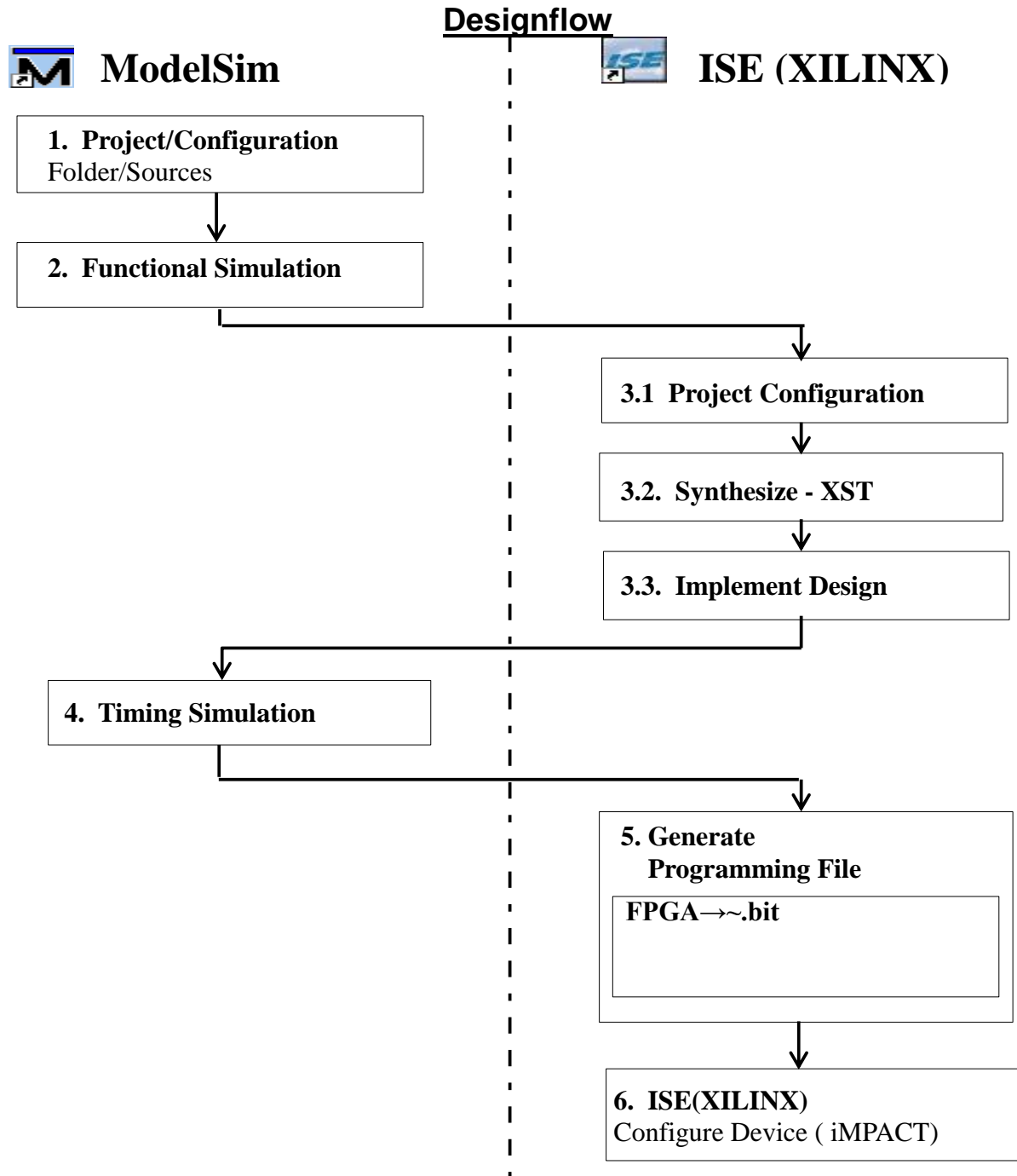


# Anleitung

Mentor Graphics – ModelSim SE III 6.3j  
XILINX – ISE 12.4



## 0.Grundsätzliche Einstellungen

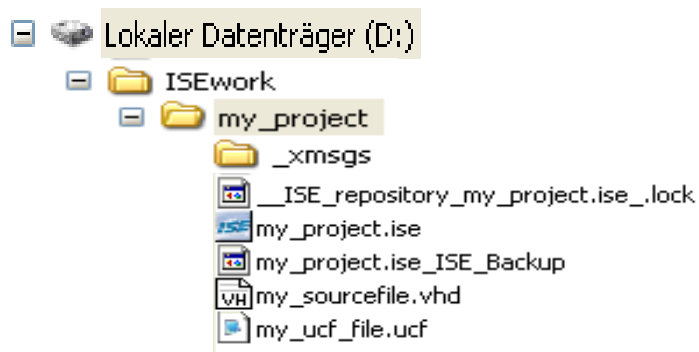
### 0.1 Vorgeschriebenes Arbeitsverzeichnis

Es ist **grundsätzlich** das Arbeitsverzeichnis **D:\ISEwork\** zu verwenden, außer es wird auf einem Speicherstick gearbeitet (nicht zu empfehlen, da lange Speicherzeiten).

### 0.2 Verzeichnis- / Projektstruktur

Die Projekte für **ISE und ModelSim** immer im gleichen Arbeitsverzeichnis in **D:\ISEwork** anlegen. Dadurch werden gleiche Änderungszustände für die Quelldateien erreicht, weil ISE und ModelSim dann auf die selben Dateien zugreifen.

**Verzeichnisstruktur:**



**Projektstruktur:**

**Zuerst** wird ein ModelSim-Projekt erstellt (siehe Punkt 1.1).

In das Projektverzeichnis D:\ISEwork\my\_project (my\_project = eigener Projektname) werden die vorbereiteten Quelldateien (~.vhd, ~.ucf) kopiert.

**Nach** der funktionalen Simulation wird im selben Projektverzeichnis ein ISE-Projekt erstellt.

Ins Projektverzeichnis <my\_project> legen ModelSim und ISE alle Arbeitsverzeichnisse und -dateien ab.

**Keine Leerzeichen** oder **Sonderzeichen** wie “ü,ö,ä,/,\?,!,”,(,),%,<,>” u.a. in Datei- oder Verzeichnisnamen verwenden!

**Keine** Dateien oder Verzeichnisse irgendwo auf **C:\** oder dem **Desktop** erstellen!

Diese gehen bei einem evtl. erforderlichen Neustart des PC's verloren!

Es ist auch erlaubt, auf dem eigenen USB-Stick statt auf D:\ISEWork zu arbeiten.

Allerdings verzögert sich der Arbeitsablauf sehr stark, da die Designtools große Datenmengen in die Arbeitsverzeichnisse schreiben bzw. von dort wieder lesen.

**Zur Beachtung!** Daten, die nach der jeweiligen Arbeitssitzung nicht auf externe Speichermedien kopiert werden, sind gegebenenfalls unwiederbringlich verloren.

# 1. ModelSim

## 1.1 ModelSim Projekterstellung

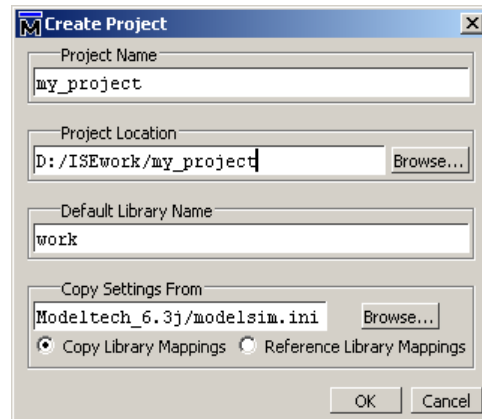
- Starten von ModelSim durch Doppelklick auf Desktop Symbol



- *File* → *New* → *Project...*

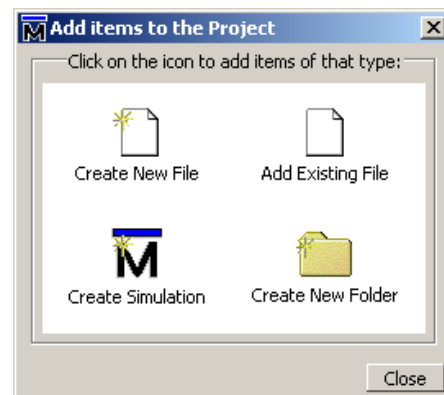
Im *Create Project* Fenster

- In *Project Location* über *Browse* das Verzeichnis *D:/ISEwork* auswählen
- Ergänzen mit dem Projektnamen *D:/ISEwork/my\_project*
- In *Project Name* den eigenen Projektnamen *my\_project* eintragen
- *Default Library Name* belassen auf *work*
- *Copy Library Mappings* auswählen
- *OK*
- Bestätigen, dass ein neuer Projektordner erstellt werden soll.



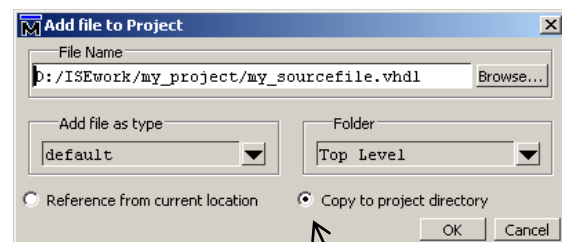
Im Fenster *Add Items to the Project*

- Klick auf *Add Existing File*

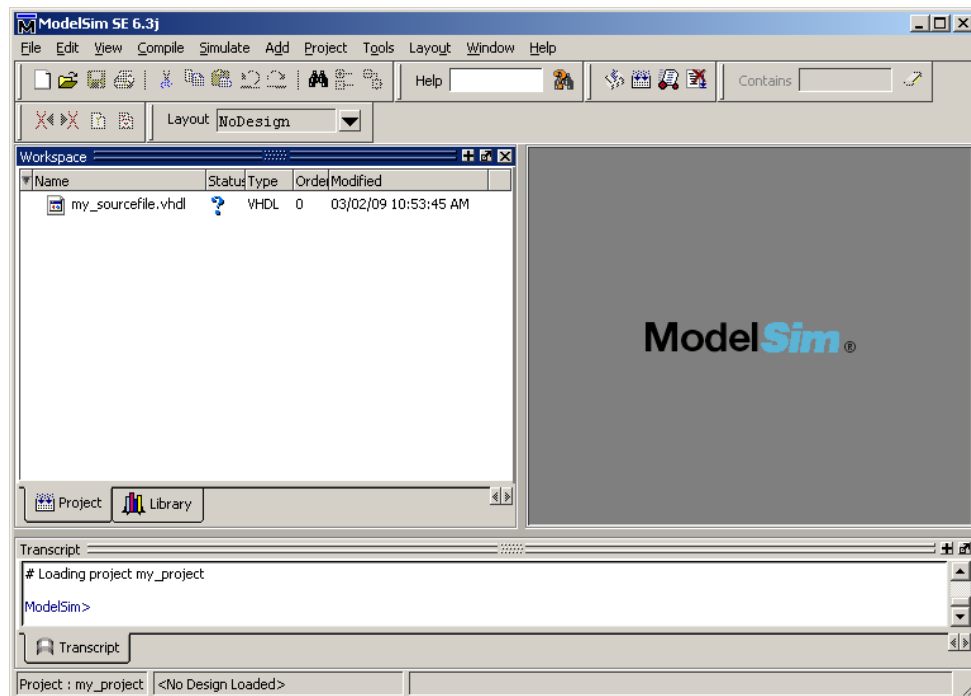


Im Fenster *Add File to Project*

- Über *Browse* Auswählen von *my\_sourcefile.vhd* aus dem Quellordner (Quellordner ist Ihr Verzeichnis, in dem die vorbereitete Datei liegt)
- *Copy to project directory* markieren
- *OK*
- Fenster *Add Items to the Project* mit *Close* schliessen



Die Datei wird dem Projekt hinzugefügt und in den Projektordner kopiert.

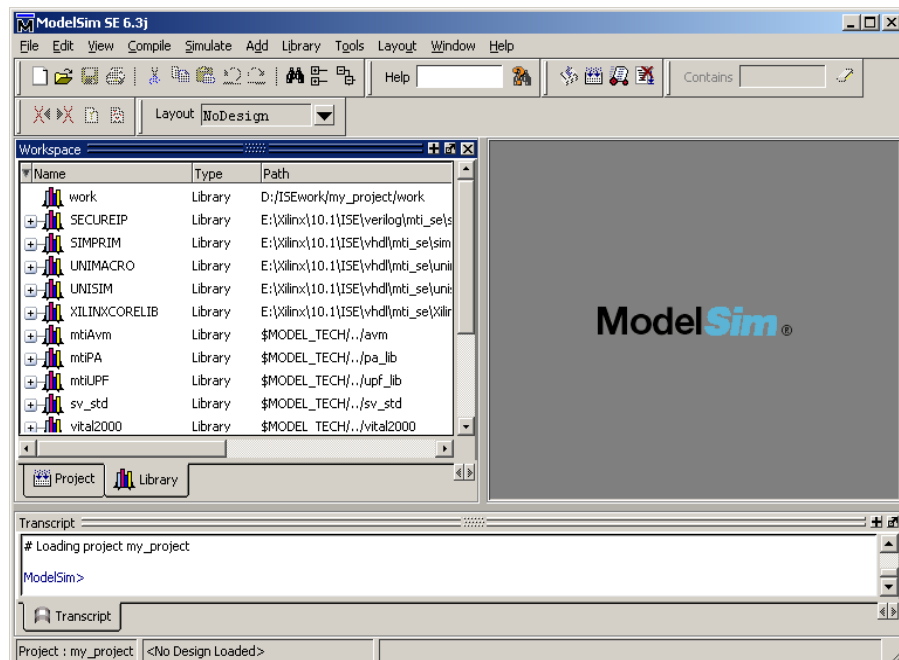


Das Projekt ist erstellt.

Im Reiter *Project* steht das eigene Quellfile. Der *Status* ist mit einem ? markiert.

Das bedeutet, das Quellfile ist noch nicht kompiliert.

Im Reiter *Library* sind die eingebundenen Libraries angegeben. Die Library „work“ ist noch leer (Es steht kein Pluszeichen davor).



## 2. ModelSim Compilierung und Simulation

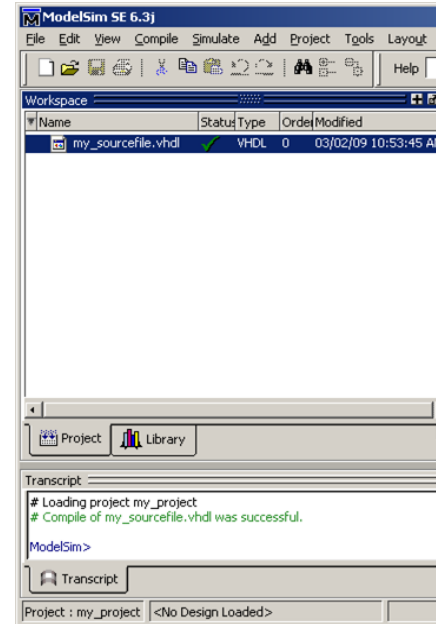
### 2.1 Compilierung

- Im Reiter *Project* das Design File *my\_sourcefile* markieren und durch Klick auf die Ikone  die Datei kompilieren.

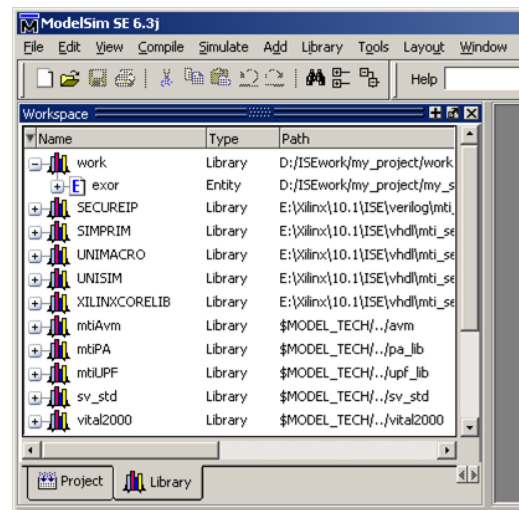
Im Reiter *Project* hat das Design File nun den Status „grünes Häkchen“, d.h. Die Compilierung ist ordnungsgemäß verlaufen.

Im Fenster *Transcript* wird in grüner Schrift eine Bestätigungsmeldung angezeigt.

Ist die Compilierung nicht erfolgreich, so ist die Fehlermeldung in roter Schrift dargestellt. Ein Doppelklick auf diese Meldung öffnet eine erweiterte Fehlerbeschreibung.

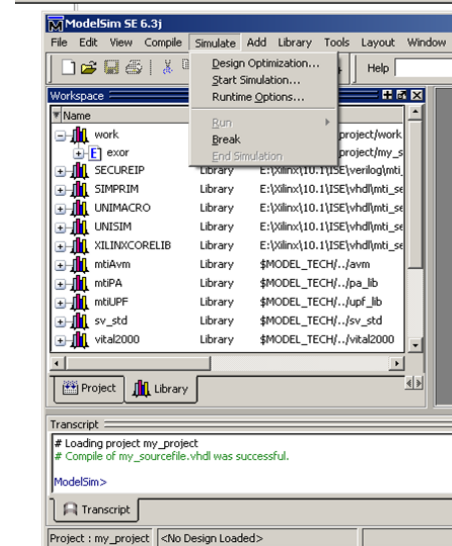


Im Reiter *Library* steht nun in *work* das compilierte Quellfile. Erkennbar am + vor *work*.



### 2.2 Simulation

- Öffne *Simulate* → *Runtime Options...*



- Setze  
*Default Run: 100 ns*  
*Default Force Type: Freeze*  
*Default Radix: Hexadecimal*

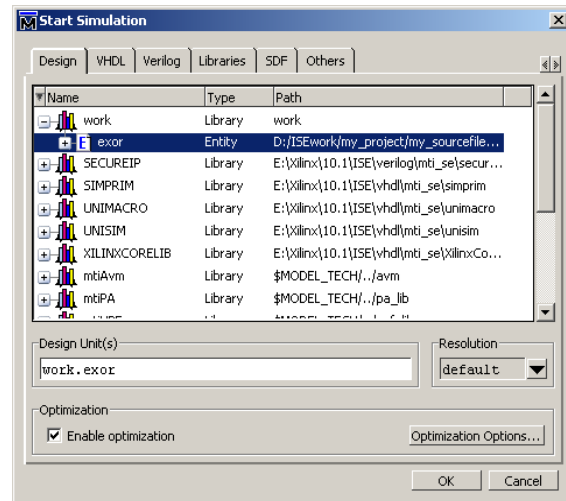
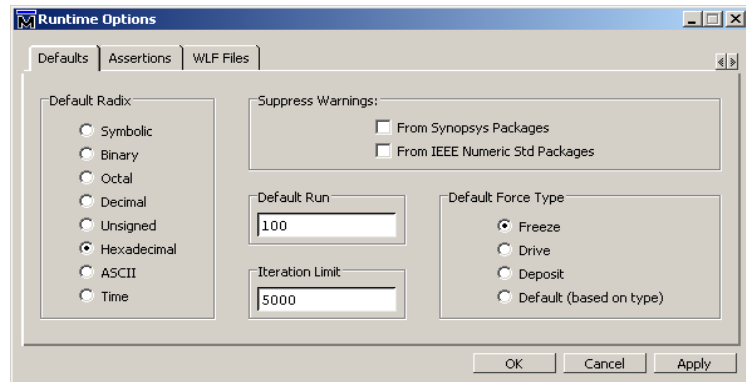
- OK

- Klick auf  (Simulate)

Es öffnet sich das Fenster  
*Start Simulation.*

- ☞ Im Reiter *Design* die Library *work* öffnen.  
zu simulierende Datei  
*my\_sourcefile.vhd* markieren.  
Beispiel heißt die zugehörige  
entity „exor“.)

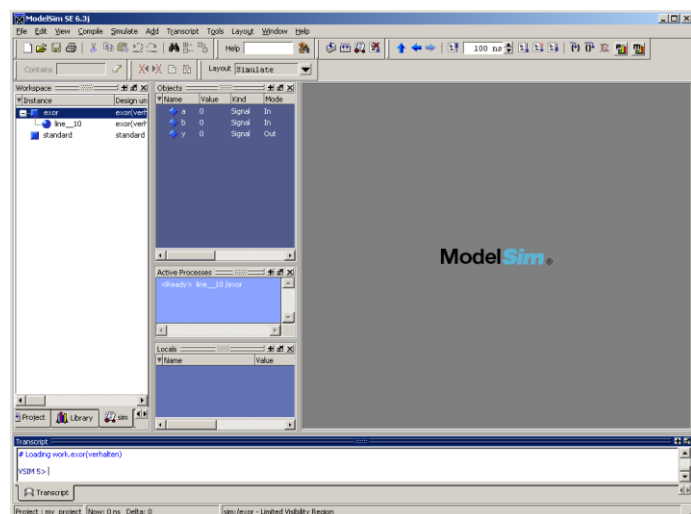
- ☞ OK  
Die Simulation wird gestartet.



Die  
(Im

Im ModelSim-Fenster erscheint ein weiteres  
Fenster *Objects*. In diesem werden die Signalnamen aufgelistet, wie sie im Design File angegeben sind.  
Im *Transcript* Fenster werden die  
zugehörigen geladenen Libraries gezeigt. Um eine Berechnung

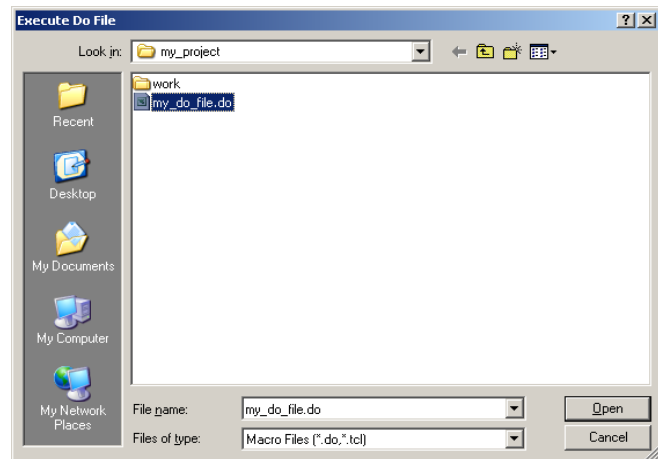
ermöglichen, werden dem Simulator  
über ein Macro (.do File) oder eine  
Testbench (.vhd file) Signalmuster für  
die Eingangssignale vorgegeben. Das  
Ergebnis ist ein Impulsmuster, welches  
im *Wave* Fenster, wie weiter unten  
gezeigt, grafisch dargestellt wird.  
Wird eine Testbench benutzt, so ist  
diese ins Projekt wie ein  
Anwendungs- File einzubinden.  
Dabei ist zu beachten, dass das  
User-Design als Komponente in die  
Testbench eingebunden ist.



- Klick auf  
*Tools* → *TCL* → *Execute Macro...*

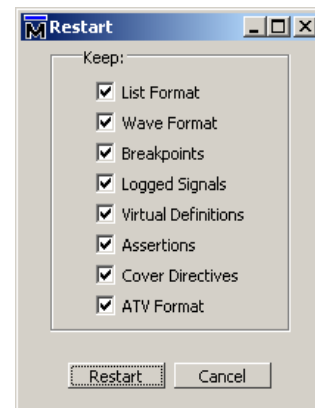
- Auswählen *my\_do\_file.do*

- Klick *Open*



(Ist das Wave Fenster noch von einer vorherigen Berechnung geöffnet, *Wave Format* abwählen, um doppelte Eintragungen im Wavefenster zu vermeiden.)

Klick *Restart*



Im Wave Fenster ist das Ergebnis zu kontrollieren.

Über die Lupenfunktionen



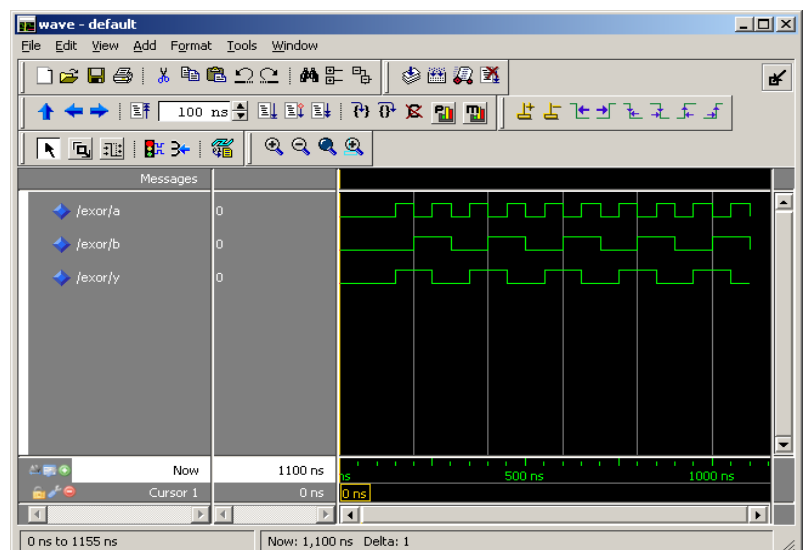
kann die Auflösung des Wave-Fensters eingestellt werden.

Durch Klick auf den Undock-Button lässt sich das Wave-Fenster ablösen und



bildschirmfüllend vergrößern.

Hiermit ist die funktionale Simulation abgeschlossen.



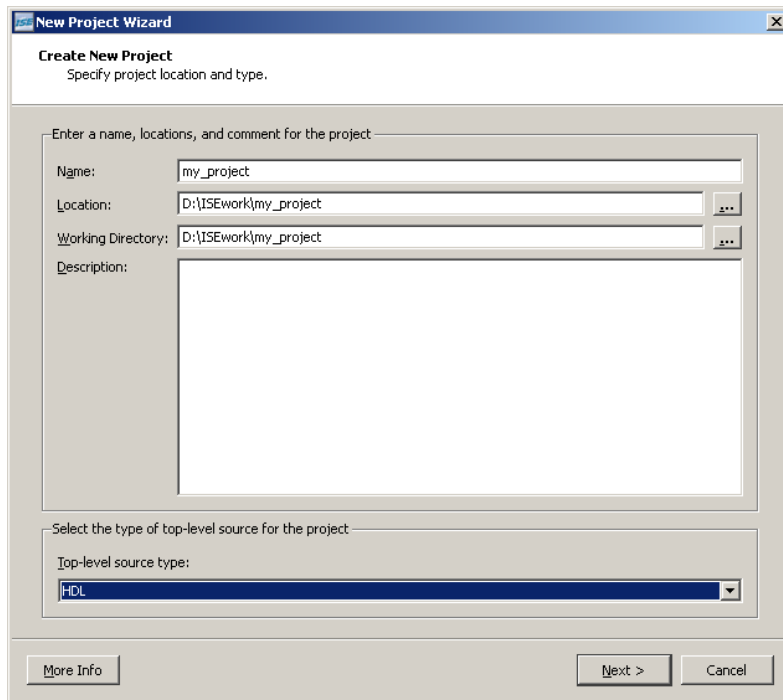
### 3. ISE

#### 3.1 ISE Projekterstellung

- Klick auf die Ikone auf dem Desktop öffnet den Projektnavigator.



*File -> New Project...* auswählen



The image shows the 'New Project Wizard' dialog box in the ISE software. The title bar reads 'New Project Wizard'. The main heading is 'Create New Project' with the subtitle 'Specify project location and type.' Below this, there is a section titled 'Enter a name, locations, and comment for the project'. It contains four input fields: 'Name' with the text 'my\_project', 'Location' with 'D:\ISEwork\my\_project', 'Working Directory' with 'D:\ISEwork\my\_project', and a large empty 'Description' text area. To the right of the 'Location' and 'Working Directory' fields are three-dot buttons. Below this section is another titled 'Select the type of top-level source for the project'. It contains a 'Top-level source type:' label and a dropdown menu currently set to 'HDL'. At the bottom of the dialog are three buttons: 'More Info', 'Next >', and 'Cancel'.

Auswahl der *Location* (unbedingt den Ordner auswählen, der in ModelSim verwendet wurde) und im Feld *Name* einen eigenen Projektnamen angeben (im Beispiel: my\_project).

**Achtung!** Keine Sonderzeichen und das Leerzeichen (Space) verwenden!

Im Feld *Project Location* wird der Eintrag my\_project **automatisch** ergänzt.

*Top-Level Source Type* auf *HDL* stellen.

*Next*



- Die Hardware über Schaltkreisfamilie, Schaltkreistyp, Gehäuse und Geschwindigkeitsklasse entsprechend Tabelle auswählen

	FPGA
<b>Family</b>	Spartan3E
<b>Device</b>	XC3S1200E
<b>Package</b>	FG320
<b>Speed</b>	-4
<b>Simulator</b>	Modelsim-SE VHDL

**New Project Wizard**

**Project Settings**  
Specify device and project properties.

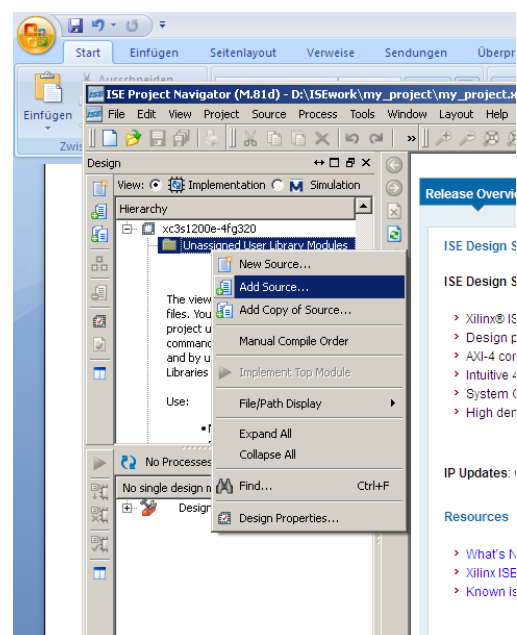
Select the device and design flow for the project

Property Name	Value
<b>Product Category</b>	All
Family	Spartan3E
Device	XC3S1200E
Package	FG320
Speed	-4
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	Modelsim-SE VHDL
Preferred Language	VHDL
Property Specification in Project File	Store all values
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input type="checkbox"/>

More Info    < Back    Next >    Cancel

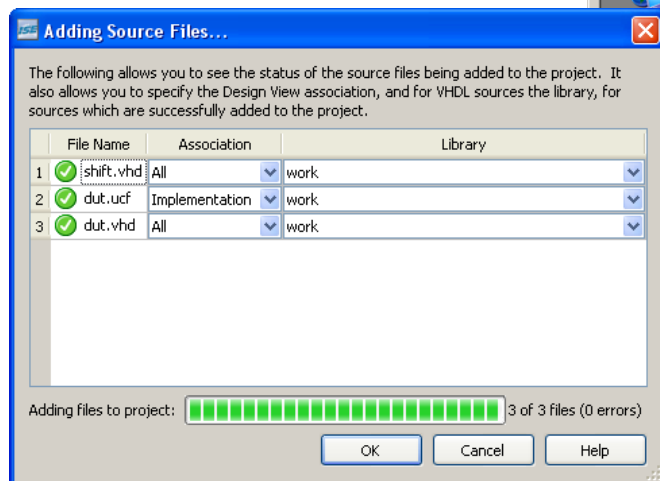
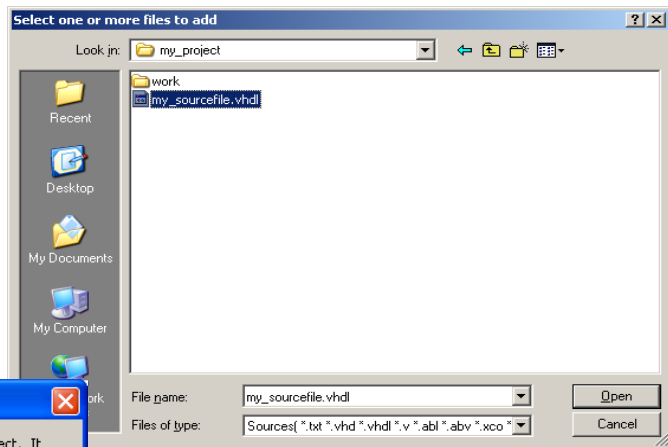
- Next
- Finish

- Über *Add Sources* werden bestehende Quellfiles eingebunden.



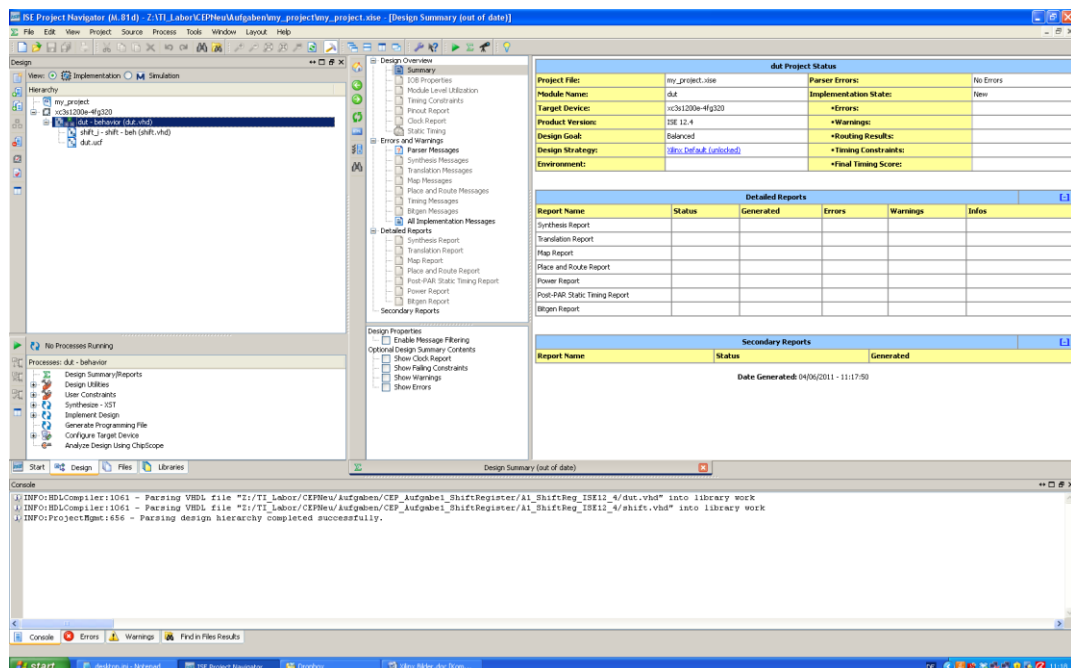
Auswählen des/der bestehenden Quellfiles. Die Dateien sollten schon im Projektverzeichnis stehen.

Es können .vhd-Files (Design-Files) und .ucf-Files (User Constraint Files) gleichzeitig eingefügt werden. (.ucf-Files beschreiben u.a. die Zuweisung der E/A-Signale auf Gehäuse-Pin.)

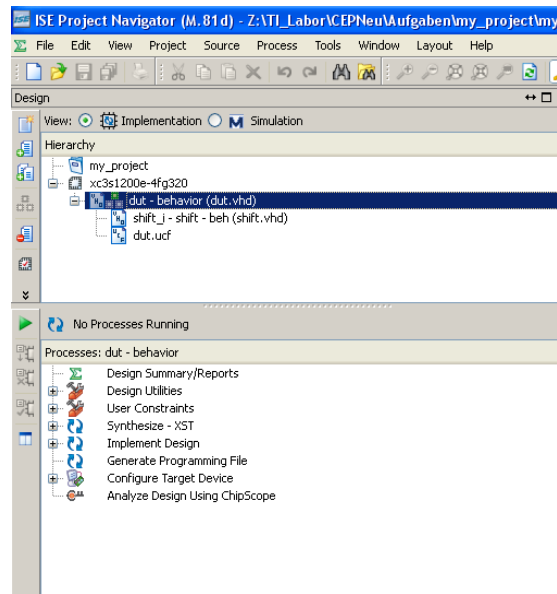


- OK

Der Bildschirm nach Beendigung des Projekt Wizzards.



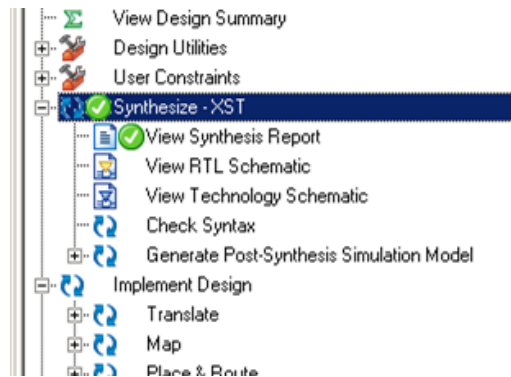
## 3.2 Synthese für FPGA



Im Registerreiter Design, Doppelklick auf „Generate Programming File“. Nun wird nacheinander „Synthesize – XST“, „Implement Design“ und „Generate Programming File“ durchgeführt. Wenn alles erfolgreich war, sieht man ein grünes Häkchen vor den entsprechenden Menü Punkten.

Bei einem gelben Dreieck sind Warnungen vorhanden, die man sich genauer ansehen sollte.

Bei einem roten Kreuz war die Synthese nicht erfolgreich.

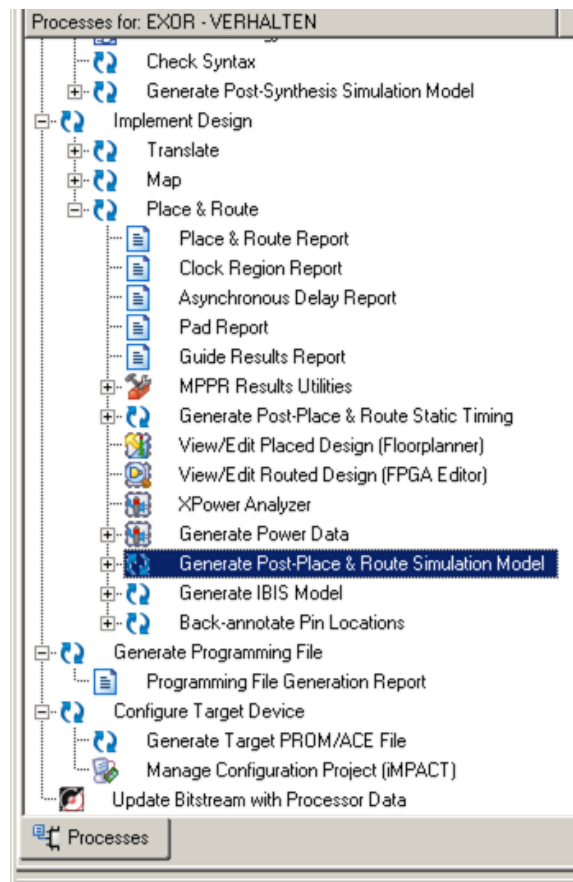


Durch Klick auf *View RTL Schematic* wird der Schaltplan angezeigt. Durch Klicken auf die Funktionsboxen lässt sich durch die Hierarchie navigieren .

### 3.3 Implementierung für FPGA

#### Achtung!

Vor der Implementierung muss unbedingt ein *User Constraint File* (*my\_ucf\_file.ucf*) in das Projekt eingebunden sein. Ohne das *User Constraint File* wird ein Programmier-File erstellt, welches wegen Benutzung von nicht erlaubten bzw. für spezielle Funktionen vorgesehene Pins eine Zerstörung des Boards verursacht.



- Im Processes Fenster -> *Implement Design* -> *Place & Route*  
Doppelklick auf *Generate Post-Place & Route Simulation Model*

Die Dateien *entityname\_timesim.vhd* und *entityname\_timesim.sdf* (z.B, *exor\_timesim.vhd*, *exor\_timesim.sdf*) werden im Verzeichnis **<project-pfad>\my\_project\netgen\par** für die Timing Simulation erzeugt.

## 4. Timing Simulation

Wenn dem Synthetisierten Code für die Timing Simulation ein Toplevel übergeordnet ist, muss die richtige Architektur, die von ISE erzeugt wurde, im Toplevel aufgerufen werden.

z.B.

component pass is

port ( so : out std\_logic;

si : in std\_logic

);

end component pass;

for all : pass use entity work.pass(Structure);

Zur Timing Simulation kann man das schon vorhandene ModelSim Projekt aus der funktionalen Simulation verwenden.

Einbinden des Quellfiles *entityname\_timesim.vhd* über

- Project → Add to Project → Existing File...

- Browse

**Für FPGA:** *D:/ISEwork/my\_project/netgen/par/entityname\_timesim.vhd*

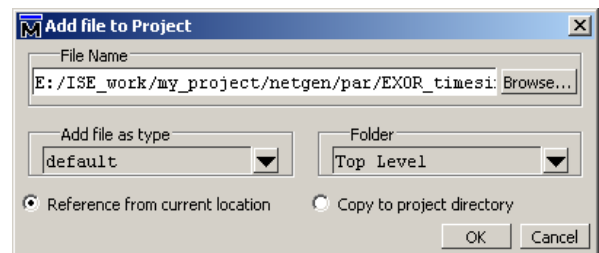
(Beispiel: Entity-Name = EXOR -> *D:/ISEwork/my\_project/netgen/par/EXOR\_timesim.vhd*)

- Reference from current location auswählen.

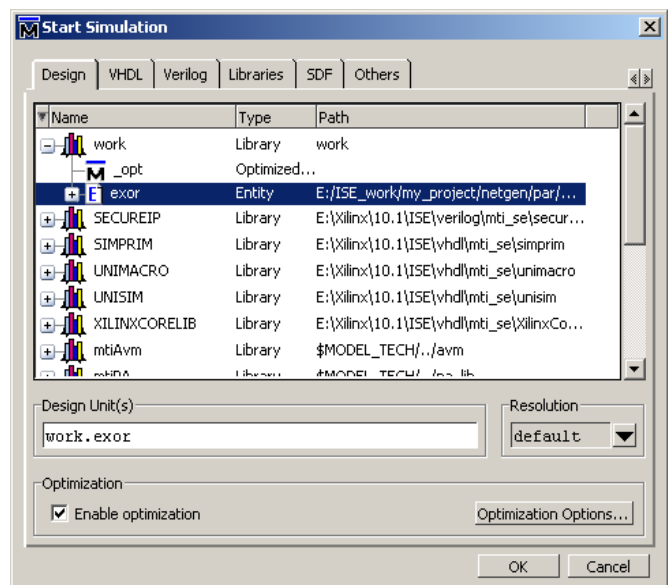


- OK

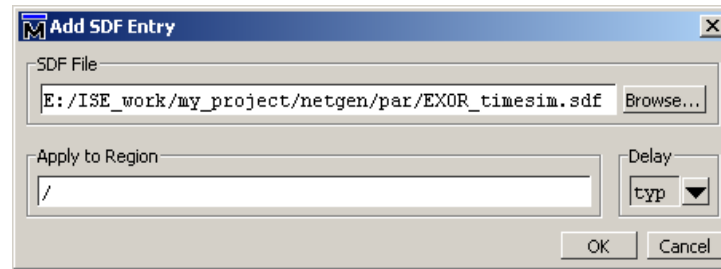
- File wie unter Pkt. 2. beschrieben compilieren und die Simulation mit starten.



- Reiter *SDF* öffnen



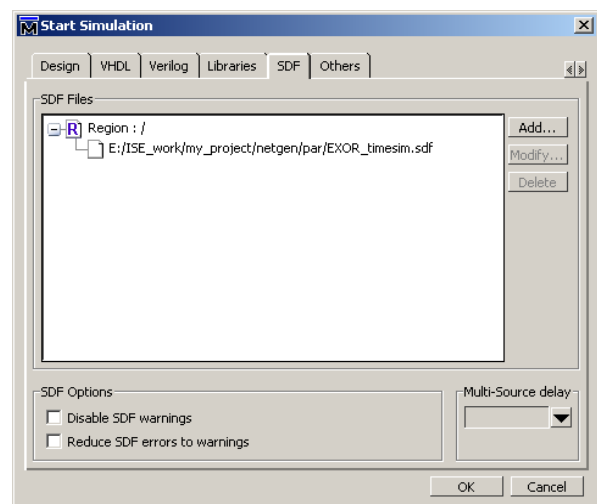
- Im Fenster *Add SDF Entry* mittels *Browse* die Datei *entityname\_timesim.sdf* aus dem Verzeichnis *D:/ISEwork/my\_project/netgen/par* auswählen.



Wenn ein Toplevel vorhanden ist, muss in **Apply to Region** der Instanzname des synthetisierten Codes eingetragen werden.

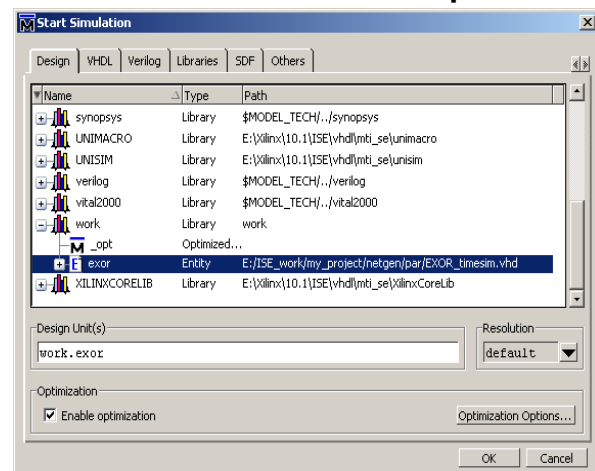
- Klicken auf *OK*

Der Pfad zur *entityname\_timesim.sdf* wird angezeigt.



- Reiter *Design* öffnen.
- Library *work* durch Klick auf **+** öffnen und **entityname\_timesim.vhd** oder wenn vorhanden den **toplevel** auswählen.

- *OK*  
Die Simulation wird gestartet



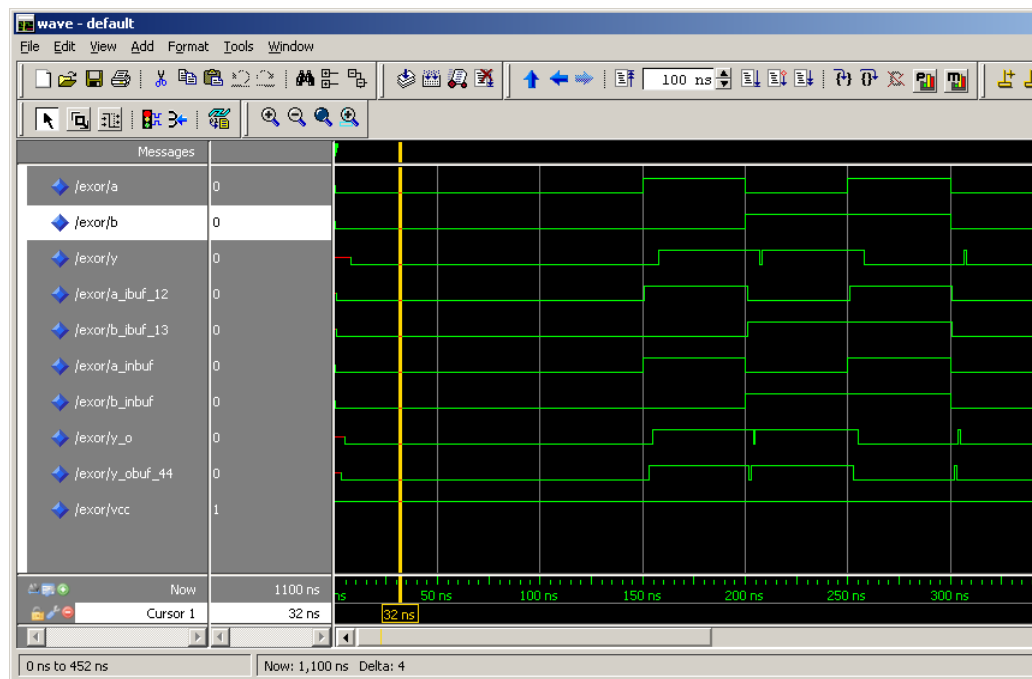
Im Object Fenster werden nun neben den Signalen aus der port map der Top-Entity weitere Signale entsprechend des Files entityname\_timesim.vhd angezeigt.

- Weiter wie in „Funktionale Simulation“ *Tools* → *TCL* → *Execute Macro...*

Es kann die gleiche .do-Datei verwendet werden, wie bei der funktionalen Simulation.

Dabei ist zu beachten:

Falls in der .do-Datei das Kommando `vsim work.xxxxx` benutzt wird, ist dieses für die Timingsimulation mit einem vorangestellten `#` auszukommentieren.



Charakteristisch für die Wavedarstellung einer Timingsimulation sind die Zeitabschnitte der „Einschwingvorgänge“ am Anfang der Zeitskala  
-hier rot dargestellt- für die der Simulator noch keine definierten Werte berechnen konnte.

- Die Funktion ist anhand der Wavetable zu kontrollieren.

## 6. Programmieren des IC

### 6.1 FPGA-Programmierung (Spartan3-Board)

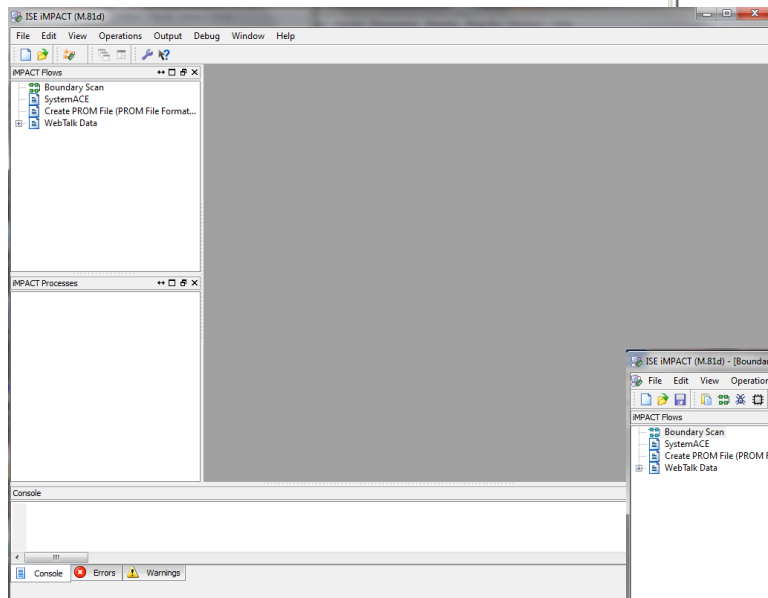
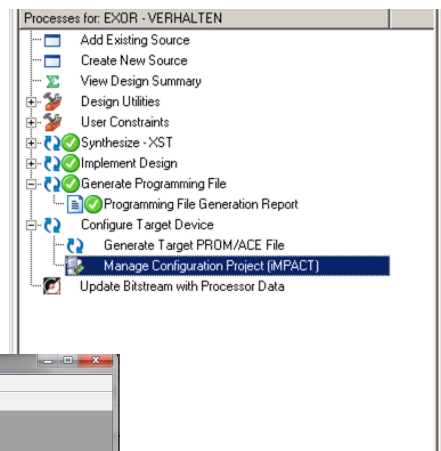
Durch *Doppelklick auf Generate Programming File* wurde ein Programmierfile *entityname.bit* im Projektordner erzeugt. Es dient als Quellfile für das Programmiertool iMPACT.

#### Vorbereitung:

Zur Programmierung des Spartan3-Boardes muss das „Platform Cable USB II“ mit dem PC über USB verbunden werden. Ist die Spannungsversorgung an das Boardes angeschlossen, so sollte die grüne Lampe am „Platform Cable USB II“ leuchten.

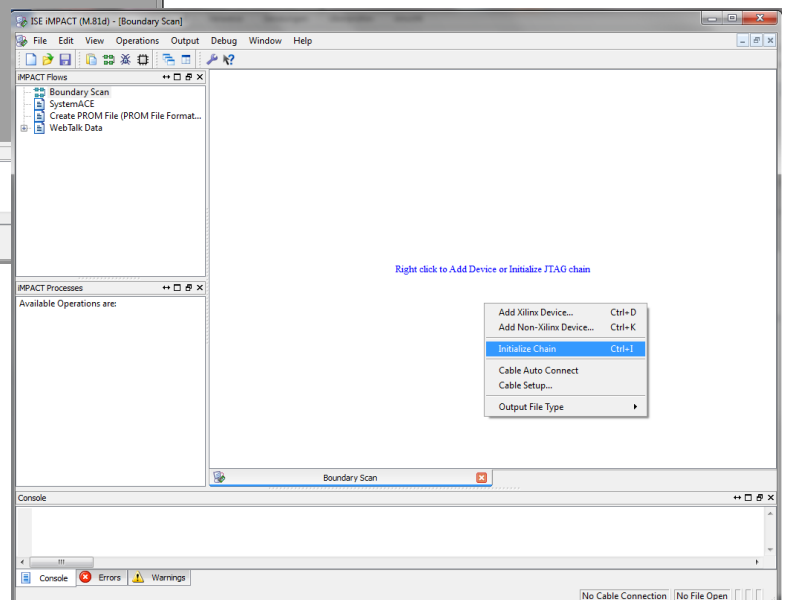
#### Programmierung:

In ISE ist das Programm iMPACT zu starten  
Configure Target Device =>  
Manage Configuration Project (iMPACT)

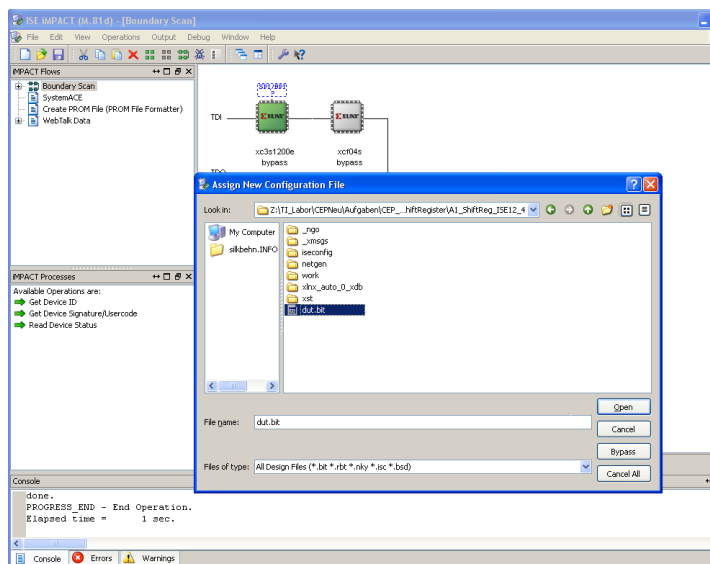


Boundary Scan  
doppelklicken

Rechtsklick im weißen Bereich  
⇒ Initialize Chain ....

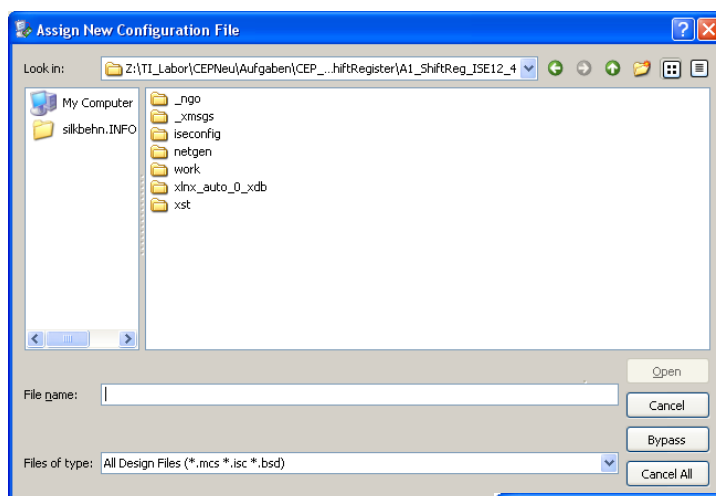
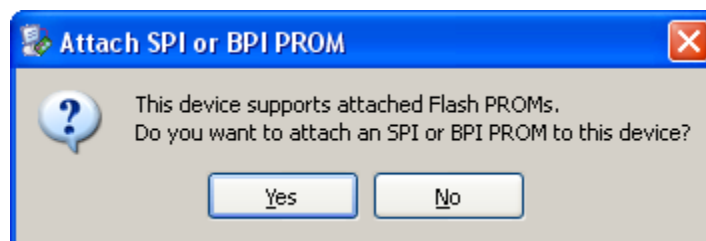






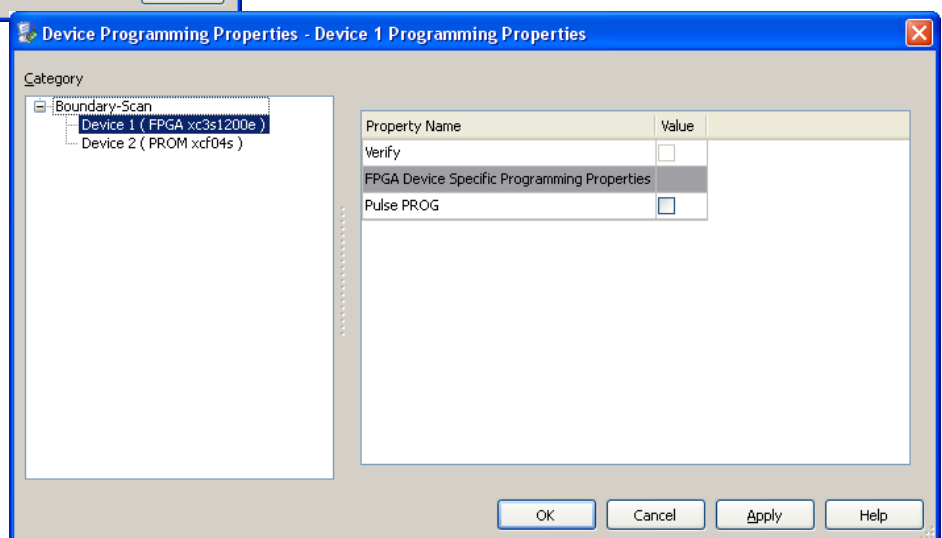
Bit File auswählen

Nein

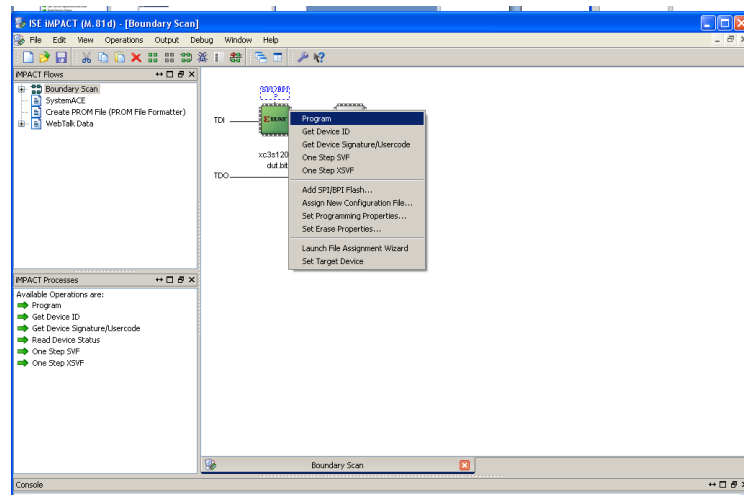


Bypass

OK



Program



Die Programmierung beginnt.

Im Fenster *Progress Dialog* wird ein Fortschrittbalken angezeigt.

Wurde die Programmierung ordnungsgemäß beendet, wird dies mit der Information *Program Succeeded* angezeigt.