Topics

- Introduction to schematic capture and behavioral simulation based on computer aided design tools
- Familiarization with the design tools that will be used in the practical classes.
- Capture of schematic diagrams (logic schemes) and behavioral simulation of elementary components.

Summary

This practical work is divided into 2 parts. In Part I we intend to introduce the basic aspects of the tools that will be used in the practical classes of Introduction to Digital Systems (ISDig), based on a simple project using schematic capture and behavioral simulation. Part II discusses an example of a hierarchical project based on an equality comparator.

Summary of the Design Flow

Figure 1 illustrates the typical design flow of digital systems, based on reprogrammable logic devices, such as Field Programmable Gate Arrays (FPGAs). In ISDig only the initial steps are covered. The rest will be the subject of study in the subsequent course "Digital Systems Laboratory - LSDig".

The design entry stage consists of modeling, coding, or introducing the intended functionality. For such purposes the designer may use hardware description languages, schematic diagrams, state transition diagrams, or other methods. In the case of ISDig, the capture of schematic diagrams and/or state transition diagrams will be used, depending on which one is more appropriate for each system. For convenience, the Integrated Development Environment (IDE) "Quartus Prime" will be used, which has schematic and state diagram editors, as well as other tools needed to perform all steps of the project flow.

Once the system has been modeled, the next step is its synthesis (logic), i.e. the compilation of the model to create a netlist (i.e. a set of logic gates, flip-flops, multiplexers, other components and their interconnections) that implements the desired functionality. This step is performed by software tools commonly developed by the manufacturer of the implementation technology used.

System validation can be performed at various stages of the project flow, either by simulation or real hardware verification. Various tools for time analysis, energy analysis, logic resources usage, etc. are also normally available from the manufacturer. In this practical work only behavioral simulation will be performed.

Important Notes: Due to network speed limitations in accessing the personal directory on arca.ua.pt, on classroom PCs it is recommended to use a USB stick to store your projects and files. If this is not possible, save and access your work from a directory of drive Z :.

In either case do not use spaces or special characters (e.g. accents) in the project and file paths as this will cause problems with the use of the tools (this means you should not save your projects into sub directories of "Desktop" or "My Documents").

Tip: In Windows create, for example in C:\Users\<User>, a directory structure to save your ISDig projects (e.g. C:\Users\<User>\ISDig\P5\Part1\<Project Name>); on Linux you can do so at /home/<user>/ISDig/P5/Part1/<Project Name>).

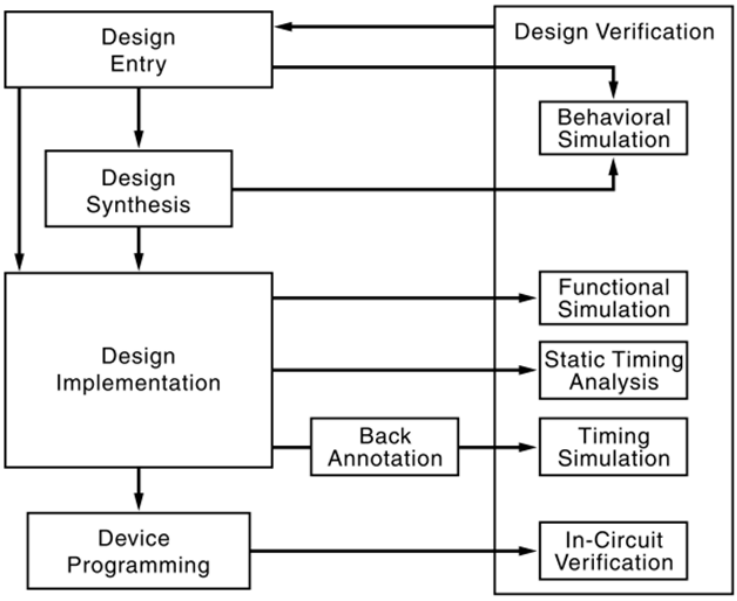


Figure 1 - Design flow for FPGA-based digital systems (source: www.xilinx.com).

Part I

Demonstration of key steps in project flow based on schematic capture and behavioural simulation

1. Open the "Quartus Prime" application and create a new project (menu "File→New Project Wizard") according to the following steps (Figures 2 to 9).

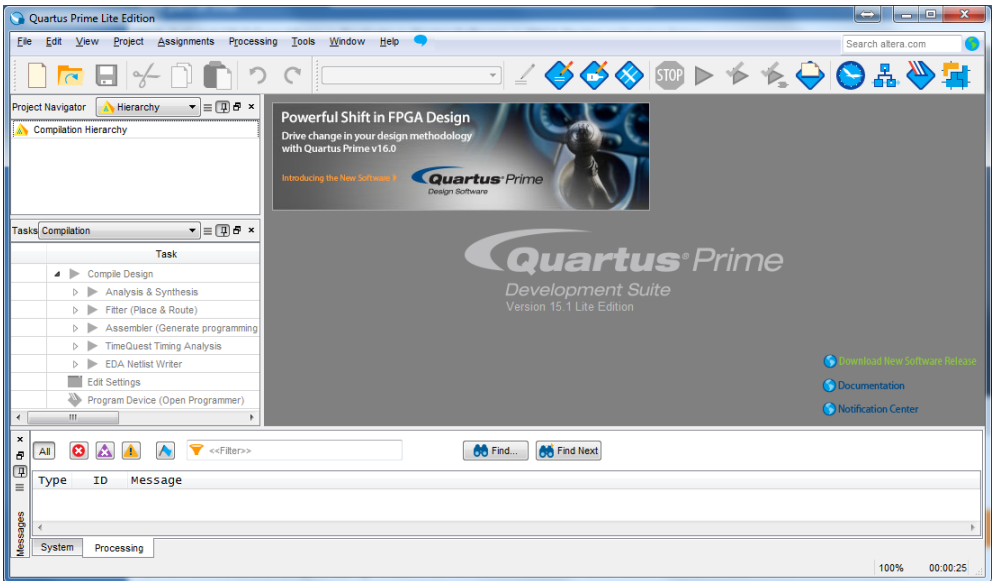


Figure 2 - Initial aspect of the "Quartus Prime" application (without any project opened).

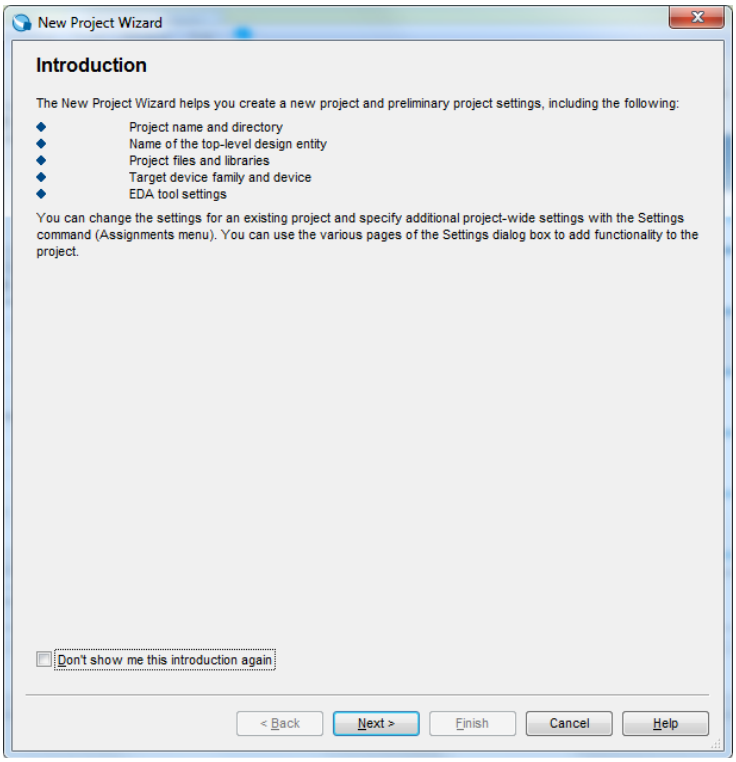


Figure 3 - Introductory initial step (can be disabled).

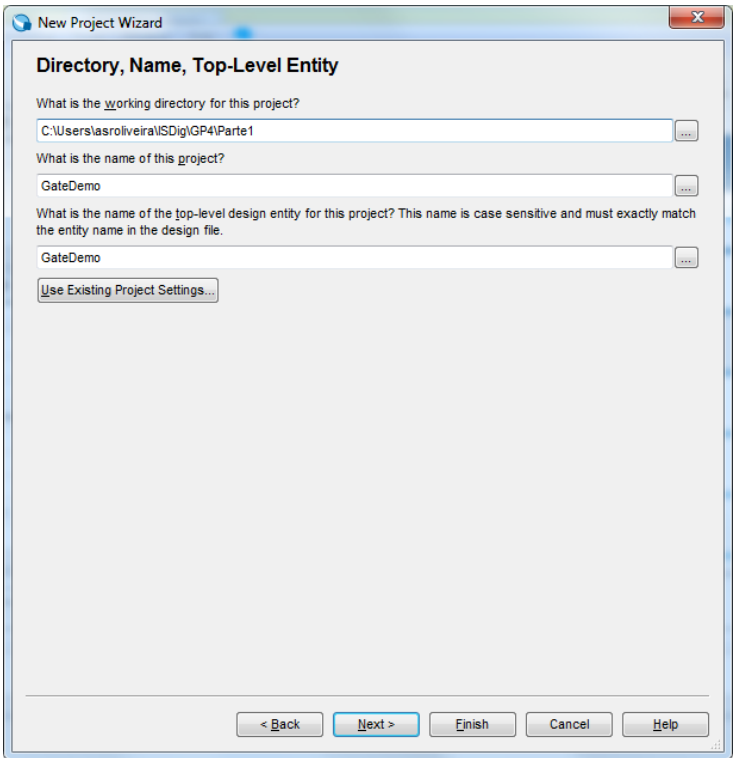


Figure 4 - Step 1 - Identification and location of the project in the file system - adapt according to the directory used, which cannot contain spaces or special characters, e.g. accents.

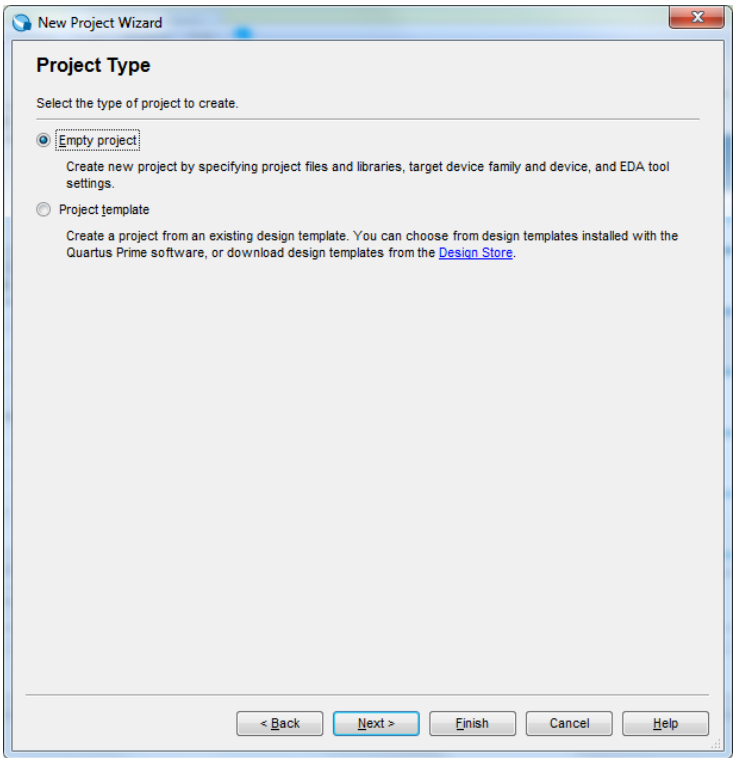


Figure 5 - Step 2 - Select the type of project to create (empty project in this case).

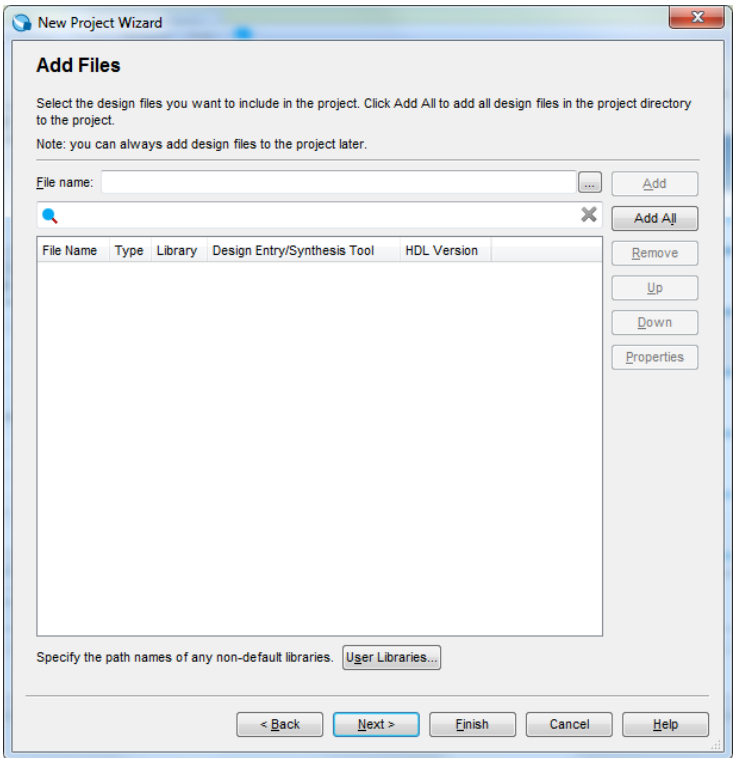


Figure 6 - Step 2 - Adding pre-existing files (not used in this project).

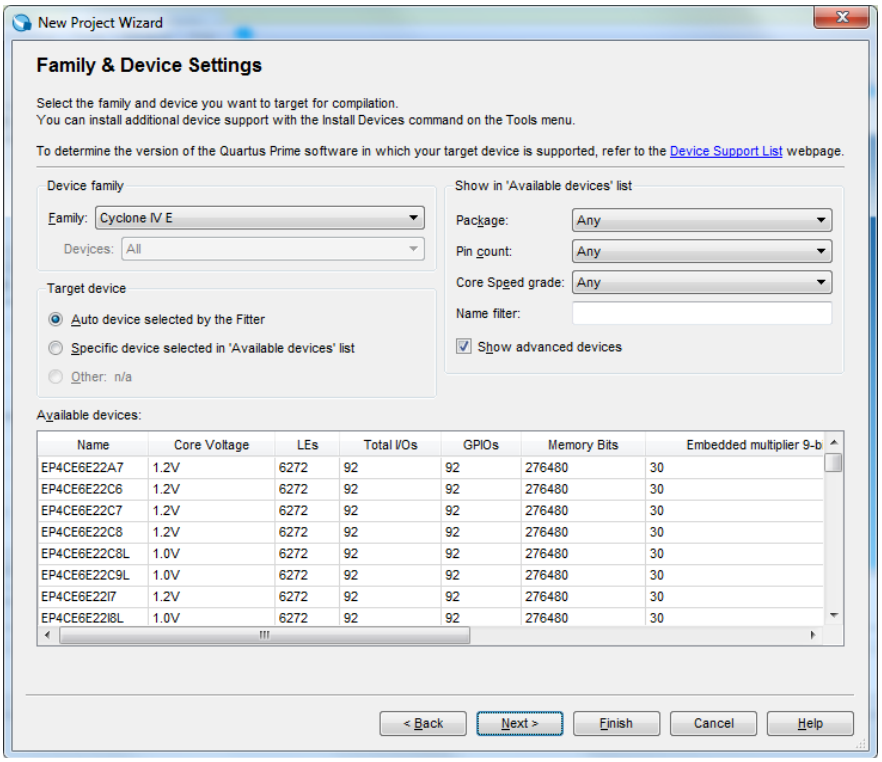


Figure 7 - Step 3 - Selection of the device (FPGA) used in the implementation (as the only behavioural simulation is intended, the selection of a concrete device can be omitted).

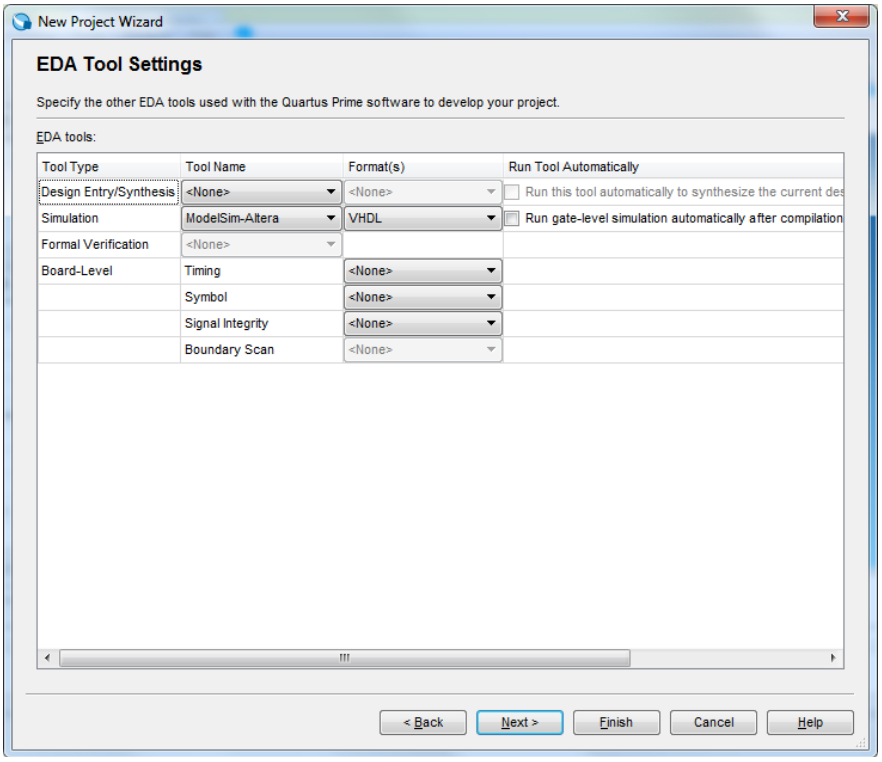


Figure 8 - Step 4 - Selection of the tools and languages used in the project flow (use default values).

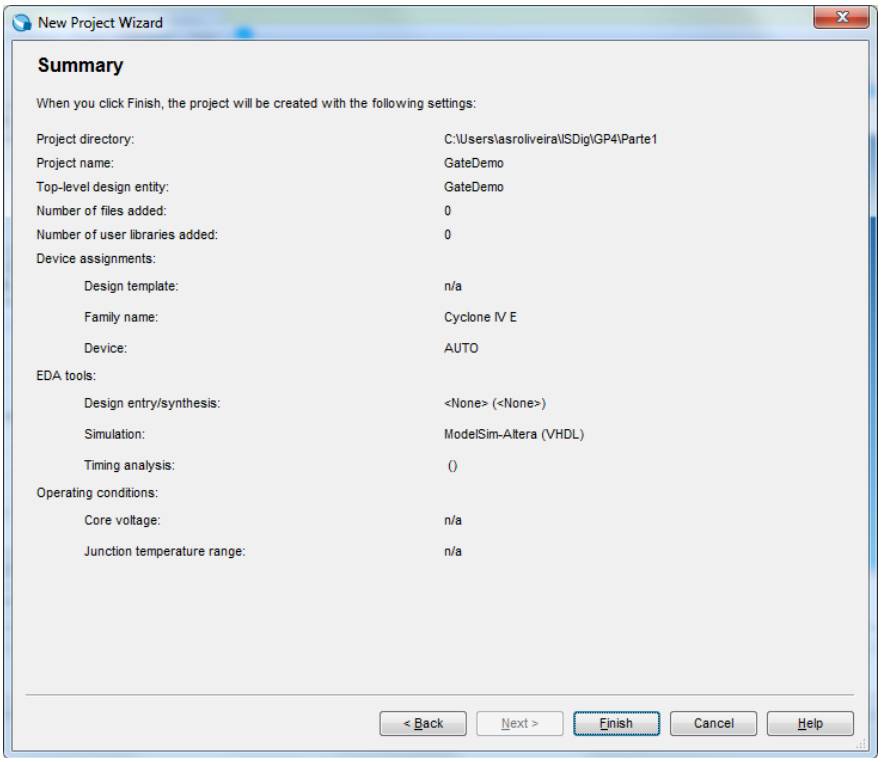


Figure 9 - Step 5 - Final summary of project creation.

2. After pressing "Finish" the "Quartus Prime" IDE should look like Figure 10.

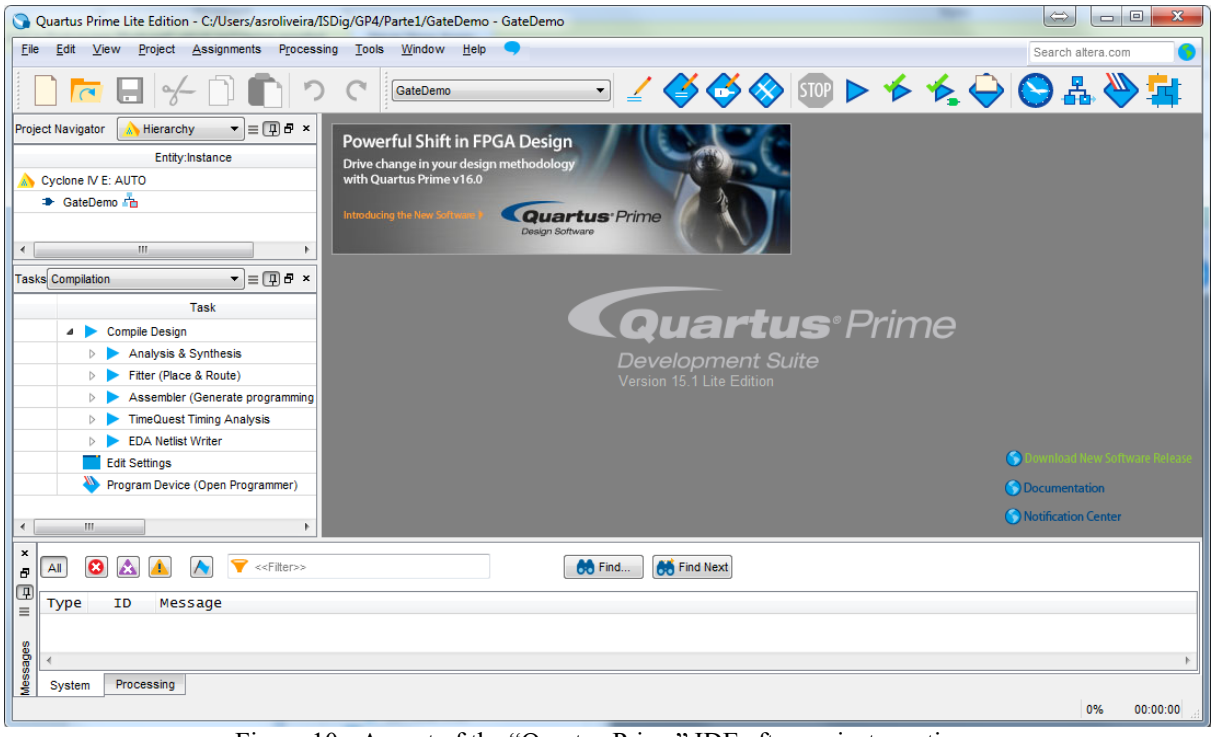


Figure 10 - Aspect of the "Quartus Prime" IDE after project creation.

3. Create a new file for a schematic diagram ("File→New" menu) according to Figure 11.

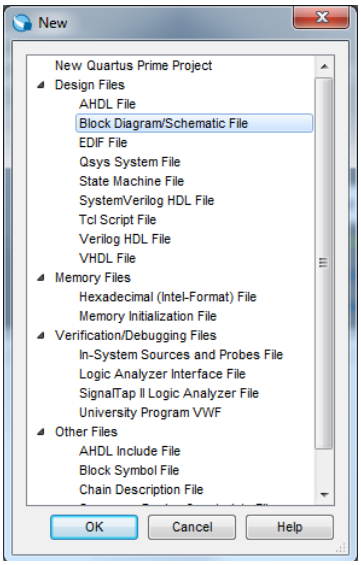
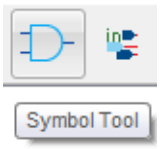


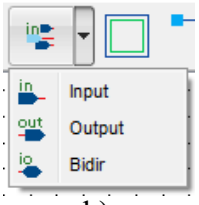
Figure 11 - Selection of the file type to create (Block Diagram / Schematic File).

4. Add a 2-input AND logic gate using the "Symbol Tool" toolbar button (Figure 12 a)) and choosing the component according to Figure 12 c).

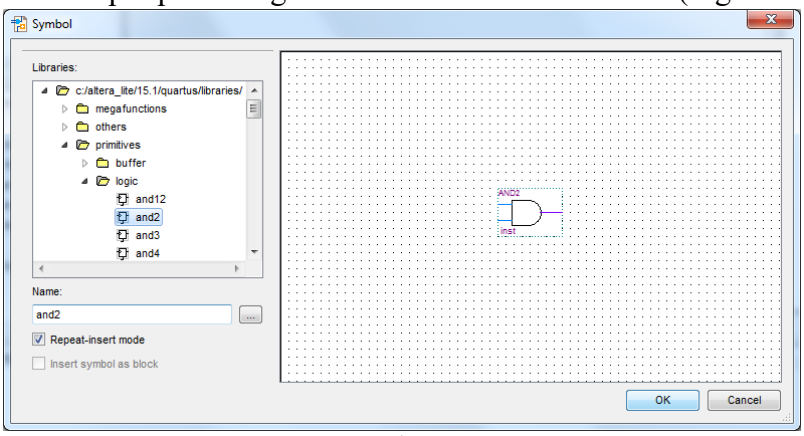
5. Add two input ports and one output port using the "Pin Tool" toolbar button (Figure 12 b)).



a)



b)



c)

Figure 12 - "Tool Tool" and "Pin Tool" toolbar buttons and AND logic port selection in the Quartus Prime component library.

6. Link the input, output, and logical ports according to Figure 13. Identify each of the elements according to the names shown in Figure 13.

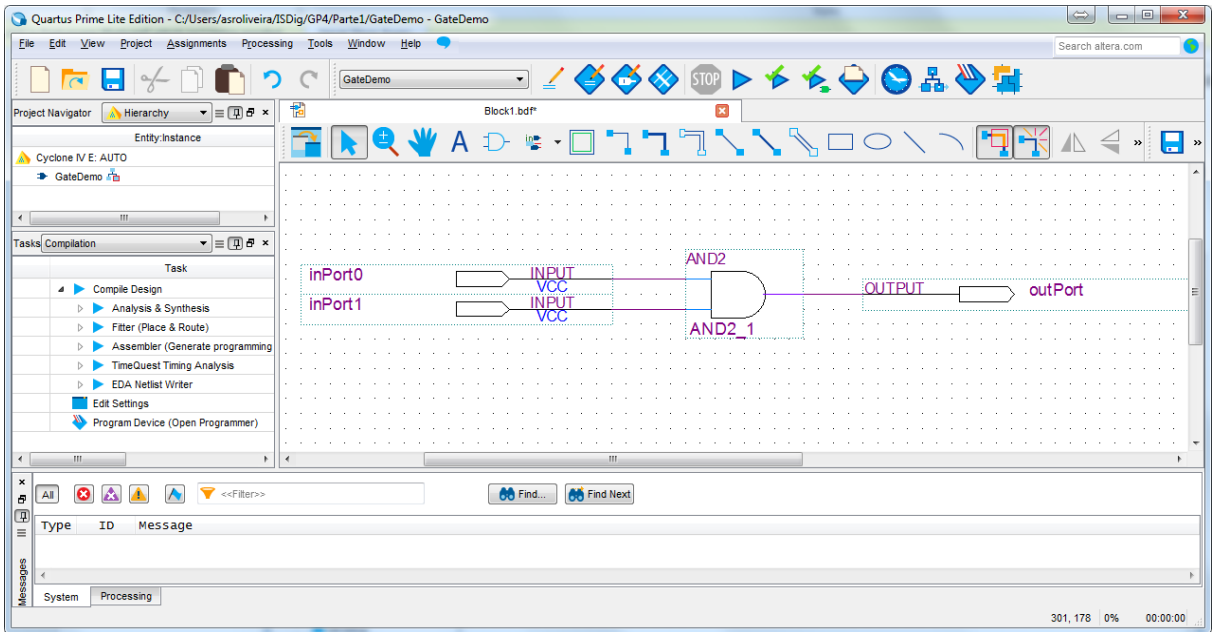


Figure 13 - Interconnection of AND logic port and input and output ports and identification of the various circuit elements (to rename a component or port, double click on the current name).

7. Save the file, whose name should be "GateDemo.bdf" (Figure 14).

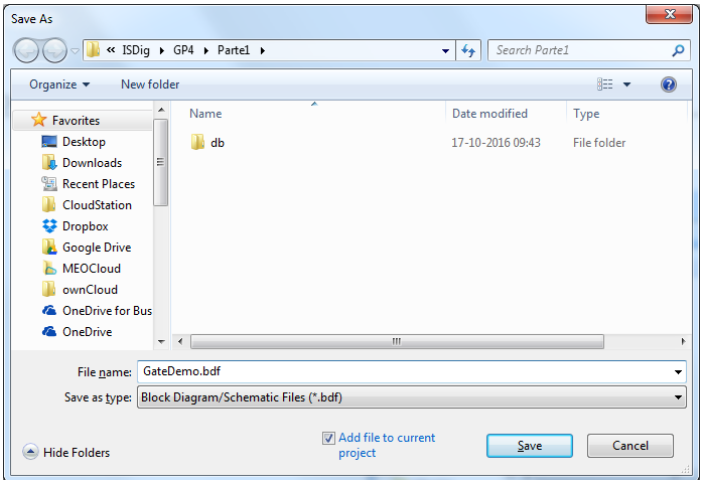


Figure 14 - Dialog for saving the GateDemo.bdf file.

8. Now the behaviour of the logic gate used will be validated by simulation. However, before performing the simulation, execute the "Analysis & Synthesis" option to analyse the structural correction of the project. After running Analysis & Synthesis the IDE should look like Figure 15.

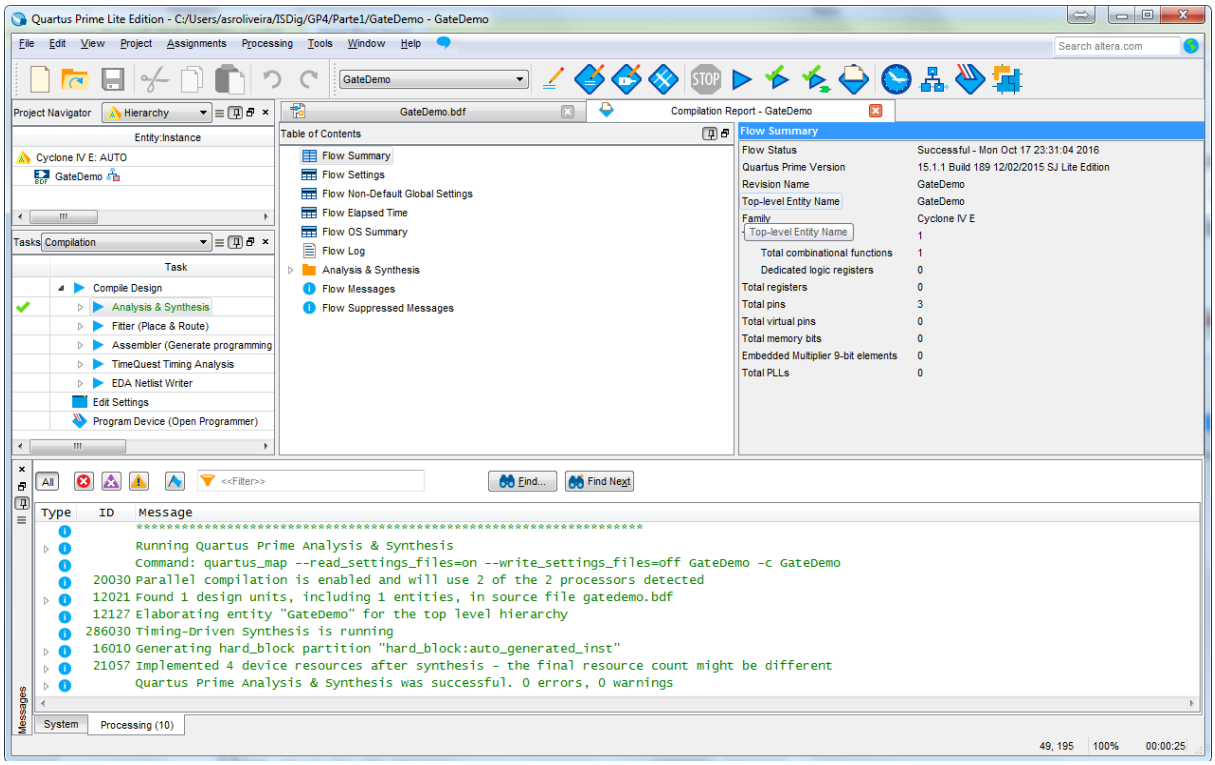


Figure 15 - "Quartus Prime" IDE after running "Analysis & Synthesis".

9. Simulate the behaviour of the logical port by creating a VWF file ("File→New" menu) according to the steps described in the following points (described in Figures 16 to 18).

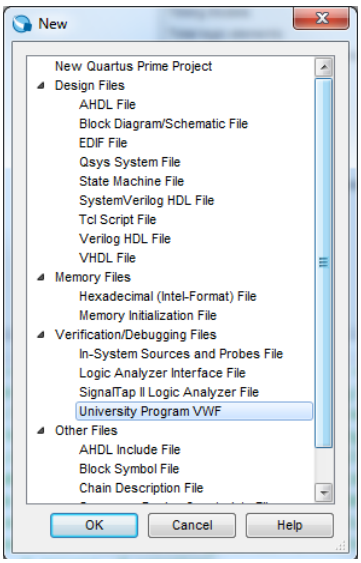


Figure 16 - Selection of file type to create ("University Program VWF").

10. After pressing "OK" the next window should open, where the signals to be used in the simulation should be indicated (Figure 17).

11. Through the "Edit→Insert→Insert Node or Bus" menu, then pressing the "Node Finder" and "List" buttons, select all input and output ports of the circuit (Figures 17 and 18).

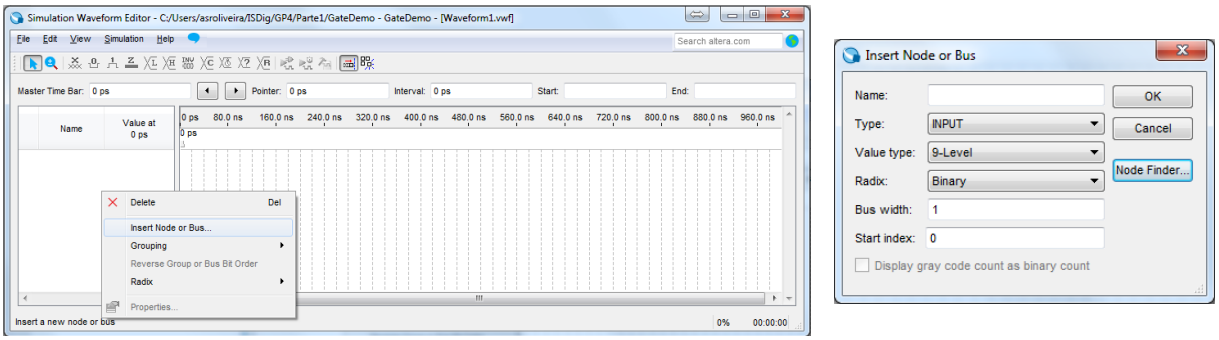


Figure 17 - Simulator window before specifying the input and output signals to be used in the simulation.

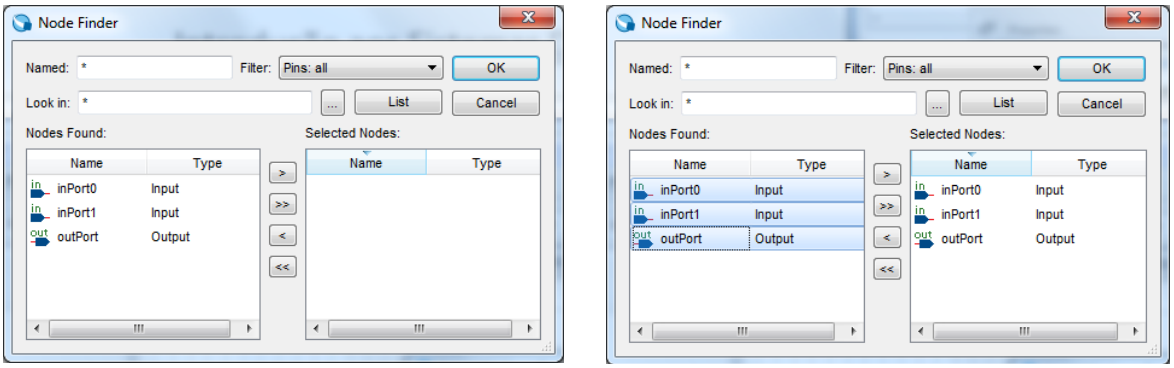


Figure 18 - Specification of input and output signals to be used in the simulation.

12. After pressing "OK" you can specify the desired values for the circuit inputs over time - the output value (s) will be determined during the simulation. Use the mouse to select the time-frame sections of the desired signal with the logical value you want it to assume (Figure 19).

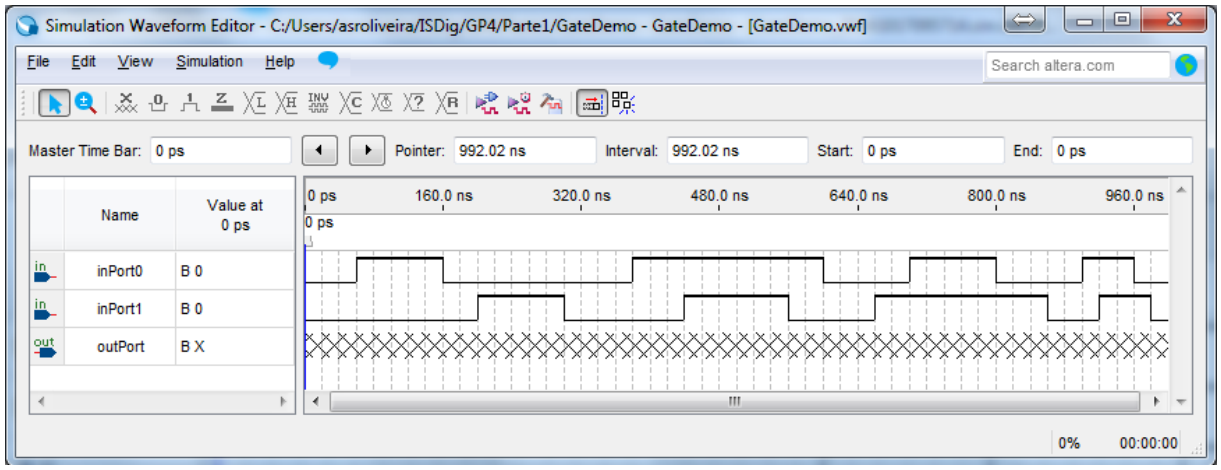


Figure 19 - Simulator window after specifying the input signal waveforms (vectors).

13. After specifying the simulation values (vectors), save the file with the name "GateDemo.vwf" (Figure 20).

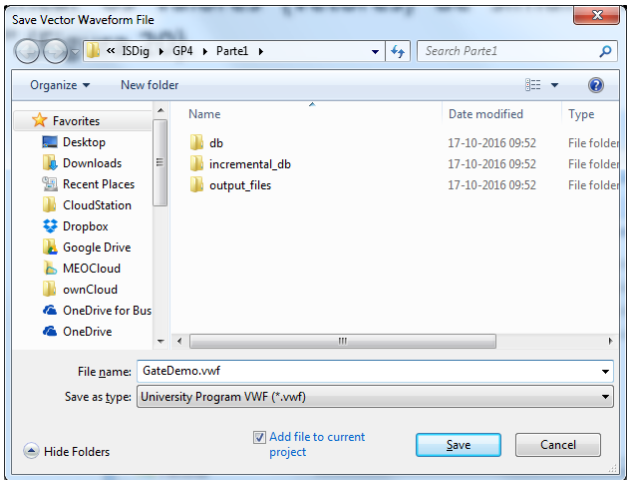


Figure 20 - Dialog for saving the file "GateDemo.vwf".

14. Run the simulation through the "Simulation→Run Functional Simulation" menu, which should open a window similar to Figure 21. After the simulation you should get the value of the logic gate output corresponding to the inputs you specified (Figure 22).

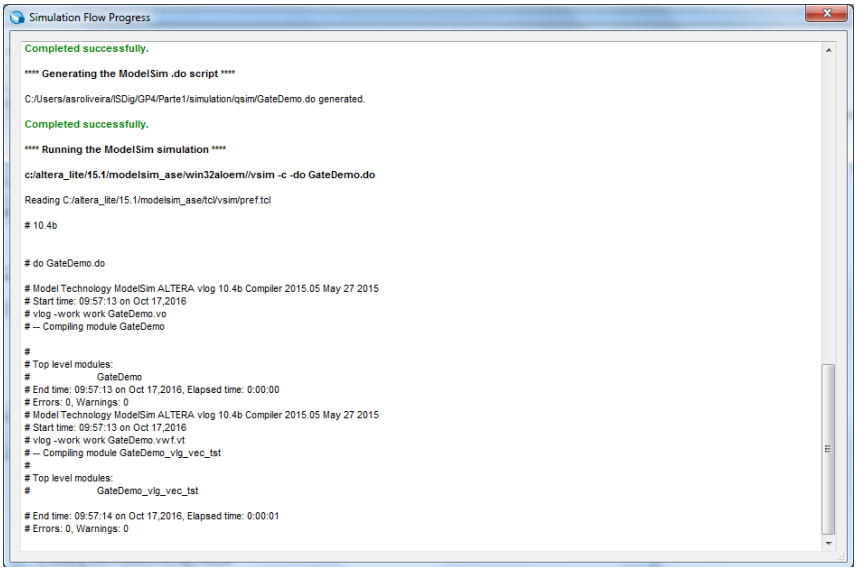


Figure 21 - Circuit simulation compilation and execution window with the specified input vectors.

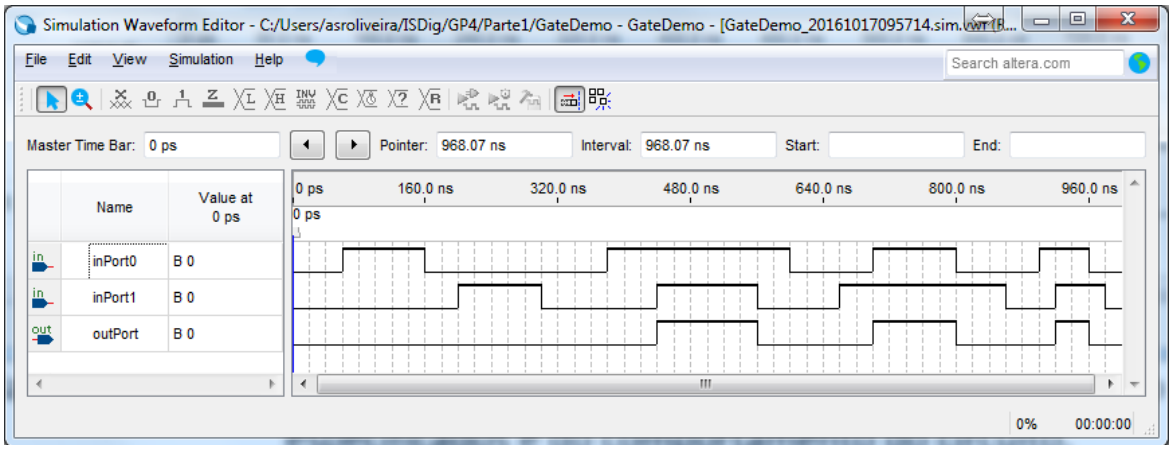


Figure 22 - Simulator window with output signal waveform as a function of the specified input vectors and circuit behaviour.

15. Add to the "GateDemo.bdf" file the instantiation of an inverter to construct a NAND logical port (instantiating a component from a library means using it in the context of a project). Perform the behavioral simulation of the set.

Part II

Hierarchical Design Demonstration

1. Create a new project by replicating the steps in part 1 of this guide. Designate the new project and its top-level entity as "EqCmpDemo".

2. Create a new file for a schematic diagram called "EqCmp4.bdf" to implement a 4-bit word equality comparator (Figure 23). At the end of editing the logic scheme, the circuit should look similar to Figure 24. Use the "xnor" and "and4" components of the Quartus Prime library accessible through the Symbol Tool.

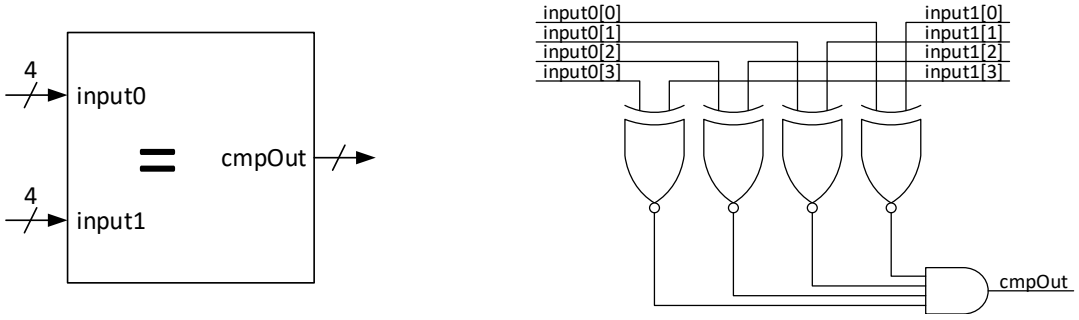


Figure 23 - Logical diagram of the 4-bit word equality comparator.

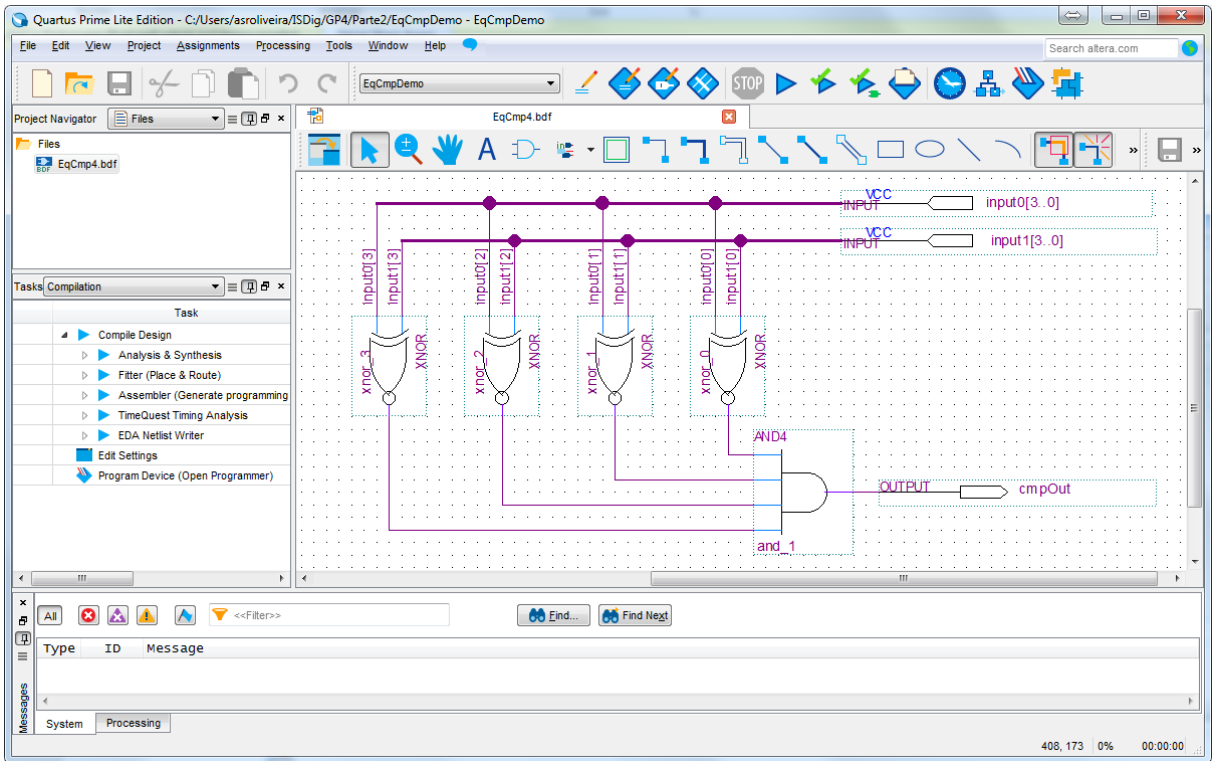


Figure 24 - Logical diagram of the 4-bit word equality comparator with interconnection and identification of the various circuit elements.

3. Create a symbol for the "EqCmp4" module so that it can be used in a schematic diagram as shown in Figure 25 and write it with the name "EqCmp4.bsf" (Figure 26).

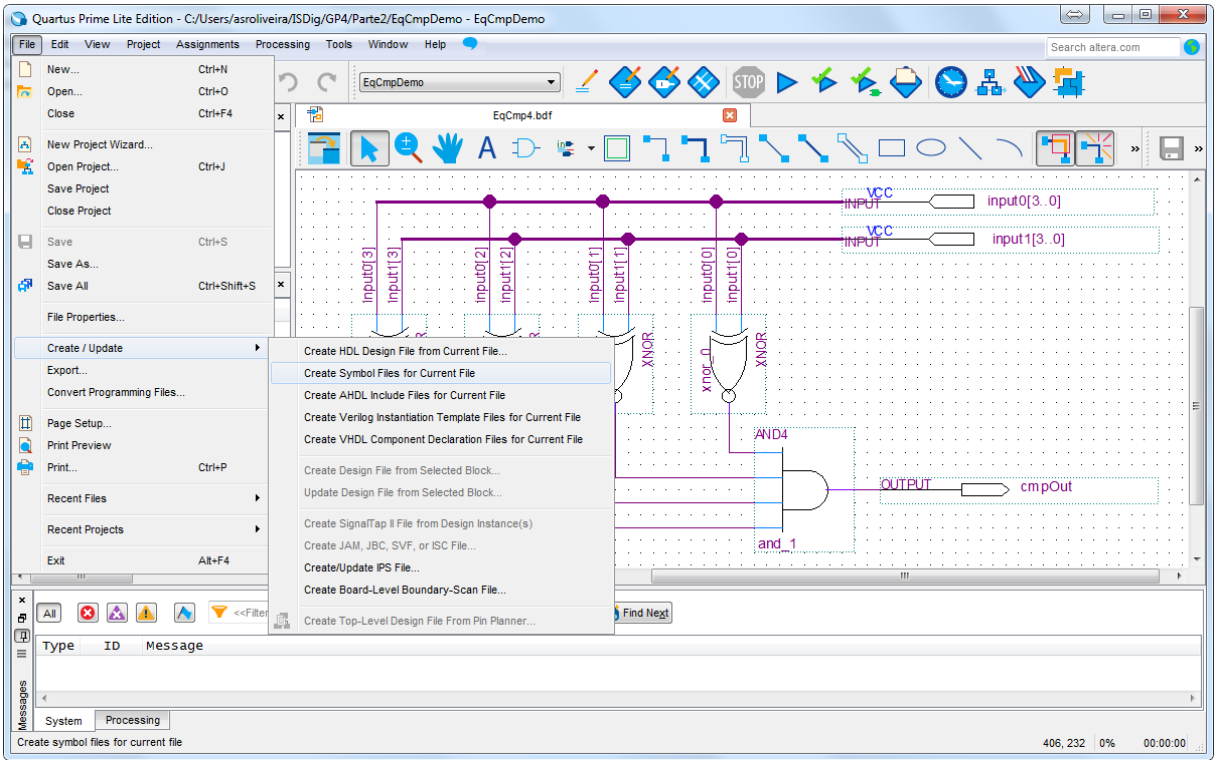


Figure 25 - Creation of a symbol for the "EqCmp4" module.

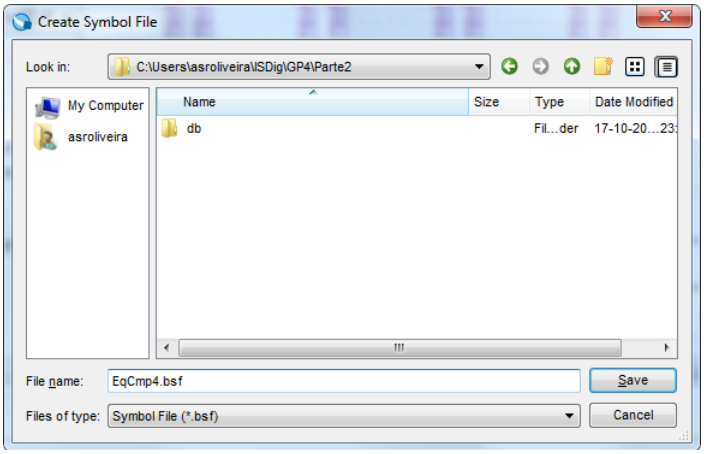


Figure 26 - Recording of file "EqCmp4.bsf" relative to module symbol "EqCmp4".

4. Create a new file for a schematic diagram called "EqCmpDemo.bdf" to instantiate the equality comparator built in the previous point (Figure 27) and link its ports as illustrated in Figure 28.

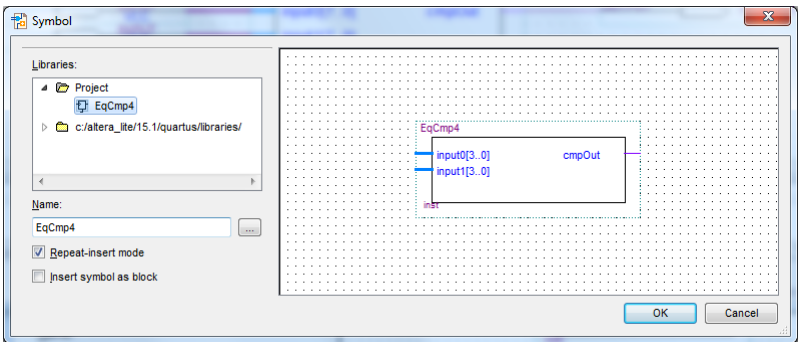


Figure 27 - Instantiation in a schematic diagram of the module “EqCmp4”.

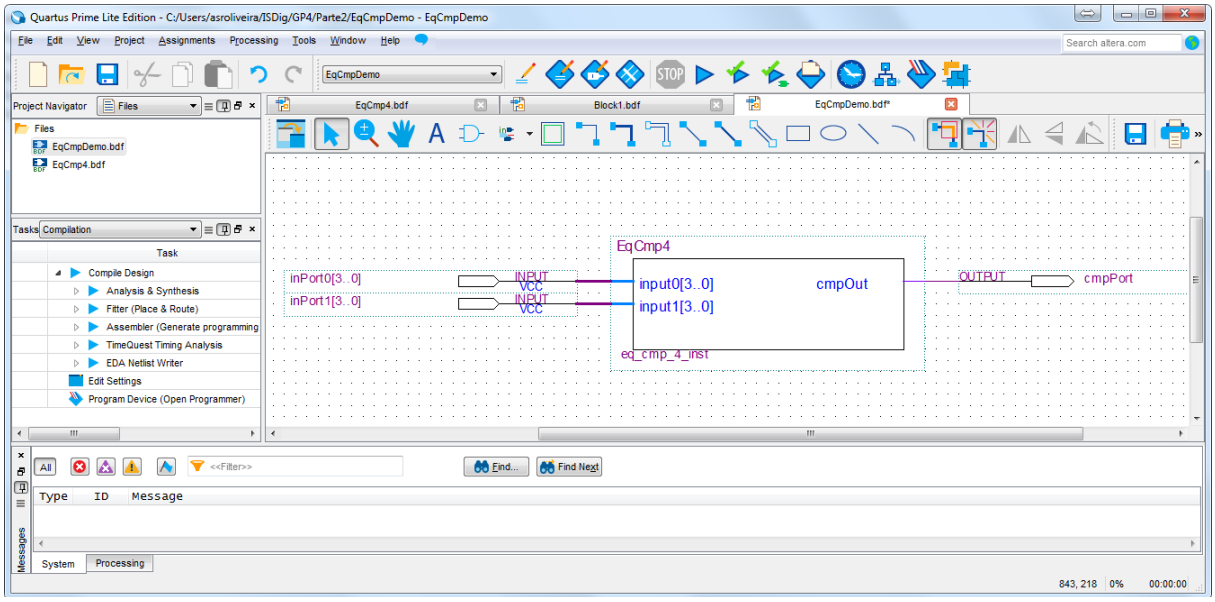


Figure 28 - Interconnection of the “EqCmp4” module and the input and output ports and identification of the various circuit elements in the “EqCmpDemo” module.

5. Perform the behavioural simulation of the comparator.

6. Build and simulate a 16-bit comparator from the instantiation of 4-bit comparators and in as few elementary logic gates as possible.