Ćwiczenie 2.

Emil Siatka, Marek Swakoń, Patryk Górski, Piotr Pich

Data oddania: 29.04.2025

Spis treści

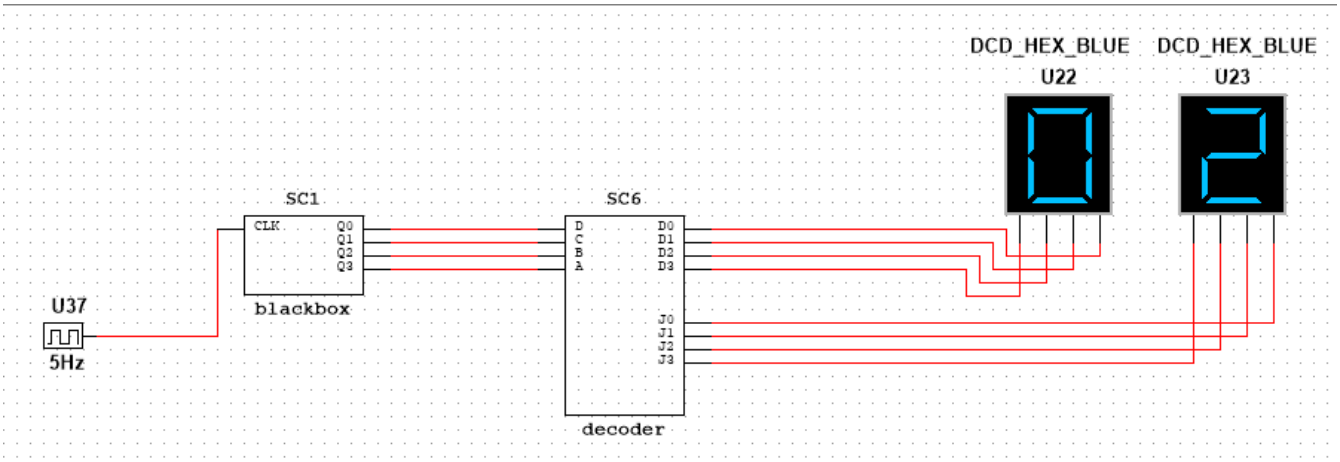
1 Treść ćwiczenia

Korzystając tylko z konkretnego jednego typu przerzutników oraz z dowolnych bramek logicznych, proszę zaprojektować czterobitowy licznik działający zgodnie z ciągiem Fibonacciego (z nieobowiązkowym upraszczającym zastrzeżeniem, że wartość "1"powinna się pojawiać tylko raz w cyklu). Po uruchomieniu licznika, w kolejnych taktach zegara powinien on zatem przechodzić po wartościach:

0, 1, 2, 3, 5, 8, 13, 0, 1, 2, 3, 5, 8, 13, 0, 1, ... itd.

Aktualna wartość wskazywana przez licznik powinna być widoczna na wyświetlaczach siedmiosegmentowych.

Interfejs naszego układu będzie wyglądał podobnie do przedstawionego na rysunku 1.



Rysunek 1: Interfejs projektowanego układu

Szczególnie interesuje nas podukład `blackbox`, który docelowo implementował będzie interesujący nas mechanizm "zliczania"kolejnych liczb ciągu Fibonacciego.

Przedstawić możemy go jako licznik którego sygnałami wyjściowymi są stany wyjścia czterech przerzutników typu T, które na wejściu T przyjmują wynik czterech czteroargumentowych funkcji zdaniowych T_0, T_1, T_2, T_3, których argumentami są sygnały wyjściowe w poprzednim cyklu zegara, funkcje T_0, T_1, T_2, T_3 wówczas, przez prostą konwersję pierwszych 8. liczb ciągu Fibonacciego na system dwójkowy, przyjmują poniższe wartości:

Tabela 1: Tabela prawdy dla funkcji T_0, T_1, T_2, T_3

$Q_0^{(n-1)}$	$Q_1^{(n-1)}$	$Q_2^{(n-1)}$	$Q_3^{(n-1)}$		T_0	T_1	T_2	T_3
0	0	0	0		0	0	0	1
0	0	0	1		0	0	1	1
0	0	1	0		0	0	0	1
0	0	1	1		0	1	1	0
0	1	0	0		x	x	x	x
0	1	0	1		1	1	0	1
0	1	1	0		x	x	x	x
0	1	1	1		x	x	x	x
1	0	0	0		0	1	0	1
1	0	0	1		x	x	x	x
1	0	1	0		x	x	x	x
1	0	1	1		x	x	x	x
1	1	0	0		x	x	x	x
1	1	0	1		1	1	0	1
1	1	1	0		x	x	x	x
1	1	1	1		x	x	x	x

Podukład `decoder` natomiast, przyjmuje 4 bitową liczbę binarną i zwraca dwie 4-bitowe liczby których wartości są cyframi liczby z zakresu 0-19 w systemie dziesiętnym.

Podukład ten traktować możemy jako układ którego sygnałami wyjściowymi jest 8 czteroarguentowych funkcji zdaniowych (D_0, D_1, D_2, D_3 dla liczby dziesiątek i J_0, J_1, J_2, J_3 dla liczby jedności) przyjmujących poniższe wartości:

Tabela 2: Tabela prawdy dla sygnałów wyjściowych podukładu `decoder`

Q_3	Q_2	Q_1	Q_0		D_0	D_1	D_2	D_3	J_0	J_1	J_2	J_3
0	0	0	0		0	0	0	0	0	0	0	0
0	0	0	1		0	0	0	0	1	0	0	0
0	0	1	0		0	0	0	0	0	1	0	0
0	0	1	1		0	0	0	0	1	1	0	0
0	1	0	0		x	x	x	x	x	x	x	x
0	1	0	1		0	0	0	0	1	0	1	0
0	1	1	0		x	x	x	x	x	x	x	x
0	1	1	1		x	x	x	x	x	x	x	x
1	0	0	0		0	0	0	0	0	0	0	1
1	0	0	1		x	x	x	x	x	x	x	x
1	0	1	0		x	x	x	x	x	x	x	x
1	0	1	1		x	x	x	x	x	x	x	x
1	1	0	0		x	x	x	x	x	x	x	x
1	1	0	1		1	0	0	0	1	0	0	0
1	1	1	0		x	x	x	x	x	x	x	x
1	1	1	1		x	x	x	x	x	x	x	x

Aby zaprojektować licznik, pozostaje zatem zminimalizować metodą tablic Karnaugh odpowiednio funkcje T_0, T_1, T_2, T_3 (dla podukładu `blackbox`) oraz $D_0, D_1, \ldots, J_3$ (dla podukładu `decoder`) a następnie zbudować go wraz ze stanowiskiem testowym w programie `Multisim`.

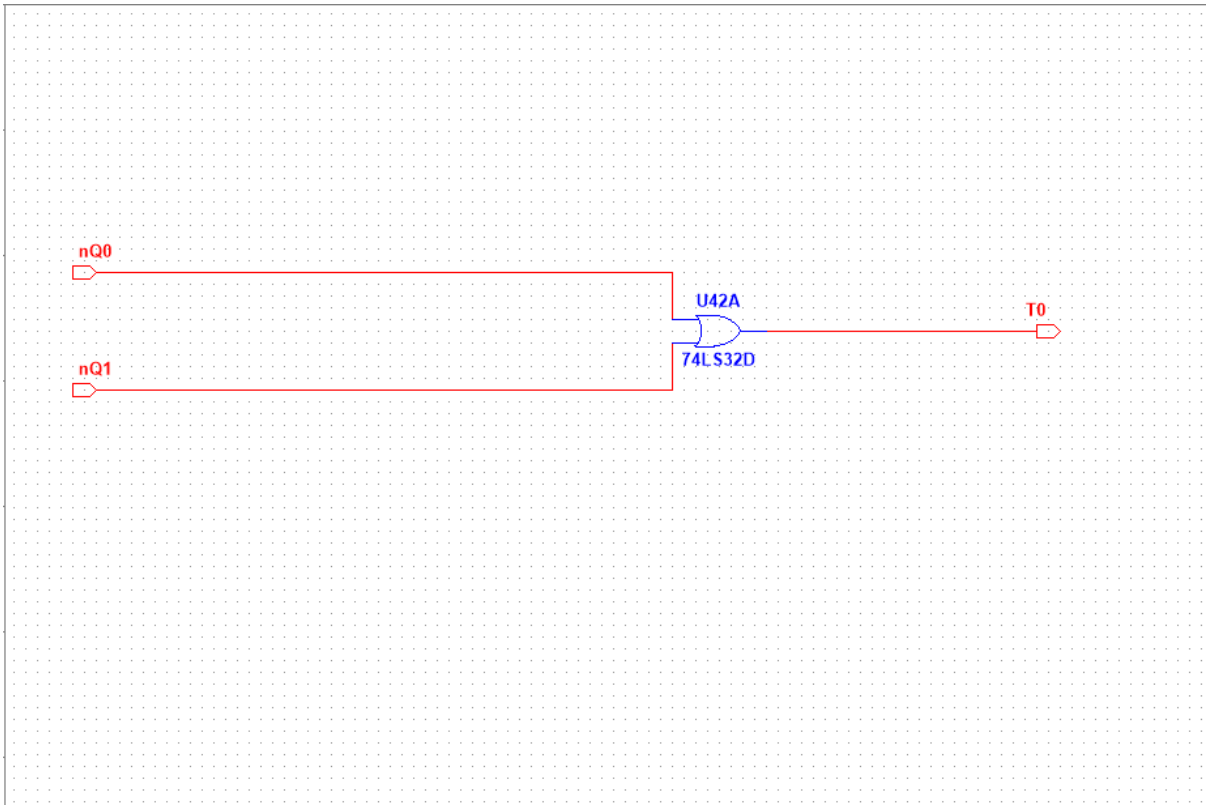
2 Minimalizacja funkcji zdaniowych

2.1 Podukład `blackbox`

Funkcje T_1, T_2, T_3 oraz T_4 z Tabeli 1. minimalizują poniższe tablice Karnaugh (Oznaczenia A, B, C, D odpowiadają argumentom funkcji Q_0, Q_1, Q_2, Q_3):

AB\CD	00	01	11	10
00	1	1	0	1
01	x	1	x	x
11	x	1	x	x
10	1	x	x	x

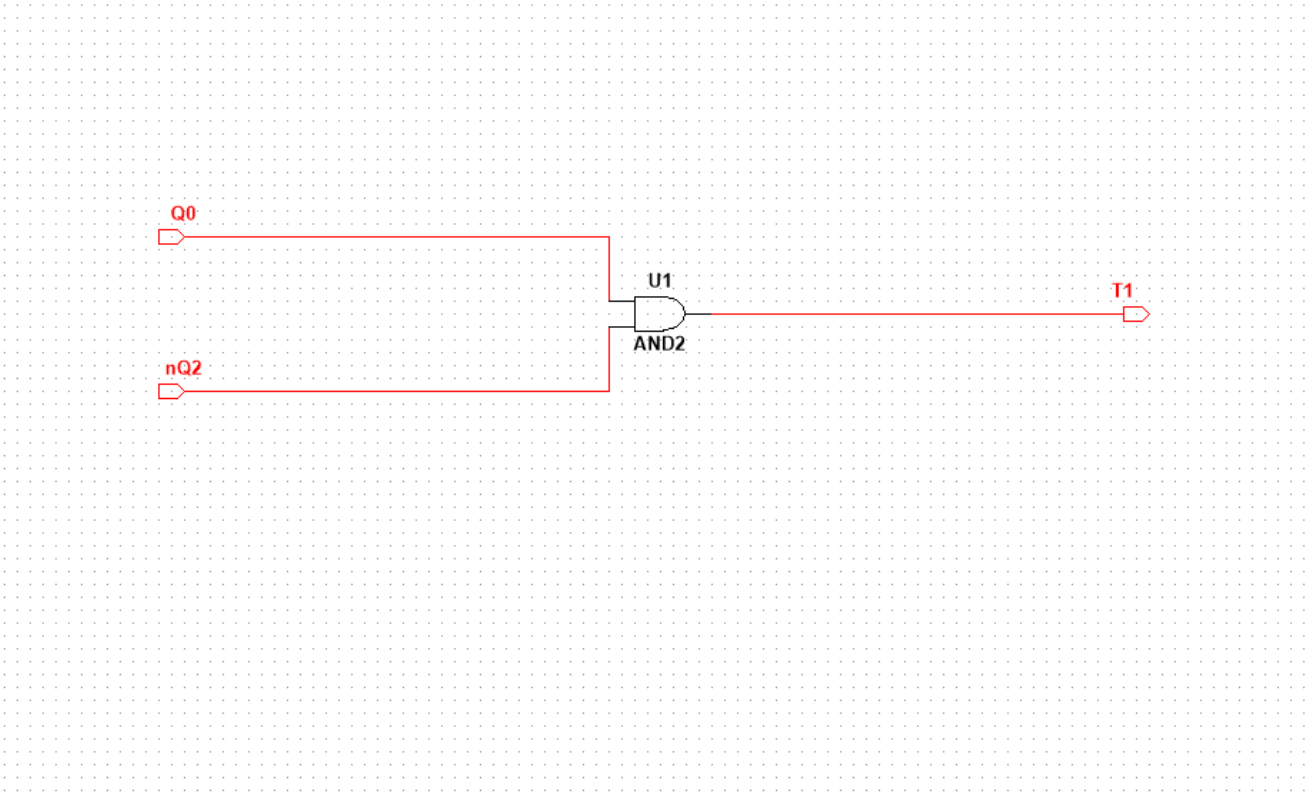
Rysunek 2: Tablica Karnaugh dla funkcji T_0



Rysunek 3: Bramka obliczająca wartości funkcji T_0

AB\CD	00	01	11	10
00	0	1	1	0
01	x	0	x	x
11	x	0	x	x
10	0	x	x	x

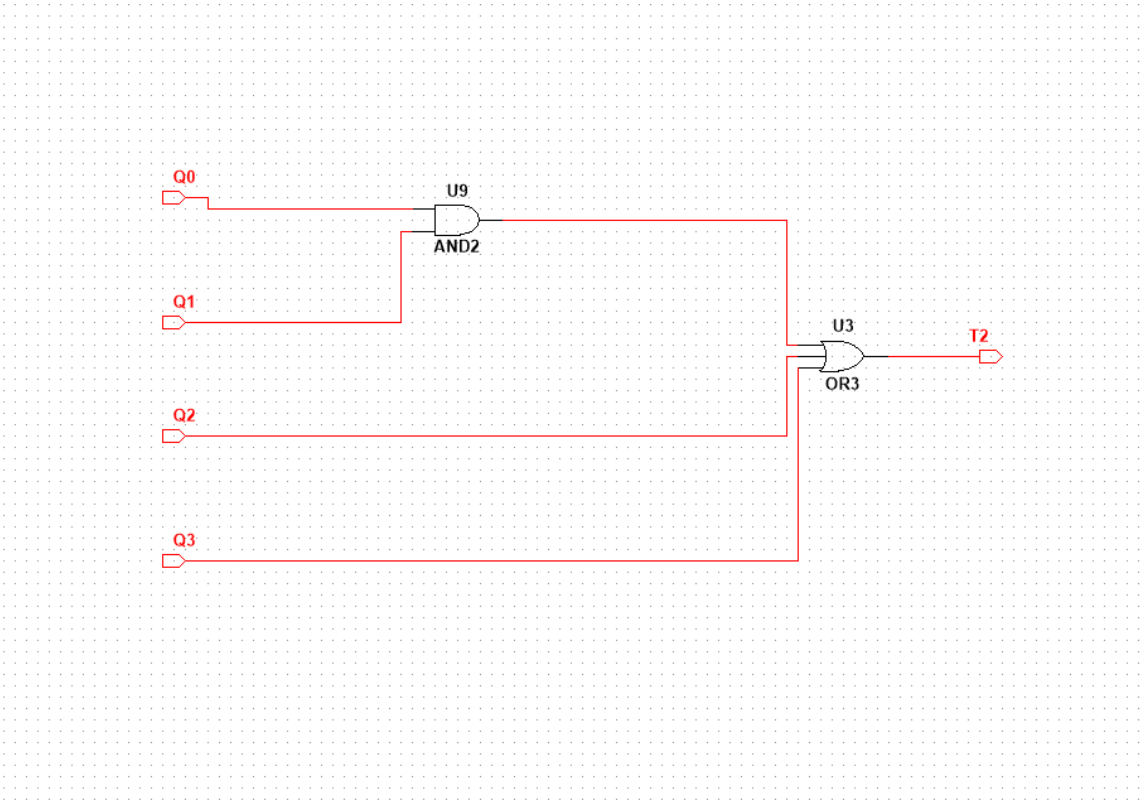
Rysunek 4: Tablica Karnaugh dla funkcji T_1



Rysunek 5: Bramka obliczająca wartości funkcji T_1

AB\CD	00	01	11	10
00	0	0	1	0
01	X	1	X	X
11	X	1	X	X
10	1	X	X	1

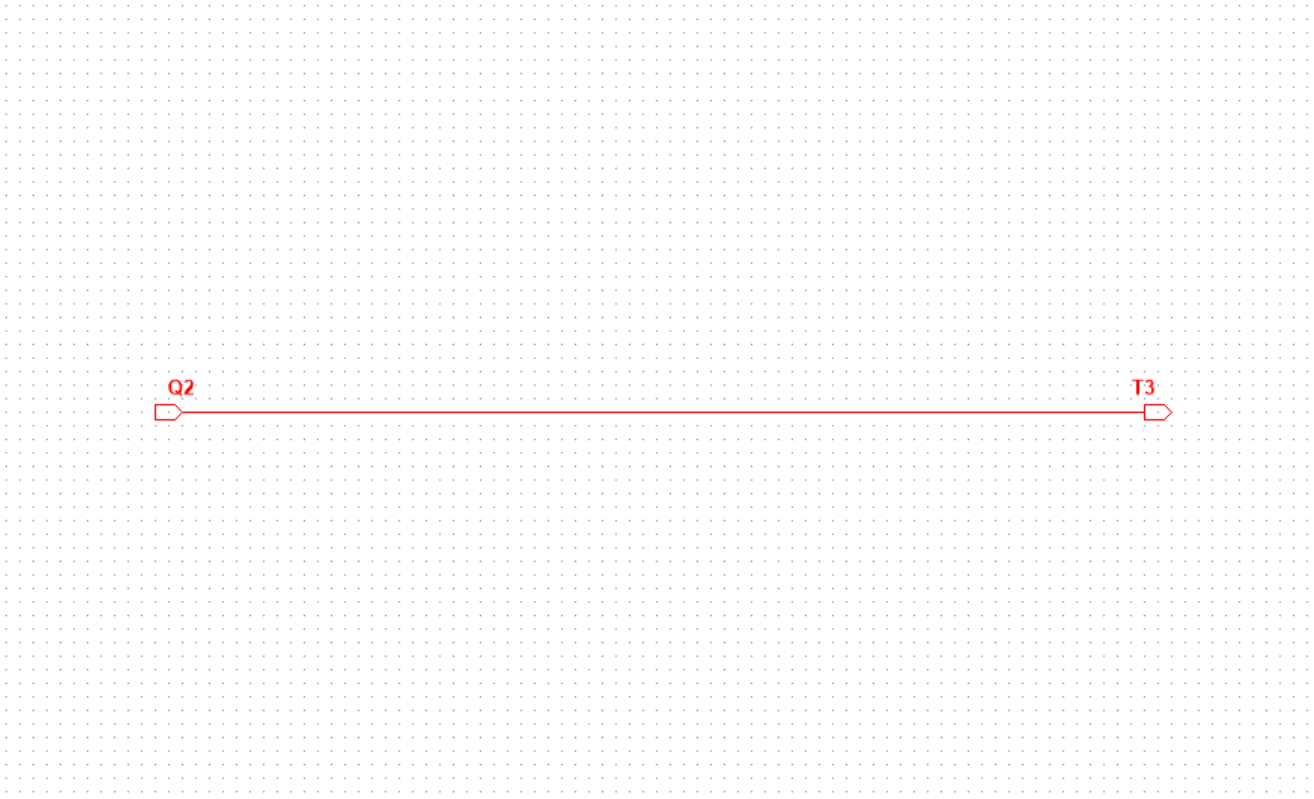
Rysunek 6: Tablica Karnaugh dla funkcji T_2



Rysunek 7: Bramka obliczająca wartości funkcji T_2

AB\CD	00	01	11	10
00	0	0	0	0
01	x	1	x	x
11	x	1	x	x
10	0	x	x	0

Rysunek 8: Tablica Karnaugh dla funkcji T_3



Rysunek 9: Bramka obliczająca wartości funkcji T_3

2.2 Podukład `decoder`

Z Tabeli 2. jak również z faktu, że interesuje na wyłącznie konwersja na liczby dziesiętne mniejsze od 20, obserwujemy trywialne uproszczenie:

$$D_1 = D_2 = D_3 = 0 \tag{1}$$

Q3Q2\Q1Q0	00	01	11	10
00	0	0	0	0
01	x	0	x	x
11	x	1	x	x
10	0	x	x	x

Rysunek 10: Tablica Karnaugh dla funkcji D_0

Q3Q2\Q1Q0	00	01	11	10
00	0	1	0	0
01	x	1	x	x
11	x	1	x	x
10	0	x	x	x

Rysunek 11: Tablica Karnaugh dla funkcji J_0

Q3Q2\Q1Q0	00	01	11	10
00	0	0	1	1
01	x	0	x	x
11	x	1	x	x
10	0	x	x	x

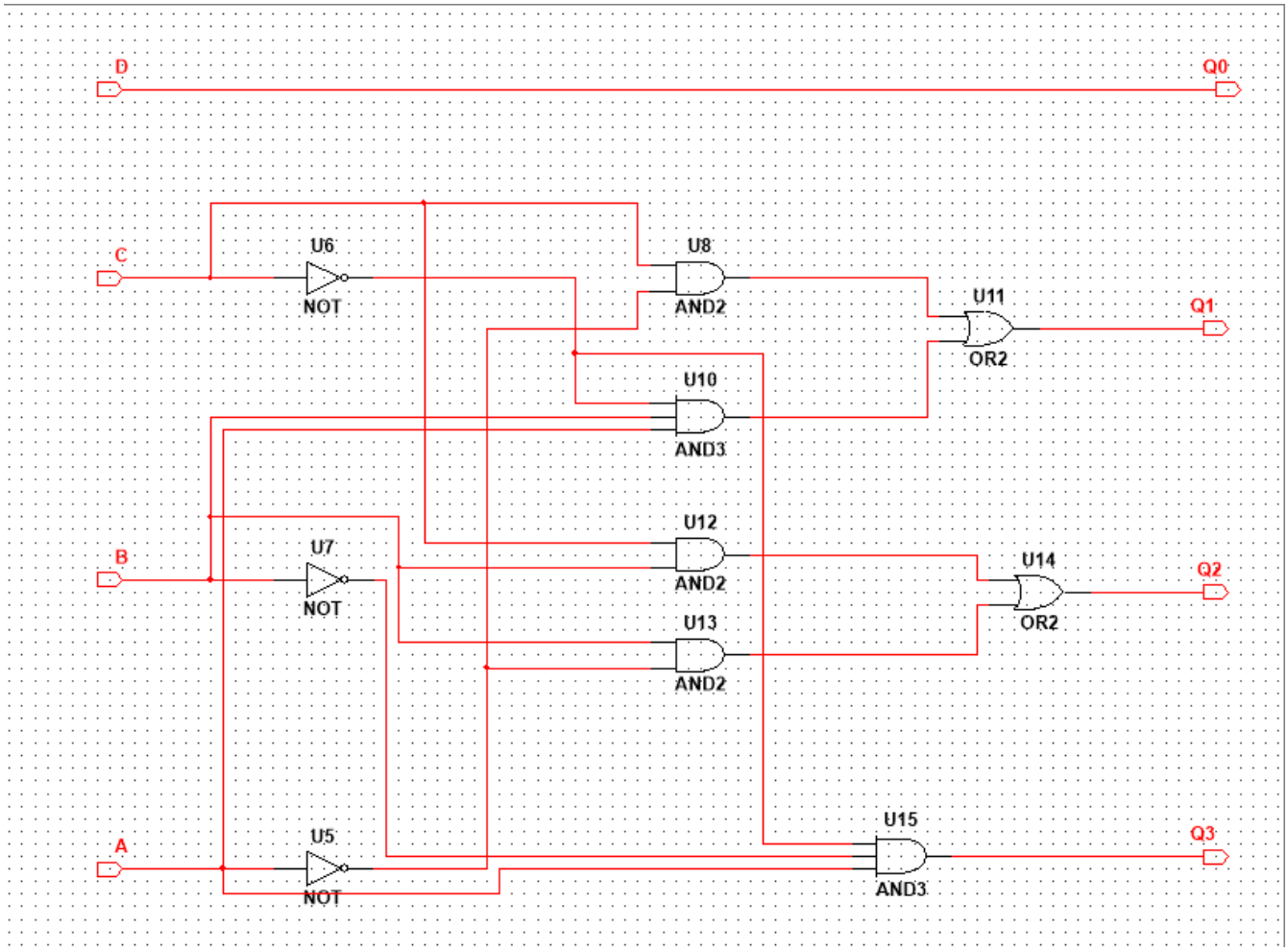
Rysunek 12: Tablica Karnaugh dla funkcji J_1

Q3Q2\Q1Q0	00	01	11	10
00	0	0	0	0
01	x	1	x	x
11	x	0	x	x
10	0	x	x	x

Rysunek 13: Tablica Karnaugh dla funkcji J_2

Q3Q2\Q1Q0	00	01	11	10
00	0	0	0	0
01	x	0	x	x
11	x	0	x	x
10	1	x	x	x

Rysunek 14: Tablica Karnaugh dla funkcji J_3

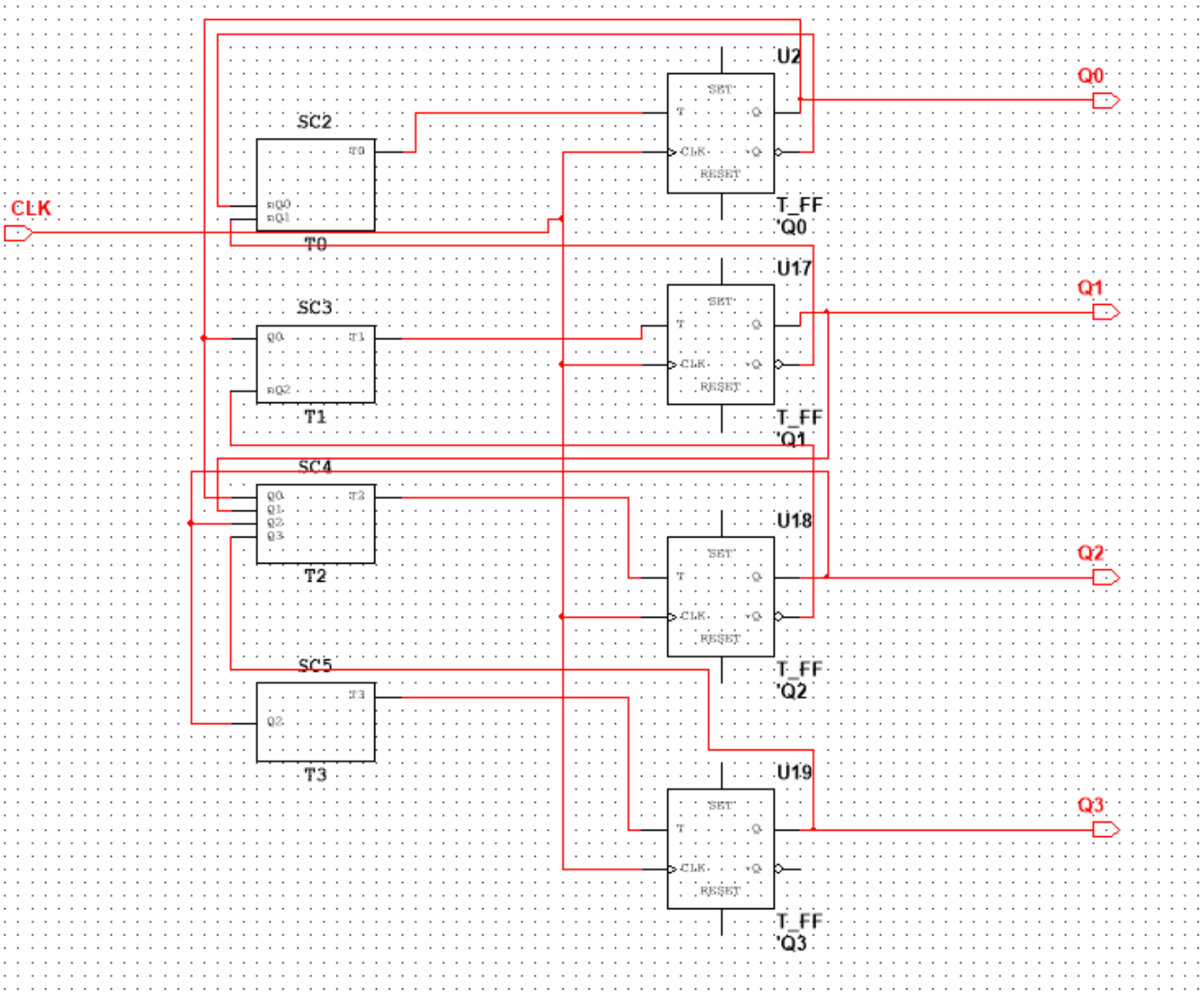


Rysunek 15: Bramka obliczająca wartości funkcji $J_0, \ldots, J_3$ (na rysunku $Q_0, \ldots, Q_3$)

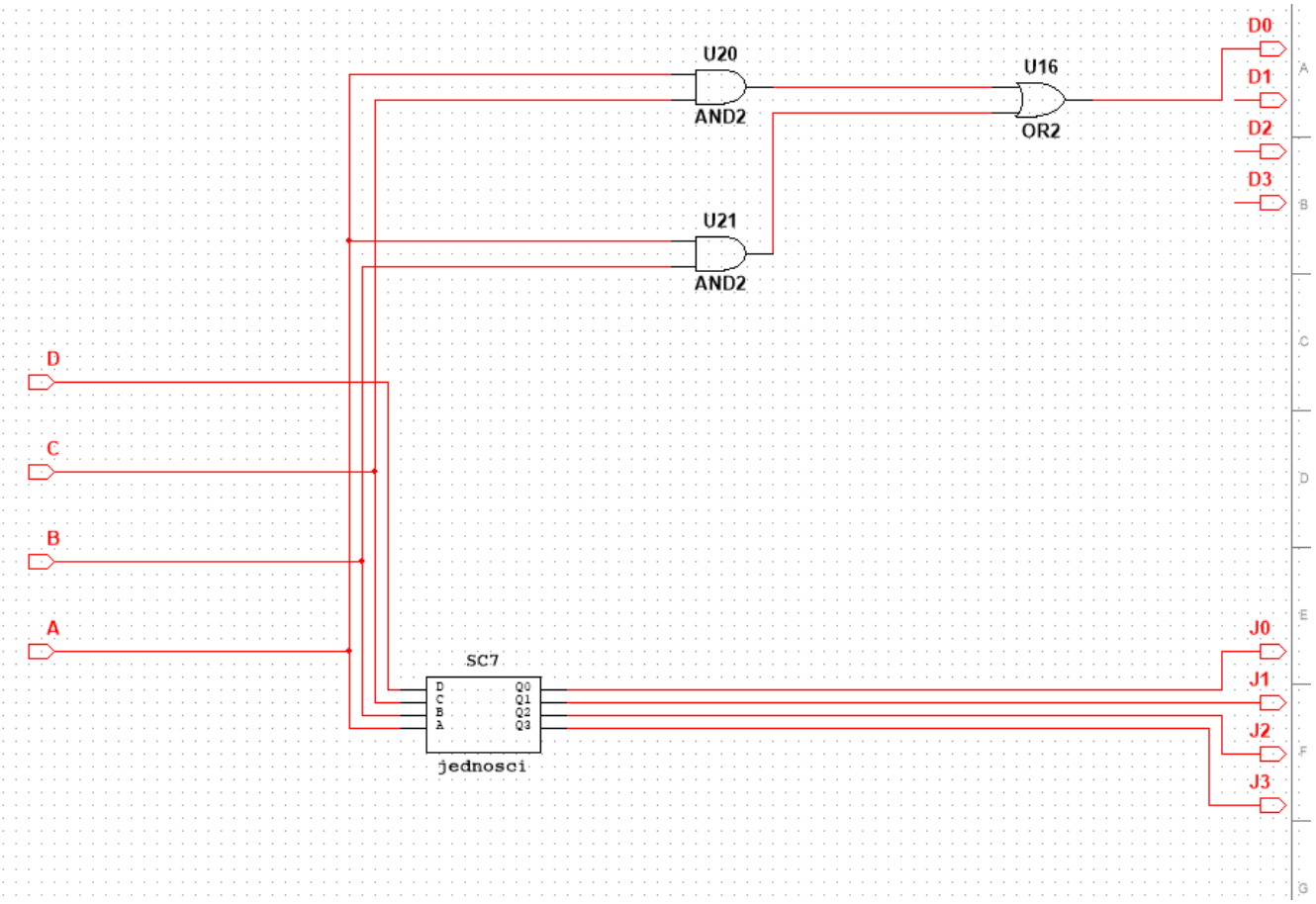
3 Schemat układu

W oparciu o wyprowadzone wzory na sygnały wyjściowe, oraz poczynione we wprowadzeniu założenia możemy stworzyć schemat licznika w programie `Multisim`.

Nasz licznik wzbogacamy dotatkowo o wejście dla sygnału resetującego, które podpinamy bezpośrednio do przerzutników typu T, co będzie przydatne podczas projektowania stanowiska testującego.



Rysunek 16: Schemat podukładu `blackbox` z podziałem na bramki w programie `Multisim`



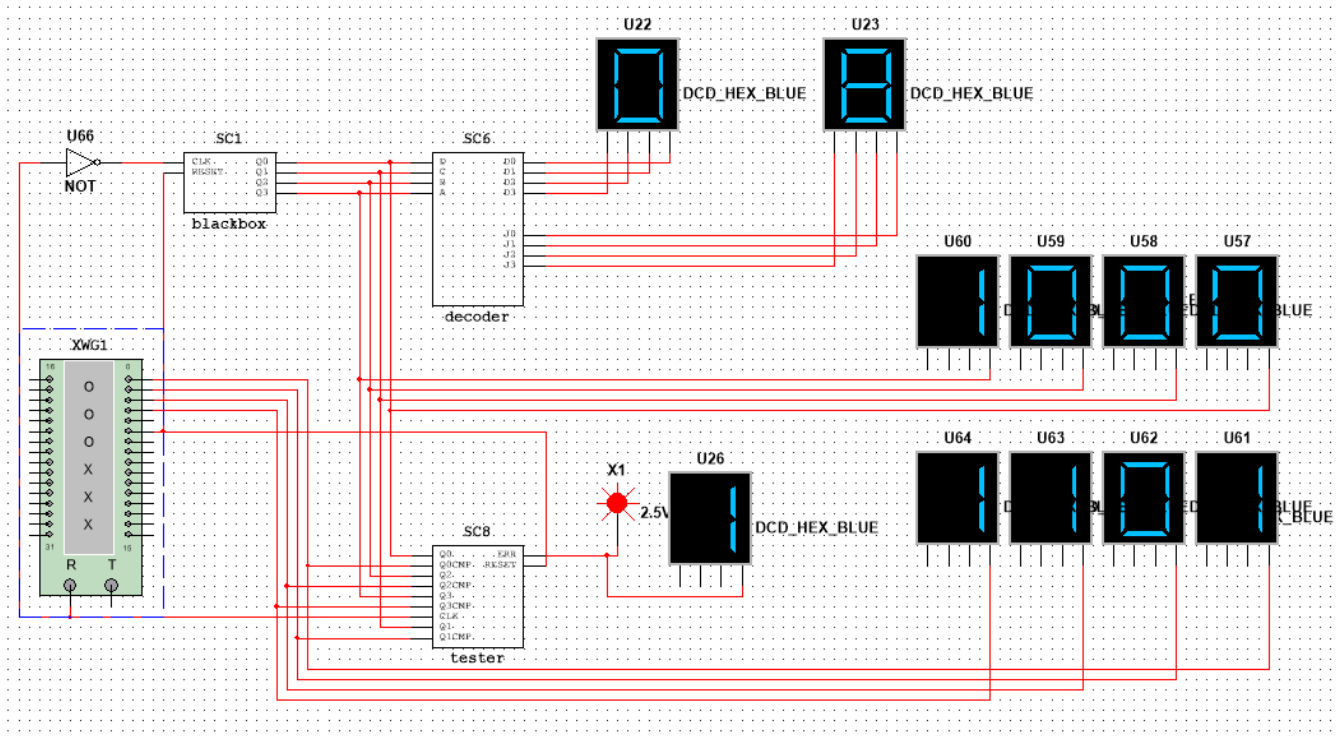
Rysunek 17: Schemat podukładu `decoder`

4 Stanowisko testujące

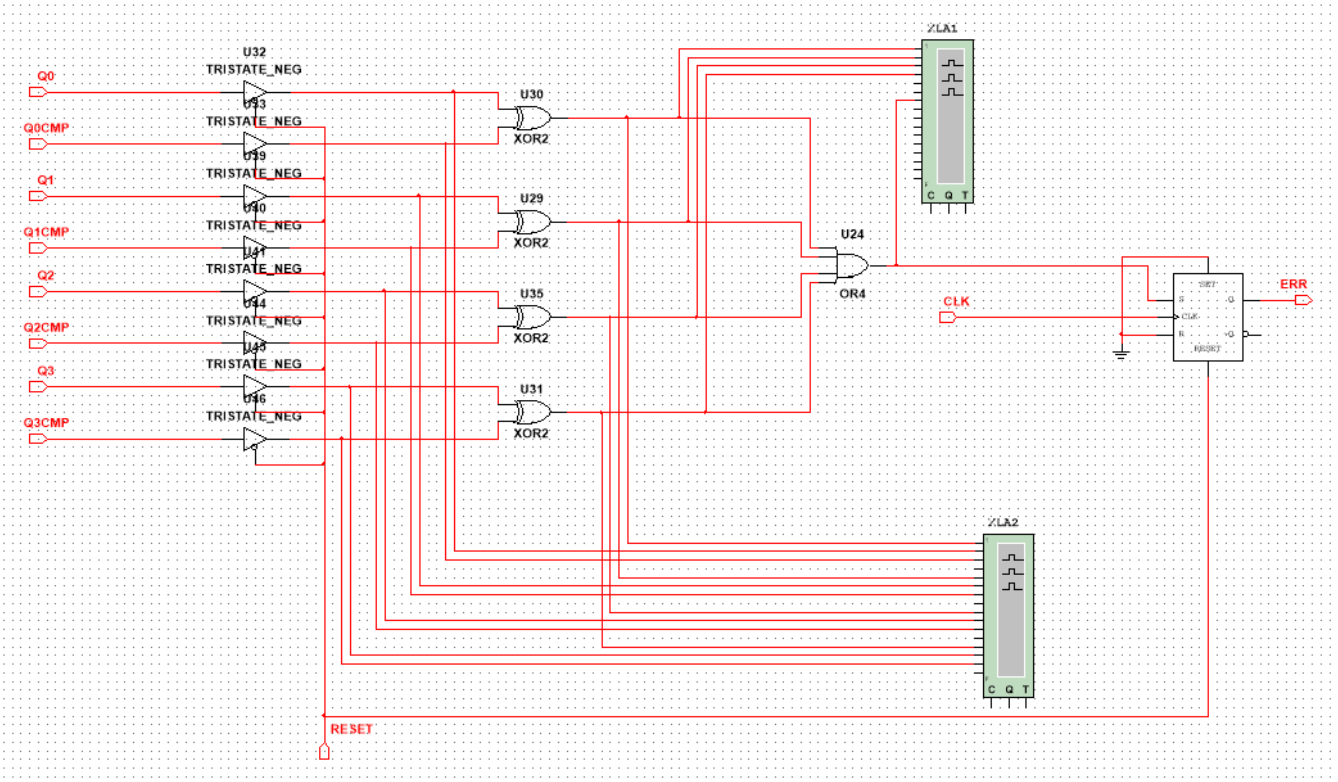
Głównym elementem układu testującego jest podukład `tester` porównujący sygnały wysyłane przez testowany licznik z sygnałami docelowymi nadawanymi równolegle przez generator słów (`XWG1`) i za pomocą przerzutnika SR wysyłający sygnał `ERR` z wartością 1 w przypadku niezgodności, co skutkuje zapaleniem diody i poinformowaniem użytkownika o błędzie.

Stanowisko umożliwia resetowanie stanu testowanego układu w dowolnym momencie w celach diagnostycznych, oraz badanie wewnętrzego stanu podukładu `tester` za pomocą analizatorów stanów logicznych (`XLA1` oraz `XLA2`). Cały układ podpięty jest do wspólnego zegara co gwarantuje poprawność wyników testów.

Wyjścia testowanego układu oraz generatora słów wyświetlane są na wyświetlaczach siedmiosegmentowych co umożliwia stałą kontrolę działania oraz szczegółowe badanie rozbieżności.



Rysunek 18: Podłączenie układu testującego



Rysunek 19: Podukład `tester`

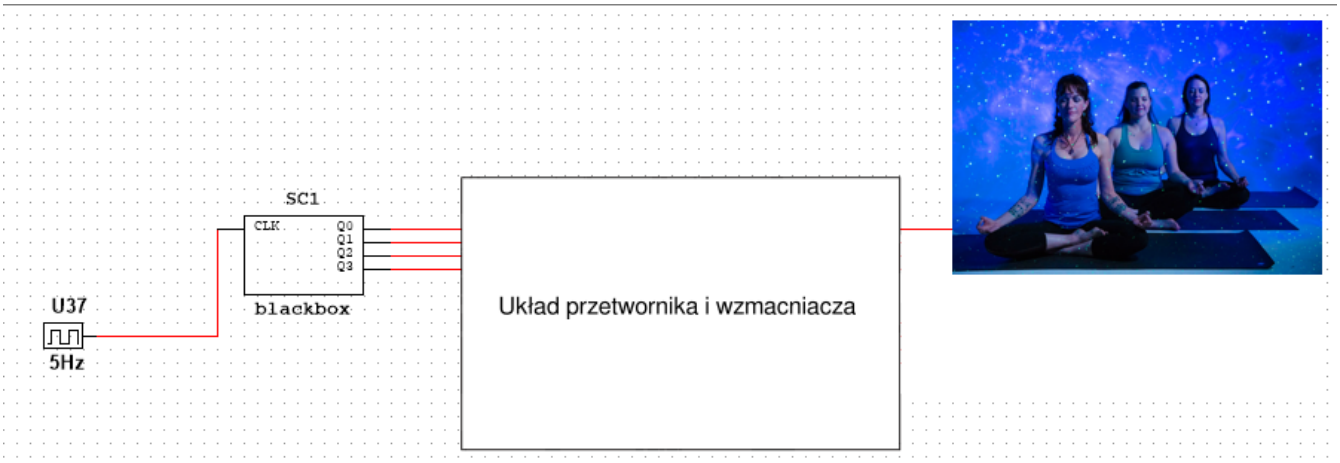
5 Podsumowanie oraz wnioski

Najważniejszym elementem projektu było poprawne zdefiniowanie funkcji sterujących przerzutnikami oraz wyjściem dekodera konwertującego stan licznika na wyświetlacz siedmiosegmentowy. Wybór rozwiązania opartego o przerzutniki typu T (chociaż nie jedyny możliwy, można było zastosować także chociażby prerzutniki typu JK) znacznie ułatwił ten

etap oraz uzasadnienie poprawności proponowanego rozwiązania. Bardziej standardowy podukład `decoder` również mógłby wyglądać zupełnie inaczej, gdyby zdecydowano się na inną metodę wyświetlania (a co za tym idzie konwersji), co jednak przełożyłoby się również na dodatkowe skomplikowanie układu.

Praktyczne zastosowania zaprojektowanego licznika to m.in.:

- Sterowanie animacjami w reklamach cyfrowych, np. dynamiczne zmiany wyświetlanych wzorów na billboardach LED, wykorzystujące niestandardowe sekwencje do przyciągnięcia uwagi.
- Moduły czasowe w systemach IoT do generowania nieregularnych interwałów sygnałowych, np. w celu redukcji kolizji pakietów w sieci.
- Generatory muzyczne
- Biomimetyczne systemy oświetleniowe



Rysunek 20: Przykładowe zastosowanie układu - oświetlenie o "naturalnym"wzroście intensywności

Wykonane zadanie jest dobrym ćwiczeniem z zakresu projektowania niestandardowych liczników opartych na logice kombinacyjnej oraz zastosowania i wyboru odpowiedniego rodzaju przerzutników w projektowanych układach.