16.8.4 Global clocking past and future sampled value functions

(Note to the editor: Please shift the subsequent clauses accordingly.)

This subclause describes the system functions available for accessing the nearest past and future values of an expression as sampled by the global clock. *They can only be used when global clocking is defined* (see Editor please insert reference to global clocking). These functions include the capability to access the sampled value at the global clock tick that immediately precedes or follows the timestep at which the function is called. Sampled value is explained in 16.4. The following functions are provided:

Global clocking past functions:

```
$past_gclk(expression)
$rose_gclk(expression)
$fell_gclk(expression)
$stable_gclk(expression)
$changed_gclk(expression)
```

Global clocking future functions:

```
$future_gclk(expression)
$rising_gclk(expression)
$falling_gclk(expression)
$steady_gclk(expression)
$changing_gclk(expression)
```

The behavior of the global clocking past sampled-value functions can be defined using the sampled-value functions as follows:

```
$past_gclk(v)     ≡  $past(v,,,  @$global_clocking)
$rose_gclk(v)     ≡  $rose(v,    @$global_clocking)
$fell_gclk(v)     ≡  $fell(v,    @$global_clocking)
$stable_gclk(v)   ≡  $stable(v,  @$global_clocking)
$changed_gclk(v)  ≡  $changed(v, @$global_clocking)
```

The global clocking future sampled-value functions are similar except that they use the subsequent value of the expression.
`$future_gclk(v)` is the sampled value of `v` at the next global clocking tick.
The other functions are defined as follows:

— `$rising_gclk(expression)` returns true if the sampled value of the LSB of the expression is changing to 1 at the next global clocking tick. Otherwise, it returns false.

— `$falling_gclk(expression)` returns true if the sampled value of the LSB of the expression is changing to 0 at the next global clocking tick. Otherwise, it returns false.

— `$steady_gclk(expression)` returns true if the sampled value of the expressions does not change at the next global clock tick. Otherwise it returns false.

— `$changing_gclk(expression)` is the complement of `$steady_gclk`, i.e., `!$steady_gclk(expression)`.

The use of these functions is limited to assertion features only. It shall be an error to invoke these functions outside of property expressions. The global clocking past value functions are a special case of the sampled value functions, and therefore the regular restrictions imposed on the sampled value function arguments apply (see 16.8.4). Additional restrictions are imposed on the usage of the global clocking future value functions: they shall not be nested and they shall not be used in assertions containing sequence match items (see 16.9, 16.10). The following example illustrates the illegal usage of the global clocking future value functions:

Example.

```
// Illegal: global clocking future value functions shall not be nested
a1: assert property (@clk $future_gclk(a || $rising_gclk(b));

sequence s;
   bit v;
   (a, v = a) ##1 (b == v)[->1];
endsequence : s

// Illegal: a global clocking future value function shall not be used in
// an assertion containing sequence match items
a2: assert property (@clk s |=> $future_gclk(c));
```

The main reason for introducing the next value functions is their efficiency in formal verification: the performance of many formal verification tools is better when the next values of the variables are referenced in the assertions, and not the past ones. Also, using the next value functions may sometimes make assertions more intuitive. In formal tools the next value is the sampled value at the next global clocking tick.

An action block of an assertion containing next value functions is performed at the global clocking tick that follows the assertion clock tick at which the final boolean expression of the assertion is evaluated. This allows the evaluation of the next value functions to be delayed until after the next values of the signals referenced have been computed.

Example 1:

The Table 16-25 (Note to the editor: Please adjust numbering as needed) shows the values returned by the next value functions for `sig` at different time moments.

The following assertion states that the signal may change only on falling clock:

```
a: assert property (@$global_clock $changing_gclk(sig)
                                   |-> $falling_gclk(clk))
   else $error("sig is not stable");
```

In the Figure 16-4 we see that this property is violated at time 80. The vertical arrows indicate the ticks of the global clock. The error message `error("sig is not stable")` is executed at time 90.

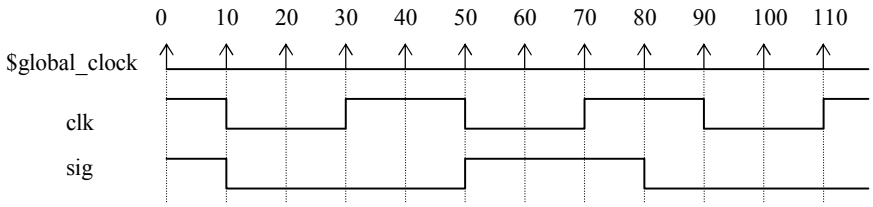

Figure 16-4 Future value change

Table 16-25: Future value functions

Time	$sampled(sig)	$future_gclk(sig)	$rising_gclk(sig)	$falling_gclk(sig)	$changing_gclk(sig)	$steady_gclk(sig)
10	1'b1	1'b0	1'b0	1'b1	1'b1	1'b0
30	1'b0	1'b0	1'b0	1'b0	1'b0	1'b1
40	1'b0	1'b0	1'b0	1'b0	1'b0	1'b1
50	1'b0	1'b1	1'b1	1'b0	1'b1	1'b0
80	1'b1	1'b0	1'b0	1'b1	1'b1	1'b0

Example 2:

The following assumptions states that a signal `sig` must remain stable between two falling edges of a clock `clk` as sampled by global clocking. This differs from the property in Example 1 in the case where the first falling edge of `clk` has not yet occurred. In Example 1, `sig` is not allowed to change in that case, but in this example `sig` can toggle freely while we wait for `clk` to begin.

```
a: assume property(@$global_clocking
          $falling_gclk(clk) ##1 (!$falling_gclk(clk)[*1:$]) |->
                                                $steady_gclk(sig));
```

19.11 Assertion system functions

Note to the editor - add the following text at the end of this sub-clause:
The following functions allow to access the sampled value of an expression at the immediate past and future ticks of the global clock and to detect changes in the sampled value from the past (resp. current) tick of the global clock to its current (resp. next) tick.

Global clocking past functions:

```
$past_gclk(expression)
$rose_gclk(expression)
$fell_gclk(expression)
$stable_gclk(expression)
$changed_gclk(expression)
```

Global clocking future functions:

```
$future_gclk(expression)
$rising_gclk(expression)
$falling_gclk(expression)
$steady_gclk(expression)
$changing_gclk(expression)
```

These functions are discussed in 16.8.4.