During the laboratory practice we will use *Altera Quartus II* development environment software and a *DE0* development board. We will learn the steps of synthesis through the RTL model of a counter which was examined on the previous lab.

```
5  entity counter is
6      port (clk:            in  std_logic;
7            reset_n:        in  std_logic;
8            enable:         in  std_logic;
9            direction:      in  std_logic;
10           parallel_in:    in  std_logic_vector (7 downto 0);
11           load_n:         in  std_logic;
12           cout:           out std_logic_vector (7 downto 0));
13 end entity counter;
14
15 architecture rtl of counter is
16
17     signal delay_counter:    integer range 0 to 50000000 := 0;
18     signal enable_increment: std_logic := '0';
19     signal counter:          std_logic_vector (7 downto 0) := (others => '0');
20
21 begin
22
23     L_PRE_SCALER: process ( clk )
24     begin
25         if ( rising_edge(clk) ) then
26             if ( delay_counter = 50000000 ) then delay_counter <= 0;
27                                                  enable_increment <= '1';
28             else delay_counter <= delay_counter + 1;
29                  enable_increment <= '0';
30             end if;
31         end if;
32     end process;
33
34     L_COUNTER: process (clk, reset_n)
35     begin
36         if ( reset_n = '0' ) then counter <= (others => '0');
37         elsif ( rising_edge(clk) ) then
38             if ( enable = '1' ) then
39                 if ( load = '1' ) then counter <= parallel_in;
40                 elsif ( enable_increment = '1' ) then
41                     if ( direction = '1' ) then counter <= std_logic_vector(unsigned(counter) + 1);
42                     else counter <= std_logic_vector(unsigned(counter) - 1);
43                     end if;
44                 end if;
45             end if;
46         end if;
47     end process;
48
49     L_OUTPUT: cout <= counter;
50
51 end architecture rtl;
```

*It is an 8-bit counter*

*The load signal is changed to an active-low*

*Pre-scaler circuit*
*It is a frequency divider*
*It convert the 50MHz clock signal to a 1Hz signal.*

*Figure 1. Modifications of VHDL model*

The VHDL model of the counter is modified to fit the hardware environment of the *Altera DE0* development board. The modifications are the followings:

- The board has 10 switches: one for global enable, one is the direction of the counter (up or down). The other 8 switches set the initial value of the counter.
- The lower 8 LEDs indicate the actual value of the counter.
- The parallel load can be activated by one of the buttons.
- On the board a 50 MHz oscillator is placed. In order to decrease the frequency to a few Hz a pre-scaler is inserted.

The synthesis starts with creating a new project. Use the ***File*** menu ***New Project Wizard…*** command. Just press the ***Next*** button, name the project and set the location. The name of the top-level design has to be exactly **counter**. *(toplevel)* (see Figure 2.).

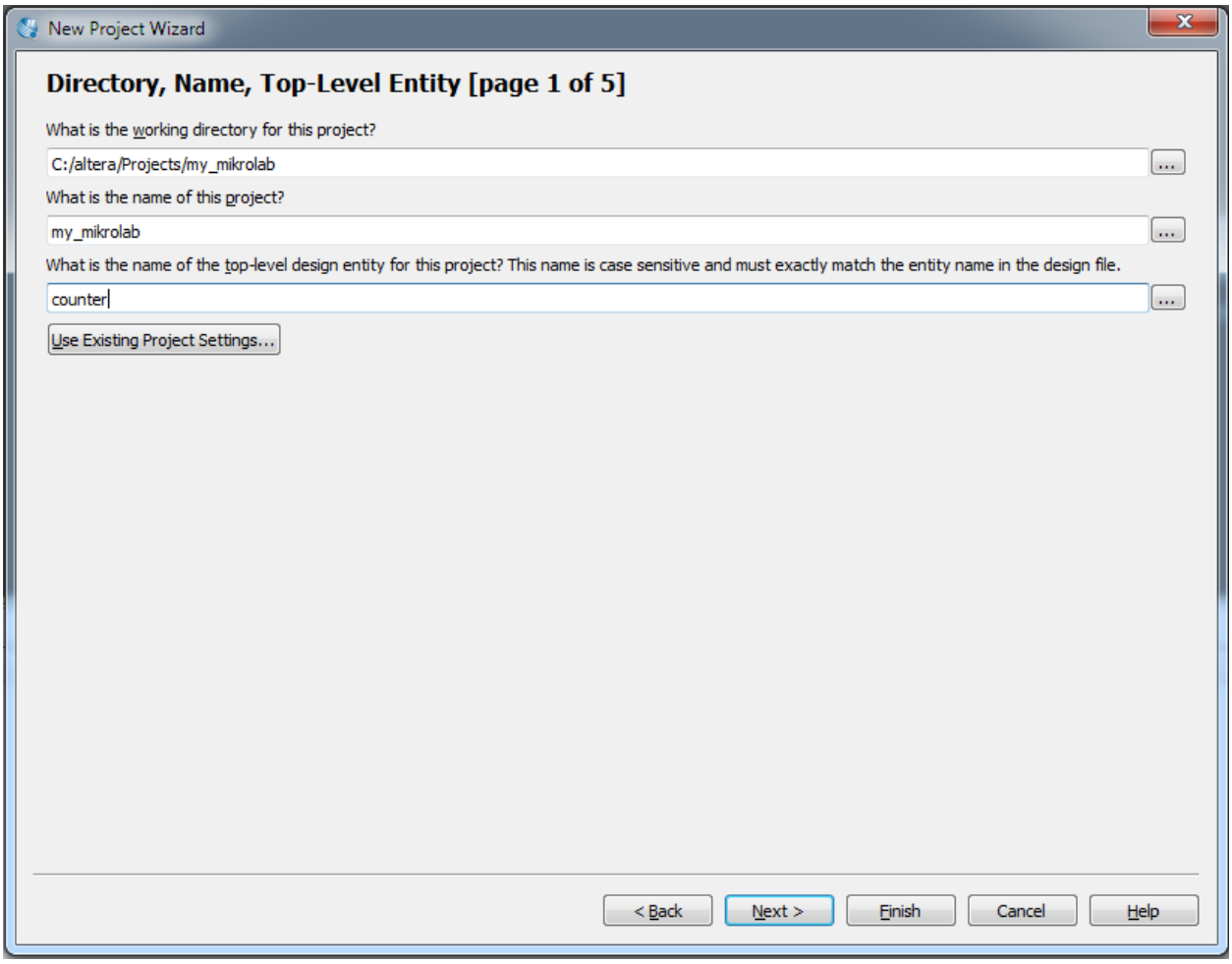


*Figure 2. The name and the place of the new project, and the name of the top-modul*

It is beneficial to place into the *Home* folder.

In the subsequent step we can add existing files. After browsing for *counter.vhd* press ***Add*** button (Figure 3.).

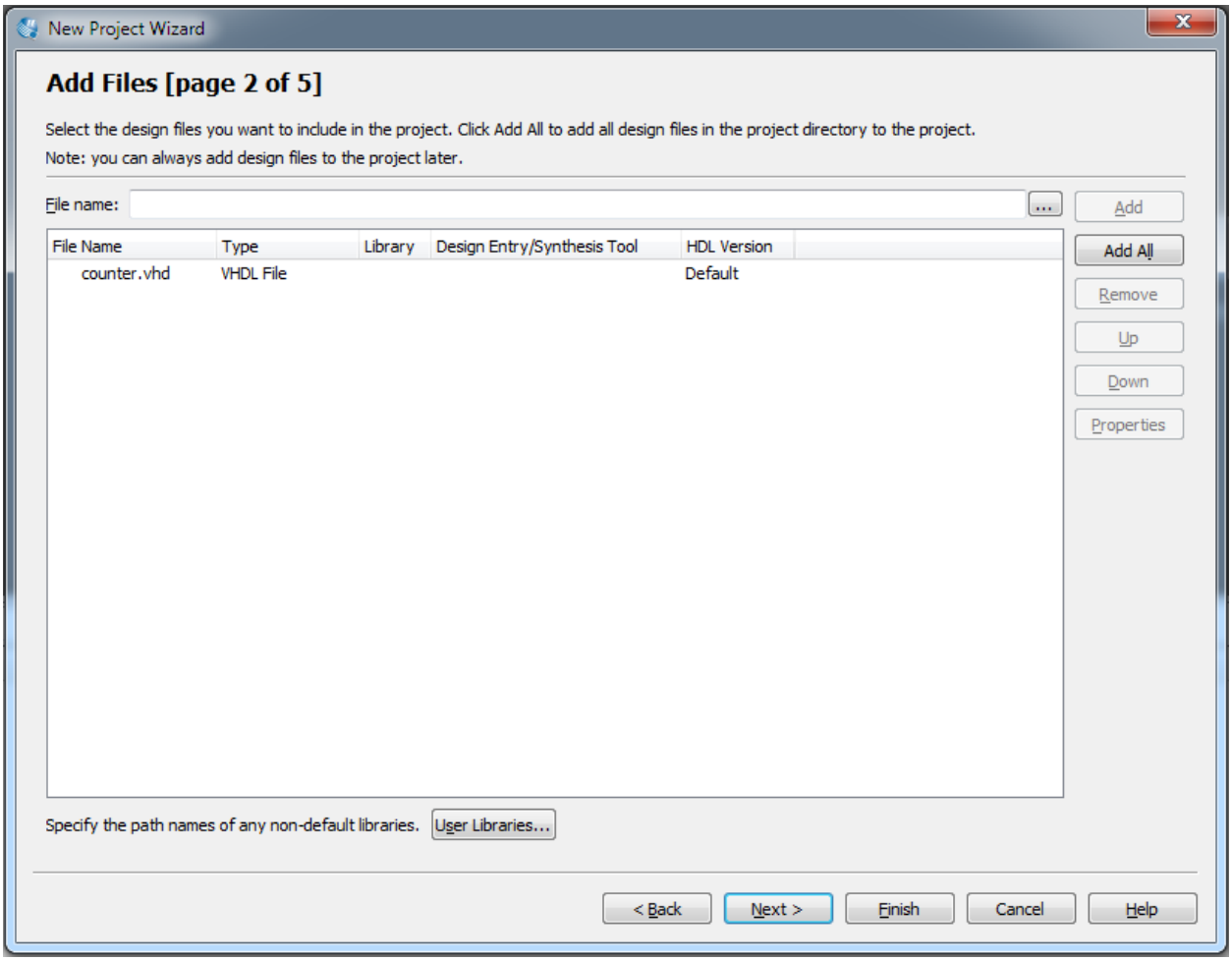


*Figure 3. Add files window*

After pressing the ***Next button***, we can select the FPGA. The type of the device is ***EP3C16F484C6.*** Please use the filter to select the proper one: the ***Family*** *is* ***Cyclone III***, the ***Package*** is ***FBGA*** *(Flip-chip Ball Grid Array)*, the ***Pin count*** is ***484,*** and the ***Speed grade*** is ***6***.

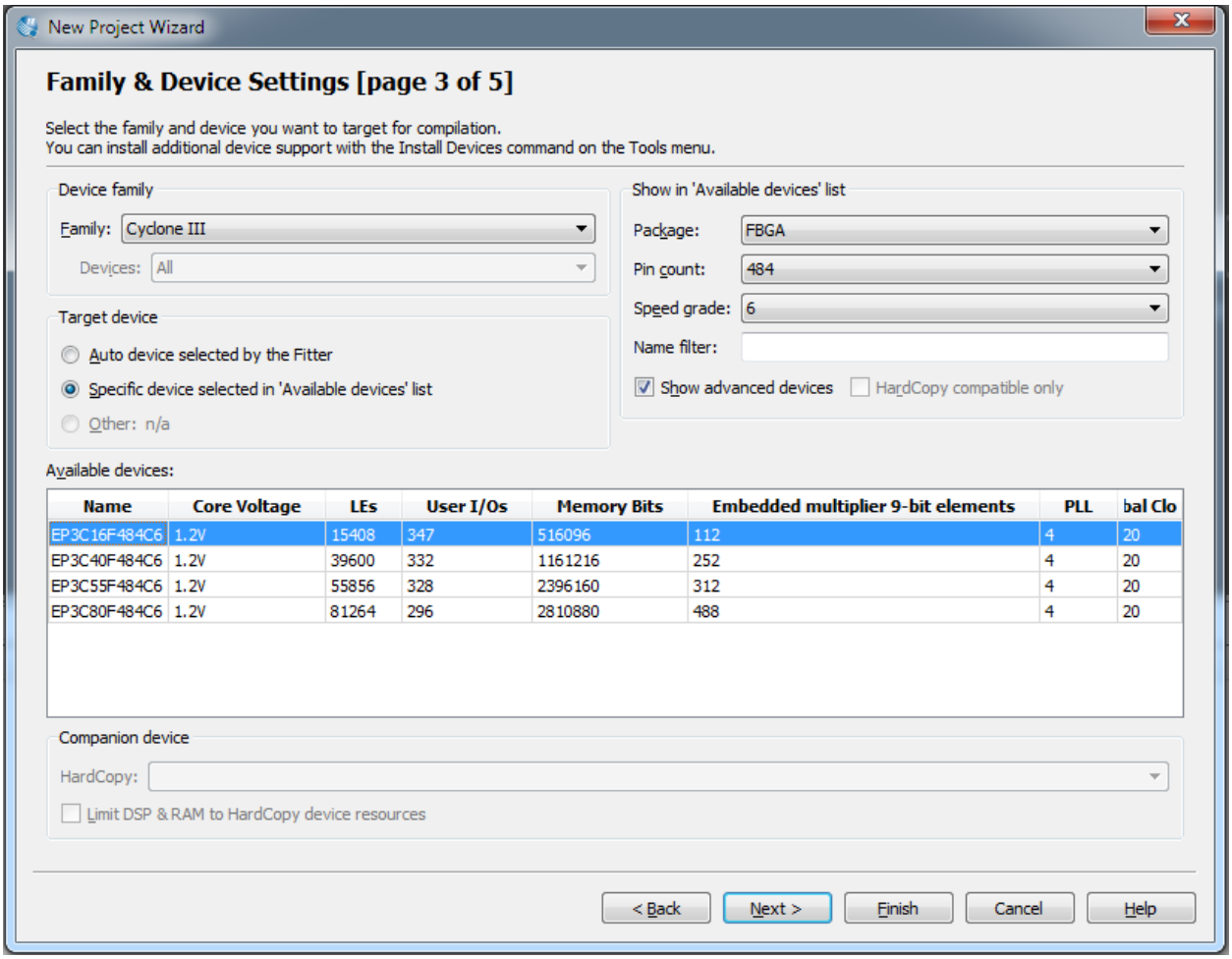


*Figure 4. Selecting the FPGA*

Open ***EDA Tool Settings*** window, and in the ***Simulation*** line ***Tool Name*** has to be ***ModelSim***, and the ***Format(s)*** is ***VHDL*** (Figure 5.).

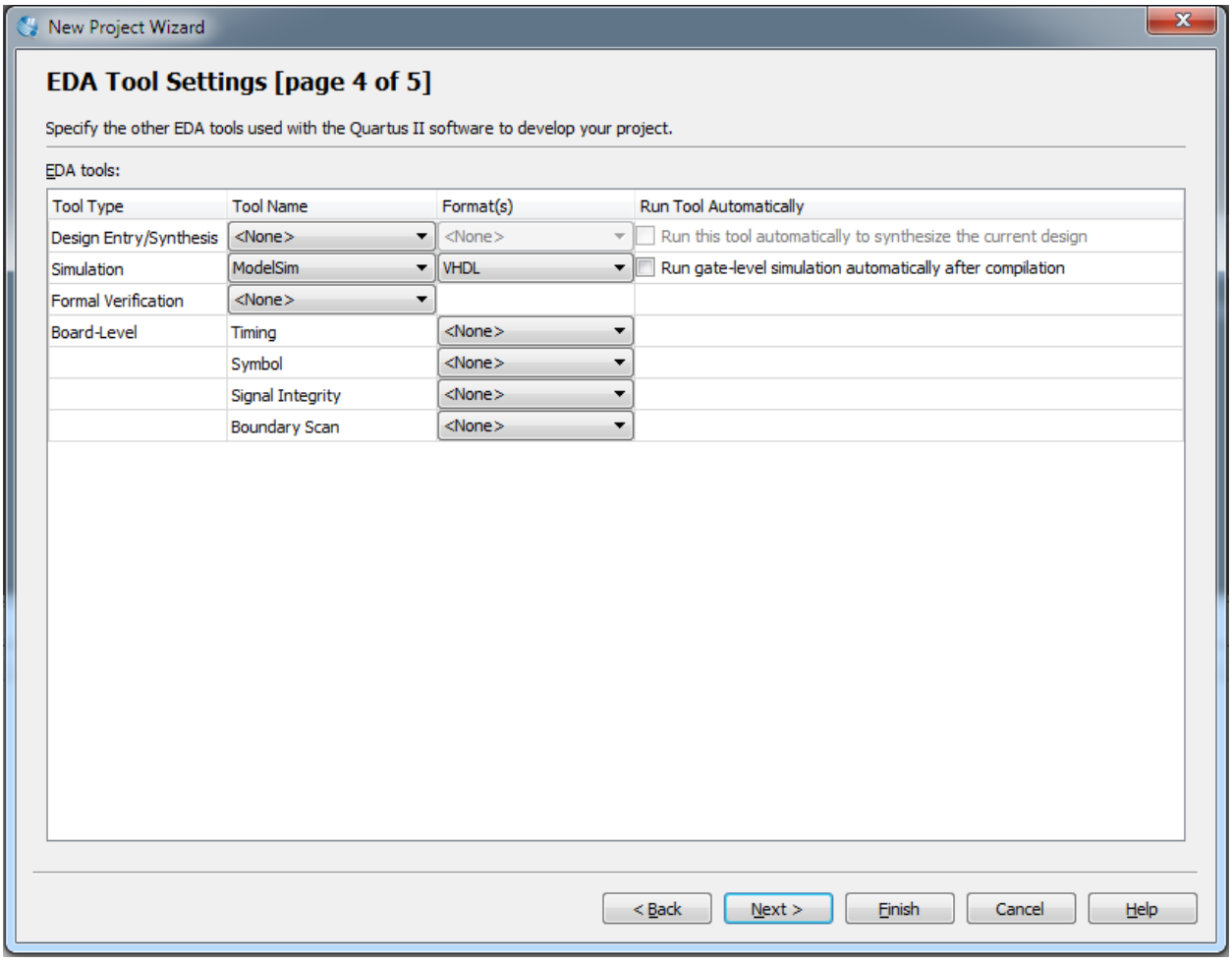


*Figure 5. EDA tool settings*

Press ***Next*** button, and you can accept all the setting pressing ***Finish*** button.

At the top of the ***Project Navigator*** the architecture of the HDL model can be seen. In the ***Flow*** drop-down menu please choose the ***Full Design*** option (Figure 6.)!

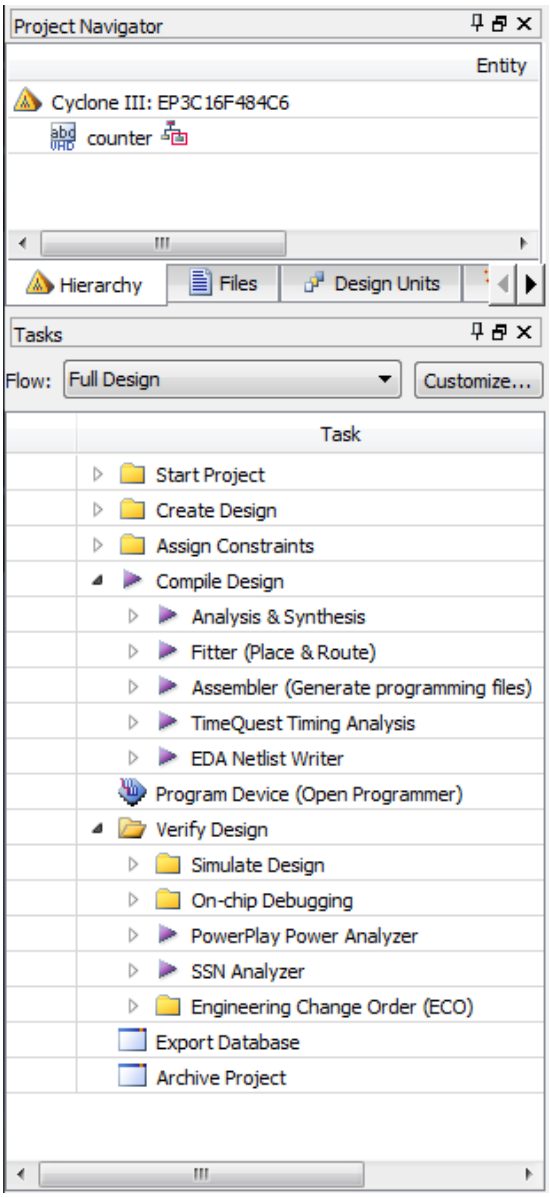


*Figure 6. The project navigator window*

In the ***Project Navigator*** window below the ***Compile Design*** in ***Analysis & Synthesis*** submenu please double click on ***Analysis & Elaboration***.

Now we have to assign the ports to the pins of the FPGA. It can be done using the ***Pin Planner*** in ***Assignments*** menu (Figure 7.).

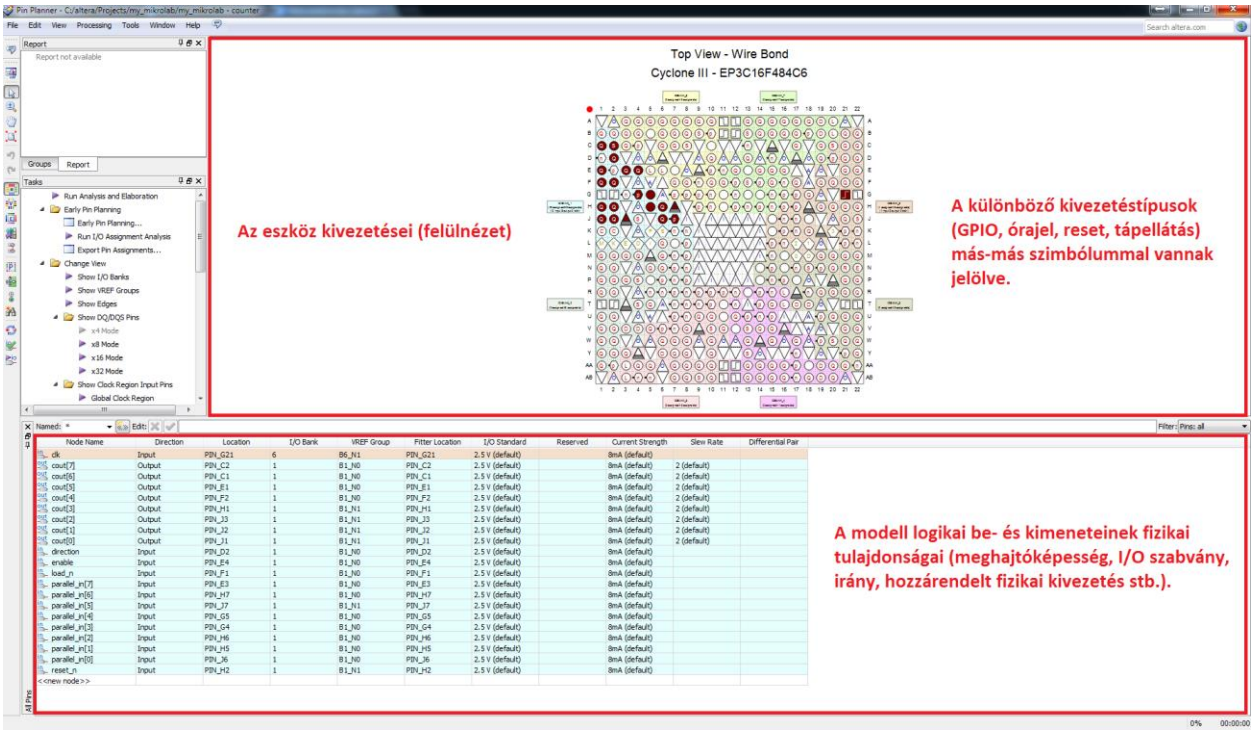


*Figure 7. Pin Planner*

Please fill in the ***Location*** column as Figure 8 depicts.

| Node Name | Direction | Location |
|---|---|---|
| clk | Input | PIN_G21 |
| cout[7] | Output | PIN_C2 |
| cout[6] | Output | PIN_C1 |
| cout[5] | Output | PIN_E1 |
| cout[4] | Output | PIN_F2 |
| cout[3] | Output | PIN_H1 |
| cout[2] | Output | PIN_J3 |
| cout[1] | Output | PIN_J2 |
| cout[0] | Output | PIN_J1 |
| direction | Input | PIN_D2 |
| enable | Input | PIN_E4 |
| load_n | Input | PIN_F1 |
| parallel_in[7] | Input | PIN_E3 |
| parallel_in[6] | Input | PIN_H7 |
| parallel_in[5] | Input | PIN_J7 |
| parallel_in[4] | Input | PIN_G5 |
| parallel_in[3] | Input | PIN_G4 |
| parallel_in[2] | Input | PIN_H6 |
| parallel_in[1] | Input | PIN_H5 |
| parallel_in[0] | Input | PIN_J6 |
| reset_n | Input | PIN_H2 |

*Figure 8. Pin assignment*

The connections are the followings:

- Clock signal (**`clk`**): 50 MHz oscillator
- reset (**`reset_n`**): BTN0 (button)
- Parallel load (**`load_n`**): BTN2 (button)
- Counter enable (**`enable`**): SW8 (switch)
- Direction (**`direction`**): SW9 (switch)
- 8bit parallel in (**`parallel_in`**): SW0-SW7 (switches)
- Value of the counter (**`cout`**): LEDG0-LEDG7 (LEDs)

The ***Pin Planner*** can be closed using ***File*** menu ***Close*** command..

After that, in the *Project Navigator* window please double click on ***Compile Design***. These steps will be performed:

- *Analysis and Synthesis*.
- *Fitter (Place & Route)*.
- *Assembler (Generate programming files)*.
- *TimeQuest Timing Analysis*
- *EDA Netlist Writer*: generating the Post-Place&Route model and the SDF file

The device can be configured using the ***Tools*** menu ***Programmer*** command (Figure 9.).

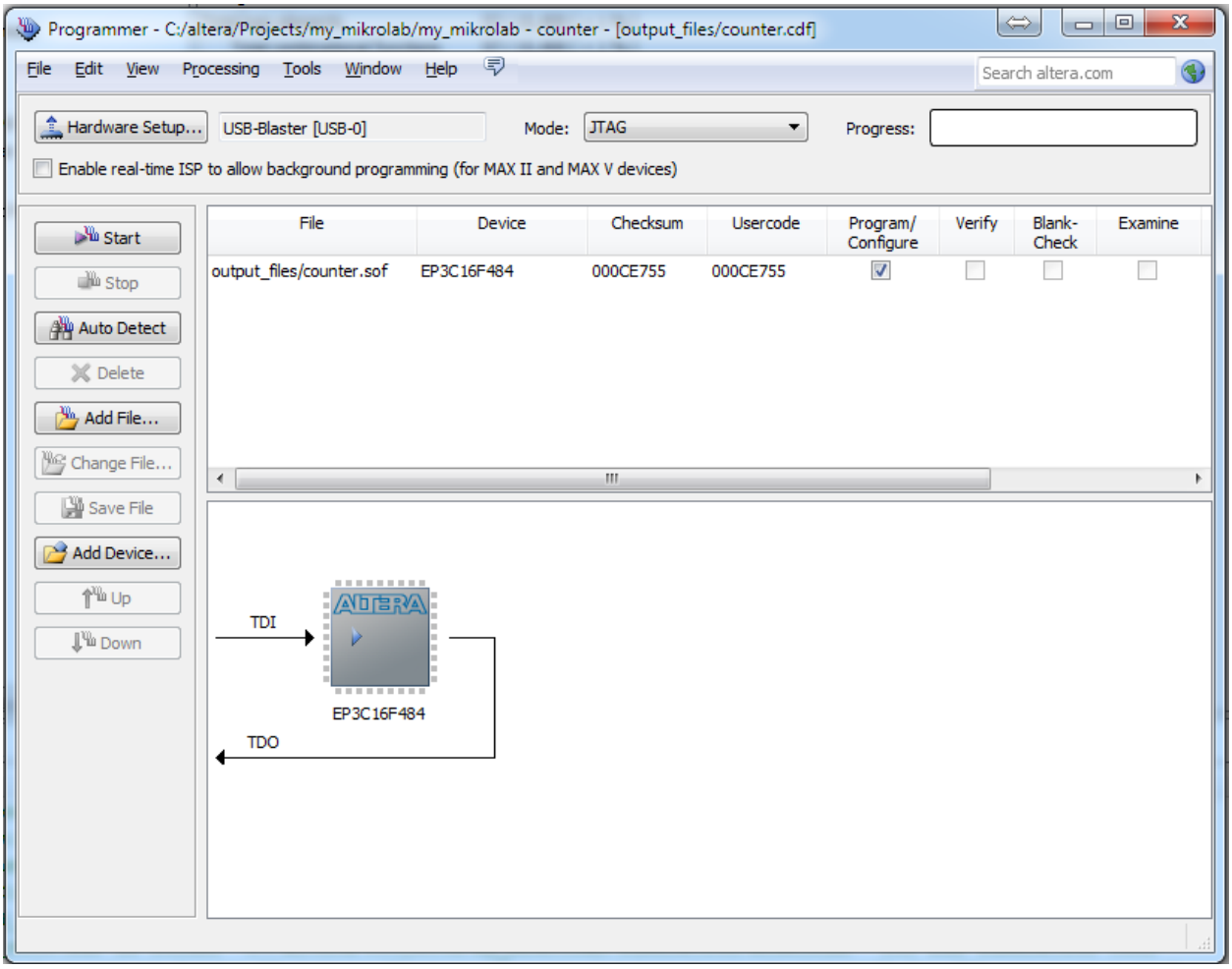


*Figure 9. Programmer*

If you can see ***No Hardware*** message in the ***Hardware Setup*** field (upper left corner), then press ***Hardware Setup*** button and choose the ***USB Blaster*** (Figure 10.).

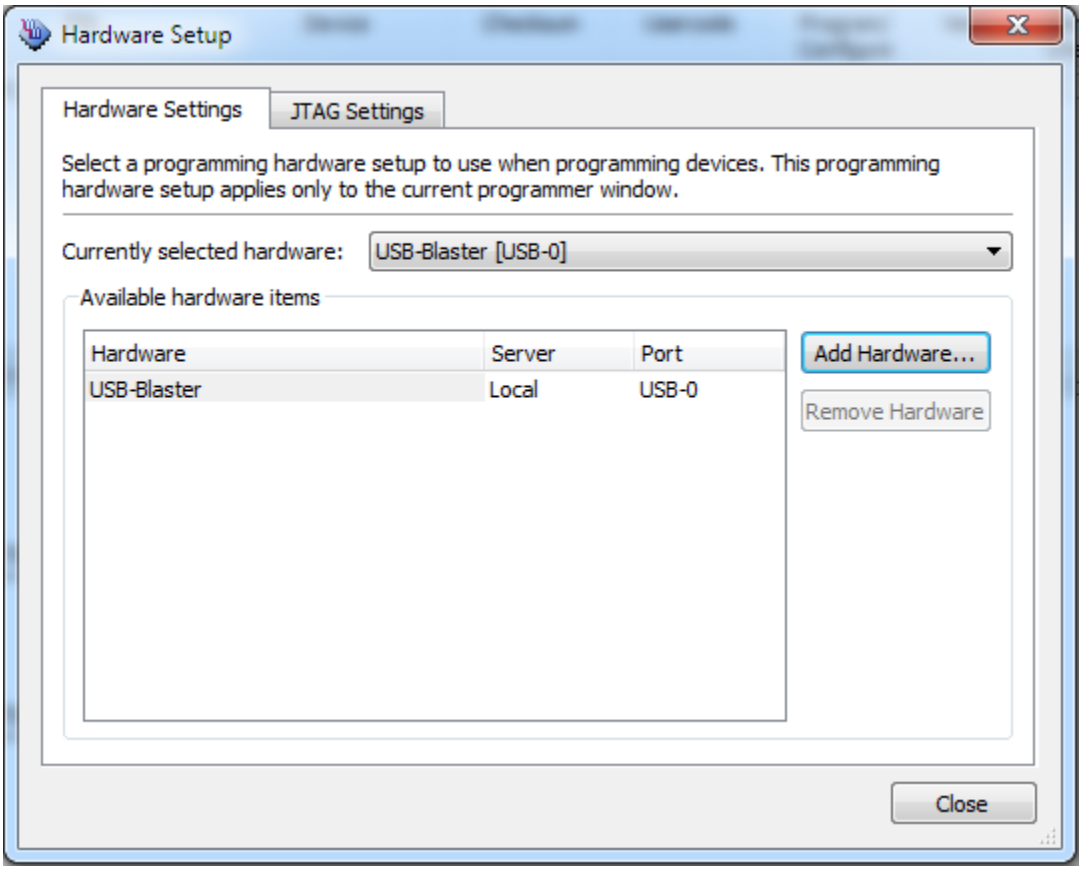


*Figure 10. Hardware setup window*

Thereafter, in the ***Programmer*** window the actual configuration file appears (*output_files/counter.sof)*. If it doesn't, we can do it manually using the ***Add File…*** button. Finally, please press the ***Start*** button on the left side, and the downloading starts. If the downloading process is successful, you can see the ***"100% (Successful)"*** message in the ***Progress*** field.

## Tasks:

Please add one or more extra feature to the counter.

Examples:

- Change the direction of the counter using a switch
- Change the speed of the counter using another switch
- Create a Knight Rider LED light scanner (shift counter, you can use multiplication by 2 or srl/sll functions).
  https://youtu.be/FpyKlLuLbcs

and do what you want to do, except going home before the end of the lab. ☺