1 Introduction

In the last exercise you learned about the first part of the FPGA design flow, the simulation. Once a system confirms the design specifications, it can be implemented on an FPGA. This implementation on the FPGA is the second part of the FPGA design flow. It consists of the synthesis (1), placement and route (2) and the generation of the bit-file (3) which is downloaded to the FPGA (see figure 1). In this exercise, a blinking light will be implemented on the GECKO4-Education.

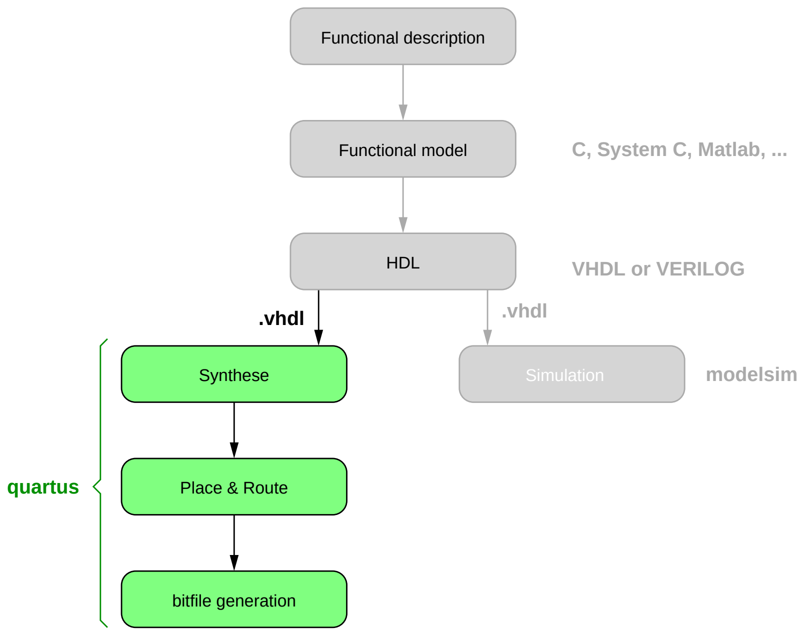


Figure 1: Second part of the FPGA design flow

2 The blinking light

You will find the file `clignotante.zip` on moodle. Uncompress this file inside a folder on your file system, which contains no space character in the folder name and the entire path. Once the zip is uncompressed, you will find four folders. The folders `modelsim`, `quartus`, and `scripts` are empty. The folder `vhdl` contain six files which have the following functionality:

- `tff_sr-entity.vhdl` and `tff_sr-behavior-generic.vhdl`: This is the same finite state machine as the one you simulated in the previous exercise.

- `divider-entity.vhdl` and `divider-behavior.vhdl`: Here, a finite state machine of the type Moore is implemented. This FSM represents a counter.
- `laboratory6_top-entity.vhdl` and `laboratory6_top-behavior.vhdl`: This is the top level design file of the blinking light. In this file the complete functionality is instantiated.

Exercise 1:

Review the file `divider-behavior.vhdl` and try to understand its functionality. Answer all the questions asked in this file.

Exercise 2:

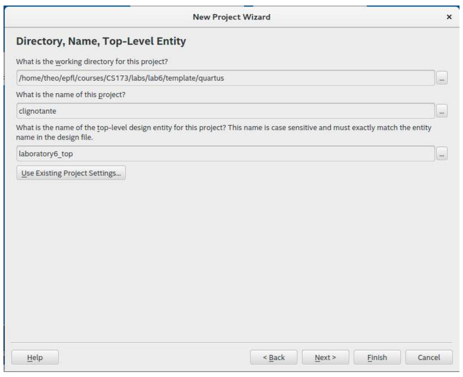
Review the file `laboratory6_top-behavior.vhdl` and draw the block diagram of the entire system.

3 The synthesis, placement and route, and the generation of the bit file

For the synthesis, placement and route, and the generation of the bit file, we will use Quartus. Start Quartus by (1) clicking on the Quartus pictogram in Windows, or (2) by inserting the USB stick into your computer, opening a terminal and launching Quartus with the command `Altera quartus`. Once Quartus is launched, we will create a new project as following:

1. Create a new project with `File > New Project Wizard`.
2. Click the button `Next >`.
3. In the line `What is the working directory for this project?` select the folder `quartus` of the uncompressed `clignotante.zip` file.
4. Name the project `blinking_light` in the line:
   `What is the name of this project?`.
5. Name the entity `laboratory6_top` in the line:
   `What is the name for the top-level design ....`
6. The window should look like the figure 2a. Click the button `Next >`.

7. In the following window, select `Empty project` and click `Next >`.
8. In the following window, click `...` in the line `File name`. Navigate to the folder `vhdl` and select all vhdl files. Click `Open`.
9. The window should look like the figure 2b. Click `Next >`.



New Project Wizard

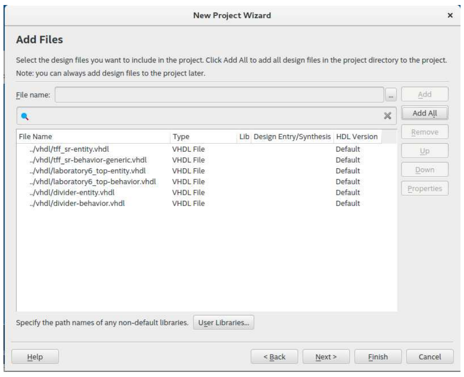
Directory, Name, Top-Level Entity

What is the working directory for this project?
/home/theo/epfl/courses/CS173/labs/lab6/template/quartus

What is the name of this project?
clignotante

What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.
laboratory6_top

(a) Create a project



New Project Wizard

Add Files

Select the design files you want to include in the project. Click Add All to add all design files in the project directory to the project.
Note: you can always add design files to the project later.

File name:

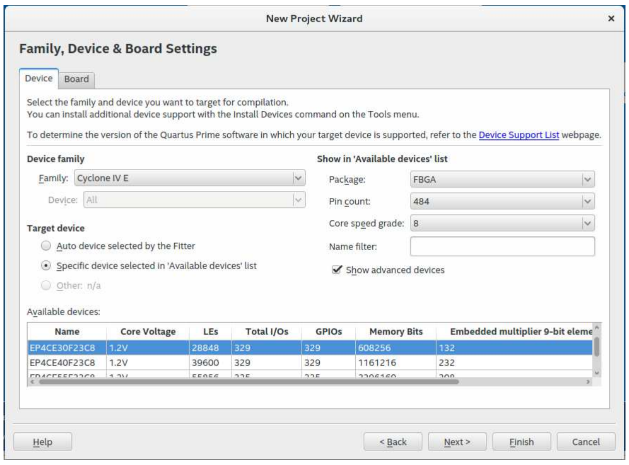
File Name	Type	Lib	Design Entry/Synthesis	HDL Version
../vhdl/tff_sr-entity.vhdl	VHDL File			Default
../vhdl/tff_sr-behavior-generic.vhdl	VHDL File			Default
../vhdl/laboratory6_top-entity.vhdl	VHDL File			Default
../vhdl/laboratory6_top-behavior.vhdl	VHDL File			Default
../vhdl/divider-entity.vhdl	VHDL File			Default
../vhdl/divider-behavior.vhdl	VHDL File			Default

Specify the path names of any non-default libraries.

(b) Add files

Figure 2: Create a project in Quartus.

10. In the following window, select `Cyclone IV E` in the box `Family`, `FBGA`in the box `Package`, `484` in the box `Pin Count`, `8` in the box `core speed grade`, and `EP4CE30F23C8` in the box `Available devices`.
11. The window should look like the figure 3.Click `Finish`.



New Project Wizard

Family, Device & Board Settings

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.

To determine the version of the Quartus Prime software in which your target device is supported, refer to the Device Support List webpage.

Device family

Family: Cyclone IV E
Device: All

Target device

Auto device selected by the Fitter
[✓ Specific device selected in 'Available devices' list]
Other: n/a

Show in 'Available devices' list

Package: FBGA
Pin count: 484
Core speed grade: 8
Name filter:
[✓ Show advanced devices]

Available devices:

Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Embedded multiplier 9-bit eleme
EP4CE30F23C8	1.2V	28848	329	329	608256	132
EP4CE40F23C8	1.2V	39600	329	329	1161216	232
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Figure 3: Selection of the FPGA in use

Quartus is now configured to synthesize the system, but it still lacks the information to place the different connections from your top level to the FPGA's IO's. This information depends on the board used, in our case the GECKO4Education . Open a web browser and go to *http://gecko.microlab.ch*. Click on the image of the GECKO4Education. To find the required information, we will inspect each of the inputs and outputs of our top level:

- **clock**. This is the main clock of the system, which means we need to provide a clock signal. Click the tab `clocks` in the web browser to open the information about the available clock signals on the board. There are two clocks, a 12 MHz clock and a 50 MHz clock. We will use the 50 MHz clock For the blinking light. We can see in the table, that the 50 MHz clock is connected to the T1 pin of the FPGA. To simplify our lives, we can use the tcl script files to assign the pins in Quartus. To do this, click `Here` in the opposite of `you find an example....` and download the tcl script file to the `scripts` folder. Open the just downloaded file (clocks.tcl) and insert a # at the start of the line of the 12 MHz clock. The # indicates a comment line. Replace `<ENTITY_PORT_NAME_CONNECTED_TO_CLOCK_50MHZ>` by `clock` to indicate Quartus that the `clock` input of your system needs to be connected to the pin T1 of your FPGA.
  Now, click `here` in the opposite of `the corresponding ...` and download the file to the `quartus` folder. Open the just downloaded file (clocks_sdc.tcl) and insert a # at the start of the 12 MHz clock line. Replace `<ENTITY_PORT_NAME_CONNECTED_TO_CLOCK_50MHZ>` by `clock`. This file indicates Quartus the frequency and signal type of the clock signal. This information is needed to optimize the system in terms of speed and/or to verify if the system can operate at the selected clock frequency.

- **n_reset**. This is the reset signal of the system. This signal is low active. It means that if the reset is not activated, a logic **1** is given. If the reset is activated, a logic **0** is given. In vhdl, we indicate a low active signal often with the prefix `n_` or with the suffix `_bar`. A button is used for the reset signal. Click the tab `buttons` in your web browser. Click `here` in the opposite of `you find an ....` in the section `Buttons(s) information` and download the file to the `scripts` folder. Open the just downloaded file (switches.tcl) and place # in the beginning of the lines SW2..SW7.
  Replace `<ENTITY_PORT_NAME_CONNECTED_TO_SW1>` by `n_reset`.

Exercise 3:

Proceed with the same procedure as for the clock and buttons for the **LED** pins. You can use an LED of the LED-array or one segment out of a seven segment display. You can find the information needed in the `leds` tab in your web browser.

IMPORTANT:

The TCL scripts are case sensitive. This means you need to use the same uppercase or lowercase letters for the signal names in the TCL script as you did in the entity.

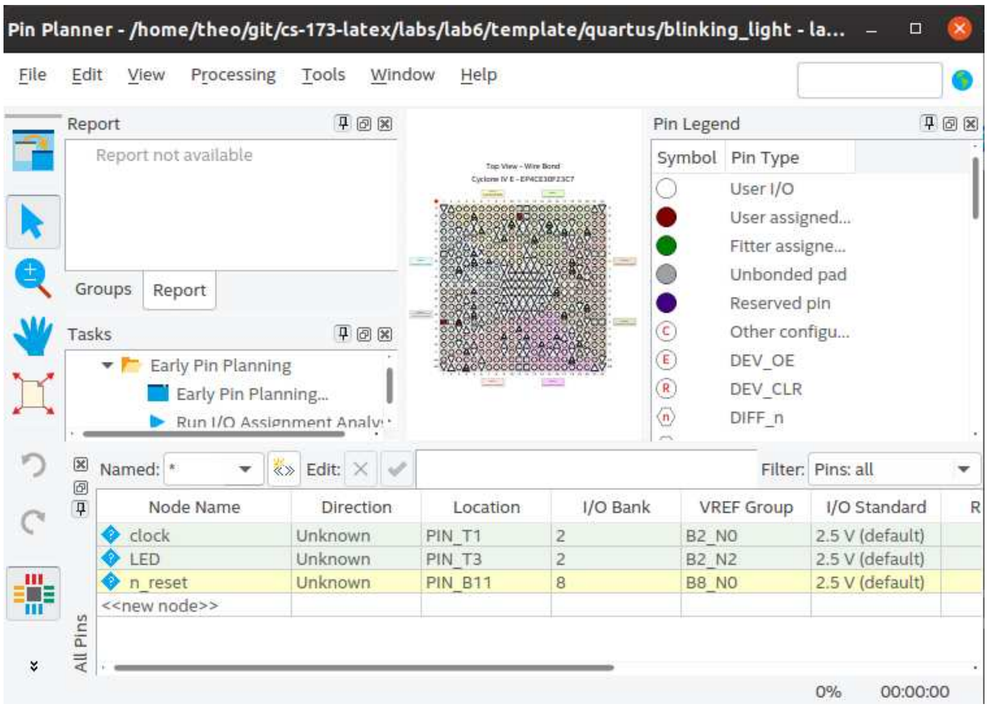
To activate the TCL scripts in Quartus, proceed as following for each script file in the `script` folder:

1. Click `Tools > Tcl Scripts`.
2. Click the `Add to Project` button.
3. Select all tcl files int the `scripts` folder and click the `Open` button.
4. In the tcl window, click on each tcl script, and apply `Run`.

To verify that everything went well, open the pin-planner by `Assignments > Pin Planner`. For each input and output there should be an entry in the tab `Location`. The pin-planner should look like the figure 4.

IMPORTANT:

All inputs and outputs need to be connected to a pin. If not, Quartus will automatically assign the inputs and outputs to a pin. Since Quartus doesn't know about which pin is connected to which element on the Board it can happen that two outputs are connected together. This causes short cuts and finally the holy smoke.


Pin Planner - /home/theo/git/cs-173-latex/labs/lab6/template/quartus/blinking_light - la...

File Edit View Processing Tools Window Help

Report

Report not available

Groups Report

Tasks

Early Pin Planning
Early Pin Planning...
Run I/O Assignment Analy

Pin Legend

Symbol	Pin Type
○	User I/O
●	User assigned...
●	Fitter assigne...
●	Unbonded pad
●	Reserved pin
Ⓒ	Other configu...
Ⓔ	DEV_OE
Ⓡ	DEV_CLR
ⓝ	DIFF_n

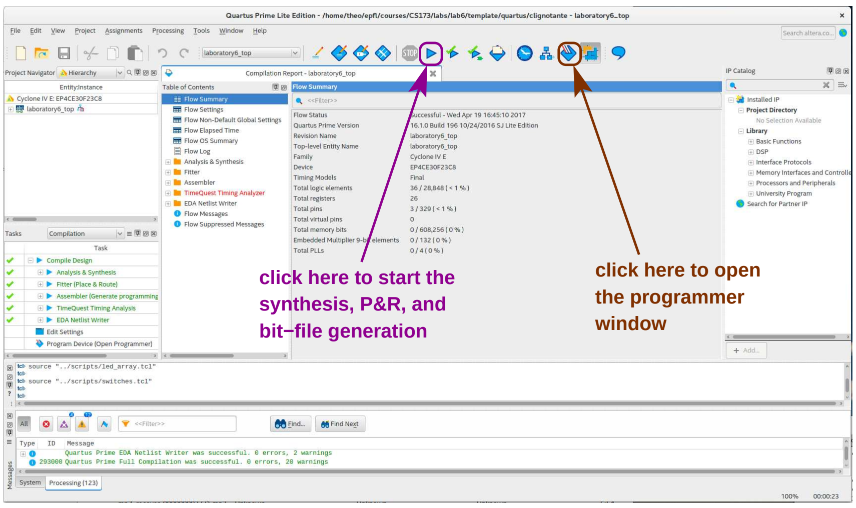
Named: * Edit: Filter: Pins: all

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	R
clock	Unknown	PIN_T1	2	B2_N0	2.5 V (default)	
LED	Unknown	PIN_T3	2	B2_N2	2.5 V (default)	
n_reset	Unknown	PIN_B11	8	B8_N0	2.5 V (default)	
<<new node>>						

0% 00:00:00

Figure 4: Pin assignments in the pin-planner.

If everything went well, you can now close the pin-planner and continue with the next steps. If not, repeat the procedures described above.
Quartus is now configured to execute the synthesis, place and route and the generation of the bit file. Click the indicated symbol displayed in figure 5 to start the synthesis, P&R and generation of the bit file. Now you need to wait until the process finishes. You can increase the speed of the process by connecting your laptop to the wall and change your battery saving settings to full power.


click here to start the synthesis, P&R, and bit−file generation
click here to open the programmer window
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate programming
TimeQuest Timing Analysis
EDA Netlist Writer
Edit Settings
Program Device (Open Programmer)

Quartus Prime Lite Edition - /home/theo/epfl/courses/CS173/labs/lab6/template/quartus/clignotante - laboratory6_top

Compilation Report - laboratory6_top

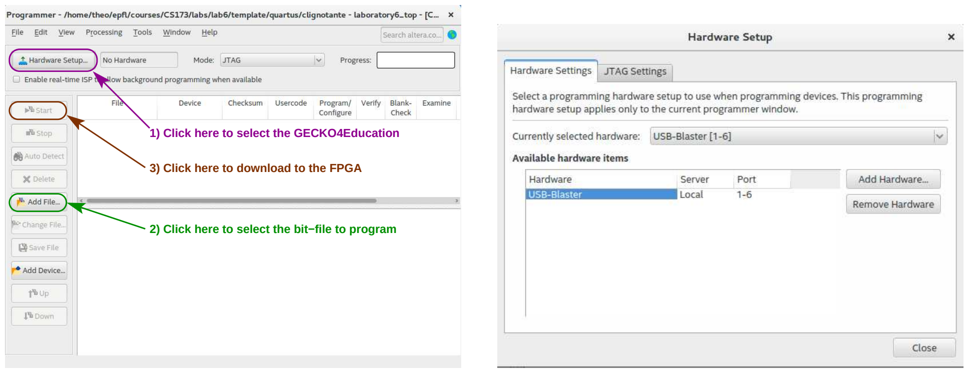
Flow Summary

Flow Status	Successful - Wed Apr 19 16:45:10 2017
Quartus Prime Version	16.1.0 Build 196 10/24/2016 SJ Lite Edition
Revision Name	laboratory6_top
Top-level Entity Name	laboratory6_top
Family	Cyclone IV E
Device	EP4CE30F23C8
Timing Models	Final
Total logic elements	36 / 28,848 (< 1 %)
Total registers	26
Total pins	3 / 329 (< 1 %)
Total virtual pins	0
Total memory bits	0 / 608,256 (0 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)

```
tcl> source "../scripts/led_array.tcl"
tcl>
tcl> source "../scripts/switches.tcl"
tcl>
tcl>
```

```
Type  ID     Message
             Quartus Prime EDA Netlist Writer was successful. 0 errors, 2 warnings
      293000 Quartus Prime Full Compilation was successful. 0 errors, 20 warnings
```

Figure 5: Quartus main window.



Programmer - /home/theo/epfl/courses/CS173/labs/lab6/template/quartus/clignotante - laboratory6_top - [C...

Select a programming hardware setup to use when programming devices. This programming hardware setup applies only to the current programmer window.

Currently selected hardware: USB-Blaster [1-6]

Available hardware items

Hardware	Server	Port
USB-Blaster	Local	1-6

(a) Programming window (b) Board selection

Figure 6: Download a bit file to the FPGA

Once Quartus finished, we can download the bit file to the FPGA. To do this, open the programmer by clicking the icon showed in the figure 5. The programmer window will open (like in figure 6a). To download the bit file to the FPGA, proceed as following:

1. Connect the GECKO4Education to your computer.
2. Click the configuration button to select the right board in the programmer window (look at 1 in figure 6a). The configuration window should look like the figure 6b. Click two times on `USB-Blaster` to select the GECKO4Education and close this window.

3. Click the button `Add File ...` (see 2 in figure 6a). Navigate to the folder `output_files` and select the file `laboratory6_top.sof`. Click `Open`.

4. Click `Start` (see 3 in figure 6a).

Now, the FPGA is configured and the blinking light should start to blink with a frequency of 1 Hz.

Exercise 4:

Modify the parameter of the clock such as the clock is connected to the 12 MHz clock source. Regenerate the bit file and download it to the GECKO4Education. What do you expect to observe?

IMPORTANT:

It is important that you understand and that you can fully reproduce all the explained steps in this exercise. In the next exercises and in the following semester we expect you to use Modelsim and Quartus on your own.

Exercise 5:

Draw the output signal of the counter qualitatively. How many clock periods will the signal be 1 and how many periods will it be 0?

Exercise 6:

Simulate the entire system in Modelsim.

Exercise 7:

Modify the VHDL description such as the blinking light blinks with a frequency of 10 Hz using the 12 MHz clock.