FACULTATE: AUTOMATICĂ ȘI CALCULATOARE
Specializare: Calculatoare și tehnologia informației

Proiect: Linear Feedback Shift Register Counter (LFSR Counter)

Laboratory teacher:
Lorena Dăian

Student:
Niță Eduard

Cuprins

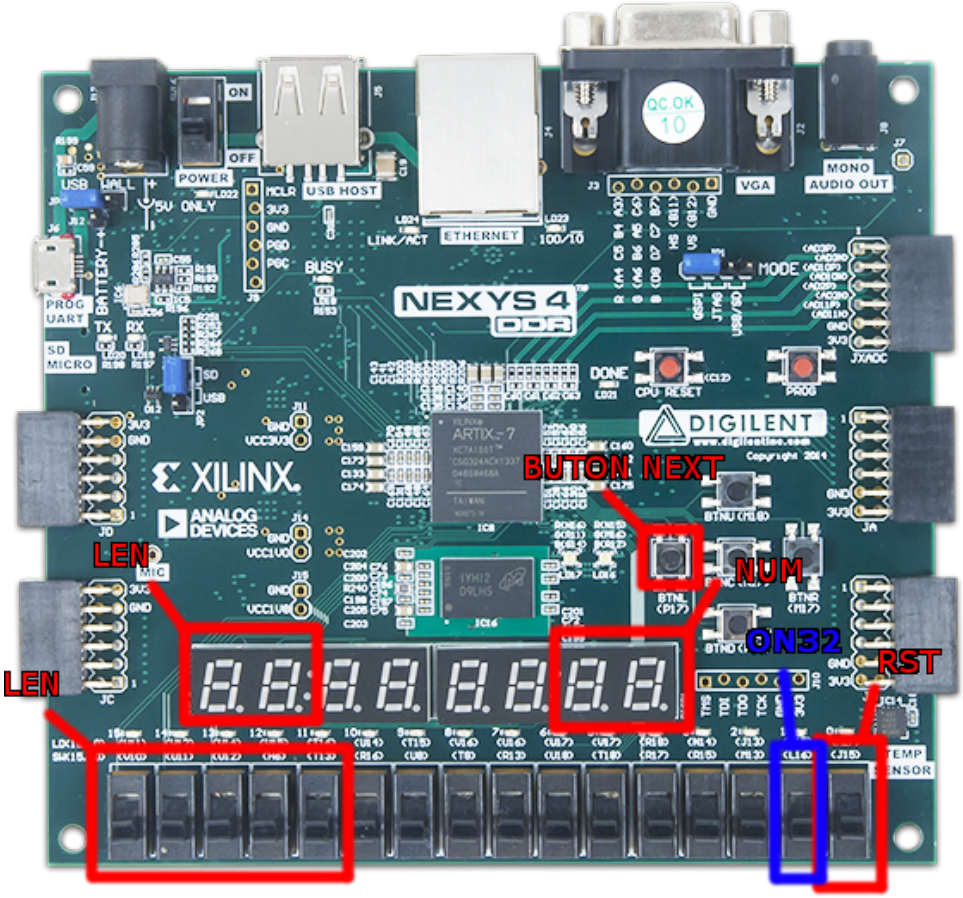
Specificația proiectului

Tema proiectului

Să se implementeze schemele de construire a **numărătoarelor cu reacție liniară** (LFSR), conform documentației existente în cartea “Proiectarea sistemelor numerice folosind tehnologia FPGA” de S. Nedevschi, Z. Baruch și O. Creț, în capitolul 4 al lucrării. Sistemul proiectat primește ca intrări lungimea buclei de numărare.

Proiectul a fost implementat pe o placă FPGA Nexys 4, intrările și ieșirile sistemului fiind plasate în concordanță cu black box-ul următor:

I/O-ul black box-ului corespunde cu următoarele componente ale plăcuței Nexys 4:



Numărătorul LFSR

Orice numărător LFSR generează o secvență pseudo-aleatoare de “0”-uri și “1”-uri de mare lungime. Această secvență nu este strict aleatoare de vreme ce ea se repetă începând cu un moment dat, și de asemenea ea urmează o secvență predictibilă mathematic. Dar pentru majoritatea aplicațiilor practice, această secvență poate fi considerate aleatoare.

De exemplu, pentru un numărător LFSR pe 63 de biți ce merge cu o frecvență de 50 MHz, un astfel de numărător ajunge să își repete stările după mai mult de 5000 de ani.

Un numărător LFSR de n biți cu secvență de lungime maximă constă dintr-un registru de deplasare de n biți care are în bucla de reacție o poartă logica XNOR între anumite ieșiri, această poartă intrând în registru.

Poarta XNOR face ca starea de blocare să fie starea în care toate Q-urile sunt “1”; adăugând o poartă logică SAU-EXCLUSIV facem ca această stare să fie “00…0”.

Pentru a obține o secvență maximă de lungime n, poarta XNOR trebuie să fie legată conform tabelului următor (unde valorile din registru încep de la index-ul 1):

n	XNOR from	n	XNOR from	n	XNOR from	n	XNOR from
3	3,2	45	45,44,42,41	87	87,74	129	129,124
4	4,3	46	46,45,26,25	88	88,87,17,16	130	130,127
5	5,3	47	47,42	89	89,51	131	131,130,84,83
6	6,5	48	48,47,21,20	90	90,89,72,71	132	132,103
7	7,6	49	49,40	91	91,90,8,7	133	133,132,82,81
8	8,6,5,4	50	50,49,24,23	92	92,91,80,79	134	134,77
9	9,5	51	51,50,36,35	93	93,91	135	135,124
10	10,7	52	52,49	94	94,73	136	136,135,11,10
11	11,9	53	53,52,38,37	95	95,84	137	137,116
12	12,6,4,1	54	54,53,18,17	96	96,94,49,47	138	138,137,131,130
13	13,4,3,1	55	55,31	97	97,91	139	139,136,134,131
14	14,5,3,1	56	56,55,35,34	98	98,87	140	140,111
15	15,14	57	57,50	99	99,97,54,52	141	141,140,110,109
16	16,15,13,4	58	58,39	100	100,63	142	142,121
17	17,14	59	59,58,38,37	101	101,100,95,94	143	143,142,123,122
18	18,11	60	60,59	102	102,101,36,35	144	144,143,75,74
19	19,6,2,1	61	61,60,46,45	103	103,94	145	145,93
20	20,17	62	62,61,6,5	104	104,103,94,93	146	146,145,87,86
21	21,19	63	63,62	105	105,89	147	147,146,110,109
22	22,21	64	64,63,61,60	106	106,91	148	148,121
23	23,18	65	65,47	107	107,105,44,42	149	149,148,40,39
24	24,23,22,17	66	66,65,57,56	108	108,77	150	150,97
25	25,22	67	67,66,58,57	109	109,108,103,102	151	151,148
26	26,6,2,1	68	68,59	110	110,109,98,97	152	152,151,87,86
27	27,5,2,1	69	69,67,42,40	111	111,101	153	153,152
28	28,25	70	70,69,55,54	112	112,110,69,67	154	154,152,27,26
29	29,27	71	71,65	113	113,104	155	155,154,124,123
30	30,6,4,1	72	72,66,25,19	114	114,113,33,32	156	156,155,41,40
31	31,28	73	73,48	115	115,114,101,100	157	157,156,131,130
32	32,22,2,1	74	74,73,59,58	116	116,115,46,45	158	158,157,132,131
33	33,20	75	75,74,65,64	117	117,115,99,97	159	159,128
34	34,27,2,1	76	76,75,41,40	118	118,85	160	160,159,142,141
35	35,33	77	77,76,47,46	119	119,111	161	161,143
36	36,25	78	78,77,59,58	120	120,113,9,2	162	162,161,75,74
37	37,5,4,3,2,1	79	79,70	121	121,103	163	163,162,104,103
38	38,6,5,1	80	80,79,43,42	122	122,121,63,62	164	164,163,151,150
39	39,35	81	81,77	123	123,121	165	165,164,135,134
40	40,38,21,19	82	82,79,47,44	124	124,87	166	166,165,128,127
41	41,38	83	83,82,38,37	125	125,124,18,17	167	167,161
42	42,41,20,19	84	84,71	126	126,125,90,89	168	168,166,153,151
43	43,42,38,37	85	85,84,58,57	127	127,126		
44	44,43,18,17	86	86,85,74,73	128	128,126,101,99		

Implementarea unui Numărător LFSR pe 5 biți

Pentru implementarea acestui numărător am urmat schema pusă la dispoziție în cartea "Proiectarea sistemelor numerice folosind tehnologia FPGA".

Pentru un numărător LFSR de 5 biți clasic, bucla de reacție fundamentală de la Q_5 și Q_3 exclude starea 11111. Decodificând cele două stări în care cei patru biți mai puțin semnificativi sunt toți "1" și inversând reacția pentru aceste stări, numărătorul LFSR pe 5 biți numără modulo 32 și nu se blochează în nici o stare.

Schema pusă la dispoziție este aceasta:

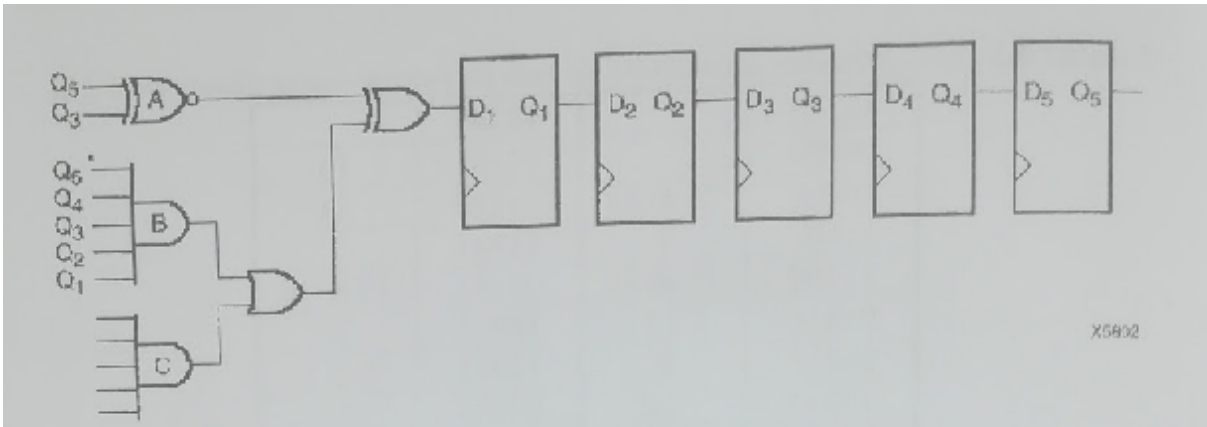


Figura 4.6 *Numărător modulo n pe 5 biți*

Figura de mai sus nu este suficientă pentru implementarea unui numărător LFSR, însă ea este acompaniată de către un set de reguli, dar și un tabel pentru decodificarea semnalului de reacție pentru numărătoare pe 5 biți.

Tabelul 4.2 *Decodificarea semnalului de reacție pentru numărătoare pe 5 biți*

12345		12345	
00000	-	11101	13
10000	23 & 22	11110	31
11000	7	11111	-
11100	19 & 18	01111	15 & 14
01110	17	10111	29
00111	8	11011	6 & 5
10011	12	01101	26
01001	28 & 27	10110	3
00100	15	01011	11 & 10
00010	11	00101	17 & 16
10001	9	10010	20
01000	4	11001	25
10100	30	01100	6
01010	21	00110	24
10101	2	00011	21
11010	26	00001	31

La numărătoarele pe 5 biți, pentru a construe un numărător modulo n, se utilizează poarta ȘI "C"din figură pentru decodificarea șabloanelor binare înscrie în tabelul de mai sus, alături de lungimea dorită a buclei. În funcție de prima coloană a tabelului vom alege intrările

porții C. Dacă bit-ul de pe poziția n este “1”, atunci în poarta C va intra Q_n, iar dacă este “0”, atunci în poarta C va intra $\acute{Q}_n$.

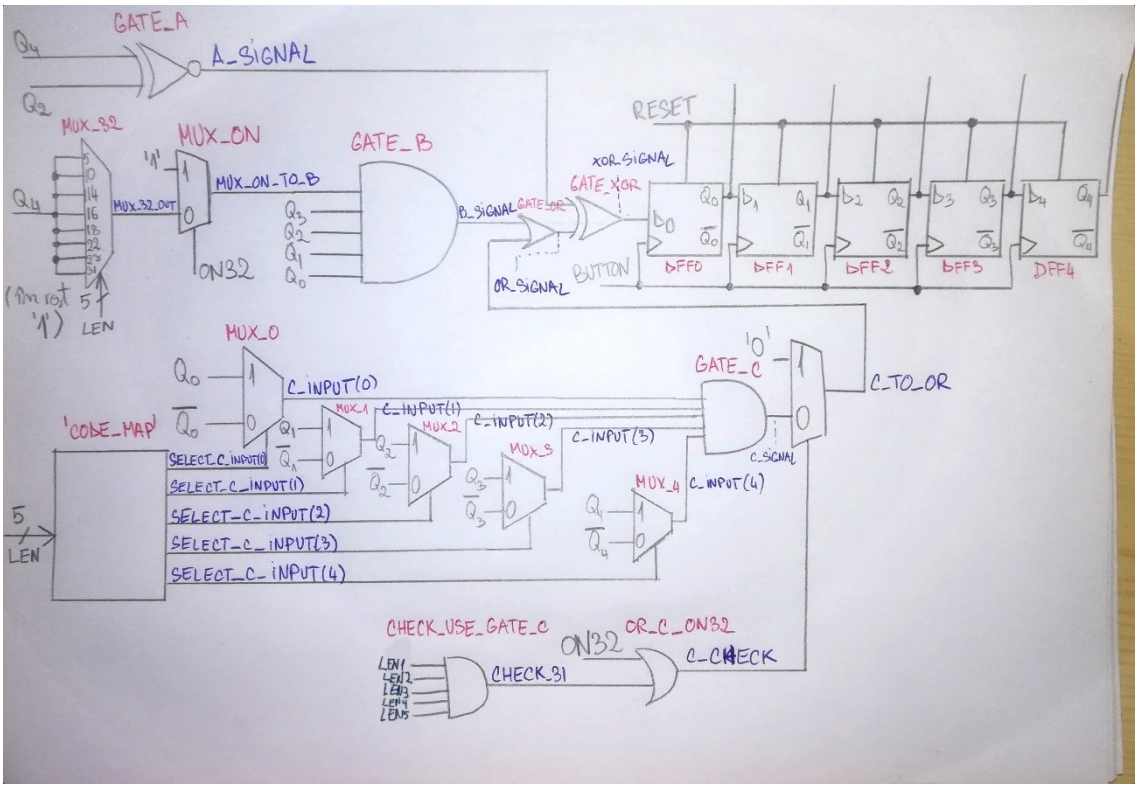
De exemplu, dacă pe prima coloană avem “10101”, atunci intrările porții C vor fi $Q_1\acute{Q}_2 Q_3\acute{Q}_4Q_5$.

Pentru numărul 31 și orice alt număr înscris la dreapta simbolului „&”, se va adăuga de asemnea semnalul Q_5 la intrarea porții ȘI „B”, pentru evitarea blocării în starea „11111”.

Vom implementa tabelul de mai sus ca și o memorie RAM, unde adresa va fi lungimea buclei de numărare, iar conținutul ce se află la acea adresă va fi șablonul care ne va ajuta să alegem conținutul porții C.

Pentru a evita blocarea în starea „11111”, vom respecta anumite reguli:

1. Pentru numărul 32: nu se conectează Q_5 la intrarea porții ȘI “B”; nu se utilizează poarta ȘI “C”
2. Pentru numărul 31: se conectează Q_5 la intrarea porții ȘI “B”; nu se utilizează poarta ȘI ”C”
3. Pentru orice număr mai mic decât 31: se programează poarta ȘI “C” conform tabelului
4. Pentru numerele 5, 10, 14, 16, 18, 22, 27, 31: se conectează Q_5 la intrarea porții ȘI “B”
5. Pentru celelalte numere: nu se conectează Q_5 la intrarea porții ȘI “B”



Observație: numerotarea bistabilelor în implementarea mea începe de la 0, așa că informațiile ce se regăsesc în carte sunt echivalente atâta timp cât:

Q_n (carte) ~ Q_{n-1} (implementare)

Implementarea schemei bloc de mai sus a fost realizată printr-o arhitectură structurală, iar schema bloc de mai sus reprezintă bucata de cod din entitatea principală ce leagă elementele necesare:

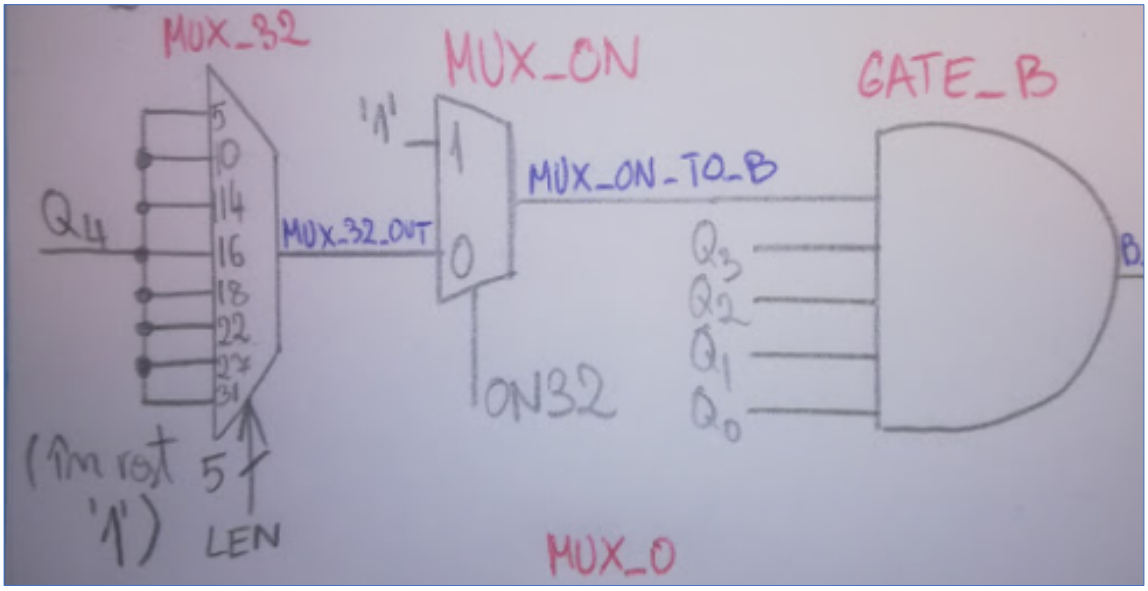
```
CODE_MAP: ROM_5_TO_5 port map(LEN, SELECT_C_INPUT);
MUX_4: MUX_2_TO_1 port map(SELECT_C_INPUT(4), Q(4), Qn(4), C_INPUT(4));
MUX_3: MUX_2_TO_1 port map(SELECT_C_INPUT(3), Q(3), Qn(3), C_INPUT(3));
MUX_2: MUX_2_TO_1 port map(SELECT_C_INPUT(2), Q(2), Qn(2), C_INPUT(2));
MUX_1: MUX_2_TO_1 port map(SELECT_C_INPUT(1), Q(1), Qn(1), C_INPUT(1));
MUX_0: MUX_2_TO_1 port map(SELECT_C_INPUT(0), Q(0), Qn(0), C_INPUT(0));
GATE_C: AND_5 port map(C_INPUT(4), C_INPUT(3), C_INPUT(2), C_INPUT(1), C_INPUT(0), C_SIGNAL);
CHECK_USE_GATE_C: AND_5 port map(LEN(4), LEN (3), LEN (2), LEN (1), LEN (0), CHECK_31);
OR_C_ON32: OR_2 port map(CHECK_31, ON32, C_CHECK);
MUX_C: MUX_2_TO_1 port map(C_CHECK, '0', C_SIGNAL, C_TO_OR);
GATE_A: XNOR_2 port map(Q(4), Q(2), A_SIGNAL);
MUX_32: MUX_32_TO_1 port map(LEN, MUX_32_INPUT, MUX_32_OUT);
MUX_ON: MUX_2_TO_1 port map(ON32, '1', MUX_32_OUT, MUX_ON_TO_B);
GATE_B: AND_5 port map(MUX_ON_TO_B, Q(3), Q(2), Q(1), Q(0), B_SIGNAL);
GATE_OR: OR_2 port map(B_SIGNAL, C_TO_OR, OR_SIGNAL);
GATE_XOR: XOR_2 port map(OR_SIGNAL, A_SIGNAL, XOR_SIGNAL);
DFF0: DFF port map(XOR_SIGNAL, BUTTON, RESET, Q(0), Qn(0));
DFF1: DFF port map(Q(0), BUTTON, RESET, Q(1), Qn(1));
DFF2: DFF port map(Q(1), BUTTON, RESET, Q(2), Qn(2));
DFF3: DFF port map(Q(2), BUTTON, RESET, Q(3), Qn(3));
DFF4: DFF port map(Q(3), BUTTON, RESET, Q(4), Qn(4));
```

Implementarea regulilor

Regula 1: “*Pentru numărul 32: nu se conectează Q_5 la intrarea porții ȘI “B”; nu se utilizează poarta ȘI “C”* “

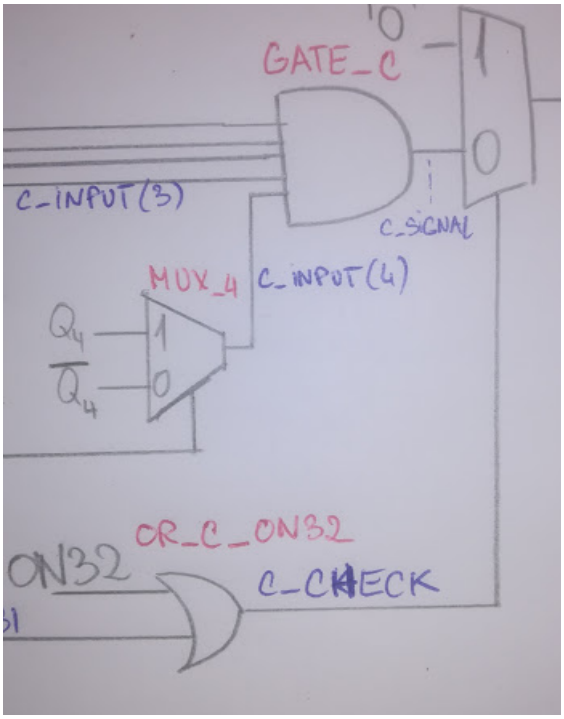
Implementarea primei reguli pare a fi dificilă la prima vedere, deoarece numărătorul nostru este unul pe 5 biți. Vom introduce un semnal de 1 bit, acesta fiind ON32, ce va determina dacă este vorba de numărul 32. O altă opțiune ar fi să avem semnalul LEN pe 6 biți, însă acest lucru ar permite introducerea unor lungimi ce nu sunt de folos pentru un numărător LFSR *pe 5 biți*.

Pentru prima parte a regulii, “*nu se conectează Q_5 la intrarea porții ȘI “B”* vom adăuga un multiplexor(MUX_ON) cu semnalul de selecție ON32, care va determina dacă numărătorul merge conform celorlalte reguli sau în modul special de a genera o secvență de 32 de numere. Acest multiplexor va intra în poarta B.



Astfel, dacă ON32 este ‘0’, atunci numărătorul va funcționa normal. Dacă ON32 este ‘1’ atunci în poarta ȘI B va intra semnalul ‘1’, ce nu va afecta deloc poarta B.

Pentru a doua parte a reguli, “*nu se utilizează poarta ȘI “C” ”,* vom amplasa un multiplexor cu selectia ON32 după poarta C, iar dacă ON32 este ‘0’, atunci prin multiplexor va trece semnalul ce iese din poarta C, iar dacă ON32 este ‘1’, atunci prin multiplexor va trece semnalul ‘0’, deoarece semnalul trebui să treacă printr-o poartă SAU. Astfel, semnalul nu va afecta poarta SAU.

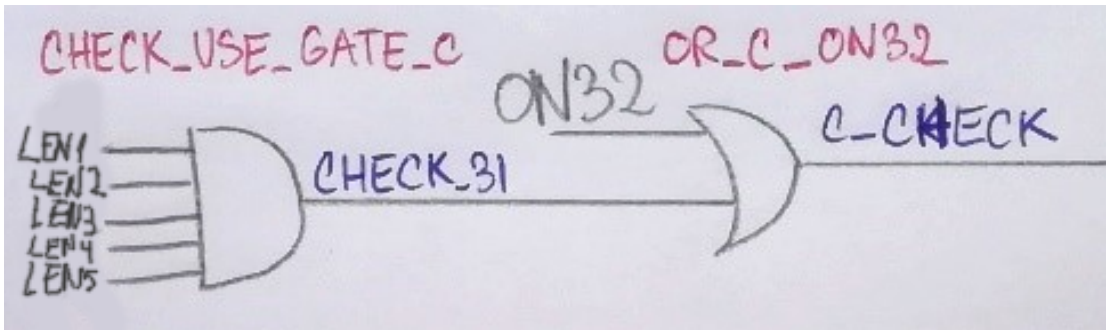


Semnalul ON32 trece printr-o poartă SAU deoarece mai există o condiție pentru folosirea semnalului ce iese din poarta C.

Regula 2: „Pentru numărul 31: se conectează Q_5 la intrarea porții ȘI “B”; nu se utilizează poarta ȘI ”C”””

Prima parte a regulii 2, “se conectează Q_5 la intrarea porții ȘI “B”” este inclusă în regula 4, așa că nu o vom detalia acum.

A doua parte a regulii 2, „nu se utilizează poarta ȘI ”C””, este ușor de implement deoarece numărul 31 fiind reprezentabil pe 5 biți și fiind numărul maxim pe 5 biți putem trece semnalul LEN printr-o poartă ȘI pentru a dedecta numărul 31. Semnalul ce va rezulta va trece printr-o poartă SAU alături de ON32 și va intra în multiplexorul ce condiționează ieșirea porții C.

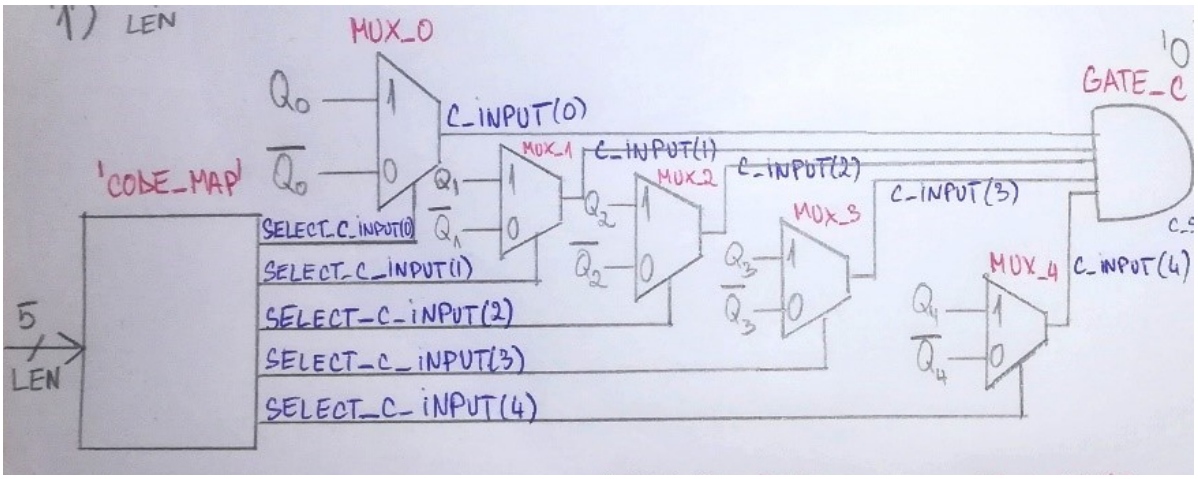


Regula 3: “Pentru orice număr mai mic decât 31: se programează poarta ȘI “C” conform tabelului”

Tabelul va fi implementat cu ajutorul unei memorii ROM, unde adresa este lungimea buclei de numărare, iar conținutul(prima coloană) este modul în care se va codifica poarta C.

În funcție de prima coloană a tabelului vom alege intrările porții C. Dacă bit-ul de pe poziția n este “1”, atunci în poarta C va intra Q_n, iar dacă este “0”, atunci în poarta C va intra $\acute{Q}_n$.

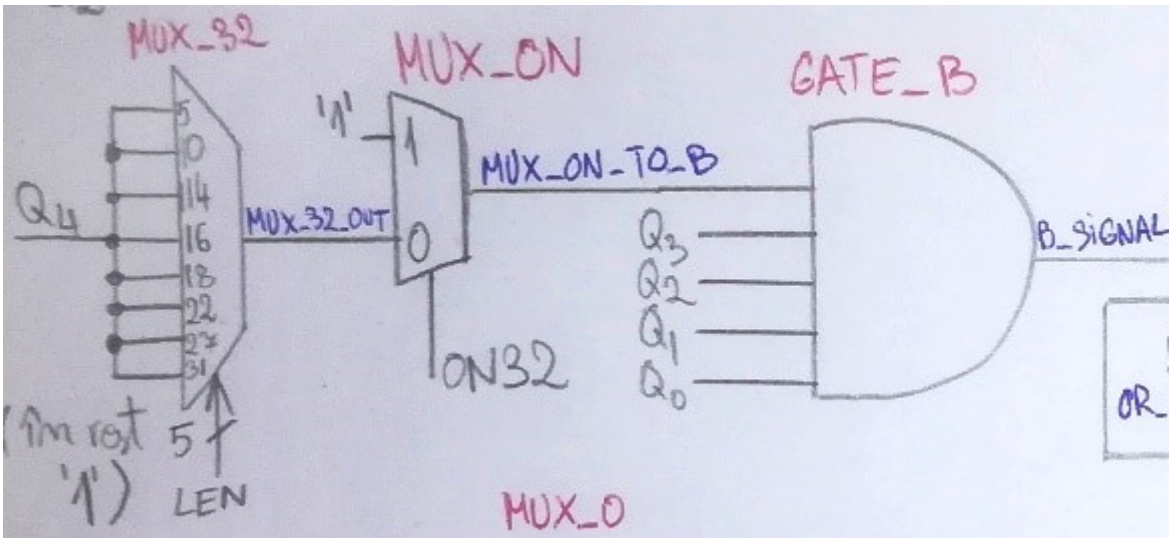
Pentru a implementa acest lucru, vom lega fiecare ieșire n din memorie la un multiplexor ca și semnal de selecție. Dacă semnalul este este “1”, atunci în poarta C va intra Q_n, iar dacă este “0”, atunci în poarta C va intra $\acute{Q}_n$.



Regula 4: „Pentru numerele 5, 10, 14, 16, 18, 22, 27, 31: se conectează Q_5 la intrarea porții ȘI “B” ”

Regula 5: „Pentru celelalte numere: nu se conectează Q_5 la intrarea porții ȘI “B” ”

Aceste două reguli sunt complementare și le putem implementa printr-un mod simplu: Vom folosi un multiplexor 32:1, unde semnalul de selecție este LEN, iar în intrările menționate în regula 4 vom introduce semnalul Q_5 (Q_4 în reprezentarea noastră). În intrările celelalte vom introdce semnalul constant ‘1’, deoarece ieșirea multiplexorului trebuie să intre în poarta B. Astfel, semnalul ‘1’ nu va afecta o poartă ȘI.



Desigur, înainte ca semnalul să intre în poarta B, acesta trebuie să treacă mai întâi prin multiplexorul ce verifică dacă a fost ales modul pentru numărul 32, deoarece acesta are prioritate.

Accesarea codului sursă

Pentru a nu aglomera documentația cu bucăți din codul sursă și deoarece componentele folosite sunt unele uzuale, am ales să postez întregul cod sursă pe github:

https://github.com/snoobism/LFSR-Counter