Practicum 3.1: meer 7-segment displays

1. Start een nieuw project, voeg een VHDL file toe (bijvoorbeeld “decimals.vhd”) met de volgende entity:

```
-- decimals
-- input: 8-bit number 0 to 255
-- output: 3 bcd codes for hundreds, tens and ones
entity decimals is
port ( int : in std_logic_vector(7 downto 0);
       bcd100, bcd10, bdc1 : out std_logic_vector(3 downto 0)
     );
end entity decimals;
```

2. Maak een architecture voor deze entity die een integer omzet in hondertallen, tientallen en eenheden. De berekeningen om de factoren te krijgen zijn:

```
u100 <= uint / 100;
u10  <= (uint / 10) rem 10;
u1   <= uint rem 10;
```

   Let op dat deze berekeningen niet in een std_logic_vector kunnen. Converteer eerst int naar een unsigned en daarna de resultaten weer naar een std_logic_vector!

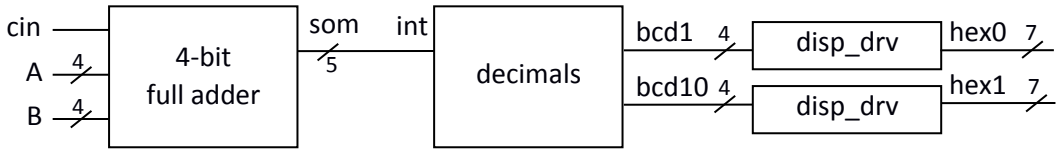
3. simuleer het ontwerp met een testbench.

4. Voeg een VHDL file toe die een *decimals* component en drie *disp_drv* componenten (uit het vorige practicum) gebruikt om een decimaal getal correct op HEX2, HEX1 en HEX0 te tonen. Bijvoorbeeld: het getal 123 geeft 1 op HEX2, 2 op HEX1 en 3 op HEX0. Sluit de input int van *decimals* aan op schakelaars sw7 t/m sw0. Gebruik port mapping voor deze VDHL file. Vergeet niet de top-level entity te zetten.

5. compileer, programmeer en test.

Practicum 3.2: full adder en 7-segment displays

6. Voeg een VHDL file toe die een 4-bit ripple full adder component, een *decimals* component en twee *disp_drv* componenten gebruikt om de som van twee 4-bits getallen (en een carry-in) op HEX1 en HEX0 te tonen. Gebruik port mapping voor deze VDHL file.



7. compileer, programmeer en test op het DE0 board;