Digitale Schaltungstechnik, WS 2019/2020

4. Übungsblatt — Abgabe: 19.11.2019

1 Simulation

Simulationen sind ein wichtiges Werkzeug, um Schaltungen zu entwickeln und Fehler zu beheben, ohne sie z.B. auf einem FPGA ausprobieren zu müssen. Durch eine Simulation können Sie untersuchen, wie sich jedes Signal verhält! Bevor Sie eine Schaltung simulieren können, müssen Sie zunächst eine *Testbench* schreiben.

Eine Testbench wird, wie Ihr Design, in Verilog geschrieben. Im Gegensatz zu Ihrem Design muss Ihre Testbench aber nicht synthetisierbar sein, d.h. es gibt wahrscheinlich kein Hardware-Gegenstück dazu.

Simulieren Sie die in der vorigen Aufgabe implementierte Schaltung:

a) Eine Vorlage für eine Testbench finden Sie auf der Vorlesungshomepage. In der Vorlage wird ein Taktsignal und ein Resetpuls für ein Design erzeugt.

   Fügen Sie die Testbench zum Projekt hinzu, und setzen Sie unter `Source Properties` die `View Association` auf Simulation. Sie finden die Testbench jetzt unter `View: Simulation`.

b) Bearbeiten Sie die Testbench, um Ihr Design hinzuzufügen und benötigte Stimuli (z.B. einen Tastendruck) zu generieren.

c) Sie starten die Simulationsumgebung dann durch Auswahl der Testbench und Doppelklick auf `Simulate Behavioral Model`. Stellen Sie sicher, dass in den Projektoptionen `ISim` als Simulator ausgewählt ist.

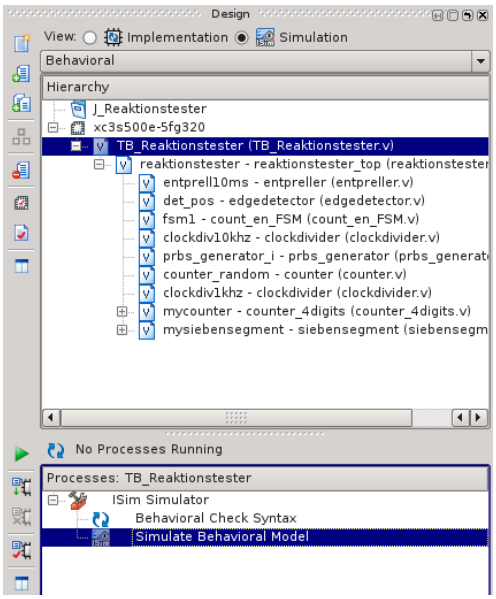

Abbildung 1: Hierarchische Ansicht des Projekts im linken Bereich des ISE-Programmfensters

Eine Einführung in die Oberfläche des Simulators finden Sie unter → `http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/ug682.pdf`.

Denken Sie an definierte Startwerte der Register oder eine Reset-Schaltung, um undefinierte Signale im Simulator zu verhindern. Eventuell müssen Sie auch Zeitparameter Ihres Designs in der Simulation anpassen, da sie sonst unnötig lange dauert.

2 Lauflicht

Benutzen Sie Ihren Taktteiler, um ein Lauflicht zu erzeugen, bei dem die acht LEDs in wählbarer Geschwindigkeit reihum nacheinander aufleuchten sollen:

a) Realisieren Sie das Lauflicht mithilfe eines Schieberegisters, dessen einzelne Einträge direkt die LEDs ansteuern (verwenden Sie den Schiebeoperator „<<“ bzw. „>>“). Verwenden Sie ein `if/else`-Konstrukt, um sicherzustellen, dass

   - das Licht von der letzten wieder in die erste Position springt und
   - durch das Drücken eines Knopfs der Inhalt des Schieberegisters auf einen sinnvollen Anfangswert gesetzt wird *(Reset)*.

b) Erstellen Sie eine zweite Version des Lauflichts, bei dem die Position des Lichts aus einer Zählervariable dekodiert wird.

   - Wieviele Bits muss die Zählervariable haben?
   - Benötigen Sie auch hier einen Reset-Knopf?

c) Bauen Sie die Möglichkeit ein, die Richtung des Lauflichts mit einem Schiebeschalter einzustellen.