

LRM issues to consider

##	Date	Section	Issue	Assigned	Source	Status	Resolved
1	030127		Clarify whether 'time 0' is start of simulation or reset state of design	HF	BC	Done	4-Oct
2	030212		Change "which" to "that" in a number of places		BC		
3	030220		Clarify whether the default vmode can inherit from other units	BC SR		Done	4-Oct
4	030304	4.2.1	Add next_event_a, next_event_e (weak versions) to table 1 (keywords)	SW	DB	Done	4-Oct
5	030318		Clarify whether 'cover {r}' is the same as 'cover {[*]; r}'	SW	HF	Done	4-Oct
6	030327	Tbl 2	Fix bad row headers - row 3 ("operatora"), row 4 ("sequence implication operators")		CE		
7	030328	1.3.2.1	Fix examples to use && instead of &; also add ';'.	BC HF		Done	4-Oct
8	030331	P2L12	Clarify that the example ena enb is a Verilog expression, and is the Verilog logical or		DW		
9	030505		Correct defn of inheritance - disallow 'inherit' in default vmode; further restrict vmode/vprop inheritance	HF – example SR – see also		Done	4-Oct
10	030615	6.1.1.1.6	6.1.1.1.6 definition of goto A[->c] : The count c does not have to be positive. MACROs may result an empty count, exactly like A[=0].		GV	Done	4-Oct
11	030615	6.1.2.1	6.1.2.1 in box 23 - sequence declaration : sequence_ParamKind : why not sequence_ParamTy ?		GV	Done	4-Oct
12	030324	4.2.2	Clarify precedence of * in section 4.2.2 (defined elsewhere)	?	BC	Open	
13	030404	4.2.2	Fix header "sequence implication operators" to read "sequence composition operators"	?	BC	Open	
14	030424		Use of hierarchical name in vunit is not consistent with Verilog usage	Defer	DS	Open	
15	030508		Clarify that replication (forall) variables cannot be used as repetition counts	Defer	AM	Open	
16	030619		I suggest we allow the usage of signal names which are identical to keywords, with a preceeding backslash : e.g. \clock.		GV		
17	030819	6.2.3	replicate example problem? http://www.eda.org/vfv/hm/1086.html	Proposal wor Defer		Open	Open

iii) Is there any order to the verification unit declarations?
 For instance, is a single pass of processing adequate?
 (inheritance of not yet processed vunits?)
 What if one unit is inherited by two different routes in one
 vunit? Is this permitted? If so, what about the declaration
 clashes it amounts to?

Example:

```
vunit ex1a {
  wire temp;
  assign temp = ack1;
}
```

```
vunit ex1b {
  inherit ex1a;
}
```

```
vunit ex1c {
  inherit ex1a, ex1b;
  assert temp;
}
```

i) SW ii) SW i

Open

19	030127		Clarify how cpp and % macros work with Verilog preprocessor macros (cpp chokes on single `) 4.2.3. Macros - %for, %if are mentioned here, but not defined elsewhere in the document. It must be clear that they are not part of any layer (probably "preprocessor layer"?) and may be used at any layer. It must be stated that cpp directives are precompiled *before* PSL macros (and in verilog flavour, also verilog directives like 'include and 'define). Add "step" keyword to the %for to allow decreasing range : %for i in 3 .. 0 step -1 do ...			CE	
20	030211		Make sure syntax allows "(posedge clk)" (and perhaps other Verilog event expressions) as clock expressions	Processing o		CE	Working
21	030304	1.3.2.1	Check example - is next[3] b allowed, or should it be next[3](b)? - see A.3.4 syntax	SW		DB	Working
22	030313		Clarify that {r} (f) describes _overlapping_ suffix implication (like ->)	SW		SW	Working
23	030402		Clarify that instantiation of a named property P is equiv. to body of P appearing inline in parentheses	SW		SW	Working
24	030406	P45L25	Move '@(clk)' up to reconstitute (a until! b)@clk	SW		CE	Working
25	030409	P50,52	Change EDL range .. to Verilog range syntax	SW		CE	Working
26	030409	index	Add "Simple Subset" to index	HF		CE	Working

27	030826		Is negative arguments to next-type operators allowed? According to appendix A in the LRM, the argument should be a 'Number' which in turn is defined as a 'integer_HDL_expression'. On the other hand in section 6.2.1.4, one gets the impression that only non-negative arguments are allowed especially the informal semantic explanation on page 47, line 50-52 and the note on page 48, line 8. If negative arguments are allowed what is the semantics when referring behind the starting point? (see also issue 1 above)		JA		
28	030826		Are counts for sequence repetition operators [=n] and [*n] allowed to be negative, and what are the semantics in that case? P. 98 in the LRM states that n should be a 'Number', i.e. an 'integer_HDL_expression'. However for goto repetition, it is explicitly stated that n should be positive -- shouldn't n for [=n] and [*n] be 'nonnegative_Number'?	JA		JA	Working
29	030826		Are names of verification units case sensitive or not?	JA		JA	Working
30	030826		p.80 fix the comments at the end of section 7.2: a vmode shall not contain (or inherit another vunit which contains.....) an assert directive. a vmode shall not inherit a vunit of type vmode or vprop. a vprop shall not contain (or inherit another vunit which contains.....) directive which are not assert directive. Excluding the following : a vprop may contain assume_gurantee and restrict_guarantee stmts (and they are treated as properties preceded by assert keyword. a vmode may contain assume_gurantee and restrict_guarantee stmts (and they are treated as assume/restrict)	SW – case st		GV	Working
31	030731	6.2.1.2.4	box 32 in section 6.2.1.2.4 requires parentheses for the operand of next and next!, but the grammar in appendix a (a.3.4, simple temporal operators) does not. this should be resolved for version 1.1 my recommendation: do not require parentheses for the simple case (next f, next f!). reason: this makes the pairs "next", "X" and "next!", "X!" interchangeable, which was the original intention.	a) SR – see a		CE	Working
32	030201		Clarify that b=[0:inf] does not make sense; disallow it via semantic restriction	SW – parens		CE	Working
33	030212	4.4.3	Clarify defn of 'safety' property - Is (req until ack) a safety property if ack never happens?	JA		BC	Remove 1-Oct
34	030422	6.2.1.4.2	Check defn of before!* - requires the left FL Prop (or should it be the right FL Prop?) to hold	Non-issue		JA	Remove 1-Oct
35	030427	4.2.2	Clarify precedence, and reference Verilog/VHDL operators	Non-issue		BC	Remove 1-Oct
36	030615	3.1.39	3.1.39 non-zero range : "range" applies both to sub-vectors and to time-range. [5..4] is zero-range if it describes time slots, but not boolean sub-vectors. there we want both directions [4..5] and [5..4] to be legal.	BC		GV	Remove 1-Oct
				GV			Remove 1-Oct

40 030826

In 4.2.3 the PSL macro layer is defined. However, the information is extremely sparse about what the semantics really is. This needs substantial clarification.

JA

Examples of what needs to be clarified:

1) What tokens are possible for /var/, /item/ and so on?
Is for instance the following OK?

```
%for directive in { assert, cover } do
    directive my_property;
%end
```

2) What is meant by cpp-style? (LRM says "All flavors support cpp-style pre-processing directives")
cpp works differently for C89 and C99, for instance.

Relation to VHDL/GDL comments, what is the semantics of

```
#define foo bar
%for interface in 1..2 do
-- This is a comment regarding the foo interface
%end
```

Possible interpretations:

i)
-- This is a comment regarding the foo interface
-- This is a comment regarding the foo interface

ii)
-- This is a comment regarding the bar interface
-- This is a comment regarding the bar interface

iii)

41 030826

next is reserved word in two layers : temporal and modeling. make sure it is consistent. can next(expr) in GDL return a non-binary value ?

Non-issue, al Remove 1-Oct

42 030826

clock is a bad reserved word. I suggest using default_clock, and default_vunit, and remove clock and default from the keywords.

GV Remove 1-Oct

43 030826

p.79 7.2 ,Box74 shows that Inherit stmts come before Vunit_Items. since the order of stmts is not important in any other context, I suggest it will not be important here too. also the current syntax does not allow two inherit keyword statements. The syntax should be changed to
Inherit_Spec | Vunit_Item { , Inherit_Spec | Vunit_Item }

Dup of #35 Remove 1-Oct

GV Remove 1-Oct