

Title: Robust Detection of Pseudo-Noise Sequences by Descramblers

Author: Peter Lablans, Ternarylogic LLC.

Date: June 8, 2005

LFSR based binary sequence generators

The most common way to generate binary sequences (especially maximum length or m-sequences) for applications requiring Pseudo-noise sequences is by way of Linear Feedback Shift Registers. The selection of the length of the shift register and the taps is a matter of finding the irreducible or primitive polynomials that one wants to realize. It is beyond the scope of this brief article to explain the theory of creating m-sequences. The book: Sequence Design for Communications Applications by Fan and Darnell is a good source. Actual usable polynomials can be found in the book as well on-line at:

<http://fchabaud.free.fr/English/default.php?COUNT=1&FILE0=Poly>

As an example the following figure 1 shows a binary sequence generator determined by the polynomial: $x^5 + x^2 + 1$.

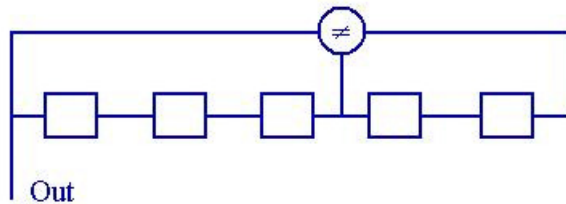


Figure 1.

The sequence generated by this circuit depends on the initial content of the shift register. The generator will produce an m-sequence, with 31 different but cyclically identical sequences. The generated 31 chips sequence with the initial content of the register = [1 0 1 0 0] is: [1 0 0 1 1 1 1 1 0 0 0 1 1 0 1 1 1 1 0 1 0 1 0 0 0 0 1 0 0 1 0 1]. Its auto-correlation graph is shown in Figure 2.

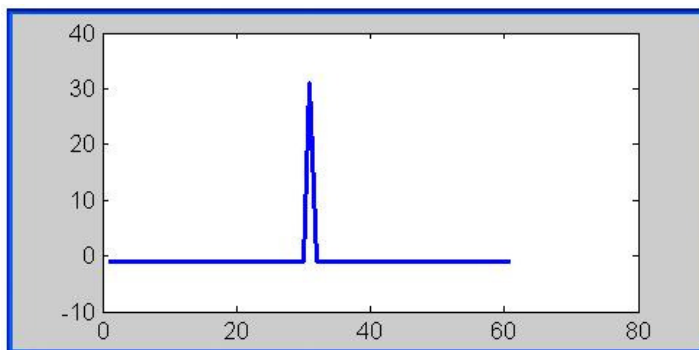


Figure 2.

Another binary m-sequence generator based on polynomial: $x^5 + x^3 + 1$ is shown in Figure 3.

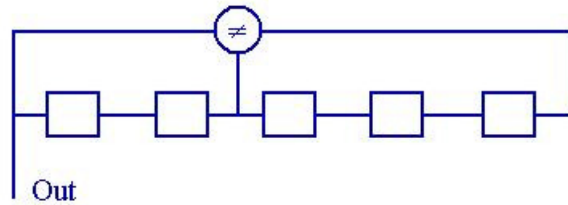


Figure 3.

The generated 31 chips sequence with the initial content of the register = [1 0 1 0 0] is: [0 1 1 1 0 1 1 0 0 0 1 1 1 1 1 0 0 1 1 0 1 0 0 1 0 0 0 0 1 0 1]. This sequence is different (even cyclically different) from the previous sequence. However its auto-correlation graph is the same as shown in Figure 2.

The cross-correlation graph of the two m-sequences is shown in Figure 4.

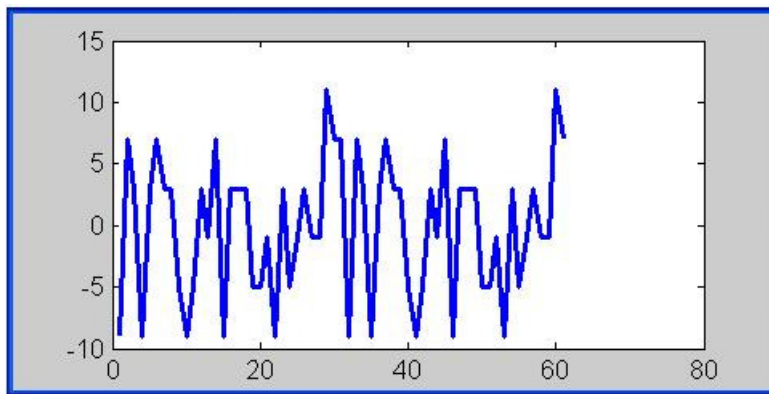


Figure 4.

The cross-correlation graph shows that one can use a correlator to detect an m-sequence and also distinguish between m-sequences. The benefit of using correlator circuits is that one can take advantage of orthogonality of sequences. However application of correlator circuits requires that locally a copy of expected sequences have to be available, and that the local copy of the sequence has to be in-phase with the expected sequence.

The descrambler as sequence detector

Elsewhere the author has shown that an LFSR based sequence generator may be considered to be a degenerated LFSR based scrambler, wherein a sequence of all identical elements is scrambled. In that case the generated sequence is the result of a scrambling process and may be descrambled by a corresponding LFSR descrambler.

A 'partial' descrambler may be constructed that is the mirror image of the sequence generator. The 'partial' descrambler for the sequence generator of Figure 1 is shown in Figure 5.

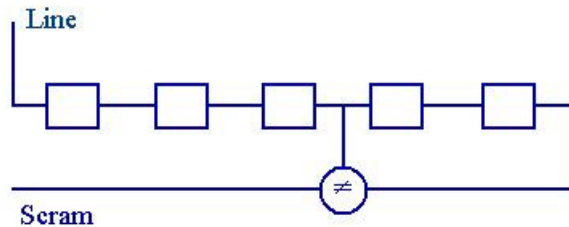


Figure 5.

The descrambler has as its input a signal Line. This signal enters the descrambler through the shift register, via a binary function XOR and provides a signal Scram. It can be shown that if the signal Line is identical to the signal Out as generated by the sequence generator of Figure 1 that Scram and Line are identical. If the signal Line is different from that sequence then Scram and Line will be significantly different.

This provides the way to insert an additional function to 'detect' the sequence. This detecting function is an EQUAL function with Line and Scram as inputs and Result as output. When Line and Scram are identical then Result will be all 1s. The detecting configuration is shown in Figure 6.

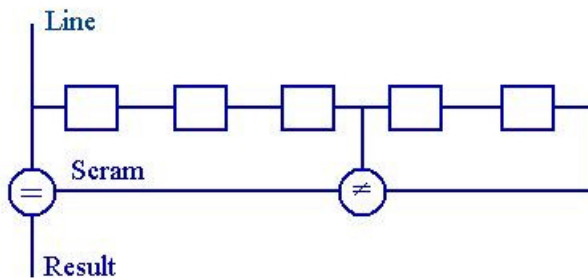


Figure 6.

So when the signal Line is identical to the sequence Out generated by the LFSR based sequence generator of Figure 1 the signal Result will be a sequence of all 1s. One can replace the EQUAL function by an XOR function. In that case the detector will be a common descrambler and generate an all 0s sequence. This indicates that the LFSR sequence generator is equivalent with a common LFSR scrambler with an all 0s sequence as input.

The output signal Result of the circuit of Figure 6 when signal Line is generated by the circuit of Figure 3 is [0 0 1 0 1 0 1 1 1 1 0 1 1 0 1 0 0 1 1 0 0 0 0 0 1 1 1 0 0 1 0]. This ‘descrambled’ sequence clearly does not consist of all 1s and will not lead to a ‘detected sequence’.

Flushing the shift register

The LFSR based descrambler is self-synchronizing. This means that no synchronizing signal is required to establish, maintain or re-establish the correct output. In the previous section it was assumed that the initial content of the shift registers of the sequence generator as well as the descrambler were identical. This identity is not really required. If in actuality the shift registers have different contents, then the detected sequence can only go wrong for a number of chips or bits that is equal to the length of the shift register (which is 5 in the illustrative example).

One can say that the incoming signal ‘flushes’ the shift register of the descrambler of its previous content. This effect can also be used when substantial line errors are expected. The length of the expected sequence should be an order of magnitude larger than the expected number of error-affected bits. The effects of an error will be flushed through the shift register and cannot propagate to the rest of the descrambled signal beyond the length of the shift register.

Persons skilled in the art will now also recognize that certain patterns can be inserted that will allow for simple means of synchronization for instance for consecutive sequences, representing the same symbol.

For example one can insert a special sequence of bits to the rest of the sequence which, when correctly detected, will generate a predetermined pattern, while the correctly detected rest of the sequence will generate all 1s. This allows the shift register to be flushed before the all 1s sequence starts as well establish an objective start for the symbol represented by the sequence.

These and other aspects of the generation, processing and detection of binary and non-binary digital sequences by way of LFSR based methods and apparatus are described in Patent Applications submitted to the United States Patent and Trademark Office.

United States Patents and Published Pregrant Patent Applications can be viewed at www.uspto.gov .

Other sources and articles on sequence detection and synchronization

The performance of binary descramblers is well documented. Its self-synchronizing properties were already described in the original Bell Labs patent **4,304,962**.

The deterministic character of the LFSR based scrambler/descrambler is further described in Bell Labs book: Transmission System for Communications on pages 742 and 743.

An interesting publication is United States Patent Application **20050071399** by Bonaccia et al. of IBM. It is entitled: **Pseudo-random binary sequence checker with automatic synchronization**. What the inventors call a “sequence checker” is in fact a common LFSR descrambler. The descrambler there is used to detect bit-errors, as it is assumed to operate as the counterpart of a corresponding binary LFSR sequence generator. The overall purpose of that invention is to check for errors within a known received sequence which was transmitted over for instance an optical transmission line. This is of course different from detecting a particular sequence within a set of 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.