

Proiect:

“ Calculator de

Buzunar”

*Facultatea de Automatică și Calculatoare*

*An I, grupa 30218*

*Profesor: Student:*

*Diana Spatariu Casap Dumitru*

***Cuprins***

[1.Scurt istoric calculator de buzunar 1](#_Toc419407495)

[2.Proiectare 2](#_Toc419407496)

[2.1 Use case 2](#_Toc419407497)

[2.2 Flowchart 3](#_Toc419407498)

[2.3 Schema bloc 4](#_Toc419407499)

[2.4 Unitatea de comanda și de execuție 5](#_Toc419407500)

[2.5 Descrierea componentelor și semnalelor interioare. 7](#_Toc419407501)

[3.Implementare 9](#_Toc419407502)

[3.1 Implementare în simulatorul Active HDL 9](#_Toc419407503)

[3.2Implementare pe plăcuța FPGA. 16](#_Toc419407504)

[4.Posibilități de dezvoltare ulterioară 25](#_Toc419407505)

[5.Bibliografie 26](#_Toc419407506)

# 1.Scurt istoric calculator de buzunar

*Î*n prima jumătate a secolului XX lumea științifică și tehnica a fost dominată de rigla de calcul,pe care o foloseau toți inginerii.Pentru a simplifica toate calculele ,a fost inventat ,,Calculatorul de buzunar”, care era gîndit să incapă în buzunarul de la piept.Inițial el a fost produs de William Hewlett, doar pentru a fi folosit in cadrul companiei sale,iar mai tîrziu,la 1 februarie 1972, compania Hewlett Packard l-a lansat pe piață la prețul de 395$.



Prin principiul său de funcționare simplu, calculatorul nu este aproape niciodată o [mașină Turing](http://ro.wikipedia.org/wiki/Ma%C8%99in%C4%83_Turing), ci doar o unealtă special destinată operațiilor aritmetice. Calculatoarele mecanice și electromecanice s-au utilizat intensiv în secolele XIX și mai ales XX pe domenii ca de exemplu aritmetică și matematică, inginerie de toate felurile, cercetări științifice, calculul financiar și bancar, case de încasări precum și multe altele.

Dar și calculatoarele electromecanice au fost depășite la funcționalitate, viteză, capacitate și grad de miniaturizare de către calculatoarele electronice (cu tranzistori și circuite integrate). După principiul de funcționare există următoarele tipuri de calculatoare electronice:

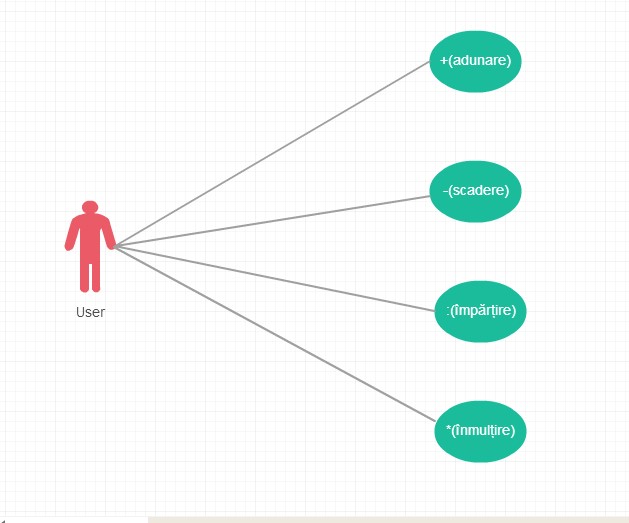
* [calculatoare analogice](http://ro.wikipedia.org/wiki/Calculator_analogic);
* calculatoare digitale (numerice); marea majoritate a calculatoarelor electronice actuale sunt digitale. Dacă tipul nu se specifică explicit, atunci e vorba întotdeauna despre calculatoare digitale.

Calculatoarele electronice de buzunar fac de obicei numai operații aritmetice, fără să fie programabile. Calculatoarele mai performante pot efectua operatii de tipul sin,cos,tg,ctg,ln,exp si altele!

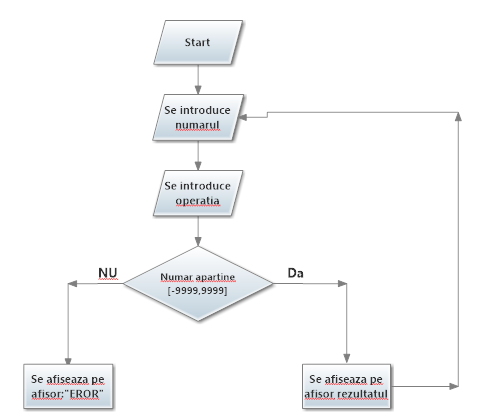
În continuare va fi prezentat un calculator simplu, proiectat in limbajul de programare hardware VHDL,care efectueaza cele 4 operatii aritmetice de bază( +,-,x,:).

# 2.Proiectare

## 2.1 Use case

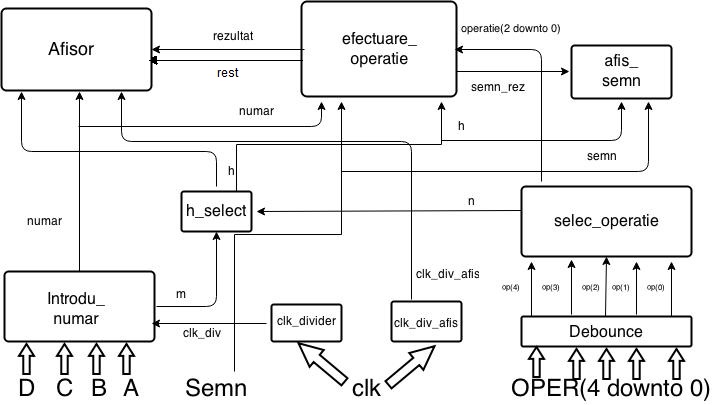


## 2.2 Flowchart

*Calculatorul de buzunar are un algoritm simplu de utilizare și funționare,care constă în introducerea a două numere și operația dorită pentru a vedea rezultatul pe ecranul calculatorului.Mai jos această simplă funcționare este descrisă printr-un flowchart.*

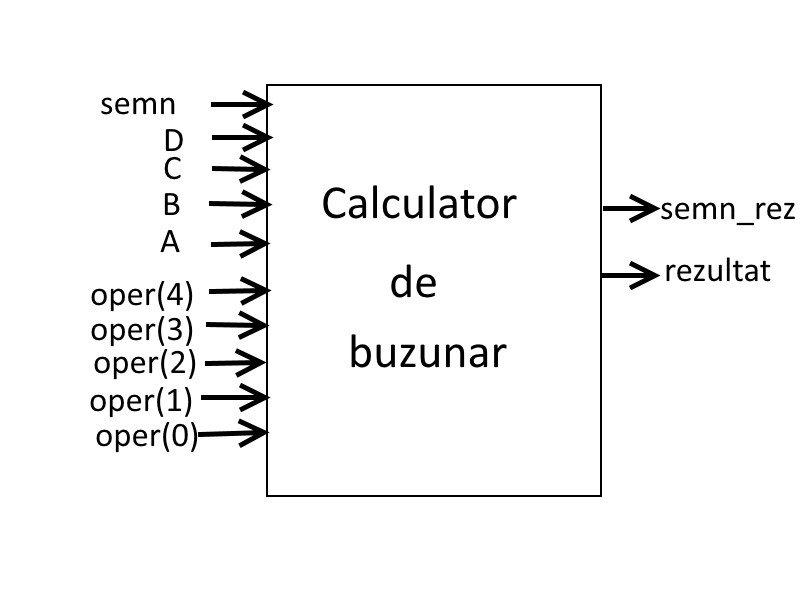
## 2.3 Schema bloc

*Dacă la prima vedere, funcționarea calculatorului pare destul de simplă, în interiorul lui această funcționare este mult mai complexă, lucru demonstrat de schimba bloc al proiectului.*

**

## 2.4 Unitatea de comanda și de execuție

În comparație cu schema bloc,prezentată mai sus, din exterior calculatorul de buzunar ar parea foarte simplu.Imaginea de mai jos redă foarte clar acest lucru.

******

În partea stîngă sunt specificate intrările calculatorului.Intrarea semn este pe un bit și este semnul numarului, care poate fi plus, in biți ‘ 0 ’ și minus, adică ‘ 1 ‘. Intrările D,C,B,A reprezintă cifra miimilor,sutimilor,zecimilor si respectiv a unităților.Celelalte cinci intrări sunt pentru selectarea operații pe care utilizatorul dorește să o efectueze:  
 oper(4)- ,, + “ oper(3)- ,, - “  
 oper(2)- ,, x “ oper(2)- ,, : “

oper(0)- ,, = “

Acestea 10 intrări stau la baza calculatorului și utilizarea lui.Ele sunt primite și procesate de catre două componente de bază: “introdu\_număr” și “selec\_operație’’ care reprezintă unitatea de comandă.Prima primește intrările D,C,B,A și formează numărul, numar=D\*1000+C\*100+B\*10+A, iar a doua selectează operația dorită, tastînd unul din butoanele: Oper(4)-Oper(0).Operația selectată este codificată pe 3 biți: 000- “+”, 001- “-“,010- ‘’x’’, 011-‘’:’’ și 100- “=”.

Avînd selectat numărul și operația dorită,urmează unitatea de execuție care este controlată de componenta “efectuare\_operație”. Are ca intrare numar,ieșirea din introdu\_numar și operatie(2 downto 0),din selec\_operatie și un semnal h, obținut din h\_select care este ’0’ dacă la moment se introduce un număr și este ‘1’ dacă a fost tastat vreo unul din butoanele pentru operație.Această componentă h\_select este controlată de semnalele m și n ieșiri din introdu\_numar și respectiv selc\_operatie.Avînd aceste 3 intrări, în efectuare\_operatie există o variabilă înglobată rez, inițializată cu zero, care rămîne neschimbată pentru h=’0’ și pentru h =’1’ se efectuează operația selectată dintre rez și numărul introdus la moment. Mai are ca intrare semn-ul numărului introdus și respectiv ca ieșire semn\_rez –semnul rezultatului. Pentru operația de împărțire există ieșirea  
rest, care va lua doar restul ultimei împărțiri efectuate.  
 Practic, acestea 3 componente reprezintă partea de bază a proiectului,ele primesc numerele introduse cu semn,operațiile pe care utilizatorul dorește să le efectueze și scot rezultatul cu semnul respectiv.După cum vedeți, pe schema logica de la 2.2(pag.4) mai există și alte componente și semnale interne,chiar dacă ar părea ca cele descrise în unitatea de comandă și execuție sunt suficiente.Explicația este că proiectul trebuie și implementat și se va lua în seamă multe lucruri caracteristice metodei de implementare.În continuare, la punctul 2.5 veți afla și înțelege mai multe despre aceasta.

## 2.5 Descrierea componentelor și semnalelor interioare.

Proiectul va fi implementat pe o placuța FPGA Nexys™3 spartan-6,care arată în felul următor.

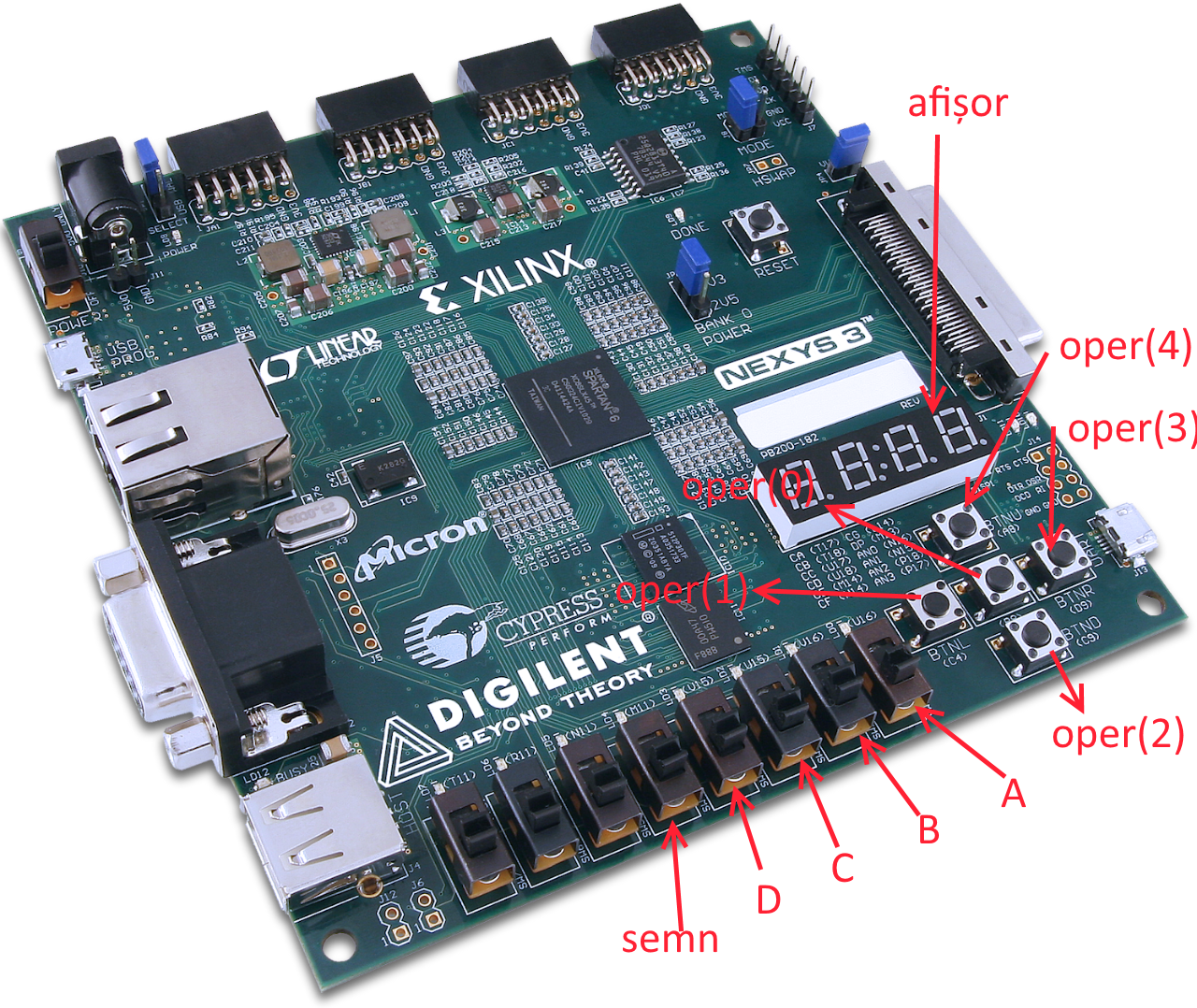


Figure 1

Această placa are un clock încorporat, care funcționează la o frecevență de 100 MHz,ceea ce este foarte rapid si nu poate fi sesizat cu ochiul uman. De aceea o componentă de bază de care avem nevoie este div,care divizează clk de pe placa de la 100 de MHz, la aproximativ 1Hz, pentru ca să sesizăm ce se înîmplă pe placă. Această divizare se face după formula

100 Mhz de la numărator reprintă frecvența clk de pe placă,iar la numitor frecvența dorită-1Hz.Rezultatul ne spune că vom avea nevoie de un numărător pînă la 100000000, iar pentru a simplifica lucrurile vom trasnforma 100 de milioane într-un vector de 24 de biți,avînd loc relația

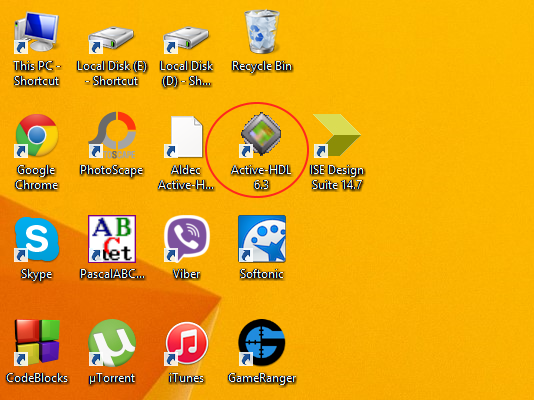
100000000~.Deci,cu ajutorul unui numărător pe 24 de biți, clk va fi divizat și noul semnal clk\_div care va avea frecventa de 1Hz și va lua valoarea ultimului bit al numărătorului și va fi folosit la componenta introdu\_număr(mai multe informații despre divizare,vedeți în bibliografie).Cînd va fi activat un switch(D,C,B sau A) la timp de o secundă cifra data va fi incrementată cu 1, iar ajungînd la 9 ea va veni înapoi în zero.Vom deconecta switchul,adica îl vom trece pe valoare logică zero,atunci cînd dorim ca aceea să fie cifra dorită.Această numarare de la 0 la 9,pe poziția miilor,sutimilor,zecimilor si unităților va fi văzută pe afisor,reprezentat pe desenul de mai sus,de atîtea ori cînd va avea loc vre-un eveniment pe una din intrările A-D,iar cînd toate vor fi active pe zero,pe afișor se va vedea numărul,care este ieșire a componentei introdu\_număr.Pe placă sunt 4 afișoare BCD-7 segmente. Există 7 intrări ’’a\_to\_g’’ pentru catozi,care va determina care dintre cele 7 segmente a afișorului să lumineze,ele fiind active pe “0”,intrarea ’’dp’’ va genera dacă va fi aprins punctul de lîngă cifra de pe afișor.Alte 4 intrări sunt pentru anozi, an(3 downto 0), care va genera care afișor să fie conectat, nu poate funcționa decît unul singur la un moment de timp,cînd un bit din vectorul an este 0.Pentru ca să putem vedea numarul întreg de 4 cifre,anozii se shiftează foarte rapid,la momente foarte mici de timp și fiecare pe rind ia valoarea “0”.Pentru ca aceasta shiftare să se facă în așa mod ca ochiul uman să vadă bine numărul întreg pe cele 4 afișoare este nevoie ca clockul de pe placă să fie divizat pîna la aproximativ 700 de Hz.Pentru aceasta vom avea nevoie de o altă componentă de divizare div\_clk\_afis care va avea ca intrare clk și ca ieșire clk\_div\_afis ,care va lua ultima valoare a unui numărător pe 16 biți,lungimea vectorului fiind determinată la fel ca mai sus.Componenta afisor va mai avea ca intrare semnalul h care a fost descris mai înainte.Dacă el va fi egal cu „0’’ pe afișor se va vedea numarul care este introdus la moment, iar contrar va fi afișat rezultatul momentat din efectuare\_operatie.   
 În comparație cu afișarea rezultatului, afișarea semnului este mult mai simplă.Vom folosi unul din ledurile de pe placă,situate mai sus de switchuri.  
Componenta afis\_semn va avea ca intrare semn , semn\_rez și h.Daca h va fi egal cu “0” va fi afișată valoarea logica a lui semn, adica dacă numărul introdus este pozitiv,ledul va fi stins,iar dacă negativa va lumina.Același lucru se va întîmpla și pentru semn\_rez,atunci cînd h va avea valoare logică “1”.  
 Un alt lucru pe care trebuie sa îl luam în considerație este funcționarea butoanelor de pe placă,care nu duc un semnal curat la apăsarea lor. De asta în scema bloc,butoanele mai întîi intră întro componentă,numită debounce. Ea constă în întîrzierea semnalului și în final duce doar unul,de care este nevoie.Pentru asta se folosesc 3 bistabile si un numărător pe 24 de biți prin care va trece semnalul trimis de pe buton ca să ajungă exact odată cu clockul divizat de pe placă.Astfel din semnalele Oper(4 downto 0) vor rezulta semnalele Op(4 downto 0).(mai multe informații despre debounce, le vedeți în bibliografie).

# 3.Implementare

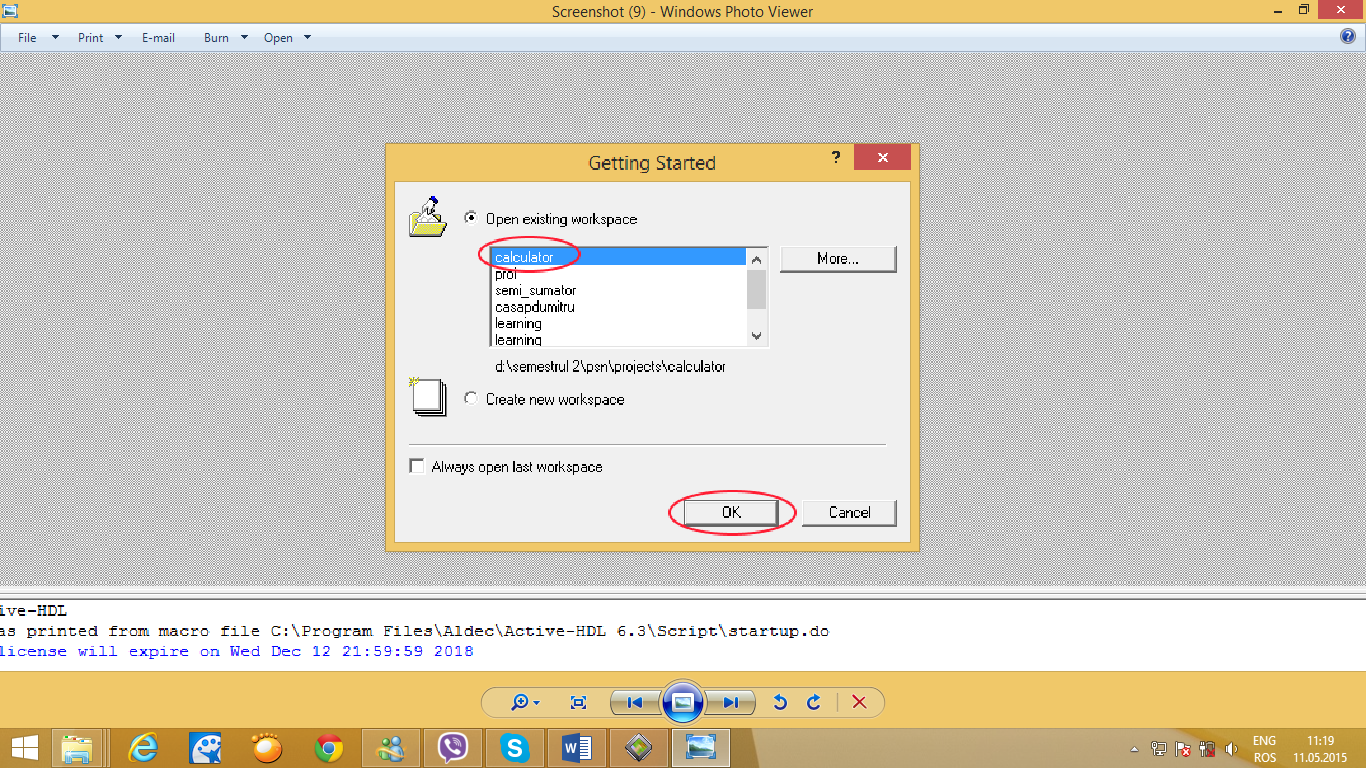
## 3.1 Implementare în simulatorul Active HDL

Codul proiectului fiind scris în Active HDL, evident el poate fi imediat și verificat.Deci pentru a efectua operații dorite și a verifica cum funcționează proiectul trebuie să instalați mai înîii Active HDL și după să urmați următorii pași:

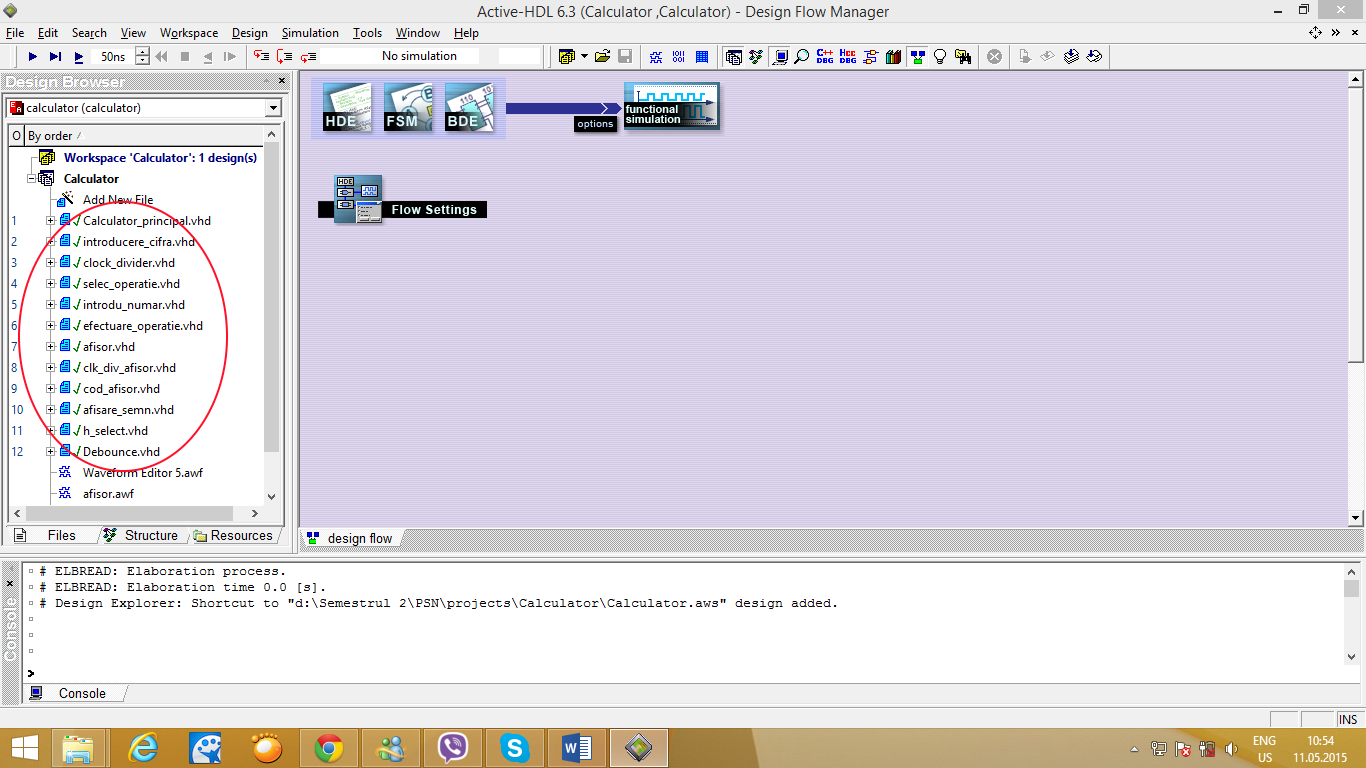
1.Deschidem Active HDL



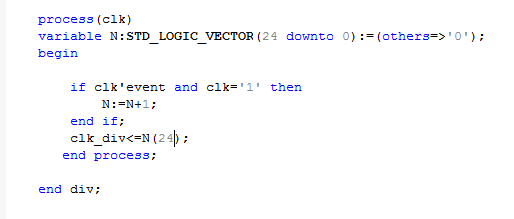
2.Selectăm proiectul calculator,pe care trebuie să îl aveți în memoria calculatorului.



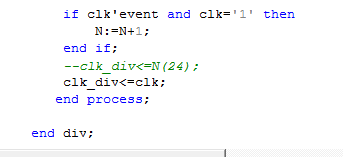
3.În partea stîngă sunt lista cu toate componentele văzute și descrise anterior.Dînd dublu-click pe oricare dintre ele, puteți vedea codul.



4.Elementele caracteristice plăcii FPGA,ca clock divider sau debounce,pe care le-am descris la capitolul 2, nu mai avem nevoie.Folosind debounce pentru OPER(4 downto 0),în simulator vom folosi direct semnelele OP(4 downto 0) pentru a evita acea componentă.Fiindcă componenta afisor nu o vom folosi,nu schimbăm nimic, iar pentru clk\_div folosit la componenta introdu\_numar,trebuie să scoatem divizarea clk.Deci,deschidem componenta clk\_divider.

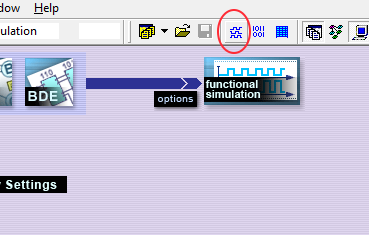


Avem acest process, si vom comenta linia clk\_div<=N(24) iar clk\_div va rămîne clk inițial.

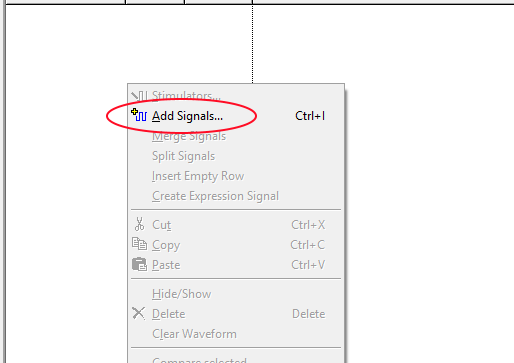


După aceasta dăm la compilare Crtl+F11

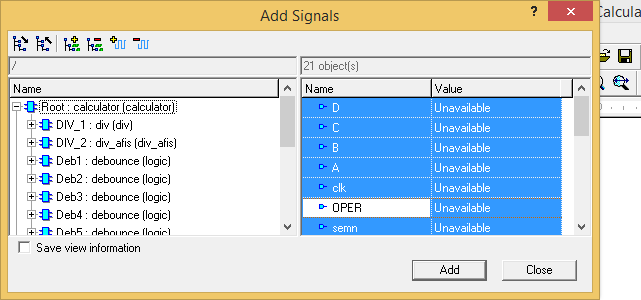
5.Tastăm new waveform



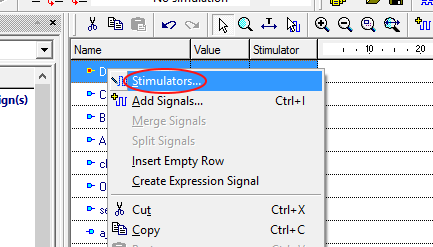
6.Alegem add signals.



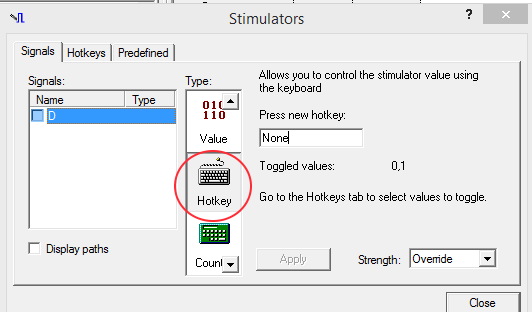
7.Selectăm toate semnalele și le adăugăm



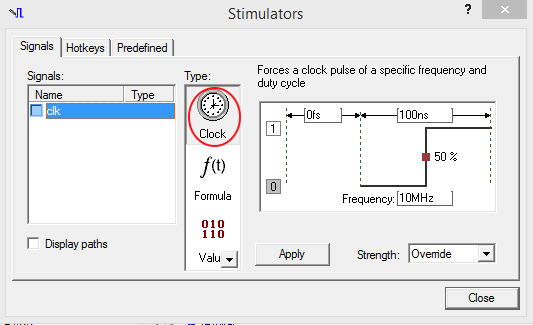
8.Selectăm stimulatorii de la tastatură, cu care vom controla intrările calculatorului.Dăm click dreapta pe semnalul care dorim să ii alegem un stimulator și tastăm stimulators.



9.Alegem hotkey și alegem un buton de pe tastatură și tastăm apply.



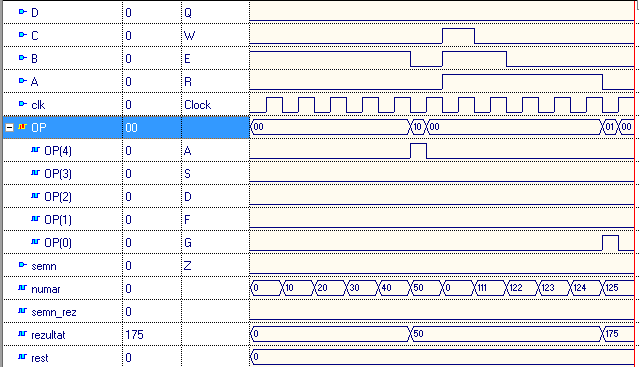
10.Pentr clk, în loc de hotkey vom alege clock și vom selecta datele asemenea desenul de mai jos.



11.După ce selectăm toți stimulatori,tastăm initialize simulation pentru a da proiectul la simulare.



12.Urmînd modul de funcționare a calculatorului,vom introduce numărul, a carui cifre vor numară , dacă este activă pe 1, odata ce tastăm F5.Dupa cifrele D,C,B,A se pun pe zero, și se selectează una din operații.După fiiecare modificare,suntem nevoiți să tastăm F5, fiindcă proiectul lucrează pe clk.Mai jos vedeți un exemplu de adunare a numerelor 50 și 125 unde puteți observa formele de undă a tuturor semnalelor.



13.Analizînd atent ce se înîmplă pe fiecare semnal e ușor de înțeles funcționarea calculatorului, iar rezultatul îl vedem pe semnalul rezultat care este la momentul dat egal cu 175.

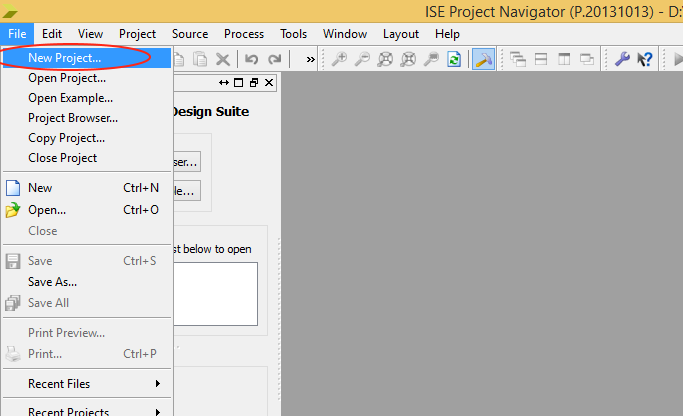
## 3.2Implementare pe plăcuța FPGA.

Este un mod de implementare mult mai elegant și mai aproape de adevărata funcționare a unui calculator de buzunar.Pentru aceasta aveți nevoie de ISE Design Suite-Xilinx și de o plăcuța FPGA Nexys™3 spartan-6.Avînd acestea două la dispoziție trebuie să urmați următorii pași:

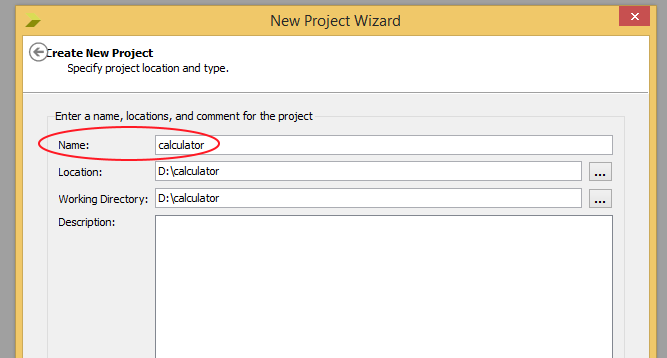
1.Deschideți ISE Design Suite-Xilinx



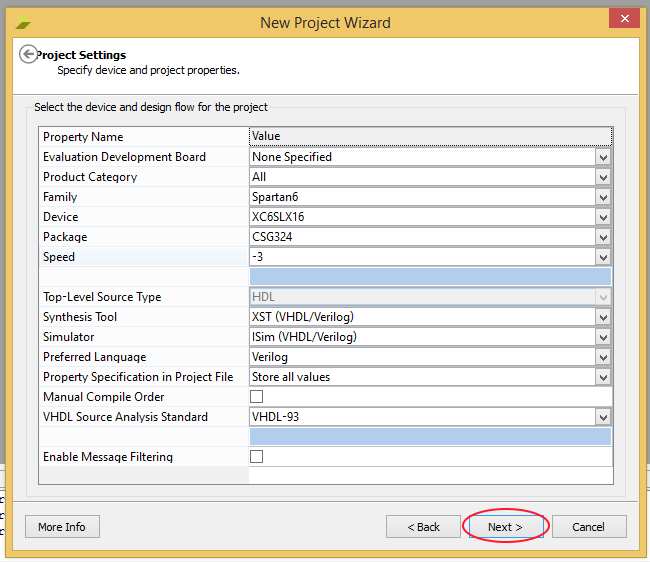
2.Selectăm New Project



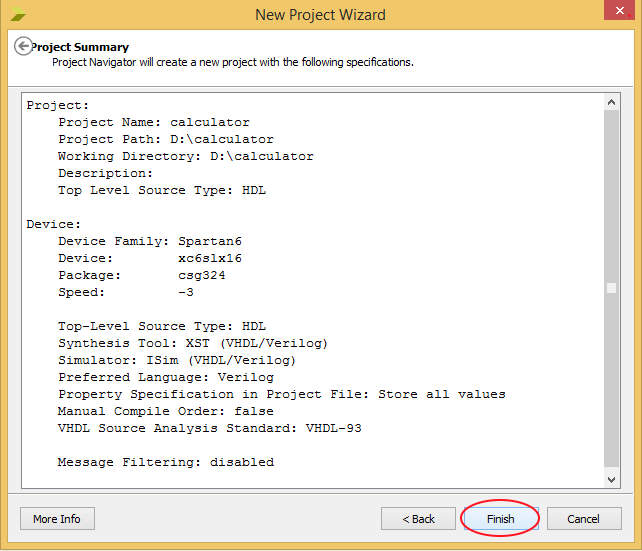
3.Dam un nume proiectului și selectăm mapă unde să îl salvăm.



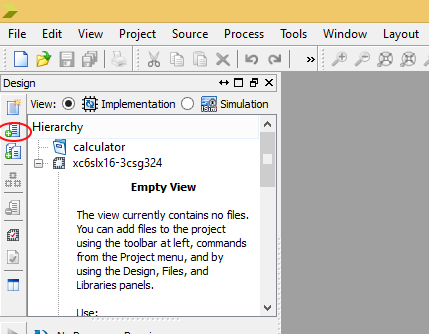
4.Alegem toate acestea setari, exact cum sunt indicate pe poză, că de altfel proiectul nu va funcționa pe placă și după tastăm next.



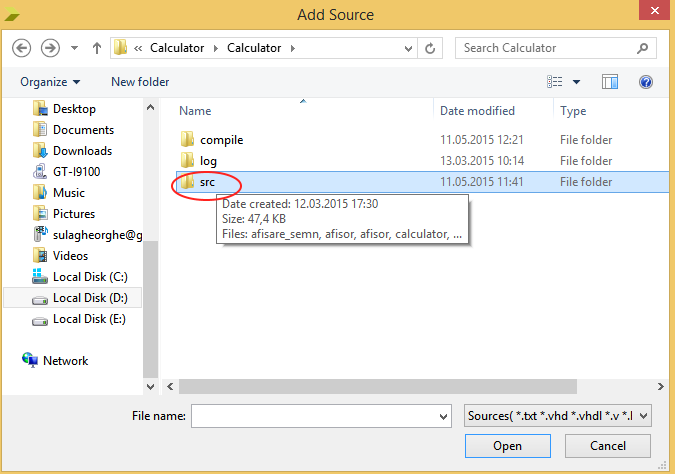
5.Click Finish



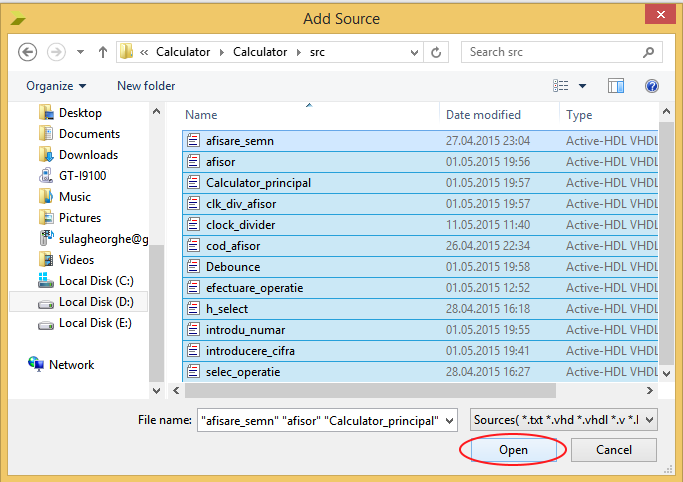
6.Acum trebuie să adăugăm proiectul, selectăm Add Source



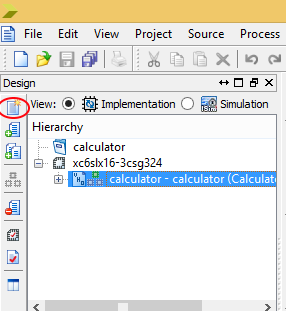
7. Cautăm mapa cu proiect,după deschidem src.



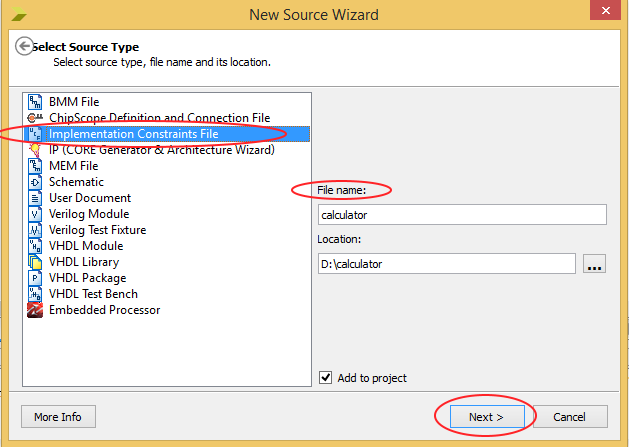
8.Selectăm toate componentele și tastă open.



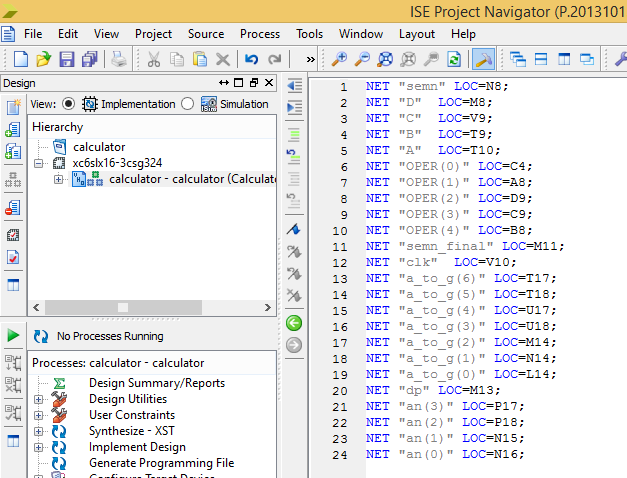
9.Acum trebuie sa facem legaturile dintre semnalele din program cu swichiurile,butoanele,clock-ul , anozii, catozii de pe placă.Selectăm new source.



10.Selectăm Implemntation Constraints File, îi dăm un nume și tastăm next.

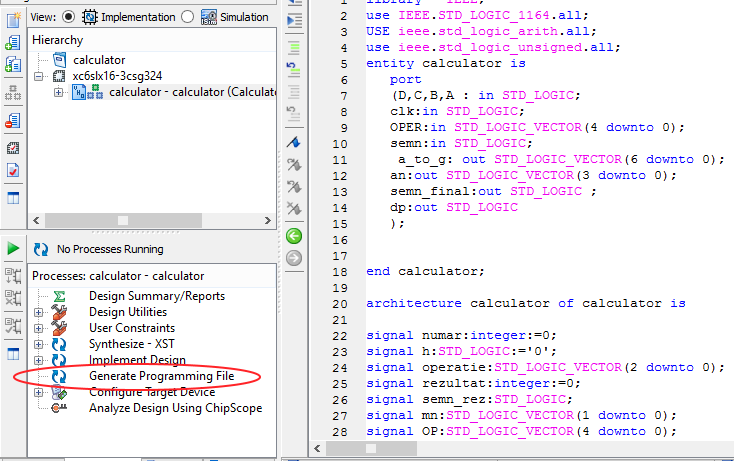


11.Acum trebuie să scrieți UCF și veți face asta exact așa cum este scris mai jos.

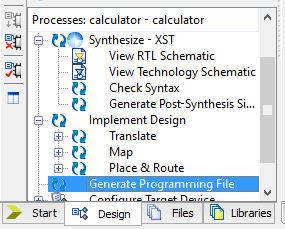


În partea stînă sunt semnalele din proiect, iar în dreapta pinii de pe placă.

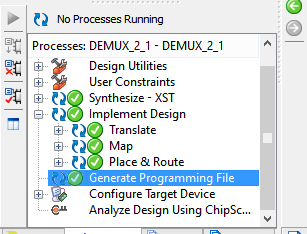
12.Dăm dublu click pe generate programing file



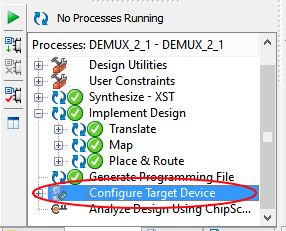
13.Așteptăm că sa se facă sintetizarea și implenentarea proiectului.El fiind volominos, va dura ceva timp, aproximativ 5 minute.



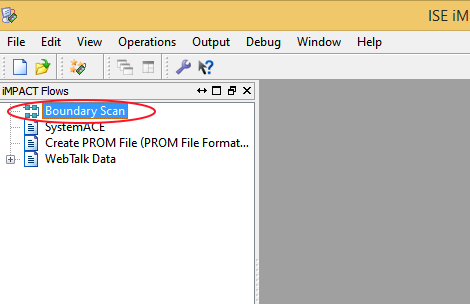
14.Dacă toate apar cu bife verzi, înseamnă că ați facut totul bine și mergeți mai departe.



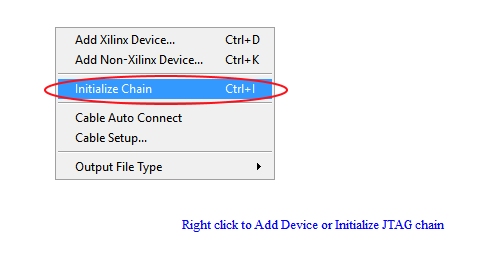
15.Mai departe tastăm Configure Target Device



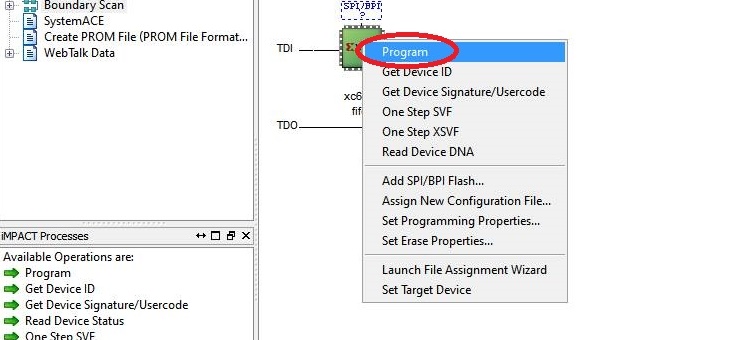
16.După alegem Boundary Scan



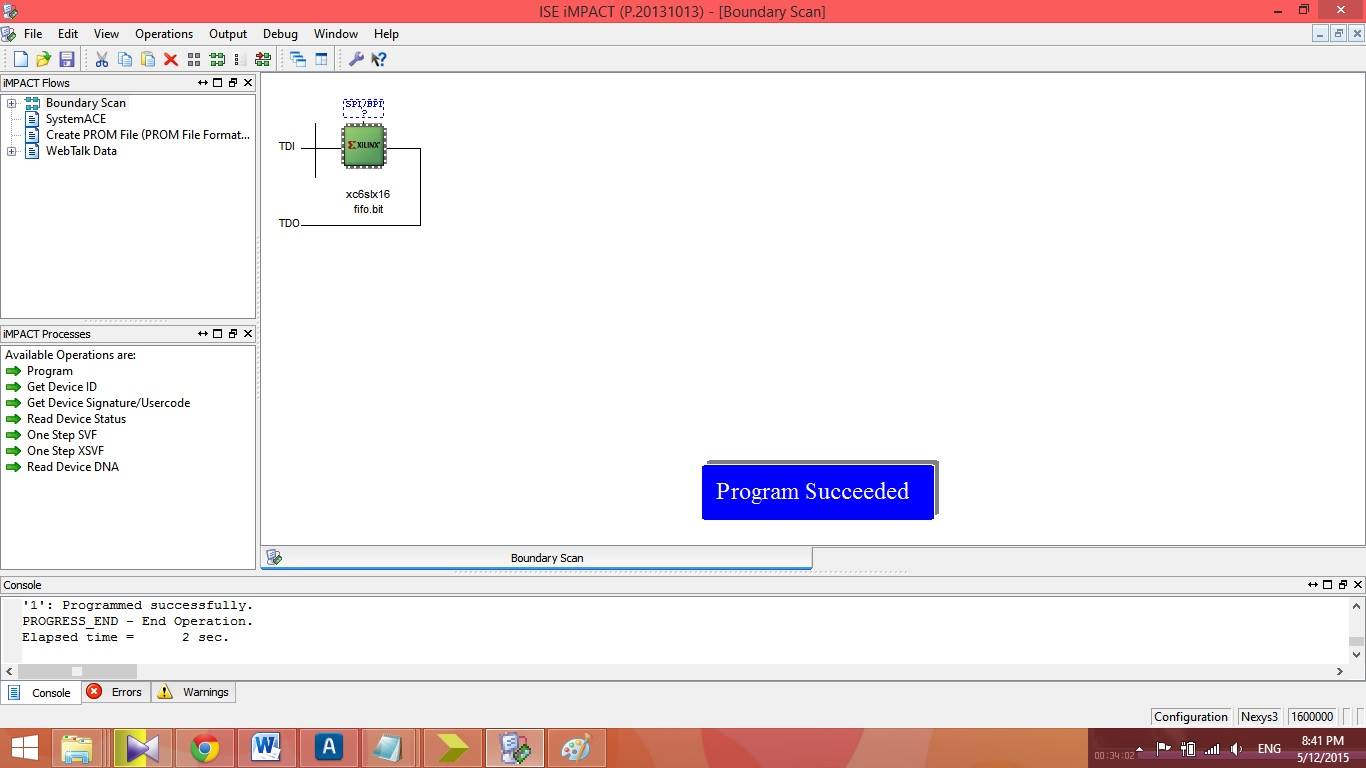
17.Conectăm placă prin USB la calculator și selectăm Initialize Chain



18.Asteptăm să se facă legatură cu placa și după dăm click dreapta porțiunea verde și alegem program.



19.După ce apare Program Succeeded, pe afișoarele plăcii vor apărea patru de zero și deja putem să intruducem numere,să efectuăm diverse operații.Vom folosi switchiurile și butoanele reprezentate în imaginea Figure 1(click pentru a o revedea) .



# 4.Posibilități de dezvoltare ulterioară

Proiectul a fost gîndit să fie cît de bun posibil,însă desigur el poate fi făcut mai bun, mai elegant, mai rapid.Deci, următoarele schimbări considerabil vor îmbunătăți proiectul:

1.În primul rînd,ar fi bine să se lucreze în proiect doar cu numerele în binar, asta l-ar face mult mai rapid, fiindcă absolut toate operațiile în binar sunt efectuate cu viteză mai mare.

2.Sa se perfecționeze operațiile cu rest,să fie mai exact și să se ieie în calcul la operațiile următoare, odată ce apare.

3.Să se mărească domeniul de valabilitate a numărului și rezultatului,care acum este de la -9999 pînă la 9999.O variantă ar fi să schimbăm baza de intruducere și de afișare a numerelor, spre exemplu în hexazecimal. Dacă vom hotărî să lucrăm în această bază, numărul maxim din 4 unități care să potă fi văzut pe afișor este FFFF, deci vom putea opera cu numere de la -65535 pînă la 65535.

4.Calculatorul să poată efectua mai mult decît doar acestea 4 operații.Să fie intruduse operațiile ca spre exemplu: radical,ridicarea la putere,logaritm,sinus,cosinus,tangentă,contagentă.

# 5.Bibliografie

*[Convert 8bit binary number to BCD in VHDL](http://stackoverflow.com/questions/23871792/convert-8bit-binary-number-to-bcd-in-vhdl)*[. 28 Mai 2014.](http://stackoverflow.com/questions/23871792/convert-8bit-binary-number-to-bcd-in-vhdl)

[*Debounce Logic Circuit (with VHDL example)*. 21 Martie 2013.](https://www.eewiki.net/display/LOGIC/Debounce+Logic+Circuit+(with+VHDL+example))

[*Making a clock divider*. 31 Octombrie 2013.](http://stackoverflow.com/questions/19708301/making-a-clock-divider)

Richard E. Haskell, Darrin M. Hanna. "Digital Design Using Diligilent FPGA Boards VHDL/Active-HDL Edition." 2010. 392.