# LFSRs - configuração Galois (PSI-3451/20)

Um linear feedback shift-register, LFSR, é um registrador de deslocamento composto de N registradores que possui algum tipo de realimentação entre saídas e entradas passando por operações Booleanas. No caso particular da configuração de Galois (existe também a configuração de Fibonacci, que não será estudada neste curso) a realimentação se dá entre a saída do LFSRs e as entradas de alguns de seus registradores passando por portas do tipo OU EXCLUSIVO (ver figura 1). Este circuito apresenta diversas utilidades no mundo eletrônico, sendo que no nosso caso ele será utilizado como gerador de uma sequência de número pseudo-aleatórios.

### 1. Geração de LFSR e Polinômio Primitivo

Na configuração Galois, o LFSR de grau *N* é um registrador de deslocamento onde o próximo estado de todos os elementos de armazenamento, da direita para a esquerda (LSB para o MSB), como mostrado na Figura 1, é igual a

$D_i=Q_{i-1}\oplus a_iQ_{n-1}$, para i = 1..n-1 (Equação 1),

O próximo estado de cada registrador (*i*) pode assumir 1 entre 2 valores possíveis:

- o valor do estado atual do registrador precedente (*i-1*)

  $D_i=Q_{i-1}$, para i = 1..n-1 (logo, $a_i=0$) (Equação 2).

ou

- o resultado da soma exclusiva entre o estado atual do registrador precedente (*i-1*) e o estado do último registrador do encadeamento (*N*).

  $D_i=Q_{i-1}\oplus Q_{n-1}$, para i = 1..n-1 (logo, $a_i=1$) (Equação 3).

O primeiro registrador do encadeamento (o mais à esquerda na figura 1, de índice zero) recebe sempre o valor do estado atual do registrador *N*.

Observe-se que se num determinado estado do LFSR o valor do MSB for $D_{n-1}=0$, então todos os operadores XOR terão a entrada da realimentação com valor lógico '0', portanto,

$D_i=Q_{i-1}$, para i = 1..n-1 (Equação 4).

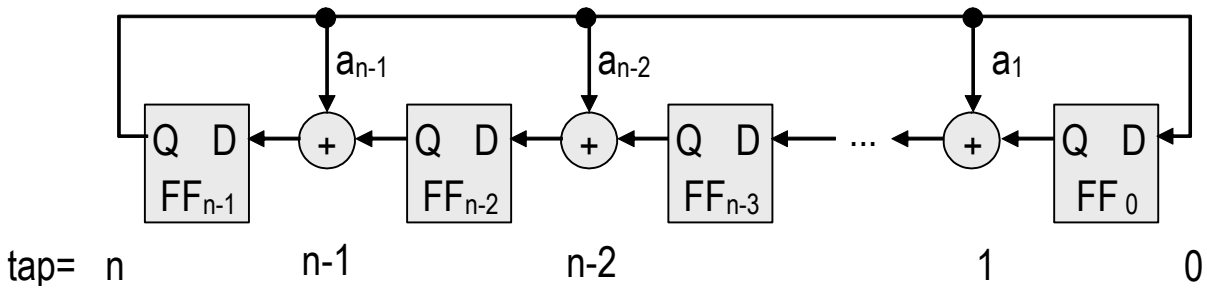


Figura 1. LFSR- configuração Galois

Cada realimentação é conhecida por *tap* recebendo o índice (enumeração) do registrador cuja entrada será afetada. Por exemplo, os *taps* do LFSR da Figura 2 são os de índices 2 e 3. A saída do último estágio do LFSR (registrador *N*) é o *tap N* (na Figura 2 é o *tap* 4). Observe-se que o índice *N* (correspondente ao MSB) sempre aparece na enumeração de *taps*, apesar de nunca haver um XOR nesta posição; ela é necessária para estabelecer o tamanho do LFSR. Nunca haverá *tap* no registrador 0 (correspondente ao LSB).

O exemplo da Figura 2 ilustra o caso de LFSR (4, 3, 2).

A equação de estados do LFSR pode ser expressa por um **polinômio característico** na forma:

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + ... + a_1 x + a_0$$

onde **obrigatoriamente** $a_0=1$ e $a_n=1$ para um polinômio de grau *n* (correspondendo a um LFSR com N registradores). Os demais coeficientes serão 0 ou 1 de acordo com a existência ou não das realimentações. Então, o polinômio característico do LFSR da figura 1 é:

$$p(x) = x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + ... + a_1 x + 1$$

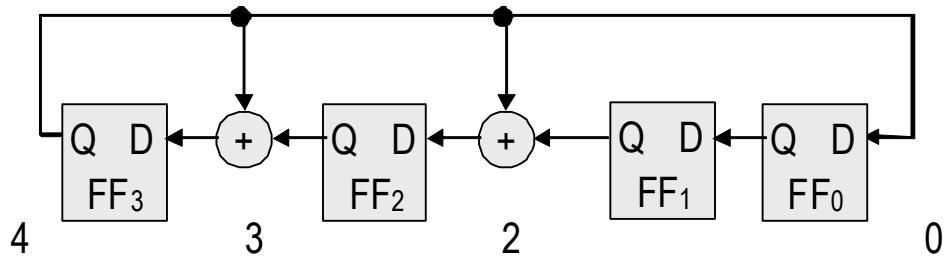


Figura 2. LFSR (4, 3, 2)

O polinômio característico do LFSR da figura 2 é

$$p(x) = x^4 + x^3 + x^2 + 1$$

A máquina de estados correspondente do LFSR gera uma sequência pseudo-aleatória de vetores representados pelos bits das saídas dos registradores (estado atual), $Q_{n-1}$, $Q_{n-2}$,...$Q_1$, $Q_0$. Há duas questões bastante relevantes no uso dos LFSRs como geradores de sequências aleatórias:

1) Tamanho da sequência: a teoria diz que a sequência psudo-aleatória é máxima somente quando o polinômio for primitivo, ou seja, ele não é divisível por outro polinômio qualquer. Por exemplo, o polinômio de LFSR (4, 3) é primitivo, com uma sequência de tamanho máximo $n^4$-1=15. Quando o polinômio não for primitivo, pode-se ter várias sequências independentes de tamanhos menores.

2) Semente: os LFSRs devem adotar uma condição ou estado inicial para os registradores. Mas, a semente não pode ser uma sequência de 0 (0,0,..,0). Neste caso o LFSR NUNCA MUDARIA DE ESTADO. O estado inicial pode ser qualquer outra combinação de valores (por exemplo, no caso do LFSR da figura 2 a semente, por exemplo poderia ser 0001 ou 1111, ou outra combinação de 4 valores binários exceto a sequência de 4 zero´s.

No caso de polinômios não primitivos, não se conhecem métodos que permitam descobrir de antemão, qual deve a ser a semente para se obter a maior sequência dentre as existentes.

**2. Simulação por software on-line (**https://leventozturk.com/engineering/crc/**)**

O sítio da Internet acima apresenta opções de cálculo on-line de geradores aleatórios, assim como a obtenção automática de código de hardware e software em algumas linguagens de referência. Inclui-se aí o gerador baseado no LFSR-Galois. Este recurso será utilizado pelo aluno durante o teste de hardware projetado, para aumentar a confiabilidade do projeto implementado.

A utilização dos recursos on-line é razoavelmente autoexplicativa, porém, para maior produtividade no seu uso, algumas dicas são adiantadas a seguir:

1) Bloco Configure:

- Atentar que há necessidade de se selecionar "Galois LFSR" no campo *Type* a cada computação. Infelizmente, o programa não guarda a último tipo utilizado e sempre retorna à opção CRC.

- Para o polinômio, utilize o formato $X_n$ to $X_0$. Todos os n *taps* devem estar presentes: os de índice n e 0 são obrigatórios.

- Para a *seed* (semente) no campo *Initialise*, o posicionamento é o seguinte: MSB ($Q_{n-1}$ do LFRS) deve ficar posicionado mais à esquerda e o LSB ($Q_0$ do LFRS), mais à direita. ATENÇÃO: é obrigatória a adição de um "0" extra à esquerda da semente de n bits (ou seja, o campo conterá n+1 bits para a computação do software), apesar deste bit não ter papel real no LFSR.

- *Data Width* representa o número de estados percorridos pelo LFSR (número de Clock´s) antes de se observar o valor das saídas. Deve ser um valor entre "1" e "63". O simulador on-line realiza a computação de uma sequência de tamanho definido Data Width, porém só o último padrão resultante é mostrado no campo de saída. A fim de se conhecer a sequência de M vetores de saída a partir de uma semente qualquer é preciso fazer M computações individuais inserindo-se valores crescentes no campo de Data Width (1, 2,3, .....M). Para verificar a sequência toda, caso ela maior que 63, deve-se repetir a computação, usando-se o resultado da computação mais longa como nova semente.

Obs. Deixe o campo *Process Direction* no default d[n] to d[0]; parece não ter efeito nos cálculos para um LFSR Galois

2) Bloco Generate Code - não é usado

3) Bloco Calculate Output:

- Campo *Input Data* : não se aplica a LFSR Galois. Pode-se deixar com valor 1.

- Para o campo *Output Format*, adote o[n] para o[0], em razão de manter o mesmo padrão [MSB,..,LSB] usado para o *tap* e *seed*.

- O campo *Galois LFSR output* apresenta o padrão final após a sequência de padrões de tamanho definido por *Data Width*. Estará no formato o[n] para o[0], como selecionado acima.