Sprawozdanie liczniki i automaty

Technika Cyfrowa 2021/22

Dominik Grzesik, Sebastian Kozak, Szymon Słota, Marcin Mikuła, Jakub Łubkowski

mmikula@student.agh.edu.pl szymonslota@student.agh.edu.pl lubkowski@student.agh.edu.pl
domgrze@student.agh.edu.pl sebkozak@student.agh.edu.pl

Spis treści

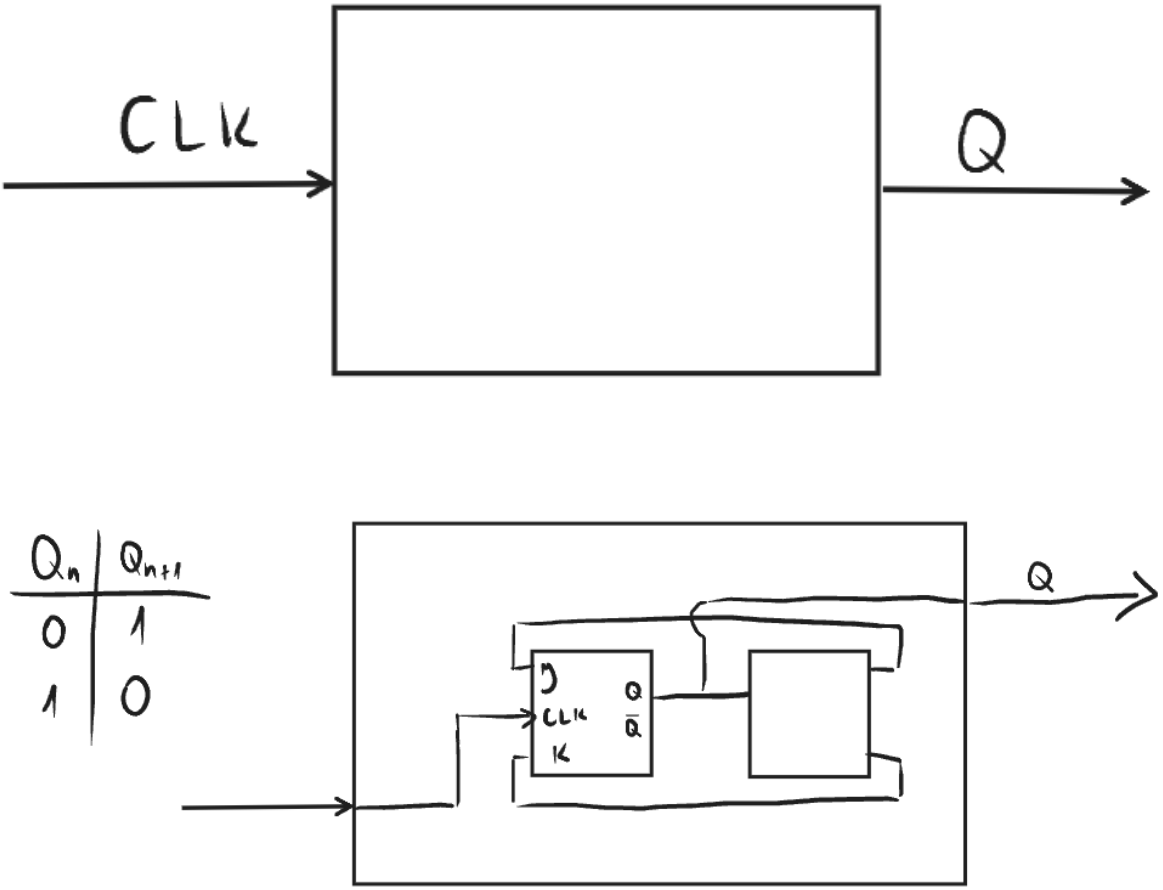
1. Zadanie 3 a) 3
1.1. Wprowadzenie 3
1.2. Wyprowadzenie 5
1.3 Implementacja układów w multisimie 9
1.8 Wnioski 16
1.9 Przykładowe zastosowania 16
2. Zadanie 3 b) 18
2.1. Wprowadzenie 18
2.2 Wyprowadzenie 19
2.5 Implementacja układu - czterobitowy licznik liczący w kodzie Gray'a. 19
2.6 Układ testujący utworzony synchroniczny licznik liczący w kodzie Gray'a. 20
2.7 Generator słów 21
2.8 Układ testujący w programie multisim 22
2.9 Analizator słów 23
2.10 Wnioski 24
2.11 Przykładowe zastosowania 25

\

1. Zadanie 3 a)

Zaprojektować, zrealizować i przetestować dwójkę liczącą w oparciu o przerzutnik "JK". Następnie w oparciu o jeden z wariantów zaprojektowanej dwójki liczącej, zaproponować i przetestować asynchroniczny licznik modulo 8.

1.1. Wprowadzenie



Rysunek 1. Poglądowy rysunek przerzutnika RS.

Właściwości bramki XOR:

x	y	XOR
0	0	0
1	0	1
0	1	1
1	1	0

Tabela 1. Tablica bramki XOR.

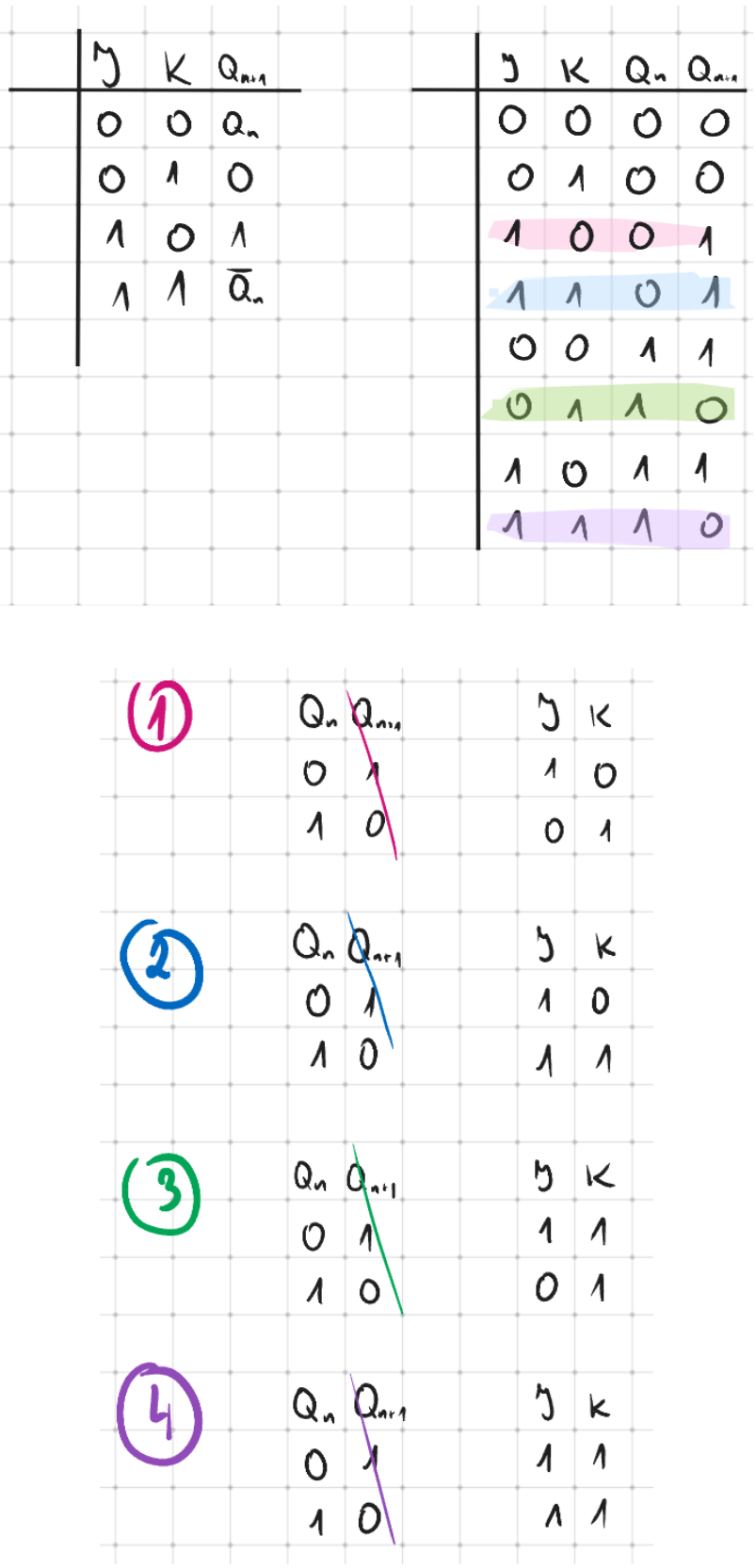
1.2. Wyprowadzenie

Właściwości przerzutnika JK:

J	K	Q(t)	Q(t+1)	Opis
0	0	0 lub 1	0 lub 1	podtrzymanie
0	1	0 lub 1	0	przejście na 0
1	0	0 lub 1	1	przejście na 1
1	1	0 lub 1	1 lub 0	~Q

Tabela 2. Tabela wzbudzeń przerzutnika JK.

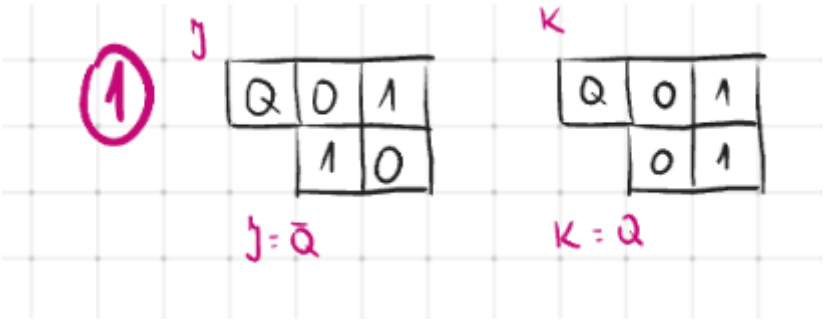
Analizując powyższą tabelę, możemy zauważyć, iż podanie jedynki logicznej na wejście J powoduje ustawienie przerzutnika (co skutkuje pojawieniem się stanu wysokiego na wyjściu Q). Ustawienie wejścia K w stan wysoki przestawia przerzutnik w stan niski. Jeżeli jedynka logiczna zostanie ustawiona na obydwu wejściach (J i K) to nastąpi zmiana stanu przerzutnika na przeciwny (czyli jeżeli układ był w stanie wysokim to przejdzie w stan niski i odwrotnie).



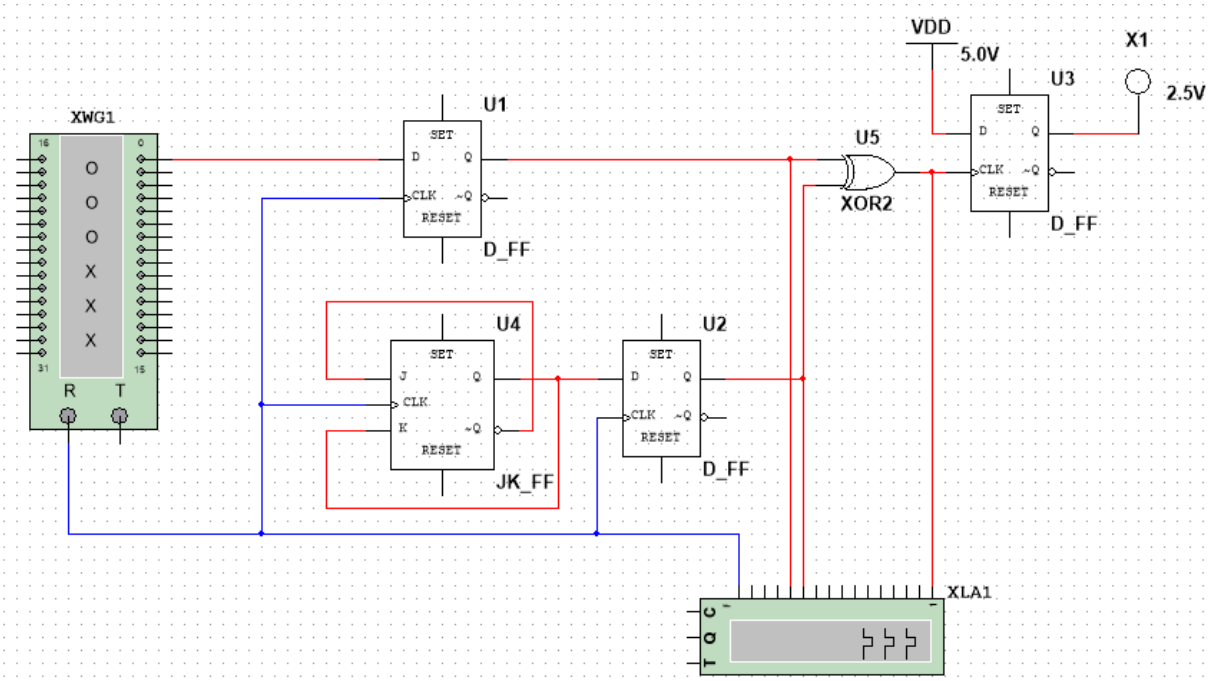
Rysunek 3. Tabela Karnaugh dla Q, oraz wyprowadzenie wzoru na Q.

mmikula@student.agh.edu.pl szymonslota@student.agh.edu.pl lubkowski@student.agh.edu.pl
domgrze@student.agh.edu.pl sebkozak@student.agh.edu.pl

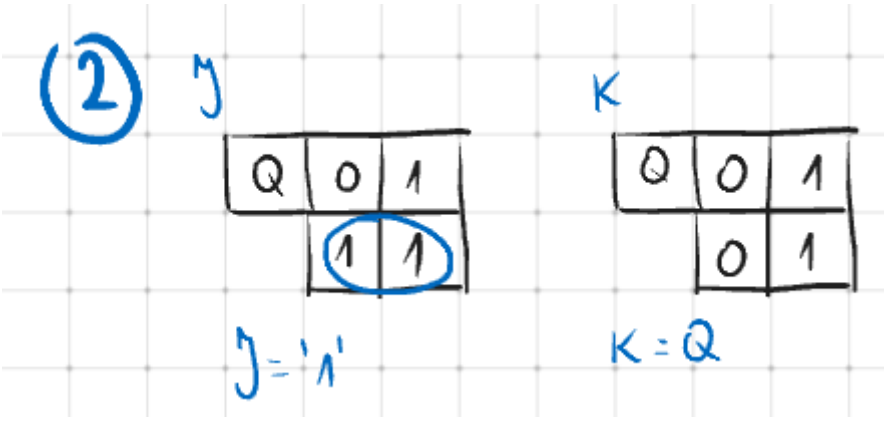
1.3 Tabele prawdy i implementacja układów w multisimie



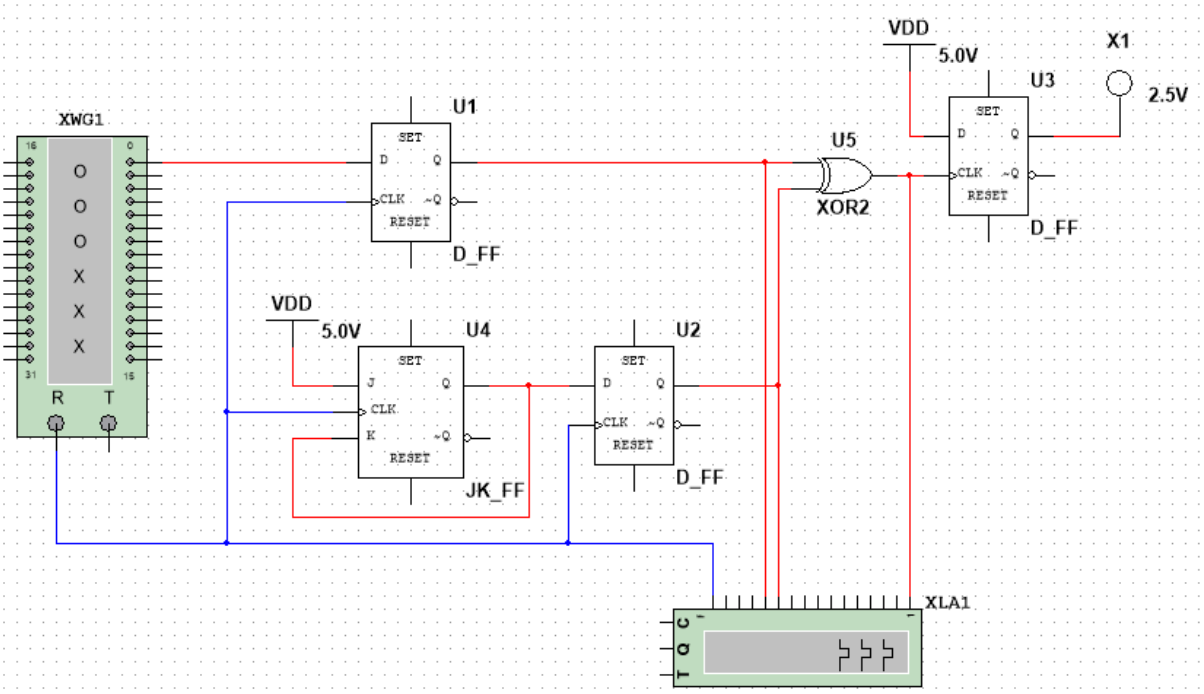
Rysunek 4. Wyprowadzenie wzoru dla poniższej implementacji.



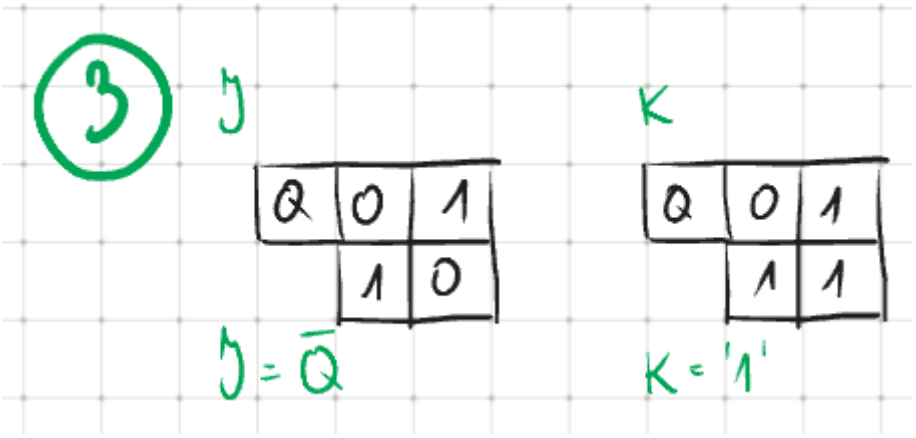
Rysunek 5. Implementacja dwójek liczących w programie Multisim (1).



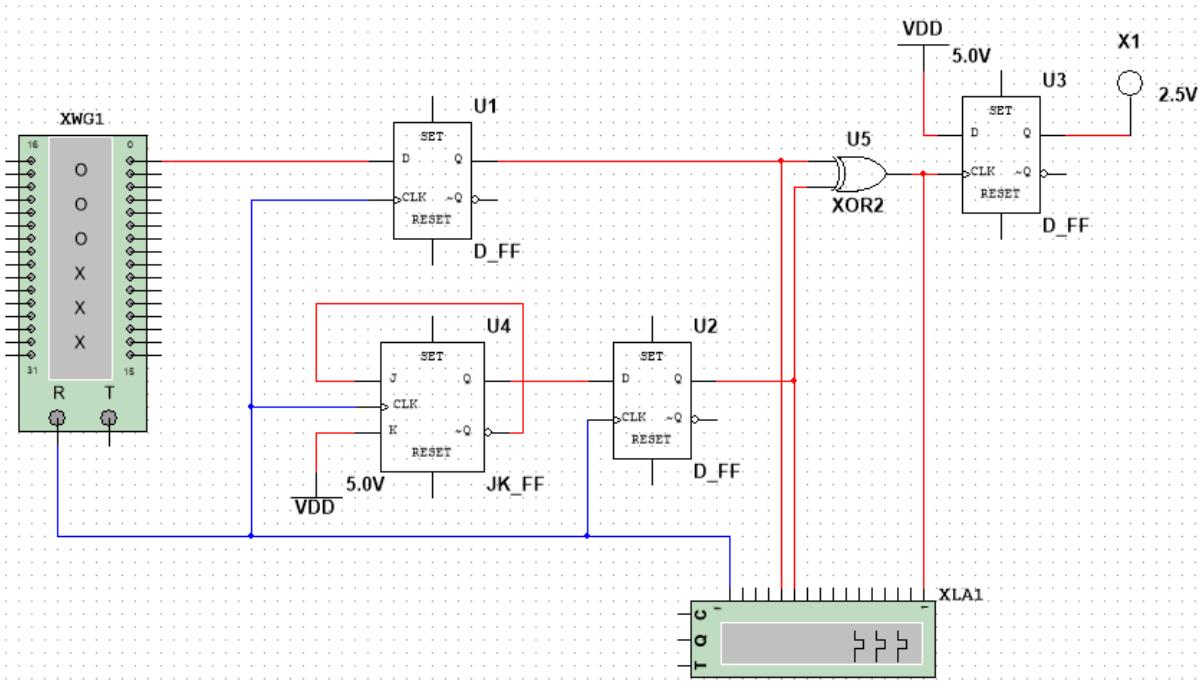
Rysunek 6. Wyprowadzenie wzoru dla poniższej implementacji.



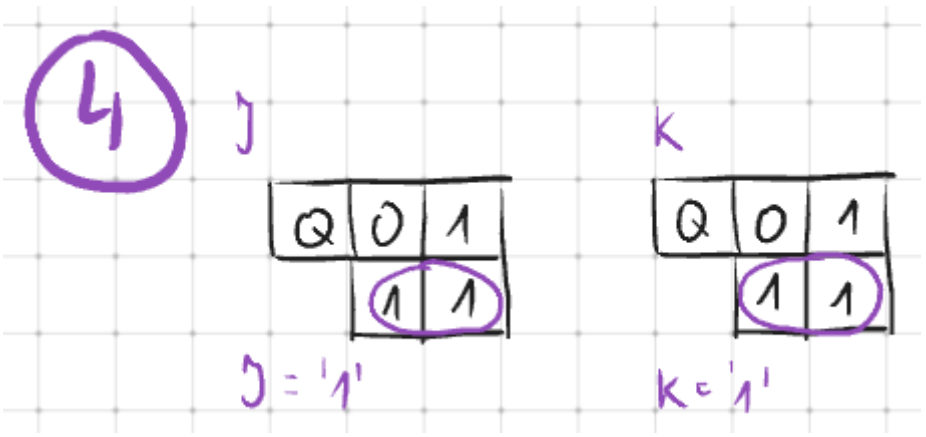
Rysunek 7. Implementacja dwójek liczących w programie Multisim (2).



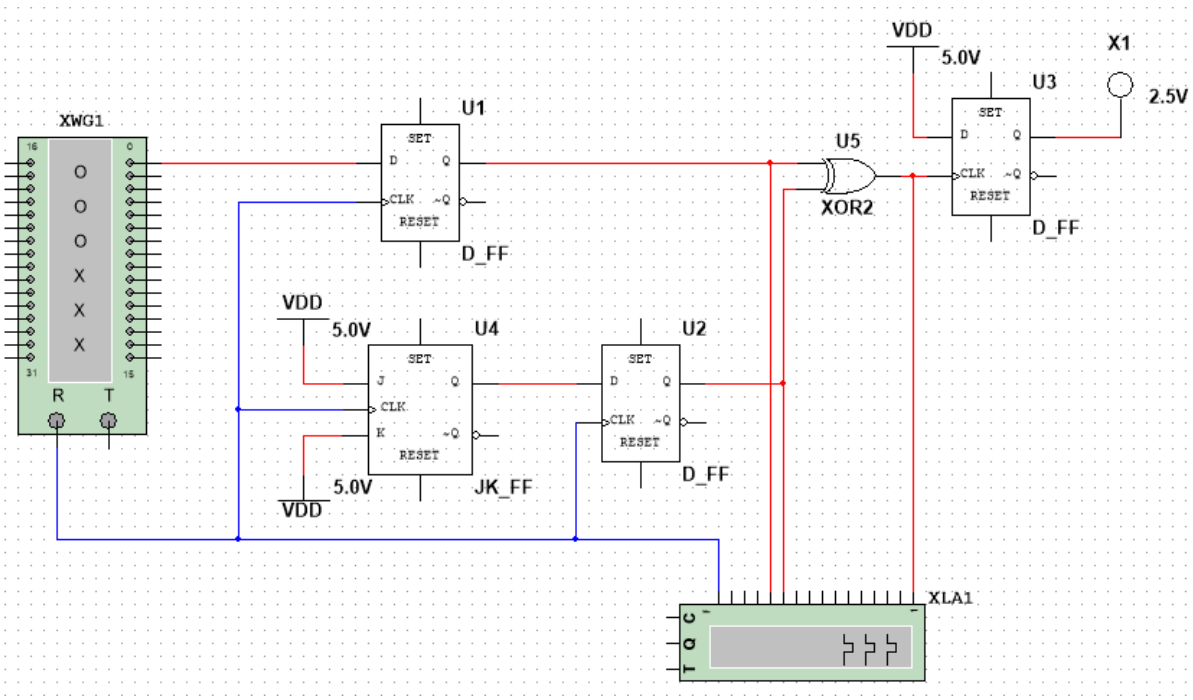
Rysunek 8. Wyprowadzenie wzoru dla poniższej implementacji.



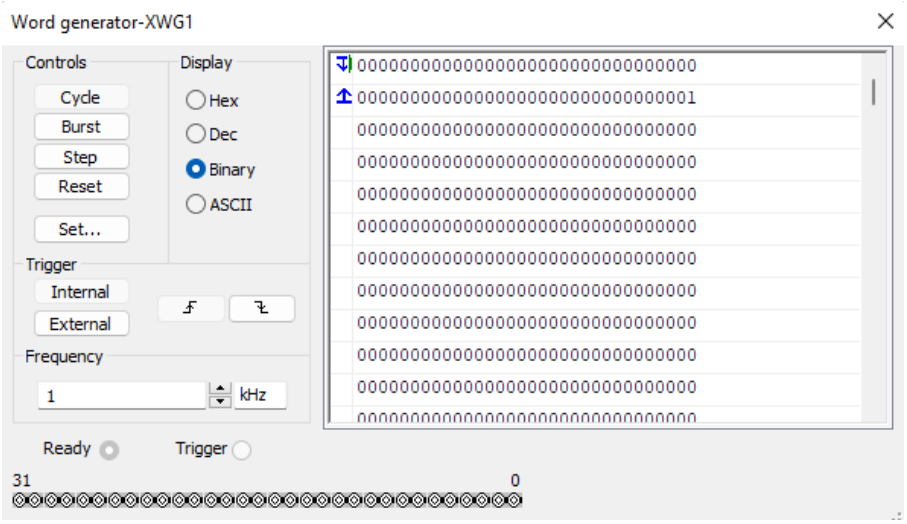
Rysunek 9. Implementacja dwójek liczących w programie Multisim (3).



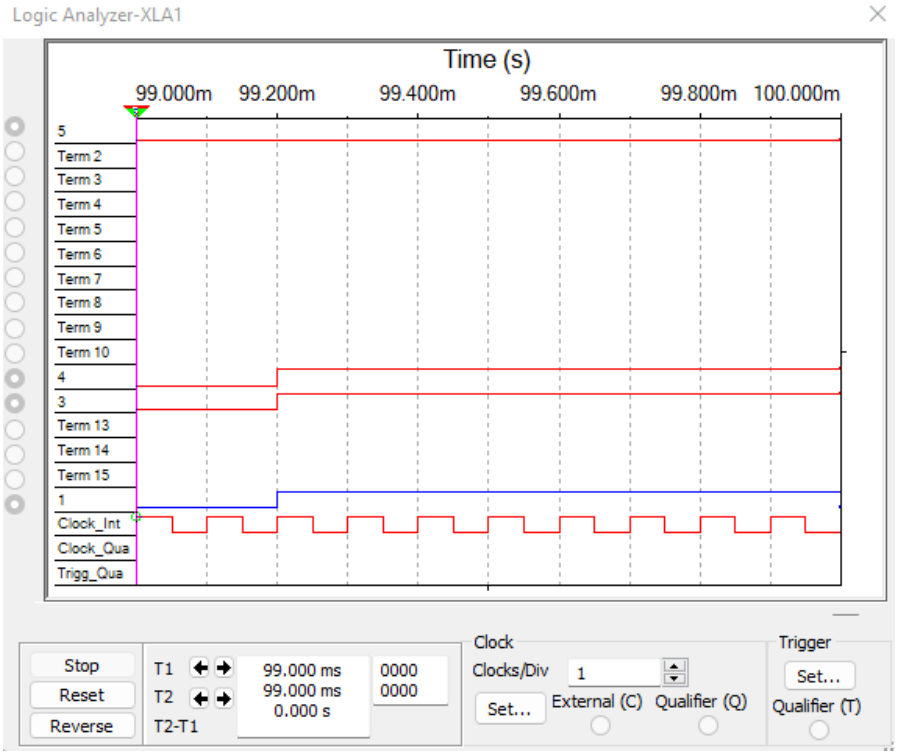
Rysunek 10. Wyprowadzenie wzoru dla poniższej implementacji.



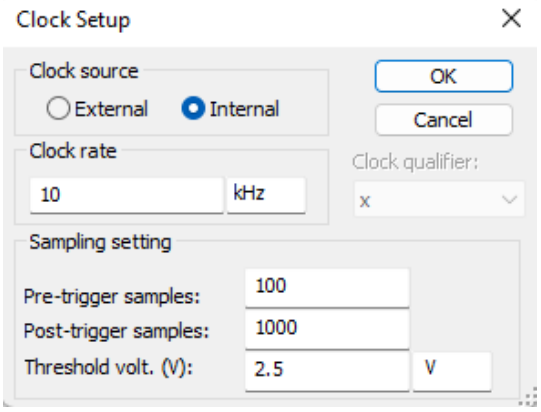
Rysunek 11. Implementacja dwójek liczących w programie Multisim (4).



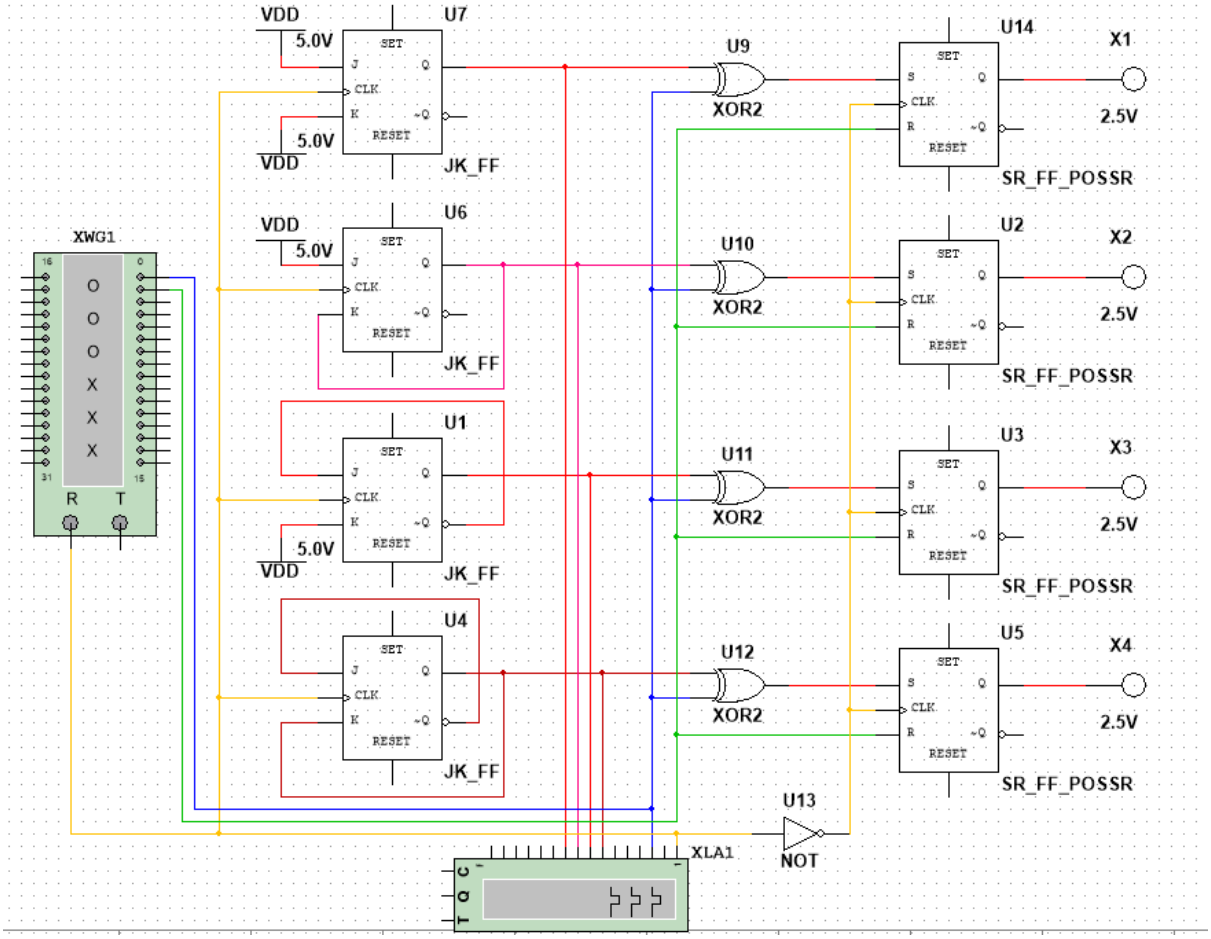
Rysunek 12. Ustawienia generatora słów dla dwójek liczących.



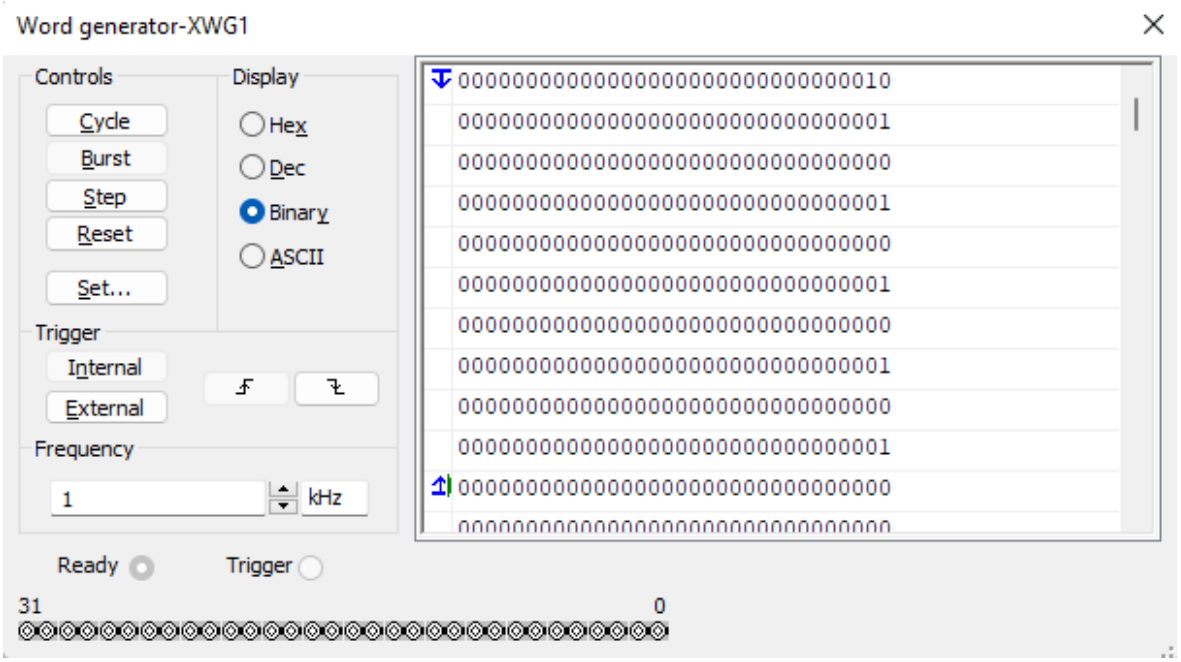
Rysunek 13. Analizator słów dla dwójek liczących.



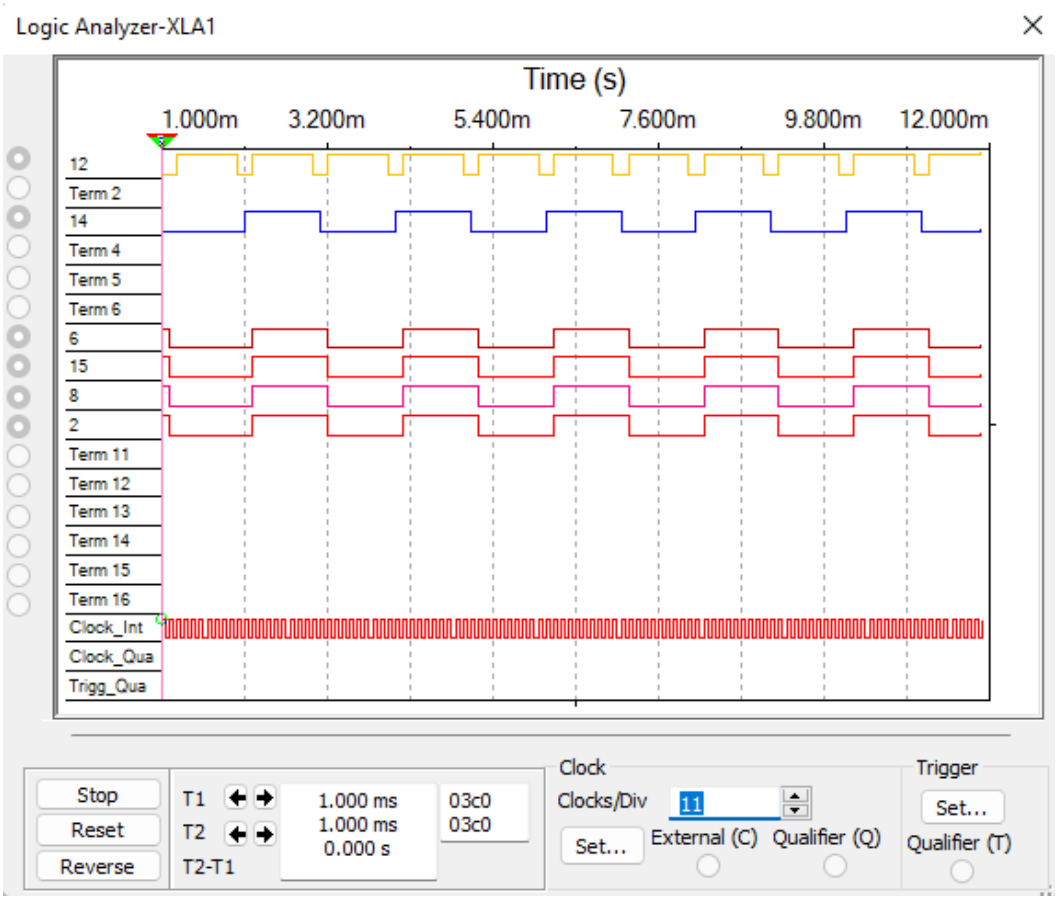
Rysunek 14. Ustawienia analizatora słów dla dwójek liczących.



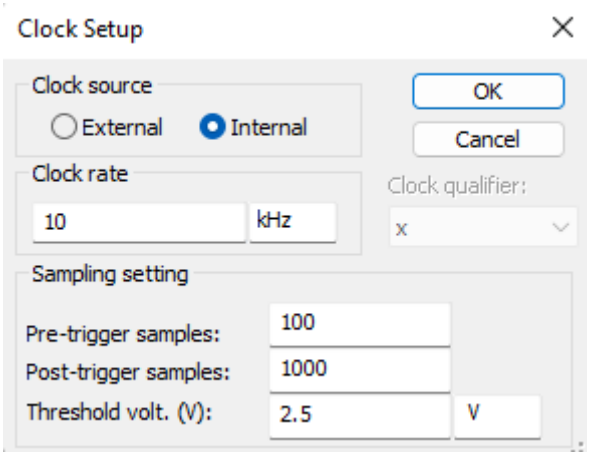
Rysunek 15. Zaprojektowany układ testowy w programie Multisim.



Rysunek 16. Generator słów dla układu testowego.

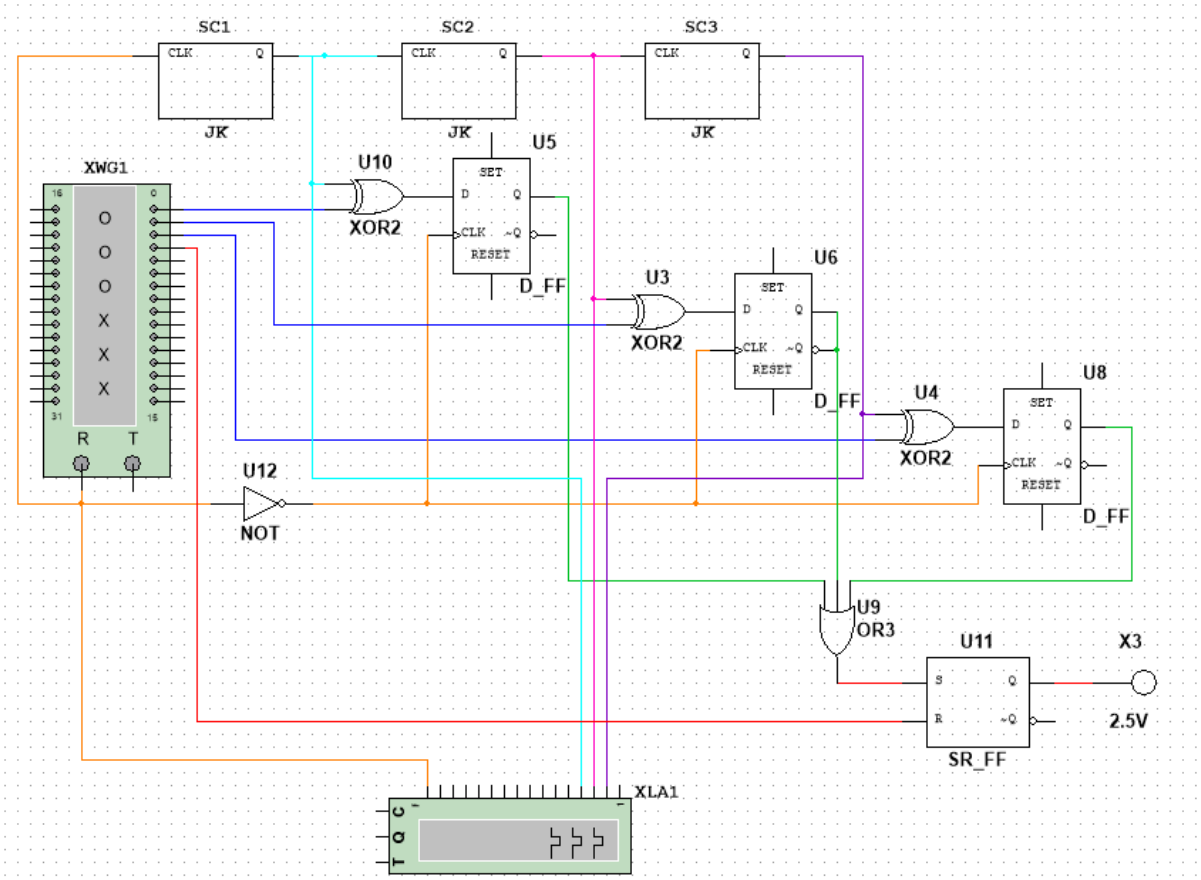


Rysunek 17. Analizator słów dla układu testowego.

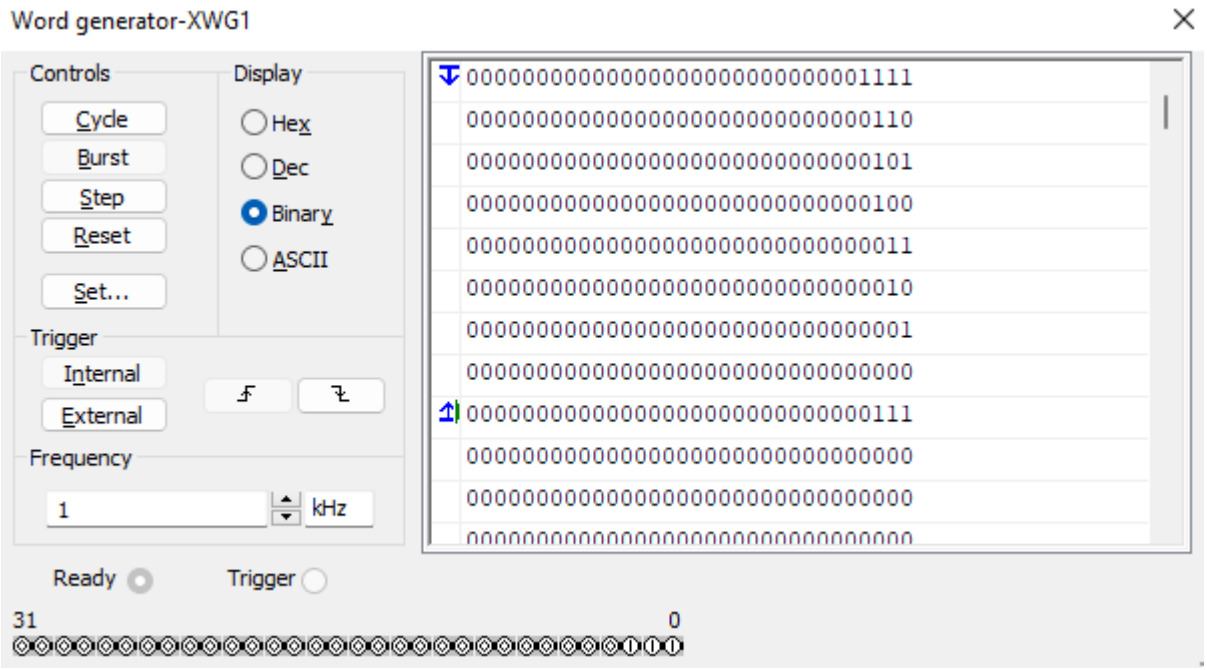


Rysunek 18. Ustawienia analizatora słów dla układu testowego.

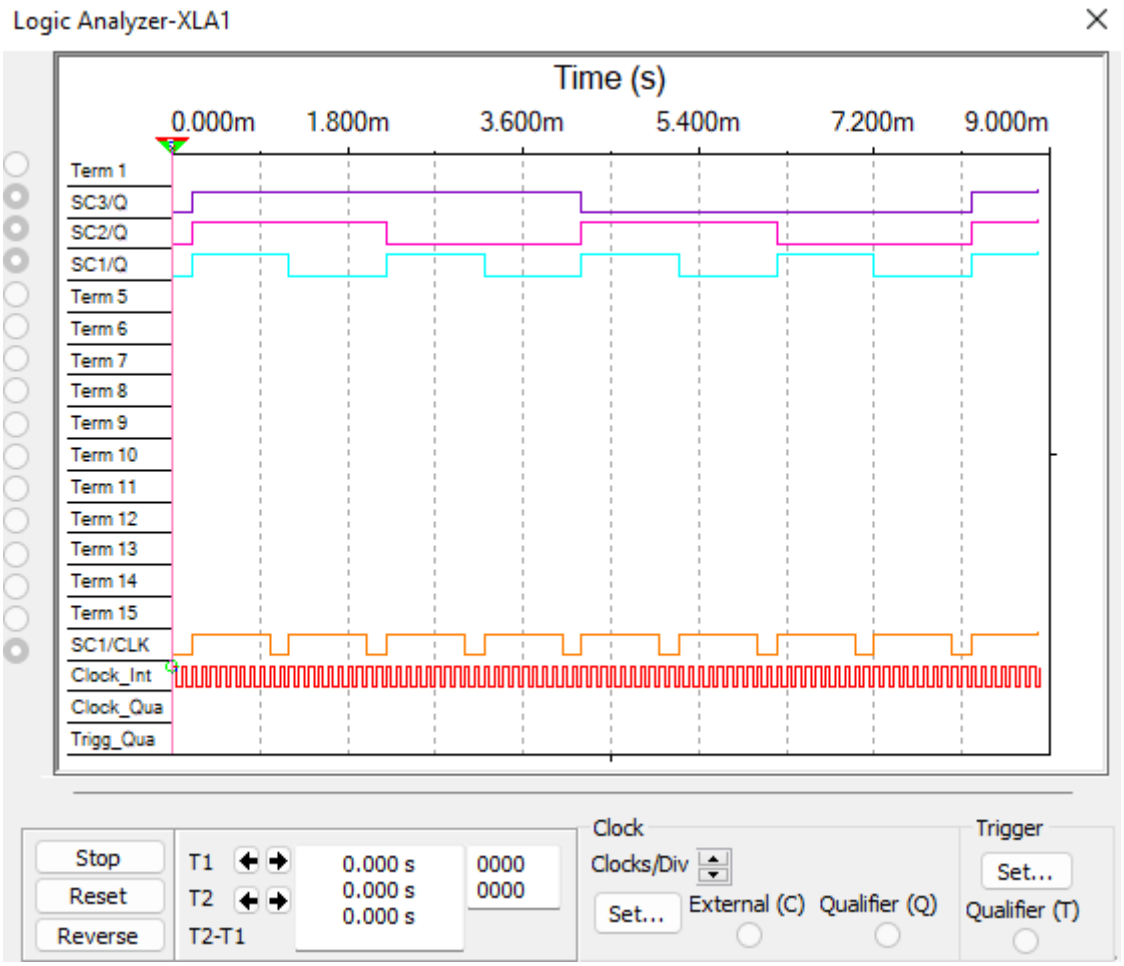
Propozycja asynchronicznego licznika modulo 8



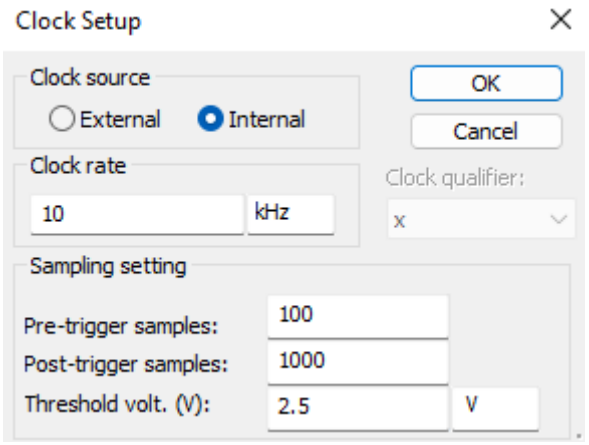
Rysunek 19. Asynchroniczny licznik modulo 8 z testami w programie Multisim.



Rysunek 20. Generator słów.



Rysunek 21. Analizator słów.



Rysunek 22. Ustawienia analizatora słów.

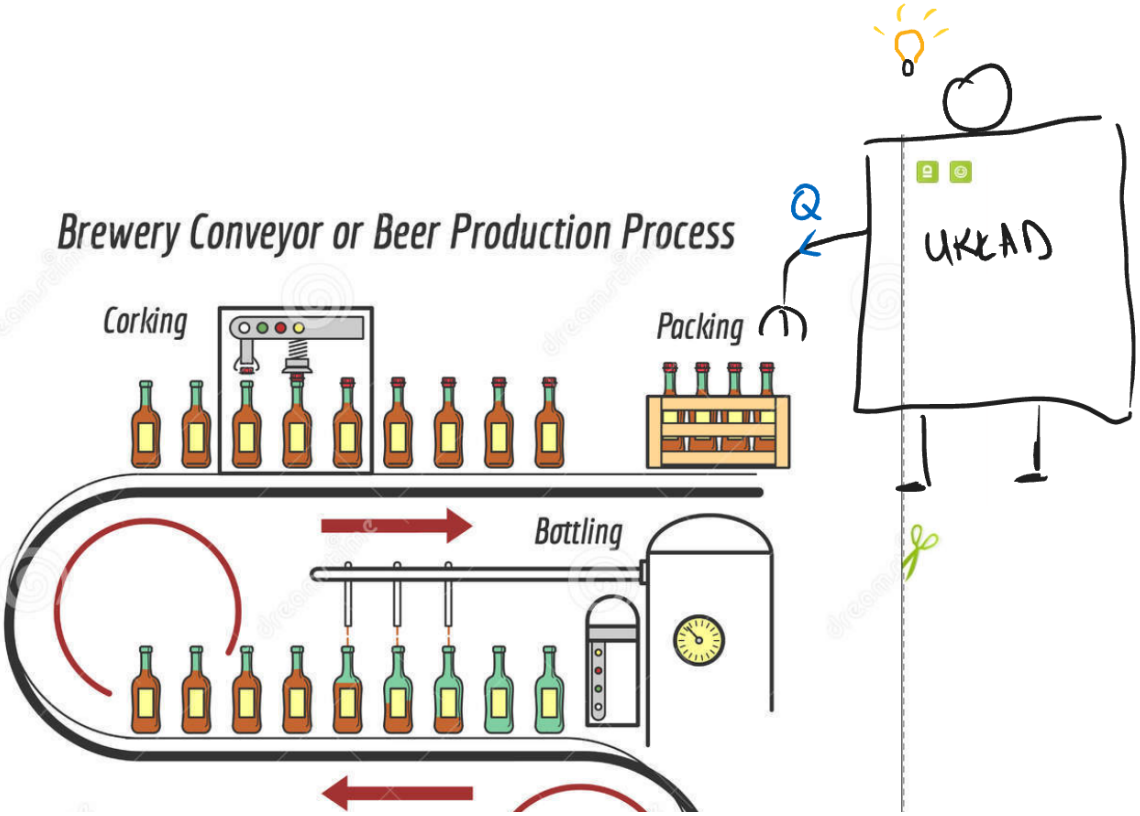
1.8 Wnioski

1. Na podstawie przedstawionych wyprowadzeń można stwierdzić że istnieją 4 różne sposoby realizacji dwójki liczącej na przerzutnikach JK.

2. Dla wyjść Q zaproponowany licznik modulo 8 odlicza w kolejności malejącej, a dla wyjść ~Q w kolejności rosnącej.

1.9 Przykładowe zastosowania

1. Taśma produkcyjna - używając licznika modulo 8 jesteśmy w stanie kontrolować przebieg produkcji 8-paków soku pomarańczowego.



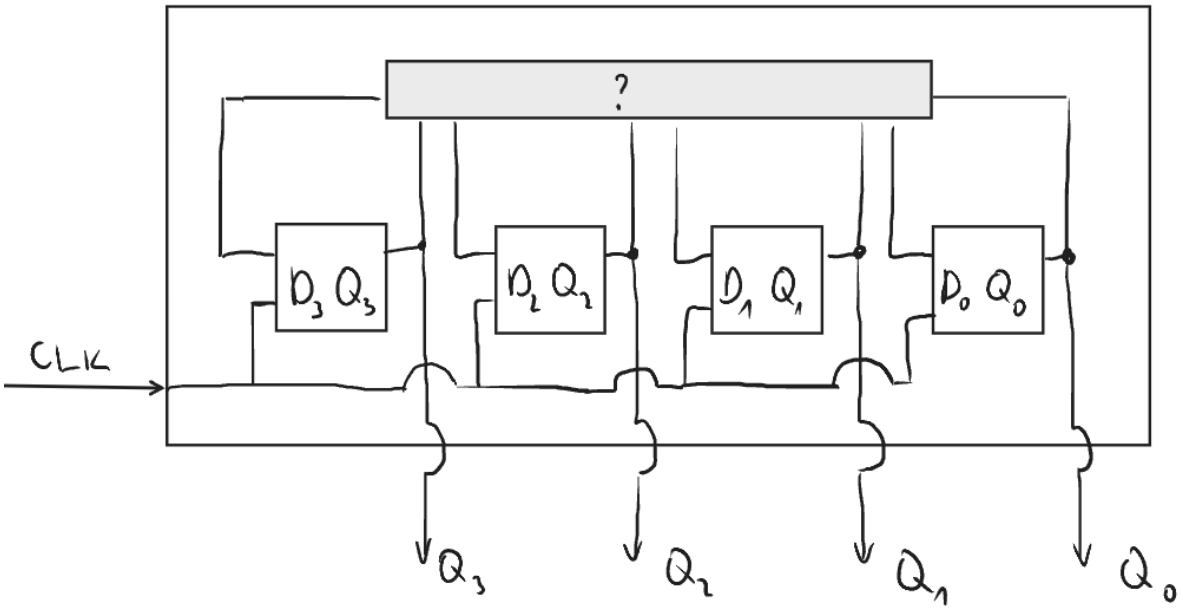
Rysunek 23.

2. Zadanie 3 b)

Bazując na dowolnie wybranych przerzutnikach, zaprojektować, zbudować i przetestować synchroniczny czterobitowy licznik liczący w kodzie Gray’a.

2.1. Wprowadzenie

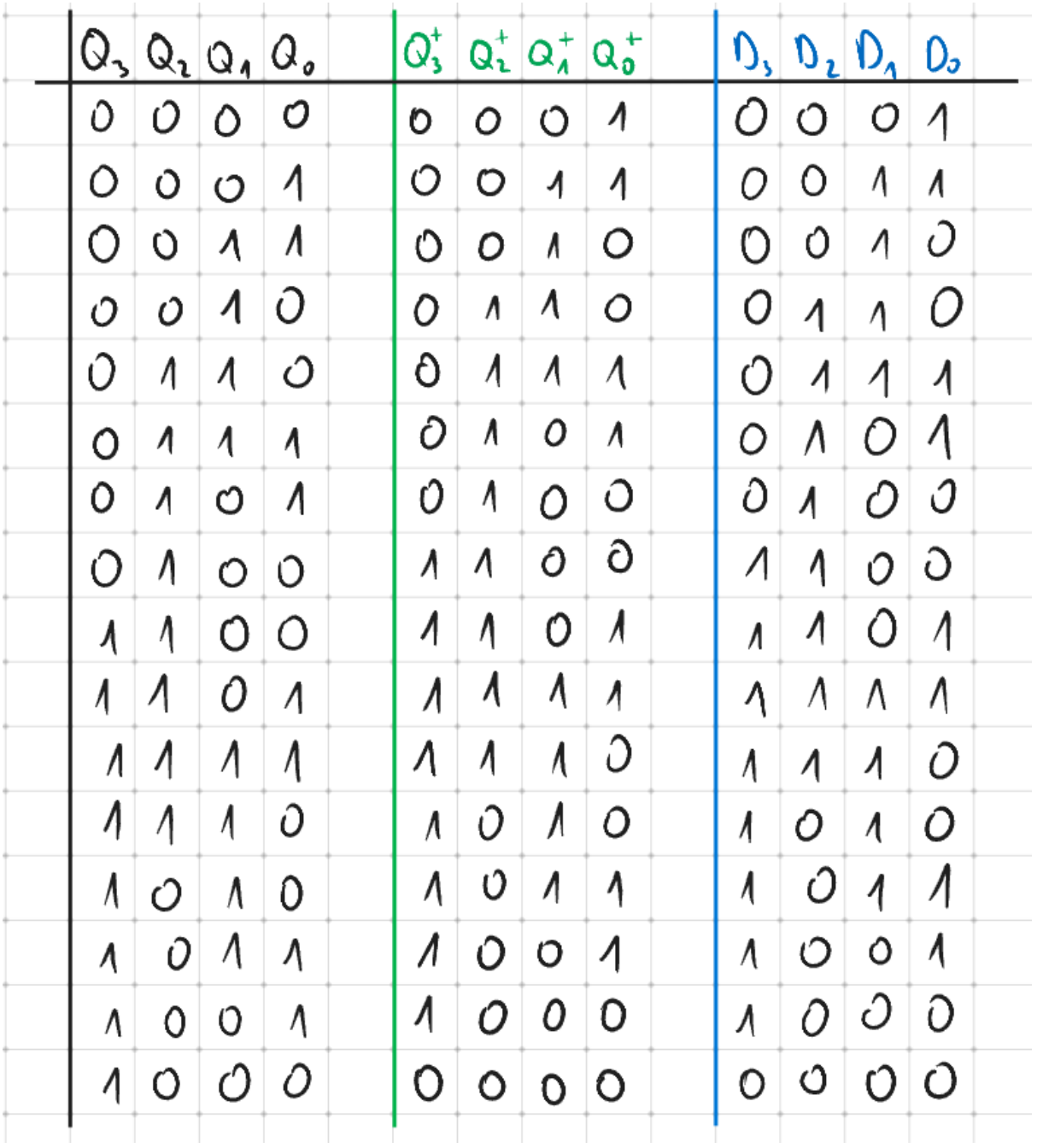
Kod Gray’a - dwójkowy kod bezwagowy niepozycyjny, który charakteryzuje się tym, że dwa kolejne słowa kodowe różnią się tylko stanem jednego bitu.



Rysunek 24. Poglądowy rysunek projektowanego licznika.

2.2 Wyprowadzenie

Do skonstruowania licznika użyliśmy przerzutników D. Poniżej ich działanie dla 4 bitów:



Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	1	0	0	1	1
0	0	1	1	0	0	1	0	0	0	1	0
0	0	1	0	0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	1	0	1	1	1
0	1	1	1	0	1	0	1	0	1	0	1
0	1	0	1	0	1	0	0	0	1	0	0
0	1	0	0	1	1	0	0	1	1	0	0
1	1	0	0	1	1	0	1	1	1	0	1
1	1	0	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	0
1	1	1	0	1	0	1	0	1	0	1	0
1	0	1	0	1	0	1	1	1	0	1	1
1	0	1	1	1	0	0	1	1	0	0	1
1	0	0	1	1	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0

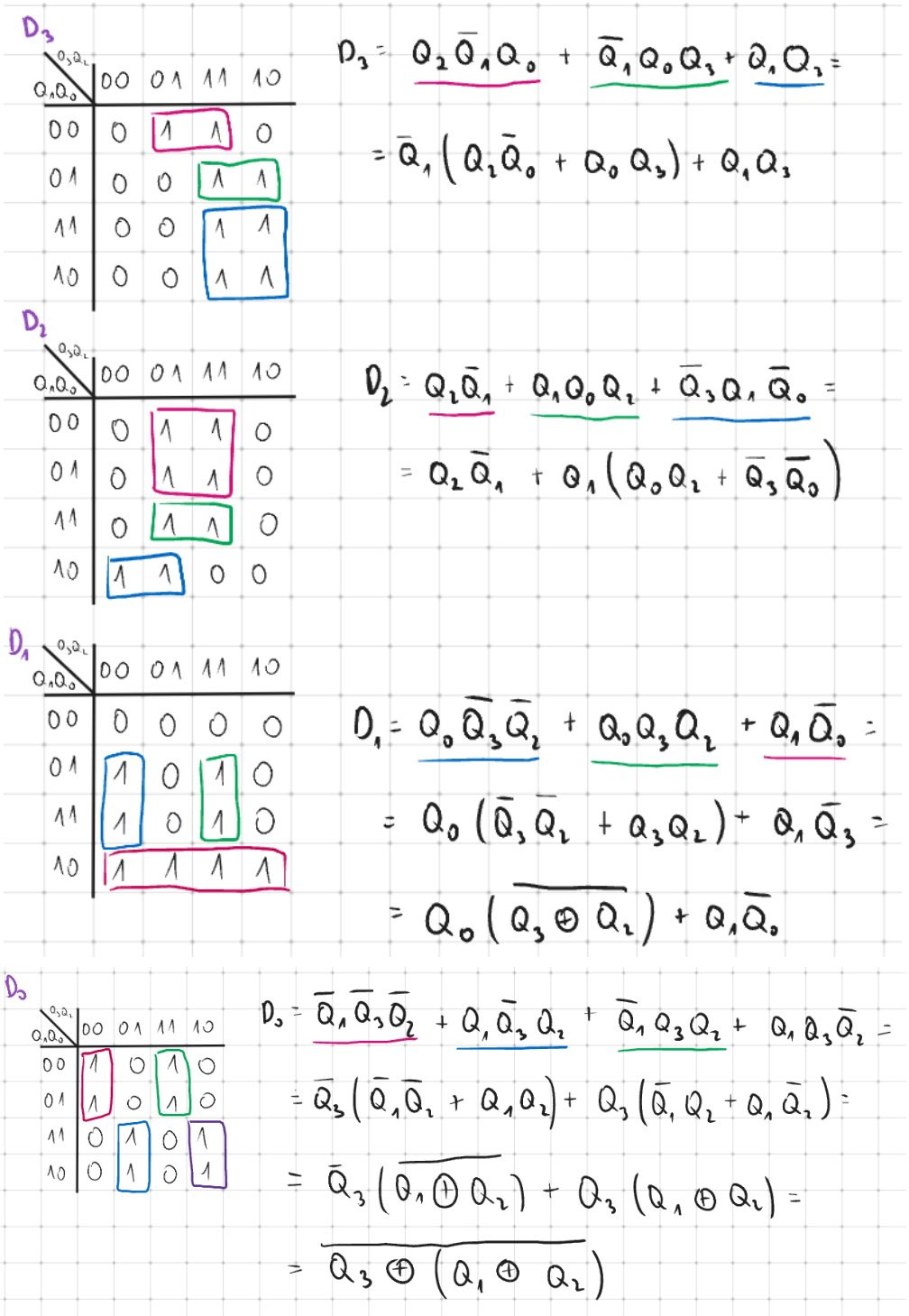
Rysunek 25. Działanie licznika liczącego w kodzie Gray'a.

$p \veebar q = (p \wedge \neg q) \vee (\neg p \wedge q)$, co jest równoznaczne z

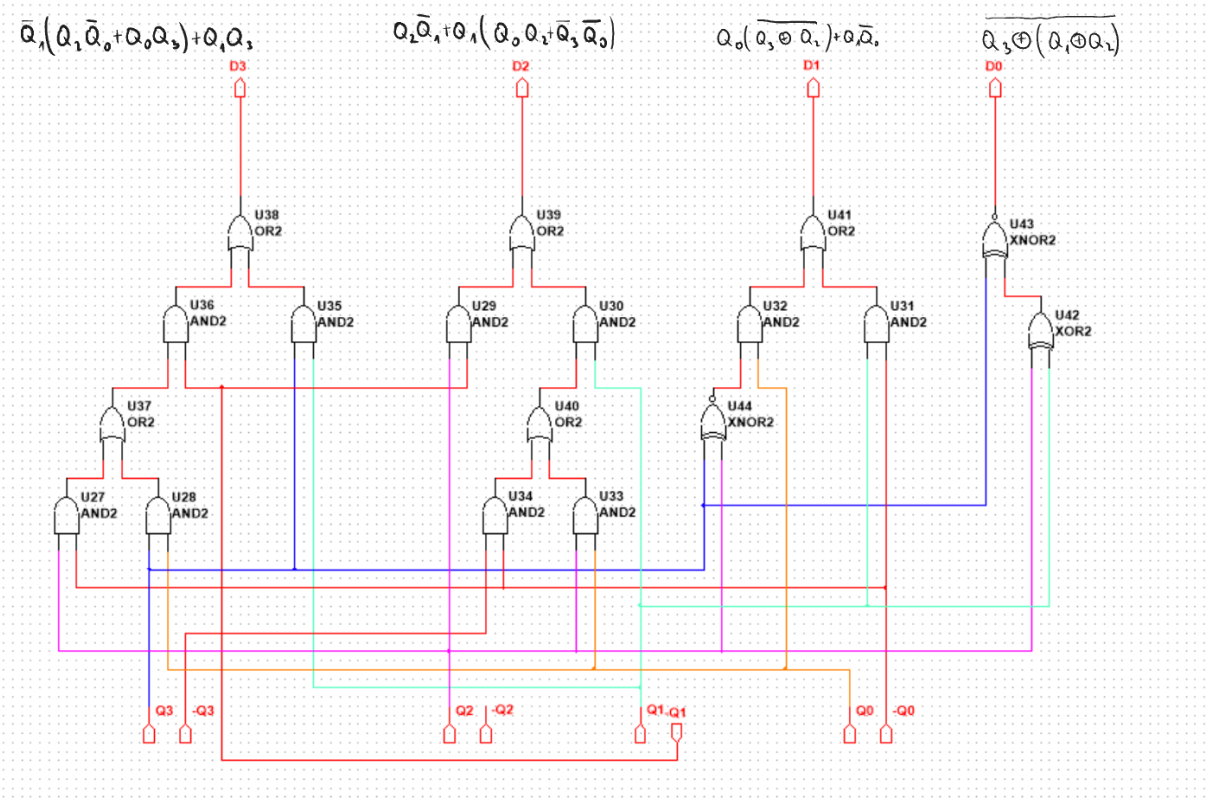
$p \veebar q = (p \vee q) \wedge \neg(p \wedge q).$

Rysunek 26. Funkcja logiczna XOR.

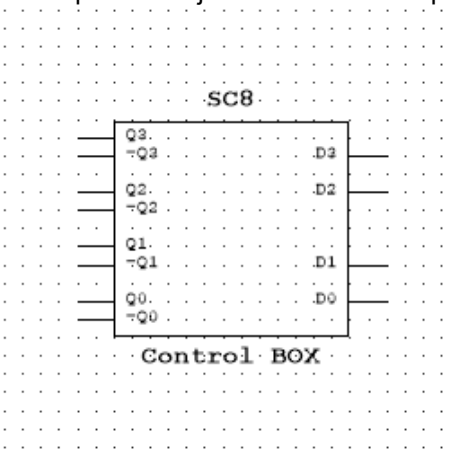
Wzory wejść dla poszczególnych przerzutników - działanie pudełka Control BOX



Rysunek 27. Wyprowadzenie wzorów dla przerzutników.

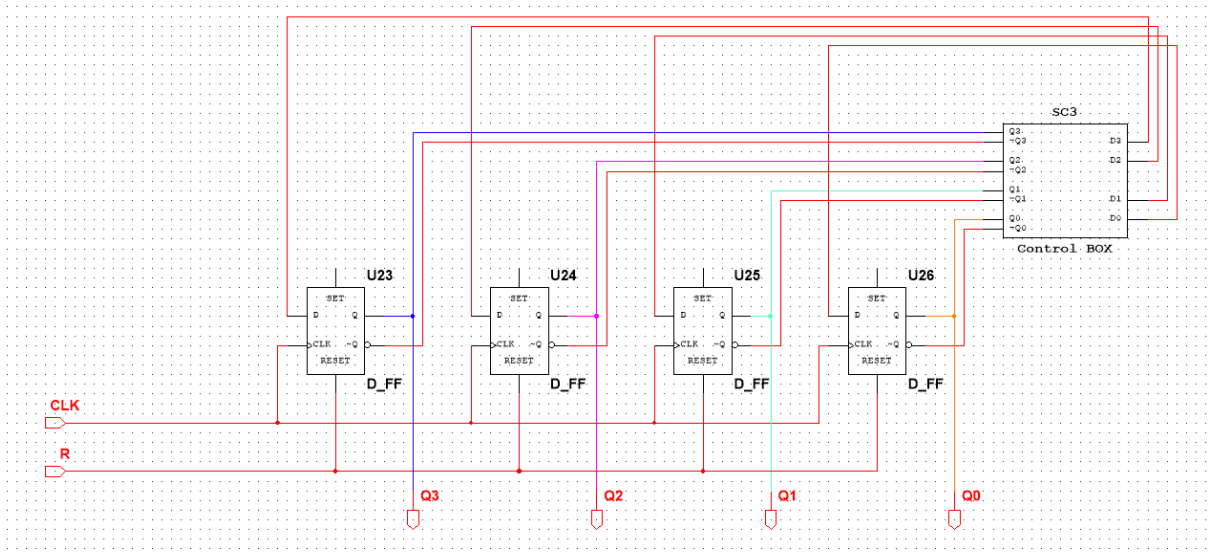


Rysunek 28. Implementacja Control BOX'a w programie Multisim.

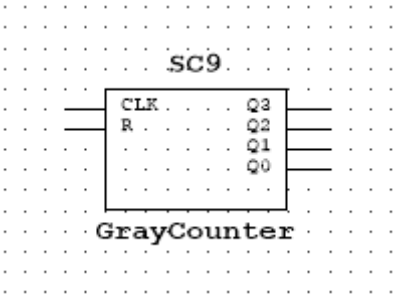


Rysunek 29. Control BOX w układzie.

2.5 Implementacja układu - czterobitowy licznik liczący w kodzie Gray’a.

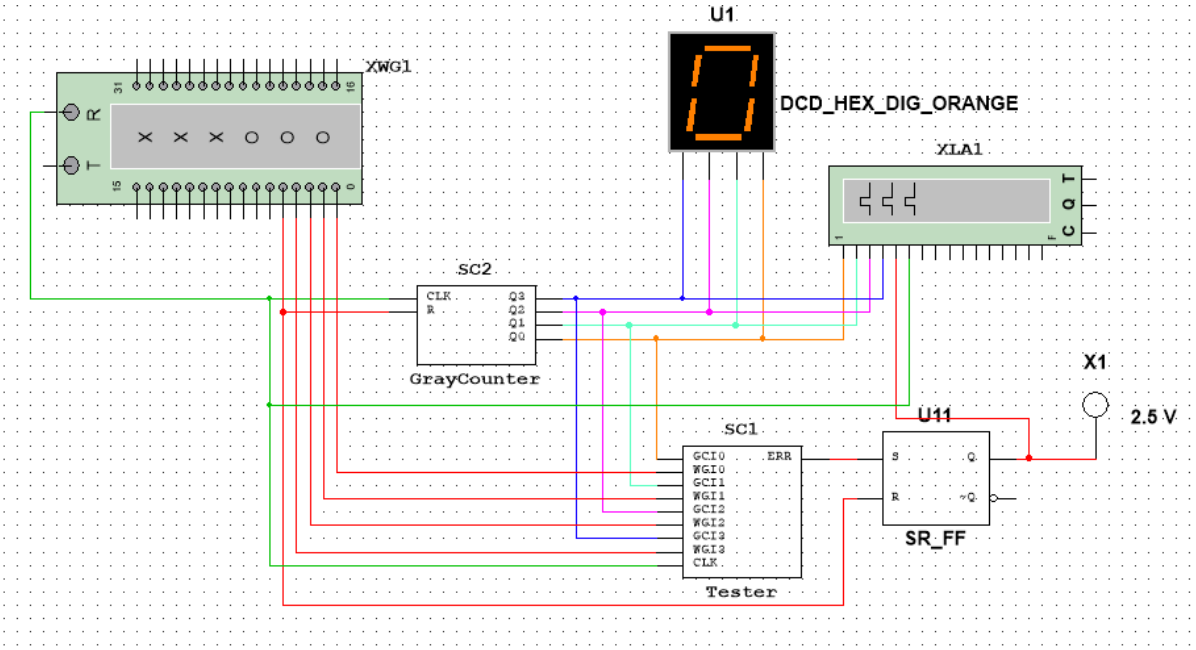


Rysunek 29. Implementacja licznika w programie Multisim.



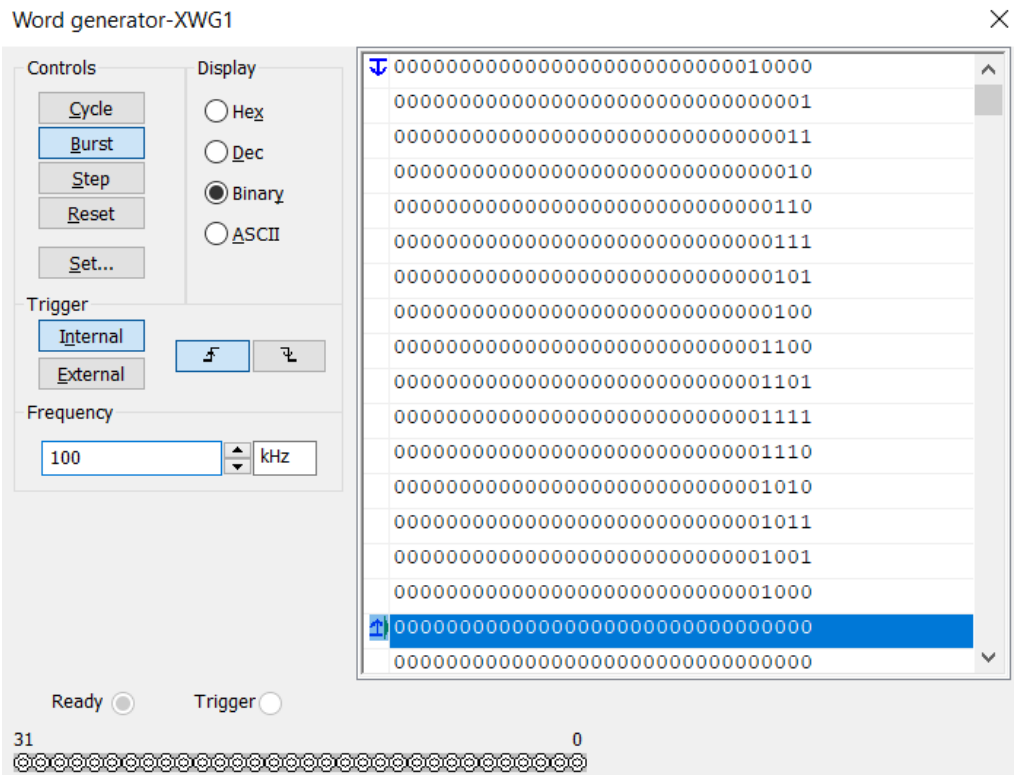
Rysunek 30. Czterobitowy licznik liczący w kodzie Gray’a.

2.6 Układ testujący utworzony synchroniczny licznik liczący w kodzie Gray'a.



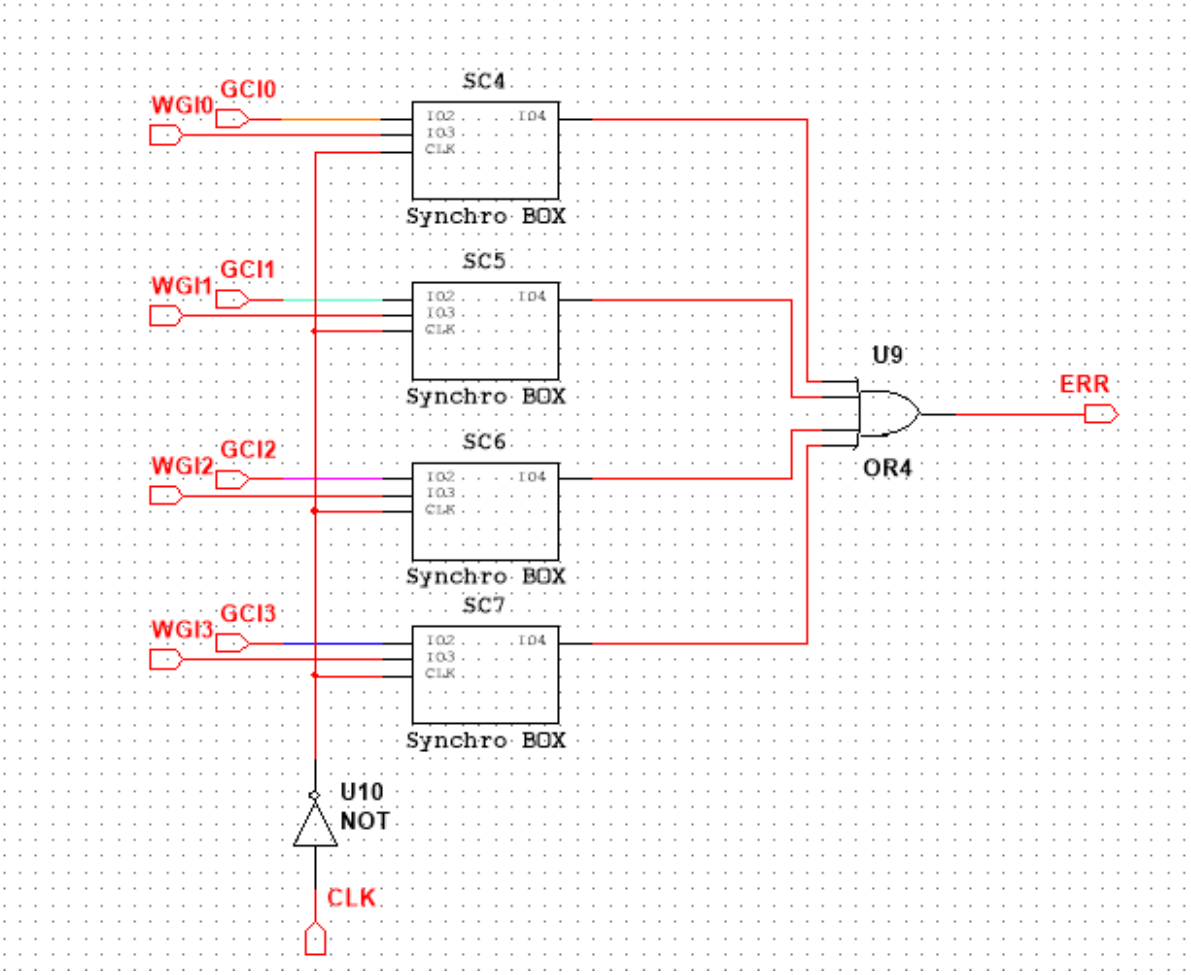
Rysunek 31. Implementacja synchronicznego licznika liczącego w kodzie Gray'a.

2.7 Generator słów

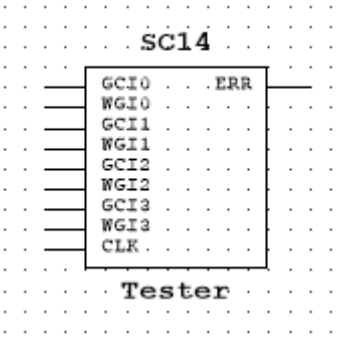


Rysunek 32. Ustawienia generatora słów.

2.8 Układ testujący w programie multisim

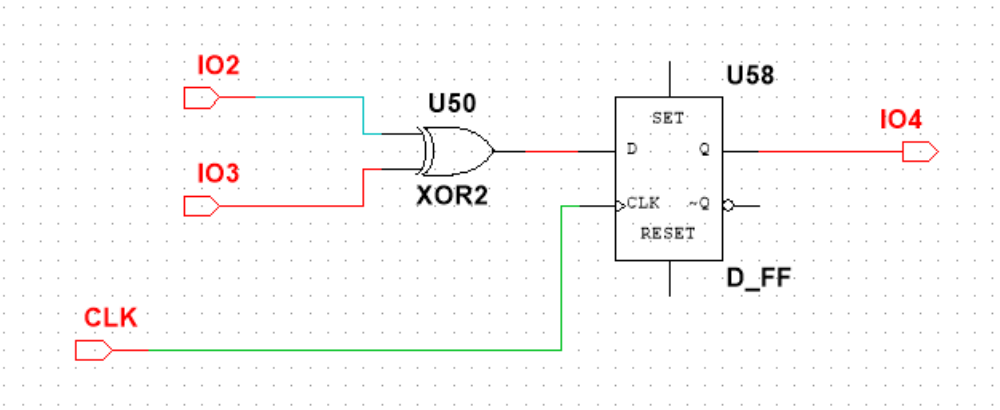


Rysunek 33. Implementacja testera w programie Multisim.

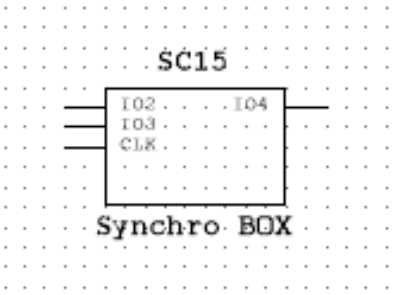


Rysunek 34. Układ podukładów testujących.

Każdy z Synchro BOX'ów(podukład testujący z zaimplementowanym synchronizatorem) w środku zawiera układ:

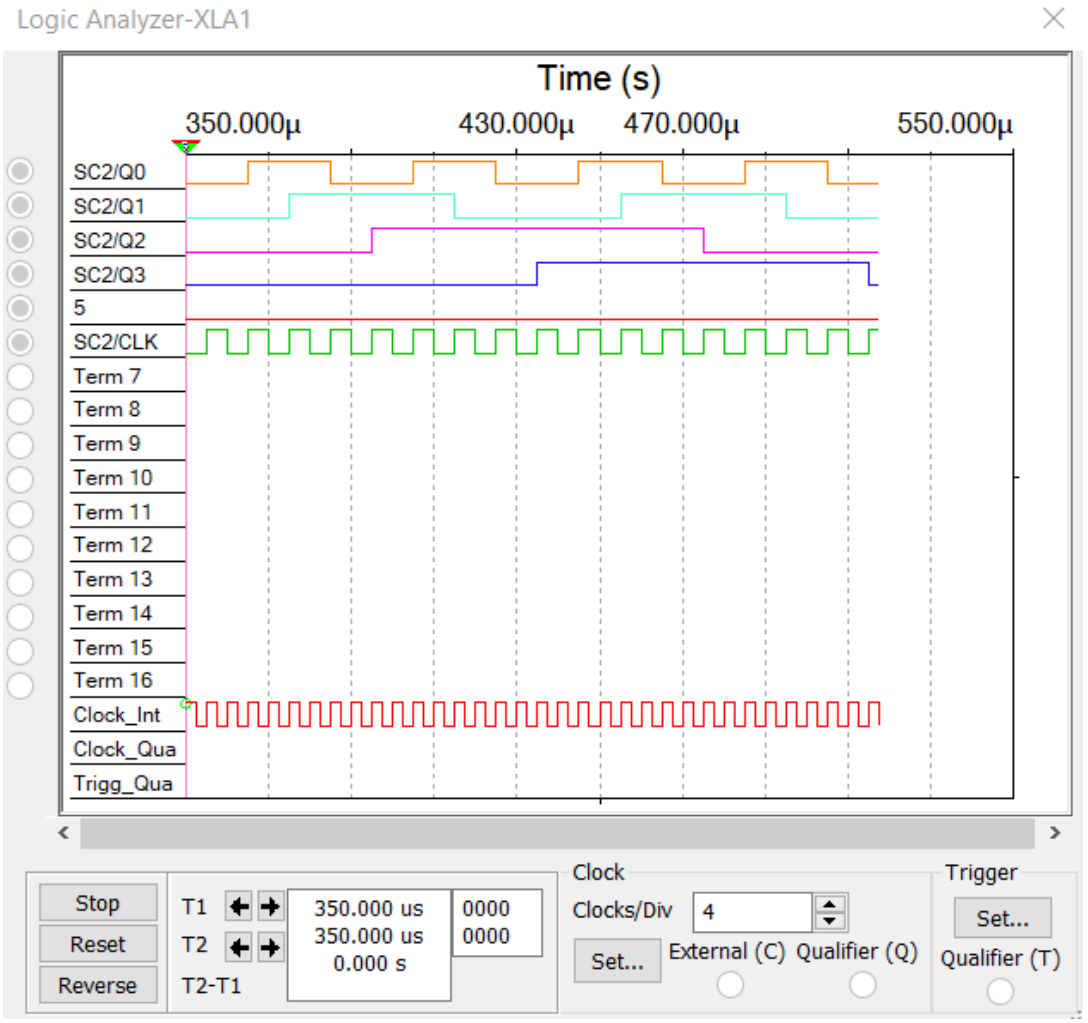


Rysunek 35. Implementacja Synchro BOX'a w programie Mulstisim.

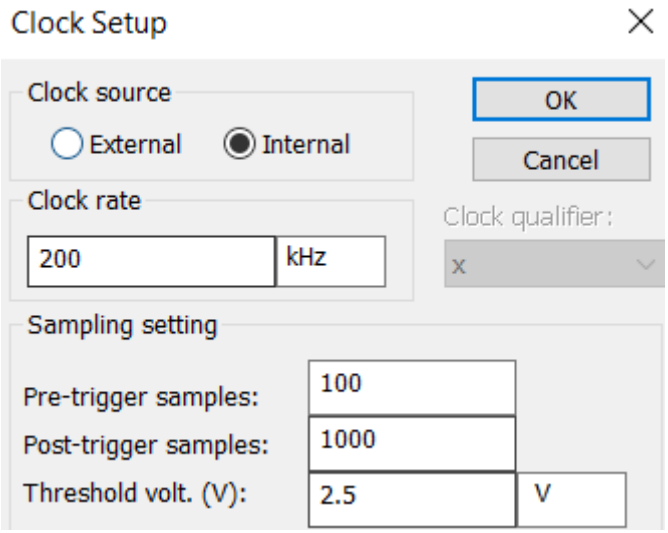


Rysunek 36. Podukład testujący pojedynczy bit.

2.9 Analizator słów



Rysunek 37. Logic analyzer.



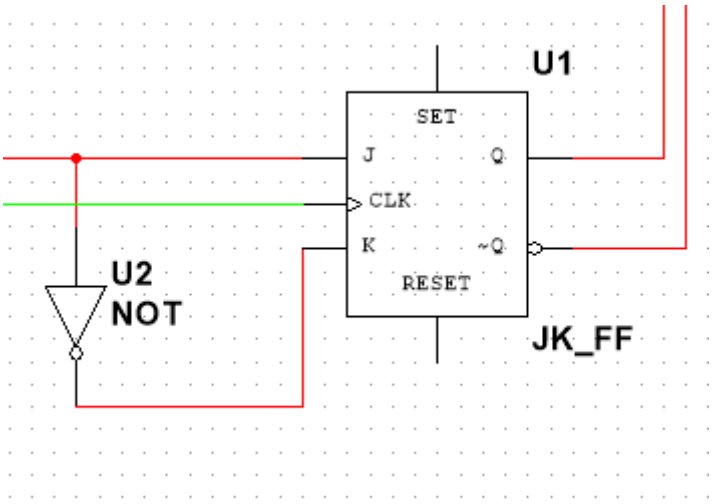
Rysunek 38. Konfiguracja logic analyzera.

2.10 Wnioski

1. Po każdym generowaniu nowego słowa stan tylko jednego przerzutnika ulega zmianie, w związku z tym wykrywanie ewentualnej usterki nie sprawia problemów.

2. Licznik oparty na kodzie Graya przydaje się, gdy wymagana jest bardzo duża precyzja.

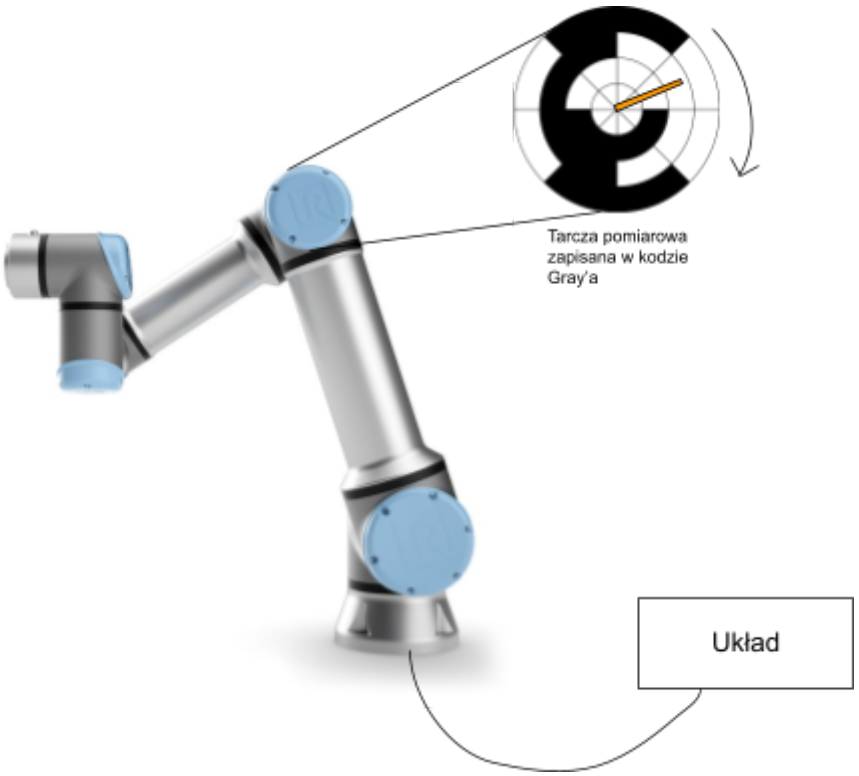
3. Licznik można stworzyć przy pomocy przerzutników JK stosowanych tak samo jak przerzutniki typu D. Z przerzutnika JK bardzo łatwo można bowiem stworzyć przerzutnik D:



Rysunek 39. Przerzutnik D stworzony na bazie przerzutnika JK.

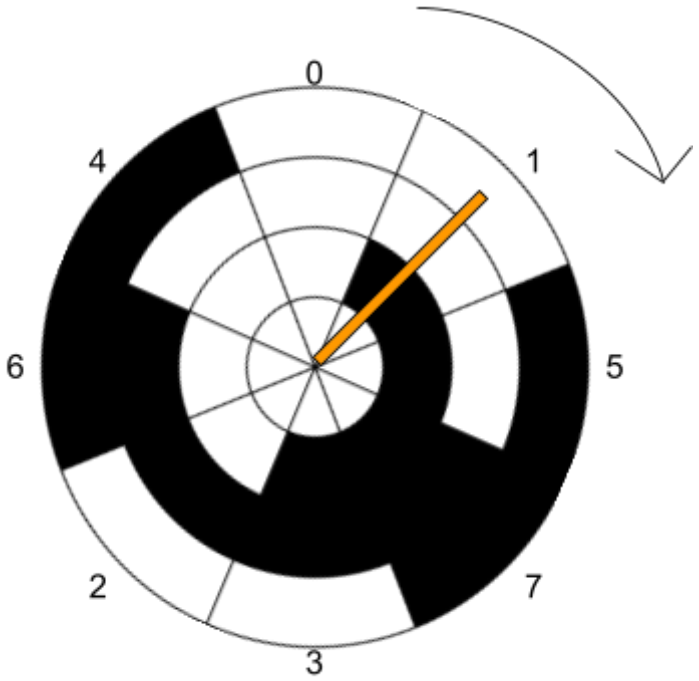
2.11 Przykładowe zastosowania

1. System czytający aktualną pozycję mechanicznego ramienia w robotach przemysłowych. W zależności od pozycji ramienia, licznik przechodzi po pozycjach i zwraca znaleziony stan.



Rysunek 40. Mechaniczne ramię robota przemysłowego.

2. Tarcza do rzutek. W zależności od części tarczy, w którą się trafi, otrzymuje się określoną ilość punktów, odczytywaną za pomocą licznika w kodzie Gray’a.



Rysunek 41. Tarcza do rzutek z kolejnymi wartościami zgodnymi z kodem Gray’a.

3. Zadanie 3 c)

Bazując na przerzutnikach "D", zaprojektować, zbudować i przetestować automat realizujący detekcję litery Q przekazywanej alfabetem Morsa, czyli sekwencji bitów: "— — • —". Za kreskę proszę przyjąć stan logiczny '1', a za kropkę stan logiczny '0'. Proszę zaproponować własny ale skuteczny i praktyczny sposób wprowadzania do urządzenia sygnału wejściowego.

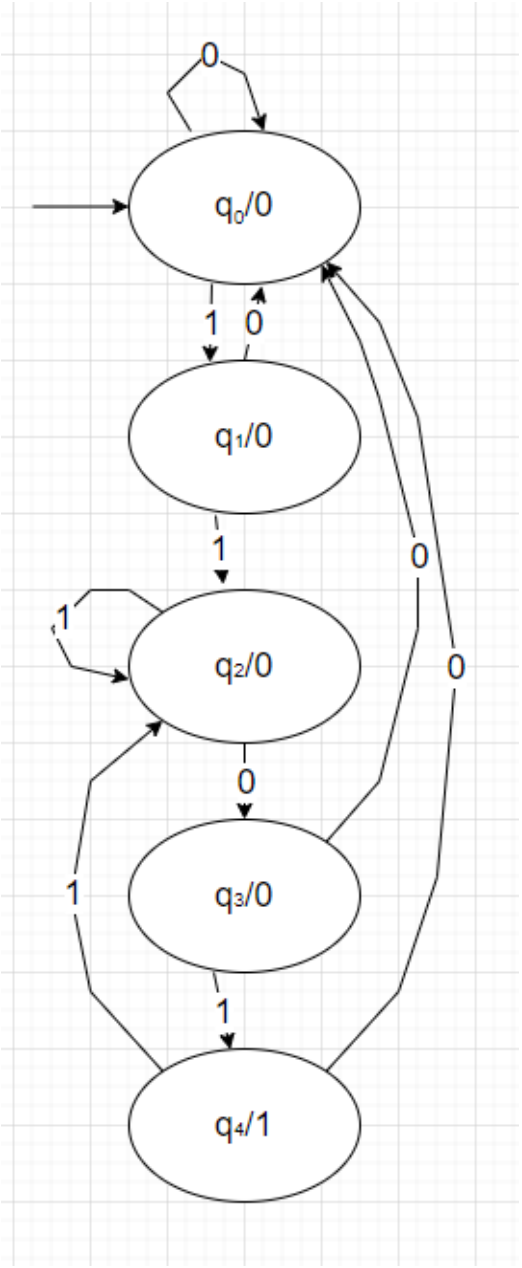
3.1. Wprowadzenie

Rysunek 42. Początkowa idea.

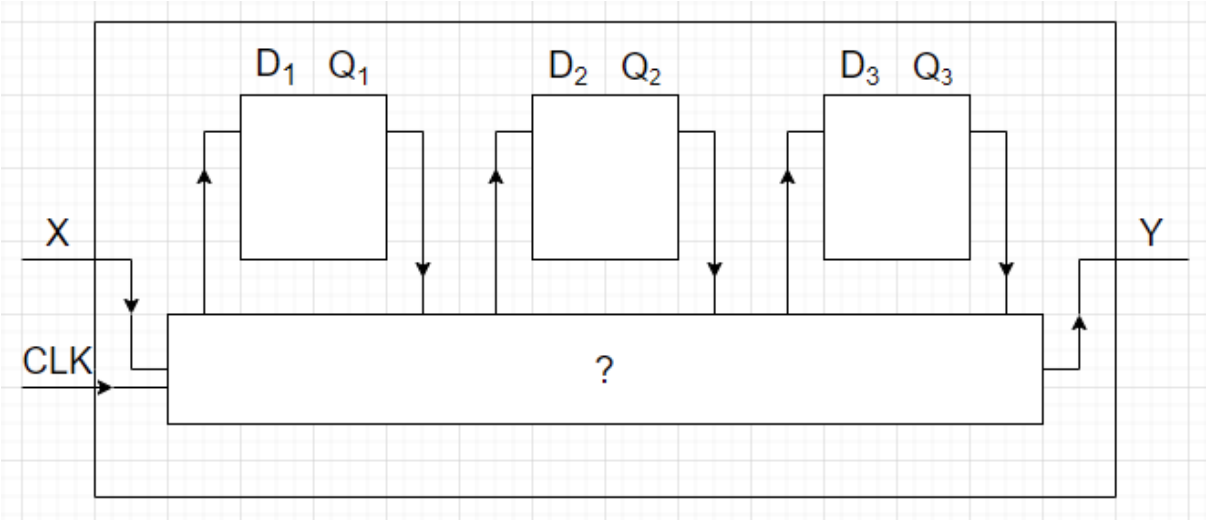
3.2. Automat Moore’a

Stan	Q_1	Q_2	Q_3
q_0	0	0	0
q_1	0	0	1
q_2	0	1	1
q_3	1	1	0
q_4	1	1	1

Tabela 43. Tabela stanów.



Rysunek 44. Diagram automatu Moore’a.



Rysunek 45. Rozbudowana idea.

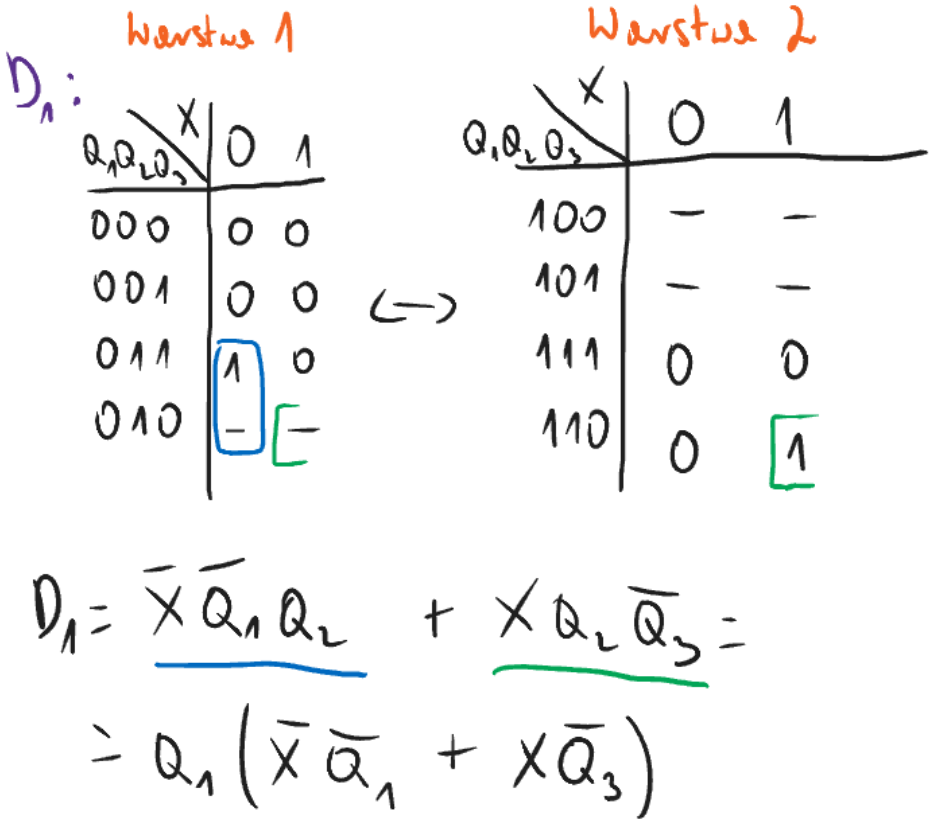
$Q_1Q_2Q_3\backslash X$	0	1
000	000	001
001	000	011
010	-	-
011	110	011
100	-	-
101	-	-
110	000	111
111	000	011

$Q_1Q_2Q_3$	Y
000	0
001	0
010	-
011	0
100	-
101	-
110	0
111	1

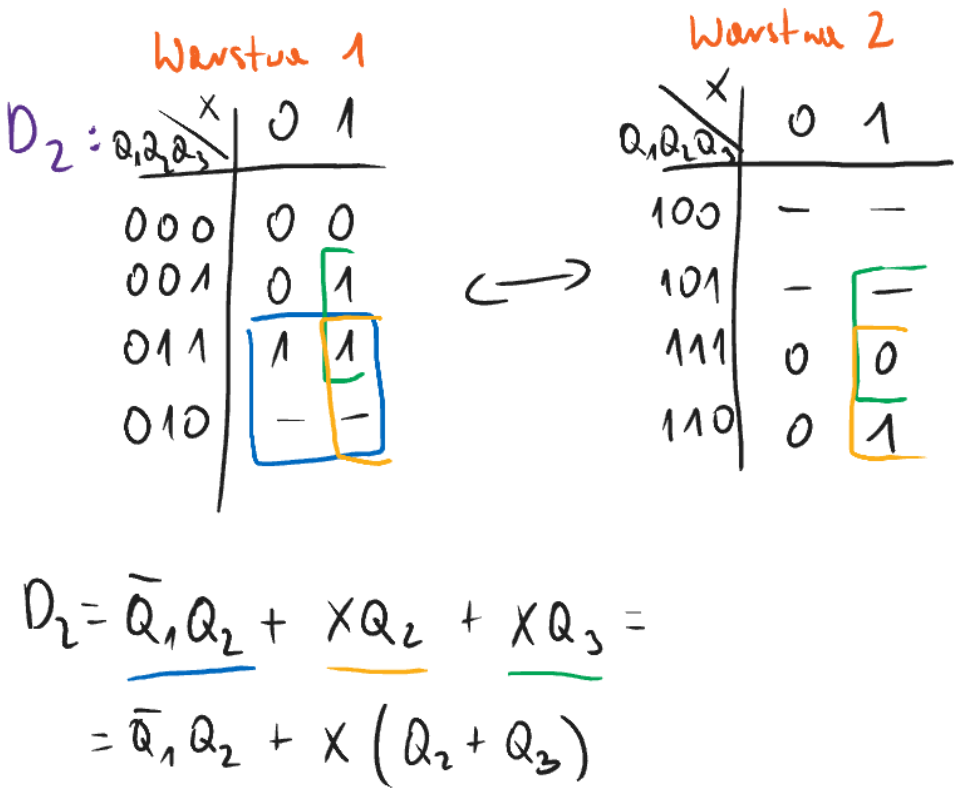
Tabela 3 i tabela 4. Tabela przejść pomiędzy stanami oraz tabela wyjść.

3.2 Wyprowadzenie

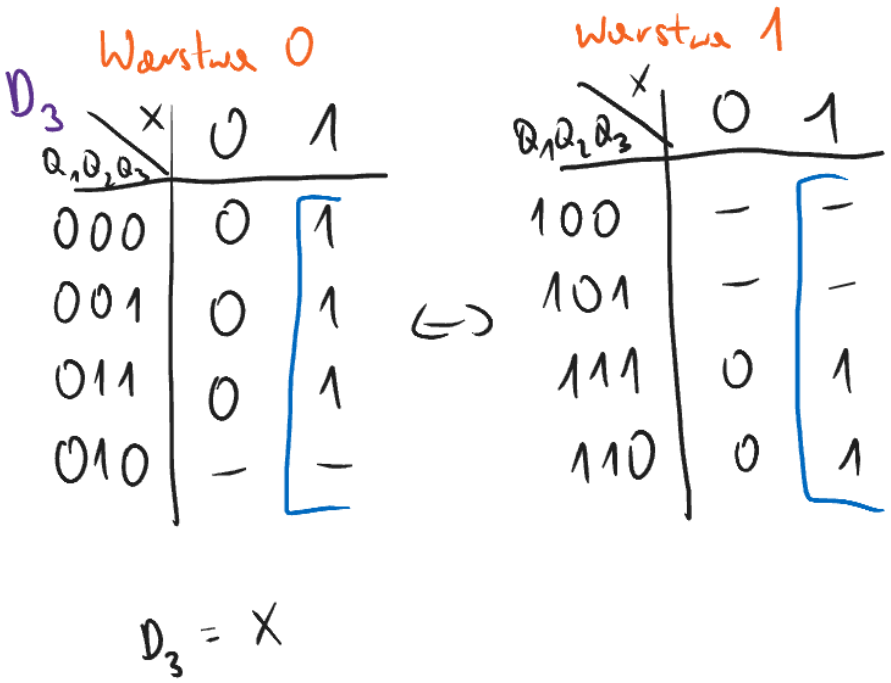
Tabele Karnaugha i wzory dla przerzutników D:



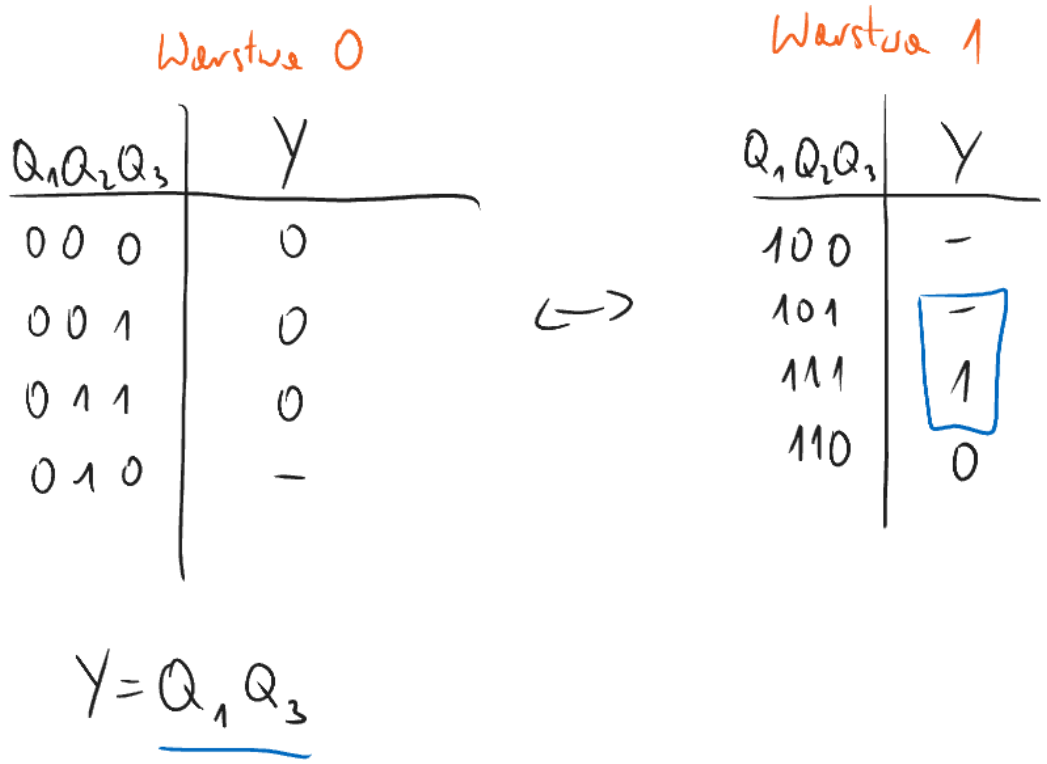
Rysunek 46. Wyprowadzenie dla D_1.



Rysunek 47. Wyprowadzenie dla D_2.

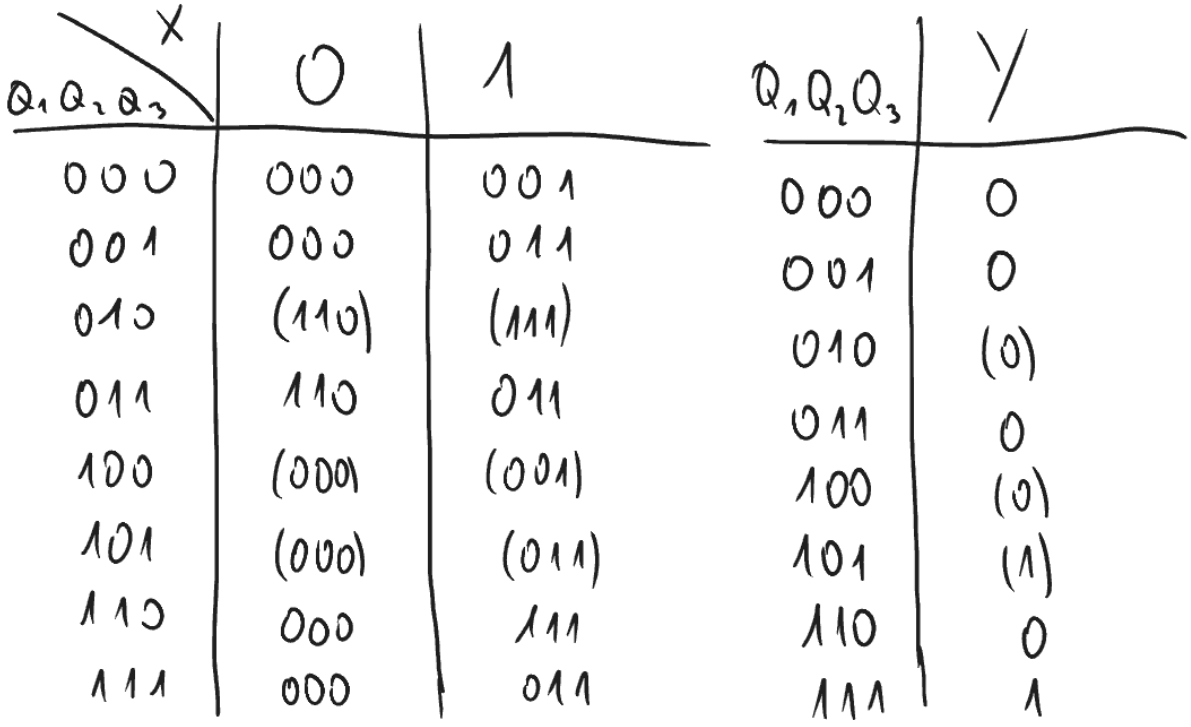


Rysunek 48. Wyprowadzenie dla D_3.



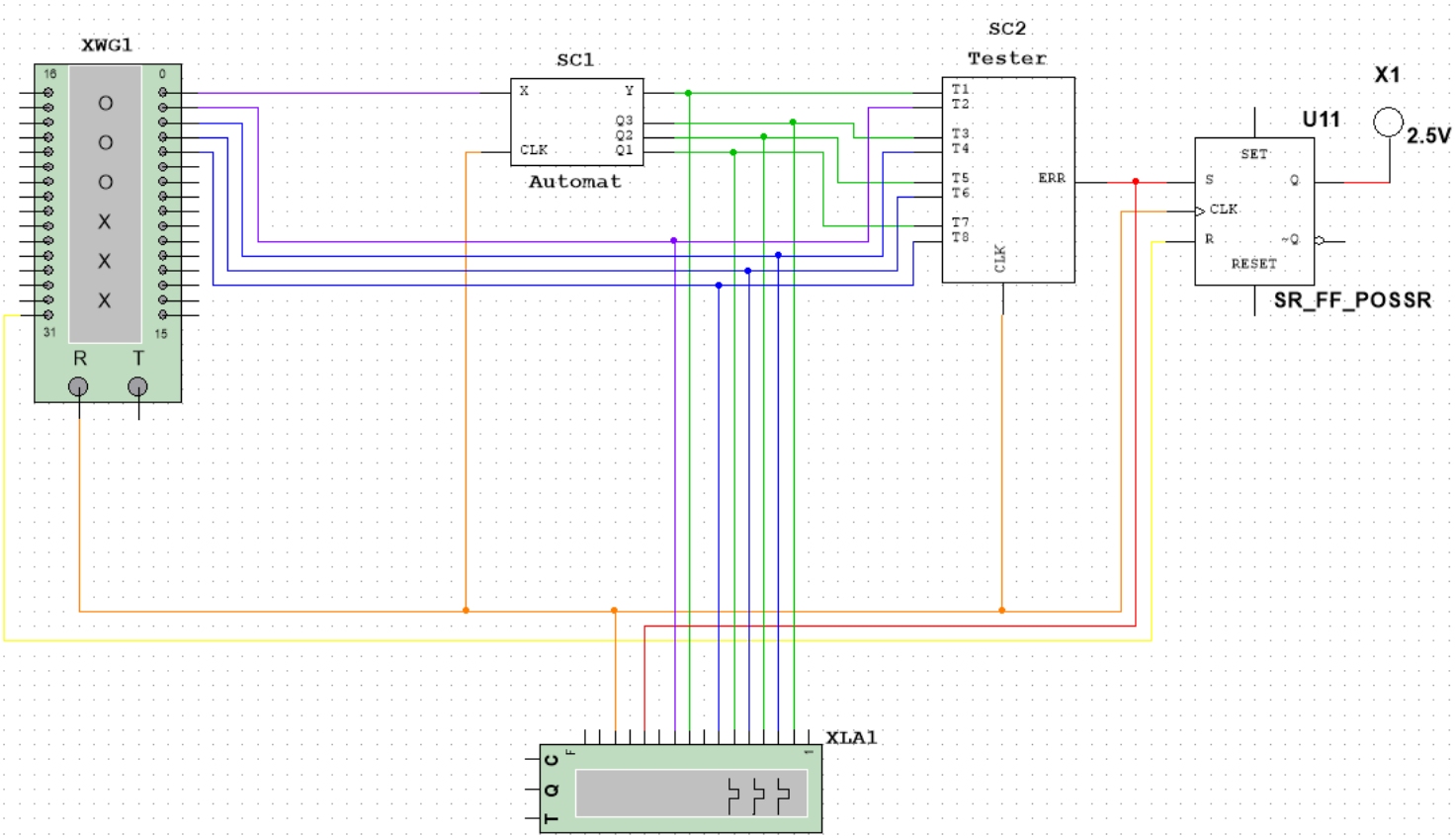
Rysunek 49. Wyprowadzenie dla wyjścia.

Tabela przejść pomiędzy stanami i tabela wyjść z określonymi stanami nieosiągalnymi określonymi tak, aby z tabel Karnougha otrzymać prostsze wzory:



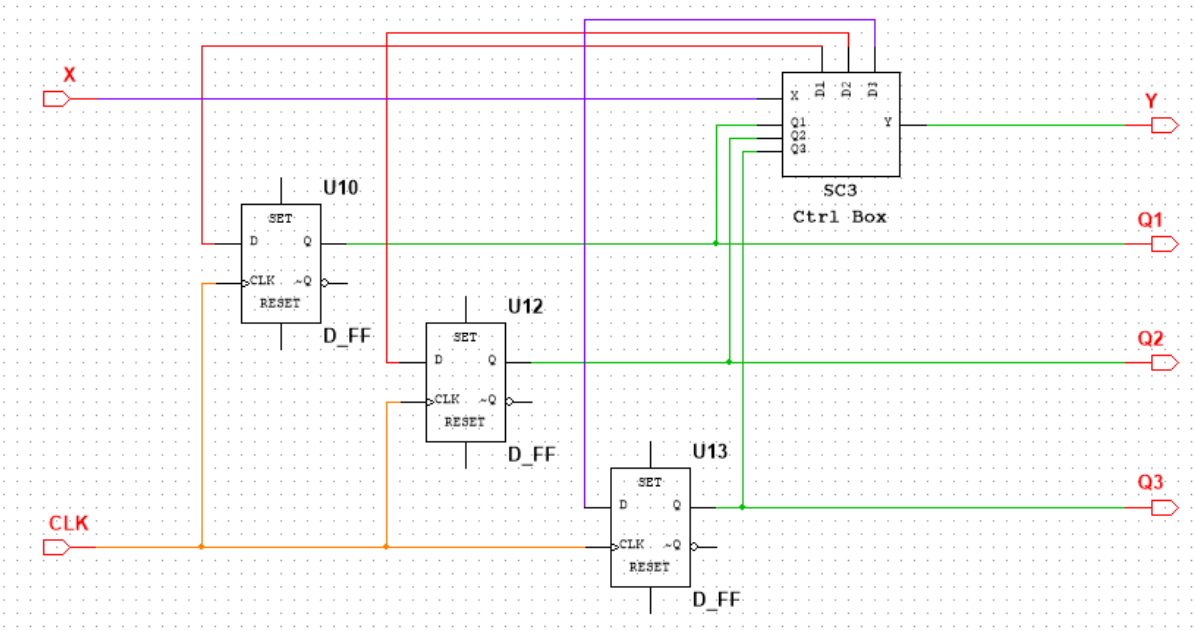
Rysunek 50. Wyprowadzenie dla wyjścia.

3.3 Implementacja układu

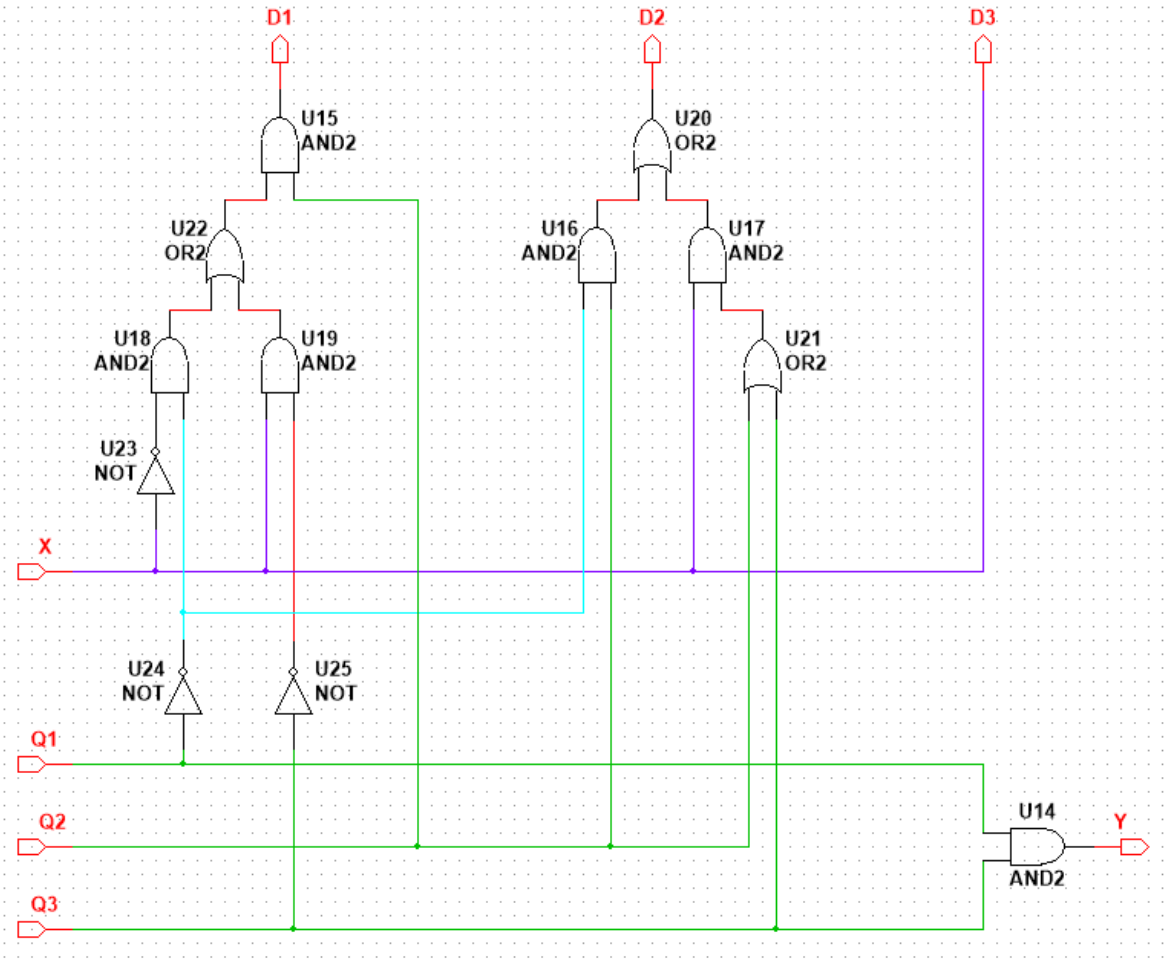


Rysunek 51. Implementacja układu wraz z testami w programie Multisim.

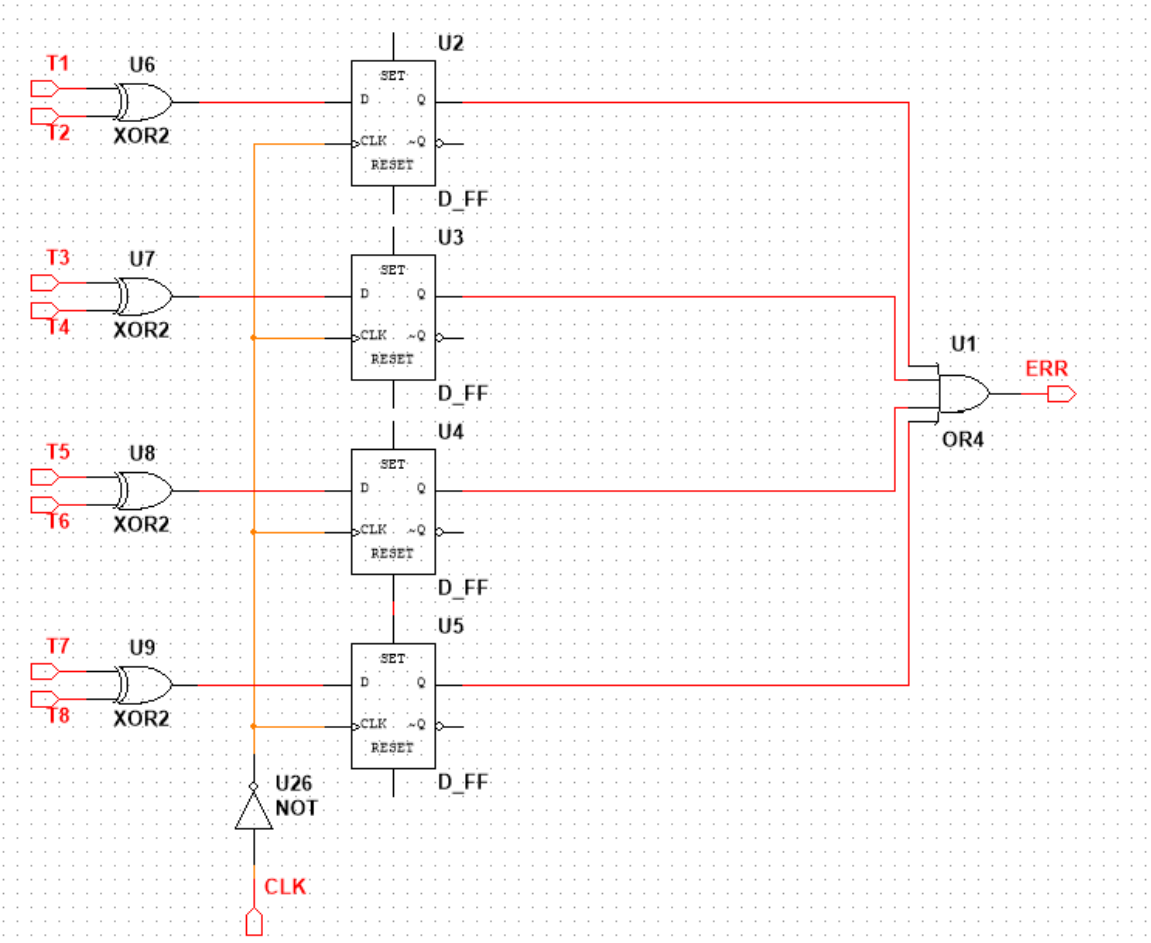
3.4 Podukłady



Rysunek 52. Implementacja układu automatu Q w programie Multisim.

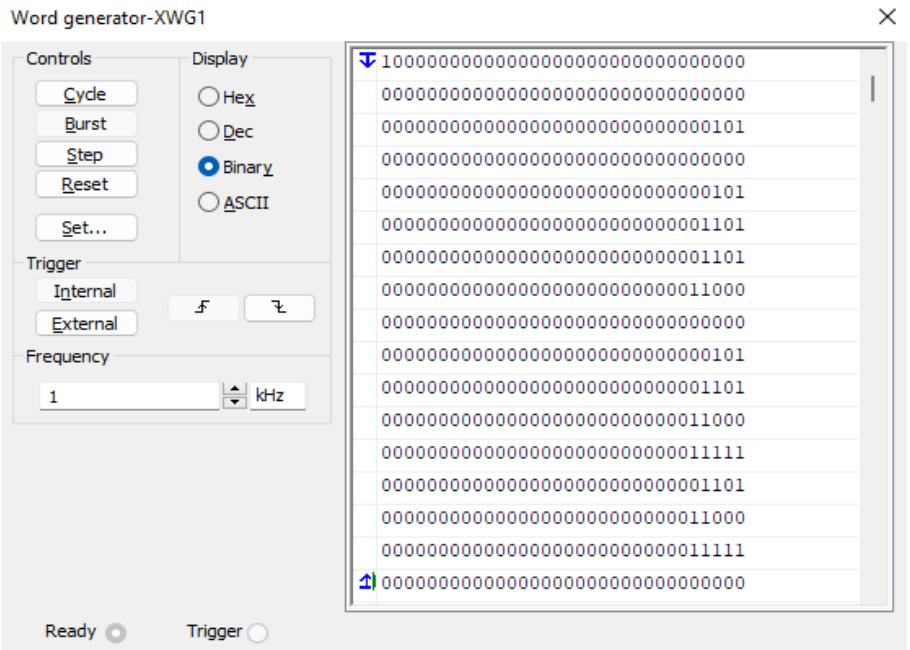


Rysunek 53. Implementacja control-boxa w programie Multisim.



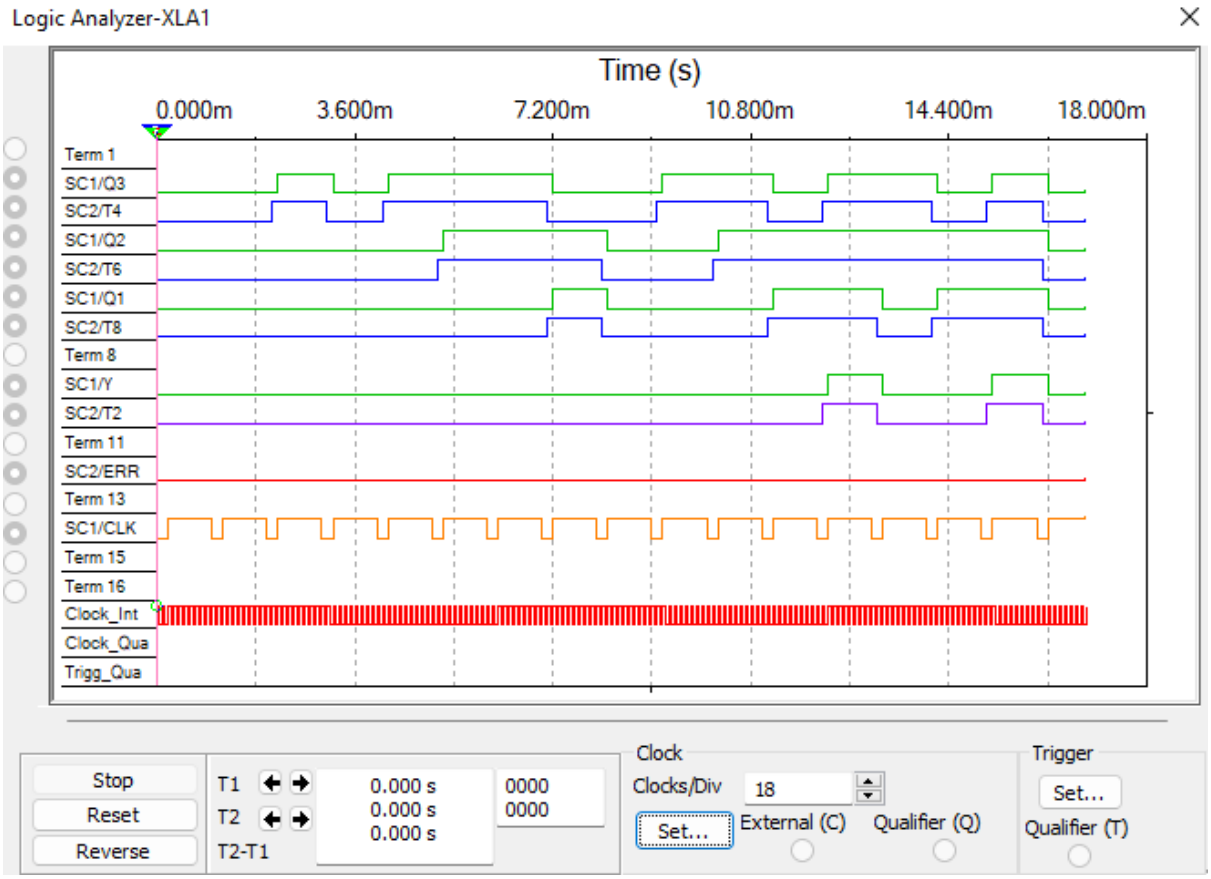
Rysunek 54. Implementacja testera w programie Multisim.

3.5 Generator słów

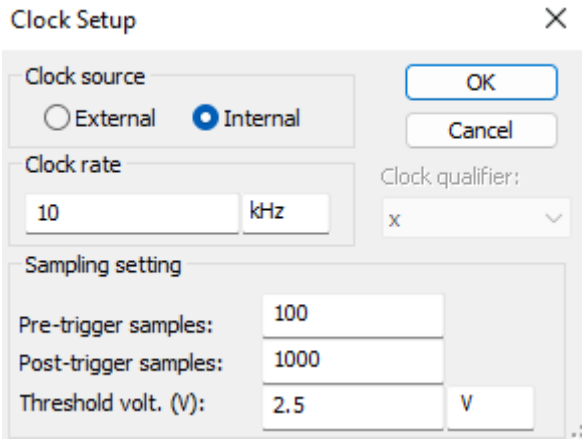


Rysunek 55. Ustawienia generatora słów.

3.6 Analizator słów



Rysunek 56. Analizator słów.



Rysunek 57. Ustawienia analizatora słów.

3.7 Wnioski

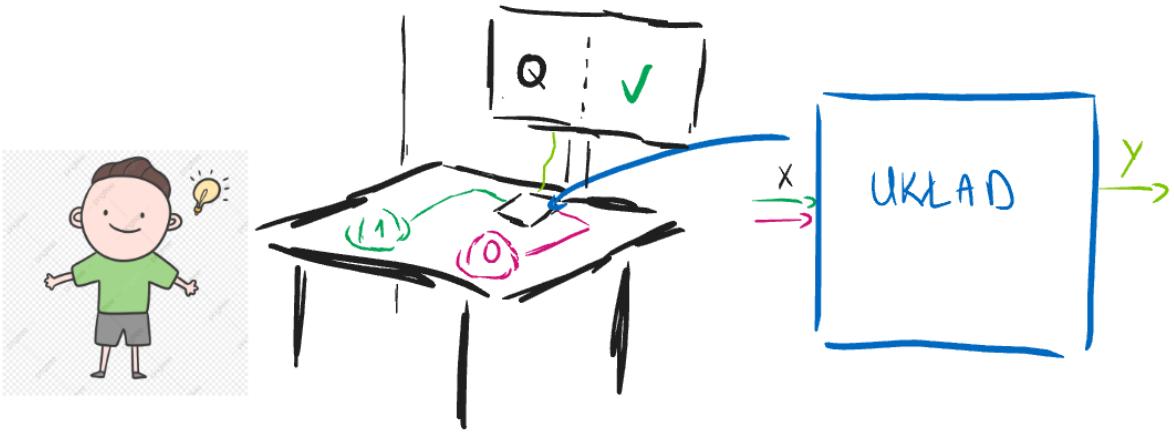
1. Zaczynanie rozwiązywania zadania od narysowania automatu w postaci grafu ułatwia jego późniejszą realizację.

2. Automaty są optymalnym rozwiązaniem w projektowaniu skomplikowanych logicznie sekwencyjnych układów cyfrowych.

3. Alternatywne rozwiązanie: zamiast korzystania z automatu Moore’a można by się posłużyć automatem Meale’a(Wymogiem w realizacji zastosowania automatu Meale’a jest niestety zapamiętywanie poprzedniego stanu, jest to więc pewnego rodzaju minus).

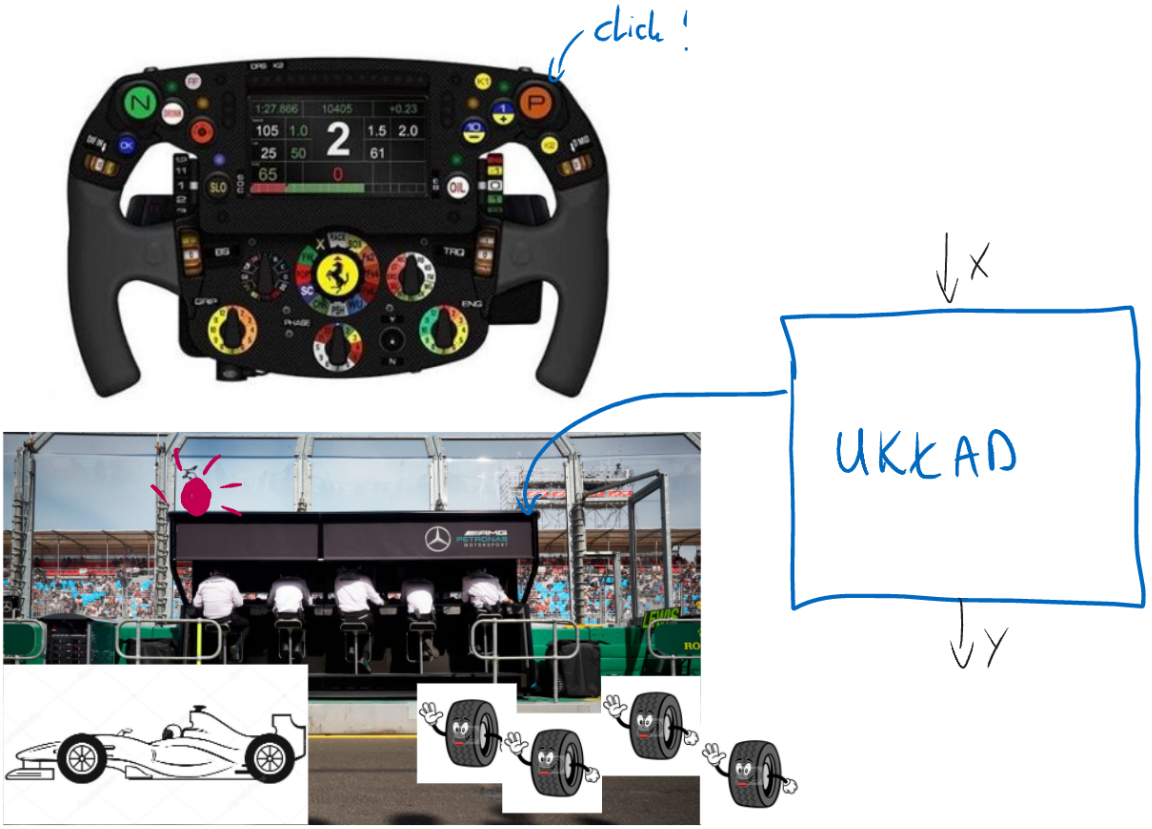
3.8 Przykładowe zastosowania

1. Zabawka dla dziecka, polegająca na detekcji kolejnego znaku w celu nauki alfabetu morse'a.



Rysunek 58. Zabawka edukacyjna wykorzystująca zaprojektowany automat.

2. System pozwalający na sygnalizację potrzeby wymiany opon dla kierowców formuły 1. Każdemu kierowcy biorącemu udział w wyścigu przypisujemy literę alfabetu morse'a. Jeżeli zaistnieje potrzeba wymiany opon, zawodnik wysyła kodem Morse'a swoją literę, natomiast centrala odbiera sygnał, następnie zachodzi detekcja za pomocą automatu, dzięki czemu wiemy, która drużyna zaraz będzie wymieniać opony.



Rysunek 59. System pozwalający na sygnalizację potrzeby wymiany opon dla kierowców formuły 1.