1.1 Access to sampled values

This proposal addresses the need to access sampled values of variables consistent with the sampling of variables specified in assertions.

The method currently available is to declare a clocking domain with the intended clock, specify required variables in the clocking domain, and access the sampled value by the variable name prefixed with the clocking domain name. For example,

```
clocking_domain gclk @(posedge sysclk);
   input a,b,c
endclocking
property p1;
   @gclk a ##1 b ##1 c;
endproperty;
assert (p1) $display("success for p1, a = %d;",gclk.a);
```

To allow users a direct access to sampled variable values without the use of explicit clock domain, a system function $sampled is proposed. This function can be invoked anywhere in the design to access a variable's sampled value with respect to a clock.

1.1.1 Syntax of $sampled function

sampled_function ::=

$sampled (expression[**,** event_expression] **) ;**

1.1.2 Semantics of $sampled function

expression must comply with the restrictions on the expression usage in assertions. Refer to Section 17.4 for details on the restrictions.

Optionally, clocking for the expression is specified by event_expression. The value of the expression is sampled by event_expression. event_expression may be omitted in two cases:

— $sampled is used in an action block of a singly clocked assertion. The clock of the assertion is inferred for the function.

— $sampled is used in a block for which default clocking is specified. The default clocking is inferred for the function.

If clocking is specified, no inference of clocking is applied.

$sampled may be used anywhere in the code as a function call. The value returned by $sampled is the latest sampled value for the expression.

1.1.3 Example for $sampled

The example presented in Section 1.4 is rewritten using $sampled below:

```
property p1;
   @gclk a ##1 b ##1 c;
endproperty;
assert (p1) $display("success for p1, a = %d;",$sampled(a));
```

In this case, the sampled value of a is with respect to @gclk.

```
task out_1;
   out1 = b && $sampled(a, posedge clk); //samples with respect to posedge clk
   -
   -
endtask;
```

