

Extension subcommittee issues to consider

##	Date	Section	Issue	Resolved	Who
1			Allow specification of error message and severity on assertions		EM
2			Allow assertion failure to be 'caught' by higher-level assertion and reported differently		EM
3			Allow an assert directive to include a unique name -- Propose the optional "label" statement, similar to SystemVerilog		EM
4			Allow {r} (f) to be expressed as {r} -> (f) for consistency	Done Alig	EM
5			Allow data values to be sampled during recognition of a sequence, and used later		EM
6	030217		Allow VHDL flavor assertions to read ports of mode out		BC
7	030309		Provide a better way to express non-determinism - new operators, or function calls ...		CE
8	030324		Define a standard way of embedding PSL assertions in HDL source		BC
9	030324		Clarify that all built-in functions (e.g., rose, fell) are defined for all flavors of PSL		BC
10	030324		Allow VHDL flavor assertions to read OUT ports		BC
11	030331		Allow sampling of expressions within PSL, and allow checking for previous values of a signal		MC
12	030331		Allow detection of signal changes "without using default clock construct"		MC
13	030416		Define a way to give TB to access coverage info from PSL - allow "write COVERAGE_PROP signal"		BC
14	030427		Add 'trans' stmt to EDL (now GDL) flavor (specifying transition constraint?)		SR
15	030428		Consider interpreting X and Z pessimistically rather than as in an if condition		CE
16	030430	5.4	Allow more than one default clock declaration (last one overrides)		BC
17	030506		Allow use of Verilog `define in a separate PSL file (add boolean declaration)		RN
18	030528		Add default "assert" (email discussion indicated group opposed this)		BC
19	030612		The lack of explicit instantiation construct for vunits limits the kinds of hierarchical sets of properties that can be constructed. http://www.eda.org/vfv/hm/1060.html		BC
20	030615	4.2.4	4.2.4 comments - it is advisable that EDL flavor will support both kinds of comments. // and -- and /* reasoning : verifying a golden model in EDL, the using the same files for the real verilog design. should we add definition of smart comments (like // synopsys full_case in verilog) or let each vendor choose his ?		GV

21	030615	6.1.1.1	<p>6.1.1.1 - The definition of clocked property ("extracting from a given path exactly....") is fine as long as the clock is a boolean event. Generally, it is incorrect following the real design clock, e.g. when we use @posedge(clk). The simplest example is describing the behaviour of a single Flip-Flop, where ff_d is the input of the FF, ff_q is the output of the FF, and it is gated by the clock "clk", and the input ff_d is driven from a faster clock, thus may change in between the ticks where clk is active. property posedge_triggered_flip_flop { G (ff_d = 0) -> X (ff_q = 0) } @ posedge(clk)</p> <pre> time 0 1 2 3 4 5 6 7 clk 0 0 1 1 0 0 1 1 posedge(clk) 0 0 1 0 0 0 1 0 ff_d 1 1 0 1 1 1 1 1 ff_q 1 1 1 1 1 1 1 1 </pre> <p>The clocked formula extracts only cycles #2 and #6, where the formula does not hold. Also see: http://www.eda.org/vfv/hm/1066.html</p>	GV
22	030621		Default abort construct: http://www.eda.org/vfv/hm/1064.html	BC
23	030826		define an "executable" vunit (one which can be run, executed, and get results for all its assertions, vs. a vunit which serves as a vmode only or has named properties (and no instantiations)).	GV
24			Demoting / relaxing flavor macros	
25	030612		Request for clarification in LRM related to vunits/vprops/vmodes http://www.eda.org/vfv/hm/1060.html	DC
26	030615	4.4.5	<p>4.4.5 - Simple subset - bad name. does not say what it means. may I suggest OTF : "On-The-Fly subset ?" - Here we detail only the FL properties requirements to be OTF-subset. What about CTL props ? some of them are still "simple".</p>	GV
27	030615	5.1	<p>- multiple definitions of same name - I suggest full hierarchy, and scope of definition exactly like "C" language. The name of an identifier, if ambiguous, is resolved from the innermost vunit that was inherited. (E)</p>	GV
28	030615	6.2.1.6.5	6.2.1.6.5 Logical NOT, restrictions : the operand must be boolean. why ? "!F(p)" is exactly "G(!p)" and can be computed OTF.	GV