



Ausgabe: 26. Oktober 2015

Abgaben	{		Theorie	entfällt
			Praxis	entfällt
			Rücksprache	entfällt

Aufgabe 1: Prolog

Schwerpunkte

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

Im Rahmen des ersten Übungstermins sollen anhand eines einfachen Beispiels die notwendigsten Features des HDL¹-Simulators erfahren werden. Wer sich genauer für die Mächtigkeit der kompletten Umgebung interessiert, kann seinen Wissensdurst mit dem über 1000 Seiten starken *User's Manual* [4] 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. [1]) oder die Folien aus der Rechnereinführung (z.B. [5]).

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 14

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

Aufgabe 3: Arbeitsumgebung einrichten

Die notwendigen Werkzeuge wurden im AFS-Bereich des Fachgebietes installiert. Um sie verwenden zu können, müssen einige Umgebungsvariablen um zusätzliche Verzeichnisse erweitert werden. Das Setzen der Pfade erfolgt durch ein Skript und den Aufruf einer Hilfsfunktion, der bei jedem Start eines Terminals ausgeführt werden sollte. Folgende Vorgehensweise wird empfohlen:

¹HDL: Hardware Description Language (deutsch: Hardwarebeschreibungssprache)

1. Einfügen der folgenden Zeile in die eigene `.bashrc`:²

```
source /afs/tu-berlin.de/units/Fak_IV/aes/scripts/techgi2pr.sh
```

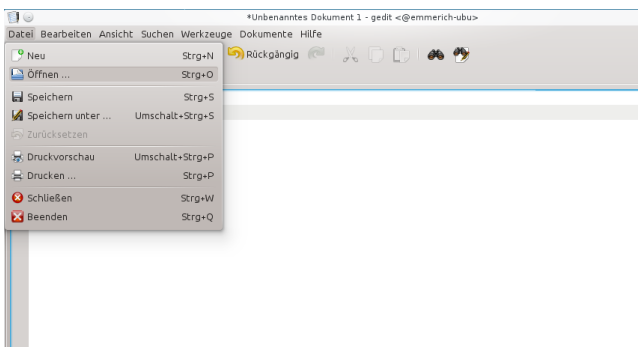
2. Vor dem Starten einer „TechGI2-TI“-Anwendung auf der Kommandozeile des gewünschten Terminals das Kommando

```
techgi2pr
```

eingeben. Hiermit werden alle nötigen Einstellungen in den Umgebungsvariablen der aktuellen Shell vorgenommen.

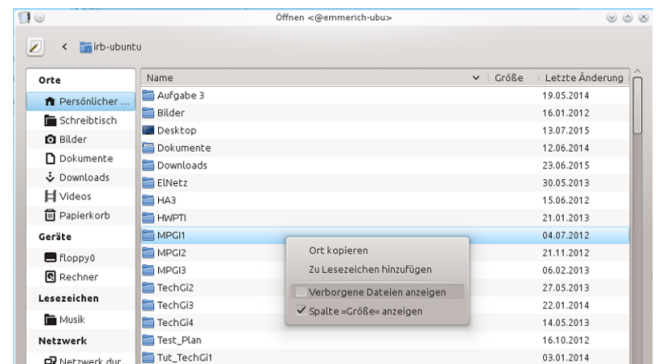
Es folgt eine einfache Schritt-für-Schritt-Anleitung, wie Sie mit Hilfe der grafischen Oberfläche in die `.bashrc` eine Zeile einfügen können:³

Öffnen Sie hierzu den Editor *gedit*, indem Sie im **Startmenü** (Symbol in der unteren linken Ecke) einfach **gedit** eingeben und mit Enter bestätigen.



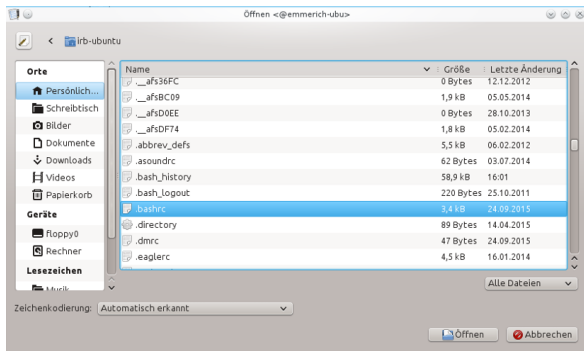
Öffnen Sie den Dialog zum Dateiöffnen.

Lassen Sie sich verborgene Dateien anzeigen, indem Sie das Kontextmenü innerhalb des Datei-Fensters öffnen.



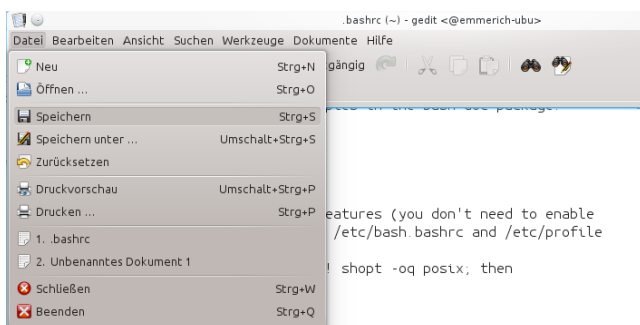
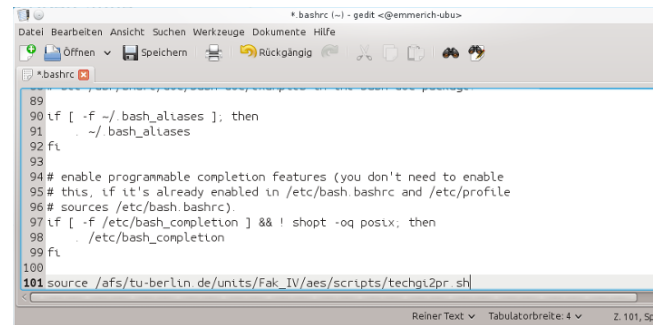
²Dieser Schritt ist nur ein einziges Mal auszuführen. Wer weiß was er tut, kann die im folgenden beschriebenen Schritte gern auch durch eine adäquate Kommandozeile ersetzen.

³Eine deutlich kürzere Alternative im Terminalfenster finden Sie im Anschluss an diese Schritt-für-Schritt-Anleitung.



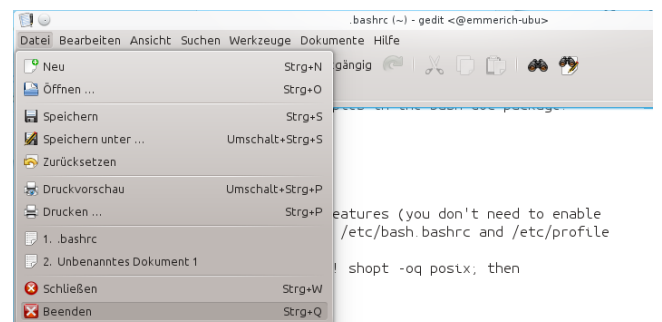
Öffnen Sie die Datei `.bashrc`.

Fügen Sie *am Ende* der Datei die oben angegebene Zeile ein.



Speichern Sie die Datei.

Beenden Sie *gedit*.



Alternativer Weg. Statt unter Verwendung des grafischen *gedit* können Sie alternativ auch in einem Terminalfenster (vgl. Aufgabe 3)

```
echo "source /afs/tu-berlin.de/units/Fak_IV/aes/scripts/techgi2pr.sh" >> ~/.bashrc
```

ausführen, mit `exit` das Terminal schließen und anschließend mit der nächsten Aufgabe fortfahren.

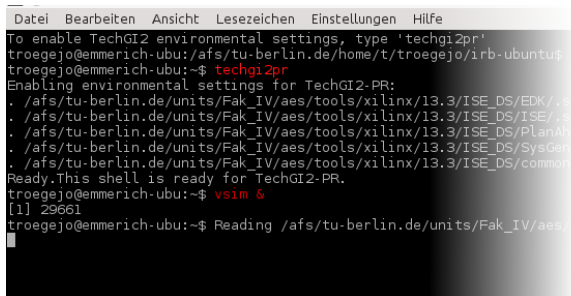
Bitte beachten Sie, dass die `.bashrc` bei reinen SSH-Verbindungen u.U. nicht ausgeführt wird. Um

dies zu umgehen, fügen Sie z.B. die `source...`-Zeile ebenfalls in die `.bash_login` ein.

Aufgabe 4: Den Simulator *ModelSIM* starten

Zu Simulationszwecken werden Sie während des Praktikums den HDL-Simulator *ModelSIM* verwenden.

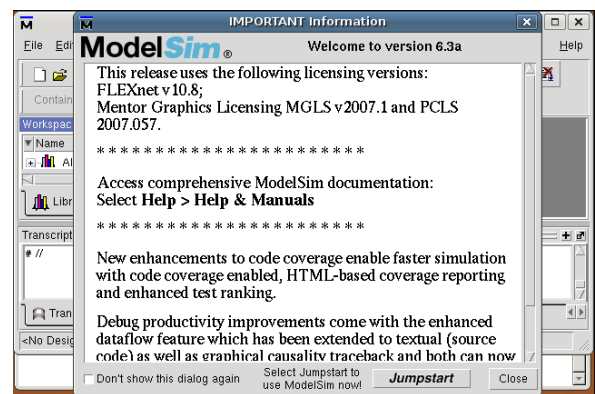
Starten Sie ein *Terminal*, indem Sie über das **Startmenü**, **Konsole** eintippen und mit Enter bestätigen.



```
Datei Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe
To enable TechGI2 environmental settings, type 'techgi2pr'
troegejo@emmerich-ubu:/afs/tu-berlin.de/home/t/troegejo/irb-ubuntu$ techgi2pr
Enabling environmental settings for TechGI2-PR:
. /afs/tu-berlin.de/units/Fak_IV/aes/tools/xilinx/13.3/ISE_DS/EDK/.
. /afs/tu-berlin.de/units/Fak_IV/aes/tools/xilinx/13.3/ISE_DS/ISE/.
. /afs/tu-berlin.de/units/Fak_IV/aes/tools/xilinx/13.3/ISE_DS/PlanAl
. /afs/tu-berlin.de/units/Fak_IV/aes/tools/xilinx/13.3/ISE_DS/SysGen
. /afs/tu-berlin.de/units/Fak_IV/aes/tools/xilinx/13.3/ISE_DS/commo
Ready. This shell is ready for TechGI2-PR.
troegejo@emmerich-ubu:~$ vsim &
[1] 29661
troegejo@emmerich-ubu:~$ Reading /afs/tu-berlin.de/units/Fak_IV/aes/
```

Starten Sie *ModelSIM*, indem Sie im Terminal `techgi2pr` eingeben und die Eingabe mit Enter-Taste bestätigen. Anschließend geben Sie das Kommando `vsim &` ein und bestätigen Sie dieses mit der Enter-Taste.

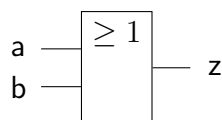
Schließen Sie den Willkommens-Dialog von *ModelSIM*.



ModelSIM besteht aus mehreren Fenstern. Sie können den Simulator mit der Maus bedienen oder ihn durch Kommandos steuern. Kommandos geben Sie im Fenster *Transcript* (großes Fenster im unteren Bereich des Simulators) ein. Im Folgenden wird sowohl die Bedienung per Maus als auch mit Hilfe von Kommandos dargestellt.

Aufgabe 5: Entwurf eines einfachen Gatters

Ihr erstes VHDL-Projekt in *ModelSIM* ist ein einfacher *ODER*-Gatter mit zwei Eingängen:



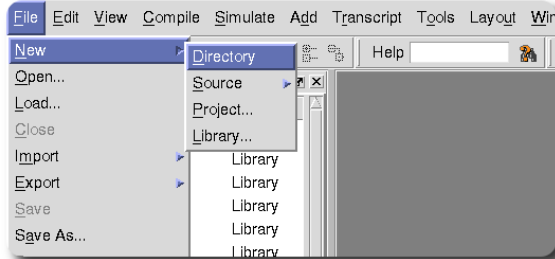
a	b	z
0	0	0
0	1	1
1	0	1
1	1	1

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. Legen Sie ein Projekt in einem entsprechenden Ordner an.

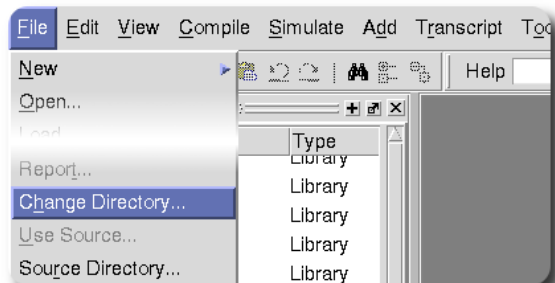
- Erstellen Sie einen Ordner für Ihr Projekt, beispielsweise `rorgpr/blatt0/aufgabe4`, über **File > New > Directory**



Transcript

```
mkdir -p rorgpr/blatt0/aufgabe4
```

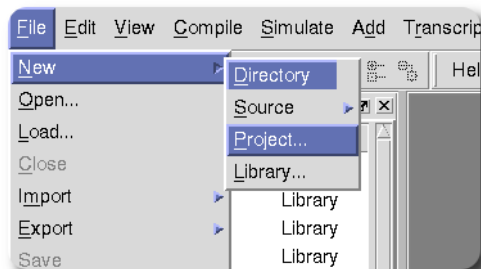
- Wechseln Sie in den eben angelegten Ordner mittels **File > Change Directory...**



Transcript

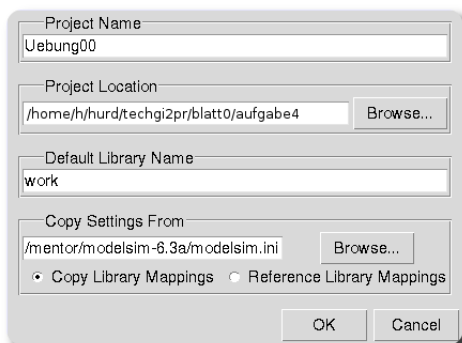
```
cd rorgpr/blatt0/aufgabe4
```

- Erstellen Sie ein neues Projekt, mit dem Namen `Uebung00` (oder einem anderen Namen) im aktuellen Ordner (.) über **File > New > Project...**. Sollte sich der Dialog „Add item to the Project“ öffnen, schließen Sie diesen.



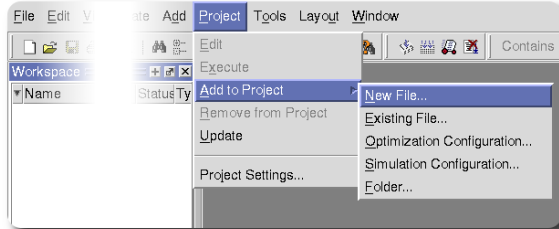
Transcript

```
project new . Uebung00
```



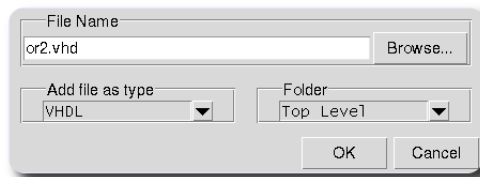
2. Legen Sie eine VHDL-Beschreibung für das *ODER*-Gatter an.

- Legen Sie eine neue VHDL-Datei namens `or2.vhd` für das *ODER*-Gatter an über **Project > Add to Project > New File**



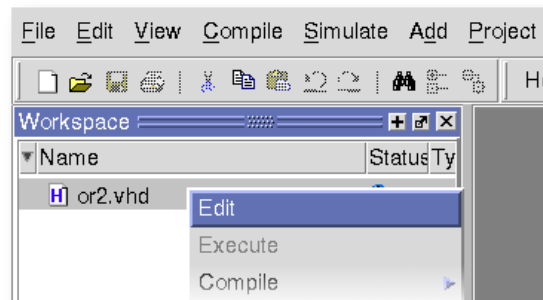
Transcript

```
project addfile or2.vhd
```



(Wenn die Datei noch nicht existiert: `touch or2.vhd` im *Transcript*-Fenster eintippen)

- Öffnen Sie die Datei zum Editieren (Kontextmenü von `or2.vhd` > **Edit**)



Transcript

```
edit or2.vhd
```

3. Übertragen Sie den folgenden VHDL-Code in die Datei `or2.vhd` mit Hilfe des eben geöffneten Editors und speichern Sie die veränderte Datei.

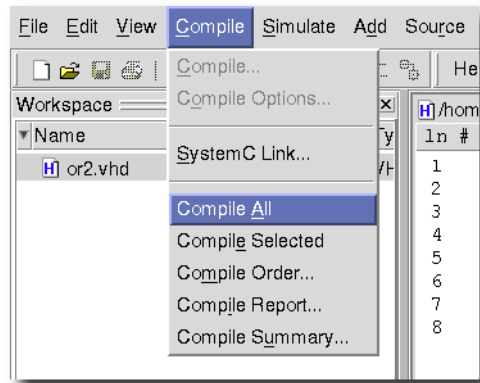
```
library IEEE;
use IEEE.std_logic_1164.all;

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;
```

4. Bei *ModelSIM* handelt es sich um einen übersetzenden Simulator, d.h. die Schaltung liegt in einer Quellsprache vor (z.B. VHDL) und wird mittels eines Compilers in einen kompakten Simulator-Code übersetzt. Dabei werden Design-Entities vom Compiler in einer Bibliothek abgelegt. Der eigentliche Simulator arbeitet schließlich auf den Design-Entities in den Bibliotheken.

- Übersetzen Sie Ihr *ODER*-Gatter

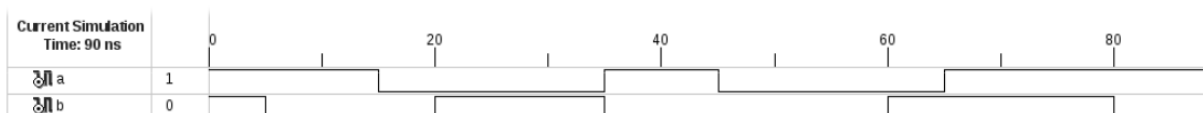


Transcript

vcom or2.vhd

Aufgabe 6: 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.



Legen Sie eine Datei namens `or2_tb.vhd` an und übertragen Sie den folgenden Quellcode. Gehen Sie dabei genau so vor, wie Sie es in der letzten Aufgaben getan haben. Diesmal wird jedoch kein neues Projekt angelegt, sondern das bereits erstellte erweitert (→ siehe Aufgabe 4 ab Schritt 2).

Hinweis: Sollte das Menü "Project" nicht zur Verfügung stehen, so klicken Sie einmal in das Fenster mit dem Namen "Workspace".

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

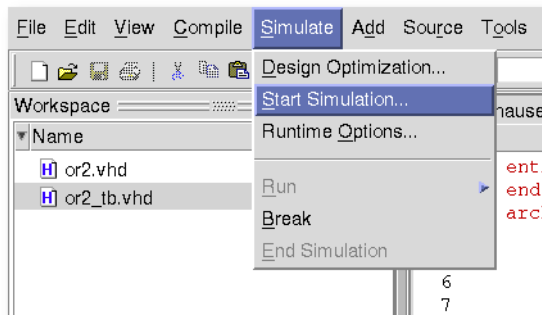
end behaviour;

```

Aufgabe 7: Simulation des *ODER*-Gatters

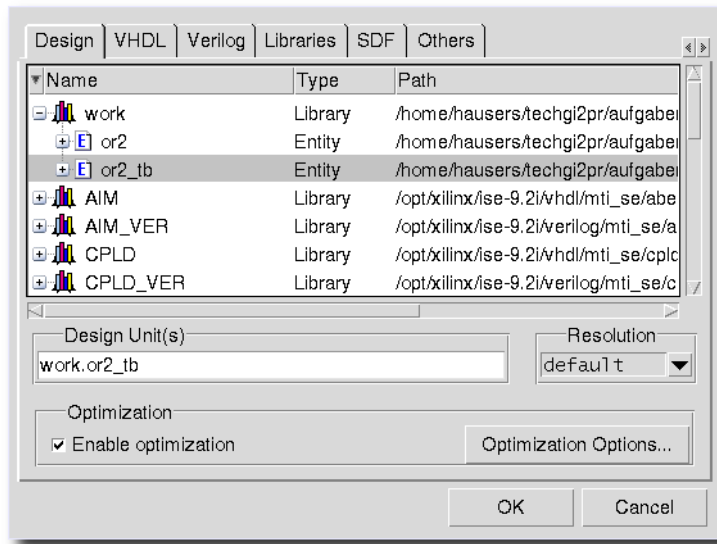
Nun soll eine Simulation des Gatters für einen sinnvollen Zeitraum durchgeführt werden. Hierzu wird die Testbench `or2_tb` in den Simulator geladen und der zeitliche Verlauf von Ein- und Ausgangssignalen graphisch validiert.

- Simulation für `or2_tb` starten

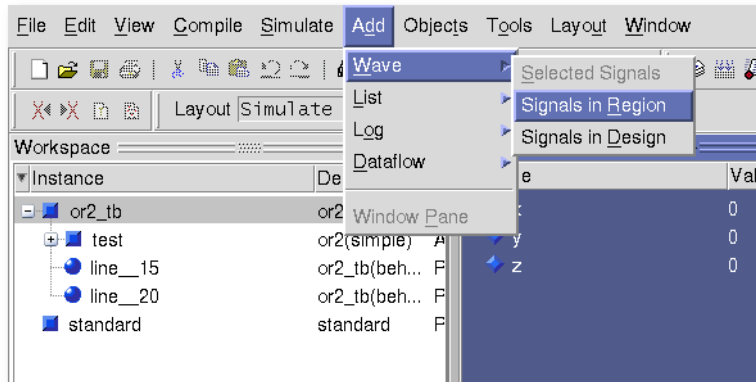


Transcript

```
vsim or2_tb
```

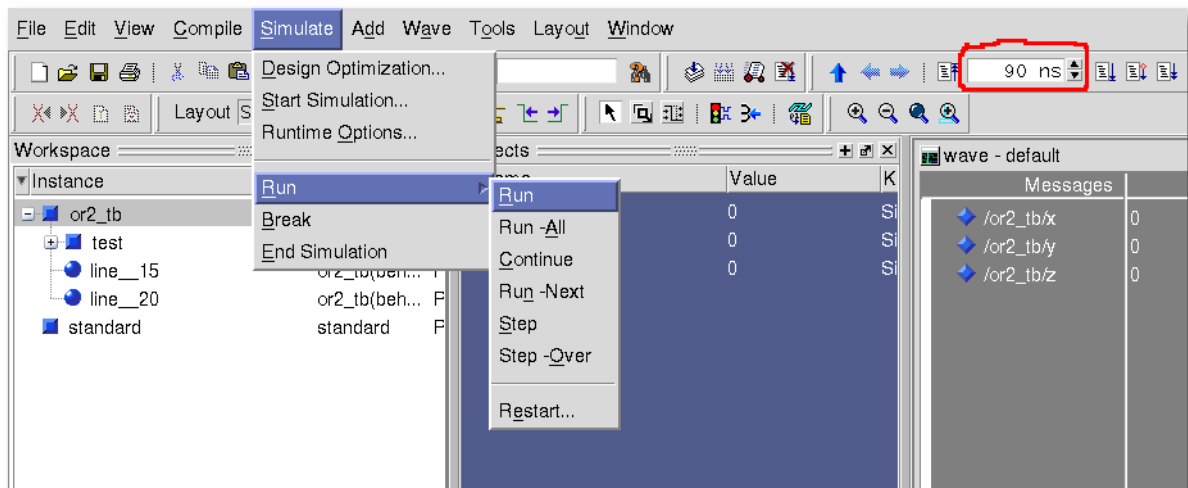
- Alle Signale zur Waveform hinzufügen



Transcript

```
add wave sim:/or2_tb/*
```

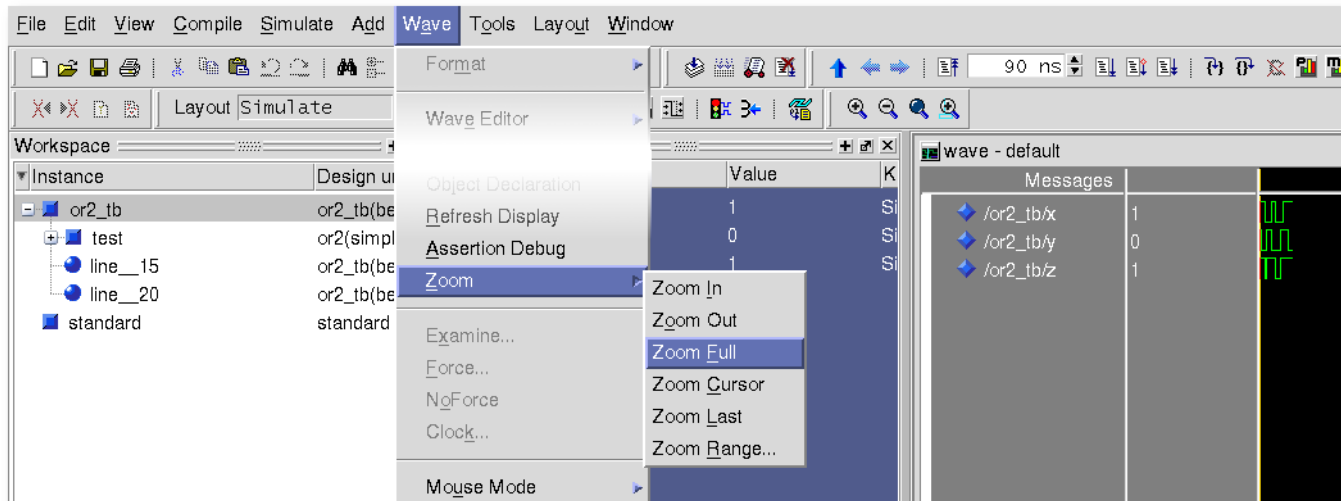
- Simulation für 90 ns laufen lassen



Transcript

```
run 90 ns
```

- Sichtbaren Ausschnitt verändern



Transcript

wave zoomfull

Überzeugen Sie sich anhand des grafisch dargestellten Signalverlaufes von der korrekten Funktionsweise des *ODER*-Gatters.

Literatur

- [1] Informatik Rechnerbetrieb der Fakultät IV (TU Berlin). *Leitfaden zum Arbeiten im Fakultätsnetz*.
- [2] Mentor Graphics Corporation. *ModelSim SE Reference Manual*, 6.4a edition.
- [3] Mentor Graphics Corporation. *ModelSim SE Tutorial*, 6.4a edition.
- [4] Mentor Graphics Corporation. *ModelSim SE User's Manual*, 6.4a edition.
- [5] Nicolas Schier, Daniel Feller, and Stefan Hauser. *Einführung in das IRB-Netz (Solaris)*. unter ISIS abrufbar.