



Ausgabe: 30. Oktober 2017

Abgaben	{	 Theorie	entfällt
		 Praxis	entfällt
		Rücksprache	entfällt

Aufgabe 1: Prolog

Schwerpunkte

- Erste Schritte mit der Simulations-Umgebung
- Simulation elementarer Gatter auf struktureller Ebene

Im Rahmen des ersten Übungstermins sollen anhand eines einfachen Beispiels die notwendigsten Features des HDL¹-Simulators und des Signal-Betrachters erfahren werden. Wer sich genauer für die Mächtigkeit der kompletten Umgebung interessiert, kann seinen Wissensdurst mit den Anleitungen der verwendeten Programme *ModelSIM*[5] und *GTKWave*[1] stillen.

Das Aufgabenblatt soll, ähnlich wie ein Tutorial, Schritt für Schritt am Rechner abgearbeitet werden. Die einzelnen Schritte sind in Aufgaben eingeteilt. Zu Beginn jeder Aufgabe findet sich ein einleitender Text, der die Vorgehensweise erläutert. Anschließend wird Schritt für Schritt beschrieben, was Sie tun sollen. Grundkenntnisse der Rechnerumgebung der Fakultät IV werden als bekannt vorausgesetzt. Bei Bedarf nutzen Sie die Informationen aus dem Leitfaden des Informatik Rechnerbetriebs (z.B. [2]).

Aufgabe 2: Arbeitsumgebung starten

Um die für das Praktikum notwendigen Werkzeuge benutzen zu können, müssen Sie im Fakultätsnetz das Betriebssystem Ubuntu nutzen. Dazu sollten Sie im Begrüßungsbildschirm der Arbeitsplätze in den Rechnerräumen im MAR- und TEL-Gebäude (*IRB Session Box*)

Ubuntu 16

auswählen. Danach geben Sie ihre tubIT-Login-Daten ein.

Aufgabe 3: Die Simulationsumgebung vorbereiten

Zu Simulationszwecken werden Sie während des Praktikums den HDL-Simulator *ModelSIM* und den Betrachter für Signalverläufe *GTKWave* verwenden. Die genauen Optionen und Parameter für die Programme werden durch ein in den Vorgaben enthaltenen *Makefile* vorgegeben, sodass Sie sich nicht mit den genauen Optionen auseinandersetzen müssen. Eine kurze Anleitung für die Benutzung der Umgebung finden Sie in dem *GTKWave-Tutorial*, welches im ISIS-Kurs hochgeladen wurde.

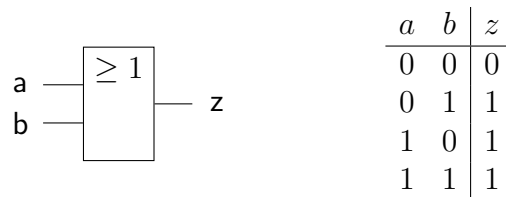
¹HDL: Hardware Description Language (deutsch: Hardwarebeschreibungssprache)

Starten Sie ein *Terminal*, indem Sie über das **Startmenü, Konsole** eintippen und mit Enter bestätigen. Entpacken Sie das Vorgaben-Archiv, welches Sie auf der ISIS-Seite finden und navigieren Sie in das Verzeichnis in welches Sie die Dateien entpackt haben.

Zum Editieren der Dateien aus den Vorgaben wird ein einfacher Text-Editor verwendet, den Sie selber wählen können. Für unerfahrene Nutzer ist ein grafischer Editor mit Syntax-Highlighting, wie zum Beispiel *kate* oder *gedit* zu empfehlen.

Aufgabe 4: Entwurf eines einfachen Gatters

Ihr erstes VHDL-Projekt ist ein einfaches *ODER*-Gatter mit zwei Eingängen:



Das Gatter hat ideales Verzögerungs- und Flankenverhalten, d.h. es treten keinerlei Transportverzögerungen auf und die Flanken sind unendlich steil. Die Beschreibung in VHDL ist sehr kompakt, da sich die Gatterfunktionalität durch die elementare `or`-Funktion aus dem VHDL-Sprachumfang ausdrücken lässt. Diese ist ebenfalls auf die vordefinierte Signalklasse `bit` aus der Bibliothek `std` anwendbar, welche nicht explizit eingebunden werden muss.

1. Öffnen Sie die vorgegebene Datei `or2.vhd` mit einem Texteditor. Die Datei ist im Ordner, in den Sie die Vorgaben entpackt haben, zu finden.
2. Übertragen Sie den folgenden VHDL-Code in die Datei `or2.vhd` mit Hilfe eines Editors und speichern Sie die veränderte Datei.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity or2 is
port( A, B : in  std_logic;
      Z    : out std_logic );
end entity or2;

architecture simple of or2 is
begin
    Z <= A or B;
end architecture simple;
```

Aufgabe 5: Die Testbench für das *ODER*-Gatter

Um das eben entworfene *ODER*-Gatter zu testen, müssen Sie eine Testbench schreiben. Dabei sollen die Signale `a` und `b` wie folgt stimuliert werden.



Öffnen Sie die Datei `or2_tb.vhd` aus den Vorgaben und übertragen Sie den folgenden Quellcode.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity or2_tb is — Schnittstelle nicht notwendig
end or2_tb;

architecture behaviour of or2_tb is
    signal X, Y, Z : std_logic; — Signale zur Verdrahtung
begin
    dut: entity work.or2(simple) — Gatter-Instanz erzeugen
    port map(A => X, — und explizit verdrahten
             B => Y,
             Z => Z);

    X <= '1', — Stimmulationssequenz Eingang X
        '0' after 15 ns,
        '1' after 35 ns,
        '0' after 45 ns,
        '1' after 65 ns;
    Y <= '1', — Stimmulationssequenz Eingang Y
        '0' after 5 ns,
        '1' after 20 ns,
        '0' after 35 ns,
        '1' after 60 ns,
        '0' after 80 ns;
end behaviour;
```

Aufgabe 6: Simulation des *ODER*-Gatters

Nun soll eine Simulation des Gatters für einen sinnvollen Zeitraum durchgeführt werden. Hierzu wird die Testbench `or2_tb` in den Simulator geladen und der zeitliche Verlauf von Ein- und Ausgangssignalen graphisch validiert.

- Das Übersetzen der Quelldateien und die Simulation kann durch den Befehl `make clean all` automatisiert durchgeführt werden. Navigieren Sie dazu in der Konsole in den Ordner in dem sich die Dateien `or2.vhd`, `or2_tb.vhd` und `Makefile` befinden. Geben Sie dann den Befehl ein und bestätigen Sie ihre Eingabe mit der Enter-Taste.
- Überzeugen Sie sich, dass keine Fehler bei der Simulation aufgetreten sind, indem Sie die Ausgaben des Programmes analysieren.

- Wenn die Simulation erfolgreich durchgeführt wurde können Sie mit dem Kommando `make view_wave` den Betrachter *GTKWave* starten. Ein Anleitung zum Bedienen dieses Betrachters ist auf ISIS verfügbar (GTKWave-Tutorial). Konsolidieren Sie dieses Tutorial und überzeugen Sie sich von der korrekten Funktionsweise ihres Gatters durch das Betrachten der Signalverläufe.
-

Literatur

- [1] BSI. *GTKWave 3.3 Wave Analyzer User's Guide*, 2014.
- [2] Informatik Rechnerbetrieb der Fakultät IV (TU Berlin). *Leitfaden zum Arbeiten im Fakultätsnetz*.
- [3] Mentor Graphics Corporation. *ModelSim SE Reference Manual*, 6.4a edition.
- [4] Mentor Graphics Corporation. *ModelSim SE Tutorial*, 6.4a edition.
- [5] Mentor Graphics Corporation. *ModelSim SE User's Manual*, 6.4a edition.