Entwurf digitaler Systeme
Übungsblatt 2

16. Mai 2017

1. Implementieren Sie einen Volladdierer.
2. Realisieren Sie einen 16-bit Addierer unter Verwendung des Volladdierers über eine strukturelle Beschreibung.
3. Erweitern Sie den Addierer, so dass die Bitbreite generisch angepasst werden kann. Definieren Sie auch einen Standardwert.
4. Passen Sie die Beschreibung des Addierers aus der letzten Übung so an, dass innerhalb der Architektur anstelle der festen Konstanten zur Beschreibung der Bereichsgrenzen Attribute verwendet werden.
5. Erweitern Sie den Addierer nun so, dass er neben der Addition auch die Subtraktion durchführen kann. Verwenden sie dazu die Bildung des Zweierkomplements.
6. Beschreiben Sie den Addierer nun verhaltensorientiert.
7. Erweitern Sie den Addierer nun so, dass das Ergebnis in einem Register gespeichert wird.

```
    process (<clk>,<reset>)
    begin
        if (<reset> = <reset_active_value>) then
            -- reset the sequential elements
        elsif (<clk>'event and <clk> = '1') then
            -- change the elements of the sequential elements
        end if;
    end process;
```

8. Gegeben sein folgendes Verhaltensmodell einer ALU:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity add_sub is
    generic ( WD : integer := 16 );
```

```
        port ( a, b : in std_logic_vector(WD-1 downto 0);
               s : in std_logic_vector(1 downto 0);
               o : out std_logic_vector(WD-1 downto 0));
    end add_sub;
12  architecture Behavioral of add_sub is
    begin
        process (a,b,s)
        begin
            if (s = "10") then
17              o <= a+b;
            elsif (s = "01") then
                o <= a-b;
            elsif (s = "11") then
                o <= not a;
22          end if;
        end process;

        process (o)
        begin
27          if (clk'event and clk = '1') then
                s <= o;
            end if;
        end process;

32  end Behavioral;
```

- Übernehmen Sie die ALU von der Vorlesungswebseite in Ihr Projekt
- Was fehlt in der Beschreibung bzw. was ist eventuell inkorrekt?
- Korrigieren Sie die groben syntaktischen und semantischen Fehler und synthetisieren Sie die Schaltung. Wie sieht das Ergebnis der Synthese aus?
- Implementieren Sie den Addierer mit einem Überlaufbit.

9. Verifizieren Sie die Funktionsweise der ALU. Legen Sie dazu eine Testbench (TB) wie folgt an:
   - Rechtsklick im Source-Browser → New Source...
   - Wählen Sie VHDL Test Bench, geben Sie einen Namen ein, assoziieren Sie die TB mit der ALU-Entität und schließen Sie das Erstellen der TB ab.
   - Applizieren sie diverse Eingaben an den Eingangsports der ALU.
   - Simulieren Sie die Testbench:
     - Wählen Sie dazu oberhalb des Source-Browsers Behavioral Simulation
     - Selektieren Sie die TB im Source-Browser
     - Expandieren Sie Xilinx ISE Simulator im Process-Browser und doppelklicken Sie auf Simulate Behavioral Model