Lucrarea 4

Metodologia XILINX-ISE. Descrierea schematică. Porţi logice

4.1 Obiective

Lucrarea are următoarele obiective:

- Familiarizarea cu mediul XILINX WebPack ISE;
- Utilizarea editorului Schematics pentru crearea de scheme logice.

4.2 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 implementează funcţia:

$$F = AB + CD \tag{4.1}$$

4.2.1 Pornirea ISE

Pentru a lansa *ISE WebPack* daţi dublu clic pe iconiţa asociată programului sau apelaţi 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 împărţită în patru subferestre:

- **sub-fereastra surselor (source panel)** în care sunt afişate toate fişierele asociate proiectului curent;

- **sub-fereastra proceselor (process panel)** în care se află listate 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.

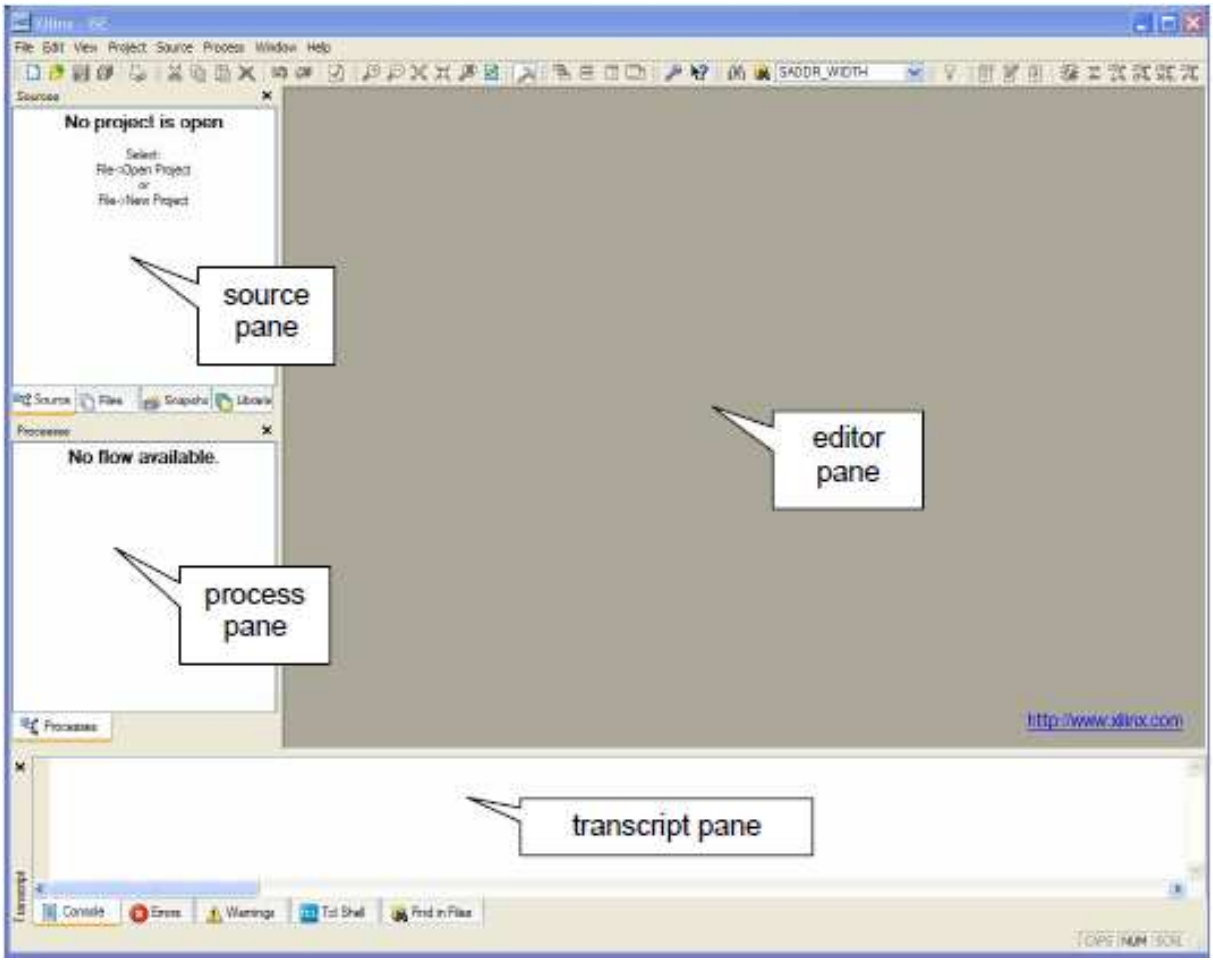

Figura 4.1 Fereastra principală XILINX ISE Webpack.

4.2.2 Crearea unui proiect nou

Pentru a crea un proiect nou apelaţi meniul:

File —> New Project

Completaţi câmpurile cu numele proiectului, locaţia, iar în câmpul *Top-level source* selectaţi **Schematics**. Apoi clic *Next*. Selectaţi (şi reţineţi) directorul în care se va crea proiectul. Denumiti proiectul **functionF**. Proiectul constă dintr-un set de fişiere ce se vor crea în directorul selectat.

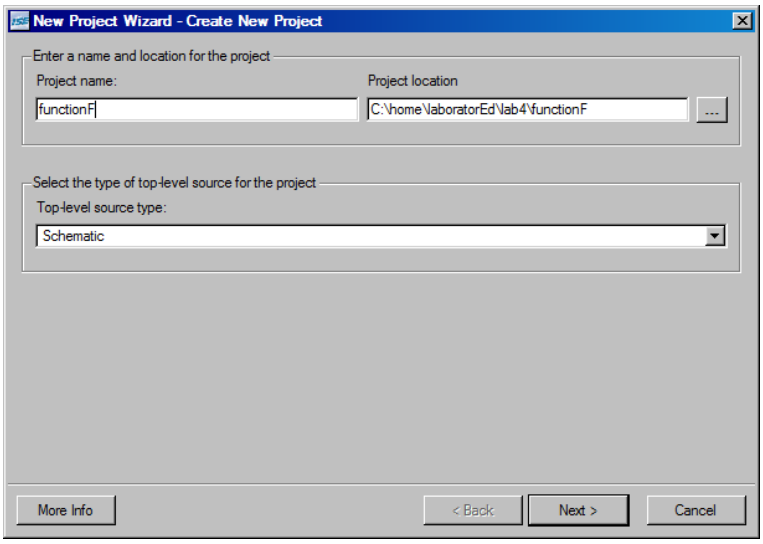

Figura 4.2 Fereastra *New Project Wizard Create New Project*.

În fereastra de descriere a dispozitivului, prezentată în figura 4.3, completaţi câmpurile ce descriu circuitul XILINX utilizat. Apoi clic *Next*.

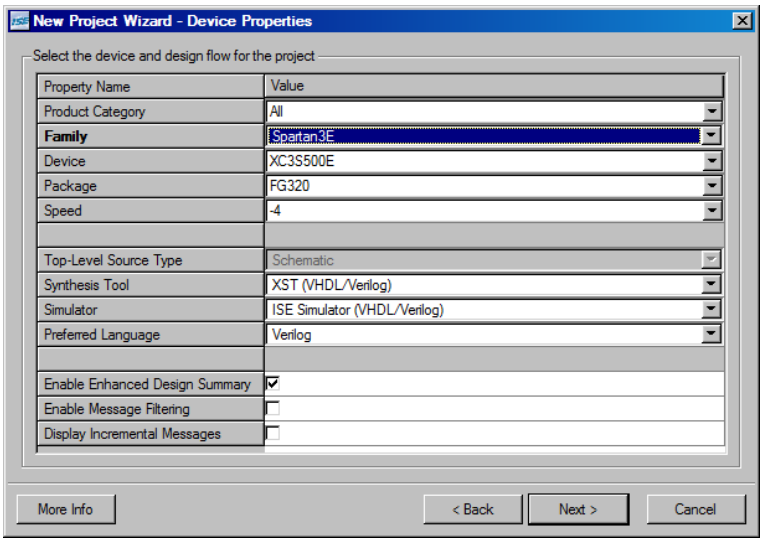

Figura 4.3 Fereastra *New Project Wizard Device Properties*.

Placa utilizată conţine un dispozitiv FPGA cu următoarele caracteristici:

Device Family: Spartan3E
Device: XC3S500E
Package: FG320
Speed Grade: -4

Următoarele două ferestre care vor apărea, *New Project Wizard Create New Source* şi *New Project Wizard Add Existing Sources* sunt prezentate în figurile 4.4 şi 4.5. 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...

Amânaţi includerea fişierelor în proiect şi faceţi clic pe *Next* în cele două ferestre.

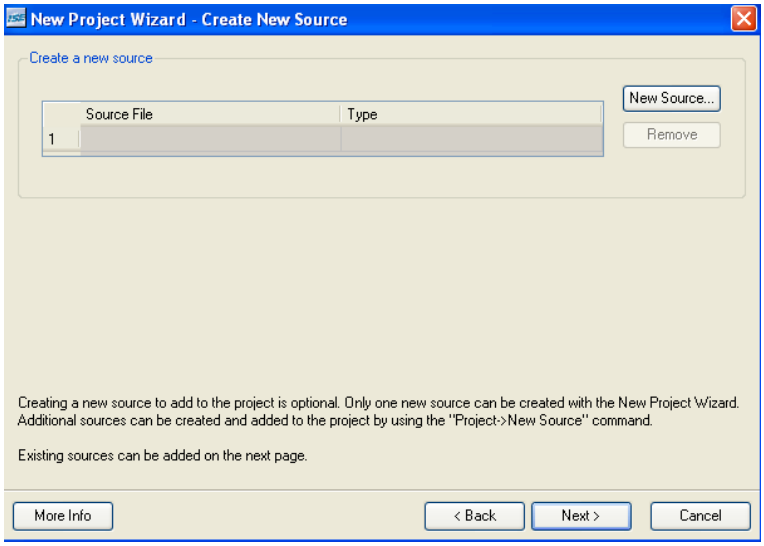

Figura 4.4 Fereastra *New Project Wizard Create New Source.*

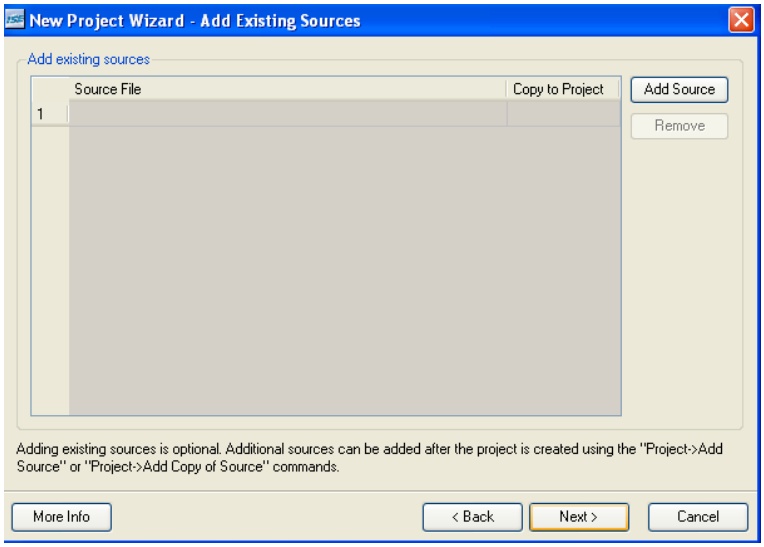

Figura 4.5 Fereastra *New Project Wizard Add Existing Sources.*

În ultima fereastră care va apare, *New Project Wizard Project Summary*, veţi găsi un sumar cu toate datele referitoare la proiect şi la modelul folosit. Clic *Finish.*

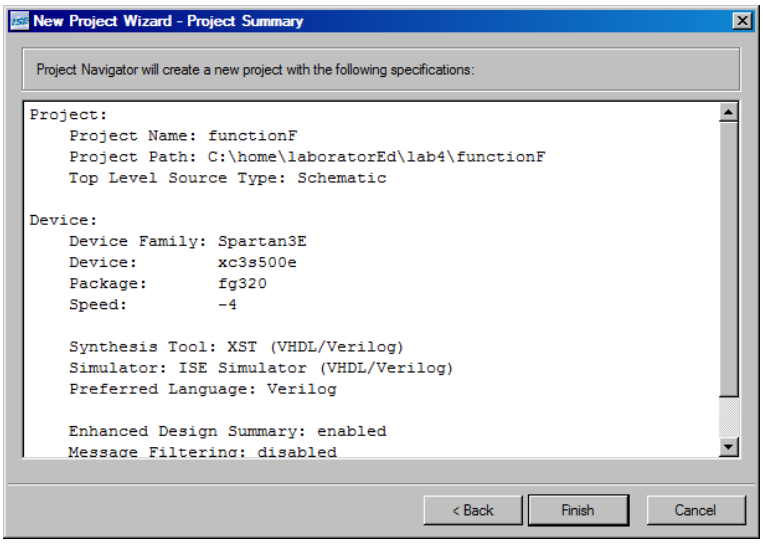

Figura 4.6 Fereastra *New Project Wizard Project Summary*.

4.2.3 Crearea unui fişier pentru o schemă (.sch)

Pentru a crea un fişier sursă schemă, selectaţi în sub-fereastra surselor *Sources* modelul **xc3s500e-4fg320**, iar în sub-fereastra proceselor *Processes* daţi dublu clic pe *Create New Source*.

O alternativă constă în apelarea meniului:

Project —> New Source...

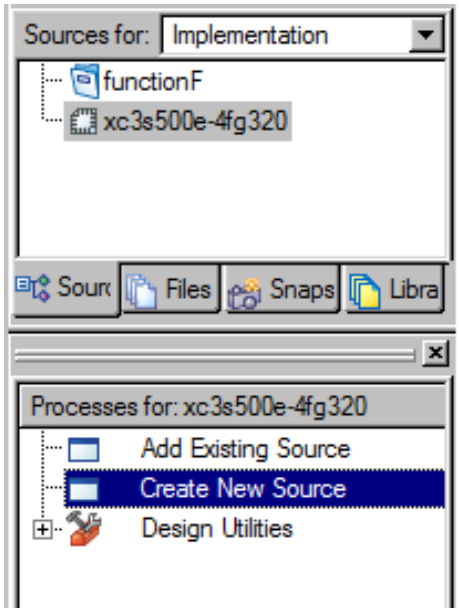

Figura 4.7 Sub-fereastra *Sources* şi sub-fereastra *Processes*.

În fereastra *New Source Wizard Select Source Type* selectaţi **Schematic** şi completaţi numele fişierului sursă ce se va genera şi locaţia acestuia în sistemul de fişiere al calculatorului. Clic *Next* şi *Finish*. Denumiţi fişierul **functionF**. Fişierul salvat va avea extensia **.sch**. Asiguraţi-vă ca aţi selectat opţiunea **Add to project**.

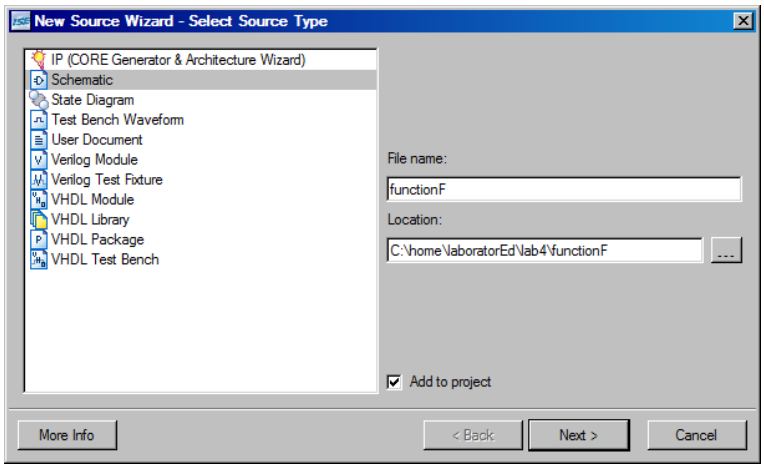

Figura 4.8 Fereastra *New Source Wizard Select Source Type*.

4.2.4 Adăugarea componentelor

Schema de porţi logice asociată acestei funcţii este prezentată în figura 4.9.

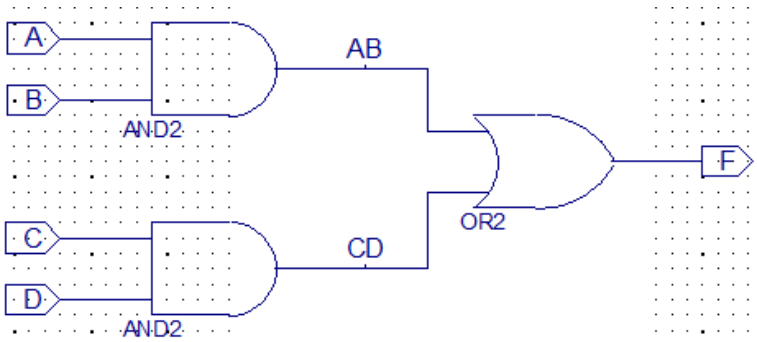

Figura 4.9 Schema de porţi logice asociată funcţiei *F*.

În partea stângă în fereastra *Sources* se găseşte lista cu simboluri şi cea cu categorii de simboluri. Pentru a găsi cu uşurinţă simbolurile necesare (aici porţi logice) selectaţi la *Categories* **Logic**. În lista *Symbols* se vor afişa doar simbolurile de la porţi logice. Dacă doriţi să căutaţi un simbol după denumire, scrieţi primele litere din denumirea simbolului în *Symbol Name Filter*, realizându-se un filtru, iar în lista *Symbols* vor apărea doar simbolurile a căror denumire începe cu acele litere.

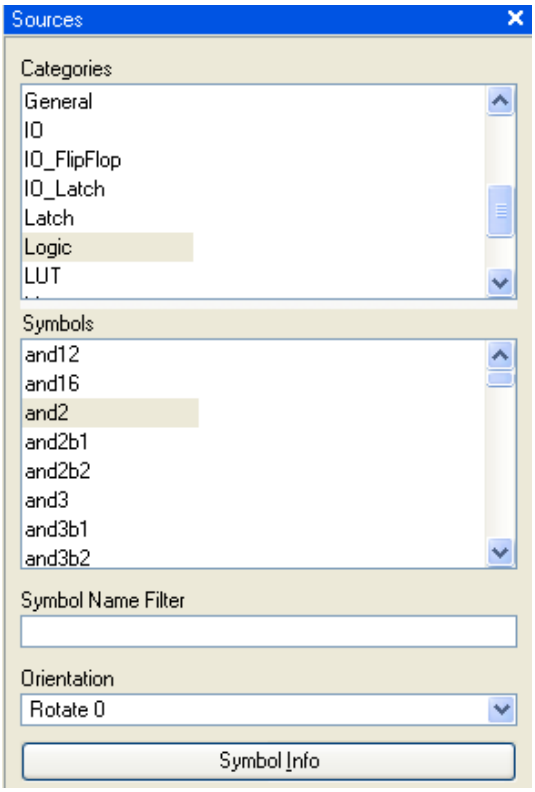

Figura 4.10 Fereastra simbolurilor.

Selectaţi simbolul dorit. Pentru a-l plasa în cadrul schemei, faceţi clic în locul în care doriţi a fi pus. Plasaţi astfel toate simbolurile schemei.

4.2.5 Editarea de conexiuni

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

Pentru a realiza o conexiune apelaţi comanda:

Add –> Wire (CTRL+W)

sau

clic pe iconiţa *Add Wire*  .

Pentru a plasa o conexiune, faceţi clic pe primul pin şi dublu-clic pe cel de al doilea pin.

Unui fir i se poate asocia o denumire utilizând comanda:

Add –> Net Name (CTRL+D)

sau

clic pe iconiţa *Add Net Name*  .

Realizaţi conexiunile dintre porţi, ca în fugura 4.9. Etichetaţi firele cu numele precizate.

4.2.6 Adăugarea terminalelor de intrare şi de ieşire

Pentru a adăuga terminale de intrare şi de ieşire utilizaţi comanda:

Add –> I/O Marker (CTRL+G)

sau

clic pe iconiţa *Add I/O Marker*  .

Plasaţi porturile pe schemă, aşa ca în figura 4.9. Modificaţi (prin redenumire) numele implicite ale porturilor cu numele recomandate în specificaţii (figura 4.9). Pentru a redenumi un terminal, daţi clic dreapta pe el şi selectaţi *Rename port*.

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

4.2.7 Verificarea schemei

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

Tools –> Check Schematic

sau

clic 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:

```
Start DRC ...
No error or warning is detected
```

4.2.8 Introducerea de elemente grafice şi text

Pe lângă elementele efective ale unei scheme pot fi inserate şi elemente grafice şi text cum ar fi: blocuri pentru titlul schemei, cercuri, arcuri de cerc, linii, dreptunghiuri sau text.

Pentru a introduce în schemă un bloc pentru titlu se procedează la fel ca la introducerea unei componente. Selectaţi în fereastra *Sources* la *Categories* **General**, iar la *Symbols* selectaţi componenta **title**. După ce plasaţi blocul în pagină, pentru a putea edita câmpurile, daţi dublu clic pe el.

În fereastra *Object Properties*, figura 4.11, puteţi modifica numele, titlul, puteţi adăuga şi alte câmpuri (de exemplu informaţii suplimentare referitoare la schemă) sau puteţi modifica diferite opţiuni.

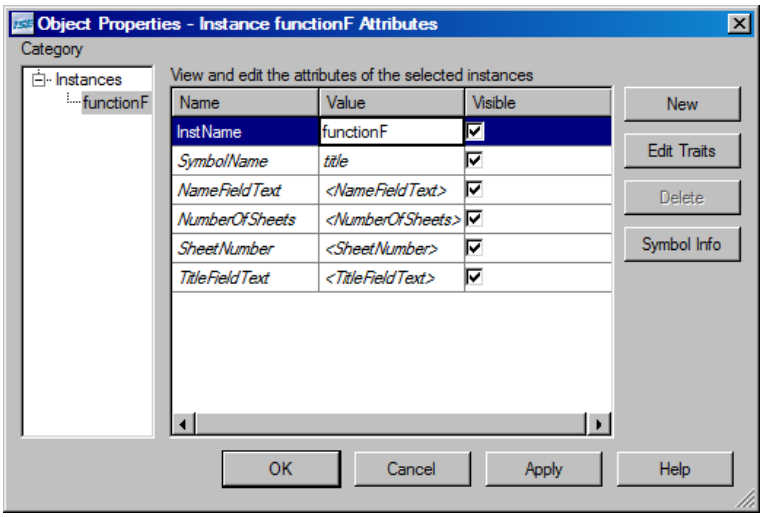

Figura 4.11 Fereastra *Object Properties*.

Alte elemente grafice:

Dreptunghiuri: clic *Add* –> *Rectangle* sau clic pe butonul *Add Rectangle*;
Cercuri: clic *Add* –> *Circle* sau clic pe butonul *Add Circle*;
Linii: clic *Add* –> *Line (CTRL+L)* sau clic pe butonul *Add Line*;
Arcuri: clic *Add* –> *Arc* sau clic pe butonul *Add Arc*.

4.2.9 Salvarea schemei

Pentru a salva schema apelaţi meniul:

File –> *Save (CTRL+S)*

sau

clic *Save*.

Salvaţi schema în fişierul **functionF.sch**.

4.2.10 Crearea simbolurilor şi utilizarea lor

Pentru a vă uşura munca pe viitor, vă puteţi realiza propriile voastre simboluri sau puteţi modifica alte simboluri existente în bibliotecă. Noile simboluri create vor fi plasate în biblioteca locală de simboluri. Pe viitor, când veţi avea nevoie să utilizaţi din nou această schemă o veţi putea folosi direct ca un simbol.

Pentru a realiza un simbol nou clic *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 schemă. În cazul în care selectaţi *Specify manually* toate porturile vor trebui introduse manual.

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

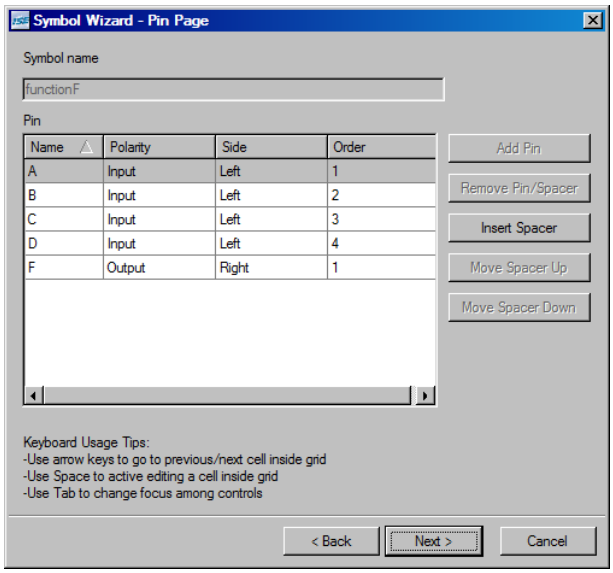

Figura 4.12 Fereastra *Symbol Wizard Pin Page*.

În ultima fază, simbolul trebuie salvat într-un fişier cu extensia **.sym**. Numele simbolului va fi numele fişierului. Pentru a folosi simbolul în alt proiect, copiaţi în directorul noului proiect fişierul **.sym** şi fişierul **.sch** asociat (schema internă a simbolului).

4.3 Testarea pe placă

Pentru a testa schema pe placă, fiecare terminal de intrare şi de ieşire se asociază cu câte un pin al circuitului FPGA. De exemplu, intrările se conectează la switch-uri, iar ieşirea se conectează la un led. Acest lucru se realizează prin existenţa unui fişier cu extensia **.ucf** (*User Constraints File*) ce conţine o asociere între porturile de intrare/ieşire şi pinii circuitului FPGA.

Sintaxa asocierii dintre numele unui port şi un pin FPGA este:

```
NET "numePort"  LOC = "locatiePin";   # comentariu
```

Fişierul de constrângeri **functionF.ucf** arată astfel:

```
# LED
NET "F"  LOC = "F12";  # LED0

# SWITCH
NET "A"  LOC = "L13";  # SW0
NET "B"  LOC = "L14";  # SW1
NET "C"  LOC = "H18";  # SW2
NET "D"  LOC = "N17";  # SW3
```

Cu un editor de texte simple (**Notepad**), editaţi conţinutul fişierului **functionF.ucf**. Pentru a adăuga fişierul în proiect, daţi clic dreapta pe model, apelaţi *Add Source* şi selectaţi fişierul **functionF.ucf**.

Vizualizaţi fişierele din proiect în sub-fereastra *Sources*. Fereastra este similară cu cea prezentată în figura 4.13.

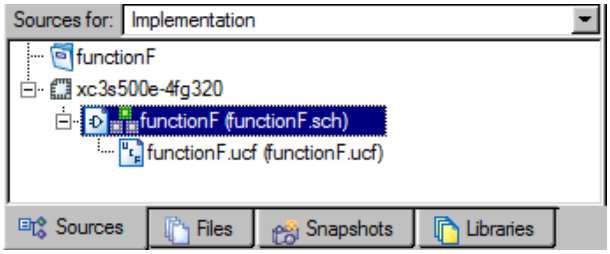

Figura 4.13 Sursele proiectului **functionF**.

Toate fişierele necesare fiind incluse, se poate trece la implementare. Selectaţi fişierul **functionF.sch**. În sub-fereastra *Processes for: functionF* (figura 4.14) sunt prezentate etapele de implementare a circuitului în FPGA. Etapele sunt:

- **Sinteză** (*Synthesize - XST*) - convertirea descrierii schematice sau textuale a proiectului într-o listă de primitive asociate tehnologiei FPGA Xilinx. Sinteza este urmată de generarea unui raport, a unor reprezentări grafice a circuitului rezultat şi generarea unui model de simulare post-sinteză.

- **Implementare** (*Implement Design*) - etapă formată din trei sub-etape.

  - **Translatare** (*Translate*) - convertirea proiectului de la diverse tipuri de descrieri (limbaje, scheme, grafuri, etc.) la un mod unitar de reprezentare.

  - **Asociere** (*Map*) - asocierea dintre operatori logici abstracţi şi resurse existente în circuitul FPGA.

  - **Plasare şi rutare** (*Place & Route*) - plasarea resurselor în structura circuitului şi rutarea acestora pentru a îndeplini funcţia proiectată.

- **Generarea fişierului de programare** (*Generate Programming File*) - generarea fişierului ce conţine informaţia despre modul de programare a tuturor resurselor hardware existente pe circuitul FPGA. Fişierul va avea extensia **.bit**.

- **Configurarea dispozitivul** (*Configure Target Device*) - lansarea unui utilitar software care permite trimiterea fişierului **.bit** de la calculator la circuitul FPGA. Ca o alternativă, se poate converti fişierul **.bit** într-o imagine a unei memorii ROM din care să se programeze circuitul FPGA la fiecare punere sub tensiune.

Aceste etape pot fi parcurse una câte una sau toate deodată. După fiecare etapă se generează rapoarte ce pot fi monitorizate de către proiectant.

Pentru acest prim proiect, ignoraţi etapele intermediare şi daţi dublu clic pe **Generate programming file**. Etapele se vor rula succesiv şi în final se va genera un fişier bitstream cu extensia **.bit**. Acest fişier, **functionF.bit**, va fi încărcat pe placa *Spartan3E* şi va programa dispozitivul FPGA cu funcţia proiectată.

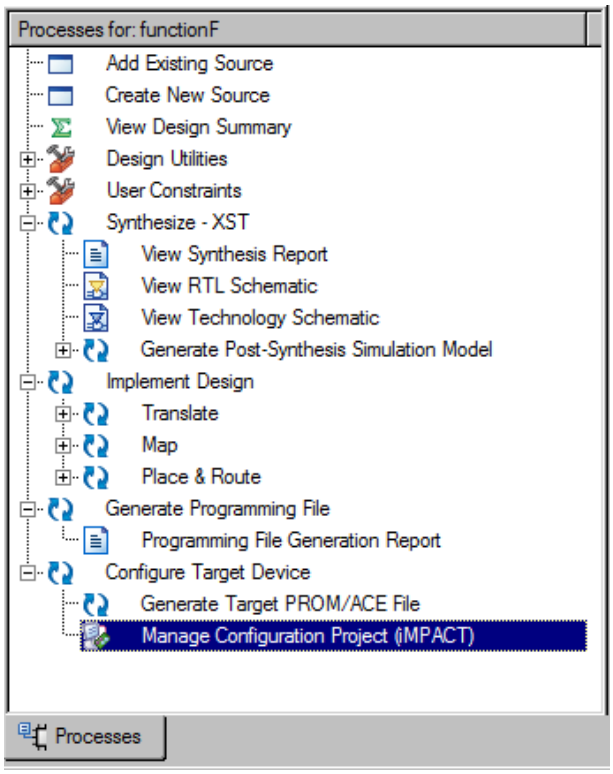

Figura 4.14 Etapele de procesare pentru implementare pe FPGA.

În urma procesării, se creează un fişier cu modelul Verilog asociat descrierii schematice, fişier ce va avea extensia **.vf**. În cazul schemei prezentate, modelul Verilog conţine instanţieri de porţi logice:

```
‘timescale 1ns / 1ps

module functionF(A,
                 B,
                 C,
                 D,
                 F);

    input A;
    input B;
    input C;
    input D;
   output F;

   wire AB;
   wire CD;

   AND2 XLXI_1 (.I0(B),
                .I1(A),
                .O(AB));
   AND2 XLXI_2 (.I0(D),
                .I1(C),
                .O(CD));
```

```
   OR2 XLXI_3 (.I0(CD),
               .I1(AB),
               .O(F));
endmodule
```

4.3.1 Încărcarea pe placă a fişierului bitstream (.bit)

Pentru a încărca pe placă fişierul bitstream, expandaţi în sub-fereastra *Processes* **Configure Target Device** şi daţi dublu clic pe **Manage Configuration Project (iMPACT)**.

Circutul FPGA este programat printr-o interfaţa JTAG. Pe macheta 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 iniţială 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 apare o reprezentare grafică a lanţului celor trei circuite, aşa ca în figura 4.15.

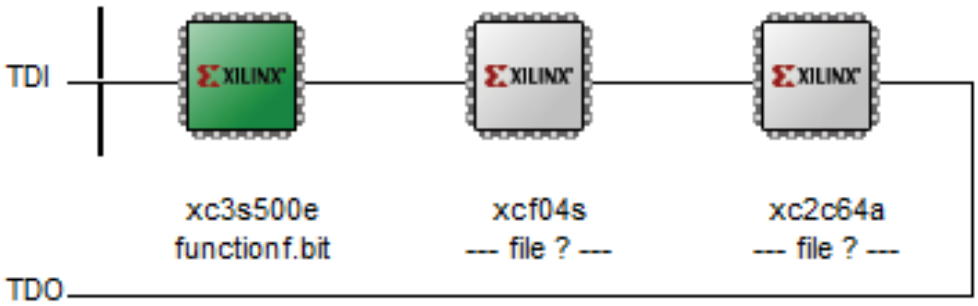

Figura 4.15 Reprezentarea grafică a lanţului JTAG cu cele trei componente: FPGA, CPLD, EEPROM.

Fiecare din cele trei circuite pot fi programate prin interfaţa JTAG şi are asociat câte un fişier de programare. În prima fereastră ce apare faceţi clic pe *Finish*. După aceea va trebui să selectaţi fişierul ce va fi încărcat pe circuitul FPGA (primul în lanţ) şi să faceţi clic pe **Open**. La următoarele două modele circuite nu selectaţi niciun fişier şi faceţi clic pe *Bypass*.

În ultima fază, faceţi clic dreapta pe circuitul FPGA **xc3s500e** şi clic **Program**.

Prin comutarea switch-urilor în toate stările, completaţi tabelul de adevăr (4.1) pentru funcţia implementată şi verificaţi-l prin comparaţie cu expresia analitică a funcţiei.

Tabelul 4.1

Tabelul de adevăr al funcţiei *F*.

A	B	C	D	F
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

4.4 Testarea cunoştinţelor

Realizaţi schema pentru următoarele funcţii şi testaţi-le pe placă. Funcţiile G şi H depind de aceleaşi intrări.

$$\begin{aligned} G &= A + B\overline{C} + D \\ H &= ((A \oplus B)C)D \end{aligned} \tag{4.2}$$

Scrieţi fişierul **.ucf** prin care asociaţi intrările cu switch-uri şi ieşirile cu leduri.

Completaţi tabelul de adevăr (4.2) pentru funcţiile implementate şi verificaţi-l.

Tabelul 4.2

Tabelul de adevăr al funcţiilor *G* şi *H*.

A	B	C	D	G	H
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		