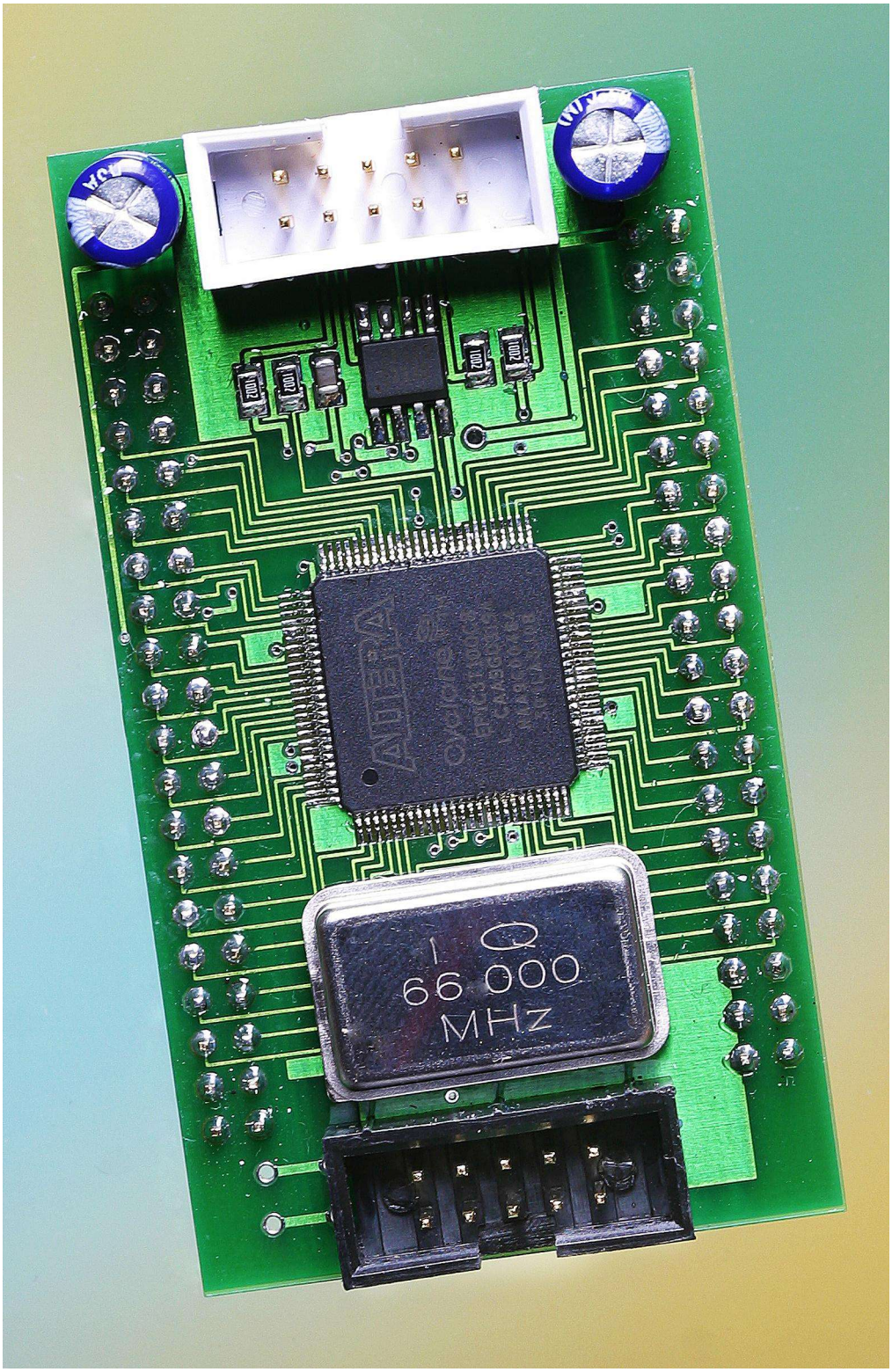


Sphinx-C100 Benutzerdokumentation

Inventronik GmbH
Finkenstraße 48, D-70199 Stuttgart, http://www.inventronik.de

Diese Seite ist absichtlich leer.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

History

Issue	Date	Reason For Changes
01	20040119	Initial issue
02	20040413	Minor bugfixes

References

[1]

Einführung

Sehr geehrte Damen und Herren,

sie haben mit diesem Produkt ein hochmodernes FPGA-Modul erworben, das es Ihnen ermöglicht, die Entwicklungszyklen für Ihre modernen digitalen Logikschaltungen zu optimieren.

Bitte lesen sie diese Benutzerdokumentation vollständig durch, bevor sie das Modul in Betrieb nehmen. Auf unseren ständig aktualisierten Internetseiten (http://www.inventronik.de) finden sie immer die neuesten Dokumente, Tipps und Tricks die Ihnen bei der Lösung Ihrer Aufgaben behilflich sein können.

Sicherheitshinweise

Die Firma Inventronik weist ausdrücklich darauf hin, dass dieses Produkt den heutzutage üblichen hohen Qualitätsansprüchen entspricht. Dennoch können wir keine Garantie auf Fehlerfreiheit übernehmen. Benutzen sie dieses Produkt, in Applikationen, in denen bei Defekten unmittelbare Gefahren für den Menschen entstehen wie beispielsweise medizinische Systeme, Schutzeinrichtungen oder ähnliches nur in redundanten Systemen.

Die Firma Inventronik übernimmt keine Haftung für entstehende Schäden bzw. Folgeschäden.

Dieses Produkt ist ausschließlich für den Gebrauch an einer Versorgungsspannung von 5V konzipiert. Höhere Spannungen können zu Funktionsstörungen bzw. zum Ausfall der Module führen oder sogar eine Gefahr für das menschliche Leben beinhalten. Benutzen Sie zur Versorgung der Module daher nur zugelassene 5V Spannungsquellen.

Lieferumfang

Im Lieferumfang dieses Produktes sind folgende Artikel enthalten:

- 1 FPGA-Modul Sphinx-C100
- 1 Produkt-CD mit Datenblättern, Dokumentation und Software

Hardware Benutzerdokumentation

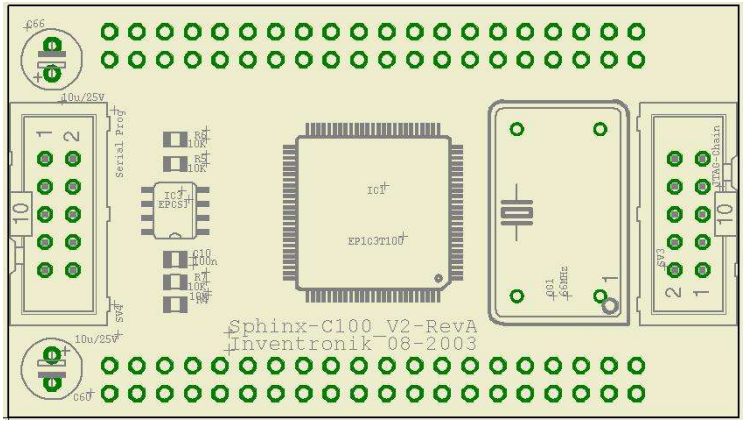


Abbildung 1: Modulansicht des Sphinx-C100 (Ansicht von oben)

Beschreibung des Moduls

Die Module Sphinx-C100 sind ein modernes, neues Produkt von Inventronik. Sie sind nach dem „ready for use“ Konzept entworfen und ermöglichem daher dem Systementwickler den schnellen Einstieg in die Thematik der abstrakten Modellierung digitaler Schaltungen. Zur Inbetriebnahme wird lediglich eine 5V Spannungsversorgung, ein Byte Blaster II Kabel sowie die frei erhältliche Software Quartus II Web Edition von Altera benötigt.

Die Module basieren auf dem **F**ield **P**rogrammable **G**ate **A**rray (FPGA) „Cyclone EP1C3T100“ der Firma Altera. Obwohl das Modul gerade einmal 38mm x 66mm gross ist, sind alle notwendigen Komponenten zum Betrieb des FPGA vorhanden. Die Spannungsversorgung der I/O-Zellen sowie des FPGA-Cores erfolgt „on board“, so dass Sphinx-C100 mit einer einzigen Spannung von 5V auskommt. Für Boundary Scan Tests ist eine JTAG-Schnittstelle vorhanden, während der auf dem Modul befindliche Konfigurationsbaustein über eine zweite Schnittstelle programmiert wird. Ein Taktgenerator in unmittelbarer Nähe des Cyclone sorgt für eine stabile Frequenz. Die 65 digitalen Ein-bzw. Ausgänge des FPGA sind auf Pfostenleisten aufgelegt. Der EP1C3T100 verfügt über 2910 Logical Elements, 59904 Byte RAM, eine PLL und unterstützt unter anderem DDR.

Das Modul eignet sich für allgemeine Steuer- und Regelungsapplikationen wie zum Beispiel:

- Schnelle Verriegelungsmechanismen (Interlocks)
- „System on a silicon chip“ (komplette digitale Systeme inclusive MCUs)
- Digitale Filterschaltungen
- Digitale Signalverarbeitung
- Codeumsetzer, Komponentenersatz, Zustands-Folgesteuerungen usw.

Vorbereitende Modulkonfiguration

Für die Inbetriebnahme müssen am Sphinx-C100 keine Einstellungen vorgenommen werden. Der Baustein ist lediglich mit einer Spannung von 5V zu versorgen. Die Belegung der Anschlüsse des Moduls entnehmen Sie bitte Tabelle 3 oder der Abbildung 2. Die Stromaufnahme des Moduls hängt im wesentlichen von der angeschlossenen Peripherie ab und ist daher vom Schaltungsentwickler individuell zu ermitteln. Die von Inventronik gelieferten Netzgeräte liefern einen Maximalstrom von 1A bei 5V. Die Module sind durch eine Verpolschutzdiode in der Versorgungsleitung gegen Verpolung geschützt.

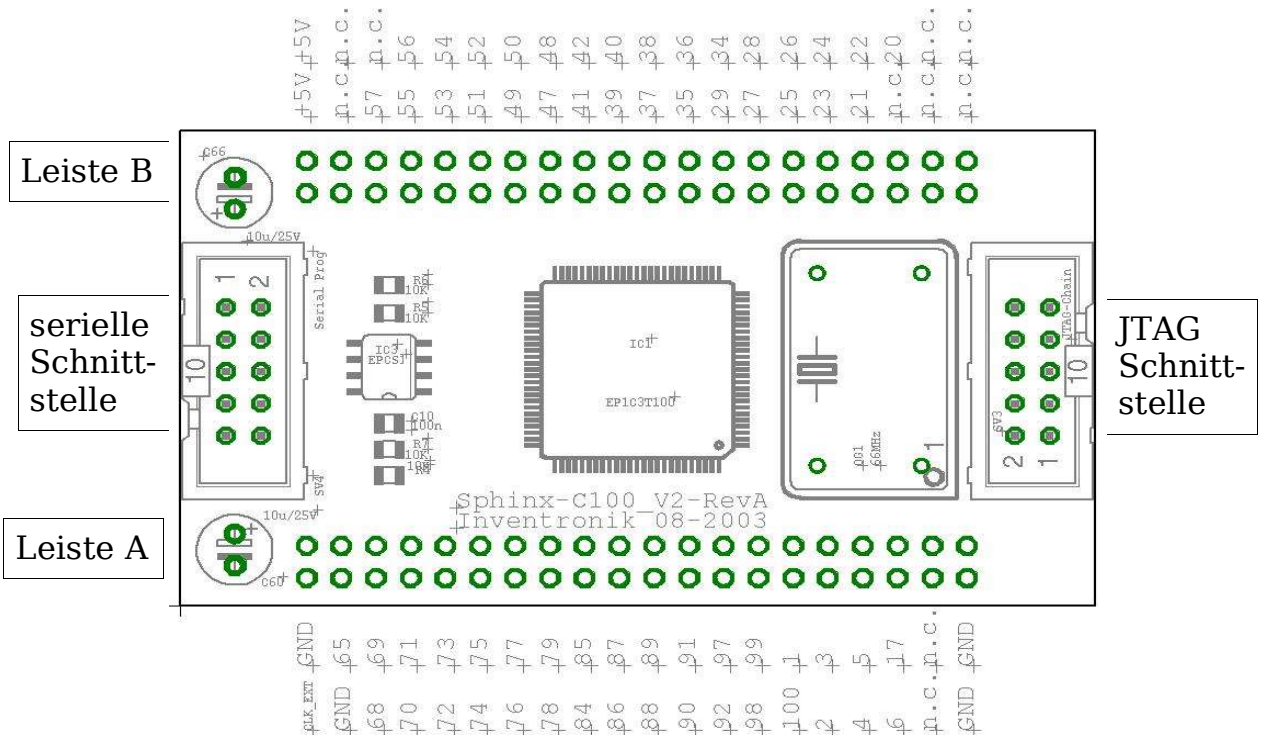


Abbildung 2: Sphinx-C100 Anschlussbelegung

Auf dem Modul Sphinx-C100 befindet sich ein Quarzoszillator. Die Frequenz dieses Oszillators liegt zwischen 8 MHz und 100 MHz, je nach Bestell-Option. Wird dennoch eine andere Frequenz zum Betrieb der digitalen Schaltung benötigt, so lässt sich diese über Pin 2 (CLK_EXT) des Moduls zuführen. Die Umschaltung vom internen zum externen Taktgenerator erfolgt durch die Wahl des Pinouts in der Entwicklungsumgebung. CLK_EXT liegt auf Pin 66 wohingegen der moduleigene Oszillator an Pin 10 des FPGA angeschlossen ist.

Anschluss des Sphinx-C100

Nachdem die Spannungsversorgung von 5V an das Modul angeschlossen ist (siehe vorangehenden Abschnitt), wird noch eine Verbindung zum Entwicklungs-PC benötigt. Diese Verbindung wird durch ein Byte-Blaster II Kabel der Firma Altera hergestellt. Sie haben hierbei die Möglichkeit via JTAG- (**J**oint **T**est **A**ction **G**roup) oder ISP-Schnittstelle zu arbeiten. Im folgenden werden die beiden Varianten gegenübergestellt. Für detailliertere Informationen sei auf das Cyclone Device Handbuch von Altera verwiesen, welches sich auf der mitgelieferten Produkt-CD befindet oder als Download zur Verfügung steht.

Der FPGA Cyclone kann die programmierte Logik nicht permanent speichern (der FPGA verfügt lediglich über ein flüchtiges SRAM) und muss daher beim Systemstart die Konfigurationsinformation aus einem Flash-ROM basierten (nichtflüchtiger Speicher) Konfigurationsbaustein laden. Dieser Baustein ist mit dem FPGA verbunden und verfügt zudem über eine serielle Programmierschnittstelle, die an einem Steckverbinder des Sphinx-C100 herausgeführt ist. Der Entwicklungs-PC überträgt über diese Schnittstelle die zuvor erstellte Konfigurationsdatei. Das Laden der Konfiguration vom Konfigurationsbaustein in den FPGA erfolgt automatisch beim Anlegen der Versorgungsspannung.

Der SRAM Speicher des FPGAs kann auch direkt mit dem Konfigurationsfile aus dem Entwicklungs-PC geladen werden. Hierzu steht die JTAG Schnittstelle zur Verfügung, die ebenfalls über einen Steckverbinder des Sphinx-C100 herausgeführt ist. Die so aufgespielte Konfiguration des FPGA bleibt, wie bereits erwähnt, beim Abschalten der Versorgungsspannung nicht erhalten. Der Vorteil in dieser Programmiermethode liegt in den deutlich verkürzten Ladezeiten. Gerade bei häufigem Wechsel der Konfigurationsdatei (z.B. beim Debugging) ist diese Methode vorzuziehen.

Beide Anschlüsse sind als 10 polige Wannenstecker ausgeführt, die einen direkten Anschluss des Byte Blaster II ermöglichen. Verbinden Sie hierzu das Byte Blaster II-Kabel mit der entsprechenden Schnittstelle des Sphinx-C100 abhängig davon, ob JTAG oder serielle Programmierung gewünscht wird (siehe auch Abbildung 2). Der Byte Blaster II wird PC-seitig an der Parallelschnittstelle betrieben.

JTAG Schnittstelle

Die JTAG Schnittstelle wurde ursprünglich für Testzwecke entworfen. Hierüber lassen sich sogenannte Boundary Scan Tests durchführen (siehe hierzu Altera Application Notes und einschlägige Literatur). Das Laden der Konfigurationsdatei unter Zuhilfenahme eines Altera Byte Blaster II Kabels wird von der Designsoftware Quartus II unterstützt (siehe unten). Die Belegung des JTAG-Steckverbinders auf dem Sphinx-C100 ist in der folgenden Tabelle wiedergegeben:

Stecker-Pin	Funktion	Bemer-kung	Stecker-Pin	Funktion	Bemer-kung
1	TCK	Clock	6	VCCIO	3,3V
2	GND	0V	7	n.c.	not used
3	TDO	Out	8	n.c.	not used
4	VCCIO	3,3V	9	TDI	in
5	TMS	Mode	10	GND	0V

Tabelle 1: Belegung des JTAG Programmiersteckers

Serielle Programmierschnittstelle

Im Gegensatz zur JTAG- dient die serielle Schnittstelle ausschließlich zur Programmierung des Konfigurationsbausteines. Die Steckerbelegung ist in Tabelle 2 wiedergegeben.

Stecker-Pin	Funktion	Bemer-kung	Stecker-Pin	Funktion	Bemer-kung
1	DCLK	Clock	6	NCE	
2	GND	0V	7	DATA0	Data
3	CONF_DONE		8	/CSO	Chipsel.
4	VCCIO	3,3V	9	ASDI	
5	NCONFIG		10	GND	0V

Tabelle 2: Belegung des seriellen Programmiersteckers

Stiftleisten-Belegungen

Das Modul Sphinx-C100 verfügt über zwei 40 polige Stiftleisten an die alle frei wählbaren Eingangs-/Ausgangspins sowie einige spezielle Pins des Cyclone Bausteines angeschlossen sind. Dem Schaltungsentwickler stehen bei Verwendung des Sphinx-C100 somit 65 frei programmierbare Pins (Eingänge oder Ausgänge) zur Verfügung. Die gerichteten Eingänge haben alternative Funktionen und können, je nach Konfiguration nicht verwendet werden. An dieser Stelle sei auf das Cyclone Benutzerhandbuch verwiesen. Die Orientierung der Stiftleisten ist in Abbildung 2 wiedergegeben. Ferner ist in dieser Abbildung ebenfalls eine Übersicht über die Belegung der einzelnen Pins ersichtlich.

Pin-Nummer	Funktion / alternative Funktion	Alternative Funktion	Bemerkungen
1 (Leiste A)	GND		0V
2 (Leiste A)	CLK_EXT		Externer Takt
3 (Leiste A)	I/O: Cyclone-Pin 65		general purpose I/O
4 (Leiste A)	GND		0V
5 (Leiste A)	I/O: Cyclone-Pin 69		general purpose I/O
6 (Leiste A)	I/O: Cyclone-Pin 68	VREF1B3	I/O-Zellen-Referenz
7 (Leiste A)	I/O: Cyclone-Pin 71		general purpose I/O
8 (Leiste A)	I/O: Cyclone-Pin 70		general purpose I/O
9 (Leiste A)	I/O: Cyclone-Pin 73	VREF0B3	I/O-Zellen-Referenz
10 (Leiste A)	I/O: Cyclone-Pin 72	DPCLK4	Dual Purpose CLK
11 (Leiste A)	I/O: Cyclone-Pin 75		general purpose I/O
12 (Leiste A)	I/O: Cyclone-Pin 74		general purpose I/O
13 (Leiste A)	I/O: Cyclone-Pin 77		general purpose I/O
14 (Leiste A)	I/O: Cyclone-Pin 76		general purpose I/O
15 (Leiste A)	I/O: Cyclone-Pin 79		general purpose I/O
16 (Leiste A)	I/O: Cyclone-Pin 78		general purpose I/O
17 (Leiste A)	I/O: Cyclone-Pin 85	VREF0B2	I/O-Zellen-Referenz
18 (Leiste A)	I/O: Cyclone-Pin 84		general purpose I/O
19 (Leiste A)	I/O: Cyclone-Pin 87		general purpose I/O
20 (Leiste A)	I/O: Cyclone-Pin 86		general purpose I/O
21 (Leiste A)	I/O: Cyclone-Pin 89		general purpose I/O
22 (Leiste A)	I/O: Cyclone-Pin 88	VREF1B2	I/O-Zellen-Referenz
23 (Leiste A)	I/O: Cyclone-Pin 91	VREF2B2	I/O-Zellen-Referenz
24 (Leiste A)	I/O: Cyclone-Pin 90		general purpose I/O
25 (Leiste A)	I/O: Cyclone-Pin 97		general purpose I/O
26 (Leiste A)	I/O: Cyclone-Pin 92	DPCLK2	Dual Purpose CLK
27 (Leiste A)	I/O: Cyclone-Pin 99		general purpose I/O
28 (Leiste A)	I/O: Cyclone-Pin 98		general purpose I/O
29 (Leiste A)	I/O: Cyclone-Pin 1		general purpose I/O
30 (Leiste A)	I/O: Cyclone-Pin 100		general purpose I/O
31 (Leiste A)	I/O: Cyclone-Pin 3		general purpose I/O
32 (Leiste A)	I/O: Cyclone-Pin 2		general purpose I/O
33 (Leiste A)	I/O: Cyclone-Pin 5	VREF1B1	I/O-Zellen-Referenz
34 (Leiste A)	I/O: Cyclone-Pin 4	VREF0B1	I/O-Zellen-Referenz

Pin-Nummer	Funktion / alternative Funktion	Alternative Funktion	Bemerkungen
35 (Leiste A)	I/O: Cyclone-Pin 17	ASDO	Serial data output
36 (Leiste A)	/CSO		Chip select output
37 (Leiste A)	not connected		not used
38 (Leiste A)	not connected		not used
39 (Leiste A)	GND		0V
40 (Leiste A)	GND		0V
1 (Leiste B)	+5V		Supply
2 (Leiste B)	+5V		Supply
3 (Leiste B)	not connected		not used
4 (Leiste B)	not connected		not used
5 (Leiste B)	not connected		not used
6 (Leiste B)	I/O: Cyclone-Pin 57	VREF2B3	I/O-Zellen-Referenz
7 (Leiste B)	I/O: Cyclone-Pin 56		general purpose I/O
8 (Leiste B)	I/O: Cyclone-Pin 55		general purpose I/O
9 (Leiste B)	I/O: Cyclone-Pin 54		general purpose I/O
10 (Leiste B)	I/O: Cyclone-Pin 53		general purpose I/O
11 (Leiste B)	I/O: Cyclone-Pin 52		general purpose I/O
12 (Leiste B)	I/O: Cyclone-Pin 51		general purpose I/O
13 (Leiste B)	I/O: Cyclone-Pin 50		general purpose I/O
14 (Leiste B)	I/O: Cyclone-Pin 49		general purpose I/O
15 (Leiste B)	I/O: Cyclone-Pin 48		general purpose I/O
16 (Leiste B)	I/O: Cyclone-Pin 47		general purpose I/O
17 (Leiste B)	I/O: Cyclone-Pin 42	DPCLK6	Dual Purpose CLK
18 (Leiste B)	I/O: Cyclone-Pin 41	VREF0B4	I/O-Zellen-Referenz
19 (Leiste B)	I/O: Cyclone-Pin 40		general purpose I/O
20 (Leiste B)	I/O: Cyclone-Pin 39		general purpose I/O
21 (Leiste B)	I/O: Cyclone-Pin 38	VREF1B4	I/O-Zellen-Referenz
22 (Leiste B)	I/O: Cyclone-Pin 37		general purpose I/O
23 (Leiste B)	I/O: Cyclone-Pin 36		general purpose I/O
24 (Leiste B)	I/O: Cyclone-Pin 35	VREF2B4	I/O-Zellen-Referenz
25 (Leiste B)	I/O: Cyclone-Pin 34	DPCLK7	Dual Purpose CLK
26 (Leiste B)	I/O: Cyclone-Pin 29		general purpose I/O
27 (Leiste B)	I/O: Cyclone-Pin 28		general purpose I/O
28 (Leiste B)	I/O: Cyclone-Pin 27		general purpose I/O

Pin-Nummer	Funktion / alternative Funktion	Alternative Funktion	Bemerkungen
29 (Leiste B)	I/O: Cyclone-Pin 26		general purpose I/O
30 (Leiste B)	I/O: Cyclone-Pin 25		general purpose I/O
31 (Leiste B)	I/O: Cyclone-Pin 24		general purpose I/O
32 (Leiste B)	I/O: Cyclone-Pin 23		general purpose I/O
33 (Leiste B)	I/O: Cyclone-Pin 22		general purpose I/O
34 (Leiste B)	I/O: Cyclone-Pin 21		general purpose I/O
35 (Leiste B)	I/O: Cyclone-Pin 20	VREF2B1	I/O-Zellen-Referenz
36 (Leiste B)	not connected		not used
37 (Leiste B)	not connected		not used
38 (Leiste B)	not connected		not used
39 (Leiste B)	not connected		not used
40 (Leiste B)	not connected		not used

Tabelle 3: Sphinx-C100-Stiftleisten-Belegung

Inbetriebnahme des Sphinx-C100

Betreiben Sie das Sphinx-C100 nur auf einer isolierten Unterlage, um Kurzschlüsse und somit Beschädigungen zu vermeiden. Beachten Sie die maximale Stromaufnahme des Moduls, die vor allem aus der Beschaltung der I/O Pins (d.h. Der Summe der den Pins entnommenen Einzelströme) resultiert. Speisen Sie keine unzulässigen Ströme ein. Bei Überlastung des Moduls wird die entstehende Verlustleistung zu unzulässiger Erwärmung führen und ggf. das Modul beschädigen.

Mechanik des Sphinx-C100

In der folgenden Abbildung sind die wichtigsten Maße des Sphinx-C100 dargestellt. **Alle Angaben sind in mm**.

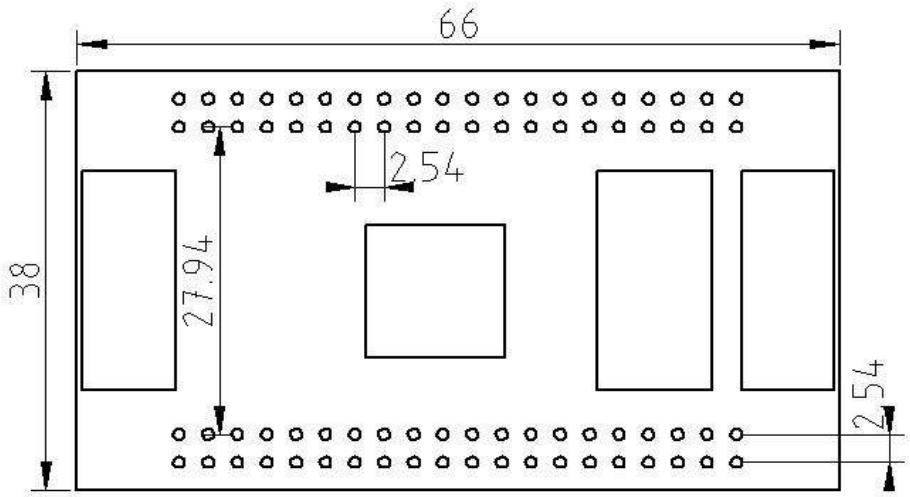


Abbildung 3: Mechanik des Sphinx-C100

Technische Daten des Sphinx-C100

FPGA:
- Typ: Cyclone EP1C3T100
- I/O Pins: 65
- Logical Elements: 2910
- RAM: 59904 Bytes
- Weitere Merkmale : PLL, DDR (s. Datenblatt)

Oszillator:
- Frequenz 66MHz,
- Optionale Frequenzen: 1MHz bis 100MHz (bitte anfragen)

Konfigurationsbaustein:
- Flash: ca. 100000 Programmierzyklen

Programmierschnittstellen:
- JTAG (Joint Test Action Group), auch für Boundary Scan Tests
- Seriell (active serial programming)

Elektrische Versorgung:
- Versorgungsspannung: 5V ±10%, geregelte Gleichspannung
- Stromaufnahme: ca. 50mA, ohne Belastung der Portpins
- Maximal erlaubte Stromaufnahme: 1A

Sonstiges:
- Arbeitstemperaturbereich: 0°C bis +70°C
- Platinenmaße: 66mm x 38mm
- Bauhöhe: 20mm
- Gewicht: ca. 35g

Systemvoraussetzungen bei Verwendung von Quartus II sind eine in Ihren PC eingebaute Netzwerkkarte und folgende (Originalauszug aus der Altera Web-Site):

- to download and run the Quartus II Web Edition software, your computer must meet the following requirements:
  - Pentium II PC running at 400 MHz or faster
  - Microsoft Windows NT version 4.0 (Service Pack 3 or higher), Windows 2000, or Windows XP
  - Microsoft Windows-compatible graphics card and SVGA monitor
  - Microsoft Windows-compatible 2- or 3-button mouse
  - One or more of the following ports:
    - USB port for use with the USB Blaster or MasterBlaster communications cable (Windows 2000 & Windows XP only)
    - Parallel port for use with the ByteBlaster II, ByteBlasterMV, or ByteBlaster parallel-port download cables
    - Serial port for use with the MasterBlaster communications cable

Weiterführende Informationen

Die Produkt-CD beinhaltet eine Sammlung von Datenblättern. Zum besseren Verständnis des FPGAs sei auf die jeweiligen Datenblätter und Application Notes verwiesen Informationen finden sich auf der Inventronik Web-Site: http://www.inventronik.de und bei Altera: http://www.altera.com. Ferner steht auf http://www.inventronik.de ein Forum zur Verfügung, in dem spezielle Probleme erörtert werden können.

Software zum Sphinx C-100

Überblick

Zur Umsetzung Ihrer Designs kann die Entwicklungssoftware QUARTUS II von Altera benutzt werden. Diese ermöglicht sowohl grafische als auch textbasierte Schaltungseingabe. Auf der Produkt-CD befinden sich VHDL Beispiele, die Ihnen einen schnellen Einstieg ermöglichen. Sie finden im Verzeichnis „VHDL-QUARTUS-Projects“ vorgefertigte, ladbare Quartus-projekte und im Verzeichnis „VHDL-Samples“ reine VHDL Quellcodes; letztere für den Fall, dass Sie eine andere Entwicklungsumgebung als QUARTUS II bevorzugen. Die Software QUARTUS II (Web Edition) ist in der Version 3.0 auf der Produkt-CD enthalten. Ferner steht die Web-Edition auf der Altera Web Site unter folgendem Link zum Download bereit:

https://www.altera.com/support/software/download/altera_design/quartus_we/dnl-quartus_we.jsp

Die folgende Beschreibung bezieht sich auf die Verwendung von Quartus II und soll Ihnen einen schnellen Einstieg in die Synthetisierung Ihrer Schaltungen ermöglichen. Die mitgelieferten Beispiele sind ausschließlich textbasierte VHDL-Entwürfe, da die abstrakte Modellierung von Schaltungen auf diese Weise sicherlich ein zukunftsorientiertes Entwurfsverfahren mit der in IEEE1076.1 standardisierten VHDL Programmiersprache darstellt.

Installation von Quartus und Lizenzierung

Die Quartus Designsoftware lässt sich durch ein Windows-Installationsprogramm auf ihrem PC einrichten. In der Programmgruppe „Altera“ finden Sie danach die Software „Quartus II 3 Web Edition Full“, die sie nun starten können. Nach dem Start erscheint eine Meldung „Evaluation Expired“ wie in Abbildung 4 dargestellt. Durch Auswahl von „Run the Quartus II software“ können Sie das Programm trotzdem verwenden, allerdings ist das Programm in diesem Zustand nicht in der Lage Designs zu bearbeiten, da zuerst eine Lizenzierung von Quartus vorgenommen werden muss. Hierzu ist eine Lizenzdatei erforderlich, welche von Altera frei bezogen werden kann und drei Monate Gültigkeit hat. Nach Ablauf dieser Frist kann wieder eine neue Datei angefordert werden. Die Datei erhalten Sie unter folgendem Link:

http://www.altera.com/cgi-bin/authcode91.pl?where_am_i=product&product=q2_web_nic_id

Bitte beachten Sie die Anleitung auf der Altera Web-Site. Sie benötigen für diesen Vorgang eine sogenannte MAC (**M**edia **A**ccess **C**ontrol) Nummer. Diese Nummer ist in ihrer Netzwerkkarte gespeichert, identifiziert ihren PC im Netzwerk und ist weltweit einmalig. Altera verwendet diese MAC Nummer als Referenz zur Erstellung der Lizenzdatei. Auf der im Link angegebenen Seite finden Sie eine Anleitung, wie Sie diese individuelle MAC Nummer Ihres PCs ermitteln können.
Nach dem Ausfüllen eines Online-Formulars erhalten die Lizenzdatei per E-mail. Sie müssen diese anschließend lediglich in das Verzeichnis kopieren, in

welchem sich Quartus befindet (in der Regel ..\Programme\Quartus oder .. \program-file\quartus). Anschließend starten Sie bitte Quartus II neu und wählen Sie die Option „Run the Quartus II software“, falls der in Abbildung 4 dargestellte Hinweis wieder erscheint. Danach können Sie im Menü „Tools-Lisence Setup“ die Lizenzdatei auswählen.

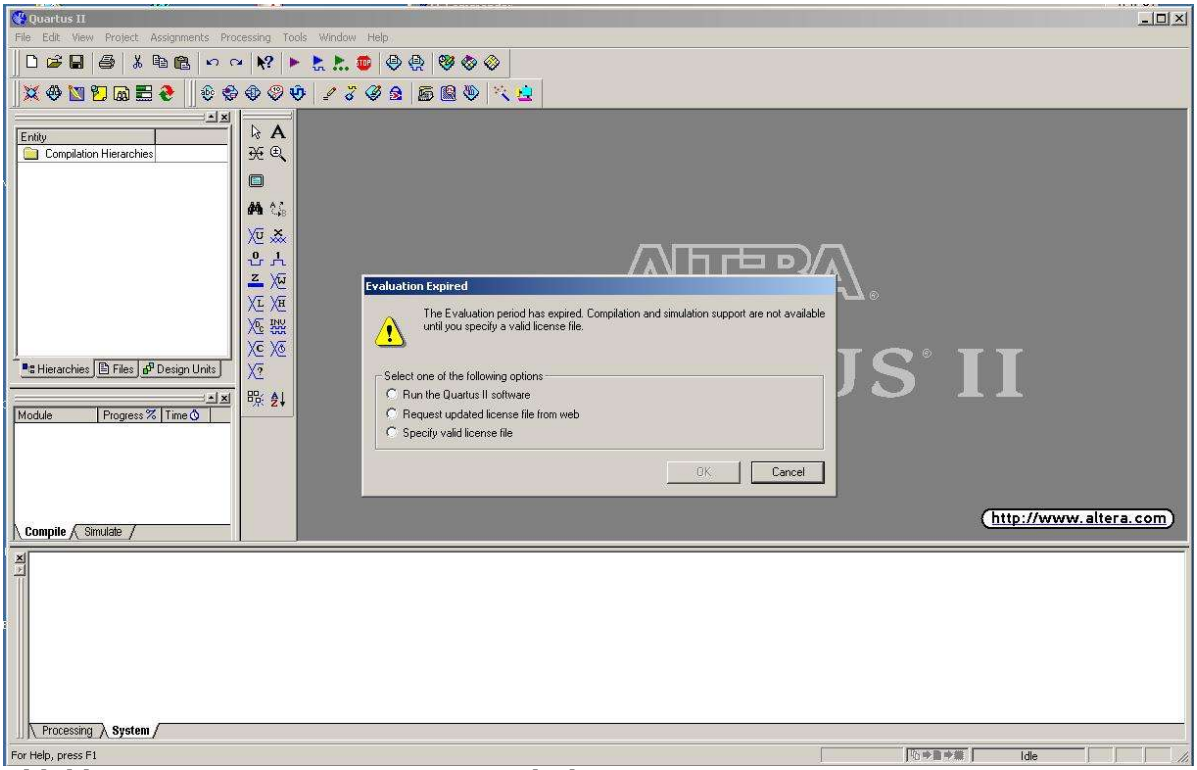


Abbildung 4: Quartus: erster Start, nicht lizenziert

Es ist nicht zwingend notwendig, diese Datei im Quartus Installations-verzeichnis zu speichern, vielmehr kann jedes beliebige Verzeichnis verwendet werden. Abbildung 5 Zeigt ein Beispiel für diesen Vorgang. Wenn Sie die Lizenzdatei richtig angegeben haben, dann wird Quartus II beim nächsten Start ohne die in Abbildung 4 dargestellte Fehlermeldung starten.

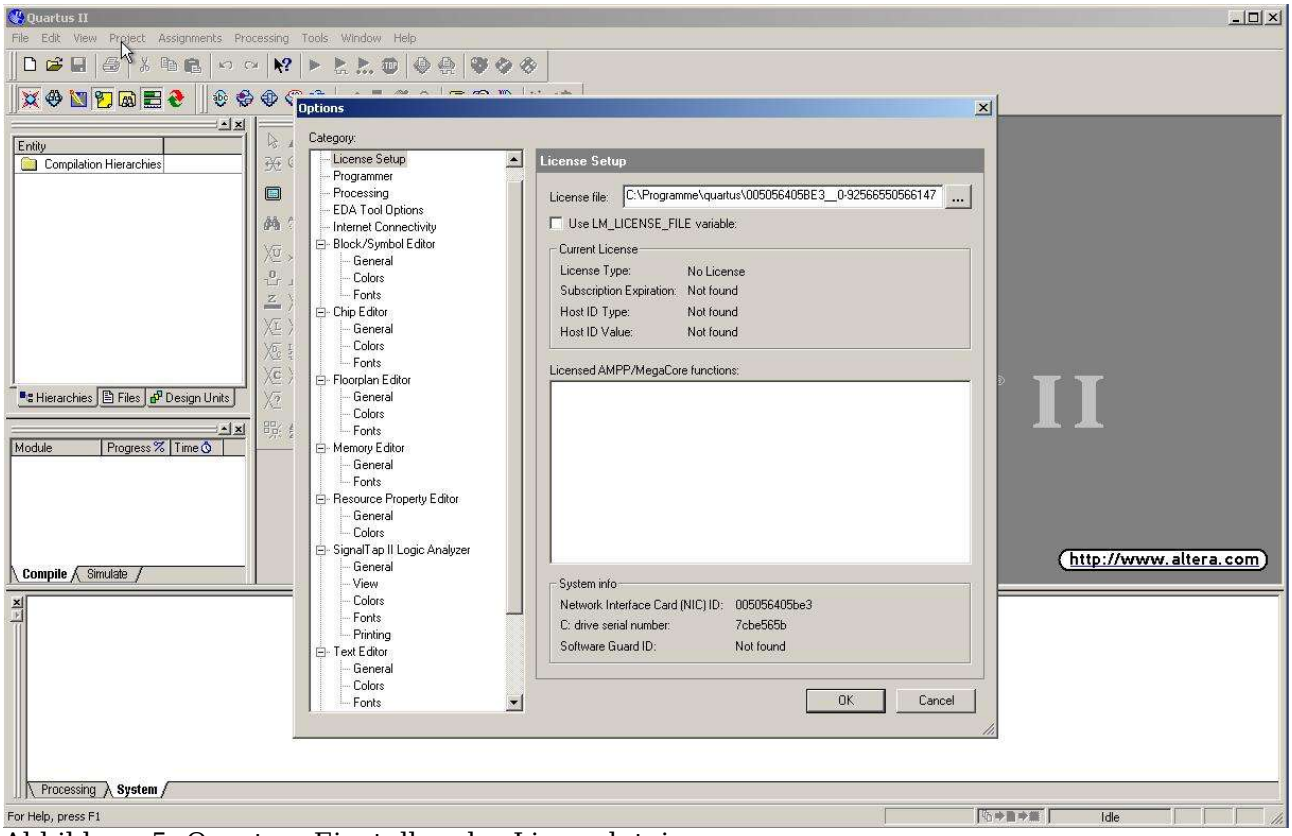


Abbildung 5: Quartus: Einstellen der Lizenzdatei

Installation des Treibers für den Download-Adapter Byte Blaster II, Installation des Adapters.

Für das Programmieren der SPHINX-C100 Module ist eine Verbindung zwischen PC und SPHINX notwendig. Hierfür gibt es Download-Adapter von Altera in verschiedenen Varianten. Die Beschreibung im Folgenden bezieht sich auf den „Byte-Blaster II“, der an die parallele Schnittstelle Ihres PCs angeschlossen wird. Das andere Ende des Adapters wird in die JTAG-Schnittstelle oder in die serielle Schnittstelle (AS) des Sphinx eingesteckt. Zum Betrieb müssen Sie den Sphinx-C100 noch mit einer Spannung von 5V versorgen (s. o.).

Der Treiber für diesen Adapter ist im Quartus Programmverzeichnis enthalten und liegt in ..\quartus\drivers. Die Installation geht folgendermaßen vor sich (am Beispiel Windows 2000):

1. Begeben Sie sich zur Systemsteuerung (Start-Menü, Einstellungen, Systemsteuerung).
2. Doppelklick auf das Hardware-Symbol.

Folgender Dialog erscheint:

Abbildung 6: Installations-Dialog 1

3. Wählen Sie die Schaltfläche „Weiter >“
4. Im nächsten Dialog wählen Sie die erste Option „Gerät hinzufügen .. “ und anschließend die Schaltfläche „Weiter >“
5. Im nächsten Schritt wird nach neuen Hardwarekomponenten gesucht. Das kann einige Zeit dauern. Anschließend erscheint eine Liste mit Komponenten aus der Sie jetzt „Neues Gerät hinzufügen auswählen“. Und mit „Weiter >“ bestätigen.

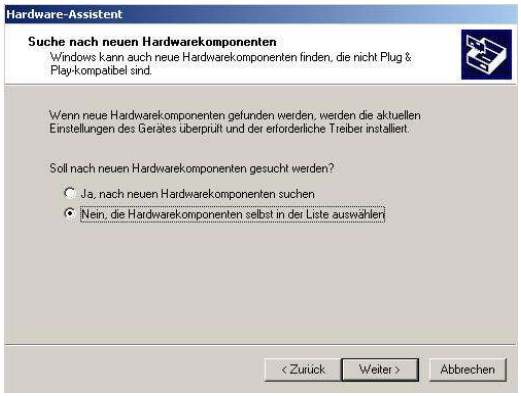


Abbildung 7: Installations-Dialog 2

6. Im nächsten Dialog ist die erste Option voreingestellt selektiert. Sie müssen zur Installation allerdings an dieser Stelle die zweite Option auswählen: „Nein die Hardwarekomponenten selbst in der Liste auswählen“. Quittieren Sie mit „Weiter >“.
7. Im nächsten Schritt selektieren Sie aus der Liste des Dialoges die Option „Audio-, Video- und Gamecontroller“. Der Byte Blaster II wird als Audiogerät behandelt :-).

   Bestätigen Sie die Auswahl mit „Weiter >“.
8. Der nächste Bildschirm beinhaltet eine geteilte Optionsauswahl. Im linken Teilfenster muss die Option „Unknown“ ausgewählt werden. Anschließend wählen Sie die Schaltfläche „Datenträger“ und in der Folge „Durchsuchen“. Wählen Sie das Quartus Installationsverzeichnis auf dem entsprechenden Laufwerk aus (Beispiel: Quartus ist im Verzeichnis C:\Programme\Quartus auf Laufwerk C installiert; dann lautet die Auswahl C:\Programme\Quartus\drivers) und bestätigen mit „Öffnen“ und anschließend „OK“. Der in Abbildung 8 dargestellte Dialog erscheint:
9. Der Byte Blaster Treiber von Altera besitzt keine digitale Signatur. Daher erscheint dieser Dialog. Sie können diesen Dialog mit „Ja“ quittieren, so dass der Vorgang fortgesetzt wird.
10. Im nächsten Dialog wählen Sie schließlich Altera Byte Blaster aus der Liste aus und bestätigen die Wahl mit „Weiter >“.
11. Das nächste Dialogfenster weist darauf hin, dass die Hardware mit Standard-einstellungen installiert werden kann. Dieser Dialog muss ebenfalls mit „Weiter >“ quittiert werden.

Abbildung 8: Installations-Dialog 3

12. Anschließend erscheint nochmals der Hinweis auf die digitale Signatur. Bestätigen sie diesen mit „Ja“.
13. Das letzte Dialogfenster zeigt die erfolgreiche Treiberinstallation an. Mit „Fertigstellen“ wird der Installationsvorgang abgeschlossen. Es ist ein Neustart erforderlich, damit der Treiber aktiv wird.

Alle notwendigen Installationen und Vorbereitungen sind hiermit abgeschlossen. Ihnen steht nun ein sehr leistungsfähiges Entwicklungssystem zur Verfügung, das Ihnen das Umsetzen sehr komplexer Designs auf den Sphinx-C100 erlaubt. Die Oberfläche von Quartus II sieht auf den ersten Blick etwas verwirrend aus! Im Folgenden wird anhand eines Beispiels gezeigt, wie ein Design auf den Baustein programmiert werden kann, wie die Simulationsumgebung aussieht und wie ein neues Projekt begonnen wird. Eine detailliertere Beschreibung kann an dieser Stelle nicht erfolgen, da diese den Rahmen dieses Benutzerhandbuches sprengen würde. Wir verweisen auf die hervorragende Hilfe in Quartus II und auf einschlägige Literatur.

Umsetzen eines Designs in den Sphinx-C100

Einführung

Die Programmierung des Sphinx-C100 erfordert eine Programmierdatei. Dieses wird durch die Synthetisierung einer abstrakten VHDL- oder Verilog-Beschreibung oder einer in QUARTUS II grafisch erstellten Schaltung erzeugt. Hierzu muss dem Compiler zunächst mitgeteilt werden, wie diese Eingabedatei heisst. Anschließend erfolgt das „Fitting“ auf den ausgewählten Baustein. Logischerweise muss der Fitting-Software bekannt gemacht werden, welcher Baustein vorliegt. Ferner gibt es zahlreiche Einstellungen, die Umgebungsvariablen, Optimierungsvarianten etc. beinhalten. Alle diese Informationen sind in einem Projekt zusammengefasst. Auf der Produkt-CD befindet sich ein solches vorgefertigtes und getestetes Projekt. Sie können dieses über den Dialog File -> Open Project -> ...\sphinx_counter_ex1 auswählen, welches im entsprechenden Beispielverzeichnis auf der Produkt-CD zu finden ist. Die Dateiendung von QUARTUS Projektfiles lautet „.quartus“. Wenn dieses Projekt geladen wird, befindet sich auf der linken Seite in einem kleinen Projekt-Navigationsfenster die VHDL-Beschreibungsdatei sphinx_counter_ex1. Durch Doppelklick auf diesen Eintrag öffnet sich in der Mitte die Beschreibungsdatei. Sie sehen hier die abstrakte Darstellung eines parametrierbaren, parallel ladbaren Modulo-N-Vor/Rückwärtszählers. In der dargestellten Form ist dieser 12 Bit breit und inkrementiert bzw. dekrementiert mit jeder steigenden Taktflanke um drei. Der Zähler ist asynchron löschbar. Bereits an dieser Stelle lässt sich die Leistungsfähigkeit des Entwicklungssystems abschätzen, wenn die kurze Beschreibung im Hinblick auf die relativ komplexe Funktionalität des Zählers betrachtet wird. In der folgenden Abbildung ist ein Screenshot der Entwicklungsumgebung mit geladenem Projekt und geöffneter Designdatei dargestellt.

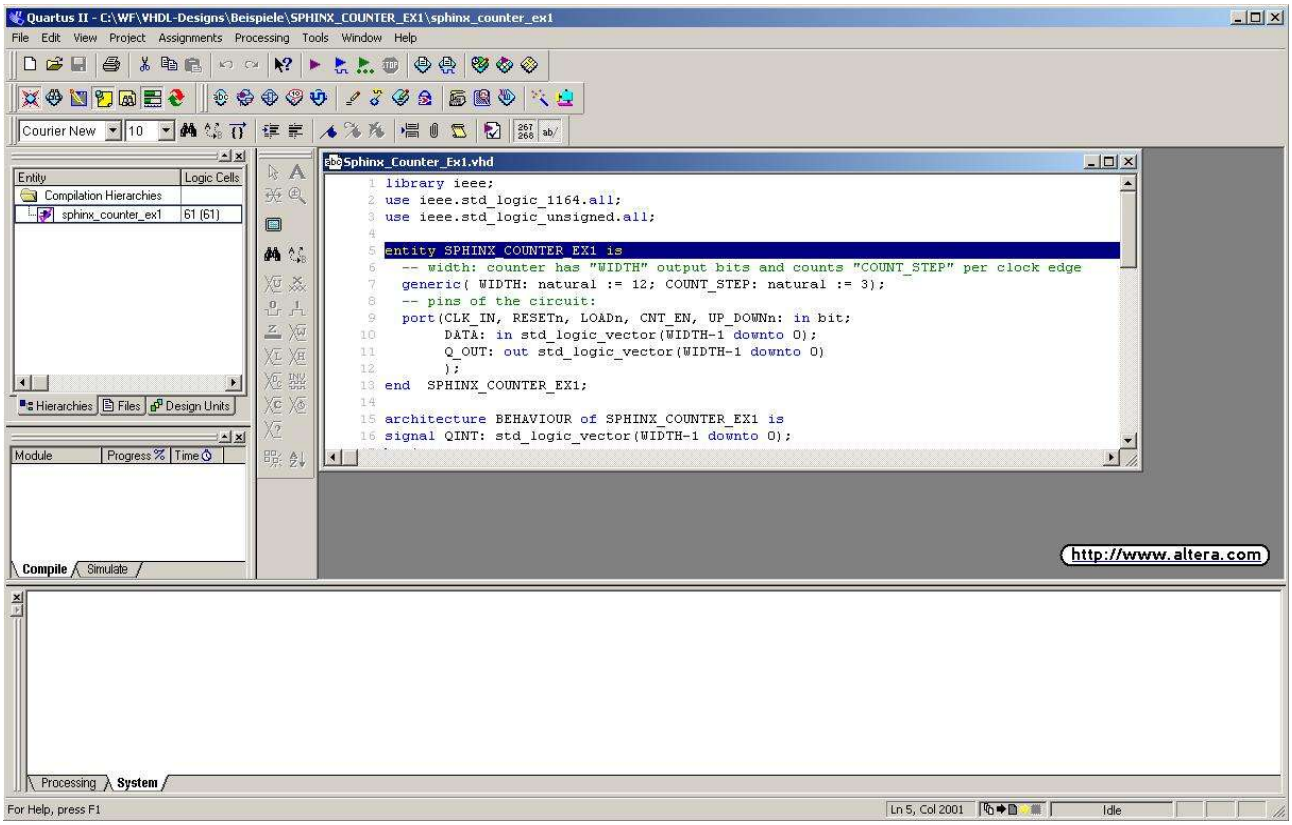


Abbildung 9: Geladenes Quartus Projekt mit geöffneter Designdatei

Unterhalb des Navigations-Fensters sehen sie am linken Rand ein Status Fenster, in welchem der Fortschritt des betreffenden Arbeitsvorganges dargestellt wird. Am unteren Rand befindet sich ein breites Nachrichten-fenster, das Fehlermeldungen, Infos, Warnungen, etc. aufnimmt. Oberhalb dieser 4 Teilfenster befindet sich eine Menüleiste mit Symbolen und darüber eine Menüleiste mit Pulldown-Menüs. Nehmen Sie sich etwas Zeit, um sich mit dieser Umgebung vertraut zu machen. Quartus beinhaltet mehrere Programmteile, die sich im Menü Tools befinden und/oder durch Processing steuern lassen. Kernkomponenten sind hierbei:

1. Compiler: Dieser übersetzt den Programmcode oder die grafisch erstellte Schaltung in einen Objektcode, der als Ausgangsbasis für den Fitter dient, der die so generierte Schaltung auf dem ausgewählten Baustein „anordnet“.
2. Simulator: Jede Schaltung kann nach erfolgreicher Compilierung ohne Zielhardware simuliert werden. Die Übereinstimmung der Simulations-ergebnisse mit tatsächlich gemessenen Werten auf der Zielhardware sind erstaunlich!
3. Programmer (im Menü Tools): Dieses Programm dient zum Programmieren der Zielhardware, also in diesem Fall des Sphinx-C100.

Übersetzen der Designdateien

Die im Hauptfenster dargestellte VHDL Beschreibungsdatei soll im nächsten

Schritt compiliert und für den Sphinx-C100 Baustein gefittet werden. Alle erforderlichen Einstellungen sind bereits in der Projektdatei hinterlegt. Der Compiliervorgang kann im Menü „Processing“ durch „Start „Compiler“ oder durch anklicken des entsprechenden Symbols gestartet werden. Dieser Vorgang kann einige Zeit in Anspruch nehmen. Sie können dabei einige Systemmeldungen im Nachrichtenfenster verfolgen und den Status des Vorganges im Statusfenster beobachten. In Abbildung 10 ist ein Screenshot mit gestartetem Compiler dargestellt.

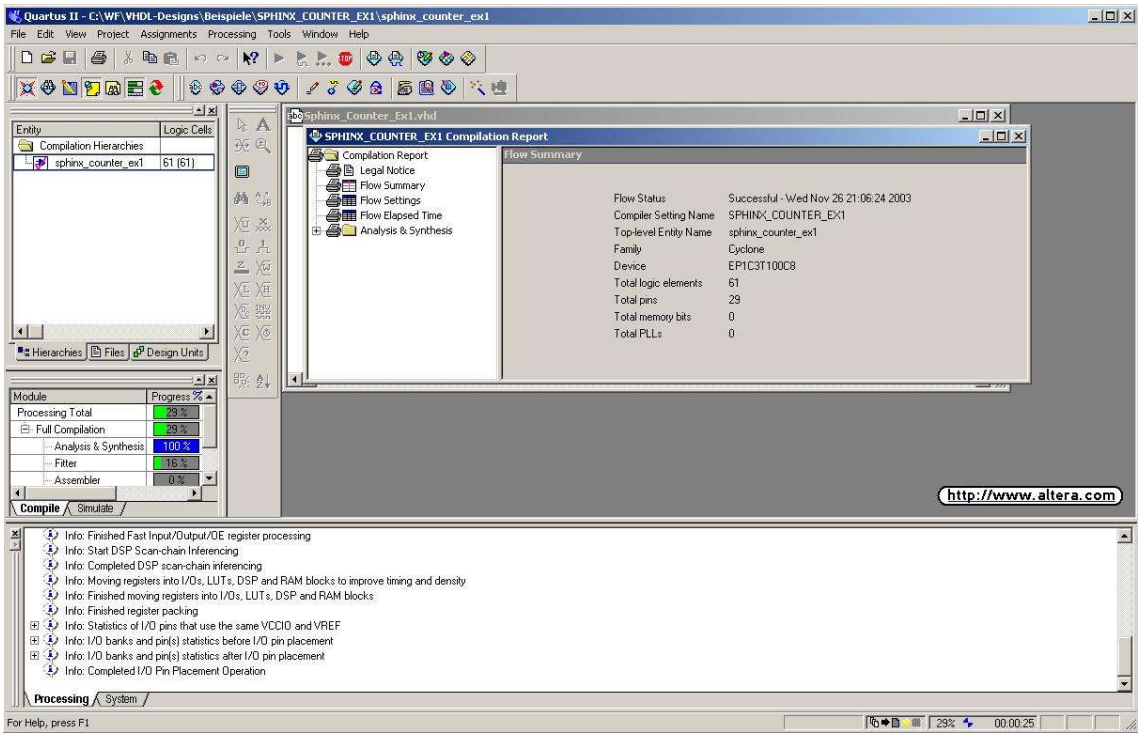


Abbildung 10: Quartus II Compilerlauf

Der Compiliervorgang und der Fitter wird im Falle eines fehlerfreien Designs mit der Meldung „Full Compilation was successful“ abgeschlossen. Bestätigen Sie diese Meldung mit „OK“. Nach der Compilierung und dem Fitting finden Sie im Design-Verzeichnis verschiedene Dateien, die während des Compilerlaufes erzeugt worden sind. Besonderes zu erwähnen sind zwei Dateien mit den Endungen .sof und .pof. Diese dienen zur Programmierung des Sphinx-C100-Moduls. Verwenden Sie diese Dateien (.sof oder .pof), wenn über die JTAG Schnittstelle programmiert wird und die Datei .pof, wenn stattdessen die serielle Programmierung (AS) des Konfigurationsbausteines ausgeführt werden soll.

Programmierung des Sphinx-C100

Zur Programmierung öffnen Sie bitte unter „Tools“ den Eintrag „Programmer“. Es erscheint ein Dialogfenster, in dem noch einige Einstell-ungen vorgenommen werden müssen:

1. Auswahl des angeschlossenen Programmieradapters: Wählen Sie „Hardware“ und dann „Add Hardware“ aus. In dem zuletzt geöffneten Dialogfensters muss anschliessend die Programmierhardware (in diesem

Fall Byte Blaster MV oder Byte Blaster II) und die Schnittstelle, an die die Hardware angeschlossen ist (in diesem Fall lpt1) ausgewählt werden. Nach dieser Auswahl muss die Hardware dann noch mit „Select Hardware“ in die Einstellungen übernommen werden (Byte-Blaster II in der Liste anwählen). Der Vorgang wird mit „Close“ beendet.

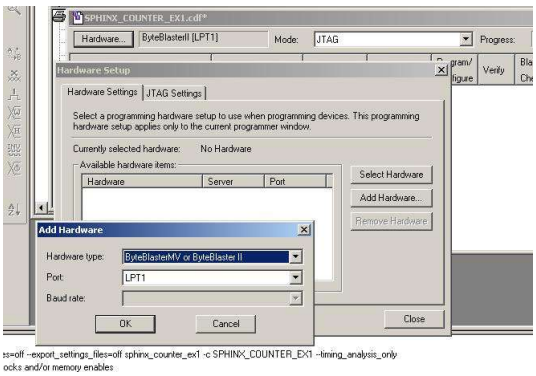


Abbildung 11: Programmieradapter-Auswahl

2. Auswahl des Programmiermodus: hierfür findet sich ein hinter „Mode:“ angeordnetes Drop-Down Menü. Interessant sind die beiden Einträge JTAG oder Active Serial (AS) Programming. Der Mode JTAG wird verwendet, wenn über die JTAG Schnittstelle direkt auf den FPGA programmiert wird. Der AS Mode (letzter Eintrag in der Liste) ist auszuwählen, wenn der Konfigurationsbaustein über die AS-Schnittstelle programmiert werden soll.
3. Letztendlich fehlt nur noch die Programmierdatei, die über ein Mausklick (rechte Maustaste) in den Hauptbereich des Programmierfensters übertragen wird (Add File). In dieser Datei ist die Information, welcher Bausteintyp programmiert werden soll, bereits vorhanden, so dass die Auswahl durch den „Rechts-Mausklick“ (Add Device) entfallen kann.
   Zur Information: auf dem Sphinx-C100 befindet sich ein Konfigurations-baustein vom Typ **EPCS1**. Die erfolgreiche Konfiguration des Programmiertools sieht für das geöffnete Projekt sphinx_counter_ex1 folgendermaßen aus:

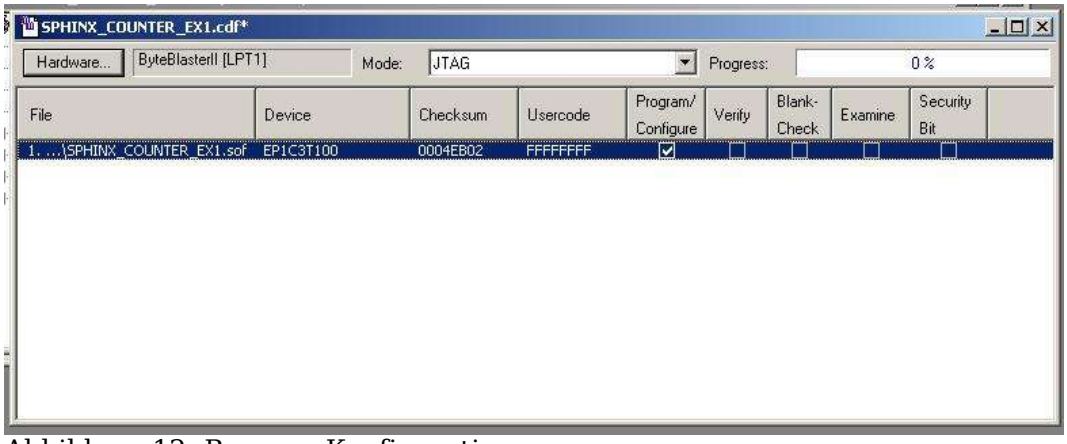


Abbildung 12: Brenner-Konfiguration

In Abbildung 12 ist die ausgewählte Konfigurationsdatei blau hinterlegt dargestellt. Dahinter befinden sich Informationen über die Checksumme und den Usercode (FFFFFFFF bedeutet, dass kein Usercode programmiert wurde). Rechts von diesem sind die Optionen, die das jeweilige Programmierverfahren zur Verfügung stellt, in Auswahlkästchen anwählbar. Im Falle von JTAG kann nur „Program/Verify“ angewählt werden. Im Falle von AS ist „Program/Configure“, „Verify“, „Blank-Check“ und „Examine“ möglich.

Der Download-Vorgang wird schließlich im Quartus II Hauptmenü „Processing“ -> „Start Programming“ gestartet. Die Programmierung des

Sphinx-C100 dauert nur wenige Sekunden.

Sie haben, wenn Sie die Dokumentation bis an diese Stelle nachvollzogen haben, den kompletten Entwicklungszyklus für die Umsetzung eines digitalen Designs durchgeführt. Sie können sich im Hauptmenü unter „Assignments“ -> „Last Compilation Floorplan“ die Pinbelegung anschauen und Ihre Hardware auf Funktion überprüfen. Was jetzt noch fehlt, ist die Eingabe des Designs an sich, welches im hier besprochenen Beispiels als VHDL-Datei mit dem Projekt geladen wurde. Wenn Sie ein eigenes Design erstellen wollen, so ist das Erstellen eines neuen Projektes notwendig. Dies wird im Folgenden kurz beschrieben.

Ein neues Quartus II Projekt erstellen

Die Erstellung eines neuen Projektes ist sehr einfach, wenn sie im Menü „File“ den Eintrag „New Project Wizard“ verwenden. Dies ist ein Menü-geführtes Projekterstellungssystem mit umfangreichen Hilfefunktionen. Beachten Sie, dass Sie ein neues Verzeichnis und nicht versehentlich ein bestehendes Verzeichnis verwenden, in dem sich bereits Dateien eines Projekts befinden. Wird der „New Project Wizard“ beendet, so sind alle erforderlichen Einstellungen vorgenommen.

Sehr wichtig bei der Erstellung eines neuen Projekts ist die Auswahl des Bausteines. Der Sphinx-C100 ist mit einem Cyclone EP1C3T100C8 bestückt.

Starten Sie durch Auswahl des Menüs „File“ eine neue Datei (grafisch oder VHDL oder Verilog). Diese Datei ist mit dem gleichen Namen wie das Projekt zu benennen (oberste Hierarchie).

Beispiel: Ein Projekt wird unter dem Namen „Christbaumlauflicht“ erstellt. Alle Pfadangaben weisen auf ein leeres Projektverzeichnis z. B. auf „Christbaumlauflichtverzeichnis“ und das Projekt heisst „Christbaumlauflicht“ (diese Einstellungen werden mit dem „New Project Wizard“ durchgeführt). In diesem Verzeichnis befinden sich nach erfolgreicher Compilation alle erzeugten Dateien (die Datenbasis). Wenn Sie nun eine neue Designdatei öffnen (beispielsweise eine grafische Beschreibungsdatei), dann sollten Sie diese unter dem Namen „Christbaumlauflicht“ im Projektverzeichnis speichern. Der Compiler sucht dann nach dieser Projektdatei und beginnt mit dieser Datei. Wenn VHDL-Dateien erzeugt werden, so muss zusätzlich sichergestellt werden, dass die Entity der „top level“-Datei ebenfalls „Christbaumlauflicht“ heisst.

Unter Beachtung dieser, anfänglich oft verwirrenden Zusammenhänge sollte es nun leicht möglich sein, erstellte Designs zu compilieren und in den Sphinx-C100 umzusetzen.

Simulation eines Projektes am Beispiel sphinx_counter_ex1

Die Simulation eines Projektes kann mit dem in Quartus II integrierten Simulator erfolgen. Die Beschreibung im Folgenden bezieht sich auf diesen Simulator. Die Simulation läuft folgendermaßen ab:

Nach erfolgreicher Compilierung des Projektes wurden neben den Programmierdateien (s. o.) unter Anderen auch eine Timing-Datei erstellt. Diese Datei enthält Angaben für den Simulator, definiert den ausgewählten Baustein und berücksichtigt die Anordnung der Schaltung auf dem Baustein. Es ist daher nicht möglich, eine Simulation vor dem Compilieren und dem Fitten durchzuführen.

Zur Simulation muss dem Simulator mitgeteilt werden, welche Eingänge berücksichtigt und welche Ausgänge nach dem Simulationslauf grafisch dargestellt werden sollen. Diese Informationen werden in einer „Vector Waveform Datei“ gesichert, die Sie mit der in der folgenden Abbildung dargestellten Schaltfläche öffnen können.

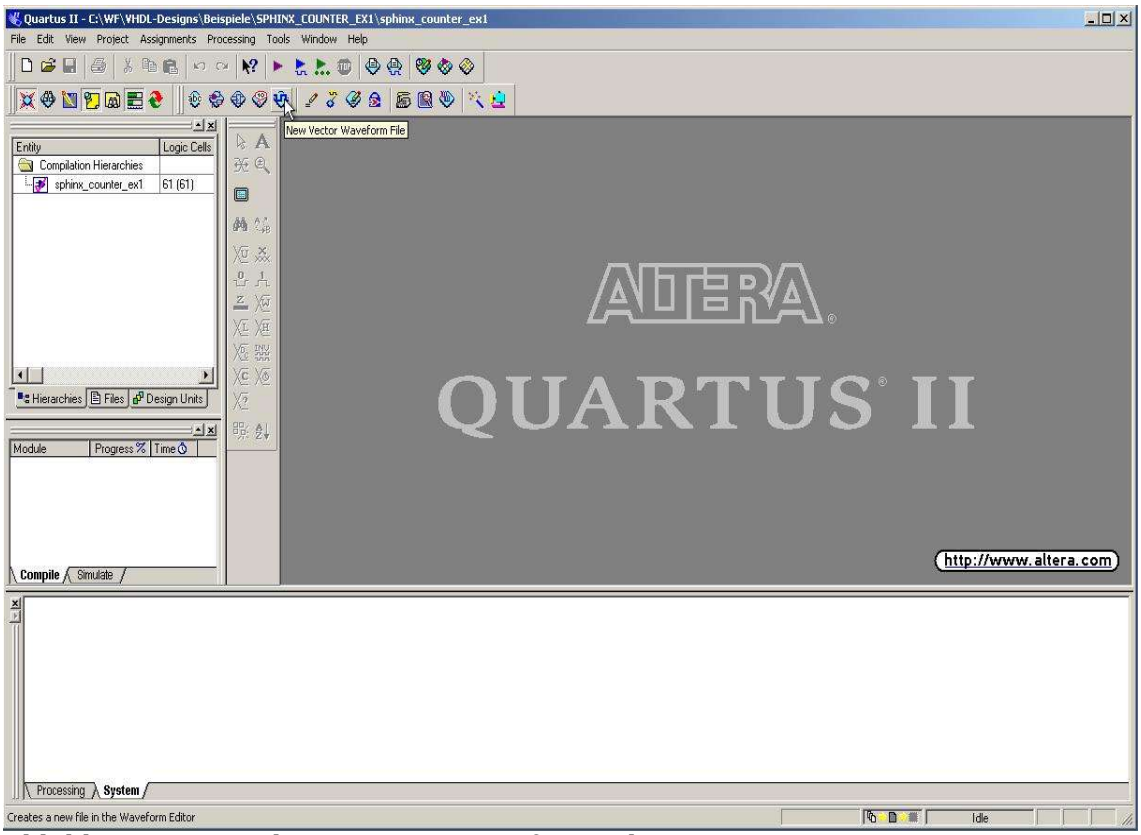


Abbildung 13: Simulator: Vector Waveform File

Nach dem Öffnen der Waveform Datei sehen Sie ein zweigeteiltes Fenster. Klicken Sie mit der rechten Maustaste in das linke Teilfenster, worauf ein Dialog die Auswahl „Insert Node or Bus“ anbietet. Wählen Sie diesen Eintrag. Im darauf folgenden Fenster wählen Sie den „Nodefinder“ aus. Es wird ein weiteres Fenster geöffnet, in welchem die Ein- und Ausgänge der Simulation zugewiesen werden können. Zunächst sehen Sie zwei leere Teifenster, die nach Auswahl des „Start“ Schaltfläche im linken Teilfenster alle möglichen Signale des Designs erscheinen lässt. Wählen Sie diejenigen, die Sie für die Simulation verwenden wollen aus und kopieren Sie diese mit dem hierfür vorgesehenen Schaltknopf „>“ in das rechte Teilfenster. Wenn alle Signale

ausgewählt wurden wird mit „OK“ bestätigt. In Abbildung 14 ist das Beispiel für die Simulation des Counters dargestellt.

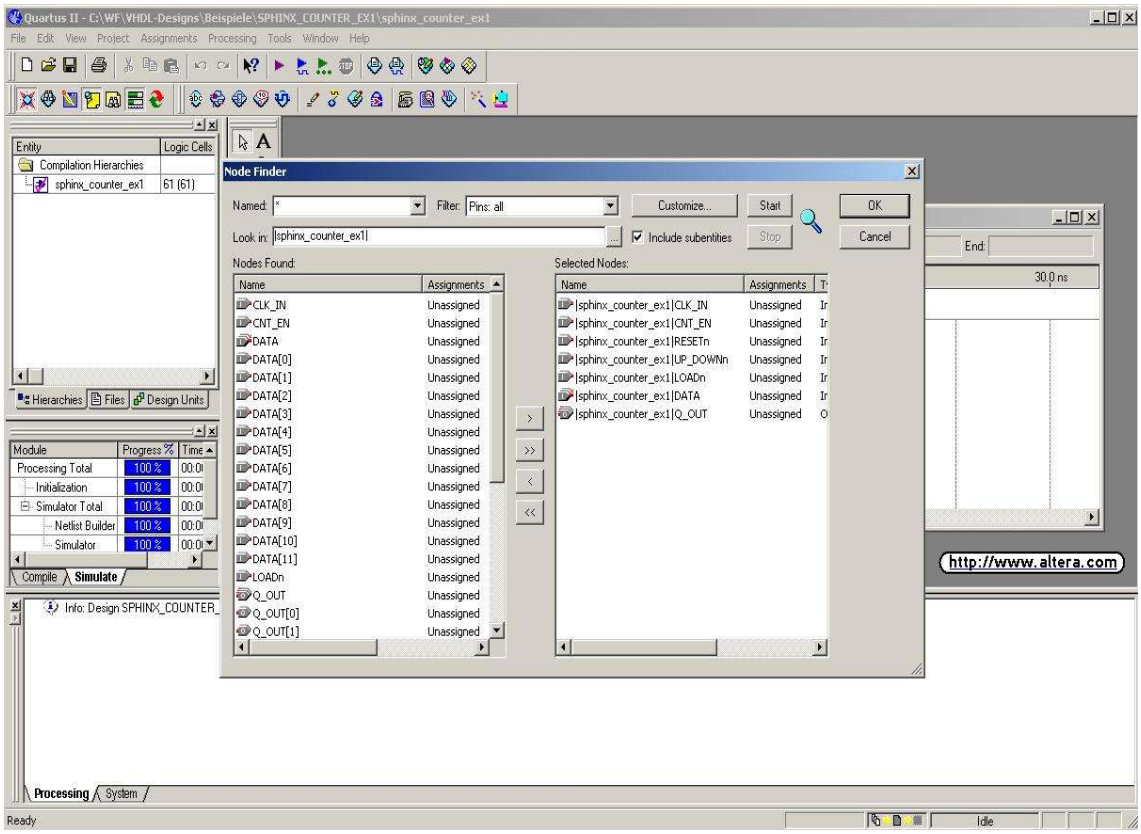


Abbildung 14: Simulator: Node-Finder

Nachdem die Signale ausgewählt und bestätigt wurden, öffnet sich schließlich das Simulatorfenster. In diesem werden die Signale als Spuren dargestellt. Für die Simulation ist es notwendig die Eingänge mit Werten zu belegen. Der Simulator ermittelt aus diesen Vorgaben das Verhalten der Ausgänge. Die Wertzuweisung erfolgt, indem mit der rechten Maustaste die Eingangssymbole ganz links angewählt werden. Mit Value lassen sich die unterschiedlichsten Zuweisungen festlegen. Bitte lesen Sie hierzu die Online Hilfe von Quartus oder „spielen“ sie etwas mit den angebotenen Möglichkeiten, um ein Gefühl hierfür zu entwickeln.

Anmerkung: Die Gesamtzeit der Simulation kann im Pulldown-Menü „Edit“ -> „End Time“ eingestellt werden. Wenn alle Eingänge stimuliert sind, wird der Simulationsvorgang im Pulldown-Menü „Processing“ -> „Start Simulation“ gestartet.

Im folgenden, letzten Screenshot, ist das Simulationsergebnis dargestellt. Die sichtbaren Bereiche können vergrößert werden und es lassen sich Messungen von Zeiten durchführen usw. (s. hierzu die Online Hilfe). In dieser Abbildung ist dargestellt, wie die Eingangssignale stimuliert wurden. Für sinnvolle Simulationen ist natürlich etwas Übung, Fingerspitzengefühl und natürlich viel Erfahrung notwendig.

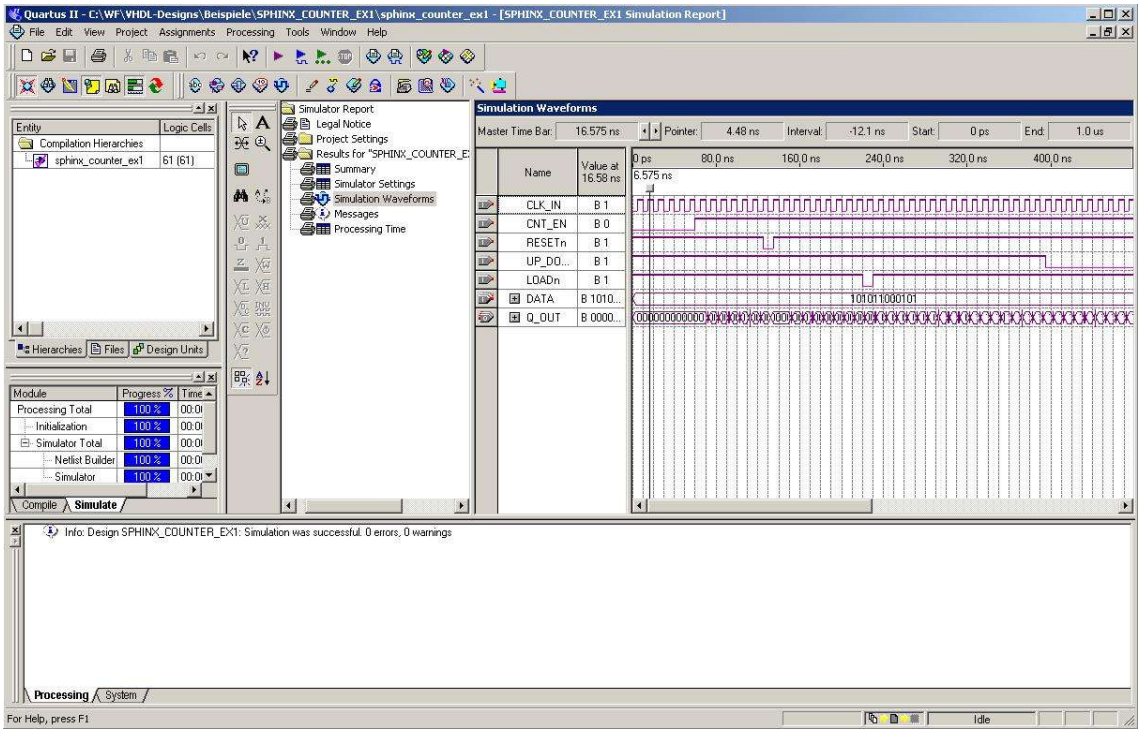


Abbildung 15: Simulationsergebnis für sphinx_counter_ex1

Sie haben nun die wichtigsten Werkzeuge kennengelernt, um eigene Designs in Hardware umzusetzen und zu simulieren. Wir hoffen, dass Sie mit diesem Produkt zufrieden sind und dass es Ihnen hilft, Entwicklungszyklen zu beschleunigen und moderner zu gestalten. Sie finden unter http://www.inventronik.de immer aktualisierte Informationen und Beispiele, die Sie sich gerne herunterladen können.

Wir wünschen Ihnen viel Spass mit diesem Produkt und viel Erfolg bei der abstrakten Modellierung Iher digitalen Schaltungen.

Anhang

Urheberrechtshinweis

Haftungsauschluss

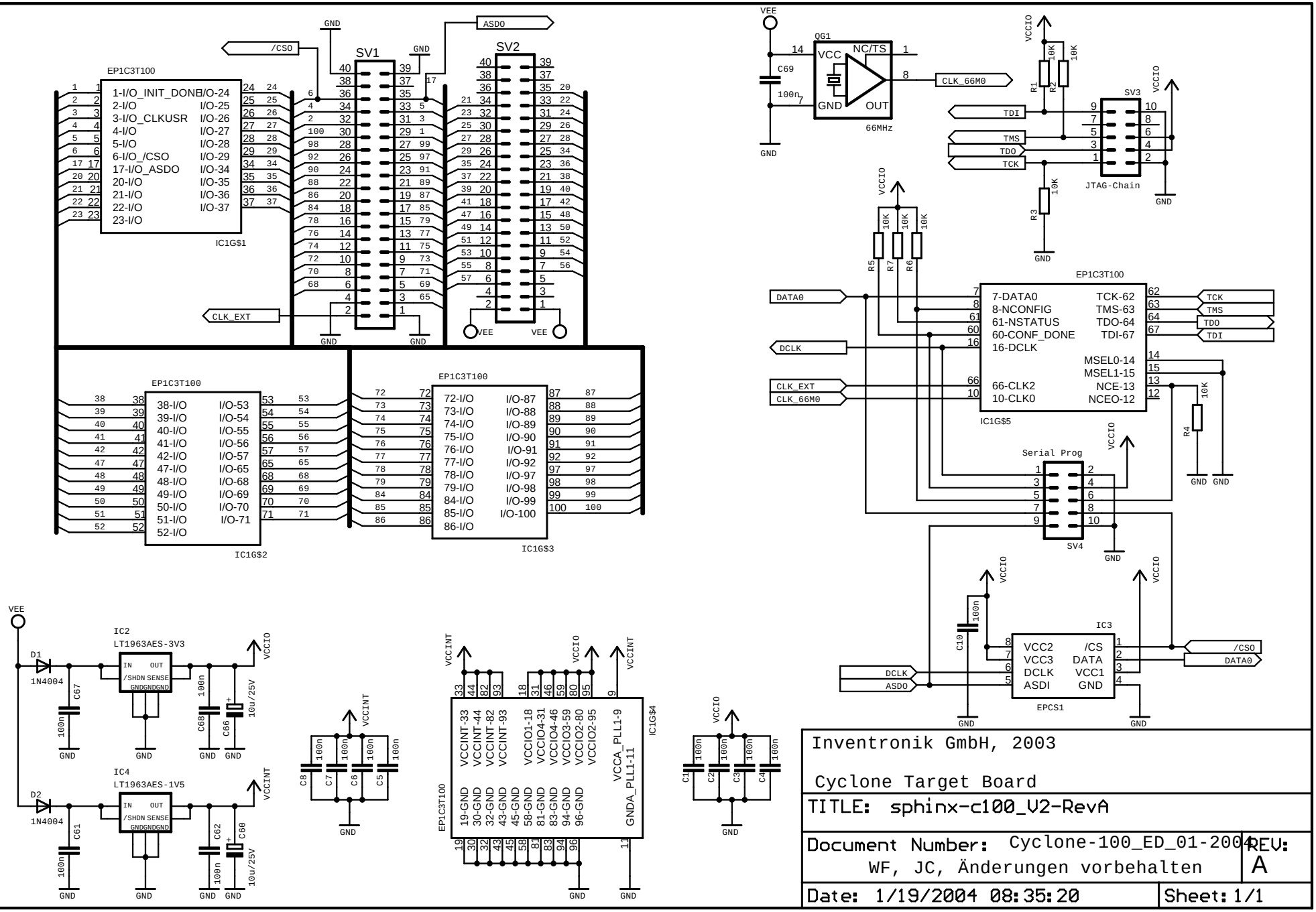
Die Firma Inventronik übernimmt keinerlei Haftung für die Verwendung des Sphinx-C100 sowie der mitgelieferten oder von unserer Internetseite heruntergeladenen Software. Die Firma Inventronik übernimmt ebenfalls keinerlei Haftung, falls durch die Verwendung dieser Produkte Patent- oder Lizenzrechte Dritter verletzt werden. Die Firma Inventronik behält sich das Recht vor, technische Änderungen im Sinne von Produktverbesserungen ohne vorherige Ankündigung durchzuführen sowie diese Dokumentation zu verändern und zu ergänzen.

Marken und Warenzeichen

Die in diesem Dokument erwähnten Markennamen oder Warenzeichen dienen lediglich der Identifikation und sind Eigentum ihrer jeweiligen Inhaber.

Garantiebestimmungen (Gewährleistung)

Die Firma Inventronik gewährt bei sachgemässer Handhabung eine Garantie von zwei Jahren auf dieses Produkt. Die Garantie erlischt durch unsachgemässe Behandlung oder Manipulation.



Inventronik GmbH, 2003

Cyclone Target Board

TITLE: sphinx-c100_V2-RevA

Document Number: Cyclone-100_ED_01-2004 REV: A

WF, JC, Änderungen vorbehalten

Date: 1/19/2004 08:35:20 Sheet: 1/1