2. Implementarea porţilor logice în ISE Xilinx.

2.1 Utilizarea editorului de scheme din ISE WebPack

XILINX ISE (Integrated Software Enviroment) este un mediu de proiectare a sistemelor digitale integrate implementate pe circuitele programabile XILINX, care oferă o gamă largă de aplicaţii pentru proiectare utilizând circuite FPGA sau CPLD. Proiectarea se poate realiza utilizând atât descrierea schematică, cât şi limbaje de descriere hardware (VHDL sau Verilog).

Pentru exemplificare, se consideră proiectarea unui sistem digital combinaţional care să cuprindă tipurile de porţi logice existente. Pentru aceasta se consideră ca date de intrare 2 biţi care intră succesiv în cele 4 porţi (INV, OR, XOR, AND).

2.2 Pornirea Xilinx ISE

Pentru a lansa *ISE WebPack,* dublu click pe iconiţa asociată programului sau se apelează meniul:

Start —>All Programs —>Xilinx ISE —> ISE —> Project Navigator.

Fereastra principală, *Project Navigator*, oferă o interfaţă care organizează toate fişierele şi programele asociate cu modelul sistemului digital proiectat. Fereastra este împarţită în patru sub-ferestre:

- sub-fereastra surselor (source panel) în care sunt afişate toate fişierele asociate proiectului curent;
- sub-fereastra proceselor (process panel) în care se află toate procesele şi operaţiile disponibile pentru fişierul selectat;
- sub-fereastra consolă (transcript panel) în care este prezentată starea proceselor, atenţionările şi erorile rezultate în urma unui proces;
- sub-fereastra editor (editor panel) unde se afişează codul unui fişier selectat.

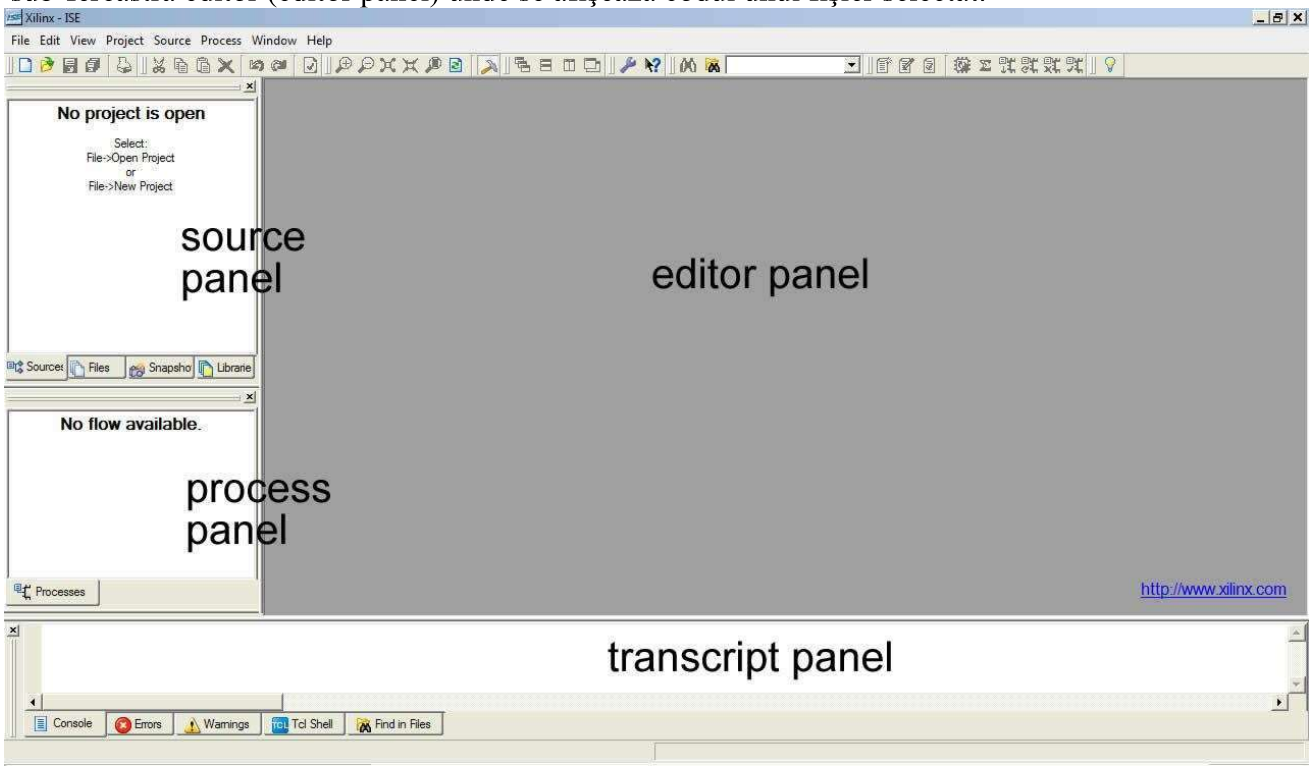


Figura 1.4 Fereastra principală XILINX ISE Webpack.

2.3 Crearea unui proiect nou

Pentru a crea un proiect nou se apelează meniul *File —> New Project*

Se completează câmpurile cu numele proiectului, locaţia, iar în câmpul *Top-level source type* se selectează Schematics. Apoi click *Next*. Se denumeşte proiectul functionQ. Proiectul constă dintr-un set de fişiere ce se vor crea în directorul selectat.

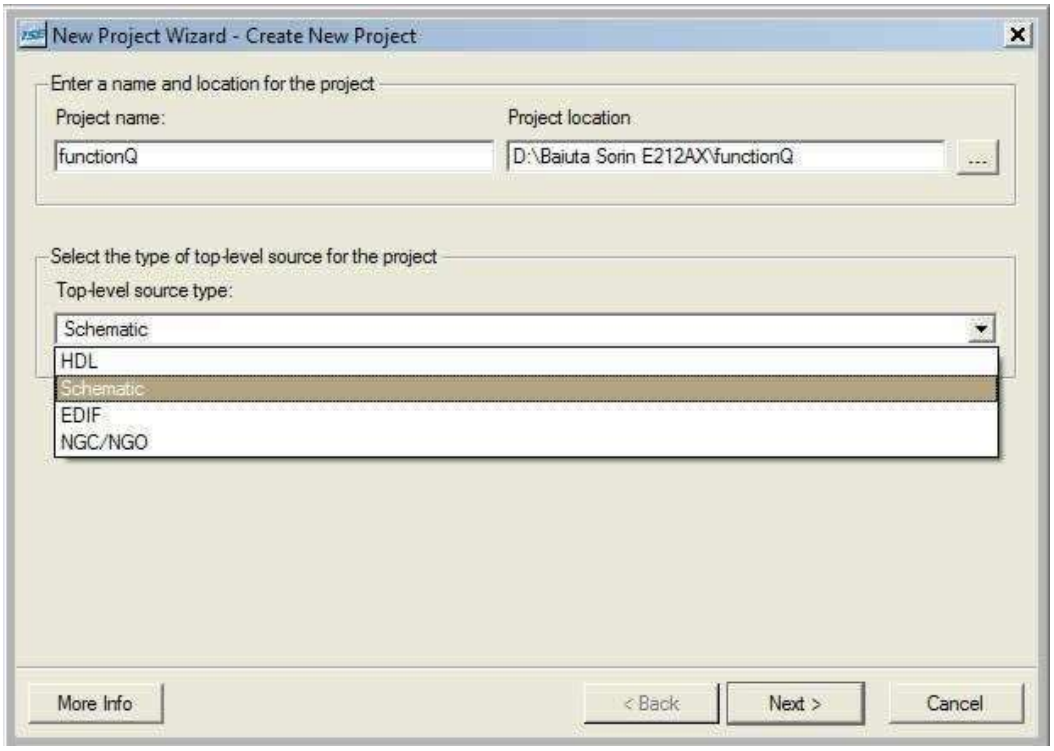


Figura 1.5 Fereastra *New Project Wizard Create New Project*.

În fereastra de descriere a dispozitivului, prezentat in figura 1.6, se completează, câmpurile ce descriu circuitul Xilinx utilizat. Apoi click *Next*.

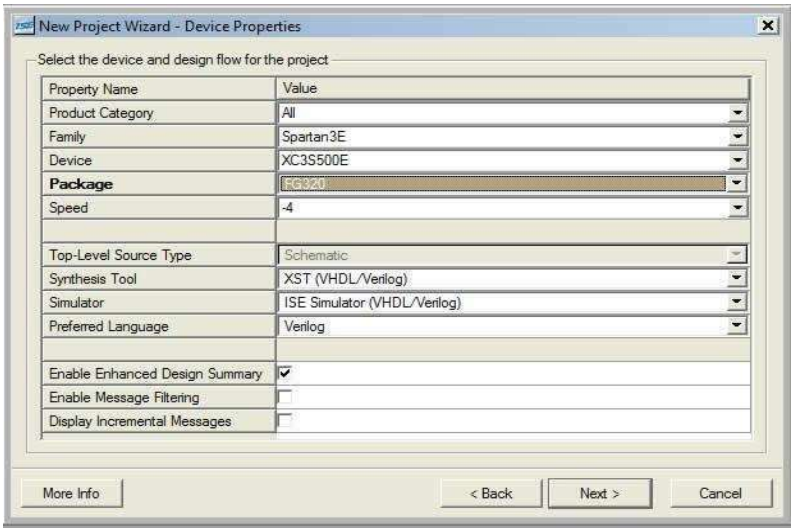


Figura 1.6 Fereastra *New Project Wizard Device Properties*.

Placa utilizată conţine un dispozitiv FPGA cu următoarele caracteristici:
Device Family: Spartan3E
Device: XC3S500E
Package: FG320
Speed Grade: -4
Synthesis Tool: XST (VHDL/Verilog)
Simulator: ISE Simulator (VHDL/Verilog)
Apoi se debifează *Enable Message Filtering*.

Următoarele două ferestre care vor apărea, *New Project Wizard Create New Source* şi *New Project Wizard Add Existing Sources* sunt prezentate în figurile 1.5 şi 1.6. Prin aceste două ferestre se pot crea fişiere sursă noi sau se pot include în proiect fişiere sursă existente. Atât crearea unor descrieri noi cât şi includerea în proiect a unor surse existente se pot face şi ulterior creării proiectului, apelând la meniul *Project —> New Source... ; Project —> Add Source...*
Se amâna includerea fişierelor în proiect, apoi se face click pe *Next* în cele două ferestre.

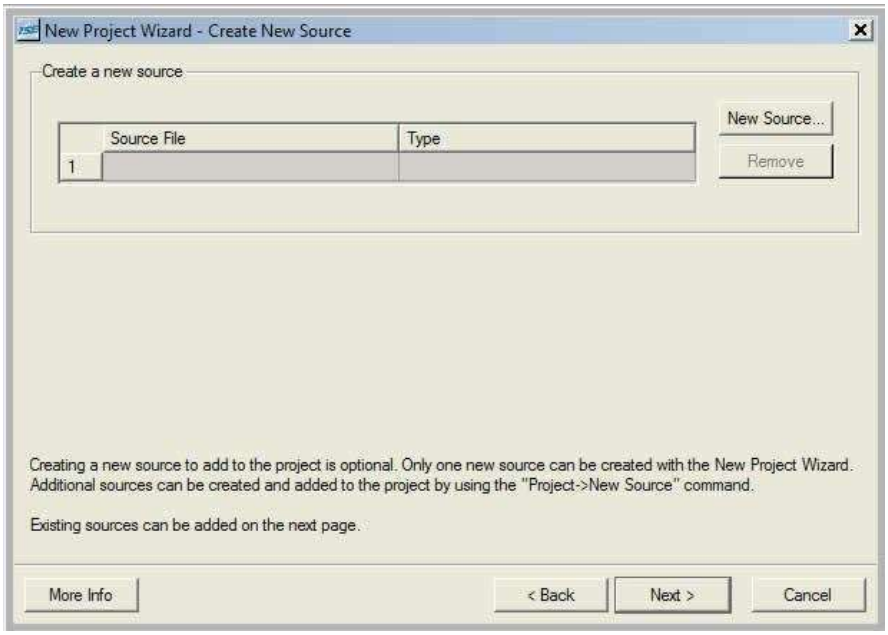


Figura 1.7 Fereastra *New Project Wizard Create New Source*.

În ultima fereastră care va aparea, *New Project Wizard Project Summary*, se găseşte un sumar cu toate datele referitoare la proiect şi la modelul folosit. Click *Finish*.

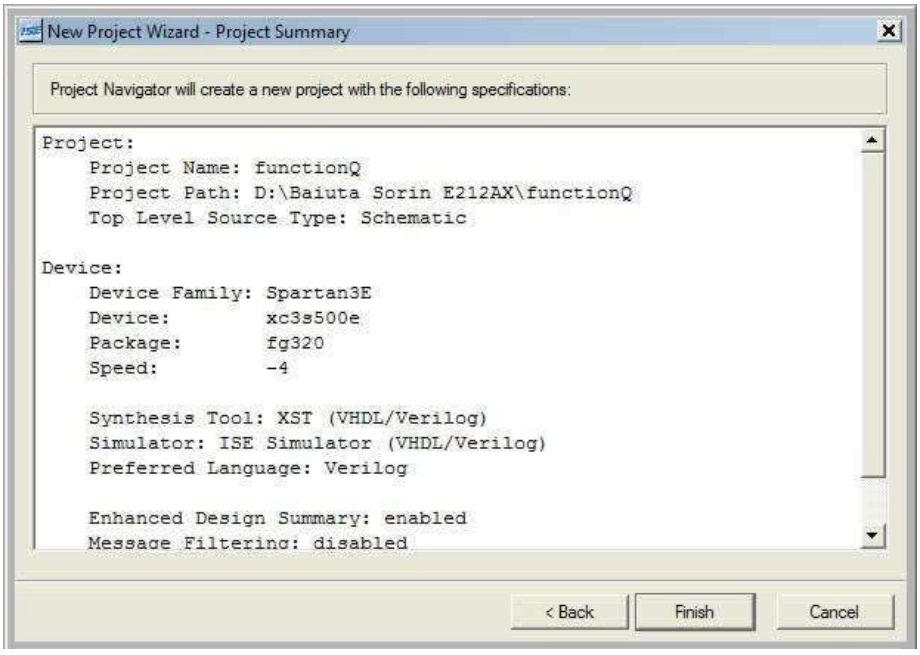


Figura 1.8 Fereastra *New Project Wizard Project Summary*.

2.4 Crearea unui fişier pentru o schemă (.sch)

Pentru a crea un fişier sursă schemă, se selectează în sub-fereastra surselor *Sources* modelul xc3s500e-4fg320, se dă click dreapta pe el, apoi click stânga pe *New Source...*

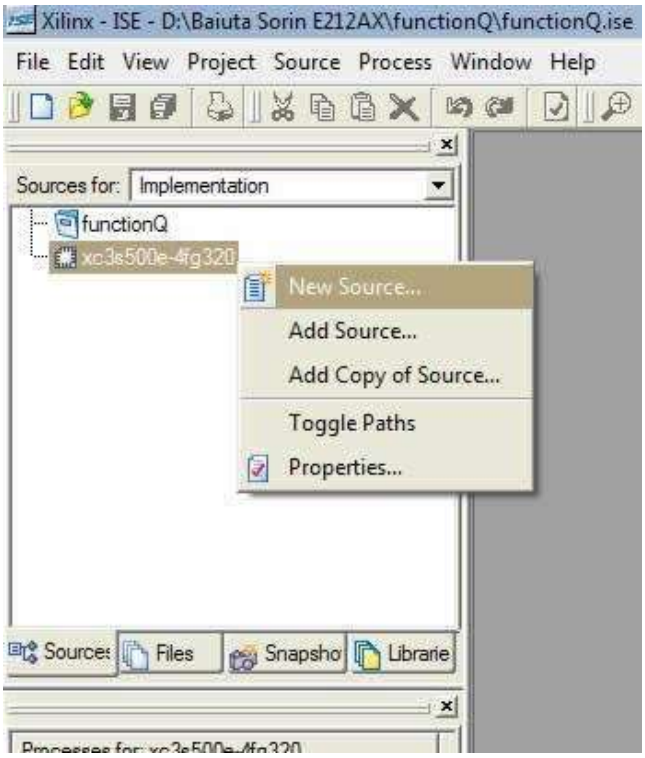


Figura 1.9 Sub-fereastra *Sources* şi sub-meniul proiectului.

În fereastra *New Source Wizard Select Source Type* se selectează Schematic şi se completează numele fişierului sursă ce se va genera şi locaţia acestuia în sistemul de fişiere al calculatorului. Se denumeşte fişierul functionQ. Fişierul salvat va avea extensia .sch. Se asigură selectarea opţiunii Add to project. Click *Next* şi *Finish.*

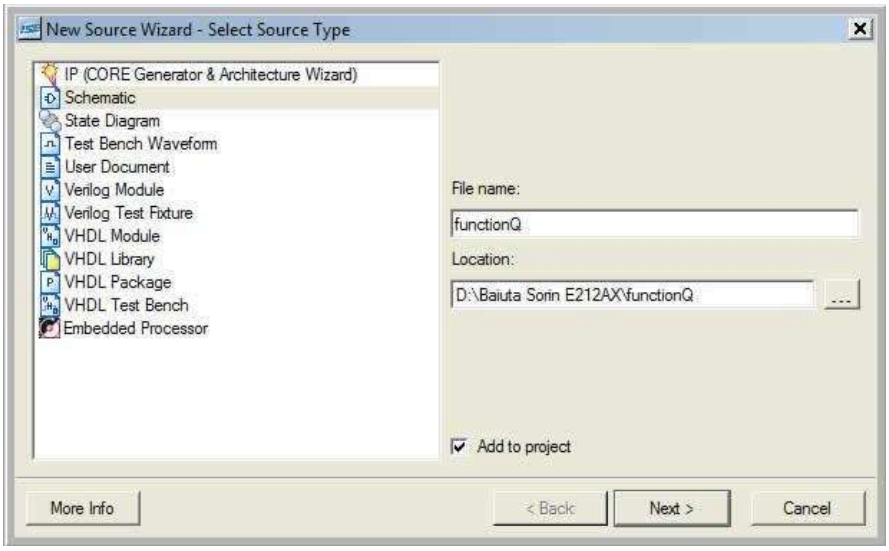


Figura 1.10 Fereastra *New Source Wizard Select Source Type*.

2.5 Adăugarea componentelor

În partea stangă în fereastra *Sources* se găseşte lista cu categoriile de simboluri şi cea cu simbolurile propriu-zise. Pentru a se găsi cu uşurinţa simbolurile necesare (aici porţile logice) se selectează la *Categories* Logic. În lista *Symbols* se vor afişa doar simbolurile de la porţile logice. Dacă se doreşte căutarea unui simbol după denumire, se scriu primele litere în *Symbol Name Filter*, realizându-se un filtru, iar în lista *Symbols* vor apărea doar simbolurile a căror denumire începe cu acele litere.

Se selectează simbolul dorit. Pentru a-l plasa în cadrul schemei, se face click în locul în care se doreşte a fi pus. Astfel se plasează toate simbolurile schemei, astfel:

- Un inversor (INV); o poartă „sau" (OR2); o poartă „sau exclusiv"(XOR2); o poartă „şi"(AND2).

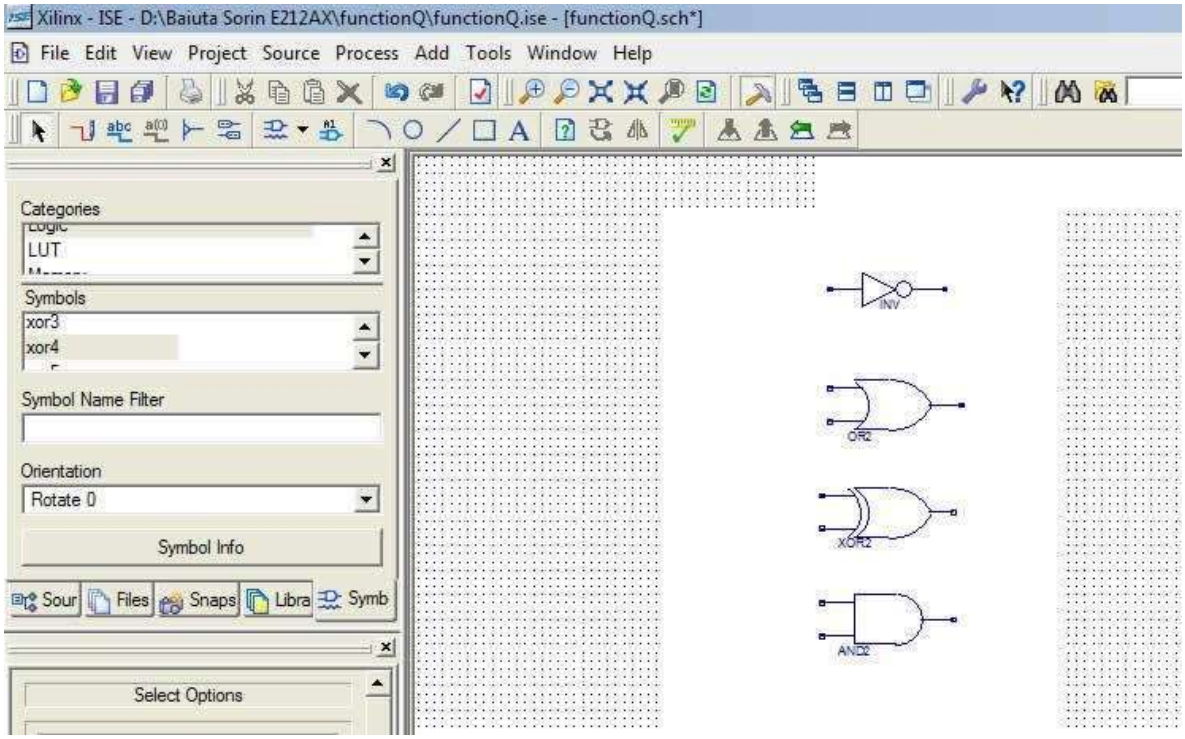


Figura 1.11 Adăugarea simbolurilor în schema.

2.6 Editarea de conexiuni

Odată plasate componentele, acestea trebuie legate între ele prin intermediul unor conexiuni simple sau magistrale.

Pentru a realiza o conexiune se apelează comanda:

Add –> Wire (CTRL+W)

sau

click pe iconiţa *Add Wire*  .

Pentru a plasa o conexiune, după ce aţi dat click pe iconiţa faceţi click pe primul pin şi dublu+click pe cel de al doilea pin. Unui fir i se poate asocia o denumire utilizând comanda:

Add –> Net Name (CTRL+D)

sau

click pe iconiţa *Add Net Name*  .

2.7 Adăugarea terminalelor de intrare şi de ieşire

Pentru a adăuga terminale de intrare şi de ieşire se utilizează comanda:
Add –> I/O Marker (CTRL+G)

sau

click pe iconiţa *Add I/O Marker*  .

Se modifică (prin redenumire) numele implicite ale porturilor cu numele specifice exemplului ales (IN1 IN2). Pentru a redenumi un terminal, se efectuează click dreapta pe el şi se selectează *Rename port*.

Se remarcă faptul că denumirile de fire şi porturi depind de literele majuscule sau minuscule ce le conţin (eng. ”case sensitive”).

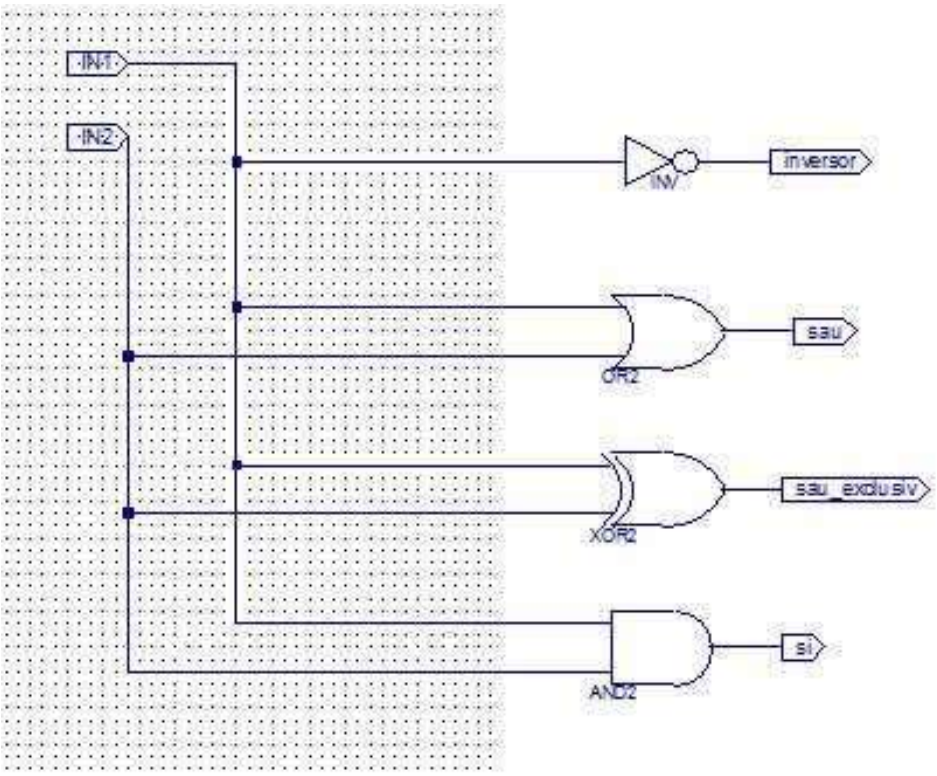


Figura 1.12. Schema de porţi logice asociată funcţiei Q

2.8 Verificarea schemei

Pentru a verifica schema din punct de vedere electric se execută comanda:

Tools –> Check Schematic

sau

click pe iconiţa *Check Schematic* .

În urma verificării schemei se vor raporta posibilile erori de conectare a firelor, porturilor sau componentelor. În cazul unor erori acestea vor fi raportate în fereastra consolă, împreună cu mesaje explicative. Dacă nu sunt erori, mesajul raportat este:

StartDRC...

No error or warning is detected

2.9 Salvarea schemei

Pentru salvarea schemei se execută comanda:

File –> Save (CTRL+S)

sau

click *Save*.

Se salvează schema în fişierul functionQ.sch.

2.10 Crearea simbolurilor şi utilizarea lor

Pentru a se uşura munca pe viitor, se pot realiza propriile simboluri sau se pot modifica alte simboluri existente în bibliotecă. Noile simboluri create vor fi plasate în bibliotecă locală de simboluri. Pe viitor, când e nevoie de utilizarea din nou a unei scheme se poate folosi direct ca un simbol.

Pentru a realiza un simbol nou se efectuează click pe *Tools –> Symbol Wizard*. În prima fereastră puteţi selecta forma simbolului şi modul de denumire a pinilor. Pentru uşurinţă selectaţi *Pin Name Source = Using Schematic*. În următoarea fereastră, *Symbol Wizard Pin Page* figura 4.12, porturile vor apărea completate cu denumirile folosite în schema. În cazul în care selectaţi *Specify manually* toate porturile vor trebui introduse manual.

După editarea simbolului, click *Next –> Next –> Finish*.

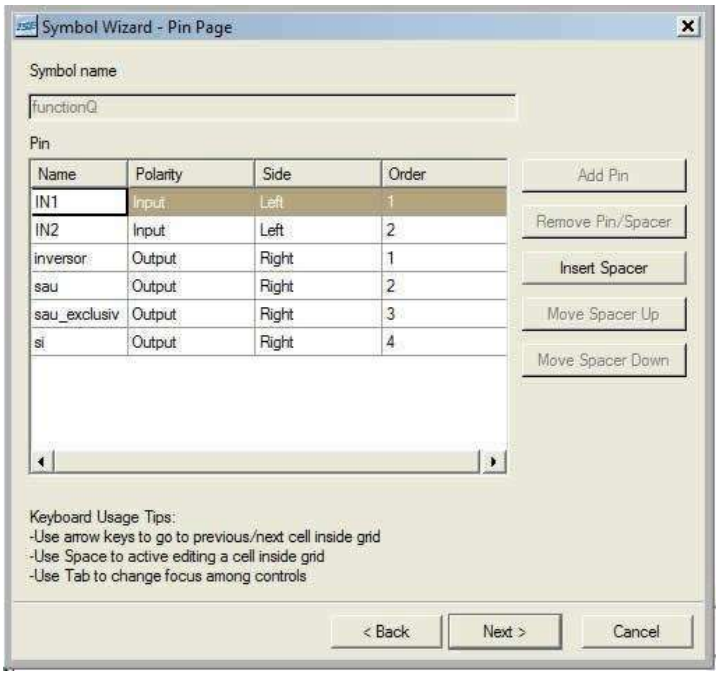


Figura 1.13 Fereastra *Symbol Wizard Pin Page*.

În ultimă fază, simbolul trebuie salvat într-un fişier cu extensia .sym. Numele simbolului va fi numele fişierului.

Atenţie! Pentru a folosi simbolul în alt proiect, se copiază în directorul noului proiect fişierul .sym si fişierul .sch asociat (schema internă a simbolului).

2.11 Testarea schemei.

Pentru a testa funcţionalitatea modelului schematic, trebuie creat un fişier test bench în care se furnizează valorile datelor de intrare pentru funcţia Q. Acest lucru se face astfel:

În submeniul *Source Panel* selectăm *Sources* apoi se adaugă un nou fişier sursa acestui proiect. Acesta este de tip *Test Bench Waveform*. Se dă numele *test_schematic* apoi Next, iar căsuţa *Add to project* trebuie bifată. Dupa ce s-a apăsat Next, apare o fereastră în care trebuie selectat fişierul cu care face asocierea fişierul de test creat (în cazul de faţa proiectul are un singur fişier ca sursă şi anume *functionQ*). Next şi Finish.

Noua fereastră care se deschide permite selectarea tipului de circuit al fişierului de test. Nu avem nevoie de un circuit secvenţial, deci la *Clock Information* se selectează *Combinatorial* iar la *Initial Length of Test Bench* se completează 1700 ns pentru a introduce toate valorile posibile la intrare, apoi Finish.

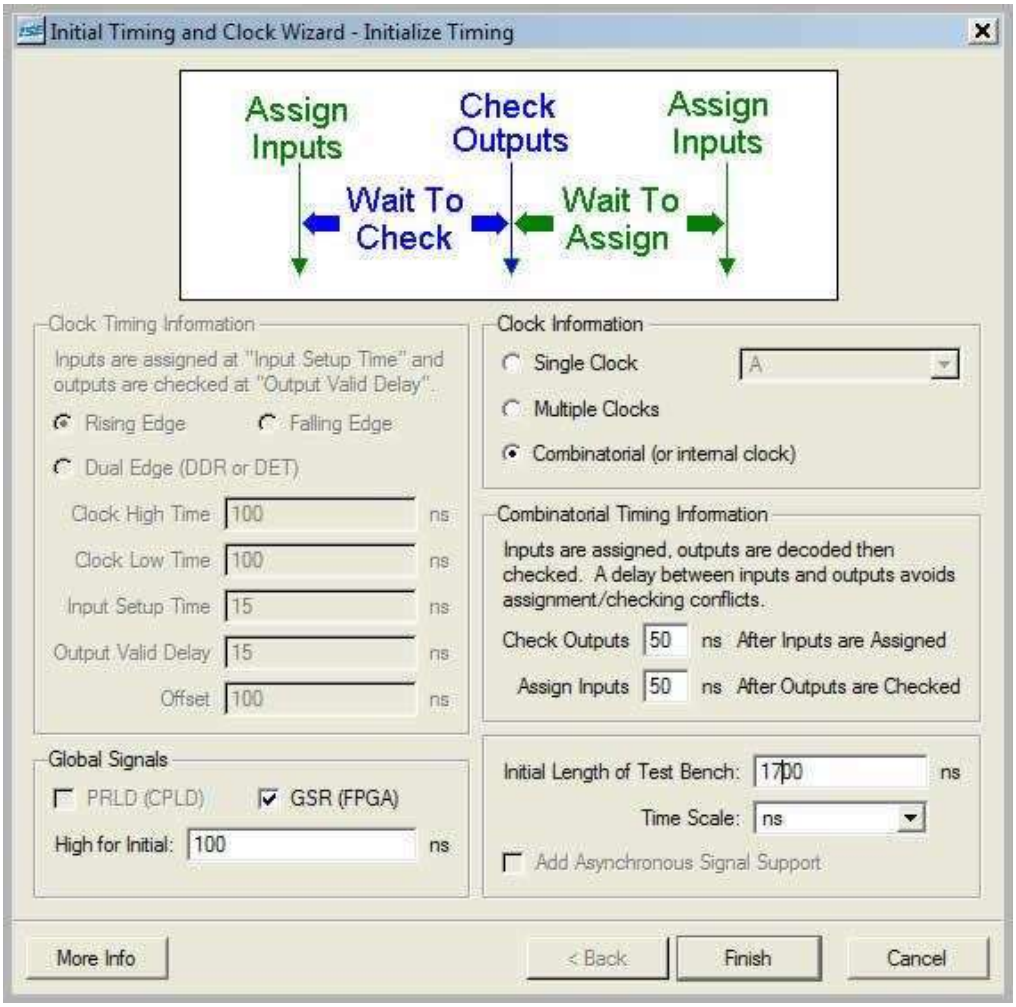


Figura 1.14. Fereastra Initialize Timing

Extensia noului fişier creat în proiect este .tbw.

În cadrul acestui fişier, se setează valorile de intrare prin click pe zonele albastre ale fiecărei intrări, astfel valorile intrărilor basculeaza între 0 şi 1. Se încearcă acoperirea tuturor posibilităţilor:

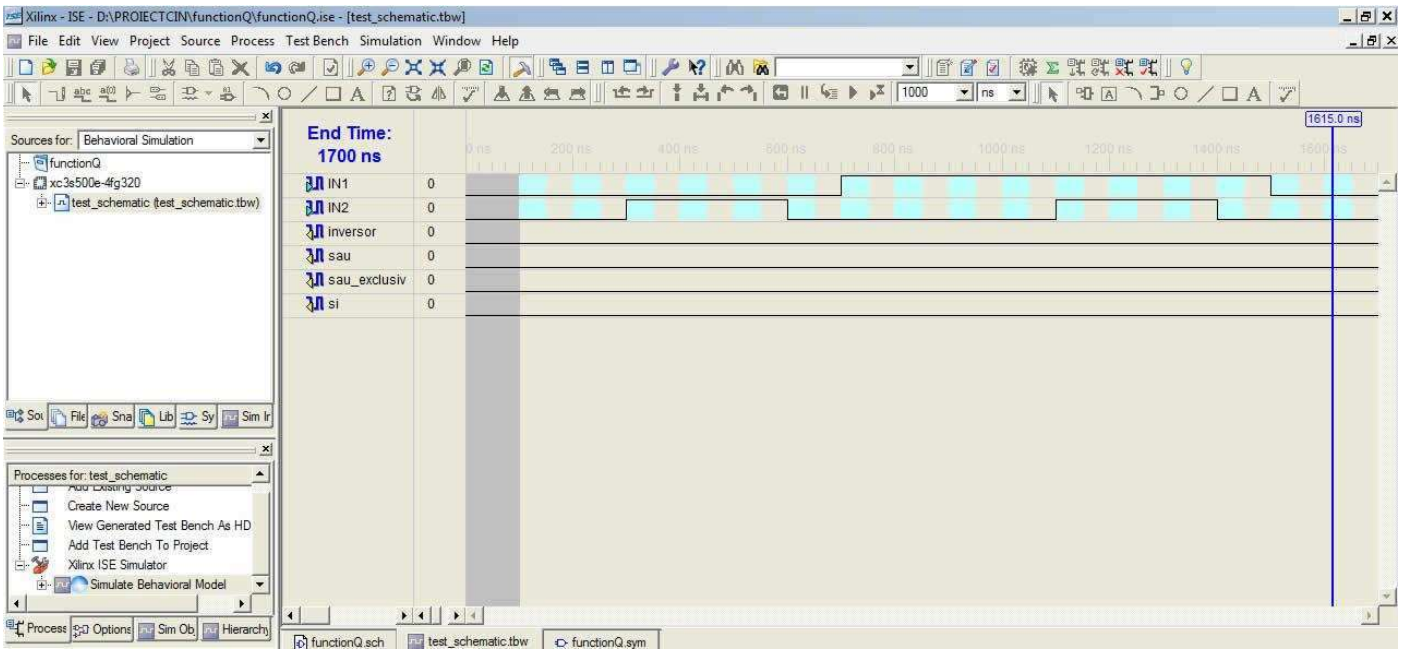


Figura 1.15. Schema fişierului de test pentru schematics

După ce s-a salvat fişierul *test_schematic.tbw*, în source panel se alege *Sources for: Behavioral Simulation*, se selectează fişierul *test_schematic (test_schematic.tbw)* iar din process panel se expandează Xilinx ISE Simulator, click dreapta pe *Simulate Behavioral Model -> Properties* iar la Simulation Run Time se completează 1700 ns, OK, apoi se efectuează dublu click pe Simulate Behavioral Model.

Astfel se obţine următoarea reprezentare:

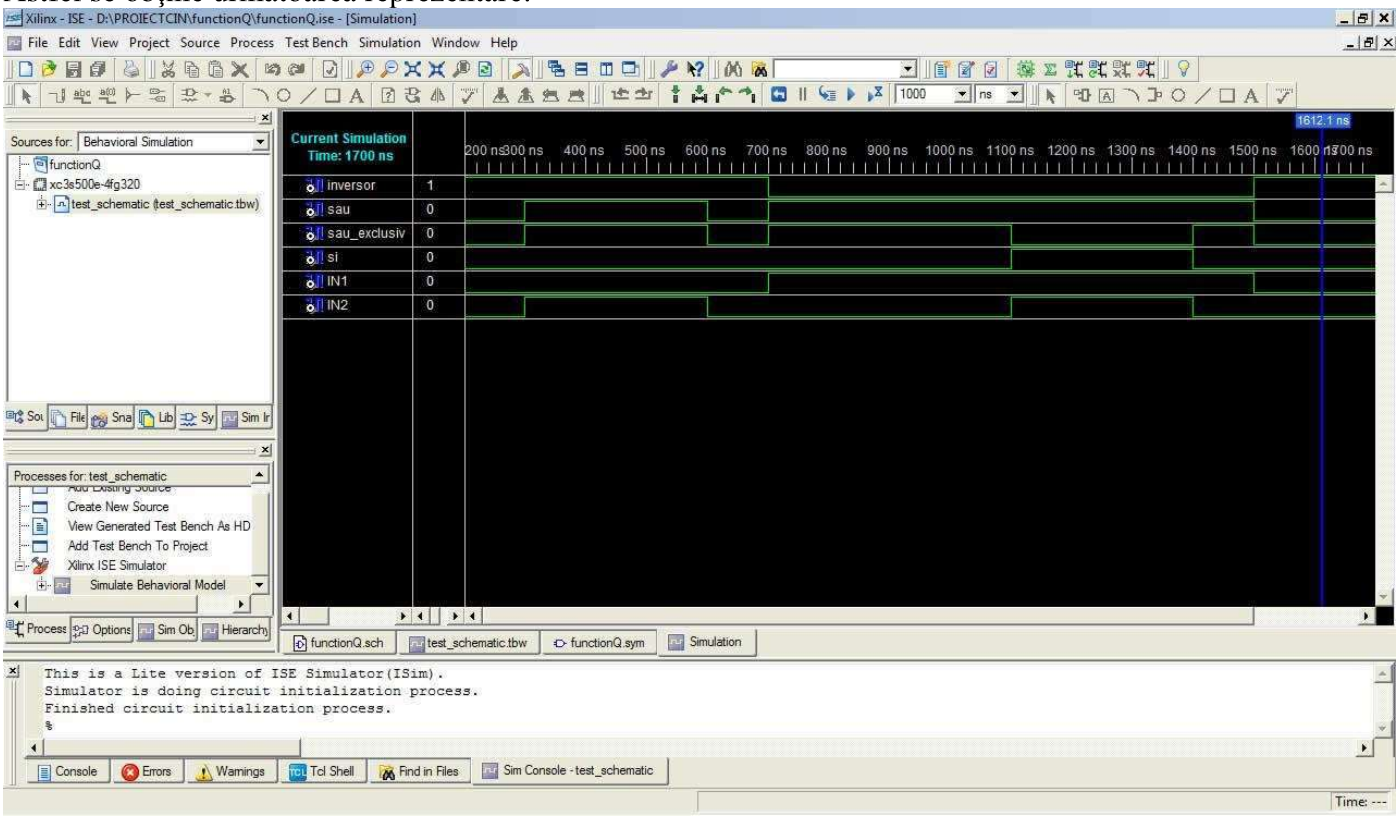


Figura 1.16. Rezultatul simularii

2.12 Implementarea în cod VHDL

În cadrul aceluiaşi proiect, se adaugă o noua sursă de tip *VHDL Module* care se numeşte *functionQ_vhdl*, apoi se introduc variabilele de intrare IN1 şi IN2 precum şi ieşirile inversor, sau, sau_exclusiv, şi.

Se completează codul generat astfel încât să ajungem la următorul cod:

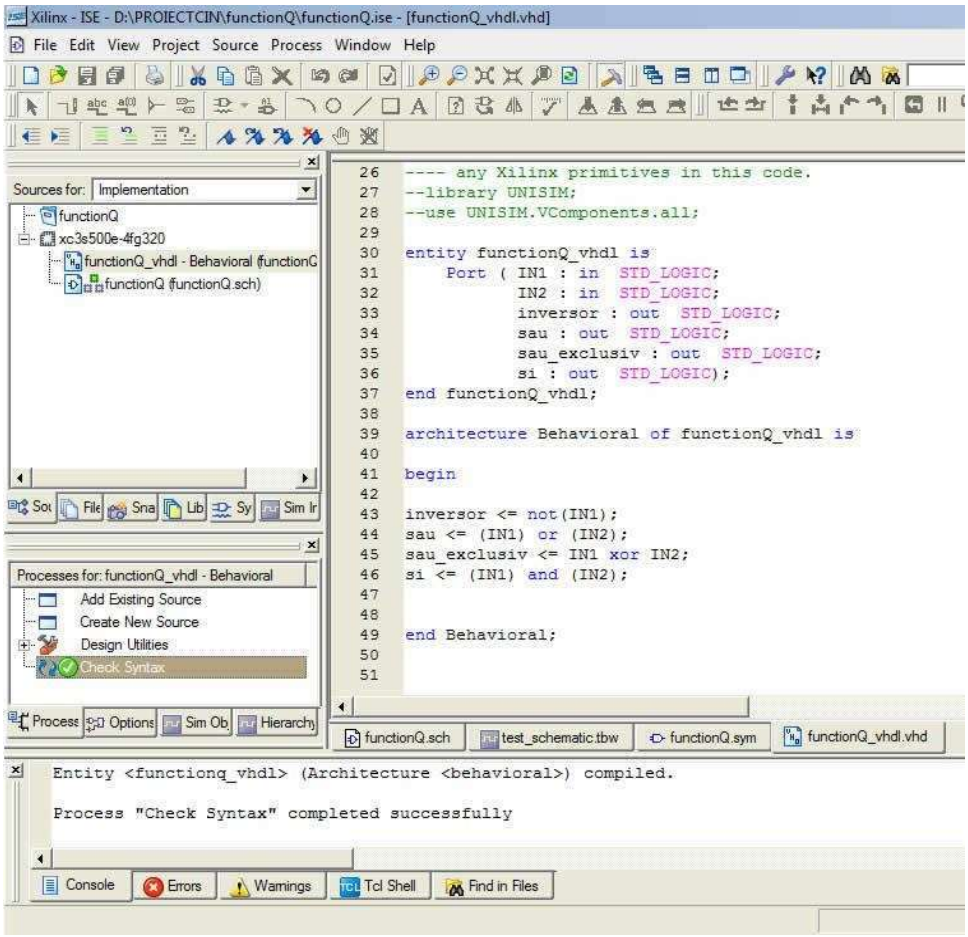


Figura 1.16. Codul VHDL

Pentru testarea codului VHDL, se adaugă o noua sursă de tip *Test Bench Waveform* în mod similar şi cu aceleaşi modificări făcute ca şi la fişierul de test pentru schematic. Valorile pentru datele de intrare se setează exact ca la testul anterior.

În urma executării noului fişier de test, trebuie sa obţinem o simulare identică cu cea de la schematic.

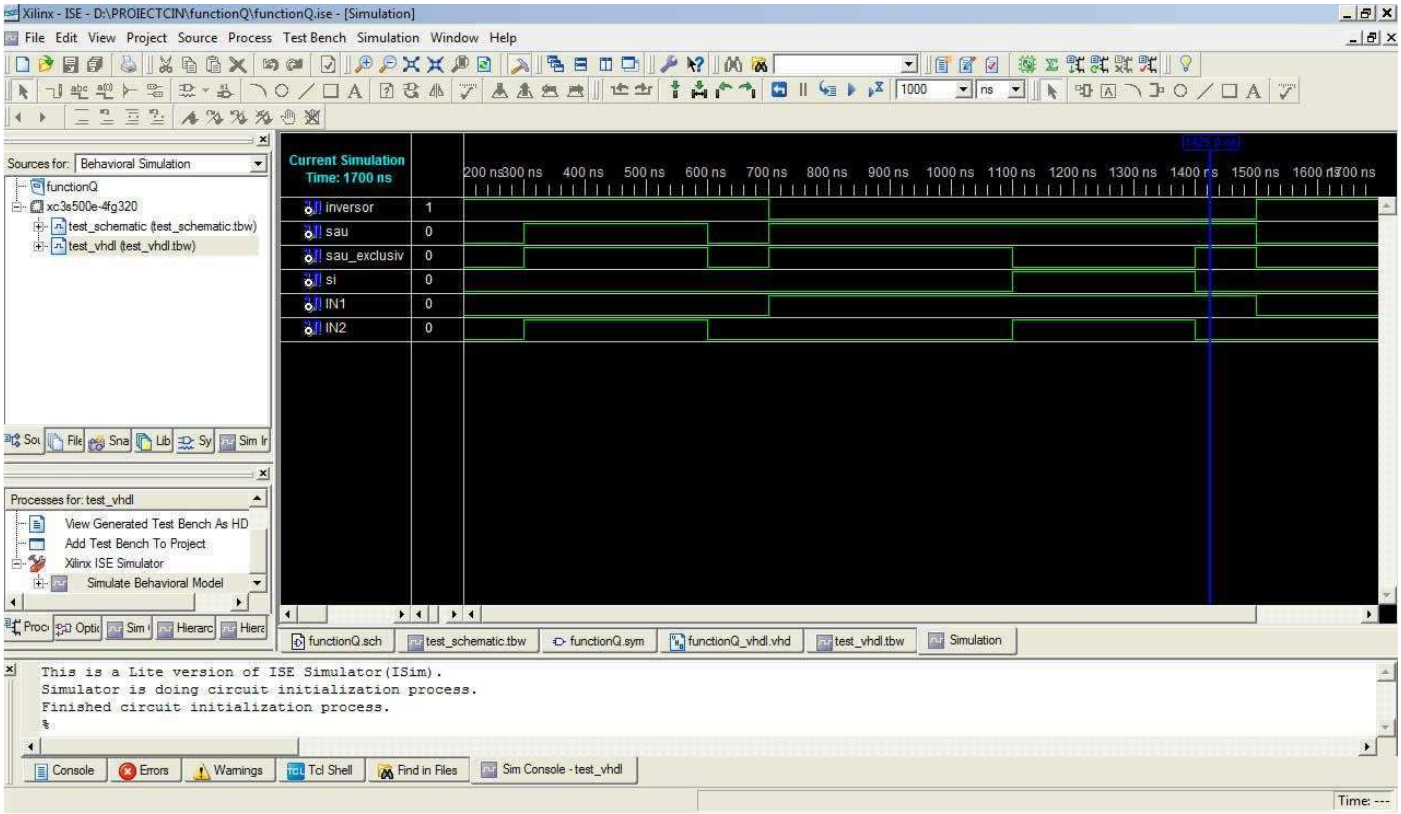


Figura 1.17. Rezultatul simulării codului VHDL

Se verifică schema, având în vedere modul de operare al porţilor logice din punct de vedere teoretic, prezentat în tabelul de mai jos:

Operator	Formulă
AND	Adevărat (1) când toate intrările sunt adevărate(1)
OR	Adevărat (1) când cel puţin o intrare este adevărată (1)
XOR	Adevărat (1) dacă şi numai dacă o intrare, dar nu amândouă, are valoare de adevăr(1)
NOT	Adevărat (1) când intrarea este falsă (0)

Aşadar, conform schemei :

	200-300ns	300-600ns	600-700ns	700-1100ns	1100-1400ns
IN1	0	0	0	1	1
IN2	0	1	0	0	1
Inversor(NOT) pt IN1	1	1	1	0	0
Sau (OR)	0	1	0	1	1
Sau_exclusiv (XOR)	0	1	0	1	0
Şi (AND)	0	0	0	0	1

3.Cosimulare

Cosimularea este o metodologie de simulare care permite unor componente individuale să fie simulate de către diferite platforme simultan care interschimbă informaţii între ele.

Pentru procesul de CoSimulare e nevoie de următoarele platforme instalate:

1.)Matlab Simulink

2.)ModelSim

Se deschide un nou model de tip Simulink.

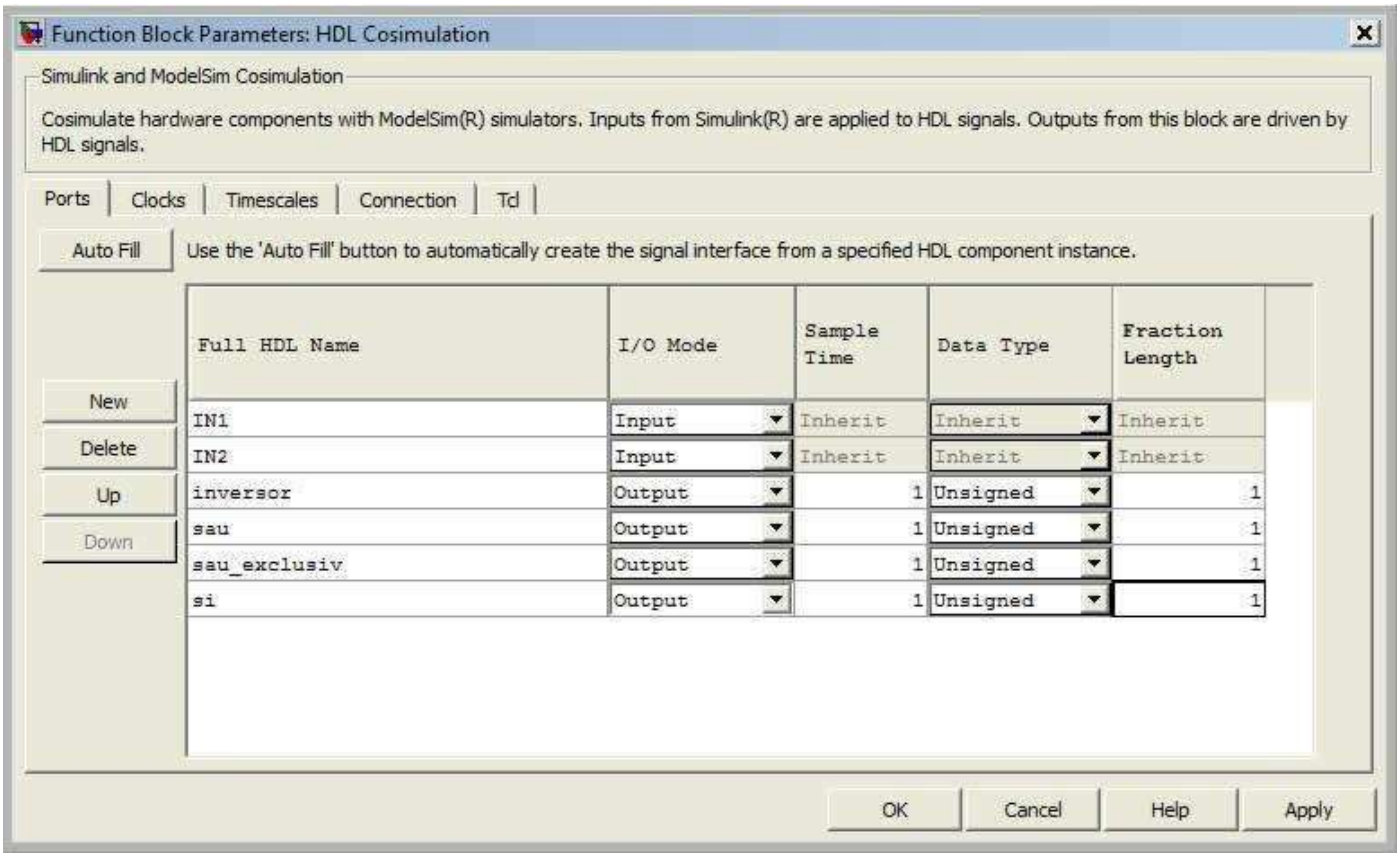
În modelul Simulink curent se fac următoarele setări:

-În *Simulation -> Configuration parameters* se setează Fixed Step(auto) iar Stop Time se setează la 20(secunde).

-Din Libraria EDA Simulator Link MQ se inserează blocul HDL Cosimulation

Se da dublu click pe blocul inserat în model şi se setează urmatorii parametrii:

Din meniul Ports se setează intrările şi ieşirile astfel încât acestea să coincidă cu numele din codul VHDL.

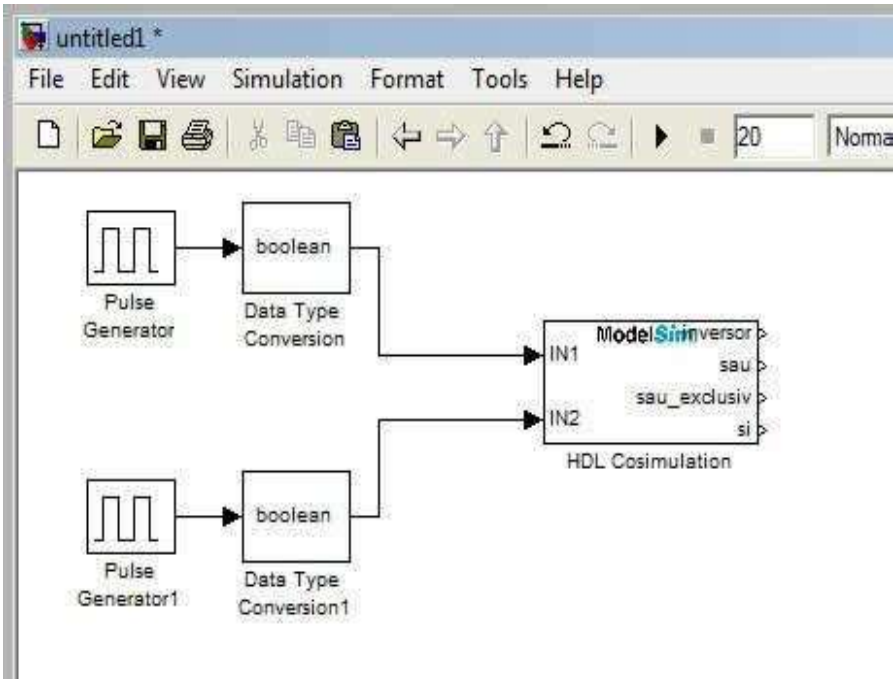


Din Biblioteca *Simulink ->Sources* se inserează 2 blocuri de tip Pulse Generator.Aceste blocuri vor avea următoarele setări

Blocul 1:Amplitudine 1,Perioada(Period) 10,Lungimea Pulsului 10%,Defazaj(Phase delay) 0 secunde

Blocul 2: Amplitudine 1,Perioada(Period) 10,Lungimea Pulsului 90%,Defazaj(Phase delay) 0 secunde

După ce se efectuează aceste setări se introduc 2 blocuri de conversie în date booleene(Blocul *Data Type Conversion* din Libraria *Simulink->Commonly Used Blocks* care se seteată ca Boolean).



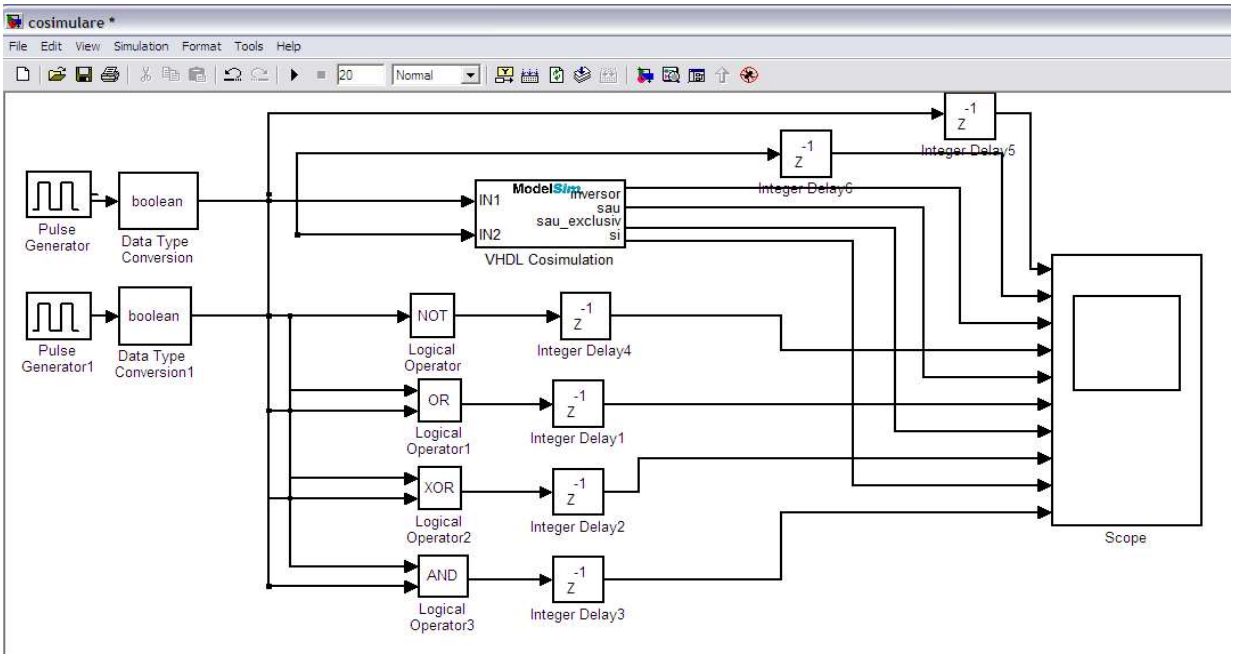
Ulterior, se introduce modelul Simulink al porţilor logice în noul model Simulink însă se şterg intrările de tip Signal Generator + Constant şi se leagă la Pulse Generator convertit la boolean.

Semnalul care iese din modelul creat în Simulink trebuie întârziat cu o unitate de timp pentru a fii în fază cu semnalul care iese din Blocul HDL Simulation (semnalul din acest bloc ieşind întârziat cu o unitate de timp adică 1 nanosecunda care în Simulink înseamnă 1 secundă).

Se introduc 4 blocuri de *Integer Delay* din biblioteca *Simulink ->Discrete* cu următoarele setări (Iniţial Condition 0,Sample time -1,Number of Delays 1)

Acest bloc are rolul de a întârzia semnalul cu o unitate de timp si se aplică ieşirilor modelului simulink.

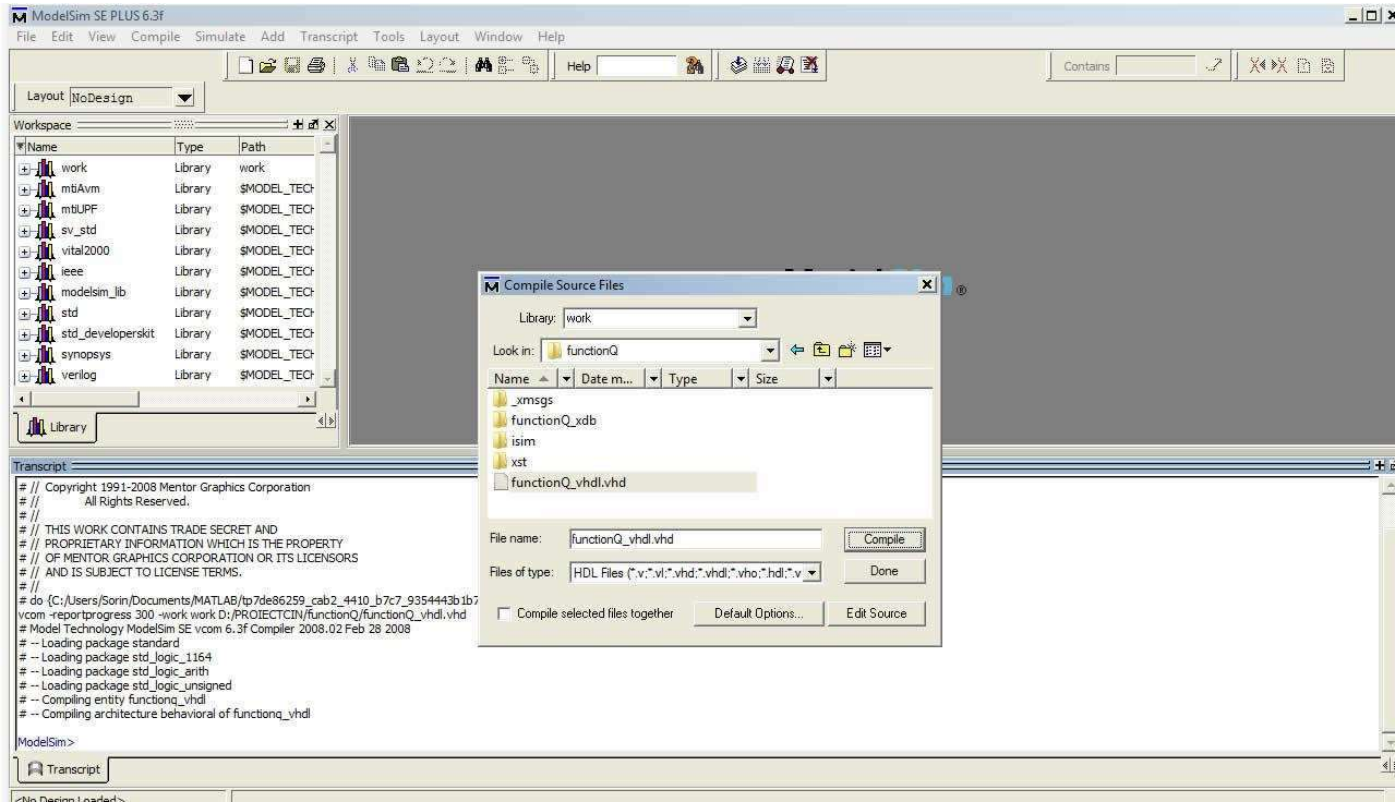
Din biblioteca *Simulink>Sinks* se introduce un bloc *Scope* cu 10 intrări(se debifează Limit Data Points To 5000) .



Primele 2 intrări de la Scope sunt Semnalele de intrare iar apoi concomitent prima ieşire de la ModelSim cu prima de la modelul în simulink până la ultimele ieşiri. Pentru vizualizarea rezultatului mai trebuiesc urmaţi câţiva paşi.

Din Matlab Workspace se tastează comanda vsim() pentru a se deschide o instanţă a programului ModelSim.

Se compileaza codul VHDL dorit (se alege fişierul sursă în care avem codul VHDL, anume functionq_vhdl.vhd). Se apasă Butonul Compile pentru a compila acest cod apoi se apasă Done pentru a încheia operaţiunea.



Urmatorul pas presupune simulare codului VHDL compilat anterior.

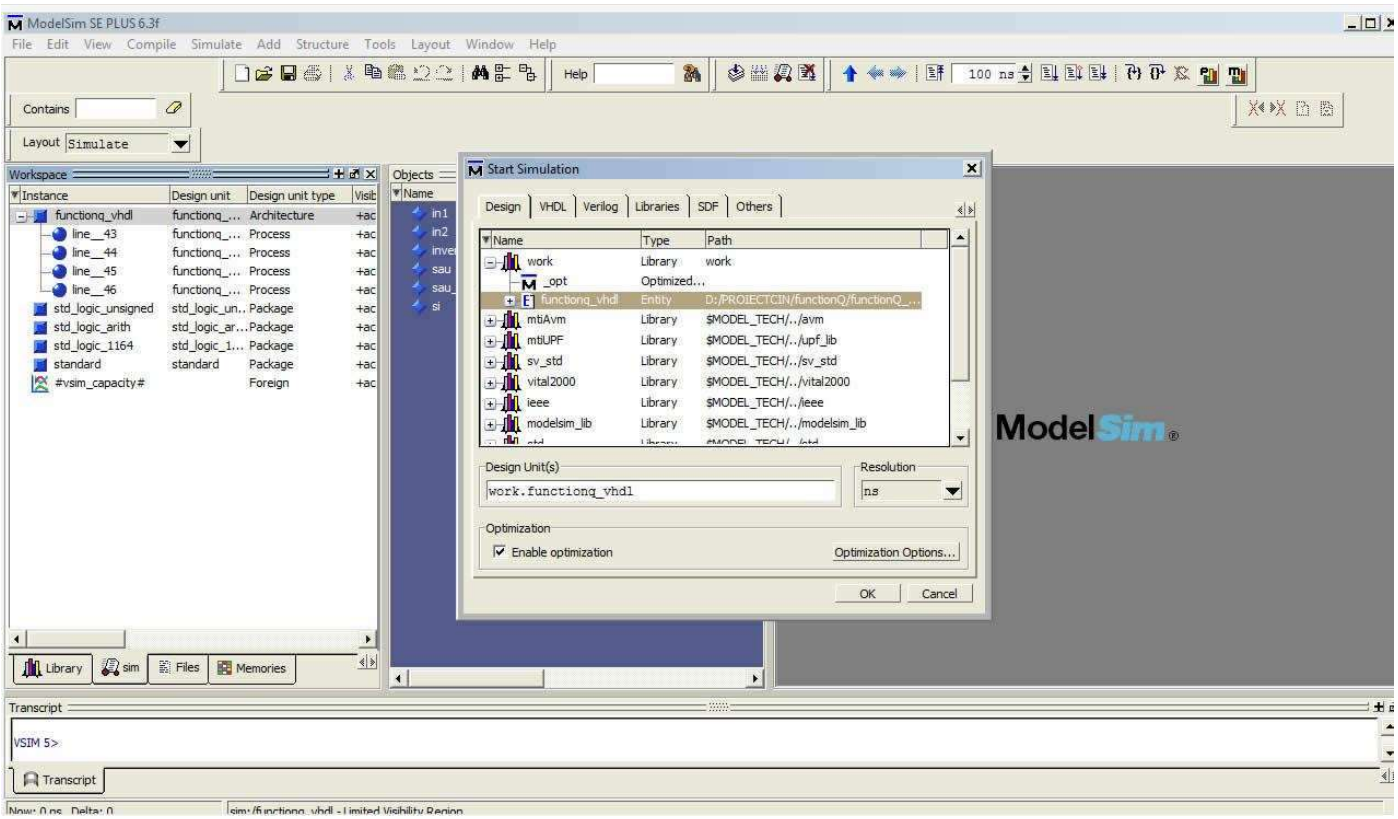
Astfel, din butonul Simulate al meniului principal se selectează Start Simulation... Astfel se deschide o noua instantă a ferestrei Start Simulation.

În urma compilării, fişierul VHDL a fost inserat în folderul „work”.

În fereastra Start Simulation se expandează folderul „work” şi se selectează functionq_vhdl.

În subfereastra Resolution se setează timpul de simulare în nanosecunde (ns adică o secunda din simularea în simulink este echivalenta cu o ns în modelsim).

Se apasă butonul „OK” pentru a încheia operaţiunea.



În fereastra de transcript a instanţei ModelSim curente se introduc următoarele comenzi:

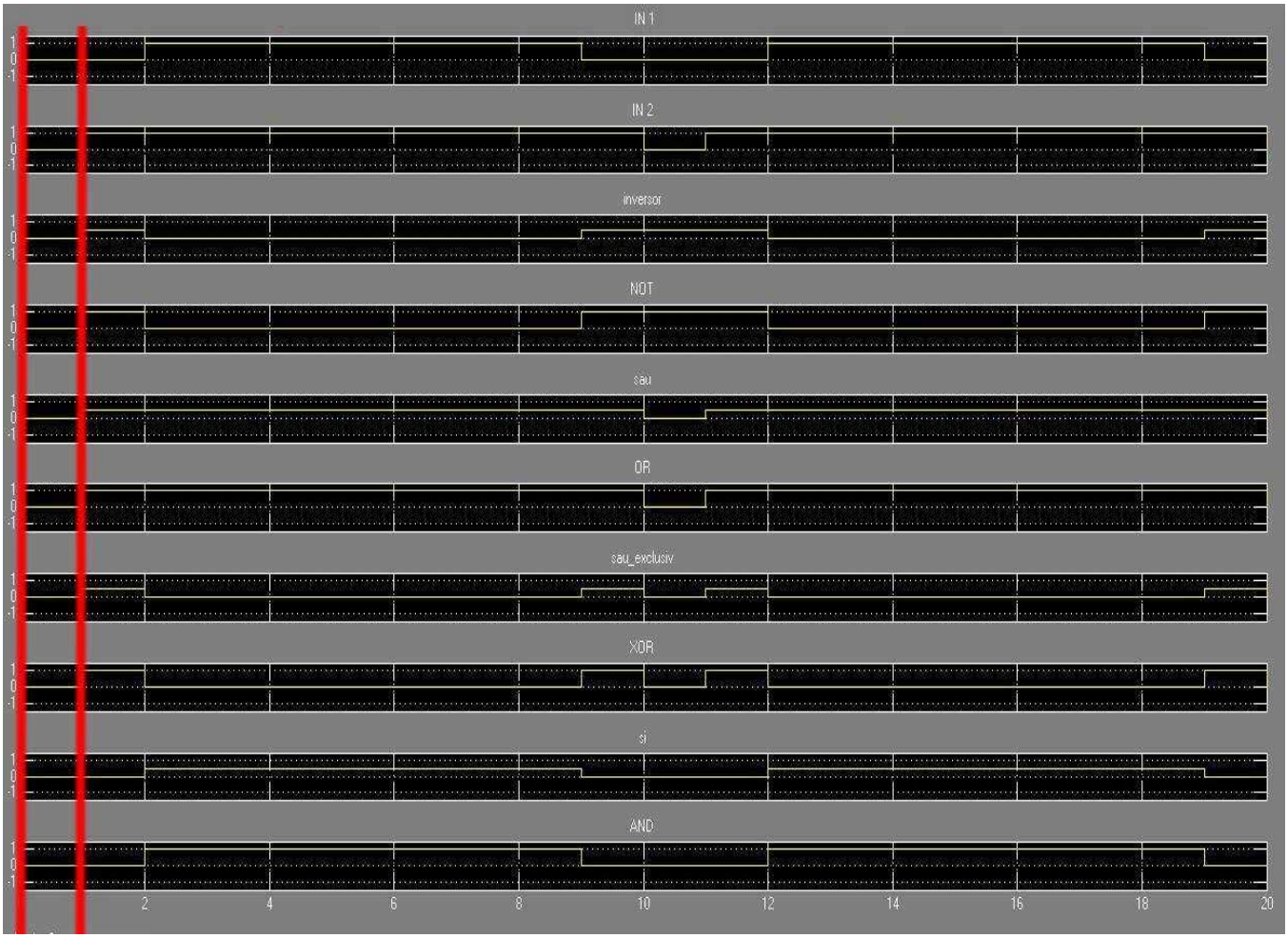
>>vlib work

>>vmap work work

>>vsimulink functionq_vhdl

Funcţia vlib work se apelează pentru linkarea între sistemul de librării şi fisierul work în care este salvat fişierul *.vhd adăugat anterior.

Se revine la modelul simulink pentru vizualizarea rezultatului.

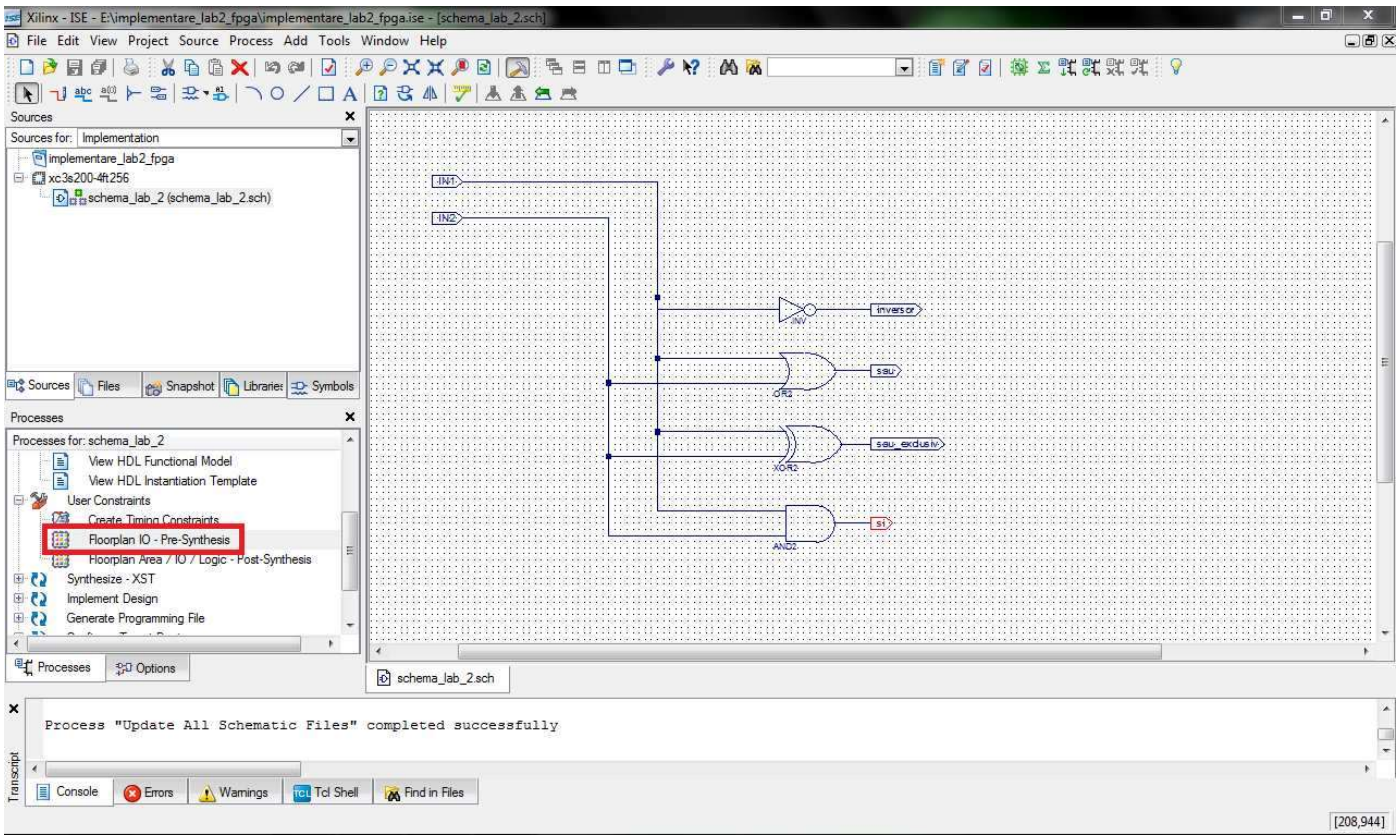


Observaţie:

Timpul de corelare reprezintă perioada de timp în care Simulink aşteaptă raspunsul de la ModelSim . Deoarece cele doua simulatoare nu pot lucra simultan exista acest interval de timp , unul dintre ele aşteptându-l pe celălalt. Deci prorpiu-zis cosimularea începe de la secunda 1.

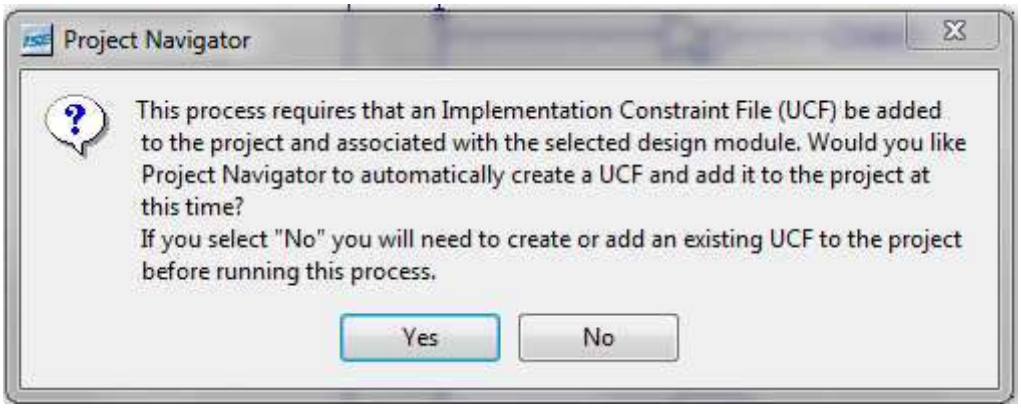
4.Testarea pe placa FPGA

Pentru a testa schema pe placă, fiecare terminal de intrare şi de ieşire se asociază cu câte un pin al circuitului FPGA.



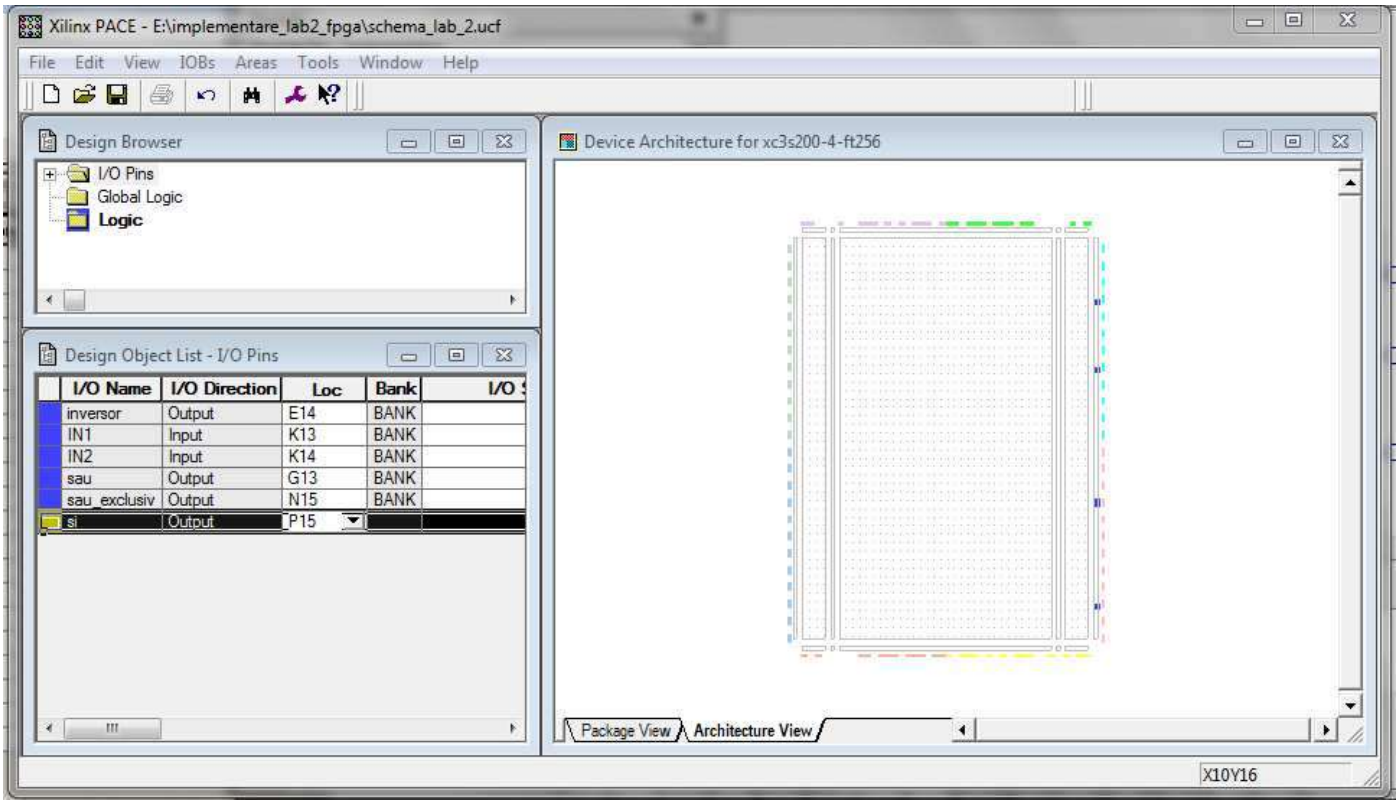
Pentru acest lucru, se dă click în meniul Processes -> User Constraints -> Pre – Synthesis.

Se va primi o alertă ca în figura de mai jos:

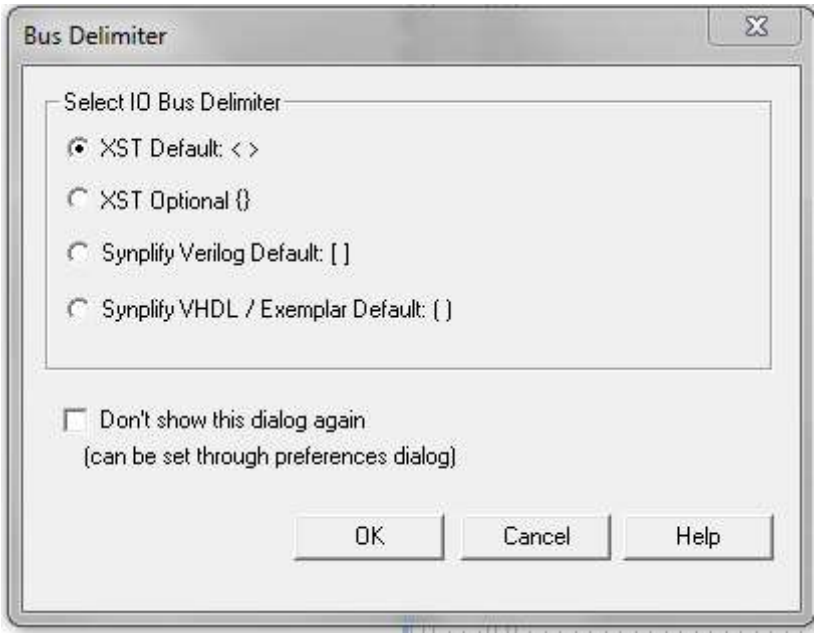


Aceasta face referire la existenţa fişierului UCF aferent schemei selectate. Se alege opţiunea YES pentru a asocial pinii cu ajutorul utilitarului Pace.

În urmatoarea fereastă se face asocierea efectivă a pinilor, astfel încât cele 2 switch-uri (etichetate pe placă K13 şi K14) să fie de tip ‘input’ şi cele 4 ieşiri să fie aferente celor 4 LED-uri (E14, G13, N15, P15).

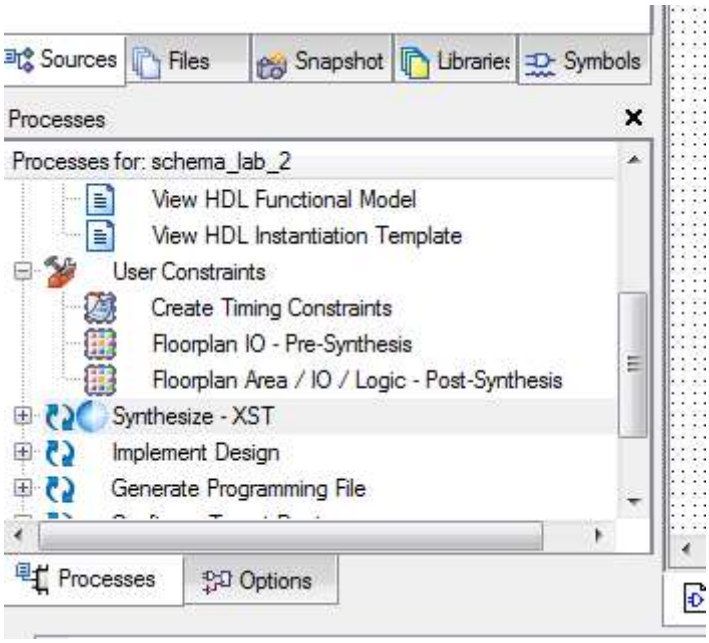


Se salvează şi va apărea urmatoarea fereastră:

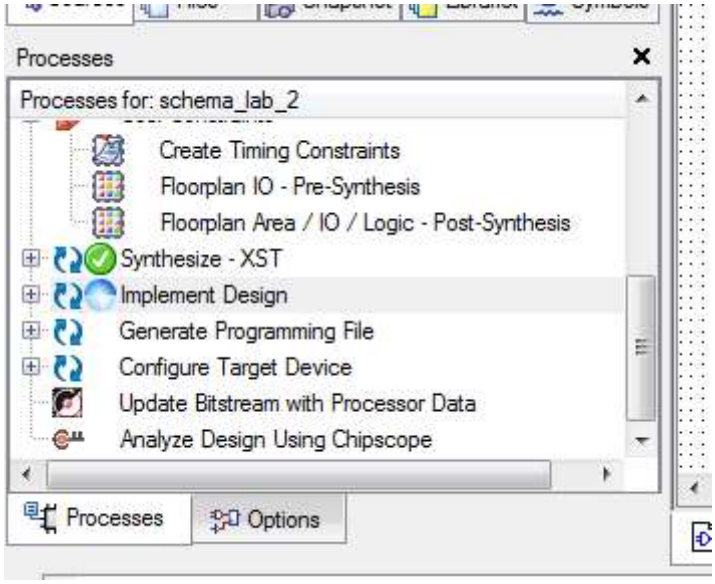


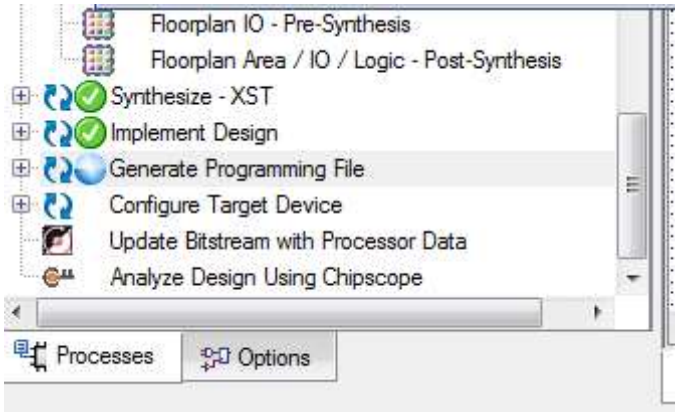
Nu se modifica nimic, ci doar se apasă OK.

Ulterior se selectează opţiunea Synthesize prin dublu – click pe aceasta.

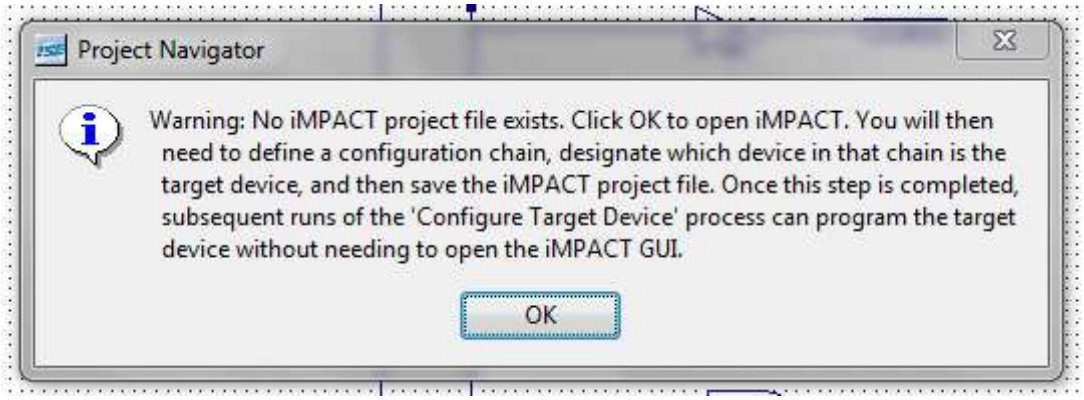


Prin aceeaşi metoda se selectează şi opţiunea Implement Design şi Generate Programming File.

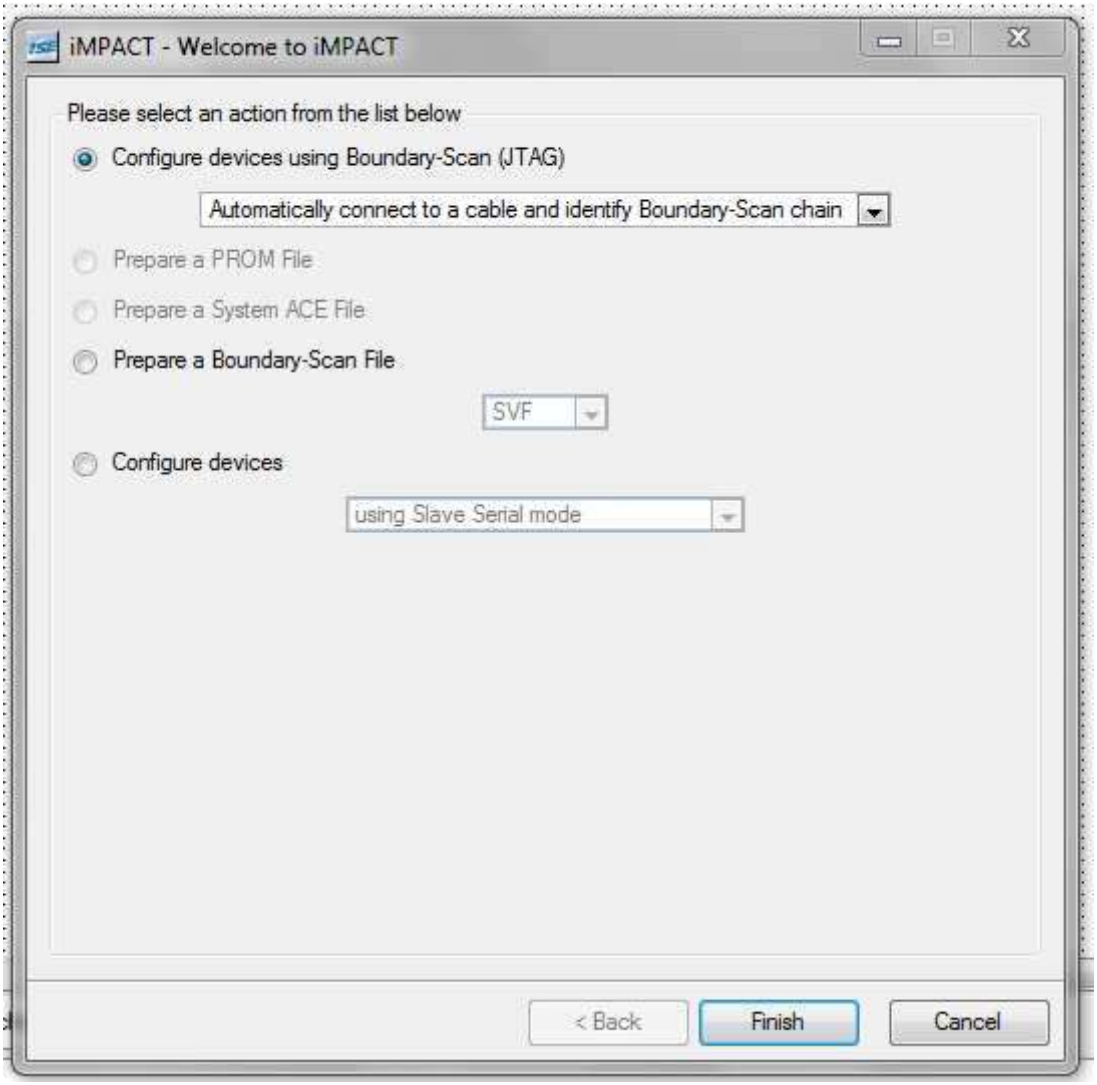




În urma generării va apărea urmatorul warning:



Se apasă Ok, dupa care va apărea fereastra de detectare a dispozitivului FPGA.



Se apasa Finish.

4.2 Încărcarea pe placă a fişierului bitstream (.bit)

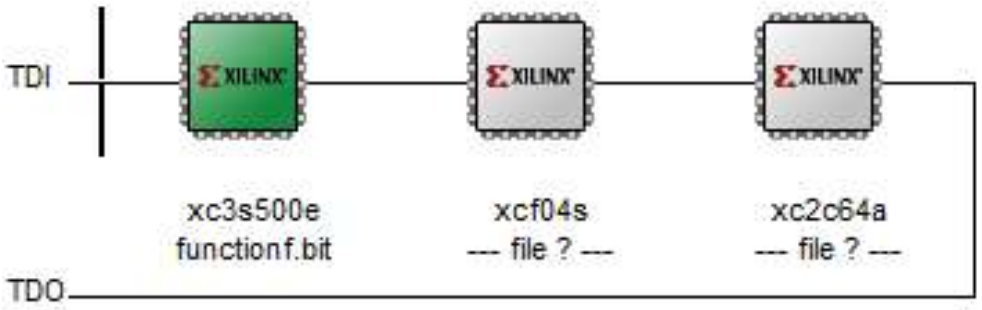
Pentru a încărca pe placă fişierul bitstream, se expandează în subfereastra *Processes* Configure Target Device şi se efectuează dublu click pe Manage Configuration Project(iMPACT).

Circuitul FPGA este programat printr-o interfaţă JTAG. Pe machetă există trei circuite cu interfaţă JTAG, conectate în serie:

- xc3s500e circuitul FPGA (folosit pentru studiu);
- xcf04s circuitul EPROM (conţine programul pentru aplicaţia implicită pentru FPGA, utilizată la testarea initiala a machetei).

• xc2c64a circuitul CPLD (conţine modulul hardware pentru programarea circuitului FPGA cu conţinutul din memoria EEPROM, la punerea sub tensiune a machetei);

După depistarea automată a tipului de cablu de conectare dintre macheta FPGA şi calculator, va apărea o reprezentare grafică a lanţului celor trei circuite, aşa ca în următoarea figură:



Reprezentarea grafică a lanţului JTAG cu cele trei componente: FPGA, CPLD, EEPROM.

Fiecare din cele trei circuite pot fi programate prin interfaţa JTAG şi are asociat câte un fişier de programare. În prima fereastră ce apare se efectuează click pe *Finish*. După aceea va trebui să se selecteze fişierul ce va fi încărcat pe circuitul FPGA (primul în lanţ) şi sa se efectueze click pe Open. La următoarele două modele de circuite nu se selectează nici un fel de fişier şi se efectuează click pe *Bzpass*. În ultimă fază, se efectuează click dreapta pe circuitul FPGA xc3s500e şi click Program.