Nonlinear ternary feedback shift registers

D. H. Green and R. G. Kelsch

Digital Processes Research Laboratories, Department of Electrical Engineering and Electronics, University of Manchester Institute of Science and Technology, P.O. Box No. 88, Sackville Street, Manchester M60 1QD

A flexible theory of the general nonlinear ternary feedback shift register (fsr) is presented so that the inherent advantages of the ternary domain may be fully exploited in the fields of digital computers, communications, coding theory, and other areas where the device finds application. The authors show that the description afforded by the modulo-3 arithmetic functions may be adapted to provide a polynomial domain representation of these devices which is more flexible than other ternary operations. Methods of transforming the sequence domain behaviour of the device into this polynomial form, and vice-versa, are presented. Certain properties are isolated and the theory is extended by deriving the transforms required to produce certain related polynomial forms which correspond to simple operations in the sequence domain of the original fsr. The mechanism whereby two factor polynomials may be combined algebraically to produce a composite polynomial with exactly the same cycle set as a cascade connection of the two factors is fully investigated. Results concerning the related forms of these composite types are presented together with certain identities under the polynomial transforms.

(Received August 1972)

1. Introduction

Several authors (Turecki, 1968; Santos and Arango, 1964; Godfrey, 1966; Lee and Lee, 1972) have indicated the advantages in terms of increased speed and capacity together with decreased cost and complexity of employing three state or ternary switching devices in place of the conventional binary elements. These advantages are especially relevant to applications in digital computers and automatic control. Also, more efficient error protection schemes (Kelsch and Green, 1971) and self-synchronising dictionaries are available to this ternary régime. Systems which require sources of pseudo-random sequences (Green and Kelsch, 1972) greatly benefit from a translation to a three symbol representation.

In the foreseeable future the improvements in performance of conventional binary logic devices will become few and far between; an indication that circuit components are approaching their practical limit in size and speed. However, the demand on data handling and processing systems will inevitably maintain its explosive growth as more and more disciplines become oriented to the digital mode. To combat this designers must now begin seriously to consider the possibility of constructing systems to operate with multivalued logic devices.

The feedback shift register (fsr) is one of the most versatile components in binary applications and the device and its sequences find numerous applications in many diverse fields. Preliminary investigations (Kelsch, 1972; O'Carroll, 1972) have indicated that a similar, if not better, utility is evident in the ternary régime. To make full use of this device and its flexibility we first require to understand the fundamental theory describing its behaviour. This understanding will lead, hopefully, to a discovery of many interesting properties, design procedures, and applications. This paper is devoted to the study of the ternary non-linear feedback shift register and its autonomous sequences.

2. Ternary feedback shift registers

The general form of the ternary feedback shift register is shown in **Fig. 1**. It consists of a cascade of ternary memory elements (tristables) which hold past values of the output from the modulo-3 addition of the input digit (if any) and the feedback digit as derived from the feedback function. Although we are usually interested in the autonomous behaviour of the device it is important to derive a form of representation which is applicable to the forced mode also. The 'output' digit stream Z of the system is related to the input stream X and the feedback stream Z^* as follows,

$$Z = X \oplus Z^* \tag{1}$$

Now the function $f(x_1, x_2, \ldots, x_n)$ may be regarded as a recursion in an 'operator' x, so that x_i may be interpreted as the ith application of this recursion to Z, as well as the name of the ith register position. Clearly, we are using the x's in the same way as Huffman's (1956) delay operator D because x_i holds the ith delayed version of Z, namely Z_i, which is equivalent to $D^i.Z$. This means that we may represent the feedback digit stream as the output stream Z weighted by the feedback function. That is,

$$Z^* = f(Z_1, Z_2, \ldots, Z_n) = f(x_1, x_2, \ldots, x_n).Z \tag{2}$$

so that equation (1) becomes

$$Z = X \oplus f(x_1, x_2, \ldots, x_n).Z \tag{3}$$

or,

$$2X = 2Z \oplus f(x_1, x_2, \ldots, x_n).Z$$

so that,

$$\frac{Z}{X} = \frac{2}{2 \oplus f(x_1, x_2, \ldots, x_n)} = \frac{2}{F(x)} \tag{4}$$

So the system is characterised by a describing polynomial $F(x)$ which has the form,

$$F(x) = 2 \oplus f(x_1, x_2, \ldots, x_n) \tag{5}$$

where $f(x_1, x_2, \ldots, x_n)$ corresponds directly to the feedback

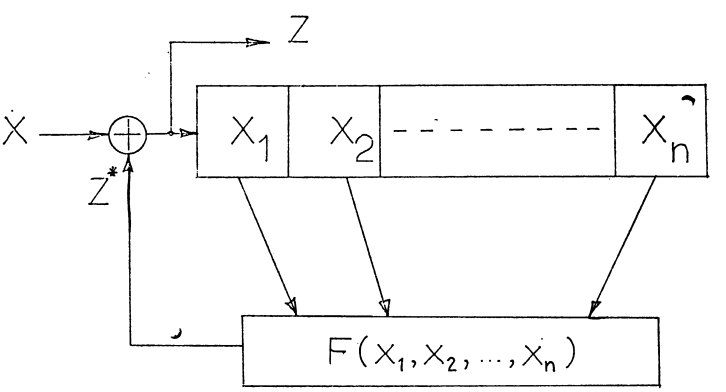

Fig. 1 The general ternary fsr

function which defines the autonomous behaviour of the system. Note that all polynomials of this type will involve the constant term 2, which does not play any part in the autonomous behaviour but is the weighting given to the input stream in the forced mode. To avoid confusion of this constant term with the modulo-3 constants **1** and **2** which may be present in $f(x_1, x_2, \ldots, x_n)$ these latter will also be represented as single or double primes respectively, covering the whole polynomial.

2.1. *Linear ternary fsrs*

When the feedback function is restricted to include only modulo-3 additions of the register contents the device may be considered to be linear. Since three is a prime number, the results due to Elspas (1959) concerning the composition of the characteristic polynomial of the device and its cycle set structure remains valid.

2.2. *Nonlinear ternary fsrs*

When the linear restriction on the composition of the feedback function is relaxed the availability of a characteristic polynomial disappears because the device becomes nonlinear. The describing polynomial, as we have seen, remains a valid form of description, and we may use it to investigate the properties of the devices in this régime.

In the main, these properties will be manifested in the sequence domain behaviour of the fsr so it is important that the relationship between the sequence and polynomial domains be thoroughly understood. The state S, of the shift register will be represented as the number, expressed either in ternary or decimal notation, corresponding to the stored digits and in which the nth digit makes the most significant contribution. A change of state corresponds to a shifting of this number one place, equivalent to multiplying by three, and the addition of a new least significant digit. Thus if P is the new state created in this way, we may write

$$P = 3.S + d_S \text{ (modulo-}3^n\text{)} \tag{6}$$

where $d_S = 0$, 1, or 2 and is the digit produced for input when the fsr is in state S. Obviously, there are three different values of S, differing only in the value of the most significant ternary digit, which can become P when followed by an appropriate d_S, so that each state may have any of three predecessors as well as any of three successors.

The deterministic nature of the state transition mechanism ensures that each state will have a unique successor state. However, each state need not have a unique predecessor and in fact it may have none or up to three distinct predecessors, each corresponding to the occasions when more than one of the states S, $S + 3^{n-1}$, and $S + 2.3^{n-1}$ give the same value of feedback digit and therefore proceed to state P, as indicated by equation (6). If $F(x)$ is the describing polynomial we may interpret $F(S)$ to be the value of the feedback digit when the fsr is in state S. Thus $d_S = F(S)$ and equation (6) becomes

$$P = 3.S + F(S) \text{ (modulo-}3^n\text{)} \tag{7}$$

2.3. *Polynomial and sequence domains*

It is evident that the particular form the state transition diagram of the device takes is completely determined by the describing polynomial. An alternative means of representing the sequence domain operation is afforded by the 'next-digit' map, in which the 3^n cells correspond to the 3^n available states, and are identified by suitable ternary coding of the rows and columns. The contents of each cell is the value of the function $F(S)$ for each state S and therefore indicates the next-digit following the state S. **Fig. 2** demonstrates the arrangement for the two variable case in which (*a*) represents this 'sequence domain' map and (*b*) is the piterm (Green and Dimond, 1970a) or 'polynomial domain' map.

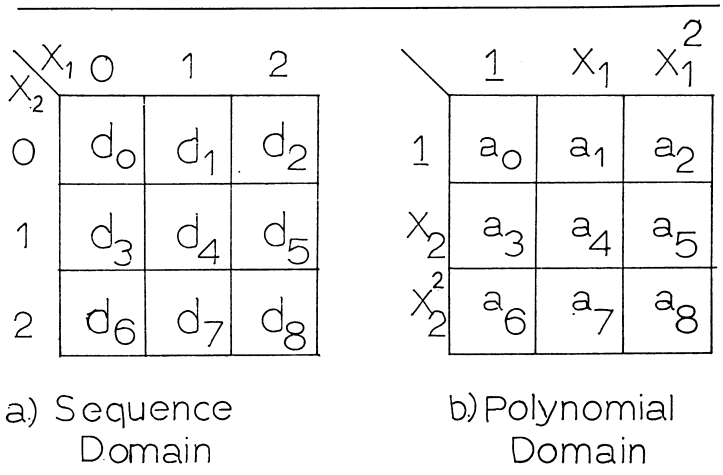

Fig. 2 Sequence and polynomial domain maps

An obvious requirement now is the ability to devise the contents of one map representation from a knowledge of the contents of the other. To determine the transformation between the a's and d's we employ the fact that the digit d_S is the 'value' of the function when the fsr is in state S. Consequently, 3^n simultaneous equations may be formed by substituting the values of variables $x_1, x_2, \ldots, x_n$ corresponding to each state S, in the general form of the function, and deriving the appropriate d_S in terms of the coefficients of the polynomial map. When $n = 2$, for example, we find

$$\begin{aligned}
d_0 &= a_0 \\
d_1 &= a_0 \oplus a_1 \oplus a_2 \\
d_2 &= a_0 \oplus 2a_1 \oplus a_2 \\
d_3 &= a_0 \oplus a_3 \oplus a_6 \\
d_4 &= a_0 \oplus a_1 \oplus a_2 \oplus a_3 \oplus a_4 \oplus a_5 \oplus a_6 \oplus a_7 \oplus a_8 \\
d_5 &= a_0 \oplus 2a_1 \oplus a_2 \oplus a_3 \oplus 2a_4 \oplus a_5 \oplus a_6 \oplus 2a_7 \oplus a_8 \\
d_6 &= a_0 \oplus 2a_3 \oplus a_6 \\
d_7 &= a_0 \oplus a_1 \oplus a_2 \oplus 2a_3 \oplus 2a_4 \oplus 2a_5 \oplus a_6 \oplus a_7 \oplus a_8 \\
d_8 &= a_0 \oplus 2a_1 \oplus a_2 \oplus 2a_3 \oplus a_4 \oplus 2a_5 \oplus a_6 \oplus 2a_7 \oplus a_8
\end{aligned} \tag{8}$$

The matrix equivalent of these equations is

$$\mathbf{D} = P_2.\mathbf{A} \tag{9}$$

where, **D** and **A** are column matrices containing the d's and a's and

$$P_2 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 2 & 1 & 1 & 2 & 1 & 1 & 2 & 1 \\
1 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 1 & 2 & 2 & 2 & 1 & 1 & 1 \\
1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1
\end{bmatrix} \tag{10}$$

Evidently, P_2 includes the lower order matrices P_1 and P_0 where

$$P_1 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \end{bmatrix} \text{ and } P_0 = [1]$$

Thus, we have in matrix terms

$$\mathbf{D} = P_n.\mathbf{A} \tag{11}$$

where the general recursive form of this transition matrix is apparent,

$$P_n = \left[\begin{array}{c|c|c}
P_{n-1} & 0 & 0 \\ \hline
P_{n-1} & P_{n-1} & P_{n-1} \\ \hline
P_{n-1} & 2P_{n-1} & P_{n-1}
\end{array}\right] \quad \text{for } n > 0 \text{ and } P_0 = [1] \tag{12}$$

We may now transform any polynomial map into the corresponding sequence domain map, and hence derive the sequences, by a single application of this matrix P_n.

Whilst this is useful, the more desirable transformation is the one which permits the derivation of the polynomial form corresponding to a given sequence domain structure, as represented on the next-digit map. Clearly, this process is represented by a matrix equation of the form,

$$\mathbf{A} = S_n.\mathbf{D} \tag{13}$$

and comparing this with the general form of equation (9) it follows that,

$$S_n = P_n^{-1} \tag{14}$$

Once more we may derive the general form of S_n,

$$S_n = \left[\begin{array}{c|c|c} S_{n-1} & 0 & 0 \\ \hline 0 & 2S_{n-1} & S_{n-1} \\ \hline 2S_{n-1} & 2S_{n-1} & 2S_{n-1} \end{array}\right] \quad \text{for } n > 0 \quad \text{and } S_0 = [1] \tag{15}$$

2.4. *Cyclic fsrs*

An important criterion to isolate in the general nonlinear régime is that which determines the property that all the states of a given fsr lie on pure branchless cycles. We have seen that in general all states of a particular fsr have a unique successor state but not necessarily a unique predecessor state. A necessary and sufficient condition for the generation of pure state cycles is that every state must have a unique predecessor state. In the sequence domain this restriction requires that the three states

$$\begin{aligned} S &\equiv b_1, b_2, b_3, \ldots, b_n \\ S + 3^{n-1} &\equiv b_1, b_2, b_3, \ldots, b_n \oplus \mathbf{1} \\ S + 2.3^{n-1} &\equiv b_1, b_2, b_3, \ldots, b_n \oplus \mathbf{2} \end{aligned}$$

shall give rise to a different feedback digit so their successor states will be distinct. This restriction clearly partitions the sequence domain map into three regions each one corresponding to the value assigned to the nth digit and covering 3^{n-1} cells. We may use this fact to enumerate these 'cyclic' functions. One region may be entered arbitrarily in the full ternary range; that is, in $3^{3^{n-1}}$ ways. The second region of 3^{n-1} cells is limited to take on any selection from the two remaining values not used in the first region; a process which may be performed in $2^{3^{n-1}}$ ways. The third region is forced to take the remaining ternary value in each case and can therefore be filled-in in only one way. Thus the total number of ways $C_{3,n}$ of filling in the nth order sequence domain map to ensure cyclic behaviour is given by,

$$C_{3,n} = 2^{3^{n-1}}.3^{3^{n-1}} = 6^{3^{n-1}} \tag{16}$$

Incidentally, this argument may be generalised to give the number of p-nary cyclic fsr's of degree n.

$$C_{p,n} = (p!)^{p^{n-1}} \tag{17}$$

We now require to derive the corresponding restrictions on the polynomial form of the cyclic fsr and introduce a theorem concerning these aspects.

Theorem I

A ternary autonomous fsr will generate pure state cycles if, and only if, its describing polynomial $F(x)$ is of the form,

$$F(x) = 2 \oplus f(x_1, x_2, \ldots, x_n)$$

and

$$f(x_1, x_2, \ldots, x_n) = f_1(x_1, x_2, \ldots, x_{n-1}) \oplus [\mathbf{1} \oplus f_2(x_1, x_2, \ldots, x_{n-1})] \odot x_n \tag{18}$$

where f_1 is any ternary function of the first $n-1$ variables and f_2 is any function which takes on only the values 0 or 1; i.e. it is a function equal to its own square (Kelsch, 1972).

An obvious consequence of confining the fsr to produce only pure state cycles is that a reverse-sequence fsr will exist. This is a new fsr whose sequences are the sequences of the original fsr in time reverse. This sequence domain relationship corresponds to the polynomial domain property defined by Theorem II.

Theorem II

If the describing polynomial $F(x)$ of a cyclic ternary autonomous fsr is of the form

$$F(x) = 2 \oplus f_1(x_1, \ldots, x_{n-1}) \oplus (\mathbf{1} \oplus f_2(x_1, \ldots, x_{n-1})) \odot x_n$$

then the polynomial $\mathbf{R}F(x)$ describes a new fsr the sequences of which are the time reverses of those of $F(x)$, and

$$\mathbf{R}F(x) = 2 \oplus 2 \odot f_1(x_{n-1}, \ldots, x_1) \odot (\mathbf{1} \oplus f_2(x_{n-1}, \ldots, x_1)) \oplus (\mathbf{1} \oplus f_2(x_{n-1}, \ldots, x_1)) \odot x_n \tag{19}$$

3. Related polynomial forms

We now consider the polynomial domain operations which enable certain related forms of ternary fsr to be described by a transform operation on the original polynomial. We shall illustrate these operations by means of an order-2 polynomial but the results will be completely general.

3.1. *Trinal form*

If, in the sequences of a ternary fsr with describing polynomial $F(x)$, we replace each digit d_i by $d_i \oplus \mathbf{1}$, then we have defined the trinal sequences of $F(x)$. The describing polynomial of the new fsr, which generates these sequences naturally, is termed the trinal polynomial form of $F(x)$ and is written $\mathbf{T}F(x)$.

Now, the above sequence domain operations indicate, for example, that if in the original fsr, the state 0 (i.e. $x_1 = 0$, $x_2 = 0$) is followed by digit d_0 (i.e. next state is $x_1 = d_0$, $x_2 = 0$), then in the trinal fsr state 4 (i.e. $x_1 = 1$, $x_2 = 1$) is followed by digit $d_0 \oplus \mathbf{1}$. Similarly, if state 5 is followed by digit d_5 in the original, then state 6 is followed by digit $d_5 \oplus \mathbf{1}$ in the trinal. In general, therefore, the trinal of the state is followed by the trinal of the next-digit. We may regard the complete sequence domain operation as a transformation of one set of successor digits d_i from the sequence domain map, into a new set $\tilde{d}_i$, representing the trinal.

Thus, for $n = 2$, we find

$$\begin{aligned} \tilde{d}_0 &\rightarrow d_8 \oplus \mathbf{1} \\ \tilde{d}_1 &\rightarrow d_6 \oplus \mathbf{1} \\ \tilde{d}_2 &\rightarrow d_7 \oplus \mathbf{1} \\ \tilde{d}_3 &\rightarrow d_2 \oplus \mathbf{1} \\ \tilde{d}_4 &\rightarrow d_0 \oplus \mathbf{1} \\ \tilde{d}_5 &\rightarrow d_1 \oplus \mathbf{1} \\ \tilde{d}_6 &\rightarrow d_5 \oplus \mathbf{1} \\ \tilde{d}_7 &\rightarrow d_3 \oplus \mathbf{1} \\ \tilde{d}_8 &\rightarrow d_4 \oplus \mathbf{1} \end{aligned} \tag{20}$$

or, in general matrix form

$$\tilde{D} \rightarrow U_n.D \oplus J_n \tag{21}$$

where J_n is a unit column vector (i.e. every element = **1**), $\tilde{D}$ and D are column vectors containing the $\tilde{d}_i$ and d_i respectively. The transformation matrix has the general structure

$$U_n = \left[\begin{array}{c|c|c} 0 & 0 & U_{n-1} \\ \hline U_{n-1} & 0 & 0 \\ \hline 0 & U_{n-1} & 0 \end{array}\right] \quad \text{for } n > 0 \quad \text{and } U_0 = [1] \tag{22}$$

Now, corresponding to this new sequence domain map $\tilde{D}$, there is a new polynomial domain map $\tilde{A}$, with coefficients $\tilde{a}_i$. Using the transform described in section 1, we may describe the new polynomial in terms of the original sequence map.

$$\tilde{A} = S_n.\tilde{D} \tag{23}$$

in general,

$$\tilde{A} = S_n(U_n.D \oplus J_n) \tag{24}$$

Furthermore, corresponding to the original sequence domain map D, there is a polynomial map A with coefficients a_i, which may be extracted by the relation

$$D = P_n.A \tag{25}$$

In general we have

$$\begin{aligned}\tilde{A} &= S_n.U_n.P_n.A \oplus J_n^* \\ &= T_n.A \oplus J_n^*\end{aligned} \tag{26}$$

where J_n^* is a column vector in which the first element $= \mathbf{1}$ and all the others are zero, and
where, $T_n = S_nU_nP_n$ and has the form

$$T_n = \left[\begin{array}{c|c|c} T_{n-1} & 2T_{n-1} & T_{n-1} \\ \hline 0 & T_{n-1} & T_{n-1} \\ \hline 0 & 0 & T_{n-1} \end{array}\right] \quad \begin{array}{l}\text{for } n > 0 \\ \text{and } T_0 = [1]\end{array} \tag{27}$$

Thus the trinal form may be evolved by a single application of this transform matrix to the original function. This transformation may be shown to be equivalent to certain algebraic manipulations of the polynomial (Kelsch, 1972), for if

$$F(x) = 2 \oplus f(x_1, x_2, \ldots, x_n) \tag{28}$$

then

$$\begin{aligned}\mathbf{T}F(x) &= 2 \oplus f(x_1 \oplus \mathbf{2}, x_2 \oplus \mathbf{2}, \ldots, x_n \oplus \mathbf{2})' \\ &= [2 \oplus f(x_1 \oplus \mathbf{2}, x_2 \oplus \mathbf{2}, \ldots, x_n \oplus \mathbf{2})]'\end{aligned} \tag{29}$$

For example, if

$$F(x) = 2 \oplus 2x_1 \oplus 2x_1^2 \oplus x_1x_2 \oplus x_1^2x_2 \oplus 2x_2$$

its sequences are (*a*) 2 2 1 0 2 0 1 1 and (*b*) 0 i.e. a cyclic fsr with cycle set (8, 1).
Now,

$$\begin{aligned}\mathbf{T}F(x) &= 2 \oplus 2(x_1 \oplus \mathbf{2}) \oplus 2(x_1 \oplus \mathbf{2})^2 \oplus (x_1 \oplus \mathbf{2})(x_2 \oplus \mathbf{2}) \\ &\quad \oplus (x_1 \oplus \mathbf{2})^2(x_2 \oplus \mathbf{2}) \oplus 2(x_2 \oplus \mathbf{2}) \oplus \mathbf{1} \\ &= [2 \oplus 2x_1 \oplus x_1^2 \oplus 2x_1x_2 \oplus x_1^2x_2 \oplus 2x_2]''\end{aligned}$$

and the trinal sequences are (*a*) 0 0 2 1 0 1 2 2 and (*b*) 1; again a cycle set of (8, 1).

3.2. *Bitrinal form*

If, in the sequences of a ternary fsr with describing polynomial $F(x)$, we replace each digit d_i by $d_i \oplus \mathbf{2}$, then we have defined the bitrinal sequences of $F(x)$. The describing polynomial of the new fsr, which generates these sequences naturally, is termed the bitrinal form of $F(x)$ and is written $\mathbf{B}F(x)$. Clearly, $\mathbf{B}F(x) = \mathbf{T}(\mathbf{T}F(x)) = \mathbf{T}^2F(x)$.

By repeating the arguments of the previous section we may derive the sequence domain transformation involved in forming the bitrinal sequences.

$$\tilde{D} = V_n.D \oplus K_n \tag{30}$$

where $K_n = 2J_n$, i.e. a column vector in which each element $= \mathbf{2}$, and where V_n has the general structure

$$V_n = \left[\begin{array}{c|c|c} 0 & V_{n-1} & 0 \\ \hline 0 & 0 & V_{n-1} \\ \hline V_{n-1} & 0 & 0 \end{array}\right] \quad \begin{array}{l}\text{for } n > 0 \\ \text{and } V_0 = [1]\end{array} \tag{31}$$

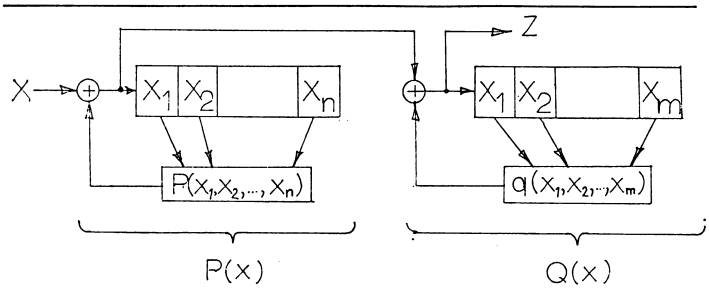

Fig. 3 Cascade connection of two ternary fsrs

Again, we may derive the equivalent polynomial domain transformation, because

$$\begin{aligned}\tilde{A} &= S_n.V_n.P_n.A \oplus S_n.K_n \\ &= B_n.A \oplus K_n^*\end{aligned} \tag{32, 33}$$

where $K_n^* = 2J_n^*$, i.e. a column vector with first element $= \mathbf{2}$ and all others zero. The bitrinal transform B_n is found to have the general structure

$$B_n = \left[\begin{array}{c|c|c} B_{n-1} & B_{n-1} & B_{n-1} \\ \hline 0 & B_{n-1} & 2B_{n-1} \\ \hline 0 & 0 & B_{n-1} \end{array}\right] \quad \begin{array}{l}\text{for } n > 0 \\ \text{and } B_0 = [1]\end{array} \tag{34}$$

Also, if

$$F(x) = 2 \oplus f(x_1, x_2, \ldots, x_n)$$

then

$$\mathbf{T}F(x) = 2 \oplus f(x_1 \oplus \mathbf{2}, x_2 \oplus \mathbf{2}, \ldots, x_n \oplus \mathbf{2})'$$

and therefore

$$\mathbf{B}F(x) = \mathbf{T}(\mathbf{T}F(x)) = 2 \oplus f(x_1 \oplus \mathbf{1}, x_2 \oplus \mathbf{1}, \ldots, x_n \oplus \mathbf{1})'' \tag{35}$$

Taking our previous example we find

$$\begin{aligned}\mathbf{B}F(x) &= 2 \oplus 2(x_1 \oplus \mathbf{1}) \oplus 2(x_1 \oplus \mathbf{1})^2 \oplus (x_1 \oplus \mathbf{1})(x_2 \oplus \mathbf{1}) \\ &\quad \oplus (x_1 \oplus \mathbf{1})^2(x_2 \oplus \mathbf{1}) \oplus 2(x_2 \oplus \mathbf{1}) \oplus \mathbf{2} \\ &= [2 \oplus x_1^2x_2 \oplus x_2]'\end{aligned} \tag{36}$$

which generates the bitrinal sequences (*a*) 1 1 0 2 1 2 0 0 and (*b*) 2.

3.3. *Twin form*

If, in the sequences of a ternary fsr with describing polynomial $F(x)$, we replace each digit d_i by $2 \odot d_i$ then we have defined the twin sequences of $F(x)$. The describing polynomial of the new fsr which generates these sequences naturally, is termed the twin polynomial form of $F(x)$ and is written $\mathbf{Tw}F(x)$.

By repeating the arguments of the previous sections we may set up the matrix form of this sequence domain transformation

$$\tilde{D} = W_n.D \tag{37}$$

where,

$$W_n = \left[\begin{array}{c|c|c} W_{n-1} & 0 & 0 \\ \hline 0 & 0 & W_{n-1} \\ \hline 0 & W_{n-1} & 0 \end{array}\right] \quad \begin{array}{l}\text{for } n > 0 \\ \text{and } W_0 = [2]\end{array} \tag{38}$$

This leads to an equivalent polynomial transformation of the form

$$\begin{aligned}\tilde{A} &= S_n.W_n.P_n.A \\ &= Tw_n.A\end{aligned} \tag{39}$$

where

$$Tw_n = \left[\begin{array}{c|c|c} Tw_{n-1} & 0 & 0 \\ \hline 0 & 2Tw_{n-1} & 0 \\ \hline 0 & 0 & Tw_{n-1} \end{array}\right] \quad \begin{array}{l}\text{for } n > 0 \\ \text{and } Tw_0 = [2]\end{array} \tag{40}$$

Equivalent manipulations to those of previous sections may also be performed to reveal that this transform is identical to that obtained by replacing

$$F(x) = 2 \oplus f(x_1, x_2, \ldots, x_n)$$

with

$$F^*(x) = 2 \oplus 2f(2x_1, 2x_2, \ldots, 2x_n) \tag{41}$$

so that

$$\mathbf{Tw}F(x) = 2 \oplus 2f(2x_1, 2x_2, \ldots, 2x_n) \tag{42}$$

Using our example we find that

$$\mathbf{Tw}F(x) = 2 \oplus 2(2.2x_1 \oplus 2(2x_1)^2 \oplus 2x_1.2x_2 \oplus (2x_1)^2.2x_2 \oplus 2.2x_2) \quad (43)$$
$$= 2 \oplus 2x_1 \oplus x_1^2 \oplus 2x_1x_2 \oplus x_1^2x_2 \oplus 2x_2$$

and the twin sequences are (*a*) 1 1 2 0 1 0 2 2 and (*b*) 0.

3.4. *Other related forms*

It is obvious that combinations of the three polynomial operators could be used to define up to six related forms. In addition, the reverse operator **R** can be employed to extend this to twelve related forms, if only cyclic describing polynomials are considered. Although more combinations of the operators are apparent we shall see later that certain ones are equivalent to some member of this basic set and give rise to the same related form. However, each transform associated with the distinct related forms may obviously be derived from a combination of the appropriate single transforms. For example, the twin trinal form $\mathbf{TwT}F(x)$ is formed by taking the twin of the trinal.

Thus

$$F(x) = 2 \oplus f(x_1, x_2, \ldots, x_n)$$

and

$$\mathbf{T}F(x) = 2 \oplus f(x_1 \oplus \mathbf{2}, x_2 \oplus \mathbf{2}, \ldots, x_n \oplus \mathbf{2})'$$

so

$$\mathbf{TwT}F(x) = 2 \oplus 2f(2x_1 \oplus \mathbf{2}, 2x_2 \oplus \mathbf{2}, \ldots, 2x_n \oplus \mathbf{2})'' \quad (44)$$

The matrix equation representing this complete transform is

$$\tilde{A} = TwT_n.A \oplus L_n^* \text{ where } L_n^* = Tw_n.J_n^* = K_n^*$$

and the corresponding transformation matrix TwT_n is formed as follows

$$TwT_n = Tw_n.T_n = \left[\begin{array}{c|c|c} Tw_{n-1} & 0 & 0 \\ \hline 0 & 2Tw_{n-1} & 0 \\ \hline 0 & 0 & Tw_{n-1} \end{array}\right] \cdot \left[\begin{array}{c|c|c} T_{n-1} & 2T_{n-1} & T_{n-1} \\ \hline 0 & T_{n-1} & T_{n-1} \\ \hline 0 & 0 & T_{n-1} \end{array}\right]$$

$$= \left[\begin{array}{c|c|c} TwT_{n-1} & 2TwT_{n-1} & TwT_{n-1} \\ \hline 0 & 2TwT_{n-1} & 2TwT_{n-1} \\ \hline 0 & 0 & TwT_{n-1} \end{array}\right]$$

$$\text{for } n > 0 \text{ and } TwT_0 = [2] \quad (45)$$

4. Cascaded fsrs and composite polynomial forms

We have seen that the 'transfer function' approach to the analysis of the ternary fsr relates the output sequences Z to the input sequences X in the following way.

$$H = \frac{Z}{X} = \frac{2}{F(x)} \quad (46)$$

where $F(x)$ is called the describing polynomial of the fsr and has the structure

$$F(x) = 2 \oplus f(x_1, x_2, \ldots, x_n) \quad (47)$$

wherein $f(x_1, x_2, \ldots, x_n)$ is the feedback function which describes the physical connections to the n-stage ternary shift register in terms of modulo-3 sums and products. For convenience and brevity, equations of the type of (47) will, from time to time, be written as

$$F = 2 \oplus f \quad (48)$$

We now turn our interest to the consideration of cascaded interconnections of two ternary fsrs. This arrangement involves the forced response of one fsr to either the autonomous or the forced sequences of the other, as depicted in **Fig. 3.** Evidently, this cascade arrangement has an 'overall' transfer function which is related in some way to the transfer functions of the individual fsrs. That is, we may define a 'product' in the polynomial domain which evolves a higher order composite polynomial which parallels the sequence domain operation of cascading (Green and Dimond, 1970b). This new polynomial describes a higher order single fsr with the structure shown in **Fig. 4** whose output sequences are identical to those from the cascade. This then extends the concept of the composite polynomial to this general nonlinear régime.

Let the sequence domain operation of cascading be reflected in the polynomial domain by the operation

$$H(x) \equiv H_1(x) \rightarrow H_2(x) \quad (49)$$

where

$$H_1(x) = 2/P(x) = 2/(2 \oplus p),\ H_2(x) = 2/Q(x) = 2/(2 \oplus q),$$

and

$$H(x) = 2/F(x) = 2/(2 \oplus f)$$

is the overall transfer function of the cascade. We now wish to relate the structure of the composite describing polynomial $F(x)$ to that of the factor polynomials $P(x)$ and $Q(x)$. If we consider the arrangement of Fig. 3 we may establish that the present value of the digit b_i depends on the previous values of the digits b_i and the present value of the digit a_i from the first fsr whereas the present value of the digit a_i depends on its own previous values and the input digit (if any).

We may write for the autonomous mode

$$\left.\begin{array}{r} a_i = p(a_{i-1}, \ldots, a_{i-n}) \\ a_{i-1} = p(a_{i-2}, \ldots, a_{i-n-1}) \\ \text{etc.} \end{array}\right\} \quad (50)$$

and

$$\left.\begin{array}{r} b_i = a_i \oplus q(b_{i-1}, \ldots, b_{i-m}) \\ b_{i-1} = a_{i-1} \oplus q(b_{i-2}, \ldots, b_{i-m-1}) \\ \text{etc.} \end{array}\right\} \quad (51)$$

From equations (50) and (51) we find

$$b_i \oplus 2q(b_{i-1}, \ldots, b_{i-m}) = p(a_{i-1}, \ldots, a_{i-n}) \quad (52)$$

Therefore,

$$b_i = q(b_{i-1}, \ldots, b_{i-m}) \oplus p(a_{i-1}, \ldots, a_{i-n}) \quad (53)$$

and we may substitute for each a_k in equation (53) using equation (51).

$$b_i = q(b_{i-1}, \ldots, b_{i-m}) \oplus p[b_{i-1} \oplus 2q(b_{i-2}, \ldots, b_{i-m-1}), \ldots, b_{i-n} \oplus 2q(b_{i-n-1}, \ldots, b_{i-n-m})] \quad (54)$$

To transfer this recursion into the general polynomial notation we replace b_{i-j} by x_j and derive the feedback function f. So that

$$f = q(x_1, \ldots, x_m) \oplus p[x_1 \oplus 2q(x_2, \ldots, x_{m+1}), \ldots, x_n \oplus 2q(x_{n+1}, \ldots, x_{n+m})] \quad (55)$$

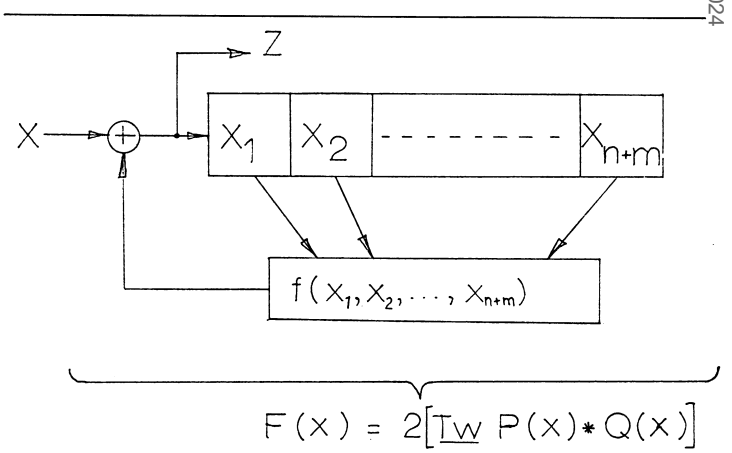

Fig. 4 Equivalent single fsr

and

$$F = 2 \oplus f$$

It is possible, moreover, to describe a multiplication procedure to operate between $P(x)$ and $Q(x)$ which derives $F(x)$ directly. Let us define

$$F = 2[(2 \oplus p) \rightarrow (2 \oplus q)] \quad (56)$$

where $\rightarrow$ signifies some polynomial domain operation equivalent to cascading in the sequence domain and we assume for the purpose of constants that this operation is identical to modulo-3 multiplication, so that we may write

$$\begin{aligned} F &= 2[1 \oplus 2q \oplus p \rightarrow (2 \oplus q)] \\ &= 2 \oplus q \oplus 2p \rightarrow (2 \oplus q) \end{aligned} \quad (57)$$

Note that the overall multiplication by 2 is required to restore F to the same standard polynomial form (i.e. with a constant 2) as P and Q. From (57) we observe that

$$f = q \oplus 2p \rightarrow (2 \oplus q) \quad (58)$$

A comparison of equations (55) and (58) confirms the previous assumption and reveals that

$$2p \rightarrow (2 \oplus q) = p[x_1 \oplus 2q(x_2, \ldots, x_{m+1}), \ldots, x_n \oplus 2q(x_{n+1}, \ldots, x_{n+m})] \quad (59)$$

The right-hand side of equation (59) is formed by replacing each variable x_i, in the function p, with $x_i \oplus 2q(x_{i+1}, \ldots, x_{i+m})$. At this point let us define a new polynomial domain multiplication operation denoted by *, which behaves as follows.

(*a*) for a constant term 2 $\quad 2^*Q(x) = 2 \odot Q(x) = 2Q(x)$

(*b*) for a linear term x_i

$$\begin{aligned} x_i^*Q(x) &= x_i^*(2 \oplus q(x_1, \ldots, x_m)) \\ &= 2x_i \oplus q(x_{i+1}, \ldots, x_{i+m}) \end{aligned} \quad (60)$$

(*c*) for a nonlinear term $x_i \odot x_j \odot \ldots, \odot_k$

$$(x_i \odot x_j \odot \ldots, \odot x_k)^*Q(x) = [x_i^*Q(x)] \odot [x_j^*Q(x)] \odot \ldots, \odot [x_k^*Q(x)] \quad (61)$$

Each subproduct in equation (61) is resolved using equation (60) and the total function is expanded using the rules of modulo-3 multiplication. Note that whereas modulo-3 multiplication increases the 'span' (i.e. number of variables) of a term, this new multiplication increases the 'order' of the term, e.g.

$$(x_i) \odot (x_j) = x_i \odot x_j \text{ or } x_ix_j \text{ whereas } (x_i)^*(x_j) = x_{i+j}$$

We can now rewrite equation (59) as follows

$$2p \rightarrow (2 \oplus q) = p[x_1^*(1 \oplus 2q(x_1, \ldots, x_m)), \ldots, x_n^*(1 \oplus 2q(x_1, \ldots, x_m))] \quad (62)$$

and so each variable x_i has been replaced by

$$2x_i^*(2 \oplus q(x_1, \ldots, x_m))$$

which equals $2x_i^*Q(x)$.

Therefore,

$$\begin{aligned} 2p \rightarrow (2 \oplus q) &= p[2x_1^*Q(x), \ldots, 2x_n^*Q(x)] \quad (63) \\ &= 2\mathbf{Tw}\, p[x_1^*Q(x), \ldots, x_n^*Q(x)] \quad (64) \end{aligned}$$

On the understanding that * involves a term-by-term operation between each term from p and $Q(x)$ we may write (64) in a more convenient form

$$2p \rightarrow (2 \oplus q) = 2[\mathbf{Tw}\, p^*Q] \quad (65)$$

So that

$$F = Q \oplus 2[\mathbf{Tw}p^*Q] \quad (66)$$

So that

$$\begin{aligned} &= 2[2 \oplus \mathbf{Tw}p^*Q] \quad (67) \\ &= 2[\mathbf{Tw}P^*Q] \quad (68) \end{aligned}$$

We now have the result

$$F(x) = P(x) \rightarrow Q(x) = 2[\mathbf{Tw}P(x)^*Q(x)] \quad (69)$$

So to form the product polynomial we first derive the twin polynomial of $P(x)$ and then multiply $Q(x)$ by each term from $\mathbf{Tw}P(x)$ using the order increasing multiplication defined previously. Finally, the coefficient of each term in the resulting polynomial is multiplied by 2. Note that if $P(x)$ is a linear polynomial then $\mathbf{Tw}P(x) = P(x)$ and so the multiplication can proceed directly. As an example of this procedure consider $P(x) = 2 \oplus 2x_1x_2 \oplus x_3$ and $Q(x) = 2 \oplus x_2^2 \oplus x_3$. We wish to find $F(x) \equiv P(x) \rightarrow Q(x)$.

Now

$$\begin{aligned} F(x) &= 2[\mathbf{Tw}P(x)^*Q(x)] \\ &= 2[\mathbf{Tw}(2 \oplus 2x_1x_2 \oplus x_3)^*Q(x)] \\ &= 2[(2 \oplus x_1x_2 \oplus x_3)^*Q(x)] \\ &= 2[2^*Q(x) \oplus x_1^*Q(x) \odot x_2^*Q(x) \oplus x_3^*Q(x)] \\ &= 2[(1 \oplus 2x_2^2 \oplus 2x_3) \oplus (2x_1 \oplus x_3^2 \oplus x_4) \\ &\quad \odot (2x_2 \oplus x_4^2 \oplus x_5) \oplus (2x_3 \oplus x_5^2 \oplus x_6)] \\ &= [2 \oplus 2x_1x_2 \oplus x_2^2 \oplus 2x_3 \oplus x_2x_3^2 \oplus 2x_4 \oplus x_2x_4 \\ &\quad \oplus x_1x_4^2 \oplus 2x_3^2x_4^2 \oplus x_1x_5 \oplus 2x_4x_5 \oplus 2x_3^2x_5\, 2 \oplus x_3^2 \\ &\quad \oplus 2x_6] \end{aligned}$$

5. Product formation with polynomials containing 1 or 2

The procedures described in the previous section assume that neither of the factor polynomials contain a ternary constant additive in the feedback function. We now wish to consider the formation of products involving polynomials of the form $[P(x)]'$ or $[P(x)]''$ where the primes represent the presence of **1** or **2** in the feedback function contained in $P(x)$. Thus

$$[P(x)]' = 2 \oplus p(x_1, \ldots, x_n) \oplus \mathbf{1} \quad (70)$$

and

$$[P(x)]'' = 2 \oplus p(x_1, \ldots, x_n) \oplus \mathbf{2}$$

There are three distinct situations in which a constant may arise.

1. *Only first factor has a constant*

In this case we are dealing with products of $[P(x)]'$ or $[P(x)]''$ with the second factor $Q(x)$ which does not contain a constant of this type. Consider the case of $[P(x)]'$. Equations (50) now become

$$\left.\begin{aligned} a_i &= p(a_{i-1}, \ldots, a_{i-n}) \oplus \mathbf{1} \\ a_{i-1} &= p(a_{i-2}, \ldots, a_{i-n-1}) \oplus \mathbf{1} \\ &\text{etc} \end{aligned}\right\} \quad (71)$$

Equations (51) are unchanged but (54) becomes

$$b_i = q(b_{i-1}, \ldots, b_{i-m}) \oplus p[b_{i-1} \oplus 2q(b_{i-2}, \ldots, b_{i-m-1}), \ldots, b_{i-n} \oplus 2q(b_{i-n-1}, \ldots, b_{i-n-m})] \oplus \mathbf{1} \quad (72)$$

The remainder of the procedure is as before, the only change being the inclusion of the constant term which now 'covers' the product function f.

So

$$\begin{aligned} F(x) \equiv [P(x)]' \rightarrow Q(x) &\equiv 2\{\mathbf{Tw}[P(x)]'^*Q(x)\} = \\ &2\{[\mathbf{Tw}P(x)]''^*Q(x)\} \\ &= 2\{[\mathbf{Tw}P(x)^*Q(x)]''\} \\ &= [2\mathbf{Tw}P(x)^*Q(x)]' \end{aligned} \quad (73)$$

By a similar argument we find

$$[P(x)]'' \rightarrow [Q(x)] = [2\mathbf{Tw}P(x)^*Q(x)]'' \quad (74)$$

2. *Only second factor has a constant*

In this case we are dealing with the products of $P(x)$ which does not have a constant, with $[Q(x)]'$ or $[Q(x)]''$. Consider the case for $[Q(x)]'$. Equations (50) remain unchanged whereas (51) become

$$\left.\begin{aligned} b_i &= a_i \oplus q(b_{i-1}, \ldots, b_{i-m}) \oplus \mathbf{1} \\ b_{i-1} &= a_{i-1} \oplus q(b_{i-2}, \ldots, b_{i-m-1}) \oplus \mathbf{1} \\ &\text{etc.} \end{aligned}\right\} \quad (75)$$

Thus

$$b_i \oplus 2q(b_{i-1}, \ldots, b_{i-m}) \oplus \mathbf{2} = p(a_{i-1}, \ldots, a_{i-n}) \quad (76)$$

and

$$b_i = q(b_{i-1}, \ldots, b_{i-m}) \oplus p[(b_{i-1} \oplus 2q(b_{i-2}, \ldots, b_{i-m-1}) \oplus \mathbf{2}, \ldots, b_{i-n} \oplus 2q(b_{i-n-1}, \ldots, b_{i-n-m}) \oplus \mathbf{2}] \oplus \mathbf{1} \quad (77)$$

Examination of equation (77) reveals that we must still take the twin of the first factor but, because each variable now has the constant **2** associated with it, we must also take the bitrinal of $\mathbf{Tw}P(x)$. Furthermore, this second operation accounts for the constant **1** covering the whole function. Hence

$$[P(x)] \rightarrow [Q(x)]' = 2\{\mathbf{Tw}P(x)^*[Q(x)]'\} = 2[\mathbf{BTw}P(x)^*Q(x)] \quad (78)$$

In the event that $\mathbf{BTw}P(x)$ develops a constant this must be brought out and transferred to cover the entire product according to the considerations under case 1. Also, by a similar argument, one can show that

$$[P(x)] \rightarrow [Q(x)]'' = 2[\mathbf{TTw}P(x)^*Q(x)] \quad (79)$$

3. *Both factors contain a constant*

This situation may be resolved using a combination of the previous results. First, the constant in the second factor is removed by the appropriate transform on the twin of the first factor. Second, any constant remaining in the transformed version of the first factor is then transferred to cover the whole product. As an example consider $P(x) = (2 \oplus x_1x_2 \oplus x_3)'$ and $Q(x) = (2 \oplus x_1^2 \oplus x_3)''$. We wish to find $F(x) \equiv P(x) \rightarrow Q(x)$. Now

$$\begin{aligned} F(x) &= 2[\mathbf{Tw}(2 \oplus x_1x_2 \oplus x_3)'^*(2 \oplus x_1^2 \oplus x_3)''] \\ &= 2[(2 \oplus 2x_1x_2 \oplus x_3)''^*(2 \oplus x_1^2 \oplus x_3)''] \\ &= 2[\mathbf{T}(2 \oplus 2x_1x_2 \oplus x_3)''^*(2 \oplus x_1^2 \oplus x_3)] \\ &= 2[(2 \oplus x_1 \oplus x_2 \oplus 2x_1x_2 \oplus x_3)'^*(2 \oplus x_1^2 \oplus x_3)] \\ &= \{2[(2 \oplus x_1 \oplus x_2 \oplus 2x_1x_2 \oplus x_3)^*(2 \oplus x_1^2 \oplus x_3)]\}'' \end{aligned}$$

and the multiplication can proceed as before within the square brackets.

6. Some results concerning related forms of composite polynomials

An investigation of the consequences of performing the polynomial transforms on composite polynomial types reveals the following results which are presented here without proofs.

6.1. *Trinal form*

Let $F(x) = P(x) \rightarrow Q(x) = 2[\mathbf{Tw}P(x)^*Q(x)]$ be a composite ternary describing polynomial. Then

$$\begin{aligned} \mathbf{T}F(x) &= P(x) \rightarrow \mathbf{T}Q(x) \\ &= 2[\mathbf{Tw}P(x)^*\mathbf{T}Q(x)] \end{aligned} \quad (80)$$

6.2. *Bitrinal form*

From 6.1. it follows that

$$\begin{aligned} \mathbf{T}[\mathbf{T}F(x)] = \mathbf{B}F(x) &= P(x) \rightarrow \mathbf{T}[\mathbf{T}Q(x)] \\ &= P(x) \rightarrow \mathbf{B}Q(x) \\ &= 2[\mathbf{Tw}P(x)^*\mathbf{B}Q(x)] \end{aligned} \quad (81)$$

6.3. *Twin form*

$$\begin{aligned} \mathbf{Tw}F(x) &= \mathbf{Tw}P(x) \rightarrow \mathbf{Tw}Q(x) \\ &= 2[\mathbf{TwTw}P(x)^*\mathbf{Tw}Q(x)] \\ &= 2[P(x)^*\mathbf{Tw}Q(x)] \end{aligned} \quad (82)$$

6.4. *Reverse form*

We have three cases

(*a*) $$\mathbf{R}F(x) = \mathbf{R}P(x) \rightarrow \mathbf{R}Q(x) = 2[\mathbf{RTw}P(x)^*\mathbf{R}Q(x)] \quad (83)$$

if

$$P(x) = 2 \oplus p(x_1, \ldots, x_{n-1}) \oplus a_nx_n$$

and

$$Q(x) = 2 \oplus q(x_1, \ldots, x_{m-1}) \oplus 2x_m$$

where $a_n = 1$ or 2 and n and m are the degrees of $P(x)$ and $Q(x)$ respectively.

(*b*) $$\mathbf{R}F(x) = \mathbf{TwR}F(x) \rightarrow \mathbf{R}F(x) = 2[\mathbf{R}P(x)^*\mathbf{R}Q(x)] \quad (84)$$

if

$$P(x) = 2 \oplus p(x_1, \ldots, x_{n-1}) \oplus a_nx_n$$

and

$$Q(x) = 2 \oplus q(x_1, \ldots, x_{m-1}) \oplus x_m$$

where $a_n = 1$ or 2 and n and m are the degrees of $P(x)$ and $Q(x)$ respectively.

(*c*) $$\mathbf{R}F(x) \neq \mathbf{R}P(x) \rightarrow \mathbf{R}Q(x) \quad (85)$$

if either $P(x)$ or $Q(x)$ is nonlinear in their highest order variables (x_n or x_m).

7. Equivalent related forms

The three polynomial operators **T**, **B**, and **Tw**, may be used singly or in combination to produce up to six related forms. If only cyclic forms are to be considered then the reverse operator **R** may also be employed enabling up to twelve related forms to be derived. These are

the original	$F(x)$	the reverse	$\mathbf{R}F(x)$
the trinal	$\mathbf{T}F(x)$	the reverse trinal	$\mathbf{RT}F(x)$
the bitrinal	$\mathbf{B}F(x)$	the reverse bitrinal	$\mathbf{RB}F(x)$
the twin	$\mathbf{Tw}F(x)$	the reverse twin	$\mathbf{RTw}F(x)$
the twin trinal	$\mathbf{TwT}F(x)$	the reverse twin trinal	$\mathbf{RTwT}F(x)$
the twin bitrinal	$\mathbf{TwB}F(x)$	the reverse twin bitrinal	$\mathbf{RTwB}F(x)$

Although more combinations are apparent we find that many of these are equivalent. In some cases the equivalence is obvious; thus

$$\mathbf{TB}F(x) = \mathbf{BT}F(x) = \mathbf{TTT}F(x) = F(x) \quad (86)$$

and

$$\mathbf{TwTw}F(x) = F(x) \quad (87)$$

also

$$\mathbf{RTw}F(x) = \mathbf{TwR}F(x) \quad (88)$$

Others are less obvious, for example

$$\mathbf{TwB}F(x) = \mathbf{TTw}F(x) \quad (89)$$

$$\mathbf{TwT}F(x) = \mathbf{BTw}F(x) \quad (90)$$

$$\mathbf{TwBTw}F(x) = \mathbf{T}F(x) \quad (91)$$

$$\mathbf{TwTTw}F(x) = \mathbf{B}F(x) \quad (92)$$

$$\mathbf{TTwT}F(x) = \mathbf{BTwB}F(x) = \mathbf{Tw}F(x) \quad (93)$$

$$\mathbf{TTwB}F(x) = \mathbf{TwT}F(x) \quad (94)$$

$$\mathbf{BTwT}F(x) = \mathbf{TwB}F(x) \quad (95)$$

We see that an algebra of these operators soon begins to emerge.

8. Conclusions

The general describing polynomial of the nonlinear ternary fsr has been set up and various properties investigated. The related forms which correspond to simple operations in the sequence domain have been derived and transform methods have been set up which enable each one to be generated from the original form.

The concept of polynomial composition has been introduced and the process of forming the product of two nonlinear polynomials has been fully described. The sequence domain behaviour of this composite form is then equivalent to a cascade arrangement of the factor fsrs.

The consequences of performing the above polynomial operations on this composite form have also been investigated. Finally, a number of results concerning certain equivalences under the application of combinations of the polynomial operators are listed.

The results presented in this paper go some way to providing a system for the analysis, design, and classification for the ternary fsr. Within the general division into cyclic and non-cyclic forms we may envisage a further partition into irreducible and composite types. Also, the related forms of these

polynomials may be considered, in a sense, to be equivalent, differing only by the application of a suitable transform, and the cycle set being identical in each case. Furthermore, invariances under these operators will often lead to special types with interesting sequence domain behaviour, and in so doing, provide a further classification technique.

One property arising in this nonlinear ternary regime which has no counterpart in the equivalent binary field is that associated with certain maximal length cyclic fsrs. Such devices generate a single cycle of length 3^n digits and unlike the binary equivalent, which can be shown to be generated only by irreducible describing polynomials, certain of these emanate from composite polynomial forms. What is more, the factors of these polynomials are themselves polynomials describing maximal length fsrs of lower order. For example, $(2 \oplus x_1)'$ describes an order 1 maximal length fsr which generates a single cycle of length 3 which is 201. The polynomial $(2 \oplus 2x_1 \oplus x_1^2 \oplus 2x_1^2x_2 \oplus 2x_2)''$ describes an order 2 maximal length fsr which generates the single cycle of length 9, 220021101. The product of these two polynomials, which represents a cascade connection of the two factor fsrs, is $(2 \oplus x_1^2 \oplus 2x_1^2x_2 \oplus 2x_2^2 \oplus x_2^2x_3 \oplus x_3)'$ which describes an order three composite maximal length fsr which generates a single cycle of 27 digits, namely,

$$222011020001202121112210100\,.$$

Since modular algebra is functionally complete for a p-valued system if p is prime, it is not difficult to generalise most of the preceding results to the p-nary case. For example the generalised polynomial-to-sequence domain transformation matrix for a function of a single variable is easily shown to have the form

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & \ldots, 0 \\ 1 & 1 & 1 & \ldots, 1 \\ 1 & 2 & 2^2 & \ldots, 2^{p-1} \\ 1 & 3 & 3^2 & \ldots, 3^{p-1} \\ . & . & . & . \quad . \\ . & . & . & . \quad . \\ 1 & (p-1) & (p-1)^2 & \ldots, (p-1)^{p-1} \end{bmatrix}.$$

The inverse transformation may be found by inverting this matrix using the operations of addition and multiplication modulo-p. Higher order matrices are built up as before. Similarly, we can imagine many related forms in the general modulo-p case, which result from the sequence domain operations of adding or multiplying by the integers modulo-p.

Difficulties arise when one considers non-prime radices because in these cases straightforward modular algebra breaks down due to the existence of divisors of zero. For example, $2 \odot 3 = 6 \equiv 0$ modulo-6, which implies that $0 \div 2 = 3$. However, we can devise consistent algebras in certain cases. If the radix is a power of a prime, i.e. $r = p^k$, then it is possible to construct a viable algebraic system using the powers of a primitive element from the Galois field $GF(p^k)$ as symbols rather than the integers modulo-r. For example, the addition and multiplication tables for $r = 4 = 2^2$ using the symbols 0, $a^0 = 1$, $a^1 = a$, and $a^2 = b$, are found to be

$\odot$	0	1	a	b
0	0	0	0	0
1	0	1	a	b
a	0	a	b	1
b	0	b	1	a

$\oplus$	0	1	a	b
0	0	1	a	b
1	1	0	b	a
a	a	b	0	1
b	b	a	1	0

Using this algebra we may, once again, build up a theoretical description of the base-4 feedback shift register.

Unfortunately, when the radix is a product of distinct primes (e.g. $r = 6$) no straightforward algebra with similar structure to those described above seems to be available.

References

ELSPAS, B. (1959). The theory of autonomous linear sequential networks, *IERE Trans. on Circuit Theory*, Vol. CT-6, No. 1, p. 45.

GODFREY, K. R. (1966). Three-level *m*-sequences, *Electronics Letters*, Vol. 2, No. 7, p. 241.

GREEN, D. H. and DIMOND, K. R. (1970a). Polynomial representation of nonlinear feedback shift registers, *Proc. IEE*, Vol. 117, No. 1, p. 56.

GREEN, D. H. and DIMOND, K. R. (1970b). Nonlinear product feedback shift registers, *Proc. IEE*, Vol. 117, No. 4, p. 681.

GREEN, D. H. and KELSCH, R. G. (1972). Ternary pseudonoise sequences, *Electronics Letters*, Vol. 8, No. 5, p. 112.

HUFFMAN, D. A. (1956). The synthesis of linear sequential coding networks, *Proc. 3rd London symp. on Inf. Theory*, Butterworth.

KELSCH, R. G. and GREEN, D. H. (1971). Nonbinary negacyclic code which exceeds Berlekamp's (p-1)/2 bound, *Electronics Letters*, Vol. 7, No. 22, p. 664.

KELSCH, R. G. (1972). *Non-binary logic systems*, a Ph.D. Thesis to be submitted to the University of Manchester.

LEE, S. C. and LEE, E. T. (1972). On multivalued symmetric functions, *IEEE Trans. Computers*, March, p. 312.

O'CARROLL, L. M. (1972). *A simulation of ternary sequential systems*, an M.Sc. Thesis to be submitted to the University of Manchester.

SANTOS, J. and ARANGO, H. (1964). Base 3 vs. Base 2 synchronous arithmetic units, *IEEE Trans. Computers*, October, p. 608.

TURECKI, A. (1968). The ternary number system for digital computers, *Computer Design*, February, p. 66.