Praktikum Digitaltechnik SS 2011

Versuch 3

Stand: 16. 6. 11

Versuchsplattform:
Lehrgerät 10a mit CPLD Evaluation Board (Pollin) und zwei Siebensegmentanzeigen 10a. Eingebaute Ein- und Ausgabemittel: vier Taster, acht LEDs, serielle Schnittstelle. Entwicklungssoftware: Xilinx ISE 9.x.

Aufgaben:
- Den Entwicklungsgang kennenlernen.
- Passende Taktfrequenzen herstellen.
- Lauflicht.
- Zwei Binärzähler mit Handbetätigung und Anzeige.
- Stoppuhr.
- Prüfgenerator für serielle Schnittstellen.

Aufgabe 1: Den Entwicklungsgang kennenlernen
Das erste Projekt betrifft ein RS-Flipflop und ein Toggle-Flipflop, die mit Tasten und Leuchtdioden zu verbinden sind.

1. Software starten.

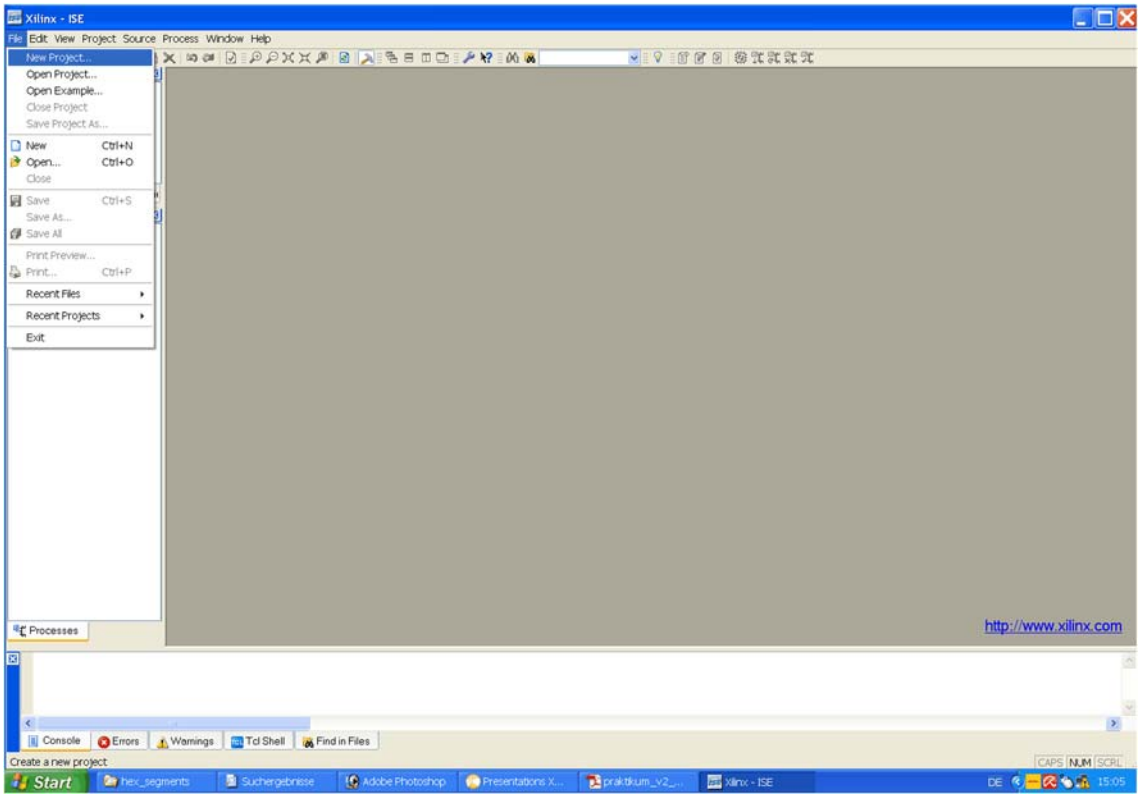


Abb. 1 Anlegen eines neuen Projekts. Funktionsaufruf.

2. Projekt benennen.

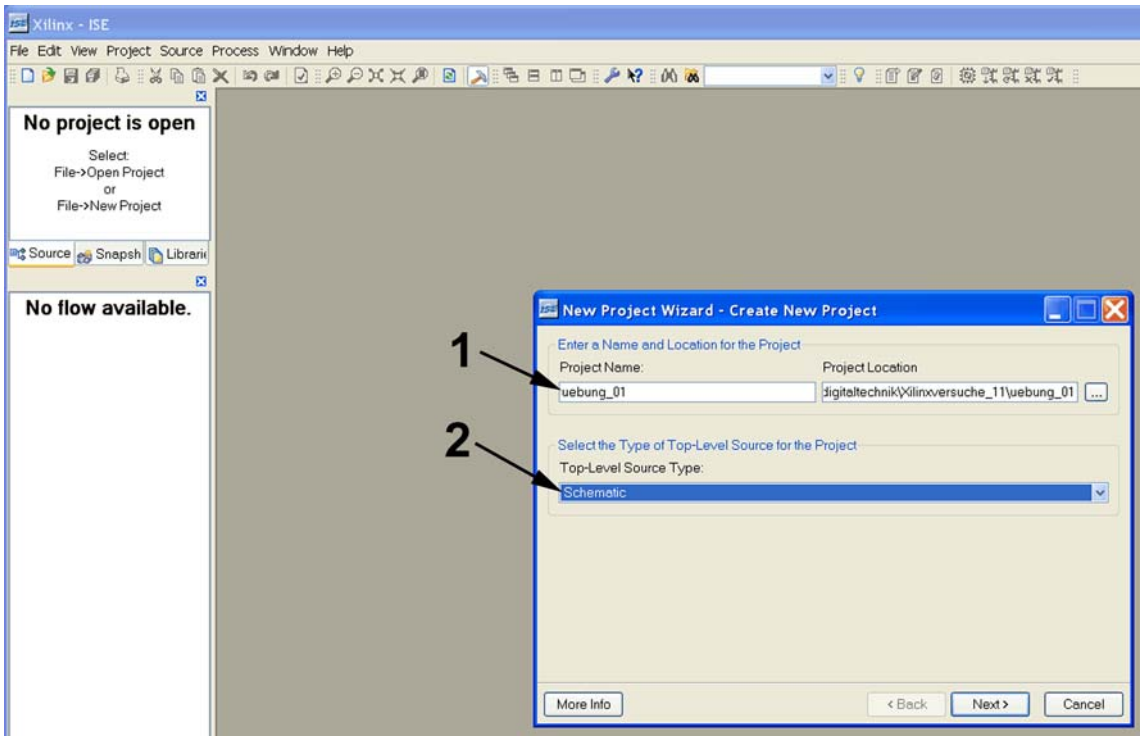


Abb. 2 Das neue Projekt erhält einen Namen. 1 - Namenseingabe. 2 - das Projekt soll als Schaltplan erfaßt werden.

3. Schaltkreis auswählen.

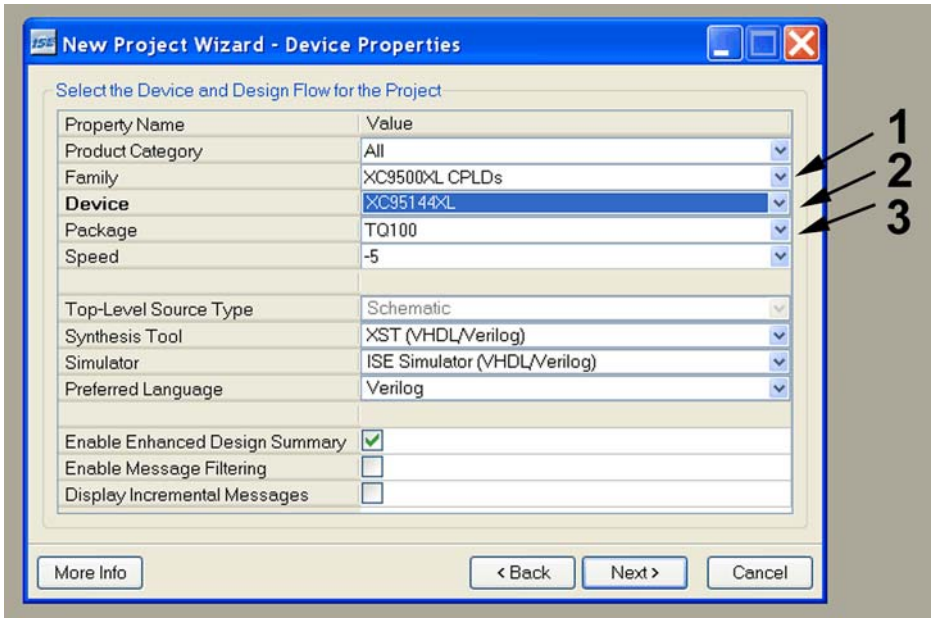


Abb. 3 Schaltkreisauswahl. 1 - es ist die Baureihe 9500XL. 2 - es ist der Typ 95144. 3 - der Schaltkreis befindet sich in einem Flatpack-Gehäuse mit 100 Anschlüssen.

4. Die erste Quelldatei einrichten.

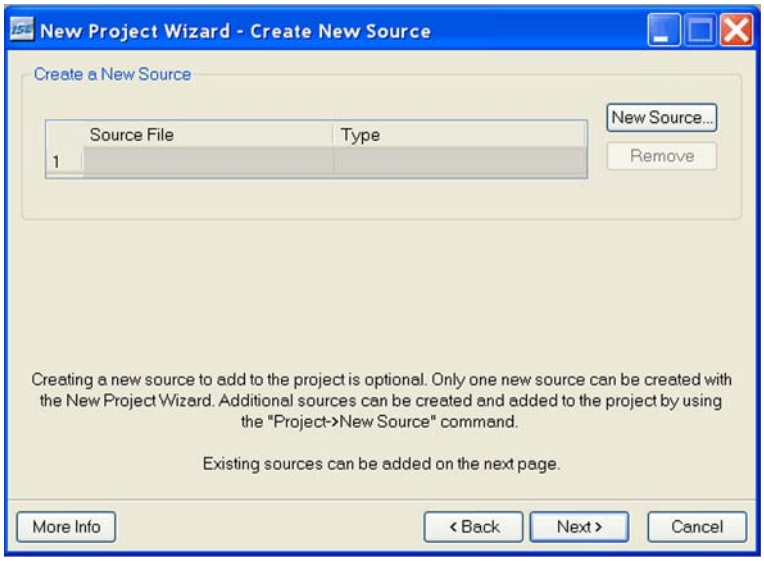


Abb. 4 Einrichten der ersten Quelldatei.

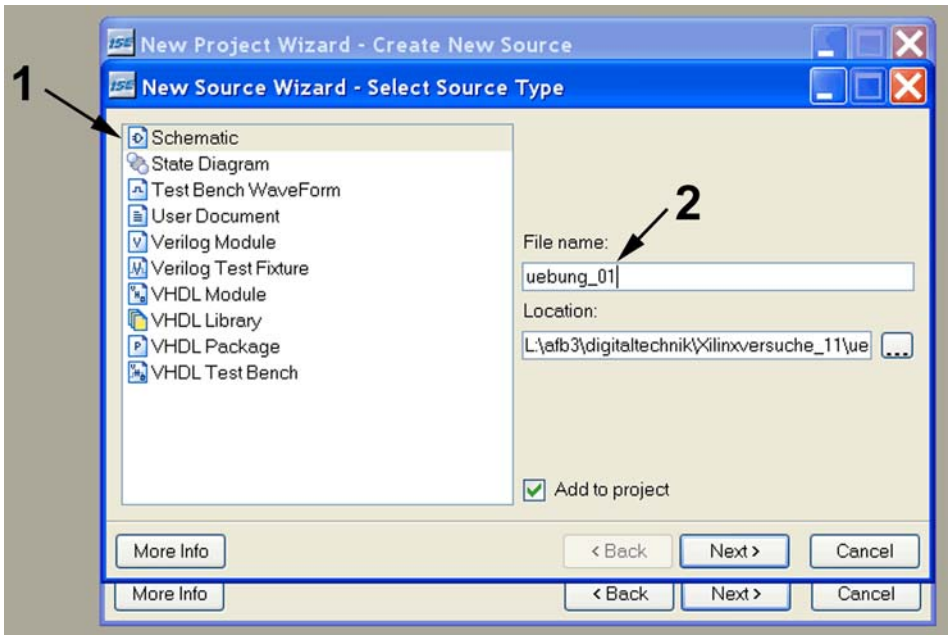


Abb. 5 Es soll ein Schaltplan sein. 1 - den Typ der Quelldatei wählen. 2 - Namen eingeben (beliebig; Dateiendung wird automatisch nachgesetzt).

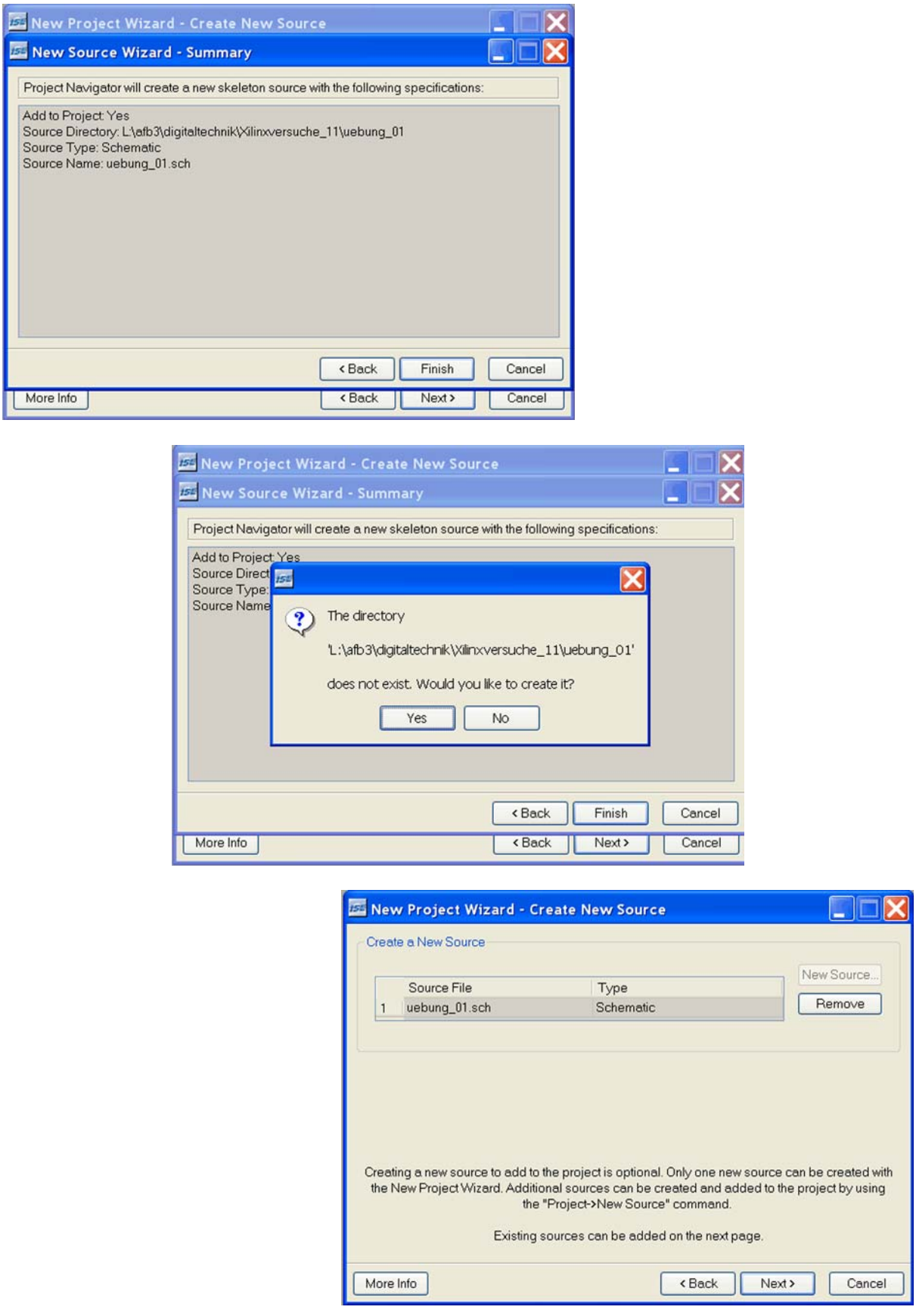


Abb. 6 Alle diese Dialoge sind zu durchlaufen, um die Quelldatei tatsächlich zu erzeugen.

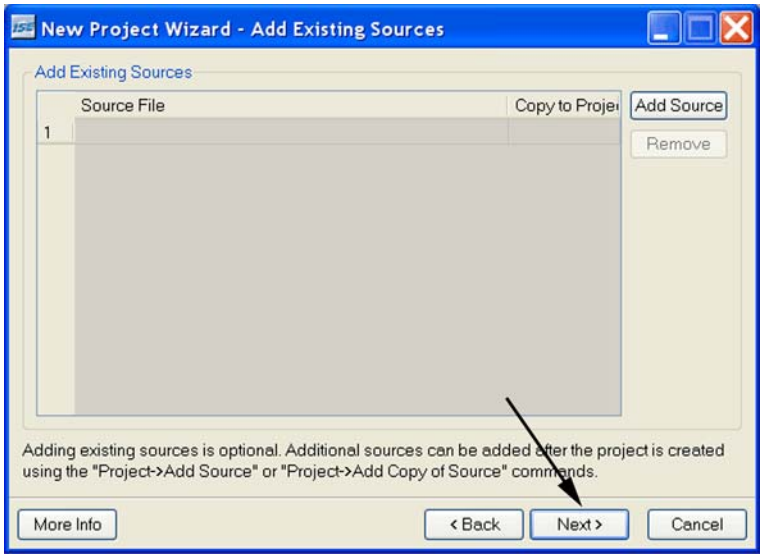


Abb. 7 Hier wird die Gelegenheit geboten, weitere Quelldateien hinzuzufügen. Brauchen wir aber nicht. Also gleich weiter...

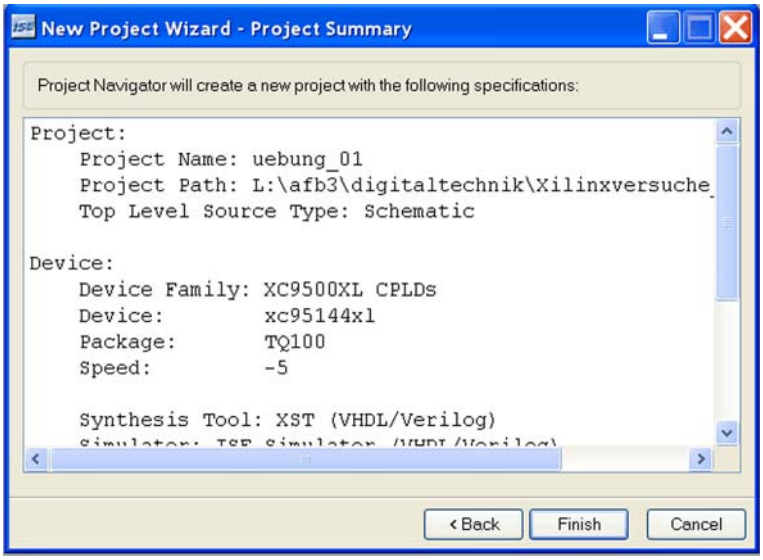


Abb. 8 Das ist der abschließende Überblick über das neue Projekt.

5. Schaltplan eingeben.

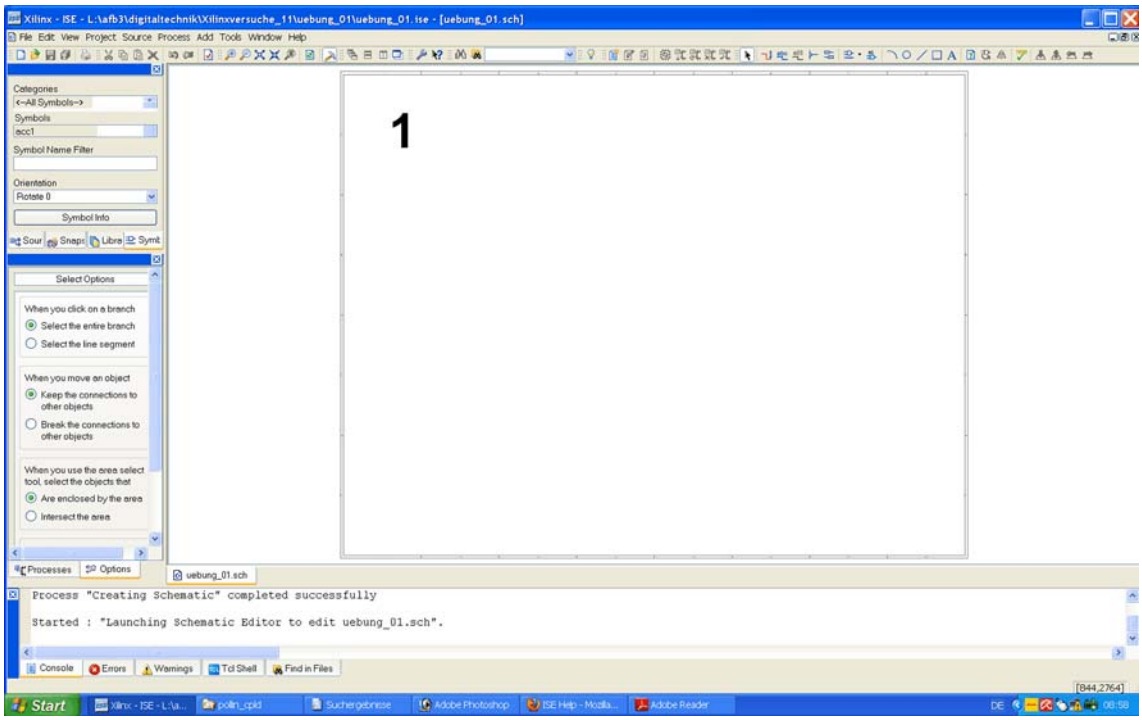


Abb. 9 Der Schaltplaneditor. 1 - die Zeichenfläche.

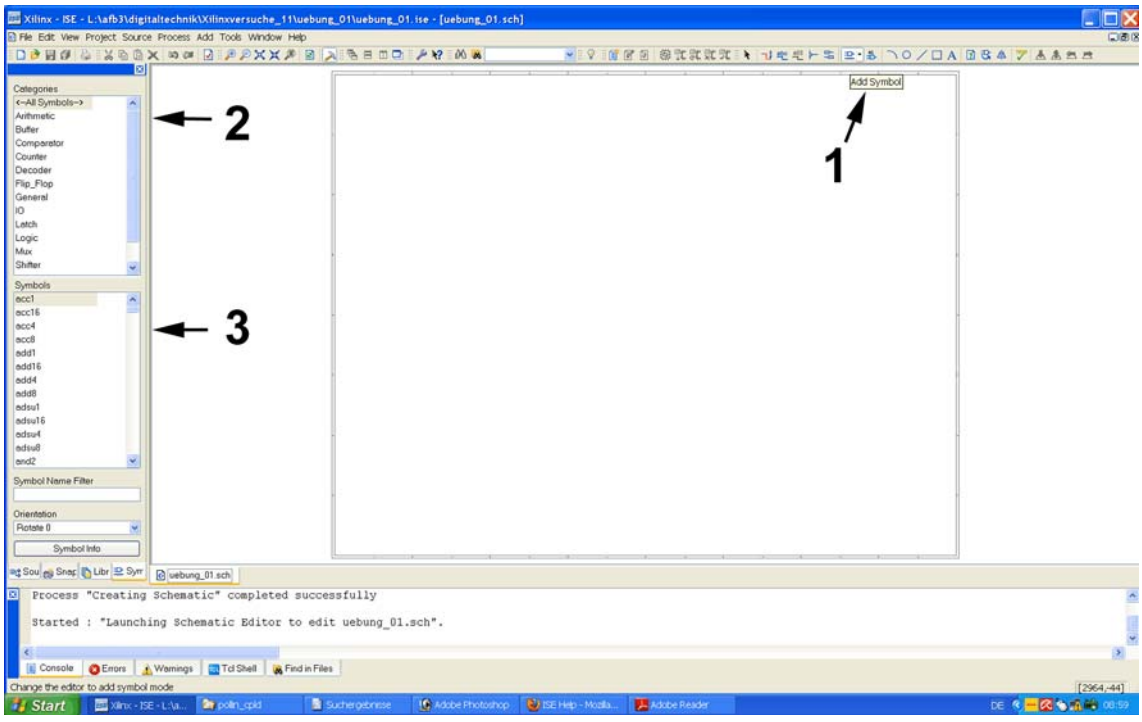


Abb. 10 Funktionselemente auswählen. 1 - Auswahlfunktion. Am linken Rand erscheint ein Katalog der verfügbaren Funktionselemente. 2 - Auswahl der Kategorie. 3 - hier wird das jeweilige Funktionselement aufgerufen.

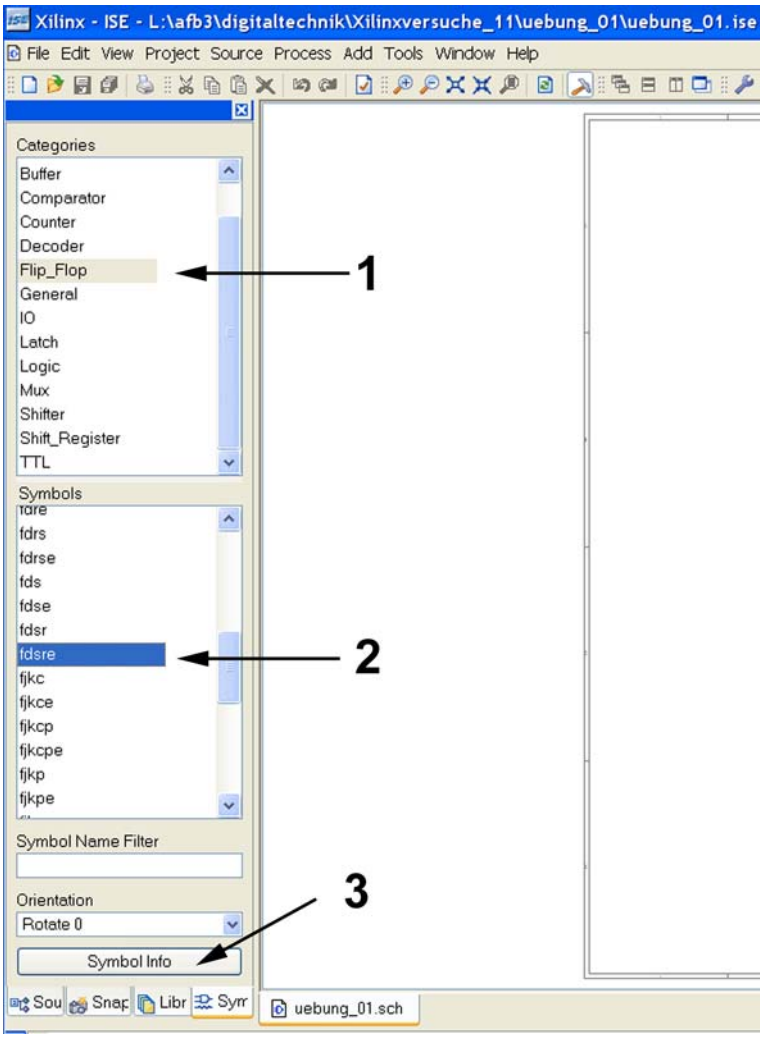
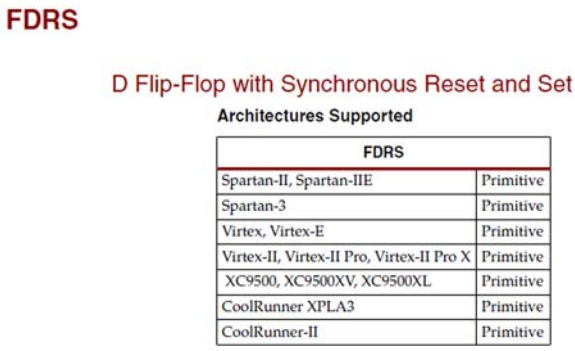


Abb. 11 Das erste Funktionselement soll ein RS-Flipflop sein. 1 - nachsehen, ob unter den Flipflops etwas Passendes zu finden ist. 2 - wir nehmen ein D-Flipflop mit zusätzlichen Setz- und Rücksetzeingängen. Vielleicht tut es der Typ *fdrs*? 3 - hierüber kann man genauer nachsehen.



FDRS

D Flip-Flop with Synchronous Reset and Set

Architectures Supported

FDRS	
Spartan-II, Spartan-IIE	Primitive
Spartan-3	Primitive
Virtex, Virtex-E	Primitive
Virtex-II, Virtex-II Pro, Virtex-II Pro X	Primitive
XC9500, XC9500XV, XC9500XL	Primitive
CoolRunner XPLA3	Primitive
CoolRunner-II	Primitive

Abb. 12 Die *Symbol Info* ist im Grunde ein Datenblatt. Offensichtlich ist dieses Flipflop ungeeignet, da die Setz- und Rücksetzeingänge synchron wirken.

Bei Xilinx sind Set (S) und Reset (R) synchron wirkende Eingänge. Die entsprechenden asynchron wirkenden Eingänge heißen Preset (P; PRE) und Clear (C; CLR). Also nehmen wir besser ein Flipflop *fdcp*.

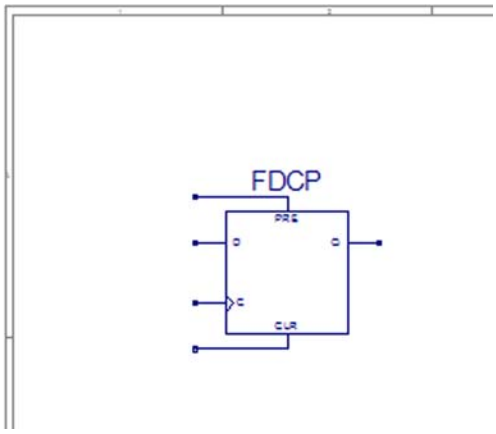


Abb. 13 Das gewünschte Flipflop auf der Zeichenfläche.

Die Taster der Versuchsplattform wirken invertiert, die Flipflopeingänge jedoch nicht (anders als bei den Flipflopschaltkreisen der ersten beiden Versuche). Wir müssen also Negatoren vorschalten.

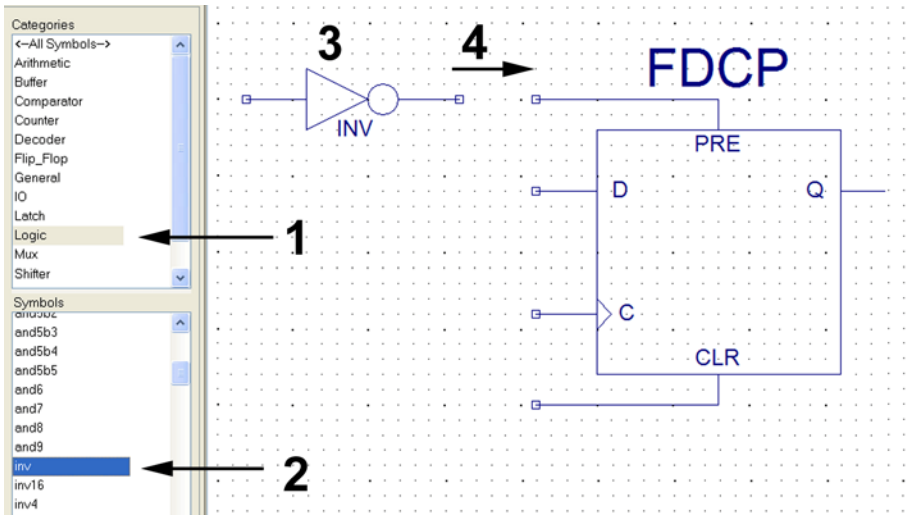


Abb. 14 So wird ein Negator eingebaut. 1 - er ist in der Kategorie Logic zu finden. 2 - er heißt hier inv (Inverter). 3- das ausgewählte Funktionselement. 4 - die Anschlüsse können direkt miteinander verbunden werden (Andocken).

Nun ist die Schaltung mit den Schaltkreisanschlüssen zu verbinden. Jeder Anschluß erfordert zwei Funktionselemente, einen *Buffer* und einen *Marker*. Ein Eingang braucht einen Input Buffer und einen Input Marker, ein Ausgang einen Output Buffer und einen Output Marker.

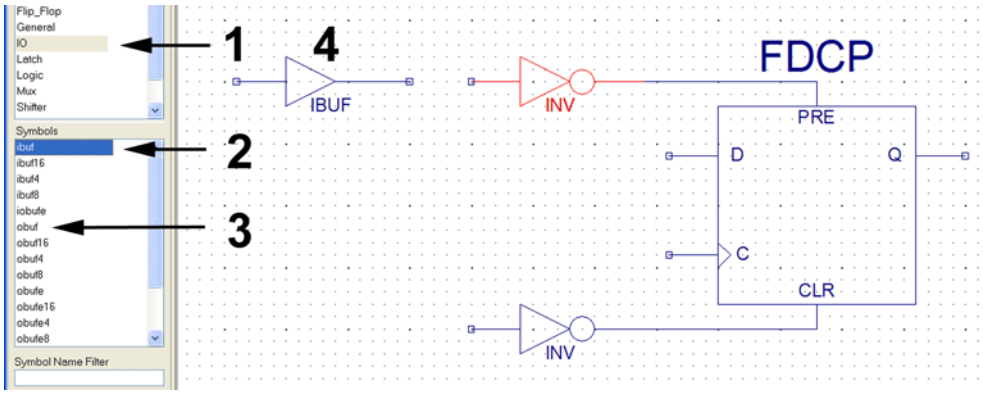


Abb. 15 Puffer einbauen. 1 - diese Funktionselemente finden wir in der Kategorie IO. 2 - der Input Buffer heißt ibuf. 3 - der Output Buffer heißt obuf. 4 - ein ausgewählter Input Buffer.

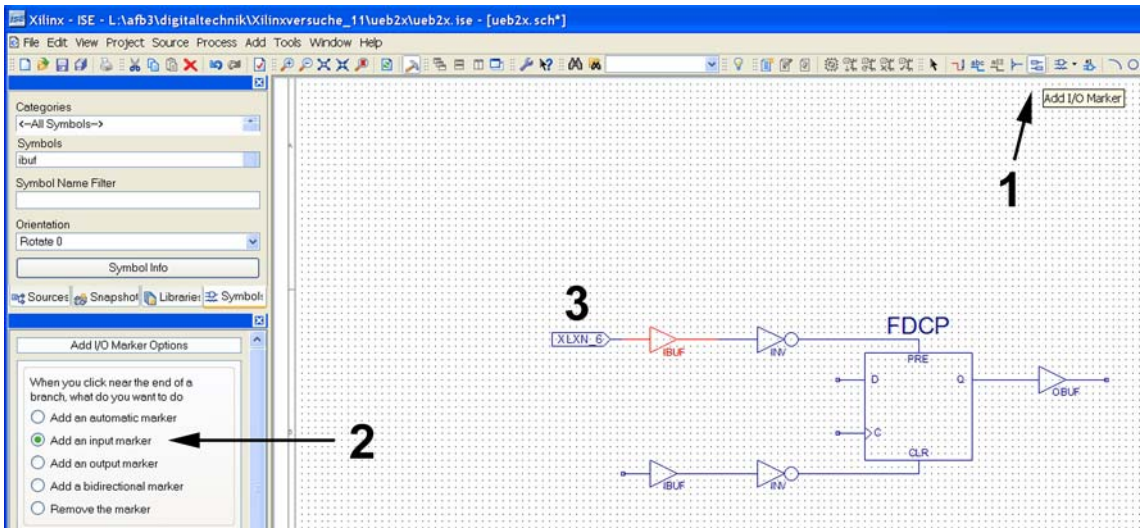


Abb. 16 Jetzt werden die Marker nachgesetzt. 1 - Funktionsauswahl. 2 - Auswahl der Art des Markers. 3 - ein Input Marker, der bereits angedockt wurde.

Die Signalbezeichner mit XLXN sind eine Eigenart des Entwicklungssystems. Sie werden automatisch vergeben. Auf Dauer kommt man damit nicht zurecht. Wir sollten schon Bezeichner haben, die man sich merken kann. Hierzu müssen wir die Signale umbenennen.

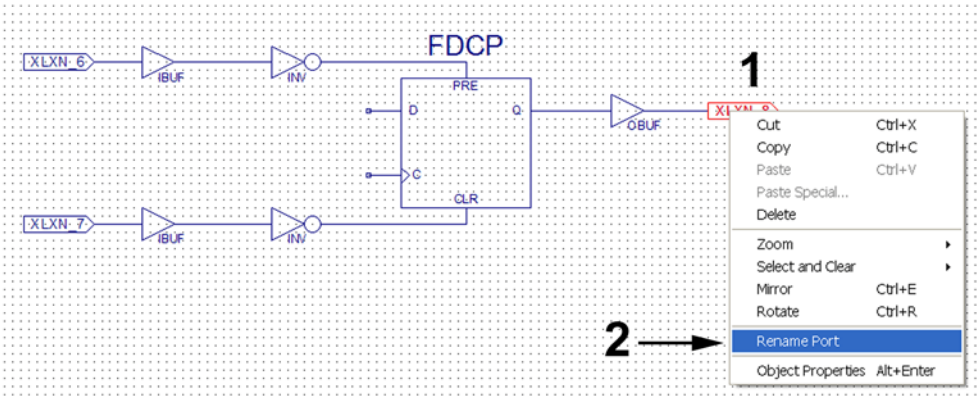


Abb. 17 Den Signalbezeichner in einem Marker ändern. 1 - Marker auswählen. Dann rechte Maustaste. 2 - wir benötigen die Funktion *Rename Port*.

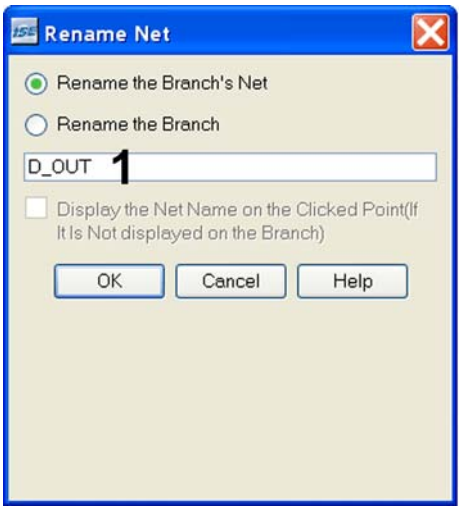


Abb. 18 Die Umbenennung. 1 - hier wird der neue Name eingegeben.

Die Schaltung hat noch offene Eingänge. Die darf es aber nicht geben. Also sind sie mit Festwerten zu beschalten.

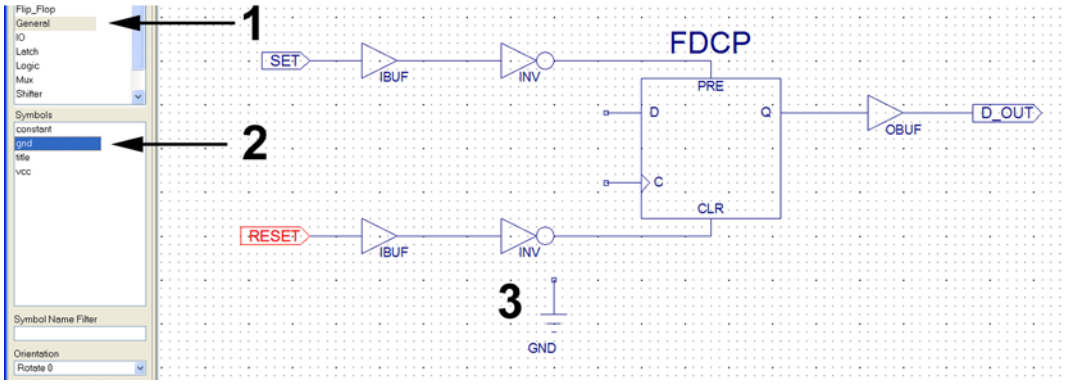


Abb. 19 Beschalten mit Festwerten. 1 die Festwerte finden wir in der Kategorie *General*. 2 - *gnd* = Low, *vcc* = High. 3 - ein ausgewählter Festwert Low.

Die Low-Festwerte (*gnd*) lassen sich direkt andocken, die High-Festwerte (*vcc*) nicht. Wenn das Andocken nicht klappt oder zu unschön aussieht, sind Drahtverbindungen zu ziehen. Zumeist genügt es, nur die Anschlußpunkte auszuwählen. Das System kann die Leitungen automatisch verlegen (Autorouting). Wenn es nicht um Schönheit geht, ist das in vielen Fällen ausreichend.

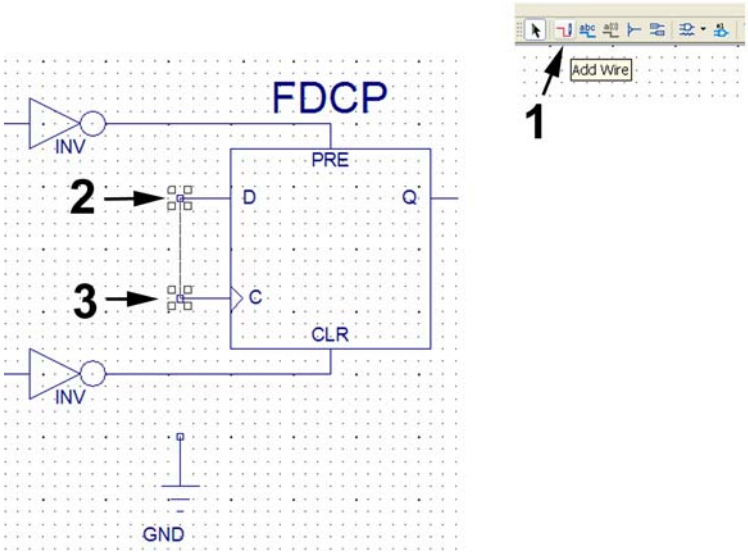


Abb. 20 Einen Draht ziehen. 1 - Funktionsauswahl. 2 - den ersten Anschluß anklicken. 3 - Verbindung zum nächsten Anschluß ziehen und dort klicken. Daß ein Anschluß richtig ausgewählt wurde, ist an den typischen vier kleinen Quadraten zu erkennen.

Die erste Teilschaltung ist fertig. Sie ist aber noch mit einem Toggle-Flipflop zu ergänzen.

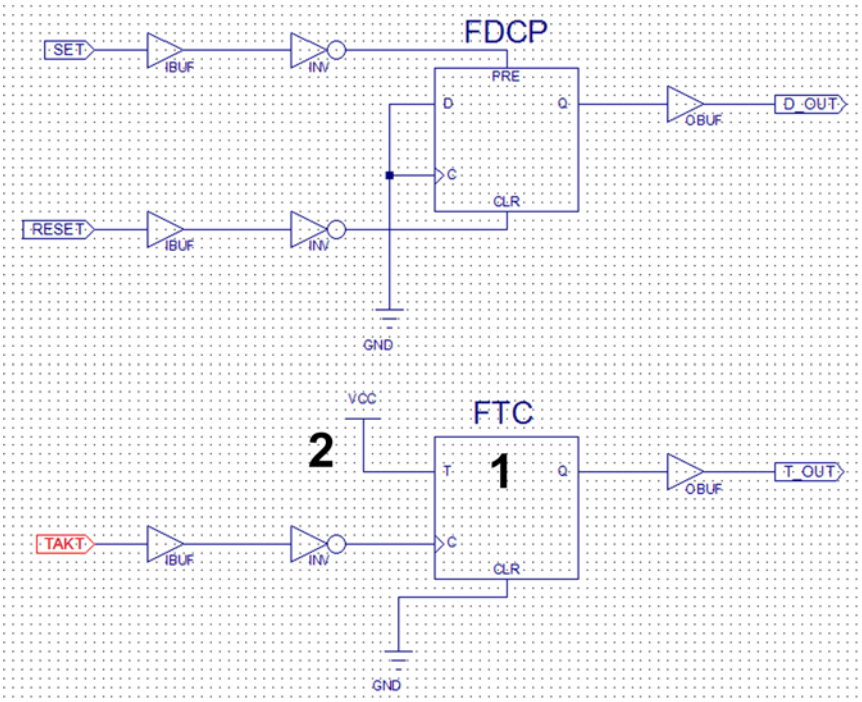


Abb. 21 Die fertig eingegebene Schaltung. 1 - in der Kategorie Flip-Flop finden wir ein passendes Toggle-Flipflop ftc. 2 - der Löscheingang muß inaktiv gehalten werden (Low), der Toggle-Eingang aktiv (High). Das Taktsignal kommt von der Taste. Deshalb ist es zu invertieren.

Jetzt kann die Schaltung implementiert werden.

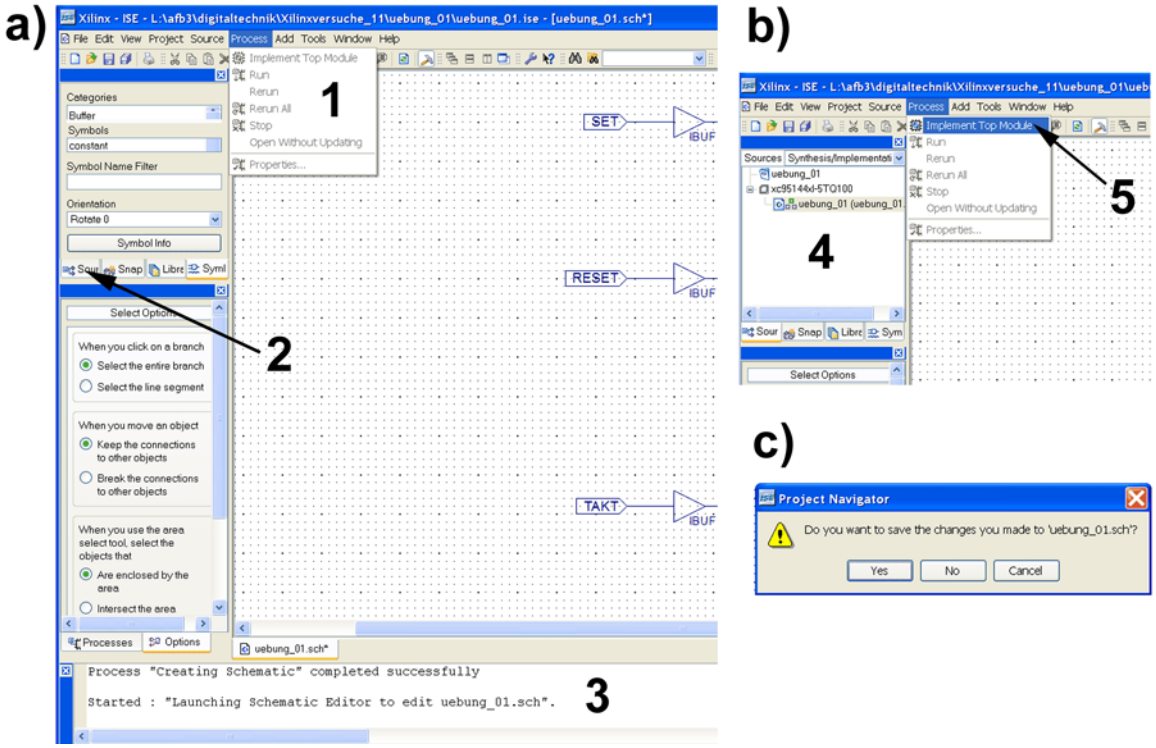


Abb. 22 Implementierung einer eingegebenen Schaltung. 1 - die Funktion heißt *Implement Top Module*. Sie ist im Process-Menü zu finden. Im Betriebszustand a) kann man sie aber nicht aufrufen. 2 - deshalb müssen wir den Source-Dialog in den Vordergrund bringen (b). 3 - Einzelheiten des Implementierungsablaufs werden im Konsolfenster angezeigt. 4 - der Source-Dialog. 5 - jetzt läßt sich die Implementierung auslösen. c) diese Aufforderung sollte bestätigt werden...

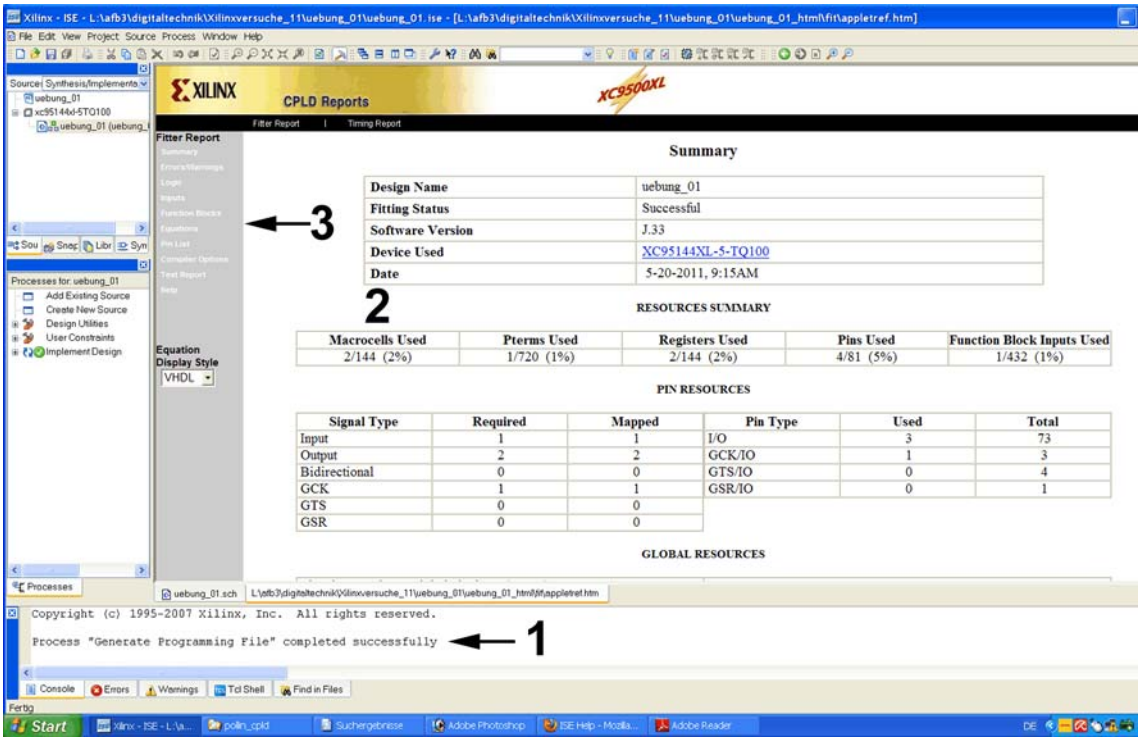


Abb. 23 So muß es am Ende aussehen. 1 - die entscheidende Erfolgsaussage. 2 - Es wurden zwei Makrozellen verbraucht. Wir haben zwei Flipflops eingesetzt. Deshalb ist es o.k. 3 - hier können detailliertere Berichte angefordert werden, darunter die Booleschen Gleichungen und die Anschlußbelegungen.

Das Entwicklungssystem belegt die Schaltkreisanschlüsse automatisch. Auf der Versuchsplattform sind aber die Tasten, LEDs usw. auf bestimmte Anschlüsse geführt. Damit alles zusammenpaßt, braucht man eine weitere Quelldatei, das sog. Constraints File. Zunächst müssen wir aber noch einige weitere Einstellungen vornehmen. Sie dienen dazu, die Versuchschaltung betriebssicher zu machen.

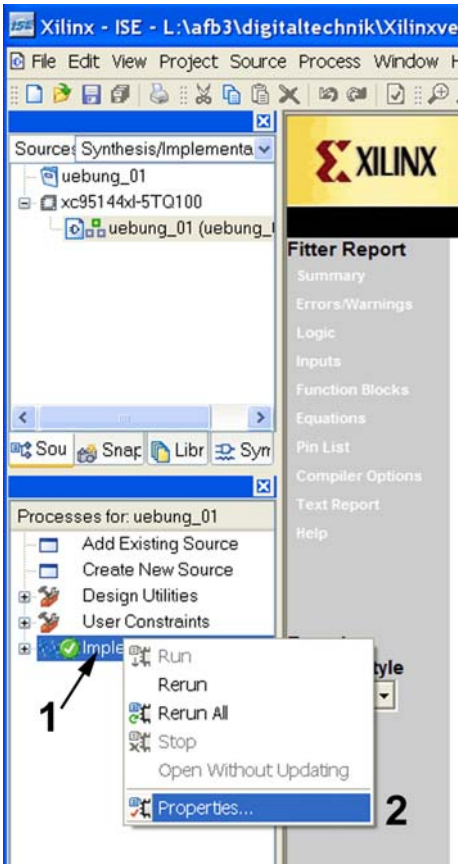


Abb. 24 Die Dialoge Sources und Processes sind in den Vordergrund zu holen. 1 - mit der rechten Maustaste auf *Implement Design* klicken. 2 - auf *Properties* klicken.

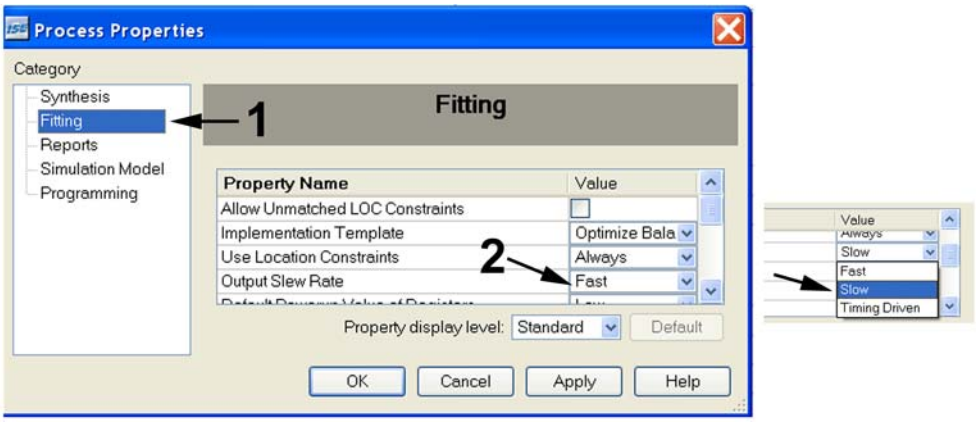


Abb. 25 Verringerung der Flankensteilheit (um das Übersprechen (Crosstalk) gering zu halten). 1 - Vorauswahl, 2 - Flankensteilheit (Slew Rate). Auf Slow stellen.

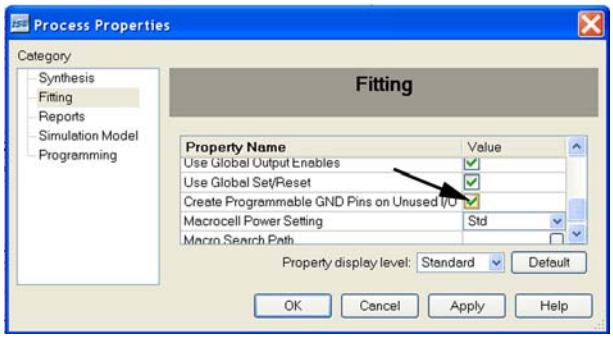


Abb. 26 Was tun mit den ungenutzten Schaltkreisanschlüssen? – Verbinden wir sie vorsichtshalber mit Masse (Programmable GND Pins on Unused I/O).

Nun können wir die Anschlüsse zuweisen. Dazu brauchen wir die neue Quelldatei (Constraints File).

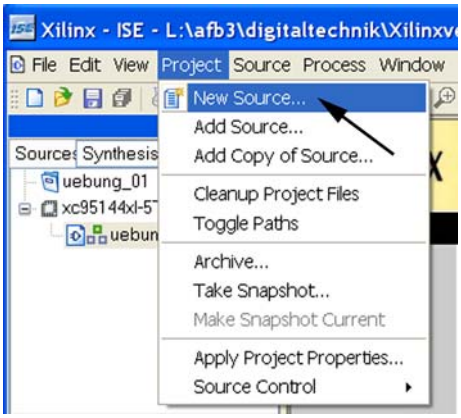


Abb. 27 Auswahl des Quelldateiassistenten über Project – New Source.

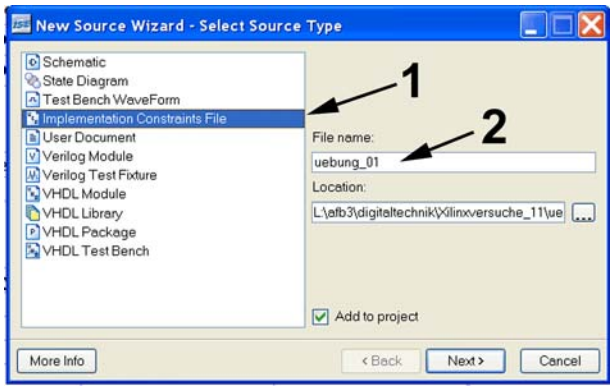


Abb. 28 Die Datei wird erzeugt. 1 - wir brauchen eine Constraints-Datei. 2 - Eingabe des Dateinamens (beliebig; die Dateiendung wird automatisch gebildet).

Hinweis: Eine Übersicht über alle hier genutzten Anschlüsse befindet sich am Ende dieser Versuchsanleitung.

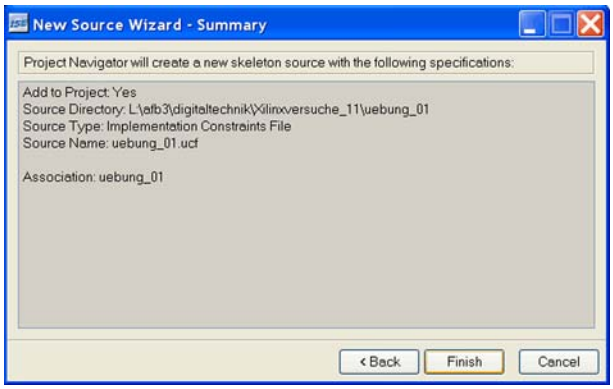


Abb. 29 Anschließend ist dieser Dialog zu durchlaufen.

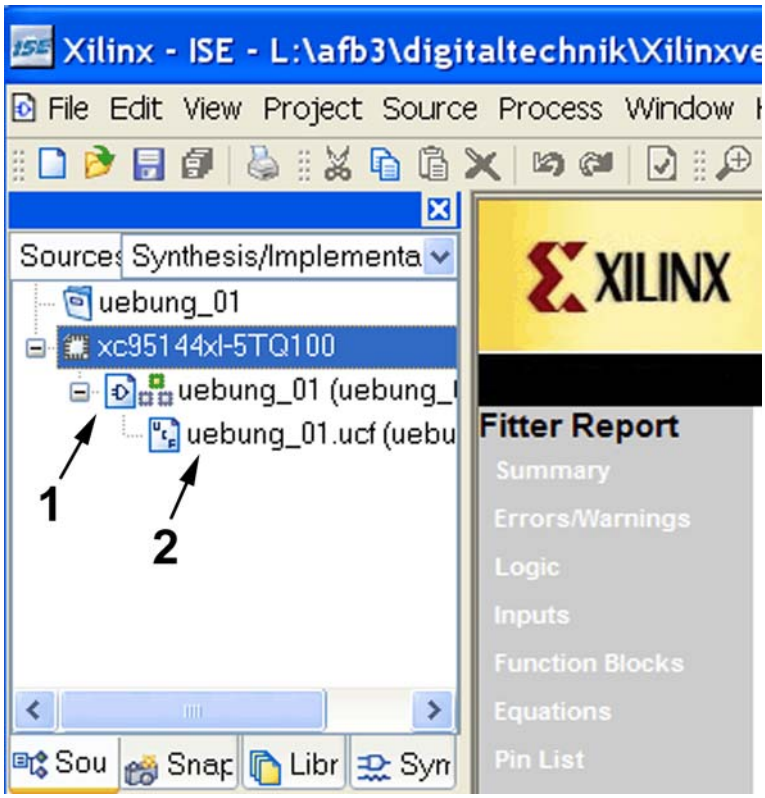


Abb. 30 Die neue Datei erscheint im Quellenverzeichnis. 1 - hier klicken; 2 - die neue Datei.

Wenn wir auf die Datei doppelklicken, sollte sie geöffnet werden, und zwar mit einem passenden Editor. Wir arbeiten hier mit dem sog. *Floorplan Editor*. Wird statt dessen ein anderes Programm gestartet, sind die Einstellungen zu ändern. Wird nichts gestartet, ist der Floorplaneditor im Programmstartmenü (Windows) zu aktivieren. Das Programm heißt *Pace*.

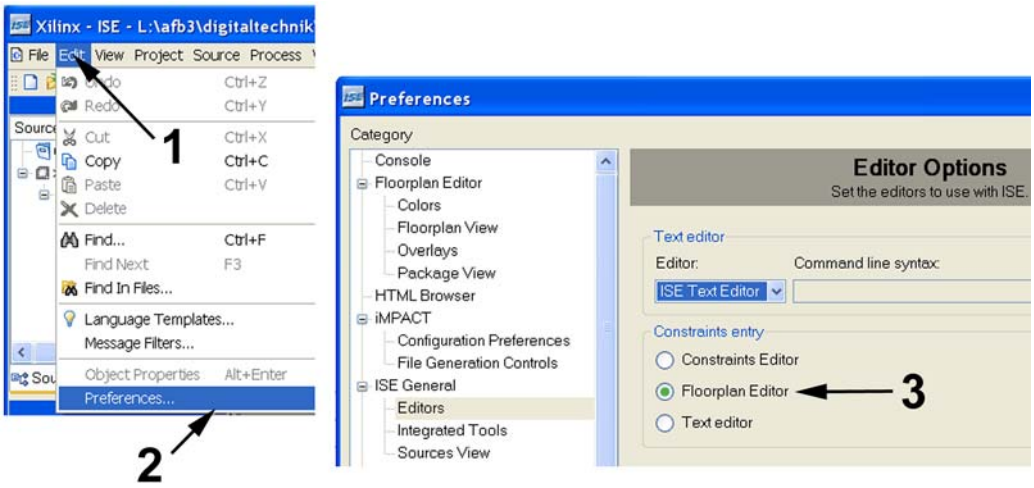


Abb. 31 So wird – bei Bedarf – der richtige Editor ausgewählt. 1 - Edit-Dalog. 2 - Vorzugseinstellungen (Preferences). 3 - Auswahl des Floorplan-Editors.

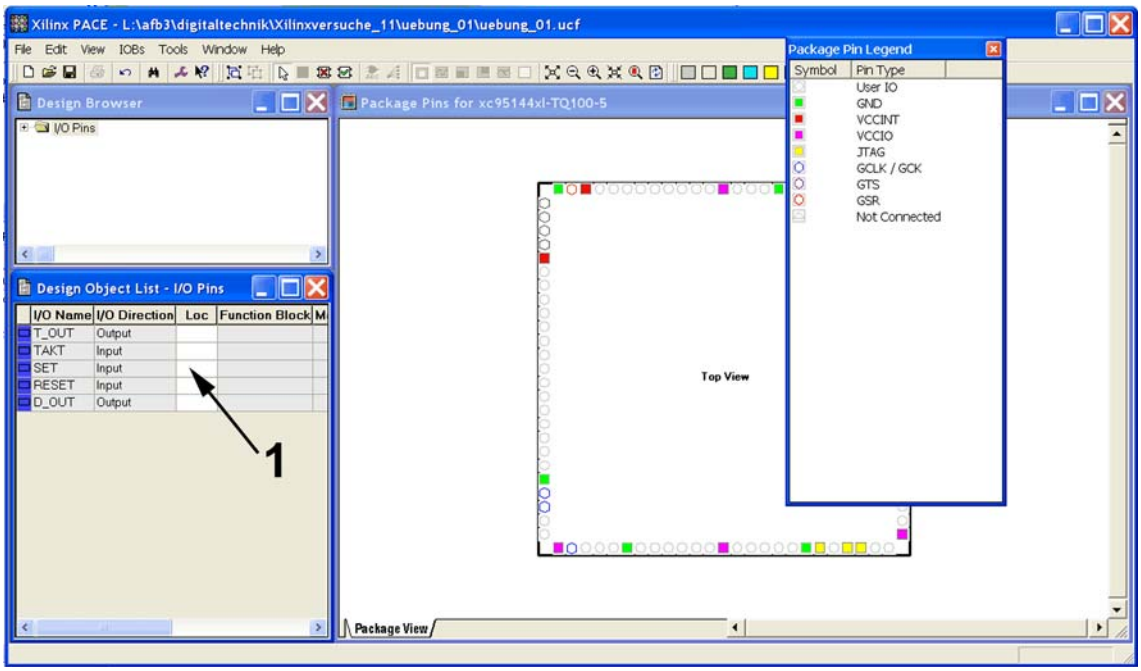


Abb. 32 Der geöffnete Floorplan-Editor. 1 - hier sind die Pin-Nummern einzutragen.

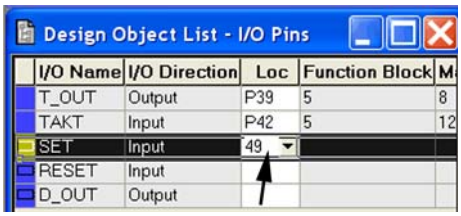


Abb. 33 Die Pin-Nummern werden eingetippt. Einzelheiten in der folgenden Abb. Nach der Eingabe ist die Datei zu schließen, wobei die neuen Inhalte zu speichern sind.

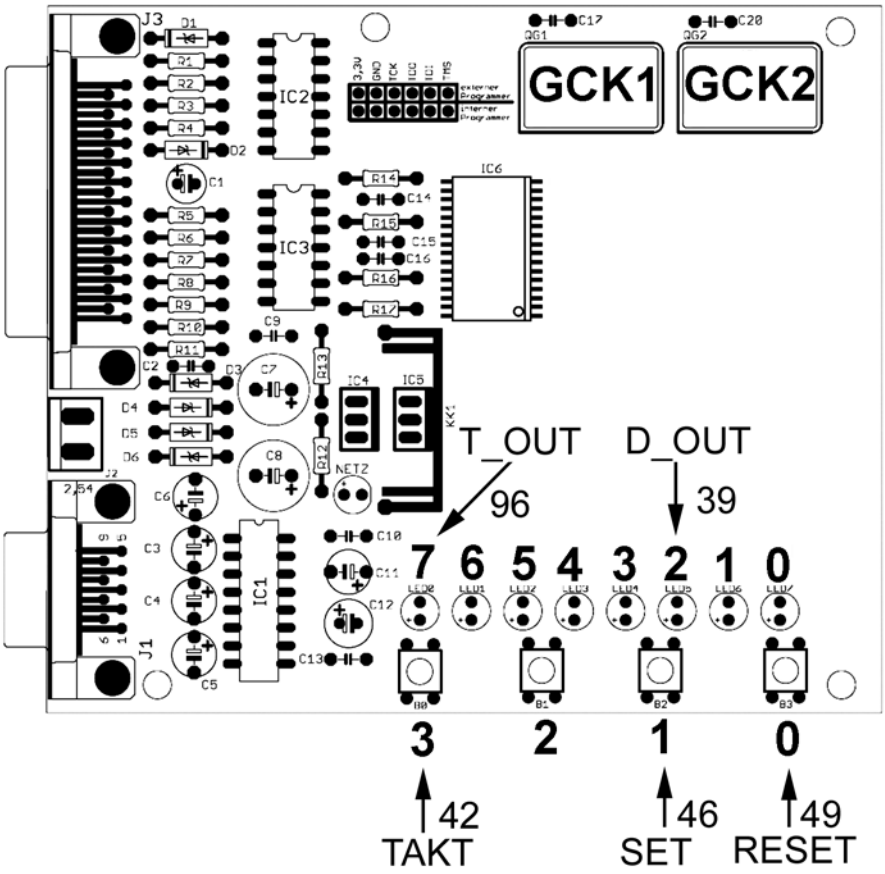


Abb. 34 Eine zweckmäßige Anschlußbelegung.

Wurden die Anschlußbelegungen eingegeben, ist die Implementierung erneut zu starten. Nach deren Durchlauf stehen die Programmierdaten bereit. Jetzt kann das Programmierprogramm aufgerufen werden. Es heißt *IMPACT*. Versuchsplattform einschalten. Ggf. Verbindung zur Parallelschnittstelle des PC herstellen.

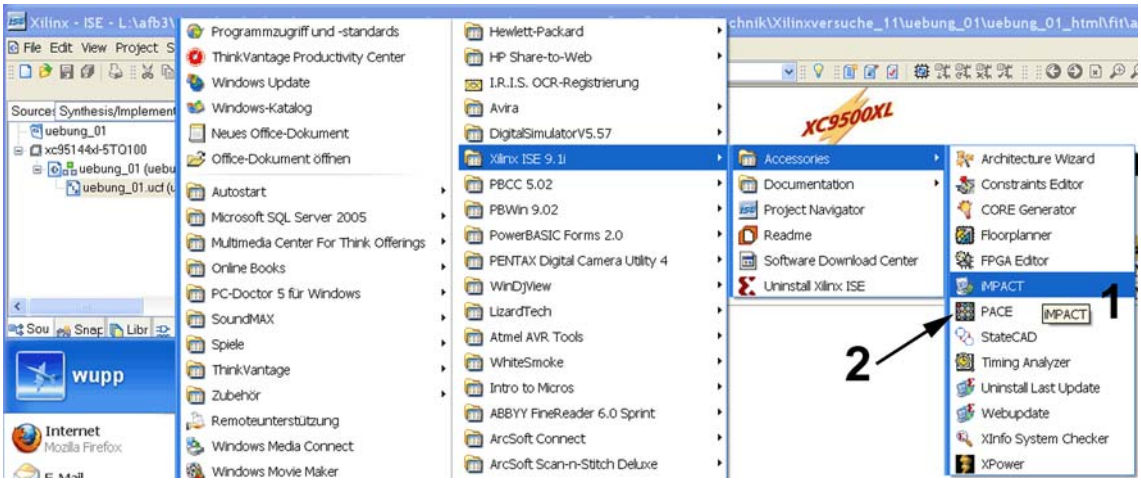


Abb. 35 Programmauswahl im Windows-Startmenü. 1 - der Programmierer (IMPACT). 2 - der Floorplaneditor (PACE).

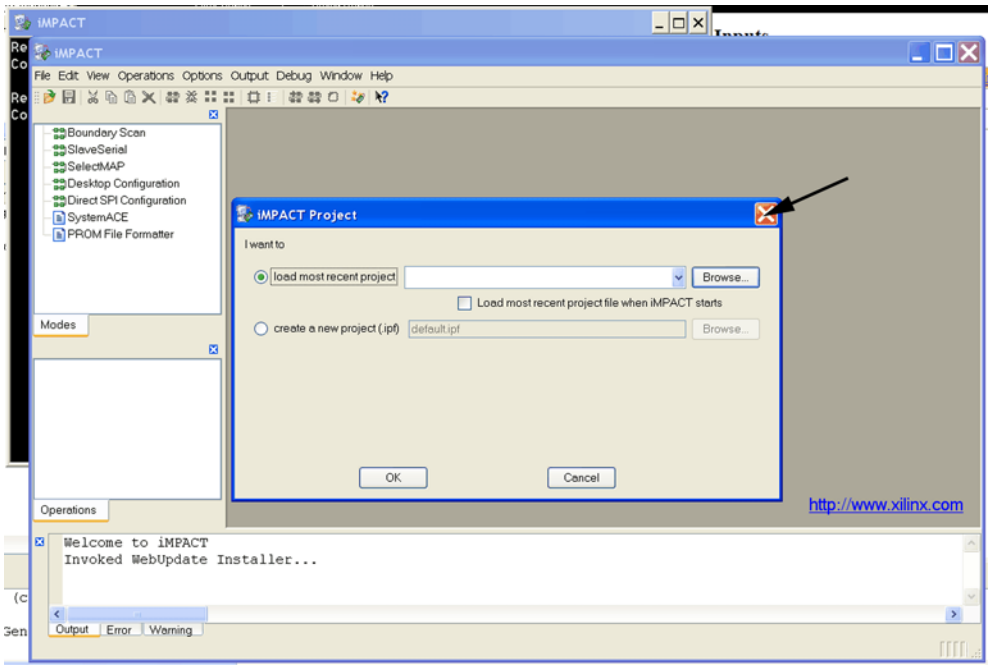


Abb. 36 Die IMPACT-Projekte haben mit den ISE-Projekten nichts zu tun. Brauchen wir hier nicht. Fenster schließen.

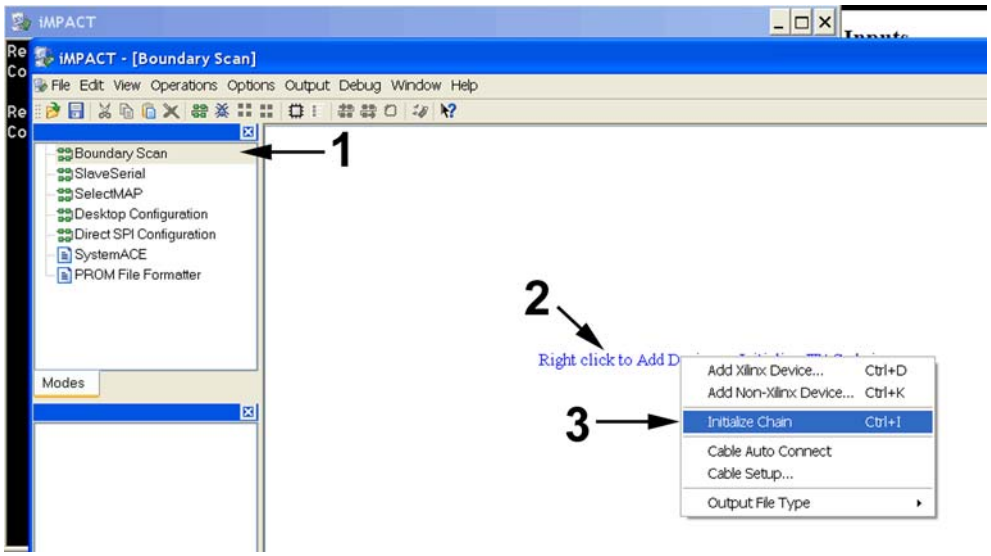


Abb. 37 So wird die Programmierschnittstelle angesprochen. 1 - Boundary Scan auswählen. 2 - mit der rechten Maustaste draufklicken. 3 - Funktion Initialize Chain ausführen.

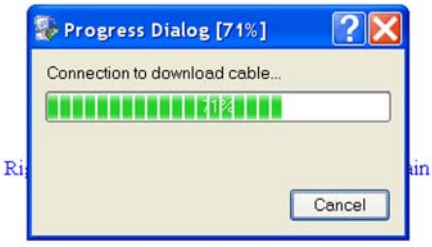


Abb. 38 Der Programmierer versucht, die Verbindung herzustellen.

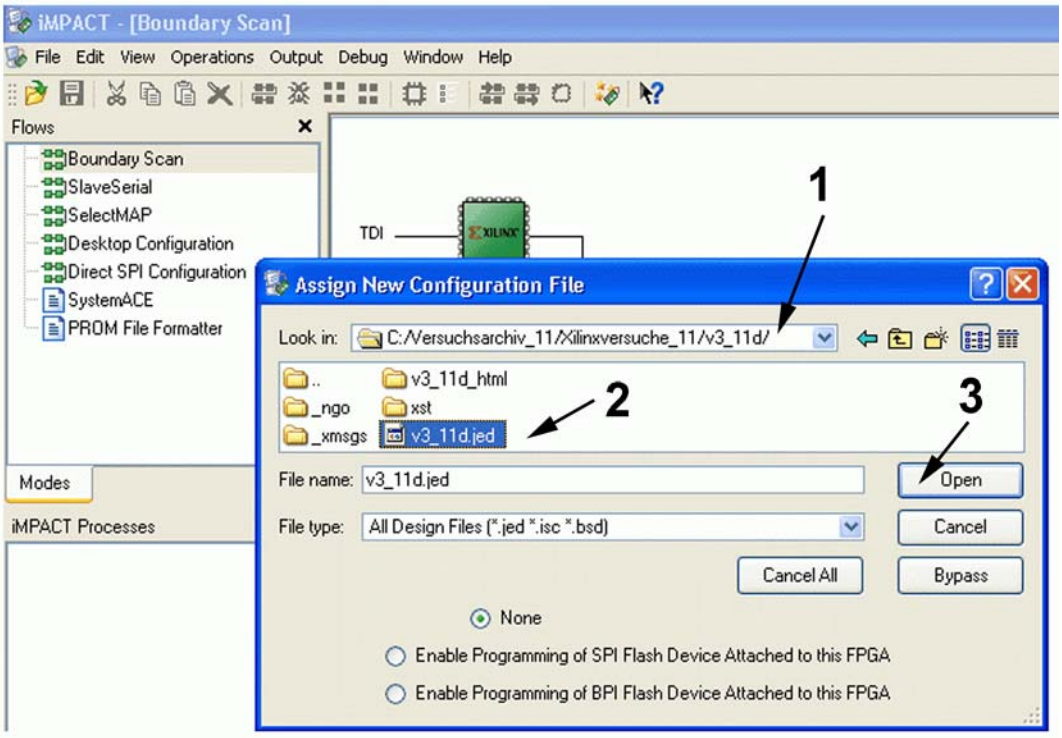


Abb. 39 Wenn die Verbindung in Ordnung ist, erscheinen ein grünes Schaltkreissymbol sowie ein Dateidialog. Dann muß die Programmierdatei gesucht werden. Die Dateiendung heißt *.jed* (Jedec File). 1 - Verzeichnis des Projekts suchen. 2 - Jedec-Datei auswählen. 3 - weiter...

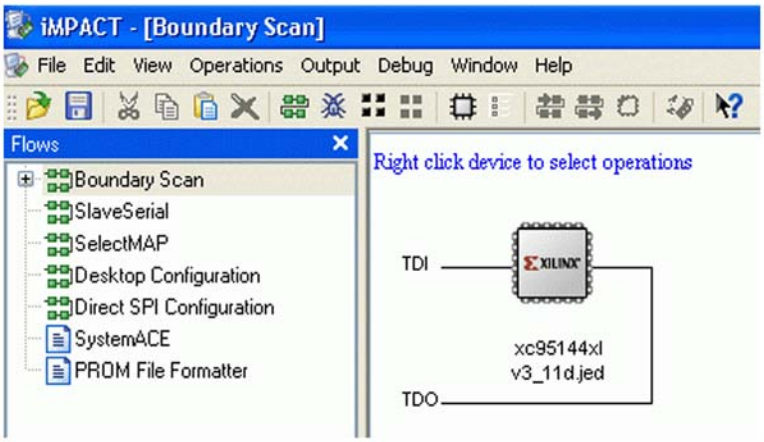


Abb. 40 Ist alles richtig ausgewählt, wird das Schaltkreissymbol aluminiumfarben. Jetzt weiter mit Dienst nach Vorschrift ...

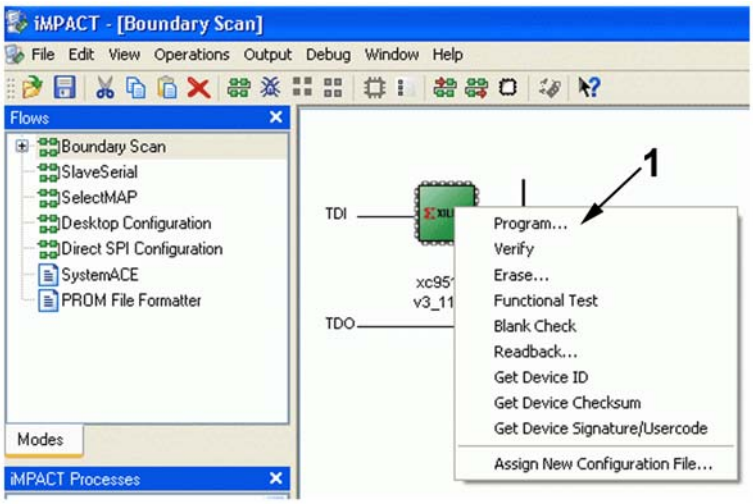


Abb. 41 Mit der rechten Maustaste auf das Schaltkreissymbol klicken. Symbol wird wieder grün. 1 - dann auf *Program* klicken.

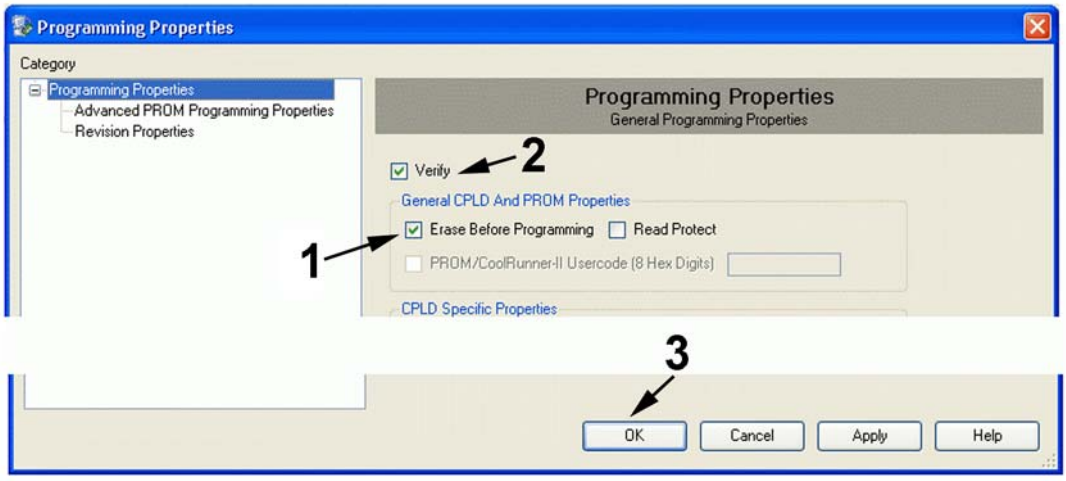


Abb. 42 Diese Eigenschaften der Programmierfunktion brauchen wir. 1 - vor dem Programmieren löschen. 2 - nach dem Programmieren überprüfen, ob auch alles stimmt. 3 - weiter ...

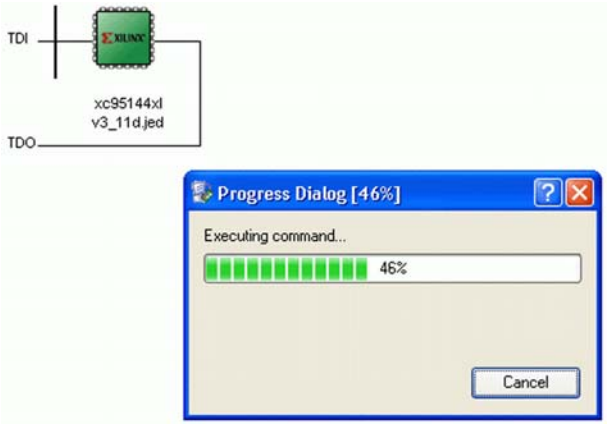


Abb. 43 Die Programmierung läuft.

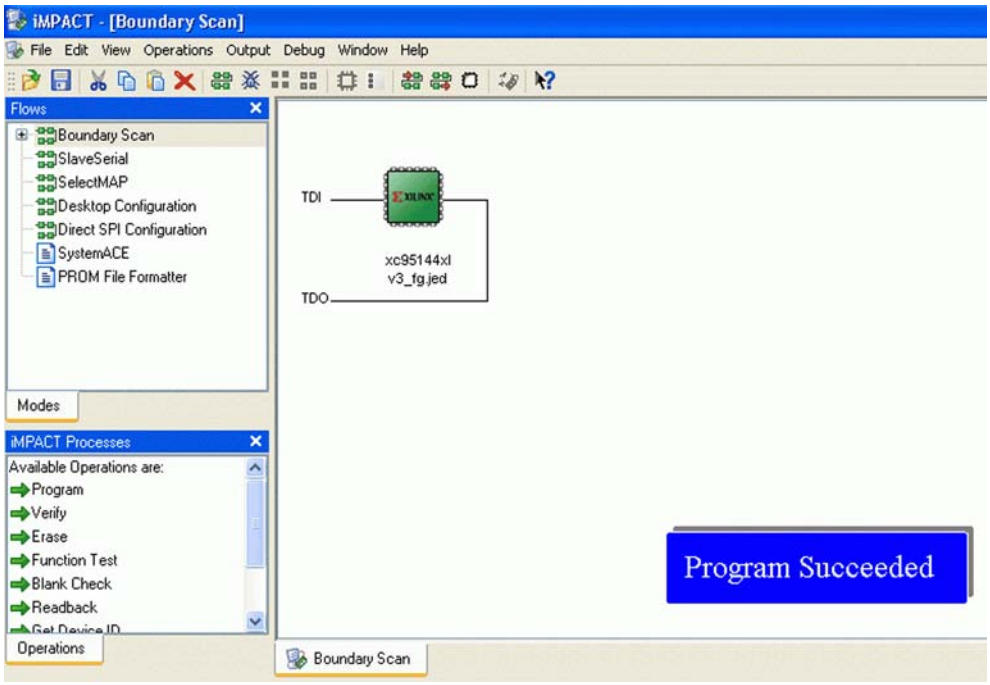


Abb. 44 Am Ende sollte es so aussehen. Nun kann die Schaltung ausprobiert werden. Das IMPACT-Programm nicht schließen, sondern nur minimieren. Bei den nachfolgenden Versuchen genügt dann ein erneutes Anklicken.

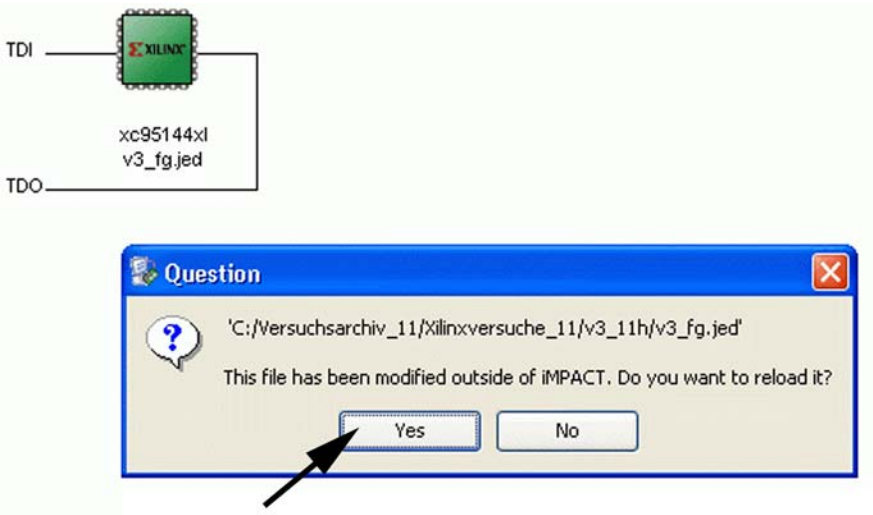


Abb. 45 Wird *IMPACT* wieder aktiv, so erscheint diese Anfrage. Sie muß mit *Yes* bestätigt werden, da die Programmierdatei vom Entwicklungssystem neu erzeugt wurde. Dann wird erneut programmiert. Erscheint die Anfrage nicht, ist etwas falsch gelaufen...

Aufgabe 2: Passende Taktfrequenzen herstellen

Das Toggle-Flipflop wird nicht immer richtig funktionieren. Das liegt daran, daß die Taste prellt. Man kann Tasten entprellen, indem man sie mit einem Takt von einigen 10 bis wenigen hundert Hz synchronisiert. Auch brauchen wir einen langsamen Takt, um beispielsweise ein Lauflicht vorführen zu können. Die Versuchsplattform hat aber nur einen Grundtakt von 16 MHz. Niedrigere Taktfrequenzen sollen erzeugt werden, indem wir diesen Takt im Verhältnis $1:2^{24} = 16\,777\,216$ teilen. Hierzu bauen wir einen 24-Bit-Zähler auf. Passende Zähler finden wir in der Kategorie *Counter*.

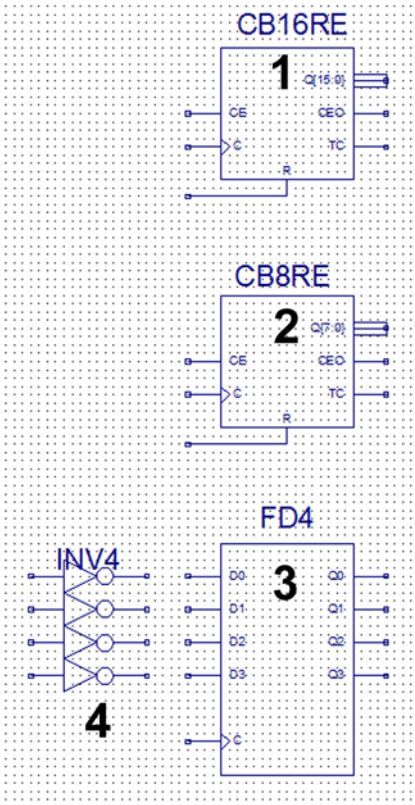


Abb. 46 Erst einmal die Funktionselemente bereitlegen. Die bisherige Schaltung wird gelöscht. 1 - 16-Bit-Zähler. 2 - 8-Bit-Zähler. 3 - wir entprellen gleich alle vier Tasten der Versuchsplattform (sozusagen auf Vorrat). In der Kategorie *Flip-Flop* finden wir ein Register mit vier D-Flipflops. 4 - ergänzend dazu aus der Kategorie *Logic* Negatoren im Viererpack (inv4).

Hinweis: Wer schneller zum Erfolgserlebnis kommen will, kann die Tasten und das Entprell-Register zunächst weglassen. Diese Teilschaltung brauchen wir erst für die Aufgabe 4.

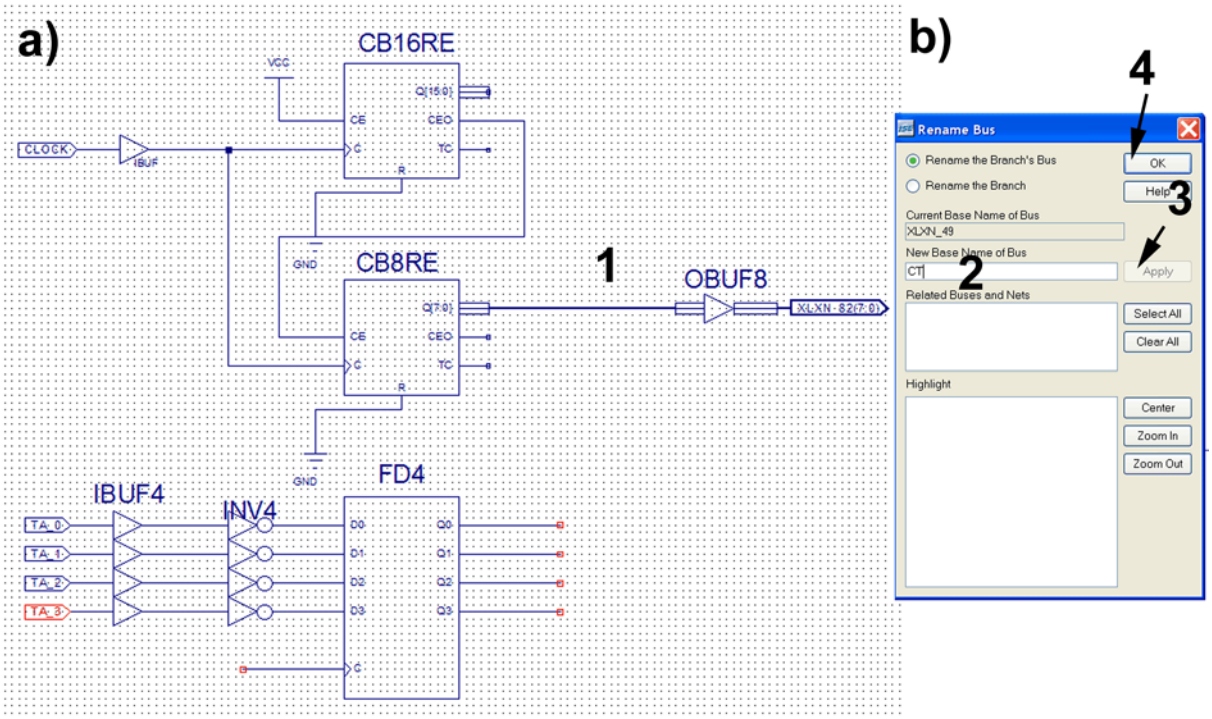


Abb. 47 Die (fast) fertige Schaltung. a) die Ausgangssignale der höchstwertigen acht Zählerstellen sollen auf die LEDs geführt werden. Auch die Puffer gibt es in Vierfach- und Achtfachausführung. Einige Signale müssen noch benannt werden. 1 - etwas Neues; ein Bus. b) so erhält der Bus seinen Namen. Auf den Bus klicken (wird rot). Rechte Maustaste. Dann auf *Rename Selected Bus* klicken. 2 - Namen eingeben. 3, 4 - in dieser Folge anklicken.

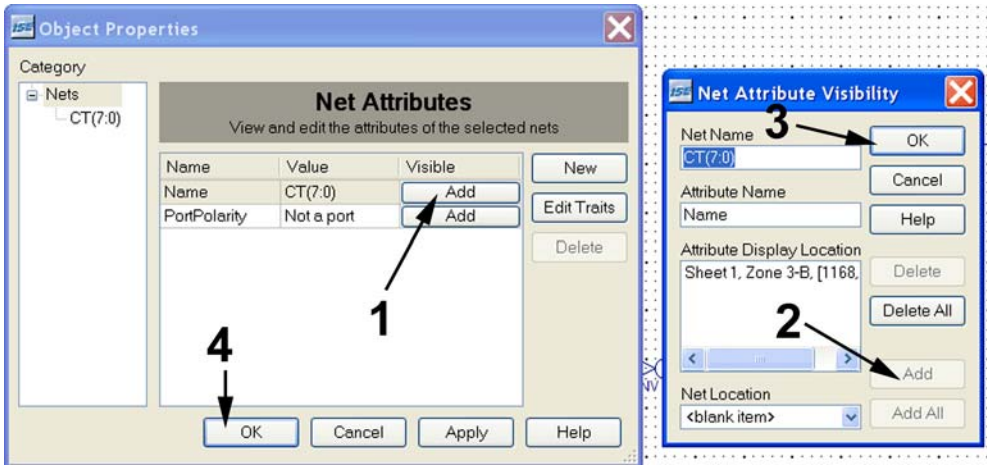


Abb. 48 Der Name ist aber noch nicht zu sehen. Wie machen wir ihn sichtbar? Den Bus nochmals anklicken. Rechte Maustaste. Dann auf *Object Properties* klicken. Dann in der Reihenfolge 1 – 2 – 3 – 4 klicken.

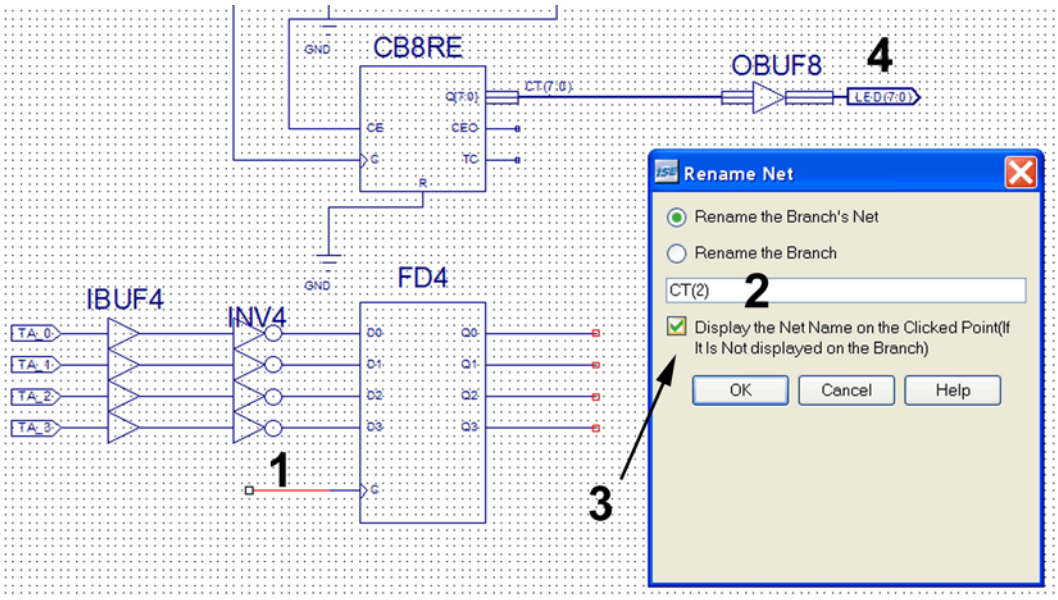


Abb. 49 Jetzt kann der Takteingang der Entprellflipflops benannt werden. Wir schließen ihn an den dritten Ausgang des Zählers an. 1 - Leitung anklicken. Rechte Maustaste, dann *Rename Selected Net*. 2 - Namen eintragen. 3 - dieses Häkchen setzen, damit der Name im Schaltbild erscheint. 4 - die Ausgänge werden ebenso benannt wie der Bus (anklicken, dann *Rename Port*).

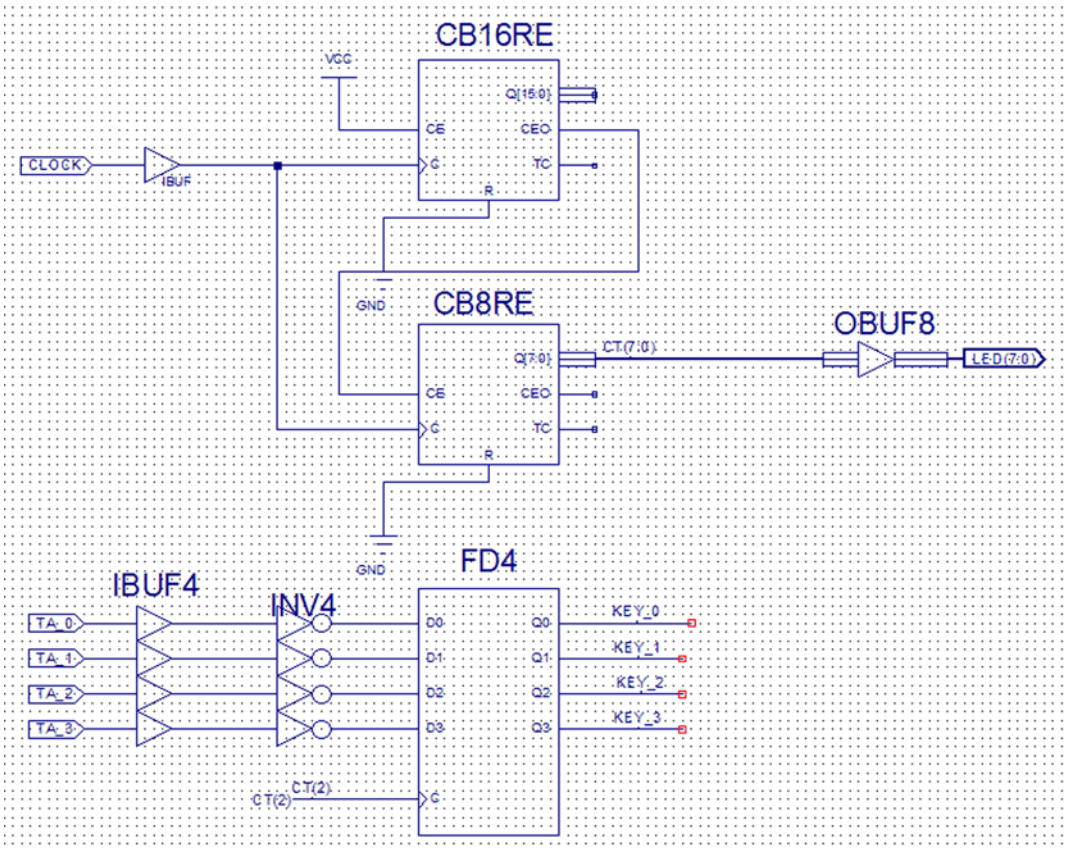


Abb. 50 Die fertige Schaltung. Auch die entprellten Tastensignale haben Namen erhalten.

Jetzt die Constraints-Datei aufrufen und alle Pinnummern löschen. Die Datei wieder schließen (speichern). Nun alles implementieren.

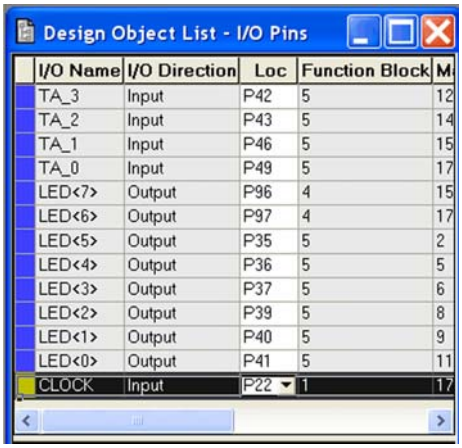
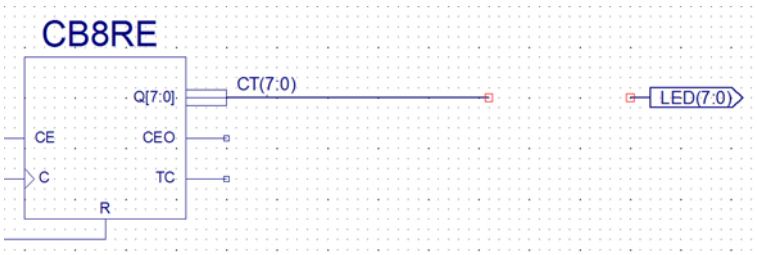


Abb. 51 Jetzt wird die Constraints-Datei wieder aufgerufen, um die neuen Pinnummern einzugeben.

Constraints-Datei schließen, nocmlasi implementieren und dann programmieren. An den LEDs muß das typische Zählverhalten zu beobachten sein

Aufgabe 3: Lauflicht
Die Schaltung wird um ein Schieberegister ergänzt, dessen Ausgänge an die LEDs angeschlossen werden. Um die Benennungen zu erhalten, wird zunächst der *Output Buffer* gelöscht.



Durch Löschen des Output Buffers wurde der Signalweg getrennt, so daß nunmehr das Schieberegister eingebaut werden kann. Wir verschalten es als selbsteinschwingenden Ringzähler. In der Kategorie *Shift_Register* finden wir passende Typen, in der Kategorie *Logic* das NOR-Gatter mit sieben Eingängen.

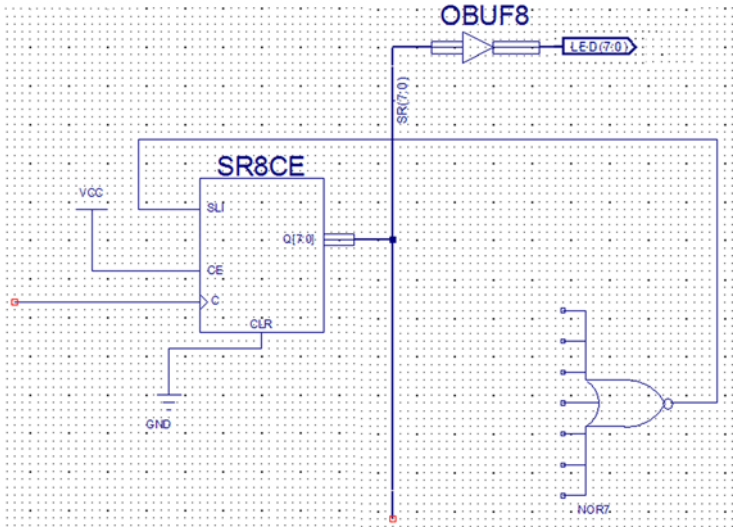


Abb. 52 Die Grundschaltung. Das Schieberegister hat einen Busausgang. Er ist hier schon benannt und über einen Output Buffer mit den LED-Ausgängen verbunden worden.

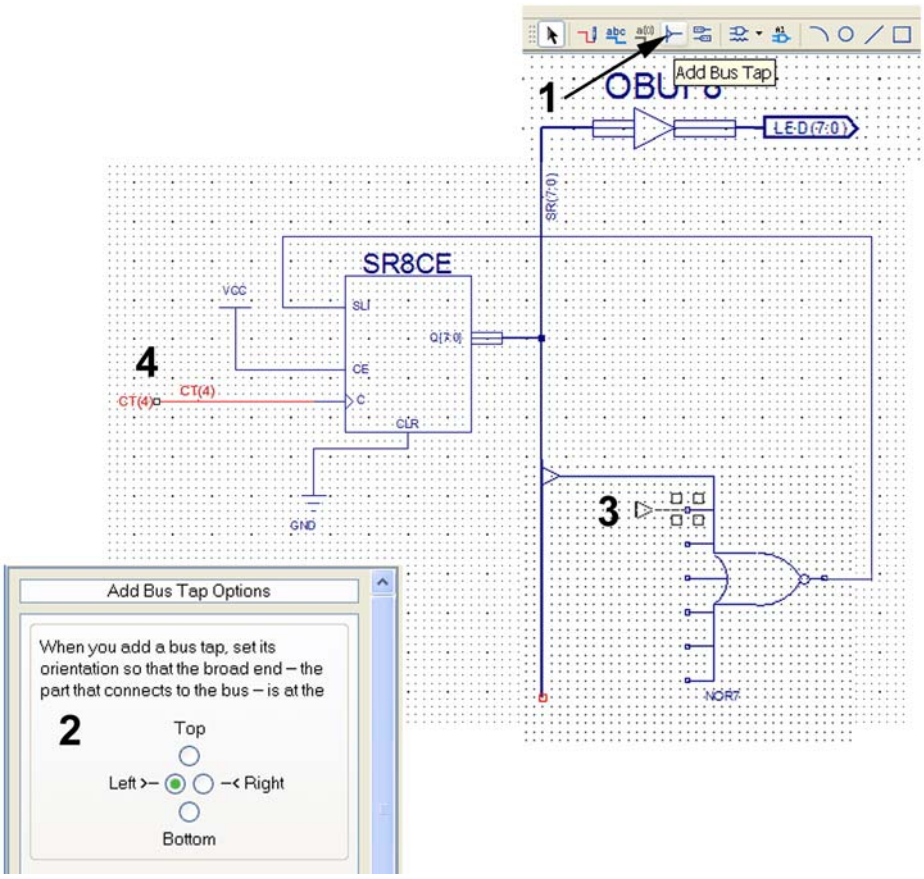


Abb. 53 Jetzt wird der Bus mit dem NOR-Gatter verbunden. Die einzelnen Leitungen werden über Anzapfungen, sog. *Bus Taps*, angeschlossen. 1 - Funktionsauswahl zum Hinzufügen von *Bus Taps*. 2 - hier wird die Orientierung der Anzapfung gewählt. Mitdenken! 3 - so wird ein *Bus Tap* angefügt. Es dockt automatisch an den Bus an. 4 - der Takteinang wurde bereits benannt und auf diese Weise mit einem hinreichend niederfrequenten Takt verbunden.

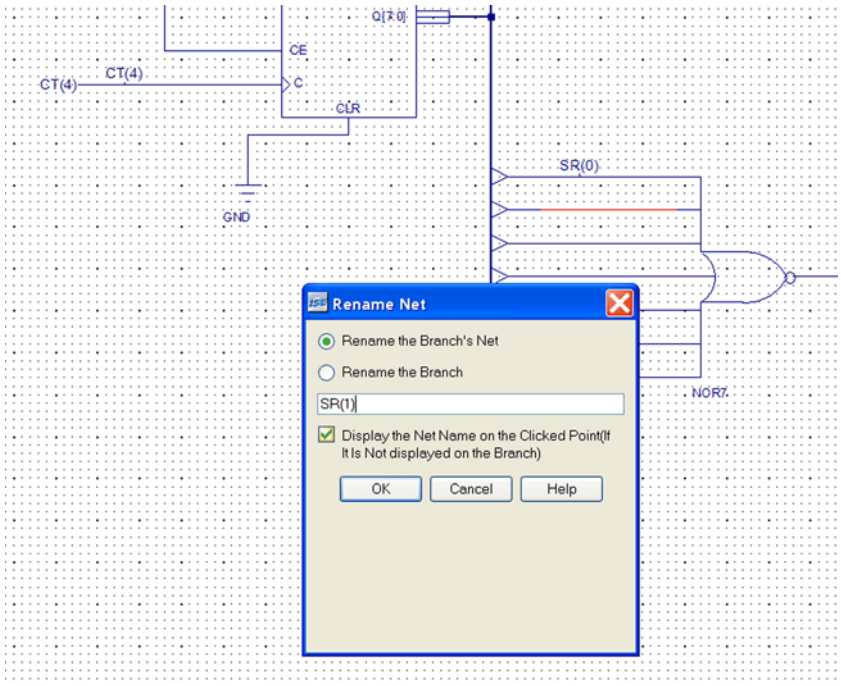


Abb. 54 Nun müssen die *Bus Taps* einzeln benannt werden – eine mühevolle Arbeit; es hilft alles nichts...

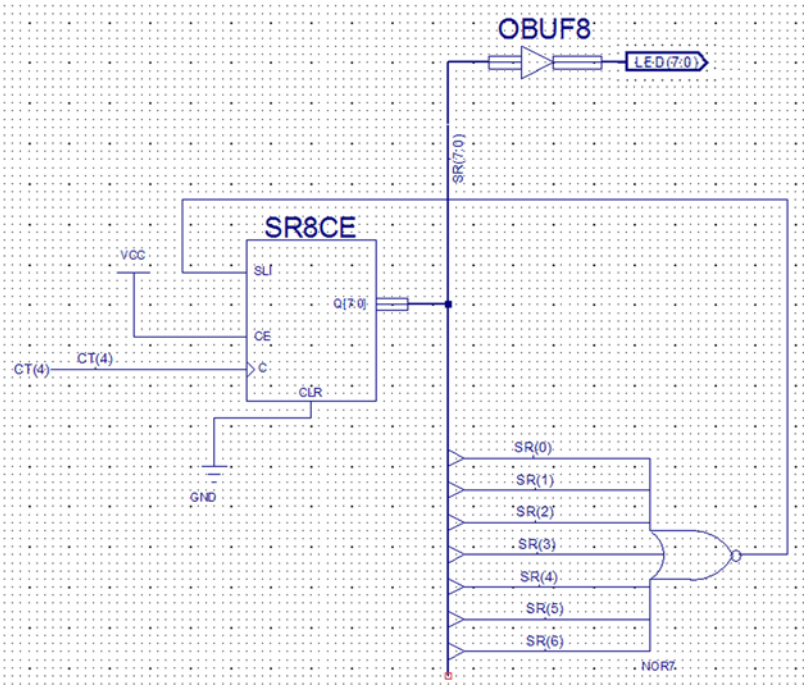


Abb. 55 Die fertige Schaltung. An der Constraints-Datei ist nichts zu ändern. Implementieren, programmieren und zusehen, ob es funktioniert.

Aufgabe 4: Zwei Binärzähler mit Handbetätigung und Anzeige

Als Endziel wollen wir einen Impulsmustergenerator als Prüfgenerator für serielle Schnittstellen bauen. Hierbei ist ein Prüfzeichen von acht Bits Länge einzugeben. Wir haben aber keine acht Kippschalter. Deshalb ordnen wir zwei Einstellzähler zu vier Bits an, deren Ausgänge auf die LEDs geführt und deren Takteingänge von zwei Tasten angesteuert werden. Der bishierige Schieberegisterentwurf wird hierzu gelöscht. Die LED-Ausgänge bleiben.

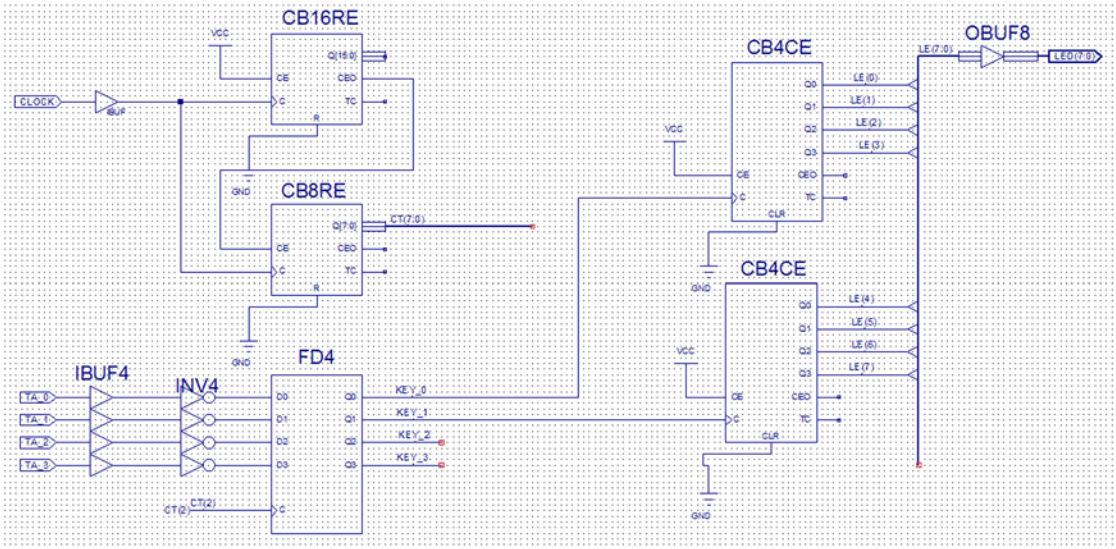


Abb. 56 Die fertige Schaltung. Ausprobieren...

Jetzt sollen – um den Bedienkomfort zu verbessern – zwei Siebensegmentanzeigen angeschlossen werden. Die Siebensegmentdecoder sind als Funktionselemente schon vorgefertigt. Sie sind als Kopien externer Quellen dem Projekt hinzuzufügen.

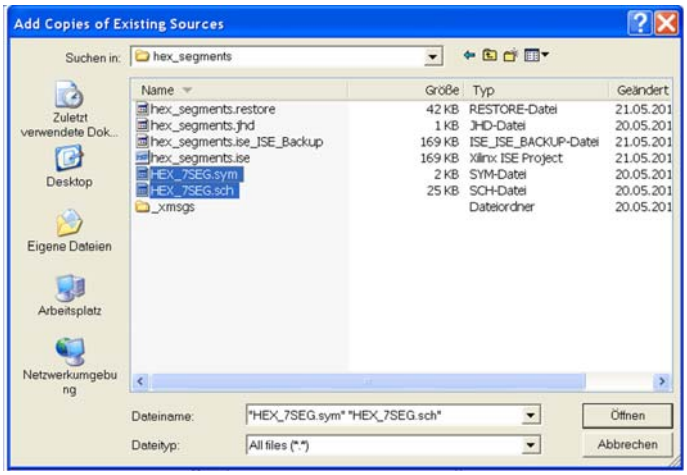


Abb. 57 Quellenauswahl über *Project – Add Copy of Source*. Wir brauchen den Schaltplan (.sch) und das Schaltsymbol (.sym). Also Datentyp auf *All files*.

Hinweis: Beide Dateien müssen den gleichen Namen haben, sonst funktioniert es später nicht (hier: HEX_7SEG.sch (Schaltplan) und HEX_7SEG.sym (Schaltsymbol).

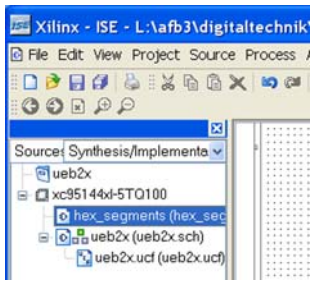


Abb. 58 Der hinzugefügte Schaltplan sollte jetzt im Verzeichnis der Quellen auftreten.

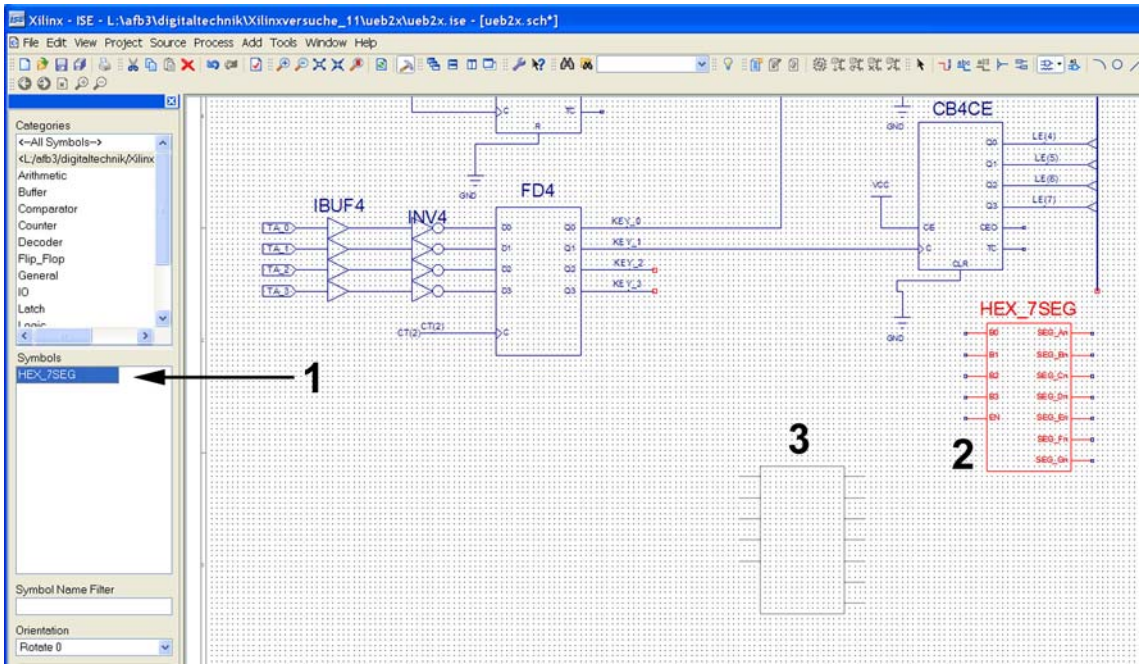


Abb. 59 Die Funktionselemente werden in die Schaltung eingebaut. 1 - Symbolauswahl. 2 - ein bereits eingefügtes Schaltsymbol. 3 - Schaltsymbol während des Einfügens.

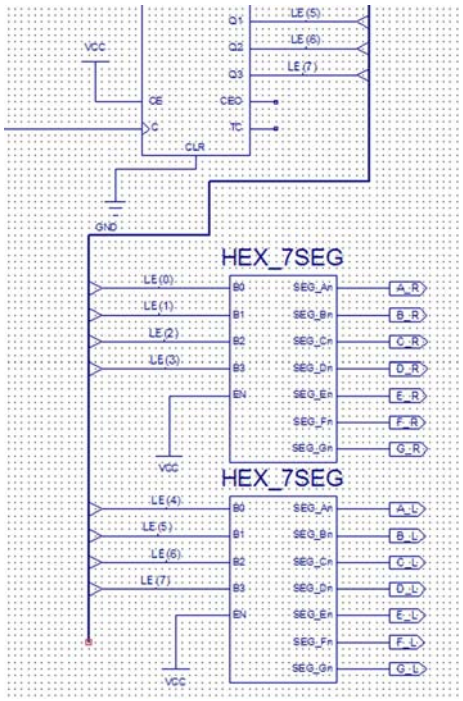


Abb. 60 Die fertige Siebensegmenterweiterung. R = rechte, L = linke Anzeige. Die Ausgabepuffer sind bereits in die Decoder eingebaut. Constraintsdatei gemäß Anschlußtabelle ergänzen. Ausprobieren...

Das Praktikum schließt mit zwei wahlweisen Aufgaben. Wir bauen entweder eine Stoppuhr (Aufgabe 5) oder einen Prüfgenerator für serielle Schnittstellen (Aufgabe 6). Beides nur provisorisch unter Verzicht auf letzte Feinheiten. *Näheres im Wahlfach Entwurf digitaler Schaltungen (EDS).*

Aufgabe 5: Stoppuhr
Die Stoppuhr soll im Raster von 0,1 s bis 9,9 s zählen. Betätigung: Taste 0: Zählen/Stop (Toggle-Funktion); Taste 3: löschen. Wir wechseln zunächst die Binärzähler gegen Dezimalzähler aus (Binärzähler löschen und Dezimalzähler CD4CE einfügen). Dann schalten wir beide zu einem zweistelligen Zähler zusammen. Das Zählen wird von einem Toggle-Flipflop über den Zählerlaubniseingang der niederwertigen Zählstelle gesteuert. Als Zähltakt wählen wir CT(3) oder CT(4). CT(3) ist das Ausgangssignalk eines insgesamt 20steilligen Binärzählers (16 + 4 Stellen), CT(4) das Ausgangssignakl eines 21stelligen. Mit CT(3) werden die 16 MHz durch 2^{20} geteilt (15,26 Hz), mit CT(4) durch 2^{21} (7,63 Hz). Das ergibt eine Taktperiode der Stoppuhr von 65 oder 130 ms.

Die Grundschaltung ist schnell aufgebaut, hat aber zwei auffallende Mängel:

1. Sie zählt nicht genau im Rhythmus einer Zehntelsekunde, würde also zur Zeitmessung nicht zugelassen werden.

2. Sie hält nicht bei 9,9 s an.

Um den ersten Mangel zu beseitigen, müssen wir die 16 MHz exakt durch 1 600 000 teilen. Hierfür ist die Zählerstellung 1 599 999 zu decodieren und damit der Zähler zurückzusetzen.

1 599 999 = 1 1000 0110 1001 1111 1111B.

Wir brauchen also ein UND-Gatter mit 21 Eingängen. Dessen Ausgang wird mit den Rücksetzeingängen der beiden Frequenzzähler verbunden. Das Ausgangssignal liefert zugleich die Zählerlaubnis für die Dezimalzähler.

Um den zweiten Mangel zu beseitigen, ist die Endstellung der Dezimalzählung (99 = 1001 1001B) zu decodieren und damit das Weiterzählen über den Zählerlaubniseingang verhindern. Die Decodierung ist aber schon in die Zähler eingebaut (Ausgänge TC). Lassen Sie sich was einfallen...

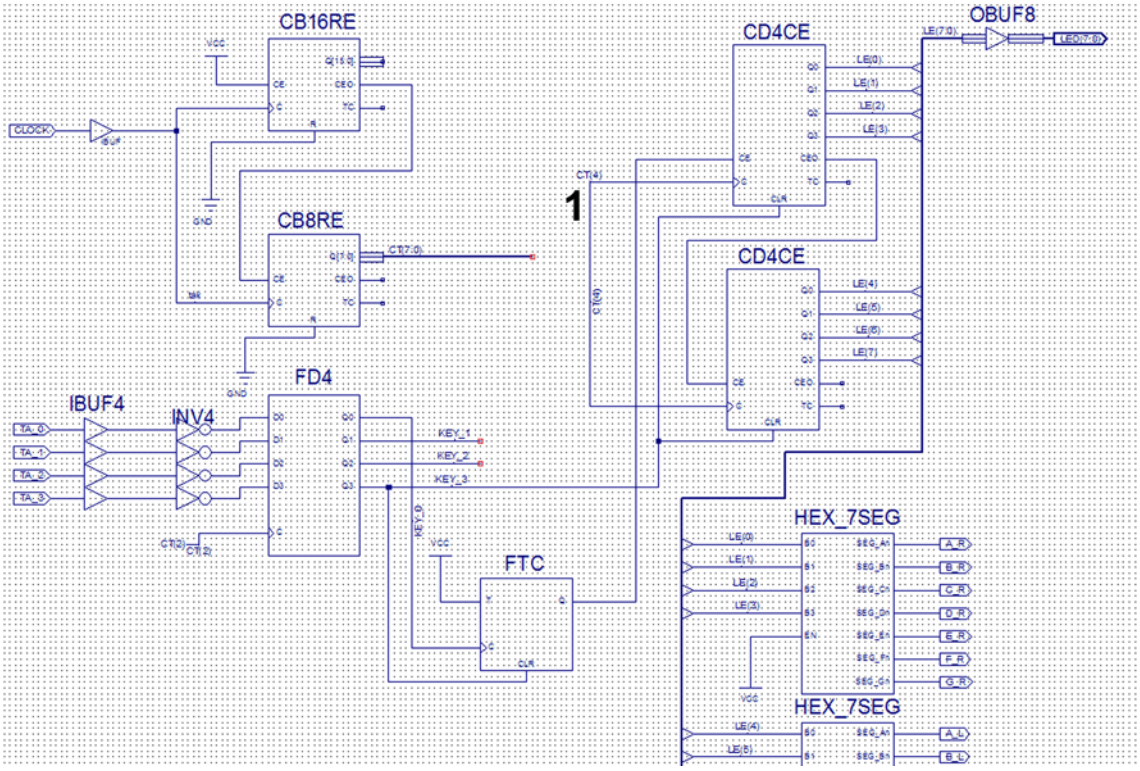


Abb. 61 Die Grundschaltung der Stoppuhr. 1 - der Zähltakt. Ausprobieren...

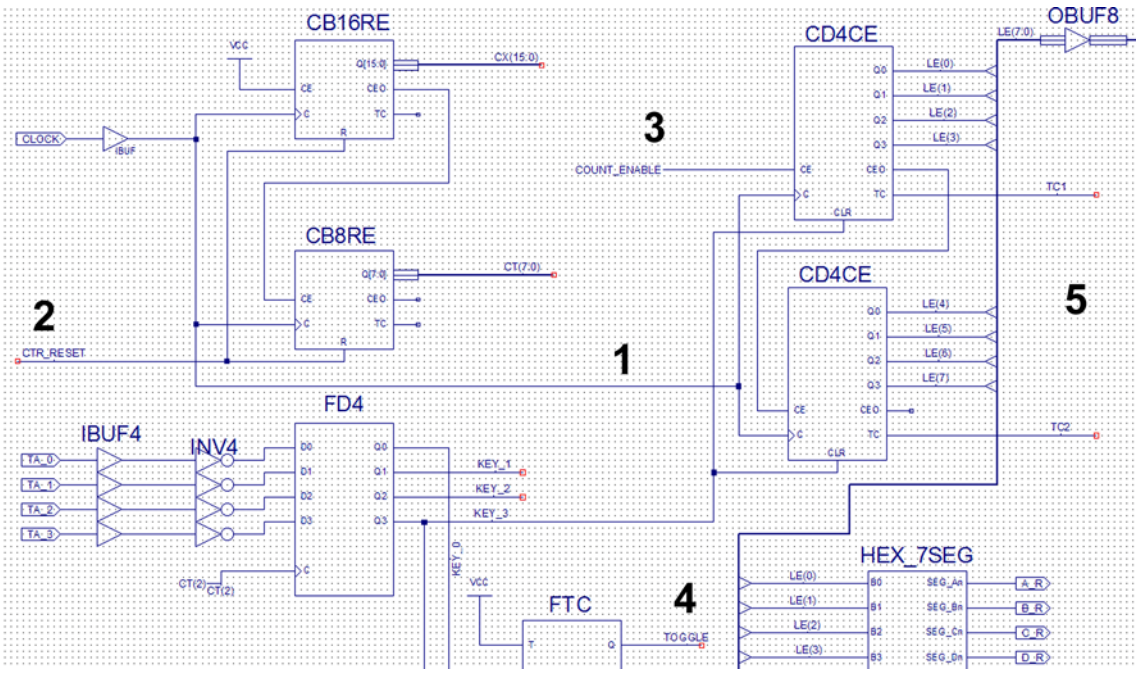


Abb. 62 Die Zählschaltungen sind umzubauen. 1 - die Zähler der Stoppuhr werden an den 16-MHz-Takt angeschlossen. 2 - die Zähler der Frequenzteilung werden an den neuen Decoder angeschlossen. 3 - die Zählerlaubnis kommt von einer neuen Schaltung. 4, 5 - diese Signale werden benötigt, um das Zählerlaubnissignal 3 zu bilden.

Es ist zweckmäßig, die zusätzlichen Gatterschaltungen auf einem neuen Blatt zu zeichnen (wie man dazu kommt, steht weiter hinten).

Den Bus des 16-Bit-Zählers nennen wir CX(15:0). Es sind sieben Nullen und 14 Einsen zu decodieren. Damit wir nicht so viel zu zeichnen haben, bauen wir das große UND-Gatter aus Gattern mit negierten und mit nichtnegierten Eingängen auf. Wir brauchen sieben negierende und 14 nichtnegierende Eingänge.

Die zu decodierende Binärzahl:

CT(...)					CX(...)															
4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	1	1	0	1	0	0	1	1	1	1	1	1	1	1	1

Zählerlaubnis: Die Stoppuhr soll zählen,

- wenn der Frequenzteiler durch Null geht (also der Decoder aktiv wird) UND
- wenn das Toggle-Flipflop aktiv ist UND
- wenn die Endstellung (99) NICHT erreicht ist, wenn also TC1 und TC2 nicht beide zugleich Eins sind (NAND-Verknüpfung).

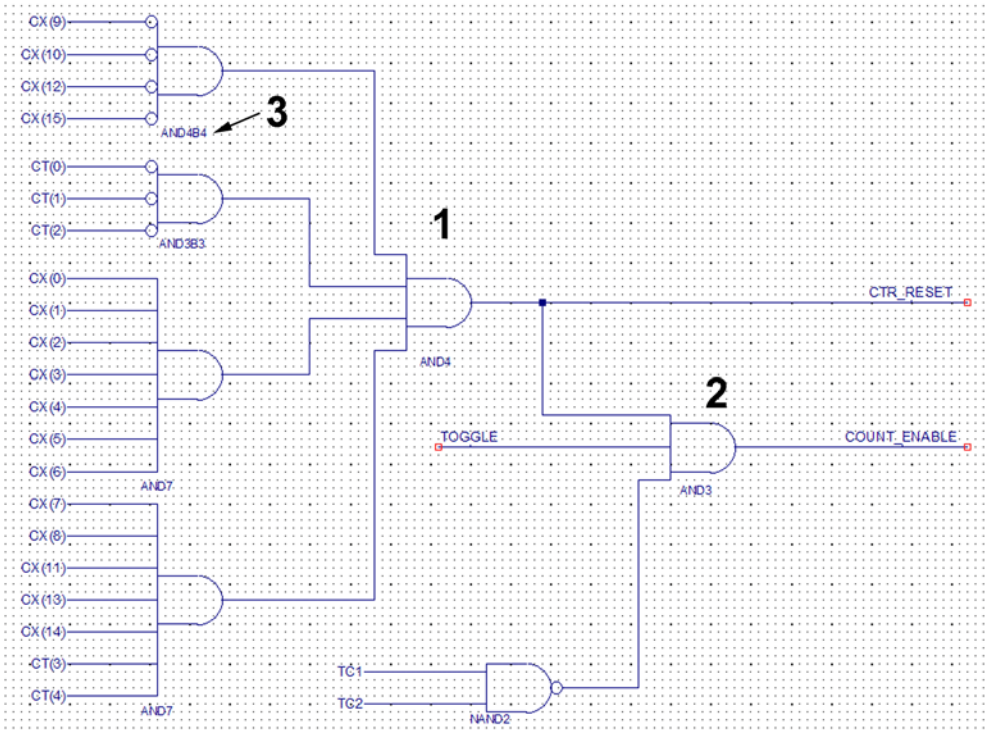


Abb. 63 Die zusätzlichen Schaltungen. 1 - Decoder. 2 - Zählerlaubnis. 3 - so kommen wir zu den UND-Gattern mit negierten Eingängen. Diese Funktionselemente haben ein B in der Bezeichnung. AND3B1 heißt: drei Eingänge,davon einer negiert. Ein Gatter AND4B4 hat demzufolge vier Eingänge, die alle negiert sind.

Aufgabe 6: Prüfgenerator für serielle Schnittstellen
Erprobung: (1) mit Oszilloskop, (2) mit Windows-PC und Hyperterminal.

Der Prüfgenerator soll ein einzelnes Prüfzeichen senden. Das gesamte Prüfmuster ist 16 Bits lang:

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.	15.	16.
Start	Prüfzeichen (ASCII)								Stopbit + Ruhezustand						
0	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	1	1	1	1	1	1	1

Das Prüfzeichen (ASCII):

Einstellzähler 2 (KEY1)				Einstellzähler 1 (KEY0)			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Der ASCII-Code:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20:		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50:	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70:	p	q	r	s	t	u	v	w	x	y	z	{	¦	}	~	

Beispiel: Prüfzeichen "1" = 31H, Prüfzeichen "A" = 41H.

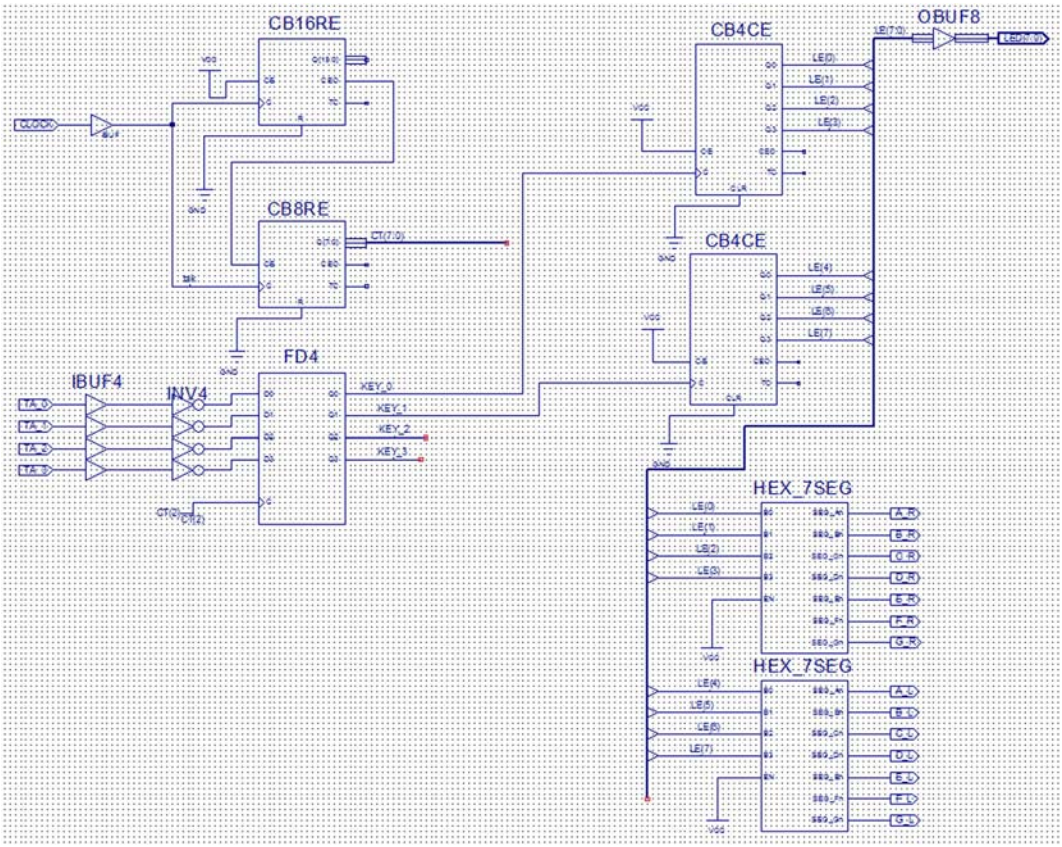


Abb. 64 Die Hilfsschaltungen. Taktteiler, Entprellung, Einstellzähler, Anzeige.

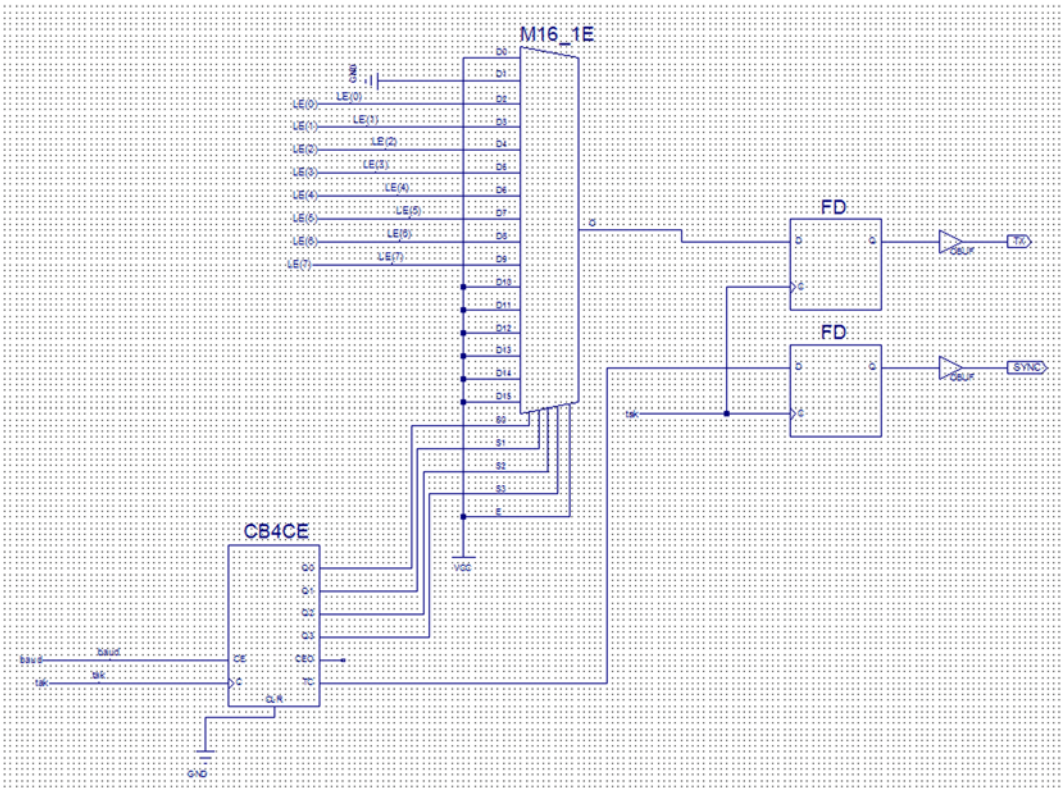


Abb. 65 Die Grundschaltung: Ein Multiplexer wird mit dem zu sendenden Bitmuster belegt und zyklisch abgefragt. Darstellung auf einem neuen Blatt des Schaltplans (wie man dazu kommt, wird weiter unten erklärt).

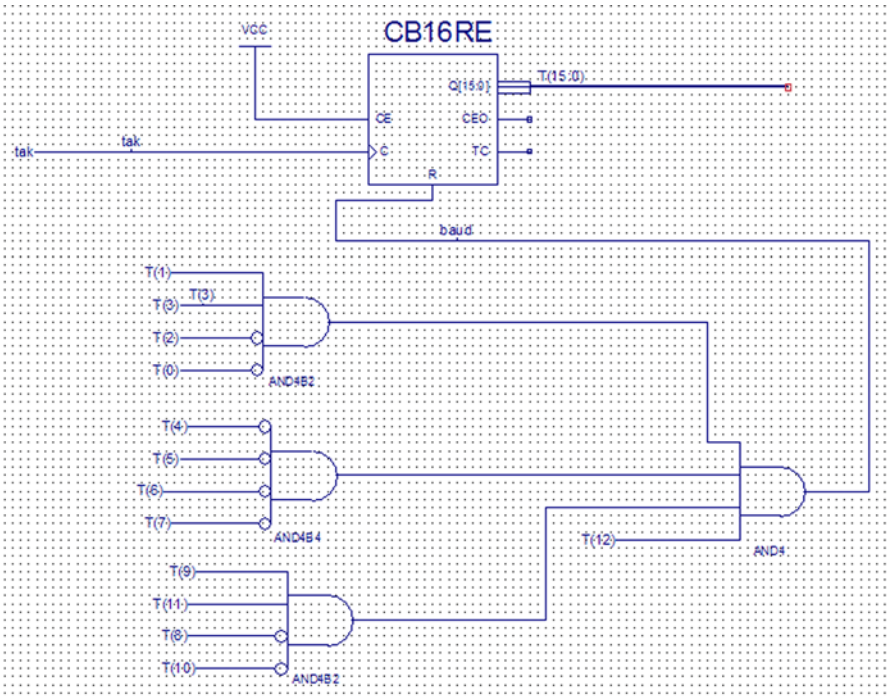


Abb. 66 Der Frequenzteiler für den Bittakt (Baudrate). Wir wollen mit 2400 Bits/s senden. Dazu sind die 16 MHz durch 667 zu teilen.

Neue Ausgänge (zum Eintragen in die Constraints-Datei):

TX: Pin 94 (TXD der seriellen Schnittstelle).

SYNC: Pin 85 (Pin 2 des 40poligen Steckverbinders).

Schritt 1: Grundschaltung wie beschrieben implementieren (zyklischer Umlauf) und zunächst mit dem mit Oszilloskop, dann mit dem Hyperterminal erproben. Ggf. probeweise den heruntergeteilten Abfragetakt auf SYNC legen und nachsehen, ob es stimmt.

Schritt 2: Die Anordnung soll nur senden, solange die Taste 2 niedergehalten wird (ansonsten schreibt es den Bildschirm zu schnell voll). Es ist eine vollsynchrone Lösung zu finden, so daß nur ganze Prüfmuster gesendet werden und im Ruhezustand TX auf Eins gehalten wird (ansonsten synchronisiert das Hyperterminal nicht richtig und zeigt Mist an). Lassen Sie sich also was einfallen ... *Praxistip:* Den Datengenerator immer umlaufen lassen und nur das Senden freigeben oder sperren. Taste 2 nach jedem Umlauf abtasten.

Schritt 3: Erweiterung auf das Senden einzelner Zeichen. Betätigung von Taste 3 soll bewirken, daß nur ein einziges Zeichen gesendet wird (Single-Shot-Betrieb).

Wie kommen wir zu weiteren Blättern im Schaltplan?

1. Mauscursor irgendwo auf die freie Zeichenfläche.

2. Rechte Maustaste. Dann *Object Properties.*

3. Dann *New – OK – OK.*

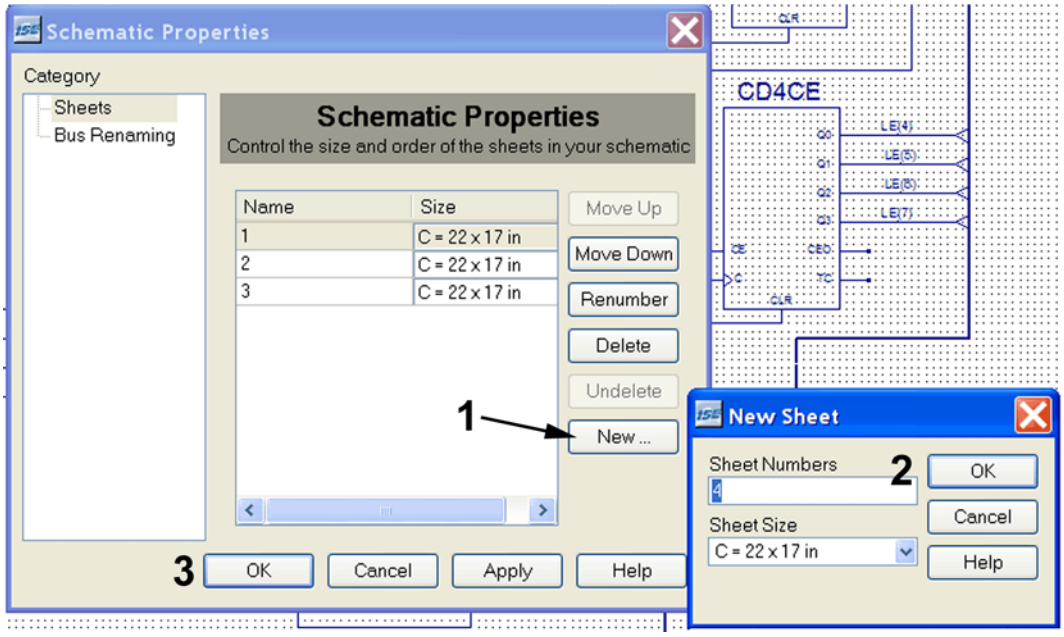
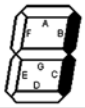
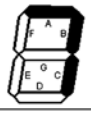
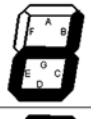
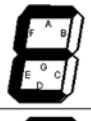
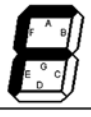


Abb. 67 Anfordern eines neuen Blattes. 1 – 2 – 3: in dieser Reihenfolge klicken. Ggf. zuvor Papierformat auswählen (*Sheet Size*).

Praxistip: Wenn der Schaltplan gedruckt werden soll (Projekt, Abschlußarbeit usw.), auf Format A4 gehen und die Schaltung auf entsprechend viele Blätter aufteilen. Macht zwar etwas Arbeit, sieht aber VIEL besser aus...

Anschluß	**Pin**
Taste 0	49
Taste 1	46
Taste 2	43
Taste 3	42
LED 0	41
LED 1	40
LED 2	39
LED 3	37
LED 4	36
LED 5	35
LED 6	97
LED 7	96
Quarztakt	22
Segment A links	72
Segment B links	71
Segment C links	70
Segment D links	68
Segment E links	67
Segment F links	66
Segment G links	65
Taste oder Dezimalpunkt links	64
Segment A rechts	85
Segment B rechts	82
Segment C rechts	81
Segment D rechts	80
Segment E rechts	79
Segment F rechts	78
Segment G rechts	77
Taste oder Dezimalpunkt rechts	76
Serielle Schnittstelle, Sendedaten (TXD)	94
Serielle Schnittstelle, Empfangsdaten (RXD)	92

Die hexadezimale Siebensegmentanzeige:

d	c	b	a	G	F	E	D	C	B	A	
0	0	0	0	0	1	1	1	1	1	1	
0	0	0	1	0	0	0	0	1	1	0	
0	0	1	0	1	0	1	1	0	1	1	
0	0	1	1	1	0	0	1	1	1	1	
0	1	0	0	1	1	0	0	1	1	0	
0	1	0	1	1	1	0	1	1	0	1	
0	1	1	0	1	1	1	1	1	0	1	
0	1	1	1	0	0	0	0	1	1	1	
1	0	0	0	1	1	1	1	1	1	1	
1	0	0	1	1	1	0	1	1	1	1	
1	0	1	0	1	1	1	0	1	1	1	
1	0	1	1	1	1	1	1	1	0	0	
1	1	0	0	0	0	1	1	0	0	1	
1	1	0	1	1	0	1	1	1	1	0	
1	1	1	0	1	1	1	1	0	0	1	
1	1	1	1	1	1	1	0	0	0	1	

Die Innenschaltung des Siebensegmentdecoders: