Practical 5 / *Prakties 5*

Hierdie opdrag moet gedemonstreer word vir die dosent of 'n demi voor die einde van jou groep se sessie, en 'n punt van 0, 1, of 2 sal toegeken word. U *.vhd leêr moet ook op SUNLearn gelaai word voor 17h00 16 April 2015.

*This assignment must be demonstrated to the lecturer or a demi before the end of your group's session, and a mark of 0, 1, or 2 will be awarded. Your *.vhd file must also be uploaded onto SUNLearn before 17h00 16 April 2015.*

Komponente - *Components*

Elke student moet 'n DE0 bordjie uitteken en hou tot die einde van die semester. As die bordjie aan die einde van die semester nie teruggegee word in dieselfde toestand as waarin dit uitgeteken is nie, sal daar geen punte vir hierdie module vir die betrokke student toegeken word nie.

Werk op jou eie. Elke student moet sy/haar eie kode skryf en op SUNLearn oplaai.

Every student must sign out a DE0 board and keep it until the end of the semester. If the board is not returned at the end of the semester in the same condition as it was signed out, no marks will be given to the student who signed it out.

Work on your own. Each student must write his/her own code and upload it onto SUNLearn.

Opdragte - *Tasks*

Laai die projek templaat op SUNLearn af en unzip dit. Maak die projek oop deur op die .qpf leêr te dubbel-kliek. Implementeer al the vrae in die bestaande argitektuur, en programmeer die DE0 bordjie vir demonstrasie.

1. Skryf VHDL kode om 'n omkeerder te implementeer deur gebruik te maak van 'n enkele *concurrent statement*. Die omkeerder neem *Button 0* as intree, en die uittree word vertoon op *LED 0*.

2. Skryf VHDL kode om 'n twee-intree XOF-hek te realiseer deur gebruik te maak van 'n eenvoudige *concurrent statement*. Gebruik *Button 0* en *Button 1* vir die intrees, en *LED 1* vir die uittree.

Download and unzip the project template on SUNLearn. Open the project by double clicking on the .qpf file. Implement all the questions in the existing architecture, and program them onto the DE0 board for demonstration.

1. *Write VHDL code to implement an inverter using a single simple concurrent statement. The inverter takes Button 0 as input, and the output is given on LED 0.*

2. *Write VHDL code to implement a two-input XOR gate byt using a simple concurrent statement, which takes Button 0 and Button 1 as inputs, and presents the output on LED 1.*

3. Die projek templaat bevat reeds die *altera_primitives_components* biblioteek. In hierdie biblioteek bestaan daar 'n D-wipkring komponent met clear en preset, met die volgende koppelvlak:

3. *Die project template already includes the altera_primitives_components library. In this library there is a D flip-flop with clear and preset, with the following interface:*

```
component dff
  port(
    d, clk, clrn, prn :  in  std_logic;
    q                 :  out std_logic);
end component;
```

Figure 1: The port definition of the D flip-flop in the altera_primitives_components library.

Gebruik 'n *structural assignment* (PORT MAP) om 'n wipkring as volg te verbind:

Use a structural assignment (PORT MAP) to connect a flip-flop as follows:

- The **inverse** of *Button 0* is connected to the D input;
- The **inverse** of *Button 1* is connected to the CLK input;
- *Button 2* is connected to the CLRN (note that the N means negative logic);
- PRN is connected to '1' (note that the N means negative logic);
- Q is connected to *LED 2*.

Bevestig die wipkring se gedrag deur die knoppies te druk en die uittree op die LED te vergelyk met die tabel:

Confirm the behaviour of the flip-flop by using the buttons and comparing the output on the LED with the table:

Inputs				Output
prn	clrn	clk	d	q
0	0	x	x	N/A
0	1	x	x	1
1	0	x	x	0
1	1	Rising edge	0	0
1	1	Rising edge	1	1
1	1	otherwise	x	Q

Figure 2: The characteristic table of a D flip-flop with preset and clear.

4. Gebruik 'n *concurrent selected signal assignment* om *Switch 3 downto 0* as 'n 4-bis ongetekende binêre getal te interpreteer. Hierdie getal moet vertoon word as 'n heksadesimale karakter op die 7-segment vertooneenheid aan die regterkant. Die *concurrent selected signal assignment* behoort soortgelyk te wees aan:

4. *Use a concurrent selected signal assignment to interpret Switch 3 downto 0 as a 4-bit unsigned binary number. This number must be displayed as a hexadecimal character on the right-most 7-segment display. The concurrent selected signal assignment should be similar to:*

```
WITH SW(3 DOWNTO 0)  SELECT
        HEX0_D <= "0000001"   WHEN   "0000",
                  ____   WHEN ____
                  ____   WHEN ____;
```

Figure 3: Example of the form of a concurrent selected signal assignment.

Bevestig elke binêre getal van "0000" tot "1111" deur van die skakelaars gebruik te maak, en maak seker dat die ooreenstemmende heksadesimale getalle vertoon word.

Confirm each binary number from "0000" to "1111" by using the switches, and make sure that the correct hexadecimal number is shown for each binary number.

5. Skryf VHDL kode en gebruik 'n *PROCESS* om die 50 MHz kloksein te verdeel na 'n 1 Hz kloksein. Gebruik die 1 Hz sein om *LED 3* te laat flikker teen 1 Hz. U sal 'n SIGNAL benodig vir die resulterende klok (gedefinieer buite die *PROCESS* maar binne die *ARCHITECTURE*) en 'n *VARIABLE* om te tel (intern tot die *PROCESS*).

5. *Write VHDL code and use a PROCESS to divide the 50 MHz clock down to a 1 Hz signal. Use the downscaled clock to flash LED 3 at 1 Hz. You will need a SIGNAL for the resulting clock (declared external to the PROCESS, but inside the ARCHITECTURE) and a VARIABLE to count (internal to the PROCESS).*

6. Skryf VHDL kode om die twee linker 7-segment vertooneenhede te laat tel van 00 na 99 teen 1 Hz (gebruik die klok van die vorige vraag), waarna dit oorrol en weer van voor af begin tel. U sal moontlik die *REM* operator wil gebruik, wat die res van 'n deling gee. As 'n opsionele ekstra, gebruik Switch 9 om te bepaal of die teller optel of aftel.

6. *Write VHDL kode to make the two left-most 7-segment display units count from 00 to 99 at 1 Hz (use the clock of the previous question), whereafter it should roll over and start from 00 again. You may want to use the REM operator, which gives the remainder of a division. As an optional extra, use Switch 9 to determine whether the counter should count up or down.*

Onthou om jou kode op SUNLearn op te laai voor 17h00 16 April 2015!

Remember to upload your code to SUNLearn before 17h00 16 April 2015.