CentraleSupélec

ICS M2 Degree - 2021-2022 - Advanced Digital Electronics

Practical Session N°3
Realization of sequential operators

We want to create a clock divider, that is to say that starting from a clock at a frequency f we obtain a clock at a frequency f / N, N being an integer defined in advance.

First, we will make a simple frequency divider with an even division factor = 2N.

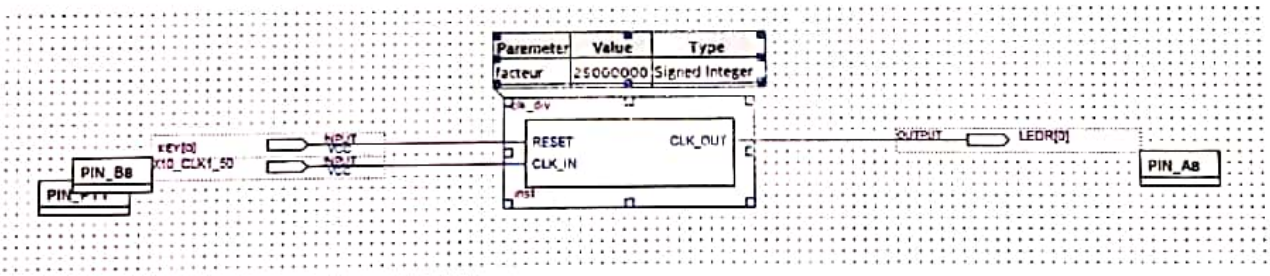
We are going to create an entity called clock_divider. The number N will be a generic parameter of the entity provided as an integer.
The clock divider will be achieved by means of a countdown counter from (N-1) to 0.
Each time the counter returns to 0, its output is inverted.

For our application, this clock divider will be used in an Altera DE10-lite board which has a clock at 50 Mhz. We will choose N = 25,000,000 so as to obtain a clock at 1 Hz. This clock will be connected to one of the LEDs on the board in order to visualize the oscillation.

1 °) Opening of the project

We are still working in BE_ALTERA in the TD3 subdirectory.

You launch quartus and you open the clock_divider project. The project contains an already prepared clk_div module containing the entity. You will write the architecture.



2°) Coding the clock divider

The divider will be described in a synchronous process controlled by the input clock and the reset signal.
The half division factor is defined by a generic parameter of the entity: factor
We will use 2 signals: 1 integer signal which serves as a counter (clkcnt), and a binary signal which codes the output (clkout).
The algorithm is as follows
At each clock edge
  If clkcnt = 0 then
    Clkcnt = factor-1
    Clkout = not clkout
  else
    Clkcnt = clkcnt-1

As in the previous BE, you write the code, you check it, then you build the project and you test on the board.

→ 1 kHz blink

3 °) Code analysis:
What is the logic instantiated by the line: "Clkcnt = clkcnt-1"
Same question for the line "Clkcnt = factor-1"

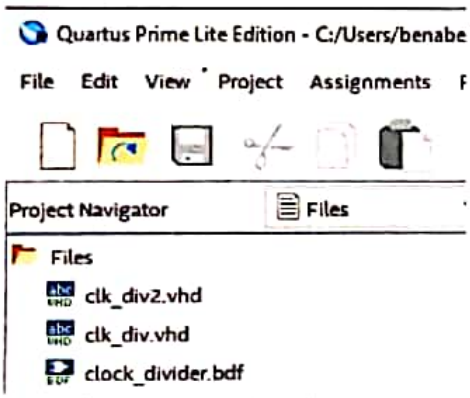
} repeated recurrent

substract factor by 1 each clk count

4 °) We now want to create a clock divider whose factor is programmable (it becomes an input for the entity) and we remove the parity condition from the division factor. The input factor becomes the actual division factor. In fact the effective division factor will be factor + 1.
The counter will now be encoded using an unsigned signal.

Copy the cld_div.vhd file to clk_div2.vhd

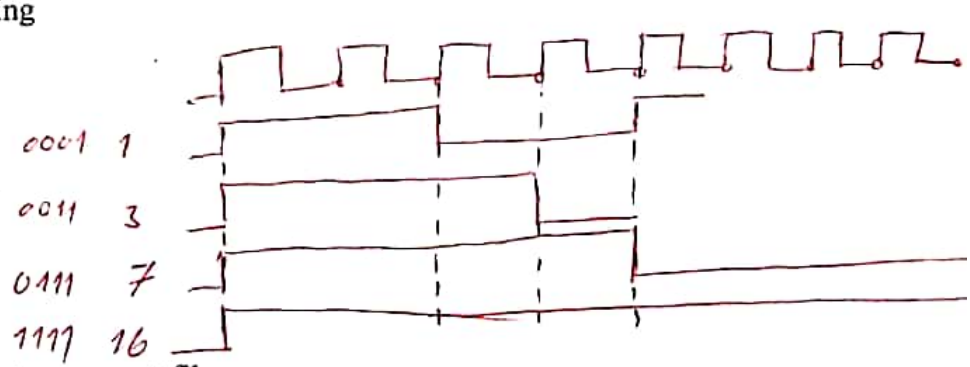
Integrate the new file into the project using the project -> add_remove file in project menu
Edit the file in quartus by double_clicking on its name in the project navigator



range
SW[3..0]
name
1 Hz
value
initialization
of a signal → reset
SW 0
SW 1
SW 2
SW 3
} /2 1/2

Edit the file accordingly
- Change of the name of the entity
- Factor becomes an input (we choose to encode it on 4 bits in order to have a division factor ranging from 1 to 16 by shifting the value by one (2 is represented by 1, ... 16 is represented by 15))
- Clkcnt becomes a 4-bits unsigned type vector
- The algorithm becomes the following

```
If clkcnt = 0 then
        Clkcnt = factor
        Clkout = 1
else if clkcnt = factor / 2 + 1 then
        Clkout = 0
else
        Clkcnt = clkcnt-1
```



Analyze the code: Processing -> Analyze current file
Create the symbol of the new divider File -> Create / Update -> create symbol for current file

Add the new divider after the previous one and connect its output to LEDR [1]
Then compile the project download and test.

5 °) Subsidiary question
Does the preceding code work if factor = 0?
What would have to be done to make the divisor work even for this value?

8 4 2 1 10
16 8 9 2
9 5 3 0

clk_out <= clk_in when factor = "0000" else
clk_out ;