Hochschule Karlsruhe – Technik und Wirtschaft
Fakultät für Informatik und Wirtschaftsinformatik
Prof. Dr. A. Ditzinger / Dipl.-Inform. (FH) O. Gniot / Dipl.-Ing. J. Krastel

Arbeiten mit ispLEVER Classic und Active HDL

ispLEVER - VHDL

ispLEVER Classic Version 1.6 arbeitet unter Win XP und Win 7. (**isp**: **i**n **s**ystem **p**rogrammable). Momentan aktuelle Version: 1.6.00.07.29.12.

ispLEVER Classic dient zum Erstellen von Programmfiles zur Programmierung von PLD's und CPLD's.

Aufruf von ispLEVER Classic 1.6:
Start /Programme /Lattice Semiconductor ispLEVER Classic 1.6/ispLEVER Classic Project Navigator.

Es öffnet sich das Fenster des ispLEVER Classic Project Navigators (siehe Bild 1).

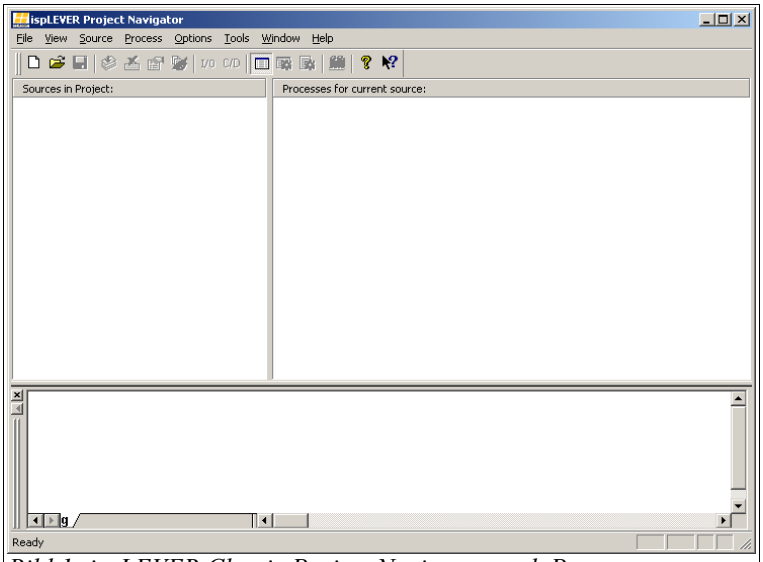


Bild 1: ispLEVER Classic Project Navigator nach Programmstart

Der „ispLEVER Classic Project Navigator" hat drei Fenster. Im linken Fenster „Sources in Project:" werden alle Quell- (Source-) Dateien, die im aktuellen Projekt verwendet werden angezeigt. Im rechten Fenster „Processes for current source:" werden alle Prozesse angezeigt, die mit der ausgewählten Quell-Datei ausgeführt werden können. Im unteren Fenster (Output Panel) werden alle Aktionen des Projektnavigator protokolliert und in die Datei „automake.log" geschrieben.

In allen erstellten Dateien sollten keine Umlaute (ä, ö, ü) und auch nicht das „ß" verwendet werden.

Damit sich ispLEVER so verhält wie es in der Anleitung beschrieben ist, müssen Sie noch ein paar Einstellungen vornehmen – siehe Anhang: „Einstellungen im ispLEVER Projekt Navigator" und „Einstellungen im Active-HDL 9.1 Simulator".

Anlegen eines neuen VHDL Projekts

Um ein neues Projekt anzulegen, wählen Sie in der Menüleiste: ***„File /New Project"*** oder „Strg-N". Füllen Sie die Felder entsprechend Bild 2 aus:

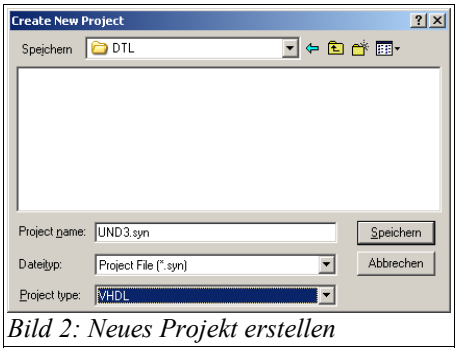


Bild 2: Neues Projekt erstellen

Location:	**N:\DTL\UND3**
Project Name:	**UND3.syn**
Dateityp:	**Project File (*syn)**
Project Type:	**VHDL**

Nach Drücken des „Speichern“-Buttons wird wird Ihr Projekt angelegt (siehe Bild 3). In unserem Projekt wird aus der Bausteinfamilie (Family) „Gal Device“ der Baustein (Device) „GAL16V8D“ automatisch ausgewählt. Der ausgewählte Baustein lässt sich durch einen Doppelklick auf den Bausteinnamen ändern.

Der Baustein „GAL16V8D“ wurde nur deshalb ausgewählt, weil er in der Datei ispsys.ini als DefaultDevice definiert wurde.

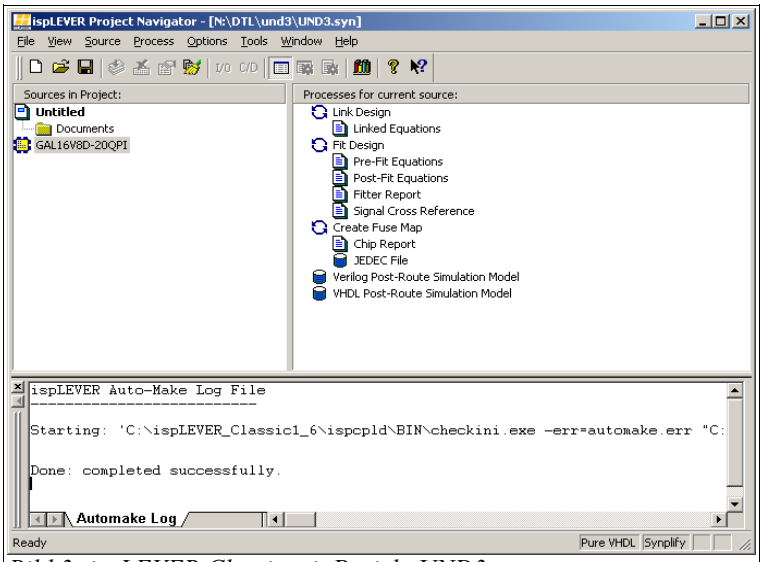


Bild 3: ispLEVER Classic mit Projekt UND3

VHDL Modul erstellen

VHDL ist eine Hardware Beschreibungssprache (VHDL = **V**HSIC **H**ardware **D**escription **L**anguage //VHSIC = **V**ery **H**igh **S**peed **I**ntegrated **C**ircuit). Zur Beschreibung der Schaltung kann ein beliebiger ASCII-Texteditor benutzt werden. In unserem Beispiel wird der integrierte Texteditor von Lattice benutzt.

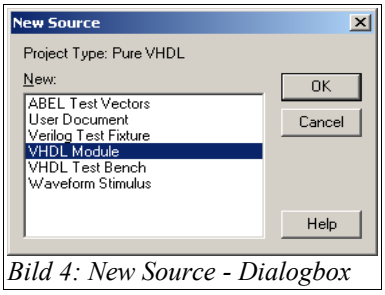


Bild 4: New Source - Dialogbox

Um ein VHDL Modul zu erstellen, wählen Sie in der Menüleiste: ***„Source /New“***. Wählen Sie in der „New Source:“ - Dialogbox (siehe Bild 4) „***VHDL Module***“ aus. Nach Drücken des OK-Buttons wird der „Text Editor“ und die „New VHDL Source“ - Dialogbox geöffnet (siehe Bild 5). Tragen Sie nachfolgende File-, Entity- und Architecture Namen ein.

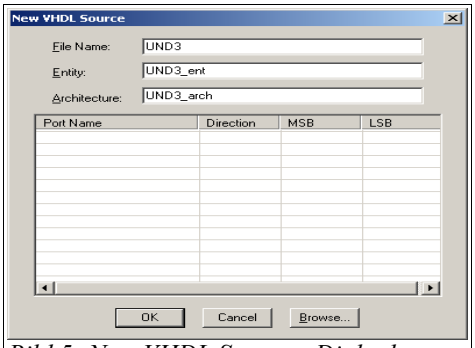


Bild 5: New VHDL Source - Dialogbox

File Name: **UND3**
Entity: **UND3_ent**
Architecture: **UND3_arch**

Nach Betätigen des OK-Buttons werden die Eingaben in den Texteditor übernommen (siehe Bild 6).

Schreiben Sie nachfolgendes Programm für ein UND-Gatter mit 3 Eingängen.

```
-----------------------------------
entity UND3_ent is
  PORT (a, b, c : IN  std_logic;
        y       : OUT std_logic);
end;
-----------------------------------
architecture UND3_arch of UND3_ent is
begin
  y <= (a AND b AND c);
end UND3_arch;
-----------------------------------
```

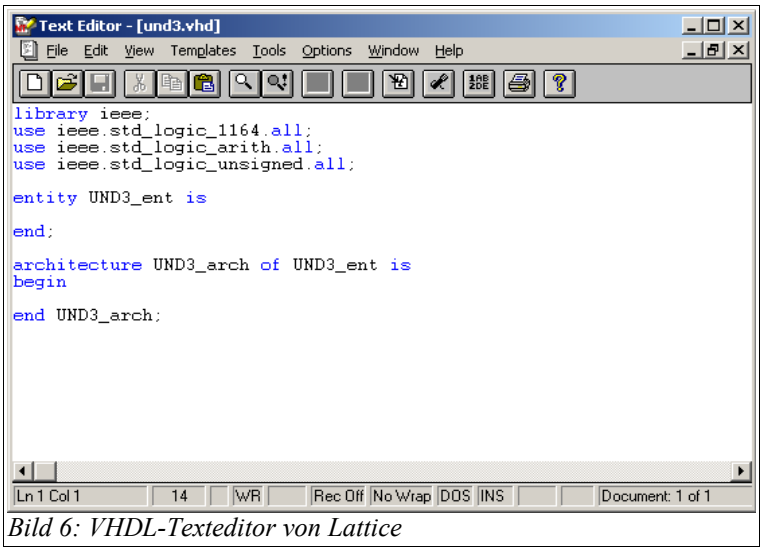


Bild 6: VHDL-Texteditor von Lattice

Nach der Eingabe des Programms speichern Sie es ab und beenden den Texteditor. Ihr Projekt müsste jetzt wie in Bild 7 dargestellt aussehen, wenn Sie als Baustein den GAL16V8D ausgewählt haben.

Im Fenster „Sources in Project“ ist das Modul und3_ent dazugekommen.

Zur Simulation der Schaltung benutzen Sie den „Active-HDL 9.1LWE“ Simulator der Firma Aldec.

Um mit dem Active-HDL Simulator eine Schaltungen zu testen ist es am einfachsten, wenn Sie sich eine Test Bench erstellen. Damit ist sichergestellt, dass alle Pfade und Directories auf das aktuelle Projektverzeichnis verweisen.

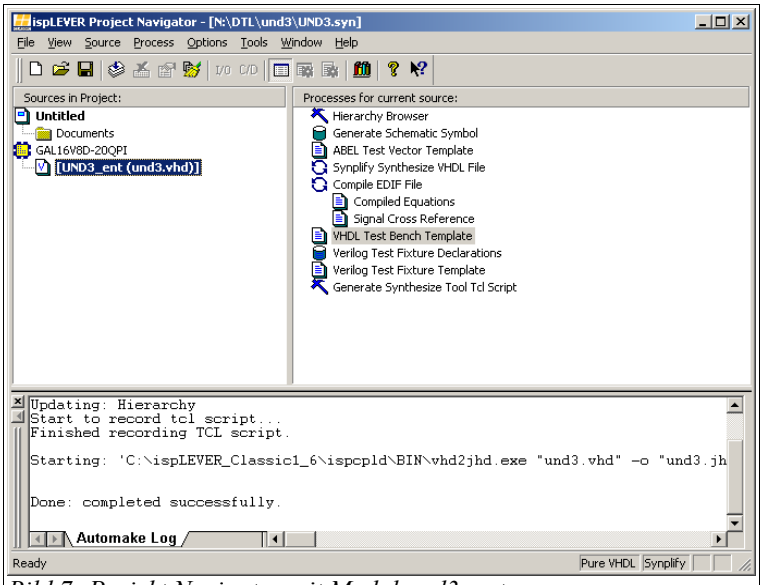


Bild 7: Projekt Navigator mit Modul und3_ent

Beachten Sie noch nachfolgende Kapitel beim Bearbeiten Ihres Projekts

Fehlerbeseitigung mit Hilfe des Simulators auf Seite 6.

Ein paar Tips zum effizienten Arbeiten mit ispLEVER auf Seite 10.

Arbeiten mit einer Test Bench

Eine Test Bench dient zur Simulation eines Schaltungsentwurfs. Man erspart sich das wiederholte Eingeben von force-Befehlen. Die Test Bench wird einmal für das Projekt erstellt. Nach Aufruf der Test Bench wird der Aldec VHDL Functional Simulator gestartet, alle benötigten Programme compiliert und im Wave-Fenster das Simulationsergebnis angezeigt (wenn keine Fehler auftreten).

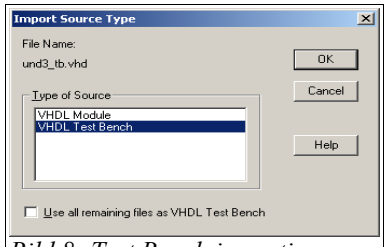


Bild 8: Test Bench importieren

Test Bench erstellen

Erstellen Sie sich als erstes ein „VHDL Test Bench Template“. Dazu müssen Sie im linken Fenster „UND3_ent“ markieren und im rechten Fenster auf „VHDL Test Bench Template“ doppelklicken (siehe Bild 7). Es wird die Datei „und3_ent.vht“ im aktuellen Projektverzeichnis erstellt und im Report Viewer angezeigt. (Environment Options müssen entsprechend gesetzt sein, siehe Anhang)

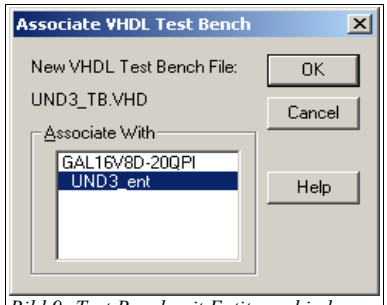


Bild 9: Test Bench mit Entity verbinden

Speichern Sie das erstellte VHDL Test Bench Template unter dem Namen „**und3_tb.vhd**“ im aktuellen Projektverzeichnis ab.

Test Bench dem Projekt hinzufügen

Mit nachfolgendem Befehl wird eine Test Bench dem aktuellen Projekt hinzugefügt: **Menüleiste: Source /Import /UND3_tb.vhd**. Es öffnet sich das Dialogfenster: „Import Source Type“ (siehe Bild 8). Wählen Sie „**VHDL Test Bench**“ aus und quittieren Sie mit OK. Es öffnet sich das Dialogfenster „Associate VHDL Test Bench“. Wählen Sie in diesem Fenster das zu testende VHDL Modul aus (und3_ent) und quittieren Sie mit OK. Die Test Bench wird in Ihr Projekt importiert (siehe Bild 10). Wenn Sie im linken Fenster „UND3_tb.vhd“ auswählen, wird im rechten Fensterbereich „Aldec VHDL Functional Simulation“ angezeigt.

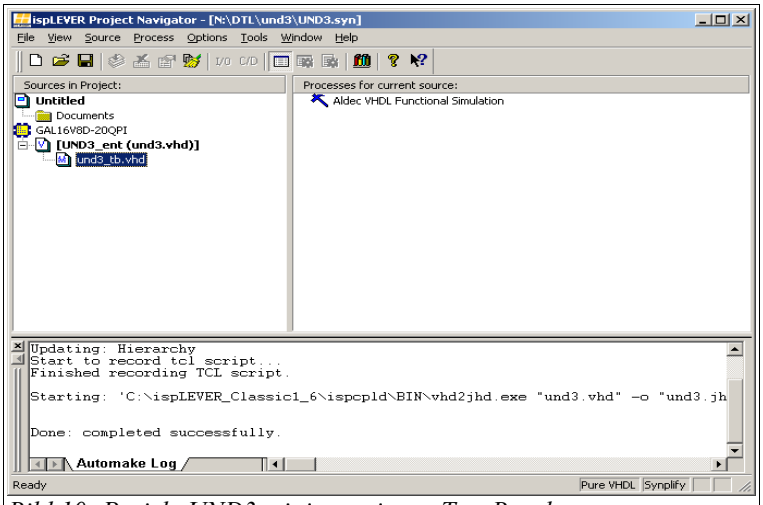


Bild 10: Projekt UND3 mit importierter Test Bench

Anpassen und starten der erstellten Test Bench

Der ENTITY-Name der erstellten Test Bench heißt testbench. Auch die Architecture bezieht sich auf den Namen testbench (siehe Bild 11). Wenn Sie mehrere Test Benchs in Ihrem Projekt verwenden, müssen Sie den Namen testbench umbenennen. Der ENTITY-Name der Test Bench sollte nach dem Dateinamen der Test Bench-Datei umbenannt werden. In unserem Fall UND3_tb. Sie müssen also insgesamt an drei Stellen in der Test Bench den Namen ändern.

```
LIBRARY ieee;
LIBRARY generics;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE generics.components.ALL;

ENTITY testbench IS
END testbench;

ARCHITECTURE behavior OF testbench IS

        COMPONENT UND3_ent
        PORT(
                a : IN std_logic;
                b : IN std_logic;
                c : IN std_logic;
                y : OUT std_logic
                );
```

Bild 11: Auszug aus der Test Bench

Um die Schaltung zu simulieren doppelklicken Sie auf „Aldec VHDL Functional Simulation“. Der Aldec VHDL Simulator wird gestartet (siehe Bild 12).

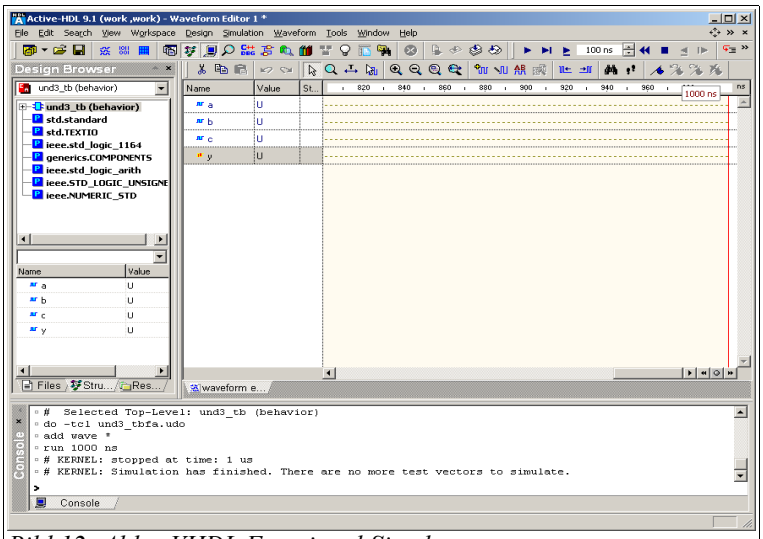


Bild 12: Aldec VHDL Functional Simulator

Sie müssen jetzt die Eingangs-Signale manuell vorgeben. Dazu klicken Sie mit der rechten Maustaste auf das zu simulierende Signal z. B. auf „a“. Es öffnet sich eine Dialogbox (siehe Bild 13). Wählen Sie „Stimulators“ aus. Es öffnet sich eine weitere Dialogbox (siehe Bild 14). Wählen Sie das zu verändernde Signal aus. In unserem Beispiel das Signal „a“. Als (Signal-) Typ wählen Sie „Clock“. Ändern Sie die Zeitdauer für eine Periode auf 50ns ab. Quittieren Sie mit „Apply“ und „Close“. Machen Sie die gleichen Änderungen auch für die Signale b und c. Wählen Sie für b eine Zeitdauer von 100ns und für c eine Zeitdauer von 200ns.

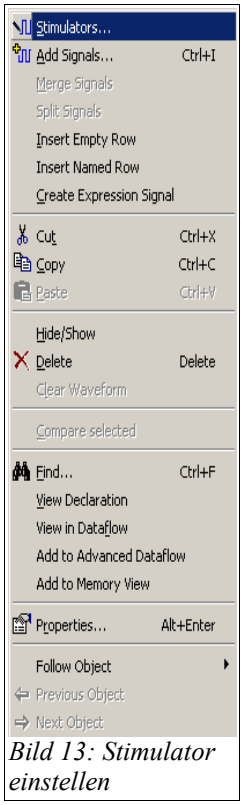


Bild 13: Stimulator einstellen

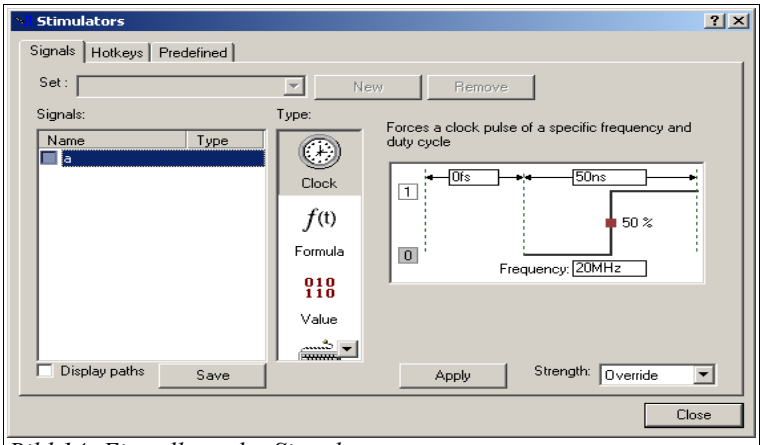


Bild 14: Einstellung des Signals

Geben Sie in der Console den Befehl „run 400ns“ ein. Die Simulation wird für 400ns angezeigt (siehe Bild 15).

In gleicher Weise wie oben beschrieben können Sie auch andere Signaltypen auswählen.

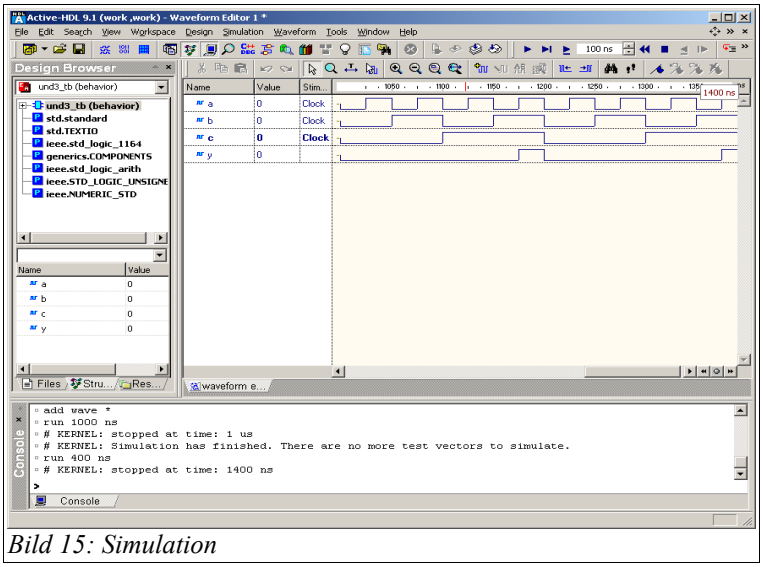


Bild 15: Simulation

Test Bench erweitern

Um die Test Bench zu ändern doppelklicken Sie auf „UND3_tb.vhd“ (siehe Bild 10). Fast am Ende der geöffneten Test Bench steht:
-- *** Test Bench - User Defined Section ***
(siehe Bild 16). Nach diesem Eintrag können Sie Ihr Testprogramm einfügen (siehe Bild 17).

```
-- *** Test Bench - User Defined Section ***
  tb : PROCESS
  BEGIN
    wait; -- will wait forever
  END PROCESS;
-- *** End Test Bench - User Defined Section ***
END;
```

Bild 16: Einfügen von Testvektoren

Es gibt verschiedene Möglichkeiten Signalverläufe zu definieren. Für Signal a wird die Zeit direkte eingegeben. Für die Si-

gnale b und c werden die Testsignale in einer Schleife gebildet. Für jedes Signal wird ein eigener Process erzeugt.

```
-- *** Test Bench - User Defined Section ***
 a_sig: PROCESS
  BEGIN
    a <= '0', '1' after  50 ns,  '0' after 100 ns, '1' after 150 ns,
              '0' after 200 ns, '1' after 250 ns, '0' after 300 ns,
              '1' after 350 ns, '0' after 400 ns;
    wait;
 end process;

 b_sig: PROCESS
  BEGIN
      b <= '0';
      loop
       wait for 100 ns;
       b <= not b;
      end loop;
 END PROCESS;

 c_sig: PROCESS
  BEGIN
      c <= '0';
      loop
       wait for 200 ns;
       c <= not c;
      end loop;
 END PROCESS;
-- *** End Test Bench - User Defined Section ***
```

Bild 17: Test Bench für UND3

Test Bench ausführen

Wählen Sie im Project Navigator im linken Fensterbereich das Test Benchprogramm „UND3_tb.vhd“ aus. Doppelklicken Sie im rechten Fensterbereich auf den Eintrag „VHDL Functional Simulation“ (siehe Bild 10). Die Test Bench wird ausgeführt und die Simulation wird gestartet. Der Vorteil dieser Erweiterung ist nun, dass Sie die Signalverläufe für a, b und c nicht mehr eingeben müssen.

Fehlerbeseitigung mit Hilfe des Simulators

Sollten gelegentlich Fehler in der VHDL-Datei auftreten, werden diese im Active-HDL Simulator in der entsprechenden VHDL-Datei rot unterlegt angezeigt.

Beseitigen Sie den Fehler und speichern Sie dann die Datei ab (Strg-S). Zum erneuten Starten der Test Bench wählen Sie nachfolgenden Befehl: Menüleiste: Tools /Execute macro ... /UND3_tb_activehdl.do.

„UND3_tb_activehdl.do“ ist der Makroname der für Ihr Projekt automatisch erstellt wurde. Bei einem anderen Projektnamen heißt das Makro entsprechend anders.

Beispiel mit einem Bus: x2..x0

```
entity UND3_Bus_e is
  PORT (x : in  std_logic_vector (2 downto 0);
        y : out std_logic);
end;

architecture UND3_Bus_a of UND3_Bus_e is
begin
  y <= (x(2) and x(1) and x(0));
end UND3_Bus_a;
```

und die passende Test Bench dazu

```
USE ieee.std_logic_unsigned.ALL;

-- *** Test Bench - User Defined Section ***
  x_sig: PROCESS
         BEGIN
           x <= "000";
           loop
           wait for 100 ns;
           x <= (x + 1);
           end loop;
         END PROCESS;
-- *** End Test Bench - User Defined Section ***
```

Einem Projekt einen Baustein zuordnen

Nachdem die Simulation erfolgreich abgeschlossen wurde können Sie, falls noch nicht geschehen, Ihrem Projekt einen Baustein zuordnen. Wählen Sie den GAL-Baustein „GAL16V8D“ aus (siehe Bild 3 und auf Seite2).

```
entity und3_ent is
  PORT (a, b, c : IN  std_logic;
        y       : OUT std_logic);

  attribute loc      : string;
  attribute loc of a : signal is "P2";
  attribute loc of b : signal is "P3";
  attribute loc of c : signal is "P4";
  attribute loc of y : signal is "P19";
end;
```

Bild 18: Erweiterung des Quelltextes

Um den PORTs a, b, c und y Bauteile-Pins zuzuordnen, müssen Sie die entity Ihres Quelltextes wie in Bild 18 dargestellt erweitern. Den PORT's a, b, c und y werden die Pins 2, 3, 4 und 19 zugewiesen. Wenn Sie diese Erweiterung nicht machen werden die Pins vom Programm festgelegt und Sie müssen im Chip-Report nachschauen wo die Anschlüsse zu finden sind.

Kompilieren der fertigen Schaltung

Im Sources-Fenster den Baustein „**GAL16V8D**“ auswählen. Im Processes-Fenster auf „**JEDEC File“** doppelklicken. Die Kompilierung beginnt. Bei fehlerfreier Kompilierung erscheint ein grüner Haken links neben JEDEC File. Es wird eine Fuse Map-Datei mit dem Namen „und3.jed“ erzeugt. Außerdem können Sie sich eine Chip-Report Datei erzeugen lassen. Sie hat den Namen und3.rpt. In dieser Chip-Report Datei ist ein „Chip Diagram“ vorhanden (siehe Bild 20). Hier können Sie kontrollieren ob die vorgegeben Pins mit den verwendeten Pins übereinstimmen.

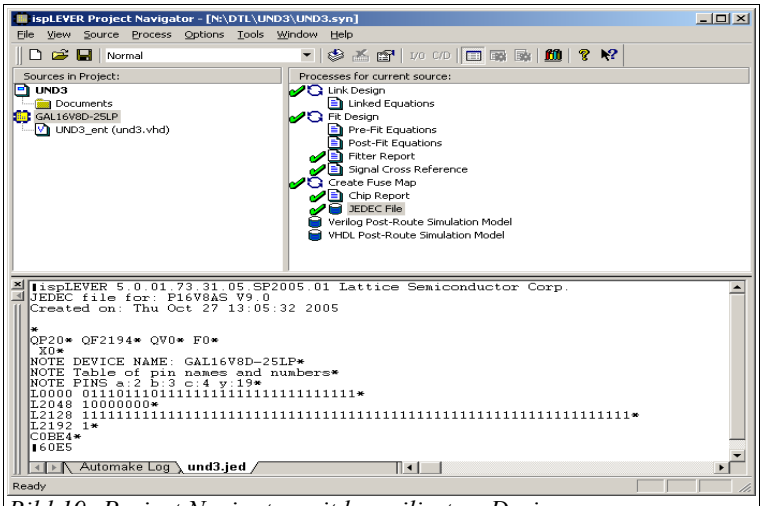


Bild 19: Project Navigator mit kompiliertem Design

Ist beim Kompilieren ein Fehler aufgetreten, wird ein Report Viewer gestartet, der den möglichen Fehler anzeigt. Nach Beseitigung des Fehler muss nochmals kompiliert werden.

Im Projektverzeichnis wird bei fehlerfreier Kompilierung ein JEDEC-File mit dem Namen „und3.jed“ erzeugt. Diese Datei wird an das Programmiergerät übertragen.

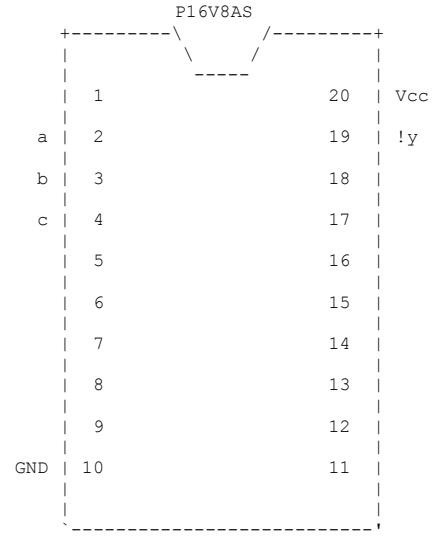


Bild 20: Teil des Chip-Reports

Bedienung des Programmiergerätes GALEP-5 von CONITEC

Das Programmiergerät GALEP-5 (siehe Bild 21) wird über die USB- Schnittstelle angeschlossen und kann mit jedem PC im Digital- und Mikrocomputer-Labor verwendet werden.

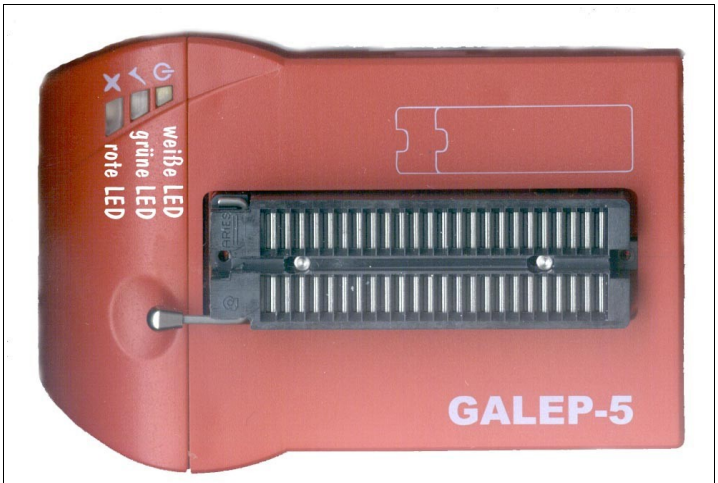


Bild 21: Programmiergerät GALEP-5

Nach dem Anschluss an eine USB-Schnittstelle durchläuft das Programmiergerät eine Initialisierungsphase die nachfolgend beschrieben ist:

- Nach 5 Sekunden fängt die weiße LED an zu blinken.
- Nach 15 Sekunden leuchtet die weiße LED.
- Nach 60 Sekunden leuchten die weiße und rote LED. Das Programmiergerät ist nun betriebsbereit.

Das Programmiergerät GALEP-5 wird mit dem Programm „GALEP5“ bedient, dass mit nachfolgendem Befehl gestartet wird:

- Start /Programme /GALEP5/GALEP5

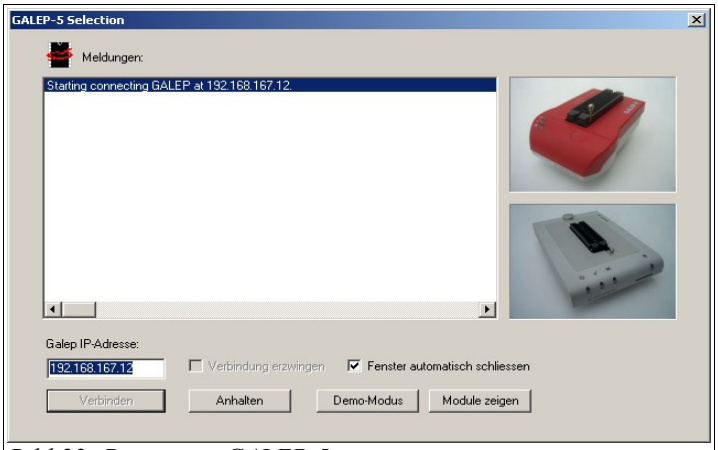


Bild 22: Programm GALEP-5

Es öffnet sich das Fenster „GALEP-5 Selection“ siehe Bild 22. Ist das Programmiergerät betriebsbereit versucht sich das gestartete Programm mit dem Programmiergerät zu verbinden. Nach einer weiteren Initialisierungsphase geht die rote LED aus und die grüne LED an. Das Programmfenster sieht nun etwa aus wie in Bild 24. Es ist noch kein Bauteil ausgewählt.

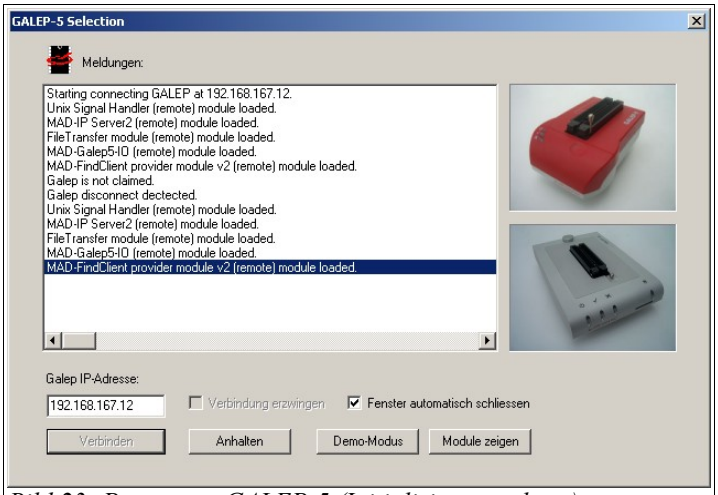


Bild 23: Programm GALEP-5 (Initialisierungsphase)

Bauteile auswählen

Zur Auswahl eines Bauteils drücken Sie die Funktionstaste F8 oder wählen das Register „Bauteile“ aus. Es öffnet sich ein Fenster in dem Sie unter verschiedenen Bauteilefamilien auswählen können. Wählen Sie den Baustein „PLD/ LATTICE/ **GAL 16V8D**“ im DIL-Gehäuse aus - siehe Bild 26. Wenn Sie danach das Register „Arbeitsbereich“ auswählen, sehen Sie das ausgewählte Bauteil und den entsprechenden Daten-Puffer.

Programm-Datei laden

Um Ihre mit ispLever erzeugte Programm-Datei (*.jed) zu laden, verwenden Sie nachfolgenden Befehl: „Menüleiste: Datei /Laden“. Wählen Sie Ihre Progamm-Datei aus und quittieren Sie mit „Laden“. Die Programm-Datei wird in einen Daten-Puffer im PC geladen.

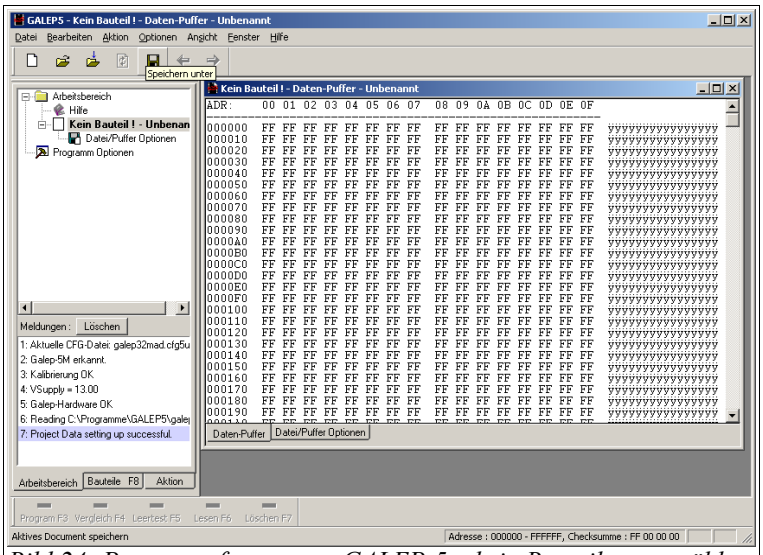


Bild 24: Programmfenster von GALEP-5 – kein Bauteil ausgewählt

Baustein programmieren

In Bild 25 ist der untere linke Bereich aus Bild 24 vergrößert dargestellt. Durch Drücken des entsprechenden Buttons bzw. der entsprechenden Funktionstaste kann ein Bauteil programmiert F3 oder mit dem Inhalt des Daten-Puffers verglichen werden F4. Setzen Sie das Bauteil erst in das Programmiergerät, wenn Sie dazu aufgefordert werden und zwar so, wie es auf dem Bildschirm dargestellt ist. Bevor Sie das Bauteil einsetzen, müssen Sie einen kleinen silbernen Hebel nach oben drücken. Dadurch werden die Pins des Nullkraftsockels [ZIF-Sockel (**Z**ero **I**nsertion **F**orce)] geöffnet. Setzen Sie das Bauteil ein und drücken Sie den kleinen Hebel wieder nach unten. Das Bauteil wird in dem Sockel festgeklemmt. Drücken Sie den Start-Button. Die Programmierung beginnt. Nach fehlerfreier Programmierung sollte die Meldung „Program OK“ erscheinen. Nach Hochdrücken des kleinen Hebels können Sie das Bauteil herausnehmen.

Bild 25: Funktionstasten

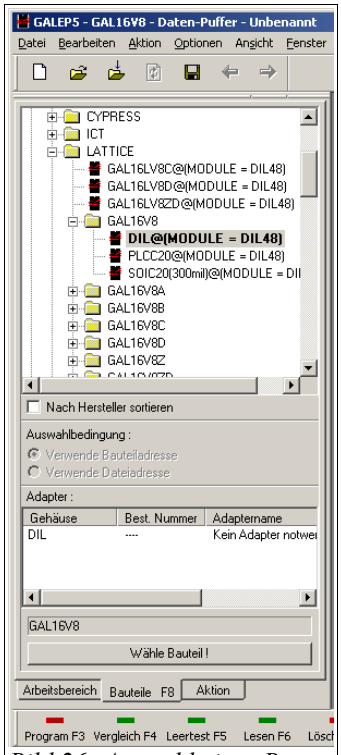


Bild 26: Auswahl eines Bauteils

Mit der Funktionstaste F6 können Sie den Inhalt eines Bauteils in den Daten-Puffer einlesen (Auslesen des Bauteils) und mit der Funktionstaste F7 wird das Bauteil gelöscht.

Datenblatt des Bausteins GAL 16V8D

Das Datenblatt des Herstellers ist in nachfolgend aufgelisteten Verzeichnissen auf dem LAT-AD-01 Server zu finden.

Arbeitsplatz: W:\Datenblätter\PALCE16V8.pdf

Pin 11 (OE - Output Enable) des PALCE16V8 muss an Pin 10 (GND) angeschlossen werden.

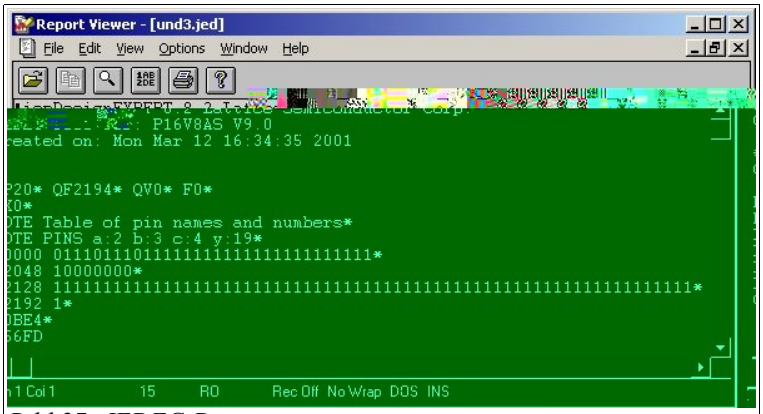


Bild 27: JEDEC-Datei

Ein paar Tips zum effizienten Arbeiten mit ispLEVER

ispLEVER besteht aus 3 Teilen (Project Navigator, Simulator und Sythesetool), von denen wir 2 interaktiv nutzen. Jedes dieser Tools hat einen eigenen Editor und eine eigene Bedienung. Wichtig ist es, das jeweils richtige Tool für den Zweck zu nehmen und jeweils nur einen Editor zu aktivieren.

1. Debugging

Wenn Sie ein neues Design mit dem Navigator angelegt haben, können Sie den Quellcode mit dem Editor des Navigators schreiben. Wenn es aber ans Debuggen geht, sollten Sie nicht auf "Synplify Synthesize VHDL File" drücken, denn das dauert erstens lang und zweitens ist die Diagnose nicht so doll. Stattdessen machen Sie zunächst mal den Editor des Navigators zu!! Sonst bleibt die alte Version des Quellcodes hier stehen und beim späteren Schließen des Fensters antworten Sie auf die Frage Speichern? zu 99% mit Ja. Typischer Spruch: "Also ich kann mir das gar nicht erklären, ganz plötzlich sind die ganzen Fehler, die ich schon ausgebaut hatte, wieder drin."

Zum Debuggen starten Sie gleich den Simulator über eine Test Bench. Fehler werden in der VHDL-Datei rot unterlegt angezeigt. Korrigieren Sie den/die Fehler, speichern nicht vergessen und starten Sie das Makro (Menüleiste: Tools /Execute macro ... /xxx.do) (xxx ist Ihr Makroname). So haben Sie schnell ein korrektes Design.

2. Debuggen der Test Bench

Sie können die Test Bench aus dem Navigator heraus aufrufen. (Doppelklick auf VHDL Functional Simulation). Intern wird dann ein Simulator Makro (irgendwat.do) erstellt und gestartet. Das Makro zeigt gleich die richtigen Signale an und startet die Simulation. Feine Sache.

Was aber, wenn die Test Bench fehlerhaft ist? Dann gilt das unter 1 Gesagte: Bleiben Sie im Simulator, doppelklicken Sie auf die Fehlermeldung und korrigieren Sie den Fehler. Dann lassen Sie einfach das Makro wieder ablaufen (Menüleiste: Tools /Execute macro ... /irgendwat.do)" absuchen und ausführen lassen.

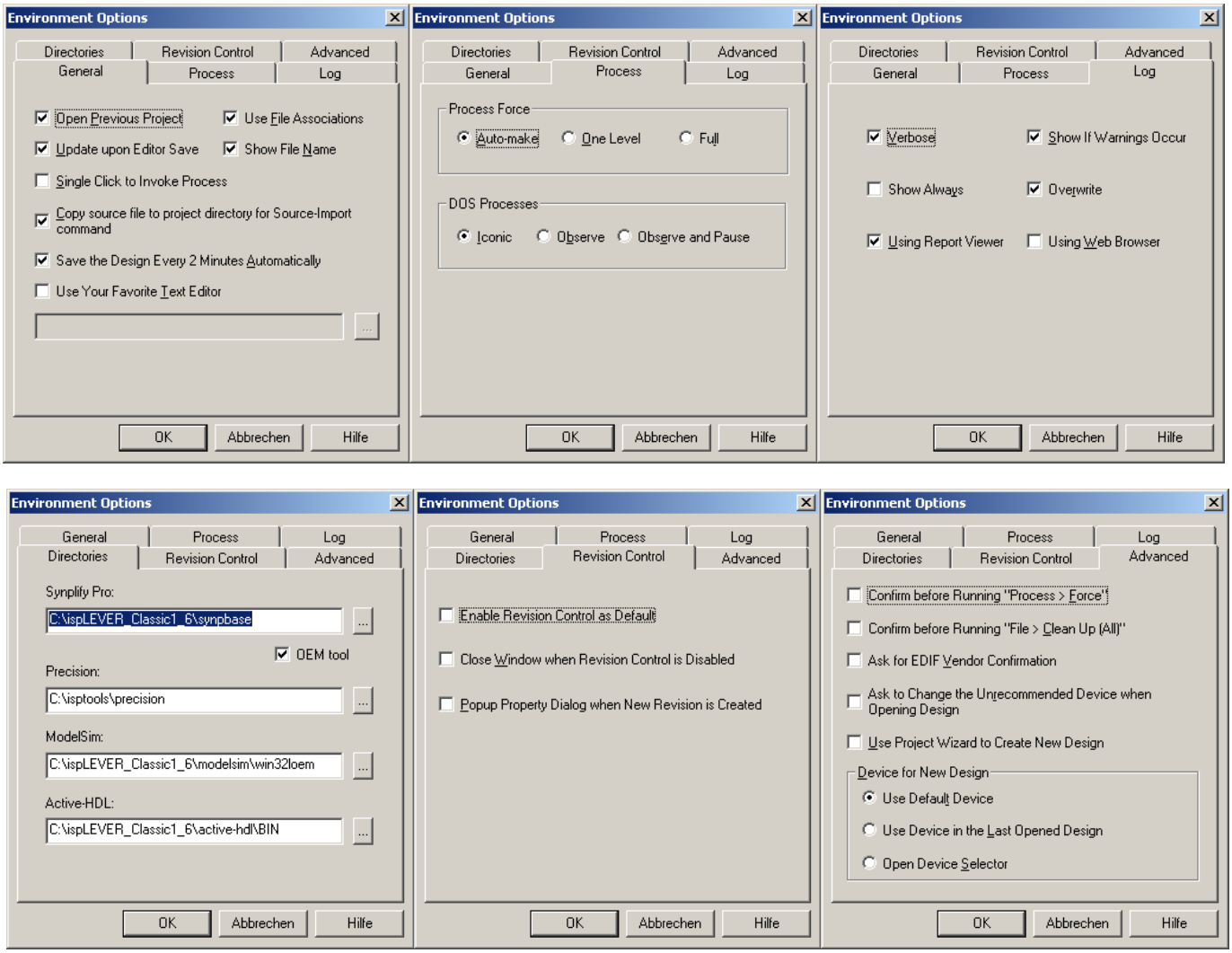
Ach ja, und sollten Sie beim Beenden des Simulators ganz zufällig auf den Editor des Navigators stoßen, in dem die Test Bench auch noch angezeigt wird, so sagen Sie sicher "speichern = nein!", oder?

Viel Spaß

Anhang

Einstellungen im ispLEVER Project Navigator

Menüleiste: Options /Environment ...



Wo finden Sie die Software

Wenn Sie sich an einem PC im Digital-Labor (Raum LI 008b) anmelden, finden Sie im nachfolgenden „Laufwerk:\Verzeichnis“ die Software und Konfigurations-Dateien, zur Installation auf Ihrem eigenen PC um die Versuche auch zu hause vorbereiten können.

W:\IWI-I\DTL-Software\ispLeverClassic_1.6
W:\IWI-I\DTL-Software\Keil3

Mit nachfolgendem Link können Sie die ispLEVER Software kostenlos lizenzieren:
http://www.latticesemi.com/licensing/

```
[Settings]
DefaultProduct = 1.6
DefaultProductName = ispLEVER CLASSIC
URL="http://www.latticesemi.com/news"
;
;DefaultDevice = LC4064ZE-5UMN64C
DefaultDevice = GAL16V8D-20QPI
;Inserted for 10.0 BETA
;DefaultDevice = LFX1200B-05F900C
;
;To change default device, need to change the
;following files:
;   ispsys.ini
;   virtual.sds (DieName, [Generic Devices]
;  section, default FDK)
;   blank.lci (default settings in [Device] and
; [Revision] section

IniFileName=lsc_1_6.ini

[Default Devices]
;GDF = ispGDX160VA-5Q208
GDF = GAL16V8D-20QPI
```

Bild 28: geänderte Datei: ispsys.ini

Die Datei `ispsys.ini` befindet sich bei einer Standart-Installation im Verzeichnis C:\ispLEVER_Classic1_6\ispcpld\config* (siehe Bild 28).

Einstellungen im Active-HDL 9.1 Simulator

Active-HDL 9.1 Simulator aufrufen: ***Start /Programme /Lattice Semiconductor ispLEVER Classic 1.6 /Accessoiries /Active-HDL Lattice Edition***

Nachfolgende Änderungen machen.

Menüleiste: Tools /Preferences ...

Default waveform Viewer/Editor auf Standard Waveform Viewer/Editor einstellen (siehe Bild 29).

Läßt sich nur ändern, wenn der Simulator angehalten wurde.
Menüleiste: Simulation /End Simulation

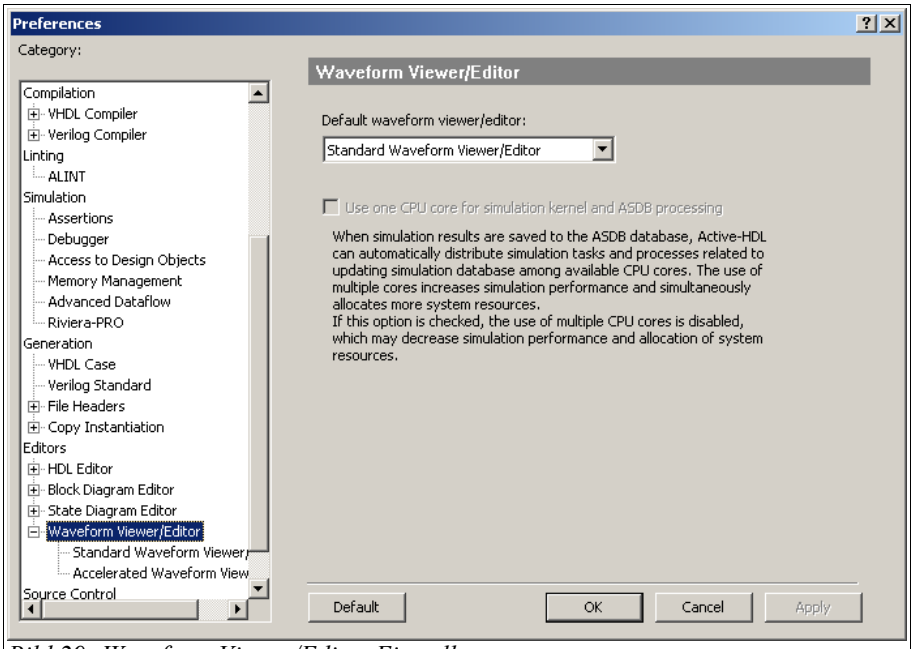


Bild 29: Waveform Viewer/Editor Einstellungen