IN3160: Oblig3

Trym Auren

Februar 2023

Contents

a

The output data signal changes at 450ns. During run: - `mclk` (clock) switch between low and high. One clock cycle lasts for 100 ns.

1. At start of run: rst_n (reset) is '0', switches to '1' after 100 ns.
   Elapsed time: 100 ns.
2. Clock has gone through 1 cycle during this time. Since reset has been '0' throughout, `data(1,2,3,4,5)` has been assigned the value '0'.
   Elapsed time: 100 ns.
3. When reset is now set to high (1) after 100 ns, the if- test `elsif rising-edge` is executed two times (because of two elapsed clock- cycles), but `data(1,2,3,4,5)` is still '0', because `indata` is '0'.
   Elapsed time: 200 ns.
4. Now `indata` is assigned "11110000". Since reset is '1', `data1` is set to `indata = "11110000"` when the bulletpoint above (the previous process) is finished.
   Elapsed time: 200 ns.
5. This time, `data2` is set to `data1` immediately, since its a variable. At the end of the process, `data3` is set to `data2`.
   Elapsed time: 300 ns.

6. This time, `data4` is set to `data3` immediately, since its a variable. At the end of the process, `data5` is set to `data4`.
   Elapsed time: 400 ns.

7. When the next process is done (this is normally done after 50 ns + minor delay), `data5` is finally the same value as `"11110000"`, and outdata is set to this value. This happens at 450 ns.
   Elapsed time: 500 ns.

Note: Elapsed time: xx ns does not mean elapsed time right after previous described action, but is more of an index.

b

The output data signal changes at 750 ns.
The output data signal is equal to "UUUUUUUU" at 50 ns, because the signal is not set before 100 ns.

c

`output(7 downto 6)` is always equal to `output(3 downto 2)`, because both standard logic vectors are set to the same signal, and the value of the signal assigned to `output(7 downto 6)` is not changed before after the process is done with an iteration.
`output(5 downto 4)` is always different from `output(1 downto 0)`, because the vectors are assigned a variable, which is updated between the update of `output(1 downto 0)` and `output(5 downto 4)`.
I.e., the difference lies in changing value of variables, which happens when the process is running, versus changing value of signals.

d

The question boils down to what a sensitivity list is, and how the process is dependent on the list. When `sig1` and `sig2` is removed from the sensitivity list, the process is only invoked when there is a change in `indata` - the last parameter remaining. Since nothing changes to `indata` before after 100 ns (and then after 200 ns), the process will not run until at 100 ns (and at 200 ns). At task **c**, the process is triggered immediately at startup, because it listens to `sig1` and `sig2`.

Figures

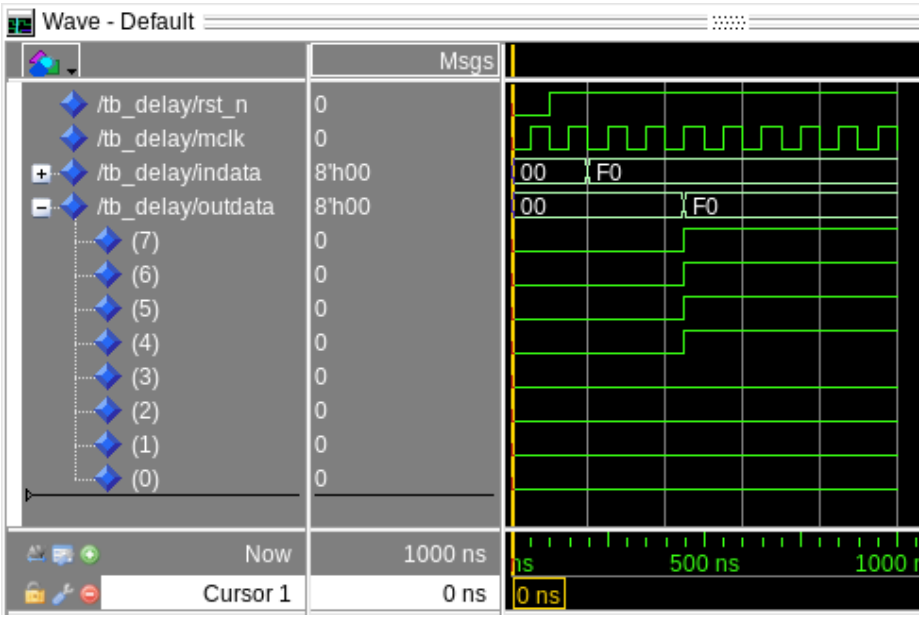

Figure 1: Task a

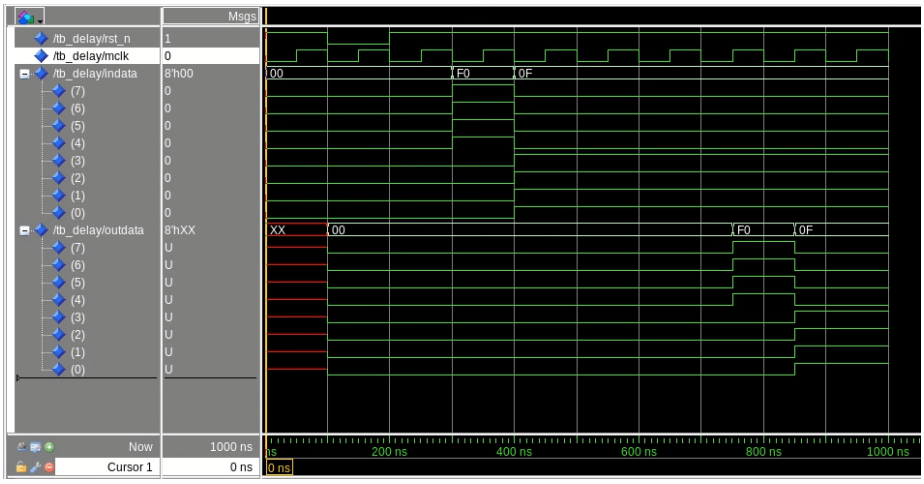

Figure 2: Task b

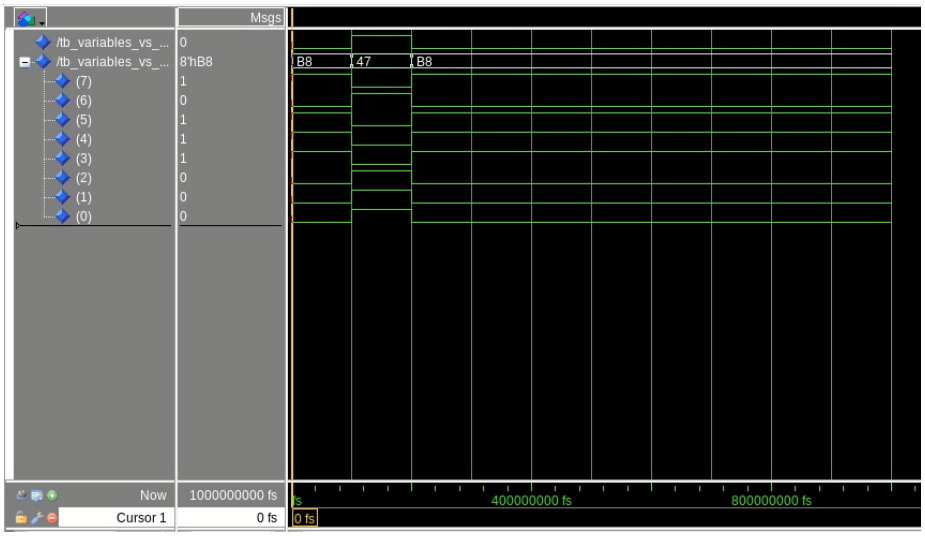

Figure 3: Task c

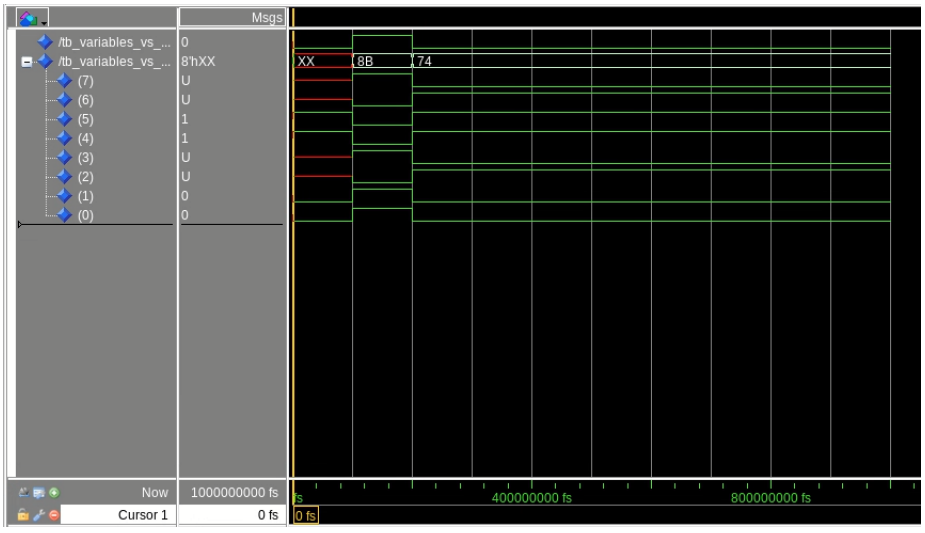

Figure 4: Task d