FH-OÖ/Hagenberg/HSD
Chip-Design 3 — CHD3
Markus Pfaff © 2025
Übung 7
Lauflicht, Zähler

1. Aufgabe *FSM: Lauflicht*

Das Lauflicht (Name der Unit: `RunningLight`) aus der Vorlesung soll nun auf dem Board realisiert werden. Drei LEDs dienen zur Ausgabe des Status. Als Taktquelle ebenso wie für den asynchronen Reset verwenden Sie Taster. Achten Sie dabei auf die Eingangspegel, da diese eventuell invertiert werden müssen.
Die Entity hat folgendes Aussehen:

```
1 entity RunningLight is
2   port (
3     iClk          : in  std_ulogic;
4     inResetAsync  : in  std_ulogic;
5     oState        : out std_ulogic_vector(2 downto 0));
6 end RunningLight;
```

Beschreiben Sie das Lauflicht als synthesefähigen VHDL-Code in den drei Varianten (also in drei *architectures*), welche in der Vorlesung vorgestellt wurden:

- Zwei *processes*, Zustandsüberführungsfunktion von Hand (per Karnaugh-Diagramm) ermittelt (*architecture* `TwoProcessesHandmade`),
- zwei *processes*, Zustandsüberführungsfunktion mittels *case statement* (*architecture* `TwoProcessesWithCase`),
- ein *process*, Zustandsüberführungsfunktion mittels *case statement* (*architecture* `OneProcessWithCase`).

Weisen Sie die richtige Funktion Ihrer Beschreibungen mittels einer automatisierten *Testbench* nach, in welcher alle Beschreibungen zugleich geprüft werden.
Realisieren Sie alle drei Beschreibungen auf dem Board indem Sie für alle drei Beschreibungen jeweils ein eigenes *Testbed* vorsehen. Stellen Sie für alle drei Varianten die relevanten Informationen aus dem *Compilation Report* zusammen. Gibt es *Warnings*, welche problematisch sind?

2. Aufgabe *Lauflicht: Speed of Light*

Bei unseren bisherigen Beispielen schleicht der Takt geradezu von Tastendruck zu Tastendruck. Ihr Lauflicht beispielsweise kann nur so schnell laufen, wie Sie die Taste drücken. Der vordergründig wichtigste Vorteil dedizierter Hardware ist aber die Arbeitsgeschwindigkeit, wie sie beispielsweise ein Echtzeit-Bildverarbeitungsalgorithmus fordert.
Dabei könnte das Lauflicht viel schneller *laufen*. Finden Sie mittels der Ausgaben der Timinganalyse heraus, mit welcher maximalen Frequenz die Zustände des Lauflichts wechseln könnten.

Ist der Betrieb mit 50 MHz, die uns auf dem Board unmittelbar zur Verfügung stehen, möglich?

Wie groß ist die Zeitdauer über welche ein Zustand des Lauflichts nun existieren würde?

Wenn Sie statt des Taktflankentasters den 50 MHz-Takteingang `CLK0` verwenden *würden*, welches Verhalten erwarten Sie dann von den Leuchtdioden?

3. Aufgabe *Ein Zähler*

Nun soll ein einfacher Zähler entworfen werden. Verwenden Sie einen einzigen *process* zur Modellierung. Dieser Zähler soll zunächst einmal nichts anderes tun, als die steigende Taktflanken des 50 MHz-Takts zu zählen. Damit wir das Zählen beobachten können, dimensionieren Sie die Bitbreite so groß, das die höchste Stelle nicht häufiger als einmal in einer Sekunde von '0' nach '1' oder umgekehrt wechselt.
Wie viele Stellen benötigen Sie, um dies sicherzustellen?
Wie viel Zeit vergeht *genau*, bis die höchste Stelle einmal von '0' nach '1' und wieder zurück gewechselt hat?

Beschreiben Sie diesen Zähler nun in VHDL (`Counter(Rtl)`). Hierbei soll das *signal*, welches das Zählregister darstellt, vom *type* `unsigned` sein. Dessen Bitbreite sollen Sie auch später noch einstellen können, weshalb diese mit einem *generic* angegeben wird: `gCountBitWidth`. Wenn der Zähler beim Maximum angekommen ist, soll er wieder bei 0 beginnen, was Sie mithilfe der Modulo-Funktion realisieren sollten.
Weisen Sie die korrekte Funktion Ihres Zählers in der Simulation nach.

Wie lange braucht der Simulator um einen vollen Zählerdurchlauf zu simulieren? Falls Sie so lange nicht warten wollen, schätzen Sie die erforderliche Zeit an Hand einer teilweisen Simulation ab. Um Ihren Zähler dennoch prüfen zu können, setzen Sie einfach die Bitbreite (per *generic*) auf einen „erträglichen" Wert. So läßt sich auch der Überlauf des Zählers leicht simulativ erfassen.

Die 10 höchstwertigen Stellen des Zählers ändern sich am langsamsten und sollen daher zur Visualisierung an die LEDs `LEDR0` bis `LEDR9` angeschlossen werden. Verwenden Sie hierzu ein *Testbed*.

Hat das Syntheseergebnis mehr FlipFlops als Sie erwartet hatten? Wenn ja, finden Sie eine Ursache dafür? Realisieren Sie den Entwurf auf dem Prototyping-Board.
Wie viele FlipFlops werden nun im Place&Route-Ergebnis angegeben (siehe *Fitter Summary*) und mit welcher maximalen Taktfrequenz können diese angesteuert werden (siehe Report der Timinganalyse)?