Data Payload with surrounding headers .....	736
Figure 28-10 — DCCy_DIRA Register/Data Payload content relationships .....	737
Figure 28-11 — Data payload and data channel data alignment relationship .....	738
Figure 28-12 — Conceptual view of the Transport State Machine .....	740
Figure 28-13 — Queuing a TP to follow an SP .....	742
Figure 28-14 — Queuing a TP to follow an SP with the OScan2 Scan Format .....	743
Figure 28-15 — Queuing a TP to follow an SP with the OScan1 Scan Format .....	744
Figure 28-16 — Starting directive processing/ending TP processing .....	745
Figure 28-17 — PDCx/DCC interface signals with a single DCC connected to the DCC .....	746
Figure 28-18 — PDCx/DCC interface signals with multiple DCCs sharing the PDC .....	747
Figure 28-19 — Conceptual view of T5 TAP.7 Transport Control .....	750
Figure 28-20 — PDCx/DCC interface signaling/TP_NOP and TP_END Directives .....	754
Figure 28-21 — DCC interface signaling/TP_DCx Directives .....	755
Figure 28-22 — PDCx/DCC interface signaling/nonterminating eight-bit directives .....	756
Figure 28-23 — PDCx/DCC interface signaling/TP_RSO and TP_RSZ Directives .....	757
Figure 28-24 — PDCx/DCC interface signaling/12-bit directives other than TP_CRR/TP_CRW .....	758
Figure 28-25 — PDCx/DCC interface signaling with the TP_CRR Directive .....	759
Figure 28-26 — PDCx/DCC interface signaling with the TP_CRW Directive .....	760
Figure 28-27 — DCC interface signaling required for Custom Operation .....	761
Figure 28-28 — Pipelining data transfers .....	761
Figure 28-29 — Conceptual view of the Transport Function .....	762
Figure 28-30 — Example Transport State Machine .....	763
Figure 28-31 — Behavior of the <i>next_dir</i> signal .....	764
Figure 28-32 — Multi-use register bit input configurations .....	765
Figure 28-33 — Multi-use Register values/TSM relationships .....	767
Figure 28-34 — Transport drive and DCR input selection .....	771

Figure 28-35 — TP with fixed-length data transfer with nonpipelined scan operations .....	773
Figure 28-36 — TP with fixed-length data transfer with pipelined scan operations .....	774
Figure 28-37 — TP with continuous data transfer .....	775
Figure 28-38 — TP with no data transferred .....	776
Figure 29-1 — Typical test application – testing component interconnections.....	779
Figure 29-2 — Unit-under-test description .....	784
Figure 31-1 — Serial connection using one TMS signal.....	822
Figure 31-2 — Connection in two paralleled serial chains.....	824
Figure 31-3 — Basic Star Scan Topology .....	825
Figure 31-4 — Basic hierarchical topology.....	826
Figure A-1 — IEEE 1149.1 TAPC State Machine .....	830
Figure A-2 — IEEE 1149.1 signaling .....	831
Figure A-3 — Zero-bit DR Scan state sequences, path A .....	832
Figure A-4 — Zero-bit DR Scan state sequences, path B .....	833
Figure B-1 — MScan and OScan SP content .....	834
Figure B-2 — MScan transaction .....	836
Figure B-3 — OScan0 transaction.....	837
Figure B-4 — OScan1 transaction.....	838
Figure B-5 — OScan2 transaction.....	839
Figure B-6 — OScan3 transaction.....	840
Figure B-7 — OScan4 transaction .....	841
Figure B-8 — OScan5 transaction.....	842
Figure B-9 — OScan6 transaction.....	844
Figure B-10 — OScan7 transaction.....	845
Figure B-11 — SScan0 transaction/first SP stall profile .....	848
Figure B-12 — SScan0 transaction/all stall profile .....	849
Figure B-13 — SScan1 transaction/first SP stall profile .....	851
Figure B-14 — SScan1 transaction/no stall profile .....	852
Figure B-15 — SScan2 transaction/first SP stall profile .....	854
Figure B-16 — SScan2 transaction/all stall profile .....	855
Figure B-17 — SScan3 transaction/first stall profile.....	857
Figure B-18 — SScan3 transaction/no stall profile .....	858
Figure B-19 — OScan1 Scan Format/TPs with fixed-length Data Element following SPs .....	859
Figure B-20 — OScan7 Scan Format/TPs with fixed-length Data Element following SPs .....	859
Figure B-21 — OScan1 Scan Format/TPs with variable-length Data Element following SPs .....	860
Figure B-22 — OScan7 Scan Format/TPs with variable-length Data Element following SPs .....	860
Figure D-1 — Power-control states.....	905
Figure D-2 — Consolidated view of CSM states .....	906
Figure D-3 — TAP initialization and topology determination.....	908
Figure D-4 — TS initialization.....	910
Figure D-5 — Scan Topology Training.....	915
Figure D-6 — Branch interrogation .....	916
Figure D-7 — Series Branch interrogation.....	919
Figure D-8 — Reads of CNFG0–CNFG3 Registers in a Series Scan Topology .....	920
Figure D-9 — Determining the type/class of series TAPs .....	922
Figure D-10 — Star-4 Branch interrogation—part one .....	924
Figure D-11 — Star-4 Branch interrogation—part two.....	925
Figure D-12 — Star-2 Branch interrogation—part one .....	927
Figure D-13 — Star-2 Branch interrogation—part two.....	928
Figure E-1 — Electrical characteristics .....	932
Figure F-1 — Typical TCK(C) frequency versus number of TAPs .....	935
Figure F-2 — Connection topologies .....	936
Figure F-3 — Series termination scheme .....	937
Figure F-4 — Parallel termination schemes .....	937
Figure F-5 — The use of input buffers with hysteresis .....	938
Figure F-6 — Signal filtering using a one shot .....	938

Figure F-7 — Model of circuit used to compare the rise/fall time effects on signal quality .....	940
Figure F-8 — Overshoot and undershoot with the output driver's rise/fall time at 0.5 ns .....	940
Figure F-9 — Overshoot and undershoot with the output driver's rise/fall time at 1.5 ns .....	941
Figure F-10 — Lumped load .....	942
Figure F-11 — Distributed load .....	942
Figure F-12 — Series termination with impedance mismatch causing under/overshoots .....	945
Figure F-13 — Series termination with impedance mismatch causing monotonicity issues.....	946
Figure F-14 — Proper connection of TAP signaling on a board a Line configuration.....	948
Figure F-15 — Proper connection of TAP signaling on a board and T configuration .....	948
Figure F-16 — Proper connection of TAP signaling on a board and X configuration .....	948
Figure F-17 — Improper connection of TAP signaling on a board with parallel termination.....	949
Figure F-18 — Proper connection of TAP signaling with more than one connection.....	949
Figure F-19 — Basic series termination waveform.....	950
Figure F-20 — Simple parallel DC termination .....	951
Figure F-21 — Ideal parallel DC termination .....	952
Figure F-22 — Parallel DC termination, 2 ns edge-rates, four 3 picofarad loads .....	953
Figure F-23 — Realistic parallel termination model .....	953
Figure F-24 — Realistic AC parallel termination model.....	954
Figure F-25 — Ideal parallel AC termination, No Load .....	955
Figure F-26 — Parallel AC termination model with one load.....	955
Figure F-27 — Parallel AC termination simulation results .....	956
Figure F-28 — Basic parallel termination waveform with capacitive load isolation .....	958
Figure F-29 — DTS and chip models.....	959
Figure F-30 — Topologies with a single device providing K bias .....	960
Figure F-31 — Hierarchical Package with TAPC and K Bias at Chip Level .....	961
Figure F-32 — Hierarchical package with top-level I/O management.....	962
Figure F-33 — Multi-TAPC package with shared I/O pins .....	963
Figure F-34 — Programmable DTS source impedance.....	964
Figure F-35 — DTS input capacitance load isolation .....	964
Figure F-36 — Emitter-follower driver.....	965
Figure F-37 — Voltage translator .....	966
Figure F-38 — Point-to-Point transmission-line model .....	968
Figure F-39 — Point-to-Point/series termination .....	968
Figure F-40 — One source/one destination/unidirectional signaling/parallel AC termination .....	969
Figure F-41 — Single source/destination/bidirectional signaling/series termination .....	969
Figure F-42 — Bidirectional scenario—DTS drives/chip/TAPC receives.....	970
Figure F-43 — Bidirectional scenario—Chip/TAPC drives, DTS receives .....	970
Figure F-44 — Line configuration of a transmission line with series termination .....	971
Figure F-45 — Model for a line transmission line with 1 to 16 TAPs .....	972
Figure F-46 — Line configuration, two loads—25 mm intervals, 50 mm DTC trace.....	974
Figure F-47 — Line configuration, four loads—25 mm intervals, 50 mm DTC trace .....	975
Figure F-48 — Line configuration, eight loads—25 mm intervals, 50 mm DTC trace.....	976
Figure F-49 — Line configuration, 16 loads—25 mm intervals, 50 mm DTC trace.....	977
Figure F-50 — Multiple TAP connection to a Line configuration .....	978
Figure F-51 — Parallel AC termination, one load at end of transmission line .....	979
Figure F-52 — Parallel AC termination, four loads at 25 mm intervals .....	979
Figure F-53 — Bidirectional scenario—chip/TAPC drives, DTS receives, multiple TAPCs .....	981
Figure F-54 — Bidirectional multiple TAPCs, 100 MHz, $R_{ISO} = 10 \Omega$ .....	982
Figure F-55 — Bidirectional multiple TAPCs, 100 MHz, $R_{ISO} = 50 \Omega$ .....	982
Figure F-56 — Line configuration, bidirectional multiple TAPCs, 40 MHz, $R_{ISO} = 50 \Omega$ .....	983
Figure F-57 — Line configuration, 4 TAPs, C2 and C3 drive, 50 MHz .....	984
Figure F-58 — Line configuration 4 TAPs, C2 and C4 drive, 50 MHz .....	985
Figure F-59 — Line configuration, four TAPs, four drives, 50 MHz .....	986
Figure F-60 — Line configuration: 8 TAPs, two drives (C1 and C6 drive), 50 MHz.....	987
Figure F-61 — Line configuration: 8 TAPs, two drives (C2 and C4), 50 MHz.....	988
Figure F-62 — Line configuration: 8 TAPs, two drives (C2 and C6), 50 MHz.....	989

Figure F-63 — Logical view of T configuration of a transmission line .....	990
Figure F-64 — T model with parallel DTS drivers and two nodes/leg .....	991
Figure F-65 — Impedance profile of the T configuration .....	992
Figure F-66 — T configuration: 4 TAPs, DTS drive, 50 MHz .....	993
Figure F-67 — T configuration, 4 TAPs, U1A drives, 50 MHz.....	994
Figure F-68 — T configuration, 4 TAPs, one drive (U1B), 50 MHz .....	995
Figure F-69 — T configuration, 4 TAPs, two drives (U1A and U1B), 50 MHz.....	996
Figure F-70 — T configuration, 4 TAPs, two drives (U1B and U1D drive), 50 MHz.....	997
Figure F-71 — Logical view of X configuration of a transmission line.....	998
Figure F-72 — X model with parallel DTS drivers and four nodes/leg .....	999
Figure F-73 — Contrasting matched and unmatched source and load impedances .....	1000
Figure F-74 — X configuration, 8 TAPs, DTS drive, 100 MHz, at destination .....	1001
Figure F-75 — X configuration. 8 TAPs, DTS drive, 50 MHz .....	1002
Figure F-76 — X configuration, 8 TAPs, two TAPs/leg, one drive (U1A), 50 MHz.....	1003
Figure F-77 — X configuration, 8 TAPs, two TAPs/leg, one drive (U1B), 50 MHz.....	1004
Figure F-78 — X configuration, 8 TAPs, two TAPs/leg, two drives (U1A and U1B), 50 MHz.....	1005
Figure F-79 — X configuration, 8 TAPs, two TAPs/leg, four drives, 50 MHz .....	1006
Figure F-80 — X configuration, 8 TAPs, two TAPs/leg, all drive, 50 MHz.....	1007
Figure F-81 — Logical view of XT configuration of a transmission line .....	1008
Figure F-82 — Impedance profile of the XT configuration .....	1008
Figure F-83 — Bidirectional, balanced delay, 16 TAPCs, detailed model.....	1009
Figure F-84 — XT configuration, 16 TAPs, nodes driven by DTS, 50 MHz, at destination .....	1010
Figure F-85 — XT configuration, DTS drive, 50 MHz, at source .....	1011
Figure F-86 — XT configuration, 16 TAPs, one drives (U1A), 50 MHz.....	1012
Figure F-87 — XT configuration, 16 TAPs, one drives (U1B), 50 MHz.....	1013
Figure F-88 — XT configuration, 16 TAPs, one drive and neighbor, 50 MHz.....	1014
Figure F-89 — XT configuration, 16 TAPs, two drives, 50 MHz.....	1015
Figure F-90 — XT configuration, 16 TAPs, four drives, 50 MHz.....	1016
Figure F-91 — XT configuration, 16 TAPs, all drive, DTS observed, 50 MHz .....	1017
Figure F-92 — Recommended series termination scheme with one to four chips .....	1019
Figure F-93 — Recommended parallel termination scheme with one to four chips .....	1019
Figure F-94 — Recommended series termination scheme for highest performance four/six chips .....	1020
Figure F-95 — Determining signal load classification.....	1020
Figure G-1 — SScan0/SScan2 DMA Transfers .....	1023
Figure G-2 — SScan0/SScan2 transfers with Paced Block.....	1025
Figure G-3 — Example RDY generation for SScan1 or SScan3 Paced-Block transfers .....	1026
Figure G-4 — SScan1/SScan3 mixed Paced-Bit and Paced-Block transfers .....	1026
Figure H-1 — An IEEE 1149.1 TAP with RTCK signal .....	1028
Figure H-2 — Expected behavior for the RTCK and TCK .....	1028
Figure H-3 — Unexpected behavior for RTCK .....	1029
Figure H-4 — RTCK connected to an APU .....	1029
Figure H-5 — RTCK generation for multiple TAPs .....	1030
Figure H-6 — System-wide RTCK behavior .....	1031
Figure H-7 — System-wide RTCK generation logic .....	1031
Figure H-8 — Exclusion of TAP which outputs RTCK2.....	1032
Figure H-9 — Inclusion of TAP which outputs RTCK2 .....	1032
Figure H-10 — RTCK behavior after a TAP.7 Controller power-up .....	1033
Figure H-11 — RTCK behavior with Standard-to-Advanced Protocol change .....	1033
Figure H-12 — RTCK behavior with Advanced-to-Standard Protocol change .....	1034
Figure H-13 — Use of the RTCK signal as an auxiliary function.....	1034

## Tables

Table 1-1 — TAP.7 signal list.....	67
Table 1-2 — TAP.7 Class and scan topology relationships .....	68
Table 1-3 — Relationship TAPC state name alias/TAPC state.....	78
Table 1-4 — Table format.....	79
Table 4-1 — TAPC hierarchy/allowed TAPC parking-state relationships.....	96
Table 4-2 — Selection state terminology summary .....	98
Table 4-3 — Contrast of series and star operation .....	101
Table 4-4 — TAP.7 Class/Escape relationships.....	107
Table 4-5 — TAP.7 Class/Protocol relationships.....	108
Table 6-1 — Events causing TMSC pin utilization changes with a T4 TAP.7 .....	160
Table 6-2 — Events causing TMSC pin utilization changes (with transport) .....	173
Table 6-3 — TAP.7 feature group summary .....	176
Table 7-1 — IEEE 1149.1 and IEEE 1149.7 attributes affecting system architectures.....	177
Table 7-2 — Branch/technology deployment permissibility.....	179
Table 8-1 — TAP.7 Class relationship to selection capability .....	185
Table 8-2 — TAPC parent/child relationships .....	185
Table 8-3 — Selection and deselection states .....	187
Table 9-1 — TAP.7 Controller register list managed commands.....	200
Table 9-2 — Reset values of TAP.7 Controller registers managed with commands.....	201
Table 9-3 — TAP.7 Controller command list .....	202
Table 9-4 — TAP.7 Controller command definition.....	206
Table 9-5 — Command, TAPC State Machine states, and ZBS representations in examples .....	209
Table 9-6 — TAP.7 Scan Path for IR Scans .....	212
Table 9-7 — TAP.7 Scan Path data for DR Scans .....	213
Table 9-8 — EPU Scan Path selection/control level relationships.....	215
Table 9-9 — EPU Scan Path utilization with control level two .....	215
Table 9-10 — Factors causing selection of the bit, string, and auxiliary paths .....	215
Table 9-11 — Scan counts for string operations in a Series Scan Topology.....	218
Table 9-12 — EPU Scan Path characteristics.....	220
Table 9-13 — RDBACK0 Register format .....	227
Table 9-14 — RDBACK1 Register format .....	227
Table 9-15 — Configuration Register zero format.....	230
Table 9-16 — Configuration Register one format.....	231
Table 9-17 — Configuration Register two and three format .....	231
Table 10-1 — Reset effects .....	237
Table 10-2 — Reset state descriptions .....	239
Table 10-3 — Conditions causing assertion of reset types.....	240
Table 10-4 — Reset effects and references .....	241
Table 10-5 — Start-up option determination.....	246
Table 10-6 — TAP.7 Class/Start-up option relationships .....	246
Table 10-7 — Reset values for TAP.7 Start-up options .....	247
Table 10-8 — TAP.7 Class/Escape deployment relationships .....	251
Table 10-9 — Escapes.....	256
Table 10-10 — Selection Alert sequence .....	261
Table 11-1 — Online Activation Code.....	280
Table 11-2 — Extension Code .....	282
Table 11-3 — Drive conflict protection is activated .....	282
Table 11-4 — Global Register State loaded .....	284
Table 11-5 — Creation of a TAP.7 Controller TAPC parking state with the Standard Protocol .....	288
Table 11-6 — Creation of a TAP.7 Controller TAPC parking state with the Advanced Protocol .....	289
Table 11-7 — TAP.7 Controller behavior in the CSM STD state .....	293

Table 11-8 — TAP.7 Controller behavior while in the CSM <i>ADV</i> state.....	294
Table 11-9 — TAP.7 Controller behavior in the CSM <i>OLW</i> state .....	295
Table 11-10 — TAP.7 Controller behavior while in the CSM <i>TEST</i> state.....	301
Table 11-11 — SCNFMT Register value resulting from the Short-Form Selection Sequence .....	302
Table 11-12 — OAC value/TOPOL Register relationships required for placement Online .....	302
Table 11-13 — CP Directives .....	307
Table 11-14 — TAP.7 Controller behavior while in the CSM <i>CHK</i> state .....	308
Table 11-15 — TAP.7 Controller behavior while in the CSM <i>OLS</i> state.....	311
Table 12-1 — Relationship of TAP.7 Classes/signal names .....	318
Table 12-2 — Relationship of TAP.7 Classes/Signal Characteristics .....	319
Table 12-3 — Signal bias effects transferred to signals sharing a connection .....	321
Table 12-4 — TCK/TCKC signal behavior relationships .....	322
Table 12-5 — TMS/TMSC signal behavior relationships with start-up Online .....	330
Table 12-6 — TMS/TMSC signal behavior relationships with start-up Offline .....	330
Table 12-7 — TDI/TDIC signal behavior relationships .....	332
Table 12-8 — TDO/TDOC signal behavior relationships .....	334
Table 12-9 — CLTAPC State Machine walk to provide IEEE 1149.7-Other Behavior .....	340
Table 13-1 — TAP.7 Class/TDO(C) Drive Policy component applicability .....	345
Table 13-2 — Defaults for unimplemented functions when determining TDOC Drive Policy .....	351
Table 13-3 — SGCC state definition .....	363
Table 13-4 — PSGMCL state definition .....	364
Table 13-5 — Scan Group Candidate Count operation .....	366
Table 13-6 — Potential Scan Group Membership Count Last operation .....	367
Table 13-7 — SGMCL state definition .....	368
Table 13-8 — Scan Group Membership Count Last operation .....	368
Table 13-9 — Conditions governing Idle Group Candidacy .....	370
Table 13-10 — Idle Group Membership .....	370
Table 13-11 — Conditions governing Pause-IR and Pause-DR Group Membership .....	370
Table 13-12 — Only Scan Group Member delayed determination .....	370
Table 13-13 — Conditional Group Member Count state definition .....	372
Table 13-14 — Operation of the Conditional Group Membership Count State Machine .....	374
Table 13-15 — Only Conditional Group Member determination .....	375
Table 13-16 — Signal descriptions in Figure 13-13 .....	377
Table 13-17 — SGCC and PSGMCL State Machine encoding .....	377
Table 13-18 — CGMC state encoding .....	379
Table 14-1 — RDY Bit Drive Policies .....	389
Table 14-2 — TDO Bit Drive Policy with the System Path .....	393
Table 14-3 — Voting and single enables .....	394
Table 14-4 — TDO Bit Drive Policy with the Control Paths .....	394
Table 14-5 — Miscellaneous signal name definitions .....	396
Table 14-6 — TDO bit data source selection terms .....	399
Table 15-1 — EMTAPC operation .....	403
Table 16-1 — Permitted methods to control the EMTAPCs' operating modes .....	408
Table 16-2 — Permitted combinations of methods for controlling EMTAPCs .....	408
Table 16-3 — Nomenclature used in description .....	418
Table 16-4 — An example instruction set for the IR managed by an SiP-TAP .....	424
Table 17-1 — ZBS detect state definitions .....	430
Table 17-2 — ZBS count locking states .....	431
Table 17-3 — ZBS use state characteristics with EPU Operating States .....	434
Table 17-4 — State description for Figure 17-6 .....	437
Table 17-5 — Control levels .....	440
Table 18-1 — T1 TAP.7 function/register/command relationships .....	444
Table 18-2 — T1 TAP.7 Controller register descriptions .....	446
Table 18-3 — Specifying the use of T1 TAP.7 optional functions .....	448
Table 18-4 — Precedence of operations affecting the FRESET Register .....	454
Table 18-5 — TAP.7 Controller power-control mode use cases .....	455

Table 18-6 — TAP.7 Controller Power-Control Mode use cases .....	456
Table 18-7 — DTS characteristics required to orchestrate TAP.7 Controller power-down.....	456
Table 18-8 — Power-down request generation summary .....	468
Table 18-9 — Power-control states .....	472
Table 18-10 — Changes to power-down and power-up enables .....	472
Table 19-1 — T2 TAP.7 function/new register/command relationships.....	481
Table 19-2 — T2 TAP.7 new register descriptions .....	482
Table 19-3 — Specifying the use of T2 TAP.7 configurations .....	483
Table 19-4 — T2 TAP.7 Scan Format characteristics.....	484
Table 19-5 — Current Scan Group Candidacy creation.....	489
Table 19-6 — Maximum synchronization stays in <i>Run-Test/Idle</i> state.....	491
Table 19-7 — Factors determining the Scan Selection State for a T2 TAP.7 .....	493
Table 20-1 — T3 TAP.7 function/new register/command relationships.....	499
Table 20-2 — T3 TAP.7 new register definitions .....	500
Table 20-3 — T3 TAP.7 Scan Format characteristics.....	502
Table 20-4 — TCA format.....	503
Table 20-5 — AT format.....	506
Table 20-6 — CID allocation support with a T3 TAP.7.....	510
Table 20-7 — Participation of a TAP.7 Controller in CID allocation.....	510
Table 20-8 — External-AT derivation during CR Scan <i>Shift-DR</i> states .....	510
Table 20-9 — TDOC behavior/undirected allocation method and JScan3 Scan Format .....	512
Table 20-10 — Executed SSD/ <i>Run-Test/Idle</i> /group membership relationships.....	514
Table 20-11 — Executed SSD/ <i>Pause-xR</i> /group membership relationships .....	514
Table 20-12 — SSD Processing State Machine state names.....	517
Table 20-13 — SSD state description .....	519
Table 20-14 — SSDs and address payloads .....	527
Table 20-15 — SSD/SSD state relationships .....	530
Table 20-16 — Permitted SSD processing .....	530
Table 20-17 — Scan Selection Directive/SGC relationships .....	530
Table 20-18 — Scan topology connectivity and continuity tests .....	534
Table 20-19 — Topology Tests/TOPOL Register relationships .....	536
Table 20-20 — TOPOL Register initialization.....	536
Table 21-1 — Transport Function capability .....	549
Table 21-2 — TAP signal behavior.....	551
Table 21-3 — Summary of APU Operating State change relationships.....	555
Table 21-4 — Description of APU Operating State change relationships.....	557
Table 21-5 — Description of packet combinations .....	559
Table 21-6 — Packet and TAPC state relationships.....	562
Table 22-1 — Variable-length delay examples .....	571
Table 23-1 — T4 TAP.7 function/register/command relationships .....	581
Table 23-2 — T4 TAP.7 register descriptions.....	584
Table 23-3 — Specifying the use of T4 TAP.7 optional functions .....	586
Table 23-4 — STCKDC register relationship to <i>sys_tck</i> signal generation.....	595
Table 23-5 — TMS value when the Advanced Protocol is used .....	599
Table 23-6 — <i>sys_tms</i> signal generation .....	599
Table 23-7 — TDI value when the Advanced Protocol is used .....	600
Table 23-8 — <i>sys_tdi/epu_tdi</i> signal generation .....	600
Table 23-9 — <i>epu_tdo_in</i> signal generation .....	601
Table 23-10 — SSD detection with a T4 and above TAP.7 .....	605
Table 24-1 — SP Delay Element characteristics.....	615
Table 24-2 — Variable-length SP Delay Element Directives .....	616
Table 24-3 — Payload State Machine states for the MScan Scan Format .....	620
Table 24-4 — Ready State Machine states.....	621
Table 24-5 — Delay State Machine state .....	623
Table 25-1 — OScan SP Payload Element contents .....	629
Table 25-2 — Labels in Figure 25-15/TAPC state cross-reference.....	637

Table 25-3 — Cross reference of labels in Figure 25-16 and Figure 25-17/TAPC state.....	639
Table 25-4 — Payload State Machine states for the OScan Scan Format.....	641
Table 26-1 — HDR[2] interpretation .....	657
Table 26-2 — HDR[1:0] interpretation .....	658
Table 26-3 — SScan Scan Formats optimization summary .....	660
Table 26-4 — SScan0 header information and SP payload contents .....	661
Table 26-5 — SScan2 header information and SP payload contents .....	661
Table 26-6 — SScan1 header information and SP payload contents .....	662
Table 26-7 — SScan3 header information and SP payload contents .....	662
Table 26-8 — SScan1 no stalls and input-only Data Segments .....	664
Table 26-9 — SScan3 no stalls and input-only Data Segments .....	665
Table 26-10 — SScan1 no stalls and input/output Data Segment .....	665
Table 26-11 — SScan3 no stalls and input/output Data Segment .....	666
Table 26-12 — SScan2/SScan3 output-only Data Segments/no RDY bit in the SP payload.....	670
Table 26-13 — SScan2/SScan3 output-only Data Segments/RDY bit(s) in the SP payload.....	671
Table 26-14 — Cross-reference of the TAPC state labels in Figure 26-19 .....	674
Table 26-15 — TAPC advance with Data Segments other than SScan2–SScan3 output only.....	677
Table 26-16 — TAPC advance with SScan2–SScan3 output-only Data Segments .....	678
Table 26-17 — Payload State Machine state definition .....	683
Table 26-18 — Escape Detection State Machine state definition .....	684
Table 27-1 — T5 TAP.7 function/register/command relationships .....	693
Table 27-2 — T5 TAP.7 register descriptions.....	696
Table 27-3 — PDCx_DCCy_CR0 – PDCx_DCCy_CR7 write descriptions .....	697
Table 27-4 — PDCx_DCCy_CR0 – PDCx_DCCy_CR7 read descriptions .....	698
Table 27-5 — Reset values of TAP.7 Controller registers managed with Transport Directives .....	699
Table 27-6 — Specifying the use of T5 TAP.7 optional functions .....	700
Table 27-7 — T5 TAP.7 Controller Transport Register configurations.....	700
Table 27-8 — DCC register configurations.....	700
Table 27-9 — Real TAPC state and transport relationships.....	703
Table 27-10 — Transport Directive nomenclature .....	707
Table 27-11 — Transport Directive encoding .....	708
Table 27-12 — Selection Directive operation (conditional relationships) .....	712
Table 27-13 — Register Transfer Directive operation (conditional relationships) .....	713
Table 27-14 — Data Transfer Directive operation (conditional relationships) .....	714
Table 27-15 — DCC/Data Element relationships .....	719
Table 28-1 — Register Element content creation.....	733
Table 28-2 — Register Element content utilization .....	734
Table 28-3 — Data Element content .....	737
Table 28-4 — TSM state names and encoding.....	740
Table 28-5 — TSM state behavior .....	741
Table 28-6 — PDCx/DCC interface signal functions .....	747
Table 28-7 — PDCx/DCC interface signal description, PDC sourced signals.....	748
Table 28-8 — PDCx/DCC interface signal description, DCC-sourced signals.....	749
Table 28-9 — PDCx/DCC interface signaling relationships.....	752
Table 28-10 — TP_CRR or TP_CRW Directive (eight-bit Register Element).....	768
Table 28-11 — TP_DCx Directive – variable-length transfer (eight-bit Data Element).....	769
Table 28-12 — TP_DCx Directive – fixed-length transfer (eight-bit Data Element) .....	770
Table 29-1 — Contrasting large-system applications against small-system applications .....	785
Table 30-1 — Scope of BSDL.7 .....	788
Table 30-2 — BSDL.7 reserved words .....	790
Table 31-1 — Scope of HSDL.7 .....	808
Table 31-2 — HSDL.7 reserved words .....	809
Table C-1 — TMSC drive and delay relationships .....	862
Table C-2 — MScan scan sequence/DR Scan (010) and IR Scan (111), delay as specified .....	863
Table C-3 — SP/CP combination following an MScan SP .....	863
Table C-4 — OScan0 scan sequence/DR Scan (010) and IR Scan (111), delay as specified.....	864

Table C-5 — SP/CP combination following an OScan0 SP.....	864
Table C-6 — OScan1 scan sequence/DR Scan (010) and IR Scan (111), delay as specified.....	865
Table C-7 — SP/CP combination following an OScan1 SP.....	865
Table C-8 — OScan2 scan sequence/DR Scan (010) and IR Scan (111) with no delay .....	866
Table C-9 — OScan2 scan sequence/DR Scan (010) and IR Scan (111) with delay .....	867
Table C-10 — SP/CP combination following an OScan2 SP.....	867
Table C-11 — OScan3 scan sequence/DR Scan (010) and IR Scan (111) with no delay .....	868
Table C-12 — OScan3 scan sequence/DR Scan (010) and IR Scan (111) with delay .....	869
Table C-13 — SP/CP combination following an OScan3 SP.....	869
Table C-14 — OScan4 scan sequence/DR Scan (010) and IR Scan (111).....	870
Table C-15 — SP/CP combination following an OScan4 SP.....	870
Table C-16 — OScan5 scan sequence/DR Scan (010) and IR Scan (111).....	871
Table C-17 — SP/CP combination following an OScan5 SP.....	871
Table C-18 — OScan6 scan sequence/DR Scan (010) and IR Scan (111) with no delay .....	872
Table C-19 — OScan6 scan sequence/DR Scan (010) and IR Scan (111) with delay .....	873
Table C-20 — SP/CP combination following an OScan6 SP.....	873
Table C-21 — OScan7 scan sequence/DR Scan (010) and IR Scan (111) with no delay .....	874
Table C-22 — OScan7 scan sequence/DR Scan (010) and IR Scan (111) with delay .....	875
Table C-23 — SP/CP combination following an OScan7 SP.....	875
Table C-24 — SScan0 – SP sequence following a CP/all stall profile (IN, I/O).....	877
Table C-25 — SP/CP combination following an SScan0 Segment with an all SP stall profile .....	877
Table C-26 — SScan0 – SP sequence following a CP/all stall profile (IN, OUT).....	878
Table C-27 — SScan0 – SP sequence following a CP/all stall profile (I/O, OUT).....	879
Table C-28 — SScan0 – SP sequence following a CP/first stall profile (IN, I/O) .....	880
Table C-29 — SP/CP combination following an SScan0 segment with first SP stall profile .....	880
Table C-30 — SScan0 – SP sequence following a CP/first stall profile (IN, OUT) .....	881
Table C-31 — SScan0 – SP sequence following a CP/first stall profile (I/O, OUT) .....	882
Table C-32 — SScan1 – SP sequence following a CP/no stall profile (IN, I/O).....	883
Table C-33 — SP/CP combination following an SScan1 segment with a no stall profile.....	883
Table C-34 — SScan1 – SP sequence following a CP/no stall profile (IN, OUT).....	884
Table C-35 — SScan1 – SP sequence following a CP/no stall profile (I/O, OUT).....	885
Table C-36 — SScan1 – SP sequence following a CP/first stall profile (I/O, OUT) .....	886
Table C-37 — SScan1 – SP sequence following a CP/first stall profile (IN, I/O) .....	887
Table C-38 — SP/CP combination following an SScan1 segment with first SP stall profile .....	887
Table C-39 — SScan1 – SP sequence following a CP/first stall profile (IN, OUT) transactions .....	888
Table C-40 — SScan1 – SP sequence following a CP/first stall profile/I/O, OUT transactions .....	889
Table C-41 — SScan2 – SP sequence following a CP/all stall profile (IN, I/O).....	890
Table C-42 — SP/CP combination following an SScan2 segment with an all SP stall profile .....	890
Table C-43 — SScan2 – SP sequence following a CP/all stall profile (IN, OUT).....	891
Table C-44 — SScan2 – SP sequence following a CP/all stall profile (I/O, OUT).....	892
Table C-45 — SScan2 – SP sequence following a CP/first stall profile (IN, I/O) .....	893
Table C-46 — SP/CP combination following an SScan2 segment with first SP stall profile .....	893
Table C-47 — SScan2 – SP sequence following a CP/first stall profile (IN, OUT) .....	894
Table C-48 — SScan2 – SP sequence following a CP/first stall profile (I/O, OUT) .....	895
Table C-49 — SScan3 – SP sequence following a CP/no stall profile (IN, I/O).....	896
Table C-50 — SP/CP combination following an SScan3 segment with a no stall profile.....	896
Table C-51 — SScan3 – SP sequence following a CP/no stall profile (IN, OUT).....	897
Table C-52 — SScan3 – SP sequence following a CP/no stall profile (I/O, OUT).....	898
Table C-53 — SScan3 – SP sequence following a CP/first stall profile (I/O, OUT) .....	899
Table C-54 — SScan3 – SP sequence following a CP/first stall profile (IN, I/O) .....	900
Table C-55 — SP/CP combination following an SScan3 segment with first SP stall profile .....	900
Table C-56 — SScan3 – SP sequence following a CP/first stall profile (IN, OUT) transactions .....	901
Table C-57 — SScan3 – SP sequence following a CP/first stall profile/I/O, OUT transactions .....	902
Table D-1 — TAP power and selection states viewed by the DTS .....	905
Table D-2 — Selection-Sequence initiation .....	906
Table D-3 — Selection-Sequence criteria .....	907

Table D-4 — Criteria for placement Online.....	907
Table D-5 — TAP power and selection states viewed by the DTS .....	908
Table D-6 — Placement of Offline-at-Start-up Online with TAPC state sequence .....	912
Table D-7 — TAPC state mnemonics .....	914
Table D-8 — Other short-form synchronization state sequences .....	915
Table D-9 — Configuration Register reads for TAPs [n:0] .....	920
Table F-1 — Factors affecting system performance and reliability .....	934
Table F-2 — Recommendations summary .....	1018
Table F-3 — Recommended connection/termination schemes .....	1021

# **IEEE Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture**

## **1. Overview**

### **1.1 Scope**

The standard will define a link between IEEE 1149.1 interfaces in Debug and Test Systems (DTS) and IEEE 1149.1 (JTAG) interfaces in Target Systems (TS). The link defined by this standard introduces an additional layer between these legacy interfaces. This layer may be viewed as an adapter that provides new functionality and features while preserving all elements of the original IEEE 1149.1 (JTAG) interfaces. The standard will define the link behavior (including timing characteristics of signals), protocols, and functionality of the adapters deployed within the DTS and TS. The standard will not modify or create inconsistencies with IEEE 1149.1 (JTAG). The standard will define a superset of the IEEE 1149.1 specification and achieve compliance with IEEE Std 1149.1<sup>TM</sup>.<sup>1</sup>

### **1.2 Purpose**

The purpose of the standard is to define a debug and test interface that meets an expanding set of challenges facing Debug and Test Systems (many of which have emerged since the inception of the original IEEE Std 1149.1) while preserving the hardware and software investments of the many industries currently using IEEE Std 1149.1.

### **1.3 Word usage**

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).<sup>2,3</sup>

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*should* equals *is recommended that*).

---

<sup>1</sup> Information on references can be found in Clause 2.

<sup>2</sup> The use of the word *must* is deprecated and cannot be used when stating mandatory requirements, *must* is used only to describe unavoidable situations.

<sup>3</sup> The use of *will* is deprecated and cannot be used when stating mandatory requirements, *will* is only used in statements of fact.

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

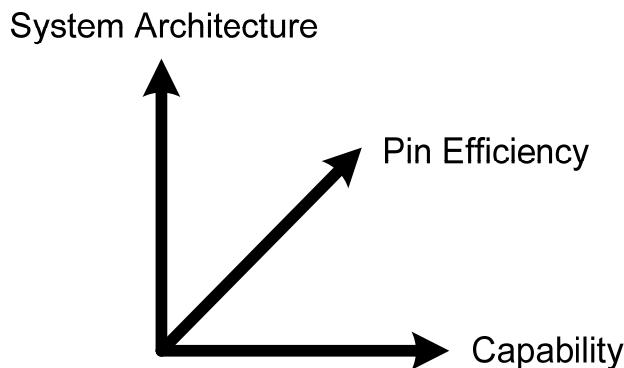
The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

## 1.4 Contrasting IEEE Std 1149.1 and this standard

IEEE Std 1149.1 was introduced to address issues related to the manufacturing test of boards and systems. It has since been used as a debug portal for system integration and applications debug. IEEE Std 1149.1 has provided an effective framework for the test circuitry included in systems, enabling a variety of compatible products to evolve across the entire electronics industry.

This standard is complementary to IEEE Std 1149.1 and does not seek to replace it. It uses IEEE Std 1149.1 as its foundation. It expands the impact of IEEE Std 1149.1 in the three dimensions shown in Figure 1-1 while remaining compatible with this standard.

This standard differs from IEEE Std 1149.1 as it intentionally blurs the boundary between boards, packages, and chips. It recognizes that multi-chip solutions become single-chip solutions and that boards become multi-chip and single-chip solutions over time. It acknowledges that these inevitable integration steps create chips and packages with multiple Test Access Port Controllers (TAPCs). This is accomplished without modifying or creating inconsistencies with IEEE Std 1149.1.



**Figure 1-1 — IEEE 1149.7 impact areas**

It provides support for sophisticated system architectures by supporting:

- The use of a hierarchy of on-chip TAPCs while maintaining the behavior specified by IEEE Std 1149.1
- Power control to allow the power-down of debug and test logic
- Selection protocols that may be adopted by other technologies so they may share a common connection to a DTS with 1149.7 Test Access Ports (TAPs)

Pin efficiency is provided by supporting:

- Operation with both four-pin and two-pin TAPs with Star Scan Topologies and a four-pin TAP with a Series Scan Topology

- The functionality of the five-pin TAP [Test Reset (TRST\*), Test Clock (TCK), Test Model Select (TMS), Test Data Input (TDI), and Test Data Output (TDO)] specified by IEEE Std 1149.1 with a two-pin TAP
- Interleaved use of a two-pin TAP for scan and non-scan transfers
- Operation with other technologies and multiple scan topologies sharing a common DTS connection

Additional capability is provided by supporting:

- Test and functional reset generation on-chip
- Scan transactions optimized for test and for debug
- Data transport between a chip and combinations of the DTS and/or other chips
- The use of the TAP by custom functions

In addition, this standard provides:

- Backward compatibility with intellectual property (IP) developed using IEEE Std 1149.1
- A layered implementation creating six operating classes, each with increasing capability

## 1.5 Challenges

Increasing chip integration and the growing focus on power management have created new challenges that were not considered when IEEE Std 1149.1 was originally developed. These challenges originate from debug, functional, and test requirements, and the creation of TAPs whose operation includes behavior that deviates from the behavior specified by IEEE Std 1149.1. These industry trends have made it difficult to build chips that exhibit the behavior specified by IEEE Std 1149.1. These challenges are outlined as follows.

Debug:

- The emphasis on scan performance when the TAP is used for debug purposes.
- Additional functionality such as data transport is needed.

Functional:

- The miniaturization of systems, systems on a chip, and systems in a package has created the need for a reduced pin TAP.
- Design requirements for very low power consumption in chips have created a need to power-down test and debug logic when it is not being used.

Test:

- Behavior that is specified by IEEE Std 1149.1:
  - A chip has become a board with multiple TAP controllers being deployed on the same chip.
  - Packaging creates multiple TAPs in a package (multi-chip modules and stacked dies).
- Separate test and debug views are needed:
  - A test view of the chip boundary.
  - A debug view of multiple internal chip components, each having a TAP controller and associated test architecture.

Deviations from the standard have created:

- IP with nonconforming IEEE 1149.1-like interfaces.
- Components that are not interoperable when used in the manner specified by IEEE Std 1149.1.
- Problems for integrating hardware components from different suppliers.
- Interoperability problems for the tools supporting the standard.

## 1.6 Important considerations

Five important considerations have influenced the standard:

- Ensuring the interoperability of components based on this standard with components based on IEEE Std 1149.1
- Maximizing compatibility with chip, TS, and DTS architectures
- Reducing, by half in some instances, the number of required signals for a TAP while at the same time providing more capability when compared with an IEEE 1149.1 TAP
- Supporting TAPs with behavior that is either IEEE 1149.1-Specified or IEEE 1149.1-Non-disruptive (see 1.7.2 for a description of these behaviors)
- Providing the capability where a chip architect implements only what is needed

## 1.7 Nomenclature

### 1.7.1 References to technology and standards

References to IEEE 1149.1 within the remainder of this document are to be interpreted as references to the technology defined by IEEE Std 1149.1. References to IEEE 1149.7 within the remainder of this document are to be interpreted as references to the technology defined by IEEE Std 1149.7. References to IEEE Std 1149.1 are to be interpreted as references to the actual standard.

### 1.7.2 Describing Test Access Port behaviors

This standard describes the operation of a TAP as having one of three behaviors:

- |                           |   |
|---------------------------|---|
| — Specified Behavior      | Obeyes all applicable rules of the standard governing its operation   |
| — Non-disruptive Behavior | Obeyes some but not all rules of the standard governing its operation and operates in a manner where it is interoperable with a TAP with Specified Behavior         |
| — Other Behavior          | Obeyes a limited set of the rules of the standard governing its operation and operates in a manner where it is not interoperable with a TAP with Specified Behavior |

These behaviors apply to both IEEE Std 1149.1 and this standard. A component having “IEEE 1149.1-Specified Behavior” obeys all of the rules of IEEE Std 1149.1. Likewise a component having “IEEE 1149.7-Specified Behavior” obeys all of the rules of this standard that are applicable to the TAP.7 Class implemented.

A component having “1149.1-Non-disruptive Behavior” obeys a subset of the rules of IEEE Std 1149.1. Its behavior does not prevent its use with TAPs having IEEE 1149.1-Specified Behavior. It contains a TAPC that exhibits the exact behavior specified by IEEE Std 1149.1. It shifts scan data through the Instruction and Data Registers only during the *Shift-IR* and *Shift-DR* states. However, the mandatory Instruction Register

(IR) length, the Data Register (DR) length for the *BYPASS* instruction, and the TDO signal operation in TAPC State Machine states other than *Shift-xR* all may differ from IEEE 1149.1-Specified Behavior. The use of a TAP with IEEE 1149.1-Non-disruptive Behavior with TAPs that have IEEE 1149.1-Specified Behavior in a system neither prevents accessing the TAPs with IEEE 1149.1-Specified Behavior nor modifies the behavior of the these TAPs as observed by the DTS.

A component having “1149.7-Non-disruptive Behavior” obeys a subset of the rules of this standard. Its behavior does not prevent its use with TAPs having 1149.7-Specified Behavior. The use of a TAP with 1149.7-Non-disruptive Behavior with TAPs that have 1149.7-Specified Behavior in a system neither prevents accessing the TAPs with 1149.7-Specified Behavior in a system nor modifies the behavior of the these TAPs as observed by the DTS.

A component having “1149.1-Other Behavior” has characteristics that prevent its use with TAPs with 1149.1-Specified Behavior in systems that utilize all aspects of the TAPC and IR and DR Scan Paths. A component having “1149.7-Other Behavior” has characteristics that prevent its use with TAPs with 1149.7-Specified Behavior in systems.

### 1.7.3 Describing TAPs and TAP controllers

Since 1149.1 and 1149.7 TAPs can have different signal counts and behaviors, a reference to a TAP within this standard are precise enough to identify a specific TA. The following nomenclature is followed when referring to TAPs:

- “TAP” means a Test Access Port without defining its type.
- “TAP.1” means a TAP providing 1149.1-Specified Behavior.
- “TAP.7” means a TAP providing 1149.7-Specified Behavior.
- “Tx TAP.7” means the TAP for 1149.7 Class Tx.
- “Tx and above TAP.7” means the TAP for 1149.7 Class Tx and those TAP.7 Classes above Tx.
- “Tx and below TAP.7” means the TAP for 1149.7 Class Tx and those TAP.7 Classes below Tx.
- “Tx - Ty TAP.7” means the TAP for 1149.7 Classes Tx through Ty.

The term “TAPC” is used to describe the TAP controller (Finite State Machine) specified by IEEE Std 1149.1. This definition does not include the scan paths and other functions within the test architecture that are managed by the TAPC. A scan path is never provided by a TAPC; it is merely associated with it. Within this standard, the following nomenclature is followed when referring to TAP controllers:

- “TAPC” is the TAP controller (the Finite State Machine) specified by IEEE Std 1149.1.
- “ADTAPC” is the adapter TAPC implemented within the TAP.7 Controller.
- “CLTAPC” is the TAPC implemented at the chip level and connected to the TAP.7 Controller.
- “EMTAPC” is a TAPC other than the CLTAPC implemented within the System Test Logic.

Note that use of the word “controller” in this standard has an expanded meaning when compared with its use in IEEE Std 1149.1. The term “TAP.7 Controller” is used to describe the bridge function connecting the TAP signals to the CLTAPC signals. This function includes an ADTAPC and other logic supporting TAP.7 Controller capability. The terminology describing the relationships of the TAP.7 Controller, the CLTAPC, and EMTAPCs within the System Test Logic are described in 4.2.3.3.

Also note that the term “System Test Logic” has a meaning different than the use of the term “system logic” in IEEE Std 1149.1. Within this standard, the term “System Test Logic (STL)” refers to test and debug logic that is as follows:

- Implemented within the chip
- Not part of the TAP.7 Controller
- Accessed via the TAP.7

The term “functional logic” means logic within the chip other than test and debug logic. This includes test and debug logic accessed by means other than the TAP during the normal operation of the circuit. This definition is similar to the use of the term “system logic” in IEEE Std 1149.1.

#### 1.7.4 Describing scan exchanges

Within this document, the following scan terminology is used:

- |                                       |   |
|---------------------------------------|---|
| — Data Register Scan (DR Scan)        | A TAPC state progression beginning with the <i>Select-DR</i> state and progressing to the <i>Update-DR</i> state without passing through the <i>Select-IR</i> or <i>Test-Logic Reset</i> states           |
| — Instruction Register Scan (IR Scan) | A TAPC state progression beginning with the <i>Select-IR</i> state and progressing to the <i>Update-IR</i> state without passing through the <i>Test-Logic Reset</i> state                                |
| — Control Register Scan (CR Scan)     | A DR Scan that accesses TAP.7 Controller registers following certain TAP.7 Controller commands  |
| — Zero-Bit Scan (ZBS)                 | A DR Scan that is created with a TAPC state progression that begins with the <i>Select-DR-Scan</i> state and culminates with the <i>Update-DR</i> state without passing through the <i>Shift-DR</i> state |

#### 1.7.5 Describing TAP signals

Note that the nomenclature for the Test Reset signal (TRST\*) described by IEEE Std 1149.1 has been changed to the Not Test Reset signal (nTRST) in this standard. Test Reset Signals with default inactive (nTRST) and active [Not Test Reset Pull-Down (nTRST\_PD)] states are supported. The nomenclature for other signals described by IEEE Std 1149.1 is changed when the function of the signal has changed as shown below and remains the same as that used by IEEE Std 1149.1 when the function of the signal is not changed:

- |        |   |
|--------|---|
| — TCKC | The IEEE 1149.1 Test Clock Compact signal with added 1149.7 functionality       |
| — TMSC | The IEEE 1149.1 Test Mode Select signal with added 1149.7 functionality         |
| — TDIC | The IEEE 1149.1 Test Data Input Compact signal with added 1149.7 functionality  |
| — TDOC | The IEEE 1149.1 Test Data Output Compact signal with added 1149.7 functionality |

The signal names below indicate a signal with either the 1149.1 or 1149.7 functionalities:

- |          |                             |
|----------|-----------------------------|
| — TCK(C) | The Test Clock signal       |
| — TMS(C) | The Test Mode Select signal |
| — TDI(C) | The Test Data Input signal  |
| — TDO(C) | The Test Data Output signal |

## 1.8 Ensuring transparency to IEEE 1149.1 intellectual property

### 1.8.1 IEEE 1149.1 constraints

This standard places a premium on being as transparent as possible to the users of IEEE Std 1149.1. This document accommodates the constraints created by:

- Large installed bases of hardware and software IP based on IEEE Std 1149.1
- Components deployed with a mixture of IP developed over extended periods of time (1–10 years)
- Inertia of the industry and resistance to discontinuities

### 1.8.2 IEEE 1149.1 constraints/requirements balancing

This standard balances the constraints created by the industry inertia and emerging requirements by:

- Providing compatibility with IEEE Std 1149.1
- Tolerating IP with deviations from IEEE Std 1149.1
- Delivering the standard’s functionality with a scalable solution
- Creating an extensible platform allowing customization

### 1.8.3 IEEE 1149.1 upgrade path

This standard provides for the use of DTS and TS intellectual property using the upgrade path shown in Figure 1-2. An IEEE 1149.7 Adapter is added to existing IEEE 1149.1 IP to create 1149.7 intellectual property. The term “TAP.7 Controller” is used to represent the IEEE 1149.7 TS Adapter within a chip in this document.

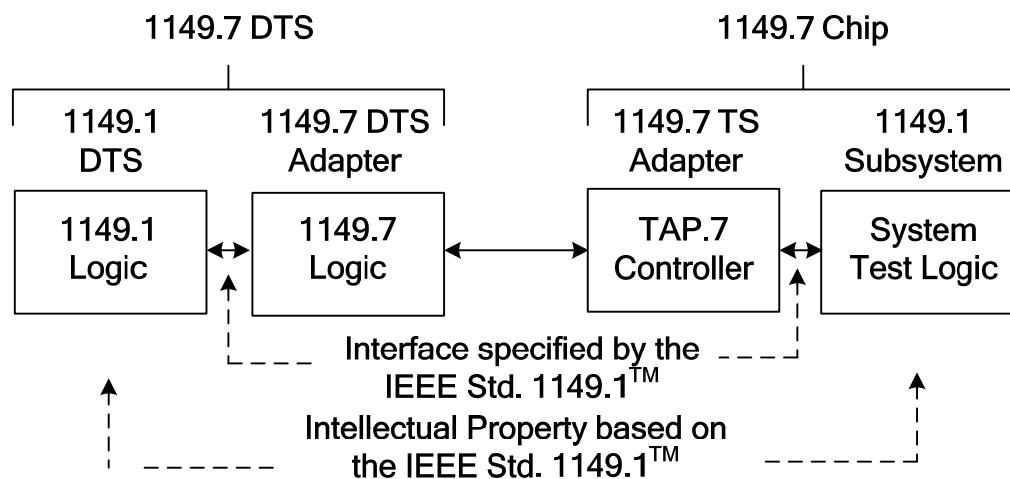


Figure 1-2 — IEEE 1149.7/IEEE 1149.1 upgrade path

## 1.9 Maximizing compatibility with 1149.1 IP

The architectures of chips, systems, and DTS equipment were considered to maximize compatibility provided by this standard. The areas given serious consideration are as follows:

- Legacy test architecture IEEE 1149.1 and variant implementations
- Infrastructure Instructions, scan paths, and boundary-scan capability
- Test Clock treatment Chip, TS, and DTS

### 1.9.1 IEEE 1149.1 infrastructure

#### 1.9.1.1 IEEE 1149.1 instructions

The IEEE 1149.1 architecture does not change or add instructions to those specified by IEEE Std 1149.1. The IEEE 1149.7 Controller is managed using DR Scans associated with either the *BYPASS* or the *IDCODE* Instruction. These instructions are used for IEEE 1149.7 purposes following a seldom if ever used TAPC state progression. Since the DR Scan Paths of these two instructions do not change the operation of circuitry associated with a TAP Controller in any meaningful way, these instructions may be used for IEEE 1149.7 purposes without adverse consequences.

#### 1.9.1.2 IEEE 1149.1 Scan Paths

The IR and DR Scan Paths of the System Test Logic shown in Figure 1-2 are accessible via the TAP with all their native characteristics (length, capture/update characteristics, etc.) preserved. Existing software drivers do not need to be changed to comprehend operation with the IEEE 1149.7 Adapter. An additional driver layer is added to manage the TAP.7 Controller using conventional DR Scans with the *BYPASS* or *IDCODE* Instructions.

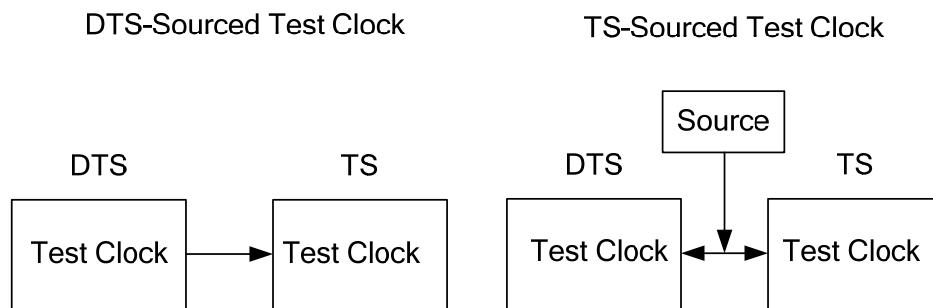
#### 1.9.1.3 IEEE 1149.1 boundary-scan capability

The boundary-scan capability specified by IEEE Std 1149.1 is available in its entirety without modification with this standard. The ability to create a scan operation in a Star Scan Topology equivalent to that used in a Series Scan Topology is provided.

### 1.9.2 Test Clock treatment

#### 1.9.2.1 Test Clock source

The Test Clock may be sourced by either the DTS or from within the TS as shown in Figure 1-3.



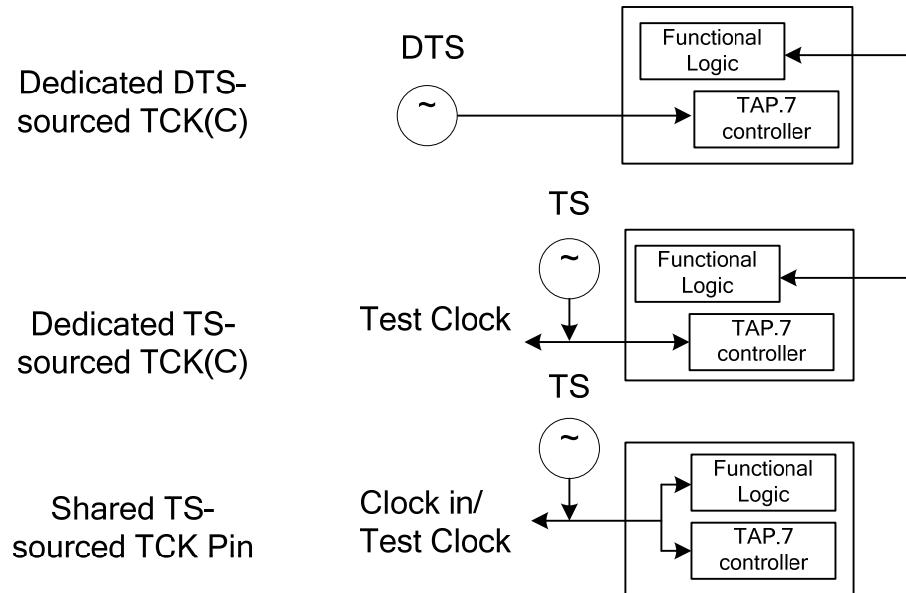
**Figure 1-3 — Systems where the DTS and TS source the Test Clock**

### 1.9.2.2 Test Clock shared or dedicated use

A chip may be built in a manner where the Test Clock is one of the following:

- Dedicated to the Test Clock function
- Shared with another chip function

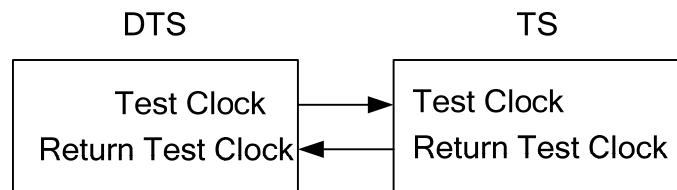
When the Test Clock is shared with another chip function, this clock is generally required to be free running and is generated within the target system. With this configuration, the Test Clock is also sourced to the DTS. Both Dedicated and Shared Test Clock configurations are shown in Figure 1-4.



**Figure 1-4 — Dedicated and Shared Test Clock signal**

### 1.9.2.3 Unorthodox use of Test Clock

In some instances, systems with IEEE 1149.1-based IP have been implemented with unorthodox Test Clock behavior. This behavior is inconsistent with the Test Clock behavior specified by IEEE Std 1149.1. With this IP, the DTS sources a Test Clock but receives a return Test Clock as shown in Figure 1-5. In this case, the DTS-sourced Test Clock may be free running or controlled in response to the Return Test Clock. In the case where the Test Clock is free running, the TS-generated Return Test Clock occurs at a rate that is compatible with the TS and at the rate TAPC state changes occur. Hence, the ratio of the Test Clock and the Return Test Clock may or may not be one to one.



**Figure 1-5 — Systems with a Return Test Clock**

All considerations related to systems and chip IP utilizing a Return Test Clock are viewed as deprecated in this specification. These considerations are covered in Annex H but only for completeness. This annex may be ignored if it is not relevant to a chip's architecture. Creation of IP with this behavior does not provide the behavior that is specified by IEEE Std 1149.1 and is strongly discouraged.

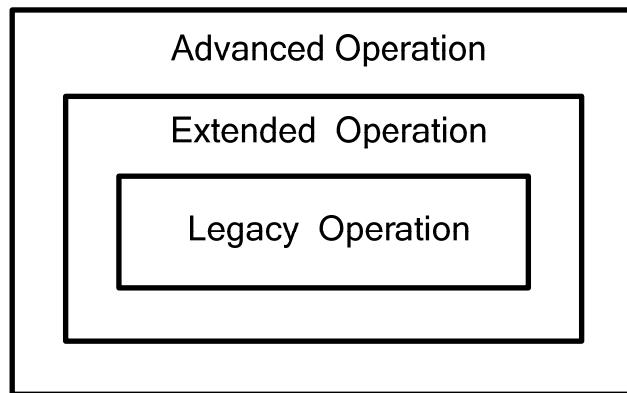
## 1.10 Scalability

### 1.10.1 Types of operation/capability classes

This standard provides a scalable solution to meet the needs of varied component and system complexities. This solution supports the three types of operation shown in Figure 1-6, and is delivered with the six TAP.7 Classes (T0–T5) shown in Figure 1-7.

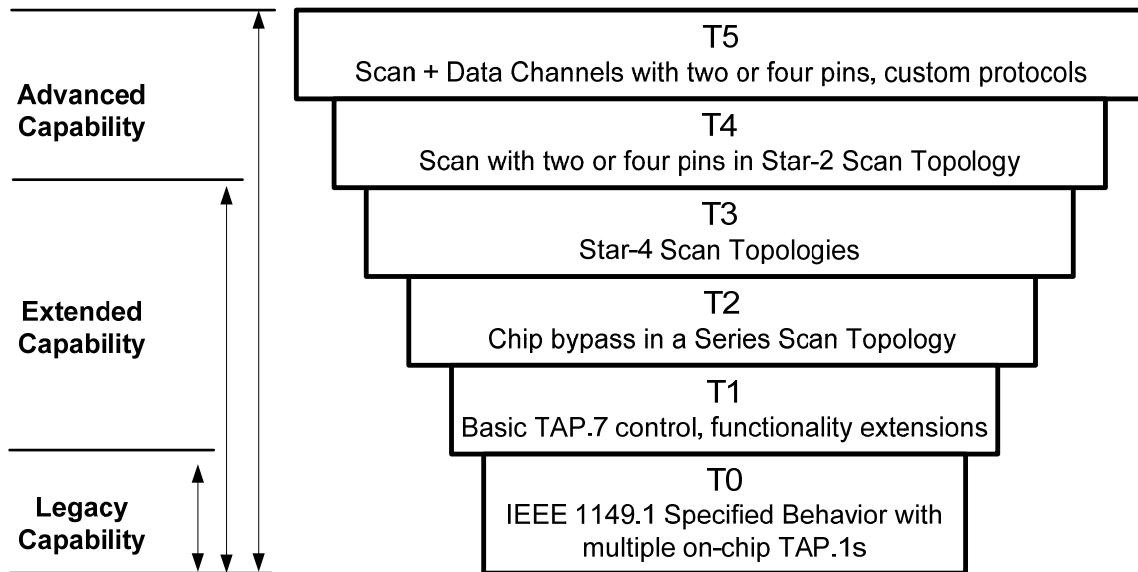
Legacy operation provides for building chips with one or more TAP controllers that provide the test view required by IEEE Std 1149.1. This capability is supported with all TAP.7 Classes. Extended operation is added to the legacy operation with T1 and above TAP.7s. This creates a capability that is a superset of IEEE Std 1149.1. Extended operation provides 1149.1-Non-disruptive Behavior and is managed with the Standard Protocol (that is used by IEEE Std 1149.1). Some of the Extended Capabilities are managed by using additional TCK and TMS signaling sequences called Escapes (see 10.4). The use of an Escape is transparent to the operation of an IEEE 1149.1 TAP.

Advanced operation provides for both scan and non-scan transfers using only the TCKC and TMSC signals. With this the case, the Advanced Protocol is not compatible with the Standard Protocol. However, the Standard Protocol may be used to switch to the use of the Advanced Protocol and vice versa.



**Figure 1-6 — Types of TAP.7 operation**

The six TAP.7 Classes deliver a scalable set of features. Mandatory features of a class enable the functionality of all classes above it. Optional features of a class either improves the TAP performance or is needed by some application types but not others. This approach delivers a set of capabilities that increases with each TAP.7 Class. Chip architects choose the precise feature set required for their application.



**Figure 1-7 — TAP.7 capability classes and their layered capability**

An overview of the features of each class is provided as follows:

- T0 A TAP.7 allowing multiple on-chip TAPCs using the Standard Protocol.
- T1 A T0 TAP.7 plus infrastructure supporting TAP.7 Controller commands/registers, test and functional resets, and test/debug power-management logic.
- T2 A T1 TAP.7 plus a one-bit IR and DR chip bypass path that may be used to improve system scan performance when the chip's scan paths are of no interest. An option to use the chip bypass at startup is provided. This option supports the physical connection of a DTS to a running system without corrupting its operation.
- T3 A T2 TAP.7 plus support for the use of the TAP in a Star-4 Scan Topology. Direct addressability of TAP.7 Controllers is added along with a means to synthesize the equivalent of series scan operations in a Star Scan Topology.
- T4 A T3 TAP.7 plus serialized scan transfers using the Advanced Protocol. The TDIC and TDOC signals become optional. Either narrow (two-signal) or wide (four-signal) TAP.7s may be implemented. A wide TAP.7 provides T3 TAP.7 capability using the IEEE 1149.1 protocol (Standard Protocol) or two-signal debug and test scan transfers (Advanced Protocol) while the TDIC and TDOC signals are used for other functions.
- T5 A T4 TAP.7 plus non-scan transfers with eight data channels using the Advanced Protocol. Custom debug technologies (for example, data logging).

A chip may be built with only Legacy Operation (Class T0), Legacy and Extended Operation (Class T1–T3), or Legacy, Extended, and Advanced Operation (Class T4–T5). Each of three types of operation may be implemented in a manner supporting the shared use of the TAP signals with other technologies and/or a mixture of Legacy, Extended, and Advanced Operation. The TAP.7 Classes are described at a high level in Clause 5 and Clause 6 and in detail in Clause 16, Clause 18 through Clause 20, Clause 23, and Clause 27.

## 1.10.2 Signaling

The TAP signals for the T0–T5 TAP.7 Classes are listed in Table 1-1. The (C) notation for signal name suffixes indicates the functionality of the signal has been expanded beyond that provided by IEEE Std 1149.1 as described in 1.7.5.

**Table 1-1 — TAP.7 signal list**

Signal name	Description	TAP.7 Class					
		T5	T4	T3	T2	T1	T0
TCK/TCKC	Test Clock	MC <sup>a</sup>	MC	M <sup>b</sup>	M	M	M
TMS/TMSC	Test Mode Select	MC	MC	M	M	M	M
TDI/TDIC	Test Data Input	OC <sup>c</sup>	OC	MC	M	M	M
TDO/TDOC	Test Data Output	OC	OC	MC	M	M	M
nTRST	Test Reset (see note)	O <sup>d</sup>	O	O	O	O	O
nTRST_PD	Test Reset Pull-Down	O	O	O	O	O	O
Mandatory signal count		2	2	4	4	4	4

<sup>a</sup>MC = Mandatory, expanded functionality, and signal name ends in C.

<sup>b</sup>M = Mandatory.

<sup>c</sup>OC = Optional, expanded functionality and signal name ends in C.

<sup>d</sup>O = Optional.

NOTE—The notation for the active state of a signal being a logic 0 in this standard is a lowercase n preceding a signal name. The TRST\* signal defined within IEEE Std 1149.1 becomes nTRST with this standard, with the two signals having the same functionality.<sup>4</sup>

## 1.11 Flexibility

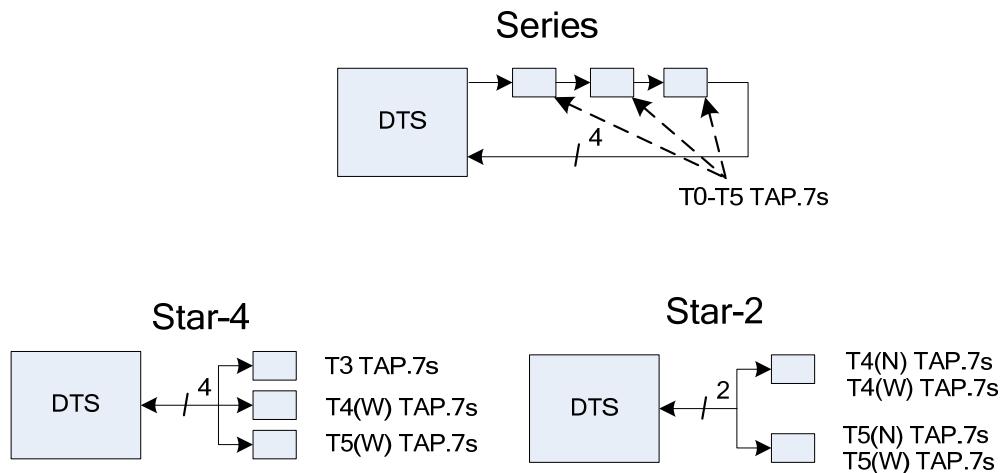
The flexibility of the TAP.7 architecture is highlighted by its support of varied scan topologies and the ability to share the TAP signals with other technologies.

### 1.11.1 Supporting various scan topologies

At the board level, TAP.7s can be connected in the traditional Series Scan Topology, a four-wire (wide) Star Scan Topology, or a two-wire (narrow) Star Scan Topology as shown in Figure 1-8, or a mixture of these scan topologies.

---

<sup>4</sup> Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.



**Figure 1-8 — Utilization of TAP.7 Classes within scan topologies**

The Series Scan Topology is the legacy topology. It provides an excellent solution for systems with a large number of chips. The Star-4 Scan Topology is attractive for systems where TAPs can be incrementally added or removed from the systems (e.g., card racks). A Star-2 Scan Topology is attractive for systems with a limited number of chips where there is a desire to utilize the TAP for functions in addition to scan. The Star Scan Topology introduces issues (e.g., direct addressability and control) not found in a Series Scan Topology. These issues are fully comprehended within the TAP.7 Controller.

The T0–T2 TAP.7s support a Series Scan Topology with the legacy four-wire interface. The T3 TAP.7 supports both the Series and Star Scan Topologies, also with the four-wire interface. The T4–T5 TAP.7s support a two-wire Star Scan Topology. They also support a four-wire Series and Star-4 Scan Topologies when the optional TDI and TDO signals are implemented. This is summarized in Table 1-2.

**Table 1-2 — TAP.7 Class and scan topology relationships**

TAP.7 Class	Mandatory signal count	Use permitted with these scan topologies		
		Series	Star-4	Star-2
T0	4	Yes	No	No
T1	4	Yes	No	No
T2	4	Yes	No	No
T3	4	Yes	Yes	No
T4(N) <sup>a</sup>	2	No	No	Yes
T4(W) <sup>b</sup>	4	Yes	Yes	Yes
T5(N)	2	No	No	Yes
T5(W)	4	Yes	Yes	Yes

<sup>a</sup>Tx(N) = a narrow TAP.

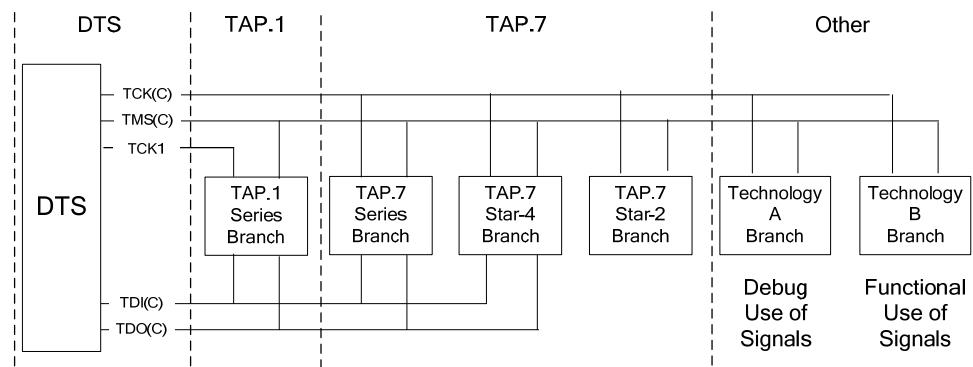
<sup>b</sup>Tx(W) = a wide TAP.

When the TAP.7 architectural guidelines and rules are followed, TAPs based on IEEE Std 1149.1 are interoperable with other TAPs that may be deployed in the same scan topology even though they are different classes or the same class with different feature sets. The interoperability of a TAP.7 is also ensured when custom features are implemented using the guidelines in this standard, provided these features do not produce IEEE 1149.7-Other Behavior.

### 1.11.2 Operation with more than one scan topology/other technologies

The TAP.7 architecture provides for the operation of Series, Star-4, and Star-2 Scan Topologies in parallel using the same DTS signal connections, provided all TAPs connected in these scan topologies are TAP.7s. It may be desirable to operate components using legacy technologies in conjunction with the TAP.7 architecture and present a unified interface to the DTS. Each of these scan topologies is considered a branch of the system scan topology. It also allows the use of TAP.1s with these TAP.7 topologies, provided separate Test Clock signals are used for the TAP.1 Branch and TAP.7 Branches. The TAP.7 architecture also provides a means to operate other technologies with a dedicated clock and single control signal using the same DTS connectivity as the TAP.7. A wrapper can be placed around these technologies to make them compatible with the TAP.7 architecture.

A system where Series, Star-4, and Star-2 Scan Topologies share a DTS connection with a TAP.1 Series Scan Topology and one or more other technologies is shown in Figure 1-9. The DTS can select and use the Series, Star-4, and Star-2 Scan Topologies and other technologies using this connectivity. Two of these technologies are based on IEEE Std 1149.1 and IEEE Std 1149.7. This standard describes how TAP.1s and TAP.7s based on these standards operate within this system environment.



**Figure 1-9 — A system with TCK(C) and TMS(C) shared between TAP.7 and technologies**

When multiple TAP.7 Branches are deployed, the TAP.7 Controllers may be trained to recognize the branch in which they reside. Once this training is complete, the DTS can select individual TAP.7 Branches as each branch appears to be a separate technology. A means to select all TAP.7 Branches is provided for test purposes.

Each branch shown in Figure 1-9 may be operated individually, whereas other branches are dormant (their TAP.7 Controllers are Offline). The TAP.7 Branches may also be operated together, or the TAP.1 and TAP.7 Branches may also be operated together. This flexibility satisfies both debug needs and provides for the use of boundary-scan testing across components in any of the TAP.1 and TAP.7 Branches.

The TAP.7 architecture defines the attributes that allow operation in the system configuration shown in Figure 1-9. Wrapping legacy technologies with a small block of logic incorporating these attributes makes them operable in this configuration. Changes to the normal operation of these technologies are not required as this logic merely provides a selection mechanism and does not change the operation of the technology when it is selected.

### 1.12 Document content

Subsequent clauses of the document contain the specifications for particular features of this standard. Two classes of material are contained in these clauses:

- Description
- Specifications

### 1.12.1 Descriptive material

Material that is not contained in subclauses titled “Specifications” is descriptive material that illustrates the need for the features being specified, the operation of these features, or the application of these features. This material includes descriptive text, schematics, state diagrams, and flowcharts that illustrate the operation of, or a possible implementation of, the specifications in this standard. The descriptive material may also include a description of design decisions made during development of this standard. Material within the description that is explicitly referenced from a Specifications subclause is considered normative material and included in the Specifications by reference.

The descriptive material within this standard also describes the operation of the DTS that interacts with the IEEE 1149.7 Technology. Throughout this specification, requirements for the DTS to properly operate with this technology are stated as DTS operational requirements. These requirements are expressed in a manner that differs from the terminology used to describe the operation of the IEEE 1149.7 Technology. While outside the scope of this specification, these requirements are provided for the proper use of the IEEE 1149.7 Technology. These requirements should be viewed as statements of fact and are usually expressed in statements with the word “will.”

NOTE—The descriptive material contained in this standard is for illustrative purposes only and does not define a preferred implementation. Implementation examples are provided throughout this standard to illustrate the operation of certain features. If discrepancies between examples and illustrations in a description subclause and material in a specifications subclause occur, the specifications always take precedence. Readers should exercise caution when using these examples as an implementer is responsible for ensuring an implementation exhibits IEEE 1149.7-Specified Behavior. In particular, it is emphasized that the examples are included to effectively communicate the meaning of this standard. As such, they are logically correct; however, a particular implementation may not operate properly with respect to timing and other parametric characteristics.

### 1.12.2 Specification material

Subclauses titled “Specifications” contain the rules, recommendations, and permissions that define this standard using the Word Usage definitions outlined in 1.3:

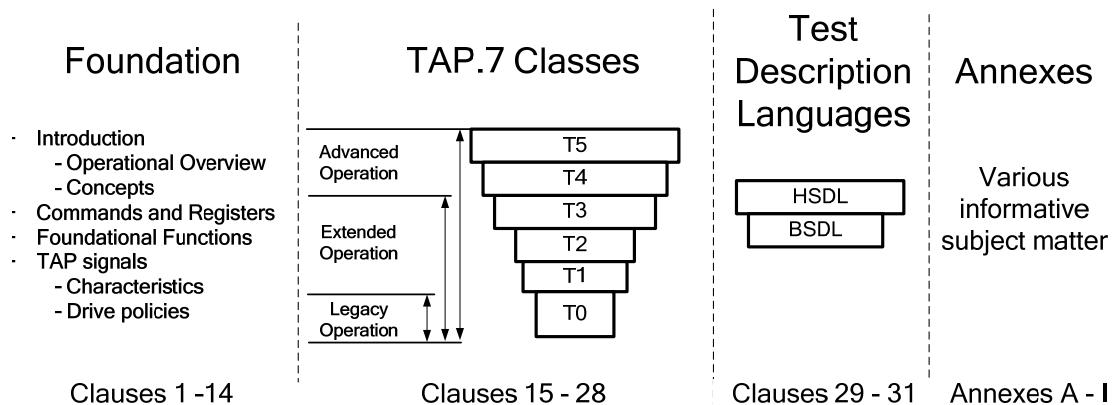
- a) Rules contain the word “shall” and specify the mandatory aspects of this standard.
- b) Recommendations contain the word “should” and indicate preferred practice for designs that seek to conform to this standard.
- c) Permissions contain the word “may” and show how optional features may be introduced into a design that seeks to conform to this standard. These features will extend the capability delivered by this standard.
- d) References to material not included in a specification subclause shall be deemed included in the specification subclause by reference when they are explicitly referenced by a Rule, Recommendation, or Permission.

## 1.13 Document organization

### 1.13.1 Partitioning

This standard contains a large volume of information. It is partitioned in a manner that allows the reader to move among the various levels of this specification depending on the reader’s background, knowledge, and interest. A user that is familiar with the concepts, terminology, and capabilities of the TAP.7 architecture may elect to skip Clause 2 through Clause 8, Clause 15, Clause 17, Clause 21, and Clause 22. These clauses contain only background information and do not contain rules. A reader that is not familiar with the TAP.7 architecture may use the description of the standard in this subclause to determine the clauses that appear to be required reading.

The organization of this document allows the reader to explore TAP.7 concepts, class functionality, and detailed operation, to the level desired. As the subject areas change, some conceptual material is repeated to provide better continuity of descriptions. The organization of this standard is shown in Figure 1-10.



**Figure 1-10 — IEEE 1149.7 standard organization**

### 1.13.1.1 Foundation

Clause 1 through Clause 14 describe major aspects of the TAP.7 capability in a top-down manner, with rules beginning in Clause 9.

Clause 1 through Clause 3 provide:

- An introduction
- Normative references
- Definitions, acronyms, and abbreviations

Clause 4 through Clause 8 provide a high-level view of:

- TAP.7 concepts (Clause 4)
- T0–T3 and T4–T5 TAP.7 operation (Clause 5 and Clause 6)
- System concepts (Clause 7)
- Selection concepts (Clause 8)

These four clauses highlight the differences in the TAP.7 architecture and its predecessor, the TAP.1 architecture created with IEEE Std 1149.1. They provide a description of the TAP.7 architecture's support for the Series, Star-4, and Star-2 Scan Topologies, along with the sharing of the TCK(C) and TMS(C) signals with other technologies. They also describe how the hierarchy of TAP.7 Controllers (ADTAPC, CLTAPC, and EMTAPCs) is selected and used within the Series, Star-4, and Star-2 Scan Topologies, or a mixture of the Scan Topologies. Reading this subject matter may be sufficient to gain a high-level understanding of the TAP.7 capability.

Clause 9 through Clause 14 provide a description of fundamental TAP.7 capability sequentially covering:

- Registers, commands and registers/scan paths (Clause 9)
- Ancillary services such as resets and signaling methods (Clause 10)
- Offline/Online capability (ADTAPC selection and deselection; Clause 11)
- TAP.7 signals and their drive policies (Clause 12 through Clause 14)

Four additional concepts clauses (Clause 15, Clause 17, Clause 21, and Clause 22) provide a more detailed overview of legacy (T0), extended (T1–T3), and advanced (T4–T5) concepts and operation. These clauses may also be read to enhance one's understanding of the TAP.7 architecture's capability. The reader should not have to read clauses that describe the capabilities of a TAP.7 Class that is above the class that is implemented.

### 1.13.1.2 TAP.7 Classes

Clause 15 through Clause 28 describe the incremental deployment and use of the capabilities described in Clause 1 through Clause 14. Clause 15, Clause 17, Clause 21, and Clause 22 supplement the description of the foundation as mentioned previously with a mid-level view of this capability. These clauses precede clauses describing TAP.7 Classes. The first of these concept clauses, Clause 15, describes the concepts required to provide IEEE 1149.1-Specified Behavior and introduces the T0 TAP.7 capability class. The second concepts clause, Clause 17, describes the Extended Protocol concepts needed to support the T1, T2, and T3 TAP.7 capability classes. The third concepts clause, Clause 21, introduces the Advanced Protocol concepts needed to support the T4 and T5 TAP.7 capability classes.

The TAP.7 Classes are described in:

- T0 TAP.7 (Clause 16)
- T1 TAP.7 (Clause 18)
- T2 TAP.7 (Clause 19)
- T3 TAP.7 (Clause 20)
- T4 TAP.7 (Clause 23 through Clause 26)
- T5 TAP.7 (Clause 27 and Clause 28)

### 1.13.1.3 Test description languages

Clause 29 through Clause 31 form the test description languages section of the document. Clause 29 provides an overview of test concepts followed by Clause 30 and Clause 31 describing two test descriptions languages (IEEE 1149.7 BSDL and IEEE 1149.7 HSDL). The IEEE 1149.7 BSDL is an IEEE 1149.7-specific Boundary-Scan Description Language (BSDL) that is a superset of the IEEE 1149.1 BSDL. The HSDL provides a means to describe TAPC connectivity and TAPC hierarchy.

### 1.13.1.4 Annexes

A number of annexes are included. Their subject matter includes:

- IEEE 1149.1 reference material
- Scan examples in timing diagram form
- Scan examples in tabular form
- Programming considerations
- Recommended electrical characteristics
- Connectivity and electrical recommendations
- Utilizing Segmented Scan (SScan) Formats
- The RTCK signal
- Bibliography

### 1.13.2 Pictorial view

A pictorial view of the document organization is shown in Figure 1-11.

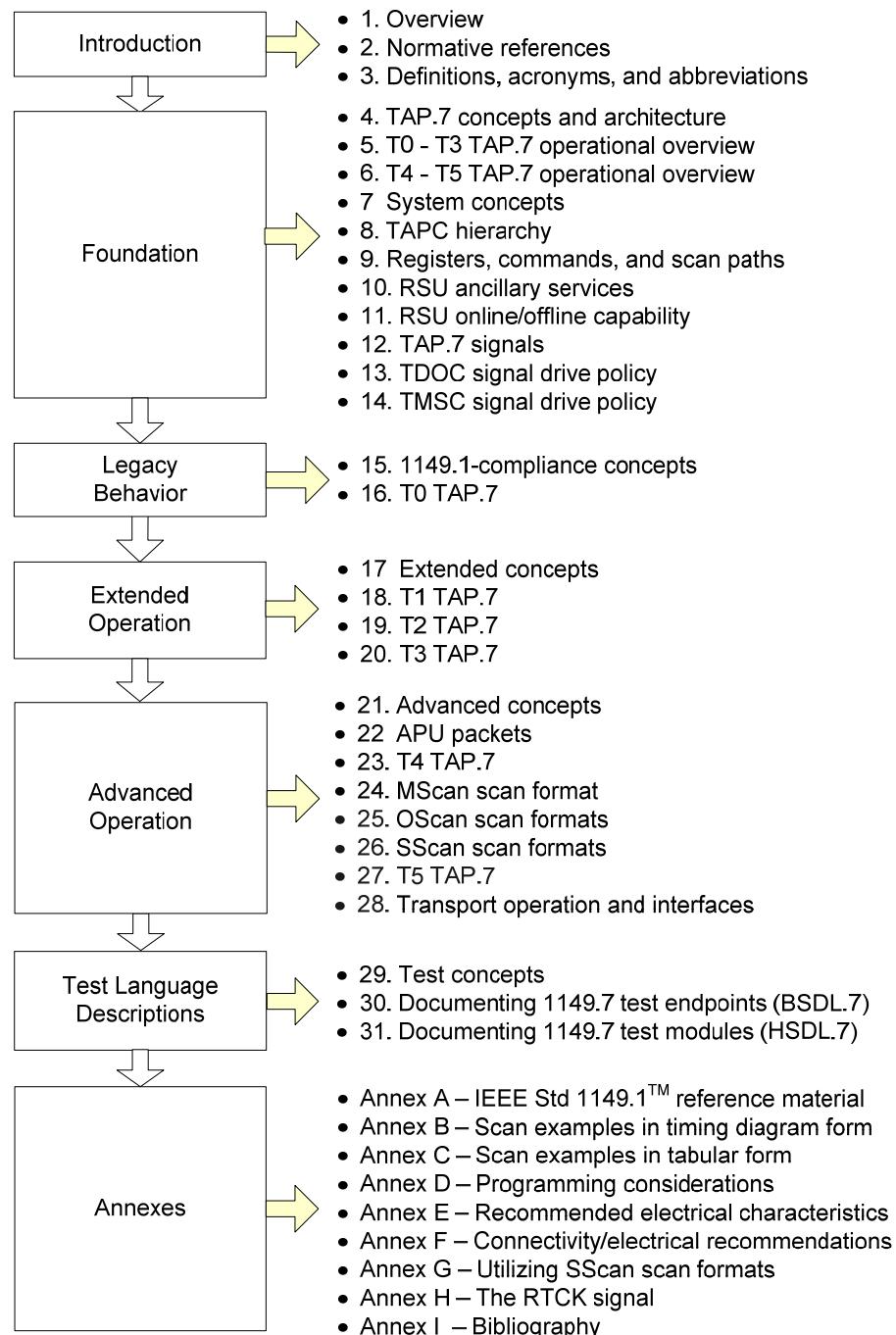


Figure 1-11 — Document organization

## 1.14 Using the standard

### 1.14.1 User background knowledge

The reader of this document should be familiar with IEEE Std 1149.1 and have knowledge of the operation of DTS, the operation of TS, and the connection of a DTS and TS.

### 1.14.2 Types of users

This standard provides information needed by chip designers, board designers, and programmers. Additionally, the test community may be interested in a subset of the capability provided by the TAP.7 Class. Selective reading pertaining to mandatory features and optional features affecting the use of the TAP (for example, selection and power management) is recommended.

#### 1.14.2.1 Chip designer

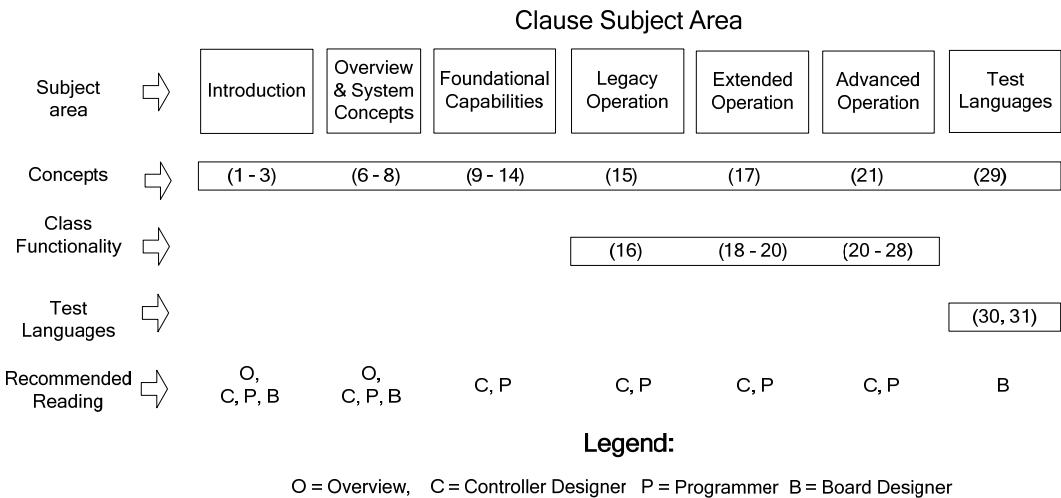
Clause 1 through Clause 28 provide a chip designer the information needed to design the TAP.7 Controller. Annex E provides recommendations for electrical timing. The clauses up to and through the TAP.7 Class being designed are relevant. Annex H provides recommendations for dealing with IP with a Return Test Clock. It is **strongly** recommended that the chip design recommendations in Annex F be reviewed and followed.

#### 1.14.2.2 Programmer

Clause 1 through Clause 8 provide a programmer an overview of the use of the TAP.7 architecture. Clause 10 and Clause 11 cover resets and selection mechanisms. The concepts clauses (Clause 15, Clause 17, Clause 21, and Clause 22) may also be useful as mentioned previously. Clause 9 provides a description of the TAP.7 Controller commands and registers. Clause 18 provides information on optional features such as resets and power management supported by T1 and above TAP.7s. Clause 19 provides information on TAP startup modes and the bypass of the System Test Logic for T2 and above TAP.7s. Clause 20 provides information related to the operation of a T3 and above TAP.7 in a Star-4 Scan Topology. Clause 23 provides information on a T4 TAP.7, with Clause 24 through Clause 26 detailing the operation of the scan portion of the two-pin operation. Clause 27 provides information on non-scan transfers with Clause 28 detailing its operation. Clause 29 through Clause 31 cover test concepts and test description languages.

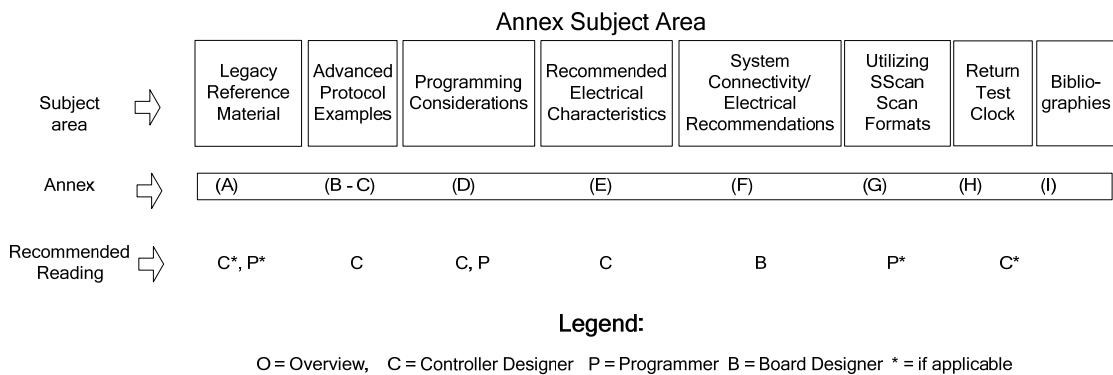
### 1.14.3 Use summary

The recommended reading for different user classes is summarized in Figure 1-12.



**Figure 1-12 — Clause subject areas**

The nine annexes of this document, shown in Figure 1-13, provide practical information covering the design and use of the TAP.7 architecture. The reader is strongly encouraged to utilize the TAP timing recommendations in Annex E and the system connectivity electrical recommendations in Annex F.

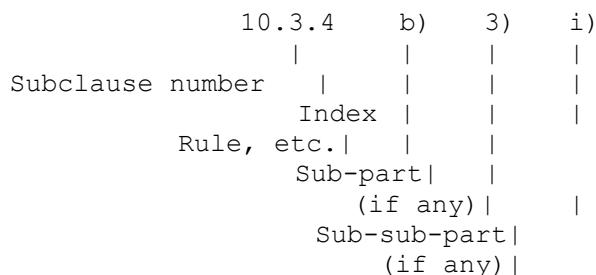


**Figure 1-13 — Annex subject areas**

## 1.15 Conventions

The following conventions are used in this standard:

- a) The rules, recommendations, and permissions in each Specification subclause are contained in a single alphabetically indexed list. References to each rule, recommendation, or permission are shown in the form:



- b) Instruction and state names defined in this standard are shown in italic type in the text of this standard.
- c) IEEE 1149.7 register names are capitalized when used in the description and specification sections.
- d) Names of state and signals that pertain to the test Data Registers referenced by this standard contain the characters DR, whereas those that pertain to the Instruction Register referenced by this standard contain the characters IR.
- e) Names for signals that are active in their low state have an n as the initial character, e.g., nTRST.
- f) A positive logic convention is used; i.e., a logic 1 signal is conveyed as the more positive of the two voltages used for logic signals.
- g) A TAP Controller State Machine state that is followed by *\_f* (for example, *Update-DR\_f*) is the TAP state created with the rising edge of a clock delayed to the falling edge of the clock (the signal changes on the falling edge of the clock).
- h) The classes to which a rule applies will be expressed in one of the following manners:
  - 1) For T4 and above TAP.7s, the XY shall ...
  - 2) For T3 and below TAP.7s, the XY shall ...
  - 3) For T0 TAP.7s, the XY shall ...
  - 4) For T1 and T2 TAP.7s, the XY shall ...
  - 5) For T1, T2, and T3 TAP.7s, the XY shall ...
- i) A rule with two conditions could be expressed as follows:
  - 1) For T4 and above TAP.7s, the XY shall AB when either C or D.
  - 2) For T4 and above TAP.7s, the XY shall AB when C and D.
- j) Presentation shown in Convention i) above shall be equivalent to the following:
  - 1) For T4 and above TAP.7s, when either C or D, the XY shall AB.
  - 2) For T4 and above TAP.7s, when C and D, the XY shall AB.
- k) Presentation shown in Convention i) 1) above shall be equivalent to the following:
  - 1) For T4 and above TAP.7s, the XY shall AB when any of the following are true:
    - i) C.
    - ii) D.
- l) A rule may define subsequent rules as applicable to a class to a range of classes.
- m) Wherever possible tables are used to express complex logical expression because they are:
  - 1) Easier to read.
  - 2) Facilitate creation of a test plan.
  - 3) Provide a structure to create logical expressions.
- n) The logical expression is well suited to the use of lists and the words **ANY** and **ALL** where:
  - 1) Any means a logical OR.
  - 2) All means a logical AND.

NOTE—The following examples illustrate the representation of logical expressions.

- o) For example, requiring that XY occurs as a result of the following logical expression ( $A \& B \& C \& (D|E)$ ) is expressed as:

XY shall occur provided all of the following are true:

- 1) A.
- 2) B.
- 3) C.
- 4) Any of the following are true:
  - i) D.
  - ii) E.

- p) For example, requiring that XY occurs as a result of the following logical expression ( $(A|B | C | (D\&E))$ ) is expressed as:

XY shall occur provided any of the following are true:

- 1) A.
- 2) B.
- 3) C.
- 4) All of the following are true:
  - i) D.
  - ii) E.

- q) For example, requiring that XY occurs as a result of the following logical expression ( $((A|B) \& C \& (D|E))$ ) is expressed as:

XY shall occur provided all of the following are true:

- 1) Any of the following are true:
  - i) A.
  - ii) B.
- 2) C.
- 3) Any of the following are true:
  - i) D.
  - ii) E.

- r) Miscellaneous terminology:

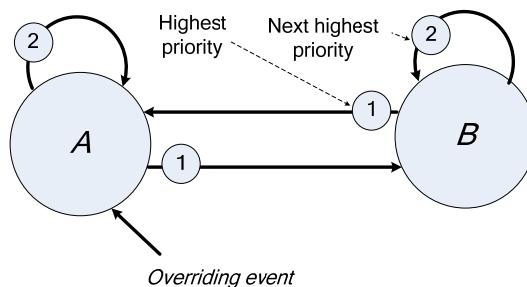
- 1) 1–3 means 1, 2, and 3.
- 2) 1, 4–7 means 1, 4, 5, 6, and 7.
- 3) [n] means a single element of a set of indexes.
- 4) [m:n] a set of indexes from m to n including m and n.
- 5) [ITEM] an item enclosed in brackets does not occur in all cases.
- 6)  $\sim$  means the inverse of.
- 7)  $\neq$  means not equal.
- 8)  $\&$  means the logic AND of Boolean terms.

- 9) | means the logical OR of Boolean terms.
- 10) A single equals sign (=) means an assignment.
- 11) A double equals sign (==) means equivalent to.
- 12) N/A means not applicable.
- 13) An x within a table entry means “Don’t Care”.
- 14) — within a table entry means not applicable.
- 15) ! means logical not.
- 16) A signal name ending in \_R means the signal is a register output.
- t) The TAPC state names aliases shown in Table 1-3 shall be considered as having the relationship to the TAPC state name(s) shown in this table.

**Table 1-3 — Relationship TAPC state name alias/TAPC state**

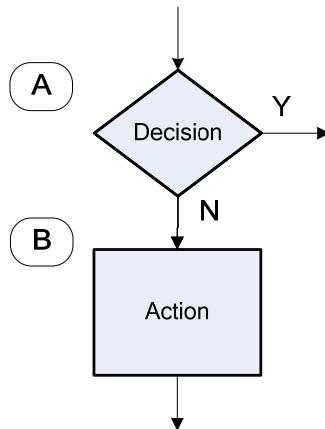
Alias	TAPC state
TLR	<i>Test-Logic-Reset</i>
RTI	<i>Run-Test/Idle</i>
<i>Select-xR-Scan</i>	<i>Select-IR-Scan</i> or <i>Select-DR-Scan</i>
<i>Capture-xR</i>	<i>Capture-IR</i> or <i>Capture-DR</i>
<i>Shift-xR</i>	<i>Shift-IR</i> or <i>Shift-DR</i>
<i>Exit1-xR</i>	<i>Exit1-IR</i> or <i>Exit1-DR</i>
<i>Pause-xR</i>	<i>Pause-IR</i> or <i>Pause-DR</i>
<i>Exit2-xR</i>	<i>Exit2-IR</i> or <i>Exit2-DR</i>
<i>Update-xR</i>	<i>Update-IR</i> or <i>Update-DR</i>

- u) State diagrams are drawn with arcs having precedence. The arcs are numbered by priority as shown in Figure 1-14.



**Figure 1-14 - Arc priority**

- v) Flowchart decisions or actions that are expanded in another flowchart are identified with a symbol containing a letter as shown in Figure 1-15 or a reference to another flowchart within the flowchart. A symbol provides an index into a table that provides a cross reference.



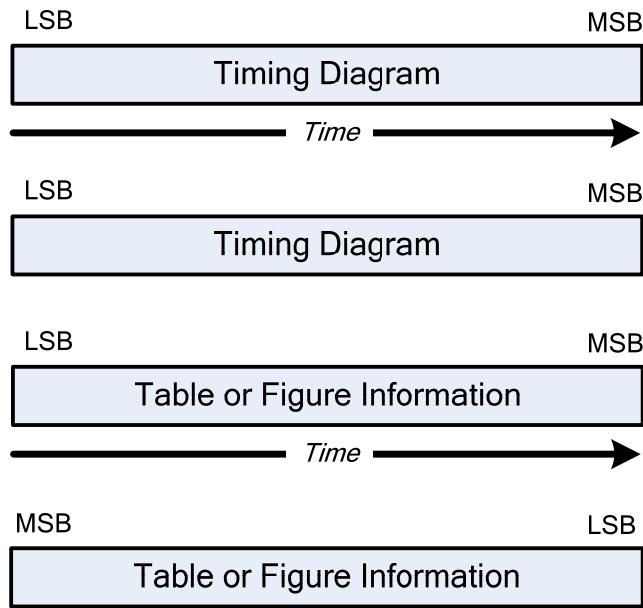
**Figure 1-15 - Flowchart symbol expansion**

- w) Tables showing cause and effect follow the format shown in Table 1-4. The inputs are to the left, and the outputs are to the right. The output headings are shaded gray with a dark line separating inputs and outputs. In some cases tables have more than one set of inputs and outputs side by side.

**Table 1-4 — Table format**

A	B	C	D	E

- x) The most significant bit (MSB)/least significant bit (LSB) relationships depicted in tables, figures, and timing diagrams are represented as shown in Figure 1-16. The timing diagram begin with the earliest time and LSB left most in the diagram with time proceeding from left to right. Text or figure that has a “time arrow” included also follows this convention. Text or figure information that does not include a “time arrow” depicts a value with the MSB being the left-most position in the figure and the LSB being the right-most position in the figure.



**Figure 1-16 - MSB/LSB conventions**

- y) Italicized words with a minimum of one capital letter are state names.
- z) Italicized words with no capital letter are signal names.

## 2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE Std 1149.1™, IEEE Standard Test Access Port and Boundary-Scan Architecture.<sup>5, 6</sup>

---

<sup>5</sup> IEEE publications are available from The Institute of Electrical and Electronics Engineers (<https://standards.ieee.org/>).

<sup>6</sup> The IEEE standards or products referred to in Clause 2 are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated.

### 3. Definitions, acronyms, and abbreviations

#### 3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online*<sup>7</sup> should be referenced for terms not defined in this clause.

**3.1.1 1149.1-Non-disruptive Behavior** (see 1.7.2).

**3.1.2 1149.1-Other Behavior** (see 1.7.2).

**3.1.3 1149.1-Specified Behavior** (see 1.7.2).

**3.1.4 1149.7-Non-disruptive Behavior** (see 1.7.2).

**3.1.5 1149.7-Other Behavior** (see 1.7.2).

**3.1.6 1149.7-Specified Behavior** (see 1.7.2).

**3.1.7 active:** When associated with logic level (e.g., active-low), this term identifies the logic level to which a signal shall be set to cause the defined action to occur. When referring to an output driver (e.g., an active drive), this term describes the state in which the driver is capable of controlling the voltage of the network to which it is connected.

**3.1.8 Advanced Protocol:** The signaling that directs the Test Access Port Controller (TAPC) State Machine state progression and conveys Test Data Input (TDI) and Test Data Output (TDO) information using only the Test Clock Compact (TCKC) and Test Mode Select Compact (TMSC) signals as specified by this standard.

**3.1.9 Advanced Protocol Unit (APU):** The circuitry that handles the Advanced Protocol signaling.

**3.1.10 aliasing:** The process of allocating and de-allocating 1 of 16 four-bit aliases [Controller Identification Codes (CIDs) to a TAP.7 Controller Address (TCA)], making the use of TAP.7 Controller commands and Scan Selection State management more efficient.

**3.1.11 allocation:** The process that is used to associate 1 of 16 four-bit aliases (CID—Controller Identification Codes) with a TAP.7 Controller Address (TCA).

**3.1.12 asserted:** When associated with logic level, this term identifies the logic level to which a signal shall be set to cause the defined action to occur.

**3.1.13 Background Data Transfer (BDX):** The transfer of non-scan information using the Test Mode Select Compact (TMSC) signal by the data transfer function defined by this specification where the direction of the transfer is predefined.

**3.1.14 bidirectional signal:** A signal that can be used to either send information to another destination or receive information from another source.

**3.1.15 capture:** Load a value into a Data Register or the Instruction Register as a consequence of entry into the *Capture-DR* or *Capture-IR* TAP controller state, respectively.

---

<sup>7</sup> The *IEEE Standards Dictionary Online* is available at <https://dictionary.ieee.org/>. An IEEE Account is required for access to the dictionary, and one can be created at no charge on the dictionary sign-in page.

**3.1.16 clock:** A signal where transitions between the low and high logic levels (or vice versa) are used to indicate when a stored-state device, such as a Flip-Flop or latch, may perform an operation.

**3.1.17 command:** A 12-bit binary data value created by a pair of Data Register Scans that cause the TAP.7 Controller to perform an action.

**3.1.18 compliant to a standard:** Obeying all applicable rules of the standard.

**3.1.19 Controller Identification Code (CID):** A four-bit code assigned to a TAP.7 Controller to allow it to be uniquely identified in a Star Scan Topology.

**3.1.20 control level:** A state created by certain Test Access Port Controller (TAPC) state sequences that permits a TAP.7 Controller to interpret Data Register Scan activity in a manner that differs from that defined by IEEE Std 1149.1.

**3.1.21 Control Protocol:** The signaling that directs the selection of TAP.7 Controllers and other technologies. This signaling includes the checks to determine whether the TAP.7 Controller supports the Standard/Advanced Protocols that is about to be used following register programmation.

**3.1.22 coupled:** The Scan Selection State when the state of the chip-level Test Access Port Controller (CLTAPC) and adapter TAPC (ADTAPC) move in lock-step, i.e., with matching states, through a progression of TAPC states.

**3.1.23 control level lock:** The first *Shift-DR* TAPC state following the zero-bit DR Scans (ZBSs) incrementing the control level to one or higher when the operation of the TAP.7 Controller has not been suspended.

**3.1.24 Custom Data Transfer (CDX):** The transfer of information using the Test Mode Select Compact (TMSC) signal by a custom function with this function defining the transfer direction on a bit-by-bit basis.

**3.1.25 custom commands:** The TAP.7 commands whose function is not defined by this specification and whose function may be defined uniquely for a chip subject to the restrictions set forth in this specification.

**3.1.26 de-asserted:** When associated with logic level, this term identifies the logic level to which a signal shall be set to not cause the defined action to occur.

**3.1.27 decoupled:** The Scan Selection State when the state of the chip-level Test Access Port Controller (CLTAPC) and adapter TAPC (ADTAPC) do not move in lock-step, i.e., with the ADTAPC state moving through a progression of TAPC states while the CLTAPC state does not change.

**3.1.28 Escape:** A special signaling sequence where Test Mode Select Compact (TMSC) signal edges are counted to convey control information while the Test Clock Compact (TCKC) signal is a logic 1.

**3.1.29 Extended Protocol:** The IEEE 1149.1 protocol sequences that are used to implement the TAP.7 Controller capabilities such as control levels and commands.

**3.1.30 falling edge:** A transition from a high to a low logic level. In positive logic this means a change from a logic 1 to a logic 0. Events that are specified to occur on the rising or falling edge of a signal should be completed within a fixed (frequency-independent) delay specified by the component supplier.

**3.1.31 false:** When associated with logic level (e.g., active-low), this term identifies the logic level to which a signal shall be set to not cause the defined action to occur.

**3.1.32 hard chip-level reset:** The reset that causes the maximum initialization of Chip-Level Logic.

**3.1.33 high:** The higher of the two voltages used to convey a logic level. This means a logic 1 for positive logic.

**3.1.34 high impedance:** When referring to an output driver (e.g., an inactive drive), this term describes the state in which the driver is not capable of sourcing or sinking current.

**3.1.35 higher TAP.7 Class:** When referring to a first TAP.7 Class, it shall be considered higher (above) than a second TAP.7 Class when the class number of the first class is greater than the class number of the second class.

**3.1.36 inactive:** When associated with logic level (e.g., active-low), this term identifies the logic level to which a signal shall be set to cause the defined action not to occur. When referring to an output driver (e.g., an inactive drive), this term describes the state in which the driver is not capable of controlling the voltage of the network to which it is connected.

**3.1.37 initiator:** An entity that is responsible for launching a transaction.

**3.1.38 initiator/responder:** A relationship between entities in a system where initiators are responsible for launching transactions and responders are responsible for satisfying the transaction requests.

**3.1.39 input signal:** A signal that receives information through a connection to an external source.

**3.1.40 instruction:** A binary data word shifted serially into the test logic in order to define its subsequent operation.

**3.1.41 keeper behavior:** Input signal circuitry that, in the absence of an input being driven, produces a logic response identical to the application of the last logic level driven on the input.

**3.1.42 least significant bit (LSB):** The digit in a binary number representing the lowest numerical value. For shift registers, the bit located nearest to the serial output, or the first bit to be shifted out. The least significant bit of a binary word or shift register is numbered 0.

**3.1.43 least significant bits (LSBs):** The digits in a binary number beginning with the LSB and moving toward the MSB of the binary number.

**3.1.44 legacy:** The function specified by IEEE Std 1149.1.

**3.1.45 low:** The lower of the two voltages used to convey a logic level. This means a logic 0 for positive logic.

**3.1.46 lower TAP.7 Class:** When referring to a first TAP.7 Class, it shall be considered lower (below) than a second TAP.7 Class when the class number of the first class is less than the class number of the second class.

**3.1.47 most significant bit (MSB):** The digit in a binary number representing the greatest numerical value. For shift registers, this bit is the bit farthest from the serial output or the last bit to be shifted out. Logic values expressed in binary form are shown with the most significant bit on the left.

**3.1.48 most significant bits (MSBs):** The digits in a binary number beginning with the MSB and moving toward the LSB of the binary number.

**3.1.49 non-biased behavior:** Input signal circuitry that, in the absence of an input being driven, does not produce a specific logic response.

**3.1.50 nonclock:** A signal where the transitions between the low and high logic levels do not themselves cause operation of stored-state devices, such as a Flip-Flop or latch.

**3.1.51 output signal:** A signal that sends information through a connection to an external destination.

**3.1.52 packet:** A number of bits of information transmitted serially in a specified order representing data, control information, or both. A packet may be as short as one bit.

**3.1.53 parked:** A Test Access Port Controller's (TAPC's) state is prevented from leaving a specific state by some means.

**3.1.54 pin:** The point at which connection is made between the integrated circuit and the substrate on which it is mounted (e.g., the printed circuit board). For packaged components, this would be the package pin; for components mounted directly on the substrate, this would be the bonding pad.

**3.1.55 private:** A design feature intended solely for use by the component manufacturer.

**3.1.56 public:** A design feature, documented in the component data sheet, which may be used by purchasers of the component.

**3.1.57 pull-down behavior:** Input signal circuitry that, in the absence of an input being driven, produces a logic response identical to the application of a logic 0. This is a weak logic 0 and is designed to be driven to a logic 1 by another device.

**3.1.58 pull-up behavior:** Input signal circuitry that, in the absence of an input being driven, produces a logic response identical to the application of a logic 1. This is a weak logic 1 and is designed to be driven to a logic 0 by another device.

**3.1.59 ready:** An indication that logic circuit may proceed with its next action when true, and wait to perform this action when false.

**3.1.60 reset:** The establishment of an initial logic condition that can be either a logic 0 or a logic 1, as determined by the context.

**3.1.61 Reset and Selection Unit (RSU):** The circuitry that handles detection of Selection Alert Bit Sequences and Escapes used for the reset, deselection, and selection of the TAP.7 Controller.

**3.1.62 responder:** An entity that is responsible for responding to a transaction request.

**3.1.63 rising edge:** A transition from a low to a high logic level. In positive logic, this is a change from a logic 0 to a logic 1. Events that are specified to occur on the rising (falling) edge of a signal should be completed within a fixed (frequency-independent) delay specified by the component supplier.

**3.1.64 running:** A Test Access Port Controller's (TAPC's) state is allowed to change state, which is the opposite of parked.

**3.1.65 scan design:** A design technique that introduces shift register paths into digital electronic circuits and thereby improves their testability.

**3.1.66 scan format:** A value that defines the content of information sent to represent the activity associated with a TAPC state.

**3.1.67 Scan Packet:** A number of bits spanning one or more Test Clock Compact (TCKC) periods that represents the Test Mode Select (TMS), Test Data Input (TDI), and Test Data Output (TDO) information, or a subset thereof associated with a TAPC state. A Scan Packet may also contain information that is utilized only by the Debug and Test Systems (DTS) and TAP.7 Controller.

**3.1.68 scan path:** The shift register path through a circuit designed using the scan design techniques.

**3.1.69 Standard Protocol:** The signaling 1 that directs the TAPC State Machine state progression as defined by IEEE Std 1149.1, with data being transferred using the Test Data Input (TDI) and Test Data Output (TDO) signals.

**3.1.70 system:** Pertaining to the non-test function of the circuit.

**3.1.71 system logic:** Any item of logic that is dedicated to realizing the non-test function of the component or is at the time of interest configured to achieve some aspect of the non-test function.

**3.1.72 System Test Logic:** Any item of logic other than that included in the TAPC Controller that is dedicated to realizing the test function of the component or is at the time of interest configured to achieve some aspect of the test function.

**3.1.73 system pin:** A component pin that feeds, or is fed from, the on-chip system logic.

**3.1.74 Test Clock (Compact) TCK(C) period:** The time measured from one falling edge of the Test Clock to the next falling edge of the Test Clock as measured when there is no special signaling (an Escape) associated with this TCK(C) period.

**3.1.75 test logic:** Any item of logic that is a dedicated part of the test logic architecture or is at the time of interest configured as a part of the test logic architecture.

**3.1.76 true:** When associated with logic level (e.g., active-low), this term identifies the logic level to which a signal shall be set to cause the defined action to occur.

**3.1.77 update:** Transfer of logic value from the shift register stage of a Data Register cell or an Instruction Register cell into the latched parallel output stage of the cell as a consequence of the falling edge of the Test Clock input in the *Update-DR* or *Update-IR* controller state, respectively.

**3.1.78 zero-bit Data Register (DR) Scan:** A sequence of Test Access Port Controller (TAPC) states that begins with a *Select-DR-Scan* state and moves to an *Update-DR* state without the occurrence of a *Shift-DR* TAPC state between the *Select-DR-Scan* and *Update-DR* states.

## 3.2 Acronyms and abbreviations

Acronyms for TAPC Controller register names and commands are found in Table 9-1 and Table 9-3, respectively. Other acronyms and abbreviations are shown as follows:

AC	alternating current
ADTAPC	adapter TAPC (the TAPC within the Extended Protocol Unit)
ADV	advanced state
APFCS	auxiliary pin function control supported
APU	Advanced Protocol Unit
AT	aliasing target
BCE	boundary compliance enable
BDX	Background Data Transfer
BNF	Backus-Naur Form
BPA	Bypass Active

BSDL	Boundary-Scan Description Language
BSDL.1	Boundary-Scan Description Language defined by IEEE Std 1149.1
BSDL.7	Boundary-Scan Description Language defined by this standard
CDX	Custom Data Transfer
CGM	Conditional Group Membership
CGMC	Conditional Group Membership Count
CID	Controller Identification Code
CIDA	Controller ID Allocate
CIDD	Controller ID De-allocate
CIDI	Controller ID Invalid
CLASS	TAP.7 Class
CLTAPC	Chip-Level TAPC
CL6_OFFSETS	Control Level Six Offline supported/unsupported
CNFGFMT	Configuration Register Format
CP	Check Packet
CPA	Check Process Active
CPU	central processing unit
CSM	Control State Machine
DAS	deselected at startup
DCC	Data Channel Client
DCR	Data Channel Router
DIMM	Dual In-line Memory Module
DLYC	Delay Control
DP	Delay Packet
DR	Data Register
DTS	Debug and Test Systems
EMTAPC	embedded TAP controller
END	control bit inserted in a Scan Packet (SP) of a segment to end the segment
EPU	Extended Protocol Unit
EOT	End of Transfer—an Escape overlaid on a TMS bit in a segment to end a segment
ESC	Escape
ESCAPESS	Escapes supported
FIFO	first in, first out
HDR	Header
HSDL	Hierarchical Scan Description Language
HSDL.7	Hierarchical Scan Description Language defined by this standard
IC	integrated circuit

I/O	input/output
IP	intellectual property
IR	Instruction Register
JScan	JTAG Scan
LCA	Logical Channel Address
LDC	Logical Data Channel
LFSR	Linear Feedback Shift Register
LSB	least significant bit
MCM	Multi-Chip Module
MIPI	Mobile Industry Processor Interface
MSB	most significant bit
MSC	Make Scan Group Candidate
MScan	Multi-use Scan Format
MTCP	Multi-TAP Control Path
MTDP	Multi-TAP Data Path
NB	nonbiasing behavior where an undriven input assumes neither a logic 1 nor a logic signal value
NIP	No Interface Power
nTDI	Not Test Data Input
nTRST	Not Test Reset
nTRST_PD	Not Test Reset Pull-Down
<i>OLS</i>	Offline-at-Start-up option
<i>OLW</i>	Offline waiting state
ONE	logic 1 value inserted into a bit of an SP
OScan	Optimized Scan
OSCANS	OScans supported/unsupported
PMDSRC	Power-mode default source
PCA	Physical Channel Address
PCB	printed circuit board
PC0	Precharge 0
PC1	Precharge 1
PD	Power-down
PD Behavior	Pull-down behavior where an undriven input assumes a logic 0 signal value
PDC	Physical Data Channel
PDC0	Physical Data Channel 0
PDC1	Physical Data Channel 1
<i>PDWN</i>	TAP.7 power-control state when power-down is in progress

<i>P<sub>OFF</sub></i>	TAP.7 power-control state when the power is off
<i>P<sub>ON</sub></i>	TAP.7 power-control state when the power is on
PoP	package-on-package
POR	Power on Reset
PSGM	Potential Scan Group Member
PSGMCL	Potential Scan Group Membership Count Last EPU TCK period
PT	Payload Type
PU	pull-up behavior where an undriven input assumes a logic 1 signal value
<i>PUP</i>	TAP.7 power-control state when power-up is in progress
PWRMODE	power mode
PWRMODES	PWRMODES supported/unsupported
RDY	ready bits inserted in the SP payload output bit-frames as control information for STL stall opportunities
RDBKS	RDBACK supported
REV	TAP.7 specification revision
RSU	Reset and Selection Unit
RTCK	Return Test Clock
SCNB	Scan Bit
SCNFMT	Scan Format
SCNS	Scan String
SEL_ALERTS	Selection Alerts supported
SGCC	Scan Group Candidate Count
SGMCL	Scan Group Membership Count Last EPU TCK period
SiP	system in a package TAPC: TAP controller
SoC	System-on-Chip
SP	Scan Packet
<i>SPA</i>	Scan Packet Active
SScan	Segmented Scan
SSCANS	SScans supported/unsupported
SSD	Scan Selection Directive
SSM	Scan State Machine
STCKDCS	<i>sys_tck</i> duty cycle supported
STL	System Test Logic
STFMT	Store Format
STDCST	Store Data Channel State
STMC	Store Miscellaneous Control
SUSPEND	Suspend

TAP	Test Access Port
TAPC	Test Access Port Controller (the Finite State Machine) specified by IEEE Std 1149.1
TAPWIDS	TAP Width Default supported
TAPWLCK	TAP Width Locked option
TAP.1	IEEE 1149.1 TAP
TAP.7	IEEE 1149.7 TAP
TCA	TAP.7 Controller Address
TCK	Test Clock
TCKC	Test Clock Compact
TDI	Test Data Input
TDIC	Test Data Input Compact
TDO	Test Data Output
TDOC	Test Data Output Compact
TDO_OE	Test Data Output Enable
<i>TEST</i>	test state
TRESETS	TRESET supported
TMS	Test Mode Select
TMSC	Test Mode Select Compact
TP	Transport Packet
TPA	Transport Packet Active
TPP	Transport Protocol Processing
TRST*	Test Reset
TS	Target System
TSM	Transport State Machine
T0 TAP.7	Class 0 TAP.7
T1 TAP.7	Class 1 TAP.7
T2 TAP.7	Class 2 TAP.7
T3 TAP.7	Class 3 TAP.7
T4 TAP.7	Class 4 TAP.7
T5 TAP.7	Class 5 TAP.7
UUT	unit under test
VHDL	VHSIC Hardware Description Language
ZBS	zero-bit DR Scan
ZBSINH	ZBS Inhibit

## 4. TAP.7 concepts and architecture

### 4.1 Introduction

This clause provides a high-level description of the concepts used with the T0–T5 TAP.7s. It also provides a high-level description of the architecture and its partitioning. The concepts used within this standard support the standard's emphasis on the system architecture, pin efficiency, and capability vectors shown in Figure 1-1. Subsequent clauses provide a more detailed description of TAP.7 functionality, expanding the description of these concepts.

Key aspects of the TAP.7 architecture are listed as follows:

- Uses the IEEE Std 1149.1 as its foundation
- Delivers capability beyond that specified by the IEEE Std 1149.1
- Operates with IP having either IEEE 1149.1-Specified Behavior or IEEE 1149.7-Specified Behavior
- Insures the interoperability of components implemented using this standard
- Provides a framework for future extensibility and customization

IEEE 1149.7 users are encouraged to utilize the framework described above as its use should prevent the creation of IEEE 1149.7-Other Behavior. This is especially important as the large-scale integration of some of today's chips includes debug and test IP from a number of suppliers. This framework does not sanction IEEE 1149.1-Non-disruptive Behavior but is tolerant of it. It supports the deployment of IEEE 1149.1-based IP with IEEE 1149.1-Non-disruptive Behavior within a chip.

The subject matter within this clause is organized in the following manner:

- 4.2 Concepts supporting system architecture
- 4.3 Concepts supporting pin efficiency
- 4.4 Concepts supporting capability
- 4.5 IEEE 1149.7 architecture
- 4.6 Operating models

### 4.2 Concepts supporting system architecture

The concepts within the TAP.7 architecture that support test and debug within complex system architectures include but are not limited to the following:

- Transparency to existing IP (both hardware and software)
- A TAPC hierarchy
- Operation with Series, Star-4, and Star-2 Scan Topologies

#### 4.2.1 Maximizing compatibility with IEEE Std 1149.1

The TAP.7 architecture maximizes the compatibility with IEEE Std 1149.1 by building on the foundation of the TAP.1 architecture. It provides for the use of existing Hardware Software IP, with hardware layers merely added to support this standard. The proper partitioning of these additions can make the use of this standard compatible with the majority of the infrastructure already developed for IEEE Std 1149.1.

#### 4.2.1.1 Hardware components

The TAP.7 architecture maximizes compatibility with Hardware IP created with IEEE Std 1149.1 as follows:

- Using signaling concepts that are ignored by this IP
- Creating TAP.7 control mechanisms that operate in parallel with the scan paths within this IP
- Utilizing the IEEE 1149.1 *BYPASS* and *IDCODE* instructions in a manner that does not materially affect this IP
- Utilizing TDI data in non-*Shift-xR* states in a manner compatible with this IP
- Utilizing additional resets in a manner compatible with this IP

From the hardware standpoint, the TAP.7 architecture may be viewed as placing both DTS and TS IEEE 1149.7 adapters between existing DTS and TS 1149.1 components as shown in Figure 1-2.

#### 4.2.1.2 Software components

The TAP.7 architecture maximizes compatibility with Software IP supporting IEEE Std 1149.1 as follows:

- Preserving the use of all instructions and TAPC state sequences by this IP
- Maintaining the exact scan chain lengths of this IP when viewed from the TAP
- Managing the TAP.7 Controller capabilities using both IEEE 1149.1 TAPC state sequences and the signaling methods described in 4.3.1
- Providing for TAP.7 Controller management and scan topology management in an added software layer

From the software standpoint, the TAP.7 architecture may be viewed as placing an additional layer of TAP.7 management software between existing layers of the DTS software. This layer manages the TAP.7 architectures features, TAP hierarchy, and creation of a series-equivalent operation with either of the Star Scan Topologies.

#### 4.2.2 TAPC hierarchy

##### 4.2.2.1 Overview

The natural progression of integration trends has been as follows:

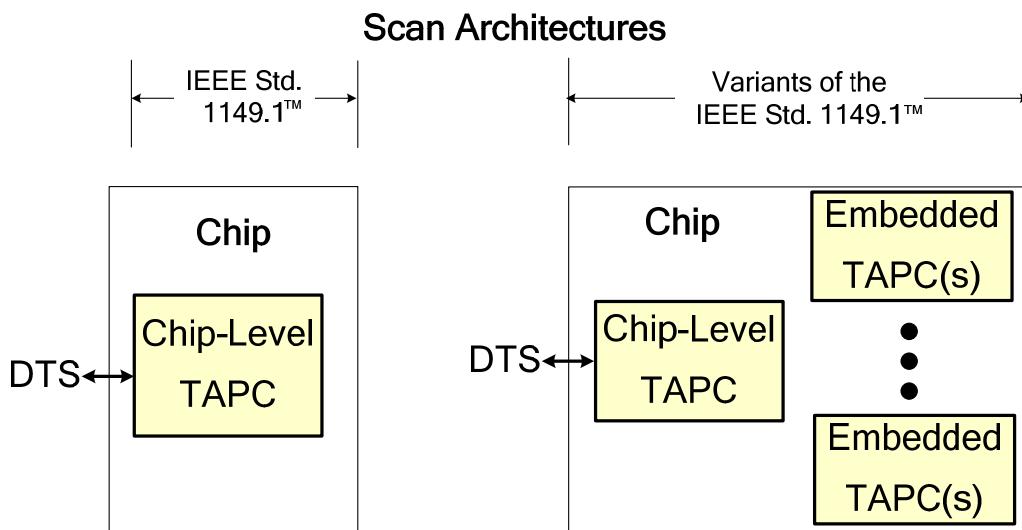
- Board functions become chips.
- Chips have larger and more complex subsystems.
- Subsystems are deployed in many chips.

Because of the constant time-to-market pressure, a large number of chip designers and integrators reuse existing designs to create complex system chips in a reasonable amount of time. Due to the large-scale reuse of existing hardware IP modules, current and future system chip designs do and will contain multiple TAPCs, to control all kinds of functionality, from boundary scan to internal debug control. Simply put, the use of multiple TAPCs on a chip is inevitable, in some cases.

This has created a conflict with IEEE Std 1149.1. Two scan architectures have emerged, the first consistent with that specified by IEEE Std 1149.1 and the second, a multiple-on-chip TAPC variant of the IEEE 1149.1 architecture. Since the use of multiple TAPCs within an on-chip scan architecture has been

defined by the chip vendors, the use of more than one TAPC within a scan architecture predates this specification. These variants of the IEEE 1149.1 architecture have, in some cases, incorporated multiple TAPCs on a chip in a manner that has created deviations from IEEE Std 1149.1.

Conceptual views of both IEEE Std 1149.1 and a multi-TAPC variant of the IEEE 1149.1 scan architecture are shown in Figure 4-1. Furthermore, a mix of these scan architectures is deployed in some instances. This diversity of scan architectures is at odds with the purpose of IEEE Std 1149.1.



**Figure 4-1 — Scan architectures utilizing IEEE Std 1149.1**

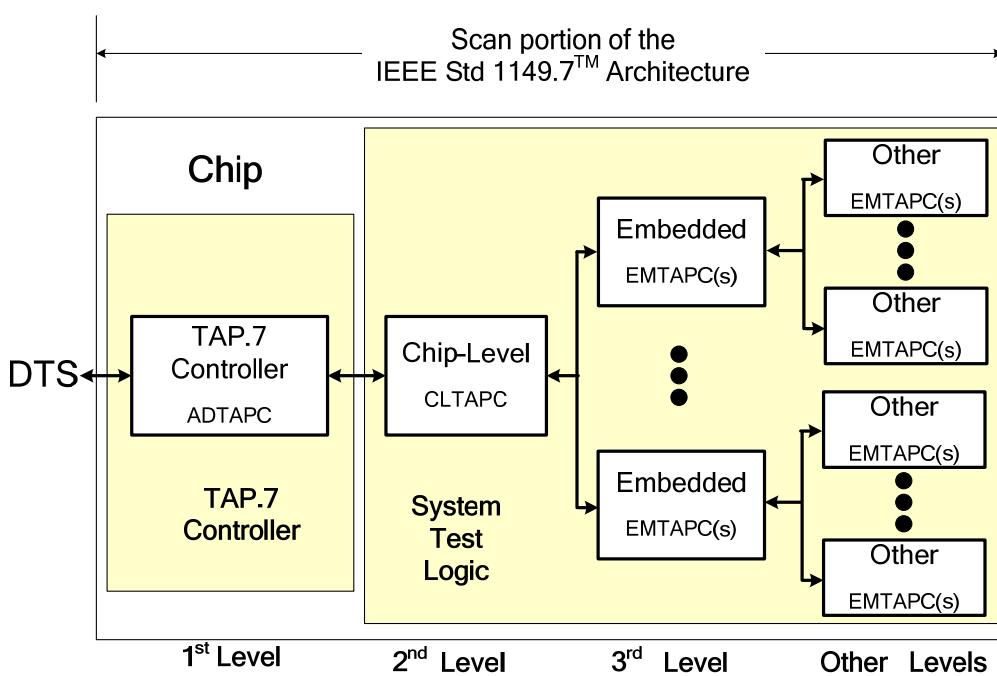
#### 4.2.2.2 Hierarchy levels

The TAPC hierarchy supports three or more levels of TAPC hierarchy as shown in Figure 4-2. These levels are as follows (from top level to deeper ones):

- 1<sup>st</sup> level      TAPC within the TAPC Controller (ADTAPC)
- 2<sup>nd</sup> level      TAPC at the chip level (CLTAPC)
- 3<sup>rd</sup> level      Embedded TAPCs (EMTAPC)
- Other levels    If needed

The TAPC hierarchy is supported in a manner that provides IEEE 1149.1-Specified Behavior. Note that this is quite different than the one TAPC per chip supported by IEEE Std 1149.1.

A TAPC at a specific level of the hierarchy is the parent of the TAPCs that it manages at the deeper hierarchy level below it. Likewise, the TAPC at a specific hierarchy level is a child of the TAPC managing it at the hierarchy level above it. The TAPC within the TAPC Controller is the child of the DTS TAPC or DTS entity controlling it.



**Figure 4-2 — TAP.7 TAPC hierarchy**

The TAPC hierarchy provides a means for the DTS to access the hierarchy levels as follows:

- 1<sup>st</sup> level      None, one, or all ADTAPCs
- 2<sup>nd</sup> level      None, one, or more than one CLTAPCs
- 3<sup>rd</sup> level      None, one, or more than one EMTAPCs
- Other levels    Same as EMTAPCs

The ability to access none, one, and more than one TAPC at each level of the hierarchy enables a rich set of TAP.7 features. The description of the TAPC hierarchy and its use is expanded in various clauses within this specification.

### 4.2.3 Parking the TAPC state

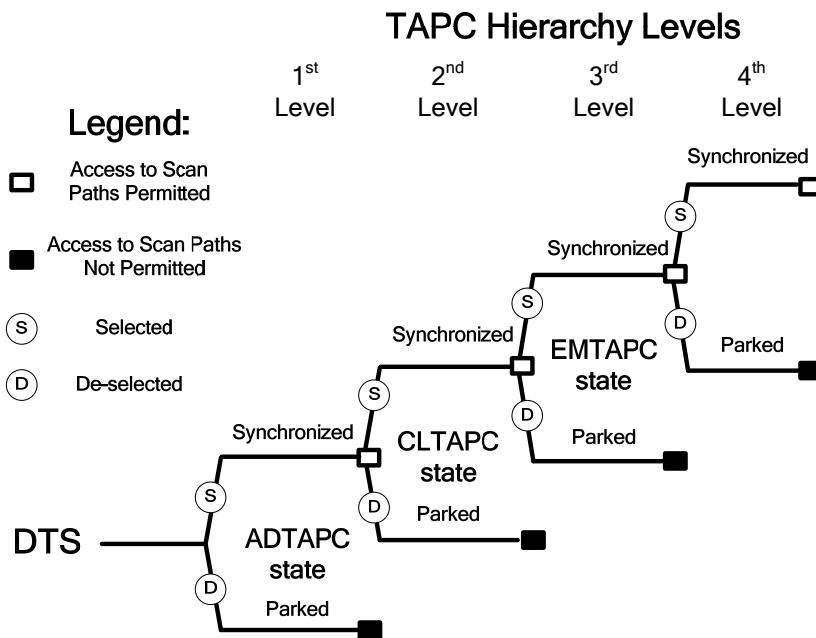
#### 4.2.3.1 Overview

This standard utilizes a concept called “parking” the TAPC state within the TAP.7 TAPC hierarchy. When the TAPC state is parked, the TAPC state of its parent may change while its state does not change. In this case, its state is no longer synchronized with the operation of TAPs at higher levels of the TAPC hierarchy. The state of a TAPC at a specific hierarchy level may be parked when the scan paths managed by the TAPC are of no interest or are not available for use (for example, powered-down). The TAP.7 architecture provides TAPC selection and deselection facilities to manage the parking of the TAPC state at each level of the hierarchy. These facilities operate differently at each level of the TAPC hierarchy and are described in Clause 8.

Parking the state of the TAPC at level n of the hierarchy prevents the TAPC state from advancing in hierarchy levels lower than the level where the TAPC state is parked. TAP.7 Scan Path continuity is maintained even though the TAPC states are parked as the scan paths managed by the parked TAPCs are bypassed to maintain scan path continuity.

#### 4.2.3.2 Parking-state relationships

The relationship of the parking of the TAPC state at each level of the hierarchy and access to scan paths the TAPC manages is shown in Figure 4-3. When the TAPC state is not parked, the operation of its TAPC is synchronized to the operation of its parent's TAPC. In this case, the TAPC state of the parent TAPC and child TAPC operate in lock-step at all times. The terms “selected” and “deselected” are used to describe the synchronized and parked operation of the TAPC at any level of the TAPC Hierarchy. The TAPC hierarchy and its use are described further in Clause 8 along with other terminology describing the operation of the levels of the TAPC hierarchy.



**Figure 4-3 — Relationships of parking the TAPC state to its associated resources**

The parking states supported for each level of the TAPC hierarchy are shown in Table 4-1. Note that the first level of the TAPC hierarchy supports the most parking states.

**Table 4-1 — TAPC hierarchy/allowed TAPC parking-state relationships**

TAPC parking state	1 <sup>st</sup> level (ADTAPC)	2 <sup>nd</sup> level (CLTAPC)	3 <sup>rd</sup> level (EMTAPC)	Other levels (EMTAPCs)
<i>Test-Logic-Reset</i>	Yes	Yes	Yes	Yes
<i>Run-Test/Idle</i>	Yes	Yes	Yes	Yes
<i>Pause IR</i>	Yes	Yes	No	No
<i>Pause-DR</i>	Yes	Yes	No	No
<i>Select DR-Scan</i>	Yes	No	No	No
Other TAPC states <sup>a</sup>	Yes	No	No	No

<sup>a</sup>Selection requires a reset of the TAP.7 Controller to set the state to *Test-Logic-Reset* when the TAPC state is parked in one of these states.

#### 4.2.3.3 Parking-state terminology

##### 4.2.3.3.1 ADTAPC selection states

The ADTAPC may be parked in any state. A TAP.7 Controller reset is required to restart its operation when the ADTAPC state is parked in a state other than *Test-Logic-Reset*, *Run-Test-Idle*, *Pause-xR*, or *Select-DR*.

The ADTAPC is described as having one of the following scan selection states:

- **Online:** The ADTAPC operates in lock-step with both the TAPCs of other Online TAP.7 Controllers and the DTS. The scan paths managed by the ADTAPC are accessible to the DTS.
- **Offline:** The ADTAPC operates with its state parked in any of the TAPC states. The operation of its Scan and Transport functions is frozen in place. The scan paths managed by the ADTAPC are not accessible to the DTS.

The ADTAPC is considered selected when the scan selection state is *Online* and deselected when the scan selection state is *Offline*.

Hereafter, the term “Online” is used to describe a technology that is selected, and the term “Offline” is used to describe a technology that is deselected. The terms “Offline TAP.7 Controller” and “TAP.7 Controller is Offline” are used to describe a TAP.7 Controller whose ADTAPC state is parked. The terms “Online TAP.7 Controller” and “TAP.7 Controller is Online” are used to describe a TAP.7 Controller whose ADTAPC state is advancing.

##### 4.2.3.3.2 CLTAPC selection states

The CLTAPC may be parked in the *Test-Logic-Reset*, *Run-Test/Idle*, *Pause-IR*, and *Pause-DR* states. These states were chosen as parking states as these states are stable states in the IEEE 1149.1 TAPC state diagram. The *Test-Logic-Reset* parking state is created by some TAP.7 Controller start-up options and when the CLTAPC state is parked and the CLTAPC state is asynchronously reset.

The CLTAPC is described as having one of the following scan selection states:

- **Coupled:** The CLTAPC runs in lock-step with the ADTAPC. The IR Scan Path managed by the CLTAPC during Instruction Register Scans is the scan path provided by the TAP.7 Controller during Instruction Register Scans, and the DR Scan Path managed by the CLTAPC is the scan path provided by the TAP.7 Controller during Data Register Scans.
- **Decoupled:** The CLTAPC state is parked in either the *Test-Logic-Reset* or *Run-Test/Idle* state. The IR Scan Path managed by the CLTAPC is *not* part of the scan path provided by the TAP.7 Controller during Instruction Register Scans, and its DR Scan Path managed by the CLTAPC is *not* part of the scan path provided by the TAP.7 Controller during Data Register Scans. The TAP.7 Controller provides a One-bit Scan Path for both IR and DR Scans.

The CLTAPC is considered selected when the scan selection state is *Coupled* and deselected when the scan selection state is *Decoupled*. The DTS initiates *Coupled* operation of the CLTAPC when it desires access to the STL Scan Paths. It DTS initiates *Decoupled* operation of the CLTAPC when it desires to bypass the STL Scan Paths. The terms “CLTAPC is *Decoupled*” and “STL is *Decoupled*” are used to describe a TAP.7 Controller whose CLTAPC state is parked. The terms “CLTAPC is *Coupled*” and “STL is *Coupled*” are used to describe a TAP.7 Controller whose CLTAPC state is advancing.

It is often desirable to connect a DTS to a running system without changing the system’s behavior. For this to be the case, the random TAP.7 signaling activity created by “hot connecting” a DTS to an operating TS cannot disturb the system’s operation in any way. The TAP.7 Controller’s start-up options support “hot connecting” a DTS to an operating TS. These options permit the creation of either the *Test-Logic-Reset* or *Run-Test/Idle* parking state at start-up. Creating the *Test-Logic-Reset* state initiates “hot-connect protection.”

With hot-connect protection, the random signaling caused by physically connecting a DTS to an operational target system is unlikely to disturb the operation of the target system since a lengthy TAPC State Machine state sequence needed to change a TAP.7 Controller register and therefore affect the operation of the STL. When hot-connect protection is expected, other non-TAP signals affected by the connection of the DTS and TS cannot disturb system operation (e.g., a signal to reset the target system, etc.).

#### 4.2.3.3 EMTAPC selection states

EMTAPCs may be parked in the *Test-Logic-Reset* or *Run-Test/Idle* states. The *Test-Logic-Reset* parking state is created when the EMTAPC state is parked and the TAPC state is asynchronously reset. The *Run-Test-Idle* EMTAPC parking state is created by selection and deselection of EMTAPCs. The participation of an EMTAPC in scan operations is defined by one of the following scan selection states:

- **Normal:** The EMTAPC runs in lock-step with the CLTAPC. The IR Scan Path managed by this TAPC is part of the scan path as viewed from the CLTAPC during the Instruction Register Scans, and its DR Scan Path is part of the scan path during Data Register Scans.
- **Excluded:** The EMTAPC runs in lock-step with the CLTAPC. The IR Scan Path managed by this TAPC is part of the scan path during the Instruction Register Scans, and the DR Scan Path managed by this TAPC is *not* part of the scan path during Data Register Scans.
- **Isolated:** The EMTAPC is parked in either the *Test-Logic-Reset* or *Run-Test/Idle* state. The IR Scan Path managed by this TAPC is *not* part of the scan path during the Instruction Register Scans and the DR Scan Path managed by this TAPC is *not* part of the scan path during Data Register Scans.

An EMTAPC is considered selected when the selection state is *Normal* and is considered deselected otherwise. The DTS initiates Normal operation of an EMTAPC when it desires access to the TAPC's IR and DR Scan Paths. With series selection approaches, the DTS initiates Excluded operation of the EMTAPC when it does not desire access to the TAPC's DR Scan Paths. With star selection approaches, the DTS initiates Isolated operation of an EMTAPC in star selection schemes when it has no desire to access the IR and DR Scan Paths managed by the TAPC. This scheme can also be deployed by the EMTAPCs to determine which of its children participate in scan operations in which the EMTAPC participates.

Henceforth, in this and subsequent clauses, the scan selection state of an EMTAPC is referred to as *Normal*, *Excluded*, or *Isolated*.

#### **4.2.3.3.4 Selection state summary**

Table 4-2 provides a summary of the terminology used to describe entity operation. The term “services” in this table means DR Scan Paths are accessible.

**Table 4-2 — Selection state terminology summary**

TAPC	TAPC Synchronized/ Provides all services	TAPC Synchronized/ Provides no services	TAPC Synchronized/ Provides no services
EMTAPC	<i>Normal</i>	<i>Excluded</i>	<i>Isolated</i>
CLTAPC	<i>Coupled</i>	N/A	<i>Decoupled</i>
ADTAPC, TAP.1 Branch, Other technology	<i>Online</i>	N/A	<i>Offline</i>

#### **4.2.4 Choice of parking states**

The *Run-Test/Idle* and *Run-Test/Idle* states were chosen as parking states for the following reasons:

- These states are stable states in the IEEE 1149.1 TAPC state diagram.
- The *Run-Test/Idle* parking state is used by board-level devices such as scan-path linkers used to include and exclude scan chains from a primary scan path.
- A TAPC whose state is parked is easily synchronized to the *Run-Test/Idle* of a TAPC whose state is not parked.

#### **4.2.5 Parking methods**

**The gating of the *sys\_tck* signal to park the CLTAPC state is strongly recommended** as it provides timing advantages and other benefits over other approaches such as using the *sys\_tms* signal. When the TAP.7 is used as a debug portal, the *sys\_tck* signal parking method has some significant advantages over other methods as it provides precise control of the number of *Run-Test/Idle* states (where this is critical). The gating of the *sys\_tck* signal is also required for some IEEE 1149.1-based IP with IEEE 1149.1-Other Behavior.

#### **4.2.6 Operation of the TAP.7 Controller**

##### **4.2.6.1 TAP.7 Controller management**

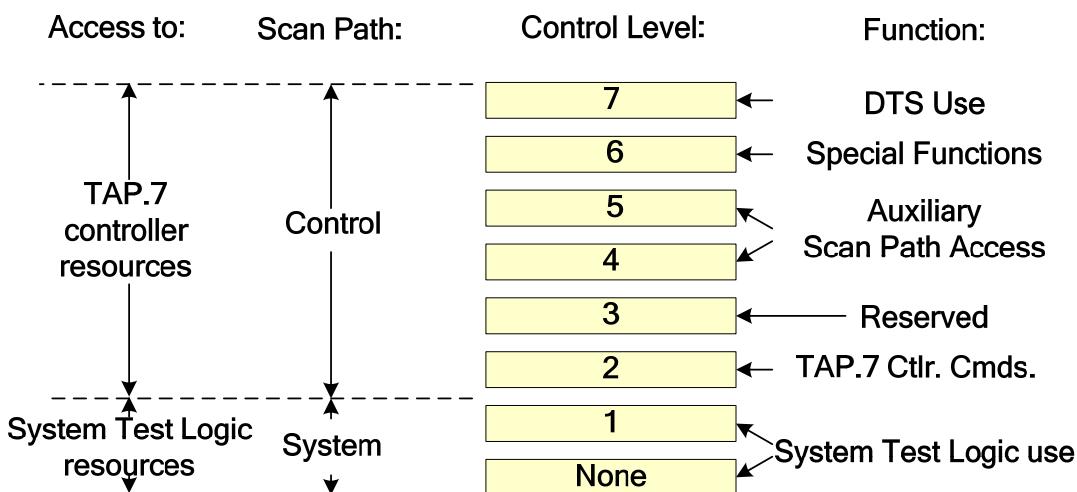
With the TAP.7 architecture, a unique TAPC state progression is used to dynamically redefine the function of DR Scans to manage TAP.7 Controller resources. This concept, described further in Clause 17, adds

TAP.7 functionality to the *BYPASS* and *IDCODE* instructions. This use of these instructions is not apparent to System Test Logic that is based on IEEE Std 1149.1.

The unique TAPC state progression and subsequent TAPC state sequences are used to:

- Create TAP.7 Controller commands
- Change the TAP.7 Controller operating modes
- Access both System Test Logic resources and TAP.7 Controller resources

The dynamic redefinition of the function of DR Scans can be viewed as paged operation of the TAPC state diagram. One page of the TAPC state diagram provides the TAPC function defined by IEEE Std 1149.1. Other pages of the TAPC state diagram provide functionality that differs from the TAPC function defined by IEEE Std 1149.1. The term “Control Level” is used to describe a page of the TAPC state diagram providing functionality that is not defined by IEEE Std 1149.1. It is the DTS’ responsibility to coordinate the use of the unique TAPC state progression with the *BYPASS* and *IDCODE* instructions to both create and use control levels. The description of a T1 TAP.7 in 5.3 includes a description of this infrastructure. A conceptual view of this concept is shown in Figure 4-4.



**Figure 4-4 — TAP.7 Controller management**

#### 4.2.6.2 System and Control Paths

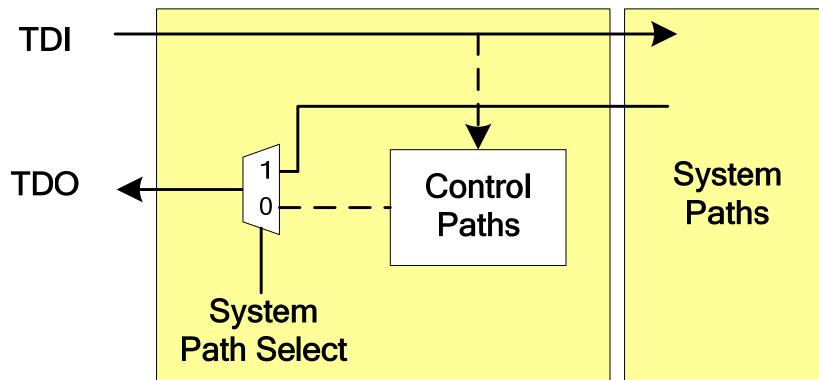
From the DTS viewpoint, there are two types of scan paths, system and control. System Scan Path access resources within the System Test Logic while Control Paths access resources associated with the TAP.7 Controller.

A high-level view of Control Paths and their relationship with the System Test Logic is shown in Figure 4-5. Control Scan Paths (Control Paths) are managed in a manner that differs from than the CLTAPC’s management of the IR and DR Scan Paths (System Paths) associated with it. Control Paths are implemented in parallel with System Paths. They are implemented in a manner that does not add bits in series with the scan chains within the System Test Logic. This makes the existence of Control Paths transparent to the software drivers managing the System Paths. The TAP.7 Controller contains no Instruction Registers and a limited number of optional DR Scan Paths accessible using Control Levels 4 and 5. Control Paths have attributes that may be different than System Paths. They may be one of the following:

- Read only
- Write only

- Read/Write
- Conventional IEEE 1149.1 Scan Paths

The TDI and TDO signals defined by IEEE Std 1149.1 are not used with some write-only Control Paths. These Control Paths are managed entirely with TAP.7 Controller commands created with TAPC state progressions. This is highlighted in Figure 4-5 with the dotted line connecting the TDI and TDO signals to the Control Paths.



**Figure 4-5 — System and Control Path relationship**

The Control and System Paths and their management are described in detail in Clause 9.

#### 4.2.6.3 Power management

With power-conscious applications, it is desirable to power-down the TAP.7 Controller and as much System Test Logic as practical unless a DTS connection exists and the DTS intends to use the TAP. The TAP.7 architecture accommodates this need with optional TAP.7 power-management features. The chip architect determines whether all, part, or none of the TAP.7 Controller power management function is implemented. The power-management features utilize the normal TAP signaling and do not increase the TAP.7 signal count. The signaling controlling power-management features is compatible with TAP.1s and TAP.7s whose controllers do not implement power management. The DTS may use these features to perform the following functions:

- Power-up a TAP.7 Controller after the DTS' initial connection to a target system
- Power-down a TAP.7 Controller after its use
- Power-up a TAP.7 Controller after the DTS has permitted its power-down.

A Chip-Level Power Manager detects a fixed set of conditions to initiate power-up. Once powered-up, the TAP.7 Controller power-management facilities perform the following two-step operation:

- Power-up confirmation
- Power-down initiation

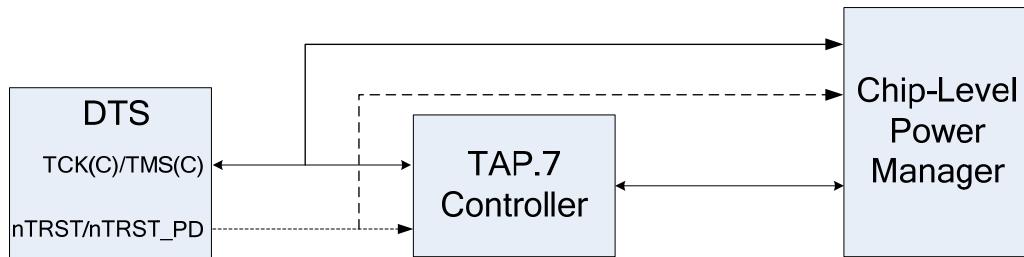
After power-up, the power-management facilities first confirm power-up was neither accidental nor erroneous. If confirmation fails, the TAP.7 Controller is powered-down. Once an appropriate power-up is confirmed, the TAP.7 Controller monitors the TAP.7 state for the conditions specified by the PWRMODE Register. Four Power-Control Modes (Modes 0–3) define the conditions initiating power-down and specifying power-down criteria compatible with the following:

- A DTS- or TS-sourced TCK(C)

- The DTS stopping TCK(C)
- No power-down

Power-down is initiated when these conditions are met.

A typical system utilizing TAP.7 power management is shown in Figure 4-6. The DTS creates both the conditions permitting power-down and initiating power-up. The TAP.7 Controller detects the conditions initiating power-down while the Chip-Level Power Manager detects the conditions initiating power-up. Power management is described in detail in 18.10.



**Figure 4-6 — Components involved in TAP.7 Controller power management**

## 4.2.7 Scan topologies

### 4.2.7.1 Series and Star Scan Topologies

Although the TAP.1 architecture supports the deployment of TAPs in the Series Scan Topology with a minimum of four signals, the TAP.7 architecture supports the deployment of T0–T5 TAP.7s with a minimum of four signals in the Series Scan topology, the deployment of T3–T5 TAP.7s and Star-4 Scan Topologies, and the deployment of T4–T5 TAP.7s with either two or four signals in the Star-2 Scan Topologies.

The differences in the operation of a TAP.7 Controller in both Series and Star Scan Topologies can be better understood when the operation of common functions in Series and Star Scan Topologies are contrasted as shown in Table 4-3.

**Table 4-3 — Contrast of series and star operation**

Function	Series operation	Star operation
No operation	Bypass Instruction CLTAPC deselection (T2TAP.7)	CLTAPC deselection
Scans	Data passes through all TAP.7 Controllers	Data passes through one TAP.7 Controller at a time. A method to create the equivalent of a series scan is provided
Addressability	By position on the scan chain	Directly addressable with either 1) A TAP.7 Controller Address, or 2) A TAP.7 Controller Identification Code
TAPC Parking States	<i>Test-Logic-Reset, Run-Test/Idle</i>	<i>Test-Logic-Reset, Run-Test/Idle, Pause-IR, Pause-DR</i>
TDO Drive	IEEE 1149.1 style	Requires drive conflict prevention

#### 4.2.7.2 Scan Topology Training

The TAP.7 architecture supports a DTS command sequence that stimulates a T3 and above TAP.7 Controller in a manner that causes it to recognize the Scan Topology in which it is deployed. This command sequence is called the Scan Topology Training Sequence and is described in detail in 20.11.

The Scan Topology Training Sequence utilizes the fact that the connectivity attributes of Series, Star-2, and Star-4 Scan Topologies are unique. It makes these unique attributes visible to the TAP.7 Controller, with the controller observing these attributes and recording the scan topology in which it is deployed. The scan topology is conveyed to the TAP.7 Controller with TDIC and TDOC signal values driven by the DTS and recorded by the TAP.7 Controller at multiple points in the sequence, with no drive of these signals by TAP.7 Controller during this sequence.

The Scan Topology Training Sequence is commonly used by the DTS at start-up as part of the DTS discovering the technology branches to which it is connected and the attributes of the devices of each branch. Since the Device Identification Code is mandatory with the TAP.7 Controller, the discovery process may be extended to the second and third levels of the TAP.7 Controller hierarchy when this capability is supported (discovery at levels two and more of the TAPC hierarchy is not covered by this standard).

Upon completion of the scan topology discovery process, the DTS knows the types and numbers of TAP.7 Branches, the number of TAP.7 Controllers connected to these branches, the class of each TAP.7 Controller, and information describing the optional features supported by the TAP.7 Controllers (provided this optional information is available). It may subsequently select a TAP.7 Technology Branch (a scan topology) and communicate with one or more than one TAP.7 Controllers associated with the branch.

#### 4.2.7.3 Sharing of signaling with other technologies

The TAP.7 architecture provides for the sharing of the TCK(C) and TMS(C) signals with other technologies or across different TAP.7 Technology Branches as described in 1.11.2. In this case, the DTS (the parent) determines whether the state of all ADTAPCs (its children) is parked. When the ADTAPC state of all TAP.7 Controllers sharing the DTS connection is parked, the TCK(C) and TMS(C) signals may be used to access other technologies. The DTS may extend the start-up discovery process mentioned in 4.2.7.2 to determine whether the scan topology includes technologies branches other than TAP.7 Technology Branches. The Selection and Deselection Escapes described in Clause 10 manage the selection and deselection of ADTAPCs and other technologies. The selection and deselection of ADTAPCs is described in detail in Clause 11.

#### 4.2.7.4 Direct addressability for Star Scan Topologies

Direct addressability is needed for operation in Star-4 and Star-2 Scan Topologies. This addressability is built into the TAP.7 Controller. TAP.7 Controller Addresses are derived from the Device Identification Code value in combination with a Node Identification Code (NODE\_ID). The Device Identification Register is mandatory with a TAP.7 while the Node Identification Code is added with T3 and above TAP.7s. The combination of the Device Identification Code and Node Identification Code provide for the use of one or more of the same or different devices in a Star Scan Topology. The TAP.7 Controller Address (TCA) is formed using 27 bits of the Device Identification Code in combination with an eight-bit Node Identification Code as described in 20.8. The TAP Controller Address is aliased to a four-bit Controller Identification Code called a Controller Identification Code (CID). The Controller ID provides a compact way of referencing an ADTAPC with commands. Commands are provided to discover the TAP Controller Addresses and allocate a Controller ID to an ADTAPC. TAP.7 Controller direct addressability is described in detail in 20.8.

#### 4.2.7.5 Series-Equivalent Scans for Star Scan Topologies

The TAP.7 architecture provides for the creation of Series-Equivalent Scans in either a Star-4 or Star-2 Scan Topology (described in more detail in 5.5.3.4). Series-Equivalent Scans provide the functionality of DR and IR Scans in a Series Scan Topology. Series-Equivalent Scans are created by selecting the CLTAPCs of interest while in the *Run-Test/Idle* state, moving the state of the selected CLTAPCs to *Pause-xR*, sequentially scanning the scan paths of interest within each scan topology beginning and ending the scans in the *Pause-xR* state. Once these scans are complete, the update/capture operation is performed simultaneously in these CLTAPCs. The TAPC state may be moved from the *Update-xR* state to either the *Run-Test/Idle* state (where the selection of CLTAPCs may be changed) or *Select-DR* state on to the *Pause-xR* state where the sequential scans can again be performed. This process preserves the timing between the *Update-xR* and *Capture-DR* states when the portion of the operation that does not include the *Shift-xR* state is performed using the Standard Protocol. It uses special TCK(C) and TMS(C) signaling sequences called Escapes (see 4.3.1).

The steps comprising a typical Series-Equivalent Scan are shown as follows:

- (1) Select all branches
- (2) Select all CLTAPCs in all branches; specify the use of the Standard Protocol
- (3) Move to the *Pause-xR* state without encountering the *Shift-xR* state
- (4) With TAPC state of *Pause-xR*, select a single branch; specify the use of protocol compatible with the selected branch
- (5) With TAPC state of *Pause-xR*, select one CLTAPC within the branch
- (6) With TAPC state of *Pause-xR*, perform the scan without encountering an *Update-xR* state
- (7) Repeat steps 5 and 6 until all System Paths of interest within this branch are scanned
- (8) Repeat steps 5, 6, and 7 until all System Paths of interest are scanned for all branches
- (9) With TAPC state of *Pause-xR*, select all branches
- (10)- With TAPC state of *Pause-xR*, select all CLTAPCs, and specify the use of the Standard Protocol
- (11) Move through the *Update-xR* and the *Capture-xR* states without encountering a *Shift-xR* state, ending in the *Pause-xR* state
- (12) Go to step 4

With the sequence above, all or some CLTAPCs in all branches progress from either the *Test-Logic-Reset* or *Pause-xR* state to the *Pause-xR* state without encountering a *Shift-xR* state. A Selection Escape is used to select all ADTAPCs within a branch and deselect all ADTAPCs within other branches (the Selection Escape is used to select one and only one branch and specify the protocol used to communicate with the branch). Selection of the ADTAPCs occurs while in the *Pause-xR* or *Run-Test-Idle* state.

Once a single branch is selected, a function called a “Scan Selection Directive” (SSD) is used to select one and only one CLTAPC to provide the System Path of interest. At this point, only one CLTAPC is selected. After the CLTAPC’s selection, the CLTAPC state is moved from the *Pause-xR* state to the *Shift-xR* state to the *Pause-xR* state without traversing the *Update-xR* state while the states of other CLTAPCs remain parked in the *Pause-xR* state. Once the scan of the scan path of interest is completed, a different CLTAPC is selected, with the scan of the scan path of interest subsequently completed. When the scan of all scan paths of interest is completed, the next sequential branch is selected, and the process outlined herein is repeated.

When the scan of the scan paths of interest in all branches is completed, all branches are selected. At this point, the TAPC state of all TAPCs is *Pause-xR* and the TAPC state is moved from *Pause-xR* to *Pause-xR*, traversing the *Update-xR* and *Capture-xR* states.

With Series-Equivalent Scans, Selection and Deselection Escapes (described in 4.3.1 and 10.4) are used to select and deselect ADTAPCs within a branch. SSDs (described in 5.5.3.5 and 20.10) are used to select and deselect CLTAPCs within the selected branch.

#### 4.2.7.6 Output-drive characteristics

When operating in a Star Scan Topology, the TAP.7 Controller manages the drive of the TDOC and TMSC signals to prevent drive conflicts. The following four types of TDO(C) and TMS(C) drive are supported:

- Single      The signal is driven with a value of the TAPC's choosing.
- Joint        The signal is driven with a predetermined logic 1 value.
- Voting       The signal is driven with a logic 0 value only when the output of a logic 0 data value is desired.
- Inhibited    The signal is not driven at a time when it may otherwise be driven.

Single drive is the legacy drive. The TAP.7 Controller is free to drive the signal with a data value of its choosing as it presumes it is the only TAP.7 Controller that will drive the signal. Joint drive is used with the TMSC signal when a TAP.7 Controller determines there are either no drive candidates or more than one drive candidate. It is combined with voting drive with the TDOC and TMSC signals to create precharge/discharge operation of these signals as part of CID allocation. Voting drive is literally open drain operation of the drive and is used only in combination with Joint drive as described previously. With inhibited drive, certain conditions cause the signal to remain high impedance when it could otherwise be driven. An example of this is when it has determined it is not a drive candidate (none of the other drive types have been invoked).

The drive policy is determined by the number of drive candidates, and in the case of CID allocation, the execution of the command and the participation of the TAP.7 Controller in the CID allocation process.

The TDOC and TMSC Drive Policies are formulated by describing the STL and Extended Protocol Unit (EPU) as being a member of a group of STLs and EPUs with similar operating characteristics. This concept identifies the number of STLs and EPUs whose scan paths are accessible and whether this TAP.7 Controller is the only drive candidate. With T3 and above TAP.7s, the TAP.7 Controller monitors the actions selecting and deselecting CLTAPCs to determine the number of candidates for driving the TDOC and TMSC signals. Further description of this concept is deferred to the description of T2 and T3 TAP.7s in 5.4 and 5.5, respectively.

When operation with a Star-4 or Star-2 Scan Topology is specified, CID allocation is not taking place, and certain other inhibiting conditions are inactive, the drive policy for the TDOC and TMSC signals permits TAP.7 Controller drive of these signals with a data value determined by the TAP.7 Controller provided:

- The ADTAPC is a drive candidate.
- Only one ADTAPC is a drive candidate.

This ensures only one TAP.7 Controller drives a TAP signal at a time with a data value determined solely by the TAP.7 Controller. Certain other conditions related to the generation of and operation within control levels inhibits the drive of the TDOC signal, even when operation within a Series Scan Topology is specified. This supports the Scan Topology Discovery Sequence. The TDOC Drive Policy is described in Clause 13. The TMSC Drive Policy is described in Clause 14.

When operating in a Star Scan Topology, the drive of the TDOC and TMSC signals is handled in a manner that supports the Wire-AND of TMSC and TDOC output during the allocation of CIDs. In these cases, the TMSC and TDOC signals may simultaneously be driven with the same logic level (all TAP.7 Controllers

drive a logic 1 or all TAP.7 Controllers drive a logic 0). The TMSC signal is also operated in a Wire-ANDED mode in some cases where more than one CLTAPC is selected.

#### 4.2.7.7 Scan formats

With T2 and above TAP.7s, the DTS establishes certain operating characteristics of the TAP.7 Controller using a TAP.7 Controller register called the Scan Format Register. This register specifies the use of signaling protocols compatible with operation in a Series, Star-4, or Star-2 Scan Topology. These protocols define the signaling attributes and information content of scan exchanges. Among the signaling attributes are the TDOC drive characteristics required to operate T3 and above TAP.7s in a Star-4 Scan Topology and the TMSC drive characteristics required to operate T4 and above TAP.7s in a Star-2 Scan Topology.

Four scan formats [JTAG Scan (JScan0–JScan3)] specify operation with a Series Scan Topology with T0–T2 TAP.7s. With T3 and above TAP.7s, the JScan3 Scan Formats specifies operation compatible with a Star-4 Scan Topology. The difference in the operation of the JScan3 Scan Format with T0–T2, T3, T4(W), and T5(W) TAP.7s allows the detection of an erroneous deployment of T0–T2 TAP.7s in a Star-4 Scan Topology.

The JScan0–JScan3 Scan Formats also specify the use of the Standard Protocol. These scan formats provide the following behaviors:

- JScan0 Provides IEEE 1149.1-Specified Behavior
- JScan1 Provides IEEE 1149.1-Non-disruptive Behavior: Bypass of the System Test Logic, providing a one-bit IR and DR Scan Path
- JScan2 Provides IEEE 1149.1-Non-disruptive Behavior: Selectable Bypass of the System Test Logic, providing a one-bit IR and DR Scan Path when the System Test Logic is bypassed
- JScan3
  - T0–T2 TAP.7s Provides the same behavior as JScan2
  - T3–T5 TAP.7s Provides 1149.1-Other Behavior—Bypass of the System Test Logic, Drive of the TDOC signal in a manner that prevents drive conflicts

Either the JScan0 or the JScan1 Scan Format may be specified as the default scan format at start-up. With a JScan0 default value, the TAP exhibits IEEE 1149.1-Specified Behavior at start-up. With a JScan1 default value, the TAP exhibits IEEE 1149.1-Non-disruptive Behavior at start-up. The JScan1 default provides “hot-connect protection” as it is highly unlikely that connecting the DTS to an already running system will affect system operation. Hot-connect protection and the effects of bypassing the System Test Logic are described further in Clause 19.

With T3 and above TAP.7s, the JScan3 Scan Format specifies operation with a Star-4 Scan Topology and the use of the Standard Protocol. This scan format enables the use of TDOC Drive Policies that prevent drive conflicts.

With T4 and above TAP.7s, 13 Advanced Scan Formats (three mandatory and ten optional) specifying the use of the Advanced Protocol may be added to the four JScan Scan Formats. These scan formats match a number of use cases and provide tradeoffs between capability and performance. The DTS software selects a scan format that is compatible with the test or debug application. These additional scan formats are listed as follows:

- MScan Primarily used to access the System Test Logic simultaneously in multiple TAPs where the System Test Logic in any of the TAP.7 Controllers may stall the TAPC state progression. This scan format is also used to allocate a CID alias to a TAP.7 Controller Address.

- OScan0–OScan7      Used to access the System Test Logic of one TAP.7 where the TAP.7 Controller may or may not stall the TAPC state progression of more than one TAP.7 where none of the TAP.7 Controllers can stall the TAPC state progression. OScan Scan Formats provide various optimizations that increase scan performance. Optimizations determine the information exchanged by the DTS and TS during the *Shift-xR* and remaining TAPC states.
- SScan0–SScan3      Used to access the System Test Logic of a single TAP.7. SScan Scan Formats provide optimizations that increase scan performance by (1) dividing the consecutive *Shift-xR* states associated with a scan into segments and (2) transferring only the information of interest for each segment (for example, all TDI data/no TDO data, both TDI and TDO data, or no TDI data/all TDO data).

The MScan and OScan0–OScan1 Scan Formats are mandatory, whereas the OScan2–OScan7 and SScan0–SScan3 Scan Formats are optional. The optional scan formats provide scan efficiency improvements tailored to various test and applications debug use cases. These scan efficiency improvements may provide as much as two or more times the scan performance of the MScan Scan Format.

#### 4.2.7.8 Interoperability

The TAP.7 Controller operates in a manner that provides for the interoperability of controllers supporting different sets of scan formats and data transport capability. This is described in detail in 21.2.3.

### 4.3 Concepts supporting pin efficiency

Pin efficiency is defined as maximizing the number of functions performed while minimizing the number of pins performing these functions. The concepts within the TAP.7 architecture supporting pin efficiency include but are not limited to the following:

- Signaling methods
- Protocols
- Serialization of scan transfers
- Interleaving of scan and non-scan transfers

#### 4.3.1 Signaling methods

The DTS and TS exchange information in one of two ways as follows:

- A TCK(C) signal edge sampling the TMS(C) signal value
- A count of TMS(C) signal edges while the TCK(C) signal is a logic 1

Both of these means may be used concurrently within the same TCK(C) bit period. The following signaling characteristics make this possible.

With both the Standard and the Advanced Protocols, the TMS(C) signal value changes no more than once per TCK(C) period, provided information is conveyed only with the TCK(C) signal sampling the TMS(C) signal value. The TAP.7 architecture uses this fact to convey the second form of information during a TCK(C) bit period utilizing a special TCK(C)/TMS(C) signaling sequence with an Escape. With an Escape, the DTS toggles the TMS(C) signal an even number of times while the TCK(C) signal is a logic 1 to convey both forms of information. The DTS creates an even number of TMS(C) edges to restore the original value of TMS(C) established at the beginning of the bit period by the falling edge of TCK(C).

There are four types of Escapes. The DTS generates an Escape by edges creating an even number of TMS(C) edges while TCK(C) is a logic 1. The number of TMS(C) edges for each Escape is shown in parenthesis below. This edge count is in addition to a TMS(C) edge that may occur as a result of the TMS(C) edge establishing the TMS(C) value for the bit period, as follows:

- No Escape (0)
- Custom (2)      End of transfer
- Deselection (4)      Deselect the TAP.7 Controller
- Selection (6)      Begin a Selection Sequence capable of selecting or deselecting a TAP.7 Controller
- Reset (8 or more)      Reset the TAP.7 Controller upon the falling edge of the TCK(C) signal, inhibit TMSC drive after the detection of eight edges

The TAP.7 Controller accommodates the possibility that the TMS(C) edge establishing the TMS(C) value for the bit period may occur after the TCKC signal is a logic 1. This can occur as a result of analog delays in the system (see 10.4 for an explanation of this phenomenon). These means the TAP.7 Controller interprets an odd number of edges occurring while TCK(C) is a logic one as the next lowest even number. This makes the TAP.7 Controller's Escape detection logic insensitive to the timing relationship of the TCK(C) falling edge and a change in the TMS(C) value generated by the TCK(C) falling edge.

The relationship between the TAP.7 Class and Escapes is shown in Table 4-4.

**Table 4-4 — TAP.7 Class/Escape relationships**

<b>TAP.7 Class</b>	<b>Escapes</b>
T0–T2	With RSU option
T3	Mandatory
T4 and above	Mandatory

Escapes are ignored by TAP.1s and TAP.7 Controllers that do not support their detection. This makes the use of Escapes transparent to the TAPCs within the STL and elsewhere in a system where their detection is not supported.

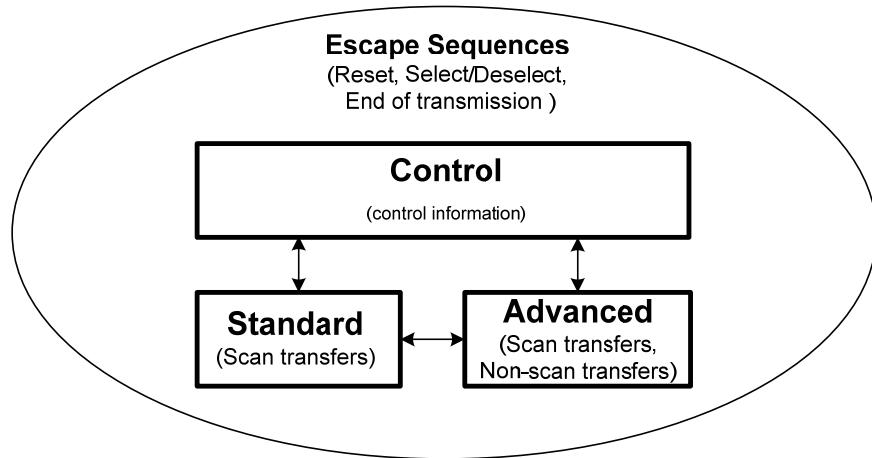
In cases where the TS sources the TCK(C) signal, the DTS cannot stop the TCK(C) signal at a logic 1 to create the proper TCK(C) and TMS(C) signaling needed to create Escapes. Escapes cannot be used in these cases.

Other optional mechanisms for the selection and deselection of ADTAPCs are supported. The use of these mechanisms is restricted, however. A specific 132-bit signal pattern called a "Selection Alert" may be optionally utilized to perform the same function as a Selection Escape, provided there are no technology branches selected (TAP.7 or other technologies). Concurrent use of the Selection Alert and the Selection Escape is permitted, provided the above guideline is followed. A combination of the *Run-Test/Idle* state and Control Level Six called a "Deselection Alert" may be optionally utilized to perform the same function as a Deselection Escape.

## 4.3.2 Protocols

### 4.3.2.1 Protocol types

The TAP.7 architecture utilizes the three protocols shown in Figure 4-7.



**Figure 4-7 — TAP.7 Protocols and Escapes**

The protocols shown in this figure transfer information by sampling the TAP signal values with a TCK(C) edge. Escapes may be used with each of these protocols. The Standard Protocol may be used to switch to the use of the Advanced Protocol and vice versa. The Control Protocol also establishes whether the Standard or the Advanced Protocol follows a Selection Escape or a Selection Alert. The mandatory and optional deployment of these protocols with the TAP.7 Classes is shown in Table 4-5.

**Table 4-5 — TAP.7 Class/Protocol relationships**

TAP.7 Class	Protocol		
	Standard	Control	Advanced
T0–T2	Mandatory	Optional	—
T3	Mandatory	Mandatory	—
T4 and above	Mandatory	Mandatory	Mandatory

#### 4.3.2.2 Standard Protocol

The Standard Protocol signaling is defined by IEEE Std 1149.1. The Standard Protocol provides the functionality specified by IEEE Std 1149.1 when the TDI and TDO signal functions are available (T0–T3 TAP.7s and T4(W) and T5(W) TAP.7s when the TDIC and TDOC signals provide these functions).

Recall that TAP.7 commands are implemented using only the TCK(C) and TMS(C) signals. This means the Standard Protocol can be used to manage the TAP.7 Controller functionality independently of the availability of the TDI and TDO signals (even with T4(N) and T5(N) TAP.7s). Data transfers with this protocol require the use of TDI(C) and TDO(C) signals.

#### 4.3.2.3 Advanced Protocol

The Advanced Protocol transfers information using only the TCK(C) and TMS(C) signals with T4 and above TAP.7 configurations. Two types of packetized information are used to transfer information, as follows:

- Scan Packet (SP)      Serialized TDI, TMS, TDO, and control information for one TAPC state, causes the advance of the TAPC state

- Transport Packet (TP)      Serialized non-scan information

Combinations of these packets are associated with a TAP controller state as follows:

- SP                          Exchange of scan information
- SP, TP                    Exchange of scan information followed by exchange of non-scan information

Each SP in one of the above sequences advances the TAPC state. There is one case where an SP is considered part of a Control Protocol sequence where it does not advance the TAPC state.

Certain scan formats provide for STL generated stalls of the TAPC state progression. These scan formats support the use of a component within the STL that performs the following task:

- Uses a Return Test Clock (the unorthodox use of Test Clock described in 1.9.2.3)
- Stalls a scan transaction until previously delivered scan data is dispositioned
- Stalls a scan transaction until scan data that is to be delivered becomes available

The DTS can also stall the TAPC state progression with any scan format when this feature is enabled.

#### **4.3.2.3.1 Interleaving of scan and non-scan information**

The Control Protocol transfers the following three types of TAP.7 Controller control information with T0 and above TAP.7s:

- Selection                    Identifies attributes required for ADTAPC selection.
- Global Register Load     Loads the state of the Global Registers needed operation with the Advanced or Standard Protocol.
- CP                         Serialized control information. It determines whether the specified operation is supported (the ADTAPC is placed Offline when not supported).
- SP                         Serialized TDI, TMS, TDO, and control information, no TAPC state advance. It determines whether the TAPC state advance created by a prior SP has completed and precedes a Check Packet.

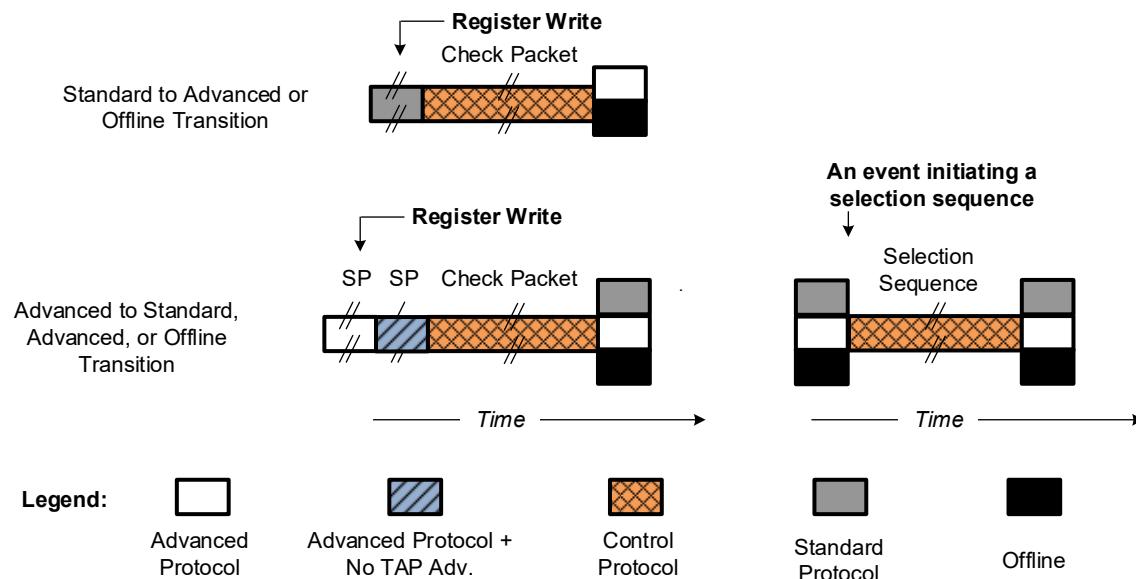
Combinations of this information are used to manage the ADTAPC's operation as follows:

- Selection Sequence      Places the ADTAPC either Online or Offline (Selection, Check Packet, and a Global Register load if specified).
- Std.-to-Adv. transition   Checks whether the advanced operation specified is supported (Check Packet).
- Configuration check     Checks whether the advanced operation specified following a TAP.7 Controller command is supported. The check is performed once the completion of the TAPC state advance mandated by the prior Scan Packet has been verified.

The use of the CP and SP/CP combinations is shown as follows and in Figure 4-10:

- CPs
  - Upon transition from the use of the Standard Protocol to the use of the Advanced Protocol

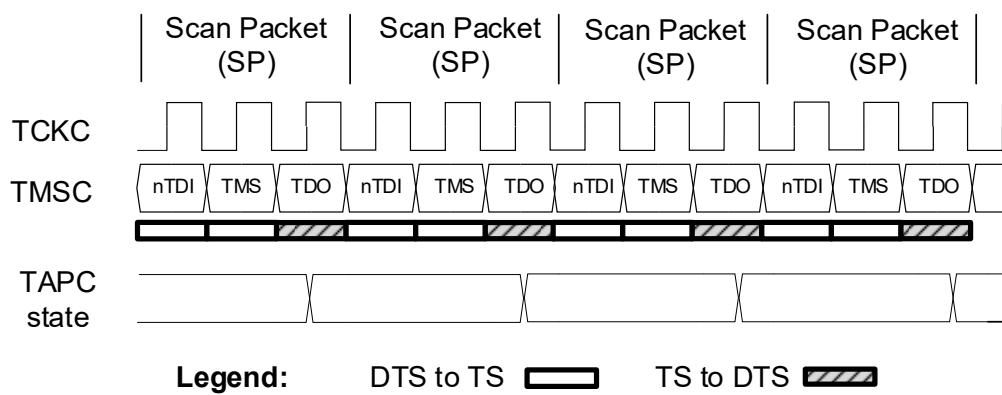
- Completing a Selection Sequence
- An SP/CP combination following the write of a TAP.7 register while using the Advanced Protocol



**Figure 4-8 — Control/Standard/Advanced Protocol relationships**

#### 4.3.2.3.2 Serialization of scan information

With the Advanced Protocol, the TMS, TDI, and TDO information used with the Standard Protocol is exchanged in a serial manner using only the TCK(C) and TMS(C) signals. The Advanced Scan Formats specify the Scan Packet content (the information exchanged). Each scan format specifies Scan Packet content that is optimized for a specific use case. A typical information exchange with a series of Scan Packets is shown in Figure 4-9. It can be seen from this figure that the Advanced Protocol is not compatible with the Standard Protocol.



**Figure 4-9 — Expanded view of an SP sequence with the OScan1 Scan Format**

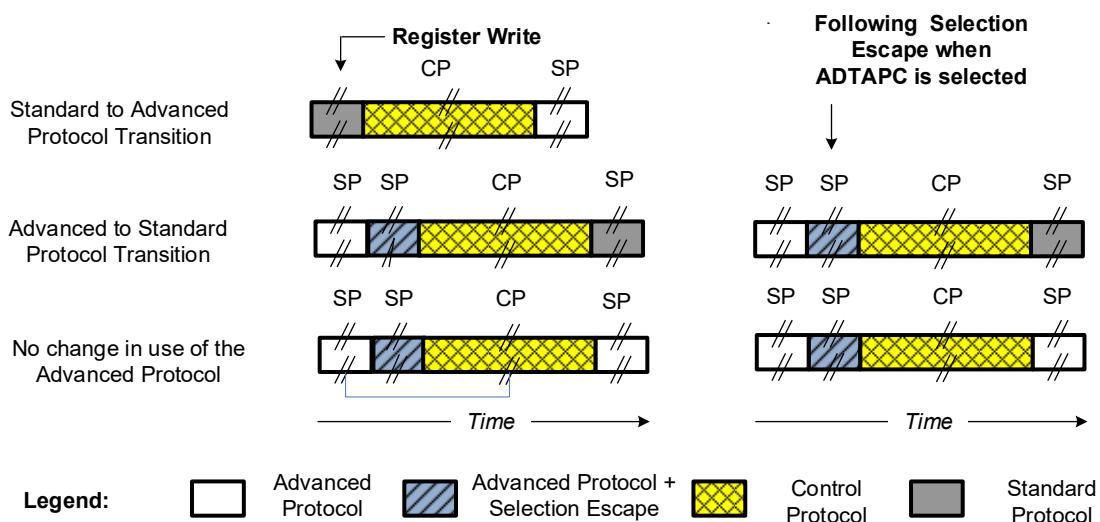
#### 4.3.2.4 Control Protocol

The Control Protocol transfers information using only the TCK(C) and TMS(C) signals with T0 and above TAP.7s. Two types of packetized information are used to transfer control information, as follows:

- SP                         Serialized TDI, TMS, TDO, and control information, no TAPC state advance
- CP                         Serialized control information

Combinations of these packets form the control packet sequence shown as follows and in Figure 4-10:

- CPs
  - Upon transition from the use of the Standard Protocol to the use of the Advanced Protocol
  - Completing a Selection Sequence
- An SP/CP combination following the write of a TAP.7 register while using the Advanced Protocol

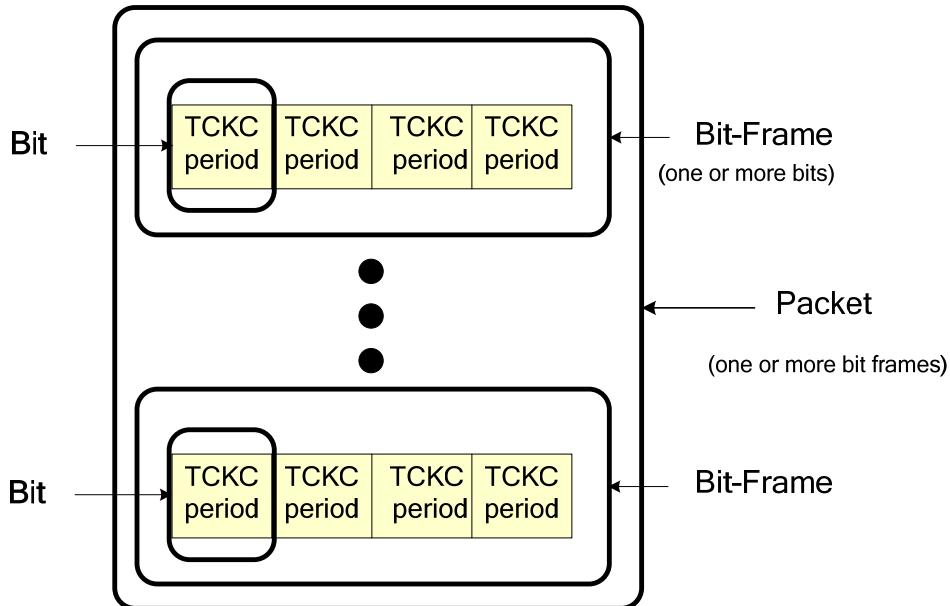


**Figure 4-10 — Control Protocol/Standard and Advanced Protocol relationships**

#### 4.3.3 Advanced and Control Protocol characteristics

##### 4.3.3.1 Packets, bit-frames, and bits

The Control and Advanced Protocols transfer packets of information related to scan (SPs), transport (TPs), and configuration changes (CPs). A packet contains one or more bits of information (for example, TMS data in the case of scan). A bit of information (TMS, TDI, TDO, or control information) is passed between the DTS to the TAP.7 Controller or between the TAP.7 Controller and DTS in a single TCKC signal period. A bit-frame is constructed from one or more bits of these bits of information. A packet is constructed from one or more bit-frames. This is shown in Figure 4-11.



**Figure 4-11 — Packets, bit-frames, and bits**

If a packet contains one bit-frame and the bit-frame contains one bit, then the packet, the bit-frame, and the bit span the same period of time.

Henceforth, in this and subsequent clauses, a reference to:

- A “bit” shall mean the information conveyed during a single TCKC signal period.
- A “bit-frame” shall mean a group of bits.
- A “packet” shall mean one or more bit-frames.

#### 4.3.3.2 Data and control information

The description of the Advanced Protocol uses the terms control information (control) and data information (data). Both the DTS and TAP.7 Controller clients are the source and destination of data information. Both the DTS and TAP.7 Controller may generate control information. The DTS is the source of most control the control information, with the TAP.7 Controller sourcing only control information that may stall Scan Packet progression that is included in certain scan formats.

Two terms, headers and directives, are used to describe the control information provided by the DTS. A header defines the format of subsequent control and data information. A directive defines an action and is similar to a command embedded in the Advanced Protocol bit stream. Henceforth, in this and subsequent clauses, a reference to the following is found:

- A “header” shall mean a sequence of one or more bits that defines the format of subsequent control and data information.
- A “directive” shall mean a sequence of one or more bits that causes an action that affects the TAP.7 Controller.
- A “stall” shall mean a sequence of one or more bits that delays the completion of a Scan Packet. Both the DTS and TAP.7 Controller (depending on the scan format) may delay the completion of the SP.

Directives embedded within a Scan Packet have one of the following functions:

- |                |  |
|----------------|--|
| — No operation | The DTS indicates it is stalling the completion of the SP.           |
| — End          | The DTS indicates it is no longer stalling the completion of the SP. |
| — Reset        | The DTS indicates it is initiating a TAP.7 Controller reset.         |

Directives embedded within a Check Packet have one of the following functions:

- |                |  |
|----------------|--|
| — No operation | The DTS indicates it is stalling the completion of the CP.           |
| — End          | The DTS indicates it is no longer stalling the completion of the CP. |
| — Reset        | The DTS indicates it is initiating a TAP.7 Controller reset.         |

Directives embedded within a Transport Packet have one of the following functions:

- |                            |   |
|----------------------------|---|
| — No operation             | The DTS indicates another directive will follow the no-operation directive. |
| — End                      | The DTS indicates it is initiating completion of the TP.                    |
| — Reset                    | The DTS indicates it is initiating a TAP.7 Controller reset.                |
| — Initiate data transfer   | Initiates a data transfer with Chip-Level Logic.                            |
| — Read and write registers | Programs Transport Control Registers.                                       |
| — Transport control        | Perform miscellaneous Transport Control Functions.                          |

The construction of packets related to scan (Scan Packets) provides for embedding none, either, or both DTS and TS stall opportunities in these packets depending on the format of the chosen Scan Packet. TS stalls provide one or more TAP.7 Controllers time to delay the completion of the packet if they are unable to either consume data sent to the TS or supply the data required from the TS. DTS stalls provide a means for the DTS to delay the beginning of a new packet until the DTS processes the data provided by the TS and new DTS data can be generated for a subsequent packet. Packets related to transport do include stall opportunities.

#### **4.3.4 Performance**

##### **4.3.4.1 TAP operation**

###### **4.3.4.1.1 Signal timing**

The TAP.7 architecture provides software selectable signal timing with the use of the Advanced Protocol, as follows:

- Half-cycle [TCK(C) falling edge to TCK(C) rising edge]
- Full-cycle [TCK(C) falling edge to TCK(C) falling edge]

The timing used with the Advanced Protocol is selectable using the Standard Protocol and, when programmed in this manner, takes effect before the Advanced Protocol is used. Full-cycle timing provides more than two times the performance. Half-cycle timing may be used to avoid marginal DTS/TS timing that may create hold time problems, as the timing mimics that of the Standard Protocol.

When falling-edge sampling is used, a full TCK(C) period is provided to propagate the TMSC signal between the DTS and TS. With half-cycle timing, generally half the TCK(C) period is provided to propagate the TMSC signal between the DTS and TS with a TCK(C) with a 50% duty cycle. Assuming

buffer, gate, and clock tree delays are comparable with these two forms of timing, there is substantially more time available to propagate signals between the DTS/TS with full-cycle timing.

#### 4.3.4.1.2 Registering of signals

With a typical TAP.7 Controller implementation, the TAP.7 architecture permits very shallow logic trees (~four gates) in input paths, and registered outputs. This minimizes the percentage of the TCK(C) period consumed with on-chip buffering and logic.

#### 4.3.4.2 Scan transfer efficiency

The architect may choose to implement a number of optional OScan and SScan Scan Formats. These scan formats are optimized to provide the best scan performance considering constraints such as follows:

- TCK(C) source (DTS or TS)
- TS pacing the TAPC state advance (for example, the STL contains IP that uses a Return Test Clock)
- Simultaneous operation of TAPCs in different chips at the same time
- Operation of TAPCs in a single chip at one time
- Debug scan performance

Scan performance is optimized by exchanging the minimum amount of information between the DTS and TS sufficient to satisfy the needs of the application. This is important to maximize performance of the link; as in the simple case shown in Figure 4-9, three times the amount of information moves across the TMSC pin with the Advanced Protocol when compared to the Standard Protocol. It can easily be seen that reducing the information volume increases the scan performance. When scan performance related to application debug is important, the implementation of some or all the optional scan formats may be desired.

The use of a combination of full-cycle timing and some highly optimized scan formats can provide scan performance better than that achievable with the Standard Protocol and its half-cycle timing. This, together with registering the TMSC signal at the TAPC input and output, has the potential of allowing TCK(C) rates that are in excess of twice those achievable with the TCK and TMS signal relationships defined by IEEE Std 1149.1 (sampling the TMSC signal with the rising edge of TCK).

With the combination of doubling of the TCKC frequency and use of a scan format with certain optimizations, the TAP.7 architecture may deliver scan performance equal to or in excess of that achievable with IEEE Std 1149.1.

### 4.4 Concepts supporting capability

#### 4.4.1 Concepts already described

Many of the concepts supporting the capabilities provided by the TAP.7 architecture also support system architecture and pin efficiency and have been described previously. They are listed here again as follows to highlight their existence but with no further description.

- Provide all IEEE 1149.1 capability
- Star Scan Topologies/two-pin operation
- Test and debug power management
- Both scan and non-scan transfers with two-pin operation

- Shared use of the TCK(C) and TMS(C) pins with other technologies and multiple scan topologies
- Scalable implementation, use only what is needed

Two additional capabilities are highlighted, as follows:

- Chip-level functional and test resets
- Customizable private commands and registers

#### 4.4.2 Resets

The T1 and above TAP.7s may be implemented with one or more functional resets and one or more Test Resets. These resets are controlled with TAP.7 Controller commands. The functional resets are used to initialize subsystems within the functional logic. The Test Resets are used to initialize subsystems within the System Test Logic.

#### 4.4.3 Private commands and registers

The T1 and above TAP.7s may be implemented with one or more private commands and registers. The TAP.7 architecture permits the use of the same private commands in all TAP.7 Controllers sharing the DTS connection in both Series and Star Scan Topologies.

### 4.5 IEEE 1149.7 architecture

#### 4.5.1 Components

The TAP.7 hardware architecture is shown in Figure 4-12. It is described with the hardware layers listed as follows:

- STL System Test Logic—logic with the IEEE 1149.1-Specified Behavior. This logic may have an underlying TAP hierarchy (Foundation)
- RSU Reset and Selection Unit—a hardware layer placed between the APU, EPU, or STL and the TAP.7 signals (added as an option to support the use of the Control Protocol)
- EPU Extended Protocol Unit—a hardware layer placed between the STL and RSU or between the STL and the TAP.7 signals (added for the T1, T2, and T3 TAP.7s)
- APU Advanced Protocol Unit—a hardware layer placed between the RSU and the EPU (added for the T4 and T5 TAP.7s)

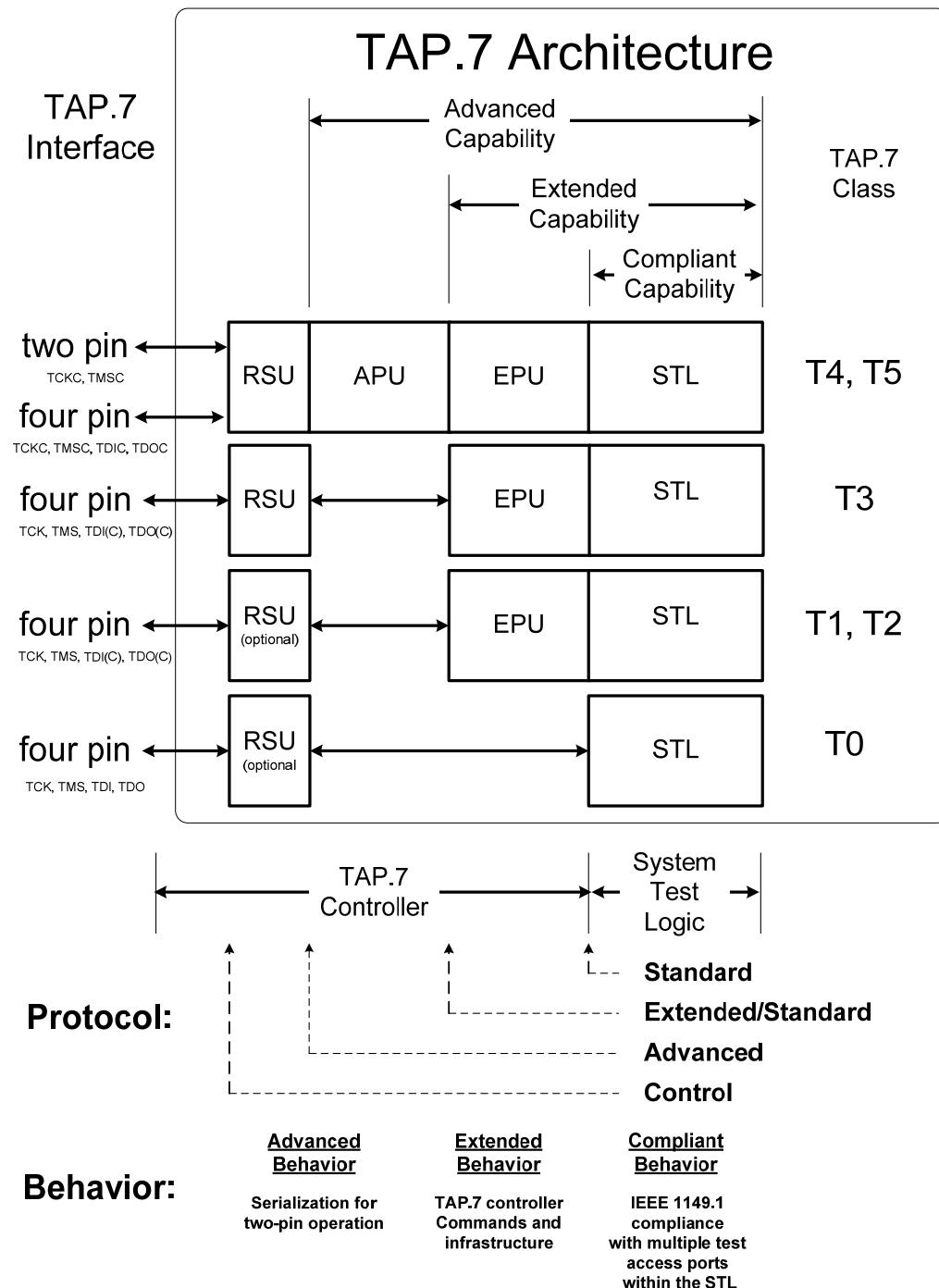


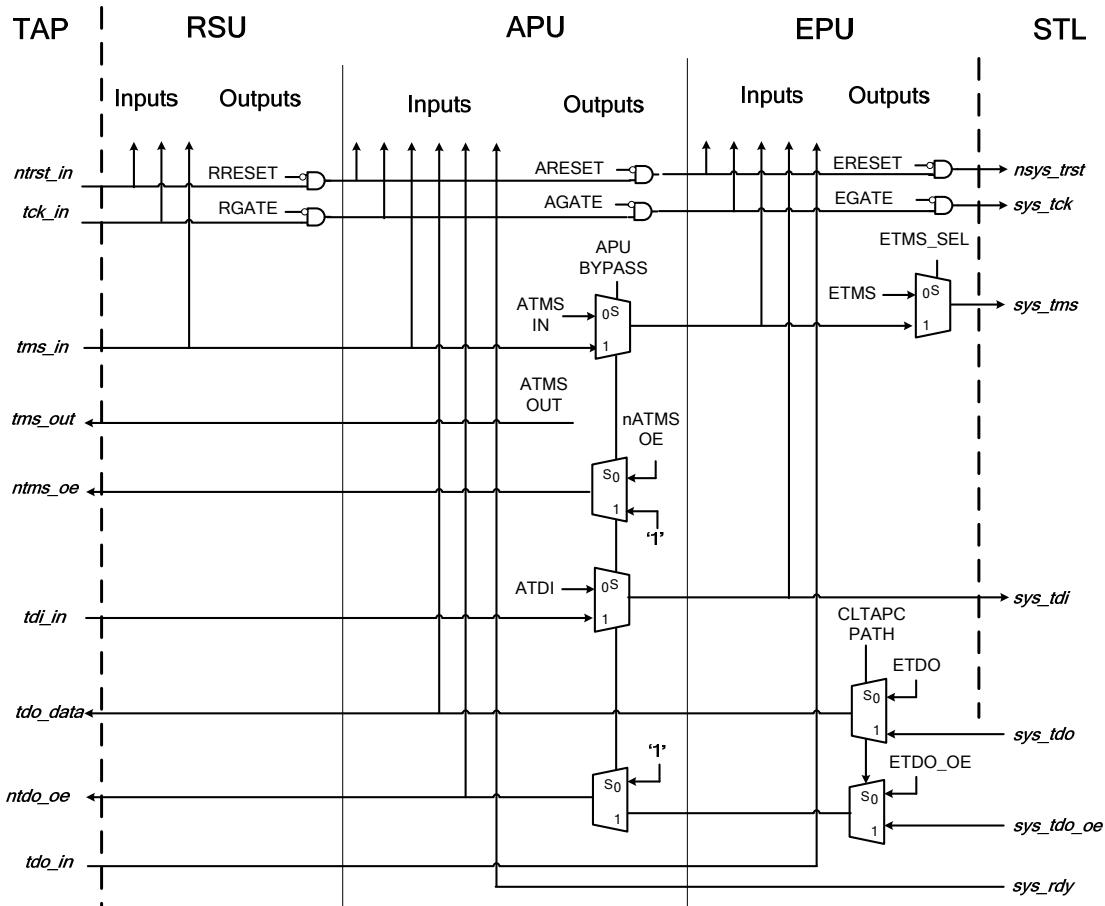
Figure 4-12 — TAP.7 architecture

Within this specification, the STL refers to chip test logic. A chip's functional logic is not included in the STL. It provides an IEEE 1149.1 interface for the T0 TAP.7. The RSU provides reset and TAP.7 Controller selection services. The EPU provides an IEEE 1149.1 interface for the T1-T3 TAP.7s. The APU provides a T4-T5 TAP.7 interface that is either narrow or wide, with the wide version providing an IEEE 1149.1 interface. The scan paths associated with the TAP.7 architecture are referenced using the entity in which they reside, for example a TAP.7 Controller Scan Path, an STL Scan Path, etc. With this architecture, the Test Reset signal is optional with all TAP.7 Classes and when implemented may have either pull-up

(nTRST) or pull-down (nTRST\_PD) behavior. These signals should be connected separately to the DTS unless they are continuously driven by a driver within the TS.

The TAP.7 Controller may be constructed with any of the combinations of RSU/EPU/APU shown in Figure 4-12. The TAP.7 Controller is partitioned in this manner to provide building blocks needed to construct TAP.7 Controllers for all the TAP.7 Classes and allow the RSU capability to be utilized with other technologies. A TAP.7 Controller constructed with an RSU is interoperable with other TAP.7 Controllers constructed with an RSU, independently of the scan topology in which the TAP.7 Controllers are deployed. A TAP.7 Controller with an RSU may also be operated with other technologies as shown in Figure 1-9 provided these technologies are utilize the RSU's selection and deselection functions.

The RSU, APU, and EPU functions provide a bridge between the TAP signals and the STL. A conceptual view of this bridge is shown in Figure 4-13.



**Figure 4-13 — Conceptual view of the interface bridge**

The bridge modifies the interface signals as they pass between the TAP and STL as shown in Figure 4-13. These signal modifications are described briefly as follows:

- *nsys\_trst*: The RSU, APU, and EPU can all asynchronously reset the STL.
- *sys\_tck*:
  - RSU gates to place Offline
  - APU gates to control the advance of the TAPC state

- EPU gates for initialization purposes
- tck\_in otherwise
- *sys\_tms*:
  - APU sources with the Advanced Protocol
  - EPU sources for initialization
  - tms\_in otherwise
- *sys\_tdi*:
  - APU sources with the Advanced Protocol
  - tdi\_in otherwise
- *ntms\_oe*:
  - APU sources with the Advanced Protocol
  - Inactive otherwise
- *tms\_out*:
  - APU sources with the Advanced Protocol
  - A don't care otherwise
- *tdo\_data*:
  - EPU sources when the STL is bypassed
  - EPU sources when its resources are accessed
  - STL sources otherwise
- *ntdo\_oe*:
  - EPU sources when the STL is bypassed
  - EPU sources when its resources are accessed
  - CLTAPC sources otherwise

Note that the APU's inputs include *sys\_tdo* and *sys\_tdo\_oe* to allow serialization of the TDO information. It also receives System Ready information as the Advanced Protocol allows a stall of the transfer by the STL, in some cases (depending on the scan format).

The RSU, APU, and EPU place only combinatorial logic in series with the STL Scan Paths. This means they do not affect the length of the STL Scan Paths seen by the DTS. This process facilitates the separation of the TAP.7 Controller and device drivers in the DTS software stack controlling a chip or chips. Existing drivers for components within the STL may be used to control these components with T1–T5 TAP.7 Controllers when the STL is connected to the DTS. The connection of the DTS and STL is generally managed by a software layer that is independent of the functions using the STL Scan Paths.

#### 4.5.2 Reset types

The TAP.7 Controller supports the following six types of resets:

- Type-0      Reset of the TAP.7 power management logic
- Type-1      Externally generated Test Reset
- Type-2      Chip-level generated Test Reset
- Type-3      TAP.7 Controller generated reset

- Type-4      *Test-Logic-Reset* state
- Type-5      Register controller reset of the CLTAPC

The Type-1, Type-2, and Type-4 Resets are supported by IEEE Std 1149.1. The Type-0 Reset is added when the optional TAP.7 power-management logic is deployed with a T1 and above TAP.7. The Type-3 Reset is added with the use of an RSU or the Advanced Protocol. The Type-5 Reset is an option available with T1 and above TAP.7s.

#### 4.5.3 Start-up options

The four TAP.7 start-up options are listed as follows:

- IEEE 1149.1-Compliant
- IEEE 1149.1-Compatible
- IEEE 1149.1-Protocol Compatible
- Offline-at-Start-up

The IEEE 1149.1-Compliant start-up option provides IEEE 1149.1-Specified Behavior. With this start-up option, the TAP.7 Controller executes the following actions:

- Presumes the TAP signaling is the Standard Protocol.
- Advances the ADTAPC state.
- Advances the CLTAPC state.
- Connects the STL Scan Path to the TAP.

The IEEE 1149.1-Compatible start-up option provides IEEE 1149.1-Non-disruptive Behavior. With this start-up option, the TAP.7 Controller executes the following actions:

- Presumes the TAP signaling is the Standard Protocol.
- Advances the ADTAPC state.
- Parks the CLTAPC in the *Test-Logic-Reset* state.
- Connects a one-bit TAP.7 Controller Scan Path to the TAP, bypassing the STL Scan Path.

The IEEE 1149.1-Protocol Compatible start-up option provides IEEE 1149.1-Other Behavior. With this start-up option, the TAP.7 Controller executes the following actions:

- Presumes the TAP signaling is the Standard Protocol.
- Advances the ADTAPC state.
- Parks the CLTAPC in the *Test-Logic-Reset* state.
- Connects a one-bit TAP.7 Controller Scan Path to the TAP, bypassing the STL Scan Path.
- Forces the STL's TDI signal to a logic 1 and ignores the STL's TDO and TDO output enable signals.
- The TAP TDI(C) and TDO(C) signals are either non-existent or are used for auxiliary functionality.

The Offline-at-Start-up (*OLS*) option provides IEEE 1149.1-Other Behavior. With this start-up option, the TAP.7 Controller:

- Presumes the TAP signaling is the Control Protocol.

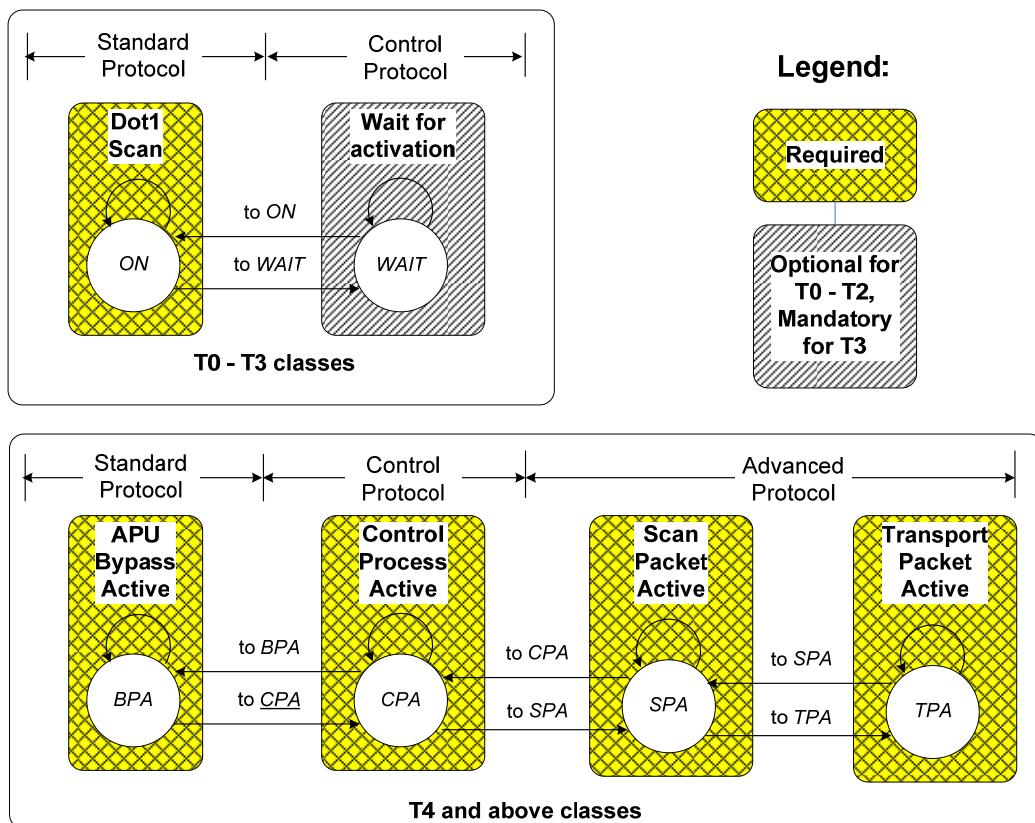
- Moves the CLTAPC state to the *Run-Test/Idle* state irrespective of the TMS(C) input and then parks the CLTAPC state in this state.
- Waits for a signaling sequence activating the TAP.7 Controller (placing it Online) once the ADTAPC state is parked.

The IEEE 1149.1-Protocol Compatible start-up option is only available with T4 and above TAP.7.

## 4.6 Operating models

### 4.6.1 Model types

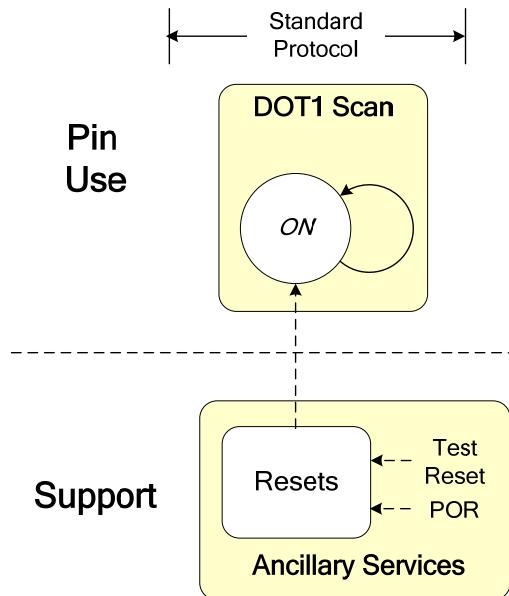
The operation of the TAP.7 Controller is described in this document with the models shown in Figure 4-14. The model used for T0–T3 and T4 and above TAP.7s use different operating state names [*ON* and *WAIT* vs. *BPA*, *Check Process Active (CPA)*, *Scan Packet Active (SPA)*, and *Transport Packet Active (TPA)*]. This name change highlights the fact that the selection/deselection functions of T4–T5 TAP.7s are a superset of the very similar T0–T3 TAP.7 selection/deselection functions.



**Figure 4-14 — Operating model types**

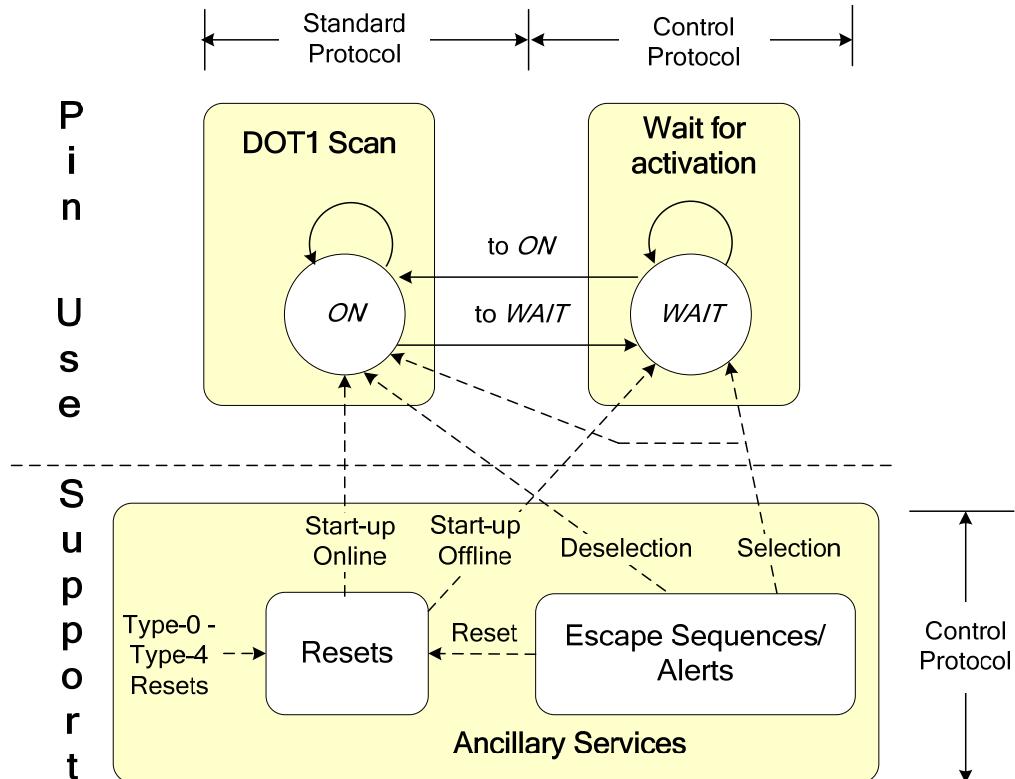
### 4.6.2 Standard models

The operation of a T0–T2 TAP.7 Controller operation, without an RSU, is shown in Figure 4-15.



**Figure 4-15 — Conceptual view of basic T0-T2 TAP.7 operation without RSU**

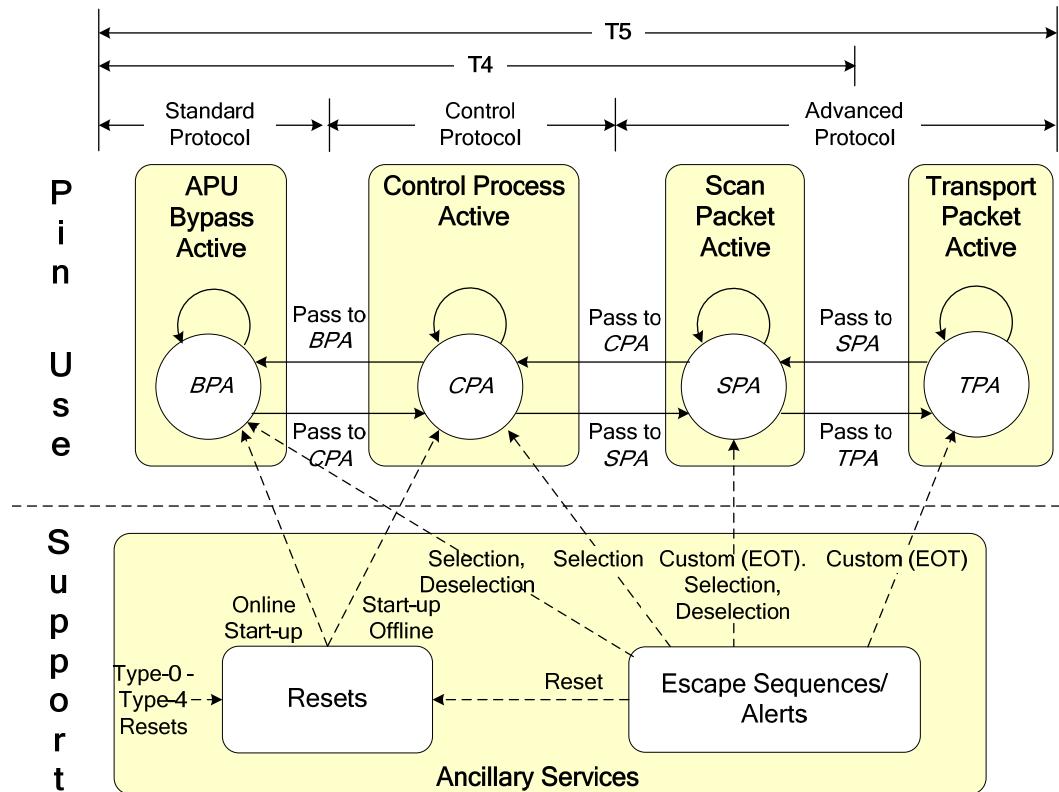
The operating model of a T0-T3 TAP.7 with the RSU is shown in Figure 4-16. In this case, the use of a Reset Escape to reset the TAP.7 Controller may be considered in lieu of implementing the Test-Reset signal function. The RSU also provides the ADTAPC selection and deselection facilities required to operate these TAP.7s as part of a Custom Scan Topology. This topology may be a combination of Series, Star-4, and Star-2 Scan Topologies.



**Figure 4-16 — Conceptual view of T0-T3 TAP.7 operation with an RSU**

#### 4.6.3 Advanced models

The operation of a T4–T5 TAP.7 Controller operation is shown in Figure 4-17. The Transport Packet Active state is present only with a T5 TAP.7. These TAP.7 Controllers use the Standard, Control, and Advanced Protocols, with the scan portions of this protocol used by both T4 and T5 TAP.7s and the transport portion of this protocol used only by a T5 TAP.7.



**Figure 4-17 — TMSC pin utilization flow diagram for a T5 TAP.7**

The states shown in this figure are called APU Operating States. These states are conceptual in nature and are associated with the APU functions sharing the use of the TMSC signal. They describe the passing of control of the TMSC signal from one function to another. The state names shown in this figure have the following meaning:

- *BPA* Bypass Active
- *CPA* Check Process Active
- *SPA* Scan Packet Active
- *TPA* Transport Packet Active

With both T4 and T5 TAP.7s, the *BPA*, *SPA*, and *CPA* states support the following functionality:

- *BPA* Scan transfers using the Standard Protocol
- *SPA* Scan transfers using the Advanced Protocol
- *CPA* Online/Offline Operation/Standard and Advanced Protocol switches using the Control Protocol

With a T5 TAP.7s, the *TPA* state supports the data transport functionality.

## 5. T0–T3 TAP.7 operational overview

### 5.1 Introduction

This clause provides an overview of the T0–T3 TAP.7s and their capabilities. The T0–T2 TAP.7s may be deployed only in Series Scan Topologies, and the T3 TAP.7 may be deployed in either the Series or Star-4 Scan Topologies.

The subject matter within this clause is organized in the following manner:

- 5.2 T0 TAP.7
- 5.3 T1 TAP.7
- 5.4 T2 TAP.7
- 5.5 T3 TAP.7

### 5.2 T0 TAP.7

#### 5.2.1 Overview

Due to the large-scale reuse of existing IP modules, current system chip designs contain multiple TAPCs connected in series and visible after a *Test-Logic-Reset* state. These TAPCs manage a range of functionality from boundary scan to debug control. This approach violates the rules IEEE Std 1149.1 imposes on the length of the DR Scan Path for the *BYPASS* and *IDCODE* instructions as IEEE Std 1149.1 described the function of a package or device boundary. This approach is at odds with the natural evolution of the industry, where multiple TAPCs are deployed within a package or device. This standard provides for the deployment of multiple TAPCs with a chip, package, device, board, and system boundary. This standard supports test language features that allow the description of systems under test constructed with combinations of chips, packages, devices, boards, and subsystems.

A T0 TAP.7 provides a Chip-Level TAPC architecture (MTAP) where multiple TAP.7 Controllers may be utilized. This architecture provides IEEE 1149.1-Specified Behavior following the *Test-Logic-Reset* state independent of the TAPC hierarchy.

When the selection and deselection of a T0 TAP.7 is desired (operation with TAP.7s configured in Scan Topologies other than Series or with other technologies utilizing the RSU function), the optional RSU is placed between the TAP signals and the Chip-Level Logic as shown in Figure 5-1. The RSU contains reset and/or selection/deselection capability depending on the options implemented. It supports the parking of the ADTAPC state. It does not add bits in series with the STL Scan Paths. The actions taken by the RSU are initiated with Escapes and optionally with Selection Alert Bit Sequences and Deselection Alert Bit-Sequences.

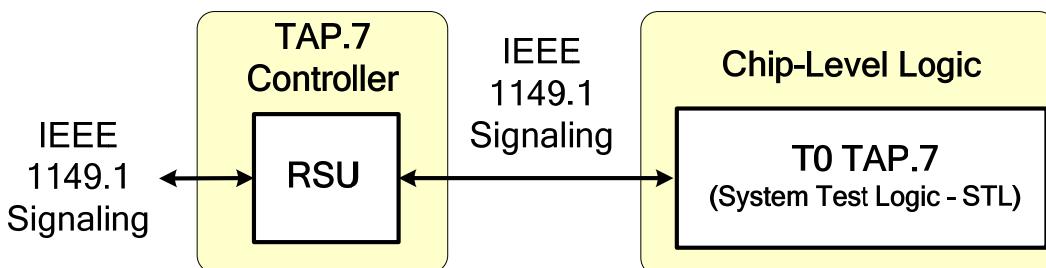


Figure 5-1 — T0 TAP.7 chip block diagram

When a T0 TAP.7 provides IEEE 1149.7-Specified Behavior, the Chip-Level Logic shown in Figure 5-1 exhibits IEEE 1149.1-Specified Behavior with the following additional provisions:

- The Device Identification Code and associated *IDCODE* instruction is mandatory with the TAP.7.
- EMTAPCs are permitted as long as they are not visible to the DTS following a Test Reset.

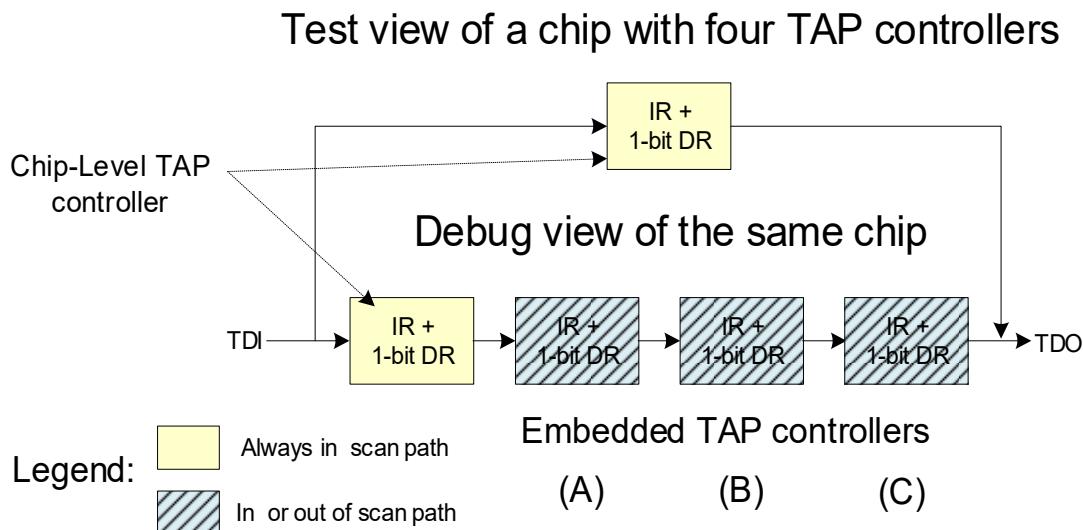
The RSU may be added with the T0 TAP.7 to support operation with Scan Topologies other than Series and with other technologies, such as follows:

- Deselection, Selection, and Reset Escapes
- Selection Alerts (optional)
- Deselection Alerts (optional)

### 5.2.2 Operating modes and capabilities

Board-level testing requires a TAP test view that meets the intention of IEEE Std 1149.1 that, at the board level, all chips behave in a standard way when providing board-test capability. The TAP used for the manufacturing test of boards has also increasingly become a debug portal, providing access to chip facilities that support applications debug and system integration. Many existing chips use the TAP to provide applications debug support and access to multiple EMTAPCs.

Test and debug processes desire two different views of a chip as shown in Figure 5-2.



**Figure 5-2 — T0 TAP.7 test and debug views of the chip**

The manufacturing test of boards does not require visibility to the system components related to applications debug, and debug does not require visibility to the interconnection between the chips. The test process manages all chips while debug manages one or more components across one or more chips. A T0 TAP.7 provides both the test and the debug views to meet these requirements.

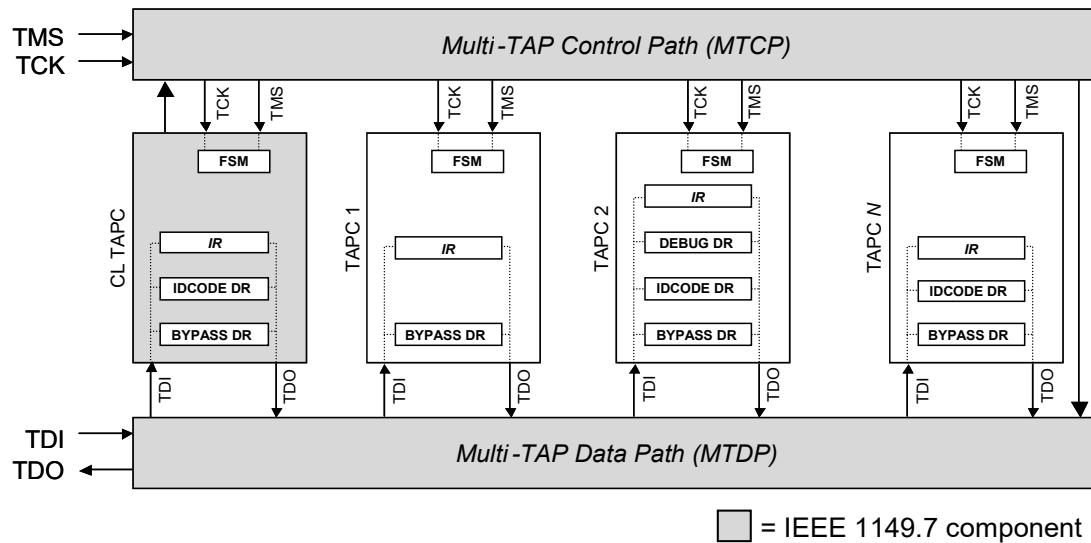
With a T0 TAP.7, the chip has a CLTAPC and may have EMTAPCs. The *Test-Logic-Reset* state creates the test view. With this view, the CLTAPC is the only TAPC visible from the TAP. The test view provides IEEE 1149.1-Specified Behavior until an action is initiated via the CLTAPC to provide the debug view. In most cases, the test view can also be created with STL scan activity.

The debug view provides IEEE 1149.1-Non-disruptive Behavior and access to TAPCs embedded in the chip. The chip looks like a board with components appearing as chips. The EMTAPCs are managed in a way that accommodates their being unavailable for use (for instance embedded in a block that is powered down). In most cases, a TAPC embedded with an on-chip component controls the component but does not have a boundary scan associated with it. An EMTAPC may or may not support the Device Identification Code and associated *IDCODE* instruction. The Device Identification Code and associated *IDCODE* instruction is mandatory for the CLTAPC.

### 5.2.3 Operation

#### 5.2.3.1 Multi-TAPC architecture

A generic view of the IEEE 1149.7 multi-TAPC architecture for accessing multiple on-chip TAPCs is shown in Figure 5-3. With a T0 TAP.7, the CLTAPC is always present in the scan path with the STL providing the scan path after the *Test-Logic-Reset* state. After exiting this state, the DTS can dynamically alter the Chip-Level Scan Path using the CLTAPC and the scan paths it manages. EMTAPCs are connected to the Chip-Level Scan Path in a manner where the Data and Instruction Registers associated with the CLTAPC are connected to the TAP.7 TDI signal, where the Data and Instruction Registers associated with the EMTAPCs are connected in the scan chain thereafter.



**Figure 5-3 — System-on-Chip (SoC) with IEEE 1149.7 multiple TAPC architecture**

#### 5.2.3.2 Selection and deselection of EMTAPCs

With the Multiple TAP architecture, the DTS may park one or more EMTAPCs in the *Run-Test/Idle* state while the state progression of their parent, the CLTAPC, continues. The DTS may park the EMTAPC state when there is no desire to use the scan paths it manages and parks the EMTAPC state when, for any reason, the EMTAPC state is not viable (for example, the EMTAPC is powered-down).

Parking the EMTAPC state postpones actions that are initiated by a stay in the *Run-Test/Idle* state of more than one TCK period or upon the exit from the *Run-Test/Idle* state until resynchronization occurs. This allows the synchronization of actions such as running or halting an applications processor attached to different TAPs within a chip or different chips within a system using the mechanisms described above. The

EMTAPC state may be set to the *Test-Logic-Reset* by power cycling or another event while parked. The continuity of the STL Scan Path is maintained when the EMTAPC state is parked.

The DTS may resynchronize the state progression of an EMTAPC with a previously parked state to the operation of the CLTAPC after it verifies the EMTAPC is powered and presents a viable scan path. This action synchronizes the EMTAPC state to the CLTAPC state with their state progressions proceeding in lock-step thereafter. A two-state stay in the *Run-Test/Idle* state is sufficient to resynchronize an EMTAPC *Run-Test/Idle* or *Test-Logic-Reset* state to the CLTAPC *Run-Test/Idle* state. Information generated prior to reaching the *Run-Test/Idle* state determines whether the EMTAPC state is parked or resynchronized when the *Run-Test/Idle* state is reached.

The DTS adjusts its scan transactions to accommodate the scan path changes created by parking and synchronizing actions. The DTS is responsible for ensuring an EMTAPC is operable and presents a viable scan path before synchronizing the EMTAPC's operation to the CLTAPC.

With this approach, scan path control information generated prior to reaching the *Run-Test/Idle* state determines whether the inclusion or exclusion of EMTAPC managed DR Scan Paths occurs when this state is reached. This information may be developed with any of the following:

- The CLTAPC Instruction Register
- One or more CLTAPC Data Registers
- One or more device signals that comply with the IEEE 1149.1 specifications for **Subordination of this standard within a higher level test strategy**

Note that the DTS will have knowledge of the use of the device signals described above in order to understand the System Scan Path topology provided these signals are used. The failure to provide this knowledge may result in system failures as the DTS-generated scan activity will not match the actual Chip-Level Scan Paths.

#### 5.2.4 T0 TAP.7 high-level block diagram

With a T0 TAP.7, the RSU is placed between the STL and TAP signals when Offline operation (parking the ADTAPC state) is desired. This is required for the use of a T0 TAP.7 with other technologies or with TAP.7 Controllers configured in scan topologies other than Series. The RSU is omitted and the STL is connected to the TAP pins when Offline operation is not desired. A block diagram of a T0 TAP.7 with the RSU deployed is shown in Figure 5-4.

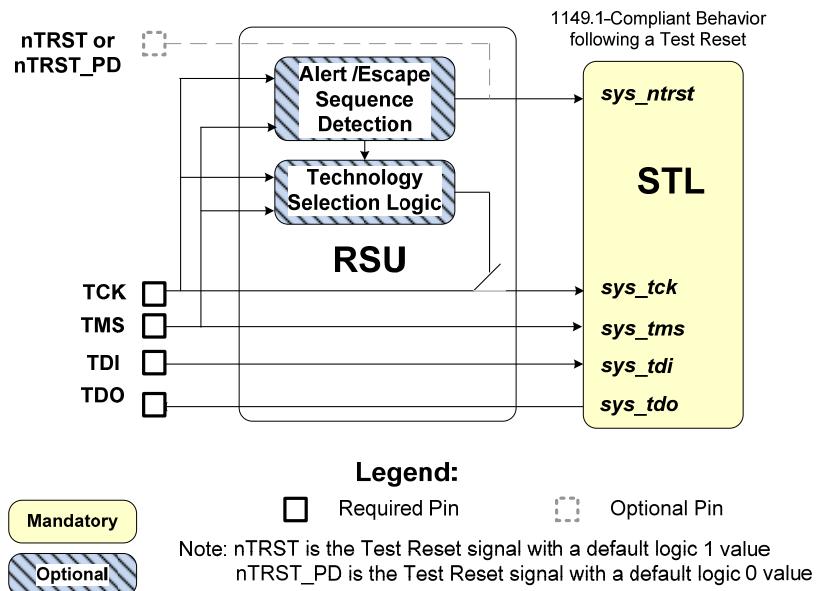


Figure 5-4 — High-level T0 TAP.7 functional block diagram

## 5.3 T1 TAP.7

### 5.3.1 Overview

A T1 TAP.7 inherits the T0 TAP.7 mandatory features. It provides IEEE 1149.1-Specified Behavior and supports optional features such as the generation of functional/test resets to Chip-Level Logic and TAP.7 Controller power management.

A conceptual view of a T1 TAP.7 is shown in Figure 5-5. The EPU is placed between the TAP signals and a T0 TAP.7 to create a T1 TAP.7. The RSU is placed between the EPU and the TAP signals when Offline operation is desired. The T0 TAP.7 is commonly called the System Test Logic (STL) when describing the T1-T5 TAP.7 Classes. The EPU provides the foundation for the capabilities provided by these TAP.7 Classes.

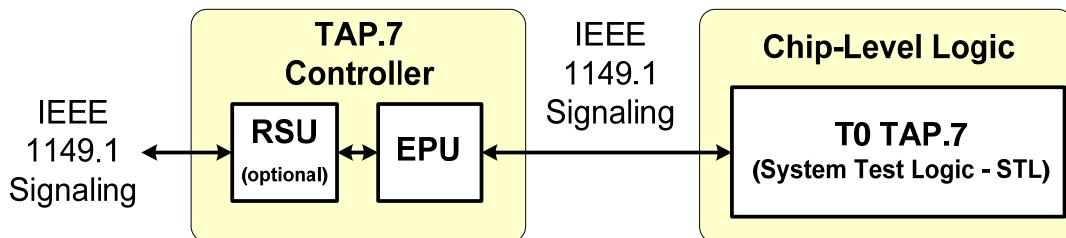


Figure 5-5 — T1 TAP.7 chip block diagram

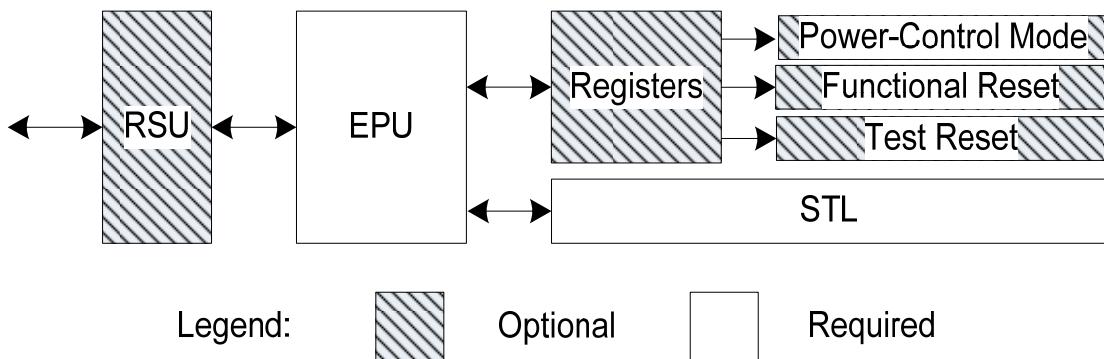
The EPU includes the Adapter TAPC State Machine (ADTAPC), contains no instruction and data registers, and adds no bits in series with the T0 TAP.7 Scan Paths. The EPU is controlled with the ADTAPC State Machine state progressions, where TDI and TDO data is not needed. These TAPC state progressions are benign and do not change the state of a TAP.1 Controller in any noticeable way. This makes the EPU transparent to the TAPCs within the STL and elsewhere in a system. The EPU functions are managed using the Control Scan Paths described in 4.2.6.2.

### 5.3.2 Operating modes and capabilities

The mandatory aspects of T0 TAP.7 are also mandatory for a T1 TAP.7. The options available for a T0 TAP.7 are also options for a T1 TAP.7. The T1 TAP.7 provides a standardized implementation for the test and functional reset functions currently used in IEEE 1149.1 systems but implemented in non-standard ways. It also supports the power-down of test and debug logic. The optional T1 TAP.7 functions include the following:

- Requesting the system logic to generate a functional reset
- Asserting a test reset to the STL
- Power-Control Modes allowing the following:
  - The power-down of test/debug logic during normal system operation
  - The power-down and power-up of the TAP.7 Controller under DTS control
  - Always powered operation of the TAP.7 Controller as directed by the DTS

A conceptual view of these functions is shown in Figure 5-6.



**Figure 5-6 — T1 TAP.7 features**

### 5.3.3 Operation

#### 5.3.3.1 Adding TAP.7 functionality to the *BYPASS* and *IDCODE* instructions

The TAP.7 architecture uses IEEE 1149.1 TAPC state sequences to control TAP.7 capabilities in a manner transparent to TAP.1s. The DTS controls the EPU using a combination of the *BYPASS* and/or *IDCODE* instructions and seldom, if ever, used IEEE 1149.1 TAPC-state sequences called zero-bit DR Scans (ZBSs). The functionality assigned to these instructions by IEEE Std 1149.1 is not changed.

#### 5.3.3.2 Zero-bit DR scans

A zero-bit DR Scan is a TAPC state sequence that begins with the *Select-DR-Scan* state and ends with an exit from the *Update-DR* state without an intervening *Shift-DR* state. This TAPC State Machine state sequence is rarely, if ever, used with the *BYPASS* and *IDCODE* instructions in legacy systems as it performs no real function. These TAPC state sequences depend on the ability of the IEEE 1149.1 *BYPASS* and *IDCODE* instructions to tolerate the use of capture data as update data during ZBS sequences. The use of ZBSs with these instructions is benign and does not materially change the state of the STL. The TAPC state progressions that are considered a ZBS are shown in Annex A, Figure A-3, and Figure A-4.

The use of ZBSs may begin immediately after the *Test-Logic-Reset* state as this state initializes the Instruction Registers with a value specifying either the *BYPASS* or the *IDCODE* instruction (the *IDCODE* instruction in the case of a T0–T5 TAP.<sup>7</sup>).

### 5.3.3.3 Utilizing ZBSs for TAP.7 Controller functionality

Beginning with a ZBS count of zero, the ZBS count is incremented with each consecutive occurrence of a ZBS. This count cannot be incremented past seven. *Run-Test/Idle* is the only state that may occur between consecutive occurrences of ZBSs without terminating or initializing the count of consecutive ZBSs. The TAP.7 architecture supports the use of ZBSs for both TAP.7 Controller and STL purposes as described below.

#### 5.3.3.3.1 Locking the ZBS count

When a DR Scan containing a *Shift-DR* state occurs and the ZBS count is greater than zero, the ZBS count is locked at its current value. ZBSs occurring while the ZBS count is locked do not affect the locked ZBS count. Locking the ZBS count is the equivalent of storing the count for subsequent use.

#### 5.3.3.3.2 Control levels

The creation of a control level starts with no control level and an EPU Operating State of *RUNNING* (see 5.3.3.4 for a description of EPU Operating States). EPU Operating States determine whether ZBSs are recognized and, if recognized, their use by the TAP.7 Controller to perform TAP.7 Controller functions. Locking the ZBS count activates a control level equal to the locked ZBS count (1–7) when ZBSs are being used to perform TAP.7 Controller functions and merely locks the ZBS count otherwise.

The control levels shown in parenthesis in the following associate the functionality listed with DR Scans:

- ZBSs may be used by the STL or another function (1)
- TAP.7 command generation (2)
- Reserved control level (3)
- Accesses to optional TAP.7 Controller Scan Paths (4–5)
- Force Offline operation (optional) (6)
- DTS uses (7)

Control Level One may be used by the STL since there is no TAP.7 Controller functionality associated with this control level. When the control level is two, DR Scans are used to create TAP.7 commands. These commands manage TAP.7 Controller registers and perform other functions.

Control Level Three is reserved for use with a subsequent specification revision. Control Levels Four and Five provide access to Custom (Private) Scan Paths when access to these paths is permitted by a TAP.7 register bit used for this purpose. The EPU provides a One-bit Scan Path otherwise. No other EPU functionality is associated with these control levels.

Control Level Six may be used to create a Deselection Alert to force Offline operation (the state of the ADTAPC is parked) with TAP.7 Controllers that supports this capability. The EPU provides a One-bit Scan Path for this control level.

Control Level Seven is reserved for DTS use. With this control level, DTS capabilities may be managed using the same infrastructure used to manage TAP.7 Controller capability. The EPU provides a One-bit Scan Path for this control level.

### 5.3.3.3.3 Exiting a control level

A control level is exited when one of the following occurs:

- The *Select-IR-Scan TAP.7 Controller state*
- The *Test-Logic-Reset state*
- Certain TAP.7 Controller commands
- An event that synchronizes the operation of TAP.7 Controllers

### 5.3.3.3.4 Zeroing the ZBS count

Exiting a control level zeroes the ZBS count. Scan Selection Directives (see 5.5.3.5) associated with T3 and above TAP.7 also zero the ZBS count when the control level has not been locked.

## 5.3.3.4 EPU Operating States

The three EPU Operating States listed as follows support the use of ZBSs for both EPU and STL purposes:

- *RUNNING*
  - The DTS and TAP.7 Controller assume ZBSs affect the generation of control levels.
  - The DTS should not use ZBSs in a manner that affects STL operation.
  - Commands are used to initiate the *SUSPENDED* and *INHIBITED* states.
- *SUSPENDED*
  - The DTS and TAP.7 Controller assume ZBSs do **not** affect the generation of control levels.
  - The DTS may use ZBSs in a manner that affects STL operation.
  - A ZBS count greater than seven initiates a return to the *RUNNING* state.
- *INHIBITED*
  - The DTS and TAP.7 Controller assume ZBSs do **not** affect the generation of control levels.
  - The DTS may use ZBSs in a manner that affects STL operation.
  - A TAP.7 Controller reset initiates return to the *RUNNING* state.

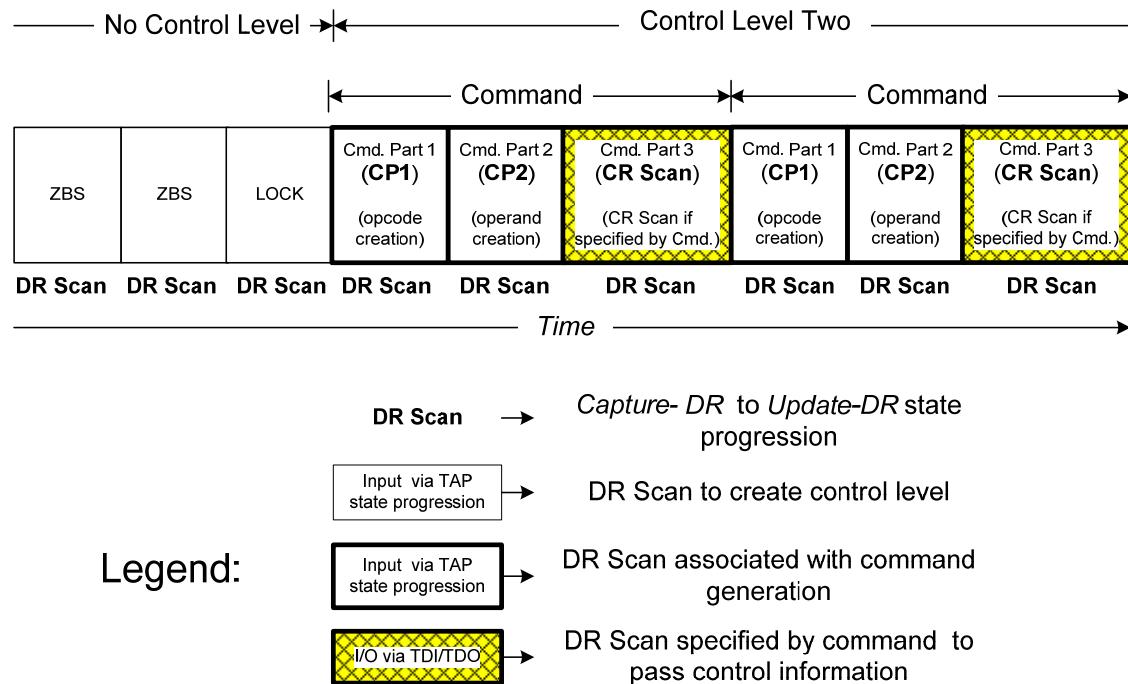
The Suspend (SUSPEND) and ZBS Inhibit (ZBSINH) Registers define the EPU Operating State. Storing the SUSPEND Register suspends the use of control levels and creates the *SUSPENDED* state. Storing the ZBSINH Register creates the *INHIBITED* state. Storing either of these registers zeroes the ZBS count. Storing the ZBSINH Register also inhibits ZBS detection.

The *Test-Logic-Reset state* creates the *RUNNING* state. Eight consecutive ZBSs prior to locking the ZBS count while in the *SUSPENDED* state creates the *RUNNING* state and zeroes the ZBS count.

### 5.3.3.5 Commands

TAP.7 Controller commands are created with two consecutive DR Scans once the control level is locked at two (the Command Control Level) as shown in Figure 5-7. The DR Scans creating a command are called Command Part One (CP1) and Command Part Two (CP2), respectively. A command created entirely with these two scans is called a two-part command. The command specified by CP1 and CP2 may specify a three-part command. In this case, CP2 is followed by a Control Register Scan (CR Scan). There are many two-part commands, but only three three-part commands, each having a special purpose. Any number of two- and three-part commands may be issued in any combination before the Command Control Level is

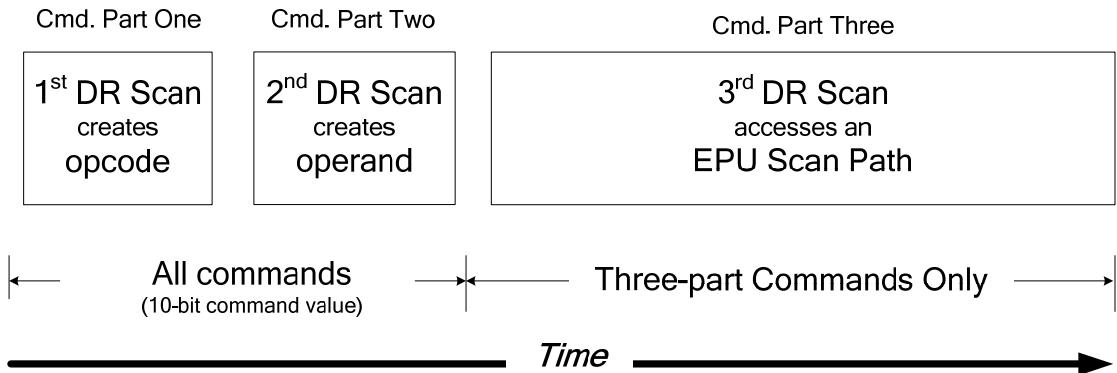
exited. If the Command Control Level is exited (the *Test-Logic-Reset* state occurs) before a two- or three-part command is completed, the command is aborted and becomes a no-operation.



**Figure 5-7 — Command generation**

In both the Series and Star Scan Topologies, two-part commands are used to store Global Registers. The action created by execution of some two-part commands is dependent on the value of the Conditional Group Membership (CGM) Register. This is called “Conditional” command execution. These commands perform their specified function provided the CGM Register value is a logic 1 and perform no operation otherwise. This provides for storing the same Local Register simultaneously in one or more TAP.7 Controllers or a Local Register in one or more TAP.7 Controllers in both the Series and Star Scan Topologies. Three-part commands are used to write Local Registers in one or more TAPCs in either a Series or Star Scan Topology. They are used to read Local Registers in one or more TAPCs in a Series Scan Topology, or read Local Registers in a single TAPC in a Star Scan Topology.

A TAP.7 Controller command is a 10-bit value. It is composed of a five-bit opcode created by CP1 and a five-bit operand created by CP2 as shown in Figure 5-8. The five-bit opcode and operand values are concatenated to form a 10-bit command. A command is decoded when the *Update-DR* state of the DR Scan associated with CP2 is reached. This decode determines whether the command is a two- or three-part command.



**Figure 5-8 — DR-Scan sequence used for command creation**

The five-bit opcode and operand values are created entirely with a TAPC-state progression. These values represent the count, truncated to five bits, of the number of *Shift-DR* TAPC states between the *Capture-DR* and *Update-DR* TAPC states of the DR Scans associated with CP1 and CP2. The *Shift-DR* TAPC state count within CP1 creates the MSBs of the command (the op-code). The *Shift-DR* TAPC state count within CP2 creates the LSBs of the command (the operand). The TDI and TDO data associated with the CP1 and CP2 DR Scans is ignored.

All two-part commands may be generated using either the Standard or the Advanced Protocol with only the TCK(C) and the TMS(C) signals since the creation of two-part commands uses only the TAPC-state progression. This becomes significant for two-signal operation with the T4 and T5 TAP.7 Classes, as either of the above protocols may be used to create all commands.

A two-part command stores a Global or Local Register, with the value stored embedded in the command operand. The target of a two-part command may be as follows:

- A Global Register
- A Local Register that is unconditionally stored by a command
- A Local Register that is conditionally stored by a command based on the value of another register bit
- A Local Register that is conditionally stored by a command, based on the Controller ID supported in T3–T5 TAP.7s

With a three-part command, the CR Scan accesses (reads, writes, or both reads and writes) Local Register values, with controller information transferred with TDI and TDO values. These values are conveyed using the TDI(C) and TDO(C) signals with the Standard Protocol and with TDI and TDO bits within Scan Packets with the Advanced Protocol. With two of the three-part commands, Local Registers are conditionally accessed only when permitted based on the value of another register bit. The third three-part command manages CID allocation.

With two-part commands, the function specified by the command is performed when CP2 completes (the *Update-DR* state of CP2 completes). The value written to a register is embedded in the operand field. In this case, this value is written to a register at the falling edge of the EPU Test Clock following an exit from the *Update-DR* state at the end of CP2.

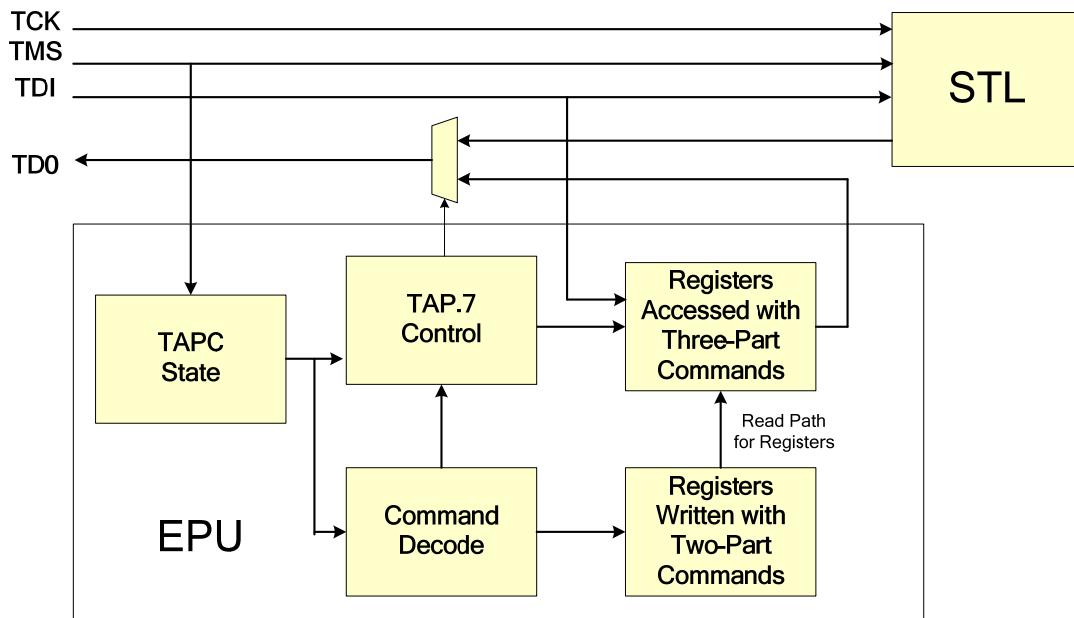
With three-part commands, the function specified by the command is performed with the CR Scan following CP2. This CR Scan has the attributes of the DR Scan as it moves data between the DTS and TAP.7 Controller with TDI and TDO information. The scan path for a CR Scan is a scan path within the TAP.7 Controller, with CR Scans provided to various EPU Scan Paths. Hence, this scan is called a

Controller-Register Scan (CR Scan). A CR Scan is a minimum of zero bits in length with there being no maximum length.

### 5.3.3.6 Registers

In contrast with TAP.1 Registers, most TAP.7 Controller registers are not accessed with IR and DR Scans. The TAP.7 Controller registers are accessed using the commands described in 5.3.3.5 and may be write only, read-only, or read/write. TAP.7 Controller registers are described in Clause 9.

A high-level view of TAP.7 Controller registers is shown in Figure 5-9. Many registers are written with two-part commands. Certain other registers may be read, written, or simultaneously read and written with the CR Scan of a three-part command. Registers written with two-part commands may be read with three-part commands.



**Figure 5-9 — High-level view of TAP.7 Controller registers**

There are two types of TAP.7 Controller registers, Global and Local, as shown in Figure 5-10. A Local Register is private TAP.7 Controller information used to manage the characteristics of a specific TAP.7 Controller. Local Registers are stored using TAP.7 Controller commands dedicated for this purpose. A Local Register with a specific name may have a different value in each TAP.7 Controller. They manage functions such as the selection of certain Control Scan Paths in a TAP.7 Controller. An example of a Local Register is the Functional Reset (FRESET) Register used to initialize a functional reset within a chip.

Local Registers are stored with three types of commands:

- A two-part command that specifies the TAP.7 Controller whose Local Register is stored with a four-bit CID embedded in the command (see 20.9)
- A two-part command that conditionally stores a Local Register based on the value of the CGM Register
- A three-part command that conditionally stores a Local Register based where the CR Scan of the command conveys the Local Register value that is stored

Using a three-part command to store a Local Register in a TAP.7 Controller conditionally provides a means to store:

- A different value in the Local Register of each TAP.7 Controller in a Series Scan Topology
- The same value in the Local Register of each TAP.7 Controller in a Star Scan Topology

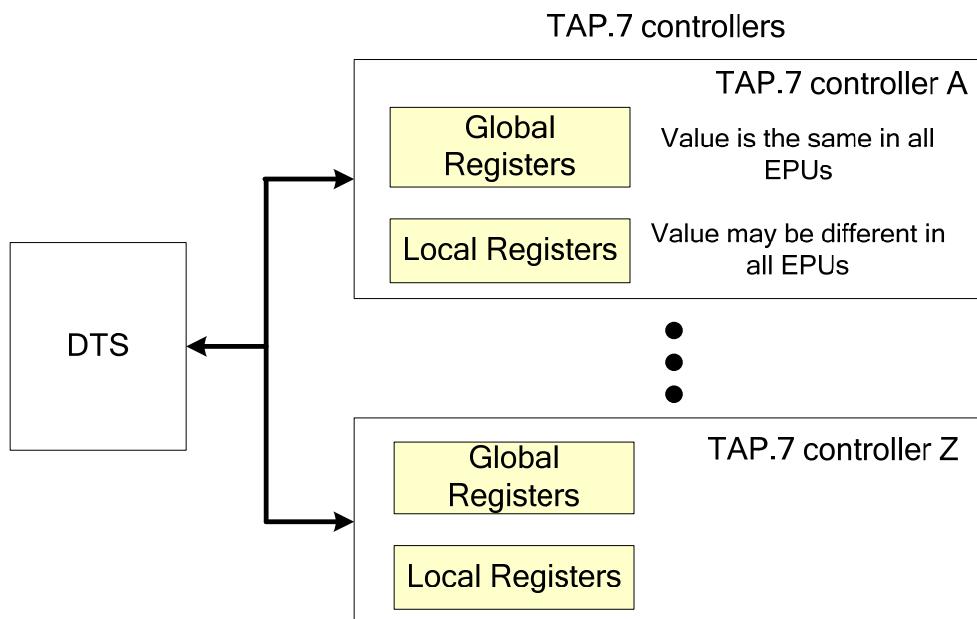
The commands that conditionally store a Local Register provide a means to store the same Local Register value in one or more TAP.7 Controllers (each TAP.7 Controller satisfying the condition).

When using a conditional three-part command to store a Local Register, the value stored in a Local Register is as follows:

- Based on the TAP.7 Controller's position on the scan chain as a legacy DR Scan transports the value to be stored in a Series Scan Topology
- The same for all TAP.7 Controllers as the DR Scan broadcasts the value to be stored in a Star Scan Topology

A Global Register is a copy of configuration information required for the joint operation of the TAP.7 Controllers sharing the DTS connection. These registers manage TAP.7 Controller functions that affect the synchronized operation of a TAP.7 Controller. An example of a Global Register is the Scan Format (SCNFMT) Register used to define the DTS/TS signaling protocol for scan transfers.

Global registers are stored using TAP.7 Controller commands dedicated for this purpose. They are stored simultaneously in all Online TAP.7 Controllers using TAP.7 Controller commands dedicated to managing Global Registers. Their value changes synchronously in all Online TAP.7 Controllers. A second means of storing Global Registers is provided with TAP.7 Controller selection via the Control Protocol following a Selection Escape. The operation of commands and attributes of the Control Protocol ensure Global Registers have the same value in all TAP.7 Controllers whose ADTAPC state is not parked (an Online TAP.7 Controller).



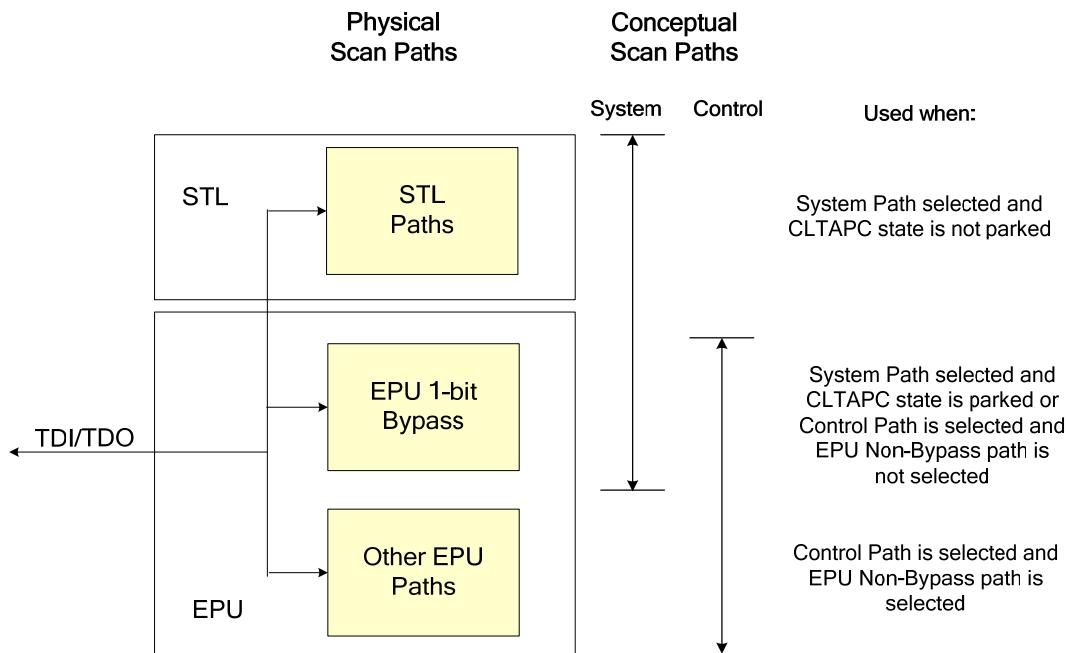
**Figure 5-10 — High-level view of TAP.7 Controller Global/Local Registers**

### 5.3.3.7 Using the System and Control Paths

The DTS' view of the T1 TAP.7 is one of using either System Test Logic or TAP.7 Controller resources. These resources are accessed using the System and Control Paths as described in 4.2.6.2. These scan paths are conceptual in nature. They do not indicate the physical scan path actually being used. With T2 and above TAP.7s, the physical scan path for the System Path is supplied by either the STL or the EPU. The physical scan path for the Control Path is always provided by the EPU. The relationship of the physical scan paths to the System and Control Paths of a T1 TAP.7 is shown in Figure 5-11.

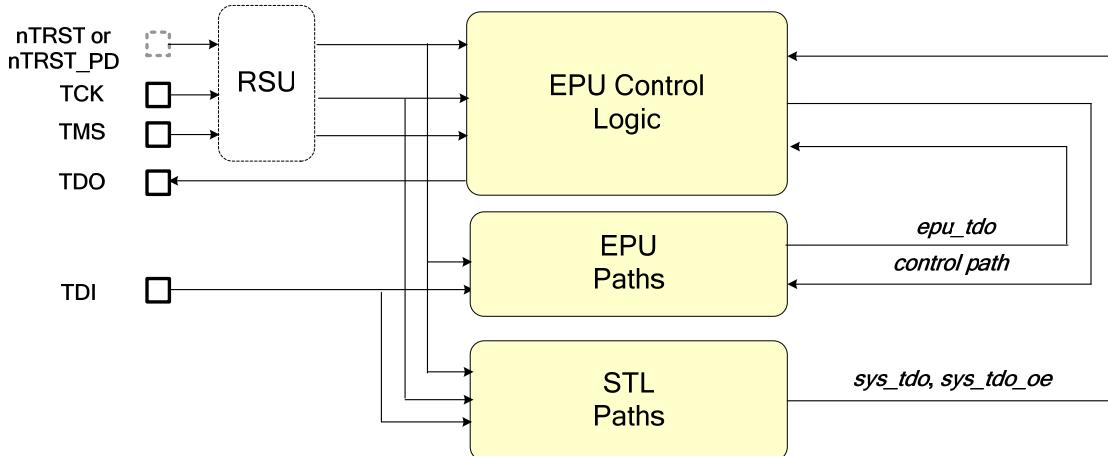
Hereafter, the term “System Path” is used to describe accessing resources with the STL. This occurs when the ZBS count is less than two or the EPU Operating State is a state other than *RUNNING*. The term “Control Path” is used to describe accessing resources within the EPU. Keep in mind that this does not imply the actual physical scan path being used.

Hereafter, the term “EPU Path” is used to describe a physical EPU Path. The term “STL Path” is used to describe a physical STL Path.



**Figure 5-11 — T1 TAP.7 physical and conceptual path relationships**

The operation of the TAP.7 Controller changes dramatically with the use of these two paths. The connectivity and management of the EPU and STL Paths is shown in Figure 5-12.



**Figure 5-12 — T1 TAP.7 Control and System Paths**

The Control Path is selected when the EPU Operating State is *RUNNING* and the ZBS count is greater than one. The System Path is used otherwise.

Scans affecting the EPU Paths may also affect the STL Scan Path. It is therefore important that that CLTAPC instruction be *BYPASS* or *IDCODE* because scans affecting the DR Scan Paths associated with these instructions have no meaningful effects. When using the Control Paths, the TDO data supplied by the STL Path is discarded. The EPU Control Logic detects the use of the Control Path and selects the *epu\_tdo* signal as the TAP's TDO data source. It selects the *sys\_tdo* signal as the TAP's TDO data source otherwise.

### 5.3.3.7.1 System Path

When the System Path is used:

- The STL supplies the TAP.7 Scan Path and controls the drive of TDO(C), provided the CLTAPC state is not parked.
- The EPU supplies the TAP.7 Scan Path (a one-bit bypass) when the CLTAPC state is parked.
- Functions supporting boundary scan in a Star Scan Topology are available.
- Scan path continuity is provided at all times.

### 5.3.3.7.2 Control Path

When the Control Path is used:

- The TDO(C) signal remains at a high-impedance level when the control level has not been locked.
- Control Levels Two through Seven may be created.
- Auxiliary Scan Paths may be accessed with Control Levels Four and Five.
- When commands are used, the TDO(C) signal remains at a high-impedance level except during *Shift-DR\_f* states within CR Scans.
- Scan path continuity is provided except for the time the TDO(C) signal remains at a high-impedance level when using the Command Control Level.

The Control Paths are described further in 9.7.

### 5.3.3.7.3 TDO Drive Policy

With a T1 TAP.7, the EPU controls the drive of the TDO(C) signal when it supplies the physical scan. The STL controls the drive of the TDO signal otherwise. Clause 13 provides a complete description of the TDO(C) Drive Policy.

### 5.3.3.8 EPU groups

Some EPU services are conditionally provided. These services are available when the value of the TAP.7 Controller's CGM Register bit is a logic 1. When this concept is expanded to multiple TAP.7 Controllers, the group of EPUs that:

- Provide conditional services is called the “Conditional Group”
- Do not provide conditional services is called the “Unconditional Group”

A Conditional Group Member provides access to both Global and Local Registers and executes the portion of the TAP.7 Controller command set that is subject to conditional execution. Hence, this group is named the Conditional Group. When a member of the Unconditional Group, the EPU provides access to Global Registers and executes the conditionally executable portion of the TAP.7 Controller command set as no operations. Conditional and Unconditional Group Membership is based solely on the Conditional Group Membership Register value for T1 and above TAP.7.

### 5.3.4 T1 TAP.7 high-level block diagram

A block diagram of a T1 TAP.7 with all options deployed is shown in Figure 5-13. It shows the addition of command processing, test and functional resets, and TAP.7 power management to the functionality of a T1 TAP.7.

## 5.4 T2 TAP.7

### 5.4.1 Overview

A conceptual view of a T2 TAP.7 is shown in Figure 5-14. The T2 TAP.7 is a superset of the T1 TAP.7. The T2 TAP.7 block diagram looks virtually identical to that of a T1 TAP.7, but there is one major difference. The T2 TAP.7 provides a “Super Bypass” function, a one-bit chip bypass for both IR and DR Scans. The DTS can use this function to make the STL either visible or invisible in a manner similar to the way the EMTAPCs are made visible and invisible with a T0 TAP.7.

The DTS may minimize the board scan-path length when there is no desire to communicate with any component within the STL and bypasses the STL when the Chip-Level Scan Path is not viable. An option is provided to start chip operation with the STL bypassed for “hot-connect protection.” When this feature is used, the physical connection of the DTS to a running system is unlikely to disturb system operation, as mentioned previously.

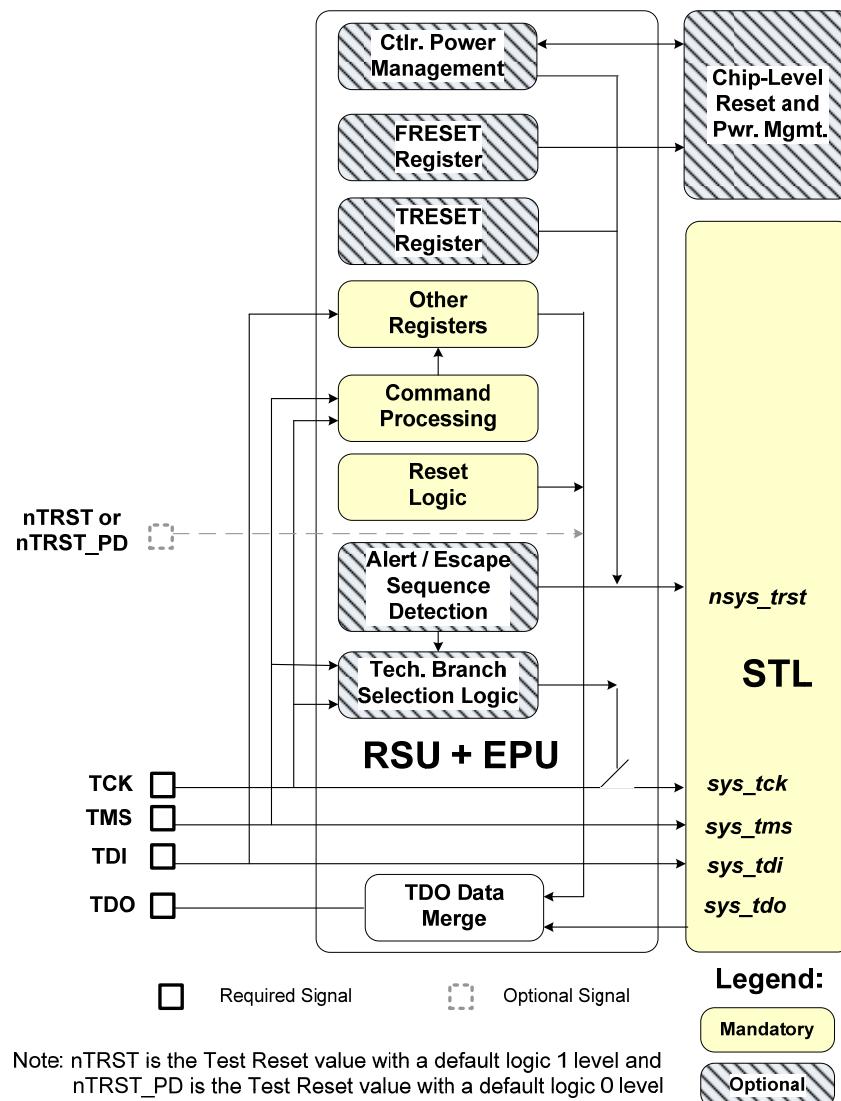


Figure 5-13 — High-level T1 TAP.7 functional block diagram

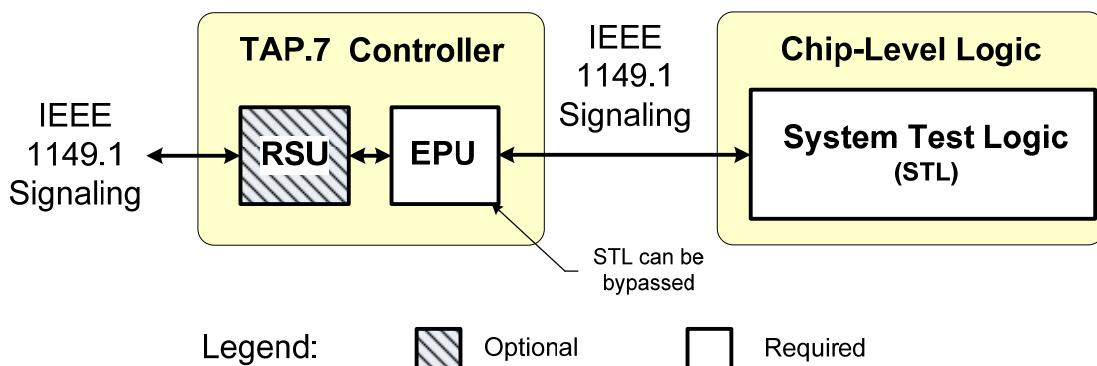


Figure 5-14 — T2 TAP.7 chip block diagram

## 5.4.2 Operating modes and capabilities

A T2 TAP.7 inherits the T0–T1 TAP.7's capabilities and options along with adding the JScan0–JScan3 described in 4.2.7.7. The mandatory aspects of T0 and T1 TAP.7s are also mandatory for a T2 TAP.7. The options available for T0–T1 TAP.7s are also options for a T2 TAP.7.

The T2 TAP.7 is the lowest TAP.7 Class supporting CLTAPC selection. The *Test-Logic-Reset* state selects or deselects the CLTAPC controller based on the startup option. Subsequently, the selection of the CLTAPC is updated when the *Run-Test-Idle* state is reached in a manner that is scan format dependent:

- JScan0              CLTAPC is selected
- JScan1              CLTAPC is deselected
- JScan2/JScan3      TAP.7 Controller is selected/deselected based on the value of the Scan Group Candidate (SGC) Register bit

The SGC Register value is delivered to TAP.7 Controller in a Series Scan Topology with the CR Scan of a three-part command. With Star Scan Topologies, this register value is delivered to TAP.7 Controllers one at a time using two-part commands that utilize TAP.7 Controller addressability when selecting or deselecting the CLTAPC attached to one TAP.7 Controller, or may be delivered to all TAP.7 Controllers with global command to simultaneously select or deselect the CLTAPC attached to each TAP.7 Controller.

As mentioned previously, the chip architect specifies the default scan format established by the *Test-Logic-Reset* state as either JScan0 or JScan1. This choice defines default chip behavior as either IEEE 1149.1-Specified or IEEE 1149.1-Non-disruptive Behavior. The latter provides hot-connect protection. Once the default scan format is established, subsequent selection changes take effect upon entry into the *Run-Test/Idle* state.

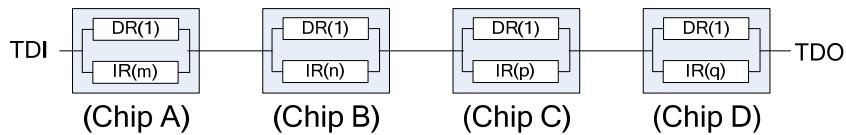
With IEEE 1149.1-Specified Behavior, the CLTAPC is selected with the STL supplying the scan path just as with a T0 TAP.7. With IEEE 1149.1-Non-disruptive Behavior, the STL is bypassed at startup with the TAP.7 Controller providing a one-bit scan IR and DR Scan Path.

An example of a Series Scan Topology providing a reduction in the lengths of the IR and DR Scan Paths with the Super Bypass function is shown in Figure 5-15. The scan-path length that may be created with the TAP.7 is compared to the scan-path length created by a traditional TAP.1 Series Scan Topology.

Information specifying the selection and deselection of the STL with the JScan2 Scan Format is stored in a SGC Register dedicated for this purpose. It is delivered to this Local Register using the serial scan chain of the Series Scan Topology. The DTS provides each TAP.7 Controller its own selection information with this mechanism.

## Series Scan Topology

### Scan-path lengths without chip bypass

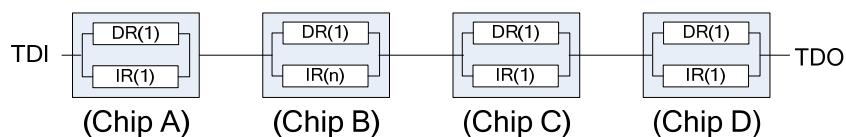


Instruction-Register scan path length =  $m + n + p + q$

Data-Register scan path length with *BYPASS* instruction =  $1 + 1 + 1 + 1$

## Series Scan Topology

### Scan-path lengths with bypass of chips A, C, and D



Instruction-Register scan path length =  $1 + n + 1 + 1$

Data-Register scan path length with *BYPASS* Instruction =  $1 + 1 + 1 + 1$

## Scan-path length savings

When A, C, and D are bypassed and B is selected:

IR path savings =  $m + p + q - 3$

DR path savings = 0

**Figure 5-15 — T2 TAP.7 scan-path length reduction**

The use of the Super Bypass function is likely to be greater for debug applications than for test applications with the rationale for this described as follows.

Debug operations generally access the scan paths associated with one CLTAPC within one device in the Series Scan Topology. Numerous debug operations occur sequentially before the CLTAPC of interest is changed. In some cases, the TAP.7 Controller of interest is never changed. More than one CLTAPC may also be selected, providing for synchronized debug actions that involve the STL of one or more TAP.7 Controllers.

Typical test operations access the scan paths of more than one device numerous times. Some or all of the devices in the scan path are included in these operations. When all of the devices in the scan path participate in the test operation, the TAP.7 bypass feature is not used and scan performance is not affected. An improvement in scan performance can be attained by using the TAP.7 bypass feature to bypass the devices in the scan path that do not participate in the test operation.

### 5.4.3 Operation

#### 5.4.3.1 Overview

With IEEE 1149.1-Specified Behavior at startup, the CLTAPC is selected and the STL provides the scan path. Note that with this behavior, the random signaling created by the physical connection of a DTS to the TAP.7 when the system is powered and operating may affect functional operation.

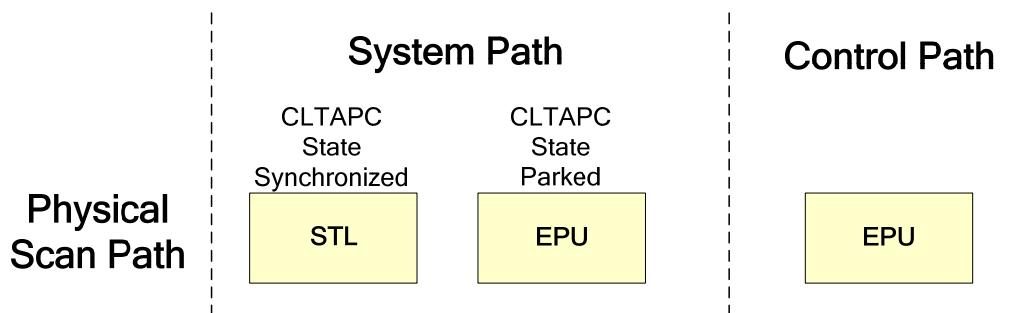
With IEEE 1149.1-Non-disruptive Behavior at startup, the CLTAPC is deselected. As a consequence of this, the EPU provides a one-bit IR and DR Scan Path as the STL cannot provide a viable scan path with the CLTAPC state parked. With this behavior, the random signaling created by the physical connection of a DTS to a TS that is powered and operating cannot easily affect functional operation as this random behavior would need to establish the Command Control level and create a two-part command. The TAPC state progressions needed for this are sufficiently long so as to make changing the system's behavior with random signaling improbable.

Subsequent to startup, the CLTAPC can be selected and deselected under DTS control. The selection and deselection of the CLTAPC is managed with the Scan Format and SGC Register values and occurs when the ADTAPC state reaches *Run-Test/Idle*. Deselection of the CLTAPC parks its state in the *Run-Test/Idle*. Although the CLTAPC is normally parked in the *Run-Test/Idle* state, a *Test-Logic-Reset* parking state can be created at startup as previously mentioned or by resetting the STL with the Test Reset (TRESET) Register while the CLTAPC is deselected.

Selection of the CLTAPC synchronizes the CLTAPC state to the TAP.7 Controller's TAPC *Run-Test/Idle* state. A one-state stay in the *Run-Test/Idle* state is required for synchronization when the CLTAPC parking state is known to be the *Run-Test/Idle* state. If the parked state is unknown or is known to be *Test-Logic-Reset*, a two-state stay in the *Run-Test/Idle* state is required to ensure proper synchronization. The stay in the *Run-Test/Idle* state may need to be increased to a maximum of four states when simultaneously selecting both the CLTAPC and EMTAPCs.

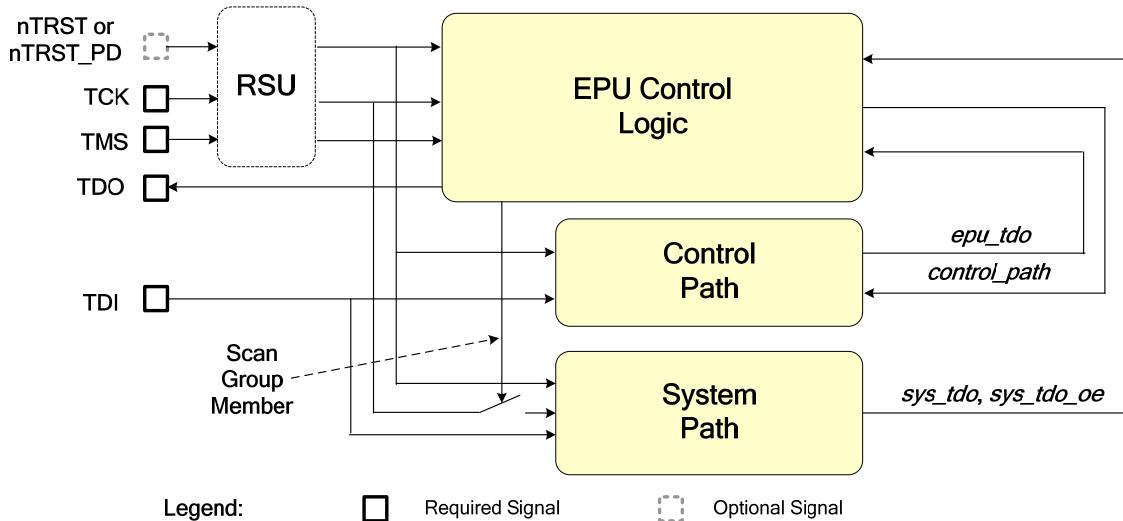
#### **5.4.3.2 Using the System and Control Paths**

The DTS' view of the T2 TAP.7's System and Control Scan Paths is similar to the view of these paths with the T1 TAP.7. The operation of the Control Path is the same with T1-T5 TAP.7s. However, there is a significant difference in the operation of the System Path. Recall that with an Online T0-T1 TAP.7 Controller, the CLTAPC operates continuously and the System Path is always operable. This is not the case with a T2 TAP.7 as the CLTAPC may be parked. With a T2 TAP.7, the operation of the System Path changes with the parking and resynchronization of the CLTAPC state. When the CLTAPC state is parked, the EPU provides a one-bit physical scan path for both IR and DR Scans, with the STL providing the physical scan path otherwise. This maintains scan-path continuity when the CLTAPC state is parked. This is shown in Figure 5-16.



**Figure 5-16 — T2 TAP.7 physical and conceptual path relationships**

The Control and System Path management is shown in Figure 5-17. Note that Scan Group Membership controls whether the CLTAPC state is parked or operating in lockstep with the ADTAPC.



**Figure 5-17 — T2 TAP.7 Control and System Paths**

### 5.4.3.3 TDO Drive Policy

The TDO Drive Policy for a T2 TAP.7 is similar to the TDO Drive Policy for a T1 TAP.7. The difference is the EPU supplies the physical scan path for the System Path when the CLTAPC state is an Idle Group Member, with the EPU controlling the drive of the TDO signal during this period. Clause 13 provides a complete description of the TDO(C) Drive Policy.

### 5.4.3.4 STL groups

STL services are provided only when the CLTAPC is selected and are unavailable when the CLTAPC is deselected. When this concept is expanded to multiple TAP.7 Controllers, STL Group Membership is used to identify STLs that operate in lockstep with the ADTAPC and those CLTAPCs that are parked in a specific state. It has two aspects, candidacy and membership, while EPU Group Membership has only one aspect, membership.

#### 5.4.3.4.1 STL group types

As described earlier, the CLTAPC state may be parked in one of the following states: *Test-Logic-Reset*, *Run-Test/Idle*, *Pause-IR*, and *Pause-DR*. When this concept is expanded to multiple TAP.7 Controllers, the group of STLs whose CLTAPC state is as follows:

- Parked in either the *Run-Test/Idle* or the *Test-Logic-Reset* states is called the “Idle Group”
- Parked in the *Pause-IR* state is called the “Pause-IR Group”
- Parked in the *Pause-DR* state is called the “Pause-DR Group”
- Operating in lock-step with its parent ADTAPC is called the “Scan Group”

With a T2 TAP.7, the STL may only be a member of the Idle Group or the Scan Group as its CLTAPC cannot be parked in either the *Pause-IR* or the *Pause-DR* state. With T3 and above TAP.7s, the CLTAPC may be a member of any of the four groups defined above. The description of the *Pause-IR* and *Pause-DR* Groups is part of the description of a T3 TAP.7 (see 5.5.3.3).

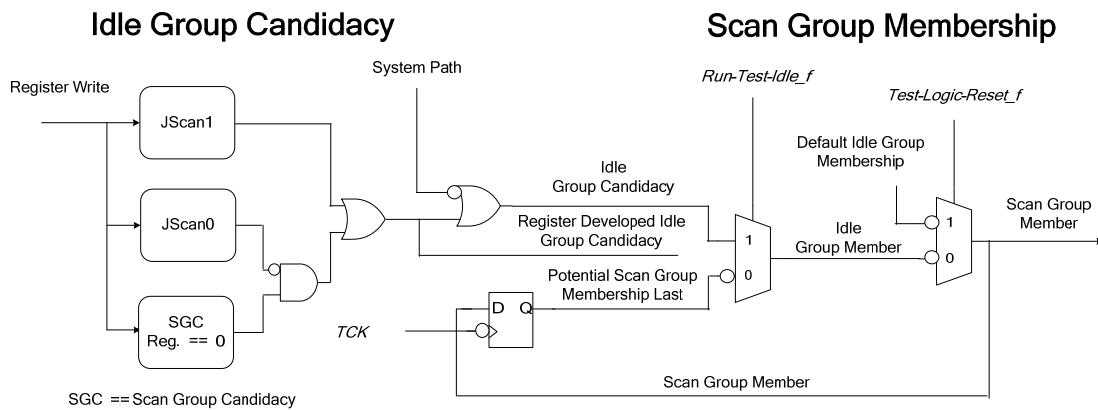
#### 5.4.3.4.2 STL Group Membership changes

STL Group Membership is established by the selection and deselection of the CLTAPC. Startup options permit the placement of the STL in either the Idle Group or the Scan Group with the *Test-Logic-Reset* state. Subsequent to this initialization, changes in group membership between the Idle Group and the Scan Group occur when the *Run-Test/Idle* state is reached, with the STL becoming a member of the Idle Group when it is an Idle Group Candidate and becoming a member of the Scan Group otherwise.

The STL is forced to be an Idle Group Candidate when using the Control Path. When the System Path is used, the STL is an Idle Group Candidate when using:

- The JScan1 Scan Format
- A scan format other than JScan0 or JScan1 and the Scan Group Candidate Register value is a logic 0

The STL is a Scan Group Candidate otherwise. The relationship of Idle and Scan Group Candidacy and Idle and Scan Group Membership is shown in Figure 5-18.



**Figure 5-18 — T2 TAP.7 group membership determination**

The concept of Scan Group Candidacy and membership may also be extended to T0 and T1 TAP.7s. With these TAP.7s, the STL is always a Scan Group Member.

#### 5.4.4 T2 TAP.7 high-level block diagram

A block diagram of a T2 TAP.7 with all options deployed is shown in Figure 5-19. It shows the addition of CLTAPC selection and deselection to the functionality of a T1 TAP.7.

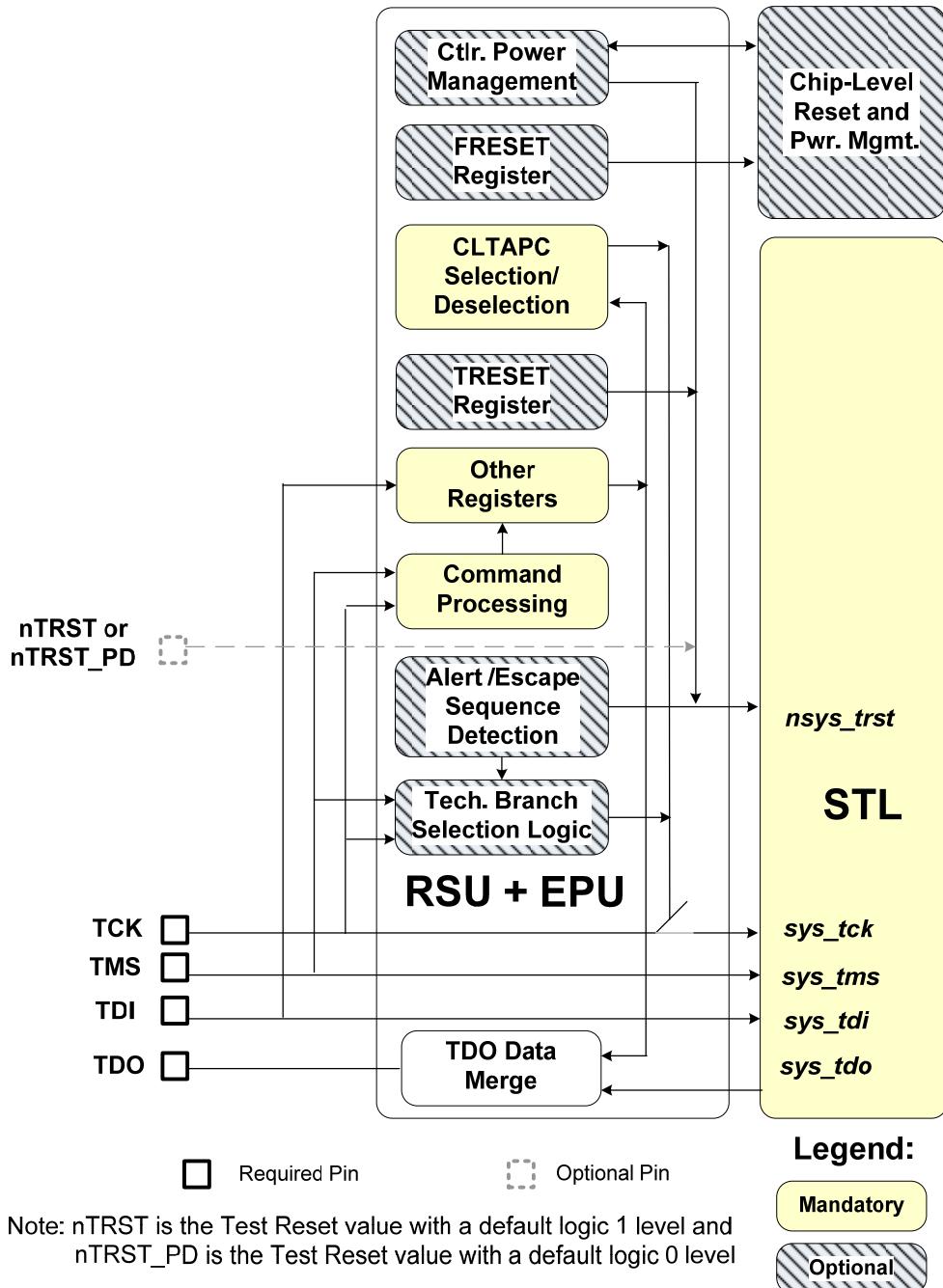


Figure 5-19 — High-level T2 TAP.7 functional block diagram

## 5.5 T3 TAP.7

### 5.5.1 Overview

A conceptual view of a T3 TAP.7 is shown in Figure 5-20. The T3 TAP.7 is a superset of the T2 TAP.7. On the surface, the T3 TAP.7 block diagram looks identical to that of a T2 TAP.7, but there is a major difference. A T3 TAP.7 adds support for operation within a Four-Wire Star (Star-4) Scan Topology.

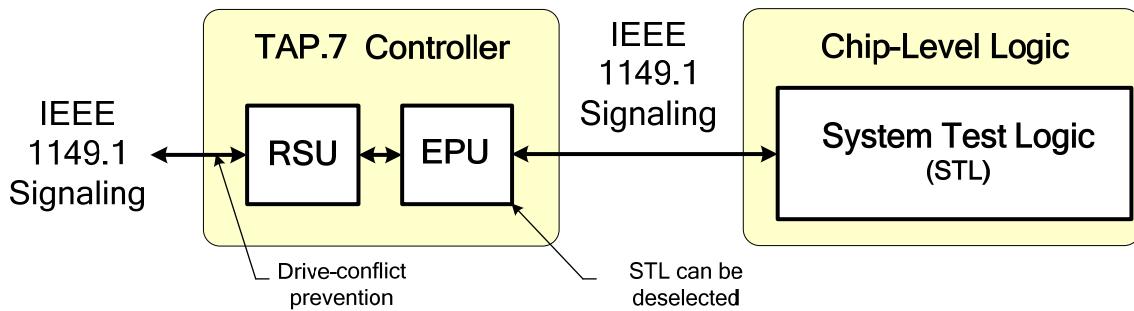


Figure 5-20 — T3 TAP.7 chip block diagram

### 5.5.2 Operating modes and capabilities

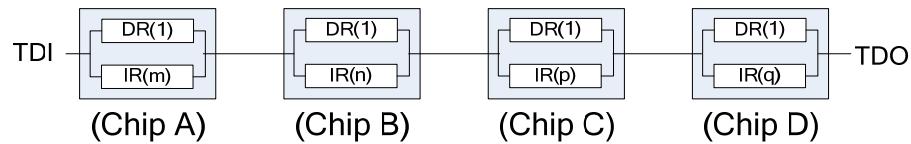
The T3 TAP.7 inherits the mandatory T2 TAP.7 functionality with the T0–T2 TAP.7 options other than the RSU being optional. The RSU becomes mandatory with a T3 and above TAP.7. The Star-4 Scan Topology supported by a T3 TAP.7 creates the shortest possible scan-chain length from the DTS to the TAP.7 of interest. This scan topology is attractive for systems where the equipment configuration varies (e.g., cards are added or removed from a system). The function of the JScan3 Scan Format is modified from that of a T2 TAP.7 to support this capability.

Creating a Star Scan Topology with IEEE Std 1149.1 requires external logic. With IEEE Std 1149.1, a DTS interface wider than four signals is needed to select a TAP of interest. Either the TCK or TMS signal of each device is gated to create a TAP selection mechanism. This is equivalent to a TAP.7 Controller Address. With this standard, the impediments to the use of a Star Scan Topology have been eliminated as T3 and above TAP.7s contain the TAP selection mechanism.

An example of a Star Scan Topology providing a reduction in the lengths of the IR and DR Scan Paths is shown in Figure 5-21. Its scan-path length is compared to the scan-path length created by a traditional TAP.1 Series Scan Topology.

## Series Scan Topology

Data Register scan-path length shown with the *BYPASS* Instruction

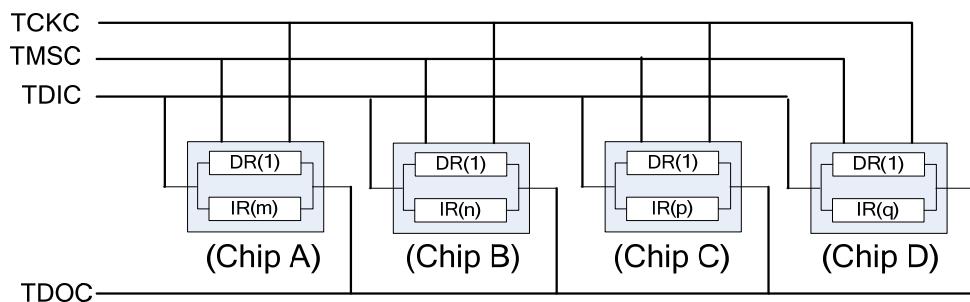


### Series scan-path length comparison using decoupling:

Scan-path length	IR-path length	DR-path length
STLs Coupled (A, B, C, D)	$m + n + p + q$	$1 + 1 + 1 + 1$
STLs Coupled (C)	$1 + 1 + p + 1$	$1 + 1 + 1 + 1$
Scan-path length Savings	$m + n + q - 3$	0

## Star-4 Scan Topology

Data Register scan-path length shown with the *BYPASS* Instruction



### Star-4 vs. Series scan-path length comparison:

Scan-path length	IR-path length	DR-path length
Series -STLs Coupled (A, B, C, D)	$m + n + p + q$	$1 + 1 + 1 + 1$
Star-4 - STLs Coupled (C)	$p$	1
Scan-path length Savings	$m + n + q$	3

Figure 5-21 — T3 TAP.7 scan-path length reduction

### 5.5.3 Operation

#### 5.5.3.1 Within series and star scan topologies

Since the T3 TAP.7 inherits the T2 TAP.7 mandatory capabilities, it operates in a Series Scan Topology with the JScan0–JScan2 Scan Formats as if it was a T2 TAP.7. The operation of the JScan3 Scan Format is different from a T2 TAP.7 as it enables the following TAP.7 Controller features supporting operation in a Star-4 Scan Topology:

- Direct addressability
- Prevention of drive conflicts, as the TAP TDI and TDOC signals are connected in parallel in a Star-4 Scan Topology
- Creation of Series-Equivalent Scans in a Star-4 Scan Topology

#### 5.5.3.2 T3 TAP.7 Controller addressability in a Star-4 Scan Topology

With a Series Scan Topology, the position of a TAP.7 Controller on the series scan chain, relative to the DTS, provides TAP.1 and TAP.7 addressability. When a Star Scan Topology is used, the addressability afforded by the Series Scan Topology disappears as the DTS sees each TAP.7 Controller in parallel, and therefore having the same relative position on the scan chain. An alternate means of addressability is therefore needed for a Star Scan Topology.

A T3 TAP.7 Controller becomes directly addressable when the JScan3 Scan Format is used. A TAP.7 Controller Address (TCA) is created from 27 bits of the IEEE 1149.1 DEVICE\_ID and an eight-bit NODE\_ID. The NODE\_ID is used to identify unique instances of chips where the same DEVICE\_ID is used. The NODE\_ID supports the use of up to 256 instances of the same device with a single DTS connection. The method used to create the NODE\_ID (if one is needed) is a chip-level function and is not covered by this specification. The TCA is not used with other JScan Scan Formats as these scan formats specify TAP.7 operation within a Series Scan Topology.

It is expected that the majority of applications have a small number of TAPs, in which case the use of the 35-bit TCA would appear as a noticeable overhead for commands used with the Star-4 Scan Topology. Consequently, a TAP.7 command is provided to alias the TCA to a four-bit CID to avoid this issue. Another TAP.7 command is provided to remove a single alias or all aliases. The DTS can use these commands to assign CIDs and move the CIDs from one TAP.7 Controller to another as needed. The 16 CIDs may be used with more than 16 TAP.7s, with the CIDs dynamically allocated to a number of TAP.7 Controllers. The TDOC signal operates as both an input and an output with high impedance, totem pole, and open-drain drive characteristics to support CID allocation.

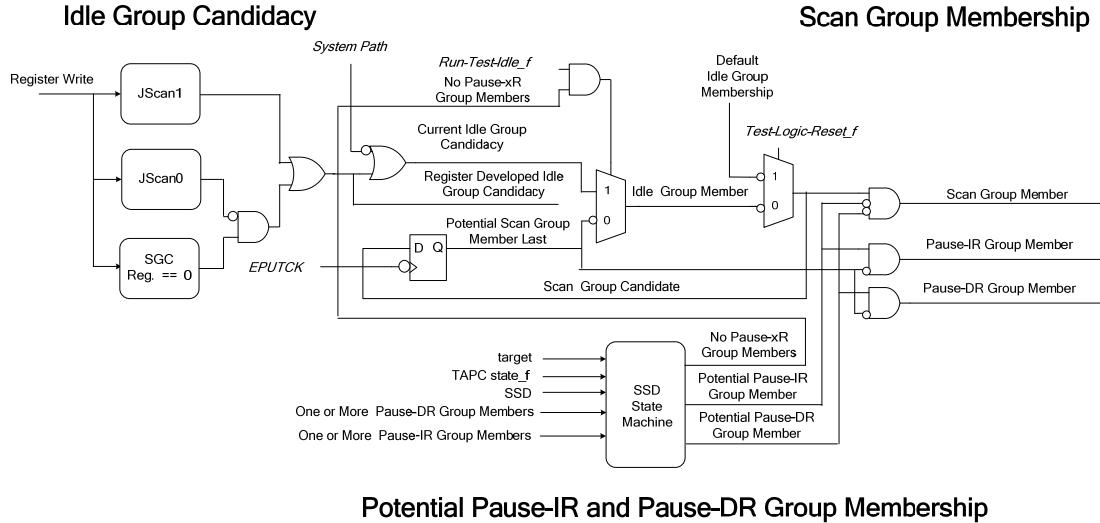
#### 5.5.3.3 Pause-IR and Pause-DR STL groups

T3 and above TAP.7s provide for the parking of the CLTAPC state in both the *Pause-IR* and the *Pause-DR* states to support operation in a Star Scan Topology. With this scan topology, the STL may be a member of any of the four STL groups.

Scan and Idle Group Candidacy determines Scan and Idle Group Membership when the STL is already a Scan or Idle Group Member, and is ignored when the STL is either a Pause-IR or a Pause-DR Group Member.

Scan Selection Directives (see 5.5.3.5) move Group Membership between Scan and *Pause-xR* and vice versa. The actions created by SSDs in the *Pause-xR* state control whether a CLTAPC's membership moves between the Scan Group and the *Pause-xR* Group. With one or more Pause-DR Group Members, the creation of Pause-IR Group Members is inhibited and Idle Group Membership cannot be changed. With one or more Pause-IR Group Members, the creation of Pause-DR Group Members is inhibited and Idle

Group Membership cannot be changed. Pause-xR Group Membership interlocks SSDs to create these conditions. This is shown in Figure 5-22.



**Figure 5-22 — T3 TAP.7 group membership related to the use of the System Path**

#### 5.5.3.4 Series/star scan equivalency

##### 5.5.3.4.1 Defining scan equivalency

Functional equivalency of series and star scan operations is the key to the utilization of the functionality afforded by the IEEE Std 1149.1 and IEEE Std 1532 [B4].<sup>8</sup> Scan operations in a Star and Series Scan Topology are functionally equivalent when:

- The operations performed by the *Capture-xR* and *Update-xR* states remain synchronized across a group of STLs of interest.
- The IR and DR Scan data associated with this group of STLs is exchanged between the *Capture-xR* and *Update-xR* states.

The T3 TAP.7 features are expanded beyond those available with a T2 TAP.7 to support this Star Scan Topology requirement.

##### 5.5.3.4.2 Creating a Series-Equivalent Scan within a Star Scan Topology

The operations performed by the *Capture-xR* and *Update-xR* states in a Series Scan Topology are duplicated in a Star Scan Topology, with the number of EPU Test Clock periods between the *Update-xR* and *Capture-xR* states remaining the same. However, the scan operation that exchanges data with the TAPs of interest between the *Capture-xR* and *Update-xR* states is different. The single scan operation targeting several devices that is used with a Series Scan Topology is mimicked using a series of scan operations with a Star Scan Topology. Each operation in this series of operations targets a single and different TAP.7 of interest.

In a Star Scan Topology, the STLs of interest are placed in the Scan Group, whereas the STLs not of interest are placed in the Idle Group. The Idle Group's CLTAPC is parked in *Run-Test/Idle*, whereas the

---

<sup>8</sup> The numbers in brackets correspond to those of the bibliography in Annex I.

Scan Group exits the *Run-Test/Idle* state to the *Select-DR* state. A Series-Equivalent Scan scan is performed with the Scan Group. The terms “Pause-IR Group” and “Pause-DR Group” become significant shortly.

The state of the CLTAPCs of the Scan Group move through the *Capture-xR* and *Update-xR* states in lockstep. Between these two states, the shift portion of a scan is divided into sections where one STL that is a Scan Group Member participates in a scan operation while the other STLs do not participate, as their CLTAPC state is parked. The CLTAPC parking state is the *Pause-xR* state when performing an IR Scan and is the *Pause-DR* state when performing a DR Scan. Beginning with the *Select-xR-Scan* state, the TAPC state of all Scan Group Members is moved to the *Pause-xR* state without traversing the *Shift-xR* state. From this point, the scan path managed by the CLTAPC in the Scan Group are sequentially scanned one at a time by dividing the scan transfer into sections, with one section for each member of the Scan Group.

One section of the scan transfer is performed by parking the CLTAPC state for all but one Scan Group Member, leaving only one Scan Group Member, with the former members of the Scan Group placed in the *Pause-xR* Group. Scan information is exchanged with this sole Scan Group Member using the *Shift-xR* TAP controller state. The CLTAPCs of other TAP.7 Controllers in the Scan Group remain parked in the *Pause-xR* state, as these CLTAPCs are *Pause-xR* Group Members. A selection mechanism called the SSD is added to provide a means to park and synchronize the CLTAPC state in the *Pause-xR* states. SSDs are described in detail in 20.10 following the description of factors that enable their use.

When the scan exchanges with all STLs are completed, all members of the *Pause-xR* Group are resynchronized to the operation of the ADTAPC. At this point, the Scan Group Membership is the same as it was when the *Run-Test/Idle* state was exited. There are no TAP.7 Controllers with a CLTAPC in either the *Pause-IR* Group or the *Pause-DR* Group. The scan is completed by moving the TAP controller state from *Pause-xR* to *Update-xR* without traversing the *Shift-xR* TAP controller state. This implements a Series-Equivalent Scan as shown in Figure 5-23. The use of SSDs is compatible with the selection mechanism managed with commands first introduced with a T2 TAP.7.

### 5.5.3.5 Scan Selection Directives

#### 5.5.3.5.1 Enabling the use of SSDs

The use of SSDs is disabled following the *Test-Logic-Reset* state until both of the following occur:

- Their use is enabled with the TAP.7 register (SSDE) used for this purpose.
- The scan format supports operation with a Star Scan Topology.

The use of SSDs is also subject to the use of the System Path.

#### 5.5.3.5.2 Types of SSDs

Scan Selection Directives are added to the TAP.7 Controller functionality to provide a means to park the CLTAPC state in the *Pause-xR* or *Run-Test/Idle* state. This is the key to creating series and star scan equivalency. SSDs provide a means to simultaneously select and deselect one or more CLTAPCs in the *Pause-DR*, *Pause-IR*, and *Run-Test/Idle* states using TDI data. The three types of SSDs are described as follows:

- **Deselect-All** Deselection of the CLTAPC: SSD\_DA (Deselect all)
- **Select-All** Selection of the CLTAPC: SSD\_SA (Select all)
- **Select-One** Select-One: SSD\_SOT (using TCA) or SSD\_SOC (using CID)

When used with:

- *Run-Test/Idle*—Selection of the CLTAPC of the TAP.7 Controller when the CID or TCA delivered with the SSD matches the TAP.7 Controller CID or TCA

- *Pause-DR* or *Pause-IR*—Selection of the CLTAPC of the TAP.7 Controller when the CID or TCA delivered with the SSD matches the TAP.7 Controller CID or TCA and deselection of the CLTAPC when the CID or TCA delivered with the SSD does not match the TAP.7 Controller CID or TCA

Henceforth, in this and subsequent clauses, references to an SSD may be made using its proper name (SSD\_xxx or SSD\_xxx) or by its function, **Select All**, **Select One**, or **Deselect All**.

### 5.5.3.5.3 SSD execution

The TAP.7 Controller hardware imposes three constraints on SSD operation to simplify the hardware bookkeeping needed to manage the prevention of drive conflicts. These rules are listed as follows:

- The members of the Idle Group can only be changed when there are no members of either of Pause-DR Group or Pause-IR Group and the ADTAPC state is *Run-Test/Idle*.
- The members of the Pause-IR Group can only be changed when there are no members of the Pause-DR Group and the ADTAPC state is *Pause-IR*.
- The members of the Pause-DR Group can only be changed when there are no members of the Pause-IR Group and the ADTAPC state is *Pause-DR*.

An SSD performs its operation only when these constraints are met, and an SSD performs no operation otherwise. This is called conditional execution. SSD execution is based on the number of associated members whether there are *Pause-IR* and *Pause-DR* members. SSDs associated with the:

- *Pause-DR* state execute provided there are no Pause-IR Group Members
- *Pause-IR* state execute provided there are no Pause-DR Group Members
- *Run-Test/Idle* state execute provided there are no Pause-IR Group Members or Pause-DR Group Members

### 5.5.3.5.4 SSD State Machine

The last executed SSD is recorded with a function called the SSD State Machine. The state of this machine affects the group membership. There are seven SSD states. These states identify the following:

- *SSD\_SA* (Select-All-any state)—All STLs that are not members of the Idle Group are Scan Group Members. The last executed SSD that was associated with either the *Pause-DR* or *Pause-IR* states was an *SSD\_SA* (Select\_All).
- *SSD\_IS\_IR* (Is Selected -IR)—The STL is a Scan Group Member, provided it is not an Idle Group Member. The STLS of other Online TAP.7 Controllers are not a member of the Scan Group. The last SSD executed was an *SSD\_SOC* or *SSD\_SOT* associated with the *Pause-IR* state and targetting the ADTAPC.
- *SSD\_IS\_DR* (Is Selected-DR)—The STL is a Scan Group Member, provided it is not an Idle Group Member. The STLS of other Online TAP.7 Controllers are not a member of the Scan Group. The last SSD executed was an *SSD\_SOC* or *SSD\_SOT* associated with the *Pause-DR* state and targetting the ADTAPC.
- *SSD\_NS\_IR* (Not Selected-IR)—The STL is a Pause-IR Group Member, provided it is not an Idle Group Member. The STL of another Online TAP.7 Controller may be a member of the Scan Group. The last SSD executed was an *SSD\_SOC* or *SSD\_SOT* associated with the *Pause-IR* state and not targeting the ADTAPC.
- *SSD\_NS\_DR* (Not Selected-DR)—The STL is a Pause-DR Group Member, provided it is not an Idle Group Member. The STL of another Online TAP.7 Controller may be a

- member of the Scan Group. The last SSD executed was an SSD\_SOC or SSD\_SOT associated with the *Pause-DR* state and not targeting the ADTAPC.
- *SSD\_DA\_IR* (Deselect-All-IR)—The STL is a Pause-IR Group Member, provided it is not an Idle Group Member. The STLs of other Online TAP.7 Controllers are not a member of the Scan Group. The last SSD executed was an SSD\_DA associated with the *Pause-IR* state.
- *SSD\_DA\_DR* (Deselect-All-DR)—The STL is a Pause-DR Group Member, provided it is not an Idle Group Member. The STLs of other Online TAP.7 Controllers are not a member of the Scan Group. The last SSD executed was an SSD\_DA associated with the *Pause-DR* state.

The state of this machine is delayed one EPU Test Clock period to determine the TDOC Drive Policy, with the delayed version of this state used in Table 13-6 and Table 13-8.

### 5.5.3.6 Series-Equivalent Scan creation

Series-Equivalent Scans are created as follows:

- Using command(s) or SSD(s) to choose the STLs that do not participate in a Series-Equivalent Scan by making them Idle Group Members (the remaining STLs are Scan Group Members) using commands or SSDs
- Moving the CLTAPC state of the STLs that remain in the Scan Group to the *Pause-xR* state
- Using an SSD to identify an STL that is the target of a scan by making other STLs that are not an Idle Group Member a member of the *Pause-xR* Group (this makes the STL of the TAP.7 Controller identified by the SSD the only member of the Scan Group, provided it exists)
- Repeating the above process for each STL that is not a member of the Idle Group
- Using an SSD vacating all *Pause-xR* Group Membership (this creates Scan Group Membership for all STLs that are not Idle Group Members (there are only Scan Group and Idle Group Members after this operation))

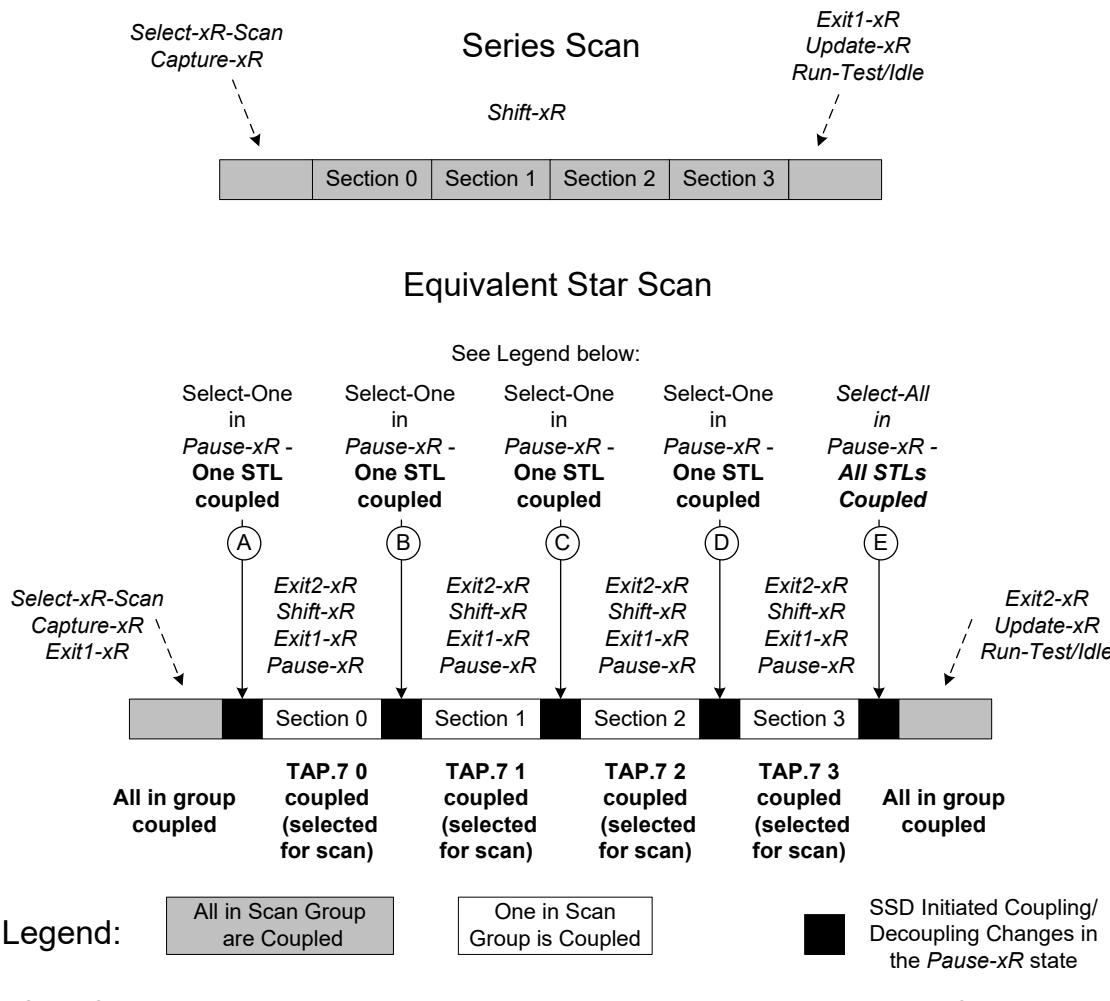
An example of a scan in a Star Scan Topology that provides the equivalent function of a scan in a Series Scan Topology is shown in Figure 5-23. This example has four T3 TAP.7s connected in a Star-4 Scan Topology. It begins with the:

- Use of SSDs enabled via the TAP.7 Controller register
- Interlocking deactivated
- *Select-DR-Scan* state
- TAP.7 ZBS count less than or equal to one
- The CLTAPCs of all TAP.7 Controllers (0–3) selected

In this figure, the ADTAPC state of each TAP.7 Controller is moved to point A through the *Capture-xR* state to the *Pause-xR* state without an intervening *Shift-xR* state. A Select-One SSD selects the CLTAPC of a single TAP.7, whereas deselection of the CLTAPC of other TAP.7 Controllers occurs at points A, B, C, and D. The CLTAPC of a different TAP.7 is selected at each of these points with other TAPCs within the Scan Group deselected.

Within sections 0–3, scan data is exchanged with the scan paths managed by the CLTAPC. When the exchange of scan data with a selected CLTAPC is completed, its state is returned to *Pause-xR*. When point E is reached, all CLTAPCs are selected with a Select-All SSD. The scan is then completed by moving the CLTAPC state through the *Update-DR* state. This process is used for both IR and DR Scans.

The TAPC state progression is limited to *Exit2-xR*, *Shift-xR*, *Exit1-xR*, and *Pause-xR* states during the scan data exchange so as to not cause an *Update-xR* state. Since only one CLTAPC reaches the *Shift-DR* state at any one point in time, there is no drive conflict at the TDOC signal and data is exchanged with the STL whose CLTAPC is selected.



**Figure 5-23 — Series-Equivalent Scan using SSDs in a Star Scan Topology**

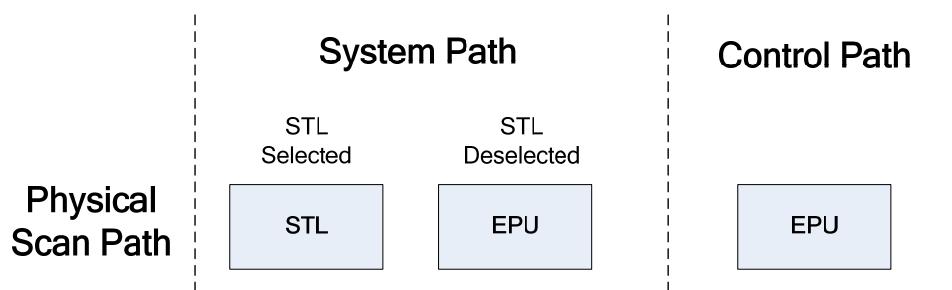
#### 5.5.3.6.1 Exclusivity of SSD and TAP.7 Controller commands

The *Test-Logic-Reset* state disables the use of SSDs. Their subsequent use may be enabled with the SSDE Register. SSDs are ignored when the scan format does not support the characteristics required for SSD operation and are only available for use when the System Path is used. This makes the use of commands and SSDs mutually exclusive. Making their use mutually exclusive reduces the validation space and permits smaller implementations. In addition there is no need for the concurrent use of TAP.7 commands and SSDs as boundary-scan testing is a function associated with the System Path and commands are a function associated with the Control Path.

A brief summary of these interactions is provided hereafter. SSD detection is inhibited when the EPU Operating State is *RUNNING* and the ZBS count is greater than or equal to two. This occurs in anticipation of establishing the Command Control Level. Prior to this point, the TAP.7 ZBS count is set to zero by the successful completion of an SSD when the control level is not locked and the TAP.7 ZBS count is less than or equal to one. The wording “successful completion of an SSD” is used because SSDs are terminated when their supporting state is exited prior to their completion. With these criteria, the ZBS count cannot reach two when at least one SSD is used with every DR Scan.

### 5.5.3.7 Using the System and Control Paths

The DTS’ view of the T2 TAP.7 is one of using either the STL or the EPU using System and Control Scan Paths just as with the T1 TAP.7. There are two significant differences, however. The EPU provides a one-bit physical scan path for both IR and DR Scans when the CLTAPC state is parked and the System Path is used as shown in Figure 5-24.

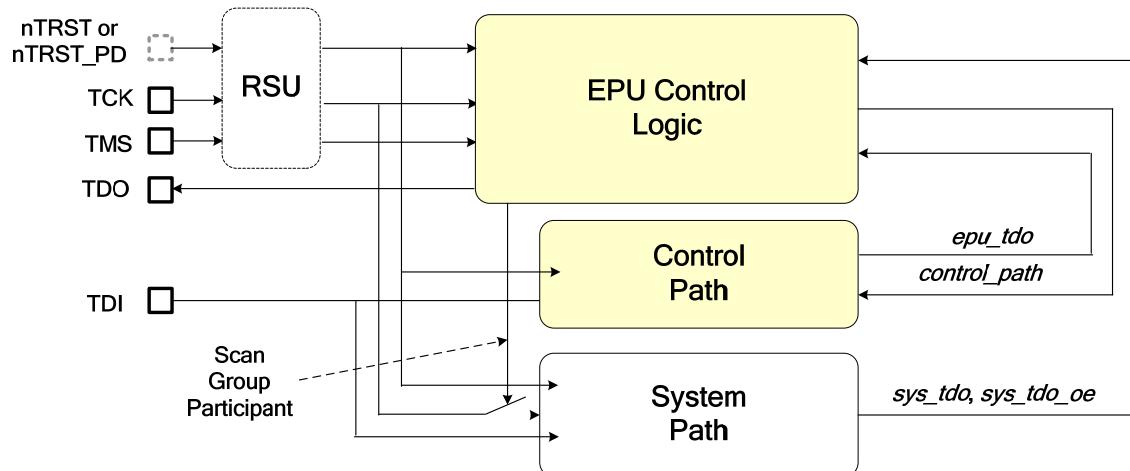


**Figure 5-24 — T2 TAP.7 physical and conceptual path relationships**

Clause 13 details the TDOC Drive Policy for using the System and Control Paths. The TDOC Drive Policy for using the System Path is determined by:

- Scan Group Membership and the number of Scan Group Members
- A ZBS count greater than or equal to two when the EPU operating state is *RUNNING*

Also, recall that with an Online T1 TAP.7 Controller, the CLTAPC operates continuously with the use of the System Path determined by the control level. With a T2 and above TAP.7s, the operation of the System Path based on whether the CLTAPC state is parked. The TAP scan-path continuity is maintained when the CLTAPC state is parked as the EPU provides the physical scan path. The operation of the Control Path remains the same with a T1 TAP.7. This is shown in Figure 5-25.



**Figure 5-25 — T2 TAP.7 Control and System Paths**

### 5.5.3.8 TDO Drive Policy

When operating with the JScan0–JScan2 Scan Formats, the TDO(C) Drive Policy is the same as for a T2 TAP.7. When operating with the JScan3 Scan Format, the TAP.7 Controller manages the drive of the TDOC signal to prevent drive conflicts based on the TDO(C) Drive Policies described in Clause 13.

When using the System Path, the number of Scan Group Members is derived by monitoring the TAP.7 Controller command and SSD history beginning from Type-0-Type-4 Resets (see 10.2 for a description of reset types). A Scan Group Member is allowed to drive the TDOC signal when the TAP.7 Controller determines its STL is the only Scan Group Member. When using the Control Path, the TDOC Drive Policy is determined by tracking the command history beginning from Type-0-Type-4 Resets to determine the number of Conditional Group Members. A Conditional Group Member is allowed to drive the TDOC signal when the TAP.7 Controller determines its EPU is the only Conditional Group Member.

### 5.5.4 T3 TAP.7 high-level block diagram

A block diagram of a T3 TAP.7 with all options deployed is shown in Figure 5-26. It shows the addition of addressability and SSDs providing Series-Equivalent Scans.

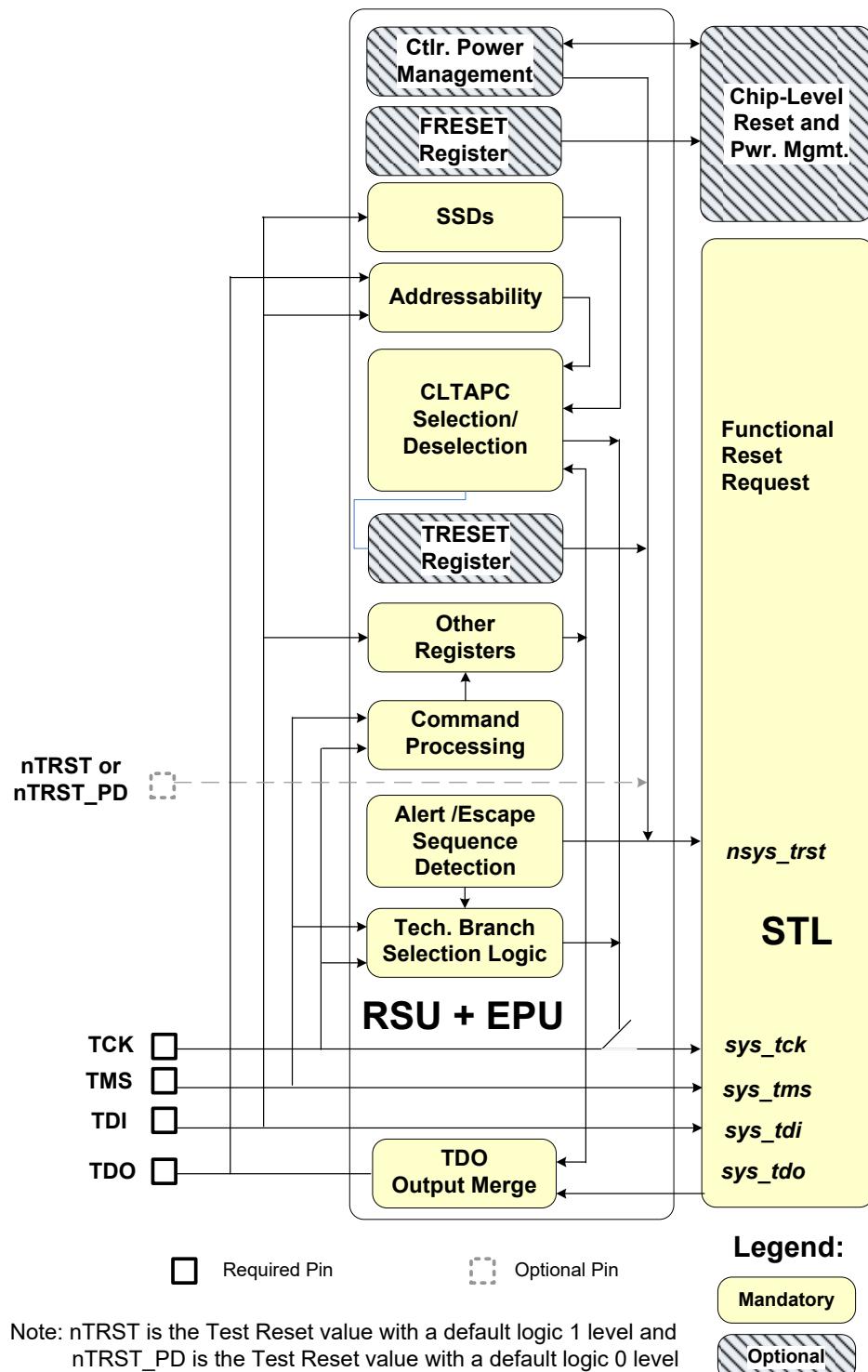


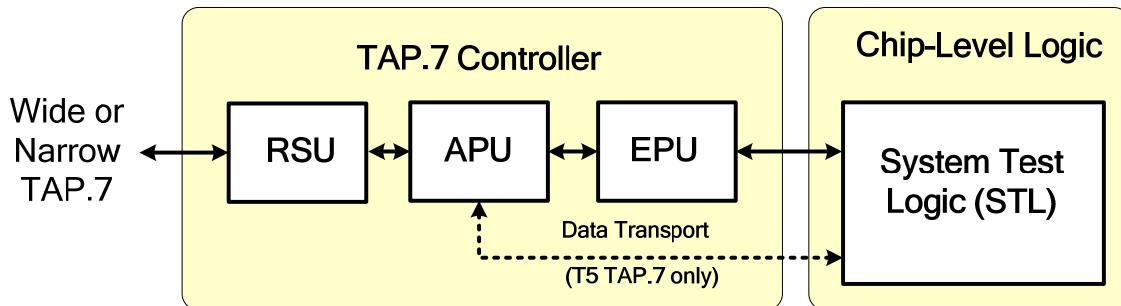
Figure 5-26 — High-level T3 TAP.7 functional block diagram

## 6. T4–T5 TAP.7 operational overview

### 6.1 Introduction

This clause provides an overview of the T4–T5 TAP.7s. The Advanced capabilities are added to a T3 TAP.7 to create these TAP.7s. They may be implemented in narrow (two-signal) or wide (four-signal) configurations as the TDIC and TDOC signals become optional. The narrow version of a T4 TAP.7 deployed in a Star-2 Scan Topology delivers the functionality of a Star-4 Scan Topology using five-signal (four signals + optional Test Reset signal) T3 TAP.7. The wide version allows operation in the Series, Star-4, and Star-2 Scan Topologies. A T5 TAP.7 adds data transport capability to the T4 TAP.7 capability. It is important to note that this capability makes the TMSC signal available to other protocols as a wire and does not define the protocols utilizing the wire.

The Advanced Protocol Unit (APU) is placed between the EPU and the RSU to create the T4 and above TAP.7 Controllers as shown in Figure 6-1. The APU delivers advanced capabilities with a number of scan formats that utilize both the Standard Protocol and the Advanced Protocol in both the narrow and wide T4 TAP.7 versions.



**Figure 6-1 — T4/T5 TAP.7 chip block diagram**

When using the Advanced Protocol, the APU performs scan transfers with the TCKC and TMSC signals by serializing the information related to a TAPC and the scan paths it supports. It multiplexes the use of the TMSC signal for T4 and above TAP.7 Controller functions. The serialization logic is bypassed to operate with the Standard Protocol. This provides full T3 TAP.7 capability using the Standard Protocol when the TDI and TDO signal functions are available and the ability to manage the TAP.7 Controller with no STL data transfers otherwise.

With a wide TAP.7, the chip architect may choose to implement the TDIC and TDOC signal functions as fixed functionality (the T3 TAP.7 TDIC and TDOC signal functions) or as switchable between the T3 TAP.7 TDIC and TDOC signal functions and an auxiliary function, with the signal function specified with a TAP.7 Controller register. When the pin function is programmable, the default signal function may be either of the programmable choices. Programmable TDIC and TDOC functionality provides a means to fully utilize these signals in any scan topology.

The entire EPU infrastructure (i.e., ZBS detection, control-level management, and command generation) is utilized when either the Standard Protocol or the Advanced Protocol is used, as these TAP.7 Controller attributes are controlled entirely from the TAPC state progression. The EPU is unaware of the APU's existence.

The subject matter within this clause is organized in the following manner:

- 6.2 T4 TAP.7

- 6.3 T5 TAP.7
- 6.4 TAP.7 feature summary

## 6.2 T4 TAP.7

### 6.2.1 Operating modes and capabilities

The T4 TAP.7 Controller delivers both the flexibility to work with any test or debug application and the scan performance needed for these applications. It utilizes a number of different optimization and serialization schemes to balance scan flexibility and scan performance with the Advanced Protocol. The TAP.7 serialization schemes are specified with the MScan, OScan, and SScan Scan Formats. Collectively, these scan formats are called Advanced Scan Formats. These scan formats complement the JScan0–JScan3 Scan Formats inherited from the T3 TAP.7. The DTS chooses a scan format that matches the constraints imposed by the application, chip components, and possibly the DTS itself.

The Advanced Scan Formats provide a number of options for the serialization of scan information, providing various tradeoffs between performance and flexibility. These tradeoffs are beneficial as the information transferred per TAPC state varies for different test and debug applications. Also, with some test and debug applications, all TAPC states do not require the same amount of information be transferred (for example, TDI and TDO in non-*Shift-xR* TAPC states). This is especially true with debug applications where specialized use of the TAP is more likely.

The DTS selects the Advanced Scan Format based on the right balance of performance and flexibility for the application using the TAP. Scan formats emphasizing performance do not exchange unneeded information. Scan formats emphasizing flexibility exchange all information related to a TAPC state and provide the System Test Logic a means to stall the TAPC state progression. These tradeoffs change the bit sequences seen at the TAP.7 signals. TAP.7 Controller TAPC state rates comparable with those achievable with a TAP.1 controller can be achieved by using:

- Full-cycle timing between the DTS and the TAP.7 is utilized to allow operation at higher TCKC frequencies.
- A scan format minimizing the number of bits transferred to advance a TAPC state is chosen.

The MScan and OScan Scan Formats address a variety of test and debug use cases. The SScan Scan formats emphasize performance for specific debug applications. The mandatory MScan and OScan0–OScan1 Scan Formats emphasize flexibility over performance. The remaining optional OScan2–OScan7 Scan Formats emphasize performance over flexibility. The SScan Scan Format utilizes the DTS' knowledge of the scan transfer to improve scan performance by transferring only needed information for *Shift-xR* states. The OScan and SScan Scan Formats are separated into two groups with the same function. The first group may be used with either a DTS- or a TS-sourced TCKC. The second group has better performance but may only be used with a DTS-sourced TCKC.

T4 and above TAP.7s supporting different features are interoperable as only the TAP.7 Controllers supporting a scan format or the use of a feature supported only by a T5 TAP.7 are allowed to operate with these features. When the use of the Advanced Protocol is specified, a T4 and above TAP.7 checks that it supports features specified by its Global Registers following Command Part Two or load of Global Registers by the Control Protocol. When it detects the use of an unsupported scan format or the use of a feature supported only by a T5 TAP.7, it halts its operation and places itself in an Offline state. This prevents its exposure to Advanced Protocol bit sequences it does not understand. This prevents a loss of synchronization with the DTS and other TAP.7 Controllers and permits resynchronization to the remainder of the system at a later point in time.

The operation of an Offline TAP.7 Controller may be resynchronized to the remainder of the system using the Control Protocol initiated by a Selection Escape or Selection Alert. The resynchronization operation

places all TAP.7 Controllers with a specified TAPC state Online with their operation synchronized using the same Global Register values. This operation can be specified as affecting a specific TAP.7 Technology Branch or all TAP.7 Technology Branches.

## 6.2.2 Operation

### 6.2.2.1 Signal behaviors

The attributes of the T4 TAP.7 signals are listed as follows:

- The TCKC signal:
  - The Test Clock
- The TMSC signal:
  - An input when the Standard Protocol is used
  - Bidirectional when the Advanced Protocol is used
- The TDI and TDO signals:
  - Deleted with a narrow TAP
  - One of two functions with a wide TAP.7:
    - Fixed As the T3 TAP.7 TDIC and TDOC pin functions
    - Programmable As the T3 TAP.7 TDIC and TDOC signal functions or auxiliary functions with a TAP.7 Controller reset establishing the default signal function

### 6.2.2.2 Rising and falling TMSC input sampling

When the Advanced Protocol is used, the TAP.7 Controller provides for sampling the TMSC signal value on either the rising or falling edge of the TCKC signal. With full-cycle timing, information propagates from source beginning with a falling TCKC edge and is sampled at its destination with the next falling TCKC edge. With half-cycle timing, information propagates from the source beginning with a falling TCKC edge and is sampled at its destination with the next rising TCKC edge. A TAP.7 register defines the type of timing used. Half-cycle timing is used as the default after a TAP.7 Controller reset. The maximum TCKC frequency is reduced with the use of half-cycle timing as only half the time is available to propagate information between the DTS and the TS when compared with the use of full-cycle timing.

This feature allows the DTS to trade TAP bandwidth for noise immunity, in some applications. Rising-edge sampling improves noise immunity as the data is driven at the time it is sampled, provided the TCKC frequency allows the driven value to propagate to the point where it is being sampled before the rising edge of TCKC occurs. Rising-edge timing can also be used to overcome a TMSC hold-time problem should one occur.

Using rising-edge TMSC input sampling has the disadvantage of allowing one half of a TCKC period to propagate a TMSC value between the DTS and the TAP.7 Controller instead of the full-clock period afforded by falling-edge sampling. With falling-edge sampling, data is sampled while the signal value is maintained by a keeper, in some cases, thereby lowering noise immunity. The type of TMSC sampling may be specified while the Standard Protocol is used before there is a dependence on this setting. The DTS will ensure the operating frequency is compatible with the TMSC input-sampling setting.

### **6.2.2.3 Controller addressability in a Star-2 Scan Topology**

The TAP.7 Controller addressability in a Star-2 Scan Topology is the same as for the Star-4 Scan Topology. The operation of SSDs and allocation of CIDs is similar to the same functions in a Star-4 Scan Topology. The information conveyed with the TDIC and TDOC pins for CID allocation with a T3 TAP.7 is conveyed entirely with the TMSC pin when the Advanced Protocol is used.

#### **6.2.2.4 RSU and APU functions**

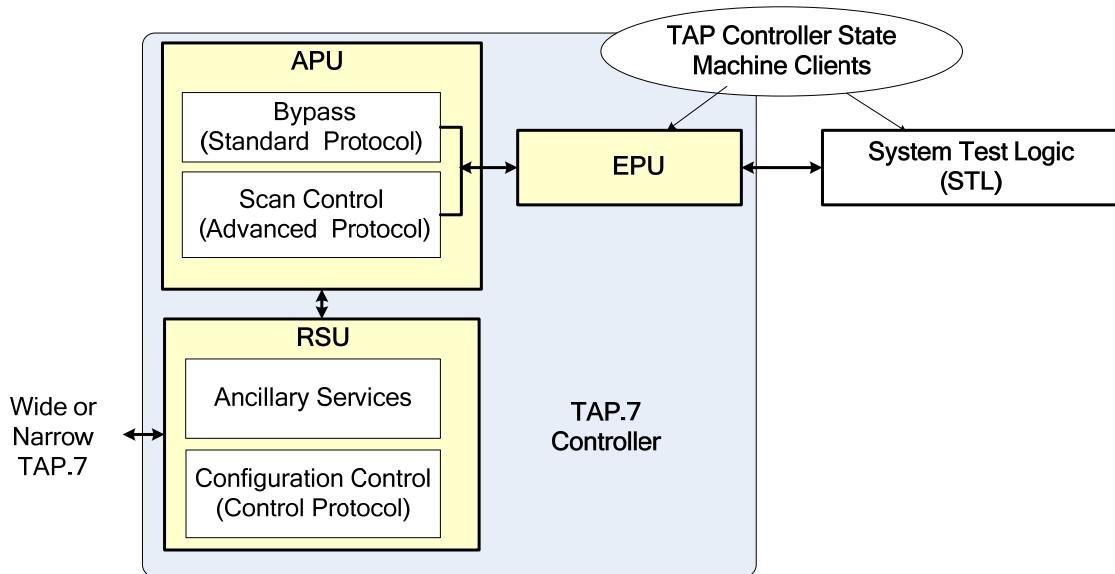
A T4 TAP.7 APU provides the following functions listed:

- Bypass Use the Standard Protocol
  - Scan Control Use the Advanced Protocol (advance the TAPC state with serialized TDI, TMS, and TDO data)

The RSU provides the following functions listed:

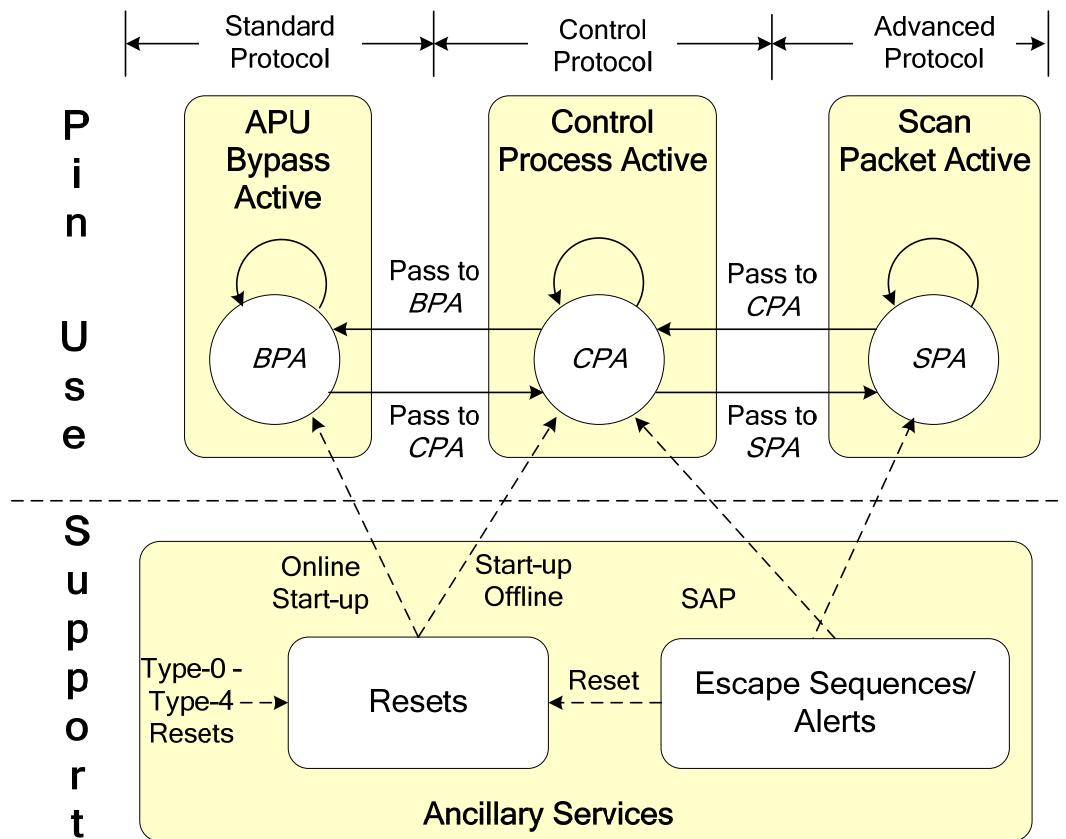
- Configuration Control Manage Online/Offline operation involving the use of unsupported features, Selection and Deselection Escapes, and Selection and Deselection Alerts
  - Ancillary Services TAP.7 Controller resets and Escape detection

The Bypass function utilizes the TMSC signal with the Standard Protocol while the Scan Control function utilizes the TMSC signal with the Advanced Protocol. The APU is the source/destination of control information within the Advanced Protocol while either the TAP.7 Controller or the STL may be the source/destination of the data information within this protocol. The RSU utilizes the TMSC signal with the Control Protocol as shown in Figure 6-2.



**Figure 6-2 — Conceptual block diagram of the functions of a T4 TAP.**

A conceptual view of APU operation is shown in Figure 6-3. This figure shows the APU functions partitioned into two groups, pin use and support.



**Figure 6-3 — TMSC pin utilization flow diagram for a T4 TAP.7**

A TAP.7 Controller views the information transferred via the TCKC and TMSC signals as the Standard Protocol in the *BPA* state. It views the information transferred as an SP with the *SPA* state and the Control Protocol with the *CPA* state. The TMSC pin use is passed between the *BPA*, *CPA*, and *SPA* states based on the events described in Table 6-1.

**Table 6-1 — Events causing TMSC pin utilization changes with a T4 TAP.7**

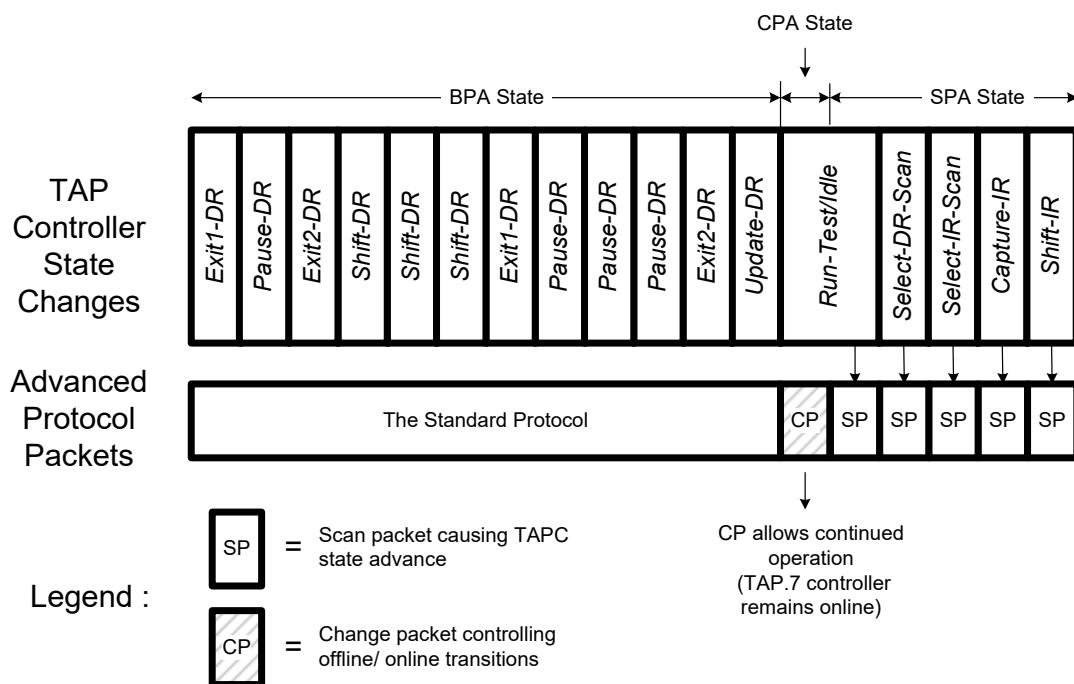
From state	To state	Events causing a change in TMSC pin use
<i>BPA</i>	<i>CPA</i>	- Selection/deselection events - After a register-write ( <i>Update-DR</i> state of CP2 (Command Part Two) while using the Standard Protocol when the use of the Advanced Protocol is specified (specifying the use of a scan format other than JScan0–JScan3).
<i>CPA</i>	<i>BPA</i>	Check Packet is complete/Standard Protocol is to be used.
<i>CPA</i>	<i>SPA</i>	Check Packet is complete/Advanced Protocol is to be used.
<i>SPA</i>	<i>CPA</i>	- Selection/deselection events - After the SP associated with the TAPC state after the SP associated with the <i>Update-DR</i> state of CP2 of any command.

#### 6.2.2.4.1 Bypass (BPA)

When the Standard Protocol is used, the APU connects the EPU inputs and outputs directly to the TAP.7 signals. If the TDIC and TDOC pin functions are not present (e.g., a narrow TAP.7 or the TDIC and TDOC signals have an auxiliary function assigned to them), the APU sources logic 1 TDI and TDO input values to the EPU and ignores the TDO value sourced by the EPU.

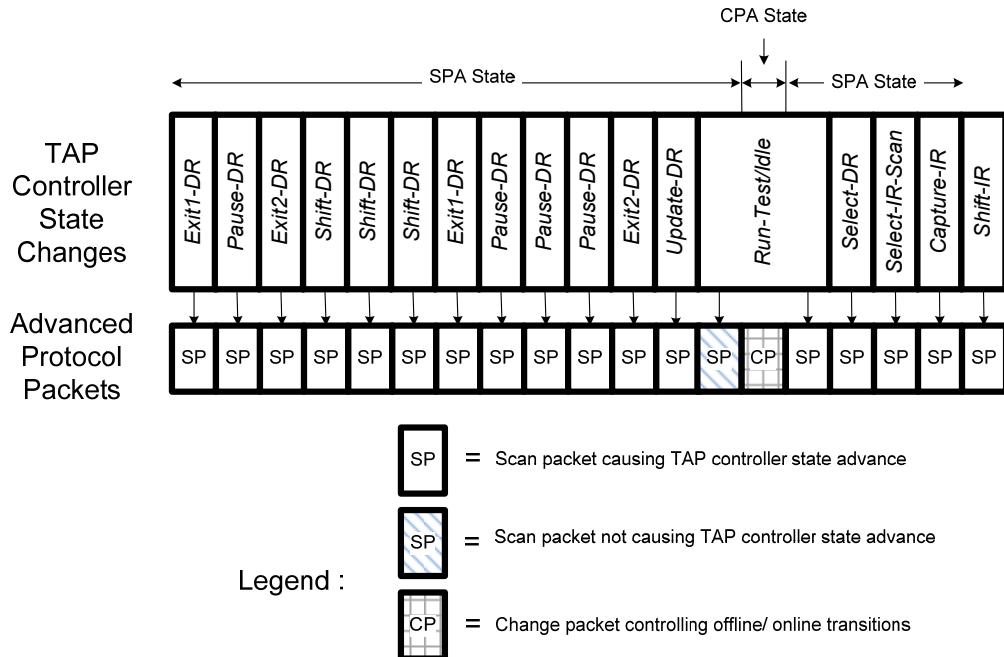
#### 6.2.2.4.2 Check Process Active (CPA)

While using the Advanced Protocol, all TAP.7 Controller configuration changes occur within the *CPA* state. The TAP.7 Controller Online and Offline scan selection state transitions also occur within the *CPA* state. When entered from the *BPA* or *SPA* state, the *CPA* state checks for the use of unsupported features that affect the Advanced Protocol bit sequences before allowing the use of the Advanced Protocol. The *CPA* state is exited only when the enabled features affecting the Advanced Protocol are supported. Figure 6-4 shows *CPA* state entry from the *BPA* state with continued Online operation following the above-mentioned check. Figure 6-5 shows *CPA* state entry from the *SPA* state with continued Online operation following the above-mentioned check.



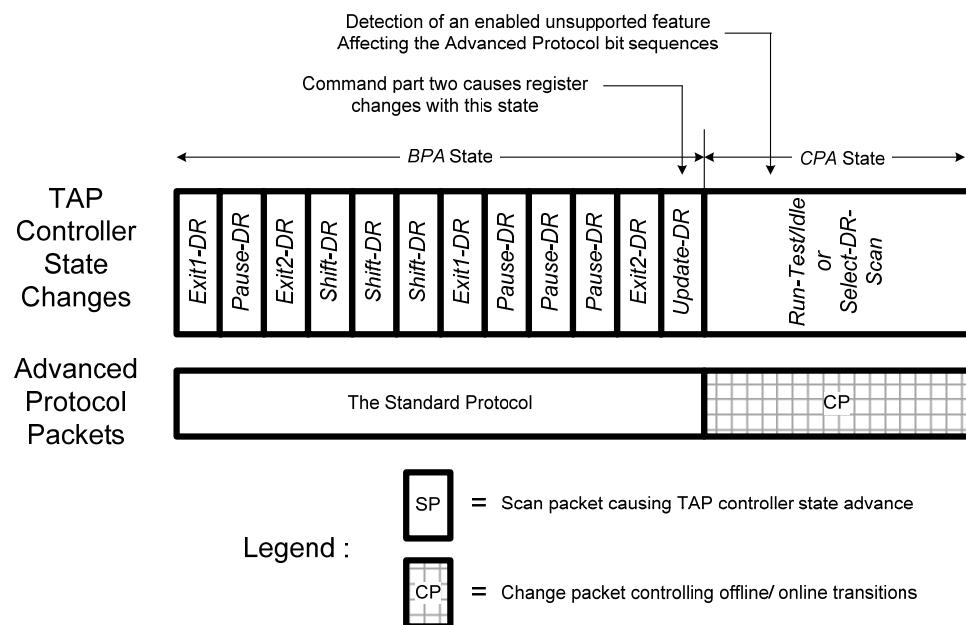
**Figure 6-4 — CPA state entry from the *BPA* state with subsequent Online operation**

An SP preceding the CP does not advance the TAPC state as shown in Figure 6-5. This SP is considered part of the Control Protocol as it performs housekeeping functions that ensure the contents of the SP preceding this SP have been fully utilized. This has significance when the SP contains control information, which allows the STL to stall the completion of an SP.

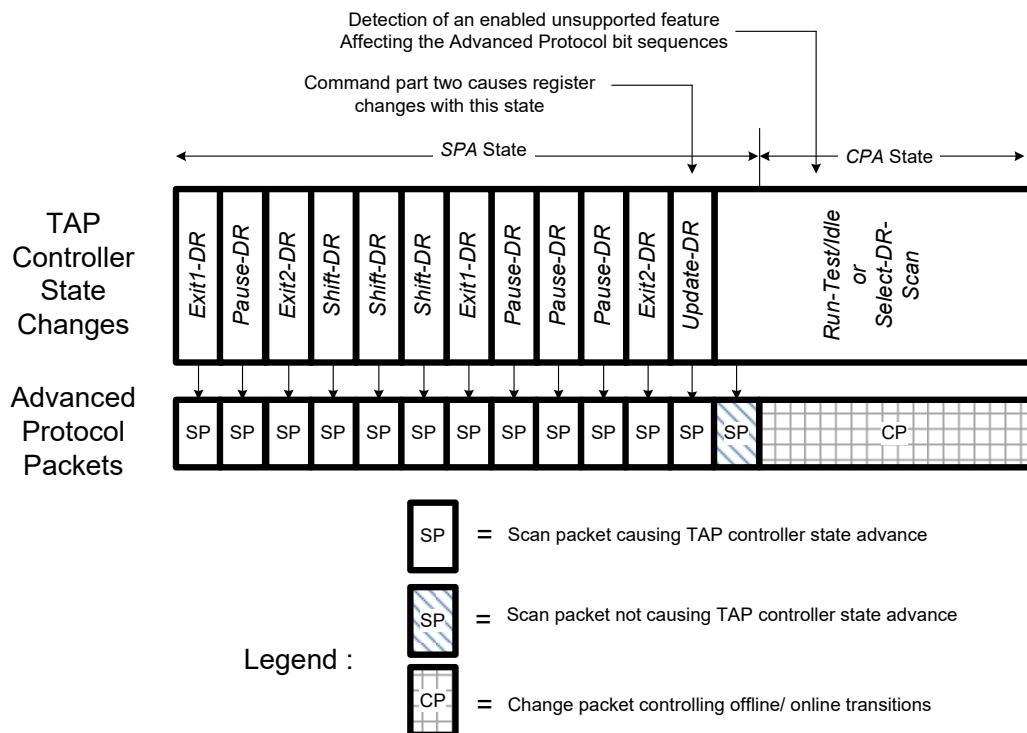


**Figure 6-5 — CPA state entry from the SPA state with subsequent Online operation**

The APU stops advancing the state of the ADTAPC and CLTAPC and suspends Data channel activity (with a T5 TAP.7) while the TAP.7 Controller is Offline. The detection of the appropriate Selection Sequence while the TAP.7 Controller is Offline places the TAP.7 Controller Online and causes an exit from the *CPA* state to either the *SPA* or *BPA* state, provided the specified feature set is supported. The state remains *CPA* otherwise. An *Offline-to-Online* selection state transition initiates the use of a predefined set of mandatory features in all *Online* TAP.7 Controllers to ensure they operate synchronously. This is shown in Figure 6-6 and Figure 6-7.

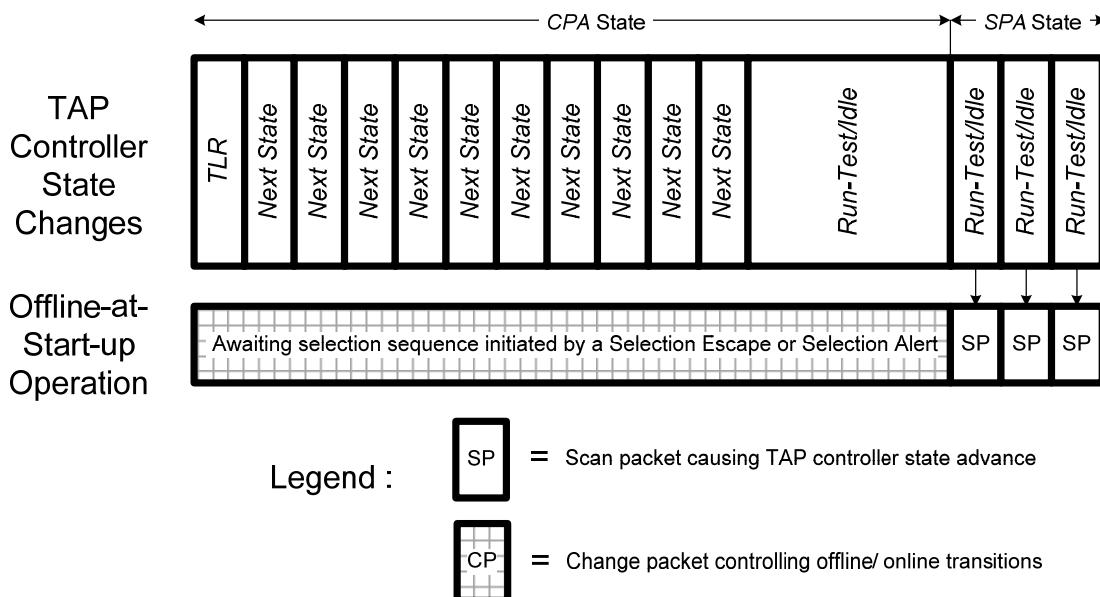


**Figure 6-6 — CPA state entry from the BPA state with subsequent Offline operation**



**Figure 6-7 — CPA state entry from the SPA state with subsequent Offline operation**

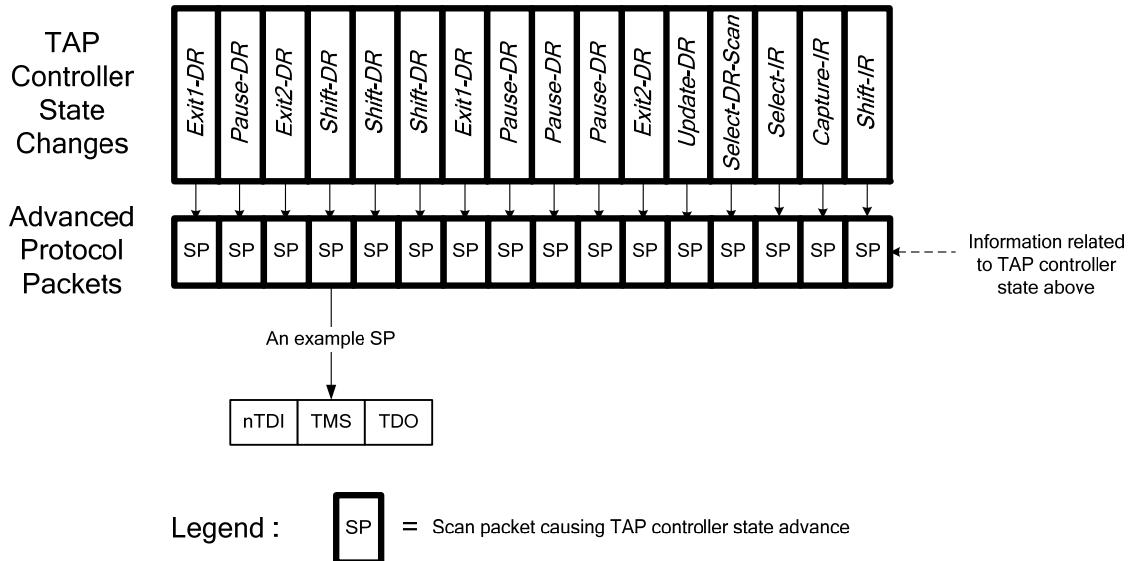
A TAP.7 Controller reset places the TAP.7 Controller Offline within the *CPA* state when the Offline-at-Start-up option is used. In this case, the TAP.7 Controller moves the STL TAPC state to *Run-Test/Idle*. From this point, the TAP.7 Controller awaits being placed Online. This is shown in Figure 6-8.



**Figure 6-8 — Reset initiated CPA state entry/awaiting a Selection Sequence for an exit**

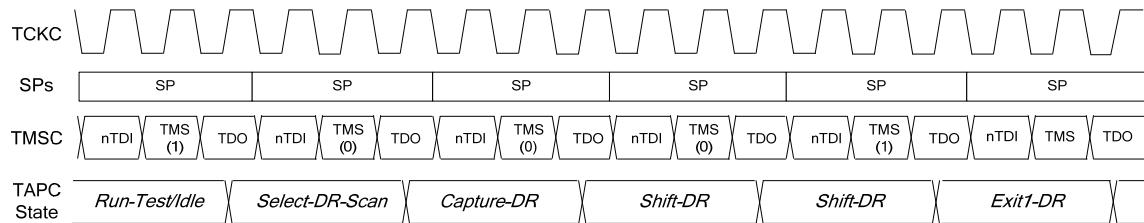
### 6.2.2.4.3 Scan Packet Active (SPA)

The scan control function is encapsulated within the *SPA* state. With the Standard Protocol, the DTS and TAP.7 exchange the scan information associated with a single TAPC state in one TCKC period using the TMS(C), TDI(C), and TDO(C) pins. With the Advanced Protocol, all or part of this information is exchanged serially within an SP in one or more TCKC periods as shown in Figure 6-9.



**Figure 6-9 — SP and TAPC state relationships**

An expanded view of an example SP bit sequence is shown in Figure 6-10.



**Figure 6-10 — Expanded view of an SP sequence**

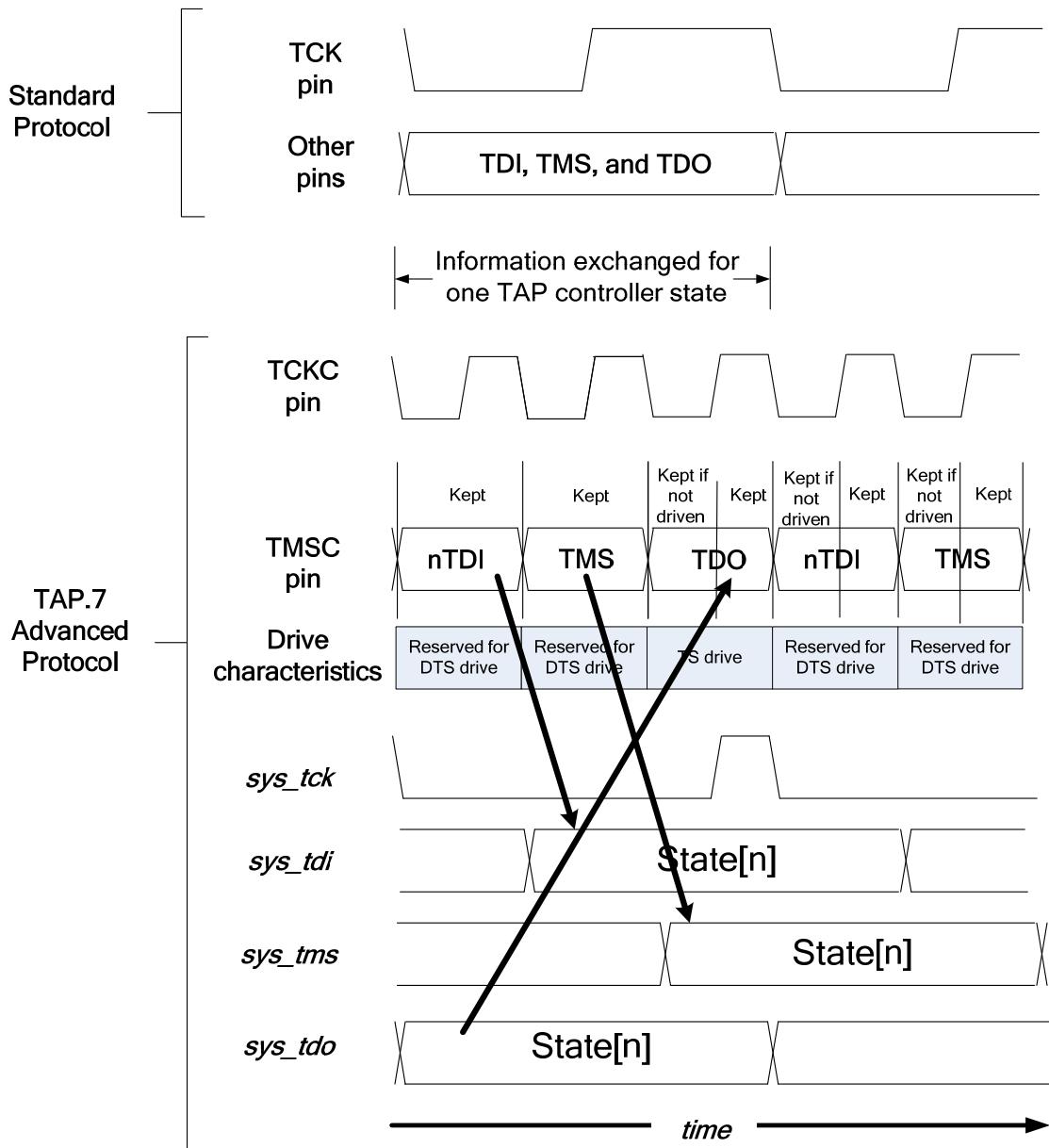
There is a one-to-one correspondence between SPs and TAPC state changes, except when the *SPA* state precedes the *CRA* state. In this case, the SP preceding a CP does not advance the TAPC state and is considered part of the Control Protocol as stated previously.

The APU manages the serialization and deserialization of the scan information within Scan Packets. The APU converts the TDI and TMS information within an SP into the TMS and TDI information presented to the EPU. It converts System and Control Path TDO data into the TDO information.

A comparison of the parallel transfer of TAP information with the Standard Protocol and the serialization and deserialization of the TAP information using the TCKC and TMSC pins with the Advanced Protocol is shown in Figure 6-11. The IEEE 1149.1 signals between the EPU and STL are prefixed with SYS\_ in this figure and elsewhere in this document.

Information within an SP is first passed from the DTS to the TAP.7 followed by information passed from the TAP.7 to the DTS. In Figure 6-11, the scan format specifies the exchange of the TMS, TDI, and TDO

information for each TAPC state and takes three TCKC periods. Other scan formats specify the exchange of different mixes of information. Control information is added with the most flexible scan formats. Some scan formats send less information and consequently take fewer TCKC periods.



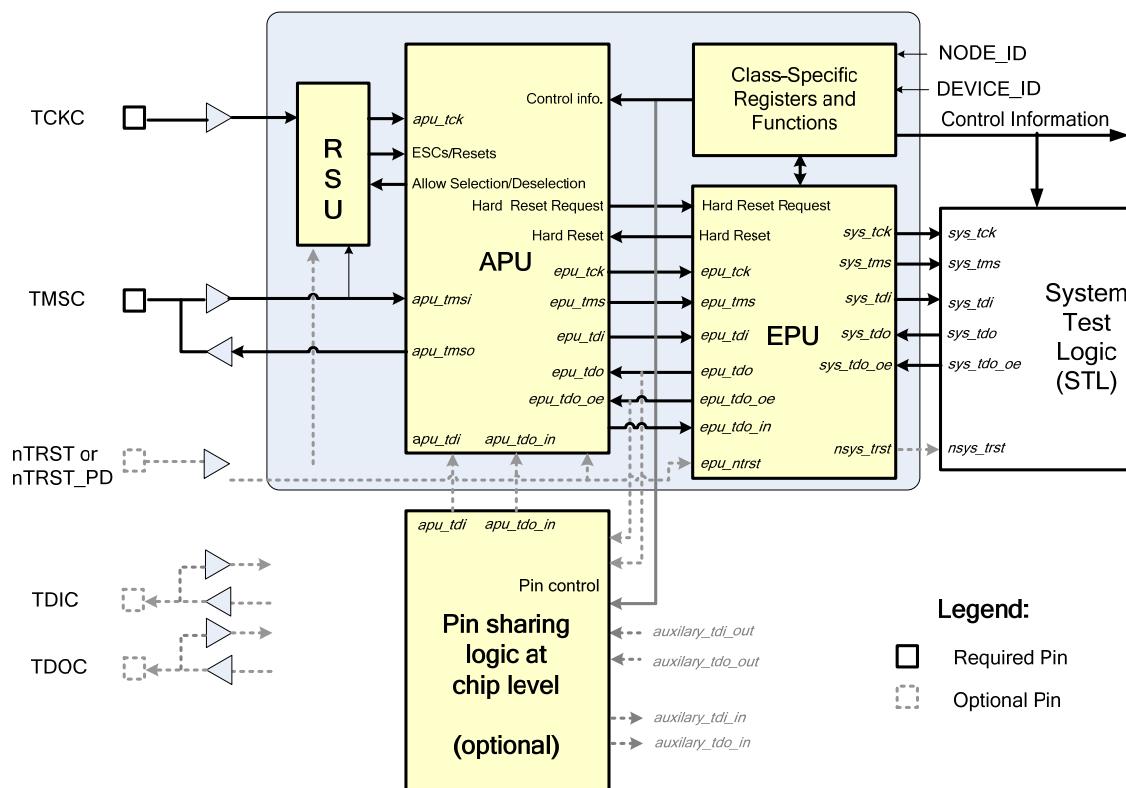
**Figure 6-11 — Serialization and de-serialization of SP information**

### 6.2.3 T4 TAP.7 high-level block diagram

A high-level block diagram of a T4 TAP.7 Controller is shown in Figure 6-12. It shows the addition of the following to the T3 TAP.7 block diagram:

- Advanced Protocol Unit (APU)
- Optional TDIC and TDOC pin-sharing logic external to the TAP.7 Controller

The APU does not change the EPU's behavior. It instead multiplexes the use of the TAP.7 pins between the EPU and APU. When the APU is connected to the TAP signals, the APU is connected to the EPU TAP. The RSU also uses the TCKC and TMSC pins for Configuration Control.



Note: nTRST is the Test Reset value with a default logic 1 level and  
nTRST\_PD is the Test Reset value with a default logic 0 level

**Figure 6-12 — High-level T4 TAP.7 functional block diagram**

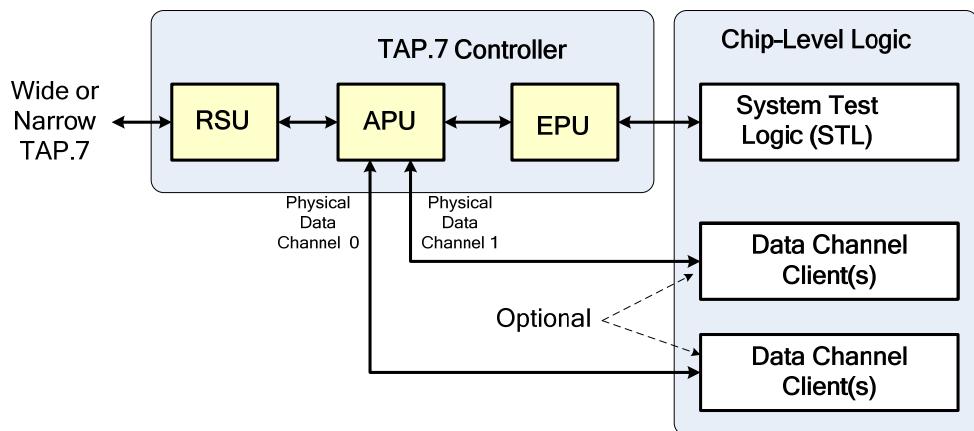
## 6.3 T5 TAP.7

### 6.3.1 Overview

The Transport Function is added to the advanced capabilities of a T4 TAP.7 to create a T5 TAP.7. All mandatory capabilities of a T4 TAP.7 are mandatory for a T5 TAP.7. The optional capabilities supported by lower TAP.7 Classes are also optional for a T5 TAP.7.

T4 and T5 TAP.7s are interoperable in a Star-2 Scan Topology. When a T5 TAP.7 Controller detects the use of an unsupported scan format or certain reserved T5 TAP.7 register values, it is placed Offline. The T5 TAP.7 is placed Online in the same manner as a T4 TAP.7.

The Transport Function with data channels and their connection to the APU are shown in Figure 6-13.



**Figure 6-13 — T5 TAP.7 chip block diagram**

A T5 TAP.7 Controller is configured with none, one, or two Physical Data Channels (PDCs) and a Transport State Machine. The Transport State Machine provides for operation when Transport Packets are included in the Advanced Protocol. This machine ensures the TAP.7 Controller remains synchronized to the Advanced Protocol independent of whether Physical Data Channels are implemented, provided only supported scan formats are used. A chip architect should consider implementing a T5 TAP.7 with no PDCs in lieu of a T4 TAP.7. Incurring the modest cost of this TAP.7 ensures the TAP.7 Controller is never placed Offline when the DTS enables the use of T5 TAP.7 transport capabilities.

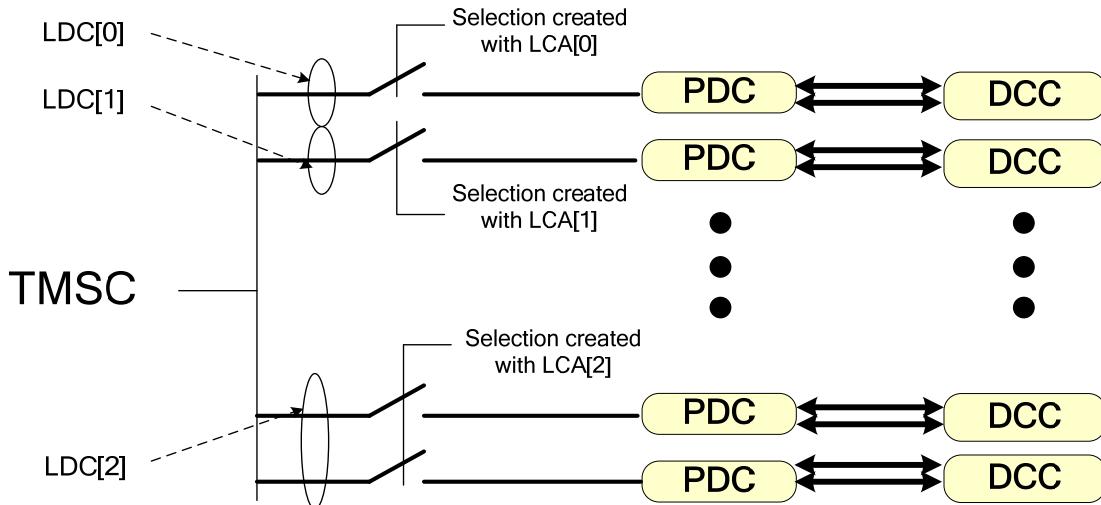
A Physical Data Channel provides for non-scan information transfers between a Chip-Level Data Channel Client (DCC) and the DTS or one or more other Data Channel Clients when the Advanced Protocol is used. It is initialized when the Standard or Control Protocol is used. Data and control information are moved from the TMSC pin to a Data Channel Client and vice versa through a Physical Data Channel. A Data Channel Client can be connected directly to the Physical Data Channel, or from 2 to 16 Data Channel Clients can be connected to a Physical Data Channel through a Data Channel Router connected to the Physical Data Channel.

A Physical Data Channel is identified with its Physical Channel Address (PCA). This address is the combination of the TAP.7 Controller's CID and the PDC number (1 or 0). This combination makes the Physical Channel Address of a Physical Data Channel unique when all TAP.7 Controllers sharing the DTS connection have a unique CID. The DTS uses this address to select a Physical Data Channel for control operations (such as accessing Transport Control Registers within the TAP.7 Controller and Data Channel Client).

The Transport Function provides for data exchanges where groups of one or more Physical Data Channels in one or more TAP.7 Controllers operate together. One or more Physical Data Channels may be associated with a Logical Data Channel. Eight separate Logical Data Channels are supported, providing a means for a DTS to sequentially communicate with eight groups of Data Channel Clients in any order for any duration as determined by the DTS. Each of the eight Logical Data Channels has a unique Logical Channel Address. The data exchanges are initiated by using a Transport Directive. The Logical Channel Address of the Logical Data Channel used for the exchange is embedded in a Transport Directive.

A Logical Data Channel (LDC) is created at run-time by aliasing a Logical Channel Address to one or more Physical Data Channels. This aliasing process is similar to the use of Controller IDs but with one significant difference. A Logical Channel Address may be allocated to more than one Physical Data Channel. This permits data exchanges between multiple Physical Data Channels and the Data Channel Clients connected to them.

The use of multiple Logical Data Channels is shown in Figure 6-14. LDC[0] and LDC[1] are formed with a single Physical Data Channel. LDC[2] is formed with two Physical Data Channels. LDC[0] and LDC[1] support data transfers between the DTS and a single Data Channel Client. LDC[2] supports data transfers between the DTS and a single Data Channel Client, multiple Data Channel Clients, or both. As stated previously, it is expected that a high-level protocol prevents TMSC drive conflicts in the case of LDC[2].



**Figure 6-14 — Logical channel configurations**

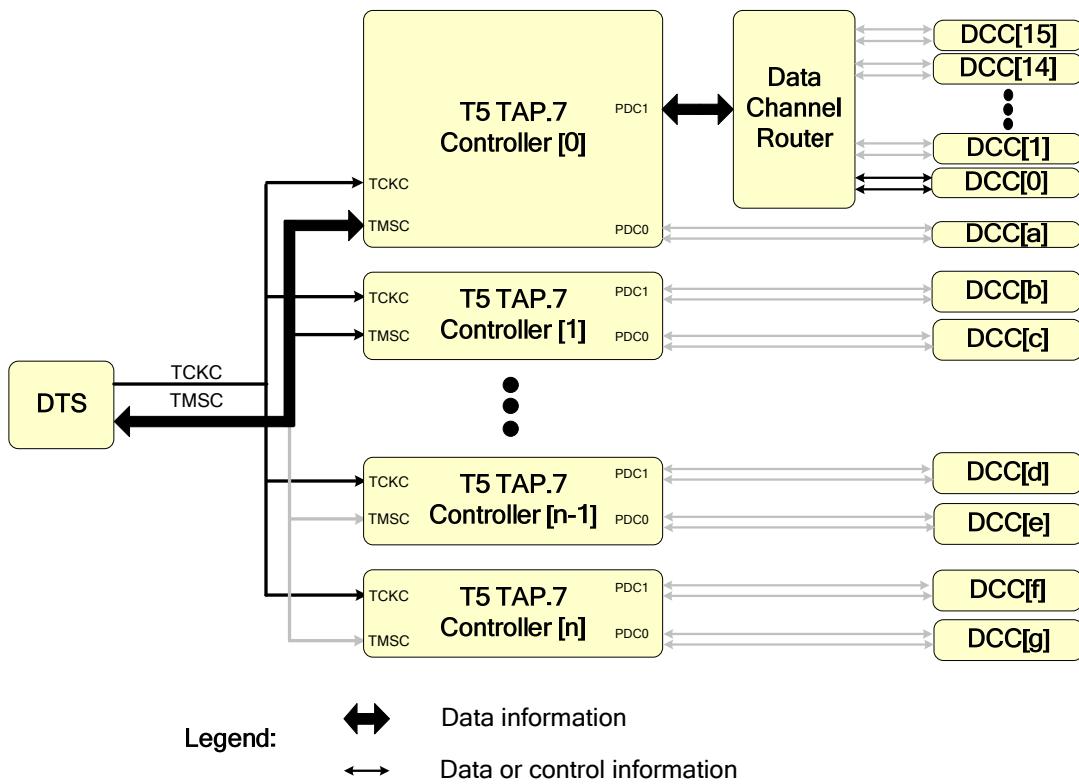
### 6.3.2 Operating modes and capabilities

#### 6.3.2.1 Transport source/destinations

The Transport Function provides for the use of each of the eight Logical Data Channels for single-client, multi-client, and client-to-client data exchanges. The DTS may interact with the Logical Data Channels in any order for any period of time. These data exchange types are described in the following subclauses.

##### 6.3.2.1.1 Single-client operation

A system where the DTS uses a Logical Data Channel to exchange data information with a single Data Channel Client is shown in Figure 6-15. Data is exchanged between the DTS and DCC [0] connected to PDC1 of TAP.7 Controller [0]. PDC1 of this TAP.7 Controller is the only PDC forming this Logical Data Channel. This figure may be repeated for each of the eight Logical Data Channels formed with a different Physical Data Channel. Other Logical Data Channels may be formed with PDCs other than PDC1 of TAP.7 Controller [0]. PDCs and the DCCs connected to them can only interact with the DTS when their LDC is activated.



**Figure 6-15 — DTS data exchanges with one client**

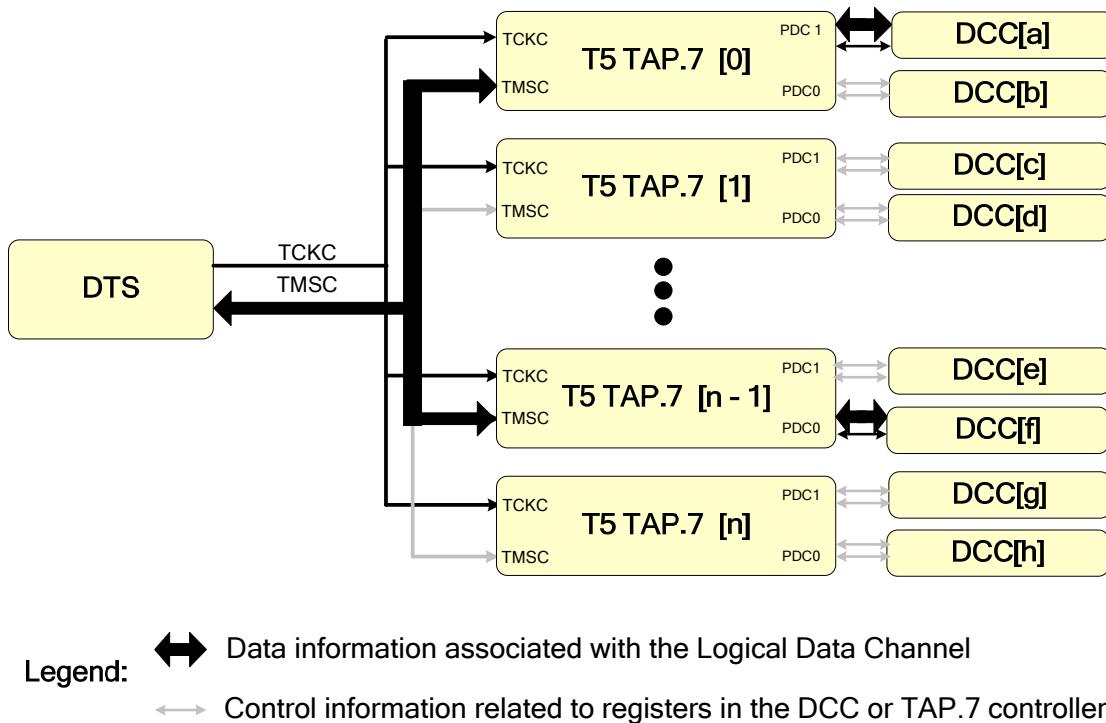
### 6.3.2.1.2 Multi-client operation

A system where the DTS uses a Logical Data Channel to exchange data information with multiple Data Channel Clients is shown in Figure 6-16. This Logical Data Channel configuration requires a higher level protocol to perform the following tasks:

- Coordinate the activity of the Data Channel Clients
- Manage TMSC signal drive during a data transfer to prevent TMSC signal-drive conflicts

This type of operation is also called “multi-drop.”

In this example, data is exchanged between the DTS and either DCC[a] connected to PDC1 of TAP.7 Controller [0] or DCC [f] connected to PDC0 of TAP.7 Controller [n – 1].



**Figure 6-16 — DTS data exchanges with more than one client**

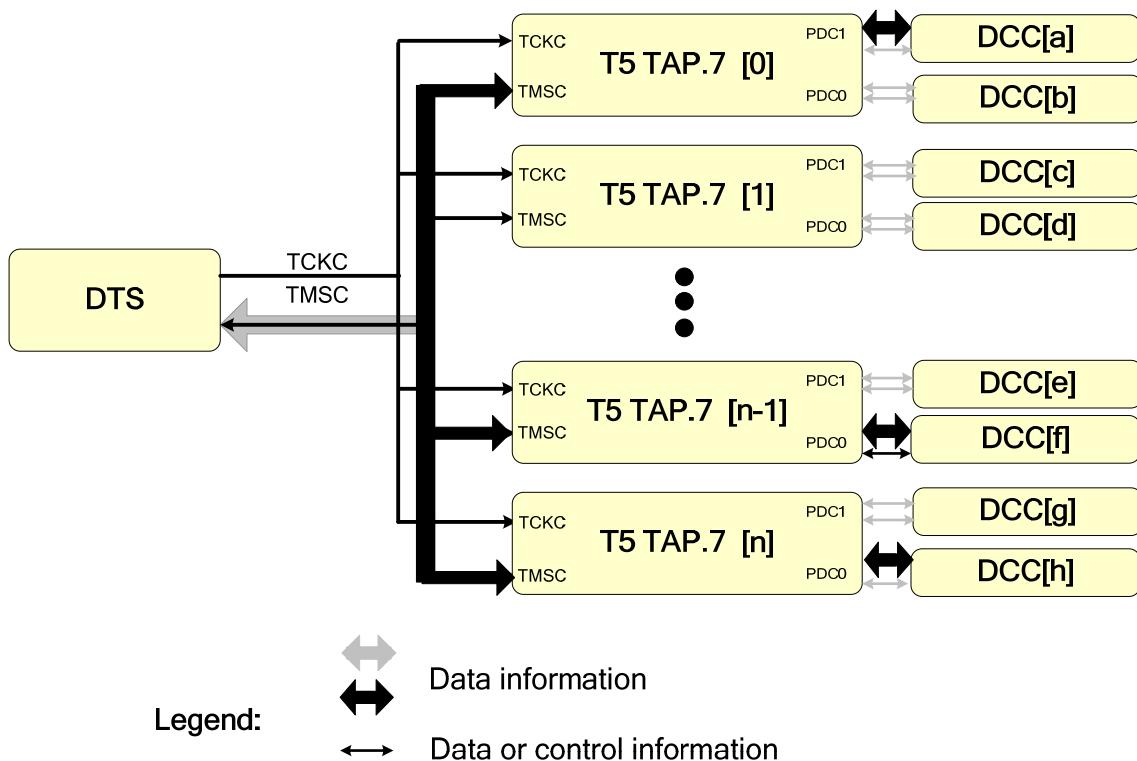
### 6.3.2.1.3 Client-to-client operation

A system where a data channel is used to exchange information between Data Channel Clients and possibly the DTS is shown in Figure 6-17. This Logical Data Channel configuration also requires participating Data Channel Clients to deploy a higher level protocol to ensure there are no TMSC signal-drive conflicts.

In this example, the data exchange occurs between the following:

- DCC[a] connected to PDC [1] of TAP.7 Controller [0]
- DCC[f] connected to PDC [0] of TAP.7 Controller [n – 1]
- DCC[h] connected to PDC [0] of TAP.7 Controller [n]

In this example, the DTS does not participate in the data exchange. It merely manages the Logical Data Channel used for these transfers. It can, however, be included in the data exchange.



**Figure 6-17 — DTS data exchange between multiple clients**

### 6.3.2.2 Transfer characteristics

Data channel transfers may be used for any purpose. A Chip-Level Data Channel Client may be a function as simple as a serial port or a much more sophisticated function utilizing a multi-drop communications protocol. More than one Physical Data Channel may be associated with a Logical Data Channel, provided the protocol used by the Data Channel Clients sharing the Logical Data Channel for data exchanges manages the TMSC Signal Drive Policy during these data exchanges to prevent drive conflicts. This means the higher level protocol ensures only one TAP.7 Controller drives the TMSC pin at any point in time during Transport Packets. The TAP.7 Controller neither utilizes nor understands the data transferred through a Physical Data Channel.

The configuration and control of transport channels are managed using Transport Directives introduced in 4.3.3.2. Certain directives write registers, within the TAP.7 Controller, read and write register within the Data Channel Client, while others initiate Data Exchanges and terminate a Transport Packet. Data Exchanges have two key attributes, the direction of transfer and the number of contiguous data bits transferred before control information is transferred. The number of contiguous data bits transferred can be either fixed or variable. A fixed-length transfer automatically terminates after 8, 16, 32, or 64 data bits (its length is programmable). A variable-length transfer is terminated with an End of Transfer (EOT) Escape and may be used only when the DTS sources the Test Clock. Fixed-length data transfers may be used with either a DTS- or TS-sourced Test Clock.

A means to allocate the data-transfer bandwidth to input and output is provided. The transfer may be specified as:

- Input      All data is transferred from the DTS to the chip-level unit
- Output     All data is transferred from the chip-level unit to the DTS

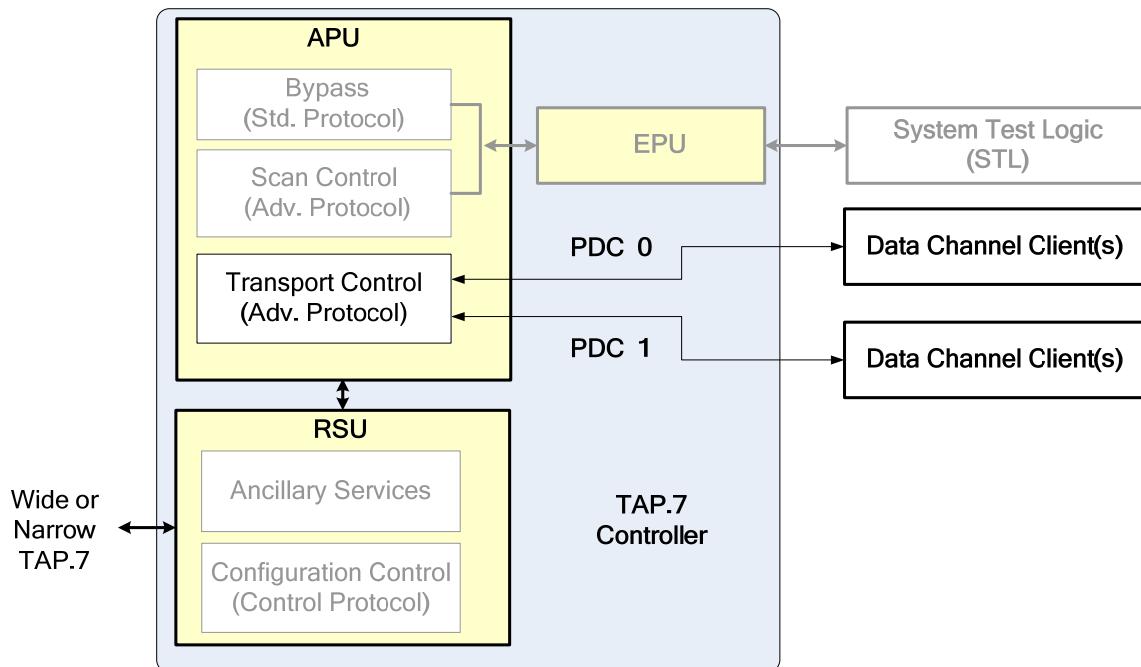
- Input/output (I/O)      The data-transfer direction changes every bit
- Custom                  The chip-level unit defines the direction of each bit transferred

A data channel application using the data channel is called a Custom Data Transfer (CDX) when the allocation of input and output bandwidth and direction of the transfer is managed in a custom manner and a Background Data Transfer (BDX) otherwise.

### 6.3.3 Operation

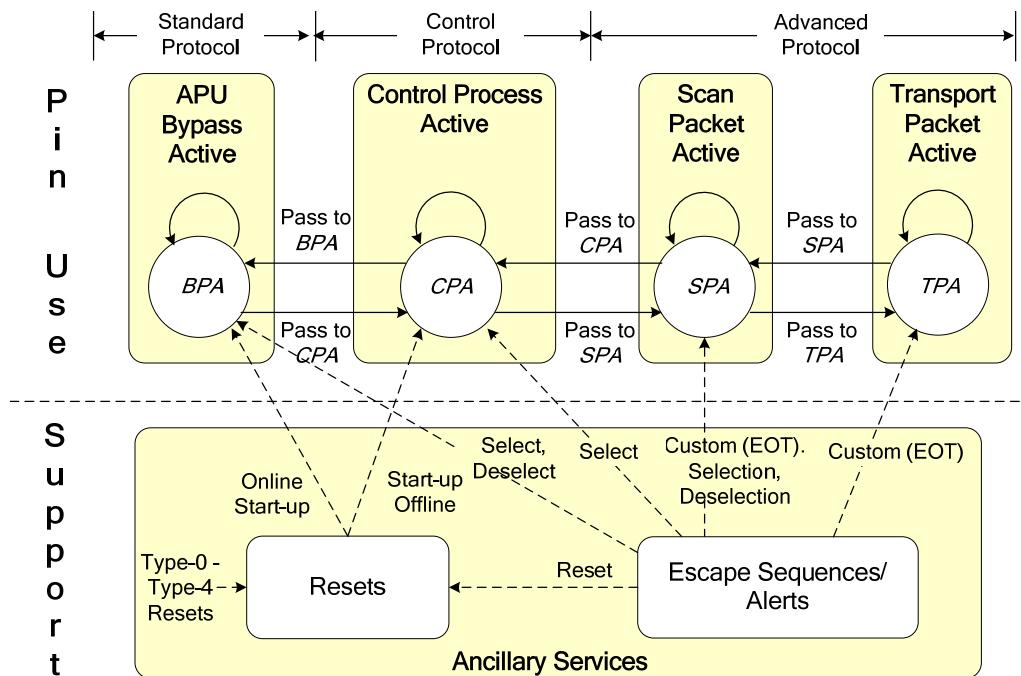
#### 6.3.3.1 Transport Control Function

With a T5 TAP.7, a Transport Control Function is added to an APU as shown in Figure 6-18. It implements the portion of the Advanced Protocol related to data channel transfers.



**Figure 6-18 — Conceptual block diagram of the functions of T5 TAP.7**

With a T5 TAP.7, an additional operating state (*TPA*) is added to those supported by a T4 TAP.7. The Transport Function controls the TMSC pin in this state. This state supports transfers between the DTS and configurations with two, one, or no Chip-Level Data Channels implemented at the chip level. The TMSC pin use is passed between the *BPA*, *SPA*, *CPA*, and *TPA* states as shown in Figure 6-19.



**Figure 6-19 — TMSC pin utilization flow diagram for a T5 TAP.7**

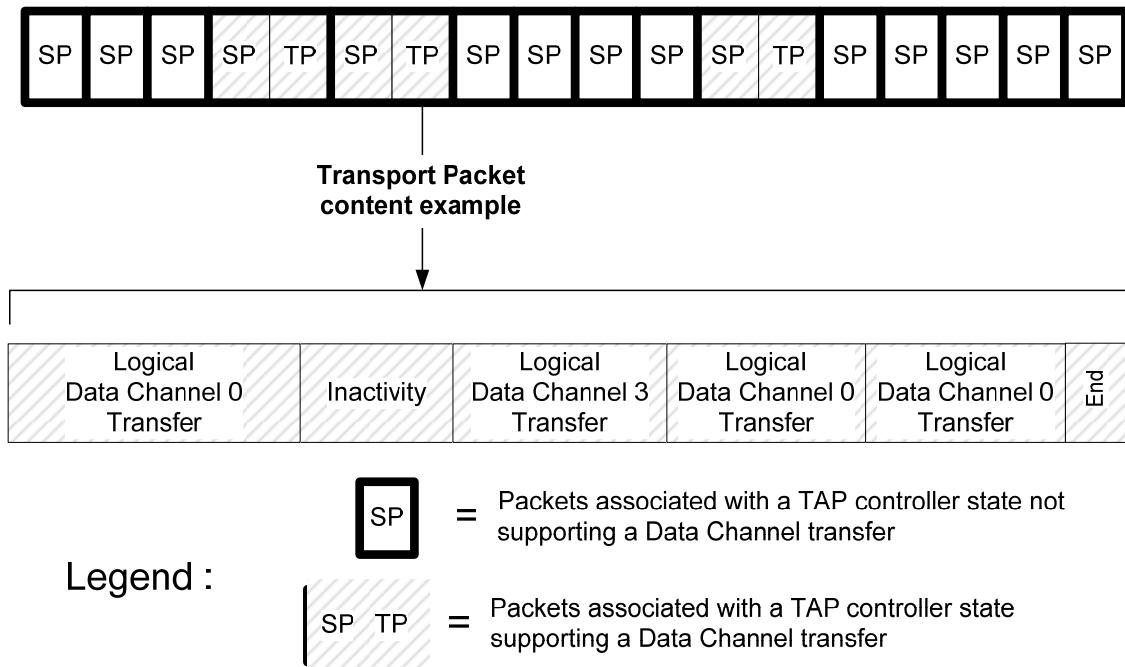
The events changing the use of the TMSC pin from scan to transport and vice versa are described in Table 6-2. These are added to the events shown in Table 6-1.

**Table 6-2 — Events causing TMSC pin utilization changes (with transport)**

Events causing a change in TMSC pin use	Current state	Next state
Following an SP associated with a TAPC state enabled to support data channel transfers.	SPA	TPA
An End Transport Directive is received in the protocol stream.	TPA	SPA

### 6.3.3.2 TPA

The Transport Control Function is encapsulated in the *TPA* state. A TP moves the data channel control and data information between the DTS and TS as shown in Figure 6-20.



**Figure 6-20 — TP placement and content**

A TP follows an SP associated with one of the TAPC states listed below:

- *Run-Test/Idle*
- *Pause-IR*
- *Pause-DR*
- *Update-IR*
- *Update-DR*

A TAP.7 Register designates any combination of these states (including none) as supporting Logical data channel transfers. When one of these TAPC states is designated as supporting Logical Data Channel transfers, a TP follows the SP associated with this state, unless a change in configuration is being initiated (a CP follows an SP). A single TP may include a mix of data exchanges with one or more Logical Data Channels, operations managing the Transport Control Function, operations managing Data Channel Client Control Registers, operations providing periods of inactivity (no operation), and an operation ending the TP.

#### 6.3.4 T5 TAP.7 high-level block diagram

A high-level block diagram of a T5 TAP.7 Controller is shown in Figure 6-21. It shows the addition of the data channels that provide non-scan communication in concert with scan transactions.

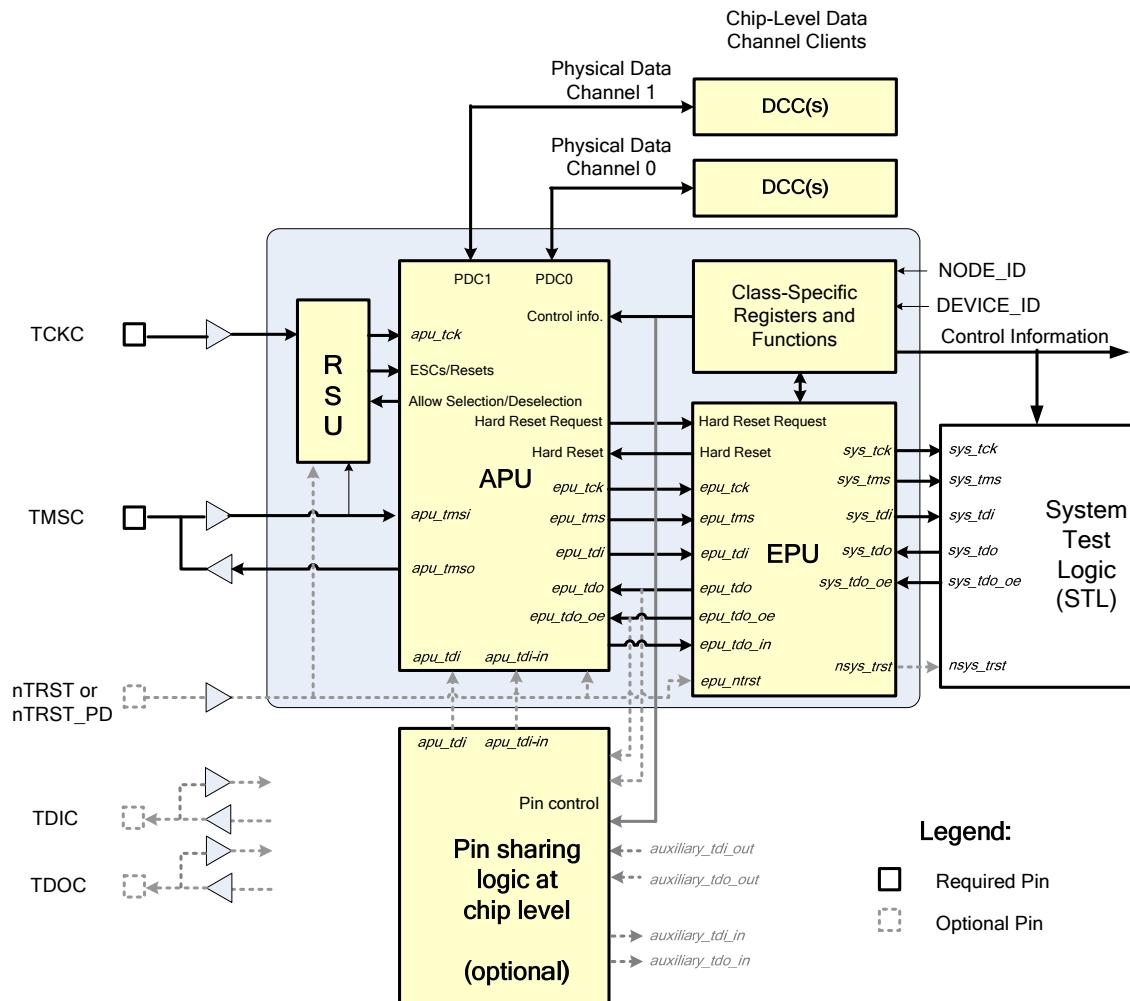


Figure 6-21 — High-level T5 TAP.7 functional block diagram

#### 6.4 TAP.7 feature summary

A summary of the TAP.7 features and the features supported by each TAP.7 Class is summarized in Table 6-3.

**Table 6-3 — TAP.7 feature group summary**

<b>Advanced – Data channels</b>					
Physical Data Channel 1					
Physical Data Channel 0					
No Physical Data Channel, don't go Offline					
Synchronized with transport protocol					
<b>Advanced – Operation within Star-2 Scan Topology</b>					
One of Four Start-up Options					
Placed Offline when unsupported feature is used					
Test Reset equivalent Escape					
Star-2 Drive Conflict Prevention					
Two/four pin (With or without TDIC/TDOC pins)					
Programmable function TDIC/TDOC pins					
<b>Scan Formats:</b>					
– Minimal Number are Mandatory					
– Very Optimized for Debug					
– Optimized for Debug					
– Optimized for Test					
– Optimized for flexibility					
<b>Extended – Operation within Star-4 Scan Topology</b>					
Directly addressable, TCA and CIDs					
Star-4 Drive Conflict Prevention					
Series/Star Scan Equivalence (SSDs)					
<b>Extended Series Performance</b>					
Selection/Deselection of CLTAPC					
Start-up With STL Deselected					
<b>Extended – Optional Functions</b>					
TAP.7 Power Control					
Test Reset Generation					
Functional Reset Request					
<b>Extended – Control Levels</b>					
Control Level Two – Cmds. and Regs.					
Control Level Three – Reserved.					
Control Level Four/Five – Scan Paths					
Control Level Six – Deselection					
Control Level Seven – DTS Use					
<b>Legacy Behavior</b>					
IEEE 1149.1-Specified Behavior					
Multiple Embedded TAPs					
Isolating/Excluding of Embedded TAPs					
<b>Foundation</b>					
TAP.7 Controller Selection/Deselection (T3-T5)					
TAP.7 Controller Reset with Escape (T3-T5)					
TAP.7 Controller Selection/Deselection (T0-T2)					
TAP.7 Controller Reset with Escape (T0-T2)					

NOTE—Features shaded in gray are options; features not shaded are mandatory for the supporting class.

## 7. System concepts

### 7.1 Introduction

This clause is applicable to T0 and above TAP.7s. It describes the TAP.7 system architecture from the viewpoint of the DTS. It describes the system connectivity possible using TAP.7s and other architectures.

The subject matter within this clause is organized in the following manner:

- 7.2 Key system attributes
- 7.3 DTS/TS connectivity with a mix of technologies
- 7.4 TAP.7 deployment scenarios
- 7.5 Chip TAPC hierarchy
- 7.6 Combined view of TAP connectivity and TAPC hierarchy
- 7.7 Chips, components, and boards

### 7.2 Key system attributes

Key system-related attributes of IEEE Std 1149.1 and this standard are shown in Table 7-1.

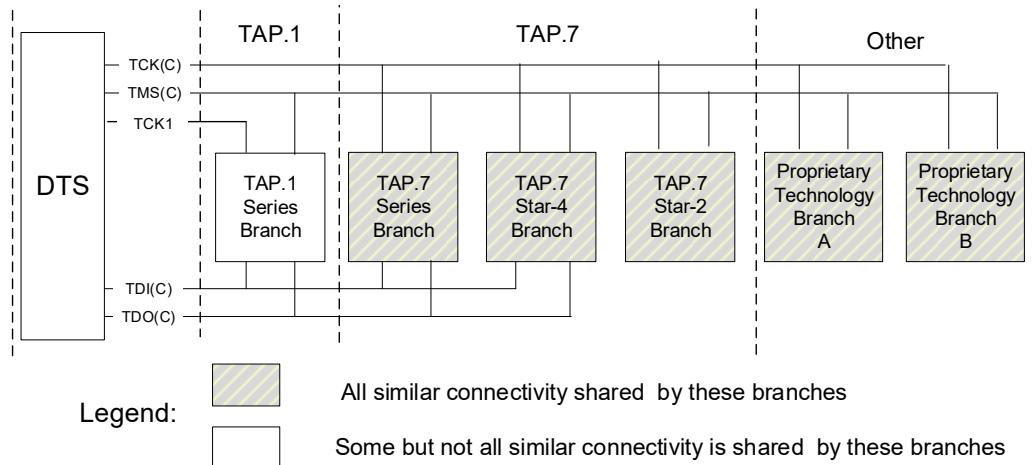
**Table 7-1 — IEEE 1149.1 and IEEE 1149.7 attributes affecting system architectures**

Attribute	IEEE 1149.1	IEEE 1149.7
Minimum pin count	4	2
TAPs per chip	1	$\geq 1$
Supports more than one TAPC per chip	No	Yes
Shared connectivity with legacy technologies	No	Yes
Shared connectivity with multiple scan topologies	No	Yes

### 7.3 DTS/TS connectivity with a mix of technologies

#### 7.3.1 Technology mixes

The TAP.7 architecture supports the deployment of any mix of TAP.1s, TAP.7s, and possibly proprietary technologies connected in the manner shown in Figure 7-1.



**Figure 7-1 — System connectivity with mixed use of technologies and scan topologies**

### 7.3.2 Technology branches

The connectivity shown in Figure 7-1 depicts the connection of technology branches. Most systems will implement only one branch of the connectivity shown in this figure. In most cases, a branch will only have a limited number of TAPs. The System Designer is responsible for following the electrical design considerations outlined in Annex F when connecting TAPCs to each other and the DTS.

Several proprietary Star-4 Scan Architectures evolved after IEEE Std 1149.1 and predate this standard. These scan architectures can also be included as additional TAP.1 Branches separate from those shown in Figure 7-1.

Each branch shown in Figure 7-1 may be operated separately for debug purposes. They may also be operated in a manner that provides Series Scan Equivalency across both the TAP.1 and TAP.7 Branches. This operation is similar to the Series/Star Scan Equivalency described in 5.5.3.4. Proprietary technologies are generally operated separately. This clause does not describe all possible combinations of branch operation.

The TAP.7 Branches have one or more TAP.7s connected in Series, Star-4, and Star-2 Scan Topologies. A branch is selectable when all TAP.7 TAPCs forming a branch are implemented with the Reset and Selection Unit (RSU). The RSU provides for the selection and deselection of these branches with Escapes. Recall that the use of the RSU is optional for T0–T2 TAP.7s and mandatory for T3 and above TAP.7s. It is recommended that an RSU is included in T0–T2 TAP.7 Controllers that may be deployed in systems where T4(N) and T5(N) devices may be deployed.

The TAP.1 Branch includes TAP.1s connected in series with one or more TAP.1s and TAP.7. It operates with the Standard Protocol and TAPs with the four IEEE 1149.1 signals. This branch is selected using the TCK1 signal shown in this figure as at least one TAPC in the branch has no built-in selection mechanism. T0–T2 TAP.7s without an RSU are included in a branch that is this type. The RSU logic may be added to the current IEEE 1149.1 technology to make it selectable using common connectivity.

Proprietary technologies that deploy the RSU function described in this specification may be connected in parallel with the TAP.7 Branches provided, provided all of the following are true:

- This technology has a dedicated clock and at least one control/data input.
- The technology implements the Technology Selection mechanism described in Clause 11.

- A technology's clock signal is connected to the TCK(C) signal and the technology's control/data signal is connected to the TMS(C) signal.

As noted previously, a TAP.1 Branch is selected by keeping its TCK or TMS signal separated from the signal of like (similar) named signals other branches. In this case, selection using the TCKC signal is recommended. TAP.1s and T0-T2 TAP.7s without the RSU are selected using this method. Any branch may be selected using this method, if desired.

TAP.1s and TAP.7s can be deployed in topologies as shown in Table 7-2. Each of these topologies is considered a technology within this document.

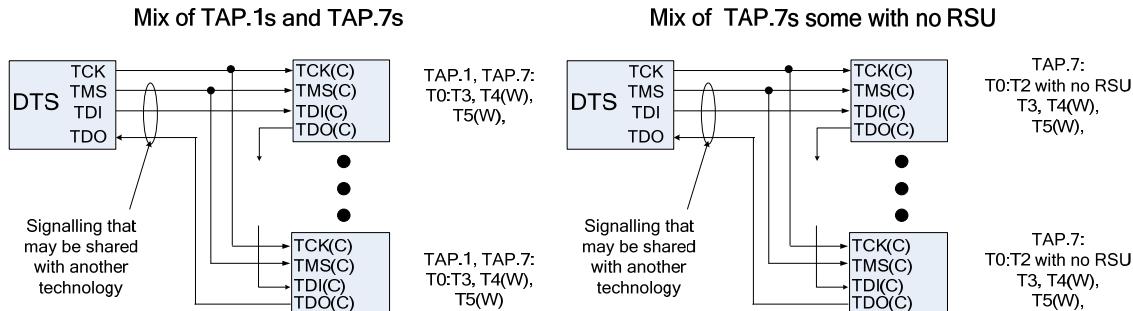
**Table 7-2 — Branch/technology deployment permissibility**

Branch	Proprietary	TAP.1	T0-T2		T3	T4/T5 wide	T4/T5 narrow
			Without RSU	With RSU			
TAP.1 Series	No	Yes	Yes	Yes	Yes	Yes	No
TAP.7 Series	No	No	No	Yes	Yes	Yes	No
TAP.7 Star-4	No	No	No	No	Yes	Yes	Yes
TAP.7 Star-2	No	No	No	No	No	Yes	Yes
Proprietary	Yes	No	No	No	No	No	Yes

## 7.4 TAP.7 deployment scenarios

### 7.4.1 TAP.1 Series Branches

Series branches may be constructed from TAP.1s and four-pin TAP.7s as shown in Figure 7-2. This scan topology may share the TMS, TDI, and TDO connectivity with another topology provided the TCKC is used to select/deselect this topology.



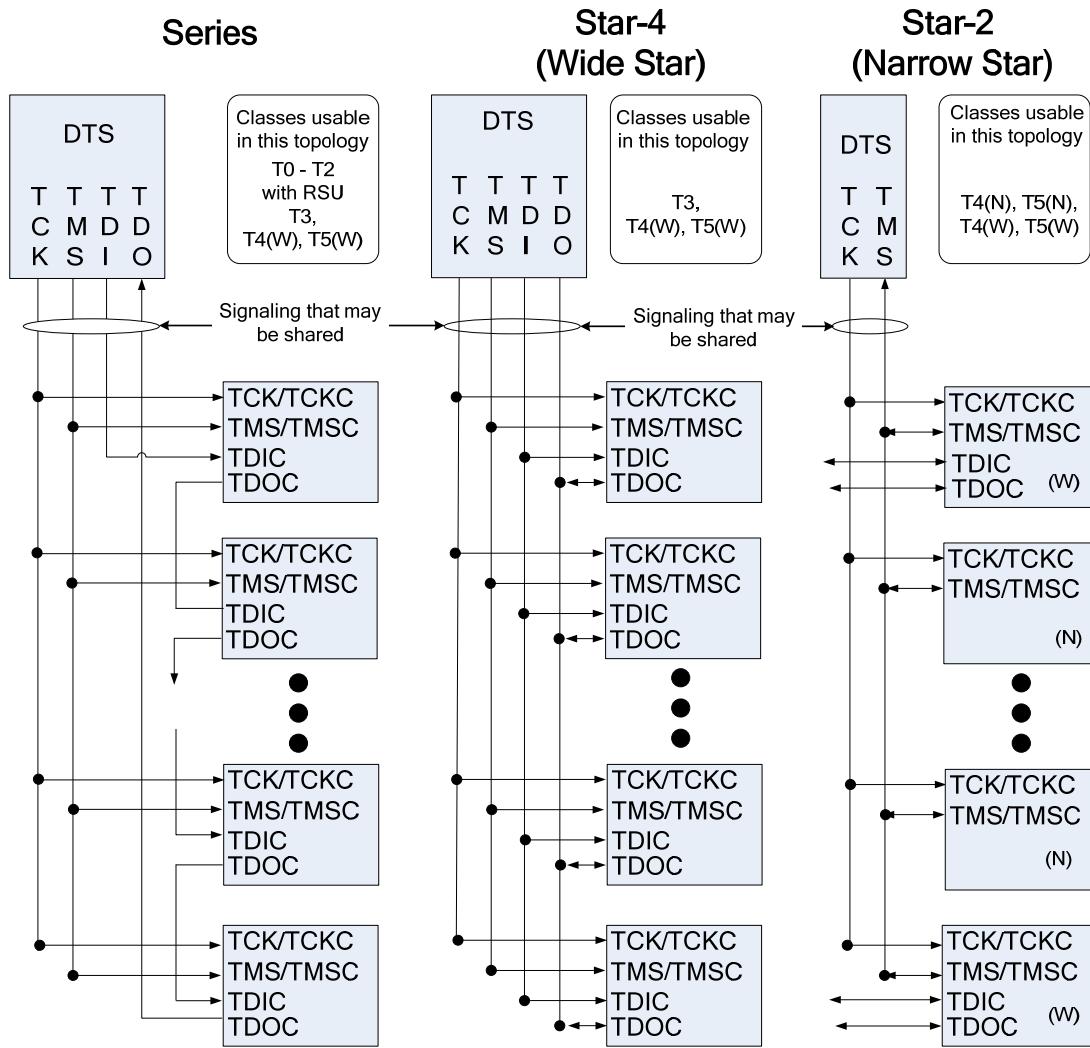
**Figure 7-2 — TAP.1 Series Branches**

Note that the convention for naming DTS TDI/TDIC and DTS TDO/TDOC signals is as follows:

- The DTS TDI/TDIC signal sources chip TDI information
- The DTS TDO/TDOC signal is the destination for chip TDO information

### 7.4.2 TAP.7 Series, Star-4, and Star-2 Branches

In most cases, one or more TAP.7s is connected in one of the scan topologies shown in Figure 7-3.



**Figure 7-3 — TAP.7 Series, Star-4, and Star-2 Scan Topologies**

The Narrow Star Scan Topology is a subset of both the Series and Wide Star Scan Topologies. This means the Series and Wide Star Scan Topologies may be operated as a Star-2 Scan Topology when every Test Access Port sharing the connection is a T4 and above TAP.7.

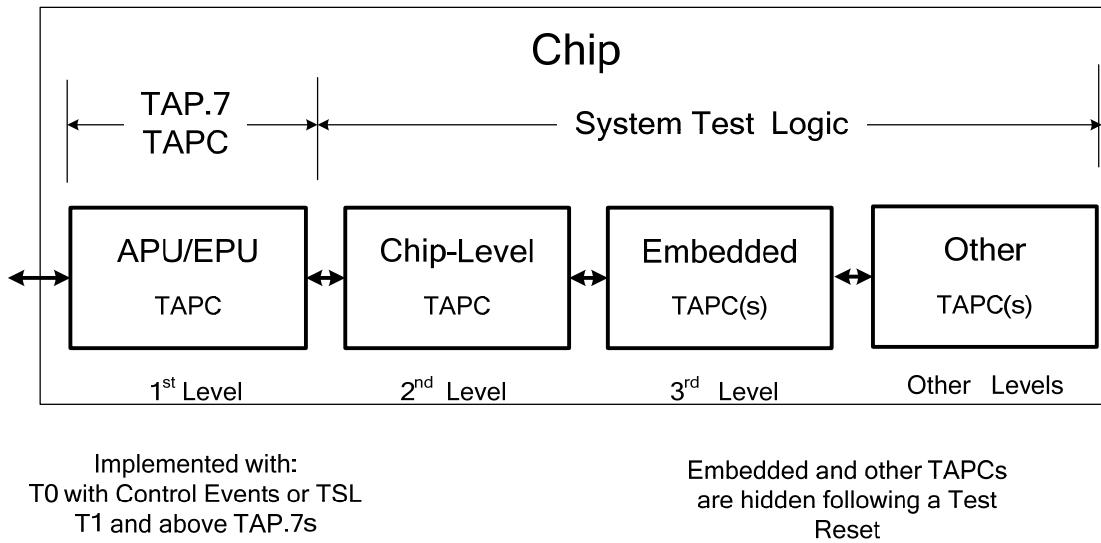
## 7.5 Chip TAPC hierarchy

A hierarchical view of TAPCs within a chip is needed to maintain IEEE 1149.1-Specified Behavior and provide solutions to the requirements identified in Clause 1.

For a given chip, three or more levels of TAPC hierarchy are accommodated as follows:

- 1<sup>st</sup> level      TAPC within the TAP.7 Controller (the ADTAPC)
- 2<sup>nd</sup> level      Chip-Level TAPC within the System Test Logic (the CLTAPC)
- 3<sup>rd</sup> level      TAPCs connected to the Chip-Level TAPC (EMTAPCs)
- Other levels    If needed

Although all levels of the TAPC hierarchy need not be present in all chips, the TAP.7 architecture accommodates them when they occur. The TAPC hierarchy is shown in Figure 7-4.



**Figure 7-4 — TAP.7 TAPC hierarchy**

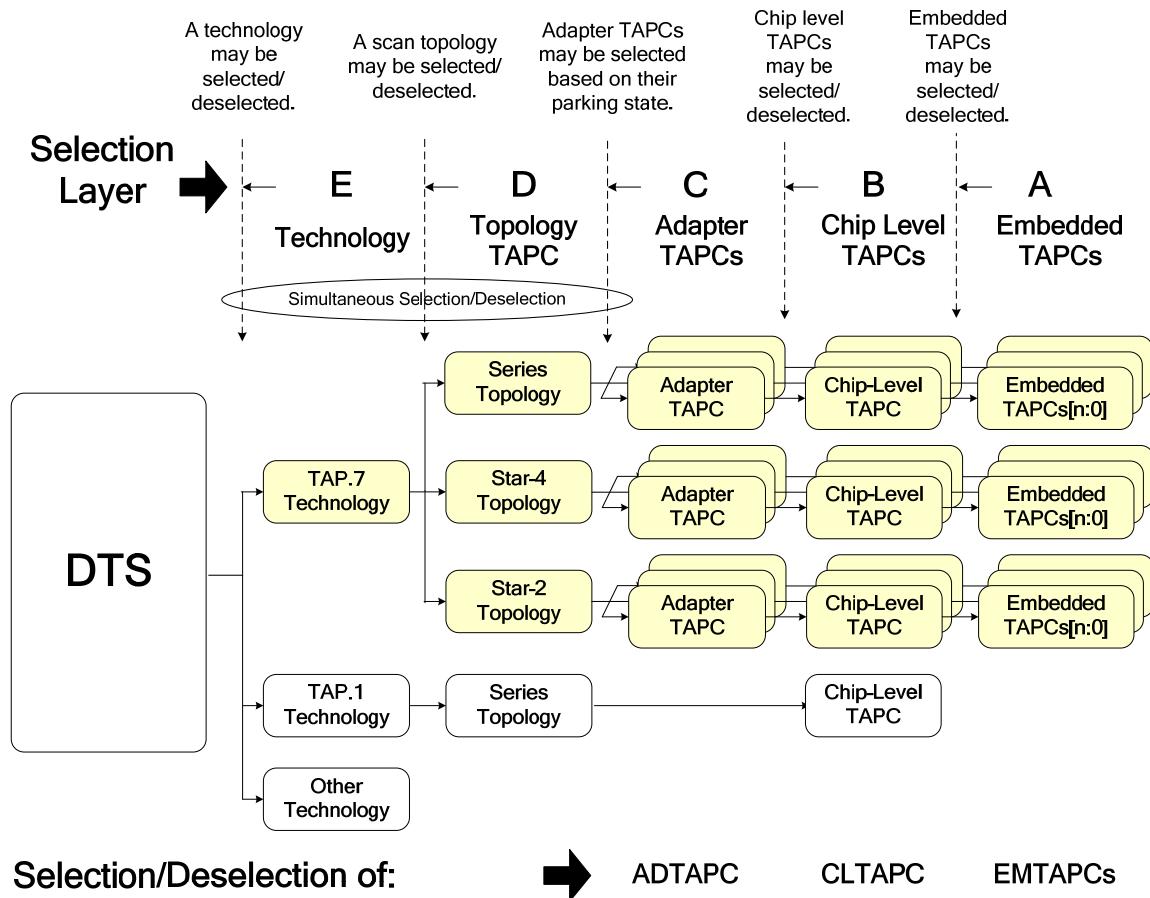
A chip with IEEE 1149.7-Specified Behavior has a Chip-Level TAPC. It may also include one or more EMTAPCs, all of which are considered subordinate to the CLTAPC. The ADTAPC is considered the parent of the CLTAPC, and the CLTAPC is considered the parent of the EMTAPC(s) as stated previously. The addition of the EPU of a T1 and above TAP.7 does not add any Instruction or Data Registers in series or parallel with scan paths associated with the CLTAPC and the EMTAPCs. The addition of the APU in a T4 or T5 TAP.7 does not add an additional TAPC or level of TAP.7 TAPC hierarchy.

The ADTAPC provides access to the CLTAPC. The CLTAPC may provide access to other EMTAPCs. The scan paths associated with these TAPCs provide IEEE 1149.1-Specified Behavior following a Test Reset. This behavior provides the test view of a component specified by IEEE Std 1149.1. The “other TAPCs” portion of the hierarchy can have additional levels of hierarchy. The TAPC hierarchy may vary dramatically by chip, with the TAPs accessed varying by application. Typically, the DTS dynamically manages the TAPC hierarchy in order to:

- Accommodate a Star Scan Topology
- Minimize the length of the scan path
- Avoid parts of the hierarchy that are not functional (e.g., powered-down, etc.)

## 7.6 Combined view of TAP connectivity and TAPC hierarchy

The combination of the system shown in Figure 7-1 and the TAP.7 TAPC hierarchy shown in Figure 7-4 creates the system connectivity and TAPC hierarchy shown in Figure 7-5. The connectivity layers (labeled A–E) are shown this figure. The TAPCs actually selected and deselected are shown at the bottom of this figure.



**Figure 7-5 — System connectivity supported by the TAP.7 architecture**

## 7.7 Chips, components, and boards

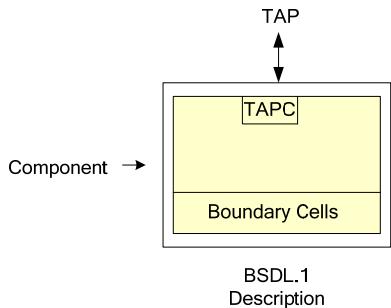
With the TAP.7 architecture, any part of this connectivity shown in Figure 7-5 may be deployed on a board, within a package, or within a chip. This accommodates the packaging and integration trends since the inception of IEEE Std 1149.1.

The IEEE 1149.1 test view allows only one TAPC per component that is placed on a board. The IEEE 1149.7 test view allows multiple TAPCs per component. With the IEEE 1149.7 test view, a component meets the following criteria:

- Is a package that contains one chip, more than one chip, or a direct-mount chip
- Provides access to one or more TAP.1 or TAP.7 Controllers through the same or separate pins
- Provides access to TAPCs with IEEE 1149.7-Other Behavior or other technologies through the same or separate pins

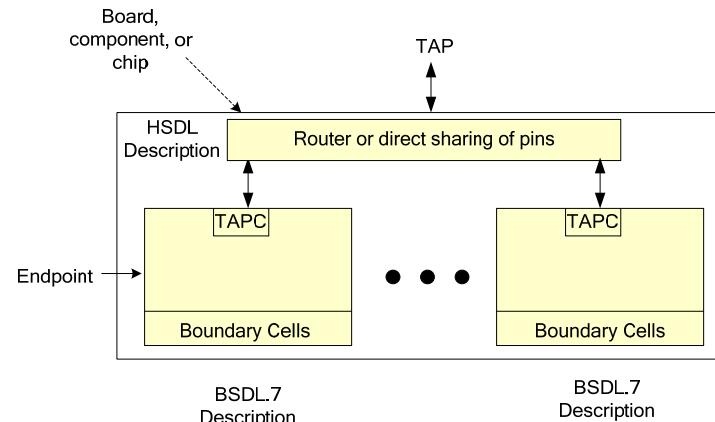
The IEEE 1149.7 test view blurs the boundary between the board, component, and chip. This blurring allows operation across those boundaries as shown in Figure 7-6. When there is only one TAPC per component, the IEEE 1149.1 and IEEE 1149.7 test views are described by Boundary-Scan Description Language (BSDL). When there is more than one TAPC per component, the IEEE 1149.7 test view requires Hierarchical Scan Description Language (HSDL), plus BSDL for each end point.

## 1149.1 Description



BSDL.1  
Description

## 1149.7 Description



BSDL.7  
Description

BSDL.7  
Description

**Figure 7-6 — Contrasting IEEE 1149.1 and IEEE 1149.7 test views**

The description of BSDL and HSDL is provided in Clause 30 and Clause 31, respectively.

## 8. TAPC hierarchy

### 8.1 Introduction

This clause is applicable to T0 and T1 TAP.7s that implement an RSU along with T2 and above TAP.7s. It describes the concepts within the TAP.7 architecture that provide for the selection of TAP.7 TAPCs and other technologies.

The subject matter within this clause is organized in the following manner:

- 8.2 Selection/deselection with the TAPC hierarchy
- 8.3 TAPC selection/deselection characteristics
- 8.4 ADTAPC selection/deselection
- 8.5 CLTAPC selection/deselection
- 8.6 EMTAPC selection/deselection
- 8.7 Using a common selection/deselection protocol across technologies
- 8.8 RSU deployment
- 8.9 Using the TAPC hierarchy
- 8.10 Test/Debug applications and the TAPC hierarchy

### 8.2 Selection/deselection with the TAPC hierarchy

#### 8.2.1 Selection choices

The DTS may interact with all or a subset of the technologies connected to it using the hierarchical selection and deselection capabilities of the TAP.7 architecture. Lower levels of the hierarchy become usable when all levels above it are selected and become unusable when any level of the hierarchy above it is deselected. Selection and deselection is provided at the following levels in the hierarchy:

- Technology branches
- A Series Branch—all ADTAPCs that are associated with the branch
- A Star-4 Branch—all ADTAPCs that are associated with the branch
- A Star-2 Branch—all ADTAPCs that are associated with the branch
- Simultaneous selection of Series, Star-4, and Star-2 Branches—all TAPCs that are associated with each of these branches
- Chip level TAPCs—the CLTAPC that is the child of a selected ADTAPC
- Embedded TAPCs—an EMTAPC that is the child of a selected CLTAPC or EMTAPC

The normal information exchanges generated with the TAP signaling specified by this standard occur within hierarchy levels C-A (see Figure 7-5) when the TAPC state of a TAPC at a specific hierarchy level advances in lock-step with the state of TAPC(s) in the hierarchical level above it (its parent).

The detailed descriptions of TAPC selection can be found in the following clauses:

- Clause 11 for the ADTAPC, as it is applicable to all TAP.7 Classes
- Clause 18 for the EMTAPCs, where their deployment with a T0 TAP.7 is first permitted

- Clause 19 for the CLTAPC, where its deployment is first permitted

### **8.2.2 Selection/deselection/class relationships**

The relationship between the TAP.7 Class and its selection capability is shown in Table 8-1.

**Table 8-1 — TAP.7 Class relationship to selection capability**

<b>Selection of:</b>	<b>Selection layer</b>	<b>TAP.7 Class</b>			
		<b>T0-T1</b>	<b>T2</b>	<b>T3</b>	<b>T4 and above</b>
EMTAPC	A	Optional	Optional	Optional	Optional
CLTAPC	B	N/A	Mandatory	Mandatory	Mandatory
ADTAPC	E-C	Optional	Optional	Mandatory	Mandatory

### **8.2.3 TAPC parent/child relationships**

The selection hierarchy described in 7.5 establishes the parent/child relationships between the levels of the selection hierarchy as shown in Table 8-2.

**Table 8-2 — TAPC parent/child relationships**

<b>Child</b>	<b>ADTAPC present?</b>	<b>Parent</b>
TAP.7 TAPC	N/A	DTS TAPC
CLTAPC	No	DTS TAPC
	Yes	TAP.7 TAPC
EMTAPC	N/A	CLTAPC

Different terminology is used to describe the parent/child relationship at each level of the TAPC hierarchy. Even though the functions may be the same, or very close to the same, different terminology is used so it is clear which parent/child relationship in the TAPC hierarchy is being described.

## **8.3 TAPC selection/deselection characteristics**

### **8.3.1 TAPC and scan path behavior**

The TAPC characteristics produced by selecting and deselecting TAPCs are described as follows:

- ADTAPC:
  - Deselected The ADTAPC state is parked. The TAP.7 Controller's scan paths are not accessible.
  - Selected The ADTAPC TAPC state advances in lock-step with the DTS. The TAP.7 Controller Scan Paths are accessible.
- CLTAPC:
  - Deselected The CLTAPC state is parked. The STL Scan Paths are not accessible.
  - Selected The CLTAPC state advances in lock-step with the TAP.7 TAPC state. The STL Scan Paths are accessible.

- EMTAPC:
  - Deselected The scan paths associated with the EMTAPC are not accessible. The TAPC state is parked.
  - Selected The scan paths managed by the EMTAPC Scan Paths are accessible. The TAPC state advances in lock-step with the TAPC state of its parent. The DR Scan Paths may be excluded in some cases, depending on the STL scan architecture.

### 8.3.2 Selection/deselection mechanisms

The selection mechanisms for layers C, B, and A (see Figure 7-5) of the selection hierarchy are listed as follows (note that they are different for each level of the selection hierarchy):

- ADTAPC Selection/Deselection Escapes or optionally Selection Alerts + subsequent Control Protocol information defining the selection criteria (TAP.7 technology, Scan Topology, TAPC state) following Selection Escapes/Selection Alerts
- CLTAPC Two mechanisms, depending on the TAP.7 Class:
  - T2–T5 TAP.7s Register information applied with the *Run-Test/Idle* state
  - T3–T5 TAP.7s SSDs
- EMTAPCs Information provided with scans applied with the *Run-Test/Idle* state

The detailed description of the selection and deselection of mechanisms for these TAPCs is deferred to Clauses 11, 16, and 19, respectively.

### 8.3.3 Parking states and resynchronization

The TAPC parking state is the state that exists when its advance is suspended. The value of this state is important as it defines the only TAPC state where the TAPC's operation may be resynchronized to the operation of the DTS and other TAPCs. When the TAPC is parked, it may be synchronized with the DTS TAPC state only when the DTS and entity's TAPC state are the same or can be made the same by a stay in the *Run-Test/Idle* state (the parked state is *Test-Logic-Reset*).

The selection and deselection states for the TAPCs in the selection hierarchy are shown in Table 8-3. When an entity's test reset is asserted while its TAPC state is parked, the TAPC state is set to *Test-Logic-Reset* with this state becoming the parking state.

**Table 8-3 — Selection and deselection states**

<b>Entity</b>	<b>Deselection permitted in:</b>	<b>Selection permitted in:</b>	<b>A test reset while deselected sets the TAPC state to:</b>
EMTAPC	<i>Run-Test/Idle</i>	<i>Run-Test/Idle</i>	<i>Test-Logic-Reset</i>
CLTAPC	<i>Pause-DR, Pause-IR, Select-DR-Scan, Run-Test/Idle</i>	<i>Pause-DR, Pause-IR, Run-Test/Idle, Test-Logic-Reset</i>	<i>Test-Logic-Reset</i>
ADTAPC	Any state	<i>Pause-DR, Pause-IR, Select-DR-Scan, Run-Test/Idle, Test-Logic-Reset</i>	<i>Test-Logic-Reset</i>

## 8.4 ADTAPC selection/deselection

### 8.4.1 Parking use cases

ADTAPC selection (parking of the ADTAPC state) supports the following use cases:

- Sharing the TCK(C) and TMS(C) signals with other technologies (IEEE 1149.1 and/or legacy/proprietary technologies).
- Multiple TAP.7 scan topologies operated together.
- The interoperability of T4 and above TAP.7 Controllers with different feature sets.
- Debug with chips where the entire chip may be powered-down and debug continues with other chips sharing the DTS connection. This includes the power-up of a TAP.7 Controller while other TAP.7 Controllers are operating and its subsequent synchronization to the operation of other TAP.7 Controllers.

Deselection of an ADTAPC is required to operate it with other technologies (IEEE 1149.1 and/or legacy/proprietary technologies). When sharing the TAP.7 signals with other functions, the ADTAPC operate in a manner where normal TCK(C) and TMS(C) signaling is ignored. It is also required to operate T4 and above TAP.7 Controllers in a manner where a TAP.7 Controller remains interoperable with other TAP.7s when capabilities affecting the Advanced Protocol are enabled but not supported.

In multi-chip power-conscious systems, entire chips may be powered-down. Any combination of chips may be powered-up and powered-down during normal operation. The ability to debug systems with these attributes is needed since this system behavior is part of normal system operation. The TAP.7 architecture provides for deselecting the ADTAPC at start-up (placing the ADTAPC Offline-at-Start-up). It may be subsequently selected after power-up when criteria preventing its erroneous selection are met. Its selection synchronizes its operation to the DTS and other already operating ADTAPCs.

ADTAPC selection is optional with a T0–T2 TAP.7s and mandatory with T3 and above TAP.7s. Clause 11 provides a detailed description of the parking of the ADTAPC and the Online/Offline operation of the TAP.7 Controller.

The selection and deselection of the ADTAPC is also referred to as Online/Offline capability. This capability is covered in detail in Clause 11.

### 8.4.2 DTS/ADTAPC relationship

The DTS (the parent) determines whether the ADTAPC (its child) participates in scan operations in which it also participates. The method used to select a child allows the selection of a specific TAP.7 Technology Branches or all TAP.7 Technology Branches provided the TAPC state matches the state specified by the

DTS. With T4 and above TAP.7s using the Advanced Protocol, an individual TAP.7 Controller deselects itself when directed to perform an operation that would cause it to lose synchronization with the DTS and other TAPCs (an unsupported feature affecting the Advanced Protocol is enabled).

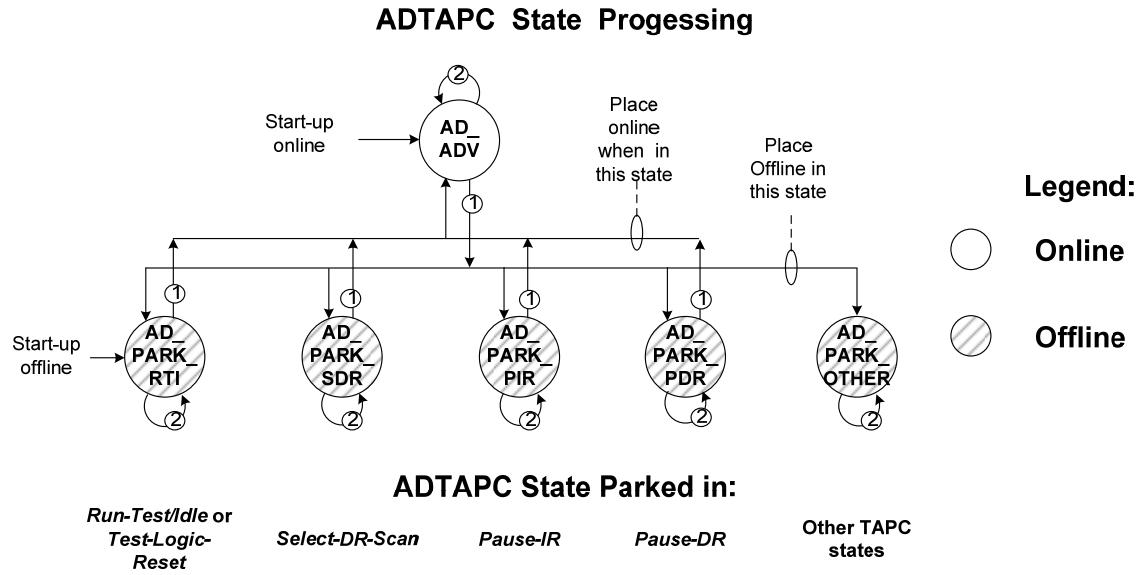
#### 8.4.3 ADTAPC operation

The parking of the Adapter TAPC state is shown in Figure 8-1. The states in this figure are described briefly below:

- *AD\_SCAN* The ADTAPC operates in lock-step with the DTS and the ADTAPCs of other TAP.7 Controller within this state.
- *AD\_PARK\_RTI* The ADTAPC state is parked in either the *Run-Test/Idle* or the *Test-Logic-Reset* state.
- *AD\_PARK\_SDR* The ADTAPC state is parked in the *Select-DR* state.
- *AD\_PARK\_PIR* The ADTAPC state is parked in the *Pause-IR* state.
- *AD\_PARK\_PER* The ADTAPC state is parked in the *Pause-DR* state.
- *AD\_PARK\_OTHER* The ADTAPC state is parked in a state other than the following:
  - *Test-Logic-Reset*
  - *Run-Test/Idle*
  - *Select-DR*
  - *Pause-IR*
  - *Pause-DR*

Parking of the ADTAPC state is an optional feature with T0–T2 TAP.7s and mandatory with T3 and above TAP.7s. The parking state created by a Deselection Escape depends on when it occurs relative to the ADTAPC state. Deselection Alerts (when implemented) park the ADTAPC state in the *Run-Test/Idle* states. Selection Escapes and Selection Alerts (when implemented) activate the ADTAPC's state progression without a reset of the TAP.7 Controller, provided the Control Protocol following these events permits this (see Clause 11 for more detail).

A Deselection Escape may park the ADTAPC in any state. It is important to note that the Control Protocol following Selection Escapes or Selection Alerts may only place the ADTAPC Online provided the TAPC state is *Test-Logic-Reset*, *Run-Test/Idle*, *Select-DR-Scan*, *Pause-IR*, and *Pause-DR*, as it specifies the technology to be selected (TAP.7 or other), the TAP.7 Branch being selected (when TAP.7 technology is selected), and the ADTAPC state required (the ADTAPC state). The TAP.7 compares its Scan Topology (derived via Scan Topology Training) and the current ADTAPC state (when operating or parked) to the values required for selection of the ADTAPC. When these values are equal the ADTAPC is placed Online. The ADTAPC is placed Offline otherwise. The selection process provides for the simultaneous selection one TAP.7 Branch and deselection of other branches, the simultaneous selection of all branches, and the simultaneous deselection of all branches. When the ADTAPC is parked in a state other than these states, a TAP.7 Controller reset is required to place the ADTAPC Online.



**Figure 8-1 — Parking the ADTAPC state**

## 8.5 CLTAPC selection/deselection

### 8.5.1 Parking use cases

CLTAPC selection (parking of the CLTAPC state) supports the following use cases:

- Bypass of Chip-Level TAPCs in a Series Scan Topology to improve performance
- Creation of Series-Equivalent Scans in a Star Scan Topology

CLTAPC selection provides a means of improving TAP.7 scan performance in a Series Scan Topology and providing Series-Equivalent Scans in Star Scan Topologies. Parking of the CLTAPC state bypasses the STL in both Series and Star Topologies (see 4.2.3.3.2).

### 8.5.2 ADTAPC/CLTAPC relationship

The ADTAPC (the parent) determines whether the CLTAPC (its child) participates in scan operations in which it also participates. The participation of a CLTAPC in these scan operations is described as the functionality of the TAPC and the scan paths it controls when viewed from the embedded TAP signals.

The DTS selects a CLTAPC when it desires access to the STL's IR and DR Scan Paths. With series selection approaches, the DTS deselects a CLTAPC when it has no desire to access the IR and DR Scan Paths managed by the CLTAPC.

### 8.5.3 CLTAPC operation

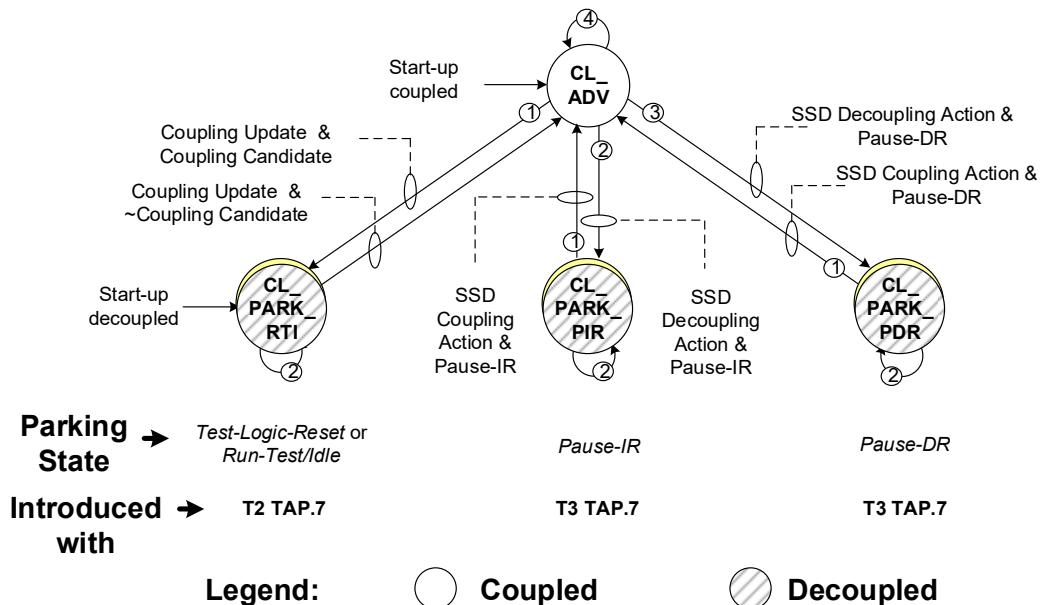
The parking of the CLTAPC state is shown in Figure 8-2. Certain conditions managed by the TAP.7 Controller deselect the Chip-Level TAPC in one of the states listed for the 2<sup>nd</sup> level in Table 4-1. The state names shown in Figure 8-2, *CL\_ADV*, *CL\_PARK\_RTI*, *CL\_PARK\_PIR*, and *CL\_PARK\_PDR* play an important role in describing the operation of the TAP.7 in both the Series and the Star Scan Topologies as they are directly related to group membership (Scan, Idle, *Pause-IR*, and *Pause-DR*). These states are described briefly as follows:

- *CL\_ADV* (Chip-Level Advance) The CLTAPC operates in lock-step with the ADTAPC. The STL is a member of the Scan Group.
- *CL\_PARK\_RTI* (Chip-Level Park in *RTI*) The CLTAPC state is parked in either the *Run-Test/Idle* or the *Test-Logic-Reset* state. The STL is a member of the Idle Group.
- *CL\_PARK\_PIR* (Chip-Level Park in *Pause-IR*) The CLTAPC state is parked in *Pause-IR*. The STL is a member of the Pause-IR Group.
- *CL\_PARK\_PDR* (Chip-Level Park in *Pause-DR*) The CLTAPC state is parked in *Pause-DR*. The STL is a member of the Pause-DR Group.

Two mechanisms, TAP.7 controller commands and SSDs, in combination with a TAPC state manage transitions between these states as described previously.

With a T0–T1 TAP.7, the CLTAPC state is never parked. Parking of the CLTAPC state is a mandatory feature with a T2 and above TAP.7, with parking of the CLTAPC permitted in either the *Test-Logic-Reset* or the *Run-Test/Idle* states. Transitions between the *CL\_ADV* and *CL\_PARK\_RTI* states occur when the *Run-Test/Idle* state occurs. The *Test-Logic-Reset* state creates either the *CL\_ADV* or the *CL\_PARK\_RTI* state depending on the start-up option.

With a T3 and above TAP.7s, the CLTAPC controller may be also be parked in the *Pause-IR* and *Pause-DR* states, provided operation with a Star Scan Topology is specified. This capability provides for the creation of Series-Equivalent Scans in a Star Scan Topology (see 5.5.3.5). SSDs are added with T3 and above TAP.7s to provide for managing the parking of the CLTAPC in the *Pause-IR* and *Pause-DR* states. SSDs used with the *Run-Test/Idle* state also create transitions between the *CL\_ADV* and *CL\_PARK\_RTI* states.



**Figure 8-2 — Parking the Chip-Level TAPC state**

The TAP.7 Controller tracks the parking state of both its CLTAPC and the CLTAPC of other TAP.7 Controllers. This information is used to prevent TDOC and TMSC drive conflicts when operating in a Star Scan Topology.

As a general rule, the parking state of the CLTAPCs of the TAP.7s within a technology branch may be a mix of *CL\_PARK\_RTI*, and either *CL\_PARK\_PIR* or *CL\_PARK\_PDR*. At no time can the CLTAPCs of the TAP.7s within a technology branch have a mix of *CL\_PARK\_PIR* and *CL\_PARK\_PDR* states. Certain operations are interlocked to ensure that TMSC and TDOC drive conflicts do not occur.

## 8.6 EMTAPC selection/deselection

### 8.6.1 Parking use cases

EMTAPC selection provides a means to deal with chip conditions that may make EMTAPCs inoperable. An example of this is the power-down of the Chip-Level Logic containing an EMTAPC. In this case, the TAPC State Machine state and its scan paths will be removed from the CLTAPC's Scan Path before they are powered-down and added to the STL's Scan Path when their use is required and they are powered. Both of these operations will be under DTS control so it understands the Instruction Register and Data Register Scan Path configurations at all times. Deselection of TAPCs that are of no interest also improves scan performance.

### 8.6.2 CLTAPC/EMTAPC relationship

The CLTAPC (the parent) determines whether the EMTAPCs (its children) participate in scan operations in which it participates. It can deny the participation of all its children or combinations of its children. This is determined by the STL implementation. The participation of an EMTAPC in these scan operations is described as the functionality of the TAPC and the scan paths it controls when viewed from the embedded TAP signals.

### 8.6.3 EMTAPC operation

The parking of an EMTAPC state is shown in Figure 8-3. The states shown in this figure are described briefly below:

- *EM\_ADV* Embedded Advance—The EMTAPC operates in lock-step with the CLTAPC.
- *EM\_PARK* EM Park *Run-Test/Idle*—The EMTAPC state is parked in either the *Run-Test/Idle* or the *Test-Logic-Reset* state.

The EMTAPC is called “Normal” or “Excluded” when the ADTAPC state is not parked and “Isolated” when the ADTAPC state is parked. These terms define a combination of the following, whether:

- The EMTAPC state is parked.
- The DR Scan paths managed by the EMTAPC are accessible.

The conditions controlling the transitions between the *EM\_ADV* and *EM\_PARK* states are constrained by this specification, with the precise implementation determined by the chip architect. This approach accommodates many approaches for managing EMTAPCs developed prior to this specification. The operation of EMTAPCs is described further in Clauses 15 and 16.

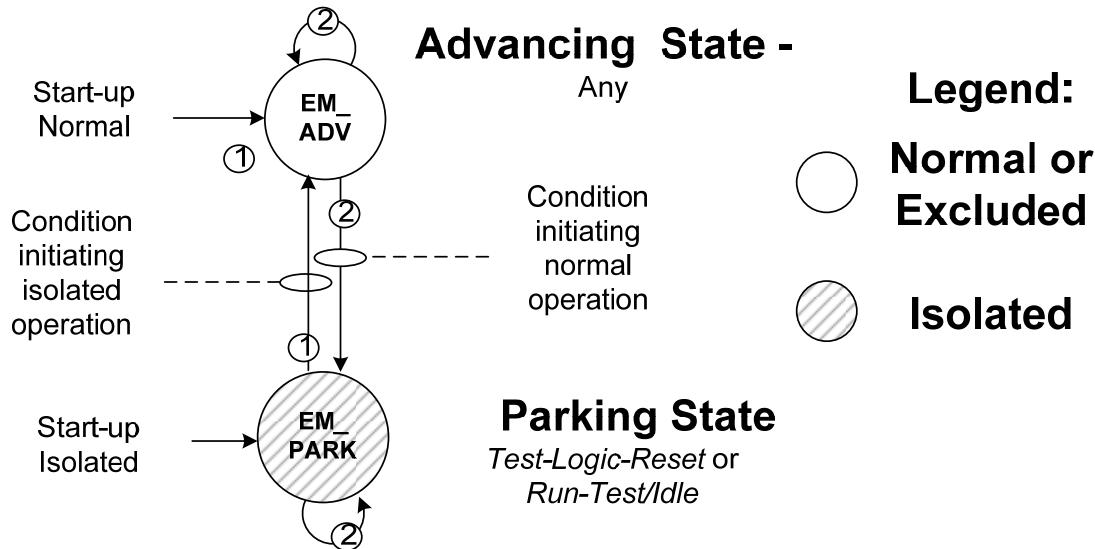


Figure 8-3 — Parking the Chip-Level TAPC state

## 8.7 Using a common selection/deselection protocol across technologies

Selection/deselection interoperability with proprietary, legacy, and TAP.7 technologies requires the use of a common set of signaling conventions for these functions. These signaling conventions cannot already be utilized by any of these technologies.

With the TAP.7 architecture, the following two options for the signaling convention are provided:

- Selection/Deselection Escapes
- Selection/Deselection Alerts

Selection and Deselection Escapes are a subset of the Escape sequences described in 4.3.1. The escape signaling space is available when the DTS sources the TCK(C) signal as it is capable of creating these signaling conventions. It is unavailable when the TS sources the TCK(C) signal because in this case, the DTS cannot stop the TCKC signal at a logic 1 to create these signaling conventions.

Selection Alerts use a long sequence of data bits in a specific pattern that is *extremely* unlikely to occur during any other debug or functional use of the interface. This signaling space is available independently of whether the DTS or TS sources the TCK(C) signal. It is available only when all the technology entities sharing a connection are Offline.

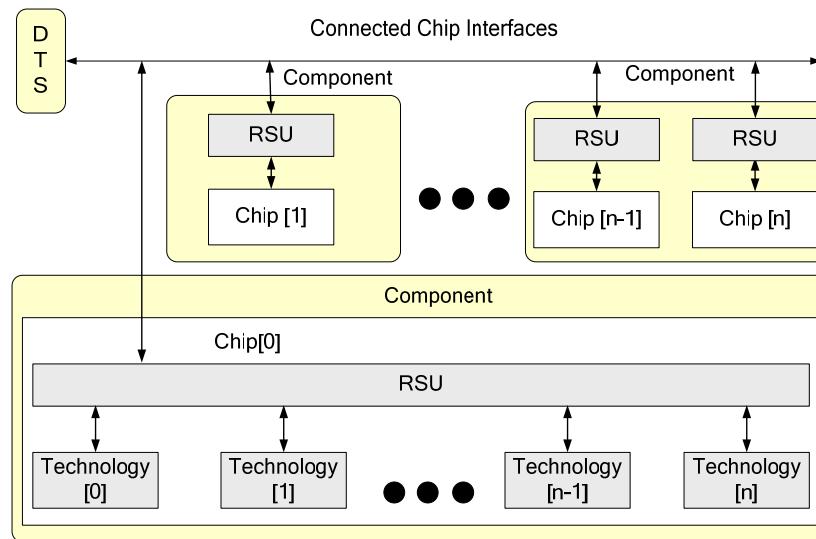
## 8.8 RSU deployment

### 8.8.1 Use with new or existing IP

The RSU uses the common signaling concept outlined above. It may be added to a new IP or an existing/hardened IP as a wrapper. The RSU is as follows:

- Is placed between this IP and the pins normally connected to a DTS
- Allows the sharing of the chip clock and test pins by multiple technologies on the same chip
- Provides for the shared connectivity of these pins with chips deployed with one or more technologies (both debug and functional)

The RSU with a system is shown in Figure 8-4. The internals of each chip may be similar to those of Chip [0].



**Figure 8-4 — Selection within a chip or system**

### 8.8.2 Using TAP pins for multiple functions

The RSU functionality provides for the sharing of interface pins between debug communication and functional uses. The RSU functionality may allow a device to be built with no incremental package pin-count cost for providing the debug communication interface when both the debug interface and the functional interface within the same device share pins. The RSU functionality may allow equipment to be built without incurring the cost of a debug communication connector when the functional interface includes a board-level connector that can be used for debug and functional purposes.

## 8.9 Using the TAPC hierarchy

### 8.9.1 Selection considerations

The DTS's management of the TAP.7 Controllers sharing a DTS connection may produce the combinations of the following conditions:

- The TAPC state of one or more TAP.7 Controllers may be parked in any state.
- Parking states may differ across the TAP.7 Controllers whose states have been parked.

Because the parking states of TAP.7 Controller A and B may be different, the DTS will supply the following to place a TAP.7 Controller Online:

- The technology being selected (TAP.7)
- The scan topology required for selection (Series, Star-4, Star-2, or any of these)
- The TAPC state which is required to synchronize the operation of the TAP.7 Controller with the DTS and other operating TAPCs (*Test-Logic-Reset/Run-Test/Idle*, *Select-DR-Scan*, *Pause-DR*, or *Pause-IR*)

The selection mechanism is capable of placing an ADTAPC Online only when its parking state is one of the states listed above. With other parking states, the TAP.7 Controller will remain Offline until a TAP.7 Controller reset creates the *Test-Logic-Reset* state.

Because the selection criterion includes the scan topology, a command sequence is provided to establish the scan topology in which a TAP.7 Controller is deployed.

### **8.9.2 Start-up considerations**

After the power-up of the target system none, one, or more than one (preferably one or none) of the technologies is provided to use the chip pins as inputs. In systems and chips where an RSU provides for the shared use of the pins by both functional and debug logic, it is recommended that the debug logic be placed Offline-at-Start-up. A technology that is Offline-at-Start-up does not drive the pins until properly stimulated by externally sourced signaling. It should use a keeper function for the pin when it is placed Offline. The DTS may select the technology of interest once it is connected to the system.

## **8.10 Test/debug applications and the TAPC hierarchy**

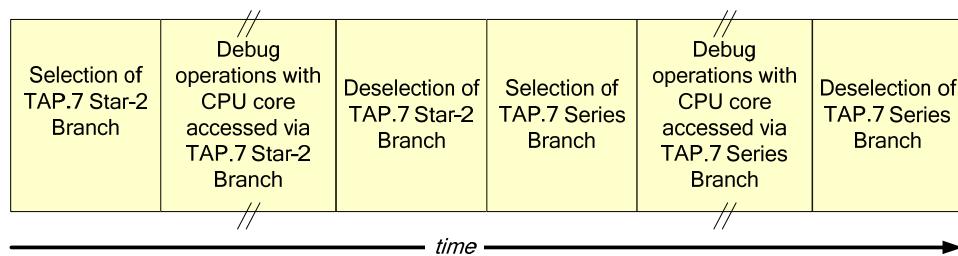
The use of technologies sharing the DTS connection is generally different for test and debug.

### **8.10.1 Debug use of the TAPC hierarchy**

Debug is likely to utilize the TAPC hierarchy in the following manner:

- Use technologies one at a time
- Use a technology for a period of time
- Perform multiple IR and DR Scans or other operations while using a technology
- Select and deselect scan-based technologies in either the *Run-Test/Idle* or the *Select-DR-Scan* state

An example of debug use of the Star-2 and TAP.1 Series Branches of system connectivity is shown in Figure 8-5. In this example, the state of a central processing unit (CPU) core is accessed through an EMTAPC in the Star-2 Branch followed by the same operation with a different CPU in the Series Branch.



**Figure 8-5 — Example of debug use of the RSU**

The example shown in Figure 8-5 begins with the two branches deselected with their TAPC state parked in the *Run-Test/Idle* state. Subsequently, the branches are sequentially used with the following steps:

- Select the Star-2 Branch by specifying these selection criteria (ADTAPC technology, Star-2, *Run-Test/Idle*)
- Operate with this branch
- Advance the TAPC state to *Run-Test/Idle*
- Deselect the Star-2 Branch

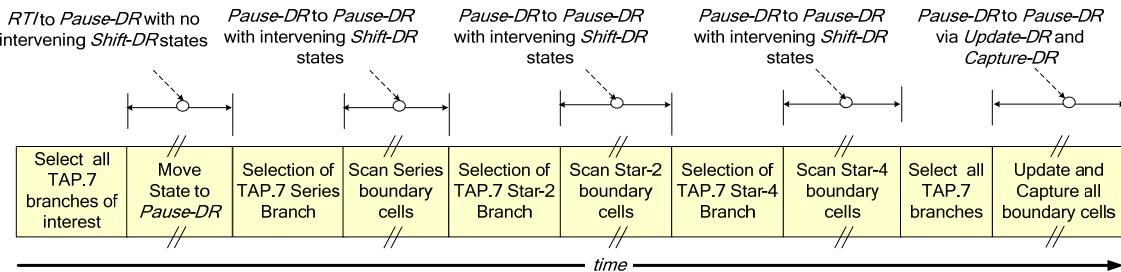
- Select the TAP.7 Series Branch
- Operate with this branch
- Deselect this branch
- Select the Star-2 Branch by specifying these selection criteria: ADTAPC technology, Star-2, and *Run-Test/Idle*

### 8.10.2 Test use of the TAPC hierarchy

Test is likely to utilize the TAPC hierarchy and the TAPC selection it provides to create Series-Equivalent Scans spanning multiple or all technologies sharing the DTS connection as shown in Figure 8-6. When creating this type of Series-Equivalent Scan, the following constraints apply:

- Most selections are performed while in the *Pause-DR* and *Pause-IR* states.
- Shift operations are performed with one branch at a time.
- Capture and update operations are performed with all scan technologies of interest (Series, Star-4, and Star-2) selected.

In the example shown in Figure 8-6, a Series-Equivalent Scan is created with the Series, Star-4, and Star-2 Branches sharing the DTS connection. The boundary cells of each branch are scanned sequentially as shown. Once this is complete, all the branches are simultaneously selected (this is supported in the selection criteria). The state is simultaneously moved from *Pause-DR* through the *Update-DR* and *Capture-DR* followed by *Pause-DR* without an intervening *Shift-DR* state. This is the same process used to create Series-Equivalent Scans for Star-4 and Star-2 Scan Topologies.



**Figure 8-6 — Using Series/Star-4/and Star-2 Scan Topologies concurrently**

Once a branch is selected, the methods used to create Series-Equivalent Scans within the technology may be used. All operations described in 5.5.3.4 can be created spanning multiple technologies. Any technology of interest may be selected or deselected in any of the parking states described in 8.3.3.

A description of the activity shown in Figure 8-6 follows. This example begins with each technology deselected with its TAP.7 TAPC state parked in the *Run-Test/Idle* state. Subsequently, a Series-Equivalent Scan is created with the following steps:

- Select the branches of interest with these criteria: ADTAPC technology, All, and *Run-Test/Idle*.
- Move the state as follows: *Run-Test/Idle* > *Select-DR-Scan* > *Capture-DR* > *Exit1-DR* > *Pause-DR*. All branches of interest are now in the *Pause-DR* state.
- Select the Series Branch.
- Scan the boundary cells in the Series Branch ending with the TAPC State Machine state in *Pause-DR* without moving through *Update-DR*. The boundary cells of this branch are scanned but without capture or update operations.

- Select the Star-2 Branch by specifying these selection criteria: ADTAPC technology, Star-2, and *Pause-DR*.
- Use SSDs to scan the boundary cells of all CLTAPCs connected to TAP.7 TAPCs in the Star-2 Branch ending with the TAPC State Machine state in *Pause-DR* without moving through *Update-DR*. The boundary cells of this branch are scanned but without capture or update operations.
- Select the Star-4 Branch by specifying these selection criteria: TAP.7 TAPC technology, Star-4, *Pause-DR*.
- Use SSDs to scan the boundary cells of all CLTAPCs connected to the TAP.7 TAPCs in the Star-4 Scan Topology ending with the TAPC State Machine state in *Pause-DR* without moving through *Update-DR*. The boundary cells of this branch are scanned but without capture or update operations.
- Select all branches by specifying these selection criteria: ADTAPC technology, all topologies, and *Pause-DR*.
- Move the state as follows: *Pause-DR* > *Update-DR* > *Select-DR-Scan* > *Capture-DR* > *Exit1-DR* > *Pause-DR*.

Repeat this sequence beginning with selection of the Series Branch (Step 3) for additional scans.

## 9. Registers, commands, and scan paths

### 9.1 Introduction

This clause is applicable to T1 and above TAP.7s. It continues the description of the TAP.7 Controller commands and registers begun in 5.3.3.5 and 5.3.3.6. It also includes a description of the TAP.7's Scan Paths. The Commands, Registers managed by these commands, and their formats are described first. This is followed by a description of the scan paths within the EPU. A description of how commands use these scan paths follows. Certain registers supporting the operation of a T5 TAP.7 are managed with directives. The description of these registers and the directives managing them is deferred to Clause 27.

The subject matter within this clause is organized in the following manner:

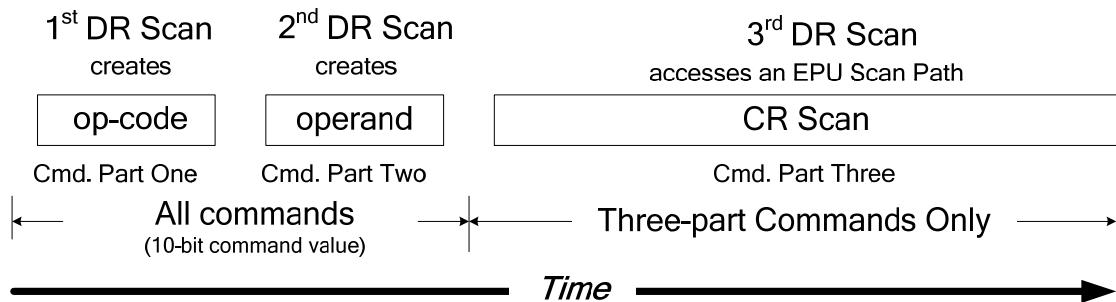
- 9.2 Command basics
- 9.3 Register portfolio
- 9.4 Command portfolio
- 9.5 Representation of commands in examples
- 9.6 Global and Local Register programming with commands
- 9.7 Scan paths
- 9.8 Two-part commands
- 9.9 Three-part commands
- 9.10 Register Read-Back and Configuration Registers
- 9.11 An approach to implementing command processing and scan paths

### 9.2 Command basics

Recall that TAP.7 Controller commands are 10-bit values created with two DR Scans. They may have two or three parts. The command is decoded when the *Update-DR* state of the second DR Scan is reached. With three-part commands, CP1 and CP2 are followed by a CR Scan that moves data to or from a scan path within the EPU.

A CR Scan is a minimum of zero bits in length with there being no maximum length. The scan path for a CR Scan is an EPU Bit, String, or Enumerate Path. These scan paths are shown in Figure 9-5. Two-part and three-part commands may be issued in any combination and in any number before the Command Control Level is exited. If the Command Control Level is exited before a two- or three-part command is completed, the command is aborted and becomes a no-operation.

The command format shown in Figure 5-8 is repeated in Figure 9-1 for reference purposes.



**Figure 9-1 — DR Scan sequence used for command creation**

The following example shows a two-part command. The example starts from a ZBS count of zero and finishes by restoring a ZBS count of zero.

- ZBS (2) Prepare for the locking of the Command Control Level
- DR Scan Lock the control level; the shift count is any value, then *Update-DR*
- DR Scan one CP1 *Shift-DR* state count is the five MSBs of the command, then *Update-DR*
- DR Scan two CP2 *Shift-DR* state count is the five LSBs of the command, then *Update-DR*
- *Select-DR-Scan*
- *Select-IR-Scan* Causes an exit from the Command Control Level

The following example shows a two-part command followed by a three-part command. The example starts from a ZBS count of zero and finishes by restoring a ZBS count of zero.

- ZBS(2) Prepare for locking of the Command Control Level
- DR Scan Lock the control level; the shift count is any value, then *Update-DR*
- DR Scan one CP1 *Shift-DR* state count is the five MSBs of the 1st command, then *Update-DR*
- DR Scan two CP2 *Shift-DR* state count is the five LSBs of the command, then *Update-DR*
- DR Scan one CP1 *Shift-DR* state count is the five MSBs of the command, then *Update-DR*
- DR Scan two CP2 *Shift-DR* state count is the five LSBs of the command, then *Update-DR*
- DR Scan three CR Scan, where the EPU Scan Path is defined by CP2, then *Update-DR*
- *Select-DR-Scan*
- *Select-IR-Scan* Causes an exit from the Command Control Level

Hereafter, the scan sequences creating two- and three-part commands are described as Command Part One (CP1) followed by Command Part Two (CP2), followed by a Control Register (CR) Scan, if the command is a three-part command. In addition, a reference to CP1, CP2, and a CR Scan implies a DR Scan.

## 9.3 Register portfolio

### 9.3.1 Description

#### 9.3.1.1 Global and Local Registers

The description of registers in 5.3.3.6 identified two types of TAP.7 Controller registers, Global and Local. The concept of Global and Local Registers is reviewed briefly in the following section.

A Global Register is a register whose value is stored at the same time and with the same value as registers with the same name in all ADTAPCs sharing a DTS connection. Global Registers manage the ADTAPC functions that affect the synchronized operation of the ADTAPCs of TAP.7 Controllers sharing the DTS connection. Local Registers are conditionally stored using the TAP.7 Controller commands dedicated for this purpose. These commands provide for changing the value of these registers with a unique value in a Series Scan Topology based on the TAP.7 Controller's position on the scan chain. Local Registers are stored with a unique value in a Star Scan Topology based on an address assigned to the TAP.7 Controller.

#### 9.3.1.2 Register loads

Global Register values are changed in one of three ways, as follows:

- With a Global Register load
- With a TAP.7 Controller command
- With a Transport Directive embedded in the Advanced Protocol stream (only certain Transport Registers)

Local Register values are changed in one of two ways, as follows:

- With a TAP.7 Controller command
- With a Transport Directive embedded in the Advanced Protocol stream (only certain Transport Registers)

Global Registers are loaded when a Selection Sequence is initiated by either a Selection Escape or Selection Alert (see 11.7). A Selection Sequence provides for two types of Global Register loads, as follows:

- Only the register defining the scan format is stored
- All Global Registers are either stored or initialized

The registers programmed with commands are described herein. Transport Registers programmed with directives are described in Clause 27.

The TAP.7 Controller register portfolio programmed by commands is covered briefly to provide background information for a subsequent description of commands. There are 16 mandatory and 10 optional registers. These registers are classified as one of the register types listed below:

- Control      TAP.7 Controller behaviors and characteristics
- Options     Control of optional functions
- Select      Controls the selection of a TAP.7 Controller for scan and execution of conditional commands
- Enumerate CID allocation and de-allocation

- Read-only Configuration and register read-back

Table 9-1 provides a brief description of the registers along with their classification as mandatory or optional. The deployment of TAP.7 features is governed by Rule in 9.4.2 k) and Permission 9.4.2 l).

**Table 9-1 — TAP.7 Controller register list managed commands**

Register type	Global /Local	Width	Register mnemonic	Name	TAP.7 Class				
					1	2	3	4	5
Control	Global	1	ECL	Exit Control Level	M	M	M	M	M
		1	SUSPEND	Suspend	M	M	M	M	M
		1	ZBSINH	ZBS Detect Inhibit	M	M	M	M	M
		5	SCNFMT	Scan Format	--	M	M	M	M
		1	SSDE	SSD Enable	--	--	M	M	M
		2	DLYC	Delay Control	--	--	--	M	M
		2	RDYC	Ready Control	--	--	--	M	M
		5	TPST	TP state Enables	--	--	--	--	M
		2	TPPREV	Transport Protocol Rev	--	--	--	--	M
Options	Local	2	TOPOL	Topology	--	--	M	M	M
		1	FRESET	Functional Reset	O	O	O	O	O
		1	TRESET	Test Reset	O	O	O	O	O
		2	PWRMODE	Power-Control Modes	O	O	O	O	O
		2	APFC	Aux. Pin Func. Cntl.	-	-	-	O	O
		2	STCKDC	STL TCK Duty Cycle	-	-	-	O	O
Select	Local	1	CGM	Cond. Group Member	M	M	M	M	M
		1	SGC	Scan Group Candidate	--	M	M	M	M
		1	SREDGE	Sampling Rising Edge	--	--	--	M	M
Enumerate	Local	4	CID	Controller ID	--	--	M	M	M
		1	CIDI	Controller ID Invalid	--	--	M	M	M
Read-only	Local	32	RDBACK0	Read-back 0	O	O	O	O	O
		32	RDBACK1	Read-back 1	O	O	O	O	O
		32	CNFG0	Configuration Reg. 0	M	M	M	M	M
		32	CNFG1	Configuration Reg. 1	O	O	O	O	O
		32	CNFG2	Configuration Reg. 2	O	O	O	O	O
		32	CNFG3	Configuration Reg. 3	O	O	O	O	O
NOTE—M is mandatory register and O is optional register.									

### 9.3.1.3 Register reset values

Table 9-2 shows the effects of the six types of TAP.7 Controller resets (see 10.2) on the register values that are managed with commands. A legend describing the symbols used in this table is located below this table. Other effects of TAP.7 Controller resets on the characteristics of TAP.7 Controller commands are described in 10.2.1.5. The effects of these resets on registers managed with Transport Directives are described in Table 27-5.

### 9.3.2 Specifications

#### Rules

The specification rules are as follows:

- a) Each subsequent specification in 9.3.2 shall only apply to T1 and above TAP.7s unless otherwise noted.

- b) Each TAP.7 Class shall implement all registers in Table 9-1 that are listed for the class as either:
  - 1) Labeled (M)—mandatory.
  - 2) Labeled (O)—when the function associated with that register is implemented.
- c) The initialization of the TAP.7 Controller registers by Type-0–Type-4 Resets shall be governed by Table 9-2 and provide asynchronous reset behavior.

**Table 9-2 — Reset values of TAP.7 Controller registers managed with commands**

Register type	Width	Register mnemonic	Name	Values for reset type			
				0	1–3	4	5
Control	1	ECL	Exit Control Level	0	0	0	NC
	1	SUSPEND	Suspend	0	0	0	
	1	ZBSINH	ZBS Detect Inhibit	0	0	0	
	5	SCNFMT	Scan Format	DF	DF	DF	
	1	SSDE	SSD Enable	0	0	0	
	2	DLYC	Delay Control	00	00	00	
	2	RDYC	Ready Control	00	00	00	
	5	TPST	TP state Enables	00000	00000	00000	
	2	TPPREV	Transport Protocol Rev	00	00	00	
Options	2	TOPOL	Topology	DT	DT	NC	NC
	1	FRESET	Functional Reset	0	0	0	
	1	TRESET	Test Reset	0	0	0	
	2	PWRMODE	Power-Control Modes	DM	NC	NC	
	2	APFC	Aux. Pin Func. Cntl.	00	00	00	
	2	STCKDC	STL TCK Duty Cycle	00	00	00	
Select	1	CGM	Cond. Group Member	0	0	0	NC
	1	SGC	Scan Group Candidate	DSGM	DSGM	DSGM	
	1	SREdge	Sampling Rising Edge	0	0	0	
Enumerate	4	CID	Controller ID	0000	0000	0000	NC
	1	CIDI	Controller ID Invalid	0	0	0	
Read-only	32	RDBACK0	Read-back 0	N/A	N/A	N/A	N/A
		RDBACK1	Read-back 1				
		CNFG0	Configuration Reg. 0				
		CNFG1	Configuration Reg. 1				
		CNFG2	Configuration Reg. 2				
		CNFG3	Configuration Reg. 3				

NOTE—NC is No change.

DF is Default Scan Format; see Table 10-7.

DM is Default Power-Control Mode; see Rule 18.10.2 i).

DSGM is Default Scan Group Membership; see Table 10-7.

DT is Default Topology; see Table 20-20.

## 9.4 Command portfolio

### 9.4.1 Description

#### 9.4.1.1 Command types

There are eleven mandatory and five optional commands as shown in Table 9-3. These commands are classified as one of the command types listed as follows:

- Store              The operand is stored in a register or causes an action.
- Select             Manage group membership and conditional command execution.
- Scan               Inputs and outputs with a CR Scan.
- Enumerate        CID allocation and deallocation.
- Private           Commands available for chip-specific definition.

The definition of the commands with their operands is shown in Table 9-3.

**Table 9-3 — TAP.7 Controller command list**

<b>Command type</b>	<b>Op-code count</b>	<b>Mnemonic</b>	<b>Name</b>	<b>TAP.7 Class</b>				
				<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
Store	0x00	STMC	Store Miscellaneous Control	M	M	M	M	M
	0x01	STC1	Store Conditional one-bit	M	M	M	M	M
	0x02	STC2	Store Conditional two-bits	M	M	M	M	M
	0x03	STFMT	Store Format	—	M	M	M	M
	0x04	STTPST	Store Transport state	—	—	—	—	M
	0x05	STTESTM	Store Test Mode	O	O	O	O	O
Select	0x06	MCM	Make Cond. Group Member	—	—	M	M	M
	0x07	MSC	Make Scan Group Candidate	—	—	M	M	M
Scan	0x08	SCNB	Scan Bit	M	M	M	M	M
	0x09	SCNS	Scan String	M	M	M	M	M
Enumerate	0x0A	CIDA	Controller ID Allocate	—	—	M	M	M
	0x0B	CIDD	Controller ID De-allocate	—	—	M	M	M
Private	0x0C–0x0F	EXC3:EXC0	Extended Commands	O	O	O	O	O
Reserved	0x10–0x1F	Rsvd.	Reserved	—	—	—	—	—

NOTE—M is mandatory command and O is optional command.

Commands and subcommands are implemented when a register they manage is implemented. The execution of certain store and select commands is predicated. Command predication is explained below.

#### 9.4.1.2 Store commands

Store Commands are two-part commands available in both the Star and Series Scan Topologies. The Store Miscellaneous Control (STMC) Command performs Control Functions that affect all TAP.7 Controllers sharing a DTS connection. It stores values in one-bit and two-bit registers.

The Store Conditional one-bit (STC1) and Store Conditional two-bits (STC2) Commands store one-bit and two-bit values in registers, respectively. These commands operate in two ways, unconditionally storing registers in all TAP.7 Controllers or storing registers in only the TAP.7 Controllers that have a logic 1 CGM Register value. The Store Format (STFMT) Command unconditionally stores a five-bit Scan Format

value. The Store Data Channel State (STDCST) Command stores a five-bit value that identifies the states supporting transport.

#### 9.4.1.3 Select commands

Select Commands are two-part commands available in both the Star and Series Scan Topologies. The Make Conditional Group Member (MCM) Command manages the CGM Register value, with the value of this register affecting the execution of conditional commands and access to certain EPU Scan Paths.

The Make Scan Group Candidate (MSC) Command manages the SGC Register. The value of this register determines whether the STL becomes an Idle Group Member when Idle Group Membership is updated while using a scan format other than JScan0 and JScan1.

These two commands are called target commands as they use the Controller ID (see 4.2.7.4) to identify the TAP.7 Controller that is the target of the command. These commands affect only a TAP.7 Controller whose Controller ID is specified by Command Part Two. These commands are treated as no-operations when a TAP.7 Controller whose Controller ID is not specified by Command Part Two or the Controller ID Invalid (CIDI) Register indicates the TAP.7 Controller's Controller ID is invalid (with a logic 1). Select commands are no operations for T2 and below TAP.7s, as these TAP.7 Controllers do not have a Controller ID.

#### 9.4.1.4 Scan commands

Scan Commands are three-part commands available in both the Star and Series Scan Topologies. The CR Scan portion of a Scan Bit (SCNB) Command is used to set and test bits while the Scan String (SCNS) Command is used to transfer strings of bits. These commands may be used to access both public and private registers.

#### 9.4.1.5 Enumerate commands

Enumerate Commands manage the allocation and deallocation of a Controller IDs. The Controller ID Allocate (CIDA) Command is a three-part command used to allocate a unique CID to a TAP.7 Controller. The Controller ID Deallocate (CIDD) Command is a two-part command used to deallocate a specific CID or all CIDs. The operation of the CIDA Command is described further in 9.9.3.

#### 9.4.1.6 Private commands

Four two-part commands are available for private use. The execution of these commands is conditional. These commands execute as a no-operation when the CGM Register bit value is a logic 0 and perform their intended function otherwise. This provides a means to use these commands with different TAP.7 Controllers without conflicts.

#### 9.4.1.7 Effects a TAP.7 Controller reset

A TAP.7 Controller reset sets the Conditional Group Membership Register value to a logic 0 and allocates a valid CID of zero (0000b). The logic 0 Conditional Group Membership Register value affects the execution of STC1, STC2, and EXC0–EXC3 commands. These commands execute as no operations until the Conditional Group Membership Register value is set to a logic 1. The SCNB and SCNS Commands cannot access EPU Scan Paths other than a Bypass Bit with a logic 0 Conditional Group Membership Register value.

Selection commands with a Controller ID of zero used following a TAP.7 Controller reset and prior to deallocation of Controller IDs perform their intended function in all Online TAP.7 Controllers sharing a DTS connection. These commands affect the Conditional Group Membership Register and Scan Group Candidate Register values of all TAP.7 Controllers. Selection commands with a CID other than zero create no operation during this period.

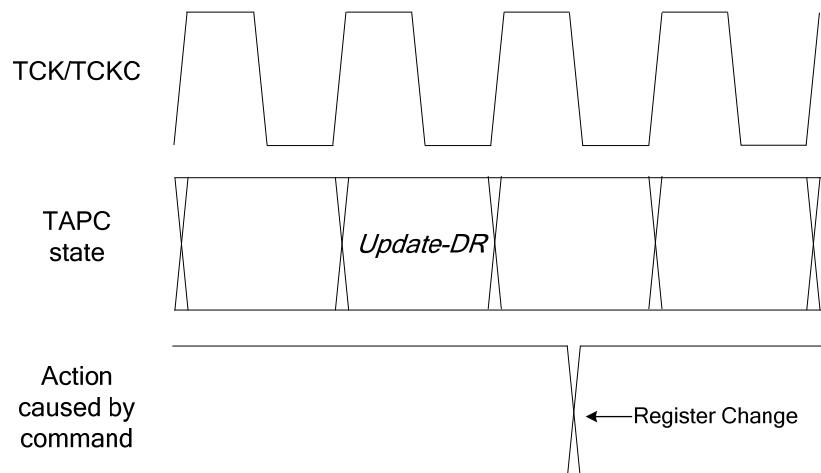
The normal use of selection commands begins after a CIDD Command is used to deallocate all Controller IDs, and a CIDA Command is used to allocate a unique Controller ID is allocated to each TAP.7 Controller that is to be individually accessed.

## 9.4.2 Specifications

### Rules

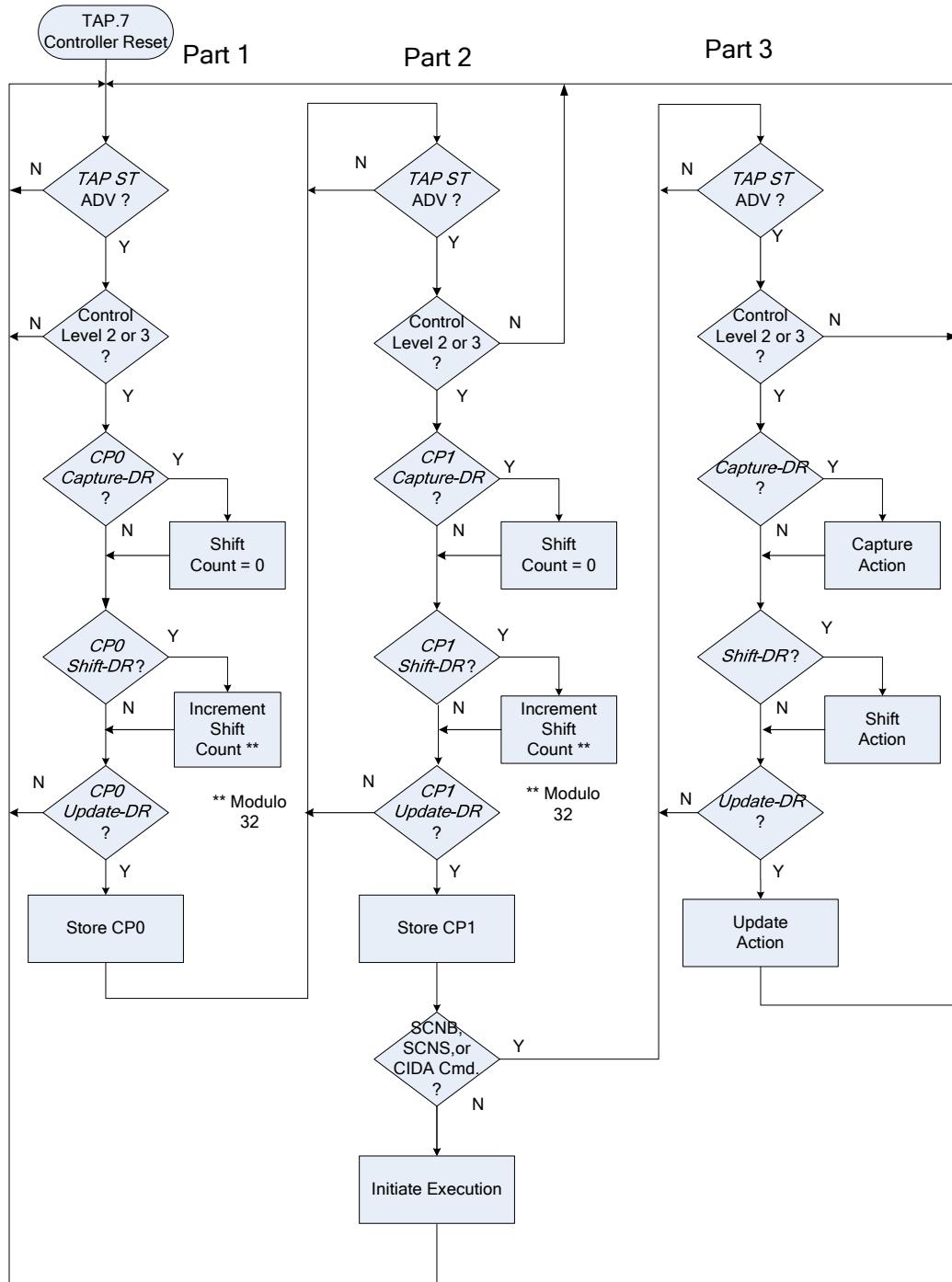
The specification rules are as follows:

- a) Each subsequent specification in 9.4.2 shall only apply to T1 and above TAP.7s unless otherwise noted.
- b) A command or subcommand shall be implemented when a register or a function it manages is implemented.
- c) Decode and execution of a TAP.7 Controller command shall be governed by Figure 9-3.
- d) The Modulo 32 count of the number of *Shift-DR* states between the *Capture-DR* and *Update-DR* states of the CP1 scan shall be used as the five-bit command op-code for the two-part and three-part commands listed in Table 9-4.
- e) The Modulo 32 count of the number of *Shift-DR* states between the *Capture-DR* and *Update-DR* states of the CP2 scan shall be used as the five-bit command operand for the two-part and three-part commands listed in Table 9-4.
- f) A defined or reserved command shall not be used to perform a function other than the following:
  - 1) Defined for the command in this specification.
  - 2) Allowed by the class of the TAP.7 Controller.
- g) Each TAP.7 Class shall implement all commands in Table 9-3 that are as follows:
  - 1) Labeled (M).
  - 2) Labeled (O), when the function associated with that command is implemented.
- h) The action generated by a two-part command shall occur coincidently with the second falling edge of the EPU Test Clock after entry into the *Update-DR* state as shown in Figure 9-2.



**Figure 9-2 — Command completion timing**

- i) The action generated by a three-part command shall occur coincidently with the second falling edge of the EPU Test Clock after entry into the *Update-DR* state of the CR Scan.
- j) The function of commands shall be governed by Table 9-4.



**Figure 9-3 — Command construction and execution**

**Table 9-4 — TAP.7 Controller command definition**

NOTE—The opcode is the left-most five-bit value and the operand is the right-most five-bit value.

<b>Store Commands</b>	
<b>STMC</b>	<b>Store Miscellaneous Control</b>
00000 bbbxy	<pre> bbb==0:StateCtl:xy==0:NOP,           1:ExitCmdLev(ECL),           2:Exit/suspend (SUSPEND = 1),           3:ZBS Inhibit(ZBSINH = 1)  1:ScanCtl:x==0:SGC=y,           1:CGM=y 2:RdyCtl:RDYC=xy, 3:DlyCtl:DLYC=xy, 4:ScnFunc:x==0:SSDE=y,           :x==1:y==0:Star-4 Topology Test,           :x==1:y==1:Rsvd 5:TPPREV:TPPREV =xy, 8-7:Rsvd </pre>
<b>STC1</b>	<b>Store Conditional one-bit</b>
00001 cbbbv	<pre> c==conditional, bbb==bit select, v==value c==0:Assignment made unconditionally,           1:Assignment made provided the CGM Reg. value ==1 bbb==0:SREDGE=v,           1:FRESET=v,           2:TRESET=v, 3-7:Rsvd </pre>
<b>STC2</b>	<b>Store Conditional two-bits</b>
00010 cbbvv	<pre> c==conditional, bb==field select, vv==value c==0:Assignment made unconditionally,           1:Assignment made provided the CGM Reg. value =1 bb==0:PWRMODE=vv,           1:STCKDC=vv,           2:APFC=vv,           3:TOPOL=vv </pre>
<b>STFMT</b>	<b>Store Scan Format</b>
00011 nnnnn	SCNFMT =nnnnn;
<b>STTPST</b>	<b>Store Transport State</b>
00100 yyyy	<p>Defines the states where a TP may follow an SP.  DCST=yyyyy;</p> <p>yyyyy==0xxxx:Run-Test/Idle state - no TP follows an SP</p> <p>1xxxx:Run-Test/Idle state - a TP follows an SP,</p> <p>x0xxx:Pause-IR state - no TP follows an SP,</p> <p>x1xxx:Pause-IR state - a TP follows an SP,</p> <p>xx0xx:Pause-DR state - no TP follows SP,</p> <p>xx1xx:Pause-DR state - TP follows SP,</p> <p>xxx0x:Update-IR state - no TP follows SP,</p> <p>xxx1x:Update-IR state - TP follows SP,</p> <p>xxxx0:Update-DR state - no TP follows SP,</p> <p>xxxx1:Update-DR state - TP follows SP</p>

**Table 9-4 — TAP.7 Controller command definition  
(continued)**

<b>STTESTM</b>	<b>Store Test Mode</b>
00101 yyyyy	CGM==0:Command executes as a no-operation, 1:Command may be used to set test and other modes.
<b>MCM</b>	<b>Select Commands</b>
00110 miiii	<p><b>Make Conditional Group Member</b></p> <p>Clears the CGM Register when the CIDI Register is a logic 1 (the Controller's CID is invalid) and otherwise sets the CGM bit in a TAP.7 Controller that has a valid CID and the CID matches iiii.</p> <p>m==0:CGM Register of a non-targeted controller is cleared.</p> <p>m==1:CGM Register of the targeted controller is set, CGM bit of a non-targeted controller is unaffected.</p> <p>The CGM Register is also stored with the STMC and SCNB Commands.</p>
<b>MSC</b>	<b>Make Scan Group Candidate</b>
00111 miiii	<p>Clears the SGC Register when the CIDI Register is a logic 1 (the Controller's CID is invalid) and otherwise sets the SGC bit in a TAP.7 Controller that has a valid CID and the CID matches iiii.</p> <p>m==0:SGC Register of the targeted controller is set. The SGC Register of a non-targeted controller is cleared.</p> <p>m==1:SGC Register of the targeted controller is set, SGC bit of a non-targeted controller is unaffected.</p> <p>The SGC Register is also stored with the STMC and SCNB Commands.</p>
<b>SCNB</b>	<b>Scan Commands</b>
<b>SCNB</b>	<b>Scan Bit</b>
01000 YYYYY + CR Scan	<p>yyyyy:Bit to be read or written when the CR Scan occurs.</p> <p>Public Registers:</p> <ul style="list-style-type: none"> <li>yyyyy==00:SGC, Scan Group Candidate, write,</li> <li>01:CGM, Conditional Group Member, write,</li> <li>05:02:CNFG0[3:0], TAP.7 Controller class, read,</li> <li>06:FRESETR, Functional Reset requested, read,</li> <li>07:Series topology detect, write,</li> <li>15:08:Rsrd., read as a logic 0</li> </ul> <p>Private Registers:</p> <ul style="list-style-type: none"> <li>31:16:Private - available for customization</li> </ul> <p>See 11.9.1 for command operation</p>

**Table 9-4 — TAP.7 Controller command definition  
(continued)**

SCNS	Scan String
01001 yyyy + CR Scan	YYYYYY:String read or written with the CR Scan. YYYYYY==0:RDBACK0 Reg. [31:00] read only (optional) 01:RDBACK1 Reg. [31:00] read only (optional) 02:CNFG0 Reg. [31:00] read only 03:CNFG1 Reg. [31:00] read only (optional) 04:CNFG2 Reg. [31:00] read only (optional) 05:CNFG3 Reg. [31:00] read only (optional) 07:06:Rsvd, bits read as a logic 0. 31:08:Private - available for customization See 9.9.2 for command operation. See Table 9-13, Table 9-14, Table 9-15, Table 9-16, and Table 9-17 for descriptions.
<b>Enumerate Commands</b>	
CIDA	<b>Controller ID Allocate</b>
01010 siiii + CR Scan	s:Source of aliasing target (AT). iiii==CID. s==0:DTS - Directed CID Allocation, 1:TS - Undirected CID Allocation See 9.9.3 for command operation
CIDD	<b>Controller ID De-allocate</b>
01011 uiiii	u:De-allocate if CID match/unconditionally de-allocate. u==0:CIDI=1, if this controller's CID == iiii 1:CIDI=1
<b>Private</b>	
EXC0-EXC3	<b>Extended Commands</b>
011xy nnnnn	nnnnn:Custom operand. Commands execute as a no-operation unless CGM == 1 xy==0:EXC0, 1:EXC1, 2:EXC2, 3:EXC3
<b>Reserved</b>	
10000 00000 through 11111 11111	Reserved for future use, cannot be customized, two part command with no operation.

- k) A TAP.7 identified as a specific TAP.7 Class (Classes T0–T5) shall implement all of the following:
  - 1) All mandatory capabilities identified mandatory to the TAP.7 Classes below it unless explicitly stated otherwise.
  - 2) The mandatory capabilities identified as supported by this TAP.7 Class.
  - 3) None of the mandatory capabilities identified as available only to TAP.7 Classes above it.
  - 4) None of the optional capabilities identified as available only to TAP.7 Classes above it.

## Permissions

The specification permissions are as follows:

- l) A TAP.7 Class (Classes T0–T5) may implement any optional feature of its class or any TAP.7 Class below its class.

## 9.5 Representation of commands in examples

Commands, TAPC State Machine states, and ZBSs are represented as shown in Table 9-5 in examples in this document.

**Table 9-5 — Command, TAPC State Machine states, and ZBS representations in examples**

Function	Mnemonic	Representation	Description
Store	STMC	CMD(STMC, ACTION) CMD(STMC, REG=value)	Initiate action Store unconditional
	STC1, STC2	CMD(STCx, U, REG=value)	Store unconditional
		CMD(STCx, C, REG=value)	Store conditional
	STFMT	CMD(STFMT, SCNFMT = format)	Store scan format
	STDCST	CMD(STDCST, DCST = states)	Store DC states
	STTESTM	CMD(STTESTM, TESTM = value)	Store Test Mode
Select	MCM, MSC	CMD(OPCODE,M[CID])	Set in Targeted, Cleared in Non-Targeted
Scan	SCNB	CMD(SCNB, BIT, value)	Bit Output
		CMD(SCNB, BIT, rdval)	Bit Input
		CMD(SCNB, BIT, value, rdval)	Bit In/Out
	SCNS	CMD(SCNS, REG, value)	String Output
		CMD(SCNS, REG, rdval)	String Input
		CMD(SCNS, REG, value, rdval)	String In/Out
Enumerate	CIDA	CMD(CIDA, D[CID])	Directed CID Allocation
		CMD(CIDA, U[CID])	Undirected CID Allocation
	CIDD	CMD(CIDD, [CID])	De-allocate CID
		CMD(CIDD, [ALL])	De-allocate all CIDs
Private	EXC3:0	CMD(EXCx, REG=value)	Store
TAPC SM state	State	TCS(State, count)	State and count
ZBS	—	ZBS(count)	Count

## 9.6 Global and Local Register programming with commands

The commands in Table 9-3 are classified as either global or local. A global command performs the same action in all TAP.7 Controllers processing the command. The local commands listed below perform their specified action in a command target based on the following criteria:

- CMD(STMC, Test for Star Scan Topology)—the TOPOL Register is stored with a controller specific value
- CMD(STC1, C, Register = value)—the register specified by the command is stored provided the CGM Register value is a logic 1
- CMD(STC2, C, Register = value)—the register specified by the command is stored provided the CGM Register value is a logic 1
- CMD(SCNB, BIT, value)—the register specified by the command is written provided the CGM Register value is a logic 1 or the specified register is a public register

- CMD(SCNB, BIT, rdval)—the register specified by the command is read provided the CGM Register value is a logic 1 and drive policies permit it being read through the pin used for output. Unique data is read based on the scan chain position in a Series Scan Topology
- CMD(SCNB, BIT, value, rdval)—combination of two previous commands
- CMD(SCNS, REG, value)—the register specified by the command is written provided the CGM Register value is a logic 1
- CMD(SCNS, REG, rdval)—the register specified by the command is read provided the CGM Register value is a logic 1 and drive policies permit it being read through the pin used for output
- CMD(SCNS, REG, value, rdval)—combination of two previous commands
- CMD (MCM, M[*CID*])—the CGM Register is stored or cleared based on the CID and the M bit in the operand
- CMD (MSC, M[*CID*])—the SGC Register is stored or cleared based on the CID and the M bit in the operand
- CMD(CIDD, [*CID*])—the CID determines the CID that is deallocated
- CMD(CIDA, D[*CID*])—the TAP Controller Address defines the command target
- CMD(CIDA, U[*CID*])—the TAP Controller Address generated by TAP.7 Controller arbitration defines the command target

## 9.7 Scan paths

### 9.7.1 Conceptual and physical views

#### 9.7.1.1 Description

##### 9.7.1.1.1 Path characteristics

This standard refers to two views of scan paths, conceptual and physical. There are two conceptual scan paths, the System Path and the Control Path. The System Path is used to access the scan paths associated within the STL. The Control Path is used to access the scan paths with the EPU. These scan paths utilize physical scan paths within the EPU and STL. The EPU Bypass Bit is used at times in both of these conceptual paths as described shortly.

##### 9.7.1.1.2 Conceptual path selection

The System Path is used when one of the following is true:

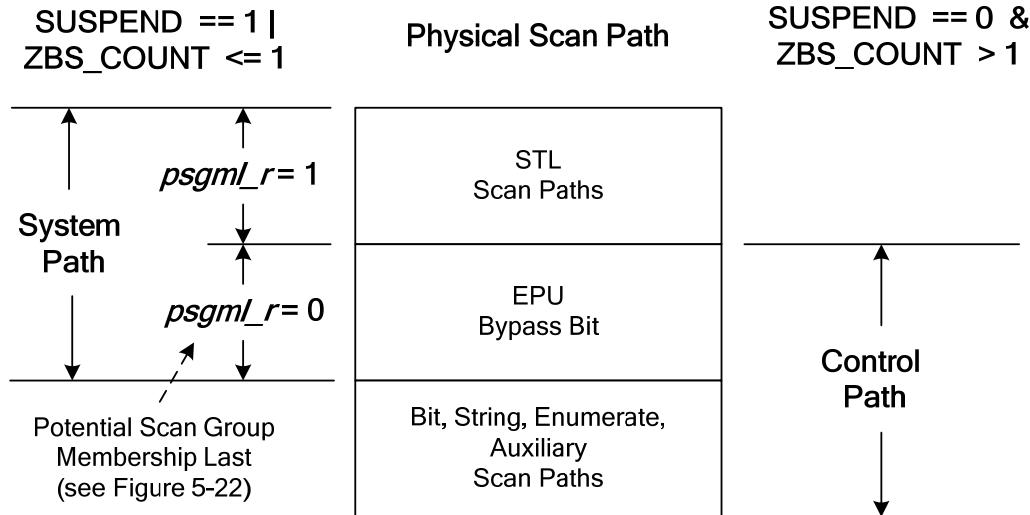
- The EPU Operating State is *SUSPENDED*
- The ZBS count is less than or equal to one

The Control Path is used when neither of the above conditions is true. The specific Control Path is defined by the following:

- Commands
- Control Level
- Conditional Group Membership Register value

### 9.7.1.1.3 Physical path selection

The mapping of conceptual to physical scan paths is shown in Figure 9-4. When the System Path is selected and the STL is not a Potential Scan Group Member, the EPU provides a one-bit bypass for IR and DR Scans as the CLTAPC state is parked.



**Figure 9-4 — Conceptual to physical path mapping paths**

As shown in this figure, when the System Path is as follows:

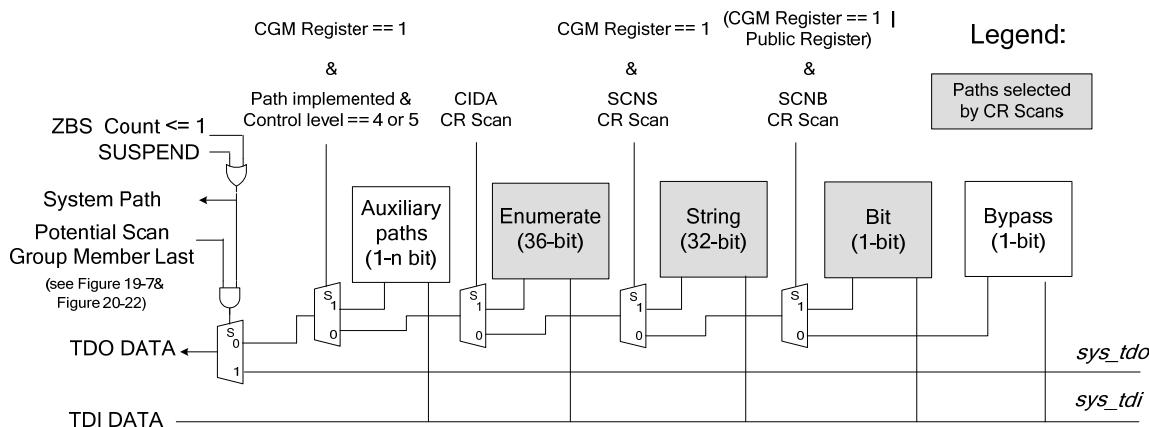
- Selected and the STL is:
  - A Potential Scan Group member, the STL provides the physical scan path
  - Not a Potential Scan Group member, the EPU provides a one-bit physical scan path
- Not selected, the EPU provides the physical scan path

When the STL provides the physical scan path the STL provides the TAP.7 Scan Path and the CLTAPC controls the TDO(C) Timing and Drive Policy. When the EPU provides the TAP.7 Scan Path and controls the TDO(C) Timing and Drive Policy.

A view of a T1 and above TAP.7's physical scan paths is shown in Figure 9-5. The physical scan path is established by a combination of the following:

- ZBS count
- Suspend Register value
- The STL being a Potential Scan Group Member

The left-most data path multiplexer shown in Figure 9-5 merges System and Control Path information. The timing for scan path selection is governed by Figure 9-6. Note that the ZBS count will be zero for IR Scans.



**Figure 9-5 — Conceptual view of the T1 and above TAP.7 Scan Paths**

In this figure, the System Path is formed with an STL Scan Path associated with the CLTAPC when the CLTAPC is selected and the EPU Bypass Bit otherwise. This means the EPU Bypass Bit is utilized at times for both the System and Control Paths. The Control Path is formed with the scan paths within the EPU when the System Path is not selected. These paths provide access to the Bypass Bit, Scan Paths utilized by three-part commands, or Auxiliary Scan Paths accessed with Control Levels Four or Five.

### 9.7.1.2 Specifications

#### Rules

The specification rules are as follows:

- Each subsequent specification in 9.7.1.2 shall only apply to T1 and above TAP.7s.
- The TAP.7 Scan Path for Instruction Register Scans shall be governed by Table 9-6.
- The TAP.7 Scan Path for Data Register Scans shall be governed by Table 9-7.

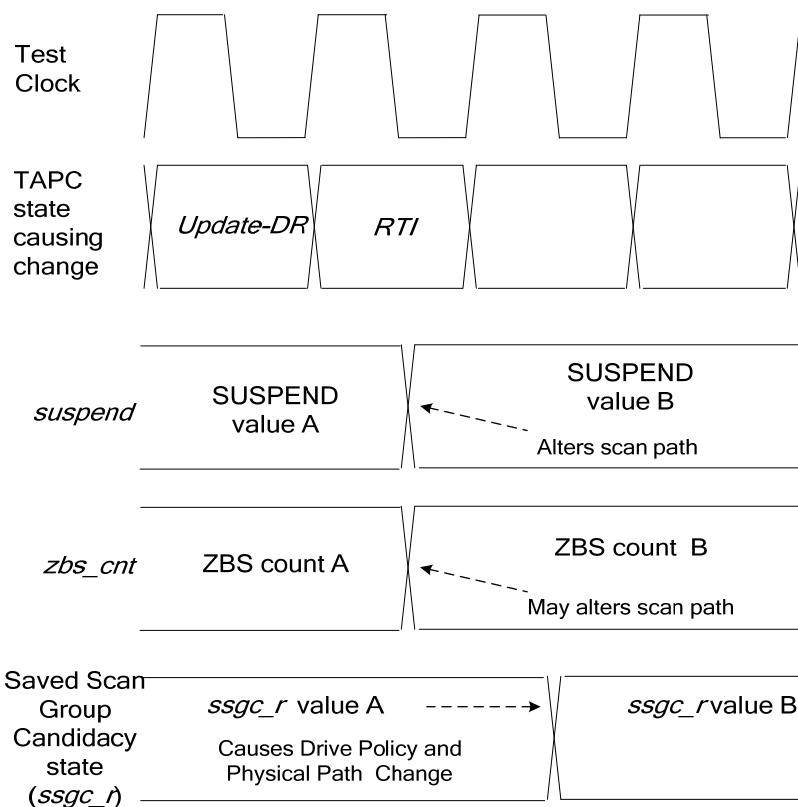
**Table 9-6 — TAP.7 Scan Path for IR Scans**

<b>TAP.7 Class</b>	<b>Potential Scan Group Member ?</b>	<b>TAP.7 Scan Path provided by the:</b>
T0-T1	N/A	STL
T2-T5	Yes	STL
	No	EPU Bypass Bit

**Table 9-7 — TAP.7 Scan Path data for DR Scans**

TAP.7 Class	Potential Scan Group Member ?	$suspend == 0 \& zbs\_cnt \geq 2$ ?	Scan Path Data provided by the:
T0	N/A	N/A	STL
T1	N/A	Yes	EPU
		No	STL
T2-T5	No	x	EPU
	x	Yes	EPU
	Yes	No	STL

- d) A change in the selection of the TAP.7 Controller Scan Path shall be governed by the timing shown in Figure 9-6.



NOTE - The TAPC state changes in this figure are shown for a TAPC with the rising edge of Test Clock causing state changes. When a TAPC with state changes occurring with the falling edge of TCK(C) is implemented, the state changes in this figure would be aligned with the falling edge of TCK. This note is applicable to this and similar figures.

**Figure 9-6 — Saved Scan Group Candidacy/ZBS-count and scan path selection relationships**

## 9.7.2 EPU Scan Paths and their selection

### 9.7.2.1 Description

The five types of EPU Scan Paths are as follows:

- Bypass A One-bit Scan Path using the EPU Bypass Bit.
- Bit A One-bit Scan Path where the register forming this scan path transports a single bit value designated by a TAP.7 Controller command.
- String An n-bit Scan Path transports 1 to n bits designated by a TAP.7 Controller command.
- Enumerate A 36-bit Scan Path transports the TAP.7 Controller Address.
- Auxiliary An n-bit Scan Path transports 1 to n bits using a conventional IEEE 1149.1-style DR Scan Path.

There are both public and private Bit and String paths. A public path is defined by this specification. A private path may be customized.

The CR Scan portion of three-part commands (SCNB, SCNS, and CIDA) access the Bit, String, and Enumerate Paths, respectively. The selection of the path specified by the SCNB and SCNS Commands is further qualified by the CGM Register as shown in Table 9-10. The CIDA Command selects the Enumerate Scan Path.

Auxiliary paths are accessed with Control Levels Four and Five with access to these paths further qualified by the CGM Register value being a logic 1. The EPU Bypass Bit is the default Control Path when no other scan path within the EPU is selected.

The utilization of the physical EPU Scan Paths for all control levels is shown in Table 9-8. The factors controlling the selection of the Bit, String, and Auxiliary Paths are shown in Table 9-10.

### 9.7.2.2 Specifications

#### Rules

The specification rules are as follows:

- a) Each subsequent specification in 9.7.2.2 shall only apply to T1 and above TAP.7s.
- b) The selection of the physical EPU Scan Path shall be governed by Table 9-8 through Table 9-10.

**Table 9-8 — EPU Scan Path selection/control level relationships**

Control level	See Table 9-10 Selected== 1 ?	Scan path
None	x	Bypass
1	x	Bypass
2	Governed by Table 9-9	Governed by Table 9-9
3	x	Bypass
4–5	Yes	Auxiliary
	No	Bypass
6–7	x	Bypass

**Table 9-9 — EPU Scan Path utilization with control level two**

Command Type	See Table 9-10 Selected== 1 ?	Command Part 1	Command Part 2	CR Scan
SCNB	Yes	Bypass	Bypass	Bit
	No			Bypass
SCNS	Yes	Bypass	Bypass	String
	No			Bypass
CIDA	x	Bypass	Bypass	Enumerate
Two-part	x	Bypass	Bypass	N/A

**Table 9-10 — Factors causing selection of the bit, string, and auxiliary paths**

Path	Selected == 1 when:
Auxiliary	Implemented & CGM == 1
Bit	Path is implemented & (CGM == 1   accessing a public register) (see SCNB entry in Table 9-4 for public bit paths)
String	Path is implemented & CGM == 1

- c) The EPU Scan Paths other than those explicitly allowed by this specification shall not be implemented.

### 9.7.3 EPU Scan Path characteristics

#### 9.7.3.1 Description

##### 9.7.3.1.1 Scan-path continuity

The EPU Bypass and EPU Bit Paths provide scan-path continuity with a One-bit Scan Path. The Auxiliary Scan Paths also provide scan-path continuity. The EPU String and EPU Enumerate Scan Paths do not provide this scan-path continuity. Because many features associated with the EPU String Path are used primarily with a Star Scan Topology, this approach provides a smaller implementation size. The same is true for the EPU Enumerate Path. The following subclauses describe the characteristics of each of these scan-path types in more detail.

### 9.7.3.1.2 EPU Bypass-Path characteristics

The EPU Bypass Bit shown in Figure 9-5 provides a one-bit bypass for Data Register Scans when the EPU supplies the EPU Scan Path and another EPU Scan Path is not selected. The capture value for the Bypass Register is a logic 0. Recall the EPU Bypass Bit is also the TAP scan path when the STL is a Potential Scan Group Member and the System Scan Path is selected. The *Update-DR* and *Update-IR* states shall perform no function when the EPU Bypass bit is the TAP scan path.

### 9.7.3.1.3 EPU Bit-Path characteristics

The EPU Bit Paths are conventional one-bit scan paths that are conditionally accessed with the CR Scan of an SCNB Command. They provide scan-path continuity between the EPU's TDI(C) to TDO(C) signals. They may be used simultaneously in one or more TAP.7 Controllers configured in either a Series or Star Scan Topology. Bit-Path Registers may be read only, write only, both readable and writable, or unimplemented.

Public Bit Paths are used to send unique information, such as SGC or CGM Register values or similar information, to all TAP.7 Controllers connected in a Series Scan Topology or broadcast the same value to all TAP.7 Controllers in a Star Scan Topology.

An EPU Bit Path can be used for one of the following operations:

- Write a one-bit register value
- Read a one-bit register value
- Simultaneously store and read a one-bit register value

An EPU Bit Path can be accessed provided any of the following are true:

- The register specified by the operand of the SCNB Command is a Public Register
- The CGM Register value is a logic 1

The EPU Bypass Register is accessed with the SCNB command otherwise.

When an EPU Bit Path Register is as follows:

- Written, then:
  - The *Shift-DR\_f* state of the CR Scan moves the TDI data value into the register forming the Bit Path
  - The *Update-DR\_f* state of the CR Scan stores the value of the register forming the Bit Path in the register specified by the operand of an SCNB Command when this state is exited
- Read, then:
  - The *Capture-DR\_f* state of the CR Scan Stores the value of the register specified by the operand of an SCNB Command in the register forming the Bit Path
  - The *Shift-DR\_f* state exports the value of this register as EPU TDO data

EPU TDO data is output via the TDO(C) or TMS(C) signal provided the Control Path drive policies permit this (see Clause 13 and Clause 14).

### 9.7.3.1.4 EPU String-Path characteristics

The EPU String Paths are unconventional scan paths that are conditionally accessed with the CR Scan of an SCNS Command. These paths may be up to 32 bits in length. They do not provide scan-path continuity

between the EPU's TDI(C) and TDO(C) signals. This means they may be used with only one TAP.7 Controller at a time in either a Series or Star Scan Topology. String-Path Registers may be read only, write only, both readable and writable, or unimplemented.

Public-String Paths are used to read register or configure information in a single TAP.7 Controller connected in a Series or Star Scan Topology. Although the storing of public-string registers is not utilized in this revision of the standard, private-string registers may be written if desired.

A String Path can be used for one of the following operations:

- Import a register value up to 32 bits in length
- Export a register value up to 32 bits in length
- Simultaneously import and export a register value up to 32 bits in length

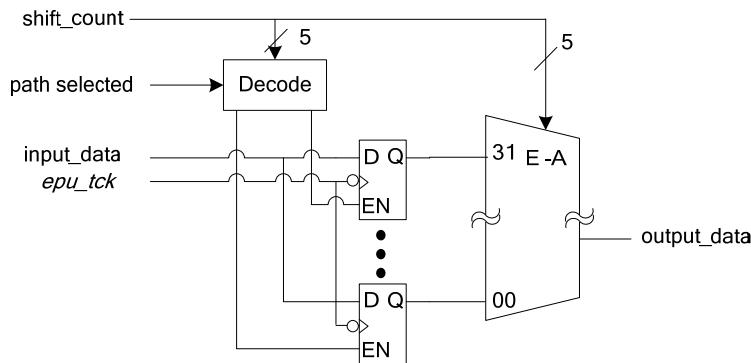
A String Path can be accessed provided the CGM Register value is a logic 1. The EPU Bypass Register is accessed with the SCNS command otherwise.

The primary use of String Paths is the reading of register information from a single TAP.7 Controller in a Star Scan Topology. They are specified in a manner that both minimizes their size and provides adequate capability for both Series and Star Scan Topologies. With this optimization, String Registers are read or written using the shift count and a multiplexer/demultiplexer as shown in Figure 9-7. This provides a very affordable method for reading registers.

An EPU String Path will not:

- Be implemented with a conventional shift register scan path
- Provide scan-path continuity from TDI(C) to TDO(C)

An EPU String Path is read or written as if it is a 32-bit value. Because it does not provide scan-path continuity, string input and output operations repeat on 32-bit intervals ( $\text{bit}[k] == \text{bit}[32 + k]$ ). Data cannot be passed from TDI to TDO because there is no shift register connecting these two terminals. Even with this optimization, a String Path may be written or read in either a Series or Star Scan Topology. Read and write operations are associated with the *Shift-DR* TAPC state. There is no action associated with either the *Capture-DR* or *Update-DR* state.



**Figure 9-7 — Conceptual view of a String Scan Path that supports reads and writes**

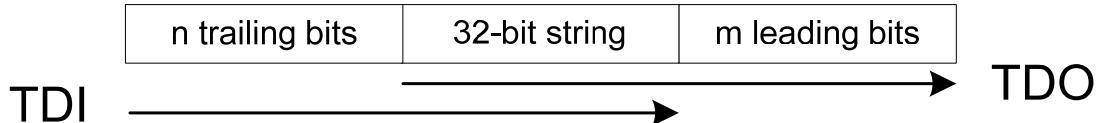
When a Bit Path Register is as follows:

- Written, the *Shift-DR\_f* state of the CR Scan moves the TDI data value into a bit of the register forming the String Path

- Read, the *Shift-DR\_f* state exports the value of this register as EPU TDO data

EPU TDO data is output via the TDO(C) or TMS(C) signal, provided the Control Path drive policies permit this (see Clause 13 and Clause 14).

An Example Series Scan Path with a String Path selected in the chip of interest is shown in Figure 9-8.



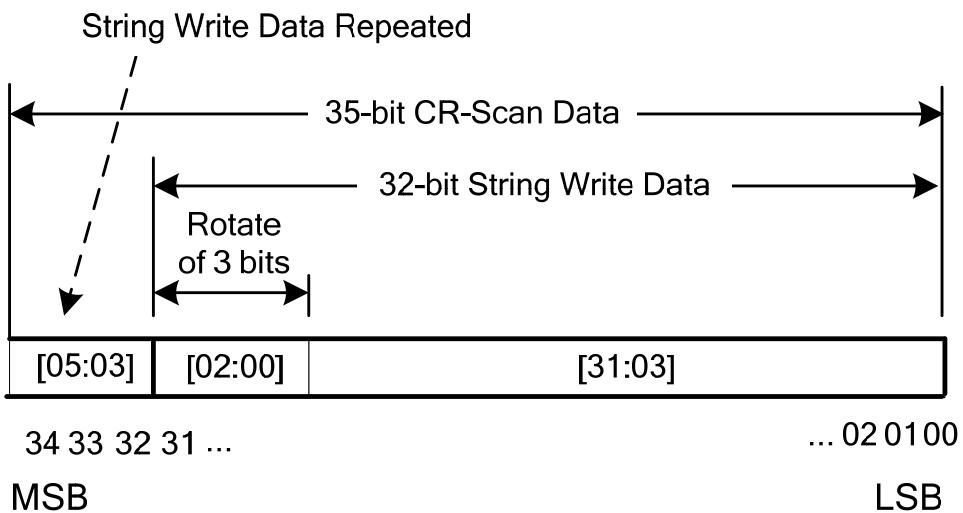
**Figure 9-8 — Example Series Scan Path with a String Path selected in a chip of interest**

The scan counts needed to perform read only, write only, and read and write scan operations is shown in Table 9-11. When m and n are both zero, this becomes the model for the Star Scan Topology.

**Table 9-11 — Scan counts for string operations in a Series Scan Topology**

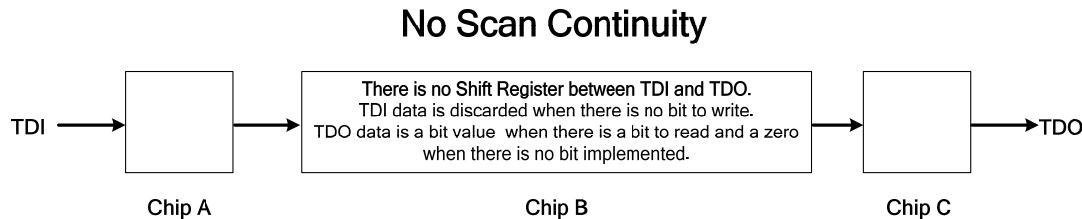
Length comparison	Scan count		
	Read only	Write only	Read and write
$m \geq n$	$32 + m$	$32 + n$	$32 + m$
$n > m$			$32 + n$

The write of a String Path requires special handling of string-write data in a Series Scan Topology. DTS creates string-write data as follows. In the example, scan paths m and n are the number of leading and trailing bits, respectively. For a CR Scan of x bits where  $x \geq 32 + n$ , the DTS rotates 32-bit string-write data right by x modulo 32 bits. For example, if  $n = 3$  and  $x = 35$ , the DTS-generated string-output data is the 32-bit value shown in Figure 9-9. This 32-bit value is repeated after every 32-bit interval. Note that n bits of the String Path are written one or more times, with the final value being the correct value. In the example scan path, the three trailing bits are written to String Registers [02:00] and then written again as string-write data bits [02:00] appear as TDI data on the last three bits of the scan. This should be considered before using string writes in a Series Scan Topology.



**Figure 9-9 — Example of DTS-generated string-write data for three trailing bits**

The system view of a String Scan Path is shown in Figure 9-10.

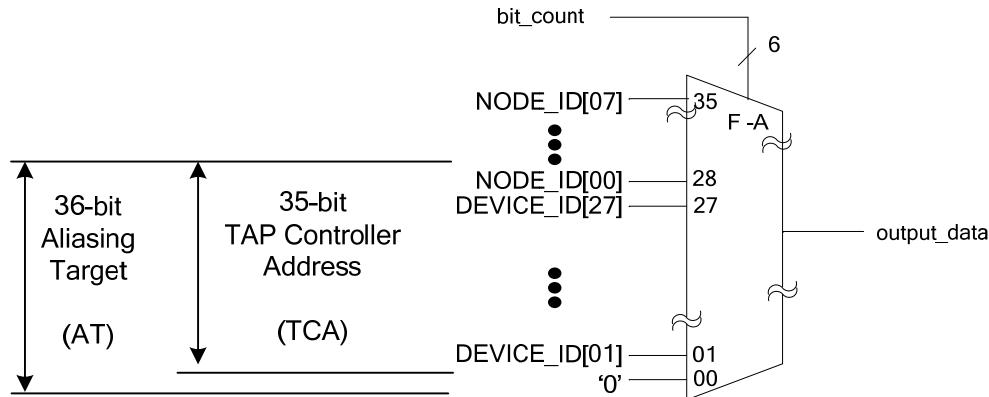


**Figure 9-10 — String Scan Path characteristics in a Series Scan Topology**

#### 9.7.3.1.5 Enumerate-Path characteristics

The Enumerate Path is a read-only path accessed with the CR Scan of a CIDA Command. It is used to allocate a four-bit alias, a CID, to a TAP Controller in a Star Scan Topology. The operation of the CIDA Command and its use of the Enumerate Scan Path are described in detail in 20.10.

A conceptual view of the Enumerate Scan Path is shown in Figure 9-11. It sources a 36-bit aliasing target (AT) during the CR Scan of the CIDA command. The AT is constructed with a logic 0 followed by the 27 LSBs of the DEVICE\_ID (Manufacture ID and Part Number), followed by an eight-bit NODE\_ID. The combination of the 27 bits of DEVICE\_ID and eight-bit NODE\_ID are called the TAP Controller Address (TCA). The NODE\_ID allows the creation of a unique TCA for multiple instantiations of the same chip.



**Figure 9-11 — Conceptual view of the Enumerate Scan Path**

#### 9.7.3.1.6 Auxiliary Path

Auxiliary Paths are optional scan paths that have the characteristics of a conventional IEEE 1149.1 DR Scan Path. They are accessed with DR Scans, provided the CGM Register value is a logic 1 and the Control Level is either four or five.

#### 9.7.3.2 Specifications

##### Rules

- a) Each subsequent specification in 9.7.3.2 shall only apply to T1 and above TAP.7.
- b) The EPU Scan Path characteristics shall be governed by Table 9-12.

**Table 9-12 — EPU Scan Path characteristics**

<b>Control Path</b>	<b>Length in bits</b>	<b>Repeats after 32 bits ?</b>	<b>Capture value</b>	<b>Update action</b>	<b>TDI to TDO data transfer continuity when path becomes the TAP.7 Scan Path?</b>
Bypass	1	No	Logic 0	No operation	Yes
Bit	1	No	Specified Bit	Store Specified Bit	Yes
String	32	Yes	N/A	N/A	No
Enumerate	36	No	N/A	Allocate CID when winning candidate	No
Auxiliary	$\geq 1$	No	Register Value	Store register	Yes

- c) Registers accessed with the Control Paths shall be read or written only, provided all the following are true:
  - 1) The path is selected per Table 9-10.
  - 2) The *Update-DR\_f* state associated with the scan path is exited.
- d) A String Register access shall have all the following characteristics:
  - 1) The *Capture-DR\_f* state of the CR Scan shall load the register providing the String Path output with one of the following:
    - i) Bit [0] of the register specified by the SCNS operand, provided this register is both implemented and readable.
    - ii) Logic 0, provided Bit[0] of this register is neither implemented nor readable.
  - 2) Bit[m] and bit[m+32] are read as the same value, provided the bit being read is read only.
  - 3) The TDI data associated with the first *Shift-DR\_f* state of the CR Scan is stored in the least significant bit (bit[0]).
  - 4) The TDO data associated with the first *Shift-DR\_f* state of the CR Scan is the value of the least significant bit (bit[0]).
  - 5) Register bit[m] shall be written with the TDI input data value associated with the *Shift-DR\_f* state [n] of the CR Scan where m is the value of n truncated to five bits (m is n Modulo 32) when this register is both implemented and writable.
  - 6) Register bit[m] shall be read with the *Shift-DR\_f* state [n] of the CR Scan where m is the value of n truncated to five bits (m is n Modulo 32) when this register is both implemented and readable.

NOTE—The value n begins at 0 following the *Capture-DR* state and increments by one each *Shift-DR* state thereafter.

- e) The EPU String and Enumerate Scan Paths shall be implemented in a manner that does not provide the capture, shift, and update characteristics of an IEEE 1149.1 DR Scan Path (scan-path continuity) when either of these scan paths becomes the TAP.7 Scan Path.
- f) Bypass, Bit, and Auxiliary Paths shall have the capture, shift, and update characteristics of an IEEE 1149.1 DR Scan Path when selected.

- g) The *Capture-IR* state shall load the EPU Bypass Bit with a logic 0 when this bit is the TAP scan path during IR Scans.
- h) The *Update-IR* state shall perform no operation within the EPU when the EPU Bypass bit is the TAP scan path.

## Permissions

The specification permissions are as follows:

- i) Private SCNB and SCNS Commands may be implemented subject to all of the following:
  - 1) Rules in 9.7.2.2.
  - 2) Rules in 9.7.3.2.
- j) Auxiliary Paths may be utilized in any manner consistent with the rules of this specification.

## 9.8 Two-part commands

Two-part commands may be used in any scan topology at any time. These commands utilize only the Bypass Path. The execution of the STC1 and STC2 Commands are used to program Local Registers. Some of these commands are subject to conditional execution, with the result of their execution dependent on the CGM Register value as described in 5.3.3.6. This can be also be considered predicated execution.

The TDO(C) Drive Policy dictates that the TDO(C) signal is not driven when:

- A Control Level is being established (ZBS count = 2 and the Control Level is not locked).
- During Command Part One.
- During Command Part Two.

These are important attributes of creating a Command Control Level and using two-part commands, as they permit the use of commands to establish the scan topology in which the TAPC is deployed.

## 9.9 Three-part commands

The SCNB, SCNS, and CIDA Commands are three-part commands. They utilize the Bit, String, and Enumerate Paths, respectively. In the case of the SCNB Command, the operand field of the command specifies the register affected. With the SCNS Command, the operand specifies the EPU String Path. With the CIDA Command, the operand specifies the CID to be assigned and the method of assignment. The TDO(C) Drive Policy dictates that the TDO(C) signal is only driven when *Shift-DR-f* the state is active in the CR Scan of these commands.

These commands may be used in any scan topology at any time, provided the proper measures have been taken to ensure there will be no drive conflicts (the combination of the scan format being used and the scan topology in which the ADTAPC is deployed is compatible).

### 9.9.1 SCNB Command characteristics

The SCNB Command's CR Scan provides a means to access the register specified by the command operand when either the register is a public register or the CGM Register value is a logic 1. This register is accessed using an EPU Bit Path. The TDO(C) and TMS(C) Drive Policies for the Control Path described in Clause 13 and Clause 14 determine whether the EPU Bit Path may be read when it is accessed. The drive policies affecting the CR Scans of the SCNB and SCNS Commands are created with a combination of the scan format and the command history affecting the CGM Register. These policies prevent the drive of these

pins in a manner that creates drive conflicts when using a scan format supporting the Star-4 and Star-2 Scan Topologies.

The SCNB Command is used to perform the following tasks:

- Store the SGC Register (CLTAPC selection information) in a Series Scan Topology
- Store the CGM Register (TAP.7 Controller selection information) in a Series Scan Topology
- Determine the TAP.7 TAPC Class (reading each bit of the class designation specified by the CNFG field of the Configuration Register with a separate SCNB Command)
- Determine whether a Functional Reset Request has completed via FRESETR
- Establish whether the TAP.7 Controller is deployed in a Series Scan Topology (SERIES\_DET)

The SCNB Command may also be used to access Private Bit Paths as shown in Table 9-4, subject to the rules in 9.7.2.2 and 9.7.3.2.

The CGM Register value determines the characteristics of the SCNB Command's CR Scan by doing the following:

- Allowing access to private Bit Paths. as follows:
  - A Bit Path when the CGM Register value is a logic 1
  - The EPU Bypass Bit when the CGM Register value is a logic 0
- Determining the TDOC and TMSC Drive Policy (determining the number of drive candidates)

A single SCNB Command that is used with a scan format supporting the following:

- Series Scan Topology:
  - May be used to simultaneously read/write a register in one or more TAP.7 Controllers
- Star-4 or Star-2 Scan Topology:
  - May be used to reads a register in a single TAP.7 Controller
  - May be used to writes a register in one or more TAP.7 Controllers

### 9.9.2 SCNS Command characteristics

The SCNS Command's CR Scan provides a means to access the EPU String Path specified by the command operand when the CGM Register value is a logic 1. It is intended to be used to access the String Path of only one of multiple TAP.7 Controllers configured in either a Series or Star Scan Topology. The TDO(C) and TMS(C) Drive Policies for the Control Path described in Clause 13 and Clause 14 determine whether the EPU String Path may be read when it is accessed.

The SCNS Command is used to access the following read-only RDBACKx and CNFGx Registers:

- Scan-related register values via the RDBACK0 Register
- Transport-related register values via the RDBACK1 Register
- Public-configuration information via the CNFG0 and CNFG1 Registers
- Private-configuration information via the CNFG2 and CNFG3 Registers

See 9.10.1 and 9.10.2 for a detailed description of these register accesses.

The CGM Register value determines the characteristics of SCNS Command's CR Scan by doing the following:

- Selecting the EPU Scan Path accessed the following:
  - A String Path when the CGM Register value is a logic 1
  - The EPU Bypass Bit when the CGM Register value is a logic 0
- Determining the TDOC and TMSC Drive Policy

A single SCNB Command that is used with a scan format supporting the following:

- Series Scan Topology:
  - Reads/writes a register in a single TAP.7 Controller
- Star-4 or Star-2 Scan Topology:
  - Reads a register in a single TAP.7 Controller
  - Writes a register in multiple TAP.7 Controllers

The public use of the SCNS Command is limited to reading register values in this specification revision. The SCNS Command may also be used to access Private String Paths as shown in Table 9-4, subject to the rules in 9.7.2.2 and 9.7.3.2. It may be used to read, write, and read and write register bits in a 32-bit Private String Path, provided the characteristics of a String Scan Path are implemented. Care should be exercised in storing string register values in a Series Scan Topology (see 9.7.3.1.4).

### 9.9.3 CIDA Command characteristics

A brief description of the CIDA Command is provided in this subclause. The complete behavior of the CIDA Command is covered in 20.9.

The CIDA Command provides a means to allocate a Controller ID to the TAP.7 Controller of T3 and above TAP.7s, provided it is Online and does not already have a CID allocated to it. Both the Standard and Advanced Protocol support CID allocation with these TAP.7 Controllers. With the Standard Protocol, CID allocation is supported with the JScan3 Scan Format. With the Advanced Protocol, CID allocation is supported with the MScan Scan Format. A Controller ID cannot be allocated with other scan formats as certain attributes provided by the parallel operation of TAP.7 Controllers supported with the JScan3 and MScan Scan Formats are needed for allocation of a CID to occur. The DTS is required to pull up the TDO signal to support CID allocation while using the JScan3 Scan Format.

The CID to be allocated is specified by part two of the three-part command. The CR Scan of the command either provides and or causes the creation of a 36-bit AT reference. The AT reference value is compared to the AT created by the Enumerate Path. When these two 36-bit values are equal, the CID is allocated to the TAP.7 Controller as the *Update-DR* state of the CR Scan is exited, provided exactly 36 bits of the AT Reference are conveyed to the TAP.7 Controller between the *Capture-DR* and *Update-DR* states of the CR Scan.

The AT reference is conveyed to the TAP.7 Controller in one of two ways, either with TDI data provided by the DTS, or by combining the TDO data of TAP.7 Controllers that do not have an allocated CID during the CR Scan of the CIDA Command. The method used to convey the AT Reference is defined by the operand of the CIDA Command as either of the following:

- Directed      The AT reference is conveyed by TDI data
- Undirected    The AT reference is constructed using the Wire-AND of TDO data

With the Directed CID Allocation, the DTS supplies the 36-bit AT reference to the TAP.7 Controller as TDI data during the CR Scan of a CIDA Command. The AT is formed with a logic 0 followed by the 35-bit TCA as described in 9.7.3.1.5. When each bit of the DTS-supplied AT Reference matches each bit of the TAP.7 Controller generated AT, the TAPC is assigned the CID embedded in the CIDA Command when the *Update-DR* state is reached. The TAP.7 Controller is not assigned a CID otherwise. The CR Scan will be precisely 36 bits for the CID assignment to take place.

With Undirected CID Allocation, the AT supplied by the Enumerate Path is output in a manner where it is Wire-ANDed with similar output generated by other TAP.7 Controllers that do not have a CID allocated. This Wire-AND of TDO data creates the AT reference that is compared to the AT supplied by the Enumerate Scan Path. In this case, all TAP.7 Controllers that do not have a valid CID (have not been allocated a CID) arbitrate for the right to be assigned the CID following the *Capture-DR* state of the CR Scan of the CIDA Command. The winner of the arbitration is assigned the CID when the *Update-DR* state is reached. The arbitration process produces a single winner as the AT of each TAP.7 Controller is required to be different.

The arbitration operates as follows. The TAP.7 Controllers vote on the value of an AT reference bit (the values are Wire-ANDed). A TAP controller is disqualified from further voting when it votes for an AT Reference value of a logic 1 and any other voting TAP.7 Controller votes for a logic 0 AT value. Since each TAP controller is required to have a different AT value there is only one TAP controller still voting after the 36-bit AT reference is created. This TAP.7 Controller is allocated the CID when the *Update-DR* state is reached, provided the CR Scan caused the creation of precisely 36 bits of the AT reference.

When Directed CID Allocation is used with the Star-4 Scan Topology, the JScan3 Scan Format uses the TDO value to remain at a high-impedance level during the CR Scan of the CIDA command. In this case, the DTS sees all one's TDO data as the DTS supplies a pull-up behavior where an undriven input assumes a logic 1 signal value as a result of the (PU) bias for the TDO(C) pin (this is required as stated previously).

When Undirected CID Allocation is used with the Star-4 Scan Topology, the JScan3 Scan Format provides for voting on the TDO value during *Shift-DR* states of the CR Scan of the CIDA command. Each bit of the AT Reference is created using a TDO signal precharge/discharge drive scheme that spans four *Shift-DR* states where the TDO pin is as follows:

- *Shift-DR* state [4n] TDO is at a high-impedance level
- *Shift-DR* state[4n+1] TDO is driven as a logic 1
- *Shift-DR* state[4n+2] TDO is a high-impedance level
- *Shift-DR* state[4n+3] TDO is driven with a logic 0 when bit[n] of the AT value supplied by the Enumerate Scan Path is a logic 0 and remains a high-impedance level otherwise

The CR Scan will be precisely 144 bits for the CID assignment to take place. TAP.7 Controllers already assigned CIDs do not drive the TDO pin and appear to vote for a logic 1. This makes them nonparticipants.

When Directed CID Allocation is used with the Star-2 Scan Topology, the TDI bits within MScan SPs convey the AT Reference. The TDO bits within the SPs are a logic 1. This scan format supports the precharge/discharge drive characteristics required for voting on TDO bit values. In this case the DTS sees all-ones TDO data as the DTS supplies a PU bias for the TDO(C) pin (this is required as stated previously). The CR Scan contains precisely 36 *Shift-DR* TAPC states for the CID assignment to take place.

When Undirected CID Allocation is used with the Star-2 Scan Topology, the MScan Scan Format supports the Precharge/Discharge operation of the TMSC pin. This type of operation supports voting on the value of TDO bits within SPs during the CR Scan of a CIDA command. The CR Scan contains precisely 36 *Shift-DR* TAPC states for the CID assignment to take place.

The TDOC and TMSC Pin-Drive Policies permit the simultaneous drive of the following:

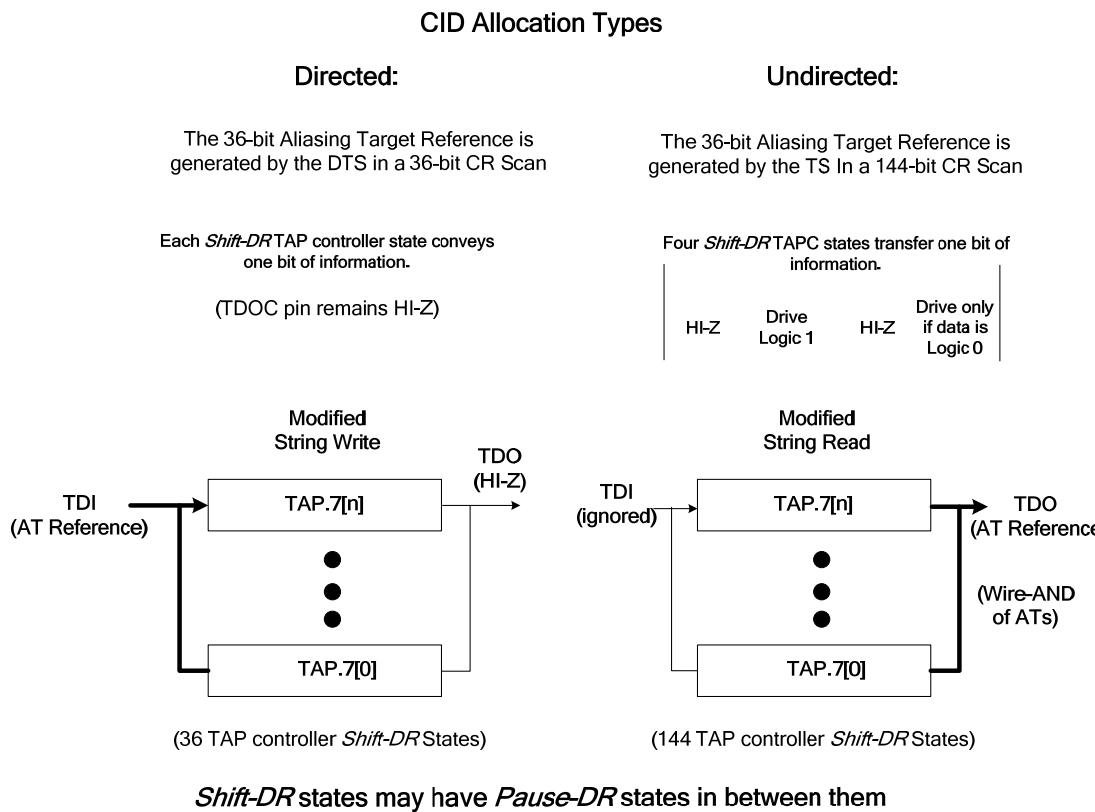
- TDOC with the JScan3 Scan Format
- TMSC with the Advanced Scan Formats

In these cases, the drive policies prevent drive conflicts by imposing precharge/discharge operation of these pins during the CR Scan. With these drive policies, the pin used for output exhibits one of the following pin behaviors:

- Its drive state remains high impedance when other ADTAPCs may drive the pin.
- Its drive state produces the same logic level as other ADTAPC's that may drive the pin at the same time.

The CGM Register value has no effect on the operation of the CIDA Command.

A conceptual view of the use of the Enumerate Scan Path with the Standard Protocol is shown in Figure 9-12.



**Figure 9-12 — Enumerate Scan Path characteristics in a Star-4 Scan Topology**

## 9.10 RDBACK<sub>x</sub> and CNFG<sub>x</sub> Registers

The Read-back and Configuration Registers contain information associated multiple classes and are accessed with the SCNS Command. These registers are described in the following subclauses.

## 9.10.1 RDBACKx Registers

### 9.10.1.1 Description

The read back of register values is supported to assist in developing debug and test applications with the TAP.7 Controller. The optional RDBACK0 and RDBACK1 Registers are used for this purpose. The RDBACK0 Register provides for the read-back of register information related to functions other than transport. The RDBACK1 Register provides for the read-back of register information related to transport.

The ZBSINH, SUSPEND, CGM, CID, and CIDI Registers are not included in the register read-back since these registers are involved in the enabling of register read-back. The RDBACK0 Register format is shown in Table 9-13. The RDBACK1 Register format is shown in Table 9-14.

### 9.10.1.2 Specifications

#### Rules

The specification rules are as follows:

- a) Each subsequent specification in 9.10.1.2 shall only apply to T1 and above TAP.7s.
- b) The RDBACK0 Register format and content shall be governed by Table 9-13.
- c) When the RDBACK0 Register is implemented, each register listed in Table 9-13 implemented for the TAP.7 Class shall be readable using this register.
- d) When the RDBACK0 Register is implemented, each unimplemented register bit listed in Table 9-13 shall be read as a logic 0.
- e) When the RDBACK0 Register is not implemented, it shall be read as an all-zeros value.
- f) The RDBACK1 Register format and content shall be governed by Table 9-14.
- g) When the RDBACK1 Register is implemented, each register listed in Table 9-14 shall be readable with this register.
- h) When the RDBACK1 Register is implemented, each unimplemented register bit that is listed in Table 9-14 shall be read as a logic 0.
- i) When the RDBACK1 Register is not implemented, it shall be read as an all-zeros value.

**Table 9-13 — RDBACK0 Register format**

<b>TAP.7 Class</b>	<b>Bit</b>	<b>Width</b>	<b>Register mnemonic</b>	<b>Name</b>
T1-T4	04:00	5	SCNFMT	Scan Format
	06:05	2	PWRMODE	Power-Control Modes
	07	1	FRESET	Functional Reset
	08	1	TRESET	Test Reset
	09	1	SGC	Scan Group Candidate
	10	1	CGM	Conditional Group Member
	11	1	SSDE	Scan Selection Directive Enable
	13:12	2	TOPOL	Topology
	14	2	SREDGE	Sample Using Rising Edge
	16:15	2	DLYC	Delay Control
	18:17	2	RDYC	Ready Control
	20:19	2	APFC	Aux. Pin Function Control.
	22:21	2	STCKDC	STL TCK Duty Cycle
	31:23	11	Reserved	Read as a logic 0

**Table 9-14 — RDBACK1 Register format**

<b>Class</b>	<b>Bit</b>	<b>Width</b>	<b>Register mnemonic</b>	<b>Name</b>
T5	01:00	2	TPPREV	Transport Protocol Revision
	06:02	5	TPST	Transport state Enables
	09:07	3	TP_DELN	Transport Data Element Length
	14:10	5	PDC0_LCA	PDC0 Logical Channel Address
	16:15	2	PDC0_SEL	PDC0 Selection State
	20:17	4	PDC0_DCC	PDC0 Data Channel Client Selection
	25:21	5	PDC1_LCA	PDC1 Logical Channel Address
	27:26	2	PDC1_SEL	PDC1 Selection State
	31:28	4	PDC1_DCC	PDC1 Data Channel Client Selection

## 9.10.2 CNFGx Registers

### 9.10.2.1 Description

#### 9.10.2.1.1 Overview

The CNFG0 Register provides public-configuration information. The implementation of the following:

- CNFG0 [11:00] is mandatory
- CNFG0 [31:12] is optional

The CNFG1 Register also provides public-configuration information as follows:

- CNFG1 [03:00] is optional
- CNFG1 [31:04] is reserved

All reserved register bits are implemented as a logic 0s.

The optional CNFG2 and CNFG3 Registers provide for private-configuration information.

### 9.10.2.1.2 CNFG0 mandatory configuration information

CNFG0 [11:00] identifies the following:

- The TAPC Class
- The IEEE 1149.7 Specification Revision when it is Class is T1 and above
- The Configuration Registers implemented (CNFG0[31:12], CNFG1, CNFG2, and CNFG3)

It also provides for the identification of TAP.7 Classes in future revisions of the specification.

### 9.10.2.1.3 CNFG0 optional configuration information

CNFG0 [31:12] identifies the following:

- Whether the following options are implemented with a T1 and above TAP.7:
  - Register read-back
  - Reset and power control functions:
    - *TRESET*
    - *FRESET*
    - *PWRMODE*
  - Whether the default Power-Control Mode is sourced by the power manager within the TAP.7 Controller
  - Whether the following options are implemented with a T2 and above TAP.7:
    - The scan format is JScan1 following a TAP.7 Controller reset, deselecting the CLTAPC
  - Whether the following options are implemented with a T4 and above TAP.7:
    - OScan Scan Formats:
      - OScan2 and OScan3
      - OScan4 and OScan5
      - OScan6 and OScan7
    - SScan Scan Formats:
      - SScan0 and SScan1
      - SScan2 and SScan3
    - Auxiliary Pin function control
    - System TCK duty-cycle control
    - TAP width is locked
  - Whether the following options are implemented with a T0 and above TAP.7:
    - An RSU is implemented with a T0–T2 TAP.7
    - Selection Alerts are implemented with an RSU
    - Deselection Alerts are implemented with an RSU

### 9.10.2.1.4 CNFG1 optional configuration information

The implementation of CNFG1 [03:00] is optional. This portion of the CNFG1 Register identifies whether the following options are implemented with a Class T5 and above TAP.7:

- Physical Data Channel 0 is:
  - Not implemented
  - Implemented with one Data Channel Client
  - Implemented with multiple Data Channel Clients
- Physical Data Channel 1 is:
  - Not implemented
  - Implemented with one Data Channel Client
  - Implemented with multiple Data Channel Clients

### 9.10.2.1.5 CNFG2 and CNFG3 Registers

The CNFG2 and CNFG3 Registers may be used for private-configuration information. Any portion of these registers that is not implemented is read as all zeros.

### 9.10.2.1.6 Determining the TAP type and class using configuration information

The CNFG0 [03:00] Register field can be used to identify the type (TAP.1/T0 TAP.7 or T1–T5 TAP.7) of each TAPC within the system. This register field can be read with the SCNB Command as well as with an SCNS Command.

If the DTS presumes all TAPCs have a CNFG0 Register (independently of whether they do or not), it can determine the TAPC type of all TAPCs in either a Series or Star Scan Topology.

T1 and above TAP.7s are distinguishable from TAP.1 and T0 TAP.7s because they contain an EPU. The Bypass Bit within the EPU makes the scan path characteristics of these TAPs different with this difference easily detectable. With a TAP.1 and a *BYPASS* instruction in the Instruction Register, the Bypass-Bit value exported during the *Shift-DR* state following a *Capture-DR* state is a logic 0 each time a read is performed with an SCNB Command. The methods used to read the CNFG0 Register with the SCNB Command produces a 0000b CNFG0 Register value for these TAP types.

With a Series Scan Topology, the TAPC Class is determined with the SCNB Command (four reads of one bit). Once the class is determined, the entire CNFG0 Register for those TAPs determined to be T1 and above TAP.7s may be read using the SCNS Command. With a Star Scan Topology, the entire CNFG0 Register for those TAPs determined to be T1 and above TAP.7s may be read using the SCNS Command without using the SCNB Command.

This is part of a larger TAP.7 capability that also provides for the determination of the scan topology and the TAP types used with the topology.

## 9.10.2.2 Specifications

### Rules

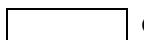
The specification rules are as follows:

- a) Each subsequent specification in 9.10.2.2 shall only apply to T1 and above TAP.7s.
- b) The CNFG0, CNFG1, CNFG2, and CNFG3 Register formats and content shall be governed by Table 9-15, Table 9-16, and Table 9-17.

**Table 9-15 — Configuration Register zero format**

<b>Field</b>	<b>Bit #</b>	<b>Width</b>	<b>Name</b>	<b>Description</b>
CNFG (Class)	03:00	4	CLASS[3:0]	TAP.7 Class. 0000 – TAP.1 or TAP.7 Class T0 0001 – TAP.7 Class T1 0010 – TAP.7 Class T2 0011 – TAP.7 Class T3 0100 – TAP.7 Class T4 0101 – TAP.7 Class T5 0110 – 1110 – Reserved 1111 – TAP.1 or TAP.7 Class T0
CNFG (Revision)	07:04	4	REV[3:0]	TAP.7 specification revision 0000b – Reserved 0001b – this document 0010b – 1111b Reserved
CNFG (Format)	11:08	4	CNFGFMT[3:0]	Configuration Register Format xxx1 – CNFG0[31:12] are implemented xx1x – CNFG1 implemented x1xx – CNFG2 implemented 1xxx – CNFG3 implemented
T1 TAP.7 Options	12	1	RDBKS	RDBACK supported 0 – Unsupported, 1 – Supported
	13	1	TRESETS	TRESET supported 0 – Unsupported, 1 – Supported
	14	1	FRESETS	FRESET supported 0 – Unsupported, 1 – Supported
	17:15	3	PWRMODES[2:0]	PWRMODES[x] – Power-Control Mode x supported where x = 0, 1, 2 (one hot) 0 – Unsupported, 1 – Supported
	18	1	PMDSRC	PMDSRC – Power manager sources default PWRMODE value 0 – No, 1 – Yes
T2 TAP.7 Options	19	1	DAS	DAS – deselected at start-up 0 – No, 1 – Yes
T4 TAP.7 Options	21:20	2	SSCANS[1:0]	0 – SScan1:0, 1 – SScan3:2 supported 0 – Unsupported, 1 – Supported
	24:22	3	OSCANS[2:0]	0 – OScan3:2, 1 – OScan5:4, 2 – OScan7:6 supported 0 – Unsupported, 1 – Supported
	25	1	APFCS	Auxiliary pin function control supported 0 – Unsupported, 1 – Supported
	26	1	TAPWIDS	TAP Width Default 0 – Two-pin supported, 1 – Four-pin supported
	27	1	TAPWLCK	TAP Width Locked 0 – Not locked, 1 – Locked
	28	1	STCKDCS	sys_tck Duty Cycle supported 0 – Unsupported, 1 – Supported
RSU Options	29	1	RSUS	RSU supported with a T0-T2 TAP.7: 0 – Unsupported, 1 – Supported
	30	1	SEL_ALERTS	Selection Alerts: 0 – Unsupported, 1 – Supported
	31	1	DSEL_ALERTS	Deselection Alerts: 0 – Unsupported, 1 – Supported

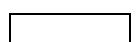
 Mandatory

 Optional

**Table 9-16 — Configuration Register one format**

<b>Field</b>	<b>Bit #</b>	<b>Width</b>	<b>Name</b>	<b>Description</b>
T5 Options	01:00	2	PDC0SUPS	Physical Data Channel 0 implemented: 00 – No 01 – With one Data Channel Client 10 – Multiple Data Channel Clients 11 – Reserved
	03:02	2	PDC1SUPS	Physical Data Channel 1: 00 – No 01 – With one Data Channel Client 10 – Multiple Data Channel Clients 11 – Reserved
Rsvd	31:04	28	Rsvd	Reserved, read as zero

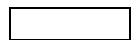
 Mandatory

 Optional

**Table 9-17 — Configuration Register two and three format**

<b>Field</b>	<b>Bit #</b>	<b>Width</b>	<b>Name</b>	<b>Description</b>
CNFG2	31:00	32	User defined	Custom definition
CNFG3	31:00	32	User defined	Custom definition

 Mandatory

 Optional

## 9.11 An approach to implementing command processing and scan paths

A conceptual view of command processing is shown in Figure 9-13. The processing of both two-part and three-part commands is handled by the state machine shown in this figure. It is supported by a five-bit counter and a Temporary Register (TEMP) as shown in Figure 9-14. The function shown in this figure assumes certain partitioning of the TAP.7 Controller. Other approaches are also viable and should be explored.

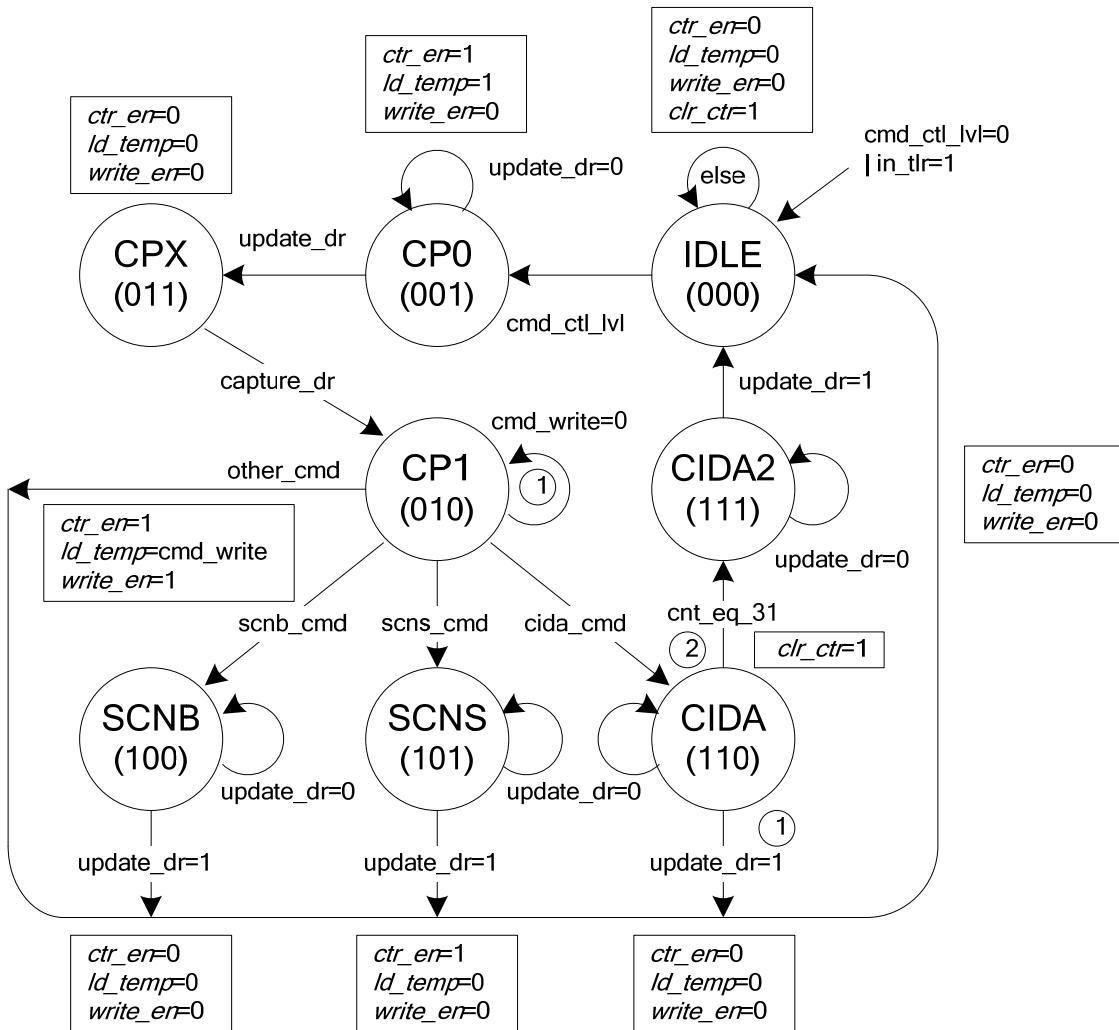
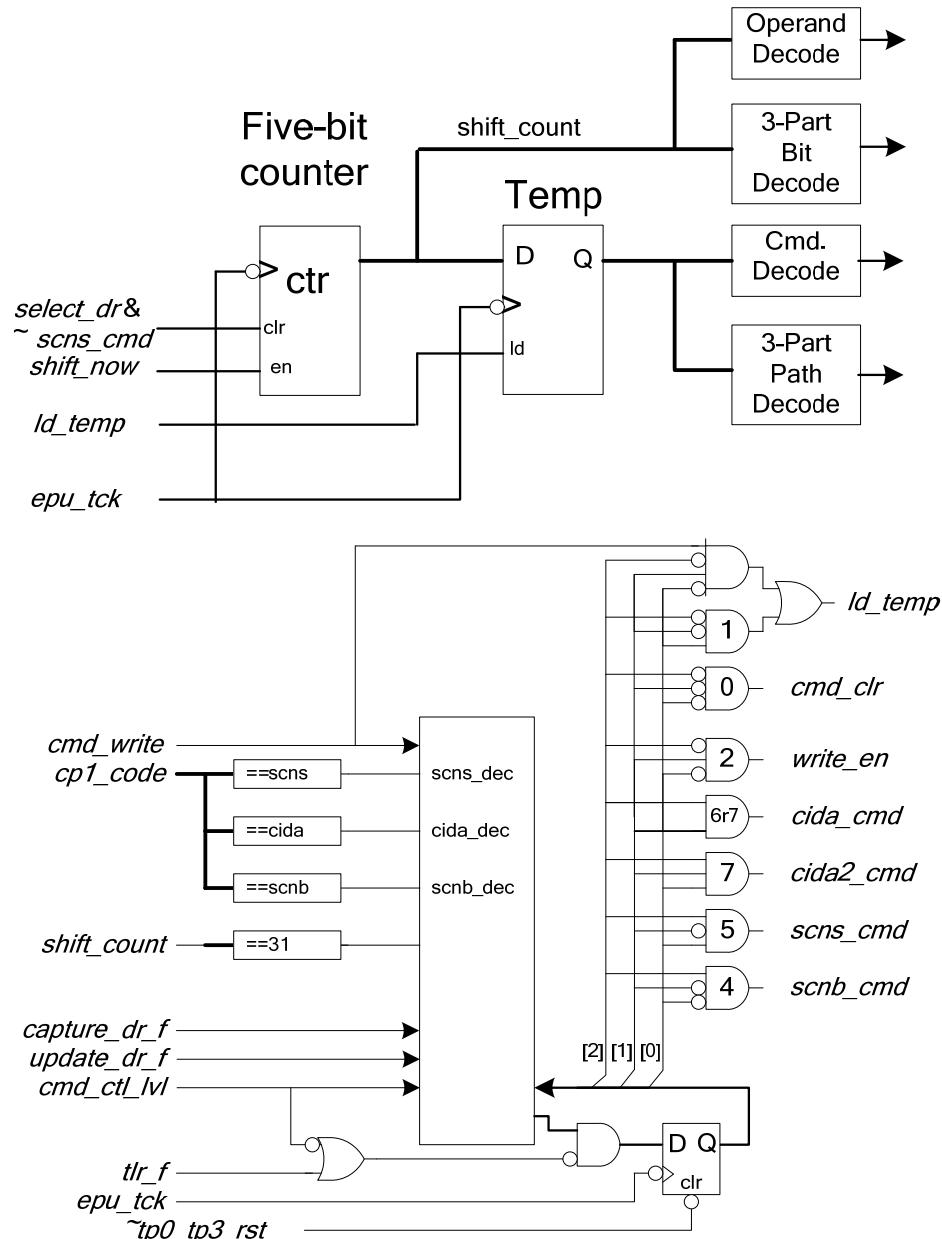


Figure 9-13 — Conceptual view of command processing



**Figure 9-14 — Conceptual view of command processor**

A conceptual view of the EPU Scan Paths is shown in Figure 9-15.

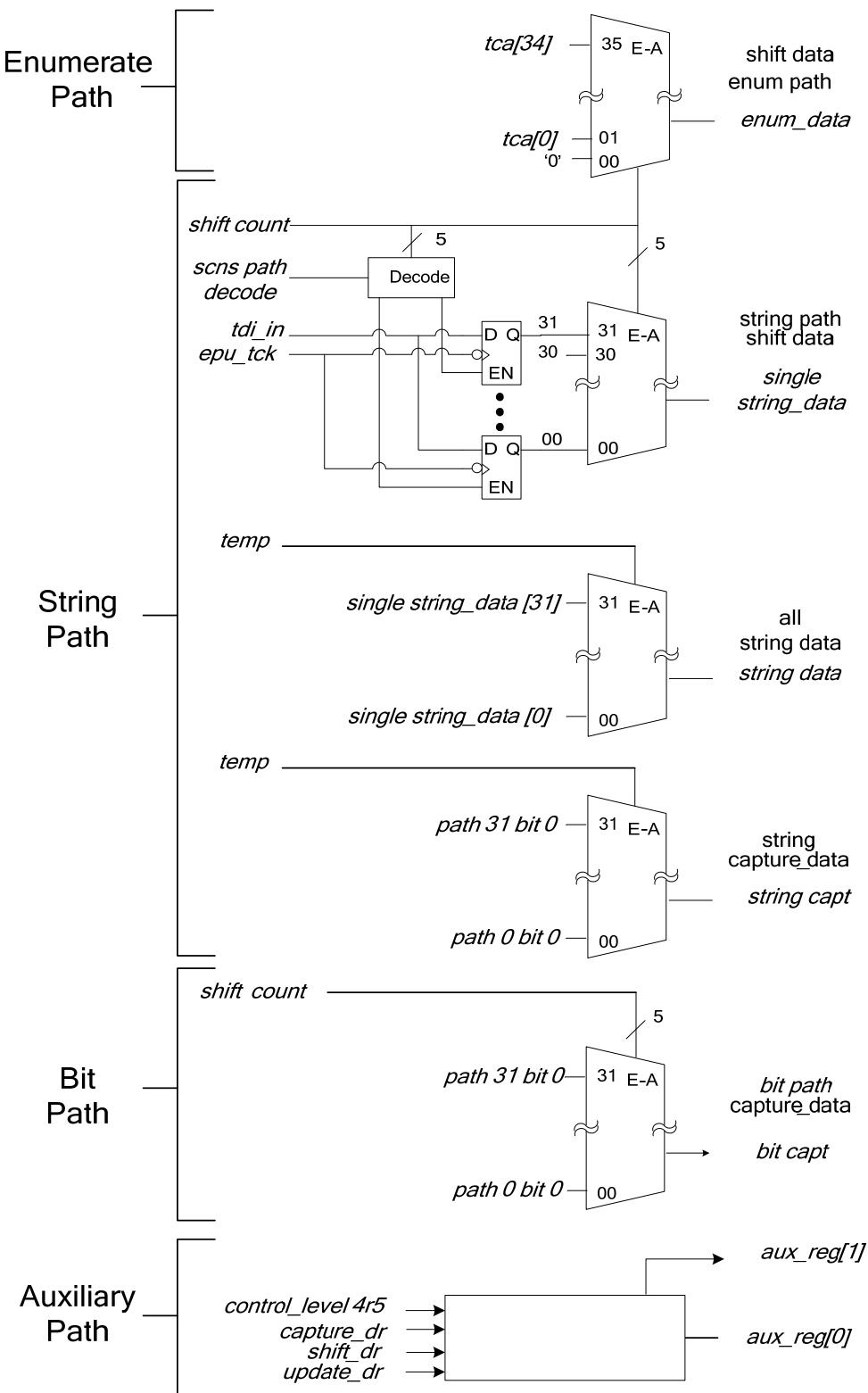


Figure 9-15 — Conceptual view of EPU Scan Paths

## 10. RSU ancillary services

### 10.1 Introduction

This clause is applicable to T0–T2 TAP.7s implementing the RSU and T3 and above TAP.7s where the implementation of the RSU is mandatory. The TAP.7 Controller's ancillary services are described in this clause. It also provides programming considerations.

The subject matter within this clause is organized in the following manner:

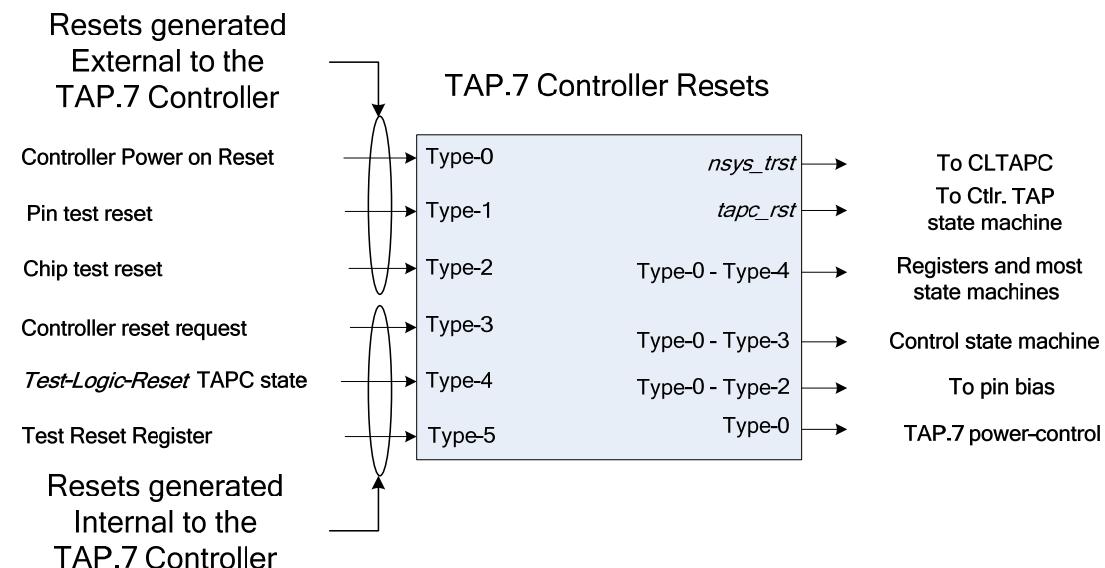
- 10.2 Resets
- 10.3 Start-up options
- 10.4 Escape detection
- 10.5 Selection alert
- 10.6 Deselection Alert
- 10.7 Programming considerations
- 10.8 ADTAPC State Machine

### 10.2 Resets

#### 10.2.1 Description

##### 10.2.1.1 Overview

The TAP.7 Controller reset function expands the reset function provided by IEEE Std 1149.1 to provide six reset types, Type-0–Type-5. It can be implemented with as few as two reset types (Type-2 and Type-4) or with as many as the six reset types described above. This is determined by the TAP.7 Class and options implemented. The TAP.7 Controller reset types are shown in Figure 10-1.



**Figure 10-1 — Conceptual view of the EPU reset logic and state machines**

These resets are described briefly as follows:

- Type-5 A reset of the CLTAPC managed with the TRESET Register
- Type-4 The *Test-Logic-Reset* state
- Type-3 Generated via logic internal to the TAP.7 Controller
- Type-2 Generated by Chip-Level Logic
- Type-1 Generated via the Test Reset pin
- Type-0 Generated by Chip-Level TAP.7 Controller power-management logic

The Type-0, Type-1, and Type-2 Resets are fully asynchronous. These resets are passed directly to the STL without modification. The Type-3, Type-4, and Type-5 Resets are fully synchronous and are initiated by a falling edge of the TCK(C).

Many IEEE 1149.1 implementations have Type-1 and Type-4 Resets, with some also having a Type-2 Reset. The Type-0 Reset is added only when the TAP.7 Controller power management is implemented and generally replaces the Type-2 Reset. The Type-3 Reset is initiated by multiple TAP.7 Controller sources related to power management and the operation of the Advanced and Control Protocols. A Type-5 Reset affects the CLTAPC but not the ADTAPC and the remainder of the TAP.7 Controller.

### 10.2.1.2 Reset considerations

**It is important the Type-0 and Type-1 Resets generated by Chip-Level Logic produce behavior that maximizes the usability of the TAP.7 Controller.** For example, a functional reset of an application should not cause a reset of the test logic. On the opposite end of the spectrum, a reset generated by chip power-up should cause a reset of the test logic.

Applications debug and test tools connected to the TAP.7 Controller suffer loss of synchronization with the TAP.7 Controller when the TAP.7 Controller is reset by a source other than these tools. The applications developer often experiences tool instability and system/tool crashes in these cases. In many instances, the cause of the failure is not obvious. This can diminish the value of a well-designed applications development toolset and make problems associated with system initialization and power management difficult to find.

If the asynchronous TAP.7 Controller reset is generated in an overly aggressive manner (with many Chip-Level Events causing this reset), the ability to debug a system with certain types of problems is severely compromised. It is important that the number of events that can initiate a TAP.7 Controller reset (and a reset of any aspect of the test logic) be minimized.

### 10.2.1.3 Reset effects

The effects of the TAP.7 Controller reset types are shown in Table 10-1.

**Table 10-1 — Reset effects**

Reset	CLTAPC And <i>nsys_trst</i>	TMSC Pin-Hi-Z	State machines other than CSM	Control State Machine (CSM)	TAPC SM state	Escape sequence detection	Pwr.-Control state
Type-5	Yes	—	—	—	—	—	—
Type-4	—	Yes	Yes	—	—	—	—
Type-3	Yes	Yes	Yes	Yes	Yes	Yes	—
Type-2	Yes	Yes	Yes	Yes	Yes	Yes	—
Type-1	Yes	Yes	Yes	Yes	Yes	Yes	—
Type-0	Yes	Yes	Yes	Yes	Yes	Yes	Yes

NOTE—SM is State Machine.

The effect of these resets on TAP.7 Controller state machines is included in the description of these state machines. The effect of resets on registers managed with commands is shown in Table 9-2. The effects of resets on Transport Registers managed with directives are shown in Table 27-5. Type-0–Type-4 Resets also establish the start-up behavior described in 10.3, creating the conditions identified in Table 10-5 through Table 10-7.

#### 10.2.1.3.1 Type-5 Reset

A Type-5 Reset is created by a logic 1 TRESET Register value. It is created only as directed by the DTS as the TRESET Register is set to its inactive state by Type-0 through Type-4 Resets. A Type-5 Reset affects only the STL. It causes the assertion of the *nsys\_trst* signal. This reset provides for establishing the CLTAPC reset at start-up when the CLTAPC may be deselected with a TAPC state other than *Test-Logic-Reset*. It may also be used to reset the STL at any time without the reset of the TAP.7 Controller and may be combined with a private register value to create more than one STL reset.

#### 10.2.1.3.2 Type-4 Reset

A Type-4 Reset is created by the ADTAPC *Test-Logic-Reset* state. It does not cause the assertion of the *nsys\_trst* signal. However, it does affect the selection and deselection of the STL based on the start-up option chosen. The Type-4 Reset initializes the TAP.7 Controller registers and most controller state machines. It establishes the default CLTAPC selection state. It affects but does not initialize the state machine controlling ADTAPC selection.

#### 10.2.1.3.3 Type-3 Reset

A Type-3 Reset is created by Reset Requests generated by the TAP.7 Controller. A Type-3 Reset performs the functions associated with a Type-4 Reset and additionally creates the *Test-Logic-Reset* state.

A Type-3 Reset is generated in response to Reset Requests generated by RSU, EPU, and APU logic, as follows:

- A power-down request (see Figure 18-20)
- A Reset Escape (see 10.4)
- A Check Packet—CP\_RSO Directive (see 11.7.9.1.3)
- A Delay Element—DLY\_RSO Directive (see 22.2.3.3)
- Transport Packet—TP\_RSO, TP\_RSZ Directives (see 27.10.1.6.2)

The requests for a Type-3 Reset are described briefly as follows and in detail with the functions that generate them.

The TAP.7 Controller power management generates a Type-3 Reset Request when it requests that the Chip-Level Power-Management Logic powers the TAP.7 Controller down. The Type-3 Reset Request is continuously asserted until the Chip-Level Logic responds with a Type-0 Reset. This provides a very orderly power-down sequence.

A Reset Escape causes a Type-3 Reset that is one TCK(C) clock period in width. The function provided by a Reset Escape is similar to but not the same as the reset function generated by a test reset pin. With this being the case, consideration may be given to implementing the TAP.7 Controller without the Test Reset signal when Escapes are implemented.

The Advanced Protocol and Control Protocol include certain bit patterns (directives) that cause a Type-3 Reset that is one TCK(C) clock period in width. These directives assist in the initialization of the TAP.7 Controller when the TCKC signal is sourced by the TS and the DTS/TS connection is broken. The DTS may also generate these bit patterns while the DTS/TS connection remains intact.

#### **10.2.1.3.4 Type-2 Reset**

A Type-2 Reset is created by Chip-Level Logic to initialize the TAP.7 Controller when TAP.7 Controller power control is not implemented. It is not implemented when a Type-0 reset is implemented. It performs all of the functions of the Type-3 Reset, in addition to initializing the Escape Detection Logic.

#### **10.2.1.3.5 Type-1 Reset**

A Type-1 Reset is created with either the nTRST or nTRST\_PD signals. It is not created when neither of these pins is implemented. It performs all the functions of the Type-2 Reset. It may initiate TAP.7 Controller power-down, provided the TAP.7 Controller Power-Control Mode enables this function.

#### **10.2.1.3.6 Type-0 Reset**

A Type-0 Reset is implemented when TAP.7 Controller power-down is implemented. It performs all the functions of a Type-1 Reset and also initializes the TAP.7 Controller logic.

#### **10.2.1.3.7 Type-0 versus a Type-2 Reset**

It is expected that either a Type-2 Reset or a Type-0 Reset is implemented but not both.

#### **10.2.1.4 TAP.7 Controller operation is ensured after power-up**

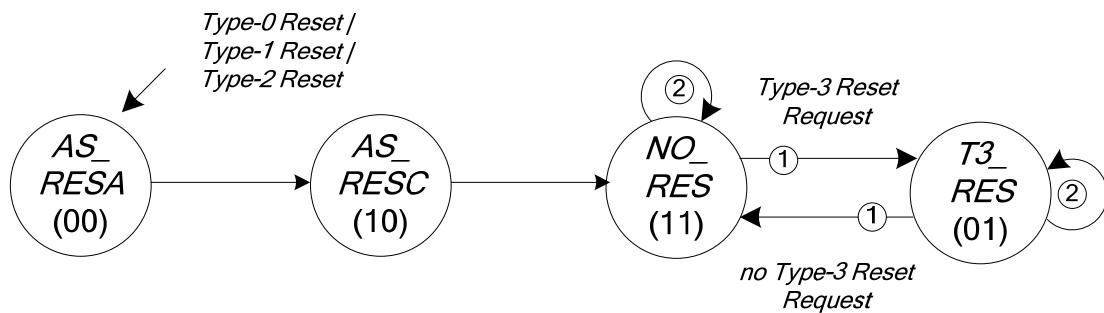
The initialization of the TAP.7 Controller creates a state supporting operation with a single TAP.7 Controller deployed in either a Series or Star Scan Topology. This ensures that the debug of a chip is available after power-up, provided the appropriate start-up protocol sequences are used.

#### **10.2.1.5 Other effects of a TAP.7 Controller reset**

A TAP.7 Controller reset allocates a valid CID of zero (0000b). The CID remains both valid and zero until it is deallocated. The targeted commands with a CID of zero (0000b) used after initialization and prior to deallocation of the CID perform their intended function in all Online TAP.7 Controllers sharing a DTS connection, as they all have a CID value of zero. Selection commands with a CID other than zero create no operation. A CIDD Command with either a CID of zero or the “deallocate all” option deallocates all CIDs (marks them as invalid). The normal use of CIDs begins when there is no more than one TAP.7 Controller with a CID (zero or otherwise) that the DTS intends to use specifying the target of TAP.7 Controller commands.

### 10.2.1.6 An approach to implementing TAP.7 Controller resets

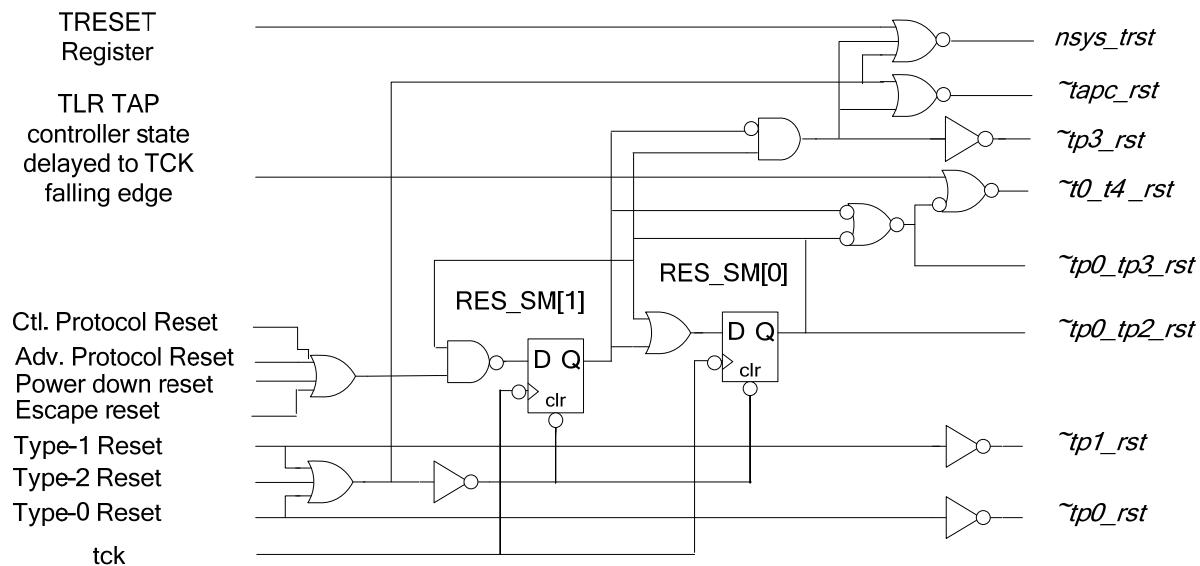
One approach to implementing the reset function views the reset function as the state machine shown in Figure 10-2. This state machine is embedded in the conceptual view of the reset function shown in Figure 10-3. The state definitions for this machine are shown in Table 10-2. This state machine assists in the translation of the reset inputs to the reset outputs used within the TAP.7 Controller and presented to the STL. It handles Type-3 Reset controller Reset Requests. It also ensures Type-0, Type-1, and Type-2 Resets are forwarded to the STL without modifying their duration. Note that this state machine allows a Type-3 Reset only when its state is T3\_RES [i.e., two TCK(C) falling edges have occurred in the absence of a Type-0, Type-1, and Type-2 Reset].



**Figure 10-2 — Reset State Machine**

**Table 10-2 — Reset state descriptions**

Reset state		Description
AS_RESA	Async_Reset Asserted	An async. reset has occurred or is occurring
AS_RESC	Async Reset Completed	An async. reset has occurred and has been released
NO_RES	No Reset	No reset
T3_RES	Type-3 Reset	A Type-3 Reset Request has been processed



**Figure 10-3 — Conceptual view of reset function**

## 10.2.2 Specifications

### Rules

The specification rules are as follows:

- Each subsequent specification in 10.2.2 shall only apply to T1 and above TAP.7s.
- The TAP.7 Controller Type-0-Type-5 Reset types shall have the characteristics shown in Table 10-3 in addition to initializing any other logic necessary for operation consistent with this specification.

**Table 10-3 — Conditions causing assertion of reset types**

Reset	Source	Active when
Type-0	Chip-Level Logic managing TAP.7 Controller power	The TAP.7 Controller power-up or power-down occurs.
Type-1	Pin test reset	nTRST or nTRST_PD signal == logic 0 when either of these pins is implemented.
Type-2	Chip test reset	Chip-Level Logic asserts test reset at chip start-up.
Type-3	TAP.7 Controller Reset Requests	Initiated by any following TAP.7 Controller events: – A power-down request – A Reset Escape detection – The Control Protocol Directive generating a Reset Request – An Advanced Protocol Directive generating a Reset Request.
Type-4	<i>Test-Logic-Reset</i>	The TAPC State Machine state is <i>Test-Logic-Reset</i> .
Type-5	TRESET Register	The DTS stores a logic 1 in this register.

- A Type-3 Reset shall be generated by any of the following:
  - A CP\_RSO Directive for either a T0 – T2 TAP.7 implemented with an RSU or a T3 and above TAP.7 (see Table 11-13).

- 2) A DLY\_RSO Directive for a T4 and above TAP.7 (see Table 24-2).
- 3) A TP\_RSO Directive for a T5 TAP.7 (see Table 27-10).
- 4) A TP\_RSZ Directive for a T5 TAP.7 (see Table 27-10).
- 5) A power-down request for a T1 and above TAP.7 supporting power-down (see Rule 18.10.2 n).
- d) The TAP.7 Controller Type-0–Type-5 Resets shall have the effects shown in Table 10-4.

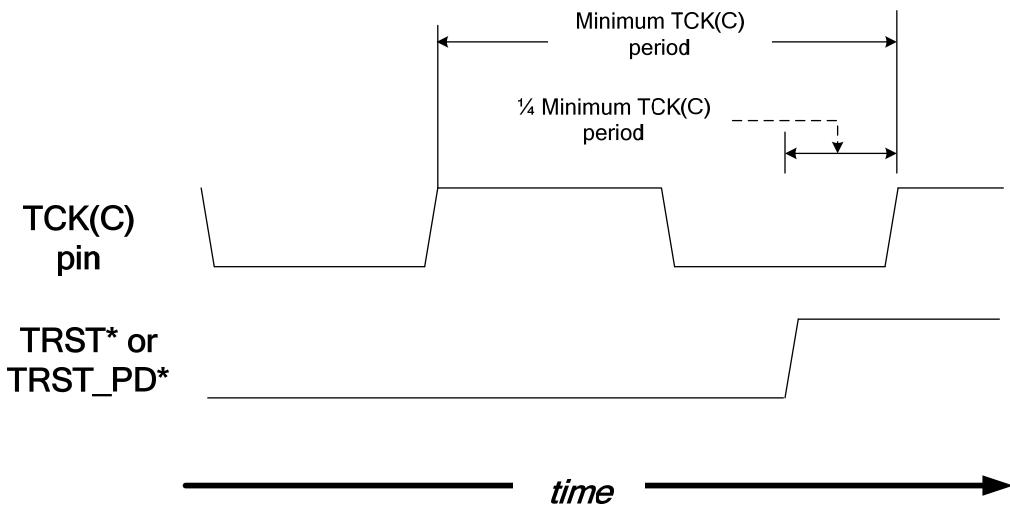
**Table 10-4 — Reset effects and references**

Reset	Effects:
Type-0	Type-1 effects + <ul style="list-style-type: none"> <li>– Initializes power control logic.</li> </ul>
Type-1	Type-2 effects + <ul style="list-style-type: none"> <li>– Causes power-down when enabled to do so</li> </ul>
Type-2	Type-3 effects + <ul style="list-style-type: none"> <li>– Initializes the Escape detection logic</li> </ul>
Type-3	Type-4 effects + <ul style="list-style-type: none"> <li>– System Test Logic (STL) via <i>nsys_trst</i></li> <li>– initializes the ADTAPC State Machine</li> <li>– Creates Offline-at-Start-up operation or Online operation with the use of the Standard Protocol</li> </ul>
Type-4	<ul style="list-style-type: none"> <li>– Default operation of the TAP.7 Controller with the use of the Standard Protocol</li> <li>– initializes TAP.7 Controller registers as per Table 9-2 and Table 27-5</li> <li>– initializes TAP.7 Controller state machines other than those managing Type-0 – Type-3 Resets</li> <li>– Creates the start-up behaviors shown as supported in Table 10-6 and described in Table 10-7</li> </ul>
Type-5	– Resets the STL by activating <i>nsys_trst</i>

- e) A Reset State Machine with the behavior shown in Figure 10-2 shall be implemented with the synchronous state changes occurring on the falling edge of the TCK(C).
- f) A Type-0 Reset shall be implemented when the T1 TAP.7 Controller power-control option is implemented.
- g) A Type-2 Reset shall be implemented when the T1 TAP.7 Controller power-control option is not implemented.
- h) The assertion and de-assertion of a Type-0, Type-1, and Type-2 Reset shall cause the assertion and deassertion of the *nsys\_trst* signal with the same period of assertion and deassertion as the Type-0–Type-2 Reset.
- i) The active state of *nsys\_trst* shall create the *Test-Logic-Reset* state in CLTAPCs connected to this signal.
- j) The effects of the Type-0, Type-1, and Type-2 Resets on the TAP.7 Controller shall appear as though they are asynchronous.
- k) A Type-3 Reset shall cause the assertion and de-assertion of *nsys\_trst*, provided two TCK(C) falling edges have occurred in the absence of a Type-0, Type-1, and Type-2 Reset.
- l) When a T1–T5 TAP.7 Controller begins operation Online and a Type-0–Type-2 TAP.7 Controller reset reaches its inactive state with the setup time defined by Rule 10.2.2 m), the

TAP.7 TAPC shall begin operation with the first rising edge of the TCK(C) with the value of the TMS(C) signal determining the TAPC state as follows:

- 1) A *Test-Logic-Reset* state to *Run-Test/Idle* state transition occurs, provided the TMS(C) value is a logic 0.
- 2) A *Test-Logic-Reset* state to *Test-Logic-Reset* state transition occurs, provided the TMS(C) value is a logic 1.
- m) The setup time referred to in Rule 10.2.2 l) shall be at most 25% of the TCK(C) period created with the maximum specified TCKC frequency of the TAP.7 Controller operating with the Standard Protocol as shown in Figure 10-4.



**Figure 10-4 — Test Reset setup time**

## Permissions

The specification permissions are as follows:

- n) The occurrence of a Type-0–Type-2 TAP.7 Controller reset may be captured asynchronously for the purpose of the subsequent synchronous initialization of the TAP.7 Controller or other logic.

## 10.3 Start-up options

### 10.3.1 Description

#### 10.3.1.1 Overview

The four start-up options introduced in 4.5.3 are as follows:

- IEEE 1149.1-Compliant T0 and above
- IEEE 1149.1-Compatible T2 and above
- IEEE 1149.1-Protocol Compatible T4 and above
- Offline-at-Start-up (Dormant) T0 and above

As stated previously, the IEEE 1149.1-Protocol Compatible option is only available with T4 and above TAP.7s, as this option involves the absence of the TDI and TDO functions.

These start-up options provide the following types of initial TAP.7 Controller operation:

- Online using the Standard Protocol with all the following configurations:
  - A narrow TAP.7
  - A wide TAP.7 with either:
    - The TDI and TDO functions available at start-up or
    - The TDI and TDO functions not available at start-up
  - Offline (dormant) waiting to be placed Online

The IEEE 1149.1-Compliant, IEEE 1149.1-Compatible, and IEEE 1149.1-Protocol Compatible Behavior options are invoked with Type-0–Type-4 Resets. The Offline-at-Start-up option is invoked with a Type-0–Type-3 Reset. With this start-up option, a Type-4 Reset creates either IEEE 1149.1-Compatible or IEEE 1149.1-Protocol Compatible Behavior, depending on whether the TDI and TDO functions are available following the reset.

#### **10.3.1.2 1149.1-compliant behavior start-up option**

With the IEEE 1149.1-Compliant Start-up option, following a Type-0–Type-4 Reset, the following will occur:

- TAP has 1149.1-Specified Behavior and 1149.7-Specified Behavior.
- Standard Protocol is used.
- TAP.7 Controller appears transparent.
- CLTAPC is selected.
- STL provides the TAP.7 Scan Path.
- APU Operating State is set to *BPA*.
- TMSC signal exhibits PU behavior from the point of power-up.
- TDIC and TDOC signals are present and provide the legacy TDI and TDO functions.
- The inputs for the EPU’s TDI and TDO signals (*epu\_tdi* and *epu\_tdo\_in*) are supplied by the TDIC and TDOC signals.

With T2 and above TAP.7s, the following will occur:

- The STL is a Scan Group Candidate (the SGC Register is set to a logic 1).
- The SCNFMT Register is set to the value specifying the use of the JScan0 Scan Format.

With this start-up option, the TAP.7 Controller may be programmed, the scan topology may be interrogated, and the TAP.7 Controller can be operated as though it is a TAP.1.

#### **10.3.1.3 1149.1-Compatible Start-up option**

With the IEEE 1149.1-Compatible Start-up option, following a Type-0–Type-4 Reset, the following will occur:

- TAP has IEEE 1149.1-Non-disruptive Behavior and 1149.7-Specified Behavior.
- Standard Protocol is used.

- CLTAPC is deselected (its state is parked to provide hot-connect protection).
- The EPU provides the TAP.7 Scan Path.
- A TAP.7 Controller reset establishes the *BPA* APU Operating State.
- TMSC pin exhibits PU behavior from the point of power-up.
- The TDI(C) and TDO(C) pins are present and provide the TDI and TDO functions.
- The inputs for the EPU TDI and TDO signals (*epu\_tdi* and *epu\_tdo\_in*) are supplied by the TDI(C) and TDO(C) pins.
- The STL is not a Scan Group Candidate (the SGC Register is set to a logic 0).
- The SCNFMT Register is set to the value specifying the use of the JScan1 Scan Format.

With this start-up option, the TAP.7 Controller may be programmed and the scan topology may be interrogated. The TAP.7 Controller can be operated as a TAP with one-bit IR and DR Scan Paths.

#### 10.3.1.4 IEEE 1149.1-Protocol Compatible

With the IEEE 1149.1-Protocol Compatible Start-up option, according to a Type-0-Type-4 Reset, the following will occur:

- The TAP has IEEE 1149.1-Other Behavior and IEEE 1149.7-Specified Behavior.
- The Standard Protocol is used.
- The EPU Bypass Bit provides the TAP.7 Scan Path.
- The *BPA* APU Operating State is established.
- The TMSC pin exhibits PU behavior from the point of power-up.
- The TDIC and TDOC pins are either not present or do not provide the TDI and TDO functions.
- The inputs for the EPU TDI and TDO signals (*epu\_tdi* and *epu\_tdo\_in*) are a logic 1.
- Hot-connect protection is provided as the CLTAPC is deselected at start-up.
- The initial TAP.7 scan-path length is one bit with a logic 0 capture value.
- The SGC Register is set to a logic 0.
- The SCNFMT Register is set to the value specifying the use of the JScan1 Scan Format.

With this start-up option, the TAP.7 Controller may be programmed and the scan topology may be interrogated, but the TDI and TDO functions are unavailable. Scan input data is logic 1 for both IR and DR Scans.

#### 10.3.1.5 Offline-at-Start-up option

The Offline-at-Start-up option provides a means to debug systems with Star Scan Topologies where the TAP.7 Controller, and possibly the STL, are powered-up and powered-down as part of normal operation. With this start-up option, a TAP.7 Controller can be as follows:

- Powered-down while the DTS communicates with other TAP.7 Controllers
- Powered-up while the DTS communicates with other TAP.7 Controllers
- Placed Online and synchronized to the DTS TAPC operation without resetting other TAPCs already operating

With this option, the CLTAPC state is moved to the *Run-Test/Idle* state prior to permitting the placement of the TAP.7 Controller Online.

When the Offline-at-Start-up option is utilized, the following operating conditions are created by the assertion and deassertion of a Type-0-Type-3 Reset:

- TAP has IEEE 1149.1-Other Behavior and IEEE 1149.7-Specified Behavior.
- Control Protocol is used.
- CLTAPC is deselected after the *Run-Test/Idle* state is created.
- TAP.7 Controller is Offline, awaiting placement Online once the TAPC state is *Run-Test/Idle*.
- CPA APU Operating State is established.
- APU Operating State is set to *CPA*.
- TMSC signal exhibits NB behavior from the point of power-up (see 12.3).
- Following the reset, the CLTAPC is selected, the ADTAPC and CLTAPC TAPC states are moved to the *Test-Logic-Reset* state and then to the *Run-Test/Idle* state where the CLTAPC is then deselected.
- TAP.7 Controller then waits to be placed Online.

With T2 and above TAP.7s, the following will occur:

- The SGC Register is set to a logic 0.
- The SCNFMT Register is set to the value specifying the use of the JScan1 Scan Format.

The Offline-at-Start-up function is described in detail in 11.10.7.

#### 10.3.1.6 Start-up behavior

The TAP.7's start-up behavior following a TAP.7 Controller reset is determined by the start-up option specified at the time the TAP.7 Controller is powered-up. When the start-up option is IEEE 1149.1-Compliant, IEEE 1149.1-Compatible, or IEEE 1149.1-Protocol Compatible, the ability to change operating states, create control levels, and generate TAP.7 Controller commands is immediately available following a Type-0 through Type-4 Reset. A T1 and above TAP.7 Controller is capable of creating control levels and processing TAP.7 Controller commands using the IEEE 1149.1 protocol. This requires the use of only the TCK(C) and TMS(C) pins [the TDI(C) and TDO(C) pins are not required]. It does not require the following:

- An understanding of the scan-path topology, for example, the number of chips on a board or whether the chips are connected in a series or star configuration.
- An understanding the Instruction or Data Register Scan Path lengths of a chip.
- The use of IEEE 1149.1 Instruction and Data Registers.

The start-up option can be programmable provided signals determining the start-up option are static while the TAP.7 Controller is powered-up and remains powered.

### 10.3.2 Specifications

#### Rules

The specification rules are as follows:

- Each subsequent specification in 10.3.2 shall apply to all TAP.7s.
- The TAP.7 Controller behavior resulting from a Type-0–Type-4 TAP.7 Controller reset shall be governed by its start-up option.
- The start-up options shown in Table 10-6 shall be determined as shown in Table 10-5.
- The start-up option of a TAP.7 Controller shall be one of the options shown in Table 10-6.

**Table 10-5 — Start-up option determination**

Type 0-Type-3 Reset ?	Offline-at-Start-up ?	Pin Width == Narrow ?	Auxiliary TDIIC and TDOC Pin functionality ?	Default Scan Format == JScan1 ?	Start-up Option
Yes	Yes	x	x	x	Offline-at-Start-up
No	x	Yes	x	x	IEEE 1149.1-Protocol Compatible
x	No				
No	x	No	Yes	x	IEEE 1149.1-Protocol Compatible
x	No				
No	x	No	No	Yes	IEEE 1149.1-Compatible
x	No				
No	x	No	No	No	IEEE 1149.1-Compliant
x	No				

**Table 10-6 — TAP.7 Class/Start-up option relationships**

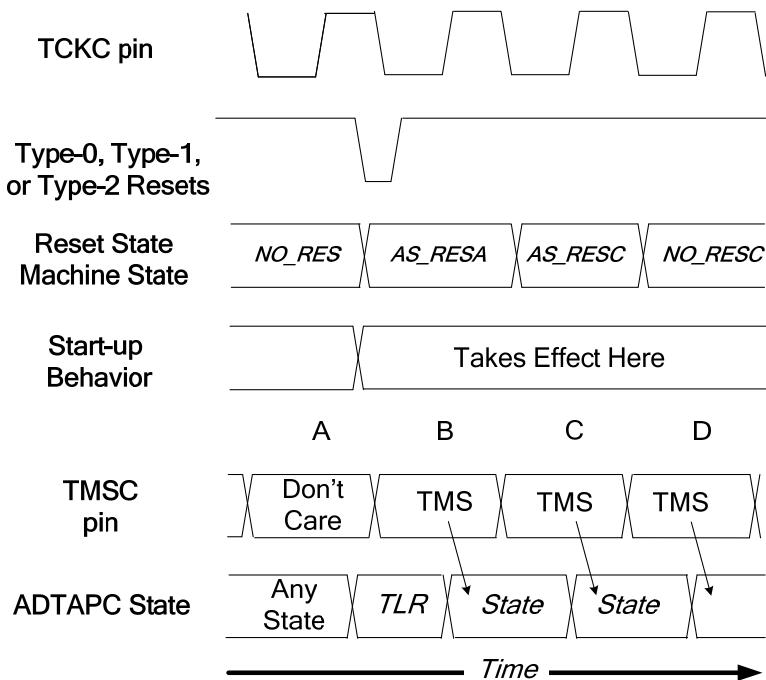
Start-up option	Start-up option allowed					
	T0 TAP.7	T1 TAP.7	T2 TAP.7	T3 TAP.7	T4 TAP.7	T5 TAP.7
IEEE 1149.1-Compliant	Yes	Yes	Yes	Yes	Yes	Yes
IEEE 1149.1-Compatible	No	No	Yes	Yes	Yes	Yes
IEEE 1149.1-Protocol Compatible	No	No	No	No	Yes	Yes
Offline-at-Start-up	Yes	Yes	Yes	Yes	Yes	Yes

- The operating characteristics created by the start-up options shown in Table 10-6 shall be governed by Table 10-7.

**Table 10-7 — Reset values for TAP.7 Start-up options**

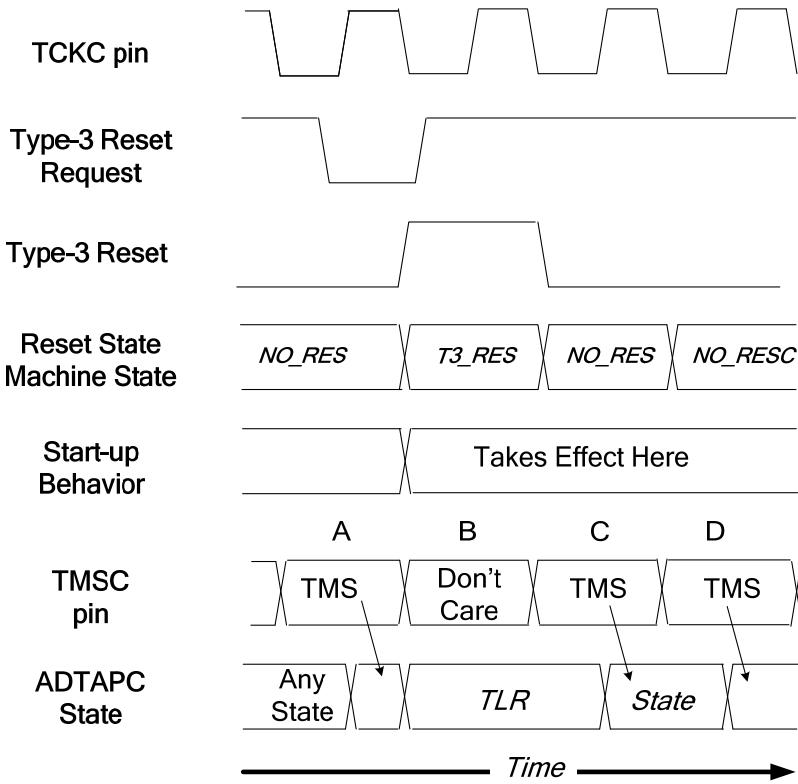
Start-up option	Before ADTAPC placement Online?	Default characteristics				
		ADTAPC placed Online with power-up	CLTAPC selected	SGC Reg. == 1	SCNFMT	Initial TMSC signal bias
IEEE 1149.1-Compliant	N/A	Yes	Yes	Yes	JScan0	PU
IEEE 1149.1-Compatible	N/A	Yes	No	No	JScan1	PU
IEEE 1149.1-Protocol Compatible	N/A	Yes	No	No	JScan1	PU
Offline-at-Start-up	Yes	No	No	No	JScan1	NB
	No	Yes	No	No	JScan1	NB
<b>NOTE—PU is pull-up and NB is no biasing (see 12.3).</b>						

- f) The signals generated external to the TAP.7 Controller affecting the start-up option shall be changed only when one of the following is true:
  - 1) The TAP.7 Controller power is not applied.
  - 2) The hardest chip-level reset is active.
- g) The effects of Type-0, Type-1, and Type-2 Resets with the IEEE 1149.1-Compliant, IEEE 1149.1-Compatible, and IEEE 1149.1-Protocol Compatible start-up options shall be governed by Figure 10-5.



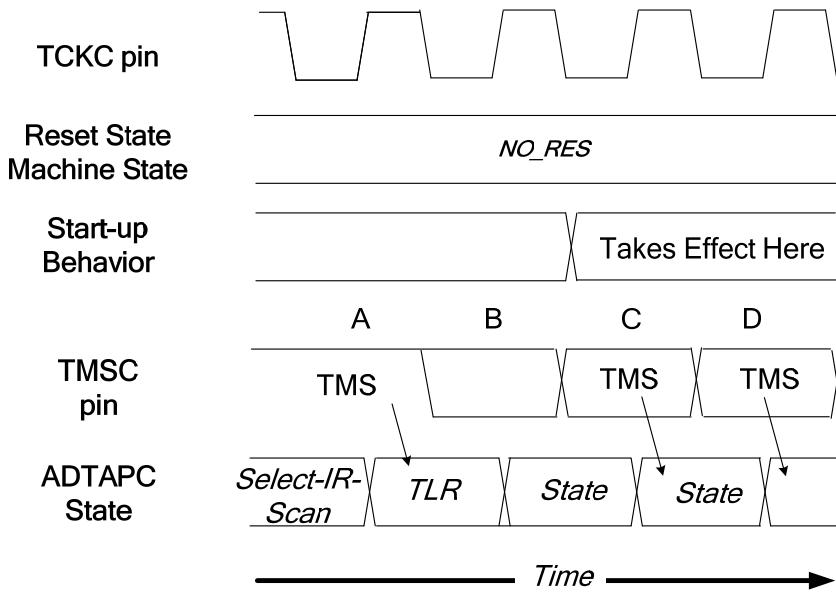
**Figure 10-5 — A Type-0 through Type-2 Reset invoking the use of the Standard Protocol**

- h) The effects of a Type-3 Reset with the IEEE 1149.1-Compliant, IEEE 1149.1-Compatible, and IEEE 1149.1-Protocol Compatible start-up options shall be governed by Figure 10-6.



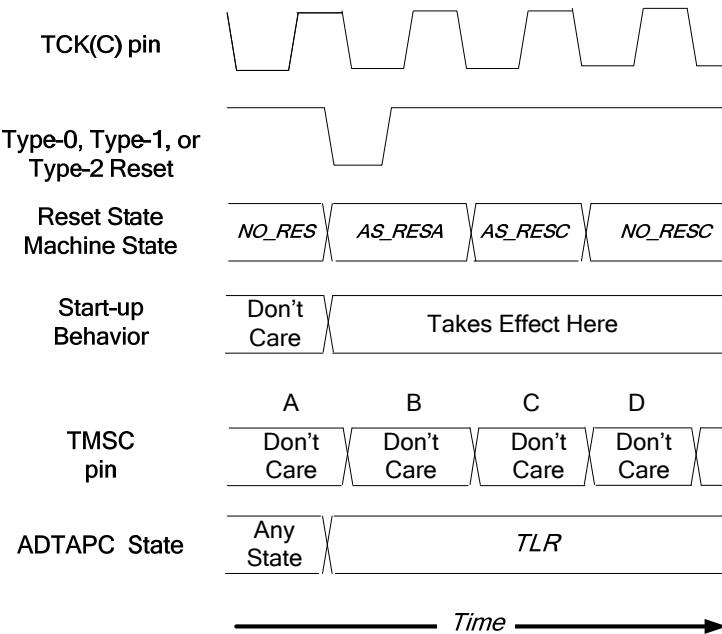
**Figure 10-6 — A Type-3 Reset invoking the use of the Standard Protocol**

- i) The effects of a Type-4 Reset shall be governed by Figure 10-7.



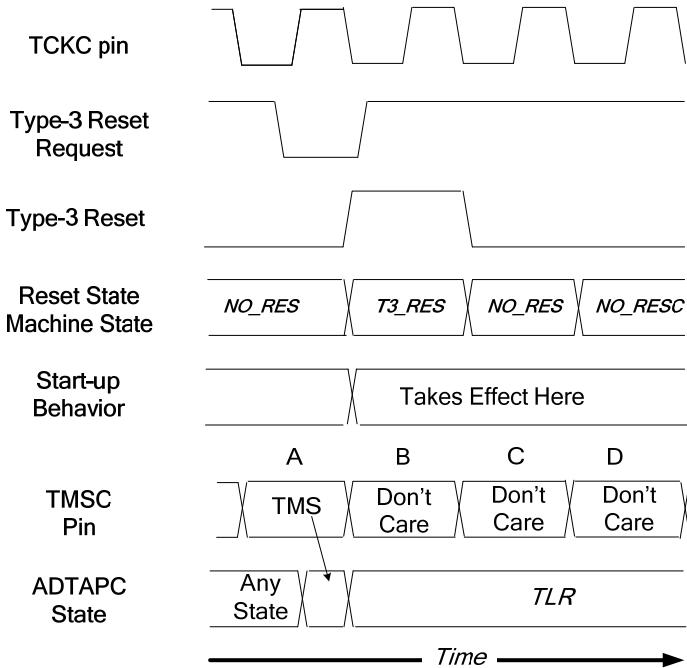
**Figure 10-7 — A Type-4 Reset invoking the use of the Standard Protocol**

- j) The effects of a Type-0–Type-2 Reset invoking Offline-at-Start-up operation shall be governed by Figure 10-8.



**Figure 10-8 — Type-0 through Type-2 Resets invoking Offline-at-Start-up operation**

- k) The effects of a Type-3 Reset invoking Offline-at-Start-up operation shall be governed by Figure 10-9.



**Figure 10-9 — A Type-3 Reset invoking Offline-at-Start-up operation**

## 10.4 Escape Detection

### 10.4.1 Description

#### 10.4.1.1 Overview

An Escape overlays control information onto the information normally transferred with the TCK(C) and TMS(C) signals without changing the normal information conveyed in a bit period. A TAP.7 Controller with an RSU interprets the count of the number of TMS(C) edges while TCK(C) is a logic 1 as one of four Escapes. Each Escape has a different function and TMSC edge count. These Escapes, their edge counts, and their functions are described as follows:

- Custom (2 or 3 edges)      Ends scan and transport transfers, may be used for other purposes with another technology
- Deselection (4 or 5 edges)      Initiates the deselection of Online technologies
- Selection (6 or 7 edges)      Initiates a Selection Sequence that places the ADTAPC Online or Offline
- Reset ( $\geq 8$  edges)      Resets all technologies (generates a Type-3 Reset)

The DTS creates an Escape by generating one or more TMS(C) edge pairs while the TCK(C) signal is a logic 1 value as shown in Figure 10-10. The DTS never creates an Escape with an odd number of TMS(C) edge pairs. The TAP.7 interprets even and odd TMS(C) edge counts beginning with two as having the same meaning. This accommodates the case where a change of state of the TMS(C) signal related to data occurs while the TCK(C) is a logic 1. This can occur because of the following reasons:

- Propagation delays when a change in state of the TMS(C) signal is initiated by a falling edge of the TCK(C)

- A change in state of the TMS(C) signal that is initiated by a rising edge of the TCK(C)

This approach allows the interoperability of technologies that change their data/control signal with the different edges of the clock and do not use signaling similar to that used to create Escapes.

The relationship between the TAP.7 Class and the deployment of Escape Detection is shown in Table 10-8.

**Table 10-8 — TAP.7 Class/Escape deployment relationships**

<b>Capability</b>	<b>TAP.7 Class</b>			
	<b>T0-T1</b>	<b>T2</b>	<b>T3</b>	<b>T4 and above</b>
Escape Detection	Only with RSU	Only with RSU	Mandatory	Mandatory

Escape Detection is mandatory for both T0-T2 TAP.7s implementing the RSU and for T3-T5 TAP.7s. Escape Detection becomes mandatory for T0-T3 TAP.7s when an RSU is implemented. Their detection and use may easily be added as a wrapper to TAP.1 and many other technologies without changing the underlying technology in any way.

Because the TMS(C) signal is used as a clock with Escapes, care should be taken to minimize board-level transmission line effects to ensure their proper operation. The board-level electrical recommendations found in Annex F **will be followed to ensure** proper operation.

#### **10.4.1.2 Detection**

Escape Detection:

- Occurs, provided a Type-0-Type-2 Reset is not asserted
- Begins and ends while the TCK(C) signal is a logic 1
- Uses the TMS(C) signal as a clock while the TCK(C) signal is a logic 1

Detection of an Escape is dependent on the criteria listed above and is not affected by the drive history of the TCK(C) and TMS(C) signals.

The detection of the four Escapes is shown in Figure 10-10.

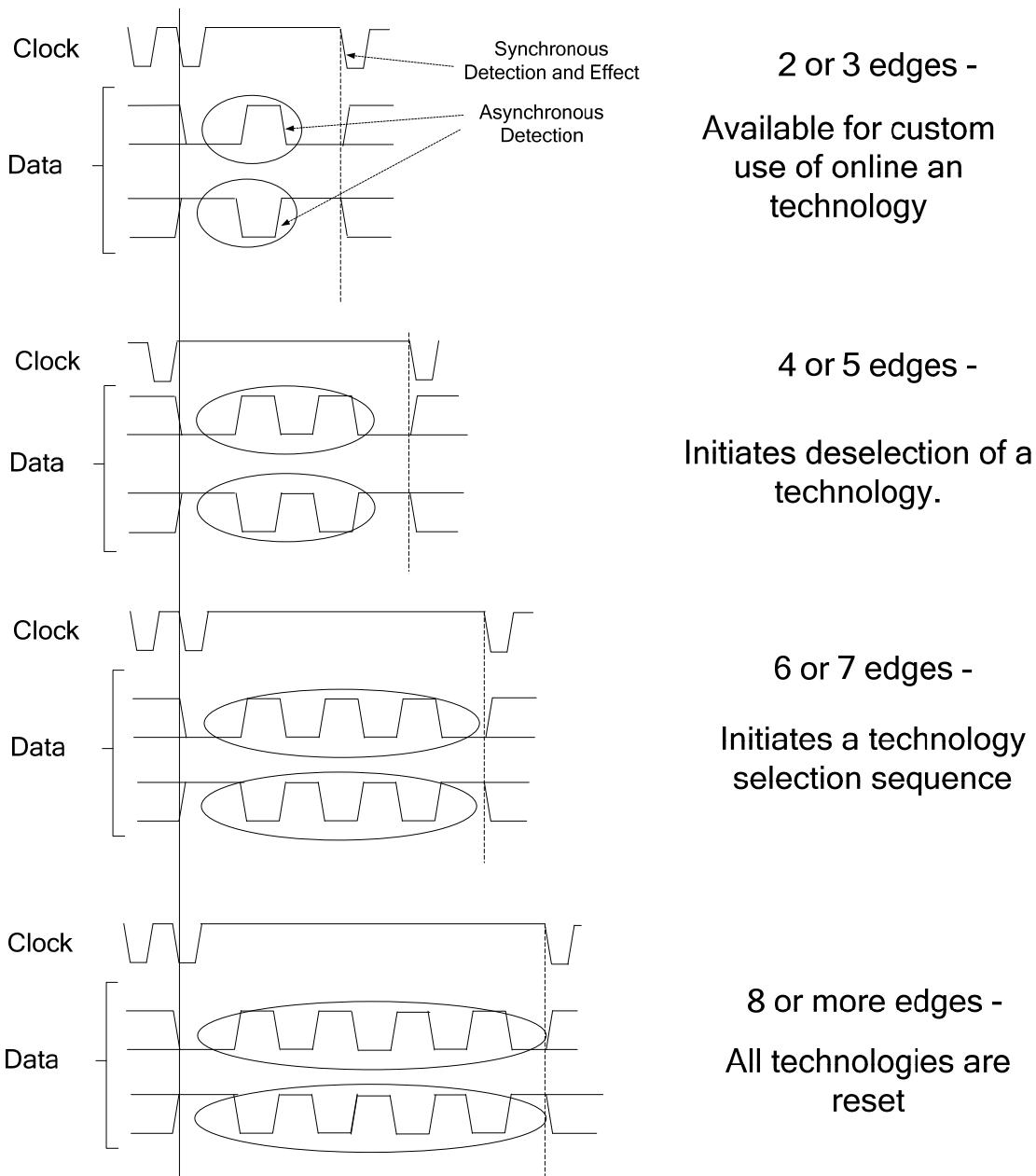


Figure 10-10 — Escape Detection

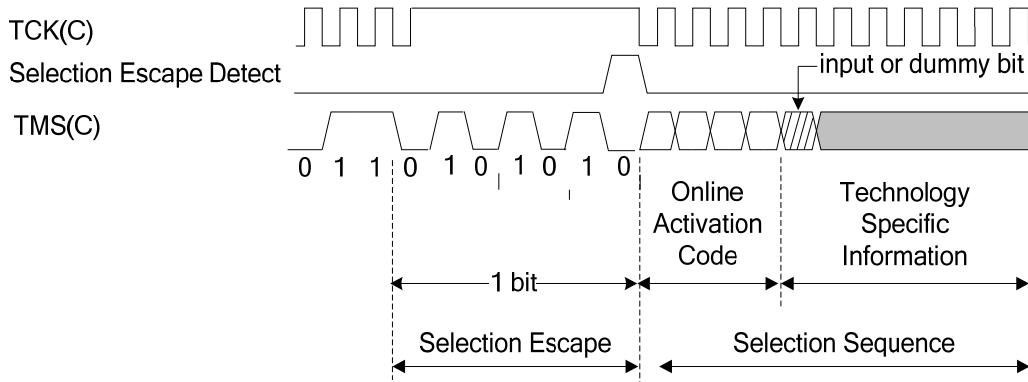
#### 10.4.1.2.1 Custom Escape

An Online technology can use the Custom Escape in any manner it chooses. It is not used with T0–T3 TAP.7s. It is used to indicate an EOT of certain scan and transport transfers with T4 and above TAP.7s.

#### 10.4.1.2.2 Selection and Deselection Escapes

The DTS may utilize Selection and Deselection Escapes to manage the selection and deselection of technology branches sharing a DTS connection as described in Clause 11. A Deselection Escape places all technologies that implement an RSU Offline. A Selection Escape informs these technologies that a Selection Sequence follows. The Selection Sequence (see 11.7) begins with the Activation Code and may

be followed with additional information that specifies technology-specific selection criteria as shown in Figure 10-11.



**Figure 10-11 — Selection Escape followed by a Selection Sequence**

The effects of Selection and Deselection Escapes, restrictions on their use (qualification by a technology), and other aspects of ADTAPC selection/deselection are described in Clause 11.

#### 10.4.1.3 Reset Escape

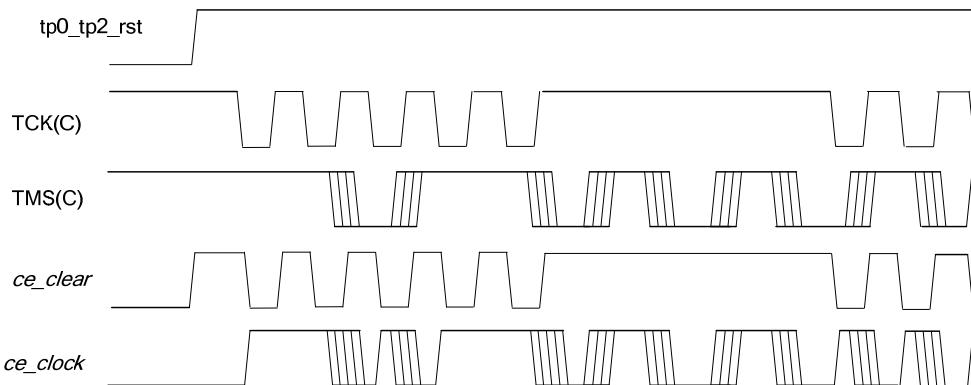
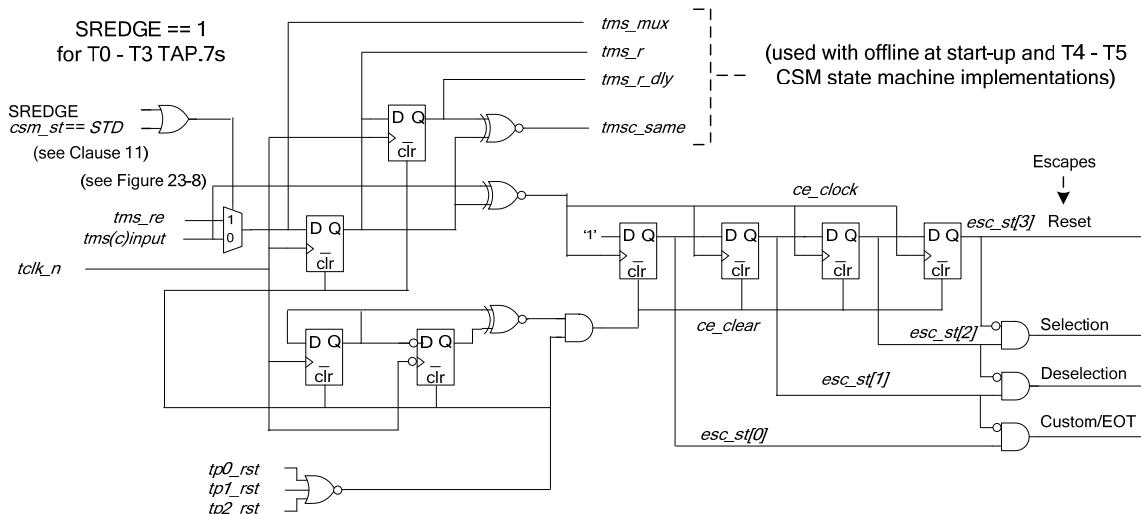
A Reset Escape initializes all technologies sharing the TCK(C) and TMS(C) connectivity by generating a reset that initializes the operation of these signals. It is recommended this reset occur with the falling edge of the TCKC. An exception to this is the TMS(C) drive management. The detection of the Reset Escape immediately initiates the TMSC signal's Dormant Drive Policy, ensuring the TMSC signal is at a high-impedance level when the TCK(C) signal returns to a logic 0. With a TAP.7 Controller, a Reset Escape generates a Type-3 Reset beginning with the falling edge of the TCK(C) following the asynchronous detection of the event.

#### 10.4.1.4 Timing considerations

Because the TCK(C) signal is a logic 1 while the TMS(C) signal is toggled for Escape generation, it is only practical to generate Escapes when the DTS sources the TCK(C) signal. The decode of even ( $n$ ) and odd ( $n + 1$ ) TMS(C) edges as the same Escape accommodates designs where, at the highest TCK(C) signal frequency, there is a possibility of a data transition on the TMS(C) signal initiated by the falling edge of the TCKC may be delayed sufficiently (e.g., by a propagation delay) so as to occur following the rising edge of the TCKC. In this case, this edge creates an odd TMSC edge-count edge, as it is counted in addition to the even number of TMSC edges purposely created to generate an Escape.

#### 10.4.1.5 An approach to implementing Escape Detection

One approach to the implementation of Escape Detection is shown in Figure 10-12. The use of other approaches should also be considered.



**Figure 10-12 — Conceptual view of the Escape Detection function**

The Escape Detection logic shown in this figure operates as follows. Type-0–Type-2 Resets initialize this logic. The *ce\_clear* signal is asserted by a falling edge of the TCK(C) and released by a rising edge of the TCK(C).

The thermometer-code counter is incremented by a rising edge of the *ce\_clock* signal after the *ce\_clear* signal is deasserted. The *ce\_clock* signal is created with the XNOR of the TMS(C) signal and the last TMS(C) signal value sampled with the falling edge of the TCK(C) signal. This ensures the first edge created by a normal change in the TMSC signal creates a falling edge of the *ce\_clock* signal. This edge is therefore discarded by the thermometer-code counter. When an even number of edges is created, there will be the same number of *ce\_clock* signal rising edges and falling edges. When an odd number of edges is created, there will be an even number of *ce\_clock* signal rising edges and an odd number of falling edges. The Escape type is decoded from the thermometer-code counter value as shown in Figure 10-12. The decoded Escapes are used as inputs of the logic trees of Flip-Flops clocked by the falling edge of the TCK(C). The Reset Escape is also used to inhibit the drive of the TMSC pin with a T4–T5 TAP.7 as mentioned previously.

When the DTS is connected to the TS, there are three cases to consider. The Escape detection is in a state created by the following:

- Initialization
- Connecting the DTS to the TS

- Operation prior to the DTS/TS connection being broken

In each of these cases, the TCK(C) signal value is a logic 1 and the TMSC signal is an input with the TCK(C) signal value a logic 1. The TMS(C) signal characteristics are determined by the TAP.7 Controller start-up option following a Type-0–Type-2 Reset and is the last TAP.7 Controller state, if the DTS/TS connection was broken without a subsequent TAP.7 Controller reset. The TMSC signal may be an input with pull-up characteristics, an input with keeper characteristics, or an input with NB characteristics. In all these cases, the DTS may drive the TMS(C) signal while the TCK(C) signal is a logic 1. This means the DTS may generate a Reset Escape without causing a change in the TAPC, provided the TCK(C) bias is PU (see 12.3).

## 10.4.2 Specifications

### Rules

The specification rules are as follows:

- a) Each subsequent specification in 10.4.2 shall only apply to any of the following unless stated otherwise:
  - 1) A T0–T2 TAP.7 implementing the RSU.
  - 2) A T3–T5 TAP.7.
- b) The use of the term “minimum period of the TCK(C) signal” used in each subsequent specification in 10.4.2 in reference to time shall mean the TCK(C) signal period when the TAP.7 is operated at its maximum specified frequency using the Advanced Protocol.
- c) An Escape shall be detected and its associated function performed, provided all the following timing constraints are met:
  - 1) The first TMS(C) signal edge associated with the Escape follows the rising edge of the TCK(C) signal by a minimum TCK(C) signal period.
  - 2) The second through the last TMS(C) signal edges associated with the Escape follow a prior TMS(C) edge by at least one minimum TCK(C) period.
  - 3) A TCK falling edge follows the last TMS(C) edge associated with the Escape by a minimum of one TCK(C) period.
- d) The count of the TMS(C) signal edges while the TCK(C) signal is a logic 1 shall begin at zero immediately after any of the following:
  - 1) A Type-0 Reset.
  - 2) A Type-1 Reset.
  - 3) A Type-2 Reset.
  - 4) A TCK(C) signal value of logic 0.
- e) The count of the TMS(C) signal edges related to Escape Detection shall be incremented when all of the following are true:
  - 1) The Type-0 Reset is inactive.
  - 2) The Type-1 Reset is inactive.
  - 3) The Type-2 Reset is inactive.
  - 4) The TCK(C) value is a logic 1.
  - 5) Any of the following are true:
    - i) The TMS(C) signal switches from a logic 1 to a logic 0.

- ii) The TMS(C) signal switches from a logic 0 to a logic 1.
- f) A TAP.7 Controller shall detect an Escape as described by Rule 10.4.2 c) independently of changes in the TMS(C) value that occurred while the TCK(C) signal was a logic 0.
- g) The count of the TMS(C) signal edges while the TCK(C) signal is a logic 1 shall be interpreted as shown in Table 10-9.

**Table 10-9 — Escapes**

<b>Escape</b>	<b>Edge count</b>	<b>Function</b>
Custom	2 or 3	End of Transfer for T4 and above, a no-operation otherwise
Deselection	4 or 5	Conditional deselection
Selection	6 or 7	Conditional Selection Sequence initiation
Reset	>7	Type-3 Reset Request generation

NOTE—The first edge of an odd number of TMSC edges occurring while TCK(C) is a logic 1 shall be presumed to be caused by establishing the TMSC value for the bit period.

- h) The data value of the bit period coincident with an Escape shall be determined by the TMS(C) value prior to the falling edge of TCK(C) following the Escape.
- i) A TAP.7 Controller that does not implement an RSU shall ignore the signaling creating the detection of Escapes shown in Table 10-9.
- j) A TAP.7 Controller with an RSU shall detect all Escapes shown in Table 10-9.
- k) The detection of Escapes shall utilize only the TCK(C) and TMS(C) signals (a functional clock shall not be used in Escape Detection).
- l) For a T0–T2 TAP.7 implementing the RSU, the RSUS bit of Configuration Register Zero shall be set to a logic 1 when the RSU function is implemented and a logic 0 otherwise.
- m) For a T3–T5 TAP.7, the RSUS bit of Configuration Register Zero shall be set to a logic 0.

## Permissions

The specification permissions are as follows:

- n) A T0–T2 TAP.7 may implement Escape Detection as described by the Rule 10.4.2 a) through Rule 10.4.2 m) provided an RSU is implemented.

## 10.5 Selection Alert

### 10.5.1 Description

#### 10.5.1.1 Overview

Selection Escapes and Selection Alerts provide selection functionality applicable to different sets of use cases. The primary use of Selection Alerts is providing a means to initiate a Selection Sequence with other technologies that do not utilize Escapes.

A Selection Alert is a specific 128-bit sequence transmitted by a DTS that is used to inform the target that a Selection Sequence will follow. It should be preceded by a minimum of eight logic 1s to ensure its detection. Because of its length, it is **extremely** unlikely to occur during any other debug or functional use of the TAP that may be shared between several technologies. It may only be used when all the technologies sharing a DTS connection are Offline as the TMS(C) signal cannot be used concurrently to both create the

Selection Alert and exchange TMS(C) data. The use of the Selection Alert with an Online technology would cause the Online technology to malfunction. In this case, the TMS(C) signal cannot be used concurrently to both create the Selection Alert and exchange TMS(C) data.

Because a Selection Alert is a redundant function within a TAP.7 Controller and is less flexible than a Selection Escape, the chip architect may consider this fact in the decision to implement or not implement this option.

### 10.5.1.2 Selection Alert Bit Sequence

The 128-bit sequence forming the Selection Alert may be expressed in several formats, as follows:

- Binary, transmitted MSB first:

0100\_1001\_1100\_1111\_1001\_0000\_0100\_0110\_1010\_1001\_1011\_0100\_1010\_0001\_0110\_0001  
1001\_0111\_1111\_0101\_1011\_1011\_1100\_0111\_0100\_0101\_0111\_0000\_0011\_1101\_1001\_1000

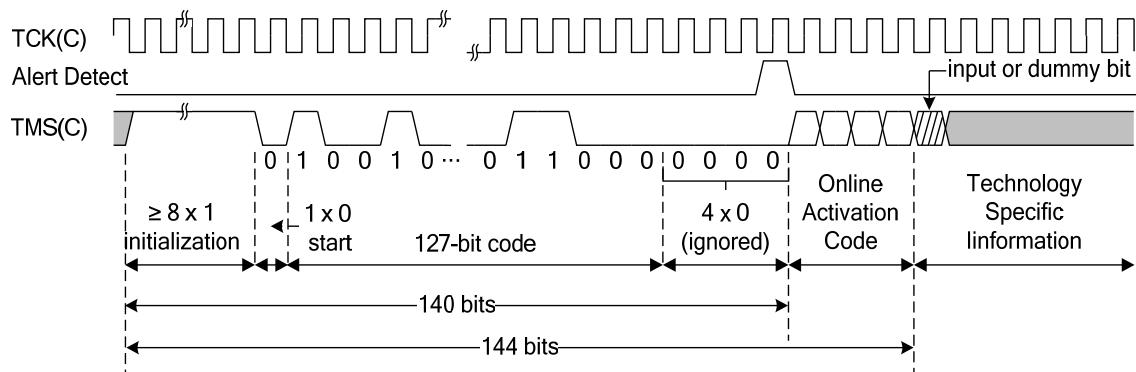
- Hexadecimal, transmitted MSB first: 0x49CF9046\_A9B4A161\_97F5BBC7\_45703D98
- Hexadecimal, transmitted LSB first: 0x19BC0EA2\_E3DDAFE9\_86852D95\_6209F392

Within this bit sequence, the longest run of successive bits of logic 1s is seven. Thus, the DTS transmits a preamble of eight or more bits of logic 1s to initialize a detector. The detector then waits for a logic 0 bit as a start bit for the Selection Alert Bit Sequence.

The Selection Alert generated by the DTS is followed by four padding-bit periods (of logic 0), which may be used by the TAP.7 Controller for pipelining purposes. It is expected that the TAP.7 Controller (or other technology) completes the processing of the Selection Alert just as the last of these bits is at the TMS(C) signal. These bit periods permit a detector to incorporate a synchronizer-pipeline of four stages. The use of such a synchronizer makes a detector capable of rejecting any bit sequences other than the Selection Alert Bit Sequence transmitted using other technologies or functional interfaces while a technology is Offline.

Figure 10-13 shows the start and end of the 127-bit Selection Alert Bit Sequence preceded by eight or more logic 1 preamble bits and a logic 0 start bit. The Selection Alert is followed by the four-bit periods (logic 0s) and then a four-bit Activation Code selecting a technology of interest. Thus, the minimum total number of bits transmitted by the DTS is 144 when the four-bit Activation Code is included.

The bit period following the Online Activation Code can only be driven by the DTS. This ensures there is no drive overlap of DTS and Technology drive (a drive conflict). This bit may be an Extension Code input bit. This provides a transition between the Online Activation Code and the selected technology's protocol when the Technology generates output following the Online Activation Code. There is no need to implement the dummy bit when the TS does not intend to drive the TMS(C) signal in the bit period following the Online Activation Code.



**Figure 10-13 — Selection-Alert Bit Sequence**

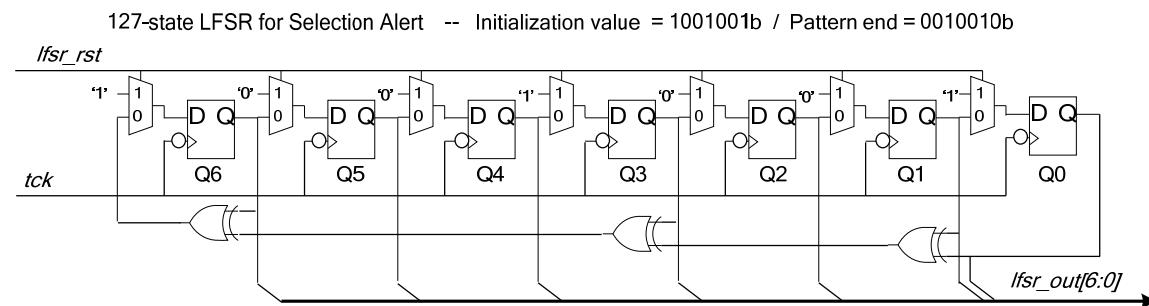
### 10.5.1.3 Selection Alert detection

The DTS will precede the Selection Alert Bit Sequence by a preamble of eight or more bits of logic 1 to synchronize a detector with the logic 0 start-bit preceding the Selection Alert Bit Pattern. The Linear Feedback Shift Register (LFSR) is initialized to 1001001b when there is a mismatch in the bit sequence. With this initialization the LFSR begins generation of the bit sequence to be detected following a logic 0 bit. The mismatch may occur on this start bit value with the LFSR generating the bit sequence to be detected in the following bit period. The Selection Alert Detector is initialized while all technologies are Offline.

### 10.5.1.4 An approach to implementing Selection Alerts

The approach to implementing Selection Alerts shown in this subclause are to be viewed only as an example of Alert-Sequence Detection. Other approaches to implementing Alert-Sequence Detection should also be considered.

The Selection Alert Bit Sequence is formed by using a logic 0 start bit, followed by the least-significant bit of a particular seven-bit LFSR moving through 127 states, followed by four-bit periods that should be logic 0 but are ignored by the receiver. The LFSR is initialized to the state 1001001b and its MSB input (LFSR[6]) is fed with the XOR of the current value of bits 6, 3, 1, and 0. This circuit is shown in Figure 10-14.



**Figure 10-14 — Selection Alert LFSR for bit sequence generation**

Logic supporting the use of the LFSR generates the *lfsr\_rst* signal shown in Figure 10-14. This signal is a logic 1 when the LFSR is not being used or is being used and the incoming bit pattern does not match the bit pattern generated by the LFSR.

Conceptually, the Selection Alert Bit Sequence is delayed four TCK(C) periods with a shift register. This shift register delays the TMSC input the same number of TCK(C) periods as there are zeros trailing the 127-state input. In most instances, this shift register is used as a synchronizer. This delay may be distributed between the input and output of the Selection Alert Detector as desired. The examples shown in this standard show all of this delay on the input of the Selection Alert Detector.

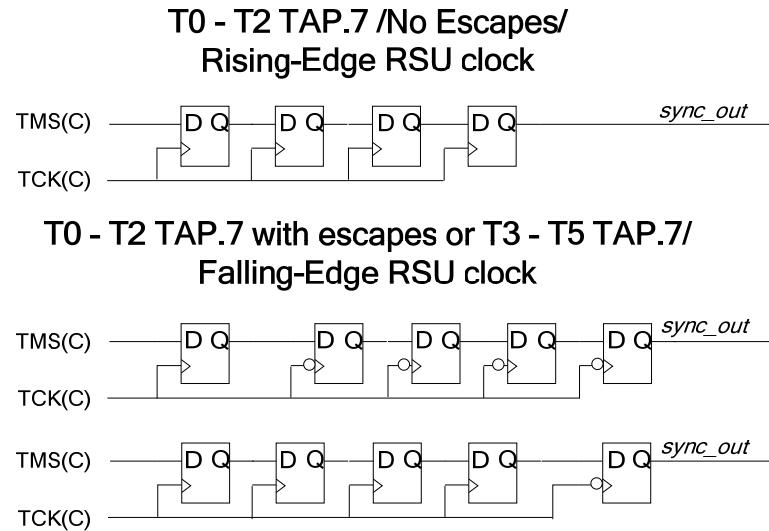
The first stage of this shift register is clocked with the rising edge of TCK(C). The clocking of the remainder of the RSU is chosen to be rising- or falling-edge based on the use of Escapes and/or the chip architect's preference. The RSU is operated with the falling edge of the TCK(C) when:

- A T0–T2 TAP.7 is implemented with Escape option (i.e., selection/deselection for boundary scan across multiple branches)
- A T3–T5 TAP.7 is implemented

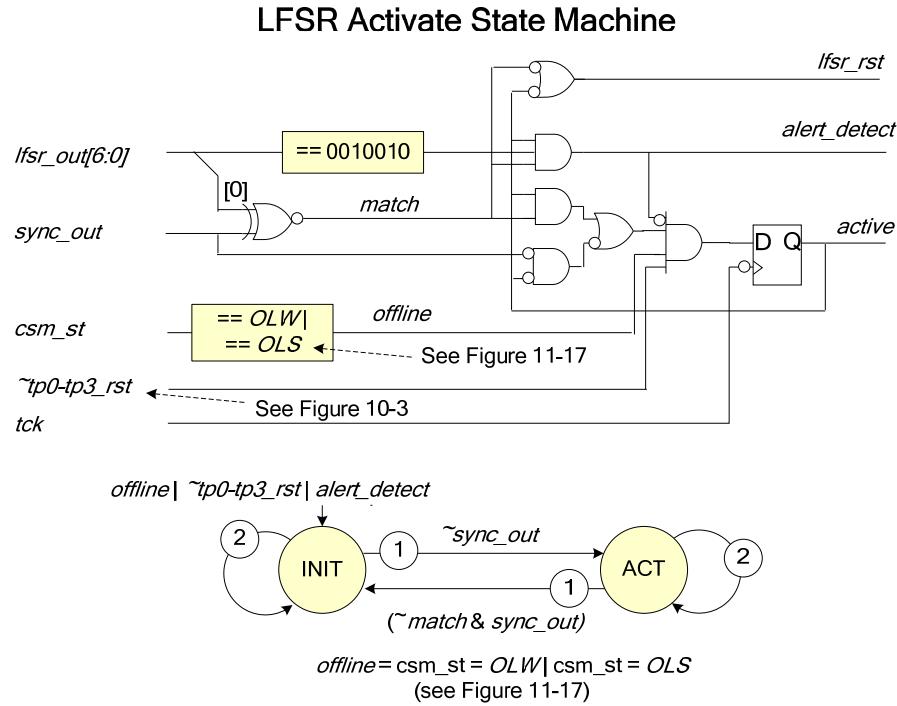
The clocking options for this delay are shown in Figure 10-15. When the RSU is clocked with the falling edge of TCK(C), an additional register stage is added running on the falling edge of the TCK(C) to properly align the synchronizer output to the RSU clock.

A two-state LFSR Activate State Machine is used to detect the start-bit and facilitate the generation of the alert detect signal as shown in Figure 10-16. This state machine moves from the *Initialize (INIT)* state to the *Active (ACT)* state when the TAP.7 Controller is Offline and there is a logic 0 start-bit that precedes the 127-bit Selection Alert pattern. It remains in this state until either the Selection Alert is detected or a pattern mismatch occurs.

With this example, the Selection Alert Detector is a combination of Figure 10-14, Figure 10-15, and Figure 10-16.



**Figure 10-15 — TMSC delay options**



**Figure 10-16 — Selection Alert support logic**

## 10.5.2 Specifications

### Rules

The specification rules are as follows:

- Each subsequent specification in 10.5.2 shall only apply to T0–T5 TAP.7s implementing Selection Alert Bit Sequences.
- The Selection Alert Detection shall have both initialized and active states.
- A Selection Alert Detector shall sample the TMS(C) signal value using the rising edge of the TCK(C) signal.
- The sampled TMS(C) signal described in Rule 10.5.2 c) shall be delayed four-bit periods before being used to determine the Selection Alert Detector's state.
- A Selection Alert Detector shall be set to its initialized state when any of the following occur:
  - The Control State Machine state (see 11.10) is a state other than *OLS* or *OLW*.
  - A Selection Alert is detected as specified by Rule 10.5.2 j).
  - All the following are true:
    - The Selection Alert Detector's state is the initialized state.
    - The sampled TMS(C) signal value described in Rule 10.5.2 d) is a logic 1.
  - All the following are true:
    - The Selection Alert Detector's state is the active state.

- ii) The sampled bit value described in Rule 10.5.2 d) is not equal to the reference bit value described in Rules 10.5.2 g), h), and i).
- iii) The sampled TMS(C) signal value as described in Rule 10.5.2 d) is a logic 1.
- f) A Selection Alert Detector shall be set to its active state, provided Rule 10.5.2 e) is not satisfied.
- g) A Selection Alert Detector shall use bit 126 of the reference bit pattern specified by Rule 10.5.2 h) sequence as the comparison value for the following TCK(C) signal period, provided any of the following conditions occur:
  - 1) All the following:
    - i) The Selection Alert Detector's state in the last TCK(C) period was the *Initialized* state.
    - ii) The Selection Alert Detector's state in this TCK(C) period is the *Active* state.
  - 2) All the following:
    - i) The Selection Alert Detector's state in this TCK(C) period is the *Active* state.
    - ii) The sampled TMS(C) signal value as described in Rule 10.5.2 d) is a logic 0.
    - iii) The sampled bit value described in Rule 10.5.2 d) is not equal to the reference bit value described in Rules 10.5.2 g), h), and i).
- h) The Selection Alert Detector shall use the next sequential bit of the reference bit pattern specified by Rule 10.5.2 i), provided Rule 10.5.2 g) is not satisfied.
- i) The reference bit pattern shall have the following characteristics:
  - 1) The 127-bit sequence having the value shown in Table 10-10.

**Table 10-10 — Selection Alert sequence**

MSB
<b>126</b>
100 1001 1100 1111 1001 0000 0100 0110 1010 1001 1011 0100 1010 0001 0110 0001
<b>LSB</b>
<b>63</b>
1001 0111 1111 0101 1011 1011 1100 0111 0100 0101 0111 0000 0011 1101 1001 1000
<b>0</b>

- 2) The next sequential bit as referenced in Rule 10.5.2 h) is the currently used bit number minus one.
- j) A Selection Alert shall be detected and its associated function performed, provided all the following conditions are met:
  - 1) The Selection Alert Detector's state is active.
  - 2) The sampled bit value described in Rule 10.5.2 d) has equaled the reference bit value described in Rules 10.5.2 g) and h) for each of the last 126 TCK(C) periods.
- k) A Selection Alert shall be detected and its associated function performed, provided all the following conditions are met:
  - 1) The Selection Alert Detector's state is active.
  - 2) The sampled bit value described in Rule 10.5.2 d) is equal to the reference bit value described in Rules 10.5.2 g) and h) during this TCK(C) period.
- l) The SEL\_ALERTS bit of Configuration Register Zero shall be set to a logic 1 when the TAP.7 Controller implements a Selection Alert Detector and set to a logic 0 otherwise.
- m) A Selection Alert Detector shall only be implemented when an RSU is implemented.

## Permissions

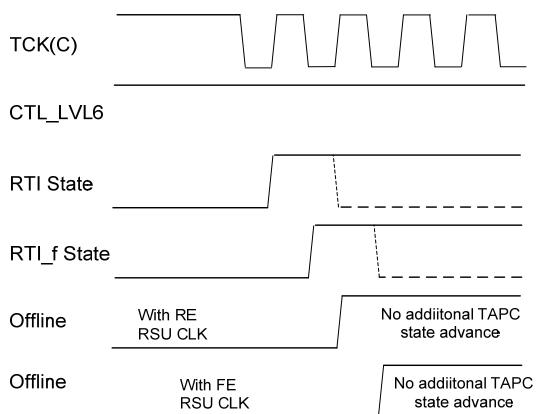
The specification permissions are as follows:

- n) A T0–T5 TAP.7 may implement a Selection Alert as described by the Rule 10.5.2 a) through Rule 10.5.2 m).

## 10.6 Deselection Alert

### 10.6.1 Description

A Deselection Alert is provided as an option. A Deselection Alert initiates Offline operation using a combination of Control Level Six and the *Run-Test/Idle* state as shown in Figure 10-17. The effects of a Deselection Alert are similar to the effects of issuing a Deselection Escape when the ADTAPC state is *Run-Test/Idle*.



**Figure 10-17 — Deselection Alert operation**

Note that a TAP.7 Controller that is placed Offline using the Deselection Alert will be parked in the:

- *Select-DR-Scan* state, provided there is a stay in the *Run-Test/Idle* state of one TCK(C) period
- *Run-Test/Idle* state, provided there is a stay in the *Run-Test/Idle* state of more than one TCK(C) period

Because a Deselection Alert is a redundant function within a TAP.7 Controller and is less flexible than Deselection Escape, the chip architect may consider this fact in the decision to implement or not implement this option.

### 10.6.2 Specifications

#### Rules

The specification rules are as follows:

- a) Each subsequent specification in 10.6.2 shall only apply to a T0–T5 TAP.7 implementing initiating Offline operation with a Deselection Alert (a combination of the *Run-Test/Idle* state and Control Level Six).
- b) The timing of the *Online-to-Offline* selection state transition created with a Deselection Alert shall be governed by Figure 10-17.

- c) The DSEL\_ALERTS bit of Configuration Register Zero shall be set to a logic 1 when a TAP.7 Controller implements Deselection Alerts and set to a logic 0 otherwise.
- d) A Deselection Alert shall only be implemented when an RSU is implemented.

## Permissions

The specification permissions are as follows:

- e) A T0–T5 TAP.7 may implement a Deselection Alert as described by the Rule 10.6.2 a) - Rule 10.6.2 d).

## 10.7 Programming considerations

### 10.7.1 Resets

Type-2, Type-3, and Type-4 Resets may be used at any time. A Type-5 Reset is used as described in Clause 19 as its use can create ADTAPC/ CLTAPC synchronization issues when not used properly.

### 10.7.2 Escapes

When using Escapes, the DTS will do the following:

- Ensure the TMS(C) signal logic level at the beginning and end of the bit period coincident with an Escape is the value required to convey the TMS(C) value for the bit period
- Separate the first TMS(C) edge associated with an Escape from the preceding TCK(C) falling edge and other TMS(C) edges associated with an Escape by at least a minimum of one TCK(C) period [see Rule 10.4.2 b)]
- Separate the last TMS(C) edge associated with an Escape and the subsequent TCK(C) falling edge by at least a minimum of one TCK(C) period
- Consider the behavior of the TAP.7 Controller power-down modes described in 18.10 when creating a logic 1 TCK(C) for extended periods for Escape generation

### 10.7.3 Selection Alerts

Recall that the Selection Alert may only be used when all TAP.7 Controllers are Offline as the operation of an Online TAP.7 Controller would be corrupted by a Selection Alert. This means they cannot be used with boundary-scan testing spanning multiple branches as the operation of an Online TAP.7 Controller would be corrupted by a Selection Alert.

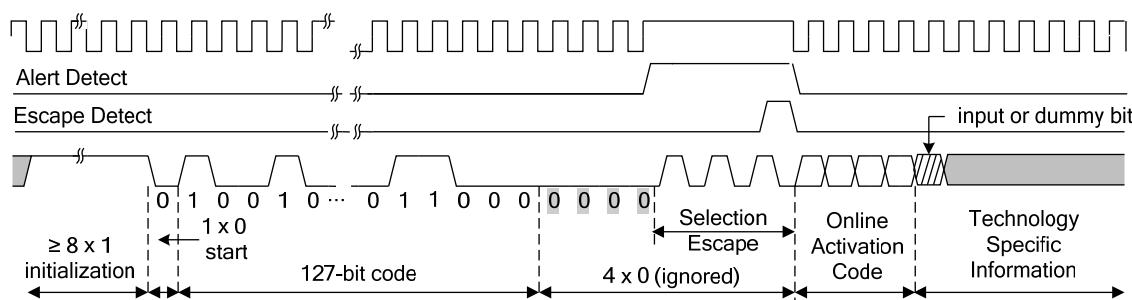
### 10.7.4 Test and debug

Boundary-scan testing involving multiple TAP.1 and TAP.7 Branches requires the use of Deselection and Selection Escapes. Concurrent debug operations (for example running and halting applications simultaneously in multiple STLs) also requires the use of Deselection and Selection Escapes. Both of these operations utilize Series-Equivalent Scans. Either a Selection Escape or a Selection Alert may be used when deselecting one branch and selecting another for debug purposes, provided all branches are deselected before another branch is selected.

### 10.7.5 Concurrent use of a Selection Escape and Selection Alert

A Selection Alert may be used concurrently with a Selection Escape by imposing a Selection Escape on the last bit of a Selection Alert. Figure 10-18 illustrates the beginning and end of a Selection Alert, accompanied by a Selection Escape embedded in the last of the padding bits.

Equipment using Selection Escapes and equipment using Selection Alerts to exit from Offline state are interoperable with each other, provided that the signaling used by these interfaces does not inadvertently initiate a Selection Sequence in any technology sharing the DTS connection. A technology that does not support Selection Escapes will not toggle the data line six or more times while the clock is a logic 1 during Online operation. For Selection Alerts, this means that the interoperable other technology will not generate the Selection Alert Bit Sequence.



**Figure 10-18 — Selection Alert with embedded Selection Escape**

## 10.8 ADTAPC State Machine

### 10.8.1 Need

T1 and above TAP.7s require knowledge of the TAPC state. With T2 and above TAPs, this knowledge will be developed independently of the CLTAPC. This is generally accomplished by implementing a TAPC within the EPU. This TAPC provides the function of an IEEE 1149.1 TAPC. Although many timing diagrams within this specification show the TAPC state changing on the rising edge of the TCK(C) signal, it is strongly recommended that this state machine be implemented to change state on the falling edge of the TCK(C) as this has numerous advantages to a TAP.7 Controller. The EPU does not contain IEEE 1149.1 Instruction or Data Registers (only a bypass bit when the CLTAPC is deselected). The APU controls the state changes of this TAPC with T4 and above TAP.7s.

### 10.8.2 An approach to implementing the ADTAPC

An approach to implementing a TAPC that changes state with the falling edge of the TCK(C) is shown in Figure 10-19. Other approaches should also be considered. This is generally accomplished by implementing a TAPC within the EPU. This TAPC provides the function of an IEEE 1149.1 TAPC.

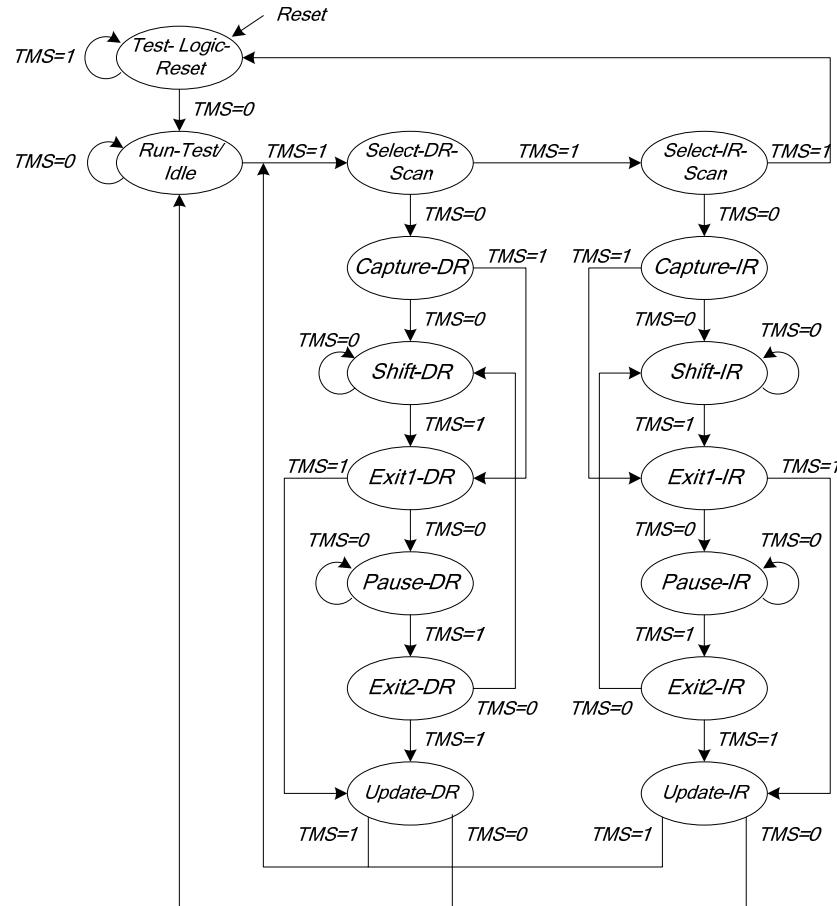
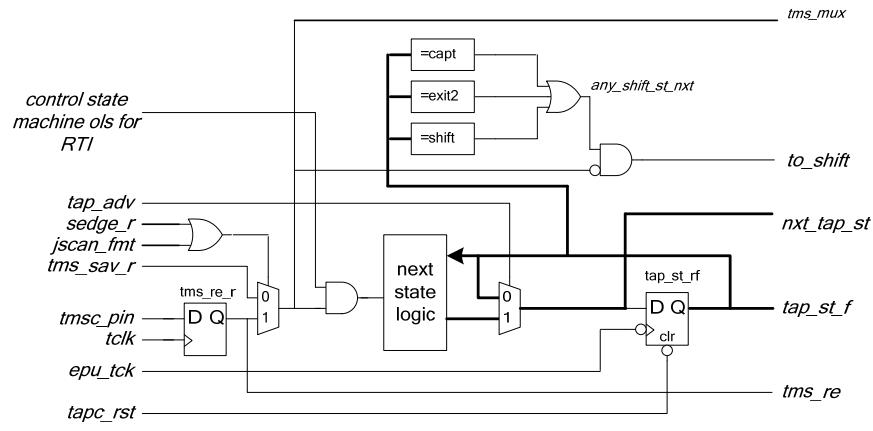


Figure 10-19 — Falling-edge ADTAPC State Machine

## 11. RSU Online/Offline capability

### 11.1 Introduction

This clause is applicable to T0–T2 TAP.7s that implement the RSU and T3 and above TAP.7s. It describes the TAP.7 architecture's Online/Offline capability and its relationship to TAP.7 and other technologies. It also provides programming considerations. With a TAP.7 technology, the Online/Offline capability is equivalent to the Selection and Deselection of the ADTAPC.

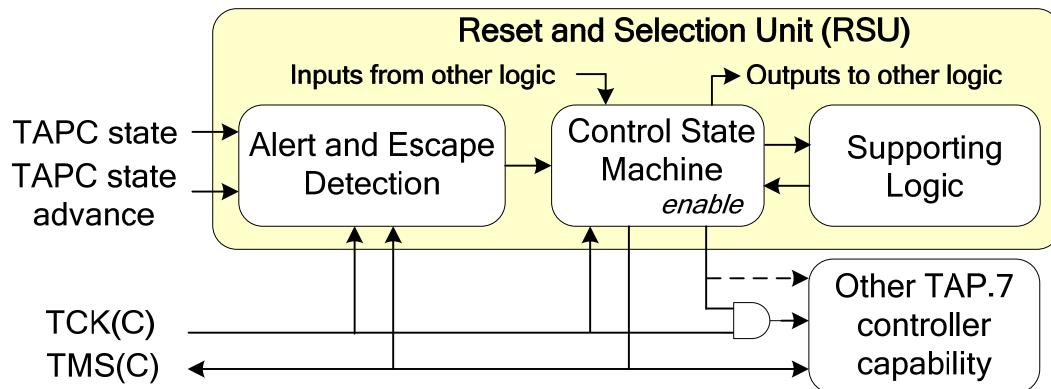
The TAP.7 Controller's full Online/Offline capability supporting boundary-scan testing may only be used in systems where the DTS sources the TCK(C) signal, as Escapes are used with this capability. A limited set of Online/Offline capability (Selection Alerts and Deselection Alerts) is available to all systems irrespective of the source of the clock (provided these optional features are implemented), as these selection/deselection mechanisms utilize only TAPC-state sequences and TMS(C) bit sequences. There is no need to implement these options with a TAP.7 Controller, in most cases, as they replicate the function of Selection and Deselection Escapes but with limitations.

The subject matter within this clause is organized in the following manner:

- 11.2 Managing Online/Offline operation
- 11.3 Online/Offline operating principles
- 11.4 Initiating Offline operation
- 11.5 Initiating Online operation
- 11.6 Context-sensitive response to Selection and Deselection Escapes
- 11.7 Selection Sequence
- 11.8 Parking-state considerations
- 11.9 Control State Machine (CSM)
- 11.10 Programming considerations

### 11.2 Managing Online/Offline operation

The RSU manages the selection/deselection of the ADTAPC. From a very high level, the RSU enables the remainder of the TAP.7 Controller functionality when the TAP.7 Controller is Online, and it disables the remainder of the TAP.7 functionality otherwise. The RSU either consumes or discards the TMS(C) information while the TAP.7 Controller is Offline. TDI(C) and TDO(C) information is discarded while the TAP.7 Controller is Offline. The relationship of the RSU and other TAP.7 capability is shown in Figure 11-1.



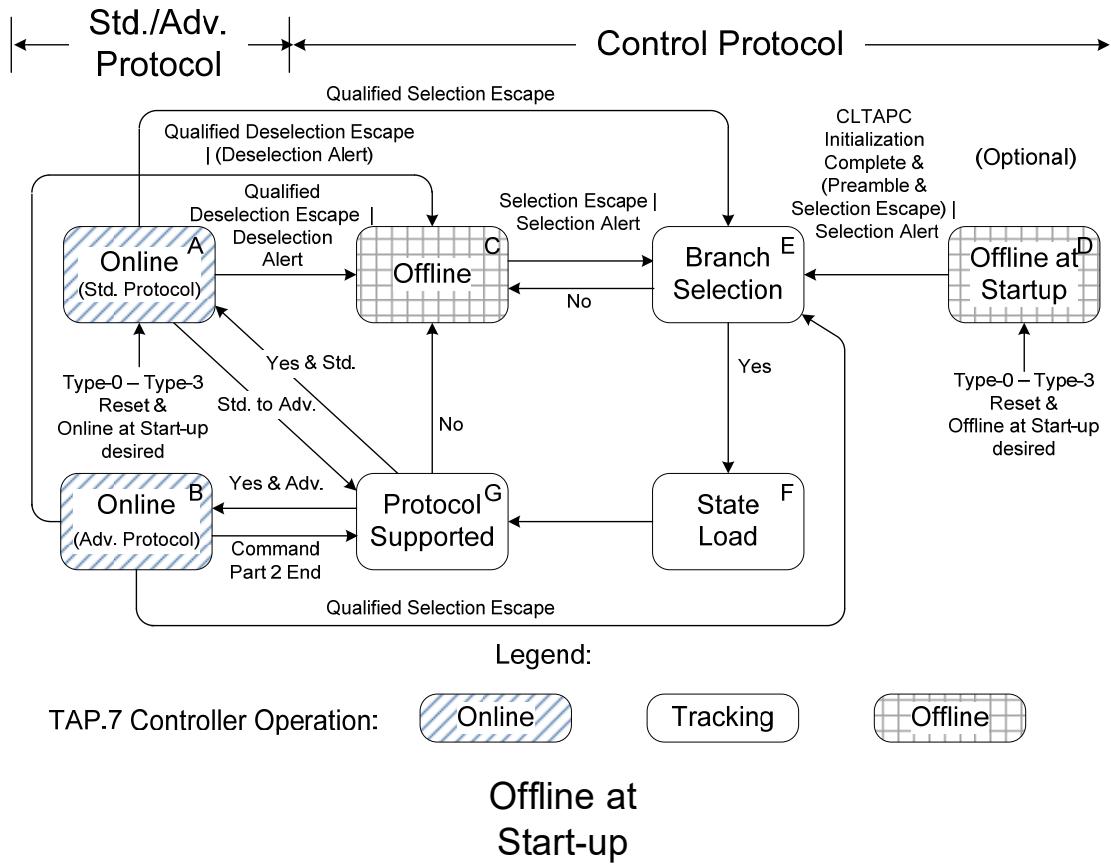
**Figure 11-1 — Conceptual view of technology selection mechanism**

The RSU function includes Escape Detection, a Control State Machine, logic supporting the operation of the Control State Machine, and optionally, Selection/Deselection Alert Detection. The RSU controls the operation of the APU and EPU by gating their clocks, enabling and disabling their operation with one or more control signals, or a executing combination of both options. With T0–T2 TAP.7s, the CSM within the RSU manages the use of the Standard and Control Protocols in addition to supervising *Online-to-Offline* and *Offline-to-Online* selection state transitions. With T3 and above TAP.7, the CSM within the RSU also manages the use of the Advanced Protocol.

### 11.3 Online/Offline operating principles

#### 11.3.1 Conceptual view of Online/Offline operation

A high-level conceptual view of Online/Offline operation with the TAP.7 Architecture is shown in Figure 11-2. The TAP.7 architecture provides for interoperability with other technologies that may be deployed using this model.



**Figure 11-2 — Online/Offline operation with a TAP.7 Controller**

The states in this figure represent an operating model managing the use of the Standard, Advanced, and Control Protocols. It can easily be adapted to other technologies. With the TAP.7 architecture, this model is implemented with a state machine called the Control State Machine. The states shown in this figure represent groups of Control State Machine states. States A–B support Online operation. The Standard Protocol used with state A, and the Advanced Protocol used with State B. States C–D support Offline operation. State C supports Offline operation created at times other than at start-up. State D supports Offline operation at start-up. Deselection/Selection Escapes and Selection/Deselection Alerts initiate transitions from states A–D. Selection events initiate a “Selection Sequence.” This sequence is formed with states E–G and determines whether a technology is placed Online or Offline.

### 11.3.2 Events affecting Online/Offline operation

Deselection events immediately place technologies Offline. Selection events initiate a Selection Sequence with this sequence determining whether a technology is placed Online. This sequence is constructed with a technology-independent part and an optional technology-dependent part. The technology-independent part specifies the selection of a technology branch. A technology-dependent part may follow. The Selection Sequence for a TAP.7 Controller is constructed with both parts.

With a TAP.7 Controller, the technology-dependent part specifies the TAP.7 Branch being selected, the ADTAPC state required for selection, and the drive management information. It may also optionally load the TAP.7 Controller Global Registers and initialize certain aspects of the TAP.7 Controller state to ensure the harmonious operation of TAP.7 Controllers being placed Online and those TAP.7 Controllers already Online. This sequence also verifies that operation specified by the Selection Sequence is supported before placing the ADTAPC Online. The ADTAPC is placed Offline when this verification fails. The state created

by power-up is either Online using the Standard Protocol (state A) or Offline-at-Start-up (waiting to be placed Online) (state D).

### 11.3.3 Summary of responses to selection/deselection events

A summary of the relationship of Offline/Online responses to Escape and Alert events are shown as follows:

- Online:
  - Responds to Deselection and Selection Escapes, provided the use of the Escape meets qualification criteria defined by the technology
- Offline:
  - Ignores a Deselection Escape
  - Responds to a Selection Escape
  - Responds to a Selection Alert, provided the detection of a Selection Alert is supported
- Offline-at-Start-up:
  - Ignores a Deselection Escape
  - Responds to a Selection Escape, provided the Escape is preceded by a preamble of at least 28 alternating logic 1s and logic 0s with no Escapes during the preamble
  - Responds to a Selection Alert, provided the detection of a Selection Alert is supported

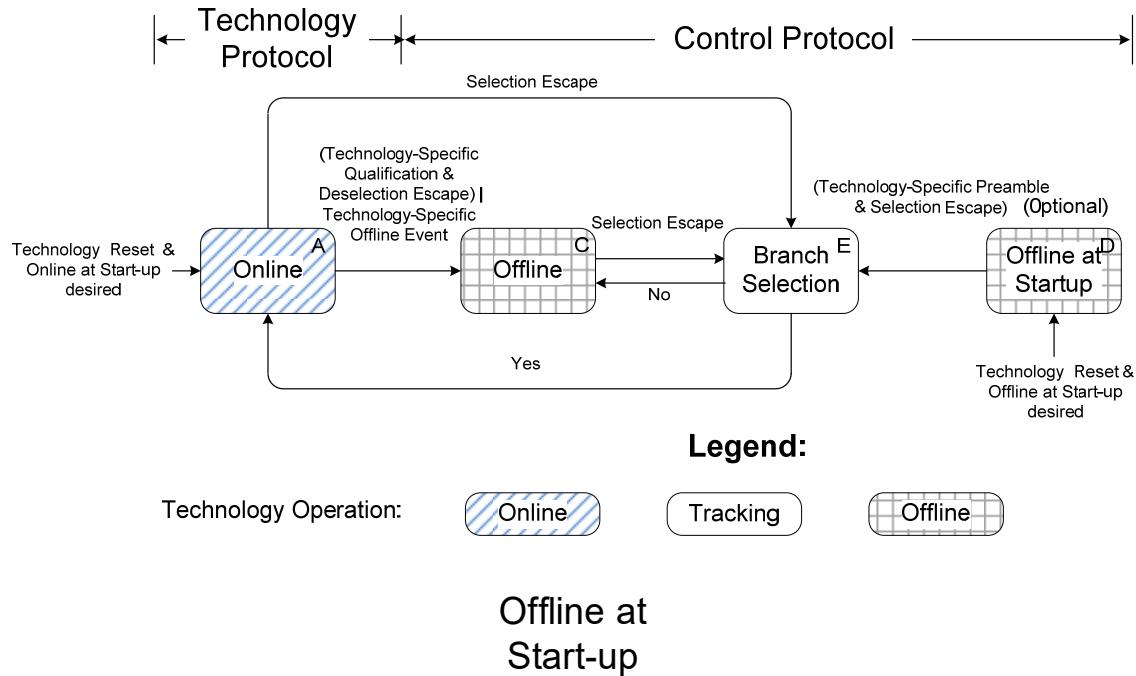
These relationships are described in the following subclauses.

### 11.3.4 Interoperability with other technologies

Technologies other than TAP.7 technology may use the following:

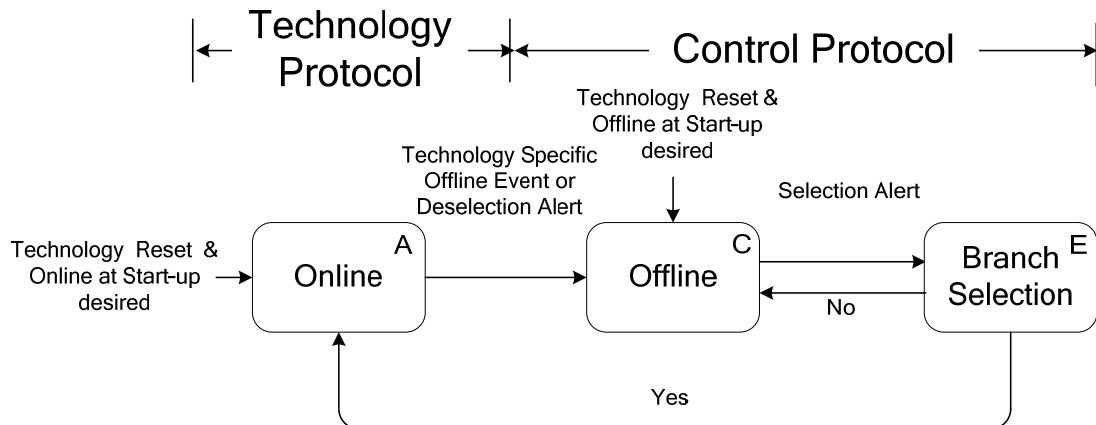
- Escapes:
  - A Deselection Escape or a technology-specific mechanism for placing a technology Offline
  - A Selection Escape for initiating a Selection Sequence
- Alerts:
  - A Deselection Alert or a technology-specific mechanism for placing a technology Offline
  - A Selection Alert for initiating a Selection Sequence, provided all technologies are Offline
- A combination of both Escapes and alerts as shown in Figure 10-18

A high-level conceptual view of Online/Offline operation with a technology other than a TAP.7 technology using an RSU supporting only Escapes is shown in Figure 11-3.



**Figure 11-3 — Online/Offline operation with another technology/RSU with Escapes**

A high-level conceptual view of the use of Selection and Deselection Alerts and/or a technology-specific mechanism is shown in Figure 11-4. The use of the RSU with Escapes is recommended for another technology that is intended to share TAP.7's TCK(C) and TMS(C) signals as it provides a means to simultaneously reset all technologies sharing connectivity of the TAP signals.



**Figure 11-4 — Online/Offline operation with another technology/RSU with Alerts**

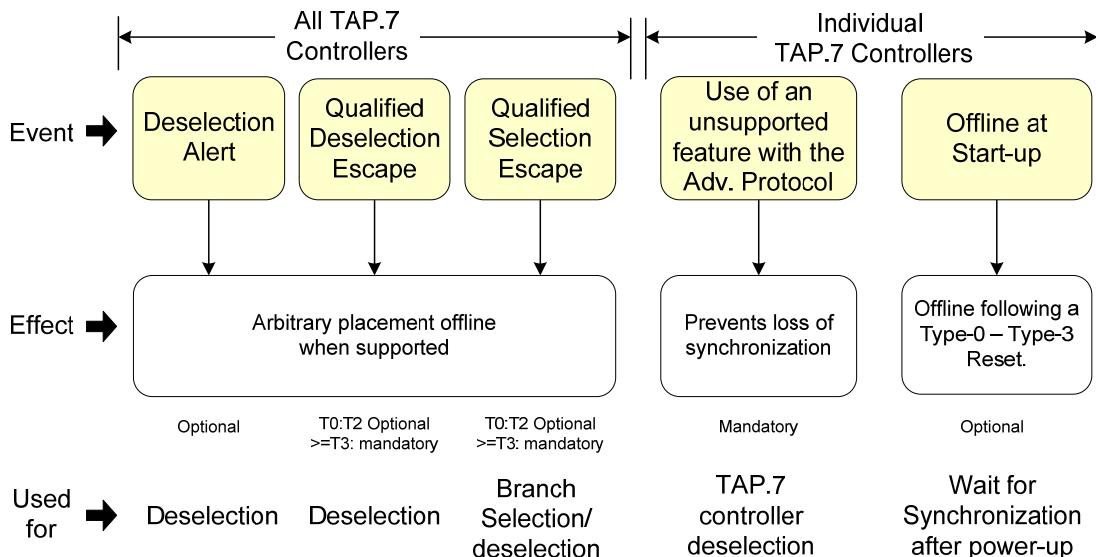
The use of both of both Selection and Deselection Escapes and Alerts concatenates the capability shown in Figure 11-3 and Figure 11-4.

## 11.4 Initiating Offline operation

### 11.4.1 Description

#### 11.4.1.1 Events initiating Offline operation

The events initiating Offline operation of a TAP.7 Controller are shown in Figure 11-5. Other technologies may also be affected in a similar manner when they implement these capabilities.



**Figure 11-5 — Offline initiation events**

### 11.4.1.2 Escapes

A Deselection Escape affects only Online technologies with an RSU (or RSU-like function). A properly qualified Deselection Escape unconditionally places an Online TAP.7 technology Offline. This TAP.7-specific qualification (see 11.6) of both Deselection/Selection Escapes requires that they occur at a specific point in relation to the Standard and Advanced Protocols. A Deselection Escape that does not meet the qualifying criteria is simply ignored.

A Selection Escape affects both Online and Offline technologies with an RSU (or RSU-like function). With a TAP.7 Controller, a Selection Escape initiates a Selection Sequence differently based on the states shown in Figure 11-2, as follows:

- Offline-at-Start-up      A Selection Escape is preceded by the appropriate preamble (see 11.6 and 11.9.7).
- Offline                  A Selection Escape initiates a Selection Sequence.
- Online                  A Selection Escape initiates a Selection Sequence, provided the Selection Escape is properly qualified (see 11.6).

It is the DTS' responsibility to ensure the use of Deselection/Selection Escapes meets the qualification criteria. Failure to do so may place an Offline technology Online in a manner where its operation is not synchronized with the operation of an already Online TAP.7 Controller.

### 11.4.1.3 Alerts

A Deselection Alert affects only Online technologies with an RSU (or RSU-like function) when this feature is supported. A Deselection Alert places an Online technology Offline and has no affect on an Offline technology.

A Selection Alert affects only Offline technologies with an RSU (or RSU-like function). It cannot be used when any technologies are Online, as its use of TMSC signal values in successive bit periods is in conflict with the normal operation of the TMSC signal while a technology is Online. It initiates a Selection Sequence with an Offline TAP.7 technology. The Selection Sequence determines whether the technology is placed Offline or Online.

### 11.4.1.4 Use of an unsupported feature

An ADTAPC places itself Offline when the use of an unsupported feature affecting the Advanced Protocol bit sequences is specified. It is placed Offline before the TAP.7 Controller can lose synchronization with the Advanced Protocol bit stream by processing bit sequences it does not comprehend. These conditions occur when any of the following are true:

- The use of the Advanced Protocol is enabled when the unsupported feature is already enabled.
- The use of the unsupported feature is enabled when the Advanced Protocol is already enabled.

These conditions are called a “Configuration Fault” (see 21.2.3).

### 11.4.1.5 Offline-at-Start-up

At start-up, a technology may be placed Offline, where it awaits being placed Online. With a TAP.7 Controller, a Type-0–Type-3 Reset places it Offline when the start-up option is Offline-at-Start-up.

## 11.4.2 Specifications

### Rules

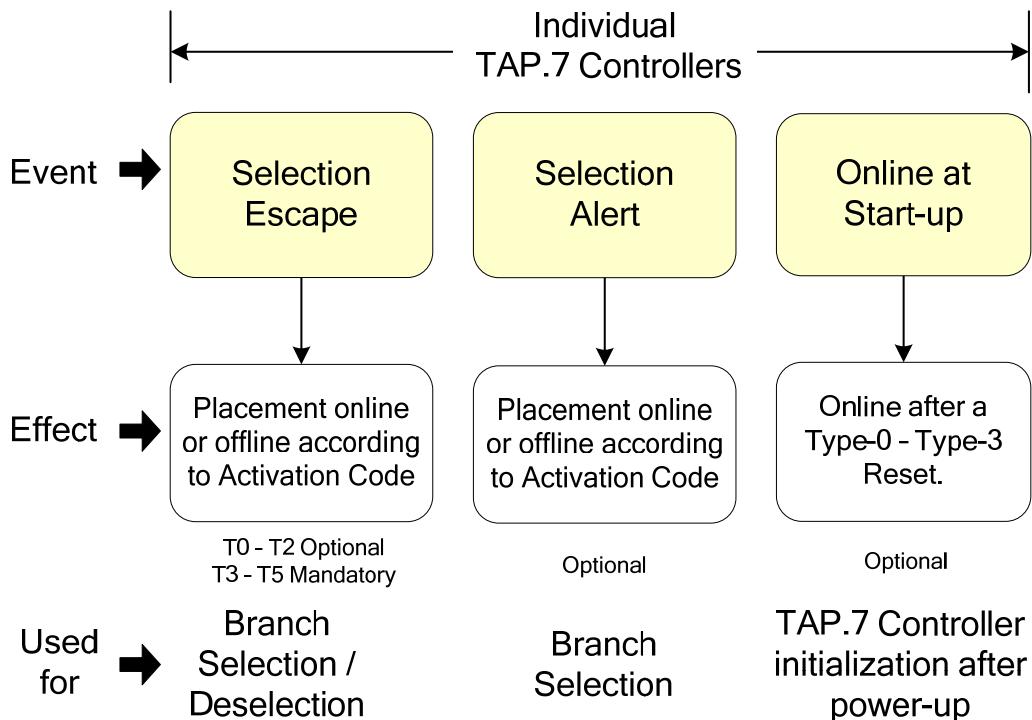
The specification rules are as follows:

- a) Each subsequent specification in 11.4.2 shall only apply to TAP.7s with an RSU.
- b) A Deselection Escape that meets the qualification criteria specified in Rule 11.6.2 b) shall place the TAP.7 Controller Offline, provided Offline/Online operation is supported (the RSU is implemented).
- c) A Deselection Alert shall place the TAP.7 Controller Offline, provided the detection of Deselection Alerts is supported.

## 11.5 Initiating Online operation

### 11.5.1 Description

The events that may initiate Online operation are shown in Figure 11-6. Other technologies may also be affected in a similar manner when they implement these capabilities.



**Figure 11-6 — Online initiation events**

Both a Selection Alert and a Selection Escape initiate a Selection Sequence as described in 11.4. The Selection Sequence determines whether the technology is placed Online or Offline. At start-up, a technology may be placed Online. With a TAP.7 Controller, a Type-0–Type-3 Reset places it Online when the start-up option is not Offline-at-Start-up.

## 11.5.2 Specifications

### Rules

The specification rules are as follows:

- Each subsequent specification in 11.5.2 shall only apply to a TAP.7 with Online and Offline operation.
- A Selection Alert shall initiate a Selection Sequence, provided any of the following are true:
  - The TAP.7 Controller is Offline-at-Start-up.
  - The TAP.7 Controller is Offline.

## 11.6 Context-sensitive response to Selection and Deselection Escapes

### 11.6.1 Description

Note the use of “Selection Escape,” “Qualified Selection Escape,” and “Qualified Deselection Escape” in the terms causing state transitions in Figure 11-2. A technology’s response to Selection and Deselection Escapes is context sensitive. It is affected by the following two factors:

- The technology’s selection state (Online or Offline)
- Criteria imposed by an Online technology

### 11.6.1.1 Escape qualification criteria during Online operation

Online technologies may impose conditions as to where Selection Escapes may be placed in relation to the bit stream of the protocol controlling the technology. In the case of a TAP.7 Controller, imposing these conditions both simplifies implementation and reduces the validation space. The use of Escapes when the criteria defined by operating technologies are *not* met will, in most cases, cause erroneous operation when another TAP.7 Controller or technology is Offline. This use is considered a programming error.

In the case of a TAP.7 technology, there are three types of restrictions on the use of Selection and Deselection Escapes, those related to the following:

- Use of the System Path
- Use of SSDs
- Position of the Deselection/Selection Escape relative to the SP content (when using the Advanced Protocol):
  - The first bit of SP content
  - When the first bit of SP content is sourced by the DTS

With a TAP.7 Controller, the use of the Standard or Advanced Protocol determines the qualification criteria. The TAP.7 Controller responds to Selection and Deselection Escapes when the criteria specified by Rule 11.6.2 b) are met.

When using the Standard Protocol, the TAP.7 Controller responds to Selection and Deselection Escapes when the following occurs:

- The System Path is being used ( $zbs\_cnt < 1$  or  $suspend == 1$ ).
- The Delayed SSD State is  $SSD\_SA\_DLY$ .
- SSD processing has not begun (the directive start bit has not activated the SSD Processing State Machine).

When using the Advanced Protocol, the TAP.7 Controller responds to Selection and Deselection Escapes when the following occurs:

- The System Path is being used.
- The Delayed SSD State is  $SSD\_SA\_DLY$ .
- SSD processing has not begun (the directive start bit has not activated the SSD Processing State Machine).
- The Escape occurs coincidentally with the first bit of an SP and the bit value is sourced by the DTS.

### 11.6.1.2 Escape qualification criteria during Offline-at-Start-up operation

A Type-0–Type-3 Reset places the TAP.7 Controller Offline when the start-up option is Offline-at-Start-up. Subsequently, it is placed Online by either a Selection Escape or a Selection Alert. The assumptions made regarding this function include the following:

- A DTS may or may not be connected when Offline-at-Start-up operation begins.
- Connecting the DTS to the TS will cause random TAP signaling.

The operation of a running system should not be disturbed by the random signaling caused by the connection of a DTS to the system. For this to be case, the erroneous initiation of a Selection Sequence by

either a Selection Escape or a Selection Alert will be avoided. This is accomplished by imposing a qualifying condition for initiating a Selection Sequence with an Selection Escape in this state. The detection of a Selection Alert is qualified by its nature.

This qualifying condition for initiating a selection with a Selection Escape will have the following attributes:

- With an Offline TAP.7 Controller, it will prevent its initiation.
- With an Online TAP.7 Controller, it will create a benign TAPC state sequence when using the Standard Protocol.

A preamble of 28 or more alternating logic 1 and logic 0 TMS(C) values meets these requirements, provided the DTS initiates the preamble at the proper time in a TAPC state progression (see Annex D). Offline-at-Start-up is covered in more detail in 11.9.7.

With an Online TAP.7 Controller, this preamble pattern produces a benign TAPC state sequence with a TAP.7 Controller using the Standard Protocol when initiated with the TAPC state progressions shown in Annex D.

### **11.6.1.3 Selection Alert during Offline-at-Start-up operation**

A Selection Alert may be used only when all technologies are Offline. It would be interpreted as data otherwise, with this causing erroneous operation. A Selection Alert is a unique 127-state bit pattern for detection. This is sufficiently long to prevent the erroneous detection of a Selection Alert due to the above mentioned DTS/TS connection and with there being an extremely low probability of it occurring during normal operation.

## **11.6.2 Specifications**

### **Rules**

The specification rules are as follows:

- a) Each subsequent specification in 11.6.2 shall only apply to a TAP.7 with Online and Offline operation.
- b) Selection and Deselection Escapes shall be recognized, provided any of the following are true:
  - 1) All the following are true:
    - i) The Standard Protocol is being used.
    - ii) The System Path is being used.
    - iii) The Delayed SSD State is *SSD\_SA\_DLY* (see Table 20-13).
    - iv) SSD processing has not begun (see Figure 20-9).

NOTE—The simultaneous use of SSDs and Selection/Deselection Escapes is considered a programming error. The sequential use of these features is required.

- 2) All the following are true:
  - i) The Advanced Protocol is being used.
  - ii) The System Path is being used.
  - iii) The Delayed SSD State is *SSD\_SA\_DLY*.
  - iv) The Escape occurs coincidentally with the first bit of an SP.

- vi) The TMSC value for the first bit of an SP is source by the DTS and shall be ignored otherwise.

## 11.7 Selection Sequence

### 11.7.1 Initiation

A summary of the conditions initiating a Selection Sequence is listed as follows, provided the DTS utilizes these events in accordance with the restrictions governing their use:

- When the TAP.7 Controller is Online with:
  - A properly qualified Selection Escape
- When the TAP.7 Controller is Offline with:
  - A Selection Escape
  - A Selection Alert
  - Both of the above occurring concurrently (as described in Figure 10-18)
- When the TAP.7 Controller is Offline-at-Start-up with:
  - A Selection Escape preceded by the proper preamble
  - A Selection Alert
  - Both of the above occurring concurrently

### 11.7.2 Format

The Selection Sequence format is shown in Figure 11-7. The Selection Sequence is constructed from the following:

- A common part used to select a technology
- A custom part imposing technology-specific selection requirements and/or conveying other information

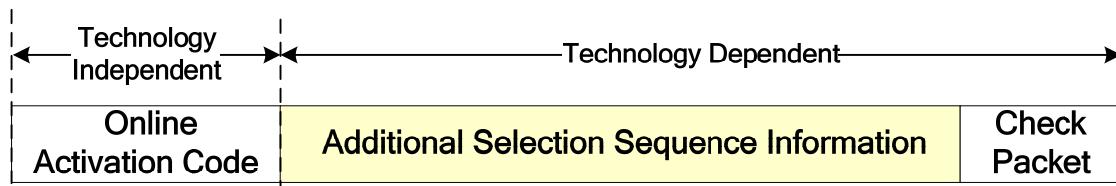


Figure 11-7 — TAP.7 Selection Sequence

### 11.7.3 Technology-independent portion

The part of the Selection Sequence that is technology independent is a four-bit Online Activation Code. It specifies the technology branch to be selected. A TAP.7 Controller is placed Offline during or following the Online Activation Code when it does not satisfy the above criteria. It becomes a candidate for Online operation otherwise.

The term “candidate for Online operation” is used to emphasize the point that the technology-dependent part may be added to impose additional selection criteria and convey other information, as is the case with Selection Sequences for TAP.7 technologies. With a TAP.7 Controller, a candidate for Online operation is placed Online when the Check Packet completing the Selection Sequence confirms that the specified features affecting the Advanced Protocol bit sequences are supported and is placed Offline otherwise.

#### 11.7.4 Technology-dependent portion

Information subsequent to the Online Activation Code is technology dependent and may not exist at all with technologies other than the TAP.7 technology. Because a technology that is not selected by the Online Activation Code is placed Offline during or immediately after this code, subsequent information affects only the technology specified by the Online Activation Code (TAP.7 or other technology). The TAP.7 technology-dependent portion of the Selection Sequence is constructed with the following Control Protocol Elements:

- Extension Code      Specifies additional selection criteria, whether Global Registers are loaded, and whether the TAP.7 Drive Policies are changed to inhibit TMSC and TDO(C) drive
- Global Register State      Values stored in the Global Registers
- Check Packet      Check that specified operation is supported, place the TAP.7 Controller either Online or Offline

The Extension Code, Global Register State, and Check Packet form the TAP.7 technology-dependent portion of the Selection Sequence shown in Figure 11-7.

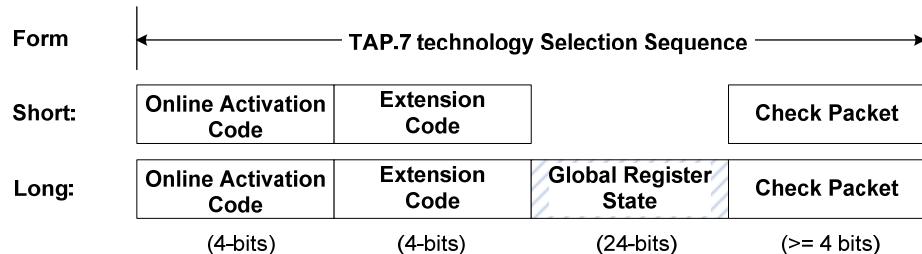
The Extension Code performs the following actions:

- Loads a predetermined scan format compatible with the type of branch(s) being selected.
- Specifies one of four combinations of TAP.7 Controller TAPC state(s) required for selection (*Test-Logic-Reset/Run-Test/Idle*, *Select-DR*, *Pause-IR*, or *Pause-DR*).
- Specifies whether the store of Global Register values occurs.
- Provides a means to prevent drive conflicts until the same drive policy is established for all Online ADTAPCs.

The Global Register State loads the Global TAP.7 Controller registers that affect the Advanced Protocol bit sequences. The Check Packet completing the Selection Sequence confirms that the specified features affecting the Advanced Protocol bit sequences are supported as stated previously.

#### 11.7.5 Forms of Selection Sequence

There are two forms of the TAP.7 Selection Sequence, short and long, as shown in Figure 11-8.



**Figure 11-8 — Conceptual view of TAP.7 Controller Selection Sequence**

This figure shows the four Control Protocol components in each form of the Selection Sequence. The Short-Form Selection Sequence excludes the Global Register State, whereas the Long-Form Selection Sequence includes the load of the Global Register State. Both provide the function related the Extension Code that is described in 11.7.4.

The Long-Form Selection Sequence may be used with either test or debug and may be used at any time. The Short-Form Selection Sequence is expected to be used with boundary-scan testing across multiple branches when creating Series-Equivalent Scans. It may also be used with debug provided its limitations are comprehended. The use of the Long- and Short-Form Selection Sequences and their limitations are described in the following section.

A Short-Form Selection Sequence presumes the values of Global Registers with the same name are the same in all TAP.7 Controllers. Its ability to place a TAP.7 Controller Online is constrained when the TAP.7 Controller was placed Offline by one of the following:

- A Check Packet
- Offline-at-Start-up operation

In these cases, it is possible that within the technology branch, the values of Global Register in all TAP.7 Controllers will diverge as TAP.7 Controllers with an Offline ADTAPC lose synchronization with the current operating characteristics of the link. This occurs because some, but not all, of the TAP.7 Controllers forming the branch may be placed Offline, with the TAP.7 Controllers remaining Online subject to having their Global Register values diverge from those expected by the DTS. This does not occur when the ADTAPC is placed Offline by a means other than the two described above, as all TAP.7 Controllers forming a technology branch are placed Offline in unison and have the same Global Register values when the technology branch is subsequently selected. This possibility is noted by the TAP.7 Controller when either of the above conditions places the ADTAPC Offline and is used to require that the Long-Form Selection Sequence be used to place the TAP.7 Controller Online.

When the ADTAPC is placed Offline by a means other than the two described above, all TAP.7 Controllers forming a technology branch are placed Offline in unison and have the same Global Register values when the technology branch is subsequently selected. In these cases, either a Long- or Short-Form Selection Sequence may be used to place a TAP.7 Controller Online.

## 11.7.6 Online Activation Code

### 11.7.6.1 Description

The value of the four-bit Online Activation Code specifies the criteria required for a technology branch to be considered as a candidate for Online operation. In cases where there is no technology-dependent part of a Selection Sequence, the Online Activation Code is sufficient to place a technology Online or Offline.

The format for the Online Activation Code is shown in Table 11-1. It specifies the following:

- A technology branch other than a TAP.7 Branch
- A specific TAP.7 Technology Branch
- All TAP.7 Technology Branches

A TAP.7 Controller becomes a candidate for placement Online when the Online Activation Code specifies one of the following:

- All TAP.7 technologies

- A specific TAP.7 technology that is for a:
  - T0–T2 TAP.7 The Series Scan Topology
  - T3 and above TAP.7 The Scan Topology defined by the TOPOL Register

The selection of all TAP.7 Technology Branches:

- May be used to select a branch of interest when there is only one TAP.7 Branch
- Is used when creating Series-Equivalent Scans involving all TAP.7 Branches, when the *Update-xR* and *Capture-xR* states are to be traversed
- Forces the use of the JScan2 Scan Format

Specifying the selection of a specific TAP.7 Technology Branch:

- May be used to select a branch of interest when there is one or more TAP.7 Branches
- Is used when creating Series-Equivalent Scans involving a single TAP.7 Branch, when the *Shift-xR* state is to be traversed
- Forces the use of the following scan formats:
  - Series Branch selection JScan2
  - Star-4 Branch selection JScan3
  - Star-2 Branch selection OScan1

The Online Activation Code is processed as follows. The TAP.7 Controller verifies that the OAC[1:0] field (the first two bits) specifies the TAP.7 Technology. The TAP.7 Controller is placed Offline when this is not the case and remains a selection candidate otherwise. When the TAP.7 Controller remains a selection candidate, it then verifies the scan topology specified by the OAC[3:2] field is one of the following:

- All or Series for a T0–T2 TAP.7
- All or matches the scan topology specified by the TOPOL Register with T3 and above TAP.7s

The TAP.7 Controller is placed Offline when this is not the case and remains a selection candidate otherwise.

When the TAP.7 Controller remains a candidate for placement Online following the Online Activation Code, the Scan Format Register is loaded with a scan format compatible with operation in the scan topology specified by the Online Activation Code as shown above. This operation is performed automatically to accelerate boundary-scan operations spanning multiple branches, as a different scan format is required to scan STL Paths with each branch, and a JScan Scan Format is required to simultaneously move the TAPC state progression of all branches through the *Update-xR* and *Capture-xR* states. This value is overwritten with the value of the scan format embedded in the Global Register state with a Long-Form Selection Sequence.

The bit period following the Online Activation Code cannot be driven by a technology within the TS as shown in Figure 10-11. This ensures there is no overlap of DTS and technology drive (a drive conflict). This bit may be used as a TS input bit.

### 11.7.6.2 Specifications

#### Rules

The specification rules are as follows:

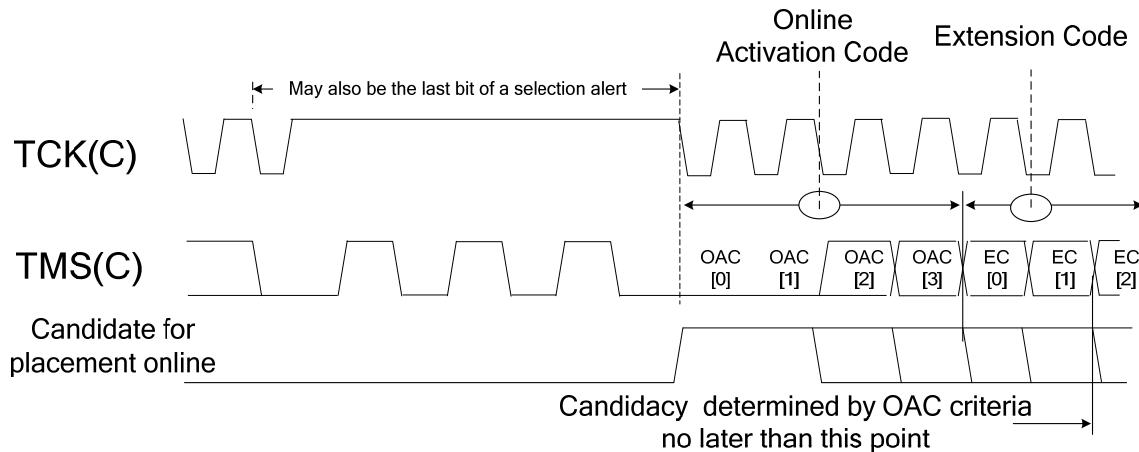
- a) Each subsequent specification in 11.7.6.2 shall only apply to a TAP.7 supporting Online and Offline operation.

- b) The format of the Online Activation Code shall be governed by Table 11-1.

**Table 11-1 — Online Activation Code**

Field	Value	OAC—Online Activation Code
03:00	yyxx	<p>Technology selection:</p> <p>xx==00:      yy==00:All of the following TAP.7 Topologies:          Series, Star-4, and Star-2      yy==01:TAP.7 Series Scan Topology,      yy==10:TAP.7 Star-4 Scan Topology,      yy==11:TAP.7 Star-2 Scan Topology,</p> <p>xx==01: reserved for TAP.7 technology      yy==00:reserved      yy==01:reserved      yy==10:reserved      yy==11:reserved</p> <p>xx==10:reserved for selecting other technologies:      yy==00:Slimbus      yy==01:USB      yy==10:ARM      yy==11:reserved</p> <p>xx==11:reserved for selecting other technologies:      yy==00:reserved      yy==01:reserved      yy==10:reserved      yy==11:reserved</p>

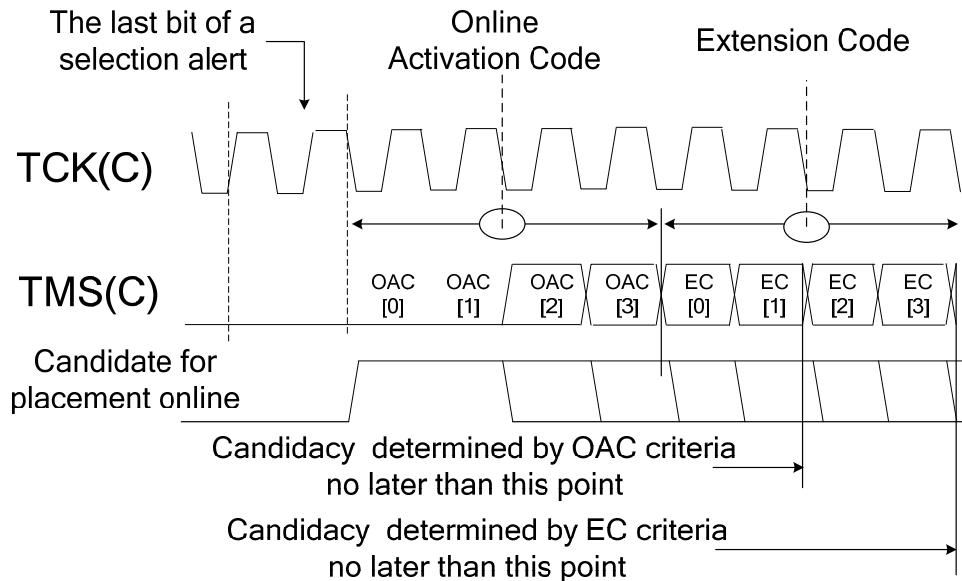
- c) The data bit following the data bit coincident with a Selection Escape shall be used as the LSB of the Online Activation Code as shown in Figure 11-9.



Note: This figure conveys timing information only and does not convey all possible combinations of data values

**Figure 11-9 — Selection Escape/subsequent data timing relationship**

- d) The data bit following the data bit coincident with the last bit of a Selection Alert shall be used as the LSB of the Online Activation Code as shown in Figure 11-10.



Note: This figure conveys timing information only and does not convey all possible combinations of data values

**Figure 11-10 — Selection Alert/subsequent data timing relationship**

- e) A TAP.7 Controller shall become a selection candidate when a Selection Sequence is initiated.
- f) When a TAP.7 Controller fails to meet the selection criteria specified by the Online Activation Code, it shall be placed Offline (its candidacy terminated) as governed by Figure 11-9 and Figure 11-10.
- g) When a TAP.7 Controller meets the selection criteria specified by the Online Activation Code, it shall remain a candidate for placement Online as governed by Figure 11-9 and Figure 11-10.

NOTE—Additional rules applicable to the Online Activation Code are defined in 11.9.5.2.

### 11.7.7 TAP.7 Extension Code

#### 11.7.7.1 Description

When the OAC specifies a TAP.7 Technology Branch, the four-bit Extension Code does the following:

- Specifies the ADTAPC TAPC state(s) required for placement Online
- Provides a means to modify the TDO(C) and TMS(C) Drive Policies (described in Clauses 13 and 14) as follows:
  - No change
  - Presume that the TDO(C) or TMS(C) Drive Policy of at least one TAP.7 Controller is not the same as other TAP.7 Controllers and/or the DTS
- Specifies whether the TAP.7 Controller Selection Sequence is Short- or Long-Form

### 11.7.7.2 Specifications

#### Rules

The specification rules are as follows:

- a) Each subsequent specification in 11.7.7.2 shall only apply to a TAP.7 supporting Online and Offline operation.
- b) When the Online Activation Code specifies a TAP.7 Controller as a selection candidate, the TAP.7 Controller shall use the data bit following the last bit (MSB) of the Online Activation Code as the LSB of the Extension Code.
- c) The format of the Extension Code shall be governed by Table 11-2.

**Table 11-2 — Extension Code**

Field	Value	Extension Code
01:00	ww	<b>STATE—Re-synchronization State</b> Re-synchronizing State - parking state required for placement Online and re-synchronization: ww==00:Run-Test/Idle or Test-Logic-Reset 01:Select-DR-Scan, 10:Pause-IR, 11:Pause-DR
02	x	<b>PROTECT</b> Specifies whether drive conflict protection is initiated: x==0:Drive conflict protection is not initiated 1:Drive conflict protection is initiated (See Rule 9.5.2 e))
03	y	<b>SHORT</b> Specifies the type of Selection Sequence: y==0:Long-Form Selection Sequence 1:Short-Form Selection Sequence

**Table 11-3 — Drive conflict protection is activated**

PROTECT == 1 ?	SHORT == 1 ?	CSM State == TESTD? (a sub-state of the TEST state)	Pass Selection Test ?	EC Last Bit ?	Drive conflict protection is activated
Yes	No	x	Yes	Yes	Yes
Yes	x	Yes	Yes	Yes	Yes
Other cases					No

- d) The bit following the last bit of the Extension Code shall be used as the LSB of one of the following:
  - 1) The Check Packet, when a SHORT bit of the Extension Code specifies a Short-Form Selection Sequence.
  - 2) Bit[00] (the LSB) of the Global Register State, when the SHORT bit of the Extension Code specifies a Long-Form Selection Sequence.
- e) When a TAP.7 Controller is a candidate to be placed Online and it fails to meet the selection criteria specified by the Extension Code, it shall be placed Offline (its candidacy terminated) as governed by Figure 11-18.
- f) When a TAP.7 Controller meets the selection criteria specified by the Online Activation Code, it shall remain a candidate for placement Online as governed by Figure 11-18.

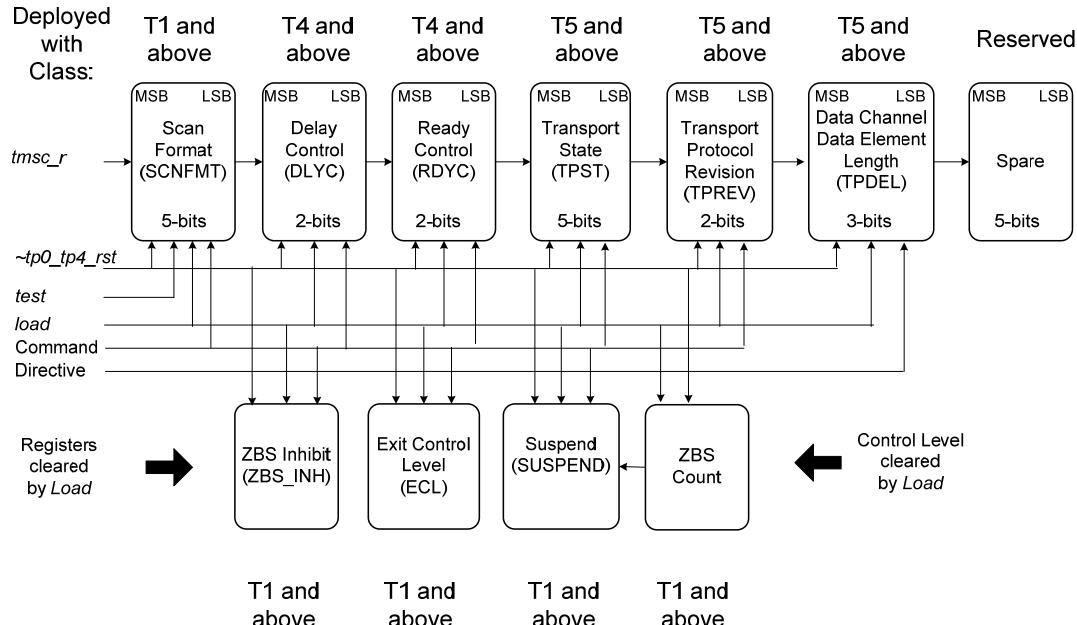
NOTE—Additional rules applicable to the Extension Code are defined in 11.9.5.2.

## 11.7.8 Global Register load

### 11.7.8.1 Description

When the SHORT Bit in the Extension Code is a logic 0, the Long Form of the TAP.7 Selection Sequence is used. A Short-Form Selection Sequence is used otherwise.

With the Long Form of the Selection Sequence, the TAP.7 Controller's Global Registers required for specifying the Standard and Advanced Protocol characteristics are configured as a shift register as shown in Figure 11-11.



Note: The assertion of the  $\sim tp0\_tp4\_rst$  signal (such as when the TAPC state is parked in the TLR state) inhibits the load of the registers above as this signal establishes the reset value of these registers as shown in Table 9-2.

**Figure 11-11 — Serial load and initialization of Global Registers**

The order of the Global Registers in this shift register is determined by the TAP.7 Class where they are first utilized. The Global Registers of a T2 TAP.7 are closest to the shift register input with the registers of incrementing classes following thereafter. Note that these register values are transmitted LSB first, so the MSB of each Global Register that is a segment of the shift register is nearest the shift register input when the Global Register Load completes. The Global Register Load affects only T2 and above TAP.7s, as T0 and T1 TAP.7s have no registers specifying the Standard and Advanced Protocol characteristics.

The DTS transmits the Global Register State information (defined in Table 11-4) with the register values supporting the T5 TAP.7 transmitted first. With this approach, all Global Register information that is associated with the TAP.7 Classes above the class implemented is merely discarded.

The ZBS Inhibit (ZBSINH), Suspend (SUSPEND), and Exit Control Level (ECL) Registers are set to a logic 0 by the *TEST* state and are not part of the shift register. Note that Rule 11.9.5.2 j) specifies that the load of the Global Register State overwrites the scan format value that is loaded as specified by Rule 11.10.5.2 f).

### **11.7.8.2 Specifications**

#### **Rules**

The specification rules are as follows:

- a) Each subsequent specification in 11.7.8.2 shall only apply to a TAP.7 supporting Online and Offline operation.
- b) The format of the Global Register State shall be governed by Table 11-4.

**Table 11-4 — Global Register State loaded**

<b>Field</b>	<b>Value</b>	<b>Global Register State</b>
04:00	yyyyy	Reserved – send as a logic 0.
07:05	yyy	TPDEL – Transport Data Element Length value
09:08	yy	TPPREV – Transport Protocol Revision value
14:10	yyyyy	TPST – Transport Data Channel State Enables value
16:15	yy	RDYC – Ready Control value
18:17	yy	DLYC – Delay Control value
23:19	yyyyy	SCNFMT – Scan Format value

- c) The lowest numbered bit of each field of the Global Register State shall be transmitted first and used as the LSB of the field.

NOTE—Additional rules applicable to the load of Global Registers are defined in 11.9.5.2.

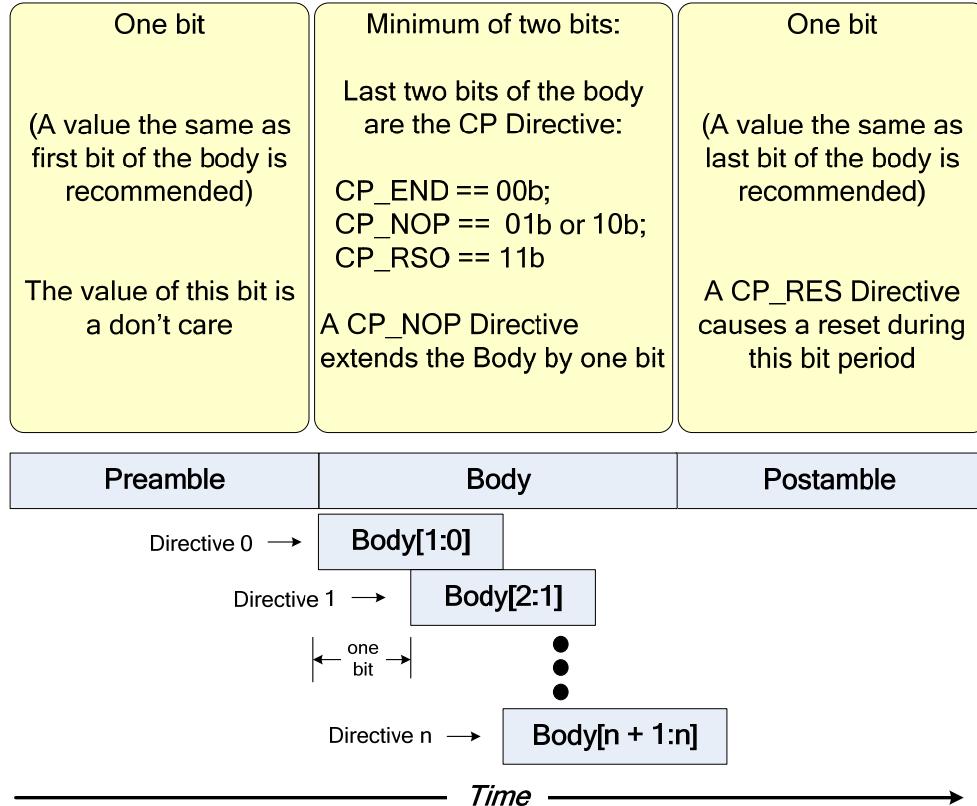
### **11.7.9 Check Packet**

#### **11.7.9.1 Description**

##### **11.7.9.1.1 Format**

The CP format is shown in Figure 11-12. The CP is constructed from the following three elements:

- Preamble Element (one bit)
- Body (two or more bits)
- Postamble (one bit)



**Figure 11-12 — CP format**

The CP Preamble Element provides a period of time where the transition from the use of the Standard Protocol to the Advanced Protocol may occur. This is helpful to some DTS implementations. The DTS may use this time period to change the TMSC pin-drive source or drive characteristics as the TAP.7 Controller ignores the value of the Preamble Element.

The CP Body Element delivers directives to the TAP.7 Controller. The TAP.7 Controller detects CP Directives by registering the TMSC input for TCKC periods  $k$  and  $k + 1$ . Directive processing begins once two bits of the Body Element are registered. The two most recent bits of the Body Element form a CP Directive as shown in Figure 11-12. The most recent TMSC pin value is the MSB of this directive.

### 11.7.9.1.2 Function

A Check Packet is used to determine whether the attributes of the TAP.7 protocol required for Online operation are supported by the TAP.7 Controller when this state ends with the CP\_END Directive. It may also create a TAP.7 Controller reset or be extended indefinitely. Its use is invoked when operating with the following:

- Standard Protocol and a value requiring the use of the Advanced Protocol is stored in the Scan Format Register.

- Advanced Protocol and Command Part Two completes.
- Control Protocol and both parts of the selection test within the Selection Sequence pass.

In each of these cases, the CP verifies that the TAP.7 Controller supports the capability required to operate Online without loss of synchronization. This decision is made when the last bit of the CP completes. The Check Packet function is implemented with the *CHK* state described in 11.9.6.

### 11.7.9.1.3 Directives

Directives (two-bit patterns within the CP) determine when and how the Check Packet terminates. The three CP Directives and their actions are listed as follows:

- CP\_NOP (01 or 10):
  - Extend the CP Body by one bit
- CP\_END (00)—one of the following:
  - Place the TAP.7 Controller Offline
  - End the CP + subsequent use of the Standard Protocol
  - End the CP + subsequent use of the Advanced Protocol
- CP\_RSO (11):
  - Initiate a Type-3 TAP.7 Controller reset

The CP\_END and CP\_RSO Directives terminate the CP after one additional bit follows these directives.

### 11.7.9.2 Specifications

#### Rules

The specification rules are as follows:

- a) Each subsequent specification in 11.7.9.2 shall only apply to a TAP.7 supporting Online and Offline operation.
- b) A candidate for placement Online shall process a Check Packet following either of the following:
  - 1) The last bit of the Extension Code, provided a Short-Form Selection Sequence is specified.
  - 2) The last bit of the Global Register State, provided a Long-Form Selection Sequence is specified.
- c) A candidate for placement Online shall be placed Offline following a CP\_END Directive of a Check Packet, provided a Configuration Fault defined by the following Rules is detected, and it will be placed Online otherwise:
  - 1) Rules 23.8.2 b) through 23.8.2 e).
  - 2) Rules 27.7.2 b).

NOTE—Additional rules defining the operation of Check Packets are defined in 11.9.6.2.

## 11.8 Parking-state considerations

### 11.8.1 Description

A TAP.7 Controller may be placed Online only when its TAPC state is one of the valid parking states or the pair of valid parking states listed as follows:

- *Pause-DR*
- *Pause-IR*
- *Select-DR-Scan*
- *Run-Test/Idle* or *Test-Logic-Reset*

The STATE field of the Extension Code specifies the ADTAPC state or pair of states that is required for placement of the TAP.7 Controller Online. The parking states listed above provide for the resynchronization of the TAP.7 Controller and the DTS TAPC states without having to reset the TAP.7 Controller.

If the TAP.7 Controller is placed Offline with its ADTAPC state parked in a state other than one of the parking states or pair of parking states listed above, then the following will occur:

- A TAP.7 Selection Sequence cannot place the TAP.7 Controller Online
- A Type-0-Type-3 Reset will be used to:
  - Place the TAP.7 Controller Online when the Offline-at- Start-up option is not specified
  - Creates the *Run-Test/Idle* state when the Offline-at-Start-up option is specified

The creation of a valid parking state while using the Standard Protocol requires qualified Selection and Deselection Escapes to be coincident with a TMS value associated with TAPC state machine, where the combination of these produces a TAPC state that is a valid parking state. The TAPC state and TMS combinations producing a valid parking state while using the Standard Protocol are shown in Table 11-5.

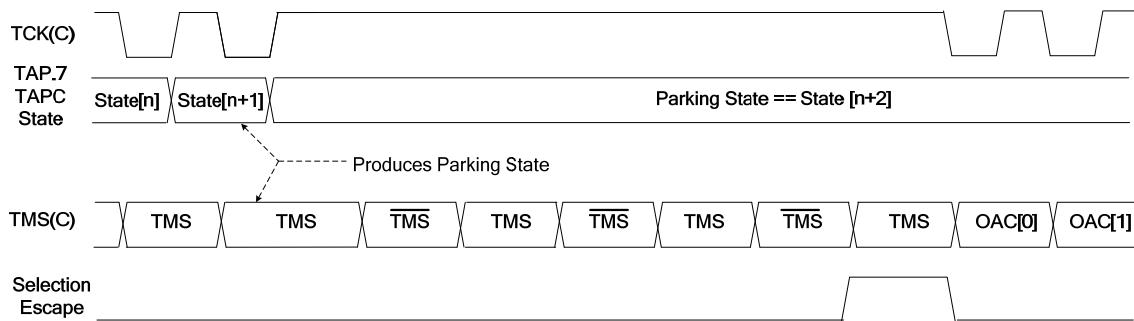
The creation of a valid parking state while using the Advanced Protocol requires qualified Selection and Deselection Escapes to be coincident with the first bit of an SP value associated with a valid parking state. The state and TMS combinations producing a valid parking state while using the Advanced Protocol are shown in Table 11-6.

### 11.8.2 Specifications

#### Rules

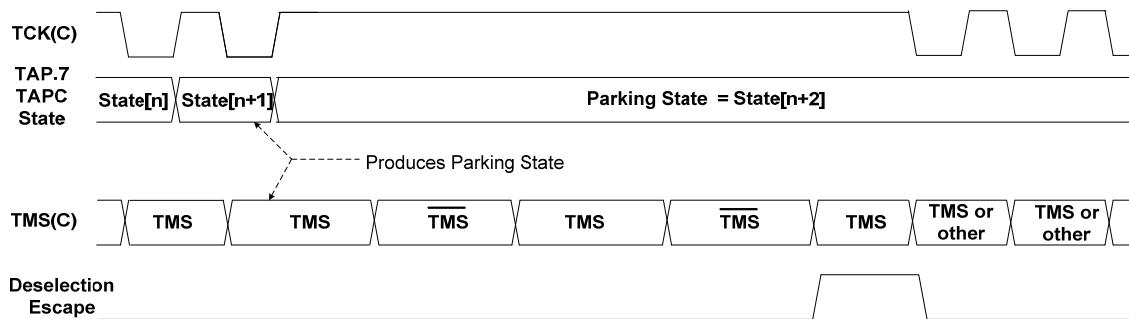
The specification rules are as follows:

- a) Each subsequent specification in 11.8.2 shall only apply to TAP.7 with Online and Offline operation.
- b) While using the Standard Protocol, the creation of a TAP.7 Controller TAPC parking state with a Selection Escape shall be governed by Figure 11-13.



**Figure 11-13 — TAPC parking-state creation with a deselection Escape/Standard Protocol**

- c) While using the Standard Protocol, the creation of an ADTAPC parking state with a Deselection Escape shall be governed by Figure 11-14.



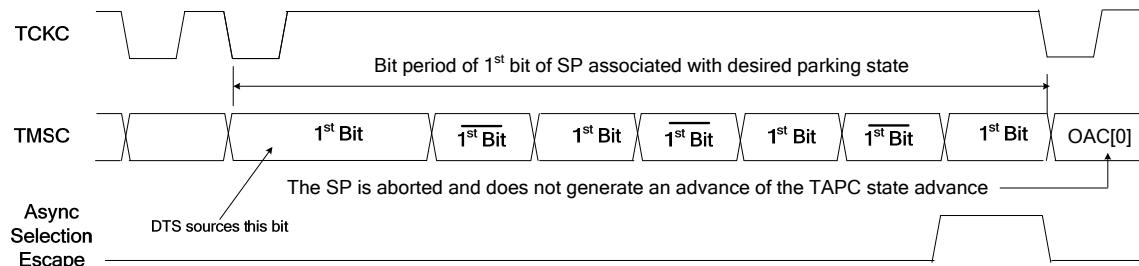
**Figure 11-14 — TAPC parking-state creation with a Selection Escape/Standard Protocol**

- d) While using the Standard Protocol, the creation of an ADTAPC parking state shall be governed by Table 11-5.

**Table 11-5 — Creation of a TAP.7 Controller TAPC parking state with the Standard Protocol**

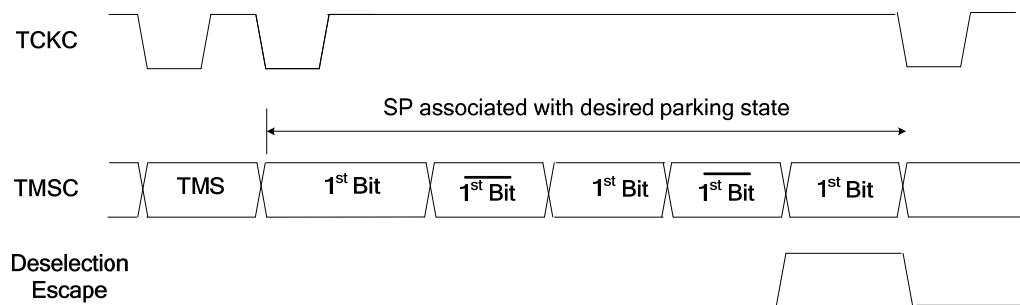
Selection and Deselection Escapes concurrent with:		Resulting parking state when placed Offline	Subsequent placement Online possible without a Type-0-Type-3 Reset
State machine state	TMS		
Update-DR, Update-IR, Run-Test/Idle, Test-Logic-Reset	0	Run-Test/Idle	Yes
Update-DR, Update-IR	1	Select-DR-Scan	Yes
Exit1-IR, Pause-IR	0	Pause-IR	Yes
Exit1-DR, Pause-DR	0	Pause-DR	Yes
Select-IR-Scan, Test-Logic-Reset	1	Test-Logic-Reset	Yes
Other combinations		None of the above	No

- e) While using the Advanced Protocol, a Selection Escape shall be considered a qualified Selection Escape provided it occurs in the first bit of a Scan Packet where the DTS sources the bit value (see Figure 11-15).



**Figure 11-15 — TAPC parking state/Selection Escape/the Advanced Protocol/relationship**

- f) While using the Advanced Protocol, the creation of an ADTAPC parking state with a qualified Deselection Escape shall be governed by Figure 11-16 and Table 11-16.



**Figure 11-16 — TAPC parking state/Deselection Escape/the Advanced Protocol/relationship**

**Table 11-6 — Creation of a TAP.7 Controller TAPC parking state with the Advanced Protocol**

Deselection-Escape concurrent with 1 <sup>st</sup> bit of SP associated with:	Resulting parking state	Subsequent selection without reset is possible
<i>Test-Logic-Reset</i>	<i>Test-Logic-Reset</i>	Yes
<i>Run-Test/Idle</i>	<i>Run-Test/Idle</i> ,	Yes
<i>Select-DR-Scan</i>	<i>Select-DR-Scan</i>	Yes
<i>Pause-IR</i>	<i>Pause-IR</i>	Yes
<i>Pause-DR</i>	<i>Pause-DR</i>	Yes
Other combinations	None of the above	No

## 11.9 Control State Machine

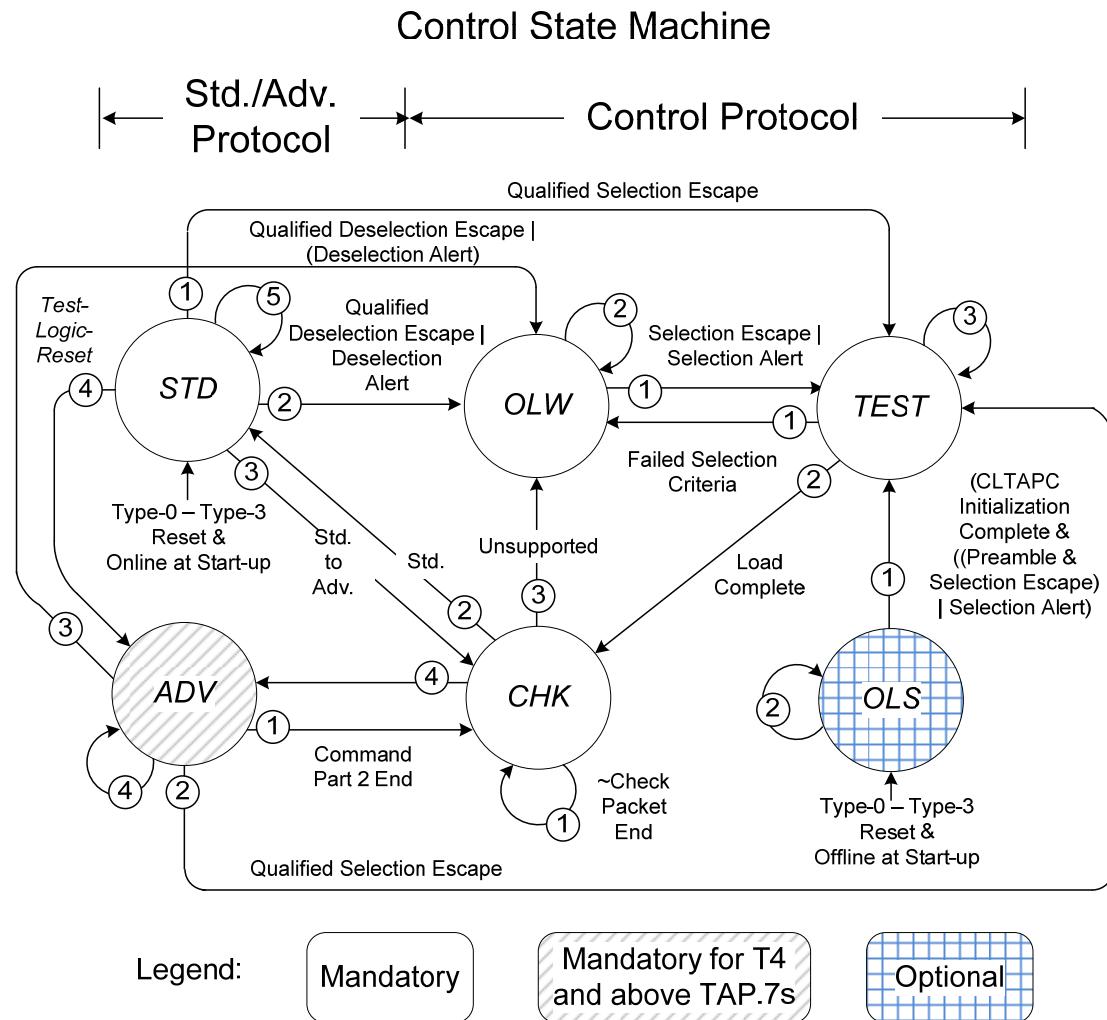
### 11.9.1 Mandatory and optional behaviors

#### 11.9.1.1 Description

The TAP.7 Controller implements the behavior shown in Figure 11-2 with a function called the CSM. A conceptual model of this state machine is shown in Figure 11-17. This model is a simplified view of this state machine function. The *OLW*, *TEST*, and *CHK* states have substates. In other figures representing this function, the following representations of CSM states may be used:

- The *STD* and *ADV* states may be represented by a single state called Online (*ONL*).
- The *OLW* state is shown as its sub-states, *OLWU* and *OLWD*.
- The *TEST* is shown as its sub-states, *TESTU* and *TESTD*.
- The *CHK* is shown as its sub-states, *PRE*, *BDY0*, *BDY1*, and *DIRP*.

The simplification provided by hiding substates aids in the understanding of the Control State Machine function. The difference in the *OLWU/TESTU* and the *OLWD/TESTD* state is the criteria required to place a TAP.7 Controller Online described in 11.7.5. A CSM state diagram with all CSM substates exposed is shown in Figure 11-28.



NOTE: The **TEST** state handles both Branch Selection and State Load behaviors.  
The **ADV** state represents the combined operation of the SPA and TPA states.  
The **CHK** state has four sub-states, **PRE**, **BDY0**, **BDY1**, and **DIRP**  
The **OLW** state has two substates, **OLWU** and **OLWD**.  
The **TEST** state has two substates, **TESTU** and **TESTD**.

**Figure 11-17 — Conceptual view of the Control State Machine**

#### 11.9.1.2 Specifications

##### Rules

The specification rules are as follows:

- Each subsequent specification in 11.9.1.2 shall only apply to a TAP.7 with an RSU.
- A TAP.7 Controller shall implement the behavior of the Control State Machine states that are shown as mandatory in Figure 11-17.

## Permissions

The specification permissions are as follows:

- c) The TAP.7 Controller may implement the behavior of the *OLS* state shown in Figure 11-17.

### 11.9.2 Standard state (*STD*)

#### 11.9.2.1 Description

With the *STD* state, the priority of conditions causing state changes is listed as follows:

- Type-0–Type-3 Resets.
- Qualified Selection and Deselection Escapes.
- Deselection Alert.
- *Test-Logic-Reset* state.
- A Check Packet is needed.

The following conditions cannot occur at the same time:

- Qualified Selection Escape.
- Qualified Deselection Escape.
- Deselection Alert.
- A Check Packet is needed.

The behavior of the *STD* state is governed by Table 11-7.

#### 11.9.2.2 Specifications

##### Rules

The specification rules are as follows:

- a) Each subsequent specification in 11.9.2.2 shall only apply to a TAP.7 supporting its selection and deselection.
- b) The CSM state changes while in the CSM *STD* state shall be governed by Table 11-7.

**Table 11-7 — TAP.7 Controller behavior in the CSM STD state**

Type-0-Type-3 Reset?	Offline-at-Start-up?	Qualified Selection Escape?	Qualified Deselection Escape?	Deselection Alert	<i>Test-Logic-Reset_f?</i>	Check Packet needed?	Next state	Next behavior
Yes	No	x	x	x	x	x	STD	Operation with the Standard Protocol
Yes	Yes	x	x	x	x	x	OLS	Offline-at-Start-up
No	x	Yes	x	x	x	x	TEST	Initiate a Selection Test
No	x	x	Yes	x	x	x	OLWD	Offline
No	x	No	x	Yes	x	x	OLWD	Offline
No	x	x	x	x	x	Yes	CHK	Begin Check Packet
No	x	x	x	No	x	No	STD	Operation with the Standard Protocol

### 11.9.3 Advanced state (ADV)

#### 11.9.3.1 Description

With the *ADV* state, the priority of conditions causing state changes is listed as follows:

- Type-0-Type-3 Resets
- Qualified Selection and Deselection Escapes
- Deselection Alert
- *Test-Logic-Reset* state
- A Check Packet is needed

The following conditions cannot occur at the same time:

- Qualified Selection Escape
- Qualified Deselection Escape
- Deselection Alert
- A Check Packet is needed

The behavior of the *ADV* state is governed by Table 11-8.

#### 11.9.3.2 Specifications

##### Rules

- a) Each subsequent specification in 11.9.3.2 shall only apply to a TAP.7 with Online and Offline operation.

- b) The CSM state changes while in the *ADV* state shall be governed by Table 11-8.

**Table 11-8 — TAP.7 Controller behavior while in the CSM *ADV* state**

Type-0-Type-3 Reset?	Offline-at-Start-up?	Qualified Selection Escape?	Qualified Deselection Escape?	Deselection Alert	Test-Logic-Reset_f?	Check Packet needed & all state machines idle?	Next state	Next behavior
Yes	No	x	x	x	x	x	STD	Operation with the Standard Protocol
Yes	Yes	x	x	x	x	x	OLS	Offline-at-Start-up
No	x	Yes	x	x	x	x	TEST	Initiate a Selection Test
No	x	x	Yes	x	x	x	OLWD	Offline
No	x	No	x	Yes	x	x	OLWD	Offline
No	x	No	No	x	Yes	x	STD	Operation with the Standard Protocol
No	x	x	x	No	No	Yes	CHK	Begin Check Packet
No	x	x	x	No	No	No	ADV	Operation with the Advanced Protocol

#### 11.9.4 Offline waiting state (*OLW*)

##### 11.9.4.1 Description

With the *OLW* state, the priority of conditions causing state changes is as follows:

- Type-0-Type-3 Resets
- Qualified Selection Escape or Selection Alert

The behavior of the *OLW* state is governed by Table 11-9.

##### 11.9.4.2 Specifications

###### Rules

The specification rules are as follows:

- a) Each subsequent specification in 11.9.4.2 shall only apply to a TAP.7 with Online and Offline operation.
- b) The CSM state changes while in the CSM *OLW* state shall be governed by Table 11-9.

**Table 11-9 — TAP.7 Controller behavior in the CSM OLW state**

Type-0-Type-3 Reset?	Offline-at-Start-up?	Selection Escape or Selection Alert?	OLWD Sub-state?	Next state	Next behavior
Yes	No	x	x	STD	Operation with the Std. Protocol
Yes	Yes	x	x	OLS	Offline-at-Start-up
No	x	Yes	No	TESTU	Initiate a Selection Test
No	x	Yes	Yes	TESTD	Initiate a Selection Test
No	x	No	x	OLW	Offline

## 11.9.5 Test state (*TEST*)

### 11.9.5.1 Description

A conceptual view of the function that is provided by the *TEST* state is shown in Figure 11-18. Activity associated with this state is as follows:

- Ensures the TAP.7 Controller meets the criteria specified by the Online Activation Code and the Extension Code for being placed Online
- Defines the TMSC/TDO(C) Drive Policies to be used when Online operation resumes
- Loads the Global Registers when specified by the Extension Code

#### 11.9.5.1.1 Selection test

A two-part selection test is performed in this state. These parts determine whether the following will occur::

- The TAP.7 Controller's technology is being selected.
- The TAP.7 Controller's TAP state is compatible with resynchronization to the state specified by the DTS.

Failing either of these tests places the TAP.7 Controller Offline. The SCNFMT Register is loaded with a value based on the Online Activation Code when both parts of the selection test are passed and remains unchanged otherwise. The Online Activation Code to Scan Format mapping is shown as follows and in Table 11-11:

- TAP.7 All JScan2
- TAP.7 Series JScan2
- TAP.7 Star-4 JScan3
- TAP.7 Star-2 OScan1

### 11.9.5.1.2 Factors requiring a Global Register load for placement Online

When the TAP.7 Controller has been placed Offline-at-Start-up or placed Offline by a Check Packet, it likely that its state does not match the state of other Online TAP.7 Controllers in its technology branch. These differences include the state of the following:

- Global Registers
- Logic determining the number of Scan Group Members

These differences in state do not arise when creating Series-Equivalent Scans using the Short-Form Selection Sequence, as all TAP.7 Controllers associated with a TAP.7 Branch are simultaneously selected and deselected.

The manner in which a TAP.7 Controller is placed Offline is recorded as substates of the *OLW* and *TEST* states. These substates identify when a Global Register Load is not synchronized to the operation of its peers. The substates of the OLW and TEST states record information with the following states:

- *OLWU* and *TESTU* states when placed Offline by a Check Packet or Offline-at-Start-up
- *OLWD* and *TESTD* states when placed Offline by a Deselection or Selection Escape while previously Online

Note that the conditions that may place a subset of the ADTAPCs sharing a branch Offline create the *OLWU* and *TESTU* states while the conditions that place all ADTAPCs sharing a branch Offline create the *OLWD* and *TESTD* states. This state information is maintained until a TAP.7 Controller is either reset or placed Online, or until the *OLWU* state is created by the *TESTD* state when a Check Packet determines the use of an unsupported feature is requested. With this the case, the manner the TAP.7 Controller is placed Offline is available for use with Selection Sequences until a TAP.7 Controller reset or ADTAPC placement Online occurs.

### 11.9.5.1.3 ADTAPC resynchronization

Because the updating of the Global Register state and drive policies ceases when a TAP.7 Controller's ADTAPC is placed Offline and the operation of its peers continues, these aspects of the TAP.7 Controller's state may not be current if it were to be placed Online without resynchronizing the state of the following across all TAP.7 Controllers, as these values may be erroneous:

- Global Registers
- Drive policies

An ADTAPC that has lost synchronization with its peers in the Branch is resynchronized with the operation of its peers using a Long-Form Selection Sequence provided both the PROTECT bit in the Extension Code is a logic 1 and the TAP.7 Controller has passed the selection criteria specified by the OAC and EC (see Table 11-3). This and a TAP.7 Controller reset are the only means to reyncronization a TAP.7 Controller with its peers. This provides a means for debug to move between technology branches without having to deal with TAP.7 Controllers that have lost synchronization with their peers. The DTS determines when both the re-synchronization of TAP.7 Controllers that have lost synchronization occurs and the placement of TAP.7 Controllers that are Offline-at-Start-up are permitted.

The resynchronization process causes all TAP.7 Controllers passing the selection criteria specified by the OAC and EC to harmonize their operating states by both performing a load of the Global Registers and modifying the criteria used to determine their TDO(C) and TMSC Drive Policies (see Clauses 13 and 14) to avoid drive conflicts.

With the resynchronization process, all Online TAP.7 Controllers do the following:

- Process the TMSC bit stream in the same manner
- Use a drive policy that prevents drive conflicts and prevents receiving data from any of the TAP.7 Controllers sharing the Branch

Once Drive Policy Protection is activated, the TAP.7 Controller will not be able to drive the TDO(C) signal or transfer data with the TMSC signal within a Star-4 and Star-2 Scan Topology with the following:

- System Path until its STL becomes the only Scan Group Member
- Control Path until its EPU becomes the only Conditional Group Member

#### **11.9.5.1.4 Priority of conditions causing state changes**

With the *TEST* state, the priority of conditions causing state changes is listed as follows:

- Type-0–Type-3 Resets (highest)
- Selection Test Fail
- SHORT bit == logic 1
- End of the State Load (lowest)

The behavior of this state is governed by Table 11-10.

#### **11.9.5.1.5 Test state function**

The function provided by the *TEST* state is shown in Figure 11-18. This figure consolidates the operation of the two *TEST* substates, *TESTU* and *TESTD*. These states have the following significance:

- *TESTU*      Global State Load is required to be placed Online
- *TESTD*      Global State Load is not required to be placed Online

There is no other difference in these states other than the effects on drive policy described in 11.7.5.

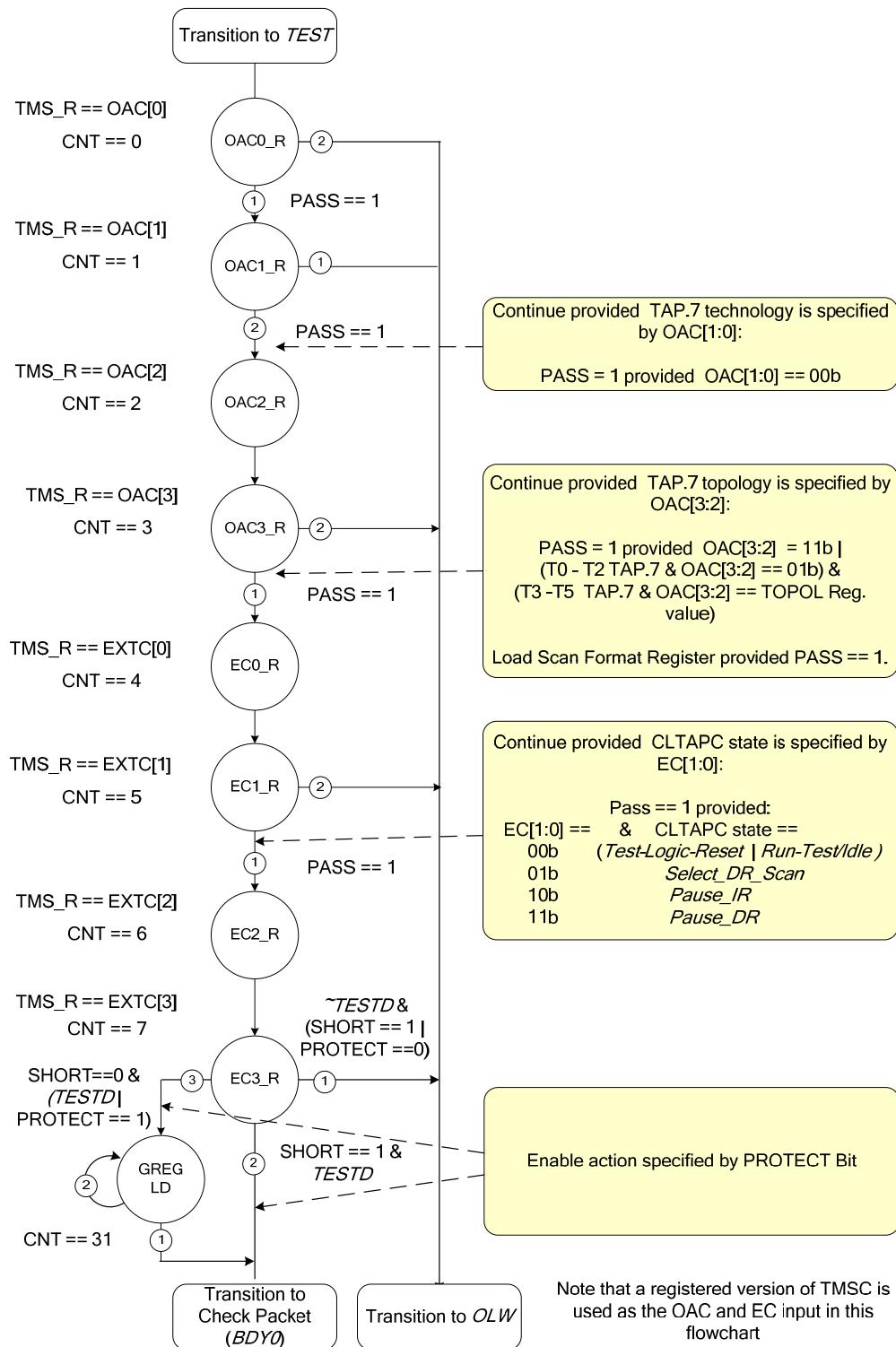


Figure 11-18 — CSM TEST state function

#### 11.9.5.1.6 Selection Sequences requiring a state load

When a Short-Form Selection Sequence is used, the transition to the *CHK* state is allowed, provided neither of the following state progressions caused *TEST* state entry:

- *OLS* >> *TESTU*
- *DIRP* >> *OLWU* >> *TESTU*

This pair of state transitions indicates there is a possibility that the states of TAP.7 Controllers sharing the DTS connection are not synchronized. When these state progressions occur and the SHORT bit is a logic 0 (a Short-Form Selection Sequence is required), a *TESTU* to *OLWU* state transition is mandated. This ensures there is a harmonious TAP.7 Controller state when a TAP.7 Controller that is Offline is placed Online.

A Long-Form Selection Sequence will therefore be used to place a TAP.7 Controller Online when:

- The start-up option is Offline-at-Start-up
- The TAP.7 Controller was placed Offline by a CP that detects an unsupported state

The detection of these conditions can be viewed as creating two versions of the *OLW* and *TEST* states, the first (*OLWU* and *TESTU*) used when the above conditions precede the entry into these states and the second (*OLWD* and *TESTD*) otherwise.

### 11.9.5.2 Specifications

#### Rules

The specification rules are as follows:

- a) Each subsequent specification in 11.9.5.2 shall only apply to a TAP.7 with Online and Offline operation.
- b) A TAP.7 Controller performing a selection test shall place the TAP.7 Controller Offline when any of the following are true:
  - 1) The TAPC state is not one of the following:
    - i) *Test-Logic-Reset*.
    - ii) *Run-Test/Idle*.
    - iii) *Select-DR-Scan*.
    - iv) *Pause-IR*.
    - v) *Pause-DR*.
  - 2) The Online Activation Code specifies any of the following:
    - i) A reserved TAP.7 technology.
    - ii) A technology other than TAP.7 technology.
  - 3) The Online Activation Code specifies the TAP.7 Star-2 Scan Topology and any of the following are true:
    - i) The TAP.7 Class is T0–T2.
    - ii) The TOPOL Register value indicates deployment in a Star-4 Scan Topology.
    - iii) The TOPOL Register value indicates deployment in a Series Scan Topology.
  - 4) The Online Activation Code specifies the TAP.7 Star-4 Scan Topology and any of the following are true:
    - i) The TAP.7 Class is T0–T2.

- ii) The TOPOL Register value indicates deployment in a Star-2 Scan Topology.
- iii) The TOPOL Register value indicates deployment in a Series Scan Topology.
- 5) The Online Activation Code value specifies the TAP.7 Series Scan Topology and any of the following are true:
  - i) The TOPOL Register value indicates deployment in a Star-4 Scan Topology.
  - ii) The TOPOL Register value indicates deployment in a Star-2 Scan Topology.
- 6) The STATE value within the Extension Code specifies *Run-Test/Idle* or *Test-Logic-Reset* and the TAPC State Machine state is not one of the following states:
  - i) *Run-Test/Idle*.
  - ii) *Test-Logic-Reset*.
- 7) All of the following are true:
  - i) The Extension Code specifies *Select-DR-Scan*, *Pause-DR*, or *Pause-IR*.
  - ii) The ADTAPC state does not match the specified state.
- c) The CSM state changes while in the CSM *TEST* state shall be governed by Table 11-10.
- d) The TESTU sub-state term used in Table 11-10 shall be true if any of the following state progressions occur to reach the *TEST* state and shall be false otherwise:
  - 1) *DIRP >> OLW >> TEST*.
  - 2) *OLS >> TEST*.

NOTE—Recording these state transitions creates the *OLWD* and *OLWU* substates for the *OLW* state and *TESTD* and *TESTU* substates for the *TEST* state.

**Table 11-10 — TAP.7 Controller behavior while in the CSM *TEST* state**

Type-0-Type-3 Reset?	Offline-at-Start-up?	SHORT Bit == 1?	PROTECT Bit == 1?	Selection Test Fail?	TESTU Sub-state	Last bit of Extension Code?	Last bit of Global Register Load?	Next state	Next behavior
Yes	No	x	x	x	x	x	x	STD	Use the Standard Protocol
Yes	Yes	x	x	x	x	x	x	OLS	Offline-at-Start-up
No	x	x	x	Yes	No	x	x	OLWD	Offline
No	x	x	x	Yes	Yes	x	x	OLWU	Offline
No	x	1	x	No	Yes	Yes	x	OLWU	Offline
No	x	0	0	No	Yes	Yes	x	OLWU	Offline
No	x	1	x	No	No	Yes	x	BDY0	Begin Check Packet
No	x	x	x	x	x	x	Yes	BDY0	Begin Check Packet
Others								TEST	Remain in state

NOTE 1—A registered version of the TMS(C) value is used within the CSM for OAC, EC, and Global Register Load processing. The CSM is realigned to the input bit stream by skipping the *POST* state of the Check Packet as shown in Table 11-10 and Figure 11-19.

- e) A TAP.7 Controller performing a selection test shall set the Potential Scan Group Membership Count Last state to *PSGMCL\_MANY*, the Scan Group Candidate Count state to *SGCC\_MANY*, and Conditional Group Membership Count (CGMC) to *CGMC\_MANY*, following the last bit of the Extension Code, provided all the following are true:
  - 1) The TAP.7 Controller has not been placed Offline per Rule 11.9.5.2 b).
  - 2) The conditions specified for Drive Protection Activation shown in Table 11-3 are met.

NOTE 2—Satisfying this rule initiates TDO and TMSC Drive Policies assuming more than one Scan Group Member and more than one Conditional Group Member (see 13.8.5 and 13.9.1.2).

- f) The scan format shall be set to a value based on the TAP.7 topology specified by the Online Activation Code[3:2] as shown in Table 11-11, provided all of the following are true:
  - 1) The TAPC state is one of the following:
    - i) *Test-Logic-Reset*.
    - ii) *Run-Test/Idle*.
    - iii) *Select-DR-Scan*.
    - iv) *Pause-IR*.
    - v) *Pause-DR*.
  - 2) OAC[1:0] is equal to 00b (a TAP.7 technology is specified).
  - 3) Any of the following are true:

- i) OAC[3:2] specifies the scan topology represented by the TOPOL Register value for a T3–T5 TAP.7.
- ii) OAC[3:2] specifies the Series Scan Topology for a T0–T2 TAP.7.
- iii) OAC[3:2] specifies all the following TAP.7 technologies, Series, Star-4, and Star-2.

**Table 11-11 — SCNFMT Register value resulting from the Short-Form Selection Sequence**

Online Activation Code[3:2]	Resulting scan format value
0—All	JScan2
1—Series	JScan2
2—Star-4	JScan3
3—Star-2	OScan1

NOTE 3—The use of the JScan2 Scan Format with T0 and T1 TAP.7s provides the behavior of using the JScan0 Scan Format.

- g) The relationship of the TAP.7 Class and controller topology selection using Online Activation Code[3:2] shall be governed by Table 11-12.

**Table 11-12 — OAC value/TOPOL Register relationships required for placement Online**

TAP.7 Class	TOPOL Register	Selectable with OAC[3:2]
T0–T2	01b—hardwired as Series	00b, 01b
T3–T5	00b—Star-4 HI-Z	00b, 10b
	01b—Series	00b, 01b
	10b—Star-4	00b, 10b
	11b—Star-2	00b, 11b

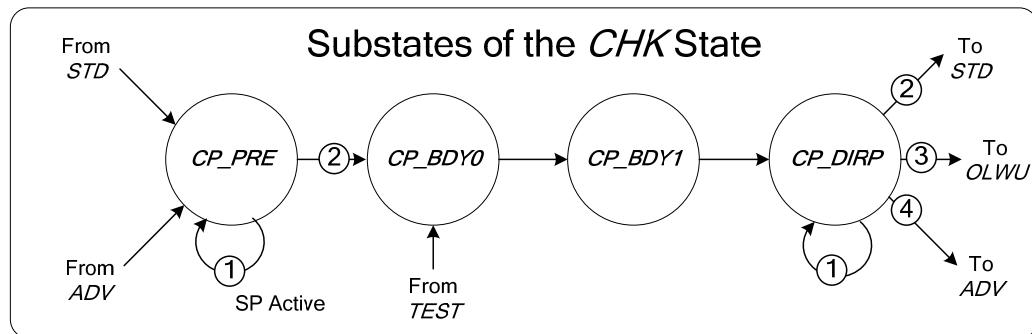
- h) The following TAP.7 Controller registers shall be set to a logic 0, provided the Global Registers are loaded:
  - 1) ZBSINH.
  - 2) SUSPEND.
  - 3) ECL.
- i) A Long-Form Selection Sequence shall store the Global Register values shown in Table 11-4.
- j) A Long-Form Selection Sequence shall overwrite the value of the SCNFMT Register that is stored as a result of Rule 11.10.5.2 g).

## 11.9.6 Check Packet state (**CHK**)

### 11.9.6.1 Description

#### 11.9.6.1.1 **CHK** substates

The **CHK** state is composed of the four sub-states shown in Figure 11-19.



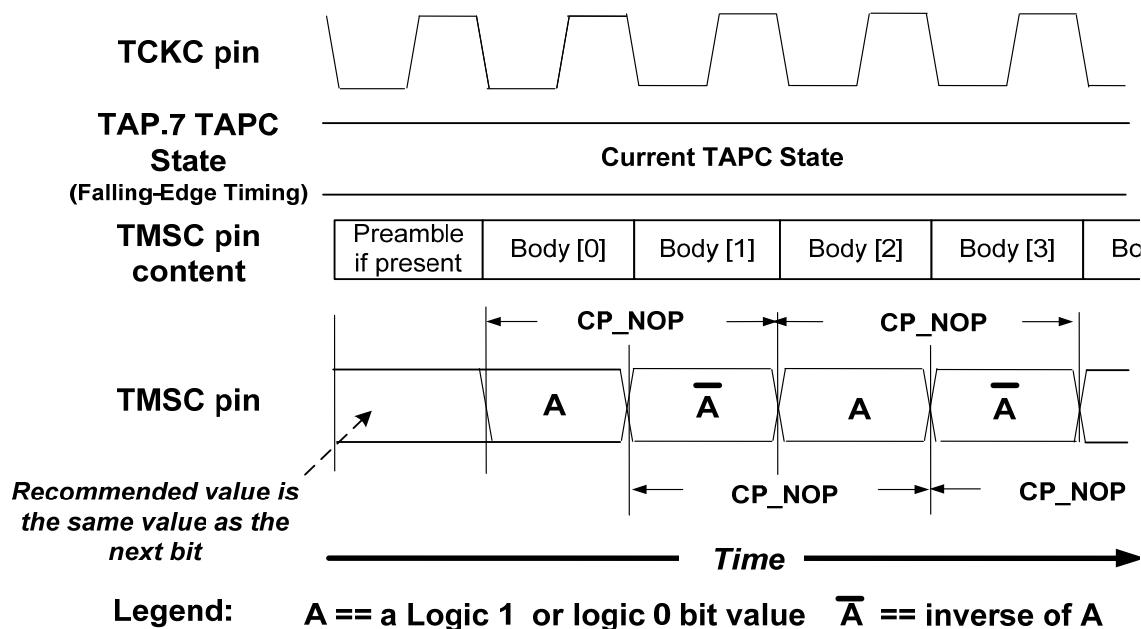
**Figure 11-19 — Substates of the CSM CHK state**

These states process a Check Packet. A Check Packet is variable length as directives embedded in the TMS(C) bit stream determine when the CP ends.

#### 11.9.6.1.2 CP examples

Examples of CP Directives and their effects are shown in the following figures:

- Figure 11-20      A CP\_NOP Directive initiated extension of the CP Body Element
- Figure 11-21      A CP\_END Directive followed by Offline operation
- Figure 11-22      A CP\_END Directive followed by the use of the Standard Protocol
- Figure 11-23      A CP\_END Directive followed by the use of the Advanced Protocol
- Figure 11-24      A CP\_RSO Directive initiating a Type-3 TAP.7 Controller reset



**Figure 11-20 — A CP\_NOP Directive initiated extension of the CP Body Element**

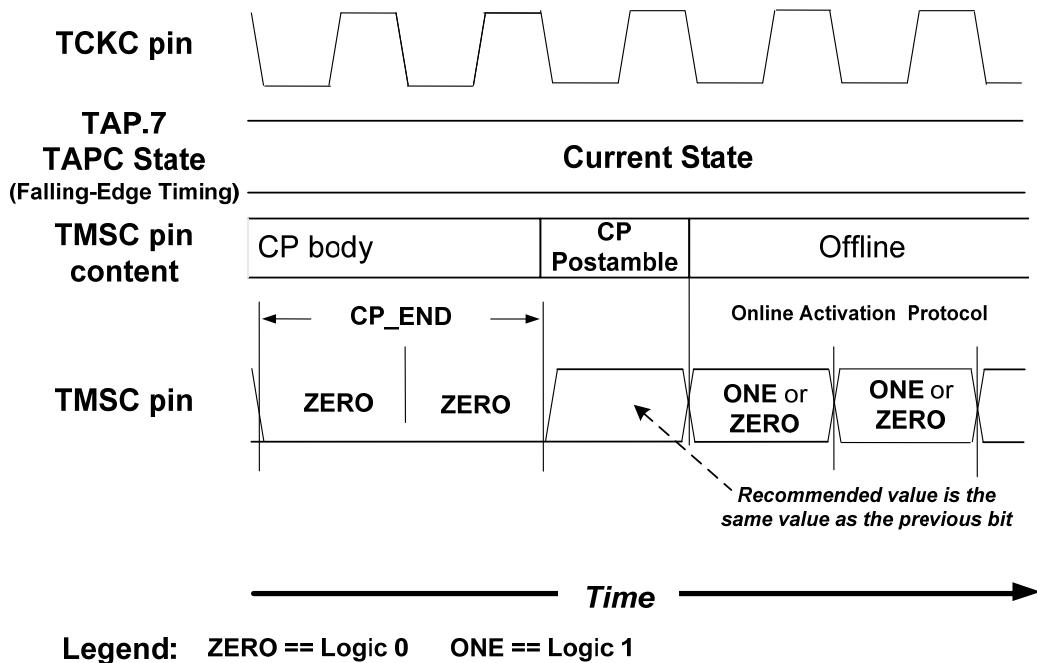


Figure 11-21 — A CP-END Directive followed by Offline operation

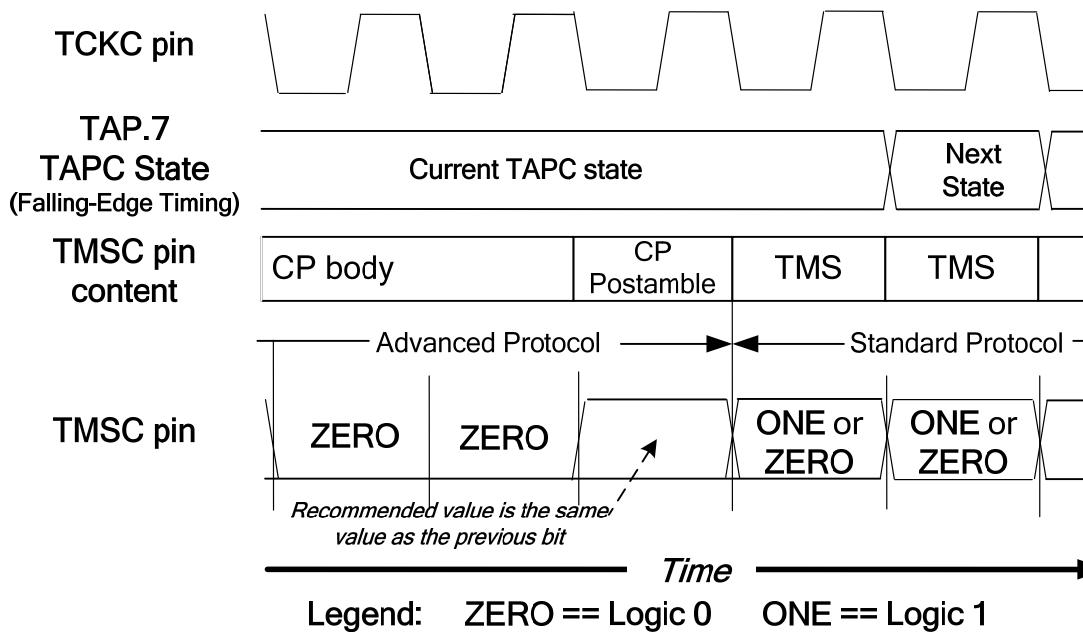


Figure 11-22 — A CP-END Directive followed by the use of the Standard Protocol

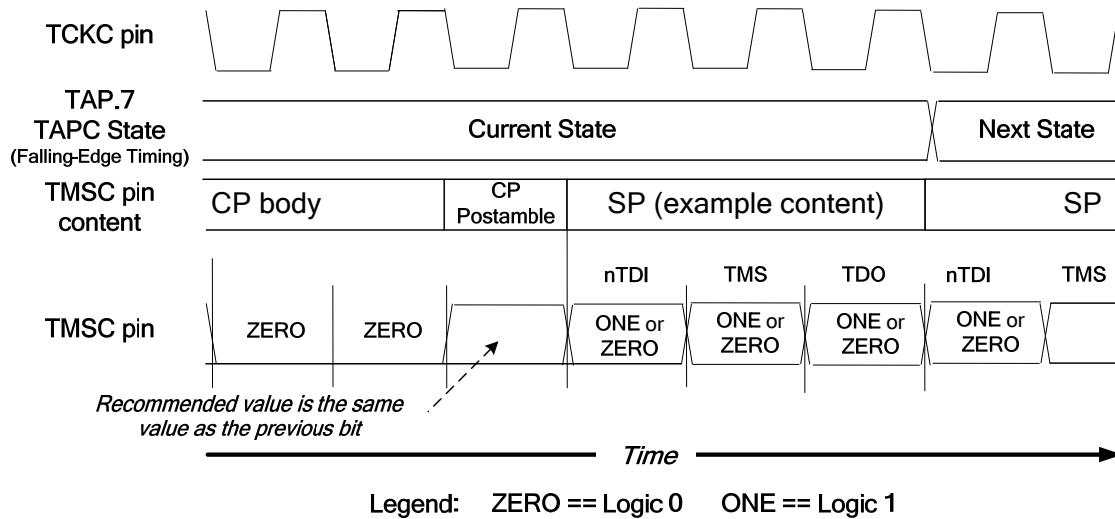


Figure 11-23 — A CP\_END Directive followed by the use of the Advanced Protocol

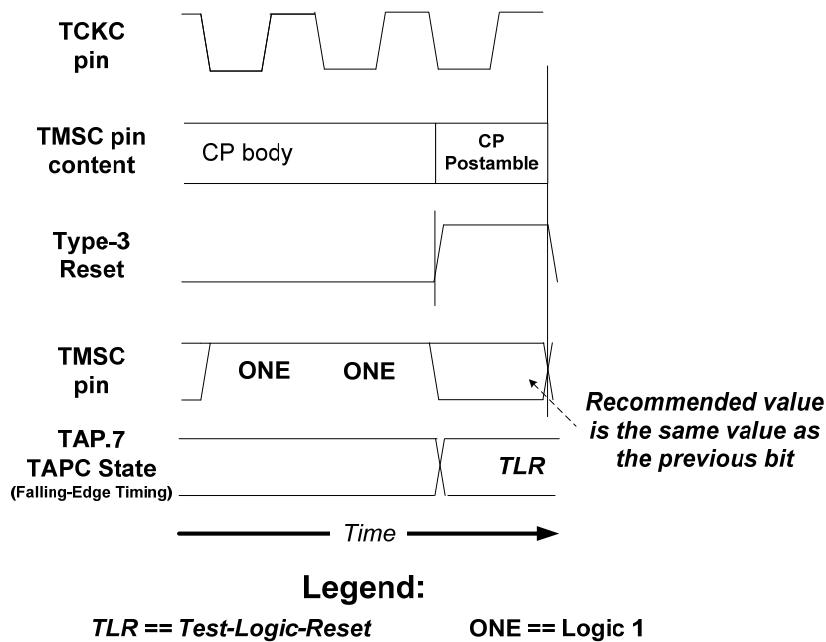
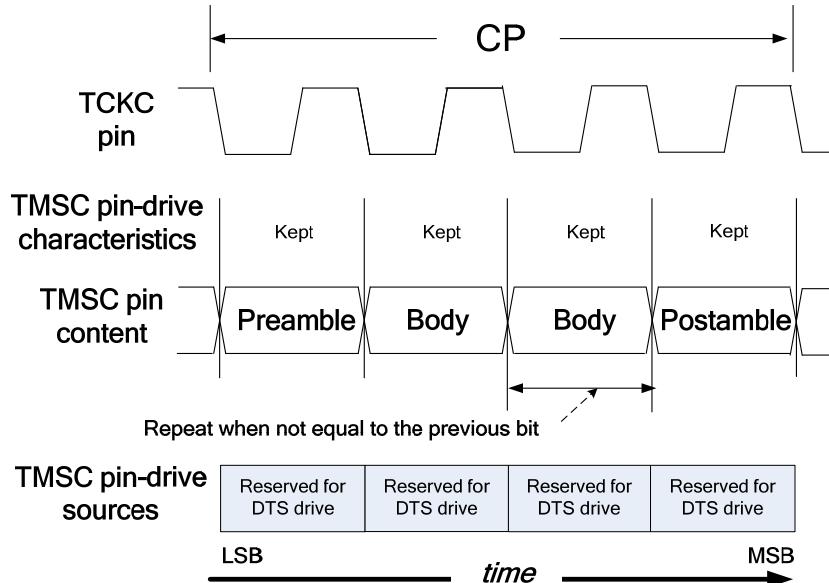


Figure 11-24 — A CP\_RSO Directive initiating a Type 3 TAP.7 Controller reset

### 11.9.6.2 Specifications

#### Rules

- Each subsequent specification in 11.9.6.2 shall only apply to a TAP.7 with Online and Offline operation.
- The CP format used by the TAP.7 Controller shall be governed by Figure 11-25.



**Figure 11-25 — CP template**

- c) The TAP.7 Controller shall begin processing the CP Preamble Element immediately after any of the following:
  - 1) The TMS value associated with the *Update-DR* state associated with Command Part Two, provided all of the following are true:
    - i) The TAP.7 Class is T4 and above.
    - ii) The TAP.7 Controller command is STFMT.
    - iii) The value to be stored in the SCNFMT Register specifies the use of the Advanced Protocol.
  - 2) The last bit of the SP associated with either a *Select-DR-Scan* or *Run-Test/Idle* state, provided these states immediately follow the *Update-DR* state associated with Command Part Two.
- d) The most recently received bit of the CP body shall be considered the MSB of the CP Directives shown in Table 11-13.
- e) Beginning with the second bit of the CP body, the last two bits of the body shall be used as a CP Directive causing the action shown in Table 11-13.
- f) The TAP.7 Controller shall use the bit following a CP Postamble Element as the first bit of a Scan Packet, provided the CSM state following the *DIRP* state is *ADV*.
- g) The TAP.7 Controller shall use the bit following a CP Postamble Element as a TMS bit, provided the CSM state following the *DIRP* state is *STD*.
- h) The TAP.7 Controller shall use the bit following a CP Postamble Element as Control Protocol, provided the CSM state following the *DIRP* state is *OLW*.
- i) The value of the Preamble and Postamble bits of a CP shall be ignored.

**Table 11-13 — CP Directives**

CP body last two bits		Action
Mnemonic	MSB LSB	
CP_END	00b	The CP Body Element is complete, Postamble Element follows
CP_NOP	01b, 10b	The CP Body Element is extended one bit period.
CP_RSO	11b	The CP Body Element is complete, Postamble Element follows, a Type-3 TAP.7 Controller reset occurs coincidently with the Postamble Element.

- j) The CSM state changes while in the CSM *CP* state shall be governed by Table 11-14.
- k) The term Configuration Fault used in Table 11-14 shall be governed by the following Rules:
  - 1) Rules 23.8.2 b) through 23.8.2 e).
  - 1) Rules 27.7.2 b).

**Table 11-14 — TAP.7 Controller behavior while in the CSM *CHK* state**

Type-0-Type-3 Reset?	Offline-at-Start-up?	State	SP-Active	Directive	Configuration Fault	JScan Scan Format	Next state	Next behavior
Yes	No	x	x	x	x	x	STD	Operation with the Std. Protocol
Yes	Yes	x	x	x	x	x	OLS	Offline-at-Start-up
No	x	C_PRE	Yes	x	x	x	C_PRE	Wait for SP completion
No	x	C_PRE	No	x	x	x	C_BDY0	Input first body bit
No	x	C_BDY0	x	x	x	x	C_BDY1	Input second body bit
No	x	C_BDY1	x	x	x	x	C_DIRP	Process directive, input additional bit
No	x	C_DIRP	x	CP_NOP	x	x	C_DIRP	Process directive, input additional bit
No	x	C_DIRP	x	CP_END	Yes	x	OLWU	Offline
No	x	C_DIRP	x	CP_END	No	Yes	STD	Operation with the Std. Protocol
No	x	C_DIRP	x	CP_END	No	No	ADV	Operation with the Adv. Protocol

NOTE—When a CP follows an SP as a result of storing a register, the SP/CP combination is queued during the first bit period of the SP associated with the Run-Test/Idle or Select-DR-Scan state following the Update-DR state of Command Part Two.

## 11.9.7 Offline-at-Start-up state (OLS)

### 11.9.7.1 Description

#### 11.9.7.1.1 Placement Online

The *OLS* state is activated by Type-0, Type-1, Type-2, and Type-3 Resets when the start-up option is Offline-at-Start-up.

#### 11.9.7.1.2 CLTAPC state initialization

The *OLS* function presumes the CLTAPC and EMTAPCs within the STL have an unknown TAPC State Machine state, even though the TAP.7 Controller's ADTAPC state is *Test-Logic-Reset*. This state initializes the CLTAPC state as follows:

- The CLTAPC is selected.

- The CLTAPC state is moved to *Run-Test/Idle* via the *Test-Logic-Reset* state using *sys\_tck* and *sys\_tms*.

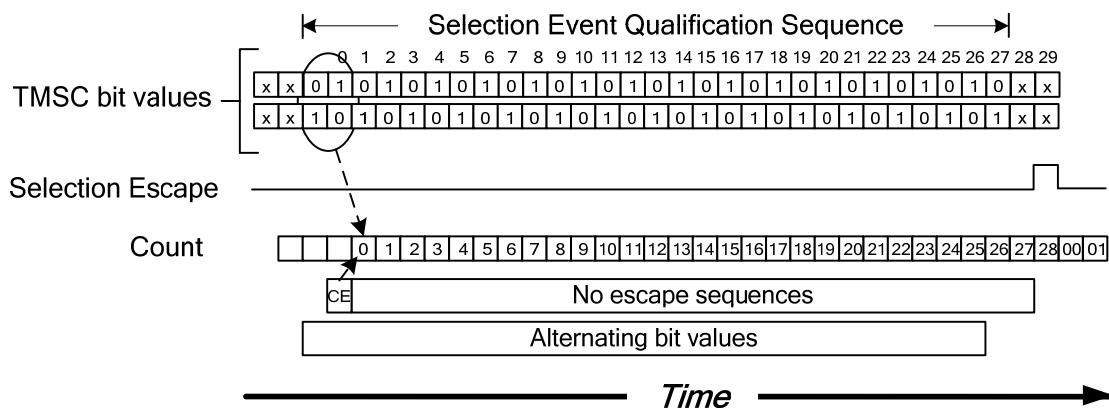
The process occurs concurrently with the detection of the TMS(C) pattern required for qualification of the Selection Escape. It begins when the Reset State Machine reaches the *NO\_RES* state and proceeds at one half the TCKC frequency. This allows the TCKC frequency to be higher than the maximum operating frequency of the STL while in the *OLS* state.

#### **11.9.7.1.3 Selection Escape qualification in the *OLS* state**

While in the *OLS* state, a Selection Escape initiates a selection test only when it is preceded by the TCK(C) and TMS(C) behavior meeting the following criteria:

- A Selection Escape occurs coincidently with bit[n]
- Each TMS(C) value in the series of 28 values [n – 2]:[n – 29] is the inverse of value of the bit preceding it
- An Escape is not detected coincidently with bits [n – 1]:[n – 28]

A sequence of TMS(C) bits satisfying the criteria is shown in Figure 11-26.



Matching bit values in bit periods [n-2] and [n-1] create a zero count value in bit period [n+1]

An escape sequence in bit period [n] creates a zero count value in bit period [n+1]

**Figure 11-26 — *OLS* Selection Escape qualification sequence**

#### **11.9.7.1.4 Example of exiting the *OLS* state**

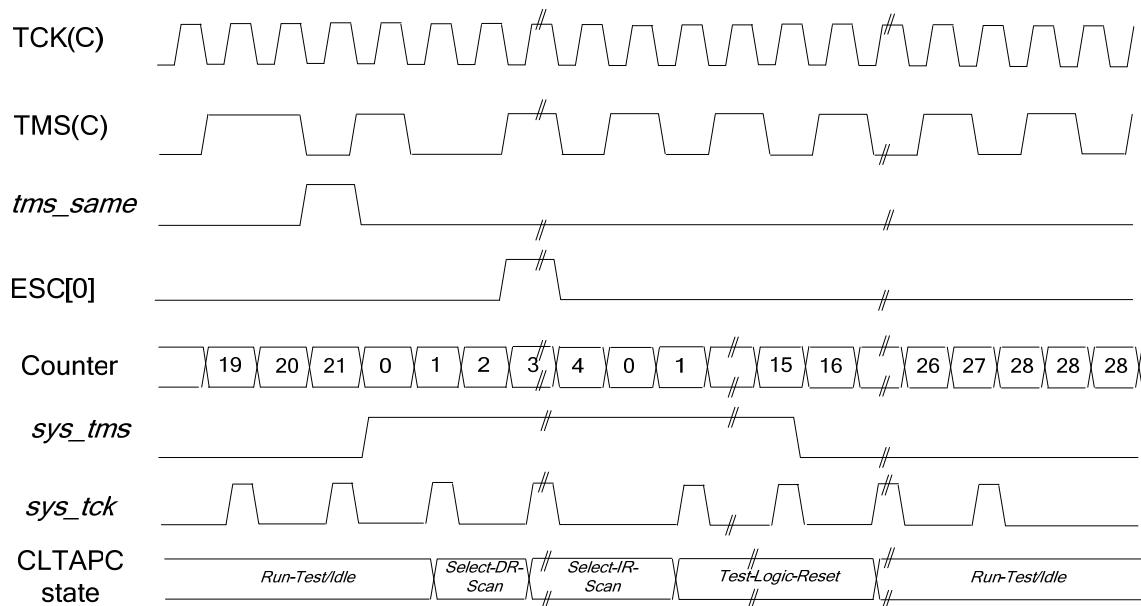
With the *OLS* state, the *CLTAPC* state can be initialized to the *Run-Test/Idle* state coincidently with Selection Escape qualification. While in the *OLS* state, a five-bit counter:

- Records the number of alternating TMS(C) values
- Is enabled to count only when the count value is less than 28
- The counter value is set to zero at the beginning of bit period k when any of the following occur:
  - The TMS(C) values in bit period k – 2 and k – 3 are the same value

- Any Escape is detected coincidently with bit period  $k - 1$

Successive data bit values are compared as shown in Figure 11-27 with the *tmsc\_same* signal. When Selection Alert Detection is implemented, a similar approach may be used to initialize the CLTAPC as it takes 127 or more TCK(C) periods before the *OLS* state may be exited as a result of a Selection Alert.

A *sys\_tck* is generated for TCK(C) periods where the count value is odd (1, 3,...etc.). The *sys\_tms* value is created with the inverse of counter bit[4]. Every time the counter is cleared, it subsequently generates a minimum of eight *sys\_tcks* with *sys\_tms* a logic 1. TMS becomes a zero for count values 16 and above. *sys\_tck* is no longer generated when the count reaches 28 and held at this value. This is shown in Figure 11-27.



**Figure 11-27 — Simultaneous Selection Escape qualification and TAPC state initialization**

### 11.9.7.2 Specifications

#### Rules

- a) Each subsequent specification in 11.9.7.2 shall only apply to a T0 and above TAP.7 with Online and Offline operation while in the *OLS* state.
- b) Following a Type-0–Type-3 Reset, once the *NO\_RES* Reset State Machine state is reached, the *CLTAPC* and *ADTAPC* states shall be moved to the *Run-Test/Idle* state using a clock frequency that is a sub-multiple of the TCKC frequency.
- c) A Selection Escape shall be qualified to initiate a selection test (i.e., cause an *OLS* to *TESTU* CSM state transition), provided all of the following are true:
  - 1) Type-0–Type-3 Reset is not occurring.
  - 2) Rule 11.9.7.2 b) has been satisfied.
  - 3) A Selection Escape occurs in coincidently with bit[n].
  - 4) Each bit value in bits [n – 2]:[n – 29] of the series of bits preceding the Selection Escape is the inverse of value of the bit preceding it.
  - 5) An Escape is not detected coincidently with bits [n – 1:n – 28].

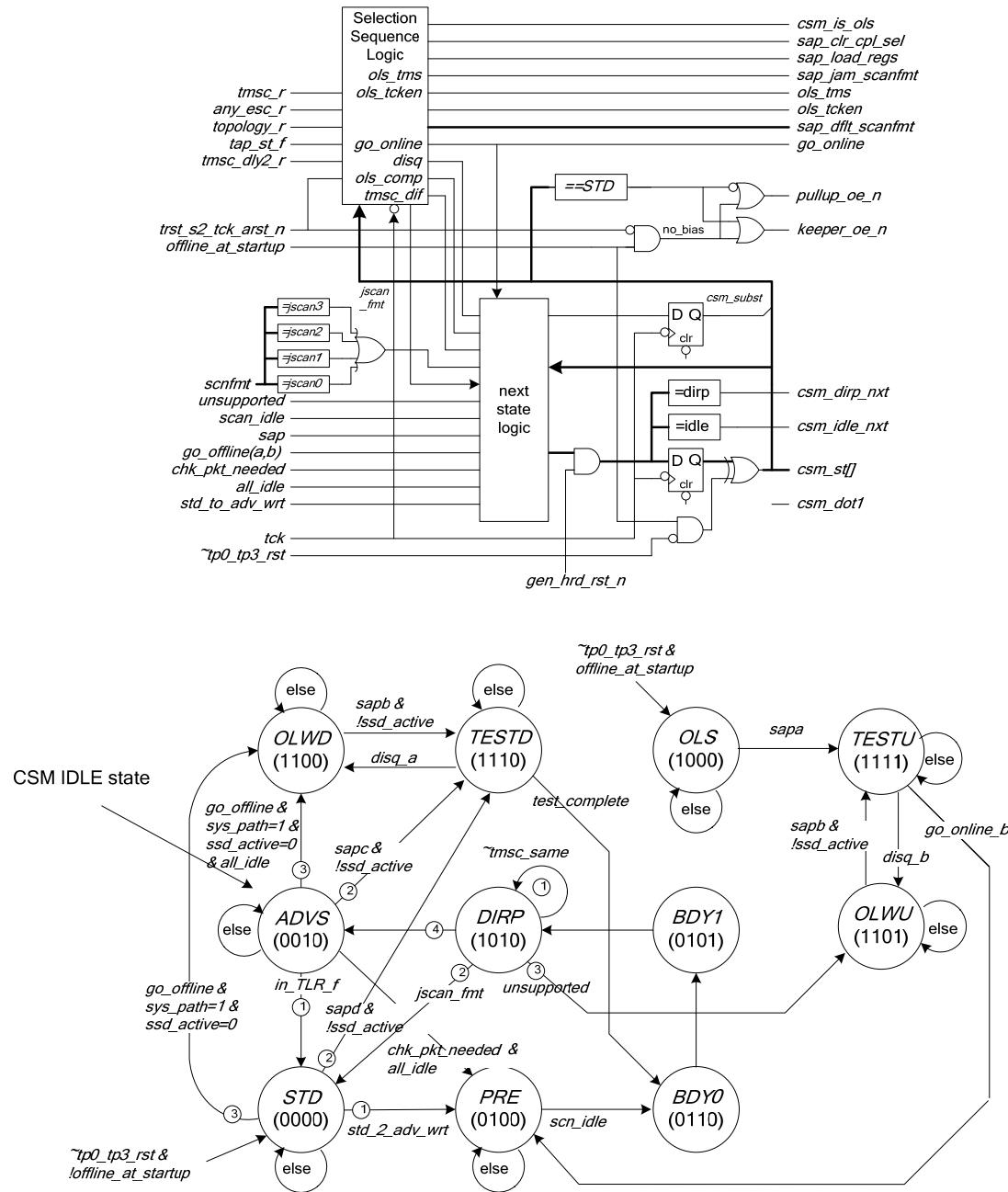
- d) The TAP.7 Controller behavioral changes while in the CSM *OLS* state shall be governed by Table 11-15.

**Table 11-15 — TAP.7 Controller behavior while in the CSM *OLS* state**

Type-0-Type-3 Reset?	Offline-at-Start-up?	Qualified OLS Selection Escape	Selection Alert	Next state	Next behavior
Yes	No	x	x	STD	Operation with the Std. Protocol
Yes	Yes	x	x	OLS	Offline-at-Start-up
No	x	Yes	x	TESTU	Initiate a Selection Test
No	x	x	Yes	TESTU	Initiate a Selection Test
No	x	No	No	OLS	Remain in state

### 11.9.7.3 An approach to implementing the CSM

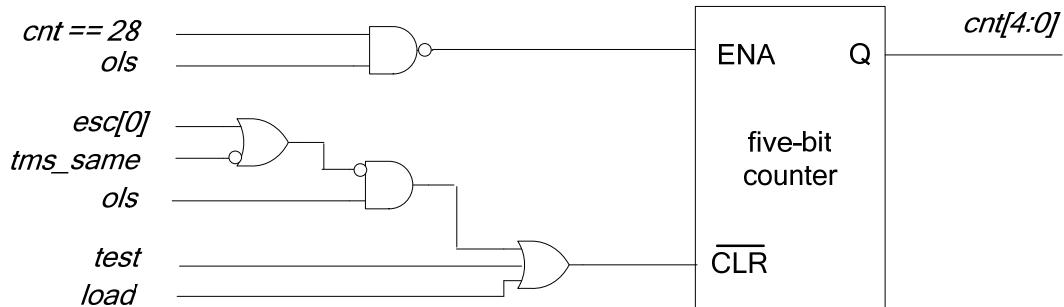
A conceptual view of the Control State Machine is shown in Figure 11-28. This function may be implemented in any manner that conforms to the rules of this specification. The implementer is encouraged to explore other implementations.



start activation code processing = *sapa* | *sapk* | *sapc* / *sapd*;  
*sapa* = (OLS & (CLTAPC initialized & ((preamble completed & selection escape) | selection alert));  
*sapk* = ((OLWU | OLWD) & (selection escape) | selection alert))  
*sapc* = (ADV & qualified selection alert);  
*sapd* = (STD | selection alert);  
*scn\_idle* = scan state machine idle;  
*std\_2\_adv\_wrt* = a value is written to the SCNFMT Register specifying the use of the Advanced Protocol while using the Standard Protocol;  
*go\_offline* = (*Run-TestIdle\_f* & control level == 6) | qualified deselection escape;  
*sys\_path* = suspend | zbs\_count > 1;  
*scr\_load* = scan state machine idle & delay state machine idle & transport state machine idle;  
*disq\_a* = selection disqualification (technology, topology, or state);  
*disq\_b* = *disq\_a* | *protect\_bit* / *short\_bit*;  
*test\_complete* = selection sequence complete | *~tmsc\_r*;  
*go\_online\_b* = selection sequence complete;  
*ssd\_active* = ssd\_processing has not begun ( *SSD\_NO* state in Figure 20-9);  
*tmsc\_diff* = (*tmsc\_r* & *~tmsc\_r\_dly*) | (*~tmsc\_r* & *tmsc\_r\_dly*);

**Figure 11-28 — Conceptual view of the CSM with all sub-states exposed**

A conceptual view of the counter function counter supporting the operation of the CSM State is shown in Figure 11-29.



See Figure 10-12 for the *tms\_same* and *esc[0]* signals

**Figure 11-29 — Counter supporting function of the OLS state**

## 11.10 Programming considerations

### 11.10.1 Escapes

#### 11.10.1.1 Selection Escape

Selection Escapes may be used to imitate a Selection Sequence provided any of the following are true:

- All technologies sharing the DTS connection are Offline
- All of the following are true:
  - The System Path is being used.
  - There are no Pause-DR Group or Pause-IR Group members.
  - Any of the following are true:
    - The Standard Protocol is being used.
    - All of the following are true:
      - The Advanced Protocol is being used.
      - The Selection Escape occurs in the first bit of an SP.
      - The first bit of the SP is sourced by the DTS.

Selection Escapes may be used concurrently with Selection Alerts as described previously.

#### 11.10.1.2 Deselection Escape

A Deselection Escape may be used provided any of the following are true:

- All technologies sharing the DTS connection are Offline (they will be ignored).
- All of the following are true:
  - The System Path is being used.
  - There are no Pause-DR Group or Pause-IR Group Members.

- Any of the following are true:
  - The Standard Protocol is being used.
  - All of the following are true:
    - The Advanced Protocol is being used.
    - The Deselection Escape occurs in the first bit of an SP.
    - The first bit of the SP is sourced by the DTS.

Although the DTS may use a Selection or Deselection Escape to park the TAP State Machine in any of its 16 states, parking the ADTAPC state in a state other than in *Pause-xR*, *Select-DR-Scan*, *Run-Test/Idle*, or *Test-Logic-Reset* requires the TAP.7 Controller be reset before it may be placed Online.

### **11.10.1.3 Reset Escape**

A Reset Escape may be used at any time.

### **11.10.2 Alerts**

#### **11.10.2.1 Deselection Alert**

A Deselection Alert may be used at any time.

#### **11.10.2.2 Selection Alert**

A Selection Alert may be used to initiate a Selection Sequence, provided:

- The technology supports the use of Selection Alerts
- All technologies sharing the DTS connection are Offline

A Selection Alert may only be used when all technology branches are Offline since an Online technology would interpret a Selection Sequence as data or control information. It may be used independently of or concurrently with a Selection Escape.

#### **11.10.2.3 Offline-at-Start-up**

It is the responsibility of the DTS to ensure that placement of a TAP.7 Controller Online when it is Offline-at-Start-up should follow these guidelines:

- When all technologies are Offline—Using either the Standard or the Advanced Protocol
- When a minimum of one TAP.7 Controller is Online—Using only the Standard Protocol

Note that placing all TAP.7 Controllers Offline before the generation of the preamble that precedes the Selection Escape or using a Selection Alert provides an easy means to ensure they do not interpret the above-mentioned preamble as TMS(C) data. The DTS should initiate this preamble as shown in Annex D when transitions from Offline-at-Start-up to Online operation are initiated while one or more TAP.7 Controllers are Online.

### 11.10.3 Selection Sequences

#### 11.10.3.1 DTS/TAP.7 Controller synchronization

The DTS is responsible for synchronizing its view of Global Registers to the Global Register values that are used with the Long-Form Selection Sequence. It is also responsible for synchronizing the value of the Extension Code STATE field to the DTS TAPC state.

#### 11.10.3.2 Short Form

It is assumed the primary use of Short-Form Selection Sequences will be the creation of Series-Equivalent Scans with multiple TAP.7 Technology Branches with a test application. With this being the case, the DTS should use Deselection and Selection Sequences in a manner so as to create a valid parking state as described in 11.8.1. It should not alter the Global Registers with commands in a manner that creates Global Register values other than the Scan Format Register value loaded by a Short-Form Selection Sequence. Should Short-Form Selection Sequences be used with a debug application, the DTS may manage Global Registers separately for each technology branch, provided the branches are selected separately.

#### 11.10.3.3 Long Form

It is assumed the primary use of Long-Form Selection Sequences will be debug applications. A Selection Sequence may be used to simultaneously load the Global Registers of all TAP.7 Controllers forming a branch. Long-Form Selection Sequences may be either Synchronizing or Non-Synchronizing, depending on the value of the PROTECT bit.

After a Long-Form Selection Sequence, the state of system is as follows:

- CLTAPCs have an inert instruction in the IR placed in the CLTAPC IR by one of the following:
  - By the DTS before a Deselection Escape (a DTS responsibility)
  - By the DTS before a command placed the TAP.7 Controller Offline
  - When power-up occurred
- The path is the System Path (there is no control level)
- The EPU Operating State is *RUNNING*

Two types of Long-Form Selection Sequences are defined using the combination of a logic 0 SHORT bit value, and the two PROTECT bit values as shown as follows:

- Non-Synchronizing (PROTECT == 0)
- Synchronizing (PROTECT == 1)

With a Non-Synchronizing Long-Form Selection Sequence, only the TAP.7 Controllers that are synchronized with their peers may be placed Online. This means only those TAP.7 Controllers performing the function of the CSM's *TEST* state with the *TESTD* sub-state may be placed Online.

With a Synchronizing Long-Form Selection Sequence, both TAP.7 Controllers that have lost and that have not lost synchronization with their peers may be placed Online. This means those TAP.7 Controllers performing the function of the CSM's *TEST* state with either the *TESTD* or *TESTU* sub-states may be placed Online. With this being the case, the number of TAP.7 Controllers that may be placed Online where their STLs are Scan Group Members is unknown. Drive Conflict Prevention is invoked since the PROTECT bit is a logic 1. Subsequent to this, the TAP.7 Controller develops its drive policy assuming multiple TAP.7 Controllers are candidates to drive the TMSC and TDOC signals (drive policies are

described in Clause 13 and Clause 14). In this case, the DTS has the following responsibilities following the Long-Form Selection Sequence:

- Established the Command Control Level.
- Re-established Scan Group Membership before interacting with CLTAPCs.
- Re-established Conditional Group Membership before interacting with the EPU using either SCNB or SCNS commands or Control Levels four or five.

If the Selection Sequence were preceded by an event permitting the placement of a TAP.7 Controller that was Offline-at-Start-up Online (either a Selection Escape with the appropriate preamble or a Selection Alert), the DTS takes the following additional steps:

- Invalidate CID zero (this CID should be reserved for placing TAP.7 Controllers that were Offline-at-Start-up Online)
- Allocate CIDs to those TAP.7 Controllers not having a valid CID

If CID zero was allocated prior to these steps, it may be reallocated to the same TAP.7 Controller using Directed CID Allocation.

Since the TAP.7 Controller's CID is zero after initialization, it is good practice not to use this CID during normal operation when either the DTS knows there is more than one TAP.7 Controller in a Star Scan Topology or does not know whether a TAP.7 Controller has been placed Offline-at Start-up. This allows placement of more than one TAP.7 Controller Online using the Re-synchronizing Long-Form Selection Sequence without immediately allocating these TAP.7 Controllers a CID. In this case, the DTS does not use CID zero until these TAP.7 Controllers allocated CIDs.

#### **11.10.4 Hang caused by a programming error**

Should a Selection or Deselection Escape occur precisely when the Delayed SSD State exits the *SSD\_SA\_DLY* state and the System Path is being used:

- The CSM state caused by a Selection or Deselection Escape inhibits SSD processing
- The SSD state blocks the qualification of Selection and Deselection Escapes

This is a hang condition created by a programming error as the guidelines above do not permit this combination. This condition is cleared with a Type-0–Type-3 Reset. It is created by the behavior specified by the combination of Rules in 11.7.2 and 20.8.2.

## 12. TAP signals

### 12.1 Introduction

This clause is applicable to T0 and above ADTAPCs. It describes the TAP.7 signals. It identifies the signals as being either mandatory or optional for the six TAP.7 Classes, T0–T5. It also describes the characteristics of these signals for each TAP.7 Class. It is applicable to all TAP.7 Classes. Clause 13 and Clause 14 define the TDO(C) and TMS(C) Signal Drive Policies.

The subject matter within this clause is organized in the following manner:

- 12.2 TAP.7 Class/signal relationships
- 12.3 Signal function and bias
- 12.4 Test Reset (nTRST and nTRST\_PD)
- 12.5 TAP.7 signal functions with corresponding IEEE 1149.1 names
- 12.6 Test Clock (TCK)
- 12.7 Test Mode Select (TMS/TMSC)
- 12.8 Test Data Input (TDI/TDIC)
- 12.9 Test Data Output (TDO/TDOC)
- 12.10 Offline-at-Start-up behavior
- 12.11 TAP connections
- 12.12 Applicability of this standard
- 12.13 Recommendations for interoperability

### 12.2 TAP.7 Class/signal relationships

#### 12.2.1 Description

The TAP.7 signals are listed as follows:

- Test Clock (TCK or TCKC)
- Test Mode Select (TMS or TMSC)
- Test Data Input (TDI or TDIC)
- Test Data Output (TDO or TDOC)
- Test Reset (nTRST)
- Test Reset Pull-Down (nTRST\_PD)

The term “signal” is used in lieu of the term “pin” to avoid the confusion that might arise from configurations involving multiple die in a package sharing the same TAP pin(s). Once a signal exits the package, it is also referred to as a signal.

A TAP.7 signal is given the same name as its TAP.1 counterpart when its characteristics are the same as those of its TAP.1 counterpart. A “C” is added to the end of a TAP.7 interface signal name when either the signal characteristics are different from those of its TAP.1 counterpart or the functionality of the signal is

different from its TAP.1 counterpart. Consequently, the TCK, TMS, TDI, and TDO signals become the TCKC, TMSC, TDIC, and TDOC signals, respectively.

All TAP.7 Classes have Test Clock and Test Mode Select signals. The Test Data Input and Test Data Output signals are mandatory with T0–T3 TAP.7s and optional for T4 and above TAP.7s. The nTRST and nTRST\_PD signals are optional with any of the TAP.7 Classes. Generally, neither of these signals nor one of these signals is implemented. The TAP.7 signal names for each class are summarized in Table 12-1.

The T4 and T5 TAP.7s are considered wide when the optional TDIC and TDOC signals are included and are considered narrow when the optional TDIC and TDOC signals are not included. A wide TAP configuration is followed by a (W) suffix and a narrow configuration is followed by an (N) suffix.

## 12.2.2 Specifications

### Rules

- a) Each subsequent specification in 12.2.2 shall apply to T0 and above TAP.7s.
- b) The names of the TAP.7 Test Clock, Test Mode Select, Test Data Input, and Test Data Output signals shall be governed by Table 12-1.
- c) The TAP.7 signals implemented for each class shall be governed by Table 12-1.

**Table 12-1 — Relationship of TAP.7 Classes/signal names**

TAP.7 Class	Signal function						Comments
	Test Clock	Test Mode Select	Test Data Input	Test Data Output	Test Reset	Test Reset Pull-Down	
T0–T2	TCK	TMS	TDI	TDO	nTRST	nTRST_PD	Legacy signaling
T3	TCK	TMS	TDIC	TDOC			TDIC/TDOC have extended functions
T4(W), T5(W)	TCKC	TMSC	TDIC	TDOC			TCKC/TMSC change function
T4(N), T5(N)	TCKC	TMSC	—	—			

- d) The relationship of TAP.7 Classes and TAP.7 Signal Characteristics shall be governed by Table 12-2.

**Table 12-2 — Relationship of TAP.7 Classes/Signal Characteristics**

TAP.7 Class	Signal function			
	T0-T2	T3	T4(W)-T5(W)	T4(N)-T5(N)
TCK(C)	IN	IN	IN	IN
TMS(C)	IN	IN	IN/OUT	IN/OUT
TDI(C)	IN	IN	IN/OUT	--
TDO(C)	OUT	IN/OUT	IN/OUT	--
nTRST	IN	IN	IN	IN
nTRST_PD	IN	IN	IN	IN

**IN/OUT** Auxiliary function permitted

NOTE—The nTRST and nTRST\_PD signals are optional signals for all TAP.7 Classes.

- e) A TAP.7 shall be implemented using one of the following options for the nTRST or nTRST\_PD signals:
  - 1) Neither the nTRST nor the nTRST\_PD signal is implemented.
  - 2) The nTRST\_PD signal is implemented, and the nTRST signal is not implemented.
  - 3) The nTRST signal is implemented, and the nTRST\_PD signal is not implemented.

#### Permissions

- f) A T4 and above TAP.7s may be implemented either with TDIC and TDOC signals or without these signals.

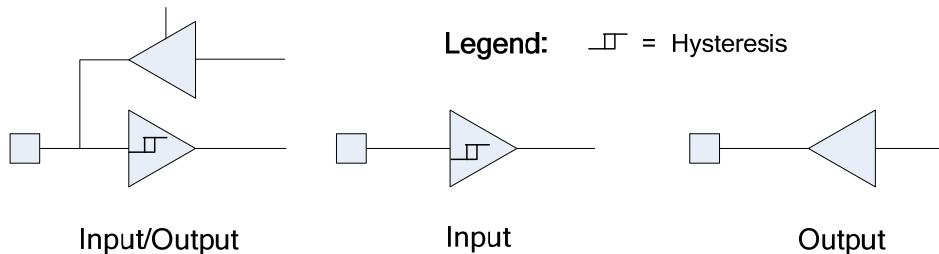
### 12.3 Signal function and bias

#### 12.3.1 Description

Two TAP.7 Signal Characteristics are covered in this standard:

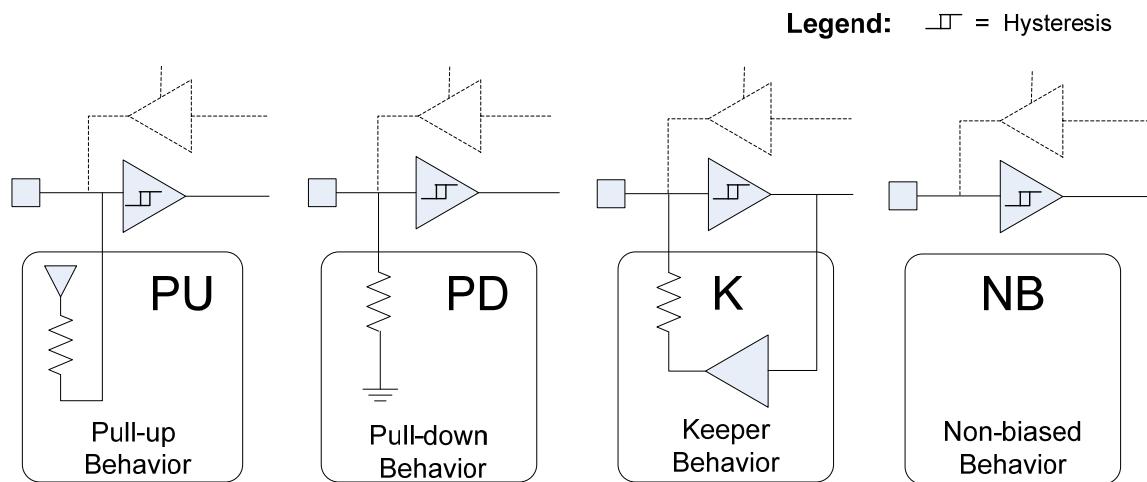
- I/O structure
- Signal bias

The I/O structures in this specification are represented as shown in Figure 12-1.



**Figure 12-1 — Conceptual view of TAP.7 signal functions**

Signals with input functionality may be operated with the various input biases shown in Figure 12-2.



**Figure 12-2 — Conceptual view of bias for TAP.7 input and input/output signal functions**

The TAP.7 functionality is specified to provide compatible signal biases. The PU, PD, and K bias functions are capable of providing a bias for other signals with an NB bias when they are connected. Signals operated with biases that are of a different type may create unacceptable logic levels or create input oscillations when they are connected but not driven. These types of connections should be avoided.

System configurations that deploy T4(N), T4(W), T5(N), and T5(W) TAP.7s should operate these devices in a manner where at least one of the following is true:

- At least one chip (or other source) supplies a Test Mode Select signal bias that is either PU or K. This ensures the input buffer is biased at all times.
- After system power-up and before a Test Clock is detected, all chips sharing connectivity operate with input circuitry on signals other than the TCK(C) that do not require an input bias (the TCKC is also permitted to not have an input bias).

### 12.3.2 Specifications

#### Rules

- a) Each subsequent specification in 12.3.2 shall apply to T0 and above TAP.7s.
- b) A signal bias designation of PD shall designate that the design of the circuitry fed from a signal to the TAP.7 Controller circuitry is such that an undriven input produces a logic 0.
- c) A signal bias designation of PU shall designate that the design of the circuitry fed from a signal to the TAP.7 Controller circuitry is such that an undriven input produces a logic 1.
- d) A signal bias designation of K shall designate that the design of the circuitry fed from a signal to the TAP.7 Controller circuitry is such that an undriven input produces the logic value last detected by the input buffer.
- e) A signal bias designation of NB shall designate that the design of the circuitry fed from a signal to the TAP.7 Controller circuitry is such that an undriven input does not provide a bias to create a specific logic level to the input buffer.
- f) The PU, PD, and K bias functions shall be capable of providing a bias for other signals to which they are connected as shown in Table 12-3.

**Table 12-3 — Signal bias effects transferred to signals sharing a connection**

Signal bias	Transferred effects
Undefined	Undefined
NB	No effect
PU	Bias to a logic 1
PD	Bias to a logic 0
K	Bias provided by keeper value

- g) When there is an expectation that the TAP.7 logic can be powered-down without disturbing the operation of other TAPs or technologies sharing the TAP signals, the TAP.7 signals shall have high impedance while the TAP.7 logic is powered-down.

### Recommendations

- h) The TAP.7 input buffers for the Test Clock and Test Mode Select signals should include hysteresis and other circuits that minimize susceptibility to noise and transmission-line reflections.

## 12.4 Test Reset (nTRST and nTRST\_PD) signals

### 12.4.1 Description

This standard provides for two signals (nTRST and nTRST\_PD) that reset the test logic. These signals initiate the test-reset function but have different pin biases. In most cases, the chip architect chooses to implement the nTRST signal, the nTRST\_PD signal, or neither of these signals. Both signals are seldom, if ever, implemented on the same chip.

The nTRST signal specified by this standard is the equivalent of the TRST\* signal specified by IEEE Std 1149.1 (see Table 1-1). When this signal is not driven (for example, left unconnected within a system), the default logic 1 signal created by the PU bias associated with the signal does not reset the test logic. In some applications, especially those that are safety conscious or in very harsh electrical environments (e.g., health care or automotive), it is desirable that the test logic be reset at all times in the application when the test equipment is not connected and utilizing this logic. The nTRST\_PD signal provides this functionality as it has a PD bias. The test logic is reset with the default a logic 0 level created by this signal bias. Some chip manufacturers have chosen to implement the nTRST\_PD signal instead of an nTRST signal when creating chips compatible with the IEEE Std 1149.1. These chips have IEEE 1149.1-Non-disruptive Behavior.

Since the TAP.7 nTRST and nTRST\_PD signals within a system have different default values, they cannot be tied together as the default logic levels conflict with each other, unless the default values are continuously overdriven by a strong driver. If they are tied together without the strong driver being present, the default logic levels create a marginal logic level and undefined behavior. Consequently, the nTRST and nTRST\_PD signals will be kept separate until they reach a driver that overdrives the weak drives creating the default logic levels. If this driver is part of the target system, these signals may be connected within the target system. If there is no driver within the target system, these signals will be brought to separate connector signals so they may be controlled by test equipment. The test equipment is free to control these signal separately or connect them within the test equipment to a continuously driven strong driver.

## 12.4.2 Specifications

### Rules

- a) Each subsequent specification in 12.4.2 shall apply to T0 and above TAP.7s.
- b) The design of the circuitry fed from the nTRST\_PD signal to the TAP.7 Controller circuitry shall include a PD bias.
- c) The design of the circuitry fed from the nTRST signal to the TAP.7 Controller circuitry shall include a PU bias.

## 12.5 TAP.7 signal functions with corresponding IEEE 1149.1 names

### 12.5.1 Description

The functionality of a TAP.7 signal name that is the same name as a TAP signal defined in the IEEE Std 1149.1 is covered by Rule 12.5.2 b).

### 12.5.2 Specifications

### Rules

- a) Each subsequent specification in 12.5.2 shall only apply to T0–T3 TAP.7s.
- b) When a T0–T3 TAP.7 signal name is the same as the name of a TAP signal name defined in IEEE Std 1149.1, the functionality of this signal shall be governed by IEEE Std 1149.1 unless stated otherwise in this specification.

## 12.6 Test Clock (TCK)

### 12.6.1 Description

The Test Clock signal provides the clock for the test logic for TAP.7 Classes defined by this standard. This signal is held at a logic 1 when there is no DTS connection to the TAP.7 as it has a PU pin bias. The DTS or another Test Clock source may stop the Test Clock at either a logic 0 or a logic 1 for any duration. Note that stopping TCK(C) at a logic 1 for greater than one millisecond periods may activate power-down modes described in Clause 18, provided these power-down modes are implemented.

### 12.6.2 Specifications

### Rules

- a) Each subsequent specification in 12.6.2 shall apply to T0 and above TAP.7s.
- b) The characteristics of the TCK(C) signal shall be governed by Table 12-4.

Table 12-4 — TCK/TCKC signal behavior relationships

TAP.7 Class	TAP interface logic is powered?	Start-up option	Signal function	Signal bias
T0 and above	No	x	Undefined	Undefined
	Yes	x	Input only	PU

- c) In the absence of a TAP.7 Controller reset, stored state devices that are clocked by TCK(C) shall retain their state indefinitely when either of the following is true:
  - 1) The signal value applied to the TCK(C) signal is stopped at either a logic 0 or a logic 1.
  - 2) There is no signal applied to the TCK(C) signal.
- d) When the TCK(C) signal bias type shown in Table 12-4 is a value other than undefined, the TAP.7 TCK(C) Signal Characteristics shall not prevent DTS communication with other TAP.7s.

### Permissions

- e) The bias for the TCK(C) signal may be implemented as NB.

### Recommendations

- f) Care should be taken to ensure the load presented by the TCK(C) signal is minimized since the TCK(C) signals for many components may be controlled from a single driver.
- g) Care should be taken to minimize the delay from the Test Clock edges to a change in state of a TAP.7 output since the TAP operating frequency is affected by this delay.

## 12.7 Test Mode Select (TMS/TMSC)

### 12.7.1 Description

The behavior of the Test Mode Select signal is determined by the start-up option and the use of the Standard Protocol.

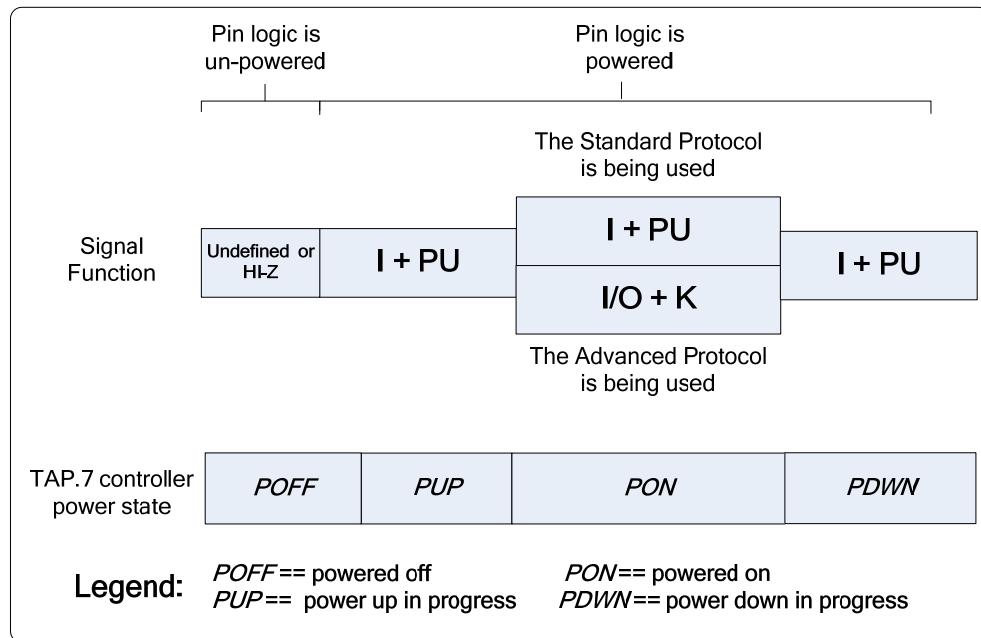
#### 12.7.1.1 Online start-up

When the TAP.7 Controller start-up provides Online operation, the TMSC signal circuitry has the behavior described as follows:

- With no power:
  - The circuitry's impedance is undefined or high impedance depending on its design
- With power:
  - The circuitry functions as an input
  - The circuitry has a PU bias

After start-up, the TMSC signal circuitry has a PU bias when the Standard Protocol is being used and a keeper bias otherwise.

This behavior is shown in Figure 12-3.



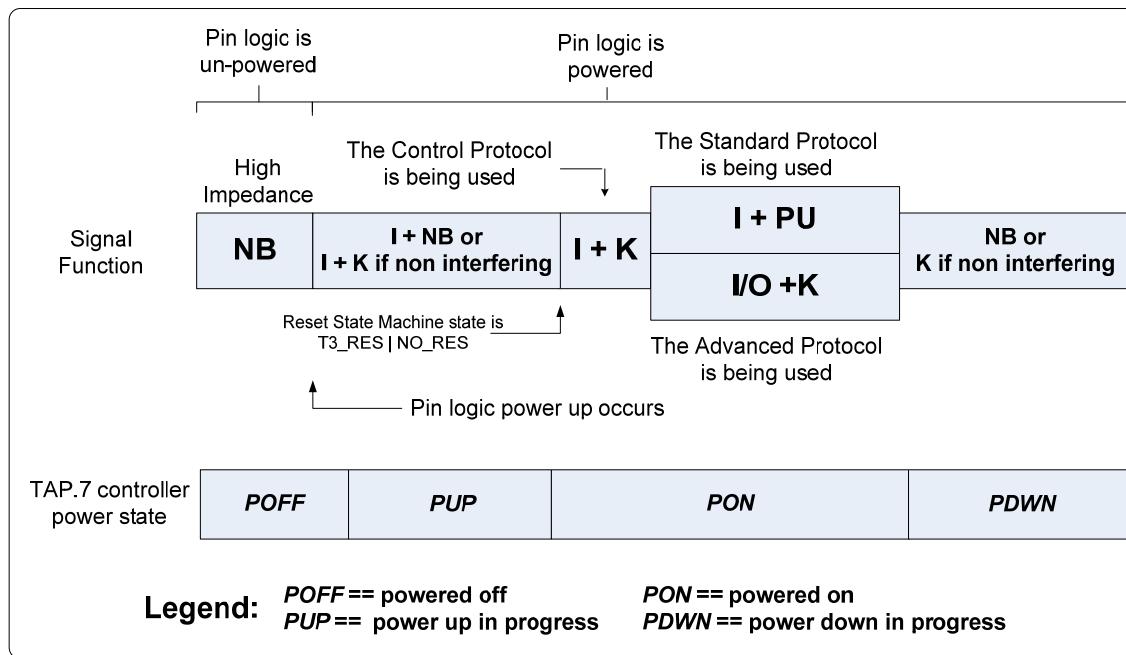
**Figure 12-3 — Online-at-Start-up Test Mode Select/signal bias relationships**

#### 12.7.1.2 Offline start-up

When the TAP.7 Controller start-up specifies Offline-at-Start-up operation, a Type-0–Type-3 Reset sets the CSM state == *OLS*. In this case, the TMSC signal circuitry has the behavior described as follows:

- With no power, the circuitry is:
  - High impedance or high impedance depending on its design
- With the interface circuitry powered and a TAP.7 Controller power state of:
  - Power is off ( $P_{OFF}$ ), power-up ( $P_{UP}$ ), and power-down ( $P_{DWN}$ )—One of the following:
    - NB bias
    - K bias, provided this bias is created by Chip-Level Logic with the falling edge of the TCK(C) in a manner that does not interfere with the use of the TAP for communication between the DTS and other TAP.7 Controllers
  - $P_{ON}$ —One of the following:
    - K bias, provided this bias is created by Chip-Level Logic with the falling edge of the TCK(C) in a manner that does not interfere with the use of the TAP for communication between the DTS and other TAP.7 Controllers
    - One of the following:
      - NB bias for *AS\_RES\_A* and *RESC* State Machine states
      - PU bias for *NO\_RES* and *T3\_RES* State Machine states when the Standard Protocol is being used
      - K bias for *NO\_RES* and *T3\_RES* State Machine states when the Standard Protocol is not being used

This behavior is shown in Figure 12-4.



**Figure 12-4 — Offline-at-Start-up Test Mode Select/signal bias relationships**

The *NO RES* and *T3 RES* Reset State Machine states indicate there have been two TCK(C) signal falling edges since a Type-0–Type-2 Reset. This function may be moved to the Chip-Level Logic. Once two clocks occur, the keeper logic should operate properly as it has access to the correct signal value. The bias is changed to PU when the Standard Protocol is used.

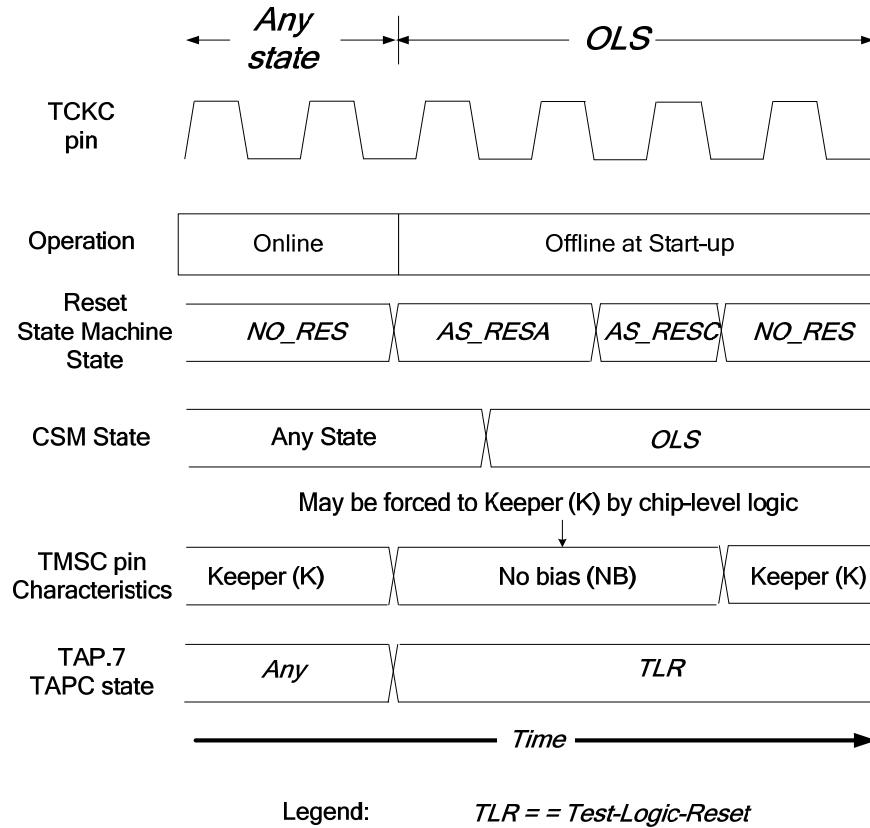
With Offline-at-Start-up operation, the DTS communication with other TAP.7s is not affected while the TAP.7 interface logic is:

- Powered
- Being powered-down
- Powered-down
- Being powered-up

When the TMSC signal bias is NB, the bias of the TMSC input buffer is handled in one of two ways:

- It is not required due to the combination of an input buffer design that accommodates floating input control and the operation of system logic (the buffer can be enabled after detection of TCK(C) activity)
- It is provided by a connection to another chip whose signal has operating bias circuitry or by another means to avoid a floating TMSC input

A Type-0–Type-2 reset is shown placing the TAP.7 in the Offline-at-Start-up state in Figure 12-5.



**Figure 12-5 — A Type-0-Type-2 Reset initiating Offline-at-Start-up operation**

#### 12.7.1.3 Combined view of Online and Offline-at-Start-up TMS(C) signal behaviors

The combined behavior of the TMS(C) signal circuitry described in 12.7.1.1 and 12.7.1.2 is shown in the form of a state machine in Figure 12-6.

This operation of the TMS(C) signal circuitry that is shown in Figure 12-6 interacts with the power-control features of the TAP.7 Controller and system operation in the following ways:

- The TAP.7 Controller power-management capability (described in 18.10) complements the Offline-at-Start-up capability
- A Chip-Level Power Controller may manage the power of the:
  - TAP I/O buffers
  - TAP.7 Controller
- TMS(C) signal biases may be a mixture of:
  - NB, PU, and K when the DTS continuously drives this signal with the Standard or Control Protocols
  - NB and K biases when using the Advanced and Control Protocols

With no chip power, the system is expected to have the following attributes:

- TAP buffers
- Have no power
- Should be at a high-impedance level, if there are expectations that the DTS may communicate with the TAPs of other chips sharing the TAP connection
- TAP.7 Controller is not powered

When a chip powers up, it is expected that one of the following Power-Control Paradigms is used:

- A Chip-Level Power Manager controls the power of:
  - The TAP buffers.
  - The TAP.7 Controller.
  - Other chip subsystems.
- The TAP.7 Controller is automatically powered at the same time as the TAP buffers and remains powered while the chip is powered.

Information on the relationship of TAP.7 Controller power-management states and the operation of the CSM is shown in Annex D, with Figure D-1 and Figure D-2.

In the case where TAP.7 Controller power is managed with a Chip-Level Power Manager:

- The TCK(C) and TMS(C) signals are connected directly to the Chip-Level Power Manager in addition to the TAP.7 Controller as detailed in 18.10.
- Once the TAP buffers are powered, TCK(C) signal activity and a logic 0 TMS(C) signal value initiates TAP.7 Controller power-up.
- A Type-0 Reset is asserted while the TAP.7 Controller is unpowered, powered-up, and powered-down.
- The cycling of TAP.7 Controller power occurs as shown in Figure 12-3 and Figure 12-4.

TCK(C) activity is required following the Type-0 Reset to create the *T3\_RES* and *NO\_RES* states. For this to occur, TCK(C) will:

- Toggle with the TMS(C) signal a logic 0 to initiate the power-up of the TAP.7 Controller
- Continue toggling for the period of time needed to get the TAP.7 Controller powered and the release of the Type-0 Reset
- Toggle two additional times to get the Reset State Machine state to *NO\_RES* and to get the TMS(C) signal K bias applied

When the TAP.7 Controller is fully powered, it is placed either Online or Offline-at-Start-up. When the TAP.7 Controller is placed Offline-at-Startup, the TMS(C) pin bias remains NB:

- Unless forced to be K by Chip-Level Logic as described shortly
- Until the Reset State Machine state advances from the *AS\_RESA* state to either the *T3\_RES* or *NO\_RES* state

The TMSC pin bias of NB is not a problem when any of the following are true:

- At least one TAP provides a pin bias

- The input buffer is designed to have no bias
- The DTS drives the TMS(C) signal

When the TMS(C) pin bias begins as NB and becomes K:

- It is expected that the TMS(C) keeper value will be established prior to the bias becoming K.
- TCK(C) is operational at this point as it causes the Reset State Machine state progression needed to cause this change.

Once a TAP.7 Controller is placed Online, its TAP has the same bias as all other Online TAPCs sharing the DTS connection.

In the case where a TAPC is placed Offline other than at start-up, the TMS(C):

- Bias becomes a K bias
- Keeper merely follows the last driven TMS(C) signal value

It does not matter whether other TAPs are operating with a PU bias or K bias when either of the following is true. The TMS(C) signal is being:

- Driven by a DTS or TS source
- Kept at a logic 0 or a logic 1 with K bias(s)

Care should be taken when breaking the DTS/TS connection as there is a possibility that a bias conflict may be created when this operation is not handled properly. The improper handing of this operation is considered a user error. A bias conflict may cause higher TS power consumption but should not cause a component failure.

If the DTS/TS connection is broken without first creating the *Test-Logic-Reset* state, there is a chance that the connection may be broken while the TMSC logic value is a logic 0 with TAPs that have a K bias keeping the logic 0. If breaking this connection also causes the reset of some TAPCs and not others, or keeper behavior creates at least one logic 1 keeper value, then some TAPs may have PU biases or logic 1 K biases while other TAPs may have logic 0 K biases. The logic 1 PU or K biases will conflict with logic 0 K biases.

The potential for a bias conflict as described above is easily avoided with a proper shutdown procedure. This shutdown procedure places all TAP.7 Controllers in the following state:

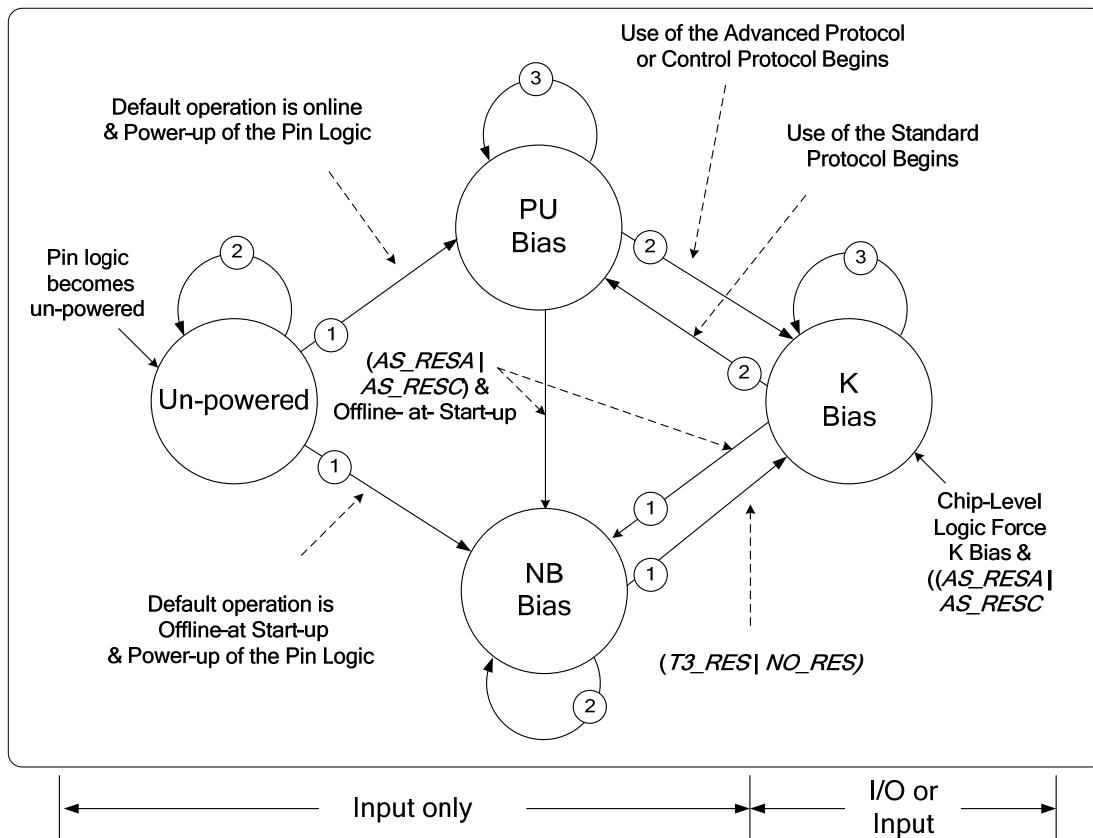
- Online
- Using a JScan Scan Format
- The *Test-Logic-Reset* TAPC state
- The TMS(C) signal parked at a logic 1

With this state, the DTS/TS connection may be broken with few if any consequences.

A Chip-Level Force K bias function is allowed as shown in Figure 12-6 and Table 12-5. This function may be implemented to establish an initial value on the TMSC pin. This feature should be used sparingly as it is provided for a specific case of initialization. It may cause system malfunctions if it is misused.

It provides a means to establish a logic 1 level with the keeper after the hardest of system resets. It is not intended to be used when a chip is expected to be powered-up when chips in the remainder of the system are already operating. A system designer should know the role of chips in his system and use this feature as

recommended, if it is used at all. It is expected that the Chip-Level Force K bias function is not implemented in most chips.



**Figure 12-6 — Test Mode Select signal bias changes**

The TMSC signal functions as an input at all times independently of whether the signal is driven.

## 12.7.2 Specifications

### Rules

- Each subsequent specification in 12.7.2 shall apply to T0 and above TAP.7s.
- The TMS/TMSC Signal Characteristics specified shall be independent of the TMSC I/O voltage level.
- The characteristics of the TMS/TMSC signal shall be governed by the following tables:
  - Online-at-Start-up—Table 12-5.
  - Offline-at-Start-up—Table 12-6.

**Table 12-5 — TMS/TMSC signal behavior relationships with start-up Online**

TAP signal logic is powered?	Signal function	Signal bias
No	Undefined	Undefined
Yes	Input only	PU

**Table 12-6 — TMS/TMSC signal behavior relationships with start-up Offline**

TAP signal logic is powered ?	TAP.7 controller power state == PON	Reset State Machine state == (AS_RESA   AS_RESC)?	STD Protocol ?	Chip-level force K bias?	Signal function	Signal bias
No	x	x	x	x	High impedance	NB
Yes	No	x	x	No	High impedance	NB
Yes	No	x	x	Yes	High impedance	K
Yes	Yes	Yes	x	No	High impedance	NB
Yes	Yes	Yes	x	Yes	Input only	K
Yes	Yes	No	Yes	x	Input only	PU
Yes	Yes	No	No	x	I/O	K

- d) When the Chip-Level Force K Bias function shown in Table 12-6 is used, it shall only be used to establish a logic 1 TMSC signal value that may subsequently be kept by this function.

### Recommendations

- e) Care should be taken to ensure the load presented by the TMS(C) signal is minimized since the TMS(C) signals for many components may be controlled from a single driver.
- f) The Chip-Level Force K Bias function should only be used provided there is no expectation that the chip is to be powered-up while the DTS communicates with other TAP.7 Controllers.

## 12.8 Test Data Input (TDI/TDIC)

### 12.8.1 Description

The electrical and timing characteristics of the TDI signal are the same as for the TDI signal in IEEE Std 1149.1 when the TDI signal provides the TDI function. This is called Legacy Signal Behavior. With this behavior, the TAP circuitry for this signal behaves as follows.

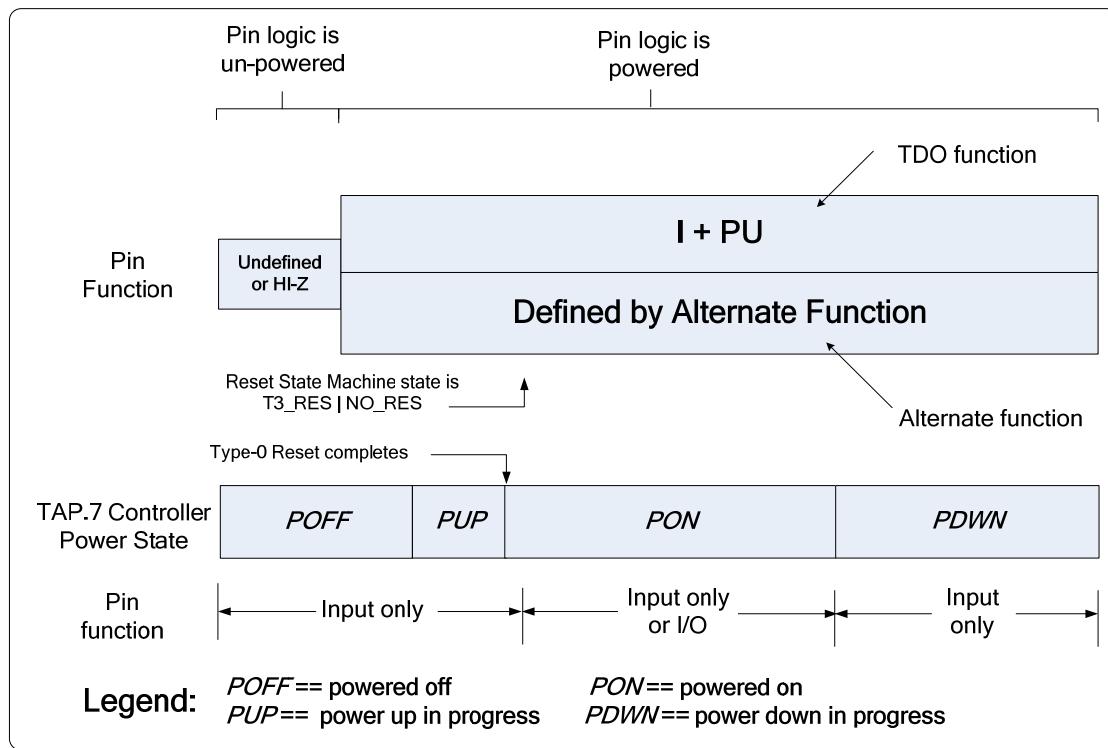
- With no power:

  - The circuitry's impedance is undefined or at a high-impedance level depending on its design

- With power:

- The signal operates as an input with a PU bias only for legacy operation
- The circuitry has a PU bias

This is shown in Figure 12-7.



**Figure 12-7 — Test Data Input/signal bias relationships for legacy operation**

See Rule 12.8.2 d) for the TDIC Signal Characteristics when the signal function is programmable with T4 and above TAP.7s.

## 12.8.2 Specifications

### Rules

- a) Each subsequent specification in 12.8.2 shall apply to T0 and above TAP.7s.
- b) The TDI/TDIC Signal Characteristics shall be independent of the I/O voltage level.
- c) The characteristics of the TDI and TDIC signals shall be governed by Table 12-7, provided the signal function is the TDI function.

**Table 12-7 — TDI/TDIC signal behavior relationships**

<b>TAP signal logic is powered?</b>	<b>Signal function</b>	<b>Signal bias</b>
No	Undefined	Undefined
Yes	Input only	PU

- d) The characteristics of the TDIC signal when the TDI/TDIC signal is providing an auxiliary function shall be defined by the auxiliary function.
- e) The TDIC signal shall be driven by the TAP.7 only when all of the following are true:
  - 1) The TAP.7 Class is T4 and above.
  - 2) The signal function is the auxiliary signal function.
  - 3) The auxiliary signal function initiates drive of the signal.
  - 4) Rule 12.8.2 f) is satisfied.
- f) When the TDIC signal has programmable functionality and the default function is an auxiliary function, the signal shall remain at a high-impedance state after chip power-up until a function understanding the use of this signal confirms the signal function is an auxiliary function within the system and enables output drive.
- g) The asynchronous or synchronous detection of a Type-0–Type-3 Reset shall cause the TDIC signal to assume the default TDIC signal function.
- h) The function of the TDIC and TDOC signals shall change at the same time when the function of either of these signals is changed.

### Recommendations

- i) Since the TDIC signals for many components may be controlled from a single driver when connected in a Star-4 Scan Topology, care should be taken to ensure that the load presented by the TDIC signal is minimized.

### Permissions

- j) The TDIC and TDOC signals may be implemented as a pair with a T4 and above TAP.7.
- k) With a T4(W) and T5(W) TAP.7, the function of the TDIC signal may be either of the following:
  - 1) The equivalent of the function of the T3 TAP.7 TDIC signal.
  - 2) Programmable with the programming selecting either the equivalent of the function of the T3 TAP.7 TDIC signal or an auxiliary function.
- l) When the TDIC signal function is programmable with a T4(W) and T5(W) TAP.7, the default signal function created by a Type-0–Type-4 Reset may be either of the following:
  - 1) The equivalent of the function of the T3 TAP.7 TDIC signal.
  - 2) An auxiliary function.
- m) When the TDIC signal function is programmable with a T4(W) and T5(W) TAP.7, the auxiliary signal function may be the T3 TAP.7 TDIC signal function.

- n) The STL may inhibit a change in the APFC Register value when it intends to use the TDIC and TDOC pins as auxiliary functions (see the TAPWLCK bit in Table 9-15).

## 12.9 Test Data Output (TDO/TDOC)

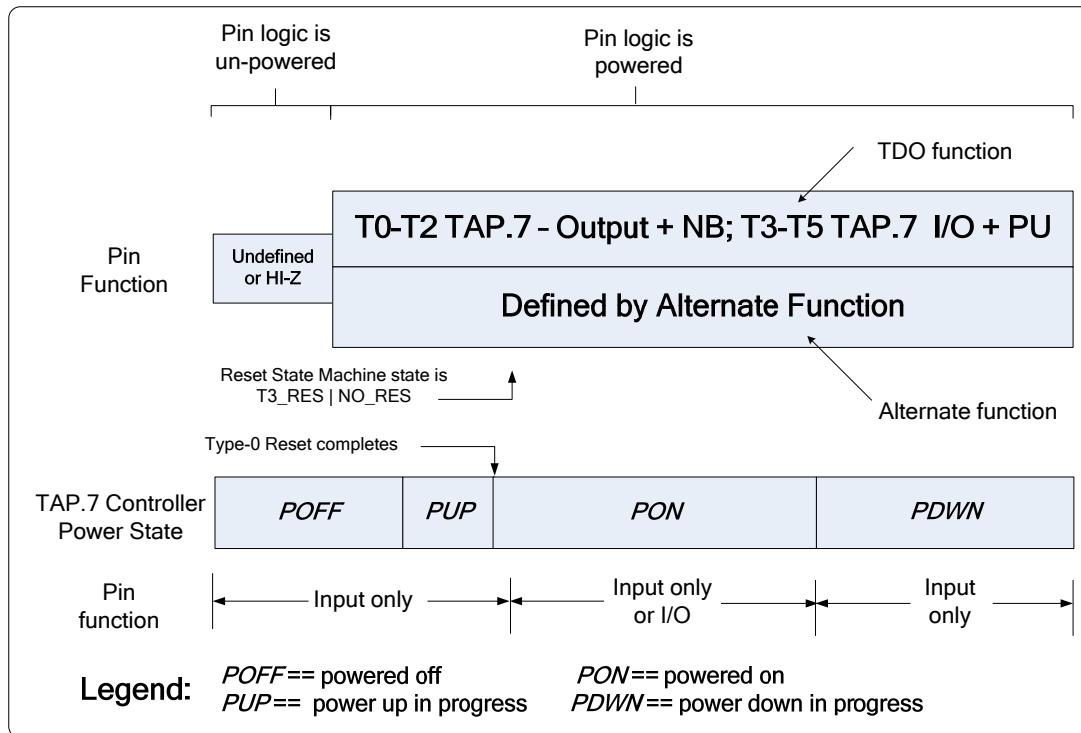
### 12.9.1 Description

With a T0–T2 TAP.7, the electrical and timing characteristics of the TDO signal are the same as for the TDO signal in IEEE Std 1149.1. This is called Legacy Signal Behavior.

- With no power:
  - The circuitry's impedance is undefined or at a high-impedance level depending on its design
- With power:
  - With a T0–T2 TAP.7
    - The circuitry functions as an input
    - The circuitry has an NB bias
  - With a T3–T5 TAP.7
    - The circuitry functions as an input and output
    - The circuitry has a PU bias

With a T3, wide T4, and T5 TAP.7s, the electrical and timing characteristics of the TDOC signal are the same as for the TDO signal in IEEE Std 1149.1 when the function of this signal is transferring TDO data. With the use of the JScan 3 Scan Format, the TDO Drive Policy inhibits the TDO drive in a number of cases with the JScan3 Scan Format. This prevents TDO drive conflicts in a Star-4 Scan Topology. The TDO Drive Policy is covered in Clause 13. With T3 and above TAP.7s, this signal becomes both an input and an output.

The TDO/TDOC Signal Characteristics are shown in Figure 12-8.



**Figure 12-8 — Test Data Output/signal bias relationships**

See Rule 12.9.2 g) for the TDOC Signal Characteristics when the signal function is programmable with T4 and above TAP.7s.

## 12.9.2 Specifications

### Rules

- Each subsequent specification in 12.9.2 shall apply to T0 and above TAP.7s.
- The TDO/TDOC Signal Characteristics specified in this subclause shall be independent of the I/O voltage level.
- The characteristics of the TDO and TDOC signals shall be governed by Table 12-8, provided the signal function is the TDO function.

**Table 12-8 — TDO/TDOC signal behavior relationships**

TAP signal logic is powered?	TAP.7 Class	Signal function	Signal bias
No	Any	Undefined	Undefined
Yes	T0-T2	Output only	NB
Yes	T3 and above	Input/Output	PU

- The characteristics of the TDOC signals when the TDO/TDOC signal is providing an auxiliary function shall be defined by the auxiliary function.

- e) The TDO(C) signal shall be driven by the TAP.7 only, provided any of the following are true:
  - 1) All of the following are true:
    - i) The TDO(C) signal provides the TDO function.
    - ii) The TDOC Drive Policy described in Clause 13 permits the drive of the TDO(C) signal.
  - 2) All of the following are true:
    - i) The TAP.7 Class is T4 and above.
    - ii) The signal function is the auxiliary signal function.
    - iii) The auxiliary signal function initiates drive of the signal.
    - iv) Rule 12.8.2 g) is satisfied.
- f) The asynchronous or synchronous detection of a Type-0–Type-3 Reset shall cause the TDO(C) signal to assume the default TDO(C) signal function.
- g) When the TDOC signal has programmable functionality and the default function is an auxiliary function, the signal shall remain at a high-impedance level after chip power-up until a function understanding the use of this signal confirms the signal function is an auxiliary function within the system (it could be the TDO function) and enables output drive.
- h) When the TDOC signal has programmable functionality and the default function is an auxiliary function, the signal's timing characteristics shall be defined by the auxiliary function subject to Rule 12.9.2 g).

## Recommendations

- i) Since the TDOC signals for many components may be connected in a Star-4 Scan Topology, care should be taken to ensure that the load presented by the TDOC signal is minimized.

## Permissions

- j) With a T4(W) and T5(W) TAP.7, the function of the TDOC signal may be either of the following:
  - 1) The equivalent of the function of the T3 TAP.7 TDOC signal.
  - 2) Programmable with the programming selecting either the equivalent of the function of the T3 TAP.7 TDOC signal or an auxiliary function.
- k) When the TDIC signal function is programmable with a T4(W) and T5(W) TAP.7, the default signal function created by a TAP.7 Controller reset may be either of the following:
  - 1) The equivalent of the function of the T3 TAP.7 TDOC signal.
  - 2) An auxiliary function.
- l) When the TDOC signal function is programmable with a T4(W) and T5(W) TAP.7, the auxiliary signal function may be the T3 TAP.7 TDOC signal function.

## 12.10 Offline-at-Start-up behavior

### 12.10.1 Description

As described in 10.3, the Offline-at-Start-up option is provided to allow the power-up and power-down of TAP.7 Controllers, while the DTS uses the TAP.7 connectivity to communicate with other TAP.7 Controllers. The operation of a TAP.7 Controller that is powered-up and powered-down while the TAP.7

connectivity is being used cannot therefore affect the use of this connectivity by the DTS and other TAP.7s in any way.

## 12.10.2 Specifications

### Rules

- a) Each subsequent specification in 12.10.2 shall apply to T0 and above TAP.7s.
- b) The TAP signal characteristics shall not interfere with the function and the use of these signals by the DTS and other TAP.7 Controllers when the Offline-at-Start-up option is used and any of the following are true:
  - 1) The state of the TAP.7 Controller power is as follows:
    - i) Powered.
    - ii) Un-powered.
    - iii) Being powered-up.
    - iv) Being powered-down.
  - 2) The state of the power to TAP.7 buffers connected to the chip signals is as follows:
    - i) Powered.
    - ii) Unpowered.
    - iii) Being powered-up.
    - iv) Being powered-down.

## 12.11 TAP connections

### 12.11.1 Description

The TCK(C) and TMS(C) signals are connected to the RSU when it is used. With the T1, T2, and T3 TAP.7s, the EPU's TAP interface signals are connected through the appropriate circuitry to create the TAP signal behavior specified in this and other clauses. With T4 and above TAP.7s, the APU's TAP interface may be wide or narrow. With T4 and above TAP.7s, the APU's TAP interface is connected to appropriate circuitry to create the TAP.7 signals required for the EPU of T4 and above TAP.7s.

The connectivity between the DTS and TAP.7s may be shared with other technologies provided these technologies are compatible with Online/Offline operation of the TAP.7 Controller described in 1.11.2 and in Clause 11. Controller may be shared as described in 1.11.2. In other words, this connectivity can be shared by IEEE 1149.7 components and other components that implement the RSU function. These components may be located on the same or different chips.

### 12.11.2 Specifications

#### Rules

- a) Each subsequent specification in 12.11.2 shall apply to T0 and above TAP.7s.
- b) The function of the TAP.7 signals shall not be used for any other purpose than defined by the standard.
- c) The TAP.7 signals shall be connected to the highest level TAPC in the TAPC hierarchy shown in Figure 7-5.

- d) The TAP.7 Controller state shall not be affected by signaling using the TCK(C), TMS(C), TDI(C), and TDO(C) signals other than the signaling that is specified as part of the following protocols:
  - 1) When the TAP.7 Controller is Online:
    - i) The Standard Protocol.
    - ii) The Advanced Protocol.
    - iii) The Control Protocol.
    - iv) Escapes.
    - v) Alerts provided they are implemented.
  - 2) When the TAP.7 Controller is Offline:
    - i) The Control Protocol.
    - ii) A Selection Escape.
    - ii) Escapes.
- e) For a component having an operating mode producing IEEE 1149.7-Specified Behavior and other operating modes that do not exhibit IEEE 1149.7-Specified Behavior, at least one mechanism shall exist to enable IEEE 1149.7-Specified Behavior.

NOTE—The rules in other subclauses of this standard apply only when the operating mode provides that is intended to provide IEEE 1149.7-Specified Behavior. Therefore, when a means is used to enable modes that provide either IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior is implemented, each rule should be considered to be prefaced by “When a mode providing 1149.7-Specified Behavior is enabled....” For example, Rule 12.11.2 b) should be read as stipulating that the function of the TAP.7 pins are defined completely by the standard and may not be used for any other purpose while IEEE 1149.7-Specified Behavior is enabled.

## **12.12 Applicability of this standard**

The applicability of this standard is governed by Rule 12.12.2 a).

### **12.12.1 Description**

### **12.12.2 Specifications**

#### **Rules**

- a) A component claiming compliance with this standard shall have at least one operating mode that provides IEEE 1149.7-Specified Behavior.

#### **Recommendations**

- b) A component chip's power-up operation should provide either of the following:
  - 1) IEEE 1149.7-Specified Behavior.
  - 2) IEEE 1149.7 Non-disruptive Behavior.

## 12.13 Recommendations for interoperability

### 12.13.1 Overview

While the test logic specified by this standard has been designed to be extensible (e.g., Custom Commands) to meet the particular needs of individual designers or companies, occasions may arise when it will be desirable to operate in modes that do not provide IEEE 1149.7-Specified Behavior. These modes may provide IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior.

The most common use of operating modes providing IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior involves two important areas:

- System security
- Manufacturing testing of chips

These two areas pose somewhat different challenges. With system security, the challenge is creating compatible operation with other components with security in place. This facilitates the debug and test of other nonsecure aspects of the system that do not have these security constraints. The need to debug in the secure environment should also be considered.

With manufacturing test, the challenge is choosing the manner in which these test modes are activated so the chances of doing so in a target system are minimized. It is also desirable that the method-change modes are known to the extent that these do not interact when these components are used in the same system.

This standard provides recommendations to address these challenges and facilitate interoperability in multi-chip systems. A chip architect may want to utilize the recommendations to maximize the usability of a chip in a multi-chip system without compromising security or other objectives. The concepts in 12.13.3 and 12.13.4 are only recommendations and are not mandatory requirements of this specification.

### 12.13.2 Power-up behavior

Because of the system uses of this standard, it is strongly recommended that a component provides either:

- IEEE 1149.7-Specified Behavior
- IEEE 1149.7-Non-disruptive Behavior

### 12.13.3 IEEE 1149.7-Non-disruptive behavior

#### 12.13.3.1 Description

Security and other policies followed by users of this specification may at times create IEEE 1149.7-Other Behavior. These policies can interfere with the operation of a system to a point that it is inoperable for both test and debug. This is an undesirable situation as it causes interoperability problems when the debug of nonsecure parts of this system is attempted.

Providing both security and IEEE 1149.7-Non-disruptive Behavior is both desirable and easy to accomplish. This is accomplished with an STL operating mode that causes every CLTAPC instruction to have the behavior of the IEEE 1149.1 *BYPASS* instruction. This provides both security and either IEEE 1149.1-Non-disruptive Behavior or IEEE 1149.7-Non-disruptive Behavior. This approach is strongly recommended. The STL Scan Path is never broken and is of no use, but it is operable to the extent it supports operation with other components with IEEE 1149.1-Specified Behavior and/or IEEE 1149.7-Specified Behavior.

### 12.13.3.2 Specifications

#### Rules

- a) Each subsequent rule in 12.13.3.2 applies to an Online T0 and above TAP.7.

#### Recommendations

- b) The *BYPASS* instruction created with the Instruction Register associated with the CLTAPC and the DR Scan Path selected by this *BYPASS* instruction and Instruction Register should be operable when viewed from the STL's TAP.

## 12.13.4 IEEE 1149.7-Other Behavior

### 12.13.4.1 Description

While the test logic specified by this standard has been designed to be extensible (e.g., Custom Commands) to meet the particular needs of individual designers or companies, occasions may arise when it will be desirable to enable IEEE 1149.7-Other Behavior. An example of this is the manufacturing test of chips using IEEE Std 1500 [B3], or similar, in modes that are suitable for use only during “standalone” component testing. This testing cannot be simultaneously operated with the debug-and-test functionality defined by this standard (targeting in-system test and applications debug). In this case, one or more modes of operation with IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior may be used to test a chip.

It is recommended that a component be placed in modes providing IEEE 1149.7-Other Behavior after the recommended IEEE 1149.7-Specified Behavior or IEEE 1149.7-Other Behavior power-up behavior is established. Some recommended ways for changing the mode of operation with the TCK(C) and TMS(C) signals are listed as follows:

- A custom TAP.7 Controller command
- A custom IEEE 1149.1 instruction
- A TAPC sequence using the *Test-Logic-Reset state*
- External signals

### 12.13.4.2 Specifications

#### Rules

- a) Each subsequent rule in 12.13.4.2 applies to an Online T0 and above TAP.7.

#### Recommendations

- b) When a chip's operation provides either IEEE 1149.7-Specified Behavior or IEEE 1149.7-Non-disruptive Behavior, modes producing IEEE 1149.7-Other Behavior should be enabled with only one of the following actions:
  - 1) The loading of an IEEE 1149.1 private instruction in the Instruction Register of the CLTAPC.
  - 2) The use of an IEEE 1149.7 private command.
  - 3) The setting of the IEEE 1149.1 boundary-scan enable signal(s) to state(s) that do not provide IEEE 1149.7-Specified Behavior.

- 4) While using the Standard Protocol and independently of the TAPC state, the processing of the TMS(C) input sequence, illustrated in Table 12-9.
- c) The bit pattern shown in Table 12-9 should have the following characteristics:

<b>MSB</b>	<b>LSB</b>
<b>20</b>	<b>00</b>
1 1100	1111 0011
1 C	F 3 D F

- d) The bit pattern shown in Table 12-9 should be used as the TMS value for sequential TAPC states while using the Standard Protocol with the LSB used for the first bit of the sequence progressing sequentially to the MSB.

**Table 12-9 — CLTAPC State Machine walk to provide IEEE 1149.7-Other Behavior**

<b>TMS(C) input sequence bit[n]</b>	<b>Resulting TAPC state sequence</b>
1	<i>unknown</i>
1	<i>Test-Logic-Reset</i>
0	<i>Run-Test/Idle</i>
1	<i>Select-DR-Scan</i>
1	<i>Select-IR-Scan</i>
1	<i>Test-Logic-Reset</i>
1	<i>Test-Logic-Reset</i>
0	<i>Run-Test/Idle</i>
0	<i>Run-Test/Idle</i>
1	<i>Select-DR-Scan</i>
1	<i>Select-IR-Scan</i>
1	<i>Test-Logic-Reset</i>
1	<i>Test-Logic-Reset</i>
0	<i>Run-Test/Idle</i>
0	<i>Run-Test/Idle</i>
1	<i>Select-DR-Scan</i>
1	<i>Select-IR-Scan</i>
1	<i>Test-Logic-Reset</i>

NOTE—Time progresses from top to bottom.

#### Permissions

- e) A component that has at least one operating mode that provides IEEE 1149.7-Specified Behavior may have other operating modes that provide either IEEE 1149.7-Non-disruptive Behavior or IEEE 1149.7-Other Behavior.

## 13. TDO(C) Signal Drive Policy

### 13.1 Introduction

This clause describes the TDO(C) Drive Policy for T0 and above TAP.7s. This drive policy supports operation with Series, Star-4, and Star-2 Scan Topologies. As the TAP.7 Controller Class increases, the TDO(C) Drive Policies change to support added capability. The TDO(C) Drive Policy determines TDOC drive characteristics when the TDIC and TDOC signals are not used for an auxiliary function with a T4 and above wide TAP.7. When these signals are used for an auxiliary function, this function determines the drive characteristics for these signals.

The TDO(C) Drive Policy is related to the scan format being used. The scan format specifies whether the TAP.7 behavior is compatible with operation in Series, Star-4, and Star-2 Scan Topologies as follows:

- Series Operation JScan0–JScan2 Scan Formats/JScan3 Scan Format with a T2 TAP.7
- Star-4 Operation JScan3 Scan Format with a T3 and above TAP.7
- Star-2 Operation MScan, OScan, and SScan Scan Formats

These terms are used in this document to describe operation with these scan formats.

This clause identifies the various factors affecting the TDO(C) Drive Policy, defines the TDO(C) Drive Policy for each TAP.7 Class and describes how the TAP.7 Controller determines whether the STL and EPU are the only Scan and Conditional Group Members. It also provides programming considerations.

The organization of the material within this clause is listed as follows:

- 13.2 TDO(C) signal drive types
- 13.3 Factors affecting TDO(C) Drive Policy
- 13.4 TDO(C) Drive Policy template
- 13.5 T0 TAP.7 TDO Drive Policy
- 13.6 T1 and T2 TAP.7 TDO Drive Policy
- 13.7 T3 and above TAP.7 TDOC Drive Policy
- 13.8 STL Group Membership
- 13.9 EPU Group Membership
- 13.10 Drive policy summary
- 13.11 Approach to implementing TDOC Drive Policy
- 13.12 Programming considerations

### 13.2 TDO(C) Signal Drive Types

#### 13.2.1 TDO(C) Signal Drive Types

Four types of TDO(C) drive are supported as described in 4.2.7.6:

- Single Drive with a value of the TAP.7 Controller's choosing
- Joint Drive with a logic 1 value

- Voting Drive with a logic 0 value only when the data desired is a logic 0
- Inhibited No drive

### 13.2.1.1 Single Drive

Single Drive is used to transfer conventional TDO data, with the value of this data determining the TDO(C) signal level.

### 13.2.1.2 Joint Drive

Joint Drive is used when a Star-4 Operation is specified to create a logic 1 TDOC signal level for certain *Shift-DR* states during CID allocation. It creates a logic 1 TDO(C) signal level. Multiple TAP.7 Controllers may simultaneously drive the TDOC signal as a logic 1, hence, the name Joint Drive.

### 13.2.1.3 Voting Drive

Voting Drive is used when Star-4 Operation is specified to create a logic 0 TDOC signal level for certain *Shift-DR* states during CID allocation. It creates a logic 0 TDOC signal level when the value of the TDO data to be output is a logic 0 and a high-impedance drive state when the value of the TDO data to be output is a logic 1. Multiple TAP.7 Controllers may simultaneously drive the TDOC signal as a logic 0. Any TAP.7 Controller driving a logic 0 creates a logic 0 TDOC signal value, hence, the name Voting Drive.

### 13.2.1.4 Inhibited Drive

Inhibited Drive is used to prevent TDO(C) drive during *Shift-DR* states in these following cases: once the Control Path is selected provided and during both Command Part One and Command Part Two. With Star-4 operation, it is used with Control Levels Three, Six, and Seven to prevent TDO(C) drive when there are none or more than one Scan Group Member while using the System Path, and none or more than one Conditional Group Member while using the EPU Path within a Star-4 Branch. It is also used to inhibit TDOC drive with wide T4 and above TAP.7s, provided this signal is not used for an auxiliary function.

## 13.2.2 Wire-ANDed TDOC data created with a combination of drives

The undirected method of CID allocation for the JScan3 Scan Format (see 9.9.3 and 20.9) utilizes the Wire-AND of TDO data as a means to arbitrate for the right to be allocated a CID. Four consecutive *Shift-DR* states (that may be separated with *Exit1-DR*, *Pause-DR*, and *Exit2-DR* states) transfer one bit of data supplied by the Enumerate EPU Path, with the TDO data value being the Wire-AND of TDOC data generated by one or more T3 and above TAP.7 Controllers. The transfer of TDO data utilizes Inhibit, Joint, Inhibit, and Voting Drives for this series of states. This sequence of drive types creates a precharge/discharge drive mechanism, provided the DTS provides PU drive of the TDOC signal (a pull-up resistor is connected to the signal within the DTS). The value of the TDO data value n is established during the bit period with Voting Drive. The Inhibit Drive periods are placed between the Joint Drive and Voting Drive bit periods to prevent drive conflicts at bit-period boundaries.

Inhibit Drive is used for any state between the *Shift-DR* states described above. Inhibit Drive is used for *Shift-DR* state[4n], where n is the index of the TDO data value (TDO value[n]) being transferred. Joint Drive establishes a logic 1 as the TDOC signal value for *Shift-DR* state[4n + 1]. Inhibit Drive is used for *Shift-DR* state[4n + 2], with the logic 1 TDOC signal value maintained by the DTS pull-up resistor during the bit period and any subsequent bit periods associated with non-shift states. Voting drive is used for *Shift-DR* state[4n + 3], with this bit period conveying the TDO data value (TDO value[n]). Any of the TAP.7 Controllers sharing a Star-4 Branch may set the TDOC signal value to a logic 0 during this bit period. When there are no logic 0 votes, the logic 1 value created by *Shift-DR* state[4n + 1] is maintained. The Inhibit Drive period associated with the following *Shift-DR* state begins with the generation of the TDO bit

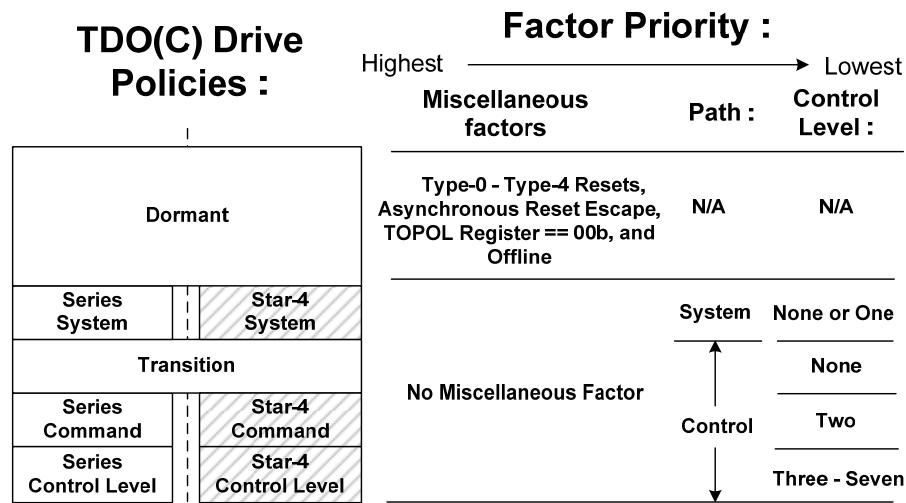
value[n + 1]. It merely prevents overlapping drive conflict between the Voting Drive period and the next Joint Drive period.

### 13.3 Factors affecting the TDO(C) Drive Policy

The TDO(C) Drive Policy is affected by the following factors:

- Miscellaneous:
  - Resets:
    - Type-0–Type-4 Resets
    - Asynchronous detection of a Reset Escape
  - TOPOL Register value
- Protocols:
  - Use of the Advanced Protocol
  - Use of the Control Protocol
  - Offline operation
- Use of the System Path:
  - With a scan format specifying operation with a Series Scan Topology
  - With a scan format specifying operation with a Star Scan Topology
- Use of the Control Path:
  - Before a control level is established
  - When the a control level greater than or equal to two is established

The relationship of the drive policies and these factors is shown in Figure 13-1.



**Figure 13-1 — TDO(C) Drive Policy/factor relationships**

## 13.4 TDO(C) Drive Policy template

### 13.4.1 General characteristics

The TDO(C) Drive Policy prevents drive conflicts with the TDO(C) signal:

- At start-up, independently of the type of operation and the technologies sharing the DTS connection
- Following start-up when determining the scan topologies sharing the DTS connection
- With Star-4 Operation

Certain conditions such as TAP.7 Controller resets, Offline operation, the use of the Advanced and Control Protocols, and a TOPOL Register value of zero also inhibit the drive of the TDO(C) signal, provided it is not used for an auxiliary function.

The TDOC Drive Policy provides for the creation of a control level following the *Select-DR-Scan* state without the drive of the TDO(C) signal. This means Control Level Two may be created at start-up (beginning with the *Test-Logic-Reset* state) or at other times (beginning with the *Select-DR-Scan* state) without the TDO(C) signal being driven. This makes commands available and permits the use of Scan Topology Training from start-up without creating drive conflicts in either a Star-4 or Star-2 Scan Topology.

With Series and Star-4 Operation, the drive of the TDO(C) signal is managed by the CLTAPC when the physical scan path is supplied by the STL and is managed by the ADTAPC otherwise. With Star-4 Operation, the drive of the TDO(C) signal is managed to ensure one of the following: 1) only one TAP.7 Controller drives the signal or 2) one or more TAP controller(s) drives the signal with the same logic level. With Star-2 Operation, the TDOC signal, if present, remains high impedance unless it is used as an auxiliary function.

### 13.4.2 TDOC drive enables

When the physical TAP.7 Scan Path is provided by the STL and the TDO(C) Drive Policy requires Single Drive, the CLTAPC controls the TDO(C) drive with the System TDO Output Enable (*sys\_tdo\_oe*). When the physical TAP.7 Scan Path is provided by the EPU and the TDO(C) Drive Policy requires Single Drive, the ADTAPC controls the TDO(C) drive with the *Shift-xR\_f* state. The drive of the TDO(C) signal is subject to drive prerequisites to prevent TDOC drive conflicts for Star-4 Operation otherwise. This matter is the subject of the majority of this clause.

### 13.4.3 TDO(C) Drive Policy components

The TDO(C) Drive Policy is a collection of eight TDO(C) Drive Policy components. These TDO(C) Drive Policy components are listed as follows. They are incrementally utilized and in some cases modified as the TAP.7 Class increases:

- Dormant              Miscellaneous conditions causing the TDOC signal to remain at a high-impedance state
- Transition            The Control Path is utilized with no control level
- Series:
  - System              The System Path is utilized with Series Operation
  - Command            Control Level Two is utilized with Series Operation
  - Control Level        A control level greater than two is utilized with Series Operation

- Star-4:
  - System      The System Path is utilized with Star-4 Operation
  - Command     Control Level Two is utilized with Star-4 Operation
  - Control Level    A control level greater than two is utilized with Star-4 Operation

The Dormant and Transition Drive Policy components are the same for Series and Star-4 Operation, while there are separate System, Command, and Control Level Components for Series and Star-4 Operation.

The combination of the Dormant, Transition, Series Command, and Star-4 Command Drive Policies provide for the use of Scan Topology Training. With T3 and above TAP.7s, these drive policy components ensure that the TDO(C) signal remains at a high-impedance level from start-up through completion of the Scan Topology Training Sequence.

#### **13.4.4 TAP.7 Class/TDO(C) Drive Policy component applicability**

The applicability of the TDO(C) Drive Policy components to the TAP.7 Classes is shown in Table 13-1.

**Table 13-1 — TAP.7 Class/TDO(C) Drive Policy component applicability**

TAP.7 Class	Dormant	Transition	System		Command		Control level	
			Series	Star-4	Series	Star-4	Series	Star-4
T0	x	—	—	—	—	—	--	—
T1	x	x	x	—	x	—	x	—
T2	x	x	x	x	x	—	x	—
T3	x	x	x	x	x	x	x	x
T4(W)	x	x	x	x	x	x	x	x
T5(W)	x	x	x	x	x	x	x	x

A detailed description of the drive policy for each TAP.7 Class begins with 13.5.

#### **13.4.5 Dormant TDO(C) Drive Policy**

The Dormant TDOC Drive Policy preempts other TDO(C) Drive Policies. With this policy, the miscellaneous conditions listed in 13.3 inhibit the drive of the TDO(C) signal. The descriptions of the TDO(C) Drive Policies that follow presume the drive of the TDO(C) signal is not inhibited by the Dormant TDOC Drive Policy.

#### **13.4.6 Transition TDO(C) Drive Policy**

The Transition Drive Policy inhibits TDO(C) drive when the Control Path is used and a control level has not been established (the ZBS count has not been locked).

#### **13.4.7 Series TDO(C) Drive Policy components**

##### **13.4.7.1 Series System**

The Series System Drive Policy manages TDO(C) drive with Series Operation when the System Path is used. The TDO(C) Signal Drive Policy is determined by whether the STL is a member of the Scan Group as follows:

- The CLTAPC directs the drive of the TDO(C) signal, provided the STL was a Scan Group Member the prior EPU TCK period.

- The ADTAPC directs the drive of the TDO(C) signal during the *Shift-xR\_f* state, provided the STL is not a Scan Group Member.

The Scan Group Membership is delayed one EPU TCK period to determine the Series System Drive Policy. The need for this approach is related to harmonizing the Drive Policies of the Standard and Advanced Protocols, and it is described further in 14.7.1.3. This means that changes in the selection of the CLTAPC of TAP.7 Controllers in the same branch that occur during the EPU TCK period n affect the TDO(C) Bit Drive Policy during EPU TCK period n + 1. In other words, changes in the system TDO(C) Drive Policy lag CLTAPC selection state changes by one EPU TCK period. This is viewed as having no consequences as selection state changes occur in states where the TDO signal is not driven. The worst case is the TDOC Drive Policy changing at the *Run-Test/Idle>Select-DR-Scan* state boundary for selection state changes not initiated by SSDs, and upon exiting the *Select-DR-Scan*, *Exit2-IR*, or *Exit2-DR* states when changes are initiated by SSDs and the SSD State Machine state changes at same time the state supporting SSDs is exited. None of these worst-case scenarios are problematic.

#### **13.4.7.2 Series Command**

The Series Command Drive Policy manages TDO(C) drive with Series Operation when the control level is two. The TDOC signal is driven only during the CR Scans of SCNB and SCNS Commands. This permits the use of the TDO(C) signal for Scan Topology Training during Command Part One and Command Part Two, with the DTS driving the TDO(C) signal as required by this training during these periods.

#### **13.4.7.3 Series Control Level**

The Series Control Level Drive Policy manages TDO(C) drive with Series Operation when the control level is greater than two.

#### **13.4.8 Star-4 TDO(C) Drive Policies**

Star-4 Operation adds prerequisites for the drive of the TDOC signal to those used with Series Drive Policy components. These prerequisites are described as follows.

##### **13.4.8.1 Star-4 System**

The Star-4 System Drive Policy manages TDO(C) drive with Star-4 Operation when the System Path is used. The CLTAPC determines when the TDOC signal is driven (with the TDO output enable signal), provided the TAP.7 Controller has determined that its STL is the only Scan Group Member the last TCK(C) clock period. Selection and deselection of CLTAPCs during EPU Test Clock period n affect the TDO(C) Bit Drive Policy during the EPU Test Clock period n + 1, just as with the Series System Drive Policy.

##### **13.4.8.2 Star-4 Command**

The Star-4 System Drive Policy manages TDO(C) drive with Star-4 Operation when the System Path is used. The CLTAPC determines when the TDOC signal is driven (with the TDO output enable signal), provided the TAP.7 Controller has determined that its STL is the only Scan Group Member the last TCK(C) clock period. Selection and deselection of CLTAPCs during EPU Test Clock period n affect the TDO(C) Bit Drive Policy during the EPU Test Clock period n + 1, just as with the Series System Drive Policy.

With this drive policy, the TDOC signal is driven during the *Shift-DR\_f* states of the CR Scans of the:

- SCNB and SCNS Commands, provided the TAP.7 Controller has determined that its EPU is the only Conditional Group Member

- CIDA Command when the characteristics of the CIDA Command described in 20.8 require the TDOC signal be driven

#### **13.4.8.3 Star-4 control level**

The Star-4 Control Level Drive Policy manages the drive of the TDO(C) signal with Star-4 Operation when the control level is greater than two. The TDOC signal is driven during the *Shift-DR\_f* state, provided the control level is either four or five and the TAP.7 Controller has determined that its EPU is the only Conditional Group Member.

There is no drive of the TDO(C) signal when the control level is as follows:

- Three, preserving the use of this control level for future expansion
- Six, as this control level is used for Deselection Alerts
- Seven, as this control level is reserved for DTS use

The TDOC Drive Policy and other behaviors provide for adding behavior to control levels in future revisions of the specification.

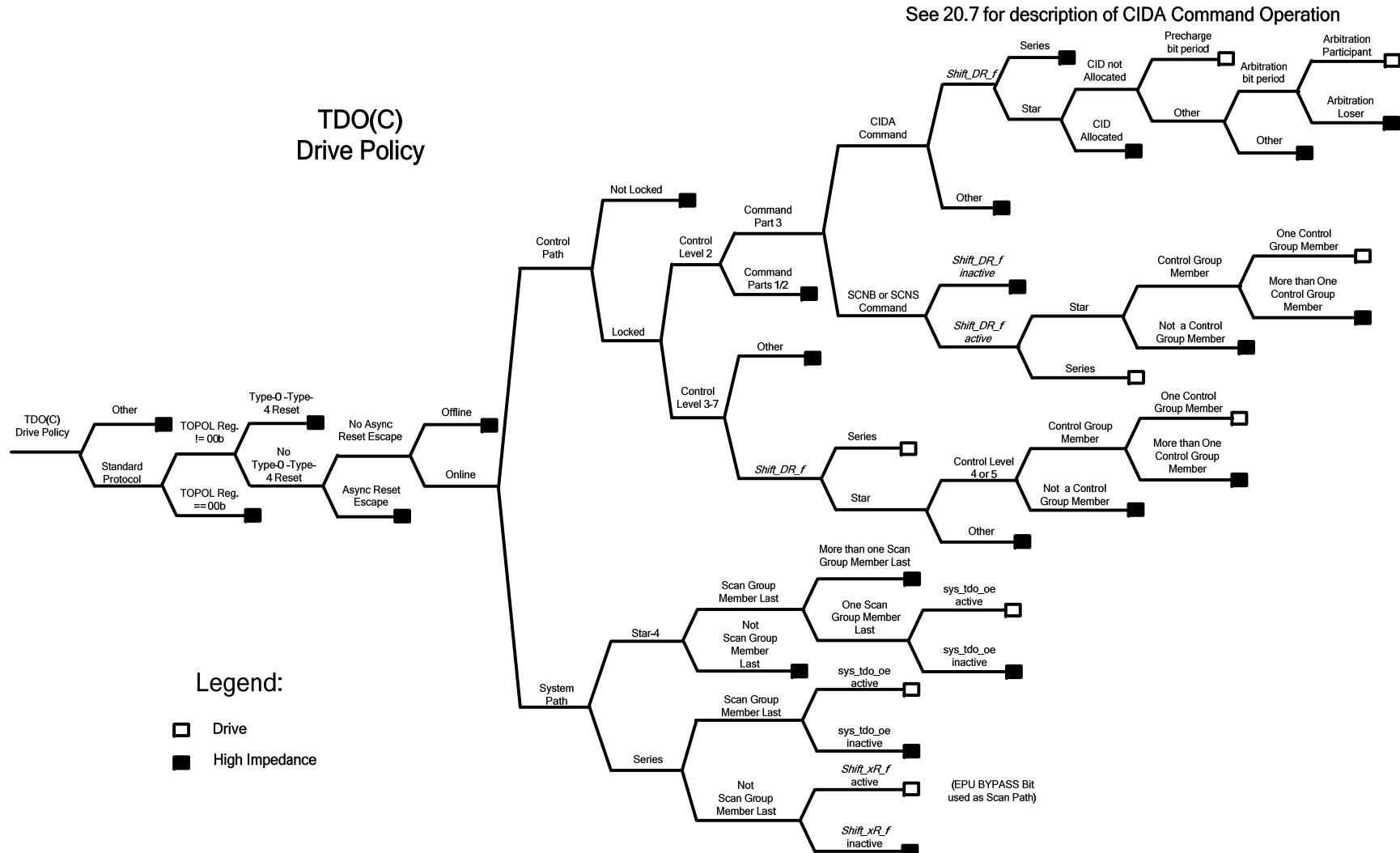
#### **13.4.9 Hierarchical and flat views of the TDO(C) Drive Policy**

A hierarchical view of the TDO(C) Drive Policies is shown in Figure 13-2.

Policy	Use Case	In Idle Group	Physical scan path provided by:	TDO(C) output enable provided by:	Conditions qualifying TDO(C) drive with:			
					Series Operation	Star-4 Operation	Star-2 Operation	
Dormant	If: any of the events listed to the right		TDOC pin is high impedance Offline, Type-0 - Type-4 Reset, Asynchronous Reset Escape Detection, Use of the Control and Advanced Protocols, TOPOL == 00			Type-0 - Type-4 Reset or Async Reset Escape		
System	Else If: $zbs\_cnt \leq 1$   suspend == 1	No	STL	CLTAPC	None	Only Scan Group Member the last EPU TCK(C) period	May be driven provided the TDOC signal provides a function other than the T3 TAP.7 TDOC, the function enables TDOC drive, and there is no Type-0 - Type-4 Reset or Async Reset Escape Detection	
		Yes	EPU (Bypass Bit)	ADTAPC				
Transition	Else If: $LOCKED == 0$		TDOC pin is high impedance			CIDA Command Arbitration Paradigm	Must be only Conditional Group Member	
Command	Else If: CONTROL LEVEL == 2	Don't Care	CIDA - Enumerate	ADTAPC	None	CIDA Command Arbitration Paradigm		
			SCNB - Bit					
			SCNS - String					
			Other - Bypass					
Control Level	Else: CONTROL LEVEL > 2	Don't Care	Ctl. Lvl. 3 - Bypass	ADTAPC	None	Must be only Conditional Group Member	Must be only Conditional Group Member	
			Ctl. Lvl. 4 or 5: CGM = 0 - Bypass CGM = 1 - AUX					
			Ctl. Lvl. 6 or 7: Bypass					

**Figure 13-2 — Hierarchical view of the TDO(C) Drive Policy**

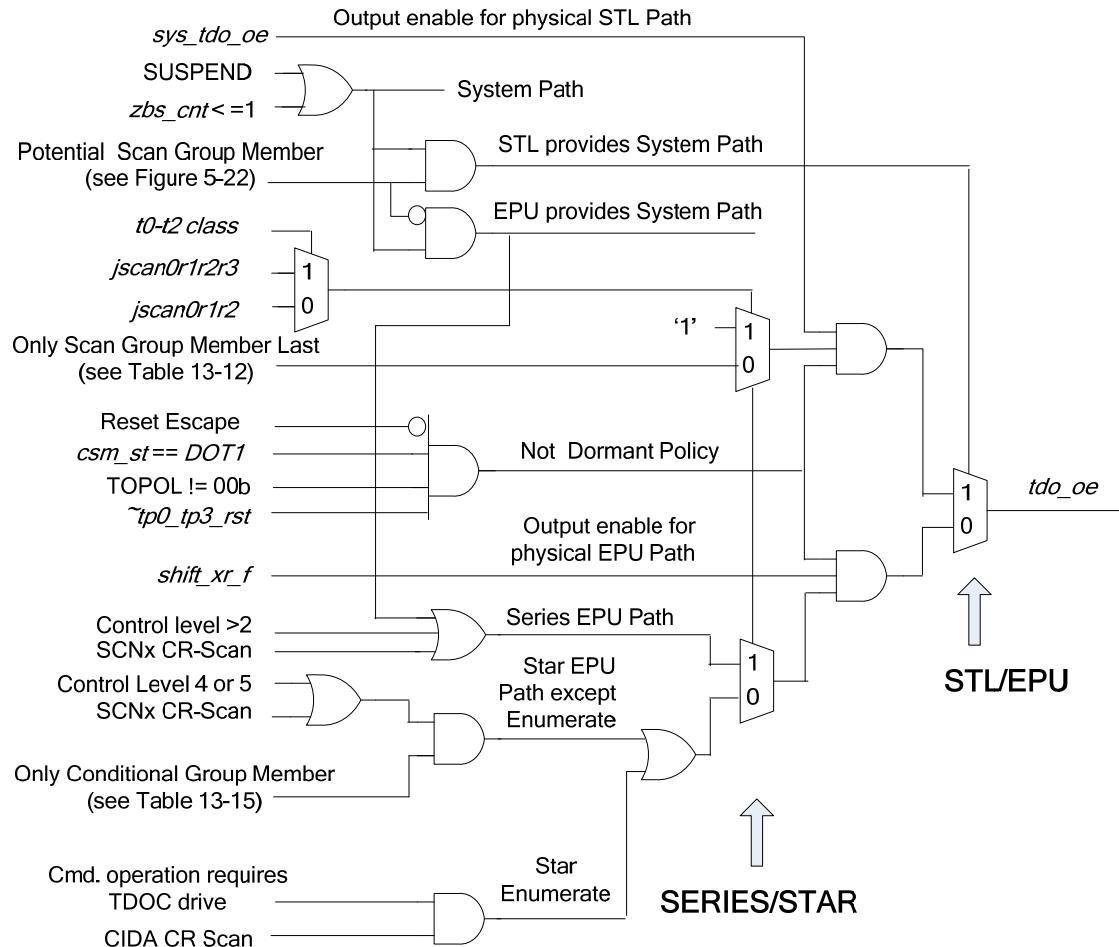
A flattened view of the TDO(C) Drive Policy is shown in Figure 13-3.



**Figure 13-3 — Flattened view of the TDO(C) Drive Policy**

### 13.4.10 Conceptual diagram of the TDO(C) Drive Policy

A conceptual diagram of the TDO(C) Drive Policy is shown in Figure 13-4. This diagram encompasses the TDO(C) Drive Policy for all TAP.7 Classes. The set or subset of this function required for each TAP.7 Class is described in the following subclauses.



**Figure 13-4 — Conceptual view of TDOC Drive Policy**

Certain signals shown in Figure 13-4 are tied off depending on the TAP.7 Class and whether the RSU is implemented. The signal tie-offs for these TAP.7 configurations are shown in Table 13-2.

**Table 13-2 — Defaults for unimplemented functions when determining TDOC Drive Policy**

TAP.7 Class	ZBS count	Control level	SUSPEND	Potential Scan Group Member	Only Scan Group Member Last	Scan format	TOPOL Register	Only Conditional Group Member	All commands
T0	0	0	0	1	1	JSca n0	01	0	0
T1	—	—	—	1	1	JSca n0	01	—	—
T2	—	—	—	—	—	—	01	—	—
T3–T5	—	—	—	—	—	—	—	—	—

When a T0–T2 TAP.7 is implemented without the RSU, the Control State Machine state is considered *STD*.

## 13.5 T0 TAP.7 TDOC Drive Policy

### 13.5.1 Description

With the T0 TAP.7, the CLTAPC enables the drive of the TDO signal. It is expected to do so only in the *Shift-DR* and *Shift-IR* states following the *Test-Logic-Reset* state, as per the IEEE Std 1149.1. An STL private instruction may be used to override this drive policy to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7 Other Behavior. The T0 TAP.7 TDO Drive Policy is illustrated in Figure 13-5.

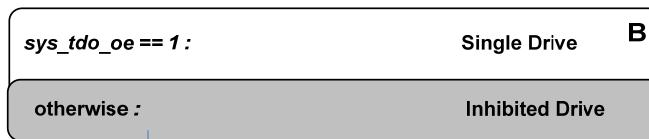
If:

Dormant  
Drive Policy



Else :

System  
Drive Policy



**Legend:**



*sys\_tdo\_oe : the STL activates drive of the TDO signal with this TDO output enable signal*

**Figure 13-5 — T0 TAP.7 TDO Drive Policy**

### 13.5.2 Specifications

#### Rules

- a) Each subsequent specification in 13.5.2 shall apply to a T0 and above TAP.7.
- b) The STL's use of the TDI and TMS information shall not be affected by the operation of the TDO(C) signal.

- c) The TDO signal drive shall be inhibited with the Dormant Drive Policy, provided any of the following are true:
  - 1) A Type-0, Type-1, Type-2, or Type-3 Reset is active, provided these resets are implemented.
  - 2) The ADTAPC is Offline, provided the RSU is implemented.
  - 3) The asynchronous detection of a Reset Escape is active, provided the RSU is implemented.
- d) Following the *Test-Logic-Reset* state, a T0 TAP.7 shall drive the TDO signal only in the *Shift-DR* and *Shift-IR* states, as per the IEEE Std 1149.1, provided the drive of the TDO(C) signal is not inhibited per Rule 13.5.2 c).

NOTE—The *Test-Logic-Reset* state creating the Type-4 Reset does not permit TDO(C) drive per IEEE Std 1149.1.

## Permissions

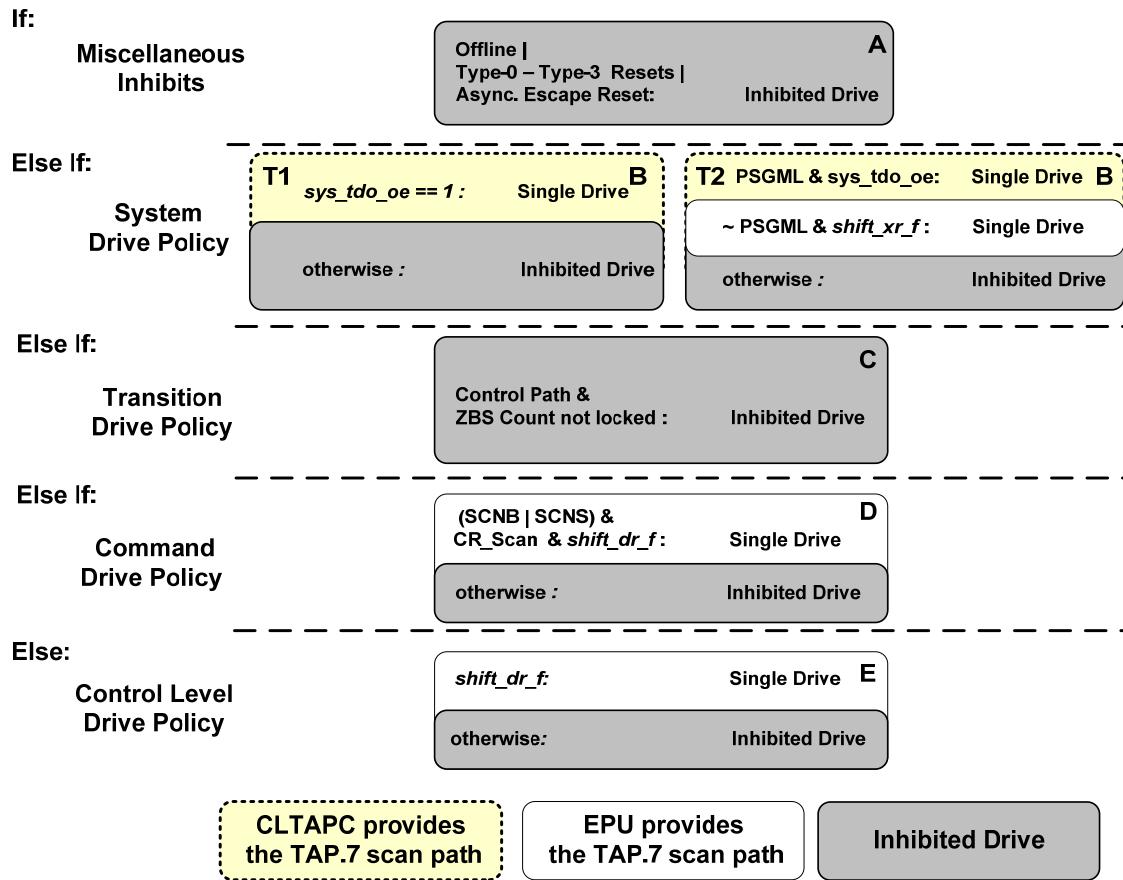
- e) Following the *Test-Logic-Reset* state, a private instruction may be used to alter the drive policy specified by Rule 13.5.2 a) through Rule 13.5.2 d) to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior.

## 13.6 T1 and T2 TAP.7 TDOC Drive Policy

### 13.6.1 Description

A T1 TAP.7 inherits the T0 TAP.7 Drive Policy, while the T2 TAP.7 inherits the T1 TAP.7 Drive Policy. Following the *Test-Logic-Reset* state, a TAP.7 Controller private command or STL private instruction may override the drive policy for T1 and T2 TAP.7s to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7 Other Behavior.

The T1 and T2 TAP.7s extend the T0 TAP.7 Drive Policy as shown in Figure 13-6. T1 and T2 TAP.7s utilize ZBS counts, control levels, and the System and Control Paths. A T2 TAP.7 provides for the selection and deselection of STLs in the *Run-Test/Idle* state (creation of Scan and Idle Groups). These features, along with the use of the JScan0–JScan3 Scan Formats, affect the TDO Drive Policy.



## Legend:

$\text{sys\_tdo\_oe}$ : the STL activates drive of the TDO signal with this TDO output enable signal  
 $\text{shift\_dr\_f}$ : the EPU activates drive of the TDO signal with this TAPC state signal  
 PSGML: Potential Scan Group Member the last TCK period

Figure 13-6 — T1-T2 TAP.7 TDO Drive Policy

### 13.6.2 Specifications

#### Rules

- a) Each subsequent specification in 13.6.2 shall apply when all of the following are true:
  - 1) The drive of the TDO(C) signal is not inhibited per Rule 13.5.2 c).
  - 2) The TAP.7 Class is T1-T2.
- b) When the STL provides the physical TAP.7 Scan Path, the drive of the TDO(C) signal shall be enabled by the CLTAPC [with the system TDO output enable ( $\text{sys\_tdo\_oe}$ )].
- c) When the EPU provides the physical TAP.7 Scan Path, the TDO(C) signal shall be driven only in the *Shift-DR\_f* and *Shift-IR\_f* states, provided any of the following are true:
  - 1) The System Path is selected (the CLTAPC is deselected).
  - 2) The CR Scan of either an SCNB or SCNS Command is ongoing.
  - 3) The control level is greater than two.

## Permissions

- d) Following the *Test-Logic-Reset* state, a private instruction and/or private command may be used to alter the drive policy specified by Rule 13.6.2 a) through Rule 13.6.2 c) to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior.

## 13.7 T3 and above TAP.7 TDOC Drive Policy

### 13.7.1 Description

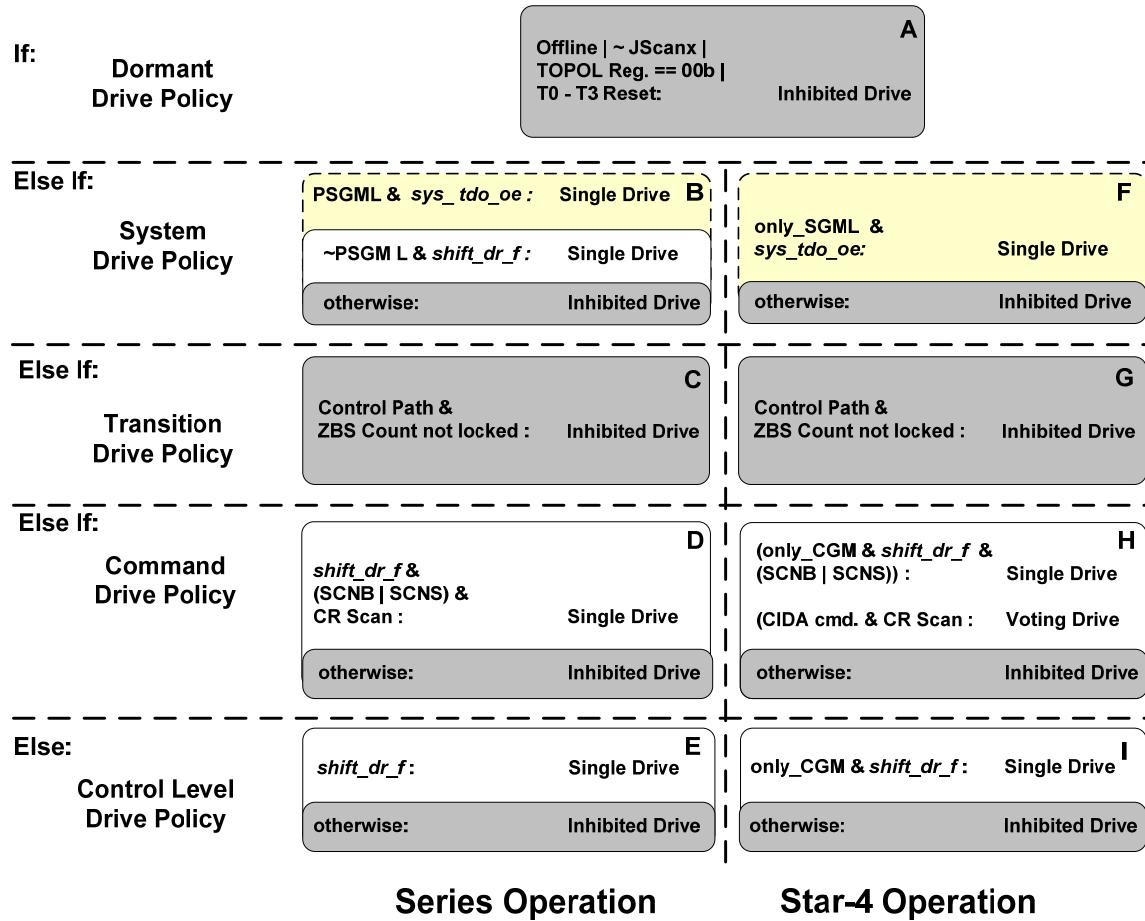
The T3 and above TAP.7s inherits the T1-T2 TAP.7 TDO Drive Policy and extends it as shown in Figure 13-7. A TAP.7 Controller private command or STL private instruction may override this drive policy to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7 Other Behavior. With these TAP.7 Classes, a TOPOL Register value of 00b inhibits the drive of the TDOC signal.

When Series Operation is specified with the JScan0–JScan2 Scan formats, the TDOC Drive Policy for T3 and above TAP.7s is the same as with a T2 TAP.7. With T3 and above TAP.7s, the JScan3 Scan Format specified Star-4 Operation while it specifies Series Operation with a T2 TAP.7. The TDO(C) Drive Policy specified with the use of the JScan3 Scan Format with a T3 and above TAP.7 is therefore dramatically different than with a T2 TAP.7, as functionality that prevents TDOC drive conflicts in a Star-4 Scan Topology is activated. The DTS may use this difference in operation to detect the erroneous use of a T0–T2 TAP.7 without an RSU in a Scan Topology with multiple branches by observing the behavior of the TDOC signal with the JScan3 Scan Format (it will be driven at times it is expected to be high impedance).

Other than with the CIDA CR Scan, the drive of the TDOC signal is permitted with a T3 and above TAP.7 provided the TAP.7 Controller has determined that:

- The System Path is selected and the TAP.7 Controller has determined its STL is the only Scan Group Member
- The Control Path is selected and the TAP.7 Controller has determined its EPU is the only Scan Group Member

The method used to determine whether the STL is the only Scan Group Member is described in 13.8.4. The method used to determine whether the EPU is the only Conditional Group Member is described in 13.9.1.3



### Legend:

CLTAPC provides the TAP.7 scan path	EPU provides the TAP.7 scan path	Inhibited Drive
only_SGML:	Only Scan Group Member the last EPU TCK period	
only_CGM:	Only Conditional Group Member	

Figure 13-7 — T3, T4(W), and T5(W) TAP.7 TDOC Drive Policies

### 13.7.2 Specifications

#### Rules

- Each subsequent specification in 13.7.2 shall only apply to all of the following TAP.7s:
  - T3.
  - T4 (W) and T5 (W).
- The TAP.7 Controller shall be considered operating in a manner compatible with a Star-4 Scan Topology when all of the following are true:
  - The TAP.7 Controller Class is T3 and above.
  - The Scan Format Register specifies the use of the JScan3 Scan Format.
  - The ADTAPC's Scan Selection State is Online.
- The drive of the TDO(C) signal shall be inhibited by a TOPOL Register value of 00b.

- d) Each subsequent specification in 13.7.2 shall only apply to a TAP.7 Controller's operation when all of the following are true:
  - 1) The drive of the TDO(C) signal is not inhibited per Rule 13.5.2 c).
  - 2) The drive of the TDO(C) signal is not inhibited per Rule 13.7.2 c).
  - 3) The TDO signal provides the TDO function (does not provide an auxiliary function with a T4 and T5 TAP.7).
- e) When the Scan Format Register specifies the use of the JScan0, JScan1, or JScan2 Scan Format, the drive of the TDO(C) signal shall be governed by Rules 13.6.2 b) and 13.6.2 c).
- f) Each subsequent specification in 13.7.2 shall only apply to a TAP.7 Controller operation when the Scan Format Register specifies the use of the JScan3 Scan Format.
- g) When the System Path is selected, the TDO(C) signal shall be driven, provided any of the following are true:
  - 1) All of the following are true:
    - i) The STL provides the physical TAP.7 Scan Path.
    - ii) The CLTAPC enables the drive of the TDO signal (with the system TDO output enable (*sys\_tdo\_oe*)).
    - iii) The TAP.7 Controller has determined its CLTAPC is the only Scan Group Member (see Table 13-10).
  - 2) All of the following are true:
    - i) The EPU provides the physical TAP.7 Scan Path.
    - ii) The TAPC state is *Shift-DR\_f* or *Shift-IR\_f*.
    - iii) The TAP.7 Controller has determined its STL is the only Scan Group Member.
- h) When the Control Path is selected, the TDO(C) signal shall be driven, provided all of the following are true:
  - 1) The TAPC state is *Shift-DR\_f* or *Shift-IR\_f*.
  - 2) The TAP.7 Controller has determined its EPU is the only Conditional Group Member (see Table 13-13).
  - 3) Any of the following are true:
    - i) The CR Scan of either an SCNB or SCNS Command is ongoing.
    - ii) The CR Scan of a CIDA Command is ongoing, and the characteristics of a CIDA Command require the TDOC signal be driven.
    - iii) The control level is either four or five.

## Permissions

- i) Following the *Test-Logic-Reset* state, a private instruction and/or private command may be used to alter the drive policy specified by Rule 13.7.2 a) through Rule 13.7.2 h) to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior.

## 13.8 STL Group Membership

### 13.8.1 Tracking the Group Membership of STLs

The TAP.7 Controller determines whether its STL is the only Scan Group Member by tracking commands, SSDs, and the TAPC state progressions that create and vacate the following across all Online TAP.7 Controllers sharing the DTS connection:

- Idle Group Membership
- Pause-IR Group Membership
- Pause-DR Group Membership

Recall that the STL is a Scan Group Member when it is not a member of any of the above mentioned STL groups. There can never be both Pause-IR Group Members and Pause-DR Group Members.

Idle Group Membership has the highest priority. An STL may become a Pause-xR Group Member only when it is not an Idle Group Member. An STL may become a Scan Group Member only when it is neither an Idle Group Member nor a Pause-xR Group Member.

### 13.8.2 STL Group Membership changes

Placing a TAP.7 Controller Offline freezes a TAP.7 Controller's STL Group Membership. With an Online TAP.7 controller, group membership changes occur in the *Test-Logic-Reset\_f* state, in the *Pause-IR\_f* state provided there are no Pause-DR group members, in the *Pause-DR\_f* state provided there are no Pause-IR group members, and the *Run-Test/Idle\_f* state provided there are no Pause-IR or Pause-DR Group Members.

#### 13.8.2.1 Group membership changes with the *Test-Logic-Reset* state

With the *Test-Logic-Reset\_f* state, membership in the groups listed below is established as follows:

- Idle When the default scan format is JScan1
- Scan When the default scan format is JScan0

#### 13.8.2.2 Group membership changes with the *Run-Test/Idle* state

With the *Run-Test/Idle\_f* state, the group memberships listed below are established as follows when there are no Pause-xR Group Members:

- Idle When the STL is an Idle Group Candidate
- Scan When the STL is not an Idle Group Candidate

Referring to Figure 5-22, the STL becomes an Idle Group Candidate when any of the following occur:

- The Control Path is used.
- The JScan1 Scan Format is used.
- The value of the Scan Group Candidate Register is a logic 0 when the use of a scan format other than JScan0 or JScan1 is used.

Both commands and executed SSDs associated with the *Run-Test/Idle\_f* state may change the Scan Group Candidacy Register value (see 13.8.3.2 for a brief description of SSDs). The Scan Group Candidacy

Register may be changed independently of whether the STL is a Scan or Idle Group Member. Idle Group Candidacy and Membership is described in 19.8.

### 13.8.2.3 Group membership changes with the *Pause-IR* state

With the *Pause\_IR\_f* state, the membership in the groups listed as follows is established by an SSD associated with the *Pause-IR* state, provided there no Pause-DR Group Members:

- Pause-IR:
  - A Deselect All SSD creates Pause-IR Group Membership for all STLs that are a Scan Group Member (all Scan Group Members become Pause-IR Group Members).
  - A Select One SSD creates Pause-IR Group Membership for all STLs that are both a Scan Group Member and the target of the SSD. There is either none or one Scan Group Member following the execution of this SSD.
- Scan:
  - A Select All SSD creates Scan Group Membership for all STLs that are a Pause-IR Group Members.
  - An Select One SSD creates Scan Group Membership for the STL that is explicitly referenced by the SSD.

### 13.8.2.4 Group membership changes with the *Pause-DR* state

With the *Pause\_DR\_f* state, the group memberships listed as follows are established by an SSD associated with the *Pause-DR* state, provided there no Pause-IR Group Members:

- Pause-DR:
  - A Deselect All SSD creates Pause-DR Group Membership for all STLs that are a Scan Group Member (all Scan Group Members become Pause-IR Group Members).
  - A Select One SSD creates Pause-DR Group Membership for all STLs that are both a Scan Group Member and not the target of the SSD. There is either none or one Scan Group Member following the execution of this SSD.
- Scan:
  - A Select All SSD creates Scan Group Membership for all STLs that are a Pause-DR Group Members (all Pause-DR Group Members become Scan Group Members).
  - A Select One SSD creates Scan Group Membership for the STL that is the target of the SSD.

## 13.8.3 Commands/SSDs affecting group Scan Group Candidacy and Membership

### 13.8.3.1 Commands affecting Scan Group Candidacy

When a TAP.7 Controller is Online, commands manage the Scan Group Candidacy Register and as follows:

- CMD(STMC, SGC = 0) Set the SGC Register value to a logic 0
- CMD(STMC, SGC = 1) Set the SGC Register value to a logic 1
- CMD(SCNB, SGC = x) Set the SGC Register value to value x
- CMD (MSC, M[CID]) m==0: Set the SGC Register value of the targeted controller to a logic 1. Set the SGC Register value of a nontargeted

controller to a logic 0. This includes a TAP.7 Controller that is invalid; the CIDI Register value is a logic 1.

- CMD (MSC, M[CID]) m==1: Set the SGC Register value of the targeted controller to a logic 1. There is no change in the SGC Register value of nontargeted TAP.7 Controller unless the CID is invalid.

With the last two commands listed above, the command identifies the targeted TAP.7 Controller using its CID.

### 13.8.3.2 SSDs affecting Scan Group Candidacy and Membership

Scan Selection Directives are described in detail in 20.10. This description covers their names, function, state machines tracking their execution, and their effects on STL Group membership.

#### 13.8.3.2.1 SSDs associated with the *Run-Test/Idle* state

Executed SSDs associated with the *Run-Test/Idle* state affect the Scan Group Candidacy Register value. These SSDs change the Scan Group Candidacy Register value of an Online TAP.7 Controller as follows:

- Set the Scan Group Candidacy Register value to a logic 0:
  - Deselect All Vacate Scan Group Candidacy (no candidates) for all STLs
- Set the Scan Group Candidacy Register value to a logic 1:
  - Select One Create Scan Group Candidacy for an STL that is the target of the SSD.  
Do not change the Scan Group Candidacy of other STLs
  - Select\_All Create Scan Group Candidacy for an STL

Because these SSDs occur while the TAPC state is *Run-Test/Idle*, they immediately affect Scan Group Membership as they change the SGC Register value and this value changes Scan Group Membership when the TAPC state is *Run-Test/Idle*.

#### 13.8.3.2.2 SSDs associated with the *Pause-IR* state

Executed SSDs associated with the *Pause-IR* state affect Scan Group and Pause-IR Group Membership as follows:

- Create no Scan Group Members:
  - Deselect All Convert all Scan Group Memberships to Pause-IR Group Memberships.
- Create a maximum of one Scan Group Member:
  - Select One Create Scan Group Membership for the STL that is the target of the SSD. Create Pause-IR Group Membership for STLs that are not the Target of the SSD, provided they are not Idle Group Members.
- Create none, one, or more than one Scan Group Member:
  - Select All Convert all Pause-IR Group Members to Scan Group Members.

#### 13.8.3.2.3 SSDs associated with the *Pause-DR* state

Executed SSDs associated with the *Pause-DR* state affect Scan Group and Pause-DR Group Membership as follows:

- Create no Scan Group Members:
  - Deselect All Convert all Scan Group Memberships to Pause-DR Group Memberships.
  - Create a maximum of one Scan Group Member:
    - Select One Create Scan Group Membership for the STL that is the target of the SSD. Create Pause-DR Group Membership for STLs that are not the Target of the SSD, provided they are not Idle Group Members.
  - Create none, one, or more than one Scan Group Member:
    - Select All Convert all Pause-DR Group Members to Scan Group Members.

### 13.8.4 Only Scan Group Member determination

#### 13.8.4.1 Criteria

The TAP.7 Controller considers its STL the only Scan Group Member when it determines all the following are true:

- Its STL is not a member of the Idle, Pause-IR, and Pause-DR Groups
- Any of the following are true:
  - There are Pause-IR Group Members
  - There are Pause-DR Group Members
  - All STLs except one are Idle Group Members

The Star-4 System TDOC Drive Policy is determined by the only Scan Group Member determination delayed one EPU TCK period.

#### 13.8.4.2 Method and information used to make determination

Each Online TAP.7 Controller sharing the DTS connection tracks commands affecting the Scan Group Candidate Register and SSDs to determine whether the TAP.7 Controller is the only Scan Group Member. It determines the number of Scan Group Members and whether it is a Scan Group Member, combining these to determine whether it is the only Scan Group Member. This is a multi-step process as outlined below.

##### 13.8.4.2.1 Idle Group Membership and membership counts

Idle Group Membership is determined first. Idle Group Membership is changed when the *Run-Test/Idle\_f* state is reached, provided there are no Pause-xR Group Members (the SSD State Machine *SSD\_SA* state indicates this). Idle Group Membership is created when the STL is an Idle Group Candidate and a change in Idle Group Membership is allowed. Idle Group Membership is vacated when the STL is not an Idle Group Candidate and a change in Idle Group Membership is allowed. The STL is an Idle Group Candidate when any of the following are true:

- The Scan Format is JScan1.
- The Scan Format is not JScan0 and the Scan Group Candidate Register value is a logic 0.
- The Control Path is selected.

The STL is a Scan Group Candidate when it is not an Idle Group Candidate.

A count of the number of Scan Group Candidates is tracked with a Scan Group Candidate Count (SGCC) by tracking the commands and SSDs that change the Scan Group Candidacy Register, independently of the command's target. The count has three states:

- |                              |                |                                    |
|------------------------------|----------------|------------------------------------|
| — No Scan Group Candidates   | or conversely, | All Idle Group Candidates          |
| — One Scan Group Candidate   | or conversely, | All but one Idle Group Candidates  |
| — Many Scan Group Candidates | or conversely, | Many are not Idle Group Candidates |

The operation of the Scan Group Candidate Count is described in 13.8.5.

#### **13.8.4.2.2 Pause-xR Group Membership and membership counts**

Pause-xR Group Membership is tracked independently of whether the STL is already an Idle Group Membership. This is acceptable as Scan Group Membership is the absence of Pause-xR and Idle Group Memberships. The SSD State Machine performs the tracking function and determines whether the STL may be a Pause-xR Group member. This state machine tracks the following:

- The last SSD executed (as SSD execution may be inhibited in some cases)
- The TAPC state associated with the last SSD executed
- Whether the TAP.7 Controller was the target of an SOC or SOT SSD when the SSD executes and is associated with the Pause-xR state

The SSD State Machine state also provides the information necessary to determine the number of Potential Pause-xR Group Members. The number of Potential Pause-xR Group Members is recorded as follows:

- None after a Deselect All SSD executed in association with the *Pause-xR* state
- One after a Select One SSD executed in association with the *Pause-xR* state
- All after a Select All SSD executed in association with the *Pause-xR* state

This information is combined.

#### **13.8.4.2.3 Scan Group Membership and membership counts**

The STL is determined to be a Scan Group Member when it is neither an Idle Group Member nor a Potential Pause-xR Group Member. The number of Scan Group Members is determined as follows:

- None:
  - All STLs are either Potential Pause-xR Group Members or Idle Group Members.
- One—Any of the following:
  - There are no STLs that are Potential Pause-xR Group Members and there is one STL that is a Potential Scan Group Member.
  - All STLs except one Potential Pause-xR Group Members and one or more Potential Scan Group Members.
- Many—All of the following:
  - There are no STLs that are Potential Pause-xR Group Members.
  - There is more than one STL that is a Potential Scan Group Member.

#### 13.8.4.2.4 Information recorded

The number STLs that are Scan Group Members and whether this STL is the only Scan Group Member is derived from information recorded by two tracking mechanisms within the EPU. The first tracking mechanism determines whether the STL is an Idle Group Member. The second tracking mechanism determines whether the STL would be a Pause-DR and Pause-IR Group Member (a Potential Group Member) provided it is not an Idle Group Member. The TAP.7 Controller records a number of pieces of pieces of information to make these determinations. This information is listed below. The TAP.7 Controller records the information listed above as follows:

- Potential Scan Group Member Last (PSGML)—A flag (*psgml\_r*—see Figure 5-22) records whether the STL was a Scan Group Member when the Idle Group Membership was updated. Note that this flag represents Potential Scan Group Membership delayed by one EPU TCK period.
- Scan Group Candidate Count (SGCC)—A state machine that records the count of the number of STLs that the TAP.7 Controller believes are not Idle Group Candidates.
- PSGMCL—A state machine that records the count of the number of STLs that the TAP.7 Controller believes were not Idle Group Members the last EPU TCK period.
- SSD state/Delayed SSD State—SSD and possibly Delayed SSD State Machines (depending on the implementation) record the last SSD that executed, the TAPC state in which it was executed, and whether the SSD created none, one, or all Potential Pause-IR or Pause-DR Group Members this and the previous TCK(C) period (see 5.5.3.5.4).

This information is used to determine the Star-4 TDO Drive Policy and plays a significant role in determining the TMSC Drive Policy described in Clause 14. The TAP.7 Controller combines the recorded information to determine the following:

- The number of Scan Group Members the prior EPU TCK period
- Whether the STL was the only Scan Group Member the prior EPU TCK period

#### 13.8.5 STL group candidate and membership counts

##### 13.8.5.1 Description

###### 13.8.5.1.1 Scan Group Candidate Count (SGCC)

The SGCC state tracks resets, commands, and SSDs that change the Scan Group Candidacy Register value. This tracking begins in an Online TAP.7 Controller with the *Test-Logic-Reset* state and continues during the use of all scan formats (Series, Star-4, and Star-2 Operation).

These states, described in Table 13-3, represent the number of TAP.7 Controllers in the technology branch associated with this TAP.7 Controller that may have a logic 1 Scan Group Candidate Register value. The use of the word “may” is described in 13.8.5.1.3.

**Table 13-3 — SGCC state definition**

SGCC state	Definition
<i>SGCC_NONE</i>	No Scan Group Candidates
<i>SGCC_ONE</i>	A maximum of one Scan Group Candidate.
<i>SGCC_MANY</i>	There may be more than one (many) Scan Group Candidates.

Changes in the Scan Group Candidate Count state are governed by Table 13-5. In addition, the following unconditionally set the SGCC state to *SGCC\_MANY*:

- The *Test-Logic-Reset* State      A TAP.7 Controller knows neither the number of TAP.7 Controllers sharing the DTS connection nor the Scan Selection State that is created at start-up for other TAP.7 Controllers that are sharing the DTS connection.
- A Selection Sequence      Where the OAC and EC selection criteria are met and Drive Protection Activation occurs (see Table 11-3).

Online TAP.7 Controllers presume that commands and SSDs affect other TAP.7 Controllers independently of whether they actually affect these TAP.7 Controllers.

Subsequent to the creation of the *SGCC\_MANY* state, commands and SSDs associated with the *Run-Test/Idle* state create SGCC state changes. The SGCC state of a TAP.7 Controller that is placed Offline while other TAP.7 Controllers remain Online is likely to have its SGCC state lose synchronization with the SGCC state of other TAP.7 Table controllers. The DTS' responsibilities related to this are described in 11.12.6.

A Scan Group Candidate Count State Machine tracks the number of Scan Group Candidates as shown in Figure 13-8.

Setting all SGC Registers to a logic 1 or  
Ambiguous operations with JScan0 - JScan2:

STMC(SGC = 1) |  
(Run-Test/Idle & SSD\_SA) |  
(JScan0 - JScan2 Scan Format & (SCNB (SGC = x) Command) | SDD\_SOC | SDD\_SOT))

Setting all SGC Registers to a logic 0:

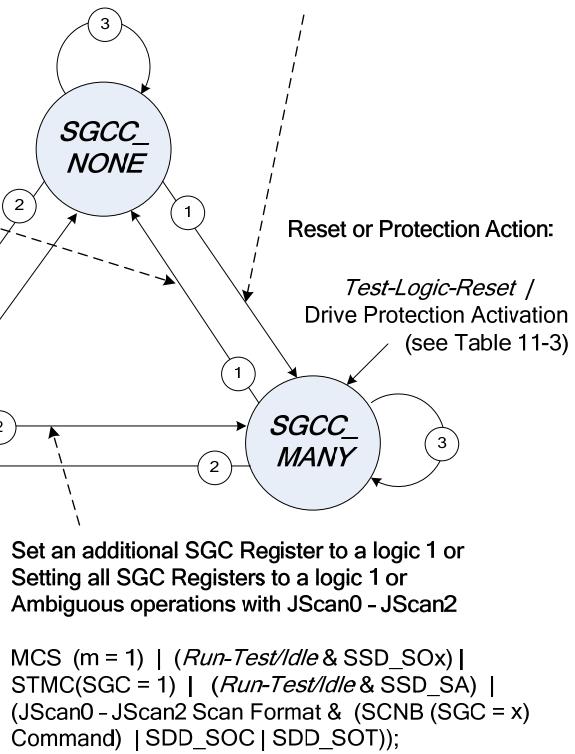
:  
STMC (SGC = 0) |  
(Run-Test/Idle & SSD\_DA) |  
(scan format other than JScan0 - JScan2 &  
SCNB (SGC = 0))  
:

Set one or an additional  
SGC Register to a logic 1:

:  
Any MCS command |  
(Run-Test/Idle & SSD\_SOx)  
:

Set only one SGC Register to a logic 1:

MCS setting only one  
SGC Register to a logic 1



**Figure 13-8 — Conceptual view of the Scan Group Candidate Count operation**

### 13.8.5.1.2 Potential Scan Group Membership Count Last

The PSGMCL represents the number of TAP.7 Controllers that were not members of the Idle Group in a TAP.7 Technology Branch in the prior EPU TCK period. It has the three states shown in Table 13-4.

**Table 13-4 — PSGMCL state definition**

PSGMCL state	Definition
PSGMCL_NONE	No Potential Scan Group Members
PSGMCL_ONE	A maximum of one Potential Scan Group Member
PSGMCL_MANY	There may be more than one (many) Potential Scan Group Members.

When the *Run-Test/Idle* state is reached and there are neither Pause-IR nor Pause-DR Group Members in the prior EPU TCK period (the Delayed SSD State is used to determine this), the PSGMCL state is set to the following settings

- SGCC state when the System Path is being used
- PSGMCL\_NONE state when the Control Path is used

The effects of the path being used when the *Run-Test/Idle* state is reached are described further in 19.8.1.4.

Changes in the PSGMCL state are governed by Table 13-6 and Figure 13-9. Additionally, all the conditions that unconditionally set the SGCC state to *SGCC\_MANY* also set the PSGMCL state to *PSGMCL\_MANY*.

### 13.8.5.1.3 Factors creating SGCC and PSGMCL ambiguity

The SGCC state is based on the number of commands and SSD actions and not the TCA or CID of controllers referenced by the commands and SSDs. There may be occasions where the TAP.7 Controller determines that there is more than one Scan Group Candidate when there may be none or only one Scan Group Candidate. These ambiguities are transferred to the PSGMCL when it is loaded with the SGCC value. These ambiguities are propagated to the computation of the number of Scan Group Members.

When the TAP.7 Controller determines that the number of Scan Group Members is larger than the actual number, Inhibited Drive is used for the TDOC signal when there would be no drive conflict if Single Drive would be used (since there may actually be only one Scan Group Member). This is quite acceptable as it merely invokes drive-conflict prevention early in part due to DTS software actions that are unneeded and in error. These DTS software actions should be avoided to ensure the desired communication with TAP.7 Controllers.

This approach for computing the number of Potential Scan Group Members is used because a TAP.7 Controller has no knowledge of TAP Controller Addresses and Controller IDs other than its own. A TAP.7 Controller cannot determine whether commands and SSDs use a TAP Controller Address or Controller ID that is:

- Not recognized by a TAP.7 Controller in the technology branch
- A command or SSD that duplicates a previous command or SSD

This means SGCC and PSGMCL ambiguity is caused by programming errors such as follows:

- A command setting the SGC Register of the same TAP.7 Controller more than once
- A command setting the SGC Register using an unallocated CID
- An SSD\_SOC SSD using a unallocated CID
- An SSD\_SOT SSD using either a TCA that does not match the TCA of a TAP.7 Controller

Each programming error above does not change the number of TAP.7 Controllers with selected STLs, but these programming errors do affect the SGCC and PSGMCL states. Additionally, ambiguous actions such as the CR Scan of the SCNB Command storing the SGC Register while using the JScan0–JScan2 Scan Formats set the SGCC state to *SGCC\_MANY* as it is impossible for a TAP.7 Controller to know the SGC values stored in the TAP.7 Controllers of other chips sharing a Series Scan Topology.

### 13.8.5.2 Specifications

#### Rules

- a) Each subsequent specification in 13.8.5.2 shall only apply to T3 and above TAP.7s.
- b) The number of TAP.7 Controllers that appear to have a logic 1 SGC Register value shall be tracked with an SGCC.
- c) The Scan Group Candidate Count state definitions shall be governed by Table 13-3.
- d) The Potential Scan Group Membership Count Last state definitions shall be governed by Table 13-4.

- e) The operation of the Scan Group Candidate Count State Machine shall be governed by Table 13-5.

**Table 13-5 — Scan Group Candidate Count operation**

<i>test_logic_reset_f?</i>	Drive Protection Activation?	JScan0–JScan2 Scan Format?	Event	SGCC	
				Current	Resulting
No	No	x	STMC, SGC= 0	X	SGCC_NONE
No	No	x	SSD_DA & <i>Run-Test/Idle</i>	X	SGCC_NONE
No	No	No	SCNB, SGC = 0	X	SGCC_NONE
No	No	No	MSC, m=0, iiii = xxxx (selects one)	X	SGCC_ONE
No	No	No	SSD_SOx & <i>Run-Test/Idle</i>	SGCC_NONE	SGCC_ONE
No	No	No	MSC, m=1, iiii = xxxx (adds to selection)	SGCC_NONE	SGCC_ONE
No	No	No		SGCC_ONE	SGCC_MANY
No	No	No		SGCC_MANY	SGCC_MANY
No	No	x	STMC, SGC= 1	X	SGCC_MANY
No	No	No	SCNB, SGC = 1	X	SGCC_MANY
No	No	No	SSD_SOx & <i>Run-Test/Idle</i>	SGCC_ONE	SGCC_MANY
No	No	No		SGCC_MANY	SGCC_MANY
No	No	x	SSD_SA & <i>Run-Test/Idle</i>	X	SGCC_MANY
No	No	Yes	SCNB, SGC = x	X	SGCC_MANY
No	No	Yes	MSC, m=x, iiii = xxxx	X	SGCC_MANY
No	Yes	x	x	X	SGCC_MANY
Yes	x	x	x	X	SGCC_MANY
None of the above				Unchanged	

NOTE 1—See Table 11-2 and Table 11-3 for a description of the Drive Protection Activation.

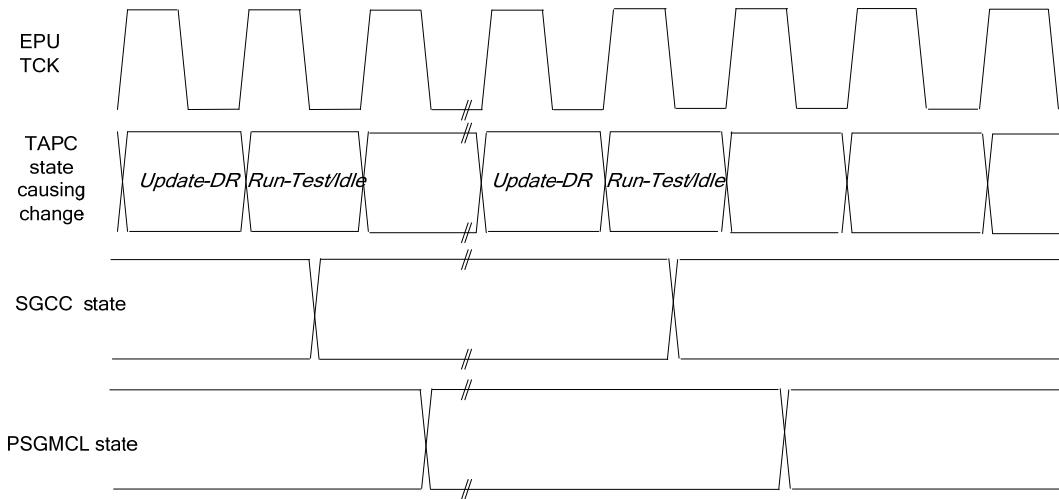
- f) The number of TAP.7 Controllers that appear to not be members of the Idle Group shall be tracked with a PSGMCL State Machine.
- g) The operation of the Potential Scan Group Membership Count State Machine shall be governed by Table 13-6.

**Table 13-6 — Potential Scan Group Membership Count Last operation**

<i>test_logic_reset_f</i> ?	Drive Protection Activation ?	<i>Run-Test/Idle f &amp; Delayed SSD State == SSD_SA_DL_Y_f</i> ?	System Path ?	Potential Scan Group Membership Count Last =	Membership description
Yes	x	x	x	<i>PSGMCL_MANY</i>	Force to many as a result of reset
No	Yes	x	x	<i>PSGMCL_MANY</i>	Force to many for drive protection
No	No	Yes	Yes	<i>SGCC state</i>	Number of the Potential Scan Group Members becomes the Scan Group Candidate Count
No	No	Yes	No	<i>PSGMCL_NONE</i>	Scan Group Candidacy is inhibited by the use of the Control Path. The number of Potential Scan Group Candidates become none.
None of the above			Unchanged	No change of state	

NOTE 2—See Table 11-2 and Table 11-3 for a description of Drive Protection Activation.

- h) Changes to the state of the Scan Group Candidate Count and Potential Scan Group Member Count Last shall be initiated by the falling edge of the EPU TCK(C) as shown in Figure 13-9.



**Figure 13-9 — SGCC and PSGMCL timing relationships**

### 13.8.6 Scan Group Membership Count Last determination

#### 13.8.6.1 Description

The Potential Scan Group Membership Count and Delayed SSD State are combined as shown in Table 13-8 to determine the Scan Group Membership Count in the last EPU TCK period (SGMCL) (the number of Scan Group Members). The PSGMCL state identifies the number of STLs that are not Idle Group members. The Delayed SSD State identifies the number of STLs that are not Pause-xR Group Members. The definition of the SGMCL states is shown in Table 13-7.

**Table 13-7 — SGMCL state definition**

SGMCL state	Definition
<i>SGMCL_NONE</i>	No Scan Group Candidates the prior EPU TCK period.
<i>SGMCL_ONE</i>	A maximum of one Scan Group Candidate the prior EPU TCK period.
<i>SGMCL_MANY</i>	There may be more than one (many) Scan Group Candidates the prior EPU TCK period.

#### 13.8.6.2 Specification

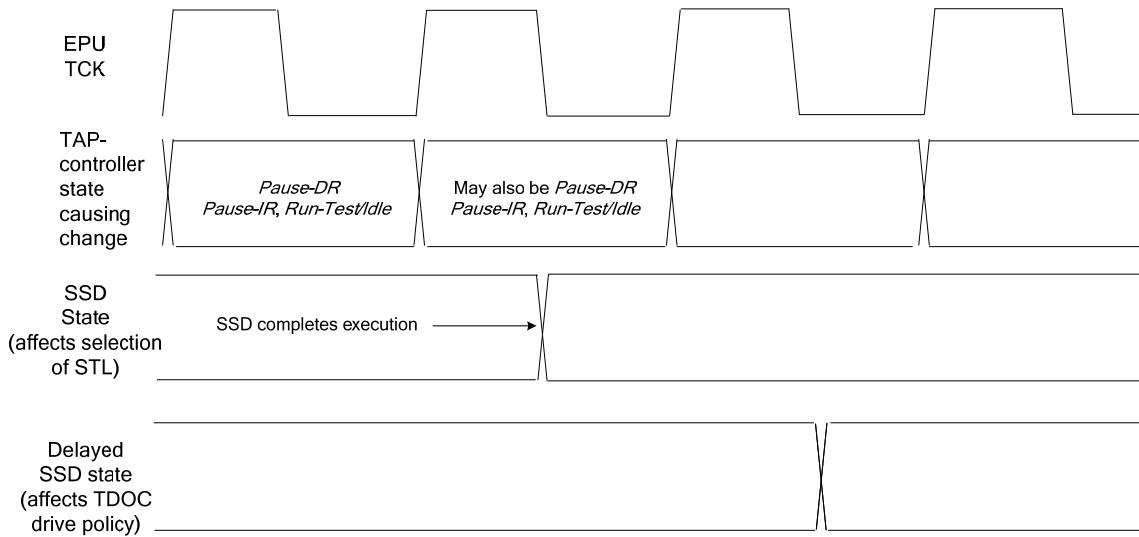
##### Rules

- a) Each subsequent specification in 13.8.6.2 shall only apply to T3 and above TAP.7s.
- b) The number of TAP.7 Controllers that appear to be Scan Group Members shall be tracked with the SGMCL.
- c) Changes to the state of the Scan Group Membership Count Last shall be initiated by the falling edge of the EPU TCK(C).
- d) The definition of the Scan Group Membership Count Last states shall be governed by Table 13-7.
- e) The generation of the Scan Group Membership Count Last state shall be governed by Table 13-8.

**Table 13-8 — Scan Group Membership Count Last operation**

Delayed SSD State	Potential Scan Group Membership Count	Scan Group Membership Count	Membership description
<i>SSD_DA_xx_DLY</i>	x	<i>SGMCL_NONE</i>	All are Pause-IR, Pause-DR, and Idle Group Members
x	<i>PSGMCL_NONE</i>	<i>SGMCL_NONE</i>	All are Idle Group Members
<i>SSD_NS_xR_DLY</i>	<i>~PSGMCL_NONE</i>	<i>SGMCL_ONE</i>	Appearance of one Scan Group Member
<i>SSD_IS_xR_DLY</i>	<i>~PSGMCL_NONE</i>	<i>SGMCL_ONE</i>	Appearance of one Scan Group Member
<i>SSD_SA_DLY</i>	<i>PSGMCL_ONE</i>	<i>SGMCL_ONE</i>	Appearance of one Scan Group Member
	<i>PSGMCL_MANY</i>	<i>SGMCL_MANY</i>	Appearance of More than one Scan Group Member

- f) The timing relationships between the TAPC state, SSD state, and the Delayed SSD States shall be governed by Figure 13-10.



**Figure 13-10 — SSD State/Delayed SSD State/timing relationships**

### 13.8.7 Only Scan Group Member Last determination

#### 13.8.7.1 Description

The Scan Group Membership Count Last and the Potential Scan Group Membership state are combined as shown in Table 13-12 to determine whether the STL is the only Scan Group Member. The Pause-xR Group Membership is governed by Table 13-11.

#### 13.8.7.2 Specification

##### Rules

- a) Each subsequent specification in 13.8.7.2 shall only apply to T3 and above TAP.7s.
- b) The STL shall be considered a Scan Group Member (the CLTAPC and ADTAPC operate in lock-step) when any of the following are true:
  - 1) All the following are true:
    - i) The ADTAPC state is *Test-Logic-Reset*.
    - ii) The default CLTAPC scan selection state is selected.
  - 2) All the following are true:
    - i) The ADTAPC state is not *Test-Logic-Reset*.
    - ii) The STL is not a member of the Idle Group as defined by Table 13-9 and Table 13-10.
    - iii) The STL is neither a member of the Pause-IR Group nor a member of the Pause-DR Group as defined by Table 13-11.
- c) The TAP.7 Controller shall combine the Scan Group Membership Count Last and Potential Scan Group Membership states as shown in Table 13-12 to determine whether the STL is the only Scan Group Member.

**Table 13-9 — Conditions governing Idle Group Candidacy**

<b>Control Path</b>	<b>Scan format</b>	<b>SGC Register == 0 ?</b>	<b>Idle Group Candidacy (IGC)</b>
Yes	x	x	Yes
x	JScan0	x	No
x	JScan1	x	Yes
x	Others	Yes	Yes
x		No	No

**Table 13-10 — Idle Group Membership**

<b>Run-Test/Idle_f == 1 ?</b>	<b>Last SSD executed == SSD_SA ?</b>	<b>Idle Group Membership (IGM)</b>
Yes	Yes	IGM = IGC
Others		No Change

**Table 13-11 — Conditions governing Pause-IR and Pause-DR Group Membership**

<b>Idle Group Member ?</b>	<b>Last executed SSD ?</b>	<b>The TAP.7 Controller the target of the last SSD executed ?</b>	<b>Results in Pause-xR Group Membership</b>
Yes	x	x	No
No	SSD_DA	x	Yes
No	SSD_SOC	No	Yes
		Yes	No
No	SSD_SOT	No	Yes
		Yes	No
No	SSD_SA	x	No

**Table 13-12 — Only Scan Group Member delayed determination**

<b>Scan Group Membership Count Last</b>	<b>Potential Scan Group Member Last ?</b>	<b>Only Scan Group Member last</b>	<b>Membership description</b>
SGMCL_NONE	x	No	No Scan Group Members
SGMCL_ONE	No	No	Appearance of one Scan Group Member but not this STL
	Yes	Yes	Appearance of one Scan Group Member and it was this STL
SGMCL_MANY	x	No	Appearance of more than one Scan Group Member

NOTE—only\_sgml== 1 when the only Scan Group Member Last is True.

## 13.9 EPU Group Membership

### 13.9.1 Description

#### 13.9.1.1 Tracking the EPU's Group Membership

With the Star-4 Operation and the use of the Control Path, the TAP.7 Controller is permitted to drive the TDOC signal, provided it determines its EPU is the only Conditional Group Member. The TAP.7 Controller determines whether its EPU is the only Conditional Group Member by tracking the activity that creates and vacates Conditional Group Membership. It combines this information with whether the EPU is a Conditional Group Member. Recall that an EPU is a Conditional Group Member when its GGM Register value is a logic 1 and is an Unconditional Group Member when the CGM Register value is a logic 0.

The TAP.7 Controller determines the number of Conditional Group Members by tracking resets and commands that change the CGM Register value. Beginning with the *Test-Logic-Reset* state, the TAP.7 Controller tracks the operations that change the CGM Register value, recording the number of EPUs that may have a CGM Register value of logic 1 with a CGMC State Machine.

#### 13.9.1.2 Conditional Group Member Count

Conditional Group Membership is defined directly with the CGM Register value. The effects of the Conditional Group Membership Register are immediate (unlike the effects of the Scan Group Membership Register which are delayed). This eliminates the need for Conditional Group Candidacy. It also simplifies both the creation of and counting of Conditional Group Members when compared to creating and counting of the Scan Group Candidates and Potential Scan Group Members.

When a TAP.7 Controller is Online, commands manage the CGM Register as follows:

- CMD(STMC, CGM = 0): Set the CGM Register value to a logic 0.
- CMD(STMC, CGM = 1): Set the CGM Register value to a logic 1. Make a Conditional Group Member.
- CMD(SCNB, CGM = x): Set the CGM Register value to value x.
- CMD (MCM, M[*CID*]): *m*==0: Set the CGM Register value of the targeted TAP.7 Controller to a logic 1. Set the CGM Register value of a nontargeted controller to a logic 0.
- CMD (MCM, M[*CID*]): *m*==1: Set the CGM Register value of the targeted controller to a logic 1. No change in the CGM Register value of nontargeted TAP.7 Controller.

With the last two commands listed above, the command identifies the targeted TAP.7 Controller using its CID.

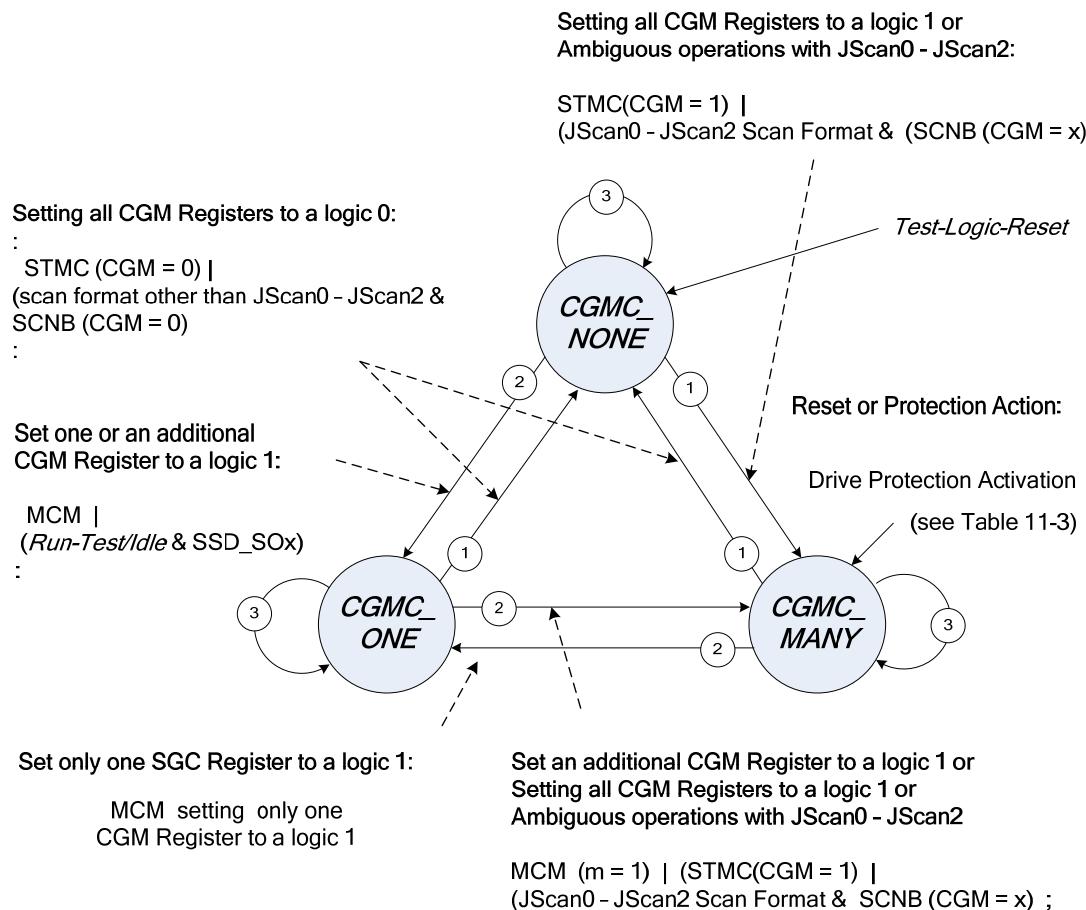
The CGMC State Machine determines the number of TAP.7 Controllers that may have a logic 1 CGM Register value by tracking the commands affecting the CGM Register.

The term “may have a logic 1 CGM Register value” is used to describe the CGMC state because the CGM Selection count may actually be larger than the number of TAP.7 Controllers with their CGM Register values a logic 1. Factors similar to those creating ambiguity in the Scan Group Membership Count described in 13.8.5.1.3 may create CGMC state ambiguity.

The three CGMC states shown in Figure 13-11 are described in Table 13-13.

**Table 13-13 — Conditional Group Member Count state definition**

CGMC state	Description
CGMC_NONE	No Conditional Group Members
CGMC_ONE	A maximum of one Conditional Group Member
CGMC_MANY	More than one (many) Conditional Group Members



**Figure 13-11 — Conceptual view of the Conditional Group Membership Count operation**

The conditions affecting the CGMC state are shown in Table 13-14.

### 13.9.1.3 Only Conditional Group Member determination

The Conditional Group Membership Count is combined with the Conditional Group Membership Register value to as shown in Figure 13-12 to determine whether the EPU is the only Conditional Group Member.

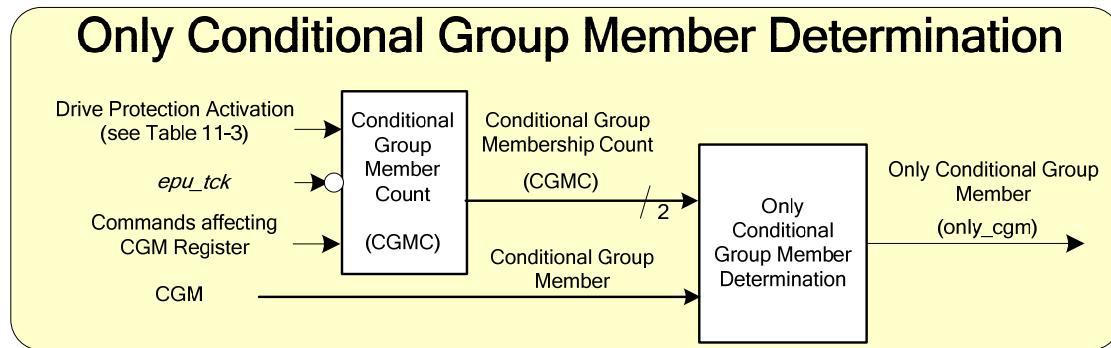


Figure 13-12 — Only Conditional Group Member determination

### 13.9.2 Specifications

#### Rules

- a) Each subsequent specification in 13.9.2 shall only apply to T3 and above TAP.7s.
- b) The number of TAP.7 Controllers that appear to have a logic 1 CGM Register value shall be tracked with a CGMC state.
- c) Changes to the CGMC state shall be initiated by the falling edge of the TCK(C).
- d) The CGMC state definition shall be governed by Table 13-13.
- e) The operation of the CGMC State Machine shall be governed by Table 13-14.

**Table 13-14 — Operation of the Conditional Group Membership Count State Machine**

<i>test_logic_reset_f</i> ?	Drive Protection Activation ?	JScan0– JScan2 Scan Format ?	Command	CGMC state	
				Current	Next
Yes	x	x	—	x	CGM_NONE
No	No	x	STMC, CGM, value = a logic 0	x	CGM_NONE
No	No	No	SCNB, CGM, value = 0	x	CGM_NONE
No	No	No	MCM, m=0, iii = xxxx (select one)	x	CGM_ONE
No	No	No	MCM, m=1, iii = xxxx (adds to selection)	CGM_NONE	CGM_ONE
No	No	No		CGM_ONE	CGM_MANY
No	No	No		CGM_MANY	CGM_MANY
No	No	No	SCNB, CGM, value = 1	x	CGM_MANY
No	No	x	STMC, CGM value == logic 1	x	CGM_MANY
No	No	Yes	SCNB, CGM, value = x	x	CGM_MANY
No	No	Yes	MCM, m=x, iii = xxxx	x	CGM_MANY
No	Yes	x	—	x	CGM_MANY
None of the above				Unchanged	

NOTE 1—See Table 11-2 and Table 11-3 for a description of the Drive Protection Activation.

- f) The TAP.7 Controller shall combine the Conditional Group Membership Count state and Conditional Group Membership Register value as shown in Table 13-15 to determine whether the EPU is the only Conditional Group Member.

**Table 13-15 — Only Conditional Group Member determination**

<b>CGMC state</b>	<b>CGM == 1 ?</b>	<b>Only Conditional Group Member</b>	<b>Description</b>
<i>CGMC_NONE</i>	x	No	No Conditional Group Members
<i>CGMC_ONE</i>	No	No	Appearance of one Conditional Group Member and not a Conditional Group Member
<i>CGMC_ONE</i>	Yes	Yes	Appearance of one Conditional Group Member and a Conditional Group Member
<i>CGMC_MANY</i>	x	No	Appearance of more than one Conditional Group Member

NOTE 2—only\_cgm === 1 when the only Conditional Group Member.

### 13.10 Drive Policy summary

A summary of the TDO(C) Drive Policy is provided below. The Dormant Drive Policy inhibits the TDOC drive when any of the miscellaneous factors described in 13.3 occur. With a T0 TAP.7, the TDO signal is driven as directed by the CLTAPC (as specified by IEEE Std 1149.1).

With a T1 and T2 TAP.7, the TDO signal is driven as directed by the CLTAPC when the STL supplies the scan path and the ADTAPC when the EPU supplies the scan path. In these cases, the drive of the TDO signal is permitted when using the following elements:

- System Path, provided any of the following are true:
  - The STL is a Scan Group Member and its CLTAPC enables the drive of the TDO(C) signal
  - The STL is a not Scan Group Member and the ADTAPC state is *Shift-xR\_f*
- Control Path, provided all of the following are true:
  - The ADTAPC state is *Shift-DR\_f*
  - A control level has been established (the ZBS count has been locked)
- One of the following is true:
  - The CR Scan of an SCNS or SCNB Command is active
  - The control level is greater than or equal to three

With a T3 and above TAP.7 with the TDOC signal, the drive policy for the TDO(C) signal is the same as for a T2 TAP.7 for Series Operation. With a T3 and above TAP.7 and Star-4 Operation, the drive of the TDOC signal is activated when using the following elements

- System Path when all of the following are true:
  - The STL is a Scan Group Member
  - The TAP.7 Controller has determined its STL is the only Scan Group Member
  - The CLTAPC enables the drive of the TDO(C) signal
- Control Path when the ADTAPC state is *Shift-DR\_f* and all of the following are true:
  - The control level is two
  - Any of the following are true:

- An SCNB or SCNS CR Scan is active and the TAP.7 Controller has determined its EPU is the only Conditional Group Member
- A CIDA CR Scan is active and the operation of the command requires the TDO(C) to be driven
- The control level is four or five and the TAP.7 Controller has determined its EPU is the only Conditional Group Member

## 13.11 An approach to implementing TDOC Drive Policy

### 13.11.1 Policy generation

An approach to the implementation of the TDOC Drive Policy for a T3-T5 TAP.7 is shown in Figure 13-13 given signal descriptions in Table 13-16. This approach considers minimizing the delays in enabling TDOC drive. Other approaches should also be considered.

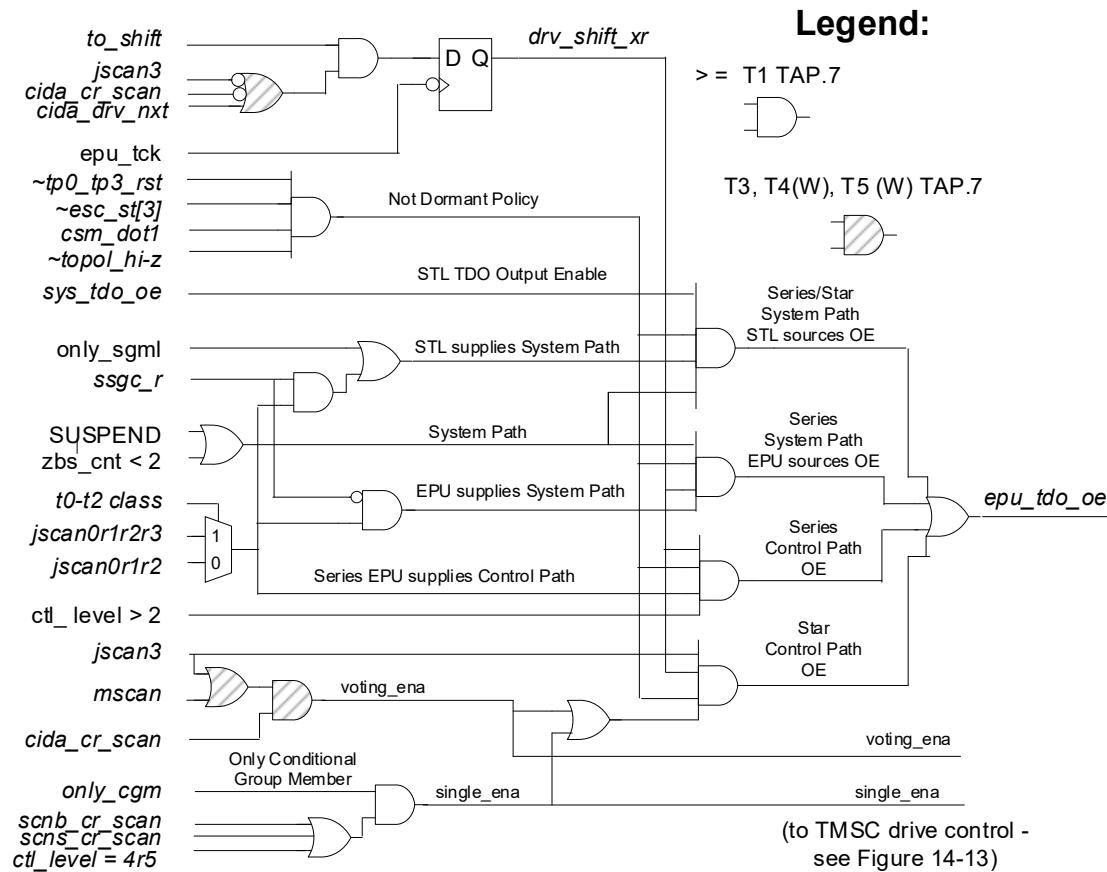


Figure 13-13 — Example of TDOC Drive Policy for a T3 and above TAP.7

**Table 13-16 — Signal descriptions in Figure 13-13**

Signal	Description	Reference
<i>cida_cr_scan</i>	The CIDA CR Scan is active.	—
<i>cida_drv_nxt</i>	The CIDA CMD. Operation requires the TDOC signal to drive the next TCK period.	—
<i>csm_dot1</i>	The CSM DOT1 state is active.	—
<i>ctl.level &gt;2</i>	The control level is greater than two.	—
<i>ctl.level == 4r5</i>	The control level is either four or five.	—
<i>esc_st[3]</i>	The detection of a Reset Escape has occurred.	—
<i>jscan0r1r2r3</i>	A JScan Scan Format has been specified.	—
<i>jscan0r1r2</i>	A JScan0 Scan Format has been specified.	—
<i>jscan3</i>	The use of the JScan3 Scan Format is specified.	—
<i>mscan</i>	The MScan Scan Format.	—
<i>only_cgm</i>	The only Conditional Group Member.	Table 13-15
<i>only_sgm_last</i>	The only Scan Group Member Last.	Table 13-12
<i>scnb_cr_scan</i>	SCNB CR Scan is active.	—
<i>scns_cr_scan</i>	SCNS CR Scan is active.	—
<i>shift_dr_f</i>	The Shift-DR state delayed to the falling edge of EPU_TCK.	—
<i>t0-t2_class</i>	T0-T2 TAP.7.	—
<i>topol_hiz</i>	The Topology Register value is 00b.	—
<i>to_shift</i>	The next TAPC state is Shift-xR.	—
<i>tp0_tp3_rst</i>	Type-0 through Type-3 Reset.	Figure 10-3

### 13.11.2 Potential Scan Group Member Last

A conceptual view of the function that determines whether the STL was the only Scan Group Member in the prior EPU TCK period is shown in Figure 13-14. The information created with this function is used with both the TDOC Drive Policy and the TMSC Drive Policy related to the System Path.

### 13.11.3 The SGCC and PSGMCL functions

An approach to the implementation of the SGCC and PSGMCL State Machines is shown in Figure 13-14. Other approaches should also be considered. These machines utilize the state encoding shown in Table 13-17.

**Table 13-17 — SGCC and PSGMCL State Machine encoding**

<i>sgxc_none_r</i>	<i>sgxc_one_r</i>	SGCC state	PSGMCL state
Logic 1	Logic 0	<i>SGCC_NONE</i>	<i>PSGMCL_NONE</i>
Logic 0	Logic 1	<i>SGCC_ONE</i>	<i>PSGMCL_ONE</i>
Logic 0	Logic 0	<i>SGCC_MANY</i>	<i>PSGMCL_MANY</i>
Logic 1	Logic 1		Illegal

A conceptual view of the function that determines whether the STL was the only Scan Group Member in the prior EPU TCK period is shown in Figure 13-14. The information created with this function is used with both the TDOC Drive Policy and the TMSC Drive Policy related to the System Path.

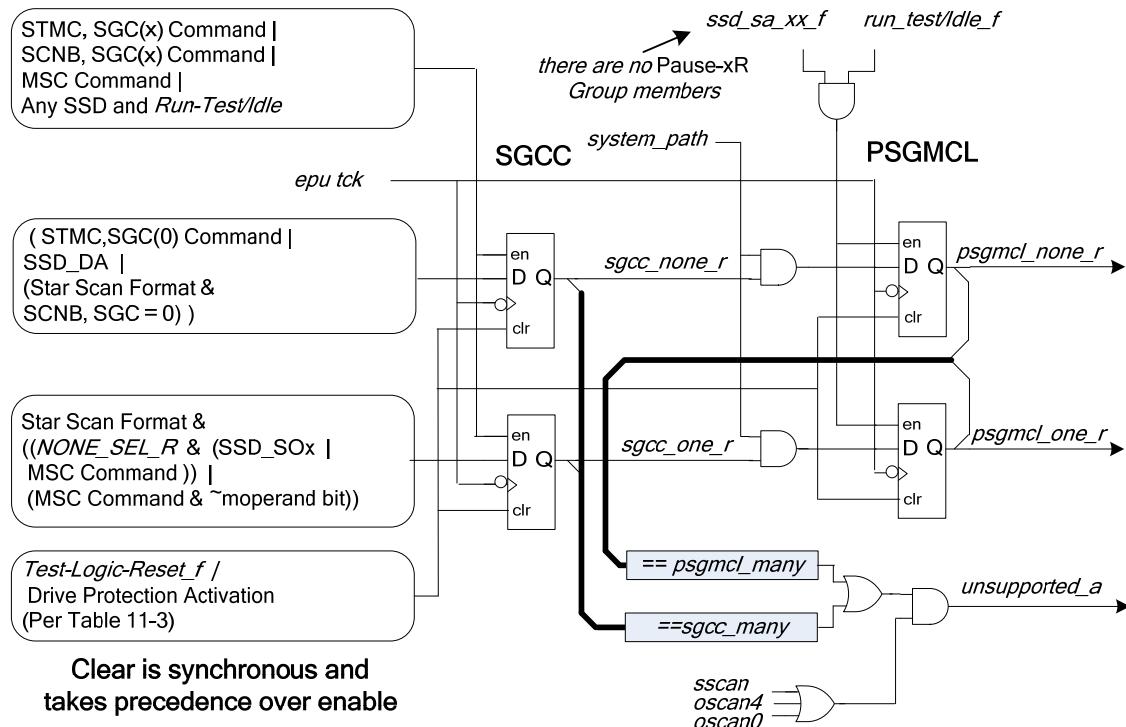


Figure 13-14 — Conceptual view of SGCC and PSGMCL State Machines

#### 13.11.4 Determining Scan Group Only Member Last/Membership Count Last

A conceptual view of the function that determines whether the STL was the only Scan Group Member the prior EPU TCK period is shown in Figure 13-14. The information created with this function is used with both the TDOC Drive Policy and the TMSC Drive Policy related to the System Path.

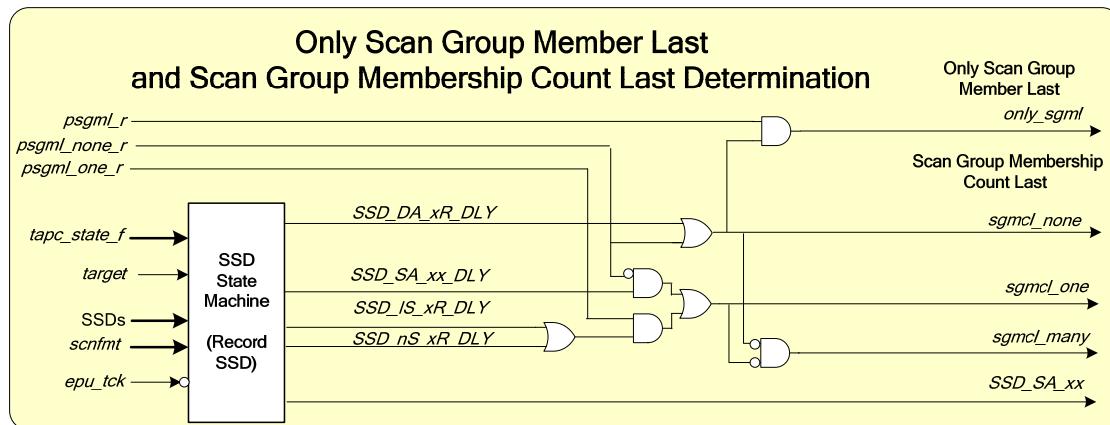


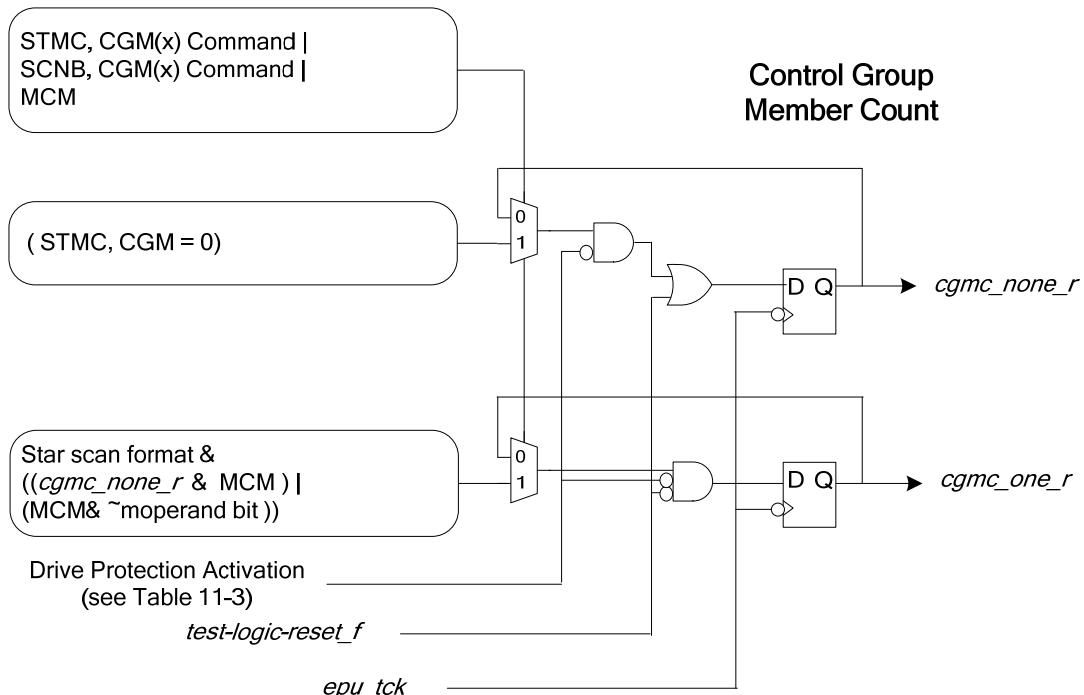
Figure 13-15 — Determining Scan Group Only Member Last/Membership Count Last

### 13.11.5 The CGMC function

An approach to the implementation of the CGMC is shown in Figure 13-16 with the state encoding shown in Table 13-18. Other approaches should be considered.

**Table 13-18 — CGMC state encoding**

<b>State name</b>	<i>cgmcs_one_r</i>	<i>cgmcs_none_r</i>	<b>Description</b>
<i>CGMC_NONE</i>	Logic 0	Logic 1	No Conditional Group Members
<i>CGMC_ONE</i>	Logic 1	Logic 0	A maximum of one Conditional Group Member
<i>CGMC_MANY</i>	Logic 0	Logic 0	More than one (many) Conditional Group Members
N/A	Logic 1	Logic 1	Illegal



**Figure 13-16 — Conceptual view of CGMC state development**

### 13.12 Programming considerations

A TAP.7 start-up sequence may be constructed to accommodate both simple and complex systems. It may include the Scan Topology Training Sequence to establish the types of TAP.7 Branches that are deployed. This start-up procedure may be constructed to detect the sharing of the TCK(C)/TMS(C) signals with other technologies when these technologies utilize the Online/Offline Feature defined by this standard. The start-up sequence described in Annex D provides for start-up of a system without drive conflicts including when the system scan topology is unknown and either TAP.1s or TAP.7s without an RSU are erroneously used with a Star-4 Scan Topology.

The use of the scan formats supporting the Series Scan Topology is restricted in a Star-4 Scan Topology to avoid drive conflicts. These restrictions are also covered in Annex D.

## 14. TMS(C) Signal Drive Policy

### 14.1 Introduction

This clause is applicable to T4 and above TAP.7s. It defines the TMSC Signal Drive Policy as a collection of drive policies for output bit types generated in the *SPA* and *TPA* Operating States. There is no TMSC output in the *CRA* and *BPA* Operating States. It also provides programming considerations.

The subject matter within this clause is organized in the following manner:

- 14.2 TMSC output bit types
- 14.3 Drive policy by output bit type
- 14.4 TMSC Signal Drive Types
- 14.5 Dormant Bit Drive Policy
- 14.6 Precharge Bit Drive Policy
- 14.7 Ready (RDY) Bit Drive Policy
- 14.8 TDO Bit Drive Policy
- 14.9 Transport Bit Drive Policy
- 14.10 An approach to implementing TMSC Drive Policy
- 14.11 Programming considerations

### 14.2 TMS(C) output bit types

#### 14.2.1 Scan Packet content

A brief overview of an SP follows to prepare for a detailed description of the TMSC Drive Policy. An SP may be composed of as many as three elements as shown in Figure 14-1. TMSC Drive Policies are only related to the output portion of the Payload Element as the TMSC signal functions as an input during the remainder of the SP.

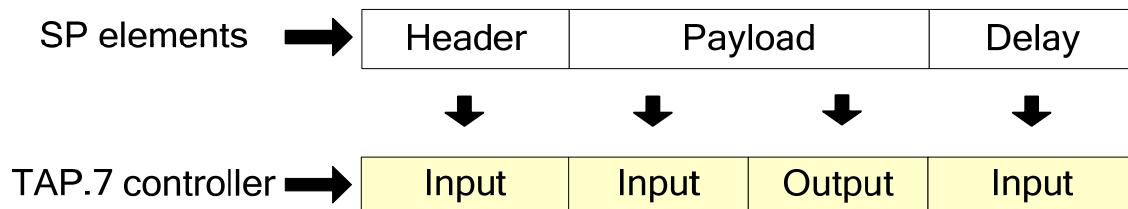


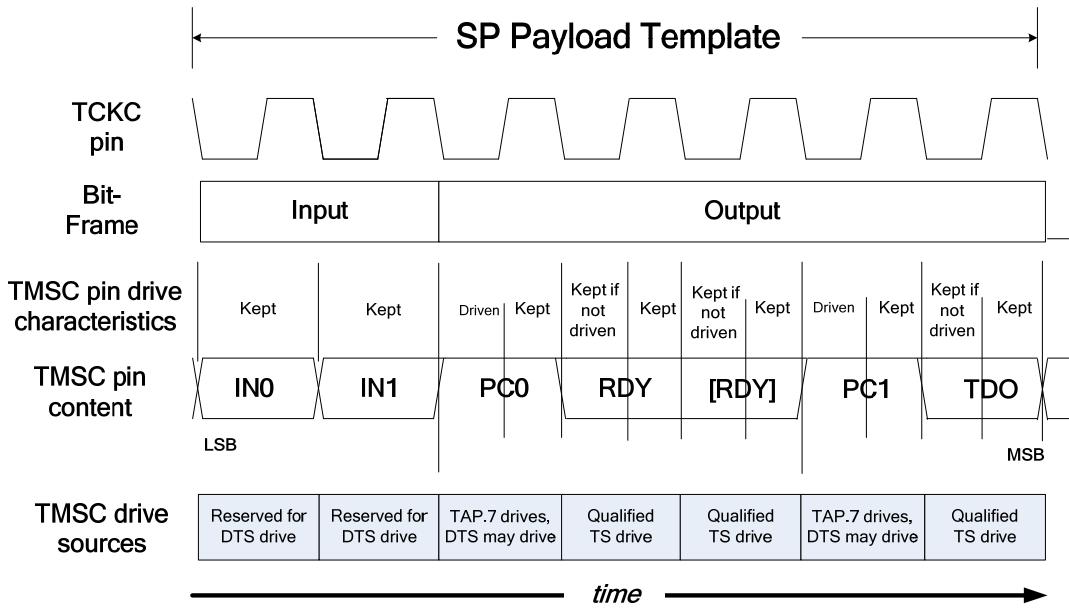
Figure 14-1 — SP content preview

The template for the SP Payload Element is shown in Figure 14-2. The input and output bit-frames are shown in this figure. The scan format specifies the bit types within the SP payload. The Delay Control Register specifies whether the SP contains a Delay Element. Within an SP, the TMSC Drive Policy permits the drive of the TMSC signal only within the output bit-frame. The function of bits within the output bit-frame is described briefly below:

- PCx Precharge of the TMSC signal to a logic 1 value

- RDY Provides a means to stall the SP progression based on the request of the component exchanging scan data with the DTS
- TDO The transfer of TDO scan information from the System and Control Paths

The SP contains one or more of the bits shown in the SP payload template. The MScan Scan Format uses all bits within the template. With the OScan and SScan Scan Formats, the Precharge 0 (PC0) and Precharge 1 (PC1) bits are deleted. The IN0, IN1, RDY, and TDO bits are also deleted with some versions of these scan formats. The SP contains a TDO bit when it contains an RDY bit. The TDO bit value may be forced to a logic 1 at times.



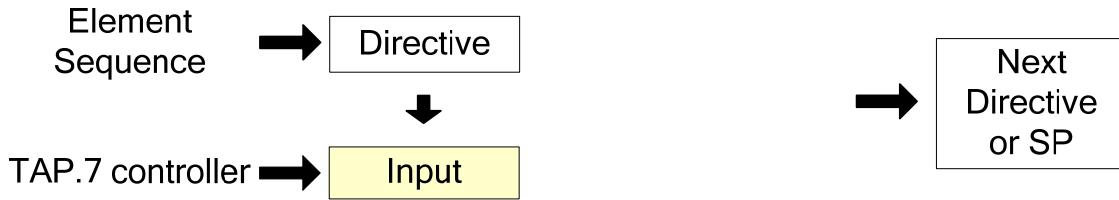
**Figure 14-2 — SP payload template**

#### 14.2.2 Transport Packet content

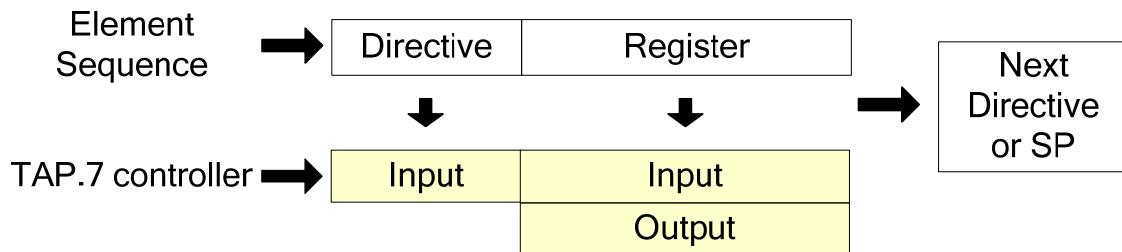
A brief overview of a Transport Packet follows to prepare for a detailed description of the TMSC Drive Policy. A Transport Packet is composed of a series of Directive, Register, and Data Elements. The three transfer types shown in this figure may be constructed with these elements as shown in Figure 14-3. The output bits within Register and Data Elements are the periods of interest for describing the TMSC Drive Policies. The TMSC signal functions as an input during other parts of the Transport Packet.

Register Element content can be designated as input or output. Data Element content can be designated as input, output, alternating input and output bits, or in a custom manner. With custom designation, the Data Channel Client defines the TMSC signal as not driven, an input, or an output on a bit-by-bit basis. The TMSC drive paradigm is fixed for a transfer designated as input, output, or alternating input and output bits. The direction of each bit of the transfer is defined by the protocol used by the Data Channel Client(s) with a custom transfer.

## Control Transfer



## Client Register Transfer



## Client Data Transfer

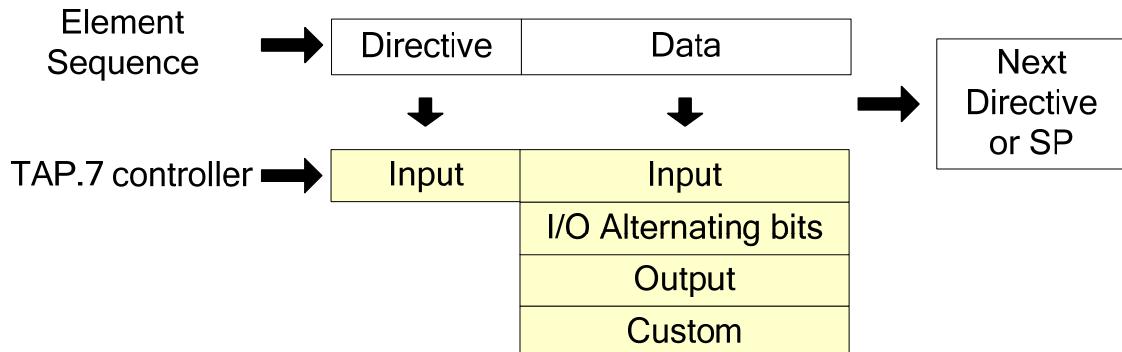
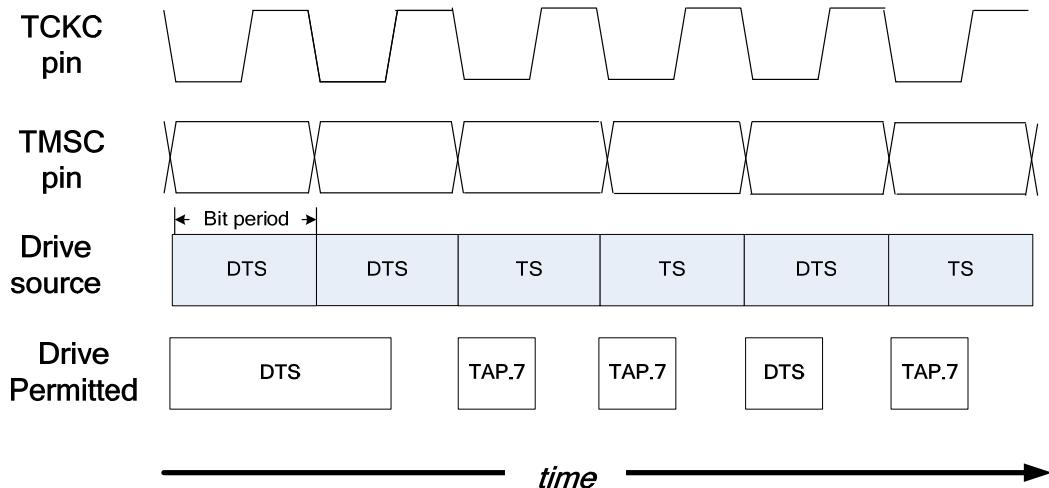


Figure 14-3 — Transport Packet content preview

### 14.2.3 Drive relationship with TCKC

Within the output bit-frame, the TMSC signal may be driven by the TAP.7, provided the TCKC signal value is a logic 0 as shown in Figure 14-4. The DTS may drive the TMSC signal continuously during a bit period when, in the subsequent bit period, it will drive the TMSC signal when the TCKC signal value is a logic 0. The DTS may drive the TMSC signal, provided the TCKC signal is a logic 0 otherwise.

Because the TAP.7 Controller may only drive the TMSC signal while the TCKC signal is a logic 0, there is no possibility of a drive conflict occurring when transitioning from one bit period to another. The TMSC value driven is kept while the TCKC signal is a logic 1 while the Advanced and Control Protocols are being used.



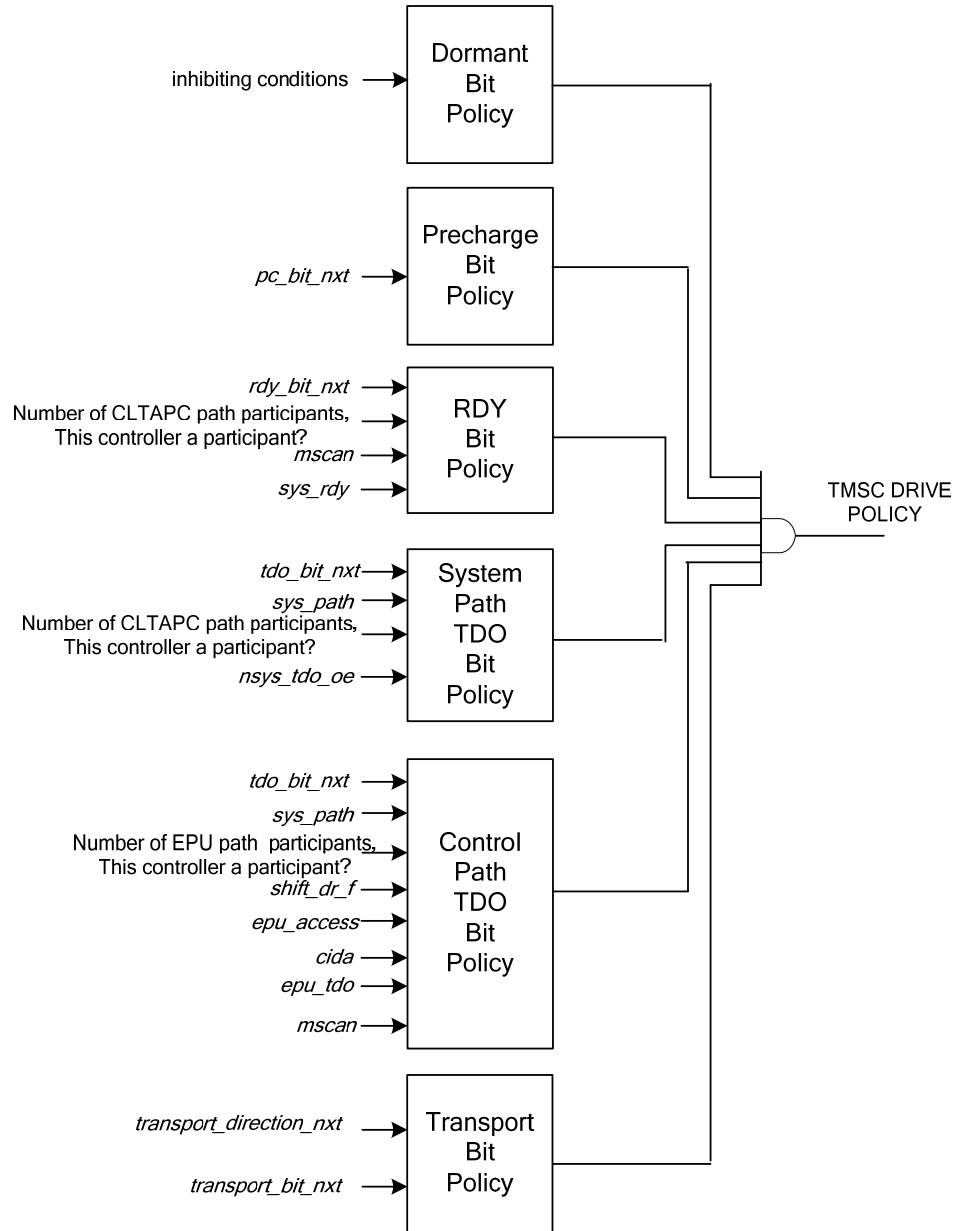
**Figure 14-4 — TMSC Advanced Protocol drive characteristics**

The combination of the DTS and TAP.7 TMSC drive characteristics allows the following:

- The drive direction to be changed every bit period without a DTS/TAP.7 drive conflict
- The use of Escapes at any time provided the TCK(C) signal is a logic 1

### 14.3 Drive policy by output bit type

A high-level view of the TMSC Drive Policy is shown in Figure 14-5. It is composed of a drive policy for each output bit type. The Dormant Bit Policy is similar to the Dormant TDO Drive Policy. The Precharge Bit Policy is arbitrary, with the TMSC signal driven to a logic 1. The RDY Bit Policy affects the ability of components within the STL to remain synchronized to the TAPC state progression. The two TDO Bit Drive Policies handle the sourcing of TDO bits by the System Path and the Control Path. The Transport Drive Policy handles TMSC drive during output bit periods within Transport Packets. A TAP.7 Controller private command or an STL private instruction may override the TMSC Drive Policy to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior.



**Figure 14-5 — The TMSC Signal's Distributed Drive Policy**

## 14.4 TMSC Signal Drive Types

#### **14.4.1 TMSC Signal Drive Types**

The following four types of TMS(C) drive are supported as described in 4.2.7.6:

- Single Drive with a value of the TAP.7 Controller's choosing
  - Joint Drive with a predetermined logic 1 value
  - Voting Drive with a logic 0 value only when the data it desires is a logic 0
  - Inhibited No drive

#### 14.4.1.1 Single Drive

Single Drive is used as the default. A single TAP.7 Controller establishes the value of the RDY and TDO bits.

#### 14.4.1.2 Joint Drive

Joint Drive provides for the simultaneous drive of the TMSC signal by multiple TAP.7 Controllers without producing drive conflicts. With simultaneous drive, the TMSC signal is driven to a logic 1 in a bit period to ensure there are no TMSC drive conflicts.

#### 14.4.1.3 Voting Drive

Voting Drive creates a logic 0 TMSC signal level when the data to be output is a logic 0 and a high-impedance state when the data to be output is a logic 1. Multiple TAP.7 Controllers may simultaneously drive a logic 0. Any TAP.7 Controller driving a logic 0 (voting for a logic 0) creates a logic 0 value, hence the name voting drive.

Voting is supported only by the MScan Scan Format. When using this scan format, TAP.7 Controllers vote on the value of the following:

- RDY bit when the selection count is *SYS\_MANY*
- TDO bit in SPs that are associated with the *Shift-DR* state of the CR Scan of a CIDA Command

In the case of the RDY bit, a single not ready-to-proceed vote (drive of a logic 0) causes a repeat of the vote with another PC0/RDY bit sequence. This means the Scan Packet progression is stalled until all TAP.7 Controllers are ready to proceed.

#### 14.4.1.4 Inhibited Drive

Inhibited Drive provides for the high-impedance state of the TMSC signal. Typical uses of this drive are use of the Standard Protocol, no drive while the Test Clock signal is a logic 1, or the use of no other drive types.

#### 14.4.2 Wire-ANDed TMSC signal values

The Joint and Voting Drive types are used to implement the Wire-AND of the TMSC signal values only with the MScan Scan Format. This drive paradigm is used with RDY bits when (1) there is more than one Scan Group Member and (2) with the CR Scans of CIDA Commands during CID allocation. Its use within a system is described in 22.2.5.

Wire-ANDed TMSC signal values are created by using a minimum of two TCK periods to transfer one meaningful bit of information. This sequence of drives types creates a precharge/discharge drive mechanism. In the precharge phase of the data exchange, either the PC0 or PC1 bit establishes a logic 1 TMSC bit value with Joint Drive. In the discharge phase of the data exchange, the Voting Drive is used to convey either a logic 0 data value by driving TMSC to a logic 0 or a logic 1 data value with no drive. A unanimous vote for a logic 1 by all TAP.7 Controllers participating in the creation of the Wire-ANDed data is required to convey a logic 1 data value.

## 14.5 Dormant Bit Drive Policy

### 14.5.1 Description

The Dormant Bit Drive Policy is described in Rule 14.5.2 b). This rule ensures that the Reset Escape can be used at any time without drive conflicts, provided the TCKC signal is a logic 1 value. The inhibiting of TMSC drive by the combination of the asynchronous version of Reset Escape and Type-0–Type-3 Resets ensures the TMSC signal is not driven by the TAP.7 Controller when the TCKC signal transitions to a logic 0 immediately following these resets.

### 14.5.2 Specifications

#### Rules

- a) Each subsequent specification in 14.5.2 shall only apply to T4 and above TAP.7s.
- b) The TMSC signal shall not be driven provided any of the following are true:
  - 1) The ADTAPC is Offline.
  - 2) TCKC is a logic 1.
  - 3) The asynchronous detection of a Reset Escape is active.
  - 4) A Type-0–Type-3 Reset is active.
  - 5) The Scan Format is not an Advanced Scan Format.

#### Permissions

- c) Following the *Test-Logic-Reset* state, a private command and/or private instruction may be used to alter the TMSC Drive Policy to produce IEEE 1149.7-Non-disruptive or IEEE 1149.7-Other Behavior.

## 14.6 Precharge Bit Drive Policy

### 14.6.1 Description

The Precharge Bit Drive Policy is independent of all factors other than the Dormant Drive Policy. With the PC0 and PC1 bit periods, Joint drive is used to drive the TMSC signal to a logic 1. The DTS may also drive the TMC signal to a logic 1 during these bit periods. This policy is summarized in Figure 14-6.



Figure 14-6 — Precharge Bit Drive Policy

### 14.6.2 Specifications

#### Rules

- a) Each subsequent specification in 14.6.2 shall only apply to T4 and above TAP.7s.
- b) The TAP.7 Controller shall drive the TMSC signal to a logic 1 during PC0 and PC1 bit periods.

## 14.7 RDY Bit Drive Policy

### 14.7.1 Description

#### 14.7.1.1 Characteristics

The RDY Bit Drive Policy is based on the number of Scan Group Members and whether the STL is the only Scan Group Member delayed by one EPU TCK cycle [SGMCL and Potential Scan Group Member (PSGM)]. It is independent of whether the System Path or Control Path is being used, although the use of these paths can affect these STL Group memberships. With this definition, changes in the RDY Bit Drive Policy lag changes in the CLTAPC selection state by one EPU Test Clock period. This is key to proper operation as RDY bits confirm the completion of the TAPC state advance of the prior SP as described in 14.7.1.3.

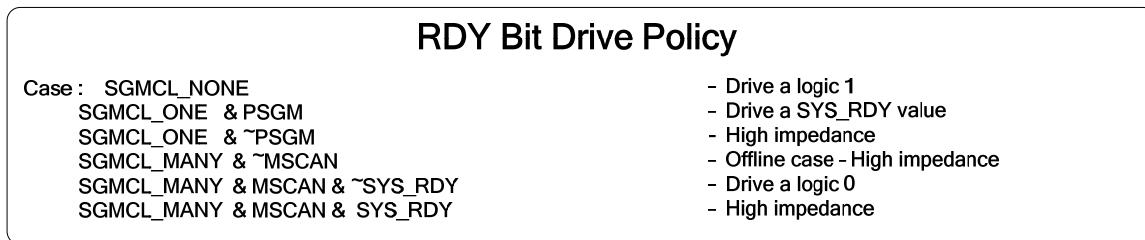
Because the DTS may create none, one, or more than one Scan Group Member through its normal use of the TS, the RDY bit drive policy has been specified in a manner that prevents RDY hangs when there is a minimum of one Online TAP.7 Controller. Logic 1 RDY bit values are generated when there are no Scan Group Members, avoiding ready hangs that would be created otherwise. When this operation is combined with placement of the ADTAPC Offline as a result of a Configuration Fault where RDY voting is or may be required but is unavailable, operation with RDY bits is robust. Recall that the simultaneous use of RDY bits and the selection or possible selection of more than one CLTAPC is supported only with the MScan Scan Format where Voting drive is supported.

#### 14.7.1.2 Policy details

The TMSC drive characteristics for RDY bit periods are determined using the PSGM and the number of Scan Group Members as shown below:

- MScan Scan Formats:
  - *SGMCL\_NONE* Joint Drive (logic 1)
  - *SGMCL\_ONE*
    - PSGM Single Drive (Ready value)
    - not PSGM Inhibited Drive
  - *SGMCL\_MANY*
    - PSGM Voting Drive (Ready value)
    - not PSGM Inhibited Drive
- OScan and SScan Scan Formats:
  - *SGMCL\_NONE* Joint Drive (logic 1)
  - *SGMCL\_ONE*
    - PSGM Single Drive (enabled data)
    - not PSGM Inhibited Drive
  - *SGMCL\_MANY* Not Applicable, blocked by Configuration Fault generation

These cases are summarized in Figure 14-7.



**Figure 14-7 — RDY Bit Drive Policy**

#### 14.7.1.3 Relationship to CLTAPC selection state changes

Changes in the selection of the CLTAPC of TAP.7 Controllers in the same branch during EPU Test Clock period n affect the RDY Bit Drive Policy during the EPU Test Clock period n + 1. This accommodates the behavior of System Ready for use cases where the DTS interacts with STL components that require a stall of the TAPC state progression. With these use cases, the RDY information in Scan Packet n indicates whether the processing of Scan Packet n – 1 has completed and whether the TDO data for Scan Packet n is available for export. Using a delayed view of the scan selection changes ensures both of the following:

- The processing of all Scan Packet information initiated by a *sys\_tck* associated with Scan Packet n – 1 completes in the STL before the progression of Scan Packet n is allowed to continue
- The STL will be ready during the Scan Packet that generates the first *sys\_tck* created by selection of the CLTAPC

This is shown in Figure 14-7.

#### 14.7.2 Specifications

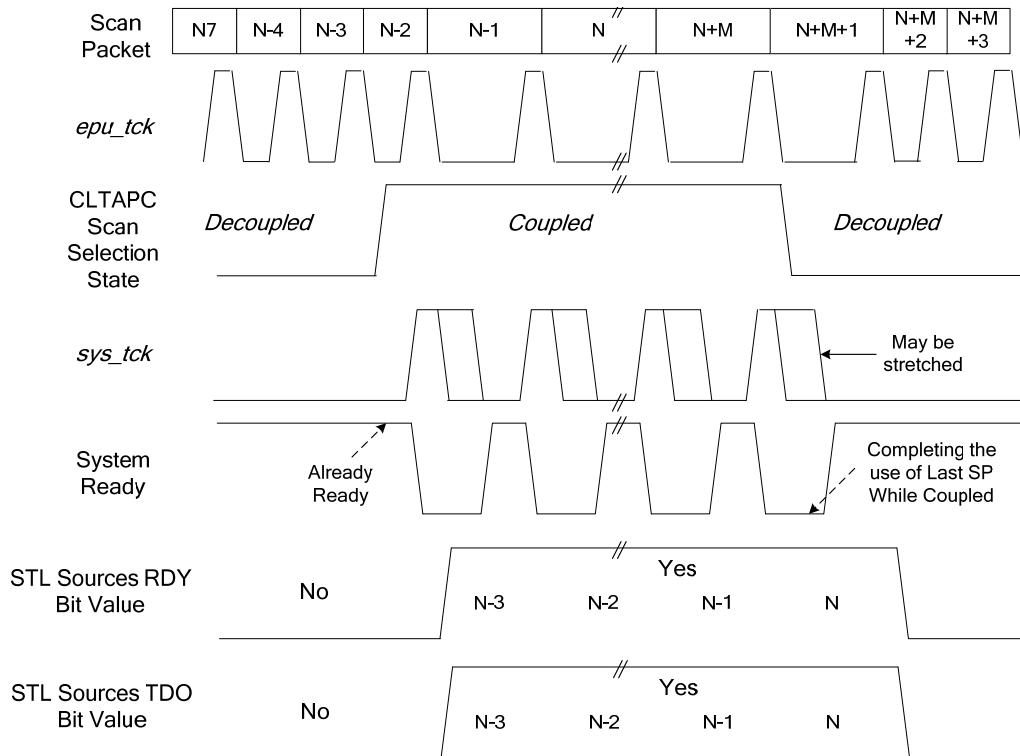
##### Rules

- a) Each subsequent specification in 14.7.2 shall only apply to T4 and above TAP.7s.
- b) The TMSC signal value created by the TAP.7 Controller during RDY bit periods shall be governed by Table 14-1.

**Table 14-1 — RDY Bit Drive Policies**

Scan Group Membership Count Last	Potential Scan Group Member ?	MScan Scan Format ?	System ready ?	Resulting TMSC Signal Drive Type	Membership description
<i>SGMCL_NONE</i>	x	x	x	Joint (logic 1)	No Scan Group Members- avoid RDY hang
<i>SGMCL_ONE</i>	No	x	x	Inhibited (Hi-Z)	One Scan Group Member, not a Scan Group Member
	Yes	x	x	Single (Ready Value)	One Scan Group Member, a Scan Group Member
<i>SGMCL_MANY</i>	x	No	x	Inhibited (Hi-Z)	ADTAPC placed Offline
	No	x	x	Inhibited (Hi-Z)	More than one Scan Group Member, not a Scan Group Member
	Yes	Yes	Yes	Voting (Hi-Z)	One Scan Group Member, not a Scan Group Member Indicate ready with no drive
	Yes	Yes	No	Voting (logic 0)	One Scan Group Member, not a Scan Group Member Indicate ready with drive of logic 0

- c) The relationship of CLTAPC selection state changes and their effects on the RDY Bit Drive Policy shall be governed by Figure 14-8.



**Figure 14-8 — CLTAPC selection state/STL sourced output value relationships**

## 14.8 TDO Bit Drive Policy

### 14.8.1 Description

#### 14.8.1.1 Characteristics

There are two drive policies for TDO bits, the first for TDO bits sourced by the System Path, and the second for TDO bits sourced by the Control Path.

##### 14.8.1.1.1 Policy details for System Path

The drive policy for TDO bits sourced by the System Path is determined by the number of Scan Group Members and whether the STL is a Scan Group Member, both delayed by one EPU TCK cycle. Changes in this drive policy lag changes in the CLTAPC selection state by one EPU Test Clock period (see Figure 14-8). This is viewed as having no consequences as selection state changes occur in states where the TDO data is not relevant. This policy may be determined using the PSGM and the number of Scan Group Members as shown:

- States other than *Shift-xR\_f* Joint Drive (logic 1)
- *Shift-xR\_f* state:
  - *SGMCL\_NONE* Joint Drive (logic 1)
  - *SGMCL\_MANY* Joint Drive (logic 1)
  - *SGMCL\_ONE*
    - Potential Scan Group Member: Single Drive (TDO value)
    - Not a Potential Scan Group Member: Inhibited Drive

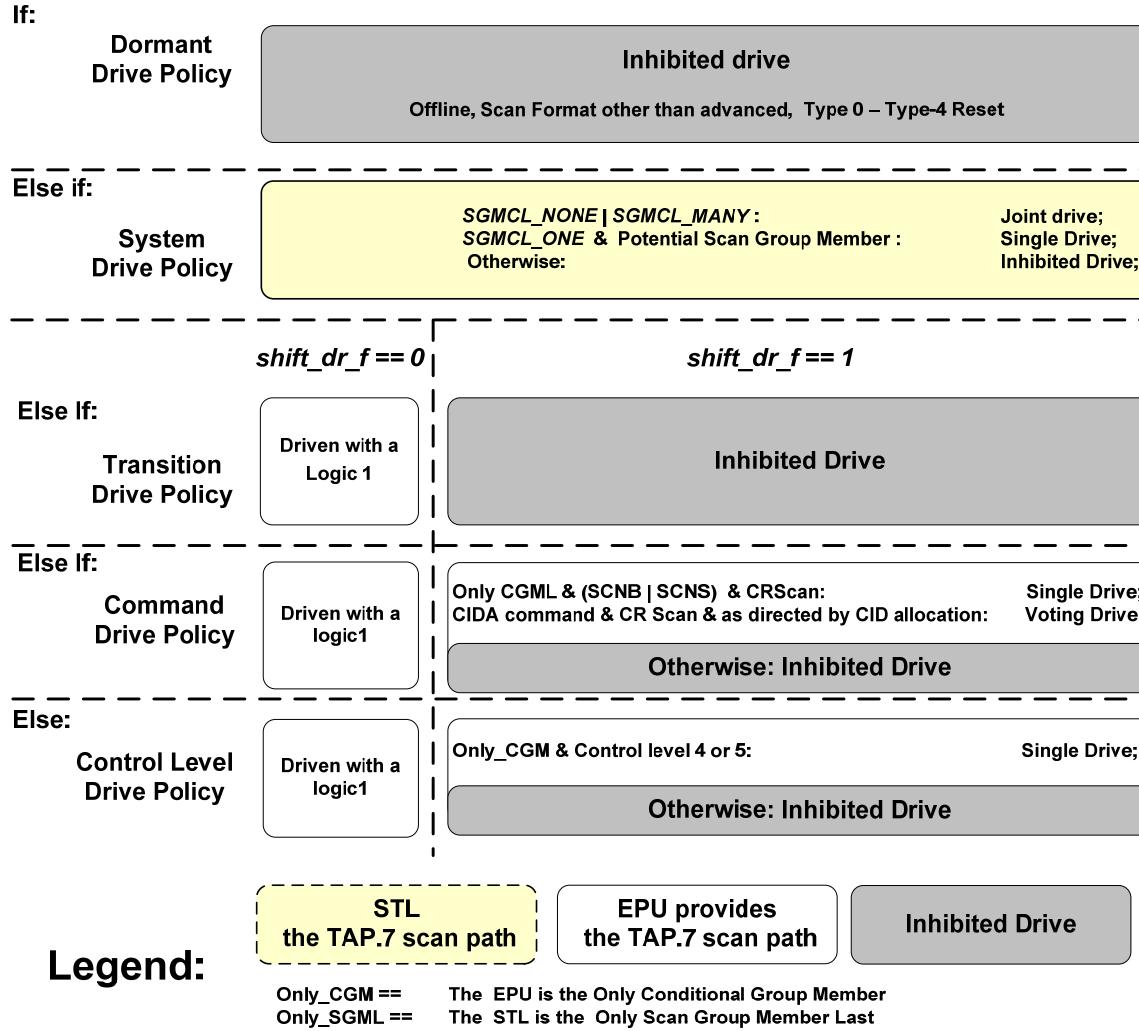
### 14.8.1.1.2 Policy details for Control Path

The drive policy for TDO bits sourced by the Control Path is determined by the number of Conditional Group Members and whether the EPU is a Conditional Group Member. With this definition changes in these memberships immediately affect this drive policy. This policy may be determined as shown below:

- |   |                                   |
|---|-----------------------------------|
| — States other than <i>Shift-DR-f</i>         | Joint Drive (logic 1)             |
| — <i>Shift-DR-f</i> state:                    |                                   |
| — A CIDA CR Scan and the MScan Scan Format:   |                                   |
| — No abstention                               | Voting Drive (Control Path value) |
| — Abstention                                  | Voting Drive (Not driven)         |
| — Only Conditional Group Member:              |                                   |
| — An SCNB/SCNS CR Scan                        | Single Drive: (TDO value)         |
| — A DR Scan with Control Levels Four and Five | Single Drive: (TDO value)         |
| — Other DR Scans                              | Inhibited Drive                   |
| — Not Only Conditional Group Member:          |                                   |
| — An SCNB/SCNS CR Scan                        | Joint Drive (logic 1)             |
| — A DR Scan with Control Levels Four and Five | Joint Drive (logic 1)             |
| — Other DR Scans                              | Inhibited Drive                   |

### 14.8.1.2 Correlation to TDO(C) Drive Policy

The correlation of the TDO(C) Drive Policy to the combination of System and Control Path Drive Policies is shown in Figure 14-9.



**Figure 14-9 — Correlation of TDO(C) and TDO Bit Drive Policies**

#### 14.8.1.3 Combined TDO bit drive summary

A summary of the TDO Bit Drive Policy for both STL and EPU sourced TDO values is shown in Figure 14-10.

## TDO Bit Drive Policy

For use of the System Path :

Potential Scan Group Member Last (PSGML)

Case :

- |                               |  |
|-------------------------------|--|
| <i>SGMCL_ONE &amp; PSGML</i>  | - Drive TDO value (See Rule 14.8.2 d)) |
| <i>SGMCL_ONE &amp; ~PSGML</i> | - High impedance                       |
| Others:                       | - Drive a logic 1                      |

For use of the Control Path :

*single\_enable = Only CGM & ((SCNB | SCNS & CR\_Scan) | (Ctl.Lvl 4|5))*  
*voting\_enable = CIDA & CR\_Scan & MScan*  
*voting\_abstain = (~cid alloc\_participant | EPU TDO data == 1)*

Case : *~Shift\_DR\_f*

- |   |                   |
|---|-------------------|
| <i>Shift_DR_f &amp; voting_enable &amp; ~voting_abstain</i> | - Drive a logic 1 |
| <i>Shift_DR_f &amp; single_enable</i>                       | - Drive a logic 0 |
| Others :  | - Drive TDO value |

- Inhibited drive

**Figure 14-10 — TDO Bit Drive Policy**

### 14.8.2 Specifications

#### Rules

- Each subsequent specification in 14.8.2 shall only apply to T4 and above TAP.7s.
- When the System Path is selected, the drive of the TMSC signal during TDO bit periods shall be governed by Table 14-2.

**Table 14-2 — TDO Bit Drive Policy with the System Path**

Scan Group Membership Count Last	Potential Scan Group Member ?	Resulting TMSC signal drive type	Drive description
<i>SGMCL_NONE</i>	X	Joint (Logic 1)	No Scan Group Members – All drive
<i>SGMCL_ONE</i>	No	Inhibited (HI-Z)	One Scan Group Member and not a Scan Group Member
	Yes	Single (TDO value)	One Scan Group Member and a Scan Group Member (TDO value created by System Path)
<i>SGMCL_MANY</i>	X	Joint (Logic 1)	More than one Scan Group Member – All drive

- When the Control Path is selected, the drive of the TMSC signal during TDO bit periods shall be governed by Table 14-3 and Table 14-4.

**Table 14-3 — Voting and single enables**

<b>MScan &amp; CIDA CR Scan?</b>	<b>Only Conditional Group Member?</b>	<b>SCNS/SCNB CR Scan?</b>	<b>Control level == 4 or 5 ?</b>	<b>Voting enable</b>	<b>Single enable</b>
Yes		x	x	Yes	No
x	Yes	Yes	x	No	Yes
x	Yes	x	Yes	No	Yes
Others				No	No

**Table 14-4 — TDO Bit Drive Policy with the Control Paths**

<b>Shift_DR_f</b>	<b>Voting enable</b>	<b>CID allocation participant ?</b>	<b>EPU TDO Data == 0 ?</b>	<b>Single enable</b>	<b>Resulting TMSC signal drive type</b>	<b>Drive description</b>
No	x	x	x	x	Joint (Logic 1)	Non-Shift-xR state
Yes	Yes	No	Yes	x	Voting (HI-Z)	CID allocation – Abstain vote
Yes	Yes	Yes	Yes	x	Voting (Logic 0)	CID allocation – Vote logic 0
Yes	Yes	Yes	No	x	Inhibited (HI-Z)	CID allocation – Abstain vote
Yes	x	x	x	Yes	Single (TDO data)	SCNB/ SCNS CR- Scan or Control Level 4 or 5 DR Scan (TDO value created by Control Path)
Others					Inhibited (HI-Z)	Not driven

NOTE—The TDO bit values for Single Drive shown in Table 14-2 and Table 14-4 are determined by Rules 25.15.2 b) and c), Rule 24.4 h) for the MScan Scan Format, Rule 25.4 n) for the OScan Scan Format, and Rules 26.5.2 f) through 26.5.2 h) for the SScan Scan Formats.

## 14.9 Transport Bit Drive Policy

### 14.9.1 Description

The Transport Bit Drive Policy, shown in Figure 14-11, is determined by the Data Channel Client using the Logical Channel for data transfers or a register-read directed at a single client. When a Transport Register is read via a directive, it is the only drive candidate. It controls the drive of the TMSC pin in bits designated as register-read data. When there is a single Physical Data Channel associated with the Logical Channel, the Data Channel Client connected to the Physical Channel manages the TMSC drive for data transferred to the DTS. When there are multiple Physical Data Channels associated with the Logical Channel, the Data Channel Clients connected to the Physical Channels will manage the TMSC drive for data transferred to the DTS with their shared protocol to avoid TMSC drive conflicts.

## Transport Bit Drive Policy

Case : Transport Output Enable	- Drive transport data
~Transport Output Enable:	- High impedance

**Figure 14-11 — Transport Bit Drive Policy**

### 14.9.2 Specifications

#### Rules

- a) Each subsequent specification in 14.9.2 shall only apply to a T5 and above TAP.7.
- b) The TMSC Drive Policy for bit periods within a Transport Packet shall be defined by the following:
  - 1) Directive Elements – Inhibited Drive.
  - 2) Register Elements – Rules 28.2.3.2 b) through 28.2.3.2 d).
  - 3) Data Elements – Rules 28.2.4.2 i) through 28.2.4.2 j).

NOTE—See Clause 28 for an expanded description of transport drive characteristics.

### 14.10 An approach to implementing TMSC Drive Policy

A conceptual view of the TAP.7 Controller's TMSC input and output function is shown in Figure 14-12 and Figure 14-13. The definitions of the miscellaneous signal names found in these figures are shown in Table 14-5.

Figure 14-12 shows the following:

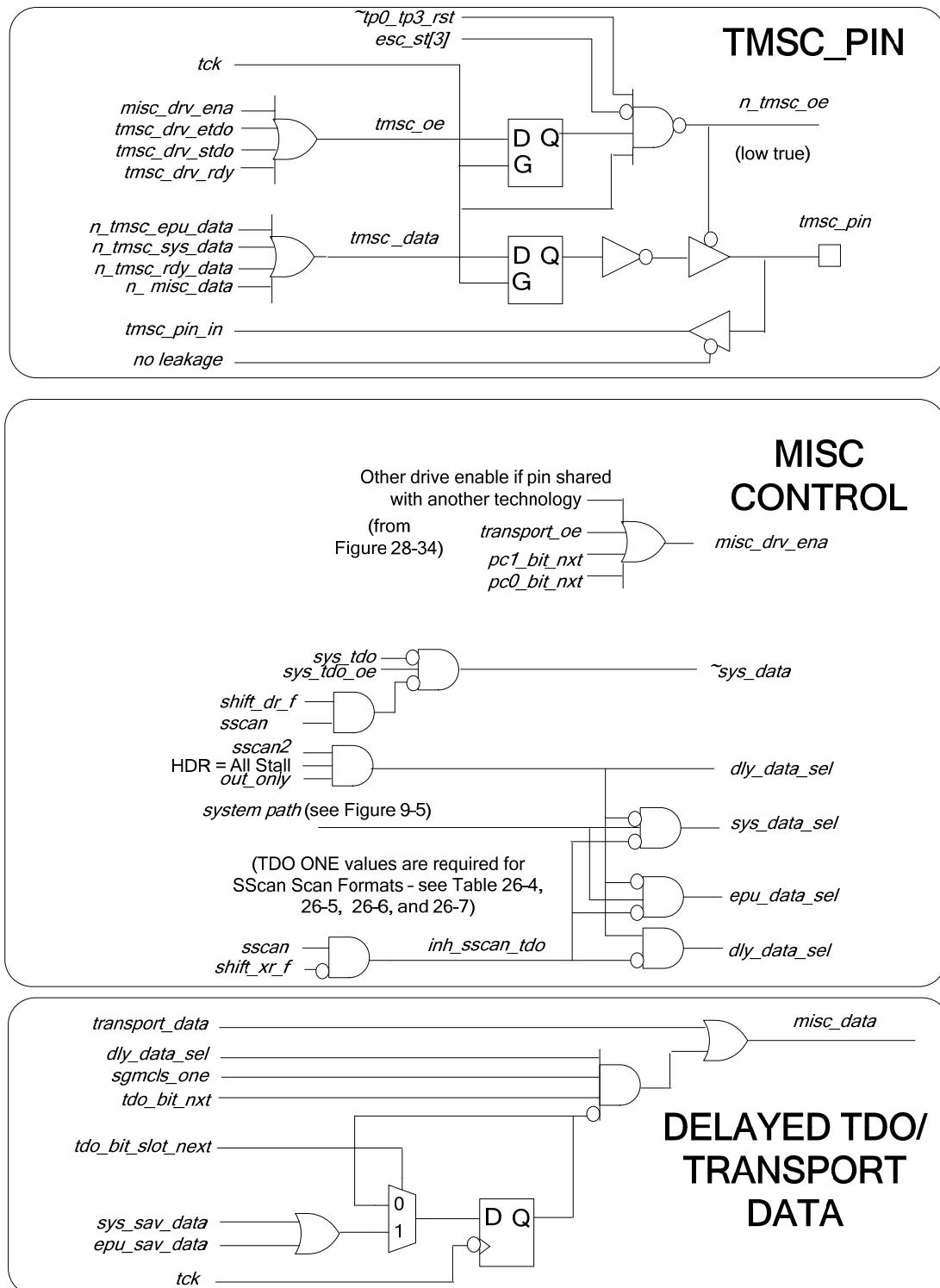
- Miscellaneous control signals used elsewhere in these figures
- The generation of TDO data when the SScan2 and SScan3 Scan Formats are implemented and output-only transfers are used with these scan formats
- Merge of transport data and data supplied by another technology

Figure 14-13 shows the following:

- RDY Bit Drive Policies and RDY data generation
- TDO Bit Drive Policies when the CLTAPC supplies the TAP.7 Scan Path
- TDO Bit Drive Policies when the EPU supplies the TAP.7 Scan Path
- TDO data generation in the absence of the use of SScan2 and SScan3 Scan Formats with output-only transfers

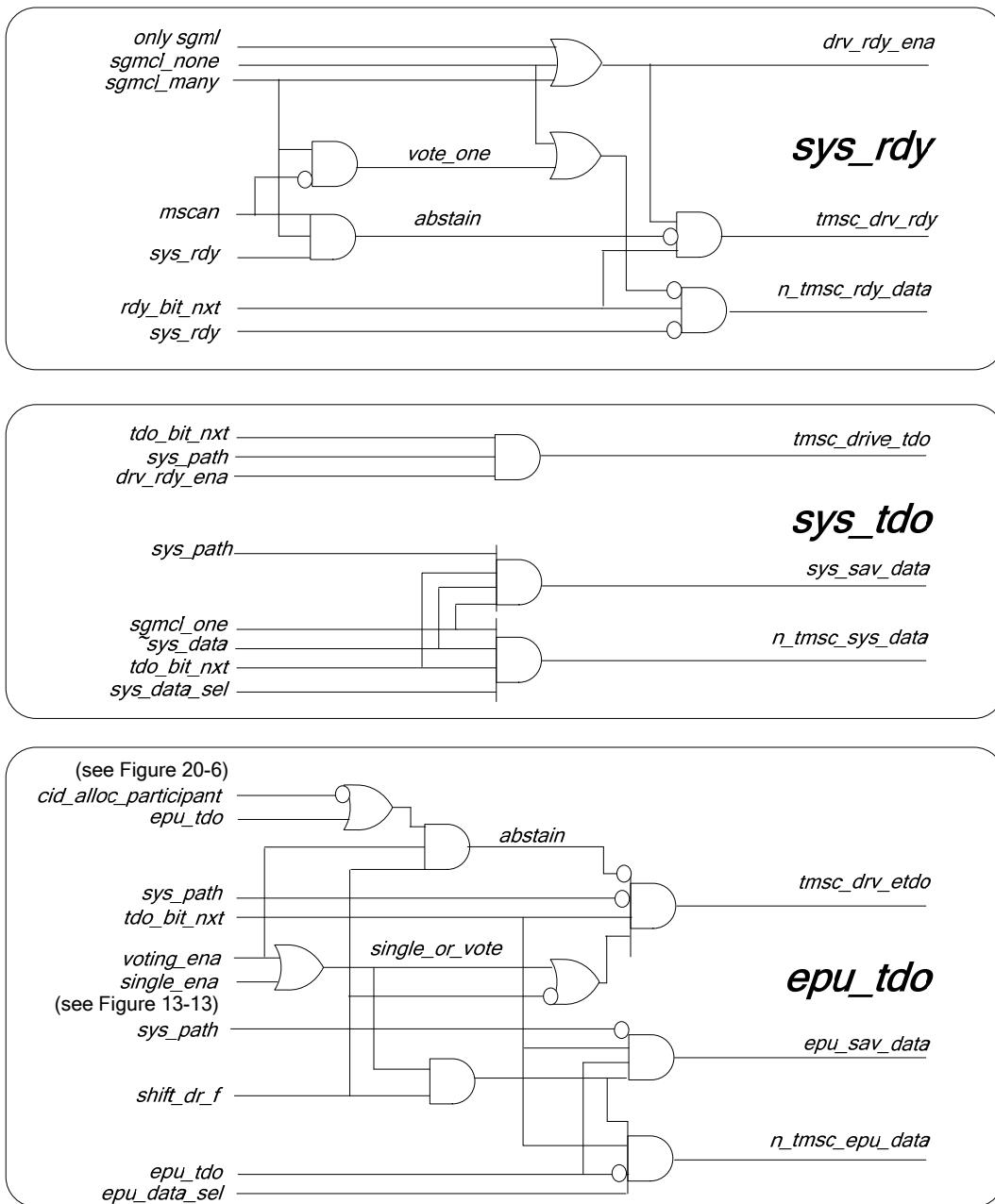
**Table 14-5 — Miscellaneous signal name definitions**

<b>Signal</b>	<b>Description</b>
<i>ctl_many</i>	More than one Conditional Group Member during the last EPU TCK period
<i>ctl_none</i>	No Conditional Group Member during the last EPU TCK period
<i>ctl_one</i>	One Conditional Group Member during the last EPU TCK period
<i>sgmcl_many</i>	More than one Scan Group Member during the last EPU TCK period
<i>sgmcl_none</i>	No Scan Group Members during the last EPU TCK period
<i>sgmcl_one</i>	One Scan Group Member during the last EPU TCK period
<i>only_sgm_last</i>	The only Scan Group Member during the last EPU TCK period
<i>only_cgm</i>	The EPU is the only Conditional Group Member.
<i>inh_sscan_tdo</i>	Force the TDO data value to a logic 1 during SScan non- <i>Shift-xR</i> states.
<i>sscan2r3_out</i>	SSCAN output-only operation



**Figure 14-12 — Conceptual view of TMSC signal output and miscellaneous support logic**

More information on the Delay of TDO data shown in Figure 14-12 may be found in the description of SScan2 and SScan3 Scan Formats in 26.5.1.4.3.



See Table 13-8 for definitions of: *sgmcl\_none*, *sgmcl\_one* and *sgmcl\_many*.

**Figure 14-13 — Conceptual view of TMSC data and drive enables**

Data for TDO bits is provided by three sources, the scan path provided by the STL to the CLTAPC, the Control Path, and a delayed version of these paths. The terms affecting the selection of delayed data are shown in Table 14-6.

**Table 14-6 — TDO bit data source selection terms**

<b>SScan2?</b> <small>(TAP Scan &amp; Oscan)</small>	<b>HDR ==110</b>	<b>1<sup>st</sup> TDO bit of segment has occurred?</b>	<b>sys_path</b>	<b>sys_data_sel</b>	<b>epu_data_sel</b>	<b>dly_data_sel</b>
0	x	0	0	0	1	0
0	x	0	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	1	1	x	0	0	1

## 14.11 Programming considerations

Note that TDO data within an SP is valid as follows:

- For the System Path, provided:
  - The STL enables the output of TDO data.
  - The STL is the only Scan Group Member.
  - There is a TDO bit in the SP.
  - Any of the following are true:
    - The scan format is MScan or Oscan.
    - The TAPC state associated with the SP is *Shift-xR*.
- For the Control Path, provided:
  - The EPU is the only Conditional Group Member.
  - There is a TDO bit in the SP.
  - Any of the following are true:
    - The Scan Format is MScan or Oscan.
    - The TAPC state associated with the SP is *Shift-xR*.

RDY bits that are continuously a logic 0 constitute a hang condition. This situation can be created by one of the following three ways:

- A system error that causes the extended not ready condition.
- The SGML state indicating one Scan Group Member the last EPU TCK period when there are really none (this is the result of a programming error).
- All TAP.7 Controllers are placed Offline while using an SP that contains RDY bits.

A system error may be caused by either the DTS or IP within the STL. A Reset Escape or the active state of nTRST or nTRST\_PD is used to remedy this situation to the point where the DTS and TS communicate properly with the use of the Standard Protocol or the use the Advanced Protocol without the use of RDY bits. However, this does not cure the root cause of the problem. Other action may be necessary to correct the condition causing the ready hang. The following case may generate this situation and be remedied without a reset of the ADTAPC.

With all ADTAPCs of a branch Offline and using OScan or SScan Scan Formats with RDY bits, it is possible to create a ready hang as follows. When there are no TMSC drive sources during output bit-frames, the TMSC keeper function creates an RDY bit value that is the same as the last bit of an input frame preceding the RDY bit. An input frame with its last bit a logic 0 therefore creates a ready hang.

There are two approaches to resolving this hang condition. With the first, the DTS controller can be initialized to use the Standard Protocol, and subsequently Advanced Protocol, without RDY bits in the SPs. A Selection Sequence resynchronizes operation with the Offline TAP.7 Controllers. With the second approach, a DTS may be designed to force a ready condition either internally or at the TMSC signal. In other words, the TMSC signal can be driven continuously as a logic 1 until a Selection Sequence is initiated (the use of the Control Protocol begins). This is acceptable as all TAP.7 Controllers are Offline, with there being no TMSC drive candidates.

The case with all ADTAPCs of a branch Offline can be detected in most cases before it causes an RDY hang condition by first using the MScan Scan Format to check that a selected TAP.7 Controller is Online with a scan operation while the command window is open and then using the desired scan format.

## 15. IEEE 1149.1-compliance concepts

### 15.1 Introduction

This clause is applicable to T0 above TAP.7s. It extends the high-level description of the concepts used with T0 TAP.7s provided by Clauses 4–8. It provides background information on a standardized multi-TAPC architecture dealing with multiple TAPCs on the same chip. It describes the differences in the views of a chip required by test and debug. It supplements the high-level description of T0 TAP.7s found in Clause 5.

The subject matter within this clause is organized in the following manner:

- 15.2 - Background
- 15.3 - Test and debug views of a system of interest
- 15.4 - An approach to implementing EMTAPC selection/deselection

### 15.2 Background

Modern design technologies allow the integration of a wide variety of functions on a single die. With the constant time-to-market pressure, a large number of chip designers and integrators adopt re-use of existing designs as the only viable option to design a complex system chip in a reasonable amount of time. Due to the large-scale re-use of existing IP modules, current and future system chip designs do and will contain multiple TAPCs, to control all kinds of functionality, from boundary scan to internal debug control.

Had such a system chip been designed as a printed-circuit board, access to these TAPCs would have been provided by connecting them in series to create a single, long, serial scan chain, and the programming of all TAPCs would take place simultaneously. This approach is currently used for production testing. Using the board approach within system chips is problematic because the implementation of a single serial chain of TAPCs inside a single chip results in a component that does not conform to IEEE Std 1149.1. This approach violates the rules imposed by IEEE Std 1149.1 on the length of the Data Register Scan Path for a specific and standardized set of instructions. This is the case for the *BYPASS* and *IDCODE* instructions.

The IEEE 1149.1 rules are in place so that at the board level, all chips behave in the same, standard way. When chips are managed with a well-defined set of rules followed by all suppliers, board testing becomes easier, with the industry developing the infrastructure needed to support end-equipment manufacturing. With the IEEE 1149.1 rules, board designers can use a number of chips, each with IEEE 1149.1-Specified Behavior, from all suppliers on a single board. Chips with 1149.1-Specified Behavior are highly desirable. Chips that provide IEEE 1149.1-Specified Behavior, IEEE 1149.7-Specified Behavior, or both behaviors should therefore be considered essential.

At a minimum, the following five rules and permissions from IEEE Std 1149.1 have to be addressed to enable the delivery of a IEEE 1149.1-Specified Behavior while having multiple, on-chip TAPCs. A component shall have the following attributes:

- Only one active TAP.
- A *BYPASS* Register with a length of exactly one bit.
- An optional Device ID Register with a length of exactly 32 bits.
- An optional nTRST signal that, when activated, resets all TAPC logic to the defined *Test-Logic-Reset* state.
- A BSDL file describing the implementation details.

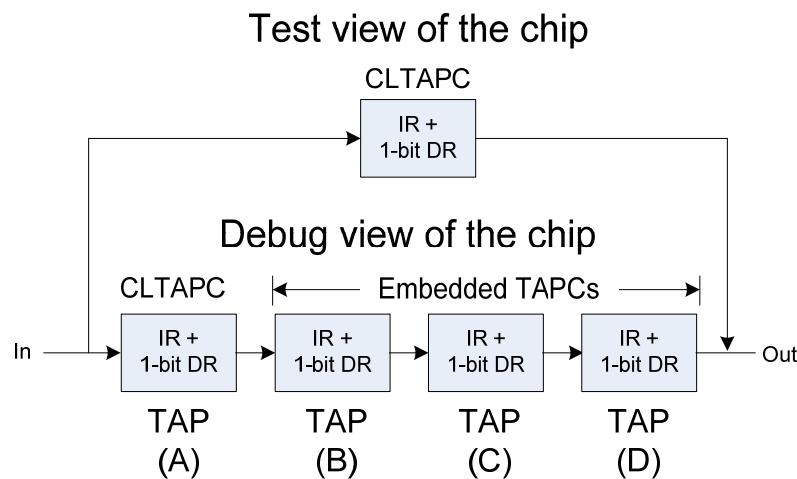
These rules provide a test view of a chip that has been essential for enabling the manufacturing test of products with the chip. With this standard, the last four attributes in the above list are created by a test reset. The optional Test Reset Signal may be either nTRST or nTRST\_PD. Note that the TAP.7 BSDL has been enhanced to allow multiple TAPs on the same component.

A chip's Test Access Port has increasingly become a debug portal, providing access to chip facilities that support applications debug and system integration. Many existing chips requiring applications debug provide access to multiple on-chip TAPCs through a serial path. It can be expected that future generations of chips requiring applications debug will also provide access to multiple TAPCs in the same way. This is based on the assumption that, for controlling a complex digital system, there should be no difference to the DTS software whether the system is implemented as a printed circuit board (during rapid silicon prototyping), as multiple dies in a package, or as a single chip (after production). In other words, all these implementations should be controllable in the same way, using the same software, and with minimal if no modifications.

While the test facilities concentrate on the interconnection of multiple components on a board, debug concentrates on the activity of components on chips and the information that travels between them. This concept is also extended to boards. Software drivers generally are written to control specific components such as CPUs, data acquisition units, or trace logic.

### 15.3 Test and debug views of a system of interest

In the general case, board manufacturing test does not require the use of the system components that are of interest to debug. Likewise, debug does not require the use of the interconnection between the chips required by board manufacturing test. Additionally, the manufacturing test is generally completed before applications debug begins, with test and debug utilizing different tools and DTS software. This means test and debug utilize two different views of a system of interest. These views are very different views of the same logic, as shown in Figure 15-1. T0 and above TAP.7s make these two views available when they are needed.



**Figure 15-1 — Inclusion and exclusion of the STL from the scan path**

The test view requires the CLTAPC provide IEEE 1149.1-Specified Behavior following a test reset that is generated either asynchronously or with the *Test-Logic-Reset* state. Subsequent to this operation, an IEEE 1149.1 private instruction or an IEEE 1149.7 custom command may be used to activate a debug view that provides access to the EMTAPCs. This view provides access to the EMTAPCs within a chip as though they were in a chip on a board, or in another manner as defined by this specification.

The test view is given priority over the debug view as the *Test-Logic-Reset* state creates this view. The CLTAPC operates with this view until an action is initiated via the CLTAPC to change the view. The test view may also be changed to the debug view using an instruction with a T0 and above TAP.7 or a custom command with a T1 and above TAP.7. With the debug view, the EMTAPCs are accessed in the manner specified by the chip vendor, in most cases, and the EMTAPC controls a portion of the component but does not have boundary scan associated with it. It may or may not have a Device Identification Code value.

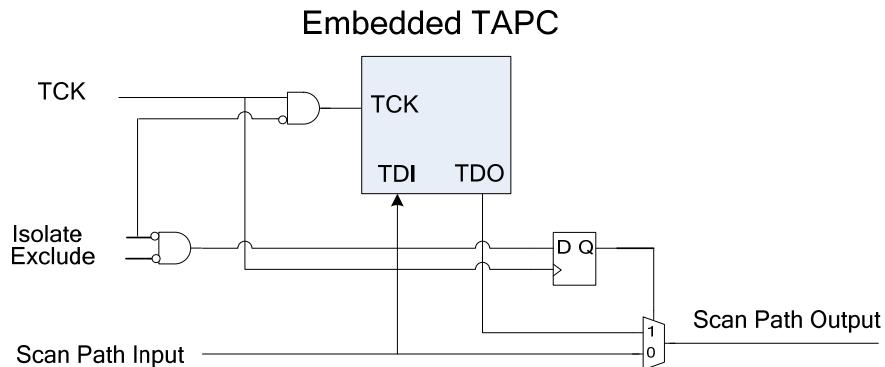
Embedded TAPs may be selected, deselected, and accessed depending on their availability (some may be powered-down or otherwise inaccessible). Changes in the selection and deselection of EMTAPCs occur upon entry into the *Run-Test/Idle* state. This allows the synchronization of actions. An example of synchronizing actions is the running or halting of applications processors attached to different TAPs with a chip or different chips within a system.

#### **15.4 An approach to implementing EMTAPC selection/deselection**

A conceptual view of a mechanism that may be used to control the operation of an EMTAPC is shown in Figure 15-2. The operation of the TAPC within this figure is governed by Table 15-1. The methods used to generate the isolate and exclude signals are covered by the standard with a set of guidelines that accommodate the behavior of proprietary implementations that predate the standard.

**Table 15-1 — EMTAPC operation**

Isolate	Exclude	TAPC State Machine state advances	In scan path	Selected
0	0	Yes	Yes	Yes
0	1	Yes	No	No
1	1	No	No	No



NOTE—This figure is conceptual in nature and is not intended to represent the desired approach to implementing the EMTAPC scan selection states.

**Figure 15-2 — Conceptual view of the operation of an EMTAPC**

When an EMTAPC scan selection state is isolated, the TAPC State Machine state is parked in a predefined state, usually *Run-Test/Idle* but possibly *Test-Logic-Reset*, while the CLTAPC State Machine state may continue to advance.

## 16. T0 TAP.7

### 16.1 Introduction

This clause is applicable to T0 and above TAP.7s. It extends the high-level description of the T0 TAP.7 provided in Clause 5. It provides the rules, permissions, and recommendations for the implementation of a standardized multi-TAPC architecture that is controlled through a single IEEE 1149.1 TAP. It describes the chip's TAP scan architecture characteristics necessary to provide IEEE 1149.1-Specified Behavior even when the chip has multiple TAPCs. It also provides programming considerations.

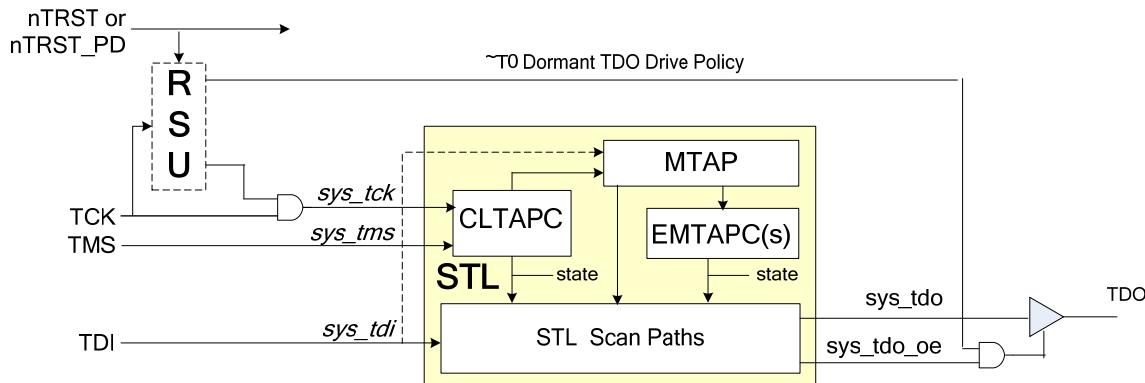
The IEEE 1149.7 multi-TAPC architecture in a class T0 TAP.7 chip provides IEEE 1149.1-Specified Behavior following the *Test-Logic Reset* state. This feature is a key enabler for providing both test and applications debug support.

The subject matter within this clause is organized in the following manner:

- 16.2 Deployment
- 16.3 Capabilities
- 16.4 Configurations
- 16.5 Start-up behavior
- 16.6 Supporting multiple on-chip TAPCs
- 16.7 Controlling the selection state of EMTAPCs
- 16.8 Control via the CLTAPC Instruction Register
- 16.9 Control via one or more CLTAPC Data Register(s)
- 16.10 Control via internal or external *tapc\_select* signals
- 16.11 Example use cases
- 16.12 Identification of on-chip TAP controllers(s)
- 16.13 Multiple dies in one package
- 16.14 Managing STL Group Membership
- 16.15 RSU operation
- 16.16 Programming considerations

### 16.2 Deployment

A high-level block diagram of the deployed T0 TAP.7 is shown in Figure 16-1.



**Figure 16-1 — Deployment of the T0 TAP.7**

### 16.3 Capabilities

A T0 TAP.7 provides both IEEE 1149.1-Specified Behavior and IEEE 1149.7-Specified Behavior following the *Test-Logic-Reset* state. It does this independently of whether EMTAPCs are deployed on a chip. Following the *Test-Logic-Reset* state, the CLTAPC and its associated scan paths exhibit the characteristics specified by IEEE Std 1149.1. Subsequently, a private instruction may modify this behavior, providing IEEE 1149.1-Non-disruptive Behavior and IEEE 1149.7-Specified Behavior.

Implementation of the RSU is an option with a T0 TAP.7. When the RSU is implemented, a T0 TAP.7 may be operated in the Series Branch of a multi-branch Scan Topology (with other TAP.7 Technology Branches and non-TAP.7 Technology Branches).

### 16.4 Configurations

#### 16.4.1 Description

The RSU function may be added to a T0 TAP.7 as an option. The following functions may also be implemented with the RSU function:

- Selection Alerts
- Deselection Alerts

#### 16.4.2 Specifications

##### Rules

- a) Each subsequent specification in 16.4.2 shall apply to a T0 TAP.7.

##### Permissions

- b) An RSU may be implemented.
- c) The following RSU options may be implemented:
  - 1) Selection Alerts.
  - 2) Deselection Alerts.

## 16.5 Start-up behavior

### 16.5.1 Description

The start-up behavior of a T0 TAP.7 is IEEE 1149.1-Compliant, with this start-up option the only start-up option that may be implemented. The start-up options and the behavior they create are described in 10.3.

### 16.5.2 Specifications

#### Rules

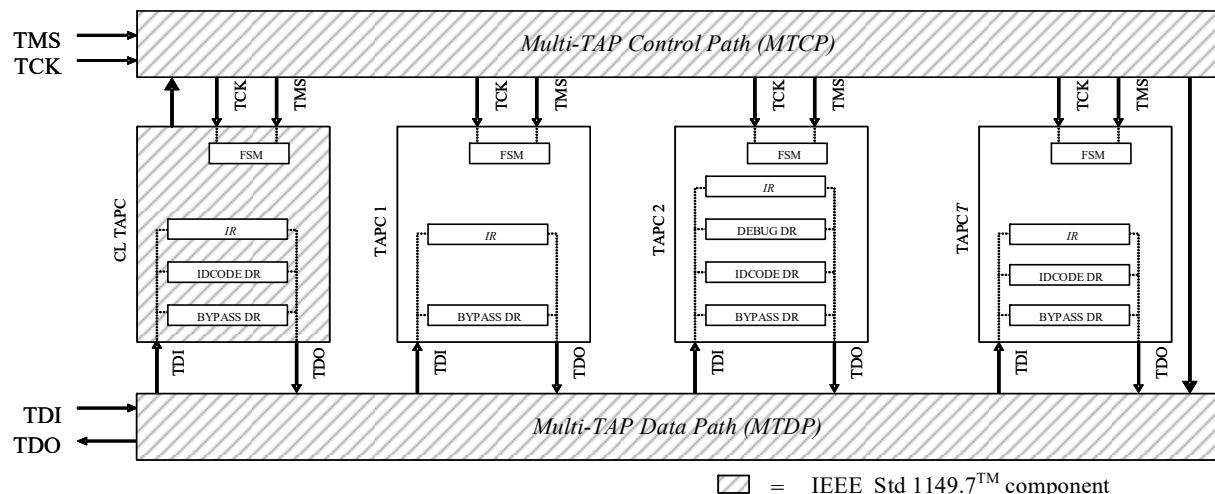
- a) Each subsequent specification in 16.5.2 shall only apply to a T0 TAP.7.
- b) Start-up behavior shall be that specified by the IEEE 1149.1-Compliant start-up option.

## 16.6 Supporting multiple on-chip TAPCs

Applications debug tools vendors already support access to TAPCs on a printed circuit board and through TAP-linking modules. As integration has driven these scan topologies on chip, maintaining the test view mandated by IEEE Std 1149.1 while still enabling access to the on-chip TAPCs is a necessity.

With a chip-scan architecture that provides IEEE 1149.1-Specified Behavior, customers of these chips can reuse their existing IEEE 1149.1 test infrastructure without making modifications. All board-level production test facilities can be reused as is. This standard requires all on-chip TAPCs used for applications debug to be accessible in an applications-debug chain. Because multiple TAPCs connected in series form a DR Scan Path that is not in compliance with IEEE 1149.7 rules, recommendations and permissions are given in this standard to create IEEE 1149.1-Specified Behavior after power-up.

An example of the IEEE 1149.7 multi-TAPC architecture for accessing multiple on-chip TAPCs is shown in Figure 16-2.



**Figure 16-2 — SoC with IEEE 1149.7 multiple TAPC architecture**

Figure 16-2 shows an IEEE 1149.1 TAPC that serves as the Chip-Level TAP controller, which is connected to other on-chip TAPCs (TAPC 1 to N), Embedded TAPCs, via a Multi-TAP Control Path (MTCP), and a Multi-TAP Data Path (MTDP). According to the definitions in 4.2.3.3.3, an EMTAPC and its associated scan paths (the Embedded TAPC complex) can operate in the following three modes: Normal, Excluded, and Isolated.

Starting from the *Test-Logic-Reset* state of the CLTAPC, the operating mode of all EMTAPCs is either Excluded or Isolated. The resulting behavior observed at the TAP chip signals is IEEE 1149.1-Specified Behavior.

An advantage of the *exclusion* of TAPCs is the total IR length is constant and the scan paths associated with each EMTAPC can be accessed with existing software tools that expect IEEE 1149.1-Specified Behavior. The external test-and-debug software does not need to cope with varying IR lengths, as any TAPC whose applications-debug DRs need not be accessed can be loaded with the IEEE 1149.1 *BYPASS* instruction. This can, however, be an issue should power-down of circuitry that contains a portion of this scan path occur.

An advantage of *isolating* the embedded TAP complex is that external test-and-debug software does not need to handle or know about an isolated EMTAPC and its associated scan paths. However, during operation, the IR length will vary depending on which EMTAPC complexes are isolated at any particular point in time. Other reasons for wanting to selectively isolate individual EMTAPC complexes from the applications debug chain are as follows:

- Power-conscious system designs may partition the chip into power domains that may include EMTAPCs. Removing power from one of these domains would cause an applications-debug chain break.
- Security-conscious system designs can require the run-time partitioning of a chip into secure domains. These secure domains may include TAPCs that were previously used to debug security algorithms running on embedded processors. Blocking access to these EMTAPCs might otherwise break the applications-debug chain.
- The correct operation of legacy-debugger software tools may require a specific, single EMTAPC to be directly connected to the chip's TAP signals.

There are four permitted methods for controlling the operating mode of an EMTAPC, as follows:

- Control via the Instruction Register managed by the CLTAPC
- Control via one or more Data Registers managed by the CLTAPC
- Control via one or more chip signals that themselves comply with the IEEE 1149.1 specifications for **Subordination of this standard within a higher level test strategy**
- Control via one or more internal application/system signals

The first two control methods listed above provide the DTS with full knowledge of the scan topology. With the third and fourth methods, the DTS will be informed as to the scan topology and it may be subject to the scan topology changing without its knowledge. Each of these methods is explained in greater detail in the following subclauses.

Henceforth, the Instruction Register managed by the CLTAPC is referred to as the “CLTAPC IR.” A Data Register managed by the CLTAPC is referred to a “CLTAPC DR.”

## 16.7 Controlling the selection state of EMTAPCs

### 16.7.1 Description

Table 16-1 provides an overview of the various methods for accessing or bypassing the scan paths associated with an EMTAPC. It lists which subclause contains more details on each particular method of control.

**Table 16-1 — Permitted methods to control the EMTAPCs' operating modes**

Control method	Subclause number
CLTAPC_IR	16.8
CLTAPC_DR	16.9
tapc_select (internal/external signals)	16.10

The permitted combinations of the methods to control EMTAPCs' operating modes are shown in Table 16-2.

**Table 16-2 — Permitted combinations of methods for controlling EMTAPCs**

Methods implemented			Permitted
CLTAPC_IR	CLTAPC_DR	External selection	
Yes	—	—	Yes
Yes	—	Yes	Yes
—	Yes	—	Yes
—	Yes	Yes	Yes
Other combinations			No

Regardless of which particular control methods are used, IEEE 1149.7-Specified Behavior provides a net result that is the same: IEEE 1149.1-Specified Behavior will be exposed to the TAP.7 after the CLTAPC has entered the *Test-Logic-Reset* state, provided the T0 TAP.7 is Online.

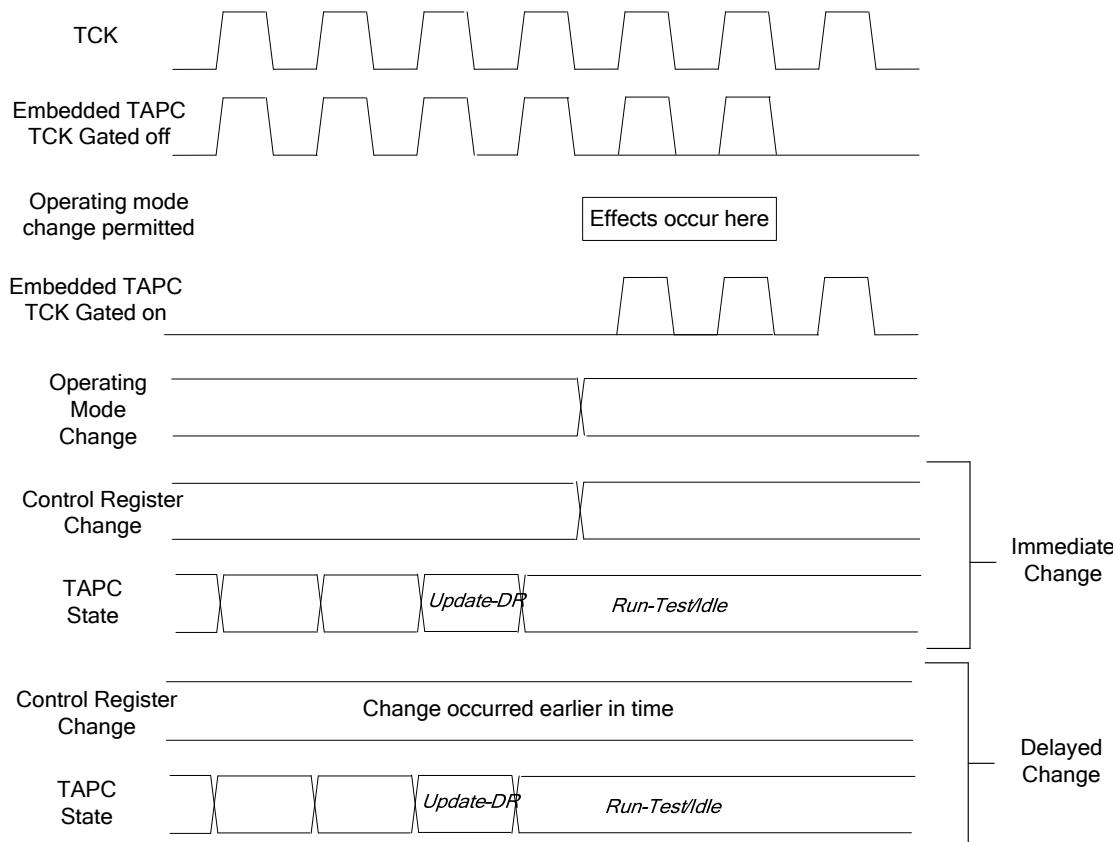
The following subclauses introduce each control method in greater detail, describe how to implement the option to select/deselect individual TAP controllers, and present each method's rules, permissions, and recommendations.

## 16.7.2 Specifications

### Rules

- a) Each subsequent specification in 16.7.2 shall apply to T0 and above TAP.7s.
- b) The CLTAPC State Machine state shall advance on each rising edge of the TCK, provided the T0 TAP.7 is Online.
- c) When the CLTAPC exits the *Test-Logic-Reset* state, the Data Register length and contents between the chip's TDI and TDO ports shall be as specified for an IEEE 1149.1 *IDCODE* instruction.
- d) All on-chip EMTAPCs used for applications debug shall only be made accessible in an applications-debug chain through one or more permitted control methods.
- e) The behavior of a TAPC in the applications-debug chain shall be governed by any of the following:
  - 1) Signals permitted by IEEE Std 1149.1.

- 2) A combination of signals permitted by IEEE Std 1149.1 and *tapc\_select* signals as permitted in Rule 16.7.2 o).
- f) The IR length of the chip shall be the sum of the lengths of the Instruction Register(s) associated with both the CLTAPC and all EMTAPCs operating in either Normal or Excluded mode.
- g) The DR length of the chip shall be the sum of the lengths of the Data Register(s) associated with both the CLTAPC and EMTAPCs operating in the Normal mode.
- h) The EMTAPC operating in Normal mode shall run in lock-step with the CLTAPC.
- i) The EMTAPC operating in Excluded mode shall run in lock-step with the CLTAPC.
- j) The state progression of an EMTAPC operating in Isolated mode shall be parked in either the *Run-Test/Idle* or *Test-Logic-Reset* state, while the state progression of the CLTAPC continues.
- k) The operating mode (Normal, Excluded, and Isolated) of all EMTAPCs shall be under control of only one of the permitted combinations of methods to control the EMTAPCs' operating modes shown in Table 16-2 at a time.
- l) Changes to the operating mode (Normal, Excluded, and Isolated mode) of any EMTAPC shall only take effect upon entering the *Run-Test/Idle* or *Test-Logic-Reset* state.
- m) When a EMTAPC operating mode is changed from isolated to Normal mode, the Multi-TAP Control Logic shall not require that the CLTAPC remains in the *Run-Test/Idle* state for more than two TCK cycles (as seen by the CLTAPC) to complete this change (as shown in Figure 16-3).



**Figure 16-3 — Operating mode changes/*Run-Test/Idle* state relationships**

## Permissions

- n) An EMTAPC control mechanism may be implemented in a manner that register(s) managed by the CLTAPC control the operating mode for all EMTAPCs subject to management by this control mechanism.
  - o) In addition to either controlling the operating mode of EMTAPCs with CLTAPC\_IR or CLTAPC\_DR control methods, the operating mode of EMTAPCs may also be controlled from external *tapc\_select* chip signals or internal *tapc\_select* signals.

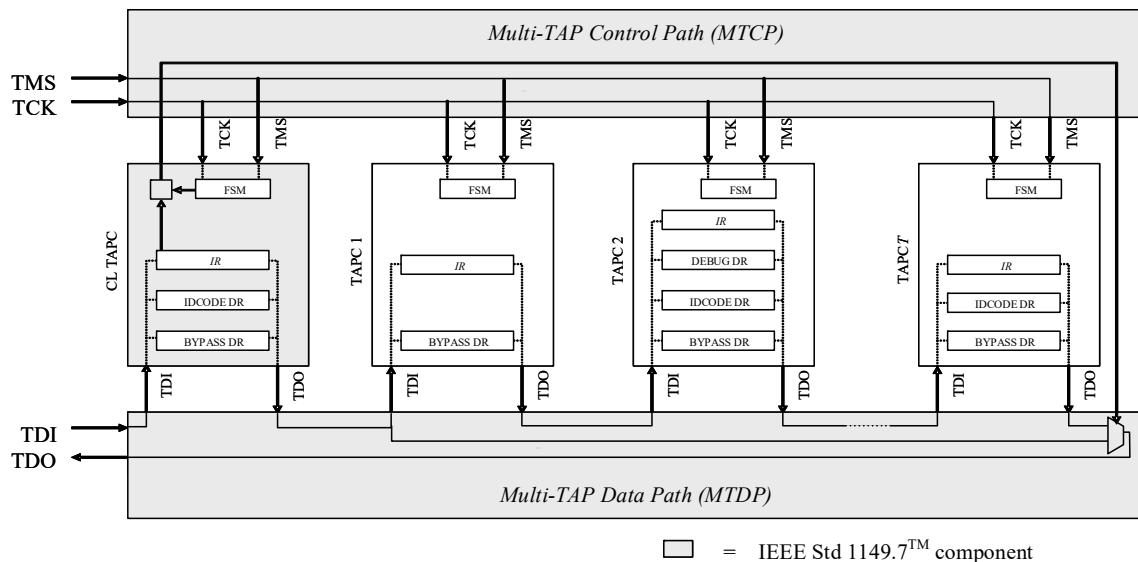
## **16.8 Control via the CLTAPC Instruction Register**

### **16.8.1 Description**

#### **16.8.1.1 Exclusion of TAPCs**

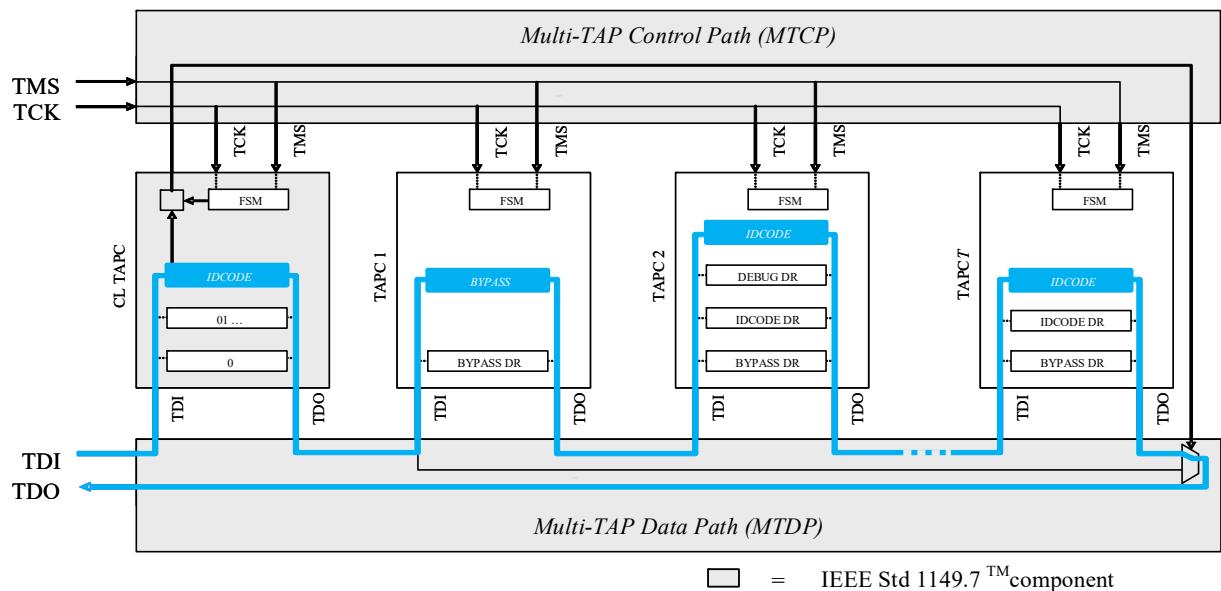
The CLTAPC IR may be used to control the operating mode of the EMTAPCs. Figure 16-4 shows an example CLTAPC with scan paths it manages connected in series with other on-chip TAPCs. In this example, the MTDP consists of one bypass multiplexer (shown in the bottom right of Figure 16-4) that is under the control of the CLTAPC IR. When the CLTAPC leaves its *Test-Logic-Reset* state and enters the *Run-Test/Idle* state, the operating mode of EMTAPC 1 to N is changed from *Normal* to *Excluded*.

Subsequently, in the example of Figure 16-4, the selection input on the one bypass multiplexer may only be changed when the CLTAPC enters the *Run-Test/Idle* state and the CLTAPC IR has been used to specify a different operation mode prior to entry into this state.

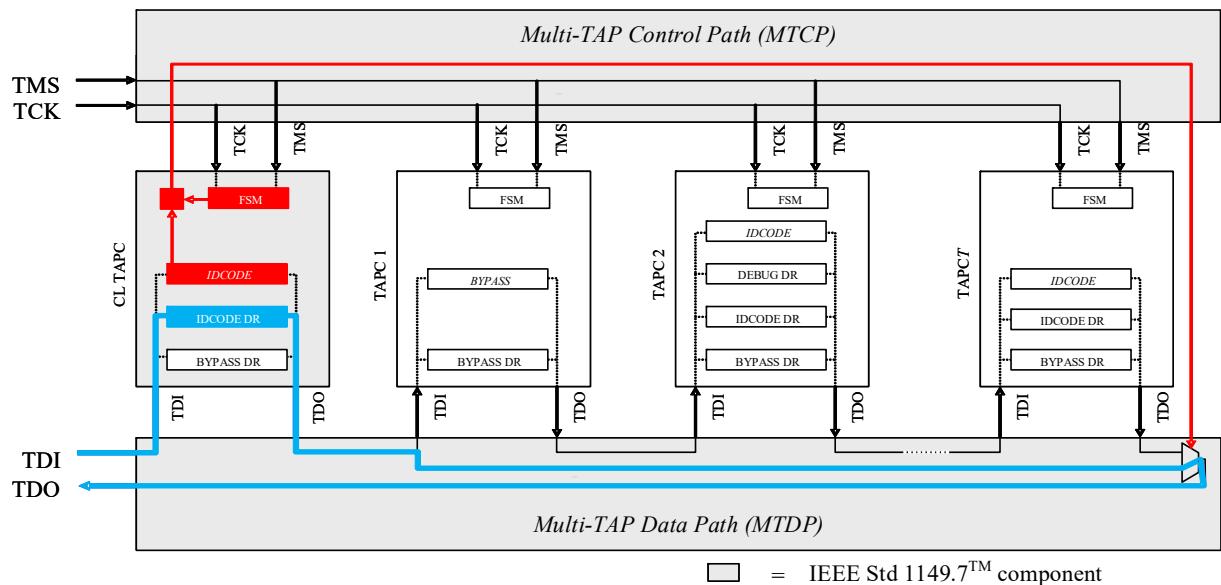


**Figure 16-4 — Example IEEE 1149.7 multi-TAP architecture with IR-controlled TAPC exclusion**

In this example, the Instruction Registers managed by the CLTAPC and EMTAPC 1 to N remain part of the IR Scan Path during IR Scans. The Data Registers of all EMTAPCs are bypassed during a DR Scan. This is shown in Figure 16-5 and Figure 16-6, respectively.

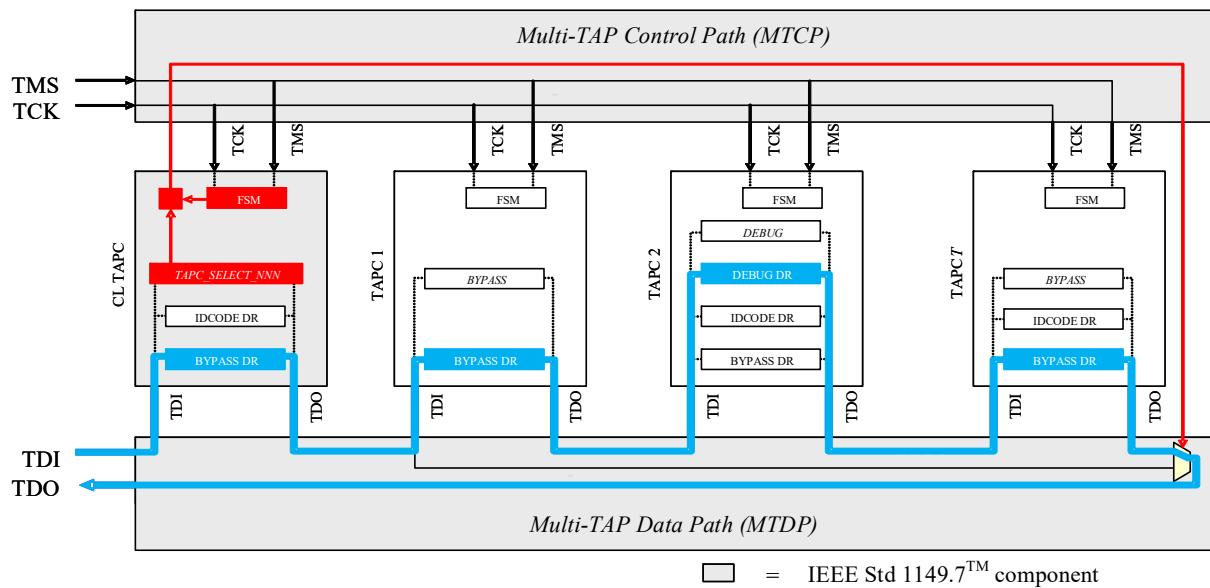


**Figure 16-5 — IR Scan Path after the CLTAPC has exited the *Test-Logic-Reset* state and entered the *Run-Test/Idle* state**



**Figure 16-6 — DR Scan Path with the *IDCODE* instruction loaded in the CLTAPC IR**

In this example, an applications-debug Data Register in TAPC 1 to N can be accessed by loading a special `TAPC_SELECT_<field>` instruction into the CLTAPC IR, which is appended with the instructions required for selecting specific DRs in TAPC 1 to N. This is shown in Figure 16-7.



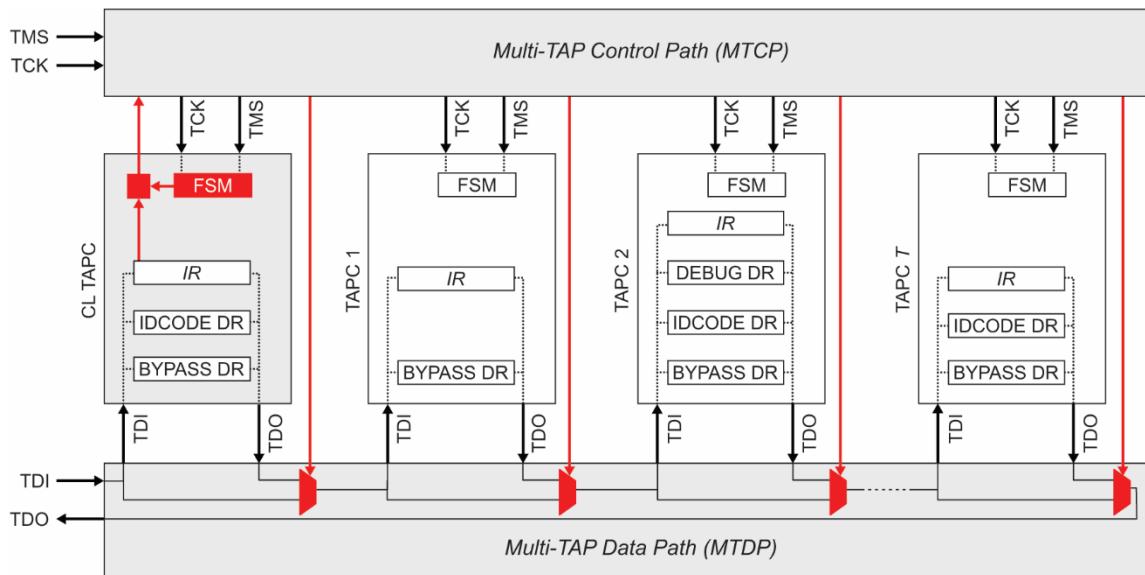
**Figure 16-7 — DR selection with the `TAPC_SELECT_NNN` instruction loaded in the CLTAPC IR**

Loading the `TAPC_SELECT_NNN` instruction (where *NNN* is described below) in the IR of the CLTAPC and subsequently entering the *Run-Test/Idle* state causes the bypass multiplexer to select the TDO output of the last TAPC in the chain, TAPC *T*, during subsequent DR Scans. The combined use of a corresponding instruction (`DEBUG`) in the IR of TAPC 2, the `TAPC_SELECT_NNN` instruction in the IR of the CLTAPC, and the `BYPASS` instruction in all other TAPCs allows access to the DEBUG DR in TAPC 2. In this case, *NNN* specifies the operating mode of each TAP controller (three in total), which is N(ormal). TAPC\_SELECT instructions convey mixes of N(ormal), I(solated), and E(xcluded) selections for any number of embedded TAP.7 Controllers.

### 16.8.1.2 Isolation of TAPCs

The CLTAPC IR control method can also place EMTAPCs in *Isolated* mode, removing them completely from the applications-debug chain, and thereby modifying the overall IR scan-path length.

As an example, a multiplexer can be placed in the Multi-TAP-Data Path for each EMTAPC, as is shown in Figure 16-8. This multiplexer allows each corresponding TAPC's IR and DRs to be bypassed during IR and DR Scans, respectively. In addition, the Multi-TAP-Control Path is extended to keep an EMTAPC's state in either *Run-Test/Idle* or *Test-Logic-Reset* while it is operating in the *Isolated mode*.



**Figure 16-8 — Example IEEE 1149.7 multi-TAP architecture to allow isolating of individual TAPCs under CLTAPC IR control**

The operating mode of TAPCs can subsequently be changed to the Isolated mode by loading a *TAPC\_SELECT\_<field>* instruction in the CLTAPC IR. A value of “N” in the mode field *<field>* of the instruction name indicates an EMTAPC that is to operate in the Normal mode, and, hence, it is accessible in the applications-debug chain. Likewise, a value of “E” in *<field>* indicates a TAPC is to operate in Excluded mode, and a value of “I” in *<field>* indicates an EMTAPC that is to operate in the Isolated mode.

#### 16.8.1.3 CLTAPC output registering of MTCP and MTDP control

The IR-controlled access mechanism closely resembles the access mechanism currently used to access individual chips on a printed circuit board with the one exception that, instead of the *BYPASS* instruction, the *TAPC\_SELECT\_<field>* instruction has to be placed in the CLTAPC IR.

Legacy IEEE 1149.1 DTS hardware may, however, have been optimized to include *BYPASS* instructions in front and after an instruction targeted at a particular EMTAPC. These *BYPASS* instructions are for the unaddressed EMTAPCs and are automatically generated in hardware as all-one bit sequences of preprogrammed lengths. This optimization is possible because all *BYPASS* instruction opcodes are all-one bit sequences. Without changes, this DTS hardware optimization mode cannot be used with a *TAPC\_SELECT\_<field>* instruction. Instead, the DTS directly controls the bit sequence. It is possible that with some DTSs, the performance of the port in this mode is lower than in the hardware mode.

This performance loss can, however, be easily avoided by registering the controlling outputs of the CLTAPC IR to the MTCP and MTDP in an Output Control Register. The purpose of this register is to keep the MTCP and MTDP in the same state even when the CLTAPC IR value is changed. This reenables the use of the *BYPASS* instructions to bypass EMTAPCs. The content of this control register is reset in the *Test-Logic-Reset* state and optionally by a dedicated CLTAPC IR value. Its reset value causes all TAPCs to operate in the Excluded or Isolated mode.

When a *TAPC\_SELECT\_<field>* instruction is loaded into the CLTAPC IR and the CLTAPC subsequently enters the *Run-Test/Idle* state, the content of this Output Control Register is updated based on the corresponding mode field *<field>*, setting the operating modes of all TAPCs. Once the operating modes of all TAPCs have been established with the value in this control register, the DTS can bypass EMTAPCs using their respective *BYPASS* instructions and operate with optimized bit sequences.

## 16.8.2 Specifications

### Rules

- a) Each subsequent specification in 16.8.2 shall apply to T0 and above TAP.7s, provided the CLTAPC\_IR method is used to control the operating mode of the EMTAPCs.
- b) One or more *TAPC\_SELECT\_<field>* instructions shall be provided in the CLTAPC to change the value of registers controlling the operating mode of TAPC 1 to N.
- c) The value of the *<field>* of the *TAPC\_SELECT\_<field>* instructions required by Rule 16.8.2 b) shall provide for all of the following:
  - 1) Specifying the EMTAPC whose operating mode is to be changed.
  - 2) Specifying all of the following operating modes for the TAPC whose operation is to be changed:
    - i) *Normal* mode.
    - ii) *Excluded* mode.
    - iii) *Isolated* mode.
- d) The *Test-Logic-Reset* state shall initialize the value of the control registers accessed by the instructions required by Rule 16.8.2 b) to a state that ensures the operation specified by Rule 16.8.2 e).
- e) When the CLTAPC transitions from the *Test-Logic-Reset* state to the *Run-Test/Idle* state, the operating mode of each EMTAPC shall be set to one of the following permitted operation modes, subject to Rule 16.10.2 c) provided the TAP\_QUERY instruction is implemented:
  - 1) Excluded.
  - 2) Isolated.
- f) The *TAPC\_SELECT\_<field>* instructions required by Rule 16.8.2 b) shall utilize the BYPASS DR Scan Path in the CLTAPC for DR Scans.
- g) When the value of the control register specifying the operating mode of the EMTAPCs is changed by a control command, the effect shall persist until any of the following occurs:
  - 1) Another control command changes the value of the register.
  - 2) The CLTAPC *Test-Logic-Reset* state initializes the control register value.

### Permissions

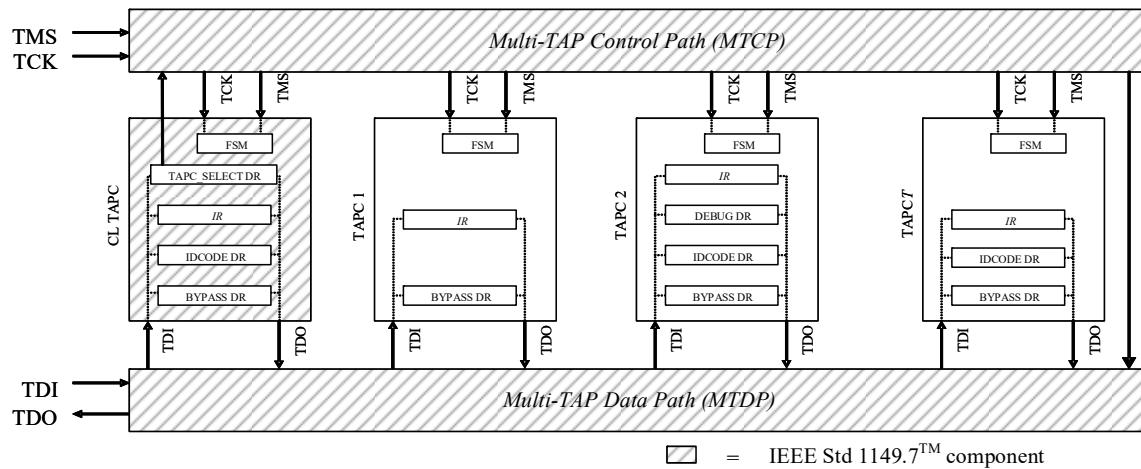
- h) The *Test-Logic-Reset* state may set the operating mode of each EMTAPC to one of the following operating modes:
  - 1) Excluded.
  - 2) Isolated.

## 16.9 Control via one or more CLTAPC Data Registers

### 16.9.1 Description

One or more CLTAPC DRs may be used to control the operating modes of the EMTAPCs. Figure 16-9 shows embedded TAPC complexes connected in series, but the operating mode of EMTAPC 1 to N can be changed from *Normal* to *Excluded* or *Isolated* (and vice versa). In this example architecture, the MTDP

consists of one or more bypass multiplexers that are under the control of one or more Data Registers selected by TAPC\_SELECT instructions associated with the CLTAPC IR.



**Figure 16-9 — Example IEEE 1149.7 multi-TAP architecture with DR-controlled TAPC exclusion or isolation**

In this example, when the CLTAPC exits the *Test-Logic-Reset* state and enters the *Run-Test/Idle* state, the TAPC\_SELECT Data Registers managed with the CLTAPC IR are reset to a value that sets the operating modes of the EMTAPC 1 to N to *Excluded* or *Isolated*. It thereby exposes an IEEE DR Scan Path to the TAP with this scan path having the behavior that is specified by IEEE Std 1149.1.

Subsequent to this, the values of the control registers may be changed with DR Scans. These changes do not affect the operating modes of the EMTAPC 1 to T until the *Run-Test/Idle* state is reached.

## 16.9.2 Specifications

### Rules

- Each subsequent specification in 16.9.2 shall apply to T0 and above TAP.7s, provided the CLTAPC\_DR method is used to control the operating mode of the EMTAPCs.
- One or more instructions shall be provided in the CLTAPC IR to access the TAPC\_SELECT DR(s) controlling the operating mode of the EMTAPCs.
- The value of the control registers accessed by the instructions required by Rule 16.9.2 b) shall provide for all of the following:
  - Specifying the EMTAPC whose operating mode is to be changed.
  - Specifying all of the following operating modes for the EMTAPCs whose operation is to be changed:
    - Normal* mode.
    - Excluded* mode.
    - Isolated* mode.
- The *Test-Logic-Reset* state shall initialize the value of the control registers accessed by the instructions required by Rule 16.9.2 b) to a state that ensures the operation specified by Rule 16.9.2 e).

- e) When the CLTAPC state transitions from the *Test-Logic-Reset* state to the *Run-Test/Idle* state, the operating mode of each EMTAPC that is controlled by the CLTAPC\_DR method shall be set to one of the following operating modes, which is subject to Rule 16.10.2 c) provided the TAP\_QUERY instruction is implemented:
  - 1) Excluded.
  - 2) Isolated.
- f) When the CLTAPC state transitions from the *Test-Logic-Reset* state to the *Run-Test/Idle* state, the content of the TAPC\_SELECT DRs shall cause the operating mode of the TAPC 1 to T to be changed to either *Excluded* or *Isolated*.

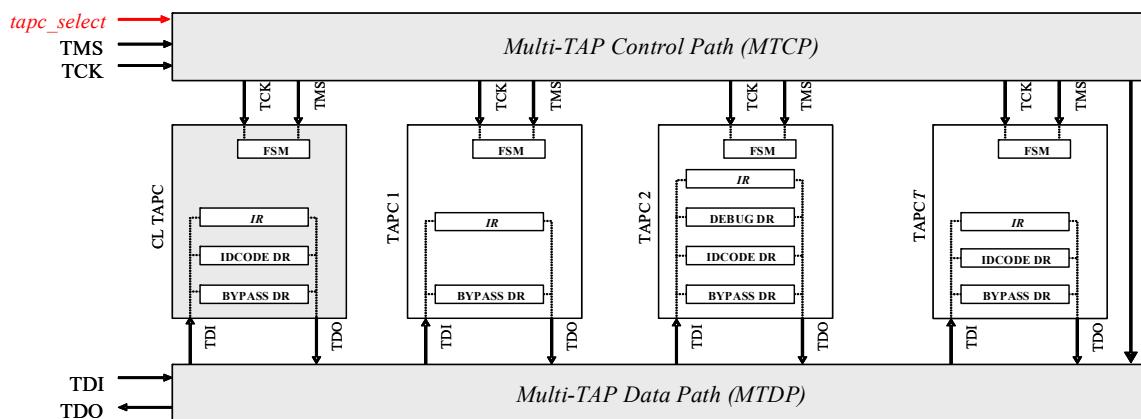
## Permissions

- g) The *Test-Logic-Reset* state may set the operating modes of each EMTAPC to one of the following permitting operating modes:
  - 1) Excluded.
  - 2) Isolated.

## 16.10 Control via internal or external *tapc\_select* signals

### 16.10.1 Description

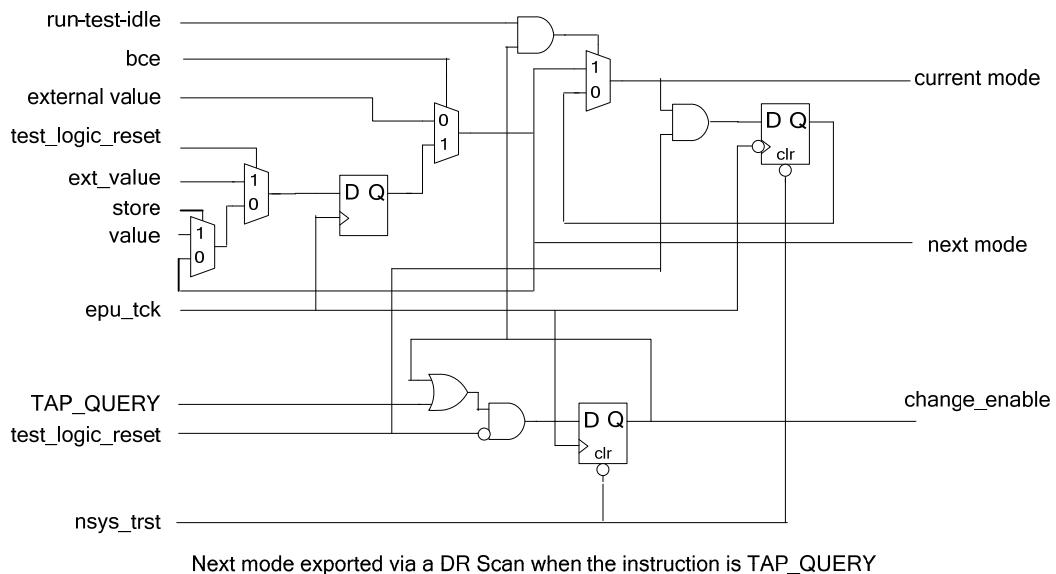
In addition to using either the IR- or DR-control methods described in 16.8 and 16.9, the operating mode of TAP controllers may also be controlled using internal chip or external chip signals, which are called *tapc\_select* signals. An example architecture that uses this method of control is shown in Figure 16-10. Internal *tapc\_select* signals may be derived from an on-chip application or system component. The external *tapc\_select* device signals are available on-chip signals and will conform to the IEEE 1149.1 specifications for **Subordination of this standard within a higher level test strategy** regarding boundary compliance enable (BCE) signals.



**Figure 16-10 — Example IEEE 1149.7 multi-TAP architecture to allow control over the operating mode of individual TAPCs by using *tapc\_select* chip signals**

Following the *Test-Logic-Reset* state, the operating mode of the EMTAPCs is governed by Rule 16.8.2 e) and Rule 16.9.2 e). The internal or external *tapc\_select* signals may initialize the control registers specifying the value used when the *Run-Test/Idle* state is reached. These registers will be capable of being read before their value is used. The value of these registers cannot affect the operating mode of the EMTAPCs until the DTS knows what the effect will be before it occurs. In this case, the use of these control registers for defining the operating mode of the EMTAPCs is inhibited until the value of these control registers is read by the DTS. This is done via a TAP\_QUERY instruction. Following this

instruction, the occurrence of the *Run-Test/Idle* state utilizes the value of the control registers to establish the operating modes of the EMTAPCs. One approach to how this may be done is shown in Figure 16-11. Other approaches should be considered.



**Figure 16-11 — Conceptual view of the TAP\_QUERY instruction**

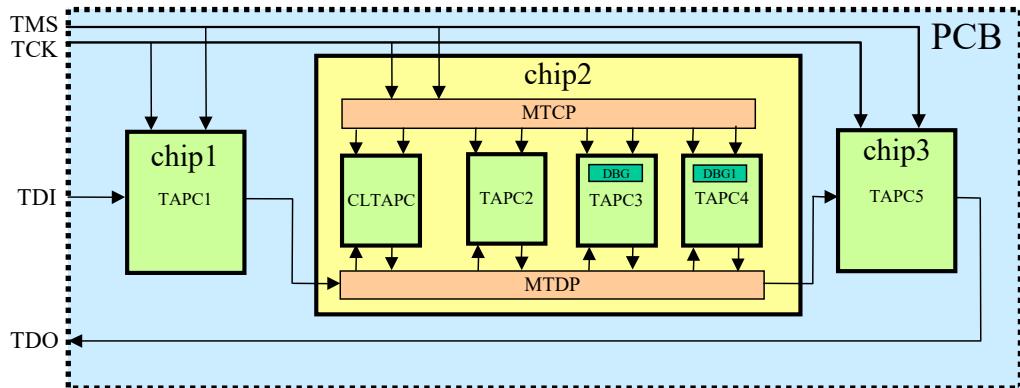
## 16.10.2 Specifications

### Rules

- Each subsequent specification in 16.10.2 shall apply to T0 and above TAP.7s when one or more *tapc\_select* signals are used to control the operating mode of EMTAPCs.
- When one or more *tapc\_select* signals are used to control the operating mode of EMTAPCs, a TAP\_QUERY instruction with an associated DR shall be provided by the CLTAPC to query all of the following:
  - The EMTAPC (1 to N), whose pending operating mode is being read
  - Operating modes of the EMTAPC
    - Normal.
    - Excluded.
    - Isolated.
- Following the *Test-Logic-Reset* state, the operating mode generated by the following rules and permissions shall persist until the *Update-DR* state is reached when the instruction is TAP\_QUERY:
  - Rule 16.8.2 e).
  - Rule 16.9.2 e).
  - Permission 16.8.2 h).
  - Permission 16.9.2 g).
- The behavior associated with external *tapc\_select* chip selection signals shall comply with the IEEE 1149.1 specifications for **Subordination of this standard within a higher level test strategy** regarding BCE chip signals.

## 16.11 Example use cases

This subclause shows example IR and DR Scans for accessing individual TAP controller registers using the control methods presented in 16.8 and 16.9. The example use case in Figure 16-12 shows TAP control for a PCB with three chips: a Chip1 with a single embedded TAP controller (TAPC1), a Chip2 with a CLTAPC and three embedded TAP controllers (TAPC2, TAPC3, and TAPC4), and a Chip3 with a single embedded TAP controller (TAPC5).



**Figure 16-12 — Example system**

The nomenclature shown in Table 16-3 is used in the description.

**Table 16-3 — Nomenclature used in description**

Term	Description
IRshift < $i_{n-1} \dots i_0$ >	Shift binary content $i_{n-1} \dots i_0$ into the IR Scan Path, with bit $i_0$ as the first bit to be applied to TDI
DRshift < $d_{n-1} \dots d_0$ >	Shift binary content $d_{n-1} \dots d_0$ into the DR Scan Path, with bit $d_0$ as the first bit to be applied to TDI
TAPCx.bp	Bypass IR Code (all ones).
TAPCx.DBGx	IR Code to select DBGx Data Register.
TAPCx.DBGx_data	Data for DBGx Data Register.
<b>For control via the CLTAPC Instruction Register</b>	
CLTAPC.TAPC_SELECT_<field>	TAPC SELECT IR Code with TAPC operating mode field (denoting Normal, Excluded, or Isolated).
<b>For control via a CLTAPC Data Register</b>	
CLTAPC.TAPC_SELECT	TAPC SELECT IR Code.
CLTAPC.TAPC_SELECT_<field>	TAPC_SELECT DR contents for controlling the operating modes of TAPCs 2, 3, and 4 as specified in <field>.
NOTE—The <field> designation shown above represents the remainder of the instruction name.	

The following assumptions are made:

- Four-bit Instruction Registers
- An eight-bit Instruction Register in the CLTAPC when the selection field is also part of the Instruction Registers
- Three bits for the mode operation field <field> and Data Register (assuming one bit per mode encoding (one-hot encoding), so either TAPC exclusion or isolation, but not both, are used)
- 32-bit DBG and DBG1 Data Registers

The following IR and DR Scan sequences are required to access a debug register (DBG) in TAPC3 followed by access to the DBG1 Register in TAPC4. The approximate number of TCK cycles is also given for the different cases.

### 16.11.1 IR control method with exclusions of TAPCs

The DBG DR of TAPC3 is accessed by selecting the TAPC\_SELECT\_ENE IR Code in the CLTAPC with a *BYPASS* IR Code in all other TAPCs. A similar approach is used to access the DBG1 register in TAPC4.

- <TAP reset>
- Select TAPC3 (takes 24 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_ENE, TAPC2.bp, TAPC3.DBG, TAPC4.bp, TAPC5.bp>
- Access DBG Register in TAPC3 (takes 35 TCK cycles):  
DRshift <1, 1, TAPC3.DBG\_data, 1>
- Select TAPC4 (takes 24 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_EEN, TAPC2.bp, TAPC3.bp, TAPC4.DBG1, TAPC5.bp>

NOTE—This IR code can be *BYPASS* (all-ones opcode) when the CLTAPC output for the MTCP and MTDP is registered, as described in 16.8.1.3.

- Access DBG1 Register in TAPC4 (takes 35 TCK cycles):  
DRshift <1, 1, TAPC4.DBG1\_data, 1>

Total # TCK cycles:  $24 + 35 + 24 + 35 = 118$  TCK cycles.

### 16.11.2 IR control method with isolation of TAPCs

First, the assumption is made that all TAPCs in Chip 2 are **operating in Normal mode** at the start of the sequence. The DBG DR of TAPC3 is accessed by selecting the TAPC\_SELECT\_INI IR Code in the CLTAPC with a Bypass IR Code in all other TAPCs. A similar approach is used to access the DBG1 register in TAPC4.

- < Previous operation(s) that isolated all TAPCs >
- Isolate TAPC2 and TAPC4 (takes 24 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_INI, TAPC2.bp, TAPC3.DBG, TAPC4.bp, TAPC5.bp>
- Access DBG Register in TAPC3 (takes 35 TCK cycles):  
DRshift <1, 1, TAPC3.DBG\_data, 1>
- Isolate TAPC3 and put TAPC4 in Normal mode (takes 16 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_IIN, TAPC3.bp, TAPC5.bp>
- Access DBG1 Register in TAPC4.DBG1 (takes 16 + 35 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_IIN, TAPC4.DBG1, TAPC5.bp>  
DRshift <1, 1, TAPC4.DBG1\_data, 1>

Total # TCK cycles:  $24 + 35 + 16 + 16 + 35 = 126$  TCK cycles.

Next, the assumption is made that all TAPCs in Chip 2 except the CLTAPC are **operating in Isolated mode** at the start. The DBG DR of TAPC3 is accessed by selecting the TAPC\_SELECT\_INI IR Code in

the CLTAPC with a Bypass IR Code in all other TAPCs. A similar approach is used to access the DBG1 register in TAPC4.

- < Previous operation(s) that isolated all TAPCs, e.g. a TAP reset >
- Put TAPC3 in Normal mode (takes 12 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_INI, TAPC5.bp>
- Access DBG Register in TAPC3 (takes 16 + 35 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_INI, TAPC3.DBG, TAPC5.bp>  
DRshift <1, 1, TAPC3.DBG\_data, 1>
- Isolate TAPC3 and put TAPC4 in Normal mode (takes 16 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_IIN, TAPC3.bp, TAPC5.bp>
- Access DBG1 Register in TAPC4 (takes 16 + 35 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT\_IIN, TAPC4.DBG1, TAPC5.bp>  
DRshift <1, 1, TAPC4.DBG1\_data, 1>

Total # TCK cycles:  $12 + 16 + 35 + 16 + 16 + 35 = 134$  TCK cycles.

#### **16.11.3 DR control method with exclusion of TAPCs**

The DBG DR of TAPC3 is accessed by selecting a CLTAPC Data Register TAPC\_SELECT of which the contents (three bit) can include/exclude TAPC2.-TAPC4. A similar approach is used to access the DBG1 Register in TAPC4.

- < TAP reset >
- Select TAPC3...4 (takes 24+5 TCK cycles = 29 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT, TAPC2.bp, TAPC3.bp, TAPC4.bp, TAPC5.bp>  
DRshift <1, CLTAPC.TAPC\_SELECT\_ENE, 1>
- Access DBG Register in TAPC3 (takes 24 + 36 TCK cycles = 60 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.bp, TAPC2.bp, TAPC3.DBG, TAPC4.bp, TAPC5.bp>  
DRshift <1, 1, TAPC3.DBG\_data, 1, 1>
- Select TAPC4...4 (takes 24+5 TCK cycles = 29 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT, TAPC2.bp, TAPC3.bp, TAPC4.bp, TAPC5.bp>  
DRshift <1, CLTAPC.TAPC\_SELECT\_EEN, 1>
- Access DBG1 Register in TAPC4 (takes 24 + 36 TCK cycles = 60 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.bp, TAPC2.bp, TAPC3.bp, TAPC4.DBG1, TAPC5.bp>  
DRshift <1, 1, 1, TAPC4.DBG1\_data, 1>

Total # TCK cycles:  $29 + 60 + 29 + 60 = 178$  TCK cycles.

#### **16.11.4 DR control method with isolation of TAPCs**

First, the assumption is made that all TAPCs are **operating in Normal mode** at the start. The DBG DR of TAPC3 is accessed by selecting a CLTAPC Data Register TAPC\_SELECT where the contents of this register puts TAPC3 in Normal mode and isolates TAPC2 and TAPC4. A similar approach is used to access the DBG1 Register in TAPC4.

- < Previous operation(s) that selected normal operating mode for all TAPCs >
- Isolate TAPC2 and TAPC4 (takes 24 + 8 = 32 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT, TAPC2.bp, TAPC3.DBG, TAPC4.bp,

- TAPC5.bp>  
DRshift <1, CLTAPC.TAPC\_SELECT\_INI, 1, 1, 1, 1>
- Access DBG Register in TAPC3 (takes 35 TCK cycles):  
DRshift <1, 1, TAPC3.DBG\_data, 1>
  - Isolate TAPC3 and select Normal operating mode for TAPC4 (takes 16 + 6 = 22 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT, TAPC3.bp, TAPC5.bp>  
DRshift <1, CLTAPC.TAPC\_SELECT\_IIN, 1, 1>
  - Access DBG1 Register in TAPC4 (takes 16 + 35 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.bp, TAPC4.DBG1, TAPC5.bp>  
DRshift <1, 1, TAPC4.DBG1\_data, 1>

Total # TCK cycles:  $32 + 35 + 22 + 16 + 35 = 140$  TCK cycles.

Next, the assumption is made that all TAPCs in Chip 2 except the CLTAPC are **operating in Isolated mode** at the start. The DBG DR of TAPC3 is accessed by selecting a CLTAPC Data Register TAPC\_SELECT of which the contents includes TAPC3 and isolates TAPC2 and TAPC4. A similar approach is used to access the DBG1 Register in TAPC4.

- < Previous operation(s) that isolated all TAPCs, e.g. a TAP reset >
- Put TAPC3 in Normal mode (takes 12 + 5 = 17 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT, TAPC5.bp>  
DRshift <1, CLTAPC.TAPC\_SELECT\_INI, 1>
- Access TAPC3.DBG (takes 16 + 35 = 51 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.bp, TAPC3.DBG, TAPC5.bp>  
DRshift <1, 1, TAPC3.DBG\_data, 1>
- Isolating TAPC3 and put TAPC4 in Normal mode (takes 16 + 6 = 22 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.TAPC\_SELECT, TAPC3.bp, TAPC5.bp>  
DRshift <1, CLTAPC.TAPC\_SELECT\_IIN, 1, 1>
- Access TAPC4.DBG1 (takes 16 + 35 TCK cycles):  
IRshift <TAPC1.bp, CLTAPC.bp, TAPC4.DBG1, TAPC5.bp>  
DRshift <1, 1, TAPC4.DBG1\_data, 1>

Total # TCK cycles:  $17 + 51 + 22 + 16 + 35 = 141$  TCK cycles.

## 16.12 Identification of on-chip TAP controller(s)

### 16.12.1 Description

The following two methods to identify the on-chip TAP controllers are recognized:

- The use of a mandatory Chip-Level Identification Code
- An optional query option of the Device Identification Code, when implemented by the on-chip TAP controllers

To support the first identification method, the logic controlled by the CLTAPC will provide a Chip-Level Device Identification Code that conforms to IEEE Std 1149.1. For an IEEE 1149.7 device, this identification method is mandatory. A detailed description of the chip's content can subsequently be obtained from external documentation based on this Device Identification Code.

When using the CLTAPC\_IR control method, the scan paths controlled by the CLTAPC are concatenated with the scan paths controlled by the EMTAPCs making up the applications-debug chain, with the CLTAPC managed paths closest to the TDI signal. With this chain configuration, the concatenation of 32-bit TAP Identification Code can be read with a DR Scan after exiting the *Test-Logic-Reset* state without an intervening IR Scan. When using the CLTAPC\_DR control method, IR Scans are required to read the 32-bit TAP Device Identification Codes of EMTAPCs.

Using this information, the DTS software can identify those TAPCs within the applications-debug chain that have a Device Identification Code Register implemented.

## 16.12.2 Specifications

### Rules

- a) Each subsequent specification in 16.12.2 shall apply to T0 and above TAP.7s.
- b) A Device Identification Register, as specified in IEEE Std 1149.1, shall be implemented in the CLTAPC.

### Permissions

- c) An *IDCODE* instruction and Device Identification Register, as specified in IEEE Std 1149.1, may be implemented in an EMTAPC.

## 16.13 Multiple dies in one package

### 16.13.1 Description

Some packaging technologies provide for more than one die in a single package. When such a multiple chip module (MCM) or system in a package (SiP) is used at the next level of design hierarchy, the user may prefer to again have the component adhere to IEEE Std 1149.1. Among others, this is required for the huge installed base of testers at the many electronic manufacturers that work according to IEEE Std 1149.1. When the resulting package needs to comply with rules described in IEEE Std 1149.1, the way of connecting the dies inside the package needs consideration.

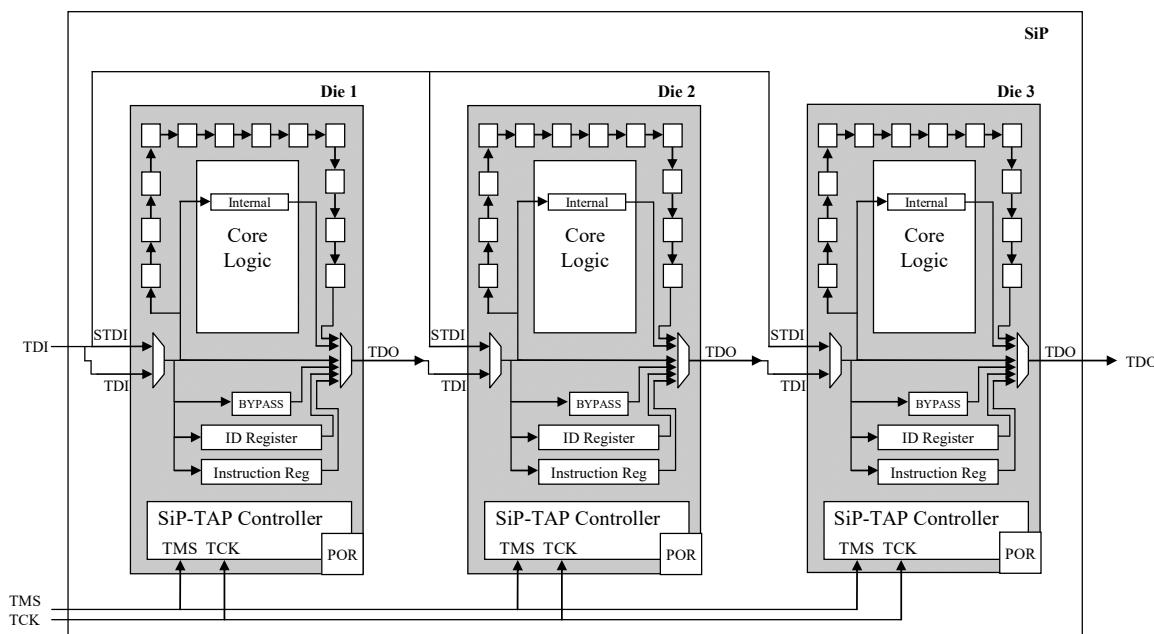
The SiP-TAP approach described within this subclause is the recommended way to address issues that arise when creating a package with the IEEE 1149.1 TAP interface where there are multiple dies in one package and there is a need to comply with rules described in IEEE Std 1149.1. The issues that would create conflicts with the rules described in IEEE Std 1149.1 unless addressed in some manner. The issues created by using a standard PCB implementation within an MCM with daisy-chained TDI-TDO connections are listed below. This creates a device with the following:

- More than one active Test Access Port. (IEEE Std 1149.1 requires a single active TAP.)
- A Data Register Scan Path that has as many bypass cells as there are digital dies involved. (IEEE Std 1149.1 requires a *BYPASS* Register with a length of exactly one bit.).
- A Device Identification Register Scan Path where the Device Identification Register length becomes greater than 32 bits. (IEEE Std 1149.1 requires a Device Identification Register, when implemented, to be a length that is exactly 32 bits.)
- An nTRST signal that does not reset all test logic when dies with and without an nTRST signal are mixed (IEEE Std 1149.1 requires that the optional nTRST signal, when implemented and activated, resets all TAP controller logic to the defined *Test-Logic-Reset* state).
- A BSDL file and netlist of the package describing the MCM as a simple PCB described with the SiP netlist. This may result in a nonoperational set of patterns not only because of the safe values that may be required in the SiP but also because of the fact that the netlist may not be complete (for example, the signal numbering is not present). At the top level, the SiP netlist will describe

SiP signals and their connection to the pads (and resulting cell order) on the selective dies. From this description, the requirements and the rules emerge for the dies. (IEEE Std 1149.1 requires a device claiming compliance to IEEE Std 1149.1 have a BSDL file describing the implementation details.)

### 16.13.1.1 Exposing an IEEE 1149.1 DR for the **BYPASS** and **IDCODE** instructions

In the IEEE 1149.7 SiP-TAP approach, a multiplexer is placed in front of the internal TDI signal in the TAP controller of at least one of the dies. This multiplexer selects between the following two die inputs: (1) the IEEE 1149.1 TDI input, and (2) a secondary TDI input called STDI. At the MCM or SiP-level, all STDI inputs are connected to the TDI input of the package. The remainder of the TAP interface signals (TCK, TDI, TDO, TMS, and the optional nTRST) of the individual dies are connected as they would be on a board (see Figure 16-13).



**Figure 16-13 — Example of the active **BYPASS** instruction**

The multiplexer inside the SiP-TAP Controller selects the STDI die input whenever the IEEE 1149.1 **BYPASS** and **IDCODE** instructions are loaded into the corresponding SiP-TAP Controller. To use this approach, at least one die in the package has to contain such an SiP-TAP Controller.

### 16.13.1.2 Exposing the complete boundary-scan chain for IEEE 1149.1 instructions

Because controllability over the complete boundary-scan chain of the SiP is required for board-level test, the TAPCs controlling the individual boundary-scan chains in each individual die have to operate in *Normal mode* during board-level test. The multiplexer inside the SiP-TAP Controller selects the TDI die input in this mode.

### 16.13.1.3 BSDL documentation

IEEE Std 1149.1 also implies that a single BSDL file is provided with the package to describe all variables of the SiP's boundary-scan implementation. This includes its public and private instructions and the connectivity of cells to the respective pads and signals. The instruction set of the SiP-TAP is a double set, where the selection of the STDI die input requires special attention. An example set of SiP-TAP instructions is shown in Table 16-4.

**Table 16-4 — An example instruction set for the IR managed by an SiP-TAP**

Instruction name	Instruction code	TDI selected	STDI selected	Chain length
<i>BYPASS</i>	1111	—	X	1*
<i>Q_BYPASS</i>	0111	X	—	1xN
<i>EXTEST</i>	0000	X	—	17
<i>Q_EXTEST</i>	1000	—	X	17
<i>IDCODE</i>	1010	—	X	32*
<i>Q_IDCODE</i>	0010	X	—	32xN
<i>PRELOAD</i>	1011	X	—	17
<i>Q_PRELOAD</i>	0011	—	X	17
<i>WIRE</i>	1101	X	—	0
<i>Q_WIRE</i>	0101	—	X	0
<i>USER1</i>	1110	X	—	10
<i>Q_USER1</i>	0110	—	X	10
<i>USER2</i>	1100	X	—	5
<i>Q_USER2</i>	0100	—	X	5

NOTE—Lengths denoted with an asterisk are mandated by IEEE Std 1149.1.

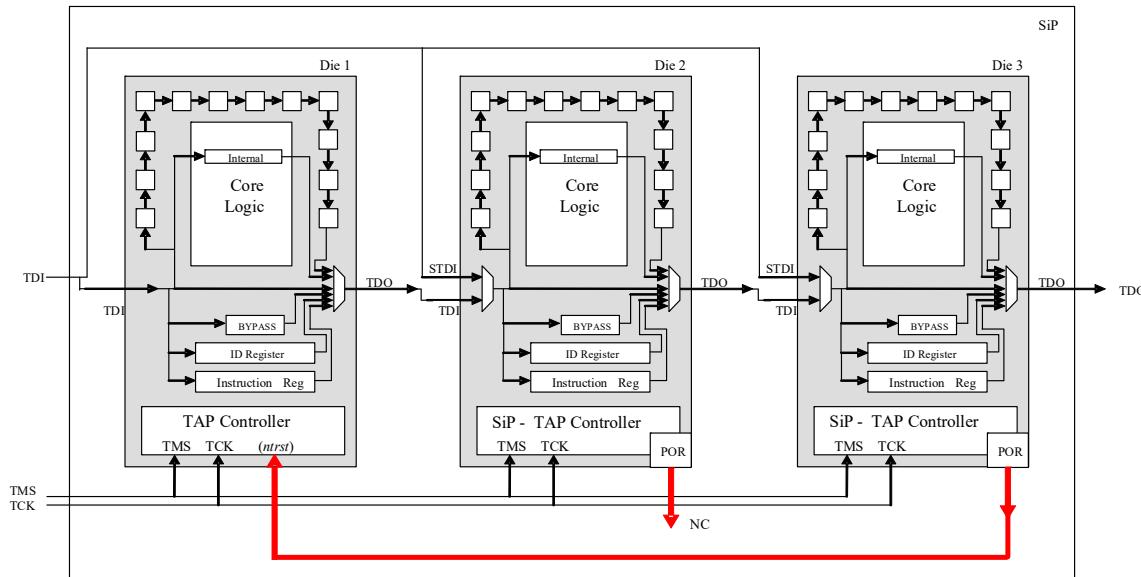
#### 16.13.1.4 Packaging dies

IEEE Std 1149.1 allows for an optional nTRST input on each die in the package. Depending on the particular combination of dies, with or without nTRST, the SiP-TAP approach uses the following approach to create IEEE 1149.1 behavior at the package level:

- All dies implement nTRST functionality.
  - The TRST\* die inputs are brought out to an nTRST package input when all dies implement TRST\* functionality.
  - All dies may simultaneously reset the test logic of all dies before they are operated together.
  - The TAP interface of the package has the following five IEEE 1149.1 signals: TCK, TMS, TDI, TDO, and nTRST.
- None of the dies implement nTRST functionality.
  - The TAP interface of the package has the following four IEEE 1149.1 signals: TCK, TMS, TDI, and TDO.
- Some dies implement nTRST functionality, other dies do not.
  - One or all of the dies with an SiP-TAP controller is used to provide a Power-On-Reset (POR\*) signal inside the package.
  - The POR\* signal is connected to the nTRST inputs of the dies that implement nTRST functionality.
  - The TAP interface of the package has the following four IEEE 1149.1 signals: TCK, TMS, TDI, and TDO.

### 16.13.1.5 SiP-TAP POR\* functionality

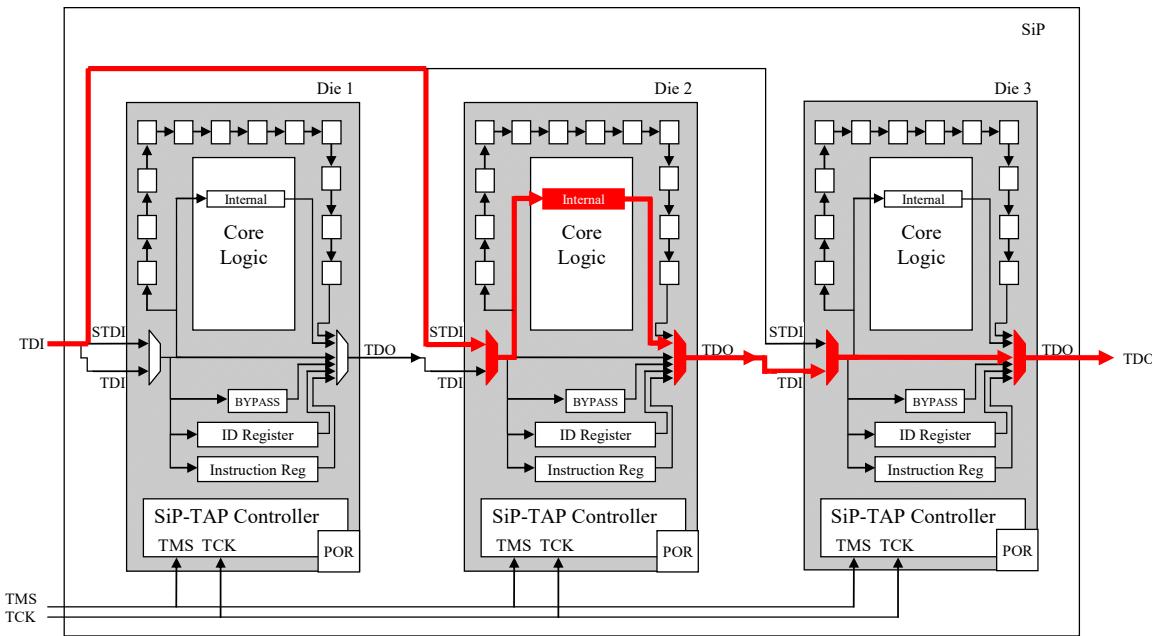
A SiP-TAP controller can implement a POR\* signal for use in packages that group dies with and without nTRST. The solution is to connect the SiP-TAP POR\* circuit of one of the dies to the nTRST signal of all dies that require the nTRST signal at power-up (refer to Figure 16-4). The output buffer of the POR\* pad is preferably implemented as an “open collector” type. When more POR\* signals are available with different timing, or when a nTRST signal is implemented on the package, these signals may be combined to give a maximum length, full-power, active low- reset pulse.



**Figure 16-14 — Package-level connectivity of the POR\* and nTRST signals**

### 16.13.1.6 DR-wire bypass

Recommendation 16.13.2 k) supports debug. The DR-wire bypass (Figure 16-15) can be directly implemented in an SiP-TAP Controller. It is used only during DR Scans. Instructions are required to be scanned through the SiP-TAP Controller’s Instruction Register.



**Figure 16-15 — The DR-wire solution in the SiP-TAP approach**

### 16.13.2 Specifications

#### Rules

- a) Each subsequent specification in 16.13.2 shall apply to T0 and above TAP.7s when multiple dies with individual IEEE 1149.1 interfaces are integrated in a single package.
- b) One of the following shall be true:
  - 1) When the package has to behave as an IEEE 1149.1 component, at least one die shall implement an IEEE 1149.7 SiP-TAP controller.
  - 2) When the package does not have to behave as an IEEE 1149.1 component, it shall be operable as if the dies are individual IEEE 1149.1 components on a printed circuit board.
- c) An IEEE 1149.7 SiP-TAP controller shall consist of a standard IEEE 1149.1 controller with the following extensions:
  - 1) An additional STDI input.
  - 2) Only one of the following:
    - i) An nPOR output.
    - ii) An nTPOR input/output, combining the POR\* output functionality with IEEE 1149.1 nTRST input functionality.
- d) The SiP-TAP controller of Rule 16.13.2 b) shall implement the *IDCODE* instruction.
- e) The SiP-TAP controller of Rule 16.13.2 b) shall select the STDI input for the IEEE 1149.1 *BYPASS* and *IDCODE* instructions.
- f) When implemented, the SiP-TAP POR\* output of Rule 16.13.2.c) 2 ) i) shall be asserted upon power-on of the SiP-TAP controller.
- g) When implemented, the SiP-TAP TPOR\* input/output of Rule 16.13.2.c) 2) ii) shall be asserted upon power-on of the SiP-TAP controller.

- h) When the SiP-TAP controller implements the optional IEEE 1149.1 nTRST functionality, it shall be combined with the POR\* output functionality on a single die input/output called TPOR\*.
- i) When a package that integrates dies with and without nTRST has to behave as an IEEE 1149.1 component, it shall:
  - 1) Integrate at least one die with the SiP-TAP controller as specified by Rule 16.13.2 b).
  - 2) Connect the nTRST of all dies to one of the following only:
    - i) The POR\* output of the SiP-TAP controller die.
    - ii) The TPOR\* input/output of the SiP-TAP controller die.
  - 3) Not provide an nTRST input on its interface.
- j) A package that integrates multiple dies shall only provide an nTRST input on its interface when all dies implement an nTRST input.

## Recommendations

- k) When the SiP-TAP controller is implemented, it should support a combinational path (nonclocked) between TDI or STDI and TDO under Instruction Register control.

## 16.14 Managing STL Group Membership

With a T0 TAP.7, the STL is a member of the Scan Group. The RSU selects and deselects a nonexistent ADTAPC, which passes the information controlling the selection of the ADTAPC to the CLTAPC. The RSU functions as an ADTAPC surrogate in this case, selecting the CLTAPC instead of the ADTAPC. This aspect of the T0 TAP.7 is shown in Figure 16-1. The *sys\_tck* signal supplied to the CLTAPC is created with the TCK signal. The RSU gates the *sys\_tck* signal when it is implemented.

## 16.15 RSU operation

### 16.15.1 Description

An RSU implemented with a T0 TAP.7 has the characteristics described in Clause 11, subject to the rules in 16.15.2. With a T0 TAP.7, the STL is a member of the Scan Group. The RSU selects and deselects a nonexistent ADTAPC, which passes the information controlling the selection of the ADTAPC to the CLTAPC. The RSU functions as an ADTAPC surrogate in this case, selecting the CLTAPC instead of the ADTAPC. This aspect of the T0 TAP.7 is shown in Figure 16-1. The *sys\_tck* signal supplied to the CLTAPC is created with the TCK signal, and this signal is gated by the RSU when the RSU is implemented.

### 16.15.2 Specifications

#### Rules

- a) Each subsequent specification in 16.15.2 shall apply to T0 TAP.7.
- b) The CLTAPC state shall be parked, provided an RSU is implemented and the Control State Machine state within the RSU is in a state other than *STD*.
- c) The scan topology supported by the RSU shall be hardwired as a Series.

## 16.16 Programming considerations

Because this clause allows the use of multiple methods for the selection and deselection of EMTAPCs, information pertaining to this subject will be obtained from sources other than this standard.

With a T0 TAP.7, the STL is a member of the Scan Group.

## 17. Extended concepts

### 17.1 Introduction

This clause is applicable to T1 and above TAP.7s. It extends the high-level description the concepts used with TAP.7 architecture provided by Clauses 4 through 8, and 15. It describes the TAP.7 Controller extensions to the IEEE 1149.1 protocol (Extended Control). This description includes the use of ZBSs, EPU Operating States, and the creation of control levels. It also provides programming considerations.

The subject matter within this clause is described in the following order:

- 17.2 Suitability of *BYPASS* and *IDCODE* instructions for extended control
- 17.3 ZBS detection
- 17.4 Incrementing, locking, and clearing the ZBS count
- 17.5 Shared use of ZBSs by the EPU and STL
- 17.6 EPU functionality associated with the ZBS count
- 17.7 Programming considerations

### 17.2 Suitability of *BYPASS* and *IDCODE* instructions for extended control

The following characteristics of the *BYPASS* and *IDCODE* instructions allow their use to extend the IEEE 1149.1 protocol:

- The *BYPASS* instruction is specified as mandatory per IEEE Std 1149.1.
- These are considered inert as:
  - The data loaded by the *Capture-DR* state is a constant value.
  - The data stored by the *Update-DR* state is not utilized.
- The *Test-Logic-Reset* state initializes the Instruction Registers of all TAPs with one of these instructions.

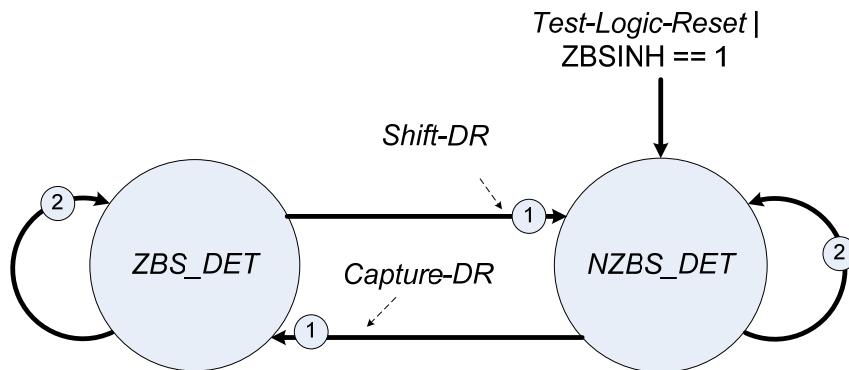
Although any appropriate inert instruction may be used for Extended Control, it is strongly recommended the *BYPASS* and *IDCODE* instructions be used for this purpose. The DTS should place a *BYPASS* or *IDCODE* instruction in the Instruction Register of all TAPCs that will operate in lock-step with the TAP.7 Controller prior to using ZBSs for the purpose of managing TAP.7 Controller capability. It is also a good practice to perform this operation before placing the TAP.7 Controller Offline.

### 17.3 ZBS detection

#### 17.3.1 Description

A ZBS is detected as specified in Rule 17.3.2 b). The detection of a ZBS is enabled by the *Test-Logic-Reset* state and is disabled by storing ZBSINH.

A state machine such as the one shown in Figure 17-1 detects the absence of the *Shift-DR* state in the TAPC state progression of a zero-bit DR Scan. A ZBS is detected when the state of this machine is *ZBS\_DET* when the *Update-DR* state is exited. The state descriptions for this machine are shown in Table 17-1.



**Figure 17-1 — Conceptual view of zero bit DR Scan detection**

**Table 17-1 — ZBS detect state definitions**

State	Description
ZBS_DET	The <i>Capture-DR</i> state has occurred when the ZBSINH Register is a logic 0.
NZBS_DET	A <i>Shift-DR</i> state has occurred since the last <i>Capture-DR</i> state.

### 17.3.2 Specifications

#### Rules

- a) Each subsequent specification in 17.3.2 shall only apply to T1 and above TAP.7s.
- b) A TAP.7 Controller shall identify both of the following TAPC-state progressions as a ZBS:
  - 1) A TAPC-state progression of *Select-DR-Scan*, *Capture-DR*, *Exit1-DR*, and *Update-DR* with no other intervening TAPC State Machine states.
  - 2) A TAPC State Machine state progression of *Select-DR-Scan*, *Capture-DR*, *Exit1-DR*, one or more *Pause-DR states*, *Exit2-DR*, and *Update-DR* with no other intervening TAPC State Machine states.
- c) A ZBSINH Register value of a logic 1 shall inhibit the detection of ZBSs as described in Rule 17.3.2 b).
- d) A TAP.7 Controller shall consider two ZBSs as consecutive, provided any of the following are true:
  - 1) A ZBS follows a ZBS with no intervening TAPC State Machine states.
  - 2) A ZBS follows a ZBS with only intervening *Run-Test/Idle* states.

### 17.4 Incrementing, locking, and clearing the ZBS count

#### 17.4.1 Description

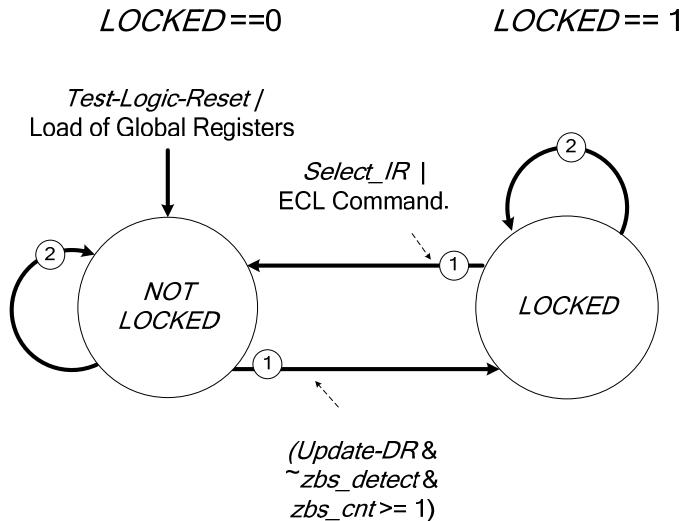
A three-bit ZBS count is used to do the following:

- Establish control levels

- Reestablish TAP.7 Controller operation when the use of control levels has been suspended by storing the SUSPEND Register (it can only be stored as a logic 1)

The *Test-Logic-Reset* state sets the ZBS count to zero. The ZBS count is incremented as described in Rule 17.4.2 b). The ZBS count is locked as described in Rule 17.4.2 c). The ZBS count is unlocked and set to zero as described in Rule 17.4.2 d).

The locking and unlocking of the ZBS count is shown in Figure 17-2. A description of these states is shown in Table 17-2.



**Figure 17-2 — Conceptual view of ZBS count locking**

**Table 17-2 — ZBS count locking states**

Name	Function	Description
NOT LOCKED	Allow Increment of ZBS count	The ZBS count is incremented upon ZBS detection but not past a count of seven.
LOCKED	Lock ZBS count	The ZBS count cannot be incremented.

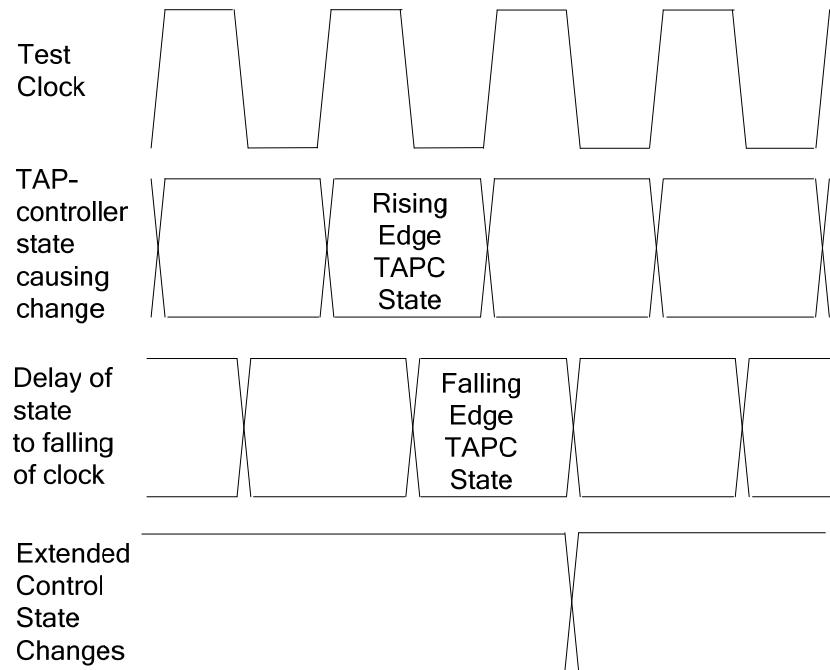
## 17.4.2 Specifications

### Rules

- a) Each subsequent specification in 17.4.2 shall only apply to T1 and above TAP.7s.
- b) The TAP.7 Controller ZBS count shall be incremented when all the following are true:
  - 1) A ZBS is detected.
  - 2) The ZBS count is not locked.
  - 3) The ZBS count is less than seven.
  - 4) The *Update-DR* state ending the ZBS TAPC State Machine state sequence is exited.

- c) The locking of the TAP.7 Controller ZBS count at its current value shall occur when all the following are true:
  - 1) The ZBS count is nonzero.
  - 2) A minimum of one *Shift-DR\_f* state has occurred since the last *Capture-DR\_f* state.
  - 3) The *Update-DR\_f* state is exited on this clock edge.
- d) The ZBS count shall be unlocked and set to zero by any of the following:
  - 1) Any of the following states:
    - i) *Test-Logic-Reset\_f*.
    - ii) *Select-IR-Scan\_f*.
  - 2) An STMC Command that sets any of the following register bits to a logic 1:
    - i) ECL.
    - ii) SUSPEND.
    - iii) ZBSINH.
  - 3) All the following are true:
    - i) The SUSPEND Register is a logic 1.
    - ii) The ZBS count is seven.
    - iii) The ZBS count is not locked.
    - iv) A ZBS is detected.
  - 4) A Selection Sequence loads the Global Registers.

- e) The ZBS count shall be set to zero when all the following are true:
  - 1) The ZBS count is not locked.
  - 2) An SSD completes without being aborted.
- f) TAP.7 Controller changes initiated by TAPC commands shall occur upon the second falling edge of the clock after entry into the TAPC State Machine state causing a state change as shown in Figure 17-3.



**Figure 17-3 — TAPC-state/control-state and ZBS-count relationships**

#### Permissions

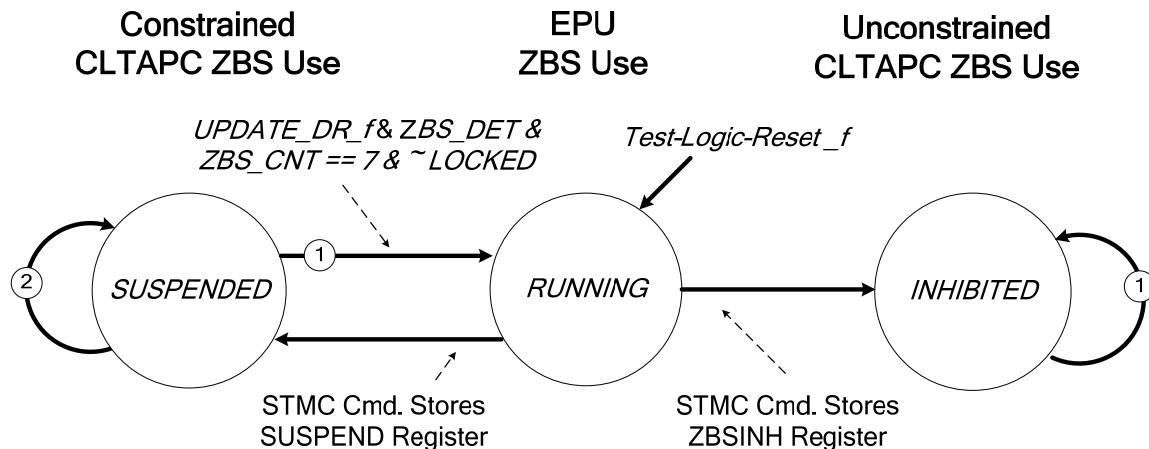
- g) The initialization of the Extended Control Logic may be either synchronous or asynchronous.

### 17.5 Shared use of ZBSs by the EPU and STL

#### 17.5.1 Description

##### 17.5.1.1 EPU Operating States

Within the TAP.7 architecture, the EPU and STL share use of ZBSs. This is accomplished by providing *RUNNING*, *SUSPENDED*, and *INHIBITED* EPU Operating States as shown in Figure 17-4.



**Figure 17-4 — EPU Operating States**

These EPU Operating States provide a means to designate DR Scans for the following:

- STL purposes
- EPU purposes

The use of ZBSs and inert instructions may be allocated to either the EPU or the STL, but not simultaneously. ZBSs intended for EPU use are called “TAP.7 ZBSs.” ZBSs intended for STL use are called “Other ZBSs.”

The values of the SUSPEND and ZBSINH Registers determine the type of operation as shown in Table 17-3. The *Test-Logic-Reset* state creates *Running* operation as it sets the SUSPEND and ZBSINH Register values to a logic 0.

**Table 17-3 — ZBS use state characteristics with EPU Operating States**

<i>zbsinh</i> == 1 ?	<i>suspend</i> == 1 ?	EPU operating state	ZBS use by	Until
No	No	<i>RUNNING</i>	EPU	Command allocates to the STL.
No	Yes	<i>SUSPENDED</i>	STL	ZBS sequence allocates use to the EPU.
Yes	No	<i>INHIBITED</i>	STL	<i>Test-Logic-Reset</i> allocates use to the EPU.
Yes	Yes			Not possible.

### 17.5.1.2 EPU Operating State characteristics

The EPU operating characteristics are shown below:

- *RUNNING* state:
  - Both the System and Control Paths may be used.
  - Control levels may be created and used.
  - The STMC Command storing the SUSPEND Register creates the *SUSPENDED* state.
  - The STMC Command storing the ZBSINH Register creates the *INHIBITED* state.
- *SUSPENDED* state:
  - Only the System Path may be used.
  - ZBSs may be used by the STL.
  - Control levels may not be created/commands may not be used.
  - The ZBS count is operable.
  - Eight ZBSs without locking the ZBS count creates the *RUNNING* state by setting the SUSPEND Register to a logic 0.
- *INHIBITED* state:
  - Only the System Path may be used.
  - ZBSs cannot be detected by the EPU.
  - Control levels may not be created and commands may not be used.

- The ZBS count is zero.
- The *Test-Logic-Reset* state creates the *RUNNING* state.
- The SUSPEND Register value is a logic 0 and the ZBSINH Register value is a logic 1.

The TAP.7 operation continues after changes in the EPU Operating State, as these changes do not disturb the state of other registers in the TAP.7 Controller register set.

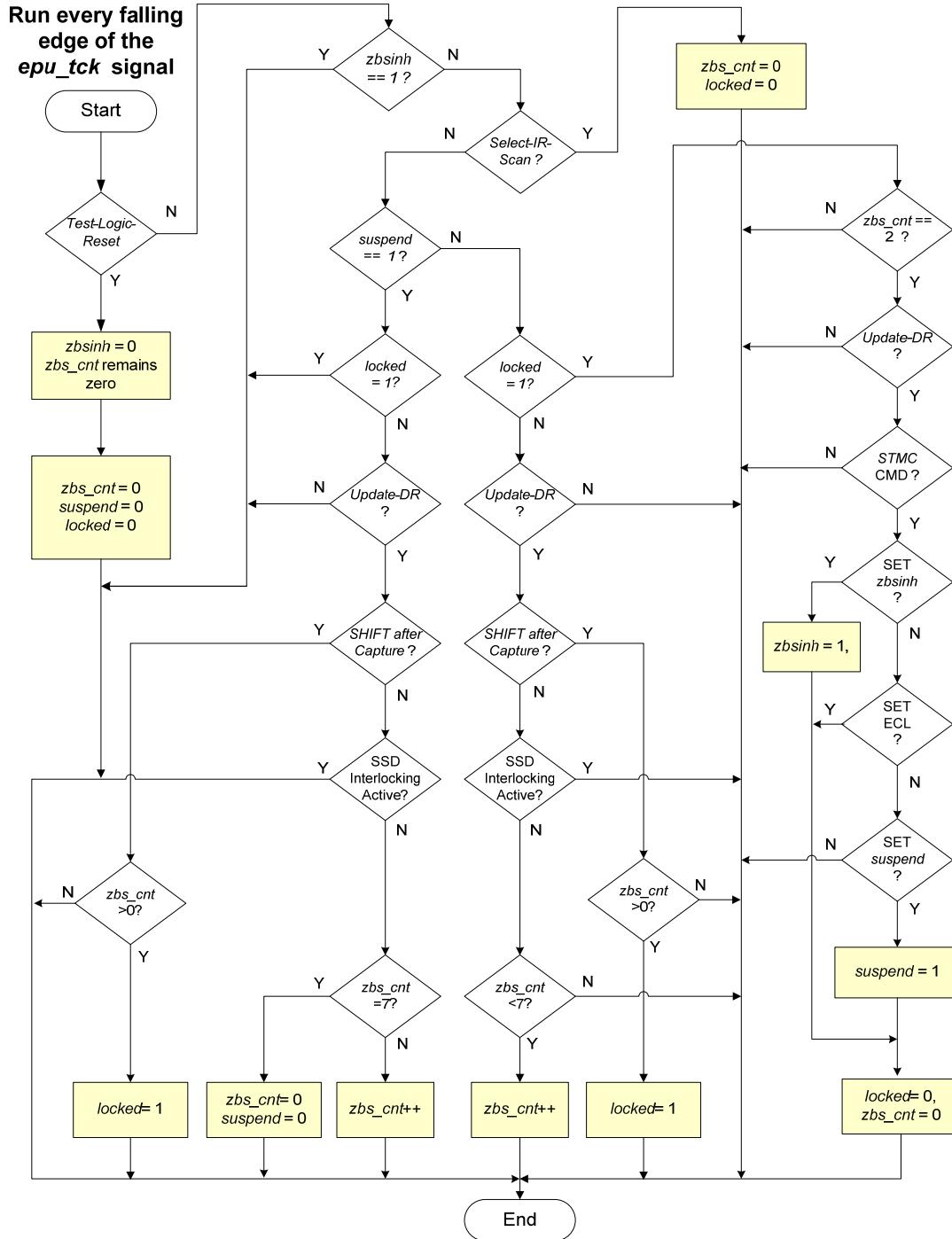
### 17.5.1.3 ZBS use that is compatible with EPU Operating States

Only the following Other ZBS use cases are compatible with the EPU Operating States:

- Case I (use only the *RUNNING* state)—There is no need to use ZBSs for STL purposes or the use of ZBSs for STL purposes is satisfied by the use of Control Level One.
- Case II (use the *RUNNING* and the *SUSPENDED* states)—There is a need to use ZBSs for STL purposes:
  - With an instruction other than *BYPASS* or *IDCODE*.
  - Using no more than seven consecutive ZBS before any of the following:
    - Locking the ZBS count.
    - Using the either the *Select-IR-Scan* or the *Test-Logic-Reset* state.
- Case III (use the *RUNNING* and the *INHIBITED* states)—There is a need to use ZBSs for STL purposes:
  - With an instruction other than *BYPASS* or *IDCODE*.
  - Using more than seven consecutive ZBS before any of the following:
    - Locking the ZBS count.
    - Using either the *Select-IR-Scan* or the *Test-Logic-Reset* state.

The following is expected:

- Use Case I will be the dominant use case by far.
- Use Case II may be encountered in only a small number of systems (or not encountered at all).
- Use Case III should almost never be encountered.
- Support for any other ZBS use cases is not required.



**Figure 17-5 — ZBS use state and ZBS-count control flow**

#### **17.5.1.4 An approach to implementing EPU Operating States**

Figure 17-5 shows a conceptual view of EPU operation combining the locking of the ZBS count shown in Figure 17-2 and the EPU Operating States shown in Figure 17-4. This is a conceptual view, and other representations of these states should be considered. Implementing this machine as part of a larger machine with more functionality may also be considered. Table 17-4 provides a description of the states shown in Figure 17-5.

A flowchart describing the operation of this state machine is shown in Figure 17-6. The ZBS count and locked functions operate the same with both TAP.7 and Other ZBSs. However, the action taken when more than seven consecutive ZBSs occur without locking the ZBS count is different.

The STMC Command setting the ECL, SUSPEND, and ZBSINH registers to a logic 1 and zeroing/unlocking the ZBS count are listed below:

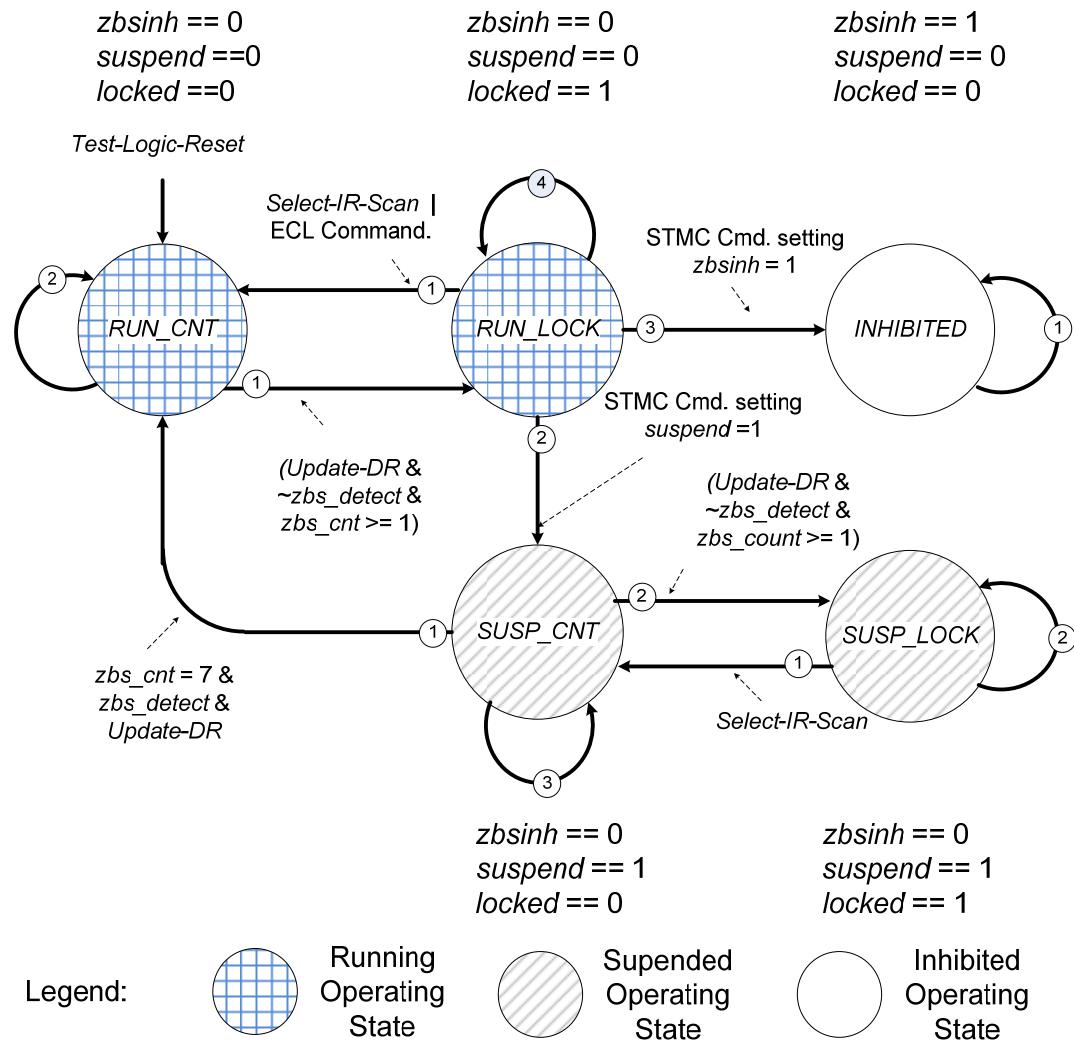
- CMD(STMC, ECL) >> moves state from *RUN\_LOCK* to *RUN\_CNT*
- CMD(STMC, SUSPEND) >> moves state from *RUN\_LOCK* to *SUSP\_CNT*
- CMD(STMC, ZBSINH) >> moves state from *RUN\_LOCK* to *INHIBITED*

It is the DTS' responsibility to ensure the following:

- An inert instruction has been placed in the Instruction Register of all TAPCs before the use of ZBSs for EPU purposes
- The STL or another function using ZBSs does not utilize ZBSs when the EPU has been notified ZBS use is for EPU purposes

**Table 17-4 — State description for Figure 17-6**

State name	Function			Description
	ZBS detection enabled	Control level:	ZBS count is locked:	
<i>RUN_CNT</i>	Yes	Creation Possible	No	— ZBS count is incremented upon ZBS detection but not past a count of seven.
<i>RUN_LOCK</i>	Yes	Created	Yes	— The ZBS count defines a control level. — The EPU may use ZBSs in any quantity and in any manner.
<i>SUSP_CNT</i>	Yes	None	No	— The STL may use up to seven consecutive ZBSs.
<i>SUSP_LOCK</i>	Yes	None	Yes	— The STL may use ZBSs in any quantity and in any manner.
<i>INHIBITED</i>	No	None	No	— The STL may use ZBSs in any quantity and in any manner.



**Figure 17-6 — Conceptual view of the ZBS use states with both TAP.7 and other ZBS use**

## 17.5.2 Specifications

### Rules

- Each subsequent specification in 17.5.2 shall only apply to T1 and above TAP.7s.
- The ECL Register value shall be set to a logic 0 immediately following the *Update-DR\_f* state that sets it to a logic 1.
- A SUSPEND Register value of a logic 1 shall inhibit the creation of control levels.
- The SUSPEND Register value shall be set to a logic 0 coincidently with the ZBS count being set to zero when all of the following are true:
  - A ZBS is detected.
  - The ZBS count is not locked.
  - The ZBS count value is seven.
  - The *Update-DR\_f* state is exited.

NOTE—Table 9-2 specifies that the ECL, SUSPEND, ZBSINH Register values are set to a logic 0 with the *Test-Logic-Reset* state.

## 17.6 EPU functionality associated with the ZBS count

### 17.6.1 Description

Functionality associated with the ZBS count when the SUSPEND Register value is a logic 0 is shown below:

- A ZBS count greater than one:
  - Causes the use of the Control Path
  - Inhibits SSD detection
  - Inhibits the use of Deselection and Selection Escapes
- A ZBS count identifies the control level when the ZBS count is locked:
  - Level 2: Utilized to implement TAP.7 Controller commands
  - Level 3: Reserved for future use
  - Levels 4 and 5: Provide access to Auxiliary Scan Paths within the TAP.7 Controller
  - Level 6: May be used to implement the Deselection Alert option
  - Level 7: Provided for a DTS to utilize for its own purposes

Functionality associated with the ZBS count when the SUSPEND Register value is a logic 1 is described in 17.4.1.

### 17.6.2 Specifications

#### Rules

- a) Each subsequent specification in 17.6.2 shall only apply to T1 and above TAP.7s.
- b) When the SUSPEND Register value is a logic 0, a TAP.7 Controller shall define a locked ZBS count as a control level as shown in Table 17-5.

**Table 17-5 — Control levels**

ZBS count	Control level	Name	Resulting function
1	1	OTHER	ZBSs may be used by the STL
2	2	CMD	TAP.7 Controller commands
3	3	RSVD	Reserved
4	4	AUX0	Access to Auxiliary Scan Paths
5	5	AUX1	
6	6	OFF	Initiate Deselection Alert or No Operation
7	7	DTS	Reserved for DTS use

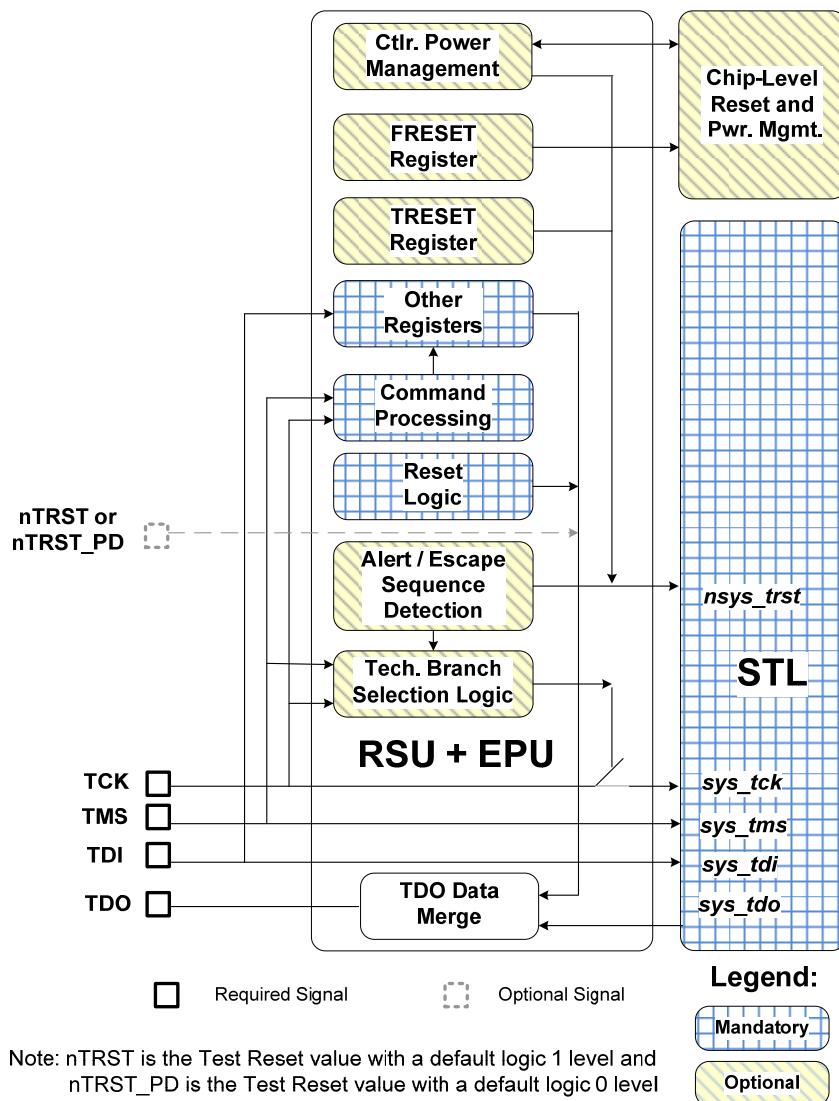
## 17.7 Programming considerations

The DTS uses the EPU Operating States to notify the EPU of the subsequent use of ZBSs (for EPU or STL purposes). This notification is required because ZBSs are detected without regard to Instruction Register values within the CLTAPC, EMTAPCs, and elsewhere in the scan topology. The EPU does not have knowledge of Instruction Register values of the TAPCs in other chips. It is therefore the DTS' responsibility to notify the EPU before a change in ZBS use begins. It is considered a programming error if either a CLTAPC or an EMTAPC utilize a ZBS at the same time as the TAP.7 Controller.

## 18. T1 TAP.7

### 18.1 Introduction

This clause is applicable to T1 and above TAP.7s. It extends the high-level description of the T1 TAP.7 provided in Clause 5. It provides the rules, permissions, and recommendations for the implementation of a T1 TAP.7. The EPU, a number of optional register bits, and the commands supporting them are added to the T0 TAP.7 to create a T1 TAP.7. The block diagram of a typical T1 TAP.7 is shown in Figure 18-1. This subclause provides a description of the optional capabilities shown in this figure. It also provides programming considerations.



**Figure 18-1 — Conceptual view of a T1 TAP.7**

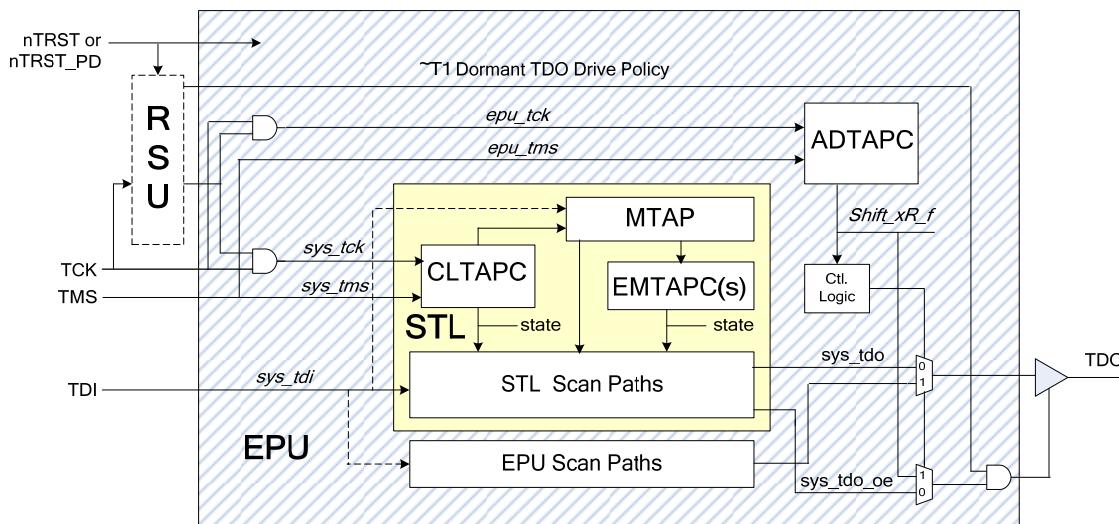
The subject matter within this clause is described in the following order:

- 18.2 Deployment
- 18.3 Capabilities

- 18.4 Register and command portfolio
- 18.5 Configurations
- 18.6 Start-up behavior
- 18.7 Conditional group membership
- 18.8 Test Reset
- 18.9 Functional reset
- 18.10 Power control
- 18.11 RSU operation
- 18.12 Programming considerations

## 18.2 Deployment

A high-level block diagram of a deployed T1 TAP.7 is shown in Figure 18-2.



**Figure 18-2 — Deployment of the T1 TAP.7**

## 18.3 Capabilities

### 18.3.1 Inherited

All mandatory capabilities of T0 TAP.7s are also mandatory for a T1 TAP.7. T0 TAP.7 options are also options for a T1 TAP.7.

The capabilities of T0 TAP.7 inherited by a T2 TAP.7 ensure that it:

- Is controlled in the same manner as lower TAP.7 Classes
- Is compatible with any TAP.1 or TAP.7 Controller operating in a Series Scan Topology

### 18.3.2 New

A T1 TAP.7 provides the following five optional capabilities:

- Reading TAP.7 Controller registers
- Issuing a test reset to the STL via the *nsys\_trst* signal
- Initiating a functional-reset request
- Providing TAP.7 power management
- Implementing an RSU

## 18.4 Register and command portfolio

### 18.4.1 Description

#### 18.4.1.1 General information

The T1 TAP.7 registers and the commands managing their contents are shown in Table 18-1. The commands shown in this table are described in Clause 9.

**Table 18-1 — T1 TAP.7 function/register/command relationships**

Register	Type	Function	Associated command	Described in subclause
<b>CGM</b>	W	Conditional Group Member	<b>STMC, MCM, SCNB</b>	13.9
<b>CNFG0</b>	R	Read public configuration	<b>SCNS</b>	9.10.2
<b>CNFG1 (optional)</b>	R	Read public configuration	<b>SCNS</b>	
<b>CNFG2 (optional)</b>	R	Read private configuration	<b>SCNS</b>	
<b>CNFG3 (optional)</b>	R	Read private configuration	<b>SCNS</b>	
<b>ECL</b>	W	Exit Control Level	<b>STMC</b>	17.4
<b>FRESET</b>	W	Functional-Reset request	<b>STC1</b>	18.9
<b>FRESETR</b>	R	Functional-Reset request return	<b>SCNB</b>	
<b>PWRMODE</b>	W	TAP.7 power management	<b>STC2</b>	18.10
<b>RDBACK0</b>	R	Read register values	<b>SCNS</b>	9.10.1
<b>SUSPEND</b>	W	Suspend TAP.7 Controller operation	<b>STMC</b>	17.4
<b>TRESET</b>	W	Test Reset	<b>STC1</b>	18.8
<b>ZBSINH</b>	W	Inhibit ZBS detection	<b>STMC</b>	17.4

NOTE—See Table 9-2 for information regarding the implementation/initialization of these registers. The new registers and commands that are added with the TAP.7 Class are shown in **BOLD** print. These registers are described in Table 18-2 with the commands supporting them described in Table 9-4.

#### 18.4.1.2 Register acronyms

The acronyms for the registers shown in Table 18-1 are shown as follows:

- CGM Conditional Group Member
- CNFG0 Configuration Register 0
- CNFG1 Configuration Register 1
- CNFG2 Configuration Register 2
- CNFG3 Configuration Register 3
- ECL Exit Control Level
- FRESET Functional RESET
- TRESET Test RESET

- PWRMODE Power MODE
- RDBACK0 Read Back Register 0
- SUSPEND Suspend Register
- TRESET Test RESET
- ZBSINH Zero Bit Scan Detect Inhibit

### 18.4.1.3 Global Registers

The registers listed as follows are Global Registers, whereas those not listed are Local Registers:

- ECL
- SUSPEND
- ZBSINH

These registers are set to a logic 0 value as a result of a Global Register load when an RSU is implemented.

### 18.4.1.4 Registers already described

The CNFG3-CNFG0, RDBACK0, ECL, SUSPEND, and ZBSINH Registers have been described elsewhere in this standard. Refer to Table 18-1 for the location of these descriptions.

### 18.4.1.5 New register descriptions

The registers listed as follows perform the following functions:

- CGM Specifies whether the EPU is a member of the Conditional Group
- TRESET Provides for resetting the STL without resetting the TAP.7 Controller
- FRESET Provides for resetting the functional logic without resetting the TAP.7 Controller
- FRESETR Provides means to determine when a reset of the functional logic is completed
- PWRMODE Defines the Power-Control Mode

Their values are described in Table 18-2, whereas their function is described in the subclauses listed in Table 18-1.

## 18.4.2 Specifications

### Rules

- a) Each subsequent specification in 18.4.2 shall only apply to T1 and above TAP.7s.
- b) The function of the registers listed in Table 18-2 shall be governed by the descriptions in this table.

**Table 18-2 — T1 TAP.7 Controller register descriptions**

Register	R/W	Width	Description	
CGM	W	1	<b>Conditional Group Member</b> —Specifies whether the EPU is a member of the Conditional Group.	
			0	Not a Conditional Group Member
			1	A Conditional Group Member
ECL	W	1	<b>Exit Control Level</b> —Storing this register zeros the ZBS count and unlocks the control level.	
FRESET	W	1	<b>Functional Reset</b> —Requests system logic to generate a functional reset.	
			0	A functional-reset request is not asserted.
			1	A functional-reset request is asserted
FRESETR	R	1	<b>Functional Reset Return</b> —Provides status as to whether there is an outstanding Reset Request to Chip-Level Logic generated by FRESET.	
			0	A functional-reset request is not asserted
			1	A functional-reset request is asserted
PWRMODE	W	2	<b>Power-Control Modes</b> —Specifies interface power-down modes.	
			00	Mode 0—Allow power-down, provided TCK(C) remains a logic 1 for at least one millisecond.
			01	Mode 1—Allow power-down in <i>Test-Logic-Reset</i> state, provided TCK(C) remains a logic 1 for at least one millisecond
			10	Mode 2—Allow power-down in the <i>Test-Logic-Reset</i> state
			11	Mode 3—No TAP.7 Controller power-down
SUSPEND	W	1	<b>SUSPEND</b> —Storing this register zeros the ZBS count and unlocks the ZBS count, and suspends the use of control levels.	
TRESET	W	1	<b>Test Reset</b> —Asserts a test reset to the STL.	
			0	The <i>nsys_trst</i> and <i>sys_tms</i> signals presented to the STL are not influenced by this bit.
			1	The <i>nsys_trst</i> presented to the STL is asserted and the <i>sys_tms</i> signal is a logic 1.
ZBSINH	W	1	<b>ZBSINH</b> —Storing this register inhibits ZBS detection.	

## 18.5 Configurations

### 18.5.1 Description

The chip architect may choose to implement the following T1 TAP.7 options:

- Register read-back
- Functional Reset
- Test Reset
- Power-down modes
- The Chip-Level Logic restores the previous power mode after the TAP.7 Controller power-down

At least one of these functions will be supported to create a T1 TAP.7.

In addition, the chip architect may choose to implement the RSU. When the RSU is implemented, the T0 TAP.7 is implemented with:

- One of the start-up options listed below is implemented:
  - IEEE 1149.1-Compliant
  - Offline-at-Start-up
- None, one, or both of the RSU options listed below
  - Selection Alerts
  - Deselection Alerts

The IEEE 1149.1-Compatible start-up option is implemented when an RSU is not implemented.

The options added to create a T1 TAP.7 are identified in the T1 TAP.7 and the RSU Options field of Configuration Register Zero as shown in Table 18-3.

### 18.5.2 Specifications

#### Rules

- a) Each subsequent specification in 18.5.2 shall only apply to a T1 TAP.7.
- b) The Class Field of Configuration Register Zero shall be set to represent the T1 Class.
- c) A minimum of one of the optional T1 TAP.7 functions shown in bold italics in Table 18-3 shall be implemented.

**Table 18-3 — Specifying the use of T1 TAP.7 optional functions**

<b>Optional functions</b>	<b>Registers implemented to support function</b>	<b>Support identified with Configuration Register bits</b>
<i>Register Read-back</i>	RDBACK0 and RDBACK1	RDBKS
<i>Functional Reset</i>	FRESET	FRESETS
<i>Test Reset</i>	TRESET	TRESETS
<i>Power Modes</i>	PWRMODE	PWRMODES[2:0], PCMR
RSU	N/A	RSUS
Selection Alerts	N/A	SEL_ALERTS
Deselection Alerts	N/A	DSEL_ALERTS

- d) A T1 and above TAP.7's support of the optional TAP.7 functions shown in Table 18-3 shall be described with the Configuration Register Zero bits that are listed in Table 18-3 and specified in 9.10.2.

### Permissions

- e) A T1 TAP.7 may be implemented with any of the options shown in Table 18-3, provided Rule 18.5.2 c) is met.

## 18.6 Start-up behavior

### 18.6.1 Description

The start-up behavior of a T1 TAP.7 is created with the IEEE 1149.1-Compliant start-up option, as this start-up option is the only start-up option that may be implemented. This behavior is subject to the behavior created by the implementation of the Power-Mode Function. The start-up options and the behavior they create are described in 10.3.

### 18.6.2 Specifications

#### Rules

- a) Each subsequent specification in 18.6.2 shall only apply to a T1 TAP.7.
- b) Start-up behavior shall be that specified by the IEEE 1149.1-Compatible start-up option, subject to the behavior created by the implementation of the Power-Mode Function.

## 18.7 Conditional Group Membership

The Conditional Group Membership Register value affects the execution of conditional commands and access to EPU Scan Paths as specified in Clause 9. It also affects the TDOC and TMSC Drive Policy as specified in Clauses 13 and 14. It is managed in the same manner for T1 and above TAP.7s.

## 18.8 Test Reset

### 18.8.1 Description

A common operation when testing modules within a chip is to apply a test reset to an on-chip module followed by module test patterns. It is desirable to provide a test reset that does the following:

- Operates uniformly across chips from multiple chip suppliers
- Provides a means to simultaneously reset the test logic within the STL of all chips within a target system
- Allows customizable test reset characteristics

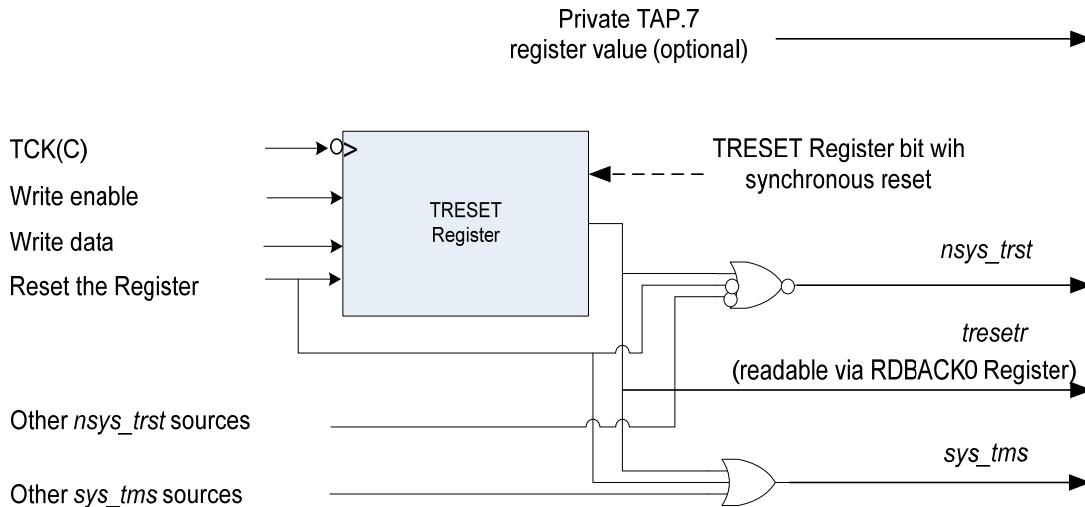
An example of customizable test reset characteristics is the generation of a test reset for some embedded TAPs and not others.

A test reset meeting these criteria may be implemented within the TAP.7 Controller using the TRESET Register. A private TAP.7 Controller register may be used to customize the characteristics of the reset generated by the TRESET Register. The value of these registers does not affect the operation of the TAP.7 Controller.

When the TRESET Register value is a logic 1, then:

- The system Test Reset ( $nsys\_rst$ ) signal is asserted.
- The system Test Mode Select ( $sys\_tms$ ) signal is a logic 1.

When the TRESET Register value is a logic 0, the value of the  $nsys\_rst$  and  $sys\_tms$  signals is determined by other logic signals. A conceptual view of the effects of the TRESET Register on the  $nsys\_rst$  and  $sys\_tms$  signals is shown in Figure 18-3.



**Figure 18-3 — Conceptual view of  $nsys\_rst$  and  $sys\_tms$  generation**

The effects of the TRESET Register may be modified by a private TAP.7 Controller register. When this private register is unimplemented or has a value of zero, the TRESET Register causes the maximum initialization of the STL with the  $nsys\_rst$  and  $sys\_tms$  being a logic 1 value. Otherwise, a nonzero value of the private register may modify the function of the TRESET Register bit.

When the TRESET Register is a logic 1, the STL Scan Path appears broken as the state of the CLTAPC remains *Test-Logic-Reset* while the ADTAPC state progression continues. When the ADTAPC and CLTAPC controllers are both selected as with a T1 TAP.7, the state of these two TAPCs may be re-synchronized to the *Run-Test/Idle* state by:

- An STC1 Command that sets the TRESET Register value to a logic 0
- A stay in the *Run-Test/Idle* state of two or more  $sys\_tck$  periods immediately following the *Update-DR* state of this command

A failure to follow the guidelines is considered a programming error and will result in erroneous system operation. In this case, the ADTAPC and CLTAPC states will not be synchronized as they progress through the state diagram.

Note that when the TRESET bit is a logic 1, the CLTAPC is held in the *Test-Logic-Reset* state making the CLTAPC resources inaccessible.

## 18.8.2 Specifications

### Rules

- a) Each subsequent specification in 18.8.2 shall only apply to a T1 and above TAP.7s, provided the TRESET Register is implemented.
- b) The value of the TRESET Register shall not affect the operation of the EPU.
- c) A TAP.7 Controller shall force the  $nsys\_trst$  value to a logic 0 for the duration of the TRESET Register being a logic 1 when either of the following are true:
  - 1) The private register that is permitted by Permission 18.8.2 h) is not implemented.
  - 2) The private register that is permitted by Permission 18.8.2 h) is implemented and the value of this register is zero.
- d) A TAP.7 Controller shall force the  $sys\_tms$  value to a logic 1 for the duration of the TRESET Register being a logic 1 when either of the following are true:
  - 1) The private register that is permitted by Permission 18.8.2 h) is not implemented.
  - 2) The private register that is permitted by Permission 18.8.2 h) is implemented and the value of this register is zero.
- e) The ADTAPC and CLTAPC shall be re-synchronized following a TRESET Register value of a logic 1, when either of the following are true:
  - 1) All of the following are true:
    - i) The CLTAPC is selected.
    - ii) An STC1 Command sets the TRESET Register value to a logic 0.
    - iii) A stay in the *Run-Test/Idle* state of two or more  $sys\_tck$  periods immediately follows the *Update-DR* state of CP2 of this STC1 Command.
  - 2) All of the following are true:
    - i) The CLTAPC is deselected.
    - ii) An STC1 Command sets the TRESET Register value to a logic 0.
    - iii) The CLTAPC is selected upon entry into the *Run-Test/Idle* state and the stay in this state is two or more  $sys\_tck$  periods.

- f) The value of the TRESET Register shall be readable via the TRESET Bit in the RDBACK0 Register, provided this register is implemented.
- g) When the private register that is permitted by Permission 18.8.2 h) is implemented, its value shall be set to zero by a Type-0-Type-4 Reset.

## Permissions

- h) The characteristics of the test reset generated by the TAP.7 Controller TRESET Register may be customized using a private register.

## 18.9 Functional reset

### 18.9.1 Description

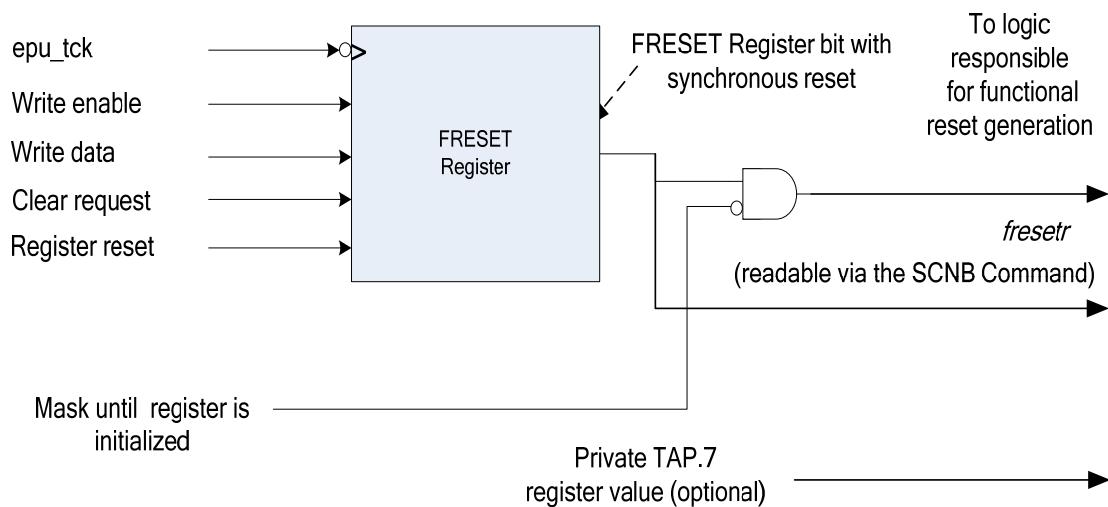
A common operation when debugging a system is applying a functional reset to the system. It is desirable to provide a functional reset that meets the following criteria:

- Operates uniformly across chips from multiple chip suppliers
- Provides a means to simultaneously reset the functional logic of all chips within a target system
- Allows customizable functional reset characteristics

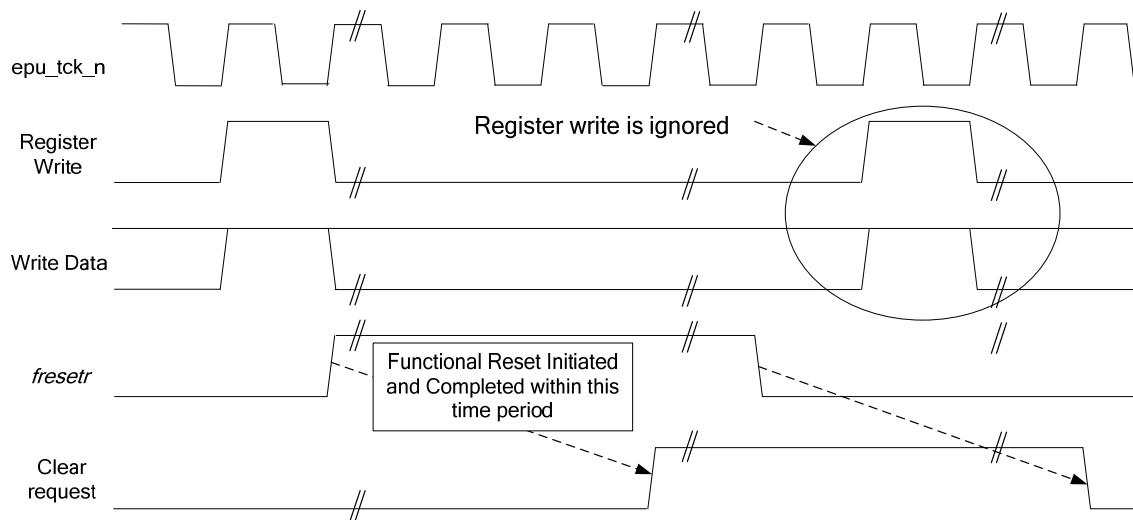
A functional reset meeting these criteria may be implemented within the TAP.7 Controller using the FRESET Register. A private TAP.7 Controller register may be used to customize the characteristics of the reset generated by the FRESET Register. The value of these registers does not affect the operation of the TAP.7 Controller.

It is important the functional reset initiated by the FRESET Register be handled in a manner that does not corrupt the memory contents of the system as this would be inconsistent with the requirements of a debug application utilizing this functionality. The FRESET Register should not be used to directly generate the functional reset as this may cause some chip interfaces to operate incorrectly (e.g., DDR DRAMS).

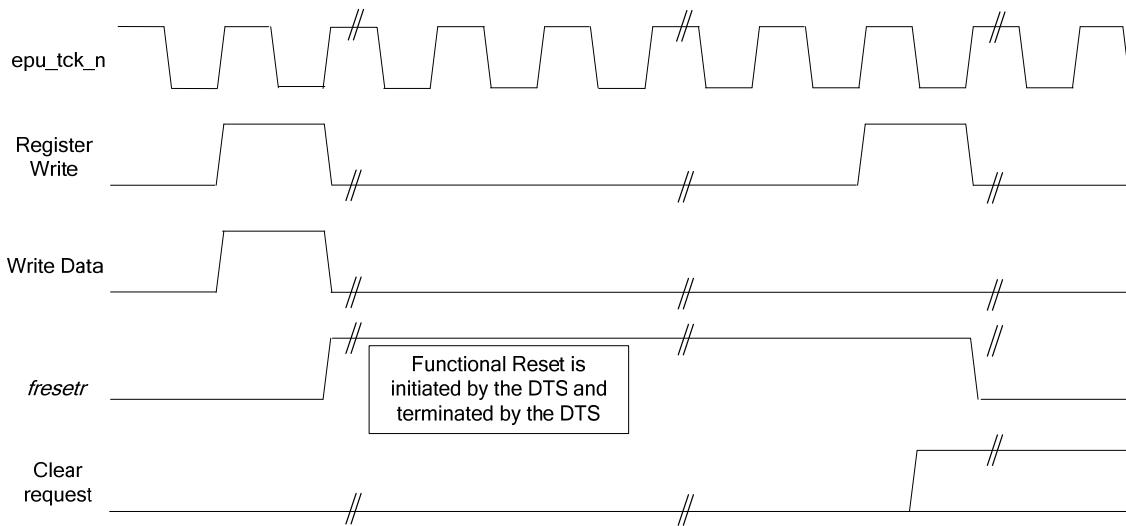
A conceptual view of the functional request generation is shown in Figure 18-4, Figure 18-5, and Figure 18-6. The actual implementation may be different. A change in state of the FRESET Register from a logic 0 to a logic 1 requests the generation of a functional reset by the chip logic normally responsible for this function. This request will be blocked as per Rule 18.9.2 g) to ensure proper chip operation.



**Figure 18-4 — Conceptual view of functional reset request generation**



**Figure 18-5 — Conceptual view of function reset request timing/system-initiated clear**



**Figure 18-6 — Conceptual view of function reset request timing with DTS-initiated clear**

## 18.9.2 Specifications

### Rules

- a) Each subsequent specification in 18.9.2 shall only apply to a T1 and above TAP.7s, provided the FRESET Register is implemented.
- b) The value of the FRESET Register shall not affect the operation of the TAP.7 Controller other than as shown in Figure 18-5 and Figure 18-6.
- c) When the FRESET Register initiates a functional Reset Request as a stimulus to the chip-reset controller per Recommendation 18.9.2 n):
  - 1) A change in state of the FRESET Register value from a logic 0 to a logic 1 shall request the chip-reset controller to generate a functional reset.
  - 2) The FRESET Register value shall be set to a logic 0 when the chip-reset controller indicates it has completed assertion and de-asserted functional reset in response to the request initiated by the FRESET Register.
- d) An STC1 Command shall be capable of storing either a logic 1 or a logic 0 in the FRESET Register.
- e) The value of the FRESET Register shall be readable via the FRESET Bit in the RDBACK0 Register, provided this register is implemented.
- f) The value of the FRESET Register shall be readable via an SCNB Command as shown in Table 9-4.

NOTE 1—The ability to set the FRESET Register bit to a logic 0 is provided for cases where the Chip-Level Logic does not provide clearing the register bit.

- g) The precedence of the operations affecting the FRESET Register shall be governed by Table 18-4.

**Table 18-4 — Precedence of operations affecting the FRESET Register**

Type-0-Type-4 Reset	Chip-Level Clear Request	Write Register	Write Data	Next FRESET value
Yes	X	X	X	0b
X	Yes	X	X	0b
No	No	Yes	1b	1b
No	No	Yes	0b	0b (see note)
No	No	No	X	No change

- h) The request to generate a functional reset shall be inactive when any of the following are true:
  - 1) The TAP.7 Controller is powered-down.
  - 2) A Type-0–Type-4 Reset is asserted.
  - 3) A Type-0–Type-4 Reset has occurred causing a pending reset of the FRESET Register.
- i) A logic 1 FRESET Register value shall cause the reset of all logic that may be initialized with reset signals activated by this register when either of the following is true:
  - 1) The private register that is permitted by Permission 18.9.2 o) is not implemented.
  - 2) The private register that is permitted by Permission 18.9.2 o) is implemented and the value of this private register is zero.
- j) When the FRESET Register is implemented, the TAP.7 Controller shall behave in one of the following manners:
  - 1) The FRESET Register initiates a Reset Request not as a direct functional reset, but as a stimulus to the chip-reset controller when the FRESET Register value changes from a logic 0 to a logic 1.
  - 2) The DTS manages the assertion and de-assertion of a functional reset with writes to the FRESET Register.
- k) When the private register that is permitted by Permission 18.9.2 o) is implemented, its value shall be set to zero by a Type-0–Type-4 Reset.

### Recommendations

- l) A functional reset generated by the FRESET Register should not cause corruption of target system memory.

NOTE 2—Normal memory initialization caused by a functional reset (e.g., a processor initializing data memory) is not considered memory corruption.

- m) When a chip-reset controller does not exist, a logic 1 FRESET Register value directly causes the assertion of a functional reset while a logic 0 FRESET Register does not cause the assertion of a functional reset.
- n) When a chip-reset controller exists, the reset controller should initiate a functional reset upon a change in the state of the FRESET Register value from a logic 0 to a logic 1.

### Permissions

- o) The characteristics of the functional reset generated by the TAP.7 Controller FRESET Register may be customized using a private register.

## 18.10 Power control

### 18.10.1 Description

#### 18.10.1.1 Overview

In power-conscious applications, it is desirable to power-down the TAP.7 Controller and as much associated logic as practical, unless a DTS connection exists and the DTS intends to use the TAP. The TAP.7 architecture accommodates this need with optional TAP.7 power-management features. These features are simply called power management in the remainder of this subclause. The power-management features utilize only the normal TAP signaling and do not increase the TAP.7 signal count. The DTS may use these features to power-up and power-down a TAP.7 Controller after its initial connection to a target system or thereafter.

Four Power-Control Modes (Modes 0–3), specified with the PWRMODE Register, define the conditions initiating power-down. The appropriate conditions for power-down provide a means to specify power-down criteria compatible with:

- A DTS- or TS-sourced TCK(C)
- The DTS stopping TCK(C)
- No power-down

The modes and the criteria for power-down are shown in Table 18-5.

**Table 18-5 — TAP.7 Controller power-control mode use cases**

Power- Control Mode	Power-down when:	
	TCK(C) is a logic 1 for at least 1 ms	ADTAPC state is <i>Test-Logic-Reset</i>
Mode 0	Yes	N/A
Mode 1	Yes	Yes
Mode 2	N/A	Yes
Mode 3	No power-down permitted	

NOTE—The 1 ms time period described in Table 18-5 is generated using a chip clock other than TCK(C).

A fixed set of conditions initiates power-up with these conditions detected at the chip level. The power-management facilities perform a two-step operation after power-up:

- Power-up confirmation
- Power-down initiation

After power-up, the power-management facilities first confirm power-up was neither accidental nor erroneous. If confirmation fails, the TAP.7 Controller is powered-down. Once an appropriate power-up is confirmed, the TAP.7 Controller monitors the TAP.7 state for the conditions specified by the PWRMODE Register. Power-down is initiated when these conditions are met.

The power-management features, when fully implemented, initiate TAP.7 Controller power-down when:

- An erroneous power-up occurs.

- There is no DTS connection at start-up.
- The power-down criteria are met once power-up confirmation has been completed.
- A break in an established DTS connection occurs (in a number of cases).

#### **18.10.1.1.1 Use cases**

The Power-Control Modes support the use cases shown in Table 18-6.

**Table 18-6 — TAP.7 Controller Power-Control Mode use cases**

Power-Control Mode	Mode is compatible with these Test Clock characteristics:		
	Occasionally stopped low	Occasionally stopped high	Free running
Mode 0	Yes	No	No
Mode 1	Yes	Yes	No
Mode 2	Yes	Yes	Yes
Mode 3	No power-down permitted		

The DTS characteristics required to create the conditions required for TAP.7 Controller power-down are shown in Table 18-7.

**Table 18-7 — DTS characteristics required to orchestrate TAP.7 Controller power-down**

Power-Control Mode	DTS characteristics needed for causing TAP.7 Controller power-down
Mode 0	Sources the Test Clock and can stop the Test Clock at a logic 1
Mode 1	
Mode 2	No special requirement
Mode 3	Not possible with this mode

#### **18.10.1.1.2 Power-control options**

The chip architect determines whether all, part, or none of the TAP.7 Controller power-down function is implemented using the following guidelines:

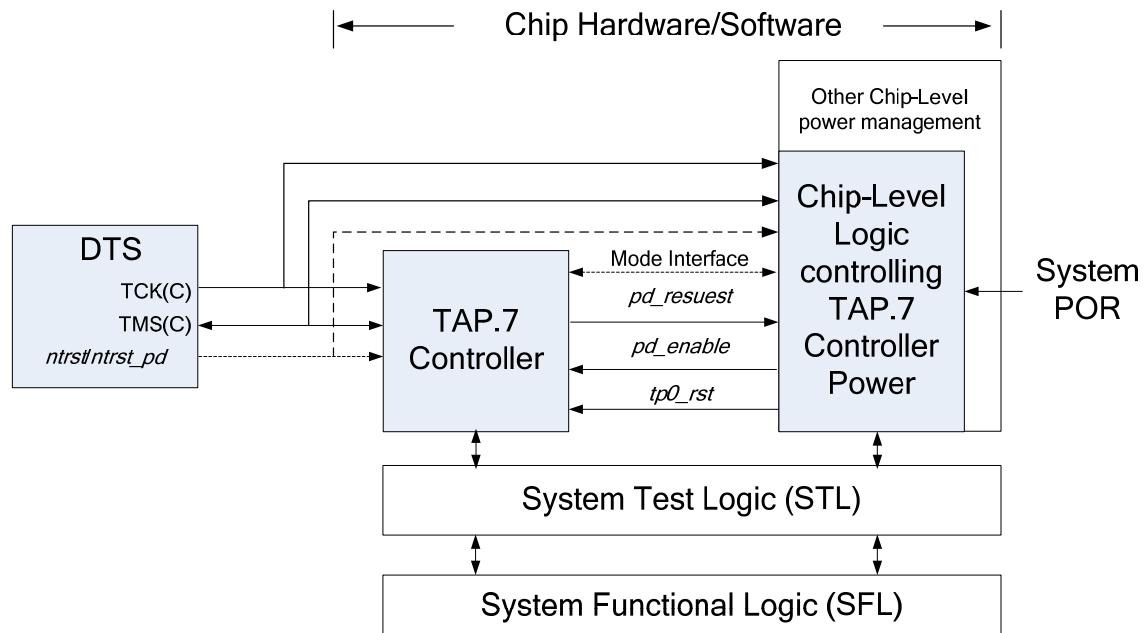
- Any combination of Power-Control Modes 0–2 may be implemented.
- When Modes 0–2 are not implemented:
  - The TAP.7 Controller is powered when any other part of the chip is powered.
  - The PWRMODE Register is not implemented.
- When at least one of Modes 0–2 is implemented, then:
  - The PWRMODE Register is implemented.
  - Mode 3 will be implemented.
- When a specific mode is not implemented and other modes are implemented, then:
  - Its function for power-up confirmation is unaffected.

- Its function for power-up initiation becomes no-power-down-allowed when the TAP.7 Controller is Online.

A chip's power-control features are dependent on the modes implemented. If neither Mode 0 nor Mode 1 is implemented, the feature that checks for TCK(C) inactivity to initiate power-down are not available.

### 18.10.1.1.3 Power management within a typical system

A typical system utilizing TAP.7 power management is shown in Figure 18-7.



**Figure 18-7 — Components involved in TAP.7 Controller power management**

The DTS, the TAP.7 Controller, and the Chip-Level Logic controlling TAP.7 Controller powers interact to provide TAP.7 Controller power management. The Chip-Level Logic controlling TAP.7 Controller power is superior to the TAP.7 Controller and may remain powered when the TAP.7 Controller is powered-down. It powers the TAP.7 Controller up and down in response to asynchronous power-up and power-down requests presented to it. The DTS initiates TAP.7 Controller power-up with TAP signaling monitored by the Chip-Level Logic controlling TAP.7 Controller power and power-down by generating the power-down conditions defined by the Power-Control Mode. When the TAP.7 Controller is powered and the reset to its power-management logic is inactive, it issues a power-down request to the Chip-Level Logic controlling TAP.7 Controller power, provided power-down conditions defined by this mode are met. Optional power-up and power-down enables are supported at the chip level. These enables are superior to other logic initiating power-up and power-down, providing a means to prevent power-up and power-down.

TAP.7 Controller power-up is initiated by a rising edge of the TCK(C) sampling a logic 0 TMS(C) value in combination with a logic 1 nTRST (or nTRST\_PD) value, provided the Chip-Level Power-up Enable permits power-up. The nTRST signal value is considered a logic 1 when neither the nTRST nor the nTRST\_PD signal is implemented. When power-up occurs, the power-up confirmation operation checks for conditions indicating a DTS is not connected or is connected and does not intend to use the TAP. The TAP.7 Controller is powered-down when these conditions are found. An accidental power-up from noise or other factors is followed by an immediate power-down, provided all the power-up confirmation criteria are checked (Mode 0 or Mode 1 is implemented.).

A check for the conditions initiating power-down begins when power-up confirmation passes, with this step called “the power-down initiation test period.” This step relies on the power-up confirmation period to provide time for a *Test-Logic-Reset* to *Run-Test/Idle* state transition. This is important since the *Test-Logic-Reset* state is used as a criterion for the power-down in the power-down initiation period. Without this delay, the power-down criteria would be met as soon as power-up occurs with certain Power-Control Modes.

A brief overview of the functionality provided by the four Power-Control Modes (Modes 0–3) follows. Mode 3 prevents power-down during both power-up confirmation and power-down initiation. Once power-up confirmation is completed, Modes 0–2 specify the same confirmation function. With power-down initiation, each of these modes defines a specific combination of TCK(C) inactivity and the *Test-Logic-Reset* state as causing power-down. This functionality is described further in 18.10.1.5.

The default Power-Control Mode used after power-up is the lowest-numbered mode implemented, unless an option where the power manager may define the default Power-Control Mode is implemented. This is described in 18.10.1.3.6. The Power-Control Mode may be changed after power-up by storing the PWRMODE Register.

#### 18.10.1.1.4 Power-control topics

The remaining material within this subclause includes a description of the following:

- TAP.7 Controller Power-Control Model
- The TAP.7 power-manager’s role in power control
- The DTS’ role in power control
- The TAP.7 Controller’s role in power control
- Examples of power-down

The description of the roles of the TAP.7 power manager and DTS components is limited to their interaction with each other and the TAP.7 Controller.

#### 18.10.1.2 Power-Control Model

##### 18.10.1.2.1 TAP.7 Controller power-management states

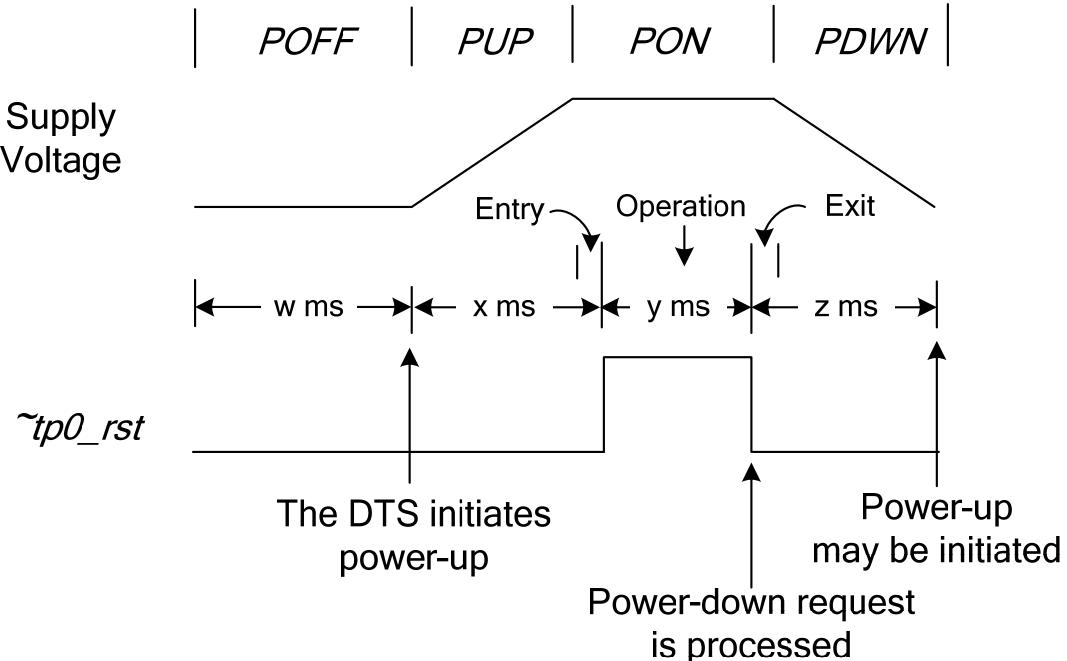
TAP.7 power management requires the use of the TAP.7 Controller Power-Control Model shown in Figure 18-8. This model describes TAP.7 power management with the following four TAP.7 Power-Control states:

- |                              |  |
|------------------------------|--|
| — Power-off ( <i>POFF</i> )  | TAP.7 Controller is powered-off        |
| — Power-up ( <i>PUP</i> )    | TAP.7 Controller is being powered-up   |
| — Power-on ( <i>PON</i> )    | TAP.7 Controller is powered-on         |
| — Power-down ( <i>PDWN</i> ) | TAP.7 Controller is being powered-down |

These states may be created with substates but they cannot be reordered. State transitions occur in the following sequence:

*P OFF* >> *P UP* >> *P ON* >> *P DWN* >> *P OFF*

The hardest of chip resets may create either the *P OFF* state or the *P ON* state.



NOTE:  $w + x + y + z \leq 100$  ms when *w* and *y* are their minimum values

**Figure 18-8 — TAP.7 Controller power-management state progression**

Note the *P ON* state is actually divided into three substates in this figure, which include entry, operation, and exit, with Type-0 Reset active during the entry and exit portions of this state. The activity within the *P ON* state is described further in this subclause.

#### 18.10.1.2.2 Key model attributes

The TAP.7 Controller Power-Control Model constrains the behavior of the TAP.7 Controller, the DTS, and the TAP.7 power manager to provide interoperability. The key attributes of the model are described as follows:

- A Type-0 Reset is asserted as a response to a power-down request shortly before the *P ON* state is exited; it remains asserted while in the *P DWN*, *P OFF*, and *P UP* states and is deasserted shortly after the *P ON* state is entered as shown in Figure 18-8.
- A power-up request may be generated only in the *P OFF* state. It causes a *P OFF* to *P UP* to *P ON* power-control state progression.
- A power-down request may be generated only in the *P ON* state when the Type-0 Reset is deasserted. It causes a *P ON* to *P DWN* to *P OFF* power-control state progression.
- The maximum amount of time needed to traverse the power-control state progression from entry into any state to entry into the same state is less than 100 ms when the stay in the *P ON* and *P OFF* states is minimal (these states are exited immediately when reached).
- DTS events initiating power-up and power-down will be separated by more than the 100 ms to ensure the operation of the DTS, TAP.7 Controller, and TAP.7 power manager remains synchronized.
- The power-control state created by a reset of the TAP.7 power management logic may be either *P ON* or *P OFF*.

- A continuous Type-3 Reset is generated as a result of a power-down request with *sys\_tck* gated off to await power-down. The Chip-Level Power Manager activates the Type-0 Reset in response to the power-down request.

### **18.10.1.3 The chip-level power manager's role in power control**

#### **18.10.1.3.1 Responsibilities**

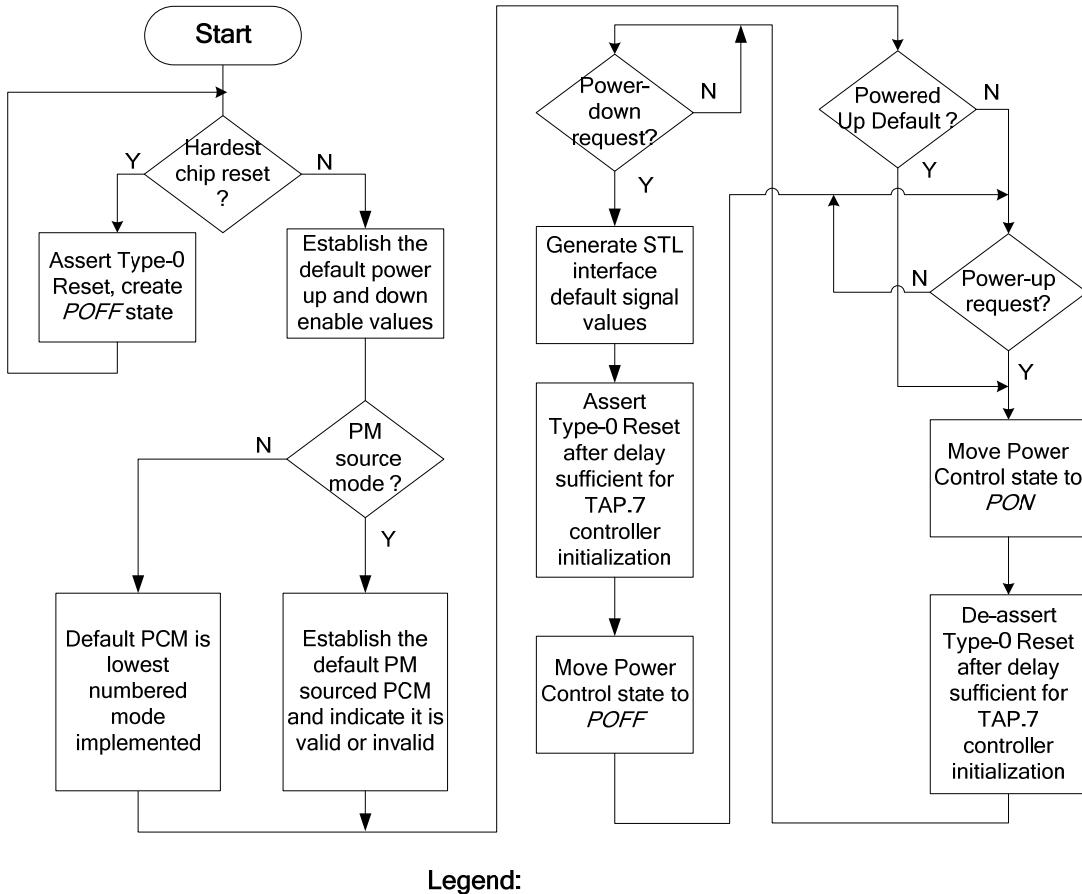
The Chip-Level Power Manager is responsible for the following:

- Implementing the Power-Control Model
- Detecting power-up requests
- Responding to power-down requests
- Providing Chip-Level Power-up and Power-down Enables (when these options are implemented)
- Establishing the default Power-Control Mode (when this option is implemented)

This is not intended to be a complete list of the Chip-Level Power Manager's responsibilities.

#### **18.10.1.3.2 Interaction with TAP.7 Controller**

The TAP.7 power-manager operation is described in Figure 18-9.



**Legend:**

PM == Chip-Level Power Manager      PCM == Power-Control Mode  
Powered Default == TAP.7 controller powered after start-up

**Figure 18-9 — TAP.7 power-manager operation**

#### 18.10.1.3.3 Periods when a Type-0 Reset is asserted

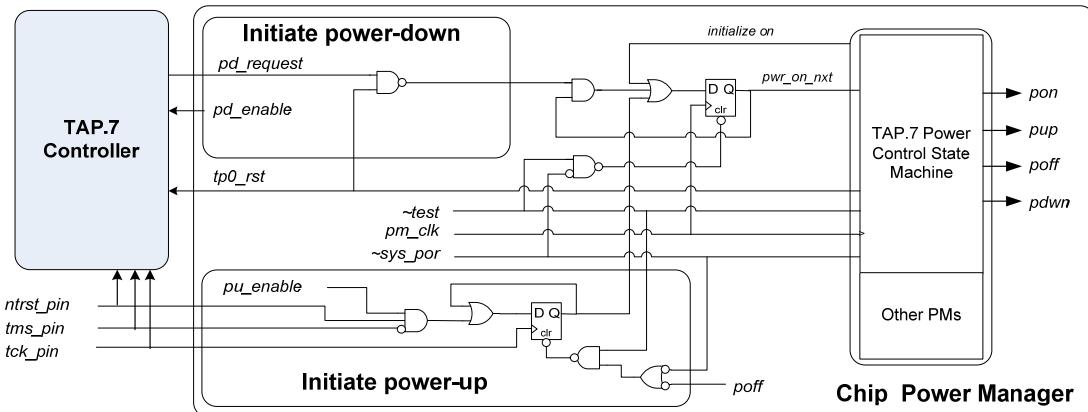
Recall that a Type-0 Reset spans the *PDWN*, *P0FF*, and *PUP* states along with short periods after entering and before exiting the *PON* state. When a Type-0 Reset is active, the *nsys\_trst* signal is a logic 0. It is recommended the remaining STL interface signals have the following values during this period:

- *sys\_tck* signal is a logic 0
- *sys\_tms* signal is a logic 1
- *sys\_tdi* signal is a logic 1

A Type-0 Reset is active for a period of time sufficient to fully initialize the TAP.7 Controller after entering and before exiting the *PON* state. The power-management logic within the TAP.7 Controller is initialized with the Type-0 Reset and no other reset type.

#### 18.10.1.3.4 Handling of power-up and power-down requests

A conceptual view of the TAP.7 power-manager's handling of power-up and a power-down request is shown in Figure 18-10. This figure highlights important aspects of the signaling between the power manager, the TAP.7 Controller, and the DTS. The actual implementation of the power manager may be different.



**Figure 18-10 — Conceptual view of the TAP.7 power manager**

Note that the events initiating power-down and power-up are as follows:

- Interlocked and never occurring simultaneously
- Asynchronous to the clock within the TAP.7 power manager
- Qualified by Chip-Level power-up and power-down enables (*pu\_enable* and *pd\_enable*)

#### 18.10.1.3.5 Chip-Level power-up and power-down enables

The Chip-Level power-up and power-down enables (*pu\_enable* and *pd\_enable*) shown in Figure 18-10 are provided for test, security, and other purposes. These signals are optional, with an unimplemented signal having a logic 1 value (i.e., enabled). Rule 18.10.2 e) describes the changes in both the power-up and power-down enable values that are permitted.

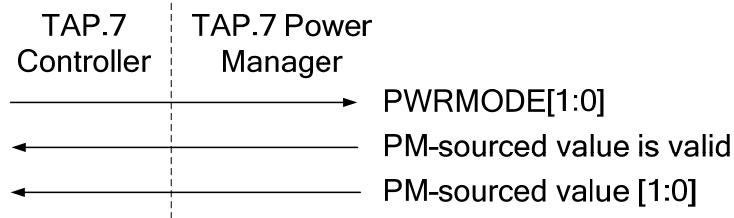
The default *pu\_enable* value is set to a logic 1 or a logic 0 while the TAP.7 power manager asserts a Type-0 Reset. It may be changed by Chip-Level Logic at any time thereafter. In the case where *pu\_enable* is inactive, the TAP.7 Controller merely remains powered when it would otherwise be powered-down.

The behavior of the *pu\_enable* is dependent on the default operation of the TAP.7 Controller. When the default operation is Offline-at-Start-up, the *pu\_enable* has the same characteristics as the *pd\_enable*. **When the start-up operation is IEEE 1149.1-Compliant, IEEE 1149.1-Compatible, or IEEE 1149.7-Protocol Compatible, the *pu\_enable* value can be set to a logic 1 only at chip start-up by the hardest of chip resets.** It can be set to a logic 0 at any time. This restriction prevents power-up of the TAP.7 Controller at times where its operation would not be synchronized to DTS operation. The TAP.7 Controller appears inoperable when powered-down and *pu\_enable* is inactive. This behavior is specified by Table 18-10.

#### 18.10.1.3.6 The default Power-Control Mode

A Type-0 Reset causes the use of a default Power-Control Mode. This value is either supplied by the power-management logic (an option) or the TAP.7 Controller. The PWRMODE Register is initialized with this default value. The TAP.7 Controller-supplied value is the lowest-numbered mode supported. The power-manager supplied value may be any of the four Power-Control Modes.

The power manager chooses either itself or the TAP.7 Controller as the source of the default value when the above mentioned option is implemented. In this case, the power manager selects the value it supplies with a valid signal. This option may be used to maintain the PWRMODE Register value through power cycling of the TAP.7 Controller (save with power-down and restore with power-up) and for other purposes. A typical set of signaling supporting this optional capability is shown in Figure 18-11.



**Figure 18-11 — Typical power-mode interface**

#### 18.10.1.4 The DTS' role in power control

##### 18.10.1.4.1 Responsibilities

TAP.7 Controller's default operation determines the way the DTS powers up the TAP.7 Controller and synchronizes the DTS and TAP.7 Controller operation. A one-step process called "directed power-up" is used when the TAP.7 Controller's default operation is IEEE 1149.1-Compliant, IEEE 1149.1-Compatible, or IEEE 1149.7-Protocol Compatible. The two-step process called "detected power-up" is used when the TAP.7 Controller's default operation is Offline-at-Start-up. These are explained shortly.

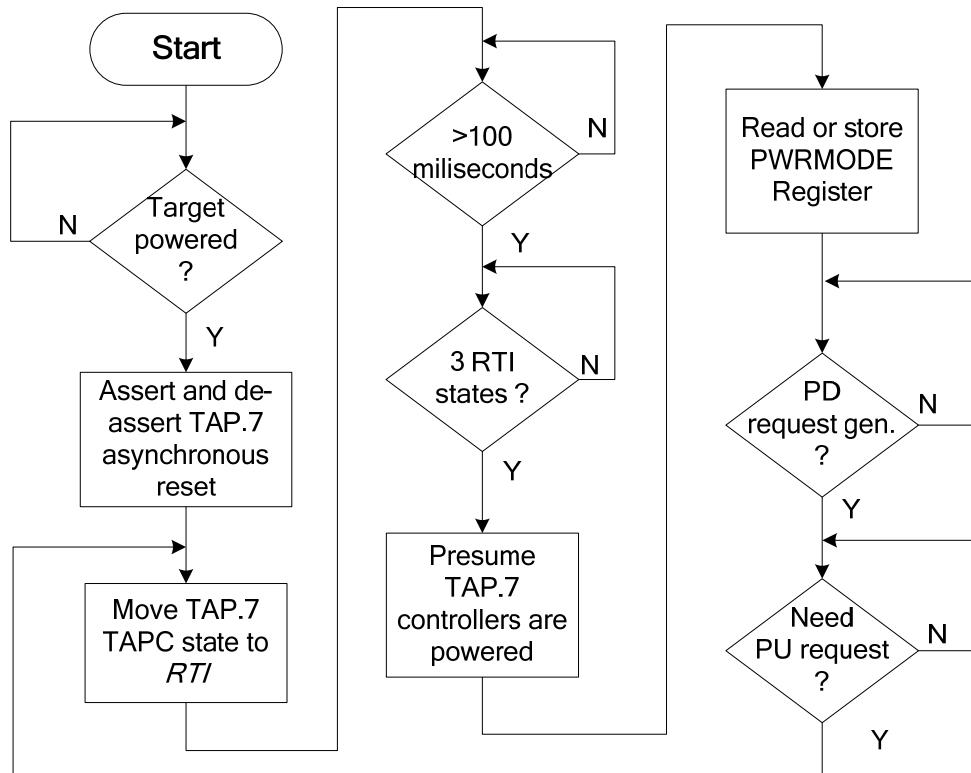
DTS responsibilities also include the following:

- Creating conditions that create power-down requests
- Tracking the power-control state of the TAP.7 Controller
- Using the Power-Control Model in a manner consistent with certain restrictions
- Managing the Power-Control Mode with the PWRMODE Register

##### 18.10.1.4.2 Directed power-up

The directed power-up of a TAP.7 Controller is shown in Figure 18-12. The DTS TAPC State Machine state transitions from *Test-Logic-Reset* to *Run-Test/Idle* causing the power-up of the TAP.7 Controller. The DTS presents a logic 0 TMS(C) value to the TS in concert with this state transition. The DTS-sourced Test Reset Signal, when supported, is a logic 1 at this time. A minimum 100 ms stay in the *Run-Test/Idle* state synchronizes the DTS and ADTAPC State Machine states. These two steps are viewed as a single operation. The signaling associated with the aforementioned state transition causes a TAP.7 Controller power-up. The stay in the *Run-Test/Idle* state provides time for the power-up of the TAP.7 Controller and a state change from *Test-Logic-Reset* to *Run-Test/Idle* once the Type-0 Reset is released. A three-state stay in the *Run-Test/Idle* state is required following the 100 ms period to confirm the intent to power-up. In other words, the DTS creates TAP signaling for a period slightly longer than the 100 ms period required to complete the power-up of the TAP.7 Controller.

Once the TAP.7 Controller is placed Online, the DTS determines the power-down modes supported and either reads the default Power-Control Mode or establishes a Power-Control Mode by storing the PWRMODE Register. With this information, the DTS can determine the power-down criteria.



Legend: RTI == *Run-Test/Idle*

**Figure 18-12 — Directed power-up of a TAP.7 Controller**

#### 18.10.1.4.3 Detected power-up

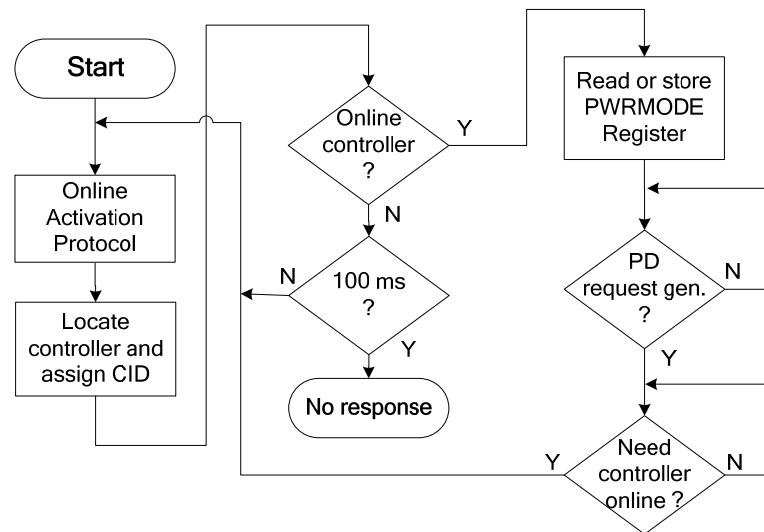
The detected power-up of a TAP.7 Controller with the Offline-at-Start-up option is shown in Figure 18-13. In this case, the power-up of the TAP.7 Controller and its synchronization to the operation of the DTS is a multi-step process. The TAP.7 Controller can be unpowered while ongoing communication with other TAP.7 Controllers occurs. At some point, the chip becomes powered. The signaling that is associated with either the communication with other TAP.7 Controllers or that is created with directed power-up causes the power-up of the TAP.7 Controller. This completes step one.

Because the power-up has no relationship to the DTS TAPC State Machine state, the TAP.7 Controller remains Offline. It moves both the ADTAPC and CLTAPC states to the *Run-Test/Idle* state. This completes step two. Here it awaits step three, a Selection Sequence placing it Online. The TAP.7 Controller is invisible to the DTS before the Selection Sequence placing it Online occurs. This Selection Sequence synchronizes operation of the DTS and ADTAPC State Machine states. This completes step three.

Once the Selection Sequence places TAP.7 Controllers Online, the DTS performs step four, determining the number of Online-TAP.7 Controllers, whether they already have CIDs allocated (CIDs  $\geq$  one), and whether TAP.7 Controllers have been placed Online after being powered-down.

At this point, the DTS detects the TAP.7 Controller's presence when it accepts the assignment of a Controller ID assignment. Hence, the name detected power-up.

Once a powered-down TAP.7 Controller is placed Online and its presence is recognized, the DTS determines the power-down modes supported and either reads the default Power-Control Mode or establishes a Power-Control Mode by storing the PWRMODE Register. With this information, the DTS can determine when power-down criteria are met.



**Figure 18-13 — Detected power-up of a TAP.7 Controller**

#### 18.10.1.5 The TAP.7 Controller's role in power control

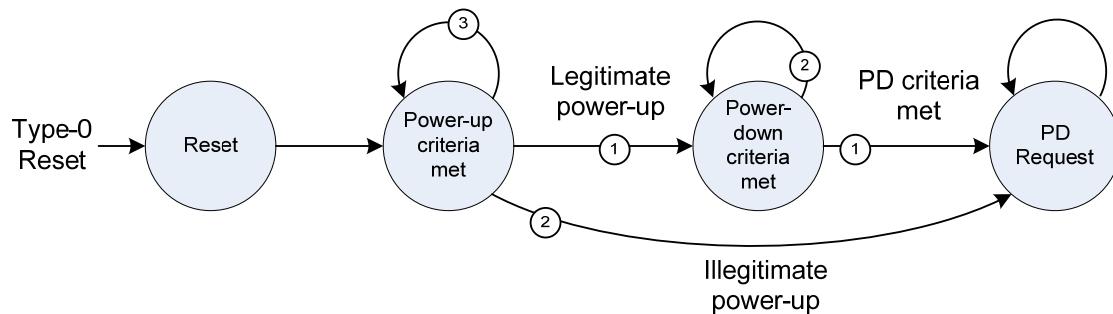
##### 18.10.1.5.1 Responsibilities

The TAP.7 Controller is responsible for the following:

- Initializing the Power-Control Logic when a Type-0 Reset is active
- Confirmation that power-up was the result of a legitimate power-up request
- Checking for power-down criteria being met
- Generating a power-down request
- Generating a continuous Type-3 Reset while a power-down request is generated
- Gating of the *sys\_tck* signal to prevent the advance of the CLTAPC state

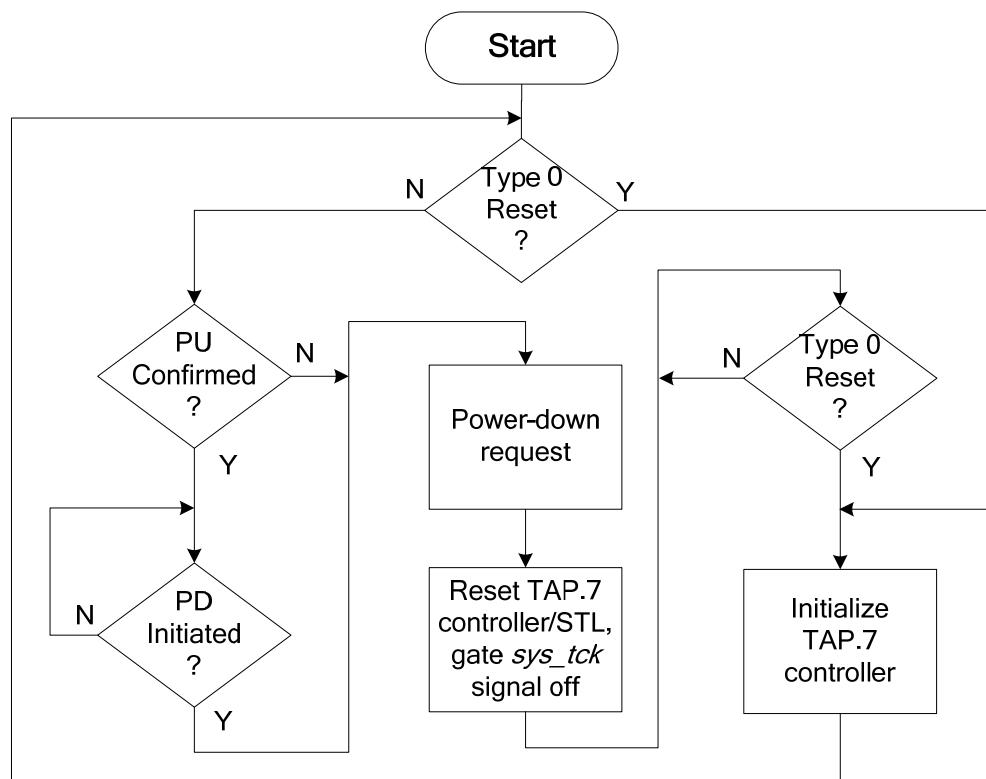
##### 18.10.1.5.2 Operation

The scheduling of the TAP.7 Controller responsibilities is shown in Figure 18-14.



**Figure 18-14 — Conceptual operation of TAP.7 Controller power control**

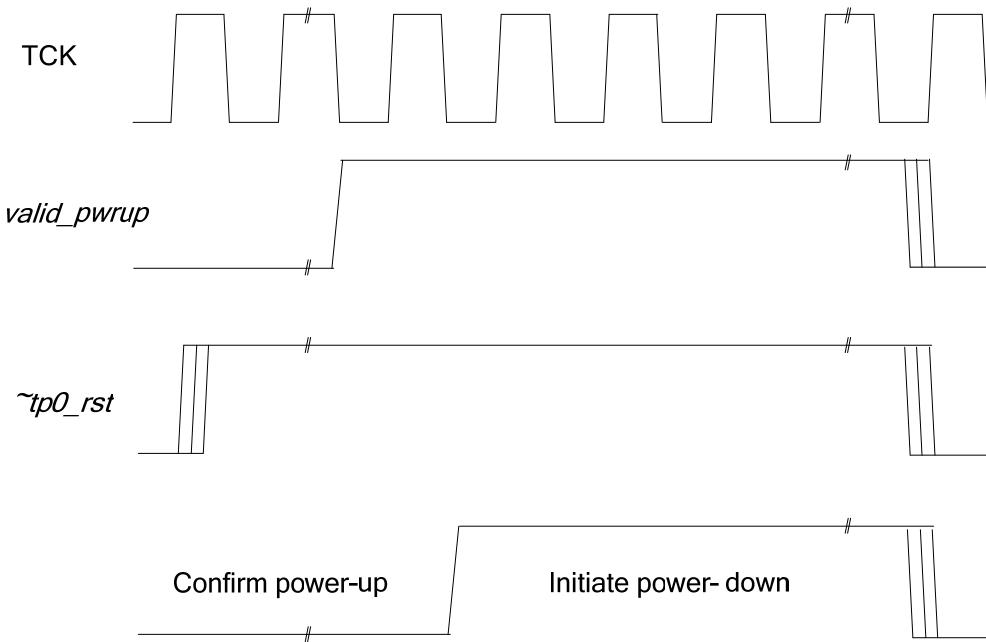
An expanded representation of these responsibilities is shown in Figure 18-15.



**Figure 18-15 — TAP.7 Controller power-management responsibilities**

#### 18.10.1.5.3 Test periods

The power-up confirmation test period begins with the Reset State Machine shown in Figure 10-2 in the *RESA* state and Type-0 Reset inactive. The power-down initiation test period begins with the TCK(C) falling edge following the latest of either the *NO\_RES* Reset State Machine state or the TAP.7 Controller being placed Online as shown in Figure 18-16.



**Figure 18-16 — Power-down criteria selection**

#### 18.10.1.5.4 Power-up confirmation test

The generation of a power-down request requires the Chip-Level Power-down Enable be active and a Power-Control Mode be Mode 0, Mode 1, or Mode 2.

With the power-up confirmation test, the TAP.7 Controller is powered-down for Modes 0–2, if any of the following occur.

- Test Reset is active—The nTRST or nTRST\_PD signal is a logic 0
- Test Clock inactivity—TCK(C) remains a logic 1 for one millisecond
- A TMS a logic 1 value—The expected TAPC state is *Run-Test/Idle*

The check for the DTS-generated TAPC State Machine state progression to the *Run-Test/Idle* state is not performed when the default TAP.7 Controller operation is Offline-at-Start-up as the TAP.7 Controller automatically creates this TAPC state. When this test is performed, the TMS value tested is sampled with the rising edge of the TCK(C).

#### 18.10.1.5.5 Power-down initiation test

With the power-down initiation test, the TAP.7 Controller is powered-down for Modes 0–2, if any of the following occur:

- Test Reset is active—the nTRST or nTRST\_PD signal is a logic 0
- Test Clock inactivity and the TAP.7 Controller is Offline
- Mode 0 and Test Clock inactivity and Mode 0 is supported
- Mode 1 and Test Clock inactivity and the *Test-Logic-Reset* state & Mode 1 is supported
- Mode and & the *Test-Logic-Reset* state & Mode 2 is supported

For both power-up confirmation and power-down initiation, the Test Clock inactivity check is performed, only if Mode 0 and/or Mode 1 are supported, as the hardware performing this check is associated with these modes and automatically passes otherwise.

#### 18.10.1.5.6 Power-down request summary

The conditions generating a power-down request are summarized in Table 18-8.

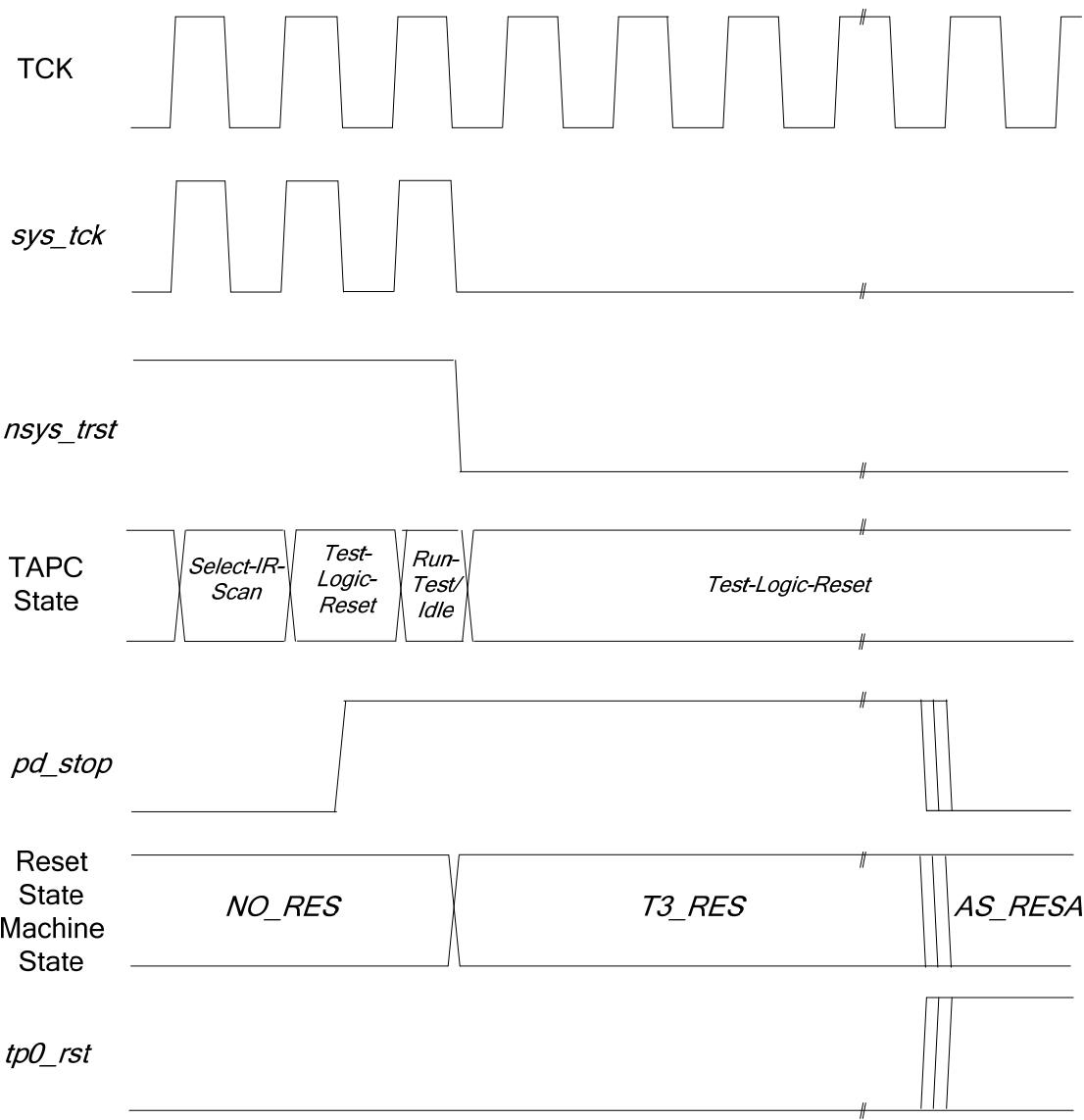
**Table 18-8 — Power-down request generation summary**

<b>Test period</b>	<b>Power-down enabled?</b>	<b>Test Reset active</b>	<b>Power-Control Mode</b>	<b>Mode 0 Supported?</b>	<b>Mode 1 Supported?</b>	<b>Mode 2 Supported?</b>	<b>TCK(C) Inactivity?</b>	<b>Test-Logic-Reset state?</b>	<b>TMS == a logic 1?</b>	<b>Offline?</b>	<b>Offline-at-Start-up?</b>	<b>PD request</b>
Both	Yes	Yes	0–2	x	x	x	x	x	x	x	x	Yes
	Yes	x	0–2	Yes	x	x	Yes	x	x	Yes	x	Yes
	Yes	x	0–2	x	Yes	x	Yes	x	x	Yes	x	Yes
Power-up Confirmation	Yes	x	0–2	Yes	x	x	Yes	x	x	x	x	Yes
	Yes	x	0–2	x	Yes	x	Yes	x	x	x	x	Yes
	Yes	x	0–2	x	x	x	x	x	Yes	x	No	Yes
Power-down Initiation	Yes	x	0	Yes	x	x	Yes	x	x	x	x	Yes
	Yes	x	1	x	Yes	x	Yes	Yes	x	x	x	Yes
	Yes	x	2	x	x	Yes	x	Yes	x	x	x	Yes

A flowchart that includes timing information related to the power-down request (PD\_REQ) is shown in Figure 18-22.

#### 18.10.1.5.7 Awaiting power-down with the TAP.7 Controller operation shutdown

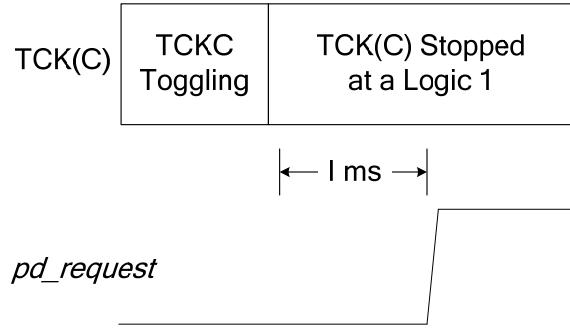
The generation of a power-down request causes a Type-3 Reset coincident with a TCK(C) falling edge. This reset persists until the Type-0 Reset is asserted. Once the Type-3 Reset is active, the clocking of the CLTAPC is inhibited until the Type-0 Reset initializes the TAP.7 Controller power-management logic. Combining the *pd\_stop* signal shown in Figure 18-17 with the *AS\_RESA*, *AS\_RESC*, or *T3\_RES* states (PD\_STOP and any reset) identifies the period where *sys\_tck* is inhibited. This condition persists until the Type-0 Reset is generated as shown in Figure 18-17



**Figure 18-17 — Mode 2 – TAP.7 Controller and STL shutdown**

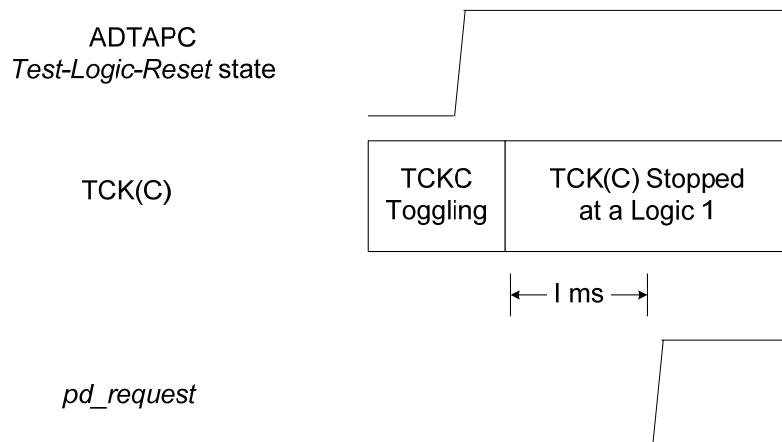
#### 18.10.1.6 Example power-down sequences

Examples of satisfying the power-down criteria are shown in Figure 18-18, Figure 18-19, and Figure 18-20. In Figure 18-18, a 1 ms timing interval begins when the TCK(C) is a logic 1 and is restarted when the TCK(C) is a logic 0.



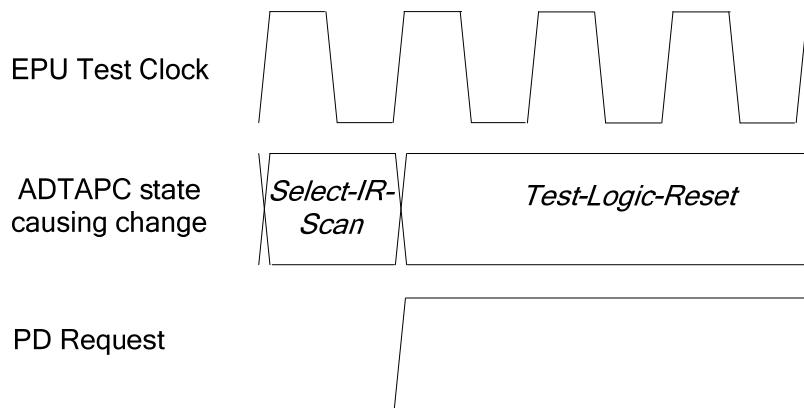
**Figure 18-18 — Mode 0 – with a TCK TIMEOUT initiating power-down**

In Figure 18-19, a 1 ms timing interval begins when TCK(C) is a logic 1 with the *Test-Logic-Reset* state.



**Figure 18-19 — Mode 1 – with the *Test-Logic-Reset* state and TCK TIMEOUT initiating power-down**

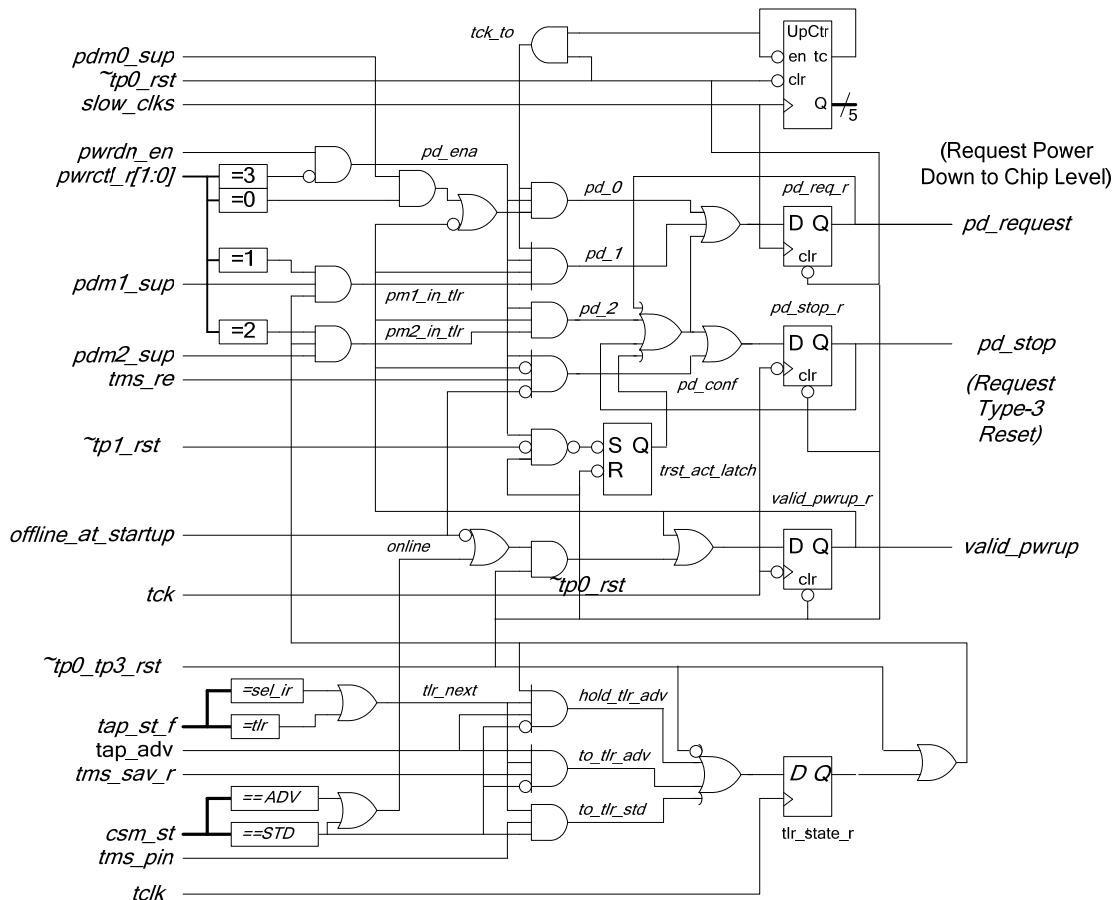
In Figure 18-20, reaching the *Test-Logic-Reset* state initiates power-down.



**Figure 18-20 — Mode 2—with the *Test-Logic-Reset* state initiating power-down**

### 18.10.1.7 An approach to implementing power control

The Power-Control Logic utilizes a number of signals with different timing sources. The power-up and power-down requests are created with a combination of signals from three timing sources (Chip-Level Logic, power manager, and TAP.7 signals). This should be given adequate consideration in the design of the power-down logic. Figure 18-21 provides some insight as to how this logic may be implemented.



**Figure 18-21 — Power-down logic example**

### 18.10.2 Specifications

#### Rules

- a) Each subsequent specification in 18.10.2 shall only apply to a T1 and above TAP.7 where Power-Control Modes 0–2 are not implemented:
  - 1) The PWRMODE Register shall not be implemented.
  - 2) The TAP.7 Controller shall be powered when any other portion of the chip is powered.
- b) Each subsequent specification in 18.10.2 shall only apply to a T1 and above TAP.7 where any of the following Power-Control Modes are supported:
  - 1) Mode 0.
  - 2) Mode 1.
  - 3) Mode 2.

- c) The TAP.7 Controller power shall be managed with the progression of Power-Control States shown in Table 18-9.

**Table 18-9 — Power-control states**

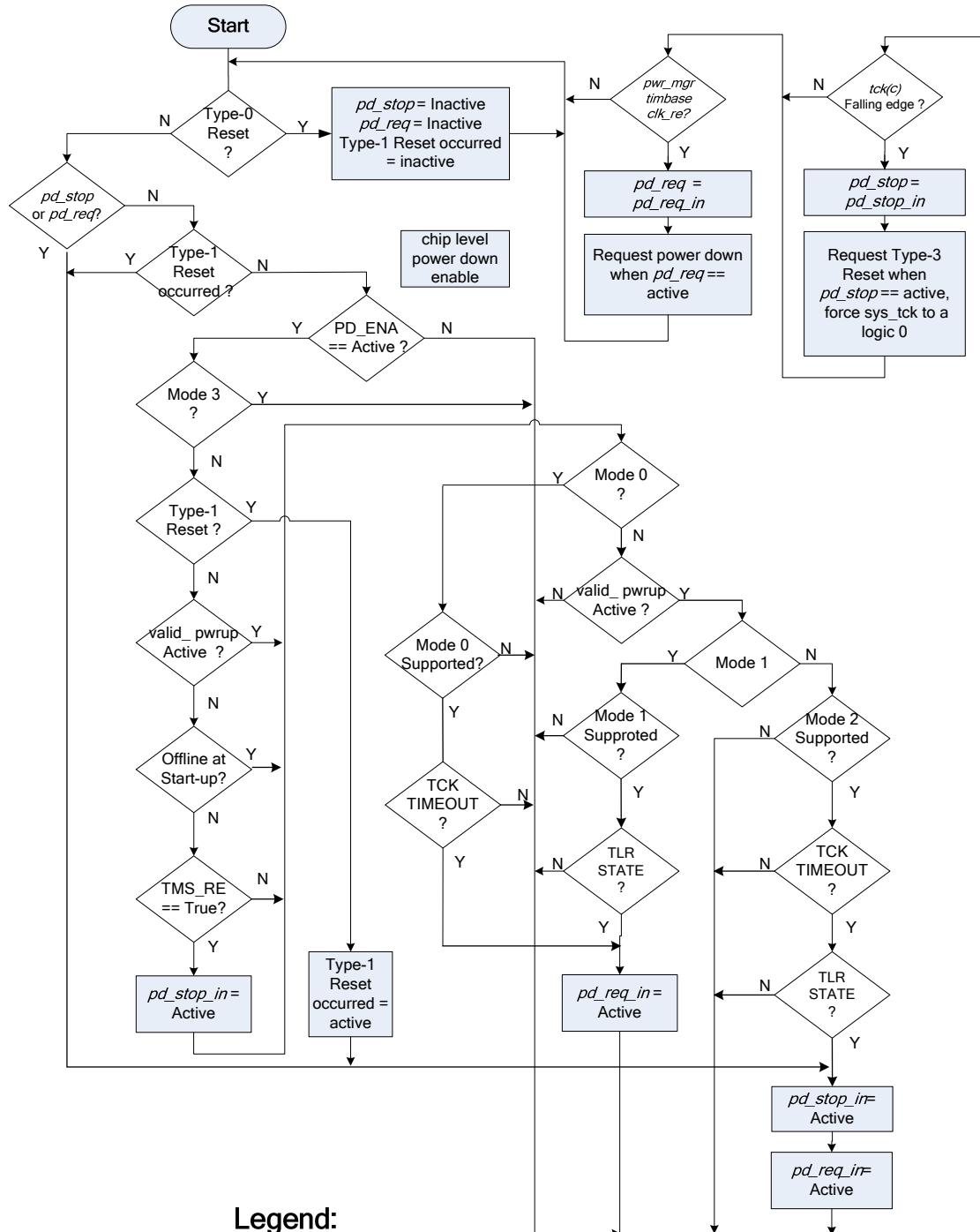
<b>State</b>	<b>Description</b>	<b>Moves to this state:</b>
<i>P<small>O</small>FF</i>	Power is off	<i>P<small>U</small>P</i> when power-up is requested.
<i>P<small>U</small>P</i>	Power-up is in progress	<i>P<small>O</small>N</i> when power-up is completed.
<i>P<small>O</small>N</i>	Power is on	<i>P<small>D</small>WN</i> when power-down is requested.
<i>P<small>D</small>WN</i>	Power-down is in progress	<i>P<small>O</small>FF</i> when power-down is completed.

- d) The time from the exit of any power-control state to the entry into the same state shall be less than 100 ms, provided all of the following are true:
  - 1) A power-up request is generated immediately upon entering the *POFF* state.
  - 2) A power-down request is generated immediately upon entering the *PON* state.
- e) Changes to the Chip-Level power-up and power-down enables shall be governed by Table 18-10.

**Table 18-10 — Changes to power-down and power-up enables**

<b>Chip-Level power-control enable</b>	<b>TAP.7 default operation</b>	<b>Set to a logic 1</b>	<b>Set to a logic 0</b>	
Power-up ( <i>pu_enable</i> )	IEEE 1149.1-Compliant	Only with the hardest of chip resets	Any time	
	IEEE 1149.1-Compatible			
	IEEE 1149.1-Protocol Compatible			
	Offline-at-Start-up			
Power-down ( <i>pd_enable</i> )	IEEE 1149.1-Compliant	Any time		
	IEEE 1149.1-Compatible			
	IEEE 1149.1-Protocol Compatible			
	Offline-at-Start-up			

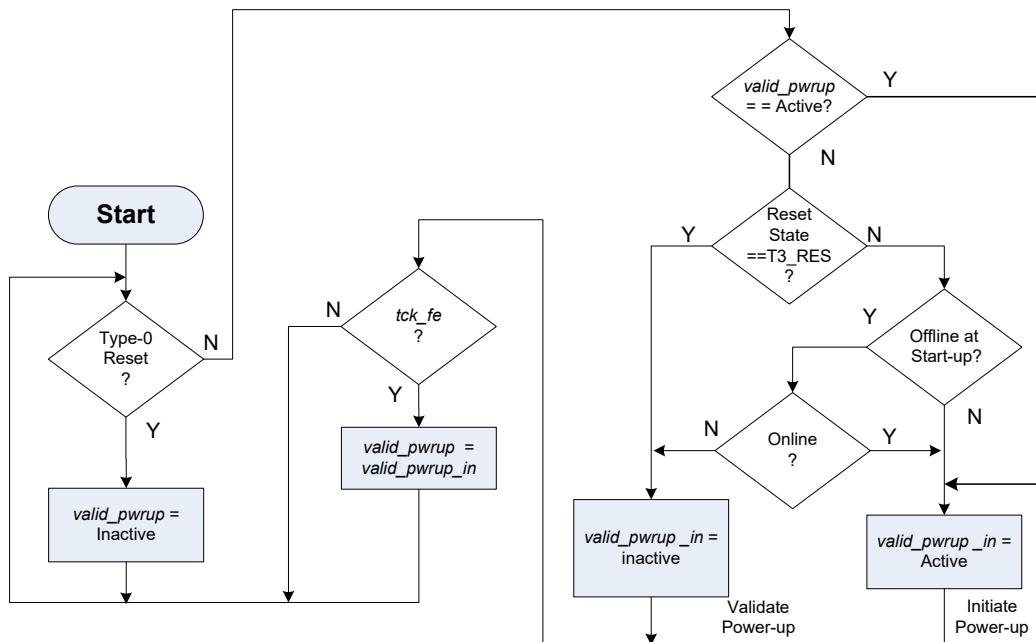
- f) Initiating power-down of the TAP.7 Controller shall be governed by the Figure 18-22.



- 1) Type-1 Reset - the *nrst/nrst\_pd* pin is implemented and has a value of a logic 0
- 2) TCK TIMEOUT - TCK(C) remaining a logic 1 for at least one millisecond
- 3) TLR STATE - The TAPC state (created with the rising edge of the TCK) is *Test-Logic-Reset*
- 4) TMS\_ONE - A TMS(C) value of logic 1 was sampled with the rising edge of TCK(C)
- 5) *pd\_stop* requests a Type-3 Reset and stops *sys\_tck*
- 6) *pd\_req* requests the Chip Level Power Management logic to power down the TAP.7 controller
- 7) *valid\_pwrup* - see Rule 18.10.2 g)

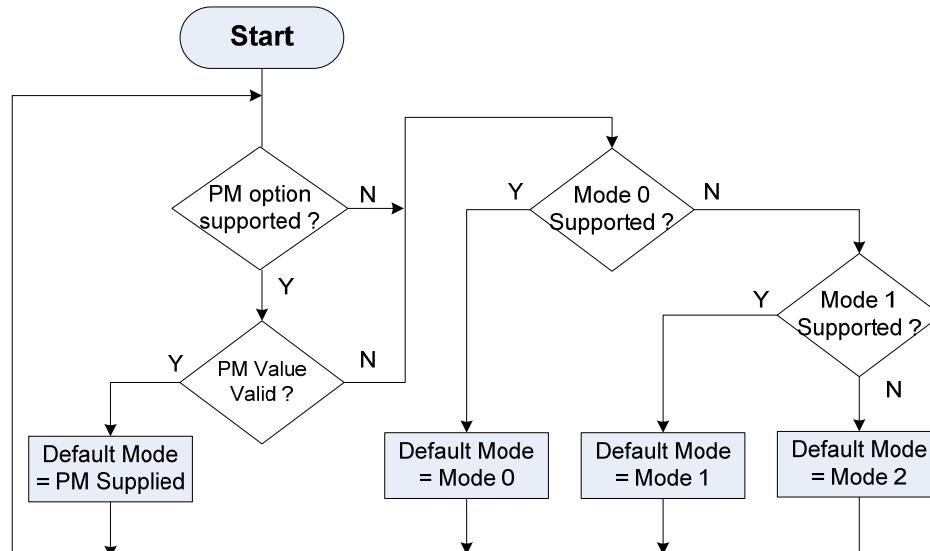
Figure 18-22 — Power-down request generation flowchart

- g) The generation of the *pd\_test* signal value shown in Figure 18-22 shall be governed by Figure 18-23.



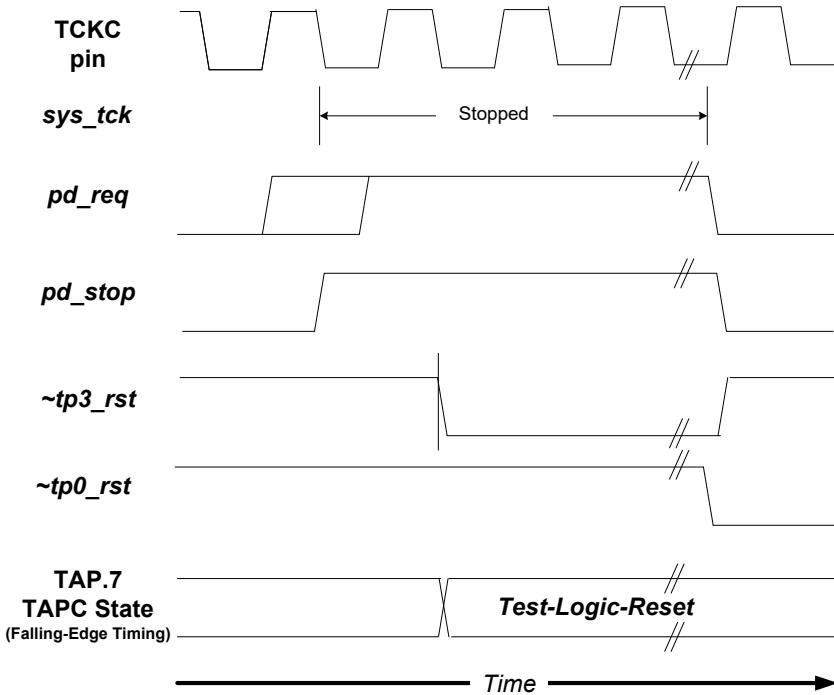
**Figure 18-23 — Power-down criteria selection**

- h) The PWRMODE Register shall be initialized with the default Power-Control Mode while PD\_TEST shown in Figure 18-23 is inactive.
  - i) The default Power-Control Mode shall be developed as shown in Figure 18-24.



**Figure 18-24 — Default Power-Control Mode**

- j) Power-up of TAP.7 Controller shall be requested, provided all of the following are true:
  - 1) The Power-Control state is *POFF*.
  - 2) The Chip-Level Power-up Enable signal is either a logic 1 or is not implemented.
  - 3) The Type-1 Reset signal is either a logic 1 or is not implemented.
  - 4) The rising edge of TCK(C) samples the TMS(C) signal as a logic 0.
- k) The Type-0 Reset signal shall be active in all of the following cases:
  - 1) While in the *PDWN*, *P OFF*, and *PUP* states.
  - 2) For a period of time sufficient to reset the TAP.7 Controller after entering the *PON* state.
  - 3) For a period of time sufficient to reset the TAP.7 Controller before exiting the *PON* state.
- l) The default Power-Control Mode shall be readable via the PWRMODE Register when all of the following are true:
  - 1) The register read-back function is implemented.
  - 2) The register has not been stored with a command following a Type-0 Reset.
- m) When a Type-0 Reset is inactive, the Power-Control Mode shall only be changed by storing the PWRMODE Register with the STC2 Command.
- n) An active state of the *pd\_stop* signal shown in Figure 18-25 shall cause a Type-3 Reset until a Type-0 Reset occurs.
- o) The *sys\_tck* signal shall be forced to a logic 0 when the *pd\_stop* signal shown in Figure 18-25 is active and the Reset State Machine state is any of the following:
  - 1) *AS\_RES A*.
  - 2) *AS\_RES C*.
  - 3) *T3\_RES*.
- p) The *nsys\_trst* signal, when implemented, shall be active while the TAP.7 Controller is powered-down.
- q) A Type-3 Reset shall be generated as a result of a power-down request as shown in Figure 18-25.



**Figure 18-25 — A PD\_STOP initiating a Type-3 TAP.7 Controller reset**

### Recommendations

- r) An active Type-0 Reset signal should create the following STL interface signal values:
  - 1) *sys\_tck*—a logic 0.
  - 2) *sys\_tms*—a logic 1 (the default IEEE 1149.1 pin value).
  - 3) *sys\_tdi*—a logic 1 (the default IEEE 1149.1 pin value).

### Permissions

- s) Power-Control States shown in Table 18-9 may be created with substates.
- t) A PWRMODE Register and its associated function may be implemented with a T1 and above TAP.7.
- u) The TAP.7 Power-Control State created by a reset of the Chip-Level Logic controlling TAP.7 power may be either *PON* or *POFF*.
- v) The Chip-Level Logic controlling TAP.7 power may source the default Power-Control Mode.

## 18.11 RSU operation

### 18.11.1 Description

An RSU implemented with a T1 TAP.7 has the characteristics described in Clause 11, subject to the rules in 18.11.2.

## 18.11.2 Specifications

### Rules

- a) Each subsequent specification in 18.11.2 shall apply to T1 TAP.7.
- b) Rules 16.15.2 b) and 16.15.2 c) shall apply.

## 18.12 Programming considerations

A few important programming considerations are listed as follows:

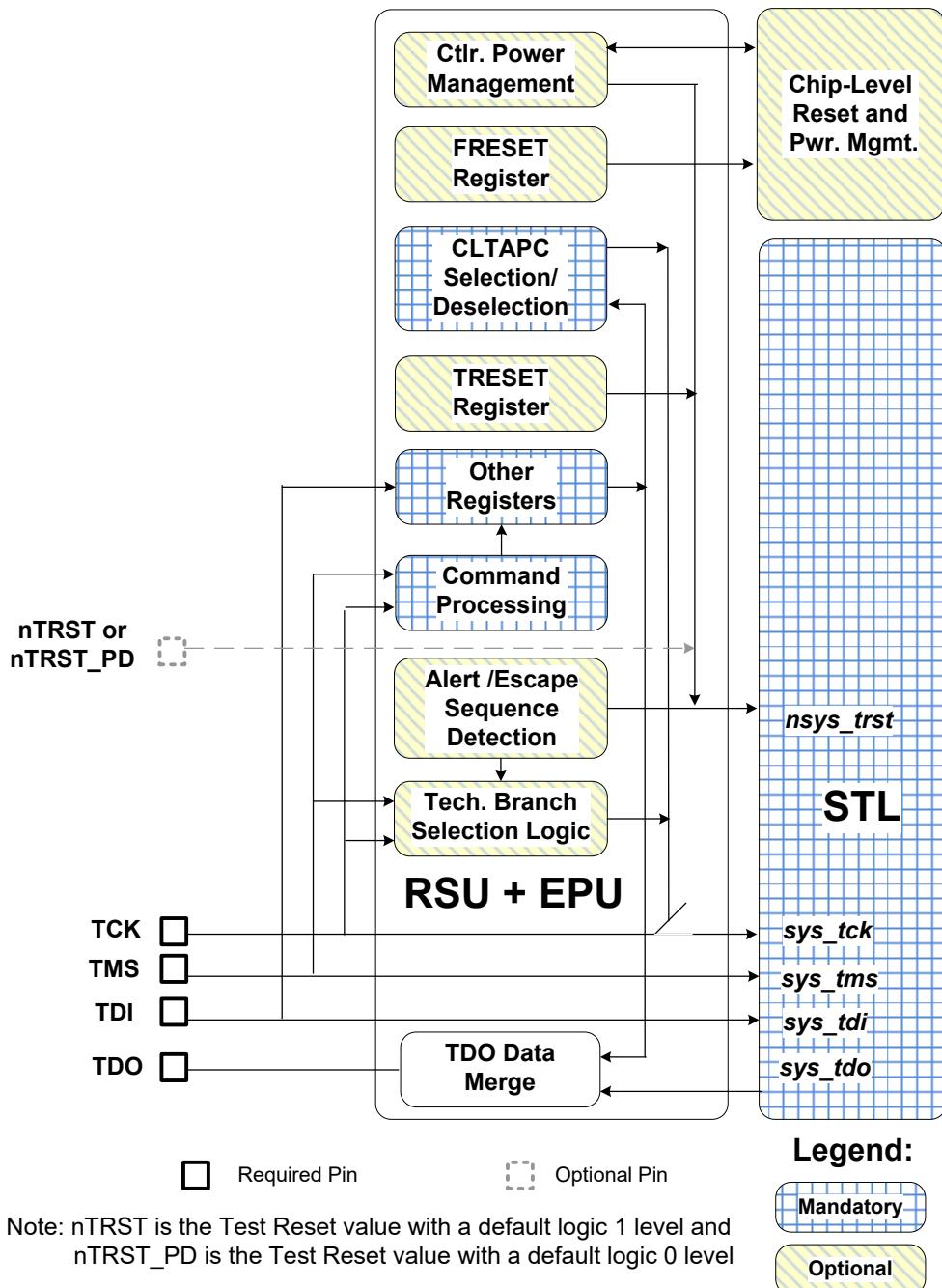
- Structure the DTS software in a manner that separates TAP.7 Controller management and CLTAPC management.
- Move the TAPC state to the *Run-Test/Idle* state following the *Test-Logic-Reset* state for a period of time sufficient to ensure all T1 and above TAPCs are powered.
- The length of the period where the TCK(C) is held at a logic 1 for Escapes will avoid any possibility of invoking power-down of a TAP.7 Controller.
- Refer to Annex D.

With a T1 TAP.7, the STL is a member of the Scan Group.

## 19. T2 TAP.7

### 19.1 Introduction

This clause is applicable to T2 and above TAP.7s. It extends the high-level description of the T2 TAP.7 provided in Clause 5. It provides the rules, permissions, and recommendations for the implementation of a T2 TAP.7. It also provides programming considerations. A number of mandatory register bits and the commands are added to the T1 TAP.7 to create a T2 TAP.7. The block diagram of a typical T2 TAP.7 is shown in Figure 19-1.



**Figure 19-1 — Typical T2 TAP.7**

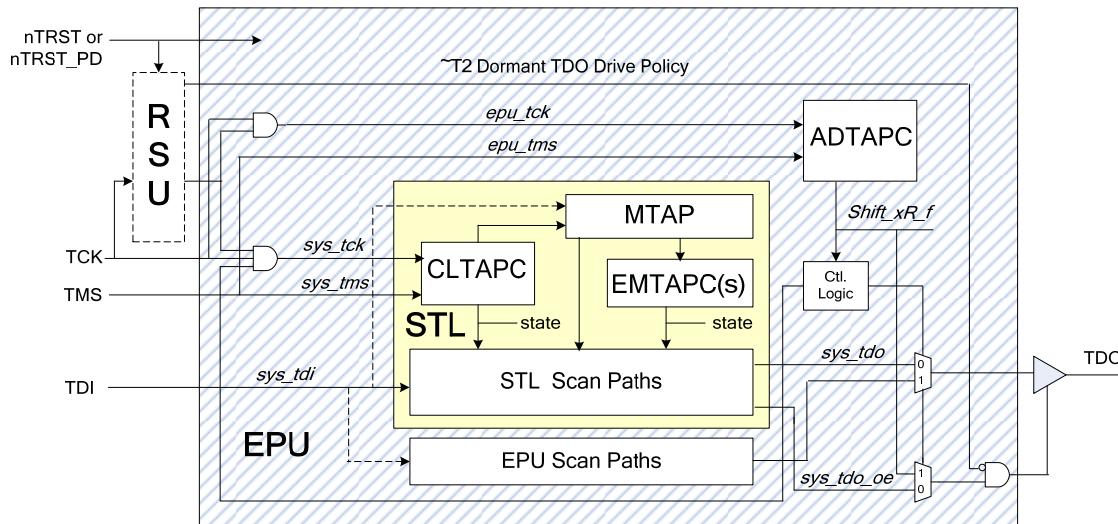
The subject matter within this clause is described in the following order:

- 19.2 Deployment
- 19.3 Capabilities
- 19.4 Register and command portfolio
- 19.5 Configurations

- 19.6 Start-up behavior
- 19.7 Scan formats
- 19.8 STL Group Membership
- 19.9 RSU operation
- 19.10 Programming considerations

## 19.2 Deployment

A high-level block diagram of a deployed T2 TAP.7 is shown in Figure 19-2.



**Figure 19-2 — Deployment of the T2 TAP.7**

## 19.3 Capabilities

### 19.3.1 Inherited

All mandatory capabilities of a T1 TAP.7s are also mandatory for a T2 TAP.7. T1 TAP.7 options are also options for a T2 TAP.7.

The T0 and T1 TAP.7's capabilities inherited by a T2 TAP.7 ensure the following:

- It is controlled in the same manner as lower TAP.7 Classes
- It is compatible with any TAP.1 or TAP.7 Controller operating in a Series Scan Topology

### 19.3.2 New

The new T2 TAP.7 capabilities provide for selecting/deselecting the CLTAPC, as follows:

- At start-up              to provide hot connect protection
- While operating        to bypass the STL and shorten the scan path

A T2 TAP.7 supports two start-up options, IEEE 1149.1-Compliant and IEEE 1149.1-Compatible. With the IEEE 1149.1-Compliant option, the CLTAPC is selected at start-up and the STL Scan Paths are immediately accessible. With the IEEE 1149.1-Compatible option, the CLTAPC state is parked in the *Test-*

*Logic-Reset* state at start-up when the CLTAPC is reset by the *nsys\_trst* signal, and it is parked at its current state when the CLTAPC is not reset by the *nsys\_trst* signal.

Subsequent to start-up, TAP.7 Controller commands in combination with the *Run-Test/Idle* TAPC state provide for the selection of the CLTAPC. A TAP.7 Controller that parks the CLTAPC state in at start-up provides hot-connect protection. The CLTAPC may be deselected/selected by the DTS depending on whether the STL's scan paths are of interest. The DTS may use this function to minimize the length of IR Scan Path in a Series Scan Topology. Deselection of the CLTAPC makes the STL an Idle Group Member while Selection of the CLTAPC makes the STL a Scan Group Member.

## 19.4 Register and command portfolio

### 19.4.1 Description

#### 19.4.1.1 General information

Two new mandatory registers control the features added above those available with the T1 TAP.7. These registers and the commands managing their contents are shown in Table 19-1.

**Table 19-1 — T2 TAP.7 function/new register/command relationships**

Register	Type	Function	Associated command	Described in subclause
<b>SCNFMT</b>	<b>W</b>	<b>Scan Format</b>	<b>STFMT</b>	19.7
<b>SGC</b>	<b>W</b>	<b>Scan Group Candidate</b>	STMC, SCNB, <b>MSC</b>	19.8

NOTE—See Table 9-2 for information regarding the implementation/initialization of these registers. The new registers and commands that are added with the TAP.7 Class are shown in **BOLD** print. These registers are described in Table 19-2 with the commands supporting them described in Table 9-4.

#### 19.4.1.2 Register acronyms

The acronyms for the registers shown in Table 19-1 are shown as follows:

- SCNFMT Scan Format Register
- SGC Scan Group Candidate

#### 19.4.1.3 Effects of a Long-Form Selection Sequence

The SCNFMT Register is loaded with a Long-Form Selection Sequence while other registers in Table 19-1 are not.

#### 19.4.1.4 New register descriptions

The new registers shown in Table 19-1 are described as follows:

- SCNFMT Specifies the scan format being used
- SGC Determines whether the STL is a Scan Group Candidate

## 19.4.2 Specifications

### Rules

- a) Each subsequent specification in 19.4.2 shall only apply to a T2 TAP.7.
- b) The meaning of the values for the new registers added for a T2 TAP.7 shall be governed by Table 19-2.

**Table 19-2 — T2 TAP.7 new register descriptions**

Register	Width	Description	
SCNFMT	5	<b>Scan Format</b> —Specifies T2 TAP.7 selection characteristics.	
		00000b	JScan0: – IEEE 1149.1-Compliant Operation. Requests the selection of the CLTAPC when the <i>Run-Test/Idle</i> state is reached
		00001b	JScan1: – IEEE 1149.1-Compatible Operation. Requests the deselection of the CLTAPC when the <i>Run-Test/Idle</i> state is reached
		00010b	JScan2: – IEEE 1149.1-Compatible Operation. Requests the deselection of the CLTAPC when the <i>Run-Test/Idle</i> state is reached based on the value of the SGC
		00011b	JScan3: – Aliased to the same function as JScan2 Allows the detection of a T2 TAP.7 Controller that is deployed in a Star-4 Scan Topology
SGC	1	<b>Scan Group Candidate</b> —Specifies the Scan Group Candidacy when the scan format is neither JScan0 nor JScan1.	
		0b	Not a Scan Group candidate
		1b	A Scan Group candidate

- c) When an RSU is not implemented, an STFMT Command whose five-bit operand is a value of 00100b–11111b shall not change the value of the SCNFMT Register.

NOTE—This rule specifies a behavior that allows the implementation of an SCNFMT Register with the three MSBs hard-wired to a logic 0 and the two LSBs of the register being programmable. The use of a scan format value other than 00000b–00011b is considered a programming error and may result in erroneous operation.

- d) A T2 TAP.7 shall place the TAP.7 Controller Offline in either the *Run-Test/Idle* or *Select-Dr-Scan* TAPC state, provided all of the following are true:
  - 1) An RSU is implemented.
  - 2) An STFMT Command attempts to store a value other than 00000b–00011b in the SCNFMT Register.

## 19.5 Configurations

### 19.5.1 Description

The DAS Register bit within the T2 TAP.7 options field of the Configuration Register (see Table 9-15) identifies whether the CLTAPC is selected or deselected following the *Test-Logic-Reset* state. The RSU Supported bit in this same register identifies whether the T2 TAP.7 is implemented with an RSU.

### 19.5.2 Specifications

#### Rules

- a) A T2 TAP.7 Controller shall be implemented with the Class Field of the Configuration Register set to the value representing the T2 TAP.7 Class.
- b) A T2 and above TAP.7's support of the optional TAP.7 functions shown in Table 19-3 shall be described with the Configuration Register Zero bits that are listed in this table and specified in 9.10.2.

**Table 19-3 — Specifying the use of T2 TAP.7 configurations**

Optional TAP.7 functions	Registers implemented to support function	Support identified with configuration register bits:
Deselected at Start-up (hot-connect protection)	SCNFMT	DAS
RSU	N/A	RSUS

#### Permissions

- c) A T2 TAP.7 may be implemented with one or more of the options shown in Table 19-3, provided Rule 19.5.2 b) is met.

## 19.6 Start-up behavior

### 19.6.1 Description

The start-up behavior of a T2 TAP.7 is created with the one of the following start-up options:

- IEEE 1149.1-Compliant
- IEEE 1149.1 Compatible

These start-up options and the behavior they create are described in 10.3. The behavior created by these start-up options is subject to the behavior created by the implementation of the Power-Mode Function.

### 19.6.2 Specifications

#### Rules

- a) Each subsequent specification in 19.6.2 shall only apply to a T2 TAP.7.
- b) Start-up behavior shall be that specified by one of the following start-up options, subject to the behavior created by the implementation of the Power-Mode Function:
  - 1) IEEE 1149.1-Compliant.

- 2) IEEE 1149.1 Compatible.

## 19.7 Scan formats

### 19.7.1 Description

The use of scan formats to specify TAP.7 Controller operating characteristics is introduced with a T2 TAP.7. The SCNFMT Register specifies one of three sets of operating characteristics with the four scan formats described in Table 19-4.

**Table 19-4 — T2 TAP.7 Scan Format characteristics**

<b>Scan format</b>	<b>Scan topology supported:</b>	<b>When the <i>Run-Test/Idle</i> state is reached, the CLTAPC is:</b>
JScan0	Series	Selected
JScan1	Series	Deselected
JScan2	Series	Deselected, provided the SGC Register value is a logic 0
JScan3		Selected, provided the SGC Register value is a logic 1

The start-up option determines the default scan format (JScan0 or JScan1) as specified by Table 10-7.

### 19.7.2 Specifications

#### Rules

- a) Each subsequent specification in 19.7.2 shall only apply to a T2 TAP.7s unless stated otherwise.
- b) The scan formats shown in Table 19-4 shall be implemented.
- c) The default scan format at start-up shall be governed by Table 10-7.

## 19.8 STL Group Membership

### 19.8.1 Description

#### 19.8.1.1 Factors affecting group membership

The T2 TAP.7 utilizes a subset of the STL's group management described in 13.8. The STL is either a member of the Scan or Idle Groups. Its group membership is managed with only the following:

- Type-0–Type-4 Resets
- The TAPC state progression
- The SGC Register value
- The use of the System and Control Paths

The Delayed SSD State appears to be continuously be *SSD\_SA\_f*.

### 19.8.1.2 Reset effects

Note that group membership may be forced to the Default Group Membership in one of the following two ways:

- A Type-0 through Type-3 Reset
- The TAPC State Machine state progression reaching the *Test-Logic-Reset* state (a Type-4 Reset)

Because Type-0 through Type-3 Resets asynchronously create the *Test-Logic-Reset* state, asynchronous changes to the group membership are caused by these resets.

A Type-5 Reset (a logic 1 TRESET Register value) converts the *Run-Test/Idle* parking state to *Test-Logic-Reset* parking state, when the CLTAPC state is already parked in *Run-Test/Idle* state.

### 19.8.1.3 TAPC state effects

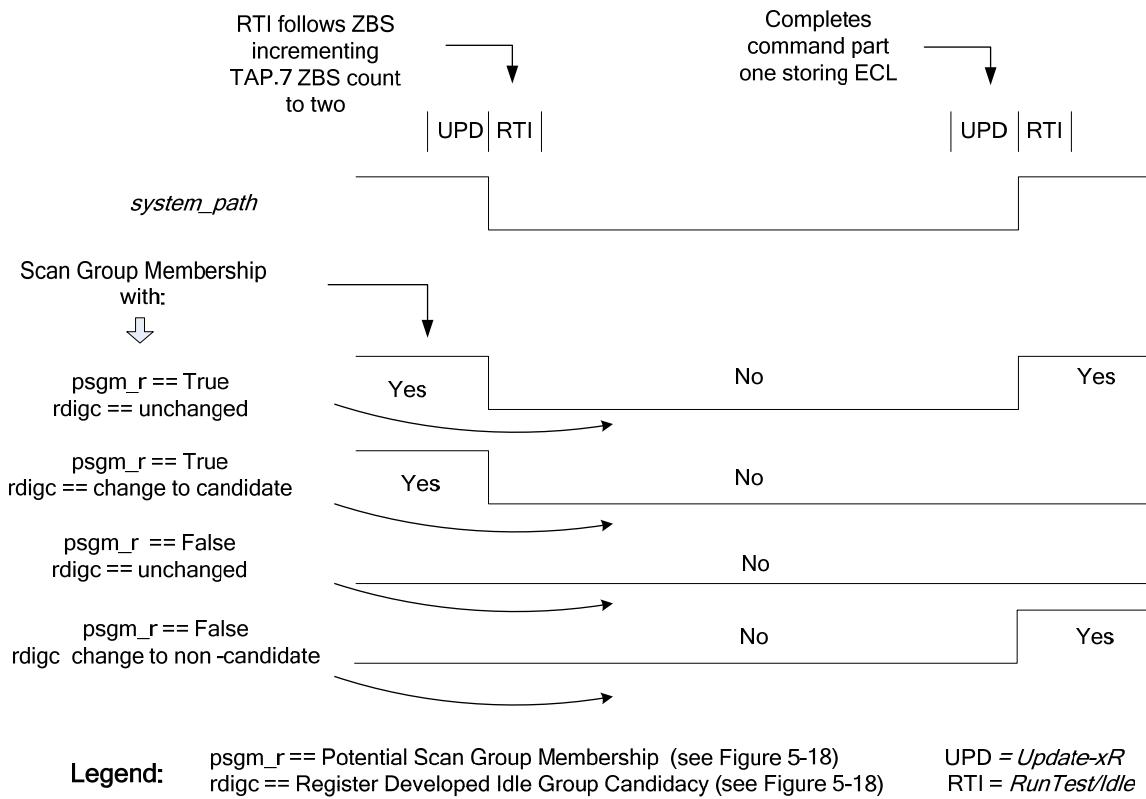
With a T2 TAP.7, group membership changes related to the current Scan Group Candidacy are initiated by one of the following ADTAPC states:

- *Test-Logic-Reset\_f* As specified by the Default Group Membership (*dsgm*)
- *Run-Test/Idle\_f* As specified by the Current Scan Group Candidacy (*csgc*)
- Neither the above states As specified by the Potential Scan Group Member (*psgm\_r*)

Either Idle or Scan Group Membership may be created by these states. Changes in group membership occur simultaneously in all Online TAP.7 Controllers sharing a DTS connection.

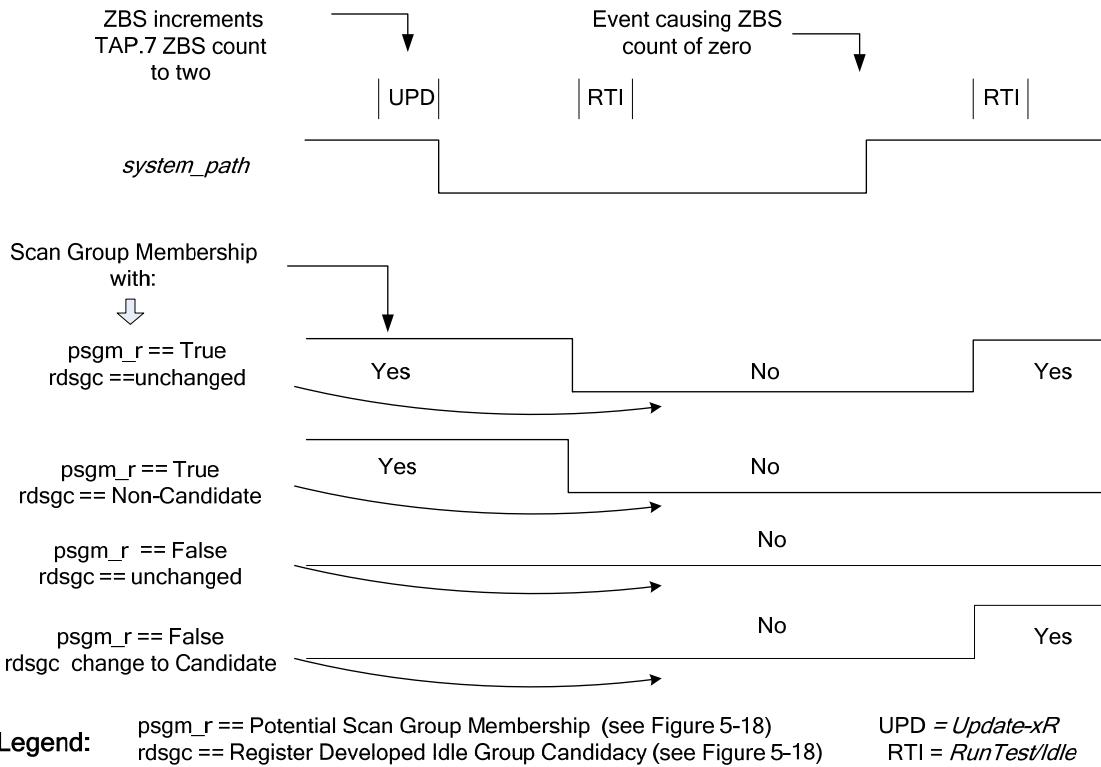
### 19.8.1.4 Control Path effects

When the Control Path is selected (*sys\_path == 0*), Current Scan Group Candidacy is inhibited. Figure 19-3 shows the effects of a *Run-Test/Idle* state following changes in the System Path and Control Path selection.



**Figure 19-3 — Run-Test/Idle/Control Path/Scan Group Candidacy relationships**

Figure 19-4 shows a *Run-Test/Idle* state occurring at a point in time after the Control Path is selected. The combined effects of commands, transitions between use of the System and Control Paths, and the *Run-Test/Idle* state are shown in this figure.



**Figure 19-4 — Skewed Run-Test/Idle/Control Path/Scan Group Candidacy relationships**

#### 19.8.1.5 Parked state/selection relationships

Since *Run-Test/Idle* and *Test-Logic-Reset* are both parking states, selection of the CLTAPC will consider the following two cases:

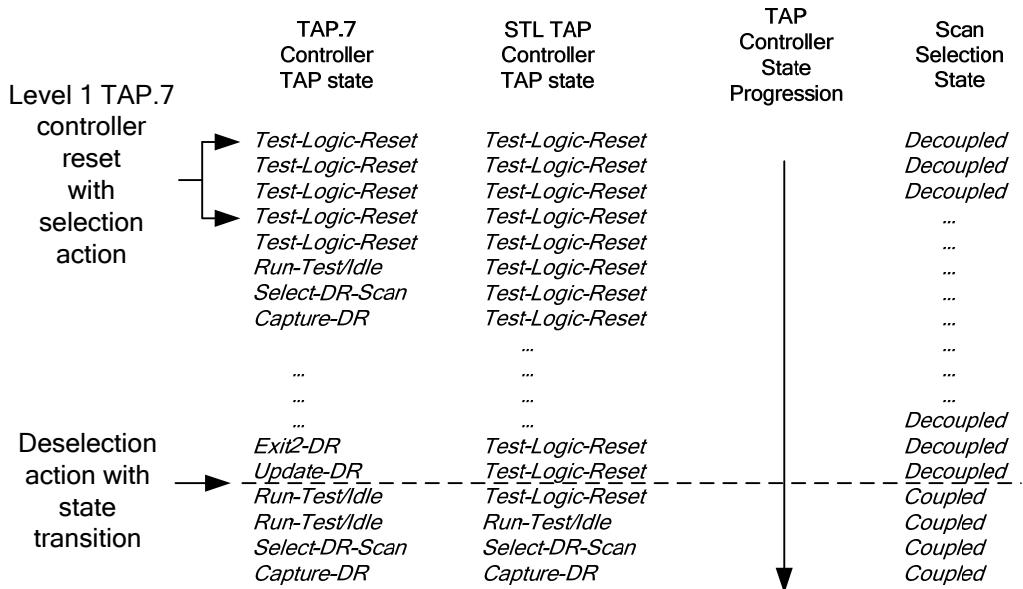
- The ADTAPC and CLTAPC states are both in the *Run-Test/Idle* state.
- The ADTAPC state is *Run-Test/Idle* and the CLTAPC state is *Test-Logic-Reset*.

Proper synchronization of the ADTAPC and CLTAPC states requires the ADTAPC state remains *Run-Test/Idle* state for a minimum of the following:

- One state when the ADTAPC and CLTAPC states are both *Run-Test/Idle*.
- Two states when the ADTAPC state is *Run-Test/Idle* and CLTAPC state is *Test-Logic-Reset*.

Because CLTAPC selection state changes occur when the ADTAPC State Machine state is *Run-Test/Idle*, the ADTAPC and CLTAPC states match in the second *Run-Test/Idle* state, when the CLTAPC parking state is *Test-Logic-Reset* and immediately, when the CLTAPC parking state is *Run-Test/Idle*.

The synchronization of the ADTAPC and CLTAPC states when they are different is shown in Figure 19-5. The *sys\_tck* signal is stopped to park the CLTAPC state in this example. A Type-1 TAPC Reset creates a *Test-Logic-Reset* parking state and parks the CLTAPC state as shown. Subsequent to this, the CLTAPC is selected and a two-state stay in the *Run-Test/Idle* state resynchronizes the ADTAPC and CLTAPC states.

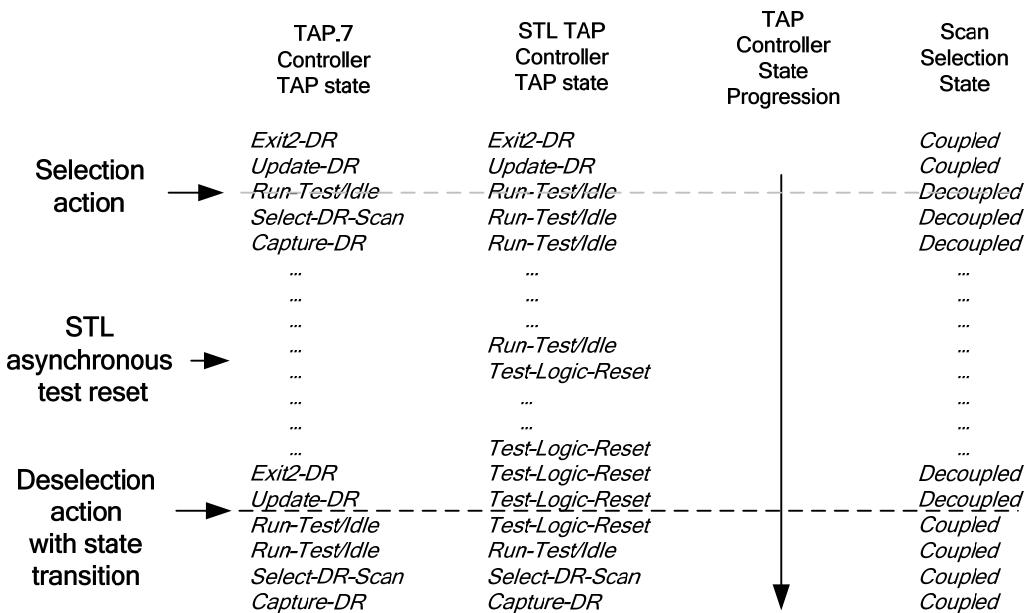


**Figure 19-5 — TAP.7 Controller reset creating *Test-Logic-Reset* parking state**

#### 19.8.1.6 Concurrent CLTAPC and EMTAPC selection changes

Because the selection and deselection of both the CLTAPC and EMTAPCs use the *Run-Test/Idle* and *Test-Logic-Reset* as parking states, it is possible the selection of the CLTAPC and EMTAPC can both occur upon entry into the *Run-Test/Idle* state. When this is the case, an additional two *Run-Test/Idle* TAPC states provide for the simultaneous coupling or decoupling of both the CLTAPC and EMTAPCs. Combinations of CLTAPC and EMTAPC selection and deselection will be completed when the ADTAPC state remains *Run-Test/Idle* for at least four states.

The interaction of *nsys\_trst* and its effect on the synchronization of the ADTAPC and CLTAPC is shown in Figure 19-6. Following deselection of the CLTAPC, the *nsys\_trst* signal is subsequently asserted and deasserted with the TRESET Register while the CLTAPC is deselected. This creates a CLTAPC *Test-Logic-Reset* parking state. Subsequent to this, the CLTAPC is selected and a two-state stay in the *Run-Test/Idle* state synchronizes the ADTAPC and CLTAPC states.



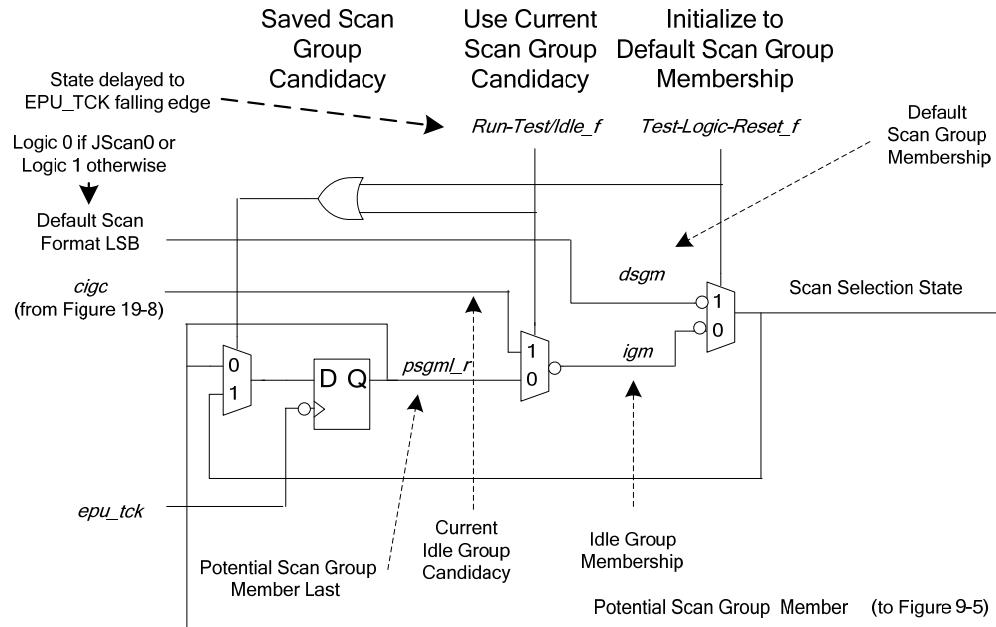
**Figure 19-6 — Re-synchronization of ADTAPC and CLTAPC states following assertion of *nsys\_trst***

**Table 19-5 — Current Scan Group Candidacy creation**

Control Path used?	SCNFMT	SGC	Current Scan Group Candidacy == 1
Yes	x	x	No
No	JScan0	x	Yes
No	JScan1	x	No
No	Other	0	No
No	Other	1	Yes

### 19.8.1.7 An approach to implementing CLTAPC selection with the T2 Class

A conceptual view of CLTAPC selection for a T2 TAP.7 is shown in Figure 19-7.

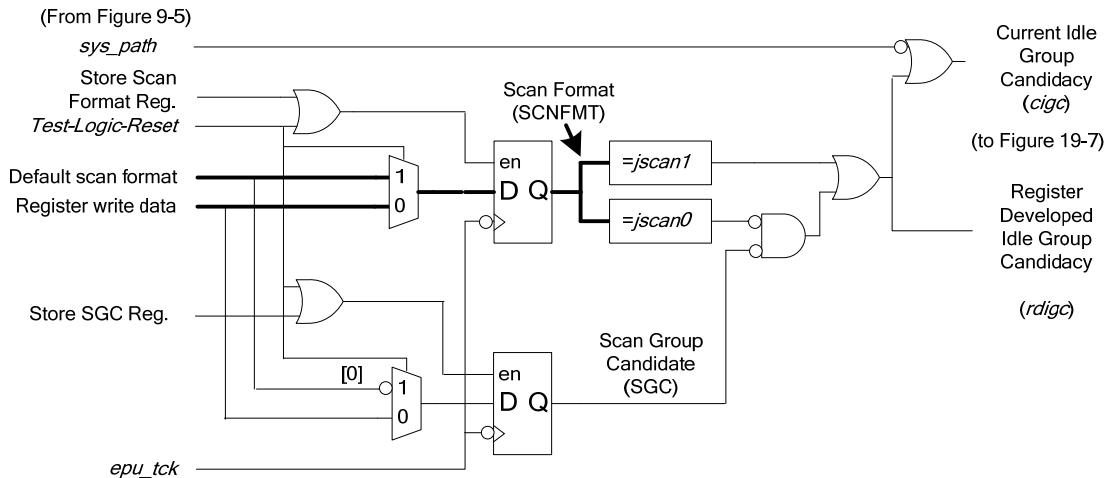


**Figure 19-7 — Conceptual view of T2 TAP.7 Scan Group Membership management**

Current Scan Group Candidacy (*csgc*) is created with following information:

- Scan Format Register value (SCNFMT)
- Scan Group Candidate Register value (SGC)
- System Path use (*sys\_path*)

Commands change the SCNFMT and SGC Register values with a T2 TAP.7. A conceptual view of Current Scan Group Candidacy for a T2 TAP.7 is shown in Figure 19-8.



**Figure 19-8 — Conceptual view of T2 TAP.7 current Scan Group Candidacy development**

The *sys-path* signal masks the Register Developed Scan Group Candidacy value (*rdsgc* shown in Figure 5-18) created with the SCNFMT and SGC Registers.

## 19.8.2 Specifications

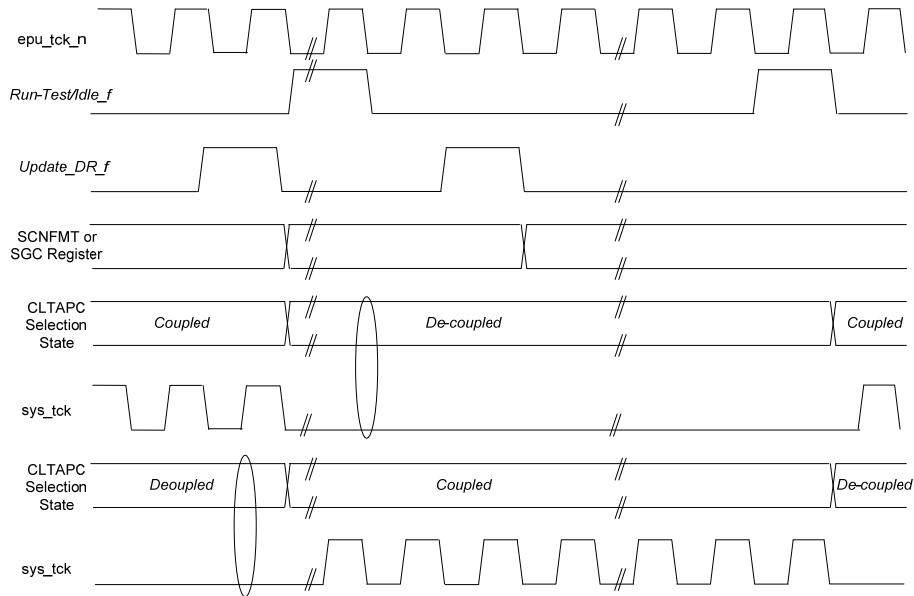
### Rules

- a) Each subsequent specification in 19.8.2 shall only apply to a T2 and above TAP.7s unless stated otherwise.
- b) The number of *Run-Test/Idle* TAP.7 states required to support the synchronization of the ADTAPC and CLTAPC state when the CLTAPC state is parked shall be governed by Table 19-6.

**Table 19-6 — Maximum synchronization stays in *Run-Test/Idle* state**

CLTAPC state	Consecutive <i>Run-Test/Idle</i> states required
<i>Run-Test/Idle</i>	1
<i>Test-Logic-Reset</i>	2

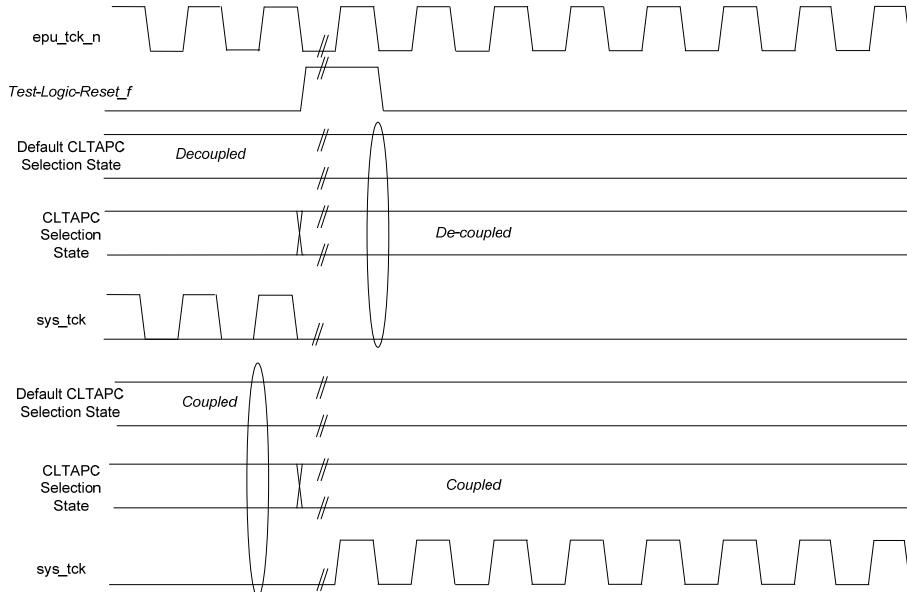
- c) The number of *Run-Test/Idle* ADTAPC states required to support the concurrent synchronization of the ADTAPC, CLTAPC, and EMTAPC states when the CLTAPC and EMTAPC states are parked shall be a maximum of four.
- d) The CLTAPC shall be parked in the *Test-Logic-Reset* state when any of the following are true:
  - 1) All of the following are true:
    - i) The ADTAPC state is *Test-Logic-Reset\_f*.
    - ii) The start-up option is IEEE 1149.1-Protocol Compatible.
  - 2) The TRESET Register value is a logic 1 when the parking of the CLTAPC state occurs.
  - 3) The CLTAPC state is parked when the TRESET Register value is set to a logic 1.
- e) The timing of changes in the CLTAPC selection state initiated by entry into the *Run-Test/Idle* state shall be governed by Figure 19-9.



NOTE—The parking of the CLTAPC state is represented by the gating of **sys\_tck** in this figure. Although this is the recommended method of parking the CLTAPC state, other means of parking of the CLTAPC state may be used.

**Figure 19-9 — CLTAPC selection state change/Run-Test-Idle\_f/register relationships**

- f) The changes in the CLTAPC selection state initiated by entry into the *Test-Logic-Reset* state shall be governed by Figure 19-10 and Table 10-7.



**Figure 19-10 — CLTAPC selection state change/Test-Logic-Reset\_f**

- g) Changes in the value of CLTAPC selection state initiated by entry into an ADTAPC state shall be governed by Table 19-7.

**Table 19-7 — Factors determining the Scan Selection State for a T2 TAP.7**

ADTAPC state	Start-up Option == IEEE 1149.1-Compliant	Control Path?	Scan Format	Scan Group Candidate Register == 1?	CLTAPC selected after reaching state
<i>Test-Logic-Reset</i>	Yes	X	X	X	Yes
	No				No
<i>Run-Test/Idle</i>	X	Yes	X	X	No
	X	No	JScan 0	X	Yes
	X	No	JScan1	X	No
	X	No	Other	No	No
	X	No		Yes	Yes
Others	X	X	X	X	No change

- h) The asynchronous generation of the ADTAPC *Test-Logic-Reset* state shall cause a corresponding asynchronous change in the CLTAPC Selection State per Table 19-7.

### Recommendations

- i) The gating of the clock to the CLTAPC should be used to parking the CL TAPC state.

### Permissions

- j) When the CLTAPC *Test-Logic-Reset* State Machine state has been created by an asynchronous test reset and deassertion of the asynchronous reset occurs while the CLTAPC state is parked, the CLTAPC state may subsequently be parked in either of the following states:
  - 1) *Test-Logic-Reset*
  - 2) *Run-Test/Idle*

## 19.9 RSU operation

### 19.9.1 Description

An RSU implemented with a T2 TAP.7 has the characteristics described in Clause 11, subject to the rules in 19.10.2.

### 19.9.2 Specifications

#### Rules

- a) Each subsequent specification in 19.9.2 shall apply to T2 TAP.7.
- b) Rules 16.15.2 b) and 16.15.2 c) shall apply.

## 19.10 Programming considerations

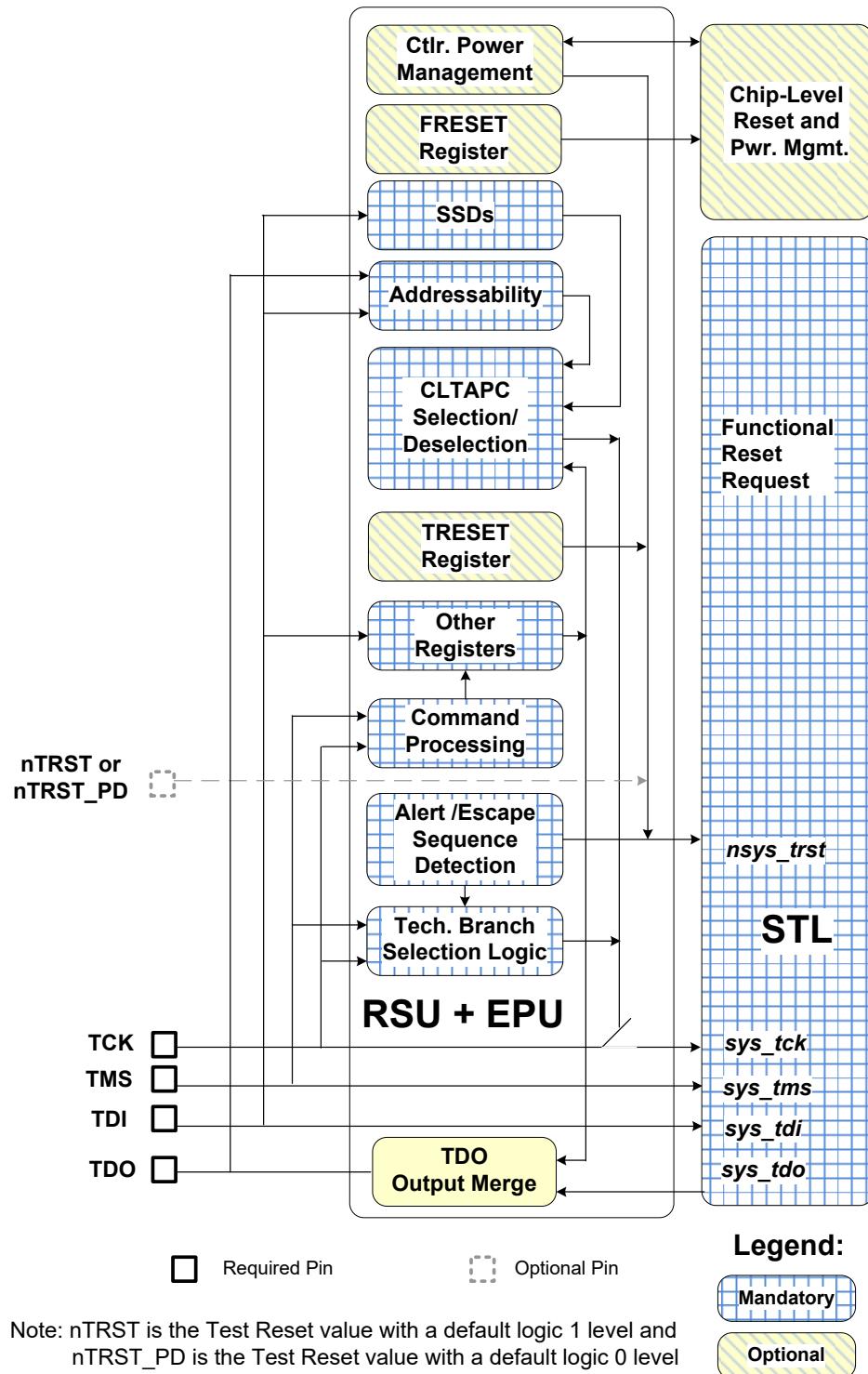
When the *Run-Test/Idle* state is reached and the Control Path is being used, the CLTAPC is deselected as the Current Scan Group Candidacy is inhibited. Subsequent to this, the *Run-Test/Idle* state will also be reached when the System Path is being used to allow the Current Scan Group Candidacy to determine the Scan Group Membership.

This characteristic provides for accessing the System and Control Paths independently. The Control Path may be accessed without regard to the scan format being used. This is significant when using the Advanced Protocol.

## 20. T3 TAP.7

### 20.1 Introduction

This clause is applicable to T3 and above TAP.7s. It extends the high-level description of the T3 TAP.7 provided in Clause 5. It provides the rules, permissions, and recommendations for the implementation of a T3 TAP.7. It also provides programming considerations. The T3 TAP.7 supports operation within a Star Scan Topology. A number of register bits and commands are added to the T2 TAP.7 to create a T3 TAP.7. The block diagram of a typical T3 TAP.7 is shown in Figure 20-1.



**Figure 20-1 — T3 TAP.7**

With a T3 TAP.7, the TDI and TDO signals are named TDIC and TDOC, respectively. This name change reflects the new behaviors that are added to these signals with a T3 TAP.7. The TDOC signal becomes both an input and output while the TDIC signal remains input only. This name change continues with T4 and T5 TAP.7s when the TDIC and TDOC signals are implemented.

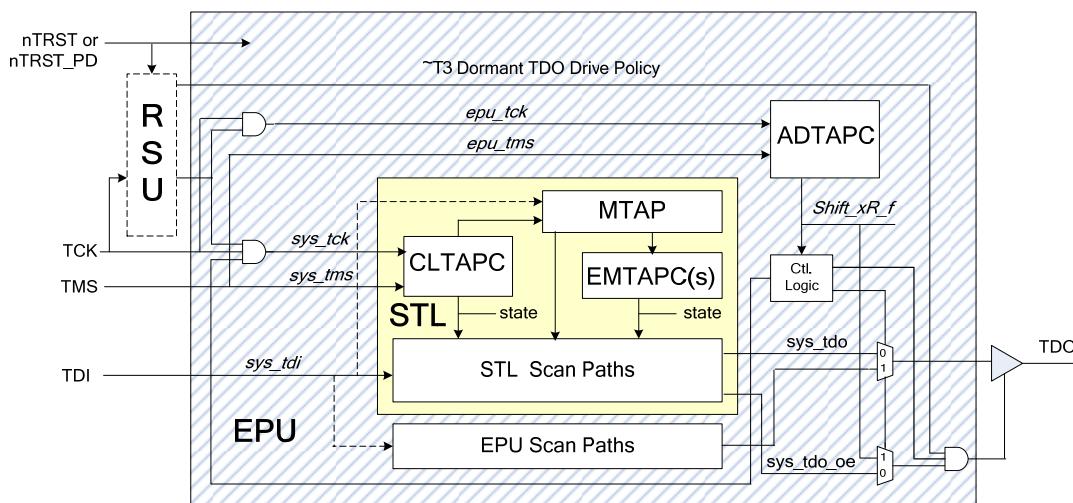
The subject matter within this clause is described in the following order:

- 20.2 Deployment
- 20.3 Capabilities
- 20.4 Register and command portfolio
- 20.5 Configuration
- 20.6 Start-up behavior
- 20.7 Scan formats
- 20.8 TAP.7 Controller Address (TCA)
- 20.9 Aliasing the TCA to a Controller ID
- 20.10 Scan Selection Directives
- 20.11 Scan Topology Training Sequence
- 20.12 Managing STL Group Membership
- 20.13 RSU operation
- 20.14 Programming considerations

A number of the subjects described in this clause are fundamental to the operation of the T4 and above TAP.7s. These subjects include the creation of the TCA, its use in CID allocation, and the use of SSDs to create series scan equivalency. This clause completes the description of the EPU's capability.

## 20.2 Deployment

A high-level block diagram of a deployed T3 TAP.7 is shown in Figure 20-2.



**Figure 20-2 — Deployment of the T3 TAP.7**

## 20.3 Capabilities

### 20.3.1 Inherited

All mandatory capabilities of a T2 TAP.7 are also mandatory for a T3 TAP.7. Optional capabilities supported by T2 TAP.7s are also options for a T3 TAP.7.

The T2 TAP.7 capabilities inherited by a T3 TAP.7 ensure the following:

- It provides IEEE 1149.1-Compliant operation after a Type-0–Type-4 Reset when the default scan format is JScan0, and IEEE 1149.1-Compatible operation when the default scan format is JScan1.
- It uses the command infrastructure provided by the EPU.
- It operates in a manner compatible with any TAP.1 or TAP.7 Controller capable of operating in a Series Scan Topology.
- It provides equivalent operation to that of a T2 TAP.7 with the JScan0–JScan2 Scan Formats.

### 20.3.2 New

The T2 and T3 TAP.7s have equivalent operation with the JScan0–JScan2 Scan Formats and different operation with the JScan3 Scan Format. The new T3 TAP.7 capabilities supporting a Star-4 Scan Topology provide the following:

- Star Scan Topology performance improvements via scan-path length reductions
- The minimum scan-path length between the STL and the DTS
- Direct addressability within a Star Scan Topology
- TAP controller addresses (TCAs) formed from the 27 bits of the chip Device Identification Code and an eight-bit Node Identification Code
- Aliasing of up to 16 TCAs to four-bit CIDs
- Dynamic allocation and reallocation of the CID aliases
- Series-Equivalent Scans in a Star-4 Scan Topology
- Scan Selection Directives to both select and deselect the CLTAPC in the *Pause-IR* and *Pause-DR* states and manage the SGC Register bit in the *Run-Test/Idle* state
- Scan Topology Training Sequence and the Scan Topology Register and to establish the scan topology in which the TAP.7 is deployed
- TDOC drive conflict prevention with the JScan3 Scan Format
- Initiation of actions simultaneously in more than one chip (for example, debug Run/Halt CPU) as the CLTAPC of more than one chip may be selected while TDOC output is inhibited

## 20.4 Register and command portfolio

### 20.4.1 Description

#### 20.4.1.1 General information

The T3 TAP.7 Star Scan Topology support is created with the addition of new registers and commands affecting inherited registers as shown in Table 20-1.

**Table 20-1 — T3 TAP.7 function/new register/command relationships**

Function	Register	Associated commands
<b>Controller ID</b>	<b>CID</b>	<b>CIDA</b>
<b>Controller ID invalid</b>	<b>CIDI</b>	<b>CIDA, CIDD</b>
Scan Format	SCNFMT	STFMT
<b>Scan Selection Directive Enable</b>	<b>SSDE</b>	STMC
<b>Scan Topology</b>	<b>TOPOL</b>	SCNB, STMC, STC2

NOTE—See Table 9-2 for information regarding the implementation/initialization of these registers and the commands that support them. The new registers and commands that are added with the TAP.7 Class are shown in **BOLD** print. These registers are described in Table 20-2.

The operation of the MSC, MCM, CIDD, and STMC Commands is covered in Clause 9 along with a brief description of the CIDA Command. A detailed description of the CIDA Command is found in 20.9.

#### 20.4.1.2 Register acronyms

The acronyms for the registers shown in Table 20-1 are shown as follows:

- CID Controller Identification code
- CIDI Controller Identification code Invalid
- SCNFMT Scan Format
- SSDE Scan Selection Directive Enable
- TOPOL Scan Topology

#### 20.4.1.3 Effect of a Long-Form Selection Sequence

The SCNFMT Register is loaded with a Long-Form Selection Sequence while other registers in Table 20-1 are not. It should be noted that the SSDE Register is not treated as a Global Register for purposes of the Global Register load created by a Selection Sequence as it is expected that the DTS establishes the value of this register following start-up and does not dynamically change its value.

### 20.4.2 Specifications

#### Rules

- a) Each subsequent specification in 20.4.2 shall only apply to T3 and above TAP.7s.
- b) The definition of the CID, CIDI, SCNFMT, and SSDE Registers shall be governed by Table 20-2.

**Table 20-2 — T3 TAP.7 new register definitions**

<b>Register</b>	<b>Width</b>	<b>Description</b>	
<b>CID</b>	4	<b>Controller Identification Code</b>	
		Allocated by the DTS—used to identify the TAP.7 Controller "targeted" for a command	
<b>CIDI</b>	1	<b>Controller Identification Code Invalid</b>	
		0b	The CID is valid, targeted commands are processed.
<b>SCNFMT</b>	4	<b>Scan Format</b> —specifies the selection characteristics of the T3 TAP.7	
		00000b	<b>JScan0</b> —IEEE 1149.1-Compliant operation, requests a coupling action.
		00001b	<b>JScan1</b> —IEEE 1149.1-Compatible operation, requests a decoupling action.
		00010b	<b>JScan2</b> —IEEE 1149.1-Compatible operation, requests a coupling or decoupling action based on the value of the SGC Register.
		00011b	<b>JScan3</b> —Operation that requests a coupling or decoupling action based on the value of the SGC Register. It also enables TDOC signal behavior that prevents drive conflicts in a Star-4 Scan Topology and the use of SSDs.
<b>SSDE</b>	1	<b>Scan Selection Directive Enable</b> —Specifies whether the Scan Selection State may be managed using the Scan Selection Directives.	
		0b	Scan Selection Directives are not enabled.
		1b	Scan Selection Directives are enabled.
<b>TOPOL</b>	2	<b>Topology</b> —Specifies the scan topology in which the TAP.7 Controller is deployed. (See commands affecting this register.)	
		00b	Star-4 Scan Topology, TDOC drive is not permitted.
		01b	Series Scan Topology (Default Topology when Offline-at-Start-up).
		10b	Star-4 Scan Topology.
		11b	Star-2 Scan Topology (Default Topology when Offline-at-Start-up).

- c) The TAP.7 Controller of a T3 TAP.7 shall be placed Offline in either the *Run-Test/Idle* or *Select-Dr-Scan* TAPC state when an STFMT Command attempts to store a value other than 00000b–00011b in the SCNFMT Register.

## 20.5 Configurations

### 20.5.1 Description

All features added with a T3 TAP.7 are mandatory. There are no Configuration Register changes from a T2 TAP.7 other than the TAP.7 Class designation.

## 20.5.2 Specifications

### Rules

- a) With a T3 TAP.7 Controller, the Class field of the Configuration Register shall be set to indicate a T3 TAP.7.

## 20.6 Start-up behavior

### 20.6.1 Description

The start-up behavior of a T3 TAP.7 is created with the one of the following start-up options:

- IEEE 1149.1-Compliant
- IEEE 1149.1 Compatible
- Offline-at-Start-up

These start-up options and the behavior they create are described in 10.3. The behavior created by these start-up options is subject to the behavior created by the implementation of the Power-Mode Function.

### 20.6.2 Specifications

### Rules

- a) Each subsequent specification in 20.6.2 shall only apply to a T3 TAP.7.
- b) Start-up behavior shall be that specified by one of the following start-up options, subject to the behavior created by the implementation of the Power-Mode Function:
  - 1) IEEE 1149.1-Compliant
  - 2) IEEE 1149.1 Compatible
  - 3) Offline-at-Start-up

## 20.7 Scan formats

### 20.7.1 Description

The scan formats introduced with a T2 TAP.7 are also used with T3 and above TAP.7s. There is one difference. With a T3 and above TAP.7, the JScan3 Scan Format provides for operation in a Star-4 Scan Topology, whereas with a T2 TAP.7, this scan format provides the same functionality as the JScan2 Scan Format. The SCNFMT Register specifies one of four sets of operating characteristics with the four scan formats described in Table 20-3.

**Table 20-3 — T3 TAP.7 Scan Format characteristics**

<b>Scan format</b>	<b>Scan topology supported</b>	<b>When the <i>Run-Test/Idle</i> state is reached, the CLTAPC is:</b>
JScan0	Series	Selected
JScan1	Series	Deselected
JScan2	Series	Deselected, provided the SGC Register value is a logic 0 Selected, provided the SGC Register value is a logic 1
JScan3	Star-4	Deselected, provided the SGC Register value is a logic 0 Selected, provided the SGC Register value is a logic 1

The start-up option determines the default scan format (JScan0 or JScan1) as specified by Table 10-7.

## 20.7.2 Specifications

### Rules

- a) Each subsequent specification in 20.7.2 shall only apply to a T3 and above TAP.7 unless stated otherwise.
- b) The scan formats shown in Table 20-3 shall be implemented with the characteristics shown in this table.
- c) The default scan format at start-up shall be governed by Table 10-7.

## 20.8 TAP.7 Controller Address (TCA)

### 20.8.1 Description

The TAP.7 Controller Address is formed by concatenating 27 bits of the mandatory Device Identification Code with an eight-bit node-identification code (NODE\_ID) as described previously. The Device Identification Code (DEVICE\_ID) portion of the TCA identifies different chip types while the node-identification code identifies individual instantiations of the same chip type. The combination of these two values makes a TAP.7 Controller uniquely addressable. More than one TAP.7 Controller having the same TCA within a branch is considered a system-design problem. The features utilizing this address are described shortly.

The IEEE 1149.1 DEVICE\_ID Elements that are utilized are listed as follows:

- 16-bit Part number
- 11-bit Manufacturer Identification Number

The eight-bit NODE\_ID provides for 256 uniquely addressable TAP.7 Controllers with identical DEVICE\_ID Elements. The NODE\_ID is created at the chip level using any one of several methods or a mix of these methods:

- From external signal(s) whose value(s) are latched when a chip hard reset is released
- Fusible elements
- Programmable elements such as EEPROMs
- A register loaded by the application

- Fixed (hardwired)
- Other methods

The method used to generate the NODE\_ID may also be varied for test and normal use if desired.

The Device Identification Code portion of the TCA is a fixed value, while none, part, or the entire Device Identification Code portion of the TCA may be made programmable. When a portion of the NODE\_ID is programmable, the following occurs:

- The programmability begins with the LSB of the NODE\_ID and proceeds toward the MSB.
- A valid flag is implemented to indicate the TCA has reached its final value.
- The valid flag will be in its active state, indicating the TCA has reached its final value before it can be used in any manner.
- The TCA cannot be changed when the valid flag is in its active state.
- The valid flag will only be cleared at chip initialization and will not be cleared by other events once it reaches its active state (the value of the valid flag survives a TAP.7 Controller reset).

## 20.8.2 Specifications

### Rules

- a) Each subsequent specification in 20.8.2 shall only apply to T3 and above TAP.7s.
- b) The format of the TCA shall be governed by Table 20-4.

**Table 20-4 — TCA format**

<b>MSB</b> <b>34      27</b>	<b>26</b>	<b>11</b>	<b>LSB</b> <b>00</b>
<b>Chip-level</b>	<b>Device ID [26:00]</b>		
<b>NODE_ID</b>	<b>Part Number</b>		<b>Manufacturer ID</b>
wwwwwwww	xxxxxxxxxxxxxxxxxx		yyyyyyyyyyyy

- c) The Part Number and Manufacturer ID shown in Table 20-4 shall be a fixed value equivalent to the same fields used for Bits 27:01 of the Device Identification Code specified by IEEE Std 1149.1.
- d) Any programmable NODE\_ID bits shall have bit numbers that are numerically lower than bit numbers for NODE\_ID bits that have fixed values.
- e) When any portion of the NODE\_ID is not a fixed value, all of the following apply:
  - 1) The NODE\_ID shall be considered invalid after a hard Chip-Level reset.
  - 2) Once valid the NODE\_ID shall remain valid and unchanged until the next hard Chip-Level reset.
  - 3) The valid indication of the Node ID shall not be affected by any TAP.7 Controller reset type.
- f) Unimplemented bits of the NODE\_ID shall be a logic 0.

### Recommendations

- g) If it is anticipated that a chip may be deployed in a system with other chips that have the same value for TCA bits 26:00, at least a portion of the NODE\_ID should be programmable.

- h) Where a portion of the NODE\_ID is programmable, it should be set to the same value each time it is made valid.

## Permissions

- i) The NODE\_ID shown in Table 20-3 may be either fixed, programmable, or a combination of both fixed and programmable.

## 20.9 Aliasing the TCA to a Controller ID

### 20.9.1 Description

Because TCAs are 35 bits in length and it is anticipated many systems will have 16 or fewer TAP.7 Controllers, a substantial performance improvement is attained by allocating a four-bit alias to up to 16 TCAs. This alias is called a CID. A number of controller commands incorporate a CID as part of the command's immediate operand. The use of CIDs makes TAP.7 Controller commands targeting a single TAP.7 Controller efficient. Some TAP.7 functions are available using only commands with embedded CIDs.

A TAP controller's CID is either a number between 0 and 15 or is invalid. The CIDI Register records the validity of the CID (a logic 1 indicates an invalid CID). A TAP.7 Controller reset allocates a CID of zero to the TAP.7 Controller with the CIDI Register value set to indicate the CID is valid. Henceforth, the term "allocate a CID" as used herein is used to mean associating a four-bit alias to a TCA, and the term "deallocate a CID" is used to mean removing a four-bit alias associated with a TCA.

The DTS manages the aliasing of the TCAs to the 16 four-bit CIDs with the following commands:

- CIDA
- CIDD

The combination of these commands may be used to do the following:

- Discover TCAs (if they are not already known)
- Alias a CID to a specific TCA, when this address is known
- Simultaneously discover TCAs and alias a CID to them

A CIDD Command may be used to invalidate a single CID or all CIDs. A CIDA Command can allocate a CID by either identifying the TAP.7 Controller that is to be allocated a CID (directed method) or by the TAP.7 Controllers arbitrating for the right to be assigned the CID (undirected method). A TAP controller with a valid CID cannot be assigned a new CID until its CID is invalidated.

The CIDA Command allocates the CID specified by the command operand to a TAP.7 Controller. The CIDD Command deallocates either a single CID or all CIDs based on its operand. The CIDA command specifies one of two CID allocation methods, directed and undirected. With the directed method, the DTS identifies the TAP.7 Controller to be allocated the CID. With the undirected method, the TAP.7 Controller that will be allocated the CID is identified by an arbitration process. A high-level description of these methods was included in the description of the EPU's Enumerate Scan Path in 9.7.3.1.5.

#### 20.9.1.1 CID Allocate Command (CIDA)

The CIDA Command is a three-part command. Command Part One (CP1) specifies the command. Command Part Two (CP2) specifies the:

- Method used to create the TCA that is the target of the CID allocation:

- |                           |   |
|---------------------------|---|
| — The directed method     | TCA to be aliased is identified with CR Scan TDI data |
| — The undirected method   | TCA to be aliased is identified with CR Scan TDO data |
| — CID value being aliased |   |

The CR Scan provides a means to identify the TAP.7 Controller that will be the sole candidate for CID allocation. This DR Scan either identifies the TAP.7 Controller Address that is the target of CID allocation using TDI data with the directed Method of CID allocation or provides a means for the TAP.7 Controllers that are candidates for CID allocation to arbitrate for the right to be allocated the CID by voting on the TDO data values with the undirected method of CID allocation.

All TAP.7 Controllers with a valid TCA and an invalid CID participate in the CID-assignment process with both the directed and undirected Method during the CR Scan. The participant that is assigned the CID is identified with a 36-bit AT conveyed with the CR Scan of the CIDA Command. The AT begins with a logic 0 and is followed by the 35-bit TCA of the TAP.7 Controller that is the target of the aliasing process. When the undirected method is used, the leading zero confirms the presence of a TAP.7 Controller desiring allocation of a CID.

#### **20.9.1.1.1 CID-allocation criteria**

The *Update-DR* TAP controller state of the CR Scan allocates the CID in the CIDA Command operand to the TAP.7 Controller when the following criteria have been met:

- The TAP.7 Class is T3 and above
- The scan format is JScan3 (or a scan format supporting CID allocation with T4 and above TAP.7s)
- The TAP.7 Controller has not been allocated a CID (i.e., CIDI Register == a logic 1)
- The externally supplied AT and the internally generated TAP.7 AT are equal
- The TAP.7 Controller's TCA is valid prior to the *Capture-DR* TAP controller state of the CR Scan
- Exactly 36 AT bits are transferred

When these criteria are not met, the CIDA Command is treated as an NOP and the CID and CIDI Register values are unchanged.

#### **20.9.1.1.2 CID-allocation candidates**

Following the *Capture-DR* TAP controller state of the CR Scan, a TAP.7 Controller with a valid TCA whose CIDI Register value is a logic 1 becomes a “participating candidate” for CID allocation, and it becomes a nonparticipating candidate otherwise. It competes for the right to be allocated the CID but may be disqualified during the competition. It becomes a “nonparticipating candidate” once it is disqualified. Once a candidate is disqualified, it remains disqualified until the next CIDA Command.

#### **20.9.1.1.3 CID-allocation process**

The CID allocation process utilizes the AT as follows during the CIDA Command's CR Scan as follows. The TAP.7 Controller compares the 36-bit externally supplied AT with the internally supplied AT as each sequential bit of the externally supplied AT is presented to the TAP.7 Controller. The format of the internally generated AT is shown in Table 20-5.

**Table 20-5 — AT format**

<b>MSB</b> 35	01	<b>LSB</b> 00
<b>TCA</b>		<b>Zero</b>
TCA [34:00]		0

When a comparison mismatch occurs with AT[n], the TAP.7 Controller is disqualified as a CID allocation participant and no longer participates as of AT[n + 1]. When the internal and external AT values are equal, the TAP.7 Controller becomes the sole CID-assignment candidate, provided its TCA is unique (if this is not the case, there is a system design error).

#### **20.9.1.1.4 Directed CID Allocation**

Directed CID Allocation is supported with the JScan3 Scan Format for T3. With Directed CID Allocation, the DTS sources the AT. With the JScan3 Scan Format, the AT is delivered as TDI data with the TDIC signal. The externally supplied TDI data is compared with the CIDA data, as described previously.

#### **20.9.1.1.5 Undirected CID Allocation**

Undirected CID Allocation is supported with the JScan3 Scan Format for T3 and above TAP.7s. With Undirected CID Allocation, the TS generates the AT value during the *Shift-DR* states of the CIDA CR Scan. The AT of all CID allocation participants is output as the TDO data in a manner that Wire-ANDs the TDO data of the TAP.7 Controller. Only participating candidates affect the value of the TS-sourced AT. The method used to Wire-AND the TDO data require one bit of the AT data to be conveyed to the DTS over more than one TCK(C) period using a precharge/discharge drive scheme.

#### **20.9.1.2 External AT generation with the JScan3 Scan Format**

The precharge/discharge operation of the TDOC signal described in 9.7.3.1.5 is expanded in this subclause. With the JScan3 Scan Format, each bit of the AT information for Undirected CID Allocation is transferred with four *Shift-DR* TAP controller states called an “AT bit-frame.” These states are utilized in a manner that creates the Precharge/Discharge Drive Scheme needed for the Wire-AND of the TDOC data. The TDOC-drive scheme is described hereafter and is shown in Figure 20-3. The four *Shift-DR* states’ transfer of one AT bit of information occurs as follows:

- *Shift-DR* (4n + 0) TDOC is at a high-impedance level.
- *Shift-DR* (4n + 1) TDOC driven to a logic 1 precharging the TDOC signal.
- *Shift-DR* (4n + 2) TDOC is at a high-impedance level.
- *Shift-DR* (4n + 3) TDOC is driven to a logic 0, provided the TAP.7 Controller is a participating candidate, and the AT data value to be output is a logic 0 and remain at a high-impedance level otherwise.

The TDOC signal value is the Wire-AND of the AT bit[n] of all participating candidates from the falling edge of the TCK during the *Shift-DR* TAP controller state (4n + 3) to the subsequent TCK falling edge. Care should be taken in the ADTAPC design to ensure the value to be driven during *Shift-DR* TAP controller state (4n + 1) and during *Shift-DR* TAP controller state (4n + 3) are valid preceding and following the output enable allowing the TDO drive during these bit periods. Failure to do so may cause momentary drive conflicts or the creation of an erroneous data value for these bit periods.

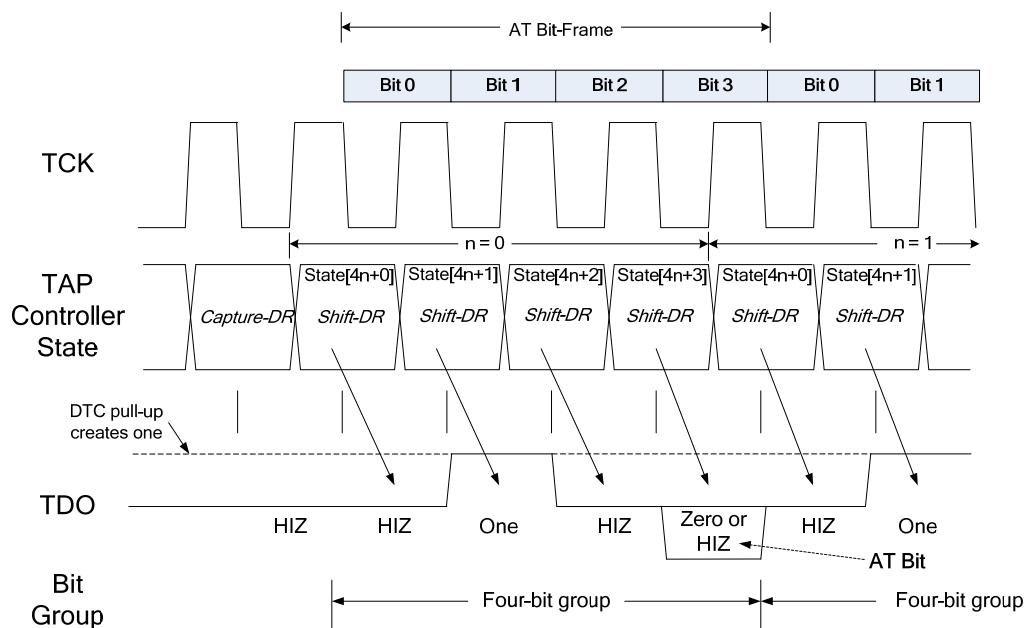
**The DTS will pull-up the TDOC signal (provide a PU bias).** This ensures the precharged logic 1 value created in *Shift-DR* state 4n + 1 is maintained during *Shift-DR* state, as follows:

- $4n + 2$  as there are no TAP.7 Controllers driving the TDOC signal during this state.
- $4n + 3$  when the TDOC pin is not driven by any TAP.7 Controller as a result of either the:
  - AT data of all participating candidates is a logic 1.
  - There are no participating candidates for CID allocation.

The AT bit number is represented by n. The *Shift-DR* TAP controller states may be separated with the *Exit1-DR*, *Pause-DR*, and *Exit2-DR* TAP controller states. This approach produces the following characteristics:

- Prevents drive conflicts as:
  - All TAP.7 Controllers drive a logic 1 in the bit period ( $4n + 1$ )
  - A participating candidate may drive only a logic 0 in bit period ( $4n + 3$ )
  - There can be no drive overlap because of the high-impedance bit periods ( $4n + 0$ ,  $4n + 2$ ).
- Produces a:
  - Logic 0 TDOC signal value in the AT data bit period ( $4n + 3$ ), when a candidate drives this signal
  - A logic 1 TDOC signal value in the AT data bit period ( $4n + 3$ ), when no candidates drives the TDOC signal

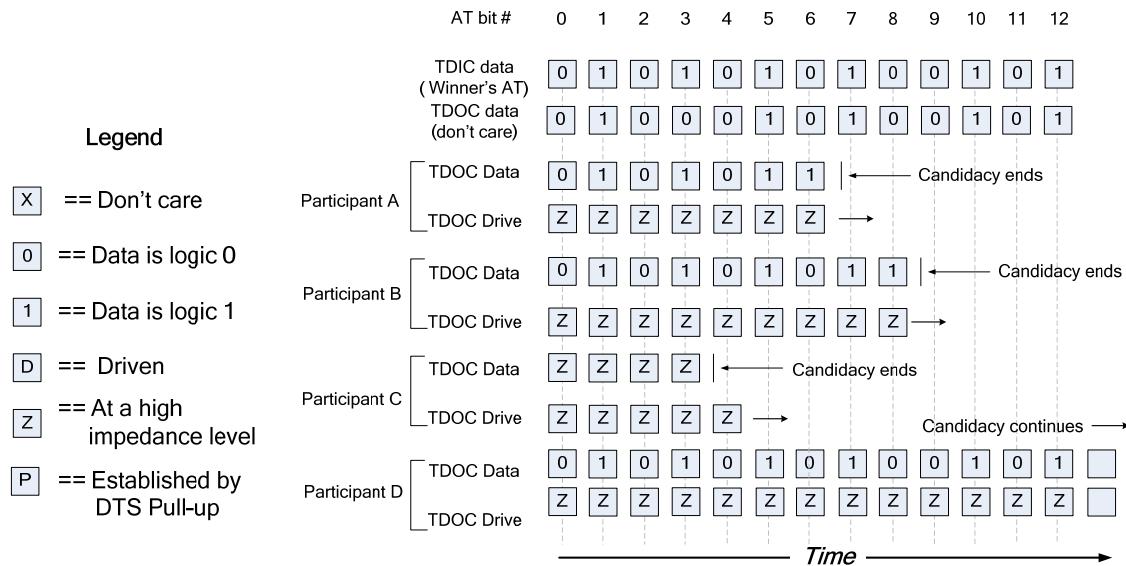
The TDOC drive characteristics described above are illustrated in Figure 20-3.



**Figure 20-3 — TDOC output with the JScan3 Scan Format within the CR Scan**

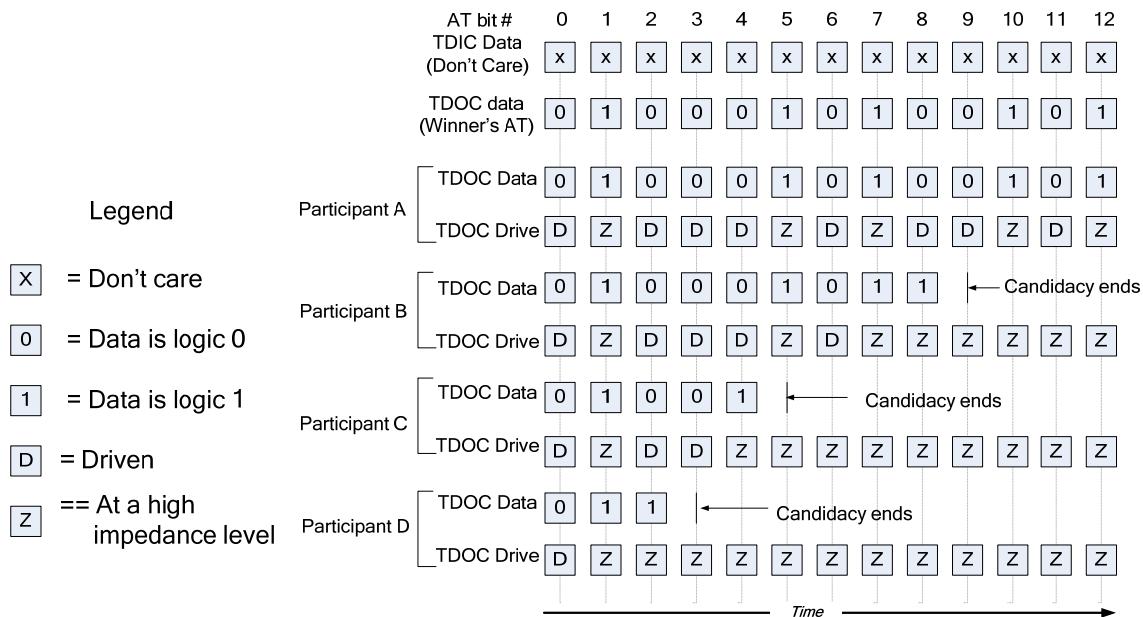
### 20.9.1.3 CID-allocation examples

An example of CID allocation using the directed method with four participating candidates is shown in Figure 20-4. Note that the TAP.7 Controller candidacy for CID allocation ends when bit[n] of the AT does not equal bit[n] of the TCA. These examples refer to AT bit numbers.



**Figure 20-4 — CID allocation with a DTS-sourced AT**

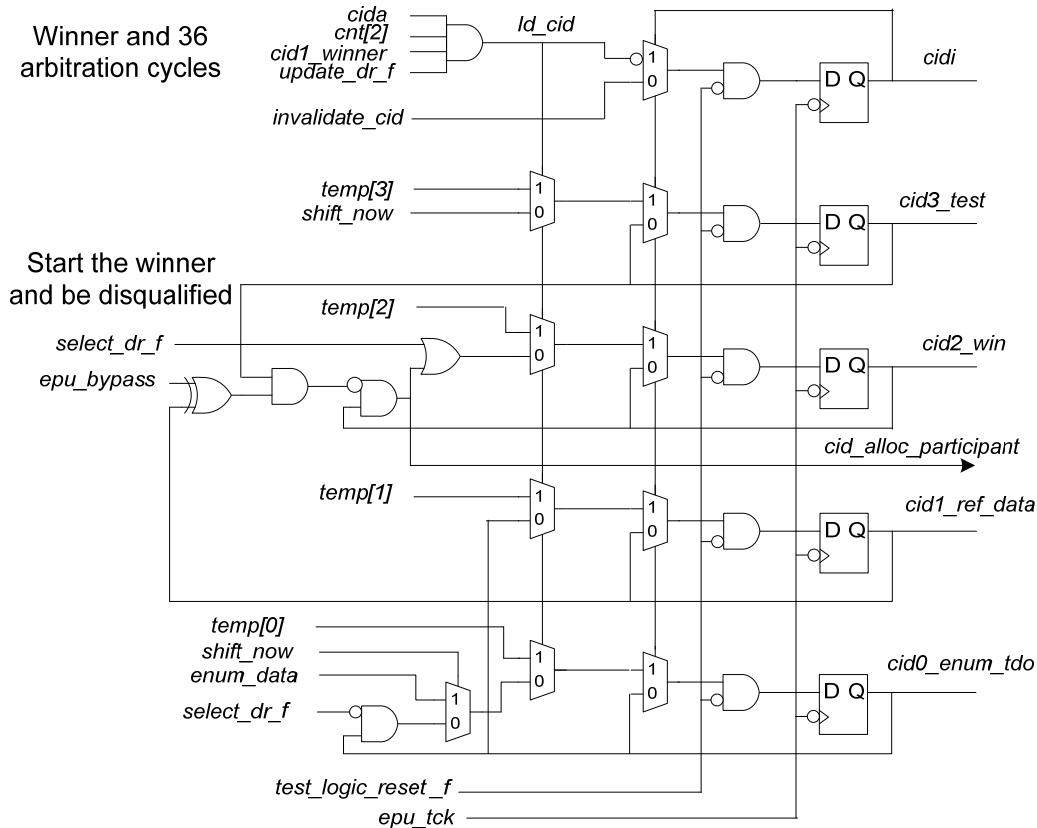
An example of CID allocation using the undirected method with four participating candidates is shown in Figure 20-5. Note that the TAP.7 Controller candidacy for CID allocation ends when bit[n] of the externally supplied AT does not equal bit[n] of the internal AT.



**Figure 20-5 — CID allocation with a TS-sourced AT**

#### 20.9.1.4 An approach to implementing CID allocation

A conceptual view of the CID and CIDI Registers is shown in Figure 20-6. This conceptual view uses the CID Register for two purposes. The first is to store the CID. The second is to perform the arbitration process when the CIDI Register indicates the CID is invalid.



**Figure 20-6 — Conceptual view of CID and CIDI Registers**

## 20.9.2 Specifications

### Rules

- Each subsequent specification in 20.9.2 shall only apply to a T3 and above TAP.7 unless stated otherwise.
- The methods of CID allocation supported by the T3 TAP.7 Scan Formats shall be governed by Table 20-6.
- A TAP.7 Controller shall participate in the CID-assignment process beginning with the *Capture-DR* state when all the following are true:
  - The method of CIDA allocation (directed or undirected) is supported by the scan format.
  - The CIDI Register value is a logic 1.
  - The TCA value is valid before the *Capture-DR* state of the CR Scan portion of the CIDA Command.

**Table 20-6 — CID allocation support with a T3 TAP.7**

Scan Format	T3 TAP.7	
	Directed	Undirected
JScan0–JScan2	No	No
JScan3	Yes	Yes

- d) Subsequent to the *Capture-DR* state identified in Rule 20.9.2 c), a TAP.7 Controller's participation in the CID-allocation process shall be governed by Table 20-7.

**Table 20-7 — Participation of a TAP.7 Controller in CID allocation**

Current state					Next state
AT Bit[n]	TAP Controller state	Participant	External AT[n]	Internal AT[n]	Participant
Initialization	<i>Capture-DR</i>	x	x	x	Initial value established by Rule 20.9.2 c)
0	<i>Shift-DR</i>	No	x	x	No
		Yes	Logic 1	x	No
		Yes	Logic 0	X	Yes
1 ≤ n ≤ 35	<i>Shift-DR</i>	No	x	x	No
		Yes	Logic 0	Logic 0	Yes
		Yes	Logic 0	Logic 1	No
		Yes	Logic 1	Logic 0	No
		Yes	Logic 1	Logic 1	Yes
n > 35	<i>Shift-DR</i>	x	x	x	No

- e) With a T3 TAP.7, the externally supplied AT shall be derived from the TDI/TDO data associated with *Shift-DR* states of the CIDA Command's CR Scan as shown in Table 20-8.

**Table 20-8 — External-AT derivation during CR Scan *Shift-DR* states**

AT Bit[n]	Undirected Allocation	Directed Allocation
0 ≤ n ≤ 35	TDO signal state[4n+ 03]	TDI signal state[n]
Precise number of <i>Shift-DR</i> states within CR Scan to permit CID allocation	144	36

- f) The CID presented by a CIDA Command shall be allocated to a TAP.7 Controller, provided all the following are true:
- 1) The CR Scan included at least one *Shift-DR* state.
  - 2) AT[35] was associated with the last *Shift-DR* state of the CR Scan per Table 20-8.
  - 3) The TAP.7 Controller is a participant in the CID-allocation process when the *Update-DR* TAP controller state of the CR Scan of a CIDA Command is reached.

- g) The TDOC signal shall only be driven during the CR Scan of a CIDA Command when all the following are true:
  - 1) The scan format is JScan3.
  - 2) The undirected allocation method is specified.
  - 3) The TAP controller state is *Shift-DR*.
  - 4) Drive is specified in Table 20-9.
- h) A TAP.7 Controller shall ignore the TDI data of the CIDA CR Scan during Undirected CID Allocation.
- i) A TAP.7 Controller shall ignore the TDO data of the CIDA CR Scan during Directed CID Allocation.
- j) The allocation of a CID shall not be affected by any *Exit1-DR*, *Pause-DR*, and *Exit2-DR* TAP controller states within the CIDA CR Scan.

**Table 20-9 — TDOC behavior/undirected allocation method and JScan3 Scan Format**

Current state					
AT bit number	TAP Controller state	Participation	AT[n] value	epu_tdo	TDOC
	For JScan3				
0	Shift-DR[4n + 0]	x	x	1	HI-Z
	Shift-DR [4n + 1]	x	x	1	Logic 1
	Shift-DR [4n + 2]	No	x	1	HI-Z
		Yes		0	HI-Z
	Shift-DR [4n + 3]	No	x	1	HI-Z
		Yes		0	Logic 0
$n \geq 1$ and $n \leq 35$	Shift-DR [4n + 0]	x	x	1	HI-Z
	Shift-DR [4n + 1]	x	x	1	Logic 1
	Shift-DR [4n + 2]	No	x	1	HI-Z
		Yes	Logic 0	0	HI-Z
	Shift-DR [4n + 3]	Yes	Logic 1	1	HI-Z
		No	x	1	HI-Z
		Yes	Logic 0	0	Logic 0
	Yes	Logic 1	x	1	HI-Z
$n \geq 36$	Shift-DR [4n + 0]	x	x	1	HI-Z
	Shift-DR [4n + 1]	x	x	1	Logic 1
	Shift-DR [4n + 2]	x	x	1	HI-Z
	Shift-DR [4n + 3]	x	x	1	HI-Z

## 20.10 Scan Selection Directives

### 20.10.1 Description

#### 20.10.1.1 Overview

Scan Selection Directives add the following CLTAPC selection/deselection capabilities in T3 and above TAP.7s:

- Selecting and deselecting the CLTAPC in the *Pause-DR* and *Pause-IR* states
- Inhibiting a change in Idle Group Membership in the *Run-Test/Idle* state when the TAP.7 Controller determines that there may be a Pause-xR Group candidate in a TAP.7 Controller in the technology branch associated with this TAP.7 Controller

There are four Scan Selection Directives, as follows:

- SSD\_DA      Deselect the CLTAPC
- SSD\_SOC     Select One TAP.7 Controller using its CID
- SSD\_SOT     Select One TAP.7 Controller using its TCA
- SSD\_SA      Select the CLTAPC

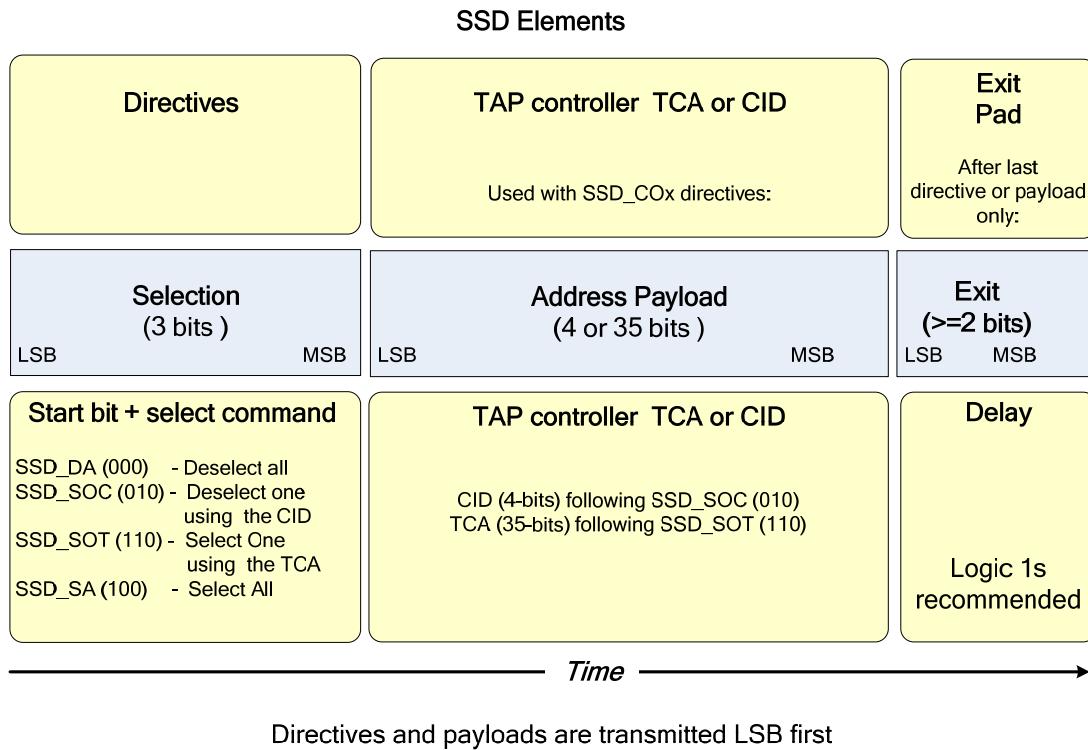
They are created with the TDI data in the *Run-Test/Idle*, *Pause-IR*, and *Pause-DR* TAP controller states.

### 20.10.1.2 SSD format

An SSD is constructed from a combination of the three elements listed as follows:

- A directive
- An address payload
- An exit

This is shown in Figure 20-7. The encoding and format of SSDs is also described in Table 20-10.



**Figure 20-7 — Conceptual view of an SSD**

Note that the LSB of all Scan Selection Directives is a logic 0. Because these directives are transmitted LSB first, the LSB of each directive functions as a start bit. An Exit Element (a period with no new directives) follows the last SSD before exiting the TAP.7 Controller state supporting the use of SSDs. The Exit Element provides time for processing the last SSD preceding the Exit Element before the supporting TAPC state is exited.

### 20.10.1.3 SSD effects on STL Group Membership

The function of SSDs is ADTAPC State Machine state dependent as described in 13.8.3.2. An SSD executed with the ADTAPC *Run-Test/Idle* state may change the value of the SGC Register value. When this occurs, the SGC Register value immediately affects Idle Group Membership. An SSD executed with the ADTAPC *Pause-xR* state may change the STL's Group Membership (provided the STL is not an Idle Group Member).

With the *Run-Test/Idle* state, an executed SSD has the function shown in Table 20-10. The SSD\_SOC SSD performs no operation when the SSD-provided CID does not match the TAP.7 Controller's CID. Likewise,

the SSD\_SOT SSD performs no operation when the SSD-provided TCA does not match the TAP.7 Controller's TCA.

**Table 20-10 — Executed SSD/Run-Test/Idle/group membership relationships**

SSD	TCA match?	CID match?	Resulting group membership	Resulting SGC Register value
SSD_DA	N/A	N/A	Idle	Logic 0
SSD_SA	N/A	N/A	Scan	Logic 1
SSD_SOC	N/A	No	No change	No change
	N/A	Yes	Scan	Logic 1
SSD_SOT	No	N/A	No change	No change
	Yes	N/A	Scan	Logic 1

With the *Pause-xR* states, executed SSDs have the function shown in Table 20-11.

**Table 20-11 — Executed SSD/Pause-xR/group membership relationships**

SSD	TCA match?	CID match?	Current group membership is idle?	Resulting group membership
SSD_DA	N/A	N/A	Yes	Idle
	N/A	N/A	No	Pause-xR
SSD_SA	N/A	N/A	Yes	Idle
	N/A	N/A	No	Scan
SSD_SOC	N/A	x	Yes	Idle
	N/A	No	No	Pause-xR
	N/A	Yes	No	Scan
SSD_SOT	x	N/A	Yes	Idle
	No	N/A	No	Pause-xR
	Yes	N/A	No	Scan

#### 20.10.1.4 Enabling SSD processing

SSD processing is enabled, provided the criteria listed as follows are met:

- An SSDE Register value of logic 1 enables SSD processing.
- The System Path is being used.
- The TAPC state is one the following:
  - *Run-Test/Idle*.
  - *Pause-IR*.
  - *Pause-DR*.
- The TMS value for the state is a logic 0.

When these criteria are not met, SSD processing is inhibited and in some cases is aborted, if ongoing.

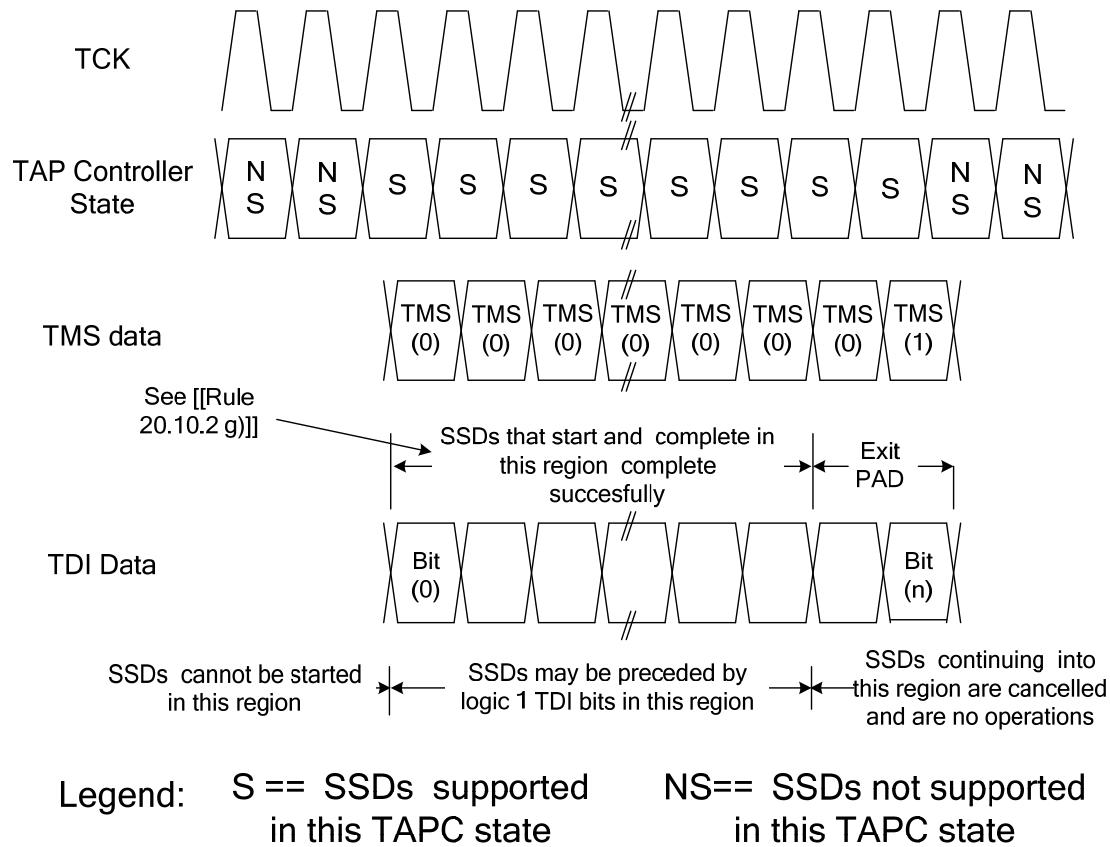
#### 20.10.1.5 SSD processing

From zero to n, logic 1 TDI values may precede an SSD because the LSB of all SSDs is a logic 0. SSD processing begins with the first a logic 0 TDI bit associated with a supporting state. Once an SSD processing is initiated with its LSB, the two subsequent TDI values determine the SSD type. With an Select All (SSD\_SA) or Deselect All (SSD\_DA) SSD, the SSD completes with the third SSD bit, with the SSD

start-bit detection resuming afterward. With a Select One (SSD\_SOC or SSD\_SOT) SSD, the SSD processing continues past the third bit. With an SSD\_SOC SSD, the processing logic compares the CID conveyed by the SSD payload conveying to the TAP.7 Controller's CID. The action created by the SSD is based on the comparison of these CIDs and the state as described above. With an SSD\_SOT SSD, the processing logic compares the TCA conveyed by the payload to the TAP.7 Controller's TCA. The action created by the SSD is based on the comparison of these TCAs. In both of these cases, the detection of an SSD start-bit resumes after the last of the TCA payload.

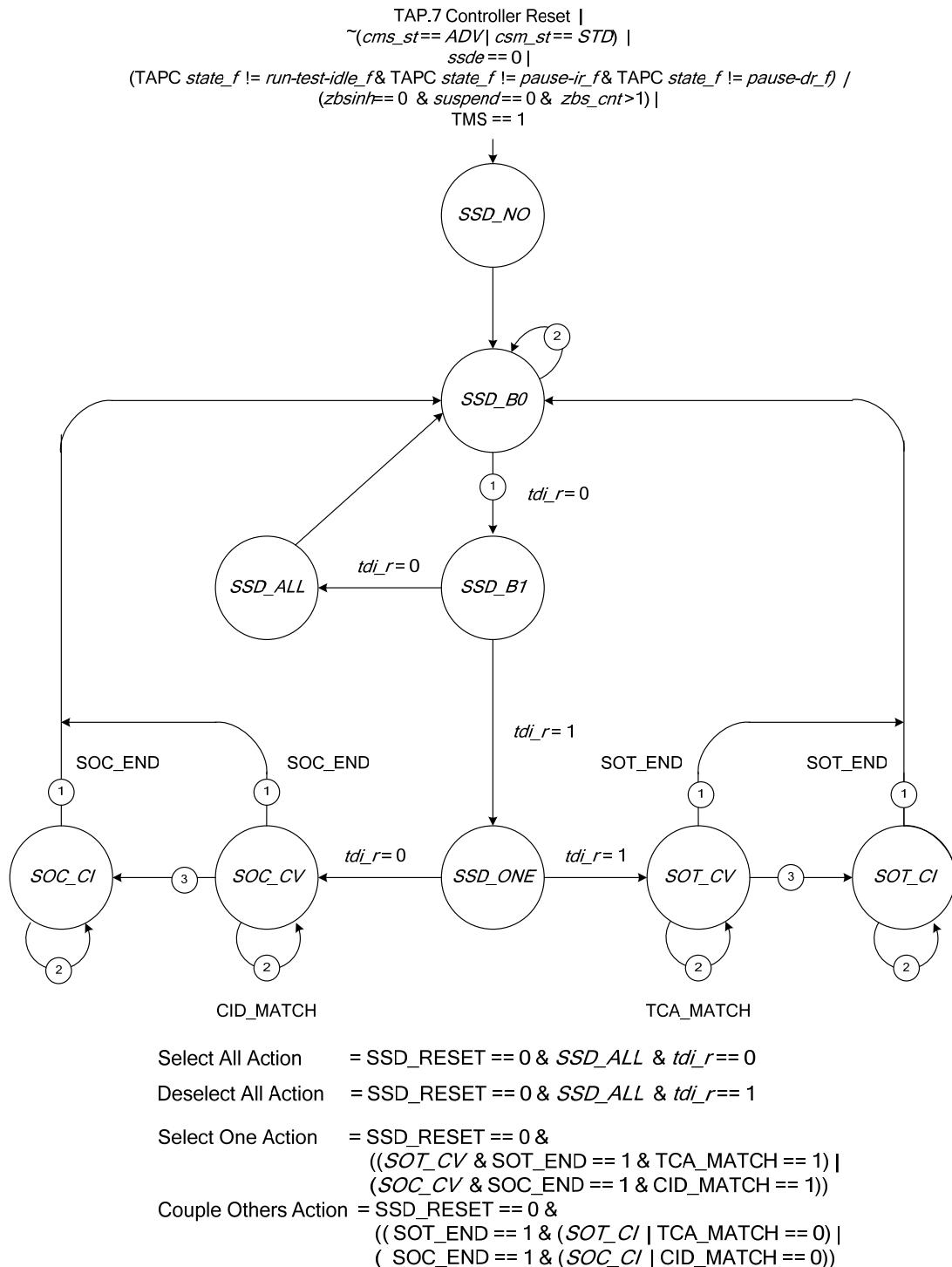
An SSD completes successfully and performs its intended function, provided the TMS value coincident with any bit period of an SSD and the bit period following the last bit period of the SSD is a logic 0. This applies to all SSDs. SSD processing is aborted when a logic 1 TMS value occurs before the completion of this processing or aligned with the TDI bit position following the last bit of the SSD, with the partially processed SSD having no effect. The latter case is included to prevent an SSD action affecting group membership after the supporting state is exited, as this could cause a change in the CLTAPC Selection State when the CLTAPC state is a state other than *Run-Test/Idle*, *Pause-IR*, and *Pause-DR*.

SSDs that start and end in the completion region shown in Figure 20-8 and are not followed by a logic 1 TMS bit in the subsequent EPU TCK period perform their specified function and perform no operation otherwise.



**Figure 20-8 — SSDs detection and completion regions**

A conceptual view of SSD processing is shown with the state machine in Figure 20-9. The state names used with this state machine are described in Table 20-12.



**Figure 20-9 — Conceptual view of SSD Processing State Machine**

Note that a Control State Machine state other than *ADV* or *STD* aborts SSD processing while having no effect on the SSD state and Delayed SSD State.

**Table 20-12 — SSD Processing State Machine state names**

<b>State name</b>	<b>TAP controller state</b>
<i>SSD_NO</i>	No SSD detection
<i>SSD_B0</i>	SSD Bit 0
<i>SSD_B1</i>	SSD Bit 1
<i>SSD_ALL</i>	Select or Deselect All SSD
<i>SSD_ONE</i>	SSD with address payload
<i>SSD_SOC_CV</i>	SSD_SOC CID compare valid
<i>SSD_SOC_CI</i>	SSD_SOC CID compare invalid
<i>SSD_SOT_CV</i>	SSD_SOT TCA compare valid
<i>SSD_SOT_CI</i>	SSD_SOT TCA compare invalid

#### **20.10.1.6 Conditional SSD execution**

The execution of SSDs is conditional. The action created by an SSD's execution (normal or no-operation) is determined by the ADTAPC state enabling SSD detection in combination with the number of Pause-IR and Pause-DR Group Members at the time the SSD executes. This creates a single-threaded process changing group membership. The execution of an SSD associated with the TAPC states shown as follows performs its specified function when the following conditions are met and performs a no-operation otherwise:

- *Run-Test/Idle* There appears to be no Pause-IR or Pause-DR Group Members.
- *Pause-IR* There appear to be no Pause-DR Group Members.
- *Pause-DR* There appear to be no Pause-IR Group Members.

Conditional SSD execution simplifies TDOC drive-conflict prevention and makes the determination of group membership orderly. The conditions inhibiting the execution of SSDs also inhibit the initiation of a Selection Sequence with a Selection Escape or Selection Alert and the deselection actions that are normally created with a Deselection Escape or Deselection Alert.

#### **20.10.1.7 SSD interaction with other controller functions**

The ZBS count is zeroed by an executed SSD, provided the ZBS count is not locked (independently of the value of the ZBSINH or SUSPEND Register values). With this the case, a control level greater than one cannot be established with the continued use of one or more SSDs between subsequent *Update-DR* states. This means commands cannot be interleaved with the continued use of SSDs described above.

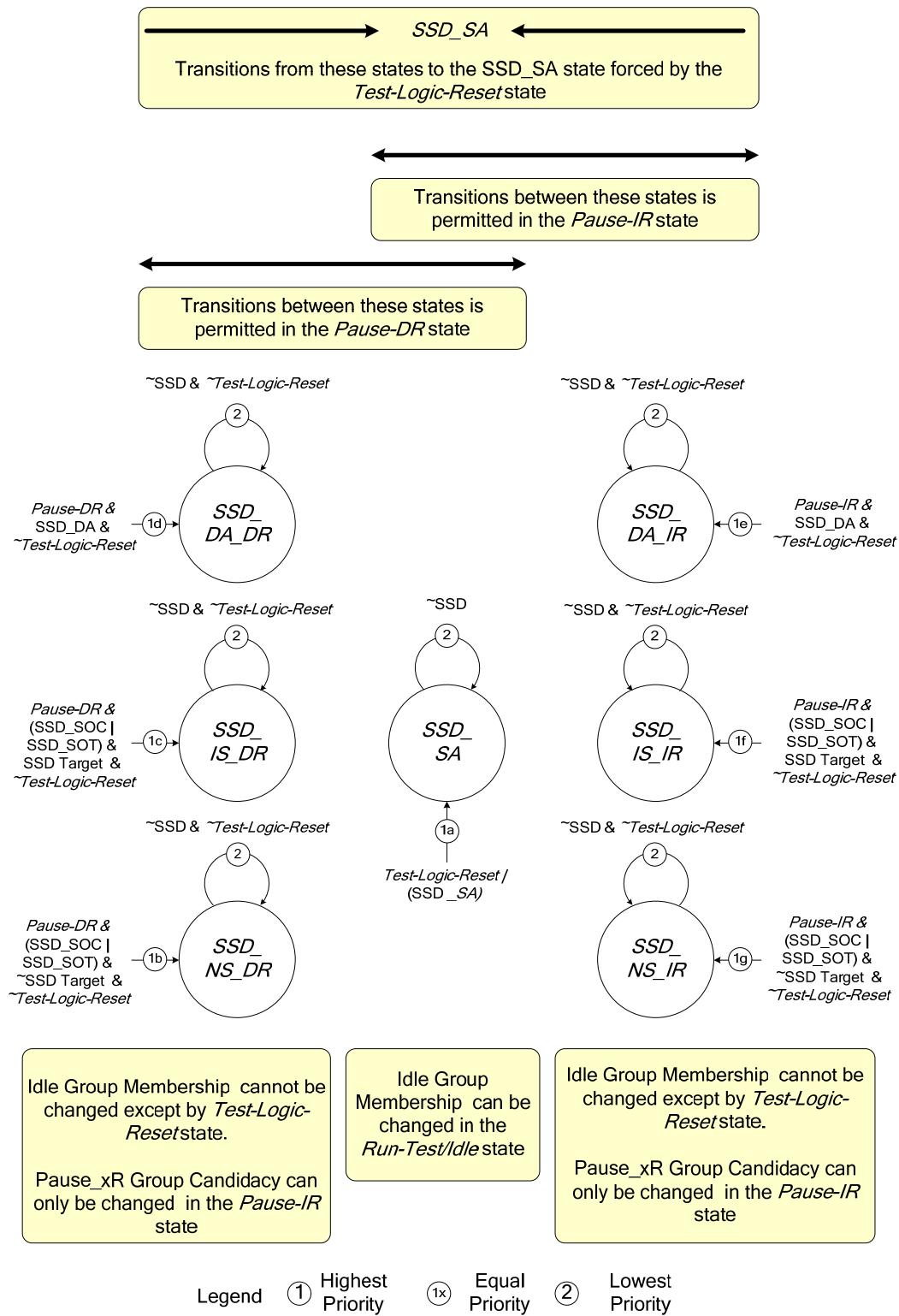
In addition, the following operations are inhibited when the SSD state identifies there are either Pause-IR or Pause-DR Group Members:

- The initiation of a Selection Sequence normally initiated by a Selection Escape or Selection Alert
- The deselection action normally initiated by a Deselection Escape

#### **20.10.1.8 SSD State Machine**

The last executed SSD and the TAPC State Machine state in which it was processed is recorded with the SSD State Machine shown in Figure 20-10. This state information identifies whether there are any Pause-xR Group Members and whether this CLTAPC is a Pause-xR Group Member. This state is also delayed

one EPU TCK period to produce the Delayed SSD State that is used to form both the TDOC and TMSC Drive Policies for the System Path. The SSD state names and their description are shown in Table 20-13.



**Figure 20-10 — SSD Interlocking State Machine**

**Table 20-13 — SSD state description**

SSD state	Description
<i>SSD_SA</i>	<ul style="list-style-type: none"> <li>— The STLs are only Idle and Scan Group Members.</li> <li>— There are no Pause-xR Group Members.</li> <li>— One of the following occurred:           <ol style="list-style-type: none"> <li>1) The <i>Test-Logic-Reset</i> state</li> <li>2) An <i>SSD_SA</i> SSD was executed</li> </ol> </li> </ul>
<i>SSD_DA_IR</i>	<ul style="list-style-type: none"> <li>— The STLs are only Idle and Pause-IR Group Members.</li> <li>— This CLTAPC is deselected and the STL is a Pause-IR Group Member.</li> <li>— The last SSD was an <i>SSD_DA</i> processed in the <i>Pause-IR</i> state.</li> </ul>
<i>SSD_DA_DR</i>	<ul style="list-style-type: none"> <li>— The STLs are only Idle and Pause-DR Group Members.</li> <li>— This CLTAPC is deselected and the STL is a Pause-DR Group Member.</li> <li>— The last SSD was an <i>SSD_DA</i> processed in the <i>Pause-DR</i> state.</li> </ul>
<i>SSD_NS_IR</i>	<ul style="list-style-type: none"> <li>— This STL is a member of either an Idle or a Pause-IR Group Member.</li> <li>— There may be a maximum of one STL that is a Scan Group Member.</li> <li>— The last SSD was:           <ol style="list-style-type: none"> <li>1) Associated with the <i>Pause-IR</i> state.</li> <li>2) One of the following:               <ol style="list-style-type: none"> <li>a) An <i>SSD_SOC</i> that did not identify this STL as the target of the SSD.</li> <li>b) An <i>SSD_SOT</i> that did not identify this STL as the target of the SSD.</li> </ol> </li> </ol> </li> </ul>
<i>SSD_NS_DR</i>	<ul style="list-style-type: none"> <li>— This STL is a member of either an Idle or Pause-DR Group Member.</li> <li>— There may be a maximum of one STL that is a Scan Group Member.</li> <li>— The last SSD was:           <ol style="list-style-type: none"> <li>1) Assocaited with the <i>Pause-DR</i> state</li> <li>2) One of the following:               <ol style="list-style-type: none"> <li>a) An <i>SSD_SOC</i> that did not identify this STL as the target of the SSD.</li> <li>b) An <i>SSD_SOT</i> that did not identify this STL as the target of the SSD.</li> </ol> </li> </ol> </li> </ul>
<i>SSD_IS_IR</i>	<ul style="list-style-type: none"> <li>— This STL is either an Idle or the Scan Group Member.</li> <li>— All other STLs are either Idle or the Pause-IR Group Members.</li> <li>— The last executed SSD was:           <ol style="list-style-type: none"> <li>1) Associaed with the <i>Pause-IR</i> state.</li> <li>2) One of the following:               <ol style="list-style-type: none"> <li>a) An <i>SSD_SOC</i> that identified this STL as the target of the SSD.</li> <li>b) An <i>SSD_SOT</i> that identified this STL as the target of the SSD.</li> </ol> </li> </ol> </li> </ul>
<i>SSD_IS_DR</i>	<ul style="list-style-type: none"> <li>— This STL is either an Idle or Scan Group Member.</li> <li>— All other CLTAPCs are either an Idle or Pause-DR Group Member.</li> <li>— The last executed SSD was:           <ol style="list-style-type: none"> <li>1) Associated with the <i>Pause-DR</i> state.</li> <li>2) One of the following:               <ol style="list-style-type: none"> <li>a) An <i>SSD_SOC</i> that identified this STL as the target of the SSD.</li> <li>b) An <i>SSD_SOT</i> that identified this STL as the target of the SSD.</li> </ol> </li> </ol> </li> </ul>

NOTE—The names of the Delayed SSD State are the same with a *\_DLY* suffix.

#### 20.10.1.9 SSD states allowing Scan Group Membership

Scan Group Membership is allowed, provided the SSD state is one of the following:

- *SSD\_SA*
- *SSD\_IS\_IR*
- *SSD\_IS\_DR*

This combination of states indicates one of the following:

- The *SSD\_SA* state indicates there are no Pause-IR or Pause-DR Group Members.
- The *SSD\_IS\_IR* and *SSD\_IS\_DR* states indicate this STL is not a Pause-IR or Pause-DR Group Member.

#### 20.10.1.9.1 Using SSDs to create Series and Star-Equivalent Scans

A Series-Equivalent Scan with a Star Scan Topology is created using the template described below. This template supports the scan of any number of scan sections associated with different CLTAPCs. This scan is created with a Preamble, Sections, and a Postamble of TAP controller states. The Preamble, Sections, and Postamble TAP controller state sequences are listed below:

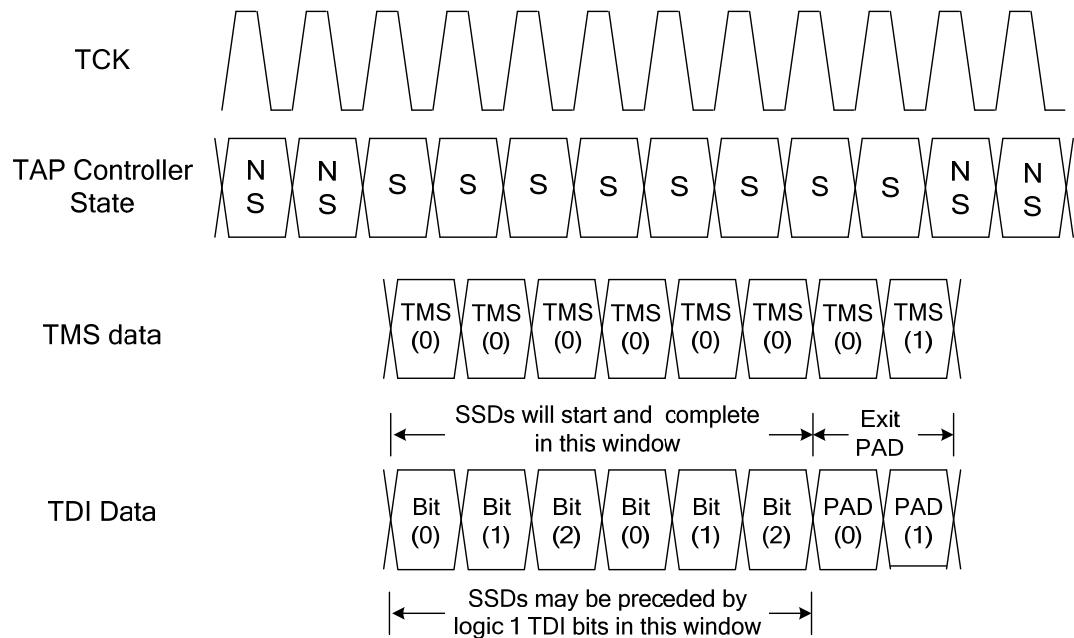
- Preamble              *Capture-xR >> Exit1-xR*
- For each section      *(Pause-xR(N) + SSD\_SOx) >> Exit2-xR >> Shift-xR(N) >> Exit1-xR*
- Postamble              *(Pause-xR(N) + SSD\_SA) >> Exit-xR >> Update-xR*

Multiple *Shift-xR >> Exit1-xR >> Pause-xR* state sequences may be inserted between the *Exit2-xR* and *Shift-xR* TAP controller states in a section, if desired. These inserted state sequences should contain no SSDs in the *Pause-xR* TAP controller states.

An example of SSDs used to create a Series-Equivalent Scan in a Star Scan Topology is shown in Figure 5-23.

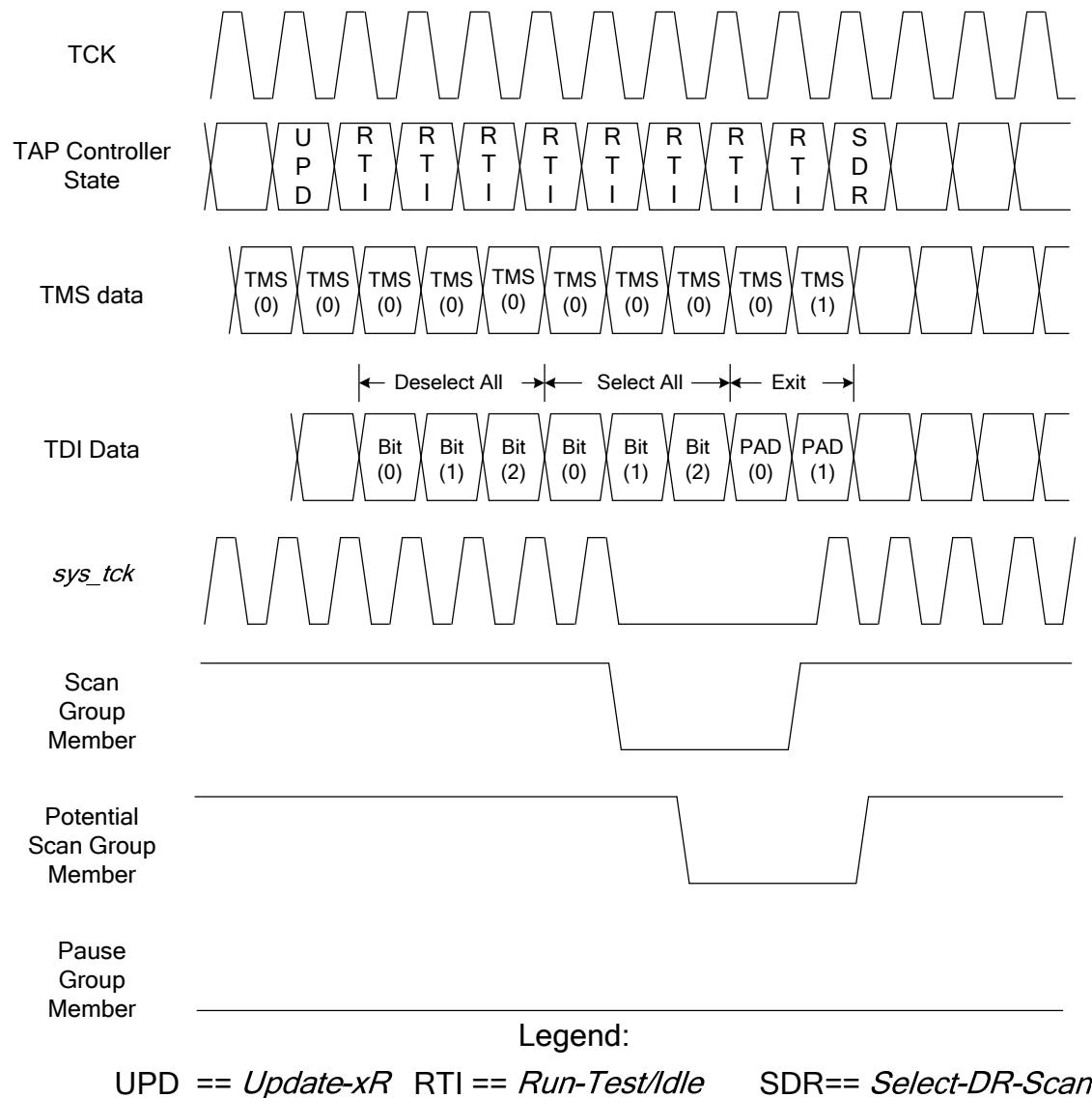
#### 20.10.1.9.2 Examples of SSD use

Some examples of SSD use are shown in Figure 20-11 through Figure 20-15. Note the Exit Element following the last SSD and the adjacent spacing of the *SSD\_DA* and *SSD\_SA* Directives. In Figure 20-12 and Figure 20-14, the SSD executes and changes group membership. In Figure 20-13 and Figure 20-15, the last SSD does not execute, is discarded, and does not change the group membership. It is though this SSD was never presented to the TAP.7 Controller.

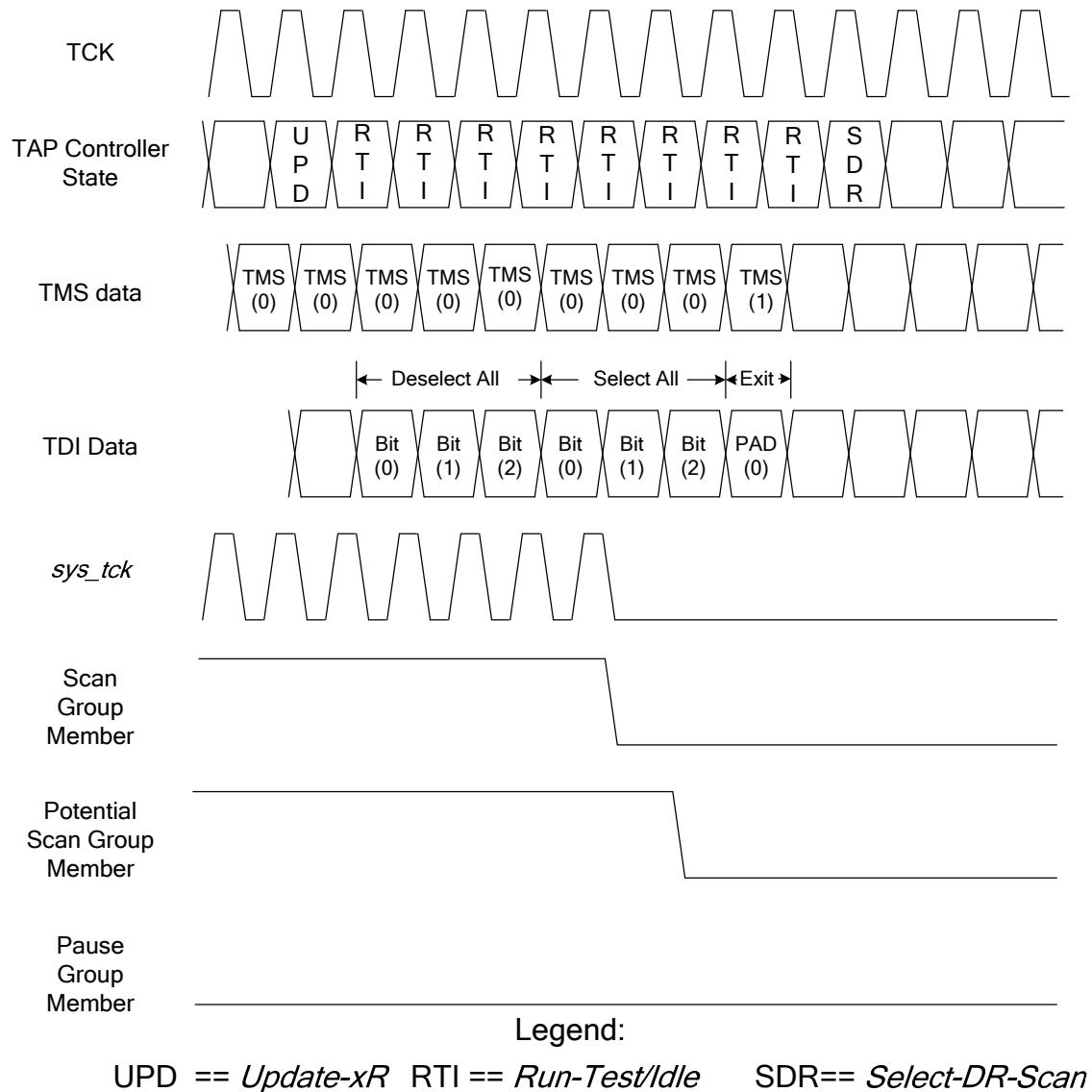


Legend: S== SSDs supported NS== SSDs not supported

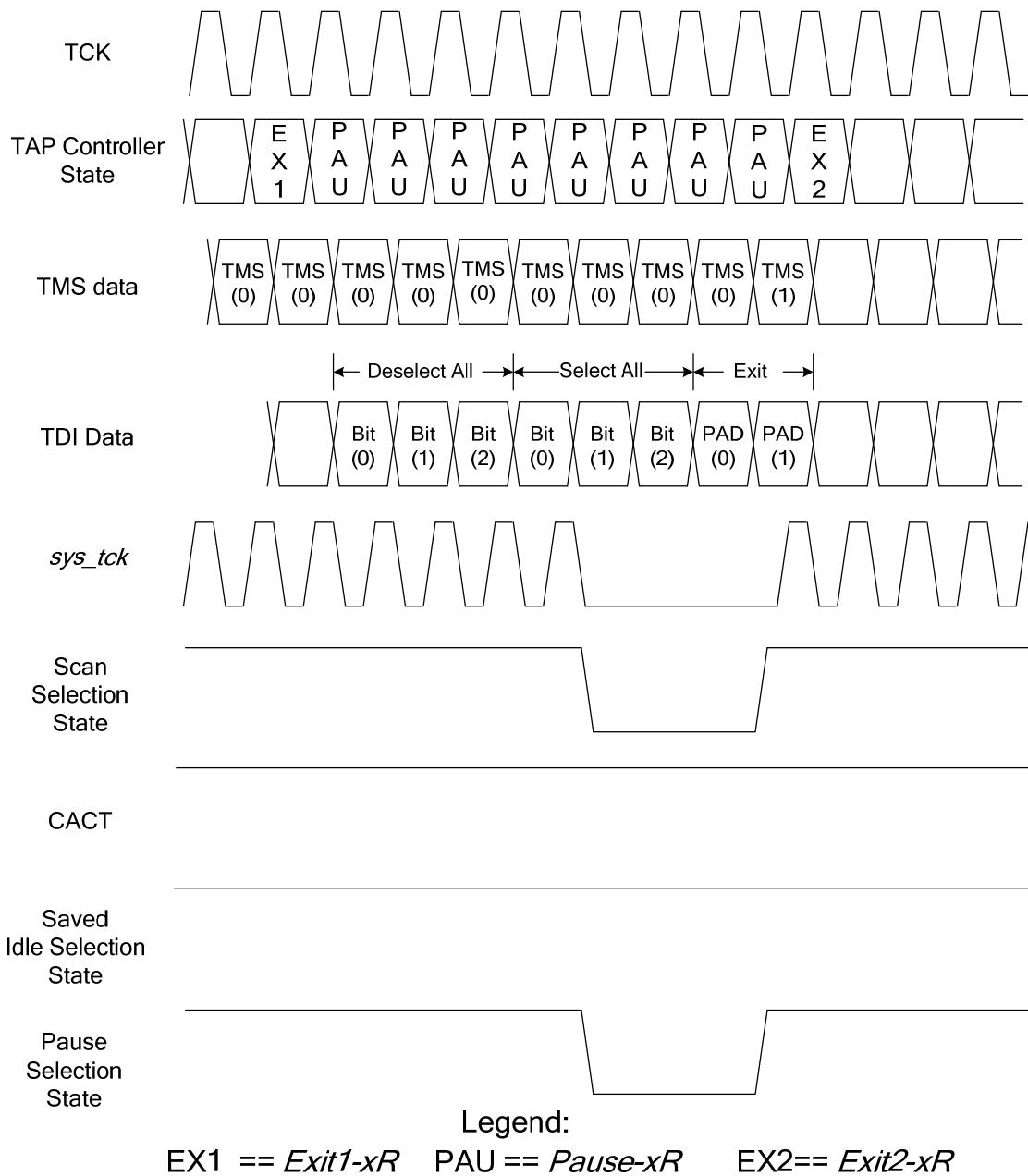
**Figure 20-11 — SSD detection and completion window**



**Figure 20-12 — SSDs changing the Scan Group Membership in the RTI state**



**Figure 20-13 — SSD\_SA ignored because of early exit from the *RTI* state**



**Figure 20-14 — SSDs changing the Pause Selection state**

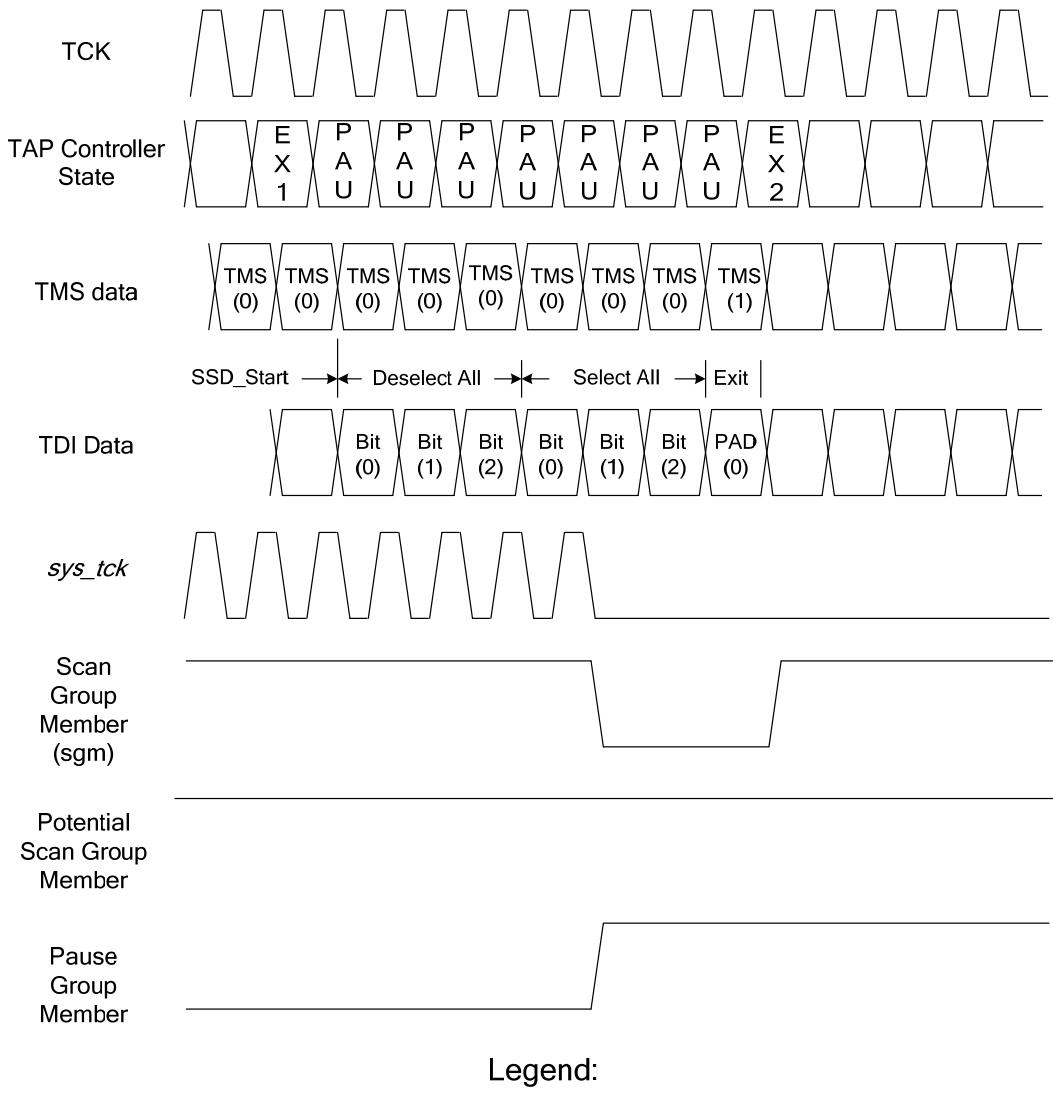
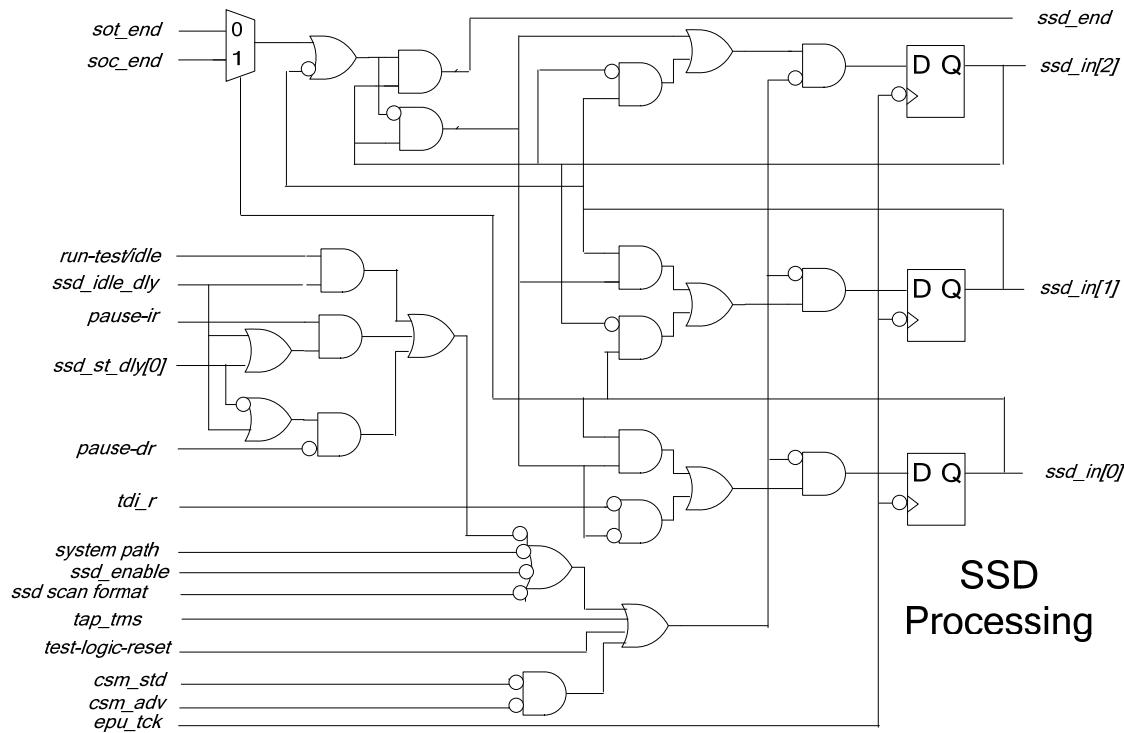


Figure 20-15 — SSD SA ignored in the *Pause-xR* state

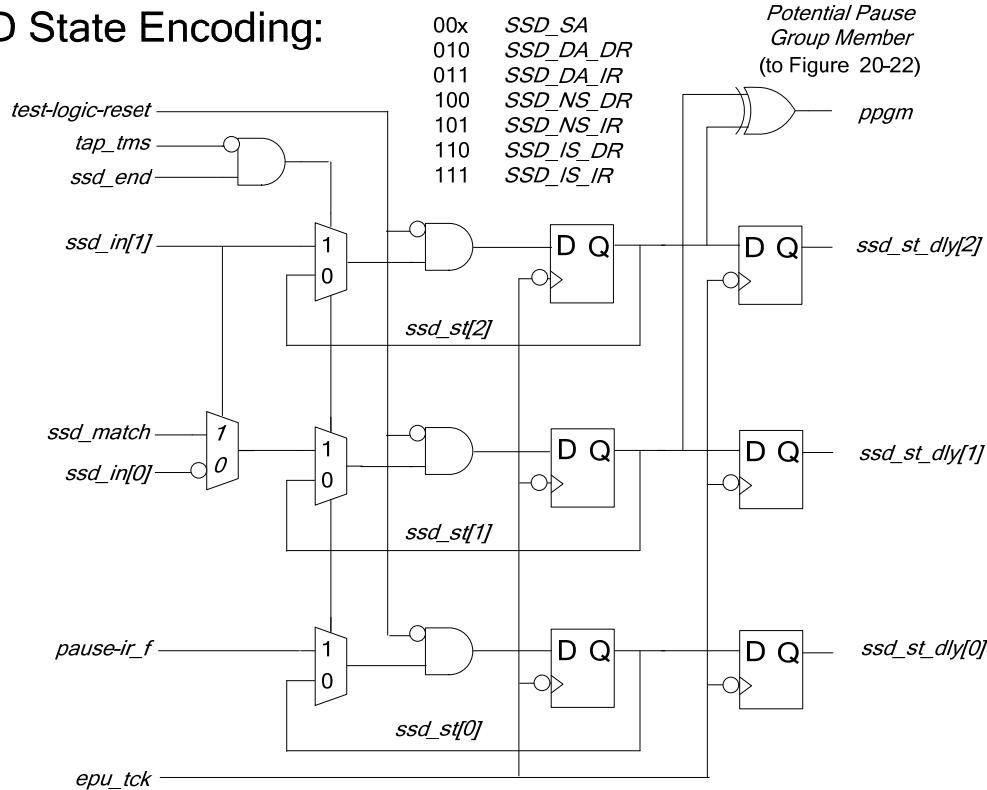
#### 20.10.1.10 An approach to implementing the SSD function

An approach that may be used to perform SSD processing and record the SSD state is shown in Figure 20-16 and Figure 20-17. The CID and TCA comparison is shown in Figure 20-18. The TCA comparison may be shared with the CIDA Command as shown.

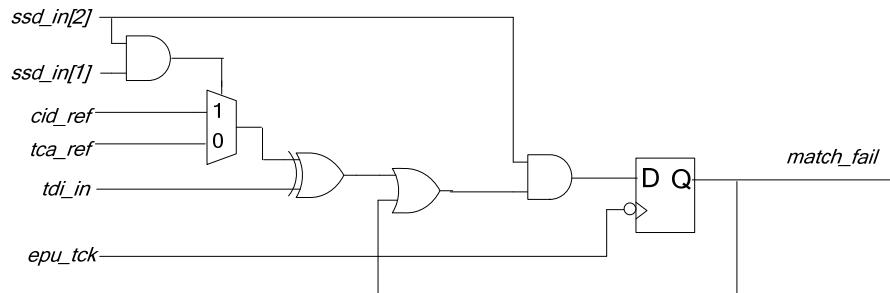


**Figure 20-16 — Conceptual view of SSD processing**

### SSD State Encoding:



**Figure 20-17 — Conceptual view of SSD state and SSD state delayed**



**Figure 20-18 — Conceptual view of TCA and CID comparison**

## 20.10.2 Specifications

### Rules

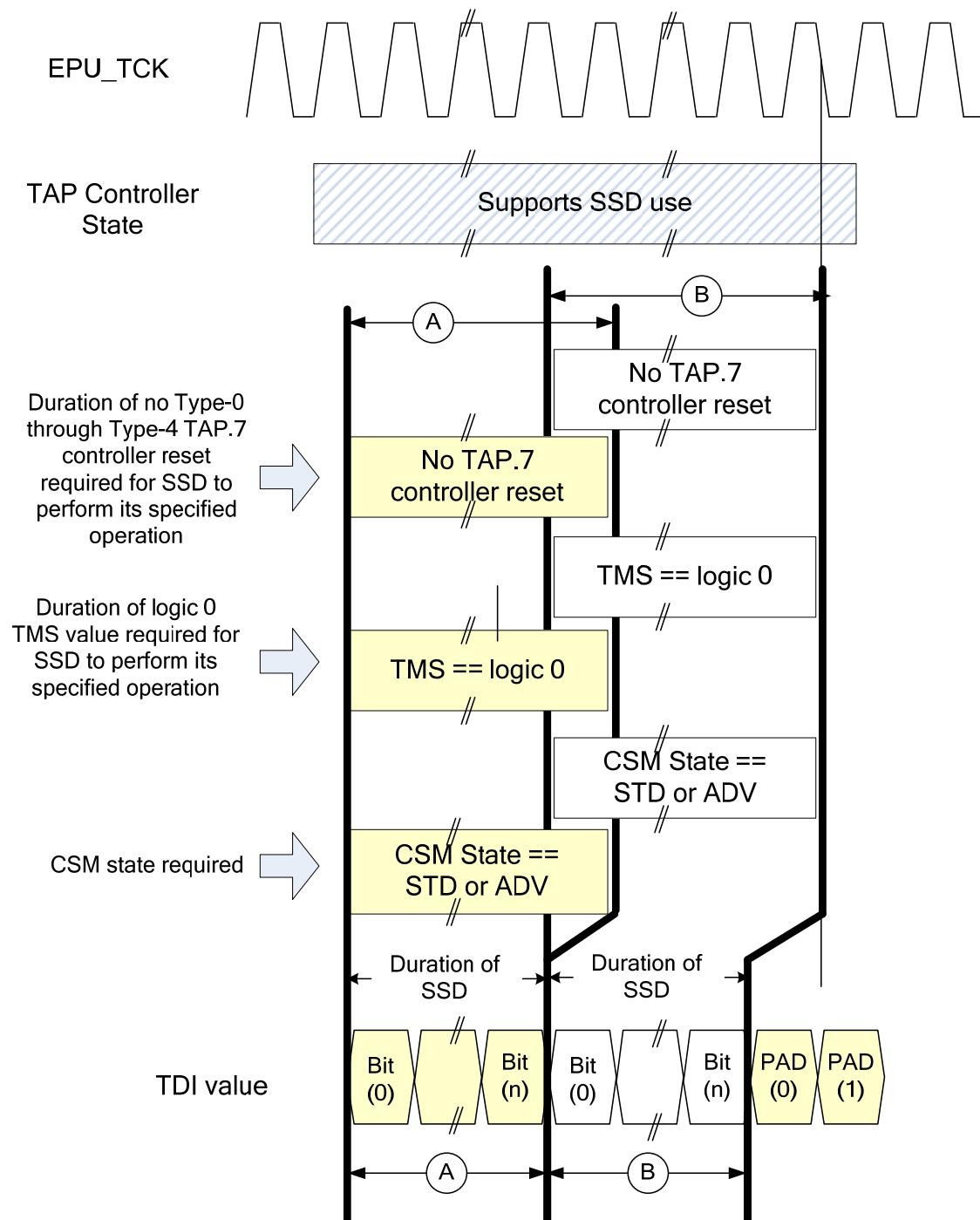
- a) Each subsequent specification in 20.10.2 shall only apply to T3 and above TAP.7s unless otherwise specified.
- b) The format of SSDs and address payloads shall be governed by Table 20-14.
- c) The Selection and Payload Elements shown in Table 20-14 shall be transmitted LSB first with the directive preceding the payload.

**Table 20-14 — SSDs and address payloads**

Selection Element	Mnemonic	Directive value	SSD size in bits	Address payload size in bits	Payload description
Deselect All	SSD_DA	000b	3	0	N/A
Select one	SSD_SOC	010b	7	4	A CID
	SSD_SOT	110b	38	35	A TCA
Select all	SSD_SA	100b	3	0	N/A

- d) Any number (including zero) of logic 1 TDI values preceding the first bit of an SSD shall be permitted.
- e) The processing of the SSDs (see Figure 20-9) shall be initiated, provided all the following seven items are true:
  - 1) The value of the SSDE Register is logic.1.
  - 2) The ADTAPC state is any of the following:
    - i) *Run-Test/Idle*.
    - ii) *Pause-IR*.
    - iii) *Pause-DR*.
  - 3) The TDI and TMS values for the supporting ADTAPC state are both a logic 0.
  - 4) Any of the following:
    - i) SSD processing has not been initiated since entering the supporting state.

- ii) All bits of any preceding SSDs have been received.
- 5) The System Path is being used.
- 6) The scan format supports the detection of SSDs.
- 7) The CSM indicates either the Advanced or Standard Protocol is being used.
- f) All SSD bits shall be conveyed with a TDI bit value associated with a TAPC State Machine state supporting SSDs as shown in Figure 20-19.
- g) Once SSD processing is initiated, it shall be completed with the SSD performing its specified function (completed successfully), provided the TMS value, Type-0 through Type-4 TAP.7 Controller resets, and an SSD have the relationships shown in Figure 20-19.



**Figure 20-19 — TMS and reset value required for SSD to perform its designated function**

- h) Once initiated, SSD processing shall be aborted, provided Rule 20.10.2 g) is not satisfied.
- i) The definition of the states of the SSD State Machine required by Rule 20.10.2 j) shall be governed by Table 20-13.
- j) Successful execution of an SSD as described by Rule 20.10.2 g) shall be recorded with an SSD State Machine.

- k) The Delayed SSD State shall be the SSD State delayed by one EPU TCK period.
- l) The SSD that is executed and performs its specified function as specified by Rule 20.10.2 g) shall create the SSD State Machine state shown in Table 20-15.

**Table 20-15 — SSD/SSD state relationships**

SSD	CID match ?	TCA match ?	SSD state when SSD is associated with:		
			Run-Test/Idle	Pause-IR	Pause-DR
SSD_SA	x	x	SSD_SA	SSD_SA	SSD_SA
SSD_DA	x	x	SSD_SA	SSD_DA_IR	SSD_DA_DR
SSD_SOC	No	x	SSD_SA	SSD_NS_IR	SSD_NS_DR
	Yes	x	SSD_SA	SSD_IS_IR	SSD_IS_DR
SSD_SOT	x	No	SSD_SA	SSD_NS_IR	SSD_NS_DR
	x	Yes	SSD_SA	SSD_IS_IR	SSD_IS_DR

- m) SSD state changes shall be permitted, provided the TAP controller and Delayed SSD State are a combination identified with a Yes in Table 20-16.

**Table 20-16 — Permitted SSD processing**

SSD_STATE_DLY (SSD state delayed)	SSD processing allowed in:		
	Run-Test/Idle	Pause-IR	Pause-DR
SSD_SA_DLY	Yes	Yes	Yes
SSD_DA_DR_DLY	No	No	Yes
SSD_DA_IR_DLY	No	Yes	No
SSD_NS_DR_DLY	No	No	Yes
SSD_NS_IR_DLY	No	Yes	No
SSD_IS_DR_DLY	No	No	Yes
SSD_IS_IR_DLY	No	Yes	No

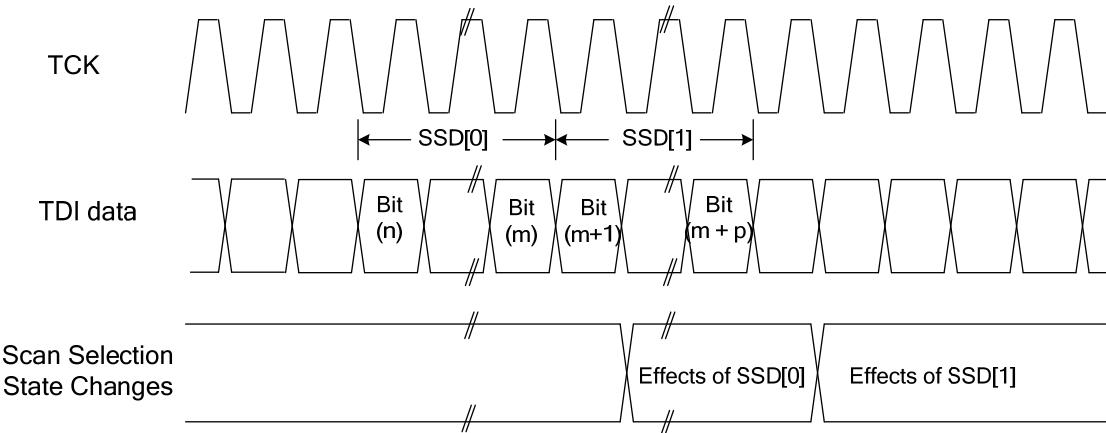
- n) Successful completion of an SSD shall cause the changes to the SGC Register as shown in Table 20-17.

**Table 20-17 — Scan Selection Directive/SGC relationships**

SSD	SSD payload == Controller CID	SSD payload == Controller TCA	Run-Test/Idle state	Pause-xR state
SSD_DA	N/A	N/A	SGC = 0	
SSD_SOC	No	x	No change in SGC value	No change in SGC value
	Yes	x	SGC = 1	
SSD_SOT	x	No	No change in SGC value	No change in SGC value
	x	Yes	SGC = 1	
SSD_SA	—	—	SGC = 1	

- o) The *Test-Logic-Reset* state shall initialize the SSD state to *SSD\_SA*.

- p) The CLTAPC shall be considered a Scan Group Member, provided all the following are true:
  - 1) The CLTAPC is not a Member of the Idle Group.
  - 2) The SSD State is any of the following:
    - i) *SSD\_SA*.
    - ii) *SSD\_IS\_IR*.
    - iii) *SSD\_IS\_DR*.
- q) The effects of an SSD shall take place as shown in Figure 20-20 when the SSD completes successfully as specified by Rule 20.10.2 g).



**Figure 20-20 — SSD effects**

## 20.11 Scan Topology Training Sequence

### 20.11.1 Description

#### 20.11.1.1 Overview

T3 and above TAP.7s have the capability to determine the scan topology in which they are deployed (Series, Star-4, and Star-2) when the DTS stimulates them with the Scan Topology Training Sequence. Because T3 and above TAP.7s may be deployed in more than one type of TAP.7 Technology Branch, this capability provides the information needed to select/deselect the ADTAPC when the branch in which it resides is selected/deselected with the Selection Sequence described in 11.7.

Because T0–T2 TAP.7s can only be deployed in a Series Scan Topology, it is hard-wired to know it is deployed in a Series Scan Topology. The Scan Topology Training Sequence performs no function with these T0–T2 TAP.7s.

#### 20.11.1.2 Topology Register Function

With a T3 and above TAP.7, the Scan Topology Training Sequence stores the TAP.7 Scan Topology determined by this operation in the Topology (TOPOL) Register. With a T3 and above TAP.7, the value of this register determines the TAP.7 Controller's response to the Technology Selection portion of a Selection Sequence.

The Topology Register value affects the following two operating characteristics:

- Creates the Dormant TDO Drive Policy when a 00b value is created by storing the register or at start-up when the start-up option is Offline-at-Start-up
- Determines the TAP.7 Controller's response to the Technology Selection portion of a Selection Sequence

A Type-0-Type-3 Reset initializes the Topology Register value to specify a Series Topology when the TAP.7 Controller is placed Online-at-Start-up and Star-4 with TDO HI-Z when it is placed Offline-at-Start-up. This ensures the behavior specified by other start-up options is provided. With the Offline-at-Start-up Start-up option, the Dormant Drive Policy is activated at start-up to provide to the DTS the opportunity to deselect the STL of TAP.7 Controllers placed Online with a Selection Sequence without creating a drive conflict, with the Scan Topology Training Sequence and CID allocation deferred to a later point in time.

With a T3 and above TAP.7, the TOPOL Register specifies the TAP.7 Scan Topology that will be specified by OAC[3:2] during a Selection Sequence for the TAP.7 Controller to be placed Online. A T3 and above TAP.7 Controller compares the Topology Register value to the Online Activation Code bit [3:2] to determine whether the TAP.7 Controller becomes a selection candidate.

The Topology Register values specify the topology as follows:

- 00b—Star-4 with TDO HI-Z
- 01b—Series
- 10b—Star-4 with TDO drive allowed
- 11b—Star-2

At start-up, the DTS is expected to initialize the TS operation relative to the scan topology with one of the following actions:

- Doing nothing, provided there is a single TAP.7 Series Technology Branch and this is known by the DTS
- Loading the Topology Register with the value specifying the scan topology, provided there is a single TAP.7 Technology Branch, it is not a Series Technology Branch, and the Scan Topology is known
- Performing Scan Topology Training

### 20.11.1.3 Use cases

#### 20.11.1.3.1 Operation with a single TAP.7 Branch

When there is a single TAP.7 Branch and the TAP.7 Controller is Online:

- With the Topology Register set to Series at power-up, the Topology Register, when implemented may be as follows:
  - Written directly.
  - Changed with Scan Topology Training.
- With the Topology Register set to Series at power-up and subsequently placed Offline with a Deselection Escape, the TAP may be placed Online with a Selection Sequence by specifying either of the following:
  - Any topology.
  - A Series Scan Topology.

- At this point, the Topology Register may be changed as stated above.
- With the Topology Register set to Star-4 HI-Z after power-up, the TAP may be placed Online with a Selection Sequence by specifying any scan topology, provided the Selection Sequence includes a state load. At this point, the Topology Register may be changed as stated above.

### 20.11.1.3.2 Operation with more than one TAP.7 Branch

When there is more than one TAP.7 Branch, then the following occurs:

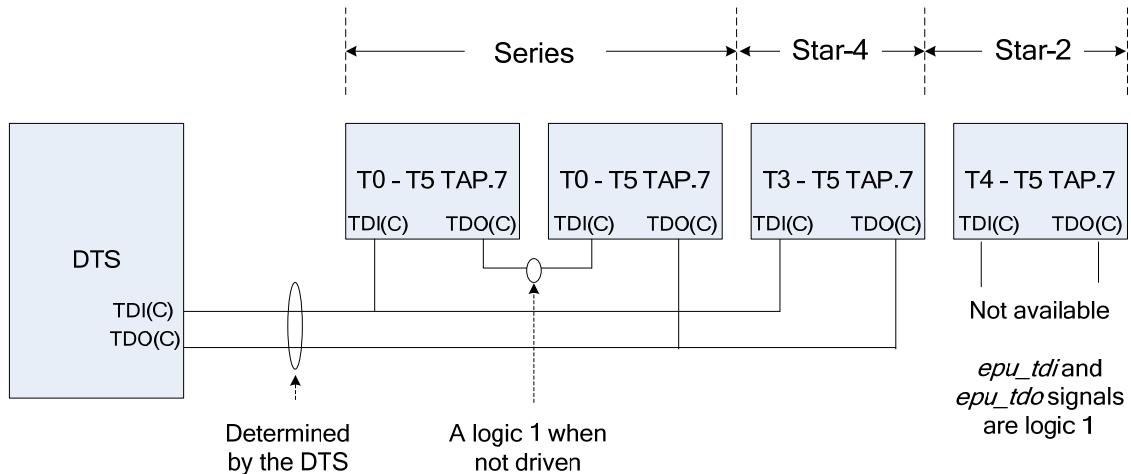
- It is expected that the DTS performs Scan Topology Training at start-up and when TAP.7 Controllers that may have been powered-down are brought Online:
  - The TOPOL Register is set to a value specifying Star-4 HI-Z operation.
  - A Scan Topology Training session follows.
- The Topology Register should be changed only with Scan Topology Training unless it is set to a value specifying Star-4 HI-Z operation prior to Scan Topology Training.
- With the Topology Register set to Series at power-up, the TAP may be placed Online with a Selection Sequence by specifying either of the following:
  - Any topology.
  - Series topologies.
- With the Topology Register set to Star-4-HI-Z at power-up, the TAP.7 Controller may be placed Online with a Selection Sequence by specifying:
  - Any topology.
  - A state load is included in the Selection Sequence.

### 20.11.1.4 Scan-path characteristics used to determine the scan topology

The following attributes of Series, Star-4, and Star-2 Scan Topologies are used to determine the scan topology in which a TAP.7 Controller is deployed:

- A Series Scan Topology:
  - Provides scan-path continuity via TDI(C) and TDO(C).
  - Only one of the TDI(C) and TDO(C) signals may be connected to the DTS unless there is only one TAP connected to the DTS.
- A Star-4 Scan Topology:
  - Provides scan-path continuity via TDI(C) and TDO(C).
  - Both the TDI(C) and TDO(C) signals are connected to the DTS.
- A Star-2 Scan Topology:
  - The EPU's TDI signal and TDO inputs are fixed at a logic 1 value when these pins are either not implemented or the TDI(C) and TDO(C) pins are used for auxiliary functions.

These characteristics are shown in Figure 20-21.



**Figure 20-21 — Scan topology characteristics used in Scan Topology Training**

These attributes are used to determine the scan topology as follows:

- A logic 0 created by the DTS at the TDIC and TDOC signals can only be observed as such by TAP.7 Controllers deployed in a Star-4 Scan Topology.
- An all-zeros scan of the EPU Bypass Bit (with an SCNB Command) using a JScan Scan Format will produce a logic 0 value in the EPU Bypass Bit of a TAP.7 Controller deployed in a Series or Star-4 Scan Topology and a logic 1 value in the EPU Bypass Bit of a TAP.7 Controller deployed in a Star-2 Scan Topology.

These two operations are sufficient to determine the scan topology as shown in Table 20-18.

**Table 20-18 — Scan topology connectivity and continuity tests**

TOPOL	Connectivity test		Continuity test
	(Only values affecting the results of the test for Star-4 Scan Topology)	(Only value affecting the results of the test for Series Scan Topology)	EPU Bypass Register
TDIC	TDOC		
Series	0b	1b	0b
Series	1b	0b	0b
Series	1b	1b	0b
Star-4	0b	0b	0b
Star-2	1b	1b	1b

### 20.11.1.5 Scan Topology Training Command Sequence

Scan Topology Training utilizes the following sequence of commands to perform the connectivity and continuity tests described in Annex D:

- CMD (STC2, APFC = Standard function)
- Configures the TDIC and TDOC pins of T4 and above TAP.7s as the T3 TAP.7 pin functions unless inhibited by the Chip-Level Logic

- CMD (STC2, TOPOL=Star-4-HI-Z, unconditional)
  - Sets the Topology Register for all T3 and higher TAP.7s to Star-4-HI-Z
- CMD (STMC, Test for Star-4 Scan Topology)
  - Tests for a Star-4 Scan Topology while the DTS drives both TDI(C) and TDO(C) to a logic 0 during the *Update-DR* state of CP2 of the command
- CMD (SCNB, Test for Series Scan Topology)
  - Tests for a Series Scan Topology with a CR Scan of all zeros whose length is longer than the longest possible scan path

The TOPOL Register records the results of connectivity and continuity tests performed by the STMC and SCNB Commands. The values of this register and their meaning are shown in Table 20-2. A Type-0-Type-3 Reset initializes this register value to 00b indicating Star-4-HI-Z when the startup option is Offline-at-Start-up-and to 01b indicating Series. Note the *Test-Logic-Reset* state (a Type-4 Reset) does not change this register value. Subsequent to this, Scan Topology Training or commands may alter this value.

#### **20.11.1.5.1 Connectivity test**

The STMC Command is used to perform the connectivity test. This command determines whether the scan topology is Star-4. The DTS drives the TDI(C) and TDO(C) signal values to a logic 0 value prior to the *Update-DR* state of Command Part Two. The logic 0 TDI(C) and TDO(C) values can be easily accomplished using the following TAPC State Machine state sequence to end Command Part Two of the STMC Command:

*Exit1-DR > Pause-DR (N) > Exit2-DR > Update-DR > Select-DR-Scan >  
Capture-DR > Exit1-DR > Pause-DR (N)*

The DTS may drive both TDI(C) and TDO(C) signals to a logic 0 beginning at a point within the first *Pause-DR* state and release the drive of the TDO(C) signal (or both signals) at a point within the second *Pause-DR* state of this series of states. TAP.7 Controllers deployed in a Star-4 Scan Topology see both the TDI(C) and TDO(C) signals as a logic 0. TAP.7 Controllers that are deployed in either a Star-2 or Series Scan Topology see at least one of these signals as a logic 1. The *Update-DR* state of this sequence samples the TDI(C) and TDO(C) values and sets the Scan Topology Register to 00b when TDI(C) and TDO(C) values are both a logic 0 and sets the register to 01b otherwise.

#### **20.11.1.5.2 Continuity test**

When the TOPOL Register value is 00b, the drive of TDO(C) is inhibited. This allows the use of an SCNB Command to perform the continuity test without a drive conflict. With this TOPOL Register value, the TDO(C) drive is inhibited until the SCNB part of the test is completed. This prevents TDO(C) drive conflicts that would otherwise be possible during the CR Scan of the SCNB Command. A Series Branch will drive TDO whereas Star-4 and Star-2 Branches will not. Only one TAP.7 Controller may drive the DTS TDO(C) connection.

The SCNB Command determines whether the scan topology is Star-2 or Series with TAP.7 Controllers that have not already determined they are deployed in a Star-4 Scan Topology. The CR Scan of this command uses all-zeros data with a length greater than or equal to the series scan-chain length of the Series Scan Topology. The EPU Bypass Bit of a TAP.7 Controller in a Star-2 Scan Topology is a logic 1 at the end of this CR Scan wheras the EPU Bypass Bit of a TAP.7 Controller in a Series Scan Topology is a logic 0 at the end of this CR Scan. This difference defines the scan topology when the *Update-DR* state of the CR Scan occurs. The *Update-DR* state of the SCNB CR Scan also converts the 00b TOPOL Register value to 10b.

## 20.11.2 Specifications

### Rules

- a) Each subsequent specification in 20.11.2 shall only apply to T3 and above TAP.7s.
- b) Changes in the Topology Register value from TAP.7 Controller commands shall be governed by Table 20-19.
- c) The Connectivity Test entry in Table 20-19 shall occur in the Command Part Two *Update-DR* state of the STMC Command specifying Star-4 Scan Topology Detect.
- d) The Continuity Test entry in Table 20-19 shall occur in the CR Scan *Update-DR* state of the SCNB Command specifying Series Scan Topology Detect.

**Table 20-19 — Topology Tests/TOPOL Register relationships**

Test	TDIC == 0 & TDOC == 0 ?	EPU BYPASS Bit == 1 ?	TOPOL Register	Resulting TOPOL Register value:
Connectivity	Yes	x	x	00b—Star-4 HI-Z
	No	x	x	01b—Series
Continuity	x	x	00b	10b—Star-4
	x	No	01b	01b—Series
	x	Yes	01b	11b—Star-2

- e) A Type-0–Type-3 Reset shall initialize the TOPOL Register value as shown in Table 20-20.

**Table 20-20 — TOPOL Register initialization**

Type-0–Type-3 Reset ?	Offline-at-Start-up ?	Resulting TOPOL value:
No	x	Governed by Table 20-9
Yes	Yes	00b—Star-4 HI-Z
Yes	No	01b—Series

## 20.12 Managing STL Group Membership

### 20.12.1 Description

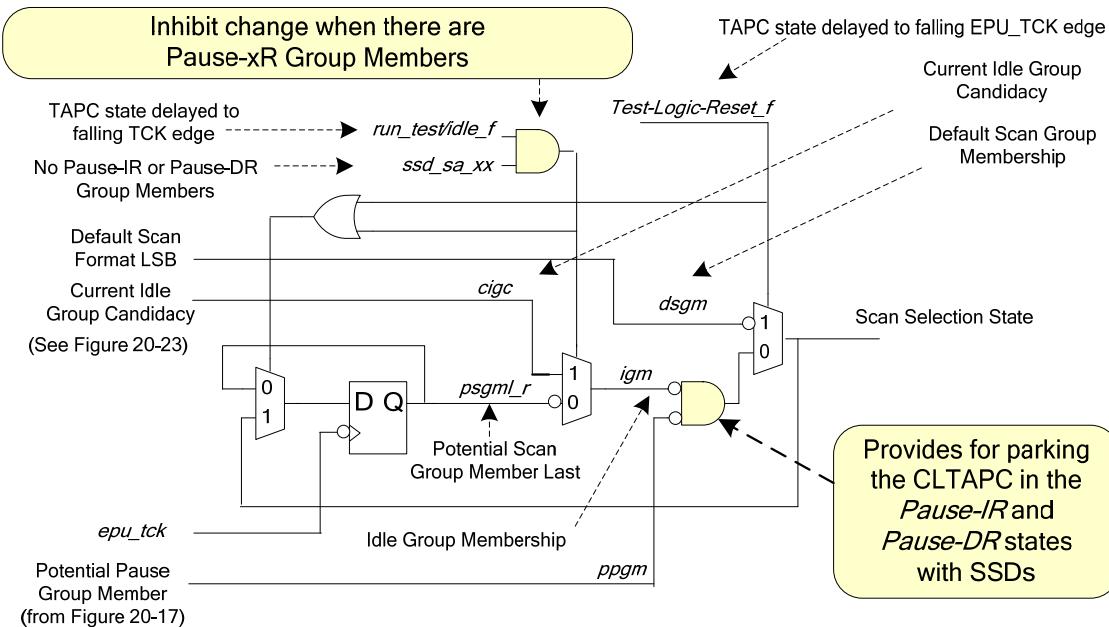
#### 20.12.1.1 Factors affecting group membership

Recall that with a T3 and above TAP.7, the STL of T3 and above TAP.7 is a member of the *Pause-IR*, *Pause-DR*, Idle, or Scan Groups. STL Group Membership is determined as described in 13.8. With a T3 TAP.7, the effects of Type-0–Type-5 Resets, the use of the Control Path, and the TAPC state are the same as with a T2 TAP.7.

#### 20.12.1.2 An approach to implementing CLTAPC selection with T3 and above classes

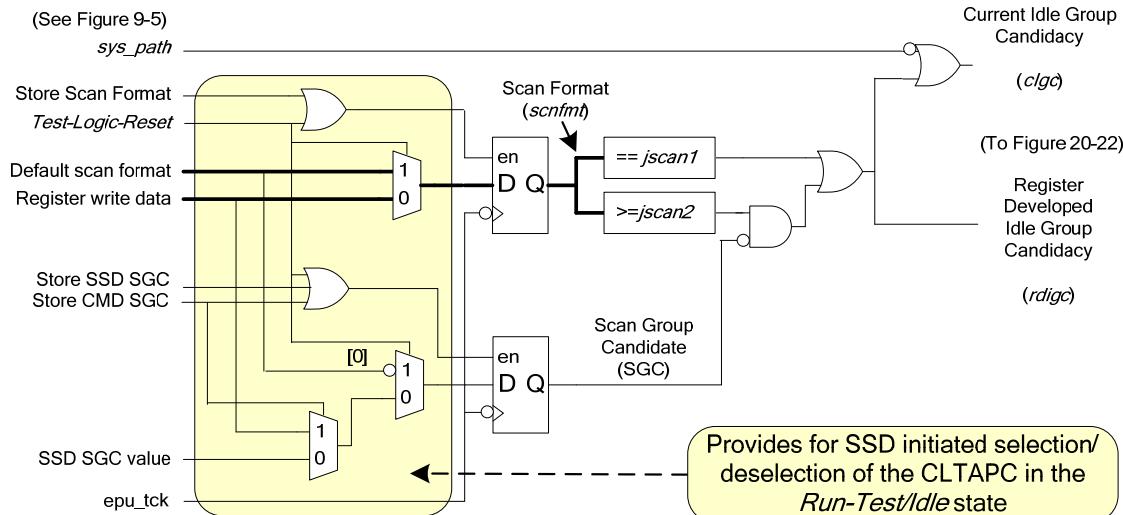
A conceptual view the management of group membership for T3 and above TAP.7s is shown in Figure 20-22. It is a superset of the management of group membership for a T2 TAP.7. Additions to support SSDs are

highlighted in this figure. The Current Scan Group Candidacy is developed with SSDs affecting the SGC Register bit as shown in Figure 20-23.



**Figure 20-22 — Conceptual view of current Scan Group Candidacy development—T3 and above TAP.7s**

A conceptual view of Current Scan Group Candidacy for T3 and above TAP.7s is shown in Figure 20-23. The changes to the T2 TAP.7 version of this function to support SSD-initiated coupling and decoupling of the CLTAPC in the *Run-Test/Idle* state are highlighted.

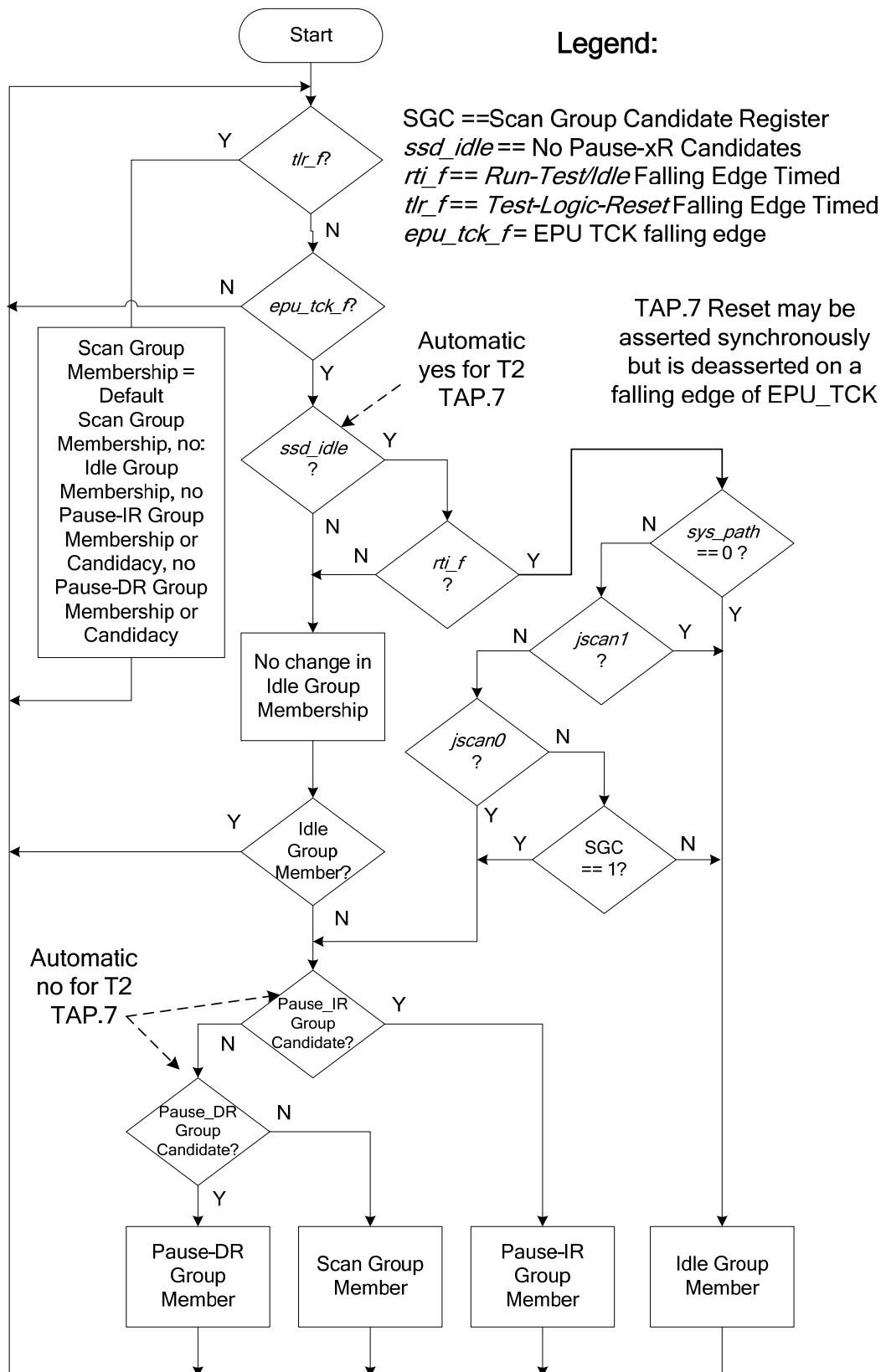


**Figure 20-23 — Current Scan Group Candidacy development T3 and above TAP.7s**

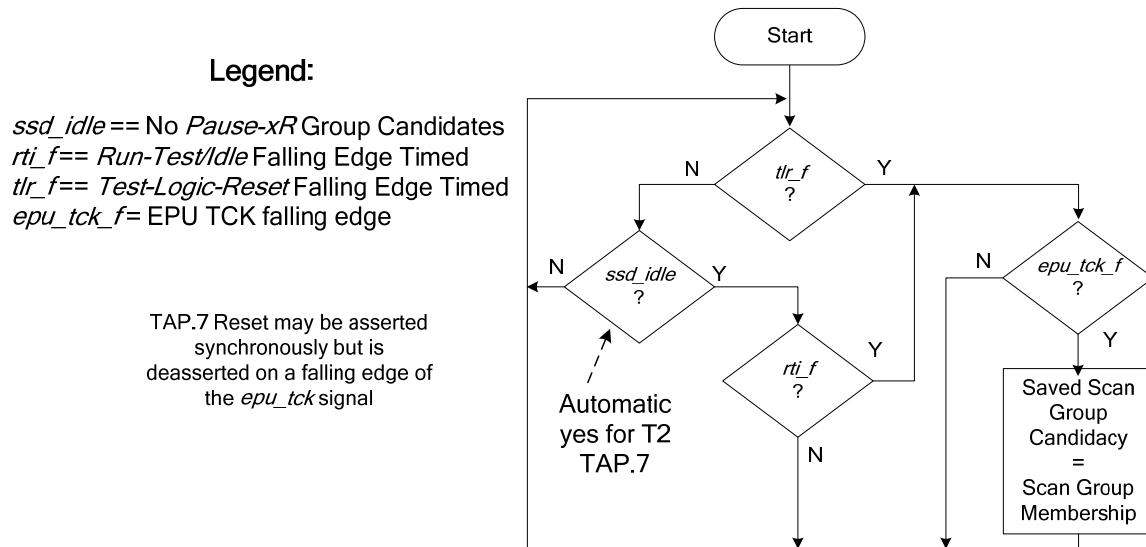
## 20.12.2 Specifications

### Rules

- a) Each subsequent specification in 20.12.2 shall only apply to a T2 and above TAP.7s unless stated otherwise.
- b) Scan Group Membership shall be governed by Figure 20-24 and Figure 20-25.

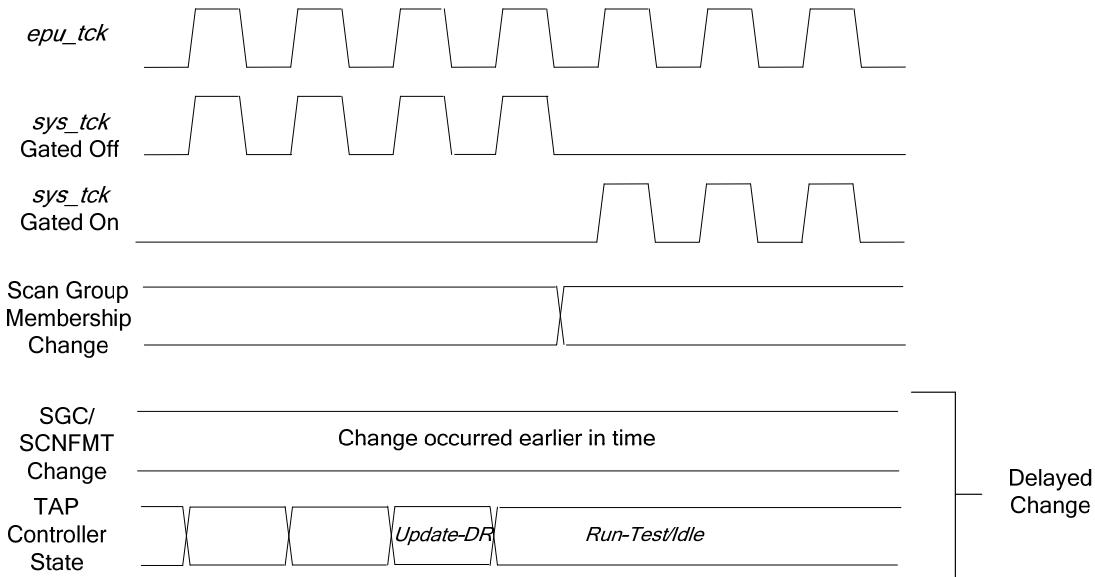


**Figure 20-24 — Group membership operation**



**Figure 20-25 — Saved Scan Group Candidacy operation**

- c) Changes to the SCNFMT or SGC Register values shall affect Scan Group Membership as shown in Figure 20-26.



**Figure 20-26 — *sys\_tck* gating/Scan Group Membership change relationships**

## 20.13 RSU operation

### 20.13.1 Description

An RSU implemented with a T3 TAP.7 has the characteristics described in Clause 11 subject to the rules in 20.13.2.

## 20.13.2 Specifications

### Rules

- a) Each subsequent specification in 20.13.2 shall apply to T3 TAP.7.
- b) The TAP.7 Controller shall be placed Online following the last bit of the Check Packet completing the Selection Sequence, provided all of the following are true:
  - 1) The TAP.7 Controller is a selection candidate.
  - 2) The scan format is JScan0–JScan3.

## 20.14 Programming considerations

Important T3 and above TAP.7 programming considerations are listed as follows:

- The interaction of SSDs and other factors described in 20.10.1.4 and 20.10.1.7 should be kept in mind.
- The use of a command to directly storing the TOPOL Register value should be used carefully as storing this register when the following occurs:
  - ADTAPCs of multipled branches are selected (for example, when multiple branches are selected at start-up).
  - The number and types of branches selected is unknown.
  - The value of the register does not reflect the type of branch that is selected.
  - The concurrent use of Selection/Deselection Escapes and SSDs is considered a programming error.

Selection/Deselection Escapes manage selection/deselection at Levels E/D of the TAPC hierarchy shown in Figure 7-5. SSDs manage selection/deselection at Level C of this hierarchy. There is never a need to manage both Levels E/D and Level C simultaneously.

## 21. Advanced concepts

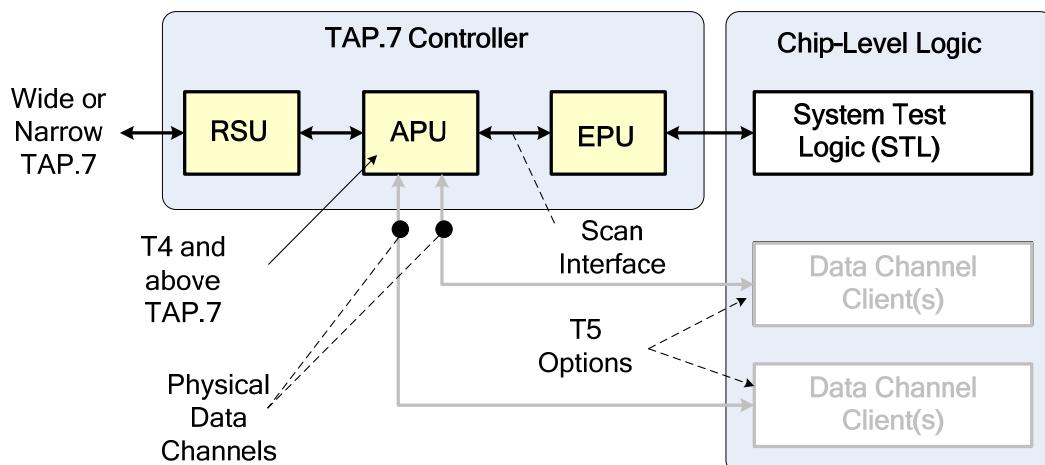
This clause is applicable to T4 and above TAP.7s. It describes the concepts that support the implementation and use of the Advanced Protocol. It extends the high-level description of the concepts used with T4 and T5 TAP.7s provided by Clauses 4 through 8, Clause 15, and Clause 17.

The subject matter within this clause is described in the following order:

- 21.1 Architecture
- 21.2 Advanced capabilities
- 21.3 Comparing the Advanced and Standard Protocols
- 21.4 APU functions
- 21.5 APU interfaces
- 21.6 APU function/Operating State relationships
- 21.7 TAPC state/packet relationships
- 21.8 User's and implementer's views of the Advanced Protocol
- 21.9 An approach to scheduling APU Operating State scheduling
- 21.10 Structure of the clauses describing T4 and above TAP.7s
- 21.11 Structure of the clauses describing T4 and above TAP.7s

### 21.1 Architecture

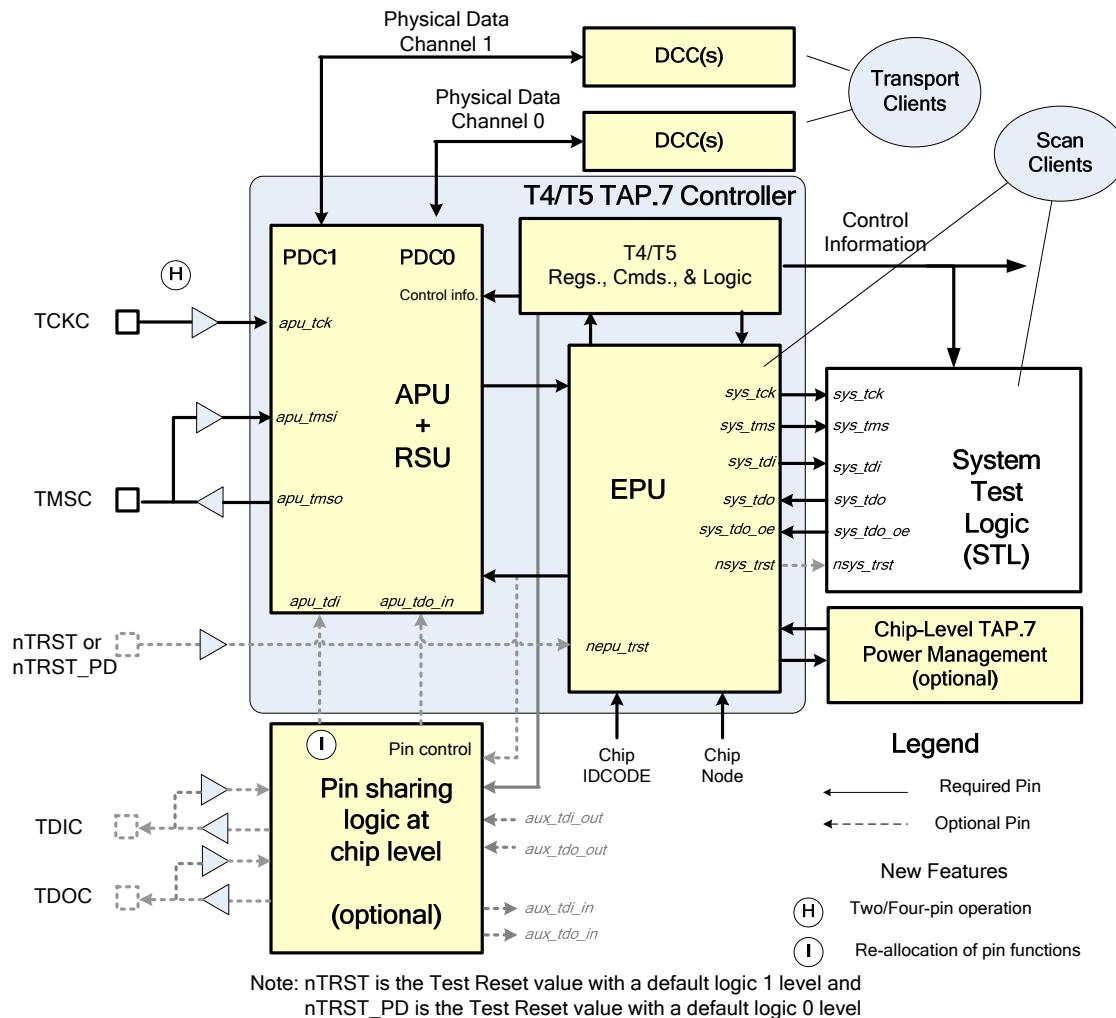
The APU is inserted between the RSU and EPU to create both wide and narrow T4 and above TAP.7s as shown in Figure 21-1. All features added with a T4 and above TAP.7 are managed with the RSU/APU combination. Both the Standard and Advanced Protocols may be used with these TAP.7s. The use of the Standard Protocol is limited to TAP.7 Controller command generation when the T3 TAP.7 TDIC and TDOC signal functionality is not available.



**Figure 21-1 — T4 and above TAP.7 chip block diagram**

Figure 21-2 shows a conceptual block diagram of fully deployed T5 TAP.7 capability. It shows the APU's relationship to the EPU, Chip-Level Data Channel Clients, and logic that provides for the sharing of the

TDIC and TDOC pins with non-Scan Functions. The T4 TAP.7 is a subset of the T5 TAP.7, with logic supporting Physical Data Channels deleted from this figure.



**Figure 21-2 — Conceptual view of fully deployed APU functionality**

## 21.2 Advanced capabilities

### 21.2.1 Mandatory and optional capabilities

A high-level description of the mandatory and optional capabilities of a T4 TAP.7 follows. A minimal set of mandatory capability is added with both the T4 and T5 TAP.7s. The majority of the capability associated with these TAPs is optional. The capabilities of these classes include the following:

- Mandatory capability that is inherited from a T3 TAP.7:
  - Full T3 TAP.7 capability with a wide TAP.7
  - Abbreviated T3 TAP.7 capability (TAP.7 Controller commands) with a narrow TAP.7
- Mandatory capability is added (a minimal set):
  - For T4 and T5 TAP.7s—MScan, OScan0, and OScan1 Scan Formats providing basic test and debug capability

- For a T5 TAP.7—Tracking Data Channel activity with no data transferred (zero Physical Data Channels) and remaining synchronized to the Advanced Protocol when transport functionality is used)
- Placement Offline before a Configuration Fault causes a loss of synchronization (making chips with different sets of T4 and T5 TAP.7 options interoperable)
- Stall of a scan transfer at any TAPC state (by both the DTS and STL)
- One of four start-up options described in 10.3 will be deployed
- Optional T4 TAP.7 capability:
  - Scan formats to improve Advanced Protocol scan efficiency
  - Scan formats to facilitate debug specific high performance scan operations
- Optional T5 TAP.7 capability:
  - One or two Physical Data Channels
  - One to 16 clients sharing a Physical Data Channel (only one is connected to the Physical Data Channel at a time)
  - Interoperability with other chips with different sets of T4 and T5 TAP.7 options is provided

### **21.2.2 Online and Offline operation**

With Online operation, the ADTAPC remains synchronized to the DTS-generated TAPC state sequence. The EPU processes TAP.7 Controller commands delivered with the Advanced Protocol via the APU or with the Standard Protocol with the APU bypassed. With Offline operation, the TAP.7 Controller awaits being placed Online. The Online/Offline operation of all TAP.7s is described in Clause 11.

### **21.2.3 Interoperability with T0–T3 TAP.7s**

A narrow TAP.7 cannot be deployed in a Series Scan Topology as the TDIC and TDOC signals are not implemented. Wide T4 and T5 TAP.7s are interoperable with TAP.1s and T0–T3 TAP.7s when they are deployed in a Series Scan Topology and the TDIC and TDOC signals are either of the following:

- Fixed as the T3 TAP.7 TDIC and TDOC functions
- Programmable with one selectable function being the T3 TAP.7 TDIC and TDOC functions, and this function selected

### **21.2.4 Interoperability with T4 and above TAP.7s**

Wide T4 and above TAP.7s are interoperable regardless of their feature set. This interoperability is provided even though most T4 and T5 TAP.7 features are options and many of these optional features change the Advanced Protocol bit sequences.

Interoperability is ensured by the way the RSU handles Configuration Faults. Recall that a Configuration Fault is a combination of TAP.7 Controller state and register values that produce unsupported Advanced Protocol bit sequences. The TAP.7 Controller places itself Offline before a loss of synchronization occurs when a Configuration Fault is detected. This occurs before the unsupported Advanced Protocol bit sequences occur, thereby preventing a loss of synchronization with the DTS. The TAP.7 Controller may later be placed Online.

A Configuration Fault is declared when any of the following conditions is detected:

- Any of the following with a T4 and above TAP.7:
  - An unsupported scan format.
  - A scan format with RDY bits in the SP when the voting on the value of RDY bits is unavailable but may be or is needed.
- With a T4 TAP.7, data transport capability is enabled.
- With a T5 TAP.7, the use of an unsupported data transport protocol when the data transport capability is enabled.

## 21.3 Comparing the Standard and Advanced Protocols

With the Standard Protocol:

- The TCKC and TMSC signals are sufficient to:
  - Advance the TAPC state.
  - Manage the TAP.7 Controller capabilities.
- The TMSC signal conveys TMS information every TCKC period.
- The TDIC and TDOC signals are required for data transfers.

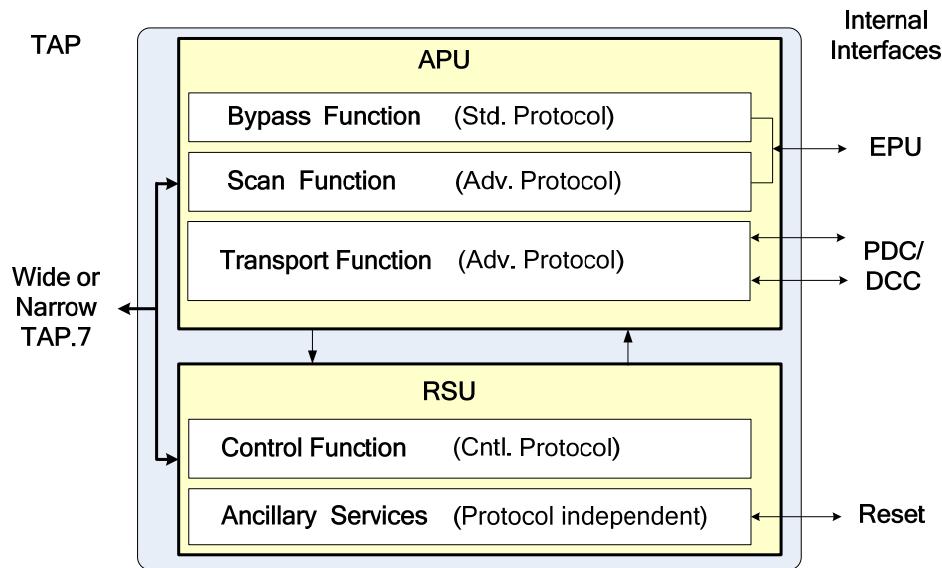
With the Advanced Protocol:

- The TCKC and TMSC signals are sufficient to:
  - Advance the TAPC state.
  - Manage the TAP.7 Controller capabilities.
- Support data transfers with the STL Scan Paths and data/control data transfers with DCCs in the STL.
- The TMSC signal becomes bidirectional and is used to transfer serialized:
  - Scan client TDI, TMS, TDO, and control information.
  - Transport client serialized control and data information.
  - TAP.7 Controller information.
- The TDIC and TDOC signals are not required.
- The Advanced Protocol content changes as different capabilities are enabled.

## 21.4 APU functions

### 21.4.1 Conceptual view

A conceptual view of the APU and RSU functions is shown in Figure 21-3.



**Figure 21-3 — Conceptual block diagram of the APU and RSU functions**

#### 21.4.2 Bypass Function

The Bypass Function supports the use of the Standard Protocol. With a wide TAP.7, it connects the EPU directly to the TAP signals, bypassing the protocol converter within the Scan Function. The Standard Protocol provides the functionality of a T3 TAP.7 when the T3 TAP.7 TDIC and TDOC pin functionality is available. When the TDIC and TDOC signal functions are not available (a narrow TAP.7 or allocated to an auxiliary function), then the APU does the following:

- Connects the TAP's TCKC and TMSC signals to the EPU's TCK and TMS signals
- Connects the EPU's TDI and TDO\_IN signals to a logic 1
- Ignores the *sys\_tdo* and *sys\_tdo\_oe* signals generated by the STL

While using the Bypass Function with the TDIC and TDOC signal functions not available:

- Two-part commands are fully functional
- The function of three-part commands is altered as follows:
  - The CR Scans of SCNS and SCNB commands can only store register bits as a logic 1.
  - The CIDA Command cannot allocate CIDs as Directed CID Allocation with all-ones TDI data and as Undirected CID Allocation with all-ones TDO\_IN data disqualifies CID allocation participants.
  - SSD detection is inhibited by all-ones TDI data.
- The Scan Topology Training Sequence determines the TAP.7 Controller is operating in a Star-2 Scan Topology.

The Standard Protocol may be used to initiate the use of the Advanced Protocol and change other APU and EPU characteristics at all times, independently of the availability of the T3 TDIC and TDOC functions. This capability is sufficient to place a *BYPASS* instruction in the CLTAPC and EMTAPC Instruction Registers and subsequently utilize two-part TAP.7 Controller commands.

### 21.4.3 Scan Function

The combination of the Scan Function and the RSU provide the functionality of a five-pin TAP.1 using only the TCKC and TMSC signals. The APU functions as a protocol converter, exchanging TMS, TDI, and TDO information between the DTS and the EPU and CLTAPC using only the TAP.7 TMSC signal. The APU converts the Scan Packets used at the DTS/TAP interface to the Standard Protocol signaling used at the ADTAPC/CLTAPC interface.

Because the Advanced Protocol serializes and deserializes the EPU's TMS, TDI, and TDO information, the ADTAPC state advances at a rate slower than the TCKC frequency in most but not all cases. This throughput penalty can be offset or diminished in most instances by a combination of doubling the TCKC frequency when falling-edge to falling-edge timing is used and exchanging only the required TMS, TDI, and TDO information (minimizing the information exchanged for each TAPC state).

The entire T3 TAP.7 EPU infrastructure (i.e., ZBS detection, control-level management, and command generation) is utilized. This infrastructure is not affected by the use of the Advanced Protocol as it is controlled entirely by the TAPC state progression. Both two- and three-part commands are fully functional when they are used with the Scan Function. The TAP.7 Controller addressability in a Star-2 Scan Topology is the same as for the Star-4 Scan Topology. The use of SSDs is the same as with the Star-4 Scan Topology, with TDI information embedded in Scan Packets.

The following description expands the description of the scan formats in 4.2.7.7, the T4 TAP.7 in 6.2, and the drive policies associated with them in Clause 14. With T4 and above TAP.7s, three mandatory and ten optional scan formats are added to the JScan Scan Formats to support scan transfers (MScan, OScan0–OScan7, and SScan0–SScan3). Some of these scan formats provide maximum flexibility, others provide maximum performance, and still others provide a blend of scan performance and flexibility. Recall that performance is maximized by minimizing the information exchanged for each TAPC state while flexibility is maximized by transferring the maximum amount of information for each TAPC state. Scan formats with the maximum amount of information accommodate both TS and DTS stalls of the TAPC state progression and support the exchange of TDI and TDO information in all TAPC states (supporting IEEE 1149.1-Non-disruptive Behavior of STL components).

The MScan Scan Format has the largest Scan Packet payload, making it the most flexible scan format. The OScan and SScan Scan Formats provide two forms of the same functionality, the first optimized for systems where the TCKC signal is sourced by the TS and the second for systems where the TCKC signal is sourced by the DTS. The OScan Scan Formats provide various levels of optimization with the optimizations for *Shift-xR* and non-*Shift-xR* states handled differently for some of the OScan Scan Formats. The SScan Scan Formats provide for highly optimized transfers targeting a number of different use cases related to debug (see 26.1.2 and Annex G).

The scan formats may be used as follows:

- MScan: When there are any number of Scan Group Members
- OScan: When there are:
  - No RDY bit(s) in the SPs With any number of Scan Group Members when there is an RDY bit in the SPs
  - RDY bit(s) in the SPs One Scan Group Member when the TAP.7 Controller determines there is no more than one:
    - Scan Group Candidate
    - Potential Scan Group Member

- SScan: When the TAP.7 Controller determines there is no more than one of the following:
  - Scan Group Candidate
  - Potential Scan Group Member

A configuration fault is declared when the TAP.7 Controller detects both of the following conditions:

- The use either an SScan Scan Format or an OScan Scan Formats with a RDY bit in the SPs is specified.
- More than one Scan Group Member or more than one Potential Scan Group Member.

The Advanced Scan Formats provide the STL a means to stall the TAPC state progression as follows:

- MScan Independent of the number of Scan Group Members
- OScan With one Scan Group Member when there are RDY bits in the SPs
- SScan With one Scan Group Member when there are RDY bits in the SPs

The Advanced Scan Formats support CID allocation as follows:

- MScan Both the directed and undirected methods of CID allocation
- OScan Only the directed method of CID allocation
- SScan No CID allocation

The CID-allocation criteria are the same for both the Standard and the Advanced Protocol with the exception that 36-bit CR Scans used for both the directed and undirected methods. The information used with both the directed and undirected CID-allocation methods is conveyed entirely with the TMSC signal (including the value of the TDO bit created by voting during the arbitration process).

#### 21.4.4 Transport Function

The Transport Function supports APU configurations implementing zero, one, or both of the APU's Physical Data Channels. It maintains synchronization with the Advanced Protocol with all of these configurations. In the configuration where there are no PDC/DCC connections, the TAP.7 Controller remains synchronized to the Advanced Protocol during the *TPA* state but no data is transferred. When one or two Physical Data Channels are implemented, the Transport Function provides for the exchange of data via the TAP.7 TMSC signal between the following:

- The DTS and a Chip-Level Data Channel Client
- Data Channel Clients connected to Physical Data Channels located in the same or different chips
- Both of the above

The following description expands the description of the Data Transport Function in 6.3 and the drive policies associated with it in Clause 14. Non-scan data may be exchanged with a Chip-Level Data Channel Client that is connected to one of the APU's two PDCs (PDC0 and PDC1) shown in Figure 21-1. These PDCs are referenced by a PCA. This address is created from the combination of the TAP.7 Controller's CID and the Physical Data Channel number (one or zero). Only one DCC is connected to a Physical Data Channel at a time. The connection of up to 16 different DCCs to a Physical Data Channel is supported, with their connection to the Physical Data Channel scheduled by the DTS.

The PCA may be aliased to a Logical Channel Address (LCA). The Logical Channel Address references an LDC. One or more Physical Channel Address may be aliased to the same Logical Channel Address,

associating them with the same Logical Data Channel. Aliasing a single Physical Channel Address to a Logical Channel Address provides for transfers between a PDC and the DTS. Aliasing more than one Physical Channel Address to the same Logical Channel Address provides for transfers between PDCs and between the DTS and between PDCs. This configuration requires a higher level protocol manage TMSC drive during these transfers to avoid drive conflicts.

Eight Logical Data Channels are supported. The DTS may sequentially exchange data with Logical Data Channels with these exchanges activated using the Logical Channel Address. These exchanges are initiated with Transport Directives specifying the LCA. An unimplemented Physical Data Channel cannot be allocated a Logical Channel Address. The Physical Data Channel Registers, logic associating Logical Channel Address with a Physical Data Channel, and logic creating the Physical Data Channel/DCC interface/data path are implemented only when a Physical Data Channel is implemented.

In the simplest case, a Logical Channel Address is allocated to only one Physical Channel Address. This provides for a data exchange between the DTS and the DCC connected to the Physical Data Channel using one of the eight Logical Channel Addresses. In more sophisticated cases, a Logical Channel Address may be allocated to more than one Physical Data Channel as stated previously. This provides for non-scan data exchanges between the DTS and the DCCs connected to these Physical Data Channels and non-scan data exchanges between these DCCs. When the Logical Channel Address is allocated to multiple Physical Data Channels, the DCCs connected to the Physical Data Channel will understand this. They will utilize a higher level protocol to avoid TMSC drive conflicts as mentioned previously. It is expected that the dominant use case will be a Logical Channel Address allocated to a single Physical Data Channel.

Certain Transport Function registers within the TAP.7 Controller (TPST and TPPREV) Registers are written with TAP.7 Controller commands. The remaining Transport Function Registers within the TAP.7 Controller are written with Transport Directives. All these registers may be read with the RDBACK1 Register when this register is implemented. Other Transport Registers within the STL are also written with Transport Directives.

A number of Transport Directives support the operation of the Transport Function. These directives are embedded in the Advanced Protocol and perform a variety of operations. These directives provide a means to specify the operating characteristics of the Transport Function, initiate Data Channel Client data transfers, and read and write Data Channel Client Registers managed by the DTS.

A register-write may either store a value in a register in a single Data Channel Client or store a value the same register written in all Data Channel Clients accessed with the same Logical Channel Address. A register-read returns the value of the specified register from a single Data Channel Client.

With data transfers, the Data Channel Client specifies the direction of the transfer as DTS to TS, TS to DTS, Bidirectional (50% In /50% Out), or Custom. With a custom transfer the Data Channel Client defines the transfer direction for each bit of the transfer.

The capability provided by the Transport Function is summarized in Table 21-1.

**Table 21-1 — Transport Function capability**

<b>PDC implemented?</b>	<b>Transport enabled?</b>	<b>Capability</b>
No	No	Operates as a T4 TAP.7, no data transport
No	Yes	Remains Online when transport is used, no data may be exchanged as there is no DCC connection
Yes	No	Operates as a T4 TAP.7, no data transport
Yes	Yes	Remains Online when transport is used, data may be exchanged with a DCC

### 21.4.5 Bypass/Scan/Control Function interactions

The RSU's Control Function handles Offline/Online Operation using the Control State Machine as with lower classes. With a T4 and above TAP.7s, the role of the Control Function (the CSM) is expanded beyond processing Selection/Deselection Escapes and Alerts to check for a Configuration Fault following a command.

This check is performed by the Check Packet that completes the use of the Control Protocol. This check is initiated by the following CSM state transitions (see Figure 11-17):

- *STD* to *CHK* Following Command Part Two of an STFMT Command instantiating the use of the Advanced Protocol as shown in Figure 6-4
- *ADV* to *CHK* Following Command Part Two of any command as shown in Figure 6-5

These state transitions pass control of the TMSC signal to the Control Function. The resulting Check Packet places the TAP.7 Controller Offline (*CHK* to *OLW*) when the check detects any of the Configuration Faults described in 21.2.3 is detected. In the absence of a Configuration Fault, the Control Function initiates the use of the Standard (*CHK* to *STD*) or the Advanced Protocol (*CHK* to *ADV*) based on the scan format. The Scan Function controls the TMSC signal in the *ADV* state while the Bypass Function controls the TMSC signal in the *STD* state.

## 21.5 APU interfaces

### 21.5.1 TAP

A T4 and above TAP.7 may implement either a wide (four-signal) or narrow (two-signal) TAP. With a wide configuration, the use of the Advanced Protocol does not require the use of the TDIC and TDOC signals. The function provided by these signals may be either fixed as the T3 TDIC and TDOC signal functionality or switched between two sets of functionality: the T3 TAP.7 signal functionality and an auxiliary function. The start-up option defines the initial signal functionality (as one of the above) following a Type-0-Type-3 Reset. Subsequently, the function of these signals is controlled by the APFC Register. Functional logic may lock the value of this register to prevent the DTS changing the default function of these signals as they may be performing system functions. The nTRST and nTRST\_PD TAP signals may be included with either the wide or narrow TAP configurations.

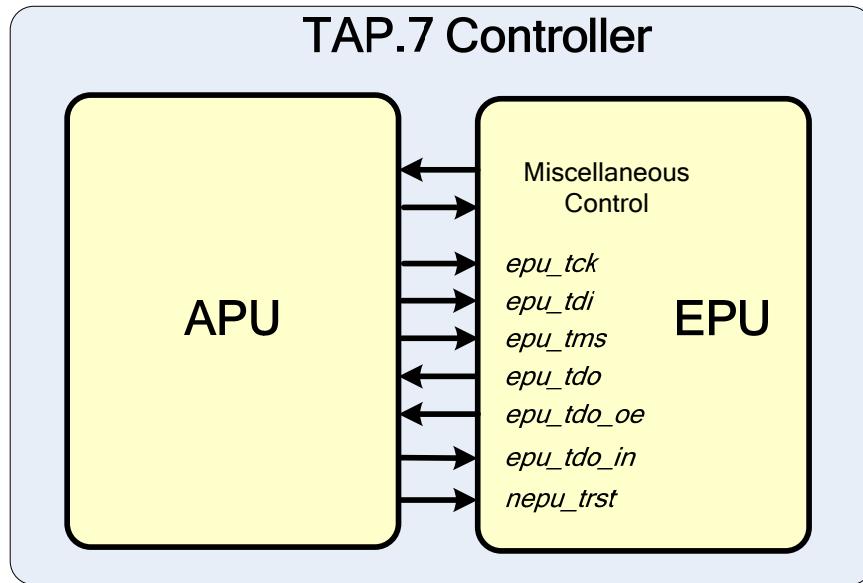
The TAP signal behavior is determined by the factors shown in Table 21-2. The entries for the TDIC and TDOC signal are applicable only to wide T4 and above TAP.7s.

**Table 21-2 — TAP signal behavior**

<b>Signal ?</b>	<b>Standard Protocol ?</b>	<b>Signal function determined by:</b>	<b>Drive inhibited by:</b>	
TCKC	N/A	Fixed as Test Clock		
TMSC	Yes	Connection to EPU's TMS input	N/A	
	No	The function using the signal	TMSC Drive Policy	
TDIC	Yes	AFPC Register value + Start-up option	N/A	
	No		The function of the signal utilizing the signal	
TDOC	Yes	AFPC Register value + Start-up option	TDOC Drive Policy	
	No		The function of the signal utilizing the signal	

### 21.5.2 EPU

The APU/EPU interface is shown in Figure 21-4. It is essentially an IEEE 1149.1 TAP interface without the input and output buffers required to drive the TAP pins plus other miscellaneous signaling such as APU control with EPU registers. The miscellaneous signaling between the APU and EPU is implementation specific. The EPU is unaware of the existence of the RSU and APU other than the processing of APU-generated Type-3 Reset Requests.

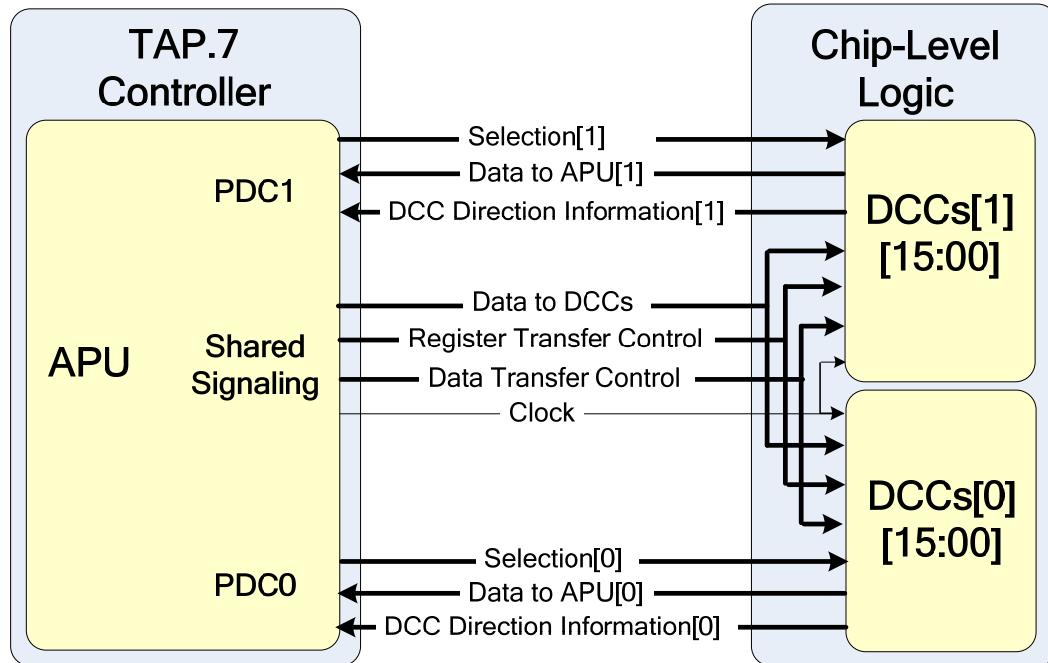


**Figure 21-4 — APU/EPU interface**

### 21.5.3 DCC

A conceptual view of the APU/DCC interface is shown in Figure 21-5. Each PDC supports connection to up to 16 DCCs. Only one DCC may be connected to a PDC at a time with the APU selecting the DCC that

is connected. The APU is the initiator on this interface with the DCCs that may be connected to this interface are responders.



**Figure 21-5 — APU/DCC(s) interface**

The APU defines the following elements:

- Transfer type and timing with a set of control signals shared by the DCCs.
- Direction of register transfers.
- Direction of data transfers as specified by the DCC connected to the Physical Data Channel performing the transfer.

Transport drive conflicts are avoided as follows:

- Register transfers allow transfers to/from PDC(s) selected for register transfers with:
  - Write operations having:
    - The TMSC signal sourced with data information by the DTS during a register-write operation
    - The TMSC signal value forwarded to one or more PDCs and the DCCs connected to them
  - Read operations having:
    - The TMSC signal sourced by only one PDC with the register data provided by the DCC connected to the PDC
    - The TMSC data forwarded to the DTS
- Data transfers allow transfers between the DTS and one or more PDCs, or between one or more PDCs selected for a data transfer with:
  - The TMSC signal sourced with data information provided by one of the following:
    - The APU
    - One or more PDCs
    - One or more DCCs

- The DTS sourcing the TMSC signal
- A single PDC, with the data supplied by the DCC connected to it
- The TMSC signal value forwarded to any of the following destinations:
  - One or more PDCs, with the data forwarded to the DCCs connected to them
  - The DTS

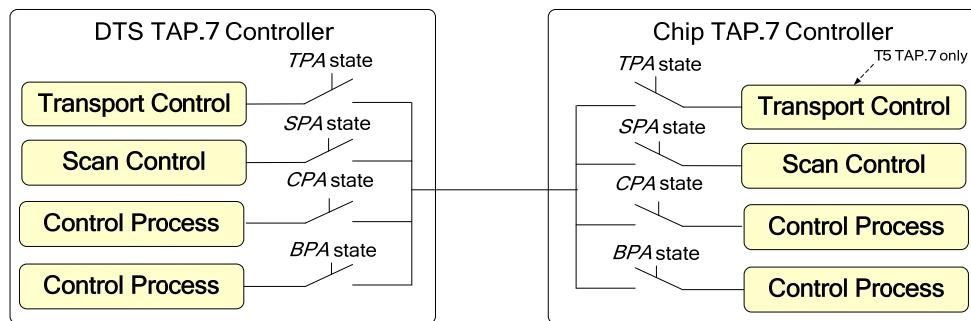
In the case of data transfers where multiple PDCs are allocated the same LCA, a higher level protocol is used to determine which of these PDCs is the following:

- Source of the TMSC bit value (the DCC directly controls the TMSC signal drive)
- Destination of the TMSC bit value as this value is forwarded to all PDCs with the LCA

## 21.6 APU function/Operating State relationships

### 21.6.1 Operating State/function relationships

Control of the TMSC signal and its connectivity is passed between the Bypass, Scan, Control, and Transport Functions based on the operating state as shown in Figure 21-6. The APU Operating States and their functional relationships are shown in Figure 21-7. The TAP.7 control sequences created with Check Packets, Scan Packets, and Transport Packets open and close these switches. Only one pair of switches (one in the DTS and one in the TAP.7 Controller) is closed at a time, providing exclusive use of the TMSC signal to only one of these functions at a time. The RSU's protocol independent ancillary services are described in Clause 10.



**Figure 21-6 — Conceptual view of TAP.7 use with a T5 TAP.7 Controller**

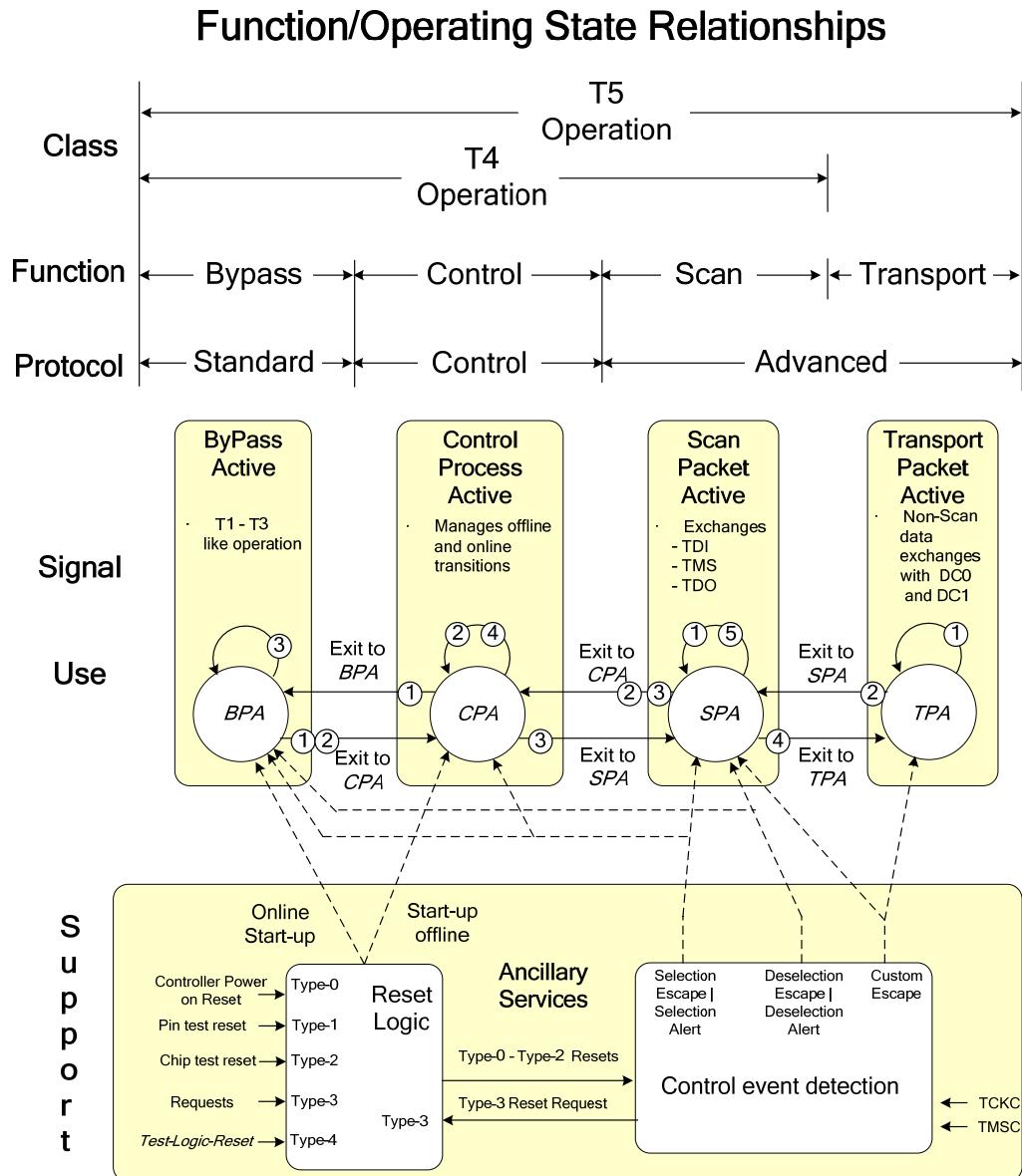


Figure 21-7 — APU functions/APU Operating State relationships

### 21.6.2 APU Operating State changes

The events causing changes in the APU Operating State are described in Table 21-3.

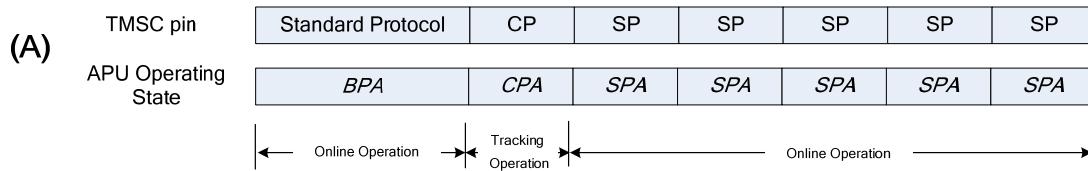
**Table 21-3 — Summary of APU Operating State change relationships**

State	Exit arc.	Description
<i>BPA</i>	①	A qualified Selection or Deselection Escape or Alert.
	②	The <i>Update-DR</i> state of CP2 of an STFMT is storing a scan format other than a JScan Scan Format in the SCNFMT Register.
	③	Neither ① nor ②.
<i>CPA</i>	①	The TAP.7 Controller is Online, a CP ends, and the SCNFMT Register specifies the use of a JScan Scan Format.
	②	The TAP.7 Controller is Online, a CP ends, and the TAP.7 Controller is placed Offline because of a Configuration Fault.
	③	The TAP.7 Controller is Online, a CP ends.
	④	Neither ①, nor ②, nor ③.
<i>SPA</i>	①	The SP has not completed.
	②	A qualified selection or Deselection Escape or Alert.
	③	The SP is associated with the <i>Run-Test/Idle</i> or <i>Select-DR-Scan</i> TAPC state following the <i>Update-DR</i> state of CP2.
	④	The SP is associated with a TAPC state that is enabled for transport.
	⑤	Neither ①, nor ②, nor ③, nor ④.
<i>TPA</i>	①	TP has not completed.
	②	Not ①.

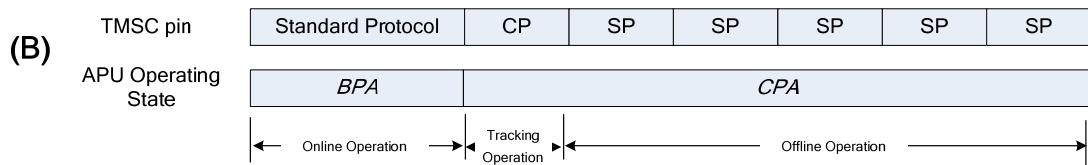
### 21.6.3 Example operating state sequences

A number of TAP.7 Controller operating state sequences are illustrated in Figure 21-8. These cases are described in Table 21-4.

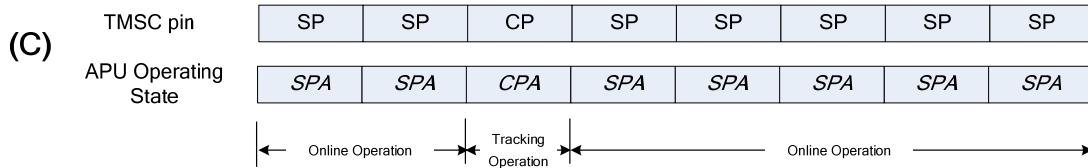
### Case      Online Operation after Standard-to-Advanced Protocol change



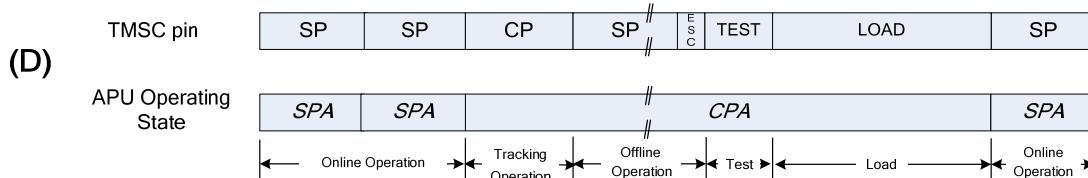
### Offline Operation after Standard-to-Advanced Protocol change



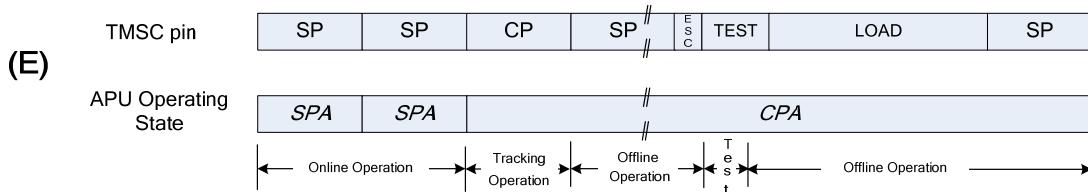
### Advanced Protocol - Online operation following commands



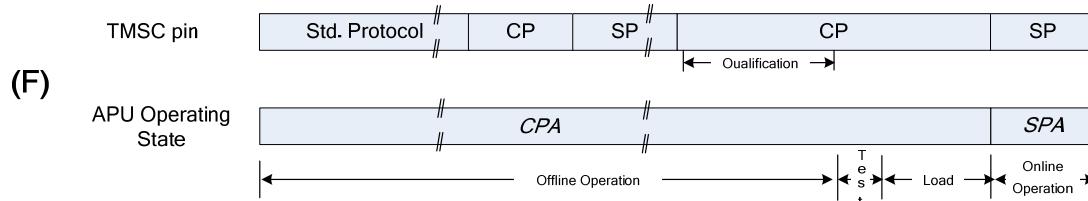
### Adv. Protocol - Offline Operation after a Command, Re-synchronization



### Adv. Protocol - Offline after a Command, No Re-synchronization



### Offline-at-Start-up, Synchronization



**Figure 21-8 — Operating state sequence examples**

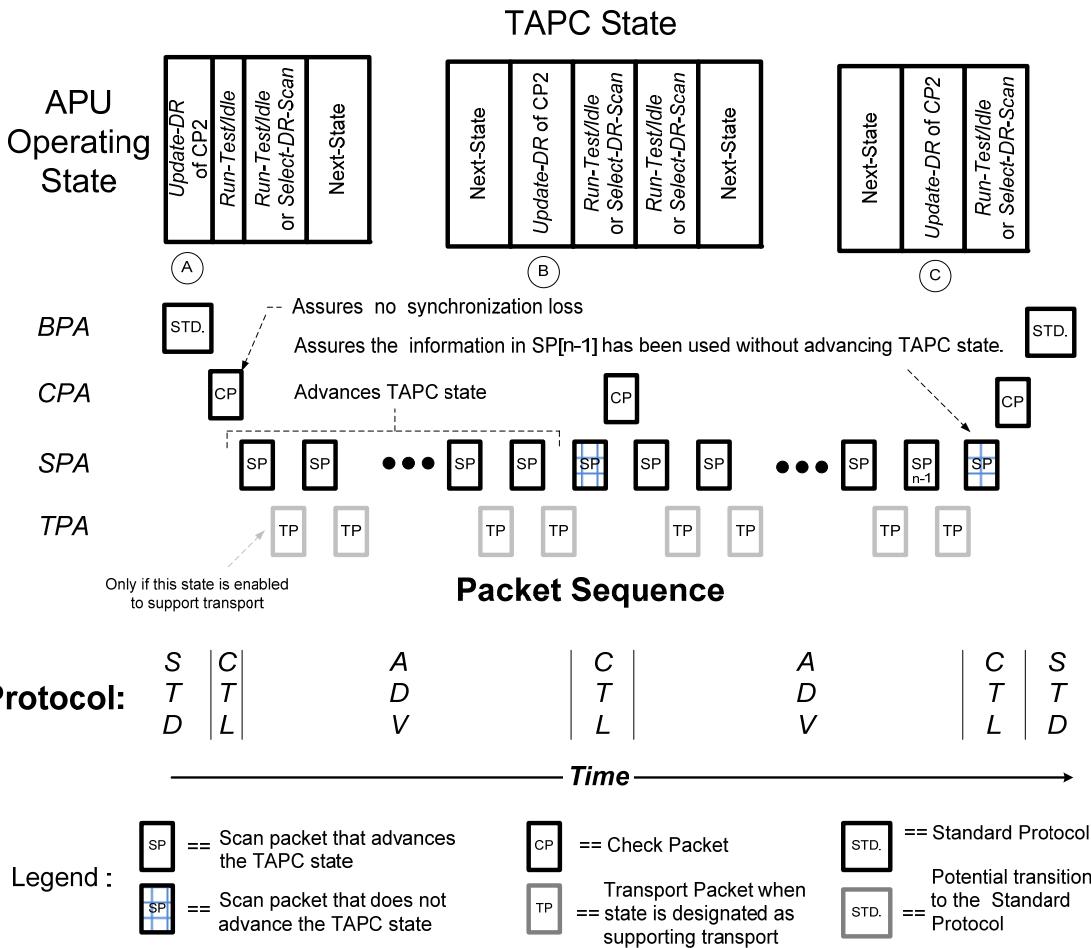
**Table 21-4 — Description of APU Operating State change relationships**

Case	Description
A	Online Operation after a Standard to Advanced Protocol change
	An STFMT Command causes a <i>BPA</i> to <i>CPA</i> state transition. A CP-END Directive within a CP causes a <i>CPA</i> to <i>SPA</i> state change, as the APU will understand all aspects of the Advanced Protocol that will be used when the CP is exited.
B	Offline Operation after a Standard to Advanced Protocol change
	An STFMT Command causes a <i>BPA</i> to <i>CPA</i> state transition. A CP-END Directive within a CP causes the TAP.7 Controller to be placed Offline with the state remaining <i>CPA</i> .
C	Advanced Protocol—Online Operation following commands
	A command causes an <i>SPA</i> to <i>CPA</i> state change. A CP-END Directive within a CP causes a <i>CPA</i> to <i>SPA</i> state change, as the APU will understand all aspects of the Advanced Protocol that will be used when the CP is exited.
D	Advanced Protocol—Offline Operation following a command, re-synchronization.
	A command causes an <i>SPA</i> to <i>CPA</i> state change. A CP-END Directive within a CP causes the TAP.7 Controller to be placed Offline with the state remaining <i>CPA</i> . Subsequently a selection event qualification sequence is detected and the TAP.7 Controller is resynchronized with the DTS.
E	Advanced Protocol—Offline Operation following a command, no re-synchronization.
	A command causes an <i>SPA</i> to <i>CPA</i> state change. A CP-END Directive within a CP causes the TAP.7 Controller to be placed Offline with the state remaining <i>CPA</i> . A selection event qualification sequence is not detected and the TAP.7 Controller remains Offline.
F	Offline Operation following a TAP.7 Controller reset, synchronization.
	The TAP.7 Controller is placed Offline by a controller reset state. The selection event qualification sequence is detected with subsequent synchronization of DTS and TAP.7 Controller operation.

## 21.7 TAPC state/packet relationships

### 21.7.1 TAPC state and packet sequence relationships

The relationship of the Standard, Control, and Advanced Protocols and the SP, CP, and TP packets to the TAPC state is shown in Figure 21-9.



**Figure 21-9 — Typical DTS-initiated packet sequences and APU state transitions**

The packets with black outlines occur whereas packets with gray outlines occur sometimes or do not occur at all (more on this shortly). The transitions between the use of the Standard, Control, and Advanced Protocols are shown. Recall that an SP advances the TAPC state unless it is followed by a CP.

A *BPA* to *CPA* state transition occurs at point A in the figure when an STFMT Command specifies the use a scan format other than a JScan Scan Format while using the Standard Protocol. A CP follows the TMS value associated with the *Update-DR* TAPC state of Command Part Two of this STFMT Command. In this case the CP does not detect a Configuration Fault with the TAP.7 Controller remaining Online (a Configuration Fault would be detected if an unsupported feature is enabled by the STFMT command or a command preceding it). Subsequent to the completion of the CP, an SP occurs for each TAPC state. A TP follows an SP when transport is enabled for the TAPC state associated with the SP. This repeats for each TAPC state.

A second command occurs at some later point in time at point B in the figure. This command causes an SP/CP packet combination following the packets (either an SP or an SP/TP packet pair) associated with the *Update-DR* TAPC state of Command Part Two of this command. In this case, the CP does not detect a Configuration Fault with the TAP.7 Controller remaining Online. The SP of this SP/CP packet combination is associated with either the *Run-Test/Idle* or *Select-DR-Scan* TAPC state. It ensures the information within the SP preceding it has been utilized so any change in scan format as a result of the command begins with all packets completely processed. The TAPC state is not advanced by the SP of the SP/CP combination,

with the TAPC state being either the *Run-Test/Idle* or *Select-DR-Scan* after completion of the CP. The use of the Advanced Protocol continues as the command does not specify the use of the Standard Protocol.

A third command (an STFMT Command) occurs at point C in the figure. An SP/CP packet combination follows an SP or SP/TP packet pair. The SCNFMT Register specifies the use of the Standard Protocol prior to the last CP. The use of the Standard Protocol begins following the completion of the last CP. Again the CP does not detect a Configuration Fault.

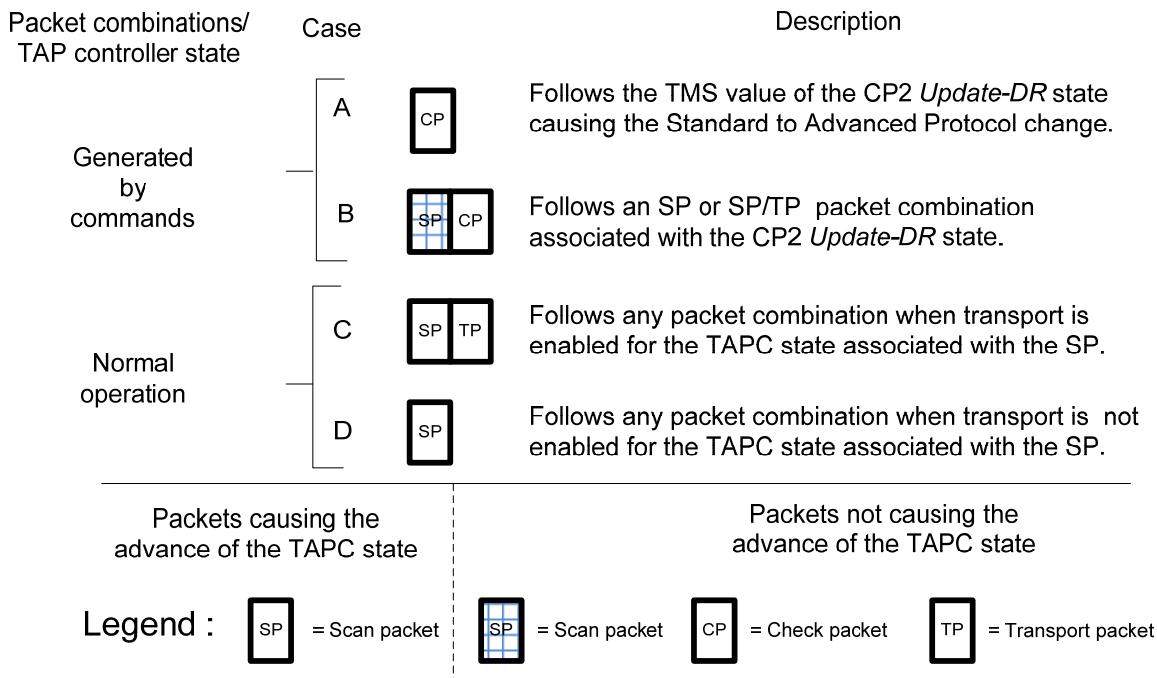
### 21.7.2 Constructing packet sequences

The packet sequences shown in Figure 21-9 and all other packet sequences can be constructed from the four combinations of CP, SP, and TP packets described in Table 21-5 and shown in Figure 21-10.

**Table 21-5 — Description of packet combinations**

Case	Packets	Description
A	CP	A CP follows the TMS value for the CP2 <i>Update-DR</i> TAPC state conveyed with the Standard Protocol. A CP also follows the last bit of a Short-Form or Long-Form Selection Sequence.
B	SP, CP	An SP/CP packet combination associated with the <i>Run-Test/Idle</i> or <i>Select-DR-Scan</i> state Command Part Two <i>Update-DR</i> TAPC state. The SP does not advance the TAPC state.
C	SP, TP	An SP/TP packet combination follows any packet combination provided the Transport Function is enabled for the TAPC state associated with the SP. The SP advances the TAPC state.
D	SP	An SP packet follows a CP, an SP, or an SP/TP combination where either transport is not enabled for the TAPC state associated with the SP or transport is never permitted for the state. The SP advances the TAPC state.

The use of the Advanced Protocol begins with combination A and ends with combination B. Scan Packet combinations B–D may be used between A and B.



**Figure 21-10 — Packet combinations used to advance the TAPC state**

### 21.7.3 Packet combinations/TAPC state relationships

The relationship of the packet combinations shown in Figure 21-10 and TAPC states is shown in Table 21-6. This table is to be interpreted as follows:

- The same TAPC state name may have different TAPC combinations associated with it.
- A packet combination associated with a TAPC state cannot contain both a CP and TP.
- Only five TAPC states (*Run-Test/Idle*, *Update-xR*, and *Pause-xR*) support the Transport Function. Transport can be individually enabled or disabled for each of these states.
- An SP/CP combination can only be associated with either the *Run-Test/Idle* or the *Select-DR-Scan* TAPC state when all the following are true:
  - Following an SP or SP/TP combination associated with the *Update-DR* TAPC state of Command Part Two.
  - When the Advanced Protocol is already being used.
- A CP-only packet combination can only be associated with the *Run-Test/Idle* and *Select-DR-Scan* TAPC states when the following is true:
  - When the Standard Protocol is being used and the STFMT Command initiates the use of the Advanced Protocol.

### 21.7.4 Scheduling of packets

The first bit of an SP, where the DTS sources this bit, is called the “queuing bit-period.” A decision can be made as to the SP content and whether a CP or TP follows the SP during this bit period. The qualification of Selection and Deselection Escapes also occurs during the queuing bit period. When either of these Escapes is qualified, the following occurs:

- The SP is cancelled after one bit by creating no subsequent SP content.
- The queuing of a CP or TP that would normally follow the SP is inhibited.
- An advance of the TAPC state that would normally be generated by the SP is cancelled.
- The first bit of the SP is followed by:
  - An *SPA* to *CPA* state change.
  - One of the following:
    - A Selection Sequence.
    - Placement of the TAP.7 Controller Offline.

In the absence of a qualified Selection or Deselection Escape during the queuing bit period, the following occurs:

- The SP content is determined.
- The queuing of a CP or TP that follows the SP occurs.
- The SP is allowed to advance the TAPC state, provided a CP will not follow the SP.

**Table 21-6 — Packet and TAPC state relationships**

TAPC state	Case	TAPC state advance	Packet scheduling		
			Time →		
			Scan (SP)	Check (CP)	Transport (TP)
<i>Test-Logic-Reset</i>	D	Yes	Yes	—	—
<i>Run-Test/Idle</i>	A	—	—	Yes	—
	B	—	Yes	Yes	—
	C	Yes	Yes	.	Yes
	D	Yes	Yes	—	—
<i>Select-IR-Scan</i>	D	Yes	Yes	—	—
<i>Capture-IR</i>	D	Yes	Yes	—	—
<i>Shift-IR</i>	D	Yes	Yes	—	—
<i>Exit1-IR</i>	D	Yes	Yes	—	—
<i>Pause-IR</i>	C	Yes	Yes	—	Yes
	D	Yes	Yes	—	—
<i>Exit2-IR</i>	D	Yes	Yes	—	—
<i>Update-IR</i>	C	Yes	Yes	—	Yes
	D	Yes	Yes	—	—
<i>Select-DR-Scan</i>	A	—	—	Yes	—
	B	—	Yes	Yes	—
	D	Yes	Yes	—	—
<i>Capture-DR</i>	D	Yes	Yes	—	—
<i>Shift-DR</i>	D	Yes	Yes	—	—
<i>Exit1-DR</i>	D	Yes	Yes	—	—
<i>Pause-DR</i>	C	Yes	Yes	—	Yes
	D	Yes	Yes	—	—
<i>Exit2-DR</i>	D	Yes	Yes	—	—
<i>Update-DR</i>	C	Yes	Yes	—	Yes
	D	Yes	Yes	—	—

## 21.8 User's and implementer's views of the Advanced Protocol

The implementer of a TAP.7 Controller will have detailed knowledge of the operation of the Advanced Protocol. A user of a TAP.7 does not need to understand the implementer's perspective of the TAP.7 and requires little knowledge of its operation. Two views are described in this document, a user's view that describes the packet sequences related to TAPC states and an implementer's view that details the point where the ADTAPC and CLTAPC states change. These two views are described briefly in the following subclauses.

### 21.8.1 User's view

A user of the Advanced Protocol should view the packet combinations described in Table 21-6 as managing a virtual TAPC State Machine attached to the TAP.7 pins. The effects of each SP occur when the SP completes. The real ADTAPC and CLTAPC states may change state before or after the virtual state

machine, but this is of no consequence to the DTS. The state of the virtual TAPC is called the virtual TAPC State or “VState.” The SP and SP/TP combinations advance the VState while the CP and SP/CP combinations do not advance the VState. When the VState changes, by definition the state change occurs with the TCKC signal rising edge within the last bit of the last packet associated with the TAPC state. This model is sufficient for using all TAP.7 functionality.

### 21.8.2 Implementer's view

An implementer of a T4 and above TAP.7 should view the packet combinations described in Table 21-6 and shown in Figure 21-9 as managing the real ADTAPC state. This view understands the exact timing of the real TAPC state changes. The relationship of SPs and the advance of the TAPC state are described in 22.3.

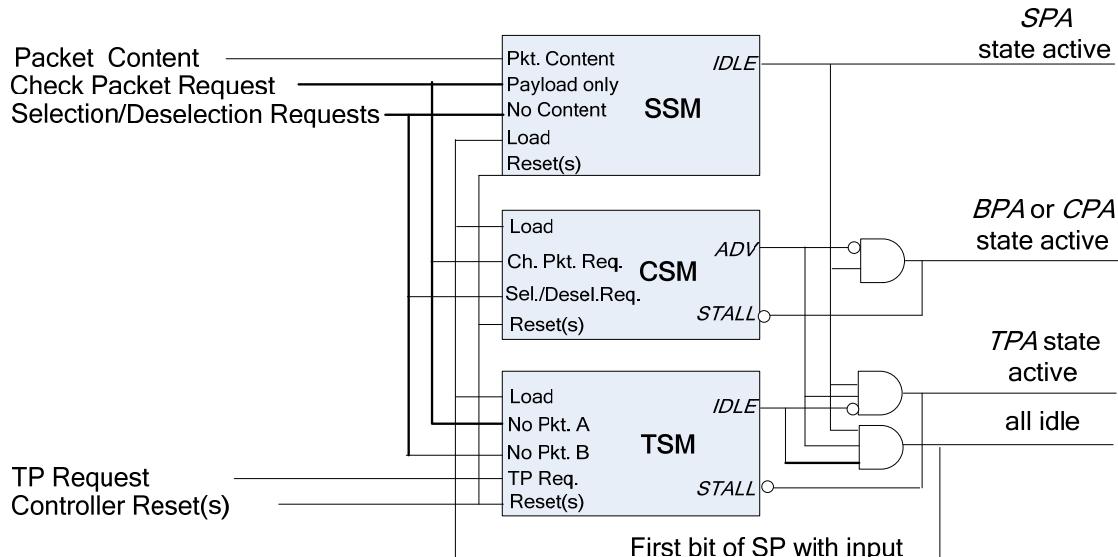
### 21.9 An approach to implementing APU Operating State scheduling

One approach to the scheduling of APU Operating States is shown in Figure 21-11. This description emphasizes the scheduling of the *BPA*, *CPA*, *SPA*, and *TPA* state's control of the TAP pins. It does not cover other aspects of the APU's operation. Approaches other than the one describe below may also be used.

This approach uses the following APU Operating State mapping:

- BPA/CPA*      The Control State Machine (CSM) handles the Control and Bypass Functions.
- SPA*              The Scan State Machine (SSM) handles the Scan Function.
- TPA*              The Transport State Machine (TSM) handles the Transport Function.

Each machine listed above may be constructed with a number of smaller machines. Other miscellaneous machines, flags, and functions support the operation of these machines (i.e., Escape detection, serialization and de-serialization logic, and logic managing when the TAPC is advanced, etc.).



**Figure 21-11 — Conceptual view of packet scheduling**

A conceptual view of the CSM is shown in Figure 11-28. The conceptual view of the SSM is described incrementally in Clause 24 through Clause 26 as the MScan, OScan, and SScan Scan Formats are

described. This machine is constructed with multiple submachines and helper flags. The conceptual view of the TSM is described in 28.3.

The conceptual view of packet scheduling shown in Figure 21-11 divides the functionality of the SSM between these machines to match closely the APU functions described in Clause 22. The partitioning also considers the incremental addition of optional features. This example shows a series of building blocks that complement each other. However, these building blocks alone do not provide a complete implementation. Collectively, this partitioning provides an architectural approach to implementing a T4 and above TAP.7 Controller. Since these examples illustrate architectural concepts, any implementation based on them should be thoroughly tested to ensure the implementation provides the behavior specified by this standard. Other approaches may also be utilized.

With this approach, the following occurs:

- The *SPA* state is active when either of the following occur:
  - The CSM state is *Idle* and the TSM state is *Idle*.
  - The SSM state is not its *Idle* state.
- A *TPA* state is active when the SSM state is *Idle* and the TSM state is not *Idle*.
- A *CPA* state is active when the SSM state is *Idle* and the CSM state is not *Idle*.
- The queuing bit-period is active when the SSM, CSM, and TSM states are all *Idle*.

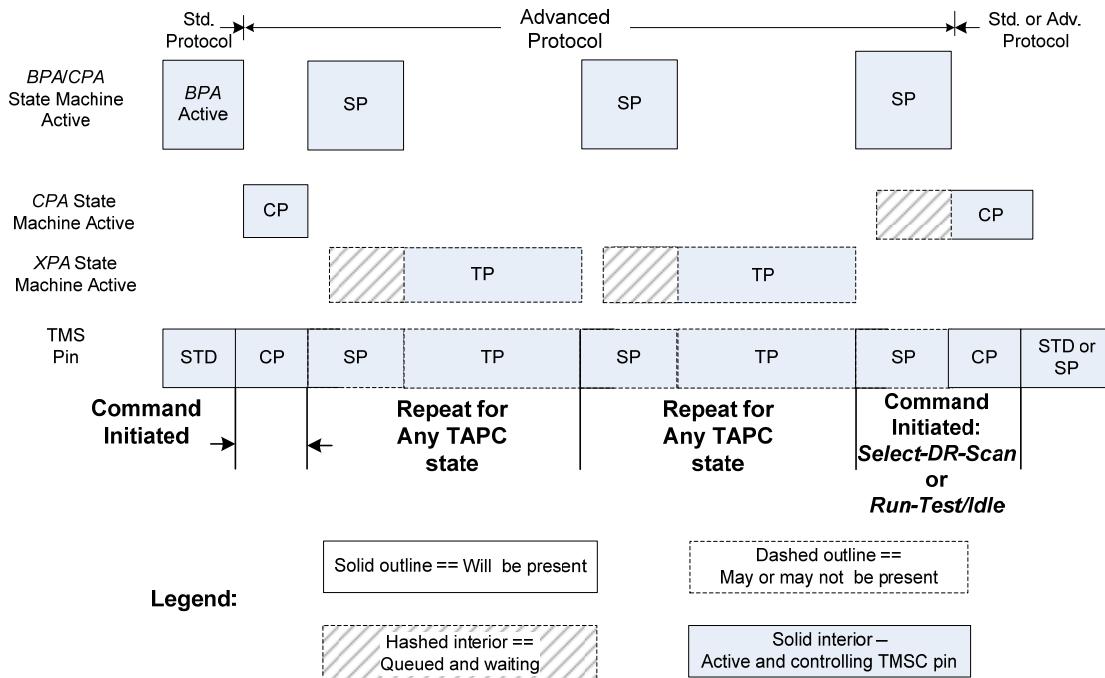
A Type-0–Type-3 Reset activates the CSM State Machine as defined by the start-up option and creates the Scan and Transport State Machine Idle states. Recall that the CSM state is initialized to a state other than its *Idle* state (a state other than *ADV*). This creates either the *BPA* or *CPA* Operating State, inhibiting the Advanced Protocol processing provided by the *SPA* and *TPA* states. The CSM state remains *STD* when the Standard Protocol is used. This blocks the use of the Advanced Protocol as the SSM and TSM states are *Idle* until the CSM state also reaches *Idle*.

Subsequent to the completion of the CP created by the transition from the use of the Standard Protocol to the use of the Advanced Protocol, every APU Operating State that will be visited for a specific TAPC state is determined during the queuing bit period. All packets associated with a TAPC state are queued at the end of the queuing bit period as shown in Figure 21-12. For SPs with only an output bit-frame, this determination is made during the last bit period of the SP (for SScan output-only Data Segments). The operation of these state machines states are interlocked to sequence in the proper order thereafter. In this case, only SPs may be queued. Once queued, the SP completes first followed by a CP or TP. A queued CP or TP waits until the SP completes before it begins. In the case where neither a CP nor a TP is queued, the CSM and TSM remain in their *Idle* states. A TP cannot be queued when a CP is queued. When a qualified Selection or Deselection Escape occurs, the CSM activates with the SSM and TSM moving to their *Idle* states. At this point, either the Selection Sequence is active or the TAP.7 Controller is Offline.

The CSM, SSM, and TSM are in their respective idle states during the queuing bit period. Since the APU Operating State sequence is known for any TAPC state (*SPA* followed by a *CPA*, a *TPA*, or neither the *TPA* nor the *CPA* states), the state machines creating this sequence are easily queued at this time. Since the operation of these machines is interlocked, they leave their *Idle* state and wait until their turn at managing the TMSC signal. Once all machines reach their *Idle* state, the process begins anew with a new queuing bit-period. The DTS may schedule packets with a similar scheduling approach.

Note the similarity of the content of Figure 21-9 and Figure 21-12. The notable difference is the packets in Figure 21-9 are initiated by the completion of the packets scheduled before them, whereas the Packets in Figure 21-12 are queued and await the completion of packets that precede them. The interlocking logic in Figure 21-11 (the state active signals) imposes the scheduling model shown in Figure 21-9 on the model in Figure 21-12.

## APU Operating State Machine Activity



**Figure 21-12 — Example of scheduling and sequencing of state machine activity**

### 21.10 Structure of the clauses describing T4 and above TAP.7s

The operation of the *SPA* and *TPA* states and the SSM and TSM supporting them are described in subsequent clauses and subclauses. Figure 21-13 depicts the structure of the clauses related to the T4 and T5 TAP.7s. The material conveyed in this clause and Clause 22 provides a description of the APU infrastructure.

Clauses 23 and 27 describe the T4 and T5 TAP.7 capabilities and configurations. These clauses begin with a description of the registers and commands used in each class. The mandatory and optional features are identified for the class. An explanation of the capability provided by the class follows. Clauses 24 through 26 and Clause 28 provide a description of the operation of the *SPA* and *TPA* operating states. This description includes conceptual state diagrams and flowcharts. A description of the signal activity related to the operation of the *SPA* and *TPA* states, and use cases, are included.

The specific subject matter covered by each of these clauses is listed as follows:

- Clause 22      APU Scan Packets
- Clause 23      T4 TAP.7
- Clause 24      MScan Scan Format
- Clause 25      OScan Scan Formats
- Clause 26      SScan Scan Formats
- Clause 27      T5 TAP.7
- Clause 28      *TPA* state operation

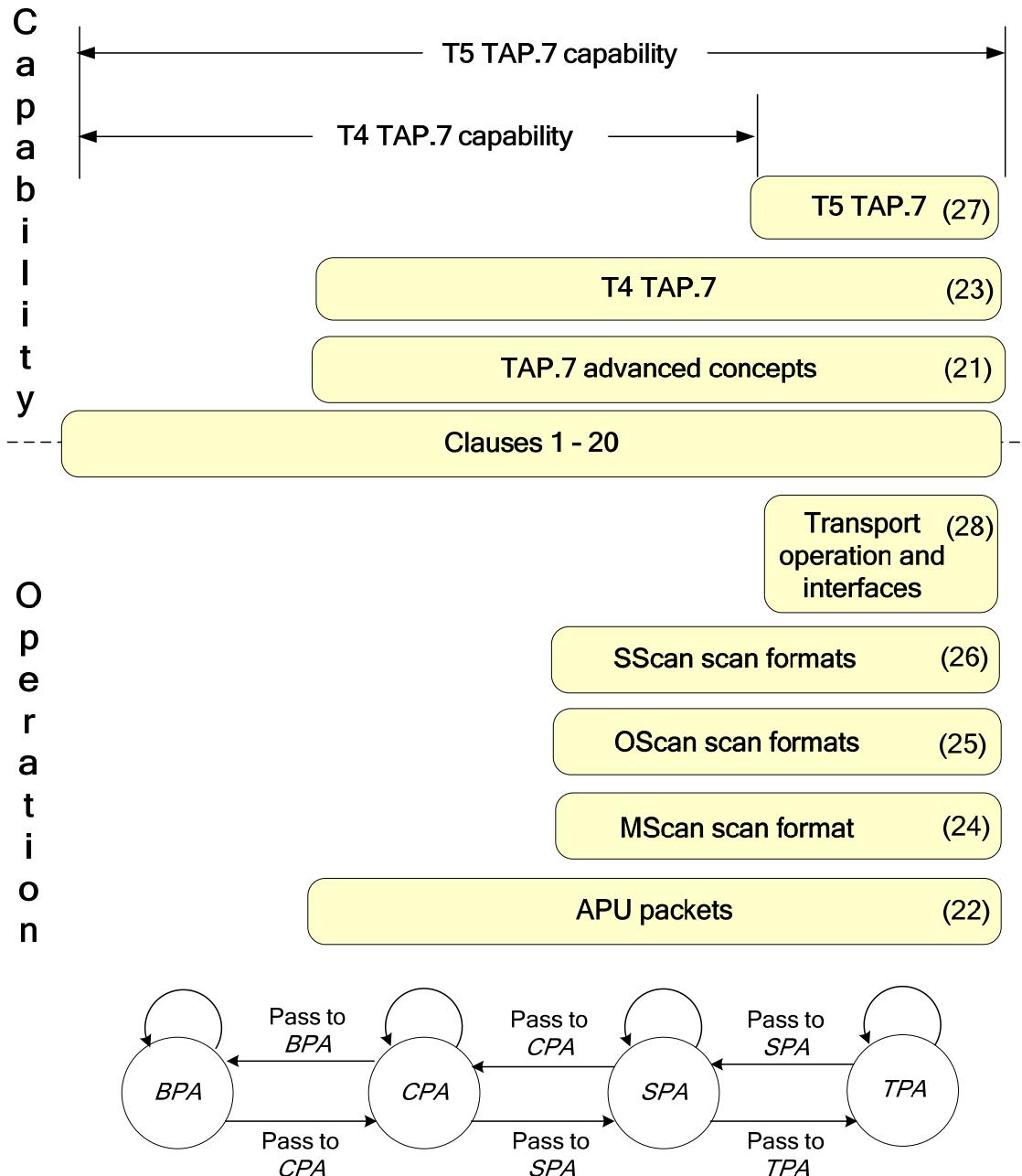


Figure 21-13 — Advanced operation function/description relationships

## 22. APU Scan Packets

This clause is applicable to T4 and above TAP.7s. It provides a high-level overview of the Check, Scan, and Transport Packets and the functionality they deliver. The detailed description and rules governing the operation of these packets is deferred to subsequent clauses. The APU state diagram is also included.

The subject matter within this clause is described in the following order:

- 22.1 CPs
- 22.2 SPs
- 22.3 SPs that advance the TAPC state
- 22.4 TPs
- 22.5 APU state diagram
- 22.6 An approach to implementing packet scheduling

### 22.1 CPs

Check Packets are described in 11.9.6. The entry into the *CP\_PRE* substate from the *STD* state shown in Figure 11-19 occurs when the SCNFMT Register is stored with a value specifying a scan format other than a JScan Scan Format. The entry into the *CP\_PRE* substate from the *ADV* state shown in this figure occurs following Command Part Two of both two- and three-part commands. The state transitions correspond to the *STD* to *CHK* and *ADV* to *CHK* CSM state transitions shown in Figure 11-17.

When a CP follows an SP, the CP provides a period of TMSC signal activity permitting the following:

- The DTS and TAP.7 may adjust their behavior based on recently changed register values.
- The DTS is provided one TCKC signal period (the Preamble and the Postamble bits) to change the operation of the TMSC signal when transitioning between the use of the Standard and Advanced Protocols and vice versa.
- The DTS may extend the length of the CP indefinitely (to suspend the TAP.7 interface while it uses another TAP interface, changes operating modes, or for any other reason).

The format of a CP is shown in Figure 11-25. The DTS may use the Preamble and Postamble Element bit periods to switch the TMSC signal drive sources when a change between the use of the Standard Protocol and Advanced Protocol occurs as described above. Clause 24 provides an additional description of command-initiated CP scheduling while using the Advanced Protocol.

### 22.2 SPs

#### 22.2.1 Conceptual view of an SP

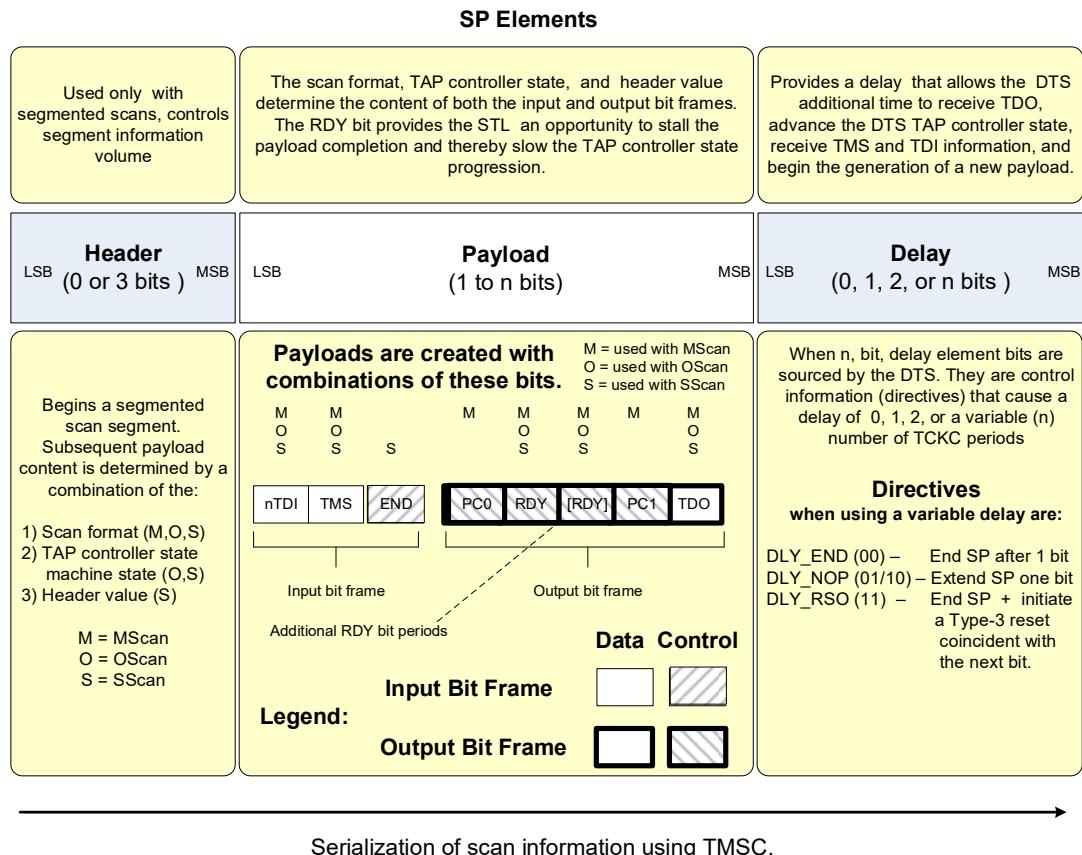
An SP is synonymous with the *SPA* state. The DTS and TAP.7 Controller should view an SP as the following:

- An exchange of the information related to a single TAPC state (TDI, TMS, and TDO or some subset thereof) that meets the needs of both the DTS and TS
- Providing the CLTAP an opportunity (if needed) to stall the completion of the SP and delay the advance of the TAPC state

- Providing the DTS an opportunity (if needed) to stall the completion of the SP after the advance of the TAPC state
  - Causing an advance of the TAPC state when it completes provided it is not followed by a CP
  - Cancelled after the first bit by a qualified Selection or Deselection Escape

## 22.2.2 SP format

An SP serially exchanges TDI, TMS, and TDO signal information or a subset thereof between the DTS and TAP.7 Controller. Other control information may be added to this exchange. A conceptual view of an SP is shown in Figure 22-1.



**Figure 22-1 — Conceptual view of a Scan Packet**

The SP Elements are described as follows:

- Header Control information that precedes the Payload Element of the first SP of a Data Segment of SScan Scan Formats. When present, a header specifies the content of subsequent Payload Elements within the SPs of the segment.
  - Payload Data + control information. The Payload Element content is varied to trade off performance and flexibility. Payload information sent to the TAPC Controller forms an input bit-frame. Payload information sent to the TAPC Controller forms an output bit-frame. Scan formats provide to the STL a means to stall progression of the TAPC state include RDY bit(s) in the output bit-frame.

Delay      Optional control information providing separation in time of adjacent payloads. A Delay Element supports simple DTS implementations where a Delay Element is used to stall the progression of the TAPC state while it develops a new SP following the completion of the prior payload.

The Header Element, Delay Element, and control information within the Payload Element are consumed by the TAP.7 Controller and are not visible to either the ADTAPC or the CLTAPC.

### 22.2.3 SP content

#### 22.2.3.1 Header Element content

Recall that a Header Element is used only with SScan Scan Formats. These scan formats divide a scan transfer into the following elements:

- Control Segments associated with consecutive states other than *Shift-xR* states
- Data Segments associated with consecutive *Shift-xR* states

A header defines the content of SP payloads within a segment together with the TAPC state. The SP preceding a CP never contains a Header Packet.

#### 22.2.3.2 Payload Element content

Within this document, the SP Payload Element is described as constructed from an input bit-frame, an output bit-frame, or both. Payload Element content sourced by the DTS is considered to be in an input bit-frame while the Payload Element content sourced by the TAP.7 Controller is considered to be in an output bit-frame.

Input bit-frames may contain the following:

- A TMS bit (a true TMS value)
- A Not Test Data Input (nTDI) bit (an inverted TDI value)
- An END bit:
  - Indicating the end of a Data Segment with SScan0–SScan1 Scan Formats
  - Indicating the end of a Control Segment with SScan0–SScan1 Scan Formats/Header Element combinations allowing multiple consecutive Control Segments
- An EOT Escape overlaid on any of the following:
  - nTDI bits to cause an exit from the *Shift-xR* state with OScan4–OScan7 Scan Formats
  - nTDI bits to cause the end of a Data Segment with SScan2–SScan3 Scan Formats
  - TMS bits to cause the end of a Control Segment with SScan2–SScan3 Scan Formats/Header Element combinations specifying multiple consecutive Control Segments

Output bit-frames may contain the following:

- A TDO bit
- PC0 and PC1 precharge bits with the MScan Scan Format
- RDY bit(s) providing a rate-dependent component within the STL the opportunity to stall the completion of the SP payload (with MScan, some OScan, and all SScan Scan Formats)
- EOT Escape overlaid on any of the following:

- An RDY bit
- The TDO bit to cause an end to the Data Segment with SScan2–SScan3 Scan Formats

NOTE—See Annexes G and H for information regarding forms of rate-dependent component.

When an input bit-frame arrives at the TAP.7 Controller, the TDI and TMS signal information within it is deserialized and presented in parallel to the ADTAPC and CLTAPC. The nTDI bit was chosen to be the inverted value of the TDI value as this aids in the detection of a broken DTS/TS connection when the TS sources the TCKC signal. The output bit-frame begins when the input bit-frame is completed or is the first information in the SP payload when there is no input frame. The payload is completed when the CLTAPC and the Embedded TAPs are ready to utilize the information with the input bit-frame, and the data needed to create the TDO bit of an output bit-frame is available. This process is repeated for each SP.

When an SP precedes a CP, the payload content (the bits in the SP and not the bit values) of the SP is the same as the payload content of the SP following:

- The preceding SP in the case of MScan and OScan Scan Formats
- The last SP containing a Header Element in the case of the SScan Scan Format

### 22.2.3.3 Delay Element content

The function of a Delay Element is the same for all scan formats. A Delay Element may be zero, one, two, or n bits in length. One- and two-bit Delay Elements are fixed length. Delay Elements n bits in length are called a variable-length delay. The value of the DLYC Register value specifies the minimum separation between SP payloads as the following:

- 00 None
- 01 One TCKC signal period
- 10 Two TCKC signal periods
- 11 A variable number of three or greater TCKC signal periods

The TAP.7 Controller does not drive the TMSC signal during Delay Element bit periods. Although the DTS may drive the TMSC signal during Delay Element bit periods of a fixed delay, it is expected that the DTS does not drive the TMSC signal during these bit periods as it may use this delay period to input the last TDO value. When this is the case, the TMSC signal is kept at the last value driven by either the DTS or TS. With a variable-length delay, the DTS drives the TMSC signal during all Delay Element bit periods except the last, and may drive the TMSC signal during the last bit period of the Delay Element. The string of Delay Element bits in a variable-length Delay Element is a series of directives that are virtually the same as those used by a Check Packet:

- DLY\_NOP 01 or 10 Extends the Delay Element by one bit period
- DLY\_END 00 Causes completion of the Delay Element
- DLY\_RSO 11 Initiates a Type-3 Reset

Examples of Delay Elements are shown in Table 22-1. A postamble bit follows both a DLY\_END and DLY\_RSO Directive. It is recommended the DTS supplies a logic 0 delay postamble bit following a DL-END Directive and a logic 1 delay postamble bit following a DL\_RSO Directive.

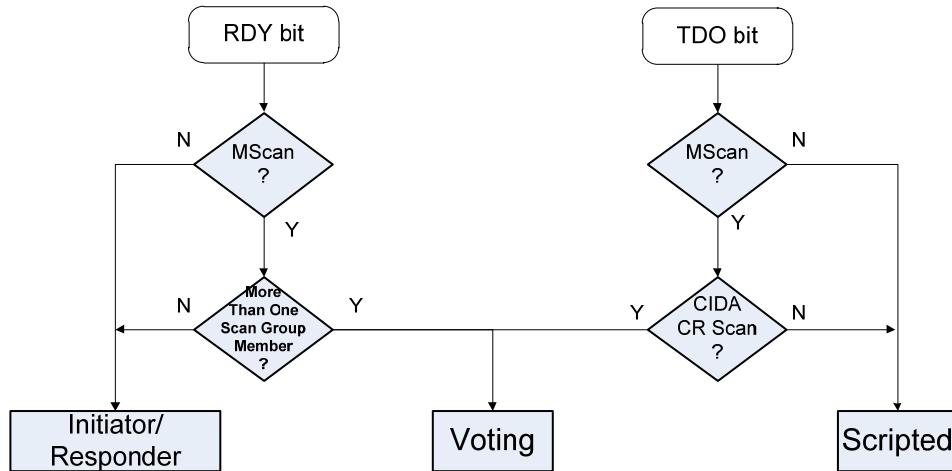
**Table 22-1 — Variable-length delay examples**

Delay length in bits	Number of Directives	Directive terminating Delay Element	
		DLY_END	DLY_RSO
3	1	00x	11x
4	2	100x	011x
5	3	0100x	1011x
6	4	10100x	01011x

The SP preceding a CP never contains a Delay Element. An expanded description of a Delay Element is located in 24.5.

#### 22.2.4 Types of output bit-frame transactions

The MScan, OScan, and SScan Scan Formats create three types of scan transactions within output bit-frames (Voting, Initiator/Responder, and Scripted) as shown in Figure 22-2. The TMSC Signal Drive Policy described in Clause 14 creates these types of exchanges. Voting, Initiator/Responder, and Scripted scan transactions are described briefly in the subsequent paragraphs.



**Figure 22-2 — Scan exchange types**

The MScan Scan Format creates voting transactions. Voting on a bit value occurs with the MScan Scan Format in the following cases:

- RDY bits when the CLTAPC of more than one TAP.7 Controller is selected
- TDO bits during the CR Scan of a CIDA Command

All TAP.7 Controllers input the TMSC value created by the vote. The SP payload completes following RDY bit voting producing a logic 1. TDO bit voting creates the AT bit value used with Undirected CID Allocation.



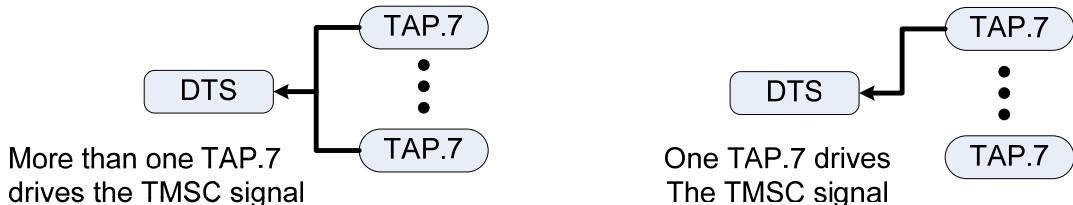
**Figure 22-3 — Voting bit transaction**

An Initiator/Responder bit transaction is shown in Figure 22-4. This transaction type is used with RDY bits created with SScan and OScan Scan Formats. A single TAPC drives the TMSC pin to either a logic 0 or a logic 1 for the bit period. This controller is the transaction initiator(s). The remaining TAP.7 Controllers are responders to the initiator. Responders input the value of the bit and utilize its value. The SP payload completes with a TDO or ONE bit following a number of logic 1 RDY bits equal to the RDYC Register value plus one. With OScan and SScan Scan Formats, once a logic 1 RDY bit is generated, it will remain a logic 1 until the number of RDY bits equals the RDYC Register value plus one. Recall that only one initiator is permitted with scan formats that include an RDY bit. The TAP.7 Controller is placed Offline otherwise, as this constitutes a Configuration Fault. A initiator is equivalent to TAP.7 Controller which STL is a Scan Group Member when an OScan or SScan Scan Format with RDY bits is used.



**Figure 22-4 — Initiator/Responder bit transaction**

An SP without an RDY bit within the Payload Element creates a Scripted Transfer as shown in Figure 22-5. The payload completes in the same number of TCKC periods as there are bits in the payload.



**Figure 22-5 — Scripted scan exchange**

### 22.3 SPs that advance the TAPC state

The implementer should understand the precise relationship of the Advanced Protocol and TAPC state changes. The TAPC state changes are affixed to the SP payload in the manner outlined below to ensure the consistent operation of the test capability of IEEE Std 1149.1. The TAPC state advance is initiated by an SP payload where the SP is not followed by a CP or cancelled by a Selection or Deselection Escape. From the TAP.7 designer's perspective, the point at which the TAPC state changes as a result of an MScan, OScan, or an SScan SP is described by the following cases:

- A) Case: The scan format is either MScan or OScan and the SP contains a TDO bit:

- Coincident with the TDO bit.
- B) Case: The scan format is either OScan or SScan and the SP does not contain a TDO bit:
  - Coincident with the bit period following the last bit of the SP payload.
- C) Case: The scan format is SScan, the SP is associated with TAPC state other than *Shift-xR*, and the SP contains a TDO bit:
  - Coincident with the TDO bit.
- D) Case: The scan format is an SScan Scan Format, the SP is associated with TAPC state other than *Shift-xR*, and the SP does not contain a TDO bit:
  - Coincident with the bit period following the last bit of the SP payload.
- E) Case: The scan format is SScan, the SP is associated with the *Shift-xR*, state and the SP contains a TDO bit:
  - 1) Subcase: The SP is the last SP of the Data Segment.
    - An advance occurs coincidentally with the HDR[2] bit of the next SP.
  - 2) Subcase: The SP contains only an output bit-frame and is the next to last SP in the Data Segment.
    - No advance is generated.
  - 3) Subcase: The SP is not the last SP of the Data Segment and contains an input bit-frame and an output bit-frame.
    - Coincident with the TDO bit in the SP.
  - 4) Subcase: The SP is not the last SP of the Data Segment and contains an input frame and no TDO bit.
    - Coincident with the bit following the last bit of the SP payload.

Scan format specific descriptions pertaining to advancing the TAPC state are found in Clauses 24 -26. The information above is expanded with additional description, and rules.

## 22.4 TPs

### 22.4.1 Conceptual view of a TP

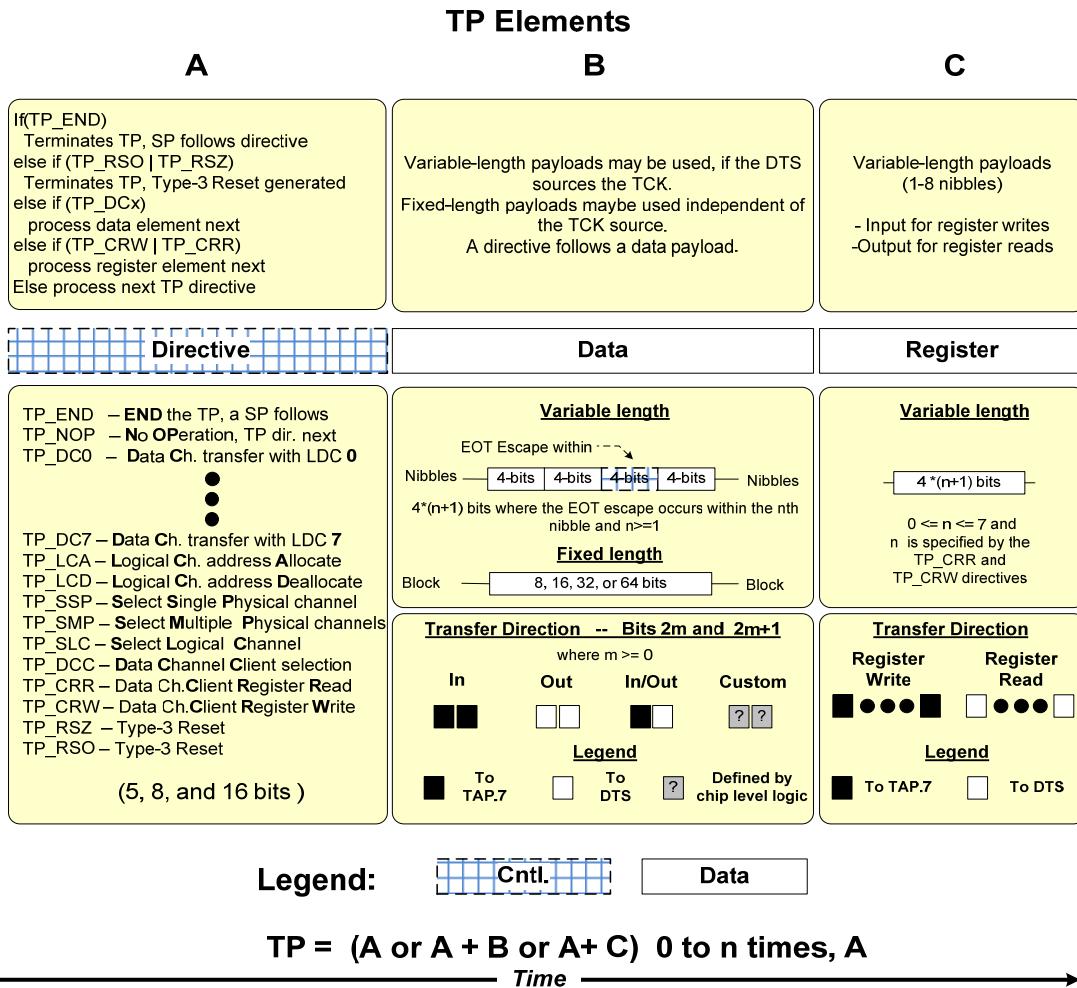
A TP defines the period of TMSC signal activity where the DTS may exchange data with a chip-level hardware component. A TP is not visible to TAPCs. The TP control information is not visible to the DCCs. The DTS should view a TP as the follows:

- Following SPs associated with TAPC states designated as supporting transport via the TPST Register, provided a CP is not needed
- Providing communication with data transfers to one or more Chip-Level DCCs
- Providing a means to write certain Transport Function Control Registers
- Providing a means to read and write Data Channel Client Control Registers

### 22.4.2 TP format

A TP is constructed with the three TP Element types (directive, data, and register) shown in Figure 22-6. Data and Register Elements are preceded by a Directive Element. It is constructed from a series of the following in any mix or sequence followed by a Directive Element terminating the TP:

- A Directive Element
- A Directive Element followed by a Data Element
- A Directive Element followed by a Register Element



NOTE—See 10.4 for a description of an EOT (custom) Escape.

**Figure 22-6 — Conceptual view of a TP**

TP Directives are consumed by the APU. The data within TP register and Data Elements is consumed-produced by the Data Channel Client. The connection to the Data Channel Clients may be through the TAP.7 Controller (implying pipelining of the transfer) or directly between the TMSC signal and Data Channel Client.

When a TP follows an SP, the control of the TMSC signal is transferred to the APU's Transport Control Function. Once this function controls the TMSC signal, the DTS uses the TP Directives to:

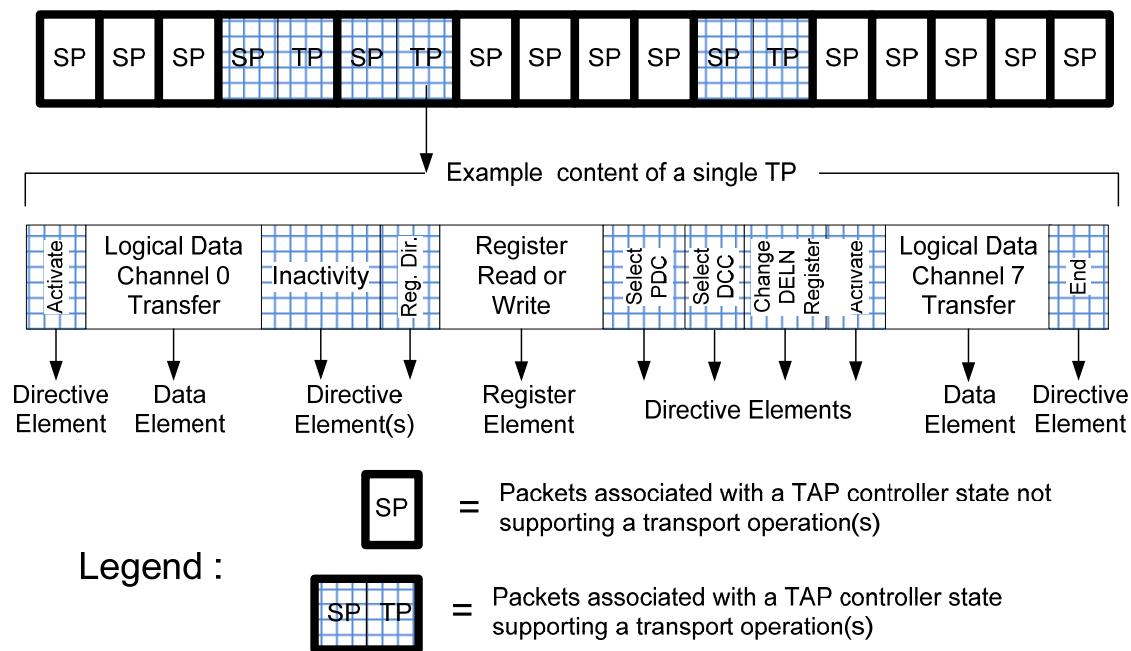
- Write Transport Function Register values
- Read and write DCC Register values

- Initiate data exchanges with the Channel Clients using a Logical Data Channel (Data Channel Clients connected to the Physical Data Channels associated with the Logical Data Channel)

The TP Elements are described further in Clause 27.

### 22.4.3 Content

A TP contains a minimum of one TP Directive. It may have many Directive Elements, Register Elements, and Data Elements. An example of the content of a single TP is shown in Figure 22-7.



**Figure 22-7 — TP placement and content**

### 22.5 APU state diagram

The description of the APU operation is based on the APU State Diagram shown in Figure 22-8. This state diagram portrays the APU's operation as hierarchical groups of states. The detailed operation of the *SPA* and *TPA* states is described in subsequent clauses.

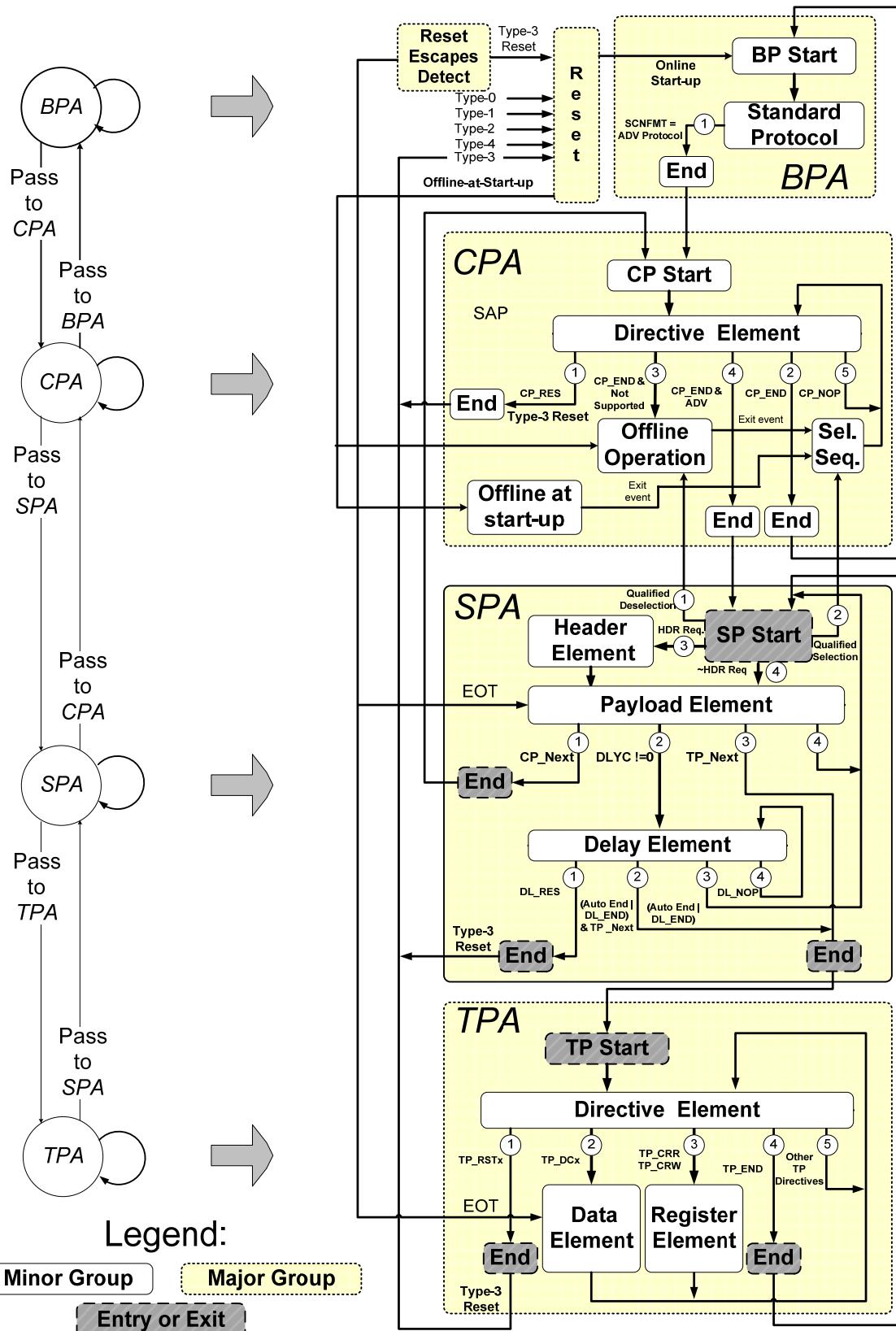


Figure 22-8 — APU state diagram

## 22.6 An approach to implementing packet scheduling

### 22.6.1 Pipelining and its effects

The description of advancing the TAPC state in 22.3 reveals the relationship of the TAPC state changes relative to the last bit of the SP payload. The TAPC state may change either before, when, or after an SP ends.

The term “pipelining” describes the case where the TAPC state changes after the SP ends. It is defined as the transmission of the first bit of a new SP before the previous SP has caused the advance of the TAPC state. The effects of pipelining on the APU implementation can be minimized by using the TAPC state look-ahead logic. This logic is used to create the TAPC state used for determining the SP content and activating a TP when needed. The use of look-ahead logic simplifies the TAP.7 Controller design as it hides the pipelining effects and provides a means to view the remainder of the APU’s operation as if pipelining did not exist. A conceptual view of TAPC state look-ahead logic is shown in Figure 22-9, along with a block diagram of the function used to activate the CSM, SSM, and TSM.

It is recommended the TAP.7 Controller be built using virtual TAPC states to guide the operation of the state machines implementing the protocol-related functions of the *SPA* and *TPA* states. This localizes the point in the implementation dealing with pipelining effects, hiding these effects from the remainder of the implementation. It also provides the TAP.7 Controller a means to identify the payload input and output as it appears on the TMSC signal.

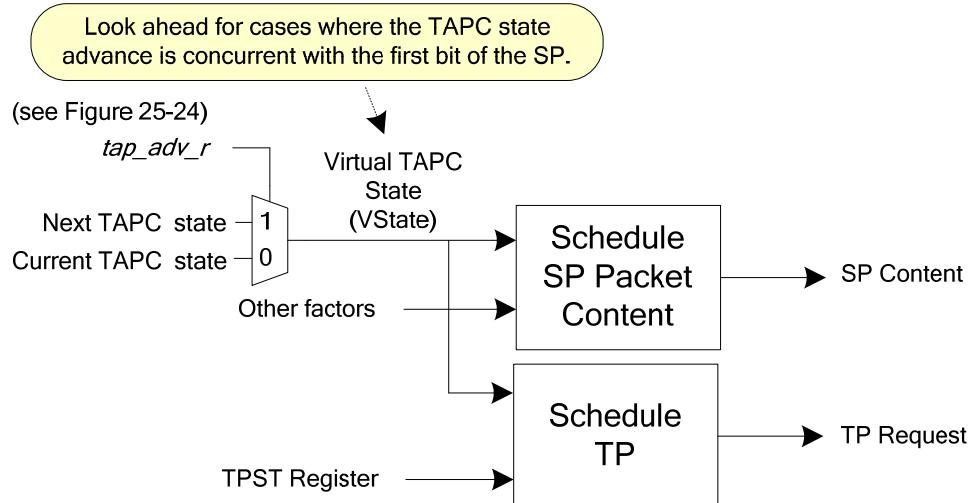


Figure 22-9 — Use of the virtual TAPC state

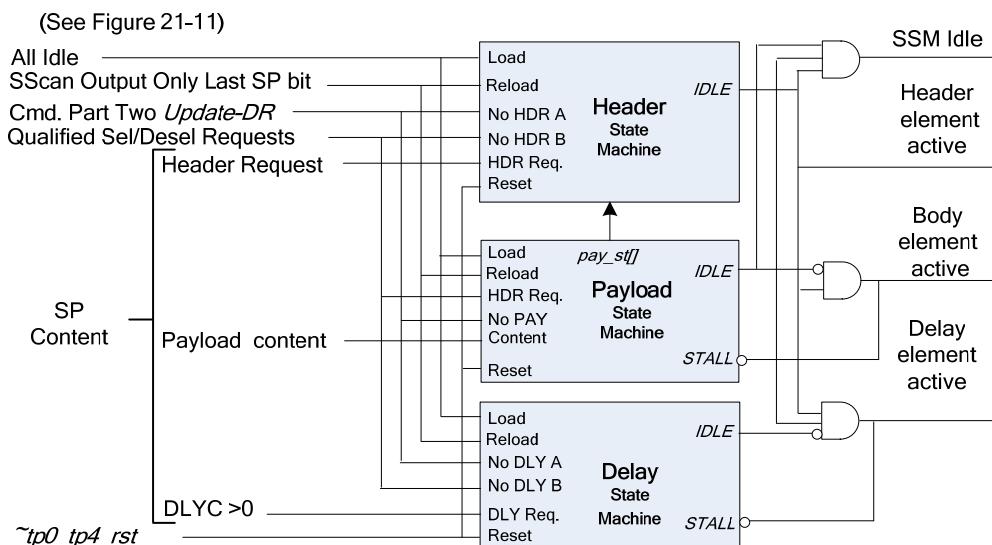
### 22.6.2 SP Element scheduling

Many approaches to SP processing may be used. One conceptual approach is shown in Figure 22-10. With this concept, a number of small state machines implement the SP function. The scheduling mechanism shown in this figure is very similar to the one used to schedule the SSM, CSM, and TSM State Machine activity shown in Figure 21-11. The SSM may be implemented as a number of smaller machines or by consolidating these machines. Examples shown in this standard consolidate the Header and Payload State Machines.

During the queuing bit period, it is possible to determine the entire SP content when the SP does not contain a Header Element. In the case where a Header Element is included, the SP payload content is

determined when the Header Element is complete. This decision is made at the first bit of the SP Payload Element.

In the approach shown, the SSM shown in Figure 21-11 is implemented with three state machines Ready, Payload, and Delay, with the Escape Detection State Machine added when SScan Scan Formats are supported). The Payload State Machine can be made to perform the function of both the Header and Payload State Machines shown in this figure, as is shown in later examples. These state machines may be partitioned into a number of smaller state machines and flags as needed. Other approaches may also be used.



**Figure 22-10 — Conceptual view of a Scan Packet Element scheduling**

These three machines may be combined in any combination as desired. The examples in this specification consolidate the Header and Payload State Machines. When these machines are activated, they run to their *Idle* states in the following order: Header, Payload, and Delay.

### 22.6.3 TP Element scheduling

Many approaches to TP processing may be used. The approach described in 28.9 includes a Transport State Machine that handles the scheduling of the Transport Elements. This machines creates some of the signaling needed for the Physical Data Channels/Data Channel Client interface. The scheduling of the Transport Elements is based on the states supporting transport and the Transport Directives. See 28.9 for additional information on implementing the Transport Function.

## 23. T4 TAP.7

### 23.1 Introduction

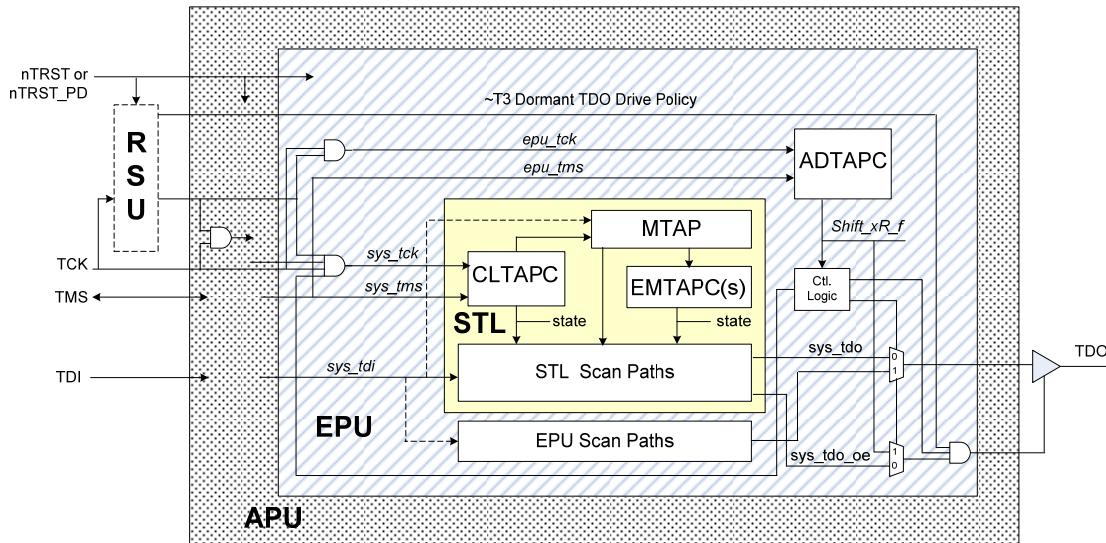
This clause is applicable to T4 and above TAP.7s. It provides the rules, permissions, and recommendations for the implementation of a T4 TAP.7 when combined with Clause 22 through Clause 26. It also provides programming considerations.

The subject matter within this clause is described in the following order:

- 23.2 Deployment
- 23.3 Capabilities
- 23.4 Register and command portfolio
- 23.5 Configurations
- 23.6 Start-up behavior
- 23.7 Scan formats
- 23.8 Configuration Faults
- 23.9 Increasing STL performance
- 23.10 Auxiliary pin function control
- 23.11 Sample using rising edge
- 23.12 System and EPU TMS signal values
- 23.13 System and EPU TDI signal values
- 23.14 RDY bit values
- 23.15 TDO bit values
- 23.16 Advanced protocol effects on the EPU/CLTAPC relationship
- 23.17 SSD detection
- 23.18 Programming considerations
- 23.19 An approach to the implementation of a TAPC controller with maximum performance

### 23.2 Deployment

A high-level block diagram of the deployed T4 TAP.7 is shown in Figure 23-1.



**Figure 23-1 — Deployment of the T4 TAP.7**

## 23.3 Capabilities

### 23.3.1 Inherited

All mandatory capabilities of a T3 TAP.7 are also mandatory for a T4 TAP.7 provided the TDIC and TDOC signal functions are available. When these signal functions are not available, all mandatory capabilities of a T3 TAP.7 are also mandatory for a T4 TAP.7 with the exception of functions that rely on the availability the TDIC and TDOC signal functions. The ability to perform Optional capabilities supported by T3 TAP.7s are also options for a T4 TAP.7.

The T3 TAP.7 capabilities inherited by a T4 TAP.7 ensure it:

- Uses the command infrastructure provided by the EPU
- Provides equivalent operation with the JScan0–JScan3 Scan Formats provided the TDIC and TDOC signal functions are available
- Support direct addressability and the use of Controller IDs

### 23.3.2 New

The T4 TAP.7's new capabilities include but are not limited to:

- Wide or narrow configurations (see 21.5.1)
- Four start-up options
- Three mandatory and ten optional scan formats utilizing the Advanced Protocol
- Operation with either the rising or the falling edge of the TCKC signal sampling the TMSC signal when using the Advanced and Control Protocols
- Programmable or fixed TDIC and TDOC signal functions with a wide T4 TAP.7
- A means to balance the sys\_tck signal duty cycle with some scan formats to allow higher TCKC signal operating frequencies
- Configuration Fault detection

## 23.4 Register and command portfolio

### 23.4.1 Description

#### 23.4.1.1 General information

Five new registers control the T4 TAP.7 features. In addition, a five bit Scan Format Register is required. These registers and the commands managing their contents are shown in Table 23-1. Although not listed in this table, writes to the TP\_ST register will be monitored to the extent necessary to detect a Configuration Fault. When a T5 TAP.7 register or a bit within it is not implemented, the register value is hardwired as a logic 0.

**Table 23-1 — T4 TAP.7 function/register/command relationships**

Function	Register	Type	Associated command	Described in subclause
<b>Auxiliary Pin Function Control</b>	<b>APFC</b>	<b>W</b>	STC2	23.4.1.4.1
<b>Delay Control</b>	<b>DLYC</b>	<b>W</b>	STMC	23.4.1.4.2
<b>Ready Control</b>	<b>RDYC</b>	<b>W</b>	STMC	23.4.1.4.3
<b>Sample Using Rising Edge</b>	<b>SREdge</b>	<b>W</b>	STC1	23.4.1.4.4
Scan Format	SCNFMT	W	STFMT	23.2.1.4.5
<b>sys_tck Duty Cycle</b>	<b>STCKDC</b>	<b>W</b>	STC2	23.2.1.4.6

NOTE—The registers and commands that are added with this TAP.7 Class are shown in **BOLD** print. Table 23-2 provides a description of the register values.

#### 23.4.1.2 Register acronyms

The acronyms for the control registers shown in Table 23-1 are described briefly as follows:

- AFPC Auxiliary Pin Function Control
- DLYC DeLaY Control
- RDYC ReaDY Control
- SREdge Sample Rising EDGE
- SCNFMT SCaN ForMaT
- STCKDC System Test ClocK Duty Cycle control

#### 23.4.1.3 Effect of a Long-Form Selection Sequence

The DLYC, RDYC, and SCNFMT Registers are loaded with a Long-Form Selection Sequence, while other registers in Table 23-1 are not.

### 23.4.1.4 New register descriptions

#### 23.4.1.4.1 Auxiliary Pin Function Control (APFC) Register

The Auxiliary Pin Function Control (APFC) Register provides a means for the DTS software to control the function assigned to the TDIC and TDOC signals with a wide T4 and above TAP.7 when the function of these signals maybe either the T3 TAP.7 function or an Auxiliary Function. This register is not implemented when the signal function is dedicated or absent. The APFC Register should be programmed to provide the T3 TAP.7 TDIC and TDOC signal functions before using the Standard Protocol to perform data transfers (as opposed to use for only command generation) as the default TDIC and TDOC signal function may be an auxiliary function. This register may be programmed using the STC1 Command while using the Standard or the Advanced Protocol.

#### 23.4.1.4.2 Delay Control (DLYC) Register

The Delay Control (DLYC) Register provides a means for the DT to delay the completion of an SP. This delay is essentially a DTS stall of SP progression. This delay may be zero, one, two, or a variable length of TCKC signal periods. It is especially useful when the TCKC signal is sourced by the TS and a Standard-to-Advanced Protocol adapter is added to a DTS that supports only the Standard Protocol. It allows a DTS design additional time to receive the TDO signal, advance the DTS TAPC state, receive the TMS and TDI signal information, and begin generation of the next SP Payload Element. The DLYC Register should be programmed before the Advanced Protocol is used when the DTS requires a delay other than zero. It may be programmed either with the STMC Command while using the Standard or the Advanced Protocol or by the Long-Form Selection Sequence.

#### 23.4.1.4.3 Ready Control (RDYC) Register

The Ready Control (RDYC) Register defines behavior exhibited in the SP payload output bit-frames when the STL stall opportunities (RDY bits) are included as control information. This register defines the number of additional bits inserted in these bit-frames. These bits provide more time for a high-performance DTS to ascertain whether the TAP.7 Controller is ready to complete the SP payload, especially when the TS sources the TCKC. The use of RDY bits makes a high-performance DTS design easier as input and output buffer delays play less of a role in capping the TAP.7 Controller performance. The RDYC Register should be programmed before the Advanced Protocol is used. It may be programmed either with the STMC Command while using the Standard or the Advanced Protocol or by the Long-Form Selection Sequence.

#### 23.4.1.4.4 Sample Using Rising Edge (SREDGE) Register

The Sampling Rising Edge (SREDGE) Register defines the TCKC signal edge used to sample the TMSC signal input when the Control and Advanced Protocols are used with T4 and above TAP.7s. The SREDGE Register specifies sampling the TMSC signal value with the falling edge of the TCKC signal following a Type-0-Type-3 Reset. It should be programmed using the STC1 Command before the Advanced Protocol is used when sampling the TMSC signal value with the rising edge of the TCKC signal is required. It cannot be programmed using the Long-Form Selection Sequence.

#### 23.4.1.4.5 Scan Format (SCNFMT) Register

The Scan Format Register, introduced with a T2 TAP.7, defines the JScan, MScan, OScan, or SScan Scan Format. This register is five bits with a T4 and above TAP.7.

#### 23.4.1.4.6 System Test Clock Duty Cycle (STCKDC) Register

The System Test Clock Duty Cycle (STCKDC) Register provides a means to adjust the duration of the System Test Clock Logic 1 time within the *sys\_tck* signal period from 0.5 to 2.0 TCKC signal periods in 0.5 TCKC signal-period increments, subject to compatibility with the scan format. This optional feature can

be used to balance the duty cycle of the System Test Clock provided to the STL. This can allow a higher TCKC signal rate as it is likely that STL uses both edges of the System Test Clock (*sys\_tck*). The adjustment range is modulated by the scan format as SP payloads can have a minimum of one, two, three, four, or six bits.

Note that the STCKDC Register value affects the timing of boundary-scan operations utilizing the falling edge of the TCKC. One of the following actions will be taken to ensure the same timing for these boundary-scan operations at the boundary-scan endpoints described in Clause 27:

- This register will be set to zero in all TAP.7 Controllers
- This register will be set to the same nonzero value in all TAP.7 Controllers, provided the following criteria are met:
  - There are only TAP.7 Controllers in the topology branches involved in boundary-scan testing
  - Each TAP.7 Controller involved in boundary-scan testing is implemented with the STCKDC Register

### 23.4.2 Specifications

#### Rules

- a) Each subsequent specification in 23.4.2 shall only apply to a T4 and above TAP.7.
- b) The definition of the APFC, DLYC, RDYC, SCNFMT, SREDGE, and STCKDC Registers shall be governed by Table 23-2.

**Table 23-2 — T4 TAP.7 register descriptions**

<b>Auxiliary Pin Function Control (APFC)</b> – Controls the function of the TDIC and TDOC pins of a wide T4 and above TAP.7	
YY	If the default pin function is the standard pin function: yy=00:No change in the default pin function. yy=01:The pin function becomes the standard pin function. yy=1x:The pin function becomes the auxiliary pin function.
<b>Delay Control (DLYC)</b> – Defines the length of the DTS delay	
YY	yy=0:No DTS delay is added. 1:Add one TCKC signal period. 2:Add two TCKC signal periods. 3:Add a variable number of TCKC signal periods.
<b>Ready Control (RDYC)</b> – Defines the number of RDY bit periods preceding a TDO bit	
YY	With the MScan Scan Format, the number of RDY bits between precharge bits:  yy=00:One 01:Two 10:Three 11:Four  With a scan format other than the MScan Scan Format, the number of logic 1 RDY bits preceding the last bit of the SP payload:  yy=00:One 01:Two 10:Three 11:Four
<b>Scan Format (SCNFMT)</b> – Defines the scan protocol attributes	
YYYYY	YYYYY=00:JScan0 (same as T3 TAP.7). 01:JScan1 (same as T3 TAP.7). 02:JScan2 (same as T3 TAP.7). 03:JScan3 (same as T3 TAP.7). 04:SScan0 (see Clause 26). 05:SScan1 (see Clause 26). 06:SScan2 (see Clause 26). 07:SScan3 (see Clause 26). 08:OScan0 (see Clause 25). 09:OScan1 (see Clause 25). 10:OScan2 (see Clause 25). 11:OScan3 (see Clause 25). 12:OScan4 (see Clause 25). 13:OScan5 (see Clause 25). 14:OScan6 (see Clause 25). 15:OScan7 (see Clause 25). 16:MScan (see Clause 24). 17-31:Reserved.

**Table 23-2 — T4 TAP.7 register descriptions  
(continued)**

<b>sys_tck Duty Cycle (STCKDC)</b> – Specifies the duty cycle for the System Test Clock (see Table 23-4 for further register definition)	
yy	yy=00:0.5 TCK signal period. yy=01:1.0 TCK signal period, or max. permitted by the scan format. yy=10:1.5 TCK signal periods, or max. permitted by the scan format. yy=11:2.0 TCK signal periods, or max. permitted by the scan format.
<b>Sampling Rising Edge (SREDGE)</b> – Defines the TCKC signal edge used to sample the TMSC signal input	
Y	y=0: Sample the TMSC signal with the TCKC signal falling edge 1: Sample the TMSC signal with the TCKC signal rising edge

NOTE—See Table 9-2 for information regarding the initialization of these registers.

## 23.5 Configurations

### 23.5.1 Description

The options added to the T4 TAP.7 are identified in the T4 TAP.7 options field of the Configuration Register Zero described in Table 9-15. The T4 TAP.7 options field identifies the following T4 TAP.7 attributes:

- Optional scan formats supported
- TAP characteristics:
  - Default width of the TAP
  - Whether the programmable functionality of the TDIC and TDOC pins is supported
  - Whether the programmable functionality of the TDIC and TDOC pins is locked
  - Whether the sys\_tck signal duty-cycle adjustment is supported

### 23.5.2 Specifications

#### Rules

- a) A T4 TAP.7 shall be implemented with the Class Field of the Configuration Register Zero set to represent the T4 TAP.7 Class as shown in Table 9-15.
- b) Each subsequent specification in 23.5.2 shall only apply to a T4 and above TAP.7, provided Configuration Register Zero bits [31:12] are implemented.
- c) A T4 and above TAP.7's support of the optional TAP.7 functions shown in Table 23-3 shall be described with the Configuration Register Zero bits that are listed in this table.

**Table 23-3 — Specifying the use of T4 TAP.7 optional functions**

Optional functions	Registers implemented to support function	Support identified with Configuration Register bits:
OScan7–6 Scan Formats	SCNFMT[4:0]	OSCANS[2]
OScan5–4 Scan Formats	SCNFMT[4:0]	OSCANS[1]
OScan3–2 Scan Formats	SCNFMT[4:0]	OSCANS[0]
SScan3–2 Scan Formats	SCNFMT[4:0]	SSCANS[1]
SScan1–0 Scan Formats	SCNFMT[4:0]	SSCANS[0]
Auxiliary Pin Function Control	APFC[1:0]	APFCS
TAP Width	--	TAPWIDS
TAP Width Locked	--	TAPWLCK
sys_tck Duty Cycle	STCKDC[1:0]	STCKDCS

- d) The optional SScan Scan Formats shall be identified as either supported or not supported as specified by the SSCANS[1:0] field of the Configuration Register.
- e) The optional OScan Scan Formats implemented shall be identified as either supported or not supported as specified by the OSCANS[2:0] field of the Configuration Register.
- f) The implementation of the Auxiliary Pin Function Control shall be identified as either supported or not specified by the APFCS field of the Configuration Register.
- g) The default TAP width shall be identified by the TAPWIDS field of the Configuration Register.
- h) The locking of the APFC Register value shall be identified by the TAPWLCK field of the Configuration Register.
- i) The implementation of the STCKDC Register shall be identified as specified by the STCKDCS field of the Configuration Register.

## 23.6 Start-up behavior

### 23.6.1 Description

The start-up behavior of a T4 TAP.7 is created with the one of the following start-up options:

- IEEE 1149.1-Compliant
- IEEE 1149.1-Compatible
- IEEE 1149.1-Protocol Compatible
- Offline-at-Start-up

These start-up options and the behavior they create are described in 10.3. The behavior created by these start-up options is subject to the behavior created by the implementation of the Power-Mode Function.

### 23.6.2 Specifications

#### Rules

- a) Each subsequent specification in 23.6.2 shall only apply to a T4 TAP.7.

- b) Start-up behavior shall be that specified by one of the following start-up options, subject to the behavior created by the implementation of the Power-Mode Function.
  - 1) IEEE 1149.1-Compliant.
  - 2) IEEE 1149.1-Compatible.
  - 3) IEEE 1149.1-Protocol Compatible.
  - 4) Offline-at-Start-up.

## 23.7 Scan formats

### 23.7.1 Description

#### 23.7.1.1 Overview

Three groups of Advanced Scan Formats (MScan, OScan0–OScan7, and SScan0–SScan3) are added the JScan Scan Formats inherited from a T3 TAP.7 to specify the use of the Advanced Protocol as described in 4.2.7.7 and 21.4.3. The MScan, OScan0, and OScan1 Scan Formats are mandatory. The OScan2–Oscan7 and SScan0–SScan3 Scan Formats are optional and are deployed in pairs.

These Advanced Scan Formats complement the JScan Scan Formats inherited from a T3 TAP.7. The added scan formats provide a number of options for the serialization of scan information as shown in Figure 23-2. The MScan Scan Format provides a single set of capability. The OScan and SScan Scan Formats each provide four sets of capability with this capability duplicated and optimized for a DTS- and a TS-sourced TCKC signal as shown in this figure.

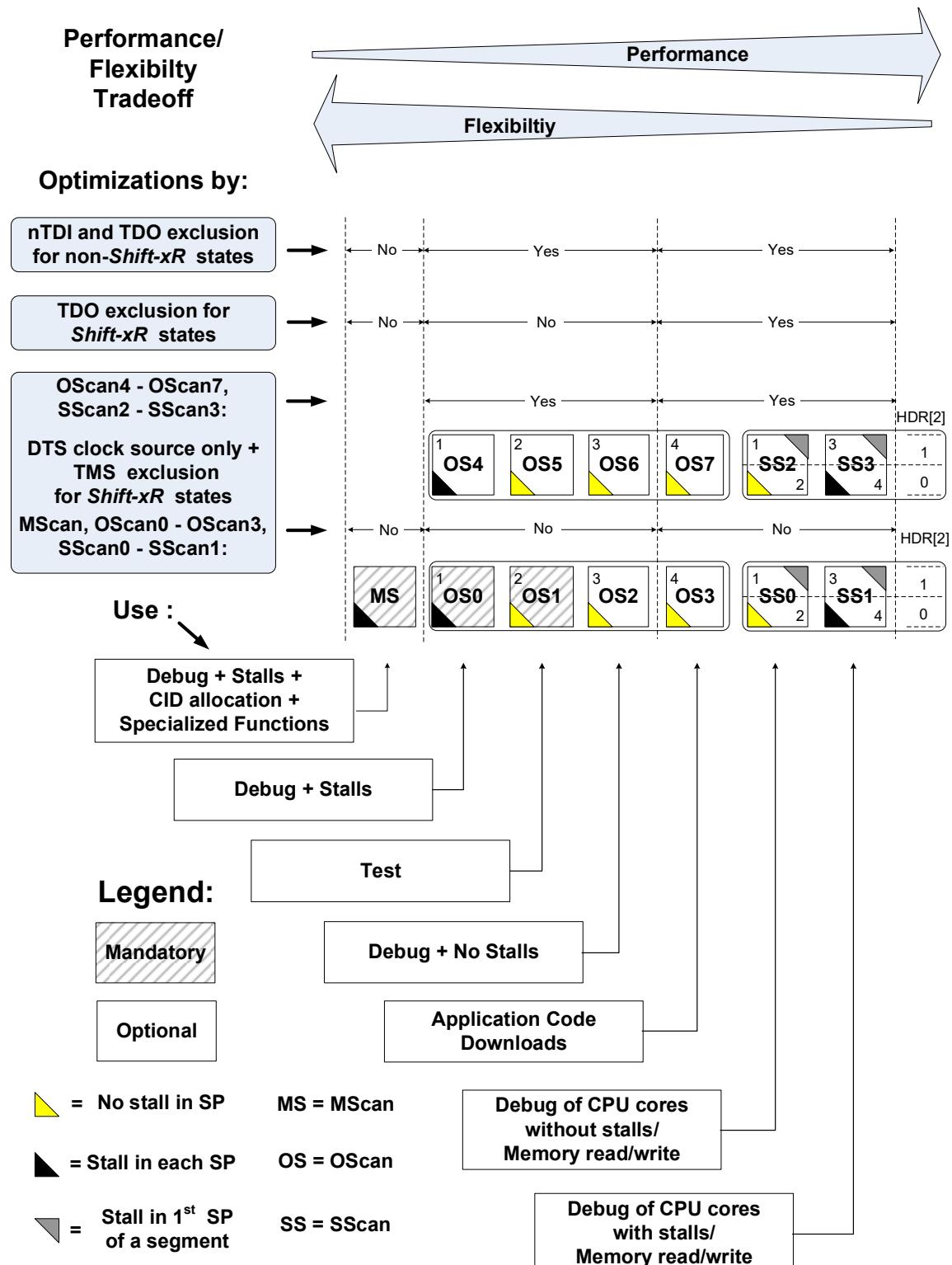


Figure 23-2 — Advanced Scan Format overview

### 23.7.1.2 Addition of optional scan formats

The optional Advanced Scan Formats may be added in groups of two with the implementation of any combination of the groups shown below permitted:

- OScan2–OScan3      Optional
- OScan4–OScan5      Optional
- OScan6–OScan7      Optional
- SScan0–SScan1      Optional
- SScan2–SScan3      Optional

### 23.7.1.3 Influences on scan format definition

The Advanced Scan Formats provide seven different functions. Five factors have influenced the definition of these scan formats:

- Supporting the capabilities of IEEE Std 1149.1, this standard, and IEEE Std 1532 [B4]
- Providing to the CLTAPC and DTS an opportunity to stall the TAPC state progression
- Providing scalable functionality
- Maximizing the efficiency of scan operations utilized for application debug operations
- Providing a rich set of capability for a high-end application debug

### 23.7.1.4 Performance and flexibility tradeoffs

The DTS chooses a scan format that best matches the performance/functionality needs of the application. The flexibility of the scan transfer is increased and its performance is decreased when the amount of data and control information included in the Payload Element is increased. The scan performance is maximized when the SP contains:

- Only the needed TDI, TMS, and TDO signal information for *Shift-xR* and non-*Shift-xR* states
- The smallest amount of control information possible

In the extreme case of optimizing for performance, the DTS may minimize the information transferred with SScan Scan Formats by using its understanding of the application to transfer only the data of interest for groups of *Shift-xR* states.

The Scan Packets created while using the MScan Scan Format are as follows:

- Defined only by the scan format
- The most flexible ones, handling both test and debug needs:
  - Debug                          With stalls/one or more Scan Group Members
  - Boundary-scan testing      With stalls/one or more Scan Group Members
- Having a unique feature as MScan is the only scan format that can be used for:
  - Operation with multiple Scan Group Members while using stalls (RDY bits in the SP payload)
  - Undirected CID Allocation with the Star-2 Scan Topology

The Scan Packets created while using the OScan Scan Format are as follows:

- Defined by a combination of the:
  - Scan format
  - TAPC state
- Optimized for various applications:
  - Debug With or without stalls
  - Boundary-scan testing With or without stalls
  - Applications code download Without stalls
- Directed CID Allocation with the Star-2 Scan Topology

The Scan Packets created while using the SScan Scan Format are as follows:

- Defined by a combination of the:
  - Scan format
  - The TAPC state
  - The most significant Header Element bit (HDR[2])
- Optimized for debug applications:
  - Accessing resources with TAP.1 characteristics i.e., CPU cores not requiring stalls
  - Accessing resources that are rate dependent i.e., CPU cores requiring stalls
  - Accessing memory Word-oriented transfers

### 23.7.1.5 Comparing the scan formats

#### 23.7.1.5.1 MScan

The MScan Scan Format creates Payload Elements that are a minimum of six TCKC signal periods. The input and output bit-frames have the same information for all TAPC states. This provides for voting on the:

- RDY bit value when there are multiple Scan Group Members
- TDO bit value during Undirected CID Allocation

The key characteristics of the MScan Scan Format are listed as follows:

- Highest flexibility, lowest performance
- Usable for any application
- Supports virtually all, if not all, IEEE 1149.1 IP with either IEEE 1149.1-Specified or IEEE 1149.1-Non-disruptive Behaviors
- Use with multiple Scan Group Members when stalls are required
- Undirected and Directed CID Allocation

#### 23.7.1.5.2 OScan

The OScan Scan Formats create Payload Elements with:

- One to three bits when RDY bits are not included

- Two to three bits + the number of RDY bits when these bits are included

The key characteristics of the OScan Scan Formats are listed as follows:

- The DTS's knowledge of the application is used to increase scan performance
- The SP Payload Element is optimized for specific test and debug applications
- Optimizations used are:
  - nTDI and TDO bit exclusion in non-*Shift-xR* states
  - TMS bit exclusion in *Shift-xR* states
  - TDO bit exclusion in *Shift-xR* states

### 23.7.1.5.3 SScan

The SScan Scan Formats create Payload Elements with:

- One to two bits when RDY bits are not included
- Two to three bits + the number of RDY bits when these bits are included

The key characteristics of the SScan Scan Formats are listed as follows:

- The DTS's knowledge of the application and data of importance within *Shift-xR* TAPC states is used to increase scan performance
- SP payloads are optimized:
  - For specific applications
  - To transfer only the needed data

Optimizations divide transfers into data and Control Segments that are individually optimized as follows:

- Control Segment (non-*Shift-xR* states) optimizations:
  - nTDI bits are excluded
  - TDO bit exclusion in most if not all SPs
- Data Segment (*Shift-xR* states) optimizations:
  - TMS bits are excluded
  - nTDI bit exclusion when desired
  - TDO bit exclusion when desired

The DTS determines the following segment characteristics:

- Number of control and Data Segments (one or multiple)
- Segment length(s)
- Segment content

## 23.7.2 Specification

### Rules

- a) Each subsequent specification in 23.7.2 shall only apply to a T4 TAP.7 while using the Advanced Protocol until stated otherwise.
- b) T4 TAP.7 implementations shall support the MScan, OScan0, and OScan1 scan formats.
- c) When implemented, the optional scan formats shall be implemented in pairs as shown in 23.7.1.2.

### Permissions

- d) The optional scan format pairs shown in 23.7.1.2 may be implemented in any combination.

## 23.8 Configuration Faults

### 23.8.1 Description

A T4 TAP.7 Controller built in accordance with this specification revision declares the following conditions a Configuration Fault when using an Advanced Scan Format:

- The scan format specified by the SCNFMT Register is not supported.
- The scan format specified by the SCNFMT Register is OScan0, OScan4, or SScan and any of the following are true:
  - The state of the Potential Scan Group Membership Count Last (PSGMCL) is *PSGMCL\_MANY* (see Table 13-6).
  - The state of the Scan Group Candidate Count (SGCC) is *SGCC\_MANY* (see Table 13-5).
- Transport is enabled with a nonzero TPST Register value.

### 23.8.2 Specifications

### Rules

- a) Each subsequent specification in 23.8.2 shall only apply to a T4 TAP.7 while using the Advanced Protocol until stated otherwise.
- b) A TAP.7 Controller state with a TPST Register value that enables the Transport Function (a nonzero value) shall be considered as specifying a Configuration Fault.
- c) Each subsequent specification in 23.8.2 shall only apply to a T4 and above TAP.7 until stated otherwise.
- d) A TAP.7 Controller state that includes any of the following shall be considered as specifying a Configuration Fault:
  - 1) The SCNFMT Register value n where all of the following are true:
    - i)  $n \geq 10001b$ .
    - ii)  $n \leq 11111b$ .
  - 2) The SCNFMT Register value specifying the use of an optional scan format that is not supported.
- e) A TAP.7 Controller state that includes all the following shall be considered as specifying a Configuration Fault:
  - 1) The SCNFMT Register value specifies the use of any of the following scan formats:
    - i) The OScan0 Scan Format.

- ii) The OScan4 Scan Format.
- ii) Any SScan Scan Format.
- 2) Any of the following are true:
  - i) The state of the Scan Group Candidate Count (SGCC) is *SGCC\_MANY*.
  - ii) The state of the PSGMCL is *PSGMCL\_MANY*.

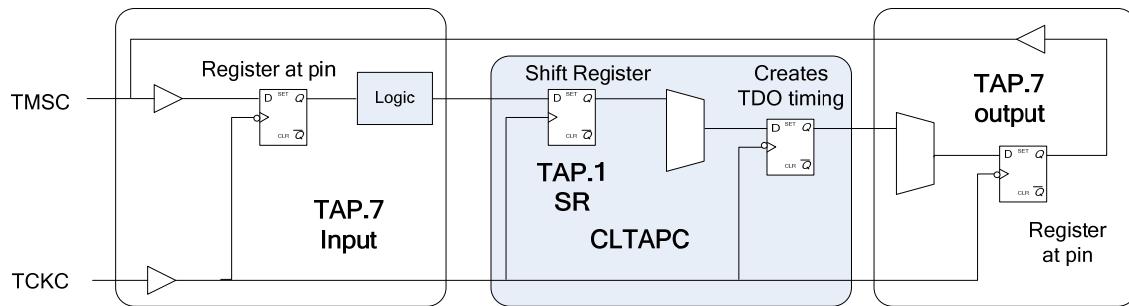
NOTE—Rule 23.8.2 e) prevents a transition from the use of the JScan0–JScan2 Scan Formats directly to the use of any of the single user scan formats listed in this rule. A transition from the use of a JScan0–JScan2 Scan Format to the OScan1 Scan Format (where the STL of interest is made Scan Group Member) followed by a transition to the use of one of the single user scan formats listed in this rule is recommended.

## 23.9 Increasing STL performance

### 23.9.1 Description

The STL performance may be increased with some scan formats using the optional TAP.7 capability to increase the logic 1 period of the *sys\_tck* duty cycle. This option is provided to balance the *sys\_tck* signal duty cycle when certain Scan Packets with multi-bit payloads are used. This may allow higher TCKC signal operating frequencies with the STL, especially when the factor limiting the operating frequency of the STL is the propagation delay of scan data generated on the rising edge of the TCKC signal to the falling edge of the TCKC signal. This capability is controlled with the STCKDC Register. It is only relevant when the timing paths within the chip between the rising and the falling edge of the TCKC signal are longer than one-half the desired maximum TCKC signal frequency.

Figure 23-3 shows a simplified TAP.7 data path using the CLTAPC Bypass Bit. The TAP.1 Shift Register is clocked on the rising edge of the TCKC signal, with the CLTAPC TDO signal timing created with the falling edge of the TCKC signal. The TAP.7 Controller registers the TMSC signal input and output on the falling edge of the TCKC signal in this example.



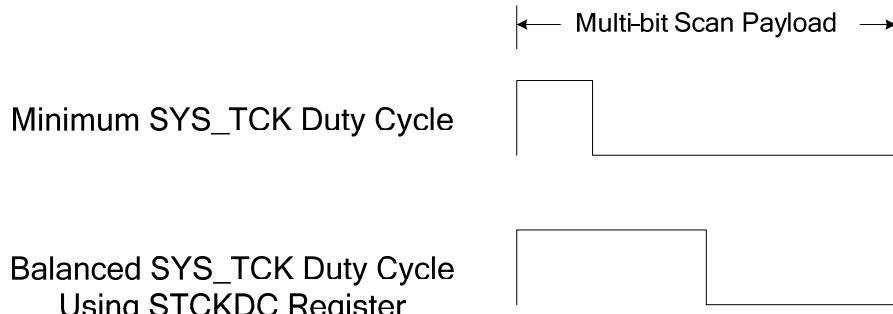
**Figure 23-3 — TAP.7 data path with positive- and negative-edge clocking**

This figure reveals the positive- and negative-edge timing in the CLTAPC. Data is moved from a Flip-Flop that is clocked with the positive edge of the TCKC signal to a Flip-Flop that is clocked with the negative edge of the TCKC signal. When the TCKC signal is merely gated to create the *sys\_tck* signal and the minimum number of bits forming an SP payload is more than one, the ratio of the time needed to transfer the minimum number of bits that form the SP payload to the time the *sys\_tck* signal remains a logic 1 is at least:

$$[(2 \times \text{the number of payload bits}) - 1] \text{ to } 1$$

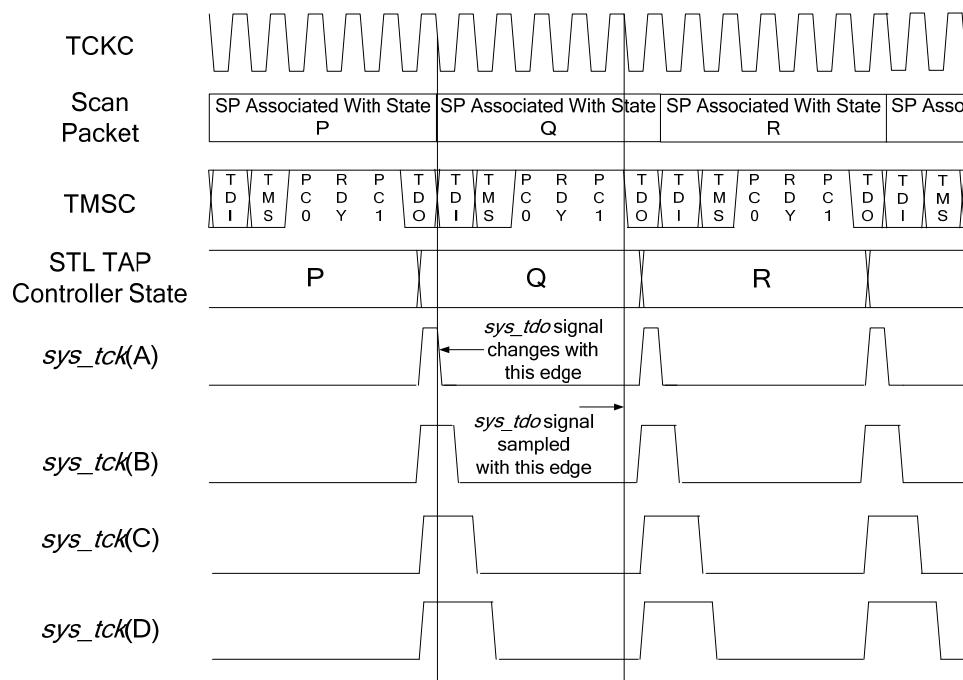
This can limit the performance of the STL, as the rising TCKC signal to the TCKC signal falling-edge timing path is merely the *sys\_tck* signal high time and the falling edge to rising-edge path is the *sys\_tck* signal low time. Assuming there is a similar number of logic levels in each path, these circuits would

operate faster if the difference between the logic 1 and a logic 0 periods is minimized. An example of this is shown in Figure 23-4.



**Figure 23-4 — The *sys\_tck* signal**

The *sys\_tck* signal duty cycle may be adjusted with the STCKDC Register, depending on the number of bits in the Payload Element. This may raise the TAP.7 Controller operating frequency, if these timing paths are greater than the TCKC signal's logic 1 time. The ability to increase the time between these clock edges is constrained by the scan format being used and the TAPC state as shown in Figure 23-5.



Legend:  $sys_tck(x)$  where  $x = A, B, C$ , or  $D$  shown in Figure 23-6

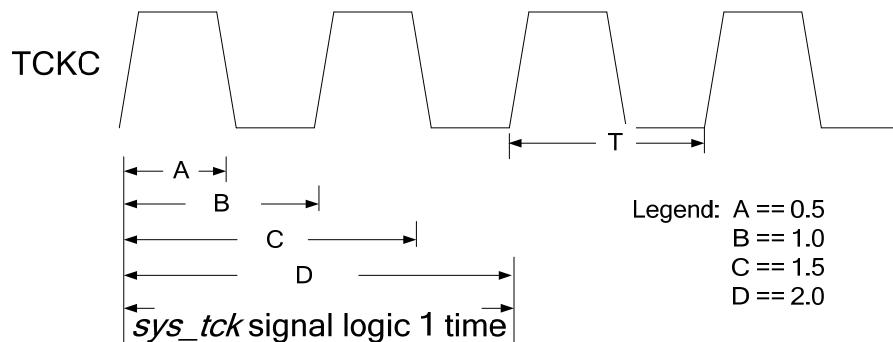
**Figure 23-5 — Example of the MScan Scan Format with *sys\_tck* signal duty-cycle options**

## 23.9.2 Specifications

### Rules

- a) Each subsequent specification in 23.9.2 shall only apply to a T4 and above TAP.7, provided the STCKDC Register is implemented.

- b) Programmability of the *sys\_tck* signal high time following its rising edge shall be governed by Table 23-3 where the widths identified in this table are governed by Figure 23-6.



**Figure 23-6 — *sys\_tck* signal logic 1 time**

**Table 23-4 — STCKDC register relationship to *sys\_tck* signal generation**

Scan format	TAPC state	STCKDC Register value			
		00b	01b	10b	11b
Any SScan	Any	0.5T	0.5T	0.5T	0.5T
OScan7	Any	0.5T	0.5T	0.5T	0.5T
OScan6	Any	0.5T	0.5T	0.5T	0.5T
OScan5	Any	0.5T	0.5T	0.5T	0.5T
OScan4	Any	0.5T	1.0T	1.5T	1.5T
OScan3	Any	0.5T	0.5T	0.5T	0.5T
OScan2	Others	0.5T	0.5T	0.5T	0.5T
	Shift-xR	0.5T	1T	1T	1T
OScan1	Any	0.5T	1T	1.5T	1.5T
OScan0	Any	0.5T	1T	1.5T	2T
MScan	Any	0.5T	1T	1.5T	2T

NOTE—Values are defined by Figure 23-6.

### Permissions

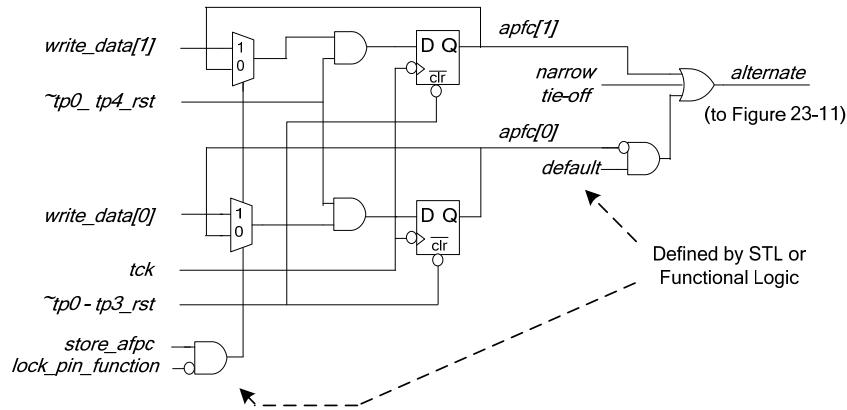
- c) With a T4 and above TAP.7, the STCKDC Register may be implemented to make the logic 1 period of the *sys\_tck* (System Test Clock) signal period programmable per Table 23-4.

## 23.10 Auxiliary Pin Function Control

### 23.10.1 Description

A Type-0–Type-4 Reset initializes the APFC Register to 00b, thereby enabling the default TDIC and TDOC signal functions. Subsequently, the register may be stored with the values shown in Table 23-2 to force the TDIC and TDOC signal functions to the T3 TAP.7 functions or auxiliary functions, provided the value of the APFC Register is not locked by the STL or functional logic.

A conceptual view of the APFC Register and its effect on the TDIC signal function is shown in Figure 23-7.



**Figure 23-7 — Conceptual view of APFC operation**

## 23.10.2 Specifications

### Rules

- a) Each subsequent specification in 23.10.2 shall only apply to a wide T4 and above TAP.7 implementing the APFC Register per permission 23.10.2 e).
- b) The function of the TDIC and TDOC signals shall be controlled by the APFC Register as described in Table 23-2.
- c) The function of the TDIC and TDOC signals shall immediately become their default function when a Type-0–Type-3 Reset becomes active.
- d) The function of the TDIC and TDOC signals shall become their default function following the first TCKC falling edge after a Type-4 Reset becomes active.

### Permissions

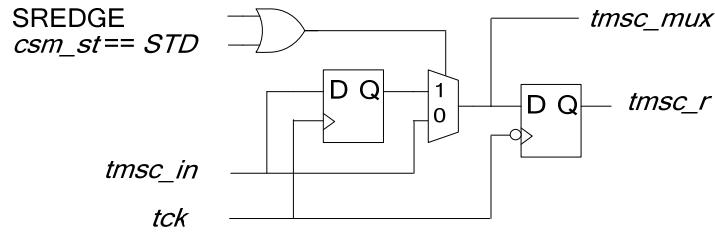
- e) With a wide T4 and above TAP.7, the function of the TDIC and TDOC signals may be made programmable by implementing the APFC Register.
- f) When the APFC Register is implemented per Permission 23.10.2 e), the value of this register may be locked by the STL or functional logic.

## 23.11 Sample Using Rising Edge

### 23.11.1 Description

Sampling the TMSC signal input with either the rising or the falling edge of the TCKC signal is a mandatory function with T4 and above TAP.7s. Using rising-edge sampling more than halves the TCKC signal operating frequency as the DTS/TS exchanges have less time to occur but may be advantageous to use if a hold-time problem is suspected with falling-edge operation.

The Sampling Rising Edge (SREdge) Register defines the TCKC signal edge used to sample the TMSC signal input as shown in Figure 23-8. The *tmsc\_mux* signal is used as the source for the TMSC input for all falling-edge-timed logic. Care should be taken to minimize the number of logic levels from the TMSC pin to registers that are clocked with the falling edge of TCKC to maximize the TCKC operating frequency.



**Figure 23-8 — SREDGE/tmsc\_mux/tmsc\_r relationships**

### 23.11.2 Specifications

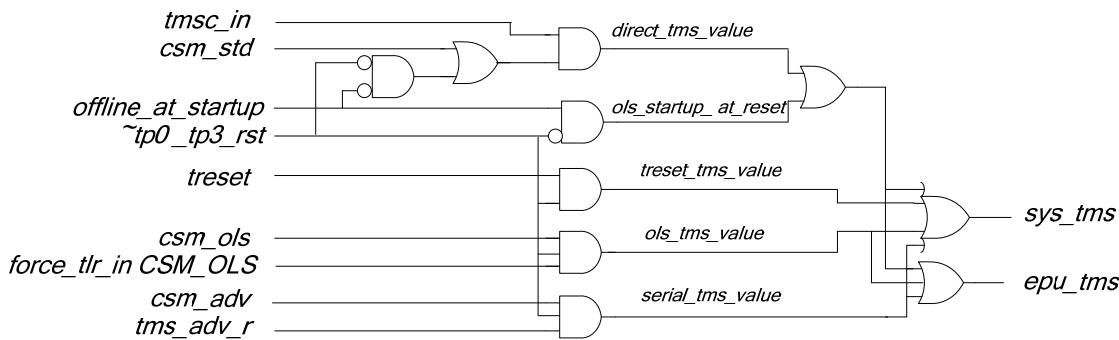
#### Rules

- a) Each subsequent specification in 23.11.2 shall only apply to a T4 and above TAP.7.
- b) The value used as the TMSC input by APU registers and state machines clocked with the falling edge of the TCKC signal shall be:
  - 1) The value of the TMSC signal sampled with the rising edge of the TCKC signal, provided the SREDGE Register value is a logic 1.
  - 2) The value of the TMSC signal, provided the SREDGE Register value is a logic 0.

### 23.12 System and EPU TMS signal values

#### 23.12.1 Description

With a T4 and above TAP.7, the APU sources the *sys\_tms* signal as shown in by Table 23-5. A conceptual view of the creation of the *sys\_tms* signal is shown in Figure 23-9. When a Type-0–Type-3 Reset is active, it determines the source of *sys\_tms* in conjunction with the Offline-at-Start-up option. Once this reset becomes inactive, the *sys\_tms* value is determined by a combination of the TRESET Register value and the TMS sources enabled by the CSM's *STD* and *ADV* states. The *STD* state utilizes the TMSC signal. The *ADV* state utilizes the TMS value created with an SP and, in some cases, an EOT Escape.



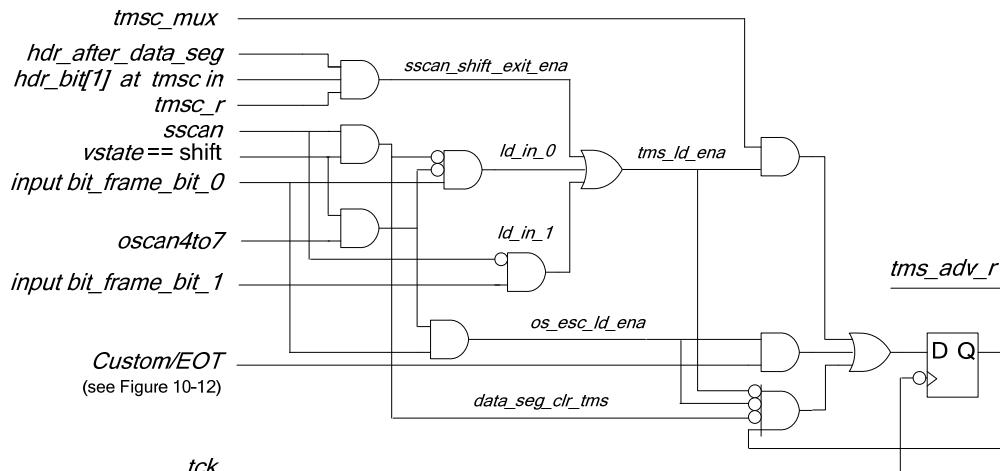
**Figure 23-9 — Conceptual view of sys\_tms generation for a T4 and above TAP.7**

A conceptual view of the creation of the TMS signal value created with the Advanced Protocol is shown in Figure 23-10. It has three components. They are created with:

- The TMSC bit for:
- All TAPC states with MScan and OScan0–OScan3 Scan Formats

- Non-*Shift-xR* TAPC states with SScan and OScan4–OScan7 Scan Formats
- An EOT Escape coincident with the nTDI bits of SPs for:
  - *Shift-xR* TAPC states with SScan and OScan4–OScan7 Scan Formats
- Header values for:
  - *Shift-xR* TAPC states with SScan Scan Formats

The TMS value for *Shift-xR* states within SScan Data Segments and OScan4–OScan7 Scan Formats is logic 0 unless forced to logic 1 by one of the terms above. The TMS value will be loaded first with the nTDI value and then the TMS value with SPs that contain both of these bits in the input frame.



**Figure 23-10 — Conceptual view of TMS generation with the Advanced Protocol**

### 23.12.2 Specifications

#### Rules

- Each subsequent specification in 23.12.2 shall only apply to a T4 and above TAP.7.
- The generation of the TMS value when the Advanced Protocol is used shall be governed by Table 23-5.

**Table 23-5 — TMS value when the Advanced Protocol is used**

Scan format	<i>Shift-xR SP ?</i>	<b>TMS value at time of TAPC state advance</b>
MScan	x	True TMS bit value in SP
OScan0-OScan3	x	True TMS bit value in SP
OScan4-OScan7	No	True TMS bit value in SP
	Yes	A logic 1 when an EOT Escape occurs coincidentally with nTDI bit of SP and a logic 0 otherwise.
SScan	No	True TMS bit value in SP
	Yes	Logic 0 until HDR[1:0] of a header following a Data Segment has a value of x11b

NOTE—All bits in the SP are sampled with the TCKC edge specified by the SREDGE Register.

- c) The value of the TMS signal (*sys\_tms*) sourced to the CLTAPC shall be governed by Table 23-6.

**Table 23-6 — *sys\_tms* signal generation**

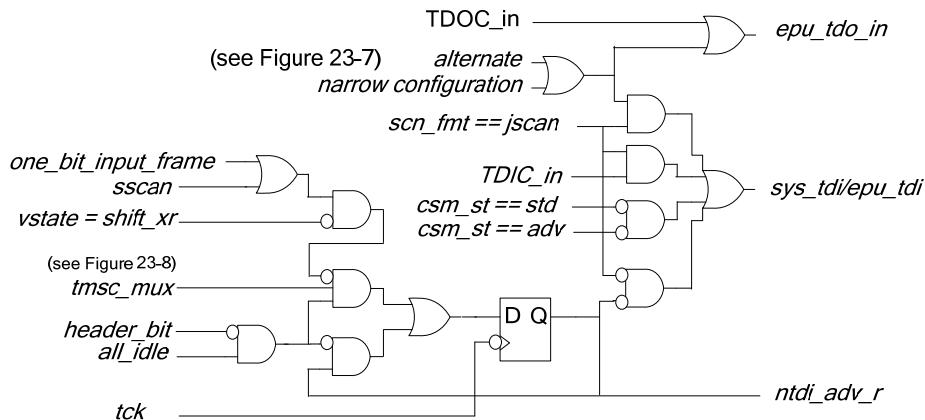
Case	Type-0-Type-3 Reset ?	Offline-at-Start-up ?	TRESET == 1 ?	CSM state	OLS initialization complete ?	<i>sys_tms</i>
A	Yes	Yes	x	x	x	Logic 1
B	Yes	No	x	x	x	TMS signal value
C	No	x	Yes	x	x	Logic 1
D	No	x	No	STD	x	TMS signal value
E	No	x	No	ADV	x	See Table 23-5
F	No	x	No	OLS	No	Logic 1
G	No	x	No	OLS	Yes	Logic 0
H	No	x	No	A state other than any listed above	x	Logic 0

- d) The value of the TMS signal (*epu\_tms*) sourced to the ADTAPC shall be governed by Table 23-6 with the TRESET input a logic 0.

## 23.13 System and EPU TDI signal values

### 23.13.1 Description

With a T4 and above TAP.7, the APU sources the *sys\_tdi* signal as shown in Table 23-7. A conceptual view of the creation of the *sys\_tdi* signal is shown in Figure 23-11. The value of the nTDI bit within an input bit-frame of an SP Payload Element is inverted before being used in the TAP.7 Controller or CLTAPC.



**Figure 23-11 — Conceptual view of sys\_tdi signal generation for a T4 and above TAP.7**

### 23.13.2 Specifications

#### Rules

- a) Each subsequent specification in 23.13.2 shall only apply to a T4 and above TAP.7.
- b) The generation TDI value when the Advanced Protocol is used shall be governed by Table 23-7.

**Table 23-7 — TDI value when the Advanced Protocol is used**

Scan format	Shift-xR SP ?	TDI value at time of TAPC state advance
MScan	x	Inverse of nTDI bit value in the SP
OScan0-OScan7	No	Inverse of nTDI bit value in the SP when this bit is present and a logic 1 otherwise
	Yes	Inverse of nTDI bit value in the SP
SScan	No	Logic 1
	Yes	Inverse of nTDI bit value in the SP when this bit is present and no change in the TDI value otherwise

NOTE—All bits in the SP are sampled with the TCKC edge specified by the SREdge Register.

- c) The value of the TDI signal sourced to the EPU and CLTAPC shall be governed by Table 23-8.

**Table 23-8 — sys\_tdi/epu\_tdi signal generation**

Case	Protocol ?	Configuration ?	Auxiliary TDIC and TDOC pin functions?	sys_tdi and epu_tdi signal values
A	Standard	Wide	No	TDIC signal value
B	Standard	Wide	Yes	Logic 1
C	Standard	Narrow	N/A	Logic 1
D	Advanced	x	x	see Table 23-7
E	Control	x	x	Logic 1

- d) The value of the TDOC input signal sourced to the EPU shall be governed by Table 23-9.

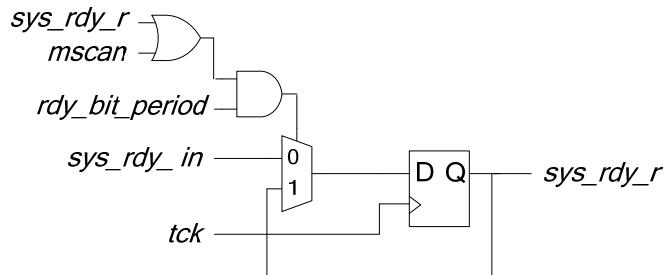
**Table 23-9 — *epu\_tdo\_in* signal generation**

<b>Case</b>	<b>Configuration ?</b>	<b>Auxiliary TDIC and TDOC pin functions?</b>	<b><i>epu_tdo_in</i></b>
A	Wide	No	TDOC signal value
B	Wide	Yes	Logic 1
C	Narrow	N/A	Logic 1

## 23.14 RDY bit values

### 23.14.1 Description

With the MScan Scan Format, the value of all RDY bits between precharge bits will be the same. A conceptual view of how this may be accomplished is shown in Figure 23-12. Note the sampling of the multiplexer output with the rising edge of the TCKC. This provides the highest performance operation with scan formats with RDY bits. The clocking of this Flip-Flop may be altered if required.



**Figure 23-12 — Conceptual view of System Ready treatment in its simplest form**

A high-level view of RDY generation with different types of stall generation is shown in Figure 23-13. Two types of System Ready generation are shown in this figure. The first is with circuitry that accommodates the unorthodox use of Test Clock (described in 1.9.2.3) to slow the pace of the TAP.7 Controller advance. This circuitry generally returns *sys\_tck* through an n-stage synchronizer clocked with a functional clock. The scan formats with RDY bits in every Scan Packet support a Return Test Clock. The Return Test Clock is asynchronous to TCKC.

The second type of System Ready is generated by interaction with a component capable of reading or writing memory. SScan Scan Formats with an RDY bit in the first SP are provided to allow this type of component to control the pace of the data exchange.

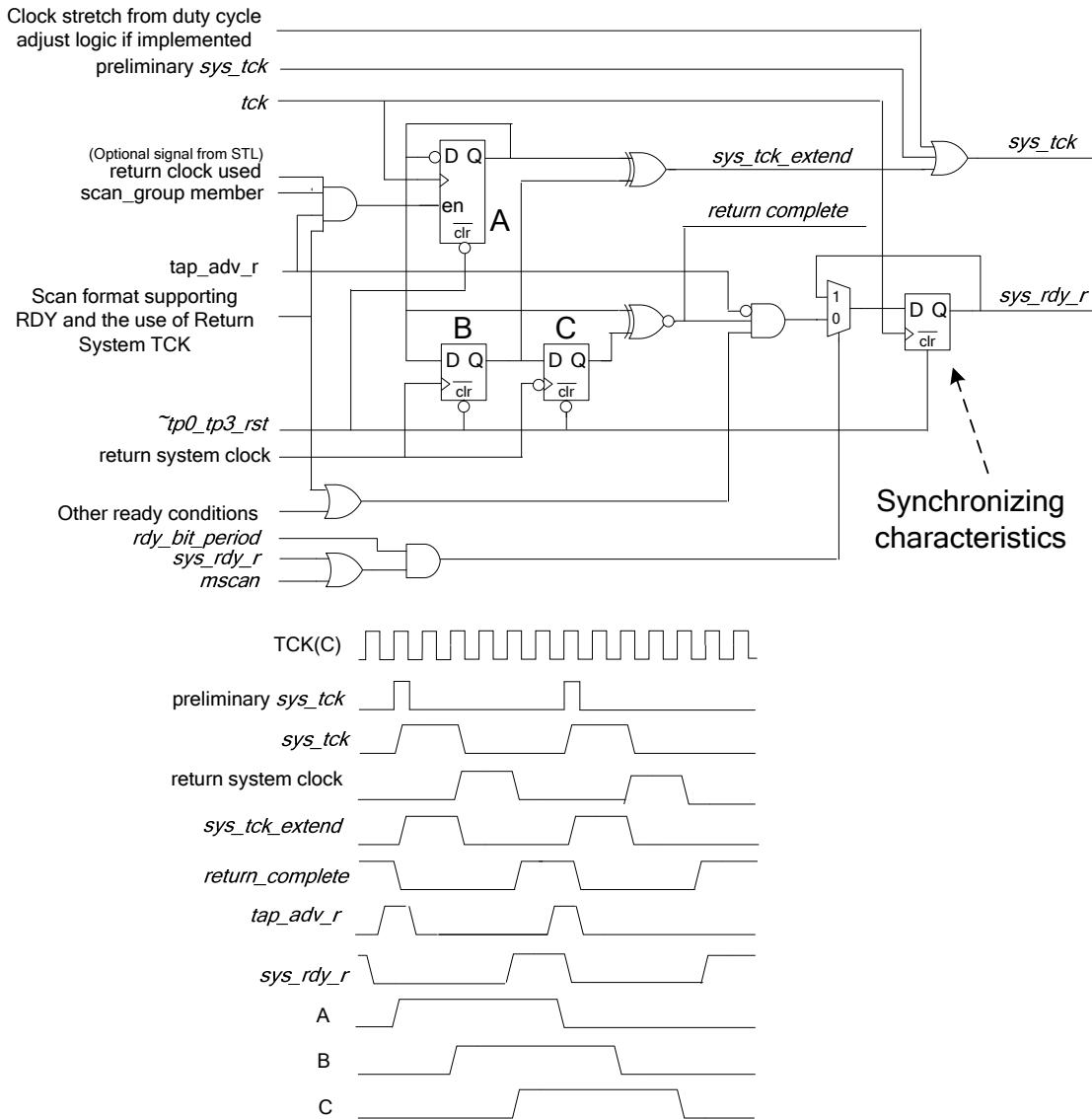


Figure 23-13 — Conceptual *sys\_rdy* generation for more complex use cases

### 23.14.2 Specifications

#### Rules

- Each subsequent specification in 23.14.2 shall only apply to a T4 and above TAP.7.
- The value determining the drive policy for the RDY bit in the SP payload shall be a logic 1, provided all of the following are true, and a logic 0 otherwise:
  - TAP.7 Controller can accept the next SP.
  - The value of the TDO bit in the SP is available for output.
  - Either of the following is true:

- i) The CLTAPC is deselected.
- ii) The state of CLTAPC is the same as the ADTAPC state.
- 4) Any further conditions permitted by Permission 23.14.2 c) are satisfied.

## Permissions

- c) The conditions used to create a logic 1 RDY bit value specified in Rule 23.14.2 b) above may include other criteria.

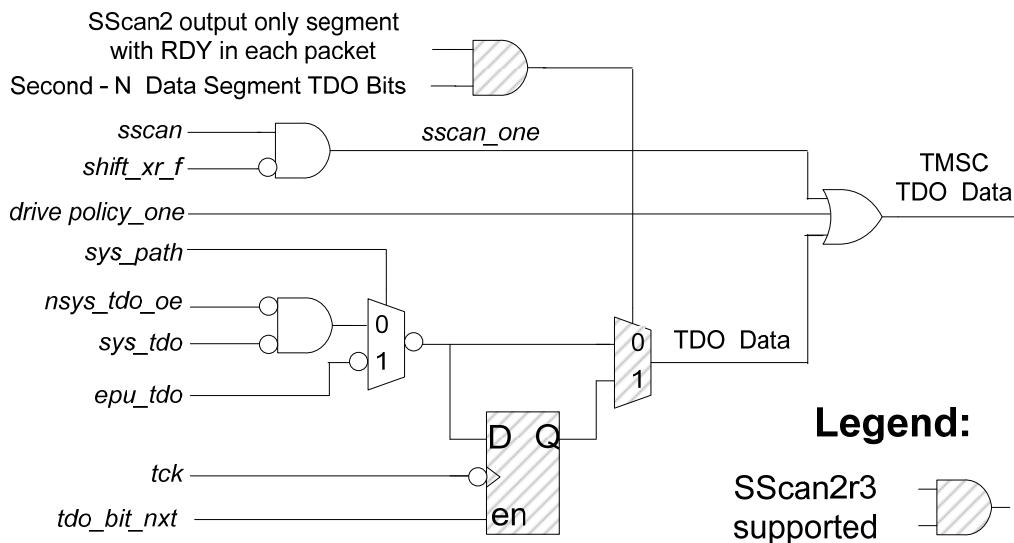
## 23.15 TDO bit values

### 23.15.1 Description

The value of TDO bits in SPs payloads when the TMSC signal is driven during TDO bit periods is established by the types of drive (Single, Joint, Voting, and Inhibited) described in Clause 14.

With SScan2 and SScan3 output-only segments, a segment associated with  $n$  *Shift-xR* states has  $n + 1$  SPs, provided  $n > 1$ . In this case, the TDO value of the first *Shift-xR* state is associated with the first two SPs, while the remaining SPs are associated sequentially with the subsequent *Shift-xR* states thereafter. This is handled with the delay of the merged TDO data as shown in Figure 23-14. Note that the delay of TDO data is only necessary with an SScan2 Scan Format and header value of six (110b). At other times, the delay of TDO data is a result of pipelining considerations. This adjustment in TDO timing provides the DTS with consistent TDO data behavior for the SScan2 and SScan3 Scan Formats.

A high-level view of the generation of the TDO bit value for the System and Control Paths is shown in Figure 23-14. With the special case for Segmented Scans, a delayed version of the TDO data value is used beginning with the second TDO bit of an SScan2 output-only segment with RDY bits in every Scan Packet in the segment as shown in this figure. This adjustment is also shown in Figure 14-12, although portrayed in a different manner. The need for this adjustment is described in 26.5.1.4.3.



**Figure 23-14 — Conceptual view of TDO data generation for output via the TMSC signal**

## 23.15.2 Specifications

### Rules

- a) Each subsequent specification in 23.15.2 shall only apply to a T4 and above TAP.7, provided the TAP sources the TDO bit value within an SP.
- b) When the System Path is selected, the TDO value associated with TAPC state[k] shall be a logic 1, provided any of the following are true:
  - 1) The TDO data supplied by the STL for this TAPC state is a logic 1.
  - 2) The CLTAPC has not enabled the drive of the TDO signal for this state.

NOTE—Within this document, the CLTAPC enables the drive of the TDO signal with the *sys\_tdo\_oe* signal asserted.

- 3) The ADTAPC state is not *Shift-DR\_f*, and the Scan Format is an SScan Scan Format and a logic 0 otherwise.
- c) When the Control Path is selected, the TDO value associated with TAPC state[k] shall be the value provided by the EPU Path selected for this state as specified by Rule 9.7.2.2 b).

## 23.16 Advanced Protocol effects on the EPU/CLTAPC relationship

### 23.16.1 Description

The use of the Advanced Protocol added with a T4 and above TAP.7 does not in any way affect the operating relationship of the EPU and the CLTAPC. All coupling and decoupling actions inherited from a T3 TAP.7 are not changed. All other aspects of the EPU/CLTAPC relationship are also not changed.

### 23.16.2 Specifications

### Rules

- a) Each subsequent specification in 23.16.2 shall only apply to a T4 and above TAP.7.
- b) The ADTAPC/CLTAPC relationship shall be the same as that of a T3 TAP.7.

## 23.17 SSD detection

### 23.17.1 Description

Scan Selection Directives (SSDs) are detected with the Advanced Scan Formats listed in Rule 23.17.2 b) and are unavailable with the remaining Advanced Scan Formats. The OScan0 and OScan4 Scan Formats are conspicuously absent and do not support SSDs even though they have TDI bits in their SPs. Since these Scan Formats have RDY bits in their SPs, the selection of more than one STL causes a configuration fault, preventing their use with more than one Scan Group Member, a requirement for use in the creation of Series-Equivalent Scans.

### 23.17.2 Specifications

### Rules

- a) Each subsequent specification in 23.17.2 shall only apply to a T4 and above TAP.7.
- b) The support for Scan Selection Directives shall be governed by Table 23-10.

**Table 23-10 — SSD detection with a T4 and above TAP.7**

Scan format	TAP width	
	Narrow	Wide
JScan3	No	Yes
MScan	Yes	Yes
OScan1	Yes	Yes
OScan5	Yes	Yes
Others	No	No

### 23.18 Programming considerations

The use of the Advanced Protocol is invoked by merely storing a value specifying the use of the Advanced Protocol in the Scan Format (SCNFMT) Register. The following registers will be programmed with values compatible with the DTS operation before invoking the use of the Advanced Protocol.

- RDYC      Ready Control
- DLYC      Delay Control

The SREDGE Register will be programmed with a value compatible with TS operation before invoking the use of the Advanced Protocol.

Only one Scan Group Member and only one Scan Group Candidate are permitted prior to specifying the use of a single user Advanced Scan Format (OScan0, OScan4, and any SScan Scan Format. Rule 23.8.2 e) prevents a transition from the use of the JScan0–JScan2 Scan Formats directly to the use of any of the single user scan formats listed in this rule.

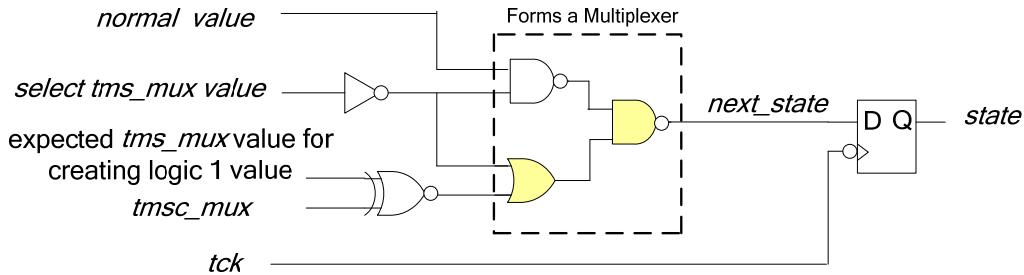
A transition from the use of a JScan0–JScan2 Scan Format to the OScan1 Scan Format (where the STL of interest is made the Scan Group to Member) followed by a transition to the use of one of the single user scan formats listed in this rule is recommended. This also provides a means to verify the TAP.7 Controller is selected and there is scan path continuity before using a Scan Format with RDY bits (creating the potential of a RDY hang in the case an ADTAPC does not respond).

### 23.19 An approach to implementing a TAP.7 Controller with maximum performance

In many cases, the processing of the Advanced Protocol may be performed using a registered TMSC value. However, in some cases, an unregistered version of the TMSC signal will be used in combinational logic determining the next state of a state machine or flag. An example of this is the generation of the Type-3 Reset with CP, SP, and TP Reset Directives. Care should be taken to implement this logic in a manner where it does not become the limiting factor limiting the TAP.7 Controller operating frequency.

The approach shown in Figure 23-15 may be used to minimize the number of logic levels from the TMSC pin to a Flip-Flop where both logic 1 and logic 0 values of TMSC affect the state of a flag or state machine. This logic is in series with the logic creating the programmable use of TMSC rising or falling-edge timing (creating the *tmsc\_mux* signal).

The NAND and OR-NAND combination shaded in this figure form a multiplexer controlled with a signal that specifies whether the *tms\_mux* signal value is used as the next state of the Flip-Flop (value). The control logic is designed to select the use of the *tms\_mux* signal. The XNOR gate connected to the input of OR-NAND gate is used to create the next state value in a programmable manner. When the signal selecting the *tms\_mux* signal is activated, a companion signal defines the *tms\_mux* signal value needed to create a logic 1 or a logic 0 *next state* signal value.



**Figure 23-15 — Using the TMSC value in next state equations**

## 24. MScan Scan Format

This clause is applicable to a T4 and above TAP.7. It provides the rules, permissions, and recommendations for the implementation of the MScan Scan Format. It describes the SP Payload and Delay Elements created by the *SPA* state associated with this scan format. It also describes Undirected CID Allocation using the MScan Scan Format.

The subject matter within this clause is described in the following order:

- 24.1 Capabilities
- 24.2 High-level operation
- 24.3 Scan Packet content
- 24.4 Payload Element
- 24.5 Delay Element
- 24.6 Advancing the TAPC state
- 24.7 CID allocation
- 24.8 Increasing STL performance with the MScan Scan Format
- 24.9 An approach to implementing the MScan Scan Format
- 24.10 Where to find examples

### 24.1 Capabilities

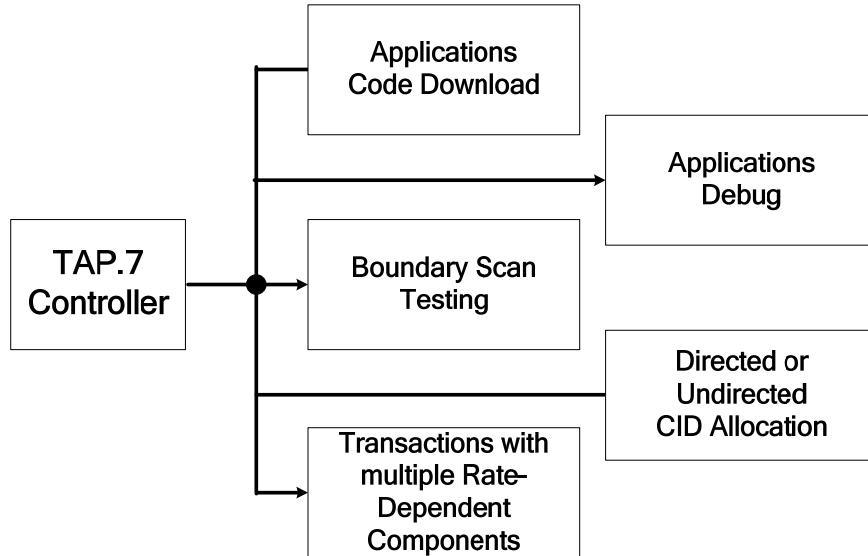
#### 24.1.1 Primary purpose

The MScan Scan Format is extremely flexible in supporting EMTAPCs within the STL having IEEE 1149.1-Specified and IEEE 1149.1-Non-disruptive Behavior, especially STL containing IP with an unorthodox Test Clock behavior. It maximizes the information transferred in an SP Payload Element to accomplish this objective.

#### 24.1.2 Application types supported

The MScan Scan Format supports all of the application types shown in Figure 24-1. It is the most flexible scan format as:

- The STL may stall the completion of each SP.
- TDI and TDO signal information is exchanged for every TAPC state.
- It operates with either a DTS- or a TS-sourced TCK signal.



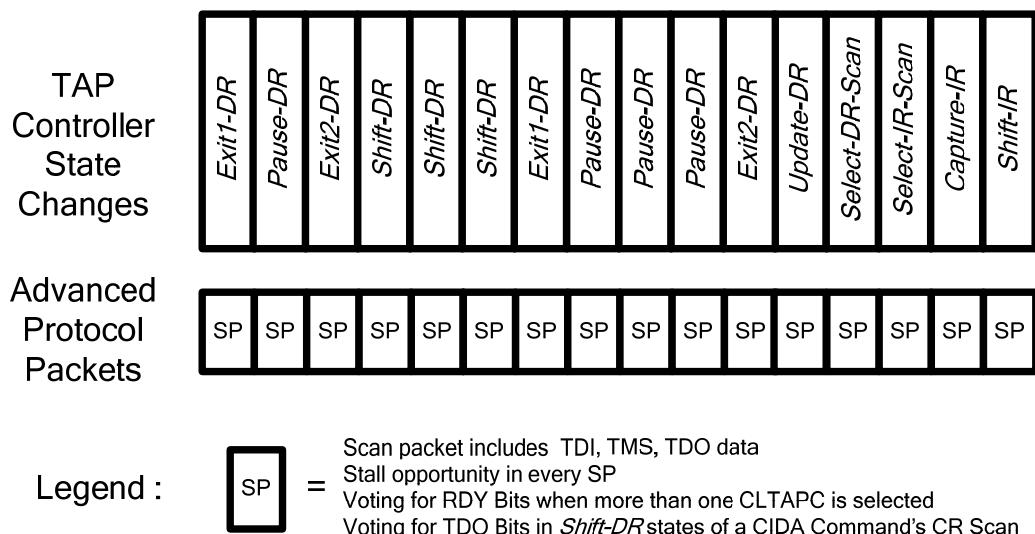
**Figure 24-1 — Application types supported by the MScan Scan Format**

#### 24.1.3 Important characteristics

The MScan Scan Format provides for the use of RDY bits independently of the number of TAP.7 Controllers with a coupled CLTAPC. This is accomplished using the voting process to create the value of RDY bits. The SP Payload Element completes when there are no TAP.7 Controllers voting to stall the payload completion. An STL with a selected CLTAPC may stall the completion of an SP indefinitely. The MScan Scan Format is the only scan format supporting both Directed and Undirected CID Allocation.

#### 24.2 High-level operation

The high-level operation of the MScan Scan Format is shown in Figure 24-2.

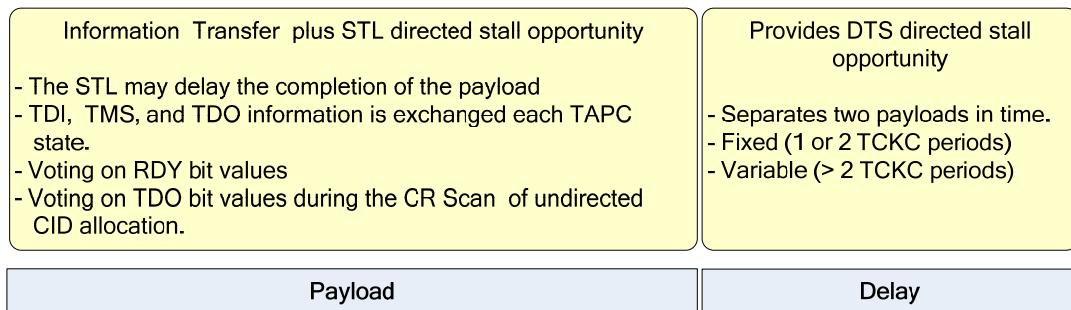


**Figure 24-2 — High-level operation of the MScan Scan Format**

## 24.3 Scan Packet content

### 24.3.1 Description

The SPs created while using the MScan Scan Format contain a Payload Element. A Delay Element follows the Payload Element when the DLYC Register is nonzero. A brief description of these SP Elements and their functions are shown in Figure 24-3.



**Figure 24-3 — Scan Packet Elements with the MScan Scan Format**

### 24.3.2 Specifications

#### Rules

- Each subsequent specification in 24.3.2 shall only apply to a T4 and above TAP.7 when using the MScan Scan Format.
- Scan Packets shall not contain a Header Element.
- Scan Packets shall contain a Payload Element.
- Scan Packets shall contain a Delay Element when all the following are true:
  - The DLYC Register value is nonzero.
  - The SP does not precede a CP.
- An EOT Escape shall be ignored when it occurs coincidentally with any bit of an SP.

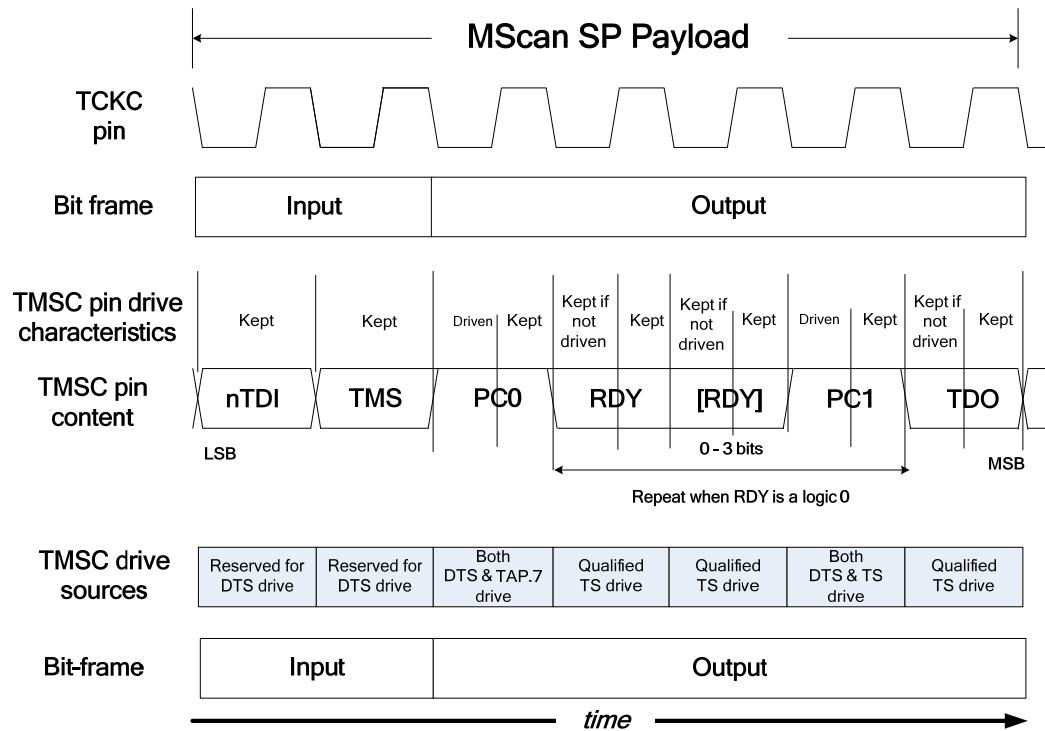
NOTE—The *CPA* state does not follow the *SPA* state provided the previous SP was not associated with the *Update-DR* TAPC state of Command Part Two.

## 24.4 Payload Element

### 24.4.1 Description

#### 24.4.1.1 Format

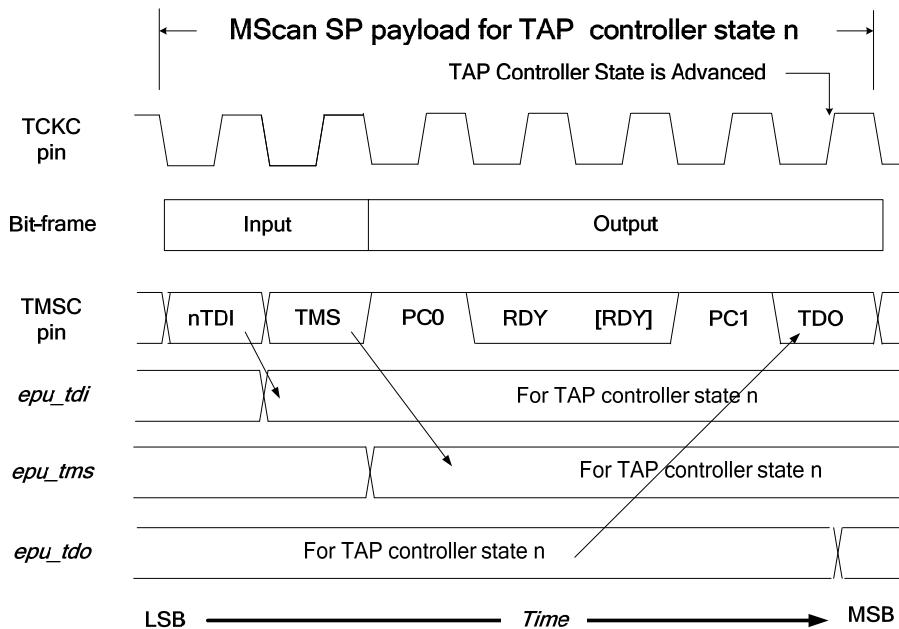
The MScan SP Payload Element is the same for every TAPC state. This Payload Element is created with the template shown in Figure 24-4. The input bit-frame of an MScan SP contains an nTDI bit (the inverted TDI value) and a TMS bit (the true TMS value). A minimal output bit-frame contains four bits (PC0, RDY, PC1, and TDO). The TDO bit is data, while the PC0, RDY, and PC1 bits are control information. One to four RDY bits, all with the same value, occur between the PC0 and PC1 precharge bits. The RDY/PC1 bit series is repeated until the value of the RDY bit(s) is a logic 1. The TMSC Signal Drive Policies for the output frame are described in Clause 14.



**Figure 24-4 — MScan payload template**

#### 24.4.1.2 Relationship to EPU signals

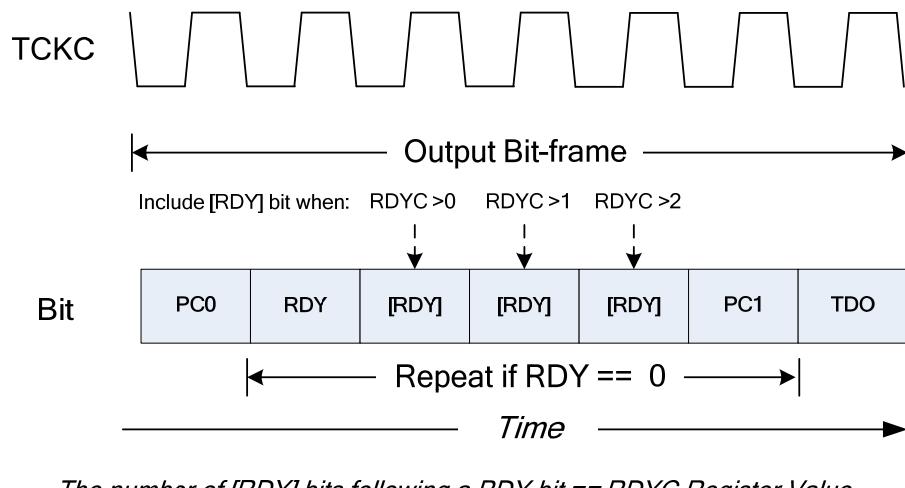
Figure 24-5 shows the association of the MScan SP payload information with the EPU's TDI, TMS, and TDO signals.



**Figure 24-5 — MScan payload/EPU input and output relationships**

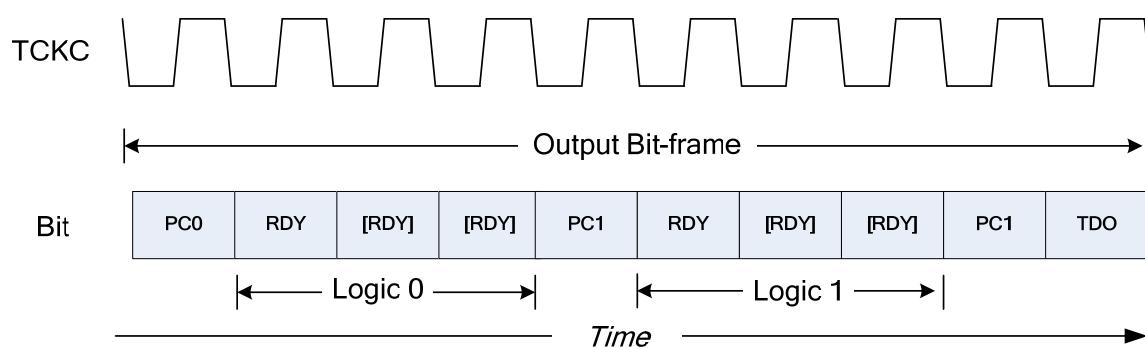
### 24.4.1.3 RDY bits

The RDYC Register value specifies the number of [RDY] bits (one to three) following the RDY bit shown in Figure 24-6. This programmability provides the DTS with additional time to monitor the value of the RDY bit before it determines whether the group of RDY bits following the PC0 bit is repeated when analog delays in the return of RDY and TDO bits become a factor with higher TCK(C) operating frequencies.



**Figure 24-6 — MScan output bit-frame**

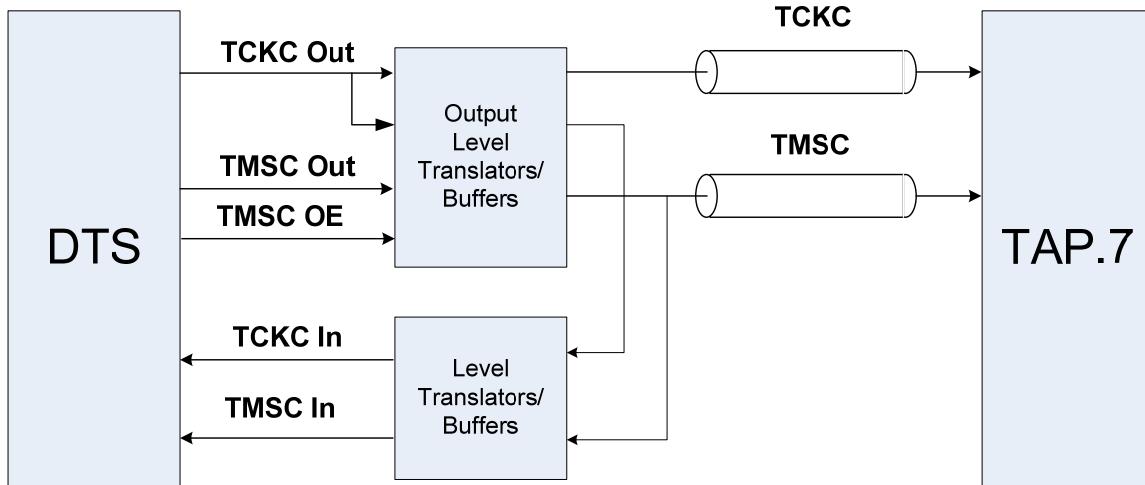
The value of all RDY and [RDY] bits following a PCx bit is the same. When the value of these RDY bit(s) are a logic 0, the RDY/PC1 bit sequence is repeated following the PC1 bit in the sequence as shown in Figure 24-7. This stalls the completion of the SP payload. When these RDY bit(s) are a logic 1, PC1 and TDO bits follow the last RDY or [RDY] bit in the sequence.



**Figure 24-7 — MScan output bit-frame with stalled completion**

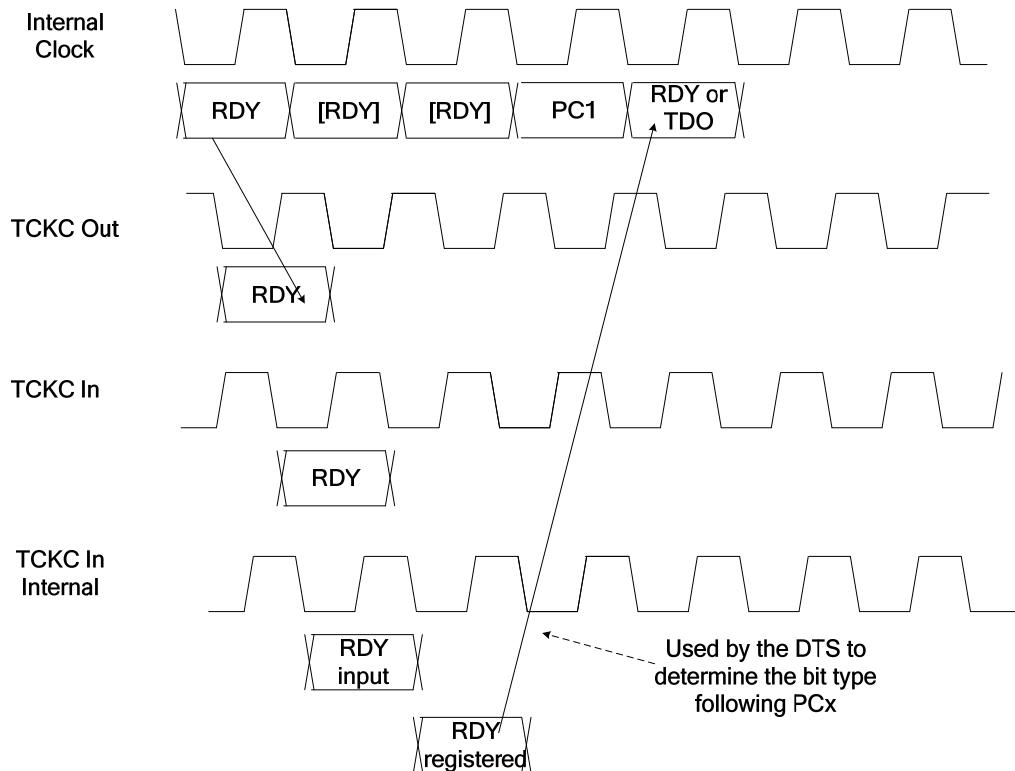
The use of more than one RDY bit following the PCx bit provides a means to operate a system with the highest TCKC signal frequency when an EMTAPC within the STL requires the delay of the payload completion. Additional RDY bits accommodate the propagation delays between the internal DTS clocks and the TAP.7 shown in Figure 24-8. These propagation delays create the DTS/TS timing relationships shown in Figure 24-9. A sufficient number of [RDY] bits will be added to ensure the DTS can view the RDY bit value in time to generate the bit type following the PC1 bit following the [RDY] bits. It is likely that most DTS implementations will be designed to ascertain this information at the beginning of the PC1

bit. Any DTS implementation will be capable of ascertaining this information in time to create the nTDI bit that follows the TDO bit.



**Figure 24-8 — Typical TCKC and TMSC signal system delays**

The timing at the various points within Figure 24-8 is shown in Figure 24-9.



**Figure 24-9 — Use of [RDY] bit(s) to accommodate propagation delays**

An alternative approach to handling these propagation delays may be used when the DTS sources the TCKC signal. The DTS may be designed to stop the TCKC signal for some number of internal DTS TCKC clock periods to stretch the RDY bit period sufficiently to accommodate these propagation delays. In this case, the RDYC value is set to zero.

## 24.4.2 Specification

### Rules

- a) Each subsequent specification in 24.4.2 shall apply only a T4 and above TAP.7 when using the MScan Scan Format.
- b) The SP payload format shall be governed by the template shown in Figure 24-4.
- c) The relationships of the SP payload content to the EPU's TDI, TMS, and TDO signals shall be governed by Figure 24-5.
- d) The content of the output bit-frame shall be governed by Figure 24-6.
- e) The drive state and value of all RDY bits between two precharge bits shall be:
  - 1) Defined by the TMSC Signal Drive Policy.
  - 2) The same for each of these RDY bits.
- f) The number or consecutive RDY bits between precharge bits shall be the RDYC Register value plus one.
- g) Rule 24.4.2 h) shall apply to TDO bit periods where the TMSC Drive Policy specifies Single Drive per Rules 14.8.2 b) and 14.8.2 c).
- h) The value of the TDO bit in the payload of SP[k] shall be the TDO value associated with TAPC state [k] (see Rules 23.15.2 b) and c).
- i) When the TAP.7 Controller is Online, an EOT Escape associated with any bit of an SP shall be ignored.

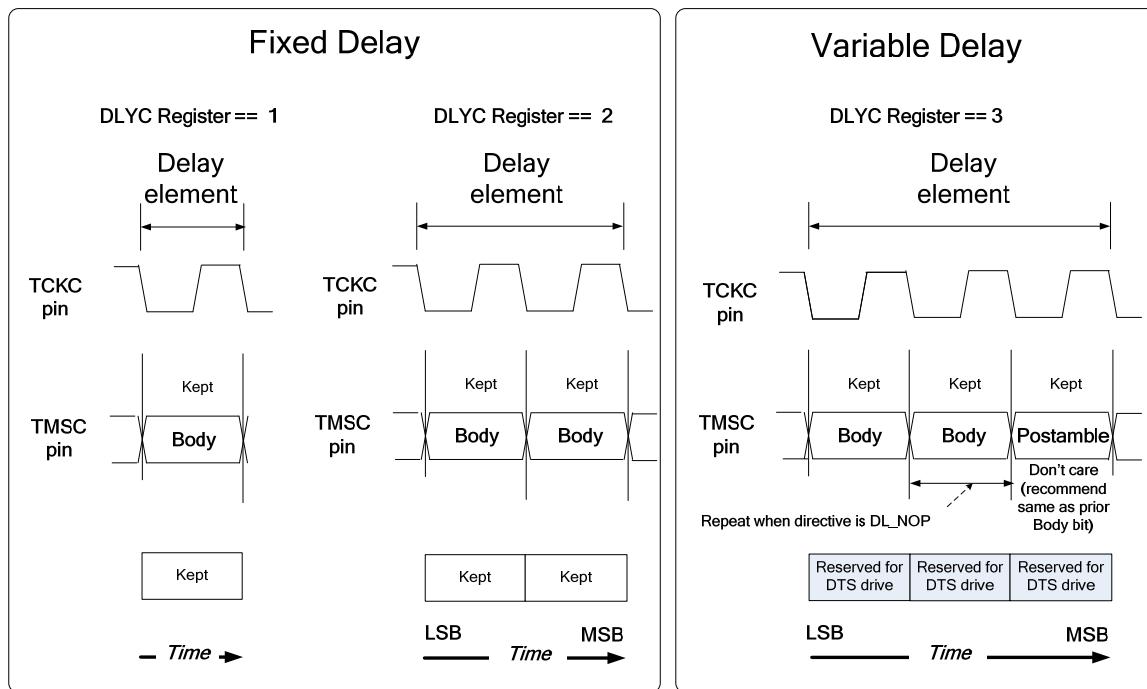
## 24.5 Delay Element

This subclause continues the description of Delay Elements in 22.2.3.3.

### 24.5.1 Description

#### 24.5.1.1 Format

The Delay Element format is shown in Figure 24-10. A variable-length delay is a minimum of three TCKC signal periods. The DTS drives the TMSC signal beginning with the first bit of the Delay Element.



**Figure 24-10 — Delay Element format**

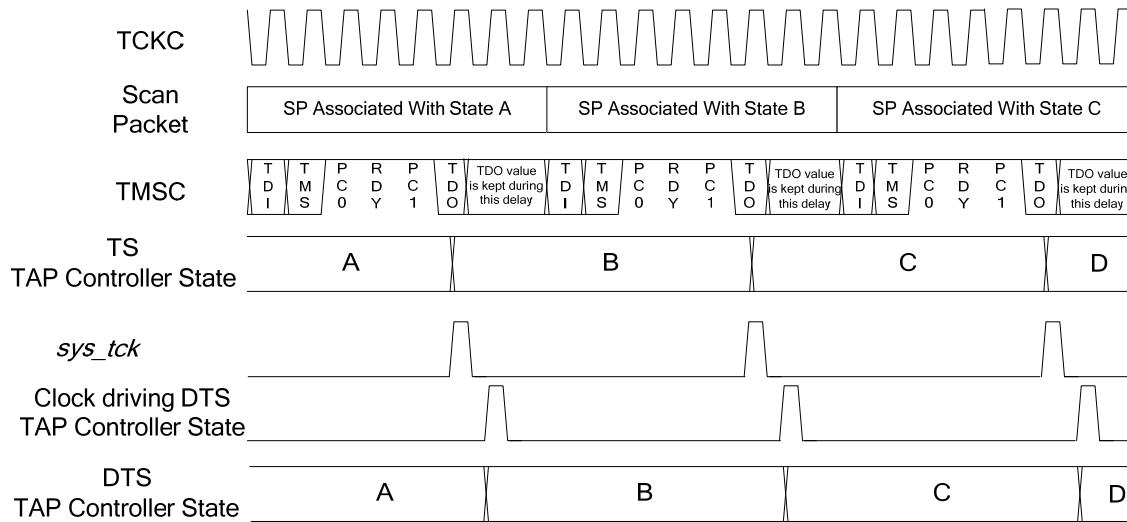
#### 24.5.1.2 Delay Element Directives

As described in 22.2.3.3, a variable delay is terminated by a directive, with these directives shown in Table 24-2. Delay Element Directives are two bits in length with the first bit of the first directive being the first bit of the Delay Element. When the DLYC Register value is three, the number of bits in the Delay Element is as follows:

- Even when:
  - The first bit is a logic 1 and the Delay Element is terminated with a DL\_END Directive
  - The first bit is a logic 0 and the Delay Element is terminated with a DL\_RSO Directive
- Odd when:
  - The first bit is a logic 1 and the Delay Element is terminated with a DL\_RSO Directive
  - The first bit is a logic 0 and the Delay Element is terminated with a DL\_END Directive

#### 24.5.1.3 Uses

An example of the use of Delay Elements with a DTS implementation that does not allow pipelining of TAPC states is shown in Figure 24-11. The use of Delay Elements with a DTS implementation that allows pipelining of TAPC states should not be needed. This type of DTS implementation is recommended as it produces better scan performance.



**Figure 24-11 — MScan SPs with Delay Elements**

### 24.5.2 Specifications

#### Rules

- a) Each subsequent specification in 24.5.2 shall only apply to a T4 and above TAP.7.
- b) When Rule 24.3.2 d) permits the inclusion of a Delay Element in an SP, the Delay Element length shall be governed by Table 24-1.

**Table 24-1 — SP Delay Element characteristics**

DLYC Register value	Delay Element length in TCKC periods	Action
00b	0	Null – no delay
01b	1	Delay of one TCKC period
10b	2	Delay of two TCKC periods
11b	Variable ( $\geq 3$ )	A minimum of three TCKC periods:

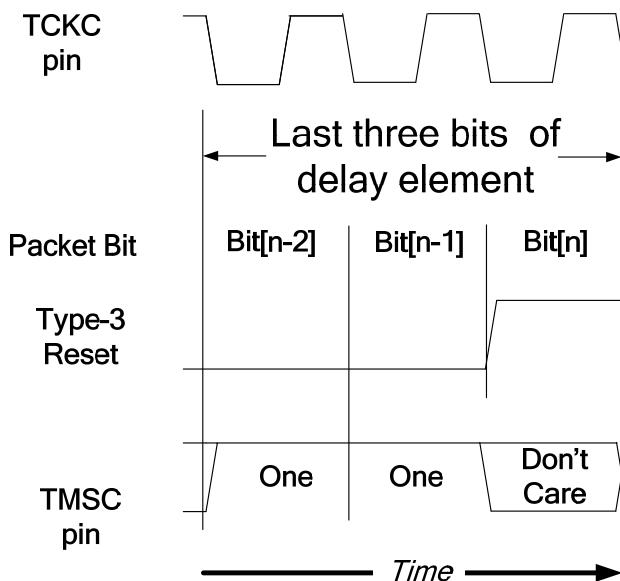
- c) The SP Delay Element format shall be governed by Figure 24-10.
- d) When the SP Delay Element is variable length, the last two Delay Element body bits shall be processed as a Delay Element Directive.
- e) When the SP Delay Element is variable length, the Delay Element Directives described by Rule 24.5.2 d) shall be interpreted as shown in Table 24-2.

**Table 24-2 — Variable-length SP Delay Element Directives**

Delay Element bit pairs		Action
Mnemonic	MSB LSB	
DL_END	00b	Delay is complete, postamble bit follows.
DL_NOP	01b, 10b	No operation, the Delay Element is extended one bit period.
DL_RSO	11b	Delay body is complete, the postamble bit follows, generates a Type-3 Reset (see Rules 24.5.2 f) and 24.5.2 g).

NOTE—A logic 0 postamble bit is recommended following a DL\_END Directive. A logic 1 postamble bit is recommended following a DL\_RSO Directive.

- f) A Type-3 Reset shall be generated by a Delay Element, provided all of the following are true:
  - 1) The DLYC Register value specifies a variable-length Delay Element ( $DLYC = 3$ ).
  - 2) A DL\_RSO Directive occurs.
- g) A Type-3 Reset generated by a variable-length Delay Element shall have the timing characteristics shown in Figure 24-12.



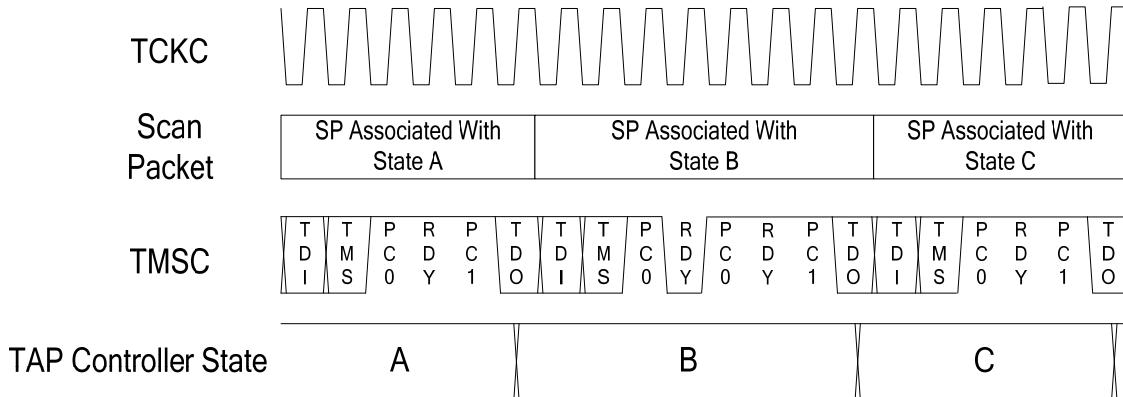
**Figure 24-12 — Type-3 Reset generated by a variable-length Delay Element**

- h) Delay Elements shall not alter the point at which the TAPC state is advanced.

## 24.6 Advancing the TAPC state

### 24.6.1 Description

An MScan SP payload causes the advance of the TAPC state progression coincidently with the TCKC signal rising edge occurring within the TDO bit of an output bit-frame as shown in Figure 24-13. The DLYC and RDYC Register values are zero in this figure.



NOTE—A Delay Element may be inserted after the Payload Element but is not shown in this figure. The number of [RDY] bits following an RDY bit can be as many as three but none are shown in this figure.

**Figure 24-13 — MScan SP payload/TAPC state advance relationship**

## 24.6.2 Specifications

### Rules

- Each subsequent specification in 24.6.2 shall only apply to an Online T4 and above TAP.7 when using the MScan Scan Format.
- The ADTAPC state shall be advanced with the rising edge of the TCKC coincident with the TDO bit within the SP payload, provided neither of the following is true:
  - The previous SP was not associated with the *Update-DR* TAPC state of Command Part Two (the *CPA* state does not follow the *SPA* state).
  - The SP is not canceled by a qualified Selection Escape or a qualified Deselection Escape.

## 24.7 CID allocation

### 24.7.1 Description

The MScan Scan Format may be used for both the directed and undirected methods of CID allocation. The allocation criteria, allocation candidates, and the comparison of the externally and internally supplied AT values is the same as for the JScan3 Scan Format described in 20.9. The differences between CID allocation with the JScan3 and MScan formats are as follows:

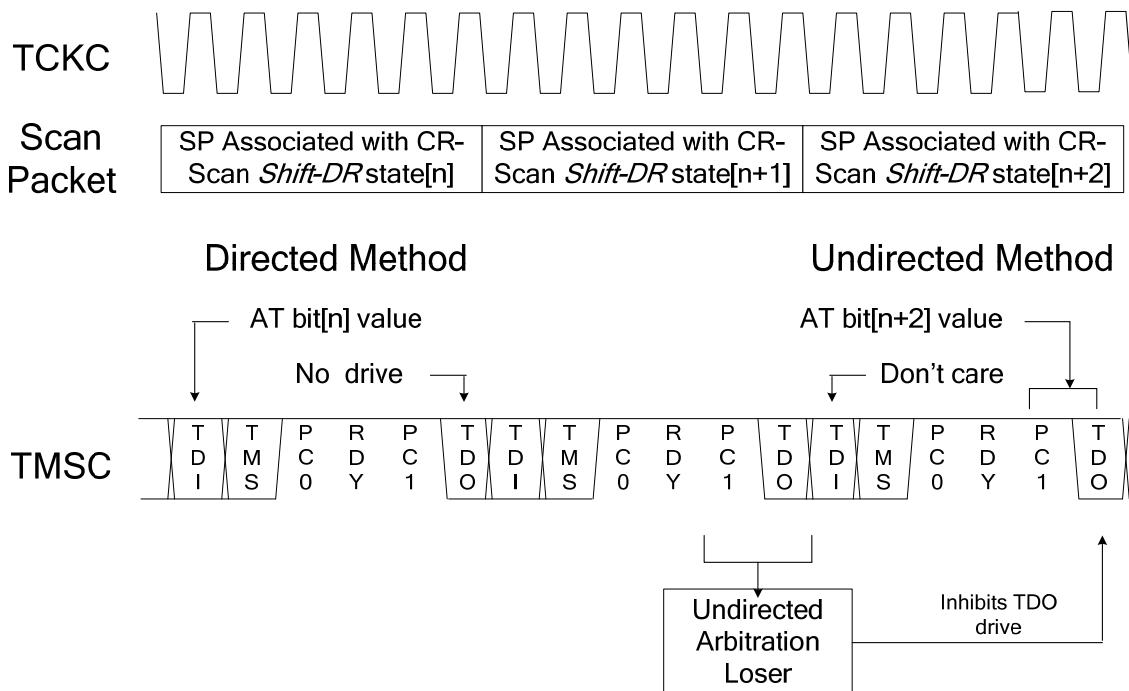
- The method used to convey the AT value to the TAP.7 Controller
- The number of *Shift-DR* states required for Undirected CID Allocation (144 versus 36)

These differences are described as follows.

With the JScan3 Scan Format, the AT value is conveyed with the TDIC signal for Directed CID Allocation and the TDOC signal for Undirected CID Allocation. With the MScan Scan Format, the AT is conveyed with the TMSC signal for both the directed and undirected methods. With the directed method, the AT is delivered with the nTDI bits within the payloads of SPs associated with the *Shift-DR* states of a 36-bit CR Scan. With the undirected method, the AT is created with the TDO bits within the payloads of SPs associated with the *Shift-DR* states of a 36-bit CR Scan. With the undirected method, the PC1 and TDO bits within the CR Scan generated by a CIDA Command provide the precharge and voting on TDO values needed to create the AT required for CID allocation. The voting capability provided by the MScan Scan Format is the reason that it supports Undirected CID Allocation and the OScan and SScan Scan Formats do not. With the

JScan3 Scan Format, a 144-bit CR Scan needed to perform the arbitration with Undirected CID Allocation since the voting facility is built into the operation of the JScan Scan Format. With the MScan Scan Format, exactly 36 AT bits are transferred with the CR Scan for both the directed and undirected methods.

The operation of both Directed and Undirected CID Allocation with the MScan Scan Format is shown in Figure 24-14. With the directed method, the TDI bit within an SP provides the function virtually identical to the function of the TDIC pin using the JScan3 Scan Format. With the undirected method, the PC1 bit period within an MScan output bit-frame provides the function virtually identical to that created with *Shift-DR* state [4n+1] of the CIDA CR Scan using the JScan3 Scan Format. The function of the TDO bit period within an MScan output bit-frame is virtually identical to that created with *Shift-DR* state [4n+3] of the CIDA CR Scan using the JScan3 Scan Format, except the TMSC signal may be driven only when the TCKC signal is low.



**Figure 24-14 — CID allocation with MScan SPs**

With the directed method:

- The nTDI bits within SPs associated with the *Shift-DR* TAPC states of a CR Scan convey the AT value.
- The TDO bits within SPs associated with the *Shift-DR* TAPC states of a CR Scan are a logic 1 as the TMSC signal is not driven during the TDO bit periods of SPs.

With the undirected method:

- The nTDI bits within SPs associated with the *Shift-DR* TAPC states of a CR Scan are ignored (discarded).
- The TMSC signal is driven as a logic 0 within TDO bits associated with the *Shift-DR* TAPC states of a CR Scan, provided the TAP.7 Controller is an arbitration participant and the AT value desired by the TAP.7 Controller for this bit period is a logic 0.

Attempting Undirected CID Allocation with an Advanced Scan Format other than the MScan Scan Format is considered a programming error and will not result in allocation of a CID. In this case, the CIDA Command functions as an NOP and the CID and CIDI Register values are unchanged.

## 24.7.2 Specifications

### Rules

- a) Each subsequent specification in 24.7.2 shall only apply to a T4 and above TAP.7 when the MScan Scan Format is used to allocate CIDs.
- b) The nTDI bits within SP payloads associated with *Shift-DR* states of the CR Scan of a CIDA Command shall be used as the AT value for Directed CID Allocation.
- c) The TDO bits in the SP payload of the MScan Scan Format shall be used as the AT for Undirected CID Allocation.
- d) The number of *Shift-DR* states in the CR Scan of a CID Command shall be exactly 36 for Directed and Undirected CID Allocation to occur.
- e) The allocation criteria, allocation candidates, and the comparison of the externally and internally supplied AT values shall be the same as for the JScan3 Scan Format.
- f) The TDO Bit Drive Policy for bit n during the Shift-DR states of the CIDA CR Scan shall be the same as the drive policy shown in Table 20-9 for bits 4n+3 when using the JScan3 Scan Format.

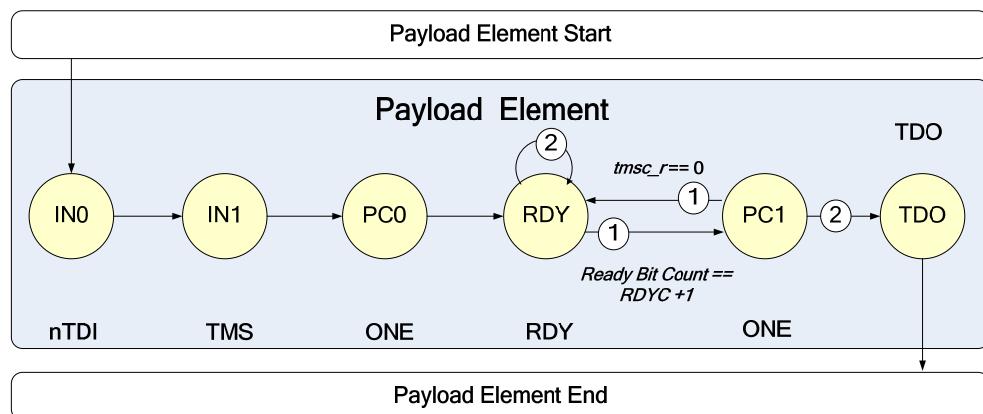
## 24.8 Increasing STL performance with the MScan Scan Format

Performance of the STL may be improved with the MScan Scan Format, in some cases, by adjusting the *sys\_tck* signal duty cycle as described in 23.9. An example of the *sys\_tck* duty cycle with the MScan Scan Format is shown in Figure 23-5.

## 24.9 An approach to implementing the MScan Scan Format

### 24.9.1 Payload State Machine

Figure 24-15 shows a conceptual view of the operation of the MScan SP Payload Element.



#### Value conveyed by the TMSC signal

<b>Legend:</b>	INO - First bit of input frame	<i>tmsc_r</i> - the value of the TMSC signal registered on the falling edge of TCKC
	IN1 - Second bit of input frame	
	PCx - Precharge bits	
	RDY - RDY + [RDY] bits	
	TDO - TDO bit	

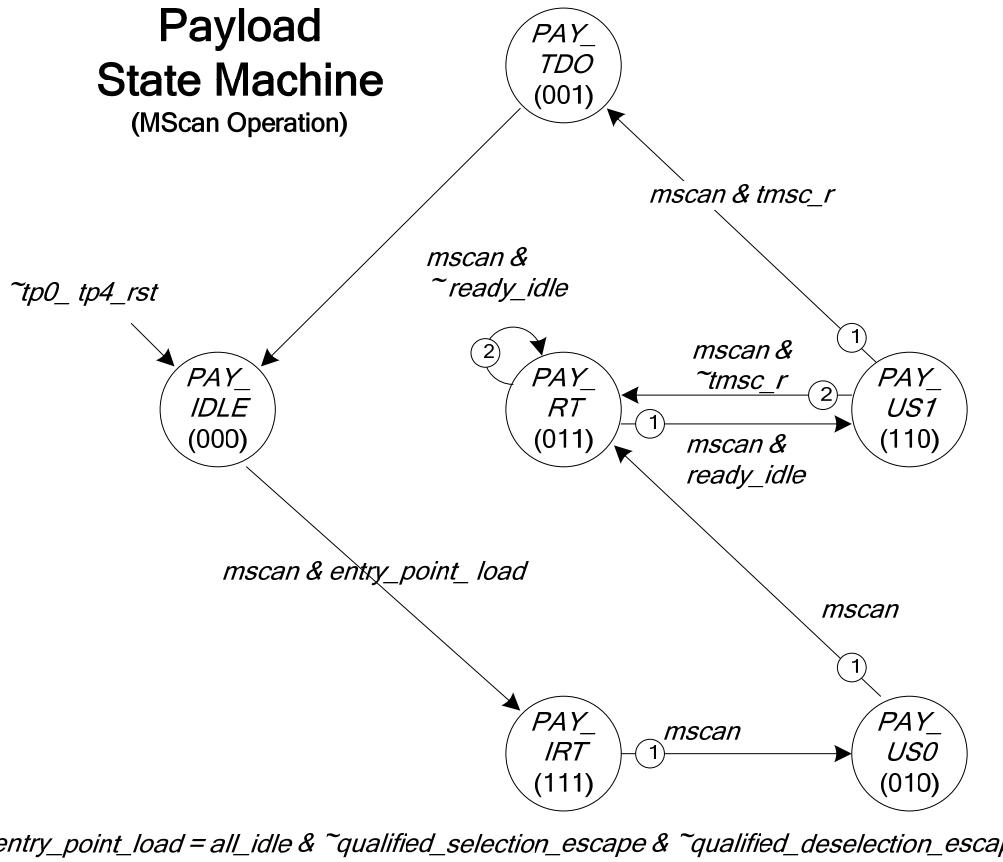
**Figure 24-15 — Conceptual view of MScan SP Payload Element behavior**

An SP Payload State Machine handles the MScan, OScan, and SScan Scan Formats. This machine has eight states. It is assisted with a flag for the OScan6 and SScan Scan Formats and a state machine tracking the end of a Data Segment with the SScan Scan Formats. The MScan Scan Format utilizes six of the eight states as shown in Figure 24-16. The state names and their functions are shown in Table 24-3.

**Table 24-3 — Payload State Machine states for the MScan Scan Format**

Mnemonic	State encoding	Name	Description
PAY_IDLE	000b	Scan Idle	Store the nTDI bit value when the CSM state == ADV and TSM state is IDLE.
PAY_IRT	111b	Scan IN1, RDY, TDO	Store the TMS signal value.
PAY_US0	010b	Scan Utility State Zero	Precharge TMSC signal.
PAY_RDY	011b	Scan Ready	RDY bit on TMSC signal.
PAY_US1	110b	Scan Utility State One	Precharge TMSC signal.
PAY_TDO	001b	Scan TDO	TDO bit on TMSC signal.

The importance of the state encoding will become apparent shortly. It allows relatively simple generation of the state following the *PAY\_IDLE* state for the MScan, OScan, and SScan Scan Formats. With the MScan and OScan Scan Formats, Bit[2] is set when the input frame has two bits: Bit[1] is set when an RDY bit is included in the payload, and Bit[0] is set when a TDO bit is included in the payload. Codes 010b and 110b are utility codes as they indicate an RDY bit without a TDO bit (an illegal combination). With the MScan Scan Format, these codes are used for the states associated with the PC0 and PC1 precharge bits.



**Figure 24-16 — Conceptual view of the Payload State Machine for the MScan Scan Format**

#### 24.9.2 Ready State Machine

A Ready State Machine is added as a helper machine for the Payload State Machine. The conceptual operation of this machine is shown in Figure 24-17. Its states are described in Table 24-4.

**Table 24-4 — Ready State Machine states**

Mnemonic	Name	Description
<i>RDY_THREE</i>	Ready Three	Three ready bits remaining.
<i>RDY_TWO</i>	Ready Two	Two ready bits remaining.
<i>RDY_ONE</i>	Ready One	One ready bit remaining.
<i>RDY_IDLE</i>	Ready Idle	No ready bits remaining.

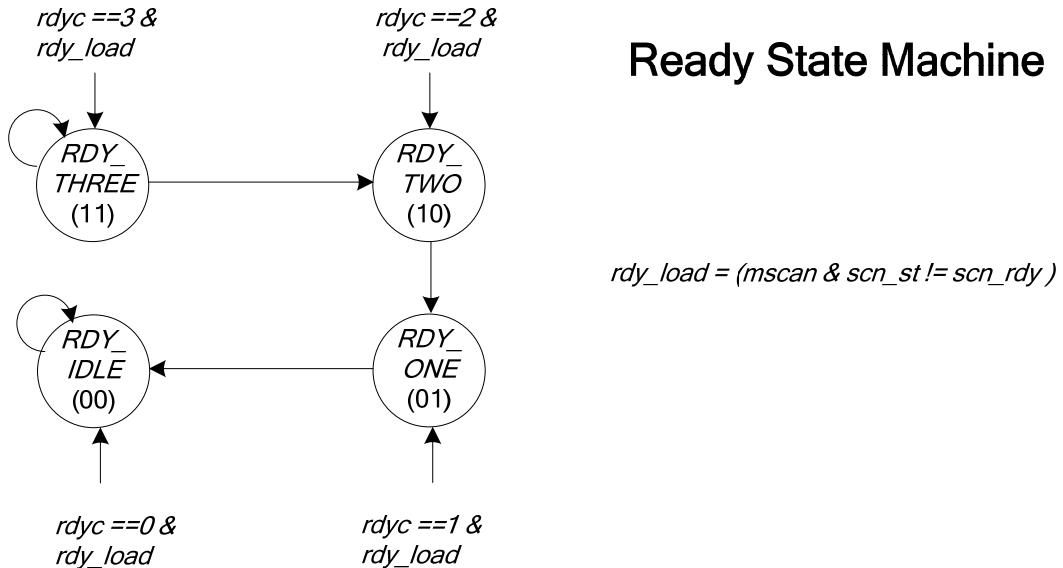


Figure 24-17 — Conceptual view of Ready State Machine MScan operation

#### 24.9.3 Delay State Machine

The conceptual operation of the Delay Element is shown in Figure 24-18.

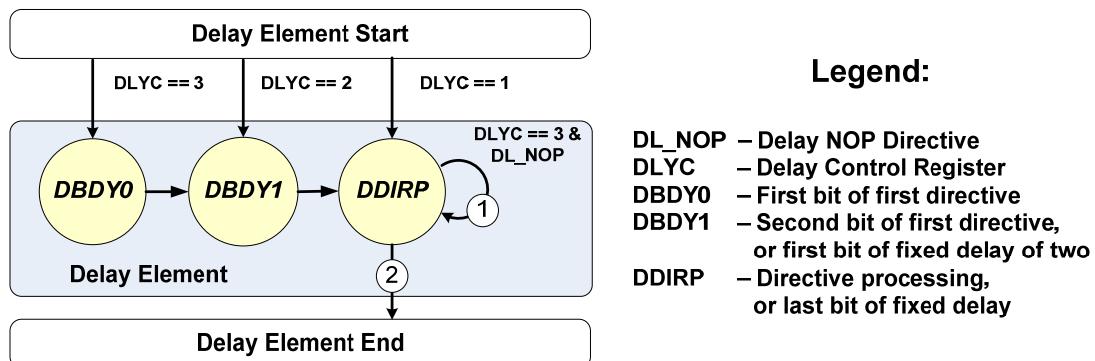


Figure 24-18 — Conceptual Delay Element operation

With a delay of one or two TCKC signal periods, there are no delay directives. In this case, the DBDY0 and DBDY1 states provide the appropriate delay and the Delay Element ends automatically. With a variable delay (DLYC == 3), the last two body bits are decoded as delay directives as shown in Table 24-2. The DBDY0 and DBDY1 bits input the first Delay Directive, with the directive being used during the DDIRP state.

The operation of the Delay State Machine is shown in Figure 24-19. It operates in parallel with the Payload State Machine as shown in Figure 24-16. Its states are described in Table 24-5.

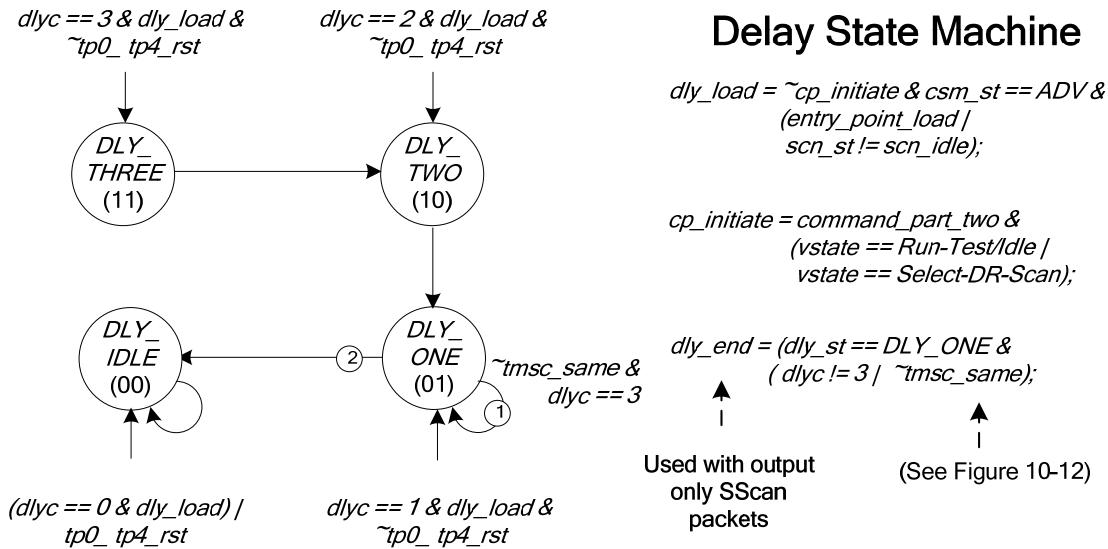


Figure 24-19 — Conceptual view of the Delay State Machine

Table 24-5 — Delay State Machine state

Mnemonic	Name	Description
DLY_THREE	Delay Three	Three or more delay bits remaining.
DLY_TWO	Delay Two	Two or more delay bits remaining.
DLY_ONE	Delay One	One or more delay bits remaining.
DLY_IDLE	Delay Idle	No delay bits remaining.

## 24.10 Where to find examples

Examples of MScan scan transfers are shown in Annex B and Annex C. Annex B contains an MScan timing diagram, while Annex C contains tabular examples of a three-bit DR Scan followed by a three-bit IR Scan while using the MScan Scan Format. These transactions are the basis for all examples. A CP following an SP is also shown. The use of delays is also described. These examples may be used to compare the Advanced Protocol's behavior using each of these scan formats with and without the use of a delay. The point at which the TAP.7 Controller state is advanced is also shown.

## 25. OScan Scan Formats

This clause is applicable to T4 and above TAP.7s. It provides the rules, permissions, and recommendations for the implementation of the OScan Scan Formats. It describes the SP Payload and Delay Elements created by the *SPA* state while using the OScan Scan Formats.

The subject matter within this clause is described in the following order:

- 25.1 Capabilities
- 25.2 High-level operation
- 25.3 Scan Packet content
- 25.4 Payload Element
- 25.5 Delay Element
- 25.6 Advancing the TAPC state
- 25.7 CID allocation
- 25.8 Increasing STL performance with OScan Scan Formats
- 25.9 An approach to implementing OScan Scan Formats
- 25.10 Where to find examples

### 25.1 Capabilities

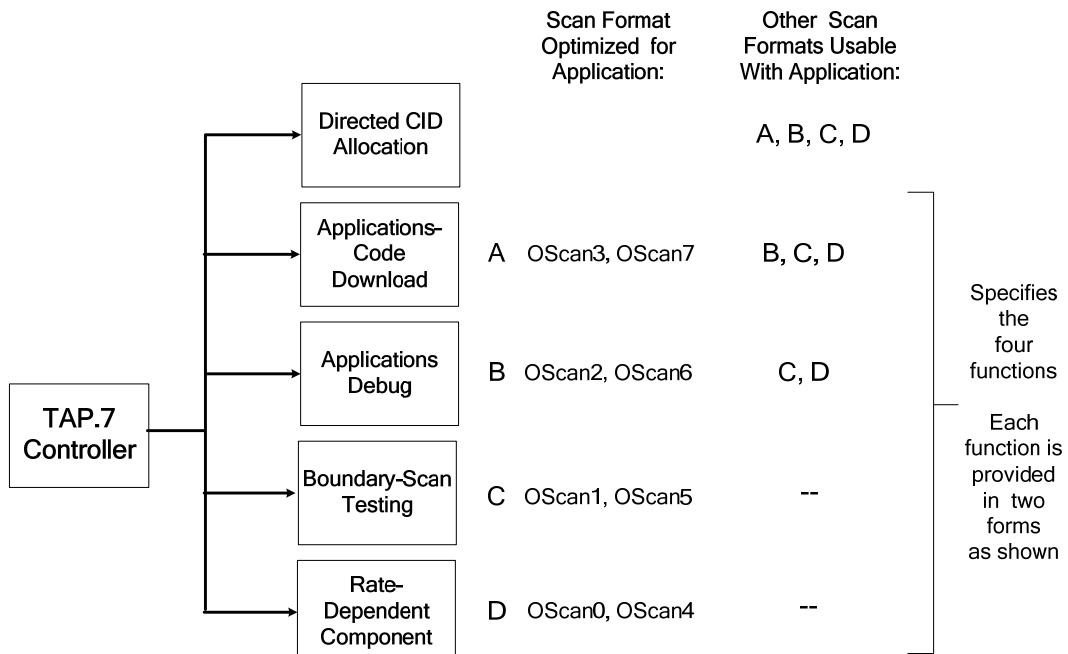
#### 25.1.1 Primary purpose

The OScan Scan Formats provide a means for the DTS to use its knowledge of the application to increase scan performance. The DTS can use this knowledge to transfer only the data needed by the application in both *Shift-xR* and non-*Shift-xR* states. The optimizations provided for use with debug applications are more extensive than those provided for test applications.

#### 25.1.2 Application types supported

The OScan Scan Formats support the four types of scan transactions shown in Figure 25-1. These applications are described briefly as follows:

- |                             |   |
|-----------------------------|---|
| — Application-code download | Requires TDI data in <i>Shift-xR</i> states and no stalls   |
| — Application debug         | Requires TDI and TDO data in <i>Shift-xR</i> states and no stalls   |
| — Boundary-scan testing     | Requires TDI data in <i>Run-Test/Idle</i> and <i>Pause-xR</i> states for the use of Scan Selection Directives |
| — Scan Selection Directives | TDI and TDO data in <i>Shift-xR</i> states  |
| — Rate-dependent component  | Requires stalls and TDI and TDO data in <i>Shift-xR</i> states  |



NOTE-The OScan4 - OScan7 Scan Formats provide more efficient scan transfers but may only be used with a DTS-sourced TCKC as they utilize EOT escape sequences.

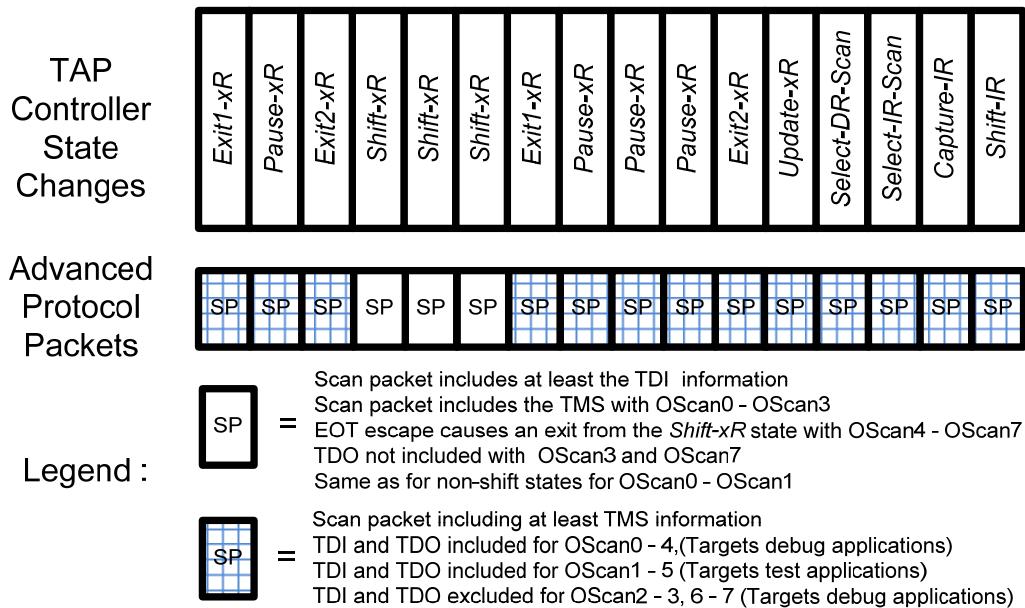
**Figure 25-1 — Application types supported by OScan Scan Formats**

### 25.1.3 Important characteristics

The OScan SPs do not support voting on output values as there are no precharge bits in the OScan SPs. This provides SPs whose minimum size ranges from one to four bits. This makes the least efficient OScan SP 150% more efficient than the MScan SP. Since voting is not supported, the TAP.7 Controller is placed Offline when an OScan Scan Format supporting stalls (with RDY bits) is used and there is either the potential of more than one Scan Group Member when the *Run-Test/Idle state* is reached, or there is more than one Scan Group Member. These states are declared Configuration Faults as stated previously.

## 25.2 High-level operation

The high-level operation of the OScan Scan Formats is shown in Figure 25-2.

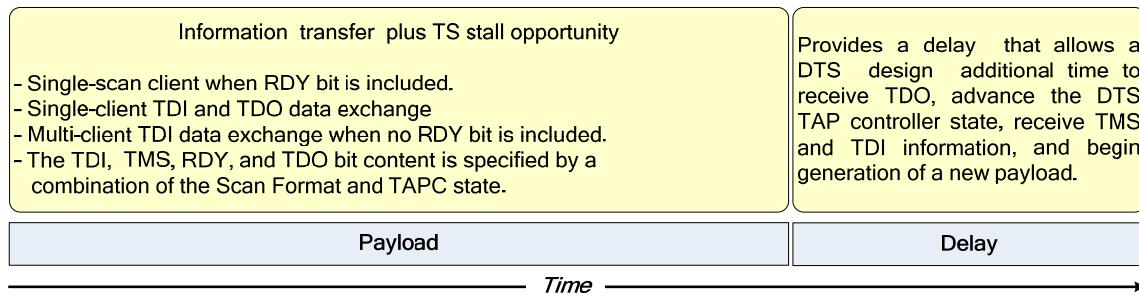


**Figure 25-2 — High-level operation of the OScan Scan Formats**

## 25.3 Scan Packet content

### 25.3.1 Description

The SPs created while using an OScan Scan Format contains a Payload Element. A Delay Element follows the Payload Element when the DLYC Register is nonzero. A brief description of these SP Elements and their functions is shown in Figure 25-3.



**Figure 25-3 — Scan Packet Elements with the OScan Scan Format**

### 25.3.2 Specifications

#### Rules

- a) Each subsequent specification in 25.3.2 shall only apply to a T4 and above TAP.7 when the OScan Scan Format defines the SP payload content.
- b) The SP shall not contain a Header Element.
- c) The SP shall contain a Payload Element.
- d) SPs shall contain a Delay Element when all the following are true:

- 1) The DLYC Register value is nonzero.
- 2) The SP does not precede a CP.

NOTE—The *CPA* state does not follow the *SPA* state provided the previous SP was not associated with the *Update-DR* TAPC state of Command Part Two.

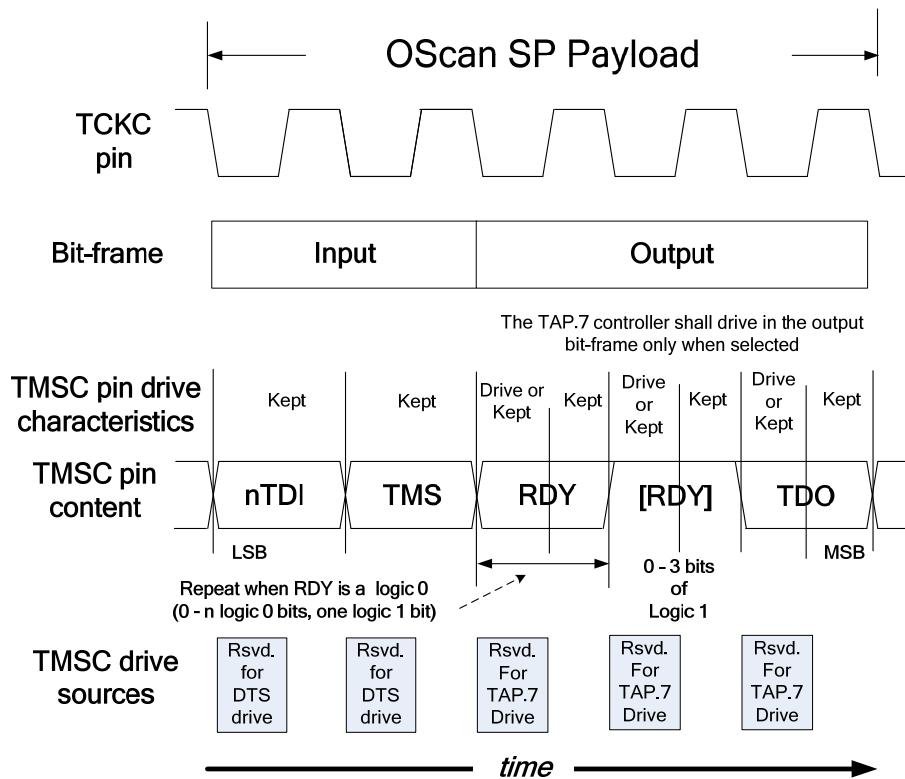
- e) An EOT Escape shall be ignored when it occurs coincidentally with any bit in the SP other than the nTDI bits specified as supporting an EOT Escape in Table 25-1.

## 25.4 Payload Element

### 25.4.1 Description

#### 25.4.1.1 Format

An OScan SP Payload Element is created using the template shown in Figure 25-4.



**Figure 25-4 — OScan payload template**

The Payload Elements for the eight OScan Scan Formats are created by deleting none, one, or more than one bit from this template. Either the nTDI or TMS bit or both of these bits are included in the input bit-frame. The RDY and TDO bits may be included in the output bit-frame or deleted entirely. The TDO bit is included in the Payload Element when an RDY bit is included. Table 25-1 governs the SP payload content. This content changes for *Shift-xR* and non-*Shift-xR* TAPC states with some of the scan formats. Scan formats with minimal information (only TMS bits) in non-*Shift-xR* states are more efficient for debug. These scan formats do not support all test functions as they lack the TDI information required for the use of SSDs.

### 25.4.1.2 Optimizations

OScan Scan Formats replicate four functions in two forms:

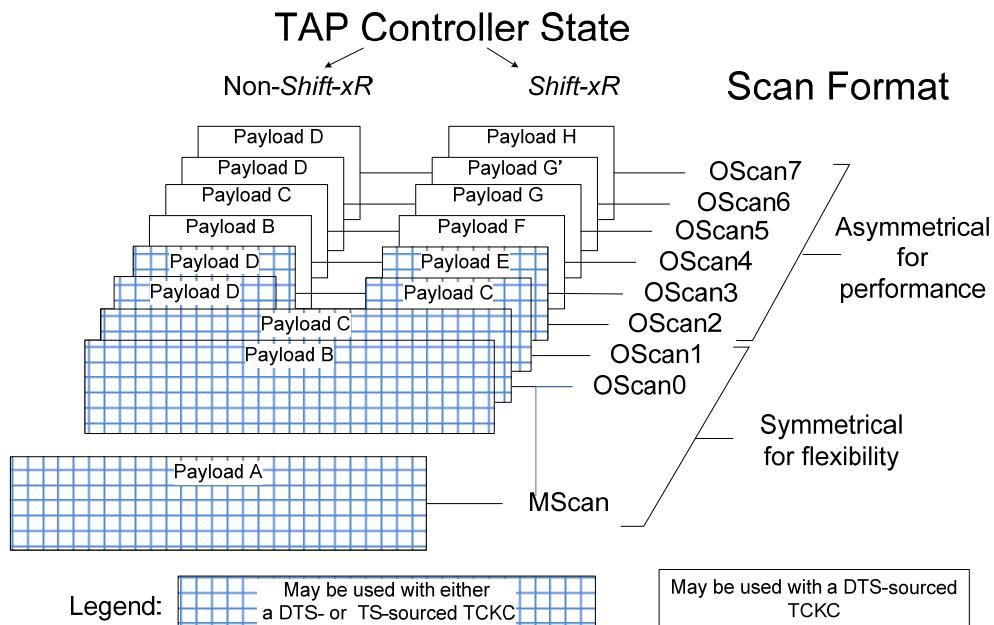
- OScan0–OScan3      Will be used when the Test Clock is sourced by the target system
- OScan4–OScan7      Provide the best performance for a Test Clock sourced by the DTS

The OScan0–OScan3 Scan Formats have a TMS bit included in the SP and may be used with either a DTS- or a TS-sourced Test Clock. The SP payloads associated with non-*Shift-xR* states are optimized more extensively than those associated with *Shift-xR* states.

The OScan4–OScan7 Scan Formats are more efficient but can only be used with a DTS-sourced TCKC signal. The SP payloads associated with non-*Shift-xR* states are optimized in the same manner as with the OScan0–OScan3 Scan Formats. The SP payloads associated with *Shift-xR* have additional optimizations. An SP associated with the *Shift-xR* state does not include a TMS bit. Instead, an EOT Escape overlaid on an nTDI bit causes an exit from the *Shift-xR* state. The state remains *Shift-xR* otherwise. A look at a 32-bit DR Scan shows the impact of this optimization. A total of 96 bits are required when each SP associated with a *Shift-xR* state includes a TMS bit, while a total of 64 bits are required when a TMS is not included in these SPs. Assuming the EOT Escape adds the equivalent of four bit periods, the number of TCK signal periods for this transfer is 68/96 or roughly 70% of the total number needed without this optimization.

With the OScan2–OScan3 and OScan6–OScan7 Scan Formats, the TDI signal information and output bit-frame are deleted from the SPs associated with non-*Shift-xR* TAPC states. The number of TCK signal periods for this transfer is 33% of the total number needed without this optimization. This compares an SP payload with only TMS bits to an SP payload with only nTDI, TMS, and TDO signal information.

The SP payload content is modulated by the scan format and the TAPC state for some scan formats as shown in Figure 25-5.



NOTE—The Payload type shown in Figure 25-5 is correlated with payload (PAY) designations in Table 25-1.

**Figure 25-5 — Payloads for *Shift-xR* and non-*Shift-xR* TAPC states**

The OScan optimizations are applied in a hierarchical manner:

- Optimization I: OScan1 and OScan5      Both RDY and [RDY] bit(s) are deleted from all SPs
- Optimization II: OScan2 and OScan6      Optimization I + nTDI and TDO bits are deleted from SPs associated with non-*Shift-xR* TAPC states
- Optimization III: OScan3 and OScan7      Optimization II + TDO bits are deleted from SPs
- Optimization IV: OScan4 through OScan7      TMS bits are deleted from SPs associated with *Shift-xR* TAPC states

The SP Payload Element content produced by these optimizations is shown in Table 25-1. The mandatory OScan0 and OScan1 Scan Formats create symmetrical payloads for *Shift-xR* and non-*Shift-xR* states. This symmetry disappears with OScan2–OScan7 Scan Formats. The optimizations provided by these scan formats affect the Payload Element content differently for *Shift-xR* and non-*Shift-xR* states.

**Table 25-1 — OScan SP Payload Element contents**

Scan format	Virtual non- <i>Shift-xR</i> TAPC states						Virtual <i>Shift-xR</i> TAPC states					
	Pay	Input bit-frame		Output bit-frame			Pay	Input bit-frame		Output bit-frame		
OScan0	B	nTDI	TMS	RDY	[RDY]	TDO	B	nTDI	TMS	RDY	[RDY]	TDO
OScan1	C	nTDI	TMS	—	—	TDO	C	nTDI	TMS	—	—	TDO
OScan2	D	—	TMS	—	—	—	C	nTDI	TMS	—	—	TDO
OScan3	D	—	TMS	—	—	—	E	nTDI	TMS	—	—	—
OScan4	B	nTDI	TMS	RDY	[RDY]	TDO	F	<i>nTDI</i>	—	<i>RDY</i>	<i>[RDY]</i>	TDO
OScan5	C	nTDI	TMS	—	—	TDO	G	<i>nTDI</i>	—	—	—	TDO
OScan6	D	—	TMS	—	—	—	G'	<i>nTDI</i>	<i>[PAD]</i>	—	—	TDO
OScan7	D	—	TMS	—	—	—	H	<i>nTDI</i>	—	—	—	—

Pay == Payload type

[RDY]—See Rule 25.4.2 l)

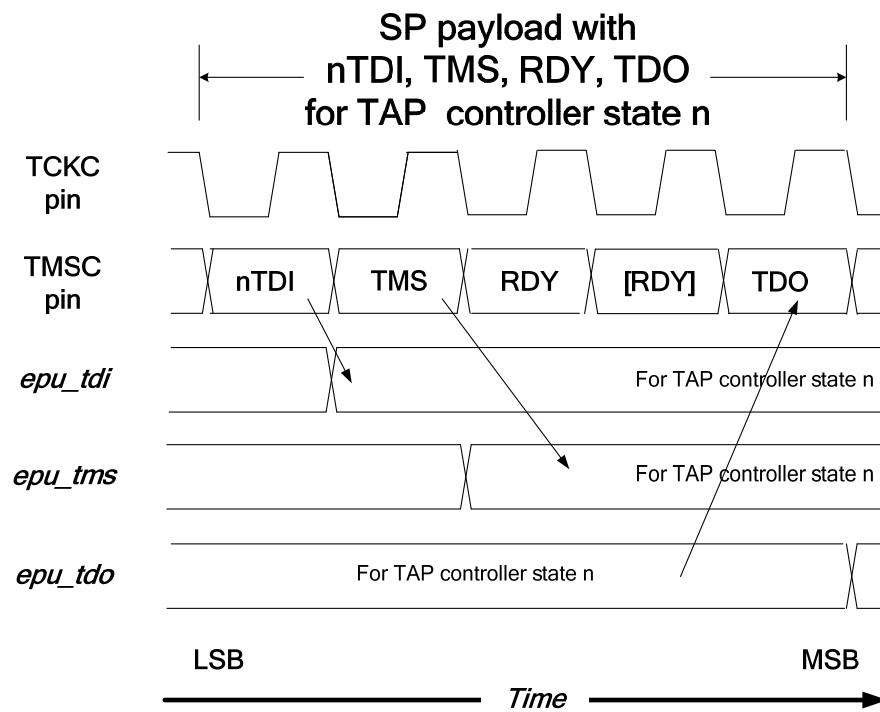
[PAD]—See Rule 25.4.2 g)

NOTE—The entries in this table and similar tables in Clause 26 are defined for the virtual TAPC States.

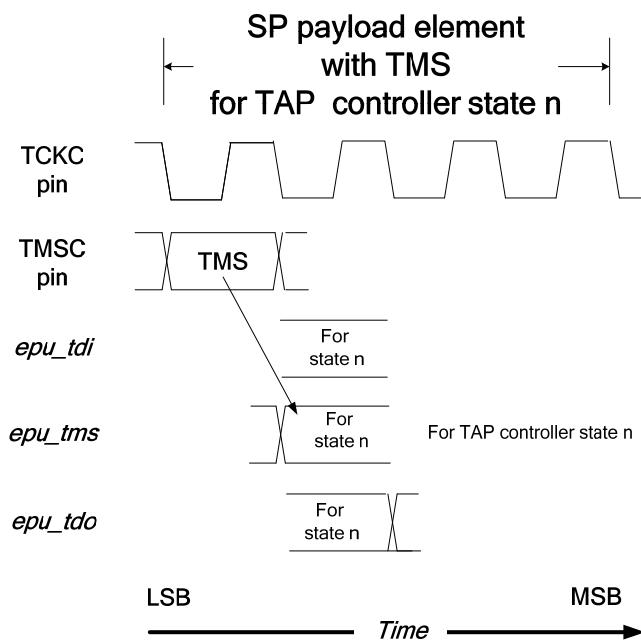
The TMS signal value used to direct the TAPC state progression for OScan4–OScan7 in *Shift-xR* TAPC states is developed as stated in Rule 25.4.2 f).

### 25.4.1.3 Relationship to EPU signals

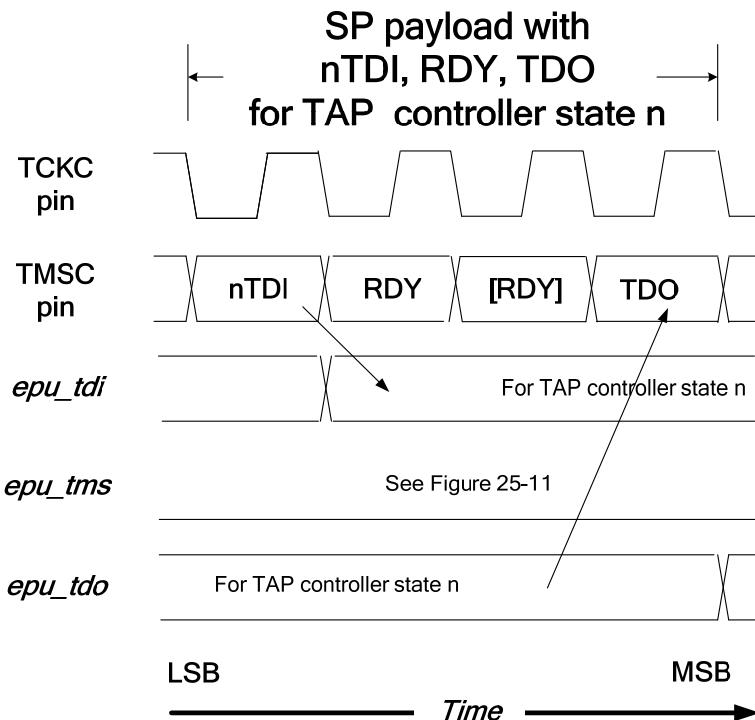
Figure 25-6 through Figure 25-10 show the relationship of OScan SP payload information with the EPU's TDI, TMS, and TDO signals for input frames with nTDI and TMS, only TMS, and only nTDI signal values. Figure 25-11 shows the generation of the TMS signal value for OScan4–OScan7 Scan Formats with an EOT (custom) Escape for an SP payload associated with the *Shift-xR* state.



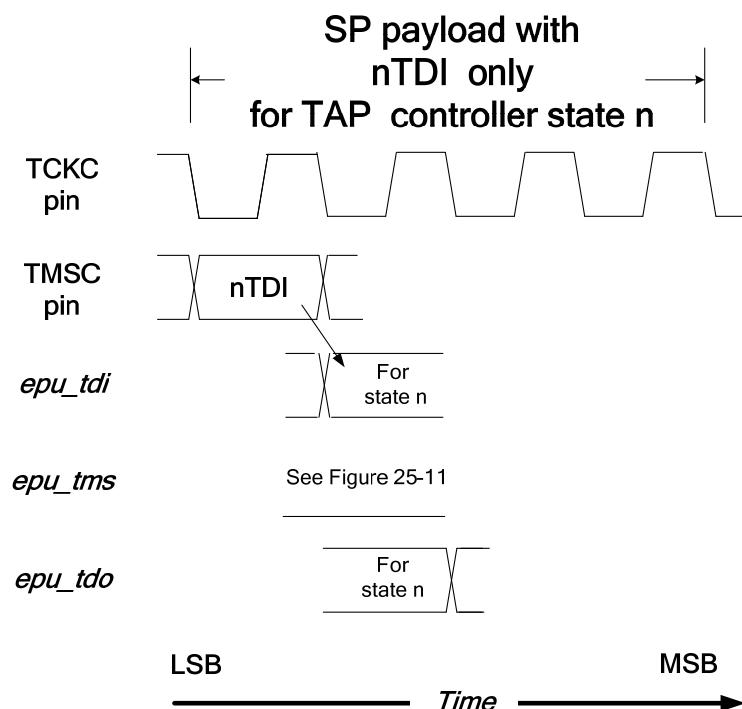
**Figure 25-6 — EPU input/output relationships with an nTDI, TMS, RDY, and TDO SP payload**



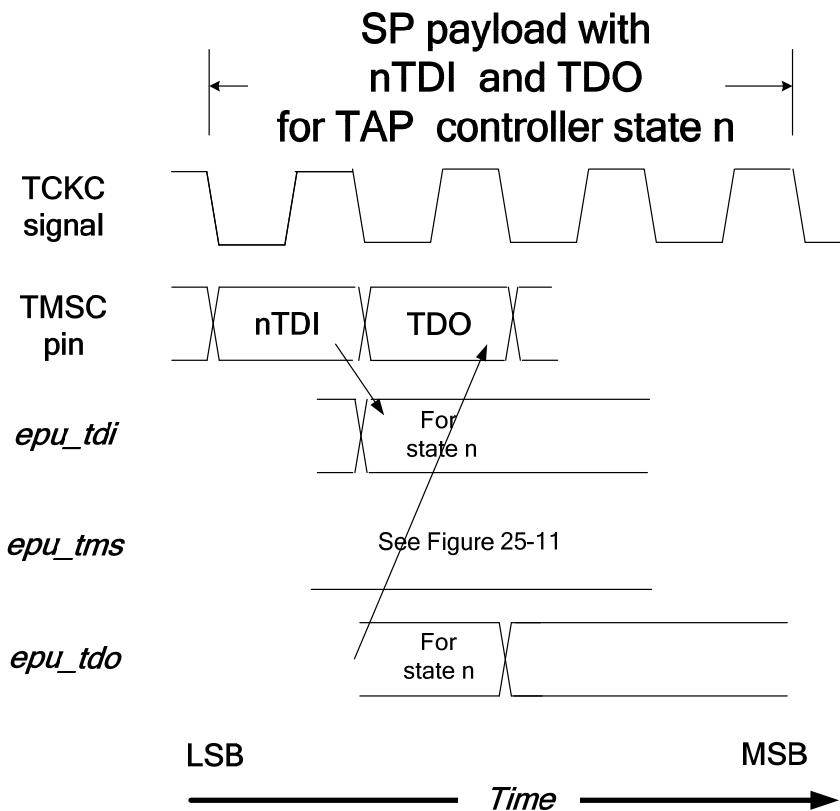
**Figure 25-7 — EPU input/output relationships with a TMS SP payload**



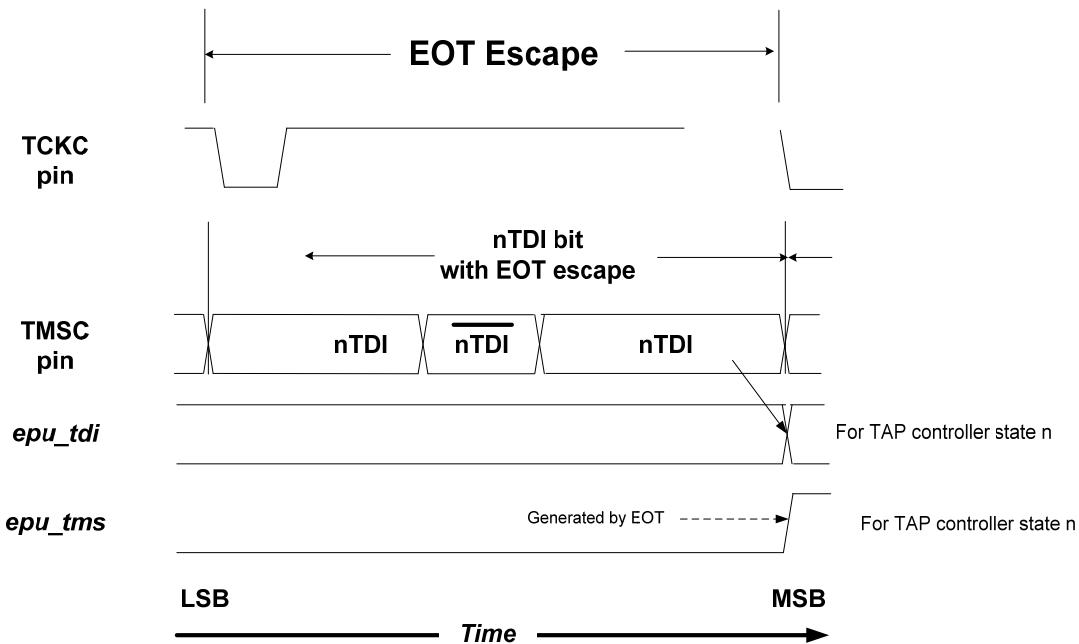
**Figure 25-8 — EPU input/output relationships with an nTDI, RDY, and TDO SP payload**



**Figure 25-9 — EPU input/output relationships with an nTDI SP payload**



**Figure 25-10 — EPU input/output relationships with an nTDI/TDO SP payload**



**Figure 25-11 — TMS generation with an EOT Escape**

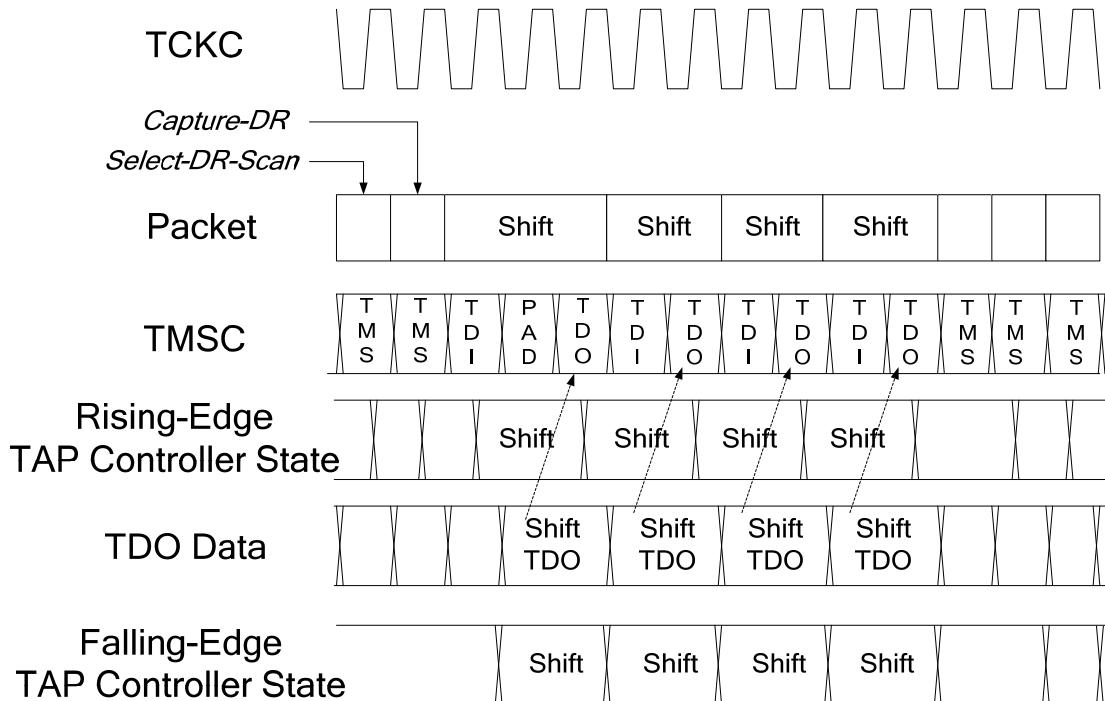
#### 25.4.1.4 Input bit-frame

The input bit-frame consists of:

- An inverted TDI signal value (excepted in D type payload)
- A true TMS signal value (excepted in F-H types payload)
- A PAD bit, in the case of the OScan6 Scan Format upon *Shift-xR* entry (G type payload)

With OScan Scan Formats, the input bit-frame is present and contains an nTDI bit, a TMS bit, or both of these bits. It contains the PAD bit (in replacement for TMS bit) when the OScan6 Scan Format is used. When the nTDI bit is not included in the bit-frame, the TDI signal value presented to the TAPCs is a logic 1. When the TMS signal value presented to the TAPCs for *Shift-xR* TAPC states is generated with an EOT Escape, the value of this signal is a logic 0 until an EOT Escape coincident with an nTDI bit generates a logic 1 TMS value.

With the OScan6 Scan Format, the PAD bit is added to allow time for the creation of the *epu\_tdo* signal value needed for the first *Shift-xR* TAPC state following the *Capture-xR* or *Exit2-xR* states. The DTS may source the [PAD] bit as a logic 1, a logic 0, or choose to not drive the TMSC signal during this bit period. The function of the PAD bit is shown in Figure 25-12. This bit does not affect the TAPC state progression and will not be used for any purpose.



**Figure 25-12 — OScan6 PAD bit and its function**

#### 25.4.1.5 Drive types

Clause 14 covers the drive policy for the OScan Scan Formats. The TDO bit value is described by Rule 23.15.2 c). The output bit-frame utilizes three types of drive: Single, Joint, and Inhibited. Recall that Inhibited Drive is used when there appears to be one Scan Group Member and this CLTAPC is not a Scan Group Member. Single drive is used when there appears to be one Scan Group Member and the STL of this

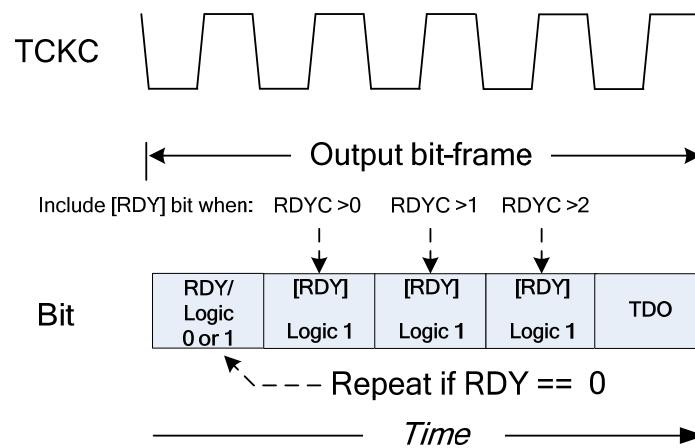
TAP.7 Controller is the Scan Group Member. Joint drive is used in two instances, when there appears to be no Scan Group Members and when there appears to be more than one Scan Group Member.

#### 25.4.1.6 RDY bits

An SP payload that permits the stall of SP payload progression contains all of:

- One or more logic 0 RDY bits
- One logic 1 RDY bit
- A number of logic 1 [RDY] bits that is equal to the RDYC Register value

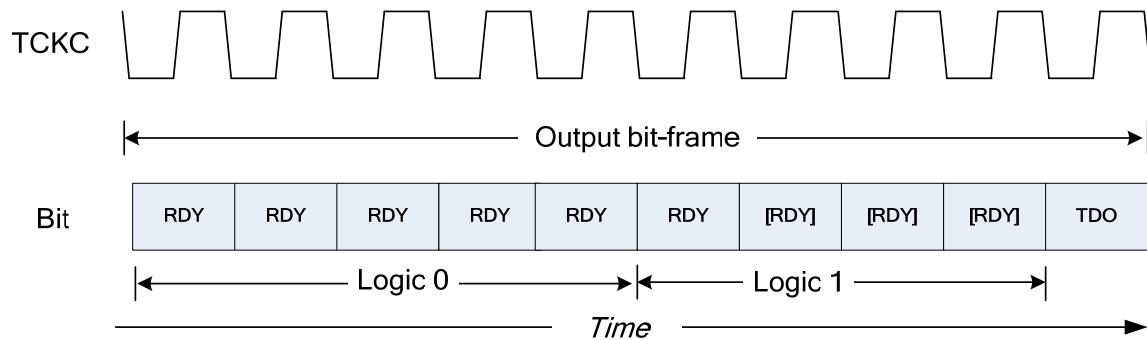
With OScan and SScan Scan Formats, the RDY bit is repeated when it is a logic 0 as shown in Figure 25-13. From zero to three RDY bits follow a logic 1 RDY bit, with this specified by the RDYC Register value. As described previously, this programmability provides to the DTS additional time to monitor the value of the RDY bit when analog delays in the return of RDY and TDO bits become a factor with higher TCK(C) operating frequencies.



The number of [RDY] bits following a RDY bit == RDYC Register Value

**Figure 25-13 — OScan output bit-frame**

An example of an output bit-frame with the CLTAPC (STL) stalling its completion is shown in Figure 25-14.



RDYC == 3, the STL is not ready to complete the SP payload when the output bit-frame begins

**Figure 25-14 — OScan/SScan bit-frame with stalled completion**

## 25.4.2 Specifications

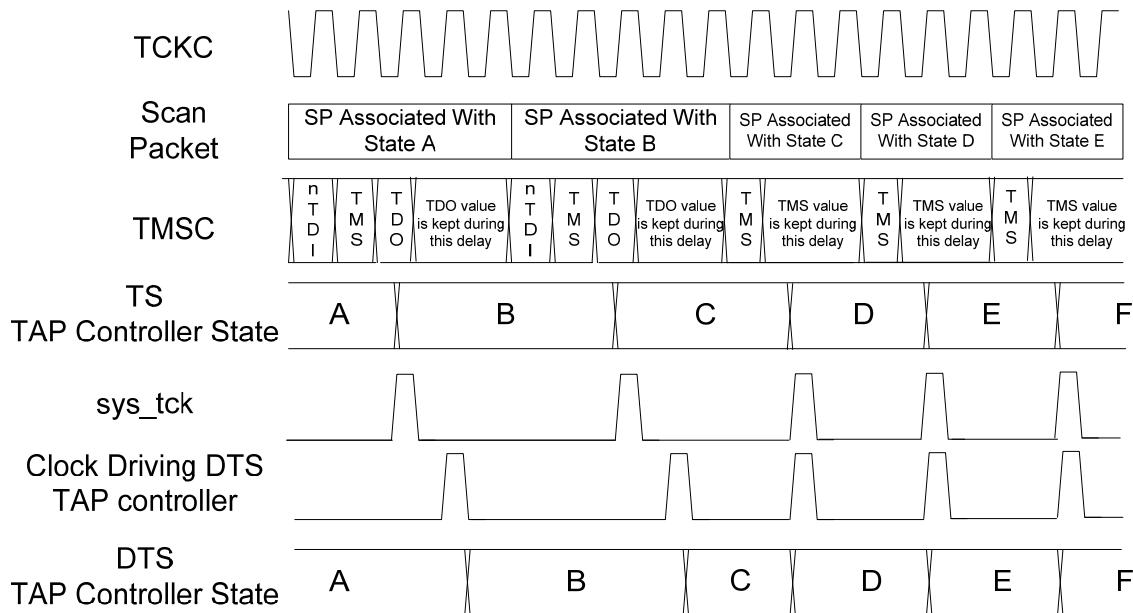
### Rules

- a) Each subsequent specification in 25.4.2 shall only apply to a T4 and above TAP.7 when using an OScan Scan Format unless stated otherwise.
- b) The SP Payload Element format shall be governed by Figure 25-4.
- c) The inclusion of bits within the input and output bit-frames in the SP Payload Element shown in Figure 25-4 shall be governed by Table 25-1.
- d) When bits are deleted from the SP Payload Element, the order of the remaining bits shown in Figure 25-4 and Table 25-1 shall remain the same.
- e) When the TAP.7 Controller is Online, the value of the EPU TDI signal associated with a virtual TAPC state shall be determined by the SP payload associated with the virtual TAPC state as shown in Table 25-1 and be one of the following:
  - 1) The inverse of the value of the nTDI bit when this bit is present in the SP Payload Element.
  - 2) A logic 1 when the nTDI bit is not present in the SP Payload Element in the SP Payload Element.
- f) When the TAP.7 Controller is Online, the value of the EPU TMS signal associated with a virtual TAPC state shall be determined by the SP payload associated with the virtual state as shown in Table 25-1 and be one of the following:
  - 1) The value of the TMS bit when this bit is present in the SP Payload Element.
  - 2) Logic 0 when all of the following are true:
    - i) The TMS bit is not present in the SP Payload Element.
    - ii) An EOT Escape is not associated with the nTDI bit in the SP Payload Element.
  - 3) A logic 1 when all of the following are true:
    - i) The TMS bit is not present in the SP Payload Element.
    - ii) An EOT Escape is associated with the nTDI bit in the SP Payload Element.
- g) The [PAD] bit shown in Table 25-1 shall be included in the SP Payload Element processing when all of the following are true:
  - 1) The scan format is OScan6.

- 2) The SP is associated with a virtual *Shift-xR* state.
  - 3) The prior SP was associated with a virtual state other than the virtual *Shift-xR* state.
- h) The [PAD] bit shown in Table 25-1 shall convey no information to the TAP.7 Controller or other circuitry other than being a placeholder in the SP Payload Element.
- i) The SP Payload Element of the SP preceding a CP shall be processed as having the content shown in Table 25-1 for the virtual non-*Shift-xR* state associated with the scan format.
- j) The timing relationships of the SP Payload Element to the EPU TDI, TMS, and TDO signals shall be governed by the following figures:
- 1) Figure 25-6.
  - 2) Figure 25-7.
  - 3) Figure 25-8.
  - 4) Figure 25-9.
  - 5) Figure 25-10.
  - 6) Figure 25-11.
- k) An EOT Escape occurring within an SP shall be ignored unless all the following are true:
- 1) It occurs coincidentally with an nTDI bit within the SP payload.
  - 2) The SP is associated with a *Shift-xR* state.
  - 3) The scan format is OScan4, OScan5, OScan6, or OScan7.
- l) When an SP contains RDY bit(s) within its SP payload, all of the following shall apply:
- 1) The number of logic 0 RDY bits that precede the first logic 1 RDY bit shall be greater than or equal to zero (see Figure 25-13).
  - 2) The number of logic 1 [RDY] bits shall be equal to the RDYC Register value plus one (see Figure 25-13).
- m) Rule 25.4.2 n) shall apply to TDO bit periods where the TMSC Drive Policy specifies Single Drive per Rules 14.8.2 b) and 14.8.2 c).
- n) The value of the TDO bit in the payload of SP[k] shall be the TDO value associated with TAPC state [k] (see Rules 23.15.2 b) and c).

## 25.5 Delay Element

Delay Elements follow the OScan SP payloads when the DLYC Register is nonzero. The purpose and operation of Delay Elements is the same as for the MScan Scan Format. The operation of the Delay Elements with the OScan SP payloads with and without output bit-frames is shown in Figure 25-15. The state labels in this figure are described in Table 25-2.



**Figure 25-15 — Using delays with the OScan Scan Formats**

**Table 25-2 — Labels in Figure 25-15/TAPC state cross-reference**

Label	Represents TAPC state
<b>A-B</b>	<i>Shift-xR</i>
<b>C</b>	<i>Exit1-xR</i>
<b>D-F</b>	<i>Pause-xR</i>

## 25.6 Advancing the TAPC state

### 25.6.1 Description

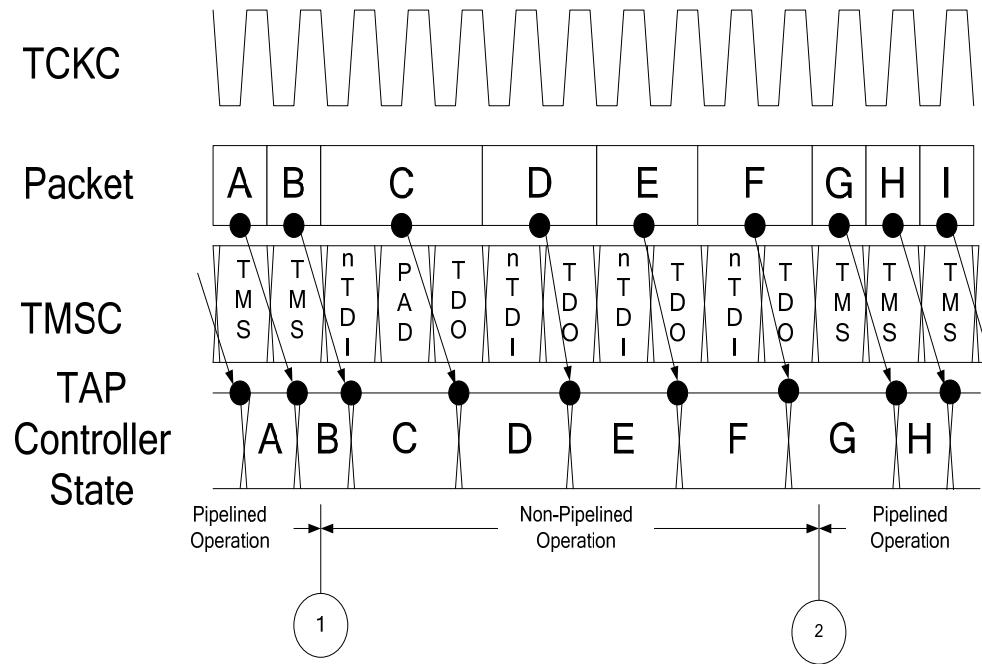
An OScan SP payload causes the advance of the TAPC state progression coincidentally with the TCKC signal rising edge occurring coincidently with:

- The TDO bit of the SP Payload Element
- The TCKC signal period following the completion of the SP Payload Element when there is no TDO bit in the SP Payload Element

These relationships are shown in Figure 25-16. The SPs and TAPC state transitions they produce are highlighted by the black dots. The TAPC state and the SP associated with it have the same letters. Note that some Scan Packets contain the TDO bit and others do not. This affects the position of the TAPC state advance relative to the SP payload.

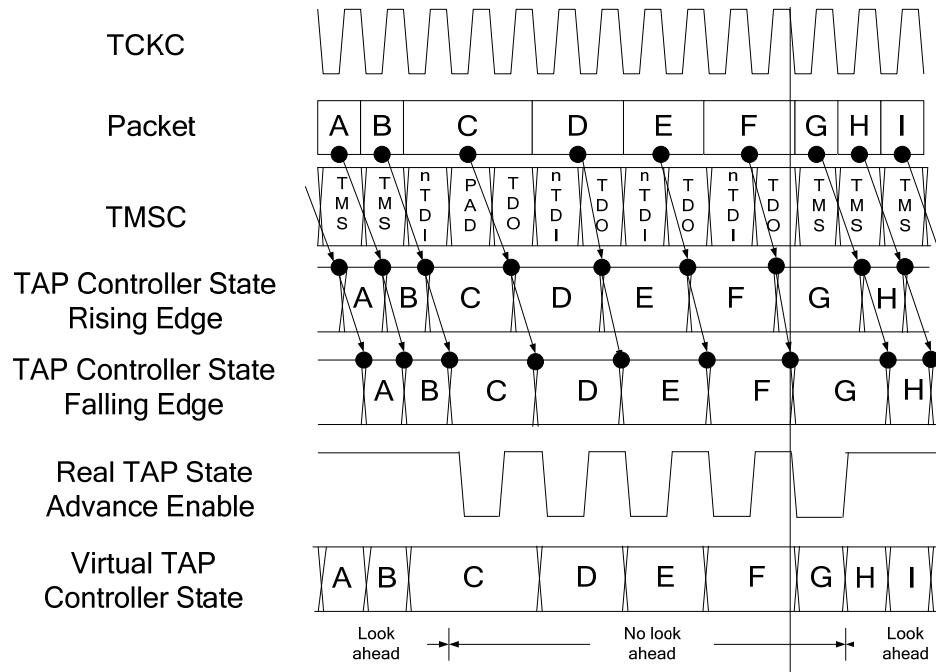
When a TDO bit is included in the SP, the relationship of the SP and TAPC state is called “nonpipelined.” The same is true when a Delay Element is included in the SP and a TDO bit is not included in the SP. This means the TAPC state is advanced before a new SP begins. When neither a TDO bit nor a Delay Element is included in the SP, the relationship of the SP and TAPC state is called “pipelined.” In this case, a subsequent SP, TP, or CP begins before the TAPC state advance caused by the prior SP occurs. The

transitions between nonpipelined and pipelined operations are shown at circle one and circle two in Figure 25-16. The effects are most noticeable when a transition from nonpipelined-to-pipelined operation occurs (shown at circle two and surrounding region). The state labels in this figure are described in Table 25-3.



**Figure 25-16 — OScan6 SP payload/TAPC state advance relationship**

The TAP.7 Controller deals with the pipelining shown above by calculating the virtual state as described in 22.6.



**Figure 25-17 — Virtual TAPC state with OScan6 Scan Format**

**Table 25-3 — Cross reference of labels in Figure 25-16 and Figure 25-17/TAPC state**

Label	Represents TAPC state
A	Select-xR-Scan
B	Capture-xR
C–F	Shift-xR
G	Exit1-xR
H–I	Pause-xR

## 25.6.2 Specifications

### Rules

- a) Each subsequent specification in 25.6.2 shall only apply to a T4 and above TAP.7 when using the OScan Scan Formats.
- b) When the TAP.7 Controller is Online, the TAPC state shall be advanced during a bit period when all of the following are true:
  - 1) The SP initiating the TAPC advance is not followed by a CP.
  - 2) The SP is not canceled by a Deselection or Selection Escape.
  - 3) Any of the following are true:
    - i) The bit period is associated with the TDO bit within the payload of an SP.
    - ii) The bit period follows the last bit period of an SP payload that does not contain a TDO bit.

## 25.7 CID allocation

### 25.7.1 Description

The CID may be allocated using the OScan Scan Formats as follows. The TAP.7 Controller inhibits Undirected CID Allocation with the OScan Scan Formats as these scan formats do not provide a means to perform the arbitration that is needed for this type of CID allocation (voting on the TDO value is not supported). Directed CID Allocation is, however, supported with the OScan Scan Formats. The nTDI bit within the input frame of the SP payload conveys the AT value.

Attempting CID allocation that does not meet the above restrictions is considered a programming error and will merely result in no CID allocation. In this case, the CIDA Command is treated as an NOP and the CID and CIDI Register values are unchanged.

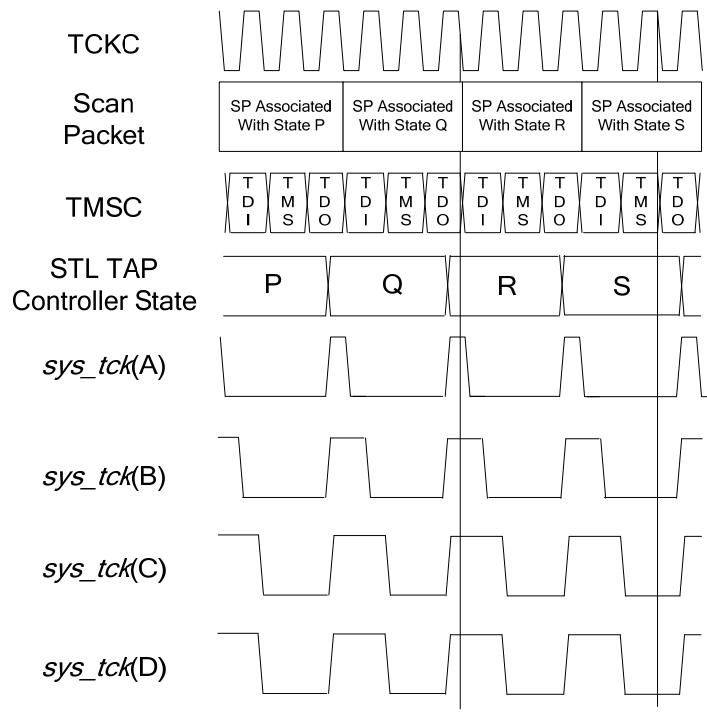
### 25.7.2 Specifications

#### Rules

- a) Each subsequent specification in 25.7.2 shall only apply to a T4 and above TAP.7 when the OScan Scan Format is used to allocate CIDs.
- b) Directed CID Allocation shall be supported.
- c) The nTDI bits within SP payloads associated with *Shift-DR* TAPC states of the CR Scan of a CIDA Command shall be used as the AT value for Directed CID Allocation.
- d) When Undirected CID Allocation is attempted:
  - 1) The CIDI and CID Register values shall not be changed.
  - 2) The CIDA Command shall be treated as a no-operation.

## 25.8 Increasing STL performance with OScan Scan Formats

The use of the *sys\_tck* signal duty-cycle adjustment to increase STL performance is only available for use with the OScan0, OScan1, OScan2, and OScan4 Scan Formats as shown in Table 23-4. An example of its use with the OScan1 Scan Format is shown in Figure 25-18.



*sys\_tck(x)* where x = A, B, C, or D shown in Figure 23-6

**Figure 25-18 — Example with the *sys\_tck* signal duty-cycle options with OScan1**

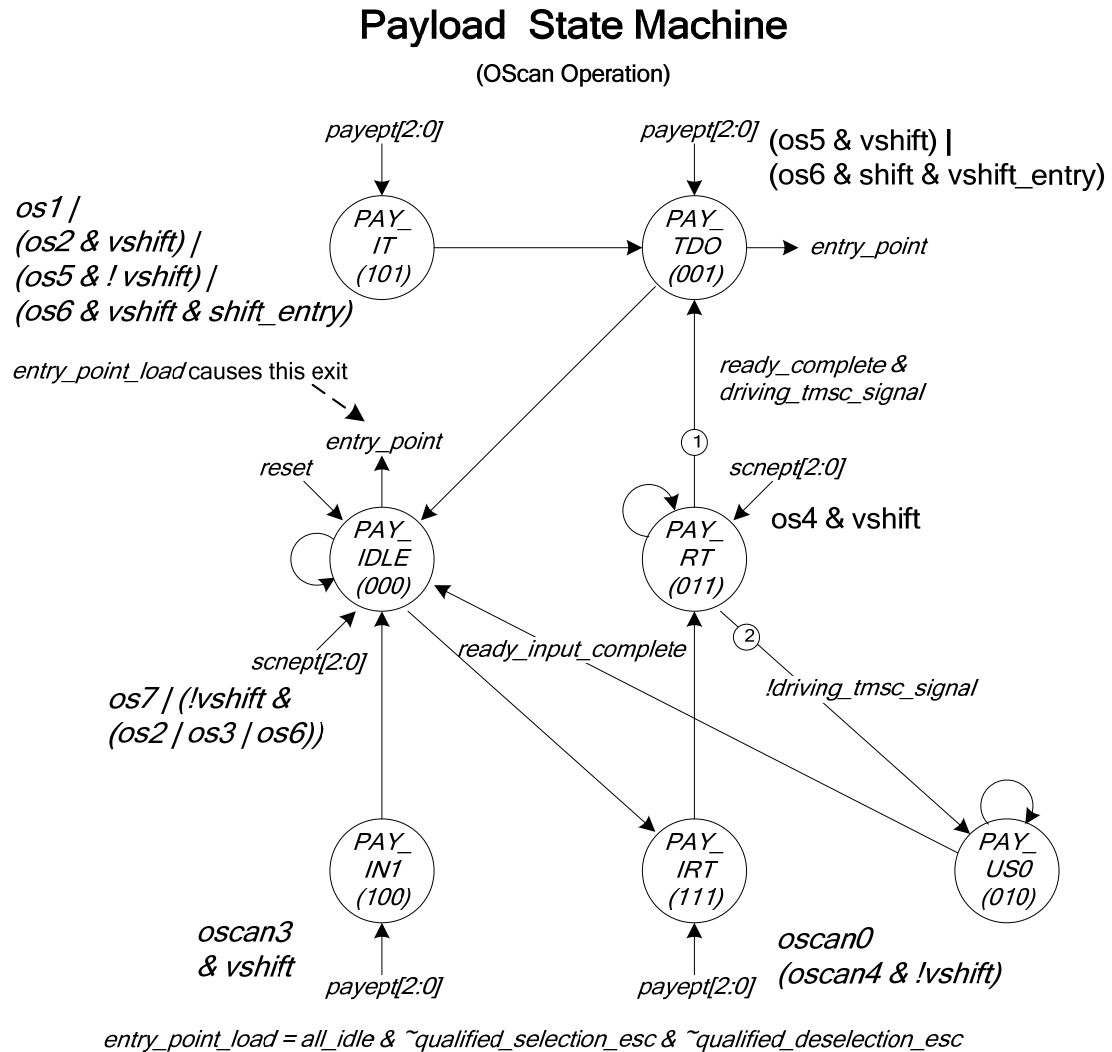
## 25.9 An approach to implementing OScan Scan Formats

### 25.9.1 Payload State Machine

The processing of OScan SP payloads can easily be handled with the same Payload State Machine that handles the processing of MScan SP payloads. The Ready State Machine is enhanced to accommodate the OScan ready characteristics and a delayed version of the *Shift-xR* state or a similar signal called *shift\_prog\_r*. The OScan Scan Formats utilize seven of the eight states of the Payload State Machine shown in Figure 25-19. The state names and their function are shown in Table 25-4.

**Table 25-4 — Payload State Machine states for the OScan Scan Format**

Mnemonic	Name	Description
PAY_IDLE	Scan Idle	When all states are idle, load <i>tdi_adv_r</i> and <i>tms_adv_r</i> .
PAY_IN1	Scan Input One	Load <i>tms_adv_r</i> , End of Payload Element.
PAY_IT	Scan Input and TDO	Load <i>tms_adv_r</i> , and PAY_TDO is the next state.
PAY_IRT	Scan IN1, RDY, TDO	Load <i>tms_adv_r</i> , PAY_RDY is the next state followed by PAY_TDO.
PAY_RDY	Scan Ready	If selected PAY_TDO when ready completes, else move to PAY_US0 for ready completion.
PAY_US0	Scan Utility State Zero	Wait for RDY completion by other TAP.7 Controllers, PAY_IDLE is the next state.
PAY_TDO	Scan TDO	TDO bit on the TMSC signal, end of Payload Element, and PAY_IDLE is the next state.



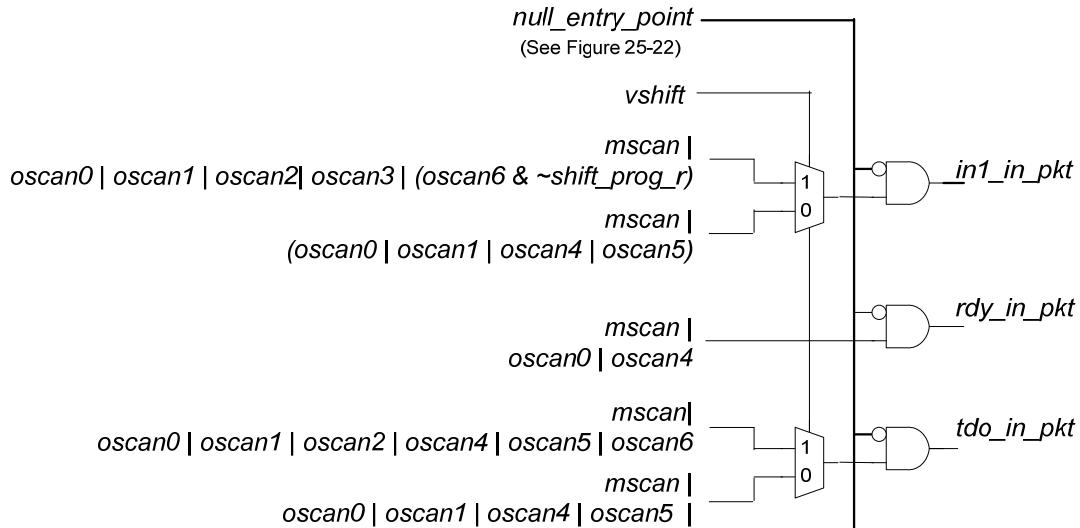
**Figure 25-19 — Conceptual view of OScan SP Payload Element behavior**

A conceptual view of the Scan State Machine and its entry point generation is shown in Figure 25-20. This machine provides the function of the Payload State Machine shown in Figure 22-10. The entry point identifies the remaining SP content as follows:

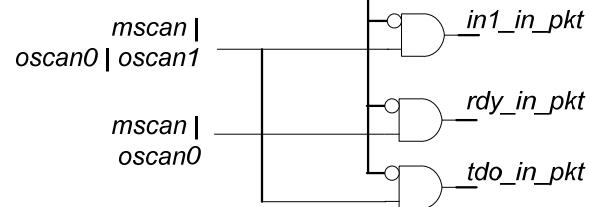
- in1\_in\_pkt Both nTDI and TMS signal values are included in the SP payload
- rdy\_in\_pkt An RDY bit is included in the SP payload
- tdo\_in\_pkt A TDO bit is included in the SP payload

The additions with SScan features will be covered with a Scan State Machine state diagram and the introduction of the Header State Machine in Clause 26.

## With All MScan and All OScan Scan Formats



## With Only Mandatory Scan Formats



**NOTE – Terms for unsupported scan formats may be deleted**

Figure 25-20 — Conceptual view of Payload State Machine entry point generation

### 25.9.2 Shift Progress flag

The Shift Progress (*shift\_prog\_r*) flag shown in Figure 25-21 indicates the payload of at least one SP associated with the *Shift-xR* state has been generated following an SP associated with the *Capture-xR* or *Exit2-xR* state.

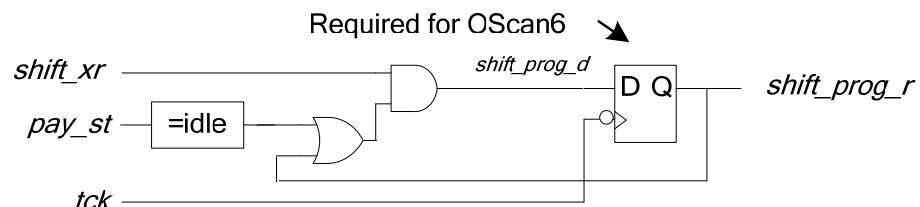
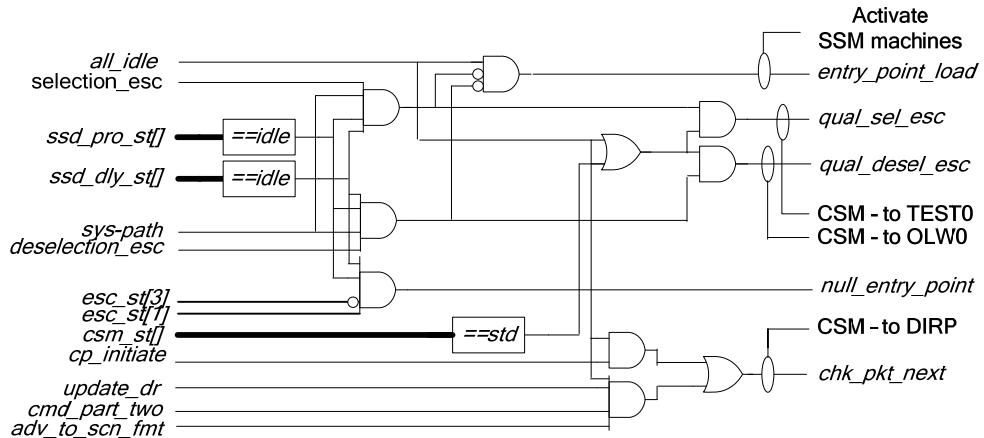


Figure 25-21 — Shift Progress flag for OScan6 support

### **25.9.3 CSM and SSM activation**

The signals activating the CSM and SSM are shown in Figure 25-22.

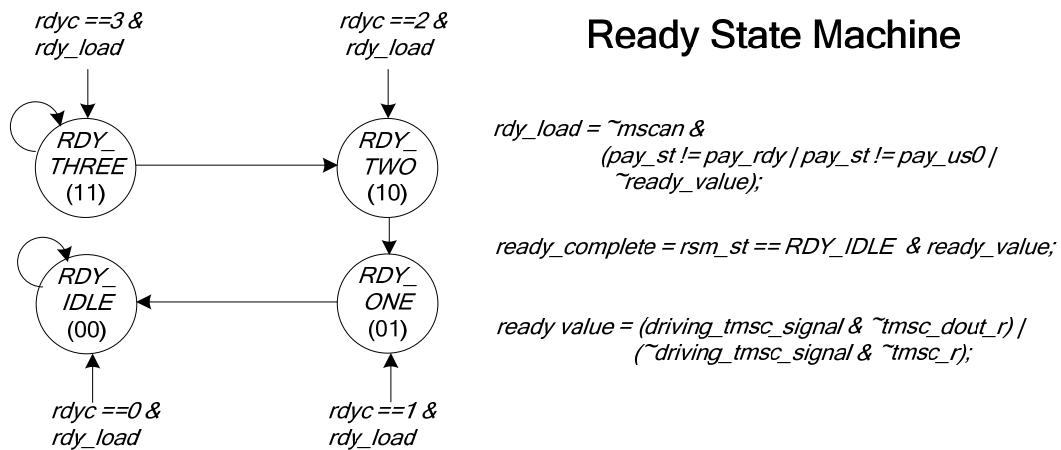


**Figure 25-22 — CSM and SSM activation**

The PAY\_US0 state is used to wait for the RDY to be generated by the TAP.7 Controller(s) with selected CLTAPCs when this TAP.7 Controller's CLTAPC is deselected.

## 25.9.4 RDY State Machine

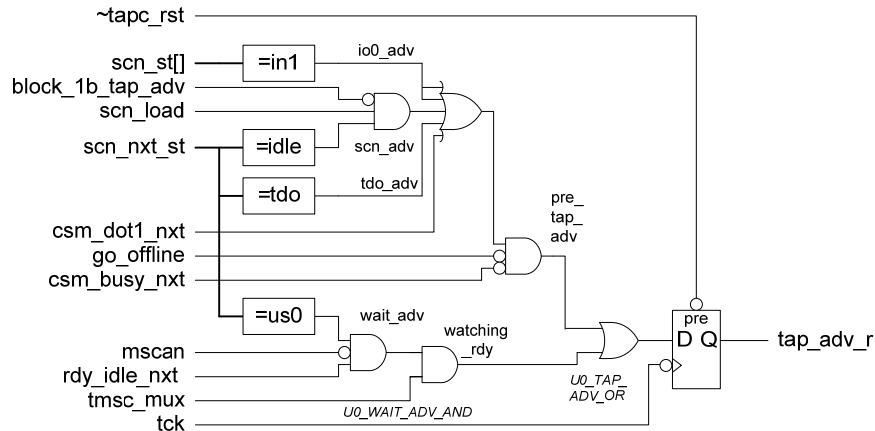
The operation of the Ready State Machine with the OScan and SScan Scan Formats is shown in Figure 25-23. This machine operates in two ways, the first is when the TAP is sourcing the RDY bit value, and the second is when it is not sourcing the RDY bit value. The RDY count is set to the RDYC Register value when the Payload State Machine State is a state other than *PAY\_RDY* or *PAY\_US0*. When the TAP sources the RDY bit value, it continues to load until the RDY bit value it sources is a logic 1. When this occurs, it counts down. When the TAP does not source the RDY bit value, it continues to load the RDY bit value until the Ready State Machine state reaches *PAY\_US0* and the sampled TMSC signal value is a logic 1. When this occurs, it counts down. It declares ready the transaction complete when the Ready State Machine state reaches *RDY\_IDLE* and the *ready\_value* signal is a logic 1.



**Figure 25-23 — Conceptual view of Ready State Machine OScan/SScan operation**

### 25.9.5 TAP advance

A conceptual view of the TAP TAPC advance enable (the *tap\_adv\_r* signal) is shown in Figure 25-24. The *sscan\_inh\_adv* and *sscan\_hdr\_adv* signals are SScan related and are tied off as a logic 0 when the SScan Scan Formats are not supported. Since the *csm\_adv\_nxt* signal comprehends a CP following an SP, the TAP advance is automatically inhibited for this case.



**Figure 25-24 — Conceptual view of TAP advance with MScan/OScan Scan Formats**

### 25.10 Where to find examples

Examples of OScan scan transfers are shown in Annex B and Annex C. Annex B contains OScan timing diagrams while Annex C contains examples of a three-bit DR Scan followed by a three-bit IR Scan. This transaction is the basis for all examples. The CP used following an SP generated for a specific scan format is also shown. The use of delays is also detailed. These examples may be used to compare the Advanced Protocol's behavior using each of these scan formats with and without the use of a delay. The point at which the TAP.7 Controller state is advanced is also shown. The pipelined and non-pipelined operation of the Advanced Protocol is also highlighted in these examples.

## 26. SScan Scan Formats

This clause is applicable to T4 and above TAP.7s. It provides the rules, permissions, and recommendations for the implementation of the SScan Scan Formats. It describes the SP header, payload, and Delay Elements created by the *SPA* state while using the SScan Scan Formats. It also describes the special TAPC advance considerations for these scan formats.

The subject matter within this clause is described in the following order:

- 26.1 Capabilities
- 26.2 High-level operation
- 26.3 Scan Packet content
- 26.4 Header Element
- 26.5 Payload Element
- 26.6 Delay Element
- 26.7 Packet sequences and factors influencing them
- 26.8 Advancing the TAPC state
- 26.9 CID allocation
- 26.10 Increasing STL performance with SScan Scan Formats
- 26.11 An approach to implementing SScan Scan Formats
- 26.12 Where to find examples

### 26.1 Capabilities

#### 26.1.1 Primary purpose

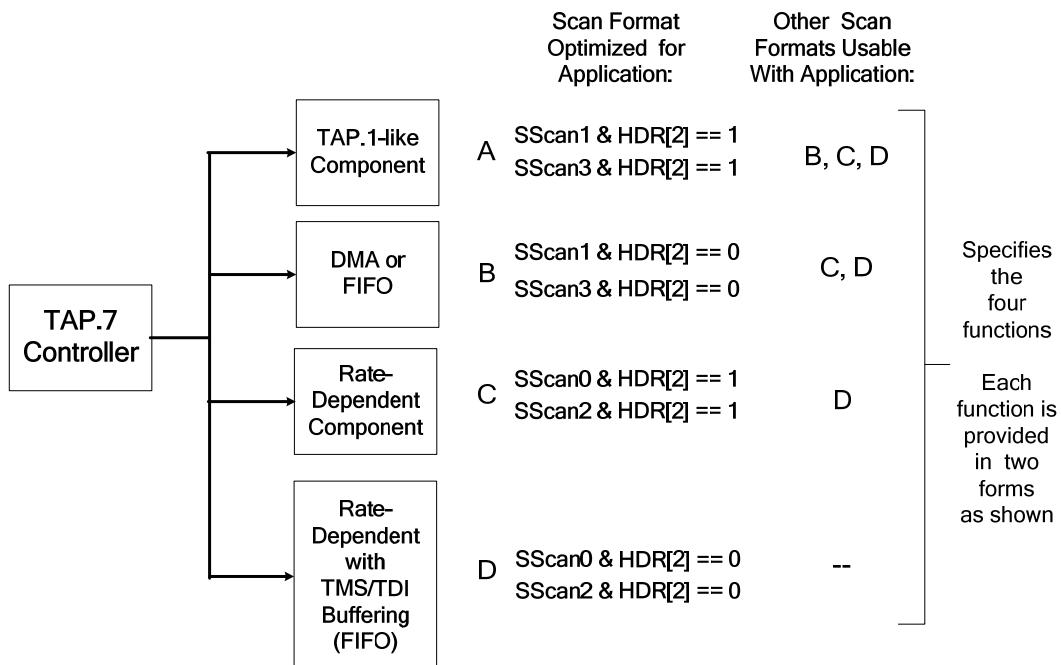
The SScan Scan Formats provide a means for the DTS to use its knowledge of the application and the data that needs to be exchanged to increase scan performance. The DTS can use this knowledge to group the SPs associated with non-*Shift-xR* states into one or more “Control Segments” and group the SPs associated with *Shift-xR* states into one or more “Data Segments.” The DTS can use its knowledge of the relevant data of a scan operation to create Data Segments that produce better scan efficiency when a data transfer may be partitioned into a number of segments whose content may be: only input, only output, and both input and output between the *Capture-xR* and *Update-xR* states. The control and Data Segments can also be optimized based on a component’s handshake characteristics (DMA, etc.), transferring only the data that is needed:

- Only TDI            An input-only segment
- TDI and TDO      An input/output segment
- Only TDO            An output-only segment

The optimization choices consider having a data-rate-dependent component involved in the scan transaction. The applications supported by the SScan Scan Formats are described first followed by a description of the attributes of control and Data Segments that support these applications.

### 26.1.2 Application types supported

The SScan Scan Formats provide a means to create high-performance debug capability. They cannot be used for boundary-scan purposes if the use of SSDs is required. These scan formats do not contain TDI information in SPs associated with non-*Shift-xR* TAPC states. These scan formats support the four scan application types shown in Figure 26-1.



NOTE-The SScan2 and SScan3 Scan Formats provide more efficient scan transfers but may only be used with a DTS-sourced TCKC as they utilize EOT escape sequences.

**Figure 26-1 — Application types supported by SScan Scan Formats**

The SScan Scan Formats are used to:

- Debug with CPU cores having TAP.1 debug interfaces (e.g., no stalls and minimize the number of bits transferred for the scan).
- Provide optimized scan access to DMAs used to access memory, registers, etc. (e.g., stalls are placed at data-segment boundaries). The DTS may create Data Segments that coincide with word boundaries or multiples of word boundaries. It can also define the content of the SP payloads within each segment using the header that precedes Data Segments. It may minimize the number of bits transferred for both DR and IR Scans in this manner.
- Optimize transfers to scan rate-dependent components where the maximum TCKC signal frequency is affected by a data-rate-dependent component with the STL (e.g., a TAPC whose operation is affected by an unorthodox use of the Test Clock).
- Buffer bursts of TDI or TMS signal information contained in a single segment, consuming this information at a different rate than the TCKC signal rate, and stalling in the first SP of the next segment until it is ready for the DTS to complete another burst of TDI or TMS signal information. In this case, the DTS controls the burst size, using a burst size that is compatible with the component being accessed within the STL. The burst size (one to n bits) may be varied dynamically on a burst-by-burst basis, if desired.

### 26.1.3 Control Segments

A Control Segment is:

- Used with both IR and DR Scans
- Associated with consecutive TAPC states other than the *Shift-IR* and *Shift-DR* states
- Composed of one or more SPs (and TPs when they follow an SP)
- Prefixed by a three-bit header (HDR[2:0]) that:
  - Identifies a segment as a Control Segment when any of following are true:
    - The TAPC state is a state other than *Shift-IR* or *Shift-DR*
    - All of the following are true:
      - The two LSBs (HDR[1:0]) are 11b
      - The SP preceding the header was associated with either the *Shift-IR* or *Shift-DR* state
    - Specifies the use of one of the two stall profiles supported by the scan format with the value of HDR[2]
  - Terminated automatically following an SP that either:
    - Causes entry into either the *Shift-DR* or the *Shift-IR* state, or
    - Is associated with the *Update-DR* state of Command Part Two of both two- and three-part commands with:
      - The SP/CP combination following this SP not considered part of a Control Segment
      - The SP in this SP/CP combination having the same Payload Element bit content as the first SP of the Control Segment that precedes it, no Header Element, and no Delay Element
      - The SP in this SP/CP combination not advancing the TAPC state
      - Any TMS, END, and TDO information within this SP of this SP/CP combination discarded
    - Terminated following an SP with an END bit in its payload where the value of the END bit is a logic 1
    - Terminated, when the scan format is SScan2 or SScan3, following an SP where an EOT Escape occurs coincidently with the TMS bit in its payload
    - Extended by an additional SP when it is not terminated following the current SP

With a Control Segment, each SP within the segment:

- Contains an input frame
- Advances the TAPC state
- The TDI value presented to the ADTAPC and CLTAPC is a logic 1
- The HDR[1:0] value:
  - Will be 11b for a Control Segment following a Data Segment
  - May be any value for a Control Segment following a Control Segment (11b is recommended)

### 26.1.4 Data Segments

A Data Segment is:

- Used with both IR and DR Scans
- Associated with consecutive *Shift-IR* or consecutive *Shift-DR* states
- A minimum of one SP in duration without any maximum number of SPs in a segment
- Prefixed by a three-bit header (HDR[2:0] that:
  - Identifies a segment as a Data Segment when all of following are true:
    - The TAPC state is either the *Shift-IR* or the *Shift-DR* state
    - Any of the following are true:
      - The SP preceding the header was associated with a TAPC state other than *Shift-IR* or *Shift-DR*
      - HDR[1:0] is a value other than 11b
    - Specifies the use of one of the two stall profiles supported by the scan format with the value of HDR[2]
  - Terminated following SP with an END bit in its payload where the value of this bit is logic 1
  - Terminated following SP[k] when an EOT Escape occurs coincidently with an nTDI bit in its payload of SP[k] when the Scan Format is SScan2 or SScan3
  - Terminated following SP[k] when an EOT Escape occurs coincidently with an RDY or TDO bit in the payload of SP[k-2] when the Scan Format is SScan2 or SScan3, the Data Segment is an output-only segment, and an EOT Escape has not already initiated termination of the Data Segment
  - Extended by an additional SP when it is not terminated following the current SP

When each SP within the Data Segment contains an input bit-frame:

- The number of SPs in a Data Segment is equal to the number of TAPC states associated with the segment.
- The TAPC state is advanced:
  - By all SPs with the segment except the last.
  - Coincident with the HDR[2] bit of the next segment.

When at least one SP within the Data Segment contains only an output bit-frame (no input bit-frame):

- The number of SPs in a Data Segment is equal to the number of TAPC states associated with the segment plus one
- The TAPC is advanced:
  - By all but the last SPs within the segment
  - Coincidently with the HDR[2] bit of the next segment

### 26.1.5 Stall profiles

There are three stall profiles, two of which may be used with each SScan Scan Format as shown in Figure 26-2.

SScan1 or SScan3	Components Supported	Stall Profile	Components Supported	SScan0 or SScan2
HDR[2] == 1	TAP.1s	No Stall - a stall free segment	N/A	N/A
HDR[2] == 0	FIFOs/DMA	First SP stall - A stall in the 1 <sup>st</sup> SP of the segment	FIFOs/DMA	HDR[2] == 0
N/A	N/A	All stall - A stall in each SP of a segment	Rate-dependent	HDR[2] == 1

**Figure 26-2 — Stall profile/application relationships**

The combination of the SScan Scan Format and header MSB (HDR[2]) defines the stall profile (one of two supported with the scan format) used with a segment. These stall profiles support the different application types described in 26.1.2.

### 26.1.6 Important characteristics

The SScan SPs do not support voting on output values as there are no precharge bits in the SScan SPs. This provides SPs whose minimum size ranges from one to four bits. Since voting is not supported, the TAP.7 Controller is placed Offline when any SScan Scan Format is used and there is more than one or there is the potential for more than one TAP.7 Controller with a selected CLTAPC (a Configuration Fault is detected). OScan and SScan SPs associated with non-*Shift-xR* TAPC states have roughly the same efficiency depending on the use case.

## 26.2 High-level operation

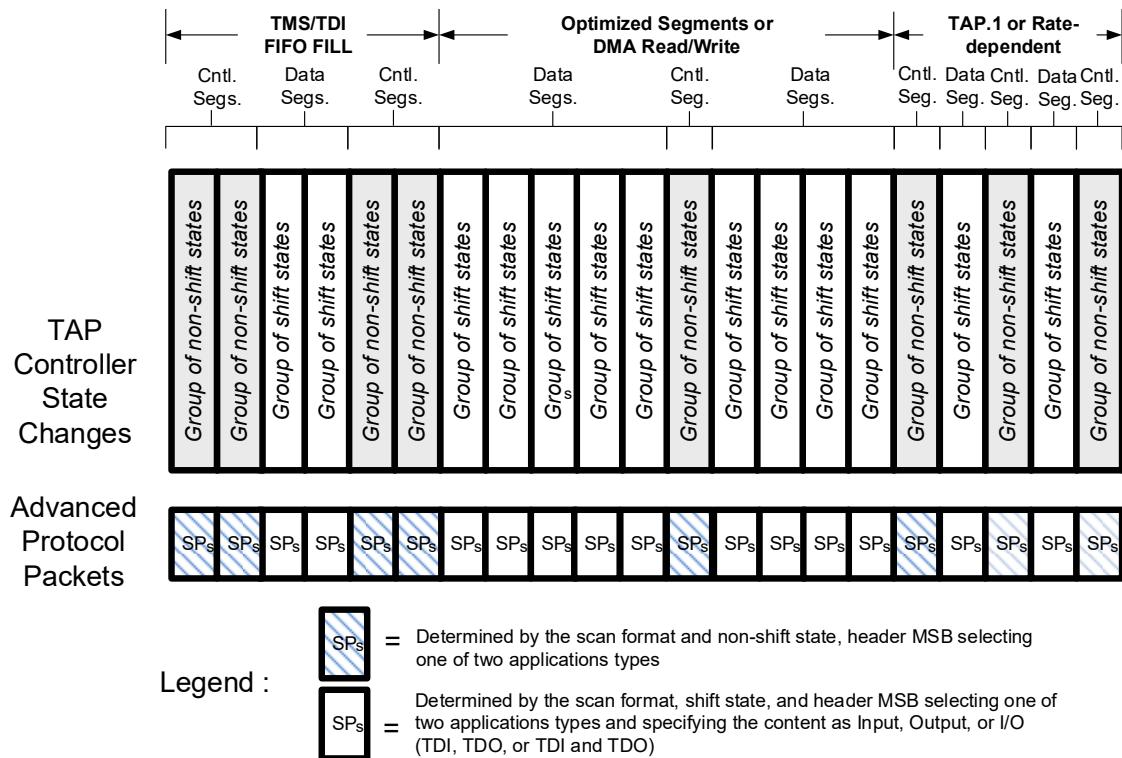
### 26.2.1 Overview

A combination of the scan format, TAPC state, and Header Element in the first SP of a segment specifies the segment attributes. Since a header precedes each segment, the segment characteristics may be changed on a segment-by-segment basis.

Segmented scan capabilities are provided in two forms, the first optimized for a TS-sourced TCKC signal and the second optimized for a DTS-sourced TCKC signal. The form that is optimized for the DTS clock source is more efficient as the SP payloads used with this form contain fewer bits. The constraints placed on the use of SScan Scan Formats with a DTS- and TS-sourced TCKC signal are the same as those for the OScan Scan Formats.

SScan Scan Formats generally produce a more efficient scan transfer than OScan Scan Formats. The sum of the number of bits in SP payloads is likely to be less with SScan Formats than that produced with the most efficient OScan Scan Formats (those providing bidirectional data transfers—OScan6 for a DTS-sourced TCK signal and with OScan2 for a TS-sourced TCKC signal). This may not be true in cases where the relevant information rapidly toggles between input and output between the *Capture-xR* and *Update-xR* TAPC states or all scan information exchanged in the *Shift-DR* and *Shift-IR* states is a fixed mix of both of input and output.

An overview of the SScan SP content with different stall profiles is shown in Figure 26-3.



**Figure 26-3 — SScan segment use cases**

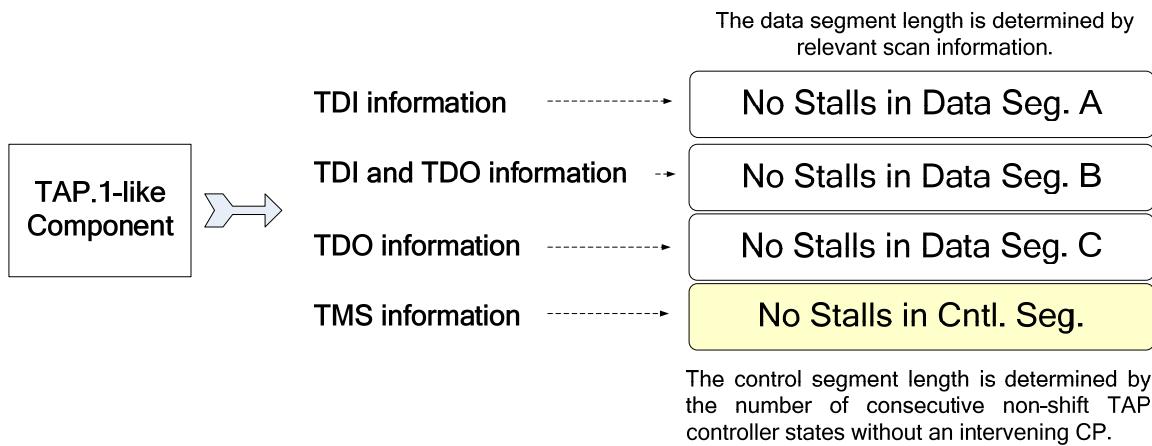
The first series of segments loads TDI and TMS signal information into a first in, first out (FIFO) interfacing to a device that can consume these values as a function of the functional-clock frequency. The second series of segments show these segments used for two purposes. With the first use case, Data Segments exchange information with a DMA or FIFO component with a fixed-word width where a check for ready is performed with each word exchanged. In the second use case, a series of Data Segment breaks a DR Scan into groups of SPs with the information of interest being the same within a group of states but different for adjacent groups of states. In the third use case, conventional scans are performed with either a stall opportunity in each SP or with stall-free segments.

## 26.2.2 Segments and their use

### 26.2.2.1 Description

#### 26.2.2.1.1 Use with a TAP.1-like component

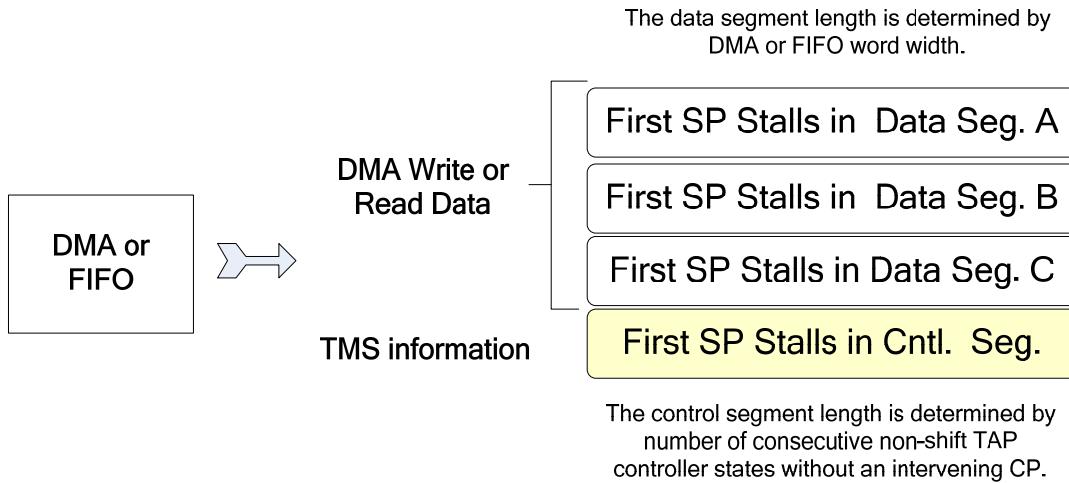
With a TAP.1-like component, as shown in Figure 26-4, a data transfer is broken into a number of Data Segments that correlate to the data of interest. There is a single Control Segment. The SScan1 and SScan3 Scan Formats provide this capability with no stalls when used with a Header Element having HDR[2] set to a logic 1.



**Figure 26-4 — SScan transactions with a TAP.1-like component**

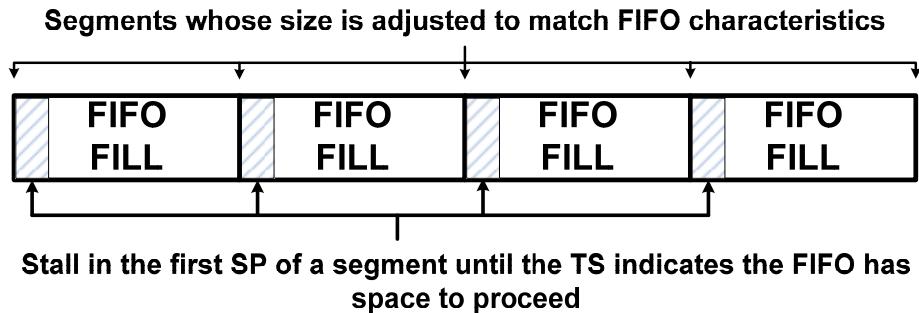
#### 26.2.2.1.2 Use with a DMA or FIFO component

With a DMA or FIFO component, as shown in Figure 26-5, a data transfer of a number of words of data is broken into Data Segments whose length is equal to the embedded component's word width. The SScan1 and SScan3 Scan Formats provide this capability with a stall at the beginning and end of each Data Segment when used with a Header Element with HDR[2] set to a logic 0. The number of SPs within the data-segment is defined by the DTS based on its understanding of the component's characteristics. The component indicates when it is ready to proceed at the beginning of each Data Segment and at the beginning of a Control Segment following a Data Segment. This eliminates the polling of the component to ensure it is ready to perform the transfer.



**Figure 26-5 — SScan transactions with a DMA or FIFO**

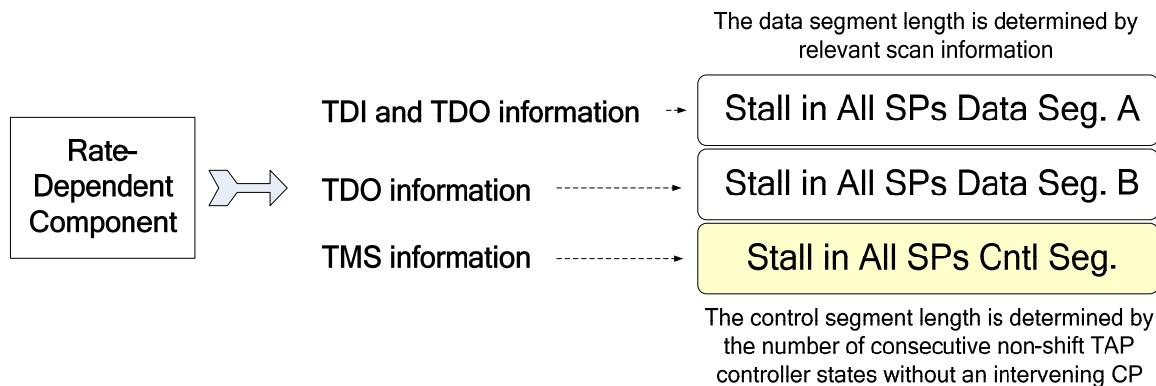
With a transfer involving an FIFO or buffer, the segment size is matched to the FIFO characteristics as shown in Figure 26-6.



**Figure 26-6 — SScan transactions with an FIFO**

#### 26.2.2.1.3 Use with a direct-access data-rate-dependent component

The SScan0 or SScan2 Scan Format is used with a Header Element with HDR[2] set to a logic 1 during these segments to provide the DTS a means to directly access a data-rate-dependent component and exchange unbuffered data. A data transfer of a number of words of data is divided into Data Segments whose lengths are dependent on the relevant data. An RDY bit is included in each SP in both control and Data Segments as shown in Figure 26-7.

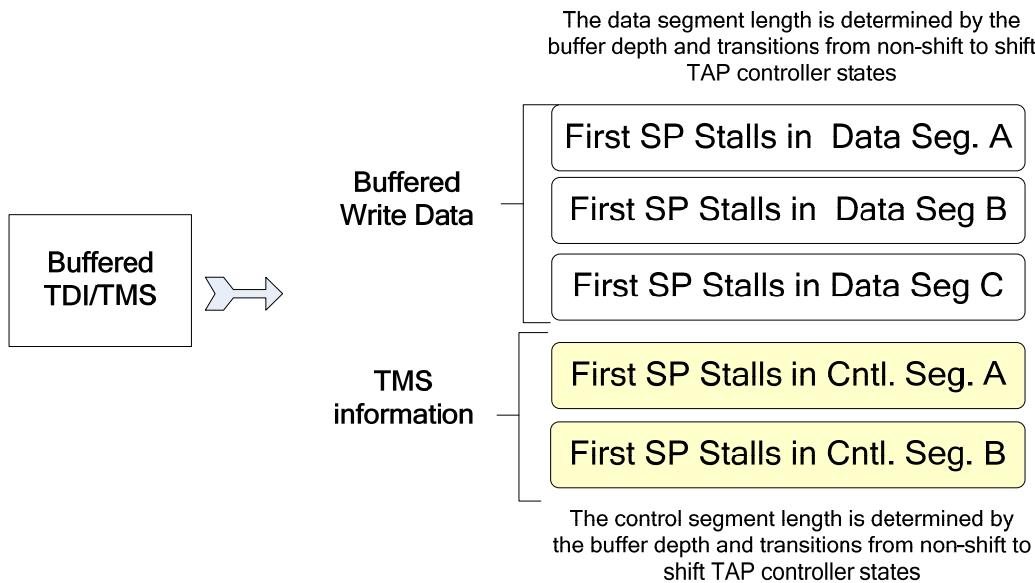


**Figure 26-7 — SScan transactions with rate-dependent components**

#### 26.2.2.1.4 Use with a buffered TDI/TMS component

The SScan0 or SScan2 Scan Format is used with a Header Element with HDR[2] set to a logic 0 during these segments to provide the DTS a means to access a data-rate-dependent component that buffers input bit-frames.

A data transfer of a number of words of data is broken into control and Data Segments whose length is some percentage of the buffer depth as shown in Figure 26-8. The DTS then ends the segment and begins a new segment. The component indicates that it has space in the buffer for a new segment of information with the RDY bit at the beginning of the new segment.

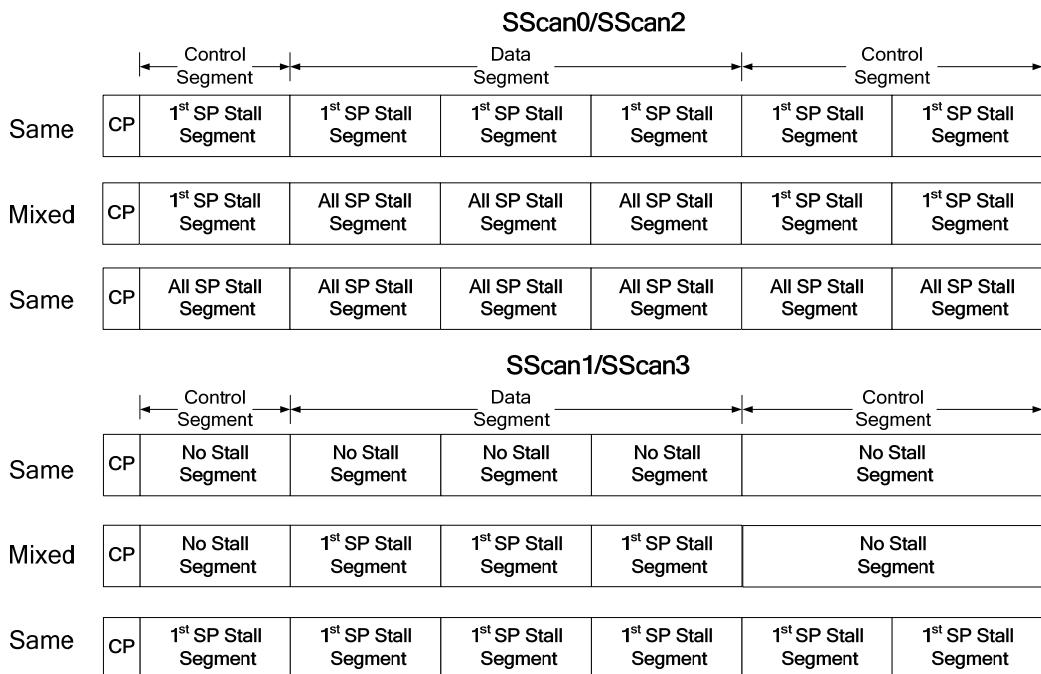


**Figure 26-8 — SScan transactions with a buffered rate-dependent component**

For applications that buffer input bit-frame bit values but provide unbuffered TDO Data Segments, either the SScan0 or SScan2 Scan Format is used with a Header Element with HDR[2] set to a logic 1 during these segments where the DTS will retrieve unbuffered TDO data. Control Segments may be operated with the buffering of input bit-frame data as described above.

#### 26.2.2.1.5 Utilizing the same scan format for two applications types

The attributes of adjacent segments may be changed subject to the segment stalls available with a scan format as shown in Figure 26-9 using the MSB of the header.



**Figure 26-9 — Stall profile combinations**

### 26.2.2.2 Specifications

#### Rules

- a) Each subsequent specification in 26.2.2.2 shall only apply to a T4 and above TAP.7 when using the SScan Scan Formats.
- b) A Control Segment shall include only SPs associated with consecutive TAPC states other than *Shift-DR* and *Shift-IR*.
- c) A Data Segment shall include SPs associated with either consecutive *Shift-DR* or consecutive *Shift-IR* TAPC states.
- d) The following packet and packet combinations shall not be considered within a segment:
  - 1) An SP/CP combination.
  - 2) A CP following the use of the Standard Protocol.
- e) A segment shall be a sequence of one or more SPs that begins as specified by Rule 26.2.2.2 f) and continues until terminated by conditions identified by any of the following:
  - 1) Rule 26.5.2 m).
  - 2) Rule 26.5.2 n).
  - 3) Cancellation of an SP by any of the following:
    - i) A qualified Selection Escape.
    - ii) A qualified Deselection Escape.
- f) The processing of a segment shall begin, provided any of the following are true:
  - 1) A segment is terminated as specified by Rule 26.5.2 m).
  - 2) All of the following are true:
    - i) A Check Packet completes as a result of a CP-END Directive.
    - ii) The scan format is an SScan Scan Format.
    - iii) A Configuration Fault is not detected.

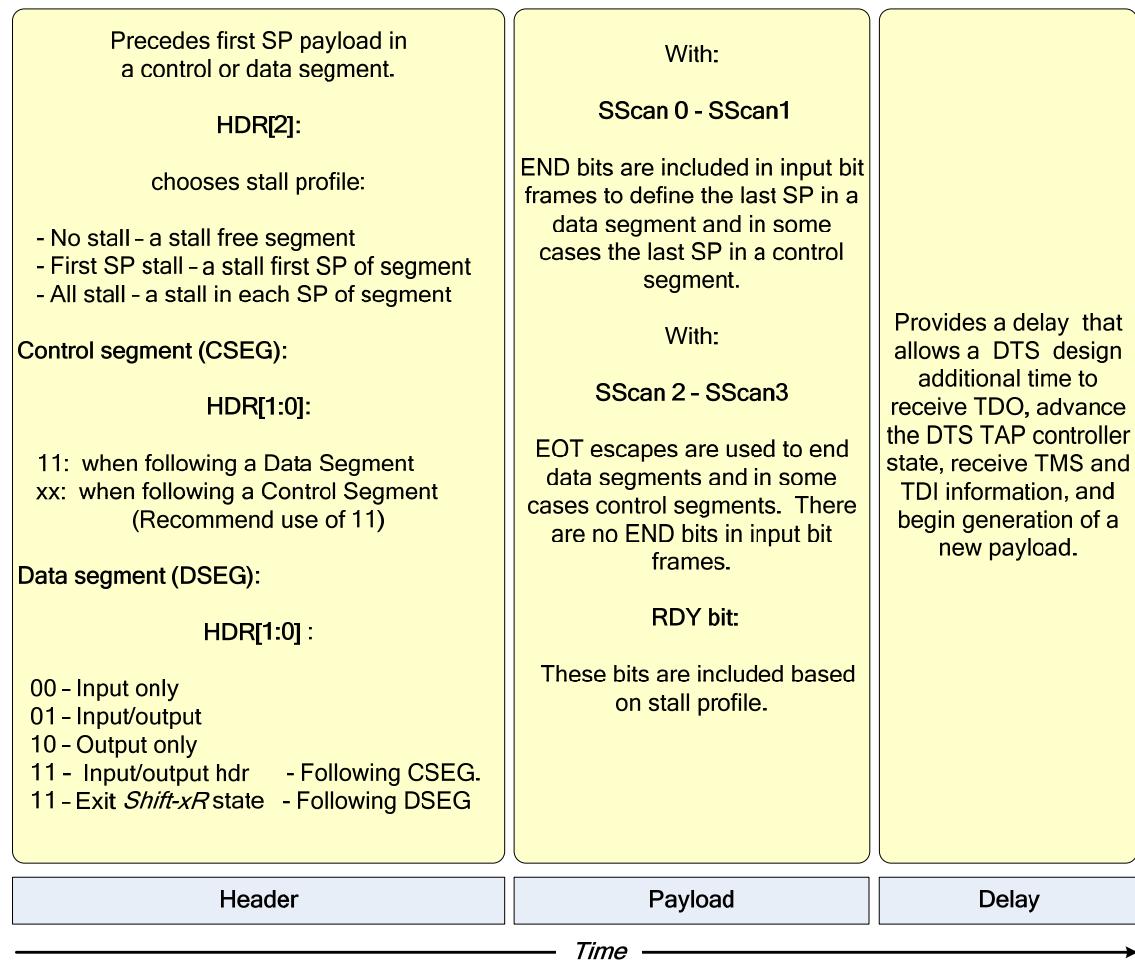
## 26.3 Scan Packet content

### 26.3.1 Description

The SPs created while using the SScan Scan Formats utilize all of the SP Elements introduced in Clause 22. A brief description of their function is shown in Figure 26-10. The Header Element is included in the first SP of a data or Control Segment. The Payload Element is present in each SP. A Delay Element is included when the DLYC Register value is greater than zero.

The SP payload for SScan Scan Formats has two additional attributes:

- Control information within input bit-frames to define the end of a segment
- Payload with only output bit-frames supporting output-only transfers



**Figure 26-10 — Scan Packet Elements with the SScan Scan Formats**

### 26.3.2 Specifications

#### Rules

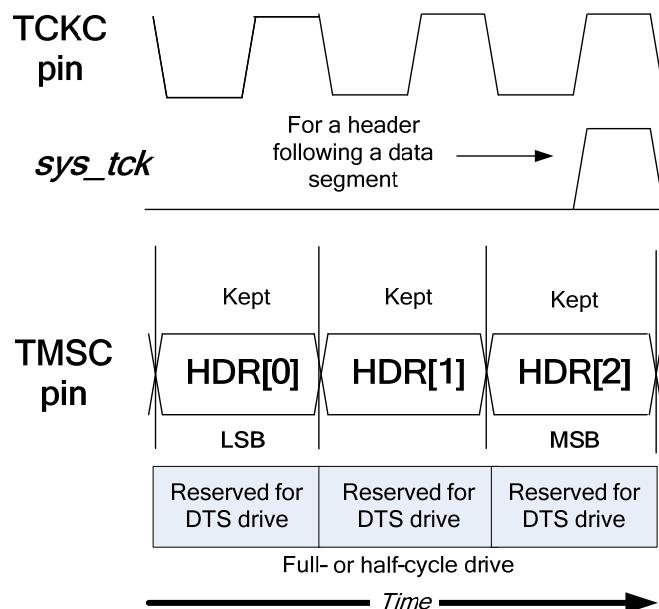
- a) Each subsequent specification in 26.3.2 shall only apply to a T4 and above TAP.7 when using the SScan Scan Formats.
- b) The first SP of a segment shall be processed as having a Header Element preceding its Payload Element.
- c) Each SP of the segment shall be processed as having a Payload Element.
- d) Each SP of the segment shall be processed as having a Delay Element following a Payload Element, provided the DLYC Register value is greater than zero.
- e) When a segment is terminated as specified by Rule 26.5.2 n), a CP/SP packet combination shall be processed following this SP or SP/TP combination.
- f) The SP preceding a CP shall have all of the following characteristics:
  - 1) No Header Element.
  - 2) No Delay Element.

- 3) A Payload Element that has content (the bits in the SP and not the bit values) of the SP preceding a CP is the same as the payload content of the first SP of the Control Segment preceding the SP.

## 26.4 Header Element

### 26.4.1 Description

The header is shown in Figure 26-11.



**Figure 26-11 — Header Element of an SP**

### 26.4.2 Specifications

#### Rules

- a) Each subsequent specification in 26.4.2 shall only apply to a T4 and above TAP.7 when using the SScan Scan Formats.
- b) A header with the format shown in Figure 26-11 shall be included in the first SP of all control and Data Segments.
- c) The interpretation of the value of HDR[2] shall be governed by Table 26-1.

**Table 26-1 — HDR[2] interpretation**

Scan format	HDR[2]	RDY bit(s) included in:
SScan0/ SScan2	0	Only the first SP in a segment
	1	All SPs in a segment
SScan1/ SScan3	0	Only the first SP in a segment
	1	No SPs in a segment

- d) The interpretation of the value of HDR[1:0] shall be governed by Table 26-2.

**Table 26-2 — HDR[1:0] interpretation**

<b>Previous VState</b>	<b>VState</b>	<b>HDR[1:0]</b>	<b>This segment is a:</b>	<b>Segment type</b>
Non- <i>Shift-xR</i>	Non- <i>Shift-xR</i>	Don't Care Recommend 11	Control Segment	N/A
Non- <i>Shift-xR</i>	<i>Shift-xR</i>	00	Data Segment	Input only
		01		Input/output
		10		Output only
		11		Input/output
<i>Shift-xR</i>	X	00	Transition to <i>Exit1-xR</i> ; the Header Element is in the first SP of the Control Segment following the Data Segment	Input only
	X	01		Input/output
	X	10		Output only
	X	11		N/A

NOTE—The VState is the TAPC state associated with the SP that includes the header. The previous VState is the virtual TAPC state that preceded the current VState.

## 26.5 Payload Element

### 26.5.1 Description

#### 26.5.1.1 Factors determining payload content

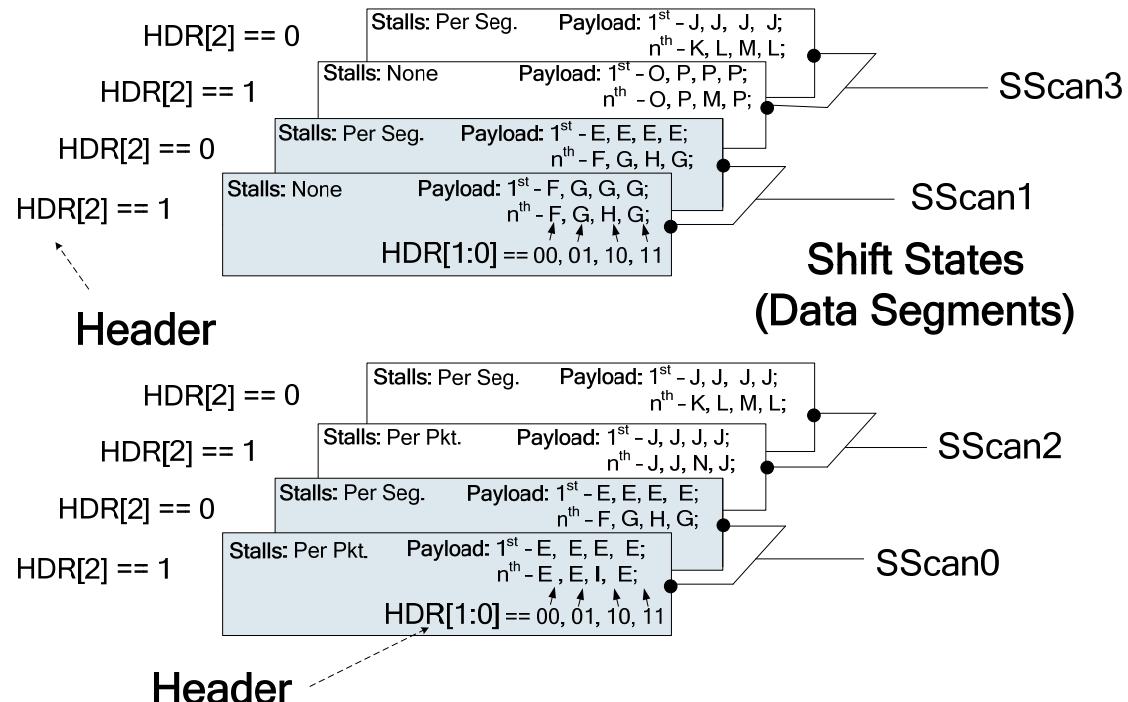
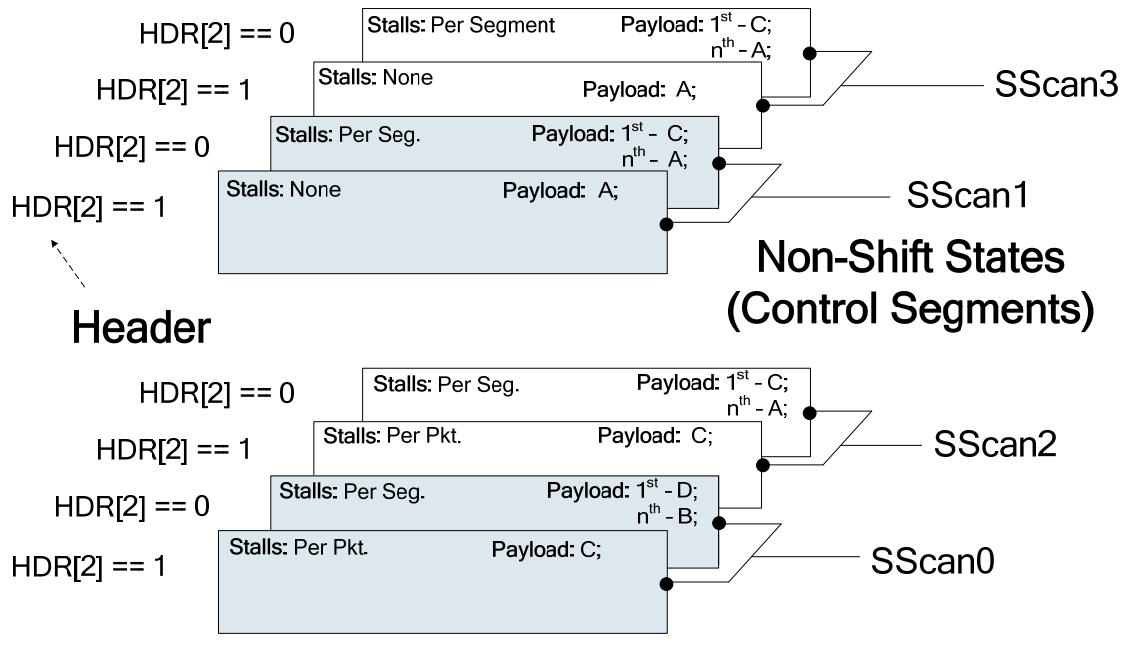
The SScan Scan Packet payload content is determined by the:

- Scan format
- TAPC state
- Header

This is illustrated in Figure 26-12. This figure highlights two key aspects of the SScan Scan Formats:

- The capability of the SScan0–SScan1 Scan Formats parallels the capability of the SScan2–SScan3 Formats with different TCKC signal sources (DTS and TS) supported
- Each SScan Scan Format supports two debug application types

## Scan Format



NOTE—The Payload type shown in Figure 26-12 is correlated with payload (PAY) designations in Table 26-4 through Table 26-7.

**Figure 26-12 — SP payloads for Shift-xR and non-Shift-xR TAPC states**

### 26.5.1.2 Optimizations

The SScan Scan Formats provide optimization choices where knowledge of the data content of the scan exchange is required. These scan formats replicate three functions (the stall profiles) in two forms (optimized for TS- and DTS-sourced TCKC) as described in 26.1.2:

- SScan0–SScan1      Will be used when the Test Clock is sourced by the TS and may be used when the DTS sources the Test Clock.
- SScan2–SScan3      May only be used when the DTS sources the Test Clock as the use of an EOT Escape is required. Provides the best performance when the DTS sources the Test Clock.

These scan formats provide three highly optimized types of scans supporting the application types described in 26.1.2:

- Using the debug facilities of CPU components embedded in the STL
- Interfacing to a DMA or FIFO memory interface
- Interfacing to a component with unorthodox use of the Test Clock where the:
  - Component's TAPC state rate is modulated by factors other than the TCKC signal frequency (an implementation approach discouraged by IEEE Std 1149.1)
  - TDI and TMS signal values are buffered to improve performance

The SScan Scan Formats optimization summary is shown in Table 26-3.

**Table 26-3 — SScan Scan Formats optimization summary**

Scan format	HDR[2]	Supporting	Performance	Flexibility	TCKC source
SScan3	0	DMA <sub>s</sub> , FIFO <sub>s</sub>	Best	Best	DTS
	1	STL IP with IEEE 1149.1-Specified Behavior			
SScan2	0	DMA <sub>s</sub> , FIFO <sub>s</sub>	Least	Good	DTS or TS
	1	STL IP with IEEE 1149.1-Non-disruptive Behavior			
SScan1	0	DMA <sub>s</sub> , FIFO <sub>s</sub>	Best	Good	DTS or TS
	1	STL IP with IEEE 1149.1-Specified Behavior			
SScan0	0	DMA <sub>s</sub> , FIFO <sub>s</sub>	Least		
	1	STL IP with IEEE 1149.1-Non-disruptive Behavior			

The Scan Packet payload contents for the SScan0, SScan1, SScan2, and SScan3 Scan Formats are shown in Table 26-4 through Table 26-7 and their associated legends. These tables identify the SP payloads generated for each of the eight header values and are shown for both Control Segments and Data Segments. Note that the TDO bits within Control Segments are forced to a logic 1 and are represented as ONE bits in these tables.

**Table 26-4 — SScan0 header information and SP payload contents**

Stall profile	Type/Seg. HDR [2:0]	SP #	TAPC information during non-Shift TAPC states					TAPC information during Shift TAPC states						
			P T	Input bit-frame		Output bit-frame		P T	Input bit-frame		Output bit-frame			
<b>1<sup>st</sup> stall</b>	IN 000	1st	D	TMS	END	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	B	TMS	END	—	—	—	F	nTDI	END	—	—	—
	I_O 001	1st	D	TMS	END	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	B	TMS	END	—	—	—	G	nTDI	END	—	—	TDO
	OUT 010	1st	D	TMS	END	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	B	TMS	END	—	—	—	H	—	END	—	—	TDO
	EXIT 011	1st	D	TMS	END	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	B	TMS	END	—	—	—	G	nTDI	END	—	—	TDO
	IN 100	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
<b>All stall</b>	I_O 101	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
	OUT 110	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	I	—	END	RDY	[RDY]	TDO
	EXIT 111	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO

**Table 26-5 — SScan2 header information and SP payload contents**

Stall profile	Type/Seg. HDR [2:0]	SP #	TAPC information during non-Shift TAPC states					TAPC information during Shift TAPC states						
			P T	Input bit-frame		Output bit-frame		P T	Input bit-frame		Output bit-frame			
<b>1<sup>st</sup> stall</b>	IN 000	1st	C	<b>TMS</b>	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
		nth	A	<b>TMS</b>	—	—	—	—	K	<b>nTDI</b>	—	—	—	—
	I_O 001	1st	C	<b>TMS</b>	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
		nth	A	<b>TMS</b>	—	—	—	—	L	<b>nTDI</b>	—	—	—	TDO
	OUT 010	1st	C	<b>TMS</b>	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	<b>RDY</b>	[RDY]	TDO
		nth	A	<b>TMS</b>	—	—	—	—	M	—	—	—	—	TDO
	EXIT 011	1st	C	<b>TMS</b>	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
		nth	A	<b>TMS</b>	—	—	—	—	L	<b>nTDI</b>	—	—	—	TDO
	IN 100	1st	C	TMS	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
<b>All stall</b>	I_O 101	1st	C	TMS	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
	OUT1 110	1st	C	TMS	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	<b>RDY</b>	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	N	—	—	<b>RDY</b>	[RDY]	TDO
	EXIT 111	1st	C	TMS	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO
		nth	C	TMS	—	RDY	[RDY]	ONE	J	<b>nTDI</b>	—	RDY	[RDY]	TDO

<b>Exit Shift</b>	Recommend for use with Control Segments	PT ==	Payload Type
END == 0	Does not end the segment	END== 1	Ends segment
<b>Italics</b>	An EOT Escape initiates the end of a Data Segment		
[RDY] == RDY bit values whose presence is conditional based on the value of the RDYC Register.			

**Table 26-6 — SScan1 header information and SP payload contents**

profile	Stall	Type/ Seg. HDR [2:0]	SP #	TAPC information during non-Shift TAPC states					TAPC information during Shift TAPC states					
				P T	Input bit-frame		Output bit-frame		P T	Input bit-frame		Output bit-frame		
<b>1<sup>st</sup> Stall</b>	IN 000	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	A	TMS	—	—	—	—	F	nTDI	END	—	—	—
	I_O 001	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	A	TMS	—	—	—	—	G	nTDI	END	—	—	TDO
	OUT 010	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	A	TMS	—	—	—	—	H	—	END	—	—	TDO
	EXIT 011	1st	C	TMS	—	RDY	[RDY]	ONE	E	nTDI	END	RDY	[RDY]	TDO
		nth	A	TMS	—	—	—	—	G	nTDI	END	—	—	TDO
	No Stall	IN 100	1st	A	TMS	—	—	—	F	nTDI	END	—	—	—
		nth	A	TMS	—	—	—	—	F	nTDI	END	—	—	—
		I_O 101	1st	A	TMS	—	—	—	G	nTDI	END	—	—	TDO
		nth	A	TMS	—	—	—	—	G	nTDI	END	—	—	TDO
		OUT 110	1st	A	TMS	—	—	—	G	nTDI	END	—	—	TDO
		nth	A	TMS	—	—	—	—	H	—	END	—	—	TDO
		EXIT 111	1st	A	TMS	—	—	—	G	nTDI	END	—	—	TDO
		nth	A	TMS	—	—	—	—	G	nTDI	END	—	—	TDO

**Table 26-7 — SScan3 header information and SP payload contents**

profile	Stall	Type/ Seg. HDR [2:0]	SP #	TAPC information during non-Shift TAPC states					TAPC information during Shift TAPC states					
				P T	Input bit-frame		Output bit-frame		P T	Input bit-frame		Output bit-frame		
<b>1<sup>st</sup> Stall</b>	IN 000	1st	C	TMS	—	RDY	[RDY]	ONE	J	<i>nTDI</i>	—	RDY	[RDY]	TDO
		nth	A	TMS	—	—	—	—	K	<i>nTDI</i>	—	—	—	—
	I_O 001	1st	C	TMS	—	RDY	[RDY]	ONE	J	<i>nTDI</i>	—	RDY	[RDY]	TDO
		nth	A	TMS	—	—	—	—	L	<i>nTDI</i>	—	—	—	TDO
	OUT 010	1st	C	TMS	—	RDY	[RDY]	ONE	J	<i>nTDI</i>	—	<i>RDY</i>	[RDY]	TDO
		nth	A	TMS	—	—	—	—	M	—	—	—	—	TDO
	EXIT 011	1st	C	TMS	—	RDY	[RDY]	ONE	J	<i>nTDI</i>	—	RDY	[RDY]	TDO
		nth	A	TMS	—	—	—	—	L	<i>nTDI</i>	—	—	—	TDO
	No Stall	IN 100	1st	A	TMS	—	—	—	O	<i>nTDI</i>	—	—	—	—
		nth	A	TMS	—	—	—	—	O	<i>nTDI</i>	—	—	—	—
		I_O 101	1st	A	TMS	—	—	—	P	<i>nTDI</i>	—	—	—	TDO
		nth	A	TMS	—	—	—	—	P	<i>nTDI</i>	—	—	—	TDO
		OUT1 110	1st	A	TMS	—	—	—	P	<i>nTDI</i>	—	—	—	TDO
		nth	A	TMS	—	—	—	—	M	—	—	—	—	TDO
		EXIT 111	1st	A	TMS	—	—	—	P	<i>nTDI</i>	—	—	—	TDO
		nth	A	TMS	—	—	—	—	P	<i>nTDI</i>	—	—	—	TDO

**Exit Shift** Recommend for use with Control Segments  
**END == 0** Does not end the segment

PT == Payload Type  
**END== 1** Ends segment

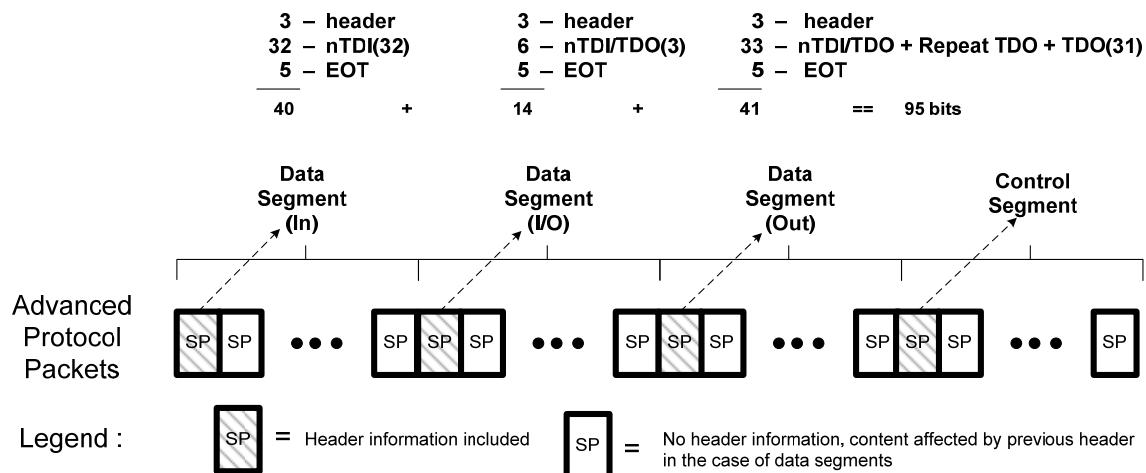
**Italics** An EOT Escape initiates the end of a Data Segment

[RDY] == RDY bit values whose presence is conditional based on the value of the RDYC Register.

With these scan formats, the DTS utilizes its knowledge of the scan characteristics to divide a series of *Shift-xR* TAPC states into Data Segment types, as this reduces the SP information volume, in most cases. For example, a series of 67 *Shift-xR* states may be broken into three Data Segments, as follows:

- Segment one An input segment only associated with 32 *Shift-xR* states
- Segment two An I/O segment associated with 3 *Shift-xR* states
- Segment three An output-only segment associated with 32 *Shift-xR* states

When this 67-bit scan is performed with the OScan1 Scan Format, there are 201 TCKC periods ( $3 \times 67$ ) associated with the *Shift-xR* state. With an SScan3 Scan Format using a no stall profile, in the SScan Scan sequence, there are only 94. This sequence provides better than a 100% improvement in performance over the OScan1 Scan Format. When this 67-bit scan is performed with the OScan6 Scan Format (the most efficient OScan Scan Format with bidirectional capability), there are 135 TCKC periods [ $(2 \times 67) + 1$ ] associated with the *Shift-xR* state. This sequence provides a 30% improvement in performance with this OScan6 comparison as shown in Figure 26-13.



**Figure 26-13 — Segment composition**

### 26.5.1.3 Input bit-frame

SScan SP input bit-frames may have content and behavioral differences when compared to an OScan input bit-frame. The characteristics of the SScan input bit-frame are described in the following paragraphs.

The input bit-frame of an SP associated with a Control Segment:

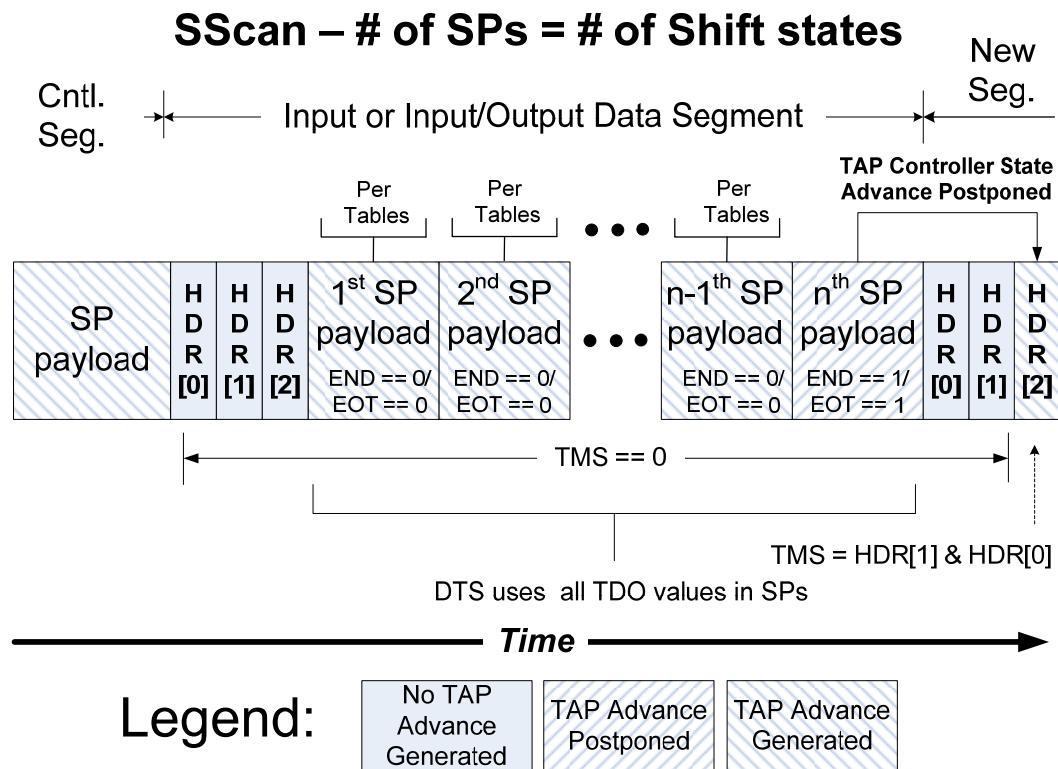
- Contains a TMS bit
- May contain an END bit

The input bit-frame of an SP associated with a Data Segment may contain all combinations of nTDI and END bits (none, only nTDI, only END, both nTDI and END).

In the case where there is an nTDI bit in the first SP of a Data Segment and there are no nTDI bits in subsequent SPs within the segment, the value of the nTDI bit in the first SP creates the TDI value presented to the EPU and STL for all TAPC states within the Data Segment.

The behavior of a Data Segment where each SP within the Data Segment has an input bit-frame is shown in Figure 26-14. The SP content in this figure is governed by Table 26-4 through Table 26-7, and their

associated legends. This figure is also used to describe the TAPC state advance for these types of Data Segments.



**Figure 26-14 — SScan input-only and input/output Data Segment processing**

The operation of an SScan1 and SScan3 input-only Data Segment using no stalls is shown in Table 26-8 and Table 26-9, respectively. These figures show sequences of TAP states for different numbers of *Shift-xR* states. Each row should be viewed separately as adjacent rows are not related. Using the first SP stall profile merely adds an output bit-frame with RDY and TDO bits to the first SP.

**Table 26-8 — SScan1 no stalls and input-only Data Segments**

# shift states	Time											
1	nTDI	<b>END</b>	HDR[0]	HDR[1]	HDR[2]							
2	nTDI	END	nTDI	<b>END</b>	HDR[0]	HDR[1]	HDR[2]					
3	nTDI	END	nTDI	END	nTDI	<b>END</b>	HDR[0]	HDR[1]	HDR[2]			
4	nTDI	END	nTDI	END	nTDI	END	nTDI	<b>END</b>	HDR[0]	HDR[1]		
n	nTDI	END	nTDI	END	nTDI	END	nTDI	END	nTDI	<b>END</b>		
SP #	0		1		2		3		4			

TAP Advance

**Bold End == 1**

End == 0

SP

**Table 26-9 — SScan3 no stalls and input-only Data Segments**

# shift states	Time →									
	nTDI	HDR[0]	HDR[1]	HDR[2]						
1	nTDI	<b>HDR[0]</b>	<b>HDR[1]</b>	<b>HDR[2]</b>						
2	nTDI	nTDI	<b>HDR[0]</b>	<b>HDR[1]</b>	<b>HDR[2]</b>					
3	nTDI	nTDI	nTDI	<b>HDR[0]</b>	<b>HDR[1]</b>	<b>HDR[2]</b>				
4	nTDI	nTDI	nTDI	nTDI	<b>HDR[0]</b>	<b>HDR[1]</b>	<b>HDR[2]</b>			
5	nTDI	nTDI	nTDI	nTDI	nTDI	<b>HDR[0]</b>	<b>HDR[1]</b>	<b>HDR[2]</b>		
6	nTDI	nTDI	nTDI	nTDI	nTDI	nTDI	<b>HDR[0]</b>	<b>HDR[1]</b>	<b>HDR[2]</b>	
SP#	0	1	2	3	4	5				

TAP Advance

**BOLD == EOT ESC**

SP

The operation of SScan1 and SScan3 input/output-only Data Segments using no stalls is shown in Table 26-10 and Table 26-11, respectively. Using the first SP stall profile merely adds an RDY bit to the output bit-frame of the first SP.

**Table 26-10 — SScan1 no stalls and input/output Data Segment**

# shift states	Time →									
	nTDI	<b>END</b>	TDO[0]	HDR[0]	HDR[1]	HDR[2]				
1	nTDI	END	TDO[0]	nTDI	<b>END</b>	TDO[1]	HDR[0]	HDR[1]	HDR[2]	
2	nTDI	END	TDO[0]	nTDI	END	TDO[1]	nTDI	<b>END</b>	TDO[2]	HDR[0]
3	nTDI	END	TDO[0]	nTDI	END	TDO[1]	nTDI	<b>END</b>	TDO[2]	nTDI
4	nTDI	END	TDO[0]	nTDI	END	TDO[1]	nTDI	END	TDO[2]	nTDI
5	nTDI	END	TDO[0]	nTDI	END	TDO[1]	nTDI	END	TDO[2]	nTDI
n	nTDI	END	TDO[0]	nTDI	END	TDO[1]	nTDI	END	TDO[2]	nTDI
SP#	0			1			2			

TAP Advance

**Bold End == 1**

End == 0

SP

**Table 26-11 — SScan3 no stalls and input/output Data Segment**

# shift states	Time →									
	nTDI	TDO[0]	HDR[0]	HDR[1]	HDR[2]					
1	<b>nTDI</b>	TDO[0]								
2	nTDI	TDO[0]	<b>nTDI</b>	TDO[1]	HDR[0]	HDR[1]	HDR[2]			
3	nTDI	TDO[0]	nTDI	TDO[1]	<b>nTDI</b>	TDO[2]	HDR[0]	HDR[1]	HDR[2]	
4	nTDI	TDO[0]	nTDI	TDO[1]	nTDI	TDO[2]	<b>nTDI</b>	TDO[3]	HDR[0]	HDR[1]
5	nTDI	TDO[0]	nTDI	TDO[1]	nTDI	TDO[2]	nTDI	TDO[3]	<b>nTDI</b>	TDO[4]
n	nTDI	TDO[0]	nTDI	TDO[1]	nTDI	TDO[2]	nTDI	TDO[3]	<b>nTDI</b>	TDO[4]
SP#	0		1		2		3		4	

TAP Advance

**BOLD == EOT ESC**

SP

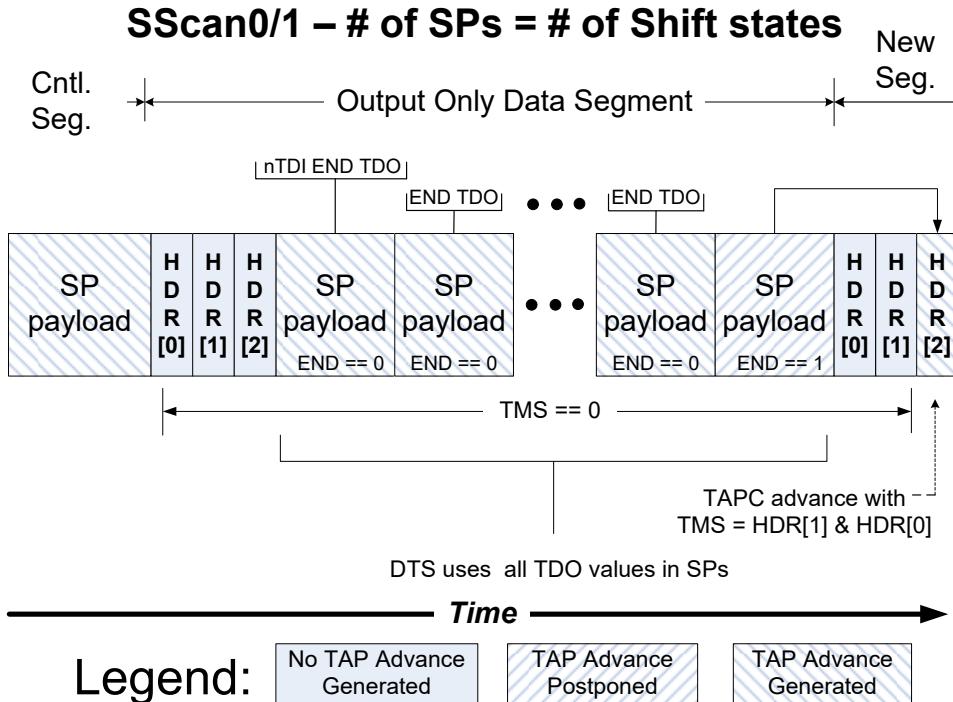
#### 26.5.1.4 Output bit-frame

##### 26.5.1.4.1 Content

The output bit-frame of an SScan SP has the same content as the OScan output bit-frame. The output bit-frame of an SP associated with a Control Segment may contain the following combinations of RDY and ONE bits (TDO bits forced to a logic 1): none, both RDY(s) and ONE. The output bit-frame of an SP associated with a Data Segment may contain the following combinations of RDY and TDO bits: none, only TDO, both RDY(s) and TDO. The RDY(s) may be a combination of RDY and [RDY] bits.

##### 26.5.1.4.2 SScan0/1 output-only segments

SScan0–SScan1 output-only Data Segments create the SP sequence shown in Figure 26-15.



**Figure 26-15 — SScan0/SScan1 output-only Data Segment processing**

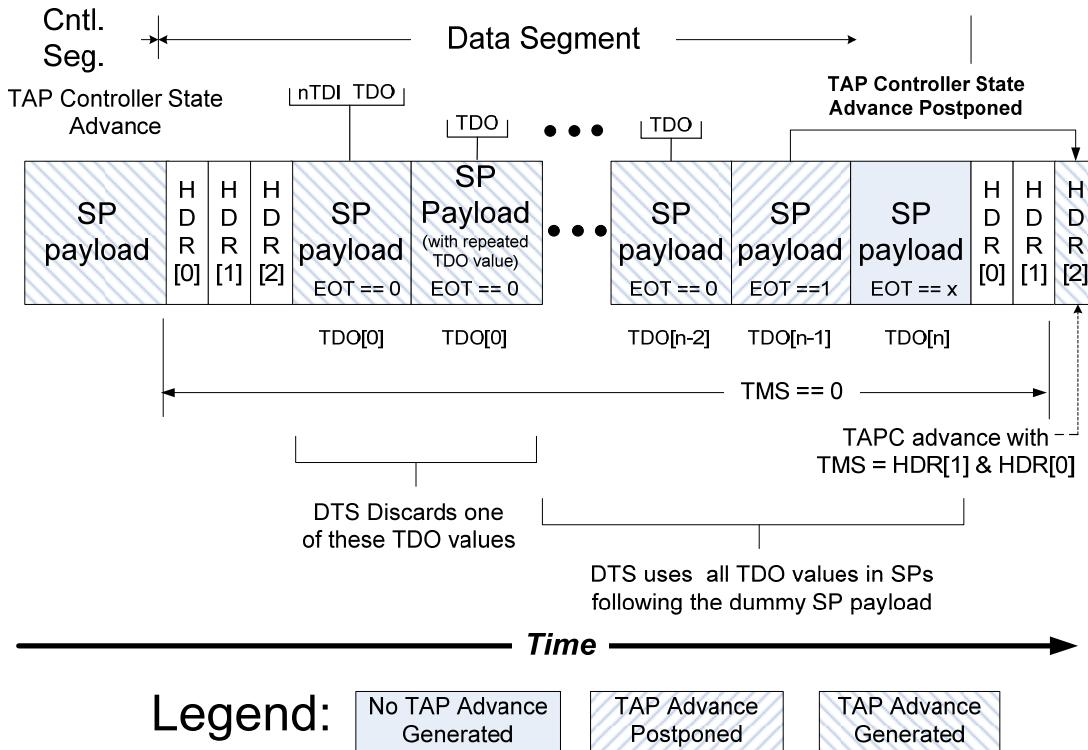
In this figure, all SPs have both an input and an output bit-frame. There is one-to-one correspondence between the number of *Shift-xR* states associated with a Data Segment and the number of SPs within the Data Segment. The input bit-frame of the first SP contains an nTDI bit and an END bit. The nTDI bit establishes the TDI value supplied to the EPU and STL for all *Shift-xR* states associated with the segment when the number of *Shift-xR* states associated with the segment is greater than one. An END bit with a logic 1 value identifies the last SP of the segment. When a segment contains more than one SP, input bit-frames subsequent to the first contain only an END bit. Although output bit-frames in this example contain only a TDO bit, they could also contain both RDY(s) bits and TDO bits. The stall profile utilized determines the output bit-frame content.

SPs other than the last one in the Data Segment advance the TAPC state coincidently with the TDO bit within the SP payload. The number of TAPC state advances coincidently with the TDO bit of an SP is  $n$  minus one where  $n$  is equal to the number of *Shift-xR* states associated with the segment. The TAPC state advance associated with the last SP of the segment is postponed to be coincidently with the HDR[2] bit of the first SP of the next segment. Note that the value of the HDR[1:0] bits determine the TMS value for the TAP controller state advance coincident with the HDR[2] bit. A value of 11b creates a logic 1 TMS value with a logic 0 TMS value created otherwise.

#### 26.5.1.4.3 SScan2/3 output-only segments

SScan2–SScan3 output-only Data Segments create the SP sequence shown in Figure 26-16.

## SScan2/3 – # of SPs = # of Shift states +1



**Figure 26-16 — SScan2/3 output-only Data Segment processing**

In this figure, the first SP of the segment has both an input and an output bit-frame. Subsequent SPs, when present, have only an output bit-frame. The input bit-frame of the first SP contains only a nTDI bit, with the value of this bit establishing the TDI value supplied to the EPU and STL for all *Shift-xR* states associated with the segment. Although output bit-frames in this example contain only a TDO bit, they could also contain both RDY bits and a TDO bit. The stall profile utilized determines the output bit-frame content when it is included in the SP.

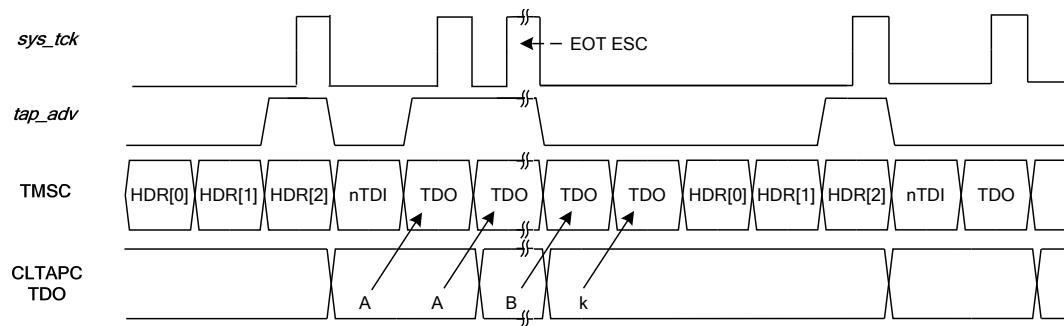
When the number ( $n$ ) of *Shift-DR* states associated with the segment is greater than one, the segment contains  $n + 1$  SPs, and only one SP otherwise. The number of TAPC state advances coincident with the TDO bit of an SP is zero when  $n$  is one and  $n - 1$  otherwise. For the case where  $n$  is one, the first SP of the segment is also the last one. In this case, the TAPC state advance associated with the last SP of the segment is postponed to be coincident with the HDR[2] bit of the first SP of the next segment. For the case where  $n$  is greater than one, both the last and the next-to-last SPs of the segment are affected since the number of SPs associated with a segment is  $n + 1$ , and the number of TAPC state advances coincident with the TDO bit in SPs is  $n - 1$ . In this case, the TAPC state advance of the next-to-last SP of the segment is postponed to be coincident with the HDR[2] bit of the first SP of the next segment, and the TAPC state advance of the last SP of the segment is inhibited. The value of the HDR[1:0] bits determines the TMS value for the TAP controller state advance coincident with the HDR[2] bit as described previously.

An EOT Escape is used to terminate a Data Segment. When it occurs coincidentally with the nTDI bit of the first SP of the segment, the segment is terminated after the first SP. Otherwise, an EOT Escape is overlaid on any of the output frame bits (RDY or TDO) in SP  $n - 2$  to cause the termination of an output-only Data Segment after SP  $n$ . The need to create the “early Escape” is described as follows.

STL TDO timing affects the construction of a segment associated with multiple *Shift-xR* states. Since the STL TDO data is registered before becoming the TMSC signal data, there is a one clock delay between the

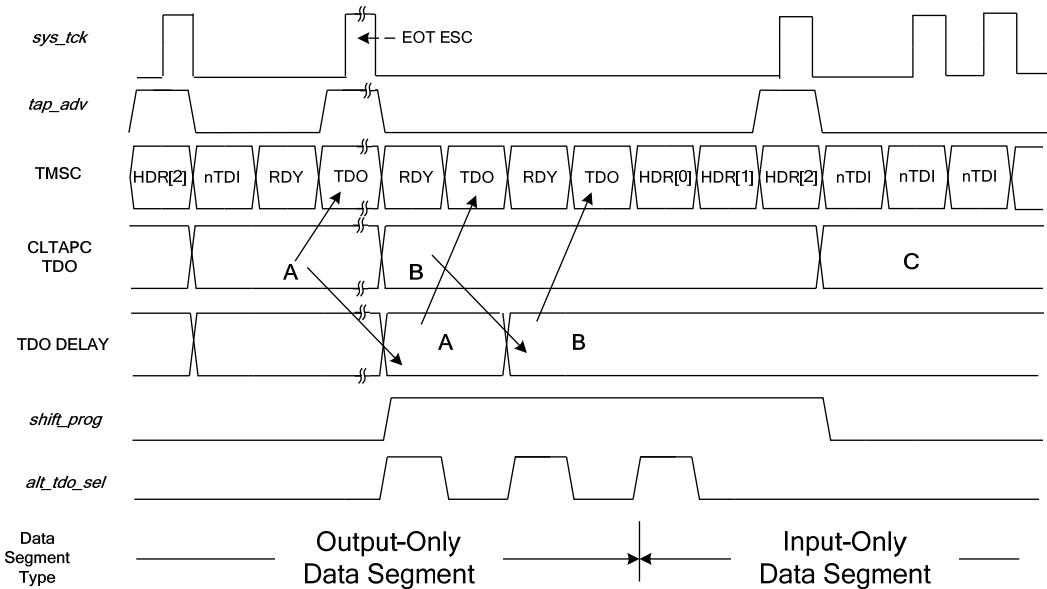
TDO data generated by the STL and its appearance at the TMSC signal as shown in Figure 26-17. With output-only segments of either SScan2 Scan Formats with the first stall profile or SScan3 Scan Formats, this causes the duplication of the first two TDO bit values associated with the *Shift-xR* state as shown in Figure 26-16. This duplication is handled by adding an extra SP following the first SP of the segment. This SP accommodates the pipelining of the TDO data used to create the TDO bit values. Hence, the number of SPs associated with a segment is one more than the number of *Shift-xR* states associated with a segment when there are two or more *Shift-xR* states associated with a segment.

A header follows the completion of the SP with TDO[n] when an EOT Escape occurs coincidently with TDO[n – 2] as shown in Figure 26-17. All SPs except the last two in the segment advance the TAPC state as though they were an OScan SP. The EOT Escape is placed in the TDO[n – 2] bit period so that it can inhibit the TAPC state advance that would normally created by TDO[n – 1]. Inhibiting this TAPC advance offsets the additional TAPC state advance that is created by the SP added after the first SP. With the addition of a TAPC state advance and the inhibiting of a TAPC state advance offsetting each other, this produces a number of TAPC advances equal to the number of *Shift-DR* states associated with the segment minus one ( $n - 1$ ) the same as with the Scan0 and Scan1 Scan Formats. This is the same number of TAPC advances produced for input-only and input/output segments with all SScan Scan Formats.



**Figure 26-17 — TDO pipelining with the SScan2/SScan3 Scan Formats**

With output-only segments used with the all-stall profile of the SScan2 Scan Format, the RDY bits preceding the TDO bits in SPs associated with the *Shift-xR* state provide sufficient time for the TDO bits to be delivered without duplication of the TDO bit values in the first two SPs of a Data Segment. With this being the only case where this opportunity exists with SScan2 and SScan3 Output-Only Data Segments, the TDO data is delayed in SPs two through  $n$  of the Data Segment to mimic the TDO behavior of the other three output-only segment types created with the SScan2 and SScan3 segments (see Figure 26-16). The pipelining needed for symmetrical operation is created artificially within the TAPC.7 Controller to provide the same relationship between TDO data associated with TAPC states and TDO data in SPs for all SScan2 and SScan3 output-only packets. With this pipelining adjustment, a delayed version of the TDO data is used for creating SP TDO bit values in SPs other than the first when RDY bits are included in all SPs. This is shown in Figure 23-14 and Figure 26-18.



**Figure 26-18 — CLTAP TDO pipelining with SScan2/SScan3 Scan Formats/with stalls**

Examples of output-only bit-frames with the SScan2 and SScan3 Scan Formats are shown in Table 26-12 and Table 26-13. Cells with italicized entries represent the TCKC signal periods where an EOT Escape terminates the Data Segment. Cells that are shaded represent TDO and HDR[2] bit periods where advance of the TAPC state occurs. See 26.8 for a description of the TAPC state advance for the SScan Scan Formats.

**Table 26-12 — SScan2/SScan3 output-only Data Segments/no RDY bit in the SP payload**

# shift states	Time →									
1	nTDI	TDO[0]	HDR[0]	HDR[1]	HDR[2]					
2	nTDI	TDO[0]	TDO[0]	TDO[1]	HDR[0]	HDR[1]	HDR[2]			
3	nTDI	TDO[0]	TDO[0]	TDO[1]	TDO[2]	HDR[0]	HDR[1]	HDR[2]		
4	nTDI	TDO[0]	TDO[0]	TDO[1]	TDO[2]	TDO[3]	HDR[0]	HDR[1]	HDR[2]	
5	nTDI	TDO[0]	TDO[0]	TDO[1]	TDO[2]	TDO[3]	TDO[4]	HDR[0]	HDR[1]	HDR[2]
<b>SP#</b>	<b>0</b>		<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>			

TAP Advance

**BOLD** == **EOT**  
**ESC**

SP

**Table 26-13 — SScan2/SScan3 output-only Data Segments/RDY bit(s) in the SP payload**

# shift states	Time →										
1	nTDI	RDY(s)	TDO[0]	HDR[0]	HDR[1]	HDR[2]					
2	nTDI	RDY(s)	TDO[0]	RDY(s)	TDO[0]	RDY(s)	TDO[1]	HDR[0]	HDR[1]	HDR[2]	
3	nTDI	RDY(s)	TDO[0]	RDY(s)	TDO[0]	RDY(s)	TDO[1]	RDY(s)	TDO[2]	HDR[0]	
4	nTDI	RDY(s)	TDO[0]	RDY(s)	TDO[0]	RDY(s)	TDO[1]	RDY(s)	TDO[2]	RDY(s)	
5	nTDI	RDY(s)	TDO[0]	RDY(s)	TDO[0]	RDY(s)	TDO[1]	RDY(s)	TDO[2]	RDY(s)	
SP#	0			1			2			3	

TAP Advance

**BOLD == EOT ESC**

SP

## 26.5.2 Specifications

### Rules

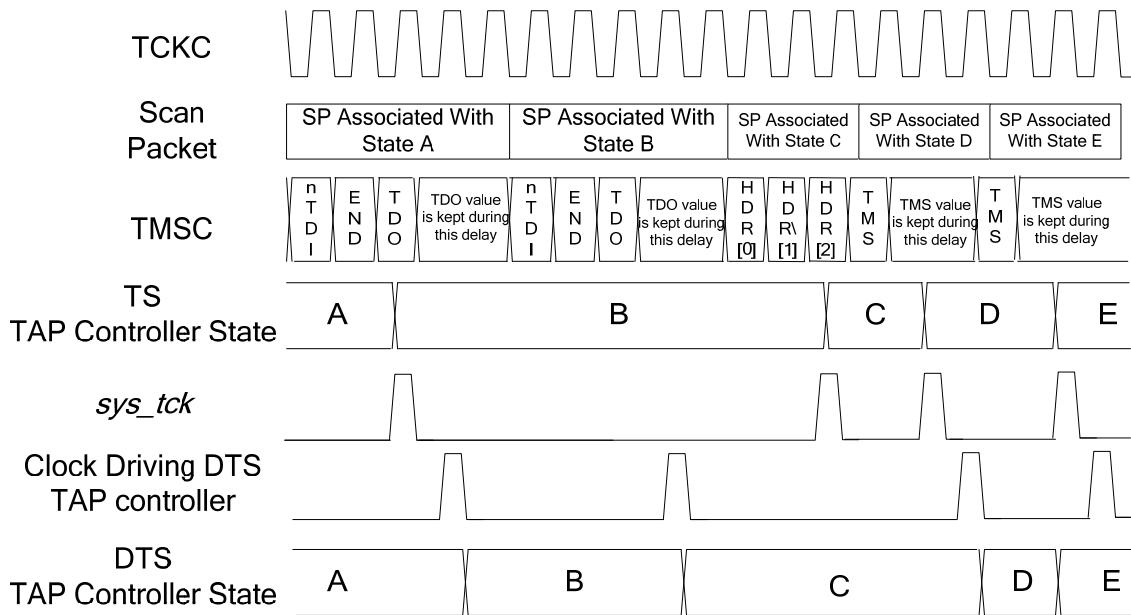
- a) Each subsequent specification in 26.5.2 shall only apply to a T4 and above TAP.7 when using SScan Scan Formats.
- b) The content of SP payloads shall be governed by Table 26-4 through Table 26-7, and the legends associated with these tables.
- c) When the TAPC state is advanced, the TDI value presented to the ADTAPC and CLTAPC shall be a logic 1 when any of the following are true:
  - 1) The TAPC state is a state other than *Shift-xR*.
  - 2) The TAPC state is *Shift-xR* and any of the following are true:
    - i) The SP payload initiating the advance contains a logic 0 nTDI bit.
    - ii) The SP payload initiating the advance does not contain an nTDI bit and the last nTDI bit was a logic 0.
- d) When the TAPC state is advanced, the TMS value presented to the ADTAPC and CLTAPC shall be a logic 1 when any of the following are true:
  - 1) The TMS bit value in the SP payload initiating the TAPC advance is a logic 1.
  - 2) The HDR[1:0] bits the first SP of a segment following a Data Segment are 11b (see Table 26-2).
 and be a logic 0 otherwise.
- e) Rule 26.5.2 f) through Rule 26.5.2 h) shall apply to TDO bit periods where the TMSC Drive Policy specifies Single Drive per Rules 14.8.2 b) and 14.8.2 c).
- f) The value of the TDO bit in the payload of SP[k] of a Data Segment shown in Table 26-4 and Table 26-6 shall be the TDO value associated with TAPC state [k] [see Rules 23.15.2 b) and c)] provided the TMSC Drive Policy dictates single drive for this bit period.
- g) The value of the TDO bit in the payload of SP[k] of a Data Segment shown in Table 26-5 and Table 26-7 shall be the TDO value associated with TAPC state [k], provided the Data Segment format is not output only (HDR[1:0] != 10b).

- h) The value of the TDO bit in the payload of SP[k] of an output-only Data Segment (HDR[1:0] ==10b) shown in Table 26-5 and Table 26-7 shall be one of the following:
  - 1) The TDO value associated with *Shift\_xR* state[k] of the Data Segment, provided the SP is the first SP of the Data Segment.
  - 2) The TDO value associated with *Shift\_xR* state[k – 1] of the Data Segment, provided the SP is not the first SP of the Data Segment.
- i) The relationship between SPs associated with a segment and the TAPC states associated with a segment shall be governed by the following:
  - 1) The number of SPs and the number of non-*Shift-xR* states associated with a Control Segment shall be equal.
  - 2) The number of SPs and the number of *Shift-xR* states associated with a Data Segment shall be equal, provided any of the following are true:
    - i) The scan format is either SScan0 or SScan1.
    - ii) The Data Segment format is not output only (HDR[1:0] != 10b).
    - iii) The Data Segment is terminated in the first SP of the segment by an EOT Escape Sequence coincident with the first bit of this SP.
  - 3) The number of SPs associated with a Data Segment shall be one more than the number of *Shift-xR* states associated with the same Data Segment, provided all of the following are true:
    - i) The scan format is either SScan2 or SScan3.
    - ii) The Data Segment format is output only (HDR[1:0] == 10b), and the Data Segment is not terminated in the first SP of the segment by an EOT Escape coincident with the nTDI bit of this SP.
- j) An EOT Escape shall be ignored, provided it occurs coincidentally with:
  - 1) Any bit within a CP.
  - 2) Any bit of an SP, provided:
    - i) The SP is followed by a CP.
    - ii) The SP is associated with the *Update-DR* state of Command Part Two.
    - iii) A bit within a Header Element.
    - iv) A bit within a Delay Element.
- k) An EOT Escape occurring coincidentally with an SP payload bit other than either an nTDI bit or TMS bit shall be ignored, provided the SP is within a:
  - 1) Control Segment.
  - 2) Input-only Data Segment (HDR[1:0] == 00b).
  - 3) Input-and output-Data Segment (HDR[1:0] == 01b or HDR[1:0] == 11b).
- l) An EOT Escape occurring coincidentally with an SP payload bit within a segment shall be ignored, provided any of the following are true:
  - 1) The termination of the segment has already been initiated by an EOT Escape.
  - 2) The scan format is SScan0 or SScan1.
  - 3) The segment is a Control Segment and either of the following are true:
    - i) The scan format is SScan3.

- ii) The value of HDR[2] is a logic 1 (the all-stall profile).
- m) A segment shall be terminated following SP[k], provided any of the following are true:
  - 1) All of the following are true:
    - i) The scan format is SScan2 and the HDR[2] is a logic 0 (first stall profile).
    - ii) The EOT Escape is coincident with the TMS bit within the Payload Element of the SP.
  - 2) All of the following are true:
    - i) The scan format is either SScan2 or SScan3.
    - ii) The EOT Escape is coincident with the nTDI bit within the Payload Element of SP[k].
  - 3) All of the following are true:
    - i) The scan format is either SScan2 or SScan3.
    - ii) The TAPC state is *Shift-xR*.
    - iii) The segment's header has designated the Data Segment as output only (HDR[1:0] == 10b).
    - iv) An EOT Escape occurs coincidentally with a bit in the output bit-frame of the payload of an SP[k-2] of the segment.
    - v) Rule 26.5.2 l) has not been satisfied.
  - 4) All of the following are true.
    - i) SP[k] is associated with a *Capture-IR*, *Capture-DR*, *Exit2-IR*, or *Exit2-DR* TAPC state.
    - ii) SP[k] contains a logic 0 TMS bit within its payload.
  - 5) The value of the END bit in the payload of the SP is a logic 1.
- n) A segment shall be terminated following SP[k], provided SP[k] is associated with the *Update-DR* state of Command Part Two of any command.
- o) When an SP contains RDY bit(s) within its payload, all of the following shall apply:
  - 1) The number of logic 0 RDY bits that precede the first logic 1 RDY bit shall be greater than or equal to zero (see Figure 25-13).
  - 2) The number of logic 1 [RDY] bits shall be equal to the RDYC Register value plus one (see Figure 25-13).

## 26.6 Delay Element

Delay Elements follow the SScan SP payloads when the DLYC Register is a nonzero value. The purpose and operation of Delay Elements is the same as for the MScan and OScan Scan Formats. The operation of Delay Elements with the SScan SP payloads with and without output bit-frames is shown in Figure 26-19. TAPC state labels are cross-referenced with TAPC states in Table 26-14.



**Figure 26-19 — Using delays with the SScan Scan Formats**

**Table 26-14 — Cross-reference of the TAPC state labels in Figure 26-19**

Label	Represents TAPC state
A-B	<i>Shift-xR</i>
C	<i>Exit1-xR</i>
D-E	<i>Pause-xR</i>

## 26.7 Packet sequences and factors influencing them

### 26.7.1 Description

The packet progression following an SP Packet associated with the *Update-DR* state is shown in Figure 26-20 for SScan0 and SScan2 Scan Formats with the 1<sup>st</sup> stall profile. Case A is with no Control Segment termination. Case B is Control Segment termination with the END bit or EOT Escape. Case C is segment termination with Command Part Two. Note that this figure also covers other non-*Shift-xR* states as described in the figure.

**SScan0 and HDR[2] == 0**

Case	SP	$n^{\text{th}}$ SP of Control Segment	
A	<i>Update-DR</i>	<i>Run-Test/Idle or Select-DR</i>	
	END == 0	Content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)	
	Not Cmd. Part Two <i>Update-DR</i>		
B	SP	$1^{\text{st}}$ SP of Control Segment	
	<i>Update-DR</i>	<i>Run-Test/Idle or Select-DR</i>	
	END == 1	Content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)	
C	SP	SP/CP Combination	SP
	<i>Update-DR</i>	<i>Run-Test-Idle or Select-DR-Scan</i>	<i>Run-Test-Idle or Select-DR-Scan</i>
	END == x	Content per Rule 26.3.2 f)	Check Packet
	Cmd. Part Two <i>Update-DR</i>	No TAPC Advance	Case 0,1, and 2

**SScan2 and HDR[2] == 0**

Case	SP	$n^{\text{th}}$ SP of Control Segment	
A	<i>Update-DR</i>	<i>Run-Test-Idle or Select-DR-Scan</i>	
	EOT Escape == 0 TMS bit	Content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)	
	Not. Cmd. Part Two		
B	SP	$1^{\text{st}}$ SP of Control Segment	
	<i>Update-DR</i>	<i>Run-Test-Idle or Select-DR-Scan</i>	
	EOT Escape == 1 TMS bit	Content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)	
C	SP	SP/CP Combination	SP
	<i>Update-DR</i>	<i>Run-Test-Idle or Select-DR-Scan</i>	<i>Run-Test-Idle or Select-DR-Scan</i>
	EOT ESC == x TMS bit	Content per Rule 26.3.2 f)	Check Packet
	Cmd. Part Two	No TAPC Advance	Case 0, 1, and 2

**Note:** Cases A and B are applicable to all states other than  
*Capture-xR*, *Shift-xR*, and *Exit2-xR*.  
(substitute states other than these in the place of *Update-DR* along with  
a corresponding next TAPC state)

**Figure 26-20 — Packets following an SP associated with the *Update-DR* and other states**

The packet progression following an SP Packet associated with the *Capture-DR* or *Exit2-DR* state is shown in Figure 26-21. Case A is with no Control Segment termination. Case B is Control Segment termination

with the END bit. Case C is Control Segment termination following an SP that causes entry into the *Shift-xR* state.

### **SScan0 and HDR[2] == 0**

Case	SP		$n^{\text{th}}$ SP of Control Segment
	<i>Capt.-xR and Exit2-xR</i>		
A	TMS==1	END == 0	SP content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)
B	SP		1 <sup>st</sup> SP of Control Segment
	<i>Capt.-xR and Exit2-xR</i>		
B	TMS==1	END == 1	SP content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)
C	SP		1 <sup>st</sup> SP of Data Segment
	<i>Capt.-xR and Exit2-xR</i>		
C	TMS==0	END == x	SP content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)

### **SScan2 and HDR[2] == 0**

Case	SP		$n^{\text{th}}$ SP of Control Segment
	<i>Capt.-xR and Exit2-xR</i>		
A	TMS==1	EOT == 0	SP content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)
B	SP		1 <sup>st</sup> SP of Control Segment
	<i>Capt.-xR and Exit2-xR</i>		
B	TMS==1	EOT == 1	SP content per Rule 24.3.2 b), Rule 24.3.2 c), and Rule 24.3.2 d)
C	SP		1 <sup>st</sup> SP of Data Segment
	<i>Capt.-xR and Exit2-xR</i>		
C	TMS==0	EOT == x	SP content per Rule 26.3.2 b), Rule 26.3.2 c), and Rule 26.3.2 d)

**Note:** The EOT escape must occur coincidently with the TMS bit in the SP. It is ignored otherwise. The notation EOT== x indicates an EOT ESC has no effect

**Figure 26-21 — Packets following an SP associated with the *Capture-xR/Exit2-xR* states**

## 26.7.2 Specifications

### Rules

- a) Each subsequent specification in 26.7.2 shall only apply to a T4 and above TAP.7 when using the SScan Scan Formats.

- b) The packet progression following an SP associated with a TAPC state other than the following states shall be governed by Figure 26-20:
  - 1) *Shift-xR*.
  - 2) *Capture-xR*.
  - 3) *Exit2-xR*.
- c) The packet progression following an SP associated with *Capture-xR* and *Exit2-xR* states shall be governed by Figure 26-21.

## 26.8 Advancing the TAPC state

### 26.8.1 Description

The TAPC state is advanced differently with SPs for non-*Shift-xR* and *Shift-xR* TAPC states when using the SScan Scan Formats. The descriptions of input frames in 26.5.1.3 and output frames in 26.5.1.4 provide substantial description of advancing the TAPC state. This subclause supplements these descriptions and establishes rules for advancing the TAPC state with the SScan Scan Formats.

#### 26.8.1.1 SP followed by a CP

An SP that precedes a CP is not considered part of a segment and does not advance the TAPC state.

#### 26.8.1.2 Control Segments

SP payloads within a Control Segment cause the TAPC state to advance in the same manner as OScan SP payloads. An SScan SP payload within a Control Segment causes the advance of the TAPC state progression coincidently with the TCKC signal rising edge occurring coincidently with:

- The TDO bit of the SP payload
- The TCKC signal period following the last bit period of the SP payload when there is no output bit-frame within the SP payload

#### 26.8.1.3 Data Segments

The TAPC advances created by SPs within Data Segments other than SScan2–SScan3 output-only Data Segments are shown in Table 26-15.

**Table 26-15 — TAPC advance with Data Segments other than SScan2–SScan3 output only**

# <i>Shift_xR</i> states	# of SPs in segment:	SPs numbered:	Advance completed just as with OScan for SP:	Advance postponed for SP:
1	1	SP[0]	none	SP[0]
2	2	SP[1:0]	SP[0]	SP[1]
3	3	SP[2:0]	SP[1:0]	SP[2]
4	4	SP[3:0]	SP[2:0]	SP[3]
n	n	SP[n-1:0]	SP[n-1:0]	SP[n]

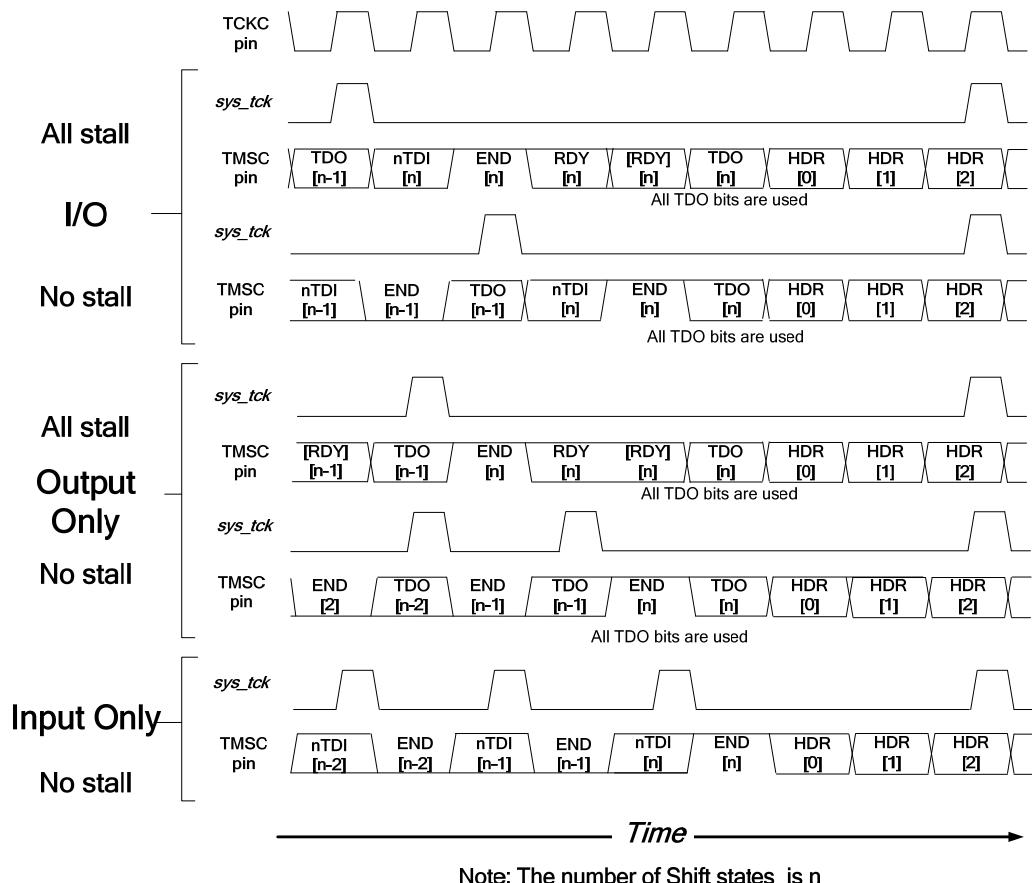
The operation of the TAPC created with a Data Segment with SScan2–SScan3 output-only Data Segments is shown in Table 26-16.

**Table 26-16 — TAPC advance with SScan2–SScan3 output-only Data Segments**

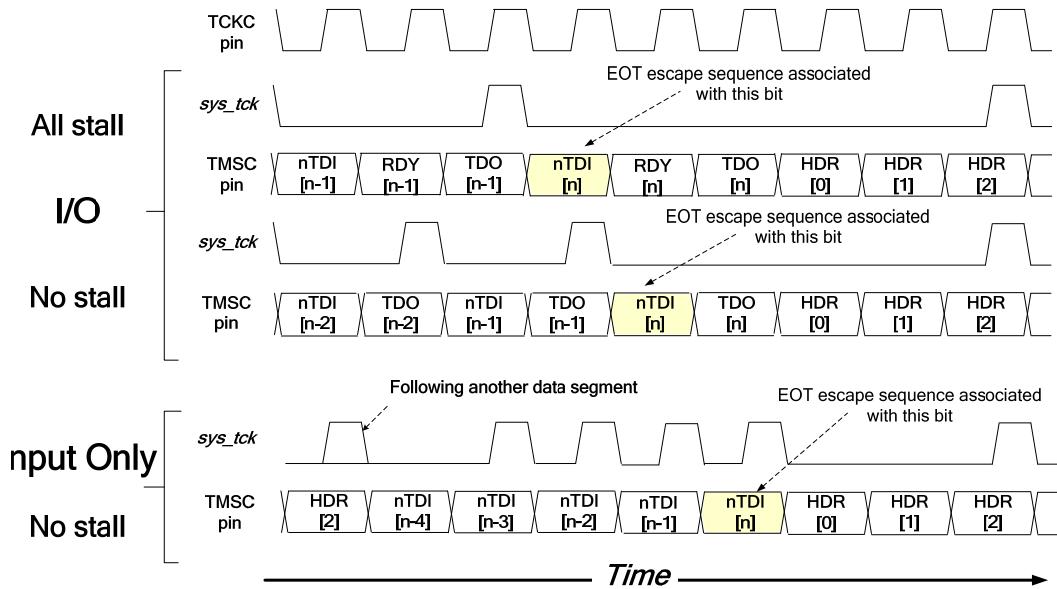
# <i>Shift_xR</i> states	# of SPs in segment	SPs numbered	Advance completed just as with OScan for SP	Advance postponed for SP	Advance discarded for SP
1	1	SP[0]	none	SP[0]	N/A
2	3	SP[2:0]	SP[0]	SP[1]	SP[2]
3	4	SP[3:0]	SP[1:0]	SP[2]	SP[3]
4	5	SP[4:0]	SP[2:0]	SP[3]	SP[4]
n	n+1	SP[n:0]	SP[n-1:0]	SP[n]	SP[n+1]

The TAP advances are discarded for the next-to-last SP when there are two or more *Shift-xR* states associated with the Data Segment. Each postponed TAPC state advance occurs when HDR[2] (the last header bit transferred) within the first SP of the subsequent segment is at the TMSC signal. The first two bits of this Header Element create the TMS signal value used with the TAPC state advance. The TAPC state advance is postponed until the HDR[1:0] bits have been input. These bits define whether an exit from the *Shift-xR* TAPC state is required. The logical AND of registered versions of these bits creates the *epu\_tms* signal value as mentioned earlier.

These behaviors are illustrated with the timing diagrams in Figure 26-22 through Figure 26-24.



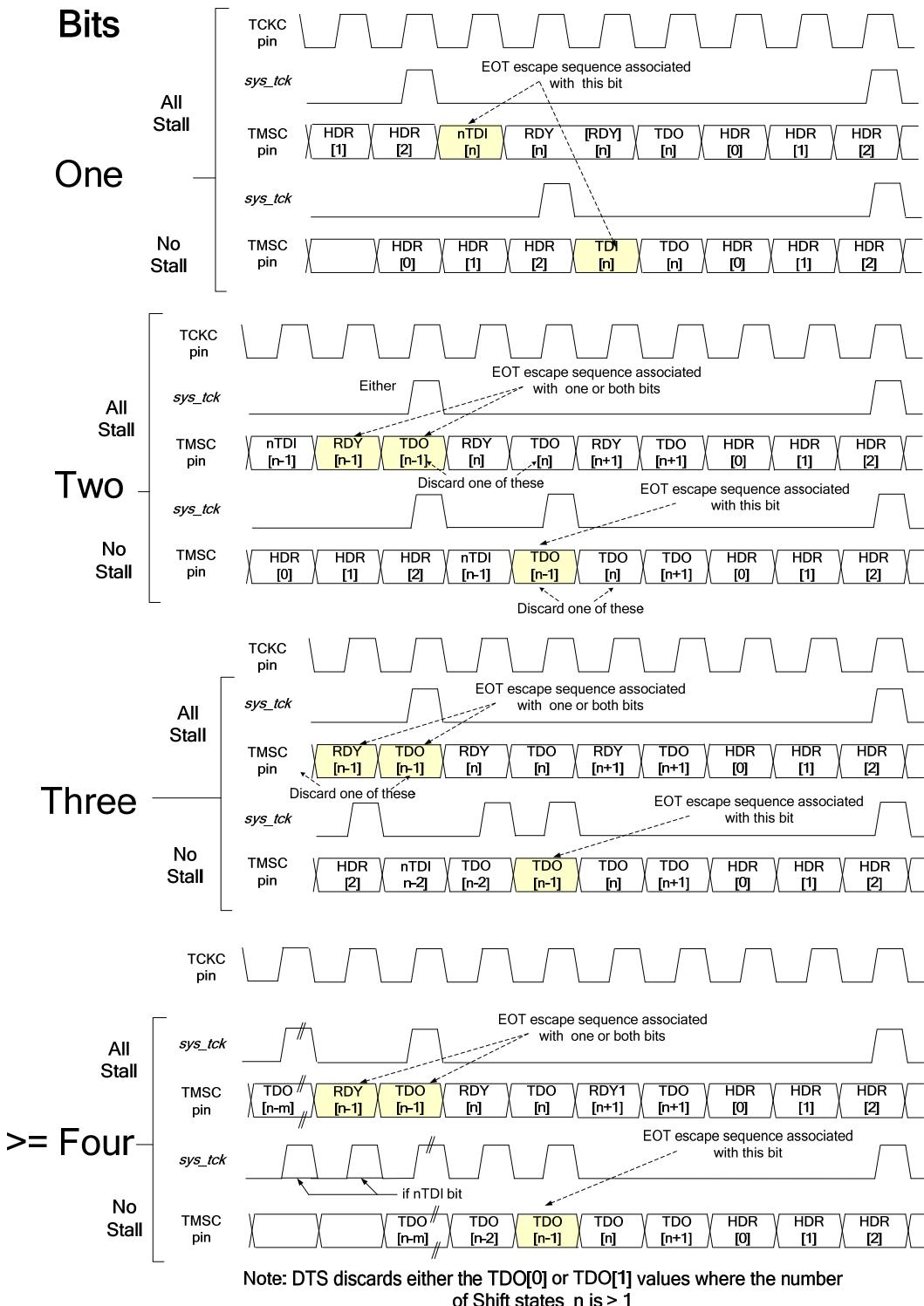
**Figure 26-22 — SScan0/1 TAPC state advance examples**



Note: The number of Shift states is n

NOTE—The EOT Escape is identified as occurring but is not shown.

**Figure 26-23 — SScan2/3 TAPC state advance examples for I/O and input-only segments**



NOTE—The EOT Escape is identified as occurring but is not shown.

**Figure 26-24 — SScan2/3 TAPC state advance examples for output-only segments**

## 26.8.2 Specifications

### Rules

- a) Each subsequent specification in 26.8.2 shall only apply to a T4 and above TAP.7 when using the SScan Scan Formats.
- b) The TAPC state shall be advanced by an SP, provided all of the following are true:
  - 1) The SP is not canceled by:
    - i) A qualified Deselection Escape.
    - ii) A qualified Selection Escape.
    - iii) A Selection Alert.
  - 2) The SP is not followed by a CP.
  - 3) The SP is not the last SP of a Data Segment (the advance is postponed in this case).
  - 4) Any of the following are true:
    - i) The SP is in a Control Segment.
    - ii) The scan format is either SScan0 or SScan1.
    - iii) The Data Segment type is not output only.
    - iv) The SP is not the next to last SP in the Data Segment.
- c) When the conditions specified by Rule 26.8.2 b) are met, the TAPC state shall be advanced coincidently with:
  - 1) The last bit within the payload of an SP provided the SP contains an output bit-frame.
  - 2) The bit period following the last bit of the SP payload, provided the SP does not contain an output bit-frame.
- d) The TAPC state shall be advanced coincidently with the HDR[2] bit of SP when the prior segment was a Data Segment.

## 26.9 CID allocation

### 26.9.1 Description

The CID cannot be allocated using the SScan Scan Formats. The MScan and OScan Scan Formats should be used for this purpose. The CIDA Command is treated as an NOP, and the CID and CIDI Register values are unchanged when using an SScan Scan Format.

### 26.9.2 Specifications

#### Rules

- a) Each subsequent specification in 26.9.2 shall only apply to a T4 and above TAP.7 when using the SScan Scan Formats.
- b) The CIDA Command shall be treated as a no-operation with no change to the CID and CIDI Register values.

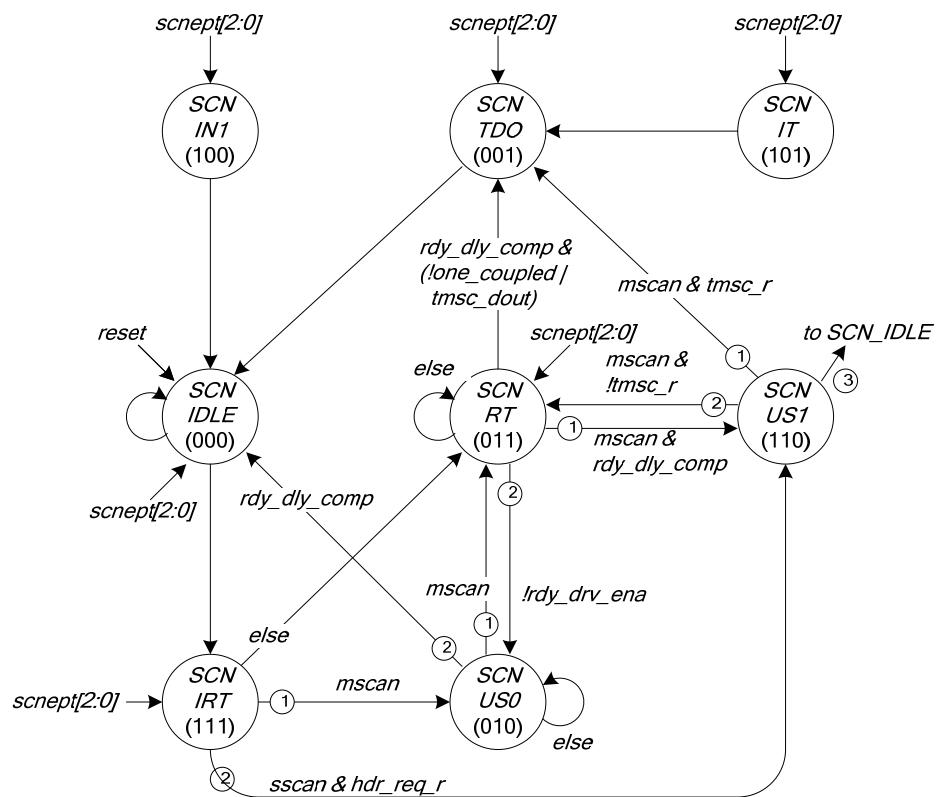
## 26.10 Increasing STL performance with SScan Scan Formats

The use of the *sys\_tck* signal duty-cycle adjustment to increase STL performance is not available with SScan Scan Formats.

## 26.11 An approach to implementing SScan Scan Formats

### 26.11.1 Payload State Machine

The management of the SScan Header is added to the conceptual Payload State Machine as shown in Figure 22-10. With the approach suggested in the following discussion, the Payload State Machine handles the function of both the Header and Payload State Machines shown in this figure. The Escape State Machine is added for SScan Scan Formats to assist the operation of the Payload State Machine. This machine handles EOT Escapes and END bits with SScan Scan Formats.



**Figure 26-25 — Conceptual view of the Payload State Machine with Header handling**

The Scan State Machine states are described in Table 26-17.

**Table 26-17 — Payload State Machine state definition**

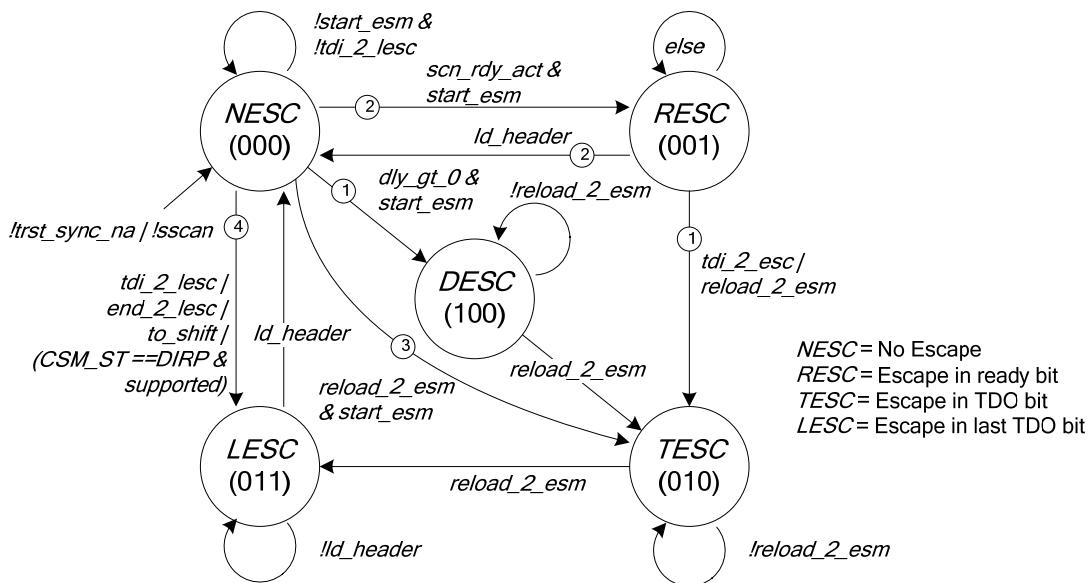
Mnemonic	scn_st[2:0]	Name	Description
<i>PAY_IDLE</i>	000b	Scan Idle	When all states are idle, load <i>tdi_adv_r</i> and <i>tms_adv_r</i> , when Escape/End State Machine is active, input first header bit
<i>PAY_TDO</i>	001b	Scan Input One	Load <i>tms_adv_r</i> , End of Payload Element.
<i>PAY_US0</i>	010b	Scan Input and TDO	Load <i>tms_adv_r</i> , and <i>PAY_TDO</i> is the next state.
<i>PAY_RT</i>	011b	Scan IN1, RDY, TDO	Load <i>tms_adv_r</i> , <i>PAY_RDY</i> is the next state followed by <i>PAY_TDO</i> .
<i>PAY_IN1</i>	100b	Scan Ready	If selected <i>PAY_TDO</i> when ready completes, else move to <i>PAY_US0</i> for
<i>PAY_US1</i>	101b	Scan Utility State Zero	Wait for RDY completion by other TAP.7 Controllers, <i>PAY_IDLE</i> is the next state when Escape/End State Machine is inactive, when Escape/End State Machine is active, input last header bit, advance TAPC state
<i>PAY_IT</i>	110b	Scan TDO	TDO bit on the TMSC signal, end of Payload Element, and <i>PAY_IDLE</i> is the
<i>PAY_IRT</i>	111b	Scan Idle	When all states are idle, load <i>tdi_adv_r</i> and <i>tms_adv_r</i> , when Escape/End State Machine is active, input first header bit

With the assistance of the Escape Detection State Machine, the Scan State Machine performs the following functions:

- Schedules a header following a Check Packet
- Schedules the header following a Control Segment
- Schedules the header following an EOT Escape in a Data Segment with SScan2 or SScan3
- Identifies the header bit positions
- Identifies the first payload /first bit to create the entry point for first SP stall profile

### **26.11.2 Escape Detection State Machine**

The synchronous detection of Escapes is shown in Figure 26-26.



**Figure 26-26 — Conceptual view of the Escape Detection State Machine**

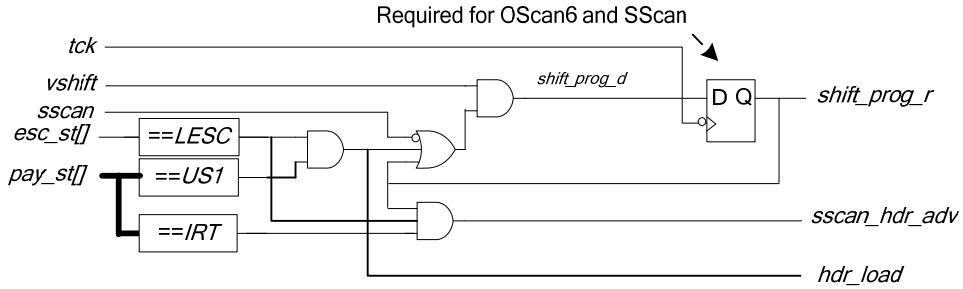
The Escape Detection State Machine states are described in Table 26-18.

**Table 26-18 — Escape Detection State Machine state definition**

Mnemonic	esc_st[2:0]	Name	Description
<i>NESC</i>	000b	No Escape	No Escape or end detected
<i>RESC</i>	001b	RDY Escape	Escape during RDY bit
<i>TESC</i>	010b	TDO Escape	Escape during TDO bit
<i>LESC</i>	011b	Last state of Escape	Last TDO bit detected, END bit == 1 detected.
<i>DESC</i>	100b	Delay Escape Detect	Delay occurrence of ESC detection.

### 26.11.3 Shift Progress flag

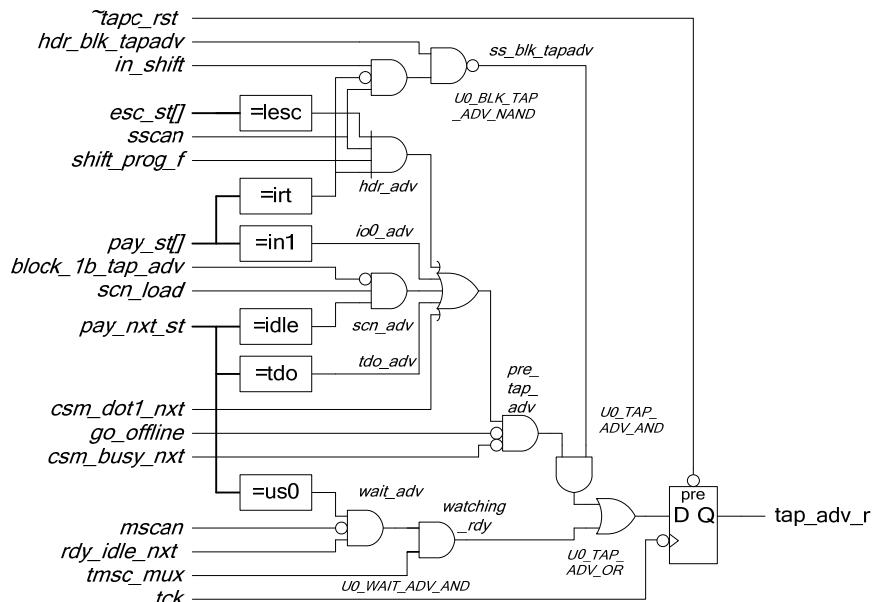
A conceptual view of the Shift Progress flag is shown in Figure 26-27. The operation of this flag is altered to support the use of the SScan Scan Formats as shown in this figure. The *vshift* signal cannot set the *shift\_prog\_r* signal to a logic 1 until the first *Shift-xR* header is completed. This ensures the TAPC state is advanced only when the virtual state is *Shift-xR* (and the real state is also *Shift-xR*) and a header follows a Data Segment.



**Figure 26-27 — SScan Shift Progress flag**

## 26.11.4 TAP advance

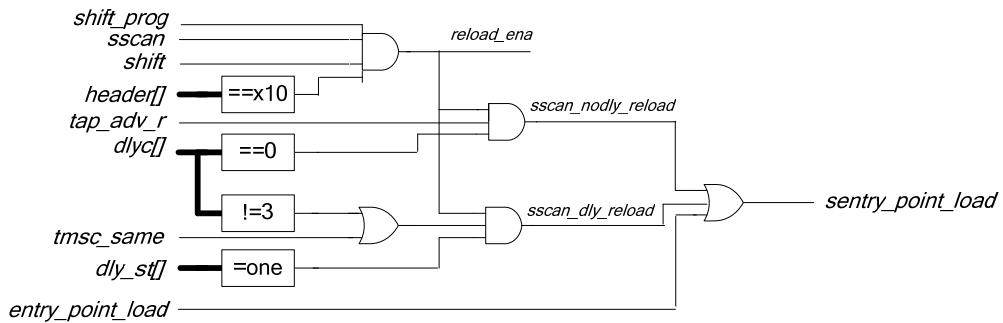
A conceptual view of the TAP TAPC advance enable (the *tap\_adv\_r* signal) is shown in Figure 26-28. Since the *csm\_adv\_nxt* signal comprehends a CP following an SP, the TAP advance is automatically inhibited for this case.



**Figure 26-28 — Conceptual view of TAP advance with MScan/OScan/SScan Scan Formats**

### 26.11.5 Additional entry point loads for output-only segments

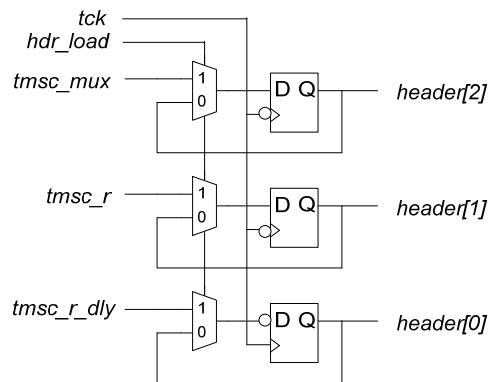
The output-only segments require additional entry point loads coincidentally with the TDO bit period when there is no Delay Element in the SP and coincidentally with the last bit of the Delay Element otherwise. This is shown in Figure 26-29.



**Figure 26-29 — SScan entry point reload additions for output only**

#### 26.11.6 Header Register

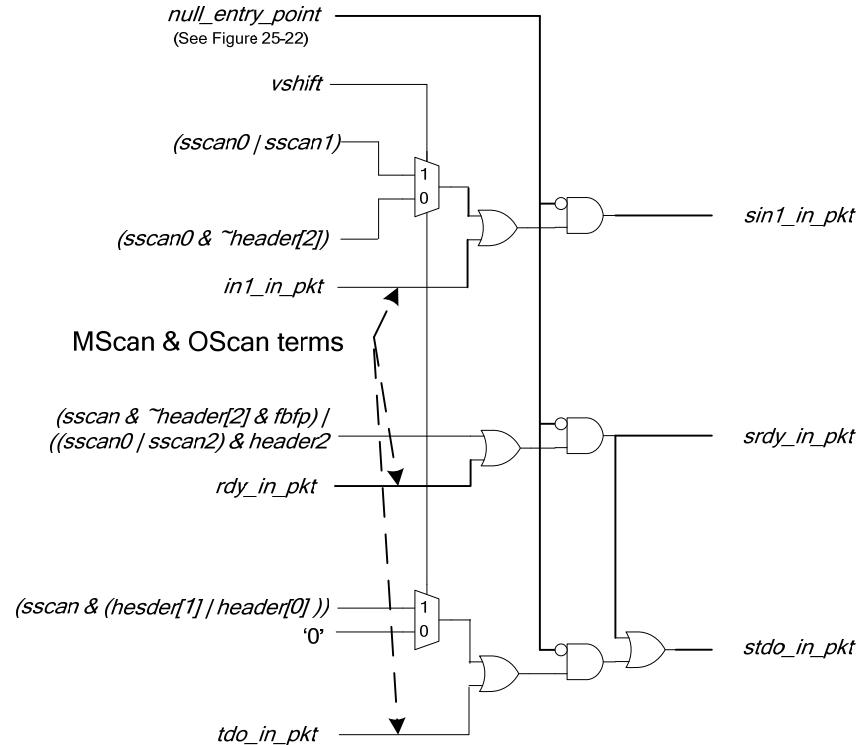
The Header Value Register, shown in Figure 26-30, does not require initialization as it is loaded before it is used.



**Figure 26-30 — Conceptual view of Header Register**

Additional Payload State Machine entry points are added with the SScan Scan Formats as shown in Figure 26-31. The entry point is forced to *PAY\_IDLE* at the first bit of an SP by:

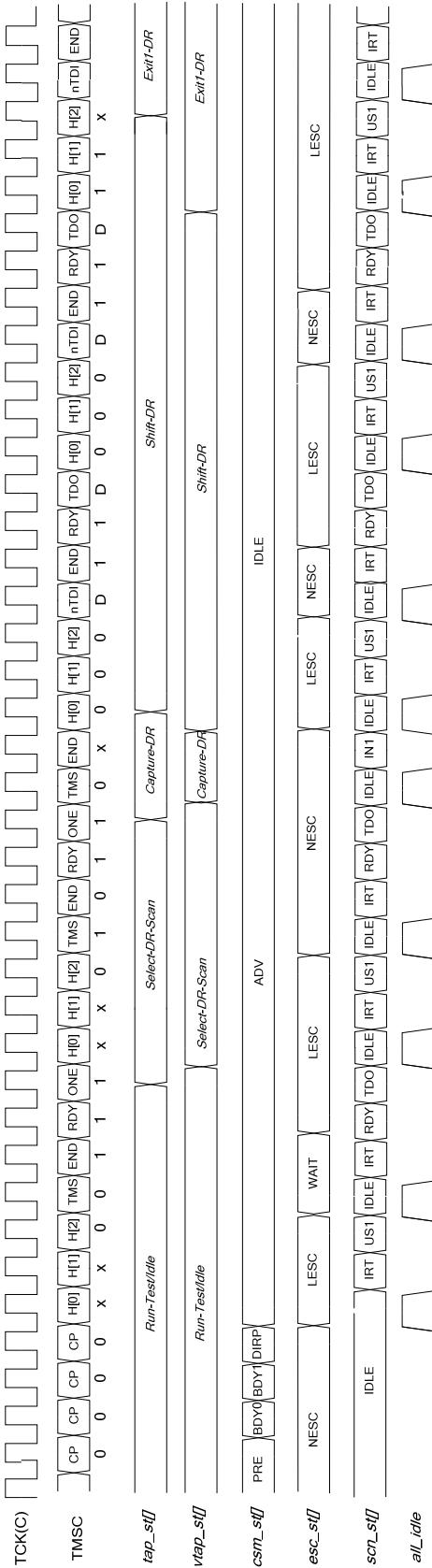
- A qualified Selection Escape
- A qualified Deselection Escape



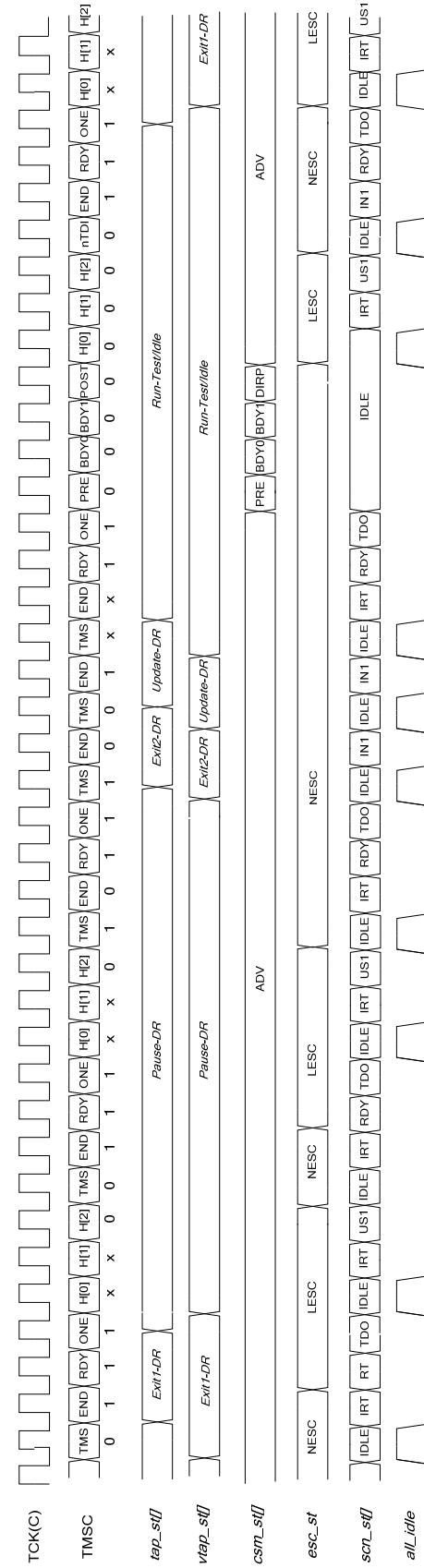
**Figure 26-31 — Conceptual view of Payload State Machine entry point with SScan support**

### 26.11.7 Timing diagrams

The joint operation of the Control, Scan, and Escape Detection State Machines is shown in Figure 26-32. The effects of a TAP.7 Controller command while using an SScan Scan Format is shown in Figure 26-33.



**Figure 26-32 — Joint operation of the state machines**



**Figure 26-33 — Joint operation of state machines with Command Part Two Update-DR**

## 26.12 Where to find examples

Examples of SScan scan transfers are shown in timing diagram form in Annex B and in tabular form in Annex C. These annexes contain examples of a three-bit DR Scan followed by a three-bit IR Scan. This transaction is the basis for all examples. The CP used following an SP generated for a specific scan format is also shown. The use of delays is also detailed. These examples may be used to compare the Advanced Protocol's behavior using each scan format with and without the use of a delay. The point at which the ADTAPC state is advanced is also shown. The pipelined and nonpipelined operation of the Advanced Protocol is also highlighted in these examples.

## 27. T5 TAP.7

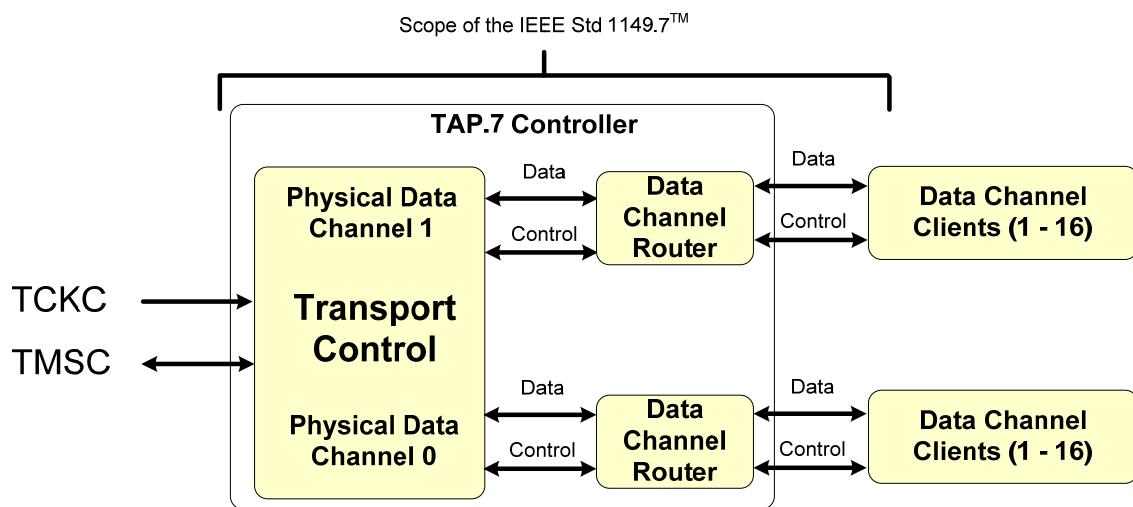
### 27.1 Introduction

This clause is applicable to T5 and above TAP.7s. It continues the description of the Data Transport Function provided in 6.3 and its drive policies described in Clause 14. It also provides the rules, permissions, and recommendations for the implementation of this function when combined with Clause 28. It also provides programming considerations.

A T5 TAP.7 Controller provides a flexible set of data transport capabilities that may be used with the Advanced Protocol. When the Data Transport Function is enabled, Transport Packets follow the Scan Packets that are associated with the TAPC states designated as supporting data transport. These states are specified at run time. Transport Packets move data between a DCC and another destination (the DTS and/or one or more DCCs) and control information between the DTS and one or more DCCs.

A conceptual view of the Data Transport Function is shown in Figure 27-1. It supports the implementation of up to two Physical Data Channels. Data Channel Clients implemented external to the TAP.7 Controller gain access to the TAP signals via one of these Physical Data Channels. From 1 to 16 Data Channel Clients may share the use of a PDC, but only 1 Data Channel Client may be connected to a PDC at any point in time. When multiple DCCs share the use of a PDC, a Data Channel Router (DCR) is placed between the PDC and the DCCs.

Physical Data Channels associated with one or more TAP.7 Controllers may be grouped to form an LDC. A Logical Data Channel provides for communication between Data Channel Clients connected to these Physical Data Channels, between the DTS and one or more Data Channel Clients, or both. Eight Logical Data Channels are supported. The DTS may allocate the bandwidth within a Transport Packet to the Logical Data Channels in any desired manner.



**Figure 27-1 — Conceptual view of T5 TAP.7 Transport Function**

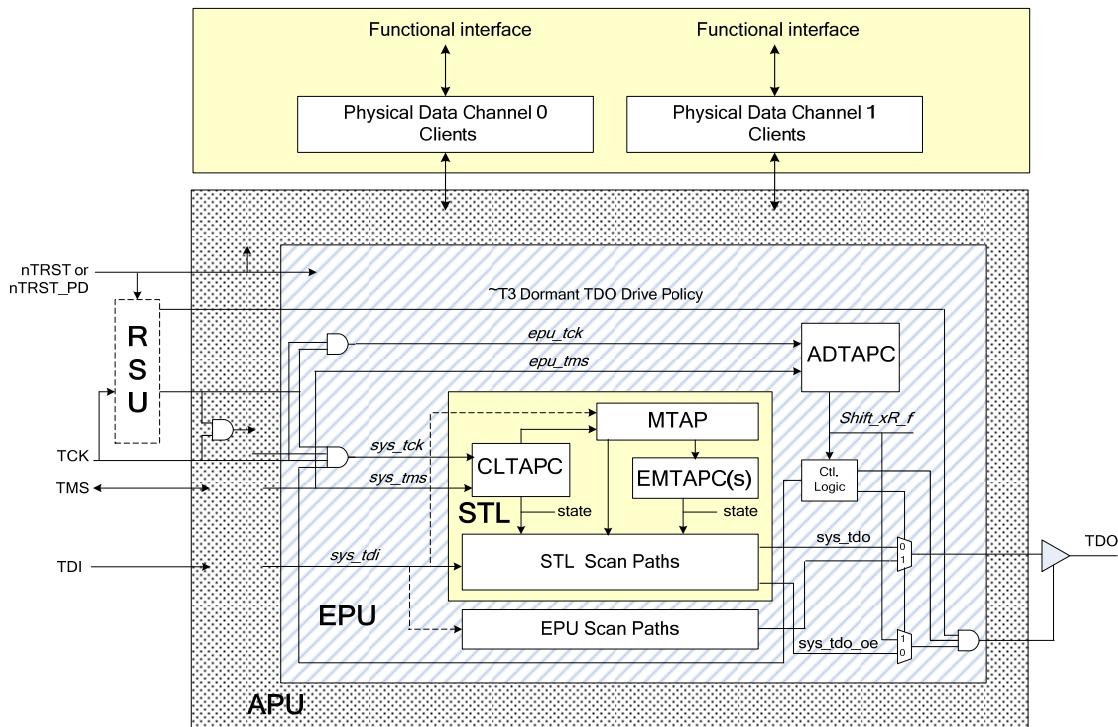
The subjects described in this clause are covered in the following order:

- 27.2 Deployment
- 27.3 Capabilities

- 27.4 Register and command portfolio
- 27.5 Configurations
- 27.6 Start-up behavior
- 27.7 Configuration Faults
- 27.8 Enabling transport
- 27.9 Transport Packet composition
- 27.10 Directive Elements
- 27.11 Register Elements
- 27.12 Data Elements
- 27.13 Selection of control and data targets
- 27.14 Data Channel Client functions
- 27.15 Partitioning of the Transport Control Function
- 27.16 Programming considerations
- 27.17 Aspects of transport not covered by this specification

## 27.2 Deployment

A high-level block diagram of the deployed T5 TAP.7 is shown in Figure 27-2.



**Figure 27-2 — Deployment of a T5 TAP.7 Controller**

## 27.3 Capabilities

### 27.3.1 Inherited

All mandatory capabilities of a T4 TAP.7 are mandatory for a T5 TAP.7. Optional capabilities supported by lower TAP.7 Classes are also optional for a T5 TAP.7.

The T4 TAP.7's capabilities inherited by a T5 TAP.7 ensure that it is:

- Controlled in the same manner as lower TAP.7 Classes
- With a wide configuration, compatible with a:
  - TAP.1 or TAP.7 Controller operating in a Series Scan Topology
  - T3 or T4 TAP.7 Controller operating in a Star-4 Scan Topology
  - T4 TAP.7 operating in a Star-2 Scan Topology
- With a narrow configuration, compatible with any T4 TAP.7 Controller operating in a Star-2 Scan Topology

### 27.3.2 New

The new T5 TAP.7 capabilities support the following:

- Physical Data Channel configurations:
  - None Data Channel Clients are not supported
  - One Data Channel Client(s) are attached to one PDC
  - Two Data Channel Client(s) are attached to both of the PDCs
- Types of data exchanges:
  - Single Client The DTS exchanges data with a single DCC
  - Multi-client The DTS exchanges data with multiple DCCs
  - Client to Client DCCs exchange data with each other and possibly the DTS

Synchronization with the transport portion of the Advanced Protocol bit sequences is maintained with each Physical Data Channel configuration.

## 27.4 Register and command portfolio

### 27.4.1 Description

#### 27.4.1.1 General information

The T5 TAP.7 Controller registers along with the commands and Transport Directives managing their contents are shown in Table 27-1. The new registers and their functions are shown in bold type. The STMC and STTPST Commands used to manage the TPPREV and TPST Registers are described in Table 9-3. All other registers related to the Transport Function are programmed using Transport Directives. The Transport Directives managing the contents of these registers are described in 27.10.

Note the register names without a separator (TPPREG and TPST) are managed with scan while register names with a separator (TP\_DELN, PDCx\_LCA, PDCx\_SEL, and PDCx\_DCC) are managed with Transport Directives. When a T5 TAP.7 register or a bit within it is not implemented, the register value is hardwired as a logic 0.

**Table 27-1 — T5 TAP.7 function/register/command relationships**

<b>Function</b>	<b>Register</b>	<b>Type</b>	<b>Associated command/directive</b>	<b>Described in subclause:</b>
Transport Protocol Revision	TPPREV	W	STMC	27.4.1.4
Transport States	TPST	W	STTPST	27.4.1.5
Data Element Length	TP_DELN	W	TP_DEL	27.4.1.6
Physical Data Channel/ Logical Channel Association	PDCx_LCA	W	TP_LCA TP_LCD	27.4.1.7
Physical Data Channel Selected for Directives	PDCx_SEL	W	TP_SLC, TP_SPC TP_SPM	27.4.1.8
Physical Data Channel DCC Selection	PDCx_DCC	W	TP_DCC	27.4.1.9
DCC Registers (0–7)	PDCx_DCCy_CRz	R/W	TP_CRR, TP_CRW	27.4.1.10

W == Writable    R == Readable

x == Physical Data Channel Number

y == Client number

z = Register number

### **27.4.1.2 Register acronyms**

The acronyms for the registers shown in Table 27-1 are shown below:

- TPPREV - Transport Protocol Revision
- TPST - Transport States
- TP\_DELN - Transport Data Element Length
- PDCx\_SEL - Physical Data Channel x SElect
- PDCx\_LCA - Physical Data Channel x Logical Channel Address
- PDCx\_DCC - Physical Data Channel x Data Channel Client
- PDCx\_DCCy\_CRz - Physical Data Channel x Data Channel Client y Control Register z

### **27.4.1.3 Effect of a Long-Form Selection Sequence**

The TPPREV, TPST, and TP\_DELN Registers are loaded with a Long-Form Selection Sequence while other registers in Table 27-1 are not.

### **27.4.1.4 Transport protocol revision (TPPREV) register**

The TPPREV Register provides a means to add other transport protocols in future revisions of the specification. A single protocol revision is supported with this specification revision. A T5 TAP.7 Controller is placed Offline when the use of an unsupported transport protocol is specified and the use of transport is enabled.

#### 27.4.1.5 Transport states (TPST)

The TPST Register defines the states where a TP follows an SP. Any combination of the *Run-Test/Idle*, *Pause-IR*, *Pause-DR*, *Update-IR*, and *Update-DR* states may be enabled to support transport. The Data Transport Function is disabled when none of these states are enabled to support transport.

#### 27.4.1.6 Data Element length (TP\_DELN)

The TP\_DELN Register defines the format of the Data Element used with data transfers. Four fixed-length formats and one variable-length format are supported. The format of the Data Element should be specified prior to activating a data transfer.

#### 27.4.1.7 Physical Data Channel to Logical Data Channel association (PDCx\_LCA)

The PDCx\_LCA Register defines the Logical Channel Address associated with the Physical Data Channel. This register value associates one of the eight Logical Channel Addresses (a valid LCA) with the PDC or identifies the case where a Logical Channel Address has not been associated with the Physical Data Channel, an invalid LCA. It also records whether one or more than one PDC has been allocated the LCA. This information affects TMSC Drive Policy during data exchanges. The PDC0\_LCA Register supports Physical Data Channel Zero while the PDC1\_LCA Registers supports Physical Data Channel One.

#### 27.4.1.8 Physical Data Channel selected (PDCx\_SEL)

The PDCx\_SEL Register defines whether the Physical Data Channel x responds to certain control directives. The PDC0\_SEL Register supports Physical Data Channel Zero while the PDC1\_LCA Register supports Physical Data Channel One. Its value also affects TMSC Drive Policy while reading Data Channel Client Registers. This information affects TMSC Drive Policy during data exchanges.

#### 27.4.1.9 Physical Data Channel DCC selection (PDCx\_DCC)

The PDCx\_DCC Register specifies the Data Channel Client connected to the Physical Data Channel when more than one Data Channel Client may be connected to the Physical Data Channel. This register is not implemented when a single Data Channel Client is connected to the Physical Data Channel.

#### 27.4.1.10 Physical Data Channel DCC Control Registers (PDCx\_DCCy\_CRz)

As many as eight DCC Control Registers (PDCx\_DCCy\_CR0–PDCx\_DCCy\_CR7) may be implemented with each DCC. These registers are managed by the DTS. DCC Control registers zero and one are defined by this standard while DCC Control Registers two through seven are user defined.

The function of the writable DCC Register bits defined by this standard are described briefly below:

- Control Register Zero:
  - DCCy\_ENAR Requests the initialization of the DCC when a logic 0, requests enabling of the DCC when a logic 1
  - DCCy\_DRNR Requests the Data Channel Client stop sending information to the host in preparation for deselection
- Control Register One:
  - DCCy\_DIRR Requests the Data Element transfer direction when the DCC data transfer direction is programmable

The function of the readable DCC Register bits defined by this standard are described briefly as follows:

- Control Register Zero:
  - DCCy\_PRES Indicates whether the DCC is implemented.
  - DCCy\_ENAA Acknowledges the enable request generated by DCCy\_DRNR.
  - DCCy\_DRNA Acknowledges the drain request generated by DCCy\_DRNR.
  - DCCy\_ENASUP Indicates whether the DCCy\_ENAR bit is implemented.
  - DCCy\_DRNSUP Indicates whether the DCCy\_DRNR bit is implemented.
  - DCCy\_PARK Indicates whether the TMSC signal should be parked at a logic 1 or logic 0 when this signal is not being used by either the DTS or the TS during a TP.
- Control Register One:
  - DCCy\_DIRA Informs the Physical Data Channel of the Data Element transfer direction when the transfer direction is either programmable or fixed.

## 27.4.2 Specifications

### Rules

- a) Each subsequent specification in 27.4.2 shall only apply to a T5 TAP.7.
- b) The meaning of the value of registers added within the T5 TAP.7 to support transport functionality shall be governed by Table 27-2.
- c) The meaning of the values of DCC Control Registers 0–7 managed by the TP\_CRR and TP\_CRW Directives shall be governed by Table 27-3 and Table 27-4, respectively.

**Table 27-2 — T5 TAP.7 register descriptions**

<b>T5 TAP.7 Register Descriptions</b>	
<b>Logical Channel Address (PDCx_LCA) – Channel association</b>	
YYYYY	yyyyy=0xxxx:PDCx is not allocated an LCA. 1xxxx:The LCA is allocated to only this PDC. 1xxxxx:The LCA may be allocated to more than one PDC. 1x000:PDCx has been allocated LCA 0. 1x001:PDCx has been allocated LCA 1. 1x010:PDCx has been allocated LCA 2. 1x011:PDCx has been allocated LCA 3. 1x100:PDCx has been allocated LCA 4. 1x101:PDCx has been allocated LCA 5. 1x110:PDCx has been allocated LCA 6. 1x111:PDCx has been allocated LCA 7.
<b>PDC x Data Channel Client (PDCx_DCC) – DCC/PDC connection</b>	
YYYY	yyyy=0 – 15:Data Channel Client connected to PDC x Not implemented if a single Data Channel Client is connected to a PDC.
<b>PDC x Selection (PDCx_SEL) – Specifies directive targets</b>	
YY	yy= 00:Not selected. 01:The PDC has been selected with the TP_SLC Directive. 10:The PDC has been selected with the TP_SSP Directive without a subsequent TP_SMP Directive. 11:The PDC is selected with either the TP_SSP or TP_SMP Directive with the last selection directive being a TP_SMP.
<b>TP Data Element Length (TP_DELN) – Data Element length</b>	
YYY	yyy= 0:8-bit fixed length. 1:16-bit fixed length. 2:32-bit fixed length. 3:64-bit fixed length. 4-7:Variable length.
<b>Transport Protocol Revision (TPPREV)</b>	
YY	yy= 0:Transport protocol revision used by this specification revision. 1-3:Reserved protocol revisions in this specification revision, causes Offline operation.
<b>Transport Protocol State (TPST)</b>	
YYYYY	yyyyy=00000:Transport is disabled. 0xxxx:Run-Test/Idle state does not support transport. 1xxxx:Run-Test/Idle state supports transport. x0xxx:Pause-IR state does not support transport. x1xxx:Pause-IR state supports transport. xx0xx:Pause-DR state does not support transport. xx1xx:Pause-DR state supports transport. xxx0x:Update-IR state does not support transport. xxx1x:Update-IR state supports transport. xxxx0:Update-DR state does not support transport. xxxx1:Update-DR state supports transport.

NOTE 1—An x within a register or command name in this table is a placeholder for a 0 or 1 and represents the PDC number.

**Table 27-3 — PDCx\_DCCy\_CR0 – PDCx\_DCCy\_CR7 write descriptions**

Bit	Control Register (CR0) Write Description	
<b>Reserved</b>		
0	–	This bit should be written as a logic 0.
<b>DCC y ENABLE (DCCy_ENAR) – Enable DCCy request</b>		
1	v	v=0:Data Channel Client y is reset. 1:Data Channel Client y is enabled.
<b>DCC y Drain (DCCy_DRNR) – Drain DCCy request</b>		
2	v	v=0:No drain request is asserted. 1:Data Channel Client is requested to prepare for deselection without causing adverse side effects.
<b>Reserved</b>		
31:02	–	These bits are reserved and should be written as a logic 0.
Bit	Control Register One (CR1) Write Description	
<b>DCC Transfer Direction (DCCy_DIRA) –Direction Request</b>		
01:00	vv	vv=0:Custom data transfer. 1:Bidirectional data transfer. 2:DTS-to-DCC data transfer (unidirectional). 3:DCC-to-DTS data transfer (unidirectional).
<b>Reserved</b>		
31:02	–	These register bits are reserved and should be written as a logic 0.
Bit	Control Register Two – Seven Write Descriptions	
<b>User defined</b>		
31:00	–	These register bits are user defined. Unimplemented bits are written as a logic 0.

NOTE 2—A letter y within a bit name in this table is a placeholder for the DCC number.

**Table 27-4 — PDCx\_DCCy\_CR0 – PDCx\_DCCy\_CR7 read descriptions**

Bit	Control Register Zero (CR0) Read Descriptions	
<b>DCC Present (DCCy_PRES) – Client is implemented</b>		
0	v	v=0:Data Ch. Client y is not implemented. 1:Data Ch. Client y is implemented.
<b>DCC Enable Return (DCCy_ENAA) – Enable Acknowledge</b>		
1	v	v=0:Data Ch. Client y is disabled (being reset). 1:Data Ch. Client y is enabled.
<b>DCC Drain Acknowledge (DCCy_DRNA) – Drain Acknowledge</b>		
2	v	v=0:Deselection will not cause adverse side effects. 1:Deselection may cause adverse side effects.
<b>DCC Enable Supported (DCCy_ENASUP) – Enable function is supported</b>		
3	v	v=0:DCCy_ENAR register bit is not supported. 1:DCCy_ENAR register bit is supported.
<b>DCC Drain Supported (DCCy_DRNSUP) – Drain function is supported</b>		
4	v	v=0:DCCy_DRNR register bit is not supported. 1:DCCy_DRNR register bit is supported.
<b>DCC TMSC Park (DCCy_PARK) – TMSC parking value for DCC</b>		
5	v	v=0:Park TMSC at logic 0 before a data transfer. 1:Park TMSC at logic 1 before a data transfer.
<b>Reserved</b>		
31:06	--	Reserved – Read as a logic 0.
Bit	Control Register One (CR1) Read Descriptions	
<b>DCC Transfer Direction (DCCy_DIRA) – Data Element Transfer Direction</b>		
00:01	vv	vv=0:Custom data transfer. 1:Bidirectional data transfer. 2:DTS-to-DCC data transfer (unidirectional). 3:DCC-to-DTS data transfer (unidirectional).  See description in 27.12.1.3.
31:02	--	Reserved – Read as a logic 0.
Bit	Control Register Two – Seven Register Read Descriptions	
<b>User defined</b>		
31:00	-	These register bits are user defined. Unimplemented bits are read as a logic 0.

- d) Unimplemented DCC Register bits shown in Table 27-4 shall be read as a logic 0.
- e) The reset values of the Transport Registers managed with directives shall be governed by Table 27-5.

**Table 27-5 — Reset values of TAP.7 Controller registers managed with Transport Directives**

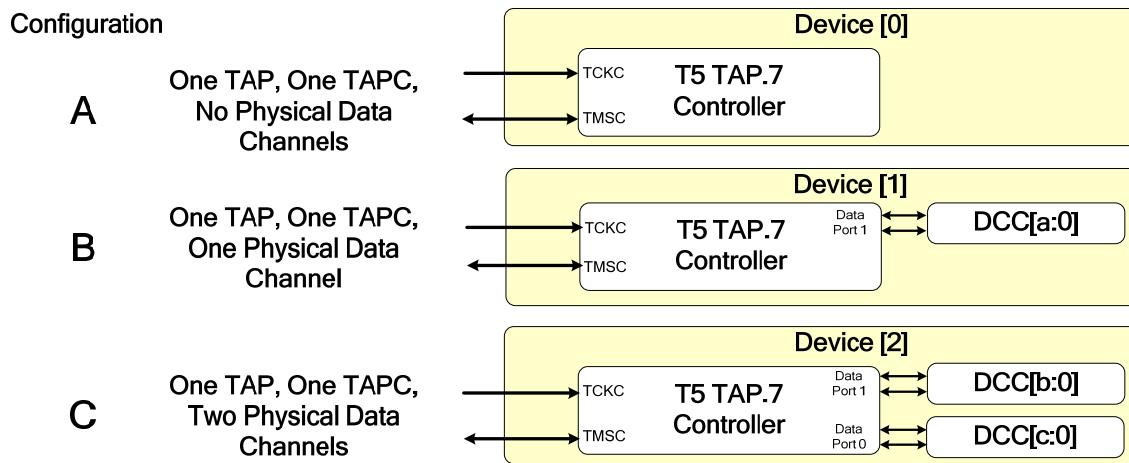
Register mnemonic	Register width	Bit name	Values for reset type	
			0–4	5
TP_DELN	4	N/A	0000	NC
PDCx_LCA	5	N/A	00000	
PDCx_DCC	4	N/A	0000	
PDCx_SEL	2	N/A	00	
PDCx_DCCy_CR0	2	DCCy_PRES	As specified in Table 27-4	
	1	DCCy_ENA, DCCy_ENAR	0	
	1	DCCy_DRNR, DCCy_DRNA	0	
PDCx_DCCy_CR1	2	DCCy_DIR, DCCy_DIRA	Lowest number supported value beginning with 00	

NOTE 3—See Table 9-2 for information regarding the initialization of the TPPREV and TPST Registers.

## 27.5 Configurations

### 27.5.1 Description

The T5 TAP.7 configurations allowed by the standard are shown in Figure 27-3.



**Figure 27-3 — Conceptual view of T5 TAP.7 Controller configurations**

The options fields of Configuration Register One (see Table 9-17) identify the number of Physical Data Channels supported (none, one, or more than one) and, for each supported Physical Data Channel, whether there is one or more Data Channel Clients connected to that Physical Data Channel.

As stated in 6.3.1, a chip architect should consider a T5 TAP.7 that is implemented with zero PDCs in lieu of a T4 TAP.7, if it is believed the chip may be deployed in systems with other T5 TAP.7s with PDCs. This configuration provides the functionality of a T4 TAP.7 Controller but with the added benefit of the TAP.7 Controller never being placed Offline when the DTS enables data transport.

## 27.5.2 Specifications

### Rules

- a) Each subsequent specification in 27.5.2 shall only apply to a T5 TAP.7.
- b) The Physical Data Channel configuration shall be described by the PDC0SUPS and PDC1SUPS bits in Configuration Register 1 as shown in Table 9-16.
- c) Support for the optional TAP.7 functions shown in Table 27-6 shall be described with the Configuration Register One bits listed in this table.

**Table 27-6 — Specifying the use of T5 TAP.7 optional functions**

Optional functions	Registers implemented to support function	Support identified with Configuration Register 1 bits:
<b>Physical Data Channel 0</b>	<b>Table 27-7, Table 27-8</b>	<b>PDC0SUPS[1:0]</b>
<b>Physical Data Channel 1</b>		<b>PDC1SUPS[1:0]</b>

- d) The implementation of TAP.7 Controller Transport Registers shall be governed by Table 27-7.

**Table 27-7 — T5 TAP.7 Controller Transport Register configurations**

PDCxSUPS	Number of DCCs	Registers implemented to support configuration		
		TPPREV, TPST TP_DEL	PDCx_LCA, PDCx_SEL,	PDCx_DCC
00	None		No	No
01	One	Yes	Yes	No
10	More than one		Yes	Yes

- e) The implementation of the PDCx\_DCC\_CR0–PDCx\_DCC\_CR7 Registers accessible from the TAP.7 Controller shall be governed by Table 27-8.

**Table 27-8 — DCC register configurations**

Register	View	Registers implemented to support DCC operation
PDCx_DCC_CR0 - PDCx_DCC_CR1	Read	As specified by rules in this subclause
	Write	
PDCx_DCC_CR2 - PDCx_DCC_CR7	Read	Optional
	Write	Optional

- f) The implementation of the PDCx\_DCCy\_CR0 and PDCx\_DCCy\_CR1 bits described in Table 27-8 shall be mandatory when a DCC is implemented unless stated otherwise.
- g) The DCCy\_ENAR and DCCy\_ENAR bits shown in Table 27-3 and Table 27-4 shall be implemented, provided the DCCy\_ENASUP bit indicates the enable function is supported, and not be implemented otherwise.

- h) The DCCy\_DRNR and DCCy\_DRNA bits shown in Table 27-3 and Table 27-4 shall be implemented, provided the DCCy\_DRNSUP bit indicates the drain function is supported, and not be implemented otherwise.
- i) The DCCy\_DIRR bits shown in Table 27-3 shall be implemented, provided the Data Element Transfer Direction is programmable and not be implemented otherwise.

## 27.6 Start-up behavior

### 27.6.1 Description

The start-up behavior of a T5 TAP.7 is the same as that for a T4 TAP.7 (see 23.6).

### 27.6.2 Specifications

#### Rules

- a) Each subsequent specification in 27.6.2 shall only apply to a T5 TAP.7.
- b) Start-up behavior shall be that specified by one of the following start-up options, subject to the behavior created by the implementation of the Power-Mode Function.
  - 1) IEEE 1149.1-Compliant.
  - 2) IEEE 1149.1-Compatible.
  - 3) IEEE 1149.1-Protocol Compatible.
  - 4) Offline-at-Start-up.

## 27.7 Configuration Faults

### 27.7.1 Description

A T5 TAP.7 Controller built in accordance with this specification revision declares a Configuration Fault when using an Advanced Scan Format:

- The SCNFMT Register specifies the use of an unsupported scan format (same as T4 TAP.7)
- All of the following are true:
  - Data Transport is enabled
  - For this specification revision, a nonzero TPPREV Register value

### 27.7.2 Specifications

#### Rules

- a) Each subsequent specification in 27.7.2 shall only apply to a T5 TAP.7 while using the Advanced Protocol.
- b) For a TAP.7 Controller built to this specification revision, the combination of a nonzero TPPREV Register value combined with a nonzero TPST Register value shall be considered as specifying a Configuration Fault.
- c) Placing the TAP.7 Controller Offline shall suspend the operation of the Transport Control Function in a manner that allows this function to be resumed, provided the following operations occur:
  - 1) Enabling the use of transport with the TPST Register as specified in Table 27-2.

- 2) For this specification revision, setting the TPPREV Register value to zero.

## 27.8 Enabling transport

### 27.8.1 Description

The DTS designates any combination of the virtual TAPC states listed as follows (including none) as supporting transport using the TPST Register:

- *Run-Test/Idle*
- *Pause-IR*
- *Pause-DR*
- *Update-IR*
- *Update-DR*

This programmability provides the DTS with means to:

- Enable/disable the use of transport
- Choose the TAPC states where a DTS desires to support transport
- Create more or less opportunities for LDC transfers
- Prevent the use of TPs where their use would cause the malfunction of either:
  - An IEEE 1149.1 TAPC within the STL with IEEE 1149.1-Non-disruptive Behavior
  - A function controlled by this TAPC

The TPST Register value is one-hot encoded, with one bit for enabling transport with each state listed above. The Transport Function is disabled with an all-zeros value, which is the value created by a Type-0–Type-4 Reset.

There should be no restrictions on the use of TPs with any of the TAPC states supporting transport when all TAPCs within the STLs provide IEEE 1149.1-Specified Behavior. The DTS typically uses the states listed above or the states following them as a spin state or stopping point after completing a scan operation.

When transport is enabled and the TPPREV Register value specifies a supported protocol revision, an SP/TP combination is created for TAPC states designated as supporting transport, except when preempted by the creation of an SP/CP combination. Recall that an *SPA/CPA* state sequence created by a register-write while using the Advanced Protocol preempts an *SPA/TPA* state sequence.

The TAPC state associated with the SP preceding the TP is shown in Table 27-9 along with the TAPC state that exists when the TP ends. This table may be of some use to DTS software that restricts the use of TPs with TAPC states where there may be incompatibilities with EMTAPCs deployed in the STL.

### 27.8.2 Specifications

#### Rules

- a) Each subsequent specification in 27.8.2 shall only apply to a T5 TAP.7.
- b) A nonzero TPST Register value shall cause the *TPA* state to follow the *SPA* state (a TP follows an SP) for the TAPC states designated as supporting transport, provided a transition from the *SPA* to the *CPA* state (an SP/CP combination) does not preempt the *TPA* state following the *SPA* state.

- c) The TAPC state associated with the *SPA* state preceding a *TPA* state shall be governed by Table 27-9.

**Table 27-9 — Real TAPC state and transport relationships**

A TP follows the SP for this TAPC state:	The value of the TMS bit within the SP:	The TP ends with the TAPC state:
<i>Run-Test/Idle</i>	logic 0	<i>Run-Test/Idle</i>
	logic 1	<i>Select-DR-Scan</i>
<i>Update-xR or Update-IR</i>	logic 0	<i>Run-Test/Idle</i>
	logic 1	<i>Select-DR-Scan</i>
<i>Pause-IR</i>	logic 0	<i>Pause-IR</i>
	logic 1	<i>Exit2-IR</i>
<i>Pause-DR</i>	logic 0	<i>Pause-DR</i>
	logic 1	<i>Exit2-DR</i>

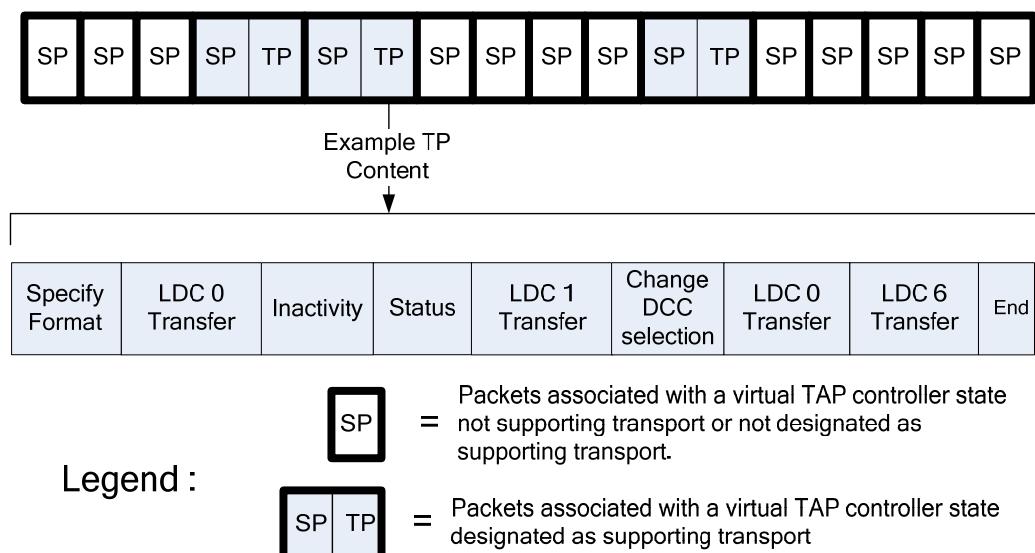
## 27.9 Transport Packet composition

### 27.9.1 Operations

A TP may include a mix of operations creating:

- Changes in TAP.7 Controller register values
- Changes in Data Channel Client register values
- A Data Exchange with the DTS, between Data Channel Client(s), or both
- Periods of inactivity

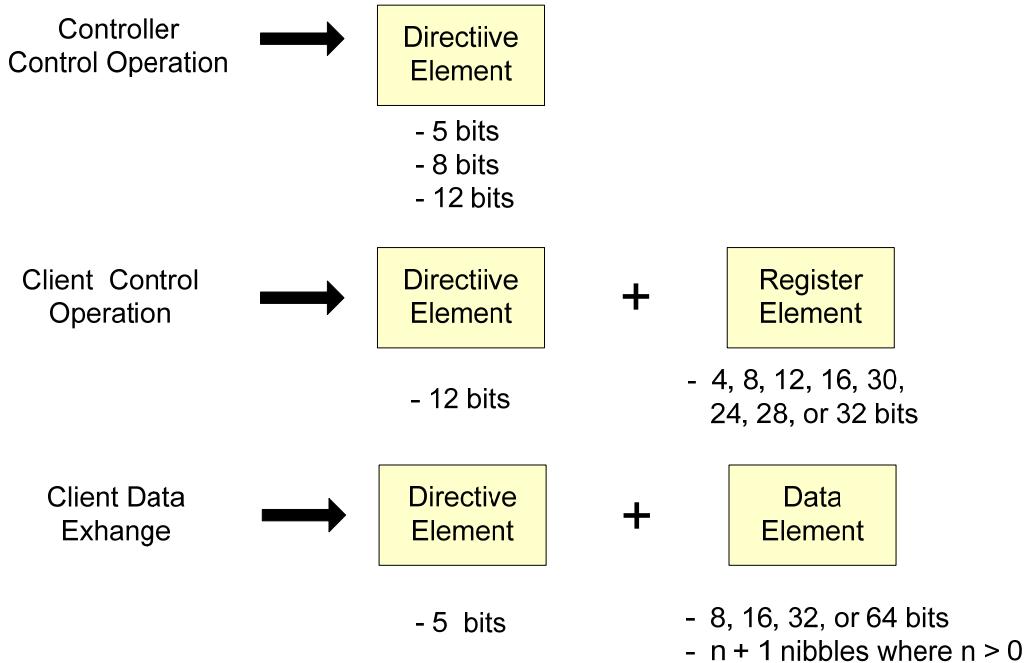
This is shown in Figure 27-4.



**Figure 27-4 — TP placement and content**

## 27.9.2 Directive, Register, and Data Elements

There are three types of transport operations, Controller Control Operations, Client Control Operations, and Client Data Exchanges. These operations are created with combinations of Directive Elements, Register Elements, and Data Elements as shown in Figure 27-5.



**Figure 27-5 — Transport Packet Elements**

A Transport Directive that either terminates the TP or is followed by another Transport Directive performs a Controller Control Operation. The combination of a Transport Directive followed by a Register Element performs a Client Control Operation. The Register Element moves control information to or from the DCC's PDCx\_DCCy\_CR0 – PDCx\_DCCy\_CR7 Registers. The combination of a directive followed by a Data Element performs a Client Data Exchange. A Data Element moves data information between the DTS and one or more DCCs or between DCCs. The DTS and DCC(s) may utilize the Data Element in any manner including sending no information. Each Transport Packet contains at least one Directive Element (to end the Transport Packet).

## 27.10 Directive Elements

### 27.10.1 Description

#### 27.10.1.1 Overview

Transport Directives are:

- Generated by the DTS
- Used by TAP.7 Controller
- Invisible to the DCC(s)

These directives:

- Perform the following controller control operations:
  - No operation
  - End the TP
  - Set the Data Element Length
  - Reset generation
  - Write certain controller registers
  - Allocate and de-allocate Logical Channel Addresses
  - Connect a DCC to a PDC
  - Select PDCs that are affected by certain subsequent directives
- Initiate Data Channel Client control functions:
  - Read DCC Registers
  - Write DCC Registers
- Initiate Client Data Exchanges

### 27.10.1.2 Directive Element acronyms

The Transport Directives and their acronyms are listed below:

— <b>TP_NOP: No Operation</b>	Performs a no-operation
— <b>TP_END: End</b>	Ends a Transport Packet
— <b>TP_LCA: Logical Channel Address Allocate</b>	Allocates the LCA
— <b>TP_LCD: Logical Channel Address De-allocate</b>	De-allocates Logical Channel Address
— <b>TP_DEL: Data Element Length</b>	Defines the Data Element Length
— <b>TP_DCx: Data Channel Transfer</b>	Initiates a data channel transfer with DCC x
— <b>TP_SSP: Select Single Physical</b>	Selects a single PDC, deselects other PDCs
— <b>TP_SMP: Select Multiple Physical</b>	Selects a PDC without deselecting other PDCs
— <b>TP_SLC: Select Logical Channel</b>	Selects the Logical Channel
— <b>TP_DCC: Data Channel Client</b>	Selects the DCC connected to the PDC
— <b>TP_CRR: Data Channel Client Register Read</b>	Reads a DCC Register
— <b>TP_CRW: Data Channel Client Register-Write</b>	Writes a DCC Register
— <b>TP_RSO: Reset Ones</b>	Generates a Type-3 Reset
— <b>TP_RSZ: Reset Zeros</b>	Generates a Type-3 Reset

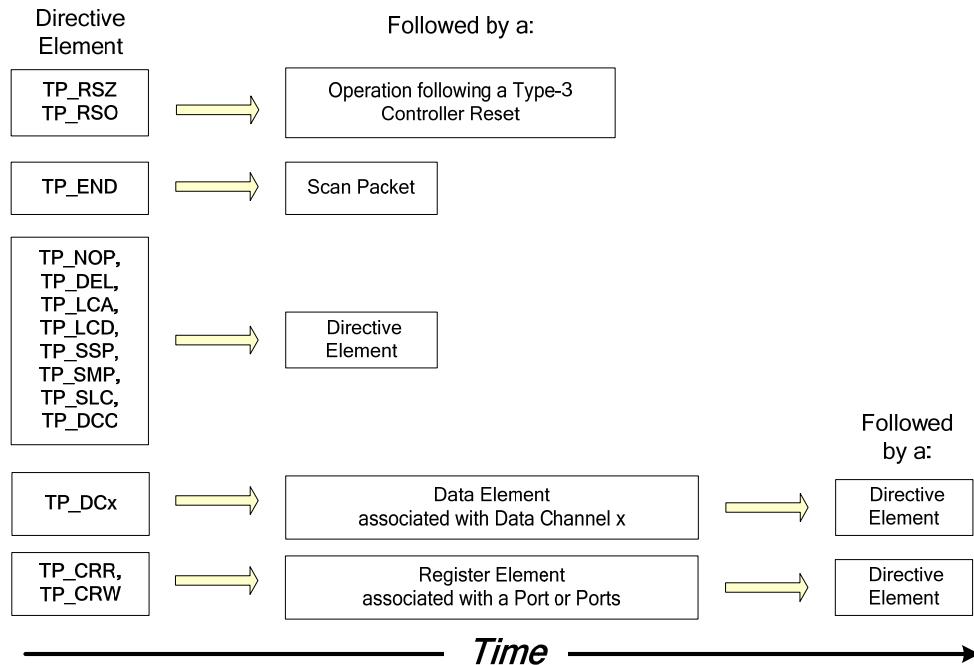
### 27.10.1.3 Surrounding context of directives

The Transport Directive Elements and the information following them are shown in Figure 27-6. A TP of any length may be formed with any sequence of the:

- TP\_DCx Directive Element and the Data Element that follows it
- TP\_CRW and TP\_CRR Directive Elements along with the Register Element that follows them

- A mix of other Directive Elements that do not terminate the TP and are not followed by either a Data Element or a Register Element

This sequence (or no sequence at all) may be followed by a TP\_END, TP\_RSZ, or TP\_RSO Directive to terminate the TP. A TP that is not infinite always has a TP\_END, TP\_RSZ, or TP\_RSO Directive.



**Figure 27-6 — TP Directive Elements and their surrounding context**

#### 27.10.1.4 Directive/transport building block relationships

The relationship of Directive Elements and the Transport Function building blocks is shown in Figure 27-7.

Building block ➡	Transport Protocol Processing	Physical Data Channel	Data Channel Router	Data Channel Client		
Unconditional Directives ➡	TP_NOP, TP_END, TP_DEL, TP_RSZ TP_RSO	None	None	None		
Conditional Directives ➡	None	TP_SSP, TP_SMP, TP_SLC  Selection directives changing PDCx_SEL  TP_LCA  Control action requires PDC selection with selection directives (PDCx_xSEL !=0b)  TP_LCD  Control action based on LCA	TP_DCC  Control action requires PDC selection with selection directives (PDCx_SEL !=0b & PDCx_DCC == client #)	TP_CRW  Control transfer requires physical or logical channel selection & PDCx_DCC value selecting client (PDCx_SEL !=0b & PDCx_DCC == client #)	TP_CRR  Control transfer requires the last selection directive to be TP_SSP (PDCx_SEL == 10b & PDCx_DCC == client #)	TP_DCx  Data transfer requires logical channel selection with this directive & PDCx_DCC value selecting client  (The x in TP_DCx specifies LCA & PDCx_DCC == client number)

**Figure 27-7 — Directive/transport building block relationships**

#### 27.10.1.5 Directive encoding

The encoding of Transport Directives is shown in Table 27-11. The nomenclature used with this encoding is shown in Table 27-10.

**Table 27-10 — Transport Directive nomenclature**

Mnemonic	Description
CCC =	Logical Channel Address (LCA[2:0]).
D =	Data (with the LSB in the lowest numbered bit position).
III =	Controller Identification Number (CID[3:0]).
LLL =	Represents the number of nibbles minus 1 forming the Register Element that follows the directive (the number of nibbles = LLL + 1).
P =	Parking value of the TMSC pin before a Data Element begins.
RRR =	The DCC Register number (0–7) accessed during the following Register Element.
x =	Don't care.
Y =	Physical Data Channel number (0 or 1).

**Table 27-11 — Transport Directive encoding**

Dir.	Type	Format			Function	Next Element or other
		MSB	Bit#	LSB		
		1100	0000	0000		
		1098	7654	3210		
<b>TP_RSZ</b>	Uncond.	0000 0000			Reset.	None (RST)
<b>TP_END</b>	Uncond.	1 0000			Transport Packet End.	None (SP)
<b>TP_CRR</b>	Cond.	LLL R RR10 0000			Read a client register.	Control
Rsvd	--	xxxx x100 0000			Reserved, no operation.	Dir.
<b>TP_SSP</b>	Select	IIII Y001			Select a single PDC.	Dir.
<b>TP_DCO</b>	Xfer.	P 0010			Initiate LDC 0 transfer.	Data
<b>TP_DC1</b>	Xfer.	P 0011			Initiate LDC 1 transfer.	Data
<b>TP_DC2</b>	Xfer.	P 0100			Initiate LDC 2 transfer.	Data
<b>TP_DC3</b>	Xfer.	P 0101			Initiate LDC 3 transfer.	Data
<b>TP_SMP</b>	Select	IIII Y110			Add a PDC selection.	Dir.
<b>TP_DEL</b>	Uncond.	DDDO 0111			Store the TP_DELN Reg.	Dir.
<b>TP_SLC</b>	Select	CCC1 0111			Select an LDC.	Dir.
<b>TP_DCC</b>	Cond.	DDDD 1000			Store PDCx_DCC Reg.	Dir.
<b>TP_DC4</b>	Xfer.	P 1010			Initiate LDC 4 transfer.	Data
<b>TP_DC5</b>	Xfer.	P 1011			Initiate LDC 5 transfer.	Data
<b>TP_DC6</b>	Xfer.	P 1100			Initiate LDC 6 transfer.	Data
<b>TP_DC7</b>	Xfer.	P 1101			Initiate LDC 7 transfer.	Data
<b>TP_NOP</b>	--	0 1111			No operation.	Dir.
<b>TP_CRW</b>	Cond.	LLL R RR01 1111			Write a client register.	Control
Rsvd	--	xxxx x011 1111			Reserved, no operation.	Dir.
<b>TP_LCA</b>	Cond.	CCC0 0111 1111			Allocate an LCA.	Dir.
<b>TP_LCD</b>	Cond.	CCC1 0111 1111			De-Allocate LCA (CCC).	Dir.
<b>TP_RSO</b>	Uncond.	1111 1111			Reset.	None (RST)

Cond. = Conditional

A number of common operations performed with these directives are described in 27.16.3.

#### 27.10.1.6 Directive types

There are four types of directives:

- Unconditional      Always executes
- Conditional      Predicates execution of the directive dependent on prior selection of PDCs
- Selection          Establishes PDCs selected for certain subsequent Conditional Directives
- Transfer           Initiates a DCC data transfer

### 27.10.1.6.1 Unconditional Directives

An Unconditional Directive:

- Performs its intended function as an atomic unit
- Requires no directives preceding or following it to perform its intended function
- Is independent of the PCA and LCA

The Unconditional Directives are:

- TP\_NOP
- TP\_END
- TP\_DEL
- TP\_RSZ
- TP\_RSO

The TP\_NOP and TP\_END Directives perform a no-operation and end a Transport Packet, respectively. The TP\_DEL Directive stores the TP\_DELN Register. The TP\_RSZ and TP\_RSO Directives reset the TAP.7 Controller.

### 27.10.1.6.2 Reset Directives

The TP\_RSZ and TP\_RSO Directives cause a Type-3 Reset and have no other function. These directives are created in one of two ways:

- Intentionally by the DTS
- When the Test Clock is sourced by the TS and the DTS/TS connection is broken while the *TPA* state is active

These directives are encoded as 00000000b and 1111111b to detect a broken DTS/TS connection while in the *TPA* state. When the connection is broken and the TCKC signal is sourced by the TS, the TMSC signal at some point will remain a logic 1 or a logic 0 (produced by the keeper-pin behavior) for an extended period. This produces either the TP\_RSZ or the TP\_RSO Directive provided fixed-length data exchanges are being used as required when the TCKC signal is sourced by the TS.

The TP\_RSZ and TP\_RSO Directives and their generation of a Type-3 Reset are shown in Figure 28-4.

### 27.10.1.6.3 Selection Directives

The Selection Directives are:

- TP\_SSP
- TP\_SMP
- TP\_SLC

The TP\_SLC relies on the Logical Channel Address created by a previous directive(s). The TP\_SSP and TP\_SMP rely on the Physical Channel Address.

These directives create the selection criteria controlling the operation of Conditional Directives. They manage the PDCx\_SEL Register as shown in Table 27-2. They affect the operation of subsequent

directives (TP\_LCA, TP\_DCC, TP\_CRW, and TP\_CRR) by establishing the PDC(s) that are the target of these directives using the PDCx\_SEL Register.

The Selection Directives have the following behavior:

- TP\_SSP:
  - When the Physical Channel Address specified by the directive matches the Physical Channel Address of the PDC, sets the PDCx\_SEL Register value to 10b indicating:
    - The PDC is the only PDC selected.
    - The TP\_SSP Directive was the last Selection Directive executed.
    - Otherwise sets the PDCx\_SEL Register value to 00b indicating the PDC is not selected.
- TP\_SMP:
  - When either the Physical Channel Address specified by the directive matches the Physical Channel Address of the PDC or the PDC has been previously selected by either a TP\_SSP or a TP\_SMP Directive, sets the PDCx\_SEL Register value to 11b indicating:
    - The PDC is selected.
    - The TP\_SMP Directive was the last Selection Directive executed.
    - Otherwise sets the PDCx\_SEL Register value to 00b indicating the PDC is not selected.
- TP\_SLC:
  - When the Logical Channel Address specified by the directive matches the Logical Channel Address of the PDC, sets the PDCx\_SEL Register value to 01b indicating:
    - The PDC is selected.
    - The TP\_SLC Directive was the last Selection Directive executed.
    - Otherwise sets the PDCx\_SEL Register value to 00b indicating the PDC is not selected.

#### 27.10.1.6.4 Conditional Directives

A Conditional Directive:

- Performs its intended function only when conditions related to either the PCA or the LCA are met
- Relies on register values specified by previous Selection Directives to identify the target of the directive

The Conditional Directives are:

- TP\_DCC
- TP\_LCA
- TP\_CRW
- TP\_CRR
- TP\_LCD

The TP\_DCC and TP\_LCA Directives store their respective target registers, provided PDCx is selected (the PDCx\_SEL Register is nonzero). The TP\_LCD Directive de-allocates the LCA when it matches the LCA embedded in the directive.

Writing a DCC Register requires:

- The DCC be connected to the PDC when the DCC connection to the PDC includes a DCR
- Selection of the PDC using any Selection Directive ( $\text{PDCx\_SEL} \neq 00\text{b}$ )
- A register-write be performed with the TP\_CRW Directive

Reading a DCC Register requires:

- The DCC be connected to the PDC ( $\text{PDCx\_DCC} == \text{DCC number}$ ) when the DCC connection to the PDC includes a DCR
- Selection of the PDC connected to DCC with the TP\_SSP Directive
- Reading the DCC Register with the TP\_CRR Directive
- There are no intervening TP\_SMP or TP\_SLC Directives between the two directives listed above (the  $\text{PDCx\_SEL}$  Register value is 10b when the TP\_CRR Directive is processed)

### 27.10.1.6.5 Transfer Directives

A Transfer Directive:

- Initiates a data transfer
- Relies on the LCA created by previous directives

The Transfer Directives are TP\_DC0–TP\_DC7. These directives contain the LCA of the LDC that is to perform the data transfer. They initiate a data transfer, provided an LCA that has been allocated to the PDC matches the LCA within the directive. All PDCs having this LCA allocated to them may participate in the transfer. The value of the  $\text{PDCx\_LCA}$  Register determines the actions performed for these Conditional Directives as shown in Table 27-12. Note that the LCA within the directive is encoded as shown:

- $\text{LCA}[2] = \text{Directive Bit}[3]$
- $\text{LCA}[1] = \text{Directive Bit}[2]$
- $\text{LCA}[0] = \text{Directive Bit}[2] \text{ XOR Directive Bit}[1] \text{ XNOR Directive Bit}[0]$

This representation of the LCA was chosen to facilitate the use of bus-parking bits as the MSB of the directive, minimizing the length of the directive, and allowing the implementation of the TP\_RSZ and TP\_RSO Functions.

### 27.10.2 Specifications

#### Rules

- a) Each subsequent specification in 27.10.2 shall only apply to a T5 TAP.7.
- b) The directive nomenclature shown in Table 27-10 shall be used as the definition of Transport Directive fields.
- c) The Transport Directives shown in Table 27-11 shall be processed within a Transport Packet.
- d) The operation of the TP\_SSP, TP\_SMP, and TP\_SLC Directives shall be governed by Table 27-12.

**Table 27-12 — Selection Directive operation (conditional relationships)**

Conditional Directive	LCA Match == TRUE ?	PCA Match == TRUE ?	Current PDCx_SEL value	Resulting PDCx_SEL value
TP_SSP	N/A	No	Don't care	00b
		Yes		10b
TP_SMP	N/A	No	00	00b
			01	00b
			10	11b
			11	11b
		Yes	Don't care	11b
		N/A	Don't care	00b
				01b

- e) The LCA Match == TRUE expression shown in Table 27-12 shall be considered TRUE when the LCA value in the TP\_LCA Directive (CCC) plus eight is equal to the value of the PDCx\_LCA Register.
- f) The PCA Match == TRUE expression shown in Table 27-12 shall be considered TRUE for the TP\_SSP or TP\_SMP Directives shown in Table 27-11 provided all of the following are true:
  - 1) The CIDI Register value is a logic 0 (the Controller ID is valid).
  - 2) The IIII value in the directive is equal to the Controller ID Register value.
  - 3) A PDC is implemented.
  - 4) Any of the following are true:
    - i) Y in the directive is a logic 0 and the PDC number is zero (PDC0).
    - ii) Y in the directive is a logic 1 and the PDC number is one (PDC1).
- g) The operation of the TP\_LCA, TP\_DCC, TP\_CRW, and TP\_CRR Directives shall be governed by Table 27-13.

**Table 27-13 — Register Transfer Directive operation (conditional relationships)**

Conditional Directive	Condition	PDCx_LCA ==	PDCx_SEL Register value	Result
TP_LCA	The PDC is not selected	0xxxx	00b	No change in PDCx_LCA value
	The PDC is not selected but the LCA specified by the command is allocated to the PDC	1xCCC	00b	PDCx_LCA = 0xxxx
	The PDC is the only PDC selected	xxxxx	10b	PDCx_LCA = 10CCC
	The PDC is selected and there may be multiple PDCs selected	xxxxx	1xb	PDCx_LCA = 11CCC
TP_DCC	The PDC is not selected		00b	No operation
	The PDC is selected		01b, 10b, 11b	PDCx_DCC = DDDD
TP_CRW	The PDC is not selected		00b	No operation
	The PDC is selected		01b, 10b, 11b	DCC Register RRR is written with a Register Element 4 × (LLL+1) bits in length
TP_CRR	The PDC is not selected or last selected with a TP_SMP or TP_SLC Directive		00b, 01b, 11b	No operation
	The PDC is selected with TP_SSP Directive with no subsequent TP_SMP or TP_SLC Directive		10b	DCC Register RRR is read with a Register Element 4 × (LLL +1) bits in length

h) The operation of the TP\_DC0 through TP\_DC7 Directives shall be governed by Table 27-14.

**Table 27-14 — Data Transfer Directive operation (conditional relationships)**

Conditional Directive	PDCx_LCA value:	Resulting operation
TP_DC0 - TP_DC7	0x111b – 0x000b	No operation—Data Element is not transferred with the PDC
TP_DC0	1x000b	Data transfer—Data Element transferred with the PDC
	1x001b – 1x111b	No operation—Data Element is not transferred with the PDC
TP_DC1	1x001b	Data transfer—Data Element transferred with the PDC
	1x000b; 1x010b – 1x111b	No operation—Data Element is not transferred with the PDC
TP_DC2	1010b	Data transfer—Data Element transferred with the PDC
	1x000 – 1x001b; 1x011b – 1x111b	No operation—Data Element is not transferred with the PDC
TP_DC3	1011b	Data transfer—Data Element transferred with the PDC
	1x000 – 1x010b; 1x100b – 1x111b	No operation—Data Element is not transferred with the PDC
TP_DC4	1x100b	Data transfer—Data Element transferred with the PDC
	1x000 – 1x101b; 1x101b – 1x111b	No operation—Data Element is not transferred with the PDC
TP_DC5	1101b	Data transfer—Data Element transferred with the PDC
	1x000 – 1x100b; 1x110b – 1x111b	No operation—Data Element is not transferred with the PDC
TP_DC6	1x110b	Data transfer—Data Element transferred with the PDC
	1x000 – 1x101b; 1x111b	No operation—Data Element is not transferred with the PDC
TP_DC7	1x111b	Data transfer—Data Element transferred with the PDC
	1x000 – 1x110b	No operation—Data Element is not transferred with the PDC

- i) The processing of the TP\_RSZ and TP\_RSO Directives shall generate a Type-3 Reset for one TCKC signal period beginning with the falling edge of the TCKC signal following the MSB of this directive (see Figure 28-4).
- j) The processing of a TP\_NOP Directive shall generate a no-operation.
- k) The TP\_END Directive shall terminate a TP with an SP beginning with the falling edge of the TCKC signal following the MSB of this directive.
- l) The TP\_LCD Directive shall de-allocate the LCA specified by the directive (set the PDCx\_LCA[4] to a logic 0), provided the CCC field plus eight equals the PDCx\_LCA Register value.

## 27.11 Register Elements

### 27.11.1 Description

#### 27.11.1.1 Overview

Register Elements follow the TP\_CRW and TP\_CRR Directives. The Register Element following a TP\_CRW Directive transfers control register-write data from the DTS to a DCC Control Register. The Register Element following a TP\_CRR Directive is DCC Control Register read data.

#### 27.11.1.2 Register Element length

The length of the Register Elements is specified by the TP\_CRW and the TP\_CRR Directive Elements that precede them. This length is  $(n + 1) \times 4$  bits where  $0 \leq n \leq 7$ .

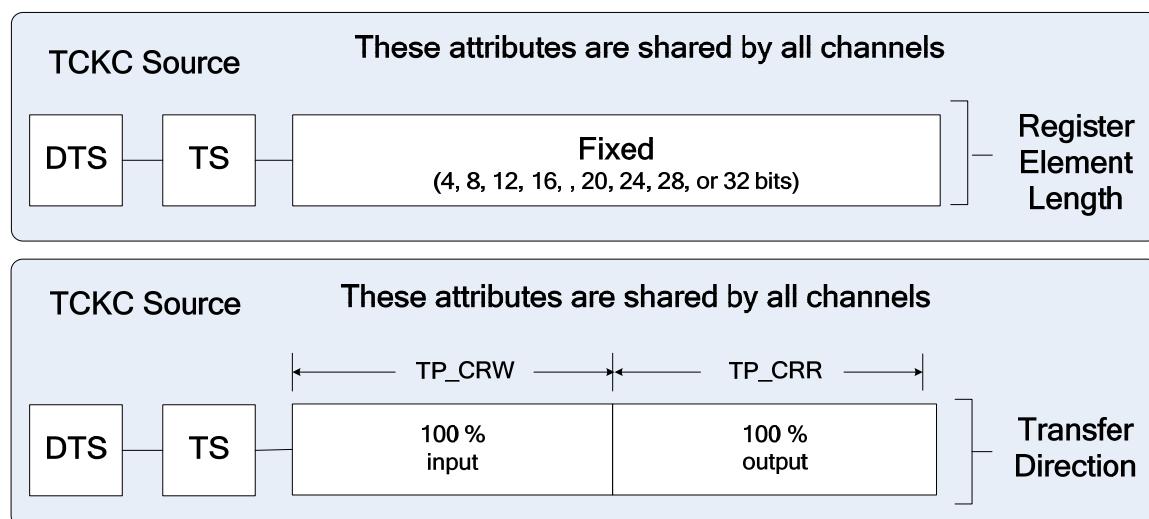
#### 27.11.1.3 Transfer direction

When a DCC is connected to a PDC, a Register Element following a TP\_CRW Directive moves DCC register-write information from the DTS to the DCC. The Register Element information following a TP\_CRW Directive is utilized by all DCCs that are connected to a selected PDC (the register-write is broadcast).

The Register Element information following a TP\_CRR Directive is sourced by the DCC last selected with a TP\_SSP Directive. PDCs are not permitted to source Register Element data when the last selection was made with either the TP\_SMP or TP\_SLC Directive. The TMSC pin remains at a high-impedance level during a Register Element following a TP\_CRR Directive when the PDC does not meet this criterion. This ensures that there are no drive conflicts with the use of the TP\_CRR Directive.

#### 27.11.1.4 Summary of Register Element characteristics

The Register Element characteristics are summarized in Figure 27-8.



**Figure 27-8 — Summary of Register Element characteristics**

## 27.11.2 Specifications

### Rules

- a) Each subsequent specification in 27.11.2 shall only apply to a T5 TAP.7.
- b) The length of a Register Element following the TP\_CRR or TP\_CRW Directives shown in Table 27-11 shall be  $4 \times ((\text{LLL}) + 1)$  bits (i.e., 4, 8, 12, 16, 20, 24, 28, or 32 bits).

## 27.12 Data Elements

### 27.12.1 Description

#### 27.12.1.1 Overview

A Data Element follows a TP\_DCx Directive. It transfers data between DCCs connected to PDCs that are allocated an LCA equal to CCC (see Table 27-10), or the DTS and a DCC connected to a PDC that is allocated an LCA equal to CCC. Data Elements may be shared by multiple DCCs, provided a higher level protocol manages their use. The Data Element should be viewed as providing a time slot where data may be transferred subject to a format or protocol used by the DCC. A Data Element has two key attributes, the transfer length and the transfer direction.

#### 27.12.1.2 Data Element length

A Data Element may be of either fixed or variable length. Its length is managed with the TP\_DELN Register. Fixed-length Data Elements may be used with either a DTS- or a TS-sourced TCKC signal. Variable-length Data Elements are more efficient but cannot be used with a TS-sourced TCKC signal as they are terminated with an EOT Escape.

A fixed-length Data Element:

- Transfers 8, 16, 32, or 64 data bits
- Terminates automatically after the required number of data bits is transferred

A variable-length Data Element:

- Transfers  $(n + 1) \times 4$  bits, where n is an integer  $\geq 1$
- Terminates after nibble n + 1 when an EOT Escape occurs in any bit of nibble n

The DTS software chooses the Data Element Length based on the Test Clock source, the latency of returning the control of the TMSC signal to scan operations, and the DTS capabilities.

#### 27.12.1.3 Transfer direction

The transfer direction of a Data Element is defined by the DCC\_DIRA field of the PDCx\_DCC\_CR1 Register of the DCC connected to the PDC participating in the transfer.

This register may define the transfer as one of the following:

- Input only All data is transferred from the TAP to the DCC
- Output only All data is transferred from the DCC to the TAP
- Input/output Bidirectional data exchange with 50% of the data transferred in each direction
- Custom Direction of each bit of the data transferred is determined by the DCC

Input-only, input/output, and output-only transfers provide fixed allocation of a Data Element's bandwidth between input and output. The DCC(s) determines the input and output bandwidth with custom transfers.

With multi-client and client-to-client transfers, the participating DCCs are responsible for drive-conflict prevention. As stated previously, this is generally handled using a higher level protocol to manage the TMSC signal drive with the custom transfer direction as described previously. The operation of this drive-conflict protection is outside the scope of this standard.

The DTS may determine whether the DCC supports more than one Data Element Transfer Direction by writing the DCCy\_DIRR field of the PDCx\_DCC\_CR1 Register and then reading the value of this register field. This value is static unless it may be programmed by the DTS. When the read value matches the written value, the programmable Data Element Transfer Direction is supported. It is not supported when the written value does not match the read value.

An application using the data channel is called a:

- BDX              When the input/output bandwidth allocation is fixed. This implies a single client for input and bidirectional transfers to prevent drive conflicts.
- CDX              When the input/output bandwidth allocation and direction of the transfer is managed in a custom manner within Data Elements. Since custom transfers imply a custom protocol comprehended by all clients, these transfers may be single client, multi-client, or client-to-client.

#### 27.12.1.4 Utilization and generation of data

A DCC's utilization of input data and generation of output data is affected by the:

- LCA specified by the TP\_DCx Directive
- Connection of the DCC to a PDC
- Use of a higher level protocol

When a DCC is connected to a PDC sharing the Logical Channel Address specified by the TP\_DCx Directive:

- Input data is:
  - Sourced to the DCC via the TMSC signal by the DTS or a DCC connected to another PDC
  - Utilized by the DCC based on a higher level protocol or other factors
- Output data is:
  - Sourced by the DCC via the TMSC signal based on a higher level protocol or other factors

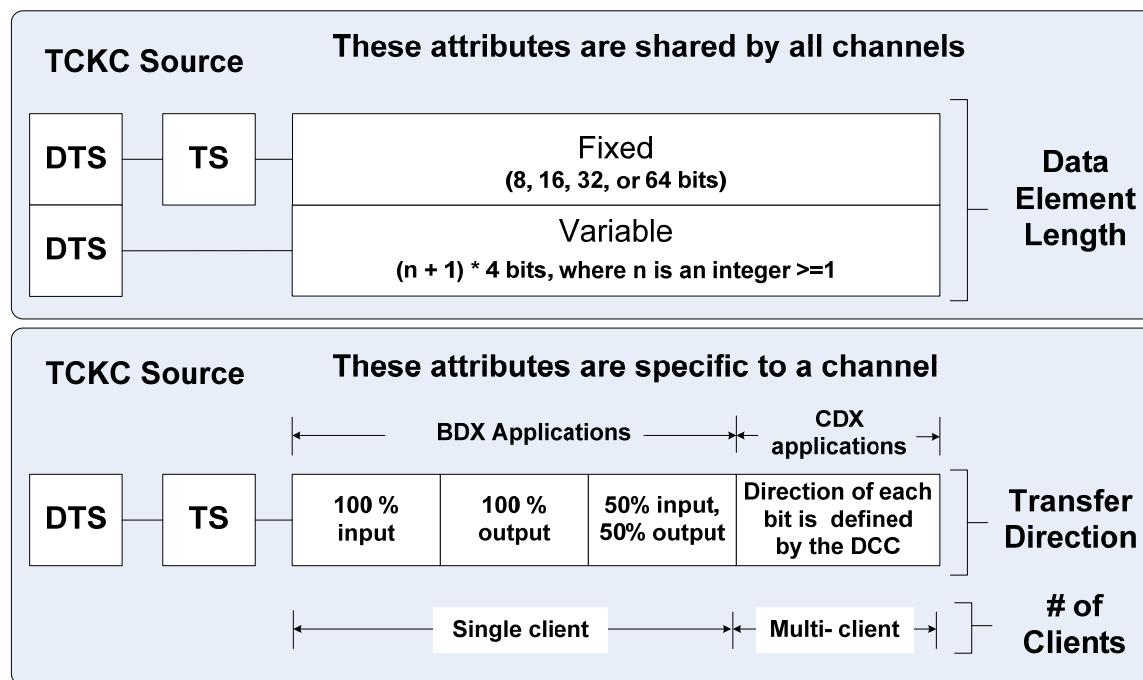
#### 27.12.1.5 Operation with single and multiple clients

The custom transfer direction shown in Table 28-3 is used with both single- and multi-client operation. With single-client operation, there are no drive conflicts within a Data Element when the DTS and client coordinate their use of the TMSC signal, as each utilizes the same high-level protocol.

With multi-client operation, input data within the Data Element is passed to all DCCs that are connected to all PDCs with the same allocated LCA. The DCC(s) use the high-level protocol to determine whether they are the destination of the input data or whether they should utilize the data in any manner. Output data within the Data Element can only be sourced by one DCC during a bit period.

### 27.12.1.6 Summary of Data Element characteristics

The Data Element characteristics are summarized in Figure 27-9.



**Figure 27-9 — Summary of Data Element characteristics**

### 27.12.2 Specifications

#### Rules

- a) Each subsequent specification in 27.12.2 shall only apply to a T5 TAP.7.
- b) The transfer length of a TP Data Element shall be defined by the TP\_DELN Register value as specified in Table 27-2.
- c) A TP\_DCx Directive specifying the PDC's LCA initiates a data exchange with the DCC.
- d) The relationships of a DCC and a Data Element shall be governed by Table 27-15.

**Table 27-15 — DCC/Data Element relationships**

PCA aliased to LCA	DCC connected to PDC	Transfer direction	Data Element	
			Input bits	Output bits
No	x	Any	Not utilized	Not generated
Yes	No			
Yes	Yes	Input only	Forwarded to the selected DCC	N/A
Yes	Yes	Output only	N/A	Transferred to DTS
Yes	Yes	Input/output	Forwarded to the selected DCC	Transferred to DTS
Yes	Yes	Custom	Forwarded to selected DCC, the DCC controls their utilization	The selected DCC populates the Data Element when the Custom Direction indicates the generation of an output bit

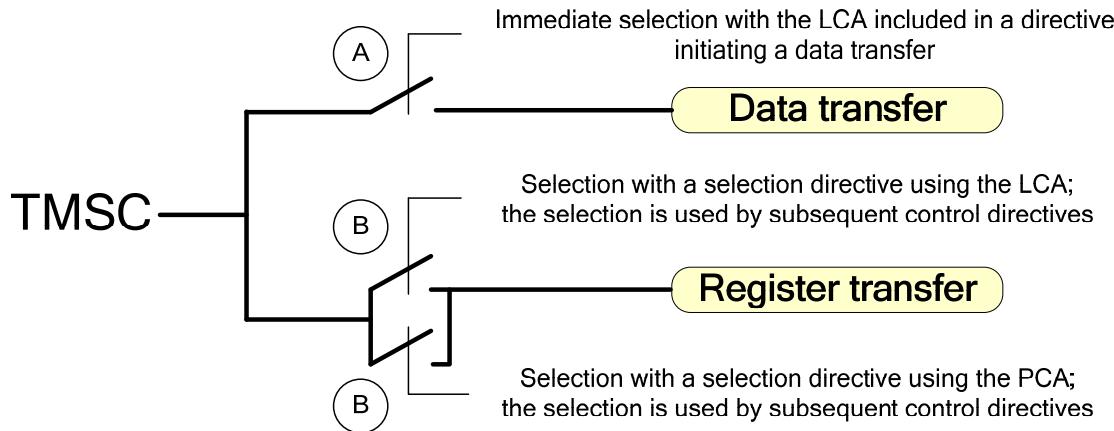
- e) When the Data Element Transfer Direction value is fixed, the DCCy\_DIRA field of the PDCx\_DCCy\_CR1 Register shall be read as specified in Table 27-4.
- f) When the Data Element Transfer Direction value is programmable, the DCCy\_DIRA field of the PDCx\_DCCy\_CR1 Register shall be read as:
  - 1) The value last written to the DCCy\_DIRR field of the PDCx\_DCCy\_CR1 Register, provided this value specifies a supported Element Transfer Direction.
  - 2) The inverted value of the last value written to the DCC\_DIRR field of the PDCx\_DCCy\_CR1 Register, provided the last value written to this register does not specify a supported Element Transfer Direction.
- g) When the DCCy\_DIRR field of the PDCx\_DCCy\_CR1 Register is programmable, the only means of programming this value shall be using the TP\_CRW Directive.

#### Permissions

- h) A DCC's utilization or generation of Data Element data may be inhibited to facilitate multi-client or client-to-client communication.

### 27.13 Selection of control and data targets

The targets of control and data transfers are identified using different selection mechanisms. Directives initiating a data transfer identify the Logical Channel Address used for a data transfer as part of the directive. Directives initiating a control transfer rely on a prior TP\_SSP, TP\_SMP, and TP\_SLC to create the PDCx\_SEL value identifying the target for a transfer. These selection mechanisms operate differently and do not interact. The two forms of selection (A and B) are illustrated in Figure 27-10.



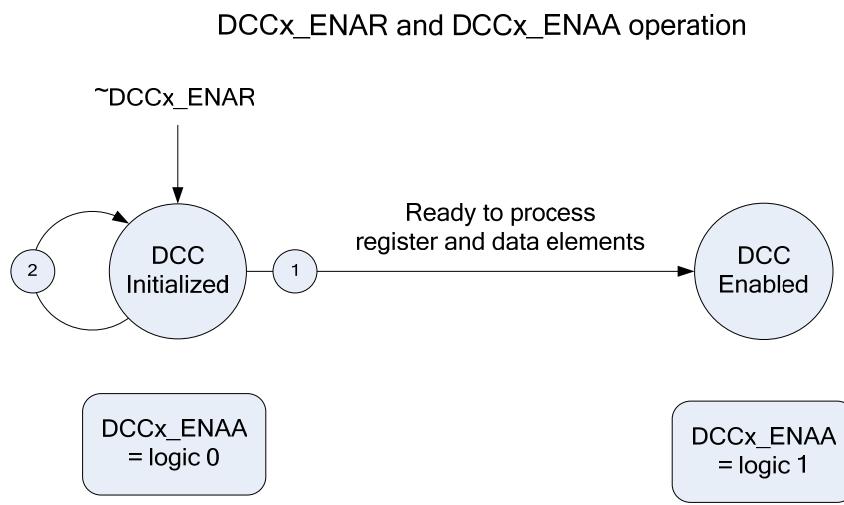
**Figure 27-10 — Selection mechanisms for data and register transfers**

## 27.14 Data Channel Client functions

### 27.14.1 Description

#### 27.14.1.1 Initializing a Data Channel Client

The optional DCCy\_ENAR register bit may be used to initialize a DCC requiring this function. A DCCy\_ENAR Register value of logic 0 initializes the DCC, while a value of logic 1 allows the operation of the DCC. The DCCy\_ENAR register bit does not reset the DCC interface as a TAP.7 Controller reset performs this function. The DCCy\_ENAR register bit provides the DTS a means to confirm that the DCC is reset or enabled. When this register bit is read as a logic 0, the DCC is being initialized. When this register bit is read as a logic 1, the DCC is ready for data transfers. The behavior of the DCC Enable Function is shown in Figure 27-11.



**Figure 27-11 — Operation of the DCCy\_ENAR and DCCx\_ENAA register bits**

#### 27.14.1.2 Orderly shutdown of a transfer

The characteristics of a DCC or the data being transferred by the DCC (attributes of a higher level protocol or other data characteristics) may require the synchronization of the deselection of the DCC with DCC

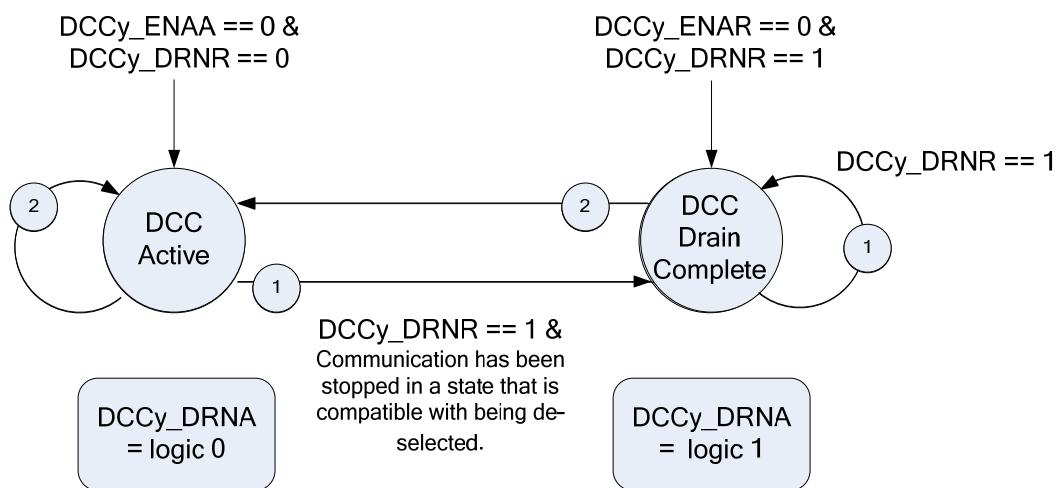
transfers. This synchronization ensures that DCC messages are completed before the DCC is deselected. The optional Drain Request (DCCy\_DRNR) and Drain Acknowledge (DCCy\_DRNA) register bits may be used for this purpose. The DTS may use these facilities to manage the shutdown of a transfer in an orderly manner. When the DCCy\_DRNR and DCCy\_DRNA register bits are implemented, the use of these registers is expected to conform to the following description.

Two cases where orderly shutdown may be desired are the impending disconnection via the PDCx\_DCC Register or the impending de-allocation of the PDC's LCA. The DTS notifies the DCC that it should prepare for shutdown using the DCCy\_DRNR register bit. The DCC indicates that it has reached a state that is compatible with being deselected (i.e., deselection will cause no adverse consequences) using the DCCy\_DRNA register bit.

The DTS notifies the DCC that it may transfer data within Data Elements at its discretion with a logic 0 DCCy\_DRNR Register value. The DCC immediately responds to a logic 0 DCCy\_DRNR register bit value with a logic 0 DCCy\_DRNA register bit value. Subsequently, the DTS notifies the DCC of its intention to deselect the DCC with a logic 1 DCCy\_DRNR register bit value. The DCC indicates that its deselection may have adverse consequences with a logic 0 DCCy\_DRNA register bit value, while the DCCy\_DRNR register bit value is a logic 1.

The DCC responds to the drain request by stopping data transfers within its Data Elements (even though Data Elements may continue flowing to this DCC) at a point where deselection may occur with no adverse consequences. When its deselection will no longer have adverse consequences, the DCC notifies the DTS with a logic 1 DCCy\_DRNA register bit value that it has reached a state where disconnecting the DCC will have no adverse consequences.

The behavior of the DCC Drain Function is shown in Figure 27-12.



**Figure 27-12 — Operation of the DCCy\_DCNR and DCCy\_DRNA register bits**

#### 27.14.1.3 Effects of Online/Offline operation on the Transport Function

If transport is being used when the TAP.7 Controller is placed Offline, the application in the TS using the PDC will see its activity suspended. The PDC activity remains suspended while the operating state is *CPA* (until the TAP.7 Controller is placed Online). Since Offline TAP.7 Controllers use the Control Protocol, they are disregarded for the remainder of the transport description.

## 27.14.2 Specifications

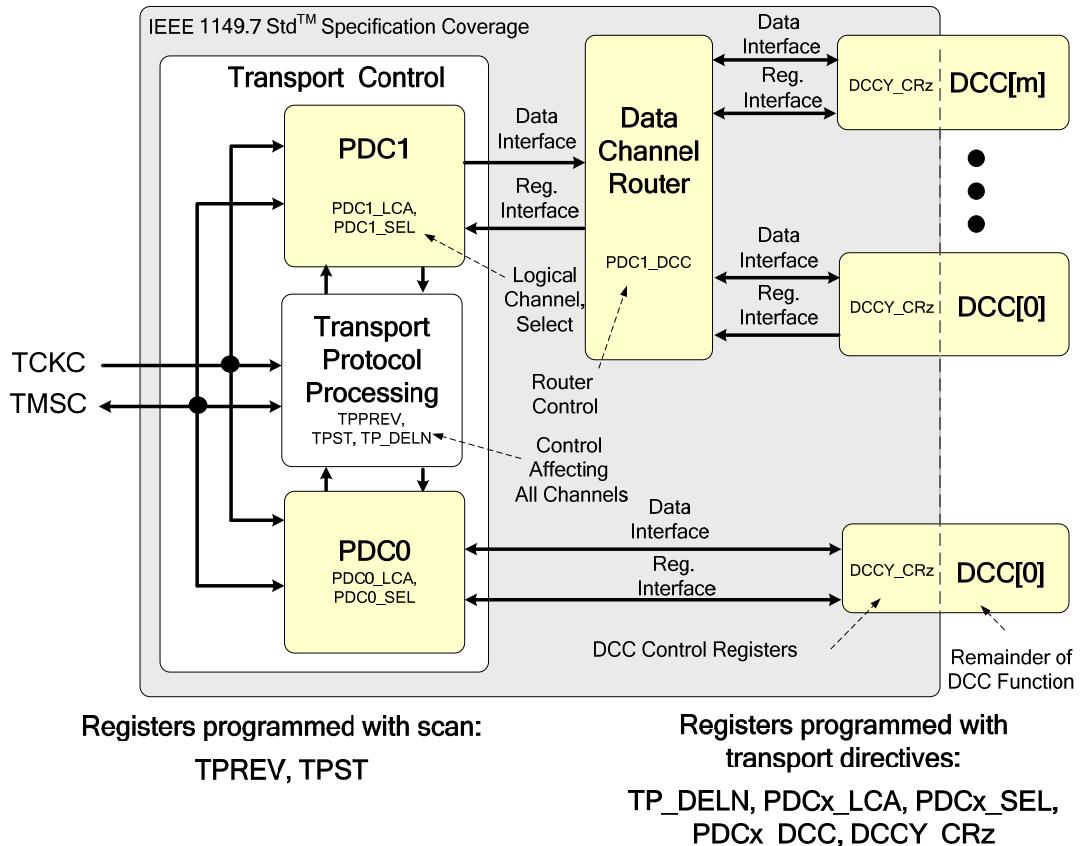
### Rules

- a) Each subsequent specification in 27.14.2 shall only apply to a T5 TAP.7.
- b) A PDCx\_DCC Register shall be implemented, provided there is a need to connect more than one DCC to a PDC.
- c) When implemented, the DCCs shall be sequentially numbered beginning at 0000b and be referenced by this number with the PDCx\_DCC Register.
- d) A Type-0–Type-4 Reset shall initialize all of the following:
  - 1) The PDCx/DCC interfaces of the PDC, DCR, and DCC.
  - 2) DCC Control Registers managed by the DTS (CR0–CR7).
- e) A DCC shall convey the transfer direction to the DTS with the DCCy\_DIRA Register field of Control Register One as specified in Table 27-3.
- f) A logic 0 DCCy\_ENAR register bit value shall initialize the DCC connected to the PDC, subject to Rule 27.14.2 g).
- g) The DCCy\_ENAR register bit value shall not reset the PDCx/DCC interfaces of the PDC, DCR, and DCC.
- h) The operation of the DCCy\_ENAR and DCCy\_ENAA register bits shall be governed by Figure 27-11.
- i) The DCC shall interpret a DCCy\_DRNR register a logic 1 value as a request to stop communication in a state that is compatible with being deselected.
- j) The operation of the DCCy\_DRNR and DCCy\_DRNA register bits shall be governed by Figure 27-12.

## 27.15 Partitioning of the Transport Control Function

### 27.15.1 Building blocks

A high-level view of the Transport Control Function is shown in Figure 27-13, providing a more detailed view of the function shown in Figure 27-1. The large rectangle that is shaded gray in the center of this figure represents the transport capability defined by this standard. The functions that are outside this rectangle are Chip-Level Logic with user-specified behavior. Commands define the states supporting transport and the Transport Protocol being used. Transport Directives control the remainder of the operation of this function. The operation of the chip-level portion of the Data Channel Client, other than its Physical Data Channel interface and the set of registers controlled by this interface, is outside the scope of this standard.



**Figure 27-13 — Conceptual view of T5 TAP.7 transport control**

The acronyms used for the Transport Control Function building blocks shown in Figure 27-13 are described briefly as follows:

— TPP:	Transport Protocol Processing	Maintains synchronization with the protocol stream
— PDCx:	Physical Data Channel x	Provides DCCs access to the TMSC TAP signal
— DCR:	Data Channel Router	Provides for the use of the PDC by multiple DCCs
— DCC:	Data Channel Client	Provides the source/destination of data information

This figure shows the association of the Transport Registers listed in Table 27-2, Table 27-3, and Table 27-4 with the Transport Control Function building blocks listed above.

The Transport Protocol Processing (TPP) block is always present, while there can be zero, one, or two Physical Data Channels as described previously. It contains the registers and state machines. The TPP block supervises the activity of Physical Data Channel 0 (PDC0) and Physical Data Channel 1 (PDC1). It ensures that the APU remains synchronized with the Advanced Protocol bit sequences generated with Transport Packets.

A Physical Data Channel is a bridge between the TMSC TAP signal and the Data Channel Client connected to the Physical Data Channel, moving data between a Data Channel Client and the TMSC signal or vice versa. It contains registers defining the Logical Channel Address and selection for accessing Physical Data Channel specific controller registers. It provides two DCC interfaces, the first to read and write DTS-managed control registers in the Data Channel Client, and the second to perform data exchanges with the Data Channel Client.

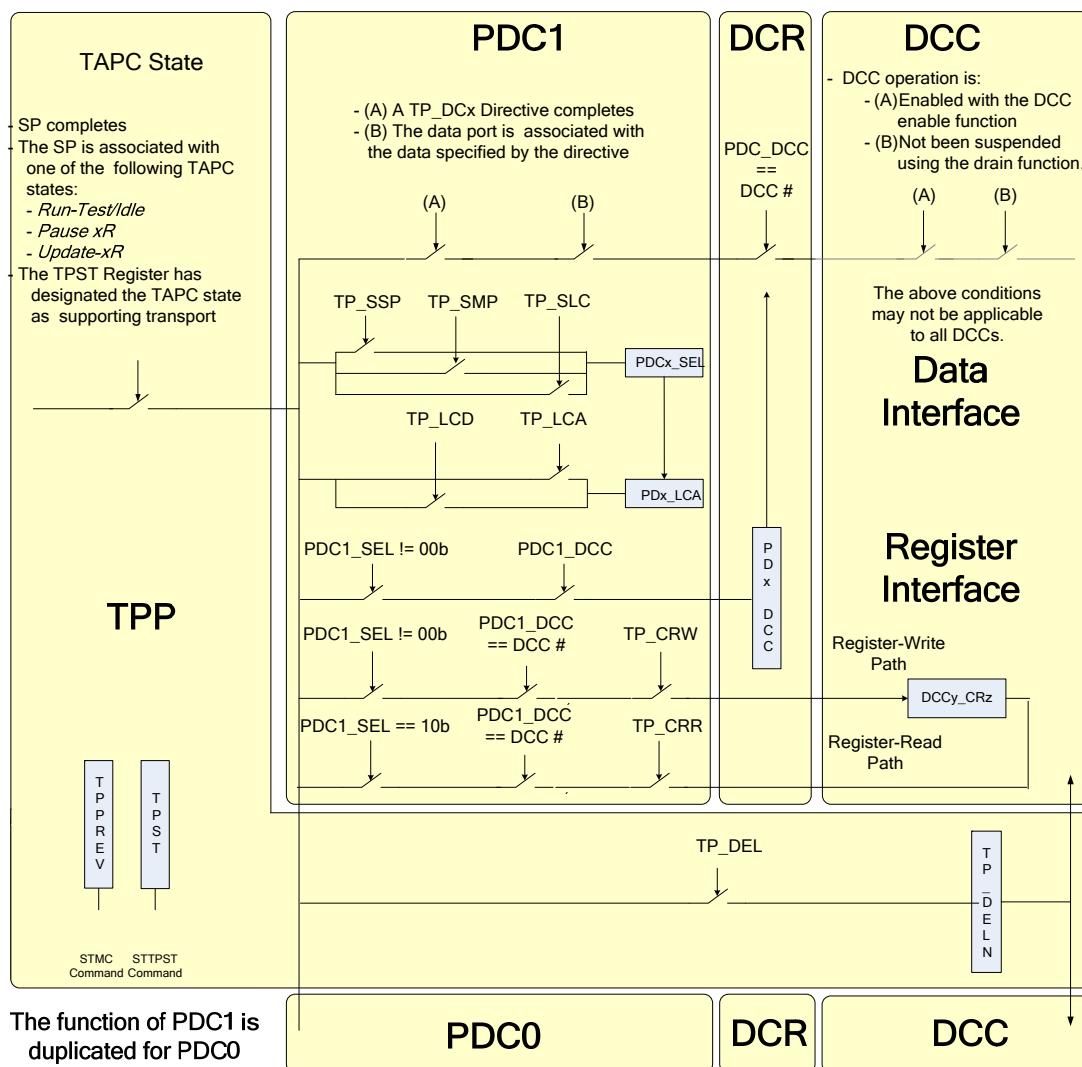
A Data Channel Router is added to allow the connection of more than one Data Channel Client to a Physical Data Channel. The PDCx\_DCC Register within the Data Channel Router selects the Data Channel Client connected to the Physical Data Channel. This register is controlled from the Physical Data Channel.

A Data Channel Client is formed with a register interface and registers defined by this standard along with a Chip-Level section not specified by this standard.

### 27.15.2 Directive/register/operational relationships

A conceptual view of the Transport Registers and their relationship to the operation of the Transport Control Function is shown in Figure 27-14. This figure shows the registers associated with the Transport Control Function and the conditions necessary to activate the Data Channel Client's Data and Register interfaces. Within this figure, directives and register values open and close the switches in this figure. All switches in series are closed to perform an action. Parallel paths indicate multiple ways of performing an action.

The switches shown below close when all of the conditions listed above the switch are satisfied.



**Figure 27-14 — Conceptual view of transport control facilitating a data exchange**

## 27.16 Programming considerations

### 27.16.1 Managing transport with the DTS

The DTS manages the Transport Function. It is considered the transport initiator with the TAP.7 Controller and the DCC(s) connected to the TAP.7 Controller being its responders. The DTS controls the transport attributes listed below. It defines the:

- TAPC states supporting transport
- Transport protocol used
- Transport topology:
  - Connects the DCC of interest to a PDC when a DCR is used
  - Associates PDCs with LDCs
- Allocates the Advanced Protocol bandwidth to:
  - Scan
  - Transport:
    - Data exchanges with each of the eight LDCs
    - Client control operations
    - Controller control operations

The DTS is responsible for establishing the Data Element Length, LDCs, and DCCs connected to the PDCs before enabling the use of transport with the TPST Register.

### 27.16.2 Single and multi-client data exchanges

The DTS should confirm the transfer direction and compatibility of the protocols to be used when creating an LDC capable of multi-client or client-to-client operation. The DTS should ensure single-client operation of an LDC when a DCC indicates it desires input/output and output-only data exchanges. The DTS may utilize an LDC for multi-client data exchanges with input-only transfers, or both multi-client or client-to-client data exchanges when a DCC indicates it will operate with custom direction control and the DTS determines the clients sharing the LDC will utilize a common higher level protocol.

### 27.16.3 Bandwidth allocation

The DTS may allocate the transport bandwidth to LDCs in any manner, including allocating all of the transport bandwidth to a single LDC for a prolonged period of time. In the extreme case, the DTS chooses to interrupt the transport operation only when there is a need to use the TAP for a scan operation.

### 27.16.4 Common and uncommon operations performed with directives

A number of common operations are performed with the directive sequences described as follows:

- Allocate an LCA to a single PDC:
  - TP\_SSP              Select one PDC
  - TP\_LCA              Allocate an LCA to this PDC
- Allocate an LCA to two or more PDCs:
  - TP\_SSP              Select PDC and clear prior PDC selections
  - TP\_SMP              Add another PDC selection

- TP\_LCA Allocate an LCA to selected PDCs
- Change the LCA of a group of PDCs already forming an LDC (uncommon):
  - TP\_SLC Select the LDC PDCs, deselect other PDCs
  - TP\_LCA Allocate an LCA to the selected PDCs
- Change the Data Element Length:
  - TP\_DEL Store the Data Element Length
- Write a DCC Register:
  - TP\_SSP Select the PDC
  - TP\_DCC Select the DCC Register
  - TP\_CRW Begin register-write, follow with Register Element
  - Register Element Data written to register
- Simultaneously select the same numbered DCC of all PDCs of an LDC:
  - TP\_SLC Select PDC, clear other selections
  - TP\_CRW Initiate register-write
  - Register Element Data written to register
- Read a DCC Register:
  - TP\_SSP Select PDC, clear other selections
  - TP\_DCC Select DCC (if necessary)
  - TP\_CRR Initiate register-read
  - Register Element Data read from DCC Register
- Initiate a data transfer:
  - TP\_DCx Initiate a data transfer
  - Data Element Data transfer

### 27.16.5 Dynamic source/destination changes

The lowest overhead for communication with two DCCs is created by using two LDCs, one for each DCC. In this case, the TP\_DCx Directive switches between DCCs.

The DTS can dynamically change the source/destination of data exchanges with DCCs connected to a PDC with the TP\_DCC Directive between data transfer, provided the PDC\_SEL Register indicates the PDC is selected. Switching between DCCs connected to the same PDC requires two directives, the first, a TP\_DCC Directive, to change the DCC connection and the second, a TP\_DCx Directive, to initiate a data transfer. In some cases, it may be desirable to provide for the connection of DCCs to both of the TAP.7 Controller PDCs so that they may be accessed using two separate data channels. The PDCx\_DCC Register value selecting them for each PDC need not be the same, in this case.

De-allocating an LCA merely suspends communication with clients that have been accessed previously with this LCA. Reallocating an LCA to a PDC permits the resumption of communication of a DCC connected to the PDC.

### 27.16.6 Transfer alignment characteristics

Alignment of the data boundaries (e.g., byte, word, double-word, etc.) and Data Element boundaries is not required and is generally not practical when data is transferred on demand (when it becomes available) or a higher level protocol is used. An atomic unit of data may span a number of Data Elements or be smaller than a Data Element. The TAP.7 Controller treats all information transferred within a Data Element as data even when it is control information used to manage data transfers (e.g., a higher level protocol).

An LCD operates in a manner that accommodates existing communications peripherals. These peripherals may be attached to a Data Channel when the pace of input and output of these peripherals can be managed to make them compatible with the occurrence and characteristics of Data Elements. In some cases, it may be advantageous to decouple the transmitter and the receiver sections of a DCC to allow a pipelined operation with the TAP.7 Controller.

### 27.17 Aspects of transport not covered by this specification

The following aspects of transport operation are not covered by this specification:

- Data format(s) used by a higher level protocol
- Flow control used by a higher level protocol
- The TMSC signal drive-conflict management required for multi-client and client-to-client transfers
- Definition of customizable control bits

These subject areas are expected to be governed by the specifications of DCCs and higher level protocols utilizing the transport facilities with its information made available through chip-level functional specifications.

## 28. Transport operation and interfaces

### 28.1 Introduction

This clause describes the processing of Transport Packets, and the operation of the TAP related to transport and the APU's DCC interface(s).

The subjects described in this clause are covered in the following order:

- 28.2 TAP interface
- 28.3 Transport State Machine
- 28.4 PDCx/DCC interface
- 28.5 Five-bit directives
- 28.6 Eight-bit directives
- 28.7 12-bit directives
- 28.8 DCC interface operation
- 28.9 An approach to implementing the Transport Function

### 28.2 TAP interface

#### 28.2.1 TPA state flow

The entry into and the exit from the *TPA* State are shown in Figure 28-1. A single TP is processed by the *TPA* state. The operation of the *TPA* state is shown at a high level in Figure 28-2.

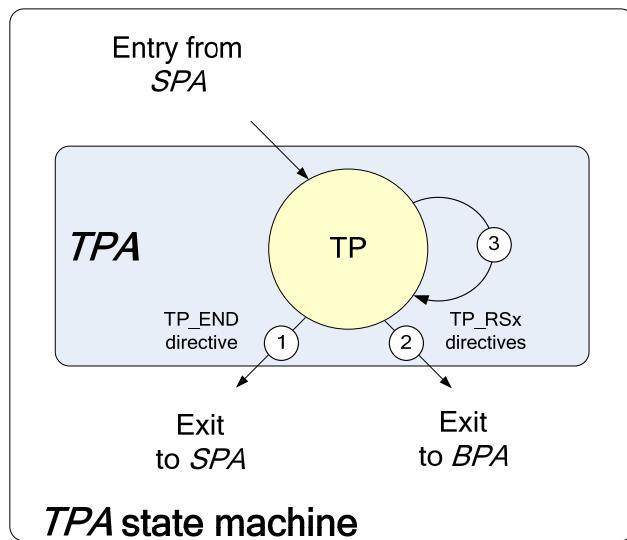


Figure 28-1 — *TPA* state behaviors

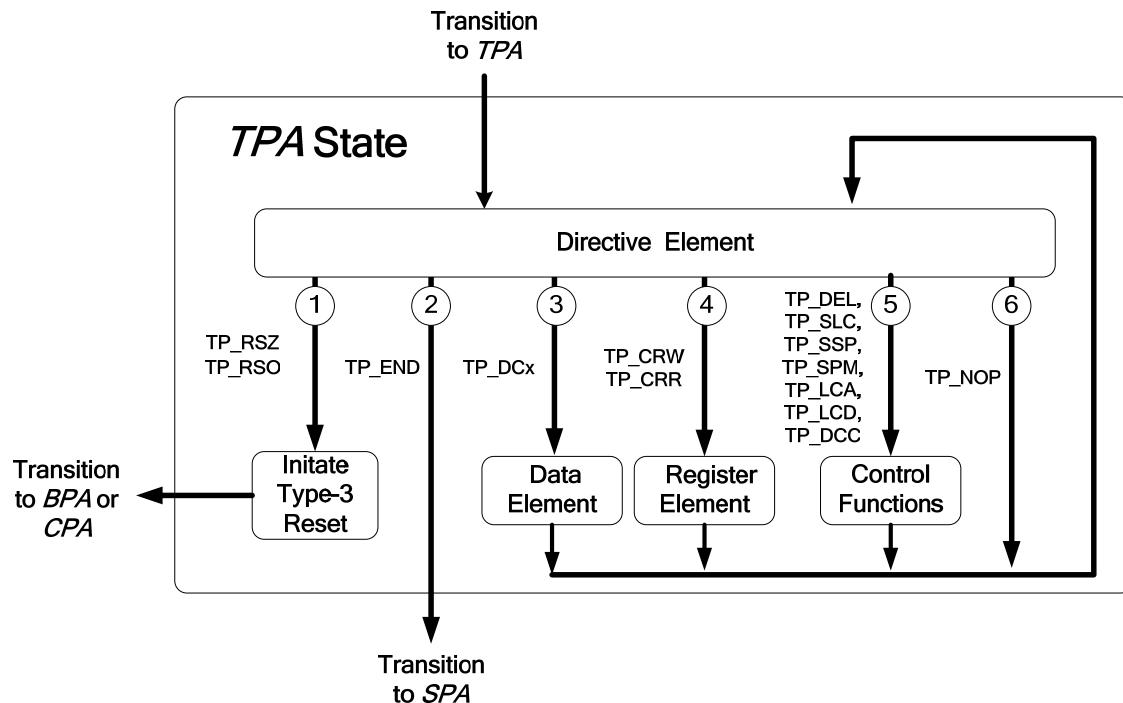


Figure 28-2 — TPA state flow

### 28.2.2 Directive Element characteristics

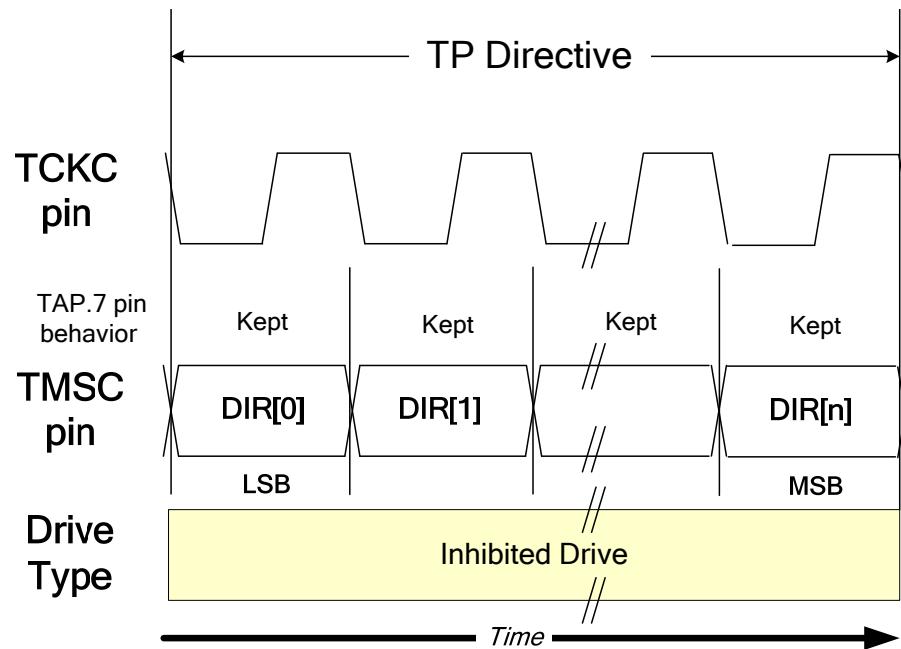
#### 28.2.2.1 Description

TP Directive Elements are five, eight, or twelve bits in length as shown in Table 27-11. They are transmitted LSB first with the format shown in Figure 28-3. The timing of the Type-3 Reset generated by the TP\_RSZ and TP\_RSO Directives is shown in Figure 28-4.

#### 28.2.2.2 Specifications

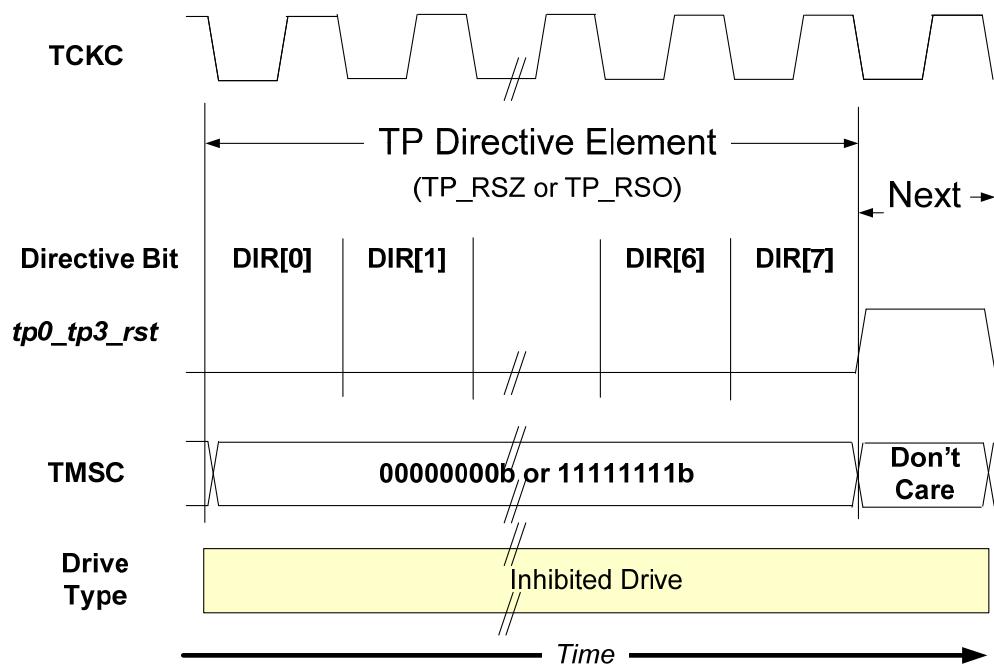
##### Rules

- a) Each subsequent specification in 28.2.2.2 shall only apply to a T5 TAP.7.
- b) The format, timing, and TMSC Signal Drive Policy utilized with TP Directive Elements shall be governed by Figure 28-3.



**Figure 28-3 — TP Directive Element format, timing, and TMSC signal drive characteristics**

- c) The timing of the Type-3 Reset generated by the TP\_RSZ and TP\_RSO Directives shall be governed by Figure 28-4.



**Figure 28-4 — A Type-3 Reset generated by the TP\_RSZ and TP\_RSO Directives**

### 28.2.3 Register Element characteristics

#### 28.2.3.1 Description

##### 28.2.3.1.1 Format, timing, and TMSC signal drive characteristics

TP Register Elements are transmitted LSB first. Their format, timing, and TMSC signal drive characteristics are shown in Figure 28-5.

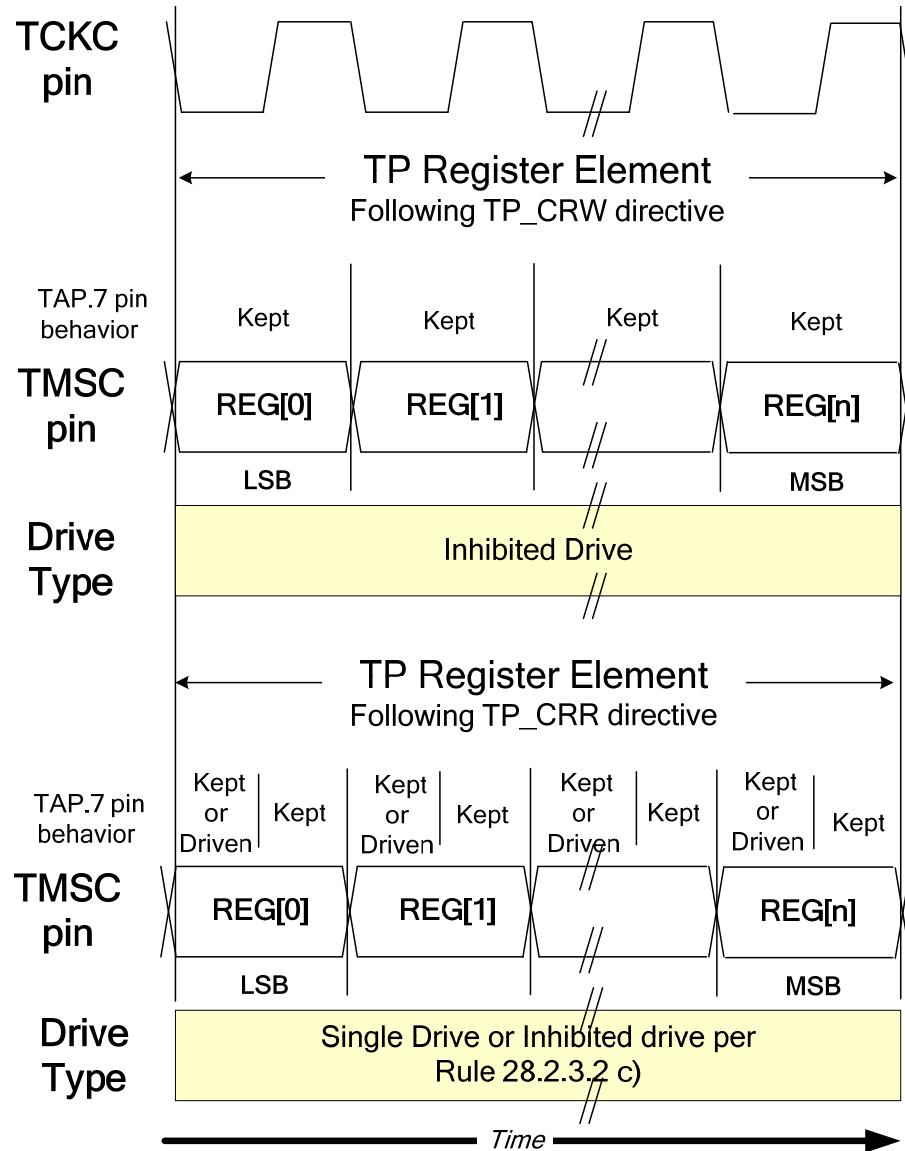
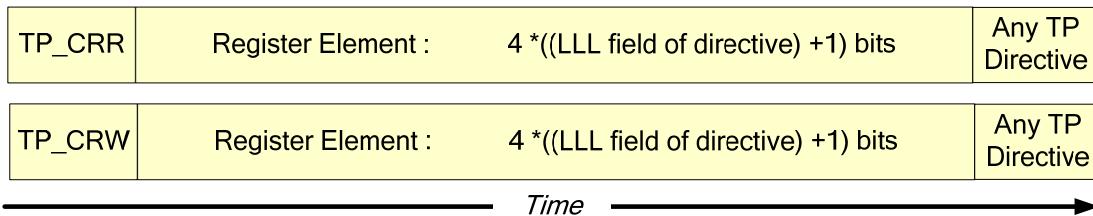


Figure 28-5 — TP Register Element format, timing, and TMSC signal drive characteristics

#### 28.2.3.1.2 Register Element length

The LLL field within the TP\_CRR and TP\_CRW Directive defines the length of the Register Element as shown in Figure 28-6. A TP Directive always follows the last bit of a Register Element.



**Figure 28-6 — Register Element with surrounding TP Directives**

#### 28.2.3.1.3 Register Element content

The content of a Register Element following the:

- TP\_CRW Directive is created by the DTS and utilized by one or more DCCs
- TP\_CRR Directive is utilized by the DTS and created by none or one DCC

The content of a Register Element following a TP\_CRW Directive is discarded in the following situations:

- There are no PDCs
- There is one PDC and any of the following are true:
  - The PDCx\_SEL Register value for this PDC is equal to 00b
  - The PDCx\_DCC Register is implemented, and this register value does not select a DCC
- There are two PDCs and the either of the following is true for both DCCs:
  - The PDCx\_SEL Register value for this PDC is equal to 00b
  - The PDCx\_DCC Register is implemented, and this register value does not select a DCC

The PDCs utilize the content of a Register Element following a TP\_CRW Directive based on their individual PDCx\_SEL Register value. Neither, one, or both PDCs may utilize the Register Element content. A PDC utilizes Register Element content by forwarding it to:

- A DCC directly connected to it
- A DCC that is selected with the PDCx\_DCC Register value when a DCR is implemented

A TAP.7 Controller creates Register Element content following a TP\_CRR Directive when the T5 TAP.7 configuration implements:

- A single Physical Data Channel and the PDCx\_SEL Register value for this channel equals 10b
- Two Physical Data Channels and either the PDC0\_SEL or the PDC1\_SEL Register value is equal to 10b

When these conditions are not satisfied, the TMSC signal remains at high impedance during the Register Element. When no TAP.7 Controllers sharing the DTS connection drive the TMSC signal during a Register Element following a TP\_CRR Directive, the Register Element data is determined by the value of the last bit of the TP\_CRR Directive as the value of this bit is kept during the Register Element.

The PDCs create the Register Element content based on their individual PDCx\_SEL Register value. Neither or one PDC can create Register Element content. A PDC creates Register Element content using:

- Register data provided by a DCC directly connected to it

- Register data provided by a DCC that is selected with the PDCx\_DCC Register value when a DCR is implemented
- An all-zeros value when a DCR is implemented and the PDCx\_DCC Register value does not select a DCC

### 28.2.3.2 Specifications

#### Rules

- a) Each subsequent specification in 28.2.3.2 shall only apply to a T5 and above TAP.7.
  - b) The format, timing, and TMSC signal drive characteristics of TP Register Elements shall be governed by Figure 28-5.
  - c) A TAP.7 Controller shall create the content of a Register Element by driving the TMSC signal as shown in Figure 28-5 with the Single Drive type, provided any of the following are true:
    - 1) All of the following are true:
      - i) A TP\_CRR Directive precedes the Register Element.
      - ii) PDC0 is implemented.
      - iii) The PDC0\_SEL Register value is 10b.
    - 2) All of the following are true:
      - i) A TP\_CRR Directive precedes the Register Element.
      - ii) A PDC1 is implemented.
      - iii) The PDC1\_SEL Register value is 10b.
- with the Inhibited Drive type used during Register Element bit periods otherwise.
- d) When the conditions in Rule 28.2.3.2 c) creating Single Drive are not satisfied, the TAP.7 Controller shall not affect the content of a Register Element.
  - e) When the conditions specified by Rule 28.3.2.2 c) 1) are satisfied, the creation of Register Element content shall be governed by Table 28-1, where the x in PDCx equals 0.
  - f) When the conditions specified by Rule 28.3.2.2 c) 2) are satisfied, the creation of Register Element content shall be governed by Table 28-1, where the x in PDCx equals 1.

**Table 28-1 — Register Element content creation**

DCR implemented with PDCx ?	DCC selected with PDCx_DCC Register ?	Register Element content created with:
No	N/A	The value of the DCC Register specified by the TP_CRR Directive in the DCC directly connected to the PDC.
Yes	No	A default all-zeros value.
Yes	Yes	The value of the DCC Register specified by the TP_CRR Directive in the selected DCC.

- g) The content of a Register Element shall be forwarded to and utilized by a DCC connected to the selected PDC when any of the following are true:
  - 1) All of the following are true:

- i) A TP\_CRW Directive precedes the Register Element.
  - ii) The DCC is connected to Physical Data Channel Zero.
  - iii) PDC0\_SEL Register value is greater than 00b.
- 2) All of the following are true:
- i) A TP\_CRW Directive precedes the Register Element.
  - ii) The DCC is connected to Physical Data Channel One.
  - iii) PDC1\_SEL Register value is greater than 00b.
- h) When Rule 28.3.2.2 g) 1) is satisfied, the utilization of Register Element content shall be governed by Table 28-2, where the x in PDCx equals 0.
- i) When Rule 28.3.2.2 g) 2) is satisfied, the utilization of Register Element content shall be governed by Table 28-2, where the x in PDCx equals 1.

**Table 28-2 — Register Element content utilization**

DCR implemented with PDCx ?	DCC selected with PDCx_DCC Register ?	Register Element content affects:
No	N/A	<b>The value of the DCC Control Register specified by the TP_CRW Directive in the DCC directly connected to the PDC.</b>
Yes	No	<b>No DCC.</b>
Yes	Yes	<b>The value of the DCC Control Register specified by the TP_CRW Directive in the selected DCC.</b>

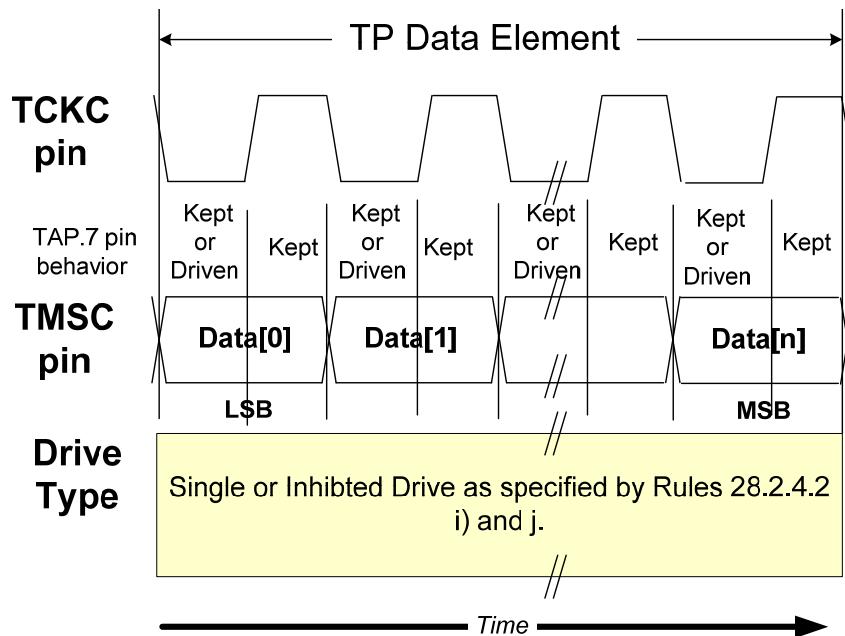
- j) When a DCC Register is the source of Register Element content, the value of bit[n] of the DCC Register specified by a TP\_CRR Directive shall be the value of bit[n] of the Register Element.
- k) When DCC Register is the destination of the Register Element content, the value of bit[n] of the DCC Register specified by a TP\_CRW Directive shall be affected by the value of bit[n] of the Register Element.

## 28.2.4 Data Element characteristics

### 28.2.4.1 Description

#### 28.2.4.1.1 Format, timing, and TMSC signal drive characteristics

TP Data Elements are transmitted LSB first. Their format, timing, and TMSC signal drive characteristics are shown in Figure 28-7.



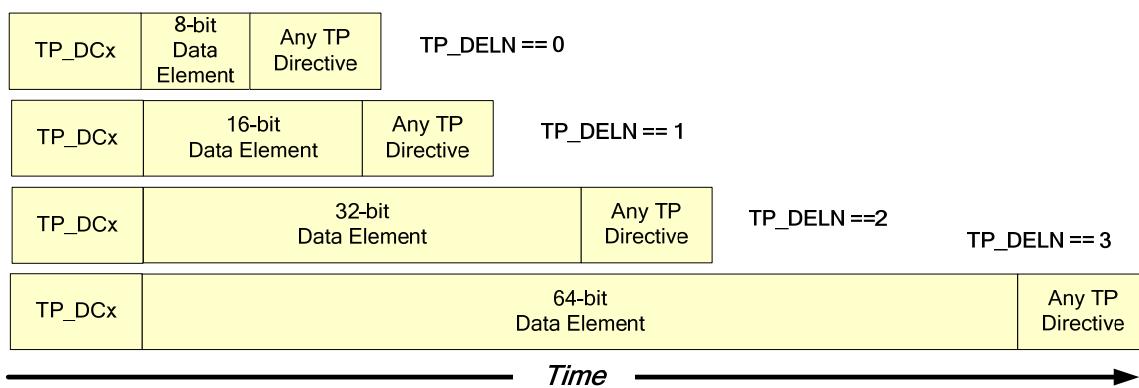
NOTE—The numbering of Data Elements has no significance to the underlying data being transferred other than conveying the transmission order of Data Element bits.

**Figure 28-7 — TP Data Element format, timing, and TMSC signal drive characteristics**

#### 28.2.4.1.2 Data Element length

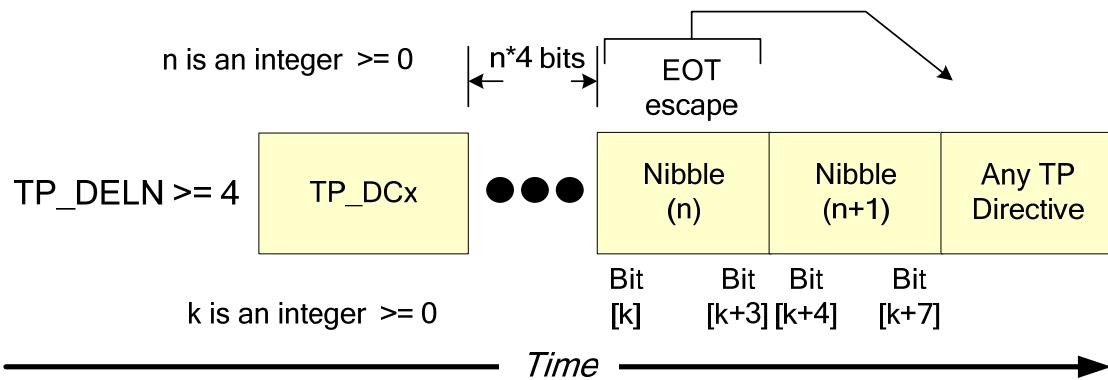
The TP\_DELN Register defines the length of the Data Elements used by all LDCs. The value of the TP\_DELN Register may be changed between Data Elements. A change to the TP\_DELN Register value affects subsequent Data Elements.

Fixed-length Data Elements and the TP\_DCx Directive initiating them are shown in Figure 28-8. A fixed-length Data Element terminates after the transfer of the number of bits specified by the TP\_DELN Register.



**Figure 28-8 — Fixed-length Data Payload with surrounding TP Directives**

A variable-length Data Element and the TP\_DCx Directive initiating it are shown in Figure 28-9. The Data Element is terminated following nibble  $n + 1$  when an EOT Escape occurs coincidently with any bit of nibble  $n$  as shown in this figure.



**Figure 28-9 — Variable-length Data Payload with surrounding headers**

#### 28.2.4.1.3 Data Element content

The creation and utilization of Data Element content is defined by one of the following:

- A DCC is connected directly to the PDCx.
- The DCC selected by the PDCx\_DCC Register when the PDCx/DCC connection is made via a DCR.
- No DCC is selected by the PDCx\_DCC Register, preventing the creation and utilization of Data Element content.

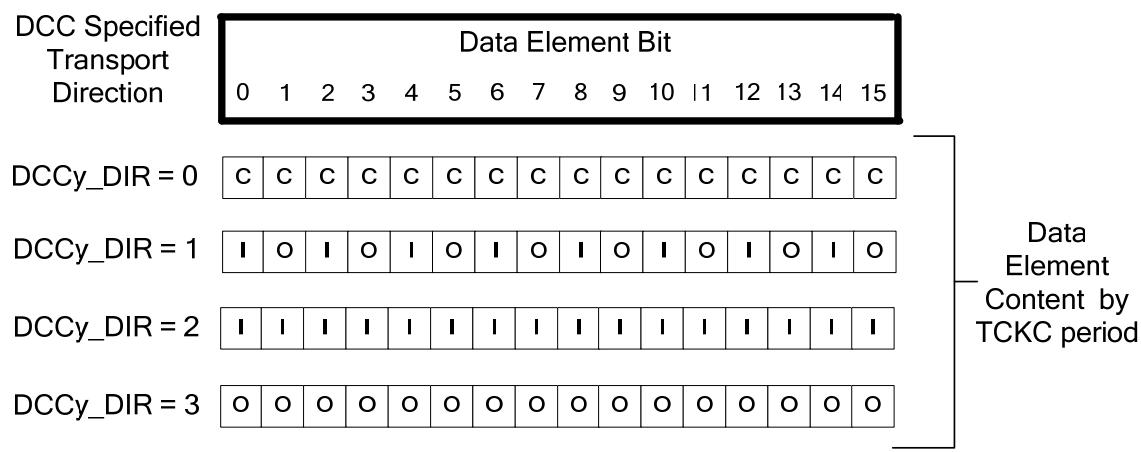
When DCCy is connected to the PDC either directly or via a DCR, the value of the DCCy\_DIRA field of its PDCx\_DCCy\_CR1 Register controls the creation and utilization of Data Element content as shown in Figure 28-10 and Table 28-3. This register defines the Data Payload content as one of the following:

- TS to DTS—All bits within the payload are transferred from the TS to the DTS.
- DTS to TS—All bits within the payload are transferred from the DTS to the TS.
- Bidirectional:
  - Even-numbered bits are transferred from the DTS to the TS starting with bit zero.
  - Odd-numbered bits are transferred from the TS to the DTS.
- Custom:
  - The DTS and the Data Channel Client(s) using the channel utilize a mutually agreed upon TMSC Drive Policy for all bits within the Data Element.
  - The TMSC drive characteristics are defined by a higher level protocol on a bit-by-bit basis.

**Table 28-3 — Data Element content**

DCCy_DIRA of the selected or directly connected DCC	Transfer direction	Source of Data Element bit	
		Odd-numbered bits (1, 3, ...)	Even-numbered bits (0, 2, ...)
00	Custom	Determined by the Data Channel Client	
01	Bidirectional	TS	DTS
10	DTS sends to TS	DTS	DTS
11	TS sends to DTS	TS	TS

When there is no DCC connected to the PDC, there is no creation and no utilization of Data Element content (the TMSC drive type is Inhibited Drive).



Legend : I = Input, O = Output, C = Custom TMSC Drive

**Figure 28-10 — DCCy\_DIRA Register/Data Payload content relationships**

#### 28.2.4.1.4 Drive characteristics

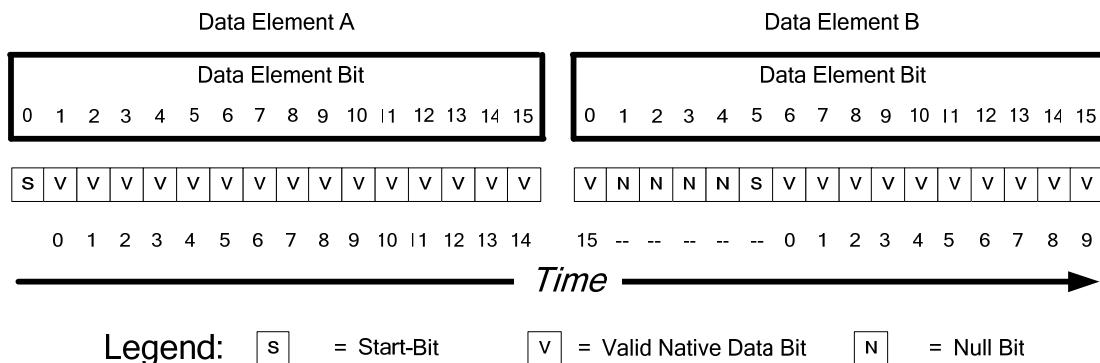
The TMSC drive initiated by a PDC is inhibited during Data Element bit periods when the LCA is invalid (0xxxxb) or indicates the LCA may be allocated to more than one PDC (11xxx) and the DCCy\_DIRA Register value presented by the selected DCC indicates a transfer direction other than Custom. It is also inhibited when the LCA is not specified by the TP\_DCx Directive initiating the Data Element. The TMSC drive during these bit periods is determined by the transfer direction otherwise.

#### 28.2.4.1.5 Multi-client data transfers

When an LDC is constructed with multiple PDCs, the DTS is responsible for ensuring the DCCy\_DIRA fields of the PDCx\_DCCy\_CR1 Register of the PDCs associated with the same LCA specify Custom direction. It is recommended this register be programmed simultaneously in all DCCs. Subsequent to this, it is the responsibility of the DTS and DCC(s) to use a high-level protocol to ensure there are no TMSC-drive conflicts during Data Element bit periods.

#### **28.2.4.1.6 Data Element alignment with the data that is transported**

The data transferred within a Data Element may not be aligned with the natural boundaries of the native data transported by it. The alignment (or misalignment) of native data with Data Elements is illustrated in Figure 28-11. In this example, the native data is transmitted using a start-bit preceding the 16-bit data values. Any number of null bits (non-start bits) may follow bit 15 of the native data (i.e., they precede a start-bit) in this example.



NOTE—The numbering of the data bits has no significance other than to convey their quantity and position. The data format is determined by the DTS and DCC(s).

**Figure 28-11 — Data payload and data channel data alignment relationship**

#### **28.2.4.2 Specifications**

## Rules

- a) Each subsequent specification in 28.2.4.2 shall only apply to a T5 TAP.7.
  - b) The format, timing, and TMSC signal drive characteristics of TP Data Elements shall be governed by Figure 28-7.
  - c) The length in bits of a Data Element shall be governed by the TP\_DELN Register value.
  - d) Changing the TP\_DELN Register value between Data Elements shall be permitted.
  - e) When the TP\_DELN Register value specifies the use of fixed-length Data Elements, the processing of a Data Element shall be terminated after the number of bits specified by this register, as shown in Figure 28-8.
  - f) When the TP\_DELN Register value specifies the use of variable-length Data Elements, the processing of a Data Element shall be terminated as shown in Figure 28-9 after the Data Element bit that is numbered  $k + 7$ , provided all of the following are true:
    - 1)  $k \geq 0$ .
    - 2)  $k$  is an integer multiple of four.
    - 3) An EOT Escape has occurred coincidentally with a Data Element bit with a number that is both  $\geq k + 0$  and  $\leq k + 3$ .
  - g) When DCCy is connected to a PDC, the DCCy\_DIRA field of the PDCx\_DCCy\_CR1 Register as specified in Table 27-4 shall govern the TMSC signal drive characteristics for Data Element bit periods as shown in Table 28-3.
  - h) When there is no DCC connected to a PDC, the PDC shall discard Data Element input content.

i) During Data Element bit periods, PDCx shall request the drive of the TMSC signal using the Single Drive type, provided any of the following are true:

- 1) All of the following are true:
  - i) The TP\_DCx Directive Element specifies a data exchange with the Logical Channel z.
  - ii) The PDCx\_LCA Register value is 10CCC where CCC is a value equal to z.
  - iii) A DCC is connected to PDCx.
  - iv) The DCC connected to PDCx specifies the direction of the transfer as either TS sends to DTS or Bidirectional with an odd Data Element Bit number (see Table 28-3) and specified by Rule 28.2.4.2 g).
- 2) All of the following are true:
  - i) The TP\_DCx Directive Element specifies a data exchange with the Logical Channel z
  - ii) The PDCx\_LCA Register value is 1xCCC where CCC is equal to z
  - iii) A DCC is connected to PDCx
  - iv) The DCC connected to PDCx specifies the direction of the transfer as Custom (see Table 28-3)
  - v) The DCC indicates the TMSC signal should be driven (with the *dcc\_cor* signal sourced by this DCC

and requests the use of the Inhibited Drive type used during Data Element bit periods otherwise.

- j) A TAP.7 Controller shall create the content of Data Element bits by driving the TMSC signal using the Single Drive type as shown in Figure 28-7, provided all of the following are true:
- 1) At least one PDC is implemented
  - 2) Either PDC0 or PDC1 requests the TMSC signal be driven during a Data Element bit period

with the Inhibited Drive type used during Data Element bit periods otherwise.

## 28.3 Transport State Machine

### 28.3.1 Description

#### 28.3.1.1 Conceptual view

The Transport State Machine (TSM) manages the use of TPs. A conceptual view of the TSM is shown in Figure 28-12. The TSM states are described in Table 28-4. The states and state encoding of this machine is fixed by this standard as the TSM state is included in the PDCx/DCC interface that is described in 28.4. This facilitates the interoperability of DCC components with T5 TAP.7 implementations.

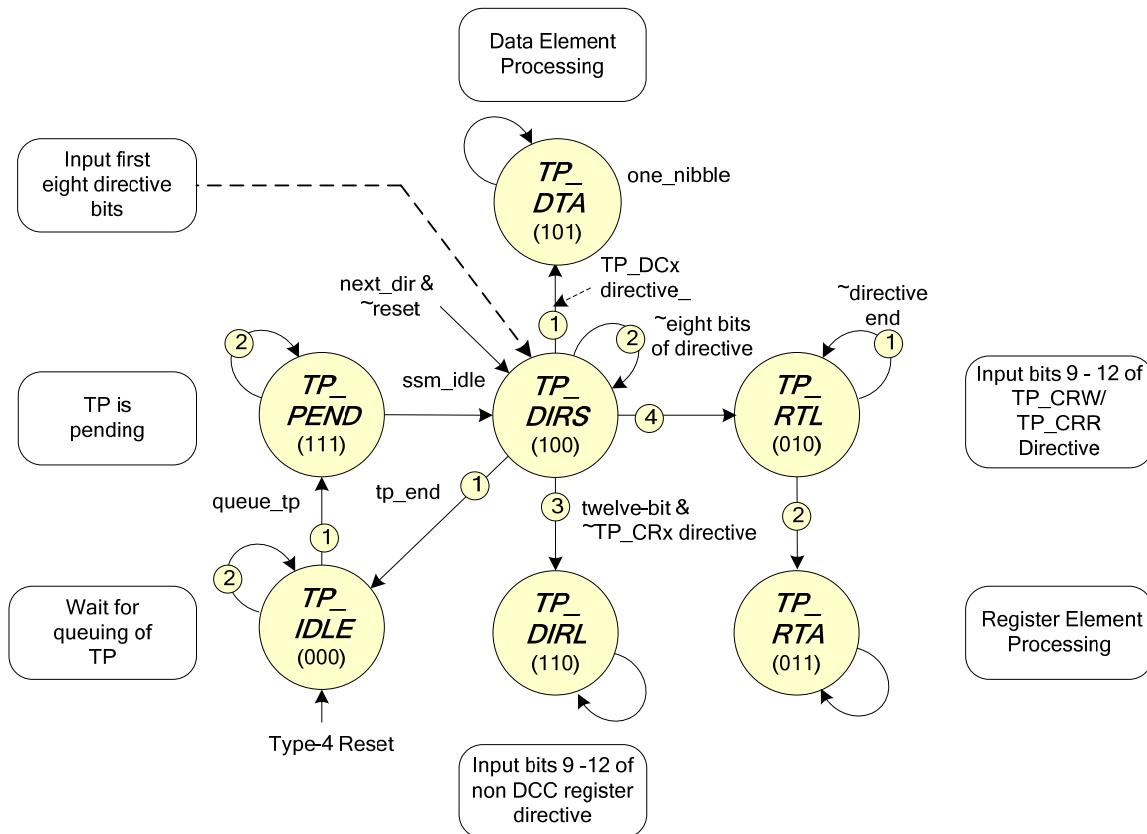


Figure 28-12 — Conceptual view of the Transport State Machine

Table 28-4 — TSM state names and encoding

Mnemonic	State encoding	Name	Description
<i>TP_IDLE</i>	000b	Transport Idle	A TP is not active or queued to follow an SP.
RSVD	001b	Reserved	This state is reserved.
<i>TP_RTL</i>	010b	Reg. Transfer Long	Bits 9–12 of a TP_CRR or TP_CRW Directive are being input, the register number is presented statically to the DCC.
<i>TP_RTA</i>	011b	Reg. Transfer Active	The transfer of nibbles n through 2 of the Register Element is ongoing. The register number is statically presented to the DCC.
<i>TP_DIRS</i>	100b	Directive Short	Bits 1–8 of a directive are being input and decoded.
<i>TP_DTA</i>	101b	Data Transfer Active	The transfer of a Data Element is ongoing.
<i>TP_DIRL</i>	110b	Directive Long	Bits 9–12 of a 12-bit directive other than TP_CRR or TP_CRW are being input and decoded.
<i>TP_PEND</i>	111b	Transport Pending	A TP is queued and awaits completion of the SP.

With the encoding shown in this table, a TSM[2] logic 0 value identifies a Register Element transaction, while a TSM[2] logic 1 value identifies a Data Element transaction. TSM[1:0] provides information that may be used to create pipelined Register Element transactions.

**Table 28-5 — TSM state behavior**

State	Type-0- Type-4 Reset	CSM State == ADV	VState supports TP & all idle & ~CP initiate?	ssm_idle ?	TP-END, TP_RSZ, or TP_RO early decode?	TP_DCx decode?	First eight bits of directive input?	Eight-bit directive?	TP_CRx Directive?	Data Element complete?	Long directive complete?	Last Register Element bit?	Next state (TP_xxxx)
x	y	x	x	x	x	x	x	x	x	x	x	x	IDLE
x	x	N	x	x	x	x	x	x	x	x	x	x	IDLE
IDLE	N	Y	N	x	x	x	x	x	x	x	x	x	IDLE
IDLE	N	Y	Y	x	x	x	x	x	x	x	x	x	PEND
PEND	N	Y	x	N	x	x	x	x	x	x	x	x	PEND
PEND	N	Y	x	Y	x	x	x	x	x	x	x	x	DIRS
DIRS	N	Y	x	x	Y	x	x	x	x	x	x	x	IDLE
DIRS	N	Y	x	x	N	Y	x	x	x	x	x	x	DTA
DIRS	N	Y	x	N	N	N	Y	N	N	x	x	x	DIRL
DIRS	N	Y	x	N	N	N	Y	N	Y	x	x	x	RTL
DIRS	N	Y	x	N	N	N	Y	Y	N	x	x	x	DIRS
DIRS	N	Y	x	N	N	N	N	N	N	x	x	x	DIRS
DTA	N	Y	x	x	x	x	x	x	x	N	x	x	DTA
DTA	N	Y	x	x	x	x	x	x	x	Y	x	x	DIRS
DIRL	N	Y	x	x	x	x	x	x	x	x	N	x	DIRL
DIRL	N	Y	x	x	x	x	x	x	x	x	Y	x	DIRS
RTL	N	Y	x	x	x	x	x	x	x	x	N	x	RTL
RTL	N	Y	x	x	x	x	x	x	x	x	Y	x	RTA
RTA	N	Y	x	x	x	x	x	x	x	x	x	N	RTA
RTA	N	Y	x	x		x	x	x	x	x	x	y	DIRS

### 28.3.1.2 TSM operation

A brief description of the operation of the TSM follows. A Type-4 Reset or a CSM state other than *ADV* forces the TSM to its *IDLE* state. This handles the TSM operation with selection and deselection events as these events generate a CSM state other than *ADV*.

The TSM state moves from *IDLE* to *PEND* when a TP is queued. This occurs when the:

- CSM, SSM, and TSM State Machine states are *IDLE*

- SP is associated with a TAPC state that supports transport
- SP is not to be followed by a CP

The TSM begins processing the TP one Test Clock period following the SSM reaching its *IDLE* state. This coincides with the registering of the first bit of the TP with the falling edge of the TCKC signal (in the register creating the *tmsc\_r* signal).

When the state is *DIRS*, the:

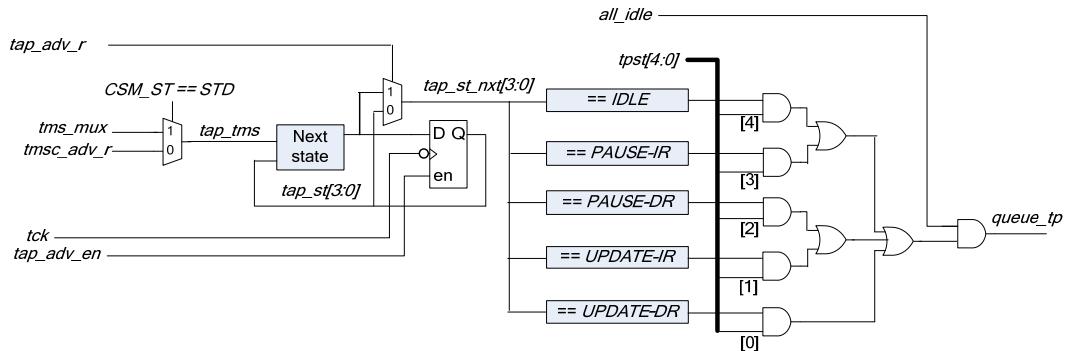
- *DIRS* to *IDLE* state transition occurs with the decoding of the TP-END, TP-RSO, and TP-RSZ Directives
- *DIRS* to *DTA* state transition occurs with the decoding of a TP-DCx Directive
- Processing of the five-bit NOP Directive and Eight-bit directives is completed and the directive processing is restarted with no change in the TSM state
- The first eight bits of 12-bit directives are processed with:
  - A *DIRS* to *DIRL* state change to process the four remaining bits of 12-bit directives other than TP-CRR and TP-CRW Directives
  - A *DIRS* to *RTL* state change to process the four remaining bits of the 12-bit TP-CRR and TP-CRW Directives

Special consideration should be given to *DIRS* to *IDLE* state transitions as they require the use of the *tmsc\_mux* signal to create an early decode of the TP-END, TP-RSO, and TP-RSZ Directives (see Figure 28-30). The early decode of these directives causes a *DIRS* to *IDLE* state transition when the last bit of the directive becomes registered (see Figure 28-16).

With the *DIRL* state, the completion of a 12-bit directive causes a *DIRL* to *DIRS* state change. With the *RTL* state, the completion of the TP-CRR and TP-CRW Directives causes a *RTL* to *RTA* state transition. A *RTA* to *DIRS* state transition occurs when the transfer of the Register Element completes.

### 28.3.1.3 Scheduling a TP

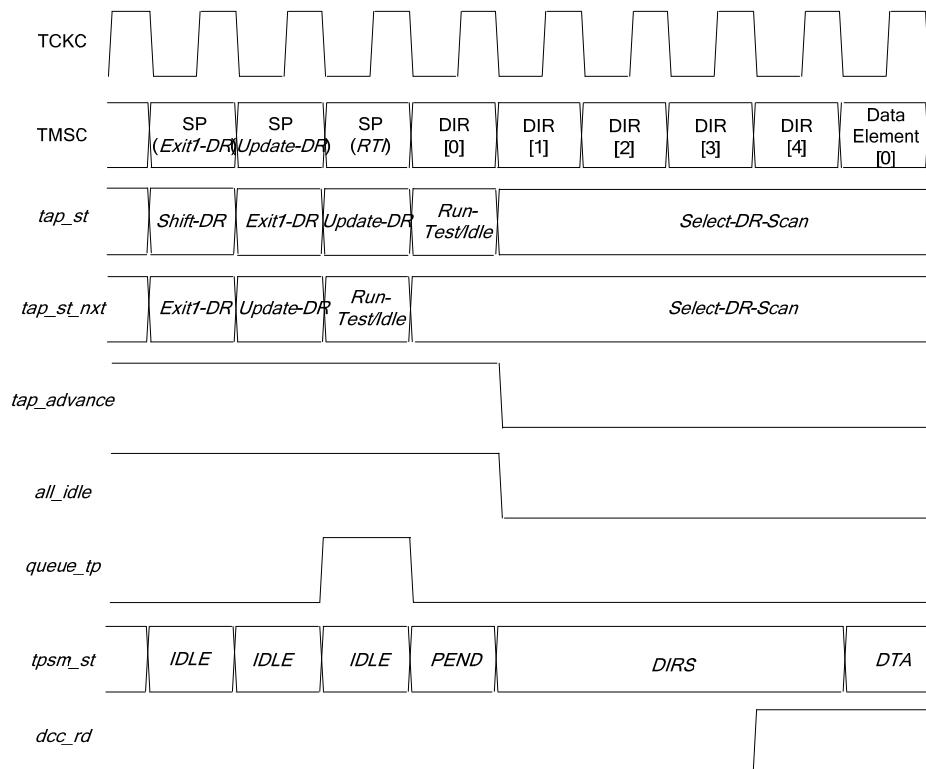
The decision as to whether a TP follows an SP is made during the first bit of an SP when the Control, Scan, and Transport State Machines are in their *IDLE* states (the *all\_idle* signal in Figure 21-11 is a logic 1). Recall that an SP may create both a nonpipelined and a pipelined operation and a minimal SP may have a payload with only one bit. A method that understands both nonpipelined and pipeline operations is shown in Figure 28-13.



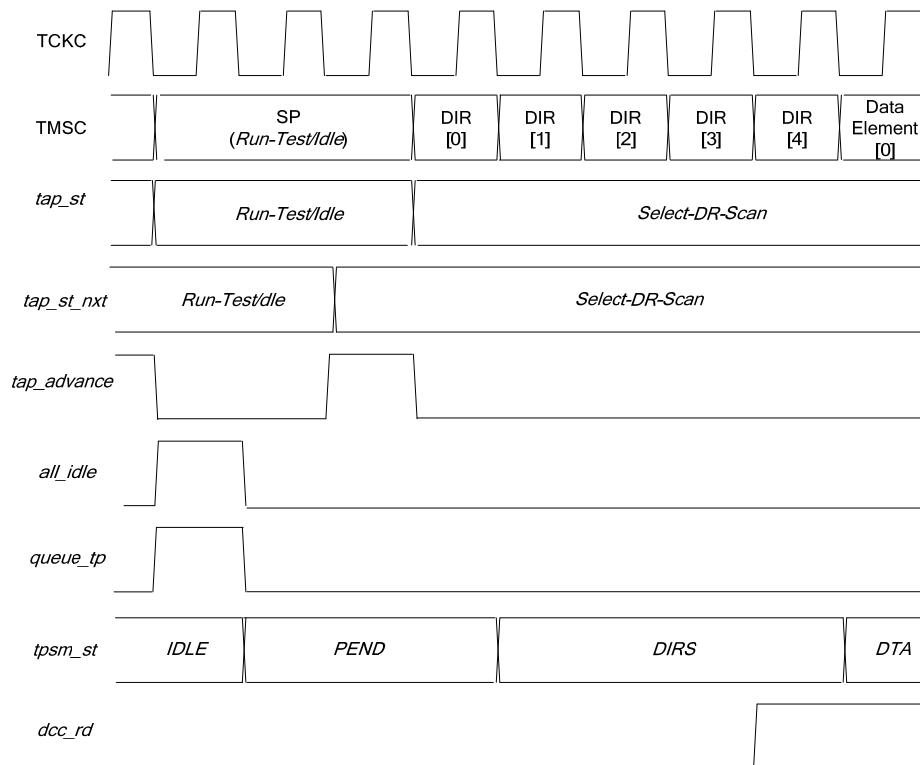
**Figure 28-13 — Queuing a TP to follow an SP**

The TP queuing decision is based on the virtual TAPC state at the first bit of an SP. This point in time represents the first bit of the SP associated with this virtual TAPC state. The real TAPC state cannot be used for the queuing decision as the TAPC state does not represent the TAPC state associated with the current SP with pipelined operation.

The queuing of a TP with the OScan2 Scan Format (pipelined operation) is shown in Figure 28-14. The queuing of a TP with the OScan1 Scan Format (nonpipelined operation) is shown in Figure 28-15.



**Figure 28-14 — Queuing a TP to follow an SP with the OScan2 Scan Format**



**Figure 28-15 — Queuing a TP to follow an SP with the OScan1 Scan Format**

#### 28.3.1.4 Starting/restarting directive processing

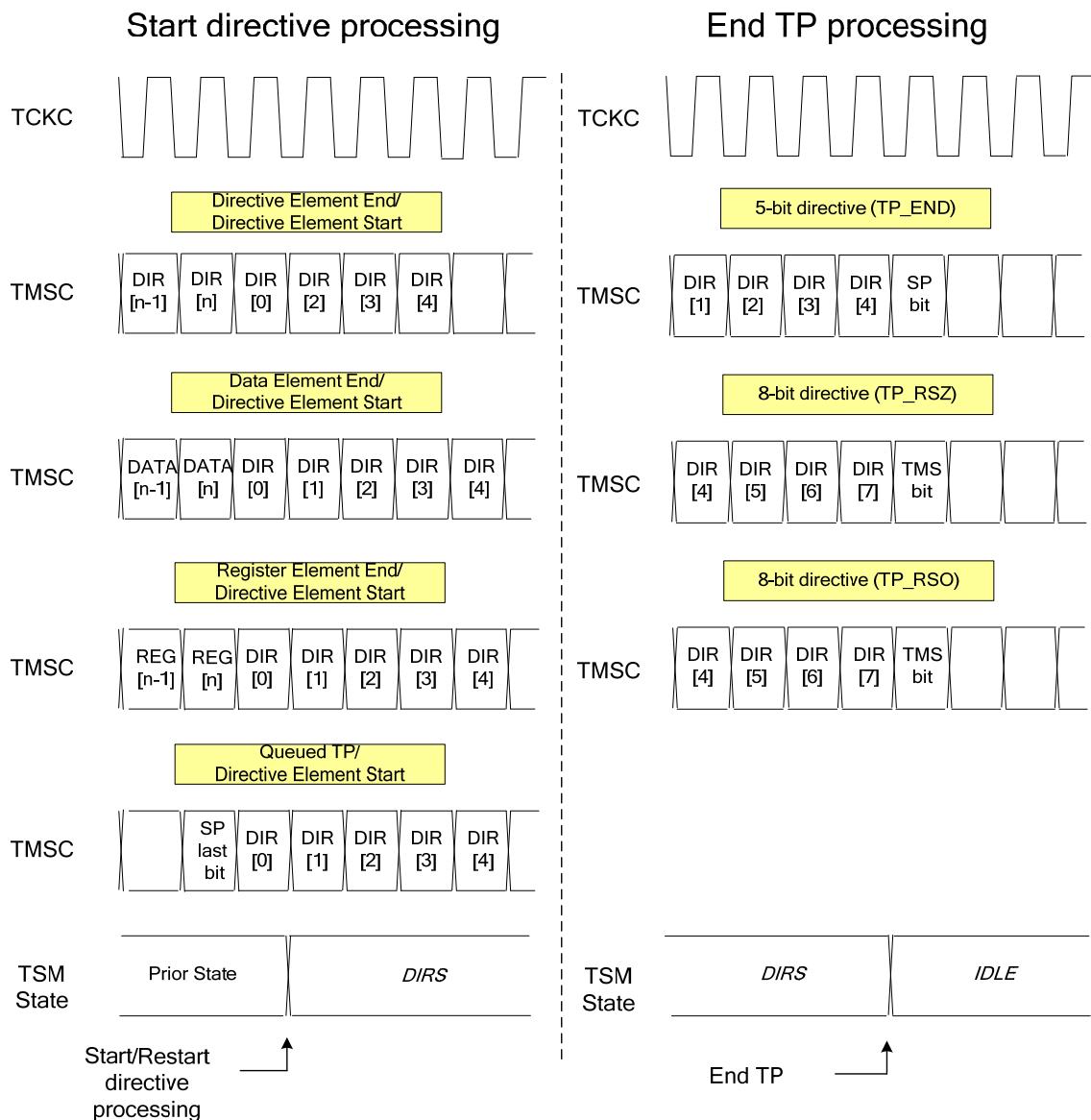
Transport Directive processing is started when TP processing is queued and an SP completes (a *PEND* to *DIRS* state transition). It is restarted by the completion of:

- |  |  |
|--|--|
| — A Five-bit directive other than <i>TP-END</i>                      | State remains <i>DIRS</i>                  |
| — An eight-bit directives other than <i>TP-RSO</i> and <i>TP-RSZ</i> | State remains <i>DIRS</i>                  |
| — A Register Element   | <i>RTA</i> to <i>DIRS</i> state transition |
| — A Data Element   | <i>DTA</i> to <i>DIRS</i> state transition |

These state changes occur coincidently with the registering of the MSB of the Transport Directive as shown in Figure 28-16.

#### 28.3.1.5 Completing a TP

The *TP-END*, *TP-RSO*, and *TP-RSZ* Directives cause the completion of a TP. They cause a *DIRS* to *IDLE* state transition. This state transition occurs coincidently with the registering of the TMSC bit that follows these directives as shown in Figure 28-16.



**Figure 28-16 — Starting directive processing/ending TP processing**

### 28.3.2 Specifications

#### Rules

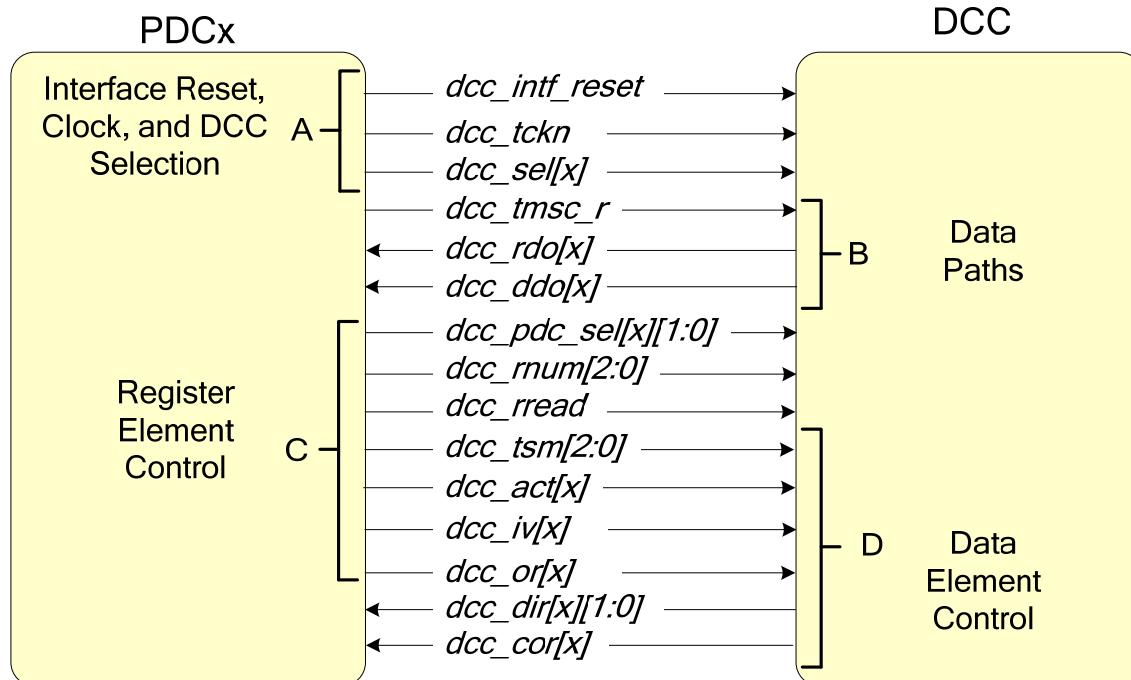
- Each subsequent specification in 28.3.2 shall only apply to a T5 TAP.7.
- The Transport Function shall include a TSM with the states and state encoding shown in Table 28-4.
- The operation of the TSM shall be governed by Table 28-5.
- A TP shall be processed after an SP when all of the following are true:
  - The SP that precedes the TP is associated with a TAPC state that supports transport.
  - The TPST Register enables transport for the TAPC state that is associated with the SP that precedes the TP.

- 3) A CP is not required to follow the SP.
- 4) The CSM state is *ADV* while processing the TP.
- e) TP processing shall end after the last bit of any of the following directives as shown in Figure 28-16:
  - 1) TP\_END.
  - 2) TP\_RSO.
  - 3) TP\_RSZ.

## 28.4 PDCx/DCC interface

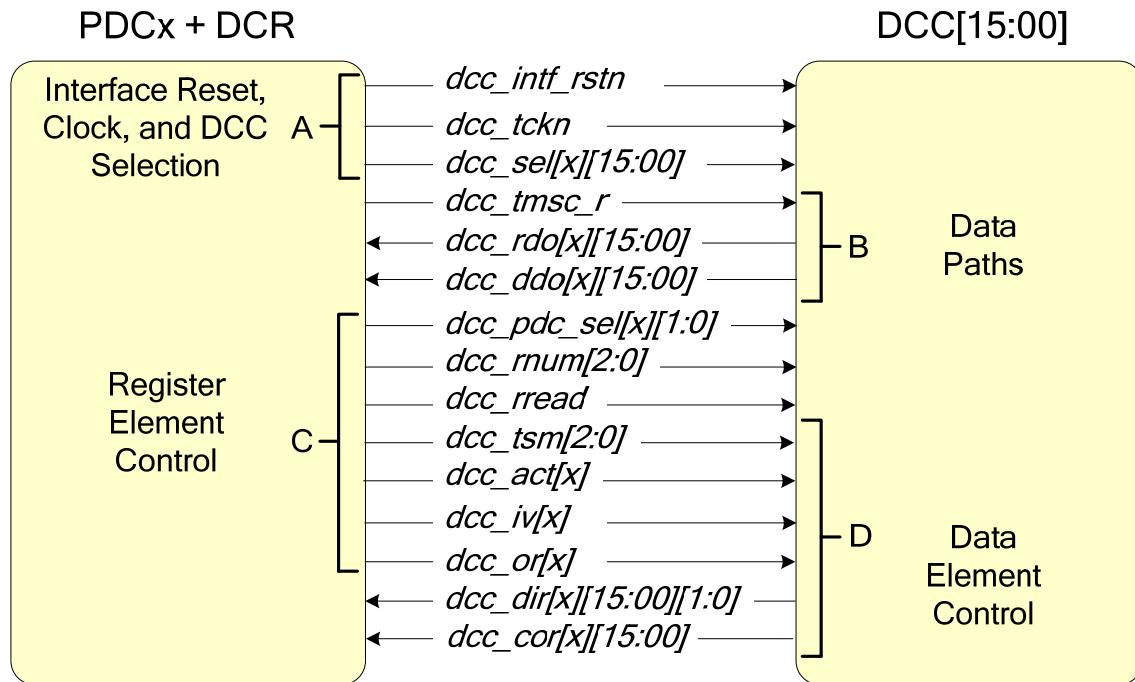
### 28.4.1 Description

The PDCx/DCC interface supports both direct connection to a single DCC as shown in Figure 28-17 and connection to multiple DCCs as shown in Figure 28-18. The connection of a specific DCC to the PDC may be either pipelined or nonpipelined. The PDC provides simple signaling for use with a nonpipelined connection along with sufficient control information to support a pipelined connection. The DCC handles the effects of pipelining using this control information. Both nonpipelined and pipelined DCCs can be used at the same time when multiple DCCs are connected to the PDC. This approach facilitates high-speed transport operation even when the DCC and PDC may be physically far from each other on a chip.



NOTE—[x] represents the PDC number (x = 0 – 1).

**Figure 28-17 — PDCx/DCC interface signals with a single DCC connected to the DCC**



NOTE—[x] represents the PDC number ( $x = 0 - 1$ ); [y] represents the DCC number (00–15).

**Figure 28-18 — PDCx/DCC interface signals with multiple DCCs sharing the PDC**

#### 28.4.1.1 Signal functions

The PDCx/DCC interface signals provide the functions shown in Table 28-6.

**Table 28-6 — PDCx/DCC interface signal functions**

Signal Group	Function	Description
A	Clock/Selection	Connects a DCC to the PDC.
B	Data Paths	Register and Data Element data is exchanged between the DCC and the PDC.
C	Register Element Control	Manages the exchange of Register Element information.
D	Data Element Control	Manages the exchange of Data Element information.

Signal Group A connects a DCC to the PDC. When a DCR is used, 1 of the 16  $dcc\_sel[x]$  signals is active. Group C signals manage the exchange of Register Element information in conjunction with the Group B signals. Group D signals manage the exchange of Data Element data in conjunction with the Group B signals. Both Group C and Group D signal activity creates Group B signal activity. Group C and Group D signals are not activated concurrently as Register and Data Elements occur at different times.

#### 28.4.1.2 Signal descriptions

The PDCx/DCC interface signals are described in Table 28-7 and Table 28-8.

**Table 28-7 — PDCx/DCC interface signal description, PDC sourced signals**

Signal	Description
<i>dcc_tckn</i>	<b>DCC Test Clock</b> The inverted value of the TCK signal.
<i>dcc_sel[x][15:00]</i>	<b>Data Channel Client select</b> 0:The DCC is not selected. 1:The DCC is selected. The decoded value of the PDC_DCC Register.
<i>dcc_intf_rstn</i>	<b>DCC interface reset</b> 0:Reset the DCC interface 1:Enable the DCC interface.
<i>dcc_tmse_r</i>	<b>DCC TMSC registered</b> The TMSC signal with falling-edge timing. This signal is derived from the <i>tmse_mux</i> signal (rising- and falling-edge TMSC sampling).
<i>dcc_pdc_sel[x][1:0]</i>	<b>DCC view of PDC_SEL Register</b> Equivalent to the PDC_SEL Register value.
<i>dcc_rnum[2:0]</i>	<b>DCC Register number</b> 0–7: The DCC Register number.
<i>dcc_read</i>	<b>DCC Register read</b> 0: The PDC sources all Register Element bits. 1: The DCC sources all Register Element bits.
<i>dcc_tsm[2:0]</i>	<b>DCC Transport State Machine</b> 000 – Transport is idle. 001 – Reserved. 010 – Input of last nibble of TP_CRR/TP_CRW Directive. 011 – Register Element is being transferred. 100 – Processing bits 0–7 of a directive. 101 – Data Element is being transferred. 110 – Input of last nibble of non-TP_CRR/TP_CRW Directive. 111 – A Transport Packet is pending.
<i>dcc_act[x]</i>	<b>DCC active</b> 0:The next <i>tmse_r</i> bit is not associated with this PDC. 1:The next <i>tmse_r</i> bit is associated with this PDC.
<i>dcc_iv[x]</i>	<b>DCC input valid</b> 0: Not the condition below. 1: The last TMSC bit period was part of a Data or Register Element associated with this PDC and was not sourced by this PDC.
<i>dcc_or[x]</i>	<b>DCC output request</b> 0: The PDC sources Data Element the bit value. 1: The DCC sources Data Element the bit value.

**Table 28-8 — PDCx/DCC interface signal description, DCC-sourced signals**

<b>Signal</b>	<b>Description</b>
<i>dcc_rdo</i> [x][15:00]	<b>DCC Register Element Data Out</b>  Read data supplied by the DCC for Register Element output.
<i>dcc_ddo</i> [x][15:00]	<b>DCC Data Element Data Out</b>  Read data supplied by the DCC for Data Element output.
<i>dcc_dir</i> [x][15:00][1:0]	<b>DCC Data Element Direction</b>  The value of the DCCy_DIRA field of the PDCx_DCCy_CR1 Register (specified in Table 27-4).
<i>dcc_cor</i> [x][15:00]	<b>DCC Custom Output Request</b>  0: The TMSC signal is not driven for Custom Data Element I/O. 1: The TMSC signal is driven for Custom Data Element I/O.

#### 28.4.1.3 Signal use

There are two types of PDCx/DCC interface signals, those that notify the DCC it should participate in the transaction and those that describe a transaction. Both Physical Data Channels share the signals that describe the transaction. These signals are sourced by the Transport Protocol Processing Unit shown in Figure 28-19.

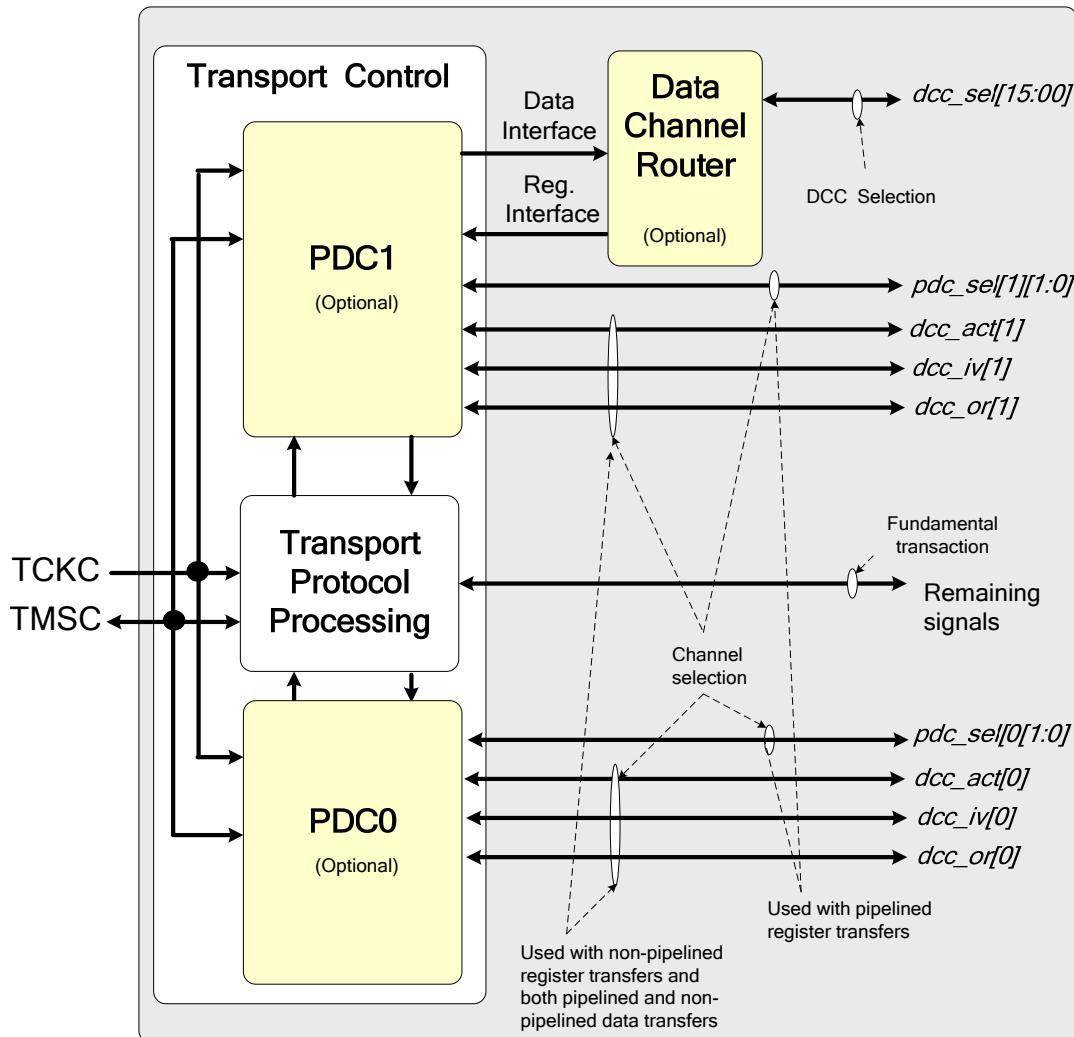


Figure 28-19 — Conceptual view of T5 TAP.7 Transport Control

Signaling initiating transactions are qualified with the  $dcc\_sel[x]\{15:0\}$  signals when a Data Channel Router is utilized. These signals are created with a four-line to 16-line decoder with the input being the PDC\_DCC Register value. A logic 1 signal connects a DCC to the PDC. A DCC presented a logic 0 signal value ignores other interface signaling.

In the case where there is no pipelining of Data and Register Element transfers and the DCC is selected, the set of signals listed as follows may be used to manage DCC register and Data Element transfers:

- $dcc\_tsm[2]$  Defines a transfer as either a Data or Register Element
- $dcc\_iv[x]$  Data are being sent to the DCC
- $dcc\_or[x]$  Data are being requested from the DCC
- $dcc\_rnum[2:0]$  DCC register number

The TSM state identifies a data exchange as a Register Element or Data Element using the  $dcc\_tsm[2]$  signal. The  $dcc\_iv$  and  $dcc\_or$  signals determine the transfer direction with a Register Element transaction. The  $dcc\_rnum$  signals define the target register for Register Element transactions.

In the case where there is pipelining of either Data and/or Register Element transfers, these or other signals may be used to manage DCC Register and Data Element transfers by creating signals with functions equivalent to the *dcc\_iv[x]* and *dcc\_or[x]* signals, only with timing accommodating pipelining. In this case, the DCC may determine whether it should participate in a Data or Register Element transaction using the *dcc\_act[x]*, *dcc\_pdc\_sel[x][1:0]*, *dcc\_tsm[2:0]*, and *dcc\_rread* signals. The *dcc\_act[x]* signal notifies the selected DCC the next TMSC bit period is a Register or Data Element bit period associated with the DCC.

With a Data Element transaction, the value of the DCCy\_DIRA field of the PDCx\_DCCy\_CR1 Register determines the direction of each bit in a Data Element as described in Table 28-3. The *dcc\_iv[x]* and *dcc\_or[x]* signals supplied by PDCx control the direction of the transfer with all input, all output, and bidirectional Data Element transactions. With custom Data Element transactions, the *dcc\_cor[x]* signal supplied by the DCC controls the direction of a transfer. In this case, *dcc\_iv[x]* and *dcc\_or[x]* signals remain inactive, with the DCC determining when input data is valid.

The *dcc\_pdc\_se[x][1:0]* signals make the PDCx\_SEL Register value available to the DCC for use in pipelining Register Element transactions. A combination of the *dcc\_pdc\_se[x][1:0]* and *dcc\_tsm[2:0]* signals may be used to identify a register transaction two clocks earlier than the *dcc\_act[x]* signal.

The PDCx/DCC interface signaling relationships are shown in Table 28-9. The Data and Register Element transactions are described by these relationships.

**Table 28-9 — PDCx/DCC interface signaling relationships**

<i>dcc_sel[x/y]</i>	<i>dcc_act[y]</i>	<i>dcc_tsm[2]</i>	<i>dcc_rread</i>	<i>dcc_dir[x/y][1:0]</i>	<i>dcc_cor[x/y]</i>	Description
0	x	x	x	x	x	The DCC is not selected and ignores other control signals.
1	0	x	x	x	x	The DCC is selected but a transaction has not been activated.
1	1	0	0	x	x	<ul style="list-style-type: none"> <li>— A Register Element associated with this PDC begins the next TMSC bit period (bit period [n]) with data sourced by the PDC.</li> <li>— The <i>dcc_iv</i> signal is active in TCK period[n + 2].</li> <li>— The <i>dcc_or</i> signal will remains inactive.</li> </ul>
1	1	0	1	x	x	<ul style="list-style-type: none"> <li>— A Register Element associated with this PDC begins the next TMSC bit period (bit period [n]) with data sourced by the DCC.</li> <li>— The <i>dcc_iv</i> signal remains inactive.</li> <li>— The <i>dcc_or</i> signal is active in TCK period[n].</li> </ul>
1	1	1	x	11	x	<ul style="list-style-type: none"> <li>— A Data Element associated with this PDC begins the next TMSC bit period (bit period [n]) with data sourced by the PDC with data provided by the selected DCC.</li> <li>— The <i>dcc_iv</i> signal remains inactive.</li> <li>— The <i>dcc_or</i> signal is active in bit period[n].</li> </ul>
1	1	1	x	10	x	<ul style="list-style-type: none"> <li>— A Data Element associated with this PDC begins the next TMSC bit period (bit period [n]) with data sourced by the DTS.</li> <li>— The <i>dcc_iv</i> signal is active in bit period[n + 2].</li> <li>— The <i>dcc_or</i> signal is inactive for the duration of the Data Element processing.</li> </ul>
1	1	1	x	01	x	<ul style="list-style-type: none"> <li>— A Data Element associated with this PDC begins the next TMSC bit period (bit period [n]) with data sourced by the PDC in even bit periods and the DTS in odd bit periods.</li> <li>— The <i>dcc_iv</i> signal is active in the bit period after a bit period where TMSC is sourced by the DTS.</li> <li>— The <i>dcc_or</i> signal is active in the bit periods before bit periods sourced by the PDC.</li> </ul>
1	1	1	x	00	0	<ul style="list-style-type: none"> <li>— The next bit period (n) is a Data Element bit that is not sourced by this PDC.</li> <li>— The <i>dcc_iv</i> and <i>dcc_or</i> signals are inactive for the duration of the Data Element processing.</li> </ul>
1	1	1	x	00	1	<ul style="list-style-type: none"> <li>— The next bit period (n) is a Data Element bit that is sourced by this PDC with data supplied by the selected DCC.</li> <li>— The <i>dcc_iv</i> and <i>dcc_or</i> signals are inactive for the duration of Data Element processing.</li> </ul>

## 28.4.2 Specifications

### Rules

- a) Each subsequent specification in 28.4.2 shall only apply to a T5 TAP.7.
- b) A PDC shall be connected to a single DCC with only a set of the interface signals shown in Figure 28-17.
- c) A PDC shall be connected to multiple DCCs with only a set of the interface signals shown in Figure 28-18.
- d) The function of the interface signals shown in Figure 28-17 and Figure 28-18 shall be governed by Table 28-7 and Table 28-8, respectively.
- e) The PDCx/DCC interface signaling relationships shall be governed by Table 28-9.

### Permissions

- f) The *dcc\_pdc\_se[x]/l[1:0]* and *dcc\_tsm[2:0]* signals may be omitted from the PDCx/DCC interface when the pipelining of Register Element transactions is not required.

## 28.5 Five-bit directives

### 28.5.1 Description

The PDCx/DCC interface timing generated with five-bit directives is shown in the following figures:

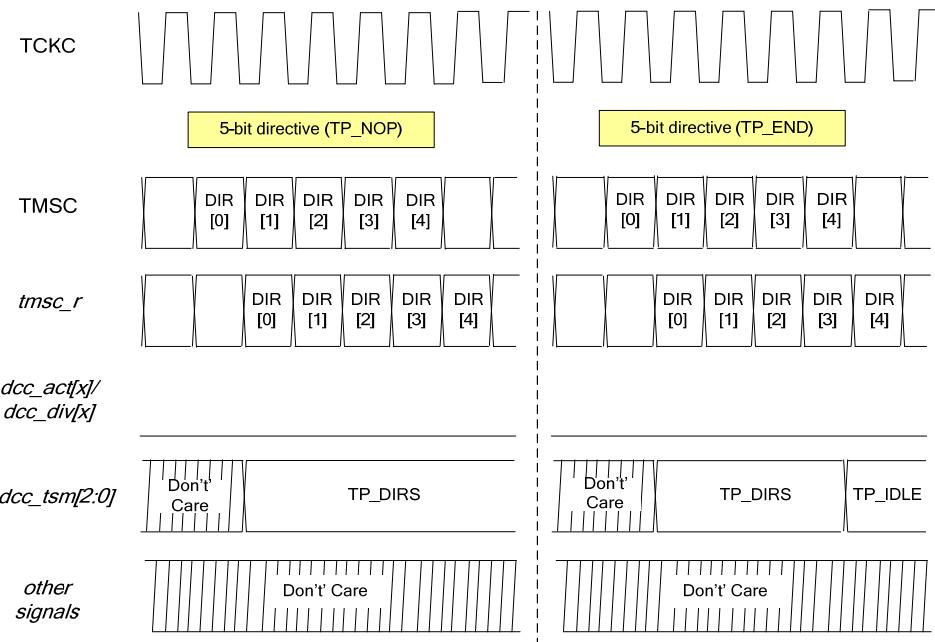
- TP\_NOP and TP\_END      Figure 28-20
- TP\_DCx      Figure 28-21

The four types of DCC data transfers specified in Table 28-3 are shown in Figure 28-21. The *dcc\_act*, *dcc\_iv*, *dcc\_or*, and *dcc\_cor* signals manage these transfers as shown. The TAP.7 Controller deals with a limited amount of data pipelining as shown in this figure.

### 28.5.2 Specifications

### Rules

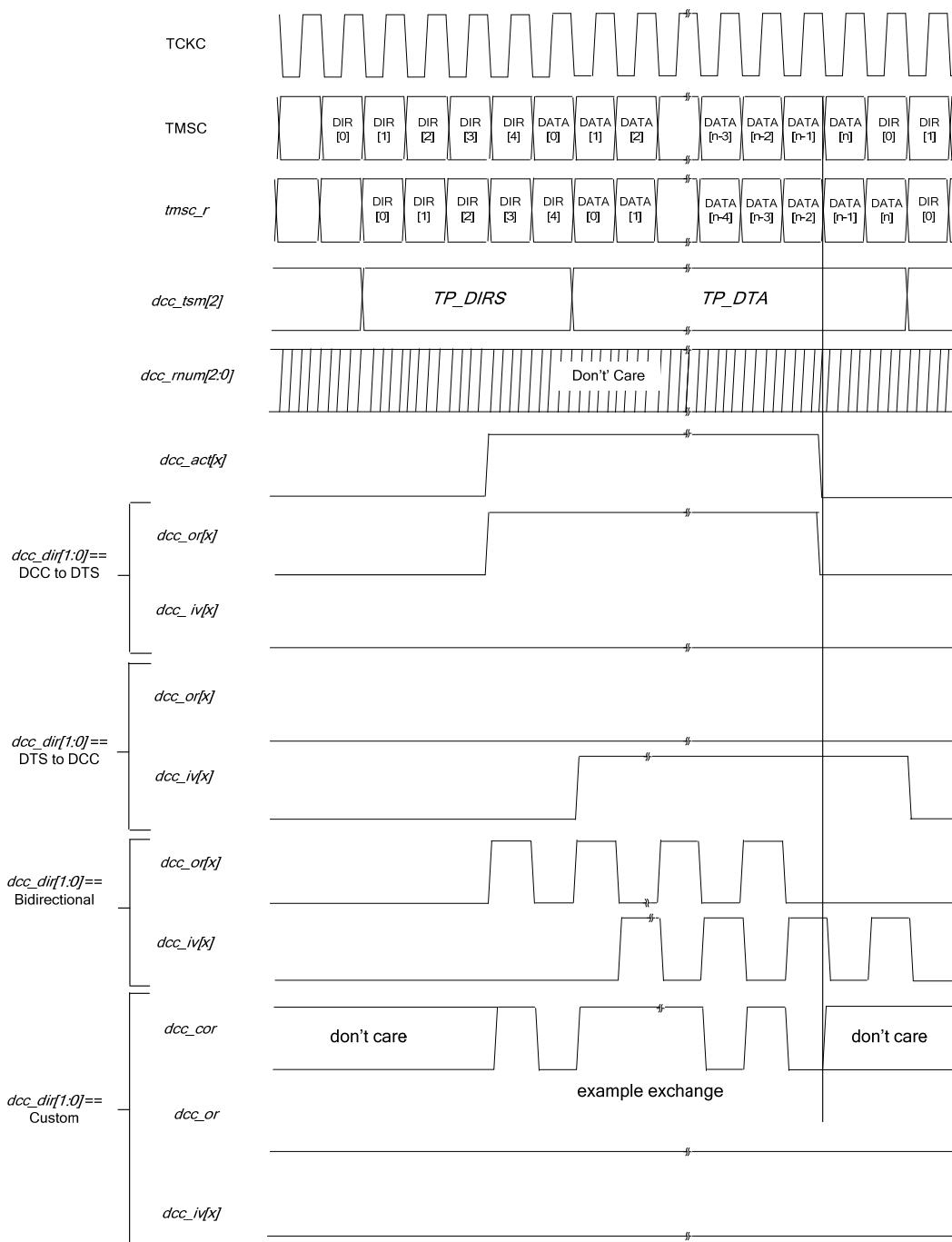
- a) Each subsequent specification in 28.5.2 shall only apply to a T5 TAP.7.
- b) The PDCx/DCC interface signaling generated by the TP\_NOP and TP\_END Directives shall be governed by Figure 28-20.



NOTE—[x] represents the PDC number (x = 0 – 1).

**Figure 28-20 — PDCx/DCC interface signaling/TP\_NOP and TP-END Directives**

- c) The PDCx/DCC interface signaling generated by the TP\_DCx Directives shall be governed by Figure 28-21.



NOTE—[x] represents the PDC number (x = 0 – 1). The number of data bits in the Data Segment is determined by the TP\_DELN Register value.

**Figure 28-21 — DCC interface signaling/TP\_DCx Directives**

## 28.6 Eight-bit directives

### 28.6.1 Description

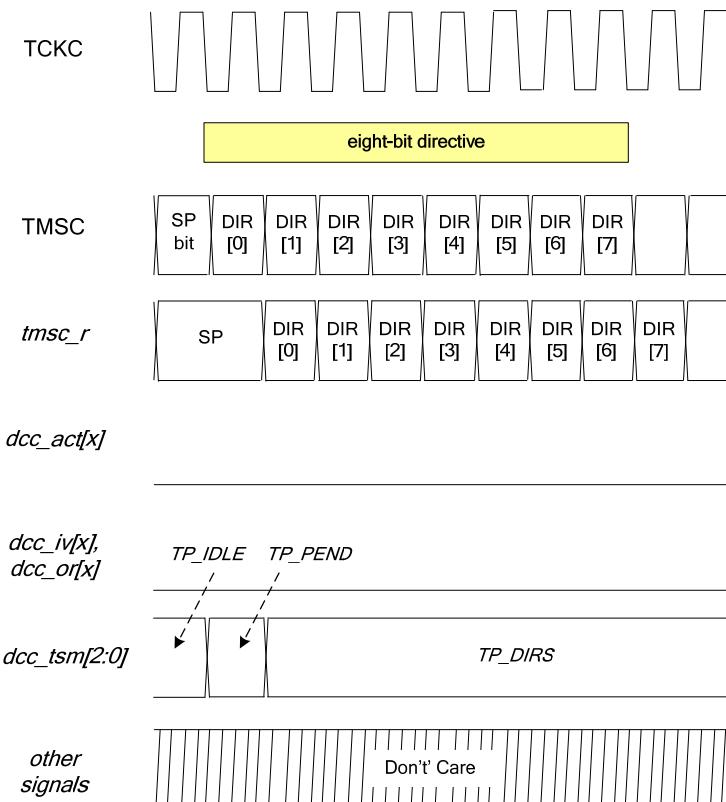
The PDCx/DCC interface timing generated with eight-bit directives is shown in the following figures:

- Eight-bit directives other than TP\_RSZ and TP\_RSO                          Figure 28-22
- TP\_RSZ and TP\_RSO    Figure 28-23

## 28.6.2 Specifications

### Rules

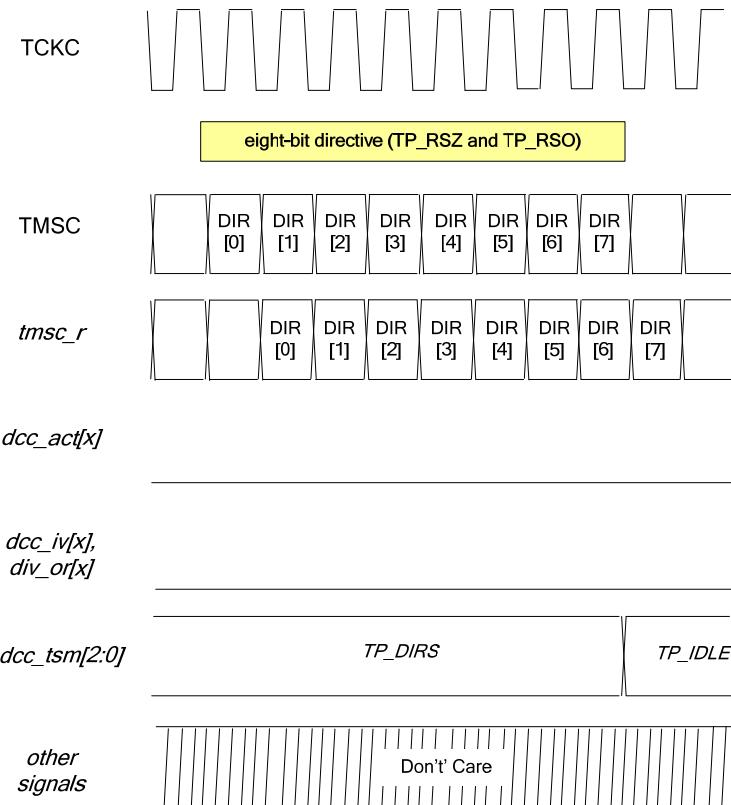
- a) Each subsequent specification in 28.6.2 shall only apply to a T5 TAP.7.
- b) The PDCx/DCC interface signaling generated by an eight-bit directive other than TP\_RSO and TP\_RSZ shall be governed by Figure 28-22.



NOTE—[x] represents the PDC number (x = 0 – 1).

**Figure 28-22 — PDCx/DCC interface signaling/nonterminating eight-bit directives**

- c) The PDCx/DCC interface signaling generated by the TP\_RSO and TP\_RSZ Directives shall be governed by Figure 28-23.



NOTE—[x] represents the PDC number (x = 0 – 1).

**Figure 28-23 — PDCx/DCC interface signaling/TP\_RSO and TP\_RSZ Directives**

## 28.7 12-bit directives

### 28.7.1 Description

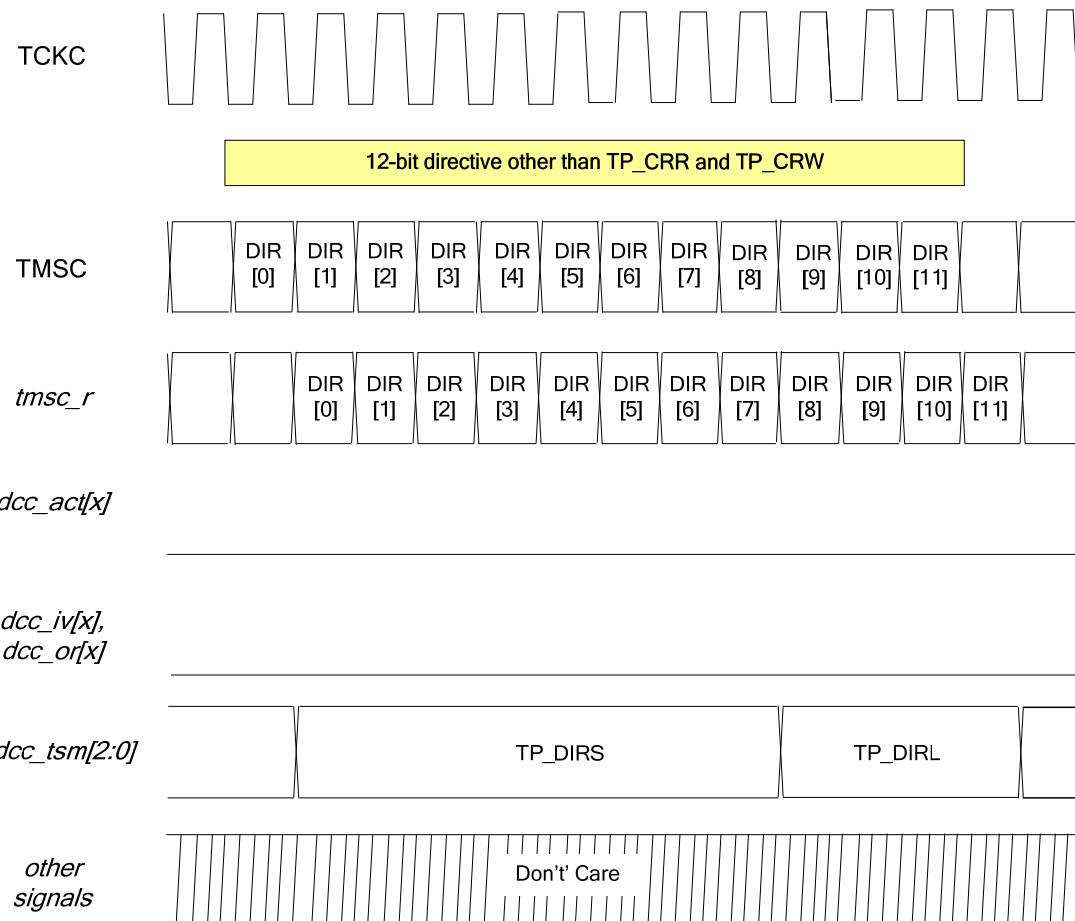
The PDCx/DCC interface timing generated with 12-bit directives is shown in the following figures:

- 12-bit directives other than TP\_CRR and TP\_CRW      Figure 28-24
- TP\_CRR      Figure 28-25
- TP\_CRW      Figure 28-26

### 28.7.2 Specifications

#### Rules

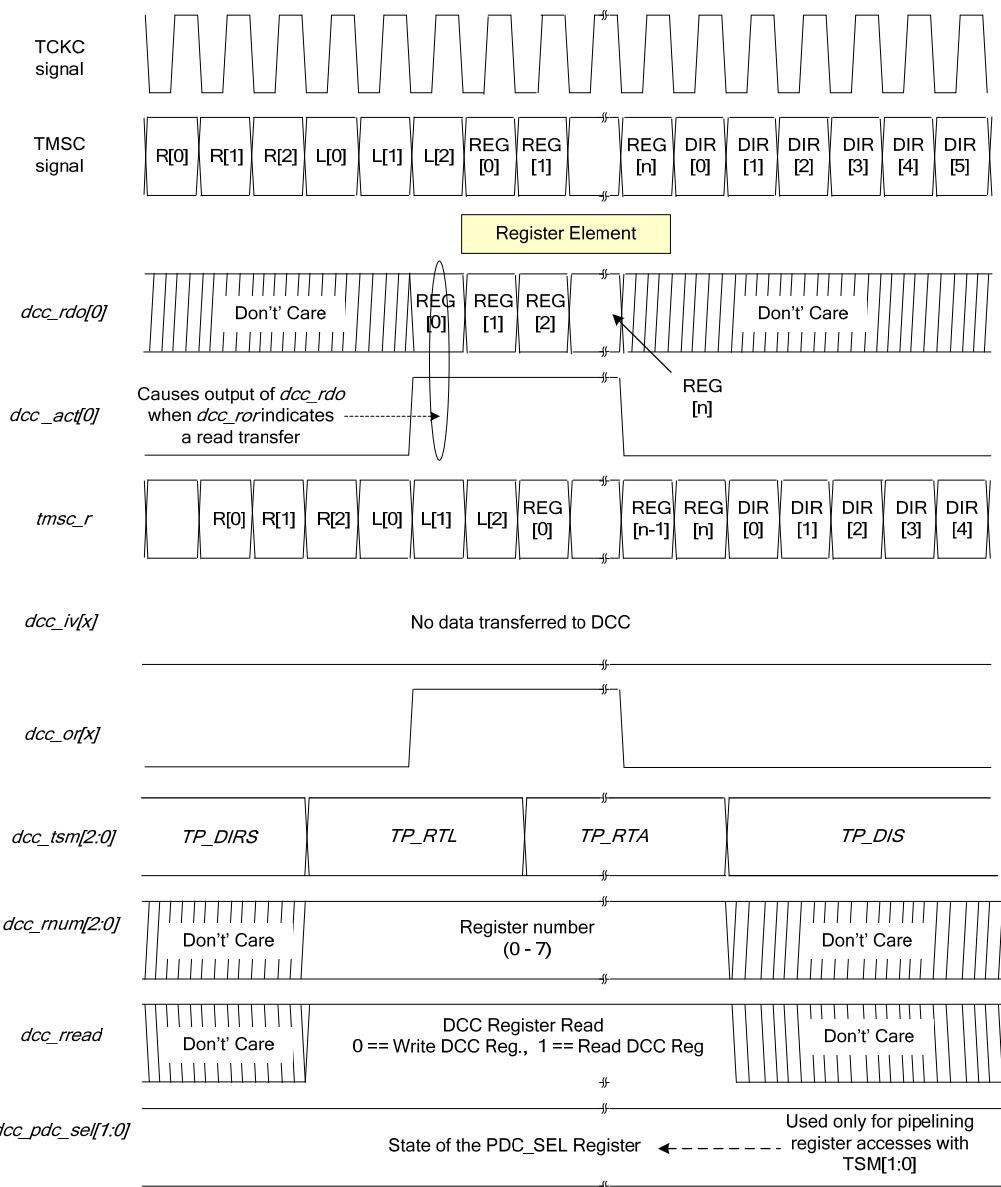
- Each subsequent specification in 28.7.2 shall only apply to a T5 TAP.7.
- The PDCx/DCC interface signaling generated by 12-bit directives other than TP\_CRR and TP\_CRW shall be governed by Figure 28-24.



NOTE—[x] represents the PDC number (x = 0 – 1).

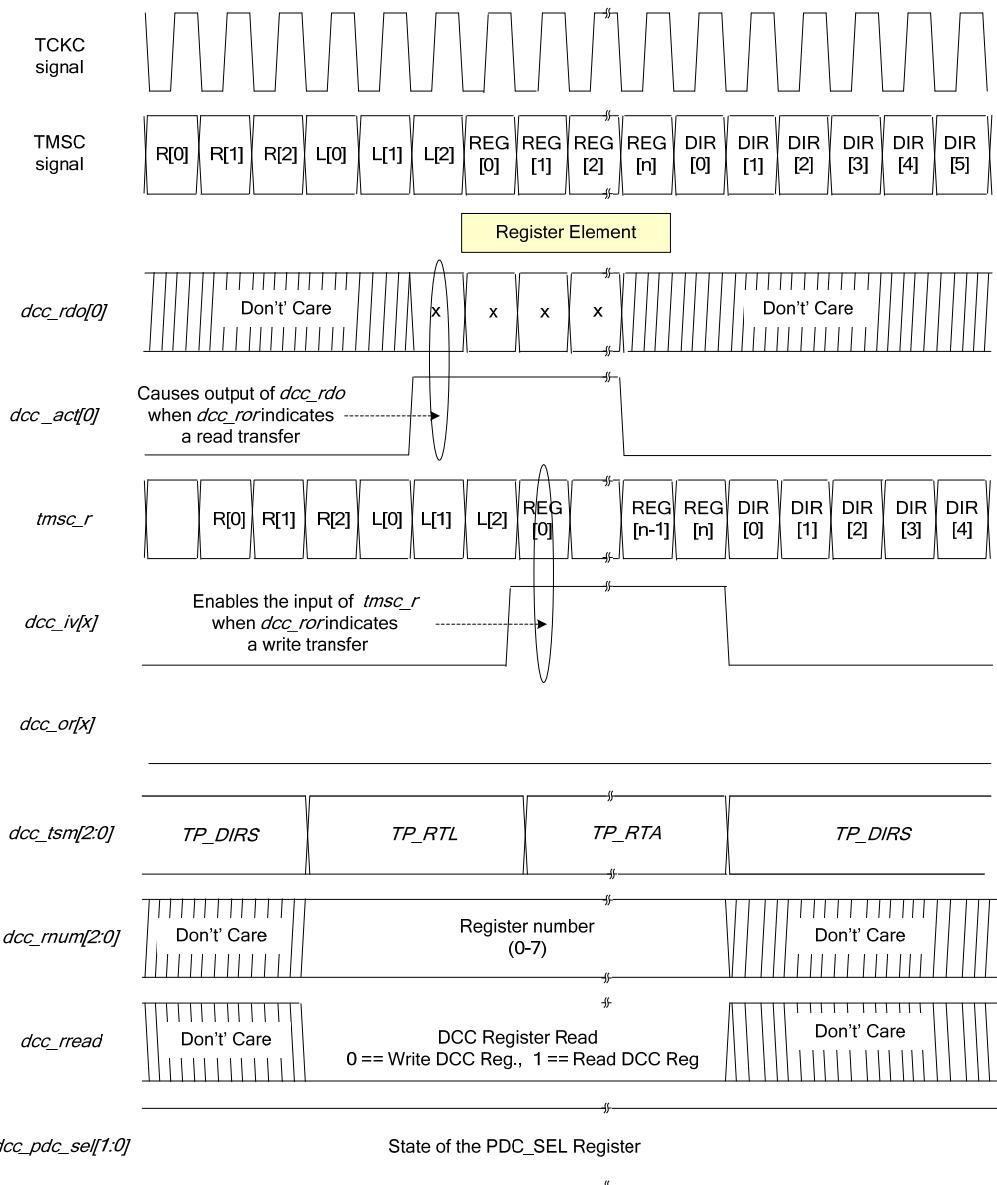
**Figure 28-24 — PDCx/DCC interface signaling/12-bit directives other than TP\_CRR/TP\_CRW**

- c) The PDCx/DCC interface signaling generated by a TP\_CRR Directive shall be governed by Figure 28-25.
- d) The PDCx/DCC interface signaling generated by a TP\_CRW Directive shall be governed by Figure 28-26.



NOTE—The number of bits transferred is governed by the LLL field of the directive described in Table 27-11.

**Figure 28-25 — PDCx/DCC interface signaling with the TP\_CRR Directive**



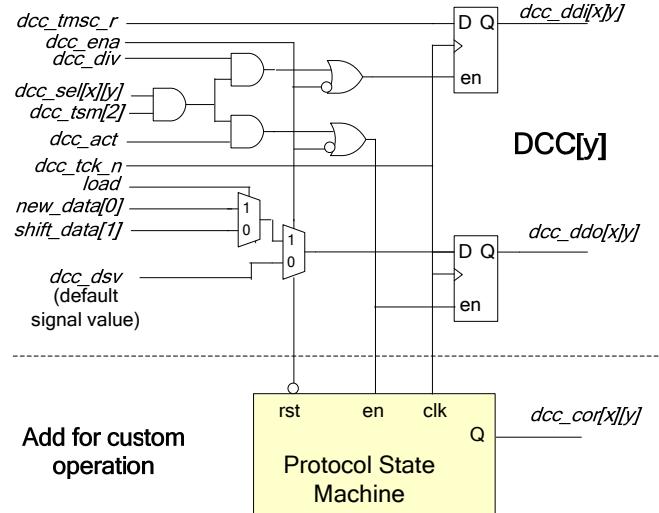
NOTE—The number of bits transferred is governed by the LLL field of the directive described in Table 27-11.

**Figure 28-26 — PDCx/DCC interface signaling with the TP\_CRW Directive**

## 28.8 DCC interface operation

### 28.8.1 Basic capability

It is expected that the DCC operates in a manner similar to that shown in Figure 28-27. Custom operation requires the generation of the *dcc\_cor* signal as shown in this figure.



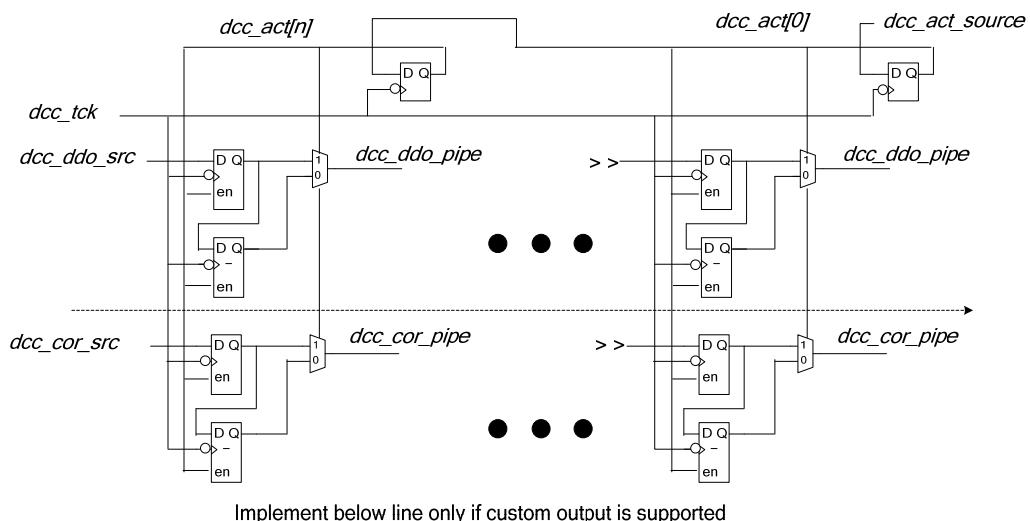
**Figure 28-27 — DCC interface signaling required for Custom Operation**

### 28.8.2 Pipelining register *dcc\_rdo* signaling

When register transfer pipelining is required, it is implemented using a combination of the TSM State Machine state and the *dcc\_pdc\_sel* signals. The TSM state notifies the DCC that a register transfer is about to occur two TCKC periods before the *dcc\_act* signal becomes active. The register number (*dcc\_rnum*), register output request (*dcc\_ror*), and PDC selection information (*dcc\_pdc\_sel*) are all valid at this time. These signals can easily be combined to create the equivalent of an early *dcc\_act* signal for register accesses when this function is needed.

### 28.8.3 Pipelining *dcc\_ddo* and *dcc\_cor* signaling

Any number of repeaters like those shown in Figure 28-28 may be used to pipeline data transfers. The delay of the *dcc\_ddo* and the *dcc\_cor* signal inputs to the TAP.7 Controller does not affect its operation. These delays do, however, cause latency and may affect the operation of the native protocol.

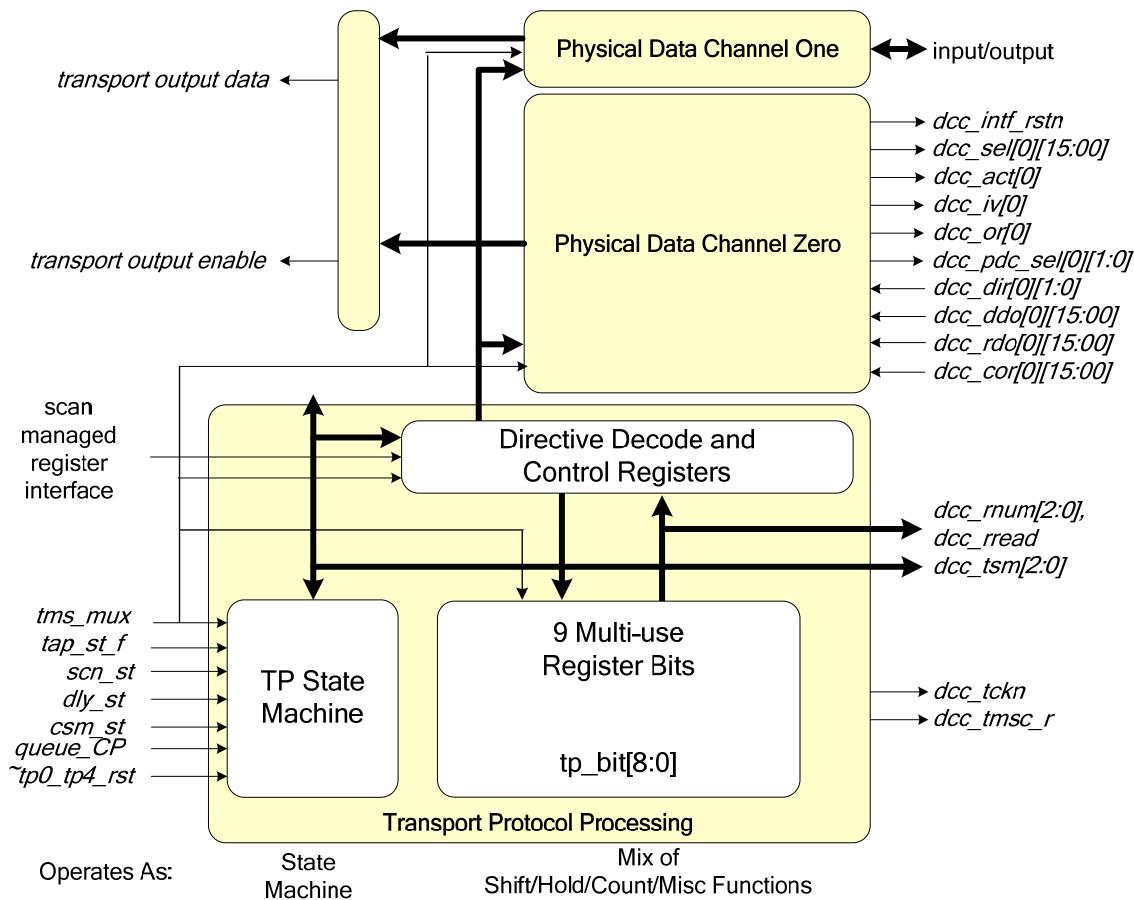


**Figure 28-28 — Pipelining data transfers**

## 28.9 An approach to implementing the Transport Function

### 28.9.1 Overview

A high-level description of the transport functionality is shown in Figure 28-29. This block diagram is an example of how the TP processing may be partitioned and implemented. Other implementation and partitioning options should be considered.



**Figure 28-29 — Conceptual view of the Transport Function**

In this example, the TP Processing Function is constructed with:

- The TSM
- Nine multi-use register bits
- Directive Decode Logic and Transport Control Registers

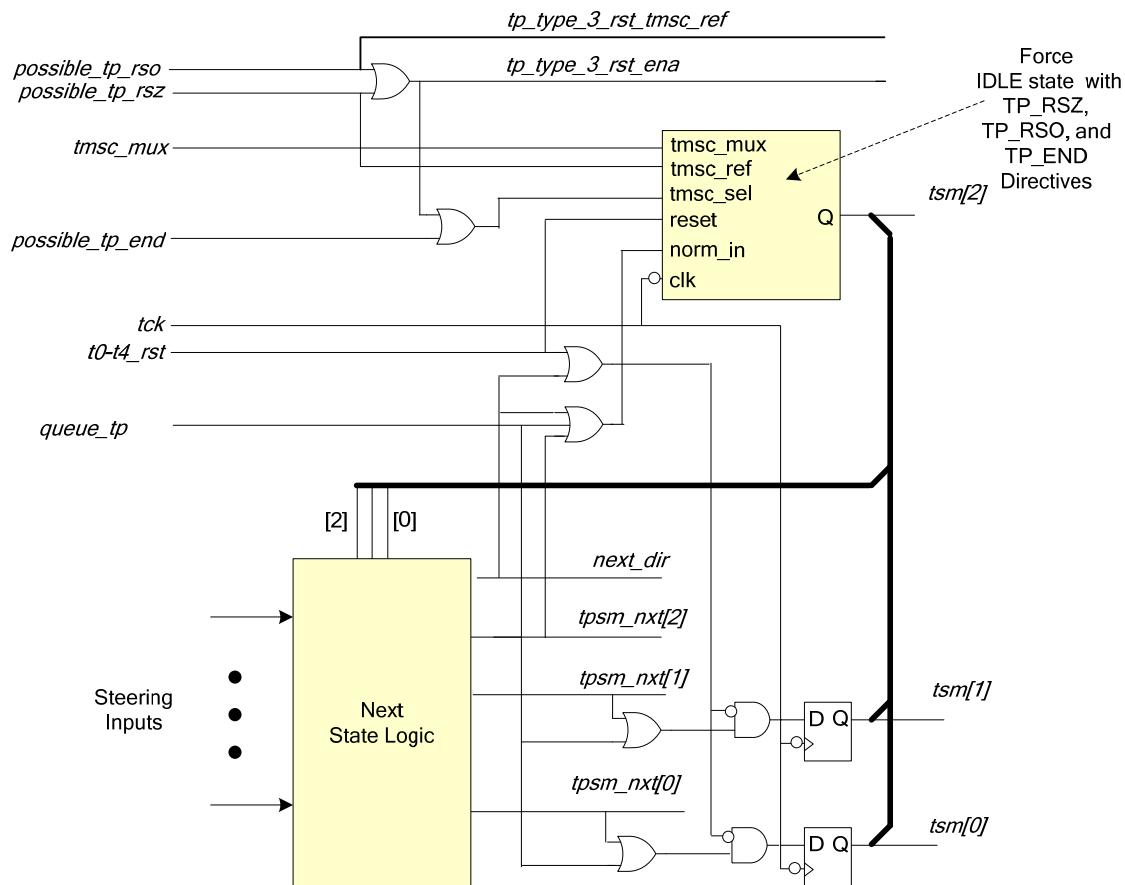
The combination of the TSM state and Multi-use register bits (*tp\_bit[8:0]*) is used in the processing of Directive, Register, and Data Elements. The PDCx/DCC interface *dcc\_rnum[2:0]* and *dcc\_rrread* signals are sourced with the *tp\_bit[8:6]* bits and the *tp\_bit[5]* bit, respectively.

### 28.9.2 The Transport State Machine

One approach to the implementation of the Transport State Machine is shown in Figure 28-30. The use of other approaches should be considered. This approach considers the performance aspects of creating the *DIRS* to *IDLE* state transition using the *tms\_mux* signal. With this state machine:

- The *DIRS* to *IDLE* state transition is handled with logic affecting only the *tsm[2]* signal
- The *next\_dir* signal jams the state to *DIRS*
- The *queue\_tp* signal jams the state to *PEND* when the state is *IDLE*
- Next state transitions other than *IDLE* to *PEND* and *DIRS* to *IDLE* are handled by next state logic

This approach utilizes the *next\_dir* signal to both establish the *DIRS* state and restart directive processing



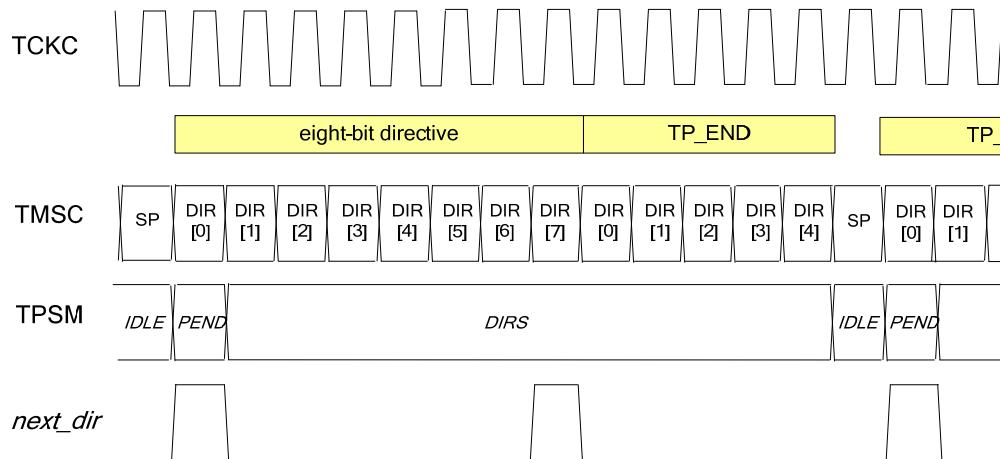
**Figure 28-30 — Example Transport State Machine**

The *next\_dir* signal is asserted when:

- Transport Directive processing is initiated.
- The last bit of the following is registered:
  - An NOP Directive.

- An Eight-bit directive.
- A 12-bit directive other than the TP\_CRR and TP\_CRW Directives.
- A Data Element.
- A Register Element.

Some of these behaviors are shown in Figure 28-31.



**Figure 28-31 — Behavior of the *next\_dir* signal**

### 28.9.3 Multi-use register bits

#### 28.9.3.1 Input configurations

The Multi-use register bits value for each TCK period is produced by one of the input configurations shown in Figure 28-32.

TSM state	Multi-use Register Bit Number								
	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
xxx_to_dirs	A	tms_mux							Load -- 10000000
dirs_to_dirs, dirl to dirl	B	tms_mux							8-bit Shift Right Register
dirs_to_dta	C		Don't care						Load -- 000000
fixed & dta to dta	D		Don't care						6-bit Up Counter
~fixed & dta to dta	E		Don't care	register eot_esc		4-bit Shift Right Register			2-bit Up Counter
dirs_to_dirl	F	tms_mux		4-bit Shift Right Register					Load -- 1000
dirs_to_rtl	G		Hold		Shift >				Load -- 1000
rtl_to_rtl	K		Hold		register ~tmsc_r		4-bit Shift Right Register		
rtl_to_rta	H		Hold			3-bit Shift Right Register			Load -- 00
rta_to_rta	I		Hold					5-bit Up Counter	
others	J								Don't care

**Figure 28-32 — Multi-use register bit input configurations**

The input configuration selected is specified using a combination of the:

- TSM state information
- TSM next state information
- Data Element Length

### 28.9.3.2 Transport Packet processing

#### 28.9.3.2.1 Directive processing

Directive processing starts with a *TP\_PEND* to *TP\_DIRS* TSM state transition when the SP completes. This state transition loads the Multi-use Register with the value shown in Input Configuration A. This occurs coincidently with the registering of the first bit of a Transport Directive. The first bit of the directive is loaded into *tp\_bit[8]*, and *tp\_bit[7:0]* is loaded with the 10000000b value, ensuring all directives have a unique decode value as subsequent bits of the directive are input. Subsequently, Multi-use Register Input Configuration B is used until a five-bit directive is decoded or the first eight bits of another directive are decoded. Either of these conditions causes the next TSM state to be *TP\_DTA*, *TP\_DIRL*, or *TP\_RTL*. These state changes choose Input Configurations C, F, or G, respectively.

#### 28.9.3.2.2 Data Element processing

While processing all bits but the last bit of a Data Element with a fixed format, Multi-use Register Input Configuration D is used. The six LSBs of this register function as an up-counter. The counter value is combined with the DELN Register value to determine the last bit of the Data Element. The DELN Register value used to mask the counter value to determine the last Data Element Bit. An all-ones decode of the masked value of *tp\_bit[5:0]* defines the last bit of the Data Element. The detection of the last bit of the

Data Element causes a *TP\_DTA* to *TP\_DIRS* TSM state transition and the use of Multi-use Register Input Configuration A.

While processing all bits except the last of a Data Element with a fixed format, Multi-use Register Input Configuration E is used. The asynchronous EOT Escape Detection is registered with *tp\_bit[6]* and fed to the input of the four-bit shift register created with *tp\_bit[5:2]*. The shift register input is *tp\_bit[6]* logically ORed with *tp\_bit[5]*, latching the detection of the EOT Escape when it occurs. This both traps and delays the occurrence of an EOT Escape to enable termination of the Data Element at the subsequent nibble boundary. The nibble boundary is defined by the 11b state of the two-bit counter created with *tp\_bit[1:0]*. An all-ones decode of *tp\_bit[5:0]* defines the last bit of the Data Element. The detection of the last bit of the Data Element causes a *TP\_DTA* to *TP\_DIRS* TSM state transition and the use of Multi-use Register Input Configuration A.

### **28.9.3.2.3 12-bit directive processing (other than TP\_CRR and TP\_CRW)**

The first seven bits of all 12-bit directives are processed in the *TP\_DIR* TSM state in the same manner as with other 12-bit directives. A logic 1 reaches *tp\_bit[0]* with bit eight. This causes both the use of Multi-use Register Input Configuration F and a *TP\_DIRS* to *TP\_DIRL* TSM state transition. With this input configuration, the *tp\_bit[8:4]* bits continue to function as a shift register while the *tp\_bit[3:0]* bits are initialized to 1000b to permit the detection of the completion of the directive four bit periods later when *tp\_bit[0]* again becomes a logic 1. The next three bits of these directives are processed using Multi-use Register Input Configuration B, with the Multi-use register bits functioning as a nine-bit shift register. The last bit of the these directives is identified by a logic 1 reaching *tp\_bit[0]* while in the *TP\_DIRL* TSM state. This causes both the use of Multi-use Register Input Configuration A and a *TP\_DIRS* to *TP\_DIRL* TSM state transition.

### **28.9.3.2.4 TP\_CRR and TP\_CRW Directive processing**

The first seven bits of a *TP\_CRR* or *TP\_CRW* Directive are processed in the *TP\_DIR* TSM state in the same manner as with other 12-bit directives, as stated previously. With bit eight, a logic 1 reaching *tp\_bit[0]* causes both the use of Multi-use Register Input Configuration G and a *TP\_DIRS* to *TP\_RTL* TSM state transition. With this input configuration, the *tp\_bit[8:5]* bits are frozen in place with the Register Number and Register Read indication, the *tp\_bit[4]* bit continues to operate as a shift register with *tp\_bit[5]* as its input, and the *tp\_bit[3:0]* bits are initialized to 1000b to permit the detection of the completion of the directive four bit periods later when *tp\_bit[0]* becomes a logic 1.

Subsequently, the next three bits of the directive are processed in the *TP\_RTL* TSM state using Multi-use Register Input Configuration K. This Input Configuration is used in this state until detection of the end of the directive is indicated by a logic 1 *tp\_bit[0]* value. With this input configuration, the *tp\_bit[8:5]* bits are frozen in place, the *tp\_bit[4]* bit is used to register the inverse of the registered value (*tmsc\_r*), and the *tp\_bit[3:0]* functions as a shift register with *tp\_bit[4]* supplying its input. This inputs the directive's nibble count value that is the inverse of the LLL field and right shifts the *tp\_bit[3:0]* value.

The last bit of the these directives is identified by a logic 1 reaching the *tp\_bit[0]* while in the *TP\_RTL* TSM state. This causes both the use of Multi-use Register Input Configuration H and a *TP\_RTL* to *TP\_RTA* TSM state transition. This Input Configuration freezes the *tp\_bit[8:5]* bits in place, completes the input of the LLL field of the directive in the *tp\_bit[4:2]* bits, and creates a 00b value for the *tp\_bit[1:0]* bits. Input Configuration I is used while the *TP\_RTA* TSM state is reached to process all but the last bit of the Register Element transfer. With this input configuration, the *tp\_bit[4:0]* bits function as an up-counter. A count of 11111b defines the last bit of the Register Element and causes both the use of Multi-use Register Input Configuration A and a *TP\_DIRL* to *TP\_DIRS* TSM state transition.

### **28.9.3.3 TSM state/register value relationships**

The Multi-use register bit values for the TSM states are shown in Figure 28-33. This figure conveys the effects of the Input Configurations used to create these values. The effects of TSM state changes on these values are highlighted by describing the value separately for the first TCKC period in a state and subsequent TCKC periods spent in a state.

		Multi-use Register Bit Number														
TPSM State	#	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]						
<i>IDLE</i>	All	Don't care														
<i>PEND</i>	All	Don't care														
<i>DIRS</i>	1st	DIR[0]	10000000b													
	Nth	DIR[next]	6-bit shift right register with tb_bit[6] input													
<i>DIRL</i>	1st	DIR[8]	1000b													
	Nth	DIR[next]	(last tp_bit[8:1])													
<i>RTL</i>	1st	Register Number ( <i>rnum[2:0]</i> )	tb_bit[6]	tb_bit[5]	1000b											
	Nth	Register Number ( <i>rnum[2:0]</i> )	<i>rread</i>	$\sim$ tmsc_r	last tp_bit[4:1]											
<i>RTA/RTE</i>	1st	Register Number ( <i>rnum[2:0]</i> )	<i>rread</i>	32 – number of register element bits												
	Nth	Register Number ( <i>rnum[2:0]</i> )	<i>rread</i>	(last tp_bit[4:0] + 1)												
<i>DTA</i> (fixed length DE)	1st	Don't care	EOT ESC	64 – fixed data element size												
	Nth	Don't care	EOT ESC	(last tp_bit[5:0] + 1)												
<i>DTA</i> (variable length DE)	1st	Don't care	EOT ESC	000000b												
	Nth	Don't care	EOT ESC	last tp_bit[6:3]   (last tb_bit[5:2] & 1000b )	(last tp_bit[1:0] + 1)											

**Figure 28-33 — Multi-use Register values/TSM relationships**

Examples of the relationships of TSM state progressions and the use of Multi-use register bits are provided as follows:

- TP\_CRx Table 28-10
  - TP\_DCx Table 28-11 and Table 28-12

**Table 28-10 — TP\_CRR or TP\_CRW Directive (eight-bit Register Element)**

State	Clk	Value for <i>tp_bit</i> :								
		[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
<i>DIRS</i>	1	DIR[0]	1	0	0	0	0	0	0	0
<i>DIRS</i>	2	DIR[1]	DIR[0]	1	0	0	0	0	0	0
<i>DIRS</i>	3	DIR[2]	DIR[1]	DIR[0]	0	0	0	0	0	0
<i>DIRS</i>	4	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0	0	0	0
<i>DIRS</i>	5	DIR[4]	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0	0	0
<i>DIRS</i>	6	ROR	DIR[4]	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0	0
<i>DIRS</i>	7	R[0]	ROR	DIR[4]	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0
<i>DIRS</i>	8	R[1]	R[0]	ROR	DIR[4]	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1
<i>RTL</i>	1	R[2]	R[1]	R[0]	ROR	DIR[4]	1	0	0	0
<i>RTL</i>	2	R[2]	R[1]	R[0]	ROR	R[2]	DIR[4]	1	0	0
<i>RTL</i>	3	R[2]	R[1]	R[0]	ROR	~C[0]	R[2]	DIR[4]	1	0
<i>RTL</i>	4	R[2]	R[1]	R[0]	ROR	~C[1]	~C[0]	R[2]	DIR[4]	1
<i>RTA</i>	1	R[2]	R[1]	R[0]	ROR	~C[2]	~C[1]	~C[0]	0	0
<i>RTA</i>	2	R[2]	R[1]	R[0]	ROR	~C[2]	~C[1]	~C[0]	0	1
<i>RTA</i>	3	R[2]	R[1]	R[0]	ROR	1	1	0	1	0
<i>RTA</i>	4	R[2]	R[1]	R[0]	ROR	1	1	0	1	1
<i>RTA</i>	5	R[2]	R[1]	R[0]	ROR	1	1	1	0	0
<i>RTA</i>	6	R[2]	R[1]	R[0]	ROR	1	1	1	0	1
<i>RTA</i>	7	R[2]	R[1]	R[0]	ROR	1	1	1	1	0
<i>RTA</i>	8	R[2]	R[1]	R[0]	ROR	1	1	1	1	1
<i>DIRS</i>	1	DIR[0]	1	0	0	0	0	0	0	0

R[2:0]== Register number RRR field of directive (see Table 27-10)

ROR==Source for *dcc\_ror* signal (see Table 27-11—directive bit DIR[5])

C[2:0]==Nibble count—CCC field of directive (see Table 27-10)

**Table 28-11 — TP\_DCx Directive – variable-length transfer (eight-bit Data Element)**

State	Clk	Value for <i>tp_bit</i> :								
		[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
<i>DIRS</i>	1	DIR[0]	1	0	0	0	0	0	0	0
<i>DIRS</i>	2	DIR[1]	DIR[0]	1	0	0	0	0	0	0
<i>DIRS</i>	3	DIR[2]	DIR[1]	DIR[0]	0	0	0	0	0	0
<i>DIRS</i>	4	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0	0	0	0
<i>DIRS</i>	5	DIR[4]	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0	0	0
<i>DTA</i>	1	X	X	EOT(0)	0	0	0	0	0	0
<i>DTA</i>	2	X	X	EOT(1)	0	0	0	0	0	1
<i>DTA</i>	3	X	X	EOT(x)	1	0	0	0	1	0
<i>DTA</i>	4	X	X	EOT(x)	1	1	0	0	1	1
<i>DTA</i>	5	X	X	EOT(x)	1	1	1	0	0	0
<i>DTA</i>	6	X	X	EOT(x)	1	1	1	1	0	1
<i>DTA</i>	7	X	X	EOT(x)	1	1	1	1	1	0
<i>DTA</i>	7	X	X	EOT(x)	1	1	1	1	1	1
<i>DIRS</i>	1	DIR[0]	1	0	0	0	0	0	0	0



**EOT Escape occurs in this bit period**

DIN==Data Element data

**Table 28-12 — TP\_DCx Directive – fixed-length transfer (eight-bit Data Element)**

State	Clk	Value for tp_bit:								
		[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
<i>DIRS</i>	1	DIR[0]	1	0	0	0	0	0	0	0
<i>DIRS</i>	2	DIR[1]	DIR[0]	1	0	0	0	0	0	0
<i>DIRS</i>	3	DIR[2]	DIR[1]	DIR[0]	0	0	0	0	0	0
<i>DIRS</i>	4	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0	0	0	0
<i>DIRS</i>	5	DIR[4]	DIR[3]	DIR[2]	DIR[1]	DIR[0]	1	0	0	0
<i>DTA</i>	1	x	x	ESC	0	0	0	0	0	0
<i>DTA</i>	2	x	x	ESC	0	0	0	0	0	1
<i>DTA</i>	3	x	x]	ESC	0	0	0		1	0
<i>DTA</i>	4	x	x	ESC	0	0	0	0	1	1
<i>DTA</i>	5	x	x	ESC	0	0	0	1	0	0
<i>DTA</i>	6	x	x	ESC	0	0	0	1	0	0
<i>DTA</i>	7	x	x	ESC	0	0	0	1	0	0
<i>DTA</i>	7	x	x	ESC	0	0	0	1	1	1
<i>DIRS</i>	1	DIR[0]	1	0	0	0	0	0	0	0

The last bit of the Data Element is detected with *tp\_bit[5:0]* masked appropriately based on the value of the DELN Register.

#### 28.9.3.4 Transport output and DCR input selection

The creation of transport output and input from a DCR is shown in Figure 28-34.

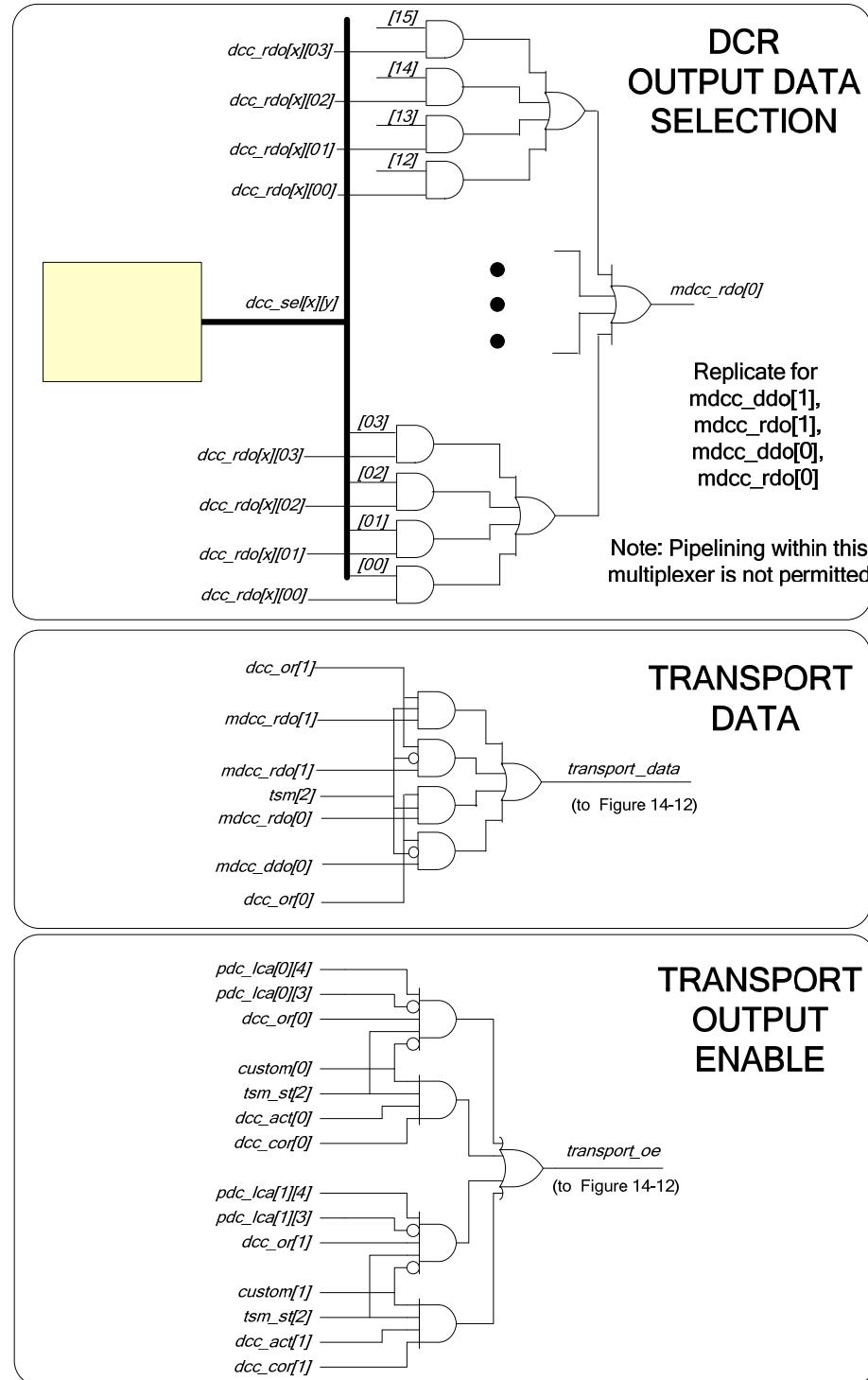


Figure 28-34 — Transport drive and DCR input selection

### 28.9.3.5 TP activity examples

Examples of transport activity are shown in the following figures:

- TP with fixed data transfer with nonpipelined scan operations      Figure 28-35
- TP with fixed data transfer with pipelined scan operations      Figure 28-36
- TP with continuous data transfer      Figure 28-37
- TP with no data transferred      Figure 28-38

The use of a fixed-length Data Element is illustrated in Figure 28-35 and Figure 28-36, while the use of a variable-length Data Element is illustrated in Figure 28-37.

The length of the Data Element affects the responsiveness of scan operations. This can be seen in Figure 28-35 and Figure 28-36. In these figures, the time between Scan Packets is governed by the length of the Data Element. With a fixed-length data Payload Element, the responsiveness of scan operations is inversely proportional to the length of the fixed-length Data Element when a Data Element is being transferred. With a variable-length Data Element, a variable-length Data Segment may be terminated in approximately 12 Test Clock periods.

The effects of designating a TAPC state as either not supporting or supporting transport are illustrated in Figure 28-37 and Figure 28-38. In Figure 28-37, the *Run-Test/Idle* TAPC state has not been designated as supporting transport, while in Figure 28-38, the *Run-Test/Idle* state has been designated as supporting transport.

For a TAPC state supporting transport, the SP preceding a TP advances the real TAPC state either before or after the TP begins, depending on the scan format used. This is illustrated in Figure 28-35 and Figure 28-36.

In the following examples:

- S == TAPC states designated as supporting transport
- NS == TAPC states designated as not supporting transport or not designated as supporting transport

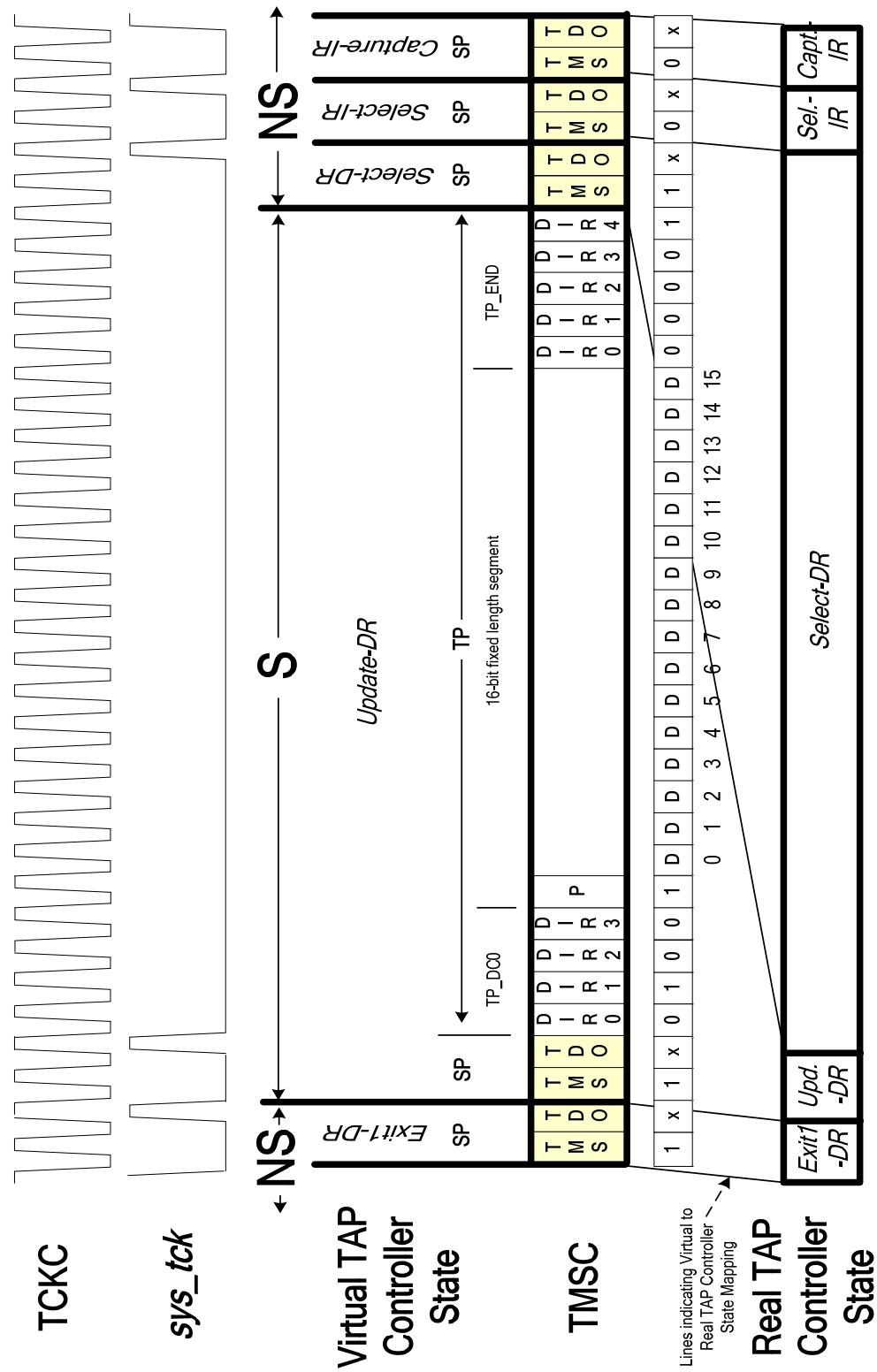


Figure 28-35 — TP with fixed-length data transfer with nonpipelined scan operations

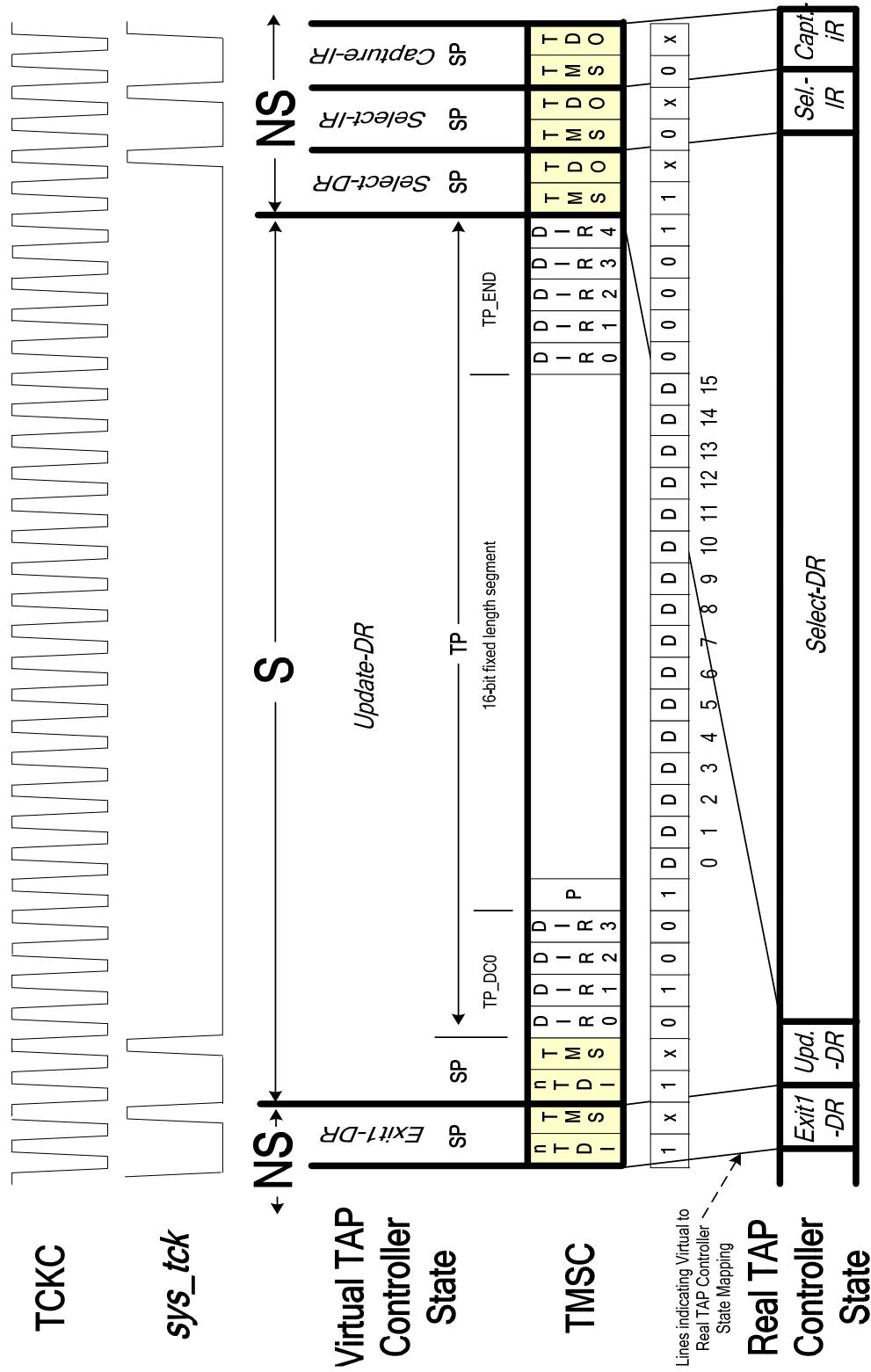


Figure 28-36 — TP with fixed-length data transfer with pipelined scan operations

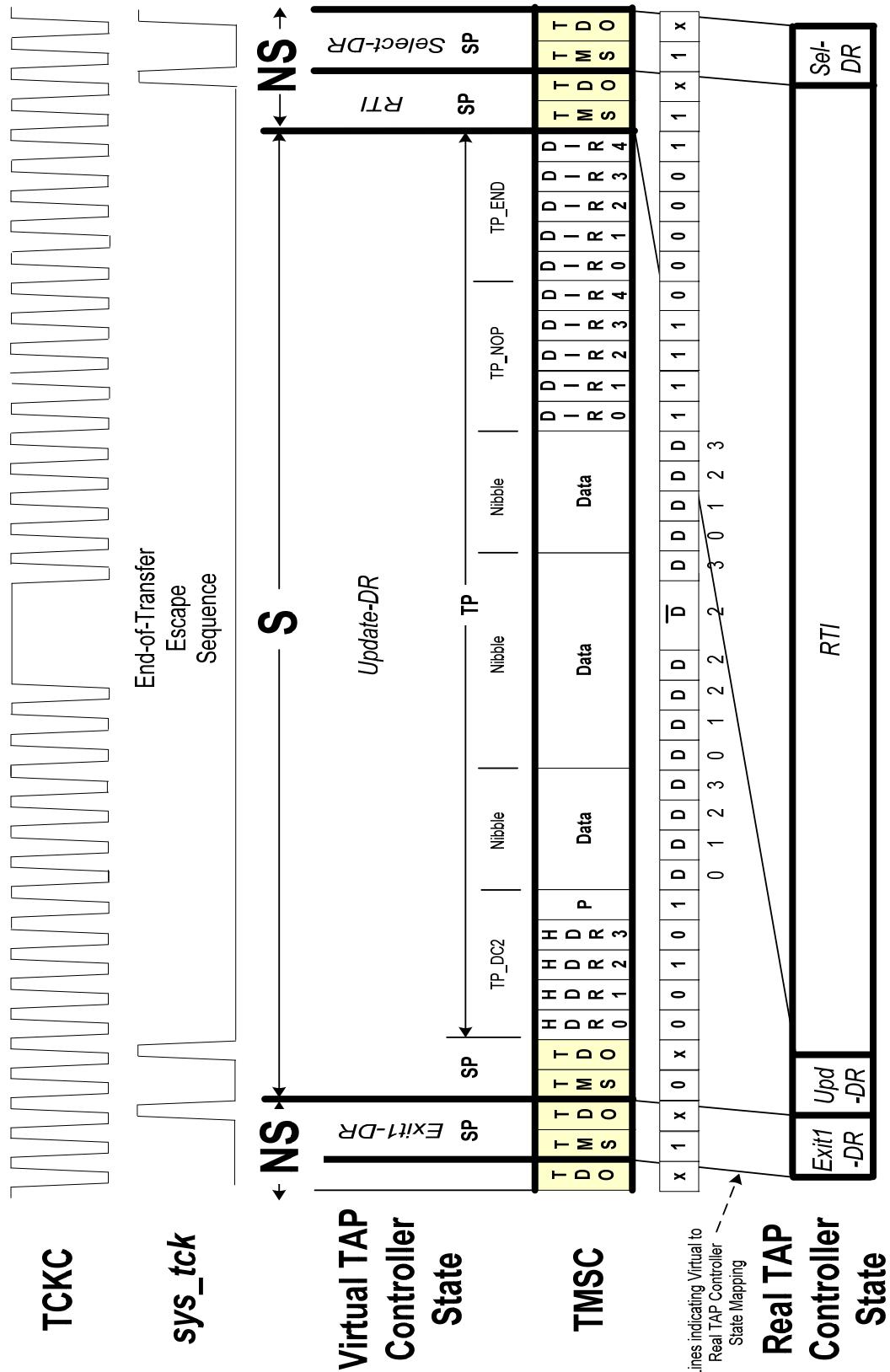


Figure 28-37 — TP with continuous data transfer

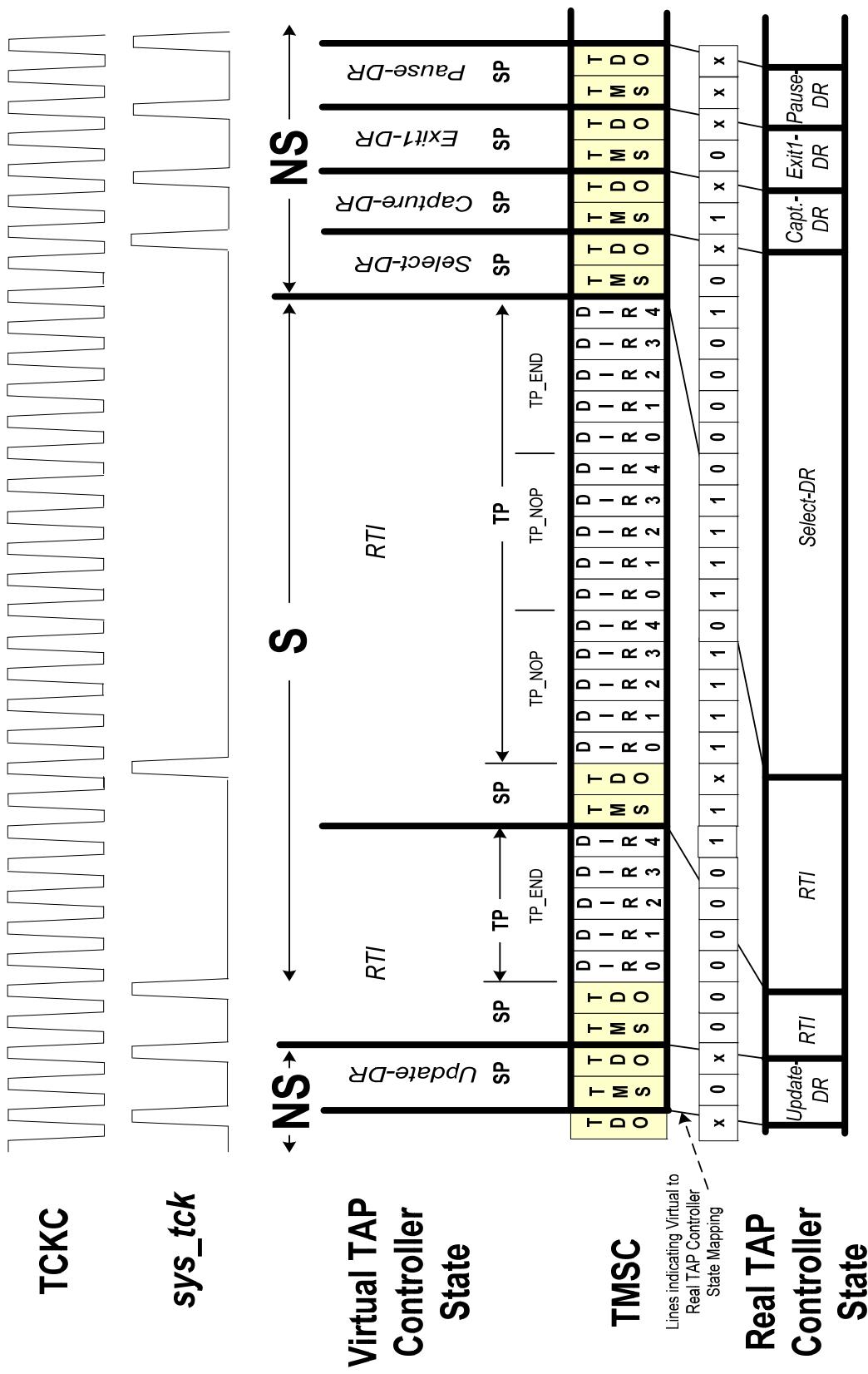


Figure 28-38 — TP with no data transferred

## 29. Test concepts

### 29.1 Introduction

Notwithstanding the fact that the enhanced functionality provided by this standard is largely focused on enabling debug applications through the TAP, note that this cannot be taken to imply any intent to abandon test applications (or other related/similar applications, such as device programming/in-system configuration). In fact, as has been described in other clauses, particularly Clause 4, Clause 5, Clause 7, Clause 8, Clause 15, and Clause 16, not only does this standard fully honor test (and related/similar) applications of the TAP, but also it builds its architecture of enhanced functionality on top of a foundation of capability that is fully compatible with such applications.

This clause provides additional background on the needs of test applications and further description as to how these needs are met within the IEEE 1149.7 paradigm. It describes the use of the various board-level and system-level scan topologies encountered where some devices in such boards and systems implement this standard. This clause, together with Annex F, which deals with the electrical considerations of boards and systems, is intended to enable the deployment of boards and systems that both perform satisfactorily with respect to electrical characteristics and can be utilized by test tools as well as for debug.

Most aspects of supporting documentation for such boards and systems are described in summary within this clause. Although some of this documentation may be represented by means that are particular to specific tool implementations, the concepts involving such documentation are described.

Detailed mandatory requirements for documentation of IEEE 1149.7 components are specified in Clause 30 and Clause 31. The subjects described in this clause are covered in the following order:

- 29.2 Interoperability
- 29.3 Construction of the unit under test
- 29.4 Background (IEEE 1149.1 paradigm)
- 29.5 Implications for test applications arising from this standard
- 29.6 Test example—a narrative
- 29.7 Describing the unit under test
- 29.8 Documentation model
- 29.9 Considerations for large-system applications

### 29.2 Interoperability

With regard to test and related/similar applications, a primary motivation/key objective of this standard is to maintain interoperability with IEEE Std 1149.1 to preserve the industry's test infrastructure. This objective is met by providing:

- Access to the standard boundary-scan architecture as specified by IEEE Std 1149.1 (see 1.8).
- A means to effect simultaneous stimulus update and simultaneous response capture even where the various components involved in such activities are involved in Star Scan Topologies (see 4.2.7.5 and 5.5.3.4).

While debug (particularly applications debug) is investigative in nature (meaning that the stimulus may be *ad hoc* and the expected response is generally unknown in some key aspect), test and related/similar applications are deterministic (meaning that both stimulus and expected response are fully defined *a priori*).

Even with much of the enhanced functionality of this standard supporting debug applications, these capabilities have been implemented so as to fully support and benefit test applications to the fullest extent possible. For example, test applications may benefit generally in that it may be easier to deploy certain multi-chip architectures, such as system-in-package (SiP) and/or package-on-package (PoP), with the Star Scan Topology enabled by this standard (for T3 and above TAPs) versus the Series Scan Topology to which IEEE Std 1149.1 is typically limited.

In addition to the hardware specified in the prior clauses, this standard also fully supports a means of describing the component-specific information required by test applications, as codified in the Boundary-Scan Description Language defined by this standard (BSDL.7) and Hierarchical Scan Description Language defined by this standard (HSDL.7), as specified in the subsequent clauses. These clauses provide a well-defined approach to meeting the information needs of test applications. The suppliers of components that implement this standard would do well to utilize these facilities to enable test applications.

### **29.3 Construction of the unit under test**

The item of particular interest to this clause will be referred to as the unit under test (UUT).

As pertains to this clause, a unit under test is constructed from a collection of components with defined interconnection. These components may be arranged on a common substrate, as in the case of board-level applications, or they may be arranged on several distinct substrates, which is generally referred to as system-level applications.

A component is defined as:

- A compound component when it is an assembly constructed from a collection of subordinate components
- A simple component when it has no subordinate components; a simple component is also referred to simply as a device

With this definition, a component may be any of the following:

- A board [e.g., a Dual In-line Memory Module (DIMM) or other board subassembly]
- A package [an MCM (Multi-chip Module), an SoC (System-on-a-chip)]
- A chip (direct attach dies)

A component is further classified as:

- A test module when it has more than one TAPC exposed to test view (an MCM, an SoC with multiple EMTAPCs), as will be the case for most (but not all) compound components
- A test endpoint when it has only one TAPC exposed to test view (e.g., an encapsulated IC), as will be the case for simple components

A test module may have more than one TAPC exposed to test at its boundary.

### **29.4 Background (IEEE 1149.1 paradigm)**

Many aspects of board-level and system-level applications within the IEEE 1149.7 paradigm are relatively novel when compared to the IEEE 1149.1 paradigm. To establish the contrast (divergence), some background in conventional aspects of board-level and system-level applications is described in the following subclauses. The two aspects of primary concern are the scan topology and scan-state sequencing.

### 29.4.1 Topology

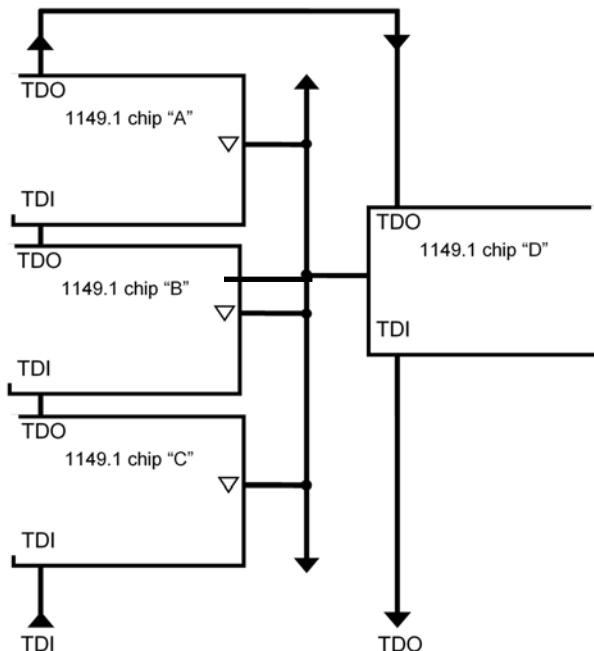
The only topology expressly elaborated by IEEE Std 1149.1 is the Series Scan Topology (four wires; for example, see Figure 31-1, as modeled after an IEEE 1149.1 serial connection using one TMS signal). Of course, within a given board or system unit, consideration is given to the case where multiple Series Scan Topologies (distinct chains) are deployed.

In fact, IEEE Std 1149.1 also mentions the possibility of deploying multiple series topologies on a single unit where the TDI wires and the TDO wires are shared among the chains in “star” fashion (that is, a common wire transmits the TDI signal to all chains and a common wire transmits the TDO signal from all chains). Of course, where these serial data signals are shared in such fashion, some means is provided to effectively select a particular chain of interest—such selection is made either by means of distinct TCK or, more commonly, by means of distinct TMS (for example, see Figure 31-2, as modeled after an IEEE 1149.1 Connection in two paralleled serial chains). While such topologies often are described with the term “star” due to common routing of all but the distinct control, they are not true “star” topologies as this standard would describe them and are not described here in that fashion as they lack built-in selectability.

NOTE—This standard describes the use of distinct TCK such that otherwise incompatible topologies can be provided with some common wiring (see Figure 7-1).

### 29.4.2 Scan-state sequencing

In order to ascertain scan-state sequencing requirements of the typical test application, an example for testing of interconnections among components of a UUT are considered, as illustrated in Figure 29-1 (as modeled after IEEE 1149.1 testing board-level test lines).



**Figure 29-1 — Typical test application – testing component interconnections**

Observing that the output signals at components A, B, and C are three-state outputs, the necessity of coordinating the application of the output enable stimuli is apparent; if it were otherwise, a contention could be present on the wired junction. Furthermore, while not strictly necessary in this most simple case,

the utility of coordinating the capture of the response at all input signals is presumed; at the very least, all response captures initiated by a given stimulus are sampled prior to a change in the applied stimuli (which constraint is best met by avoiding any update until all required captures are completed and which further is most efficient if all responses are captured in unison).

Additionally, where the case of the testing of advanced digital networks as per IEEE Std 1149.6 [B2] is considered, it is noted that all devices participating in such testing are sequenced in unison through the *Run-Test/Idle* state, where the alternating current (AC) stimuli are generated, and thence as quickly as possible through the *Capture-DR* state, where the AC responses are sampled (where all components participating in the test are IEEE Std 1149.6 [B2] compliant, this later requirement can be omitted, but it is critical where some components may not be IEEE Std 1149.6 [B2] compliant).

As such, conventional scan sequencing for test applications, presuming the Series Scan Topology, involves all of the following key events:

- Shift (serial), wherein capture responses are exported and new stimuli are imported (i.e., scan access to the Series Scan Topology in *Shift-DR*)
- Simultaneous/parallel update, wherein new stimuli are applied, at a given instant, across all device signals to which the test operation pertains (i.e., broadside update of the Series Scan Topology in *Update-DR*)
- Simultaneous/concurrent run test, wherein transient stimuli may be generated, in coordination with *Run-Test/Idle*, across all device signals to which the test operation pertains (i.e., concurrent run-test operation of the Series Scan Topology in *Run-Test/Idle*)
- Simultaneous/parallel capture, wherein new capture responses are sampled, at a given instant, across all device signals to which the test operations pertains (i.e., broadside capture of the Series Scan Topology in *Capture-DR*)

## 29.5 Implications for test applications arising from this standard

### 29.5.1 Divergences versus IEEE Std 1149.1

Two significant areas of divergence versus IEEE Std 1149.1 arise from this standard with regard to test applications:

- Topology (i.e., star versus series)
- Scan-state sequencing (i.e., star versus series)

Clearly, the most obvious topological divergence versus IEEE Std 1149.1 is the introduction of the IEEE 1149.7 Star Scan Topologies that involve star connections on all TAP.7 signals (whether Star-4 or Star-2) versus the Series Scan Topology typical of IEEE 1149.1-based systems. As contrasted versus the Series Scan Topology, the Star Scan Topologies do not facilitate test applications according to the conventional scan-state sequencing as described in 29.4.2.

A further divergence, perhaps not so obvious, is the introduction of hierarchy throughout IEEE 1149.7-based systems. As described in Clause 8, IEEE Std 1149.7 provides for the following layers of hierarchy:

- Technologies (i.e., IEEE Std 1149.7 versus other technologies that may share TAP.7 signaling)
- Topologies (i.e., Star-2, Star-4, or Series that may share TAP.7 signaling)
- TAP.7s (i.e., multiple independent components that may share TAP.7 signaling)
- TAPCs (i.e., multiple scan targets that reside behind a TAP.7 on a given component)

A given test application navigates these layers using the selection mechanisms described in Clause 8 in order to access TAPCs at various levels of the hierarchy. Once the items that pertain to a given test application have been selected, scan-state sequencing proceeds in a manner appropriate to such test application; this requires the accommodations identified in 29.5.2.

### 29.5.2 Accommodation/resolution of divergences versus IEEE Std 1149.1

Test use of the selection mechanisms defined by this standard is detailed in 8.10.2. Additional detail on the use of SSDs to create series scan equivalency is provided in 4.2.7.5 and 5.5.3.4.

A further consideration for test applications in the context of this document arises as concerns the power safe operation (avoiding inadvertent power-down of items required to support the test application).

## 29.6 Test example—a narrative

The following is a narrative example for a typical test application:

- Establish an electrical connection between the DTS and the UUT.
- Power-up the UUT.

NOTE 1—For some test applications, as for many (if not most) debug applications, the power-up step might precede the connection step.

- If the UUT type and configuration are known, continue to the next step, else (UUT type and configuration are unknown), attempt target identification (such as scanning a barcode or an automated interrogation of scan topology/path configuration).

NOTE 2—The automated interrogation of scan topology/path configuration of an unknown target where IEEE 1149.7 devices may be present as described in Annex D is complex and is not recommended for test applications in the most general case.

- If the UUT type and configuration are known, continue to the next step, else (UUT type and configuration cannot be determined), abort.
- As necessary:
  - Use the Standard Protocol to power all TAP.7s (move to *Run-Test/Idle* & wait ~100 milliseconds).
  - Place Online TAPCs that are Offline-at-Start-up as described in 11.9.7.
  - Establish the Scan Topology associated with the TAPCs as described in 20.11.

NOTE 3—The start-up sequence to power-up, place all TAPCs Online, and establish the Scan Topology associated with TAPCs is described in Annex D.

- For Star-2 Topologies, use the Advanced Protocol (MScan), all devices become directly accessible after soft reset (only required where two-wire chips power-up Offline); set always powered, set sampling to falling edge for speed-up where desired and timing-compatible, clear and apply all scan selections (decoupling all STLs), set OScan1, and de-allocate all CIDs, then CIDs can be pushed to up to 16 devices of interest using Directed CID Allocation, if desired (if more than 16 devices of interest, the remainder can be accessed by means of long-form addressing); now, enable SSD and globally select all devices (SSD); then use the method described in 8.10.2 to issue scans as required.
- For Star-4 Topologies, all devices become directly accessible after all TAP.7s are powered; set always powered, clear and apply all scan selections (decoupling all STLs), set JScan3, and de-allocate all CIDs, then CIDs can be pushed to up to 16 devices of interest as with the Star-2 Scan Topology, if desired (if more than 16 devices of interest, the remainder can be accessed by

- means of long-form addressing); now, enable SSD and globally select all devices (SSD); then use the method described in 8.10.2 to issue scans as required.
- For Series Topologies, all devices become serially accessible after all TAP.7s are powered; set always powered, clear and apply all scan selections (decoupling all STLs), set JScan0 or JScan2, enable SSD and globally select all devices (SSD); now, use “legacy” scans as required.

NOTE 4—For Star Topologies, scans that are intended to affect multiple TAPs associated with a single branch are scanned sequentially, with SSDs used to select the TAP of interest. When the scan topology has multiple branches, the branches are first selected and then scanned sequentially. Once a branch is selected, the TAP within Star Branches are selected and scanned sequentially. Branches are selected using the branch selection mechanism described previously in Clause 11, while TAPCs are selected using SSDs as described in Clause 20. These scans are performed without reaching the *Update-DR* state. Once these scans are completed, all branches are selected and the update and capture operations are activated in all branches at once with the TAPC state moved to the *Pause-DR* state where the selection of a branch and the TAPC of interest begins anew (see 8.10 for selection use cases).

- Apply scan path verification (*BYPASS, IDCODE, PRELOAD, SAMPLE*).
- Apply conventional boundary-scan external test (IEEE 1149.1: *PRELOAD, SAMPLE, EXTEST, CLAMP, HIGHZ*).
- Apply conventional sequences for boundary-scan configuration of programmable devices (IEEE 1149.1: *PRELOAD, SAMPLE, EXTEST, CLAMP, HIGHZ*).
- Apply arbitrary scan sequences per Serial Vector Format and/or Standard Test and Programming Language (or other vector formats).
- Apply conventional boundary-scan internal test (IEEE 1149.1: *INTEST*).
- Apply conventional built-in self-test (IEEE 1149.1: *RUNBIST*).
- Apply boundary-scan test for mixed-signal environments (IEEE Std 1149.4 [B1]: *PRELOAD, SAMPLE, EXTEST, PROBE, CLAMP, HIGHZ*).
- Apply boundary-scan test for advanced digital networks (IEEE Std 1149.6 [B2]: *PRELOAD, SAMPLE, EXTEST\_PULSE, EXTEST\_TRAIN, CLAMP, HIGHZ*).
- Apply scan sequences for in-system configuration of programmable devices (IEEE Std 1532 [B4]: *ISC\_ENABLE, ISC\_PROGRAM, ISC\_NOP, ISC\_DISABLE*).
- Apply scan sequences for JTAG-accessible on-chip instrumentation (e.g., IEEE Std 1687 [B5]).
- Power-down the UUT.
- Terminate the electrical connection between the DTS and the UUT.

NOTE 5—For some test applications, as for many (if not most) debug applications, the power-down step might be last and/or deferred indefinitely.

## 29.7 Describing the unit under test

A unit under test may contain:

- One or more series scan chains
- One or more scan chains in Star Scan Topology (wide and/or narrow star)
- A mix of Series and Star Scan Topologies

The generalized form of a unit under test’s scan architecture is shown in Figure 7-1 through Figure 7-3. It is possible, however, for a unit under test to have any combination of the scan topologies shown in these figures. It is the responsibility of the test generation tool to analyze the scan topology based on the netlist of the unit under test and the HSDL of test module HSDLS.

When all of the scan topologies are a Series Scan Topology, the TAPCs are naturally addressable based on their position on the scan chain. When all or part of the scan topologies uses TAP.7s in a Star Scan Topology, such natural addressability of TAPCs is absent. In this case, an HSDL is required to communicate the addressability of each module or endpoint within the Star Scan Topology.

With a TAP.7, the TAPC addressability needed to support operation within the Star Scan Topology is provided by the TAPC Address (TCA) described in Clause 20. This TCA makes each TAP.7 unique, even for otherwise identical devices. Recall the TCA is created from 27 bits of the Device Identification Code required for an ADTAPC and the 8-bit Node Identification Code that is used to make the TCA unique. The 27 bits of the Device Identification Code are identified by the BSDL describing the endpoint containing the TAPC (namely, DEVICE\_ID[27:01] as specified by IEEE Std 1149.1). The 8-bit NODE\_ID is identified in the required HSDL above the BSDL in the connectivity information hierarchy. It is the responsibility of the system designer to ensure each TAP.7 TAPC has a unique TCA.

## 29.8 Documentation model

Some documentation is required for test and configuration to ensure that the user of an integrated circuit or similar device can perform the basic functions associated with the test logic defined by IEEE Std 1149.1 and its derivatives. Some examples of these basic functions are examining the operation of a prototype system, testing assembled products for manufacturing defects, configuring programmable logic and nonvolatile memory, etc.

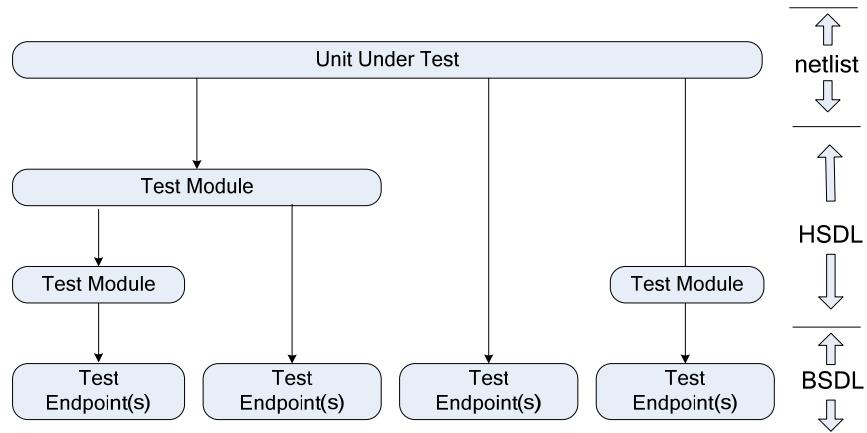
Within the context of devices claiming to provide IEEE 1149.7-Specified Behavior, the documentation required for such board- and system-level applications is expected to involve the following aspects:

- The device claiming to provide IEEE 1149.7-Specified Behavior needs to be supplied along with scan test description file(s) to include IEEE 1149.7 BSDL(s) (which provides for the description of a single “Chip-Level” TAPC exposed-to-test view) and, where applicable, IEEE 1149.7 HSDL (which provides for the description of a module that includes one or more “Chip-Level” TAPCs exposed-to-test view). Such scan-test description file(s) are typically expected to be provided by the device vendor.
- The board or system that contains IEEE 1149.7 devices configured in a Star Scan Topology needs to be supplied along with a node ID list that includes the node ID for each TAP.7 TAPC in a Star Scan Topology. Of course, where a given board or system contains multiple instances of the same device type, each such device instance is assigned a unique node ID. Such a node ID list is typically expected to be provided by the board/system designer.
- The board or system needs also to be supplied with an associated netlist, that is, a description of the devices placed on the unit and the interconnections among the device pins on the unit. Such a netlist is typically expected to be provided by the board/system designer. In a typical test generation flow, this information is analyzed by test tools to extract the scan topology for the unit. Such test tools are typically expected to be supplied by a tool vendor.

Each component within the system is described by means of HSDL if it is a test module or by means of BSDL if it is a test endpoint. In the case where the pins on a test endpoint are physically remapped, e.g., where a BSDL is provided for a die that is subsequently repackaged, it is permitted to provide HSDL for this purpose.

A component’s BSDL may be IEEE 1149.1 BSDL (for components that provide only IEEE 1149.1-Specified Behavior) or IEEE 1149.7 BSDL (for components that provide IEEE 1149.7-Specified Behavior).

An example unit under test is shown in Figure 29-2. A unit under test has at a minimum one endpoint and may have other endpoints and test modules.



## **Figure 29-2 — Unit-under-test description**

The BSDL (for a test endpoint providing IEEE 1149.1-Specified Behavior or IEEE 1149.7-Specified Behavior, respectively) includes the following:

- The signal names of the TAP (TAP.1 or TAP.7, respectively)
  - The signal names of the inputs and outputs of the system logic
  - The association by bit position of the boundary-scan register to the inputs and outputs of the system logic
  - Other miscellaneous information

The HSDL (for a test module providing IEEE 1149.7-Specified Behavior) includes the following:

- The signal names of the TAP.7(s)
  - The signal names of the inputs and outputs of the test module
  - A list of subcomponents (test endpoints and hierarchically lower test modules)
  - A list of node IDs for subcomponents requiring such (test endpoints with a T3 and above TAP.7)
  - Component-to-component signal connectivity related to the scan path(s)
  - Component-to-component signal connectivity related to functional uses (optional)
  - Other miscellaneous information

With the combination of HSDL and BSDL information listed here the test software can extract the scan chain topology and other connections needed for interconnect test generation using the component HSDLS and BSDLs. The inclusion of component-to-component functional connections in the HSDL provides for the testing of these connections where possible.

## 29.9 Considerations for large-system applications

Typical debug applications involve a small system or utilize only a portion of a large system (giving it the appearance of a small system). Many test applications, to the contrary, involve large systems that span large printed circuit boards and/or collections of many such printed circuit boards. One can get a sense of the possible scope of a test application by imagining that each box (excepting the DTS boxes) in Figure 7-3 is hierarchical in the sense that each such box could be considered to contain a wholesale instance of one of the topologies illustrated in the figure (and so on).

With this in mind, Table 29-1 presents some considerations for large-system applications as contrasted against small-system applications.

**Table 29-1 — Contrasting large-system applications against small-system applications**

Small-system applications	Large-system applications
Just a few IEEE 1149.7 components	Great numbers of IEEE 1149.7 components
Most likely on a single substrate	Generally spread across many substrates (e.g., boards in a rack)
Reduced pin count TAP highly valued – limited real estate for access to TAP signals	Reduced pin count TAP likely not highly valued – plenty of real estate for access to TAP signals
Star-2 Scan Topology provides adequate connectivity	Star-2 Scan Topology may not provide adequate connectivity throughout the system due to: <ul style="list-style-type: none"><li>— Logical constraints (too many devices of the same type on a given branch)</li><li>— Electrical constraints (too many loads on a given branch, especially for TMSC; transmission lines that are overlong and/or difficult to terminate per recommended guidelines)</li></ul>

## 30. Documenting IEEE 1149.7 test endpoints (BSDL.7)

### 30.1 Introduction

Where Clause 29 provides a summary of description and documentation needs for the unit under test in board-level and system-level test applications and, furthermore, Clause 31 defines the documentation means to be employed in the description of test modules, this clause specifies the mandatory requirements for documentation of IEEE 1149.7 devices, or parts thereof, that are test endpoints.

This clause defines a machine-readable language that allows a rigorous description of testability features in such test endpoints. The language is called the BSDL.7. It is defined as a superior superset of the IEEE 1149.1 Boundary-Scan Description Language (BSDL.1).

NOTE—Whereas the conventional abbreviation for the original Boundary-Scan Description Language, as defined by IEEE Std 1149.1, is simply BSDL, an augmented abbreviation, BSDL.1, is used herein to clearly distinguish between aspects of a description that are subject to rules of this standard versus those aspects that are subject to the rules of IEEE Std 1149.1.

The subjects described in this clause are covered in the following order:

- 30.2 Conventions
- 30.3 Purpose of BSDL.7
- 30.4 Scope of BSDL.7
- 30.5 Expectations of a BSDL.7 parser
- 30.6 Relationship of BSDL.7 to BSDL.1
- 30.7 Lexical elements of BSDL.7
- 30.8 BSDL.7 reserved words
- 30.9 Components of a BSDL.7 description
- 30.10 The entity description (BDSL.7)
- 30.11 The Standard BSDL.7 Package STD\_1149\_7\_2009
- 30.12 A typical application of BSDL.7

Detailed descriptions are presented in 30.10 for each section of a BSDL.7 description that is not found in a BSDL.1 description, as well as for each section of a BSDL.7 description that differs from the corresponding BSDL.1 description. They are documented in the order in which they are to appear in such a description. Each such subclause is organized in the following way:

- Syntax and content
- Example and additional text
- Semantic checks

Specifications within the syntax and semantic subclauses are where the language is defined; however, sometimes material in the descriptions and the examples is the key to full understanding of the language. The semantic interpretation of various lexical elements is sometimes provided in the explanatory notes.

## 30.2 Conventions

The following conventions apply to the description of BSDL.7:

- Syntax definitions are presented in Backus-Naur Form (BNF) according to BNF conventions, lexical atoms, and syntactic elements as defined for BSDL.1. (See IEEE Std 1149.1.)
- For clarity, all reserved words, predefined words, and punctuation are shown in **bold Helvetica** type within this document. VHDL reserved and predefined words will be shown in **lowercase** letters, and BSDL.7 and BSDL.1 reserved words will be shown in **UPPERCASE** letters. (BSDL.7 itself, like BSDL.1, is case-insensitive; this convention is adopted for clarity.)
- Examples are printed in *Courier* font.

## 30.3 Purpose of BSDL.7

BSDL.7 provides a machine-readable means of representing some parts of the documentary information specified for conformance and documentation requirements of IEEE Std 1149.1 and as further identified in Clause 29 of this standard. The scope of the language is defined in 30.4.

The goal of the language is to facilitate communication among companies, individuals, and tools that need to exchange information on the design of test logic that implements this standard. For example,

- A vendor of a component that supports this standard is expected to supply a BSDL.7 description to purchasers
- Automated test-generation tools may use a library of BSDL.7 descriptions to allow generation of a test for a particular loaded board
- The test logic defined by this standard could be synthesized with the support of a BSDL.7 description

BSDL.7 describes “finished” silicon, not “work-in-progress.” For example, when a bare die implementing this standard is produced, a BSDL.7 description can be provided for it. A die may be inserted into one or more types of component packages as well, with each variation described in BSDL.7. BSDL.7 for partially synthesized test logic is considered “work-in-progress,” does not necessarily implement all rules of this clause, and in most cases, is not to be transmitted beyond the synthesis environment.

## 30.4 Scope of BSDL.7

BSDL.7 is not a general-purpose hardware description language—it is intended solely as a means of describing key aspects of the implementation of this standard within a particular component. A BSDL.7 description is not itself a simulation model. Examples of features that are and are not described using BSDL.7 are listed in Table 30-1.

**Table 30-1 — Scope of BSDL.7**

<b>Features described by BSDL.7</b>	<b>Features that cannot or need not be described</b>
<ul style="list-style-type: none"> <li>- TAP.7 Class for the component.</li> <li>- Physical locations of the TAP.7 signals.</li> <li>- Availability of the optional nTRST signal (or nTRST_PD signal).</li> <li>- Instruction binary codes.</li> <li>- Device identification code.</li> <li>- Length and structure of the boundary-scan register.</li> <li>- Length and content of the configuration register.</li> </ul>	<ul style="list-style-type: none"> <li>- Zero bit scans.</li> <li>- TAP-controller state diagram.</li> <li>- Provision of <i>BYPASS</i>, <i>SAMPLE</i>, <i>PRELOAD</i>, and <i>EXTTEST</i> instructions.</li> <li>- Operation of user-defined instructions.</li> <li>- Bypass register.</li> <li>- Length of the device-identification register.</li> <li>- Escapes.</li> </ul>

Note that the language describes only features of the test logic that can vary from component to component, depending on the choice of the component designer.

Furthermore, BSDL.7 does not have a general means for providing the specification of logic levels, timing parameters, power requirements, and other similar factors. This data does not affect the logical behavior of an implementation and is most likely already described in other parts of the specification of any given component.

### **30.5 Expectations of a BSDL.7 parser**

A parser handling a form of BSDL.7 described in a version of this standard *approved by the IEEE Standards Board and published by the IEEE* is expected to handle all forms of BSDL.7 described in previous versions of this standard *approved by the IEEE Standards Board and published by the IEEE*.

A BSDL.7 parser is expected to use the Standard BSDL.7 Package content. It is also expected to reject and/or issue warning/error messages when it encounters unrecognized text. It is further expected to perform the semantic checks identified in this standard.

### **30.6 Relationship of BSDL.7 to BSDL.1**

#### **30.6.1 Description**

BSDL.7 is defined as a derivative, yet superior, superset of BSDL.1. Just as the TAP.7 architecture is an adapter that sits in front of the TAP.1, in some respects, the BSDL.7 can be thought of as an adapter that wraps the BSDL.1.

The superior aspect of BSDL.7 versus BSDL.1 means that it will be immediately failed by tools that comprehend only BSDL.1 (i.e., that do not comprehend BSDL.7). The superset aspect of BSDL.7 is adopted so as to effectively inherit the relevant BSDL.1 content that is expressly suited to the test application rather than defining it anew.

## 30.6.2 Specifications

### Rules

- a) BSDL.7 shall conform to all rules set out for BSDL.1 except as expressly specified in this clause.
- b) The IEEE 1149.7-Specified Behavior of all test endpoints shall match the behavior described in associated documentation using the BSDL.7 specified in this clause.

## 30.7 Lexical elements of BSDL.7

### 30.7.1 Description

The lexical elements of BSDL.7 are based on those specified for BSDL.1 with a small number of additions.

### 30.7.2 Specifications

### Rules

- a) The lexical elements of BSDL.7 shall be those specified for BSDL.1 except for the addition of a small number of reserved words specified in 30.8.2.

## 30.8 BSDL.7 reserved words

### 30.8.1 Description

The reserved words of BSDL.7 are specified in 30.8.2. These reserved words (identifiers) cannot be used for any other purposes in a BSDL.7 description, other than as part of a comment. For example, a reserved word cannot be used as an explicitly declared identifier.

### 30.8.2 Specifications

### Rules

- a) The identifiers listed in Table 30-2 shall be BSDL.7 reserved words having a fixed significance in the language.
- b) The identifiers listed in Table 30-2 shall not be used for any other purpose in a BSDL.7 description, other than as part of a comment.

**Table 30-2 — BSDL.7 reserved words**

CNFG0_REGISTER	T2
CNFG1_REGISTER	T3
CNFG2_REGISTER	T4N
CNFG3_REGISTER	T4W
COMPLIANCE_PATTERNS_STL	T5N
COMPONENT_CONFORMANCE_ADAPTER	T5W
COMPONENT_CONFORMANCE_STL	TAP_CLASS
CONFIG_BITS	TAP_SCAN_CLOCK_COMPACT
CONFIG_STRING	TAP_SCAN_IN_COMPACT
FMAX_PAIR	TAP_SCAN_MODE_COMPACT
STD_1149_7_2009	TAP_SCAN_OUT_COMPACT
T0	TAP_SCAN_RESET_PD
T1	

## 30.9 Components of a BSDL.7 description

### 30.9.1 Description

A BSDL.7 description is composed in the same fashion as for a BSDL.1 description with the exception that the BSDL.7 description requires two Standard BSDL.7 Packages (the superior and the subordinate; see 30.10.2.1.2).

### 30.9.2 Specifications

#### Rules

- a) A BSDL.7 description shall be composed in the same fashion as for a BSDL.1 description with the exception that both superior and subordinate Standard BSDL.7 Packages shall be provided.

## 30.10 The entity description (BSDL.7)

The entity description and supporting BSDL.7 packages makes up a BSDL.7 model of the component and is, in effect, the electronic data sheet for its test logic. It contains statements through which parameters that may vary from one chip to another are defined, as described in 30.4.

### 30.10.1 Overall structure of the entity description (BSDL.7)

#### 30.10.1.1 Syntax and content

##### 30.10.1.1.1 Description

The overall structure of the entity description (BSDL.7) is governed by 30.10.1.1.2. While VHDL permits some elements within an entity description to be in an arbitrary order, fixed ordering is required for BSDL.7. This ordering is defined to ease the development of tools that are not themselves required to be fully VHDL compliant. The <device identification register description.7>, is mandatory in a BSDL.7 description since the device identification register is mandatory for devices that provide IEEE 1149.7-Specified Behavior. This replaces the <optional register description> (which is defined as optional in a BSDL.1 description). The <configuration register description.7> is a newly inserted element, which will appear in a BSDL.7 description for a component that implements the optional configuration register.

### 30.10.1.1.2 Specifications

#### Rules

- a) The entity description and supporting BSDL.7 packages shall make up a BSDL.7 model of the endpoint.
- b) The BSDL.7 entity description shall have the following structure:

```

<BSDL.7 description> ::=

entity <component name> is
  <generic parameter>                                (see IEEE Std 1149.1)
  <logical port description>                         (see IEEE Std 1149.1)
  <standard use statement.7>                         (see 30.10.2)
  {<use statement>}                                 (see IEEE Std 1149.1)
  <component conformance statement.7>                (see 30.10.4)
  <device package pin mappings>                     (see IEEE Std 1149.1)
  [<grouped port identification>]                   (see IEEE Std 1149.1)
  <scan port identification.7>                      (see 30.10.5)
  [<compliance enable description.7>]                (see 30.10.6)
  <instruction register description>                (see IEEE Std 1149.1)
  <device identification register description.7>    (see 30.10.7)
  [<register access description>]                   (see IEEE Std 1149.1)
  <boundary-scan register description>              (see IEEE Std 1149.1)
  [<runbist description>]                           (see IEEE Std 1149.1)
  [<intest description>]                            (see IEEE Std 1149.1)
  [<configuration register description.7>]          (see 30.10.8)
  {<BSDL extensions>}                             (see IEEE Std 1149.1)
  [<design warning>]                               (see IEEE Std 1149.1)
end <component name>;

<component name> ::= <VHDL identifier>           (see IEEE Std 1149.1)

```

- c) For BSDL.7, the ordering of statements shall be fixed as shown in Rule 30.10.1.1.2 b).
- d) The elements of the <BSDL.7 description> shall be the same as those for the <BSDL description> of IEEE Std 1149.1 with the following exceptions:
  - 1) The <standard use statement.7> shall replace the <standard use statement>.
  - 2) The <component conformance statement.7> shall replace the <component conformance statement>.
  - 3) The <scan port identification.7> shall replace the <scan port identification>.
  - 4) The <compliance enable description.7> shall replace the <compliance enable description>.
  - 5) The <device identification register description.7> shall replace the <optional register description> (which is defined as optional in a BSDL.1 description).
  - 6) The <configuration register description.7> shall be a newly inserted element.
- e) The <component name> shall identify a particular type of component (e.g., part number).

#### Recommendations

- f) The <component name> should be distinct from the names of all other types of components that might be used together (i.e., that can be assembled onto a single loaded board).

### 30.10.1.2 Semantic checks

#### 30.10.1.2.1 Description

The expected semantic checks related to the overall structure of the entity description (BSDL.7) are specified in 30.10.1.2.2.

#### 30.10.1.2.2 Specifications

##### Rules

- a) Any <component name> referenced in any attribute statement shall be the same as the component name declared in the entity description.

### 30.10.2 Standard use statement (BSDL.7)

#### 30.10.2.1 Syntax and content

##### 30.10.2.1.1 Description

The syntax and content of the standard use statement are governed by 30.10.2.1.2. This statement identifies the two Standard BSDL.7 Packages in which attributes, types, constants, and other elements are defined and can be referenced elsewhere in the BSDL.7 description.

The construct of the <superior use statement.7> or <subordinate use statement.7> may not be syntactically complete for use by a given VHDL analyzer. This is because a library or default working area has not been specified. In such a case, a complete statement is “use work.STD\_1149\_7\_2009.all;” in which the preface “work.” tells a VHDL analyzer to find the Standard BSDL.7 Package in the current work area. The field “work.” could be replaced by an arbitrary library name such as “BSCAN.” telling a VHDL analyzer where in its system of libraries to find the Standard BSDL.7 Package. Since there is no standardization of library structures from one VHDL environment to another, some editing of BSDL.7 files to specify the location of Standard BSDL.7 Packages is generally unavoidable. The specification used in this subclause may cause an error in a VHDL analyzer, forcing the user to edit the BSDL.7 file for the correct location of the Standard BSDL.7 Package information.

As a general rule, future enhancements to the BSDL.7 language will be designed as extensions to previous versions of the language. Therefore, the version information contained in the <superior BSDL package identifier.7> may be used by BSDL.7-specific tools to indicate which tool versions will be able to process the information in the input file. For example, if the input file records the version as “\_2009”, then tools that can process any language variant “\_2009” or later will be able to process the input file correctly. Versions of the BSDL.7 language are not to be confused with the version of this standard that a given IC may implement. The <component conformance statement.7> identifies the version of the standard (see 30.10.4).

Within a specific system processing BSDL.7 descriptions, the Standard BSDL.7 Packages may be files somewhere in the file system of the host computer. The .all suffix is meaningful to VHDL and is not part of a file name.

### **30.10.2.1.2 Specifications**

## Rules

- a) The <standard use statement.7> shall identify the two Standard BSDL.7 Packages, superior and subordinate, in which attributes, types, constants, and other elements are defined that shall be referenced elsewhere in the BSDL.7 description.
  - b) The <standard use statement.7> shall identify the following syntax:

<standard use statement.7> ::= <superior use statement.7> <subordinate use statement.7>

<superior use statement.7> ::= **use** <superior BSDL package identifier.7>.**all**;  
<subordinate use statement.7> ::= **use** <subordinate BSDL package identifier.7>.**all**;

<superior BSDL package identifier.7>::=  
    **STD\_1149\_7\_2009** | <other package identifier.7>

<subordinate BSDL package identifier.7> ::=  
    <standard BSDL package identifier>

(see IEEE Std 1149.1)

<other package identifier.7> ::=  
    <VHDL identifier>

(see IEEE Std 1149.1)

- c) The <superior BSDL package identifier.7> and <subordinate BSDL package identifier.7> shall be the names of the Standard BSDL.7 Packages that contain the information to be included.
  - d) The suffix .all shall indicate that all declarations within the BSDL.7 packages are to be used.
  - e) While VHDL permits the use of a wider range of suffixes, .all shall be the only suffix permitted in BSDL.7.
  - f) The <superior use statement.7> shall indicate:
    - 1) An instruction to tools to read a standard package that contains BSDL.7 syntax definitions.
    - 2) The version of the BSDL.7 language (shown by the year number embedded in the identifier) that was used when creating the BSDL.7 file.
  - g) The content of the Standard BSDL.7 Packages shall be the current definition of the BSDL.7 language and shall not be modified by users.
  - h) The <standard use statement.7> shall appear in every BSDL.7 description before any <use statement> (see IEEE Std 1149.1) so that tools that are fully VHDL compliant can locate information relevant to all components that implement this standard. (Tools that are limited to the BSDL.7 language shall always use this information).

## Permissions

- i) Values for <other package identifier.7> may be assigned with future revisions as the language evolves.

### 30.10.2.2 Examples

```
use STD_1149_7_2009.all;  
use STD_1149_1_2013.all; -- Today
```

or

```
use STD_1149_7_2093.all;
```

```
use STD_1149_1_2091.all; -- Sometime in the future
```

The contents of the **STD\_1149\_7\_2009** Standard BSDL.7 Package are listed in 30.11.

The contents of the **STD\_1149\_1\_2013** Standard BSDL.7 Package and its associated Standard BSDL.7 Package Body are specified by IEEE Std 1149.1-2013.

### 30.10.3 Version control

#### 30.10.3.1 Syntax and content

##### 30.10.3.1.1 Description

The additional application of the <superior use statement.7> specified in 30.10.3.1.2 is intended to provide backward compatibility to all BSDL.7 descriptions already written and can be used in a similar manner for BSDL.7 descriptions based on future revisions of this standard.

##### 30.10.3.1.2 Specifications

###### Rules

- a) The <superior use statement.7> shall indicate whether the BSDL.7 description has been written using the syntax defined in this clause or a syntax defined in some future version.
- b) Version control as it pertains to the <subordinate use statement.7> defined by this standard shall be as per the <standard use statement> defined by IEEE Std 1149.1.

###### Permissions

- c) Specifically constructed foreign attributes may be added via <BSDL extensions> (see IEEE Std 1149.1).
- d) As such are defined as <BSDL extensions>, attributes associated with IEEE Std 1149.4 [B1], IEEE Std 1149.6 [B2], and IEEE Std 1532 [B4] may be added to a BSDL.7 description for a device that provides behavior specified by any one or more of these standards in addition to providing IEEE 1149.7-Specified Behavior.

### 30.10.4 Component conformance statement (BSDL.7)

#### 30.10.4.1 Syntax and content

##### 30.10.4.1.1 Description

The component conformance statement.7 is specified in 30.10.4.1.2.

For BSDL.7, the matter of conformance is bifurcated according to two aspects—those subject to the rules of this standard, which is to say the adapter, versus those subject to the rules of IEEE Std 1149.1, which is to say the STL.

In general, it is possible for a component designed prior to the writing of this clause to be described by the version of BSDL.7 defined herein, but this cannot imply that the component implements the rules of this standard. The <component conformance statement.7> allows tools to account for changes in the rules that may occur in future editions of this standard versus past editions or the current edition.

### 30.10.4.1.2 Specifications

#### Rules

- a) The <component conformance statement.7> shall identify the edition of this standard for which the testability circuitry of a physical component has been implemented.
- b) The <component conformance statement.7> shall identify the following syntax:

```
<component conformance statement.7>::=  
    <adapter conformance statement.7><STL conformance statement.7>  
  
<adapter conformance statement.7>::= attribute COMPONENT_CONFORMANCE_ADAPTER of  
    <component name> : entity is <adapter conformance string.7>;  
<STL conformance statement.7>::= attribute COMPONENT_CONFORMANCE_STL of  
    <component name> : entity is <STL conformance string.7>;  
  
<adapter conformance string.7>::= "<adapter conformance identification.7>"  
<STL conformance string.7>::=  
    <conformance string>                                (see IEEE Std 1149.1)  
  
<adapter conformance identification.7>::= STD_1149_7_2009
```

- c) The symbol **STD\_1149\_7\_2009** shall refer to this standard.

#### Permissions

- d) Subsequent editions of this standard may add new values to the <adapter conformance identification.7> element.

NOTE: Though this edition of this standard is subsequent to the original issue in 2009, it has been deemed unnecessary to add new values to the <adapter conformance identification.7> element, since none of the normative text has changed in any way that would benefit by assertion of such new values.

### 30.10.4.2 Examples

```
attribute COMPONENT_CONFORMANCE_ADAPTER of My_Dot7_IC : entity is  
    "STD_1149_7_2009"; -- invokes 2009 rules for IEEE Std 1149.7  
attribute COMPONENT_CONFORMANCE_STL of My_Dot7_IC : entity is  
    "STD_1149_1_2013"; -- invokes 2013 rules for IEEE Std 1149.1
```

### 30.10.4.3 Semantic checks

No semantic checks are necessary for this statement. However, some semantic checks described in future versions of this standard may be influenced by values that appear in the <adapter conformance identification.7> element. Likewise, some semantics checks described in IEEE Std 1149.1 may be influenced by the value that appears in the <STL conformance identification statement.7> element.

## 30.10.5 Scan port identification (BSDL.7)

### 30.10.5.1 Syntax and content

#### 30.10.5.1.1 Description

Scan port identification statements define the TAP.7 of the component.

### 30.10.5.1.2 Specifications

#### Rules

- a) The scan port identification statements shall define the TAP.7 of the component.
- b) The <scan port identification.7> shall identify the following syntax:

```

<scan port identification.7>::=
    <scan port tap class statement.7>
    <scan port identification list.7>

<scan port tap class statement.7>::= attribute TAP_CLASS of <component name> : entity is
    <scan port tap class string.7>
<scan port tap class string.7>::= " <scan port tap class.7> "
<scan port tap class.7>::= T0 | T1 | T2 | T3 | T4N | T4W | T5N | T5W

<scan port identification list.7>::=
    } <TCK stmt> <TMS stmt> [ <TDI stmt> ] [ <TDO stmt> ]      (see IEEE Std 1149.1)
    [ <TCKC stmt.7> ] [ <TMSC stmt.7> ] [ <TDIC stmt.7> ] [ <TDOC stmt.7> ]
    [ <TRST stmt.7> ] }

<TCKC stmt.7>::= attribute TAP_SCAN_CLOCK_COMPACT of <port ID> : signal is
    ( <rising edge fmax.7>, <falling edge fmax.7> );
<TMSC stmt.7>::= attribute TAP_SCAN_MODE_COMPACT of <port ID> : signal is true;
<TDIC stmt.7>::= attribute TAP_SCAN_IN_COMPACT of <port ID> : signal is true;
<TDOC stmt.7>::= attribute TAP_SCAN_OUT_COMPACT of <port ID> : signal is true;

<TRST stmt.7>::=
    <TRST stmt> |                               (see IEEE Std 1149.1)
    <TRST_PD stmt.7>

<TRST_PD stmt.7>::= attribute TAP_SCAN_RESET_PD of <port ID> : signal is true;

<rising edge fmax.7>::=
    <real number>                                (see IEEE Std 1149.1)
<falling edge fmax.7>::=
    <real number>                                (see IEEE Std 1149.1)

```

NOTE 1—Items enclosed between the characters “{” and “}” may appear in any order.

NOTE 2—Where it appears within a syntax element name, “stmt” can be interpreted as shorthand for “statement.”

- c) <rising edge fmax.7>, <falling edge fmax.7> shall be a pair consisting of:
  - 1) A real number that gives the maximum operating frequency for TCKC in hertz for Advanced Protocol with rising-edge timing that is valid for all supported modes. In the examples in 30.10.5.2, 27.0 MHz is specified as this maximum operating frequency.
  - 2) A real number that gives the maximum operating frequency for TCKC in hertz for Advanced Protocol with falling-edge timing that is valid for all supported modes. In the examples in 30.10.5.2, 35.0 MHz is specified as this maximum operating frequency.

### 30.10.5.2 Examples

An example for a T3 TAP.7; the optional nTRST signal is present:

```

port (TDI_TDIC : in bit;
      TDO_TDOC : out bit;
      TMS : in bit;
      TCK : in bit;
      nTRST : in bit;

      (... some BSDL deleted for brevity ...)

attribute TAP_CLASS of My_Dot7 IC_T3 : entity is "T3";
attribute TAP_SCAN_IN of TDI_TDIC : signal is true;
attribute TAP_SCAN_OUT of TDO_TDOC : signal is true;
attribute TAP_SCAN_MODE of TMS : signal is true;
attribute TAP_SCAN_CLOCK of TCK : signal is (20.0e6, BOTH);
attribute TAP_SCAN_RESET of nTRST : signal is true;
attribute TAP_SCAN_IN_COMPACT of TDI_TDIC : signal is true;
attribute TAP_SCAN_OUT_COMPACT of TDO_TDOC : signal is true;

```

An example where AUX1 and AUX2 system functions are available at TDI(C) and TDO(C) signals for a T4 TAP.7; the optional nTRST\_PD signal is present also:

```

port (TDI_TDIC_AUX1 : inout bit;
      TDO_TDOC_AUX2 : inout bit;
      TMS_TMSC : inout bit;
      TCK_TCKC : in bit;
      nTRST_PD : in bit;

      (... some BSDL deleted for brevity ...)

attribute TAP_CLASS of My_Dot7_IC_T4W : entity is "T4W";
attribute TAP_SCAN_IN of TDI_TDIC_AUX1 : signal is true;
attribute TAP_SCAN_OUT of TDO_TDOC_AUX2 : signal is true;
attribute TAP_SCAN_MODE of TMS_TMSC : signal is true;
attribute TAP_SCAN_CLOCK of TCK_TCKC : signal is (20.0e6, BOTH);
attribute TAP_SCAN_RESET_PD of nTRST_PD : signal is true;
attribute TAP_SCAN_IN_COMPACT of TDI_TDIC_AUX1 : signal is true;
attribute TAP_SCAN_OUT_COMPACT of TDO_TDOC_AUX2 : signal is true;
attribute TAP_SCAN_MODE_COMPACT of TMS_TMSC : signal is true;
attribute TAP_SCAN_CLOCK_COMPACT of TCK_TCKC : signal is
(25.0e6, 35.0e6);

```

An example for a T5N TAP.7; no Test Reset Signal is present:

```

port (TMS_TMSC : inout bit;
      TCK_TCKC : in bit;

      (... some BSDL deleted for brevity ...)

attribute TAP_CLASS of My_Dot7_IC_T5N : entity is "T5N";
attribute TAP_SCAN_MODE of TMS_TMSC : signal is true;
attribute TAP_SCAN_CLOCK of TCK_TCKC : signal is (20.0e6, BOTH);
attribute TAP_SCAN_MODE_COMPACT of TMS_TMSC : signal is true;
attribute TAP_SCAN_CLOCK_COMPACT of TCK_TCKC : signal is
(25.0e6, 35.0e6);

```

The port IDs used in the above examples closely match the signal names defined in this standard; however, arbitrary names could have been used.

The meaning of each signal in relation to this standard is to be deduced from the attribute name, not from the value of a <port ID> element in the <scan port identification.7>.

The <real number> specified for the <TCK stmt> gives the maximum operating frequency for TCK in hertz for Standard Protocol/JScan that is valid for all supported modes. In the examples above, 22.0 MHz is specified as this maximum operating frequency.

### 30.10.5.3 Semantic checks

#### 30.10.5.3.1 Description

The expected semantic checks related to scan port identification statements are specified in 30.10.5.3.2.

#### 30.10.5.3.2 Specifications

##### Rules

- a) The <TDI stmt> and the <TDO stmt> shall not be present when the value of <scan port tap class.7> is T4N or T5N. On the contrary, they shall be present when <scan port tap class.7> has other values.
- b) The <TDIC stmt> and the <TDOC stmt> shall be present when the value of <scan port tap class.7> is T3, T4W, or T5W. Otherwise, they shall not be present when <scan port tap class.7> has other values.
- c) The <TMSC stmt> and the <TCKC stmt> shall be present when the value of <scan port tap class.7> is T4N, T4W, T5N, or T5W. On the contrary, they shall not be present when <scan port tap class.7> has other values.
- d) With respect to the <port ID> elements in the <scan port identification.7>:
  - 1) The unique <port name> element in a <port ID> occurring in a <TDOC stmt> shall appear as a value in an <identifier list> of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **out** or **inout**, the latter being valid only where the value of <scan port tap class.7> is **T4W** or **T5W**.

NOTE 1—Since the <port ID> that occurs in the corresponding <TDO stmt> has the same value as that which occurs in the <TDOC stmt>, it inherits this more permissive check with respect to <pin type> notwithstanding the more restrictive check of IEEE Std 1149.1.

- 2) The unique <port name> element in a <port ID> occurring in a <TDIC stmt> shall appear as a value in an <identifier list> of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **in** or **inout**, the latter being valid only where the value of <scan port tap class.7> is **T4W** or **T5W**.

NOTE 2—Since the <port ID> that occurs in the corresponding <TDI stmt> has the same value as that which occurs in the <TDIC stmt>, it inherits this more permissive check with respect to <pin type> notwithstanding the more restrictive check of IEEE Std 1149.1.

- 3) The unique <port name> element in a <port ID> occurring in a <TMSC stmt.7> shall appear as a value in an <identifier> list of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **inout**.
- 4) The unique <port name> element in a <port ID> occurring in a <TCKC stmt.7> or a <TRST\_PD stmt.7> shall appear as a value in an <identifier list> of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **in**.

- 5) Any <port ID> appearing in the <scan port identification.7> shall be a <subscripted port name> or a <port name> defined in a <logical port description> statement with a corresponding <port dimension> equal to **bit**.
- e) No <port ID> in the <scan port identification.7> shall later appear in the <boundary-scan register description>.
- f) A given <port ID> shall occur at most once in the <scan port identification.7> excepting all of the following:
  - 1) if the <TCKC stmt.7> is present, the <port ID> that occurs in it shall also occur in the <TCK stmt>, and
  - 2) if the <TMSC stmt.7> is present, the <port ID> that occurs in it shall also occur in the <TMS stmt>, and
  - 3) if the <TDIC stmt.7> is present, the <port ID> that occurs in it shall also occur in the <TDI stmt>, and
  - 4) if the <TDOC stmt.7> is present, the <port ID> that occurs in it shall also occur in the <TDO stmt>.
- g) If a <port ID> is a <subscripted port name>, the <subscript> shall lie within the range specified for the **bit\_vector** of the relevant port.
- h) No value of a <port ID> element in the <scan port identification> shall have appeared as a <representative port> or an <associated port> in a <twin group>.

### 30.10.6 Compliance enable description (BSDL.7)

#### 30.10.6.1 Syntax and content

##### 30.10.6.1.1 Description

This portion of a BSDL.7 description is required in the description of a test endpoint if the optional compliance-enable feature of IEEE Std 1149.1 has been implemented. Otherwise, improper operation of the STL may occur during an automatically generated test.

##### 30.10.6.1.2 Specifications

###### Rules

- a) The compliance-enable portion of a BSDL.7 description shall appear in the description of a test endpoint, provided that the optional compliance-enable feature of IEEE Std 1149.1 has been implemented.
- b) The <compliance enable description.7> shall identify the following syntax:

<compliance enable description.7> ::= <STL compliance enable description.7>

<STL compliance enable description.7> ::= **attribute COMPLIANCE\_PATTERNS\_STL of**  
<component name> : **entity is** <STL compliance pattern string.7>;

<STL compliance pattern string.7> ::=  
<compliance pattern string> (see IEEE Std 1149.1)

#### 30.10.6.2 Examples

```
attribute COMPLIANCE_PATTERNS_STL of My_Dot7_IC : entity is
    "(ENA_STL, nDIS_STL) (11)";
```

Software that generates tests using test facilities defined by IEEE Std 1149.1 sets up all compliance-enable conditions before exercising the STL of the affected component. Any compliance-enable pattern is held constant for the duration of all boundary-scan testing.

### 30.10.6.3 Semantic checks

### **30.10.6.3.1 Description**

The expected semantic checks related to compliance-enable statements are specified in 30.10.6.3.2.

### **30.10.6.3.2 Specifications**

## Rules

- a) The semantics checks for the <STL compliance enable description.7> of BSDL.7 shall be those defined for the <compliance enable description> of BSDL.1.

### **30.10.7 Device identification register description (BSDL.7)**

### **30.10.7.1 Syntax and content**

### **30.10.7.1.1 Description**

The device identification register description of BSDL.7 differs from the optional register description of BSDL.1 only in the respect that it is required to appear in a BSDL.7 description, since the device identification register is mandatory for devices that provide IEEE 1149.7-Specified Behavior.

### **30.10.7.1.2 Specifications**

## Rules

- a) The <device identification register description.7> shall identify the following syntax:

NOTE—The values appearing in bit positions 27-1 of the associated <32-bit pattern string> are to be taken as the values associated with TCA bits 26-0, respectively.

### 30.10.7.2 Examples

```

attribute IDCODE_REGISTER of My_Dot7_IC : entity is
    "0011" &                                -- Version
    "0010110011101001" &                    -- Part number
    "00001010100" &                         -- Manufacturer identity
    "1";                                     -- LSB value required to be 1
attribute USERCODE_REGISTER of My_Dot7_IC : entity is
    "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX";

```

For this example, the least significant bits of the TCA (bits 26-0) are defined as 001011001110100100001010100.

### 30.10.7.3 Semantic checks

#### 30.10.7.3.1 Description

The expected semantic checks related to device identification register statements are specified in 30.10.7.3.2.

#### 30.10.7.3.2 Specifications

##### Rules

- a) The semantics checks for the <device identification register description.7> of BSDL.7 shall be those defined for the <optional register description> of BSDL.1.

### 30.10.8 Configuration register description (BSDL.7)

#### 30.10.8.1 Syntax and content

##### 30.10.8.1.1 Description

This subclause describes which optional segments of the configuration register are provided and specifies the bit patterns that represent their content per Table 9-15 through Table 9-17. This optional part of the BSDL.7 description is omitted with a T0 TAP.7.

NOTE—Use of the term “configuration” here is not to be confused with its use in IEEE Std 1532 [B4] (where the term refers to device programming).

##### 30.10.8.1.2 Specifications

##### Rules

- a) The <configuration register description.7> shall identify the following syntax:

```
<configuration register description.7>::=  
    <cfg0 stmt.7> [ , <cfg1 stmt.7> ] [ , <cfg2 stmt.7> ] [ , <cfg3 stmt.7> ]  
  
<cfg0 stmt.7> ::= attribute CNFG0_REGISTER of <component name> : entity is  
    <32-bit pattern string>;  
<cfg1 stmt.7> ::= attribute CNFG1_REGISTER of <component name> : entity is  
    <32-bit pattern string>;  
<cfg2 stmt.7> ::= attribute CNFG2_REGISTER of <component name> : entity is  
    <32-bit pattern string>;  
<cfg3 stmt.7> ::= attribute CNFG3_REGISTER of <component name> : entity is  
    <32-bit pattern string>;  
  
<32-bit pattern string> ::=  
    " <32-bit pattern>"  
                                (see IEEE Std 1149.1)
```

NOTE—Where it appears within a syntax element name, “stmt” can be interpreted as shorthand for “statement.”

#### 30.10.8.2 Examples

An example for a component with a T1 TAP.7 with only CNFG0 [11:0] implemented:

```
attribute CNFG0_REGISTER of My_Dot7 IC1 : entity is  
    "XXX" & "XXXXXXXXXX" & "X" & "XXXXXXX" & "0000" & "0001" & "0001";
```

NOTE 1—The CNFG0 register is nominally of length 32; unimplemented has been represented as “**X**”.

An example for a component with T5(N) TAP.7 with all configuration registers (CNFG0, CNFG1, CNFG2, and CNFG3) and many supported TAP.7 options:

```
attribute CNFG0_REGISTER of My_Dot7_IC2 : entity is
    "000" & "11001111" & "1" & "1111111" & "1111" & "0001" & "0101";
attribute CNFG1_REGISTER of My_Dot7_IC2 : entity is
    "00000000000000000000000000000000" & "1001";
attribute CNFG2_REGISTER of My_Dot7_IC2 : entity is
    "XXX0XXXXXXXXXXXXXXXXXXXXXX1";
attribute CNFG3_REGISTER of My_Dot7_IC2 : entity is
    "XXXXXXXXXXXXXX1XXXXXXXXXXXX0000";
```

NOTE 2—Each CNFG0, CNFG1, CNFG2, and CNFG3 register is nominally of length 32; here “**X**” has been used to mask bits in the user-defined CNFG2 and CNFG3 registers that are desired to be undocumented.

An “**X**” may be used to mask nominally required subfields within a content string that are not implemented or, where user defined, that are desired to be undocumented.

### 30.10.8.3 Semantic checks

#### 30.10.8.3.1 Description

The expected semantic checks related to the configuration register description are specified in 30.10.8.3.2.

#### 30.10.8.3.2 Specifications

##### Rules

- a) A <configuration register description.7> shall appear in a BSDL.7 description, provided T0 is not the value of the <TAP class stmt.7> element.
- b) Where 0 appears in bit position 8 in the <CNFG0 stmt.7>, 1 shall not appear in any of the bit positions 31-12 in the <CNFG0 stmt.7>.
- c) A <CNFG1 stmt.7> shall appear in a BSDL.7 description if and only if 1 appears in bit position 9 in the <CNFG0 stmt.7>.
- d) A <CNFG2 stmt.7> shall appear in a BSDL.7 description if and only if 1 appears in bit position 10 in the <CNFG0 stmt.7>.
- e) A <CNFG3 stmt.7> shall appear in a BSDL.7 description if and only if 1 appears in bit position 11 in the <CNFG0 stmt.7>.

## 30.11 The Standard BSDL.7 Package STD\_1149\_7\_2009

### 30.11.1 Description

This information defines the basis of BSDL.7 and would typically be write-protected by a system administrator as it is not to be changed. BSDL.7 descriptions that use <superior BSDL.7 package identifier.<sup>7</sup>> **STD\_1149\_7\_2009** are to be processed using this Standard BSDL.7 Package.

### 30.11.2 Specifications

#### Rules

- a) The following shall be the complete content of Standard BSDL.7 Package STD\_1149\_7\_2009:

```
package STD_1149_7_2009 is

    -- Give component conformance declaration

    attribute COMPONENT_CONFORMANCE_ADAPTER : string;
    attribute COMPONENT_CONFORMANCE_STL : string;

    -- Give scan port.7 declarations

    type FMAX_PAIR is record
        FMAX_RE : real;
        FMAX_FE : real;
    end record;

    attribute TAP_SCAN_CLOCK_COMPACT : FMAX_PAIR;
    attribute TAP_SCAN_MODE_COMPACT : boolean;
    attribute TAP_SCAN_IN_COMPACT : boolean;
    attribute TAP_SCAN_OUT_COMPACT : boolean;
    attribute TAP_SCAN_RESET_PD : boolean;

    -- Give configuration register declarations

    type CONFIG_BITS is ('0', '1', 'x', 'X');
    type CONFIG_STRING is array (31 downto 0) of CONFIG_BITS;
    attribute CNFG0_REGISTER : CONFIG_STRING;
    attribute CNFG1_REGISTER : CONFIG_STRING;
    attribute CNFG2_REGISTER : CONFIG_STRING;
    attribute CNFG3_REGISTER : CONFIG_STRING;

    -- Miscellaneous

    attribute COMPLIANCE_PATTERNS_STL : string;

end STD_1149_7_2009; -- End of 1149.7-2009 Package
```

## 30.12 A typical application of BSDL.7

```
entity My_Dot7_IC is
    generic (PHYSICAL_PIN_MAP: string := "My_Pkg");

    port (TDI_TDIC_AUX1 : inout bit;
          TDO_TDOC_AUX2 : inout bit;
```

```

TMS_TMSC : inout bit;
TCK_TCKC : in bit;
nTRST_PD : in bit;
GND : power_0 bit_vector (1 to 2);
VCC : power_pos bit_vector (1 to 2);
SYSCK : in bit;
DQ : inout bit_vector (7 downto 0);
NC : linkage_mechanical bit_vector (1 to 2);
ENA_STL : in bit;
nDIS_STL : in bit;
STRB : out bit;
AB : out bit_vector (4 downto 0) );

use STD_1149_7_2009.all;
use STD_1149_1_2013.all;

attribute COMPONENT_CONFORMANCE_ADAPTER of My_Dot7_IC :
    entity is "STD_1149_7_2009";
attribute COMPONENT_CONFORMANCE_STL of My_Dot7_IC :
    entity is "STD_1149_1_2013";

attribute PIN_MAP of My_Dot7_IC : entity is PHYSICAL_PIN_MAP;
constant My_Pkg : PIN_MAP_STRING :=
    "TDI_TDIC_AUX1 : 1," &
    "TDO_TDOC_AUX2 : 2," &
    "TMS_TMSC : 3," &
    "TCK_TCKC : 4," &
    "nTRST_PD : 5," &
    "GND : (6,21), VCC : (8,23)," &
    "SYSCK : 7," &
    "DQ : (9,10,11,12,17,18,19,20)," &
    "NC : (13,15)," &
    "ENA_STL : 14," &
    "nDIS_STL : 16," &
    "STRB : 22," &
    "AB : (28,27,26,25,24)";

attribute TAP_CLASS of My_Dot7_IC : entity is "T5W"
attribute TAP_SCAN_CLOCK of TCK_TCKC : signal is (20.0e6 , BOTH);
attribute TAP_SCAN_MODE of TMS_TMSC: signal is true;
attribute TAP_SCAN_IN of TDI_TDIC_AUX1 : signal is true;
attribute TAP_SCAN_OUT of TDO_TDOC_AUX2: signal is true;
attribute TAP_SCAN_CLOCK_COMPACT of TCK_TCKC : signal is
    ( 27.0e6 , 35.0e6 );
attribute TAP_SCAN_MODE_COMPACT of TMS_TMSC: signal is true;
attribute TAP_SCAN_IN_COMPACT of TDI_TDIC_AUX1 : signal is true;
attribute TAP_SCAN_OUT_COMPACT of TDO_TDOC_AUX2: signal is true;
attribute TAP_SCAN_RESET_PD of nTRST_PD : signal is true;

attribute COMPLIANCE_PATTERNS_STL of My_Dot7_IC : entity is
    "( ENA_STL , nDIS_STL ) ( 11 )";

attribute INSTRUCTION_LENGTH of My_Dot7_IC : entity is 4;
attribute INSTRUCTION_OPCODE of My_Dot7_IC : entity is
    "EXTEST (0011), " &
    "EXTEST (1011), " &
    "BYPASS (1111), " &

```

```
"SAMPLE (0001, 1000), " &
"PRELOAD(1001, 1000)," &
"HIGHZ (0101), " &
"CLAMP (0110), " &
"IDCODE (1101), " &
"USERCODE (1110), " &
"SECRET (1010) ";
attribute INSTRUCTION_CAPTURE of My_Dot7_IC : entity is "1X01";
attribute INSTRUCTION_PRIVATE of My_Dot7_IC : entity is "SECRET";

attribute IDCODE_REGISTER of My_Dot7_IC : entity is
  "00110010110011101001000010101001";
attribute USERCODE_REGISTER of My_Dot7_IC : entity is
  "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX";

attribute BOUNDARY_LENGTH of My_Dot7_IC : entity is 25;
attribute BOUNDARY_REGISTER of My_Dot7_IC : entity is
  " 0 ( BC_1, SYSCK, input, X, OPEN1 )," &
  " 1 ( BC_1, DQ(7), output3, X, 17, 1, Z )," &
  " 2 ( BC_1, DQ(7), input, X, OPEN1 )," &
  " 3 ( BC_1, DQ(6), output3, X, 17, 1, Z )," &
  " 4 ( BC_1, DQ(6), input, X, OPEN1 )," &
  " 5 ( BC_1, DQ(5), output3, X, 17, 1, Z )," &
  " 6 ( BC_1, DQ(5), input, X, OPEN1 )," &
  " 7 ( BC_1, DQ(4), output3, X, 17, 1, Z )," &
  " 8 ( BC_1, DQ(4), input, X, OPEN1 )," &
  " 9 ( BC_1, DQ(3), output3, X, 17, 1, Z )," &
  "10 ( BC_1, DQ(3), input, X, OPEN1 )," &
  "11 ( BC_1, DQ(2), output3, X, 17, 1, Z )," &
  "12 ( BC_1, DQ(2), input, X, OPEN1 )," &
  "13 ( BC_1, DQ(1), output3, X, 17, 1, Z )," &
  "14 ( BC_1, DQ(1), input, X, OPEN1 )," &
  "15 ( BC_1, DQ(0), output3, X, 17, 1, Z )," &
  "16 ( BC_1, DQ(0), input, X, OPEN1 )," &
  "17 ( BC_1, *, control, 1 )," &
  "18 ( BC_1, STRB, output2, 1, 18, 1, WEAK1 )," &
  "19 ( BC_1, AB(0), output3, X, 24, 1, Z )," &
  "20 ( BC_1, AB(1), output3, X, 24, 1, Z )," &
  "21 ( BC_1, AB(2), output3, X, 24, 1, Z )," &
  "22 ( BC_1, AB(3), output3, X, 24, 1, Z )," &
  "23 ( BC_1, AB(4), output3, X, 24, 1, Z )," &
  "24 ( BC_1, *, control, 1 )";

attribute CNFG0_REGISTER of My_Dot7_IC : entity is
  "000110011111111111111100010101";
attribute CNFG1_REGISTER of My_Dot7_IC : entity is
  "000000000000000000000000000000010001";
attribute CNFG2_REGISTER of My_Dot7_IC : entity is
  "XXX0XXXXXXXXXXXXXXXXXXXXXX1";
attribute CNFG3_REGISTER of My_Dot7_IC : entity is
  "XXXXXXXXXXXXXXXXXXXX1XXXXXXXXXXXX0000";

end My_Dot7_IC;
```

## 31. Documenting IEEE 1149.7 test modules (HSDL.7)

### 31.1 Introduction

Where Clause 29 provides a summary of description and documentation needs for the unit under test in board-level and system-level test applications and, furthermore, Clause 30 defines the documentation means to be employed in the description of test endpoints, this clause specifies the mandatory requirements for documentation of IEEE 1149.7 devices, or parts thereof, that are test modules.

This clause defines a machine-readable language that allows rigorous description of testability features in such test modules. The language is called the HSDL.7. It is defined as a hierarchically superior superset/subset of the BSDL.7.

The subjects described in this clause are covered in the following order:

- 31.2 Conventions
- 31.3 Purpose of HSDL.7
- 31.4 Scope of HSDL.7
- 31.5 Expectations of an HSDL.7 parser
- 31.6 Relationship of HSDL.7 to BSDL.7 (and BSDL.1)
- 31.7 Lexical elements of HSDL.7
- 31.8 HSDL.7 reserved words
- 31.9 Components of an HSDL.7 description
- 31.10 The entity description (HSDL.7)
- 31.11 The Standard HSDL.7 Package STD\_1149\_7\_2009\_module
- 31.12 Applications of HSDL.7

Detailed descriptions are presented in 31.10 for each section of an HSDL.7 description that is not found in a BSDL.7 description as well as for each section of an HSDL.7 description that differs from the corresponding BSDL.7 description. They are documented in the order in which they are to appear in such a description. Each such subclause is organized in the following way:

- Syntax and content
- Example and additional text
- Semantic checks

Specification within the syntax and semantic subclauses is where the language is defined; however, sometimes material in the descriptions and the examples is the key to full understanding of the language. The semantic interpretation of various lexical elements is sometimes provided in the explanatory notes.

### 31.2 Conventions

The following conventions apply to the description of HSDL.7.

- Syntax definitions are presented in BNF according to BNF conventions, lexical atoms, and syntactic elements as defined for BSDL.1. (See IEEE Std 1149.1.)

- For clarity, all reserved words, predefined words, and punctuation are shown in **bold Helvetica** type within this document. VHDL reserved and predefined words will be shown in **lowercase** letters, and HSDL.7, BSDL.7, and BSDL.1 reserved words will be shown in **UPPERCASE** letters. [HSDL.7 itself, like BSDL.7 (and BSDL.1), is case-insensitive; this convention is adopted for clarity.]
- Examples are printed in *Courier* font.

### 31.3 Purpose of HSDL.7

HSDL.7 provides a machine-readable means of representing some parts of the documentary information specified for conformance and documentation requirements as identified in Clause 29 of this standard. The scope of the language is defined in 31.4.

The goal of the language is to facilitate communication among companies, individuals, and tools that need to exchange information on the design of test logic that implements this standard. For example:

- A vendor of a compound component that supports this standard is expected to supply an HSDL.7 description to purchasers
- Automated test-generation tools may use a library of HSDL.7 descriptions (and associated BSDL.7 and/or BSDL.1 descriptions) to allow generation of a test for a particular loaded board
- The test logic defined by this standard could be synthesized with the support of an HSDL.7 description

HSDL.7 describes “finished” silicon, not “work-in-progress”. For example, when a module implementing this standard is produced, an HSDL.7 description can be provided for it. A module may be integrated into one or more types of higher level modules as well, with each variation described in HSDL.7. HSDL.7 for partially synthesized test logic is considered “work-in-progress”, does not necessarily implement all rules of this clause and, in most cases, is not to be transmitted beyond the synthesis environment.

### 31.4 Scope of HSDL.7

HSDL.7 is not a general-purpose hardware description language—it is intended solely as a means of describing key aspects of the implementation of this standard within a particular compound component. An HSDL.7 description is not itself a simulation model. Examples of features that are and are not described using HSDL.7 are listed in Table 31-1.

**Table 31-1 — Scope of HSDL.7**

<b>Features described by HSDL.7</b>	<b>Features that cannot or need not be described</b>
<ul style="list-style-type: none"> <li>- Physical locations of the TAP.7 signals.</li> <li>- Availability of the optional nTRST signal (or nTRST_PD signal).</li> <li>- Subcomponents that comprise the module.</li> <li>- Node ID for subcomponents with a T3 and above TAP.7.</li> <li>- Logical interconnection of subcomponent signals within a module.</li> </ul>	<ul style="list-style-type: none"> <li>- TAP.7 Class for a module.</li> <li>- Zero bit scans, TAP-controller state diagram, Escapes.</li> <li>- Means of selecting IEEE 1149.7 uses for TAP.7 signals where other technologies share these pins within a module.</li> <li>- Means of selecting a given topology use for TAP.7 signals where other topologies share these pins within a module.</li> <li>- Means of selection of a given TAP.7 for use within a given topology.</li> </ul>

Note that the language describes only features of the test logic that can vary from component to component, depending on the choice of the component designer.

Furthermore, HSDL.7 does not have a general means for providing for the specification of logic levels, timing parameters, power requirements, and other similar factors. This data does not affect the logical behavior of an implementation and is most likely already described in other parts of the specification of any given component.

HSDL.7 supports only modules that provide IEEE 1149.7-Specified Behavior. A great deal of leniency is provided when describing modules, as allowed by this standard. The language is designed in the spirit of BSDL.7, and hence, the pros and cons of BSDL.7 (and BSDL.1) syntax remain.

### **31.5 Expectations of an HSDL.7 parser**

A parser handling a form of HSDL.7 described in a version of this standard *approved by the IEEE Standards Board and published by the IEEE* is expected to handle all forms of HSDL.7 described in previous versions of this standard *approved by the IEEE Standards Board and published by the IEEE*.

An HSDL.7 parser is expected to use the Standard HSDL.7 Package content. It is also expected to reject and/or issue warning/error messages when it encounters unrecognized text. It is further expected to perform the semantic checks identified in this standard.

### **31.6 Relationship of HSDL.7 to BSDL.7 (and BSDL.1)**

#### **31.6.1 Description**

HSDL.7 is defined as a derivative, yet hierarchically superior, superset/subset of BSDL.7 (and BSDL.1).

The hierarchically superior aspect of HSDL.7 versus BSDL.7 (and BSDL.1) means that a given HSDL.7 description can invoke/incorporate one or more BSDL.7 (or BSDL.1) descriptions by reference. The superset aspect of HSDL.7 is adopted so as to effectively inherit the relevant BSDL.7 (or BSDL.1) content that is expressly suited to the test application rather than defining it anew. The subset aspect of HSDL.7 is adopted as many of the attributes required to describe test endpoints do not pertain to test modules.

## 31.6.2 Specifications

### Rules

- a) HSDL.7 shall conform to all rules set out for BSDL.7 (and BSDL.1) except as expressly specified in this clause.

## 31.7 Lexical elements of HSDL.7

### 31.7.1 Description

The lexical elements of BSDL.7 are based on those specified for BSDL.7 with a small number of additions.

### 31.7.2 Specifications

### Rules

- a) The lexical elements of HSDL.7 shall be those specified for BSDL.7 (and BSDL.1) excepting the addition of a small number of reserved words as specified in 31.8.2.

## 31.8 HSDL.7 reserved words

### 31.8.1 Description

The reserved words of HSDL.7 are specified in 31.8.2. These reserved words (identifiers) cannot be used for any other purposes in an HSDL.7 description, other than as part of a comment. For example, a reserved word cannot be used as an explicitly declared identifier.

### 31.8.2 Specifications

### Rules

- a) The identifiers listed in Table 31-2 shall be HSDL.7 reserved words having a fixed significance in the language.
- b) These identifiers listed in Table 31-2 shall not be used for any other purposes in an HSDL.7 description, other than as part of a comment.

**Table 31-2 — HSDL.7 reserved words**

COMPONENT_CONFORMANCE_MODULE MEMBERS MEMBERS_NODEIDS	MEMBERS_PORTMAP STD_1149_7_2009_module
--	---

## 31.9 Components of an HSDL.7 description

### 31.9.1 Description

An HSDL.7 description is composed in the same fashion as for a BSDL.7 description with the exception that the HSDL.7 description requires two Standard HSDL.7 Packages (the module superior and the subordinate; see 31.10.2.1.2).

### 31.9.2 Specifications

#### Rules

- a) The IEEE 1149.7-Specified Behavior of all test modules shall match the behavior described in associated documentation using the HSDL.7 specified in this clause.
- b) An HSDL.7 description shall be composed in the same fashion as for a BSDL.7 description with the exception that the Standard HSDL.7 Packages shall be provided.

### 31.10 The entity description (HSDL.7)

The entity description and supporting HSDL.7 packages comprise an HSDL.7 model of the component, which is, in effect, the electronic data sheet for its test logic. It contains elements through which parameters that may vary from one chip to another are defined, as discussed in 31.4.

As indicated in Clause 29, a test module is defined as a collection of test endpoints and other test modules where TAPs are connected to define a scan topology. Since test modules can contain other test modules as subcomponents, a hierarchy is formed (hence the name Hierarchical Scan Description Language). The simplest test module contains one or more devices connected into a single scan topology. More complicated test modules can represent multi-chip modules, etc.

A test module is described by way of HSDL.7 using much of the same syntax as that used to describe a test endpoint by way of BSDL.7 (or BSDL.1). New statements have been added to list the subcomponents of the test module and to describe how these subcomponents are interconnected. Existing BSDL.7 (and/or BSDL.1) statements have been removed where they did not apply to the description of test modules.

#### 31.10.1 Overall structure of the entity description (HSDL.7)

##### 31.10.1.1 Syntax and content

###### 31.10.1.1.1 Description

The overall structure of the entity description (HSDL.7) is governed by 31.10.1.1.2. While VHDL permits some elements within an entity description to be in an arbitrary order, fixed ordering is required for HSDL.7. This ordering is defined to ease the development of tools that are not themselves required to be fully VHDL compliant.

###### 31.10.1.1.2 Specifications

#### Rules

- a) The entity description and supporting HSDL.7 packages shall make up an HSDL.7 model of the component.
- b) The HSDL.7 entity description shall have the following structure:

```
<HSDL.7 description> ::=  
entity <component name> is  
<generic parameter>                                (see IEEE Std 1149.1)  
<logical port description>                         (see IEEE Std 1149.1)  
<module standard use statement.7>                  (see 31.10.2)  
{<use statement>}                                    (see IEEE Std 1149.1)  
<module component conformance statement.7>        (see 31.10.4)  
<module package pin mappings.7>                   (see 31.10.5)
```

<module scan port identification.7> (see 31.10.6)  
<module members declaration.7> (see 31.10.7)  
**end** <component name>;

<component name> ::= <VHDL identifier> (see IEEE Std 1149.1)

- c) For HSDL.7, the ordering of statements shall be fixed as shown in Rule 31.10.1.1.2 b).
- d) The elements of the <HSDL.7 description> shall be the same as those for the <BSDL.7 description> with the following exceptions:
  - 1) The <module standard use statement.7> shall replace the <standard use statement.7>.
  - 2) The <module component conformance statement.7> shall replace the <component conformance statement>.
  - 3) The <module package pin mappings.7> shall replace the <device package pin mappings>.
  - 4) The <module scan port identification.7> shall replace the <scan port identification>.
  - 5) The <module member declaration.7> shall be a newly inserted element.
- e) The <component name> shall identify a particular type of component (e.g., part number).

### Recommendations

- f) The <component name> should be distinct from the names of all other types of components that may be used together (i.e., that may be assembled onto a single loaded board).

## 31.10.1.2 Semantic checks

### 31.10.1.2.1 Description

The expected semantic checks related to the overall structure of the entity description (HSDL.7) are specified in 31.10.1.2.2.

### 31.10.1.2.2 Specifications

#### Rules

- a) Any <component name> referenced in any attribute statement shall be the same as the component name declared in the entity description.

## 31.10.2 Module standard use statement (HSDL.7)

### 31.10.2.1 Syntax and content

#### 31.10.2.1.1 Description

The syntax and content of the module standard use statement is governed by 31.10.2.1.2. It identifies the Standard HSDL.7 Package in which attributes, types, constants, and other elements are defined and can be referenced elsewhere in the HSDL.7 description.

The construct of the <module superior use statement.7> or <subordinate use statement.7> may not be syntactically complete for use by a given VHDL analyzer. This is because a library or default working area has not been specified. In such a case, a complete statement is “use work.STD\_1149\_7\_2009\_module.all;” in which the prefix “work.” tells a VHDL analyzer to find the Standard HSDL.7 Package in the current work area. The field “work.” could be replaced by an arbitrary library name such as “BSCAN.” telling a VHDL analyzer where in its system of libraries to find the Standard HSDL.7 Package. Since there is no

standardization of library structures from one VHDL environment to another, some editing of HSDL.7 files to specify the location of Standard HSDL.7 Packages is generally unavoidable. The specification used in this subclause may cause an error in a VHDL analyzer, forcing the user to edit the HSDL.7 file for the correct location of the Standard HSDL.7 Package information.

As a general rule, future enhancements to the HSDL.7 language will be designed as extensions to previous versions of the language. Therefore, the version information contained in the <module superior HSDL package identifier.7> may be used by HSDL.7-specific tools to indicate which tool versions will be able to process the information in the input file. For example, if the input file records the version as “**\_2009\_module**”, then tools that can process any language variant “**\_2009\_module**” or later will be able to process the input file correctly. Versions of the HSDL.7 language are not to be confused with the version of this standard that a given IC may implement. The <component conformance statement.7> identifies the version of the standard (see 31.10.4).

Within a specific system processing HSDL.7 descriptions, the Standard HSDL.7 Packages may be files somewhere in the file system of the host computer. The **.all** suffix is meaningful to VHDL and is not part of a file name.

### 31.10.2.1.2 Specifications

#### Rules

- a) The <module standard use statement.7> shall identify the Standard HSDL.7 Package in which attributes, types, constants, and other elements are defined that shall be referenced elsewhere in the HSDL.7 description.
- b) The <module standard use statement.7> shall identify the following syntax:

<module standard use statement.7> ::= <module superior use statement.7> <subordinate use statement.7>

<module superior use statement.7> ::= **use** <module superior HSDL package identifier.7> **.all** ;

<subordinate use statement.7> ::= **use** <subordinate HSDL package identifier.7> **.all** ;

<module superior HSDL package identifier.7> ::=

**STD\_1149\_7\_2009\_module** | <other package identifier.7>

<subordinate HSDL package identifier.7> ::=

<standard BSDL package identifier>

(see IEEE Std 1149.1)

<other package identifier.7> ::=

<VHDL identifier>

(see IEEE Std 1149.1)

- c) The <module superior HSDL package identifier.7> and <subordinate HSDL package identifier.7> shall be the names of the Standard HSDL.7 Packages that contain the information to be included.
- d) The suffix **.all** shall indicate that all declarations within the HSDL.7 packages are to be used.
- e) While VHDL permits the use of a wider range of suffixes, **.all** shall be the only suffix permitted in HSDL.7.
- f) The <module superior use statement.7> shall indicate:
  - 1) An instruction to tools to read a standard package that contains HSDL.7 syntax definitions.
  - 2) The version of the HSDL.7 language (shown by the year number embedded in the identifier) that was used when creating the HSDL.7 file.

- g) The content of the Standard HSDL.7 Packages shall be the current definition of the HSDL.7 language and shall not be modified by users.
- h) The <module standard use statement.7> shall appear in every HSDL.7 description before any <use statement> (see IEEE Std 1149.1) so that tools that are fully VHDL compliant can locate information relevant to all components that implement this standard. (Tools that are limited to the HSDL.7 language shall always use this information).

## Permissions

- i) The <module standard use statement.7> shall appear in every HSDL.7 description before any <use statement> (see IEEE Std 1149.1) so that tools that are fully VHDL compliant can locate information relevant to all components that implement this standard. (Tools that are limited to the HSDL.7 language shall always use this information).

### 31.10.2.2 Examples

```
use STD_1149_7_2009_module.all;
use STD_1149_1_2013.all; -- Today
```

or

```
use STD_1149_7_2093_module.all;
use STD_1149_1_2091.all; -- Sometime in the future
```

The contents of the **STD\_1149\_7\_2009\_module** Standard HSDL.7 Package are listed in 31.11.

The contents of the **STD\_1149\_1\_2013** Standard HSDL.7 Package and its associated Standard HSDL.7 Package Body are specified by IEEE Std 1149.1-2013 (wherein they are referred to as Standard BSDL Package and Standard BSDL Package Body).

## 31.10.3 Version control

### 31.10.3.1 Syntax and content

#### 31.10.3.1.1 Description

The additional application of the <module superior use statement.7> as specified in 31.10.3.1.2 is intended to provide backward compatibility to all HSDL.7 descriptions already written and can be used in a similar manner for HSDL.7 descriptions based on future revisions of this standard.

#### 31.10.3.1.2 Specifications

##### Rules

- a) The <module superior use statement.7> shall indicate whether the HSDL.7 description has been written using the syntax defined in this clause or in some future version.
- b) Version control as pertains to the <subordinate use statement.7> defined by this standard shall be as per the <standard use statement> defined by IEEE Std 1149.1.

### 31.10.4 Module component conformance statement (HSDL.7)

#### 31.10.4.1 Syntax and content

##### 31.10.4.1.1 Description

The module component conformance statement.7 is governed 31.10.4.1.2.

In general, it is possible for a component designed prior to the writing of this clause to be described by the version of HSDL.7 defined herein, but this cannot imply that the component implements the rules of this standard. The <module component conformance statement.7> allows tools to account for changes in the rules that may occur in future editions of this standard versus past editions or the current edition.

##### 31.10.4.1.2 Specifications

###### Rules

- a) The <module component conformance statement.7> shall identify the edition of this standard for which the testability circuitry of a physical component has been implemented.
- b) The <module component conformance statement.7> shall identify the following syntax:

```
<module component conformance statement.7>::=  
attribute COMPONENT_CONFORMANCE_MODULE of  
    <component name> : entity is <module conformance string.7>;  
  
<module conformance string.7>::= "<module conformance identification.7>"  
  
<module conformance identification.7>::=  
    STD_1149_7_2009
```

- c) The symbol **STD\_1149\_7\_2009** shall refer to this standard.

###### Permissions

- d) Subsequent editions of this standard may add new values to the <module conformance identification.7> element.

NOTE—Although this edition of this standard is subsequent to the original issue in 2009, it has been deemed unnecessary to add new values to the <module conformance identification.7> element since none of the normative text has changed in any way that would benefit by assertion of such new values.

#### 31.10.4.2 Examples

```
attribute COMPONENT_CONFORMANCE_MODULE of My_Dot7_Module : entity is  
    "STD_1149_7_2009"; -- invokes 2009 rules for IEEE Std 1149.7
```

#### 31.10.4.3 Semantic checks

No semantic check is necessary for this statement. However, some semantic checks described in future versions of this standard may be influenced by values that appear in the <module conformance identification.7> element.

### **31.10.5 Module package pin mappings (HSDL.7)**

### **31.10.5.1 Syntax and content**

### **31.10.5.1.1 Description**

The mapping of logical pins onto the physical pins of a particular module package is governed by 31.10.5.1.2.

### **31.10.5.1.2 Specifications**

## Rules

- a) The mapping of logical signals onto the physical pins of a particular module package shall be defined through use of an attribute statement and an associated HSDL.7 string constant.
  - b) The <module package pin mappings.7> shall identify the following syntax:

<module package pin mappings.7>::=

<pin map statement> <pin mappings> (see IEEE Std 1149.1)

### **31.10.5.2 Examples**

```

attribute PIN_MAP of My_Dot7_Module : entity is PHYSICAL_PIN_MAP;
constant MCM : PIN_MAP_STRING :=
    "CLK:1, Q:(2,3,4,5,7,8,9,10), " &
    "D:(23,22,21,20,19,17,16,15), " &
    "GND:6, VCC:18, OC_NEG:24, " &
    "TDO:11:TMS:12, TCK:13, TDI:14";

```

### 31.10.5.3 Semantic checks

### **31.10.5.3.1 Description**

The expected semantic checks related to module package pin mappings are specified in 31.10.5.3.2.

### **31.10.5.3.2 Specifications**

## Rules

- a) Semantics checks for <module package pin mappings.7> in an HSDL.7 description shall be as those for <device package pin mappings> in a BSDL description, noting the change in context (where the term “BSDL description” appears, interpret it as “HSDL.7 description”).

### **31.10.6 Module scan port identification (HSDL.7)**

### **31.10.6.1 Syntax and content**

### **31.10.6.1.1 Description**

Test endpoints have a single TAP (TAP.7 or TAP.1) by definition. Test modules, however, can have one TAP or many TAPs (TAP.7s and/or TAP.1s) and associated scan paths. Different scan paths may actually share some of the signals of their TAPs. As such, the attributes of the module scan port identification are

used to identify every port that can be connected to a member device or module. This is governed by 31.10.6.1.2.

### **31.10.6.1.2 Specifications**

#### **Rules**

- a) The module scan port identification statements shall define the TAP.7(s) and/or TAP.1(s) of the test module.
- b) The <module scan port identification.7> shall identify the following syntax:

<module scan port identification.7>::=  
     <scan port identification list.7> { <scan port identification list.7> } (see 30.10.5)

- c) The statements shall identify the specific logical signals of the component as being signals of the TAP.7. Note that for test modules, multiple logical TAP.7s (and/or TAP.1s) can be described.

#### **Recommendations**

- d) Note that the maximum frequencies defined for the test module should be determined not only by the specified maximum frequencies for the member devices, but also as a function of the module characteristics where these involve additional performance constraints.

### **31.10.6.2 Examples**

An example for a test module with two test endpoints—one having a TAP.1 and one having a T5(N) TAP.7, neither implementing the optional Test Reset Signal—where no TAP signals are shared:

```

port (TDI_1 : in bit;
      TDO_1 : out bit;
      TMS_1 : in bit;
      TCK_1 : in bit;
      TMS_TMSC_2 : inout bit;
      TCK_TCKC_2 : in bit;

      (... some BSDL deleted for brevity ...)

attribute TAP_SCAN_IN of TDI_1 : signal is true;
attribute TAP_SCAN_OUT of TDO_1 : signal is true;
attribute TAP_SCAN_MODE of TMS_1 : signal is true;
attribute TAP_SCAN_CLOCK of TCK_1 : signal is (10.0e6, BOTH);

attribute TAP_SCAN_MODE of TMS_TMSC_2 : signal is true;
attribute TAP_SCAN_CLOCK of TCK_TCKC_2 : signal is (20.0e6, BOTH);
attribute TAP_SCAN_MODE_ADVANCED of TMS_TMSC_2 : signal is true;
attribute TAP_SCAN_CLOCK_ADVANCED of TCK_TCKC_2 : signal is
      (25.0e6, 35.0e6);

```

An example for a test module with three test endpoints—one having a T3 TAP.7 that implements the optional nTRST, one having a T4W TAP.7 that implements AUX1 and AUX2 system functions at TDIC and TDOC and that implements the optional nTRST\_PD signal, and one having a T5(N) TAP with no Test Reset Signal—where some TAP signals are shared:

```

port (TDI_TDIC_1 : in bit;
      TDO_TDOC_1 : out bit;

```

```

nTRST_1 : in bit;
TDI_TDIC_AUX1_2 : inout bit;
TDO_TDOC_AUX2_2 : inout bit;
nTRST_PD_2 : in bit;
TMS_TMSC : inout bit;
TCK_TCKC : in bit;

(... some BSDL deleted for brevity ...)

attribute TAP_SCAN_IN of TDI_TDIC_1 : signal is true;
attribute TAP_SCAN_OUT of TDO_TDOC_1 : signal is true;
attribute TAP_SCAN_RESET of nTRST_1 : signal is true;
attribute TAP_SCAN_IN_COMPACT of TDI_TDIC_1 : signal is true;
attribute TAP_SCAN_OUT_COMPACT of TDO_TDOC_1 : signal is true;

attribute TAP_SCAN_IN of TDI_TDIC_AUX1_2 : signal is true;
attribute TAP_SCAN_OUT of TDO_TDOC_AUX2_2 : signal is true;
attribute TAP_SCAN_RESET_PD of nTRST_PD_2 : signal is true;
attribute TAP_SCAN_IN_COMPACT of TDI_TDIC_AUX1_2 : signal is true;
attribute TAP_SCAN_OUT_COMPACT of TDO_TDOC_AUX2_2 : signal is true;

attribute TAP_SCAN_MODE of TMS_TMSC : signal is true;
attribute TAP_SCAN_CLOCK of TCK_TCKC : signal is (20.0e6, BOTH);
attribute TAP_SCAN_MODE_COMPACT of TMS_TMSC : signal is true;
attribute TAP_SCAN_CLOCK_COMPACT of TCK_TCKC : signal is
(25.0e6, 35.0e6);

```

The port IDs used in the above examples closely match the signal names defined in this standard; however, arbitrary names could have been used.

NOTE 1—The meaning of each signal in relation to this standard is to be deduced from the attribute name, not the value of a <port ID> element in the <module scan port identification.7>.

### **31.10.6.3 Semantic checks**

#### **31.10.6.3.1 Description**

The expected semantic checks related to the module scan port identification statements are specified in 31.10.6.3.2.

#### **31.10.6.3.2 Specifications**

##### **Rules**

- a) With respect to the <port ID> elements in the <module scan port identification.7>:
  - 1) The unique <port name> element in a <port ID> occurring in a <TDOC stmt> shall appear as a value in an <identifier list> of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **out** or **inout**.

NOTE 1—Since the <port ID> that occurs in the corresponding <TDO stmt> has the same value as that which occurs in the <TDOC stmt>, it inherits this more permissive check with respect to <pin type> notwithstanding the more restrictive check of IEEE Std 1149.1.

- 2) The unique <port name> element in a <port ID> occurring in a <TDIC stmt> shall appear as a value in an <identifier list> of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **in** or **inout**.

NOTE 2—Since the <port ID> that occurs in the corresponding <TDI stmt> has the same value as that which occurs in the <TDIC stmt>, it inherits this more permissive check with respect to <pin type> notwithstanding the more restrictive check of IEEE Std 1149.1.

- 3) The unique <port name> element in a <port ID> occurring in a <TMSC stmt.7> shall appear as a value in an <identifier> list of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **inout**.
  - 4) The unique <port name> element in a <port ID> occurring in a <TCKC stmt.7> or a <TRST<sub>PD</sub> stmt.7> shall appear as a value in an <identifier list> of a <pin spec> in the <logical port description>. Furthermore, the value of <pin type> on that <pin spec> shall be **in**.
  - 5) Any <port ID> appearing in the <module scan port identification.7> shall be a <subscripted port name> or a <port name> defined in a <logical port description> statement with a corresponding <port dimension> equal to **bit**.
- b) A given <port ID> shall occur at most once in the <module scan port identification.7> excepting all of the following:
- 1) if a <TCKC stmt.7> is present, the <port ID> that occurs in it shall also occur in a <TCK stmt>, and
  - 2) if a <TMSC stmt.7> is present, the <port ID> that occurs in it shall also occur in a <TMS stmt>, and
  - 3) if a <TDIC stmt.7> is present, the <port ID> that occurs in it shall also occur in a <TDI stmt>, and
  - 4) if a <TDOC stmt.7> is present, the <port ID> that occurs in it shall also occur in a <TDO stmt>.
- c) If a <port ID> is a <subscripted port name>, the <subscript> shall lie within the range specified for the **bit\_vector** of the relevant port.

### 31.10.7 Module members declaration (HSDL.7)

#### 31.10.7.1 Syntax and content

##### 31.10.7.1.1 Description

The declaration of all the members of the test module is described in this subclause. Members represent subcomponents—devices or other test modules—that comprise the test module. Most often members represent test endpoints such that the corresponding member declaration invokes a BSDL.7 (or BSDL.1) description, but some test modules may contain scannable subcomponents that are themselves test modules such that the corresponding member declaration invokes an HSDL.7 description.

The member declaration names all the subcomponents mounted on the test module, selecting an entity and a packaging option for each. There is no need to indicate whether a given member is a test endpoint or a test module—this will be determined by the HSDL.7 translation during the reading the member's entity description.

Following the simple member instantiations, further statements declare the interconnection of the member signals among each other and to the signals of the logical port defined for the test module and declare the member addressability (where required for test endpoints that implement a T3 and above TAP.7).

### 31.10.7.1.2 Specifications

#### Rules

- a) The <module members declaration.7> shall identify the following syntax:

```

<module members declaration.7>::=
    <module members instantiation.7>
    <module members port map.7>
    [ <module members addressability.7> ]

<module members instantiation.7>::=
    attribute MEMBERS of <component name> : entity is
        " <module members table.7> " ;

<module members table.7>::=
    <module member entry.7> { , <module member entry.7> }

<module member entry.7>::=
    <module member name.7>
    ( <member entity name.7> , <member pin mapping name or default.7> )

<member pin mapping name or default.7>::= <member pin mapping name.7> | *

<module members port map.7>::=
    attribute MEMBERS_PORTMAP of <component name> : entity is
        " <module members portmap table.7> " ;

<module members portmap table.7>::=
    <module member portmap entry.7> { , <module member portmap entry.7> }

<module member portmap entry.7>::=
    <module member name.7> ( <module member port mapping.7>
        { , <module member port mapping.7> } )

<module member port mapping.7>::=
    <module member port.7> => <module top port.7>

<module member port.7>::=
    <port ID>                                (see IEEE Std 1149.1)

<module top port.7>::=
    <port ID>                                (see IEEE Std 1149.1)

<module members addressability.7>::=
    attribute MEMBERS_NODEIDS of <component name> : entity is
        " <module members addresses table.7> " ;

<module members addresses table.7>::=
    <module member address entry.7> { , <module member address entry.7> }

<module member address entry.7>::=
    <hierarchical module member name.7> ( <module member address.7> )

<hierarchical module member name.7>::=
    <module member name.7> { . <subordinate module member name.7> }

```

NOTE 1—Whitespace has been omitted in the definition of <hierarchical module member name.7> to indicate that this element is to be treated as a lexical atom, and, as such, there is to be no whitespace within a literal value for any given instance of this element. For example, while `SIP1.IC1` represents a proper such value, `SIP1 IC1` does not.

<module member address.7>::=  
     <8-bit unambiguous pattern.7>

NOTE 2—The <8-bit unambiguous pattern.7> is to be a <pattern> (see IEEE Std 1149.1) that is a contiguous sequence of exactly eight characters, none of which are **X**.

<module member name.7>::=	
<VHDL identifier>	(see IEEE Std 1149.1)
<member entity name.7>::=	
<VHDL identifier>	(see IEEE Std 1149.1)
<member pin mapping name.7>::=	
<VHDL identifier>	(see IEEE Std 1149.1)
<subordinate module member name.7>::=	
<VHDL identifier>	(see IEEE Std 1149.1)

- b) Where the asterisk (\*) is used in a <member pin mapping name or default.7>, the <default device package type> given in the <generic parameter> of the member entity description shall be taken as the effective value of the <member pin mapping name of default.7>. Of course, where no package default is defined in the member entity description, an error will result (see IEEE Std 1149.1).

### 31.10.7.2 Examples

An example for a test module with two test endpoints—one having a TAP.1 and one having a T5 TAP.7, neither implementing the optional Test Reset Signal—where no TAP signals are shared:  
 (see corresponding example 31.10.6.2 for associated logical port description and scan port identification)/

```
attribute MEMBERS of My_Dot7_Module_1 : entity is
  "U1 (My_Dot1_IC, *), " &
  "U2 (My_Dot7_IC_T5N, *)";

attribute MEMBERS_PORTMAP of My_Dot7_Module_1 : entity is
  "U1 (TDI => TDI_1, TDO => TDI_1, TMS => TMS_1, TCK => TCK_1), " &
  "U2 (TMS_TMSC => TMS_TMSC_2, TCK_TCKC => TCK_TCKC_2)";

attribute MEMBERS_NODEIDS of My_Dot7_Module_1 : entity is
  "U2 (00000010);
```

An example for a test module with three test endpoints—one having a T3 TAP.7 that implements the optional nTRST one having a T4W TAP.7 that implements AUX1 and AUX2 system functions at TDIC and TDOC and that implements the optional nTRST\_PD signal, and one having a T5(N) TAP with no Test Reset Signal—where some TAP signals are shared:

(see corresponding example 31.10.6.2 for associated logical port description and scan port identification)

```
attribute MEMBERS of My_Dot7_Module_2 : entity is
  "U11 (My_Dot1_IC_T3, *), " &
  "U12 (My_Dot1_IC_T4W, *), " &
  "U13 (My_Dot7_IC_T5N, *)";

attribute MEMBERS_PORTMAP of My_Dot7_Module_2 : entity is
  "U11 (TDI => TDI_1, TDO => TDI_1, " &
  "TMS => TMS_1, TCK => TCK_TCKC), " &
```

```
"U12 (TDI_TDIC_AUX1 => TDI_TDIC_AUX1_2, " &
      TDO_TDOC_AUX2 => TDO_TDOC_AUX2_2, " &
      TMS_TMSC => TMS_TMSC, TCK_TCKC => TCK_TCKC), " &
"U13 (TMS_TMSC => TMS_TMSC, TCK_TCKC => TCK_TCKC)";

attribute MEMBERS_NODEIDS of My_Dot7_Module_2 : entity is
  "U11 (00001011), " &
  "U12 (00001100), " &
  "U13 (00001101)";
```

### 31.10.7.3 Semantic checks

#### 31.10.7.3.1 Description

The expected semantic checks related to the module scan port identification statements are specified in 31.10.7.3.2.

#### 31.10.7.3.2 Specifications

##### Rules

- a) Every <member entity name.7> appearing in a <module member entry.7> shall have a value that likewise appears as the value of a <component name> in a separate <HSDL.7 description> or in a <BSDL.7 description> or <BSDL description>. Furthermore:
  - 1) For a <member pin mapping name.7> appearing as the value of <member pin mapping name or default.7> in the <module member entry.7>, it shall have a value that likewise appears as the value of a <pin mapping name> in the same such separate <HSDL.7 description> or in a <BSDL.7 description> or <BSDL description>; or
  - 2) For an \* appearing as the value of <member pin mapping name or default.7> in the <module member entry.7>, the same such separate <HSDL.7 description>, <BSDL.7 description>, or <BSDL description> shall present a value for <default device package type> and that value shall likewise appear as the value of <pin mapping name> in its <pin mappings>.
- b) A given <port ID> shall occur at most once among all <module member port.7> elements in a given <module member portmap entry.7>.
- c) A given <port ID> shall occur at most once among all <module top port.7> elements in a given <module member portmap entry.7>.
- d) With respect to the <port ID> elements in a <module member portmap entry.7>, if a <port ID> is a <subscripted port name>, the <subscript> shall lie within the range specified for the **bit\_vector** of the relevant port.
- e) With respect to the <port ID> elements in a given <module member port mapping.7>, these <port ID> elements:
  - 1) Both shall be either <subscripted port name> elements or <port name> elements defined in a <logical port description> statement with a corresponding <port dimension> equal to **bit**; or,
  - 2) Both shall be <port name> elements defined in a <logical port description> statement with a corresponding <port dimension> equal to **bit\_vector** and these defined ports shall have ranges of equal length.

NOTE—Module top ports in the **MEMBERS\_PORTMAP** need not appear in the module logical port description—where they do not, they are to be considered internal.

- f) A <module member address.7> shall appear in a <module members addressability.7> only where the corresponding <hierarchical module member name.7> identifies a test endpoint described with a <scan port tap class.7> having value T3, T4N, T4W, T5N, or T5W.
- g) Where <hierarchical module member name.7> elements identify test endpoints that, first, are connected with common TCKC and TMSC signals and, second, are described with the same <component name>, the corresponding <module member address.7> elements shall be mutually unique.

## 31.11 The Standard HSDL.7 Package STD\_1149\_7\_2009\_module

### 31.11.1 Description

This package is used by HSDL.7 to define the extra attributes and subtypes used in HSDL.7 entities. The package contents are not used by an HSDL.7 translator but are necessary so that an HSDL.7 description can be used successfully in a VHDL environment.

This information shall define the basis of HSDL.7 and would typically be write-protected by a system administrator. HSDL.7 descriptions that use the <module superior HSDL package identifier.7> STD\_1149\_7\_2009\_module shall be processed using this Standard HSDL.7 Package.

Note that only the package is needed for HSDL.7; no package body is required.

### 31.11.2 Specifications

#### Rules

- a) The following shall be the complete content of Standard HSDL.7 Package STD\_1149\_7\_2009\_module:

```
package STD_1149_7_2009_module is

-- Member Declarations
attribute MEMBERS : string;
attribute MEMBERS_PORTMAP : string;
attribute MEMBERS_NODEIDS : string;

end STD_1149_7_2009_module;
```

## 31.12 Applications of HSDL.7

### 31.12.1 HSDL.7 for IEEE 1149.1 serial connection using one TMS signal

This HSDL.7 example describes an IEEE 1149.1 serial connection using one TMS signal shown in Figure 31-1. This is a simple Series Scan Topology of four devices (T0–T2 TAP.7 equivalent).

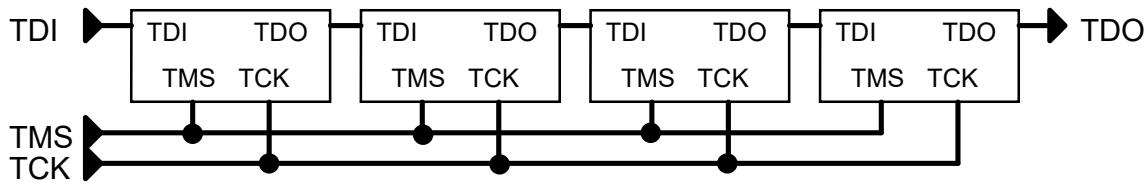


Figure 31-1 — Serial connection using one TMS signal

```
entity serial_chain_1x4 is
  -- Generic Parameter
  generic (PHYSICAL_PIN_MAP : string := "UNDEFINED");

  -- Logical Port Description
  port (TDI      : in bit;
        TDO      : out bit;
        TMS, TCK : in bit);

  -- Use Statement
  use STD_1149_7_2009_module.all;
  use STD_1149_1_2013.all;

  -- Component Conformance Statement
  attribute COMPONENT_CONFORMANCE_MODULE of serial_chain_1x4 : entity is
    "STD_1149_7_2009";

  -- Package Pin Mapping
  attribute PIN_MAP of serial_chain_1x4 : entity is PHYSICAL_PIN_MAP;
  constant ONLY_PACKAGE : PIN_MAP_STRING :=
    "TDI:1, TDO:2, TMS:3, TCK:4";

  -- Scan Port Identification
  attribute TAP_SCAN_IN    of TDI  : signal is true;
  attribute TAP_SCAN_OUT   of TDO  : signal is true;
  attribute TAP_SCAN_MODE  of TMS  : signal is true;
  attribute TAP_SCAN_CLOCK of TCK  : signal is (20.0e6, BOTH);

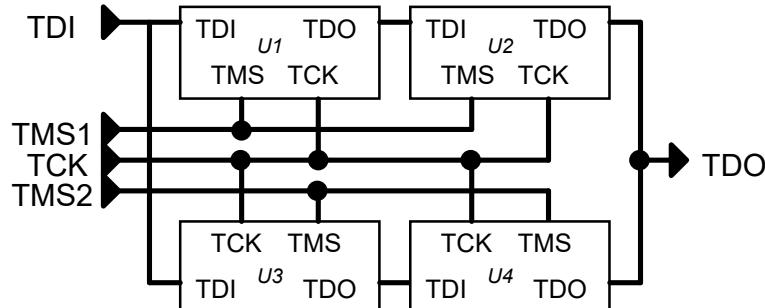
  -- Members Declaration
  attribute MEMBERS of serial_chain_1x4 : entity is
    "u1 (my_chip, my_pkg), &
     u2 (my_chip, my_pkg), &
     u3 (my_chip, my_pkg), &
     u4 (my_chip, my_pkg) ";

  attribute MEMBERS_PORTMAP of serial_chain_1x4 : entity is
    "u1 (TDI => TDI," &
      " TDO => TDO1_TDI2," &
      " TMS => TMS, TCK => TCK), " &
    "u2 (TDI => TDO1_TDI2," &
      " TDO => TDO2_TDI3," &
      " TMS => TMS, TCK => TCK), " &
    "u3 (TDI => TDO2_TDI3," &
      " TDO => TDO3_TDI4," &
      " TMS => TMS, TCK => TCK), " &
    "u4 (TDI => TDO3_TDI4," &
      " TDO => TDO," &
      " TMS => TMS, TCK => TCK)";

end serial_chain_1x4;
```

### 31.12.2 HSDL.7 for IEEE 1149.1 connection in two paralleled serial chains

This HSDL.7 example describes an IEEE 1149.1 Connection in two paralleled serial chains shown in Figure 31-2. The four devices (T0–T2 TAP.7 equivalent) are connected in two paralleled serial chains.



**Figure 31-2 — Connection in two paralleled serial chains**

```

entity parallel_chains_2x2 is
    -- Generic Parameter
    generic (PHYSICAL_PIN_MAP : string := "UNDEFINED");

    -- Logical Port Description
    port (TDI          : in  bit;
          TDO          : out bit;
          TMS1, TMS2 : in  bit;
          TCK          : in  bit);

    -- Use Statement
    use STD_1149_7_2009_module.all;
    use STD_1149_1_2013.all;

    -- Component Conformance Statement
    attribute COMPONENT_CONFORMANCE_MODULE of parallel_chains_2x2 : entity is
        "STD_1149_7_2009";

    -- Package Pin Mapping
    attribute PIN_MAP of parallel_chains_2x2 : entity is PHYSICAL_PIN_MAP;
    constant ONLY_PACKAGE : PIN_MAP_STRING :=
        "TDI:1, TDO:2, TMS1:3, TMS2:4, TCK:5";

    -- Scan Port Identification
    attribute TAP_SCAN_IN    of TDI  : signal is true;
    attribute TAP_SCAN_OUT   of TDO  : signal is true;
    attribute TAP_SCAN_MODE  of TMS1 : signal is true;
    attribute TAP_SCAN_MODE  of TMS2 : signal is true;
    attribute TAP_SCAN_CLOCK of TCK  : signal is (20.0e6, BOTH);

    -- Members Declaration
    attribute MEMBERS of parallel_chains_2x2 : entity is
        "u1  (my_chip, my_pkg), &
         u2  (my_chip, my_pkg), &
         u3  (my_chip, my_pkg), &
         u4  (my_chip, my_pkg) ";

```

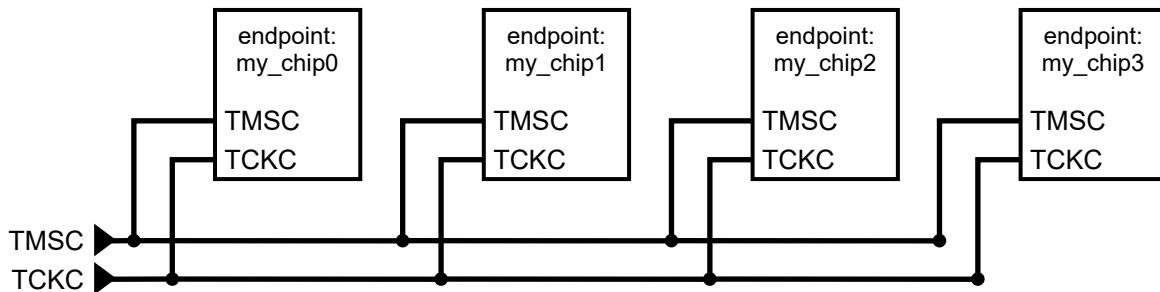
```

attribute MEMBERS_PORTMAP of parallel_chains_2x2 : entity is
  "u1  (TDI => TDI," &
  "  TDO => TDO1_TDI2," &
  "  TMS => TMS1, TCK => TCK), " &
  "u2  (TDI => TDO1_TDI2," &
  "  TDO => TDO," &
  "  TMS => TMS1, TCK => TCK), " &
  "u3  (TDI => TDI," &
  "  TDO => TDO3_TDI4," &
  "  TMS => TMS2, TCK => TCK), " &
  "u4  (TDI => TDO3_TDI4," &
  "  TDO => TDO," &
  "  TMS => TMS2, TCK => TCK) ";
end parallel_chains_2x2;

```

### 31.12.3 HSDL.7 for a basic Star Scan Topology

This HSDL.7 example describes the basic Star Scan Topology shown in Figure 31-3. Four devices [T4(N)/T5(N) TAP.7 equivalent] are connected with common TMSC and TCKC signals.



**Figure 31-3 — Basic Star Scan Topology**

```

entity dot7_basicstar is
  -- Generic Parameter
  generic (PHYSICAL_PIN_MAP : string := "UNDEFINED");

  -- Logical Port Description
  port (TMSC      : inout bit;
        TCKC      : in  bit);

  -- Use Statement
  use STD_1149_7_2009_module.all;
  use STD_1149_1_2013.all;

  -- Component Conformance Statement
  attribute COMPONENT_CONFORMANCE_MODULE of dot7_basicstar : entity is
    "STD_1149_7_2009";

  -- Package Pin Mapping
  attribute PIN_MAP of dot7_basicstar : entity is PHYSICAL_PIN_MAP;
  constant ONLY_PACKAGE : PIN_MAP_STRING :=
    "TMSC:1, TCKC:2";

```

```

-- Scan Port Identification
attribute TAP_SCAN_MODE of TMSC : signal is true;
attribute TAP_SCAN_CLOCK of TCKC : signal is (20.0e6, BOTH);
attribute TAP_SCAN_MODE_COMPACT of TMSC : signal is true;
attribute TAP_SCAN_CLOCK_COMPACT of TCKC : signal is
(25.0e6, 35.0e6);

-- Members Declaration
attribute MEMBERS of dot7_basicstar : entity is
  "u1 (my_chip2, my_pkg2)," &
  "u2 (my_chip2, my_pkg2)," &
  "u3 (my_chip2, my_pkg2)," &
  "u4 (my_chip2, my_pkg2) ";

attribute MEMBERS_PORTMAP of dot7_basicstar : entity is
  "u1 (TMSC => TMSC, TCKC => TCKC), " &
  "u2 (TMSC => TMSC, TCKC => TCKC), " &
  "u3 (TMSC => TMSC, TCKC => TCKC), " &
  "u4 (TMSC => TMSC, TCKC => TCKC)";

attribute MEMBERS_NODEIDS of dot7_basicstar : entity is
  "u1 (00001000)," &
  "u2 (00001100)," &
  "u3 (00001110)," &
  "u4 (00001111)";

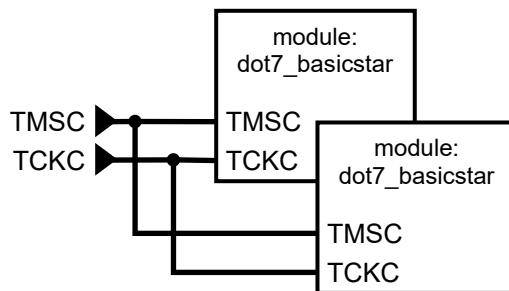
end dot7_basicstar;

```

NOTE—This example illustrates the specification of node IDs at the level in the hierarchy that immediately embraces the respective test endpoints. Where the node IDs for the respective test endpoints are also specified at a higher level in the hierarchy, the latter (higher level) specification would effectively override the lower level specification.

### 31.12.4 HSDL.7 for a basic hierarchical topology

This HSDL.7 example describes a basic hierarchical topology. Two modules that each express the basic Star Scan Topology illustrated in Figure 31-4 are connected with common TMSC and TCKC signals.



**Figure 31-4 — Basic hierarchical topology**

```

entity dot7_basichier is
  -- Generic Parameter
  generic (PHYSICAL_PIN_MAP : string := "UNDEFINED");

```

```
-- Logical Port Description
port (TMSC      : inout bit;
      TCKC      : in  bit);

-- Use Statement
use STD_1149_7_2009_module.all;
use STD_1149_1_2013.all;

-- Component Conformance Statement
attribute COMPONENT_CONFORMANCE_MODULE of dot7_basichier : entity is
  "STD_1149_7_2009";

-- Package Pin Mapping
attribute PIN_MAP of dot7_basichier : entity is PHYSICAL_PIN_MAP;
constant ONLY_PACKAGE : PIN_MAP_STRING :=
  "TMSC:1, TCKC:2";

-- Scan Port Identification
attribute TAP_SCAN_MODE of TMSC  : signal is true;
attribute TAP_SCAN_CLOCK of TCKC  : signal is (20.0e6, BOTH);
attribute TAP_SCAN_MODE_COMPACT of TMSC : signal is true;
attribute TAP_SCAN_CLOCK_COMPACT of TCKC : signal is
  (25.0e6, 35.0e6);

-- Members Declaration
attribute MEMBERS of dot7_basichier : entity is
  "m1 (dot7_basicstar, ONLY_PACKAGE), " &
  "m2 (dot7_basicstar, ONLY_PACKAGE) " ;

attribute MEMBERS_PORTMAP of dot7_basichier : entity is
  "m1 (TMSC => TMSC, TCKC => TCKC), " &
  "m2 (TMSC => TMSC, TCKC => TCKC)" ;

attribute MEMBERS_NODEIDS of dot7_basichier : entity is
  "m1.u1 (00001000), " &
  "m1.u2 (00001100), " &
  "m1.u3 (00001110), " &
  "m1.u4 (00001111), " &
  "m2.u1 (00011000), " &
  "m2.u2 (00011100), " &
  "m2.u3 (00011110), " &
  "m2.u4 (00011111)" ;

end dot7_basichier;
```

**NOTE**—This example illustrates the specification of node IDs at a level in the hierarchy above that which immediately embraces the respective test endpoints. Where the node IDs for the respective test endpoints are also specified at a lower level in the hierarchy, the latter (lower level) specification is effectively overridden by the higher level specification. Since the values of the node IDs specified for the test endpoints on test module m1 are the same as those values specified in the lower level description of test module dot7\_basicstar, their specification at the higher level, as shown here, while permitted, is logically redundant.

### 31.12.5 A typical application of HSDL.7

This HSDL.7 example was written for a typical module containing two endpoints that both express the behavior described in the example BSDL.7 of 30.12. All signals are wired in common except for GND and VCC, which are wired separately, and NC, which is not wired.

```

entity My_Dot7_module is
    generic (PHYSICAL_PIN_MAP: string := "My_PoP");

    port (TDI_TDIC_AUX1 : inout bit;
          TDO_TDOC_AUX2 : inout bit;
          TMS_TMSC : inout bit;
          TCK_TCKC : in bit;
          nTRST_PD : in bit;
          GND_A : power_0 bit_vector (1 to 2);
          VCC_A : power_pos bit_vector (1 to 2);
          GND_B : power_0 bit_vector (1 to 2);
          VCC_B : power_pos bit_vector (1 to 2);
          SYSCK : in bit;
          DQ : inout bit_vector (7 downto 0);
          NC : linkage_mechanical bit_vector (1 to 2);
          ENA_STL : in bit;
          nDIS_STL : in bit;
          STRB : out bit;
          AB : out bit_vector (4 downto 0) );

use STD_1149_7_2009_module.all;
use STD_1149_1_2013.all;

attribute COMPONENT_CONFORMANCE_MODULE of My_Dot7_module :
    entity is "STD_1149_7_2009";

attribute PIN_MAP of My_Dot7_module : entity is PHYSICAL_PIN_MAP;
constant My_PoP : PIN_MAP_STRING :=
    "TDI_TDIC_AUX1 : 1," &
    "TDO_TDOC_AUX2 : 2," &
    "TMS_TMSC : 3," &
    "TCK_TCKC : 4," &
    "nTRST_PD : 5," &
    "GND_A : (6A,21A), VCC_A : (8A,23A)," &
    "GND_B : (6B,21B), VCC_B : (8B,23B)," &
    "SYSCK : 7," &
    "DQ : (9,10,11,12,17,18,19,20)," &
    "NC : (13,15)," &
    "ENA_STL : 14," &
    "nDIS_STL : 16," &
    "STRB : 22," &
    "AB : (28,27,26,25,24)";

```

```

attribute TAP_SCAN_CLOCK of TCK_TCKC : signal is (20.0e6 , BOTH);
attribute TAP_SCAN_MODE of TMS_TMSC: signal is true;
attribute TAP_SCAN_IN of TDI_TDIC_AUX1 : signal is true;
attribute TAP_SCAN_OUT of TDO_TDOC_AUX2: signal is true;
attribute TAP_SCAN_CLOCK_COMPACT of TCK_TCKC : signal is
  ( 27.0e6 , 35.0e6 );
attribute TAP_SCAN_MODE_COMPACT of TMS_TMSC: signal is true;
attribute TAP_SCAN_IN_COMPACT of TDI_TDIC_AUX1 : signal is true;
attribute TAP_SCAN_OUT_COMPACT of TDO_TDOC_AUX2: signal is true;
attribute TAP_SCAN_RESET_PD of nTRST_PD : signal is true;

-- Members Declaration
attribute MEMBERS of My_Dot7_module : entity is
  "u1 (My_Dot7_IC, My_Pkg)," &
  "u2 (My_Dot7_IC, My_Pkg) " ;

attribute MEMBERS_PORTMAP of My_Dot7_module : entity is
  "u1 (TDI_TDIC_AUX1 => TDI_TDIC_AUX1," &
  "     TDO_TDOC_AUX2 => TDO_TDOC_AUX2," &
  "     TMS_TMSC => TMS_TMSC," &
  "     TCK_TCKC => TCK_TCKC," &
  "     nTRST_PD => nTRST_PD," &
  "     GND => GND_A," &
  "     VCC => VCC_A," &
  "     SYSCK => SYSCK," &
  "     DQ      => DQ,      " &
  "     ENA_STL  => ENA_STL,  " &
  "     nDIS_STL => nDIS_STL," &
  "     STRB    => STRB," &
  "     AB      => AB), " &
  "u2 (TDI_TDIC_AUX1 => TDI_TDIC_AUX1,
  "     TDO_TDOC_AUX2 => TDO_TDOC_AUX2," &
  "     TMS_TMSC => TMS_TMSC," &
  "     TCK_TCKC => TCK_TCKC," &
  "     nTRST_PD => nTRST_PD," &
  "     GND => GND_B," &
  "     VCC => VCC_B," &
  "     SYSCK => SYSCK," &
  "     DQ      => DQ,      " &
  "     ENA_STL  => ENA_STL,  " &
  "     nDIS_STL => nDIS_STL," &
  "     STRB    => STRB," &
  "     AB      => AB)";

attribute MEMBERS_NODEIDS of My_Dot7_module : entity is
  "u1 (00010000)," &
  "u2 (00011000)" ;

end My_Dot7_module;

```

## Annex A

(informative)

### IEEE 1149.1 reference material

The state diagram of the IEEE 1149.1 TAPC is shown in Figure A-1.

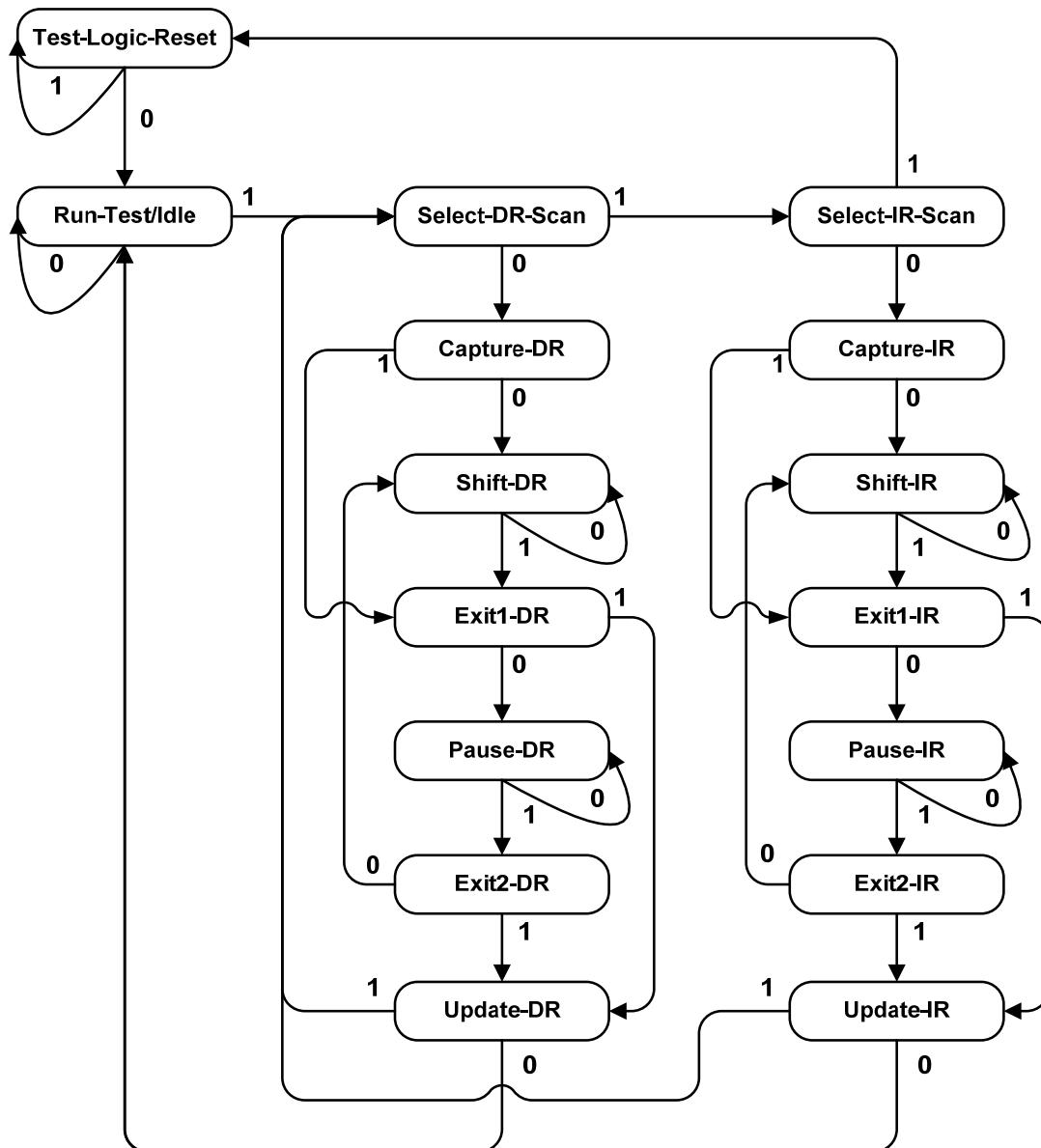
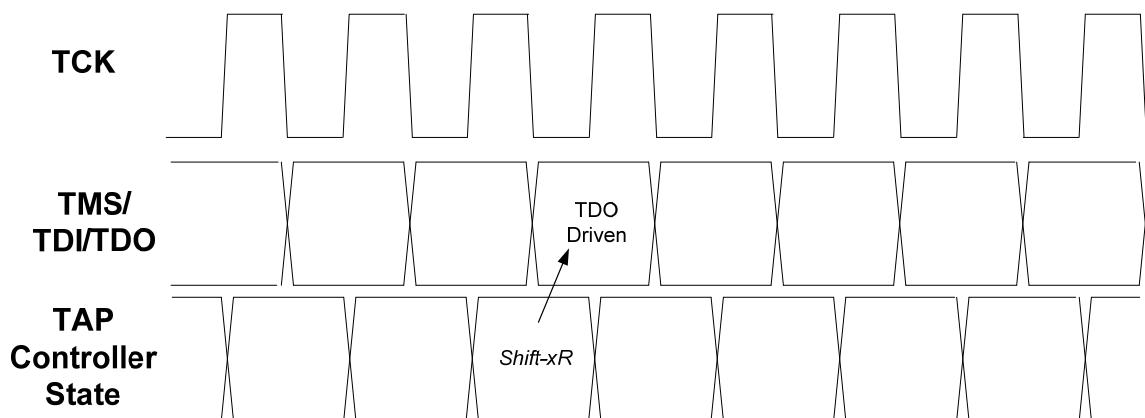


Figure A-1 — IEEE 1149.1 TAPC State Machine

The timing relationships among TCK, TMS, TDI, and TDO are shown in Figure A-2.



**Figure A-2 — IEEE 1149.1 signaling**

The paths in the TAPC state diagram that are used for ZBS generation are illustrated in Figure A-3 and Figure A-4. The thick dark lines in these figures progressing from the *Select-DR* TAPC state to the *Update-DR* TAPC state denote the state machine paths (A and B) used for ZBS generation.

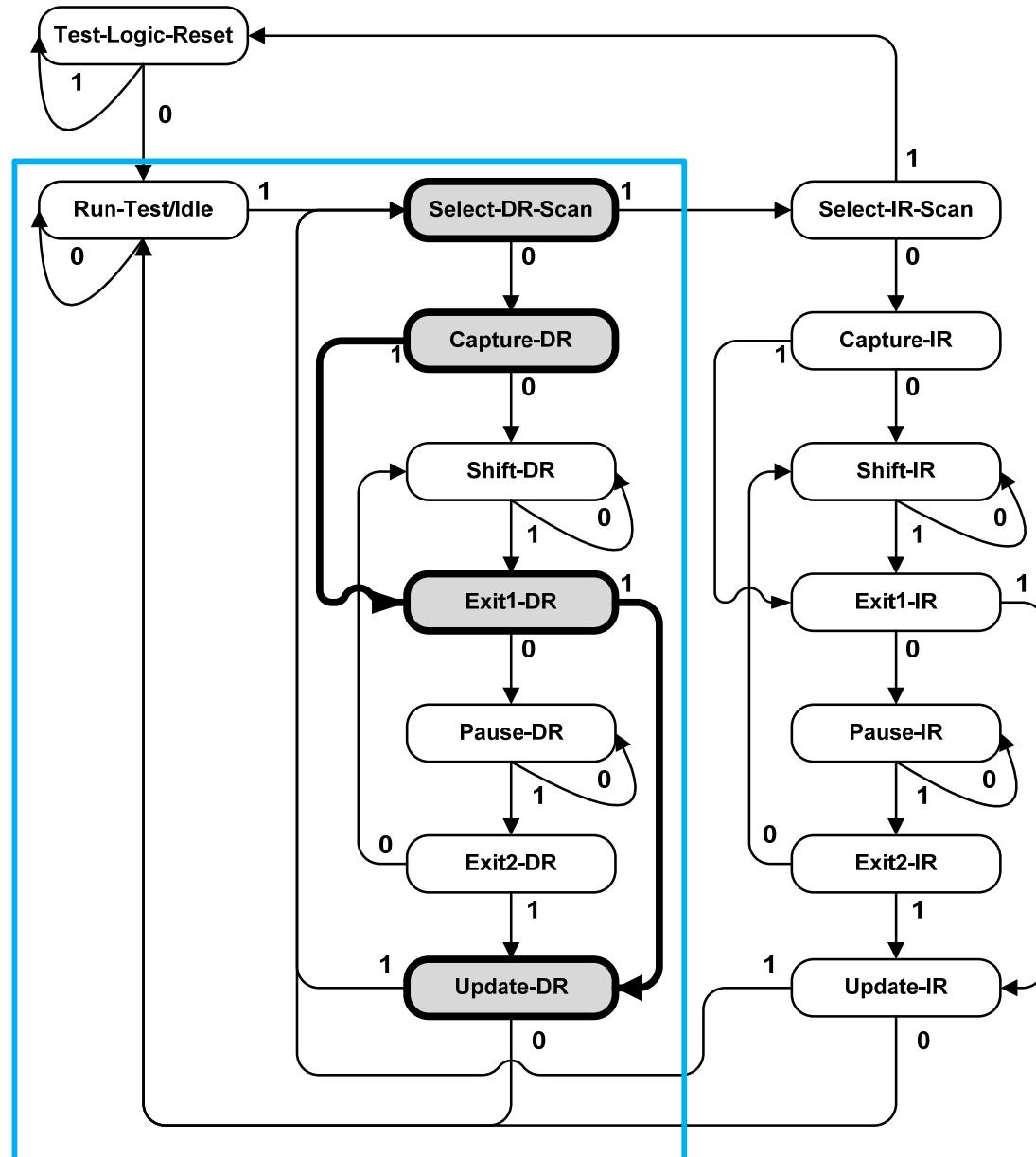


Figure A-3 — Zero-bit DR Scan state sequences, path A

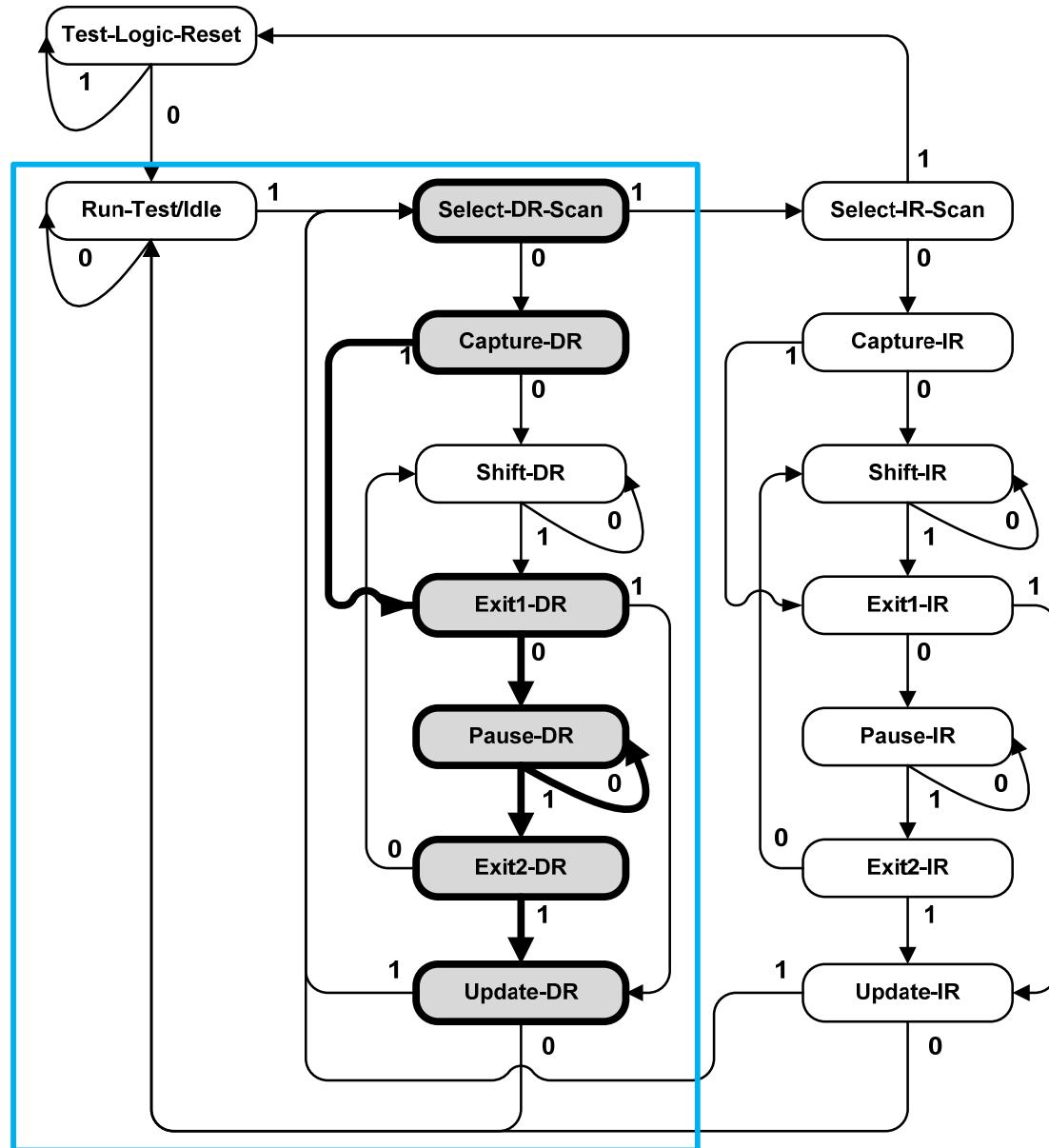


Figure A-4 — Zero-bit DR Scan state sequences, path B

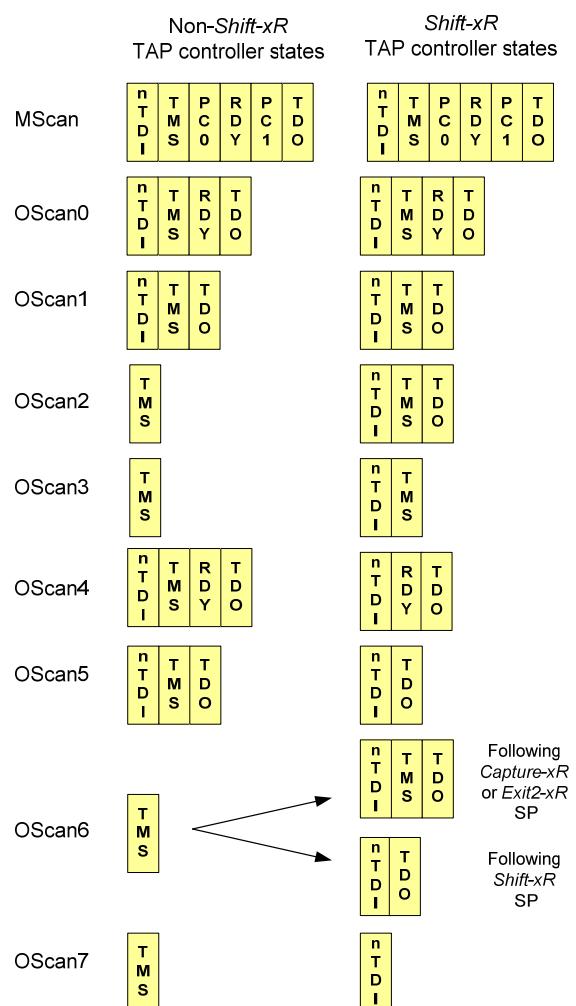
## Annex B

(informative)

### Scan examples in timing diagram form

#### B.1 MScan and OScan SP types

The SP Packet content used with MScan and OScan Scan Formats is shown in Figure B-1. With some scan formats, the packet content varies between the non-*Shift-xR* and *Shift-xR* TAP controller states. In the case of the OScan6 Scan Format, there is also a variation between the SP associated with the *Shift-xR* state following a *Capture-xR* or *Exit2-xR* TAP controller state and the SP associated with the *Shift-xR* state following a *Shift-xR* state.



**Figure B-1 — MScan and OScan SP content**

The following information is provided to aid in understanding the MScan, OScan, and SScan examples that follow:

- The area labeled “Virtual TAP Controller State” references the TAP Controller State associated with the serialized TDI, TMS, and TDO information within a Scan Packet.
- The area labeled “Contents” shows the SP payload content associated with a specific Virtual TAP Controller State.
- The area labeled “Value” shows the value required for each payload bit in the serialization. Note that the value shown for each TMS bit period is the value presented at the pin and that this value is clocked in at the rising edge of the next TCKC clock period.
- The area labeled “Real TAP Controller State” and the lines connecting the beginning of each Real TAP Controller State with each Virtual TAP Controller State depict the correlation between Scan Packets and TAPC state changes.

## B.2 MScan and OScan transactions

Transactions using MScan and OScan payloads are shown in the following figures:

- Figure B-2 MScan transaction
- Figure B-3 OScan0 transaction
- Figure B-4 OScan1 transaction
- Figure B-5 OScan2 transaction
- Figure B-6 OScan3 transaction
- Figure B-7 OScan4 transaction
- Figure B-8 OScan5 transaction
- Figure B-9 OScan6 transaction
- Figure B-10 OScan7 transaction

The contents of any TMSC Value entry shown as “A” may be either a logic 1 or a logic 0. Alternating Scan Packets are shaded. These Scan Packets denote the virtual states. The real states are also shown along with their correlation to virtual and real TAPC states. These conventions are followed in the following figures depicting scan format transactions.

### B.2.1 MScan transaction

The MScan example shown in Figure B-2 illustrates:

- The same Scan Packet content for shift and non-*Shift-xR* TAP controller states
- Transition between non-*Shift-xR* and shift TAP controller states

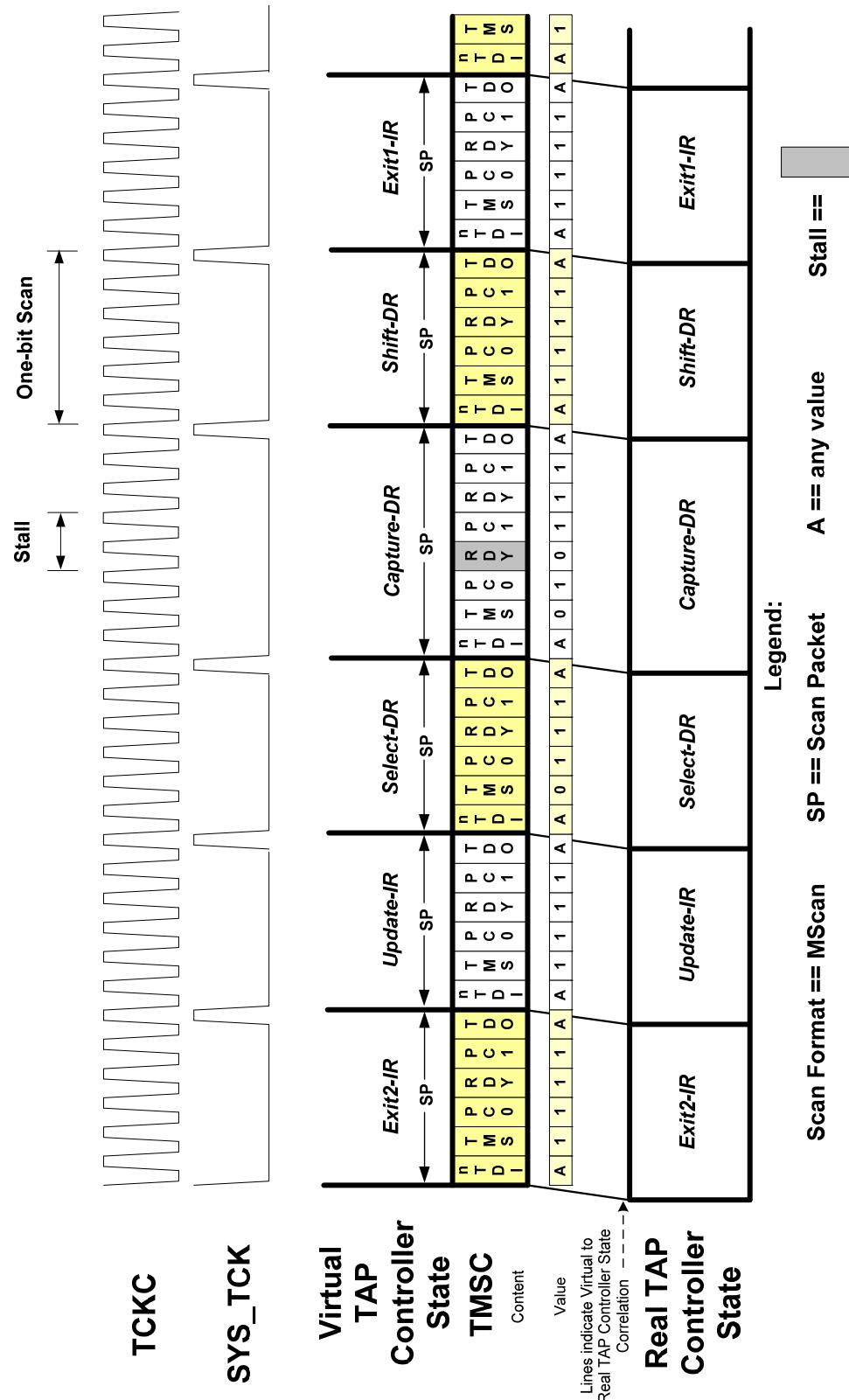
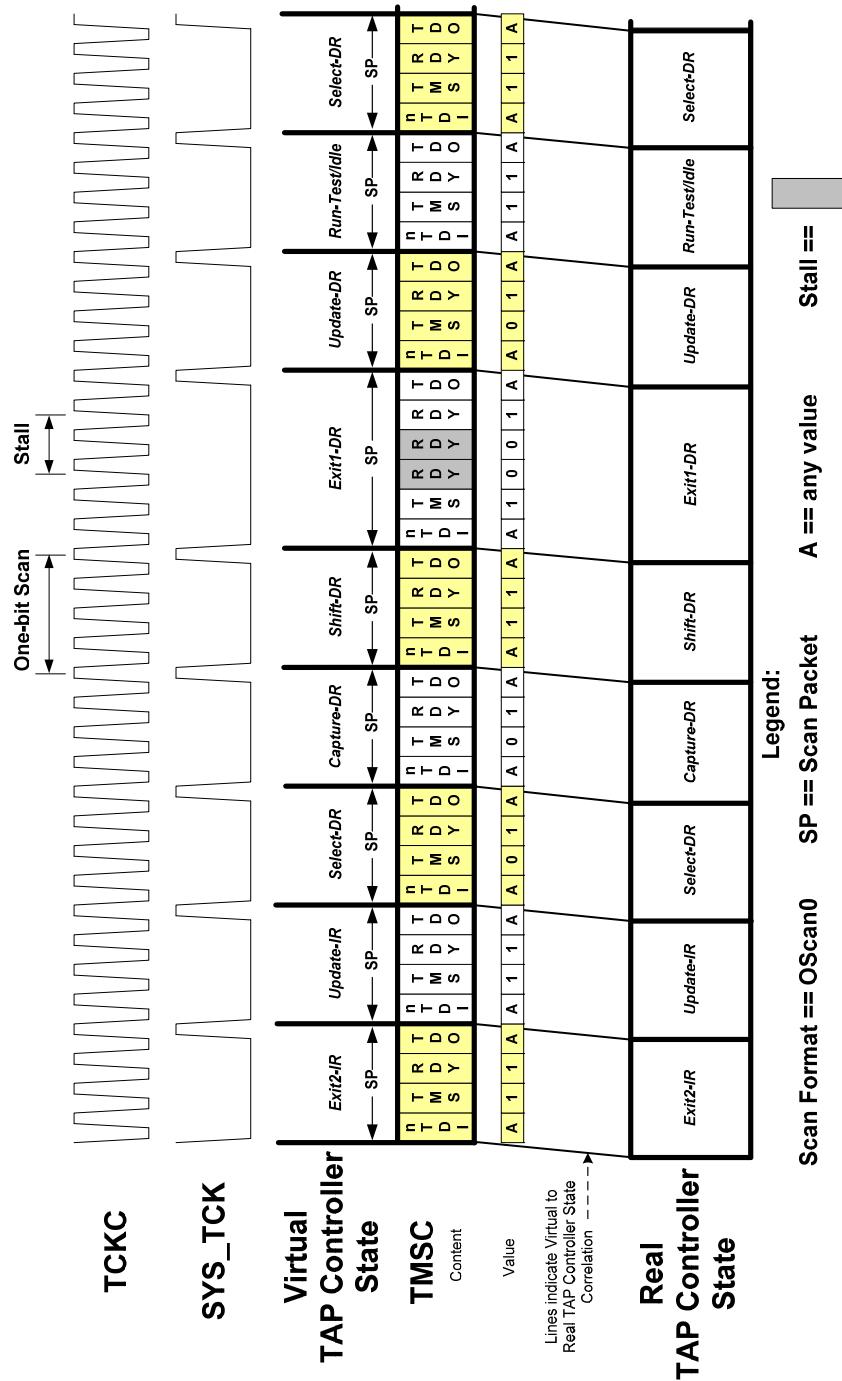


Figure B-2 — MScan transaction

### B.2.2 OScan0 transaction

The OScan0 example shown in B-3 illustrates:

- The same Scan Packet content for shift and non-*Shift-xR* TAP controller states
  - A transition between non-*Shift-xR* and *Shift-xR* TAPC states

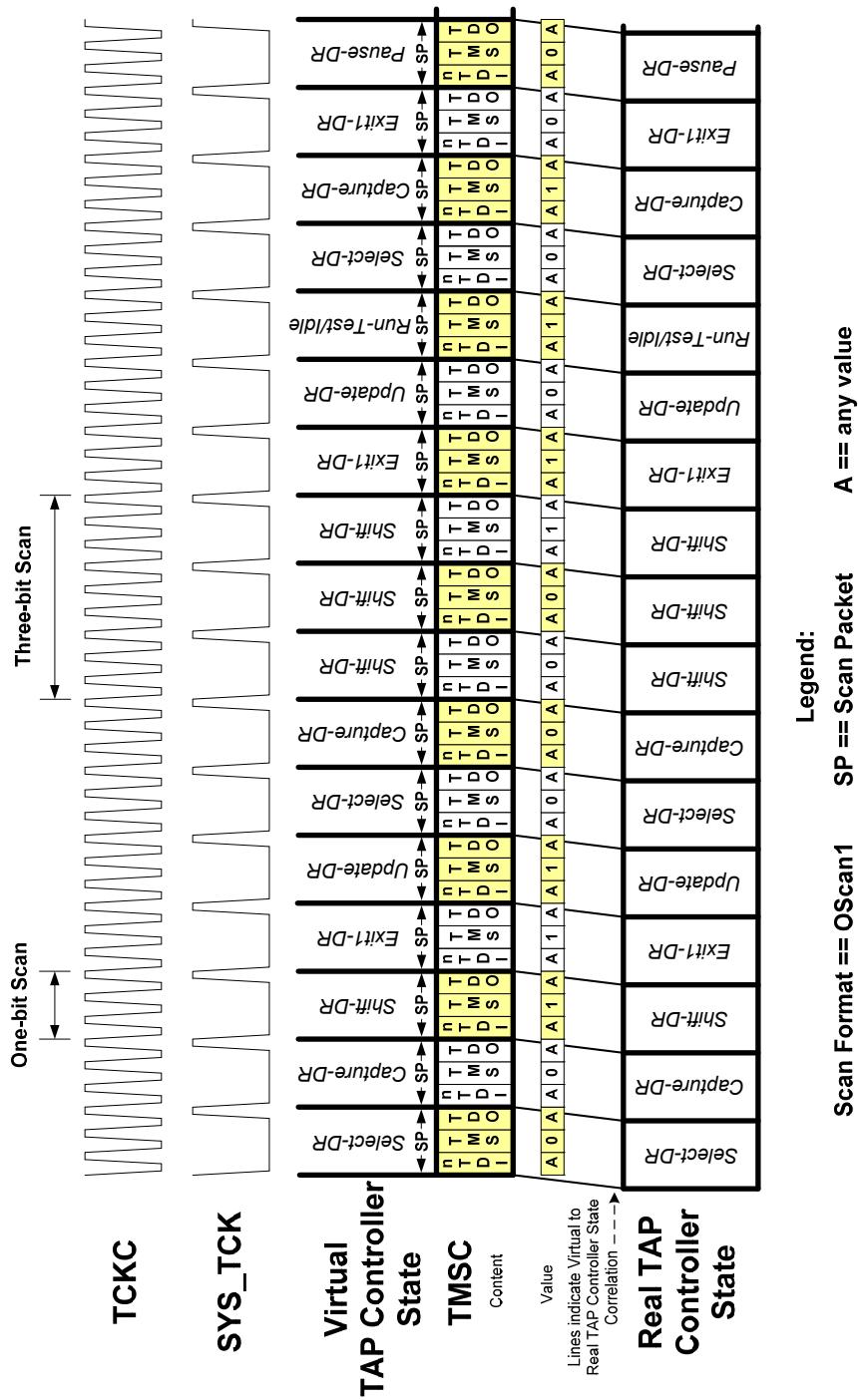


**Figure B-3 — OScan0 transaction**

### B.2.3 OScan1 transaction

The OScan1 example shown in B-4 illustrates:

- The same Scan Packet content for all TAPC states
  - A transition into and out of the TAP controller state

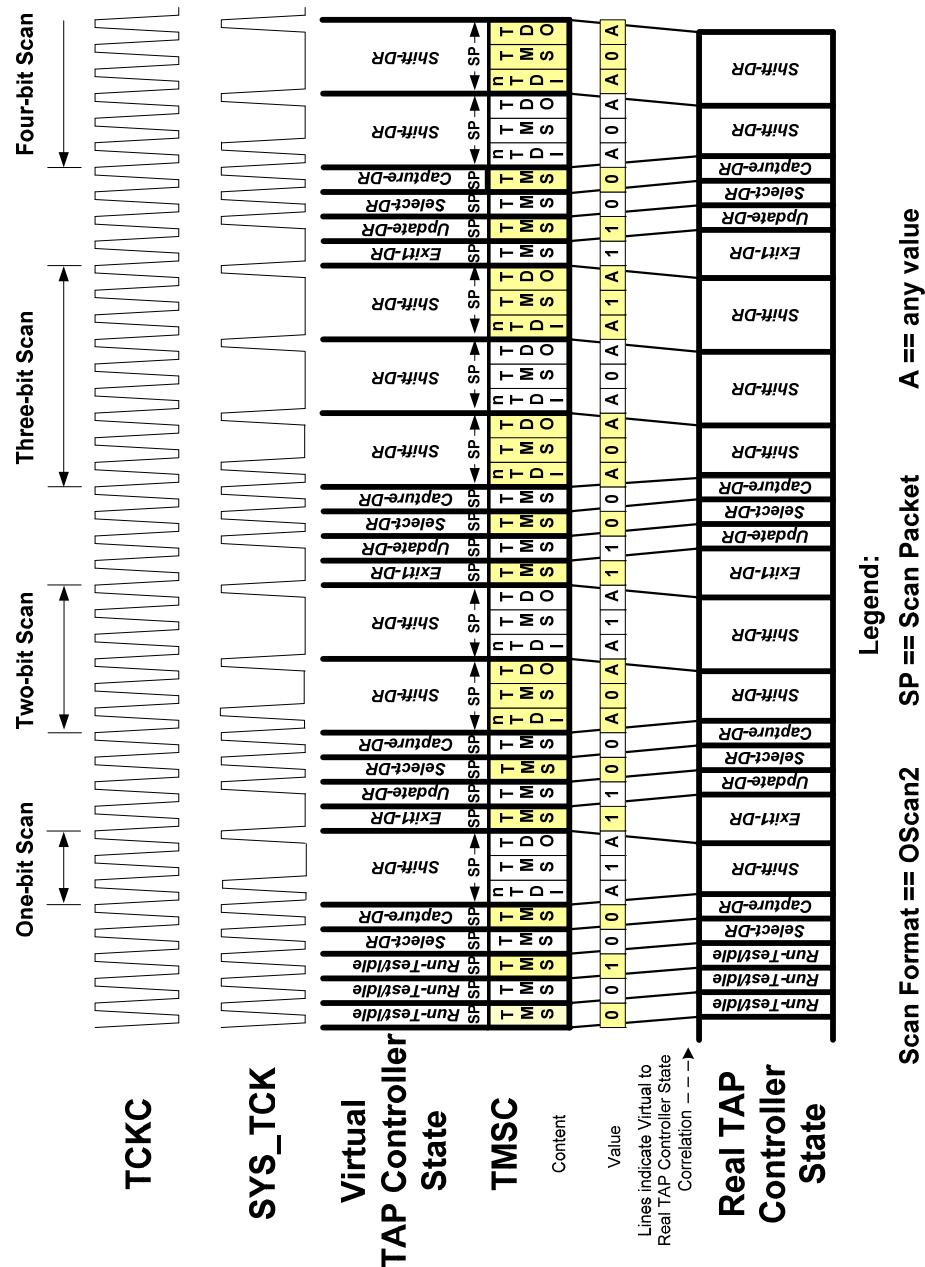


**Figure B-4 — OScan1 transaction**

#### B.2.4 OScan2 transaction

The OScan2 example shown in B-5 illustrates:

- Pipelined operation
  - One-bit Packets used for non-*Shift-xR* TAP controller states
  - Three-bit Packets used for *Shift-xR* states
  - A transition into and out of the *Shift-xR* state

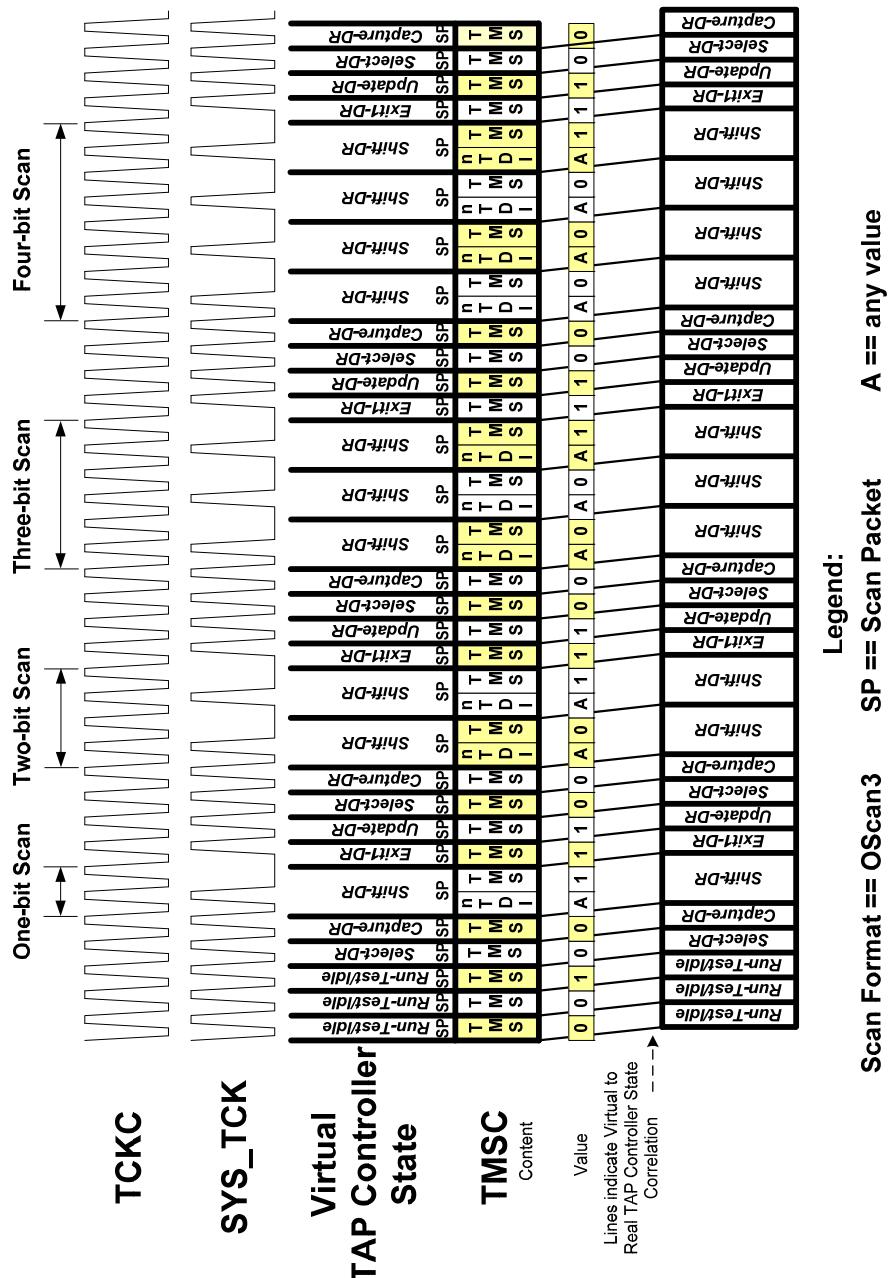


**Figure B-5 — OScan2 transaction**

## B.2.5 OScan3 transaction

The OScan3 example shown in B-6 illustrates:

- Pipelined TAP.7 Controller operation
  - One-bit Packets used for TAPC states other than *Shift-xR*
  - Two-bit Packets used for *Shift-xR* states
  - Transitions into and out of a *Shift-xR* TAP controller state



**Figure B-6 — OScan3 transaction**

## B.2.6 OScan4 transaction

The OScan4 example shown in B-7 example illustrates:

- Four-bit Packets used for TAPC states other than *Shift-xR*
- Three-bit Packets used for *Shift-xR* states
- Transitions into and out of the *Shift-xR* state

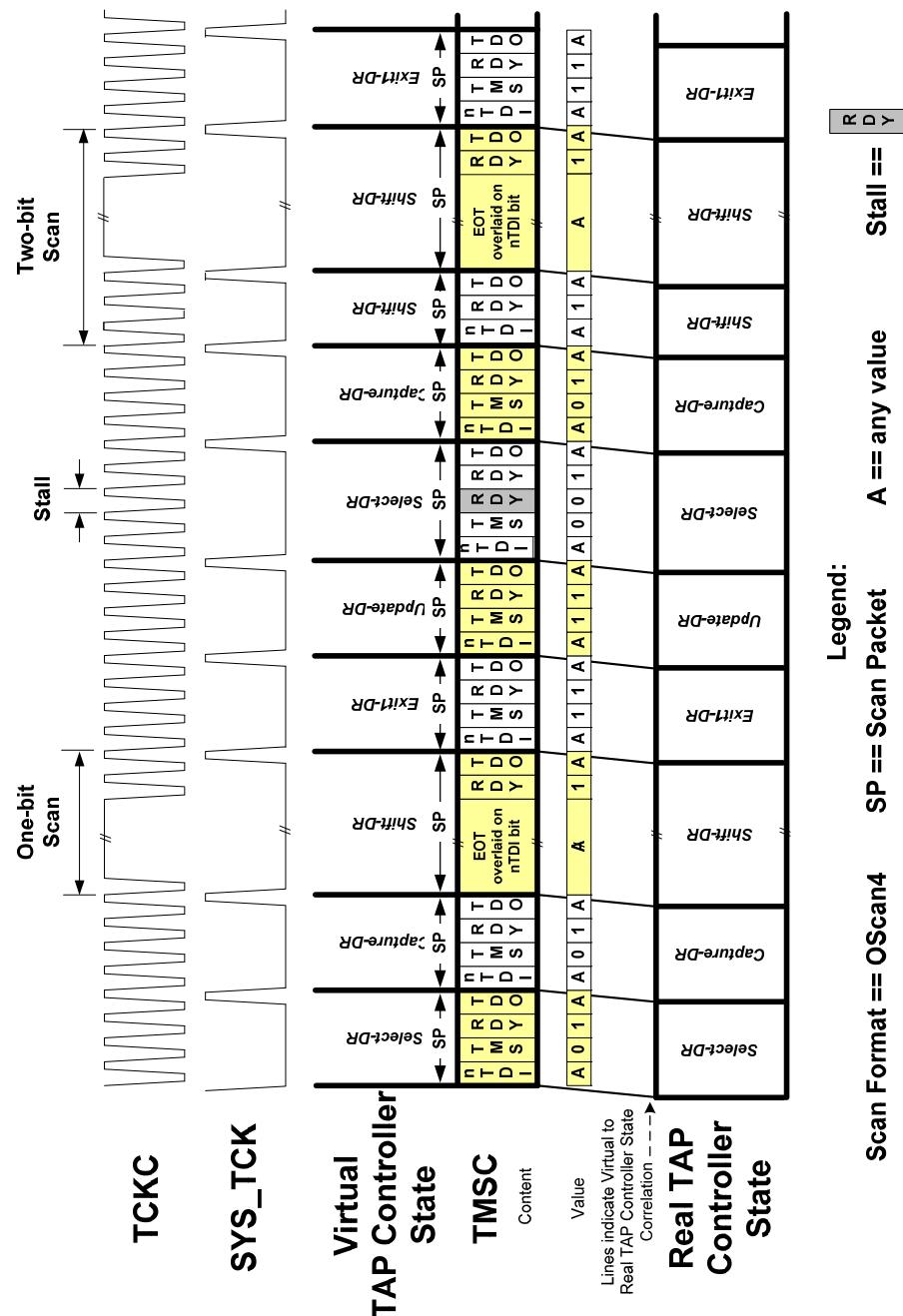


Figure B-7 — Oscan4 transaction

### B.2.7 OScan5 transaction

The OScan5 example shown in B-8 illustrates:

- Three-bit Packets used for TAPC states other than *Shift-xR*
- Two-bit Packets used for *Shift-xR* TAP controller states
- The use of an EOT Escape to cause an exit from the *Shift-xR* state
- Transitions into and out of a *Shift-xR* TAP controller state

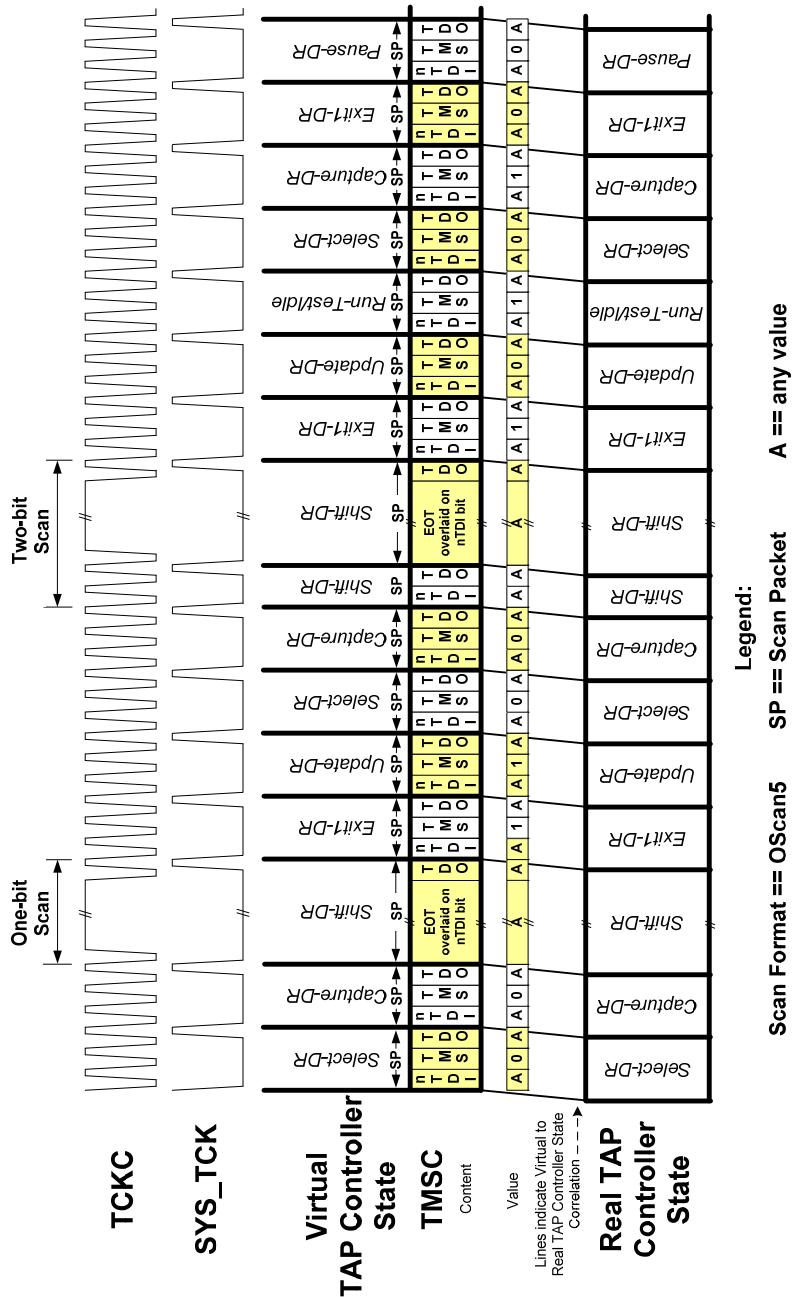
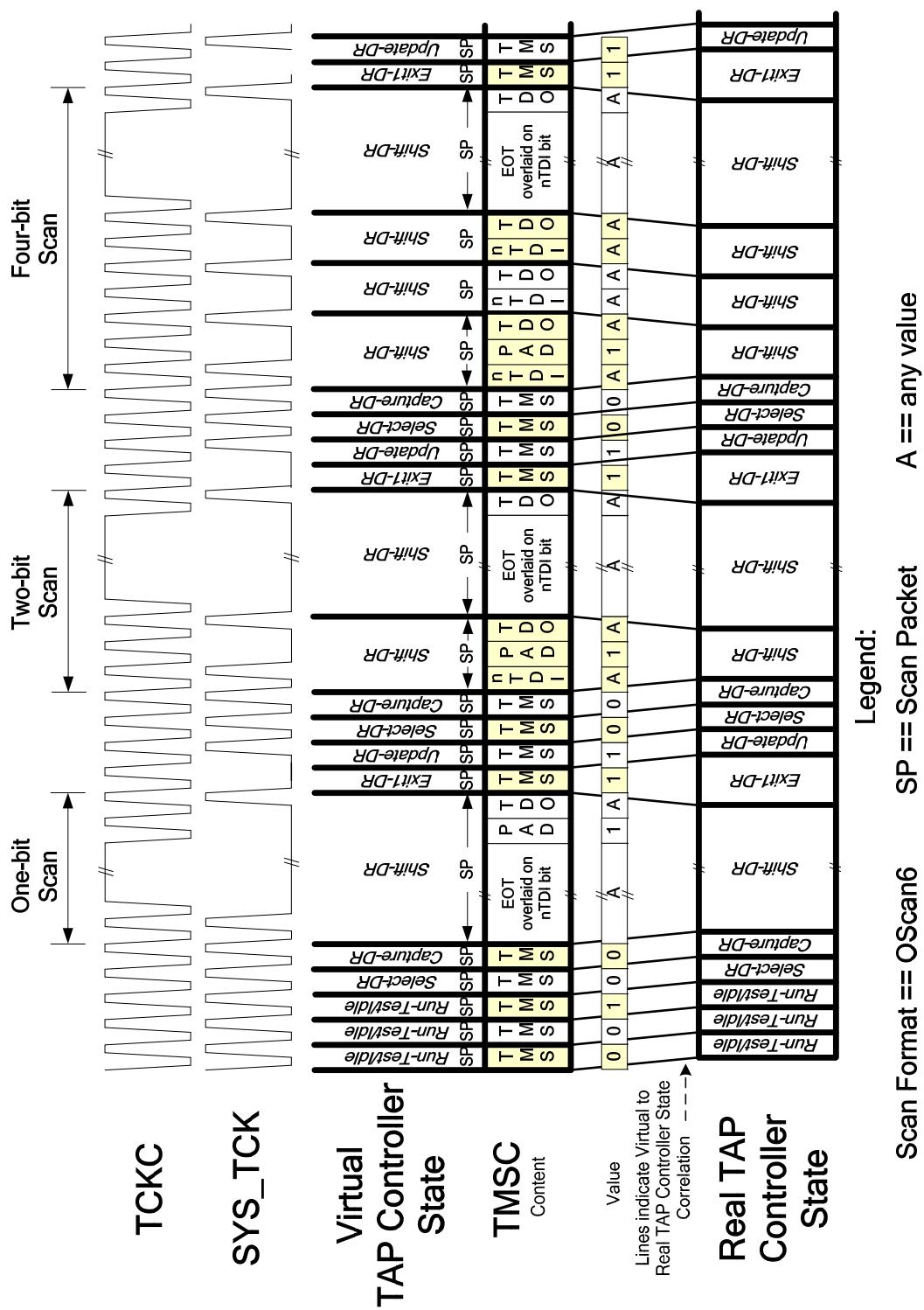


Figure B-8 — OScan5 transaction

### B.2.8 OScan6 transaction

The OScan6 example shown in B-9 illustrates:

- One-bit Packets used for TAPC states other than *Shift-xR*
- Three-bit Packets used for a *Shift-xR* state when the *Shift-xR* state follows a TAPC state other than *Shift-xR*
- Two-bit Packets used for a *Shift-xR* state when the *Shift-xR* state follows a *Shift-xR* state
- An EOT Escape causing an exit from the *Shift-xR* state
- Transitions into and out of the *Shift-xR* state

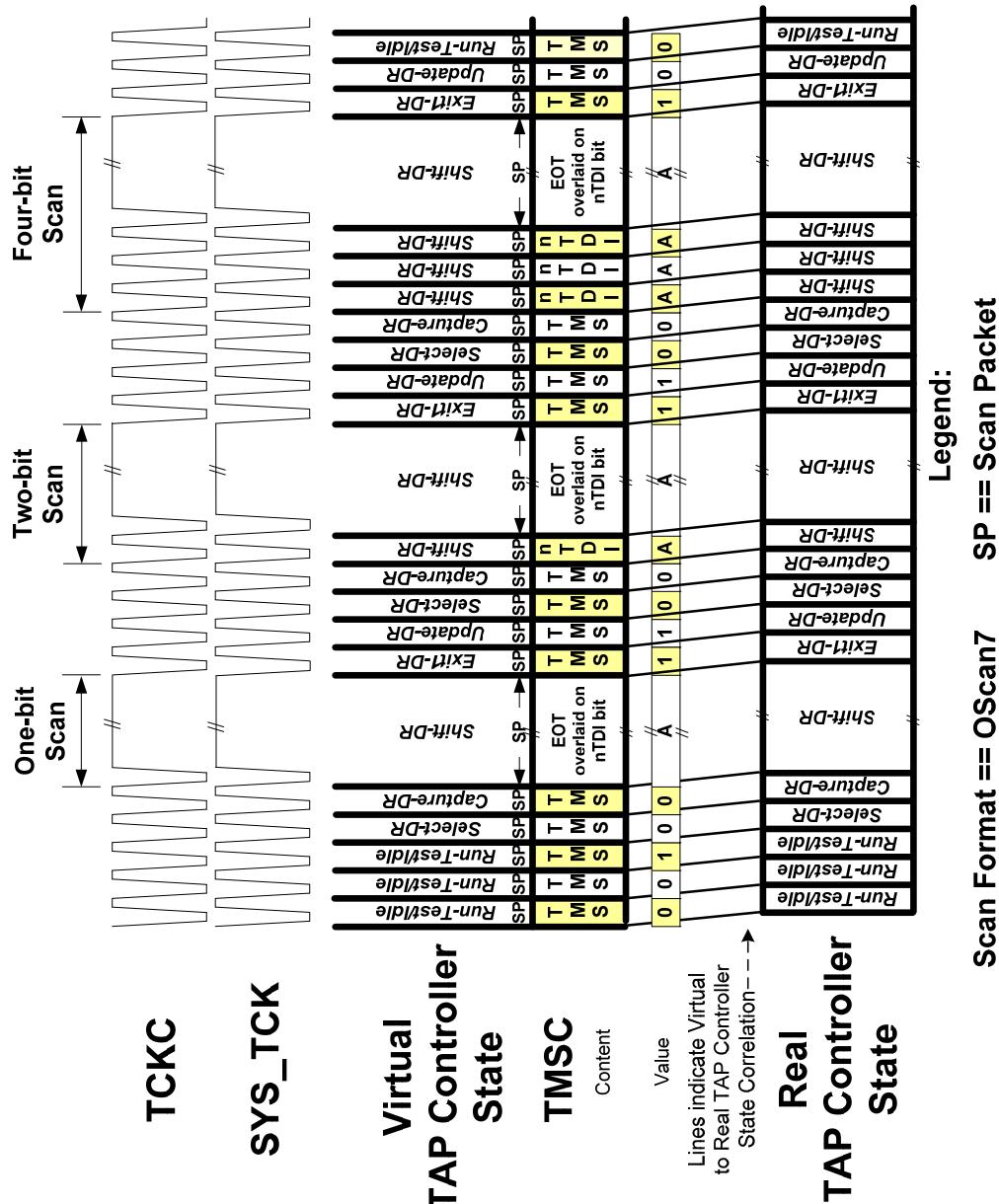


**Figure B-9 — OScan6 transaction**

### B.2.9 OScan7 transaction

The OScan7 example shown in B-10 illustrates:

- One-bit Packets used for all TAPC states
  - A EOT Escape causing an exit from the *Shift-xR* state
  - Transitions into and out of the *Shift-xR* state



**Figure B-10 — OScan7 transaction**

### B.3 SScan transactions

Transactions using SScan payloads are shown in the following figures:

- Figure B-11 SScan0 transaction/first SP stall profile
- Figure B-12 SScan0 transaction/all stall profile
- Figure B-13 SScan1 transaction/first SP stall profile
- Figure B-14 SScan1 transaction/no stall profile
- Figure B-15 SScan2 transaction/first SP stall profile
- Figure B-16 SScan2 transaction/all stall profile
- Figure B-17 SScan3 transaction/first stall profile
- Figure B-18 SScan3 transaction/no stall profile

The following conditions describe the operation of SScan Scan Formats:

- An nTDI, TMS, or END bit ends an SP, if not immediately followed by an RDY bit, a TDO bit, or a Delay Element bit.
- A TDO bit ends an SP, if not immediately followed by a Delay Element bit.
- The last bit of a Delay Element ends an SP.
- If an nTDI, TMS, or END bit is the last bit in an SP payload, the TAPC state is advanced at the rising edge of the next TCKC clock cycle (i.e., during the next bit).
- If a TDO bit is the last bit in an SP payload, the TAP controller state is advanced at the rising edge of TCKC within the TDO bit.
- The completion of an SP within a Control Segment causes a TAPC state advance.
- An SP that is followed by a CP is not part of a Control Segment and does not cause a TAPC advance.
- The completion of an SP with an input bit-frame within a Data Segment causes a TAP advance if the SP is not the last SP of the Data Segment.
- When the scan format is SScan2 or SScan3, the Data Segment is output only, and there is at least one SP without an input bit-frame, the last two SPs do not cause a TAP advance.
- A TAP advance occurs at the rising edge of the TCKC clock cycle during the HDR2 bit of the Header Element of an SP that follows a Data Segment (is the first SP of a new segment).
- A Control Segment ends in one of three ways:
  - Automatically as the *Shift-xR* state is entered.
  - As a result of a controller command.
  - As a result of an EOT Escape or a logic 1 end bit.
- The first two conditions listed immediately above are mutually exclusive. They may collide with the third condition with either of the first two conditions having precedence.
  - Automatically when the SP is associated with an *Exit2-DR* or *Capture-DR* state and the value of the TMS bit within the SP is a logic 0. The next SP is the first SP of a Data Segment.
  - A command generates an SP/CP combination. This SP is associated with either the *Run-Test/Idle* or *Select-DR-Scan* state and does not advance the TAPC state. This SP has the same content as the first SP of the preceding Control Segment. When the scan format remains an SScan Scan Format, a new Control Segment begins after the CP.

- When neither of the two above conditions is satisfied, the next packet is an SP of a new Control Segment. Otherwise, the Escape or END bit is ignored.
- A Data Segment ends in only one way, an Escape or an END bit indicates the end of a Data Segment.
- If the scan format is SScan0 or SScan1, the END bit occurs in the last SP of the Data Segment and the TAP is not advanced for this SP. Instead, it is advanced at the last header bit of the following SP. In this case, the number of SPs in the Data Segment is equal to the number *Shift-xR* states in the Data Segment.
- If the scan format is SScan2 or SScan3 and the last SP of the Data Segment has an nTDI bit, an EOT Escape occurring coincidentally with the nTDI bit of the last SP of the Data Segment and the TAP is not advanced for this SP. In this case, the number of SPs in the Data Segment is equal to the number *Shift-xR* states in the Data Segment. When there is no TDI bit in the SP, the Escape occurs in the RDY or TDO bit of the SP and is followed by two additional SPs. In this case, the TAP is advanced for the SP containing the Escape but not for the two additional SPs. In this case, the number of SPs in the Data Segment is equal to the number *Shift-xR* states in the Data Segment plus one.
- The header in the first SP of the new segment following a Data Segment determines whether the *Shift-xR* TAPC state is exited. HDR[1:0] in the header in the first SP of the new segment indicates whether the new segment is a Control Segment or Data Segment (HDR[1:0] == 11b means that a Control Segment follows). This value creates a logic 1 TMS value, and once the TMS value is generated, the TMS value is a logic 0 for TAP advances generated by SPs within a Data Segment.

Unique characteristics of Control Segments are as follows:

- HDR[1:0] are a don't care in the header of a Control Segment following a Control Segment
- SPs have no TDI bits and are constructed with TMS/END/RDY/TDO bit combinations
- Each SP in a Control Segment not preceding a CP advances the TAP controller state
- They are terminated by register-writes or *Shift-xR* state entry in addition to Escape and END bits

Unique characteristics of Data Segments are as follows:

- HDR[1:0] define subsequent SP payload and exit from the *Shift-xR* state
- SPs have no TMS bits and are constructed with nTDI/END/RDY/TDO bit combinations
- The last SP in a Data Segment and sometimes the last two SPs of a Data Segment do not advance the TAP controller state (this TAP advance occurs within the HDR[2] bit within the first SP of the next segment)

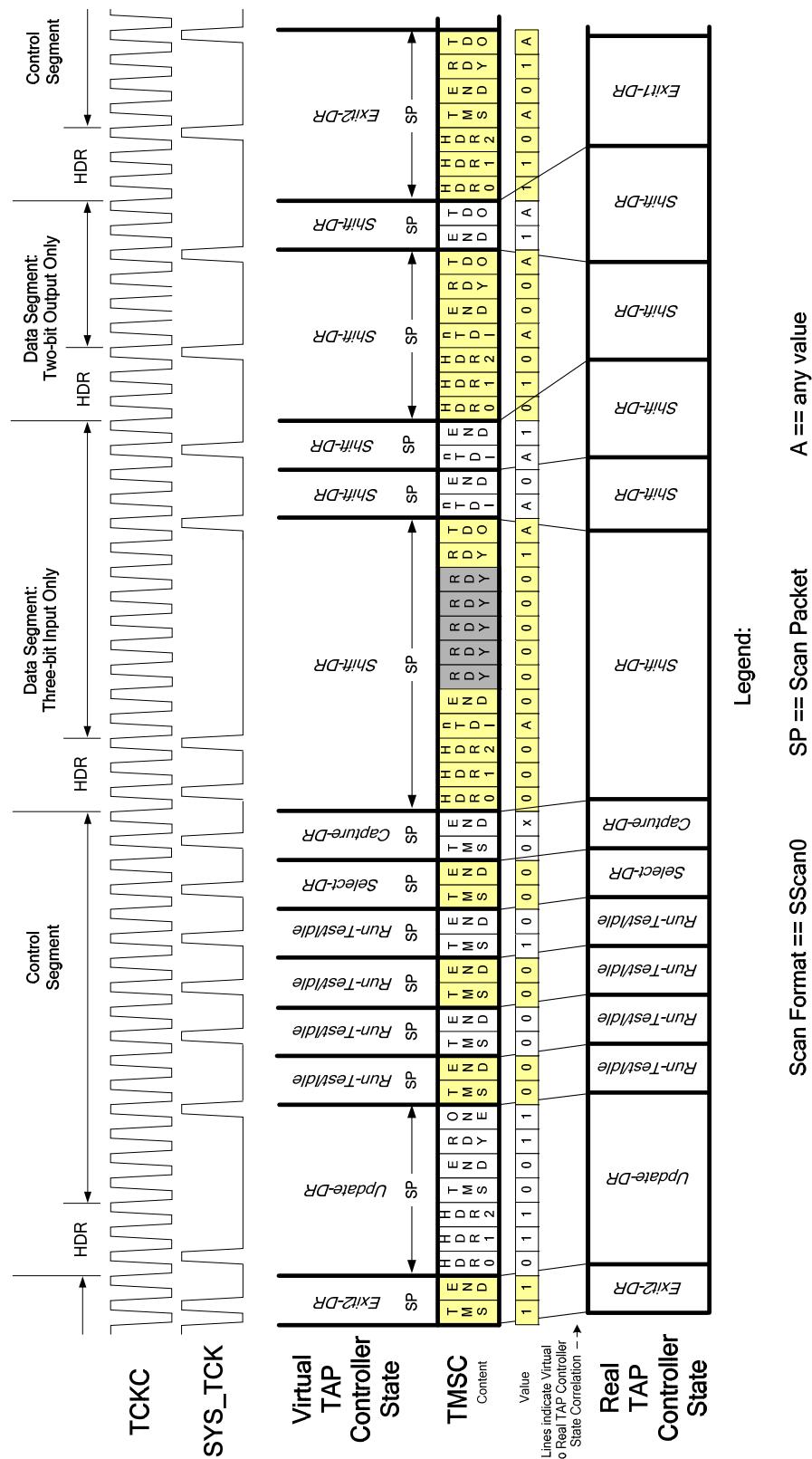
### B.3.1 SScan0 transaction

Figure B-11 shows an SScan0 transaction with a first stall profile. This transaction illustrates:

- Two Control Segments preceding *Shift-xR* state entry
- An RDY bit in the first SP in control and Data Segments
- Input-only and output-only Data Segments

Figure B-12 shows an SScan0 transaction with the all-stall profile. This transaction illustrates:

- A single Control Segment between Data Segments
- RDY bits in all SPs within control and Data Segments
- Input-only and output-only Data Segments



**Figure B-11 — SScan0 transaction/first SP stall profile**

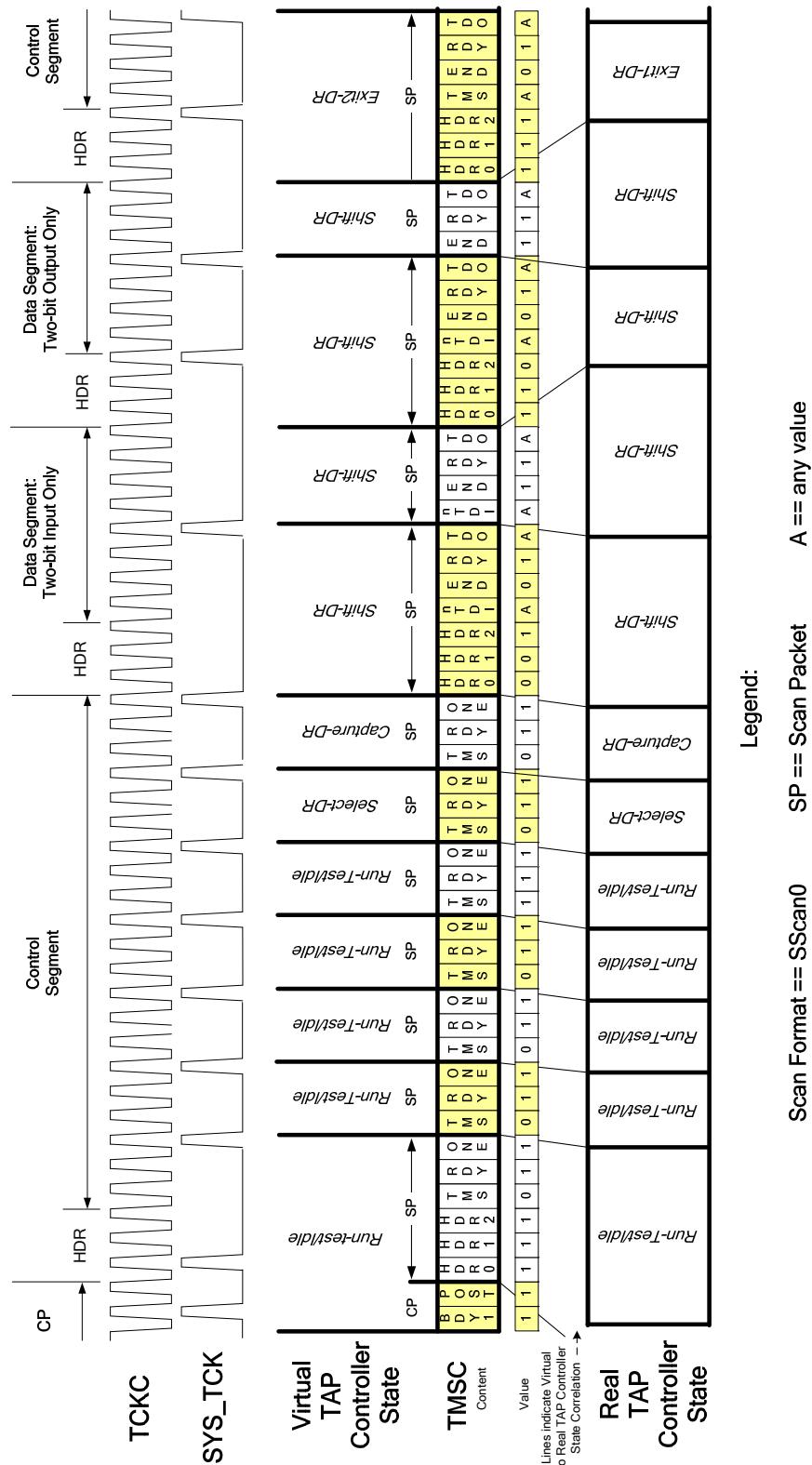


Figure B-12 — SScan0 transaction/all stall profile

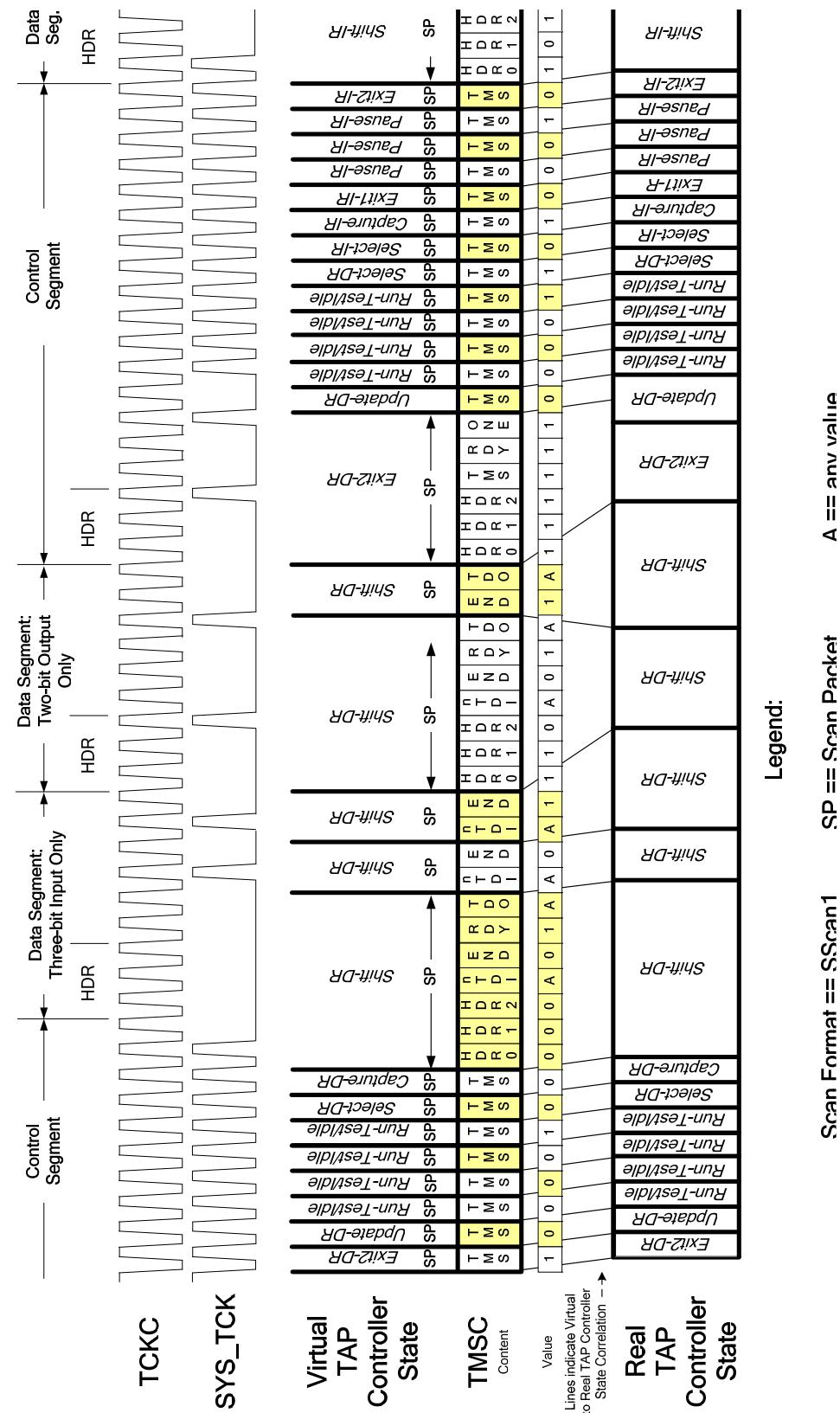
### B.3.2 SScan1 transaction

Figure B-13 shows an SScan1 transaction with a first stall profile. This transaction illustrates:

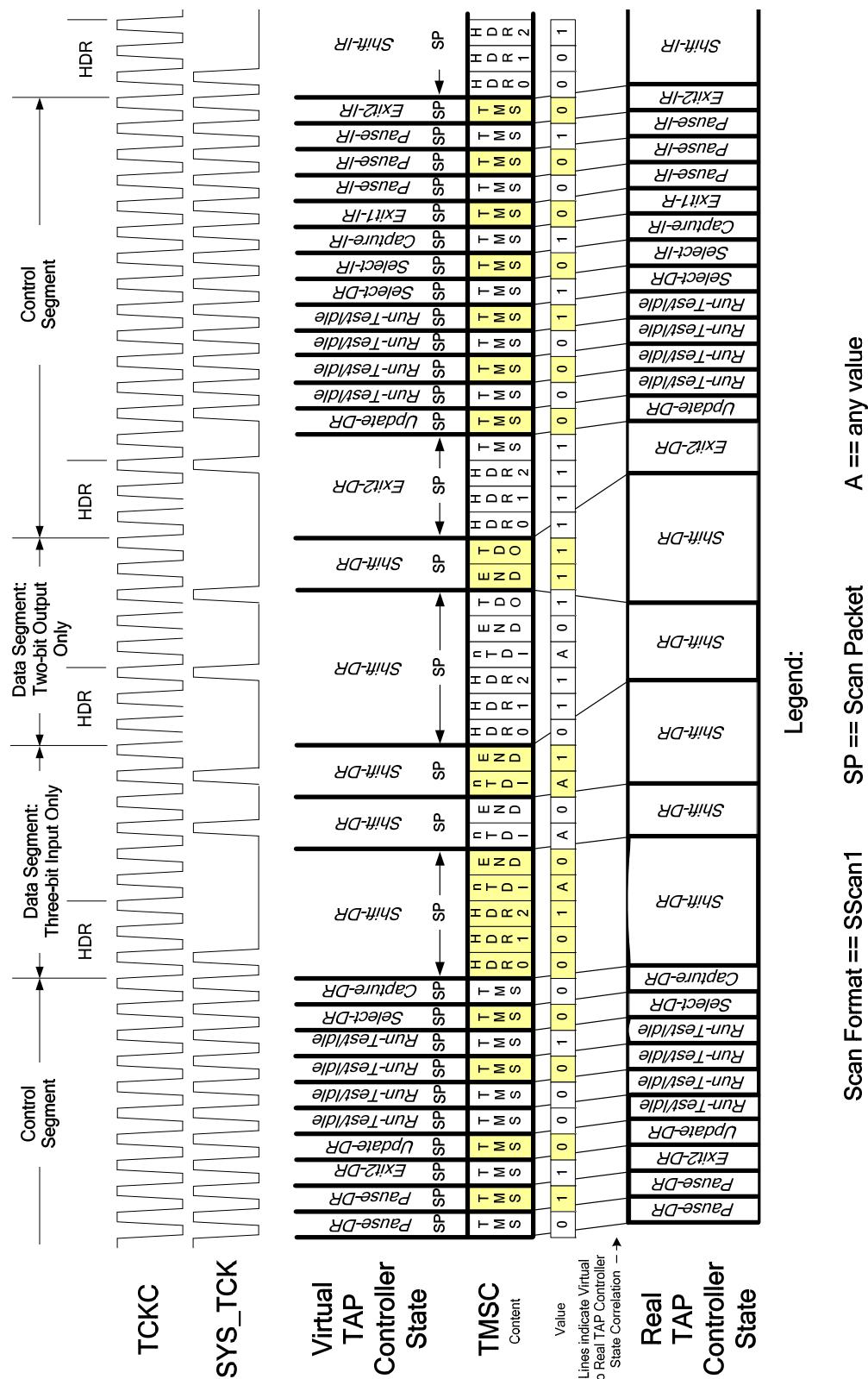
- A single Control Segment between Data Segments
- An RDY bit in the first SP in control and Data Segments (per SScan1/first stall profile)
- Input-only and output-only Data Segments
- Entry into the *Shift-xR* state from both a *Capture-xR* and *Exit2-xR* states

Figure B-14 shows an SScan1 transaction with a no stall profile. This transaction illustrates:

- A single Control Segment between Data Segments
- No RDY bits in control and Data Segments (per SScan1/no stall profile)
- Input-only and output-only Data Segments
- Entry into the *Shift-xR* state from both a *Capture-xR* and *Exit2-xR* states



**Figure B-13 — SScan1 transaction/first SP stall profile**



**Figure B-14 — SScan1 transaction/no stall profile**

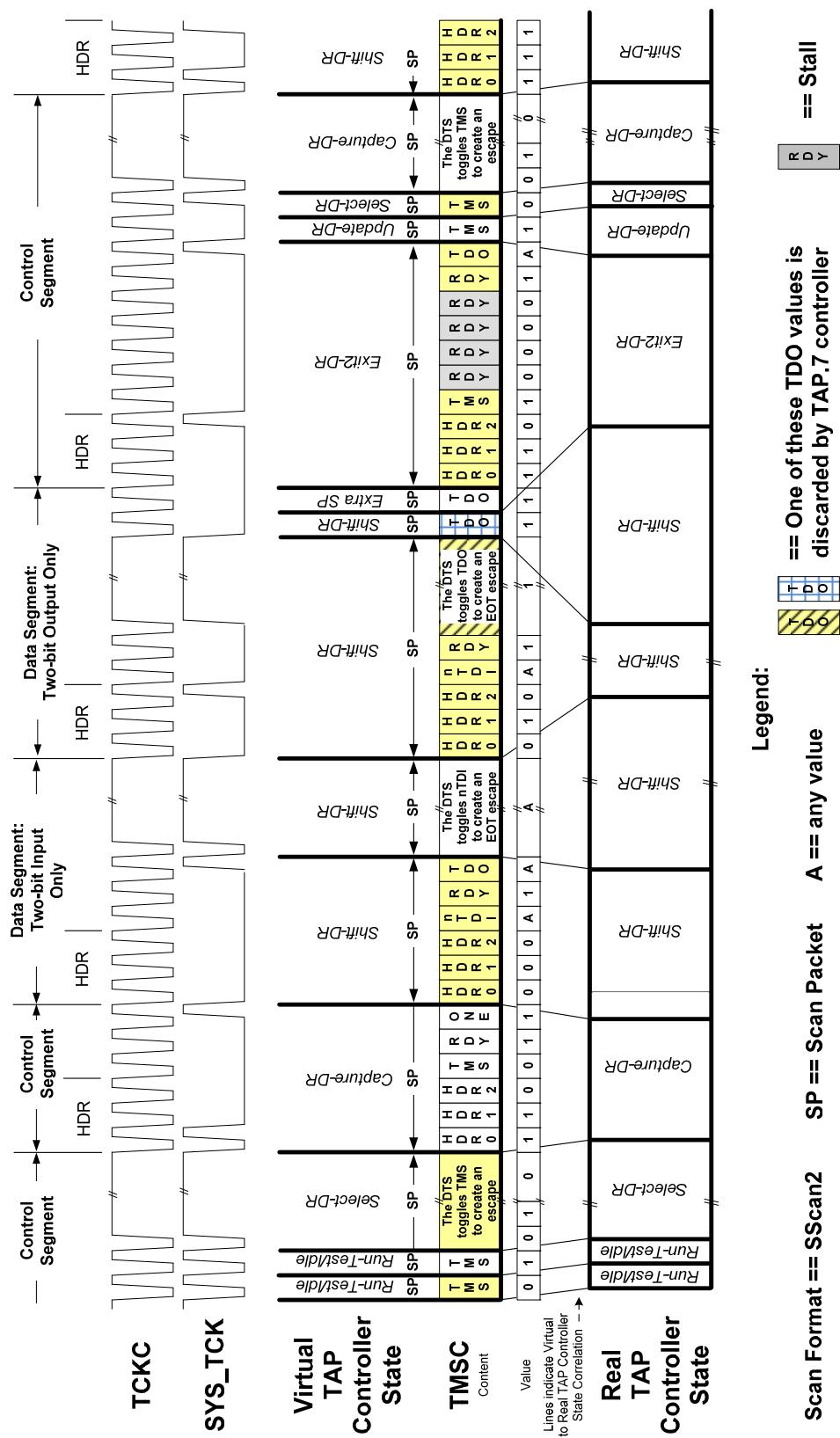
### B.3.3 SScan2 transaction

Figure B-15 shows an SScan2 transaction with a first stall profile. This transaction illustrates:

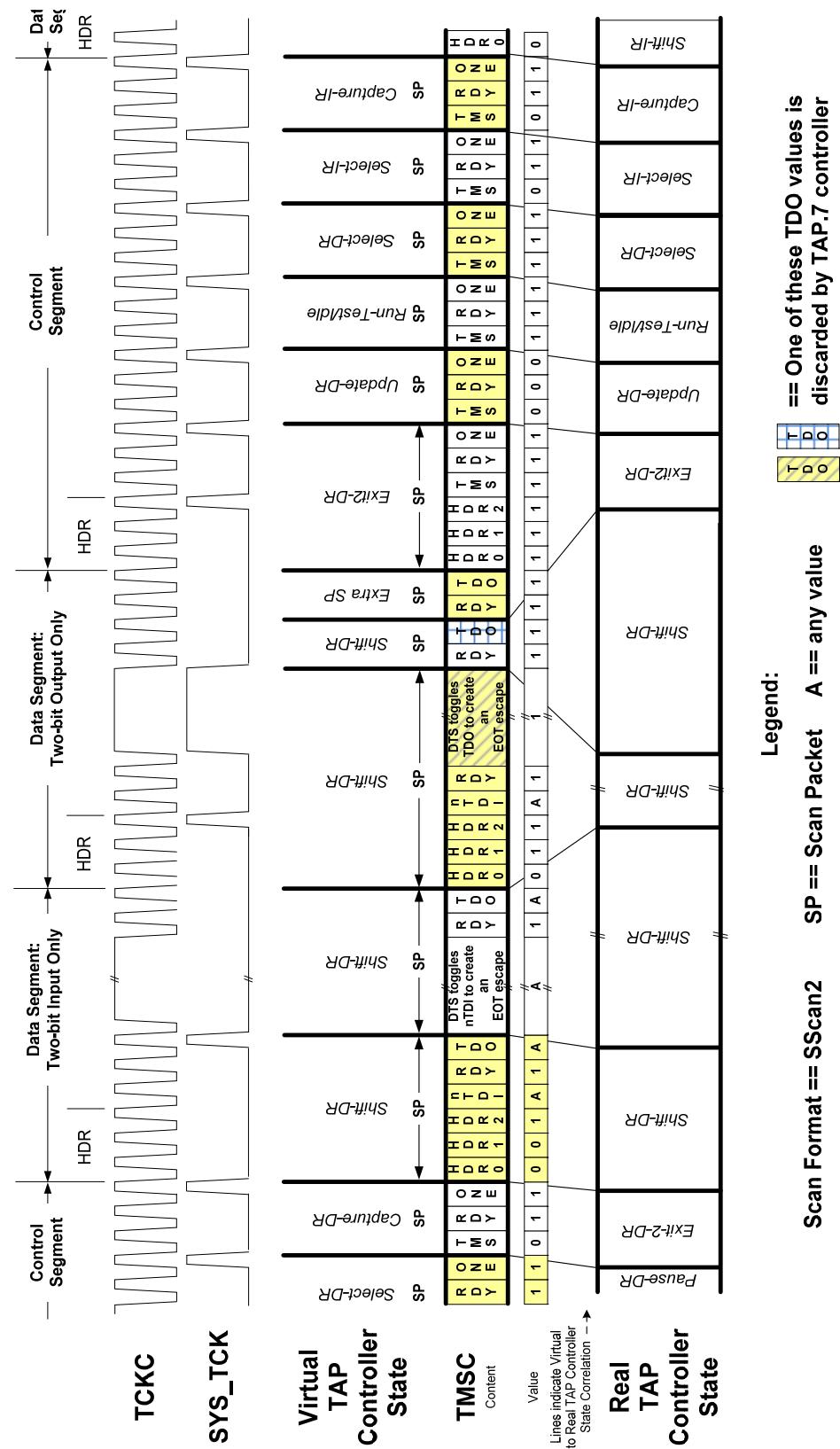
- Two Control Segments preceding *Shift-xR* state entry
- An RDY bit in the first SP in control and Data Segments
- Input-only and output-only Data Segments

Figure B-16 shows an SScan2 transaction with an all stall profile. This transaction illustrates:

- A single Control Segment between Data Segments
- RDY bits in all SPs within control and Data Segments
- Input-only and output-only Data Segments
- Entry into the *Shift-xR* state from both a *Capture-xR* and *Exit2-xR* states



**Figure B-15 — SScan2 transaction/first SP stall profile**



**Figure B-16 — SScan2 transaction/all stall profile**

### B.3.4 SScan3 transaction

Figure B-17 shows an SScan3 transaction with a first stall profile. This transaction illustrates:

- A One-Packet Control Segment preceding *Shift-xR* state entry
- An RDY bit in the first SP in control and Data Segments
- Input-only and output-only Data Segments

Figure B-18 shows an SScan3 transaction with a no stall profile. This transaction illustrates:

- A single Control Segment between Data Segments
- No RDY bits in control and Data Segments
- Input-only and output-only Data Segments
- Entry into the *Shift-xR* state from both *Capture-xR* and *Exit2-xR* states

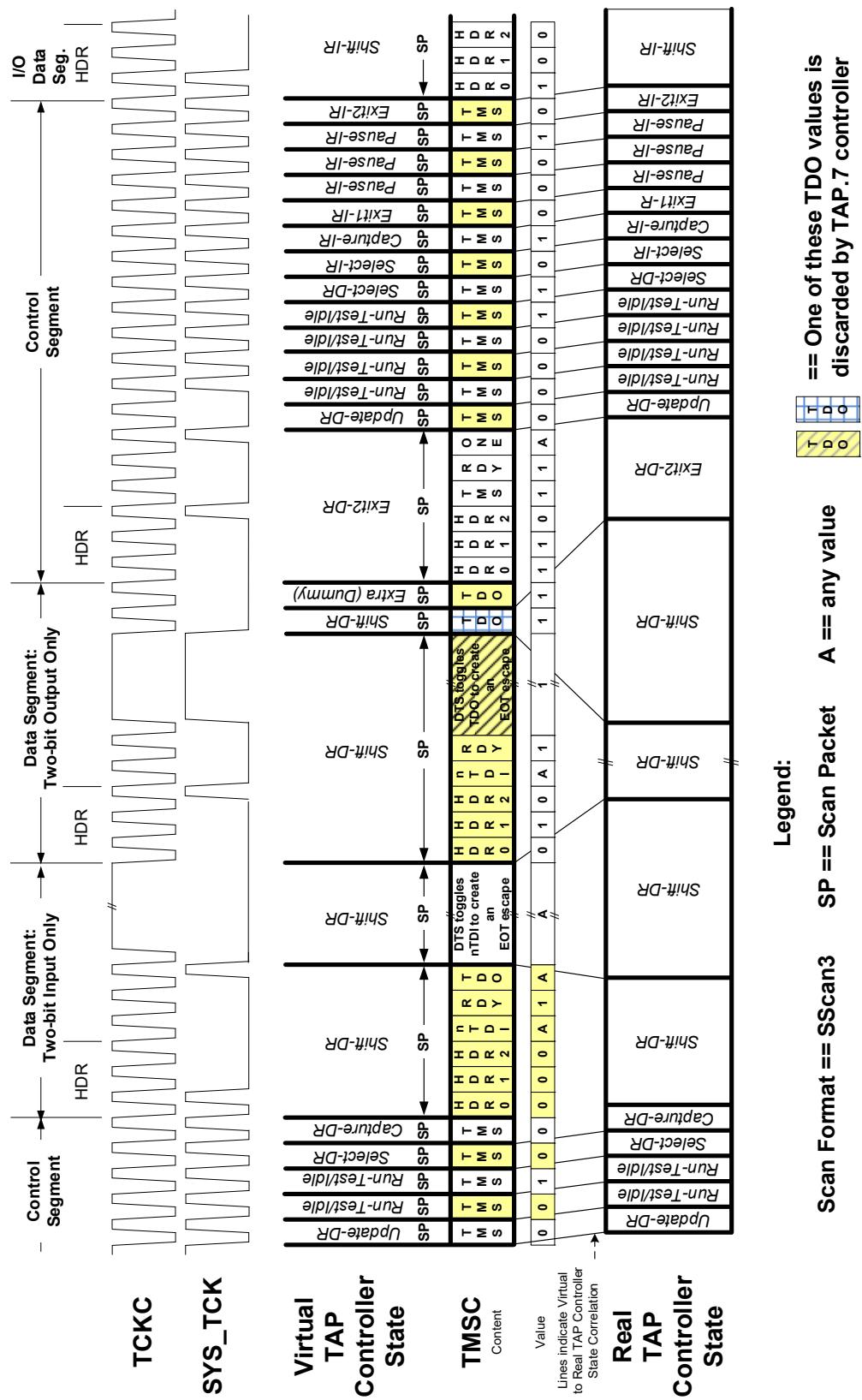
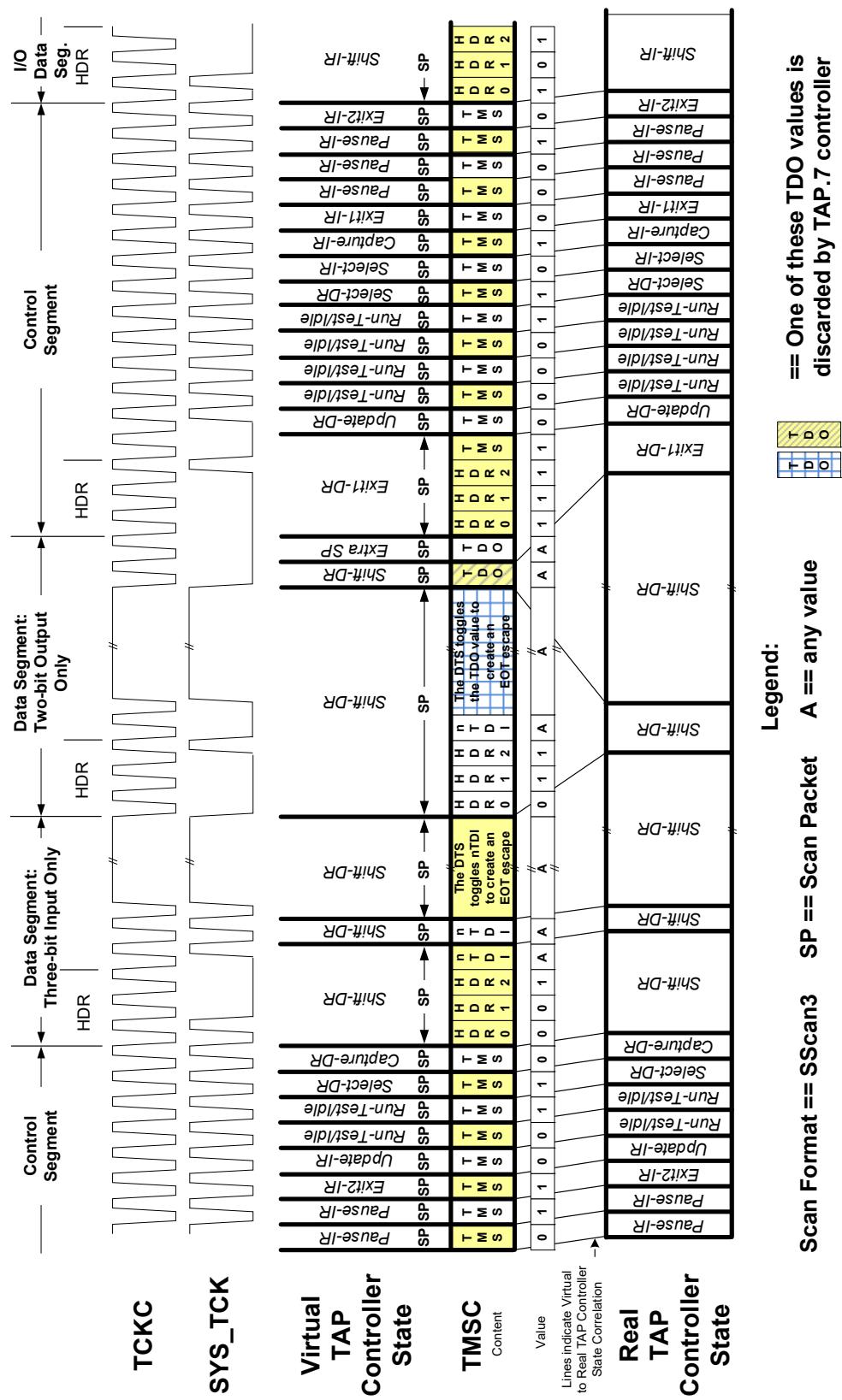


Figure B-17 — SScan3 transaction/first stall profile

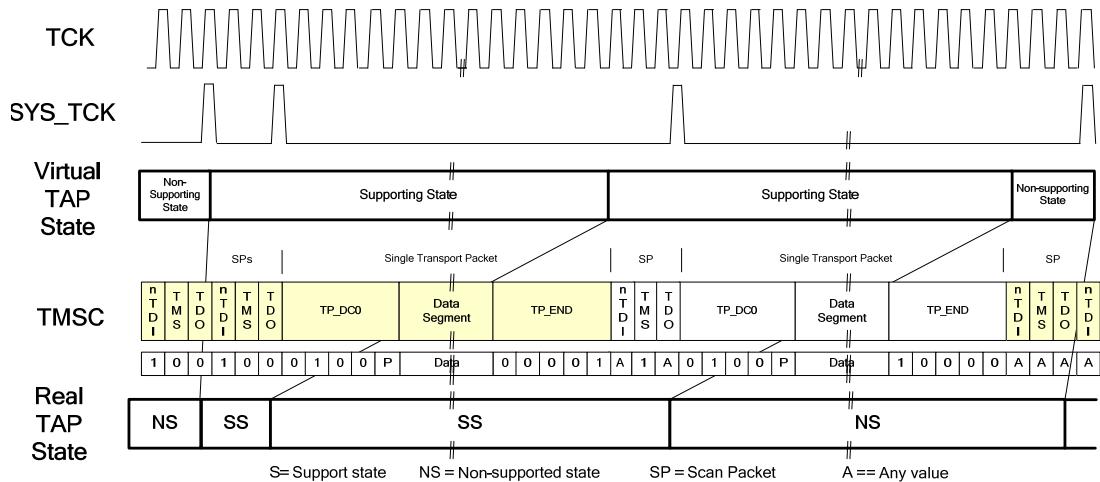


**Figure B-18 — SScan3 transaction/no stall profile**

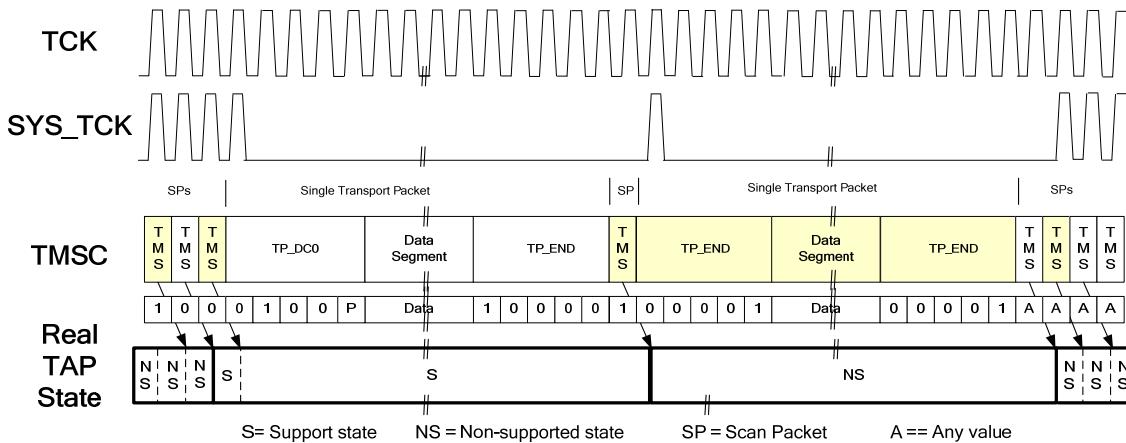
## B.4 BDX and CDX Transport Packet examples

Examples of BDX and CDX Packets used are shown for the various scan formats as listed as follows:

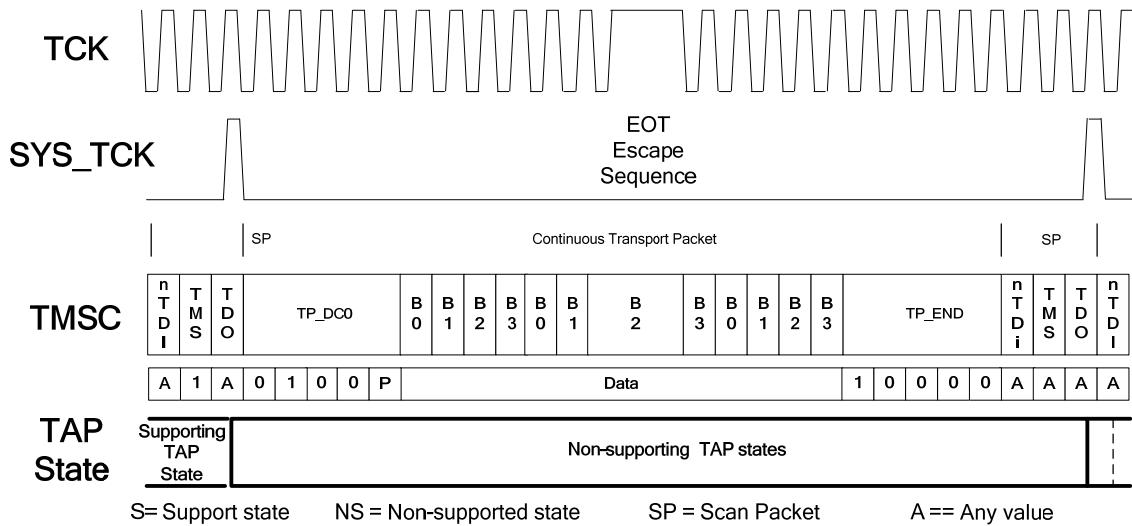
- Figure B-19 OScan1 Scan Format/TPs with fixed-length Data Element following SPs
- Figure B-20 OScan7 Scan Format/TPs with fixed-length Data Element following SPs
- Figure B-21 OScan1 Scan Format/TPs with variable-length Data Element following SPs
- Figure B-22 OScan7 Scan Format TPs with variable-length Data Element following SPs



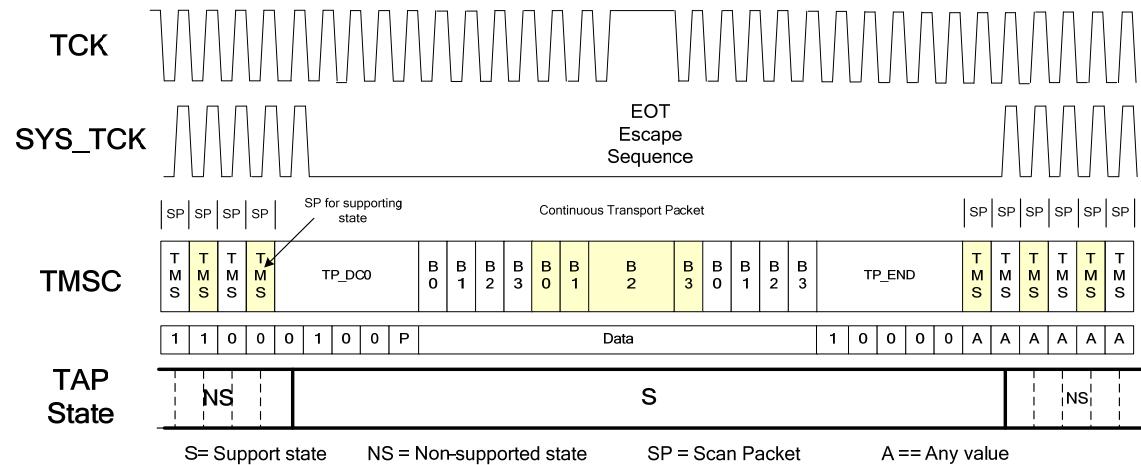
**Figure B-19 — OScan1 Scan Format/TPs with fixed-length Data Element following SPs**



**Figure B-20 — OScan7 Scan Format/TPs with fixed-length Data Element following SPs**



**Figure B-21 — OScan1 Scan Format/TPs with variable-length Data Element following SPs**



**Figure B-22 — OScan7 Scan Format/TPs with variable-length Data Element following SPs**

## Annex C

(informative)

### Scan examples in tabular form

#### C.1 Overview

This annex provides examples for MScan, OScan, and SScan Scan Formats in tabular form.

#### C.2 MScan and OScan SP types

This annex subclause provides examples for MScan and OScan Scan Formats. The RDYC Register value is a 00b for these examples. A three-bit DR Scan with 010b data is followed by a three-bit IR Scan with 111b data. In some examples, delays are added to illustrate their effect.

A second table illustrates the Check Packet content. The body of the CP is shown as two, three, and four bits in this table. Note the SP content is equivalent to that generated with a *Run-Test/Idle* or *Select-DR-Scan* TAP controller state with no Delay Elements.

The location of the TAP controller state advance caused by the SP is indicated as follows:

	None
	Advance

The tables and their relationship to specific scan formats are listed as follows:

- |          |                               |
|----------|-------------------------------|
| — MScan  | Table C-2 and Table C-3       |
| — OScan0 | Table C-4 and Table C-5       |
| — OScan1 | Table C-6 and Table C-7       |
| — OScan2 | Table C-8 through Table C-10  |
| — OScan3 | Table C-11 through Table C-13 |
| — OScan4 | Table C-14 and Table C-15     |
| — OScan5 | Table C-16 and Table C-17     |
| — OScan6 | Table C-18 through Table C-20 |
| — OScan7 | Table C-21 and Table C-23     |

The TMSC drive characteristics for delays of one through six are shown in Table C-1. In these examples, the DTS does not drive the TMSC pin during a Delay Element when the DLYC Register value is less than three. The RDYC value is zero and the DLYC Register value is zero in most cases.

The SP/CP combinations shown for each scan format follow an *Update-DR* TAPC state associated with Command Part Two

**Table C-1 — TMSC drive and delay relationships**

Delay length in TCK periods	TCKC					
Fixed delay of one	Hi-Z	—	—	—	—	—
Fixed delay of two	Hi-Z	Hi-Z	—	—	—	—
Variable delay of three	0	0	0	—	—	—
Variable delay of four	1	0	0	0	—	—
Variable delay of five	0	1	0	0	0	—
Variable delay of six	1	0	1	0	0	0

**Table C-2 — MScan scan sequence/DR Scan (010) and IR Scan (111), delay as specified**

Virtual TAPC state	nTDI	TMS	PC0	RDY	PC1	TDO	Delay
<b>TCKC →</b>	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	
<i>Run/Test/Idle</i>	X	0	1	1	1	1	
<i>Run/Test/Idle</i>	X	1	1	1	1	1	
<i>Select-DR-Scan</i>	X	0	1	1	1	1	
<i>Capture-DR</i>	X	0	1	1	1	1	
<i>Shift-DR</i>	1	0	1	1	1	DATA	
<i>Shift-DR</i>	0	0	1	1	1	DATA	
<i>Shift-DR</i>	1	1	1	1	1	DATA	
<i>Exit1-DR</i>	X	0	1	1	1	1	
<i>Pause-DR</i>	X	0	1	1	1	1	
<i>Pause-DR</i>	X	0	1	1	1	1	
<i>Pause-DR</i>	X	1	1	1	1	1	
<i>Exit2-DR</i>	X	1	1	1	1	1	
<i>Update-DR</i>	X	1	1	1	1	1	
<i>Select-DR-Scan</i>	X	1	1	1	1	1	
<i>Select-IR-Scan</i>	X	0	1	1	1	1	
<i>Capture-IR</i>	X	0	1	1	1	1	
<i>Shift-IR</i>	0	0	1	1	1	DATA	
<i>Shift-IR</i>	0	0	1	1	1	DATA	
<i>Shift-IR</i>	0	1	1	1	1	DATA	
<i>Exit1-IR</i>	X	0	1	1	1	1	
<i>Pause-IR</i>	X	0	1	1	1	1	
<i>Pause-IR</i>	X	0	1	1	1	1	
<i>Pause-IR</i>	X	1	1	1	1	1	
<i>Exit2-IR</i>	X	1	1	1	1	1	
<i>Update-IR</i>	X	0	1	1	1	1	
<i>Run/Test/Idle</i>	X	0	1	1	1	1	
<i>Run/Test/Idle</i>	X	0	1	1	1	1	

As specified

**Table C-3 — SP/CP combination following an MScan SP**

	SP					
	nTDI	TMS	PC0	RDY	PC1	TDO
<b>TCKC →</b>	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
<b>- Ready check</b>						
- Ready check	X	X	1	1	1	1
	CP					
<b>TCKC →</b>	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
- Four states	X	0	0	0	—	—
- Five states	X	1	0	0	0	—
- Six states	X	0	1	0	0	0
<b>After CP</b>						
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format					

X\*\* Recommend 0

**Table C-4 — OScan0 scan sequence/DR Scan (010) and IR Scan (111), delay as specified**

Virtual TAPC state	nTDI	TMS	RDY	TDO	Delay
<b>TCKC →</b>					
<i>Run/Test/Idle</i>	X	0	1	1	As specified
<i>Run/Test/Idle</i>	X	1	1	1	
<i>Select-DR-Scan</i>	X	0	1	1	
<i>Capture-DR</i>	X	0	1	1	
<i>Shift-DR</i>	1	0	1	DATA	
<i>Shift-DR</i>	0	0	1	DATA	
<i>Shift-DR</i>	1	1	1	DATA	
<i>Exit1-DR</i>	X	0	1	1	
<i>Pause-DR</i>	X	0	1	1	
<i>Pause-DR</i>	X	0	1	1	
<i>Pause-DR</i>	X	1	1	1	
<i>Exit2-DR</i>	X	1	1	1	
<i>Update-DR</i>	X	1	1	1	
<i>Select-DR-Scan</i>	X	1	1	1	
<i>Select-IR-Scan</i>	X	0	1	1	
<i>Capture-IR</i>	X	0	1	1	
<i>Shift-IR</i>	0	0	1	DATA	
<i>Shift-IR</i>	0	0	1	DATA	
<i>Shift-IR</i>	0	1	1	DATA	
<i>Exit1-IR</i>	X	0	1	1	
<i>Pause-IR</i>	X	0	1	1	
<i>Pause-IR</i>	X	0	1	1	
<i>Pause-IR</i>	X	1	1	1	
<i>Exit2-IR</i>	X	1	1	1	
<i>Update-IR</i>	X	0	1	1	
<i>Run/Test/Idle</i>	X	0	1	1	
<i>Run/Test/Idle</i>	X	0	1	1	

**Table C-5 — SP/CP combination following an OScan0 SP**

		SP				
		nTDI	TMS	RDY	TDO	
<b>TCKC →</b>						
- Ready check		X	X	1	X	
CP						
<b>TCKC →</b>						
- Four states		X	0	0	0	—
- Five states		X	1	0	0	0
- Six states		X	0	1	0	0
After CP						
		STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-6 — OScan1 scan sequence/DR Scan (010) and IR Scan (111), delay as specified**

Virtual TAPC state	nTDI	TMS	TDO	Delay
<b>TCKC →</b>				
<i>Run/Test/Idle</i>	X	0	1	As specified
<i>Run/Test/Idle</i>	X	1	1	
<i>Select-DR-Scan</i>	X	0	1	
<i>Capture-DR</i>	X	0	1	
<i>Shift-DR</i>	1	0	DATA	
<i>Shift-DR</i>	0	0	DATA	
<i>Shift-DR</i>	1	1	DATA	
<i>Exit1-DR</i>	X	0	1	
<i>Pause-DR</i>	X	0	1	
<i>Pause-DR</i>	X	0	1	
<i>Pause-DR</i>	X	1	1	
<i>Exit2-DR</i>	X	1	1	
<i>Update-DR</i>	X	1	1	
<i>Select-DR-Scan</i>	X	1	1	
<i>Select-IR-Scan</i>	X	0	1	
<i>Capture-IR</i>	X	0	1	
<i>Shift-IR</i>	0	0	DATA	
<i>Shift-IR</i>	0	0	DATA	
<i>Shift-IR</i>	0	1	DATA	
<i>Exit1-IR</i>	X	0	1	
<i>Pause-IR</i>	X	0	1	
<i>Pause-IR</i>	X	1	1	
<i>Exit2-IR</i>	X	1	1	
<i>Update-IR</i>	X	0	1	
<i>Run/Test/Idle</i>	X	0	1	
<i>Run/Test/Idle</i>	X	0	1	

**Table C-7 — SP/CP combination following an OScan1 SP**

		SP					
		nTDI	TMS	TDO			
<b>TCKC →</b>							
<b>- Ready check</b>		X	X	X			
<b>CP</b>							
<b>TCKC →</b>							
<b>- Four states</b>		X	0	0	0	—	—
<b>- Five states</b>		X	1	0	0	0	—
<b>- Six states</b>		X	0	1	0	0	0
<b>After CP</b>							
STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format							

**Table C-8 — OScan2 scan sequence/DR Scan (010) and IR Scan (111) with no delay**

Virtual TAPC state	nTDI	TMS	TDO
<b>TCKC →</b>	◻	◻	◻
<i>Run/Test/Idle</i>	—	0	—
<i>Run/Test/Idle</i>	—	1	—
<i>Select-DR-Scan</i>	—	0	—
<i>Capture-DR</i>	—	0	—
<i>Shift-DR</i>	1	0	DATA
<i>Shift-DR</i>	0	0	DATA
<i>Shift-DR</i>	1	1	DATA
<i>Exit1-DR</i>	—	0	—
<i>Pause-DR</i>	—	0	—
<i>Pause-DR</i>	—	0	—
<i>Pause-DR</i>	—	1	—
<i>Exit2-DR</i>	—	1	—
<i>Update-DR</i>	—	1	—
<i>Select-DR-Scan</i>	—	1	—
<i>Select-IR-Scan</i>	—	0	—
<i>Capture-IR</i>	—	0	—
<i>Shift-IR</i>	0	0	DATA
<i>Shift-IR</i>	0	0	DATA
<i>Shift-IR</i>	0	1	DATA
<i>Exit1-IR</i>	—	0	—
<i>Pause-IR</i>	—	0	—
<i>Pause-IR</i>	—	0	—
<i>Pause-IR</i>	—	1	—
<i>Exit2-IR</i>	—	1	—
<i>Update-IR</i>	—	0	—
<i>Run/Test/Idle</i>	—	0	—
<i>Run/Test/Idle</i>	—	0	—

**Table C-9 — OScan2 scan sequence/DR Scan (010) and IR Scan (111) with delay**

Virtual TAPC state	nTDI	TMS	TDO	Delay 1	Delay more
<b>TCKC →</b>					
<i>Run/Test/Idle</i>	—	0	—	HI (0)	—
<i>Run/Test/Idle</i>	—	1	—	HI (1)	—
<i>Select-DR-Scan</i>	—	0	—	HI (0)	—
<i>Capture-DR</i>	—	0	—	HI (0)	—
<i>Shift-DR</i>	1	0	DATA	HI (D)	—
<i>Shift-DR</i>	0	0	DATA	HI (D)	—
<i>Shift-DR</i>	1	1	DATA	HI (D)	—
<i>Exit1-DR</i>	—	0	—	HI (0)	—
<i>Pause-DR</i>	—	0	—	HI (0)	—
<i>Pause-DR</i>	—	0	—	HI (0)	—
<i>Pause-DR</i>	—	1	—	HI (1)	—
<i>Exit2-DR</i>	—	1	—	HI (1)	—
<i>Update-DR</i>	—	1	—	HI (1)	—
<i>Select-DR-Scan</i>	—	1	—	HI (1)	—
<i>Select-IR-Scan</i>	—	0	—	HI (0)	—
<i>Capture-IR</i>	—	0	—	HI (0)	—
<i>Shift-IR</i>	0	0	DATA	HI (D)	—
<i>Shift-IR</i>	0	0	DATA	HI (D)	—
<i>Shift-IR</i>	0	1	DATA	HI (D)	—
<i>Exit1-IR</i>	—	0	—	HI (0)	—
<i>Pause-IR</i>	—	0	—	HI (0)	—
<i>Pause-IR</i>	—	0	—	HI (0)	—
<i>Pause-IR</i>	—	1	—	HI (1)	—
<i>Exit2-IR</i>	—	1	—	HI (1)	—
<i>Update-IR</i>	—	0	—	HI (0)	—
<i>Run/Test/Idle</i>	—	0	—	HI (0)	—
<i>Run/Test/Idle</i>	—	0	—	HI (0)	—

**Table C-10 — SP/CP combination following an OScan2 SP**

	SP					
	<b>TMS</b>					
<b>TCKC →</b>						
- Ready check	X					
	CP					
<b>TCKC →</b>						
- Four states	1	0	0	0	—	—
- Five states	1	1	0	0	0	—
- Six states	1	0	1	0	0	0
	After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format					

**Table C-11 — OScan3 scan sequence/DR Scan (010) and IR Scan (111) with no delay**

<b>Virtual TAPC state</b>	<b>nTDI</b>	<b>TMS</b>
<b>TCKC →</b>		
<i>Run/Test/Idle</i>	—	0
<i>Run/Test/Idle</i>	—	1
<i>Select-DR-Scan</i>	—	0
<i>Capture-DR</i>	—	0
<i>Shift-DR</i>	1	0
<i>Shift-DR</i>	0	0
<i>Shift-DR</i>	1	1
<i>Exit1-DR</i>	—	0
<i>Pause-DR</i>	—	0
<i>Pause-DR</i>	—	0
<i>Pause-DR</i>	—	1
<i>Exit2-DR</i>	—	1
<i>Update-DR</i>	—	1
<i>Select-DR-Scan</i>	—	1
<i>Select-IR-Scan</i>	—	0
<i>Capture-IR</i>	—	0
<i>Shift-IR</i>	0	0
<i>Shift-IR</i>	0	0
<i>Shift-IR</i>	0	1
<i>Exit1-IR</i>	—	0
<i>Pause-IR</i>	—	0
<i>Pause-IR</i>	—	0
<i>Pause-IR</i>	—	1
<i>Exit2-IR</i>	—	1
<i>Update-IR</i>	—	0
<i>Run/Test/Idle</i>	—	0
<i>Run/Test/Idle</i>	—	0

**Table C-12 — OScan3 scan sequence/DR Scan (010) and IR Scan (111) with delay**

Virtual TAPC state	nTDI	TMS	TDO	Delay 1	Delay more
<b>TCKC →</b>					
<i>Run/Test/Idle</i>	—	0	—	HI-Z	—
<i>Run/Test/Idle</i>	—	1	—	HI-Z	—
<i>Select-DR-Scan</i>	—	0	—	HI-Z	—
<i>Capture-DR</i>	—	0	—	HI-Z	—
<i>Shift-DR</i>	1	0	D	HI-Z	—
<i>Shift-DR</i>	0	0	D	HI-Z	—
<i>Shift-DR</i>	1	1	D	HI-Z	—
<i>Exit1-DR</i>	—	0	—	HI-Z	—
<i>Pause-DR</i>	—	0	—	HI-Z	—
<i>Pause-DR</i>	—	0	—	HI-Z	—
<i>Pause-DR</i>	—	1	—	HI-Z	—
<i>Exit2-DR</i>	—	1	—	HI-Z	—
<i>Update-DR</i>	—	1	—	HI-Z	—
<i>Select-DR-Scan</i>	—	1	—	HI-Z	—
<i>Select-IR-Scan</i>	—	0	—	HI-Z	—
<i>Capture-IR</i>	—	0	—	HI-Z	—
<i>Shift-IR</i>	0	0	D	HI-Z	—
<i>Shift-IR</i>	0	0	D	HI-Z	—
<i>Shift-IR</i>	0	1	D	HI-Z	—
<i>Exit1-IR</i>	—	0	—	HI-Z	—
<i>Pause-IR</i>	—	0	—	HI-Z	—
<i>Pause-IR</i>	—	0	—	HI-Z	—
<i>Pause-IR</i>	—	1	—	HI-Z	—
<i>Exit2-IR</i>	—	1	—	HI-Z	—
<i>Update-IR</i>	—	0	—	HI-Z	—
<i>Run/Test/Idle</i>	—	0	—	HI-Z	—
<i>Run/Test/Idle</i>	—	0	—	HI-Z	—

**Table C-13 — SP/CP combination following an OScan3 SP**

	SP					
	<b>TMS</b>					
<b>TCKC →</b>						
- Ready check	X					
<b>CP</b>						
<b>TCKC →</b>						
- Four states	1	0	0	0	—	—
- Five states	1	1	0	0	0	—
- Six states	1	0	1	0	0	0
After CP						
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format					

**Table C-14 — OScan4 scan sequence/DR Scan (010) and IR Scan (111)**

Virtual TAPC state	nTDI	TMS	LRDY	TDO	Delay
<b>TCKC →</b>					
<i>Run/Test/Idle</i>	X	0	1	1	
<i>Run/Test/Idle</i>	X	1	1	1	
<i>Select-DR-Scan</i>	X	0	1	1	
<i>Capture-DR</i>	X	0	1	1	
<i>Shift-DR</i>	1	—	1	DATA	
<i>Shift-DR</i>	0	—	1	DATA	
<i>Shift-DR</i>	1 + EOT	—	1	DATA	
<i>Exit1-DR</i>	X	0	1	1	
<i>Pause-DR</i>	X	0	1	1	
<i>Pause-DR</i>	X	0	1	1	
<i>Pause-DR</i>	X	1	1	1	
<i>Exit2-DR</i>	X	1	1	1	
<i>Update-DR</i>	X	1	1	1	
<i>Select-DR-Scan</i>	X	1	1	1	
<i>Select-IR-Scan</i>	X	0	1	1	
<i>Capture-IR</i>	X	0	1	1	
<i>Shift-IR</i>	0	—	1	DATA	
<i>Shift-IR</i>	0	—	1	DATA	
<i>Shift-IR</i>	0 + EOT	—	1	DATA	
<i>Exit1-IR</i>	X	0	1	1	
<i>Pause-IR</i>	X	0	1	1	
<i>Pause-IR</i>	X	0	1	1	
<i>Pause-IR</i>	X	1	1	1	
<i>Exit2-IR</i>	X	1	1	1	
<i>Update-IR</i>	X	0	1	1	
<i>Run/Test/Idle</i>	X	0	1	1	
<i>Run/Test/Idle</i>	X	0	1	1	

As specified

**Table C-15 — SP/CP combination following an OScan4 SP**

		SP				
	nTDI	TMS	RDY	TDO		
<b>TCKC →</b>						
- Ready check	X	X	1	X		
CP						
<b>TCKC →</b>						
- Four states	1	0	0	0	—	—
- Five states	1	1	0	0	0	—
- Six states	1	0	1	0	0	0
	After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format					

**Table C-16 — OScan5 scan sequence/DR Scan (010) and IR Scan (111)**

Virtual TAPC state	nTDI	TMS	TDO	Delay
<b>TCKC →</b>				
<i>Run/Test/Idle</i>	X	0	1	As specified
<i>Run/Test/Idle</i>	X	1	1	
<i>Select-DR-Scan</i>	X	0	1	
<i>Capture-DR</i>	X	0	1	
<i>Shift-DR</i>	1	—	DATA	
<i>Shift-DR</i>	0	—	DATA	
<i>Shift-DR</i>	1 + EOT	—	DATA	
<i>Exit1-DR</i>	X	0	1	
<i>Pause-DR</i>	X	0	1	
<i>Pause-DR</i>	X	0	1	
<i>Pause-DR</i>	X	1	1	
<i>Exit2-DR</i>	X	1	1	
<i>Update-DR</i>	X	1	1	
<i>Select-DR-Scan</i>	X	1	1	
<i>Select-IR-Scan</i>	X	0	1	
<i>Capture-IR</i>	X	0	1	
<i>Shift-IR</i>	0	—	DATA	
<i>Shift-IR</i>	0	—	DATA	
<i>Shift-IR</i>	0 + EOT	—	DATA	
<i>Exit1-IR</i>	X	0	1	
<i>Pause-IR</i>	X	0	1	
<i>Pause-IR</i>	X	0	1	
<i>Pause-IR</i>	X	1	1	
<i>Exit2-IR</i>	X	1	1	
<i>Update-IR</i>	X	0	1	
<i>Run/Test/Idle</i>	X	0	1	
<i>Run/Test/Idle</i>	X	0	1	

**Table C-17 — SP/CP combination following an OScan5 SP**

	SP					
	nTDI	TMS	TDO			
<b>TCKC →</b>						
- Ready check	X	X	X			
	CP					
<b>TCKC →</b>						
- Four states	1	0	0	0	—	—
- Five states	1	1	0	0	0	—
- Six states	1	0	1	0	0	0
	After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format					

**Table C-18 — OScan6 scan sequence/DR Scan (010) and IR Scan (111) with no delay**

Virtual TAPC state	nTDI	TMS	TDO
TCKC →	◻	◻	◻
<i>Run/Test/Idle</i>	—	0	—
<i>Run/Test/Idle</i>	—	1	—
<i>Select-DR-Scan</i>	—	0	—
<i>Capture-DR</i>	—	0	—
<i>Shift-DR</i>	1	Don't Care	DATA
<i>Shift-DR</i>	0	—	DATA
<i>Shift-DR</i>	1 + EOT	—	DATA
<i>Exit1-DR</i>	—	0	—
<i>Pause-DR</i>	—	0	—
<i>Pause-DR</i>	—	0	—
<i>Pause-DR</i>	—	1	—
<i>Exit2-DR</i>	—	1	—
<i>Update-DR</i>	—	1	—
<i>Select-DR-Scan</i>	—	1	—
<i>Select-IR-Scan</i>	—	0	—
<i>Capture-IR</i>	—	0	—
<i>Shift-IR</i>	0	Don't Care	DATA
<i>Shift-IR</i>	0	—	DATA
<i>Shift-IR</i>	0 + EOT	—	DATA
<i>Exit1-IR</i>	—	0	—
<i>Pause-IR</i>	—	0	—
<i>Pause-IR</i>	—	0	—
<i>Pause-IR</i>	—	1	—
<i>Exit2-IR</i>	—	1	—
<i>Update-IR</i>	—	0	—
<i>Run/Test/Idle</i>	—	0	—
<i>Run/Test/Idle</i>	—	0	—

**Table C-19 — OScan6 scan sequence/DR Scan (010) and IR Scan (111) with delay**

Virtual TAPC state	nTDI	TMS	TDO	Delay 1	Delay more
TCKC →					
<i>Run/Test/Idle</i>	—	0	—	HI-Z	As specified
<i>Run/Test/Idle</i>	—	1	—	HI-Z	
<i>Select-DR-Scan</i>	—	0	—	HI-Z	
<i>Capture-DR</i>	—	0	—	HI-Z	
<i>Shift-DR</i>	1	1 (Don't Care)	DATA	HI-Z	
<i>Shift-DR</i>	0	—	DATA	HI-Z	
<i>Shift-DR</i>	1 + EOT	—	DATA	HI-Z	
<i>Exit1-DR</i>	—	0	—	HI-Z	
<i>Pause-DR</i>	—	0	—	HI-Z	
<i>Pause-DR</i>	—	0	—	HI-Z	
<i>Pause-DR</i>	—	1	—	HI-Z	
<i>Exit2-DR</i>	—	1	—	HI-Z	
<i>Update-DR</i>	—	1	—	HI-Z	
<i>Select-DR-Scan</i>	—	1	—	HI-Z	
<i>Select-IR-Scan</i>	—	0	—	HI-Z	
<i>Capture-IR</i>	—	0	—	HI-Z	
<i>Shift-IR</i>	0	1 (Don't Care)	DATA	HI-Z	
<i>Shift-IR</i>	0	—	DATA	HI-Z	
<i>Shift-IR</i>	0 + EOT	—	DATA	HI-Z	
<i>Exit1-IR</i>	—	0	—	HI-Z	
<i>Pause-IR</i>	—	0	—	HI-Z	
<i>Pause-IR</i>	—	0	—	HI-Z	
<i>Pause-IR</i>	—	1	—	HI-Z	
<i>Exit2-IR</i>	—	1	—	HI-Z	
<i>Update-IR</i>	—	0	—	HI-Z	
<i>Run/Test/Idle</i>	—	0	—	HI-Z	
<i>Run/Test/Idle</i>	—	0	—	HI-Z	

**Table C-20 — SP/CP combination following an OScan6 SP**

		SP						
		TMS						
TCKC →								
- Ready check		X						
		CP						
TCKC →								
- Four states		1	0	0	0	—	—	
- Five states		1	1	0	0	0	—	
- Six states		1	0	1	0	0	0	
		After CP						
		STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format						

**Table C-21 — OScan7 scan sequence/DR Scan (010) and IR Scan (111) with no delay**

Virtual TAPC state	nTDI	TMS	TDO
TCKC →			
<i>Run/Test/Idle</i>	—	0	—
<i>Run/Test/Idle</i>	—	1	—
<i>Select-DR-Scan</i>	—	0	—
<i>Capture-DR</i>	—	0	—
<i>Shift-DR</i>	1	—	—
<i>Shift-DR</i>	0	—	—
<i>Shift-DR</i>	1 + EOT	—	—
<i>Exit1-DR</i>	—	0	—
<i>Pause-DR</i>	—	0	—
<i>Pause-DR</i>	—	0	—
<i>Pause-DR</i>	—	1	—
<i>Exit2-DR</i>	—	1	—
<i>Update-DR</i>	—	1	—
<i>Select-DR-Scan</i>	—	1	—
<i>Select-IR-Scan</i>	—	0	—
<i>Capture-IR</i>	—	0	—
<i>Shift-IR</i>	0	—	—
<i>Shift-IR</i>	0	—	—
<i>Shift-IR</i>	0 + EOT	—	—
<i>Exit1-IR</i>	—	0	—
<i>Pause-IR</i>	—	0	—
<i>Pause-IR</i>	—	0	—
<i>Pause-IR</i>	—	1	—
<i>Exit2-IR</i>	—	1	—
<i>Update-IR</i>	—	0	—
<i>Run/Test/Idle</i>	—	0	—
<i>Run/Test/Idle</i>	—	0	—

**Table C-22 — OScan7 scan sequence/DR Scan (010) and IR Scan (111) with delay**

Virtual TAPC state	nTDI	TMS	TDO	Delay 1	Delay more
<b>TCKC →</b>					
<i>Run/Test/Idle</i>	—	0	—	HI-Z	As specified
<i>Run/Test/Idle</i>	—	1	—	HI-Z	
<i>Select-DR-Scan</i>	—	0	—	HI-Z	
<i>Capture-DR</i>	—	0	—	HI-Z	
<i>Shift-DR</i>	1	—	—	HI-Z	
<i>Shift-DR</i>	0	—	—	HI-Z	
<i>Shift-DR</i>	1 + EOT	—	—	HI-Z	
<i>Exit1-DR</i>	—	0	—	HI-Z	
<i>Pause-DR</i>	—	0	—	HI-Z	
<i>Pause-DR</i>	—	0	—	HI-Z	
<i>Pause-DR</i>	—	1	—	HI-Z	
<i>Exit2-DR</i>	—	1	—	HI-Z	
<i>Update-DR</i>	—	1	—	HI-Z	
<i>Select-DR-Scan</i>	—	1	—	HI-Z	
<i>Select-IR-Scan</i>	—	0	—	HI-Z	
<i>Capture-IR</i>	—	0	—	HI-Z	
<i>Shift-IR</i>	0	—	—	HI-Z	
<i>Shift-IR</i>	0	—	—	HI-Z	
<i>Shift-IR</i>	0 + EOT	—	—	HI-Z	
<i>Exit1-IR</i>	—	0	—	HI-Z	
<i>Pause-IR</i>	—	0	—	HI-Z	
<i>Pause-IR</i>	—	0	—	HI-Z	
<i>Pause-IR</i>	—	1	—	HI-Z	
<i>Exit2-IR</i>	—	1	—	HI-Z	
<i>Update-IR</i>	—	0	—	HI-Z	
<i>Run/Test/Idle</i>	—	0	—	HI-Z	
<i>Run/Test/Idle</i>	—	0	—	HI-Z	

**Table C-23 — SP/CP combination following an OScan7 SP**

		SP					
		TMS					
<b>TCKC →</b>							
- Ready check		X					
		CP					
<b>TCKC →</b>							
- Four states		1	0	0	0	—	—
- Five states		1	1	0	0	0	—
- Six states		1	0	1	0	0	0
		After CP					
		STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format					

### C.3 SScan SP types

This subclause provides examples for SScan Scan Formats. The RDYC Register value is a 00b for these examples. A three-bit DR Scan of with 010b data is followed by a three-bit IR Scan with 111b data. In some examples, delays are added to illustrate their effect.

A second table illustrates the Check Packet content. The body of the CP is shown as two, three, and four bits in this table. Note the ready-check portion of this table is an SP payload equivalent to that for a *Run-Test/Idle* or *Select-DR-Scan* TAP controller state less header and Delay Elements.

The location of the TAP controller state advance caused by the SP is indicated as follows:

	None
	Advance

The tables and their relationship to specific scan formats are listed as follows:

- SScan0                    Table C-24 through Table C-31
- SScan1                    Table C-32 through Table C-40
- SScan2                    Table C-41 through Table C-48
- SScan3                    Table C-49 through Table C-57

**Table C-24 — SScan0 – SP sequence following a CP/all stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	X	x	1	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-DR</i>	0	0	1	1	0	1	DATA	
<i>Shift-DR</i>	—	—	—	0	0	1	DATA	
<i>Shift-DR</i>	—	—	—	1	1	1	DATA	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-IR</i>	1	x	1	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	0	1	1	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	

**Table C-25 — SP/CP combination following an SScan0 Segment with an all SP stall profile**

SP					
	TMS	END	RDY	ONE	
<b>TCKC →</b>	□□	□□	□□	□□	
- Ready check	X	x	1	1	
CP					
<b>TCKC →</b>	□□	□□	□□	□□	□□
- Four states	1	0	0	0	—
- Five states	1	1	0	0	0
- Six states	1	0	1	0	0
After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-26 — SScan0 – SP sequence following a CP/all stall profile (IN, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-DR</i>	0	0	1	1	0	1	DATA	
<i>Shift-DR</i>	—	—	—	0	0	1	DATA	
<i>Shift-DR</i>	—	—	—	1	1	1	DATA	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-IR</i>	0	1	1	1	0	1	DATA	
<i>Shift-IR</i>	—	—	—	—	0	1	DATA	
<i>Shift-IR</i>	—	—	—	—	1	1	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	

**Table C-27 — SScan0 – SP sequence following a CP/all stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-DR</i>	1	x	1	1	0	1	DATA	
<i>Shift-DR</i>	—	—	—	0	0	1	DATA	
<i>Shift-DR</i>	—	—	—	1	1	1	DATA	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-IR</i>	0	1	1	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	—	0	1	DATA	
<i>Shift-IR</i>	—	—	—	—	1	1	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	

**Table C-28 — SScan0 – SP sequence following a CP/first stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>	□□	□□	□□	□□	□□	□□	□□	□□
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	
<i>Run-Test/Idle</i>	—	—	—	1:TMS	0	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	1	—	—	
<i>Capture-DR</i>	x	x	0	0:TMS	x	1	1	
<i>Shift-DR</i>	0	0	0	1	0	—	DATA	
<i>Shift-DR</i>	—	—	—	0	0	—	—	
<i>Shift-DR</i>	—	—	—	1	1	—	—	
<i>Exit1-DR</i>	1	1	0	0:TMS	0	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	1	—	—	
<i>Pause-DR</i>	x	x	0	1:TMS	0	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	0	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	1	—	—	
<i>Select-DR-Scan</i>	x	x	0	1:TMS	0	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	0	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	x	—	—	
<i>Shift-IR</i>	1	x	0	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	0	1	—	DATA	
<i>Exit1-IR</i>	1	1	0	0:TMS	0	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	1	—	—	
<i>Pause-IR</i>	x	x	0	1:TMS	0	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	0	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	1	—	—	
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	0	—	—	

As specified

**Table C-29 — SP/CP combination following an SScan0 segment with first SP stall profile**

		SP				
		TMS	END	RDY	ONE	
<b>TCKC →</b>		□□	□□	□□	□□	
- Ready check		x	x	1	1	
CP						
<b>TCKC →</b>		□□	□□	□□	□□	□□
- Four states		1	0	0	0	—
- Five states		1	1	0	0	0
- Six states		1	0	1	0	0
After CP						
STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format						

**Table C-30 — SScan0 – SP sequence following a CP/first stall profile (IN, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	
<i>Run-Test/Idle</i>	—	—	—	1:TMS	0	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	1	—	—	
<i>Capture-DR</i>	x	x	0	0:TMS	0	1	1	
<i>Shift-DR</i>	0	0	0	1	0	—	DATA	
<i>Shift-DR</i>	—	—	—	0	0	—	—	
<i>Shift-DR</i>	—	—	—	1	1	—	—	
<i>Exit1-DR</i>	1	1	0	0:TMS	0	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-DR</i>	x	x	0	1:TMS	0	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	0	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	1	—	—	
<i>Select-DR-Scan</i>	x	x	0	1:TMS	0	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	0	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	0	—	—	
<i>Shift-IR</i>	0	1	0	1	0	1	DATA	
<i>Shift-IR</i>	—	—	—	—	0	—	DATA	
<i>Shift-IR</i>	—	—	—	—	1	—	DATA	
<i>Exit1-IR</i>	1	1	0	0:TMS	0	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	1	—	—	
<i>Pause-IR</i>	x	x	0	1:TMS	0	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	0	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	1	—	—	
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	0	—	—	

As specified

**Table C-31 — SScan0 – SP sequence following a CP/first stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>								
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	
<i>Run-Test/Idle</i>	—	—	—	1:TMS	0	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	1	—	—	
<i>Capture-DR</i>	x	x	0	0:TMS	0	1	1	
<i>Shift-DR</i>	1	x	0	1	0	—	DATA	
<i>Shift-DR</i>	—	—	—	0	0	—	DATA	
<i>Shift-DR</i>	—	—	—	1	1	—	DATA	
<i>Exit1-DR</i>	1	1	0	0:TMS	0	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-DR</i>	x	x	0	1:TMS	0	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	0	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	1	—	—	
<i>Select-DR-Scan</i>	x	x	0	1:TMS	0	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	0	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	0	—	—	
<i>Shift-IR</i>	0	1	0	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	0	1	—	DATA	
<i>Exit1-IR</i>	x	x	0	0:TMS	0	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	0	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	1	—	—	
<i>Pause-IR</i>	x	x	0	1:TMS	0	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	0	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	1	—	—	
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	0	—	—	

As specified

**Table C-32 — SScan1 – SP sequence following a CP/no stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>	□□	□□	□□	□□	□□	□□	□□	□□
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	—	—	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	1	1	0	—	—	
<i>Shift-DR</i>	—	—	—	0	0	—	—	
<i>Shift-DR</i>	—	—	—	1	1	—	—	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	1	x	1	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	0	1	—	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-33 — SP/CP combination following an SScan1 segment with a no stall profile**

<b>SP</b>					
	TMS	END	RDY	ONE	
<b>TCKC →</b>	□□	□□	□□	□□	
- Ready check	x	x	—	—	
<b>CP</b>					
<b>TCKC →</b>	□□	□□	□□	□□	□□
- Four states	1	0	0	0	—
- Five states	1	1	0	0	0
- Six states	1	0	1	0	0
<b>After CP</b>					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-34 — SScan1 – SP sequence following a CP/no stall profile (IN, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	—	—	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	1	1	0	—	—	
<i>Shift-DR</i>	—	—	—	0	0	—	—	
<i>Shift-DR</i>	—	—	—	1	1	—	—	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	1	x	1	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	—	0	—	DATA	
<i>Shift-IR</i>	—	—	—	—	1	—	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-35 — SScan1 – SP sequence following a CP/no stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC →</b>								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	—	—	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	1	x	1	1	0	—	DATA	
<i>Shift-DR</i>	—	—	—	0	0	—	DATA	
<i>Shift-DR</i>	—	—	—	1	1	—	DATA	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	1	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	—	0	—	DATA	
<i>Shift-IR</i>	—	—	—	—	1	—	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-36 — SScan1 – SP sequence following a CP/first stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	1	x	1	1	0	1	DATA	
<i>Shift-DR</i>	—	—	—	0	0	—	DATA	
<i>Shift-DR</i>	—	—	—	1	1	—	DATA	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	1	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	—	0	—	DATA	
<i>Shift-IR</i>	—	—	—	—	1	—	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

As specified

**Table C-37 — SScan1 – SP sequence following a CP/first stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	0	1	0	1	DATA	
<i>Shift-DR</i>	—	—	—	0	0	—	—	
<i>Shift-DR</i>	—	—	—	1	1	—	—	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	0	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	0	1	—	DATA	
<i>Exit1-IR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

As specified

**Table C-38 — SP/CP combination following an SScan1 segment with first SP stall profile**

SP					
	TMS	END	RDY	ONE	
<b>TCKC→</b>					
- Ready check	x	x	1	1	
CP					
<b>TCKC→</b>					
- Four states	1	0	0	0	—
- Five states	1	1	0	0	0
- Six states	1	0	1	0	0
After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-39 — SScan1 – SP sequence following a CP/first stall profile (IN, OUT) transactions**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
TCKC→	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	0	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	0	1	0	1	DATA	
<i>Shift-DR</i>	—	—	—	0	0	—	—	
<i>Shift-DR</i>	—	—	—	1	1	—	—	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	1	0	0	1	DATA	
<i>Shift-IR</i>	—	—	—	0	0	—	DATA	
<i>Shift-IR</i>	—	—	—	0	1	—	DATA	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

As specified

**Table C-40 — SScan1 – SP sequence following a CP/first stall profile/I/O, OUT transactions**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	0	0:TMS	0	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	1	x	0	1	0	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	0	—	DATA(1)	
<i>Shift-DR</i>	—	—	—	1	1	—	DATA(2)	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	0	0	0	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	0	—	DATA(1)	
<i>Shift-IR</i>	—	—	—	—	1	—	DATA(2)	
<i>Exit1-IR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-41 — SScan2 – SP sequence following a CP/all stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-DR</i>	0	0	1	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	1	DATA(1)	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	1	DATA(2)	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-IR</i>	1	x	1	0	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	0	—	1	DATA(1)	
<i>Shift-IR</i>	—	—	—	0 + EOT	—	1	DATA(2)	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	

**Table C-42 — SP/CP combination following an SScan2 segment with an all SP stall profile**

SP					
	TMS	END	RDY	ONE	
<b>TCKC→</b>					
- Ready check	x	—	1	1	
CP					
<b>TCKC→</b>					
- Four states	1	0	0	0	—
- Five states	1	1	0	0	0
- Six states	1	0	1	0	0
After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-43 — SScan2 – SP sequence following a CP/all stall profile (IN, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-DR</i>	0	0	1	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	1	DATA(1)	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	1	DATA(2)	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-IR</i>	0	1	1	1	—	1 + EOT	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	1	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	1	DATA(2)	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	

As specified

**Table C-44 — SScan2 – SP sequence following a CP/all stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
TCKC→								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-DR</i>	1	x	1	1:TMS	0	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	0	1	DATA(1)	
<i>Shift-DR</i>	—	—	—	1+EOT	1	1	DATA(2)	
<i>Exit1-DR</i>	1	1	1	0	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-DR</i>	—	—	—	1:TMS	—	1	1	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	1	1	
<i>Capture-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Shift-IR</i>	0	1	1	0	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	1 + EOT	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	1	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	1	DATA(2)	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	1	1	
<i>Update-IR</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	1	1	

**Table C-45 — SScan2 – SP sequence following a CP/first stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	0	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	0	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	—	DATA(2)	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	1	x	0	0	—	—	DATA(0)	
<i>Shift-IR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-IR</i>	—	—	—	0 + EOT	—	—	DATA(2)	
<i>Exit1-IR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-46 — SP/CP combination following an SScan2 segment with first SP stall profile**

SP					
	TMS	END	RDY	ONE	
<b>TCKC→</b>					
- Ready check	x	x	1	1	
CP					
<b>TCKC→</b>					
- Four states	1	0	0	0	—
- Five states	1	1	0	0	0
- Six states	1	0	1	0	0
After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-47 — SScan2 – SP sequence following a CP/first stall profile (IN, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
TCKC→								
<i>Run-Test/Idle</i>	x	x	0	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	0	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-DR</i>	—	—	—	1	—	—	DATA(2)	
<i>Exit1-DR</i>	1	1	0	0	—	1	1	
<i>Pause-DR</i>	—	—	—	0	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	0	1	—	—	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	1+EOT	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	1	DATA(2)	
<i>Exit1-IR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-48 — SScan2 – SP sequence following a CP/first stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	0	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	1	x	0	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-DR</i>	—	—	—	1	—	—	DATA(2)	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	0	1	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	—	DATA(2)	
<i>Exit1-IR</i>	x	x	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-49 — SScan3 – SP sequence following a CP/no stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	—	—	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	1	1	—	—	—	
<i>Shift-DR</i>	—	—	—	0	—	—	—	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	—	—	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	1	x	1	0	—	—	DATA(0)	
<i>Shift-IR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-IR</i>	—	—	—	0 + EOT	—	—	DATA(2)	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-50 — SP/CP combination following an SScan3 segment with a no stall profile**

SP					
	TMS	END	RDY	ONE	
<b>TCKC→</b>					
- Ready check	x	x	—	—	
CP					
<b>TCKC→</b>					
- Four states	1	0	0	0	—
- Five states	1	1	0	0	0
- Six states	1	0	1	0	0
After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-51 — SScan3 – SP sequence following a CP/no stall profile (IN, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
TCKC→	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	—	—	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	1	1	—	—	—	
<i>Shift-DR</i>	—	—	—	0	—	—	—	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	—	—	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	1	0	—	—	DATA(0) + EOT	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	—	DATA(2)	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-52 — SScan3 – SP sequence following a CP/no stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
TCKC→	□□	□□	□□	□□	□□	□□	□□	□□
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	—	—	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	1	x	1	1	—	—	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	—	DATA(2)	
<i>Exit1-DR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	1	0	—	—	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(0) + EOT	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	—	DATA(2)	
<i>Exit1-IR</i>	1	1	1	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-53 — SScan3 – SP sequence following a CP/first stall profile (I/O, OUT)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
TCKC→								
<i>Run-Test/Idle</i>	x	x	1	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	1	x	0	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	—	DATA(2)	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	0	0	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(0) + EOT	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	—	DATA(2)	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-54 — SScan3 – SP sequence following a CP/first stall profile (IN, I/O)**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	0	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	0	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	—	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	—	—	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	1	x	0	0	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-IR</i>	—	—	—	0 + EOT	—	—	DATA(2)	
<i>Exit1-IR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-55 — SP/CP combination following an SScan3 segment with first SP stall profile**

SP					
	TMS	END	RDY	ONE	
<b>TCKC→</b>	□□	□□□	□□	□□	
- Ready check	x	x	1	1	
CP					
<b>TCKC→</b>	□□□	□□□	□□	□□□	□□□
- Four states	1	0	0	0	—
- Five states	1	1	0	0	0
- Six states	1	0	1	0	0
After CP					
	STD protocol or an SP (for any scan format) SP contains header if SScan Scan Format				

**Table C-56 — SScan3 – SP sequence following a CP/first stall profile (IN, OUT) transactions**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
TCKC→	□	□	□	□	□	□	□	□
<i>Run-Test/Idle</i>	x	x	0	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	0	0	0	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	—	
<i>Shift-DR</i>	—	—	—	1	—	—	—	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	0	0	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(0) + EOT	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	—	DATA(2)	
<i>Exit1-IR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

**Table C-57 — SScan3 – SP sequence following a CP/first stall profile/I/O, OUT transactions**

Virtual TAPC state	HDR[0]	HDR[1]	HDR[2]	nTDI/TMS	END	RDY	TDO/ONE	Delay
<b>TCKC→</b>								
<i>Run-Test/Idle</i>	x	x	0	0:TMS	—	1	1	As specified
<i>Run-Test/Idle</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-DR</i>	1	x	0	1	—	1	DATA(0)	
<i>Shift-DR</i>	—	—	—	0	—	—	DATA(1)	
<i>Shift-DR</i>	—	—	—	1 + EOT	—	—	DATA(2)	
<i>Exit1-DR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-DR</i>	—	—	—	1:TMS	—	—	—	
<i>Select-DR-Scan</i>	—	—	—	1:TMS	—	—	—	
<i>Select-IR-Scan</i>	—	—	—	0:TMS	—	—	—	
<i>Capture-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Shift-IR</i>	0	1	0	0	—	1	DATA(0)	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(0) + EOT	
<i>Shift-IR</i>	—	—	—	—	—	—	DATA(1)	
<i>Extra SP</i>	—	—	—	—	—	—	DATA(2)	
<i>Exit1-IR</i>	1	1	0	0:TMS	—	1	1	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Pause-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Exit2-IR</i>	—	—	—	1:TMS	—	—	—	
<i>Update-IR</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	
<i>Run-Test/Idle</i>	—	—	—	0:TMS	—	—	—	

## Annex D

(informative)

### Programming considerations

#### D.1 Overview

The concepts used to manage the DTS/TS link for both debug and test are outlined in this annex. It covers initialization of the connection, combinations of TAP.7 Controller with and without power management, and the influence of power management on start-up and steady-state operation. It also describes interrogation of the connection to determine the TAP.7 Technology Branches when connecting to an unknown system.

A summary of the TAP.7 capability when the power control and selection control are considered together is provided. This summary forms the basis for the description of programming considerations that follow. The start-up description understands the use of Series, Star-4, and Star-2 Branches both separately and together. It assumes a scenario where the following is true:

- Scan topology is unknown
- Use of both TAP.1s and T0 TAP.7s
- Use of TAP.7 Controller selection
- Use of all TAP.7 Controller power states

The test scenario is simpler as the scan topology is known. The aspects of this description dealing with scan topology interrogation may be skipped, in this case. If the DTS already knows the scan topology at start-up, it may also skip the aspects of this description dealing with scan topology interrogation. When the Scan Topology is formed with a single branch, the management of the DTS connection simplifies further. The TAP.7 power-management capability is handled when at least one TAP.7 Controller utilizes this capability.

The DTS-generated signaling and command sequences in the next subclauses consider the following:

- Multiple TAP.7 Branches
- The use of T0–T5 TAP.7 Controllers in Series Branches
- The use of T3–T5 TAP.7 Controllers in Star-4 Branches
- The use of T4–T5 TAP.7 Controllers in Star-2 Branches
- Start-up of an unknown system
- Placing a TAP.7 Controller Online while operating
- CID allocation and management

#### D.2 DTS' view of TAPs

##### D.2.1 Technology branches

The DTS' view of its connectivity to the TS is a connection to one or more technology branches. The TAPs within TAP.7 Technology Branches have a state that is the combination of the TAP.7 power-management states described in Clause 16 and the CSM states described in Clause 11. A brief overview of the TAP.7

architecture's connectivity, power management, and TAP.7 Controller's selection facilities is provided to aid the understanding of TAP.7 Controller programming considerations described in this annex.

As described in Clause 7, the DTS connectivity may include Series, Star-4, and Star-2 Technology Branches along with other technology branches. The TAP.7 Controllers of T0–T2 TAP.7s may be constructed with or without an RSU. TAP.7 Controllers without an RSU cannot be mixed with Star-4, Star-2, and other technology branches.

A Star-4 Branch is always selectable as it is constructed with only TAPs whose TAP.7 Controllers contain an RSU (T3, T4(W), and T5(W) TAP.7s). A Star-2 Branch is also always selectable as it is constructed with TAPs whose TAP.7 Controllers contain an RSU (T4 and T5 TAP.7s). A Series Branch may be constructed using combinations of the following:

- TAP.1s
- T0–T2 TAP.7s without an RSU
- T0–T2 TAP.7s with an RSU
- T3 TAP.7
- T4 (W) TAP.7
- T5 (W) TAP.7

When a Series Branch contains a TAP.1 or T0–T2 TAP.7 without an RSU, the branch is not selectable and is the only branch connected to the DTS. A Series Branch is selectable otherwise.

## D.2.2 Power-management RSU combinations

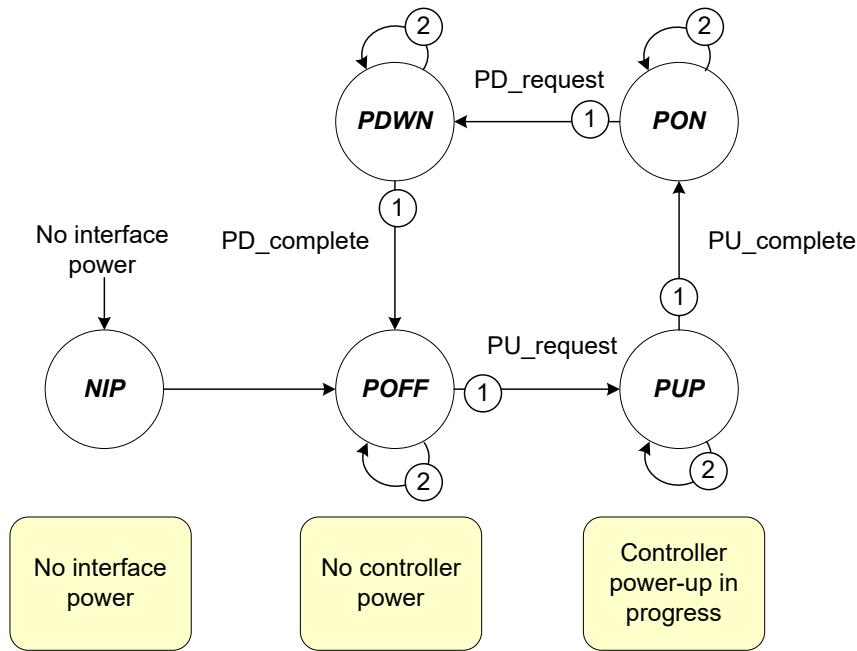
A TAP.7 Controller may be built with the combinations of TAP.7 Controller Power Management and RSU shown in Table D-1. Consideration is given to an additional power state when the TAP.7 Controller power-management logic at the chip level can remove power from the TAP input and output buffers. This state is called No Interface Power (NIP). The TAP.7 Controller cannot be powered in this state as the signaling needed to initiate power-up is inoperable in this state.

In the simplest case, a T0–T2 TAP.7 has neither selectability nor power management and has characteristics that are similar to those of a TAP.1. An RSU may be added to a T0–T5 TAP.7 to provide TAP.7 Controller selectability. Power management may be added to a T1–T5 TAP.7.

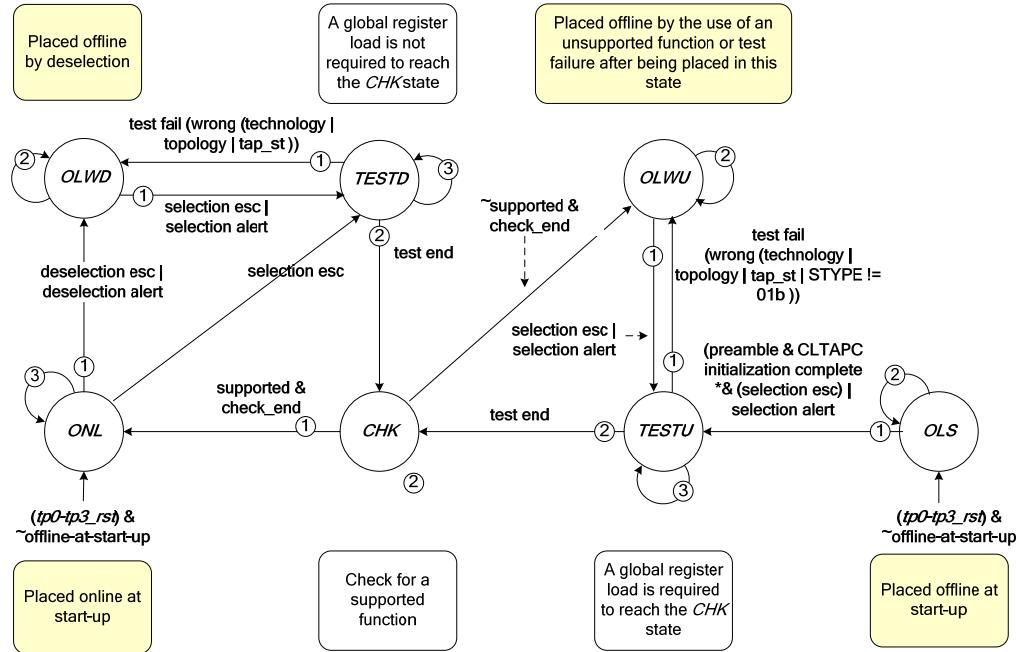
**Table D-1 — TAP power and selection states viewed by the DTS**

TAP.7	Power mgmt.	RSU	Power management states	Selection states
T0-T2	No	No	<i>PON</i>	Online
T0-T5	No	Yes	<i>PON</i>	Online Offline by Escape or alert Offline by use of unsupported feature Offline-at-Start-up
T1-T2	Yes	No	<i>PON, PUP, POFF, PDWN, NIP</i>	Online N/A
T1-T5	Yes	Yes	<i>PON</i>	Online Offline by Escape or alert Offline by unsupported feature use Offline-at-Start-up
			<i>PUP, POFF, PDWN, NIP</i>	N/A

The power-management states shown in Figure D-1 are dominant. The CSM states shown in Figure D-2 are considered substates of the *PON* Power-Management State as the CSM is reset by a Type-0 Reset generated when the power state is a state other than *PON*. Note that the *NIP* state is not permitted in a Series Scan Topology during the start-up sequence. It is permitted, however, with Star-4 and Star-2 Scan Topologies.



**Figure D-1 — Power-control states**



**Figure D-2 — Consolidated view of CSM states**

The **ONL** state is a collection of the **ADV** and **STD** CSM states. The **CHK** state expands to the **PRE**, **BDY0**, **BDY1**, and **DIRP** states shown in Figure 11-19.

### D.2.3 TAP.7 Controller deselection and selection

Three deselection states may be utilized when RSU is supported:

- **OLS** Offline-at-Start-up
- **OLU** Offline by use of an unsupported feature
- **OLD** Offline by deselection

The **OLD** and **OLU** states can only be created during steady state operation. The **OLS** may be created by a Type-0–Type-3 Reset. The events initiating a Selection Sequence with each of these states is shown in Table D-2.

**Table D-2 — Selection-Sequence initiation**

Event	State		
	<b>OLS</b>	<b>OLU</b>	<b>OLD</b>
<b>Selection Escape + Preamble</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>
<b>Selection Escape</b>	—	<b>Yes</b>	<b>Yes</b>
<b>Selection Alert</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>

The selection criteria required by the Selection Sequence when Offline in one of these state is described in Table D-3.

**Table D-3 — Selection-Sequence criteria**

State	Criteria for placement Online				
	Technology match	Topology match	TAPC state match	Resynchronization Selection Sequence	Supported feature use
<i>OLS</i>	Yes	Yes	Yes	Yes	Yes
<i>OLU</i>	Yes	Yes	Yes	Yes	Yes
<i>OLD</i>	Yes	Yes	Yes	—	Yes

Since the criteria for placement Online are different for these states, two different test states (*TESTD* and *TESTU*) are used to impose the two sets of criteria. The *TESTD* state is used to check the criteria when the originating Offline state is *OLD* and the *TESTU* state is used to check the criteria otherwise. The TAP.7 Controller is placed Offline in the *OLU* state when the originating Offline state is *OLS* or *OLU* and any of the first four criteria are failed. The TAP.7 Controller is placed Offline in the *OLD* state when the originating Offline state is *OLD* and any of the first four criteria are failed. The Global Registers are loaded to force synchronization of the Global Register state and initiate drive conflict protection.

The DTS controls the manner in which TAP.7 Controllers that have either been placed Offline-at-Start-up or been placed Offline and lost synchronization with their peers. These TAP.7 Controllers are placed Online as shown in Table D-4. The DTS determines when it must deal with the drive considerations created by a logic 1 PROTECT bit value and the CID allocation that needs to take place with the placement of a TAP.7 Controller Online whose state was previously *OLS*. The DTS software is simplified when it is designed to follow a Selection Sequence that permits placement Online of a TAP.7 Controller that is placed Offline-at-Start-up with the following:

- A Resynchronizing Selection Sequence
- A Global Register Load with a state that defines the use of mandatory capability

Should this recommendation not be followed, subsequent Non-synchronizing Selection Sequences may need to deal with CID allocation. This situation arises because the CSM *OLS* state subsequently becomes the *OLWU* state. This makes both TAP.7 Controllers placed Offline-at-Start-up and Offline by a Change Packet appear the same, capable of being placed Online with a Non-synchronizing Selection Sequence.

Note that the action specified by the PROTECT Bit occurs independently of the value of the SHORT bit, provided the TAP.7 Controller remains a candidate for being placed Online following the completion of the EC.

**Table D-4 — Criteria for placement Online**

SHORT bit	PROTECT bit	(Selection Escape + preamble)   Selection Alert?	ADTAPCs state prior to test			
			OLN	OLD	OLW	OLS
0	0	x	Yes	Yes	No	No
0	1	No	Yes	Yes	Yes	No
0	1	Yes	Yes	Yes	Yes	Yes
1	x	x	Yes	Yes	No	No

#### D.2.4 Start-up versus steady-state operation

The start-up of the DTS connection is described separately from steady state operation. With a Series Branch, the TAP.7 Controllers:

- Are required to have the same power state when the DTS connection operates

- May have different power states prior to the DTS initiating power-up of the TAP.7 Controllers
- Are required to have the same selection states

With the Star-4 and Star-2 Scan Topologies, individual TAP.7 Controllers:

- May have different power states
- May have different selection states

These behaviors are summarized in Table D-5.

**Table D-5 — TAP power and selection states viewed by the DTS**

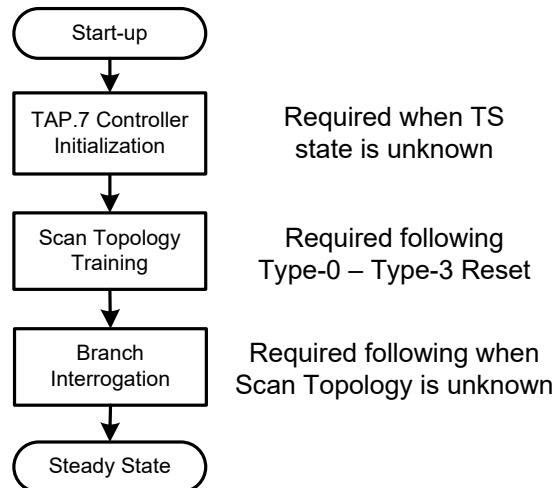
Topology branch	Steady state?	Power states are required to be the same?	Selection states are required to be the same?
Series	No	No	Yes
	Yes	Yes	At start-up only
Star-4, Star-2	No	No	No
	Yes	No	No

## D.2.5 Start-up

Start-up is a three-step process, as follows:

- Initialization of all TAP.7 Controllers
- Scan Topology Training.
- Interrogation of scan topology branches and their preparation for use

These steps are shown in Figure D-3.



**Figure D-3 — TAP initialization and topology determination**

## D.2.6 Initialization requirements

Initialization is required after any of the following actions occur:

- Connection of the DTS and TS
- DTS' reset of the TS with a Reset Escape
- DTS' reset of the TS with a Test Reset

### D.2.6.1 Initial state

When a DTS is connected to the TS, the TAPCs within the TS may be any of the power states shown in Figure D-1 or any of the CSM states shown in Figure D-2. The TAPC state may have been created by power-up of the TS, disconnecting the DTS and TS, a DTS crash, or the failure of the DTS to shutdown the TS in an orderly manner when the use of the TS is completed. This means the start-up sequence is required to understand TAPCs being in virtually any state and different TAPCs being in any state.

### D.2.6.2 Factors influencing initialization

The initialization of the TS to a point where the debug and test may begin is influenced by the following factors:

- The TAP.7 Classes utilized within the TS
- The DTS' knowledge or lack of knowledge of the scan topology
- The DTS' knowledge or lack of knowledge of the Device Identification Codes/Node Identification Code of the chips connected to each branch of the scan topology

### D.2.6.3 Initialization function

The initialization function described as follows and shown in Figure D-4 assumes the DTS has no knowledge of the factors listed above. This is the case when the DTS resets the TS after start-up without being disconnected from the TS. The initialization function includes:

- Establishing the use of the Standard Protocol using a Reset Escape
- Creating the *Test-Logic-Reset* state with a series of five TAPC states where the TMS(C) value is a logic 1 when there is no RSU)
- Power-up of any powered-down TAP.7 Controllers
- Disabling TDOC drive to prevent drive conflicts prior to placing TAP.7 Controllers that are Offline-at-Start-up Online
- Placing TAP.7 Controllers that are Offline-at-Start-up Online
- Scan Topology Training

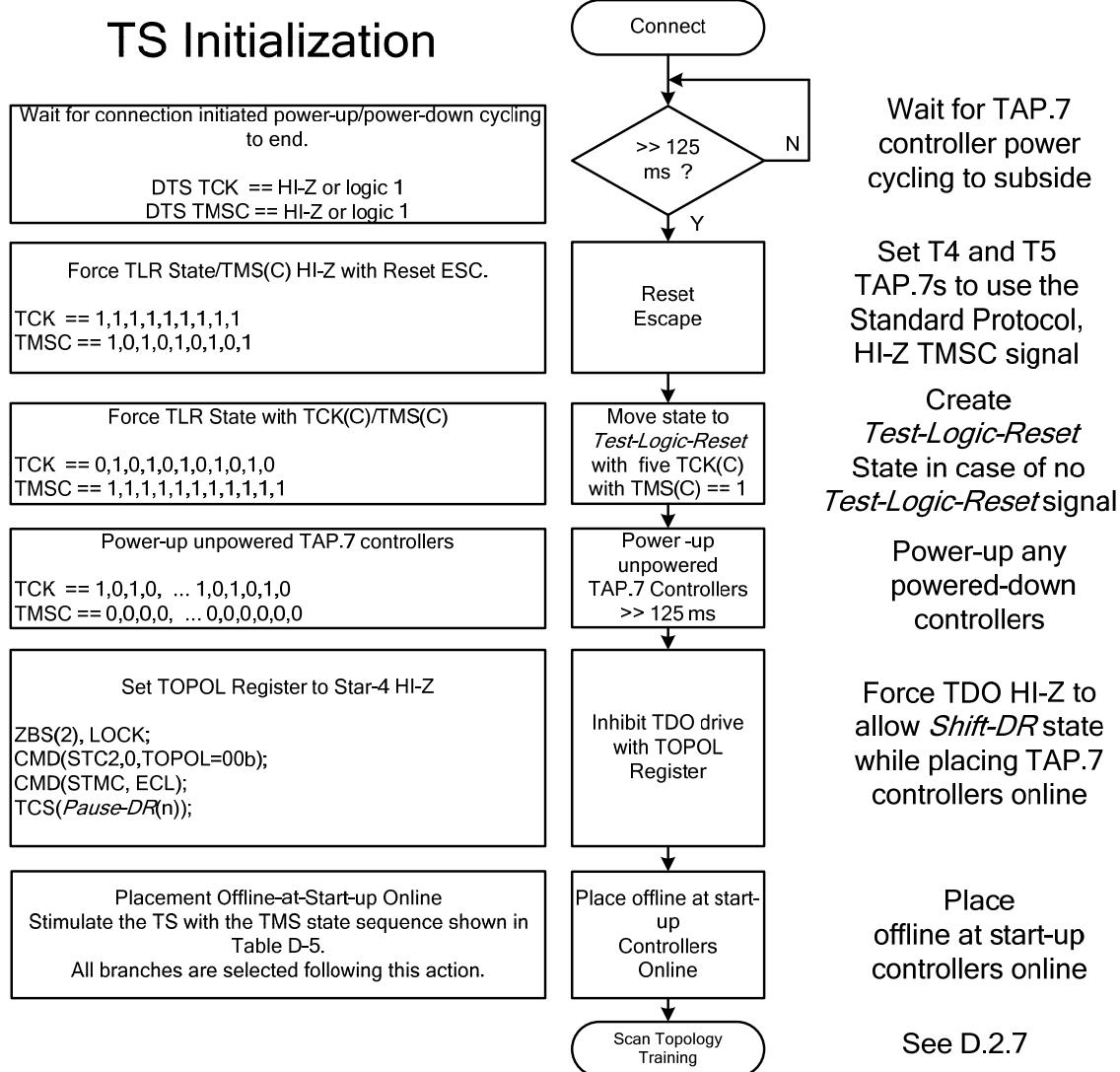


Figure D-4 — TS initialization

#### D.2.6.4 Reset Escape

TAP.7 Controllers with an RSU are reset by Reset Escape. The Reset Escape causes a Type-3 Reset with:

- Class T0–T2 TAP.7 with an RSU
- T3 and above TAP.7

It is ignored by the following:

- T0–T2 TAP.7 without an RSU
- A TAP.1

With the TCK(C) signal a logic 1, the TMSC signal cannot be driven by the TAP.7 Controller once the Reset Escape is asynchronously detected. The DTS is expected to generate eight or more TMSC signal edges, with the TMSC signal state ending as a logic 1. Making the end state of the TMSC signal a logic 1

makes the Reset Escape transparent to a TAP.7 Controller without an RSU. This step may be skipped when the TS contains only TAP.1s and T0–T2 TAP.7s without an RSU. Note that hot connection of the DTS and TS can create any number of TMSC edges while the TCKC is a logic 1. This means the Escape edge count may already be nonzero when the DTS initiates the Reset Escape. The Reset Escape does not power-up an un-powered TAP.7 Controller. It does, however, invoke the Inhibited Drive Policy, preventing the drive of the TMSC signal when the TCKC signal is a logic 0.

#### D.2.6.5 Reset TAP.7 Controllers without an RSU

TAP.7 Controllers without an RSU are reset by moving the TAPC state to *Test-Logic-Reset* state. Five or more TCK rising edges are created with the TMS(C) signal being a logic 1. This creates the following:

- The TAPC state remains in the *Test-Logic-Reset* state when it is already in this state.
- The TAPC state moves to the *Test-Logic-Reset* when it is not in this state.

Following this TAPC state sequence, a TAP.7 Controller:

- May or may not be powered
- May be Offline-at-Start-up
- Using the Standard Protocol with its TAPC state *Test-Logic-Reset*

The IEEE 1149.1 Instruction Registers within the STL contain the *BYPASS* or *IDCODE* instruction or are within logic that is not powered.

#### D.2.6.6 Power-up TAP.7 Controllers that are un-powered

With the TMS(C) signal being a logic 0, the TCK(C) signal is toggled for a period of time that is greater than 100 ms to power-up any powered-down TAP.7 Controllers. This action creates the following state:

- All TAP.7 Controllers are powered provided their I/O buffers are powered
- The TAPC state of all powered TAP.7 Controllers is the *Run-Test/Idle* state
- The default scan format of powered TAP.7 Controllers may be either JScan0 or JScan1
- Some TAP.7 Controllers may be Offline-at-Start-up

#### D.2.6.7 Placing a TAP.7 Controller Online that is Offline-at-Start-up

Should a TAP.7 Controller be Offline-at-Start-up, it must be placed Online using a Qualified Selection sequence. When TAP.1s are included in the Series Scan Topology, the Selection Sequence bit sequence used to place TAP.7 Controllers Online needs to be compatible with the TAP.1s. Two Qualified Selection Sequences with these characteristics are shown in Table D-6. The first Sequences Starts from the *Pause-DR* state while the second starts from the *Run-Test/Idle* state. Note that *only the first four* states of these sequences are different. These Selection Sequences select all technologies, with the resulting scan format being JScan2 for T2 and above TAP.7s and appears to be the equivalent of JScan0 for T0-T1 TAP.7s.

Note that with these sequences may be broken into multiple parts. Steps 0–31 may be extended in four-bit increments. Steps 33–67 may be extended in one step increments and may be further partitioned into two parts the first eight bits and the following 28 or more TMSC zero values. The sequence is concluded with the last eight bits in the sequence.

**Table D-6 — Placement of Offline-at-Start-up Online with TAPC state sequence**

Step	Preamble for Selection Sequence beginning with:			TMS(C) value	Description		
	State		Offline TAPC				
	Seq 0	Seq. 1					
0	<i>Run-Test/Idle</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[00]		
1	<i>Select-DR-Scan</i>	<i>Exit-2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[01]		
2	<i>Capture-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[02]		
3	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[03]		
4	<i>Pause-DR</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[04]		
5	<i>Exit-2-DR</i>	<i>Exit-2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[05]		
6	<i>Shift-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[06]		
7	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[07]		
8	<i>Pause-DR</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[08]		
9	<i>Exit2-DR</i>	<i>Exit2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[09]		
10	<i>Shift-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[10]		
11	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[11]		
12	<i>Pause-DR</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[12]		
13	<i>Exit-2-DR</i>	<i>Exit-2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[12]		
14	<i>Shift-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[14]		
15	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[15]		
16	<i>Pause-DR</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[16]		
17	<i>Exit2-DR</i>	<i>Exit2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[17]		
18	<i>Shift-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[18]		
19	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[19]		
20	<i>Pause-DR</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[20]		
21	<i>Exit-2-DR</i>	<i>Exit-2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[21]		
22	<i>Shift-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[22]		
23	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[23]		
24	<i>Pause-DR</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[24]		
25	<i>Exit2-DR</i>	<i>Exit2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[25]		
26	<i>Shift-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[26]		
27	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[27]		
28	<i>Pause-DR</i>	<i>Pause-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[28]		
29	<i>Exit-2-DR</i>	<i>Exit-2-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[29]		
30	<i>Shift-DR</i>	<i>Shift-DR</i>	<i>Run-Test-Idle</i>	1	Preamble[30]		
31	<i>Exit1-DR</i>	<i>Exit1-DR</i>	<i>Run-Test-Idle</i>	0	Preamble[31]		

**Table D-6 — Placement of Offline-at-Start-up Online with TAPC state sequence (continued)**

STEP	State			TMS(C) value	Description
	Seq. 0	Seq. 1	Offline TAPC		
32	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	0	ESC Cycle
33	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	0	OAC[00]
34	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	0	OAC[01]
35	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	0	OAC[02]
36	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	0	OAC[03]
37	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	0	EC[00]
38	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	0	EC[01]
39	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Run-Test-Idle</b>	1	EC[02]
40	<b>Exit2-DR</b>	<b>Exit2-DR</b>	<b>Run-Test-Idle</b>	0	EC[03]
41	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[00]
42	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[01]
43	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[02]
44	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[03]
45	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[04]
46	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[05]
47	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[06]
48	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[07]
49	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[08]
50	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[09]
51	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[10]
52	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[11]
53	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[12]
54	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[13]
55	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[15]
56	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[16]
57	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[17]
58	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[18]
59	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[19]
60	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[20]
61	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[21]
62	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[22]
63	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	GRLD[23]
64	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	PRE
65	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	BODY0
66	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	BODY1
67	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	POST
Any #	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	0	TMS
n+00	<b>Shift-DR</b>	<b>Shift-DR</b>	<b>Run-Test-Idle</b>	1	TMS
n+01	<b>Exit1-DR</b>	<b>Exit1-DR</b>	<b>Select-DR-Scan</b>	0	TMS
n+02	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Capture-DR</b>	0	TMS
n+03	<b>Pause-DR</b>	<b>Pause-DR</b>	<b>Shift-DR</b>	1	TMS
n+04	<b>Exit2-DR</b>	<b>Exit2-DR</b>	<b>Exit1-DR</b>	1	TMS
n+05	<b>Update-DR</b>	<b>Update-DR</b>	<b>Update-DR</b>	0	TMS
n+06	<b>Run-Test/Idle</b>	<b>Run-Test/Idle</b>	<b>Run-Test/Idle</b>	0	TMS
n+07	<b>Run-Test/Idle</b>	<b>Run-Test/Idle</b>	<b>Run-Test/Idle</b>	0	TMS

Selection Escape occurs in this state

The state sequences shown in Table D-8 may be used to synchronize the operation of TAPs with and without RSUs when a Qualified Selection Sequence is not required. These sequences, however, cannot be used to create Series-Equivalent Scans as they include the *Capture-xR* and *Update-xR* states. Other sequences producing *Shift-xR* synchronization may be created by modifying the TMS values at the end some of these sequences. The representations of the state names shown in Table D-8 are shown in Table D-7. Note that the TAP.7 Controller that is Online begins these sequences in the *Pause-DR* state.

**Table D-7 — TAPC state mnemonics**

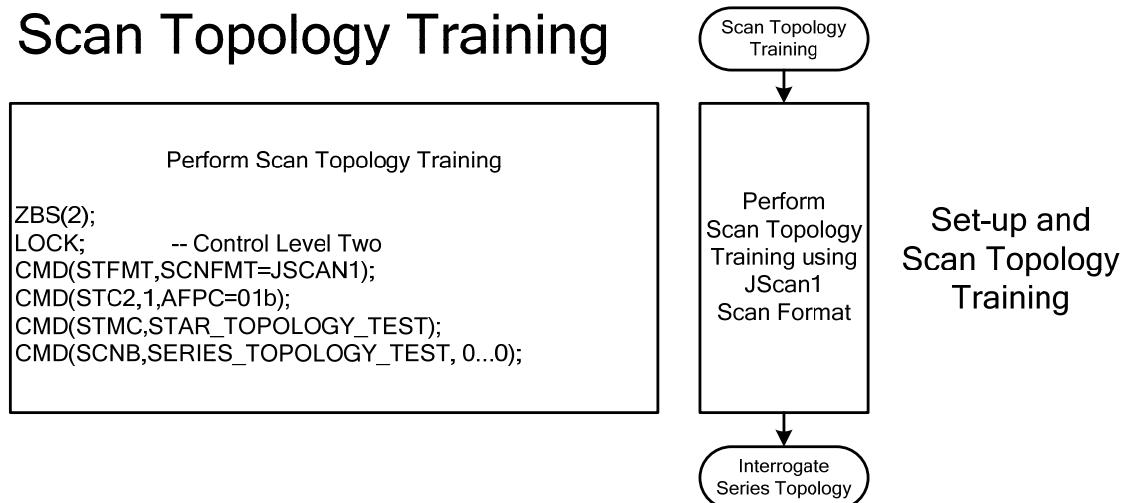
Mnemonic	Description
<b>TLR</b>	<b><i>Test-Logic-Reset</i></b>
<b>RTI</b>	<b><i>Run-Test/Idle</i></b>
<b>SL</b>	<b><i>Select-DR-Scan or Select-IR-Scan</i></b>
<b>E1</b>	<b><i>Exit1-DR or Exit1-IR</i></b>
<b>E2</b>	<b><i>Exit2-DR or Exit2-IR</i></b>
<b>PD</b>	<b><i>Pause-DR</i></b>
<b>PI</b>	<b><i>Pause-IR</i></b>
<b>SD</b>	<b><i>Shift-DR</i></b>
<b>SI</b>	<b><i>Shift-IR</i></b>
<b>CD</b>	<b><i>Capture-DR</i></b>
<b>CI</b>	<b><i>Capture-IR</i></b>
<b>UD</b>	<b><i>Update-DR</i></b>
<b>UI</b>	<b><i>Update-IR</i></b>

**Table D-8 — Other short-form synchronization state sequences**

time →																	
Synchronization of TAP.7 Controller that is Offline with an ADTAPC Run-Test/Idle state																	
#0	ESC	OAC				EAC				CP				TMS Values			
TMS	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	
ON	PD	E2	UD	RTI	RTI	RTI	RTI	RTI	RTI								
OFF	RTI	RTI	RTI														
Synchronization of TAP.7 Controller that is Offline with an ADTAPC Select-DR-Scan state																	
#1	ESC	OAC				EAC				CP				TMS Values			
TMS	0	0	0	0	0	1	0	0	1	1	0	0	1	0	1	0	0
ON	RTI	RTI	RTI	RTI	RTI	RTI	SL	CD	SD	E1	RTI	RTI	RTI	SL	CD	E1	PD
OFF	SL	CD	E1	PD													
Synchronization of TAP.7 Controller that is Offline with an ADTAPC Pause-IR state																	
#2	ESC	OAC				EAC				CP				TMS Values			
TMS	0	0	0	0	0	0	1	0	1	0	0	0	0	1	1	1	0
ON	RTI	SL	CD	E1	PD	PD	PD	PD	E1	UD	RTI						
OFF	PI	E1	UI	RTI													
Synchronization of TAP.7 Controller that is Offline with an ADTAPC Pause-DR state																	
#3	ESC	OAC				EAC				CP				TMS Values			
TMS	0	0	0	0	0	1	1	0	1	0	1	0	0	0	0	0	0
ON	PD	E2	UD	RTI	SL	CD	E1	PD	PD	PD	PD						
OFF	PD	PD	PD	PD													
Synchronization of TAP.7 Controller that is Offline with an ADTAPC Pause-DR state																	
#4	ESC	OAC				EAC				CP				TMS Values			
TMS	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0
ON	SI	E1	PI	E2	SI	SI	SI	E1	PI	PI	PI						
OFF	PD	PD	PD	PD													

## D.2.7 Scan Topology Training

Scan Topology Training is performed using the JScan1 Scan Format with the function of the TDIC and TDOC pins set to the TDI and TDO functions as shown in Figure D-5.



**Figure D-5 — Scan Topology Training**

The commands in this sequence perform the functions described as follows:

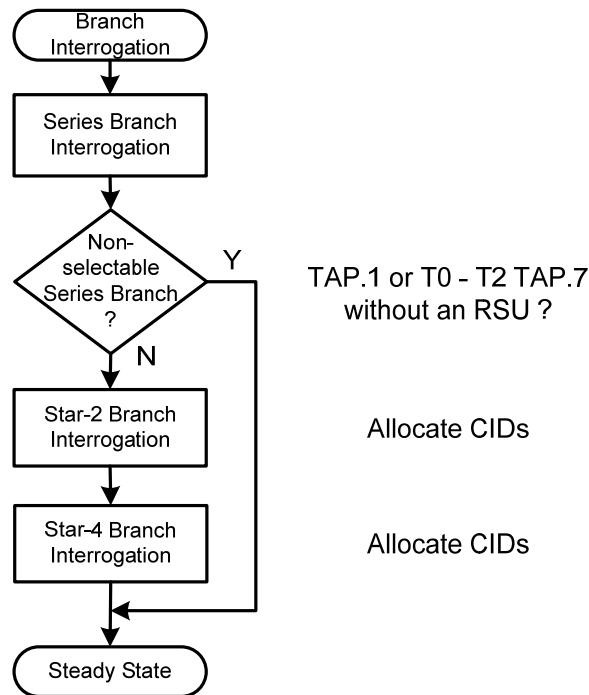
- CMD (STC2, APFC = Standard function) configures the TDIC and TDOC pins of a T4 and above TAP.7 as the T3 TAP.7 pin functions unless inhibited by the Chip-Level Logic
- CMD (STMC, Test for Star-4 Scan Topology) tests for a Star-4 Scan Topology while the DTS drives both TDI(C) and TDO(C) to a logic 0 during the *Update-DR* state of CP2 of the command
- CMD (SCNB, Test for Series Scan Topology) tests for a Series Scan Topology with a CR Scan of all zeros whose length is longer than the longest possible scan path

### D.3 Scan topology interrogation

Start-up is a three step process, as follows:

- Series Branch interrogation
- Star-2 Branch interrogation
- Star-4 Branch interrogation

These steps are shown in Figure D-6. Note that there is no need to interrogate Star-2 and Star-4 Branches when an unselectable Series Branch is discovered (i.e., a branch containing a TAP.1 or T0-T2 TAP.7s without an RSU). This means this type of Series Branch cannot be placed Offline, precluding its use with other branches.



**Figure D-6 — Branch interrogation**

This process determines the following:

- Whether Series, Star-4, and Star-2 Branches exist.
- The characteristics of the TAP.7 Controllers forming each branch
- The chip-level device identification code for each TAP provided it exists

### D.3.1 Series Branch interrogation

Interrogation of the Series Branch determines the:

- Number of TAPs
- The CNFG0–CNFG register values
- The Identification Code values of each TAP

It cannot determine the difference between a TAP.1 and T0 TAP.7. Since TAP.7 Controllers have a Device Identification Register, the presence of this register and its value may be used to identify the TAP type and characteristics of a TAP.7 Controller, provided the appropriate HSDL.7 and BSDL.7 descriptions of chips are available.

#### D.3.1.1 Series characteristics used for interrogation

The following TAP characteristics created by the *Test-Logic-Reset* state are used with Series Branch interrogation:

TAP.1:

- The Instruction Register contain either the BYPASS or the IDCODE instruction
- The Chip-Level Scan Path is either the Bypass bit or the Identification Code Register
- Scan data produced by a DR Scan is a constant length of:
  - One bit for a DR Scan with the BYPASS instruction in the Instruction Register
  - Thirty-two bits for a DR Scan with the IDCODE instruction in the Instruction Register
- Scan data produced by a DR Scan is a fixed value

T0 TAP.7:

- The Instruction Register contains the IDCODE instruction
- The Chip-Level Scan Path is the Identification Code Register
- Scan data produced by a DR Scan is a fixed value
- The length of the Scan Path for a DR Scan with the IDCODE instruction is 32 bits

T1–T5 TAP.7:

- The Instruction Register contains the IDCODE instruction
- The Scan Path is the Identification Code Register
- Scan data produced by a DR Scan is a constant when using the System Path
- Scan data produced by a DR Scan is not a constant when using the Control Path
- The TAP class may be read with the SCNB Command
- The length of the Scan Path for a DR Scan with the IDCODE instruction is 32 bits

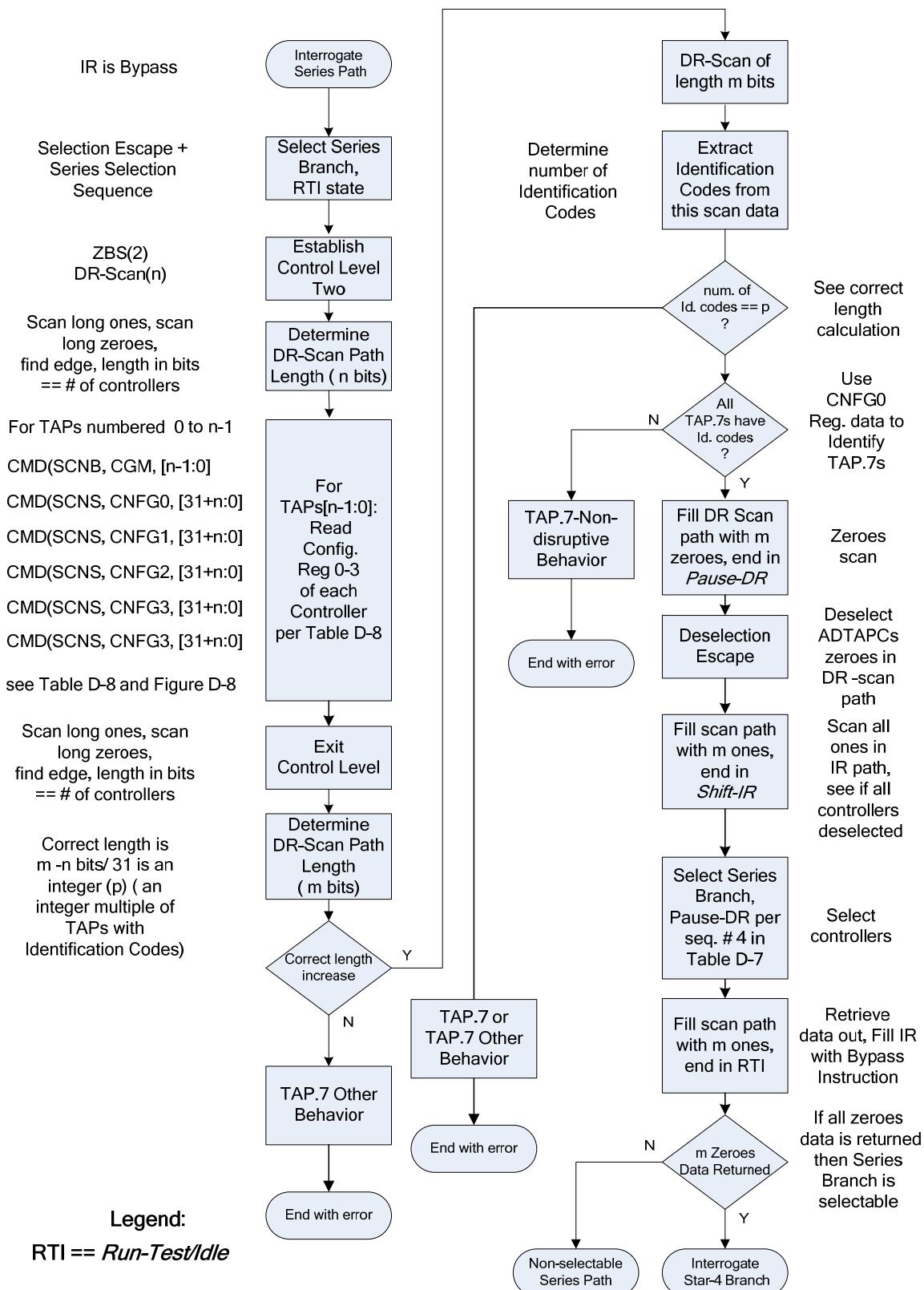
### D.3.1.2 Interrogation process

The interrogation of a Series Branch provides the following information:

- The values of the CNFG0–CNFG3 Registers (when these registers are implemented)
- The Identification Register value for each CLTAPC provided this register is implemented
- Whether the branch is selectable (all TAPs are implemented with an RSU)

This information is derived using the Series Branch interrogation shown in Figure D-7. It can be used to determine the following:

- The number of TAP.1 and T0 TAP.7 Controllers in the branch
- The number of T1, T2, T3, T4, and T5 TAP.7s in the branch
- The features supported with TAP.7s provided the CNFG0–CNFG3 Registers are implemented



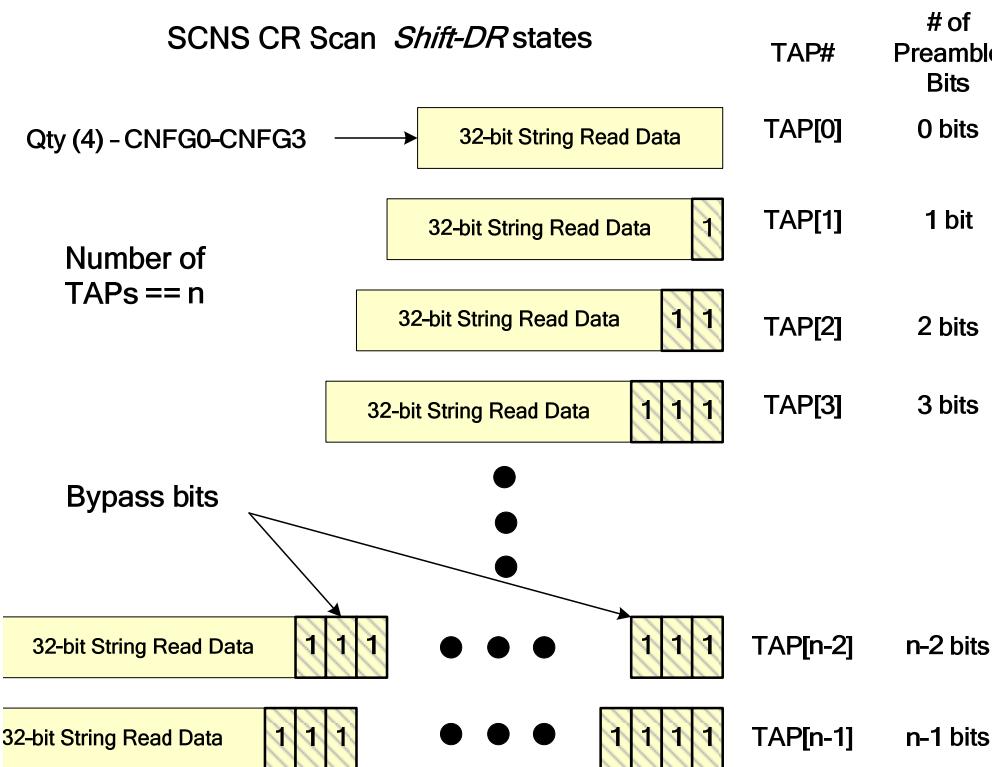
**Figure D-7 — Series Branch interrogation**

### D.3.1.3 Configuration Register reads

The CNFG0–CNFG3 Registers are read with the command sequences shown in Table D-9. The scans these command sequences produce are shown in Figure D-8.

**Table D-9 — Configuration Register reads for TAPs [n:0]**

Command	Length	Data	Preamble preceding String Data
CMD(SCNB, CGM)	n	...0001 left shifted by a number of bits equal to the TAP number	NA
CMD(SCNB,CNFG0)	32 + TAP number	String[31+n: n]:	n bits
CMD(SCNB, CNFG1)	32 + TAP number	String[31+n: n]:	n bits
CMD(SCNB,CNFG2)	32 + TAP number	String[31+n: n]:	n bits
CMD(SCNB,CNFG3)	32 + TAP number	String[31+n: n]:	n bits



**Figure D-8 — Reads of CNFG0–CNFG3 Registers in a Series Scan Topology**

#### D.3.1.4 Determining whether a Series Branch is selectable

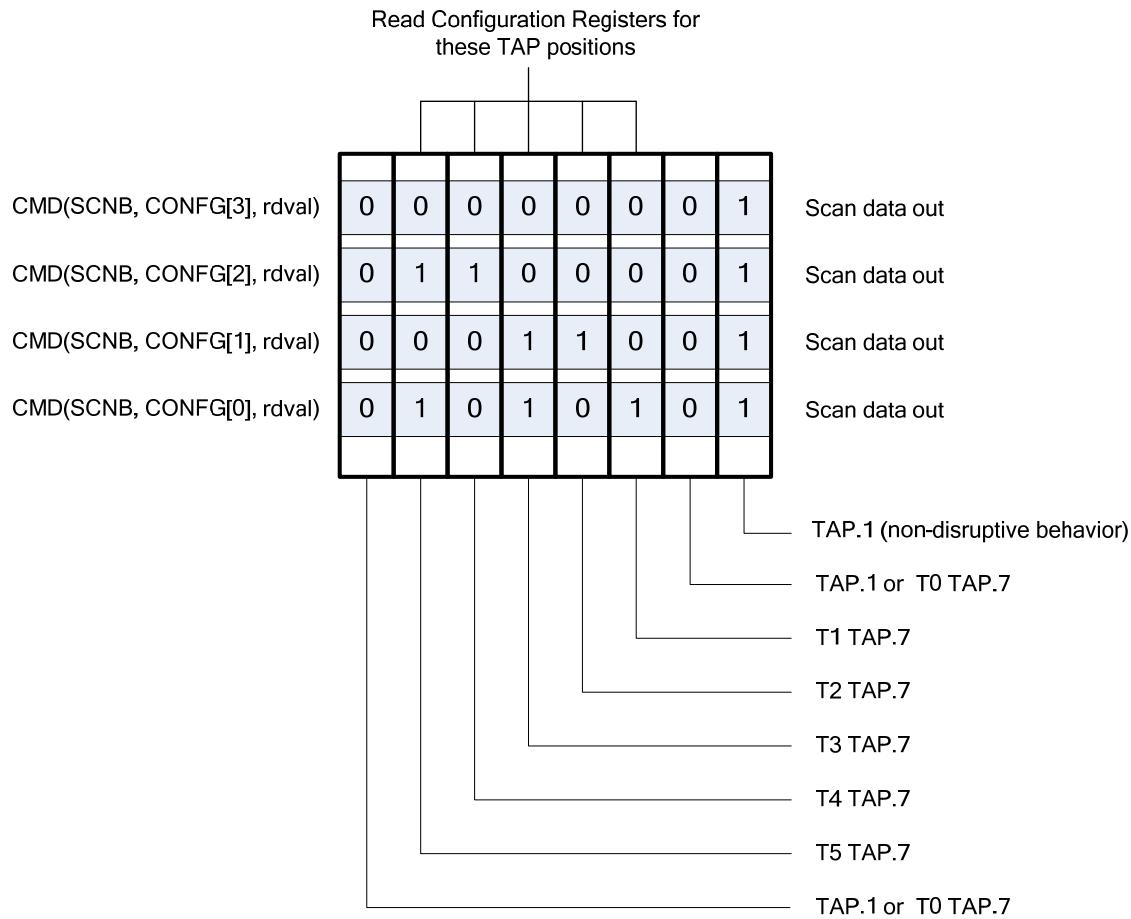
The last part of the Series Branch interrogation shown in Figure D-7 determines whether all TAPs in the series Branch are implemented with an RSU. This is determined by observing the differences in scan path behavior of TAP.7 Controllers that are Online and Offline. With all TAP Controllers Online, a DR Scan of sufficient length is performed to fill the DR Scan Path with zeros. This operation ends in the *Pause-DR* state without traversing the *Update-DR* state. A Deselection Escape is used to deselect all TAP.7 Controllers with an RSU in the *Pause-DR* state. At this point the DTS may switch to an alternate means of TMS generation while its TAPC state is the *Pause-DR* state. The TDI value remains a logic 1 during this period while this alternate method of TMS generation is used. One example of an alternate TMS generation utilizes TDI data as the TMS value in the *Shift-DR* state while forcing a TMS value of logic 0 in other states. An all-ones IR Scan is then initiated using this TMS generation method.

While this IR Scan proceeds, a second sequence of TMS values following a Selection Escape Sequence is initiated. This sequence is equivalent to that shown as sequence number four of Figure D-7. This causes the Selection of the TAP.7 Controllers and the TAPC state to move to *Pause-IR* in TAPs that do not have an RSU. Following this sequence, the TAPC state of previously Online TAP.1s and TAP.7s is *Pause-IR*, while the TAPC state of previously Offline TAP.1s and TAP.7s is *Pause-DR*. At this point, with the DTS TAPC state *Pause-DR*, the scan may be continued using the normal TMS values.

The Scan Path make-up is the DR path for previously Offline TAP.7s and the IR path for previously Online TAP.7s. A long scan of an all-zeros pattern is used to flush the DR and IR path values of the TAPCs ending in the *Pause-xR* state for further analysis. This is followed by a long scan of an all-ones pattern to place a Bypass Instruction in the IRs of all previously Online TAPCs. If the flushed scan data is returned as an all-zeros pattern, all TAPCs are TAP.7s and are implemented with an RSU. The Series Branch is therefore selectable. The Series Branch is not selectable if the flushed data is not an all-zeros pattern.

#### D.3.1.5 Quick determination of TAP types in a Series Branch

When all TAP.7 Controllers are Online, a quick way of ascertaining the classes of TAPs is determined using four SCNB Commands as shown in Figure D-8. An STMC Command setting the CGM Register to a logic 1 precedes these SCNB Commands. This method of ascertaining the classes of TAPs is not utilized in the interrogation of a series branch shown in Figure D-7.



**Figure D-9 — Determining the type/class of series TAPs**

### D.3.2 Star-4 Branch interrogation

#### D.3.2.1 Information provided

The interrogation of a Star-4 Branch provides the following services:

- Determines the number of T3, T4, and T5 TAP.7s in the branch
- Allocates the T3, T4, and T5 TAP.7s a CID
- Reads Configuration Register values of each TAP.7 Controller associated with the branch
- Reads the Identification Code Register value when it exists

#### D.3.2.2 Interrogation process

The Star-4 Branch information listed above is derived for each TAP.7 Controller sharing the branch using the Star-4 Branch interrogation process has two parts as shown in Figure D-10 and Figure D-11. It allocates CIDs to TAP.7 Controllers with the undirected method beginning with CID one until CID zero is reached or the number of CIDs allocated is fifteen or less. It continues to allocate CID zero to all subsequent TAP.7 Controllers with a CID of fifteen until an arbitrary limit is reached.

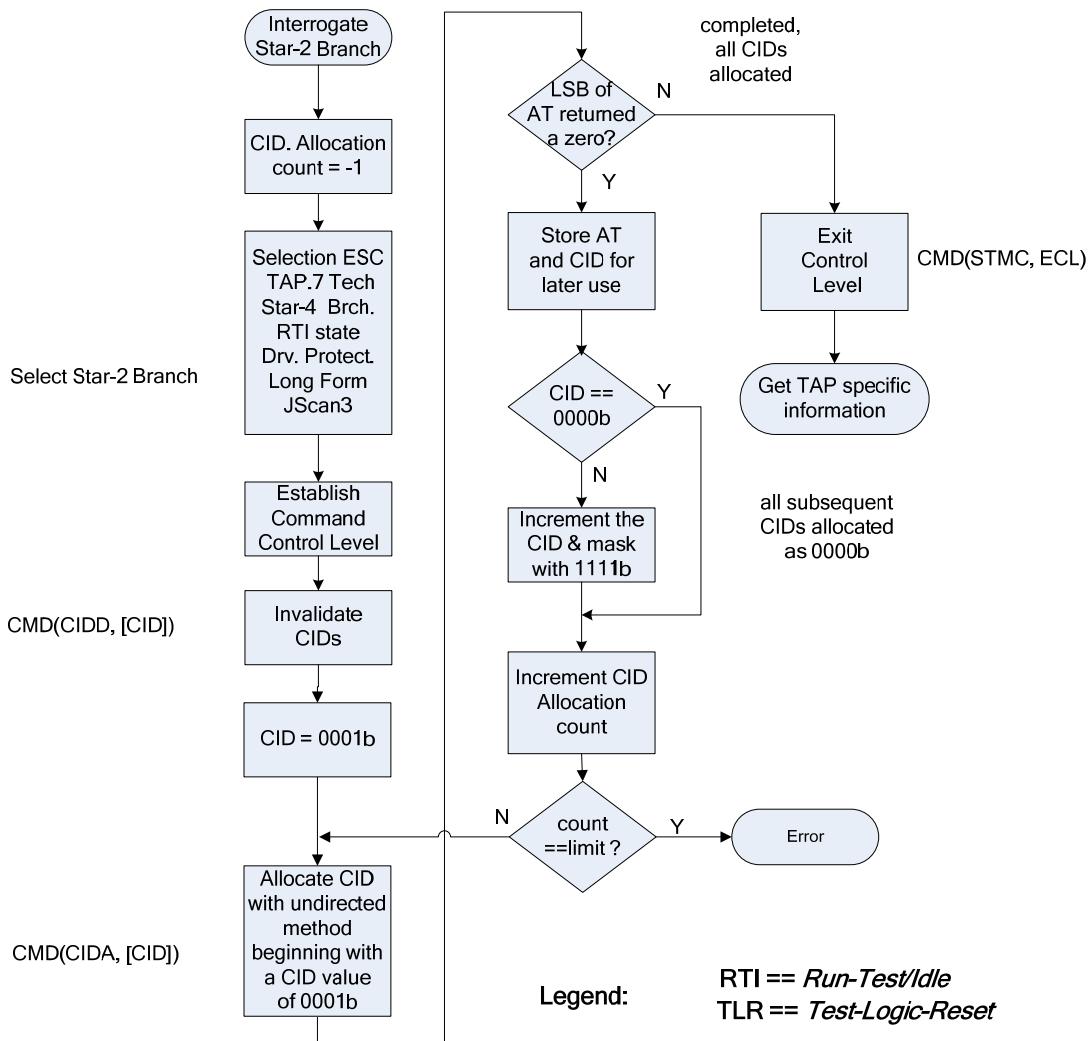
Part One of the Star-2 Branch interrogation process provides the following services:

- Establishes the maximum number of CID allocations permitted without declaration of an error.
- Selects the Star-2 Branch with a Resynchronization Long-Form Selection Sequence.
- Optionally sets the TMSC Sampling to Rising Edge—this should not be necessary unless an implementation has a TMSC hold time problem.
- Invalidates all CIDs in the Branch in preparation for CID allocation.
- Allocates CIDs in a loop:
  - Until the AT returned has an LSB of one indicating all TAPs in the branch have been allocated a CID.
  - With the first 15 CIDs allocated as 1 through 15 and 16 onward allocated CID zero.
- Passes the number of TAPs in the branch and passes this number to Part Two.

Part Two of the Star-2 Branch interrogation process provides the following services of each TAP.7 Controller discovered by Part One.:

- Reads all three Configuration Register values
- Reads the CLTAPC's Identification code

This information is derived using the Star-4 Branch interrogation process shown in Figure D-10 and Figure D-11.



**Figure D-10 — Star-4 Branch interrogation—part one**

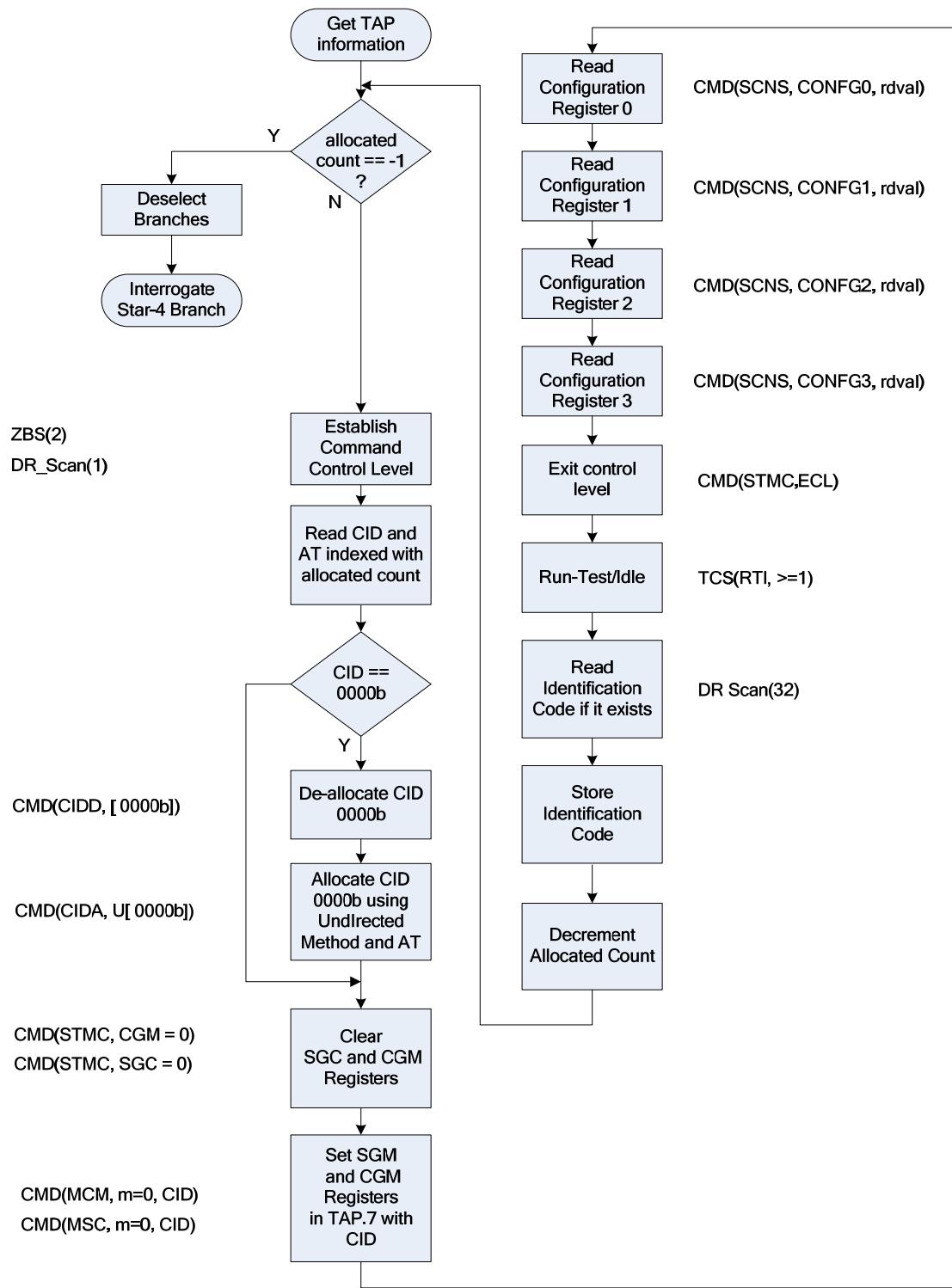


Figure D-11 — Star-4 Branch interrogation—part two

### D.3.3 Star-2 Branch interrogation

#### D.3.3.1 Information provided

The interrogation of a Star-2 Branch provides the following services:

- Determines the number of T4, and T5 TAP.7s in the branch
- Allocates the T4 and T5 TAP.7s a CID
- Reads Configuration Register values of each TAP.7 Controller
- Reads the Identification Code Register value when it exists

#### D.3.3.2 Interrogation process

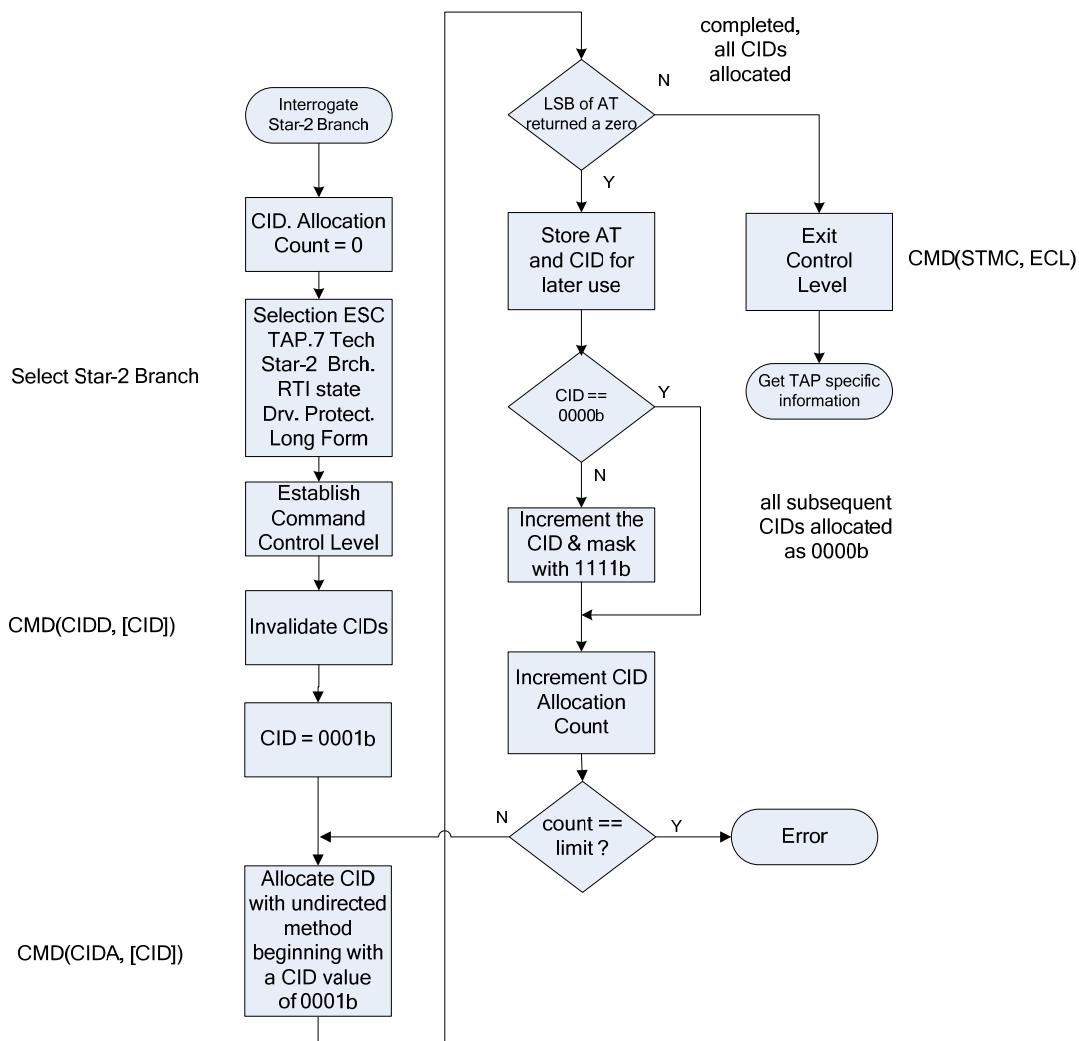
The Star-2 Branch information listed in the previous subclause is derived for each TAP.7 Controller sharing the branch using the Star-2 Branch interrogation process has two parts as shown in Figure D-12 and Figure D-13. It allocates CIDs to TAP.7 Controllers with the undirected method beginning with CID one until CID zero is reached or the number of CIDs allocated is 15 or less. It continues to allocate CID zero to all subsequent TAP.7 Controllers with a CID of 15 until an arbitrary limit is reached.

Part One of the Star-2 Branch interrogation process provides the following services:

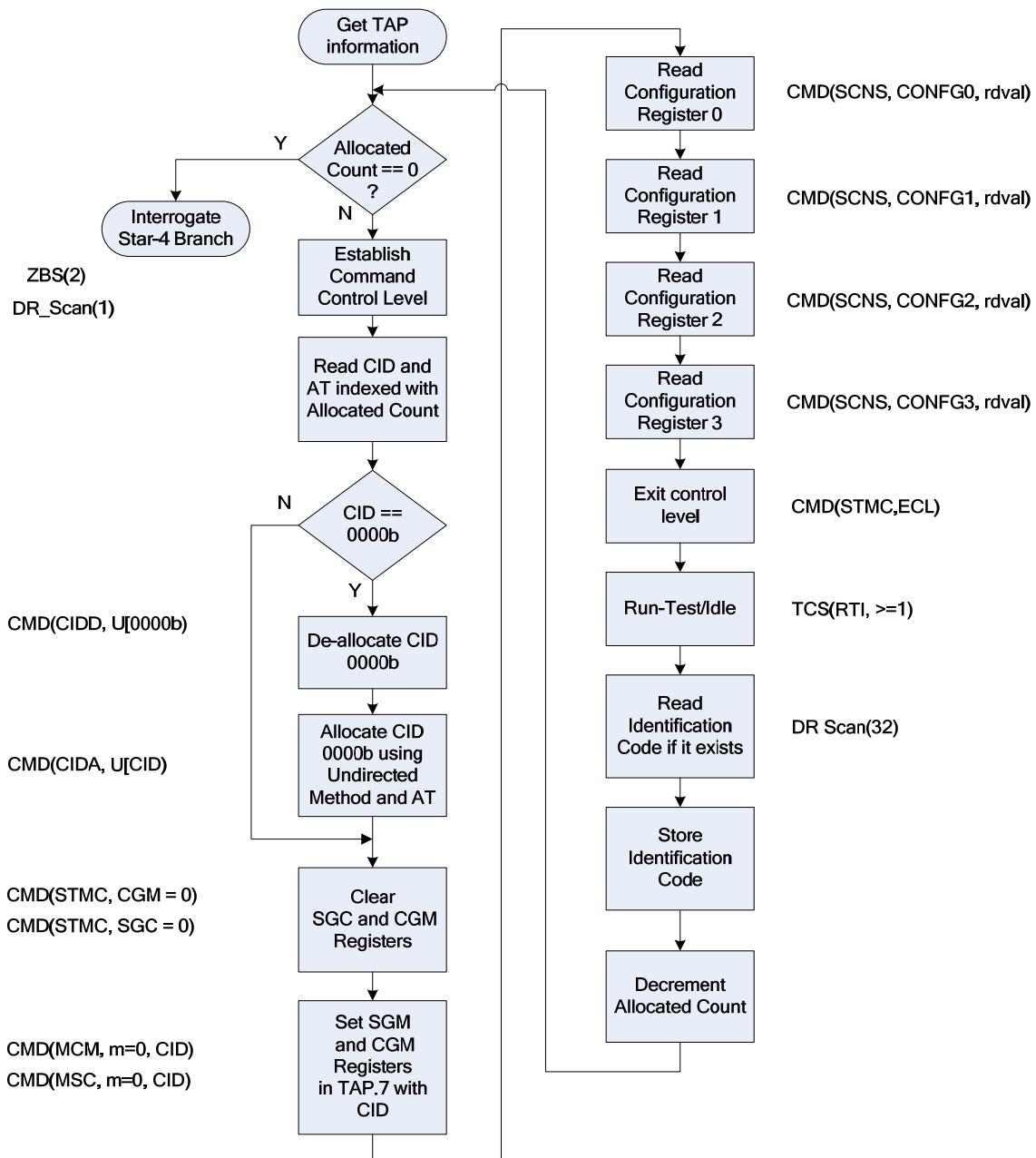
- Establishes the maximum number of CID allocations permitted without declaration of an error.
- Selects the Star-2 Branch with a Re-synchronization Long Form Selection Sequence.
- Optionally sets the TMSC Sampling to Rising Edge – this should not be necessary unless an implementation has a TMSC hold time problem.
- Invalidates all CIDs in the Branch in preparation for CID allocation.
- Allocates CIDs in a loop:
  - Until the AT returned has an LSB of one indicating all TAPs in the branch have been allocated a CID.
  - With the first 15 CIDs allocated as 1 through 15 and 16 onward allocated CID zero.
- Passes the number of TAPs in the branch and passes this number to Part Two

Part Two of the Star-2 Branch interrogation process provides the following services of each TAP.7 Controller discovered by Part One.:

- Reads all three Configuration Register values.
- Reads the Identification Code Register value associated with the CLTAPC.



**Figure D-12 — Star-2 Branch interrogation—part one**



**Figure D-13 — Star-2 Branch interrogation—part two**

## D.4 Establishing TAP.7 operating conditions

Once the start-up has been completed the TAP.7 operating conditions are established. These conditions include the following:

- Setting the sampling edge for TMSC
- Set the power-down mode
- Set the scan format

## D.5 CID management

The CID value of zero should be reserved for special purposes, some of which involve Scan Topology Training. Following initialization all TAP.7 Controllers have a CID of zero. At other times, TAP.7 Controllers that have been placed Online after being Offline-at-Start-up have a CID of zero. A CID of zero may also be allocated to a TAP.7 Controller.

After connection of a DTS and TS and if it is known that the system is to be operated with a scan format other than JScan0–JScan2, the DTS software assumes one of the following:

- A single TAP.7 Controller is connected to the DTS
- Multiple TAP.7 Controllers are connected to the DTS and their TCAs are known
- Multiple TAP.7 Controllers are connected to the DTS and their TCAs are unknown

When a single TAP.7 Controller is connected to the DTS, no action is needed as the TAP.7 Controller already has a valid CID of 0x0.

When the number of TAP.7 Controllers connected in a Star Scan Topology is unknown, the DTS software invalidates CIDs. It then allocates a CID to each controller that is ready to have a CID assigned to it. The process used to allocate a DTS-generated CID to a TAP.7 Controller is basically the same for Star-4 and Star-2 Scan Topologies. The following two different approaches may be used to assign CIDs:

- One-pass approach:
  - Assign the CIDs using the undirected method
- Two-pass approach:
  - Discover the TCAs:
    - Assign the same CID to each winner of the allocation arbitration while reading the TCA of the arbitration winner. Any number of TCA values may be read using this process
    - When the DTS reads an all-ones (1s) value, all TAP.7s with CIDs having a valid NODE\_ID have been located
  - Assign the final CIDs:
    - Invalidate the CID assigned to the TAP.7s
    - Assign CIDs with the DTS supplying the TCA. A maximum of sixteen CIDs may be assigned

When multiple TAP.7 Controllers are connected to the DTS and their TCAs are known, the DTS software may skip the “Discover the TCAs step” shown above.

Note the one-pass approach is the fastest but may not produce an optimal assignment for debug or test speed. TAP.7s with higher than expected usage rates may be assigned CIDs that are lower numbers, making their management faster.

Any CID already assigned may be de-allocated and allocated to another controller using the direct assignment method shown below:

- |                      |  |
|----------------------|--|
| 1) ZBS               | Create <i>ACTIVE</i> state with the Command Control Level            |
| 2) ZBS               | Lock control level with scan with at least one <i>Shift-xR</i> state |
| 3) DR Scan           | Deallocate the specified CID   |
| 4) CMD(CIDD, D[CID]) | Specify CID to be assigned   |
| 5) CMD(CIDA, [CID])  | Exit the Command Control Level                                       |
| 6) CMD(STMC, ECL)    |  |

## D.6 Managing simultaneous debug actions

It is sometimes desirable to initiate debug actions (run and halt in multiple devices application execution) simultaneously across multiple chips. With a Series Scan Topology, this operation is accomplished by broadcasting information specifying an action and initiating the action with the *sys\_tck* signal while in the *Run-Test/Idle* state (or another state such as *Update-xR*). The Star Scan Topology does not possess this broadcast attribute of a Series Scan Topology. A similar attribute may be constructed for a Star Scan Topology using a Series-Equivalent Scan in a Star-2 or Star-4 Scan Topology or with a Series-Equivalent Scan with a mix of TAP.7 Technology Branches.

## D.7 Operations to avoid with a Star-4 Scan Topology

### D.7.1 The use of the JScan0–JScan2 Scan Formats

The DTS application should avoid the use of the JScan0, JScan1, and JScan2 Scan Formats with a Star-4 Scan Topology other than with the start-up sequence. This is important as a TAP.7 Controller reset causes the use of either the JScan0 or JScan1 Scan Format in cases where the TAP.7 Controller is not Offline-at-Start-up.

The DTS application obeys certain constraints when these scan formats are used with a Star-4 Scan Topology if it uses these scan formats with the Star-4 Scan Topology. Only the limited use of scan without drive conflicts is possible. Starting from the *Test-Logic-Reset* state and moving immediately to the Command Control Level, the CR Scan associated with the SCNB Command may be used without drive conflicts in a Star-4 Scan Topology. This is possible since the capture data for write-only registers is a logic 0 and the DTS controls the subsequent TDO data. When every *Shift-DR* state is followed by an *ExitI-DR* state, every TAP.7 Controller in the Star Scan Topology will supply the same TDOC data. Any other use of these scan formats is likely to create TDOC drive conflicts. This presumes the TOPOL Register value permits the drive of the TDOC signal.

### D.7.2 The selection of CLTAPCs of TAP.7 Controllers allocated the same CID

After a TAP.7 Controller reset, all T3 and above TAP.7 Controllers have a valid CID of zero. Allocation of the same CID to multiple TAP.7 Controllers is also possible. This is not uncommon as this is required to discover the TCAs of more than 16 TAP.7 Controllers with undirected CIDs. The CLTAPCs of TAP.7 Controllers with the same CID remains deselected to avoid drive conflicts in a Star Scan Topologies.

## D.8 Using a T5 TAP.7

### D.8.1 Setup and use of the Transport Function

The Data Transport Function is managed primarily with Transport Directives. The transport protocol revision and TAPC states supporting transport are managed with scan commands. The remainder of the transport functionality is managed with Transport Directives.

### D.8.2 Sharing the use of a Logical Data Channel

The DTS software is responsible for allocating the same Physical Data Channel address to Physical Data Channels with compatible functions when more than one Physical Data Channel is allocated to a Physical Data Channel Address.

### D.8.3 Transferring data with Transport Packet Data Payloads

T5 TAP.7 Controller Physical Data Channels move information between the DTS and TS without regard to its information content. Generally, a protocol is embedded in the data transferred to utilize the capability afforded by the T5 TAP.7. Protocols using the T5 TAP.7 do not affect the operation of the TAP.7 Controller. Each Data Channel Unit connected to a TAP.7 Controller may use a different protocol.

Some general guidelines are as follows:

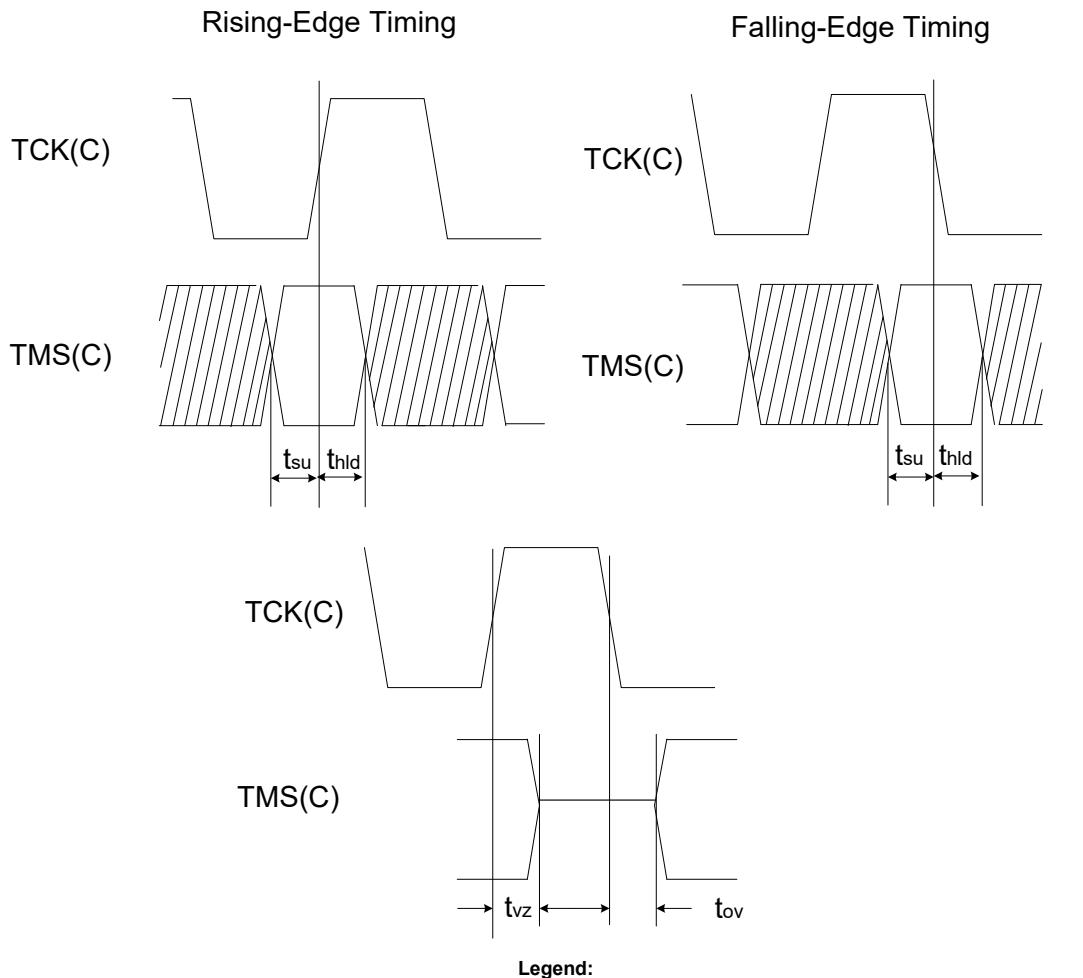
- The transport payloads should be treated as merely a mechanism to move bits from the DTS to the TS and vice versa.
- Transport Directives are available to park the TMSC pin at either a logic 1 or logic 0 allowing start bits with a value opposite of the parked value indicating the beginning of relevant information. This is important as with some custom protocols the TMSC pin will not be driven at times during the data payload of a Transport Packet.
- A Protocol Packet may not match the length of a Transport Packet Data Payload. Therefore a particular Transport Packet should not be assumed to have any timing relationship to the protocol using it because a protocol message may:
  - Span multiple Transport Packets.
  - Begin at any bit position of a Transport Packet Data Payload.
  - End at any bit position of a Transport Packet Data Payload.

## Annex E

(informative)

### Recommended electrical characteristics

The TAP.7 Controller supports both rising- and falling-edge timing for the use of the Advanced Protocol and rising-edge timing for the Standard Protocol. The setup and hold times for signaling with these protocols is shown in Figure E-1.



**Figure E-1 — Electrical characteristics**

Keepers should switch state the same as the input signal no later than a time that is 20% of the TCK(C) period at the maximum operating frequency.

## Annex F

(informative)

### Connectivity/electrical recommendations

#### F.1 Overview

This annex provides recommendations for connecting the DTS and one or more TAP.1s and/or TAP.7s (the link). These recommendations are not intended to replace proper design analysis or implementation. Failure to manage edge rates, terminations, reflections, and noise at chip inputs will impact performance and may result in an inoperable system. Proper analysis, design, and simulation are the key to producing reliable operation at any operating frequency and are essential to maximize the bandwidth provided by this connection. The failure to follow these recommendations may result in an inoperable system.

A summary of the recommendations made for Chips, Boards, and the DTS is shown in F.13. Subclauses F.2 through F.12 provides background material supporting these recommendations.

#### F.1.1 General information

A combination of proper DTS, board, and chip design is required to provide both reliable and high-performance operation. This annex contains a description of practices in each of these areas that contribute to proper system operation. Consideration is given to all aspects of physical connectivity between the DTS and the TS. These aspects include but are not limited to the following:

- DTS drive and interface characteristics
- Cabling between the DTS and the TS
- Various connectors that exist along the path between the DTS and the target TAP(s)
- The TS printed circuit board connection topology
- Chip TAP drive and interface characteristics

The above elements determine the characteristics of the signaling paths connecting the DTS and TAPs. It is important to understand that the signal quality and integrity at any signal toggle rate is adversely affected when the DTS, board, and chip design characteristics are not managed properly. This annex provides guidelines for managing these elements while providing information about four basic connectivity scenarios and three basic transmission line termination techniques.

#### F.1.2 Factors affecting reliable operation

The factors affecting reliable operation at any operating frequency and high-performance operation at TCK(C) frequencies (30 MHz to 100 MHz) are related. Reflections and nonmonotonic edges on edge-sensitive signals like TCK(C) will affect the reliability of both IEEE 1149.1 and IEEE 1149.7 operation with a system at all frequencies. Reflections and nonmonotonic edges on level-sensitive signals will also impact performance. The system designer is responsible for using a viable connection scheme, managing transmission line impedances, controlling reflections, limiting transmission line stub lengths, and implementing a good termination scheme. In addition to these factors, the following four additional measures can help ensure reliable signaling and operation:

- Capacitive load isolation

- SOC TAP input conditioning
- Signal rise and fall times
- The use of rising- and falling-edge TMSC signal sampling

These factors are listed in Table F-1 along with the areas requiring their consideration.

**Table F-1 — Factors affecting system performance and reliability**

Factor	DTS	Board	Chip
<b>Transmission line impedances</b>	<b>Yes</b>	<b>Yes</b>	
<b>Connection scheme of TAPs/ # connection points</b>	<b>Yes</b>	<b>Yes</b>	
<b>Termination schemes</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>
<b>Capacitive load isolation</b>	<b>Yes</b>	<b>Yes</b>	
<b>Signal rise and fall times</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>
<b>Source impedance</b>	<b>Yes</b>		<b>Yes</b>
<b>SOC signal conditioning</b>			<b>Yes</b>
<b>Route length and stub length</b>		<b>Yes</b>	

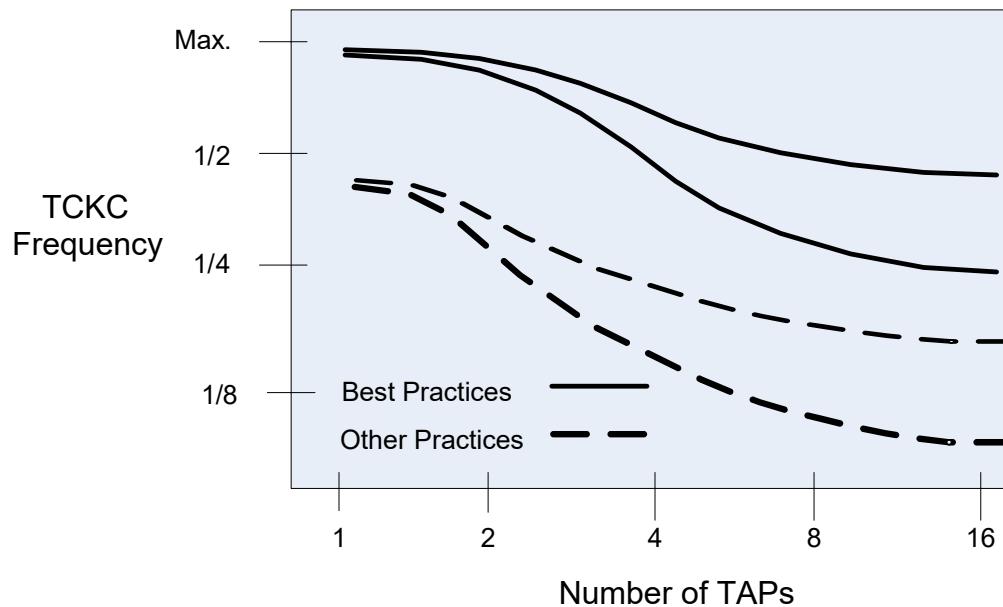
Maximizing the reliability and performance of the DTS/TS connection requires each of these factors be addressed by the party responsible for them.

### F.1.3 Factors affecting link performance

Certain factors affect the link operating frequency. These include the following:

- Factors listed in Table F-1
- The TAP electrical characteristics detailed in Annex E
- The use of rising- and falling-edge TMSC signal sampling

Typically, the maximum TCK(C) frequency drops as the number of connections of TAPs disturbing the transmission line impedance increases. This is illustrated in Figure F-1. This frequency is affected by a combination addressing the factors in Table F-1 in addition to other system considerations not described in this annex.



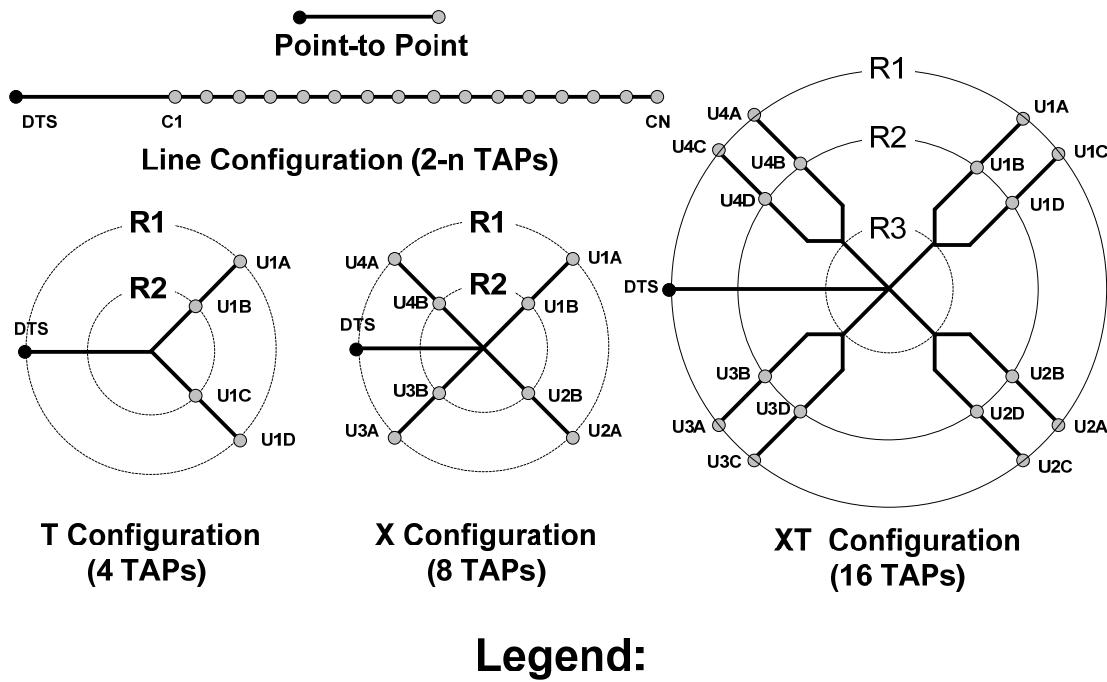
**Figure F-1 — Typical TCK(C) frequency versus number of TAPs**

The strategies for managing the transmission line effects change as either the number of TAPs or combination of TAPs presenting a capacitive load (TAP loads) increases or may be governed by DTS or TS characteristics. Individual results may deviate from the profiles shown in Figure F-1, depending on the connection topology utilized and other factors. Board designers are encouraged to both verify the functionality of their system and optimize their system characteristics by simulating its operation with tools such as Spice and more sophisticated tools.

## F.2 Physical connection topologies considered

The following five connection topologies considered in this annex are shown in Figure F-2:

- A Point-to-Point configuration
- A Line configuration—2 to n TAP loads connected to a single transmission line with no branches
- A T configuration—a single transmission line that splits into two branches
- An X configuration—a transmission line that splits into four branches
- An XT configuration—a transmission line that splits into four branches with each branch further splitting into two branches



### Legend:

- **DTS**      **R1** = Distance from common node to last TAP or DTS
- **TAP**      **R2** = Distance from common node to first TAP

**Figure F-2 — Connection topologies**

The choice of a connection topology is determined by the number of TAPs, the DTS capability, and the desired operating frequency. The Line configuration has been traditionally used with IEEE 1149.1 system. For edge-sensitive signals like TCK(C), use of the line configuration requires monotonic rising and falling edges and a noise margin sufficient to be unaffected by all reflections at all loads. This requirement can be met through controlling edge rates, drive levels, stub lengths, reflections, input buffer design techniques, and proper transmission line termination. Historically, in many designs, electrical issues related to this topology have not been properly managed. This has resulted in decreased operational frequency or loss of basic operational stability. Simulation of a Line configuration should be considered to ensure basic operation and is essential to maximize performance.

The T, X, and XT configurations are similar to the approaches used with DDR-2 technology. These approaches are used to improve the signal quality at the loads and raise the operating frequency. They, however, require that connection symmetry be emphasized. The importance of this cannot be overemphasized. The use of these configurations without adequate attention to symmetrical routing and loading will most likely produce a non-functional system. Simulation of these configurations should be considered mandatory to understand their behavior, ensure satisfactory operation, and maximize performance.

These connection topologies (Point-to-Point through XT) are described in subclauses F.8 through F.12, respectively.

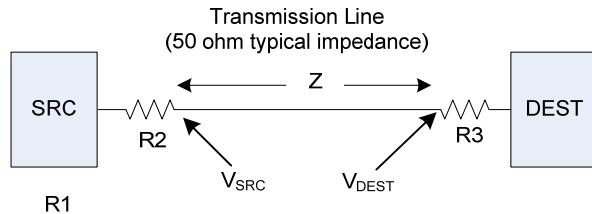
### F.3 Termination schemes considered

Various industry standard transmission-line termination schemes exist to terminate transmission lines to minimize reflections. The two basic termination techniques considered by this annex are as follows:

- Series source termination
- End-of-line parallel termination (variations include both DC- and AC-coupled termination)

These termination schemes are shown in Figure F-3 and Figure F-4 respectively.

## Series Termination

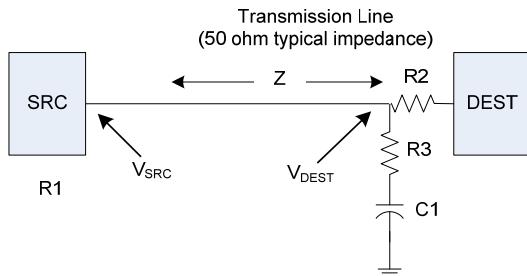


SRC, DEST == Signal source and destination  
R1 == Source internal impedance (10-30 ohms)  
R2 == Series termination resistor, external to SRC  
R3 == Capacitance Isolation resistance (10 – 27 ohms)  
Z == Transmission line (50 ohm impedance typical)  
DEST presents a capacitive load, typically 3-7 pf or more

**Figure F-3 — Series termination scheme**

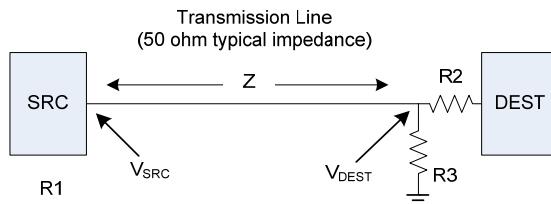
Note that the destination resistor's (R3s) effects decrease as the load capacitance decreases, and they may not be needed for single load topologies, where the load capacitance is small, or where the connection between the SRC and DEST is sufficiently short as to not qualify as a transmission line (see F.5.1). The simulation results can quantify the effects of this resistor.

## Parallel AC Termination



SRC, DEST == Signal source and destination  
R1 == Source internal impedance (10-30 ohms)  
R2 == Isolation resistance (10 – 27 ohms)  
R3 & C1 == Termination impedance (tuned for a specific freq.)  
Z == Transmission line (50 ohm impedance typical)  
DEST presents a capacitive load, typically 3-7 pf or more

## Parallel DC Termination



SRC, DEST == Signal source and destination  
R1 == Source internal impedance (10-30 ohms)  
R2 == Isolation resistance (10 – 27 ohms)  
R3 == Parallel termination resistor  
Z == Transmission line (50 ohm impedance typical)  
DEST presents a capacitive load, typically 3-7 pf or more

**Figure F-4 — Parallel termination schemes**

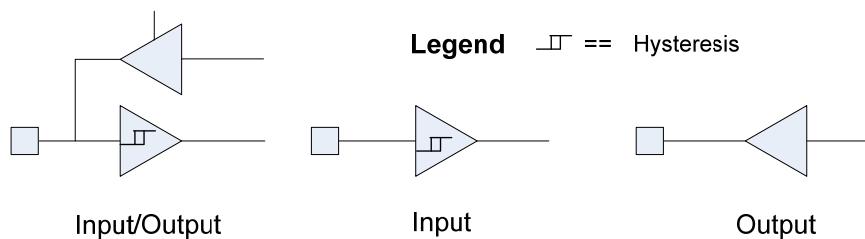
Note that the destination resistor's (R2s) effects decrease as the load capacitance decreases, and they may not be needed for single load topologies, or where the load capacitance is small. The simulation results can quantify the effects of this resistor. Isolation resistors are optional and may add unnecessary cost and space to the end product. The designer should evaluate and select the best topologies and options based on product design requirements.

## F.4 Chip considerations

### F.4.1 SOC input conditioning

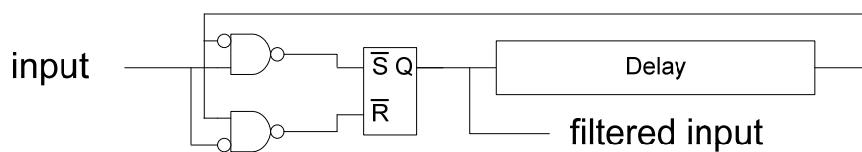
In connectivity scenarios with several TAPs, it usually is not possible to achieve the perfect termination design. In such cases, TAP input signal quality may approach a level of concern. In these scenarios, an effective countermeasure is to design tolerance for less-than-perfect signal quality into the input logic of the SOC(s). Two techniques are described here—the use of hysteresis and the use of one-shot glitch filtering techniques to limit an input's ability to respond to noise on a signal's transition or reflections as edges.

The use of hysteresis on device inputs has been used since the early days of logic, even as far back as the days of vacuum tube circuits. Commonly referred to as Schmitt trigger inputs, these devices use an internal, biased feedback technique to modify the logic-switching point, based on the current input level. A non-Schmitt input may switch at approximately the 50% level for both the rising edge and falling edge of an input signal. With a logic 0 input, a Schmitt trigger input requires a rising-edge signal to reach a higher level, such as 60–70% of the logic level, before it will declare a logic 1. Once a logic 1 is detected, it requires a falling edge to reach a lower level, such as 30–40% of the logic level before it detects a logic 0. The difference in the switching thresholds represents the signal voltage change (referenced to the previous transition point) needed to cause the detection of the opposite transition. Hysteresis provides a degree of input noise immunity and some rejection of signal transitions created by reflections by separating the logic 0 and logic 1 switching thresholds. It also provides immunity to the half levels and signal variations created by the use of series transmission-line termination. With hysteresis, once the switching threshold has been crossed, a significant voltage swing in the opposite direction is needed to produce failure scenarios such as double-clocking of the TCK(S) signal or create erroneous Escape Detection Sequences with the TMS(C) signal. The use of inputs with hysteresis (Schmitt trigger inputs) is defined in Clause 12 and shown in Figure F-5.



**Figure F-5 — The use of input buffers with hysteresis**

The second technique shown in Figure F-6 uses the concept of one-shot circuits (also referred to as monostable multivibrators, glitch filters, or glitch gobblers also date back to the days of vacuum tube circuits). When a transition on the input is detected, the one-shot logic blocks the detection of a second transition for a period of time—ranging from at least the maximum supported transition time from  $V_{IL\max}$  to  $V_{IH\min}$ , to at most  $\frac{1}{2}$  the minimum operating period. Unlike hysteresis, the one-shot can filter high-frequency transitions riding on a signal that may leak through hysteresis. Combining hysteresis and a one-shot filter on inputs of edge-critical signals like TCK(C) can significantly reduce a TAP's sensitivity to signal quality.



**Figure F-6 — Signal filtering using a one shot**

Note that both the use of inputs with hysteresis and signal filtering using one shots are design practices that add to a TS' tolerance of signal quality issues. Chip designs should utilize both of these practices as expecting perfect signal quality at all times in all systems in which a chip may be deployed is unrealistic. Simulations of the combined hysteresis and one-shot glitch filter for noise immunity should be considered mandatory. Board designs should not rely on these chip characteristics for proper system operation. Following the board design guidelines and choosing a DTS with the proper drive, edge-rate control, and termination characteristics should be the first priority. As emphasized previously, the combination of proper DTS, board, and chip design are all required to support high-performance and reliable operation of a multi-chip system.

#### F.4.2 Signal driver rise and fall times

In this standard, there are two possible signal sources—the DTS and the TAPC. The DTS drives the TCK(C), TDI(C), and TMS(C). The TAPC drives the TDO(C) signal back to the DTS, and in the cases of the Advanced Protocol, it also drives the TMS(C) signal. The characteristics of the output drivers in each of these signal sources impact signal integrity by virtue of its inherent internal output impedance and the rise/fall times of the signals that it is driving.

A method of reducing the effects of load-induced impedance discontinuities on signal integrity is simply to slow down the edge rate. Slowing down the edge rate decreases the instantaneous energy impacting and reflecting off impedance discontinuities. As a result, a signal with slower edge rates is less affected by impedance discontinuities with the resulting signal having less ringing.

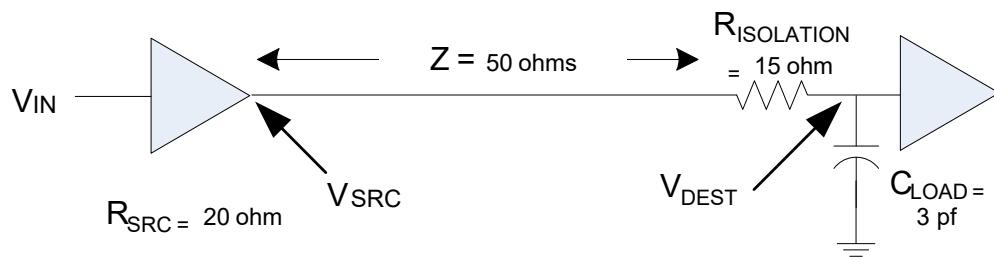
An SOC designer targets a certain performance level for an output driver, based on the performance goals (e.g., –100 MHz), the expected AC and DC loading, and operating conditions (e.g., –20 °C to 85 °C). Relative to signal integrity, a critical parameter for the output driver is its rise and fall timing.

The effects of rise/fall timing are revealed by a simple simulation model with the following characteristics:

##### Simulation Parameters

— Termination scheme	Series
— Driver source impedance	20 Ω
— PCB Trace	~50 mm (~2 in), 50 Ω
— Capacitive load	3 pF, with a 15 Ω isolation resistor
— End-of-line termination	None

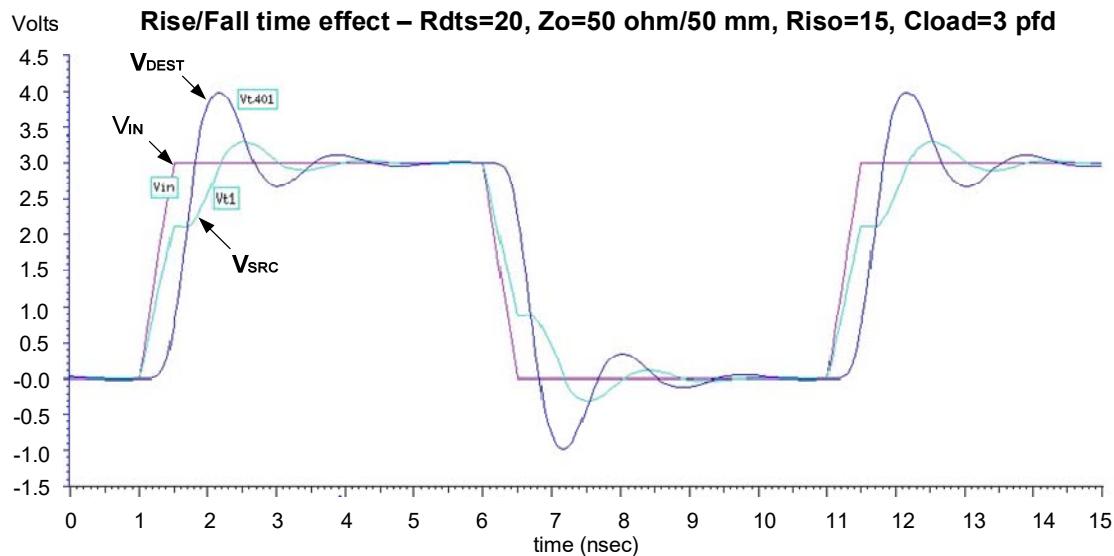
With this model, the total driving impedance (20 Ω that is predominately resistive) is intentionally mismatched with the transmission-line impedance (50 Ω) as shown in Figure F-7. Figure F-8 illustrates the effects of 0.5 ns rise/fall timing. Figure F-9 illustrates the effects of 1.5 ns rise/fall timing. Overshoot is evident in both figures but is much more pronounced with the faster rise time.



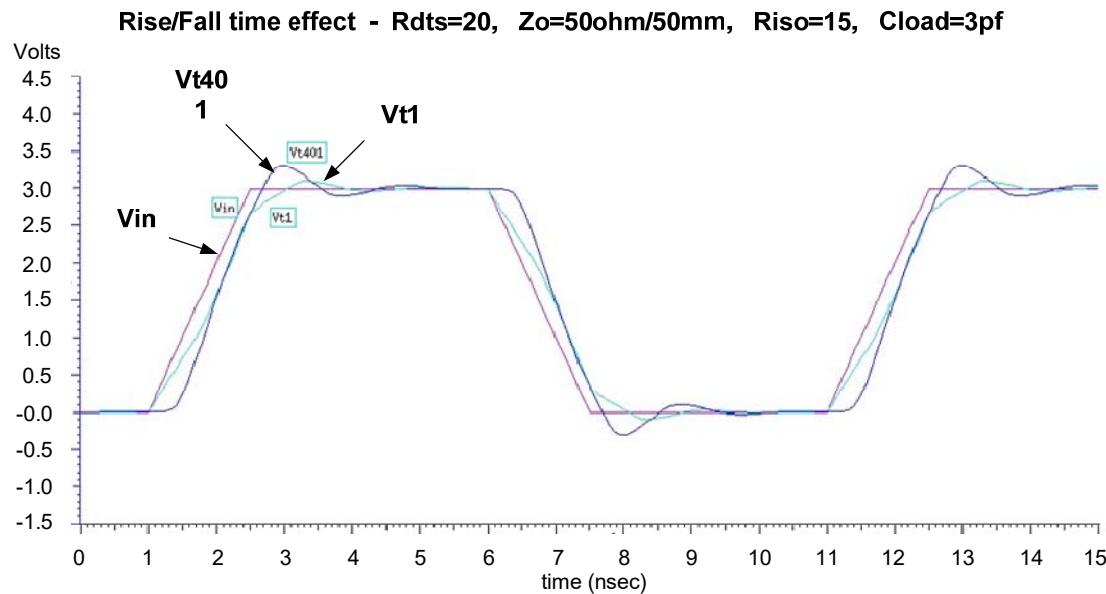
**Figure F-7 — Model of circuit used to compare the rise/fall time effects on signal quality**

The simulation results demonstrate that slowing signal rise/fall times, while keeping all other simulation parameters the same, results in better signal quality. In this simulation, a 0.5 ns rise/fall time produced overshoot of approximately 1 V while a 1.5 ns rise/fall time produced overshoot of approximately 0.3 V.

As the connectivity scenarios become more complex, achieving acceptable signal quality may require more than a single, simple termination scheme. In the more extreme cases, it may take multiple countermeasures to achieve the required results, including slowing the signal's rise/fall times, choosing the right connection topology, and derating of the performance (frequency of operation) as shown in Figure F-1.



**Figure F-8 — Overshoot and undershoot with the output driver's rise/fall time at 0.5 ns**



**Figure F-9 — Overshoot and undershoot with the output driver's rise/fall time at 1.5 ns**

#### F.4.3 Clock-to-output delays

The performance of the TAPC may be adversely affected by excessive TCK(C) to TMS(C) and TDO(C) delays. The clock-to-output timings described in Annex E should be met. It should be noted that a DTS can be designed to compensate for TCK(C) to TDO(C) route and silicon delays by adjusting the TDO recovery timing across TCK(C) periods, thus allowing the TCK(C) frequency to increase.

#### F.4.4 Supply and buffer impedances

The impedance of chip power and ground supplies along with output buffer impedances should be given adequate consideration.

#### F.4.5 Additional chip considerations related to multi-TAPC topologies

Note that subclause F.6 contains additional guidance regarding chip designs that are specific to supporting multi-TAPC topologies while using the Advanced Protocol.

### F.5 Board considerations

#### F.5.1 Signaling requiring a termination scheme

PWB loads presented to DTS and Chip output buffers fall into the following two categories:

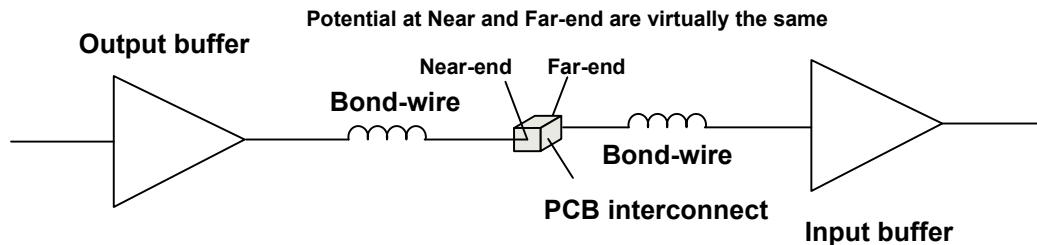
- Lumped load
- Distributed load

When a signal changes state, a rising or falling edge with certain duration begins at a one end of the trace and propagates along the trace toward the end of the trace. For a short trace (relative to the transition time), all points in the line appear to react together with a uniform potential and act as a lumped load.

With longer traces it takes some time for the signal edge to reach the end of the trace because of the propagation delay and length of the trace. As the signal propagates along the trace, the potential is not uniform at all points in the trace (distributed load) when viewed at a single point in time.

### F.5.1.1 Lumped load

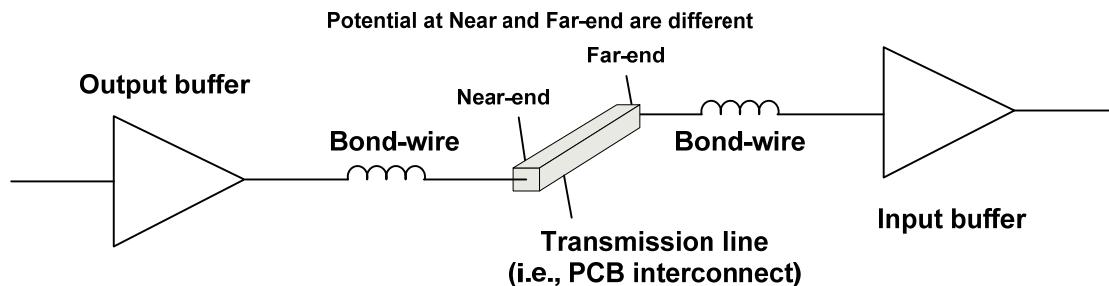
A buffer in a lumped system will see a capacitive load (PCB trace + receiver load). When the output buffer switches, the potential at the near and far end of the line are different but essentially the same, as shown in Figure F-10. In this case, the timing of the output buffer itself is valid for the total system. The timing of the I/O buffer itself is valid for the total system. Note that the buffer transition times have to be calculated with the actual lumped load capacitive value of the system.



**Figure F-10 — Lumped load**

### F.5.1.2 Distributed load

In a distributed load system, the PCB wire will act as a transmission line. When the output buffer switches, the potential at the near and far end of the line is different as shown in Figure F-11.



**Figure F-11 — Distributed load**

In this case, the timing of the output buffer itself is not valid for the total system. The characteristic PCB trace impedance (the combined effect of resistance, conductance, inductance and capacitance in the PCB line) is the load for the output buffer. The line acts like a (AC) resistive load to driver. As a result, the incident wave is the division of the source impedance and the line impedance.

### F.5.1.3 Classifying a load

The load created by a PCB signal trace is classified as either lumped or distributed based on the length of the trace connected to the output buffer and the transition time of the output buffer. This classification determines whether the signal connectivity is required to be treated as a transmission line.

The rules of thumb for classifying a load are listed as follows:

- Traces with a one-way delay five times smaller than the signal transition time may be treated as a lumped load.
- Characteristic impedances typically used on a Epoxy FR-4 printed circuit board vary from about  $50 \Omega$  to  $75 \Omega$  and have a tolerance ranging from  $\pm 20\%$  (no cost high volume),  $\pm 10\%$  (standard process), to  $\pm 5\%$  (advanced process with extra cost). The following examples target an impedance of  $50 \Omega$ . The propagation delay of a typical printed circuit board (Epoxy FR-4 substrate) is in the order of 140–160 ps/25.4 mm (1 in). These numbers vary based on the dielectric constant of the FR-4 substrate materials and the structure of the stack-up used. The actual specifications of the PCB to be built should be used.

As a rule of thumb, the maximum PCB track length between an output buffer and input buffer *without termination* is given by:

$$\text{Length (mm/in)} < \text{Tr (ps)} / 5 \times \text{Tpdt (ps per 25.4 mm (1 in))}$$

**Tr == the rise and fall time in picoseconds**

**Tpdt == the propagation delay of a transmission line or trace**

That is, the signal propagation delay from output to input is less than one fifth of the signal rise time. When the distance between the output and input buffer is less than this distance, the load can be considered a lumped load. When the distance between the output and input buffer is more than this distance, the load should be considered a distributed load or transmission line. This formula shows this length is directly proportional to the signal transition time.

For example, if a signal rise time (Tr) is 1 ns (1000 ps) and the propagation delay of the trace (Tpdt) is 140 ps to 160 ps per 25.4 mm (1 in) (typical for a PCB made with FR4 laminate), L is required to be less than  $1000 / (127 \times 160)$  mm ( $(1000 / (5 \times 160))$  in). That is, L is required to be less than 32 mm (~1.25 in). If the PCB trace length from an output buffer to an input buffer is more than 32 mm (~1.25 in), transmission-line termination should be used. The 1 ns edge rate makes this distance rather small, and transmission-line termination is needed in almost all instances. Alternately, if the edge rate can be slowed to 3 ns then traces and discontinuities less than 95.25 mm (~3.75 in) can be tolerated before transmission-line termination is needed.

#### F.5.1.4 Transmission-line impedance

An ideal transmission line consists of a continuous, perfectly distributed inductor/capacitor structure which is used to conduct an electrical signal. A coaxial cable, used to carry RF and other high-frequency signals, is a good example of a transmission line. A transmission line is typically described by two parameters: its characteristic impedance and its propagation delay. The characteristic impedance is usually in the  $50 \Omega$  range for the types of transmission lines that are involved in the use of this standard. The propagation delay is a measure of how quickly a signal propagates down a transmission line.

Within a system, every part of the signal path, from the DTS to the TS, can be viewed from a transmission-line perspective—connectors, cables, PCB traces, passive components such as resistors, and so on. The smaller elements, such as resistors, and shortest traces can often be ignored (depending on the signal edge-rates and termination scheme selected). The other elements such as cabling and the longer PCB traces cannot be ignored and require careful management. The interconnection architecture described in this annex utilizes  $50 \Omega$  transmission lines  $\pm 10\%$ . This is compatible with common PCB, cabling, and connector technology.

The I/O drivers are designed for a certain PCB trace impedance range to limit the under/overshoots and to make sure data is transported within one reflection (consider external resistance also). Do not expect good

performance when the drivers are undersized. Parallel termination is less susceptible to reflections created by the driver impedance provided the driver impedance is sufficiently low, and more susceptible to driver impedance impacts on input noise margin when the signal does cover the full input voltage swing of the input buffer

### **F.5.2 Impedance discontinuities/symmetry**

In a transmission line, any point of impedance mismatch will result in signal reflection. In some instances, this reflection results in a degradation of signal quality. In other instances, it is carefully managed as part of the circuit's operation. In all cases, the signal reflections require proactive management. Referring to the previous transmission line example of coaxial cable, a bad connector, a physical defect in the cable, improper use of splitters, or use of poor quality splitters are examples of ways such a signal delivery system can fail, creating poor signal quality and either degraded or no service. The same is true of the signal routing on a PCB when signals with high frequencies are involved (as is the case with this standard).

A voltage wave front propagates down the transmission line's inductor/capacitor structure when a signal transition occurs at the start of a transmission line. This wave front mirrors the originally applied signal edge. The driving voltage source causes current flow into the transmission line, which "charges" the capacitive portion of the transmission line. The inductive component of the transmission line resists a change in this current flow, while the resistive component of the transmission line dissipates some of the wave front's power as it propagates. As the wave front initially propagates down the transmission line, this charging current will be a constant level. If simple resistance exists in series with the transmission line during this wave front propagation period, there will be a voltage drop across this resistance, which will also appear constant during this period. Normally this resistance is small. This can be observed in the series-termination waveform shown in Figure F-12 as the flat spot that is present in both the rising and the falling edges of the signal labeled  $V_{SRC}$ .

An ideal inductor appears to have zero resistance when a constant current flow is present. However, if the impedance changes for any reason, then a corresponding voltage change occurs. Consider the basic equation for the voltage across an inductor:

$$V = L(di/dt)$$

As a wave front propagates down a transmission line, any sudden change in impedance will require a change in the current level to maintain the voltage of the wave front propagating down the transmission line. The inductive element of a transmission line resists a change in current, causing a corresponding change in the voltage level occur, per the above equation. This change in the voltage level creates a new wave front edge that propagates backward on the transmission line toward the source as more or less current is required. This "reflected" signal is additive (or subtractive, depending on its polarity) to the original signal level and will be seen by all components on the affected portions of the transmission line. This is the source of signal degradation or voltage changes. The new signal component, combined with the original signal component, can produce overshoot, undershoot, and signal edge reversals at the point where input will respond to them (monotonicity issues/glitches).

When the impedance of the driver is smaller in comparison with the line impedance, the incident wave is higher than half the supply voltage. This will result in a reflected signal that is higher than the steady-state condition, resulting in overshoots. If the impedance of the driver is larger in comparison to the impedance of the line, the incident wave is lower than half the supply voltage. This will result in a reflected signal that is lower than the steady-state condition.

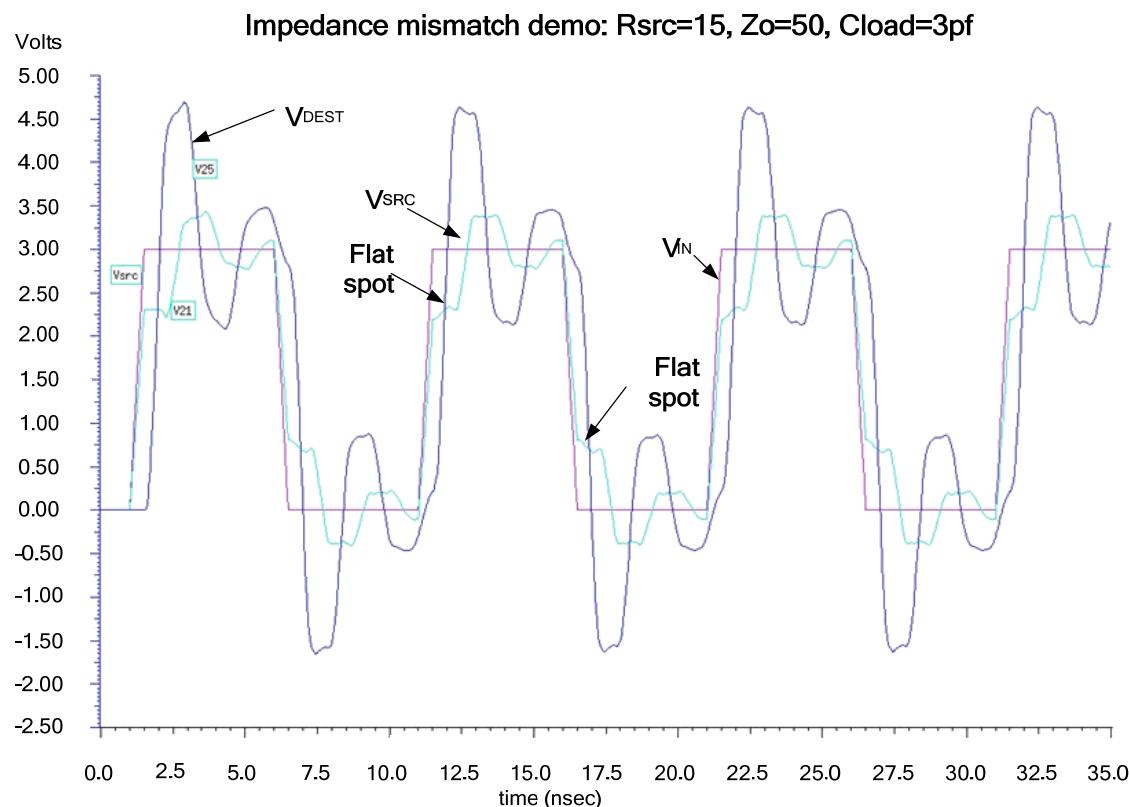
Two simulation waveform plots that demonstrate impedance mismatch between the driving source and the transmission line are shown in Figure F-12 and Figure F-13. Both are of the same circuit type—the simple series termination scheme shown in Figure F-3. In both cases, the PCB trace (the transmission line) is 100 mm in length and has a  $50\ \Omega$  characteristic impedance. The source impedance in Figure F-12 is  $15\ \Omega$  (vs. an ideal  $50\ \Omega$ ). The source impedance in Figure F-13 is  $85\ \Omega$  (vs. an ideal  $50\ \Omega$ ). The source impedance is almost entirely resistive in these cases.

**Figure F-12 Simulation Parameters**

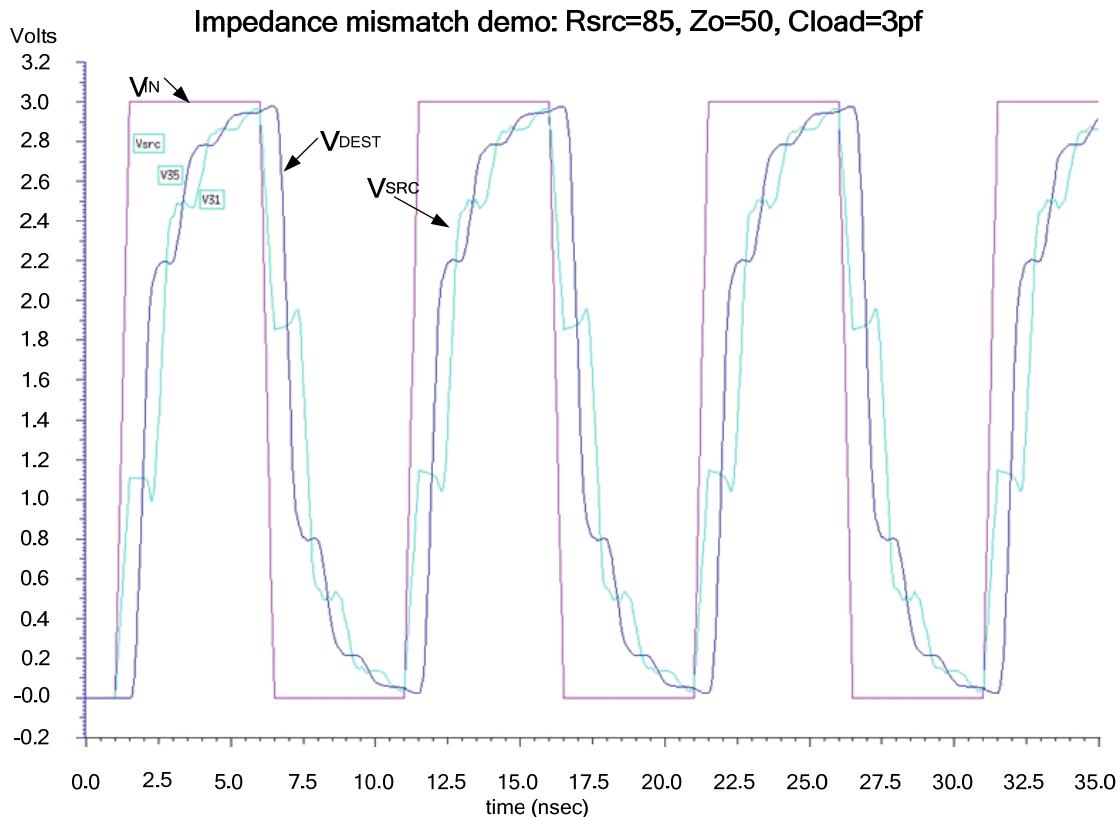
$C_{LOAD} = 3 \text{ pF}$   
 $R_{SRC} = 15 \Omega$   
 PCB trace: 100 mm/50  $\Omega$   
 $V_{21}$  = voltage at the start of the PCB trace  
 $V_{25}$  = voltage at the load

**Figure F-13 Simulation Parameters**

$C_{LOAD} = 3 \text{ pF}$   
 $R_{SRC} = 85 \Omega$   
 PCB trace: 100 mm/50  $\Omega$   
 $V_{31}$  = voltage at the start of the PCB trace  
 $V_{35}$  = voltage at the load



**Figure F-12 — Series termination with impedance mismatch causing under/overshoots**



**Figure F-13 — Series termination with impedance mismatch causing monotonicity issues**

Figure F-12 illustrates significant overshoot—as much as  $\pm 1.75$  V with the chosen parameters, relative to the original 3 V driving signal level. Figure F-13 illustrates edge-quality degradation and monotonicity issues. In both cases, the composite waveform at the load is the result of the additive effects of the original signal and multiple reflections due to impedance mismatch between the driving source and the transmission line. It is important to note that the duration of the overshoots/undershoots and the steps in monotonicity are roughly two times the signal propagation delay from the point the signal is observed to the impedance discontinuity (the time it takes for the incident wave to reach the discontinuity and the reflected wave to return) with a transmission line that is a Point-to-Point or a Line configuration shown in Figure F-2.

With the T, X, and XT connectivity configurations shown in Figure F-2, each point where the transmission line splits presents an impedance discontinuity, where the impedance seen by the source is lower. In these cases, the source impedance is lowered (especially applicable to the DTS source impedance) to raise the current in the transmission line when the incident wave is first generated, so the voltage at impedance change is higher when the current divides at the split in the transmission line. Simulations will later show lower driver impedances provide better performance in these instances. When driver impedances are not lowered, the monotonicity of signals is affected and they take longer to reach steady state.

It is important to note that the incident and reflected waves are additive in the time domain. For a series termination scheme, the signal lengths should be kept as symmetrical as possible in these configurations as it is desirable that the reflected waves in both halves of a split in the line arrive at the split point at the same time when they are reflected. Since half the power in the incident wave traverses each leg of a split, the reflected waves add when they reach the split, combining into a reflected wave with twice the power. This has the effect of making these legs look like a single leg. In the perfect series-terminated scenario, with matched loads, and only a single TAP per leg, the combined reflected wave of two legs is similar to that returned from a single leg with one load.

Note that the TAPs at radius R2 should be removed first, and entire legs should be removed when both TAPs in the leg are removed. Achieving satisfactory behavior requires this simple rule set be followed. Failure to match the length of legs will create additional reflective waves, with both the time needed for the signal to reach steady state adversely affected and the monotonicity of the signal adversely affected. Disregarding the requirement for symmetry of the leg lengths shown in Figure F-2 may result in a nonfunctional system.

### F.5.3 Transmission-line construction

#### F.5.3.1 Uniformity of transmission-line impedance

The board design requires maximizing transmission-line impedance uniformity. Board transmission lines are required to be  $50 \Omega \pm 10\%$  as the DTS expects TS signals with this impedance.

If the characteristics of the TS PCB tracks are known, then the actual trace impedance and propagation delay should be used to determine signaling behavior. The following guidelines for microstrip (track on outer layer over a ground plane) on FR4 PCB may be used otherwise:

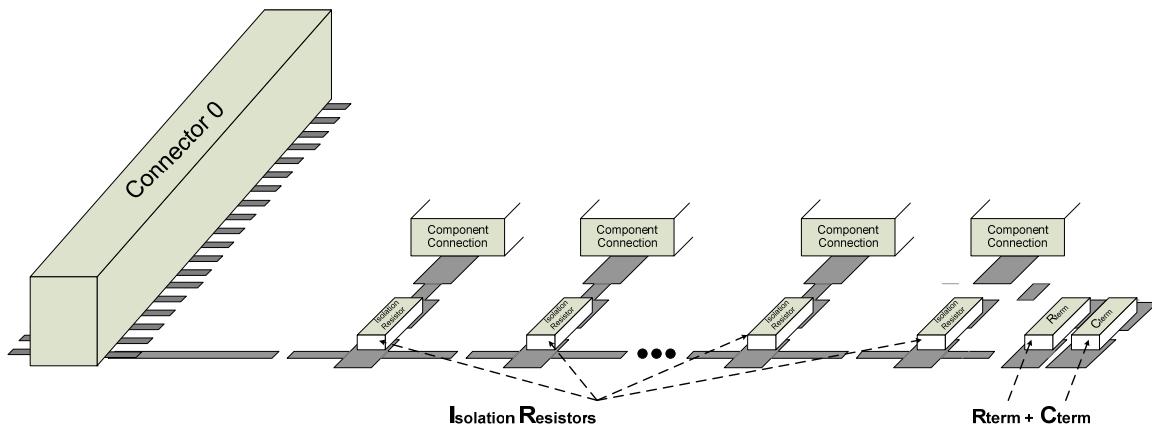
Propagation speed is approximately 55 ps/cm to 63 ps/cm (140 ps/in to 160 ps/in).

The impedance of a 1.27 mm (~.005 in) wide track as a microstrip is approximately  $50 \Omega$  on a typical six-layer foil construction board. The impedance of a track reduces as the width of the track increases. To design the Test Access Port connections, it is important to know the DTS and Chip output buffer impedance and signal edge rates for the signals of interest.

#### F.5.3.2 Proper construction

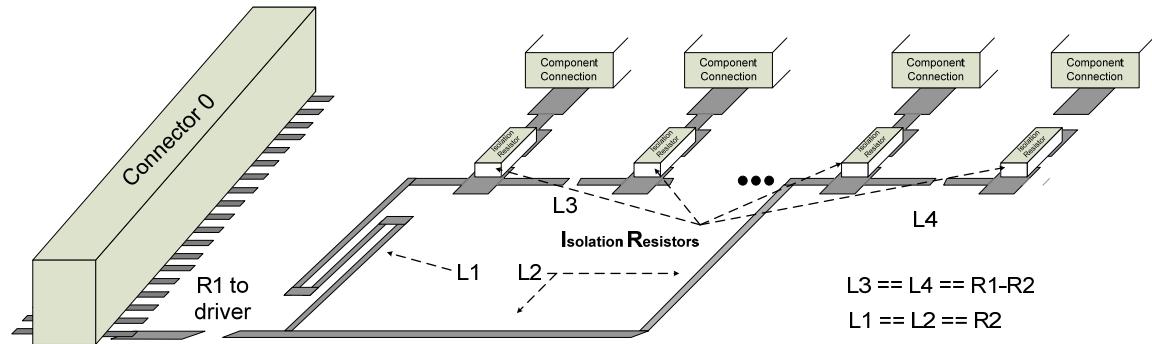
The proper connection of signals shared by the DTS and multiple TAPCs for a Line configuration is shown in Figure F-14. When possible, route the transmission line from the DTS through each load component pin of each component, and then to the termination resistor provided a parallel termination scheme is used. The termination resistor and capacitor components are omitted with a series termination scheme.

When it is not reasonable to route through each load component pin, keep the total stub length less than about one fifth the rise/fall time. Note that the choice of termination schemes and the use of isolation resistors are left up to the designer. If a parallel AC termination is used improperly (in a way where DC offsets accumulate), it may add more risk and cost than benefit. The  $R_{TERM}$  and  $C_{TERM}$  components are added as appropriate for parallel termination when this termination scheme is used. Note that if a parallel AC termination is used improperly (in a way where DC offsets accumulate), it may add more risk and cost than benefit. It is very good practice to make the stub length from the isolation resistor or chip pin to the transmission line zero (as short as possible). Ideally, the transmission line passes through the pad to which the isolation resistor is connected as shown. This resistor is placed adjacent to the component connection for the signal.



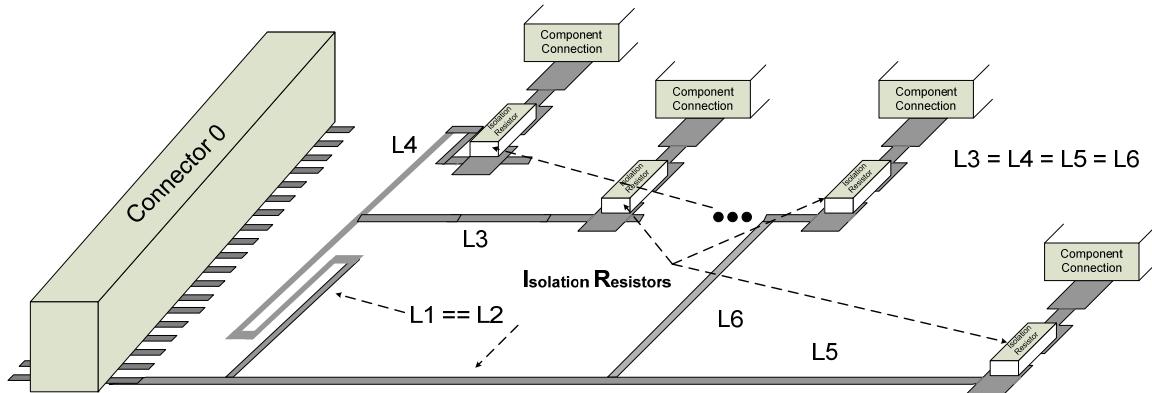
**Figure F-14 — Proper connection of TAP signaling on a board in a Line configuration**

The proper connection of signals shared by the DTS and multiple TAPCs for a T configuration is shown in Figure F-15. Note the balancing of the signal lengths.



**Figure F-15 — Proper connection of TAP signaling on a board in a T configuration**

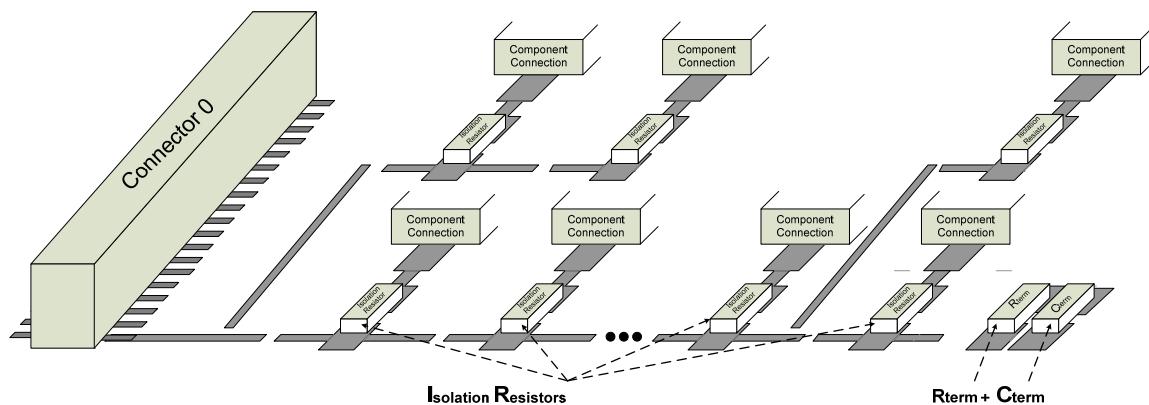
The proper connection of signals shared by the DTS and multiple TAPCs for an X configuration is shown in Figure F-16. Note the balancing of the signal lengths. The proper connection of an XT configuration is an extension of the X connection configuration.



**Figure F-16 — Proper connection of TAP signaling on a board in an X configuration**

### F.5.3.3 Improper construction

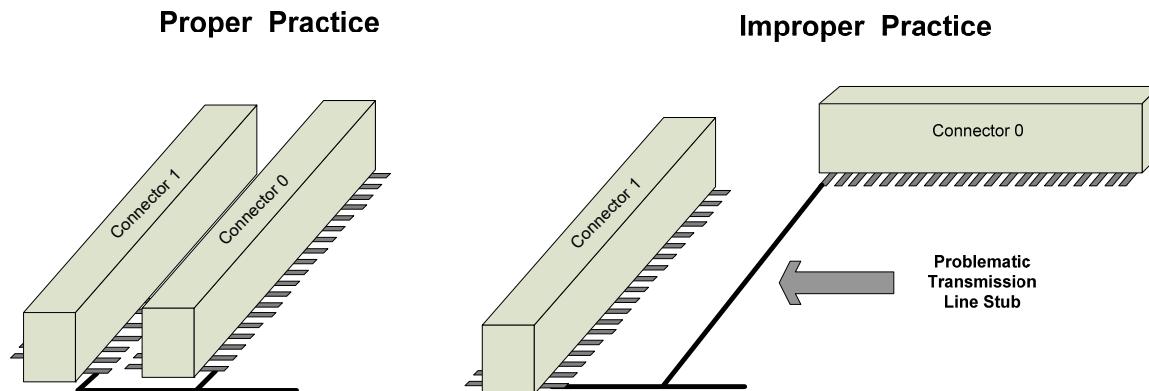
The connection of multiple TAPCs in a manner similar to that shown in Figure F-17 is a bad practice for any TAP signals and especially bad for the TCK(C) signal. Each time the transmission line splits there is an impedance discontinuity, with the transmission-line impedance being lowered. This practice should be avoided when not using a T, X, and XT configuration as it adversely affects signal quality in an extreme way. In the case of the T, X, and XT configurations, the symmetry of lengths following the splits is a requirement. Failure to follow these guidelines creates stubs that corrupt the signal's electrical integrity. The  $R_{TERM}$  and  $C_{TERM}$  components are added as needed for a parallel termination.



**Figure F-17 — Improper connection of TAP signaling on a board with parallel termination**

### F.5.3.4 Connector treatment

The connection between the DTS and TS should be made using connection technology providing an impedance of  $50 \Omega$ . In cases where multiple connectors must be used to provide DTS/TAP connectivity, the proper practice shown in Figure F-18 should be followed.



**Figure F-18 — Proper connection of TAP signaling with more than one connection**

### F.5.4 Choosing a connection configuration

A number of factors should be considered when choosing a connection configuration. A few of these factors are as follows:

- Number of TAPs

- Desired operating frequency
- DTS and chip source impedances

The system designer is responsible for ensuring these factors are properly considered.

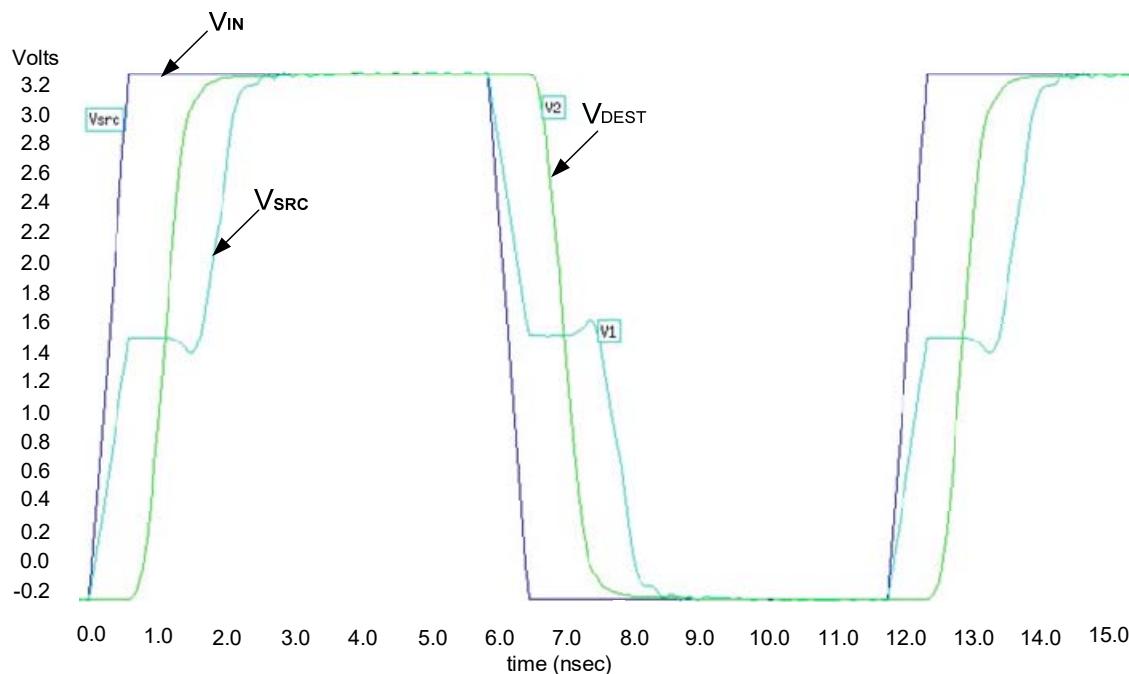
### F.5.5 Termination schemes with simple configurations

Both series and parallel termination schemes are evaluated in this subclause. The advantages and disadvantages of these termination schemes are discussed in more detail in the following subclauses. An individual knowledgeable in the subject area may consider moving to the next subclause.

#### F.5.5.1 Series termination

##### F.5.5.1.1 Overview

Series termination uses a resistor at the immediate output of the driver, establishing the connection to the PCB trace or cable that forms the transmission line. The value of that resistor plus the inherent output impedance (resistance) of the driver (such as the DTS) should equal the driven transmission line's characteristic impedance for Point-to-Point and Line configurations. This value is typically in the range of  $50 \Omega$  with the TAP.7 architecture. When properly matched for the Point-to-Point configuration, the voltage waveforms at the  $V_{SRC}$  and  $V_{DEST}$  nodes in Figure F-3 will appear as shown in Figure F-19. With the Line configuration, the nodes on the Line configuration except the last see a signal that is similar to that of  $V_{SRC}$ , with the flat spot decreasing in duration as the node is closer to the node at the end of the Line configuration.



**Figure F-19 — Basic series termination waveform**

Note that at the connecting node between the series resistor and the beginning of the transmission line and on the rising edge of  $V_{IN}$ , the voltage waveform rises to a partial value (one half) of the driving voltage, stays there for a period of time, and then continues to rise to the full value of the driving voltage. This is

mirrored on the falling edge and all subsequent edges of the driving signal waveform. This “step” in the waveform is caused by the voltage division between the combined source resistance ( $R_1+R_2$ ) and the transmission-line impedance while current is flowing from the source into the transmission line and the destination load. When  $R_1+R_2$  match the transmission-line impedance, the partial voltage level should be one half of the driving voltage level. The duration of the waveform step is equal to the time the signal takes to reach the far end of the transmission line plus the time for the reflected signal to arrive back at the driver (i.e., two times the transmission line’s propagation delay). Also note that the voltage at  $V_2$  (the destination node) rises from  $V_{MIN}$  to  $V_{MAX}$  halfway through this voltage step (the time required for the waveform to travel from the driver to the end of the transmission line). This phenomenon requires adequate consideration when choosing a termination scheme for a given connectivity scenario. For a signal used as a clock, it adds clock skew as all TAPs do not see changes in the state of the signal at the same time.

#### **F.5.5.1.2 Challenges in series termination**

A series termination scheme performs rather well in a Point-to-Point configuration as there is only one node at the end of the transmission line. When more nodes are added with the Line configuration, the signal amplitude rises to roughly one half the full amplitude of the signal. This is near the switching threshold of an input buffer without hysteresis. Small changes in signal amplitude may cause the input buffer to switch multiple times while the input signal’s amplitude is one half of the final signal amplitude. This would cause system failures when this occurs with a signal that is used as a clock. The node at the end of the line generally does not experience this phenomenon when the connection guidelines in F.5.3 are followed. Careful attention to board layout guidelines is required to produce reliable operation.

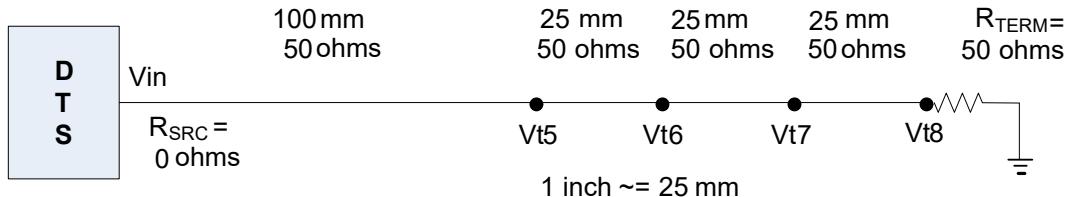
#### **F.5.5.1.3 Benefits of using series termination**

Series termination does not have a DC drive current. In simple systems, it is possible that neither isolation resistors nor termination components are needed, with this creating the lowest cost solution. At times, some systems incorporate DTS within the end product. When this is the case, the power savings provided by a series termination scheme may make the use of this termination scheme advantageous.

#### **F.5.5.2 Parallel termination**

##### **F.5.5.2.1 Overview**

Parallel termination uses a resistor as a terminating load at the far end of the transmission line as shown in Figure F-20. The impedance of the terminating load should match the characteristic impedance of the transmission line.

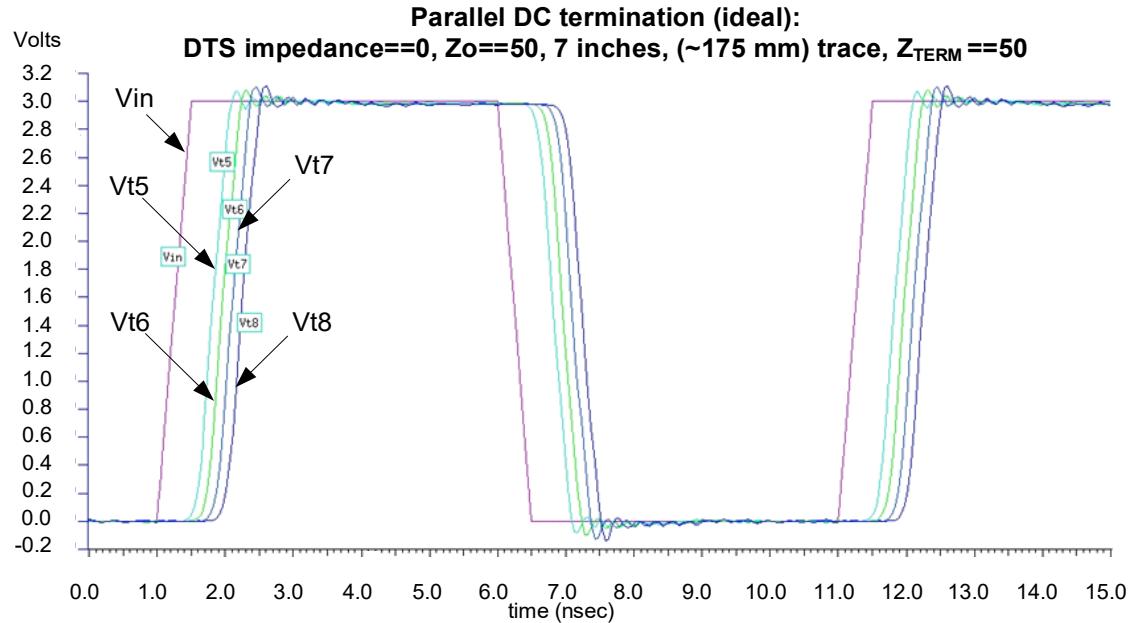


**Figure F-20 — Simple parallel DC termination**

A propagating signal edge will not encounter an impedance mismatch when it reaches the load at the end of the transmission line in this scheme. Thus, there will be no reflection and no adverse affect on signal integrity. The power in the incident wave is dissipated in the terminating resistor ( $R_{TERM}$ ). Ideally, the

driving source would have no internal resistance ( $R_{SRC} = 0$ ), and for the circuit shown below, the (ideal) waveform would appear as shown in Figure F-21, provided the termination impedance is purely resistive.

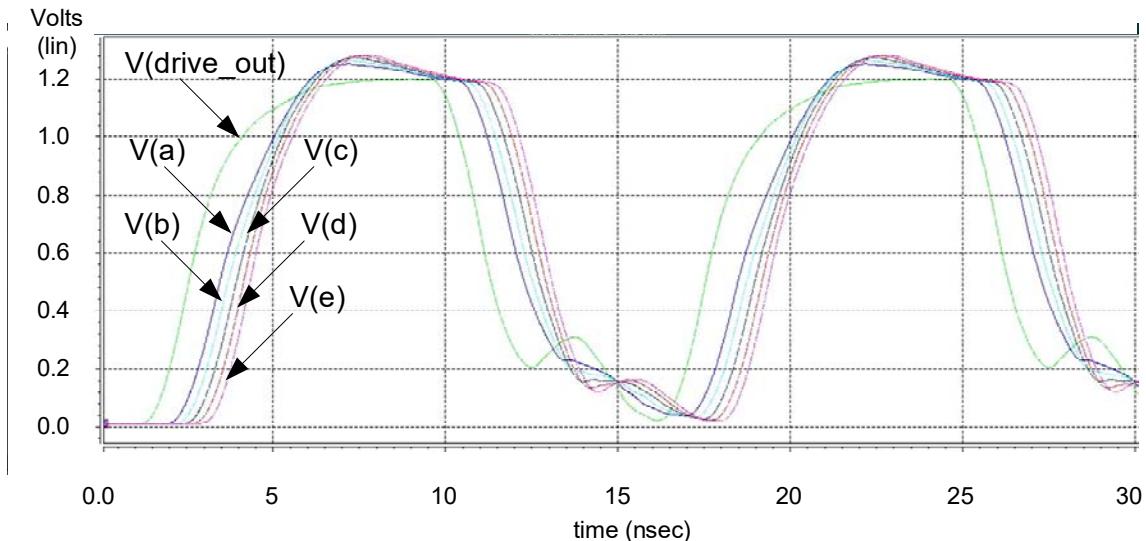
Note that the voltage waveform at the end of the transmission line ( $V_{DEST}$ ) is essentially a delayed version of the driving voltage ( $V_{SRC}$ ). As with the series termination scheme, this delay represents the transmission line's propagation delay. Also, note that  $Vt8$  is a very clean signal. When an actual end-of-line load is taken into account, the terminating impedance seen by the transmission line is no longer an ideal, perfect match, and some degree of signal degradation will occur. Taking into account component value accuracy and variations, a realistic goal is to manage this signal degradation within acceptable levels, not necessarily to achieve perfect performance.



**Figure F-21 — Ideal parallel DC termination**

Note the absence of the midlevel voltage steps, as was seen in the series termination scheme. As can be seen in the waveform, the voltages at node  $Vt5$  (~100 mm (~4 in) from the source),  $Vt6$  (~125 mm (~5 in) from the source),  $Vt7$  (~150 mm (~6 in) from the source), and  $Vt8$  (~175 mm (~8 in) from the source) are time delayed from  $V_{in}$  by the amount of time it takes the driven signal to propagate from the source to that point in the transmission line. Also note that there is no evidence of signal reflection (made possible by termination of the transmission line into a perfectly matching  $50 \Omega$  resistor). To achieve this ideal waveform, there can be no discontinuities in the signal-path impedance. To avoid impedance discontinuities, there are no TAPC loads included in this example. Also required to achieve this ideal operation is a source with zero source resistance.

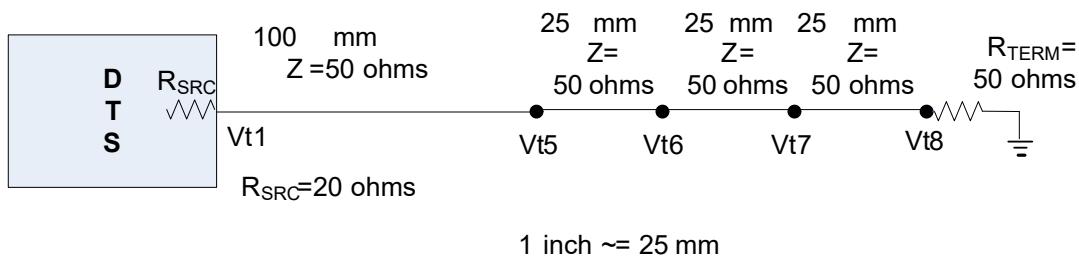
The following diagram shows the simulation results of driving 2 ns edge rates into four 3 pF loads as shown in Figure F-22 then into  $50 \Omega R_{TERM}$  after an additional 25.4 mm (~1 in) trace. The extra trace to the term shows the signal that would be detected by an additional load if the transmission line was extended. There are no isolation resistors because they are not required. The signal's rising and falling edges cleanly transition through the switching region at all loads.



**Figure F-22 — Parallel DC termination, 2 ns edge-rates, four 3 picofarad loads**

#### F.5.5.2.2 Challenges in using parallel termination

In reality, a typical driver's source impedance will not be zero, but it can be closer to  $15\ \Omega$  to  $20\ \Omega$ . Adding source impedance changes the model to look like a mix of series and parallel terminations as shown in Figure F-23, where the series termination acts as a voltage divider with the transmission line impedance to set the signal VOH level, and the parallel termination minimizes reflective reflection from the end of the line. As a result, the propagating signal will be similar to Figure F-22, except the signal's voltage swing will be 20% (at  $15\ \Omega$ ) to 30% (at  $20\ \Omega$ ) lower than with a  $0\ \Omega$  ideal driver.  $R_{SRC}$  losses can be calculated using the following equation ( $VOH = V_{SRC} * R_{SRC} / (Z_0 + R_{SRC})$ ). If left unchecked, this loss of VOH will reduce VIH input margins and could cause undesirable duty-cycle distortion at receiver inputs with a fixed 50% threshold.



**Figure F-23 — Realistic parallel termination model**

The DTS sources more DC current to drive a signal to a logic 1 with a parallel termination to ground than into a series-terminated topology. This may be a point of concern in power-sensitive applications where the DTS resides on the end product, as the steady-state current is substantial. For a DTS to drive into a parallel  $50\ \Omega$  termination to ground, it needs to source 36 mA at 1.8 V, and 20 mA to just reach 1 V.

These challenges may seem daunting, but in most cases they can be overcome through management of requirements, resources, and risk. Options a designer could use to overcome power consumption and loss signal swing while using parallel terminations are discussed subsequently. Several methods to compensate for this phenomenon are described in F.7.

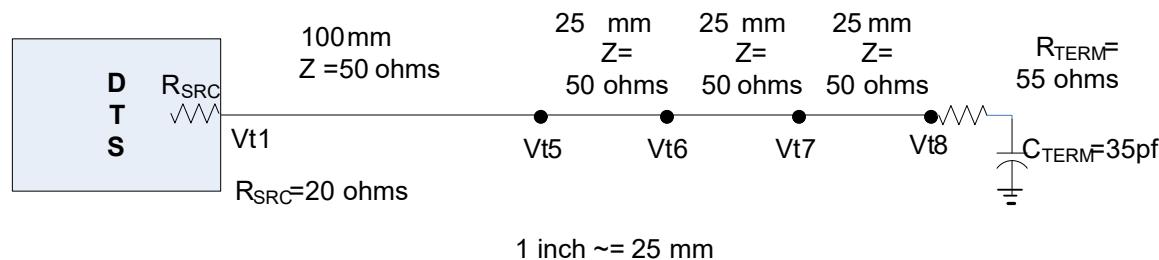
### F.5.5.2.3 Benefits of using parallel termination

Parallel termination reduces the amplitude of end-of-line reflections, which in turn reduces overshoot, undershoot, signal settling time, and additive stacking of reflections. Because signals transition quickly through all device-input transition regions (provided the logic 1 signal amplitude is sufficient), there is less reliance on hysteresis and glitch gobblers to ensure proper operation. With full-level transitions, the reflections created by minor impedance discontinuities have less impact. These benefits can be marginalized without proper logic 1 signal levels as stated previously.

### F.5.5.3 Parallel AC termination

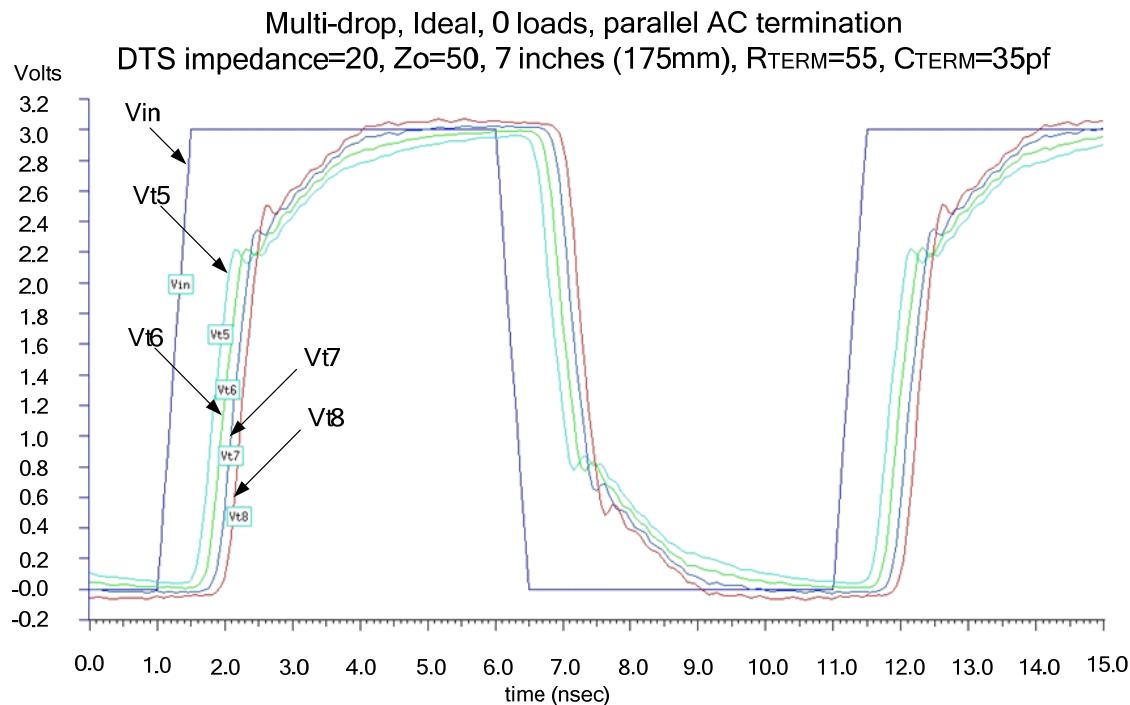
#### F.5.5.3.1 Overview

To block the DC current path, the termination scheme can be changed from simple parallel termination to parallel AC termination by the addition of a capacitor in series with  $R_{TERM}$ , as shown in Figure F-24. From a small signal and transient-analysis perspective, the termination resistor still functions as desired, but from a power perspective, the DC current path is eliminated. Note that the transitional drive-current requirements are the same as with the standard uncompensated parallel termination topology, but the driven DC loading is reduced and settling time has increased, which will directly impact operational frequency.



**Figure F-24 — Realistic AC parallel termination model**

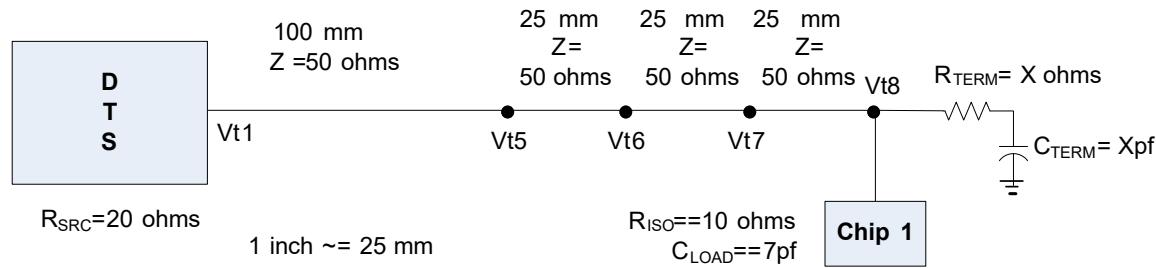
Simulating this more realistic model (there still are no loads present) yields waveforms as shown in Figure F-25 (after some iterative tuning of the various termination device values).



**Figure F-25 — Ideal parallel AC termination, No Load**

Note that the illustrated signal quality has degraded, even before loads (TAPCs) are added to the network. As the topology becomes more complex, this will become a greater and greater challenge, requiring a combination of countermeasures to insure a functional system design. In many configurations, it becomes unworkable.

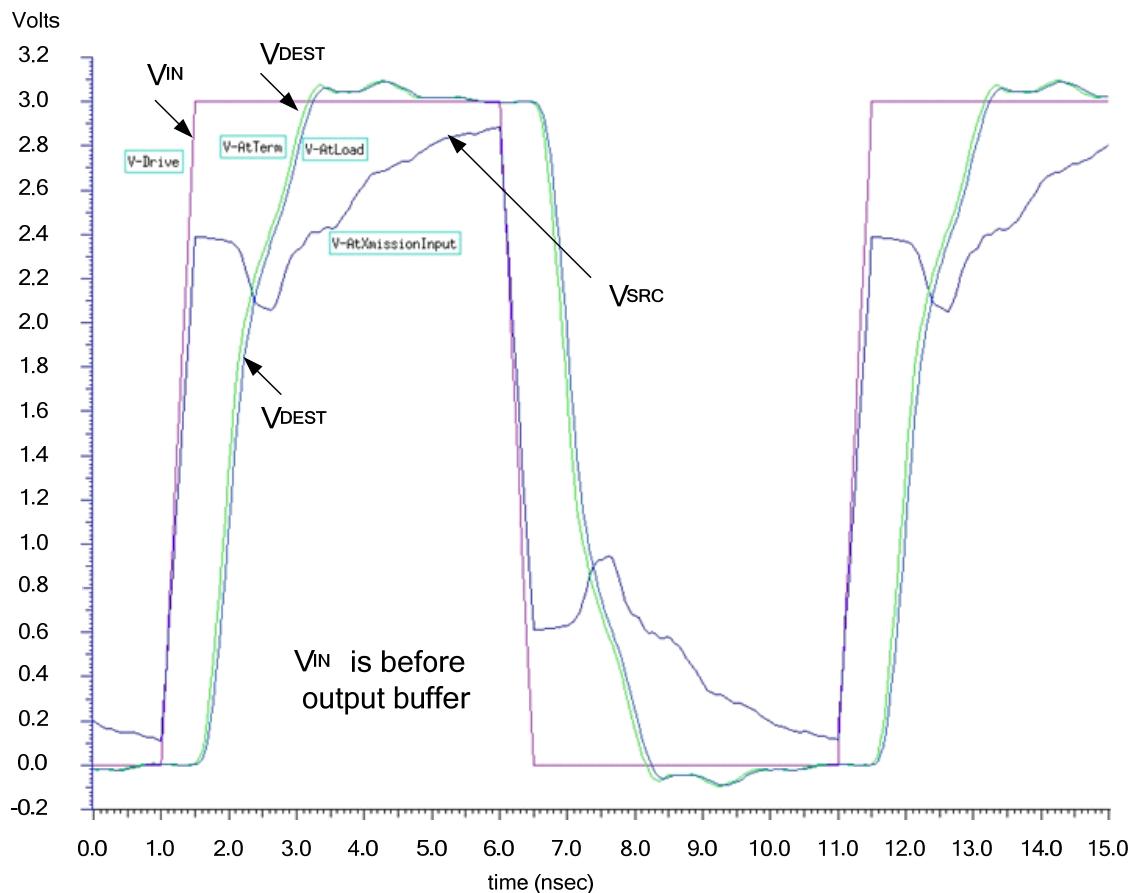
Next, one load is added to the model at the far end of the transmission line and in parallel with the end-of-line termination as shown in Figure F-26.



**Figure F-26 — Parallel AC termination model with one load**

Note that this is essentially the same circuit as the parallel AC Termination scheme shown in Figure F-4. The isolation resistor ( $R_{ISO}$ ) is discussed in more detail in F.5.7. Figure F-27 shows the results of the simulation of a simple model with typical values.

**Base Sim: RSRC=15, Zo=50, AC par term=45 ohm/40 pf, Riso=10 ohms, Cload=7pf**



**Figure F-27 — Parallel AC termination simulation results**

The actual values chosen for the parallel termination network were optimized based on iterative simulation results. In general, the more dominant the AC termination capacitor's impedance is at the highest frequency of interest, the less optimal the overall performance of parallel AC termination. In this case, a  $45 \Omega$  resistor was placed in series with a  $40 \text{ pF}$  capacitor to implement the termination scheme. The  $40 \text{ pF}$  capacitor has approximately  $8 \Omega$  of impedance at  $500 \text{ MHz}$ , meaning the resistor will be the dominant source of impedance in the termination network. The resistor's impedance is relatively constant over the frequency range of interest, while the capacitor's impedance is a direct function of frequency. Five hundred megahertz is a frequency point of interest since it is the fifth harmonic of  $100 \text{ MHz}$ , the frequency of the driver output signal shown in the simulation. Square waves are made up of the fundamental and the odd harmonics, with the third and the fifth being a reasonable number of harmonics to consider for such a simulation (use even higher harmonics for signals with faster rise/fall times). As can be seen in the waveform labeled  $V_{DEST}$  (voltage at the load capacitor), an acceptable signal quality was achieved.

#### F.5.5.3.2 Challenges in using parallel AC termination

In reality, parallel AC termination seems viable only when the termination scheme is used with a periodic signal with 50% duty cycle. The reason for this is the termination extracts a DC component from the signal when the data pattern does not meet the two requirements identified previously. This can dynamically change the switching threshold.

### **F.5.5.3.3 Benefits of using parallel AC termination**

Parallel AC termination provides some of the benefits of parallel termination and does not create a DC current path. These benefits can be marginalized in the absence of proper design and analysis. In general, a parallel AC termination scheme may need more fine tuning than the Series or Parallel termination schemes.

### **F.5.6 Effects of transmission-line length on operating frequency**

When a series termination scheme is compared to a parallel termination scheme, as the transmission line length increases, the parallel termination scheme allows operation at a higher frequency.

With a series termination scheme, a signal propagates from the source to the end of the transmission line and back to the source to produce a full-level signal at all nodes. A signal's incident wave produces a half-level signal swing (provided the source impedance and line impedance are the same) at all nodes, and the energy in incident wave reflects from all line endpoints to reach the full level of the signal at all nodes. This creates the need for the countermeasures on inputs as described previously. In a well-designed series termination scheme, the settling time of a signal at any node is 2x the propagation delay from the node to the end of the line. This settling time can be a significant portion of the timing budget.

With a parallel termination scheme, a signal's incident wave produces a full-level signal swing at all nodes, and any reflected energy may be dissipated at either driver or the termination. With the driver being such a low impedance ( $\leq 10 \Omega$ ) and the termination resistance approximately  $50 \Omega$ , most energy reflected to either end of the line is dissipated. This lessens the need for the countermeasures on inputs as described previously in a well designed parallel termination scheme. The settling time of a signal at any node is essentially zero.

Additionally, because of full-level signals, it is very likely that the signal at all nodes where a connection to the transmission line is made, settles faster as reflections are damped at both the source and the end of line, regardless of their source. This being the case, a topology with fewer reflections will settle sooner than one relying on reflections.

For series-terminated topologies, the impact of reflections on operating frequency are more significant as described below. A line that has a delay smaller than the unit interval of signal change divided by two is described as a relatively short line (unit interval to be divided by two because the signal has to propagate up and down the PCB trace).

In case two times the line delay is larger than the required settle time, the line delay determines the maximum frequency and is included in the definition to determine the maximum frequency:

$$f_{\max} = \frac{1}{(2 \times t_{t(\max)}) + (4 \times t_{p\_pcb})}$$

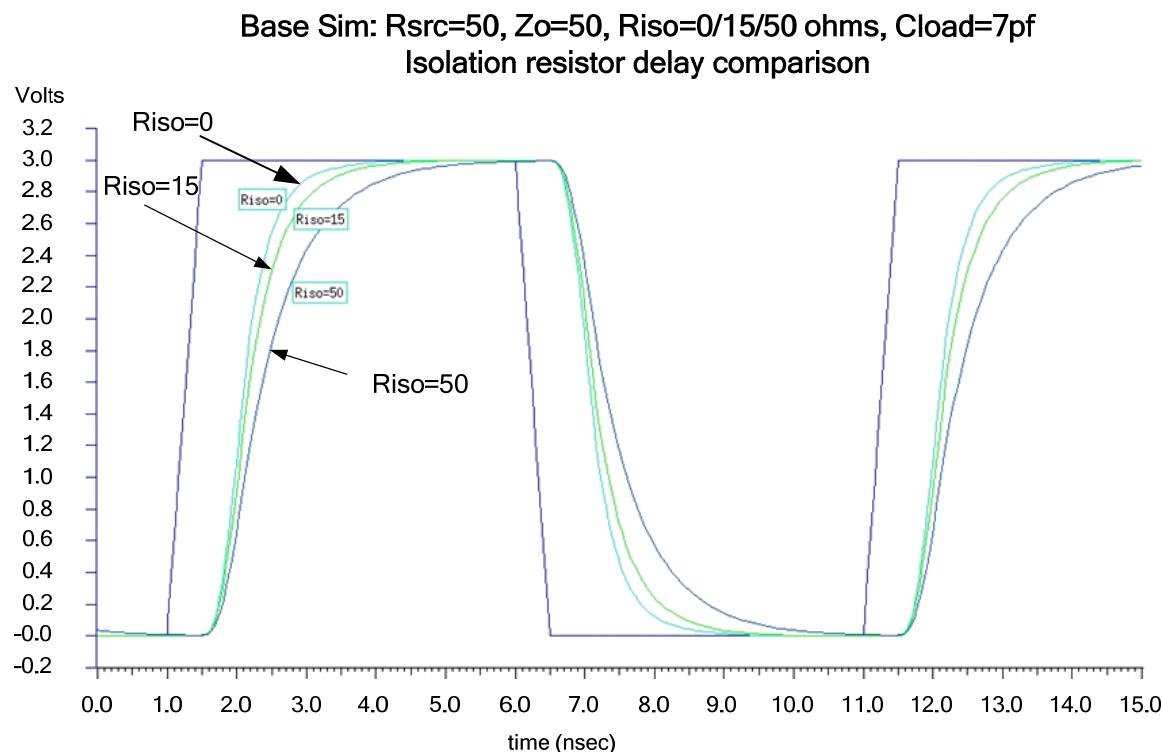
$t_t$  = signal change interval

$t_{p\_pcb}$  = pcb propagation delay

Because of mismatches between driver impedance and line impedance, series-terminated topologies should change the state of a signal once the reflection from the previous state change arrives back at the driver. Wave fronts remaining in the line could result in incorrect data transport if a signal change occurs before the energy in the wave front (reflections) generated by a previous signal change has been dissipated.

### F.5.7 Capacitive load isolation and input pin low-pass filtering

When an SOC load (or loads) are connected to this transmission line, impedance discontinuities are created at the points of connection, resulting in a degradation of signal quality. An effective way to reduce the adverse effects of these necessary connections is to use an isolation resistor at each point a connection to the transmission line is made. This resistor ( $R_{ISO}$ —resistor in series/output source impedance—isolation resistor in the case of an input) provides some degree of isolation of the mostly capacitive load from the transmission line. This reduces the adverse effect of the load on the quality of the signal. Figure F-28 illustrates that an effective amount of isolation resistance can be used without incurring excessive RC delay at the device pin. Waveforms are shown for three values of isolation resistance: 0 Ω, 15 Ω, and 50 Ω. As can be seen, the timing delay incurred by the larger value of isolation resistance is not excessive, in most cases.



**Figure F-28 — Basic parallel termination waveform with capacitive load isolation**

In addition to providing a degree of isolation of a load from the transmission line, a second benefit to the use of an isolation resistor is that it forms a low-pass filter for the TAPC input by creating an RC structure by its interaction with the TAPC input capacitance.

A third benefit associated with the use of an input isolation resistor comes into play when the associated TAPC pin is a bidirectional pin, switching from driver to receiver (and vice versa) as required. In this case, the isolation resistance also functions as a series source termination resistor when that node is in the driving mode.

### F.5.8 DTS and chip models

The DTS and chip models are shown in Figure F-29. They include the resistors that function as either the series or isolation resistors. These models are implied when representations of DTS and chips are shown in subsequent figures.

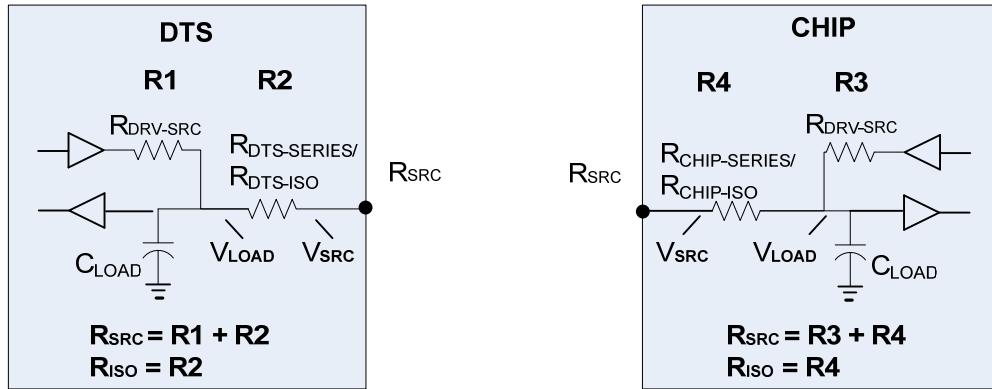


Figure F-29 — DTS and chip models

### F.5.9 Additional board considerations related to multi-TAPC topologies

Note that F.6 contains additional guidance regarding board designs that are specific to supporting multi-TAPC topologies while using the Advanced Protocol.

## F.6 System considerations for supporting Keeper bias (K bias) in multi-TAPC topologies

### F.6.1 K bias considerations at the chip and board level

In addition to the normal concerns of capacitive loading posed by multiple board traces and input buffers, the TAPCs implementing the Advanced Protocol (T4 and above) can impose significant loading issues resulting from the K bias requirement on TMS(C) when the Advanced Protocol is active. When multiple TS devices in a shared topology have activated the K bias function on TMS(C), the load seen by any driver of this signal is increased significantly. This can greatly increase the rise and fall times seen on TMS(C) and significantly degrade performance. The impact may be acceptable for topologies with a small number of TS nodes, but as can be seen from other data in this annex, topologies with a large number of TAPCs will suffer significant performance impact in general. Multiple TMS(C)s all implementing K bias simultaneously will only compound this problem.

As noted in Clause 12, and in particular 12.3 and 12.7, devices implementing the Advanced Protocol are not required to implement the K bias function as long as one chip on a shared TMS(C) topology supports K bias. So for maximum performance of the Advanced Protocol, the board and the collection of multiple TS devices would support a topology where a single chip provides the K bias function.

Since most TS devices in a multi-node, Advanced Protocol topology are not required to support the K bias function, it can be inferred that, by default, a TS would not implement this feature. Some exceptions include the following:

- The TS will never be used in a multi-drop topology.
- The TS will always be designated as the dedicated K bias device in a multi-drop topology.

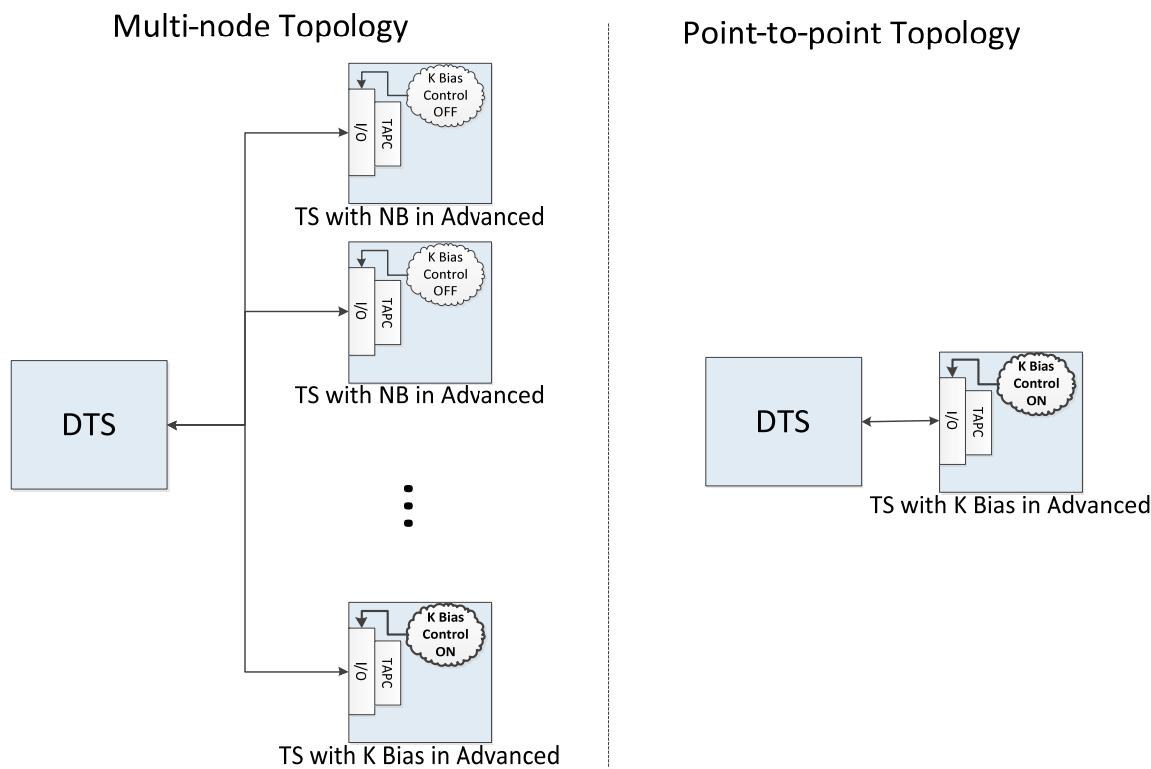
For optimum flexibility for system designers, TS devices would provide a configurable way to control the K bias function. The board designer can then select a device to provide the topology-level K bias functionality. Possible methods for providing a configurable K bias in a TS include:

- Fully static configuration where TS device vendors offer identical parts with either K bias

active or disabled permanently.

- TS device vendors provide some method of permanently changing the K bias function after packaging. In this case, the OEM can use that method to convert a device to using K bias as needed.
- A TS device pin may be sampled at device power-up and so used to control the K bias function.
- In lieu of having a TS in the multi-drop topology implement K bias, dedicated K bias devices could be created that have no functional application. But these devices would support the Advanced Protocol to the point that they would always implement the K bias function when the Advanced Protocol is active.

Figure F-30 shows both multi-drop and point-to-point topologies where a single TS is responsible for implementing the K bias function for the entire topology.



**Figure F-30 — Topologies with a single device providing K bias**

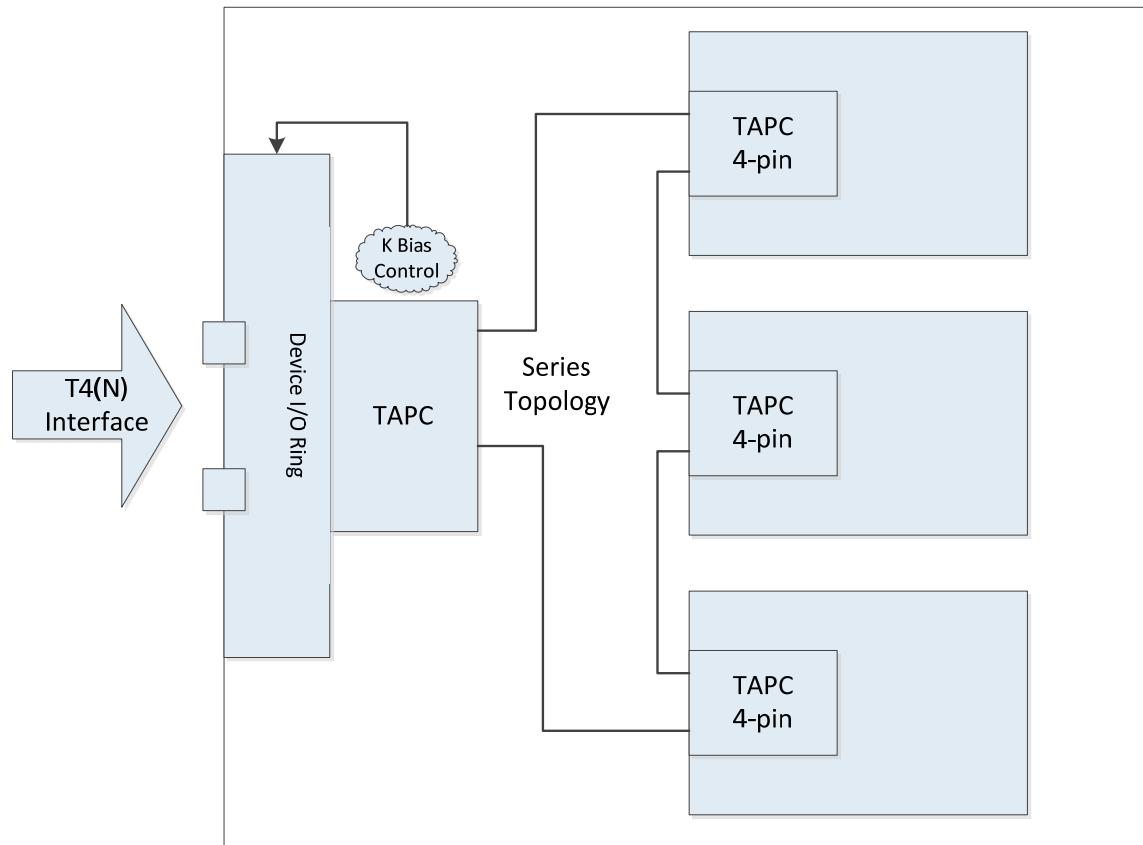
All of the methods listed above for reducing K bias loading (and likely any other method not listed) will have negative impact on the overall system design and will likely increase cost via:

- Increased TS complexity
- Increased board/system design and manufacturing complexity

Another issue that must be noted is that the 2009 version of this standard did not point out the design constraints imposed by multiple K bias loading. As such, many TS device vendors have not provided any configurability of the K bias function, and this is always active when the Advanced Protocol is active. These devices will likely prove problematic in multi-drop topologies with more than a few nodes.

## F.6.2 K bias considerations for embedded TAPC topologies

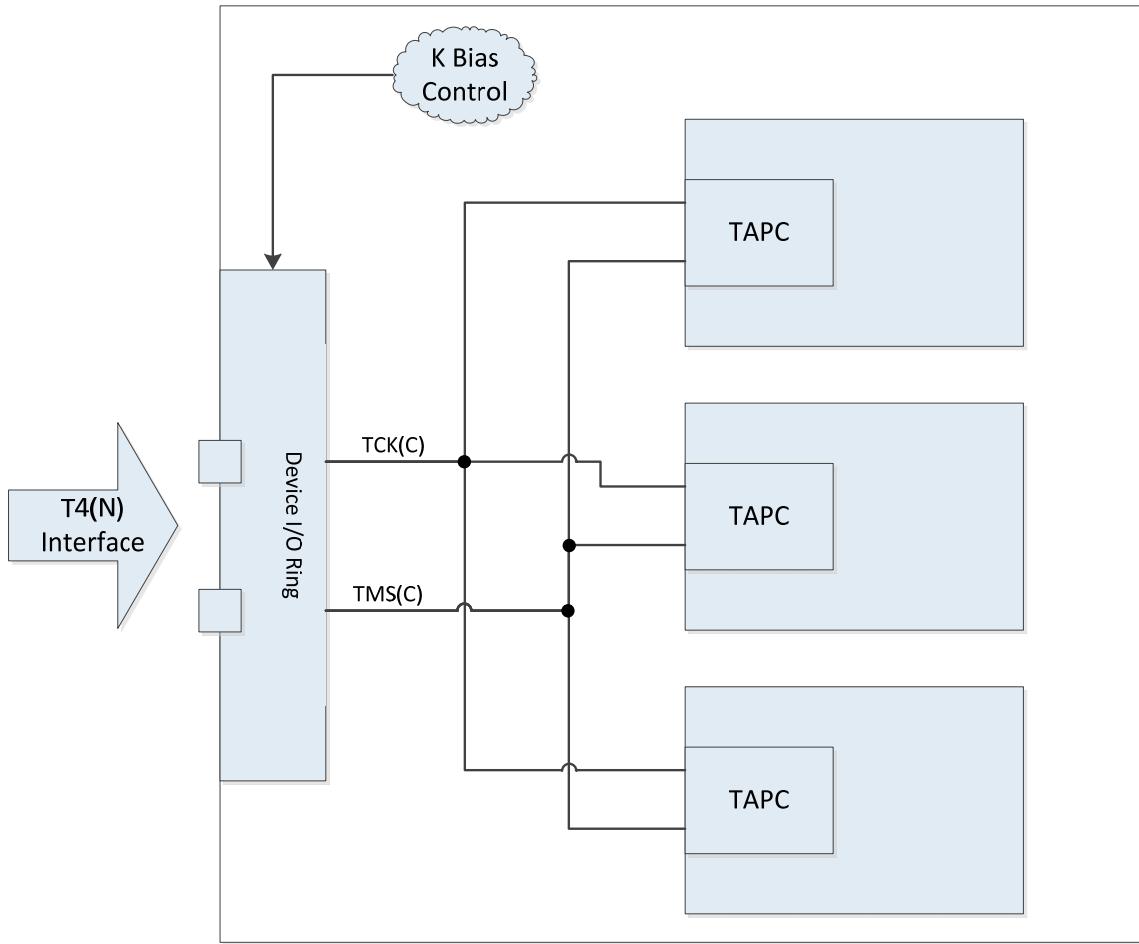
In general, when multiple TAPCs are embedded in a single TS, the TS device presents a single TAP.7 interface at the chip I/O and the embedded TAPCs can be configured to operate as T0–T3 class TAPCs using series (or possibly star) topologies. This hierarchical approach is viable because the T4(N) functionality is really only critical at the device level for pin savings. The extra signals required for T0–T3 operation do not impose any significant constraint to system design. With this configuration, the K bias functionality is managed at the top level as shown in Figure F-31.



**Figure F-31 — Hierarchical package with TAPC and K bias at chip level**

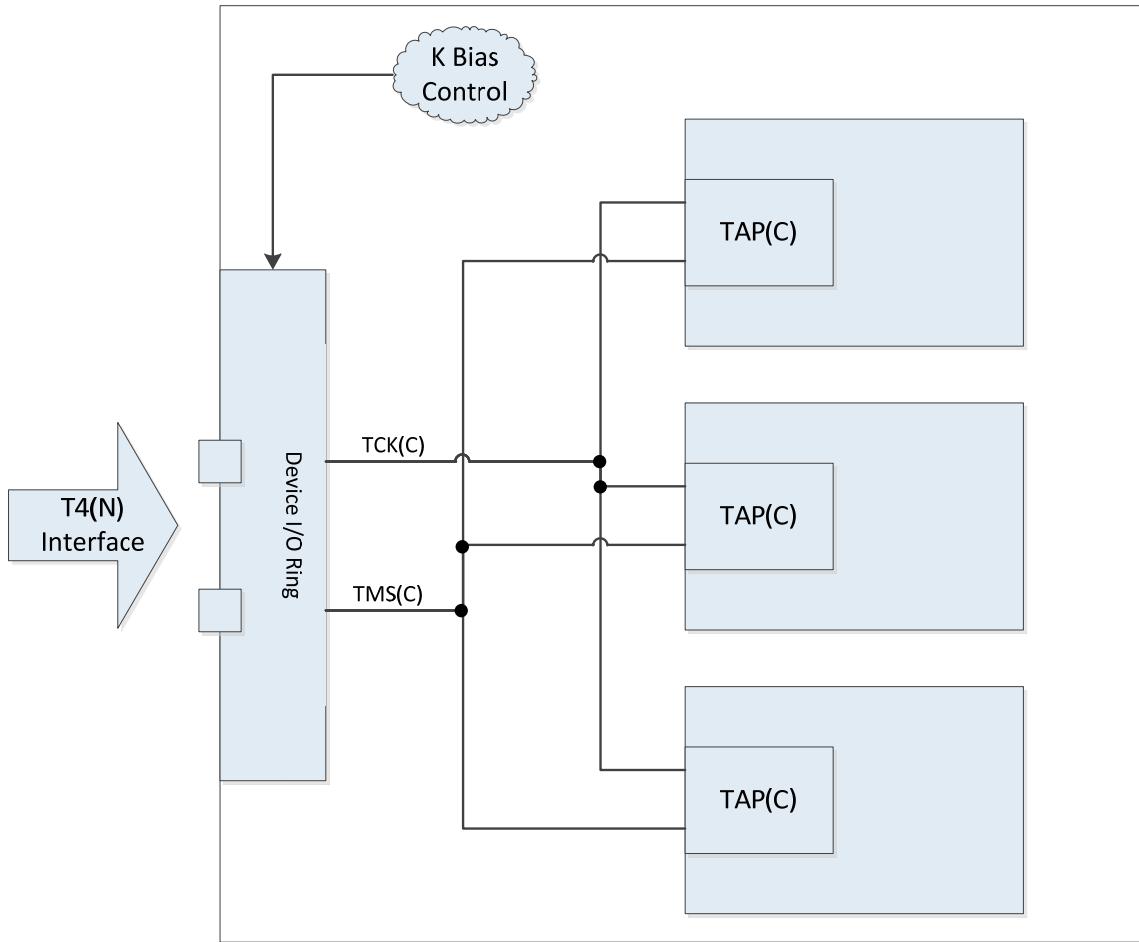
There are, however, scenarios where the hierarchical topology outlined above is not possible. In particular, a possible scenario is where multiple TAPCs with T4(N)-only capabilities are embedded in a single package. This could happen when fully hardened IP blocks are combined in a single package for cost or other reasons. In general, two approaches could be taken in this scenario.

In the first approach, the TAPC I/O logic exists close to the pins and is shared among the embedded IP. In this scenario, the design considerations regarding K bias functionality (outlined in F.6.1) would be considered for the package/device as a whole. The K bias capability is implemented in the I/O logic layer, and the controls for K bias must be implemented in some package internal logic. The primary controls are whether the device is ever required to support the K bias function (see F.6.1) and, if it is, the control for enabling it when the Advanced Protocol is active. Figure F-32 shows an example of this type of system configuration.



**Figure F-32 — Hierarchical package with top-level I/O management**

In the second approach, the TAPC I/O logic exists in each embedded IP and the pins are shared via wire bond. In this scenario, the device boundary is essentially transparent to the system and the topology behaves like a standard multi-node topology where each TAPC is a discrete device. The design considerations regarding K bias functionality (outlined in F.6.1) need to be considered for each embedded IP. Figure F-33 shows an example of this type of system.



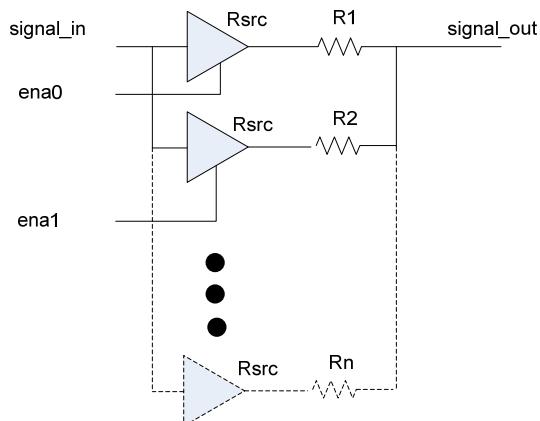
**Figure F-33 — Multi-TAPC package with shared I/O pins**

## F.7 DTS considerations

### F.7.1 Source impedance

The DTS TCK(C) and TMS(C) signals should support programmable source impedance. This may be considered for other outputs that may see different loading and termination schemes. Since the impedance characteristic of the connection topologies that may be connected to a DTS differ, a DTS should be capable of providing several different output impedances to more closely match these characteristics.

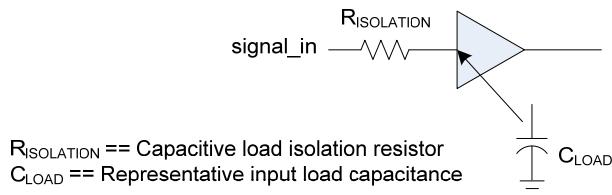
This can be accomplished using the approach shown in Figure F-34 or other approaches. Three source impedances can be created with two drivers with the number of output impedance combinations doubling for each additional driver added. These source impedances should target  $50\ \Omega$ , with one combination, with other combinations producing lower and higher impedances. DTS output impedances in the range of  $25\ \Omega$  to  $75\ \Omega$  may be required depending on the termination schemes. DTS vendors should establish the values of  $R_1-R_n$  based on the drivers used and other factors using simulation.



**Figure F-34 — Programmable DTS source impedance**

## F.7.2 Input capacitance isolation

If sufficient edge-rate control is not provided at the driver, DTS inputs can utilize input capacitance isolation resistors as shown in Figure F-35, to filter high frequencies at each input and create better transmission line integrity.



**Figure F-35 — DTS input capacitance load isolation**

### F.7.2.1 DTS output levels with parallel terminated target topologies

The DTS often supports various I/O voltages. To maintain proper output-voltage levels, the VOH output level may be elevated to compensate for the  $R_{SRC}$  losses within the DTS driver. This is accomplished by proportionally increasing the  $V_{SRC}$  drive voltage so that the resistor divider created by the driver-source impedance and the parallel-line termination creates the VOH that is expected by the TS inputs.

### F.7.2.2 DTS output edge rates

Limiting the edge rates of to DTS outputs to between 2 ns and 3 ns can produce more reliable system operation, as the reflections generated by these edge rates is substantially reduced. When the edge rates of all drivers sourcing a signal fall into this range, the board designer may consider eliminating isolation resistors, thereby reducing the number of parts required to build the endproduct.

This may have an additional benefit as the transmission-line impedance tolerance of 10% (standard process) can be used along with increasing the allowance for board + package stubs allowances can be increased up to ~2”.

### F.7.2.3 High-speed, high-current, low-voltage configurable edge-rate drivers

The NPN emitter-follower shown in Figure F-36 has been successfully used to drive high-current edge-rate controlled signals, into sub-1.8 V parallel terminated topologies at frequencies up to 100 MHz. Care should be taken to provide adequate collector-emitter voltage head room, source the base with a desired signal waveshape with the voltage offset up 1 diode drop, and the emitter follower will source current into the system tightly tracking the signal waveshape input on the base.

Care should be taken to avoid the coupling of noise onto the base of the transistor as the transistor is capable of producing drive current for the noise as well as the base signal. Transistors provide for operation with a sub-1.8 V design, have high-drive currents, have good frequency response, and are available in small dual-transistor packaging. The emitter-follower drive is provided for reference, and designers should select devices, methodologies, and topologies that best meet their design requirements.

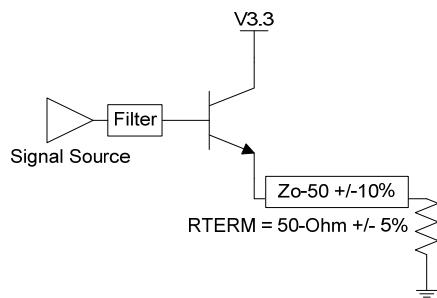
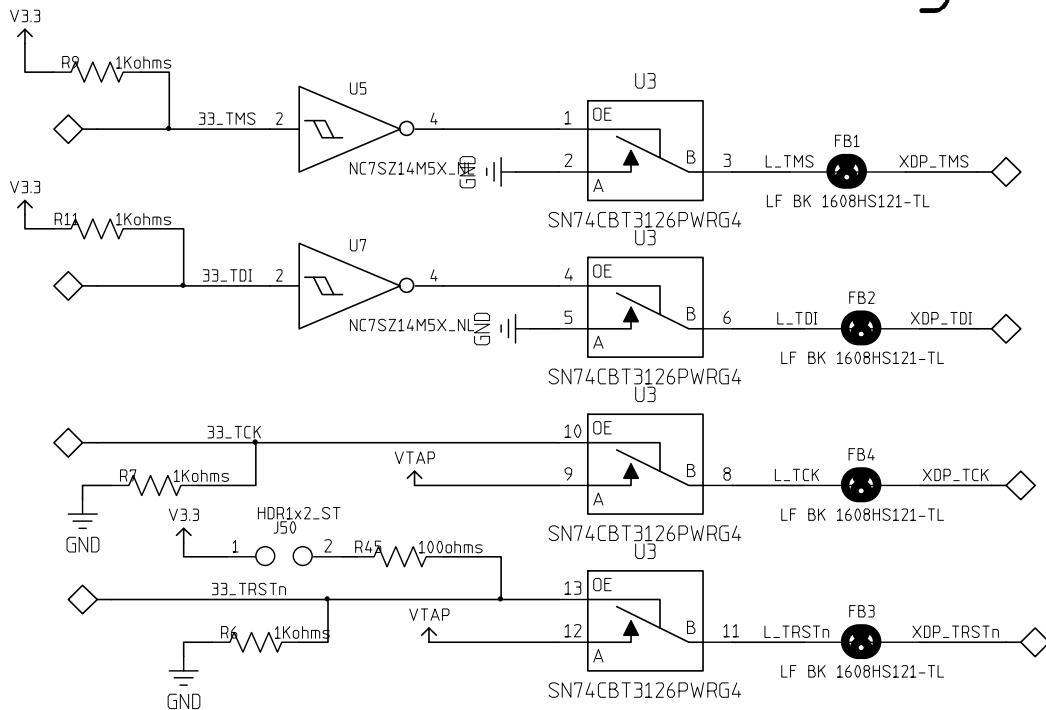


Figure F-36 — Emitter-follower driver

### F.7.2.4 Low-voltage, high-current fixed slow edge-rate drive translator

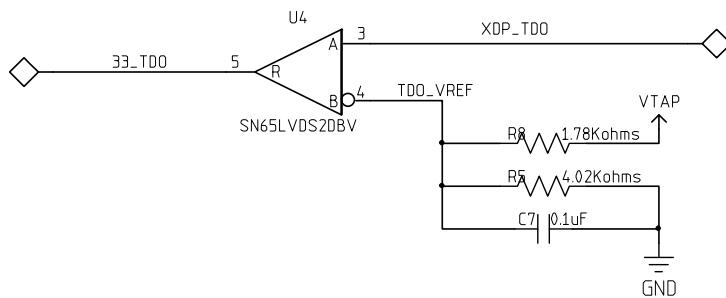
Voltage translation circuits similar to that shown in Figure F-37 have been used with a DTS to drive a slow 2.3 V signals with toggle rates of 4 MHz to 10 MHz into sub-1.5 V 50 Ω parallel terminated signals. The ferrite limits edge-rates by filtering high-frequency changes in drive-current. This slow edge-rate is very forgiving of design, loading, and layout problems in target. By selecting a new ferrite, the drive can easily be converted to operate with a faster edge rate.

# Voltage Translation Logic



Bias TRSTn LOW to ensure that the output is idle when the 3.3V master is not in use.

The Jumper is used to force TRSTn HIGH if the 3.3V JTAG master does not include a TRSTn signal.



**Figure F-37 — Voltage translator**

## F.7.2.5 In-system DTS considerations

### F.7.2.5.1 Output buffer impedance

Large rack-mount systems, concept test-boards, and validation platforms may build one or more DTS(s) directly into a volume product. With an in-system DTS, signals are often driven using FPGAs or another low-voltage device.

In this DTS use case, a designer will have to select a termination scheme compatible with the needs of the system. Normally the DTS is built into an FPGA or ASIC. With a parallel or series termination scheme requiring a low impedance driver, in this model, the power required to drive these termination schemes is provided by the TS. In the case of a parallel termination scheme, access to a higher voltage to compensate directly for the driver  $R_{SRC}$  losses may not be available. Designers can still reduce  $R_{SRC}$  losses, by paralleling output buffers. For example, if an FPGA provides 20  $\Omega$  I/O drivers, four can be used in parallel to create effectively a 5-ohm driver, which would reduce the RSRC losses from ~30% to 9%. Other low-voltage drivers can be paralleled in a similar manner. In both of these cases, calculating the current requirements of and managing simultaneous switching of paralleled drivers is required.

### F.7.2.5.2 Power considerations

Most large rack systems, concept test boards, and validation platforms are not as concerned with power consumed by the debug logic on the board during active debug. The primary concern is stability, robustness, and debug within these platforms. When not in active debug, the use of a series termination scheme does not consume significant power. The DC power consumption of parallel termination topologies can be minimized with the DTS design and use by simply driving each signal to its lowest power consumption state idle state, tristating these signals, or powering down the interface. Power is only consumed when signals are driven in a power-consuming state.

### F.7.2.5.3 Signal edge rates

Managing signal edge rates with an in-system DTS can be a little more challenging due to cost and component constraints. Various signal filtering techniques can be used to generate fixed edge rates on signals. The easiest option is generally the use of a ferrite bead with DTS outputs to attenuate the signal above the third or fifth harmonic of the maximum-operating frequency. Alternatively, various LC or single-chip low-pass passive filters can provide a low-cost, small-space solution for controlling edge rates and matching transmission-line impedances. Both LC and ferrite filters operate on changing currents; as a result, they are sensitive to loading changes in the parallel-termination resistance. A simulation of the solution that is chosen is recommended.

## F.8 Point-to-Point configuration

### F.8.1 Model

The logical model of a Point-to-Point transmission line is shown in Figure F-38.

## Logical View

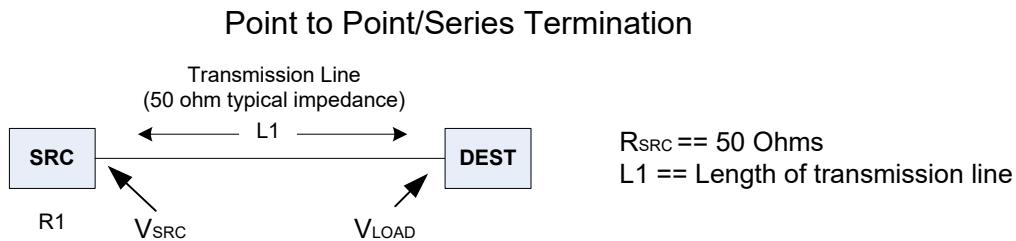


**Figure F-38 — Point-to-Point transmission-line model**

### F.8.2 Unidirectional signaling

#### F.8.2.1 Series termination

A single-driver/single-receiver (Point-to-Point configuration) configuration using unidirectional signaling with a series termination is shown in Figure F-39. In this scenario, the signal quality is of concern only at the destination of the transmission line. The logical and physical views of this signaling scheme with a series termination are shown in this figure.



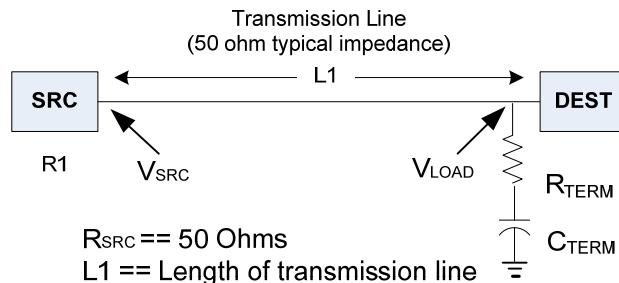
**Figure F-39 — Point-to-Point/series termination**

The use of the series termination network shown in Figure F-39 yields the waveforms similar to that shown in Figure F-19 in the subclause that introduced Series Termination when  $R_1 + R_{TERM} = 50 \Omega$  and  $R_{ISOLATION} = 18 \Omega$ .

#### F.8.2.2 Parallel AC termination

A single-driver/single-receiver (Point-to-Point configuration) configuration using unidirectional signaling with parallel AC termination is described below. In this scenario, the signal quality is of concern only at the destination of the transmission line. The logical and physical views of the parallel AC termination signaling scheme are shown in Figure F-40.

### Point to Point/Parallel AC Termination



**Figure F-40 — One source/one destination/unidirectional signaling/parallel AC termination**

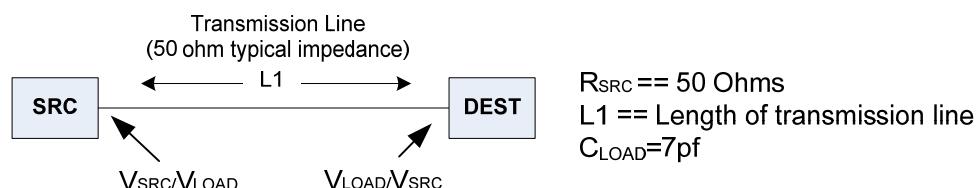
The use of the parallel AC termination network shown in Figure F-40 yields the waveforms similar to that shown in Figure F-25.

### F.8.3 Bidirectional signaling

The next level of complexity includes bidirectional signaling on a given line. The TMSC signal falls into this category when the advanced capabilities of this standard are implemented. Only series termination was considered. Parallel AC termination was discounted, in this case, as both the DTS and the TS drive the same signal.

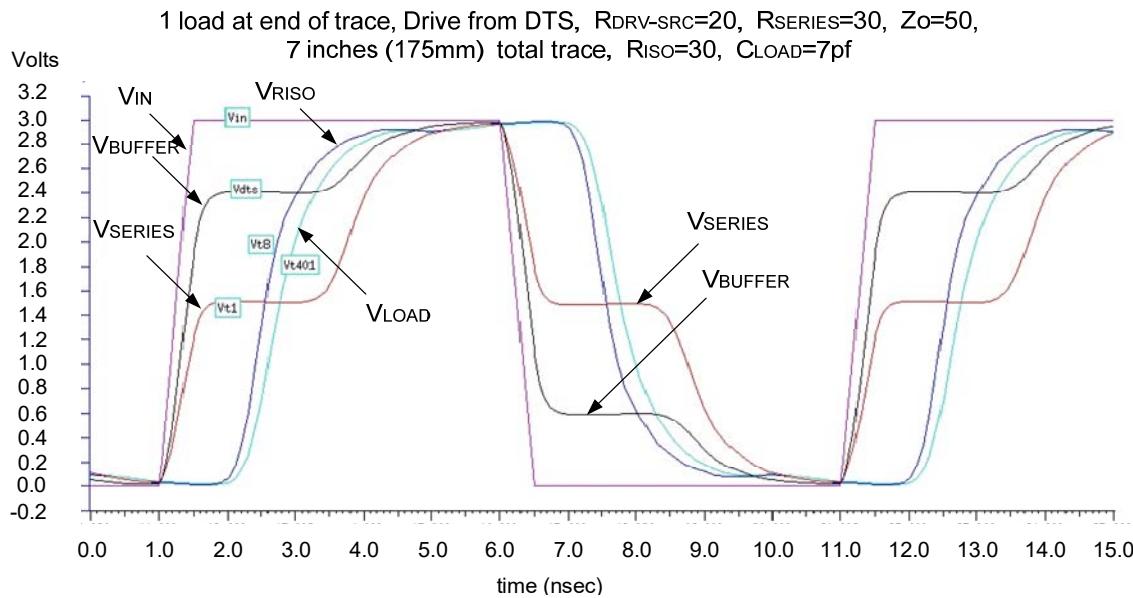
With the configuration shown in Figure F-41 the same model is used for both the DTS and the TAPC. Each has a bidirectional pin, and internal to the device is a driver with an inherent source resistance and a capacitor representing the load seen externally when the device is operating as a receiver as shown in Figure F-7.

### Point to Point/Bidirectional Signaling/Series Termination



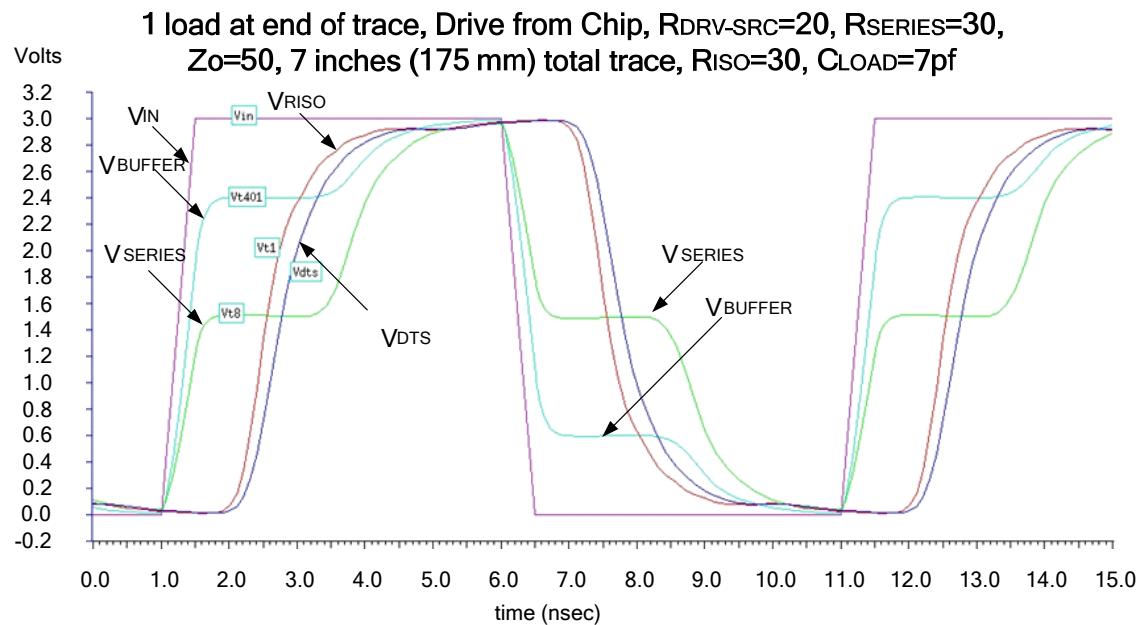
**Figure F-41 — Single source/destination/bidirectional signaling/series termination**

The symmetrical series termination scheme shown in Figure F-41 works very well. Figure F-42 shows the simulation results when the DTS is the source and the chip/TAPC is the destination. Figure F-43 shows the reverse—the chip/TAPC is the signal source and the DTS is the destination.



**Figure F-42 — Bidirectional scenario—DTS drives/chip/TAPC receives**

When the DTS is the driver/source, node  $Vt1$ , the connection between the DTS series termination resistor and the beginning of the PCB trace (transmission line) exhibits the classical midvoltage step, demonstrating that the DTS internal resistance plus the external series termination resistor matches the transmission-line impedance. In this case, the resistor at the far end is acting as an isolation resistor. When the drive direction is reversed, the simulation waveforms appear as shown in Figure F-43, and the roles of the two resistors is reversed—the resistor at the chip/TAPC becomes the series termination device, matching the chip output resistance to the transmission-line impedance, and the resistor at the DTS becomes an isolation resistor.



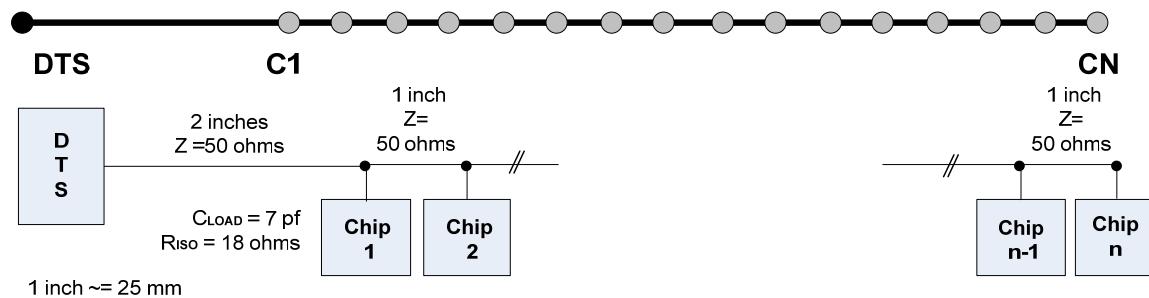
**Figure F-43 — Bidirectional scenario—Chip/TAPC drives, DTS receives**

Note that in the node  $V_{\text{SERIES}}$ , the connection of the chip-end termination resistor to the PCB trace (transmission line) now exhibits the characteristic midvoltage step, demonstrating that the drive resistance at the chip end matches the transmission-line impedance.

## F.9 Line configuration of a transmission line

### F.9.1 Model

The logical view of the Line configuration of a transmission line is shown in Figure F-44. It shows a 50 mm (~2 in) trace connecting the DTS to the first chip and chips spaced at 25 mm (~1 in) intervals thereafter. The C1-CN notation shown in this figure represents the load number (Chip 1–Chip N). This notation is used in subsequent figures generated by simulating a Line configuration with a series termination.

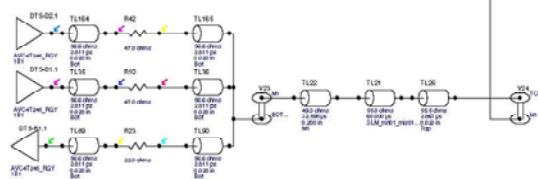


**Figure F-44 — Line configuration of a transmission line with series termination**

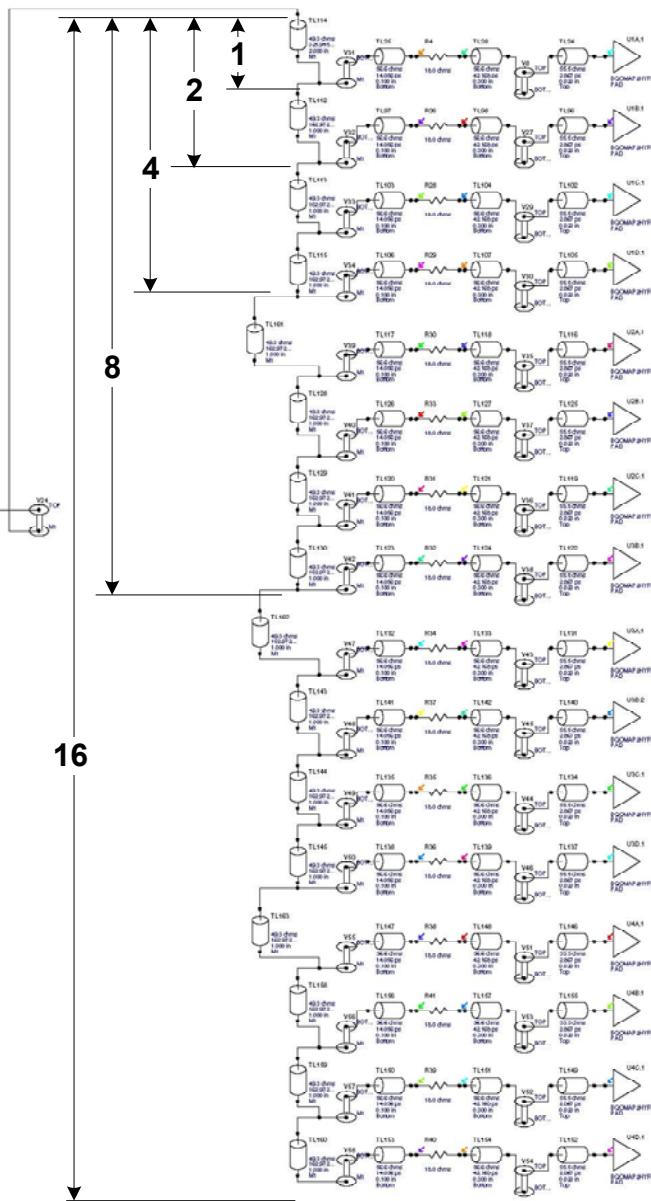
The model used to create simulation results for this configuration is shown in Figure F-45. This model is quite sophisticated as it includes connectors, small traces between the isolation resistors and the pin, and PCB vias between signal layers. Less sophisticated models may also be used. Within this figure, transmission line segments are roughly  $56 \Omega$  with  $47 \Omega$  resistors in series with the two drivers operating in parallel. Feed-throughs are roughly  $50 \Omega$ .

## Sequence of transmission line segments

## Parallel drivers



Note: this drawing is intended to convey the level of detail included in the transmission line modeling, not detailed modeling values.



**Figure F-45 — Model for a line transmission line with 1 to 16 TAPs**

The following two additional effects come into play in the multidrop scenario:

- Additional capacitive loading on the output driver
  - Disruption of impedance matching at intermediate points where TAPs connections are made

The waveform begins to have some of the characteristics shown in Figure F-13 when the source impedance is  $50 \Omega$ . In this case, the source impedance is too high to provide the current needed when the discontinuities are reached. A bit of iterative simulation determines that a more optimal value for the total source resistance is  $35 \Omega$  to  $40 \Omega$  instead of  $50 \Omega$ . Note that the DTS is modeled with two drivers operating in parallel to lower the source impedance. The models of input and output buffers are those provided by SI vendors.

## F.9.2 Unidirectional signaling

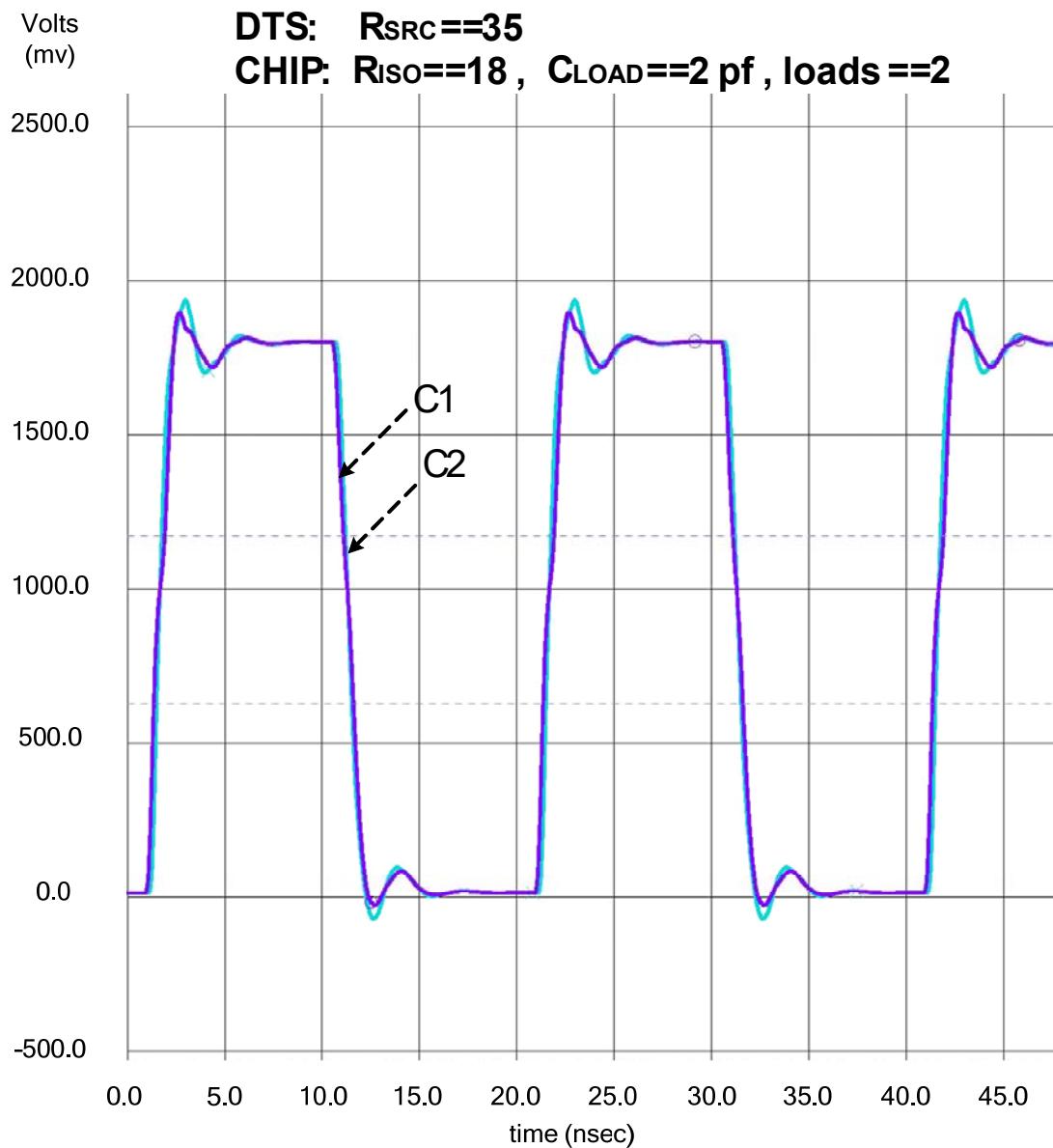
The single-source/multiple-destination connectivity scenario is an extension of the Point-to-Point connectivity scenario. Instead of a single receiver, there are multiple receivers or multiple TAPCs being driven by the DTS. The signals that fall into this category include the TCK(C) and TMS(C) in a Series Scan Topology, and TDI (in a Star-4 configuration).

With a 50 MHz 1.8 V signal, the effects of the signal length and number of loads are shown in the following figures:

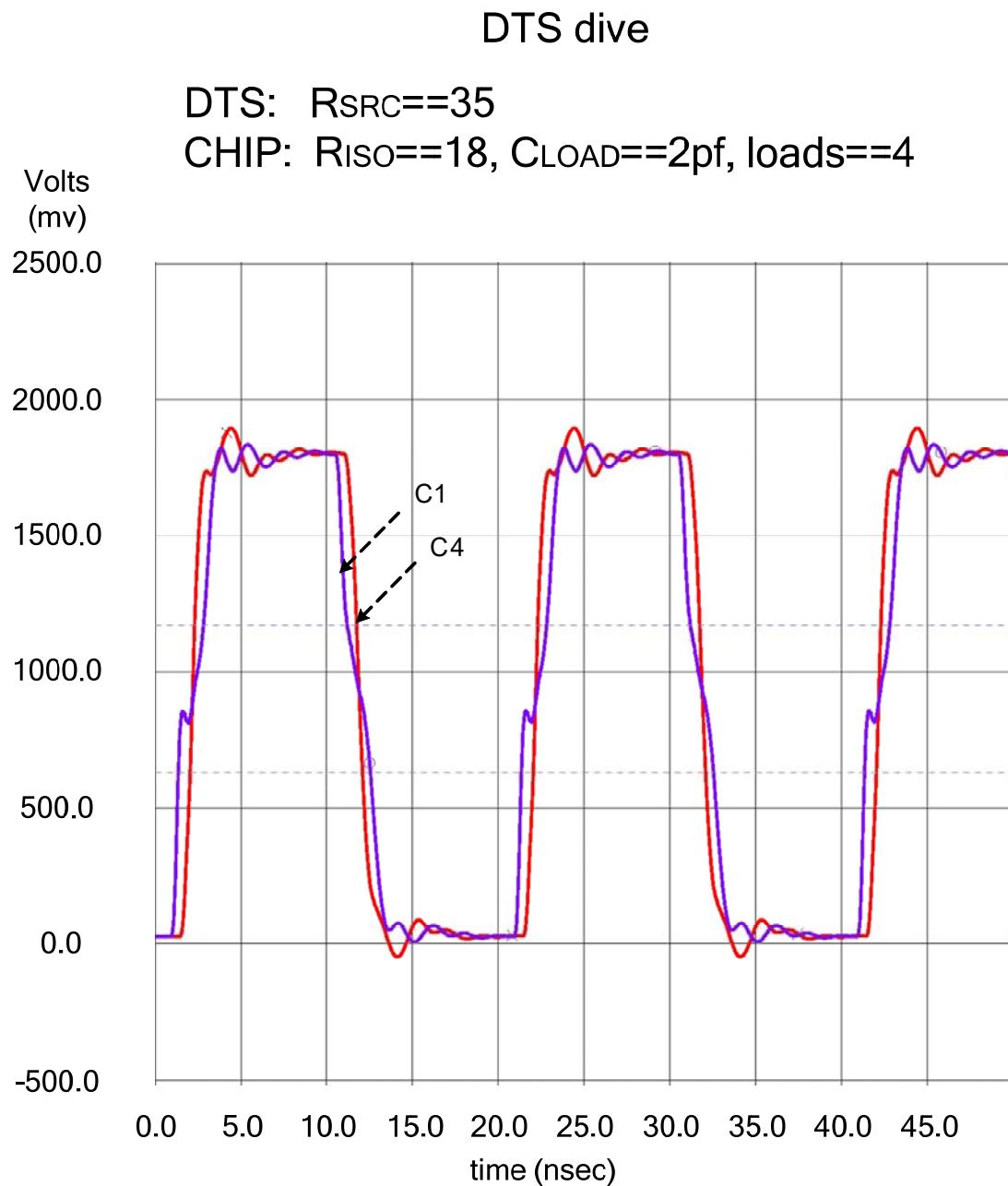
- Figure F-46 For two loads spaced at 25 mm (~1 in) intervals
- Figure F-47 For four loads spaced at 25 mm (~1 in) intervals
- Figure F-48 For eight loads spaced at 25 mm (~1 in) intervals
- Figure F-49 For 16 loads spaced at 25 mm (~1 in) intervals

The notation C1–CN in these figures refers to the signal at the terminal for Chip 1–Chip N.

## DTS drive



**Figure F-46 — Line configuration, two loads—25 mm intervals, 50 mm DTC trace**



**Figure F-47 — Line configuration, four loads—25 mm intervals, 50 mm DTC trace**

## DTS drive

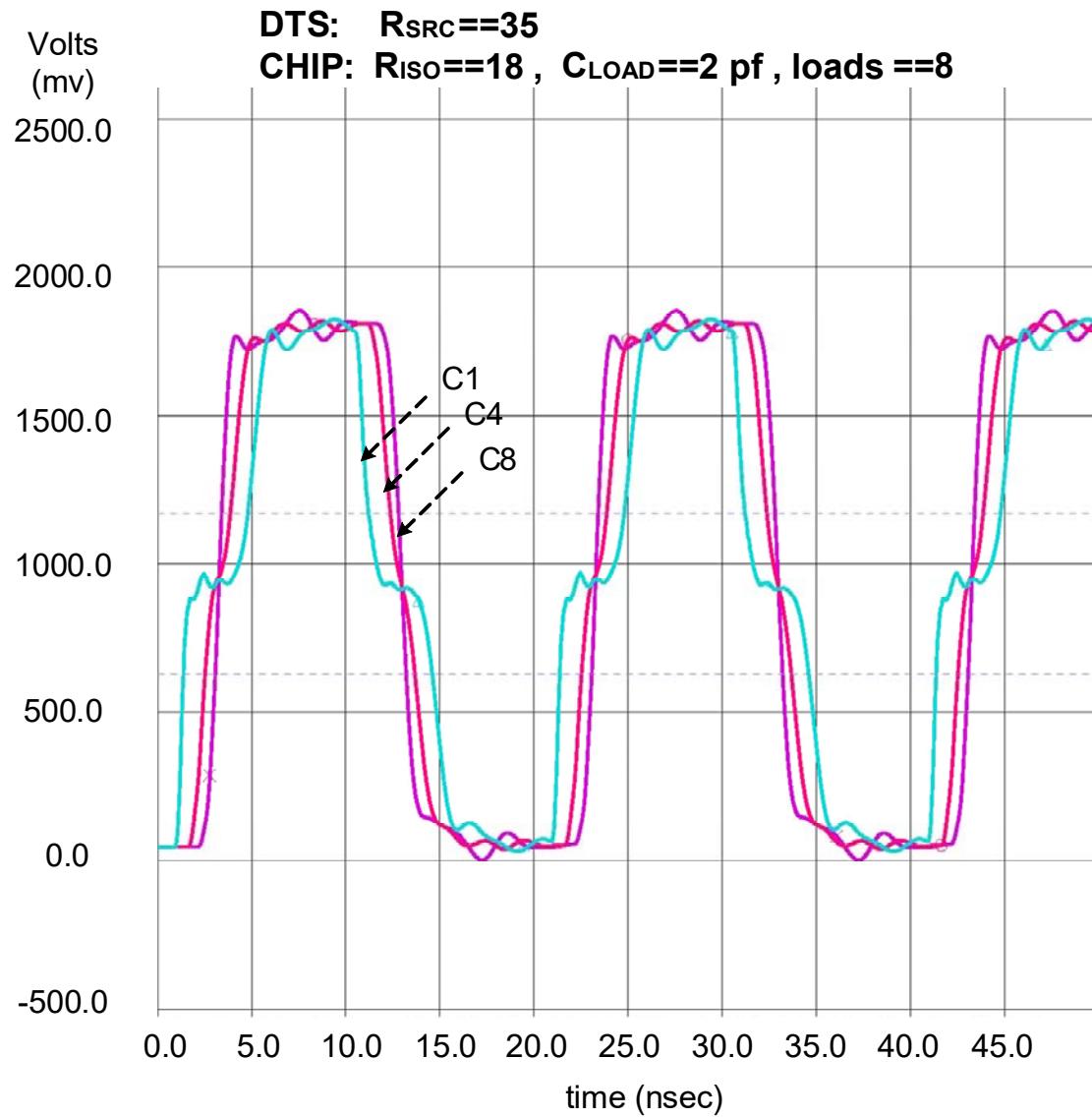
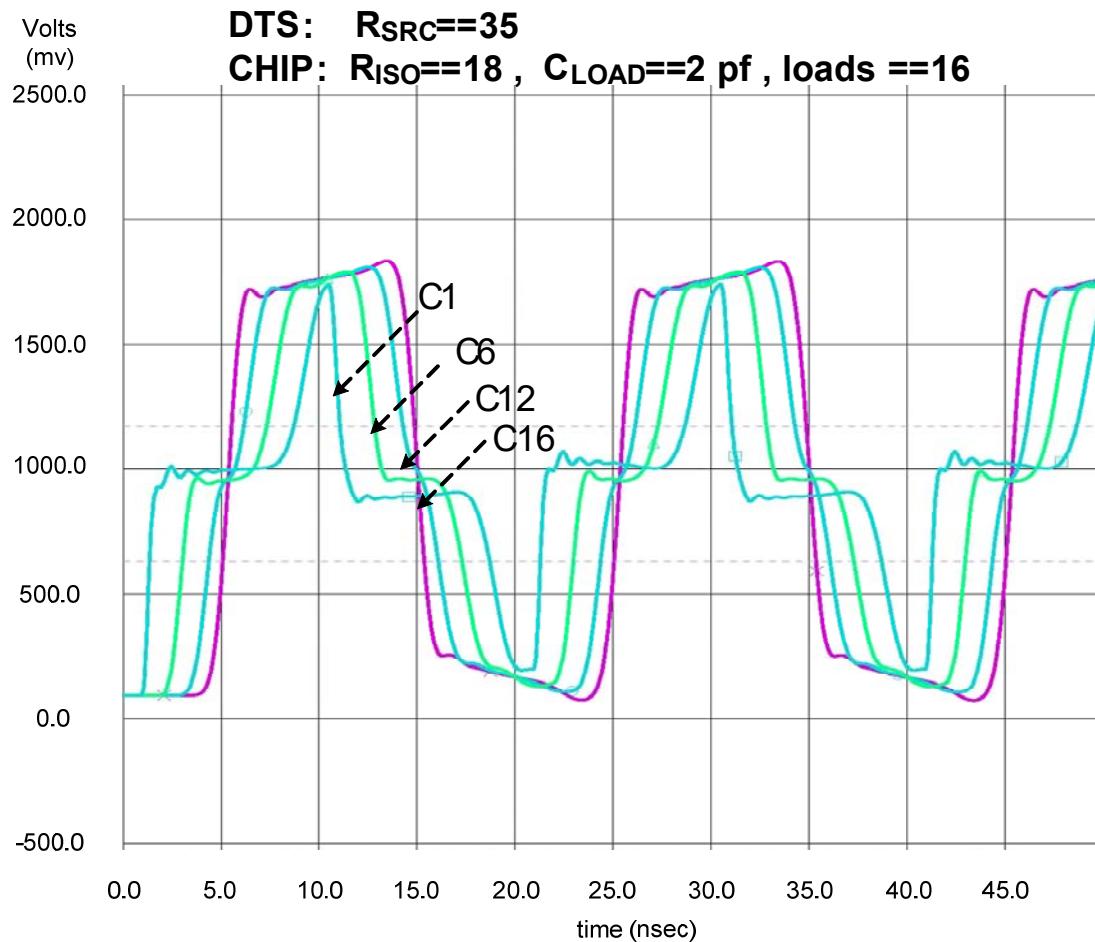


Figure F-48 — Line configuration, eight loads—25 mm intervals, 50 mm DTC trace

## DTS drive



**Figure F-49 — Line configuration, 16 loads—25 mm intervals, 50 mm DTC trace**

The undesirable aspects of the Line configuration with a series termination are highlighted in Figure F-49. The effects of the length of the transmission line are clearly seen when 2 times the length has reached 760 mm (~30 in) in length. Note the following:

- Shape of the signal at chip one showing a half level for nearly 6 ns
- Skew between signal edges at the various chips becomes quite large
- Point in time that the loads see the signal edges is substantial

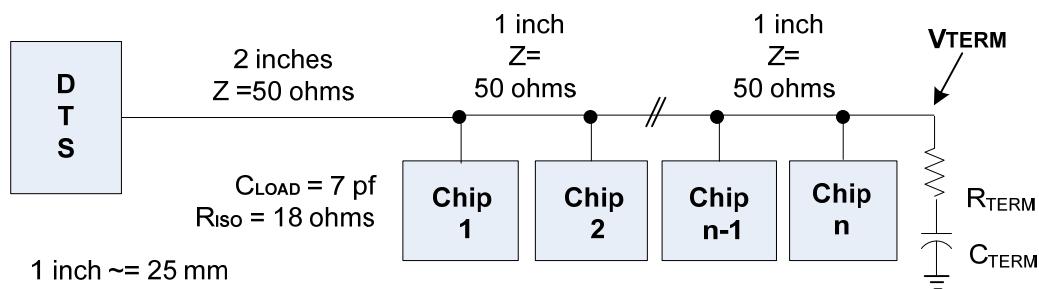
The skew in edge detection can be problematic when falling-edge timing is used. This signaling characteristic may require the use of rising-edge sampled signaling (SREDGE == a logic 1) when the round trip delay reaches 2 ns to ensure the input setup and hold-time requirements outlined in Annex E are met. The uncertainty of when the input detects a transition aggravates this problem.

Note that the voltage waveform at chip n (farthest load from the driver and at the end of the PCB trace) is monotonic and rail to rail. If the total system is tolerant of the associated delay, this may be satisfactory. In contrast, the voltage waveform at chip one (the first load, closest to the driver) includes the characteristic voltage step part-way through the rise and fall times. This is an inherent, unavoidable characteristic of series termination and was discussed earlier in this annex. Unless the receiving logic at chip one is tolerant

of this (refer to the earlier subclauses on input signal conditioning in this annex), the circuit may not function correctly. Specifically, if the switching point of the input logic of the receiver at chip one is near the voltage level of these steps, the circuit will malfunction. Keep in mind that these voltage levels and the input switching level of the node chip one receiver will vary with temperature and voltage, and sooner or later, the switching point and the waveform anomaly will align. If signal conditioning countermeasures in this document are not put in place, then the circuit will malfunction when these levels are aligned.

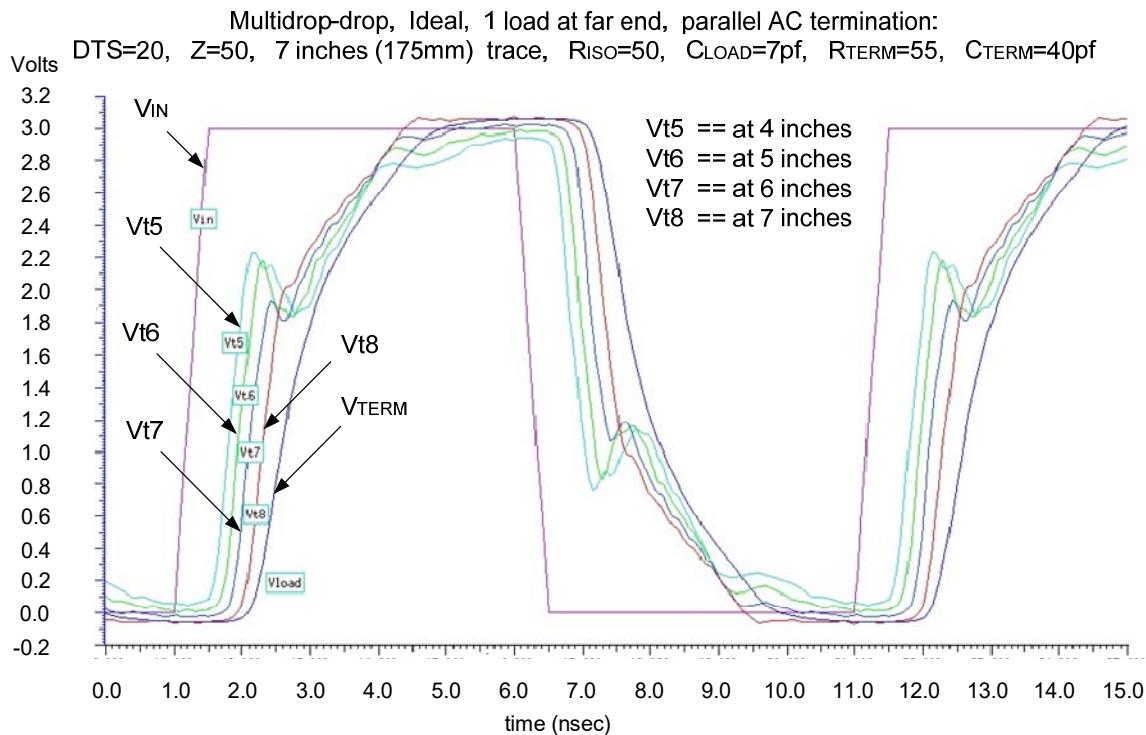
### F.9.2.1 Parallel AC termination

A Line configuration of a transmission line with multiple connected receivers and a parallel AC termination is illustrated in Figure F-50. This figure shows a 50 mm (~2 in) trace connecting the DTS to the first chip and chips spaced at 25 mm (~1 in) intervals thereafter.

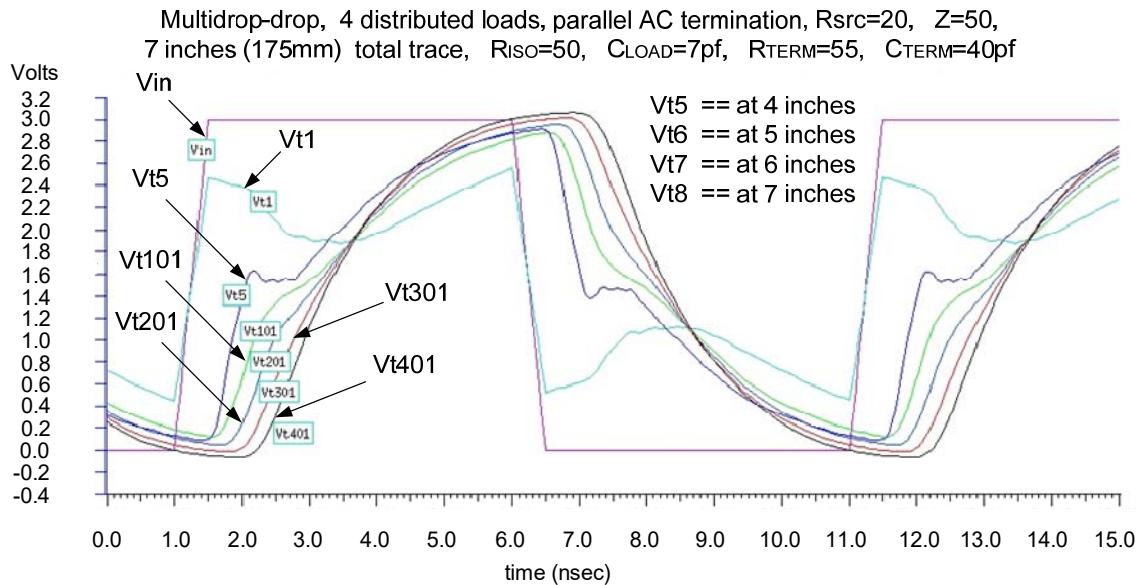


**Figure F-50 — Multiple TAP connection to a Line configuration**

The simulation showing only chip[n] where it is 175 mm (~7 in) from the source is shown in Figure F-51. The simulation results for four loads are shown in Figure F-52. Note that while signal quality issues exist at various points along the transmission line, it is probably acceptable at the far end, where the one load that is present in this model connects to the PCB trace. Also note the source impedance is  $20 \Omega$ , which is quite low. This makes parallel AC termination less attractive.



**Figure F-51 — Parallel AC termination, one load at end of transmission line**



**Figure F-52 — Parallel AC termination, four loads at 25 mm intervals**

Simulation iterations were done to determine the optimal value for the isolation resistor, R<sub>ISO</sub> ( $50\ \Omega$ ) and the termination resistor and capacitor ( $55\ \Omega/40\ pF$ ). Note that the signals at the loads are monotonic but include rise/fall timing and delay values that may be a point of concern. Rise/fall timing is affected by two things—the characteristics of the signal at the load's tap point on the transmission line (before the isolation resistor) and the RC circuit formed by the load's isolation resistor and input capacitance. The signal labeled Vt5 on the plot is the tap point (100 mm from the DTS) on the transmission line for the first load, and the

load-side of the corresponding isolation resistor is labeled Vt101. Comparing these two signals demonstrates the low-pass filtering effect of the RC network formed by the isolation resistor and the load capacitance. Note the midvoltage flat spot, which is characteristic of series source termination. This is a function of the interaction between the source impedance (inherent to the DTS), the segment of the transmission line that feeds the first load, and the apparent load at that point. The width of the flat period is a function of the propagation time of the driven signal to the far end of the transmission line and back. The slow rise time after the flat period is a function of the capacitive load encountered at the far end of the transmission line.

### F.9.2.2 Parallel termination

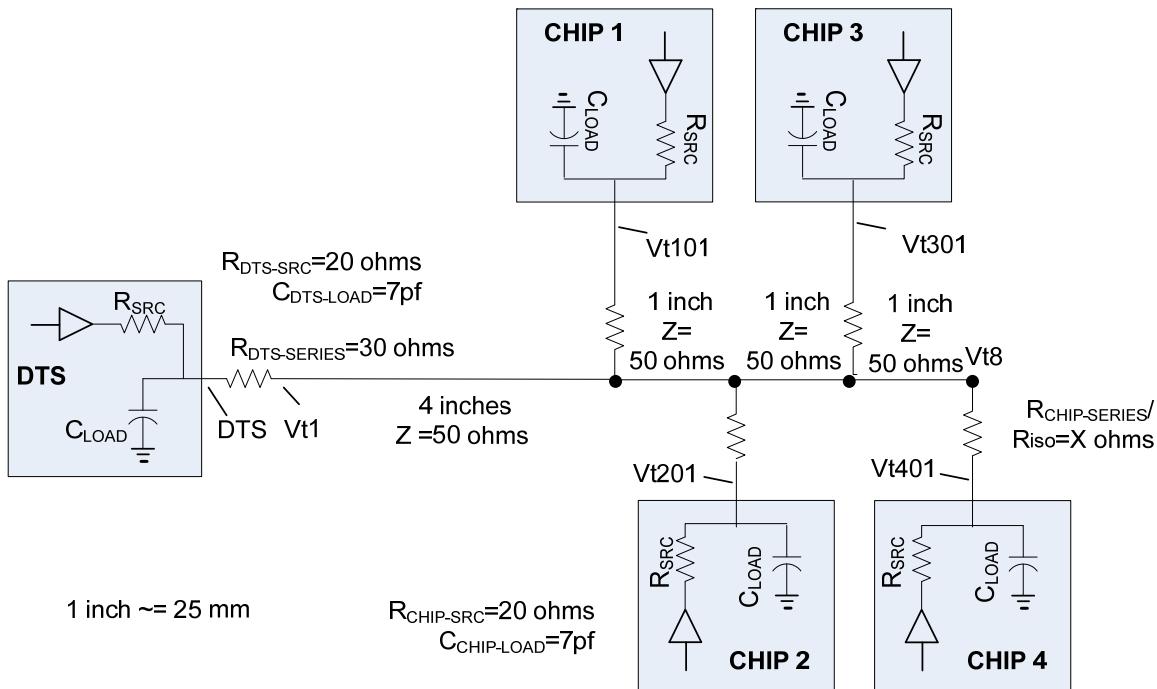
The characteristics of a parallel termination are shown in Figure F-22. For a properly implemented transmission line with output edge-rate control, the signal is full level at all nodes with a time delay as a node is farther from the source.

### F.9.2.3 Summary

The series or parallel termination scheme is recommended even though the series and the parallel AC termination schemes work acceptably with the Line configuration with four or less loads. The series termination scheme has the advantage of including one less component and being compatible with the series termination schemes used with T, X, and XT configurations and requires adequate care be taken to create signal integrity. The parallel DC termination scheme is not considered viable because of the voltage division that occurs between the driving source internal resistance and the end-of-line DC-connected termination resistor. This may prevent rail-to-rail voltage swings at the load. In addition, this termination scheme consumes a significant amount of DC current.

## F.9.3 Bidirectional signaling

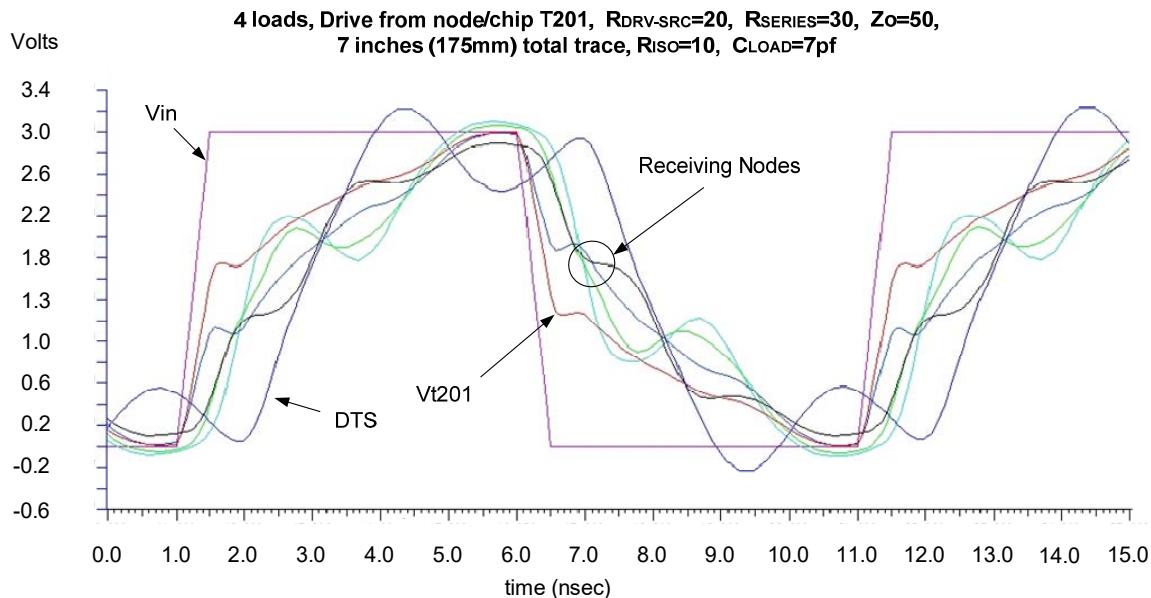
The system model shown in Figure F-53 extends the bidirectional, single-load connectivity case to a multi-load connectivity scenario. The models for the DTS and Chip are expanded to provide better clarity in comprehending the observation points. This connectivity scenario is an extension of the one just described. It is number four in the list of connectivity scenarios that ranges from the simplest to the most complex. It consists of bidirectional signaling with one or more drivers. Instead of a single receiver, there are multiple receivers or multiple TAPCs being driven by the DTS or Chip(s). The TMS(C) signal falls into this category when there is more than one driver when using the Advanced Protocol.



**Figure F-53 — Bidirectional scenario—chip/TAPC drives, DTS receives, multiple TAPCs**

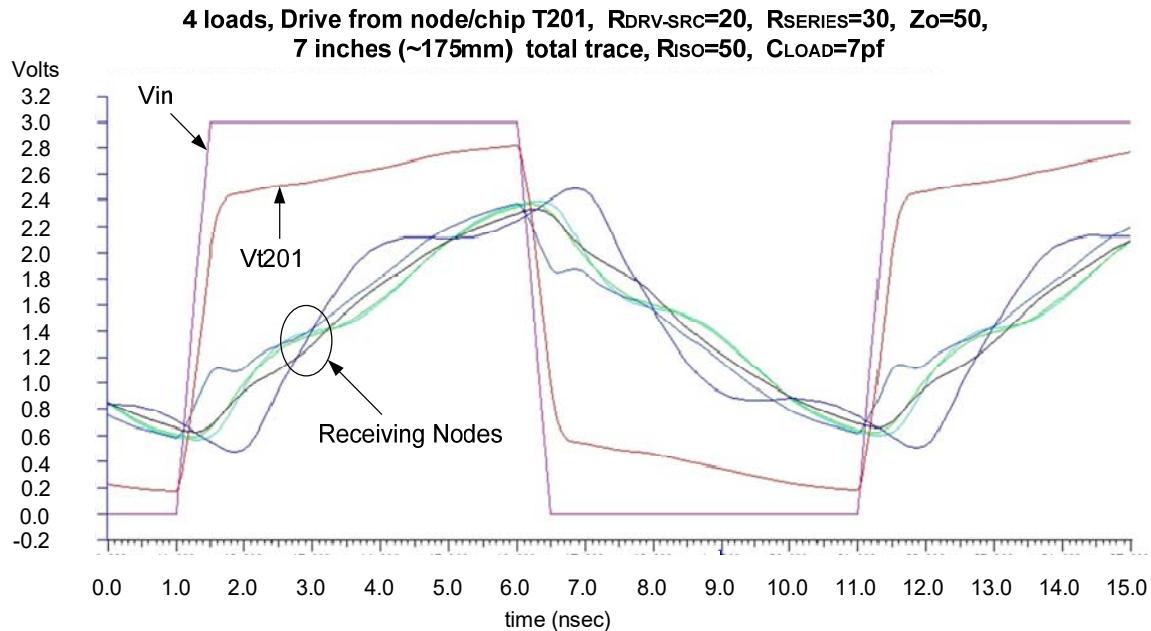
This scenario poses additional challenges as in some cases chips driving the transmission line see two  $50 \Omega$  transmission lines in parallel. This means reflections increase depending on the source or sources driving the transmission line. The simulation data highlights the need to change the termination strategy and lower the operating frequency as needed.

The simulation data for this scenario at 100 MHz with  $R_{ISO} = 10 \Omega$ , using a relatively simple model, is shown in Figure F-54. The driving node is  $Vt201$  (second chip).  $Vt101$ ,  $Vt201$ ,  $Vt301$ , and  $Vt401$  are the chip/TAPC receiver pins (after the low-pass filter formed by the  $R_{ISO}$ ). Node DTS is the input signal at the DTS. Note the time displacement between signals at each node.



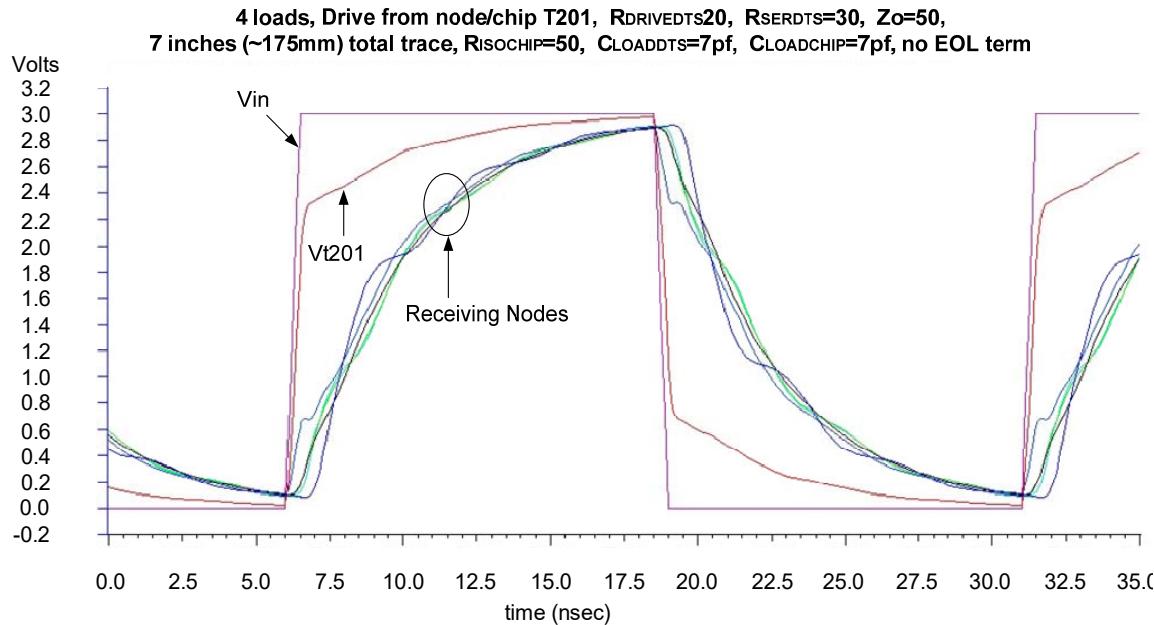
**Figure F-54 — Bidirectional multiple TAPCs, 100 MHz,  $R_{ISO} = 10 \Omega$**

The simulation data for this scenario at 100 MHz and  $R_{ISO} = 50 \Omega$  is shown in Figure F-55. The driving node is C2 (Chip 2). Other nodes are the chip/TAPC receiver pins (after the low-pass filter formed by the  $R_{ISO}$ ). Node DTS is the DTS input signal at the DTS. Increasing the value all the isolation resistors to a value from  $30 \Omega$  to  $50 \Omega$  significantly slows the signal edges down when viewed on the PCB trace. Note that the signal reaches neither a full logic 1 level nor a full logic 0 level. This is problematic when the signal is not 50% of the duty cycle as DC offsets build up depending on the data pattern. This has the effect of modulating the signal with a lower frequency signal and induces duty-cycle modulation. The TCK(C) frequency is therefore lowered for the system to operate reliably. Further simulation reveals an optimal value of  $R_{ISO}$  is between  $15 \Omega$  and  $24 \Omega$ .



**Figure F-55 — Bidirectional multiple TAPCs, 100 MHz,  $R_{ISO} = 50 \Omega$**

Simulation data for this scenario at 40 MHz and isolation resistor =  $50 \Omega$  is shown in Figure F-56. The physical configuration used with the simulation shown in Figure F-50 is also used in this case. Chip 2 (node T201) drives but with a 25 ns period (40 MHz) instead of a 10 ns period (100 MHz). This is the same model—four chips/TAPCs on a total PCB trace length of 175 mm. The signal now swings to a full level and is therefore not subject to DC offsets created by the data pattern.



**Figure F-56 — Line configuration, bidirectional multiple TAPCs, 40 MHz,  $R_{iso} = 50 \Omega$**

Note that a combination of an isolation resistor that creates source impedance that is higher than the impedance of two transmission lines in parallel together with lower operating frequencies produce well-behaved signals for the Line configuration.

### F.9.3.1 Four loads

Simulations using the more sophisticated model shown in Figure F-45 were performed with the following:

- Four chips
- Freq = 50 MHz
- 125 mm (~5 in) total trace

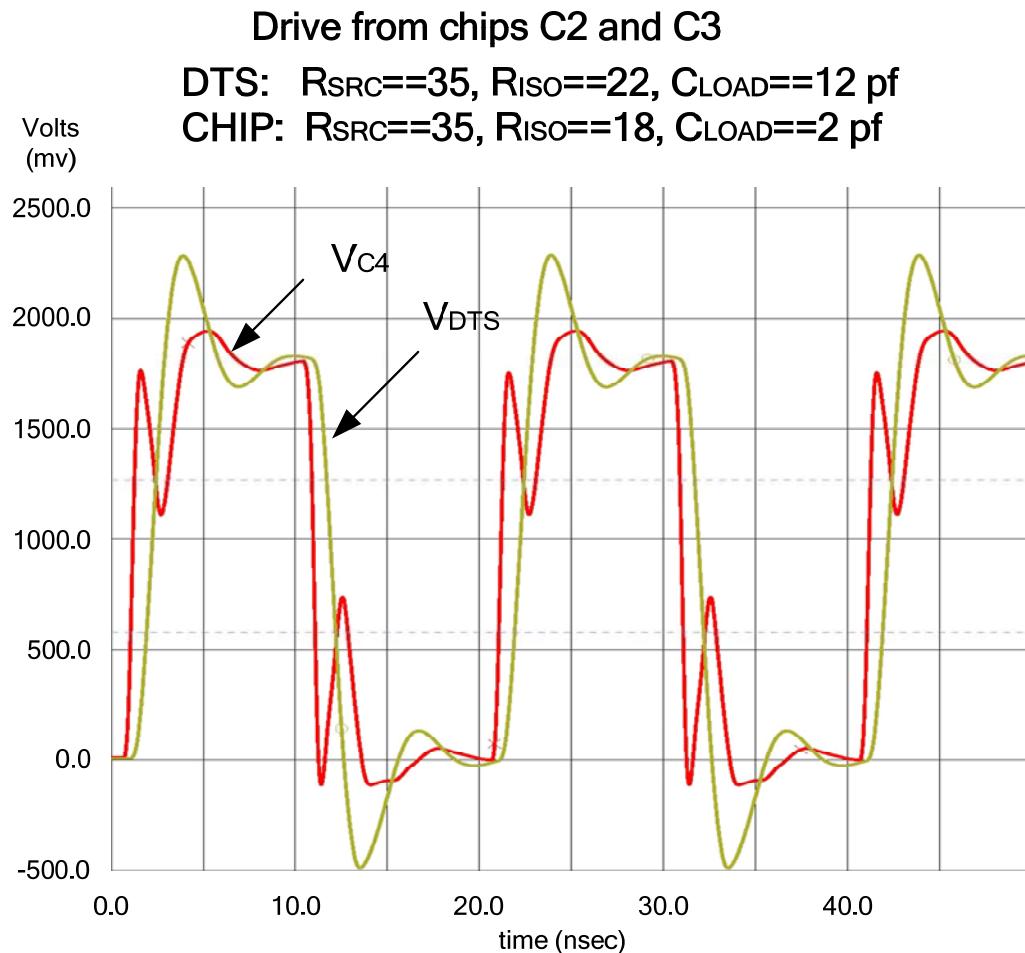
These simulations are shown in the following figures:

- Figure F-57 C2 and C3 drive simultaneously
- Figure F-58 C2 and C4 drive simultaneously
- Figure F-59 C1, C2, C3, and C4 drive simultaneously

These waveforms indicate that the connection may operate in excess of 80 MHz. They show cases where more than one TAPC is driving the TMSC signal at the same time. They provide the following insights:

- Keeping the distance between chips to 25 mm (~1 in) or less has a smoothing effect as some of the load looks lumped

- The effects of reflections settle after about 3 ns with this spacing of loads
- Higher  $R_{ISO}$  values (15–22 ohms) smooth edges adequately for inputs
- The more TAPs driving the signal, the larger the signal overshoot at the DTS



**Figure F-57 — Line configuration, 4 TAPs, C2 and C3 drive, 50 MHz**

### Drive from chips C2 and C4

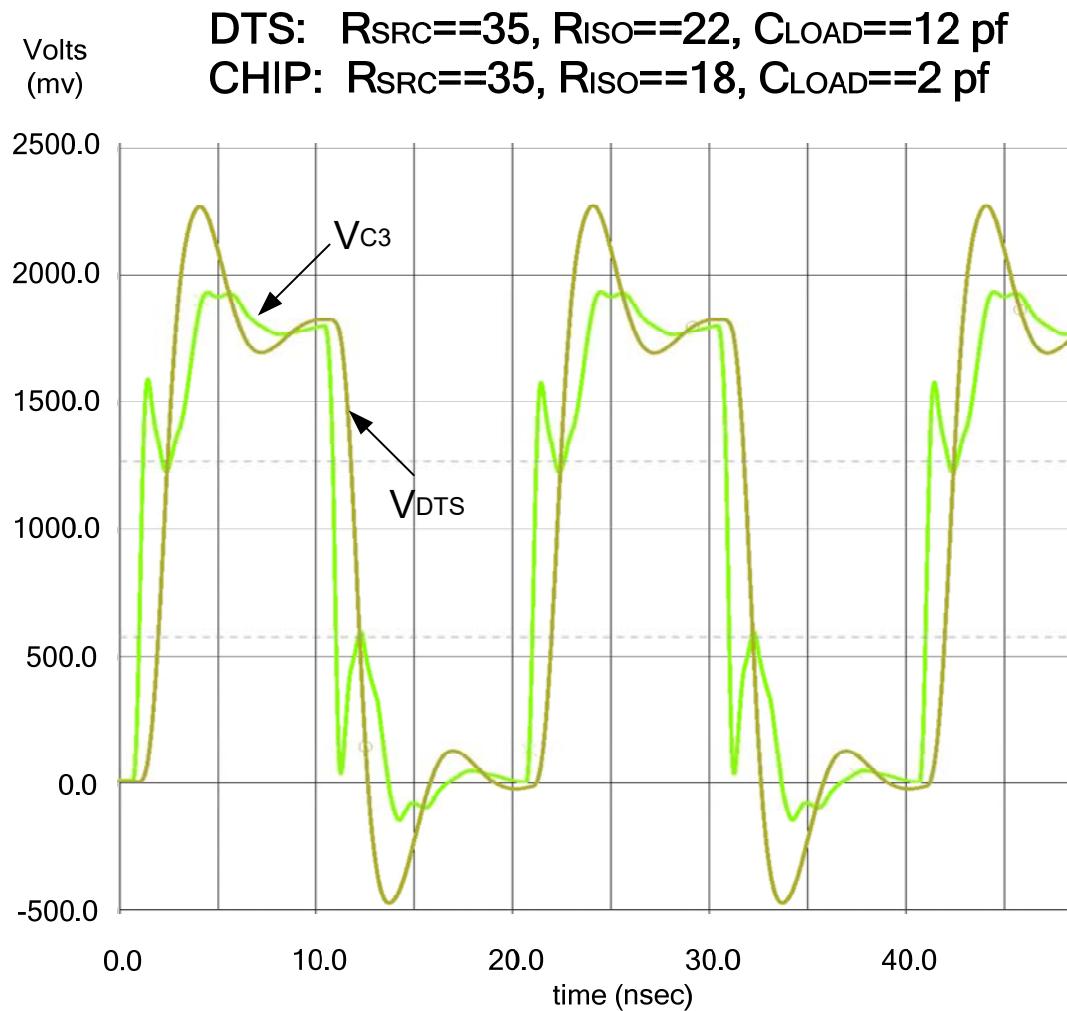


Figure F-58 — Line configuration 4 TAPs, C2 and C4 drive, 50 MHz

## Drive from chips C1, C2, C3, and C4

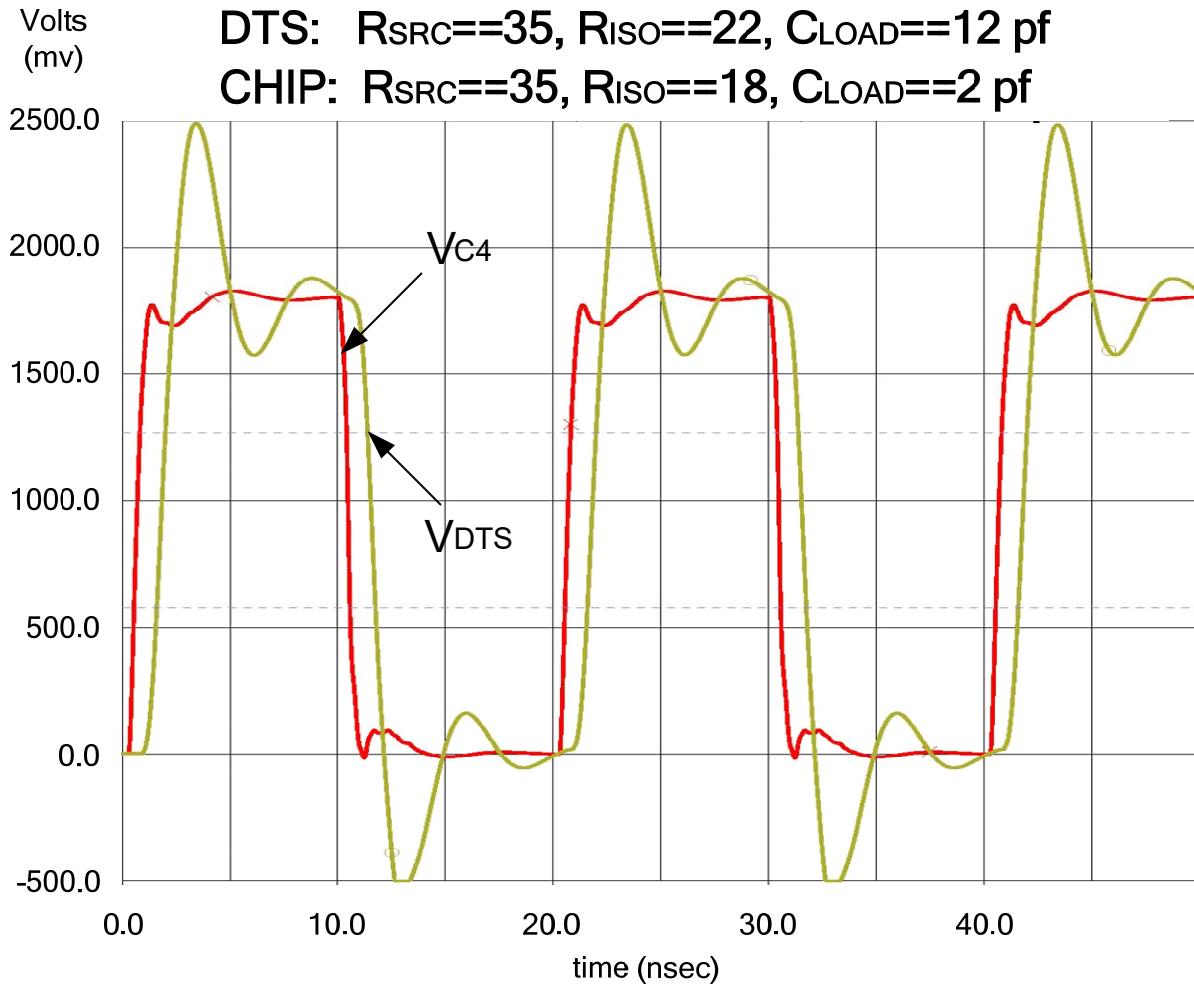


Figure F-59 — Line configuration, four TAPs, four drives, 50 MHz

### F.9.3.2 Eight loads

Simulations using the more sophisticated model shown in Figure F-45 performed with the following:

- Eight chips
- Freq = 50 MHz
- 225 mm (~9 in) total trace

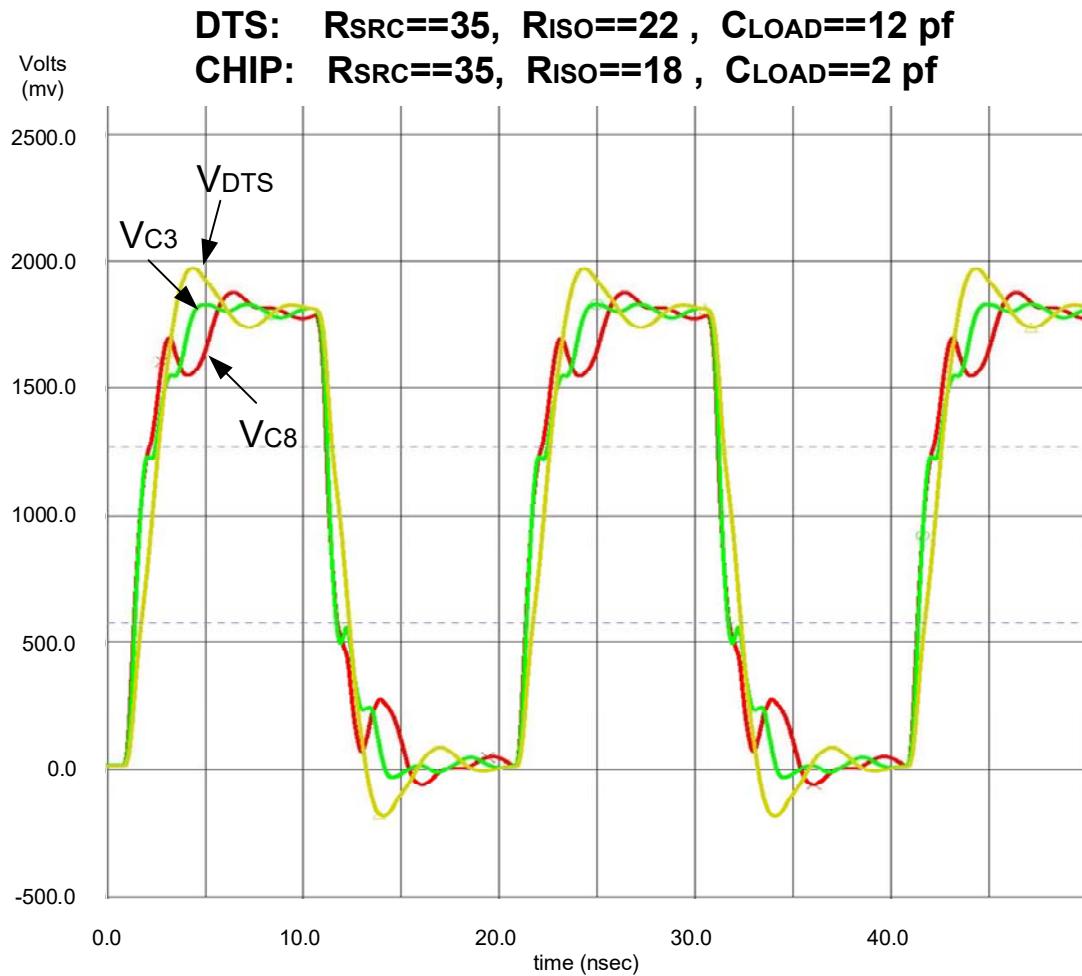
These simulations show cases where more than one TAPC is driving the TMSC signal at the same time and are shown in:

- Figure F-60 C1 and C6 drive simultaneously

- Figure F-61 C2 and C4 drive simultaneously
- Figure F-62 C2 and C6 drive simultaneously

The waveforms show the length of the line becomes a factor in addition to providing the same insights as with four TAPs. Note that these simulations do not consider the possibility of asymmetric TCKC to TMSC delays.

### Drive from chip C1 and C6



**Figure F-60 — Line configuration: 8 TAPs, two drives (C1 and C6 drive), 50 MHz**

## Drive from chip C2 and C4

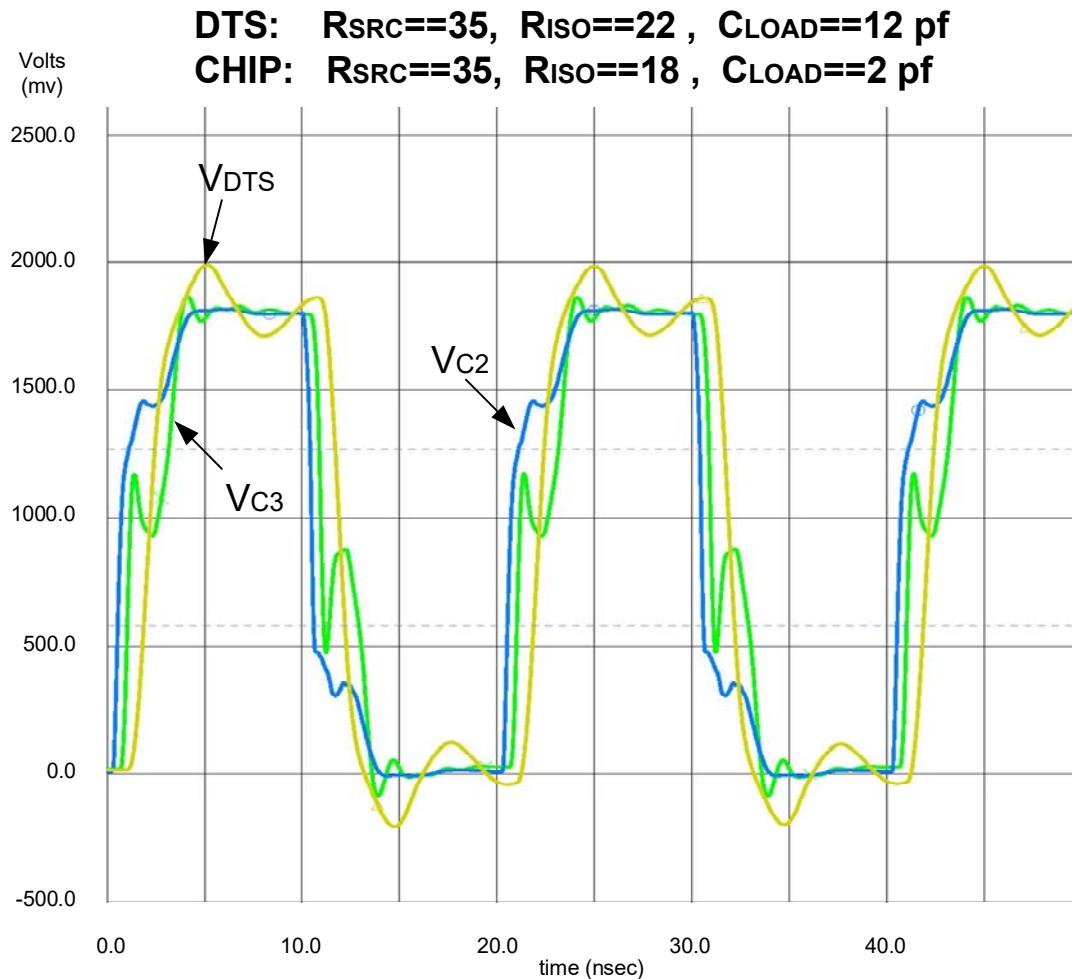
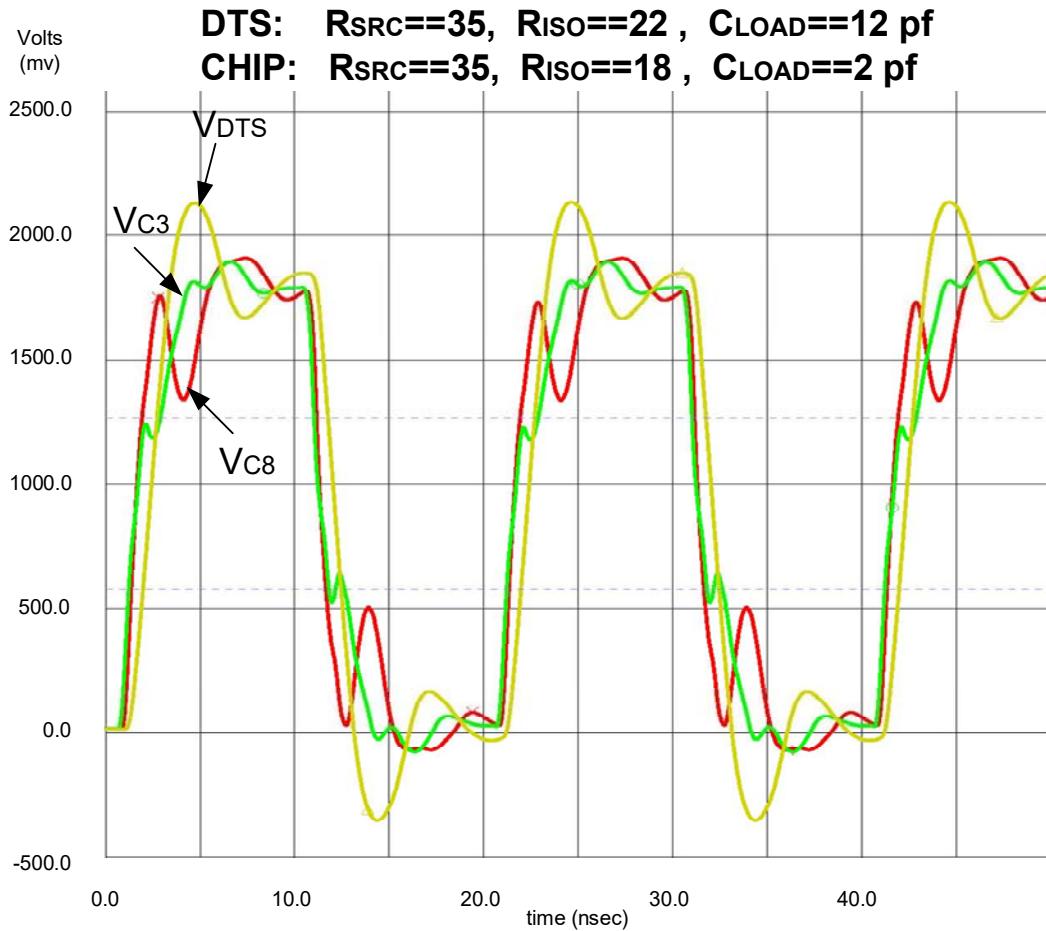


Figure F-61 — Line configuration: 8 TAPs, two drives (C2 and C4), 50 MHz

## Drive from chip C2 and C6

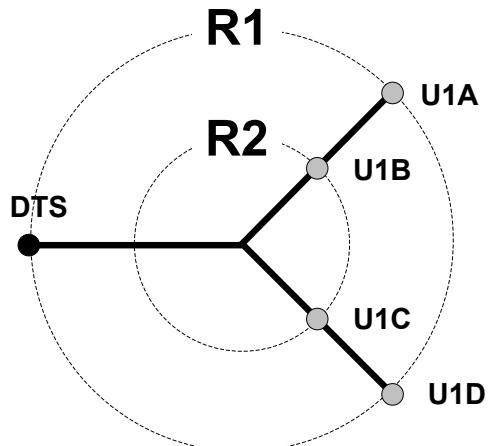


**Figure F-62 — Line configuration: 8 TAPs, two drives (C2 and C6), 50 MHz**

## F.10 T configuration of a transmission line

### F.10.1 Model

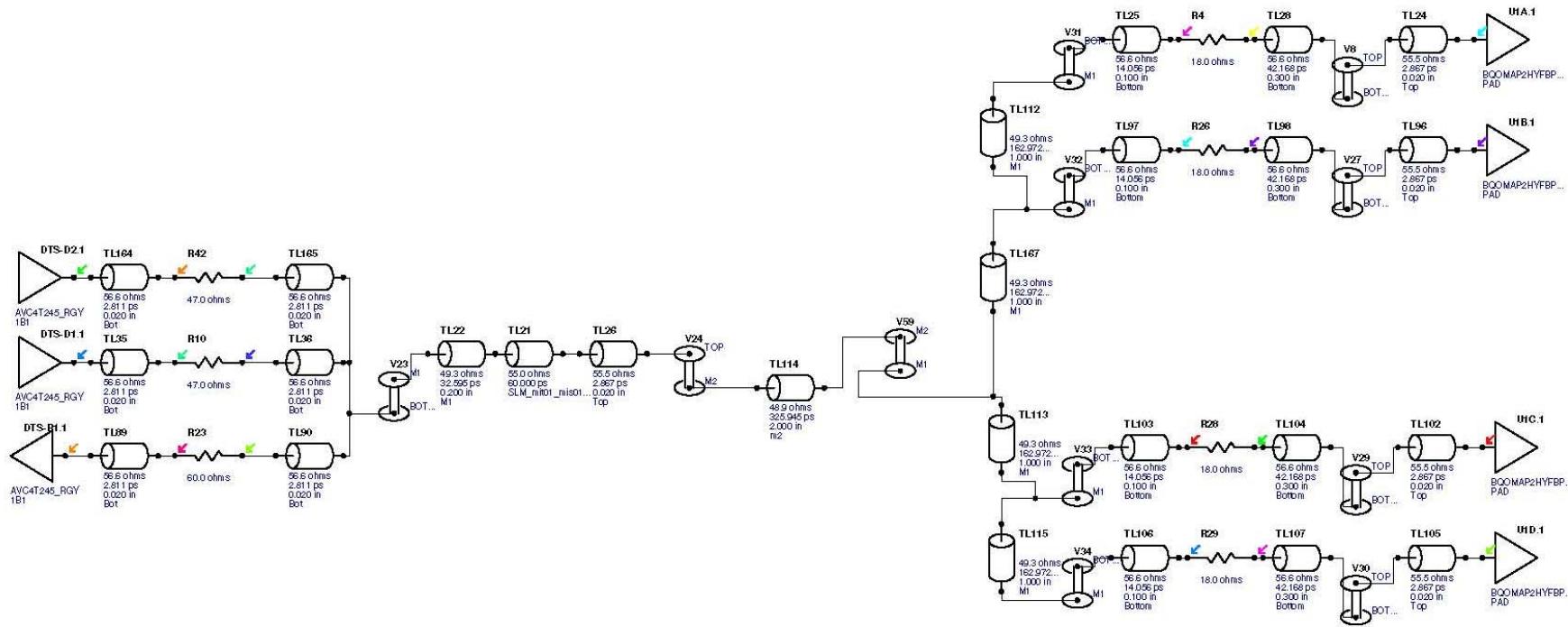
The logical view of the T configuration of a transmission line is shown in Figure F-63. It shows the DTS and TAPs connected at two distinct distances from a central point. The values for R1 and R2 used in the simulations for the T configuration are 25mm (~1 in) and 50 mm (~2 in), respectively.



## T Configuration (2 to 4 TAPs)

**Figure F-63 — Logical view of T configuration of a transmission line**

The model used for the T configuration of a transmission line is shown in Figure F-64. This figure shows all nodes as inputs. The direction of the node buffers are changed for cases when a node drives. Within this figure, the transmission-line segments are roughly  $56\ \Omega$  with  $47\ \Omega$  resistors in series with the two drivers operating in parallel. Feed-throughs are roughly  $50\ \Omega$ .



NOTE—This drawing is intended to convey the level of detail included in the transmission line modeling, not detailed modeling values.

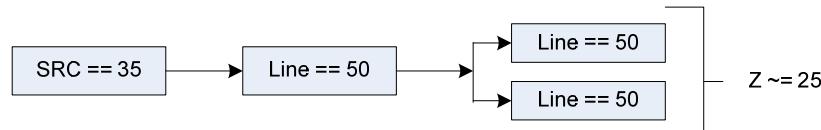
**Figure F-64 — T model with parallel DTS drivers and two nodes/leg**

## F.10.2 Unidirectional signaling

Simulations using the model shown in Figure F-64 performed with:

- Four chips
- Freq = 50 MHz
- 100 mm (~4 in) total trace ( $R_1 = 50$  mm (~2 in)), ( $R_2 = 25$  mm (~1 in))

The impedance profile of this configuration is shown in Figure F-65. With the source impedance at  $35 \Omega$ , the transmission line before the split is the conduit for the extra current needed to raise the voltage when the line splits.

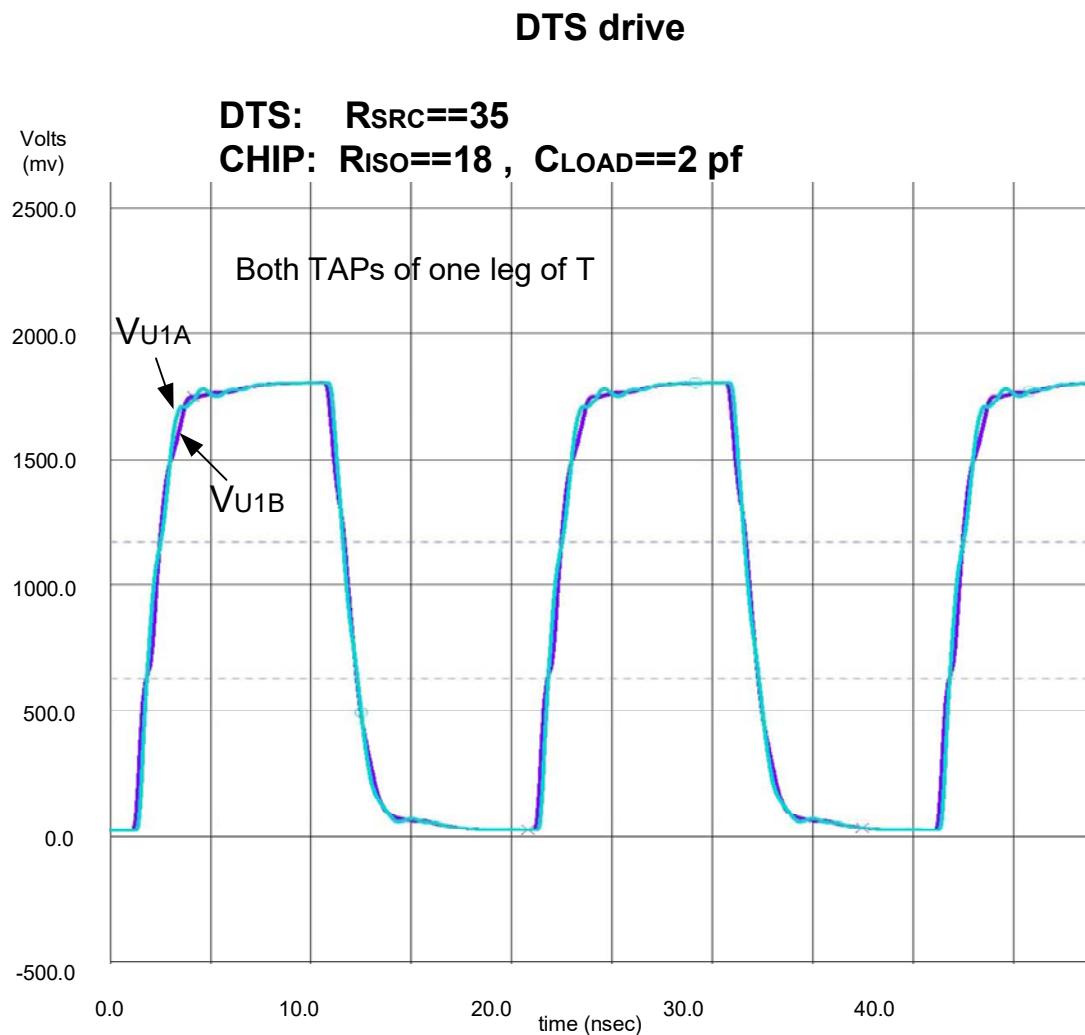


**Figure F-65 — Impedance profile of the T configuration**

Figure F-66 shows the U1A and U1B nodes of one leg of the T configuration shown in Figure F-63.

This waveform illustrates where more than one TAPC is driving the TMSC signal at the same time. They indicate the following:

- Keeping the distance between chips to 25 mm (~1 in) or less has a smoothing effect as some of the load looks lumped.
- The effects of reflections settle after about 3 ns with this spacing of loads.
- Higher  $R_{ISO}$  values ( $15 \Omega$  to  $22 \Omega$ ) smooth edges adequately for inputs.
- The more TAPs driving the signal, the larger the signal overshoot at the DTS.



**Figure F-66 — T configuration: 4 TAPs, DTS drive, 50 MHz**

### F.10.3 Bidirectional signaling

Additional simulations for bidirectional signaling are shown in the following figures:

- Figure F-67 U1A drives
- Figure F-68 U1B drives
- Figure F-69 U1A and U1B drive simultaneously
- Figure F-70 U1B and U1D drive simultaneously

These simulations use the model shown in Figure F-64 at 50 MHz with a T configuration with four chips.

### U1A drives

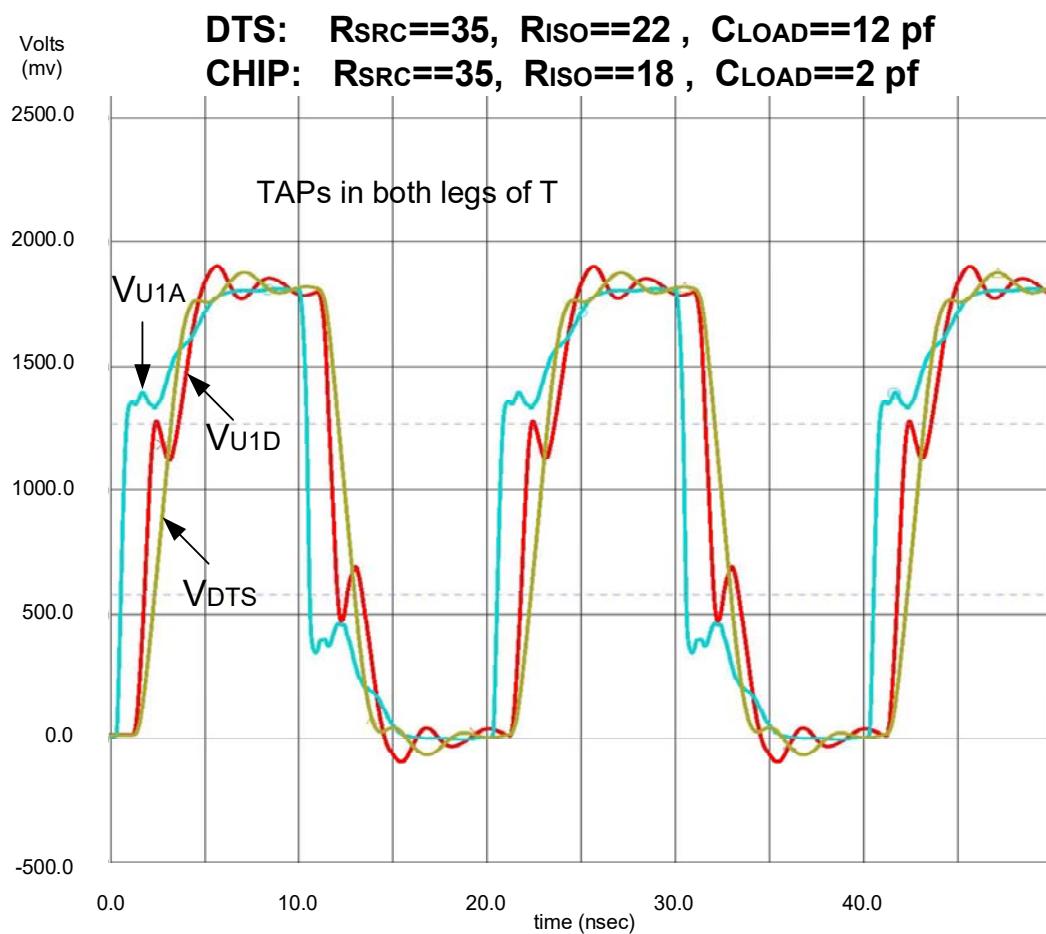


Figure F-67 — T configuration, 4 TAPs, U1A drives, 50 MHz

## U1B drives

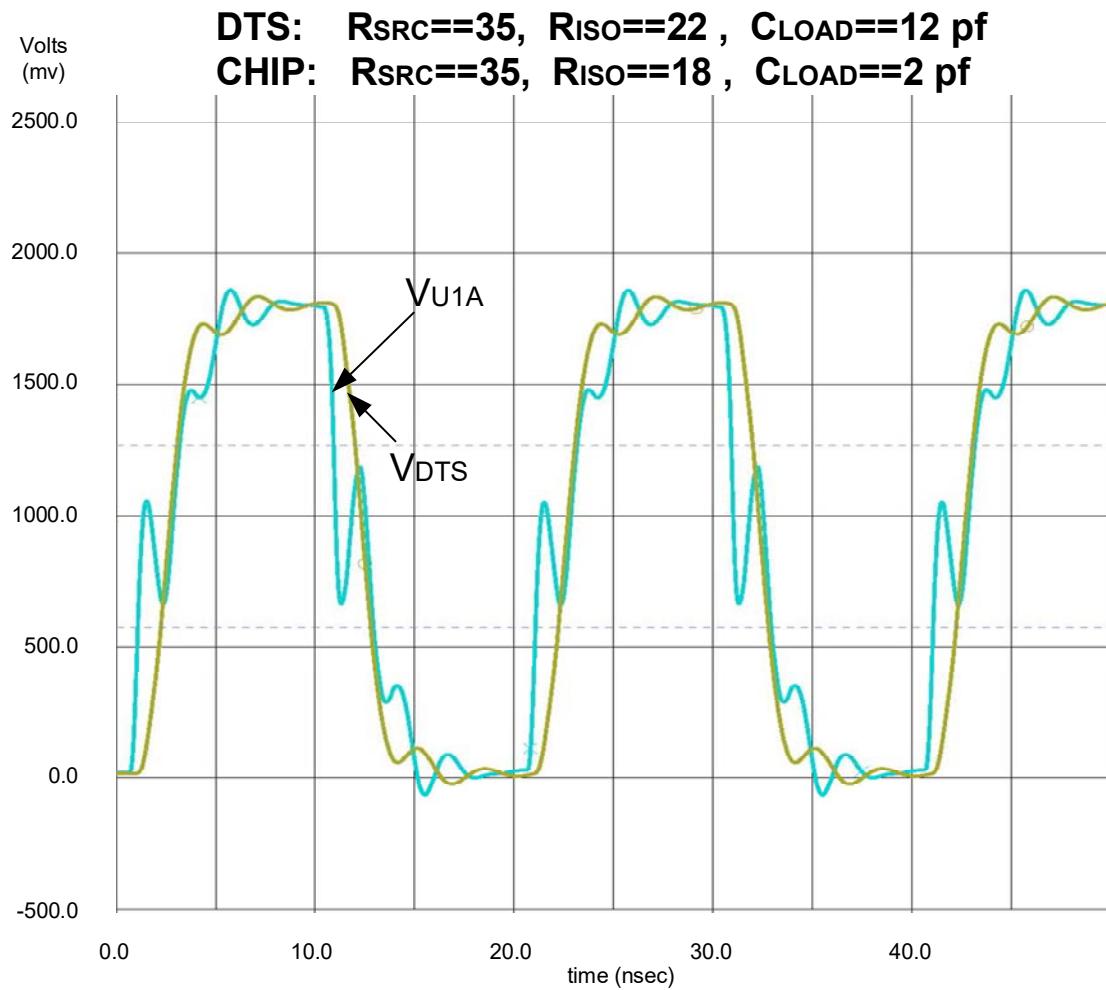


Figure F-68 — T configuration, 4 TAPs, one drive (U1B), 50 MHz

## U1A and U1B drive

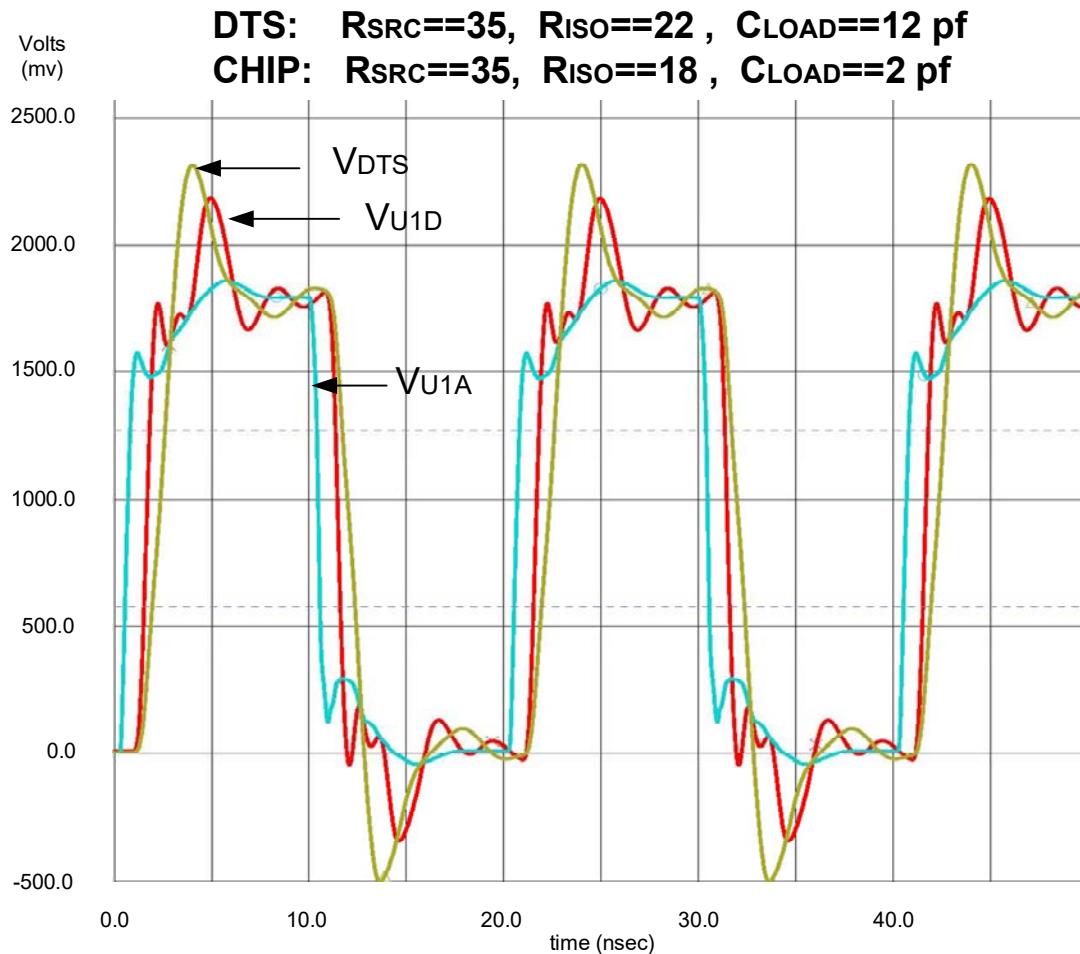
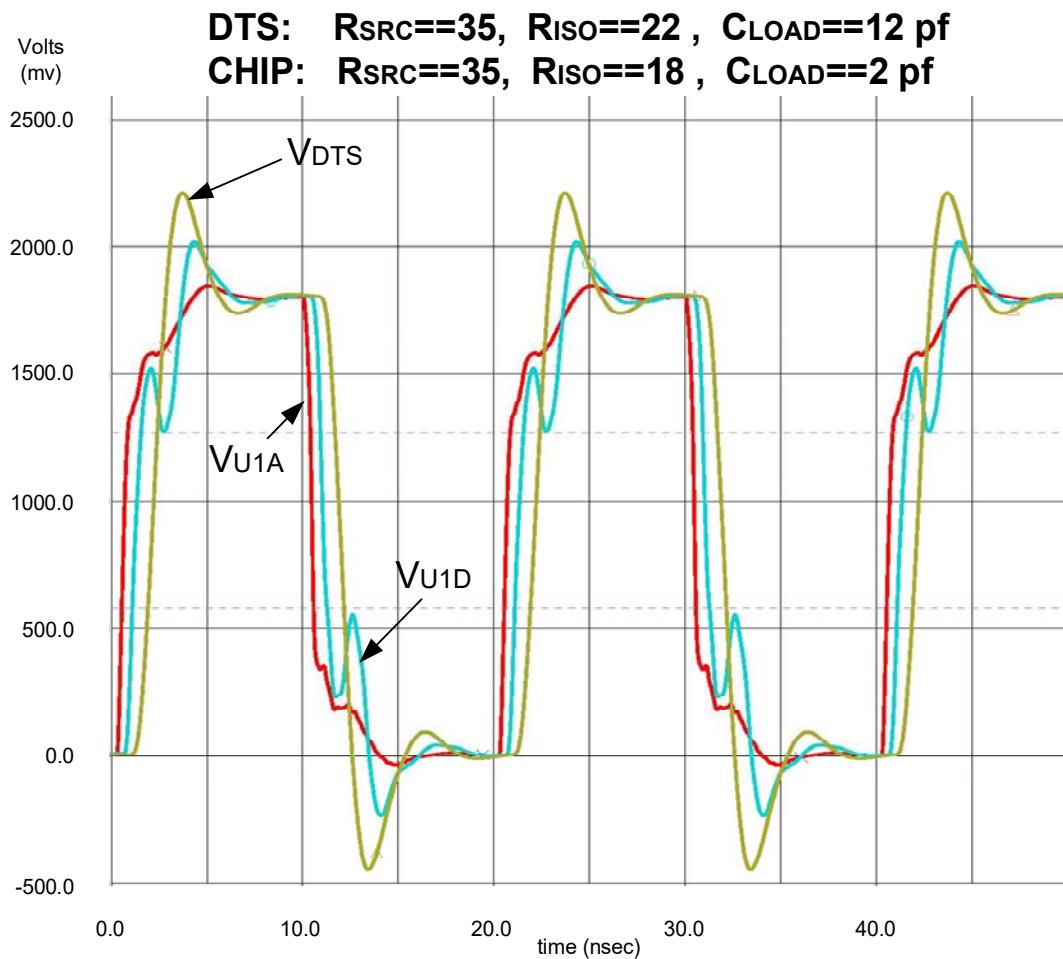


Figure F-69 — T configuration, 4 TAPs, two drives (U1A and U1B), 50 MHz

## U1B and U1D drive

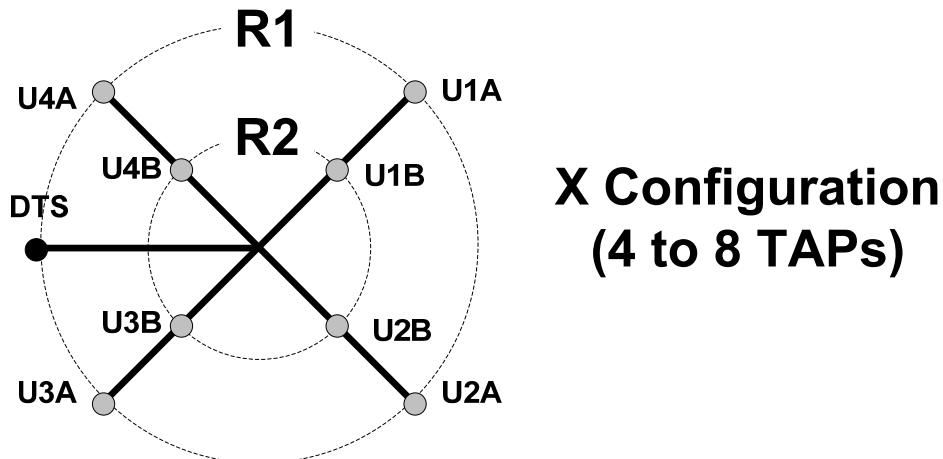


**Figure F-70 — T configuration, 4 TAPs, two drives (U1B and U1D drive), 50 MHz**

### F.11 X configuration of a transmission line

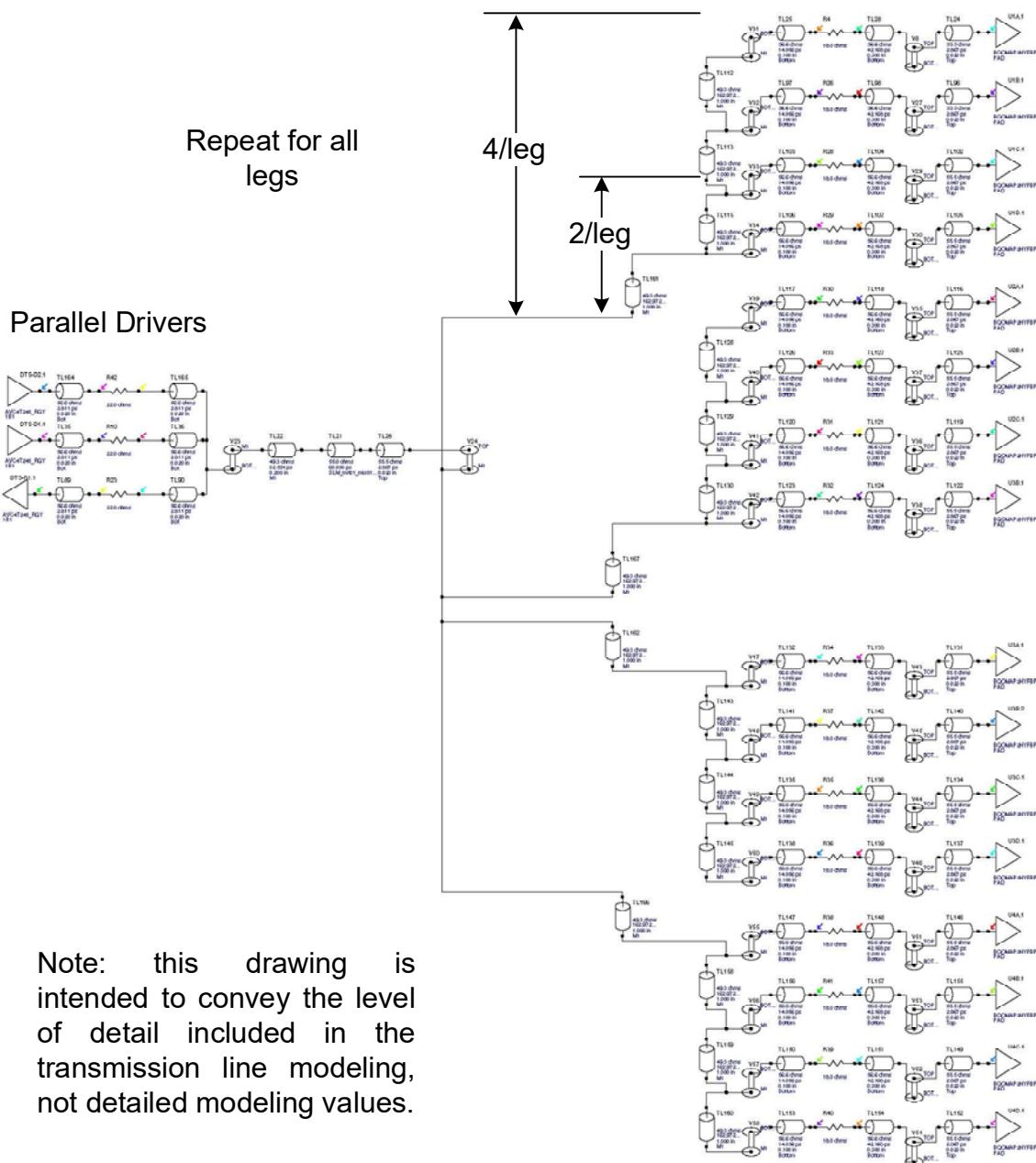
#### F.11.1 Model

The logical view of the X configuration of a transmission line is shown in Figure F-71. It shows the DTS and TAPs connected at two distinct distances from a central point. The values for R1 and R2 used in the simulations for the X configuration are one 25 mm (~1 in) and 50 mm (~2 in), respectively.



**Figure F-71 — Logical view of X configuration of a transmission line**

The model used for the X configuration of a transmission line is shown in Figure F-72. This figure shows all nodes as inputs. The direction of the node buffers are changed for cases when a node drives. Note that the source impedance has been lowered to roughly  $25 \Omega$ . This provides more current to splits in the transmission line. Within this figure, transmission-line segments are roughly  $56 \Omega$  with  $22 \Omega$  resistors in series with the two drivers operating in parallel. Feed-throughs are roughly  $50 \Omega$ .



**Figure F-72 — X model with parallel DTS drivers and four nodes/leg**

### F.11.2 Unidirectional signaling

The node (the fourth node) at the end of each leg of the X configuration shown in Figure F-71 is shown in Figure F-74 and Figure F-71. The massively parallel load makes the behavior similar to the behavior shown in Figure F-13 where the source impedance is higher than the load impedance. A comparison of the impedances in Figure F-74 and Figure F-75 are shown in Figure F-73.

Figure F-74

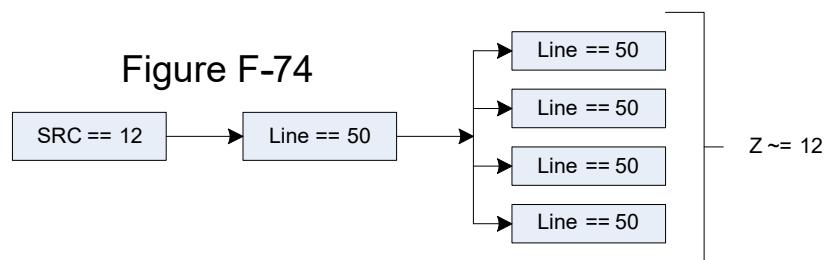


Figure F-75

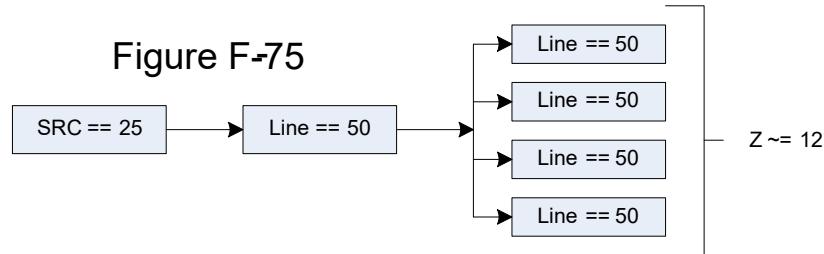
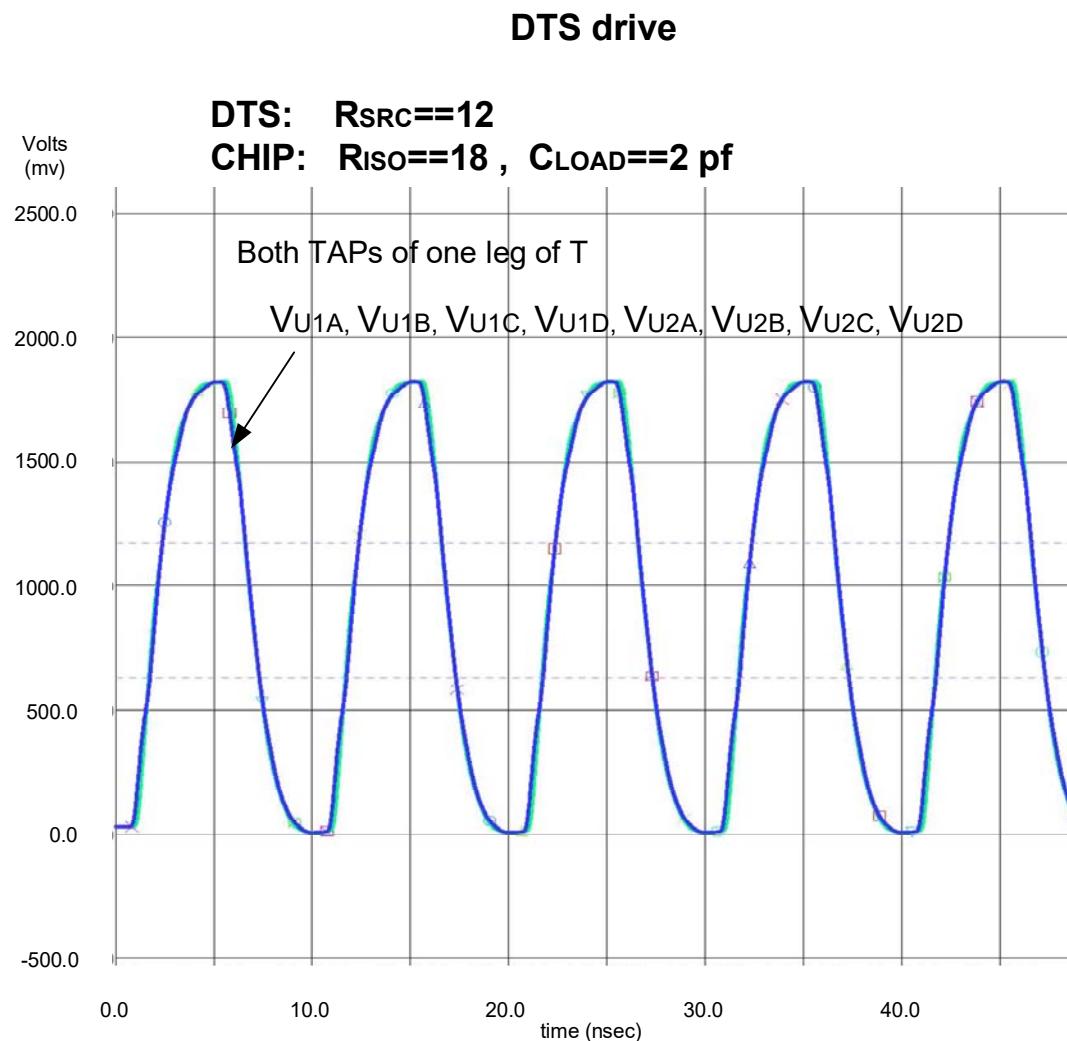
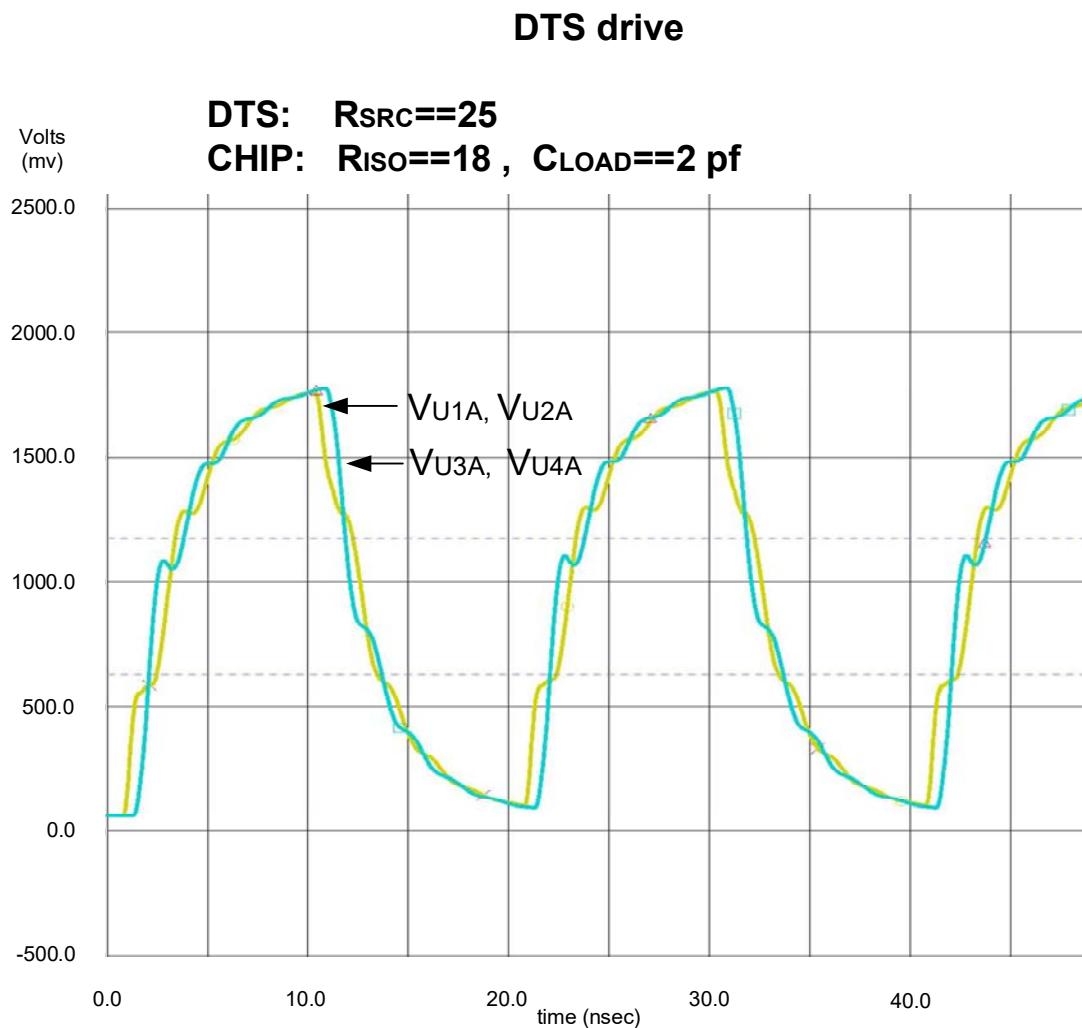


Figure F-73 — Contrasting matched and unmatched source and load impedances



**Figure F-74 — X configuration, 8 TAPs, DTS drive, 100 MHz, at destination**



**Figure F-75 — X configuration. 8 TAPs, DTS drive, 50 MHz**

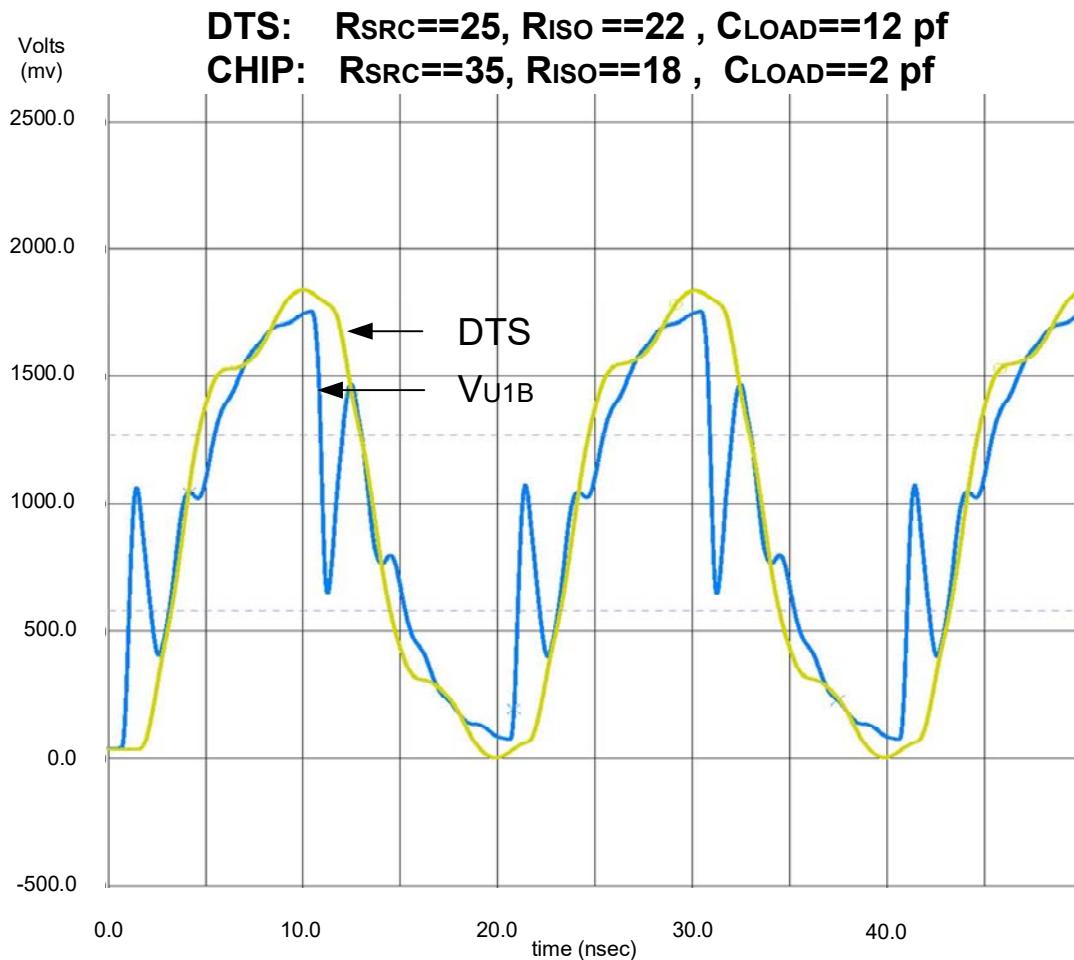
### F.11.3 Bidirectional signaling

Simulations for bidirectional signaling are shown in the following figures:

- Figure F-76 One drives (U1A)
- Figure F-77 One drives (U1B)
- Figure F-78 Two drives, U1A and U1B drive simultaneously
- Figure F-79 Four drives, U1A, U2A, U3A, and U4B drive simultaneously
- Figure F-80 All drive

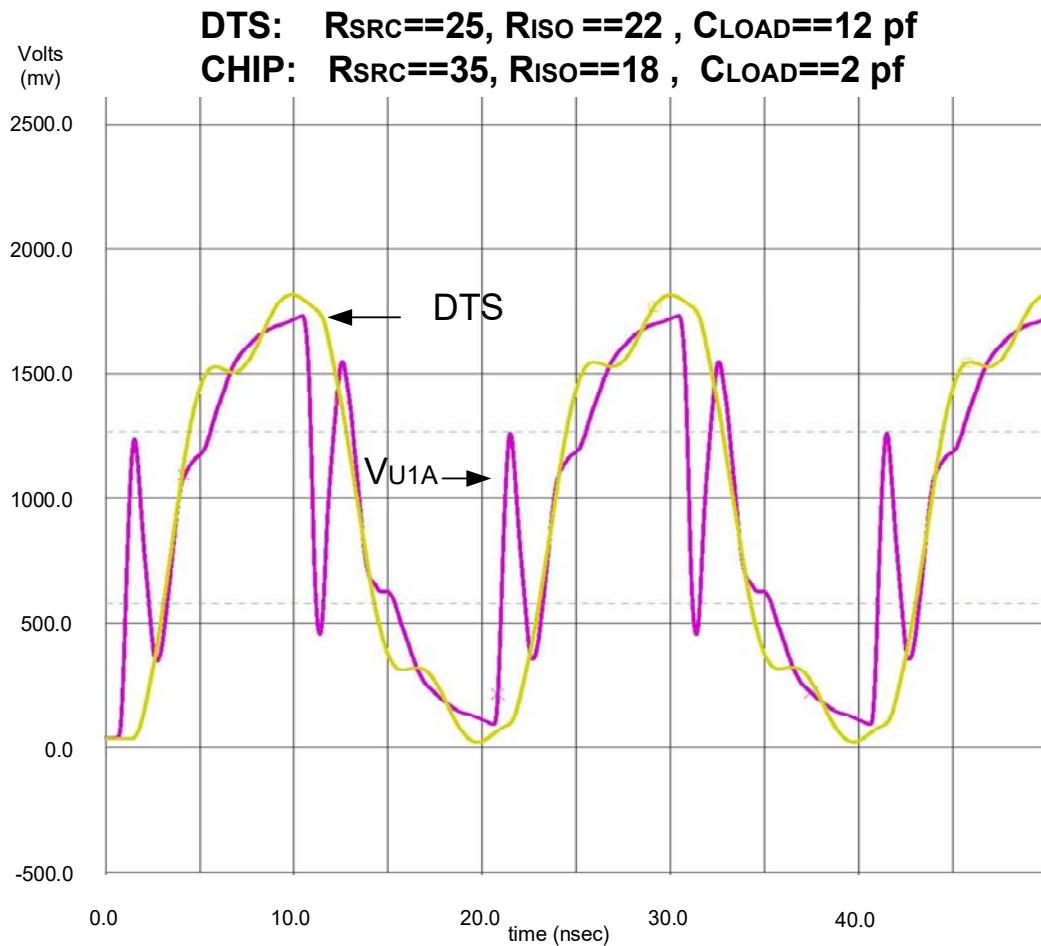
These simulations use the model shown in Figure F-72 at 50 MHz with eight chips.

## U1A drive



**Figure F-76 — X configuration, 8 TAPs, two TAPs/leg, one drive (U1A), 50 MHz**

## U1B drive



**Figure F-77 — X configuration, 8 TAPs, two TAPs/leg, one drive (U1B), 50 MHz**

## U1A and U1B drive

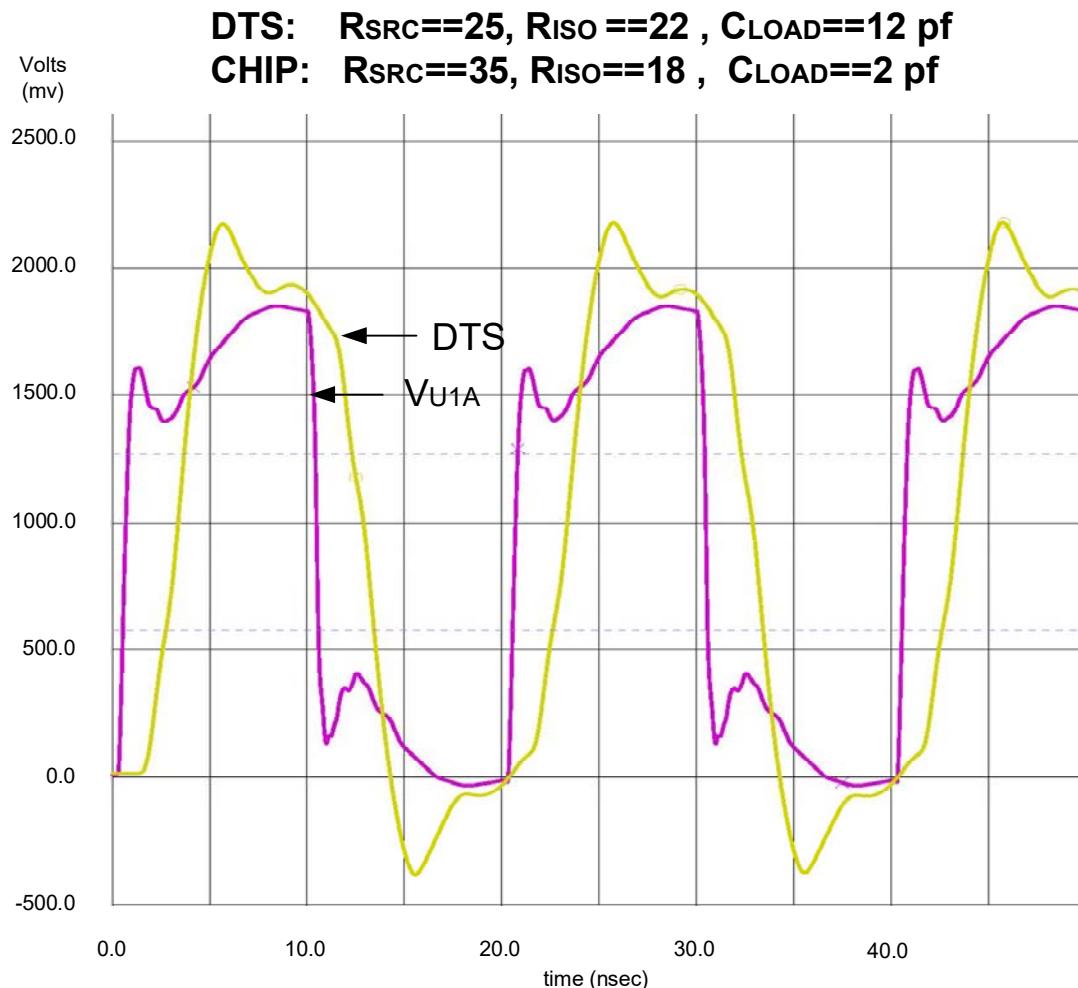


Figure F-78 — X configuration, 8 TAPs, two TAPs/leg, two drives (U1A and U1B), 50 MHz

## U1A, U2A, U3A, and U4A drive

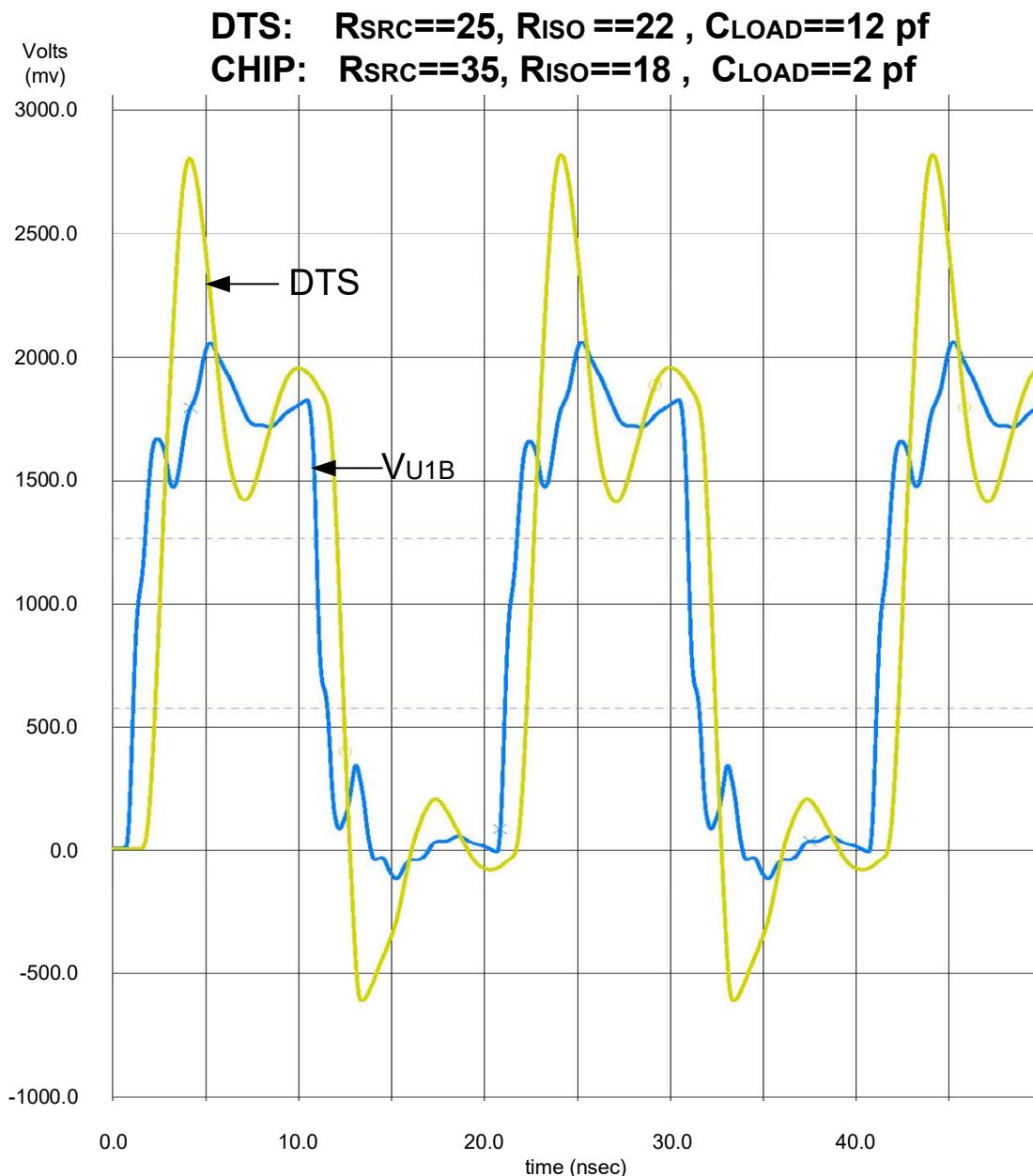


Figure F-79 — X configuration, 8 TAPs, two TAPs/leg, four drives, 50 MHz

## All drive

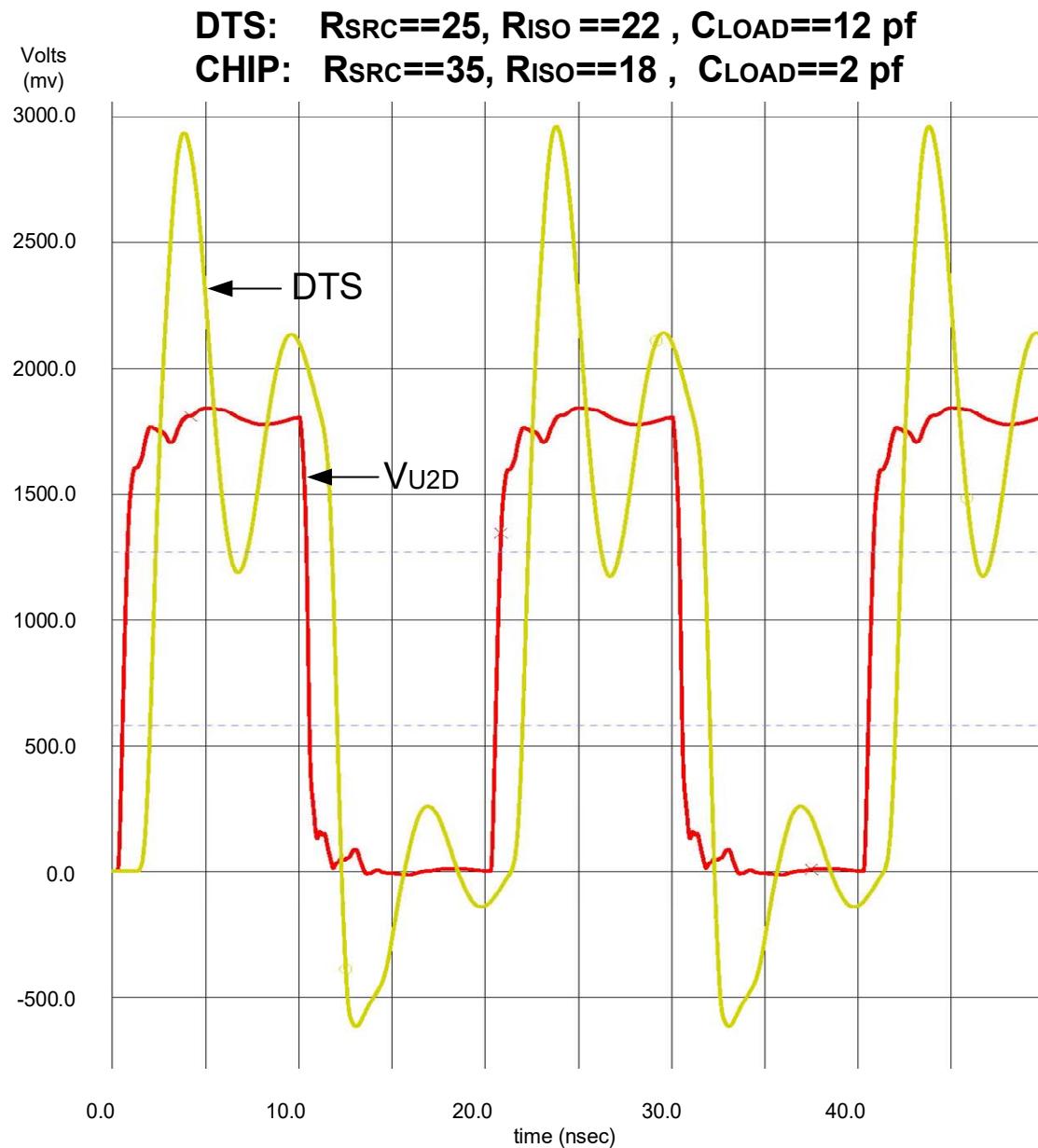
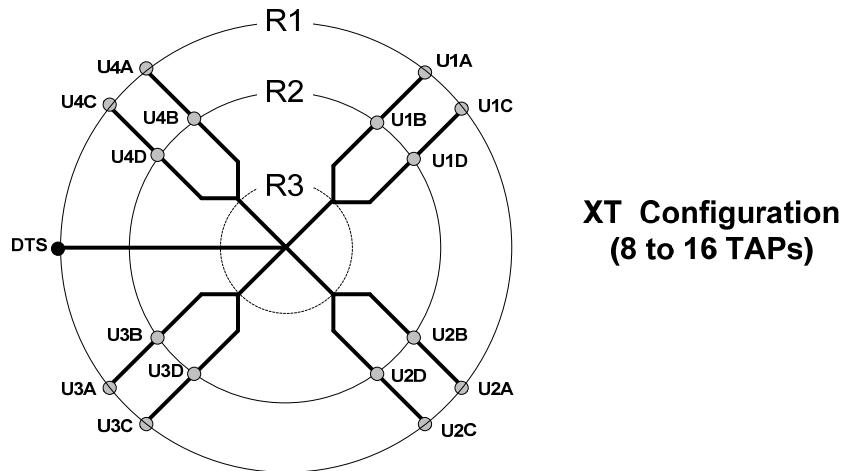


Figure F-80 — X configuration, 8 TAPs, two TAPs/leg, all drive, 50 MHz

## F.12 XT configuration of a transmission line

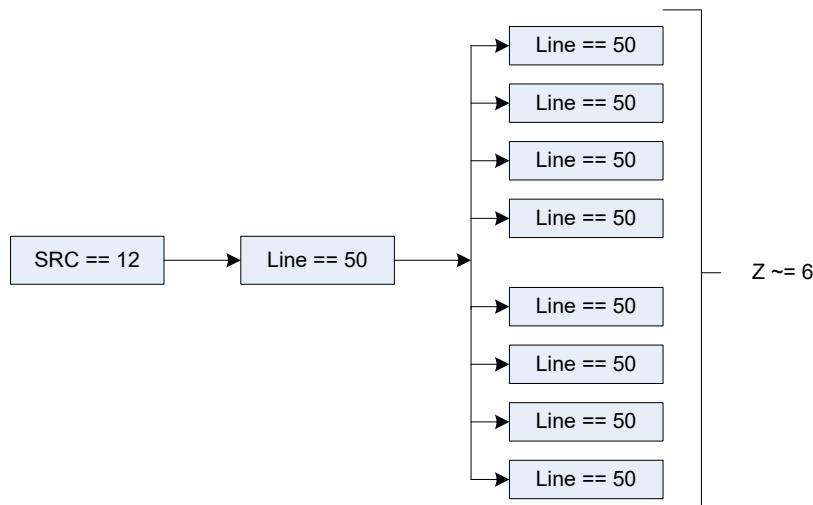
### F.12.1 Model

The logical view of the XT configuration of a transmission line is shown in Figure F-81. It shows the DTS and TAPs connected at two distinct distances from a central point. The values for R1 and R2 used in the simulations for this configuration are 25 mm (~1 in) and 50 mm (~2 in), respectively. R3 is zero.

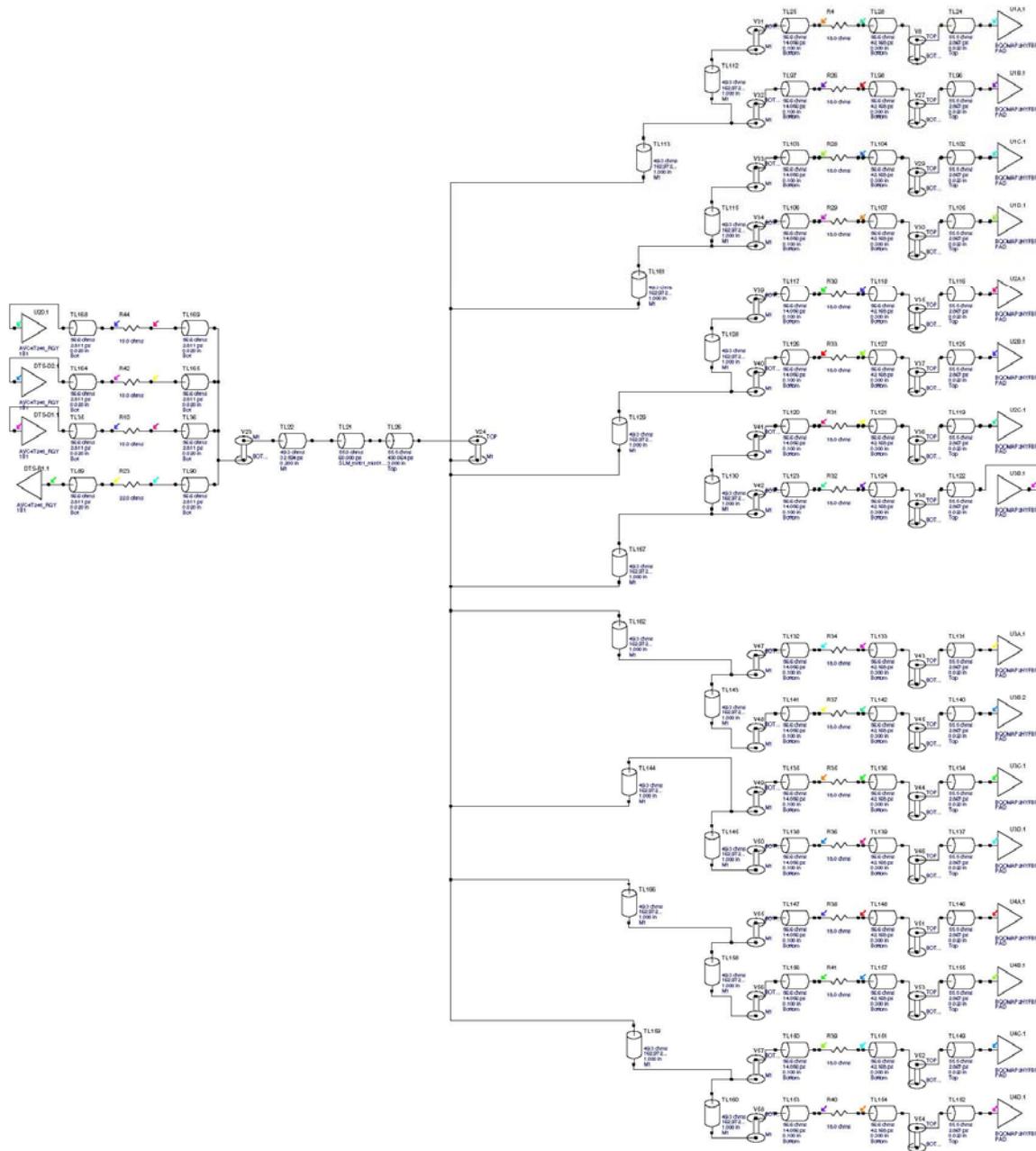


**Figure F-81 — Logical view of XT configuration of a transmission line**

The model used for the XT configuration of a transmission line is shown in Figure F-83. This figure shows all nodes as inputs. The direction of the node buffers are changed for cases when a node drives. Note that the DTS source impedance has been lowered to roughly  $12 \Omega$ . Within this figure, transmission line segments are roughly  $56 \Omega$  with  $10 \Omega$  resistors in series with the three drivers operating in parallel. Feed throughs are roughly  $50 \Omega$ . The board designer should check the DTS source impedance specifications before using this transmission-line configuration. The impedance profile of the XT configuration is shown in Figure F-82.



**Figure F-82 — Impedance profile of the XT configuration**

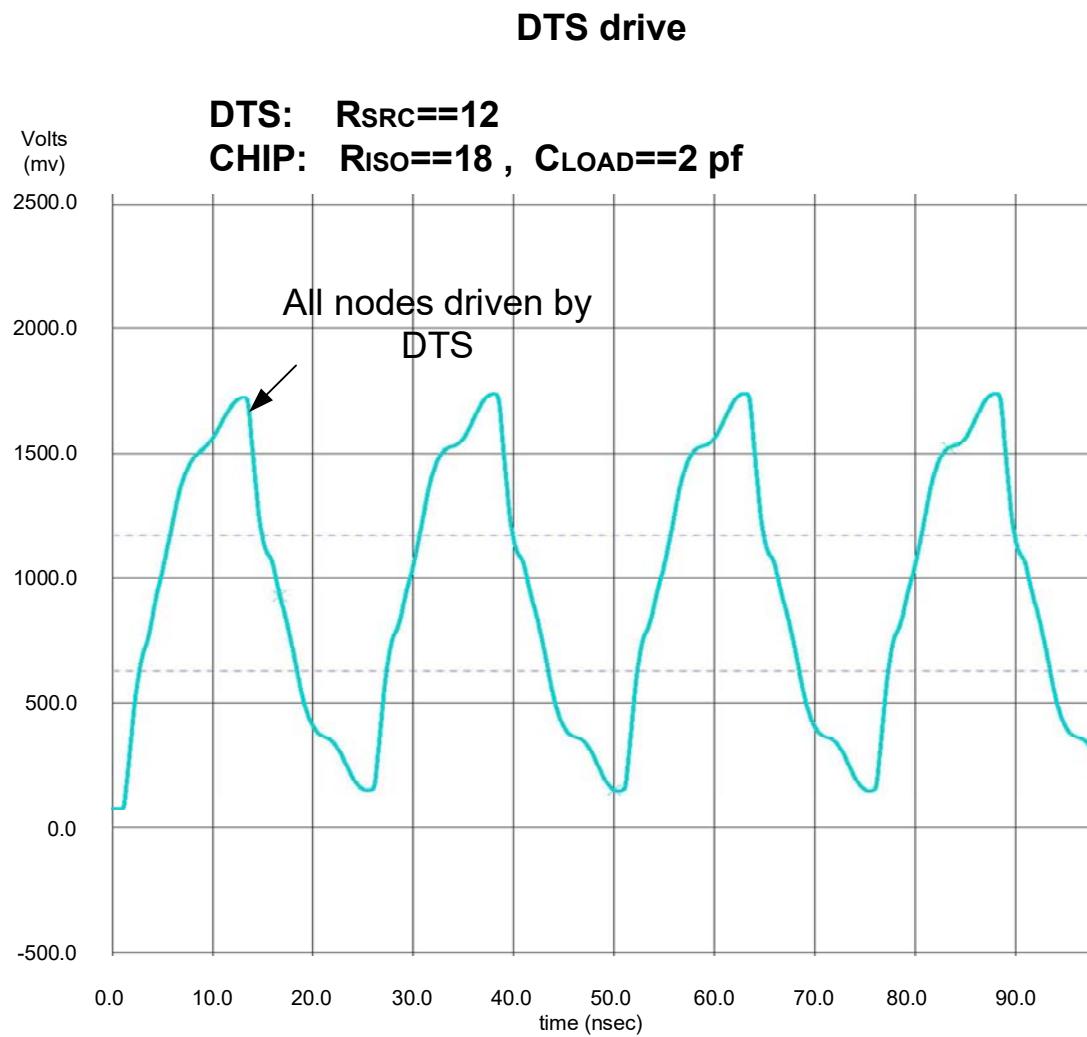


NOTE—This drawing is intended to convey the level of detail included in the transmission line modeling not detailed modeling values.

**Figure F-83 — Bidirectional, balanced delay, 16 TAPCs, detailed model**

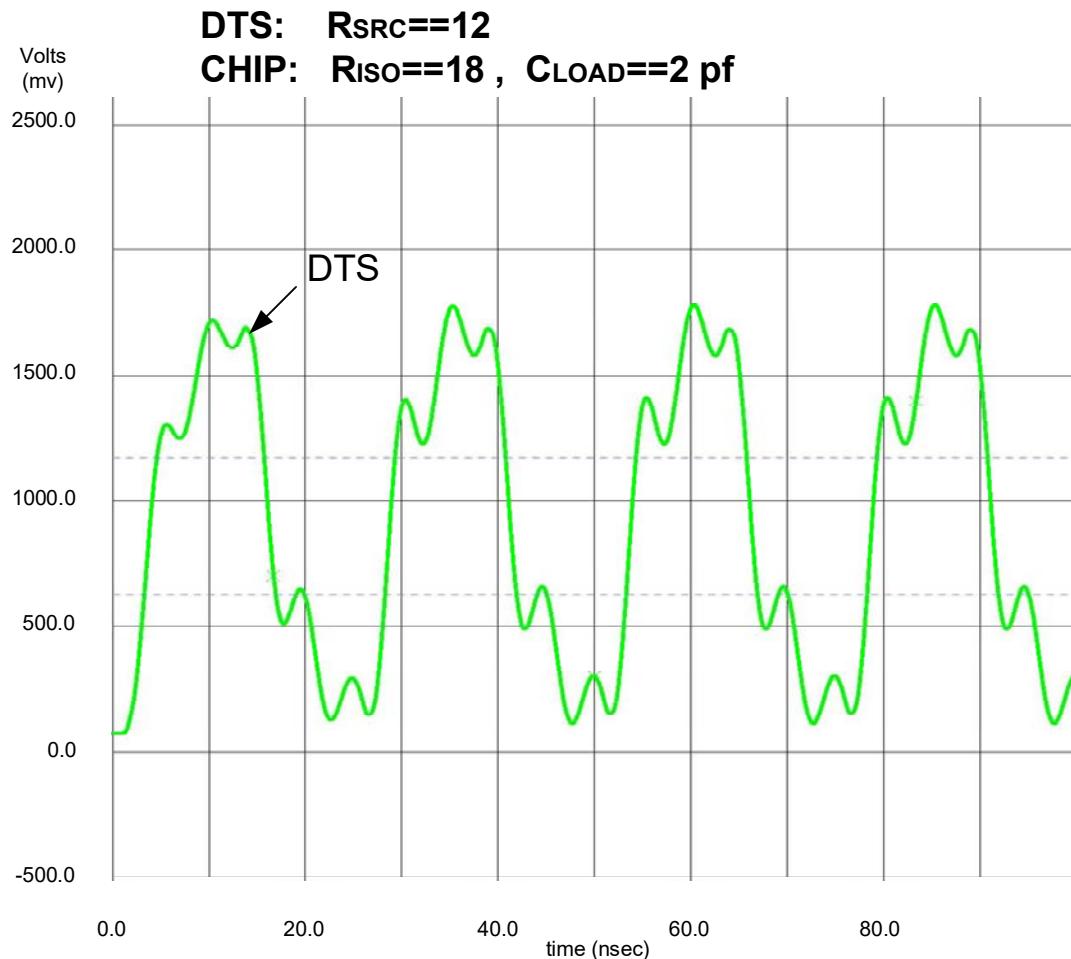
### F.12.2 Unidirectional signaling

With respect to the XT configuration shown in Figure F-81, a waveform for nodes of each leg is shown in Figure F-84 and the corresponding DTS waveform is shown in Figure F-85.



**Figure F-84 — XT configuration, 16 TAPs, nodes driven by DTS, 50 MHz, at destination**

## DTS drive



**Figure F-85 — XT configuration, DTS drive, 50 MHz, at source**

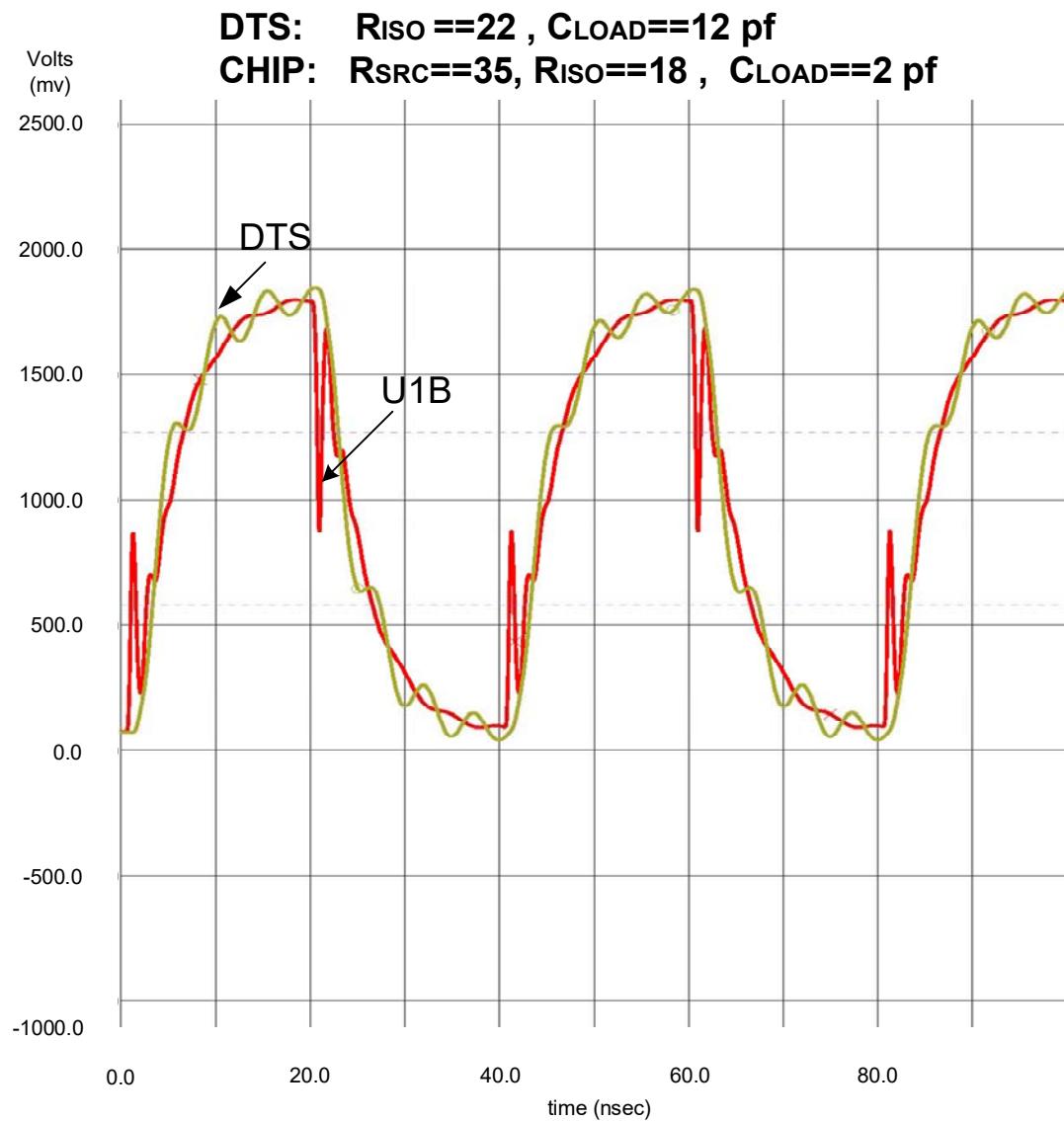
### F.12.3 Bidirectional signaling

Simulations for bidirectional signaling are shown in the following figures:

- Figure F-86 One drives (U1A)
- Figure F-87 One drives (U1B)
- Figure F-88 One drives and neighbor (U1A and U1B)
- Figure F-89 Two drives simultaneously (U2B and U4B)
- Figure F-90 Two drives simultaneously (U1B, U2B, U3B, and U4B)
- Figure F-91 All drive simultaneously, DTS signal

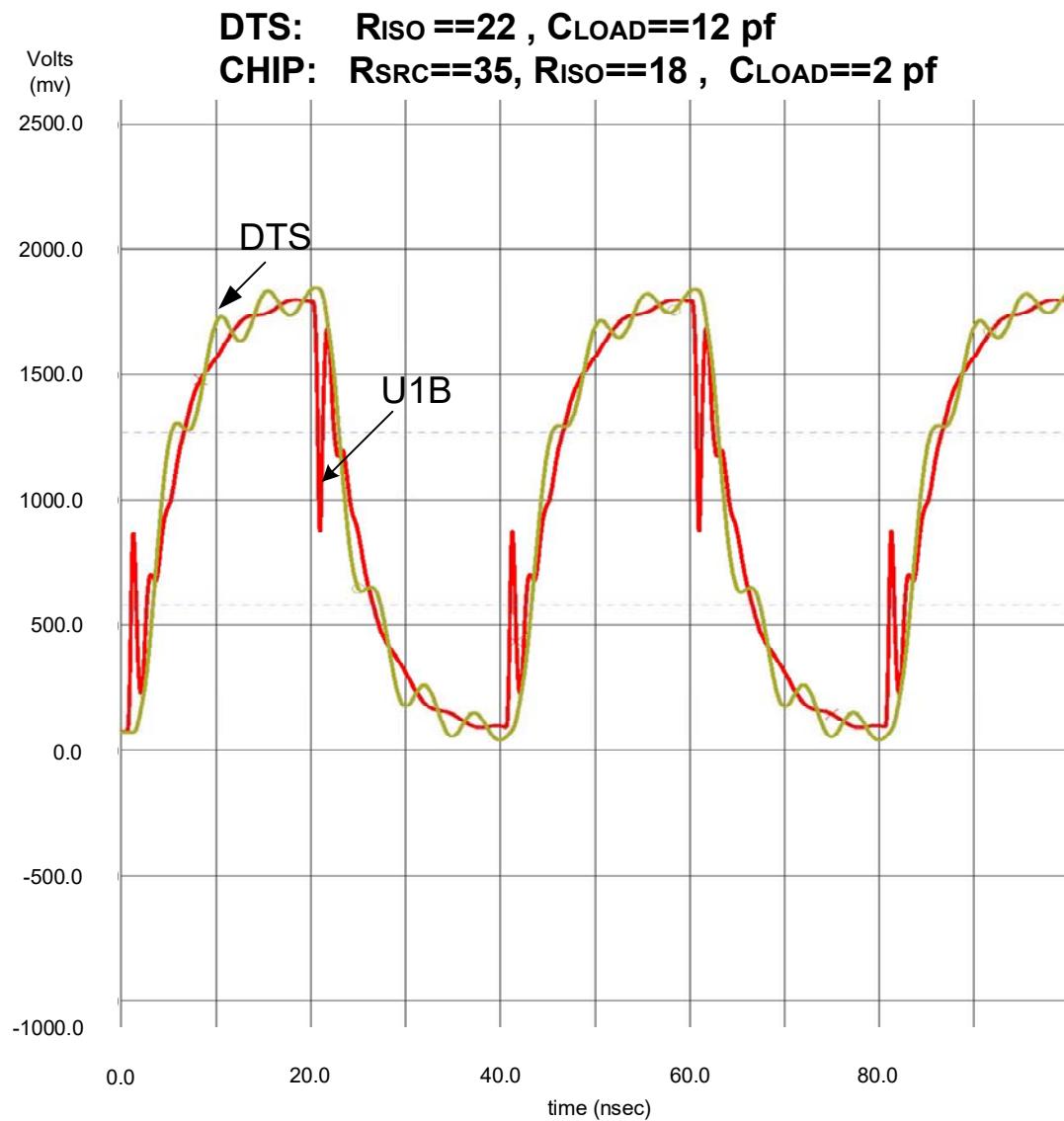
These simulations use the model shown in Figure F-83 at 50 MHz with 16 chips.

### U1A drives



**Figure F-86 — XT configuration, 16 TAPs, one drives (U1A), 50 MHz**

### U1A drives



**Figure F-87 — XT configuration, 16 TAPs, one drives (U1B), 50 MHz**

### U1A drives

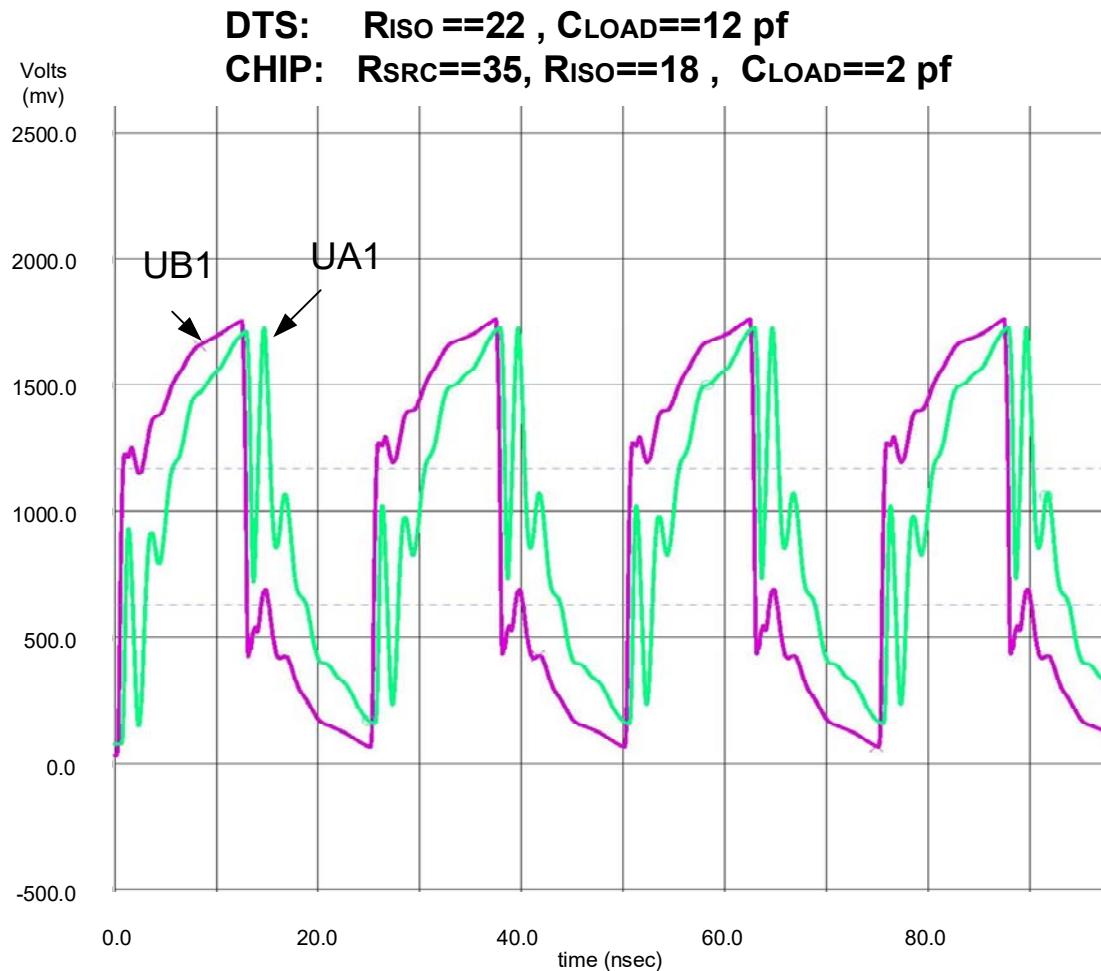
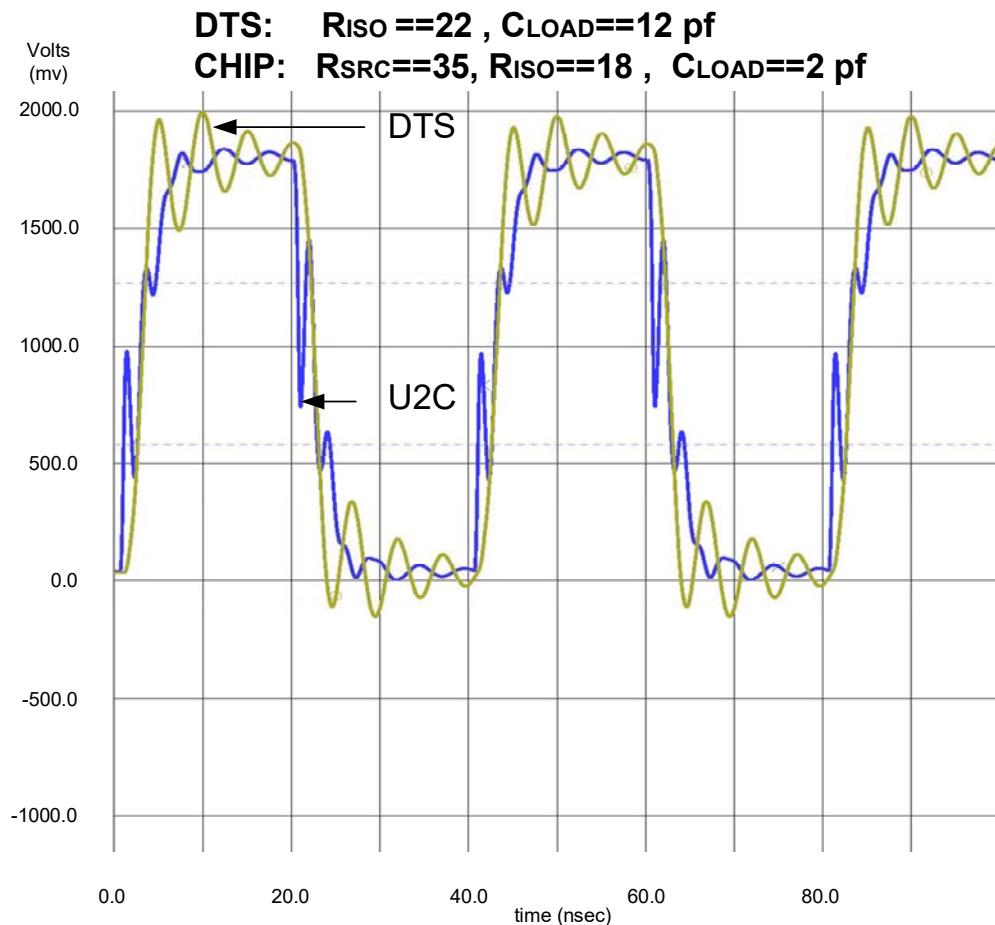


Figure F-88 — XT configuration, 16 TAPs, one drive and neighbor, 50 MHz

### U2B and U4B drive



**Figure F-89 — XT configuration, 16 TAPs, two drives, 50 MHz**

## U1B, U2B, U3B, and U4B drive

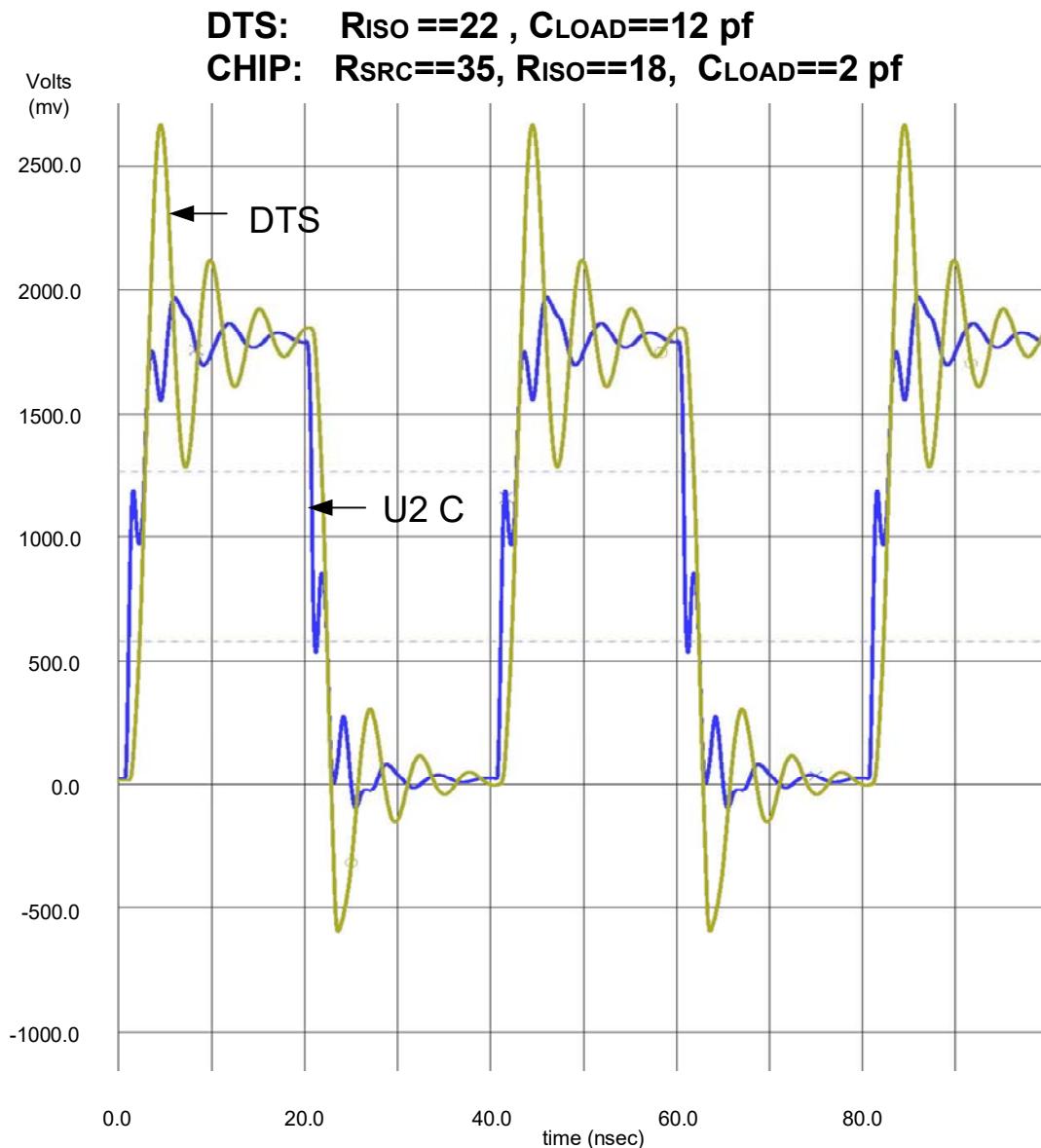
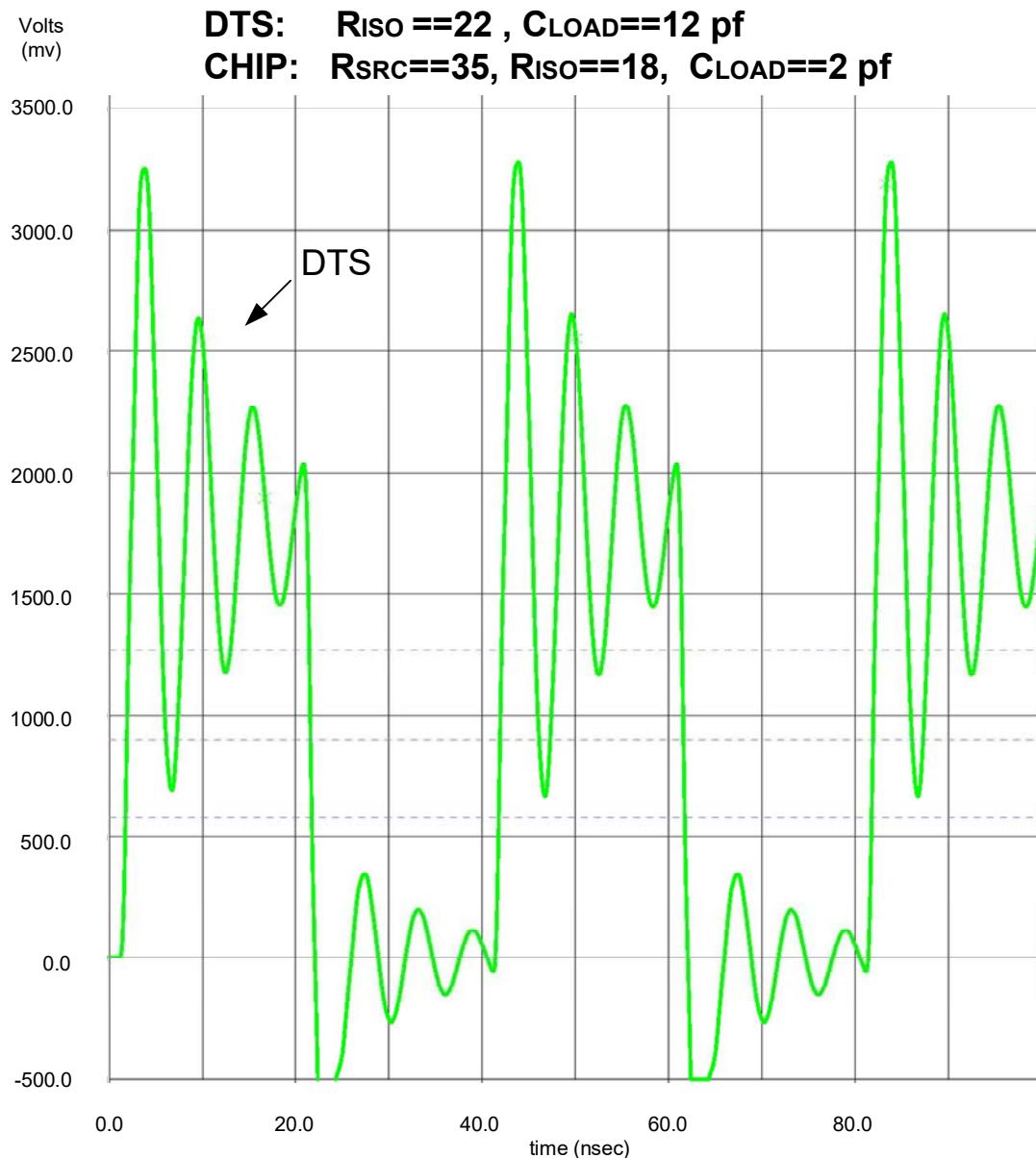


Figure F-90 — XT configuration, 16 TAPs, four drives, 50 MHz

## All drive



**Figure F-91 — XT configuration, 16 TAPs, all drive, DTS observed, 50 MHz**

## F.13 Recommendations

### F.13.1 Summary

The recommendations are summarized in Table F-2.

**Table F-2 — Recommendations summary**

Area	Recommendation
Chip	<ul style="list-style-type: none"> <li>1) Inputs with hysteresis and input filtering with a one-shot</li> <li>2) Close attention to ground/and supply issues related to simultaneously switching signals</li> <li>3) Slew-rate controlled buffer outputs</li> <li>4) Minimize clock to output valid timings</li> </ul>
Board	<ul style="list-style-type: none"> <li>1) Simulate the system to understand it</li> <li>2) Always minimize transmission-line length</li> <li>3) Eliminate or minimize the length of transmission-line stubs if elimination is not possible with all configurations (splits in T, X, and XT configurations are exceptions)</li> <li>4) Choose a connectivity scheme compatible with source impedances</li> <li>5) Choose a connectivity scheme compatible the over/undershoots that the DTS can tolerate</li> <li>6) Buffer the clock, if necessary, to protect its integrity, especially when there may be chips with no signal integrity countermeasures (trade a lower operating frequency for reliability)</li> <li>7) Match line lengths with T, X, and XT transmission-line configurations</li> <li>8) Utilize series terminations in combination with appropriate <math>R_{ISO}</math> values</li> <li>9) Utilize <math>R_{ISO}</math> resistors (resistors in series with inputs/outputs, isolation resistors) to isolate capacitive loads</li> <li>10) Increase the <math>R_{ISO}</math> value to over-damp the signal as the number of TAPs increases when signal filtering is required</li> <li>11) If possible, place chips with no countermeasures near the end or at the end of line configuration transmission lines to provide to them the best quality clock signal</li> </ul>
DTS	<ul style="list-style-type: none"> <li>1) Programmable output impedance</li> <li>2) Utilize <math>R_{ISO}</math>s (resistors in series with inputs/outputs) to isolate capacitive loads</li> <li>3) Inputs with hysteresis</li> </ul>

### F.13.2 Connectivity/termination scheme

A few recommended termination schemes are shown in Figure F-92 through Figure F-94. The termination scheme is selected based on whether the signal is bidirectional, the number of TAPs sharing the DTS connection, and the scan topology. When there are multiple technology branches, the termination schemes should be chosen based on that required by a Star-2 Scan Topology, if used, a Star-4 Scan Topology, if used, and then a Series Scan Topology, if used. The recommendations for the termination schemes for the TAP signals are shown in Table F-3. Series termination schemes are recommend for balanced star topologies (T, X), where there are too many legs, or the legs are too long, to effectively use parallel termination. The value of  $R_{CHIP-SERIES}$  is adjusted based on the value of  $R_{CHIP-SRC}$  and  $C_{CHIP}$ . The value of  $R_{DTS-SERIES}$  is adjusted based on the value of  $R_{DTS-SRC}$  and  $C_{DTS}$ .

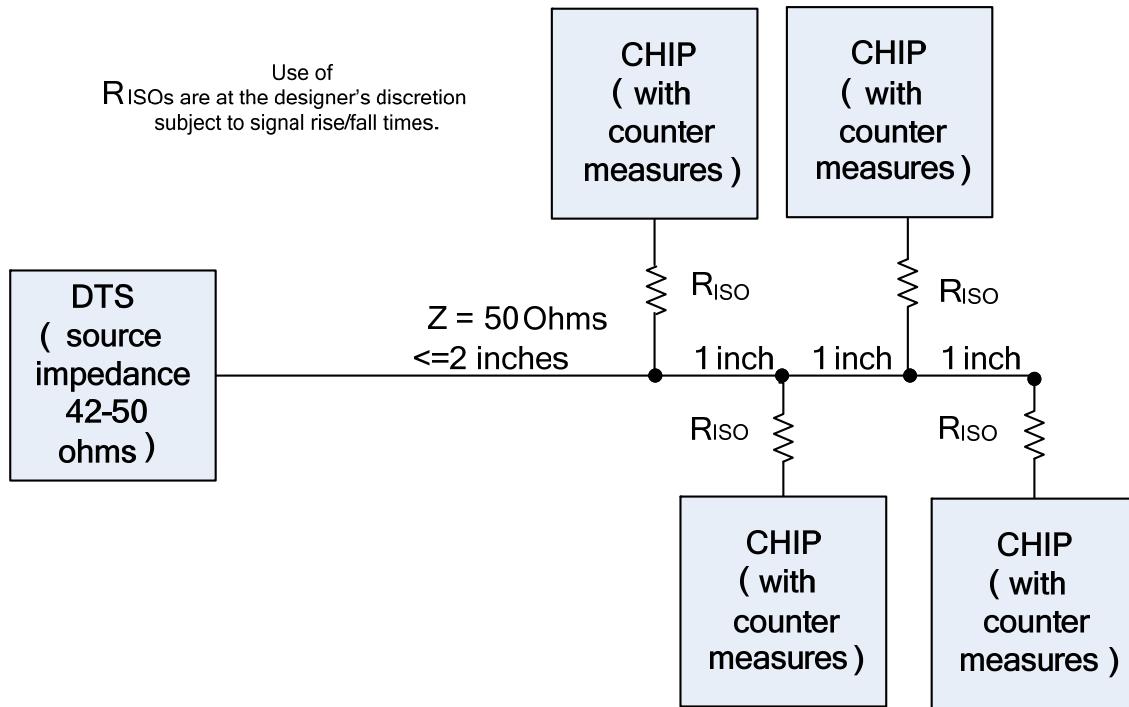


Figure F-92 — Recommended series termination scheme with one to four chips

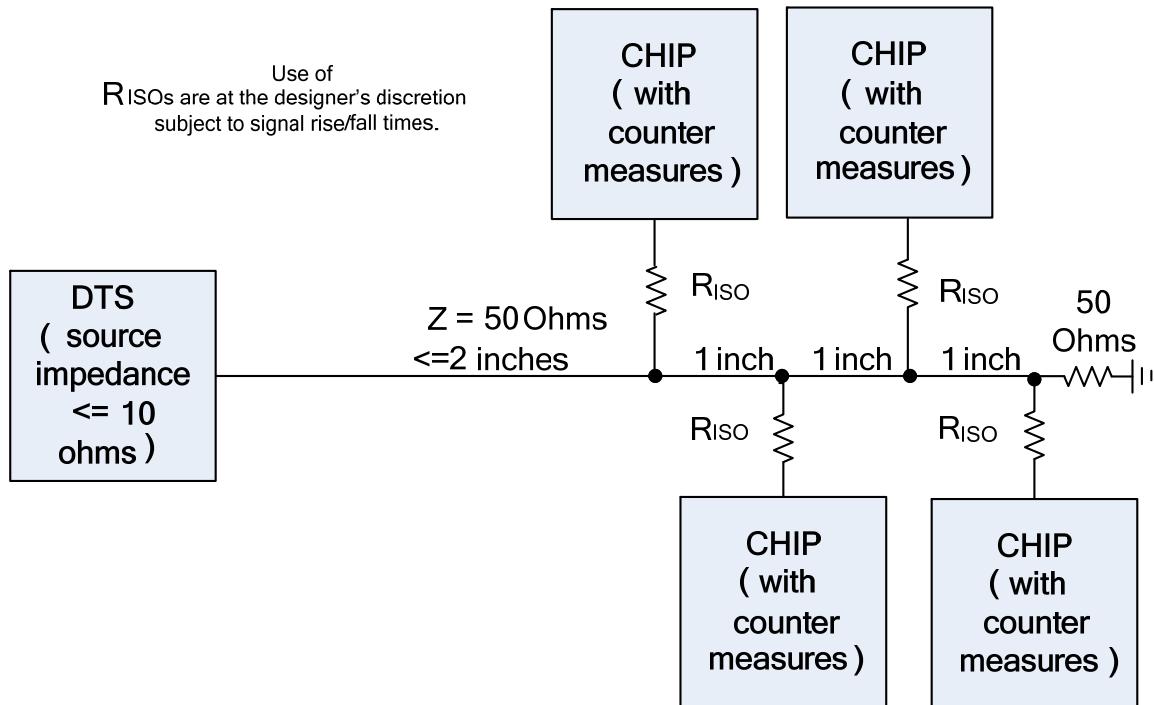
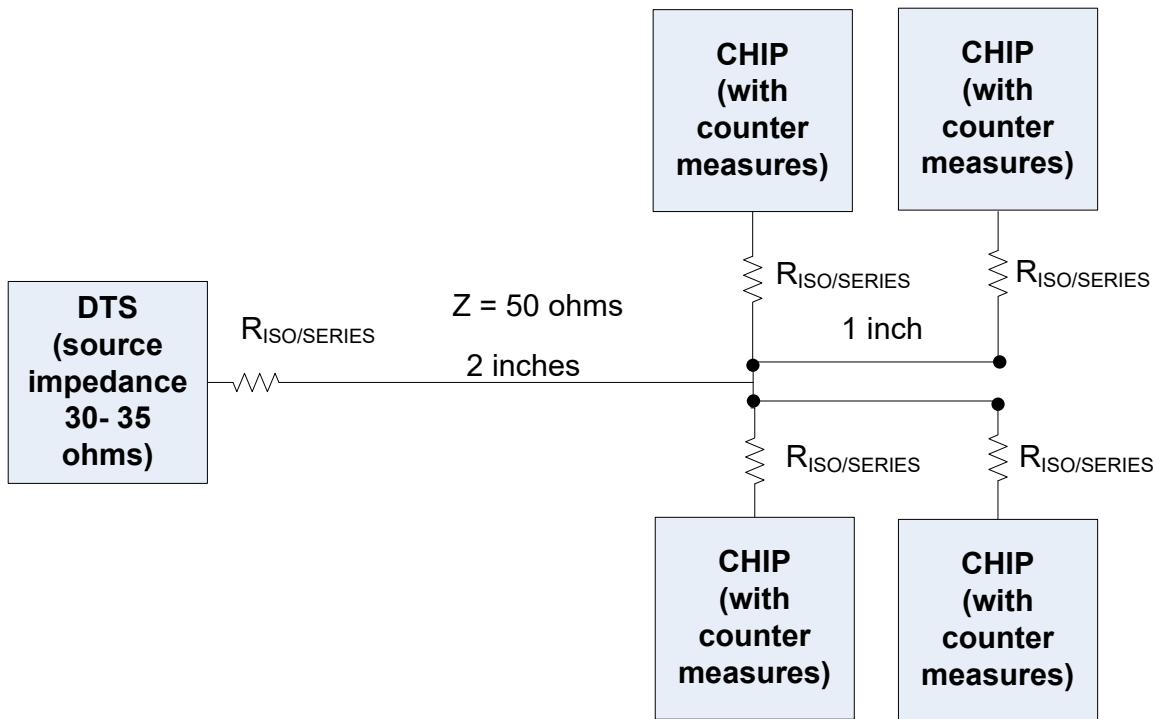


Figure F-93 — Recommended parallel termination scheme with one to four chips

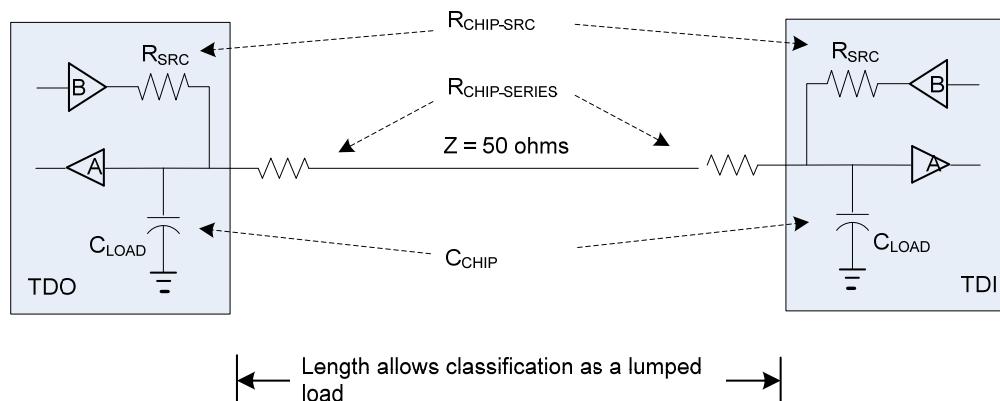


**Figure F-94 — Recommended series termination scheme for highest performance four/six chips**

Systems that have more than six chips and have chips that do not have countermeasures may need to have DTS signals that have slower slew rates.

### F.13.3 Termination scheme/capacitive load isolation/signal relationships

Note that the TDI(C) and TDO(C) signals in a Series Scan Topology may or may not require the capacitive isolation and source impedance resistors shown in Figure F-95. This determination is based on whether these signals are classified as having lumped or distributed loads as described in F.5.1.3.



**Figure F-95 — Determining signal load classification**

**Table F-3 — Recommended connection/termination schemes**

Configuration			Recommendations	
Topology	Number of TAPs	Signal	Connection scheme	Isolation resistor value
Series	1	TCK(C)	Point-to-Point	15–27
		TMS(C)		15–27
		TDI(C)		(15–27) <sup>a</sup>
		TDO(C)		(45— $R_{CHIP-SRC}$ ) <sup>a</sup>
	2–4	TCK(C)	Line	15–27
		TMS(C)		15–27
		TDI(C)		(15–27) <sup>a</sup>
		TDO(C)		(45— $R_{CHIP-SRC}$ ) <sup>a</sup>
	5–8	TCK(C)	T (if DTS compatible) or buffered on board, otherwise Line	15–27
		TMS(C)		15–27
		TDI(C)	Point-to-Point	(15–27) <sup>a</sup>
		TDO(C)		(45— $R_{CHIP-SRC}$ ) <sup>a</sup>
	9–16	TCK(C)	X (if DTS compatible) or Buffered on Board, otherwise Line	18–33
		TMS(C)		18–33
		TDI(C)	Point-to-Point	(18–33) <sup>a</sup>
		TDO(C)		(45— $R_{CHIP-SRC}$ ) <sup>a</sup>
Star-4	2–4	TCK(C)	Line	15–27
		TMS(C)		15–27
		TDI(C)		(15–27) <sup>a</sup>
		TDO(C)		(45— $R_{CHIP-SRC}$ ) <sup>a</sup>
	5–8	TCK(C)	T (if DTS compatible) or buffered on board	15–27
		TMS(C)		15–27
		TDI(C)	Point-to-Point	(15–27) <sup>a</sup>
		TDO(C)		(35— $R_{CHIP-SRC}$ ) <sup>a</sup>
	9–16	TCK(C)	X (if DTS compatible) or Buffered on Board	18–33
		TMS(C)		18–33
		TDI(C)	Line or T	18–33 <sup>a</sup>
		TDO(C)		(35— $R_{CHIP-SRC}$ ) <sup>a</sup>
Star-2	1	TCK(C)	Point-to-Point	15–27
		TMS(C)		(45— $R_{CHIP-SRC}$ ) <sup>a</sup>
	2–4	TCK(C)	Line	15–27
		TMS(C)		(45— $R_{CHIP-SRC}$ ) <sup>a</sup>
	5–8	TCK(C)	T (if DTS compatible)	15–27
		TMS(C)		(35— $R_{CHIP-SRC}$ ) <sup>a</sup>
	9–16	TCK(C)	X (if DTS compatible) or Buffered on Board	18–33
		TMS(C)		(35— $R_{CHIP-SRC}$ ) <sup>a</sup>

<sup>a</sup> When it is a transmission line (see F.5.1.3).

The  $R_{CHIP}$  value for inputs is specified assuming a  $C_{CHIP}$  value of 7 pF. If the value of  $C_{CHIP}$  is less than 7 pF, the  $R_{CHIP}$  value may be decreased proportionally. Likewise, if the value of  $C_{CHIP}$  is greater than 7 pF, the  $R_{CHIP}$  value may be increased proportionally.

#### F.13.4 Utilizing board design tools and simulation

The importance of simulating the system particularly with CAD-tool extracted PCB models cannot be overemphasized. Accommodating empirical experimentation at the prototyping level should also be considered (varying  $R_{ISO}$  values for instance), as the  $R_{CHIP}$  and  $C_{CHIP}$  values created by the chip manufacturing processes vary.

## Annex G

(informative)

### Utilizing SScan Scan Formats

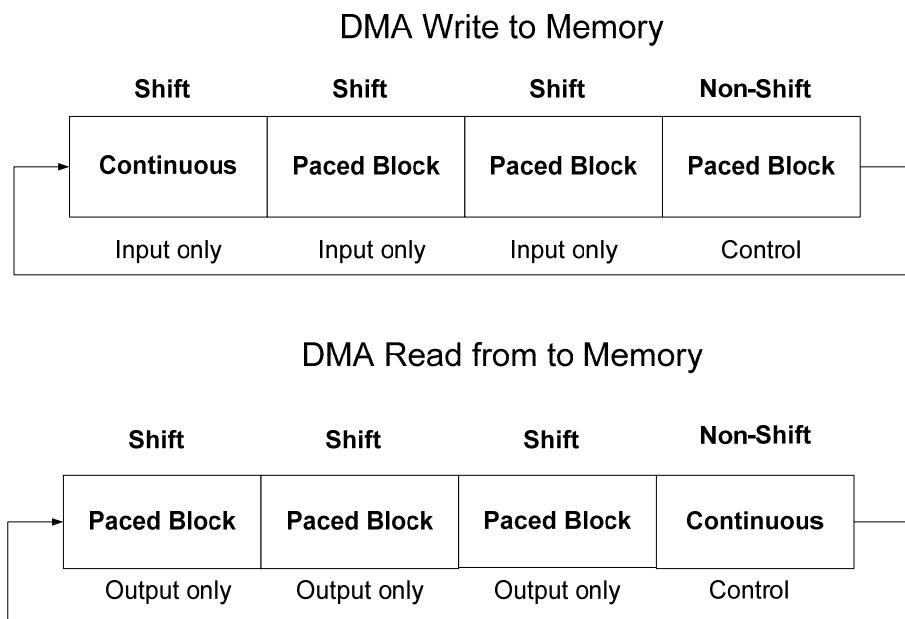
#### G.1 Communicating with a DMA or similar component

Within this annex, the term DMA is used to represent any function that can accept or generate a large number of bits per transaction. In the following description, segment types are given the following names based on their stall profile:

- Continuous No-stall profile
- Paced Block First SP stall profile
- Paced Bit All-stall profile

Typical communication with components (such as a DMA) which have variability in their data rates involves software or hardware polling per each transaction. As the polling moves higher into the data-handling infrastructure, the system performance degrades. Segmented scans provide a means to move the polling to the lowest possible point in the infrastructure, the scan itself. This can have a dramatic impact on performance in some instances.

A common debug operation is moving a block of memory or register data from the TS to the DTS or vice versa. The width of the data transferred is usually fixed, with the transfer being a very repetitive operation of moving data and, in some cases, additional status information. The number of data and status bits can be made to correspond to the number of *Shift-DR* TAP controller states within a Data Segment. The SScan0 or SScan2 Scan Format is used, depending on the source of the TCK(C) signal. Typical read and write operations are shown in Figure G-1.



**Figure G-1 — SScan0/SScan2 DMA Transfers**

The segment types shown in this figure are used to construct a DMA transfer of N words, with the segment types selected based on the behavior of the DMA component. Since the Data Segment is no particular length, status bits such as error, timeout, empty segment, or other information may be included in each segment. These examples demonstrate how the general purpose nature of segments and their ordering provides a means to tailor the data transfer to the characteristics of virtually any component.

### G.1.1 Write operation

In the case of a DMA write operation, the DTS supplies data to the DMA. The first Data Segment is specified as being continuous. Since its transfer is continuous, it provides the first word to the DMA. The second Data Segment specified as Paced Block. This segment creates a stall opportunity at the beginning of the second DMA word. This SP waits until the DMA indicates it has dispositioned the data within the continuous segment by generating a ready to proceed indication with, for example, moved to a buffer or otherwise dispositioned, provided by the Data Segment. A Paced Block segment could also be used instead of the continuous segment provided the DMA indicates it is ready to accept data when it has nothing to do. The second Data Segment proceeds when the DMA interface indicates it is ready to accept the second word. This continues segment by segment until the transfer is completed. The Control Segment following the last Data Segment waits until the DMA indicates it has dispositioned the last data word (it has nothing to do). Variations of this flow may be created for a double-buffered unit (e.g., the first two Data Segments may be continuous).

### G.1.2 Read operation

In the case of a DMA read operation, the DMA supplies data to the DTS. The first Data Segment is specified as a Paced Block. The stall opportunity at the beginning of this segment ensures the data being read is available before the first TDO bit of the segment is exported. When output data is available, the segment continues. This process repeats with subsequent Data Segments. When all words have been read, a Control Segment is used. The Control Segment following the Data Segment is specified as continuous as there is no need to insert a stall opportunity after the last word is read.

### G.1.3 Error checking

Note the first SP of a block segment contains both TDI and TDO information. The TDO bit could be used to convey a DMA error to the DTS on a word-by-word basis. Paced Block segments may also be used to provide a means to create status segments.

## G.2 Communicating with rate-dependent TAPC

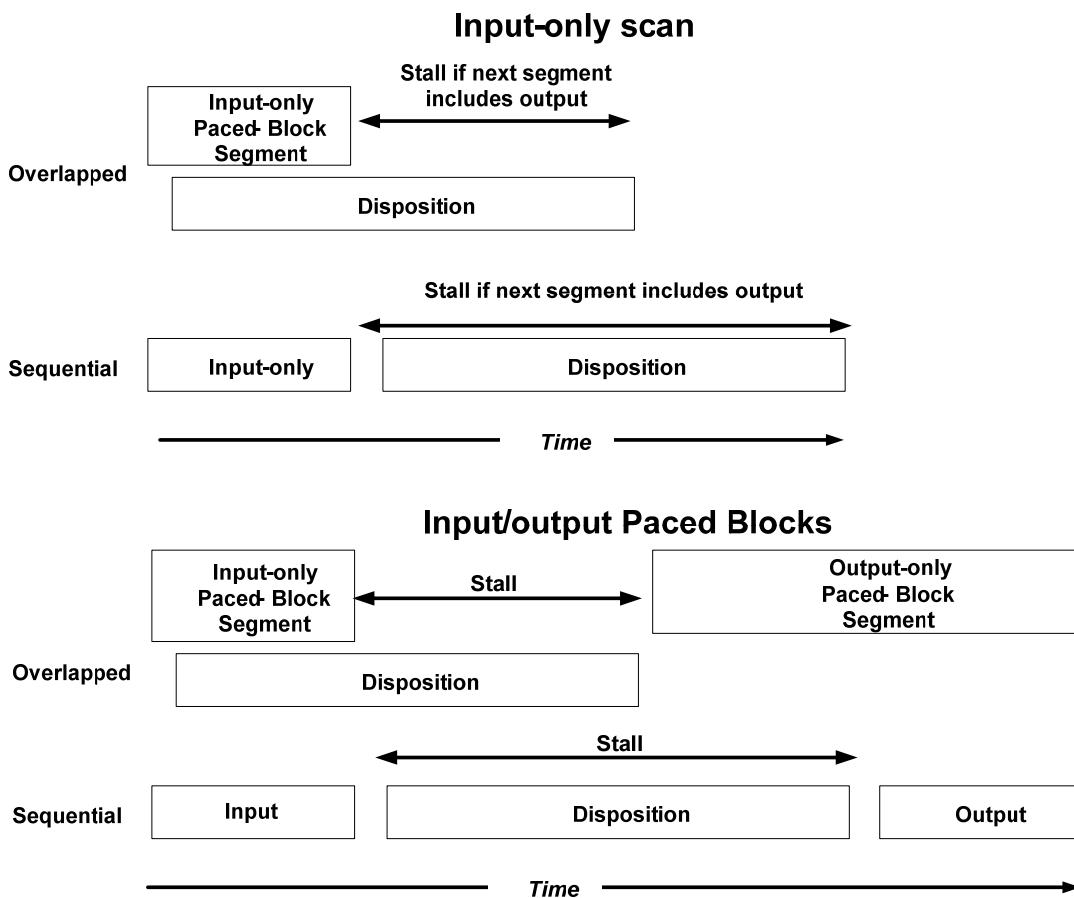
SScan Scan Formats provide support for boosting scan performance when the operation of an EMTAPC connected to the CLTAPC is entangled with a clock domain other than the TCK(C) clock domain. Several options for handling with this type of unorthodox operation are as follows:

- Using only Paced-Bit segments
- Using Paced-Block segments to control buffered input
- Using a mix Paced-Block and Paced-Bit transfers to manage input and output

### G.2.1 Buffering input and output

Instead of providing an opportunity to stall the scan transaction within each SP, it is more efficient if the input and output are buffered with an on-chip FIFO. In this case, the information carried by the SP may be dispositioned at a rate other than the TCKC rate. The dispositioning of the FIFO data may also begin as

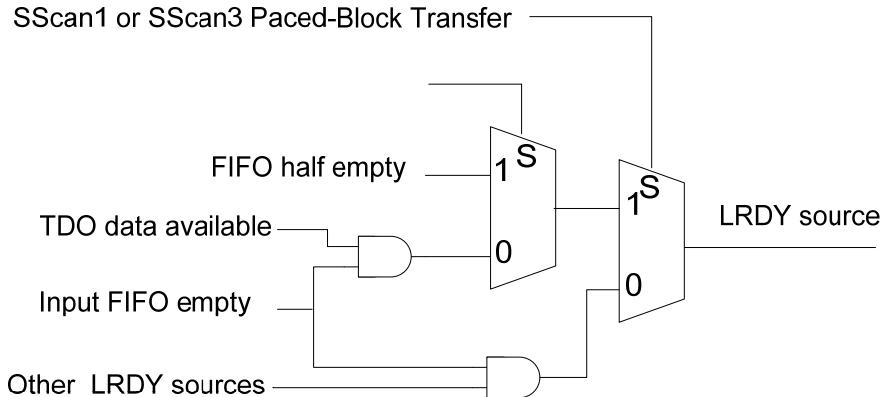
soon as data arrives. This overlaps the input and dispositioning of data. This is noticeably superior to an approach where there is no overlapping as shown in Figure G-2.



**Figure G-2 — SScan0/SScan2 transfers with Paced Block**

Using Paced-Block segments for input and output with a TAP with an added return TCK (RTCK) (a TAP1.R) signal requires on-chip buffering of input and output. This FIFO (or double buffer) buffers the TMS bits for Control Segments and data bits for Data Segments. In the examples above, providing scan input takes one segment (to provide) input. It is placed in an input FIFO. As the FIFO is being filled, it is being emptied at the rate that is compatible with the TAP1.R. The TAP1.R controller state is advanced as this FIFO is emptied. The output is either stored in a second FIFO or in the FIFO with TDI information as the TDI information is consumed. With an input-only transfer, the output data is discarded. Where the output data is the data of interest, the input scan takes place to load the output FIFO and the output-only Paced-Block segment retrieves the buffered output without causing the TAP controller state to advance.

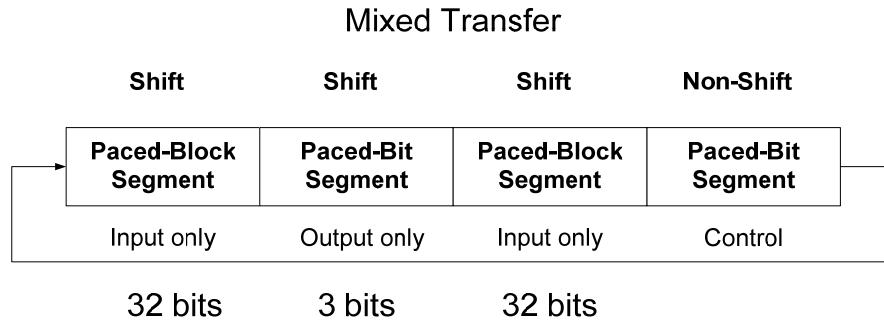
The buffering may be implemented in several ways. For illustrative purposes, assume a 32-bit FIFO is constructed on chip. Since the DTS controls the segment size, it is set to half of the size of the FIFO or 16 bits. The first Scan Packet of a segment contains RDY and TDO bits. At the beginning of each segment that is input only (control and input-only Data Segments) a logic 1 RDY bit value is generated, if the FIFO has 16 or more empty entries. If the segment is a Data Segment with output, an RDY value of one is generated when the FIFO is empty and TDO data generated by the prior segment is available. The FIFO is not filled for these segments. In this case, segments are therefore less than or equal to 16 bits. A conceptual view of the RDY generation for TAP1.R transfers is shown in Figure G-3.



**Figure G-3 — Example RDY generation for SScan1 or SScan3 Paced-Block transfers**

### G.2.2 Mixing Paced-Block and Paced-Bit transfers

In some cases, it may be advantageous to mix Paced-Block and Paced-Bit transfers when communicating with a TAP.1R. This may occur when the transfer is dominated by input. In this case, only an input buffer is implemented. It is used to transfer all input with the performance benefits discussed previously. Instead of buffering a small number of output bits, they may be transferred using a Paced-Bit transfer as shown in Figure G-4.



**Figure G-4 — SScan1/SScan3 mixed Paced-Bit and Paced-Block transfers**

This approach may be used even when input and output FIFOs are deployed. When there are a number of output bits (e.g., reading memory data), input and output transfers are used as they are more efficient. For writing memory data with a small bit of status returned, Paced-Block transfers would be used for input and Paced-Bit transfers would be used for output. When a Paced-Block transfer is followed by a Paced-Bit transfer or an input-only segment, the output FIFO contents are flushed as they are clearly of no interest.

The flexibility of Paced-Bit and Paced-Block transfers with the SScan1 and SScan3 Scan Formats provides a means to implement a number of different solutions for accelerating the scan performance to certain TAP.1Rs by more than a factor of two.

## Annex H

(informative)

### The RTCK signal

#### H.1 Introduction

At times, the IEEE 1149.1 TCK is a completely asynchronous clock to a function it is managing. This is not the case for a boundary-scan application as the boundary cells are under the control of the TAPC.

When the TAP is used for debugging an applications processor, data has to be transferred between the TCK clock domain and the applications processor clock domain. This requires a synchronization mechanism to transfer data between the two clock domains.

One approach to this has been to directly synchronize TAP controller state progression to the processor clock, although this strategy is contrary to the intent of IEEE Std 1149.1. As long as the processor clock has a fixed frequency, this approach simply introduces a higher setup-time for the TMS and TDI and a higher clock-to-output-time for TDO, but conceptually the TAP still works as expected. The additional set-up time and clock-to-output time are a consequence of the time which the synchronization logic needs to synchronize the TCK edges (which control the TAP controller state progression) to the processor clock. The only consequence of the higher set-up time and clock-to-output time is that the maximum frequency of the TCK at which the communication still works is lower.

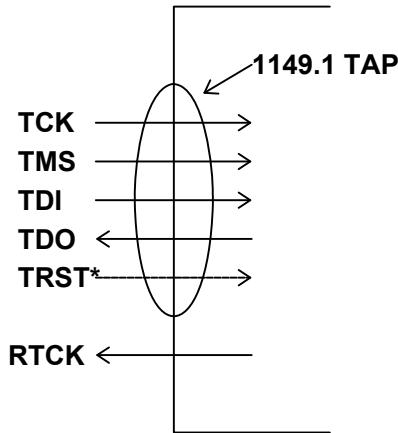
Unfortunately, the processor clock in modern systems is usually not fixed, especially in the boot phase of an application; the PLLs which drive the processor clock are often not enabled, so the processor runs with a slow clock. Even after the boot phase, modern power management will decrease and increase the processor clock frequency dynamically to conserve energy.

The result is that the maximum allowed TCK frequency changes dynamically, depending on the current frequency of the processor clock.

DTS equipment could simply use a very low TCK frequency to communicate with the processor, but usually this leads to an unacceptable performance. To get a better performance, the DTS equipment has to change the TCK frequency dynamically. The DTS equipment has to ensure the TCK frequency is never too high. It requires feedback from the TS when the approach described above is deployed.

This standard deprecates this approach, discourages its further use, but tolerates its existence.

The RTCK signal shown in Figure H-1, which is an output from the TS, is a legacy mechanism which was invented to allow dynamic adaptation of the TCK frequency. The RTCK signal basically “acknowledges” each edge (rising and falling) on the TCK. The DTS equipment has to wait for the acknowledgment before sending the next edge on the TCK.



**Figure H-1 — An IEEE 1149.1 TAP with RTCK signal**

An RTCK signal (or a similar mechanism) should not be implemented for any future IEEE 1149.1 TAP-based CPU debug interfaces, or for any other use. To enable the usage of IEEE 1149.1 TAP-based CPU debug interfaces which already contain an RTCK signal, this standard includes features that can be used to handle this kind of interface when it is embedded in the chip.

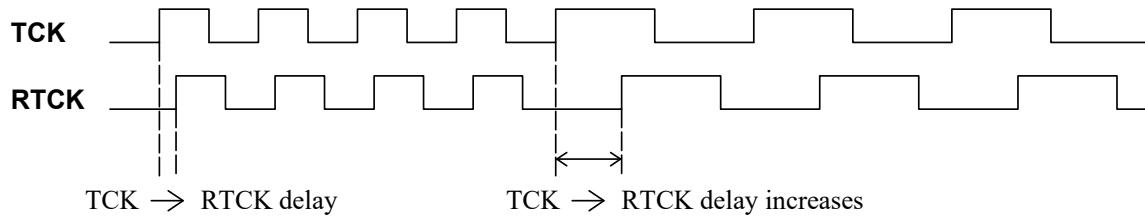
## H.2 Expected behavior of an RTCK signal

The following subclauses describe how the RTCK mechanism has been used in previous designs and what the expected behavior is, if an RTCK signal is implemented.

As described in the introduction, the RTCK signal “acknowledges” the TCK signal. The DTS is expected to change the value of the TCK only when the RTCK and TCK currently have the same value. When the TCK and RTCK have a different value, the DTS equipment has to wait until the RTCK reaches the same value as the TCK. By keeping its current value, the RTCK can, in principle, stall a transition on the TCK indefinitely. This is called “adaptive clocking”.

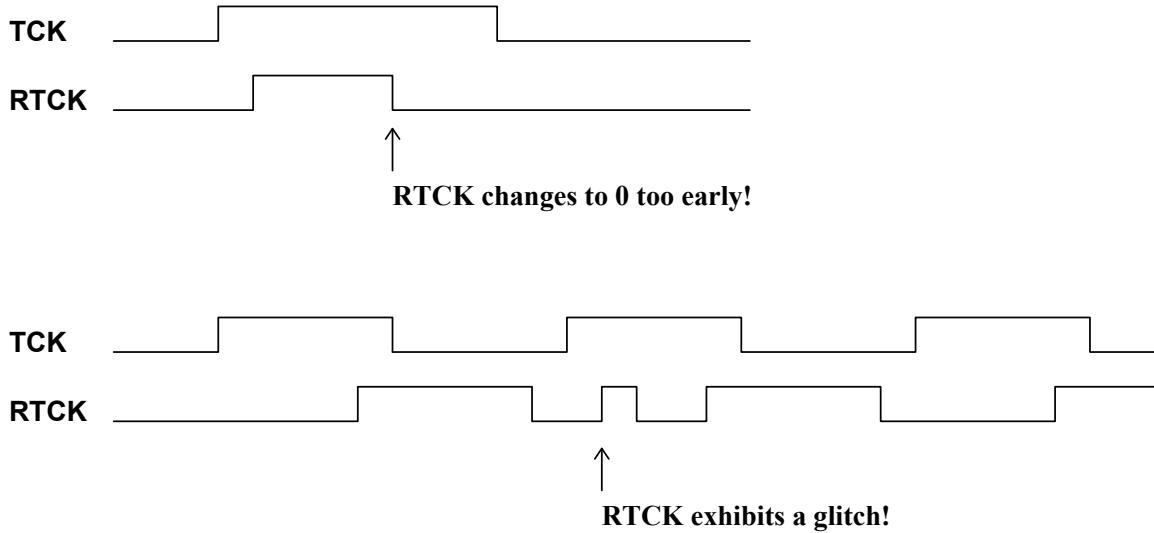
If the DTS follows this guideline, the RTCK should appear like a delayed copy of the TCK signal. The delay from the TCK to RTCK will vary, depending on the functional clock used by the TS as described in the introduction. If the DTS does not follow this guideline, then the exact behavior of the RTCK is unspecified. In this case, the RTCK might not be a delayed copy of the TCK.

Figure H-2 illustrates this behavior and shows how the TCK frequency will change dynamically, if the DTS follows the guidelines.



**Figure H-2 — Expected behavior for the RTCK and TCK**

It is expected that the RTCK follows the value of the TCK after some processor clock dependent delay. Figure H-3 illustrates unexpected and erroneous behavior.



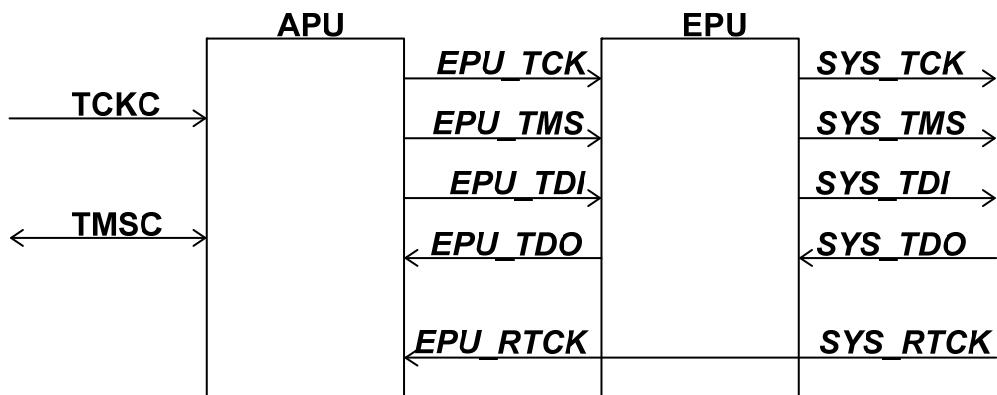
**Figure H-3 — Unexpected behavior for RTCK**

### H.3 Advanced Protocol mechanisms for RTCK support

As explained in the previous subclauses, the DTS is expected to wait until the TCK and RTCK have reached the same value, before the TCK transitions to the next value. So the RTCK signal can stall the TCK signal and thus the TAP controller state progression occurs.

In the Advanced Protocol, the IEEE 1149.7 TAP only uses the TCKC and TMSC signals for communication, but not the RTCK signal. The RTCK signal is only present as an internal signal (*sys\_rtck*) which is connected to the APU (see Figure H-4).

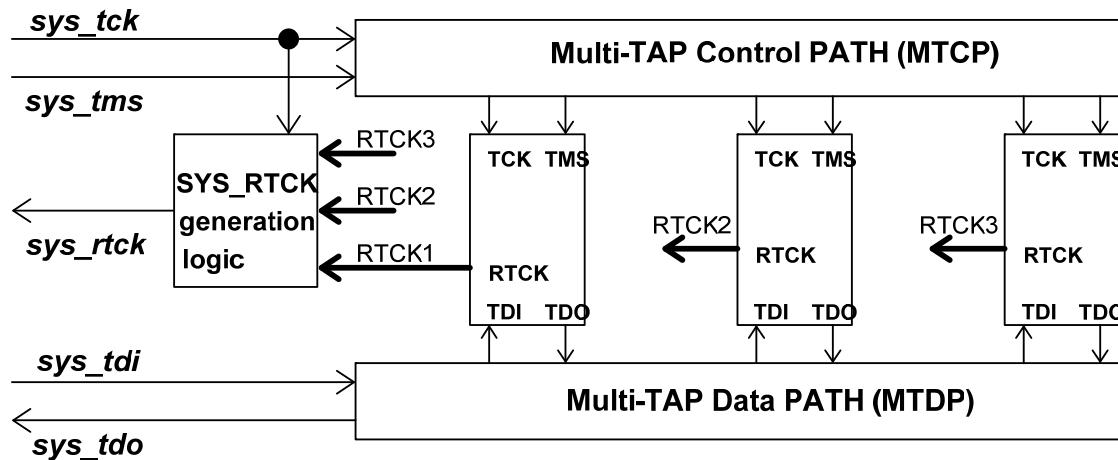
To emulate the behavior of an IEEE 1149.1 TAP with an additional RTCK signal, the Advanced Protocol offers several scan formats which include RDY bits. The RDY bits are employed to stall the completion of a Scan Packet to make sure the *sys\_tck* signal presented to the STL follows the guidelines for RTCK behavior.



**Figure H-4 — RTCK connected to an APU**

#### H.4 RTCK in multi-TAP environments

As mentioned in Clause 14, a device that implements an IEEE 1149.7 TAPC may contain several internal TAPs. If these TAPs additionally have the RTCK signals, it is necessary to generate a system-wide RTCK signal (*sys\_rtck*) which is either exported to a device pin or which is connected to an IEEE 1149.7 APU (see Figure H-5 and 23.14).

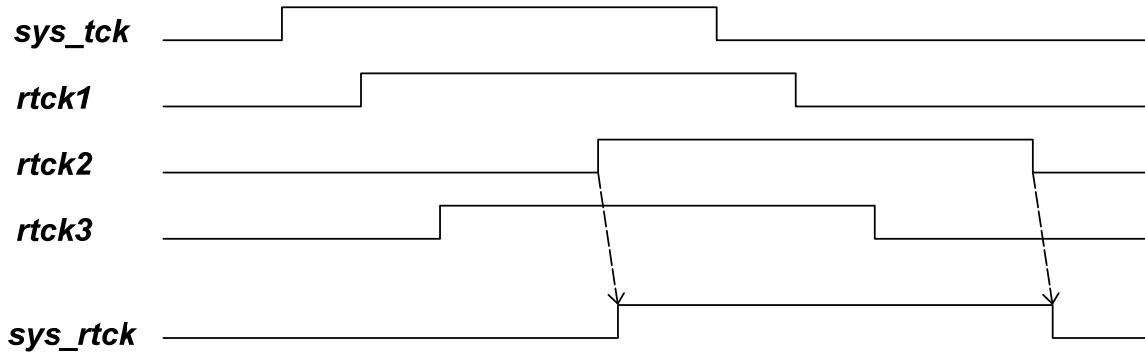


**Figure H-5 — RTCK generation for multiple TAPs**

It is the System Designer's responsibility to exercise the care necessary to ensure that the system-wide *sys\_rtck* signal follows the guidelines given in H.2. In principle, the TAP with the longest TCK-to-RTCK delay has to dictate the behavior of the system-wide *sys\_rtck* signal. The guidelines of H.2 can be extended in the following way:

- Each TCK edge is followed by a corresponding RTCK edge as described in H.2.
- When the *sys\_tck* signal transitions from a logic 0 to a logic 1, the *sys\_rtck* transitions to a logic 1, only after all individual RTCK signals have transitioned to a logic 1.
- When the *sys\_tck* signal transitions from a logic 1 to a logic 0, the system-wide *sys\_rtck* transitions to a logic 0, only after all individual RTCK signals have transitioned to a logic 0.

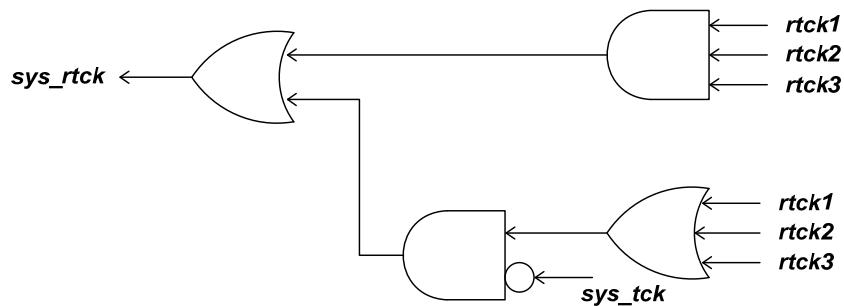
Figure H-6 illustrates these guidelines.



**RTCK2 dictates the behavior of the System-Wide RTCK**

**Figure H-6 — System-wide RTCK behavior**

Figure H-7 shows a circuit which implements the generation logic for a system-wide RTCK signal.



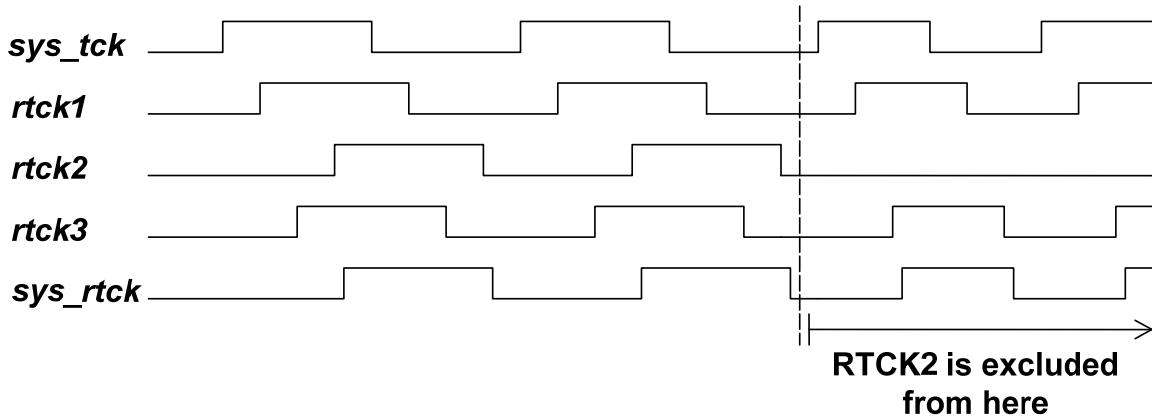
**Figure H-7 — System-wide RTCK generation logic**

Note the circuit of Figure H-7 functions correctly only if the *sys\_tck* signal follows the guidelines of H.2. This means the *sys\_tck* signal only changes its value when the *sys\_rtck* signal has reached the same value as the *sys\_tck* signal. It is also expected that a transition on the *sys\_tck* signal will reach the above circuit before any following transitions of the RTCK signals reach this circuit. (For example, assume the *sys\_tck* signal and all RTCK signals are a logic 0. When the *sys\_tck* signal transitions to a logic 1, it is expected the *sys\_tck* signal on the AND-gate transitions to a logic 1, before any of the RTCK signals transition to a logic 1).

When a device has the ability to modify the TAP chain under control of a Chip-Level TAP controller, the System Designer should exercise even more care to ensure the RTCK does not exhibit glitches or other kinds of unexpected behavior.

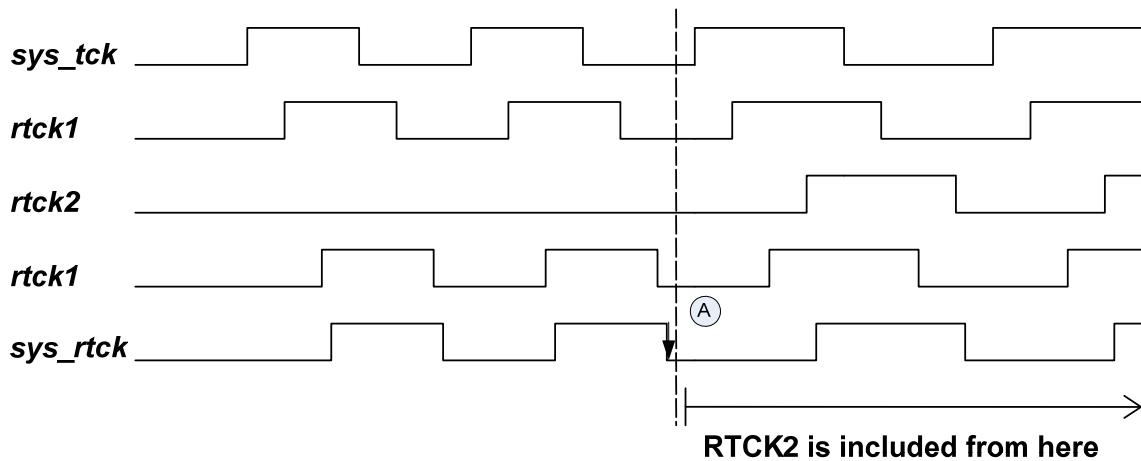
The problem is that modifying the TAP chain also modifies which RTCK signals have to be taken into account to generate the system-wide RTCK signal. If the logic which generates the system wide RTCK signal is not designed very carefully, then a change in the TAP chain will quite probably result in unexpected RTCK behavior.

Figure H-8 shows the expected behavior when the TAP responsible for RTCK2 is excluded. In this example, the exclusion of the TAPC2 is not complete until the falling edge of the *sys\_rtck* signal occurs. This falling edge of the *sys\_rtck* signal is dependent upon the final falling edge of RTCK2 (and the falling edges of the RTCK signals of all other TAPCs currently included – at A in this figure).



**Figure H-8 — Exclusion of TAP which outputs RTCK2**

Figure H-9 shows the expected behavior, when the TAP responsible for RTCK2 is included again.



**Figure H-9 — Inclusion of TAP which outputs RTCK2**

## H.5 RTCK consideration

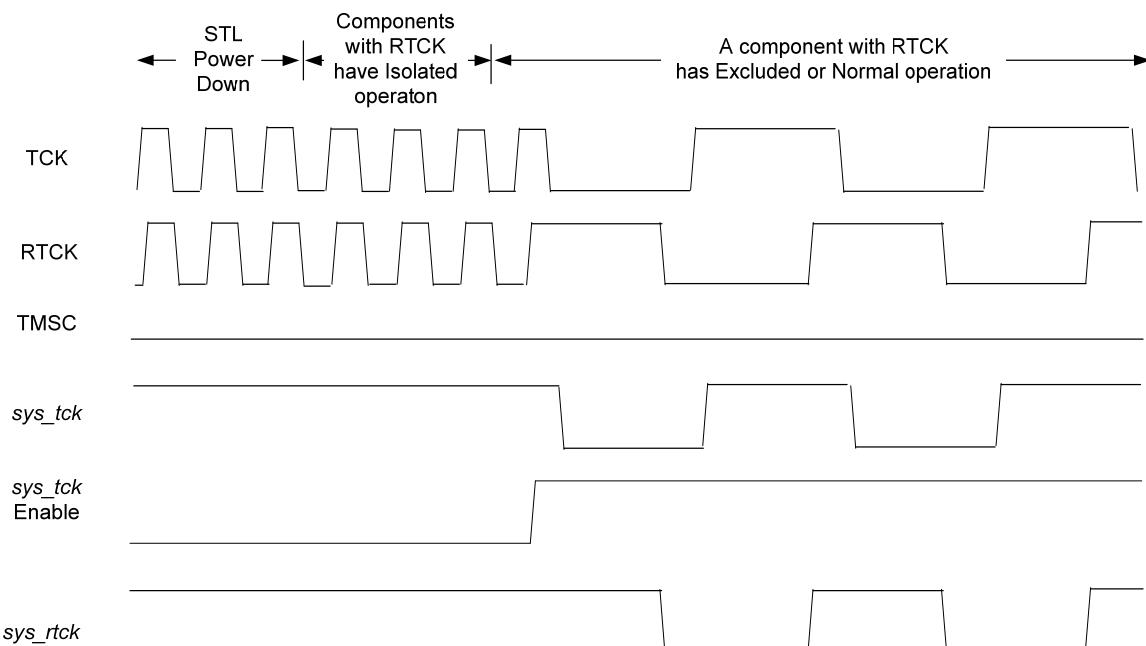
RTCK should be considered in generating the Chip-Level RTCK signal as follows:

When an STL component that includes the RTCK is powered-down, the RTCK generated by the component should be ignored when generating the *sys\_rtck* signal at the chip level.

- Component power-down      The RTCK generated by the component within the STL should be ignored when generating *sys\_rtck* at the chip level.
- Isolated                      The RTCK generated by the component within the STL should be ignored when generating *sys\_rtck* at the chip level
- Excluded/Normal            The RTCK generated by the component within the STL should be considered when generating *sys\_rtck* at the chip level.

When the STL is powered-down or all EMTAPCs within the STL with an RTCK are isolated, the RTCK merely returns the TCK throughout the power-down period. During this period, the RTCK pin merely

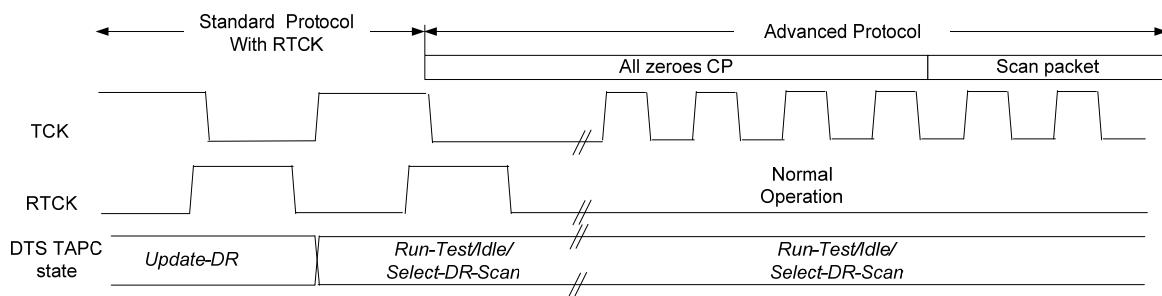
returns the TCK(C) signal. When the TAP.7 Controller is initialized, the TAP.7 Controller forwards the TCK(C) to the *sys\_tck* signal. At this point, the system-supplied RTCK (SYS\_RTCK) becomes the source for the RTCK signal as shown in Figure H-10.



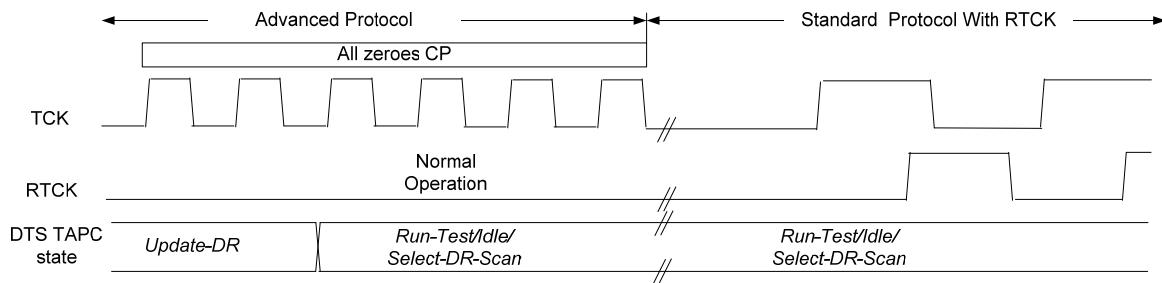
**Figure H-10 — RTCK behavior after a TAP.7 Controller power-up**

## H.6 RTCK behavior with Standard and Advanced Protocol changes

The behavior of the RTCK pin when changing from Standard to Advanced Protocol and from Advanced to Standard Protocol is shown in Figure H-11 and Figure H-12.



**Figure H-11 — RTCK behavior with Standard-to-Advanced Protocol change**



**Figure H-12 — RTCK behavior with Advanced-to-Standard Protocol change**

## H.7 Use of the RTCK signal as an auxiliary function

The function of an RTCK pin may be changed to an auxiliary function while using the Advanced Protocol as shown in Figure H-13.

Protocol	Standard Protocol	Advanced Protocol	Standard Protocol
RTCK Pin Function	RTCK	Alternate function	RTCK
Pin value	Return Clock	0	Alternate function
RTCK Pin Function Change Managed by DTS	← Driven with function shown →	→ [Yellow Box]	← Driven with function shown →

Note: The RTCK signal is driven to provide the RTCK function when using the JScan scan formats. An alternate function may be assigned to this signal only when using other scan formats.

**Figure H-13 — Use of the RTCK signal as an auxiliary function**

## Annex I

(informative)

### Bibliography

Bibliographical references are resources that provide additional or helpful material but do not need to be understood or used to implement this standard. Reference to these resources is made for informational use only.

- [B1] IEEE Std 1149.4™, IEEE Standard for a Mixed-Signal Test Bus.<sup>9, 10</sup>
- [B2] IEEE Std 1149.6™, IEEE Standard for Boundary-Scan Testing of Advanced Digital Networks.
- [B3] IEEE Std 1500™, IEEE Standard Testability Method for Embedded Core-based Integrated Circuits.
- [B4] IEEE Std 1532™, IEEE Standard for In-System Configuration of Programmable Devices.
- [B5] IEEE Std 1687™, IEEE Standard for Access and Control of Instrumentation Embedded within a Semiconductor Device.

---

<sup>9</sup> The IEEE standards or products referred to in Annex I are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated..

<sup>10</sup> IEEE publications are available from The Institute of Electrical and Electronics Engineers (<https://standards.ieee.org/>).

## Index

### 1

1149.7  
document  
    specifications, 70  
maximizing compatibility with 1149.1 IP, 63  
overview, 56  
1149.7 standard  
abbreviations, 86  
acronyms, 86  
challenges undertaken, 58  
contrasted to 1149.1, 57  
conventions, 75  
definitions, 82  
document  
    content, 56, 69  
    organization, 70  
flexibility, 67  
important considerations, 59  
normative references, 81  
scalability, 65  
use of the, 74

### A

ADTAPC  
Adapter TAPC state machine, 264  
need for, 264  
advanced interfaces  
    EPU  
        TDI signal creation, 599  
        TMS signal creation, 597  
    STL  
        system ready treatment, 601  
        TDI signal creation, 599  
        TDO values, 603  
        TMS signal creation, 597  
advanced protocol  
    effects on the EPU/CLTAPC relationship, 604  
    SSD detection, 604, 605  
advancing the TAPC state  
    SScan scan formats, 677  
alerts  
    deselection alert  
        operation, 262  
    selection alert  
        bit sequence, 257  
        detection, 258  
        offline-at-startup operation, 275  
        overview, 256  
alerts. *see* RSU, ancillary services, alerts  
architecture  
    advanced, 542  
    TAPC hierarchy, 92

### B

behavior  
    start-up behavior, 501, 586, 701  
bibliography, Annex I, 1035  
bit-frame(s)  
    input bit-frame scan formats, 633  
block diagram  
    high-level  
        T0 TAP.7, 126  
        T1 TAP.7, 137  
        T2 TAP.7, 143  
        T4 and above TAP.7, 542  
        T5 TAP.7, 174  
BSDL.7  
applications of  
    typical, 803  
components, 790  
descriptions  
    compliance enable, 799  
    configuration register, 801  
    device identification register, 800  
entity, 790  
examples, 799, 800, 801  
general information, 787  
lexical elements, 789  
parser expectations, 788  
purpose, 787  
relationships  
    to BSDL.1, 788  
reserved words, 789  
scope, 787  
statements  
    component conformance, 794  
    examples, 793, 795, 796  
    scan port identification, 795  
    standard use statement, 792  
    superior use, 794  
The Standard BSDL.7 Package STD\_1149\_7\_2009,  
description and specifications, 803  
*BYPASS* instruction  
    adding functionality to, 128  
    suitability for extended control, 429

### C

capabilities. *see* features  
advanced  
    comparison of std./adv. protocols, 545  
interoperability with  
    T0 - T3 TAP.7s, 544  
    T4 and above TAP.7s, 544  
mandatory, 543  
optional, 543  
inherited  
    T1 class, 442

T2 class, 480  
 T3 class, 498  
 T4 class, 580  
 T5 class, 692

new  
 T1 class, 443, *see features, T1 class*  
 T2 class, 480  
 T3 class, 498  
 T4 class, 580  
 T5 class, 692

T0 class, 405  
 T0 TAP.7, 124  
 T1 class, 442  
 T1 TAP.7, 128  
 T2 class, 480  
 T2 TAP.7, 139  
 T3 TAP.7, 145  
 T4 TAP.7, 580  
 T5 TAP.7, 692

check packet  
 directives, 286  
 function, 285

command  
 name  
   controller ID allocate (CIDA), 202, 499, 504  
   controller ID deallocate (CIDD), 202, 499  
   extended command 0 (EXC0), 202  
   extended command 1 (EXC1), 202  
   extended command 2 (EXC2), 202  
   extended command 3 (EXC3), 202  
   make conditional group member (MCM), 202  
   make scan group candidate (MSC), 202  
   scan bit (SCNB), 202, 499  
   scan string (SCNS), 202  
   store conditional one-bit (STC1), 202, 581  
   store conditional two-bit (STC2), 202, 499, 581  
   store format (STFMT), 202, 499, 581  
   store miscellaneous control (STMC), 202, 499,  
     581, 693  
   store test mode (STTESTM), 202  
   store transport state (STTPST), 202, 693

portfolio  
 T1 class, 443  
 T2 class, 481  
 T3 class, 498  
 T4 class, 581  
 T5 class, 692

commands  
 basics, 197  
 introduction to, 130  
 portfolio, 202  
 representation in examples, 209  
 reset  
   effects on targeted commands, 203, 238

three-part  
 CIDA, 223  
 overview, 221  
 SCNB, 221  
 SCNS, 222

two-part, 221

type  
 enumerate, 203

private, 203  
 scan, 203  
 select, 203  
 store, 202

compatibility with 1149.1 IP  
 boundary-scan capability, 63  
 infrastructure, 63  
 instructions, 63  
 scan paths, 63

concepts  
 1149.1 compliance  
 background, 401  
 test and debug views of a system, 402

advanced  
 architecture, 542  
 CID allocation, 617, 640, 681  
 Configuration Faults, 592, 701  
 functions, 545  
   bypass, 546  
   conceptual view, 545  
   interactions, 550  
   scan, 547  
   transport, 548

interfaces, 550  
 APU/EPU, 551  
 DCC, 551  
 TAP/APU, 550

packet  
 sequence construction, 559

packets(s). *see packet(s)*

pipelined transactions, 577

relationships  
 TAPC state/packet, 557

view of the adv.protocol  
 implementer's, 562  
 user's, 562

APU  
 state diagram, 575

extended  
 CID allocation, 504

operating states. *see operating states*

system  
 components, chips, and boards, 182  
 connectivity  
   with the TAPC hierarchy, 181  
 deployment of  
   TAP.1 branches, 179  
   TAP.7 branches, 179

key attributes, 177

overview of system concepts, 177

supporting  
 a mix of technologies, 177  
 a mix of technologies, 69  
 technology branches, 178  
 various scan topologies, 67

TAPC hierarchy, 180

test, 777

divergences  
 accommodation/resolution of, 781  
 versus 1149.1, 780

documentation model, 783

example by narrative, 781

- interoperability, 777
- scan-state sequencing, 779
- system considerations
  - large, 784
  - small, 784
- topology, 779
- unit under test
  - construction, 778
  - description, 782
- concepts introduction
  - capability
    - private commands and registers, 115
    - resets, 115
  - pin efficiency
    - advanced protocol, 108
    - control protocol, 111
    - interleaving of scan/non-scan information, 109
    - performance, 113
    - protocols, 107
    - serialization of scan information, 110
    - signaling methods, 106
    - standard protocol, 108
  - system architecture
    - direct addressability for star scan topologies, 102
    - maximizing 1149.1 hardware compatibility, 92
    - maximizing 1149.1 software compatibility, 92
    - operation with series and star scan topologies, 101
    - output-drive characteristics with a star scan topology, 104
    - parking the TAPC state at TAPC hierarchy levels, 94
    - power management, 100
    - scan formats, 105
    - scan interoperability of feature sets, 106
    - scan topology training, 102
    - series equivalent scans for star scan topologies, 103
    - sharing of TCK(C) and TMS(C) signaling with other technologies, 102
    - system and control paths, 99
    - TAP.7 controller management, 98
    - TAPC hierarchy, 92
  - Configuration Faults, 592, 701
  - configuration(s)
    - T0 class, 405
    - T1 class, 447
    - T2 class, 483
    - T3 TAP.7, 500
    - T4 TAP.7, 585
    - T5 TAP.7, 699
  - connectivity
    - connectivity/electrical recommendations, Annex F, 933
  - control
    - control segments scan formats, 677
  - control levels
    - exiting, 130
    - introduction to, 129
  - control path
    - introduction to, 99
  - use of with a T1 TAP.7, 136
- use of with a T2 TAP.7, 141
- control state machine
  - mandatory and optional behaviors, 290
  - states
    - ADV*, 293
    - CHK*, 302
    - examples, 303
    - OLS*, 308
    - OLW*, 294
    - STD*, 292
    - sub-states, 302
    - TEST*, 295
- controller ID
  - alias to TCA, 504
  - allocation
    - AT generation
      - with JScan3, 506
    - candidates, 505
    - criteria, 505
    - directed, 506
    - examples, 507
    - process, 505
    - undirected, 506
- CP
  - SP followed by a CP scan formats, 677
- D

  - data segments
    - scan formats, 677
  - direct addressability
    - TAP controller address, 502
    - aliasing to controller ID, 504
  - directive/register/operational relationships, 724
  - DR scans
    - zero-bit, 128
  - drive policy
    - TDO(C)
      - class/component relationships, 345
      - conceptual view, 350
      - drive enables, 344
      - drive types, 341, 384
      - factors affecting, 343
      - flattened view, 349
      - hierarchical view, 348
      - overview, 341
      - policy components, 344
        - dormant, 345
        - series command, 346
        - series control level, 346
        - series system, 345
        - star-4 command, 346
        - star-4 control level, 347
        - star-4 system, 346
        - transition policy, 345
      - summary, 375
    - T0 TAP.7, 351
    - T1 TAP.7, 352
    - T2 TAP.7, 352
    - T3 and above TAP.7, 354
  - TMSC

component policies  
 inhibiting, 386  
 PC0/PC1, 386  
 RDY, 387  
 TDO, 390  
 transport, 394

drive types  
 joint, 385  
 normal, 385  
 voting, 385

influences  
 system path, 391

output bit types, 380

relationship to TCKC, 382

scan packet content, 380

transport packet content, 381

drive types  
 scan formats, 633

**E**

electrical  
 connectivity/electrical recommendations, Annex F, 933

embedded TAPCs  
 architecture for control, 406  
 control of, 407  
 use cases, 418

DR-control  
 exclusions, 420  
 isolation, 420

IR-control  
 exclusions, 419  
 isolation, 419

with CLTAPC data register(s), 414

with CLTAPC instruction register, 410  
 isolation, 412  
 output control, 413

with CLTAPC instruction register(s)  
 exclusion, 410

with *tapc\_select* signals  
 external, 416  
 internal, 416

identification of, 421

escape sequences  
 detection, 251  
 overview, 250

types  
 custom  
 operation, 252

deselection  
 context sensitive reponse, 273  
 operation, 252  
 qualification criteria when online, 274

reset  
 operation, 253

selection  
 context sensitive reponse, 273  
 operation, 252  
 qualification criteria  
 when offline-at-start-up, 274

when online, 274

escapes. *see* RSU, ancillary services, escapes

example(s)  
 scan examples in timing diagram form, Annex B, 834

**F**

features  
 RSU  
 signal sharing, 102  
 summary, 175

T1 class  
 functional reset, 451  
 power control, 455  
 test reset, 449

T2 class  
 hot connect protection, 483

T3 class  
 Aliasing the TAP controller address, 504  
 direct addressability, 102, 502  
 scan topology training, 102, 531  
 series equivalent scans, 520

T4 class  
 sample TMSC with rising/falling TCKC edge, 596

functional reset. *see* features, T1 class

functions. *see* concepts, advanced, functions

**G**

group membership  
 EPU groups  
 conditional group membership effects, 448  
 count (CGMC), 371  
 only member determination, 372  
 tracking membership, 371

STL groups  
 changes with  
*Pause-DR*, 358  
*Pause-IR*, 358  
*Run-Test/Idle*, 357  
*Test-Logic-Reset*, 357

effects  
 commands, 358  
 paths, 485  
 reset, 485

SSDs  
 in *Pause-DR*, 359  
 in *Pause-IR*, 359  
 in *Run-Test/Idle*, 359

TAPC state, 485

factors affecting  
 T2 class, 484  
 T3 and above classes, 536

only scan group member determination  
 cases, 369  
 criteria, 360  
 information used, 360

potential scan group candidate count last (PSGMCL), 364  
scan group candidate count(SGCC), 362  
SGCC and PSGMCL ambiguity, 365  
tracking membership, 357  
groups  
EPU  
    types, 137  
STL, 142  
    candidacy and memship counts, 362  
    determining only group member, 378  
    movement between, 143  
    *Pause-IR* and *Pause-DR*, 147  
    types, 142

## H

HSDL.7  
    applications of  
        examples, 822, 824, 825, 826, 828  
    components, 809  
    declarations  
        examples, 820  
        module members, 818  
    descriptions  
        entity, 810  
    general information, 806  
    lexical elements, 809  
    mappings  
        examples, 815  
        module package pin, 815  
    parser expectations, 808  
    purpose, 807  
    relationships  
        to BSDL.1, 808  
        to BSDL.7, 808  
    reserved words, 809  
    scope, 807  
    statements  
        examples, 813, 814, 816  
        module component conformance, 814  
        module scan port identification, 815  
        module standard use, 811  
        module superior use, 813  
The Standard HSDL.7 Package  
    STD\_1149\_7\_2009\_module, 822

I  
*IDCODE* instruction  
    adding functionality to, 128  
    suitability for extended control, 429  
implementation approaches  
    ADTAPC, 264  
    alerts - selection, 258  
    CID allocation, 508  
    CLTAPC selection  
        T2 class, 489  
        T3 and above classes, 536  
    CSM and SSM activation, 644

EMTAPC selection/deselection, 403  
EPU operating states, 436  
escape-sequence detection, 253  
maximizing TAP.7 controller performance, 605  
MScan, 619  
OScan, 641  
OScan scan formats, 641  
power control, 471  
reset, 239  
scheduling  
    APU operating states, 563  
    packets, 577  
    SP elements, 577  
        the SP elements, 578  
SScan, 682  
SSDs, 525  
TDO(C) drive policy, 376  
transport state machine, 763  
inherited capabilities  
    T3 TAP.7, 498  
    T4 TAP.7, 580  
    T5 TAP.7, 692  
input bit-frame  
    input bit-frame scan formats, 633  
interfaces  
    APU, 550, 551  
    APU/EPU, 551  
interoperability. *see*  
    capabilities,advanced,interoperability

## M

MScan  
    applications types supported, 607  
    CID allocation, 617  
    high-level operation, 608  
    important characteristics, 608  
    primary purpose, 607  
    scan packet content, 609  
multiple dies in one package, 422  
    BSDL documentation, 423  
    DR wire bypass, 425  
    exposing boundary scan, 423  
    packaging the dies, 424  
    properly exposing the DR paths, 423  
    using por for proper operation, 425  
multiple TAPCs on-chip. *see* embedded TAPCs

## N

new capabilities  
    T3 TAP.7, 498  
    T4 TAP.7, 580  
    T5 TAP.7, 692  
nomenclature – 1149.7  
    describing the test and debug architecture, 59  
    terminology describing scan exchanges, 61  
    terminology describing TAP behaviors, 59  
    terminology describing TAP signals, 61

terminology describing TAPs and TAP controllers,  
60

## O

offline  
 offline-at-start-up, 308  
 CLTAPC state initialization, 308  
 initialization/selection qualification, 309  
 placement online, 308  
 selection qualification, 309  
 offline. *see* online/offline operation  
 online. *see* online/offline operation  
 online/offline operation  
   initiating offline operation, 271  
   managing, 266  
   offline operation  
     events initiating  
     overview, 271  
   online operation  
     events initiating  
     overview, 273  
     use of an unsupported feature, 272  
   operating principles, 267  
 operating characteristics  
   T1 TAP.7  
     commands, 130  
     EPU operating states, 130  
     registers, 133  
   T2 TAP.7  
     groups of STLS and EPUs, 137, 142  
   T3 TAP.7  
     addressability in a star-4 scan topology, 147  
     Pause-IR and Pause-DR groups, 147  
     within the series and star-4 scan topologies, 147  
 operating modes  
   T0 TAP.7, 124  
   T1 TAP.7, 128  
   T2 TAP.7, 139  
   T3 TAP.7, 145  
 operating states  
   APU  
     changes, 554  
     characteristics, 553  
     introduction to, 173  
     relationships with functions, 553  
   EPU  
     changes, 434  
     characteristics, 434  
     description of, 433  
     introduction to, 130  
     relationships with functions, 433  
     ZBS use compatible with, 435  
 operational overview  
   T0 TAP.7, 123  
   T1 TAP.7, 127  
   T2 TAP.7, 137  
   T3 TAP.7, 144  
 OScan  
   applications types supported, 624  
   CID allocation, 640

high level operation, 625  
 important characteristics, 625  
 primary purpose, 624  
 scan packet content, 626

## P

package  
 multiple dies in, 422  
 packet(s)  
   check, 277, 567  
   format, 284  
   combinations of, 560  
 format  
   check, 284  
   scan, 568  
   transport, 573  
 relationships with  
   TAPC state, 560  
 scan, 567  
   advance of the TAPC state, 572  
   conceptual view, 567  
   content, 569  
   elements, 569  
     header, 569  
     payload, 569  
   format, 568  
   output transaction types  
     initiator/responder, 571  
     scripted, 571  
     voting, 571  
 sequences  
   construction of, 559  
   scheduling of, 560  
 TAPC  
   state relationships, 557  
 transport, 573  
   conceptual view, 573  
   content, 575  
   format, 573  
 view of their operation  
   implementer's, 562  
   users's, 562  
 parent/child relationships. *see* TAPC hierarchy  
 parking-state considerations  
   ADTAPC, 287  
 performance  
   factors impacting, 113  
   increasing STL performance, 593, 619, 640, 682  
 power control. *see* features, T1 class  
   chip-level power manager's  
     handling of power-up/power-down requests, 461  
     interaction with TAP.7 controller, 460  
     responsibilites, 460  
     role, 460  
     type-0 reset assertion, 461  
   chip-level power-up/power-down enables, 462  
   default power-control mode, 462  
 DTS's  
   detection of power-up, 464  
   direction of power-up, 463

responsibilities, 463  
 role, 463  
 example sequences, 469  
 management within a typical system, 457  
 model, 458  
     key attributes, 459  
     power management states, 458  
 options, 456  
 overview, 455  
 TAP.7 controller's  
     awaiting power-down, 468  
     initiation test, 467  
     operation, 465  
     power-up confirmation test, 467  
     power-up request summary, 468  
     responsibilities, 465  
     role, 465  
     test periods, 466  
 topics, 458  
 use cases, 456  
 programming considerations  
     aspects of transport not covered, 727  
     common and uncommon operations, 725  
 drive conflict prevention  
     at start-up, 379, 399  
 escape sequences, 253  
 offline/online operation, 313  
 selection alerts, 263  
 T0 class, 428  
 T1 class, 477  
 T2 class, 494  
 T3 class, 541  
 transport, 725  
     bandwidth allocation, 725  
     dynamic source/destination changes, 726  
     managing with the DTS, 725  
     transfer alignment characteristics, 727  
 use of the EPU operating states, 440  
 use of the TAPC  
     hierarchy  
         with test applications, 195  
         with test/debug applications, 194  
 using the TAPC  
     hierarchy  
         selection/start-up considerations, 193  
 protocols  
     characteristics, 111  
     introduction to types, 107

**R**

recommendations  
 connectivity/electrical recommendations, Annex F, 933  
 register(s)  
     introduction to, 133  
     loading  
         global and local registers with commands, 209  
         global registers with a Selection Sequence, 283  
         methods, 199  
 name

aux. pin func. cntl (AFPC), 581  
 configuration 0 (CNFG0), 227  
 configuration 1 (CNFG1), 227  
 configuration 2 (CNFG2), 227  
 configuration 3 (CNFG3), 227  
 controller ID (CID), 499, 504  
 controller ID invalid (CIDI), 499, 504  
 data element length (TP\_DELN), 694  
 delay control (DLYC), 581, 613  
 exit control level (ECL), 438  
 functional reset (FRESET), 451  
 functional reset return (FRESETR), 451  
 PDC/LDC association (PDCx\_LCA), 693, 694  
 physical data channel DCC control registers  
     (PDCx\_DCCy\_CRz), 693, 694  
 physical data channel DCC selection  
     (PDCx\_DCC), 693, 694  
 physical data channel selected (PDCx\_SEL), 693, 694  
 power mode (PWRMODE), 455  
 read back 0 (RDBACK0), 226  
 read back 1 (RDBACK1), 226  
 ready control (RDYC), 581, 611, 634  
 sampling rising edge (SREDGE), 581, 596  
 scan format (SCNFMT), 484, 499, 501, 587  
 scan group candidate (SGC), 484  
 SSD enable (SSDE), 499, 514  
 suspend (SUSPEND), 434, 438  
 sys\_tck duty cycle (STCKDC), 581, 593, 619, 640, 682  
 test reset (TRESET), 449  
 topology (TOPOLOGY), 499, 531, 536  
 transport data element length (TP\_DELN), 693  
 transport protocol revision (TPPREV), 693, 694  
 transport protocol revision (TPREV), 693  
 transport state (TPST), 693  
 ZBS inhibit (ZBSINH), 434, 438  
 portfolio  
     T1 class, 443  
     T2 class, 481  
     T3 class, 498  
     T4 class, 581  
     T5 class, 692  
 resets  
     considerations, 236  
     directives creating, 286  
     effects, 236  
     effects on controller ID, 238  
     effects on targeted commands, 203, 238  
     introduction to, 118  
     operation assured after power-up, 238  
     overview, 235  
     type-0 vs. type -2, 238  
     type-0, 238  
     type-1, 238  
     type-2, 238  
     type-3, 237  
     type-4, 237  
     type-5, 237  
 RSU  
     ancillary services  
         alerts, 256, 262

escapes, 250  
 overview, 235  
 resets, 235  
 start-up options, 242  
 operation  
   T0 class, 427  
   T1 class, 476  
   T2 class, 493  
   T3 class, 540  
 RTCK signal, Annex H, 1027

**S**

scan examples  
   timing diagram form, Annex B, 834  
 scan format(s)  
   applications types supported  
     MScan, 607  
     OScan, 624  
     SScan, 647  
   capabilities  
     JScan, 105  
     MScan, 607  
     OScan, 624  
     SScan, 646  
   CID allocation  
     JScan3, 504  
     MScan, 617  
     OScan, 640  
     SScan, 681  
   comparison of advanced, 590  
     MScan, 590  
     OScan, 590  
   Configuration Faults, 592  
   high level operation  
     MScan, 608  
     OScan, 625  
     SScan, 650  
   important characteristics  
     MScan, 608  
     OScan, 625  
     SScan, 650  
   increasing STL performance, 593, 619, 640, 682  
   influences on definition, 589  
   overview of advanced, 587, 589  
   performance/flexibility tradeoffs, 589  
   primary purpose  
     MScan, 607  
     OScan, 624  
     SScan, 646  
   scan packet  
     content  
       MScan, 609  
       OScan, 626, 627  
       SScan, 655  
     delay element, 613, 636, 673  
       directives, 614  
       format, 613  
       uses, 614  
     format  
       OScan, 627  
   MScan  
   content, 609  
   format, 609  
   RDY bits, 611  
   relationship to EPU signals, 610  
   optimizations  
     OScan, 628  
     SScan, 660  
   OScan  
     content, 627  
     format, 627  
     optimizations, 628  
     RDY bits, 634  
     relationship to EPU signals, 629  
   relationship to EPU signals  
     MScan, 610  
     OScan, 629  
   TAPC state advance  
     MScan, 616  
     OScan, 637  
   T2 class, 484  
   T3 class, 501  
   T4 class, 587  
   T5 class, 587  
   scan packet  
     content  
       preview, 380  
   scan paths  
     physical  
       EPU  
       path selection and use, 214  
       scan-path continuity, 215  
     type  
       auxiliary, 219  
       bit, 216  
       bypass, 216  
       enumerate, 219  
       string, 216  
       types and characteristics, 215  
   scan selection directives  
     effects  
       on STL group membership, 513  
       examples of use, 520  
     format, 513  
     overview, 512  
     processing  
       conditional, 517  
       details, 514  
       enabling, 514  
       record of last processed, 517  
     relationships  
       with  
         control level, 517  
         deselection of ADTAPC, 517  
     relationships with  
       ZBS count, 517  
   SSD state machine, 517  
   states  
       allowing scan group membership, 519  
   scan topology training sequence  
     characteristics used to determine topology, 533  
     command sequence, 534

example  
     with a single TAP.7 branch, 532  
 overview, 531  
     topology register function, 531  
 segment(s)  
     control segments scan formats, 677  
     data segments scan formats, 677  
 selection  
     concepts. *see* TAPC hierarchy  
 selection of TAPCs. *see* TAPC hierarchy  
     overview of concepts, 184  
 selection sequence  
     drive policy considerations, 296  
     extension code, 281  
     format, 276  
     global register load  
         format, 283  
         required for placement online, 296  
     initiation, 276  
     online activation code, 278  
     placement online, 276  
     requiring a state load, 298  
     selection test, 295  
     TAP.7 components, 277  
 selection/deselection of TAPCs. *see* TAPC hierarchy  
 signal(s)  
     RTCK signal, Annex H, 1027  
 SP  
     SP followed by a CP scan formats, 677  
 SScan scan formats  
     applications types supported, 647  
     CID allocation, 681  
     factors determining payload content, 658  
     factors influencing packet sequencing, 674  
     header element, 657  
     high-level operation, 650  
     important characteristics, 650  
     input bit-frames, 663  
     optimizations, 660  
     output bit-frame content, 666  
     payload element, 658  
     primary purpose, 646  
     scan packet content, 655  
 segments  
     control, 648  
     data, 649  
     use of, 651  
     use of with a  
         buffered TDI/TMS component, 653  
         data-rate dependent component, 653  
         DMA/FIFO component, 652  
         TAP.1 component, 651  
         two applications types, 654  
     stall profiles, 649  
     utilizing, 1023  
 SScan0/1  
     output-only segments, 666  
 SScan2/3  
     output-only segments, 667  
 standard's  
     applicability, 337  
 start-up behavior  
     T0 class, 406  
     T1 class, 448  
     T2 class, 483  
     T3 TAP.7, 501  
     T4 TAP.7, 586  
     T5 TAP.7, 701  
 start-up options  
     introduction to, 119  
     overview, 242  
     start-up functionality, 245  
 type  
     1149.1-compatible, 243  
     1149.1-compliant, 243  
     1149.1-protocol compatible, 244  
     offline-at-start-up, 244  
 STL  
     groups, 142, 143  
         *Pause-IR* and *Pause-DR*, 147  
 summary  
     TAP.7 features, 175  
 system  
     concepts. *see* concepts, system  
 system path  
     introduction to, 99  
     use of with a T1 TAP.7, 136  
     use of with a T2 TAP.7, 141

**T**

T0 TAP.7  
     high-level block diagram, 126, 542  
     operation  
         multi-TAPC architecture, 125  
         overview of, 125  
 T1 TAP.7  
     high-level block diagram, 137  
     operation  
         overview of, 128  
 T2 TAP.7  
     high-level block diagram, 143  
     operation  
         overview of, 140  
 T3 TAP.7  
     operation  
         overview of, 147  
 T5 TAP.7  
     high-level block diagram, 174  
 TAP  
     auxiliary pin functions, 595  
     connections  
         with RSU, 336  
         without RSU, 336  
     deployment  
         T0 class, 404  
         T1 class, 442  
         T2 class, 480  
         T3 class, 497  
         T4 class, 579  
         T5 class, 691  
     interoperability recommendations

- 1149.7-Non-disruptive behavior, 338
- other behavior, 339
- overview, 338
- power-up behavior, 338
- performance
  - advanced protocol signal timing, 113
  - efficiency of scan transfers, 114
- signal/class relationships, 67
- signals
  - behavior
    - offline-at-start-up
      - general, 335
      - TMS/TMSC, 324
    - online-at-start-up
      - TMS/TMSC, 323
    - state machine representation
      - TMS/TMSC, 326
  - class relationships, 317
  - function
    - legacy, 322
    - nTRST/nTRST\_PD, 321
    - TCK/TCKC, 322
    - TDI/TDIC, 330
    - TDO/TDOC, 333
    - TMS/TMSC, 323
  - I/O structure and bias, 319
  - performance
    - registering of signals, 114
- TCK edge use to sample TMSC, 596
- transport interface, 728
- TAP.7
  - feature summary, 175
- TAP.7 architecture
  - components, 115
  - operating models, 120
    - advanced, 122
    - standard, 120
  - reset types, 118
  - start-up options, 119
- TAP.7 concepts and architecture
  - introduction, 91
- TAPC
  - hierarchy
    - concurrent CLTAPC/EMTAPC selection
      - changes, 488
    - operation
      - ADTAPC, 188
      - CLTAPC, 189
      - EMTAPC, 191
    - overview of, 184
    - parking use cases
      - ADTAPC, 187
      - CLTAPC, 189
      - EMTAPC, 191
    - relationships
      - parent/child, 185
        - ADTAPC/CLTAPC, 189
        - CLTAPC/EMTAPC, 191
        - DTS/ADTAPC, 187
    - RSU deployment, 192
    - selection/deselection
      - choice, 184
  - effects on
    - scan path behavior, 185
    - TAPC behavior, 185
  - mechanisms, 186
  - of the
    - ADTAPC, 187
    - CLTAPC, 189
    - EMTAPC, 191
  - relationships with
    - class, 185
    - parked state, 487
  - shared protocol across technologies, 192
  - within the hierarchy, 184
  - supporting
    - a mix of technologies, 177
    - a mix of technologies, 69
    - various scan topologies, 67
  - state
    - advance of by SPs, 572
    - packet relationships, 557
  - TAPCs
    - multiple embedded. *see* embedded TAPCs
  - TDO drive policy
    - introduction to T1 TAP.7, 137
    - T2 TAP.7
      - introduction to, 142
  - test clock
    - shared or dedicated use, 64
    - source, 63
    - treatment, 63
    - unorthodox use, 64
  - test reset. *see* features, T1 class
  - timing
    - timing diagrams
      - scan examples in timing diagram form, Annex B, 834
    - transparency to 1149.1 IP
      - constraints, 62
      - constraints/requirement balancing, 62
      - upgrade path, 62
  - transport
    - activity examples, 772
    - building blocks, 722
    - Configuration Faults, 701
    - data element(s), 716
      - alignment with transported data, 738
      - content, 736
      - data utilization/generation, 717
      - direction, 716
      - format/timing/drive characteristics, 734
      - length, 716, 735
    - data transfer(s)
      - multi-client, 737
    - DCC functions, 720
      - client initialization, 720
      - orderly shut-down of a transfer, 720
    - DCC interface operation, 760, 762
    - directive element
      - characteristics, 729
    - directive elements, 704
      - building block relationships, 706
      - encoding, 707

- list, 705
- surrounding context, 705
- types, 708
  - conditional, 710
  - reset, 709
  - transfer, 711
  - unconditional, 709
- effects of online/offline operation, 721
- enabling, 702
- operation with single and multiple clients, 717
- packet
  - completion, 744
  - composition, 703
- partitioning of the control function, 722
- PDCx/DCC interface
  - description and specifications, 746
  - signal functions, 747
  - signal use, 749
- protocol, 693
- register element
  - characteristics, 731
  - content, 732
  - format, timing, and drive, 731
  - length, 731
- register elements, 715
  - length, 715
  - transfer direction, 715
- scheduling a transport packet, 742
- selection and control of data targets, 719
- starting/restarting directive processing, 744
- state machine
  - description and specification, 739
- states
  - specifying, 694
- register elements, 715
- TPA* state flow, 728
- transport
  - eight-bit directives, 755
  - transport
    - 12-bit directives, 757
    - transport
      - pipelining options, 761
      - transport five-bit directives, 753
      - transport packet. *see* packet(s), transport content
        - preview, 381
  - type(s)
    - drive types scan formats, 633

## Z

- ZBS
- count
  - clearing of, 430
  - functionality associated with, 439
  - incrementing of, 430
  - locking of, 129, 430
  - zeroing, 130
- detection, 429
- introduction to, 128
- shared use by the EPU and STL, 433
- use compatible with operating states, 435
- utilizing for TAP.7 controller functionality, 129

# RAISING THE WORLD'S STANDARDS

---

## Connect with us on:

-  **Twitter:** [twitter.com/ieeesa](https://twitter.com/ieeesa)
-  **Facebook:** [facebook.com/ieeesa](https://facebook.com/ieeesa)
-  **LinkedIn:** [linkedin.com/groups/1791118](https://linkedin.com/groups/1791118)
-  **Beyond Standards blog:** [beyondstandards.ieee.org](https://beyondstandards.ieee.org)
-  **YouTube:** [youtube.com/ieeesa](https://youtube.com/ieeesa)

[standards.ieee.org](https://standards.ieee.org)  
Phone: +1 732 981 0060

