IEC 62530

Edition 2.0 2011-05

INTERNATIONAL STANDARD

IEEE Std 1800™

SystemVerilog – Unified Hardware Design, Specification, and Verification Language

INTERNATIONAL ELECTROTECHNICAL COMMISSION

PRICE CODE XX

ICS 25.040

ISBN 978-2-88912-450-3

Contents

Part One:
Design and Verification Constructs

Part Two: Hierarchy Constructs

Part Three:
Application Programming Interfaces

Part Four:
Annexes

List of figures

List of tables

List of syntax excerpts

INTERNATIONAL ELECTROTECHNICAL COMMISSION

SystemVerilog – Unified Hardware Design, Specification, and Verification Language

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.

2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.

3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.

5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.

6) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62530/IEEE Std 1800 has been processed through IEC technical committee 93: *Design automation*.

This second edition cancels and replaces the first edition published in 2007. This edition constitutes a technical revision.

The text of this standard is based on the following documents:

IEEE Std	FDIS	Report on voting
1800 (2009)	93/303/FDIS	93/305/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

IMPORTANT – The 'color inside' logo on the cover page of this publication indicates that it contains colors which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a color printer.

IEC/IEEE Dual Logo International Standards

This Dual Logo International Standard is the result of an agreement between the IEC and the Institute of Electrical and Electronics Engineers, Inc. (IEEE). The original IEEE Standard was submitted to the IEC for consideration under the agreement, and the resulting IEC/IEEE Dual Logo International Standard has been published in accordance with the ISO/IEC Directives.

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

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

The IEC and IEEE do not warrant or represent the accuracy or content of the material contained herein, and expressly disclaim any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEC/IEEE Dual Logo International Standards documents are supplied "AS IS".

The existence of an IEC/IEEE Dual Logo International 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 IEC/IEEE Dual Logo International Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEC and IEEE are not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Neither the IEC nor IEEE is undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEC/IEEE Dual Logo International Standards or IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

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

Comments for revision of IEC/IEEE Dual Logo International Standards are welcome from any interested party, regardless of membership affiliation with the IEC or IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board, 445 Hoes Lane, Piscataway, NJ 08854, USA and/or General Secretary, IEC, 3, rue de Varembé, PO Box 131, 1211 Geneva 20, Switzerland.

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

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 Standard for SystemVerilog — Unified Hardware Design, Specification, and Verification Language

Sponsor
Design Automation Standards Committee
of the
IEEE Computer Society

and the
IEEE Standards Association Corporate Advisory Group

Approved 11 November 2009
IEEE-SA Standards Board

Abstract: This standard represents a merger of two previous standards: IEEE Std 1364™-2005 Verilog hardware description language (HDL) and IEEE Std 1800-2005 SystemVerilog unified hardware design, specification, and verification language. The 2005 SystemVerilog standard defines extensions to the 2005 Verilog standard. These two standards were designed to be used as one language. Merging the base Verilog language and the SystemVerilog extensions into a single standard provides users with all information regarding syntax and semantics in a single document.

Keywords: assertions, design automation, design verification, hardware description language, HDL, HDVL, PLI, programming language interface, SystemVerilog, Verilog, VPI

IEEE introduction

The purpose of this standard is to provide the electronic design automation (EDA), semiconductor, and system design communities with a well-defined and official IEEE unified hardware design, specification, and verification standard language. The language is designed to coexist and enhance the hardware description and verification languages (HDVLs) presently used by designers while providing the capabilities lacking in those languages.

SystemVerilog is a unified hardware design, specification, and verification language based on the Accellera SystemVerilog 3.1a extensions to the Verilog HDL [B3][a], published in 2004. Accellera is a consortium of EDA, semiconductor, and system companies. IEEE Std 1800 enables a productivity boost in design and validation and covers design, simulation, validation, and formal assertion-based verification flows.

SystemVerilog enables the use of a unified language for abstract and detailed specification of the design, specification of assertions, coverage, and testbench verification based on manual or automatic methodologies. SystemVerilog offers application programming interfaces (APIs) for coverage and assertions, a vendor-independent API to access proprietary waveform file formats, and a direct programming interface (DPI) to access proprietary functionality. SystemVerilog offers methods that allow designers to continue to use present design languages when necessary to leverage existing designs and intellectual property. This standardization project will provide the VLSI design engineers with a well-defined IEEE standard, which meets their requirements in design and validation, and which enables a step function increase in their productivity. This standardization project will also provide the EDA industry with a standard to which they can adhere and which they can support in order to deliver their solutions in this area.

Notice to users

Laws and regulations

Users of these documents should consult all applicable laws and regulations. Compliance with the provisions of this standard does not imply 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.

Copyrights

This document is copyrighted by the IEEE. It is made available 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 this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

[a]The numbers in brackets correspond to the numbers in the bibliography in Annex R.

Updating of IEEE documents

Users of IEEE standards 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. 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 the IEEE Standards Association website at http://ieeexplore.ieee.org/xpl/standards.jsp, or contact the IEEE at the address listed previously.

For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA website at http://standards.ieee.org.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: http://standards.ieee.org/reading/ieee/updates/errata/index.html. Users are encouraged to check this URL for errata periodically.

Interpretations

Current interpretations can be accessed at the following URL: http://standards.ieee.org/reading/ieee/interp/index.html.

Patents

Attention is called to the possibility that implementation of this amendment may require use of subject matter covered by patent rights. By publication of this amendment, no position is taken with respect to the existence or validity of any patent rights in connection therewith. 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 amendment 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: This standard is not intended to ensure safety, security, health, or environmental protection in all circumstances. Implementers of the standard are responsible for determining appropriate safety, security, environmental, and health practices or regulatory requirements.

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Documents." They can also be obtained on request from IEEE or viewed at http://standards.ieee.org/IPR/disclaimers.html.

Part One:
Design and Verification Constructs

SystemVerilog — Unified Hardware Design, Specification, and Verification Language

1. Overview

1.1 Scope

This SystemVerilog standard (IEEE Std 1800) is a Unified Hardware Design, Specification, and Verification language. IEEE Std 1364™-2005 Verilog is a design language. Both standards were approved by the IEEE-SASB in November 2005. This standard creates new revisions of the IEEE 1364 Verilog and IEEE 1800 SystemVerilog standards, which include errata fixes and resolutions, enhancements, enhanced assertion language, merger of Verilog Language Reference Manual (LRM) and SystemVerilog 1800 LRM into a single LRM, integration with Verilog-AMS, and ensures interoperability with other languages such as SystemC and VHDL.

1.2 Purpose

The purpose of this project is to provide the EDA, Semiconductor, and System Design communities with a solid and well-defined IEEE Unified Hardware Design, Specification and Verification standard language, while resolving errata and developing enhancements to the current IEEE 1800 SystemVerilog standard. The language is designed to co-exist, be interoperable, possibly merge, and enhance those hardware description languages presently used by designers.

1.3 Merger of IEEE Std 1364-2005 and IEEE Std 1800-2005

This standard represents a merger of two previous standards: the IEEE Std 1364-2005[1] Verilog Hardware Description Language (HDL) and the IEEE Std 1800-2005 SystemVerilog Unified Hardware Design, Specification, and Verification Language. In these previous standards, Verilog was the base language and defined a completely self-contained standard. SystemVerilog defined a number of significant extensions to Verilog, but IEEE Std 1800-2005 was not a self-contained standard; IEEE Std 1800-2005 referred to, and relied on, IEEE Std 1364-2005. These two standards were designed to be used as one language. Merging the base Verilog language and the SystemVerilog extensions into a single standard enables users to have all information regarding syntax and semantics in a single document.

This standard serves as a complete specification of the SystemVerilog language. This standard contains the following:

- The formal syntax and semantics of all SystemVerilog constructs
- Simulation system tasks and system functions, such as text output display commands
- Compiler directives, such as text substitution macros and simulation time scaling
- The Programming Language Interface (PLI) mechanism
- The formal syntax and semantics of the SystemVerilog Verification Procedural Interface (VPI)
- An Application Programming Interface (API) for coverage access not included in VPI
- Direct programming interface (DPI) for interoperation with the C programming language
- VPI, API, and DPI header files
- Concurrent assertion formal semantics
- The formal syntax and semantics of standard delay format (SDF) constructs
- Informative usage examples

1.4 Special terms

Throughout this standard, the following terms apply:

- *SystemVerilog* refers to the unified Verilog base language (IEEE Std 1364) with the SystemVerilog extensions to Verilog.
- *SystemVerilog 3.1a* refers to the Accellera SystemVerilog Language Reference Manual [B3],[2] a precursor to IEEE Std 1800-2005.
- *Verilog* refers to IEEE Std 1364-2005 for the Verilog HDL.
- *Language Reference Manual* (*LRM*) refers to the document describing a Verilog or SystemVerilog standard.
- *Tool* refers to a software implementation that reads SystemVerilog source code, such as a logic simulator.

NOTE—In the IEEE Std 1800-2005 version of the standard, *SystemVerilog* referred to just the extensions to the IEEE Std 1364-2005 Verilog language, and did not include the Verilog base language.[3]

1.5 Conventions used in this standard

This standard is organized into clauses, each of which focuses on a specific area of the language. There are subclauses within each clause to discuss individual constructs and concepts. The discussion begins with an

[1]Information on references can be found in Clause 2.
[2]The numbers in brackets correspond to those of the bibliography in Annex R.
[3]Notes in text, tables, and figures are given for information only and do not contain requirements needed to implement the standard.

introduction and an optional rationale for the construct or the concept, followed by syntax and semantic descriptions, followed by examples and notes.

The terminology conventions used throughout this standard are as follows:

— The word *shall* is used to indicate 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*).

— The word *should* is used to indicate 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; or that (in the negative form) a certain course of action is deprecated but not prohibited (*should* equals *is recommended that*).

— 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*).

1.6 Syntactic description

The main text uses the following conventions:

— *Italicized* font when a term is being defined

— `Constant-width` font for examples, file names, and references to constants, especially `0`, `1`, `x`, and `z` values

— **`Boldface constant-width`** font for SystemVerilog keywords, when referring to the actual keyword

The formal syntax of SystemVerilog is described using Backus-Naur Form (BNF). The following conventions are used:

— Lowercase words, some containing embedded underscores, denote syntactic categories. For example:

```
module_declaration
```

— **Boldface-red** characters denote reserved keywords, operators, and punctuation marks as a required part of the syntax. For example:

```
module    =>    ;
```

— A vertical bar (|) that is not in boldface-red separates alternative items. For example:

```
unary_operator ::=
      + | - | ! | ~ | & | ~& | | | ~| | ^ | ~^ | ^~
```

— Square brackets ([ ]) that are not in boldface-red enclose optional items. For example:

```
function_declaration ::= function [ lifetime ] function_body_declaration
```

— Braces ({ }) that are not in boldface-red enclose a repeated item. The item may appear zero or more times; the repetitions occur from left to right as with an equivalent left-recursive rule. Thus, the following two rules are equivalent:

```
list_of_param_assignments ::= param_assignment { , param_assignment }
list_of_param_assignments ::=
      param_assignment
    | list_of_param_assignments , param_assignment
```

A *qualified term* in the syntax is a term such as *array_identifier* for which the "array" portion represents some semantic intent and the "identifier" term indicates that the qualified term reduces to the "identifier" term in the syntax. The syntax does not completely define the semantics of such qualified terms; for example while an identifier which would qualify semantically as an array_identifier is created by a declaration, such declaration forms are not explicitly described using *array_identifier* in the syntax.

1.7 Use of color in this standard

This standard uses a minimal amount of color to enhance readability. The coloring is not essential and does not affect the accuracy of this standard when viewed in pure black and white. The places where color is used are the following:

- Cross references that are hyperlinked to other portions of this standard are shown in underlined-blue text (hyperlinking works when this standard is viewed interactively as a PDF file).
- Syntactic keywords and tokens in the formal language definitions are shown in **`boldface-red text`**.
- Some figures use a minimal amount of color to enhance readability.

1.8 Contents of this standard

A synopsis of the clauses and annexes is presented as a quick reference. All clauses and several of the annexes are normative parts of this standard. Some annexes are included for informative purposes only.

Part One: Design and Verification Constructs

Clause 1 describes the contents of this standard and the conventions used in this standard.

Clause 2 lists references to other standards that are required in order to implement this standard.

Clause 3 introduces the major building blocks that make up a SystemVerilog design and verification environment: modules, programs, interfaces, checkers, packages and configurations. This clause also discusses primitives, name spaces, the `$unit` compilation space, and the concept of simulation time.

Clause 4 describes the SystemVerilog simulation scheduling semantics.

Clause 5 describes the lexical tokens used in SystemVerilog source text and their conventions.

Clause 6 describes SystemVerilog data objects and types, including nets and variables, their declaration syntax and usage rules, and charge strength of the values on nets. This clause also discusses strings and string methods, enumerated types, user-defined types, constants, data scope and lifetime, and type compatibility.

Clause 7 describes SystemVerilog compound data types: structures, unions, arrays, including packed and unpacked arrays, dynamic arrays, associative arrays, and queues. This clause also describes various array methods.

Clause 8 describes the object-oriented programming capabilities in SystemVerilog. Topics include defining classes, dynamically constructing objects, inheritance and subclasses, data hiding and encapsulation, polymorphism, and parameterized classes.

Clause 9 describes the SystemVerilog procedural blocks: **`initial`**, **`always`**, **`always_comb`**, **`always_ff`**, **`always_latch`**, and **`final`**. Sequential and parallel statement grouping, block names, statement labels, and process control are also described.

Clause 10 describes continuous assignments, blocking and nonblocking procedural assignments, and procedural continuous assignments.

Clause 11 describes the operators and operands that can be used in expressions. This clause also discusses operations on arrays, operator methods, and operator overloading.

Clause 12 describes SystemVerilog procedural programming statements, such as decision statements and looping constructs.

Clause 13 describes tasks and functions, which are subroutines that can be called from more than one place in a behavioral model.

Clause 14 defines clocking blocks, input and output skews, cycle delays, and default clocking.

Clause 15 describes interprocess communications using event types and event controls, and built-in semaphore and mailbox classes.

Clause 16 describes immediate and concurrent assertions, properties, sequences, sequence operations, multiclock sequences, and clock resolution.

Clause 17 describes checkers. Checkers allow the encapsulation of assertions and modeling code to create a single verification entity.

Clause 18 describes generating random numbers, constraining random number generation, dynamically changing constraints, seeding random number generators (RNGs), and randomized `case` statement execution.

Clause 19 describes coverage groups, coverage points, cross coverage, coverage options, and coverage methods.

Clause 20 describes most of the built-in system tasks and system functions.

Clause 21 describes additional system tasks and system functions that are specific to I/O operations.

Clause 22 describes various compiler directives, including a directive for controlling reserved keyword compatibility between versions of previous Verilog and SystemVerilog standards.

Part Two: Hierarchy Constructs

Clause 23 describes how hierarchies are created in SystemVerilog using module instances and interface instances, and port connection rules. This clause also discusses the $root top-level instances, nested modules, extern modules, identifier search rules, how parameter values can be overridden, and binding auxiliary code to scopes or instances.

Clause 24 describes the testbench program construct, the elimination of testbench race conditions, and program control tasks.

Clause 25 describes interface syntax, interface ports, modports, interface subroutines, parameterized interfaces, virtual interfaces, and accessing objects within interfaces.

Clause 26 describes user-defined packages and the std built-in package.

Clause 27 describes the generate construct and how generated constructs can be used to do conditional or multiple instantiations of procedural code or hierarchy.

Clause 28 describes the gate- and switch-level primitives and logic strength modeling.

Clause 29 describes how a User Defined Primitive (UDP) can be defined and how these primitives are included in SystemVerilog models.

Clause 30 describes how to specify timing relationships between input and output ports of a module.

Clause 31 describes how timing checks are used in specify blocks to determine whether signals obey the timing constraints.

Clause 32 describes the syntax and semantics of SDF constructs.

Clause 33 describes how to configure the contents of a design.

Clause 34 describes encryption and decryption of source text regions.

Part Three: Application Programming Interfaces (APIs)

Clause 35 describes SystemVerilog's direct programming interface (DPI), a direct interface to foreign languages and the syntax for importing functions from a foreign language and exporting subroutines to a foreign language.

Clause 36 provides an overview of the Programming Language Interface (PLI and VPI).

Clause 37 presents the VPI data model diagrams, which document the VPI object relationships and access methods.

Clause 38 describes the VPI routines.

Clause 39 describes the assertion API in SystemVerilog.

Clause 40 describes the coverage API in SystemVerilog.

Part Four: Annexes

Annex A (normative) defines the formal syntax of SystemVerilog, using BNF.

Annex B (normative) lists the SystemVerilog keywords.

Annex C (informative) lists constructs that have been deprecated from SystemVerilog. The annex also discusses the possible deprecation of the **`defparam`** statement and the procedural **`assign`**/**`deassign`** statements.

Annex D (informative) describes system tasks and system functions that are frequently used, but that are not required in this standard.

Annex E (informative) describes compiler directives that are frequently used, but that are not required in this standard.

Annex F (normative) describes a formal semantics for SystemVerilog concurrent assertions.

Annex G (normative) describes the SystemVerilog standard package, containing type definitions for mailbox, semaphore, randomize, and process.

Annex H (normative) defines the C-language layer for the SystemVerilog DPI.

Annex I (normative) defines the standard `svdpi.h` include file for use with SystemVerilog DPI applications.

Annex J (normative) describes common guidelines for the inclusion of foreign language code into a SystemVerilog application.

Annex K (normative) provides a listing of the contents of the `vpi_user.h` file.

Annex L (normative) provides a listing of the contents of the `vpi_compatibility.h` file, which extends the `vpi_user.h` include file.

Annex M (normative) provides a listing of the contents of the `sv_vpi_user.h` file, which extends the `vpi_user.h` include file.

Annex N (normative) provides the C source code for the SystemVerilog random distribution system functions.

Annex O (informative) describes various scenarios that can be used for intellectual property (IP) protection, and it also shows how the relevant pragmas can be used to achieve the desired effect of securely protecting, distributing, and decrypting the model.

Annex P (informative) defines terms that are used in this standard.

Annex Q (informative) provides a general mapping of clause numbers from IEEE Std 1364-2005 Verilog standard and IEEE Std 1800-2005 SystemVerilog standard into this standard.

Annex R (informative) lists reference documents that are related to this standard.

1.9 Deprecated clauses

Annex C lists constructs that appeared in previous versions of either IEEE Std 1364 or IEEE Std 1800, but that have been deprecated and do not appear in this standard. This annex also lists constructs that appear in this standard, but that are under consideration for deprecation in a future version of this standard.

1.10 Examples

Small SystemVerilog code examples are shown throughout this standard. These examples are informative. They are intended to illustrate the usage of SystemVerilog constructs in a simple context and do not define the full syntax.

1.11 Prerequisites

Some clauses of this standard presuppose a working knowledge of the C programming language.

2. Normative references

The following referenced documents are indispensable for the application of this standard (i.e., they must be understood and used, so each referenced document is cited in the 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.

Anderson, R., Biham, E., and Knudsen, L. "Serpent: A Proposal for the Advanced Encryption Standard," NIST AES Proposal, 1998, http://www.cl.cam.ac.uk/~rja14/Papers/serpent.tar.gz.

ANSI Std X9.52-1998, American National Standard for Financial Services—Triple Data Encryption Algorithm Modes of Operation.[4]

ElGamal, T., "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Transactions on Information Theory*, vol. IT-31, no. 4, pp. 469–472, July 1985.

FIPS 46-3 (October 1999), Data Encryption Standard (DES).[5]

FIPS 180-2 (August 2002), Secure Hash Standard (SHS).

FIPS 197 (November 2001), Advanced Encryption Standard (AES).

IEC 62530:2007, Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language.

IEEE Std 1800™-2005, IEEE Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language.

NOTE—IEEE Std 1800™-2005 was adopted as IEC 62530:2007.

IEEE Std 754™, IEEE Standard for Binary Floating-Point Arithmetic.[6, 7]

IEEE Std 1003.1™, IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®).

IEEE Std 1364™-1995, IEEE Standard Hardware Description Language Based on the Verilog™ Hardware Description Language.

IEEE Std 1364™-2001, IEEE Standard for Verilog Hardware Description Language.

IEEE Std 1364™-2005, IEEE Standard for Verilog Hardware Description Language.

IETF RFC 1319 (April 1992), The MD2 Message-Digest Algorithm.[8]

IETF RFC 1321 (April 1992), The MD5 Message-Digest Algorithm.

IETF RFC 2045 (November 1996), Multipurpose Internet Mail Extensions (MIME), Part One: Format of Internet Message Bodies.

IETF RFC 2144 (May 1997), The CAST-128 Encryption Algorithm.

---

[4]ANSI publications are available from the Sales Department, American National Standards Institute, 25 West 43rd Street, 4th Floor, New York, NY 10036, USA (http://www.ansi.org/).

[5]FIPS publications are available from the National Technical Information Service (NTIS), U. S. Dept. of Commerce, 5285 Port Royal Rd., Springfield, VA 22161 (http://www.ntis.org/).

[6]IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA (http://standards.ieee.org/).

[7]The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.

[8]IETF requests for comments (RFCs) are available from the Internet Engineering Task Force (http://www.ieft.org).

IETF RFC 2437 (October 1998), PKCS #1: RSA Cryptography Specifications, Version 2.0.

IETF RFC 2440 (November 1998), OpenPGP Message Format.

ISO/IEC 10118-3:2004, Information technology—Security techniques—Hash-functions—Part 3: Dedicated hash-functions.[9]

Schneier, B., "Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)," *Fast Software Encryption, Cambridge Security Workshop Proceedings (December 1993)*, Springer-Verlag, 1994, pp. 191–204.

Schneier, B., et al, *The Twofish Encryption Algorithm: A 128-Bit Block Cipher*, 1st ed., Wiley, 1999.

[9]ISO/IEC publications are available from the ISO Central Secretariat, Case Postale 56, 1 chemin de la Voie-Creuse, CH-1211 Genève 20, Switzerland/Suisse (http://www.iso.ch/) and from the IEC Central Office, Case Postale 131, 3 rue de Varembé, CH-1211 Genève 20, Switzerland/Suisse (http://www.iec.ch/). ISO/IEC publications are also available in the United States from Global Engineering Documents, 15 Inverness Way East, Englewood, Colorado 80112, USA (http://global.ihs.com/). Electronic copies are available in the United States from the American National Standards Institute, 25 West 43rd Street, 4th Floor, New York, NY 10036, USA (http://www.ansi.org/).