Logika cyfrowa

Praktyczna lista zadań nr 7

Termin: 24 kwietnia 2024 godzina 30:00

Uwaga! Poniższe zadania należy rozwiązać przy użyciu języka SystemVerilog, sprawdzić w DigitalJS oraz wysłać w systemie Web-CAT na SKOS. Należy pamiętać, aby nazwy portów nadesłanego modułu zgadzały się z podanymi w treści zadania. Wysłany plik powinien mieć nazwę `toplevel.sv`. **Nie przestrzeganie tych zasad będzie skutkować przyznaniem 0 punktów.**

1. Zaimplementuj układ czterobitowego licznika synchronicznego, którego wartość w każdym cyklu zegara zwiększa się lub zmniejsza o 1 lub o 2. Możesz w tym celu wykorzystać dowolny wybrany rodzaj przerzutnika. Układ powinien mieć następujące wejścia i wyjścia:

   - `clk` – wejście sygnału zegara (wyzwalanie zboczem narastającym),
   - `nrst` – wejście sygnału resetu asynchronicznego (zanegowane – stan niski oznacza reset),
   - `step` – stan niski oznacza krok 1, stan wysoki – krok 2,
   - `down` – stan niski oznacza odliczanie w górę, wysoki – w dół,
   - `out` – czterobitowe wyjście, stan licznika.

   Kod powinien specyfikować model bramkowy – nie używaj arytmetyki SystemVerilogа. Dozwolone są multipleksery specyfikowane przez wyrażenia warunkowe.

   Rozwiązanie można przetestować przy użyciu poniższego skryptu Lua.

```
sim.setinput("nrst", 0)
sim.sleep(100)
assert(sim.getoutput("out"):tointeger() == 0, "Error: reset failed")
sim.setinput("nrst", 1)
sim.sleep(100)
sim.wait(sim.posedge("clk"))
sim.sleep(25)
cnt = sim.getoutput("out"):tointeger()
for x = 1, 100 do
  local step = math.random(0,1)
  local down = math.random(0,1)
  local nextcnt = (cnt + (1 - down * 2) * (step + 1)) % 16
  sim.setinput("step", step)
  sim.setinput("down", down)
  sim.wait(sim.posedge("clk"))
  sim.sleep(25)
  local newcnt = sim.getoutput("out"):tointeger()
  assert(newcnt == nextcnt,
    "Error: step=" .. step .. " down=" .. down .. " previous=" .. cnt ..
    " expected=" .. nextcnt .. " actual=" .. newcnt)
  cnt = newcnt
end
print("OK!")
```