9.5.1 Introduction

As an initial example, the function $sampled(expression) returns the value of the expression using values for the expression's variable(s) from the preponed region. This function can be used anywhere but is not allowed in synthesizable models. The following initial block illustrates how sampled value functions work.

```
1  initial begin
2    b = 0;
3    @(posedge ck);
4    b = 1;
5    $display ("%b %b", b, $sampled(b));
6  end
```

The block assumes that there is a clock that is oscillating; this clock is waited for at line 3. When the initial block is executed, the $display statement will print "1 0" because the current value of b was set on the line just previous to the $display — it prints as 1. However, since the sampled value comes from the preponed region of the kernel, the value of b is 0 because it was sampled just before the clock edge occurred.

Of course the $sampled() function isn't very interesting by itself. The whole list of sampled value functions is shown below. ce stands for clock event (i.e., something like "@ (posedge ck)"):

- *$sampled (expression)* — The value of the expression using values sampled in the current time's preponed region.
- *$rose (expression, ce)* — TRUE if the least significant bit of the expression changed to 1 from the previous clock event to the current clock event. FALSE otherwise.
- *$fell (expression, ce)* — TRUE if the least significant bit of the expression changed to 0 from the previous clock event to the current clock event. FALSE otherwise.
- *$stable (expression, ce)* — TRUE if the value of the expression remained the same from the previous clock event to the current clock event. FALSE otherwise.
- *$changed (expression, ce)* — TRUE if the value of the expression changed from the previous clock event to the current clock event. FALSE otherwise.
- *$past (expression, numTicks, ce)* — The value of the expression using values sampled numTicks in the past, not counting the samples taken in the current clock time.

The clocking event (ce) specified in some of the functions is an optional specification. If these functions are used as part of evaluating an assertion, the property's clock is inferred and ce need not be specified. If these are used in other procedural blocks, such as a testbench, the clocking event needs to be specified using a clocking block as shown in Example 9.3.

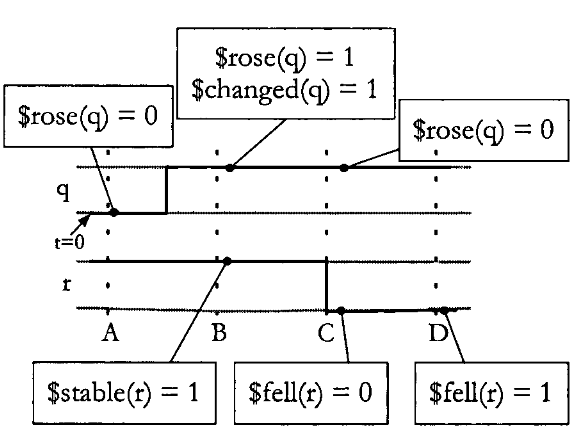

Figure 9.11 — Sampled Value Functions