Laboratorul 1 – Implementarea unei functii logice intr-un circuit programabil

Obiective

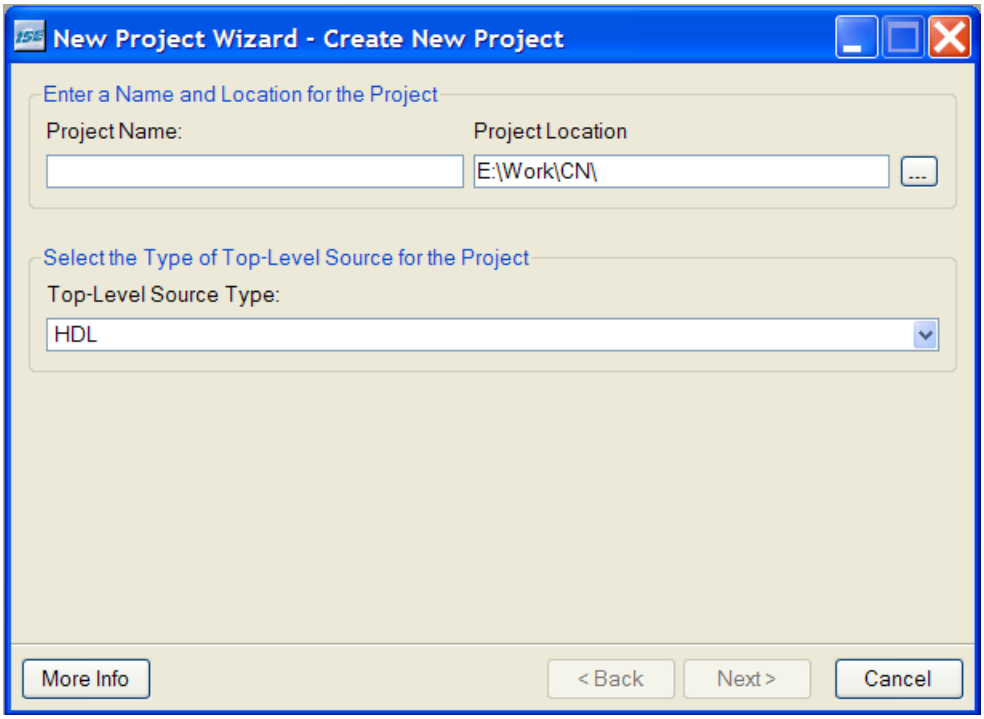
Acest laborator este o introducere in proiectarea logica utilizand Verilog cu ajutorul mediului Xilinx ISE 9.2i. In cadrul laboratorului nu vor fi prezentate noi concepte de proiectare logica. Scopul acestuia este de a va familiariza cu uneltele de proiectare pe care le veti utiliza in acest semestru.

- Xilinx ISE
- Digilent D2SB
- Mentor Graphics ModelSim

Pentru a obtine punctajul acordat acestui laborator, trebuie sa prezentati laborantului functionarea corecta in hardware a proiectului dumneavoastra.

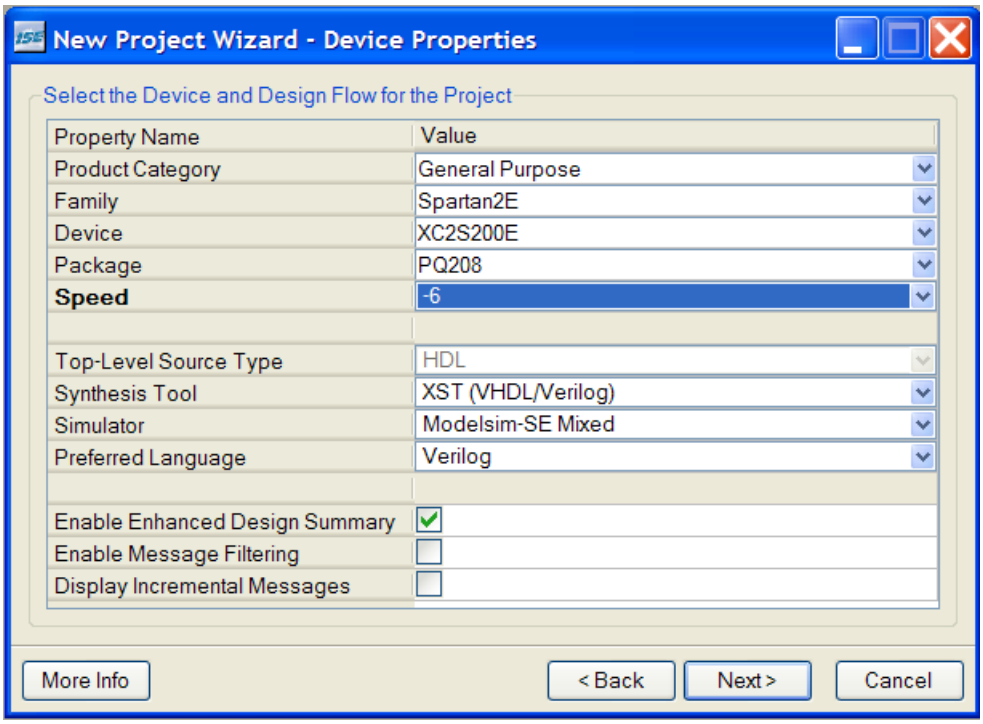
Proiectarea

Proiectul din acest laborator isi propune sa implementeze o functie logica cu doua intrari si o iesire (de exemplu un XOR). Dupa pornirea aplicatiei ISE, selectati meniul File -> New Project. Prima fereastra de dialog New Project va aparea ca in figura de mai jos:

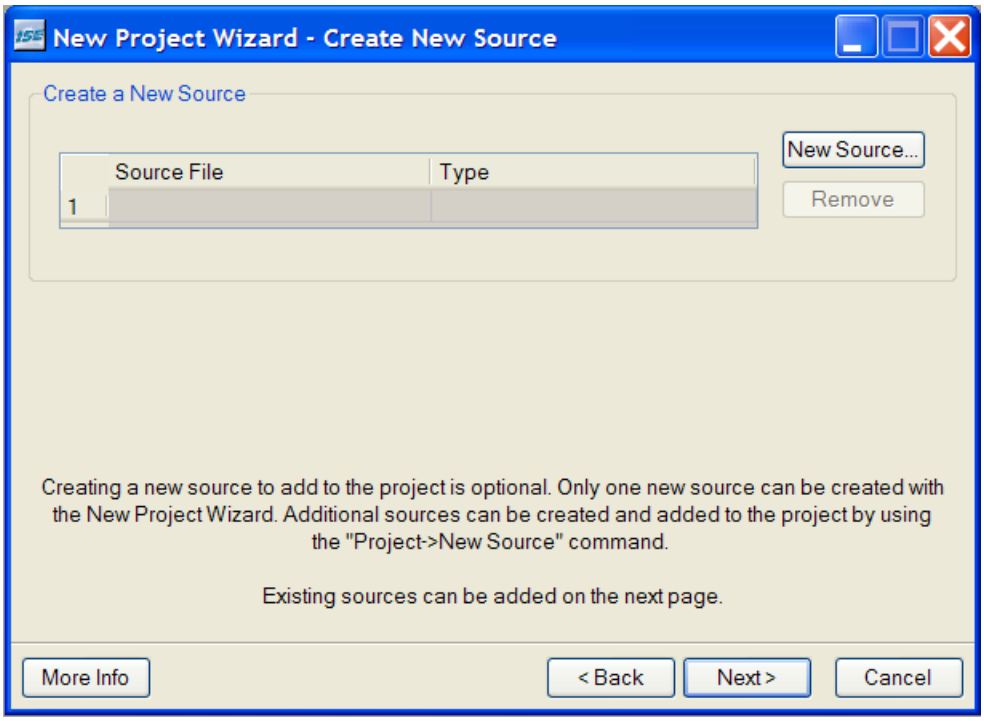


Sunteti solicitati sa introduceti numele proiectului dumneavoastra (lab1), locatia unde acesta va fi salvat si tipul modulului de top. Sunteti sfatuiti sa va salvati proiectele proprii (nu va bazati pe

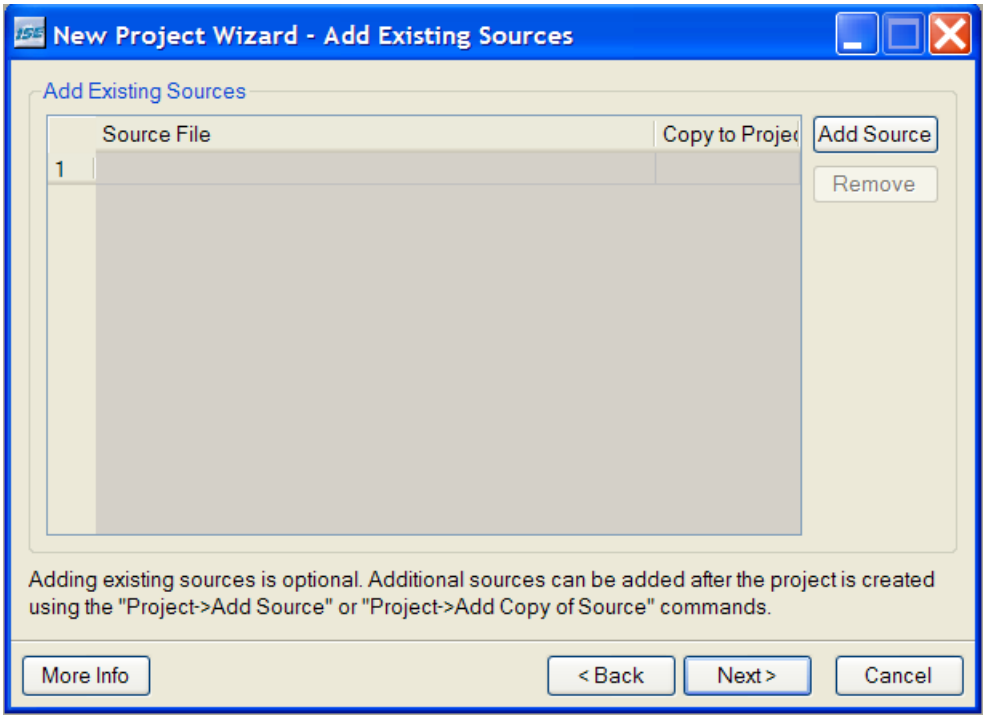
copia de pe calculatorul din laborator deoarece aceasta poate fi stearsa). Dupa introducerea datelor dorite selectati Next.
In urmatoarea fereastra de dialog sunt solicitate datele privitoare la resursele ce sunt la dispozitie pe placa de dezvoltare D2SB si limbajul de descriere, unealta pentru sinteza si cea pentru simulare (ca in figura de mai jos). Dupa selectare obtiunilor dorite selectati Next.



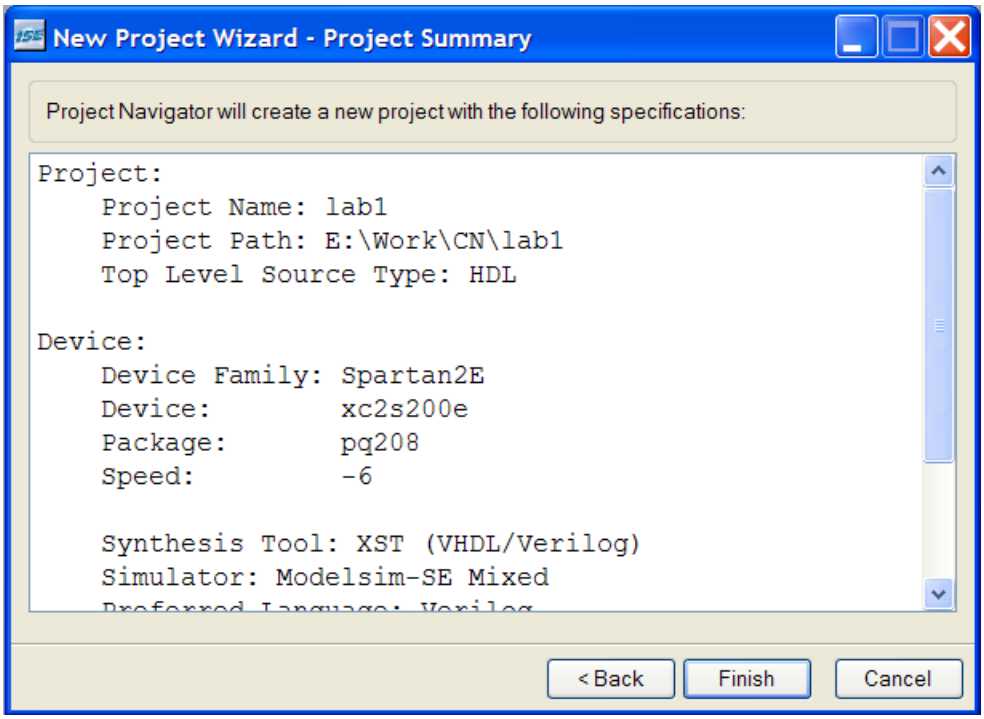
Urmatoarea fereastra de dialog va permite adaugarea unor noi surse in proiectul dumneavoastra. Nu adaugati nici o sursa si selectati Next.



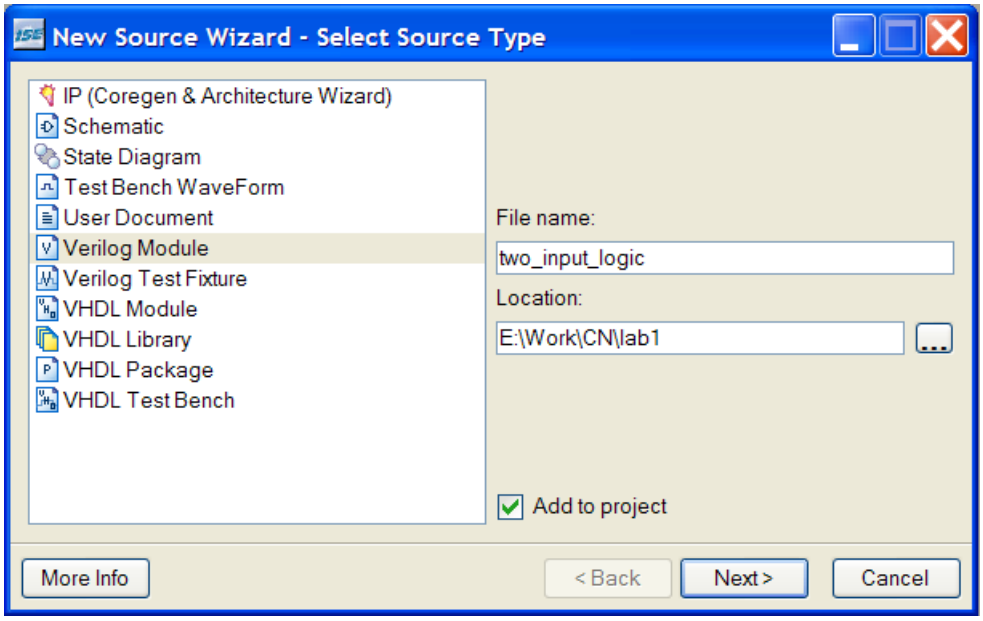
In urmatoarea fereastra de dialog aveti posibilitatea de a adauga surse existente in proiectul dumneavoastra (aceasta facilitate este utila pentru reutilizarea codului). Nu adaugati nici o sursa si selectati Next.



Ultima fereastra de dialog New Project va prezinta sumarul celor selectate anterior. Daca datele prezentate nu corespund dorinteleo dumneavoastra selectati Back si corectati erorile. Daca sunteti multumiti de rezultat selectati Finish ca in figura urmaotare.

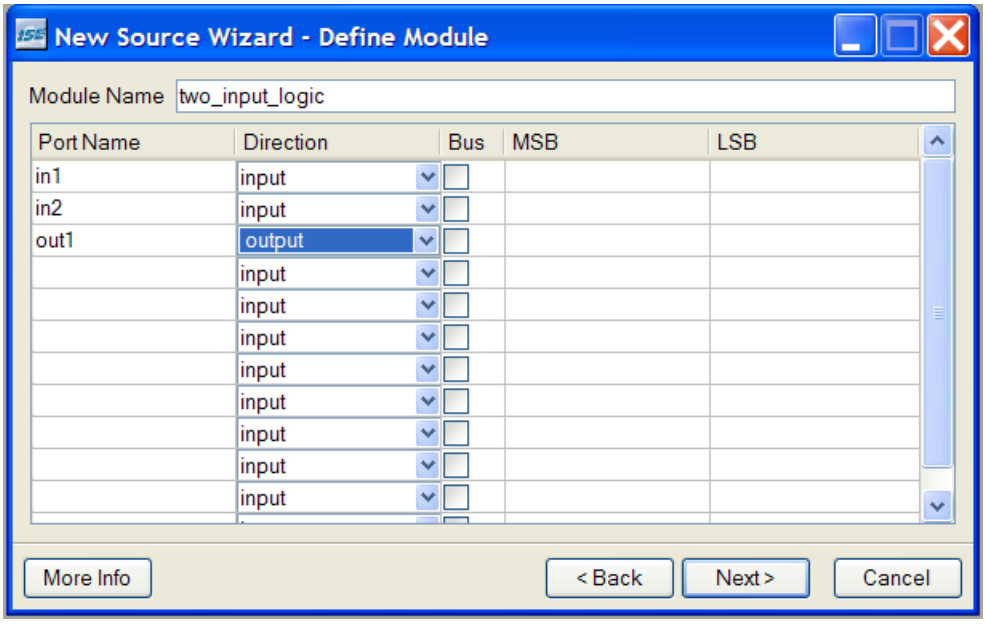


In acest moment ati creat un nou proiect ce nu contine nici o sursa. Pentru a adauga o noua sursa, selectati Create New Souce din meniul Process for: xc2s200e-6pq208 si prima fereastra de dialog New Source va aparea ca in figura de mai jos:

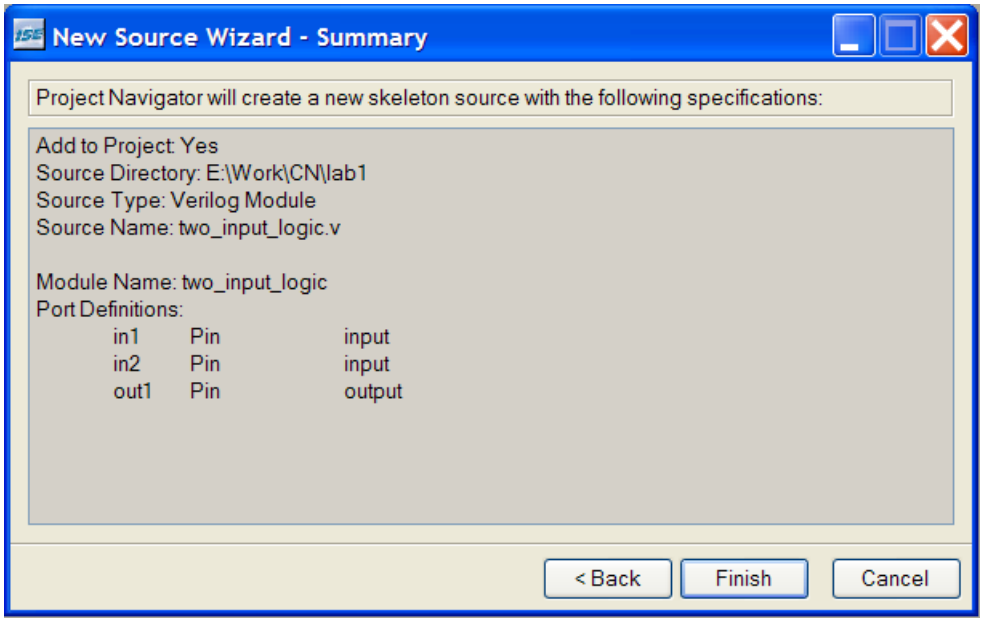


Selectati Verilog Module pentru a indica faptul ca doriti sa creati un modul verilog si atribuiti un nume pentru acesta apoi selectati Next.

Urmatoarea fereastra de dialog va perimite sa definiti intrarile si iesirile modulului dumneavoastra (acestea fot fi schimbate si ulterior in editorul de text). Dupa ce introduceti informatiile necesare selectati Next.



Ultima fereastra de dialog New Source prezinta rezumatul celor selectate anterior. Selectati Finish. Noua sursa va fi deschisa automat in editorul de text.

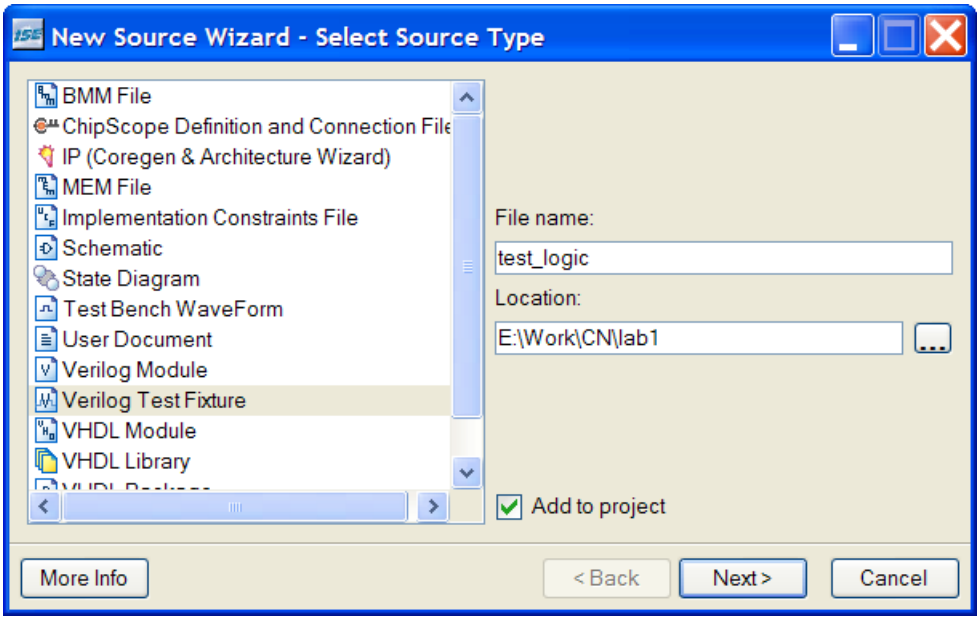


Completati sursa pentru a implementa operatia dorita dupa cum urmeaza:

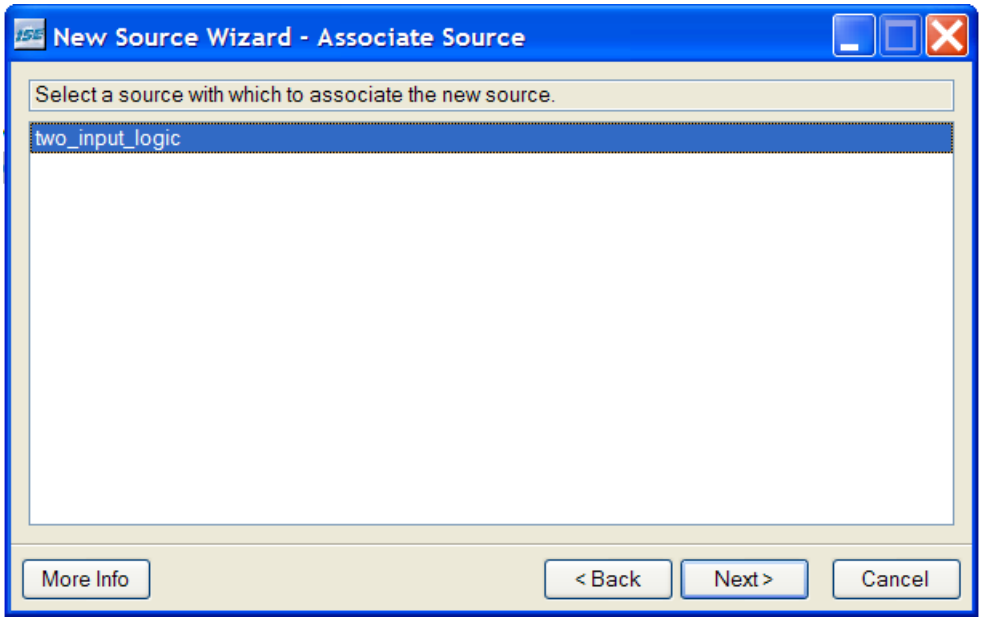
```
21  module two_input_logic(in1, in2, out1);
22      input in1;
23      input in2;
24      output out1;
25
26      assign out1 = in1 ^ in2;
27
28  endmodule
```

Simularea functionala

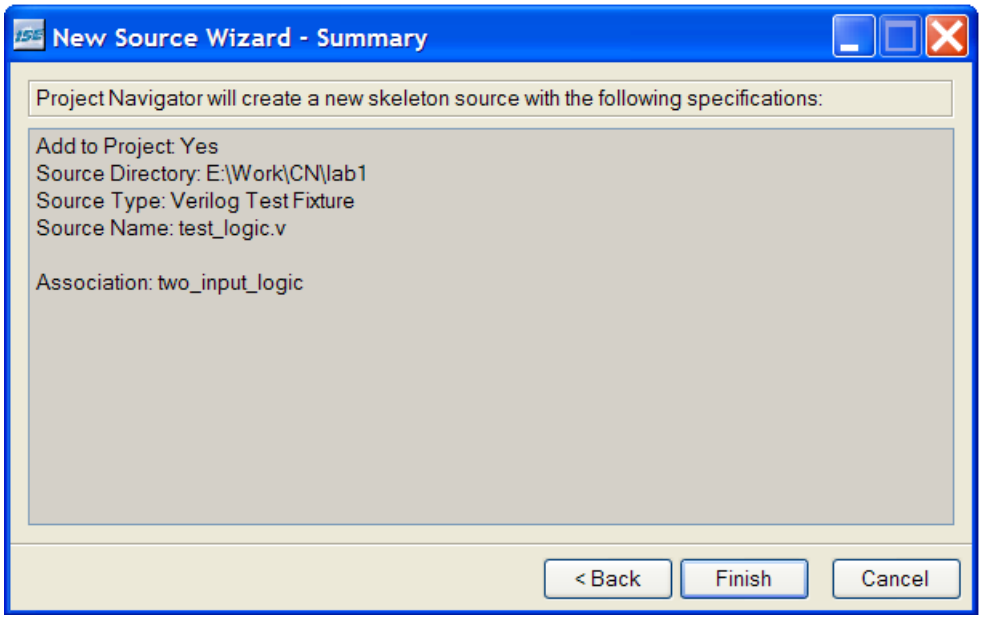
Simularea functionala are rolul de a verifica logica proiectata inainte de sinteza pentru a elimina eventualele greseli sau neconcordante intre specificatie si implementare.
Mediul ISE se integreaza perfect cu simulatorul ModelSim si permite lansarea simularii din mediul de dezvoltare. Pentru a simula un proiect este nevoie sa creati un modul de test. Acest lucru se face similar cu procesul de a crea o sursa noua selectand obtiunile ca in figura urmatoare:



Selectati Verilog Test Fixture pentru a genera un modul de test Verilog. Apoi atribuiti un nume pentru acest modul si selectati Next.
A doua fereastra de dialog va permite sa alegeti modulul pentru care doriti sa creati testul (in acest caz lista contine un singur modul: two_input_logic).



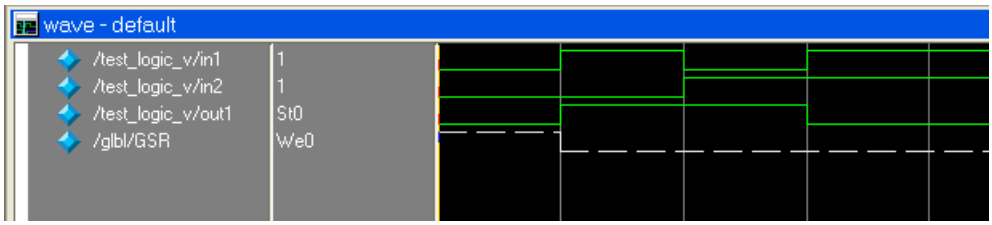
Ultima fereastra de dialog va prezinta sumarul obtiunilor anterioare. Selectati Finish.



Dupa crearea modulului de test, acesta va fi automat descis in editorul de test si trebuie completat cu valorile stimulilor de intrare (in1 si in2) pentru a testa diferitele combinatii. Un exemplu este oferit in cele ce urmeaza:

```
25  module test_logic_v;
26
27     // Inputs
28     reg in1;
29     reg in2;
30
31     // Outputs
32     wire out1;
33
34     // Instantiate the Unit Under Test (UUT)
35     two_input_logic uut (
36        .in1(in1),
37        .in2(in2),
38        .out1(out1)
39     );
40
41     initial begin
42        // Initialize Inputs
43        in1 = 0;
44        in2 = 0;
45
46        // Wait 100 ns for global reset to finish
47        #100;
48
49        // Add stimulus here
50        in1 = 1;
51        in2 = 0;
52
53        // Wait 100 ns for global reset to finish
54        #100;
55        in1 = 0;
56        in2 = 1;
57
58        // Wait 100 ns for global reset to finish
59        #100;
60        in1 = 1;
61        in2 = 1;
62
63     end
64
65  endmodule
```

Pentru a porni simularea selectati Sources for: Behaivioral Simulation pentru a putea selecta modulul de test nou creat. Dupa selectare, in meniul de proces (Processes for: test_logic_v) puteti selecta Simulate Behaivioral Model (de la ModelSim Simulator). Simulatorul va fi pornit automat si va simula circuitul implementat si stimulat de tatele de intrare (conform testului). Rezultatele pot fi urmarite in fereastra Wave din ModelSim.



Sinteza circuitului proiectat

Cu o descriere in Verilog corecta, din punct de vedere functional, se poate trece la transforamrea acesteia intr-un netlist (prin sinteza). Un netlist este o reprezentare a unei scheme logice pe intelesul masinii. In cazul de fata vom folosi XST care este integrat in ISE si poate realiza sinteza pentru dispozitive produse de Xilinx.
Pentru pornirea sintezei trebuie selectat intai modulul principal (two_input_logic) din meniul Sources for: Synthesis/Implementation si apoi din meniul Processes for: two_input_logic selectati Synthesis – XST. In consola mediului de dezvoltare ISE veti putea urmari mesajele generate in urma procesului de sinteza. In mod normal nu ar trebui sa apara nici o eroare in cadrul acestui proces deoarece codul a fost validat anterior. Pentru a analiza raportul sintezei puteti selecta View Synthesis Report.

Release 9.2.02i - xst J.38
Copyright (c) 1995-2007 Xilinx, Inc. All rights reserved.
--> Parameter TMPDIR set to ./xst/projnav.tmp
CPU : 0.00 / 0.81 s | Elapsed : 0.00 / 0.00 s

--> Parameter xsthdpdir set to ./xst
CPU : 0.00 / 0.81 s | Elapsed : 0.00 / 0.00 s

--> Reading design: two_input_logic.prj

TABLE OF CONTENTS


```
=========================================================================
*                      Synthesis Options Summary                        *
=========================================================================
```

---- Source Parameters
Input File Name : "two_input_logic.prj"
Input Format : mixed
Ignore Synthesis Constraint File : NO

---- Target Parameters
Output File Name : "two_input_logic"
Output Format : NGC
Target Device : xc2s200e-6-pq208

---- Source Options
Top Module Name : two_input_logic
Automatic FSM Extraction : YES
FSM Encoding Algorithm : Auto
Safe Implementation : No
FSM Style : lut
RAM Extraction : Yes
RAM Style : Auto
ROM Extraction : Yes
Mux Style : Auto
Decoder Extraction : YES
Priority Encoder Extraction : YES
Shift Register Extraction : YES
Logical Shifter Extraction : YES
XOR Collapsing : YES
ROM Style : Auto
Mux Extraction : YES
Resource Sharing : YES
Asynchronous To Synchronous : NO
Multiplier Style : lut
Automatic Register Balancing : No

---- Target Options
Add IO Buffers : YES
Global Maximum Fanout : 100
Add Generic Clock Buffer(BUFG) : 4
Register Duplication : YES
Slice Packing : YES
Optimize Instantiated Primitives : NO
Convert Tristates To Logic : Yes
Use Clock Enable : Yes
Use Synchronous Set : Yes
Use Synchronous Reset : Yes
Pack IO Registers into IOBs : auto
Equivalent register Removal : YES

---- General Options
Optimization Goal : Speed
Optimization Effort : 1
Library Search Order : two_input_logic.lso
Keep Hierarchy : NO
RTL Output : Yes
Global Optimization : AllClockNets
Read Cores : YES
Write Timing Constraints : NO
Cross Clock Analysis : NO

```
Hierarchy Separator              : /
Bus Delimiter                    : <>
Case Specifier                   : maintain
Slice Utilization Ratio          : 100
BRAM Utilization Ratio           : 100
Verilog 2001                     : YES
Auto BRAM Packing                : NO
Slice Utilization Ratio Delta    : 5

=========================================================================


=========================================================================
*                     HDL Compilation                   *
=========================================================================
Compiling verilog file "two_input_logic.v" in library work
Module <two_input_logic> compiled
No errors in compilation
Analysis of file <"two_input_logic.prj"> succeeded.


=========================================================================
*                  Design Hierarchy Analysis                    *
=========================================================================
Analyzing hierarchy for module <two_input_logic> in library <work>.


=========================================================================
*                     HDL Analysis                      *
=========================================================================
Analyzing top module <two_input_logic>.
Module <two_input_logic> is correct for synthesis.


=========================================================================
*                     HDL Synthesis                     *
=========================================================================

Performing bidirectional port resolution...

Synthesizing Unit <two_input_logic>.
    Related source file is "two_input_logic.v".
    Found 1-bit xor2 for signal <out1>.
Unit <two_input_logic> synthesized.


=========================================================================
HDL Synthesis Report

Macro Statistics
# Xors                           : 1
 1-bit xor2                      : 1

=========================================================================

=========================================================================
```

```
*                  Advanced HDL Synthesis                  *
=========================================================================

Loading device for application Rf_Device from file '2s200e.nph' in environment C:\Xilinx92i.

=========================================================================
Advanced HDL Synthesis Report

Macro Statistics
# Xors                                  : 1
 1-bit xor2                            : 1

=========================================================================

=========================================================================
*                  Low Level Synthesis                  *
=========================================================================

Optimizing unit <two_input_logic> ...

Mapping all equations...
Building and optimizing final netlist ...
Found area constraint ratio of 100 (+ 5) on block two_input_logic, actual ratio is 0.

Final Macro Processing ...

=========================================================================
Final Register Report

Found no macro
=========================================================================

=========================================================================
*                  Partition Report                  *
=========================================================================

Partition Implementation Status
-------------------------------

  No Partitions were found in this design.

-------------------------------

=========================================================================
*                  Final Report                  *
=========================================================================
Final Results
RTL Top Level Output File Name     : two_input_logic.ngr
Top Level Output File Name         : two_input_logic
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 3
```

Cell Usage :
BELS : 1
LUT2 : 1
IO Buffers : 3
IBUF : 2
OBUF : 1
==========

Device utilization summary:

Selected Device : 2s200epq208-6

Number of Slices: 1 out of 2352 0%
Number of 4 input LUTs: 1 out of 4704 0%
Number of IOs: 3
Number of bonded IOBs: 3 out of 142 2%

Partition Resource Summary:

No Partitions were found in this design.

==========
TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

No clock signals found in this design

Asynchronous Control Signals Information:

No asynchronous control signals found in this design

Timing Summary:

Speed Grade: -6

Minimum period: No path found
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 7.707ns

Timing Detail:

All values displayed in nanoseconds (ns)

==========

```
Timing constraint: Default path analysis
  Total number of paths / destination ports: 2 / 1
--------------------------------------------------------------------------------
Delay:               7.707ns (Levels of Logic = 3)
  Source:            in2 (PAD)
  Destination:       out1 (PAD)

  Data Path: in2 to out1
                        Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             1   0.797   0.920  in2_IBUF (in2_IBUF)
     LUT2:I0->O            1   0.468   0.920  Mxor_out1_Result1 (out1_OBUF)
     OBUF:I->O                 4.602          out1_OBUF (out1)
    ----------------------------------------
    Total                     7.707ns (5.867ns logic, 1.840ns route)
                              (76.1% logic, 23.9% route)

=========================================================================
CPU : 4.17 / 5.14 s | Elapsed : 5.00 / 5.00 s

-->

Total memory usage is 131760 kilobytes

Number of errors   :    0 (   0 filtered)
Number of warnings :    0 (   0 filtered)
Number of infos    :    0 (   0 filtered)
```

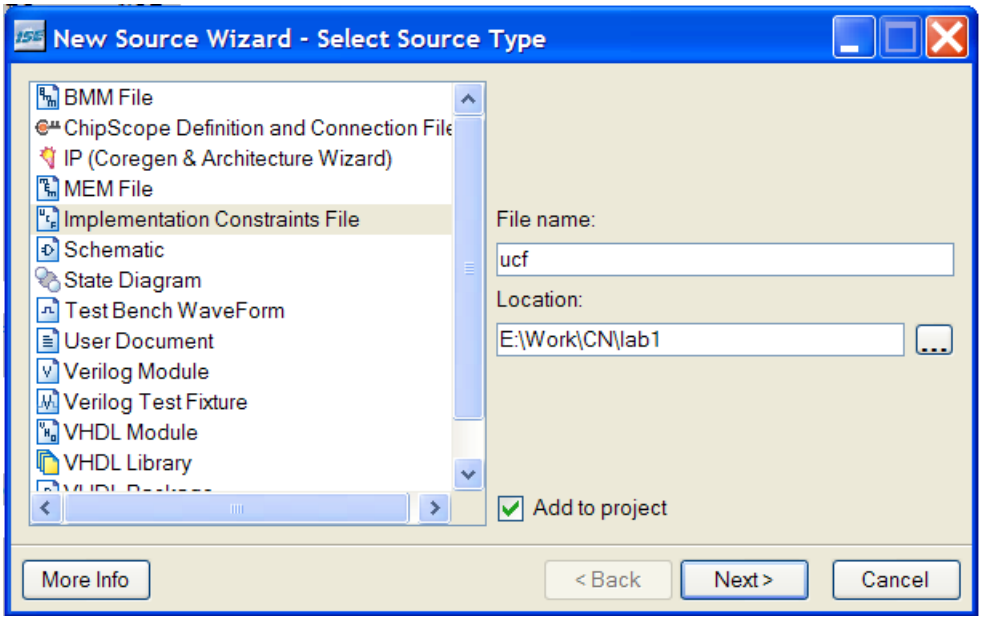
In urma analizei raportului se poate observa cate (si ce tip) de resurse utilizeaza proiectul dumneavoastra. Aceasta informatie este utila in cazul productiei cand se doreste alegerea circuitului care se potriveste cel mai bine pentru respectivul proiect (pentru a nu face risipa de resurse).

Implementarea proiectului

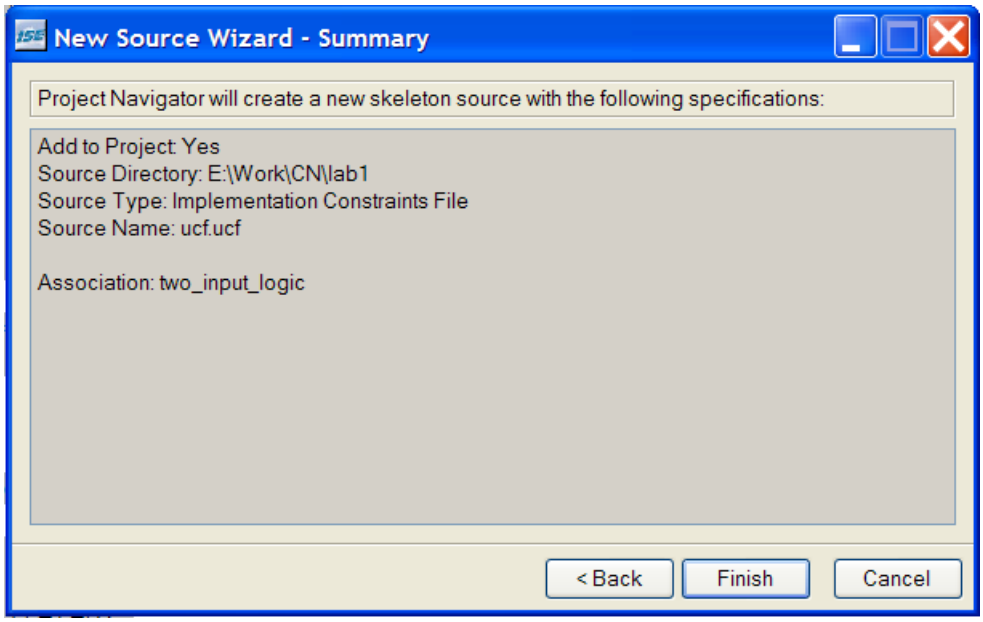
Implementarea proiectului consta intr-o succesiune de evenimente ce conduce la transforamrea codului sintetizat in netlist intr-un fisier de configurare pentru FPGA-ul ales. Uneltele de implementare (incluse in ISE) trebuie sa stie la ce pini ai FPGA-ului sa conecteze intrarile si iesirile proiectului dumneavoastra. Daca acest lucru nu este definit, ele vor fi distribuite aleator pinilor disponibili.
Pentru proiectul propus, vom folosi doua switchuri pentru intrari si un led pentru iesire.
Lista pinilor conectati la diversele resurse ale placi de I/O sunt prezentati in tabelul din anexa I.
Pentru acest proiect vom conecta:

SW1	P23
SW2	P22
LED1	P11

Dupa ce am adunat informatiile necesare, putem trece la realizarea fisierului UCF (fisierul de constrangeri ale proiectului). Acest fisier defineste corespondenta intre semnalele modulului de top si pinii FPGA-ului precum si alte setari de constrangere (cum ar fi tipul iesirilor, etc). Pentru a pastra simplitatea proiectului vom face numai setarile necesare pentru conectarea pinilor.
Este util ca toate constrangerile impuse proiectului sa fie definite intr-un fisier separat ce nu depinde de codul sintetizat.
Pentru a crea fisierul de constrangeri selectati din nou Create New Source.



Selectati Implementation Constraints File si atribuiti un nume (ucf). Apoi selectati Next.



Dupa ce analizati informatiile din sumar, selectati Finish.

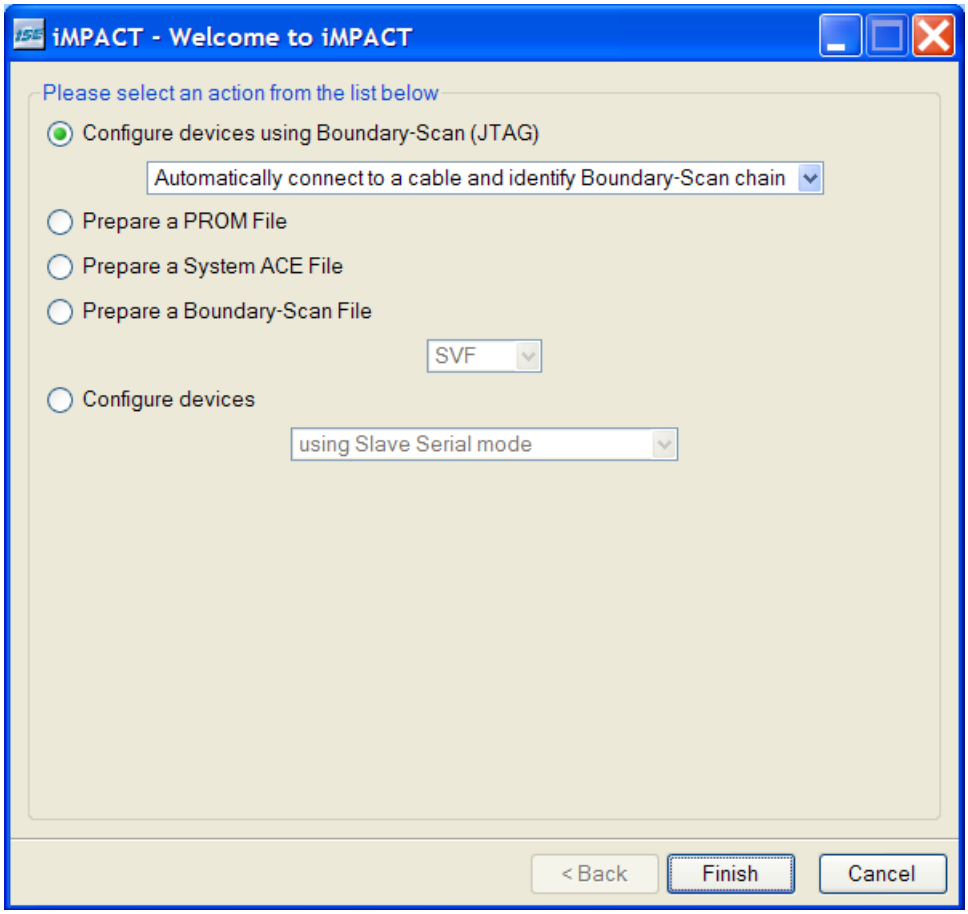
Sub ierarhia modulului de top (two_input_logic) va aparea un nou fisier (ucf.ucf). Fisierul UCF este un fisier text (ca si sursele Verilog). Pentru simplitatea exemplului il vom ediata direct. Selectati fisierul ucf.ucf din meniul Sources for: Synthesis/Implementation si apoi din meniul Processes for: ucf.ucf, Edit Constraints (Text).
Fisierul rezultat va contine urmatoarele trei linii:

```
1  NET in1 LOC="P23";
2  NET in2 LOC="P22";
3  NET out1 LOC="P11";
```

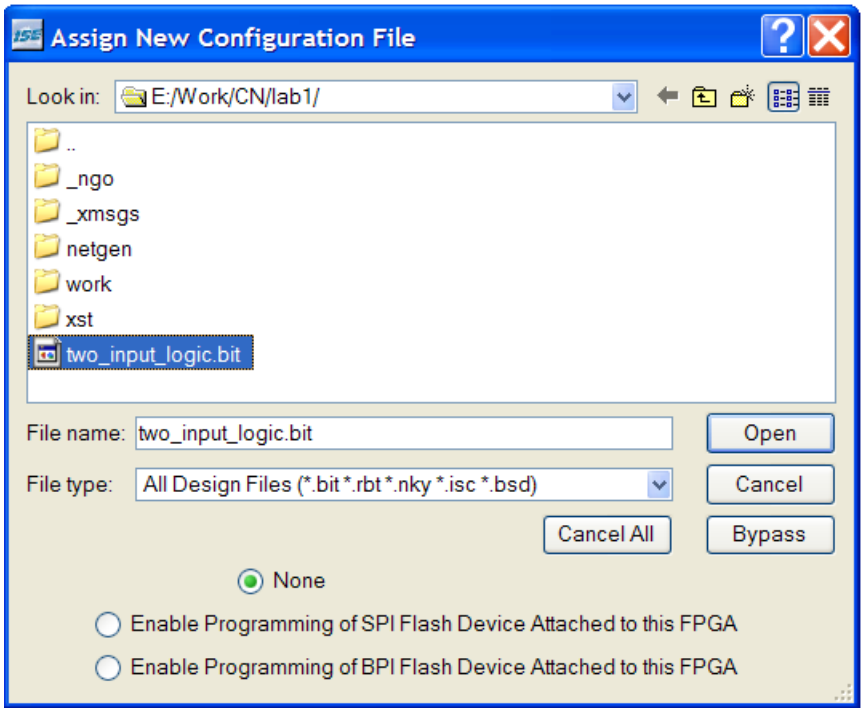
Semnificatia lor este urmaorarea: semnalului in1/in2/out1 i se atribiue pinul cu de la locatia P23/P22/P11 al FPGA-ului.
Dupa ce fisierul de constrangeri este complet, se poate trece la implementarea proiectului. Selectati din nou modulul de top (two_input_logic) si apoi Implement Design in meniul Processes for: two_input_logic.

Programarea FPGA-ului

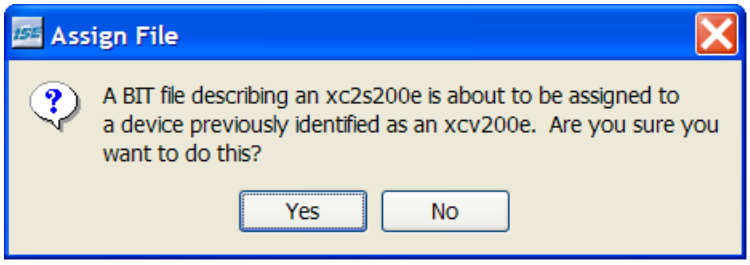
Pentru a putea programa FPGA-ul trebuie creat intai fisierul de configurare. Acesta se creaza selectand Generate Programming File din meniul Processes for: two_input_logic.
Dupa generare, fisierul (cu extensia .bit) trebuie transferat pe FPGA-ul de pe placa de test. Aceasta trebuie sa fie alimentata si conectata la calculator prin cablul de programare (JTAG).
Utilitarul ce se ocupa de configurare se numeste iMPACT si se porneste selectand Configure Device (iMPACT).



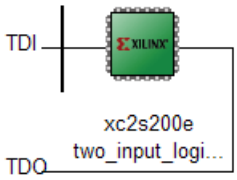
Pentru a detecta circuitul selectati Finish.



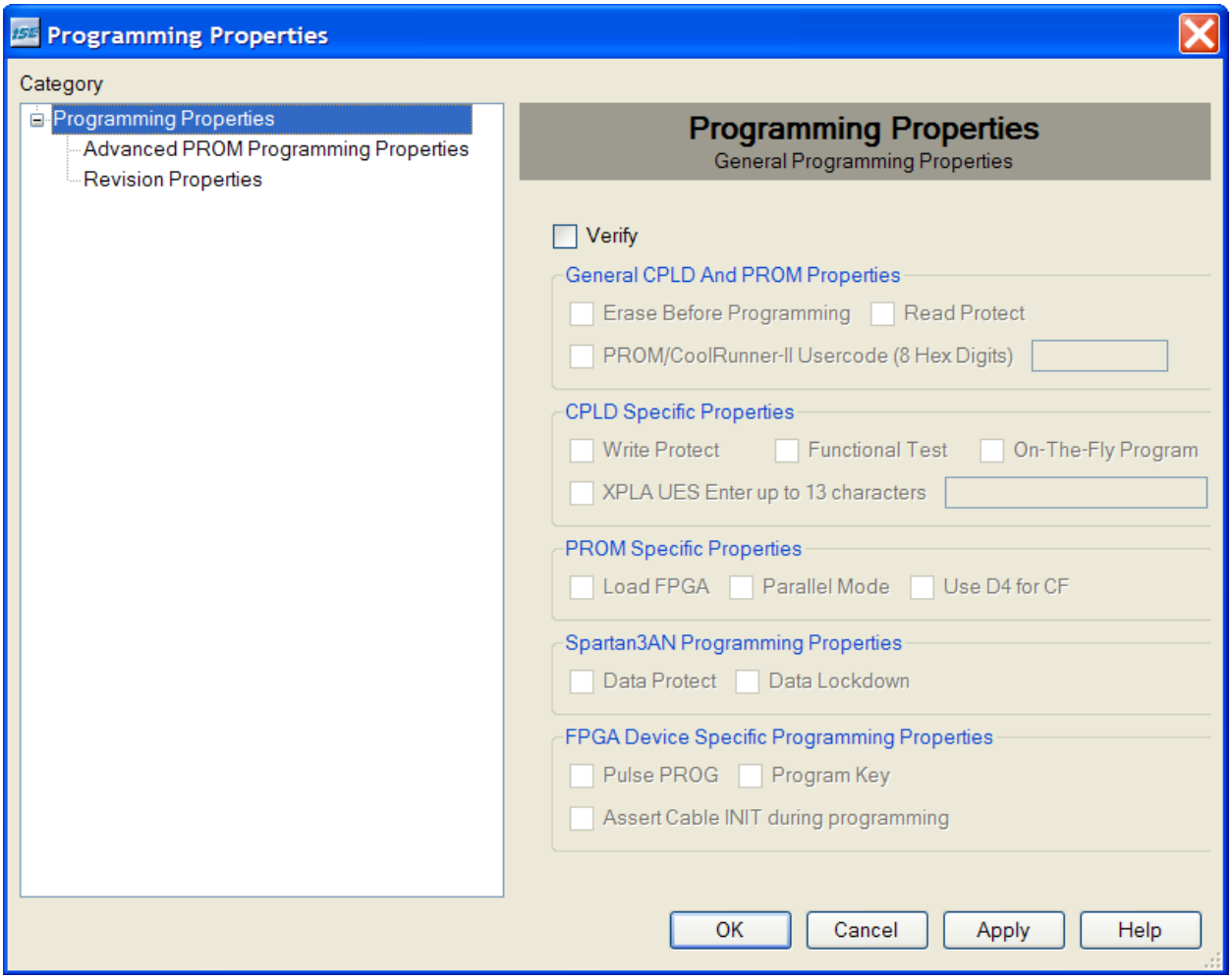
Selectati fisierul .bit ce doriti sa il transferati si apoi selectati Open. Confirmati ca doriti sa programati FPGA-ul cu fisierul .bit selectand Yes.



In acest moment lantul de programare este gata. Tot ce mai trebuie facut este sa porniti programarea selectand circuitul din lant si optiunea Program.



In fereastra de dialog deschisa, selectati OK.



Dupa programare se poate trece la verificarea functionarii prin actionarea switchurilor de pe placa de test si observarea ledului corespunzator.

Exercitii suplimentare

Pentru exersarea celor invatate in acest laborator va propunem implementarea altor functii logice (mai complexe) avand ca intrare cele 8 switchuri si ca iesire cele 8 leduri disponibile pe placa de test.