Modelsim und VHDL Einführung

VHDL (Very High Speed Integrated Circuit Hardware Description Language) ist eine Sprache zur Beschreibung von digitalen Schaltungen. Modelsim ist eine Simulationssoftware der Firma Mentor Graphics mit der das Verhalten von VHDL Beschreibungen simuliert werden kann.

Diese Einführung beginnt mit zwei einfachen VHDL Dateien, die eine einfache Schaltung beschreibt.

Vorbereitung

Kopieren Sie die VHDL Dateien in ein Verzeichnes mit dem Namen „first“.

Die beiden VHDL Dateien sind:

first.vhd - Die Beschreibung der Schaltung

first_tb.vhd - Die Beschreibung einer Testumgebung (Testbench)

Aufbau der VHDL Dateien - Entity und Architecture

Die Datei „first.vhd“:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

-- Simple module that connects the SW switches to the LEDR lights
ENTITY first IS
PORT ( SW   : IN      STD_LOGIC_VECTOR(9 DOWNTO 0);
       LEDR : OUT     STD_LOGIC_VECTOR(9 DOWNTO 0));  -- red LEDs
END first;

ARCHITECTURE Structure OF first IS
BEGIN
  LEDR <= SW;
END Structure;
```

Die Datei enthält eine Entity- und eine Architecturebeschreibung. Entity und Architecture beschreiben zusammen ein Schaltungsmodul. Die Entity beschreibt die Schnittstellen des Moduls, während die Architecture den Inhalt des Moduls beschreibt. Aus der Entitybeschreibung von „first.vhd“ kann man ersehen

a) Das Modul hat den Namen „first“

b) Es gibt einen Eingangsport mit dem Namen „SW“, der 10 Bit breit und vom Typ std_logic_vector ist.

c) Es gibt einen Ausgangsport mit dem Namen „LEDR“, der auch 10 Bit breit ist und ebenfalls vom Typ std_logic_vector ist.

Aus der Architecture kann man entnehmen:

a) Diese Architektur hat den Namen „structure“ und gehört zum Modul first

b) Dem Ausgangsport „LEDR“ wird der Wert des Eingangsports „SW“ zugewiesen. Der Operator „<=“, weißt dem Signal auf der linken Seite vom Operator, den Wert des Signals auf der rechten Seite zu.

Diese Schaltung verbindet also den Eingang SW einfach mit dem Ausgang LEDR.

Typen - std_logic und std_logic_vector

Jedes Signal hat einen Typen. Der Typ eines Signals gibt an, welche Werte das Signal annehmen kann. In dem Beispiel first haben die Eingangs- und Ausgangsports den Typen „std_logic_vector“. „std_logic_vector“ ist eine Zusammenfassung (ein Bus) von mehreren Signalen vom Typ std_logic.

Der Typ std_logic kann Werte annehmen, die den Zuständen von Leitungen in einer digitalen Schaltung entsprechen. Die wichtigsten sind „0“ und „1“. Es gibt aber noch weitere Zustände, wie beispielsweise „U“, das für „undefined“ steht, wenn ein Signal keinen definierten Zustand hat. Dieser Zustand tritt beispielsweise bei einem Flipflop auf, das noch nicht zurückgesetzt oder getaktet wurde.

Die Testbench

Die andere VHDL Datei ist „first_tb.vhd“:

```
library ieee;
use ieee.std_logic_1164.all;

-- Testbench for the first switch/led module
-- The switches are switched...

entity first_tb is
end;

architecture beh of first_tb is

-- This is the component declaration
  component first
  port (
    SW:         in std_logic_vector(9 downto 0);
    LEDR:       out std_logic_vector(9 downto 0)
    );
  end component;

-- Signal declaration for the switches and the leds
  signal switch, ledr : std_logic_vector(9 downto 0) := "0000000000";

begin

-- Here the device under test is instantiated
  first_i0 : first
    port map (
      SW                => switch,
      LEDR              => ledr);

-- This is the process where the switch is switched. Note that the
process
-- always starts from the beginning. This process is not synthesizable
because
-- of the wait statement.

  schalter : process
  begin
    wait for 1 us;
    switch <= "0000000001";
    wait for 3 us;
    switch <= "1000000000";
  end process schalter;

end; -- architecture
```

Auch die Datei „first_tb.vhd“ hat eine Entity und eine Architecture. Aus der Entitybeschreibung kann man entnehmen

a) Das Modul heißt: first_tb

b) Das Modul hat keine Eingänge und keine Ausgänge.

Aus der Architecturebeschreibung kann man entnehmen

a) Es gibt eine Komponente „first“ mit den Ein- und Ausgangssignalen, die aus der Entitybeschreibung von „first“ bekannt sind.

b) Es gibt zwei Signale „switch“ und „ledr“ mit 10 Bit Breite vom Typen std_logic_vector. Die Signale werden auf den Wert „0000000000“ initialisiert.

c) Es wird eine Instanz „first_i0“ des Moduls „first“ instantiiert. Der Eingang „SW“ des Moduls wird an das Signal „switch“ angeschlossen. Der Ausgang „LEDR“ wird an das Signal „ledr“ angeschlossen.

d) Es gibt einen Prozess „schalter“ und in diesem Prozess wird

1. Für den Zeitraum von 1 Mikrosekunde gewartet
2. Dann dem Signal „switch“ der Wert „0000000001“ zugewiesen
3. Für 3 Mikrosekunden gewartet
4. Dem Signal „switch“ der Wert „1000000000“ zugewiesen.

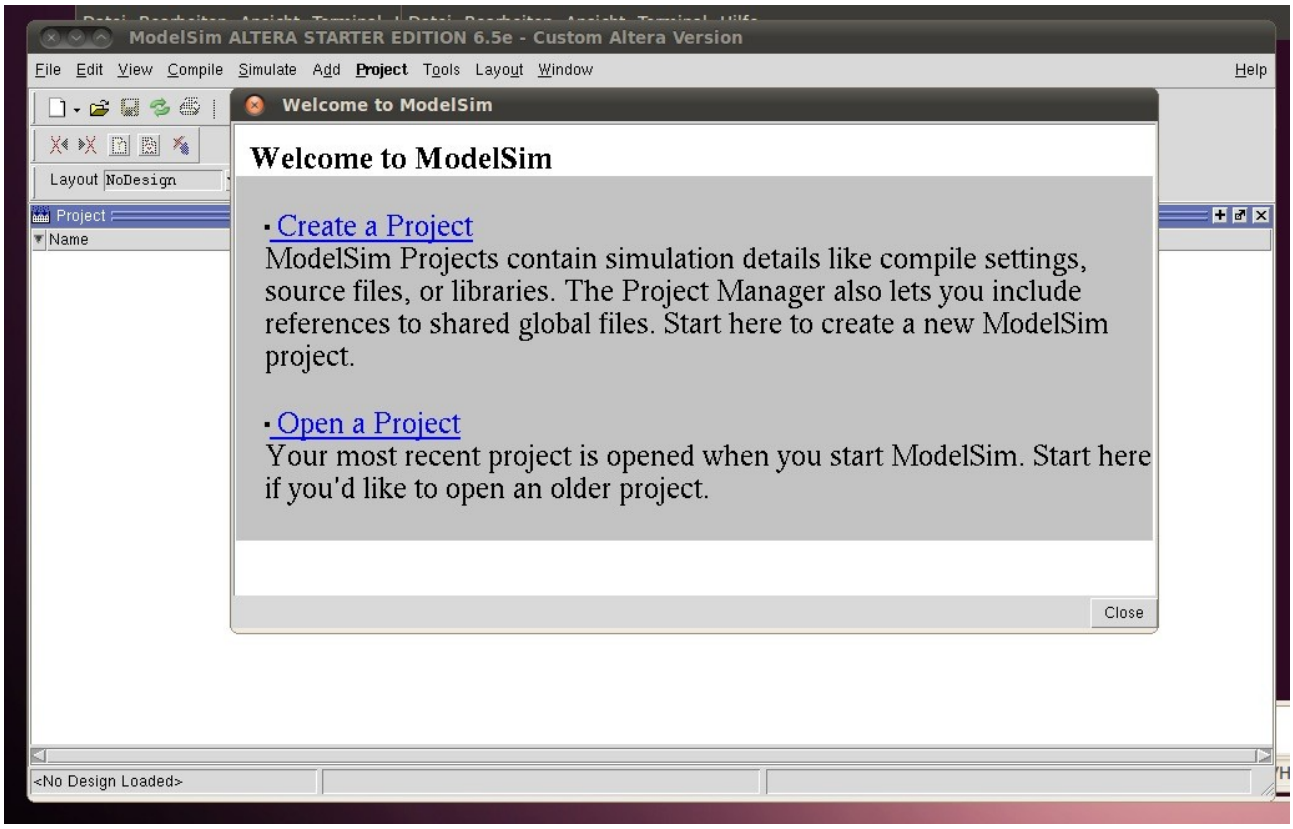
Diese Testbench „first_tb“ enthält also das Modul „first“ und einen Prozess, in dem dem Signal switch nacheinander verschiedene Werte zugewiesen werden. Dieses Signal ist an das Eingangssignal „SW“ des Moduls angeschlossen.

Modelsim

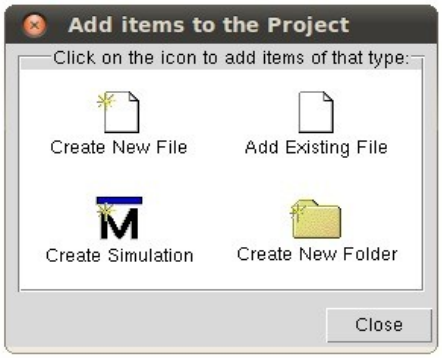
Starten Sie Modelsim.

Linux: Wechseln Sie in das Verzeichnis first und starten Sie mit „vsim“

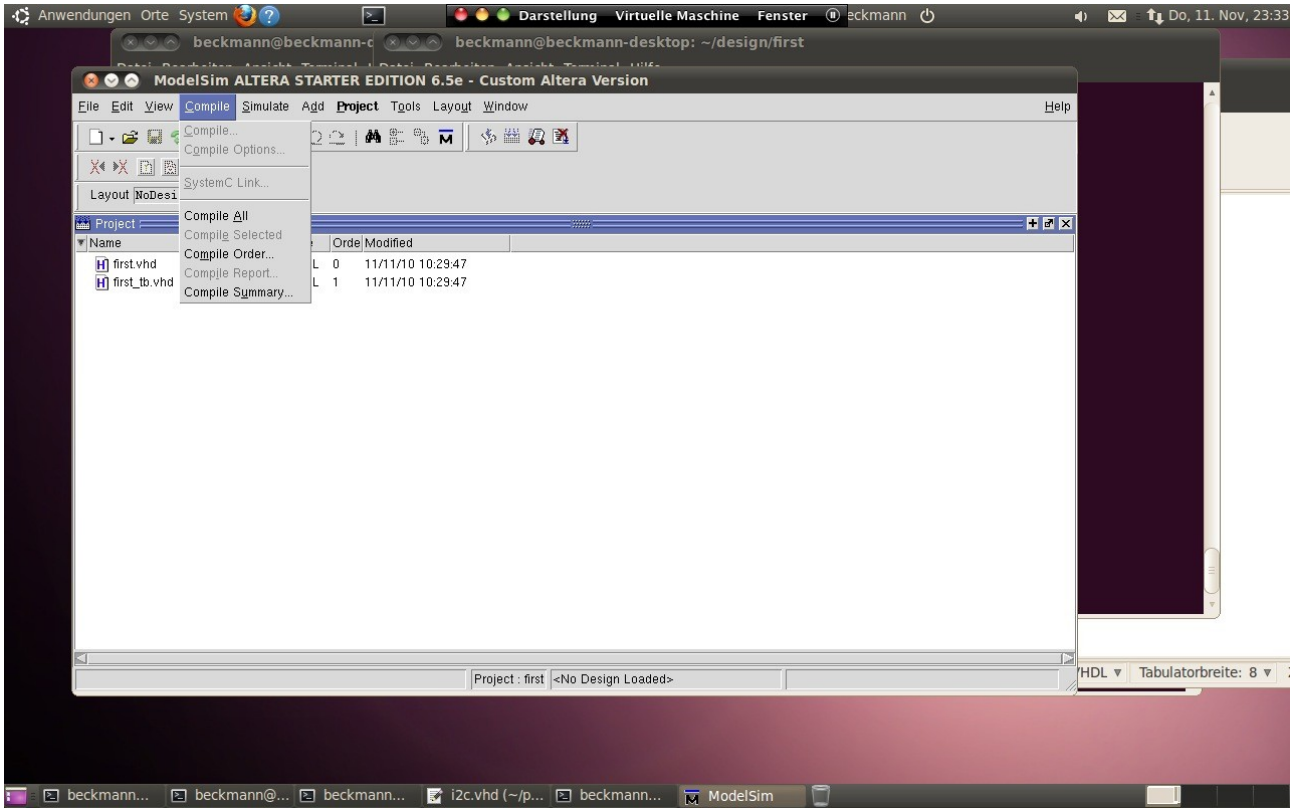
Sie sehen:



Wählen Sie: „Create a Project“. Geben Sie dem Projekt den Namen „first“.



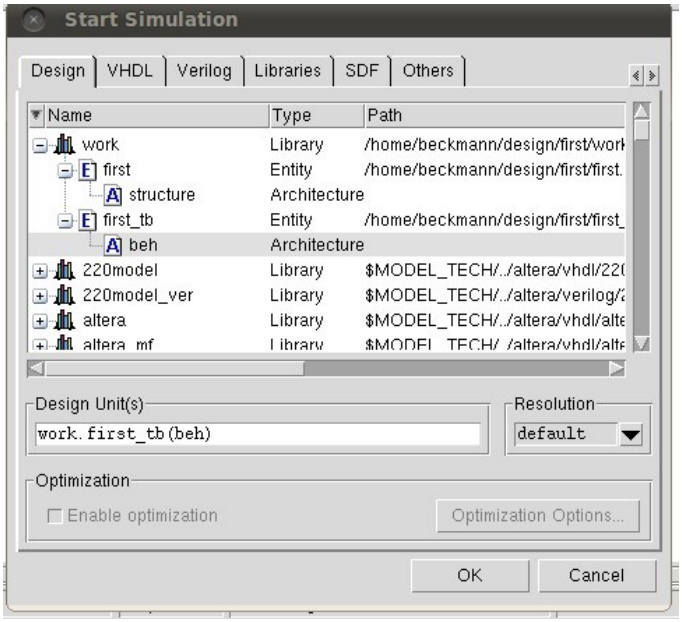
Wählen Sie „Add Existing File“ und fügen Sie „first.vhd“ und „first_tb.vhd“ zu dem Projekt hinzu.



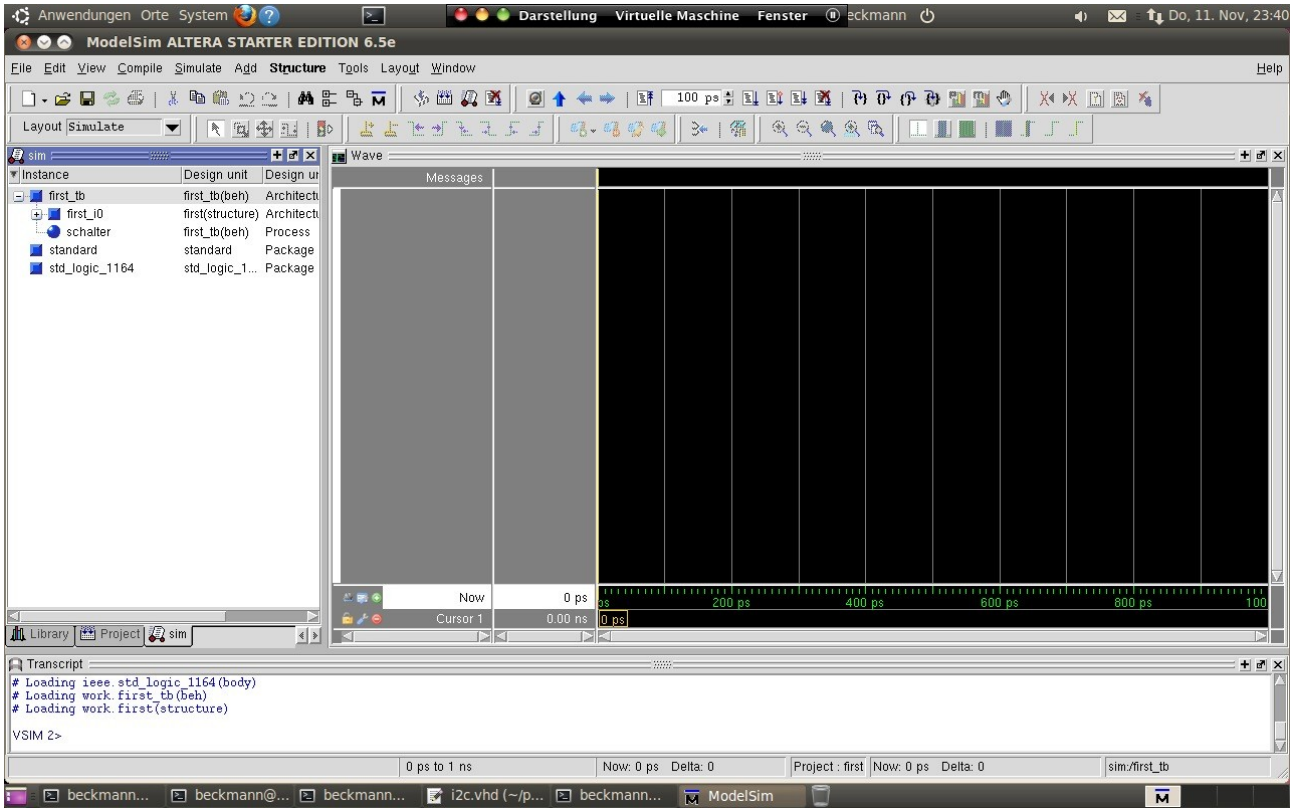
Wählen Sie dann „Compile->Compile All“.

Mit diesem Schritt kompilieren Sie die VHDL Dateien. Dieser Schritt ist notwendig um anschliessend die Simulation durchführen zu können. Die Dateien werden in eine Bibliothek mit dem Namen „work“ kompiliert.

Wählen Sie danach „Simulate->Start Simulation“. Es erscheint eine Übersicht der verschiedenen Bibliotheken.



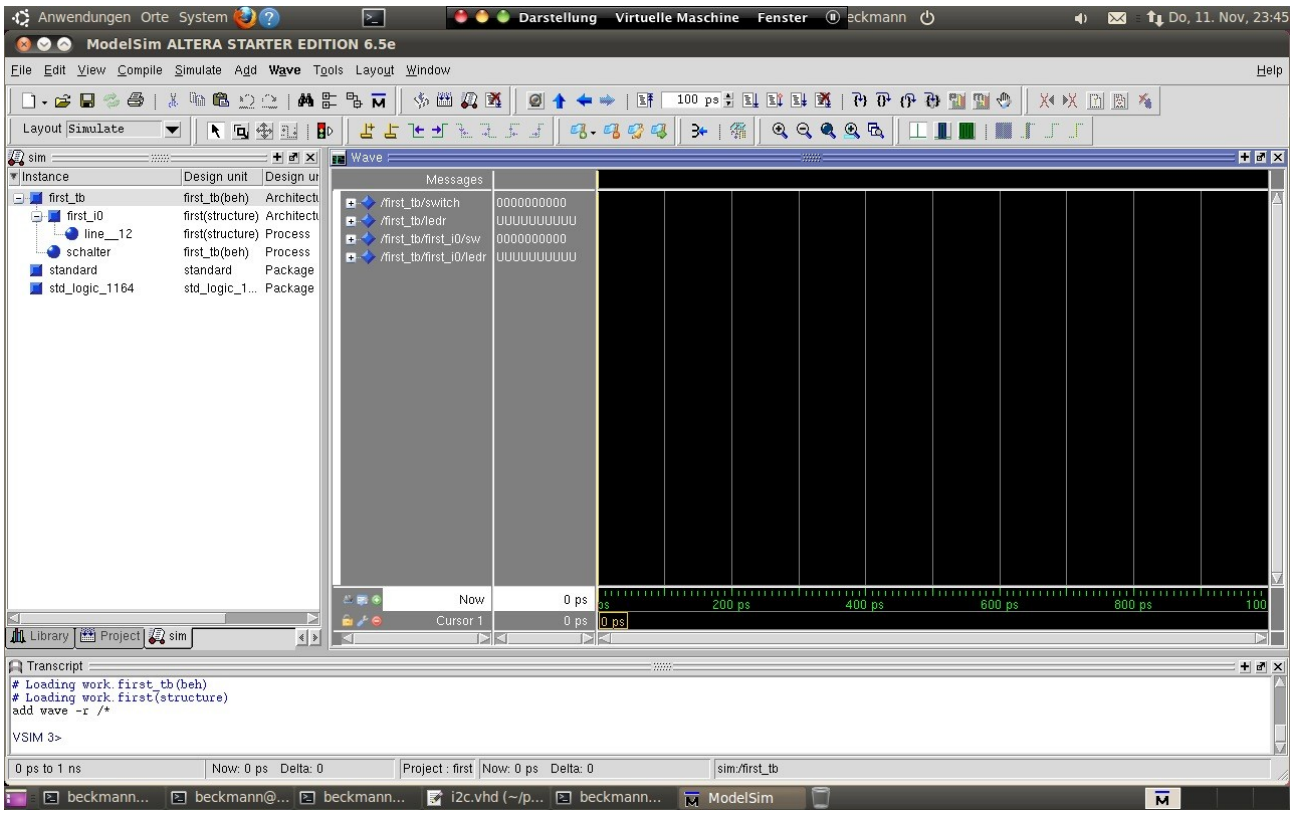
Wählen Sie die Architecture „first_tb(beh)“ aus. Es erscheint eine Übersicht der Schaltungsstruktur und ein Waveformdisplay.

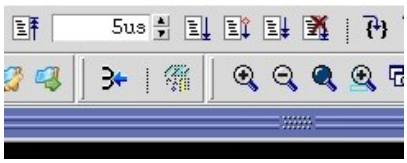


Gehen Sie im Fenster „sim“ auf first_tb und wählen Sie mit der rechten Maustaste

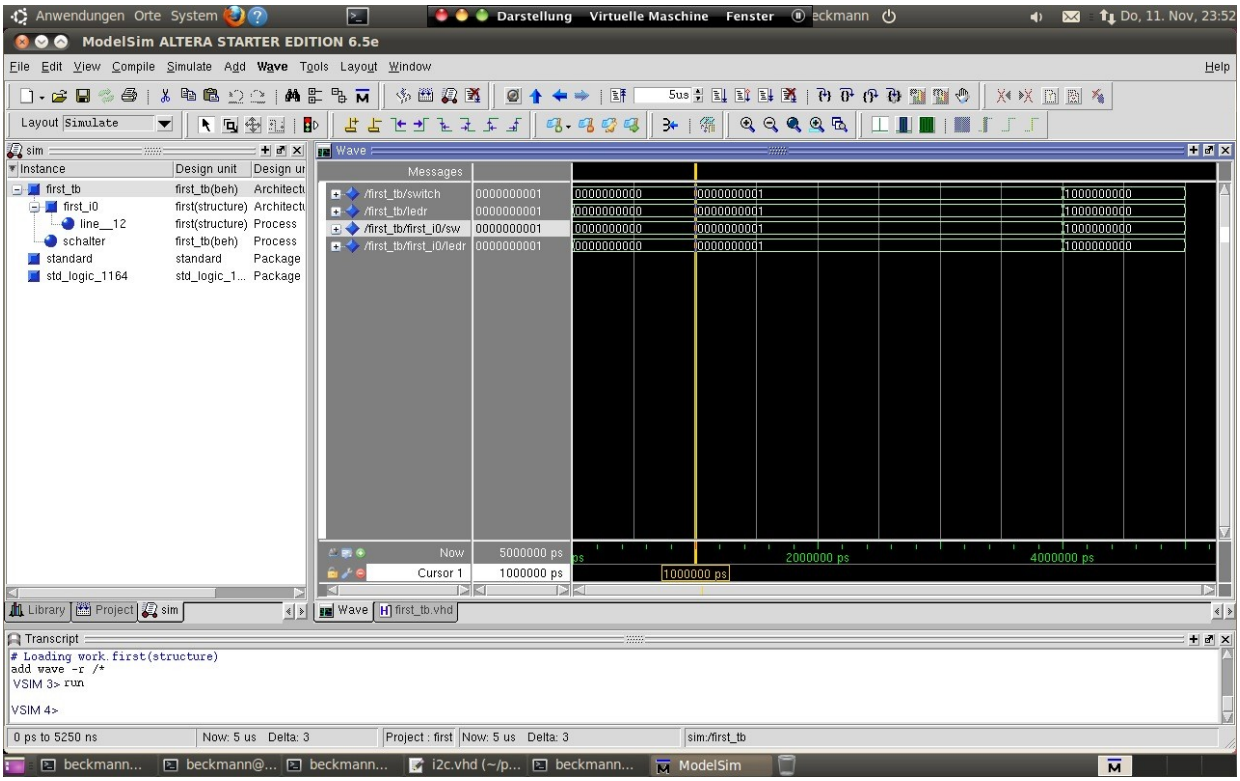
„Add->ToWave->All Items in Design“

Es erscheinen im Waveformdisplay die Signale „switch“ und „ledr“ sowie SW und LEDR. Der Wert von LEDR und ledr ist „U“, also undefined.





Nun kann man die Simulationszeit (hier 5 Mikrosekunden) einstellen und den Knopf „Run“ drücken. Danach kann man im Waveformdisplay zoomen und sich das Ergebnis der Simulation anschauen.



Nach einer Mikrosekunde ändert sich der Wert von SW und LEDR von „0000000000“ auf „0000000001“ und nach weiteren 3 Mikrosekunden auf „1000000000“.

Mit „Simulate->End Simulation“ kann man den Simulator verlassen.

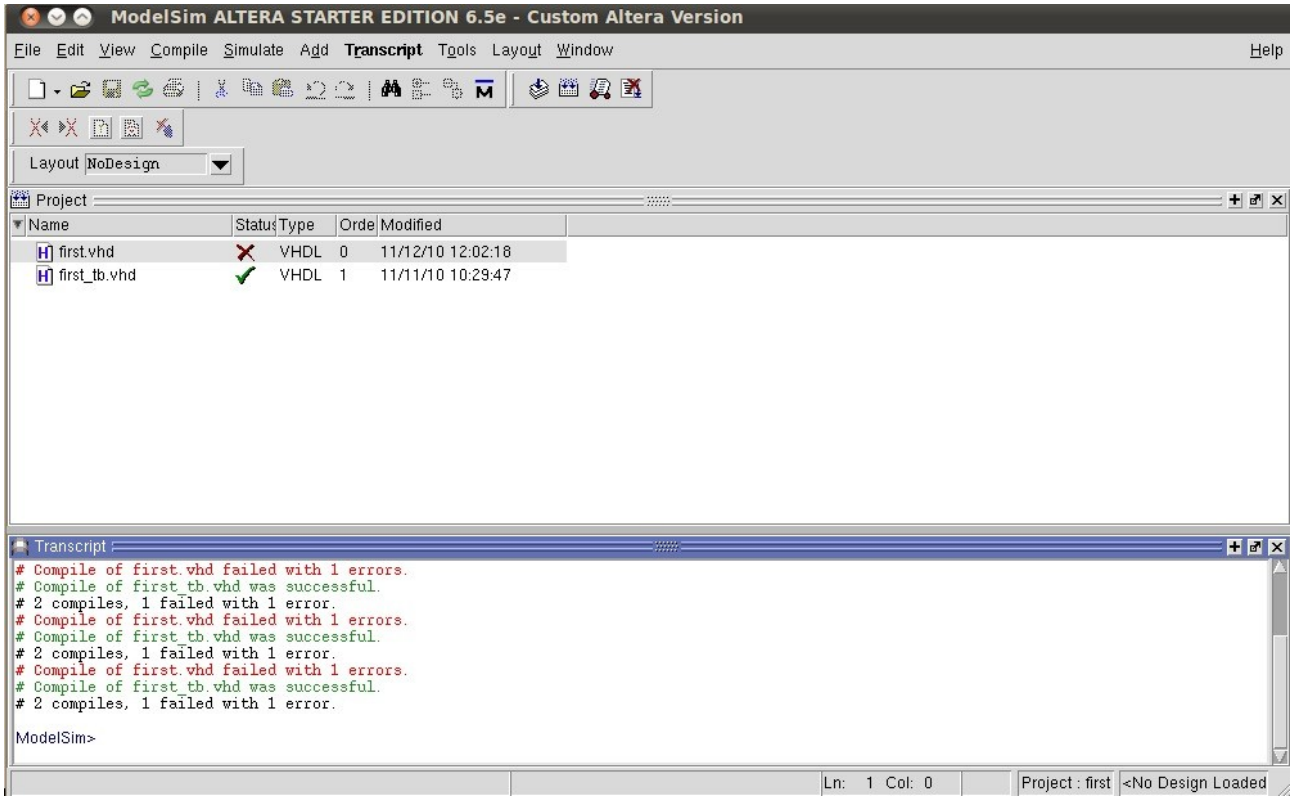
Debugging

In diesem Beispiel kommt es zu keinen Fehlern beim kompilieren des VHDL Codes. Wenn es doch zu Fehlern kommt, dann kann man im Fenster „Transcript“ nachsehen, warum es zu Problemen gekommen ist.

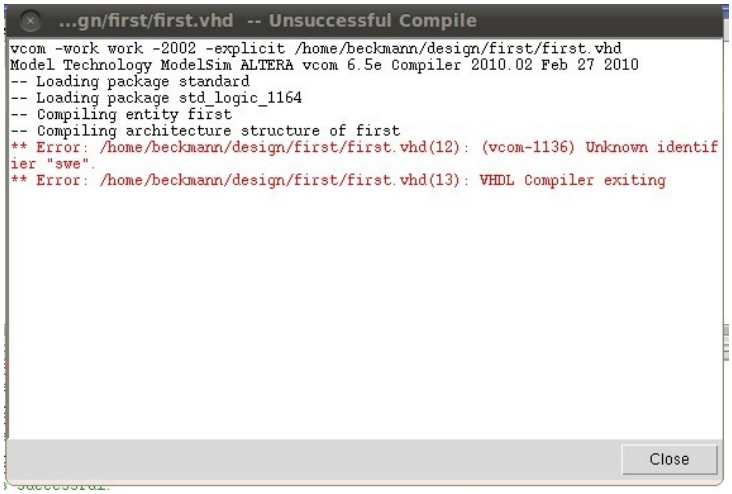
Öffnen Sie das Fenster „Project“ mit „View->Project“ und Doppelklicken Sie auf „first.vhd“. Es erscheint ein Fenster mit dem Sourcecode. Bauen Sie einen Fehler in den Code.

Starten Sie das Kompilieren mit „Compile -> Compile All“.

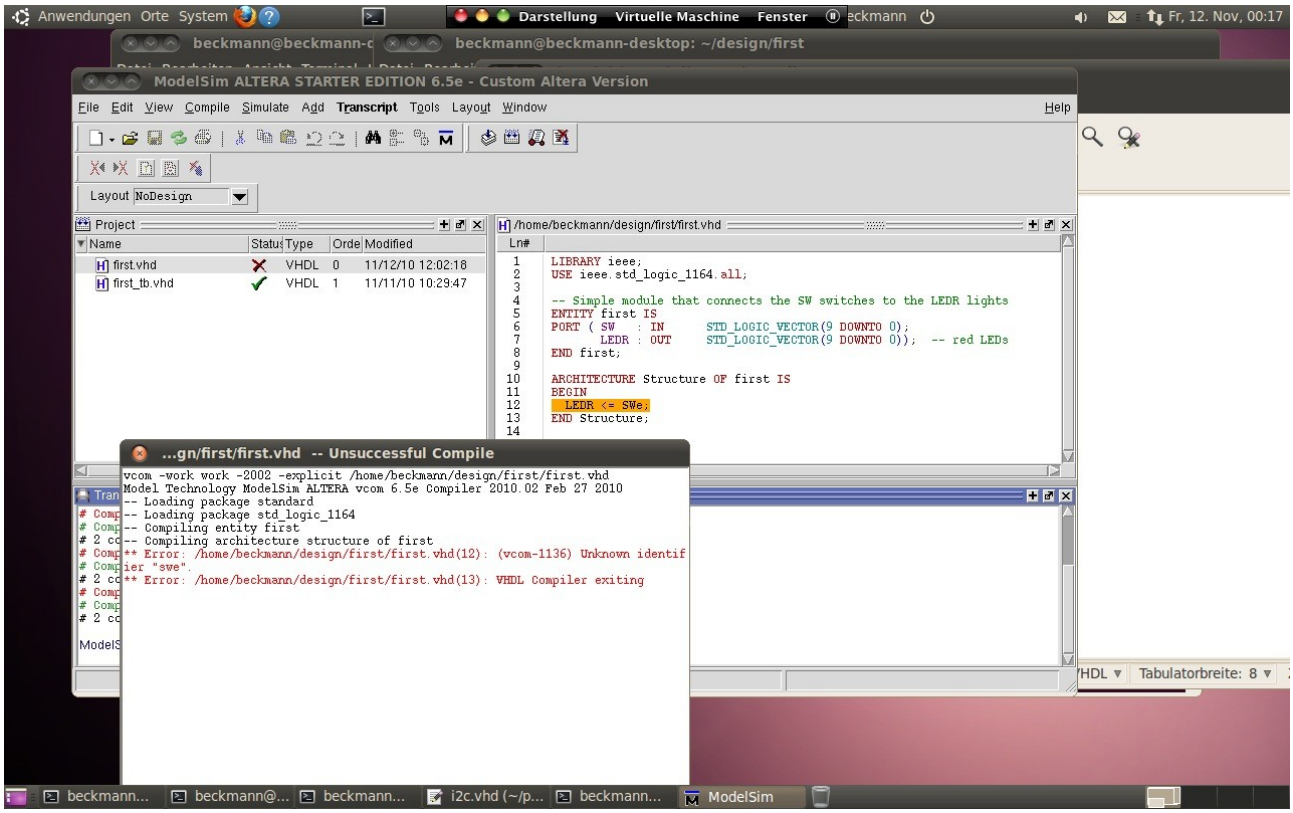
Öffnen Sie das Transcript Fenster mit „View->Transcript“.



Beim Übersetzen der Datei „first.vhd“ ist es zu einem Fehler gekommen. Sie können jetzt auf die rot markierte Fehlerzeile doppelklicken.



Es erscheint ein Fenster mit den Fehlern bei der Übersetzung. Im Beispiel hier ist ein Fehler bei der Übersetzung der Datei „first.vhd“ in Zeile 12 aufgetreten (Unknown identifier swe). Durch Doppelklicken auf diesen Fehler, wird der Fehler im Sourcecode angezeigt.



In diesem Beispiel hatte ich „SW“ durch „SWe“ ersetzt. Nach dem Korrigieren des Fehlers ist es notwendig, den Code neu zu übersetzen, bevor man eine neue Simulation durchführen kann.