Sisteme cu F. P. G. A. și D. S. P.

– Implementarea unui numărător binar în mediul Matlab – Simulink –

– Prin intermediul generatorului automat de cod Xilinx System Generator – [1]

I. INTRODUCERE:

Circuitele logice sau digitale cu rol de numărare, sunt utilizate în diverse aplicații precum: generarea semnalelor modulate în lățime, măsurarea frecvenței, măsurarea vitezei, numărarea obiectelor, contorizarea unor fenomene sau evenimente etc.

Implementarea unor astfel de circuite poate fi realizată în mod fizic (eng. hardware), cu ajutorul componentelor discrete (ex. porți logice, bi-stabile, memorii etc.), sau pe baza elementelor din componența unui nucleu FPGA.

În vederea implementării circuitelor logice (digitale) pe baza unui nucleu de tip FPGA (eng. Field Programmable Gate Array), este necesar să se cunoască cel puțin una dintre metodele de reconfigurare și programare fizică (hardware). Există deci trei astfel de procedee:
- programare sau reconfigurare pe bază cod Verilog sau VHDL;
- programare sau reconfigurare pe bază de diagrame bloc în format IP Integrator (Vivado);
- programare sau reconfigurare pe baza modelului Matlab – Simulink (System Generator);

În continuare, se va aborda metoda de programare sau reconfigurare pe baza modelului Matlab – Simulink utilizând generatorul automat de cod Xilinx System Generator. Pentru a evidenția modul de funcționare, se va utiliza platforma de dezvoltare Basys 3 cu nucleu FPGA Artix-7 și componentele instalate pe placă (ex. indicatori LED, butoane, afișaj.)

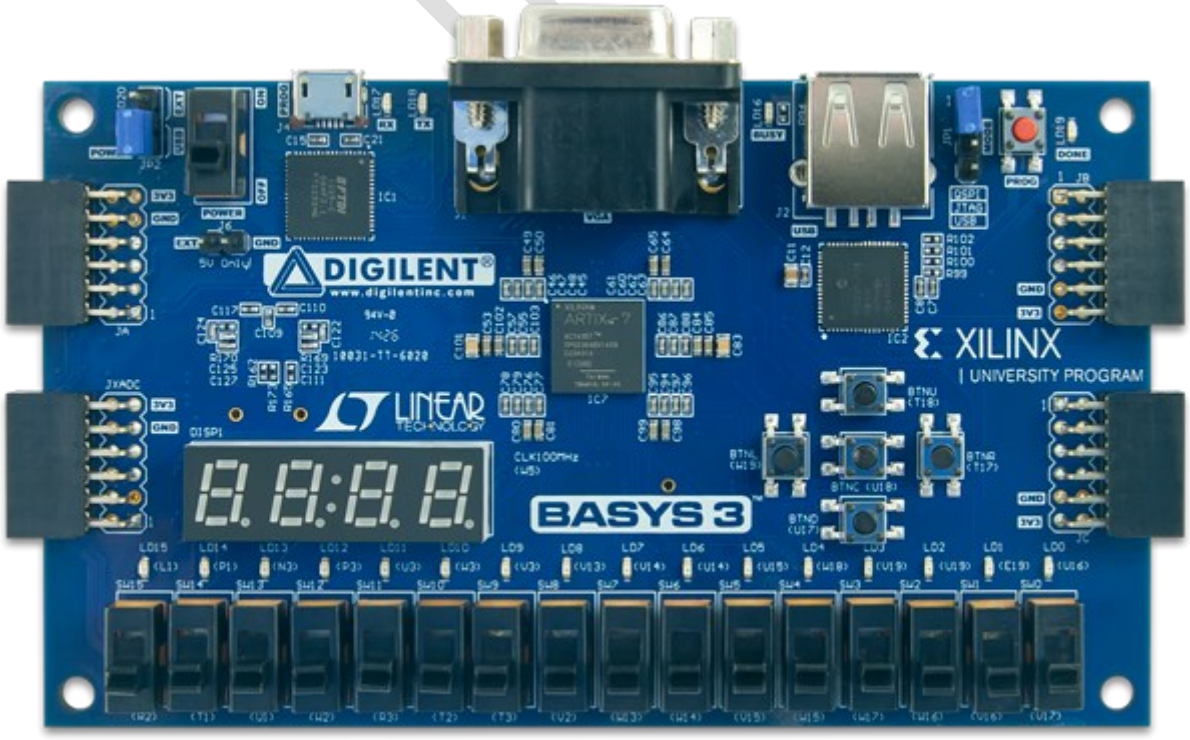


Fig. 1 - Platforma de dezvoltare Digilent Basys 3 – FPGA Xilinx Artix – 7 [2]

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

II. IMPLEMENTARE:

Odată cu instalarea pachetului de programe Vivado Design Suite – System Edition, pe lângă componentele aferente mediului Vivado Design Suite și Vivado SDK, se va instala și paleta de instrumente „Xilinx Blockset” în cadrul mediului Matlab – Simulink prin intermediul utilitarului de gestiune al generatorului automat de cod Xilinx System Generator (acr. XSG).

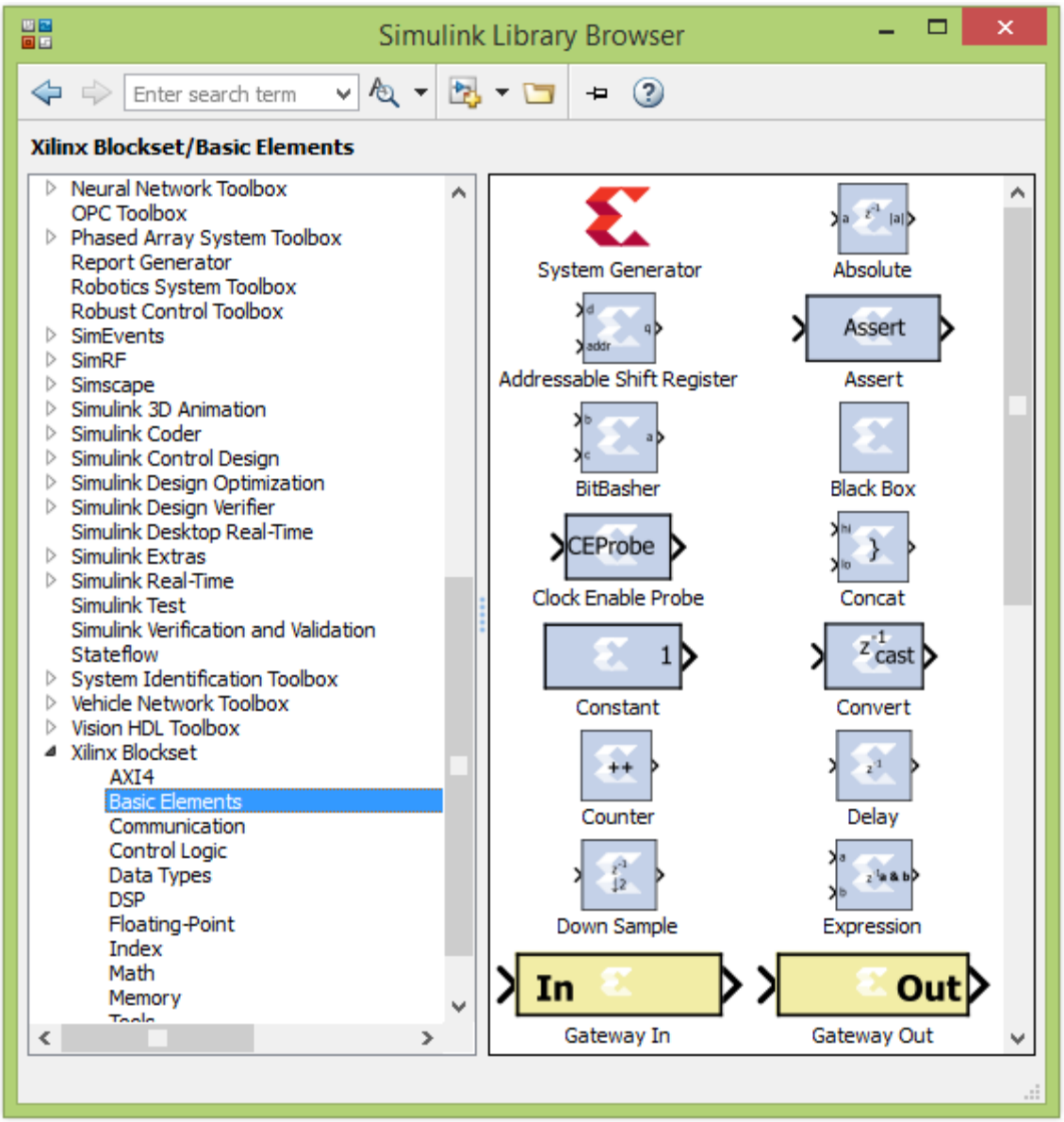


Fig. 2 – Paleta de instrumente „Xilinx Blockset” aferentă generatorului automat de cod Xilinx System Generator

În cadrul paletei de instrumente „Xilinx Blockset”, există o serie de blocuri, utile în procesul de dezvoltare al unei aplicații pentru un nucleu de tip FPGA. Spre exemplu, printre blocurile aferente categoriei „Control Logic” sunt incluse și funcțiile logice primitive (ex. ȘI, SAU, NEGARE etc.). Utilizarea acestor funcții în cadrul modelului Matlab - Simulink, conduce

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

la implementarea fizică a unor porți logice (grupări de porți), la nivelul capsulei nucleului FPGA. Astfel, pe baza elementelor din paleta de instrumente „Xilinx Blockset”, se pot implementa circuite logice (digitale) la nivel de FPGA. Prin urmare, pe baza acestor elemente, se va implementa modelul de numărător binar la nivel de nucleu FPGA.

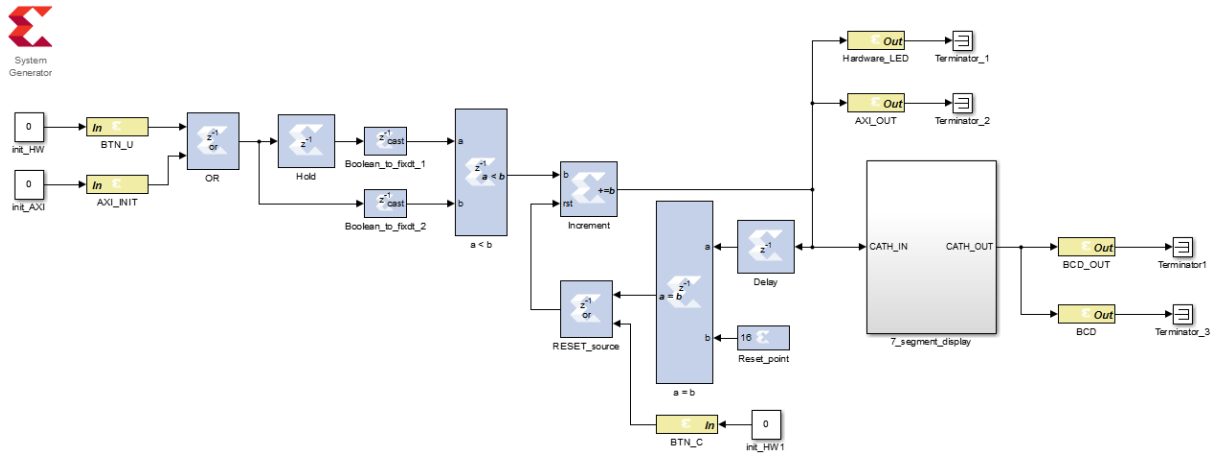
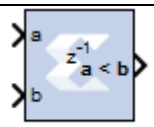
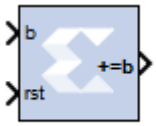
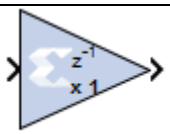
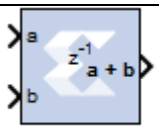


Fig. 3 – Modelul Matlab – Simulink – System Generator al numărătorului

În cadrul modelului întocmit se regăsesc următoarele blocuri:

Simbol bloc	Denumire bloc	Categorie / Subcategorie
System Generator	System Generator	Xilinx Blockset – Basic Elements
1 Constant	Constant	Simulink – Sources
In Gateway In	Gateway In	Xilinx Blockset – Basic Elements
Out Gateway Out	Gateway Out	Xilinx Blockset – Basic Elements
Terminator	Terminator	Simulink – Sinks
z⁻¹ or Logical	Logical	Xilinx Blockset – Control Logic
z⁻¹ Delay	Delay	Xilinx Blockset – Basic Elements
z⁻¹ cast Convert	Convert	Xilinx Blockset – Basic Elements

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

	Relational	Xilinx Blockset – Control Logic
	Accumulator	Xilinx Blockset – Math
	Constant	Xilinx Blockset – Basic Elements
	CMult	Xilinx Blockset – Math
	AddSub	Xilinx Blockset – Math

Principiul de funcționare al modelului implementat se bazează pe următoarele etape:
- achiziționare semnalului de la o intrare digitală furnizat de un buton sau un senzor digital;
- determinarea frontului crescător al semnalului digital (eng. rising edge detection);
- acumularea prin incrementare a numărului de impulsuri și stocarea acestuia în memorie;
- revenirea la starea inițială în momentul atingerii pragului maxim, sau acționării unui buton;

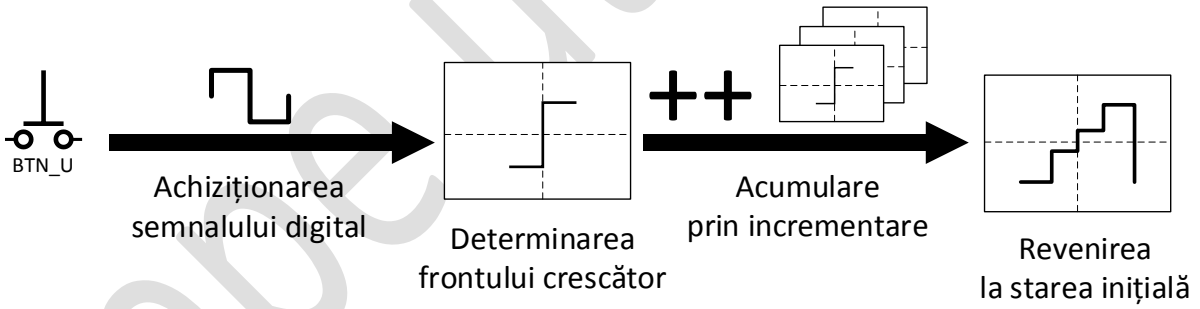


Fig. 4 – Principiul de funcționare al numărătorului

Astfel, prin apăsarea repetată a unui buton cu acțiune momentană, are loc incrementarea unei variabile, stocată într-o anumită zonă de memorie. Incrementarea variabile se va realiza până când condiția de re-inițializare nu este îndeplinită. Condiția de re-inițializare (eng. reset), poate fi indeplinită atât prin apăsarea unui buton fizic, cât și prin atingerea pragului maxim (ex. în cazul de față, valoarea 16).

Pentru a implementa funcția de numărător la nivelul platformei de dezvoltare Basys 3 se vor utiliza elementele de intrare și ieșire din componența plăcii. Simbolurile elementelor și denumirile terminalelor atașate la nucleul FPGA sunt inscripționate pe mascajul fotolitografic al platformei de dezvoltare (stratul de lac albastru).

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

Denumire element	Simbol element	Terminal atașat la nucleul FPGA
Buton cu apăsare și revenire	BTNU	T18
Buton cu apăsare și revenire	BTNC	V18
Indicator LED	LD0	U16
Indicator LED	LD1	E19
Indicator LED	LD2	U19
Indicator LED	LD3	V19
Segment de afișare „a”	CA	W7
Segment de afișare „b”	CB	W6
Segment de afișare „c”	CC	U8
Segment de afișare „d”	CD	V8
Segment de afișare „e”	CE	U5
Segment de afișare „f”	CF	V5
Segment de afișare „g”	CG	U7
Segment de afișare „V7”	DP	V7
Alimentare modul afișare 3	AN3	W4
Alimentare modul afișare 2	AN2	V4
Alimentare modul afișare 1	AN1	U4
Alimentare modul afișare 0	AN0	U2
Comutator bipozițional	SW0	V17
Comutator bipozițional	SW1	V16
Comutator bipozițional	SW2	W16
Comutator bipozițional	SW3	W17

Modul de funcționare al aplicației la nivelul platformei Basys 3 presupune:
- incrementarea variabilei rezultante în urma acționării repetate a butonului momentan;
- re-inițializarea variabilei rezultante la apăsarea unui alt buton momentan;
- aprinderea secvențială a indicatorilor luminoși la fiecare etapă de incrementare;
- afișarea valorii zecimale pe afișajul cu 7 segmente la fiecare etapă de incrementare;
- activarea modulelor de afișare în funcție de starea comutatoarelor bipoziționale;
- redarea în calculator (în modelul Simulink) a rezultatelor de simulare în timp real;

Pentru a prelua semnalul digital de la butonul fizic „BTNU”, în cadrul blocului de configurare al intrării „BTN_U” în cadrul categoriei „Basic” se vor alege următoarele opțiuni:
- Output Type: Boolean;
- Sample Time: 1e-4;

Iar în cadrul categoriei „Implementation” se vor alege următoarele opțiuni:
- Constraints – IOB timing constraint: „None”;
- IOB pad locations: {'T18'};
- IO Standards: {'LVCMOS33'};

Prin specificarea parametrilor „{'T18'}” și „{'LVCMOS33'}”, se vor impune constrângerile fizice pentru modelul dat (adică asocierile dintre terminalele capsulei FPGA și elementele fizice de pe placă). De asemenea, pe partea de intrare a modelului Simulink, se va introduce un al doilea bloc de configurare, anume „AXI_INIT”, pentru a permite modelului implementat la nivelul platformei să poată să preia informații despre condițiile inițiale de la

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

calculatorul gazdă prin intermediul interfeței de comunicare jTAG + AXI4-Lite (eng. Advanced eXtensible Interface). Comunicația dintre calculator și aplicație este deci bidirecțională.

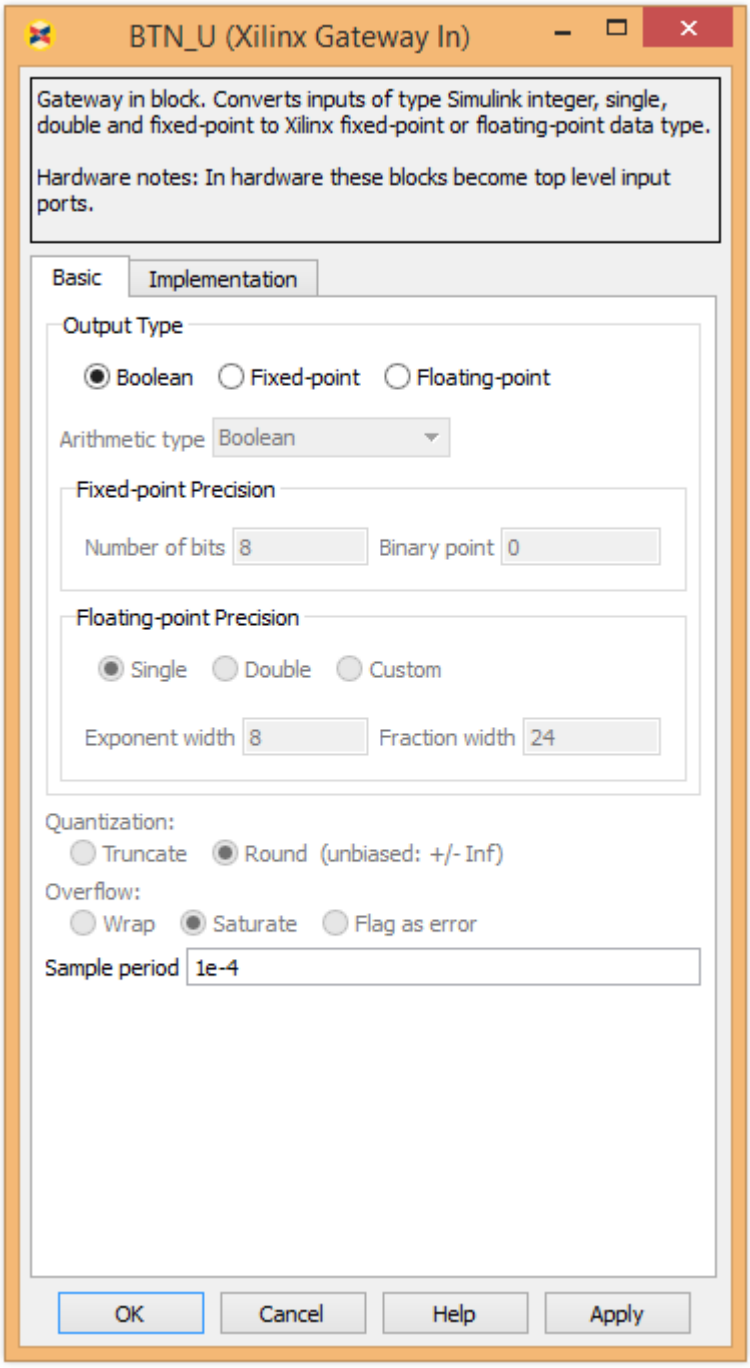


Fig. 5 – Categoria „Basic” din cadrul blocului de configurare „BTN_U”

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

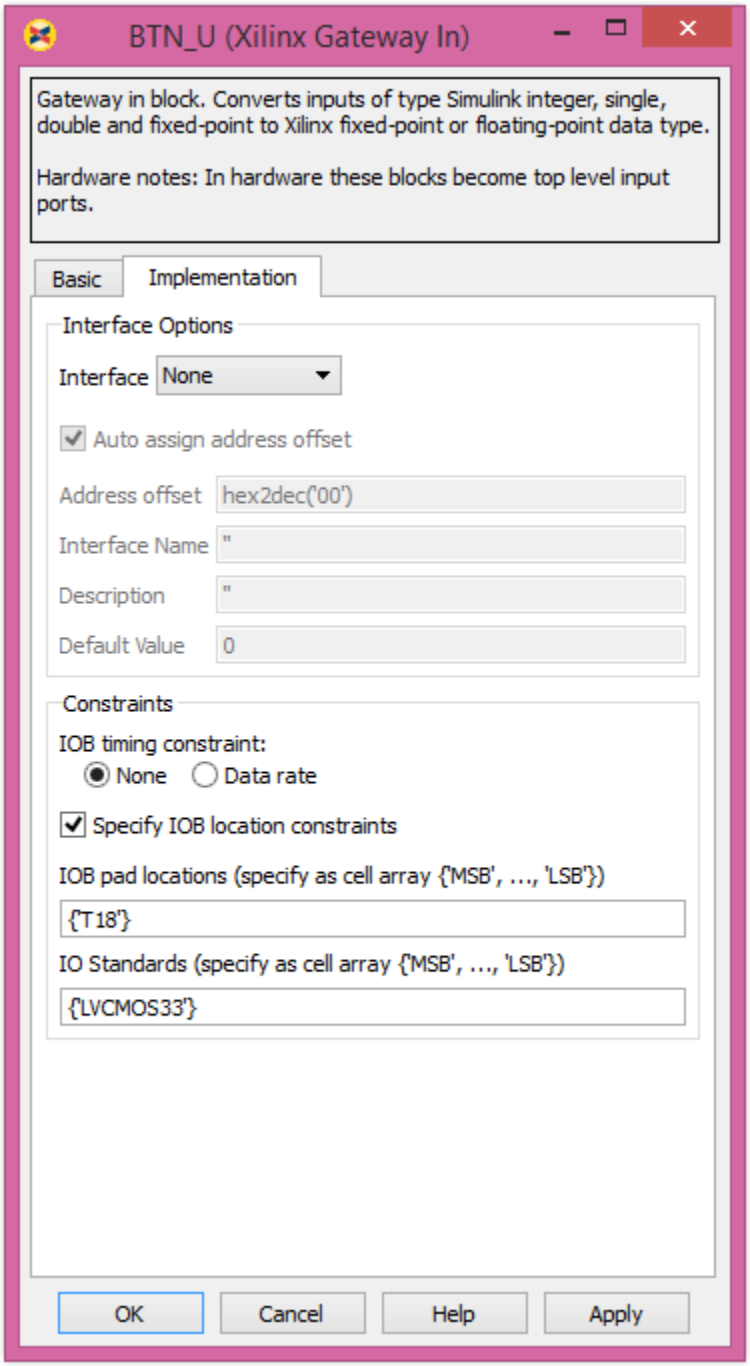


Fig. 6 – Categoria „Implementation” din cadrul blocului de configurare „BTN_U”

În cadrul blocului „AXI_INIT”, în categoria „Basic” se vor alege următoarele opțiuni:
- Output Type: Boolean;
- Sample Time: 1e-4;

Iar în cadrul categoriei „Implementation” se vor alege următoarele opțiuni:
- Interface Options – Interface: „AXI4-Lite”;
- Auto assign address offset – bifat;

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

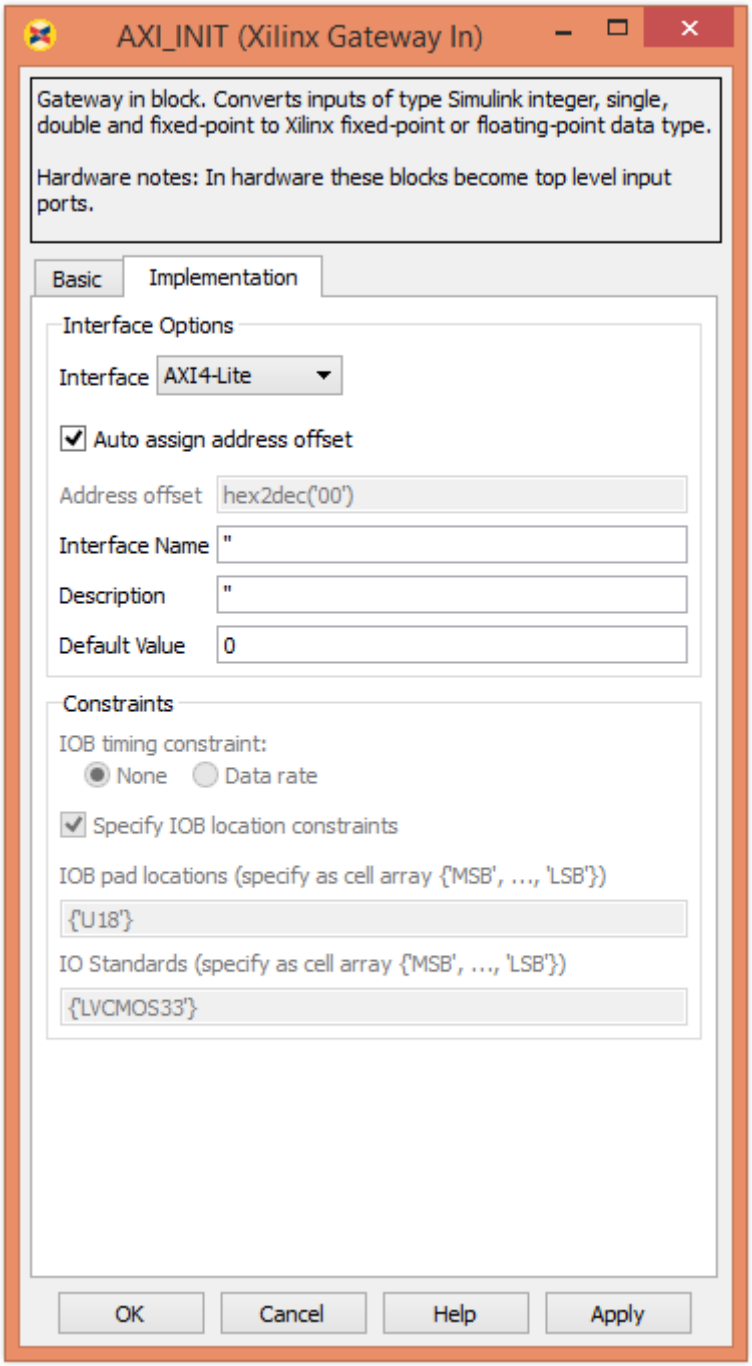


Fig. 7 – Categoria „Implementation” din cadrul blocului de configurare „AXI4-Lite”

Ambele semnale vor intra într-o poartă logică de tip „SAU” (eng. OR), care permite trecerea a cel puțin unui semnal care are valoare adevărată (fie din prima intrare, fie din a doua intrare). O astfel de poartă / funcție logică, permite suprapunerea mai multor semnale.

Semnalului rezultant (în urma funcției „SAU”) îi va fi aplicată funcția de întârziere „Delay”, pentru a permite implementarea algoritmului de determinare a frontului crescător (eng. rising edge detection). Algoritmul de determinare a frontului crescător constă în evaluarea stării inițiale și finale a semnalului digital. Dacă starea inițială a semnalului este „0” (zero logic) iar starea finală este „1” (unu logic), atunci a fost determinat frontul crescător.

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

Evaluarea stărilor (inițială sau finală), se va realiza pe baza unui bloc comparator (ex. ralational operator), care compară dacă semnalul în starea inițială „a” este mai mic decât același semnal înregistrat în starea finală „b” (a < b). Blocul comparator acceptă valori numerice finite, dar nu acceptă în schimb valori de adevăr (adevărat sau fals) sau tip de date „boolean”. Din acest motiv, au fost utilizate două blocuri pentru conversia tipului de date „boolean” la număr întreg fără semn (negativ sau pozitiv), având reprezentarea pe opt biți (valoarea maximă este 255).

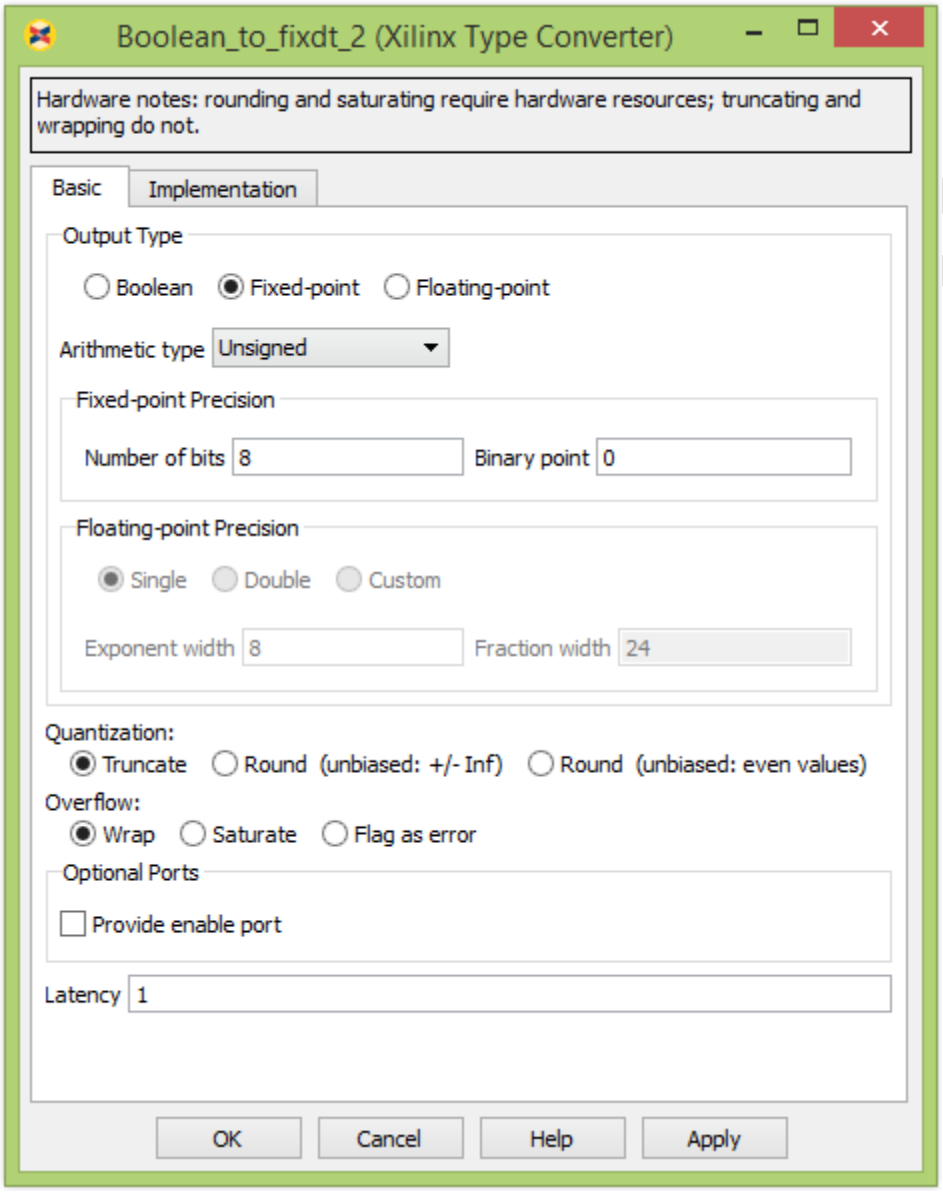


Fig. 8 – Configurarea blocului de conversie numerică

Și pentru blocul comparator, se va specifica tipul de date rezultant, ca și „Unsigned Fix” (număr întreg fără virgulă mobilă, având reprezentarea pe un număr finit de biți).

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

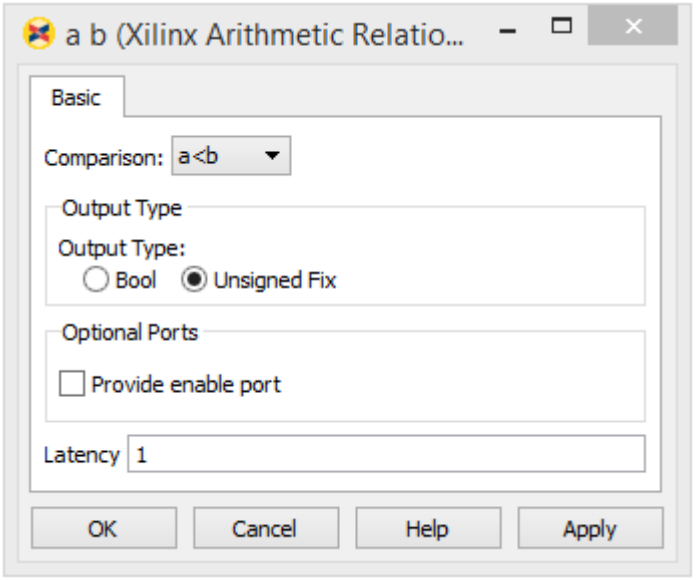


Fig. 9 – Stabilirea tipului de date pentru comparator

Pentru a contoriza numărul de impulsuri, se va utiliza un bloc de tip acumulator cu funcție de incrementare. Rolul unui astfel de bloc, este de a permite înregistrarea sau contorizarea „numărului de evenimente" care au avut ca și rezultat generarea unor semnale digitale, furnizate la intrarea blocului (acumulator). Impulsurile furnizate la intrarea „b" a blocului sunt memorate într-un registru de stocare, iar numărul de impulsuri se va incrementa la fiecare apariție a unui semnal nou la intrarea „b". Procesul de incrementare poate fi limitat de gradul de reprezentare al mărimii numerice de la ieșire (adică de numărul de biți stabilit pentru a reprezenta mărimea numerică – în cazul de față se vor alege patru „4" biți). De asemenea, procesul de incrementarea, mai poate fi limitat și printr-o comandă externă, adică prin furnizarea unui impuls (sau semnal digital) la intrarea „rst" a blocului acumulator.

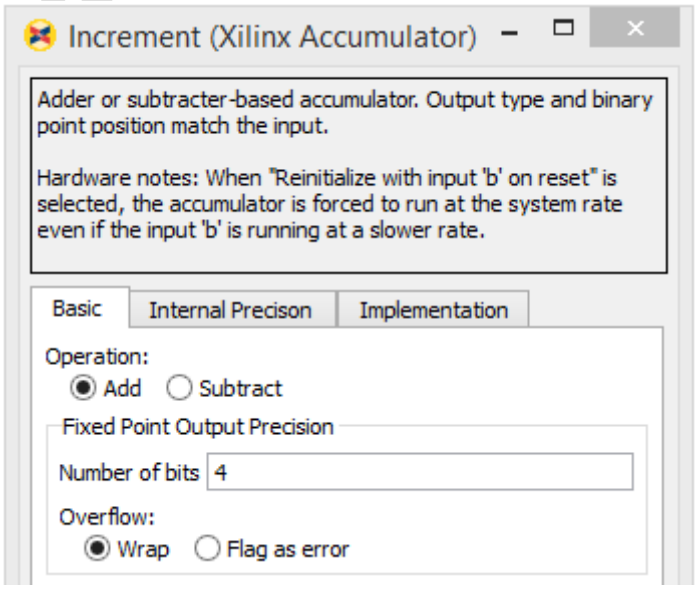


Fig. 10 – Limitarea gradului de reprezentare a mărimii numerice de la ieșire

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP"
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

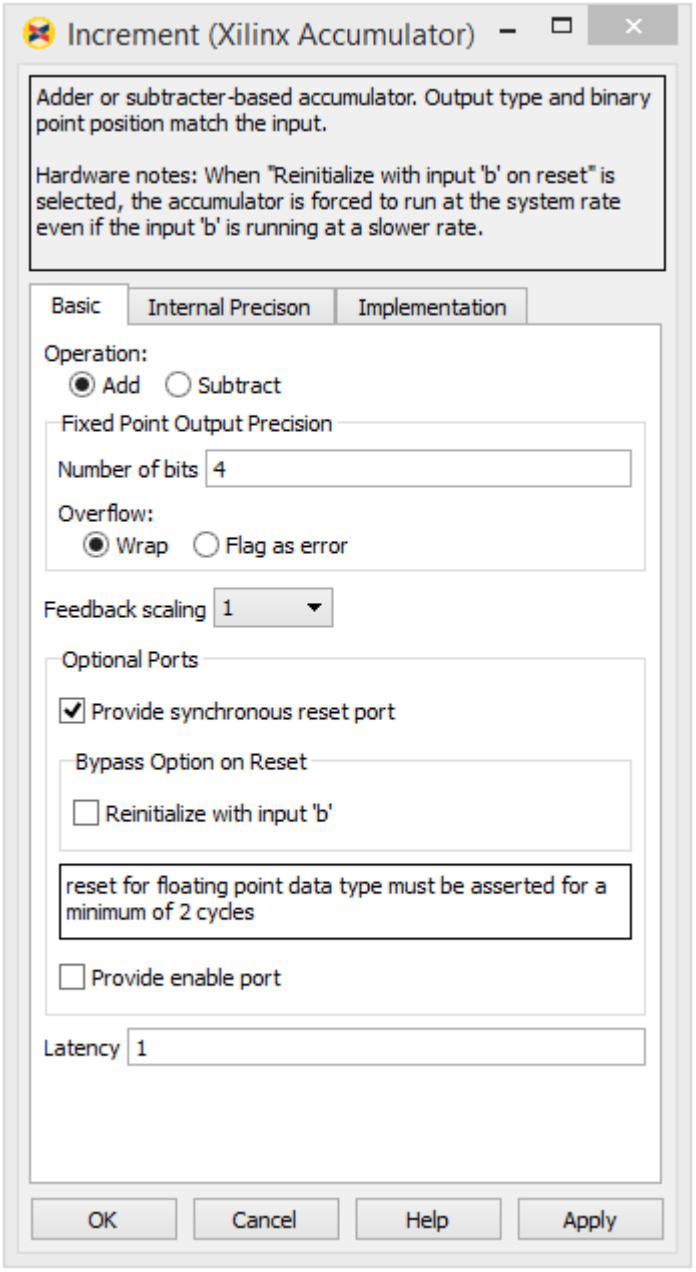


Fig. 11 – Configurarea blocului „Accumulator”

Conform schemei bloc, redată de modelul Simulink, funcția de resetare (limitare a funcției de incrementare) poate fi apelată în două moduri:
- prin atingerea la ieșire a valorii „16” stabilită prin intermediul constantei „reset point”;
- prin apăsarea butonului fizic BTNC;

Blocul de condiționare „a = b” permite verificarea faptului că, semnalul de la ieșire atinge valoarea „16”. În condițiile respective, se va genera un impuls pentru intrarea „rst”. Pentru a combina ambele metode de resetare, se va utiliza o poartă logică de tip „SAU” (eng. OR), anume poarta „RESET_source”. Funcția deci va avea loc, dacă cel puțin una dintre condiții este adevărată, adică, generează un semnal logic activ (logic „1”).

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

Semnalul rezultant, va fi un semnal de tip „dinte de fierastrău” în trepte. Acest semnal poate fi re-direcționat înspre un grup de patru indicatori LED, care la rândul lor ar reprezenta, patru biți. Astfel, prin intermediul celor patru indicatori LED, se vor urmări toate combinațiile binare, necesare pentru a descrie valoarea numerică a mărimii de ieșire pe patru biți.

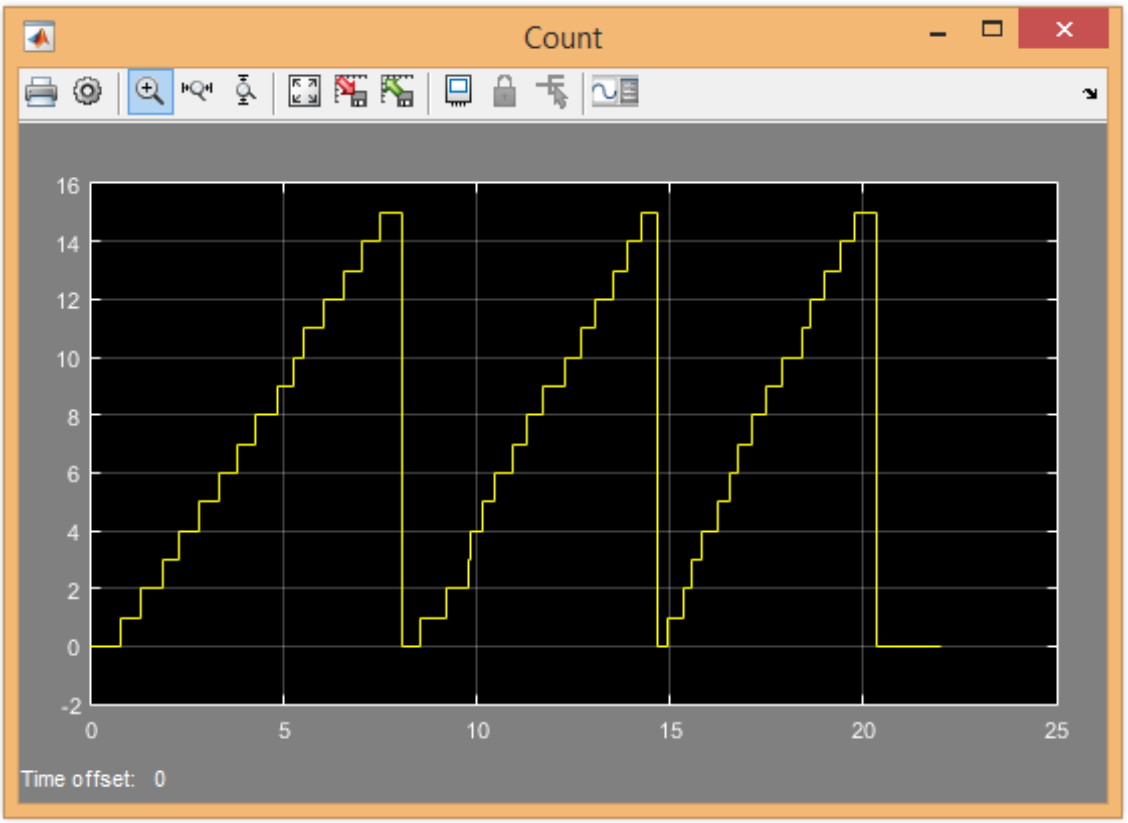


Fig. 12 – Semnalul rezultant din cadrul algoritmului de incrementare

De asemenea, pe baza unui tabel de adevăr (sau de identificare – eng. look-up table), se vor genera combinațiile necesare pentru a afișa valoarea numerică zecimală prin intermediul unui afișaj cu șapte segmente.

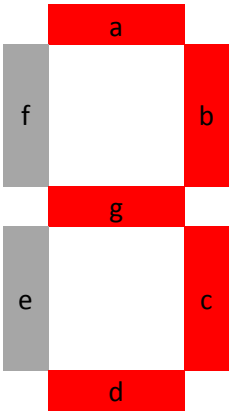


Fig. 13 – Structura unui modul de afișare cu șapte segmente

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

hex	a	b	c	d	e	f	g	Z
0	0	0	0	0	0	0	1	1
1	1	0	0	1	1	1	1	79
2	0	0	1	0	0	1	0	18
3	0	0	0	0	1	1	0	6
4	1	0	0	1	1	0	0	76
5	0	1	0	0	1	0	0	36
6	0	1	0	0	0	0	0	32
7	0	0	0	1	1	1	1	15
8	0	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0	4
A	0	0	0	1	0	0	0	8
B	1	1	0	0	0	0	0	96
C	0	1	1	0	0	0	1	49
D	1	0	0	0	0	1	0	66
E	0	1	1	0	0	0	0	48
F	0	1	1	1	0	0	0	56

Platforma de dezvoltare Basys 3 are în dotare patru module de afișare pe șapte segmente. Alimentarea acestora se realizează prin intermediul unor tranzistoare bipolare, comutate de către pinii „W4”, „V4”, „U4” și „U2” aflați la periferia capsulei FPGA. Alimentarea segmentelor de afișare se realizează în conexiune de tip anod comun, iar catodul fiecărui segment de afișare este conectat de asemenea la câte un pin situat la periferia carcasei nucleului FPGA, anume: „W7”, „W6”, „U8”, „V8”, „U5”, „V5”, „U7”, „V7”.

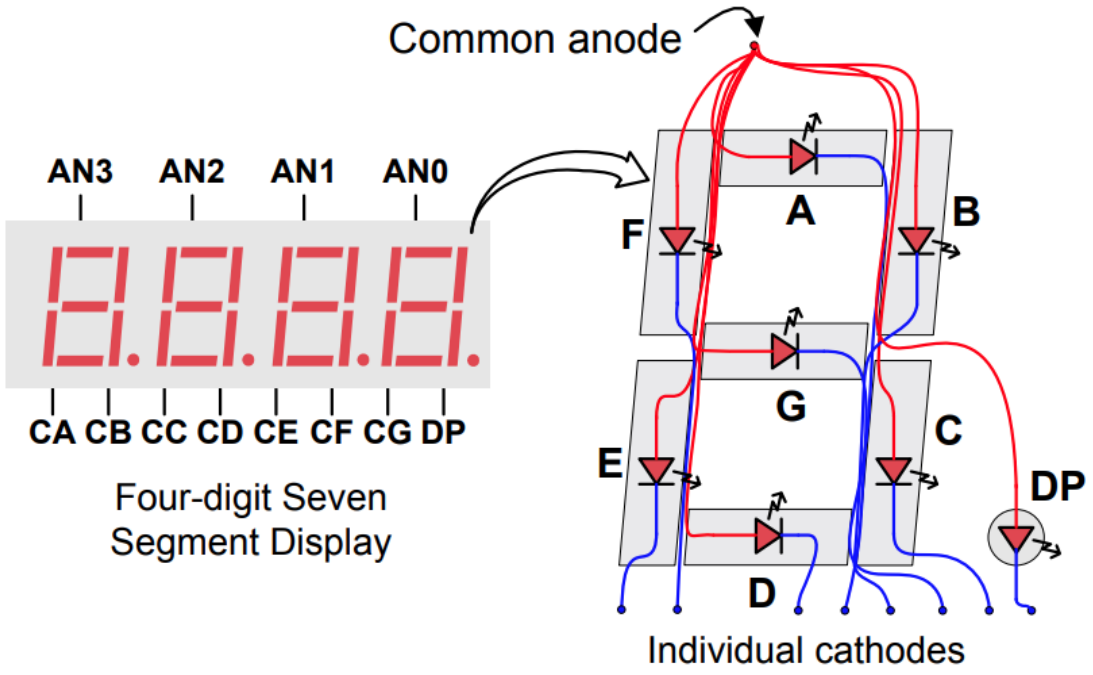


Fig. 14 – Structura unui modul de afișare cu șapte segmente [3]

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

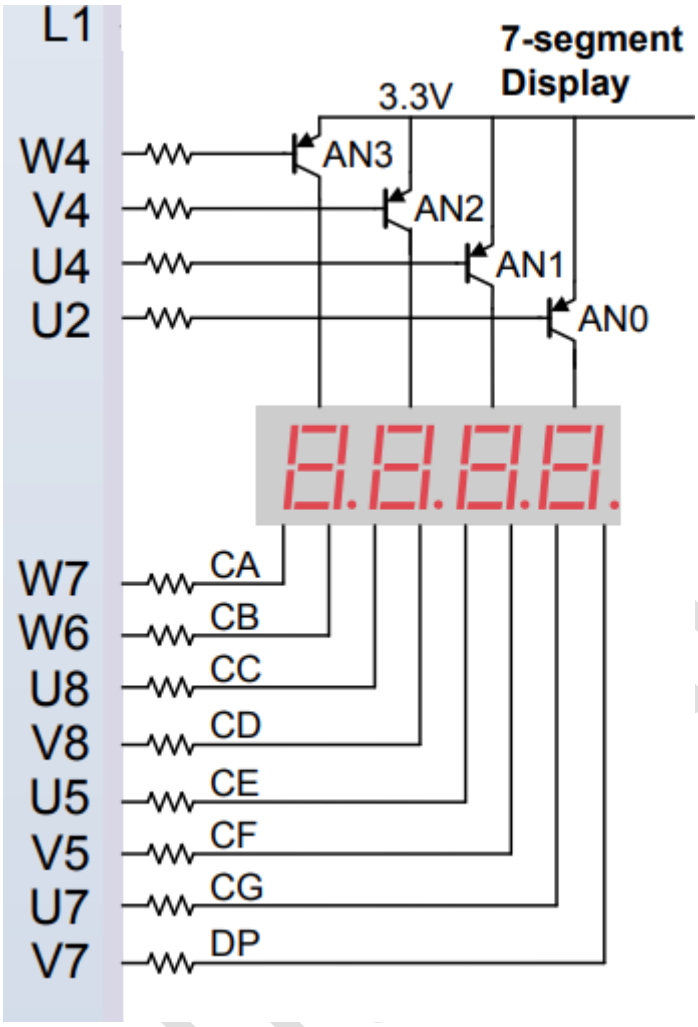


Fig. 15 – Conectarea modulelor de afișare cu șapte segmente la capsula FPGA [3]

Astfel, pentru a afișa valorile numerice rezultate în urma incrementării în sistem de numerație hexa-zecimal pe afișajul cu șapte segmente, este necesar să se activeze tranzistoarele bipolare „AN0”, „AN1”, „AN2” și „AN3” în mod succesiv, iar catodul fiecărui segment va trebui să fie pus la masă prin intermediul pinilor „W7”, „W6”, „U8”, „V8”, „U5”, „V5”, „U7”, „V7”, în funcție de combinațiile regăsite în tabelul de adevăr. În cazul de față, pentru a simplifica acest proces, se va opta pentru afișarea aceluiaș caracter pe toate cele patru module sau pe un singur modul de afișare. De asemenea, tot pentru a simplifica partea de implementare, se optează pentru prescrierea combinațiilor de activare a catozilor sub formă zecimală. Astfel, pe baza valorilor de combinații zecimale și pe baza comparațiilor succesive cu valorile rezultate în urma incrementării, se vor realiza corespondențele, anume:
- pentru a afișa cifra „0” pe afișajul cu șapte segmente este necesară introducerea valorii „1”;
- pentru a afișa cifra „1” pe afișajul cu șapte segmente este necesară introducerea valorii „79”;
- pentru a afișa cifra „2” pe afișajul cu șapte segmente este necesară introducerea valorii „18”;
Logica de comutare a secvențelor pentru catozi este inversă. Comutarea tranzistoarelor de alimentare (în modelul de față), se va realiza prin intermediul comutatoarelor „SW0”, „SW1”, „SW2” și „SW2”, conectate la pinii „V17”, „V16”, „W16”, „W17”. Atât terminalele indicatorilor LED cât și cele ale segmentelor de afișare vor fi configurate în blocuri de tip „Gateway Out”.

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

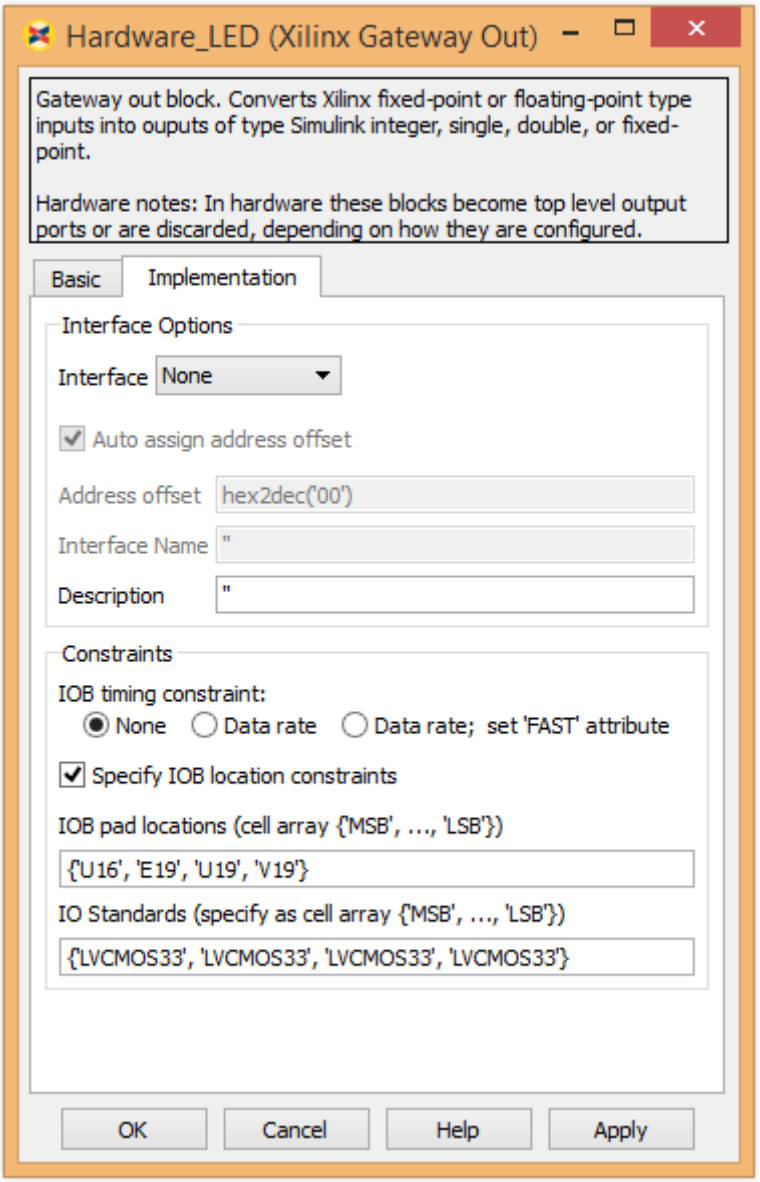
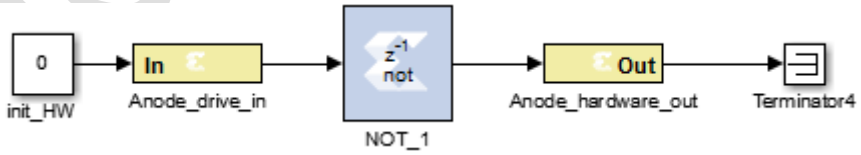


Fig. 16 – Definirea terminalelor specifice indicatorilor LED



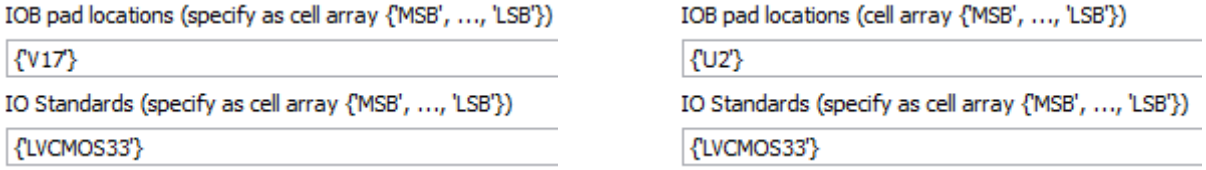


Fig. 17 – Conectarea comutatoarelor bipoziționale la tranzistoarele de alimentare pentru modulele de afișare cu șapte segmente

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

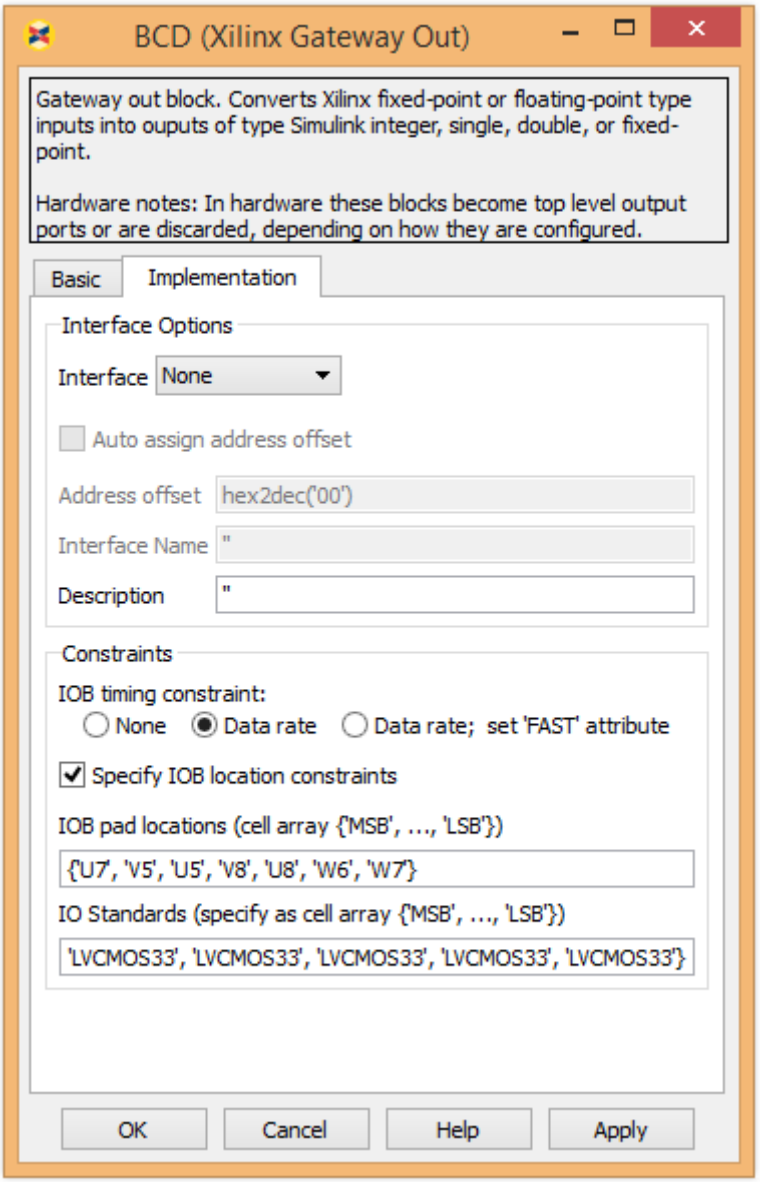


Fig. 18 – Definirea terminalelor specifice segmentelor de afișare

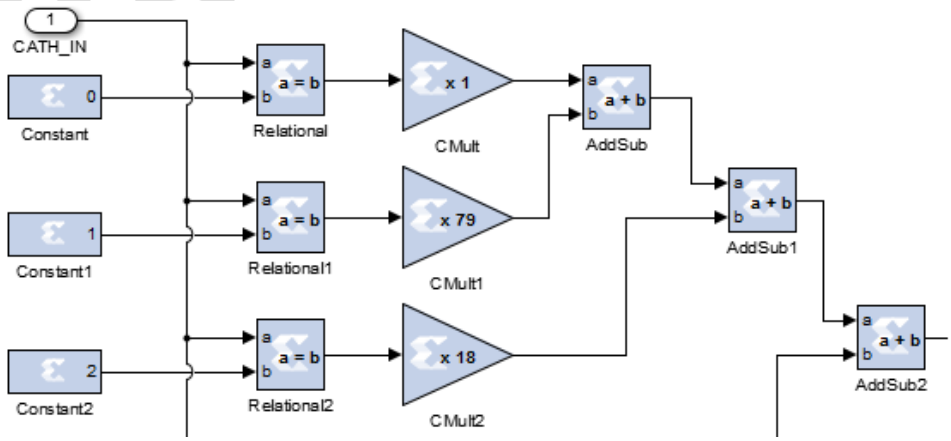


Fig. 19 – Structura blocului de identificare a combinațiilor „7_segment_display”

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

Pentru a testa în timp real funcționalitatea modelului implementat, se va opta pentru metoda de simulare și testare „Hardware Co-Simulation”. Astfel în cadrul blocului „System generator” se vor stabili parametrii pentru generarea automată a codului program. În categoria „Clocking” se vor stabili parametrii specifici oscilatorului pe bază de cuarț, anume:
- se va debifa opțiunea „Enable multiple clocks”;
- se va debifa opțiunea „Provide clock enable clear pin”;
- perioada de eșantionare – Simulink system period (sec): „1e-4”;
- metoda de analiză și testare – Perform analysis: „None”;
- tipul metodei de analiză – Analyzer type: „Timing”;

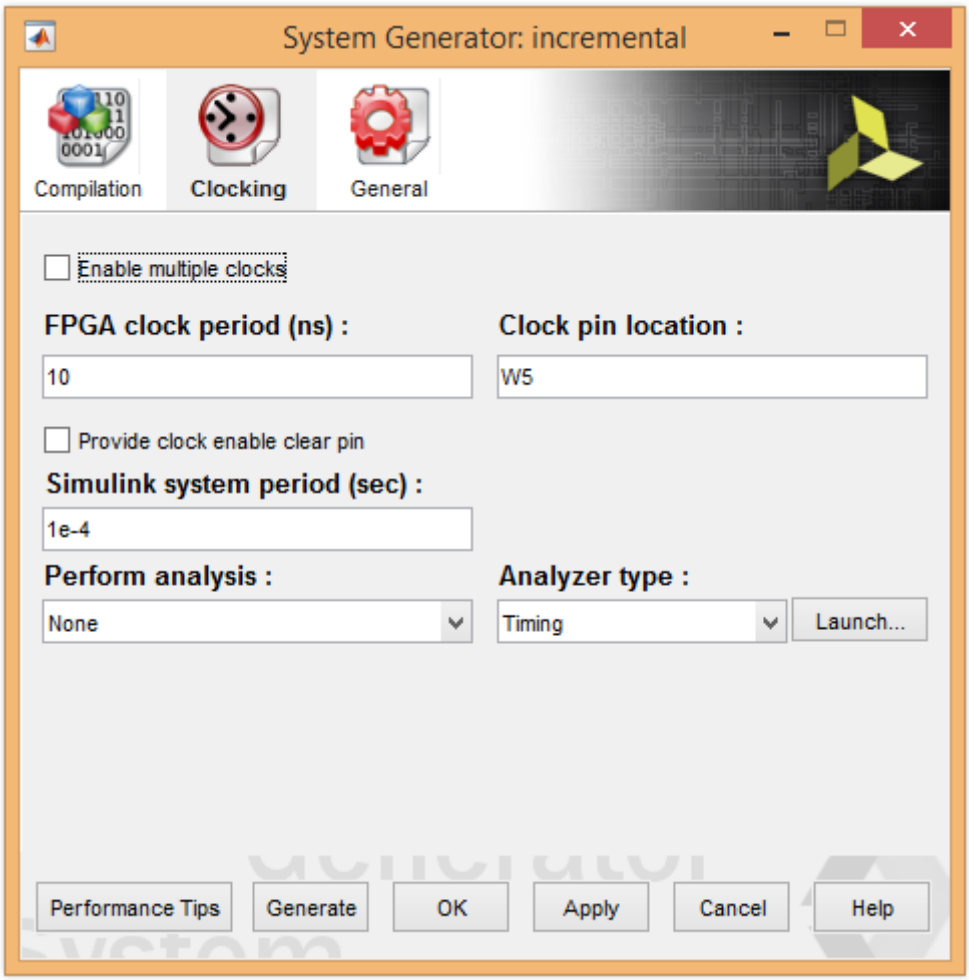


Fig. 20 – Configurarea oscilatorului pe bază de cuarț

În continuare, în prima categorie, „Compilation”, se vor alege următorii parametrii:
- platforma de dezvoltare – Board: „Basys3 C.0”;
- denumirea capsulei FPGA – Part: „Artix7 xc7a35t-1cpg236”;
- metoda de compilare – Compilation: „Hardware Co-Simulation (JTAG)”;
- limbajul de implementare – Hardware description language: „VHDL”;
- bibleoteca utilizată pentru limbajul de implementare – VHDL library: „xil_defaultlib”;
- se va debifa opțiunea „Use STD_LOGIC type for Boolean or 1 bit wide gateways”;

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

- directorul destinație – Target directory: „./netlist”;
- strategia de sintetizare – Synthesis strategy: „Vivado Synthesis Defaults”;
- strategia de implementare – Implementation strategy: „Vivado Implementation Defaults”;
- se va debifa opțiunea „Create interface document”;
- se va debifa opțiunea „Create testbench”;
- se va alege opțiunea „Apply”;
- se va alege opțiunea „Generate”;

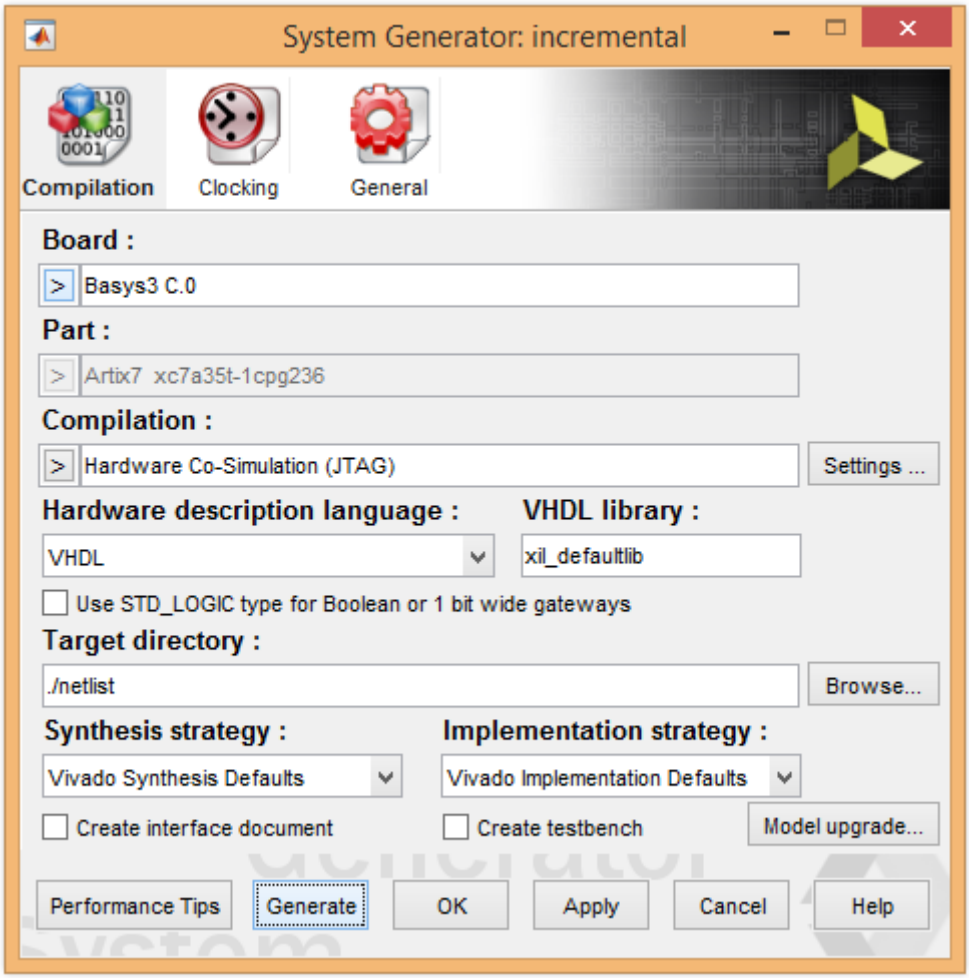
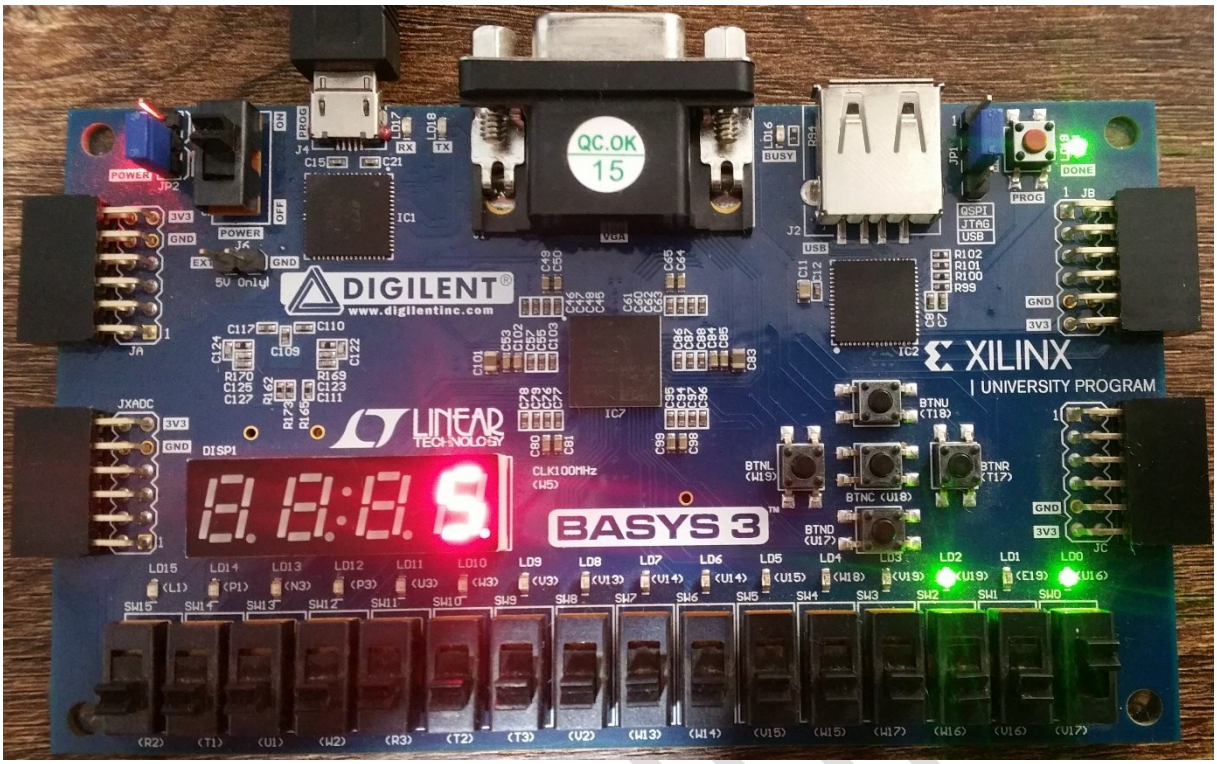


Fig. 21 – Configurarea parametrilor specifici platformei de dezvoltare și generearea aplicației executabile prin apăsarea butonului „Generate”

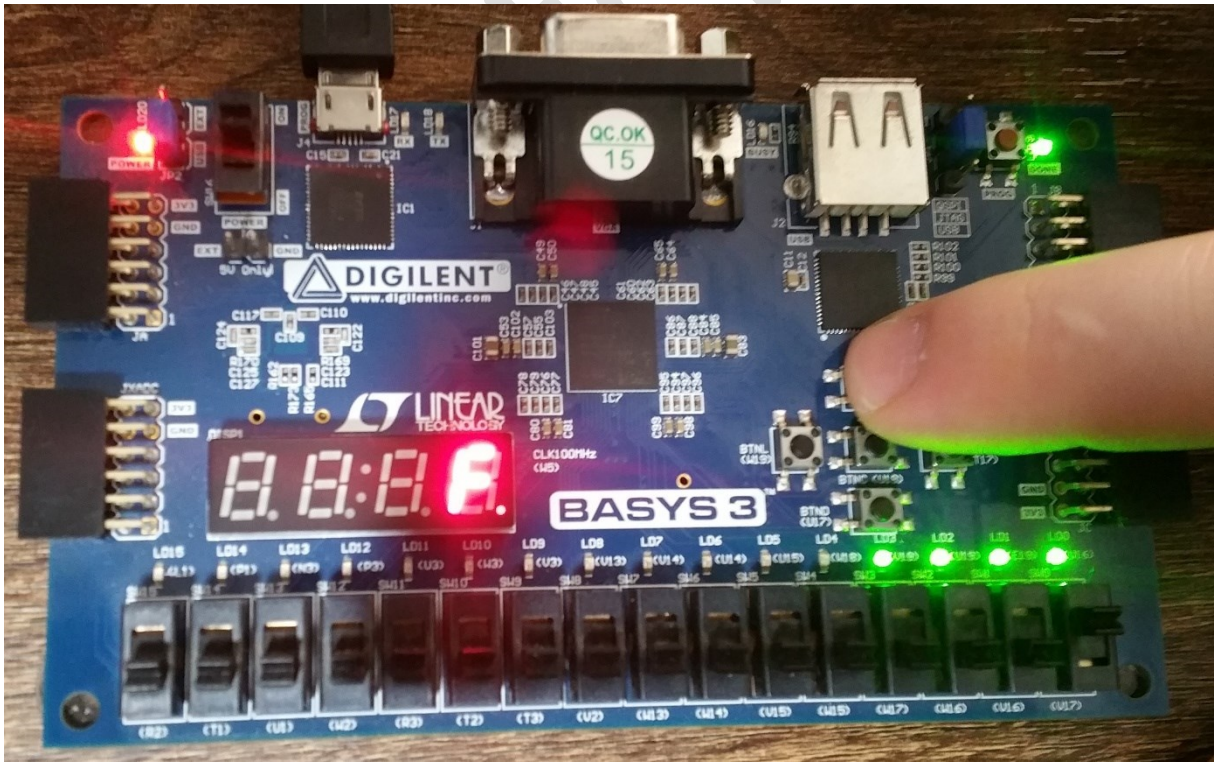
Procesul de generarea al fișierului executabil, necesar pentru re-cnfigurarea structurii interne a capsulei FPGA, este un proces îndelungat (durează aproximativ 5 minute). În cazul în care a fost aleasă metoda pentru testare „Hardware Co-Simulation”, se va genera în mod automat un bloc pentru co-simulare prin intermediul interfeței de comunicare jTAG.

Prin introducerea blocului de co-simulare în model, se va realiza legătura de comunicare între platforma de dezvoltare Basys 3 și calculatorul gazdă. Prin interfața jTAG, se va realiza schimbul de date în timp rea. Astfel, parametrii care au fost expuși prin interfața de comunicare AXI, vor putea fi modificați în timpul simulării.

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro



A.



B.

Fig. 22 – Funcționarea programului implementat la nivelul platformei Basys 3

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

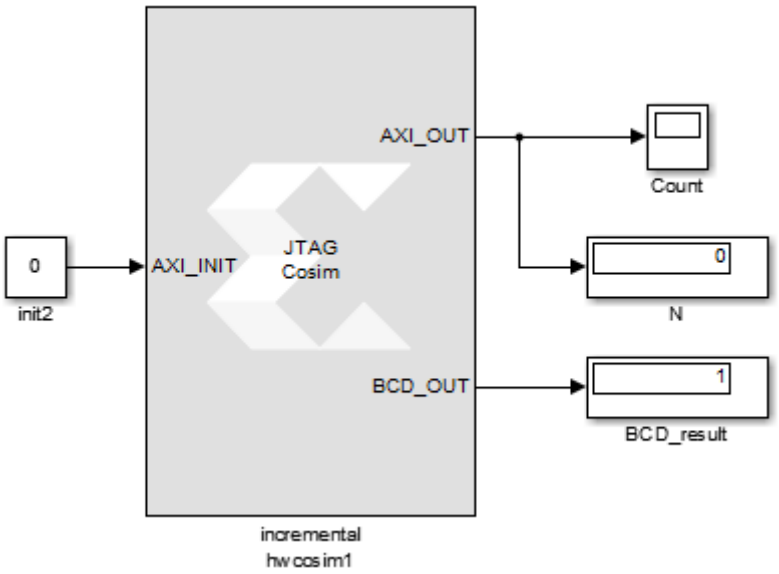


Fig. 23 – Blocul de co-simulare și comunicare în timp real prin intermediul interfeței jTAG

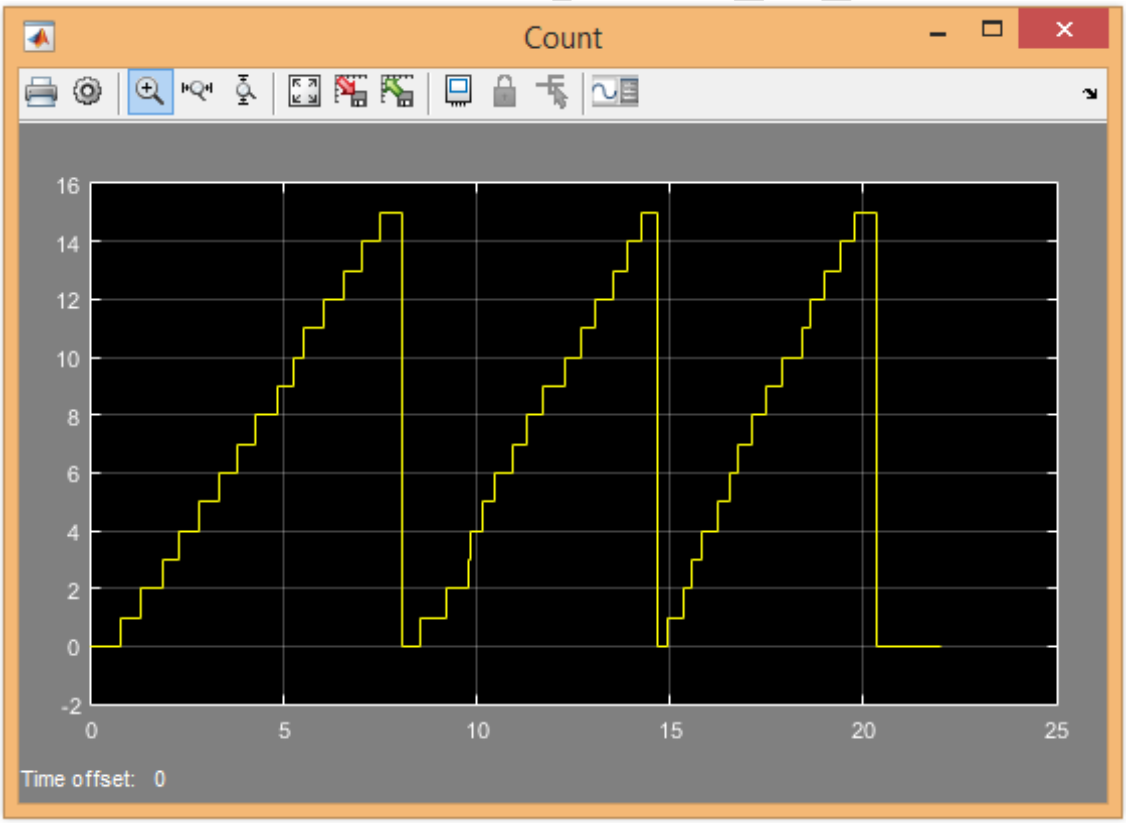


Fig. 24 – Mărimea numerică rezultantă

III. CONCLUZIE:

Utilizarea pachetului System Generator, permite atât generarea fișierului executabil de re-configurare cât și testarea sau simularea funcționalității aplicației în timp real în Simulink.

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP”
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro

IV. BIBLIOGRAFIE:

1. Vivado Design Suite User Guide – „Model-Based DSP Design Using System Generator" - UG897 (v2016.2) June 8, 2016 - © Copyright 2012–2016 Xilinx, Inc. Xilinx: (https://www.xilinx.com/support/documentation/sw_manuals/xilinx2016_2/ug897-vivado-sysgen-user.pdf);

2. rsdelivers.com – „Digilent 410-183 Basys Artix-7 Development Board": (https://ro.rsdelivers.com/product/digilent/410-183/digilent-410-183-basys-artix-7-development-board/1346451);

3. DIGILENT Basys 3 FPGA Board Reference Manual – „Seven-Segment Display": (https://reference.digilentinc.com/_media/basys3:basys3_rm.pdf);

4. Teodor Crișan Pană – „Sisteme de calcul cu microprocesoare, FPGA și DSP" – Editura UTPRESS, Cluj – Napoca, 2016 – ISBN 978-606-737-206-9;

5. Ioana – Cornelia GROS, Lucian – Nicolae PINTILIE, Teodor Crișan PANĂ – „SISTEME EMBEDDED ÎN INGINERIE ELECTRICĂ - GHID DE APLICAȚII" – Editura UTPress Cluj – Napoca, 2020 ISBN 978-606-737-431-5: (https://biblioteca.utcluj.ro/files/carti-online-cu-coperta/431-5.pdf);

Realizat de: ing. drd. Pintilie Lucian - Nicolae
Pentru disciplina: „Sisteme cu FPGA și DSP"
Adresă de e-mail: Lucian.Pintilie@emd.utcluj.ro