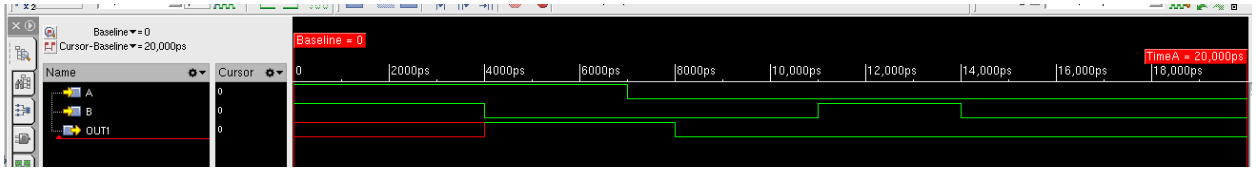
Yihao Wang 7410-1780-57

Model 1: Procedural blocking assignment with inter-statement delay
Module "or2_1" manifested this type of delay

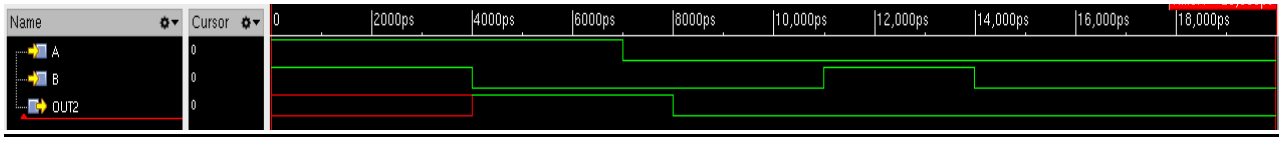


The first delay model is blocking assignment with inter-statement delay. This type of delay will delay entire blocking assignment including evaluation of RHS and assignment to LHS. The whole process will be explained as follows:

Time 0ns:
always block is triggered by valid A and B;
4ns delay;
Time 4ns:
Calculation of RHS started (A=1, B=1), result is 1'b1;
Assigned 1'b1 to OUT1;
Next always cycle was triggered because B changed from 1 to 0;
4ns delay;
Time 8ns:
Calculation of RHS started (A=0, B=0), result is 1'b0;
Assigned 1'b0 to OUT1;
Wait for next triggering;
Time 11ns:
Next always cycle triggered because B changed from 0 to 1;
4ns delay;
Time 15ns:
Calculation of RHS started (A=0, B=0), result is 1'b0;
Assigned 1'b0 to OUT1;
Waited for next triggering.

Model 2: Procedural non-blocking assignment with inter-statement delay

Module "or2_2" manifested this type of delay



The second model is non-blocking assignment with inter-statement delay. The inter-statement delay will delay the whole statement including both calculation of RHS and assignment of LHS. Non-blocking assignment means assignment started after RHS calculation of all non-blocking assignment and both LHS assignment and RHS calculation of all blocking assignment. But since there is only one assignment statement executed in each time step, there is no difference in the output of both model 1 and model 2.

The detailed mechanism will be explained as follows:

Time 0ns:

Always block is triggered by valid A and B;
4ns delay;

Time 4ns:

Calculation of RHS started (A=1, B=1), result is 1'b1; (since there is no other assignments at this time step, so assignment of LHS started immediately)
Assigned 1'b1 to OUT2;
Next always cycle was triggered because B changed from 1 to 0;
4ns delay;

Time 8ns:

Calculation of RHS started (A=0, B=0), result is 1'b0;
Assigned 1'b0 to OUT2;
Wait for next triggering;

Time 11ns:

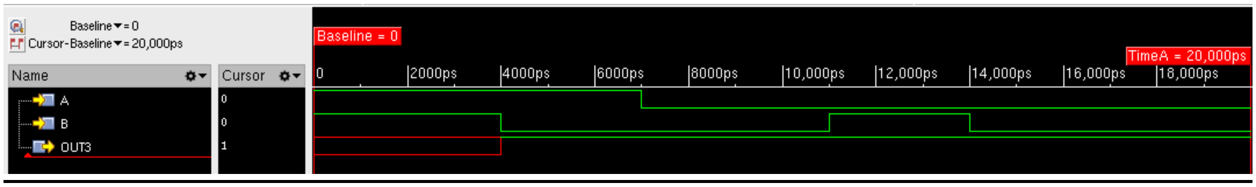
Next always cycle triggered because B changed from 0 to 1;
4ns delay;

Time 15ns:

Calculation of RHS started (A=0, B=0), result is 1'b0;
Assigned 1'b0 to OUT2;
Waited for next triggering.

Model 3: Procedural blocking assignment with intra-statement delay

Module "or2_3" manifested this type of delay



Intra-statement delay will only delay the assignment of LHS but won't delay the calculation of RHS. "OUT3 = #4 A | B" means assignment of OUT3 starts 4ns after calculation of "A|B" finished.

The mechanism will be explained as follows:

Time 0ns:

(The default value of A and B are 1'bx before assignment)
Always block is triggered by valid A and B;
Calculation of RHS started (A=1, B=1), result is 1'b1;
4ns delay;

Time 4ns:

Assigned 1'b1 to OUT3;
Next always cycle was triggered because B changed from 1 to 0;
Calculation of RHS started (A=1, B=0), result is 1'b1;

Time 8ns:

Assigned 1'b1 to OUT3;
Wait for next triggering;

Time 11ns:

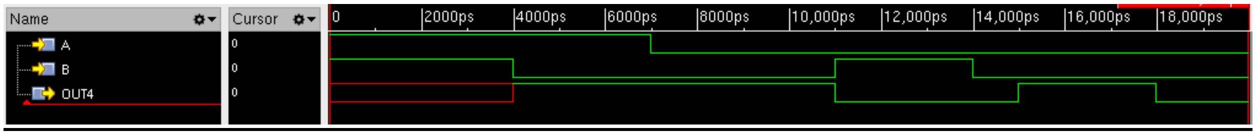
Next always cycle was triggered because B changed from 0 to 1;
Calculation of RHS started (A=0, B=1), result is 1'b1;
4ns delay;

Time 15ns:

Assigned 1'b1 to OUT3;
Waited for next triggering.

Model 4: Procedural non-blocking assignment with intra-statement delay

Module "or2_4" manifested this type of delay



The difference between model 3 and model 4 is that non-blocking has assignment scheduling. Take "OUT4 <= #4 A | B" as an example, calculation of "A | B" happened at time 0 but simulator just scheduled assignment of LHS at time 4ns and then continue to execute the next statement. Whereas if there is a blocking assignment ("OUT3 = #4 A | B"), the calculation of RHS started at time 0 and assignment of LHS happened at time 4ns. The most important is the next statement must be executed after the LHS assignment finished, so that is to say, there is no assignment scheduling in blocking assignment.

The explanation of this mechanism is as follows:

Time 0ns:

(The default value of A and B are 1'bx before assignment)
Always block is triggered by valid A and B;
Calculation of RHS started (A=1, B=1), result is 1'b1;
The assignment of OUT4 was scheduled at time 4ns;
Waited for next triggering;

Time 4ns:

Assigned 1'b1 to OUT4 (scheduled at time 0);
The next always cycle was triggered because B changed from 1 to 0;
Calculation of RHS started (A=1, B=0), result is 1'b1;
The assignment of OUT4 was scheduled at time 8ns;
Waited for next triggering;

Time 7ns:

The next always cycle was triggered because A changed from 1 to 0;
Calculation of RHS started (A=0, B=0), result is 1'b0;
The assignment of OUT4 was scheduled at time 11ns;
Waited for next triggering;

Time 8ns:

Assigned 1'b1 to OUT4 (scheduled at time 4ns);

Time 11ns:

Assigned 1'b0 to OUT4 (scheduled at time 7ns);
Next always cycle was triggered because B changed from 0 to 1;
Calculation of RHS started (A=0, B=1), result is 1'b1;
The assignment of OUT4 was scheduled at time 15ns;
Waited for next triggering;

Time 14ns:

Next always cycle was triggered because B changed from 1 to 0;

Calculation of RHS started (A=0, B=0), result is 1’b0;

The assignment of OUT4 was scheduled at time 18ns;
Waited for next triggering;

Time 15ns:

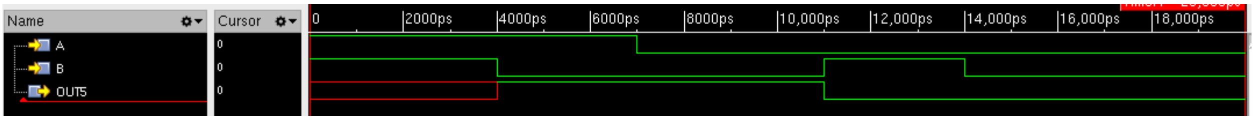
Assigned 1’b1 to OUT4 (scheduled at time 11ns);

Time 18ns:

Assigned 1’b0 to OUT4 (scheduled at time 14ns);

Model 5: Continuous assignment

Module “or2_5” manifested this type of delay



Continuous assignment means the LHS variable is continuously driven by LHS. If you declare delay in continuous assignment, it means the update of LHS happens several ns delay after the LHS signal changes. Take “assign #4 OUT5 = A | B” as an example, if A or B changes at time 0, the value of LHS starts to update at time 4ns and then continuous assignment happens at the same time step (time 4ns). But if A or B changes back to its original value before 4ns, it won’t have any effect on the outputs. In brief, in order to make the output change, the changing of input signal must last for at least 4ns.

The explanation of mechanism of above waveform is as follows:

Time 0ns:

(The default value of A and B are 1’bx before assignment)
A and B changed to its valid value;
Updating of LHS will start at time 4ns;

Time 4ns:

The updating of LHS happens, “A | B” is evaluated to be 1’b1 (A=1, B=1);
Assigned 1’b1 to OUT5;

Time 7ns:

A changed from 1 to 0;
Update of LHS will start at time 11ns;

Time 11ns:

The updating of LHS happens, “A | B” is evaluated to be 1’b0 (A=0, B=0);
Assigned 1’b0 to OUT5;

Time 14ns:

B changed from 1 to 0;
Updating of LHS will start at time 18ns;

Time 18ns:

The updating of LHS happens, “A | B” is evaluated to be 1’b0 (A=0, B=0);
Assigned 1’b0 to OUT5;