Das BASTLI CPLD-Demoboard
Anleitung und Dokumentation

Lukas Schrittwieser

30. Mai 2009

Inhaltsverzeichnis

1 Übersicht **3**

2 Hardware **4**
2.0.1 Stromversorgung 4
2.0.2 CPLD 5
2.0.3 LEDs 5
2.0.4 Taster 5
2.0.5 Siegen Segment Anzeigen 6
2.0.6 DAC 7
2.0.7 DIP Schalter 8
2.0.8 Oszillator 8
2.1 Aufbau des Demoboards 9
2.2 Aufbauhinweise 9
2.2.1 Elektrolytkondensatoren 9
2.2.2 Gleichrichter 10
2.2.3 Leuchtdioden 10
2.2.4 Siebensegment-Anzeige 10
2.2.5 Widerstandsnetzwerk 10
2.2.6 ICs 10
2.2.7 Drahtbrücken 10

3 Software **11**
3.1 Download und Installation 11
3.2 Neues Projekt erstellen 11
3.3 Grundlegende Einstellungen 12
3.4 Design Eingabe – Schematic 14
3.5 Design Eingabe – HDL 14
3.6 Übersetzen und Flashen 14

4 Linksammlung **16**

1 Übersicht

Das Bastli CPLD-Demoboard haben wir, analog zum bereits bekannten AVR-Demoboard, entworfen, um Anfängern den Einstieg in die Welt der programmierbaren Logik zu erleichtern. Es bietet neben dem CPLD, der Spannungversorgung und der Programmierschnittstelle auch einige externe Komponenten, um die verschieden Verwendungsmöglichkeiten des CPLDs ausprobieren zu können.

Diese Anleitung beschreibt den Aufbau und die Inbetriebnahme des Demoboardes. Falls Fragen oder Probleme auftauchen, kann der Bastli unter www.bastli.ethz.ch oder direkt während der Öffnungszeiten im Shop kontaktiert werden.

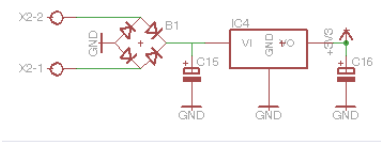
Das Board besitzt folgende Features:

- CPLD:
  - Typ: Xilinx XC9572XL
  - 3.3V Versorgungsspannung
  - 72 FlipFlops
  - 0 bis über 100MHz Taktfrequenz möglich
- 2 Sieben Segment Anzeigen
- 8 Schalter (DIP Switches)
- 5 Taster
- 4 LEDs
- 1 Digital-Analog-Converter, 4 Bit Auflösung, incl. Filter und Klinkenbuchse als Audio-Ausgang
- 3 Quarzgenerierte Taktfrequenzen (250Hz, 16kHz, 256kHz)
- Integrierte Spannungsversorgung. Nötige Speisung: mindestens 7V und 200mA nötig.

2 Hardware

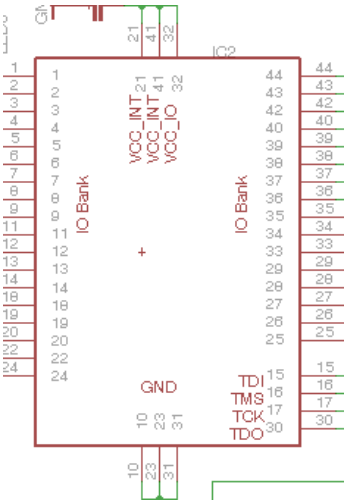
Hier sollen die einzelnen Hardwarekomponenten des Boardes etwas genauer beschrieben werden, der Schaltplan wird in den Erklärungen verwendet und kann auf der Bastli-Homepage heruntergeladen werden.

2.0.1 Stromversorgung



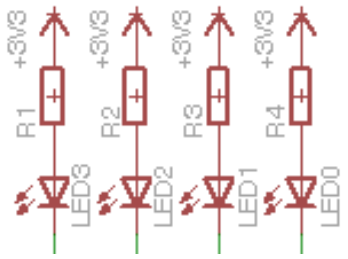
Unten rechts auf dem Demoboard befindet sich die Spannungsversorgung. Die Hauptkomponente ist mit U4 ein so genannter Linearregler, der aus einer Eingangsspannung von 5V oder mehr eine konstante Spannung von 3.3V am Ausgang liefert. Die Kondensatoren C15 und C16 gleichen Stromspitzen aus und sorgen somit für eine saubere 3.3V Spannung. Da der Regler im Linearbetrieb arbeitet, kann er sich etwas erwärmen, was normal ist.

2.0.2 CPLD



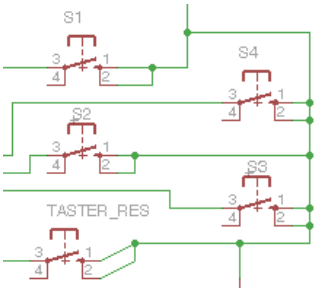
In der Mitte des Schaltplans und des Demoboardes befindet sich der CPLD, wobei er oben und unten seine Spannungsversorgung hat und rechts unten die Schnittstelle zum Programmieren. Die Kondensatoren über dem CPLD sind ebenfalls zum Puffern der Spannungsversorgung, der CPLD zieht seinen Betriebsstrom nämlich hauptsächlich impulsartig (immer dann wenn die Logik schaltet). Die Kondensatoren sorgen dafür, dass auch während eines solchen Strompulses die Spannung nicht einbricht. Daher sind sie so nahe wie möglich am CPLD angebracht.

2.0.3 LEDs



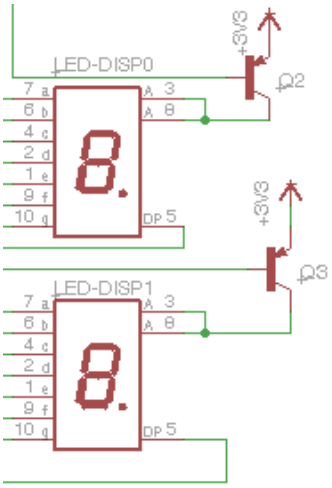
Es sind vier Leds auf dem Board verfügbar, diese leuchten immer dann, wenn die Schaltung im CPLD eine 0, also logisch falsch liefert. Die Widerstände begrenzen den Strom durch die Leds auf etwa 10mA pro LED.

2.0.4 Taster



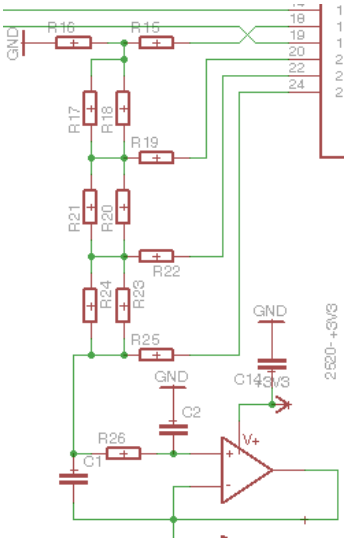
Rechts oben im Schaltplan befinden sich 5 Taster, einer davon bedient die Reset Leitung, die zum zurücksetzen aller FlipFlops im CPLD verwendet werden kann. Die Taster liefern beim Drücken GND und somit einen logische 0 in die Schaltung im CPLD. Sind die Taster nicht gedrückt, so ziehen die Pullwiderstände in RN1 die Eingänge auf Betriebsspannung und somit auf eine logische 1. Damit das Funktioniert müssen allerdings die internen Widerstände des CPLDs abgeschaltet werden, wie das geht steht im Kapitel Software unter Einstellungen. Da die Taster beim drücken kein schauberes Rechtecksignal liefern, sondern erst einige kurze Impulse (so genanntes prellen), sorgen die Kondensatoren C8 bis C12 für eine Glättung.

2.0.5 Siegen Segment Anzeigen



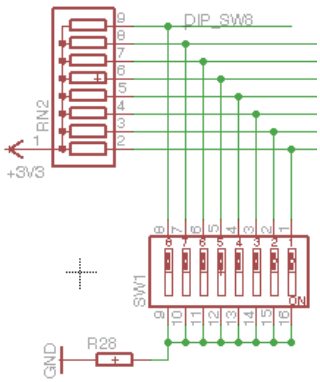
Unterhalb der Taster(im Schaltplan) befinden sich zwei Sieben Segment Anzeigen, auch hier leuchten die LEDs wieder, wenn eine logische Null ausgegeben wird. Allerdings sind die gleichen LEDs der beiden Anzeigen immer verbunden, man sagt die beiden Anzeigen sind gemultiplext. Damit man dennoch zwei unabhängige Zeichen darstellen kann wird immer nur eine der beiden Anzeigen über die Transistoren Q1 und Q2 aktiviert. Schaltet man schnell genug zwischen den beiden Anzeigen um, so erscheint für das Auge eine stehende Anzeige mit zwei Ziffern. Wie man das im CPLD realisieren kann sieht man in den Beispielen 1 und 2. Die vielleicht etwas seltsam anmutende Bezeichnung der LED Anschlüsse mit den Buchstaben a bis g ist für solche Anzeigen üblich, im Datenblatt findet man, welcher Buschstabe zu welchem LED-Segment gehört.

2.0.6 DAC



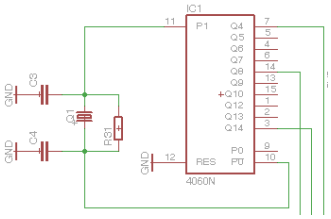
Links unten im Schaltplan befindet sich ein so genannter R2R Digital Analog Converter. Dabei wird mit Hilfe eines Netzwerks aus 12 Widerständen eine Variable Spannung am linken Anschluss von R26 erzeugt. Dabei gibt es folgenden Zusammenhang: sind alle 4 CPLD Ausgänge (18,20,22,24) logisch Null, also auf GND so ist die Ausgangsspannung ebenfalls auf GND. Geht Anschluss 18 auf 1 (und somit auf 3.3V) so ist die Änderung am Ausgang recht gering da die anderen Widerstände diese Spannung herunterteilen. Genau gesagt steigt sie auf $\frac{3.3V}{2^4} = 0.21V$ an. Pin 18 stellt somit das niederwertigste Bit dar, Pin 24 das höchstwertige. Man kann sich das Netzwerk also als eine Reale Spannungsquelle vorstellen. Ihr Innenwiderstand beträgt 5kOhm und die Leerlaufspannung $U \sup 0 = \frac{3.3V*x}{16}$ wobei x die an den 4 Ausgängen angelegte Zahl darstellt. $x = 8*(Pin24)+4*(Pin22)+2*Pin(20)+(Pin19)$ Wer eine genauere Erklärung möchte, kann im Internet nach R2R DAC suchen. C1, C2, R26 und IC3 bilden zum Abschluss noch ein Filter zweiter Ordnung(weil es zwei Cs hat), das Frequenzanteile über 15kHz herausfiltert. Die Abschneidung erfolgt nicht schlagartig bei 15kHz sondern nach und nach. Wer wissen möchte warum und wie dieses Filter funktioniert sie hier auf die Wikipedia Artikel zum Thema Butterworth Filter und Sallen-Key Topology verwiesen. Der untere der beiden OPVs puffert schliesslich das Signal, das über die Audio-Buchse an aktive PC Lautsprecher oder Ohrstöpsel geliefert werden kann.

2.0.7 DIP Schalter



Nebem dem DAC finden sich noch acht Dip Schalter, diese funktionieren gleich wie die Taster, nur sind es eben Schalter. Der Widerstand R28 sorgt dafür, dass wenn ein Schalter-Anschluss des CPLDs versehentlich aus Ausgang deklariert wird und 3.3V liefert, bei gleichzeitig einschaltem Schalter, der Strom begrenzt wird. Da der Schalter ja 0V liefert, wenn er eingeschalten ist. Trozdem kann es aber zur Zerstörung eines Anschlusses oder sogar des ganzen CPLDs kommen, wenn zwei oder mehr Schalter eingeschaltet sind und dann auch mehrere Portpins als Ausgänge verwendet werden. Ihr sollte daher die Schalteranschlüsse des CPLDs immer als Eingänge definieren. Wie das geht steht weiter unten.

2.0.8 Oszillator



Ganz links oben im Schaltplan befindet sich dann noch der Oszillator, wobei hier ein 4.096MHz Quarz zum Schwingen gebracht wird. Anschliessend wird der Takt mit Flip-Flops geteilt, so dass uns mehrere verschiedenen Frequenzen mit 256kHz, 16kHz und 250Hz zur Verfügung stehen. Realisiert wird dies durch einen CMOS Zähler, der intern aus ein paar FlipFlops aufgebaut ist. Der Widerstand R31 sorgt mit C3 und C4 dafür, dass der Quarz zu schwingen beginnt. Mit einem Oszilloskop könnte man die verschiedenen Teilerstufen des FlipFlops gut beobachten.

2.1 Aufbau des Demoboards

Die Platine des Demoboardes ist bereits industriell vorgefertigt. Ihr müsst lediglich die Bauteile mit der Platine verlöten.
Beim Board wurde Wert auf einfache Lötbarkeit gelegt. Der Aufbau sollte deshalb keine grossen Schwierigkeiten bereiten. Bei den Links finden sich auch Seiten mit Tipps zum Löten.
Für den Aufbau benötigt man einen Elektronik-Lötkolben mit ca. 40-80W Heizleistung (Dachrinnen-Löt-Monster mit 200W eignen sich nicht!...), Lötzinn, Seitenschneider und sonstiges Standard-Werkzeug. Ansonsten kann der Bastli gerne für Tipps, Material(Messplatz) und Support kontaktiert werden.

Für den Aufbau bewährt es sich, den Schaltplan und das Layout zur Hand zu haben, damit man sieht, welches Bauteil wo und wie plaziert werden muss. Auf der Homepage des Bastli finden sich ausserdem Fotos der aufgebauten Platinen, welche ebenfalls als Referenz verwendet werden können.
Die Bauteile werden durch die Platine gesteckt und auf der Unterseite verlötet. Die ICs werden gesockelt, damit man sie bei einem allfälligen Defekt einfach austauschen kann.
Die Bauteile sind alle relativ robust, eine Zerstörung durch zu langes ,,braten" an den Bauteilen mit dem Lötkolben ist nicht sehr wahrscheinlich. Dennoch sollte die Löttemperatur nicht zu hoch (ca. 300-350°C) gewählt werden.

2.2 Aufbauhinweise

Beginnt am Besten mit den kleineren, nicht sehr hohen Bauteilen wie den Widerständen, Drahtbrücken oder Tastern. Zuletzt lötet man die hohen, klobigen Bauteile wie den Spannungsregler, die Elektrolytkondensatoren oder den Wannenstecker ein.
Die Bauteile werden gemäss Layout/Bestückungsdruck/Photos in die Platine gesteckt und auf der Unterseite verlötet. Die Seite der Platine mit dem Bestückungsdruck ist die Oberseite, hier kommen die Bauteile hin und werden auf der anderen Platinenseite verlötet.

Es folgen nun einige spezielle Aufbauhinweise, da gewisse Bauteile mit einer vorgegebenen Orientierung eingebaut werden müssen.

2.2.1 Elektrolytkondensatoren

Die Elkos haben eine Polung, sie können nicht beliebig eingelötet werden. Beachtet das Layout und den Bestückungsdruck. Die Seite des Elkos mit dem weissen Streifen ist der negative Anschluss und muss ein kleineres Potential als der andere

Pin führen. Ansonsten wird der Kondensator beschädigt und kann platzen. Auf dem Bestückungsdruck hat es ein kleines „+". An diese Seite muss also diejenige Seite des Elkos, die keinen weissen Streifen hat.

2.2.2 Gleichrichter

Beim Gleichrichter ist auf die Einbaurichtung zu achten. Der Bestückungsdruck und die Fotos auf der Homepage sollten hierbei helfen.

2.2.3 Leuchtdioden

Die LEDs haben ebenfalls eine Polung und können nicht beliebig eingelötet werden. Der kurze Anschlussdraht der LED ist die Kathode. (Eselsbrücke: **k**urz für **K**athode). Die Kathoden der LEDs sind direkt mit den CPLD-Pins verbunden, die Kathoden zeigen also in Richtung CPLD. Die Anoden hängen über Widerstände an VCC (positive 3.3V-Versorgung).

2.2.4 Siebensegment-Anzeige

Der Punkt der Anzeige muss unten rechts und nicht oben links sein.

2.2.5 Widerstandsnetzwerk

Der gemeinsame Pin, welcher mit einem kleinen Punkt gekennzeichnet ist, muss jeweils auf der rechten, markierten, Seite sein.

2.2.6 ICs

Die Pins des TS912 und des HCF4060 müssen, bevor man sie in den Sockel stecken kann, etwas nach innen gebogen werden. Ansonst passen die ICs nicht in den Sockel. Dies macht man am Besten, indem man das IC mit der Breitseite auf den Tisch stellt und dann gleichzeitig alle Pins einer Seite etwas nach innen drückt, indem man das IC nach unten auf den Tisch drückt. Der CPLD kann direkt in seinen Sockel gesteckt werden.

2.2.7 Drahtbrücken

Auf der Platine müssen 3 Drahtbrücken eingelötet werden. Dazu verwendet man am Besten ein abgeknipstes Stück Draht von einem Widerstand o.Ä., biegt es zurecht, sodass es in die Bohrungen passt und verlötet es.

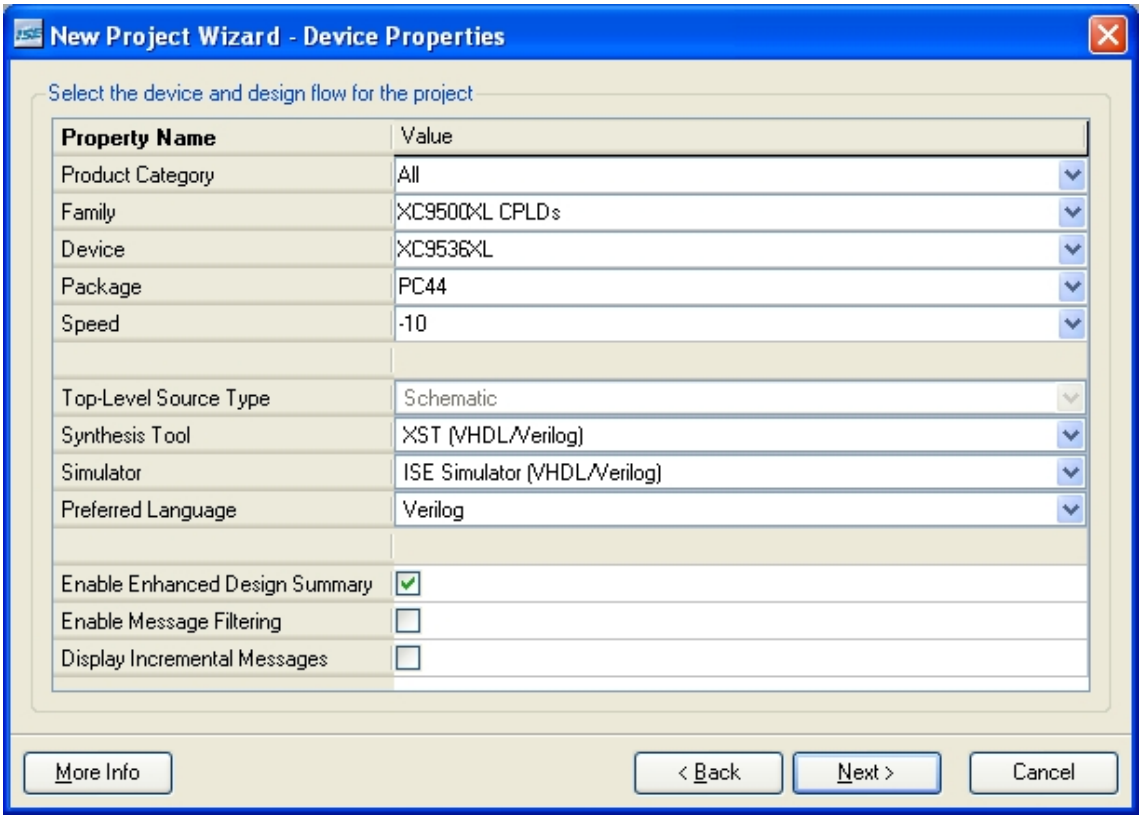
3 Software

3.1 Download und Installation

Die benötigte Software stammt wie der CPLD von Xilinx und nennt sich Free ISE Web Pack. Nach einer kostenfreien Registration unter www.xilinx.com kann man diese gebührenfrei herunterladen und unbeschränkt verwenden. Sie Funktioniert für so gut wie alle gängigen Betriebssysteme. Damit kann man alle Chips von Xilinx, sowohl CPLDs als auch FPGAs, programmieren. Die zusätzlichen Trial Versionen von verschiedenen anderen Xilinx Produkten braucht Ihr nicht unbedingt. Auch die Unterstützung für FPGAs kann man im Installationsassistenen deaktivieren. Achtung: die Software wird erst während der Installation aus dem Web geladen, das Paket ist etwa 4GB gross, es braucht daher schon einige Zeit zum Installieren. Nach der Installation sollte man unbedingt auch noch das vorgeschlagene Update durchführen.

3.2 Neues Projekt erstellen

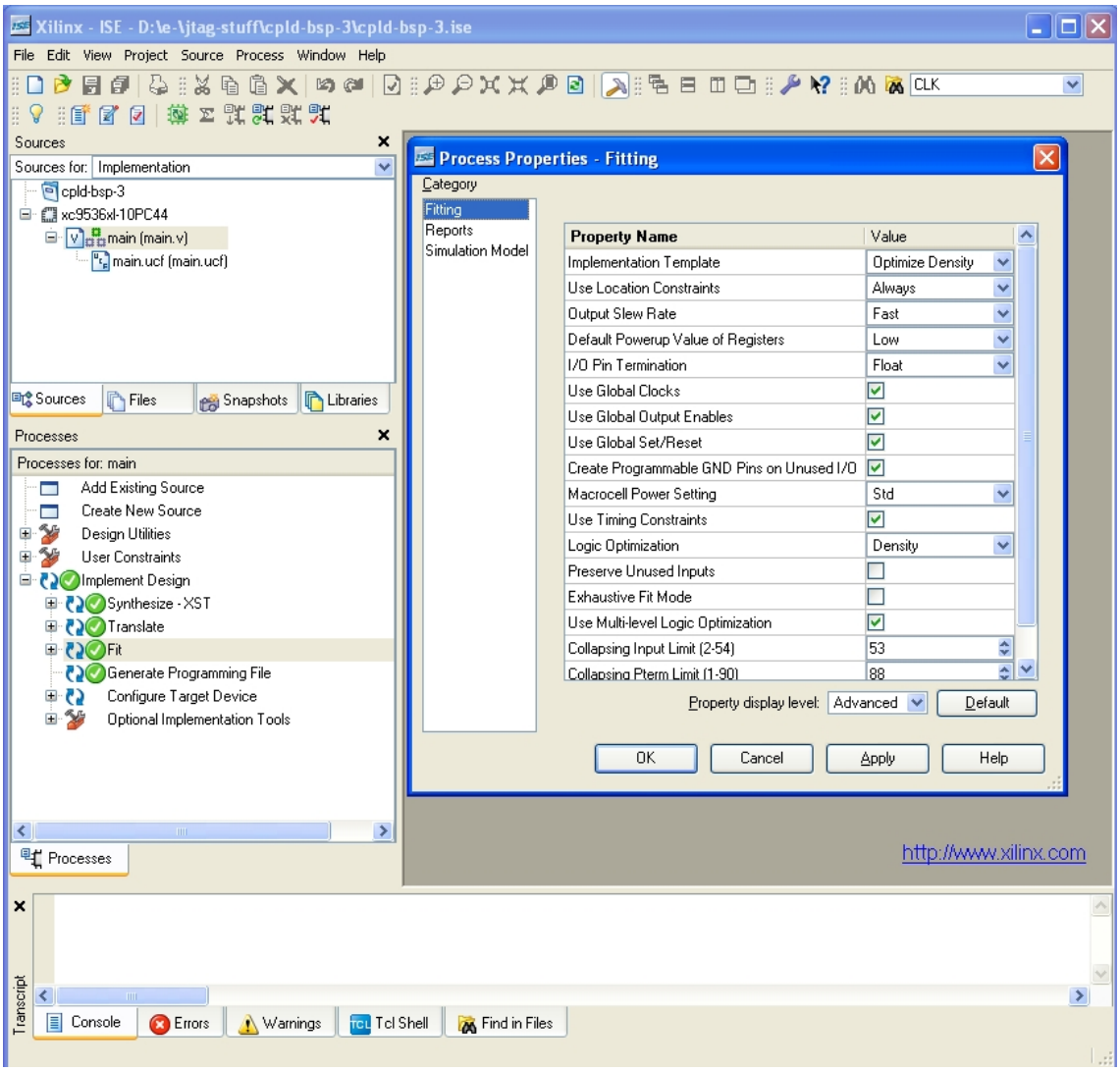
File / New Project startet einen Assistenten, im ersten Fenster gibt man einen Speicherort an. (Sinnvollerweise macht man für jedes Projekt einen neuen Ordner.) Ausserdem wählt man hier, ob man mit einem Schaltplan oder einer Text basierten Schaltungsbeschreibung (VHDL oder Verilog) beginnen möchte. Wir wählen hier Schematic also Schaltplan. Im zweiten Schritt muss eingestellt werden, welchen CPLD wir verwenden. Es ist hier ganz wichtig, das richtige Bauteil auszuwählen, es lässt sich sonst später nicht programmieren! Die Einstellung hier sollte man unbedingt zwei mal kontrollieren. Im dritten Schritt kann man neue Schaltpläne anlegen lassen, im vierten kann man schon vorhandene Dateien zum Projekt hinzufügen. Beides geht auch später noch. Zum Schluss wird noch eine Zusammenfassung angezeigt und mit einem Klick auf Finish wird das Projekt erzeugt. Vor sich hat man nun den leeren Schaltplan der im Schritt 3 erstellt wurde.



3.3 Grundlegende Einstellungen

Nachdem ein neues Projekt erstellt wurde, sollte man einige Dinge einstellen, die die Xilinx Software kennen muss um das Projekt richtig zu überstezen.
Man wählt dazu in der linken Spalte Source und klickt dort das Toplevel Modul an. Das ist die Quelldatei (Schematic oder HDL), die die Ein- und Ausgänge des CPLDs als Ports beinhaltet. (Am besten eines der Demo Projekte ansehen, dann sollte es klar werden, was gemeint ist.) Direkt darunter wählt man dann die Spalte Processes und öffnet Implement Design rechts Klick auf Generate programming file / properties. Es öffnet sich ein neues Fenster. Dort muss ,,Create IEEE1532 Configuration File" aktiviert werden. Wir brauchen nämlich dieses Dateiformat später, um unseren CPLD zu programmieren. Rechtsklick auf Fit / Properties / I/O Pin Termination / Float. diese Einstellung ist notwendig, damit die Taster- und Schaltereingänge bei unserem Board funktionieren. Wer wissen will, was hier geschieht, kann im Datenblatt des CPLD nachlesen, Stichwort keeper. Ausserdem kann man hier noch auswählen, ob man auf ein möglichst schnelles Design (bei uns nicht sinnvoll) oder auf möglichst hohe Logikdichte optimieren möchte. Letzteres ist für uns praktikabler. Zuletzt sollte man auch gleich das so genannte UCF hinzufügen. UCF steht für user constraint file und dient dazu, weitere Informationen über Hardwaredetails an die Software zu liefern. Vor allem geht es darum, der

Software zu sagen, an welchen Pins des CPLDs welche Funktionen angeschlossen sind. Am besten kopiert man das .ucf einfach von einem der Demo-Projekte in den Projektordner und fügt es mit Project / Add Source hinzu. In den .ucf aus den Beispielen sind alle Pins definiert, jedoch zum teil auskommentiert weil sie nicht verwendet werden. Entsprechend den eigenen Bedürfnissen muss man diese natürlich anpassen. Man kann das .ucf ganz normal wie jede Text-Datei bearbeiten. ACHTUNG: ihr solltet unter keinen Umständen die Pin Nummern ändern bzw. genau überlegen was ihr tut. Unter Umständen kann man sonst den CPLD zerstören, wenn man einen Pin, der ein Eingang sein sollte, aus Ausgang deklariert (dann gibt es einen Kurzschluss).



3.4 Design Eingabe – Schematic

Hat man vor sich den Schaltplan, so findet sich links eine Registerkarte Symbols. In dieser sind alle vorhandenen Schaltungselemente wie Gatter, FlipFlops, usw. aufgelistet. Man kann sie einfach auswählen und im Schaltplan plazieren. Bei komplizierten Schaltungen lohnt es sich, einen Teil des Designs in ein eigenes Modul (=neues Schaltsymbol) auszulagern das dann einen eigenen Schaltplan besitzt. Dazu gibt es unter Tools / Symbol Wizard den entsprechenden Assistenten. Das Verbinden der Schaltelemente macht man mit Add / Wire. Dabei sollte man ein paar Dinge beachten: pro Leitung darf es nur einen Ausgang geben, aber natürlich mehrere Eingänge. Hat man einen Bus(erkennt man daran das die Linie dicker ist), so kann man daraus auch einzelne Leitungen abzweigen. Dazu plaziert man mit Add / Bus Tap ein kleines Symbol, von dort geht dann eine einzelne Leitung weiter. Die Zuordnung, welche Leitung aus dem Bus mit dem Tap abzweigt, geschieht über den Namen. Mit Add / Net Name werden Leitungen benannt, wobei alle gleich benannten Leitungen verbunden werden, auch wenn sie nicht durch Linien verbunden sind. Alle Leitungen, die man nach aussen führen möchte müssen mit einem so genannten I/O Marker versehen werden. Die Namen dieser Leitungen (stehen auch im I/O Marker) müssen mit den Namen im .ucf File übereinstimmen. Das sind die wichtigsten Funktionen des Schaltplaneditors, aber bei weitem nicht alle. Wer mehr braucht sei auf die Dokumentation von Xilinx verwiesen, es gibt eine Umfassende Hilfe im Internet.

3.5 Design Eingabe – HDL

Alternativ zum Schaltplan kann man auch eine Hardware Descripition Language verwenden. Was man lieber hat hängt von individuellen Vorlieben ab, im Allgemeinen ist man mit HDLs aber schneller. Prinzipiell kann man die Sprachen Verilog, VHDL und ABEL verwenden. Sie lassen sich innerhalb eines Projektes untereinander als auch mit Schalplänen mischen. Aus Platzgründen geben wir hier keine weitere Einführung, es gibt jedoch ein Beispielprojekt, das vollständig in Verilog ausgeführt ist. Ansonsten lohnt es sich, Google zu bemühen, im Netz gibt es massenweise Tutorials. Sehr hilfreich ist auch die Funktion Edit / Language Templates, dort findet man alle wichtigen Konstrukte der unterstützten HDLs zusammengefasst, seht es euch doch einmal an.

3.6 Übersetzen und Flashen

Nach man sein Design eingegeben hat, kann man wieder Sources und Processes auswählen. Mit einem Doppelklick auf ,,Generate Programming File" startet man den Synthetisierer. Das durchlaufen eines Designs dauert je nach Rechner bis zu etwa 2 Minuten. Im Fenster ganz unten kann man beobachten, was bei den einzelnen Schritten geschieht, der Compiler erzählt einem hier so einiges.

Prinzipiell interessieren und diese Ausgaben nicht, man sollte aber unbedingt alle Warnungen und Fehlermeldungen lesen. Diese beinhalten auch immer einen Link zur Xilinx Website wo es Erklärungen dazu gibt. Wenn alles gut gegangen ist und die orangenen Fragezeichen durch grüne Häkchen ersetzt wurden, gibt es im Projektordner eine Datei mit der Endung .isc. Genau diese brauchen wir jetzt, weil damit der CPLD programmiert wird. Dazu wird das Programmierkabel USBASP 2.0 an den PC angeschlossen (bitte gesondertes Manual beachten), mit dem CPLD board verbunden und dieses mit Strom versorgt. Dann öffnet man sich eine Konsole (WinXP: Start / Ausführen / cmd.exe) und startet dort das Programm XC9500prog.exe, welches ihr auf der Bastli-Homepage zum Download findet, wobei man als Parameter noch den Pfad zum .isc File angeben muss. Beispiel:
d:\e-\jtag\XC9500prog.exe d:\e-\jtag\bsp1\main.isc
Das Programm löscht eine eventuell vorhandene alte Konfiguration aus dem CPLD und brennt die neue hinein. Diese ist dann bis zum nächsten Flashen des CPLDs gespeichert. Laut Hersteller lässt sich der CPLD mindestens 10000 mal neu beschreiben, wenn ihr dennoch einen neuen braucht wendet euch an den BASTLI ;-)

Nach einem Programmiervorgang muss der CPLD einmal aus- und wieder eingeschaltet werden, damit die Logik loslegen kann. Dazu einfach das Demoboard ab- und wieder anschalten.

4 Linksammlung

Die folgenden Links bieten zusätzliche Infos zum Löten und Programmieren:

Bastli Homepage: http://www.bastli.ethz.ch
Website von Xilinx: http://www.xilinx.com
Tutorial zum Löten: http://www.roboternetz.de/wissen/index.php/L%C3%B6t-Tutorial