Technische Universität
Clausthal Institut für Informatik
Prof. G. Kemnitz, Dr. C. Giesemann

23. März 2018

Entwurf digitaler Schaltungen: Aufgabenblatt 2
(Synthese, Logikoptimierung)

Hinweise: Schreiben Sie die Lösungen, so weit es möglich ist, auf die Aufgabenblätter. Tragen Sie Namen, Matrikelnummer und Studiengang in die nachfolgende Tabelle ein und schreiben Sie auf jedes zusätzlich abgegebene Blatt ihre Matrikelnummer.

Name	Matrikelnummer	Studiengang	Punkte von 18

VHDL-Datentypen und -Textausgabe

VHDL besitzt keine Datentypen, sondern Beschreibungsmittel, um Datentypen zu definieren. Es gibt elementare (skalare) und zusammengesetzte Typen. Die elementaren Typen umfassen:

- Zahlentypen,
- Aufzählungstypen und
- physikalische Typen.

Zahlentypen

Zahlentypen sind zusammenhängende Zahlenbereiche, aufsteigend oder absteigend sortiert. Die Definition erfolgt durch Zuordnung des Wertebereichs:

```
type Zahlentyp is range Bereich;
```

Ganzzahlige Wertebereichsgrenzen definieren ganzzahlige Typen (Integer). Kommazahlen als Wertebereichsgrenzen definieren Typen für reelle Zahlen. Untertypen können den Wertebereich weiter begrenzen, aber nicht erweitern. Wertebereichsbegrenzungen sind aber auch bei der Instanziierung möglich:

```
-- ganzzahliger Typ aufsteigend
type t_uint4_up is range 0 to 15;
-- ganzzahliger Typ absteigend
type t_uint4_down is range 15 downto 0;
-- reell aufsteigend
type t_prob is range 0.0 to 1.0;
-- Untertyp zu t_uint4_down
subtype t_u4d is t_uint4_down range 14 downto 1;
-- Variableninstanziierung mit Wertebereichseinschränkung
variable a:t_prob range 0.0 to 0.5;
```

- Für Werte identischer Zahltypen sind u.a. die arithmetischen Operationen +, - * und / definiert. Werte anderer oder unterschiedlicher Typen können nicht ohne Typkonvertierung verknüpft oder zugewiesen werden.

- Untertypen werden bei der Typprüfung wie der Basistyp behandelt.
- Die Konvertierung von einem Zahlentyp in den anderen erfolgt mit dem Typnahmen.
- Die Konvertierung von Zahlenwerten in eine Textdarstellung erfolgt mit dem 'image-Attribut.
- Mehrere Texte können mit dem Konkatenationsoperator & zu einem zusammenhängenden Ausgabetext zusammengefasst werden.
- Die Textausgabe erfolgt mit der Funktion »report«.
- Weitere Attribute, deren Wirkung Sie selbst herausfinden sollen, sind »'left«, »'right«, »'high« und »'low«.

```
1  entity test_zahlentypen is end entity;
2  architecture a of test_zahlentypen is
3   type ta is range -2 to 25;
4   type tb is range 35 downto 5;
5  begin
6   process
7    variable a: ta;
8    variable b: tb :=12;
9   begin
10   report("Anfangswert␣von␣a:␣" & ta'image(a) &
11          "␣Anfangswert␣von␣b:␣" & tb'image(b));
12   a := (ta(b) / 2) - 1;
13   report("(ta(b)␣/␣2)␣-␣1:␣␣␣" & ta'image(a));
14   report(ta'image(ta'left));
15   wait;
16  end process;
17 end architecture;
```

Für den Test installieren Sie auf ihrem Rechner unter Windows oder Linux[1] den Simulator »ghdl«. Alternativ können Sie auch die Rechner im Labor nutzen, auf denen »ghdl« instaliert ist. Dann geben Sie das Programm mit einem Editor (z.B. unter Windows notepad++) ein und speichern es in einem Arbeitsverzeichnis als »test_zahlentypen.vhd«. Weitere Schritte:

- Öffnen einer Konsole (Kommandozeileneingabe). Wechsel in das Arbeitsverzeichnis.
- Analysieren, übersetzen und ablegen in der Arbeitsbibliothek »work« (Analyse):

  ```
  ghdl -a test_zahlentypen.vhd
  ```

- Zusammenbau zu einem ausführbaren Programm (Make):

  ```
  ghdl -m test_zahlentypen
  ```

- Programm ausführen (Run):

  ```
  ghdl -r test_zahlentypen
  ```

[1] Unter Ubuntu 10.04 LTS funktionierte ghdl, unter Ubuntu 14.04 LTS gibt es Datei-Konflikte mit dem gcc. Bei erfolgreicher Installation unter neueren Linuxversionen auf 64-Bit Rechnern bitte E-Mail-Nachricht an den Dozenten.

Aufgabe 2.1: Zahlentypen und -attribute

a) Welche Ausgaben liefert das Programm? 1P

b) Welche Werte haben die Attribute »'left«, »'right«, »'high« und »'low« für die Typen ta und tb im Beispielprogramm[2]? 2P

	'left	'right	'low	'high
ta				
tb				

c) Welche Werte haben die Attribute »'low' und »'high« für die vordefinierten Zahlentpen integer, natural, positive und real? 2P

	integer	natural	positive	real
'low				
'high				

d) Was passiert, wenn die Wait-Anweisung am Ende des Prozesses fehlt? 1P

Aufzählungstypen

Die Definition eines Aufzählungstyps erfolgt durch Zuordnung zulässiger Werte:

```
-- Aufzaehlung symbolischer Werte
type t_wtag is (Mo, Di, Mi, Do, Fr, Sa, So);
-- Aufzählung von Zeichen (des Zeichensatzes)
type t_Zustand is ('0', 'S', 'D', 'X')
```

Die Werte können zugewiesen und gelesen werden. Rechnerintern werden sie durch die Listenposition als natürliche Zahlen dargestellt. Jeder Aufzählungstyp hat u.a. die folgenden Attribute:

'image Konvertierungsfunktion in eine Textdarstellung,

'pos Listenposition,

'val Wert zur Listenposition.

Beispielprogramm:

```
1  entity test_atyp is end entity;
2  architecture a of test_atyp is
3   type t_wtag is (Mo, Di, Mi, Do, Fr, Sa, So);
4   type t_z is ('0', 'S', 'D', 'X');
5  begin
6   process
7    variable w: t_wtag;
8    variable i: integer;
9   begin
10   report("AW von w: " & t_wtag'image(w));
11   w := Do; i := t_wtag'pos(w);
12   report("t_wtag'pos(w): " & integer'image(i));
13   report("t_z'val(i-1): " & t_z'image(t_z'val(i-1)));
14   i := 0;
15   for idx in t_wtag loop -- Ausgabe aller Werte von t_wtag
```

[2] Für »'left« liefert Aufgabenteil a bereits die Lösung und für die anderen erfolgt die Bestimmung analog.

```
16     report(integer'image(i) & " : " &  t_wtag'image(idx));
17     i := i + 1;
18    end loop;
19    for a in boolean loop  -- Erzeugung Wertetabelle fuer a and b
20     for b in boolean loop
21      report(boolean'image(a) & " " & boolean'image(b) & " "
22               & boolean'image(a and b));
23      end loop;
24     end loop;
25    wait;
26   end process;
27 end architecture;
```

Aufzählungstypen und Zahltypen können als Iterationsbereiche für Schleifen verwendet werden. Die Iterationsvariablen sind durch die Iterationsbereiche definiert und außerhalb der Schleife unzugänglich.

Aufgabe 2.2: Aufzählungstypen

a) Welche Ausgaben liefert das Programm? 1P

b) Untersuchen Sie für den im Package »ieee.std_logic_1164« definierten Typ »std_logic« mit den im Programm verwendeten Beschreibungsmitteln die Wertebereiche und leiten Sie daraus die Typdefinition ab[3]. 3P

c) Erzeugen Sie programmtechnisch die Wertetabelle für die Operation »a and b« mit dem Typ »std_logic« für beide Operanden. 3P

Aufgabe 2.3: Erweiterung zur Aufgabe zuvor

Ersetzen Sie im gegebenen Quelltext test_atyp.vhd von Aufgabe 2.2 die Report-Anweisung in der verschachtelten Schleife (Zeile 21 und 22) durch die beiden Anweisungen:

```
report(boolean'image(a) & " " & boolean'image(b));
assert (a and b)=true report "Es ist nicht falsch!";
```

a) Welchen Wert müssen a und b haben, damit die assert Anweisung etwas ausgibt und welche Zeile wird ausgegeben? 1P

b) Probieren Sie anstatt der oben angegebenen Zeile mit assert diese aus:

```
assert (a and b)=true report "Es ist nicht falsch!"
  severity warning;
```

Was hat sich in der Ausgabe geändert? 1P

c) Ändern Sie die assert Bedingung so ab, dass diese immer ausgeführt wird, wenn genau a oder b den Wert true hat und diese Ausgabe "Exklusives-Oder liefert true" erzeugt. 1P

d) Testen Sie die assert-Anweisung aus Teil c) abschließend mit severity failure. Was passiert und warum? 2P

[3] Erfordert die Vereinbarung der Bibliothek »ieee« und eine Use-Anweisung für »std_logic« vor dem Programm.