Pseudo noise sequences for engineers

by R.N. Mutagi

Pseudo noise (PN) sequences are widely used in digital communications and the theory involved has been treated extensively in the literature. However, a practising engineer is interested in the fundamentals and the applications of PN sequences, and the methods of generating them with hardware. This paper presents, without the mathematical rigours, some of the interesting characteristics and the use of these characteristics in the generation and acquisition of PN sequences. The series–parallel method of generating PN sequences at high speeds with low-speed devices, which is of interest to hardware designers, is discussed. Some applications of PN sequences in communications and instrumentation are discussed.

1 Introduction

Pseudo random binary sequences (PRBSs), also known as pseudo noise (PN), linear feedback shift register (LFSR) sequences or maximal length binary sequences (m-sequences), are widely used in digital communications. In a truly random sequence the bit pattern never repeats. However, generation of such a sequence is difficult and, more importantly, such a sequence has little use in practical systems. Applications demand that the data appear random to the channel but be predictable to the user. This is where the PRBS becomes useful. A pseudo random binary sequence is a semi-random sequence in the sense that it appears random within the sequence length, fulfilling the needs of randomness, but the entire sequence repeats indefinitely. To a casual observer the sequence appears totally random, however to a user who is aware of the way the sequence is generated all its properties are known. PN sequences have several interesting properties, which are exploited in a variety of applications. Because of their good autocorrelation two similar PN sequences can easily be phase synchronised, even when one of them is corrupted by noise. A PN sequence is an ideal test signal, as it simulates the random characteristics of a digital signal and can be easily generated. An exhaustive mathematical treatment is available in Reference 1. This paper describes some interesting properties, methods of generation, including the series–parallel method for high-speed generation, and some applications of PN sequences.

2 Properties of PN sequences

A PN sequence is a bit stream of '1's and '0's occurring randomly, or almost randomly, with some unique properties. These properties are used in digital communications, instrumentation and measurements. The sequence serves as a reference pattern with known random characteristics for the analysis, optimisation and performance measurement of communication channels and systems.

Run length

In a PN sequence of any length the numbers of '1's and '0's differ only by one, i.e. the number of '1's is just one more than the number of '0's. For example, the PN sequence of length 15 (=[2^4 – 1]) contains eight '1's and seven '0's. A sequence of consecutive '1's, or '0's, is called a 'run' and the number of '1's and '0's is the run length. A PN sequence of length $2^N - 1$ contains one run of N '1's, and one run of $N-1$ '0's. The number of other runs, $N-2$ to 1, of '1's and '0's increases as the power of 2, as shown in Table 1.

The carrier and clock recovery circuits in a digital communication system have pattern-dependent behaviour and hence their performance can be evaluated using long PN sequences providing long runs of '1's and '0's.

Table 1: Number of runs of '1's and '0's of various lengths in a PN sequence of length $2^N - 1$

Run length	'1's	'0's
N	1	0
$N-1$	0	1
$N-2$	1	1
$N-3$	2	2
$N-4$	4	4
..	.	.
..	.	.
2	2^{N-4}	2^{N-4}
1	2^{N-3}	2^{N-3}

R. N. Mutagi received a BE degree in Tele-communications Engineering from Karnatak University in 1971 and a DIISc degree in Electronics Design Techno-logy from the Indian Institute of Science, Bangalore in 1976. He joined the Electronics Systems Division of the Indian Space Research Organization at Ahmedabad in 1972. In 1976 he joined the Digital Communi-cation Division of the Space Applications Centre (SAC) where he was involved with the first Indian experimental communication satellite, APPLE. He became Head of the Speech Processing Section at SAC in 1985 and developed digital speech interpolation equipment, an echo canceller, a speech pattern analyser, an ADPCM transcoder and low-rate speech coders. Since 1991 he has been Head of the Baseband Processing Division at SAC. His current interests are in digital communications, digital video and digital signal processing. He also teaches 'Satellite communications' and Digital signal processing' to undergraduate classes.

Shift and add

When a PN sequence is shifted and the shifted sequence modulo-2 added to the unshifted sequence with an exclusive-OR gate, the result is the same PN sequence with some other shift. This is illustrated in Fig. 1, where a 15 bit PN sequence, $a(k)$, is arbitrarily shifted by 4 bits to get, $a(k-4)$. The two sequences when modulo-2 added give a sequence which is a 3 bit shifted version, $a(k-3)$, of the original sequence $a(k)$. Only when the PN sequence is modulo-2 added to itself without shift is the result a sequence of zeros. A direct application of this property is in the generation of two identical sequences with a known, large delay between them.

Correlation

Correlation is a measure of similarity between two sequences. When the two sequences compared are different it is the 'crosscorrelation' and when they are the same it is the 'autocorrelation'. Mathematically, the correlation between two sequences $x(k)$ and $y(k)$ as a function of the time delay m is expressed as

$$R(m)_{xy} = \sum_{k=0}^{L-1} x(k)\, y(k+m) \qquad (1)$$

For a digital bit stream the above equation can be interpreted in a simple way. $x(k)$ and $y(k)$, being digital bits, have binary values '0' or '1'. The product, therefore, is simply the modulo-2 addition. This is equivalent to exclusive NOR (XNOR) operation. Hence, the correlation of two serial bit sequences, such as PN sequences, is obtained over a length L by comparing the two binary sequences bit-by-bit using XNOR gates, and counting the places they match. Thus, if the two bits are identical the XNOR produces a '1' at the output, otherwise a '0'. The correlation is then obtained by counting all the '1's and dividing the sum by the total number of bits, L, in the sequence, or the part of the sequence over which the correlation is calculated. The correlation equation for the digital bit sequence can thus be written as

$$R(m) = \frac{\text{number of agreements}}{\text{total number of bits}} \qquad (2)$$

In some literature the numerator in the above equation is expressed as the difference of the number of agreements and the number of disagreements. However, eqn. 2 will be used here because it agrees with the general correlation eqn. 1, and also because the implementation is straight-

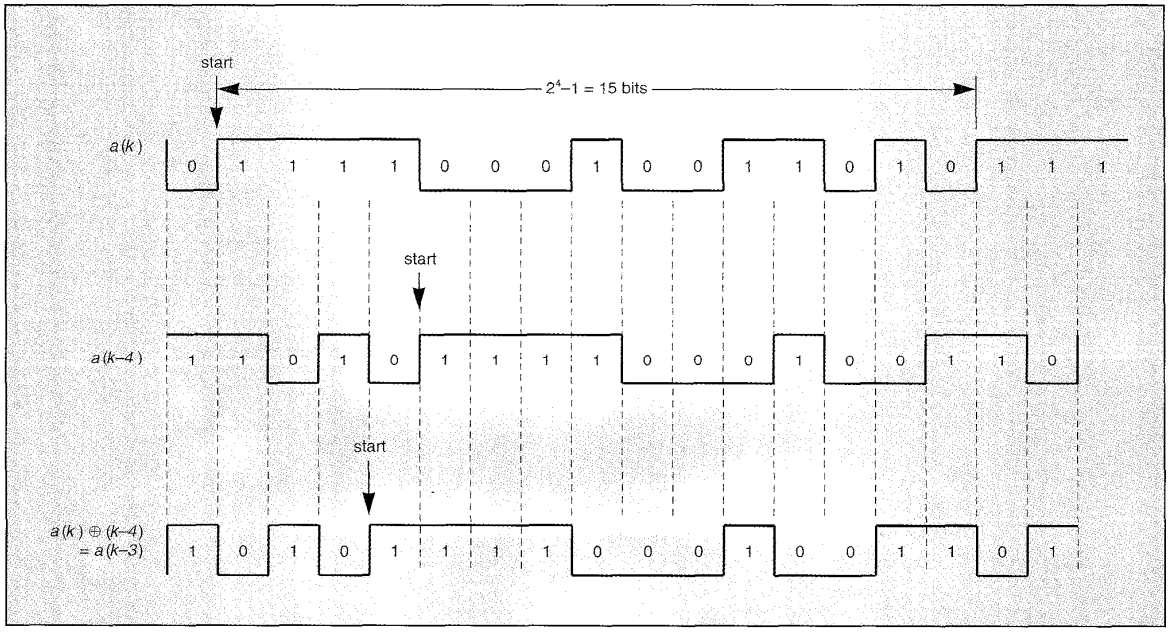


Fig. 1 Addition of PN sequences

forward as there is no need to form the difference of two sums. The difference in the results using the two definitions is only in the bias obtained in the correlation for all time lags m, where $m \neq 0$.

To find the correlation over the entire sequence length one of the sequences is fixed while the other is shifted, and the correlation is obtained at different positions. Fig. 2 shows a correlator of length K. One sequence, a_i, is shifted through a K-bit shift register and the output of each stage is applied to a set of K XNOR gates for comparison. The other inputs of the XNOR gates receive the second sequence, b_i. The output of an XNOR gate, working as a modulo-2 adder, is '1' if both inputs are the same and '0' otherwise. Summing these outputs in a K-input adder gives the correlation output, which may be divided by K to get a normalised value.

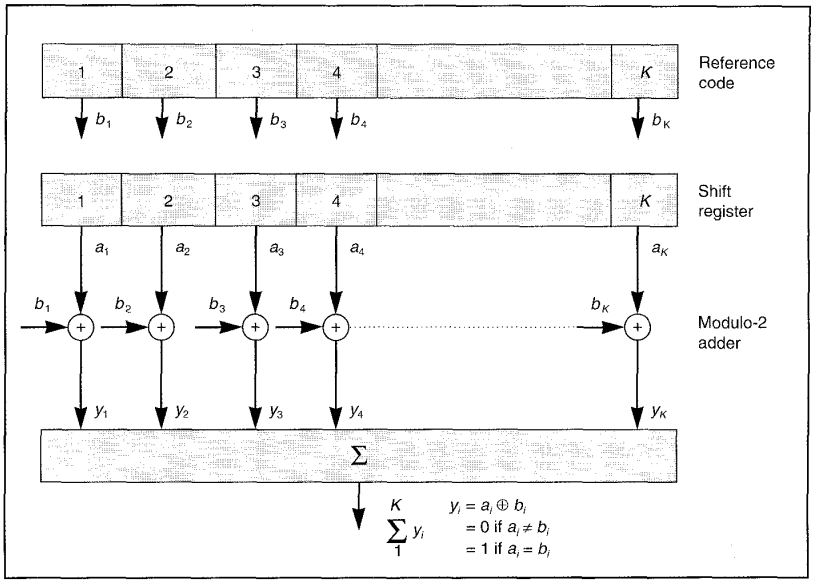


Fig. 2 A correlator

When the sequences a_i and b_i are different the sum, $\sum y_i$, is a crosscorrelation, otherwise it is an autocorrelation. When the PN sequence is autocorrelated we get $\sum y_i = K$ when both sequences are in-phase. When there is a shift of even one bit in either direction the PN sequence does not match with its shifted version and y_i forms the same sequence with a different phase (recall the 'shift and add' property discussed earlier). For large K the number of '0's and '1's is almost equal to $K/2$, giving a normalised correlation of 1/2. This is true even for one sequence length, $L = 2^N - 1$. Thus, a PN sequence has an autocorrelation of 1 at zero phase (no time shift), and 0·5 at all other phases. The discrete autocorrelation values obtained for a PN sequence of length $2^4 - 1$ is shown in Fig. 3. At zero shift, and shifts which are multiples of 15 bits, the correlation value is 15; for all other shifts in either direction it is 7. This property is used in the synchronisation, or acquisition, of PN sequences.

Subsequences

Another interesting property of PN sequences is that the alternate bits in a sequence form the same sequence at half the rate. Consider the 15 bit sequence obtained from a 4-stage feedback register. The middle waveform in Fig. 4

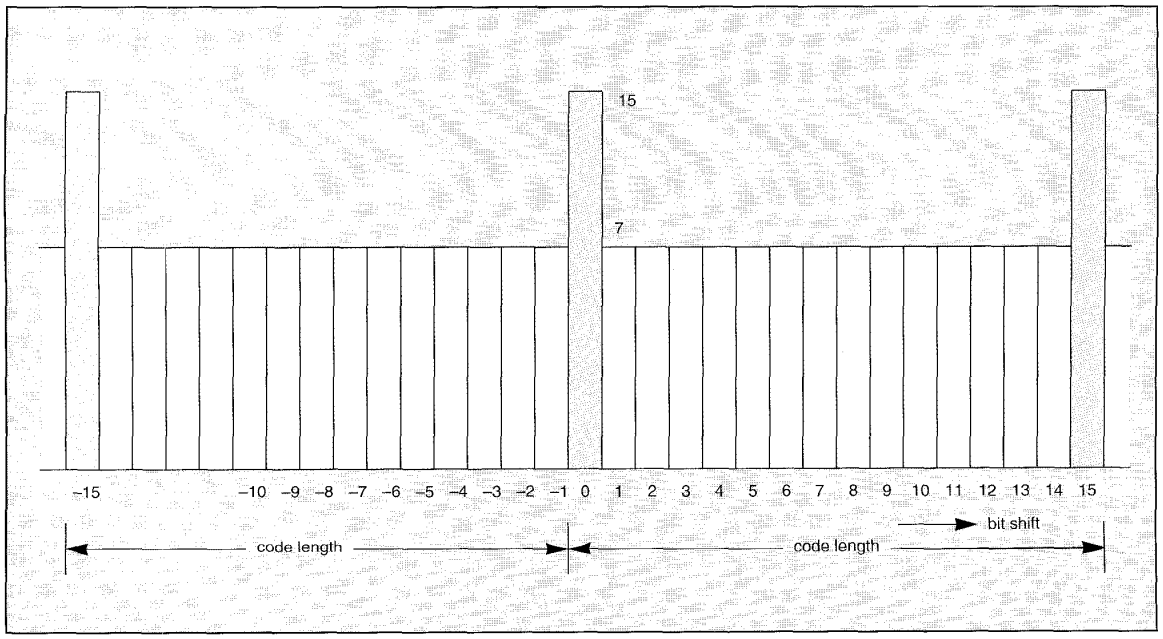


Fig. 3 Discrete autocorrelation of a PN sequence of length $2^4 - 1$

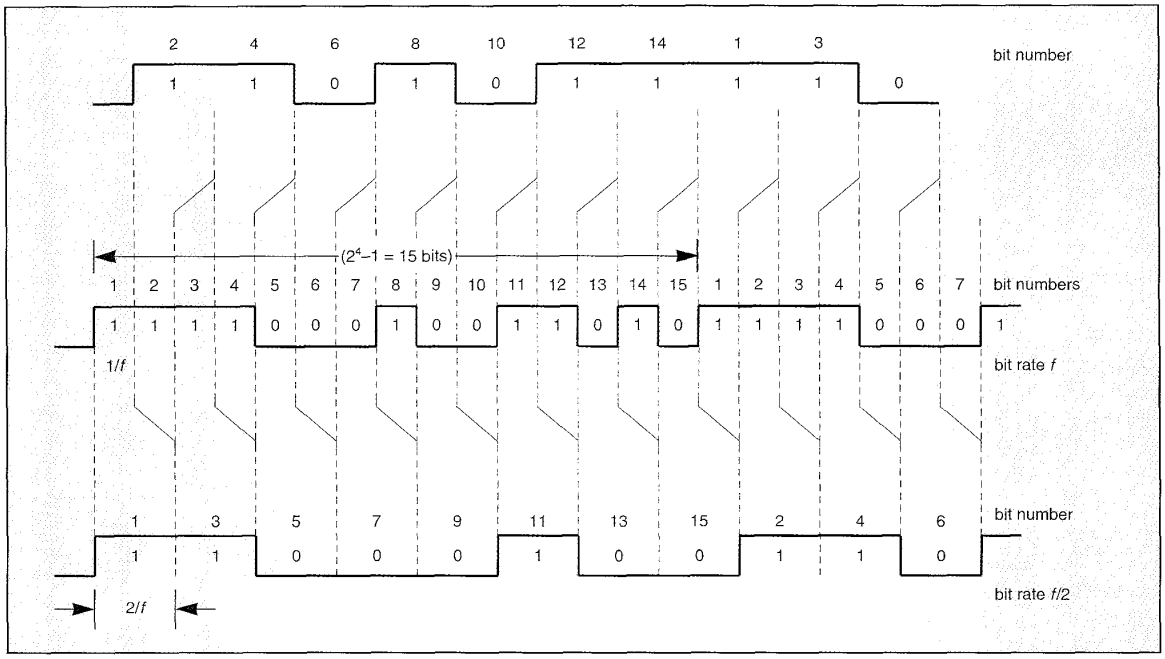


Fig. 4 Subsequences of a PN sequence

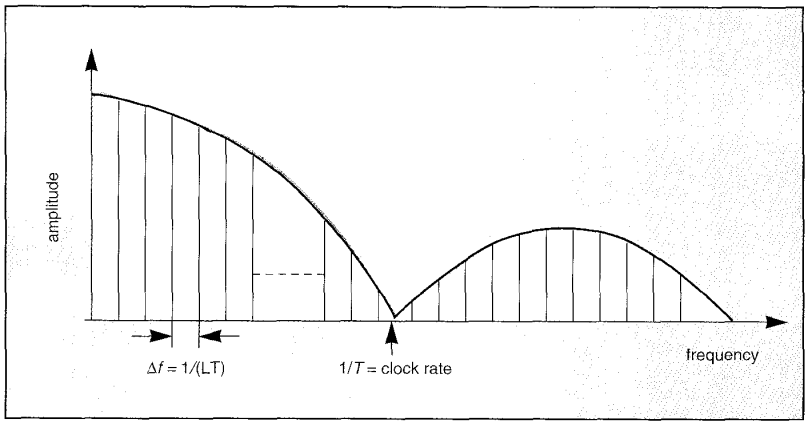
shows two cycles of the sequence repeating at 15 bits at clock rate f. The bottom waveform shows a sequence formed by taking only the alternate bits and stretching them over two bit periods. This sequence is easily recognised as the same sequence as the one in the middle, but at a clock rate $f/2$. Another sequence is formed by taking the remaining bits, as shown in the waveform at the top. This is again the same sequence at $f/2$. The sequences at the top and bottom are obtained at half the rate by decimating the sequence in the middle. The phase shift between these two sequences is 7½ clock cycles at $f/2$, which is half the sequence length.

This principle may be extended further to get higher order decimation. A sequence may be decimated by a factor R, where R is a power of 2, to obtain R subsequences each at rate f/R. For example, a sequence may be decimated into 8 subsequences at rate $f/8$. When R is even but not a power of 2, if the resulting sequence is a prime length sequence, then such a sequence may still be decimated to R subsequences. However, these sequences are the mirror images of the original sequence.

Spectra of a PN sequence

The power spectrum of an N-stage PN sequence has a $\sin^2 x/x^2$ envelope, as shown in Fig. 5. The nulls in the spectrum occur at $f = n/T$, where T is the bit duration and n is an integer. From 0 to $1/T$ the spectrum has all the line frequencies with a spacing of $1/(2^N - 1)T$ Hz. Thus, the frequency spacing can be reduced by choosing a longer PN sequence. Due to the nearly uniform density within the band of $2/T$ Hz, and the presence of all components, the PN sequence is an ideal test signal, and is widely used in data communications to simulate the normal signals. An analogue noise source, with white Gaussian characteristics, can be built with a long PN sequence generator and a D/A converter, as shown in Fig. 6. By modulo-2 addition of a PN sequence with any other data the

Fig. 5 Spectra of a PN sequence



resulting sequence will also have the same spectrum as that of the former. This property is used in scrambling the data.

3 Generation of PN sequences

Shift register with feedback

A PN sequence is generated using a shift register and modulo-2 adders. Certain outputs of the shift register are modulo-2 added and the adder output is fed back to the register. An N-stage shift register can generate a maximal length sequence of $2^N - 1$ bits. Only certain outputs, or taps, can generate a maximal length sequence. The generator output is expressed as a polynomial in 'x'. For example, the polynomial $1 + x^{14} + x^{15}$ means that the outputs from stages 14 and 15 are modulo-2 added and fed back to the input of first stage of a 15-stage register to get a $2^{15} - 1$ length sequence, as shown in Fig. 7.

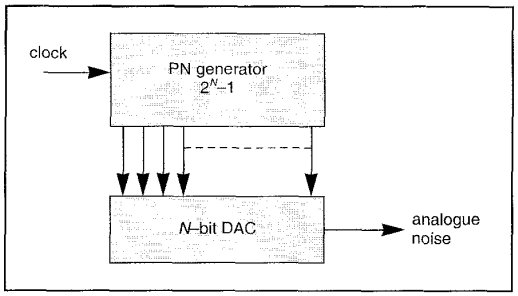


Fig. 6 An analogue noise source

Number of PN codes

With an N-bit shift register more than one sequence of length $L = 2^N - 1$ can be generated using different taps for feedback. Each sequence is different from the others, although of the same length. Table 2 gives the sequence length, L, for register lengths, N, up to 25, the maximum possible number of PN codes, C, and some of the taps to be used for feedback.[2] One interesting observation is that for a given tap set there is a mirror that also provides a maximal length sequence. For example, the mirror of {5, 3} is {5, 2}, that of {6, 5} is {6, 1} etc. In the mirror set the first number, which is the register length N, is fixed. The other numbers in the mirror are obtained by subtracting the numbers in the original set from N.

The maximum number of PN codes that can be generated with an N-bit register is of interest in spread-spectrum communication, where each user must use a different code. For a sequence length L, the maximum number, C, of possible codes is given by

$$C = \frac{1}{N}\prod\{P_i^{(\alpha_i - 1)}(P_i - 1)\} \qquad (3)$$

where P_i are the prime factors of L and α_i is the power of the ith factor. For example, for $N = 6$ the code length $L = 63$, which can be factored into $3 \times 3 \times 7 = 3^2 \times 7$, giving $P_1 = 3, \alpha_1 = 2$, $P_2 = 7$ and $\alpha_2 = 1$. Hence, the maximum number of codes is

$$C = (1/6)\{(3^{2-1})(3-1)\}\{(7^{1-1})(7-1)\} = 6$$

When L is factorable with i distinct factors, i.e. $\alpha_i = 1$ and $L = \prod P_i$, then the number of PN codes obtained is given by

$$C = \prod(P_i - 1)/N \qquad (4)$$

For example, for $N = 8$, the length 255 is factored into 3, 5 and 17. Hence the maximum number of PN codes is

$$C = (3-1)(5-1)(17-1)/8 = 16$$

For prime length sequences, i.e. when L is a prime number, shown by asterisk in Table 2, the number of possible tap sets, and hence the number of different codes, is given by

$$C = (2^N - 2)/N \qquad (5)$$

For example, for $N = 7$ there are $(2^7 - 2)/7 = 18$ possible tap sets, half of them being the mirrors of the remaining half. Not all the outputs which can be used for feedback are shown in Table 2.

Avoiding the zero state

An N-bit register can generate $2^N - 1$ states as against the 2^N states of a binary counter, as shown in the state diagram in Fig. 8. Although the counter states generate an ascending or descending sequence, the PN generator output states are apparently random. The all-zero state is missing in the PN sequence. This state is inhibited, because the generator remains latched to it. The modulo-2 adder in the feedback circuit feeds only '0's to the input. An additional circuit is needed to detect the 'all zeros' state and

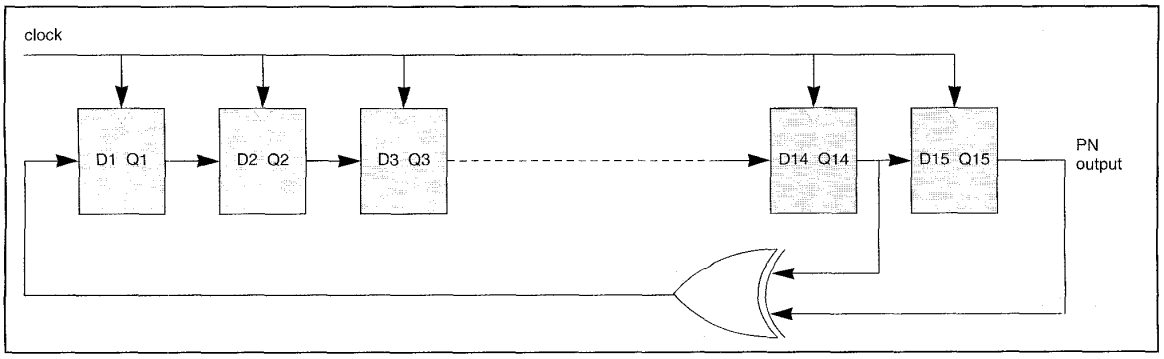


Fig. 7 A PN generator with polynomial $1 + x^{14} + x^{15}$

Table 2: Some feedback taps for maximal sequence length

No. of stages, N	Code length, L	No. of codes, C	Some tap sets used for feedback
2*	3	1	[2, 1]
3*	7	2	[3, 2] [3, 1]
4	15	2	[4, 3] [4, 1]
5*	31	6	[5, 3] [5, 2]
6	63	6	[6, 5] [6, 1]
7*	127	18	[7, 6] [7, 3] [7, 1]
8	255	16	[8, 6, 5, 4] [8, 6, 5, 3]
9	511	48	[9, 5] [9, 6, 4, 3]
10	1 023	60	[10, 7] [10, 3]
11	2 047	176	[11, 9] [11, 8, 5, 2]
12	4 095	144	[12, 6, 4, 1]
13*	8 191	630	[13, 4, 3, 1]
14	16 383	756	[14, 5, 3, 1]
15	32 767	1 800	[15, 14] [15, 4]
16	65 535	2 048	[16, 15, 13, 4]
17*	131 071	7 710	[17, 14] [17, 3]
18	262 143	7 776	[18, 11[[18, 7]
19*	524 287	27 594	[19, 6, 2, 1]
20	1 048 575	24 000	[20, 17] [20, 3]
21	2 097 151	84 672	[21, 19] [21, 2]
22	4 194 303	120 032	[22, 21] [22, 1]
23	8 388 607	356 960	[23, 18] [23, 5]
24	16 777 215	276 480	[24, 23, 22, 17]
25	33 554 431	1 296 000	[25, 22] [25, 3]

reset the PN register to a valid state.

When the register length, N, is small, a NOR gate can decode '0' outputs of the register, forcing a '1' to the feedback input, through an OR gate, as shown in Fig. 9*a*. Here, the first four outputs of a 5-bit register are applied to a 4-input NOR gate, which decodes a sequence of four '0's and provides '1' at its output. The output of the XOR gate is ORed with the NOR output and then fed back. This circuit prevents the 'all zeros' state of the sequence. Note that the last bit is not decoded. When the length N is large, more logic is needed with this technique to decode 'all zeros'.

A modulo-N down-counter may be used for decoding the 'all zeros' state in the PN sequence for large N. The PN data is applied to the load input after inversion, as shown in Fig. 9*b*. The '1's in the sequence load the counter to N, and the '0's are counted. With N consecutive zeros the counter generates a carry, forcing a '1' through an OR gate to the input of the register. Decoding $N - 1$, instead of N, zeros has an additional advantage as it provides a sync pulse which can be used to trigger an oscilloscope or to monitor the sequence length. The one-clock-wide max/min output of the counter is preferred to the carry.

Series–parallel method for high-speed PN generation

The maximum PN data rate depends on the type of logic device used. Since only one gate delay (due to the XOR gate) is introduced in the feedback path the maximum PN rate can be close to the highest operating frequency of the shift register. The operating frequency of the PN sequence can be pushed beyond the shift register clock rate by using a

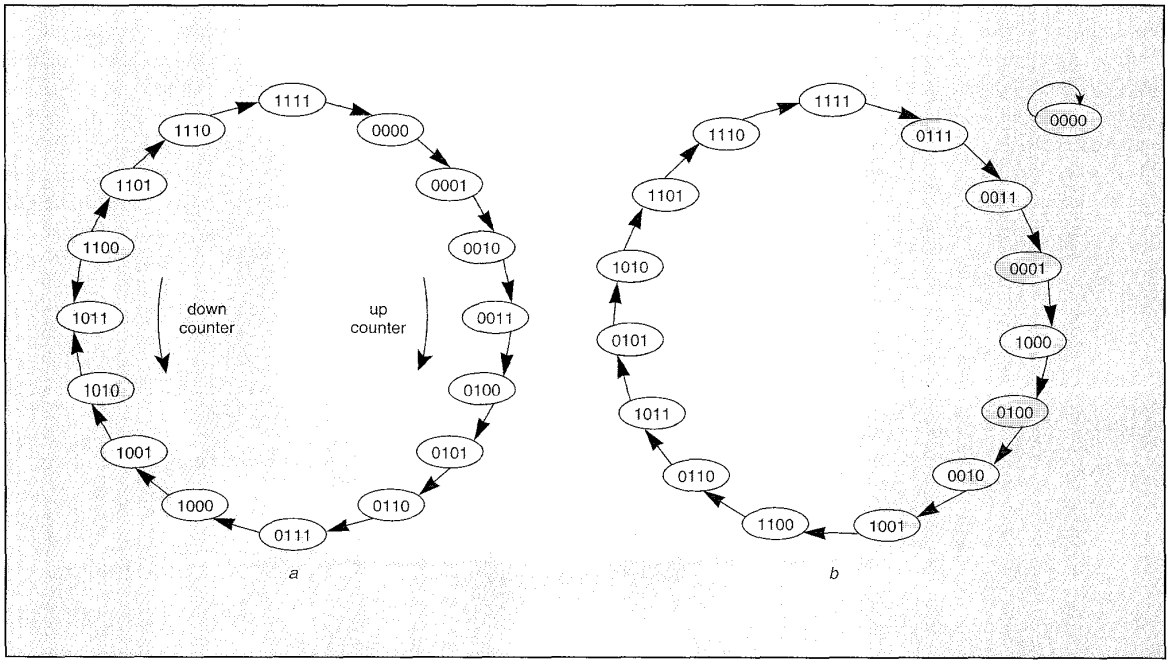


Fig. 8 State diagrams for (*a*) binary counter and (*b*) PN generator

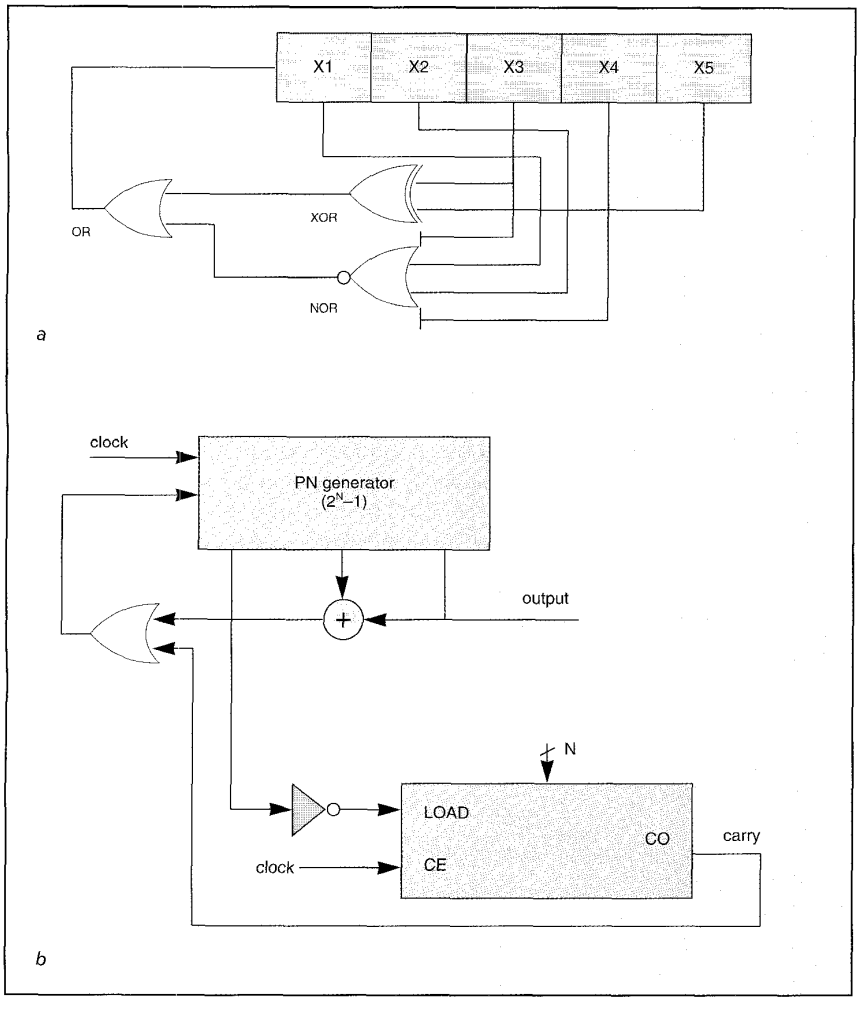


Fig. 9 *(a)* **A PN generator with 'all zeros' decoder; *(b)* 'all zeros' decoding with a counter**

high-speed multiplexer. This technique uses the 'subsequence' property of a PN sequence already discussed. As we can demultiplex a PN sequence into two similar sequences at half the rate, we can also multiplex two PN sequences to obtain a sequence at double the rate. However, the two sequences should necessarily have a phase shift of half the sequence length. For example, to generate a PN sequence of length $L = 15$ the two sequences at the lower rate should be offset by 7½ clock cycles. This can be easily achieved, for small L, by using another shift register. However, the register length becomes prohibitive for large L.

A systematic approach for obtaining decimated sequence is given by J. J. O'Reilly.[3] Using this method any PN sequence can be decimated to k parallel sequences generated at rate R/k, which can then be multiplexed to obtain the sequence at rate R. The design procedure is as follows.

First the generation polynomial, or the characteristic equation, of order n for the desired PN sequence is written as an n-dimensional vector in which each bit represents the coefficient '1' or '0' by which the corresponding output of the shift register is multiplied and modulo-2 added to get the feedback input. For example, for a PN sequence of length 15, the vector is 0011. An $n \times n$ transition matrix, T, whose rows define the excitation for each of the n stages of the register is written as below. In this, each element represents the Q output of the corresponding flip-flop. The outputs with a 1 in the characteristic equation, are modulo-2 added and applied to the D input of the flip-flop corresponding to the row.

$$
\begin{array}{c|cccccccc|l}
 & Q_1 & Q_2 & Q_3 & . & . & . & . & Q_n & \\
\hline
 & t_1 & t_2 & t_3 & . & . & . & . & t_n & D_1 = |\Sigma| t_i \\
 & 1 & 0 & 0 & . & . & . & . & 0 & D_2 = Q_1 \\
T = & 0 & 1 & 0 & . & . & . & . & 0 & D_3 = Q_2 \\
 & 0 & 0 & 1 & . & . & . & . & 0 & D_4 = Q_3 \\
 & . & . & . & . & . & . & . & . & . \\
 & 0 & 0 & 0 & . & . & . & 1 & 0 & D_n = Q_{n-1}
\end{array}
$$

where $|\Sigma|$ indicates modulo-2 summation.

For decimating an n-stage sequence by a factor k the characteristic equation is shifted to the kth row in the T matrix. The rows $k-1$, $k-2$, ... 1 are obtained by shifting the elements to the left successively. The diagonal array of '1's is shifted below the kth row. This procedure is demonstrated in the following example for a PN sequence is demonstrated in the following example for a PN sequence with 7 stages and decimation by 4 ($k = 4$). The

transition matrix for this application is shown below:

$$T^4 = \begin{array}{|ccccccc|l|} \multicolumn{7}{c}{Q_1\ Q_2\ Q_3\ Q_4\ Q_5\ Q_6\ Q_7} & \\ \hline 0 & 0 & 1 & 1 & 0 & 0 & 0 & D_1 = Q_3 \oplus Q_4 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & D_2 = Q_4 \oplus Q_5 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & D_3 = Q_5 \oplus Q_6 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & D_4 = Q_6 \oplus Q_7 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & D_5 = Q_1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & D_6 = Q_2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & D_7 = Q_3 \\ \hline \end{array}$$

The four parallel decimated outputs are available at Q_7, Q_6, Q_5, and Q_4. These outputs can be multiplexed to get the PN sequence of length $2^7 - 1$ at four times the clock rate, as shown in Fig.10. A design approach, based on this technique, for doubling the clock rate of the PN generator is described by the author in Reference 4.

4 Acquisition of PN sequences

In many applications a PN generator at the receiver needs to be synchronised to the received sequence generated at the transmitter. The autocorrelation property of the PN sequence can be used for synchronisation. Since the two sequences are generated by identical generators they are similar in all respects, except the phase. By comparing the sequences bit-by-bit the phase match can be checked. If the received sequence is in-phase with the generated sequence, then all the bits, except those corrupted by noise, match. If the sequences are out of phase, the comparison gives a mismatch of almost half the bits. The PN generator can then be inhibited for one clock cycle and the sequence can be compared again. This process can continue until the sequences are synchronised. A maximum of $2^N - 1$ searches are needed in this method for synchronisation.

A faster method for synchronisation is shown in Fig. 11.[5] Initially the switch at the input of the shift register is kept in position A, allowing N bits of the received sequence into the register. Then the feedback loop is closed by putting the switch in position B. If the bits entered in the register contain no error the sequence generated locally is synchronous to the received sequence and the modulo adder shows no errors. If the two sequences are not synchronised the modulo-2 adder gives about 50% errors which can be counted with a counter for a fixed interval. The counter output is then compared with a threshold in a magnitude comparator. Only channel errors occur when the sequences are synchronised; the threshold value can be chosen depending on the channel error rate expected. For example, for a channel error rate of 10^{-1}, one error in every 10 data bits is expected and so 10 errors can be present in a measuring period of 100 bits when the sequences are synchronised; otherwise there are about 50 errors. Hence a threshold of 25 may be chosen for this

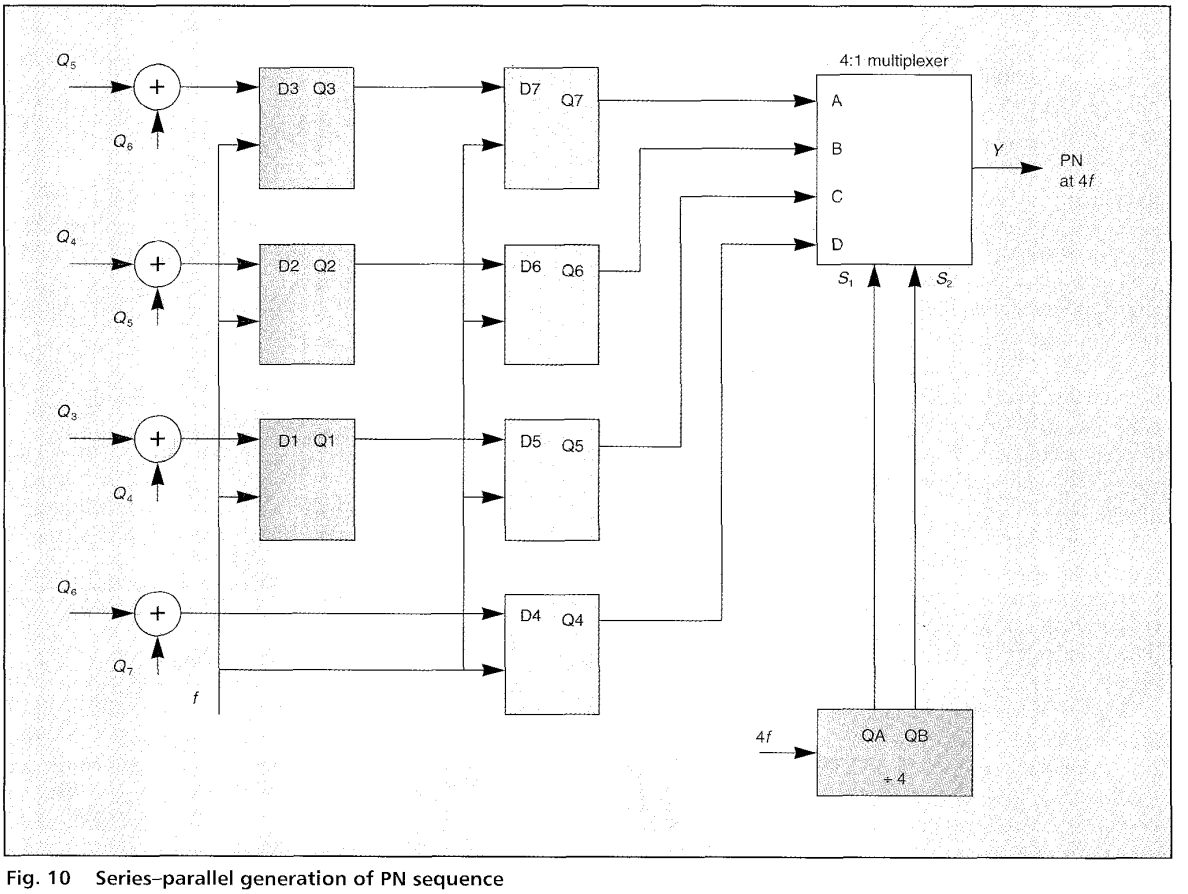


Fig. 10 Series–parallel generation of PN sequence

example. The threshold comparator triggers a pulse generator which generates an N-bit wide pulse. This pulse keeps the switch in position A and N fresh bits are loaded in the register. This method synchronises the receiver PN generator within a few attempts.

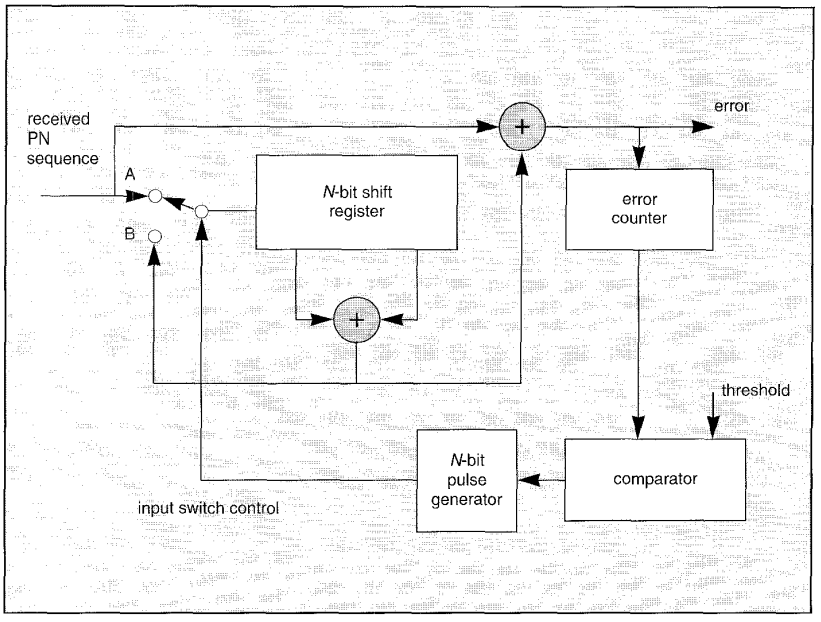


Fig. 11 PN sequence acquisition

5 Applications of PN sequences

PN sequences are used widely in digital communications, instrumentation etc. In each application some property of the PN is exploited.

The good autocorrelation property of PN sequences makes them a suitable candidate for frame synchronisation in digital communications.[6] A PN sequence can be acquired even in the presence of channel errors, minimising the detection misses. False detections can be minimised, without reducing the frame efficiency, by using a long sequence multiplexed with the data.

A PN sequence is an ideal test pattern, simulating all combinations of the data. Since an error-free reference PN sequence, synchronised to a received sequence, can be easily generated it can be compared with the received sequence to measure the channel errors.

Simulated, statistically random errors are useful in the evaluation of error-correcting and source coders and other digital communication systems. A simple technique to generate randomly distributed errors with a desired error rate using a PN sequence is given in Reference 7.

High-speed PN sequences are used in spread-spectrum modulation to spread the RF bandwidth of the signal, reducing the power spectral density. In the direct sequence method, a high-speed PN code is modulo-2 added to the low-rate data to increase the bit rate, and in the frequency-hopping method the carrier frequency is changed with the PN data using a fast switching frequency synthesiser. Correlation is used at the receiver to acquire the PN sequence and decode the data.

PN sequences are also used for scrambling the data, at the same rate, to obtain even spectral energy distribution within the signal band. Data scrambling also provides signal security.

Ranging is another application of PN sequences, which provide high accuracy and unambiguous measurement. In the sine and pulse methods of ranging, the unambiguous range is inversely proportional, and the accuracy directly proportional, to the sinewave frequency or the pulse repetition rate. However, in the PN method the unambiguous range is proportional to the sequence period, L/f, and the accuracy to the frequency, f. Hence, by choosing both the length, $L = 2^N - 1$, and the clock frequency, f, to be large both the range and accuracy requirements can be easily met.

Counters based on feedback shift register sequences have many advantages.[8] For a given logic family they can operate at higher speeds than conventional counters.

Acknowledgments

The author thanks Mr. Vilas Palsule, senior engineer of the Data Communications Division at the Space Applications Centre, for his useful comments and constructive suggestions in the preparation of this paper.

References

1 GOLOMB, S.W.: 'Shift register sequences' (Holden Day, San Francisco, CA, USA, 1967)
2 DIXON, R.C.: 'Spread spectrum systems' (John Wiley, 1984)
3 O'REILLY, J.J.: 'Series–parallel generation of m-sequences', *The Radio and Electronic Engineer*, April 1975, **45**, (4), pp.171–176
4 MUTAGI, R.N.: 'Pseudonoise generator doubles its speed', *EDN*, 22nd December 1994, pp.75
5 WARD, R.B.: 'Acquisition of pseudonoise signals by sequential estimation', *IEEE Trans. Comm. Technol.*, December 1965, **COM-13**, (4), pp.475–483
6 MUTAGI, R.N.: 'PN sequences provide robust synchronisation and BER monitoring in digital radio systems', *Wireless Systems Design*, USA, January 1996
7 MUTAGI, R.N.: 'Circuit generates statistically random pulses' *Electronic Engineering*, UK, April 1995, pp.20
8 KLEIN, B.: 'Use LFSR to build fast FPGA-based counters', *Electronic Design*, March 1994, **42**, (6), pp.87–100

First received 2nd May and in revised form 18th October 1995

The author is with the Baseband Processing Division, Space Applications Centre, SAC PO, Ahmedabad-380 053, India.