

IEEE Guide for Digital Test Interchange Format (DTIF) Application

Sponsor

IEEE Standards Coordinating Committee 20
on
Test and Diagnosis for Electronic Systems

Reaffirmed 7 December 2011
Approved 21 September 2000

IEEE-SA Standards Board

Abstract: An aid in the understanding and use of digital test interchange format (DTIF) files is provided in this guide. This information will be an aid to users in developing tools such as preprocessors and postprocessors of DTIF data and other utilities.

Keywords: automatic test equipment (ATE), digital automatic test program generator, digital test interchange format (DTIF), unit under test (UUT)

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

Copyright © 2001 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 23 March 2001. Printed in the United States of America.

Print: ISBN 0-7381-2627-6 SH94886
PDF: ISBN 0-7381-2628-4 SS94886

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

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

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

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

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

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

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

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

IEEE is the sole entity that may authorize the use of certification marks, trademarks, or other designations to indicate compliance with the materials set forth herein.

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

Introduction

[This introduction is not part of IEEE Std 1546-2000, IEEE Guide for Digital Test Interface Format (DTIF) Application.]

IEEE Std 1445-1998, IEEE Standard for Digital Test Interchange Format (DTIF), provides a mechanism for digital test data interchange independent of a specific digital automatic test program generator and automatic test equipment. This guide is supplementary to IEEE Std 1445-1998.

The purpose of this guide is to provide an aid in the understanding and use of DTIF files. This application guide will provide information that will be an aid to users in developing tools such as preprocessors and postprocessors of DTIF data and other utilities.

Participants

At the time this proposed standard was completed, the Test Application (TA) Subcommittee had the following membership:

Mukund Modi, *TA Co-Chair*
Brit Frank, *TA Co-Chair*
Harold Davis, *Secretary*
Jose De La Cruz, *Working Group Chair*

In addition, the following persons made significant contributions to this document:

Joanne K. Buckley

Hugh A. Pritchett

David E. Rolince

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

Guy Adam
Joanne K. Buckley
Bernard Dathy
Harold Davis
Somnath Deb
Jose De La Cruz
Brit Frank

Arnold M. Greenspan
Gary L. Hardenburg
Michael G. Harrison
Osamu Karatsu
Gregory A. Maston
Mukund Modi
Leslie A. Orlidge

Jean Pouilly
David E. Rolince
John W. Sheppard
William R. Simpson
Joseph J. Stanco
William John Taylor
J. Richard Weger

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

Donald N. Heirman, *Chair*
James T. Carlo, *Vice Chair*
Judith Gorman, *Secretary*

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

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

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

*Member Emeritus

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

Alan Cookson, *NIST Representative*
Donald R. Volzka, *TAB Representative*

Andrew D. Ickowicz
IEEE Standards Project Editor

Contents

1.	Overview	1
1.1	Scope	1
1.2	Purpose	1
2.	References	1
3.	Definitions of terms and acronyms	2
3.1	Terms	2
3.2	Acronyms	2
4.	DTIF functional groups	2
4.1	UUT Model group	4
4.2	Stimulus and Response group	8
4.3	Fault Dictionary group	12
4.4	Probe group	13
5.	DTIF application example	16
5.1	Application example – UUT Model group	17
5.2	Application example—Stimulus and Response group	22
5.3	Application example—Fault Dictionary group	26
5.4	Application example—Probe group	29

IEEE Guide for Digital Test Interchange Format (DTIF) Application

1. Overview

The Digital Test Interchange Format (DTIF) is intended to provide a data interchange that can be used with various test environments. The DTIF standard provides a neutral format for the development and delivery of digital test information.

1.1 Scope

This guide serves as an aid to developers who need to represent data in an application neutral manner and have chosen DTIF (see IEEE Std 1445-1998¹) to do so. The guide explains the DTIF from a user perspective, giving examples and directions showing how to implement and use the DTIF. The intent of the document is to provide clarity and detailed knowledge for users of IEEE Std 1445-1998.

1.2 Purpose

This guide is meant to provide an aid in the understanding and use of DTIF files. This information will be an aid to users in developing tools such as preprocessors and postprocessors of DTIF data and other utilities.

2. References

This standard shall be used in conjunction with the following publications:

IEEE 100, The Authoritative Dictionary of IEEE Standards Terms, Seventh Edition.²

IEEE Std 1445-1998, IEEE Standard for Digital Test Interchange Format (DTIF).

¹Information on references can be found in Clause 2.

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

3. Definitions of terms and acronyms

3.1 Terms

3.1.1 fault simulation: The process of simulating a unit under test (UUT) failure and observing its effects.

3.1.2 pin map: The interconnections between circuit pins and automatic test equipment (ATE) input/output (I/O) pins (tester channels).

3.1.3 user node: A physical point on a UUT that can be a primary input, primary output, or other net connection.

3.1.4 wired net: A schema used to represent circuit nodes containing more than one source, which utilizes virtual and actual component connections and implementations.

3.2 Acronyms

ASCII	American Standards Code for Information Interchange
ATE	automatic test equipment
CPP	clock pulses per pattern
DTIF	digital test interchange format
I/O	input/output
ITA	interface test adapter
PI	primary input
PO	primary output
POPAT	primary output pattern
TPS	test program set
TSET	timing set
TU	timing unit
UUT	unit under test

4. DTIF functional groups

There are thirty-nine DTIF files composed of one HEADER file and thirty-eight data files. The thirty-nine DTIF files are organized into four functional groups. The four functional groups are as follows:

- a) UUT Model
- b) Stimulus and Response
- c) Fault Dictionary
- d) Probe

Each functional group contains related, non-sequential American Standard Code for Information Interchange (ASCII) data files. Figure 1 depicts the DTIF data files organized into the four functional groups by file type name followed by the file number in parenthesis. This organization portrays the role of each functional group as it relates to unit under test (UUT) testing.

The UUT Model group contains all the files required to define the topological database for diagnosing faults on the UUT. This includes component instantiations, component I/O pins, primary inputs, primary outputs, and package interconnections. The Stimulus and Response group is used for fault detection testing (end-to-end) while all the other groups are used for the fault isolation process. The intent of an end-to-end test of a UUT is to verify the overall functionality of the item and to indicate a failure if one exists and can be

detected by the test pattern set developed for it. There is no explicit attempt to diagnose the cause of the failure. Fault isolation by fault dictionary is a diagnostic test technique that does not require physical probing of a circuit. Data generated by a fault simulator is analyzed and generally arranged into matrices that match fault indications with a characteristic failing response at the primary output pins of the UUT. Actual failing responses are captured on the automatic test equipment (ATE) and compared to the fault dictionary matrix for fault isolation. Isolating faults using a probe is a diagnostic test technique that utilizes a probe connected to the ATE to aid in isolating failures on the UUT. The information required to generate a probe diagnostic test includes a detailed description of the circuit that would help direct the test operator where to probe and provide the data necessary to determine whether or not a response monitored by the probe matched the good circuit response. The three mutually independent areas for testing that are covered by the DTIF files are as follows:

- end-to-end
- diagnostics using fault dictionary
- diagnostics using probe

These three areas are further broken down into static and dynamic modes.

The HEADER file does not belong to any of the four functional groups. It provides summary information and a listing of the total DTIF file set developed for a given UUT. The first record identifies the file type name, the DTIF file number, the file revision number (optional), the UUT name, and the date and time of origination. The first record in every DTIF file follows this format.

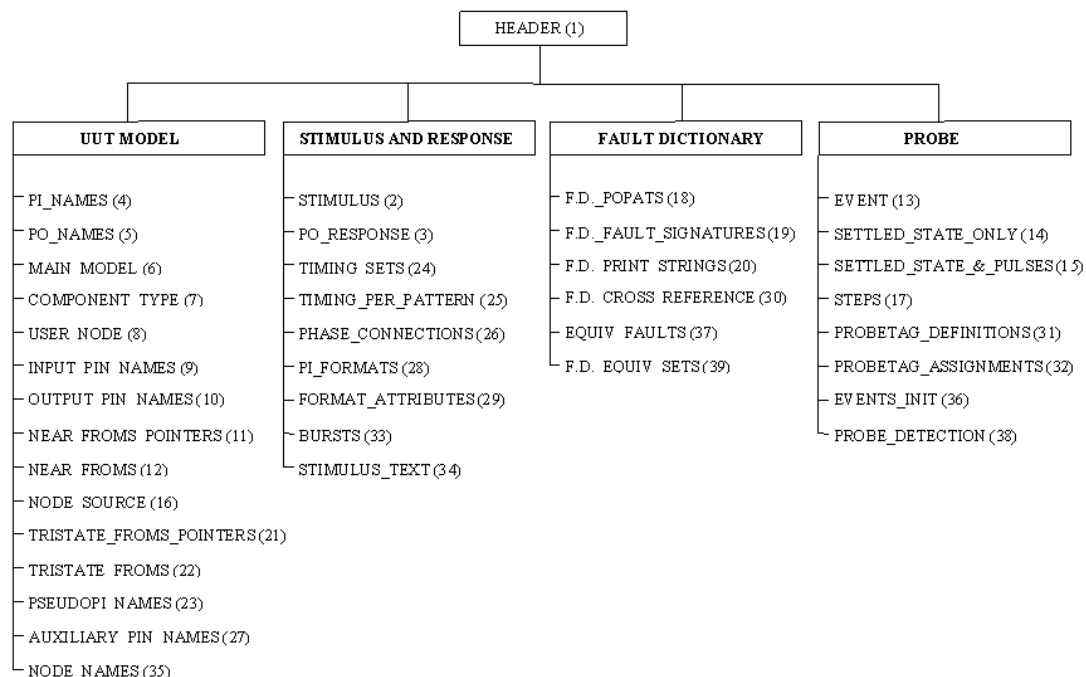


Figure 1—DTIF data file types

4.1 UUT Model group

The UUT Model group contains fifteen files that define the UUT model topology. Extensive use is made of pointers and lists within these files. Table 1 identifies all of the files in the UUT Model group and is indicated by the DTIF file type name and number and DTIF file name along with its pertinent applications.

Table 1—UUT Model group files

DTIF file type name and number	DTIF file name	End-to-end	End-to-end	Fault dict.	Fault dict.	Probe	Probe
		Static	Dynamic	Static	Dynamic	Static	Dynamic
PI_NAMES (4)	pinames.tap	X	X	X	X	X	X
PO_NAMES (5)	ponames.tap	X	X	X	X	X	X
MAIN_MODEL (6)	mainmodel.tap			X	X	X	X
COMPONENT_TYPE (7)	types.tap			X	X	X	X
USER_NODE (8)	usernodes.tap			X	X	X	X
INPUT_PIN_NAMES (9)	inputpins.tap			X	X	X	X
OUTPUT_PIN_NAMES (10)	outputpin.tap			X	X	X	X
NEAR_FROMS_POINTERS (11)	nerfrmptr.tap					X	X
NEAR_FROMS (12)	nearfroms.tap					X	X
NODE_SOURCE (16)	nodsourc.tap					X	X
TRISTATE_FROMS_POINTERS (21)	zfromptrs.tap					X	X
TRISTATE_FROMS (22)	zfroms.tap					X	X
PSEUDOPI_NAMES (23)	psupinams.tap			X	X	X	X
AUXILIARY_PIN_NAMES (27)	auxpins.tap			X	X	X	X
NODE_NAMES (35)	nodenames.tap			X	X	X	X

4.1.1 PI_NAMES (4)

This file defines each primary input (PI) pin by name and binds it to a user node number. These are used for input/output (I/O) connection identification.

4.1.2 PO_NAMES (5)

This file defines each primary output (PO) pin by name and binds it to a user node number. These are used for I/O connection identification.

4.1.3 MAIN_MODEL (6)

A UUT model can be understood in terms of six major elements:

- a) Component types
- b) Component pins
- c) Package names
- d) Package numbers
- e) User node numbers
- f) User node names

A component type and its pins only need to be identified once for a given UUT model even though there may be multiple instantiations of a component type. When used in the UUT model, packages are defined with a component type and a unique package name and number to distinguish them from their multiple usage in the UUT model. Each connection from package pin to package pin(s) (the net list) must have a user node number and can optionally have a user node name. The combination of package names, package pins, and user node numbers completely defines the schematic interconnection of the UUT model.

Only those component types contained in the COMPONENT_TYPE file are interconnected to construct the MAIN_MODEL file. Pointers in the MAIN_MODEL file lead to locations in the USER_NODE file that provides the details on the component package interconnections.

Every time a component is used in the MAIN_MODEL file, it is assigned a package number and package name. For example, if there were two component types, with component type 1 being used eleven times and component type 2 being used once, there would be twelve packages with twelve names and twelve numbers in the MAIN_MODEL file made up of only two component types.

The interconnection of packages is achieved with user nodes. Every node MUST have a unique user node number, and most have user node names (identified in the NODE_NAMES, PO_NAMES, and PI_NAMES files). The NEAR_FROMS and TRISTATE_FROMS files contain special types of user nodes that are used for probing algorithms. The NODE_SOURCE file correlates every user node to the component pin name and package name of the source driving that node.

The AUXILIARY_PIN_NAMES and PSEUDOPI_NAMES files list other special case nodes used in modeling and testing on the ATE. Examples of this are Vcc and ground.

4.1.4 COMPONENT_TYPE (7)

This file sequentially lists every component type by component name (e.g., SN74LS00) and provides pointers to the starting location of each component's input and output pin names in the INPUT_PIN_NAMES and OUTPUT_PIN_NAMES files. It also provides any alternate power and ground pins used by the component. A component type may be used multiple times in the UUT model. Each usage is identified by a unique package number and package name (e.g., U15) to discriminate one use of a component from another.

A special component type is a wired net. Wired nets provide a means of representing net list connections that have more than one source, as is the case with many bidirectional data buss connections. The term *wired net* is a scheme used to describe the actual and virtual components, connections, and implementations that are utilized and provide this information in the DTIF.

Components that have bidirectional pins can be described with different input and output pin counts than what might be intuitive. The bidirectional pins can be counted as outputs, but not counted as inputs, in the COMPONENT_TYPE file. This phenomenon propagates through to the package declarations in the MAIN_MODEL file. For connections with multiple sources, described in the USER_NODE file, that utilize these type package pins, there will occur an associated wired net package that is not present in the actual

circuit schematic. The wired net packages are instances of wired net components. The presence of a “W” in the last field in the component declaration, found in the COMPONENT_TYPE file, identifies wired net components. The input pins of the wired net packages will have the same user node numbers as the output pins of the actual packages that require the bussed node functionality.

If the connection is a PI, then the user node number that corresponds to the wired net package’s input will correspond to the name in the PI_NAMES file. The output pins of the wired net packages will have user nodes that are used to represent the connection’s output logic value. If the connection is a PO, it will also correspond to a name in the PO_NAMES file. The INPUT_PIN_NAMES and OUTPUT_PIN_NAMES files will contain names for the wired net component pins, as do all other UUT pins. The wired net component pins, however, cannot be converted to real numbers from their ASCII representations, as can the other non-wired net UUT package pins.

A non-wired net component will always have pin names that are ASCII representations of its pin number. The connectivity group entries in the PI_NAMES and PO_NAMES files can be used to identify UUT edge nodes that will be construed as wired net nodes. Any connectivity group number other than zero will uniquely identify a wired net node that is present at a point on the circuit intended for direct access. Internal nodes will not be identified in this way since they do not appear as PIs or POs.

4.1.5 USER_NODE (8)

This file lists all of the user node numbers for each package in ascending package number as listed in the MAIN_MODEL file. For each package, inputs are listed first, followed by output pins in the order that they are found in the INPUT_PIN_NAMES and OUTPUT_PIN_NAMES files, respectively. Unused pins in the model are also allocated user node numbers. Therefore, there are at least as many elements in the list as there are package pins in the MAIN_MODEL file. If wired nets are used in the UUT model, there will be more user nodes than physical nodes.

The MAIN_MODEL file directly points to the USER_NODE file. This file is the one that combines all of the packages together into a model. User nodes are used to relate the input and output pins on a package to a component pin number by referring to the information found in the NODE_SOURCE file. This file also provides a pathway from all of the other node name files to the component files.

4.1.6 INPUT_PIN_NAMES (9)

This file sequentially lists every component input pin name for every component type as they are identified in the COMPONENT_TYPE file.

4.1.7 OUTPUT_PIN_NAMES (10)

This file sequentially lists every component output pin name for every component type as they are identified in the COMPONENT_TYPE file.

4.1.8 NEAR_FROMS_POINTERS (11)

This file is a pointer into the NEAR_FROMS file. This file indirectly identifies all of the near-froms for each component by output pin number sequence. To do so, it points to the NEAR_FROMS file which, in turn, points to the INPUT_PIN_NAMES or OUTPUT_PIN_NAMES file for final identification of specific near-from pin names.

4.1.9 NEAR_FROMS (12)

This file lists all of the near-from pins associated with each component’s set of input pin names and/or output pin names. It is a second level pointer list to locations in the INPUT_PIN_NAMES and/or