DESIGN AND IMPLEMENTATION OF A 4 AND 5-BIT LFSR

Ungureanu Iulia Iarina, Stud.

Faculty of Faculty of Automation and Computer Science,

Technical University of Cluj Napoca,

Ungureanu Viorica Mariela, Assoc. Prof. PhD

"Constantin Brâncuşi" University from Târgu Jiu, ROMANIA

ABSTRACT: *Applications of LFSRs include generating pseudo-random numbers, pseudo-noise sequences, fast digital counters or whitening sequences (see [1],[3-5] and the references therein for further applications). Consequently, both hardware and software implementations of LFSRs are of great interest for engineers and researchers working in the field of information theory, coding theory and cryptography. In this paper, we presents a simple and easy way to implement a LFSR counter on 4 and 5 bits. The designed system receives as inputs the length of the counting loop and the selection of the variant of the counter: on 4 or 5 bits.*

KEY WORDS: LFSR, FPGA, VHDL

1. INTRODUCTION

Let us denote by Q_i, i = 1, ..,n the output values of a LFSR with n bits. For a classic 4-bit counter, the fundamental feedback loop from Q3 and Q4 exclude the state 1111.

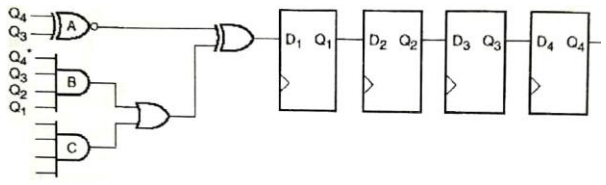


Fig.1

It is well know that, if the two states having the lower three bits equal to "1" are decoded and the feedback for those states is inverted, then the 4-bit LFSR counter counts modulo 16, and has no lock-up state [2,3].

The following table shows the additional decoding of the feedback signal that is required by the LFSR counters that have a shorter cycle N.

Table 1 4-bit counter

Q1234	N=	Q1234	N=
0000	-	0110	3
1000	7	0011	11
1100	11	1001	9, 8
1110	-	0100	14
1111	-	1010	6
0111	13,12	0101	2
1011	10	0010	5
1101	5 ,4	0001	-

It shows the connections required for the functioning of the LFSR counter in any cycle of

length up to 15. The connections will be programmed according to the length of the loop as it follows: for the length value 15, the output Q4 will be connected to the AND gate „B" and the AND gate „C" will not be used; for any length less than 15, the AND gate „C" will be programmed according to the above table; for the lengths 4, 8, 12 and 15 the output Q4 will be connected to the AND gate „B"; for all the other values, Q4 will not be connected to the AND gate "B"[2].

Analogously, in the case of a 5 bits LFSR, we have: for a cycle of length 31, the output Q5 will be connected to the input of AND gate "B" and the AND gate „C" will not be used; for any value less than 31 the AND gate „C" will be programmed according to Table 2; for the values 5, 10, 14, 16, 18, 22, 27, 31 the output Q5 will be connected to the input of the AND gate „B"; for any other values the output Q5 will not be connected to the input of the AND gate „B" [2].

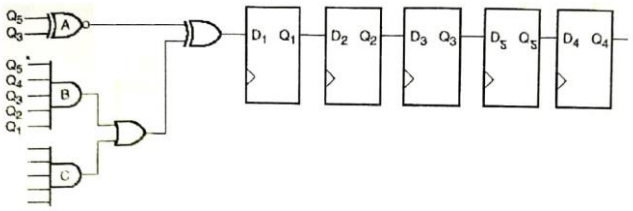


Fig. 2

Table 2 5-bit counter

Q12345	N=	Q12345	N=
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

2. THE PROPOSED IMPLEMENTATION

2.1 Our implementation has three input variables through which the user can control the device.

The first variable named MODE is introduced by a button and allows the choice of the counter variant (operation mode): on 4 or 5 bits. If the button is not pressed, then MODE is 1 logic and the variant on 5-bit is set. Otherwise, the 4-bit variant is set the value of MODE is 0 logic.

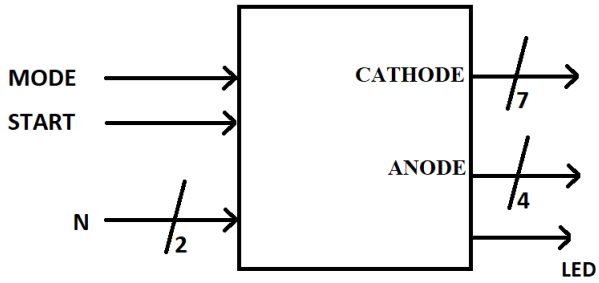
The second variable is the length N of the loop. N is a number in base 10 from 2 to 31 for a 5-bit counter and from 2 to 15 for a 4-bit counter. It will be entered by using two switches as it follows: a switch will be held down for 1s (second) to enter digit 0, 2s to enter digit 1, 3s for digit 2 and so on.

Finally, we will press the START button to start the counting.

The output variables will be **ANODE** and **CATHODE** on 4 and 7 bits, respectively, and will be used to display the numbers on the 7-segment display. All numbers to be displayed are less than or equal to 31, so two displays are used to display them and other two are used to enter N.

One **LED** it will light up for a second to mark the completion of the loop.

The black box will look like this at the end:

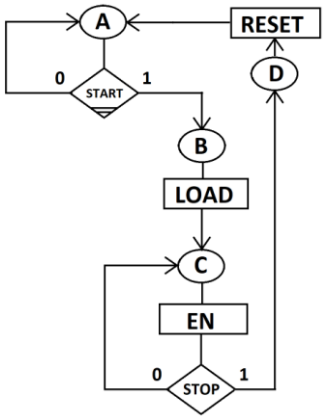


2.2. Execution unit (EU) and Control unit (CU) decomposition

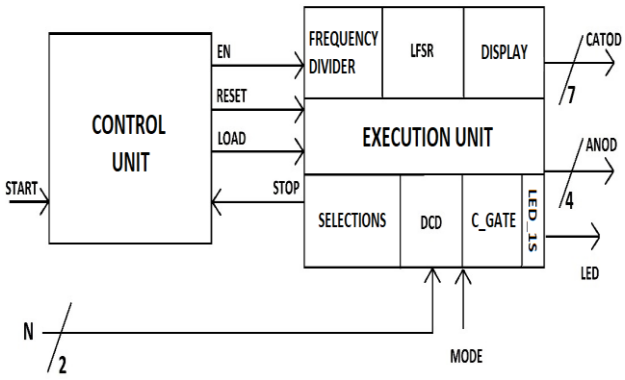
The execution unit is mainly made out of the LFRS (fig 1) itself but it also contains a lot of peripheral components required for the output and the input of the data. All these components are working synchronously guided by the control unit through three signals: ENABLE, RESET and LOAD. The inputs of the control unit are the

START button and the STOP signal that is returned by the LFRS from the EU when the loop is finished.

The next diagram shows the 4 states A-D that our system goes through and describes its behaviour. *A* is the initial state that wait for the input data (the mode that we choose and the length N) until the START button is pressed and the system passes to the next state *B*. Then, the first command is sent to the execution unit LOAD, which loads the numbers from the tables into the register. In state *C*, we have the ENABLE signal (it controls a 7-segment display) that is activated until the loop stop when the signal STOP turns 1 logic. It follows state *D*, which RESETS the LFSR and prepares it for the next loops sent by the user.

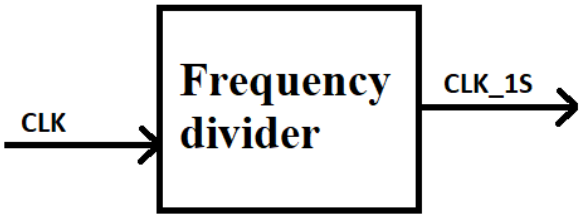


UE will receive as inputs these three signals and, additionally, the operating mode and the loop length N through the MODE button and two switches, respectively.



2.2.1 Resource list of EU: As we can see in the above image, the Execution Unit (right) consists of the following seven essential resources:

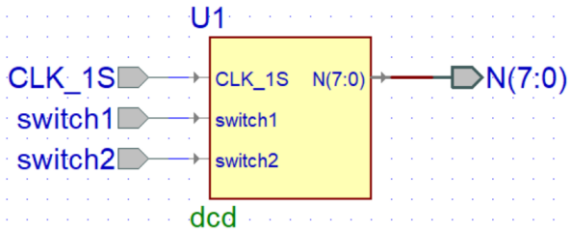
a. Frequency divider – For our implementation we used two clock frequencies: one of 1 HZ (one second) for displaying and reading numbers and one of 250 HZ needed for the human eye to capture light from the seven-segment display. To obtain them we use frequency dividers.



b. DCD – Its role is to read the number N and turns it into a binary number on 5 bits. It also send the number to the 7-segment display to be shown as it is introduced.

The two switches represent the 2-digit number N in base 10: switch 1 (denoted by N1) is the number of tens and switch 2 (denoted by N2) is the number of units. To enter the digit X, the respective switch must be activated for X+1 seconds, then it will be deactivated. In order to implement this component, we will need two counters that count the seconds the switches are activated and two registers that store these numbers for displaying them.

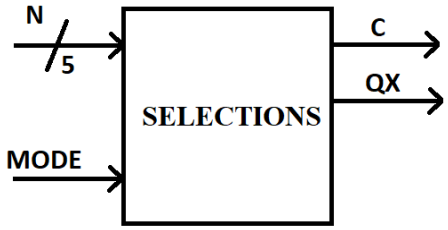
Ex: for N = 23, N1 will be active for only 2 seconds, and N2 for only 3 seconds. Number N will be converted in a 8 bits, binary number, but the most significant 3 bits will be ignored.



c. SELECTIONS – As stated in the introduction section, the connections in the LFSR will be made according to the performed selections.

The QX signal will have the value 1 if the flip-flop Q5 (for the 5-bit version) or the flip-flop Q4 (for the 4-bit version) is connected to gate B, otherwise it will take the value 0.

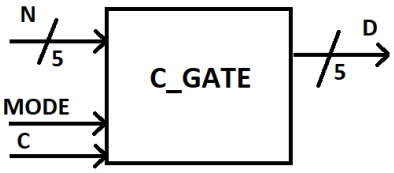
Similarly, only if the output signal C is active then gate C will be connected to the LFSR.



To determine the value of QX we will use two 16-bit and 32-bit multiplexers. On the 16-bit multiplexer, the data input of the numbers 4, 8, 12 and 15 will always be 1, the rest will be 0, and the selected input will be the number N represented on 4 bits. On the 32-bit multiplexer, the data input of the numbers 5, 10, 14, 16, 18, 22, 27 and 31 will always be 1, the rest will be 0, and the selected input will be the 5-bit number N. Finally, a 2:1 multiplexer can be used with select input MODE to return the appropriate QX.

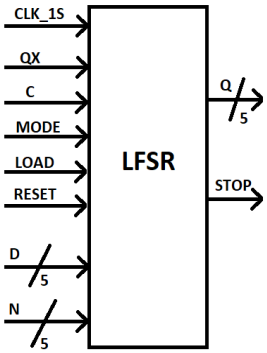
In order to determine the value of C we use two comparators on 4 or 5 bits. They will compare two 4 or 5 bit numbers, the first number being 1111 (or 11111), and the second N. The output C will be active only if the two numbers are different. At the end, it will be used a 2: 1 multiplexer, as previously.

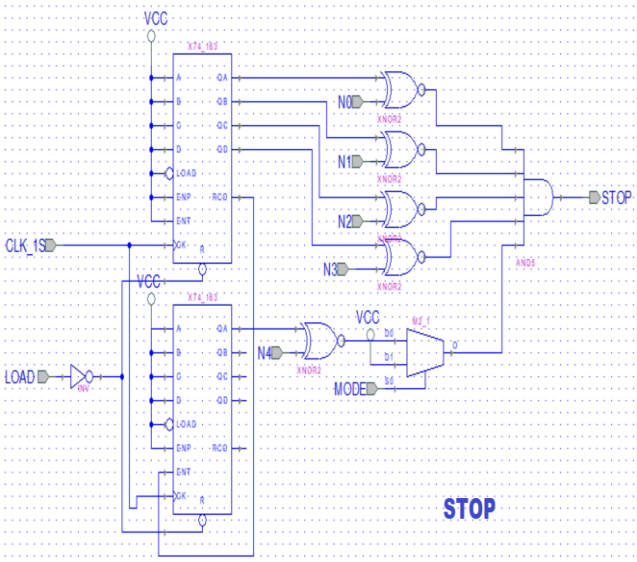
d. C_gate – This resource is used to connect the C gate to LFSR only if SELECTIONS returned 1 on C signal.

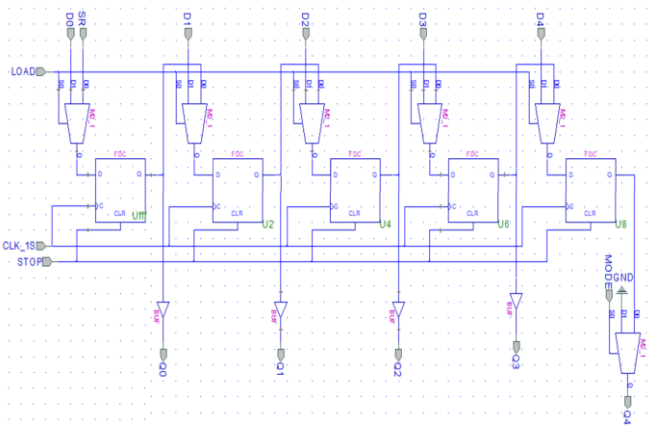


The D OUTPUT on 5 bits is generated according to the two tables presented in the introduction section and depends on the operation mode and the length N of the loop. It can be easily implemented by using two ROMs of 16x4 and 32x5 (which have as input the number N) to store the two tables. At the end, we can use a 2: 1 multiplexer on 5 bits with select input MODE to return D according to chosen number of bits.

e. LFSR - It returns the binary form of the numbers to be displayed, at an interval of one second, on the output Q. It is a linear feedback counter that, depending on the MODE input, counts on 4 or 5 bits. (It consists of five flip flops, the most significant being deactivated according to the values of the MODE input). It receives from SELECTIONS and MEMORIES the variables C, QX, D and returns a series of N numbers in relatively random order on 4 or 5 bits after the one-second signals of the clock. After completing the loop, the STOP signal is activated.







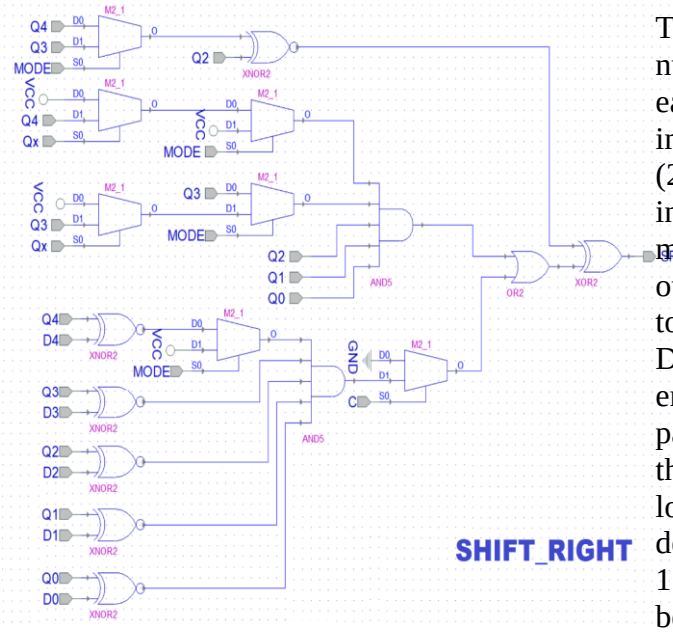
It has three components:

i) *A 5-bit register*. The most significant bit Q4 will be ignored (it will get the value 0 depending on the value of MODE). The register is loaded

with when the control unit activates the LOAD signal and immediately after it is deactivated it starts shifting right. After completing the modulo-N loop, END_CICLE will reset the register by activating the STOP signal.

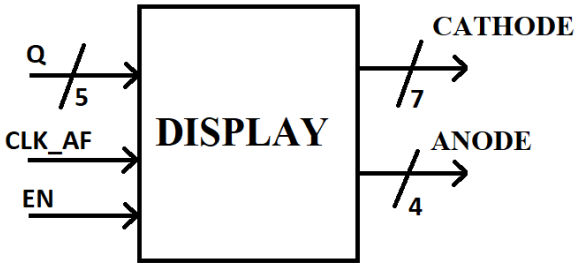
ii) END_CICLE -is a counter on 8 bits that counts up to N. When it reaches this number it returns STOP = 1 and stops the LFRS loop.

iii) SHIFT RIGHT- It computes SR according to the values calculated by the other resources (Qx, C and D(4:0)) and by MODE. (SR is a value of 0 or 1 which is returned by the SHIFT RIGHT)

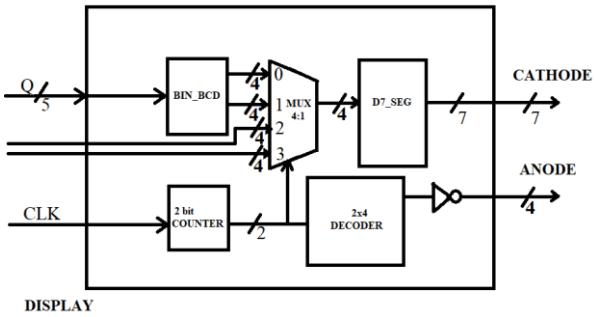


f. Display – Its role is to display numbers in decimal. It has 2 outputs on 7 bits, respectively 4 bits, which will be used to display numbers. It receives the binary number Q on 5 bits (if we have the 4-bit operation mode, the most significant bit Q5 will be zero) and turns it to BCD and then in 7 segments. Since all the numbers are less than or equal to 31, we will only need 2 displays (the most significant 2 bits of the ANODE will be 0).

In the next image, CLK_AF is a 250 hz clock.



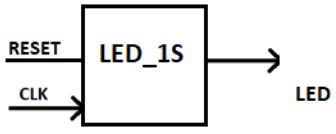
The display consists of several components:



The BIN_BCD component converts binary numbers on 5-bits to 8 bits BCD numbers, where each four bits represent a digit. For its implementation we can use a ROM of 5: 8 (25x8). After the numbers have been separated into two digits we can attach them to a 4:1t multiplexer on 4 bits. Its first two entries will be our number Q, and the other two will be attached to the numbers stored in the registers from the DCD so that the user can see the numbers he enters. The output of this multiplexer will be passed through a D_7SEG decoder that converts them from binary into 7 segments in the negative logic. To display the ANODE we need a 2x4 decoder in negative logic (values 0111, 1011, 1101, 1110). It counts up to 4 to exchange between the 4 digits that will be displayed on seven digit segments.

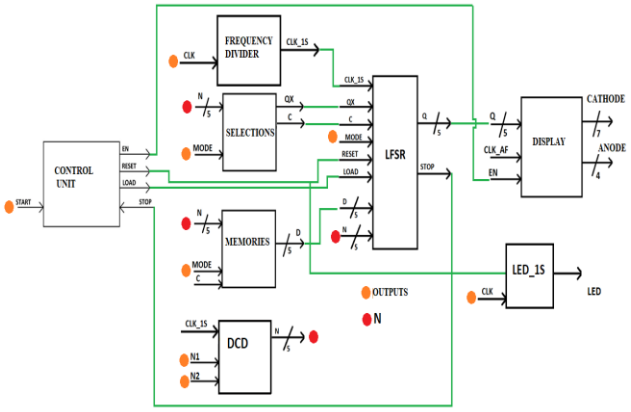
We will also need a two-bit counter COUNTER_2 that receives a 250 mhz clock and represents the multiplexer selections and the decoder input.

g. LED_1S - When CU returns 1 on RESTART, LED_1S holds this value for one second. It can be connected to a led so that the user is announced when the loop is finished.



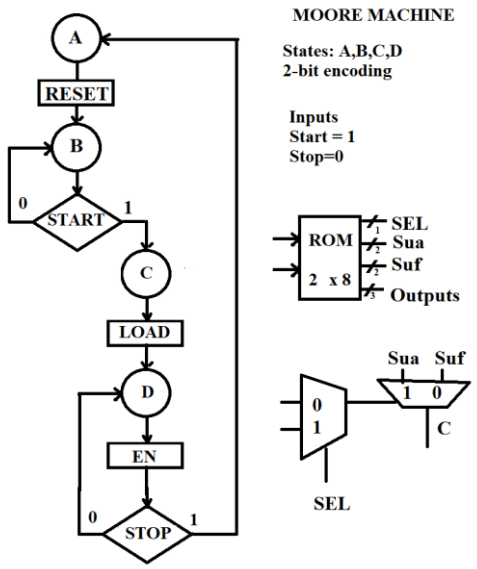
A detailed scheme of EU

The following image shows the final connections between all the resources of the EU and the connections between it and the UC.



2.2.2 The control unit will be done using a ROM 2x8 and two multiplexers 2:1, as seen in the down-right corner of the next figure. The states will be coded on two bits and the next table will be saved in the ROM in hexadecimal. Address is the code of the state, selection is the representation of the IF statement from the diagram, Sua and Suf are the code of the next state if the statement is true or false respectively and OUT is the concatenate values of the output variables (RESET, LOAD, STOP) after each state.

State	Address	Selection	Sua	Suf	Out	Hex
A	00	x	01	01	100	2C
B	01	0	10	01	000	48
C	10	x	11	11	010	7A
D	11	1	00	11	001	99



3. CONCLUSIONS

A linear feedback shift register (LFSR) is the heart of any digital system that relies on pseudorandom bit sequences (PRBS), with applications ranging from cryptography and bit-error-rate measurements, to wireless communication systems employing spread spectrum or CDMA techniques. In this paper, we present the implementation of a LFSR based counter on 4 and 5 bits. We hope that our detailed description of these implementations will be useful to all those who want to study this exciting field of electronics.

4. REFERENCES

[1] Cusick, T.W., Pantelimon S. Cryptographic Boolean functions and applications. Academic Press, 2017.

[2] Alfke, P., Efficient Shift Registers, LFSR Counters, and Long PseudoRandom Sequence Generators, XAPP 052 July 7,1996 (Version 1.1)

[3] Nedevschi S., Baruch, Z. F., Creț, O. Proiectarea sistemelor numerice folosind tehnologia FPGA, Mediamira, Cluj-Napoca, 1999, ISBN 973-9358-26-8

[4] Maxfield, C., *Bebop to the Boolean boogie: An unconventional guide to electronics*. Newnes, 2008.

[5] Marsaglia, G., "Xorshift RNGs". Journal of Statistical Software. 8 (14) 2003. doi:10.18637/jss.v008.i14

[6] Klein, A., "Linear feedback shift registers." Stream Ciphers. Springer, London, 2013. 17-58. ISBN 978-1-4471-507