Aufgabe 1: Hardwarebeschreibungssprachen

Aufgabe 2: Schaltungsentwurf mit VHDL

In dieser Aufgabe ging es darum einfache Schaltungen mit VHDL als Struktur- und Verhaltensbeschreibung zu implementieren. Dies stellte uns vor keine große Herausforderung, da wir die Vorlesung "Rechnerstrukturen" bereits gehört haben und daher schon mit den Grundlagen vertraut waren. Dennoch traten einige Probleme auf, die erst beim Simulieren und/oder Synthetisieren auffielen.

2.1 AND, OR und NOT

Dieser Aufgabenteil stellte uns aufgrund unserer Erfahrung vor keine großen Schwierigkeiten.

2.2 NAND, NOR

Bei diesem Aufgabenteil kam es zu einem Verständnisproblem. Wir wussten nicht, wie im Falle einer Strukturbeschreibung bei der Simulation und/oder Synthese die Verbindung zwischen Strukturbeschreibung, Entities und Implementierung der Unterkomponenten (Verhaltensbeschreibung) zustande kommt. Aus diesem Grund fügten wir vorerst die Entity-Deklaration zusätzlich in die Dateien der Strukturbeschreibung ein, was in Aufgabe 4 (Synthese der NAND-Strukturbeschreibung) zu einem Fehler führte, da die Entities für AND und NOT in mehreren Dateien deklariert wurden. Dann verstanden wir, dass die Reihenfolge der Kompilierung den Abhängigkeiten entsprechen muss, damit die Strukturbeschreibung Verweise auf Entities und Verhaltensbeschreibung erhält.

2.3 Impuls

Hier kam es erneut zu einem Verständnisproblem. Um den internen Zustand (also den Stand des Zählers) zu speichern verwendeten wir vorerst eine Variable. Diese wurde natürlich bei jedem Aufruf des Prozesses zurückgesetzt, da Variablen nur innerhalb von Prozessen existieren können und dort auch der Initialwert gesetzt wird. Somit war eine Variable nicht sinnvoll, um den Zählerstand zu speichern. Das Problem haben wir behoben, indem wir ein Signal anstatt einer Variable benutzten.

2.4 Zähler

Beim Implementieren des Zählers war die Schwierigkeit in der fallenden Taktflanke den Ausgang wieder auf 0 zu setzen, falls er vorher auf 1 war (somit ist der Ausgang jeweils gleich lang 1 und 0, wie in der Aufgabenstellung gefordert). Dies haben wir gelöst, indem wir ein Signal "was_1" eingeführt haben, dass den letzten gesetzten Ausgang speichert. Der Code funktioniert in der Simulation wie gedacht (bei Zähler-Überlauf wird einen halben Takt 1 und einen halben Takt 0 ausgegeben), bei der Synthese gibt es allerdings einen Fehler. Der FPGA unterstützt die Abfrage "if clk'event then" nicht (siehe: `https://www.xilinx.com/support/answers/17023.html`). Da der Erfolg der Synthese hier aber nicht gefordert war und die Simulation funktioniert, ist die Aufgabe gelöst.

Aufgabe 3: Simulation

3.1 GHDL

Die Schwierigkeit bei dieser Aufgabe bestand darin, GHDL zu installieren. GHDL und viele Abhängigkeiten von GHDL sind nicht mehr in den offiziellen Paketquellen vorhanden und GHDL war zu Beginn nicht auf den Pool-Rechnern installiert.

3.2 Modelsim

Die Handhabung von Modelsim war anfangs ziemlich komplex, da die GUI unserer Auffassung nach recht überladen (und zum Teil altbacken) ist und uns überfordert hat. Der schwierigste und zeitaufwendigste Teil der Aufgabe bestand jedoch in dem Erstellen des Skripts, da wir mit der Sprache Tcl noch nicht vertraut waren. Zudem war ein großes Problem, dass die Kommandozeilenargumente des Skripts beim Aufruf aus Modelsim nicht wie üblich in $argv geschrieben werden, sondern über $1 bis $9 zu erreichen sind.

Aufgabe 4: Xilinx ISE

Aufgabe 5: LED-Test

Aufgabe 6: Lauflicht

Aufgabe 7: Bitmuster