

Title: Generation of Pseudo-noise Sequences without LFSRs

Author: Peter Lablans, Ternarylogic LLC.

Date: July 1, 2005

Pseudo-noise or pseudo-random sequences

PN sequences are well known. They are in general sequences of 0s and 1s with some very specific statistical properties. These types of sequences are for instance used in spread-spectrum technologies, and in watermarking. Non-binary, such as ternary or 4-valued, PN sequences are also possible.

The known way to generate PN-sequences is by means of Linear Feedback Shift Registers or LFSR circuits. An LFSR is a shift register with a certain number of memory elements, say p elements. On a clock signal the content of each element (a single digit) is moved to its immediate neighbor, except the content of the last element, which is then lost. In order to preserve the content of the last element, a connection is made between the last element and the first element of the shift register. This also solves the issue how to provide new content to the first element of the shift register. The following figure shows a 4 element shift register with feedback between the last and first element.

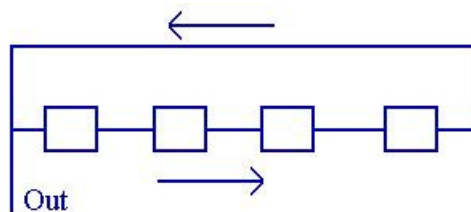


Figure 1.

One can generate a sequence on output Out (assuming that the system is binary) which will consist of 4 repeating bits. The pattern itself depends on the initial content of the shift register.

In order to create more variety in the output pattern one can create an additional connection between an output of one of the elements of the shift register and the output of the last element of the shift register. This connection is called a 'tap'. One cannot just connect the 'taps'. A 'functional' connection is needed. Though there are 16 binary functions only two of the 16 binary functions will not cause the sequence to degenerate quickly to a single signal or to a short cycle repetitive sequence. Those two binary functions are the XOR and the EQUAL functions.

Under the right conditions a shift register of p elements and with feedback through for instance a XOR function can create non-repeating sequences of $2^p - 1$ bits long. These sequences are known as maximum length sequences. In case of the example of Figure 2 the LFSR can generate an m-sequence of 15 bits. The binary sequence generator

is determined by the polynomial: $x^4 + x + 1$.

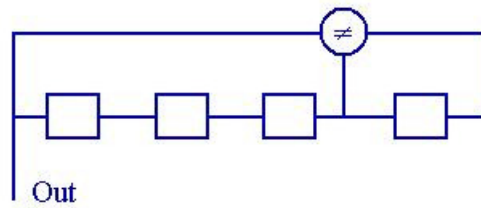


Figure 2.

The sequence generated by this circuit depends on the initial content of the shift register. The generator will produce an m-sequence, with 15 different but cyclically identical sequences. The only initial content of the shift register that is not allowed is the all 0, or [0 0 0 0] initial state. The explanation usually provided for the existence of the forbidden word is that [0 0 0 0] will cause no transition by the XOR function (as $0 \neq 0$ will generate a 0).

A different view on LFSRs: ‘word’ generators

In this article a different method will be developed to generate sequences. First of all the way the LFSR is viewed will be slightly altered. The diagram in as shown in Figure 2 will be flipped horizontally, so that the flow of signals is reversed. Also the sequence will be outputted at the end of the shift register. This is shown in Figure 3.

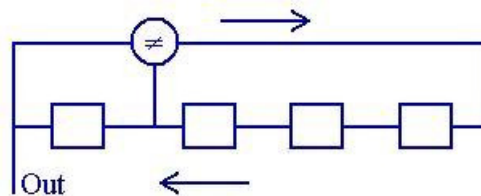


Figure 3.

This does not change how the circuit works. The non-repeating sequence generated by this LFSR with initial content [1 1 1 1] is [1 1 1 1 0 0 0 1 0 0 1 1 0 1 0].

While in general the LFSR is considered as a method to generate sequences it may also be considered a method to generate different ‘words’, wherein each ‘word’ is formed by the content of the shift register. The LFSR shown in Figure 3 may be considered a method to generate 15 consecutive different ‘words’ of 4 bits.

The 'words' have to comply with several conditions.

1. Clearly the starting 'word' cannot be [0 0 0 0], though that will be changed later.
2. Each 'word' has its last 3 bits in common with the first 3 bits of the next 'word'.
3. Each 'word' of 4 bits will only appear once in a complete cycle.
4. A complete cycle, using words of p bits, takes $2^p - 1$ 'words'.
5. After a complete cycle the original cycle starts again.

Under those conditions, and starting with for instance 4-bits 'word' [1 1 1 1] the following sequence table can be developed.

	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11	b12	b13	b14	b15	b1			
start	1	1	1	1															
word2		1	1	1	0														
word3			1	1	0	0													
word4				1	0	0	0												
word5					0	0	0	1											
word6						0	0	1	0										
word7							0	1	0	0									
word8								1	0	0	1								
word9									0	0	1	1							
word10										0	1	1	0						
word11											1	1	0	1					
word12												1	0	1	0				
word13													0	1	0	1			
word14														1	0	1	1		
word15															0	1	1	1	
																1	1	1	1
																	1	1	1
sequence	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	

The sequence is formed by the first bit of each 'word'. All 'words' occur just once, except the forbidden 'word'. One can generate a similar table with [1 1 1 1] as forbidden 'word'.

One can also create a table wherein no 'forbidden word' exists. Such a table then will contain 16 'words' in the 4-bits case. The sequence thus generated cannot be realized with an LFSR.

	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11	b12	b13	b14	b15	b16	b1		
start	1	1	1	1															
word2		1	1	1	0														
word3			1	1	0	1													
word4				1	0	1	1												
word5					0	1	1	0											
word6						1	1	0	0										
word7							1	0	0	1									
word8								0	0	1	0								
word9									0	1	0	1							
word10										1	0	1	0						
word11											0	1	0	0					
word12												1	0	0	0				
word13													0	0	0	0			
word14														0	0	0	1		
word15															0	0	1	1	
																0	1	1	1
																	1	1	1
sequence	1	1	1	1	0	1	1	0	0	1	0	1	0	0	0	0			

The generated sequence contains both the words [1 1 1 1] and [0 0 0 0].

The fact that a ‘word’ in the p-digit sequence generation method can only appear once offers opportunities in both the generation as well as the detection of sequences.

The methods here described also apply to n-valued sequences.

About Ternarylogic LLC: Ternarylogic LLC is a New Jersey based R&D company. Its mission is to create novel MVL technology. The company owns a portfolio of inventions related to scramblers/descramblers, sequence generators and sequence detectors, sequence correlators, gates and inverters based circuitry, non-binary multipliers, latches and other non-volatile memory elements, optical disk applications and MC-DSSS technology.

© 2005, Ternarylogic LLC. All rights reserved.