```
module fsm(...);
  function bit foo(bit a, bit b, …)
      ...
      a1: unique if (a)
          ...
      else if (b)
          ...
  end
  ...
  always_comb begin : b1
      some_stuff = foo(c, d, …);
      ...
  end
  always_comb begin : b2
      other_stuff = foo(e, f, …);
      ...
  end
endmodule
```

In this case, there are two different processes which may call assertion a1: b1 and b2. Suppose simulation executes the following scenario in the first passage through the Active region of each time step:

In time step 1, b1 executes with c=1 and d=1, and b2 executes with e=1 and f=1.
In the first time step, since a1 fails independently for processes b1 and b2, its failure is reported twice.

In time step 2, b1 executes with c=1 and b=1, then again with c=0 and d=1.
In the second time step, the failure of a1 in process b1 is flushed when the process is re-triggered, and since the final execution passes, no failure is reported.

In time step 3, b1 executes with c=1 and d=1, then b2 executes with e=0 and f=1.
In the third time step, the failure in process b1 does not see a flush point, so that failure is reported. In process b2, the violation check passes, so no failure is reported from that process.