**TEMAT : KODY CYKLICZNE**

**Wprowadzenie**

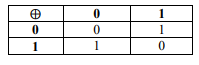
1. **Cel ćwiczenia:**

Celem ćwiczenia jest implementacja kodów cyklicznych na rejestrach przesuwających z liniowym sprzężeniem zwrotnym. Ćwiczenie będzie przeprowadzone w pakiecie MATLAB/SIMULINK.

1. **Wstęp teoretyczny:**

LFSR (Linear Feedback Shift Register) – zbudowany jest z prostych rejestrów przesuwnych z niewielką liczbą bramek XOR i liniowym sprzężeniem zwrotnym. Strumień wartości wytwarzanych przez rejestr jest całkowicie determinowany przez jego aktualny (lub poprzedni) stan ( bit wejściowy jest funkcją liniową poprzedniego stanu rejestru). Sygnał z wyjścia rejestru poprzez sumę modulo 2 ( XOR) wprowadzany jest na wejścia wybranych przerzutników.

|  |  |  |
| --- | --- | --- |
| **XOR** | **0** | **1** |
| **0** | 0 | 1 |
| **1** | 1 | 0 |



Rysunek 2 XOR

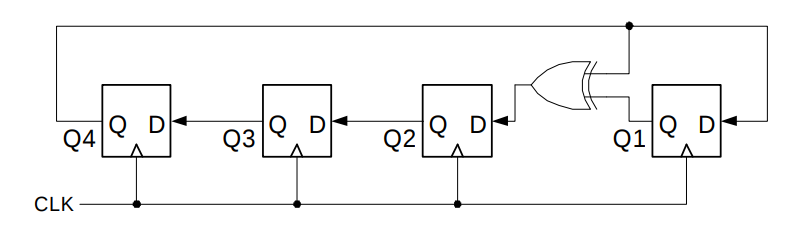
Rysunek 1 Suma modulo 2

Używany głównie do:

- generowania liczb losowych,

- jako licznik,

- generowania i sprawdzania sumy kontrolnej w kodach CRC.



Rysunek 3 Przykładowy 4-bitowy LFSR

Kombinacyjny układ sprzężenia zwrotnego reprezentowany jest przez określony wielomian P(x):

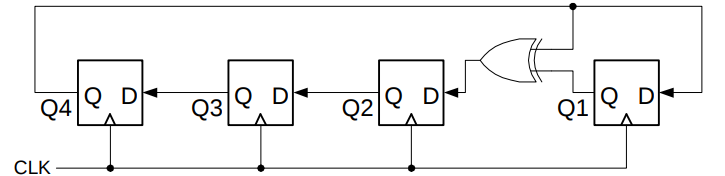
Gdzie:

* x – zmienna,
* Współczynniki mają wartość 0 lub 1 w zależności od tego czy n-te wyjście przerzutnika zostało użyte w sprzężeniu zwrotnym.

Wyrażenie na początku wielomianu odpowiada podłączeniu sprzężenia zwrotnego bezpośrednio do wejścia przerzutnika .

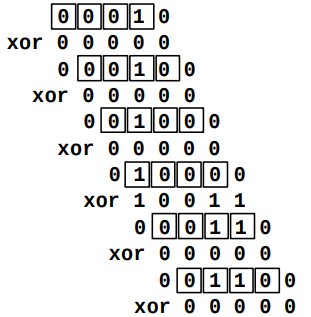
Czyli nasz przykładowy 4-bitowy LFSR jest określony wielomianem .

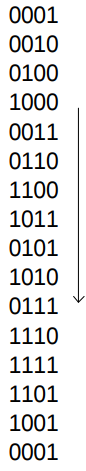
Przeanalizujmy teraz zachowanie naszego 4-bitowego LFSR.

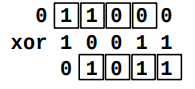




Zawartość poszczególnych przerzutników przedstawmy następująco:







Rysunek 4 Poszczególne operacje LFSR oraz zawartości przerzutników

Bity, które nie uczestniczą w sprzężeniu zwrotnym, są przesuwane o jedną pozycję w lewo bez zmian, a bity, które uczestniczą w sprzężeniu zwrotnym są XORowane bitem wyjściowym, zanim zostaną zapisane w następnej pozycji. Nowy bit wyjściowy jest kolejnym bitem wejściowym.

Skutkuje to tym, że gdy bit wyjściowy ma wartość zero, wszystkie bity rejestru przesuwają się w lewo bez zmian, a bit wejściowy staje się zerowy. Gdy bit wyjściowy ma wartość jeden, bity w sprzężeniu zwrotnym obracają się (jeśli są 0 à 1, a jeśli są 1 à 0), a następnie cały rejestr jest przesuwany w lewo, a bit wejściowy staje się 1.

Podstawowa koncepcja algorytmów CRC:

-traktujemy wiadomość jako liczbę binarną,

-dzielimy wiadomość przez inną stałą liczbę binarną,

-reszta z dzielenia jest naszą sumą kontrolną.

Przykład:

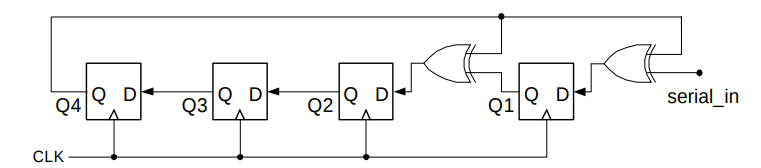
Wiadomość: D = 1011001 (7 bity)

Dzielnik: P = 10011 (5 bity)

Reszta z dzielenia: R = do obliczenia (4 bity)

Ramka do transmisji: T = 1011001XXXX

By wykonać dzielenie wprowadzamy wiadomość do LFSR.



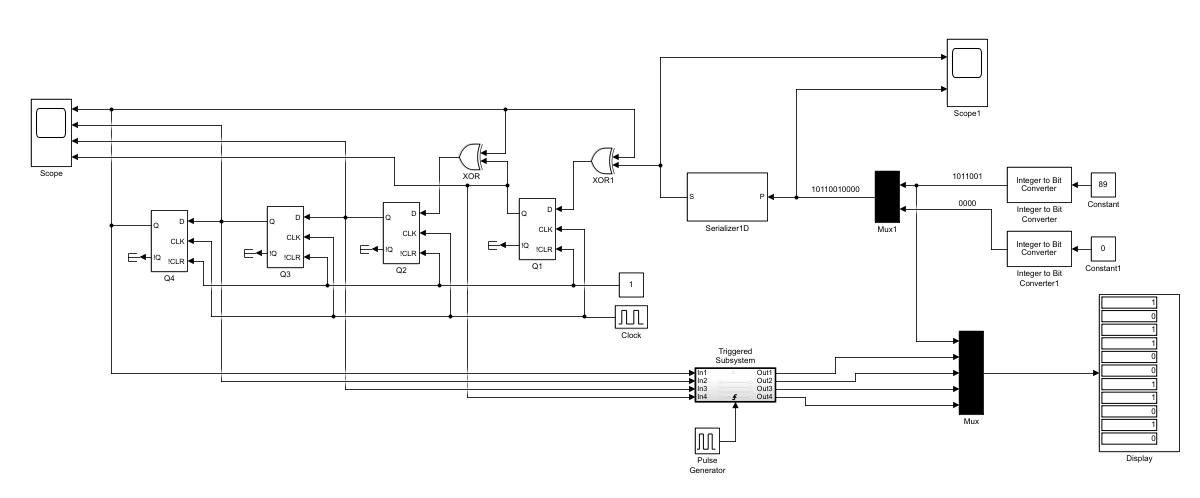
Poszczególne wartości w rejestrach, po wprowadzeniu naszej wiadomości do LFSR, przedstawione są w poniższej tabeli.

Tabela 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q4 | Q3 | Q2 | Q1 | Serial\_in |
|  |  |  |  | **10110010000** |
|  |  |  |  | **0110010000** |
|  |  |  |  | **110010000** |
|  |  |  |  | **10010000** |
|  |  |  |  | **0010000** |
|  |  |  |  | **010000** |
|  |  |  |  | **10000** |
|  |  |  |  | **0000** |
|  |  |  |  | **000** |
|  |  |  |  | **00** |
|  |  |  |  | **0** |
|  |  |  |  |  |

T = 1011001XXXX

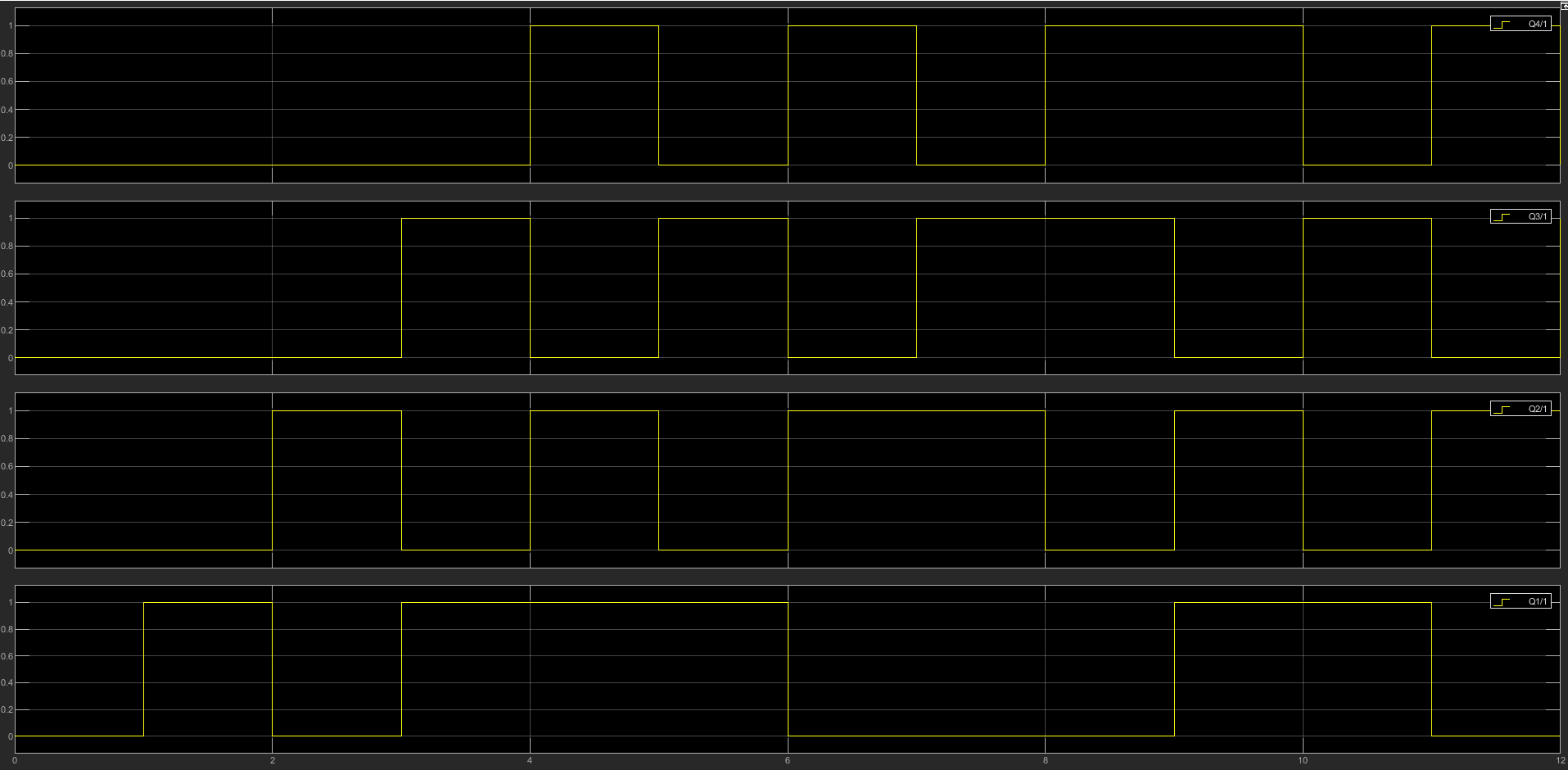
1. **Przykład:**
2. Zaimplementuj enkoder CRC na 4- bitowym LFSR



Rysunek 5 Enkoder CRC

Parametry:

* Clock à Period = 1
* Integer to Bit Converter à Number of bits per integer = 7
* Integer to Bit Converter1 à Number of bits per integer = 4
* Pulse Generator à Phase delay =11.5
* Serializer1D à Main: Ratio = 11
* Serializer1D à Signal Attributes: Input sample time = 11
* Stop time = 12

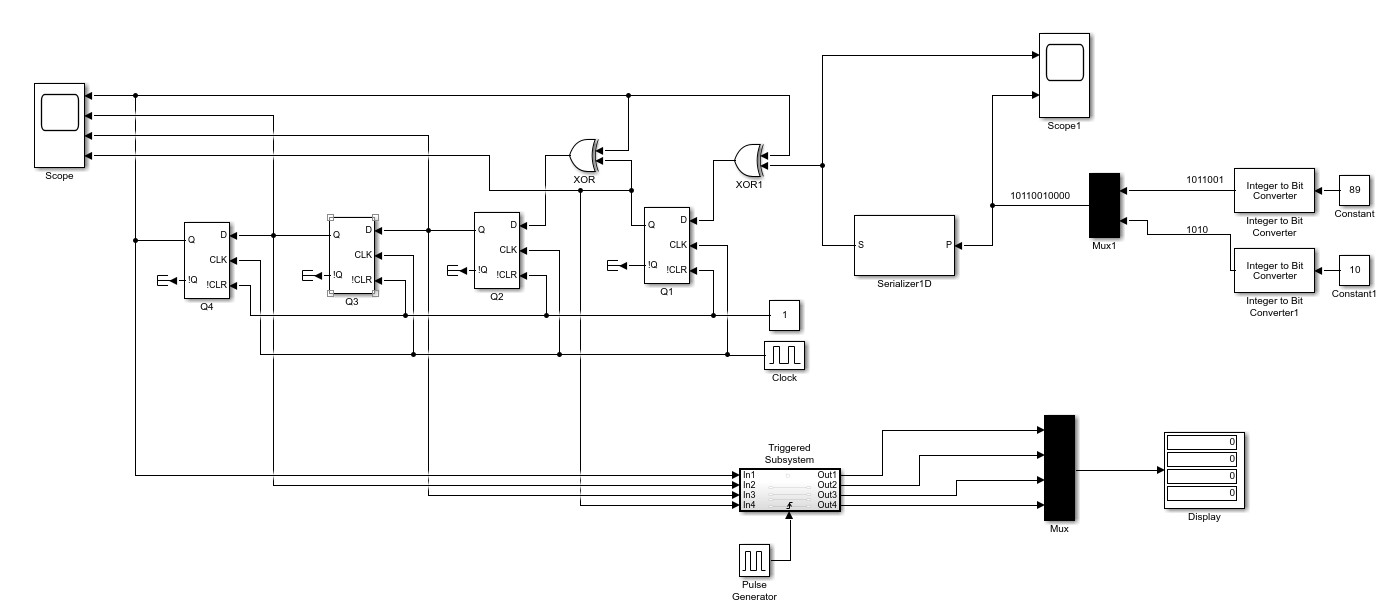
Sprawdźmy zawartość przerzutników oraz zgodność z Tabela 1.:

Rysunek 6 Zawartość Scope

Jak widzimy, na początku mamy 0000, potem 0001 itd. Zgodnie z Tabela 1.

Ostateczny wynik naszego enkodera oraz nasza ramka do transmisji:

T = 10110011010

1. Zaimplementuj dekoder CRC na 4-bitowym LFSR.

Rysunek 7 Dekoder CRC

Jak widzimy, gdy nie pojawią się żadne błędy w transmisji, wynik z dzielenia wynosi 0000.

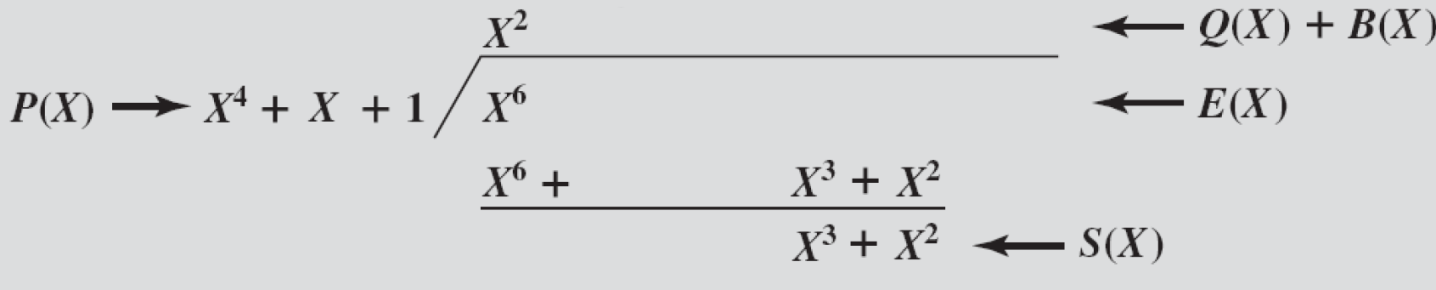
Co się stanie, gdy pojawią się błędy?

Istnieją dwa główne typy kontroli błędów.

* **ARQ** (Automatic Repeat Request) - to metoda kontroli błędów w transmisji danych, która wykorzystuje kody wykrywania błędów, komunikaty potwierdzające oraz limity czasu w celu uzyskania niezawodnej transmisji danych. Potwierdzenie to komunikat wysłany przez odbiorcę, wskazujący, że poprawnie odebrał ramkę danych. Zwykle, gdy nadajnik nie otrzyma potwierdzenia przed upływem limitu czasu, retransmituje ramkę do momentu, aż zostanie poprawnie odebrana lub błąd utrzyma się powyżej określonej liczby retransmisji.
* **FEC** (Forward Error Correction) - proces dodawania nadmiarowych danych do ramki danych, dzięki którym dane mogą zostać odzyskane przez odbiornik nawet w przypadku wystąpienia błędów.

**Cykliczna kontrola nadmiarowa (Cyclic Redundancy Check - CRC)** to jeden z typów FEC, pozwalający na wykrywanie i poprawianie przypadkowych błędów w danych cyfrowych w sieciach komputerowych. Zaletą wyboru prymitywnego wielomianu jako generatora kodu CRC jest to, że wynikowy kod ma maksymalną całkowitą długość bloku w tym sensie, że wszystkie błędy 1-bitowe w obrębie tej długości bloku mają różne reszty (zwane syndromami).

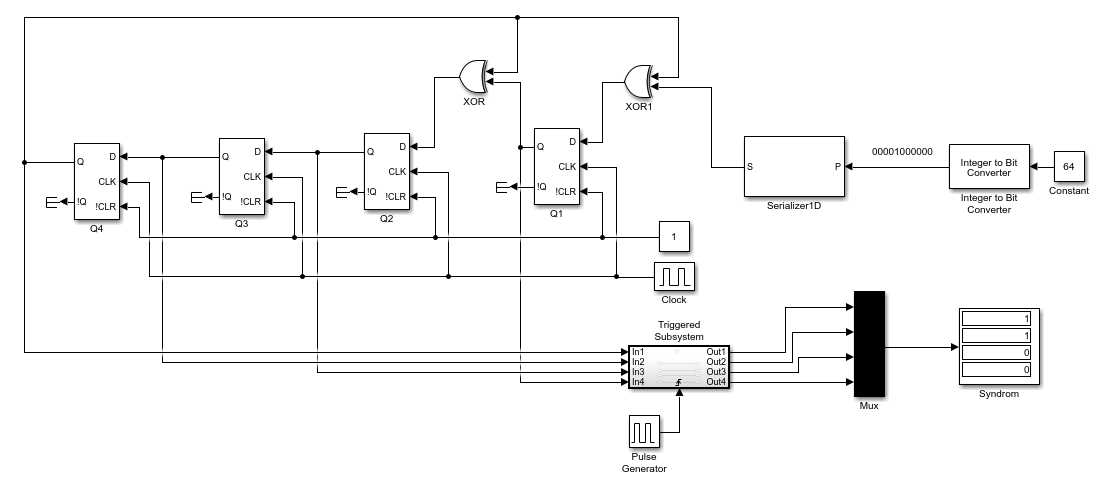
Więc, by móc wykryć błąd potrzebujemy stworzyć tabelę z syndromami. Gdy podzielimy wielomian błędu E(X) przez nasz dzielnik P(X) otrzymamy syndrom dla danego błędu (str.85 z prezentacji do przedmiotu). Można podzielić ręcznie, lub użyć do tego LFSR.



Rysunek 8 Ręczne obliczanie syndromu dla E(X) =

Obliczony syndrom wynosi , czyli binarnie 1100.

By obliczyć syndrom przy użyciu LFSR, wystarczy zaadaptować poprzedni układ.



Rysunek 9 Schemat użyty do obliczania syndromów

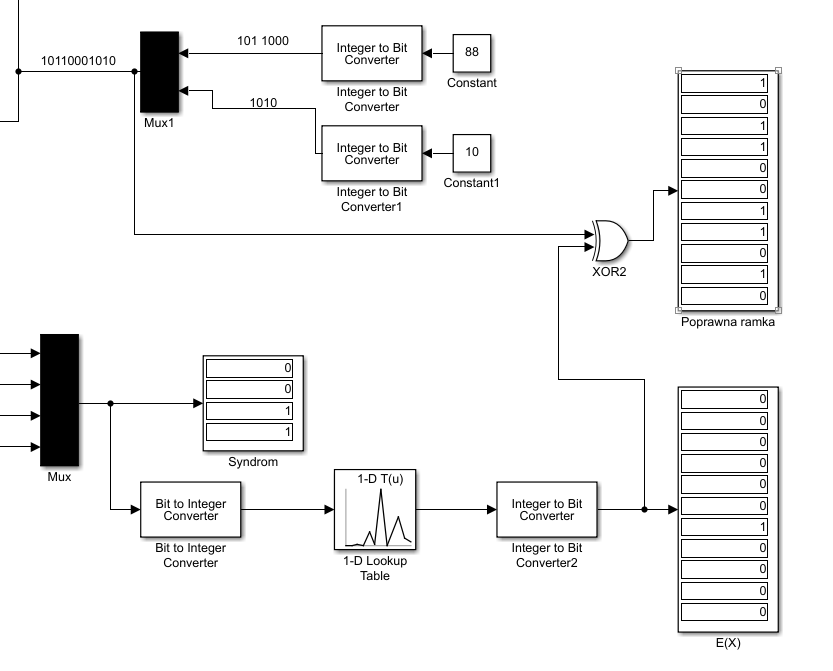
Po wprowadzeniu kodu błędu uzyskujemy odpowiedni syndrom, dla kodu błędu E(X) = 00001000000, syndrom wynosi S(X) = 1100.

Po obliczeniu wszystkich syndromów tabela dla naszego przypadku wygląda następująco:

|  |  |
| --- | --- |
| **Error Pattern E(X)** | **Syndrome S(X)** |
| 00000000001 |  |
| 00000000010 |  |
| 00000000100 |  |
| 00000001000 |  |
| 00000010000 |  |
| 00000100000 |  |
| 00001000000 |  |
| 00010000000 |  |
| 00100000000 |  |
| 01000000000 |  |
| 10000000000 |  |

Gdy już mamy tabelę syndromów, możemy korygować pojedyncze błędy które pojawią się podczas transmisji ramki.

Wprowadzono błąd przy transmisji ramki. Zamiast T = 10110011010, odbieramy T = 10110001010.



Rysunek 10 Korekcja błędu przy użyciu tabeli syndromów

Podczas transmisji ramki pojawia nam się błąd – syndrom wynosi 0011. Przy pomocy Look-up Table przypisujemy odpowiedni kod błędu do danego syndromu (0011 à00000010000), a następnie poprawiamy błędną ramkę przy pomocy funkcji XOR.

Ostatecznie nasza odebrana ramka T = 10110011010. Widzimy, że błąd który pojawił się przy transmisji został poprawiony.

**Jakie błędy możemy wykryć?**

* Wszystkie błędy jednobitowe, jeśli P(X) ma więcej niż jeden człon niezerowy
* Wszystkie błędy dwubitowe, o ile P(X) ma współczynnik z co najmniej 3 członami
* Wszystkie nieparzyste błędy, o ile P(X) zawiera X + 1 jako czynnik

Prawdopodobieństwo wykrycia błędu serii zależy od liczby bitów kontrolnych (r = n-k) użytych do skonstruowania kodu cyklicznego.

W przypadku błędu serii (burst error) o długości prawdopodobieństwo wykrycia błędu wynosi 100%.

W przypadku błędu serii o długości równej prawdopodobieństwo wykrycia błędu zmniejsza się do

.

Błąd serii o długości większej niż , prawdopodobieństwo wykrycia błędu wynosi .

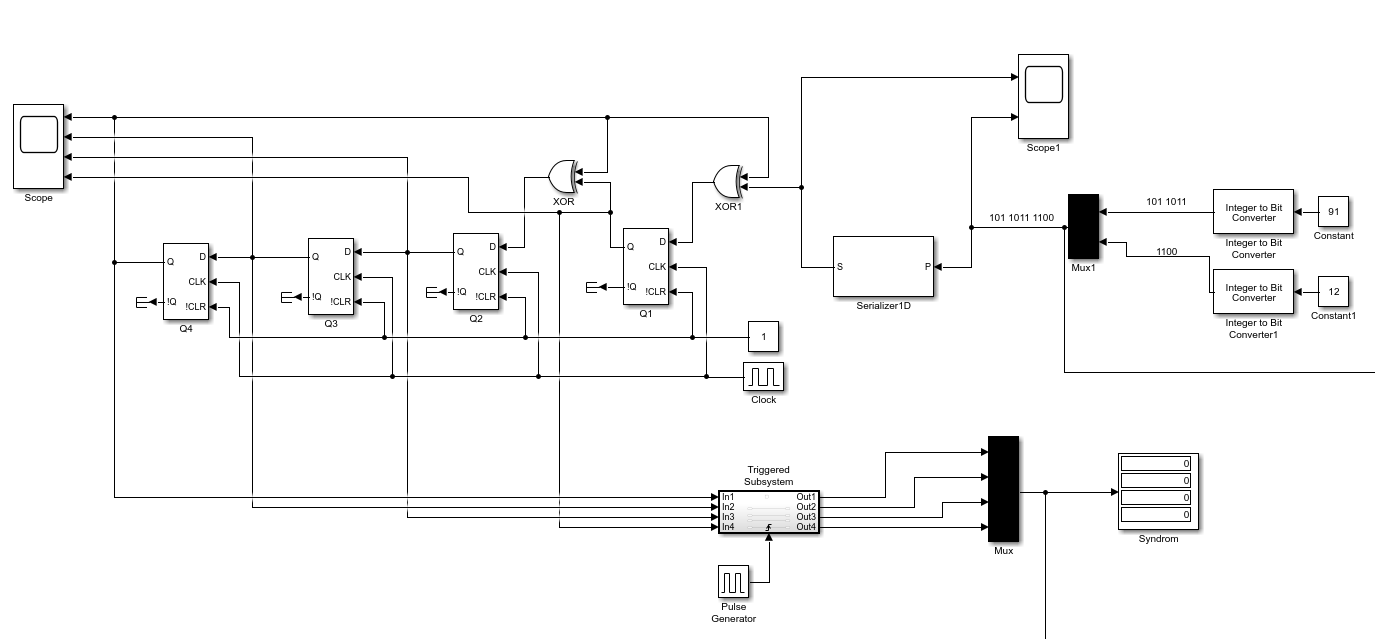
**Kiedy na pewno nie wykryjemy błędu?**

Kiedy wektor błędu E(X) będzie wielokrotnością wielomianu P(X).

Przykład:

Wielomian .

Jego wielokrotnością jest np. wielomian . Wprowadźmy ten błąd przy transmisji ramki. Zamiast T = 10110011010, odbieramy T = 10110111100.



Rysunek 11 Układ nie wykrywa błędu

**Jakie błędy możemy poprawić?**

* Żeby być w stanie poprawić wszystkie pojedyncze błędy (single-bit errors) w kodzie musimy spełnić założenie:

W naszym przypadku:

Jesteśmy w stanie poprawić wszystkie pojedyncze błędy.

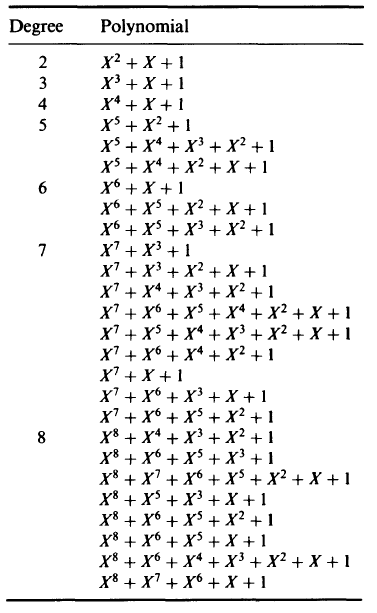
* Żeby być w stanie poprawić wszystkie pojedyncze i podwójne błędy (single- and double-bit errors) w kodzie musimy spełnić założenie:

W naszym przypadku:

Nie jesteśmy w stanie poprawić podwójnych błędów.

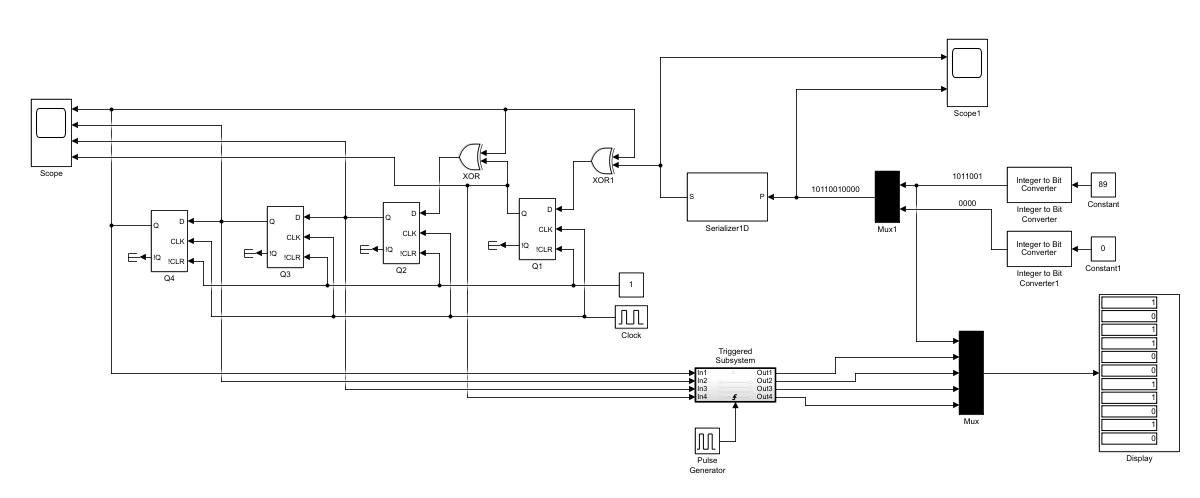
1. **Dodatki:**

Przy implementacji CRC na LFSR używamy wielomianów pierwotnych. Wielomianem pierwotnym nazywa się wielomian nierozkładalny P(x) stopnia , którego okres wynosi . Wielomianem nierozkładalnym nazywa się taki wielomian P(x), który nie daje się przedstawić jako iloczyn dwóch wielomianów.

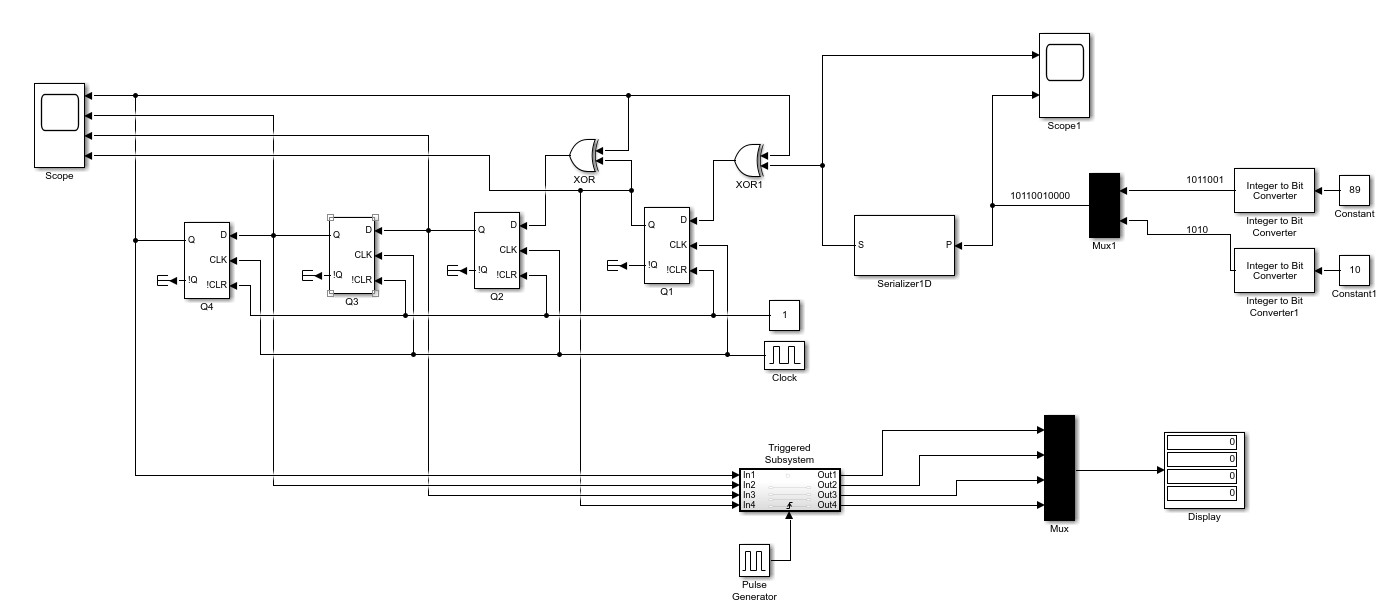
Wielomiany pierwotne:

**Zadania**

Podstawowy układ enkodera CRC na 4- bitowym LFSR (Rys. 1)



Rys. 1 Enkoder CRC



Rys. 2 Dekoder CRC

Parametry elementów układu:

* Clock à Period = 1
* Integer to Bit Converter à Number of bits per integer = 7
* Integer to Bit Converter1 à Number of bits per integer = 4
* Pulse Generator à Phase delay =11.5
* Serializer1D à Main: Ratio = 11
* Serializer1D à Signal Attributes: Input sample time = 11
* Stop time = 12

Przykład:

Wiadomość: D = 1011001 (7 bity)

Dzielnik: P = 10011 (5 bity)

Reszta z dzielenia: R = do obliczenia (4 bity)

Ramka do transmisji: T = 1011001XXXX

Zadania - Enkoder i dekoder CRC, korekcja błędów

1. Inna wiadomość\*: 0111001 0101001 1100001 1100101   
   Dla danego układu wyznaczyć analitycznie resztę z dzielenia, potwierdzić wyniki symulacją.   
   Zweryfikować poprawność transmisji analitycznie i symulacyjnie (dekoder).
2. Inna długość wiadomości\*: 10111001 10101001 11100001 11100101  
   Zmodyfikować układ do obsługi wiadomości o długości 8 bitów, podać parametry elementów układu. Wyznaczyć analitycznie resztę z dzielenia, potwierdzić wyniki symulacją.   
   Zweryfikować poprawność transmisji analitycznie i symulacyjnie (dekoder).  
   Czy jesteśmy w stanie poprawić wszystkie pojedyncze błędy? A podwójne?
3. W układzie dekodera (Rys. 2) otrzymano następujące ramki\*: 81, 121, 73, 91. W każdym przypadku suma kontrolna wynosi 10. Czy otrzymana wiadomość jest poprawna? Na którym bicie nastąpiło przekłamanie, wykonać korekcję i odtworzyć wysłaną wiadomość.
4. Inne niezerowe współczynniki wielomianu dzielnika\*: (x^4+x^2+1, x^4+1, x^4+x^3+1, x^4+x^3+x^2+1)  
   Zmodyfikować układ (lokalizacja, liczba bramek XOR), podać parametry elementów układu. Wyznaczyć resztę z dzielenia analitycznie (wiadomości: 0111001 0101001 1100001 1100101), potwierdzić wyniki symulacją. Stworzyć tabelę syndromów. Porównać z tabelą syndromów dla wielomianu x^4+x+1 (wnioski).
5. W układzie odpowiadającego wielomianowi dzielnika x^4+x^2+1. Otrzymano następujące ramki: 93, 92. W każdym przypadku suma kontrolna wynosi 10. Czy otrzymana wiadomość jest poprawna? Na którym bicie nastąpiło przekłamanie. Czy można wykonać korekcję i odtworzyć wysłaną wiadomość?
6. Inny wielomian dzielnika (x^5+x^2+1).  
   Zmodyfikować układ (dodatkowy rejestr), podać parametry elementów układu. Zweryfikować poprawność transmisji analitycznie i symulacyjnie (dekoder). Zbadać czy jednobitowe przekłamania mogą zostać poprawione przy użyciu tego układu? Dla przykładowych ramek wyznaczyć resztę z dzielenia analitycznie, potwierdzić wyniki symulacją.

\* wybrać jedną przykładową (nr z listy mod 4)