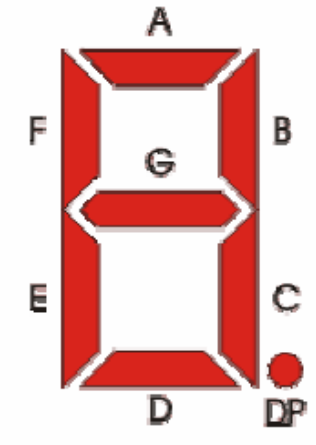
LUCRAREA 1. Decodificator HEX pentru un afişor LED 7segmente

Obiectivul lucrării este implementarea unui decodificator hexazecimal pentru un afişor LED 7 segmente si familiarizarea cu mediul de programare Xilinx Webpack precum si cu limbajul VHDL. Suportul hardware pentru implementare este alcătuit din sistemul de dezvoltare Digilent IIE (circuit FPGA Spartan IIE) si modulul de intrări-ieşiri DIO-1.



Intrări	Ieşiri	Cifra afişata
d3 …d0	s6 s5 s4 s3 s2 s1 s0 G F E D C B A	
0000	1 0 0 0 0 0 0	0
0001	1 1 1 1 0 0 1	1
0010	0 1 1 0 1 0 0	2
0011	0 1 1 0 0 0 0	3
0100	0 0 1 1 0 1 1	4
0101	0 1 0 0 0 1 0	5
0110	0 0 0 0 0 1 0	6
0111	1 0 0 0 0 0 0	7
1000	0 0 0 0 0 0 0	8
1001	0 0 1 1 0 0 0	9
1010	0 0 0 1 0 0 0	A
1011	0 0 0 0 0 1 1	b
1100	1 1 1 0 0 0 0	C
1101	0 1 0 0 0 0 1	d
1110	0 0 0 0 1 1 0	E
1111	0 0 0 1 1 1 0	F

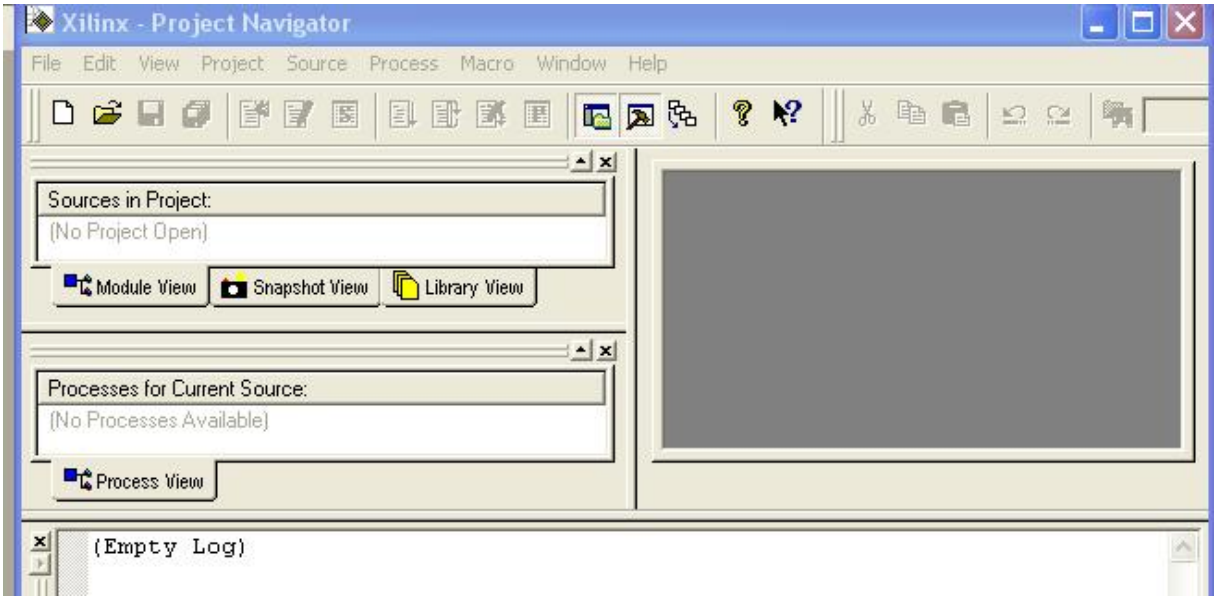
Afişor Anod Comun (CA): sx=0 segment aprins, sx=1 segment stins (tabelul de mai sus, are si greşeli, găsiți-le!!!)
Afişor Catod Comun (CC): sx=0 segment stins, sx=1 segment aprins
DP- punct zecimal

MODUL DE LUCRU

După lansarea **WebPACK Project Navigator** (double click pe icoana corespunzătoare de pe desktop  sau cu **Start, Programs, Xilinx WebPACK sau ISE 5, Project Navigator)** va apărea o fereastră cu un proiect neinițializat sau fereastra corespunzătoare ultimului proiect la care s-a lucrat(in acest caz se inchide cu File, Close Project). Fereastra respectivă are 4 panouri distincte:

1. Un **panou surse** (Sources in the Project) în care este prezentată organizarea fişierelor sursă care alcătuiesc proiectul. Există 4 sub panouri care pot fi aduse în panoul principal: Module View (implicit, în care se va lucra), FileView, Snapshots View şi Libraries View.
2. Un **panou procese** (Processes for Current Sources) în care sunt descrise diversele operații care se pot realiza asupra obiectelor din panoul surse.
3. Un **panou de raportare** (Log), în partea de jos a ferestrei, în care se vor afişa mesaje provenind de la procesele în lucru.
4. Un **panou de editare surse HDL,** în partea dreaptă, în care se editează sursele HDL sau schemele, cu editoare corespunzătoare.

În partea de sus a ferestrei principale există **o bară cu meniuri de lucru** (având fiecare o serie de submeniuri): **File**, **Edit**, **View**, **Project**, **Source**, **Process** , **Macro**, **Window**, **Help**. La unele din acestea se vor face referiri în continuare.



1. Inițializarea proiectului

Se începe prin crearea unui nou proiect: selectăm din bara de meniuri **File** ..**New Project**. Aceasta va deschide fereastra **New Project**, prin intermediul căreia vom stabili: **locația proiectului** (directorul de lucru), **numele său**, **tipul şi pinout-ul circuitului** FPGA utilizat precum şi **programul de sinteză logică** utilizat pentru translația sursei HDL (implicit si limbajul HDL utilizat).

Se apasă butonul **…** (similar Browse) de lângă câmpul **Project Location** pentru a a stabili directorul în care vor fi memorate fişierele proiectului(aplicatiei). Directorul de lucru va fi întotdeauna de forma *C:\lucru\proiectul_meu*, unde *C:\lucru* este un director in care utilizatorul are drepturi depline de acces(pentru Windows NT4.0 sau Windows2k). **Numele directorului creat astfel este identic cu numele ales pentru proiect**. In acesta vor fi create(salvate) toate fisierele aferente proiectului.

Se va da un nume proiectului, în câmpul **Project name,** de exemplu **Proiect1 sau dec7led.**

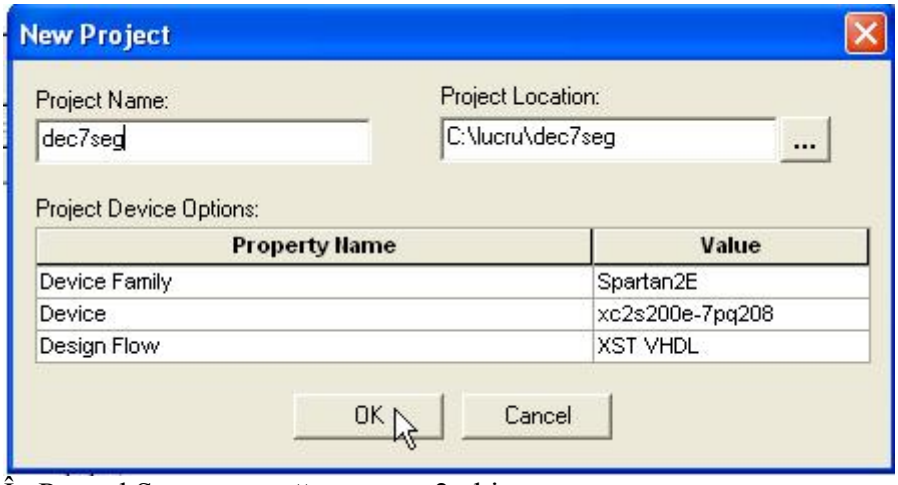
Din panoul **Property Name** se aleg şi se selectează consecutiv (meniurile sunt de tip pop-up şi se activează prin selecția câmpului şi click pe bara care apare în dreapta câmpului):

Device Family: **Spartan2E**

Device : **xc2s200e-7pq208** (acesta este circuitul utilizat pe sistemul de dezvoltare)

Synthesis Tool : XST VHDL

După finalizare se apasă butonul **OK**. În acest moment în directorul de lucru va fi creat un nou director având ca nume numele dat proiectului.



În Panoul Surse vor apărea acum 2 obiecte:

- un obiect de tip proiect numit **Untitled**
- un obiect de tip circuit numit **xc2s200e-7pq208 –XST VHDL**

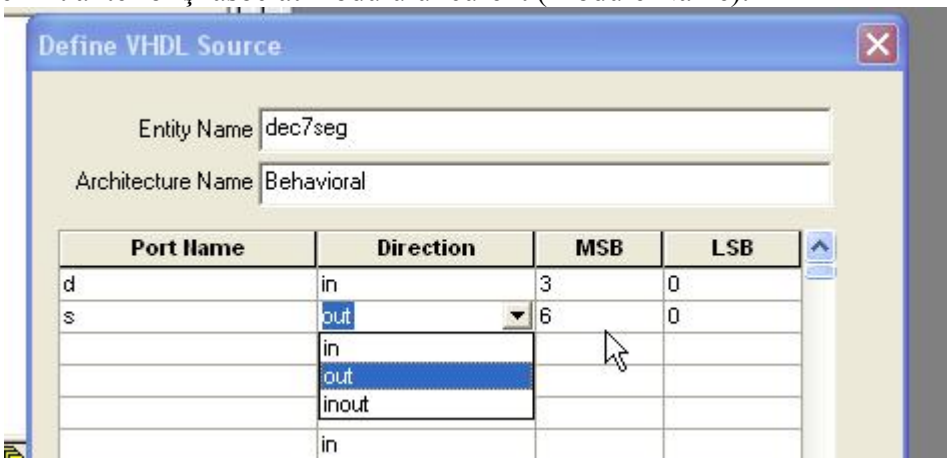
Se poate da şi aici un nume proiectului: se selectează Untitled (cu click) şi apoi cu **click dreapta** va apare un meniu pop-up din care se selectează ultimul câmp **Properties.** Se va completa câmpul **Project Title** cu acelaşi nume, **Proiect1** (se poate da orice nume de fapt!) şi apoi se apasă butonul **OK** pentru a închide fereastra.

2. Descrierea aplicației cu ajutorul unei surse VHDL

Odată finalizată inițializarea proiectului se poate începe descrierea aplicației. Vom începe prin a adăuga proiectului nostru **dec7led** o sursă (un fişier) VHDL. Se face **click dreapta** pe obiectul **xc2s200e-7pq208 –XST VHDL** şi din meniul pop-up se selectează **New Source**.

În noua fereastră **New** se va selecta (în panoul din stânga) **VHDL Module** şi apoi se va da un nume fişierului sursă în câmpul **File Name,** cum ar fi **dec7seg**. Se verifică să fie bifat checkbox-ul **Add to Project** şi apoi se va trece la fereastra următoare cu **Next** .

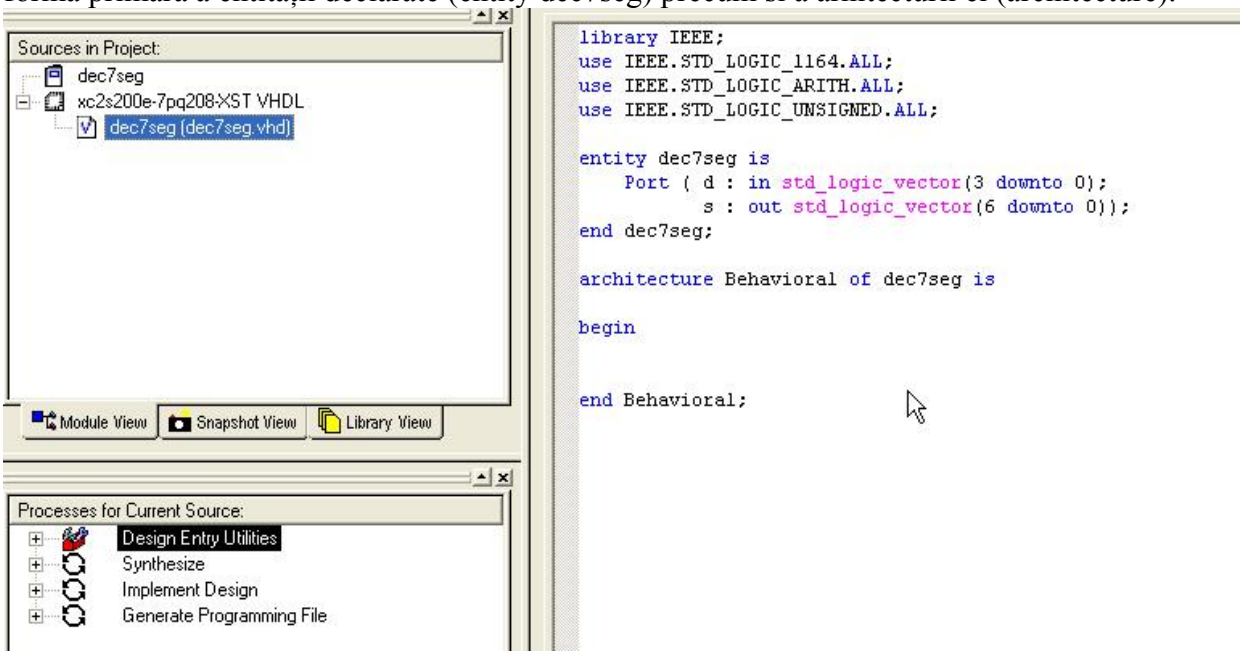
Noua fereastră se numeşte **Define VHDL Source** în care apare şi numele fişierului sursă definit anterior şi asociat modulului curent (Module Name).



Aici vom defini intrările şi ieşirile decodificatorului 7 segmente: vom completa pe rând (de sus în jos) câmpurile **Port Name**: **d** (vectorul binar pentru intrări), **Direction: in**, **MSB:3**, **LSB:0** apoi **Port Name**: **s** (vectorul binar pentru ieşirile asociate celor 7 segmente), **Direction: out**, **MSB:6**, **LSB:0**. După terminare se apasă **Next** şi în noua fereastră (New Source Information) vom vedea un doar sumar al informației introduse. Introducerea se finalizează şi fereastra se închide cu butonul **Finish** .

OBS. Se poate sări peste această etapă, definirea interconexiunilor externe făcându-se prin editarea directă a fişierului sursă (în fereastra **Define VHDL Source** se dă **Next** şi apoi **Finish**). Si in cazul nostru vom completa descrierea interconexiunilor externe in faza de editare a surselor.

În fereastra din dreapta (a editorului) va apărea acum un model (template, machetă) pentru fişierul nostru sursă VHDL în care sunt descrise bibliotecile implicite (library, use) si forma primara a entității declarate (entity dec7seg) precum si a arhitecturii ei (architecture).



În **panoul surse** apare şi fişierul nostru asociat acum obiectului **xc2s200e-7pq208 –XST VHDL** de forma nume_modul (nume_fisier.vhdl) . **Prin selectarea numelui** acestui obiect, în **panoul procese** vor apărea şi procesele disponibile: **Design Entry Utilities**, **Syntesize, Implement Design, Generating Programming File.**
Pentru fiecare din aceste procese exista un numar de sub-procese ce pot fi facute vizibile prin expandarea unui arbore corespunzator procesului (cu click pe „+")

Lansarea şi rularea oricărui proces sau subproces din lista se face prin double click pe proces sau click dreapta şi din meniul pop-up se selectează Run.

La aceste procese vom face referiri în continuare.

Fişierul trebuie evident **completat (editat)**, completarea referindu-se la **descrierea arhitecturii aplicației cu ajutorul unei tabele de adevăr(truth_table)** şi **asignarea unui număr de pin** (pentru ca aplicația să poată fi testată pe sistemul de dezvoltare). Sursa completă este dată în continuare.

Editorul utilizat este unul interactiv, orientat pe surse HDL (VHDL in cazul nostru). Apar cu culori diferite cuvintele cheie (albastru), comentariile (verde) respectiv simbolurile definite de utilizator(negru), acest lucru ajutand la corectarea sintactica in faza de editare primară a sursei HDL.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

```
-- identificarea segmentelor
--    -a-
-- f|     |b
--    -g-
-- e|     |c
--    -d-
--
--    -s0-
-- s5|     |s1
--    -s6-
-- s4|     |s2
--    -s3-
--cda in "0" pt a aprinde segmentul (afisor cu anod comun)

--declaratie entitate
entity dec7seg is
    Port ( d : in std_logic_vector(3 downto 0);
           s : out std_logic_vector(6 downto 0);
           c : out std_logic_vector(3 downto 0));

end dec7seg;

--declaratie arhitectura entitate
architecture Behavioral of dec7seg is

begin
--comenzi anozi, "1" pt a activa afisorul
c     <=    "0001";
--comenzi segmente
s      <=   "1000000" when d="0000" else --0
            "1111001" when d="0001" else --1
            "0100100" when d="0010" else --2
            "0110000" when d="0011" else --3
            "0011001" when d="0100" else --4
            "0010010" when d="0101" else --5
            "0000010" when d="0110" else --6
            "1111000" when d="0111" else --7
            "0000000" when d="1000" else --8
            "0011000" when d="1001" else --9
            "0001000" when d="1010" else --A
            "0000011" when d="1011" else --b
            "1000110" when d="1100" else --C
            "0100001" when d="1101" else --d
            "0000110" when d="1110" else --E
            "0001110";                --F

end Behavioral;
```

Se salvează fişierul editat cu **File** .. şi **Save**.. sau cu click pe icoana corespunzătoare din Toolbar  .

Se poate observa ca interfața entității a mai fost completata prin adăugarea vectorului de ieşire c3..c0 utilizat pentru comanda anozilor comuni ai afişoarelor.

TEMĂ Pentru fiecare cifră hex **0..F** din tabela de adevăr să se deseneze configurația corespunzătoare segmentelor s6 .. s0 aprinse sau stinse.

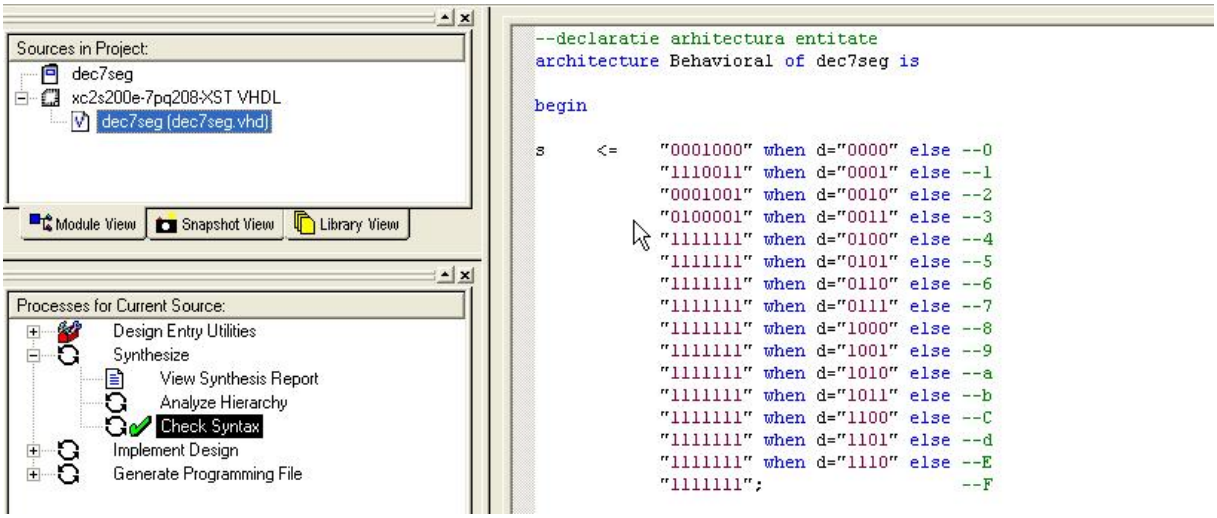
3. Verificarea sintaxei VHDL şi corectarea erorilor sintactice

În panoul procese se dă click pe semnul + (care indică prezența unui arbore de subprocese) din dreptul procesului **Synthesize** şi prin expandarea arborelui respectiv se pun in

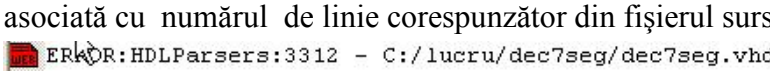
evidență subprocesele: **View Synthesis Report**, **Analyze Hierarchy** si **Check Syntax**. Procesele sau subprocesele pot fi rulate pe rând, individual, sau în secvență.

Lansarea şi rularea oricărui proces sau subproces se face prin double click pe proces sau click dreapta şi din meniul pop-up se selectează Run.

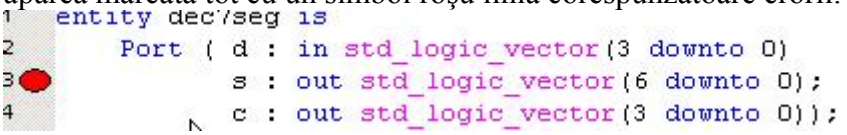
Faptul că un proces sau subproces s-a finalizat cu succes este marcat, în dreptul său, printr-un simbol checked (V) de culoare verde, în caz contrar apărând un simbol de eşec (X) de culoare roşie. Simbolul « ? » sau «!» in galben (?, !) indică o atentionare (warning) legată de rezultatul operatiei.



Pentru verificarea sintaxei se lansează procesul **Check Syntax**. În cazul în care avem o eroare de sintaxă, ea este raportată **în panoul de raportare** (de regulă cu un mesaj final de forma „*Done: failed with exit code: 0001*.” sau „*Error: XST failed*”). Se caută in panoul de raportare locul-linia in sursa unde a apărut eroarea (marcat cu **un simbol roşu**) ea fiind asociată cu numărul de linie corespunzător din fişierul sursă.



Se da un dublu click pe linia respectiva din panoul de raportare si in fişierul sursa va apărea marcată tot cu un simbol roşu linia corespunzătoare erorii.



Atenție: in cazul erorilor multiple rezolvarea/corectarea erorilor de sintaxa trebuie începuta cu prima apărută (ca in orice limbaj de programare)!

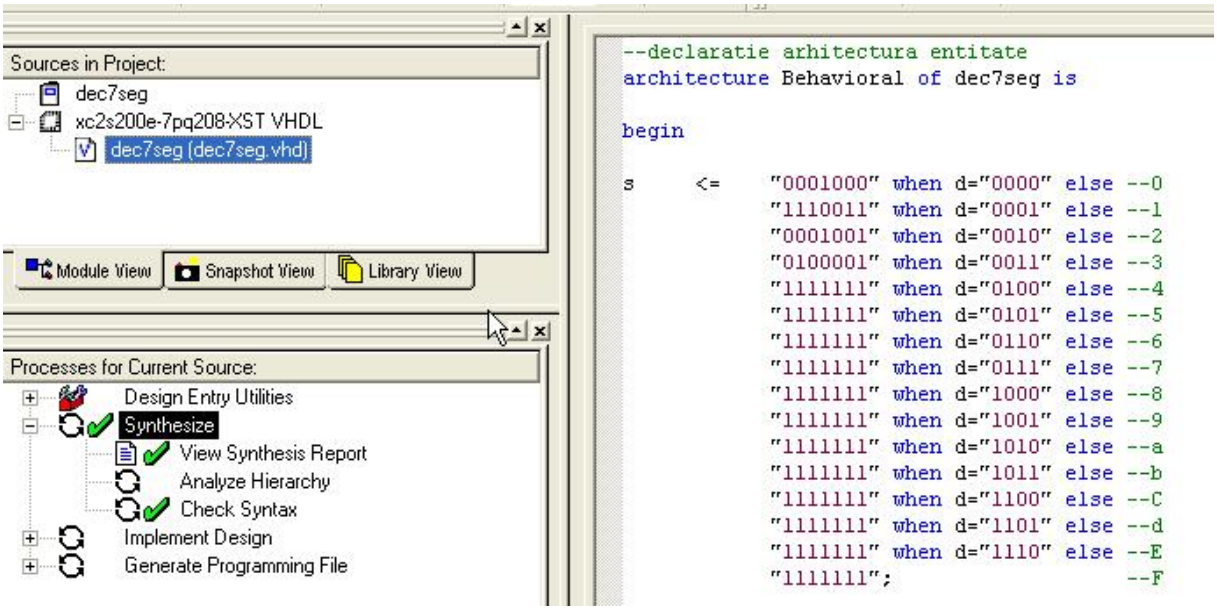
Pe marginea de jos a ferestrei Project Navigator există un indicator pentru pozitia curentă (linie, coloană) a cursorului in fereastra de editare, de forma Ln:xxxx, Col:yyyy astfel că, alternativ, se poate căuta linia sau liniile respective, se corectează eroarea sau erorile, se salvează fişierul corectat şi se repetă operația până când nu mai există erori sintactice.

Finalizarea cu succes a unui proces sau sub-proces este indicată cu mesajul „*Done: completed successfully*.” sau „*Completed process "Check Syntax"* fără nici un mesaj de tip Error!.

Atenție: cuvintele cheie VHDL nu depind de tipul de literă (mare sau mică), **fiind „case insensitive”, dar exista restricții legate de numele simbolurilor definite de utilizator !!!**

4. Sinteza logică

După verificarea sintactică se poate lansa întreg procesul **Synthesize**. Programul de sinteză va transforma descrierea HDL într-un netlist de porți logice.



După finalizarea cu succes a procesului (mesaj: *Done: completed successfully*) rezultatele sintezei si opțiunile utilizate pentru sinteza pot fi vizualizate prin intermediul unor fişiere de raportare prezente în sub arborele procesului: pentru sinteza propriu-zisă **View Synthesis Report.** Vizualizarea se face analog lansării oricărui proces: double click pe obiect sau click dreapta si **Run** din meniul pop-up, fiind lansat in mod automat un program de vizualizare (Viewer) in fereastra de editare a surselor.

Nu se va face si o verificare funcțională.

5. Implementarea aplicației

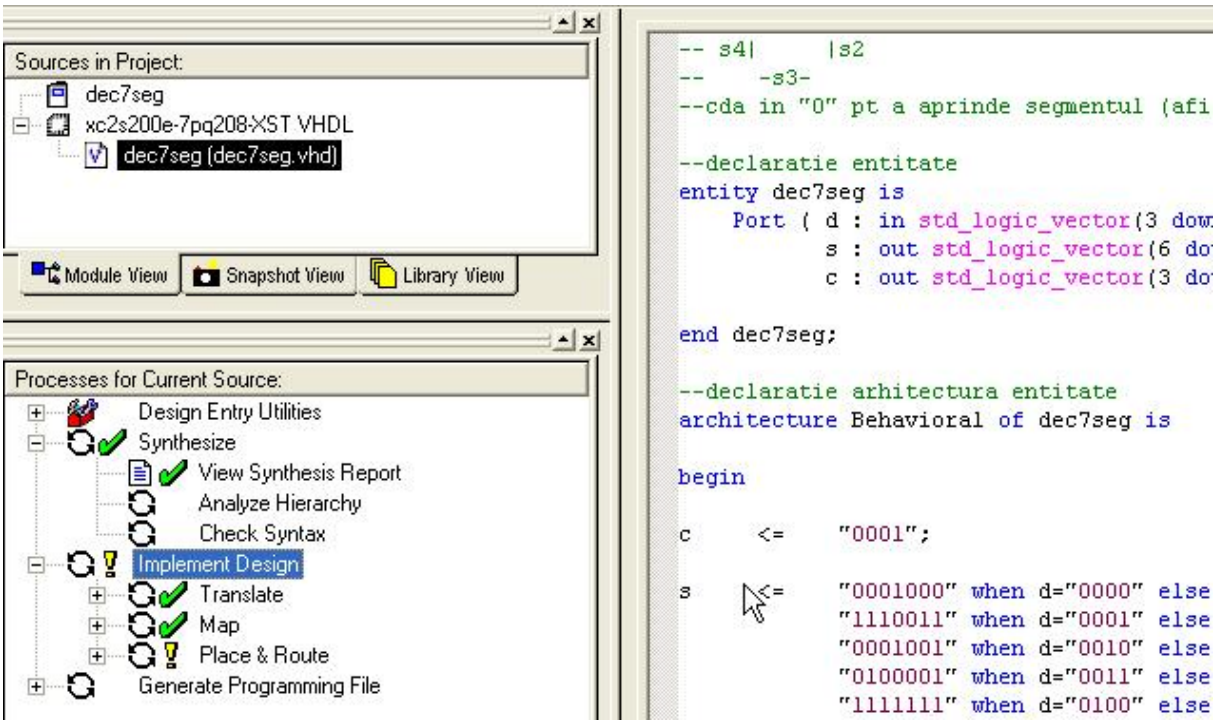
Avem acum o descriere logică a aplicației (sub forma unui netlist) şi rămâne ca aceasta să fie implementată ținând cont de resursele circuitului FPGA **xc2s200e-7pq208**, pentru ca aplicația să poată fi folosită. Se lansează procesul **Implement Design,** în care avem ca **subprocese: Translate, Map si Place & Route.** Desfăşurarea şi progresul în timp al procesului poate fi urmărit prin intermediul unei bare de stare(status bar) în partea din dreapta jos a ferestrei principale Project Navigator.

Trebuie notata apariția eventuală a simbolului de avertizare (?) in dreptul sub-procesului Place& Route care indică existenta unor avertizări sau faptul ca nu toate sub-procesele (si sub-procesele pot avea la rândul lor alte sub-procese!) au fost activate si/sau executate.

Din toate procesele, acesta este procesul cel mai intensiv computațional şi care, în consecință, durează cel mai mult (proporțional cu complexitatea aplicației).

Eventualul eşec al procesului este indicat corespunzător în fereastra de raportare (mesaj final *Done: failed with exit code: nnnn*) indicându-se şi cauza erorii. Cauzele eşecului nu sunt legate de regula de sursa VHDL, decât dacă in sursa VHDL s-a făcut si asignarea conexiunilor externe ale circuitului FPGA (de ex. exista o asignare greşită a pinilor), ceea ce la noi nu este cazul. Asignarea respectiva se va face in cazul nostru prin intermediul unui fişier special (fişier constrângeri).

Evident dacă eroarea provine din sursa VHDL, se corectează, se salvează şi **se repetă procesele de sinteză şi implementare**.

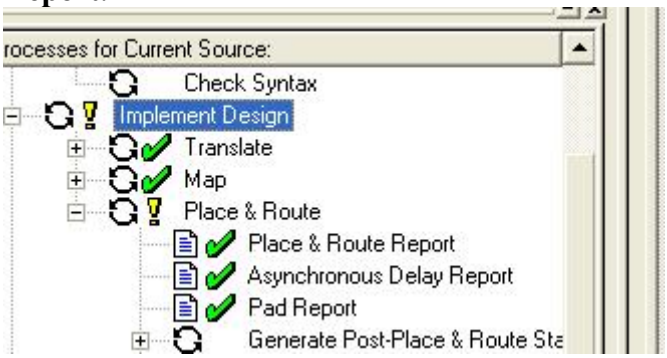


Unele din motivele pentru care acest proces poate eşua sunt:

- resursele circuitului sunt insuficiente pentru implementare
- pinii asignați (ca număr-identificator depin) nu pot fi folosiți pentru intrări/ieşiri de uz general, fiind pini de alimentare (Vcc sau GND) sau corespunzând interfeței de programare JTAG
- pinii asignați sunt dublu sau multiplu definiți (avem semnale diferite cărora le sunt asignați aceiaşi pini)

6. Verificarea modului în care s-a făcut implementarea

După finalizarea cu succes (mesaj final *Done: completed succesfully*) a procesului **Implement Design** se pot examina fişierele de raportare: **Place & Route Report** sau **Pad Report**.



Din aceste fişiere se pot extrage o serie de informații utile cum ar fi:

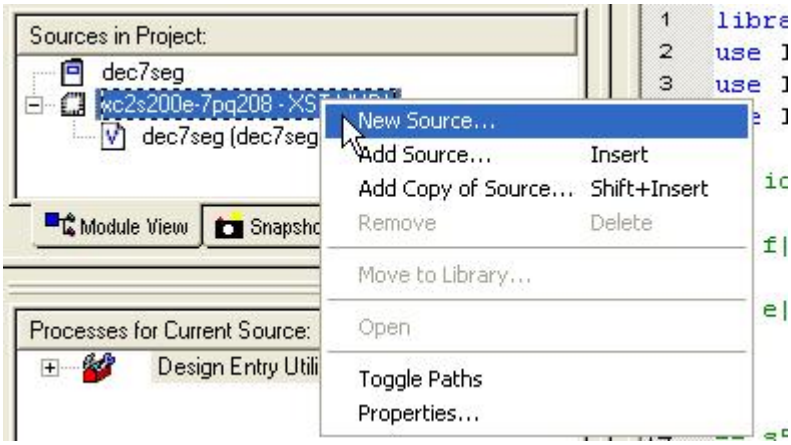
- gradul de utilizare a resurselor circuitului (**Place & Route Report**): numărul de SLICE-uri unitati elementare utilizate, numărul de blocuri intrare-iesire (IOB) utilizate
- asignarea pinilor (**Pad Report**); se poate vedea numele semnalului (ex. s(5)) si numarul de pin (ex. P35) precum şi alte informatii cum ar fi tipul de interfata electrica(LVTTL), conexiunile rămase neutilizate (UNUSED), borne de alimentare (VCCO, VREF, GND), interfața JTAG (TDI,TDO,TCK,TMS), etc.

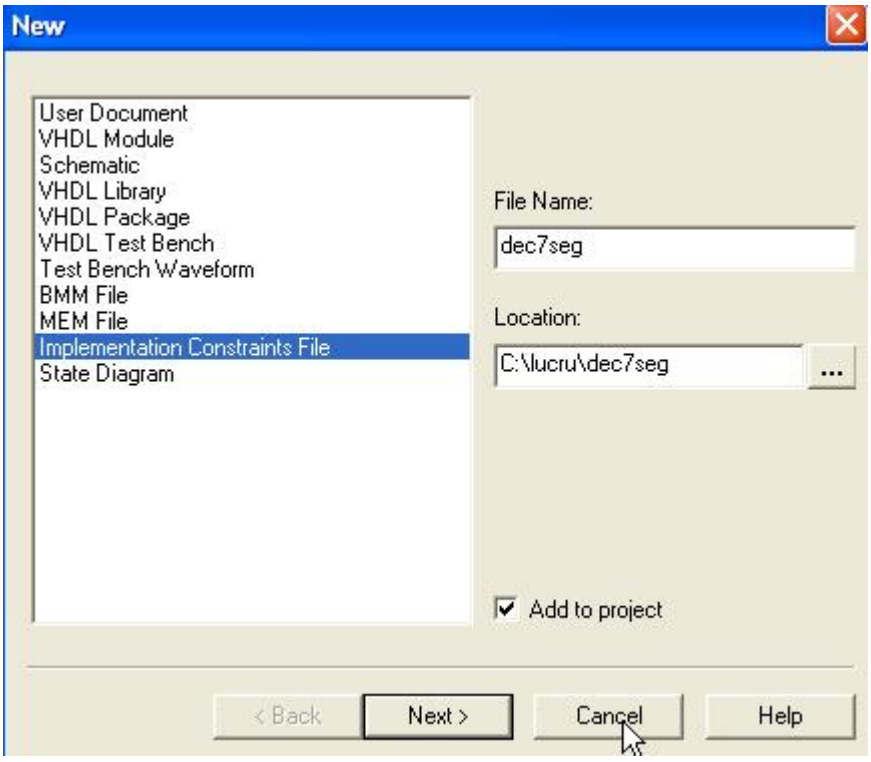
De mentionat ca in cazul nostru asignarea s-a facut automat neexistand definita nici o constrangere in acest sens (o vom face in etapa urmatoare). Acesta este si motivul avertizarii (!) la nivelul sub-procesului Place& Route.

7. Asignarea (constrângerea) conexiunilor externe ale circuitului FPGA.

Pentru ca aplicația noastră sa poate fi utilizata pe platforma hardware disponibila (DIIE+DIO-1) este necesar ca sa avem o corespondenta intre intrările-ieşirile aplicației noastre si resursele hardware.

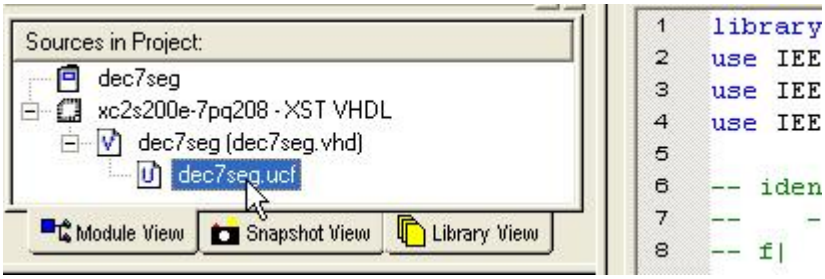
Se selectează identificatorul proiectului **xc2s200e-7pq208-XST VHDL**, apoi se da click dreapta si in meniul pop-up se selecteaza **New Source**. In fereastra New se selectează tipul **Implementation Constraint File** căruia i se va da tot numele dec7seg. Rezultatul va fi creerea unui fisier dec7seg.ucf care va fi asociat aplicației si care va fi editat in continuare.





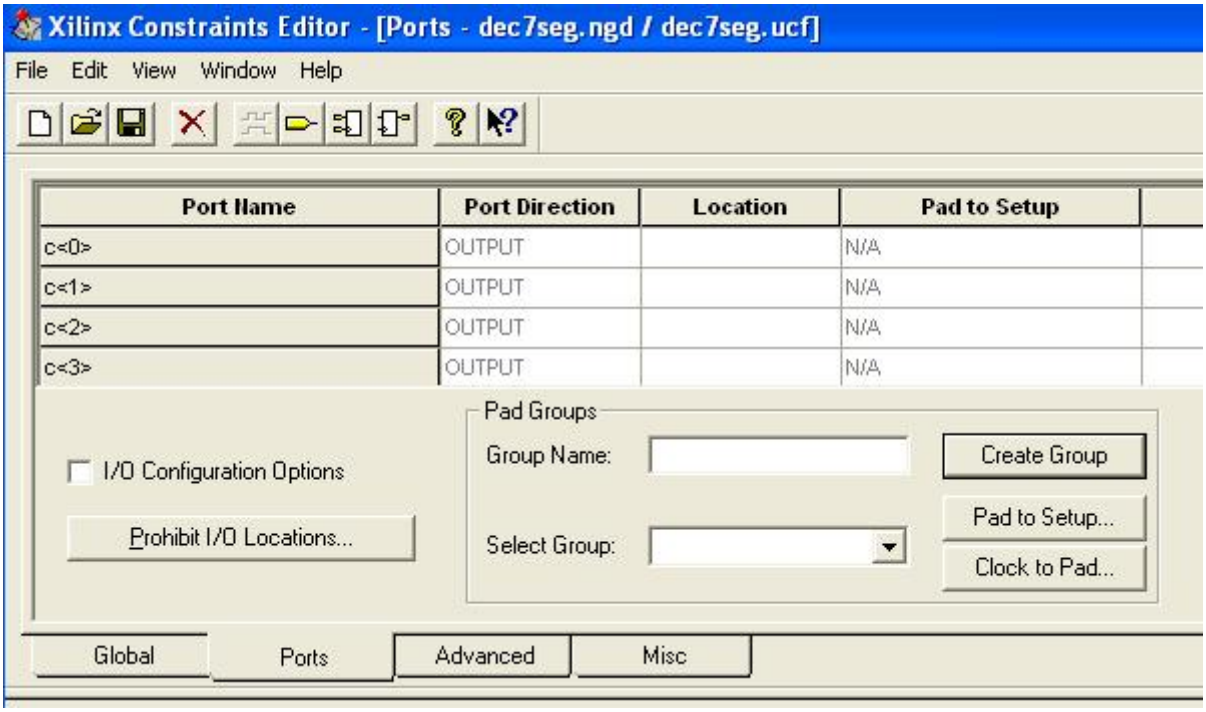
Se continua cu **Next** si **Finish**.

In continuare se va da un dublu- click pe numele fişierului nou creat.



Ca efect al acestei acţiuni se va deschide fereastra editorului Xilinx Constraints Editor. Pentru a putea constrânge alocarea interfeței externe a aplicației (a pinilor circuitului FPGA) se va selecta Tab-ul **Ports**. Aici in coloana **Port Name** vor apărea celule cu numele semnalului respectiv (ex. c(0), s(1), d(3), etc.), iar in coloana **Location** se va face atribuirea locației sub forma Pnumar_pin.(ex. P124). In coloana **Direction** este descris tipul semnalului, intrare (INPUT), ieşire (OUTPUT) sau bi-direcțional (INPUT-OUTPUT). Singura coloana care se va edita este **Location** având ca referința tabelele date in continuare. Se selectează celula (cu click) si se editează sau se poate face dublu click pe celula, in fereastra deschisa Location se completează câmpul Location si se da OK).

După editarea tuturor locațiilor se mai verifica odata si se iese cu File- Exit. Ca efect al acestei operații va apărea o fereastra de avertizare (Notice…) in care se va apăsa butonul **Reset**. Pentru ca noile locații (constrângeri) sa fie luate in considerare **trebuie repetat procesul de implementare (Implement Design).**

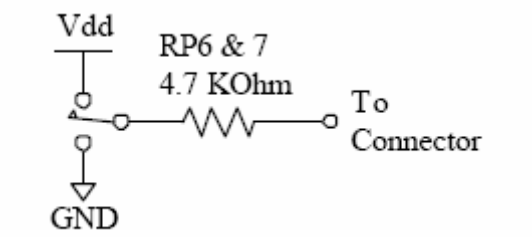


Completarea acestui fişier este legata de modul in care sunt utilizate conexiunile externe (pinii) ale circuitului FPGA pe sistemul de dezvoltare DIIE precum si de modul in care aceste conexiuni sunt dirijate prin intermediul conectorilor de expansiune spre modulul de intrări-ieşiri DIO-1.

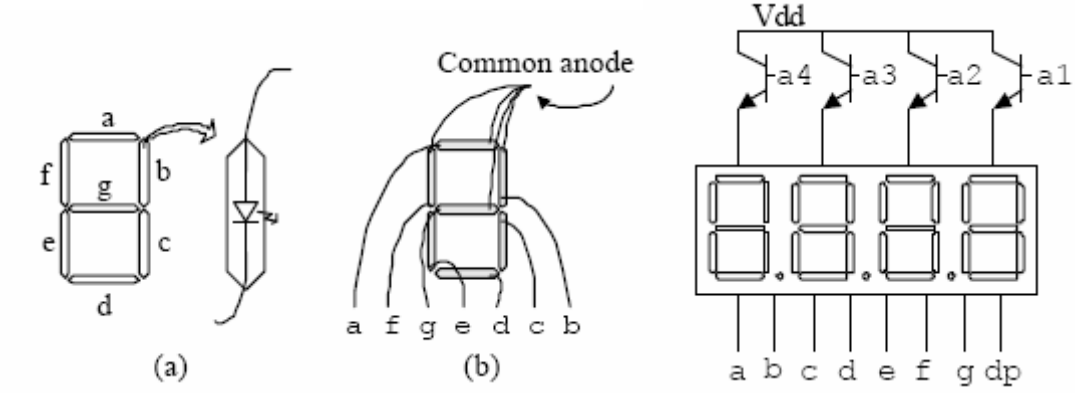
Celulele **Location** se vor completa conform tabelelor funcționale descrise in continuare.

Intrările pentru decodificator vor fi generate cu ajutorul a 4 (SW1..SW4) din cele 8 comutatoare de translație (SW1..SW8) aflate pe modulul DIO-1. Conexiunea (To connector) pentru fiecare din comutatoare este adusa prin intermediul conectorilor de expansiune pe sistemul DIIE si de aici la pinul corespunzător (P…) al circuitului FPGA.

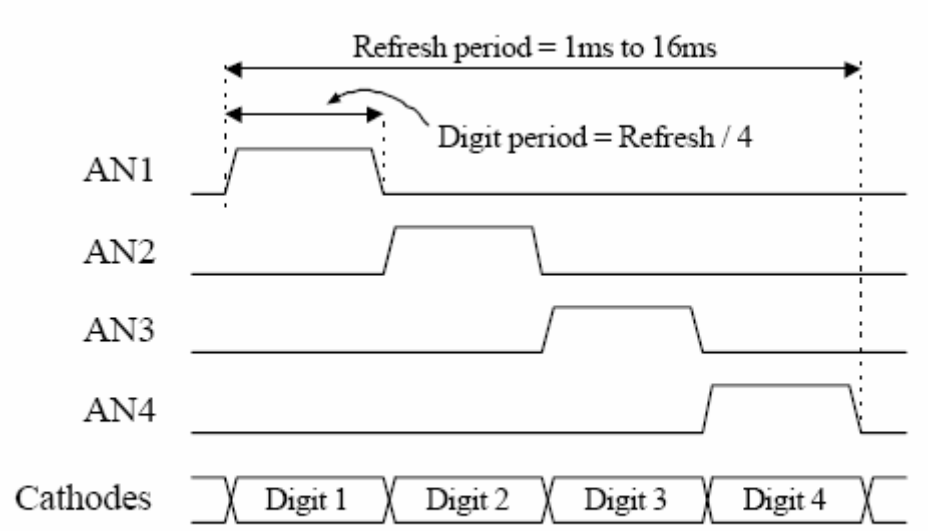
Intrări decodificator (vectorul d)	Comutator	Pini FPGA
d0	**SW1**	**P126**
d1	**SW2**	**P129**
d2	**SW3**	**P133**
d3	**SW4**	**P135**



Observația cea mai importantă legată de sistemul de afişare LED cu 7 segmente este că aceste 4 afişoare, **in varianta cu anod comun** (CA), sunt conectate pentru o utilizare cu multiplexare in timp.
Astfel segmentele corespunzătoare (a…g, dp) de la fiecare din cele 4 afişoare sunt conectate intre ele, existând doar o comandă individuală (a1..a4) pentru anodul comun al fiecărui afişor.



Pentru ca un segment al unuia din cele 4 afişoare sa fie aprins trebuie ca, in acelaşi moment de timp, linia comuna de segment sa fie comandata in „0" **si** comanda de anod individuala sa fie in „1" (de exemplu pentru a aprinde segmentul f de la afişorul din dreapta: a1="1" si f="0"). Aceasta facilitate o vom utiliza in aplicațiile următoare unde va fi necesar sa afişam valori pe mai multe cifre.
O diagramă temporală care ilustrează principiul de funcționare al unui afişaj cu multiplexare in timp este data in figura următoare.



Seven segment display refresh signals and timings

Comenzi anozi afişor (vectorul c)	Anod comun	Pini FPGA
c0	**a1**	**P160**
c1	**a2**	**P162**
c2	**a3**	**P164**
c3	**a4**	**P166**

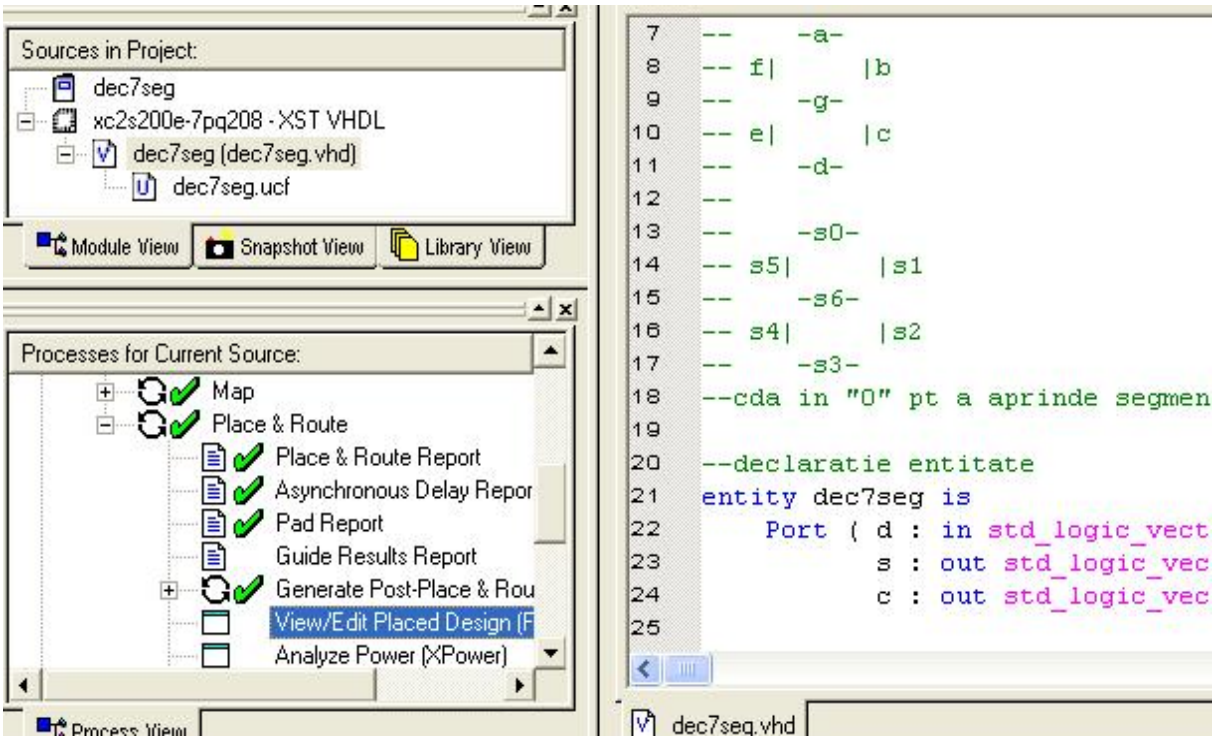
Comenzi segmente (vectorul s)	Segment	Pini FPGA
s0	**a**	**P127**
s1	**b**	**P132**
s2	**c**	**P134**
s3	**d**	**P136**
s4	**e**	**P139**
s5	**f**	**P141**
s6	**g**	**P146**

Conexiunile a1..a4 si s0..s6 sunt aduse prin intermediul conectorilor de expansiune pe sistemul DIIE si de aici la pinul corespunzător (P…) al circuitului FPGA.
Verificați aceste informații acestea folosind manualul si schemele DIIE si IO-1!

8. Vizualizarea configurației circuitului FPGA

După finalizarea procesului de implementare se poate obține o reprezentare grafia a modului in care au fost utilizate resursele circuitului FPAG, respectiv CLB-urile si blocurile de intrare-ieşire (pinii).

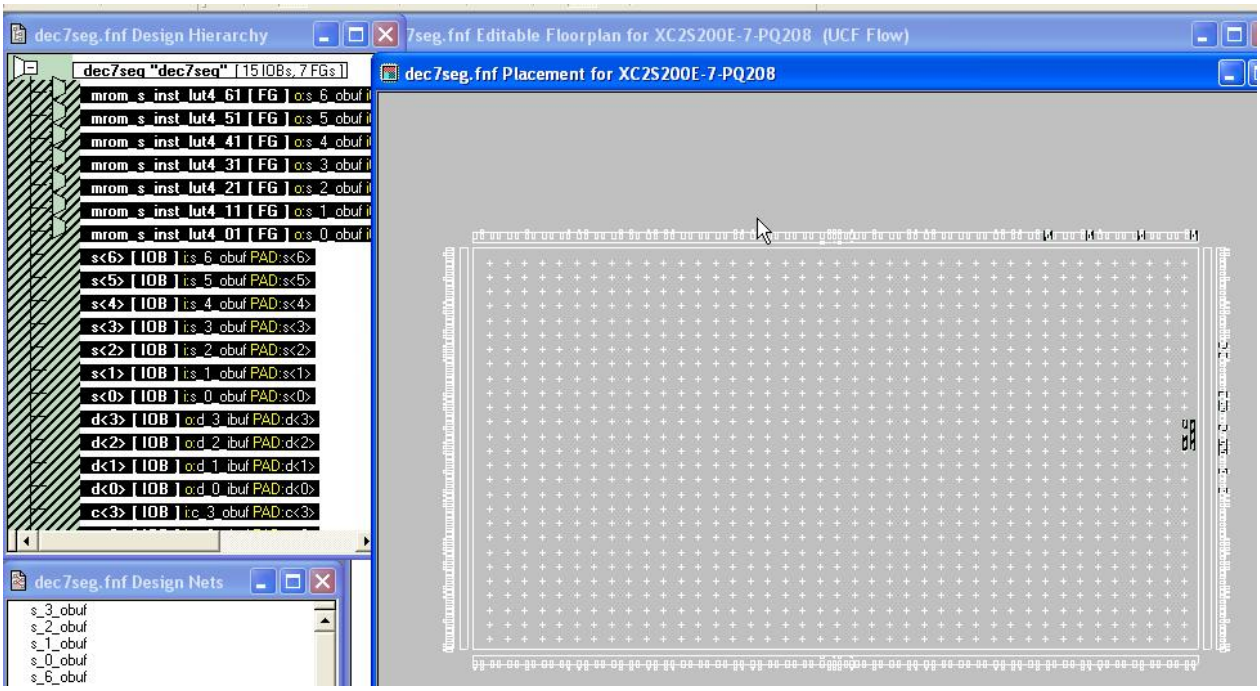
Pentru aceasta se va lansa procesul **View/Edit Placed Design** (FloorPlanner), de fapt un sub-proces al lui **Place & Route**, care va avea ca efect lansarea unei aplicații de sine stătătoare numită **Xilinx FloorPlanner**.



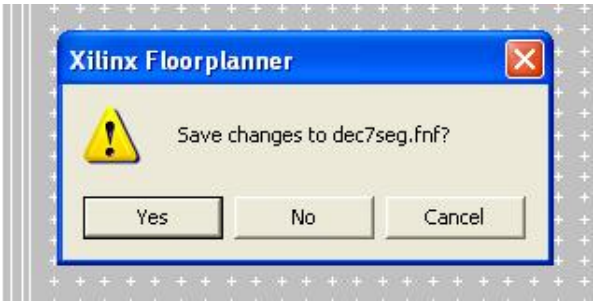
In fereastra FloorPlanner vom găsi:

- in zona **…Design Hierarchy** (stânga sus) apar intrările, ieşirile si LUT-urile utilizate pentru sinteza decodificatorului
- in zona **…Design Nets** (stânga jos) apar diversele net-uri (semnale interne) utilizate de aplicatie
- in zona ….**Placement for xc2s200E-7pq208** apare o matrice cu cele 28 x 42 de CLB ale circuitului FPGA precum si pinii distribuiți pe periferia acestei matrice

In figura de mai jos se observă că aplicația noastră foloseşte 15 IOB-uri (15IOBs, blocuri intrare-ieşire si pinii corespunzători) precum si 7 generatoare de funcții (7 FBs) implementate de fapt cu ajutorul unor LUT-uri. Resursele „ocupate" de aplicația noastră sunt practic infime in raport cu resursele totale ale circuitului! Astfel va fi nevoie sa utilizăm o scară de reprezentare corespunzătoare ca sa putem vizualiza zona respectivă din circuit (cu **View-Zoom In** sau butonul ).



In final deoarece nu dorim să modificăm nimic in aceasta configurație, se va ieşi din această aplicație cu **File – Exit,** având grijă să apăsam si butonul **No** ca răspuns la întrebarea „Save changes to…..” .



9. Generarea fişierului de programare de tip Bitstream

Ultima etapă corespunde si ultimului proces care trebuie parcurs în secvență şi anume generarea fişierului de programare. Se lansează procesul **Generate Programming File**. În principiu nu există nici un motiv pentru ca acest proces să nu se finalizeze corect, mesajul din fereastra de raportare fiind de forma: … *Saving bit stream in "dec7seg.bit"…. Bitstream generation is complete…Completed process "Generate Programming File"...*
De reținut ca informația necesară programării se afla in fişierul dec7seg.bit!

Spre deosebire de fişierele JEDEC **fişierul generat de tip .bit NU este un fişier ASCII** (text) al cărui conținut sa poată fi vizualizat (de ex. cu Notepad sau Wordpad).

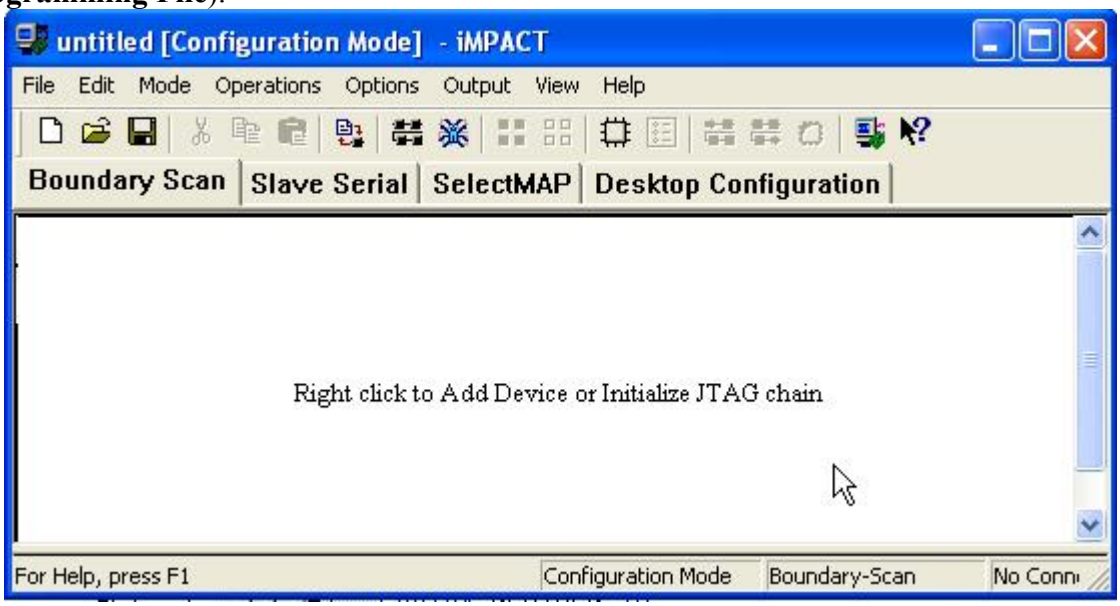
10. Programarea circuitului

Vom utiliza: **sistemul de dezvoltare DIIE** cu modulul DIO-1 deja conectat, **un alimentator extern** Digilent pentru alimentarea sistemului de dezvoltare, **un cablu de programare JTAG** care se conectează la portul paralel al calculatorului si la conectorul JTAG aferent DIIE (**ATENTIE**: borna notata Vcc a conectorului spre interiorul placii DIIE!).

Programarea circuitului FPGA utilizat pe sistemul de dezvoltare DIIE presupune **urmărirea ordinii de conectare**:

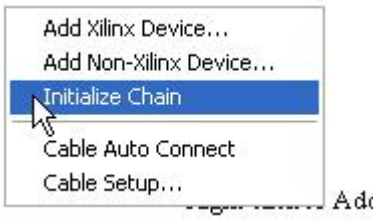
- se conectează cablul de programare JTAG la calculator si la sistemul de dezvoltare (**atenție** la **corespondenta** intre notațiile de pe conectorul JTAG al programatorului si la cele de la conectorul corespunzător pe sistemul de dezvoltare, conectorul nu are cheie si poate fi introdus si invers!!!)
- **doar apoi** se alimentează sistemul de dezvoltare (se conectează alimentatorul universal al sistemul de dezvoltare)

Se lansează sub procesul **Configure Device(Impact)** (un sub-proces al lui **Create Programming File**).



Fereastra inițiala **untitled [Configuration Mode]-iMPACT** se deschide in Tab-ul **Boundary Scan.**

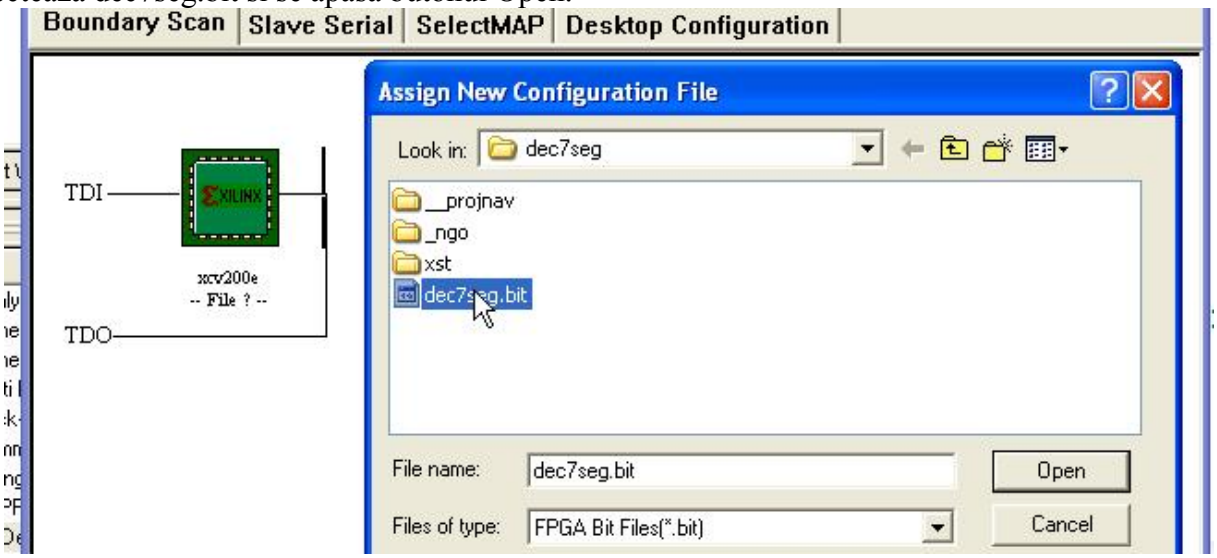
Se va face un click dreapta in fereastra corespunzătoare Tab-ului **Boundary Scan** (care trebuie sa fie selectat implicit). In meniul pop-up se selectează câmpul **Initialize Chain**.



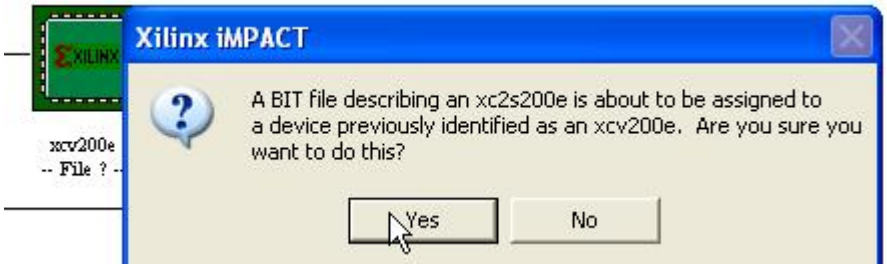
Se apasă butonul OK la fereastra care se va deschide in continuare.

In fereastra **Boundary Scan** va apărea circuitul identificat automat (identificarea este de fapt greşita el fiind văzut ca un circuit Virtex 200e!) cerându-se asocierea unui fişier de tip

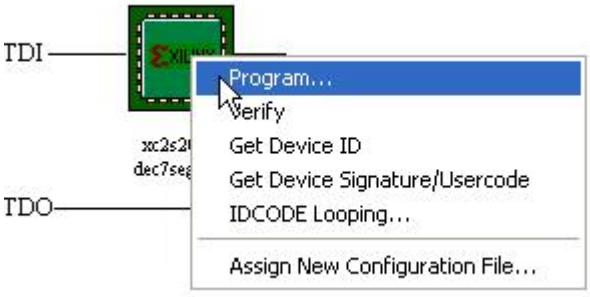
bitstream (.bit) care sa fie utilizat pentru programare (Assign New Configuration File). Se selectează dec7seg.bit si se apăsa butonul Open.



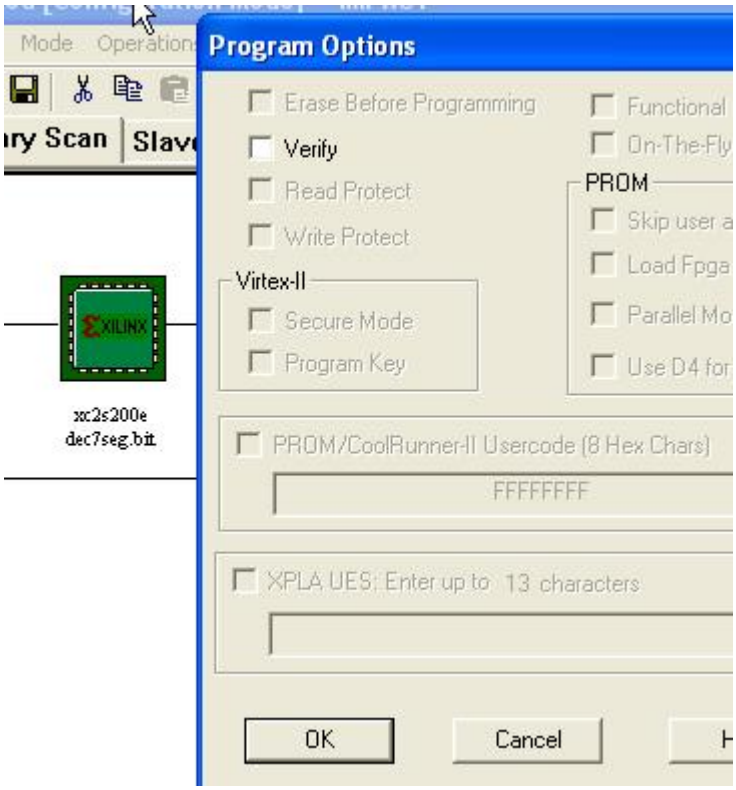
Datorita identificării greşite (un bug al aplicației IMPACT) in urma citirii fişierului (in care este codificat si tipul circuitului FPGA) va apărea si o fereastra de avertizare in care se va apasă butonul **Yes** (identificarea greşita nu va avea nici un fel de consecințe in acest caz).



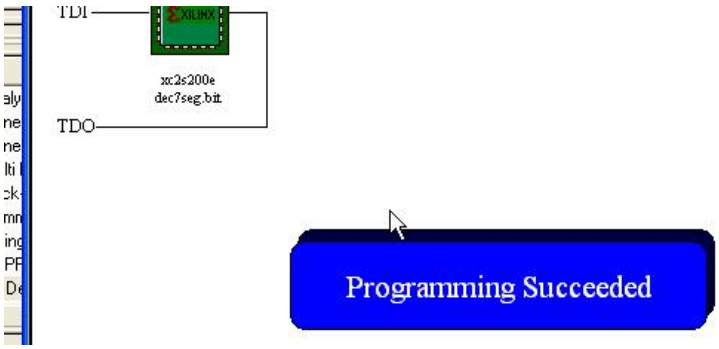
In continuare pe circuitul din fereastra (langa el apare tipul FPGA si numele fişierului bitstream) **care trebuie sa fie selectat** se va face un click dreapta. In meniul pop-up care apare se va selecta câmpul **Program**.



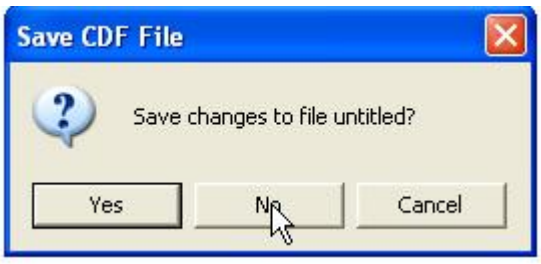
Ca rezultat se va deschide o noua fereastra **Program Options** in care se va apăsa butonul **OK**.



Dacă operația de programare a fost finalizată cu succes va apărea mesajul **Programming Succeeded**.



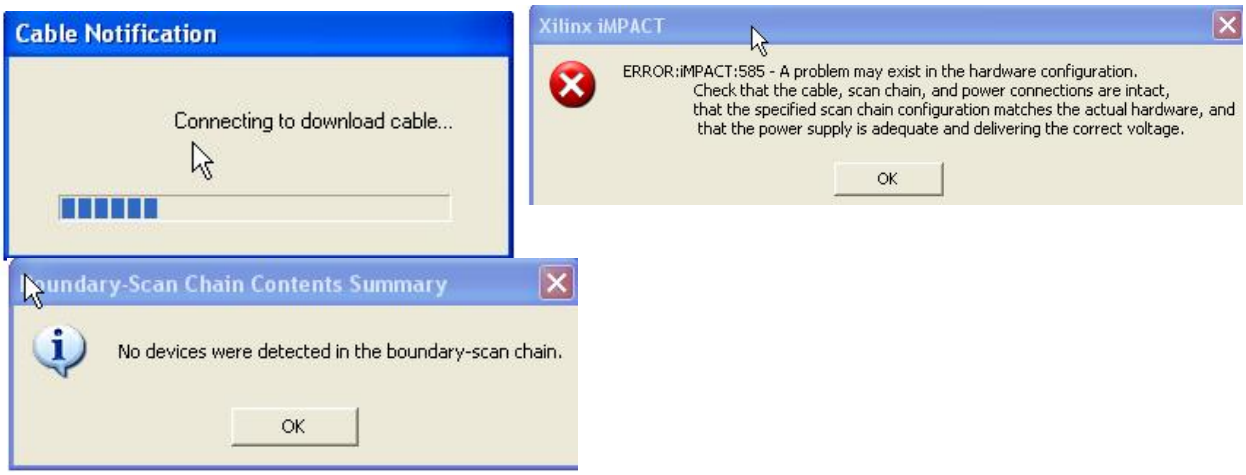
Se închide aplicația Impact cu **File-Exit** si se apasă pe butonul **No** in fereastra **Save CDF File** care apare in momentul închiderii ei.



Din acest moment aplicația trebuie sa fie funcțională!

OBSERVATIE IMPORTANTA

Fereastra inițiala **untitled [Configuration Mode]-iMPACT** apare indiferent dacă întreg lanțul de programare este funcțional sau nu (circuit alimentat, cablul de programare conectat la portul paralel al PC si conectat corect la sistemul de dezvoltare).
Dacă în momentul in care se lansează operația **Initialize Chain** apare o fereastră numită **Cable Notification** fereastră care persistă un timp îndelungat urmată de mesaje de eroare de forma celor de mai jos există o problema in lanțul de configurare. Se verifică elementele prezentate anterior (alimentări, conectări corecte) şi după înlăturarea cauzei se repetă operația.



11. Testarea aplicației

Practic imediat după terminarea operației de programare se poate trece direct la testarea aplicației.

Corespondența între mărimile de intrare ale decodificatorului nostru şi semnalele provenind de la cele 4 comutatoare este: **d0 = SW1 , d2 = SW2 , d2 = SW3 , d3 = SW4** .

Vom utiliza cele 4 comutatoare cu translatie SW1 ..SW4 (in figura este indicat SW1) pentru a genera toate combinatiile de 4 biți pentru cele 16 cifre hex de la 0 la F. Vom face pe rând**: 0000, 0001, …, 1110, 1111** şi vom verifica corectitudinea cifrei afişate (in figura este afişorul utilizat-activ).