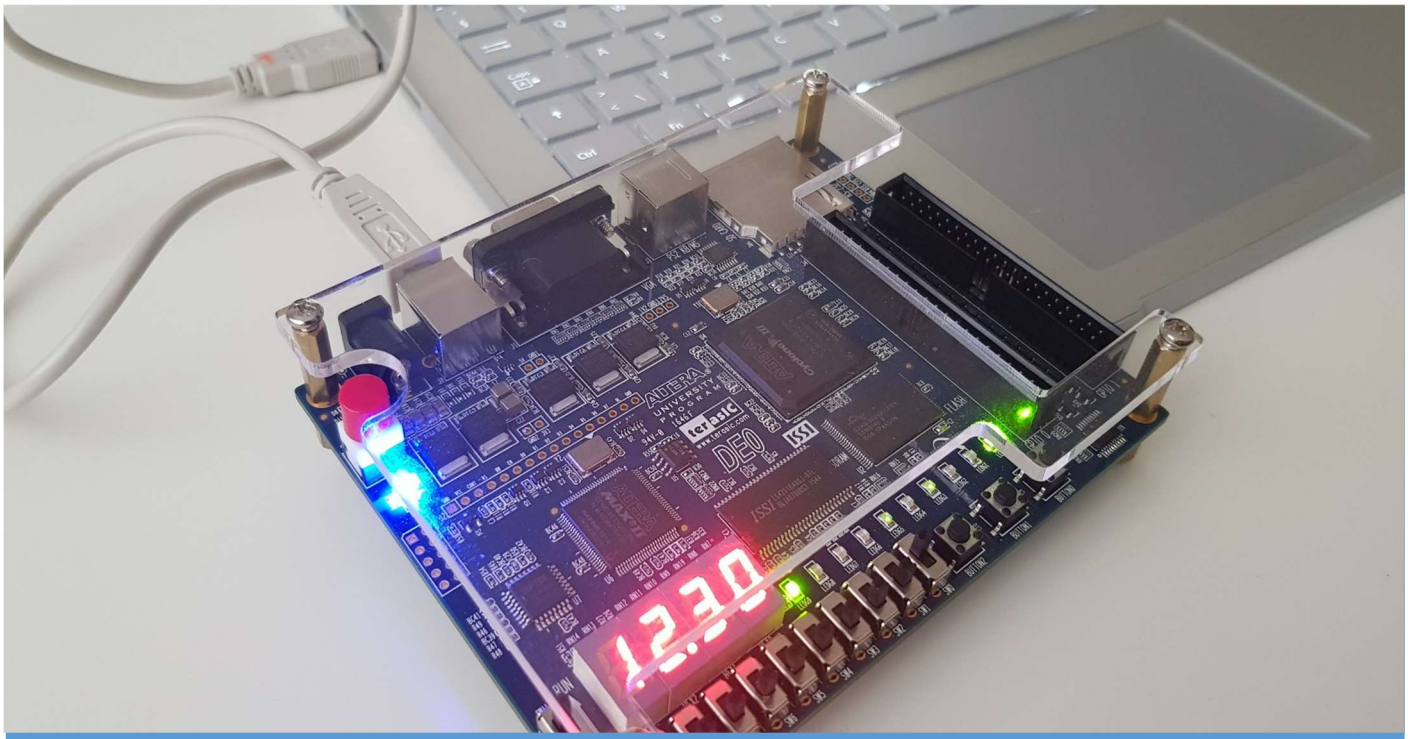
SEMESTERARBEIT FPGA / VHDL PROGRAMMIEREN EINER UHR



JUVENTUS TECHNIKERSCHULE Zürich

Inhaltsverzeichnis

Revisionsverzeichnis

Revision no.	Date	Change note	Changed by
1.0	19.12.2018	Initial release	n/a

Abbildungsverzeichnis

Quellenverzeichnis

- Skript Digitaltechnik FPGA/VHDL von Beat Käppeli
- VGA Quellcode: https://www.youtube.com/watch?v=WK5FT5RD1sU&list=PLtKY2cby05DdwRI4D3Lq3_WfcqvekikSP&index=4&t=0s
- https://www.nandland.com/

Aufgabenstellung

Technikerschule HF Zürich
www.technikerschule-hf.ch

Semesterarbeit der Technikerschule HF Zürich

Studiengang:	Digitaltechnik Labor
Semester:	4ED
Thema	**Uhr mit Ausgabe an VGA Display**
Ziel:	Es wir eine Uhr im FPGA implementiert. Die aktuelle Zeit wird im 7 Segmentdisplay ausgegeben. Mittels Tasten und Switches muss die Uhr gerichtet werden können. Optional: - Der Sekundenimpuls wird als Grafik auf einem VGA Display ausgegeben
Umfang:	▪ Analyse der Aufgabenstellung ▪ Definition der Architektur ▪ Entwurf und Realisierung der ausgewählten Variante ▪ Simulation und Timing Analyse → Dokumentation ▪ Dokumentation und Test (Es ist ein Laborbericht zu erstellen) ▪ Der Umfang der Arbeit ist laufend mit dem Betreuer abzustimmen
Bemerkung	Der Dozent unterstützt den Studenten bei der Entwicklung der Architektur
Diplomand / Diplomandin:	Philipp Frase
Betreuer:	Beat Käppeli
Auftraggeber:	
Ausgabe der Arbeit:	
Abgabe der Arbeit:	**Donnerstag, 24. Januar 2018, 23.59 Uhr** an die Dozentin / den Dozenten
Allgemein:	Als Basis dient der eingereichte Themenvorschlag
Mitbegleitende Dokumente:	▪ -

Unterschriften

Name Vorname Studierende / Studierender	Name Vorname Dozentin / Dozent	Name Vorname Studiengangleiter
_____	_____	_____

Abbildung 1 Aufgabenstellung

Analyse der Aufgabe

Um die Arbeit möglichst effizient zu gestalten, ist es nötig die Aufgabe zu analysieren.

Der Umfang der Arbeit umfasst im Wesentlichen fünf Punkte:

- Definition der Architektur
- Entwurf und Realisierung der ausgewählten Variante
- Simulation und Timing Analyse
- Dokumentation verfassen
- Testen der Schaltung und schreiben des Laborberichts

Das Ziel ist es, mittels des Terasic DE0 Evaluation Boards eine digitale Uhr zu programmieren.

Sie soll folgende Funktionen besitzen:

- Anzeige der Stunden und Minuten auf der 7-Segment Anzeige
- Die Ziffern für Stunden und Minuten sollen einzeln eingestellt werden können.
    - Dies soll über die Pushbuttons geschehen.

Optional soll die Uhrzeit ebenfalls über die vorhandene VGA Schnittstelle ausgegeben werden. Entweder im analogen oder im digitalen Design.

Definition der Architektur

Die Architektur der Hardware-Beschreibung besteht aus einer Zustandsmaschine, welche 2 Zustände besitzt. Hier folgt nur eine kurze Beschreibung des Aufbaus der Architektur. Auf die einzelnen Zustände wird im Folgenden Kapitel genauer eingegangen.

Der erste Entwurf

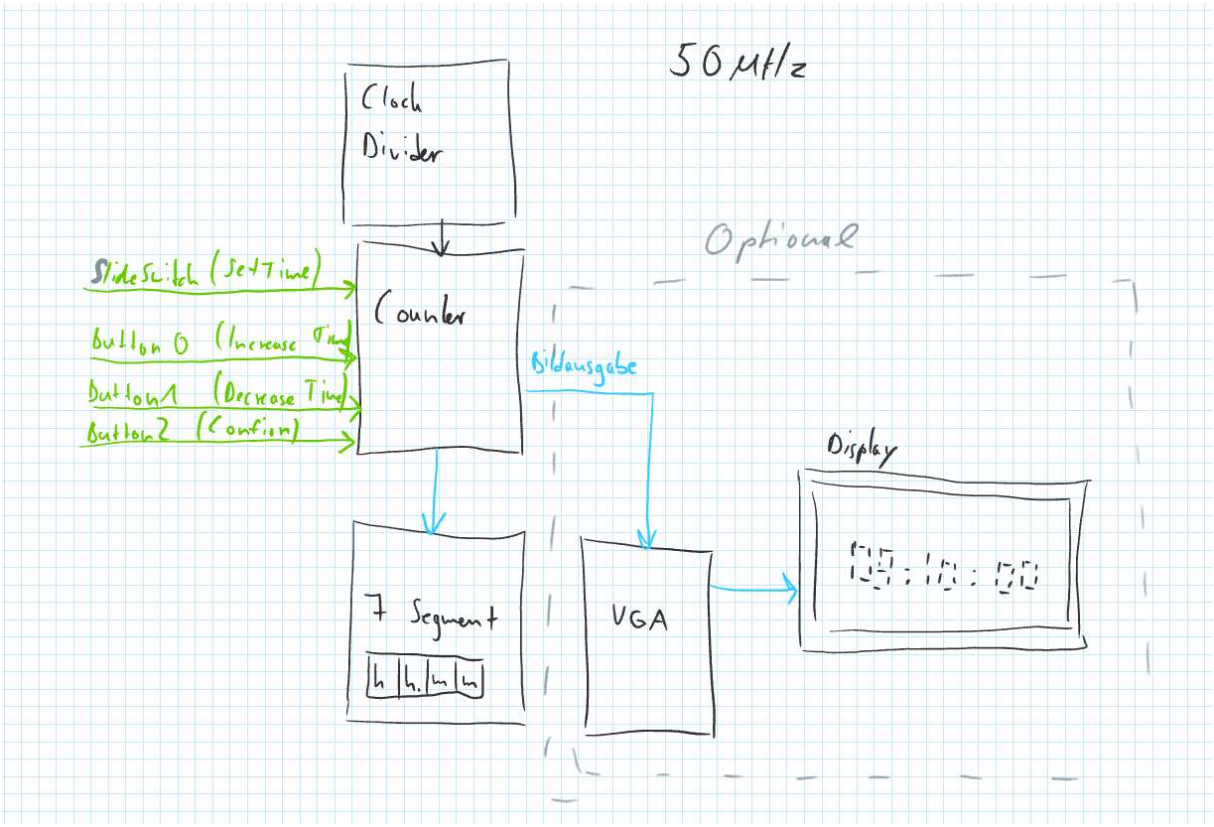


Abbildung 2 Der erste Entwurf

Semesterarbeit_FPGA_VHDL

Dies ist das „Top Level“ VHDL File.

Zustand 1 [show_clock]

Dies ist der initiale Zustand, nach programmieren des Boards. Die Uhr startet mit einem Wert von 00:00:00 und fängt direkt von null an zu zählen.

Zustand 2 [set_time]

Dieser Zustand wird erreicht, indem man den Schiebeschalter (set_switch) nach oben schiebt und das Signal somit auf „high“ gesetzt wird. Nun hält die Uhr an und man kann die Uhr einstellen.

Zuerst wird die erste Ziffer der Stunden eingestellt und mit der „confirm“ Taste bestätigt. Danach kann man die zweite Ziffer der Stunden einstellen und wiederum bestätigen. Dasselbe gilt auch für die Minuten.

VGA_SYNC

In der Komponente VGA_SYNC befindet sich das ganze Timing sowie auch die Bildinformation für den VGA Ausgang. Die Komponente ist ebenfalls mit dem Top LeveL Design File verbunden.

PLL

Ein PLL ist ein sogenannter Phase Locked Loop. Er erlaubt es eine Eingangsfrequenz in eine höhere oder tiefere Ausgangsfrequenz umzuwandeln. Da die benötigte Pixel_Clk Frequenz je nach Bildschirm unterschiedlich ist, ist der PLL hier sehr hilfreich. Er wurde als Komponente in das Top Level Design File eingebunden.

Entwurf und Realisierung

Signallaufpfad

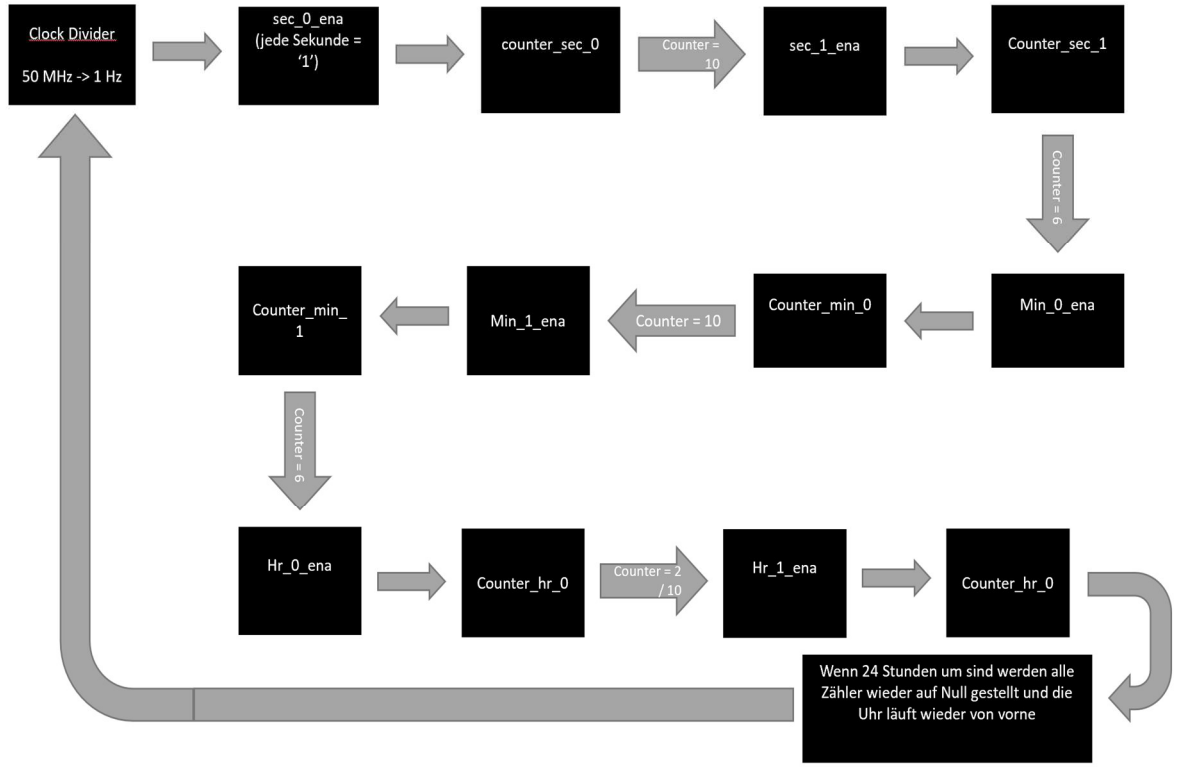


Abbildung 3 Signallaufpfad der Uhr

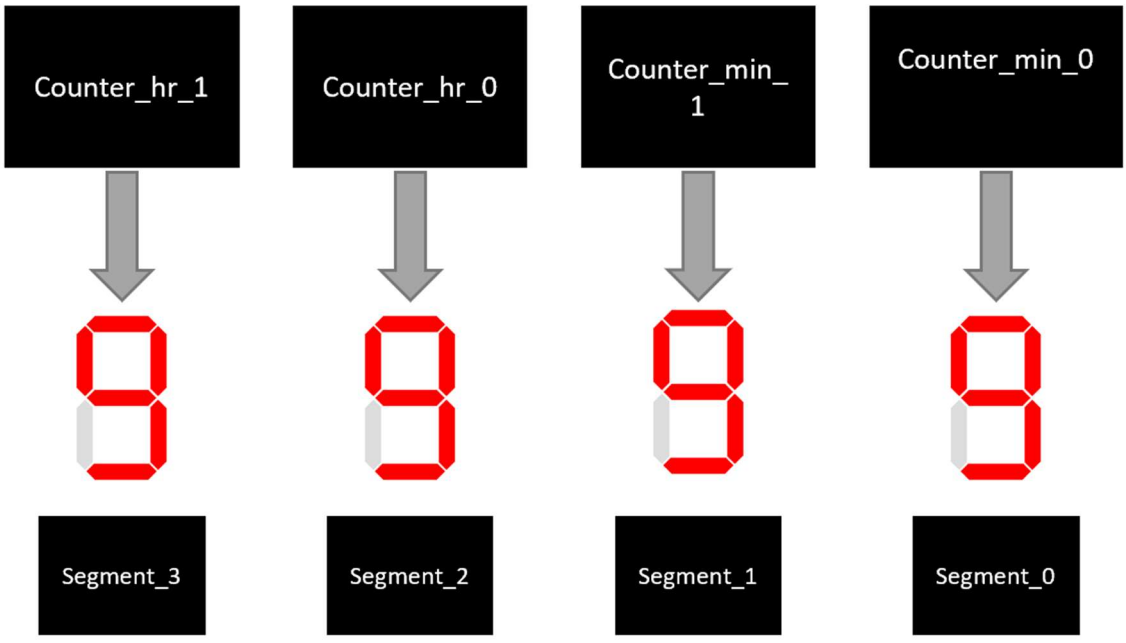


Abbildung 4 Ansteuerung der 7-Segment Anzeige

Show_clock

Die Uhr besteht aus folgenden Signalen:

- sec_0_ena
- sec_1_ena
- min_0_ena
- min_1_ena
- hr_0_ena
- hr_1_ena
- evening_ena

Jedes dieser Signale steuert seinen eigenen Zähler. Diese lauten:

- counter_sec_0 (Ziffer 0, zählt von 0-9)
- counter_sec_1 (Ziffer 1, zählt von 0-5)
- counter_min_0 (Ziffer 2, zählt von 0-9)
- counter_min_1 (Ziffer 3, zählt von 0-5)
- counter_hr_0 (Ziffer 4, zählt von 0-3 oder von 0-9, abhängig ob es tagsüber oder abends ist)
- counter_hr_1 (Ziffer 5, zählt von 0-2)

Um den 50 MHz Clock (50 Millionen Perioden pro Sekunde) auf eine Frequenz von 1 Hz runter zu bringen ist es nötig einen Zähler zu kreieren, welcher auf 50 Millionen zählt und bei Erreichen dieser Zahl ein enable (sec_0_ena) Signal auf „high“ setzt. Sobald dieses Signal auf „high“ ist wird der counter_sec_0 um eins erhöht. Das gleiche gilt für alle anderen enable Signale.

Das evening_ena Signal steuert das Zähllimit von counter_hr_0, da dieser Stundenzähler je nachdem ob es abends oder tagsüber ist unterschiedlich hoch zählen muss. Tagsüber muss der Zähler von 0-9 zählen zum Beispiel wenn es 07:00 Uhr bzw. 17:00 Uhr ist. Wenn es hingegen abends ist (ab 20.00 Uhr) darf der Zähler nur bis 3 zählen, damit die Uhr nicht eine unrealistische Uhrzeit wie zum Beispiel 27:00 Uhr anzeigt. Das heisst das Signal evening_ena wird bei einem counter_hr_1 stand von 2 auf „high“ gesetzt.

Sobald die Uhr 23:59:59 +1 Sekunde anzeigt, werden alle Zähler wieder auf null gesetzt.

Set_time

Die Uhr wird mit Hilfe eines Set Switches und zwei Pushbuttons eingestellt. Sobald der Set Switch nach oben geschoben wird befindet man sich im Zustand set_clock, in welchem die Uhr eingestellt wird.

Zuerst wird die erste Ziffer der Uhr eingestellt, also der counter_hr_1. Drückt man nun den Pushbutton 1(counter_up) und lässt ihn wieder los wird der Zähler um eins erhöht. Dies wird so oft wiederholt, bis die gewünschte Anzeige erreicht ist. Nun wird mittels Pushbutton 0 (confirm) die Ziffer bestätigt und es kann dies nächste eingestellt werden.

Zur Orientierung werden ebenfalls noch vier LEDs (nr. 6-9) beleuchtet, je nachdem bei welcher Ziffer man sich gerade befindet.

Sind alle Ziffern korrekt eingestellt kann man den Set Switch wieder nach unten schieben und die Uhr läuft ab der eingestellten Uhrzeit weiter.

RTL Simulation

Bei der Register Transfer Level Simulation wird nur die Funktionsweise der Schaltung simuliert.
Um diese Simulation durchzuführen ist es nötig einen «Testbench» zu erstellen.
Dieser bindet das komplette Design als eine Komponente ein.

Um nun das gewünschte Verhalten der Schaltung zu beschreiben ist es nötig die Stimuli zu ergänzen.
In ihr wird das logische Verhalten der Schaltung beschrieben.

Zuerst wird ein «Reset» ausgeführt, damit alle Signale einen bestimmten Zustand haben.
Danach werden alle anderen Signale vorgegeben. Zum Beispiel simuliert man, dass man den «set_switch» auf logisch eins setzt. Danach wird in den Signal Verläufen analysiert ob das gewünschte verhalten eintrifft (Zustand muss sich ändern).

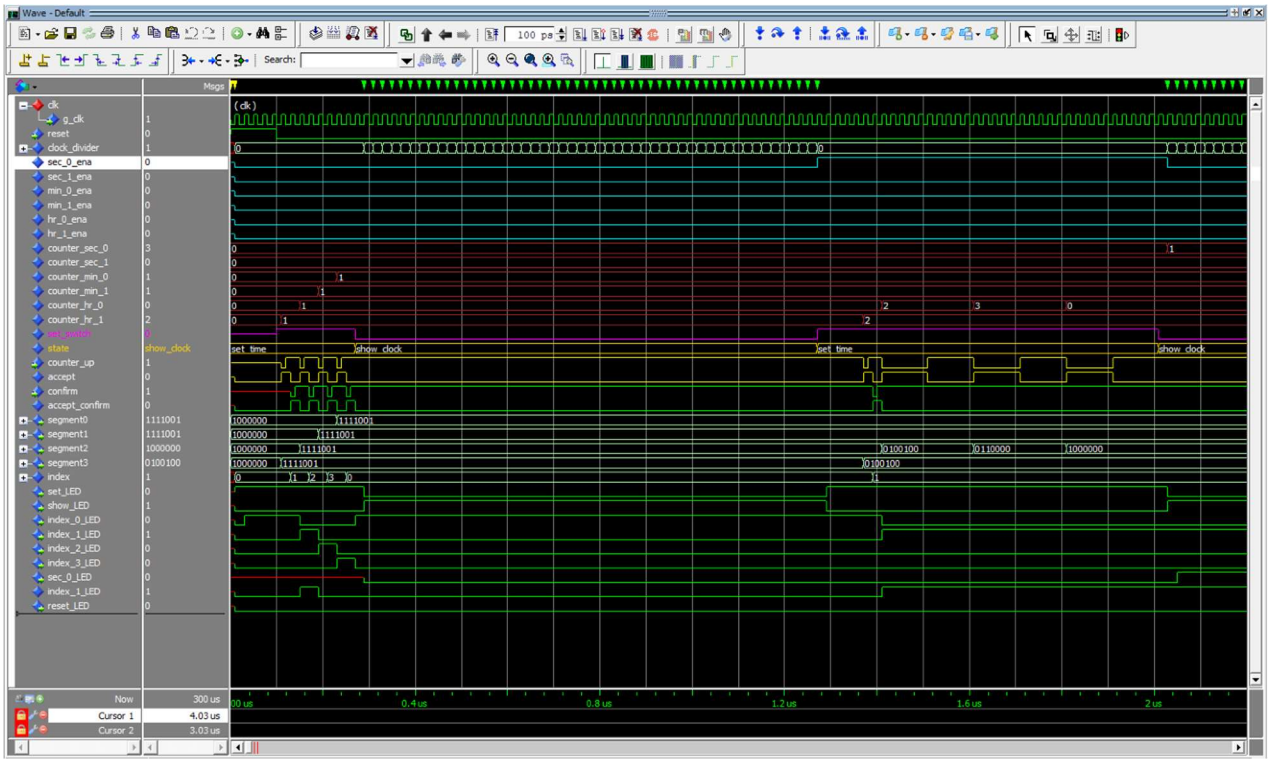


Abbildung 5 Simulation mit ModelSim

```
24          -- code executes for every event on sensitivity list
25      wait for 100 ns;
26      set_switch <= '1';
27      wait for 10 ns;
28      counter_up <= '0';
29      wait for 10 ns;
30      counter_up <= '1';    -- counter_hr_1 = 1
31      wait for 10 ns;
32      confirm <= '0';
33      wait for 10 ns;
34      confirm <= '1';
35
36      wait for 10 ns;
37      counter_up <= '0';
38      wait for 10 ns;
39      counter_up <=  '1';   -- counter_hr_0 = 1
40      wait for 10 ns;
41      confirm <= '0';
42      wait for 10 ns;
43      confirm <= '1';
44      wait for 10 ns;
```

Abbildung 6 Stimuli des Testbenches

Timing Analyse

Die Timing Analyse ermittelt die Delays (Verzögerungen) der logischen Schaltung.
Sie sollte durchgeführt werden, um die maximale Frequenz des Designs zu bestimmen und mögliche Fehler bzw. Timing Probleme aufzudecken.

Das folgende Bild zeigt die maximale mögliche Frequenz bei einer Versorgungsspannung von 1.2V und bei 85°C.

Slow 1200mV 85C Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	115.97 MHz	115.97 MHz	g_clk	
2	154.99 MHz	154.99 MHz	MyComponent_2|altpll_com...to_generated|pll1|clk[0]	

Abbildung 7 F_max bei 85°C

Die maximale Frequenz bei 1.2V und 0°C liegt bei 124.33 MHz für die g_clk gesteuerten Signale und 166.89 MHz für die PLL getriebenen Signale, also alle VGA Signale.

Slow 1200mV 0C Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	124.33 MHz	124.33 MHz	g_clk	
2	166.89 MHz	166.89 MHz	MyComponent_2|altpll_com...to_generated|pll1|clk[0]	

Abbildung 8 F_max bei 0°C

Die Maximalfrequenz ist bei 0°C höher, da die internen MOSFET Transistoren eine höhere Verstärkung besitzen und dadurch schneller schalten. Dies führt zu schnelleren Frequenzen.

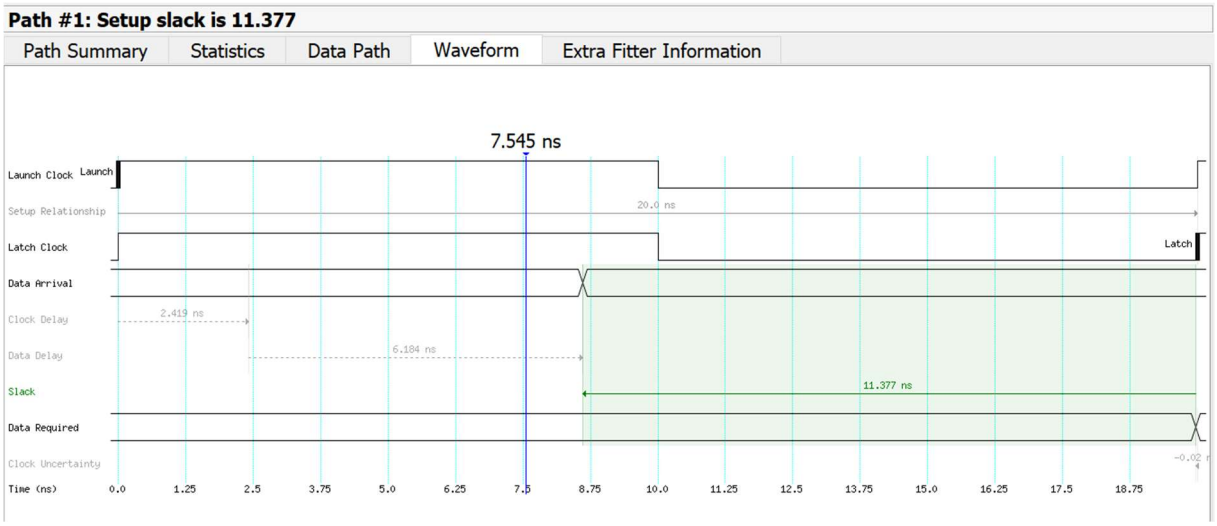


Abbildung 9 Slack Time Diagramm

Im oberen Bild im grünen Bereich sieht man die Zeit, welche für die Signale noch zur Verfügung stünde bis zur nächsten Clock Flanke. Dies ist die sogenannte Slack-Zeit. In diesem Fall wären noch 11.377 ns Zeit.

State Machine Diagramm

Das State Machine Diagramm zeigt den Aufbau der beschriebenen Maschine und wird direkt aus der Quartus IDE erzeugt.

Abbildung 10 Statemachine Diagramm

Leider zeigt das erstellte Diagramm nicht den Input des set_switches an. Dieser steuert den Zustand und bestimmt welcher gerade aktiv ist.

RTL Netz

Das RTL Netz zeigt den Schematischen Aufbau der beschriebenen Schaltung.

Leider sind die Schemas etwas ungleichmässig gross, was es schwierig macht das ganze schön auf einem A4 Blatt darzustellen.

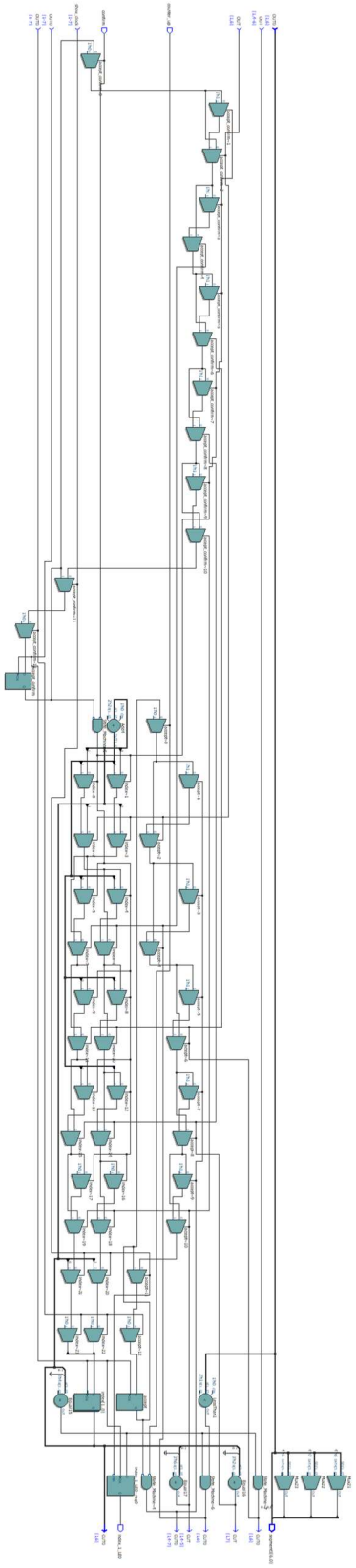


Abbildung 11 RTL Netz Seite 1

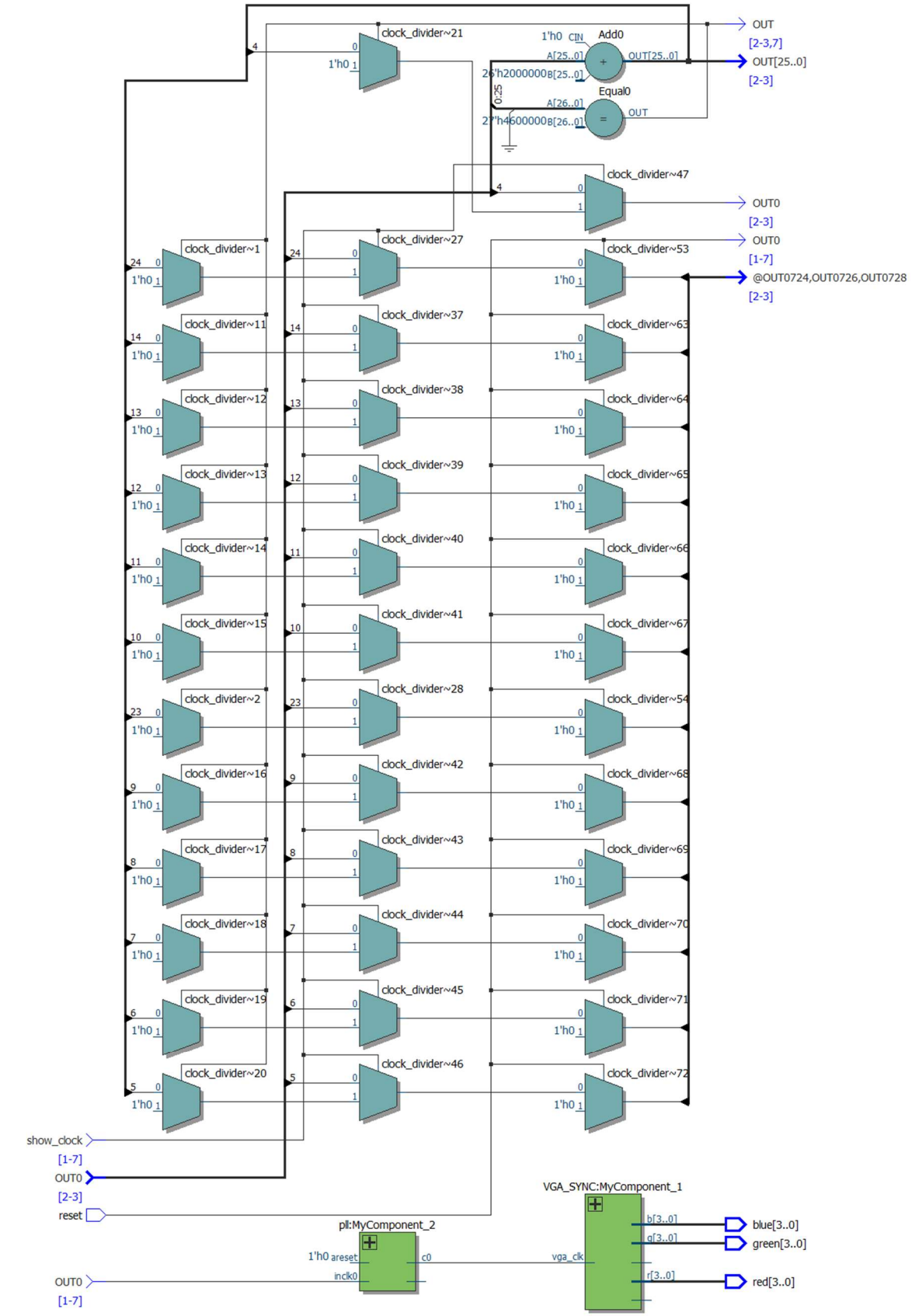


Abbildung 12 RTL Netz Seite 2

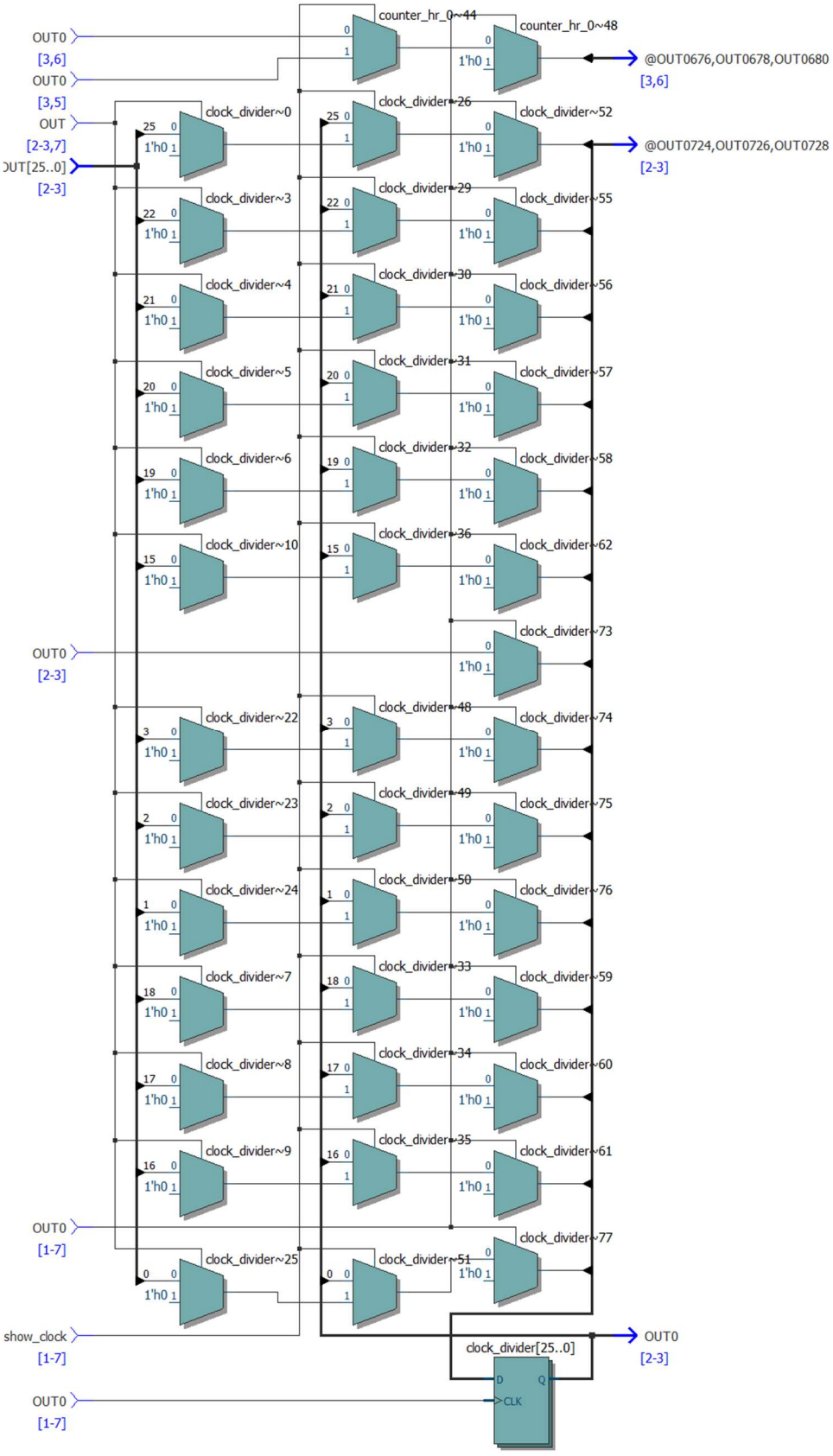


Abbildung 13 RTL Netz Seite 3

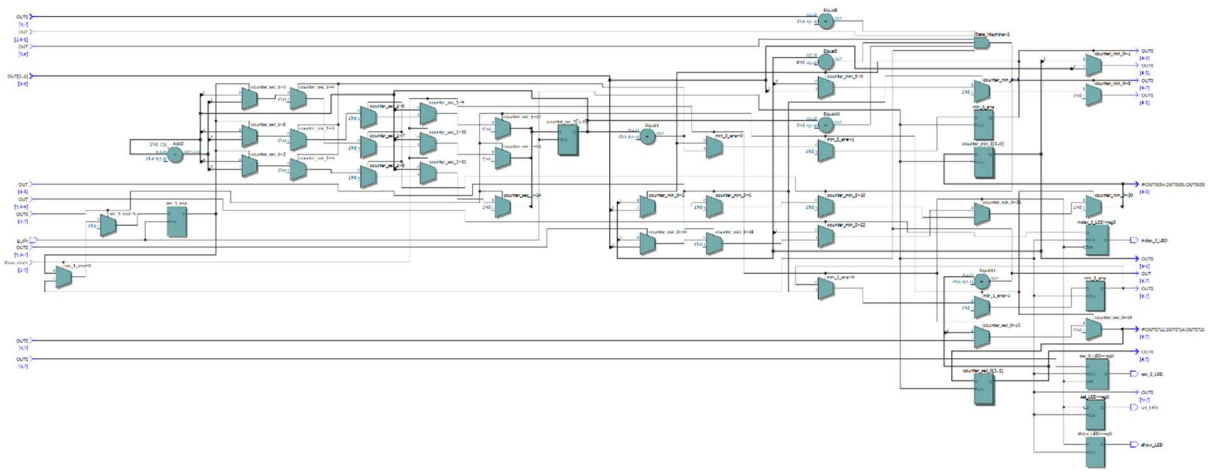


Abbildung 14 RTL Netz Seite 4

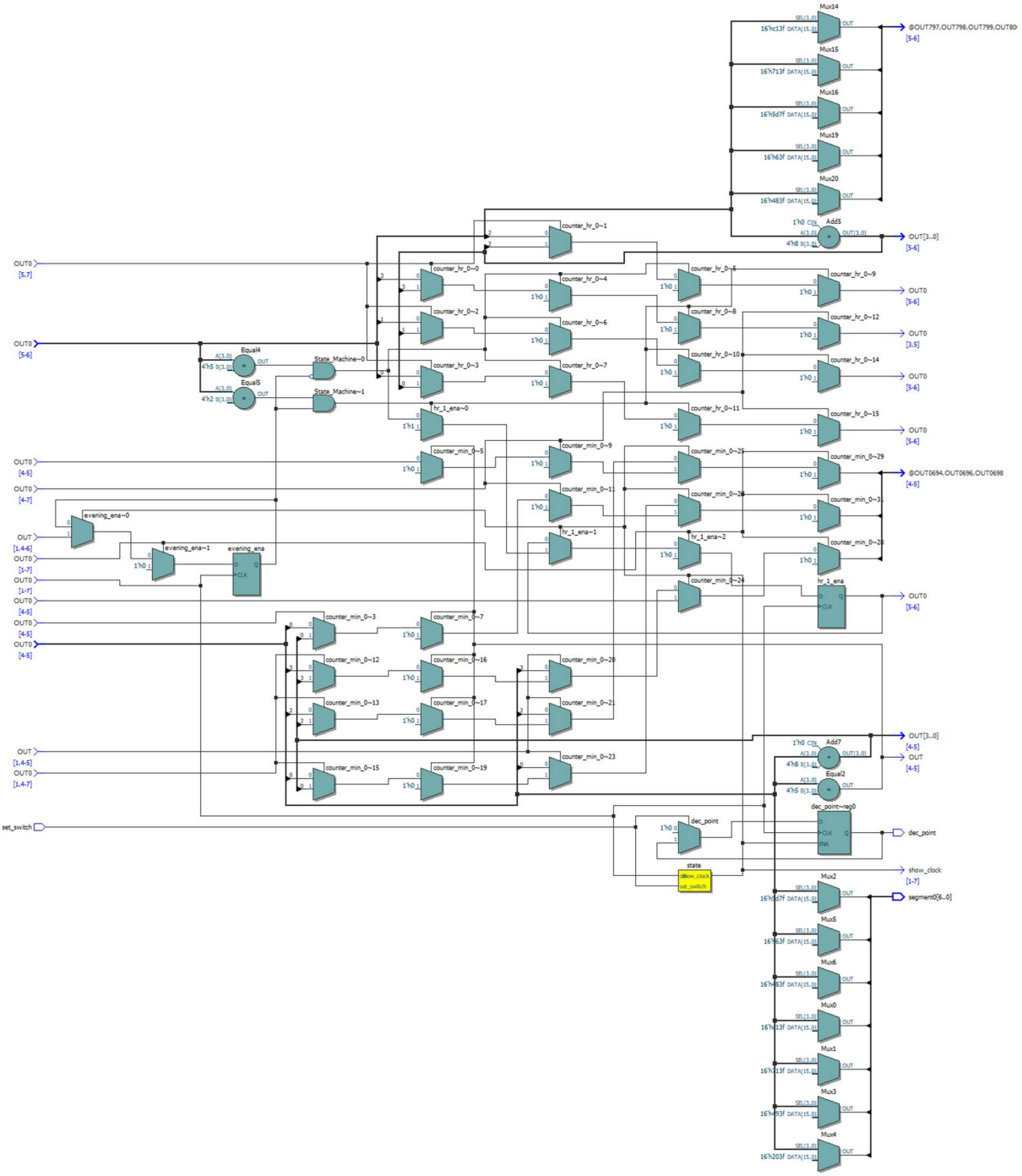


Abbildung 15 RTL Netz Seite 5

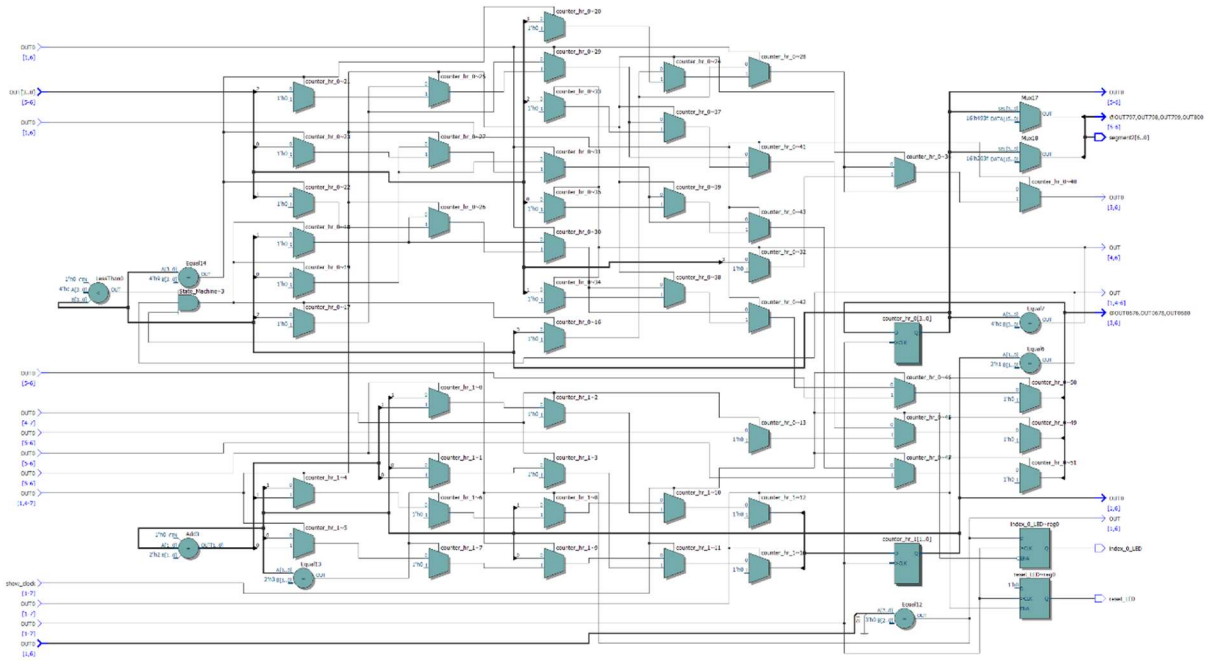


Abbildung 17 RTL Netz Seite 6

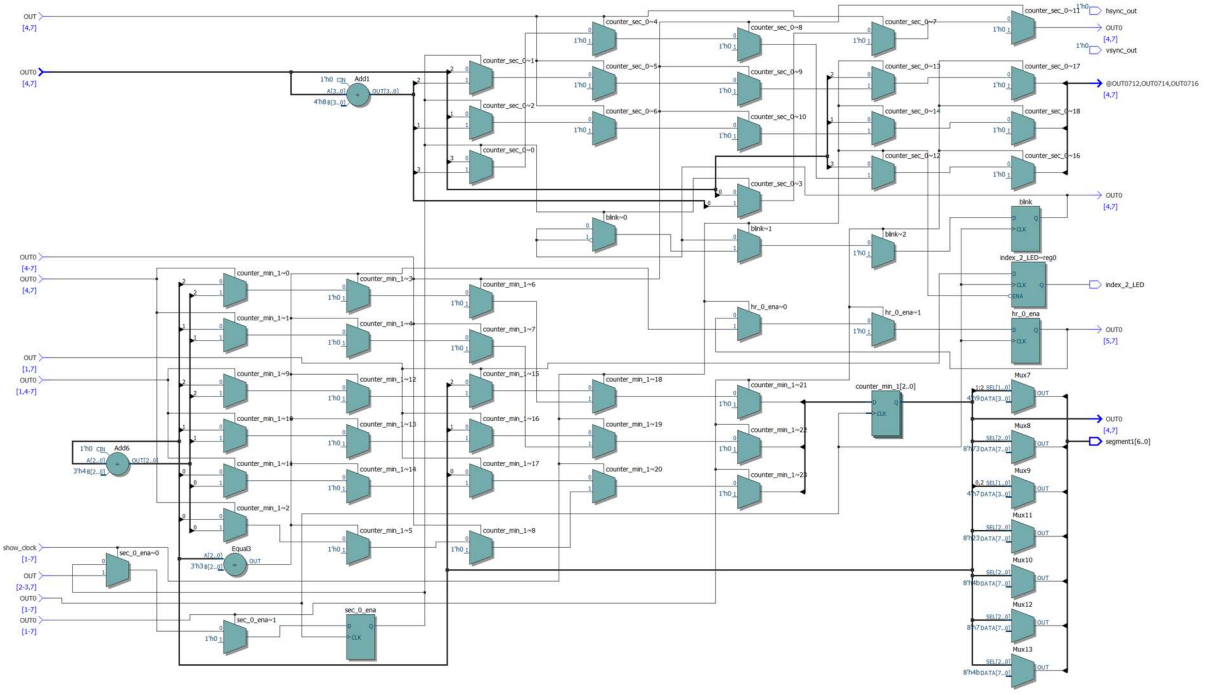


Abbildung 16 RTL Netz Seite 7

Test

Die Uhr wurde in zwei verschiedenen Schritten getestet.

Genauigkeit der Uhr

Da die Uhr durch die Normal -/Winterzeit Umstellung jedes halbe Jahr neu gestellt werden muss, habe ich mir keine grösseren Gedanken darüber gemacht, wie genau die Uhr läuft.

Jedoch liess ich die Uhr über zwei Tage zu einer bestimmten Uhrzeit laufen und hielt sie danach wieder an. Es gab keine merkbare Differenz zu meiner Referenz Zeit.

Einstellen der Uhr

Man kann jede beliebige Uhrzeit im 24 Stunden Format darstellen und es kann kein illegaler Zustand erreicht werden (z.B. 26:00 Uhr).

Bedienungsanleitung

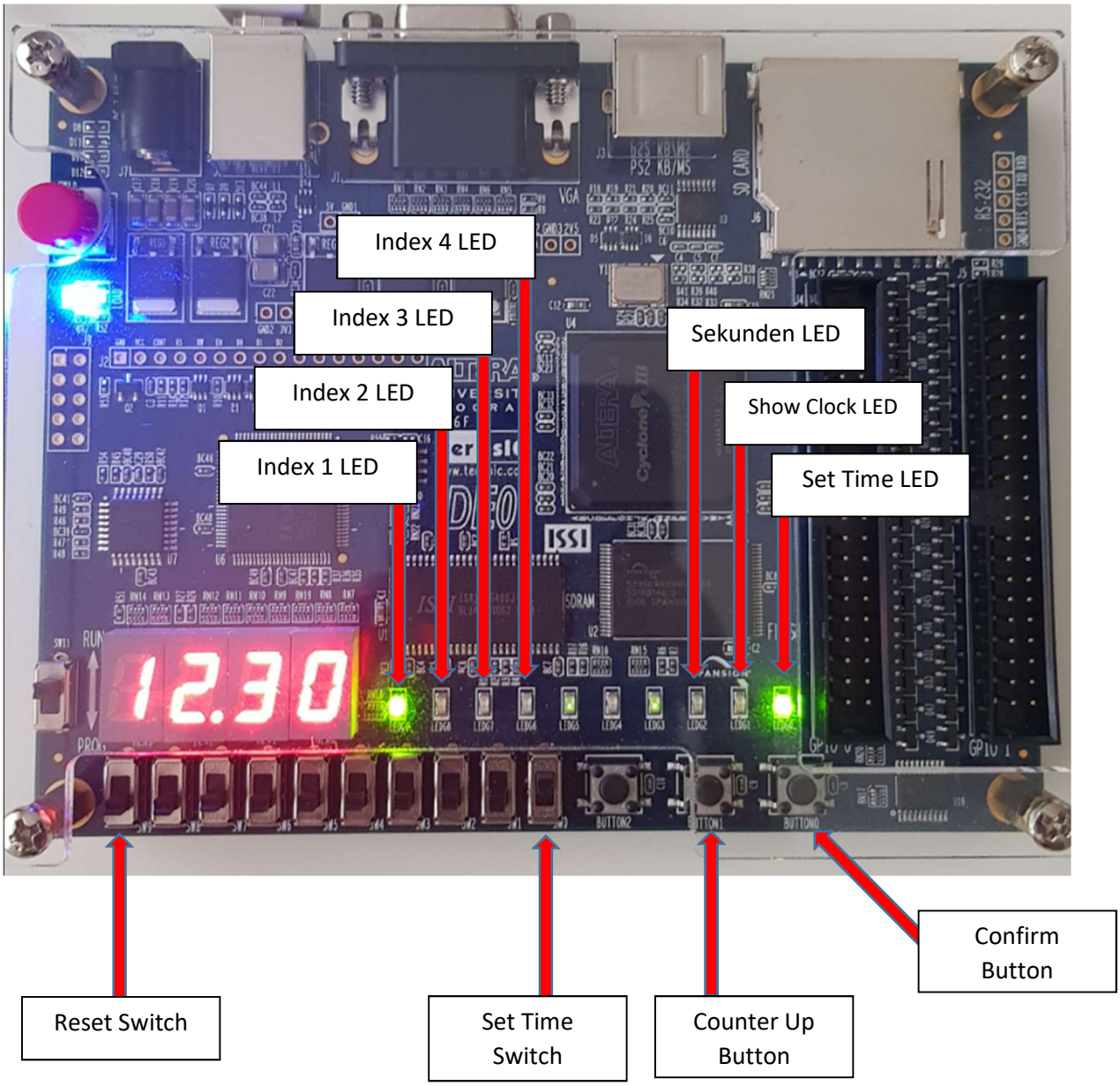


Abbildung 18 Übersicht der Bedienung

Reset Switch

Durch einen Schaltzyklus (hoch -und runter schieben) wird die Uhr wieder auf null gesetzt.

Set Time Switch

Wird der Schalter nach oben geschoben, kann man die Uhr einstellen. Ist die gewünschte Uhrzeit eingestellt, so muss der Schalter wieder nach unten geschoben werden und die Uhr läuft weiter.

Counter Up Button

Befindet sich die Uhr im Einstellzustand, so wird mit diesem Knopf die jeweilige Ziffer um eins erhöht.

Confirm Button

Ist die gwünschte Ziffer auf der 7-Segment Anzeige eingestellt, so wird nach betätigen des Confirm Buttons die nächste Ziffer eingestellt.

Index LEDs

Je nachdem, welche Ziffer gerade eingestellt wird, leuchtet die entsprechende LED. Wenn zum Beispiel die erste Ziffer eingestellt wird, so leuchtet die Index 1 LED.

Sekunden LED

Diese LED blinkt im Sekundentakt.

Show Clock LED

Diese LED leuchtet, wenn man sich im Zustand Show Clock befindet, also im Normalbetrieb.

Set Time LED

Diese LED leuchtet, wenn man sich im Zustand Set Time befindet.

Technisches Fazit

Begonnen habe ich mit einem einfachen Sekundenzähler. Darauf habe ich anschliessend alles aufgebaut. Da ich nicht zwei 7-Segment Anzeigen mit einem Zähler ansteuern konnte (zumindest nicht ohne grössere Umstände) entschied ich mich dafür, 6 verschieden Zähler zu kreieren.

Ein gravierender und sehr zeitintensiver Fehler schlich sich in meinen Code ein.
Anstatt if (xy = 1) schrieb ich ausversehen if (xy <= 1). Das hiess in meinem Fall, dass die Bedingung immer erfüllt war, was natürlich nicht der Sinn der Verzweigung war. Da der Quartus Compiler dies logischerweise nicht als Fehler sah, verbrachte ich eine Weile mit der Fehlersuche und verhalf meinem Dozenten zu einem weiteren Eintrag in seinem Skript.

Als die Uhr lief, kam die nächste Hürde; das Einstellen der Uhr. Dies realisierte ich, indem ich mein bestehendes Design in eine Zustandsmaschine implementierte.

Nun kam das Problem, dass beim Drücken von counter_up, die jeweilige Ziffer irgendeinen Wert annahm. Dies geschah, weil während des Tastendrucks alle 20 Nanosekunden (50 MHz) der jeweilige Wert um eins erhöht wurde.
Zur Lösung dieses Problems verhalf mir mein Dozent. Er empfahl mir ein Hilfssignal „einzubauen“, welches verhindert, dass genau das passiert.

Eine Uhr zu programmieren hört sich anfangs relativ einfach an. Hat man aber einmal damit begonnen, stellt man schnell fest, dass sich doch einige Stolpersteine auf dem Weg zum Ziel befinden.

Persönliche Reflexion

Der Kurs FPGA / VHDL brachte mir viele neue Erkenntnisse. Zu Anfangs wusste ich noch nicht einmal, um was es sich dabei handelt. Heute kann ich sagen, dass es mir möglich ist einfachere Schaltungen mittels FPGA Evaluation Board und Quartus IDE zu programmieren beziehungsweise zu beschreiben.

Die erste Hürde im Fach FPGA / VHDL war es, zu verstehen das es sich bei VHDL nicht um eine Programmiersprache handelt, in der sequentiell Befehle abgehandelt werden, sondern um eine Hardwarebeschreibungs-Sprache. Hat man dies einmal begriffen, wird alles nachfolgende logischerweise viel einfacher.

Ebenfalls finde ich es genial und sehr hilfreich, dass man sein Design mit Hilfe von Modelsim simulieren kann. Dies hilft enorm bei der Fehlersuche.

Leider hat die Zeit nicht mehr ganz gereicht, um das korrekte VGA Signal zu erzeugen. Da wir eine Woche vor Abgabe dieser Arbeit noch die Diplomprüfung im Fach FPGA hatten lag der Fokus ebenfalls sehr stark auf der Prüfungsvorbereitung. Sicher ist aber, dass ich in der Freizeit versuchen werden dies zu vervollständigen.

Alles in allem war dieser Kurs bisher mein Lieblingskurs, da ich sehr viel Neues dazulernen durfte. Ich bin froh, dass wir während des Semesters eine praktische Arbeit ausführen durften.