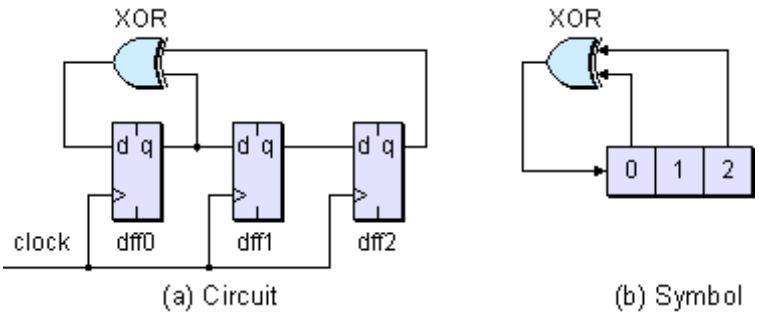
- ## Opis problemu:

LFSR (ang. Linear Feedback Shift Register) – rejestr przesuwny którego bit wejściowy jest linową funkcją jego poprzedniego stanu.

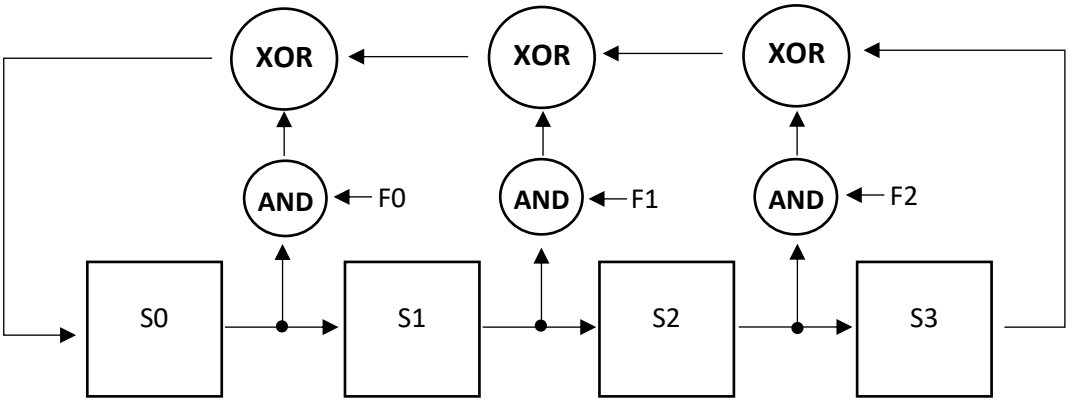


Maksymalna długość okresu jest ograniczona przez stopień stowarzyszonego z nim wielomianu i wyrażona jest wzorem: $2^d - 1$, gdzie d jest stopniem wielomianu (długością rejestru).

Okres danego LFSR jest maksymalny jeżeli stowarzyszony z nim wielomian jest wielomianem pierwotnym. Rejestr taki, nazywamy rejestrem maksymalnej długości.

- ## Opis algorytmu:

Algorytm symuluję pracę sprzętowego generatora liczb losowych LFSR.
Na podstawie danych wprowadzonych przez użytkownika lub odczytanych z pliku generuje on pseudolosową sekwencję liczb. Swoją pracę kończy po wypisaniu całego ciągu i zapisaniu go do pliku. Zależnie od ustawień, algorytm w różny sposób realizuje sprzężenie zwrotne. Idea jednak została oparta na poniższym schemacie:



***LFSH** z zewnętrzynym sprzeżeniem zwrotnym.*

$S_0 - S_n$ komórki (bity) rejestru
$F_0 - F_{n-1}$ – wektor charakterystyczny opisujący feedback
Bramki AND pozwalają na dowolną manipulację sprzężeniem zwrotnym.
Gdy $F_0 = 0$ bramka AND 'nie przepuszcza' sygnału z rejestru, zachowuje się jak przerwa w obwodzie. Z kolei gdy $F_0 = 1$ oraz $S_0 = 1$ na bramkę XOR podawana jest logiczna jedynka.

LFSR:

- Tworzę dynamiczną tablicę reg[n] **reprezentującą rejestr przesuwny**,
- Wpisuję do tablicy stan początkowy rejestru,
- Tworzę kopię tablicy reg[n], na której będę wykonywał operacje, a wyniki porównywał z oryginałem,
- Jeżeli tablice zrównają się, będzie to oznaczało że algorytm wygenerował całą sekrecje, jest to warunkiem zakończenia rekurencji,
- Przed każdym przesunięciem rejestru, realizuję sprzężenie zwrotne, wewnętrzne lub zewnętrzne,
- W przypadku **zewnętrznego**, zgodnie z wektorem opisującym sprzężenie (pomijając pozycję zero) sumuję odpowiednie bity w rejestrze a następnie wykonuję operację modulo 2. Następnie przesuwam rejestr a otrzymany z wcześniejszego działania wynik wpisuję na początek rejestru.
- W przypadku **wewnętrznego**, zgodnie z wektorem opisującym sprzężenie zwrotne w odpowiednie komórki wpisuję wynik modulo 2 sumy ostatniego bitu rejestru i komórki do której ma trafić wynik. Następnie przesuwam rejestr i na początek rejestru wpisuję wartość która znajdowała się na jego końcu.
- Przy każdym przesunięciu rejestru ostatni jego bit zostaje wypisany w konsoli Windows i zapisany do pliku output.txt

Fragment kodu:

Funkcja wywoływana rekurencyjnie:

```
int lfsr(bool *reg, bool *reg_copy, int regLength, bool *feedback, int fbType)
{
    int sum = fbSum(fbType, regLength, reg_copy, feedback);
    print(reg_copy[regLength-1]);
    for(int i=regLength-1; i>=0; i--) reg_copy[i+1] = reg_copy[i];
    reg_copy[0] = sum;
    if(!check(reg, reg_copy, regLength))
    lfsr(reg, reg_copy, regLength, feedback, fbType);
    else save();
    return 0;
}
```

Funkcja realizująca sprzężenie zwrotne:

```
bool fbSum(int fbType, int regLength, bool *reg_copy, bool *feedback)
{
    int sum=0;
    if(fbType == 1 ) {for(int i=1; i<regLength; i++)
        if(feedback[i] == 1)sum=sum+reg_copy[i];}
    else if(fbType == 2)
    {
        sum = reg_copy[(regLength - 1)];
        for(int i=1; i<(regLength - 1); i++)
            if(feedback[i] == 1) reg_copy[i] = (sum + reg_copy[i])%2;
    }
    sum = sum % 2;
    return sum;
}
```

Funkcja porównująca tablice, warunek zakończenia rekurencji:

```
1  bool check(bool *arr1, bool *arr2, int n)
2  {
3      for (int i = 0; i < n; i++)
4          if (arr1[i] != arr2[i])
5              return false;
6      return true;
7  }
```

Przykładowe wywołanie programu:

```
LinearFeedbackShiftRegister

1.Wczytaj ustawienia z pliku;
2.Wpisz ustawienia recznie;
2
Dlugosc rejestru: 4
Stan poczatkowy rejestru(np.: 10101010): 1101
Rodzaj sprzezenia:
1.Zewnetrzne;
2.Wewnetrzne;
1
Wektor opisujacy sprzezenie zwrotne (np.: 10010001): 1011
101111000100110
```

```
LinearFeedbackShiftRegister

1.Wczytaj ustawienia z pliku;
2.Wpisz ustawienia recznie;
2
Dlugosc rejestru: 4
Stan poczatkowy rejestru(np.: 10101010): 1101
Rodzaj sprzezenia:
1.Zewnetrzne;
2.Wewnetrzne;
2
Wektor opisujacy sprzezenie zwrotne (np.: 10010001): 1011
110101100100011
```

Powyższe wyniki zgadzają się z sekwencjami obliczonymi ręcznie.

```
LinearFeedbackShiftRegister
1.Wczytaj ustawienia z pliku;
2.Wpisz ustawienia recznie;
1
Podaj nazwe pliku.txt: dataIn.txt
1000101011010111111110001010011000001011100100001001101110011101110001111000111111011010
1011010001110100001010111011101101100011111111101100110100000011100000101100010110100110
1001110101010001010011100000111100100110100101010011000101010000100001110000110111101011
0101001110110010100010011110001001111100101011110010010001101001101101111001010001101
1101001001110011111101011001100010011001000000111100000111000101010100101101110000010101
0111101000110000011011011100010010011011110111101100011101111011100110111110111110000111
0011110010101011000101001000001001110010111001110011001011100011000000101110110001000110
1111100111111001110010001000100011001111111111110000010000000001010011110000001100100101
0100010000110100001111111010010110101000110101010100011100100000001110110011100001011110
0011010111011010100100101101000001011010101011001101010000001000001000110001111001111110
0011010100010010110010001111101010001111010011000111010110100001101110010101101001111001
0001010100101001111100001010010010010101100101101101000
Dlugosc sekwencji: 1022
```

8-bitowy rejestr, sprzężenie zewnętrzne, odczyt ustawień z pliku

```
LinearFeedbackShiftRegister
1.Wczytaj ustawienia z pliku;
2.Wpisz ustawienia recznie;
1
Podaj nazwe pliku.txt: dataIn.txt
1010110110101100101100001111101101111010111010001000011011000111100111001100010110100
1000101001010100111011101100111101111110100110011010100011000001110101010111110010100
0010011111111100001011110001101000000010001110001001011100000011001001001101110010000
Dlugosc sekwencji: 254
```

12-bitowy rejestr, sprzężenie wewnętrzne, odczyt ustawień z pliku

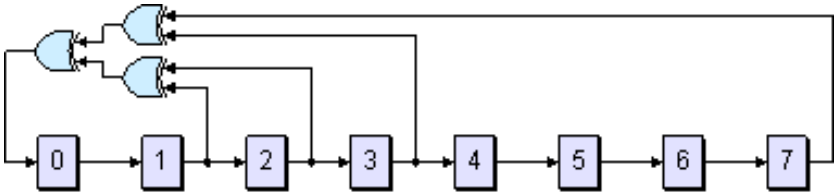
```
LinearFeedbackShiftRegister
1.Wczytaj ustawienia z pliku;
2.Wpisz ustawienia recznie;
1
Podaj nazwe pliku.txt: dataIn.txt
1100101010000001001010100101111010010010101100110110011111101100101111101001110001010111101000100110111000
0011101110101011000001001110100001001010111011111100100101111001001010111010011000010001001010101100111011
1111100000111001111010101011011000011100001111110000110001111010001011111110011000100101011111101110001001
1101110100010000000100010001110011011000010001101110101010001001000011000100110000111001010111101100000011
0001110011001010010010111011111010010101010100010111010011011111111101100110100101011000100100110011010111100
0100101001101011111100101011000000000110101100101111001101111011111001010101011000011011110111010000000100001
1100100010110111101111101000011101011100111111100001110001110111001110100010010010100001010111000101100000
1100110011100101100001110100011111111010101000000000001010100001111110100010001001100011100010110100010011
1110000000011011111010110001010010100100011011111000100001001001101111101111100101111100000110000111011000
0111110111001101111010000100001111011001010110010010001110000000000000011100000010010001101011000010110010
1100010101100001001011011011110010011100110101011100001101000100101100000011000010100100010010111001001100
1100110111111001101001000100001010010101101011010010100001110011000011010011011011011011111011011101011001
1100110111001100010110001111011000011110011111110101111101000111001011010001101011100110110100000011101011
0000111110101111001100111111101000110000011011100001110011100111010101101010011001101111000010100010100100
1010001000111101001100111110100000111101111110011011111101001001110010011101111101010001101000111110100010
010110
Dlugosc sekwencji: 1580
```

16-bitowy rejestr, sprzężenie zewnętrzne, odczyt ustawień z pliku

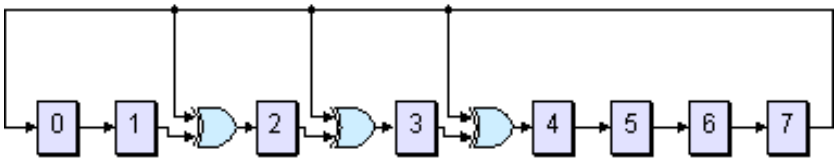
- ## Wnioski:

  Jak widać program nie zawsze generuję ciągi o maksymalnej długości. Jest to ściśle powiązane z wektorem sprzężenia zwrotnego i ilością bramek XOR które biorą w nim udział.

  W przypadku implementacji softwarowej rodzaj zastosowanego sprzężenia nie ma znaczącego wpływu na szybkość oraz stabilność programu. Inaczej jednak sytuacja wygląda w rzeczywistym układzie elektronicznym (hardware). Przy zastosowaniu zewnętrznego sprzężenia zwrotnego (rys. A) w gałęzi sprzężenia występuje wiele rożnych stanów logicznych co w znaczącym stopniu ogranicza maksymalną częstotliwość pracy całego układu. Natomiast w przypadku zastosowana wewnętrznego sprzężenia zwrotnego (rys. B) panuje tam tylko jeden stan logiczny i chociaż wygenerowane sekwencje będą się różnić to pseudo-losowość zostaje zachowana a problem rozwiązany.



(a) Many-to-one implementation

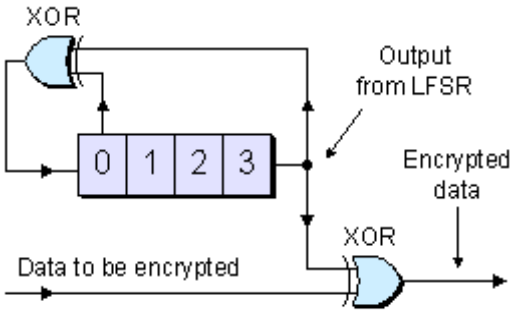


(b) One-to-many implementation

Zastosowanie

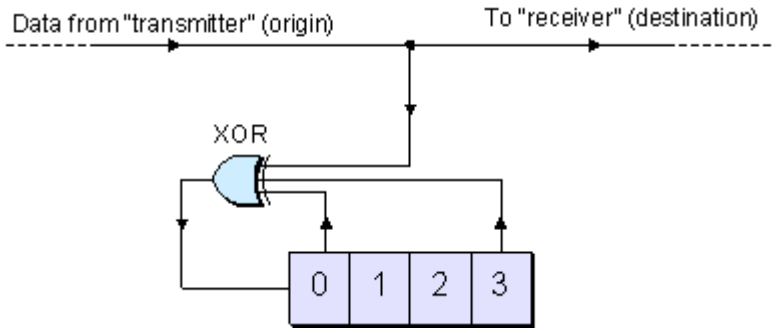
Dzięki prostocie układu, łatwości jego implementacji oraz mnogości funkcji jakie może pełnić, LFSR znalazł szerokie zastosowanie w technice komputerowej.

- **Kryptografia**



Wygenerowany pseudo-losowy ciąg bitów można wykorzystać jako klucz **szyfru strumieniowego** (stream ciphers). W celu przełamania liniowości stosuje się odpowiednie metody zapewniające większy poziom bezpieczeństwa. Do ważnych szyfrów strumieniowych należą **A5/1** oraz **A5/2** używane w sieciach GSM oraz **E0** używany w Bluetooth. Warto zaznaczyć ze standard A5/1 został skompromitowany, a A5/1 i E0 zawierają poważne słabości.

- **Protokoły komunikacyjne**
  LFSR może zostać wykorzystany jako CRC (Cyclic Redundancy Code / Cykliczny kod nadmiarowy) do wykrywana błędów przypadkowych, powstających podczas np. transmisji danych cyfrowych. Używają go takie standardy jak: USB3.0, PCI Express 3.0, 100,1000BASE-T Ethernet i wiele innych.



Są to jedne z wielu zastosowań które wydały mi się najciekawsze do przedstawienia.

- Program prawidłowo symuluje działanie rejestru przesuwnego z liniowym sprzężeniem zwrotnym. Prawidłowo waliduje dane wprowadzone przez użytkownika jak również te odczytane z pliku. Generowane ciągi wyświetla w konsoli Windows oraz zapisuje do pliku. Spełnia wszystkie wstępne założenia.

Źródła:

[1]. Christof Paar - Understanding Cryptography: A Textbook for Students and Practitioners.

[2]. Max Maxfield - Linear Feedback Shift Registers (LFSRs) – Part 1-3.