

IEEE Standards Committee Conclusions

For the benefit of individuals using IEEE Standards, this document contains carefully considered opinions, rationales, and conclusions, arrived at by experts in the field, many of whom were part of the original standards-writing group. Committee Conclusions are not developed or processed in accordance with the *IEEE Standards Manual*, and should not be construed as official Interpretations.

Circuits and Devices

Communications Technology

Computer

*Electromagnetics and
Radiation*

Energy and Power

Industrial Applications

*Signals and
Applications*

*Standards
Coordinating
Committees*



Published by the Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017-2394, USA.
No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written consent of the publisher.

AVA Documents
Document Sharing Hub

Authorized licensed use limited to: Tsinghua University. Downloaded on March 06, 2026 at 05:00:45 UTC from IEEE Xplore. Restrictions apply.

The Sense of VASG

October 1989

NOTE: This document does not constitute an alteration to IEEE Std 1076-1987, VHDL Reference Manual.

For information on the VASG or DASS Activities, contact:

The IEEE Computer Society
Standards Secretariat
1730 Massachusetts Avenue, NW
Washington, DC 20036
(202) 371-0101

Final formatting courtesy of C.A.D.onomist.

Copyright © 1990 by

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017, USA

*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.*

March 30, 1990

SF13326

FOREWORD

This document, *The Sense of the VASG*, contains a number of what the IEEE Standards Department has designated "Committee Conclusions," [① regarding IEEE Std 1076-1987, IEEE Standard VHDL Language Reference Manual] The Preface states that this document sets forth [② the considered opinion of experts in the field] ^{concerning} regarding typographical and editorial mistakes, together with ambiguities in the Language Reference Manual (LRM) of the published Standard. The *Sense of the VASG* is being made available to help users of the Standard well in advance of a complete revision, which is currently expected to be issued sometime in 1992. It should be emphasized that these "Committee Conclusions" have not been developed or processed in accordance with Clause 6 of the *IEEE Standards Manual*; therefore, they should not be construed as official Interpretations.

Table of Contents

Preface	iii
Issue 002 - Typographical and Editorial Errors	1
Issue 003 - Spaces in Abstract / Physical Literals	7
Issue 013 - Slice Prefix and Discrete Range Directions	9
Issue 021 - Non-locally-static Universal Expressions	13
Issue 027 - Generate Parameter Regions	17
Issue 032 - File I/O Inconsistencies	21
Issue 033 - OUT / INOUT Signal Parameters	25
Issue 036 - TextIO Format	27
Issue 039 - Attributes 'Last_Active, _Event, and _Value	31
Issue 044 - Access Objects and Static Names	35
Issue 051 - Resolution Function Parameters	37
Issue 077 - Passiveness of Processes	39

Page 40 is the last page of the Document

-- NOTES --

PREFACE

VHDL version 1076 became the IEEE standard hardware description language in 1987. This standard reflected reflected hard work on the part of the VHDL Analysis and Standardization Group (VASG), the working group of the IEEE chartered with instantiating VHDL.

However, the work of the VASG did not end with the coming of VHDL. Numerous issues arose regarding the language. These issues may be broken into three categories:

1. typos/editorial mistakes
2. ambiguities in the Language Reference Manual (LRM)
3. requests for new language features.

That these types of issues arose surprised no one. First, the occurrence of a fair number of typos and other obvious mistakes is very common in the first edition of any technical work. Indeed, it is surprising, given the speed at which the VHDL LRM was constructed, that more typos did not occur. Second, the LRM was constructed before any working version of the 1076 language existed. Many of the ambiguities in the LRM descriptions were found by developers, once VHDL toolset implementation began. Finally, once VHDL actually began to be used, requests for new language features were inevitable. It is one thing to debate a language in a committee setting, but quite another to have to design with it under deadline pressure.

In an ideal world, the first two types of issues would be taken care of by simply amending the VHDL LRM. Similarly, requests for language changes would be debated, much as was done during the original standardization effort, with accepted language features being added to the LRM. This, however, is not an ideal world. The IEEE rules strictly forbid amending the LRM without a rebaloting features-- even to change typos. Since rebaloting is a costly, time consuming activity not scheduled to be done again until 1992, it would seem that the issues raised about 1076 were destined to remain unanswered for some time.

Of course, this would be an unacceptable situation-- without some attempt to resolve language problems, both novice and expert users of VHDL would soon become very frustrated with the language. Perhaps even worse, the presence of ambiguities in the language would likely result in the existence of incompatible toolsets from the various vendors. This is hardly the right stuff for a newly minted standard.

Faced with this seeming impossible situation of having to clarify VHDL, but not being able to update the LRM, the VASG decided to try for a middle course. In particular, it was decided that issues of the first two types would be analyzed and corrected by the VASG, and collected in an informal "Sense of the VASG" document. You are now reading the preface of that document.

This document does not have the authority of the LRM, and in NO WAY alters the LRM. Rather, it serves as a companion document, exhibiting the considered opinion of experts in the field. Thus, one who wishes to ignore a VASG resolution of a language ambiguity when implementing, say, a simulator, can still speak of having a 1076 VHDL simulator, but may find that he/she is out of step with the rest of the community. This is particularly probable especially in light of the fact that many of the VASG participants are key developers for the major VHDL vendors.

Note that only issues of the first two types are covered by this document. Requests for language changes have been, at least up until now, put aside. There are two reasons for this. First, the VASG had its hands full with the other types of issues. More importantly, it was strongly felt by the VASG membership that adding new language features so soon after standardization would provide a moving target for designers learning the language. New features will only begun to be debated next year, with an eye towards the re-standardization in 1992.

From a logistics standpoint, the VASG works as follows. A subcommittee of the VASG, called the Issues Screening and Analysis Committee (ISAC) was formed. The ISAC meets in between full VASG meetings.

where each member is given new issues to tentatively resolve, and is given the opportunity to present suggested resolutions to older issues. The issue resolutions accepted by the ISAC are then presented to the full VASG, which usually meets in conjunction with major conferences or users' group meetings. This document contains the ISAC resolutions accepted by the full VASG.

It was decided early in this process to first attack the easiest problems, i.e., the obvious typos and editorial errors. The resolutions to all of these relatively easy issues were lumped into a single issue resolution report (#2 below). After those easy issues were resolved, a prioritization of the remaining issues was made. Those issues felt most important by the ISAC were the attacked in weighted order.

This has not been an easy matter. Of the over 150 issue reports received by the VASG, the ISAC presented 17 resolutions to the full VASG. Of these, 12 (including the lumped typos resolution) were accepted by the full VASG, usually after considerable debate. The details of these resolved issues make up the rest of these documents.

In closing, I feel a need to give special recognition to the ISAC committee members. The ISAC members are:

- Marc Alexandre
- Clive Charlwood
- Doug Dunlop
- Ray Hookway
- Bob Mackey
- Paul Menchini
- Alec Stanculesu

Without the large amounts of volunteer time expended by these individuals, the VASG could not have accomplished as much as it did.

Stanley Krolikoski

10/89

VHDL Issue Number: 0002
 Classification: Typographical and Editorial Errors
 Summary: Repair for a variety of typographical/editorial errors.
 Related Issues: none
 Relevant LRM Sections: 1.2.2, 1.3, 2.2, 2.3, 3.2.1.1, 3.3.1,
 4.3.1.2, 4.4, 7.3.2.2, 7.3.5, 9.6.1, 9.7, 12.2,
 12.2.1, 12.2.2, 12.2.4, 12.3.1.3, 13.4.2, B
 Key Words and Phrases: N/A
 Level Of Importance: N/A
 Level of Difficulty: N/A
 Current Status: Approved

 Date Submitted: 10/18/88
 Author of Submission: Doug Dunlop (and many others)
 Author's Affiliation: Intermetrics
 Author's Postal Address: 4733 Bethesda Ave #415
 Bethesda, MD 20814
 Author's Phone Number: (301) 657-3775
 Author's Net Address: dunlop%inmet.inmet.com@uunet.uu.net

Date Analyzed: 10 November 1988
 Author of Analysis: Doug Dunlop

Description of Problem

There are a number of typographical and editorial errors in the LRM (see below).

Proposed Resolution

These should be corrected.

VASG-ISAC Analysis & Rationale

Indeed these are typographical and editorial errors and should be corrected.

VASG-ISAC Recommendation for IEEE Std 1076-1987

1. The line in the example from LRM 9.6.1 on Page 9-12 that reads

```
component COMP port(A,B : inout BIT);
```

should read

```
component COMP port(A,B : inout BIT); end component;
```

This corrects illegal VHDL syntax.

2. The lines in the example from LRM 1.2.2 on Page 1-8 that read

```
component Full Adder
port(X, Y, Cin: Bit; Cout, Sum: out Bit);
```

Authorized licensed use limited to: Tsinghua University. Downloaded on March 06, 2026 at 05:00:45 UTC from IEEE Xplore. Restrictions apply.

should read

```
component Full_Adder
  port(X, Y, Cin: Bit; Cout, Sum: out Bit);
end component;
```

This corrects illegal VHDL syntax..

3. The lines in the example from LRM 2.3 on Page 2-6 that read

```
Check (Setup=>10 ns, D=>Bus, C=>Clk1);
Check (Hold=>5 ns, D=>Bus, C=>Clk2);
Check (15 ns, Bus, Clk); -- ambiguous if Data'Base = Clock'Base
```

should read

```
Check (Setup=>10 ns, D=>Wire, C=>Clk1);
Check (Hold=>5 ns, D=>Wire, C=>Clk2);
Check (15 ns, Wire, Clk); -- ambiguous if Data'Base = Clock'Base
```

This corrects illegal use of a VHDL reserved word.

4. The line from LRM 7.3.5 on Page 7-12 that reads

```
type_conversion ::= type_mark ( expression )
```

should be indented to match the other grammar productions appearing in the LRM.

5. In the glossary (Appendix B) on Page B-1, the entries for "alias" and for "aggregate" appear in the wrong order. The order of these two entries should be reversed.

6. The line in the example from LRM 1.3 on Page 1-10 that reads

```
for StructureView
```

should read

```
for Structure_View
```

This is a typographical error.

7. The line in the example from LRM 3.3.1 on Page 3-15 that reads

```
variable NEXT : LINK := HEAD.SUCC;
```

should read

```
variable NXT : LINK := HEAD.SUCC;
```

This corrects illegal use of a VHDL reserved word.

Authorized licensed use limited to: Tsinghua University. Downloaded on March 06, 2026 at 05:00:45 UTC from IEEE Xplore. Restrictions apply.

8. The line in the example from LRM 4.4 on Page 4-14 that reads

type POSITIVE is INTEGER range 1 to INTEGER'HIGH;

should read

subtype POSITIVE is INTEGER range 1 to INTEGER'HIGH;

This corrects illegal VHDL syntax.

9. The line in the example from LRM 9.7 on Page 9-14 that reads

B : block

should read

Gen : block

Similarly, the line in the same example that reads

end block B;

should read

end block Gen;

These corrections allow the signal B to be visible where it is referenced in example.

10. The last two paragraphs of LRM 2.2 on Page 2-5 are notes but are not identified as such. These two paragraphs should be preceded by the header (in italics) "Note:".

11. The last sentence of LRM 12.3.1.3 on Page 12-4 which reads

Elaboration of a size constraint consists of the evaluation of the expression.

should be deleted. Size constraints have been removed from the language.

12. Item 6 in LRM 7.3.2.2 on Page 7-11 that reads

6. As the expression defining the initial value of the drivers of one or more signals in an initialization specification, where the corresponding subtype is a constrained array subtype.

should read

6. As the expression defining the default values of signals in a signal declaration, where the corresponding subtype is a constrained array subtype.

13. The definition of "resume" in the glossary reads

A process that resumes in a given simulation cycle becomes ready to execute and will execute at the end of the given simulation cycle.

but should read

A process that resumes in a given simulation cycle becomes ready to execute and will execute at the end of the given simulation cycle.

This is a typographical error.

14. The sentence in the glossary from the definition of "specification" that reads

There are four kinds of specifications: attribute specifications, initialization specifications, configuration specifications, and disconnection specifications.

should read

There are three kinds of specifications: attribute specifications, configuration specifications, and disconnection specifications.

15. There are two definitions of "guarded signal" in the glossary. The second of these is more informative and thus the first definition should be removed.

16. The sentence immediately above the examples in LRM 13.4.2 on Page 13-5 that reads

An exponent indicates the power of the base by which the value of the based literal with the exponent.

is not well formed. It should be replaced with

An exponent indicates the power of the base by which the value of the based literal without the exponent is to be multiplied to obtain the value of the based literal with the exponent. An exponent for a based integer literal must not have a minus sign.

17. The references to "generic map clause" and "port map clause" in the 7th, 8th, 9th, and 10th items in the list beginning in LRM 3.2.1.1 on Page 3-11 should be read "generic map aspect" and "port map aspect", respectively.

18. The four references to "port map clause" in the paragraph beginning "If a subelement of a resolved signal of a composite type is ..." from LRM 4.3.1.2 on Page 4-6 should be read "port map aspect".

19. In LRM 12.2, 12.2.1, 12.2.2, and 12.2.4 on Page 12-2, the references to "generic map clause" and "port map clause" should be read "generic map aspect" and "port map aspect", respectively. There are two such references in 12.2, two in 12.2.1, one in the title of 12.2.2, one in the body of 12.2.2, one in the title of 12.2.4, and one in the body of 12.2.4.

20. The line in the example from LRM 1.3.2 on Page 1-13 that reads

use entity StdCells.PadTriState4 (StdCells.DataFlow)

should read

use entity StdCells.PadTriState4 (DataFlow)

This corrects illegal VHDL syntax.

21. The sentence in LRM 3.1.3.1 on Page 3-7 that reads

The only predefined physical is type TIME.

should read

The only predefined physical type is TIME.

This is a typographical error.

22. The sentence in LRM 3.1.3.1 on Page 3-7 that reads

The range of TIME is implementation-dependent, but it is guaranteed to include the range -2147483647 to +2147483647.

should read

The range of TIME is implementation-dependent, but it is guaranteed to include the range -2147483647 to +2147483647.

The writing of "46" for "64" is a typographical error.

23. The literals "-1E38" and "+1E38" in LRM 3.1.4.1 on Page 3-8 should be read "-1.OE38" and "+1.OE38", respectively. Similarly, the literals "-1E38" and "+1E38" in LRM 3.1.4 on Page 3-8 should be read "-1.OE38" and "+1.OE38", respectively. These changes make the type of these literals universal_real, which is the appropriate type for specifying the minimum bounds of floating point types.

24. The list item in LRM 4.3 on Page 4-3 that reads

o a formal port of a design entity.

should read

- o a formal port.

The phrase "of a design entity" mistakenly prevents the list item from applying to block ports.

25. The line from the example in LRM 3.3 on Page 3-14 that reads

```
type BUFFER_PTR is access BUFFER;
```

should read

```
type BUFFER_PTR is access MY_BUFFER;
```

This corrects illegal use of a VHDL reserved word.

26. The example in LRM 1.3.2 on Page 1-13 that reads

```
for D1 : DSP
  -- binding specified in design entity or else defaults
  for Filterer
    -- configuration items for filtering components
  end for;
  for Processor
    -- configuration items for processing components
  end for;
end for;
```

should read

```
for D1 : DSP
  -- binding specified in design entity or else defaults
  for DSP_STRUCTURE
    for Filterer
      -- configuration items for filtering components
    end for;
    for Processor
      -- configuration items for processing components
    end for;
  end for;
end for;
```

This corrects a missing block configuration.

VASG-ISAC Recommendation for Future Revisions

Correct these problems in the LRM.

VHDL Issue Number: 0003
 Classification: Examples, Notes and Appendixes
 Summary: Examples in the LRM do not have a space
 between the abstract literal and the
 unit name of physical literals
 Related Issues: None
 Relevant LRM Sections: 3.1.3, 13.2
 Key Words and Phrases: Lexical Element, Physical Literal
 Level of Importance:
 Level of Difficulty:
 Current Status: Accepted

 Date Submitted: 31 October 1988
 Author of Submission: Alec Stanculescu
 Author's Affiliation: Vantage Analysis Systems, Inc.
 Author's Post Address: 42340 Christy Street, Suite 201
 Fremont, CA 94538
 Author's Phone Number: (415) 659-0901
 Author's Net Address:

 Date Analyzed: 7 November 1988
 Author of Analysis: Ray Hookway

Description of the Problem

The fourth paragraph of section 13.2 of the LRM states that:

"... At least one separator is required between an identifier or an abstract literal and an adjacent identifier or abstract literal."

Thus a space is always required between the abstract literal and the unit name in a physical literal. Nevertheless, many of the examples in the LRM incorrectly include physical literals which contain no space between the abstract literal and the unit name.

Example

```
-- LRM Section 10.3
-- Comments have been deleted
L1: block
    signal A,B: Bit;
    begin
    L2: block
        signal B: Bit;
        begin
            A<=B after 5ns;
            B<=L1.B after 10ns;
        end block;
        B<=A after 15ns;
    end block;
```

Proposed Resolution

Yes indeed, spaces are required between the abstract literal and the unit name. The examples are in error and should be changed.

VASG-ISAC Analysis & Rationale



Documents
Document Sharing Hub