

Czterobitowy licznik działający zgodnie z ciągiem Fibonacciego

Szymon Borusiewicz, Jakub Zając, Dawid Szłapa, Radosław Szepielak

Maj 2025

1 Treść zadania

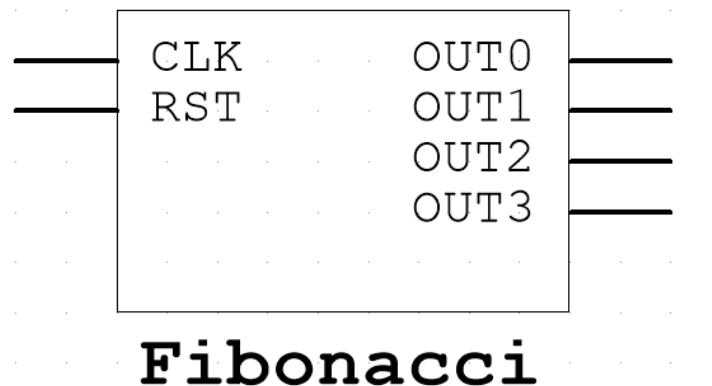
Korzystając tylko z konkretnego jednego typu przerzutników oraz z dowolnych bramek logicznych, proszę zaprojektować czterobitowy licznik działający zgodnie z ciągiem Fibonacciego (z nieobowiązkowym upraszczającym zastrzeżeniem, że wartość "1" powinna się pojawiać tylko raz w cyklu). Po uruchomieniu licznika, 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.

2 Czarna skrzynka

Układ, do którego dążymy, prezentuje się w następujący sposób:



Rysunek 1

Gdzie wejściami są:

1. CLK - sygnał z zewnętrznego zegara
2. RST - sygnał służący zresetowaniu stanu licznika
3. OUT0, OUT1, OUT2, OUT3 - wyjścia odpowiedzialne za kolejną liczbę

3 Tablica przejść automatu

R	D	C	B	A	Out3	Out2	Out1	Out0
X	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	1
1	0	0	0	1	0	0	1	0
X	0	0	1	0	0	0	1	1
X	0	0	1	1	0	1	0	1
X	0	1	0	1	1	0	0	0
X	1	0	0	0	1	1	0	1
X	1	1	0	1	0	0	0	0

4 Tablice Karnaugh

B, A, R

	000	001	011	010	110	111	101	100
D, C 00	1	1	0	1	1	1	1	1
01	-	-	0	0	-	-	-	-
11	-	-	0	0	-	-	-	-
10	1	1	-	-	-	-	-	-

R	D	C	B	A	Out3	Out2	Out1	Out0
X	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	1
1	0	0	0	1	0	0	1	0
X	0	0	1	0	0	0	1	1
X	0	0	1	1	0	1	0	1
X	0	1	0	1	1	0	0	0
X	1	0	0	0	1	1	0	1
X	1	1	0	1	0	0	0	0

Rysunek 2: Tablica Karnaugh dla wyjścia Out0

$$\overline{A} + B + \overline{C}R$$

B, A, R

	000	001	011	010	110	111	101	100
D, C 00	0	0	1	0	0	0	1	1
01	-	-	0	0	-	-	-	-
11	-	-	0	0	-	-	-	-
10	0	0	-	-	-	-	-	-

R	D	C	B	A	Out3	Out2	Out1	Out0
X	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	1
1	0	0	0	1	0	0	1	0
X	0	0	1	0	0	0	1	1
X	0	0	1	1	0	1	0	1
X	0	1	0	1	1	0	0	0
X	1	0	0	0	1	1	0	1
X	1	1	0	1	0	0	0	0

Rysunek 3: Tablica Karnaugh dla wyjścia Out1

$$\overline{A}B + \overline{A}B\overline{C}R$$

		B, A, R							
		000	001	011	010	110	111	101	100
D, C	00	0	0	0	0	1	1	0	0
	01	-	-	0	0	-	-	-	-
	11	-	-	0	0	-	-	-	-
	10	1	1	-	-	-	-	-	-

R	D	C	B	A	Out3	Out2	Out1	Out0
X	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	1
1	0	0	0	1	0	0	1	0
X	0	0	1	0	0	0	1	1
X	0	0	1	1	0	1	0	1
X	0	1	0	1	1	0	0	0
X	1	0	0	0	1	1	0	1
X	1	1	0	1	0	0	0	0

Rysunek 4: Tablica Karnaugh dla wyjścia Out2

$$AB + D\bar{C}$$

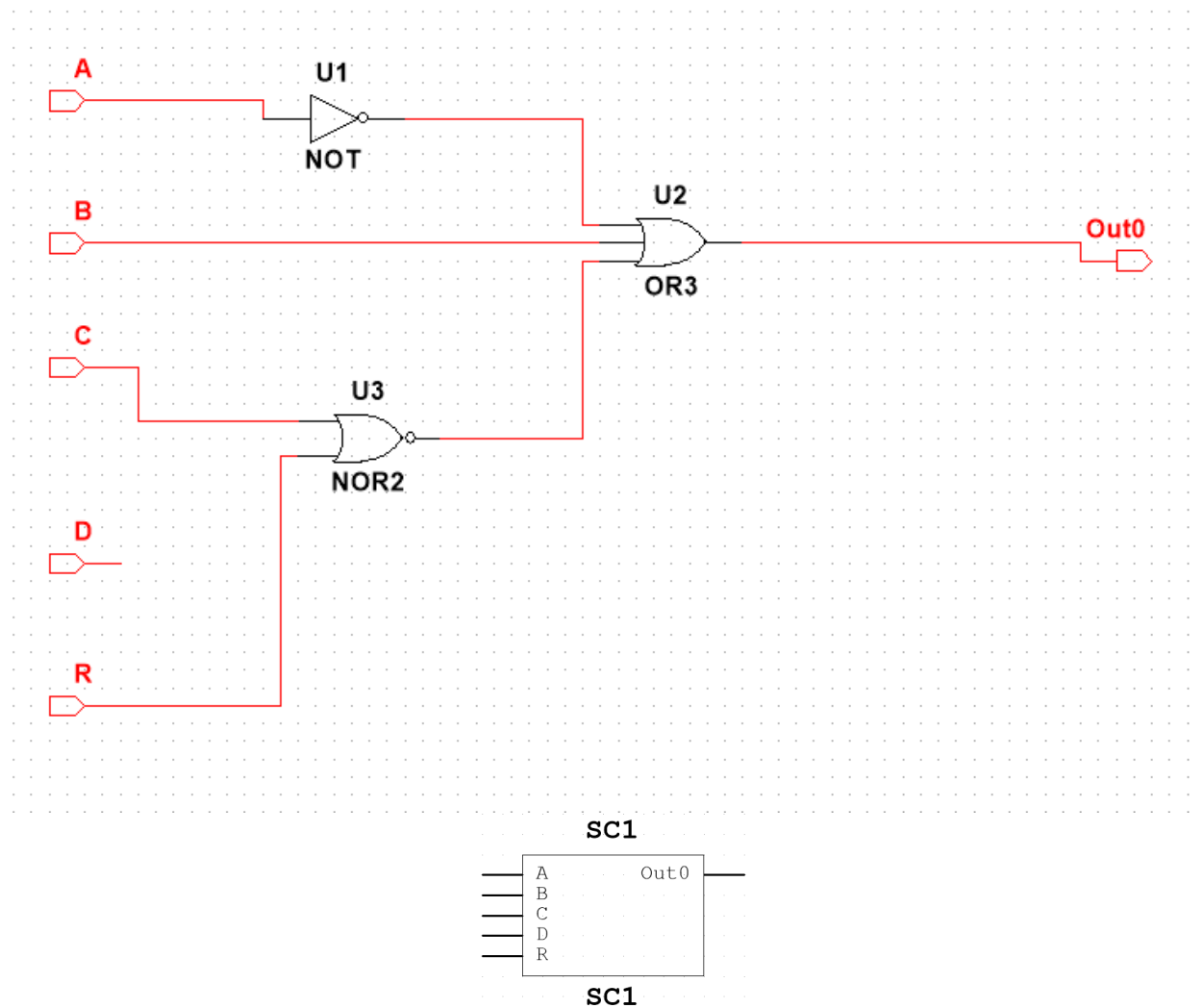
		B, A, R							
		000	001	011	010	110	111	101	100
D, C	00	0	0	0	0	0	0	0	0
	01	-	-	1	1	-	-	-	-
	11	-	-	0	0	-	-	-	-
	10	1	1	-	-	-	-	-	-

R	D	C	B	A	Out3	Out2	Out1	Out0
X	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	1
1	0	0	0	1	0	0	1	0
X	0	0	1	0	0	0	1	1
X	0	0	1	1	0	1	0	1
X	0	1	0	1	1	0	0	0
X	1	0	0	0	1	1	0	1
X	1	1	0	1	0	0	0	0

Rysunek 5: Tablica Karnaugh dla wyjścia Out3

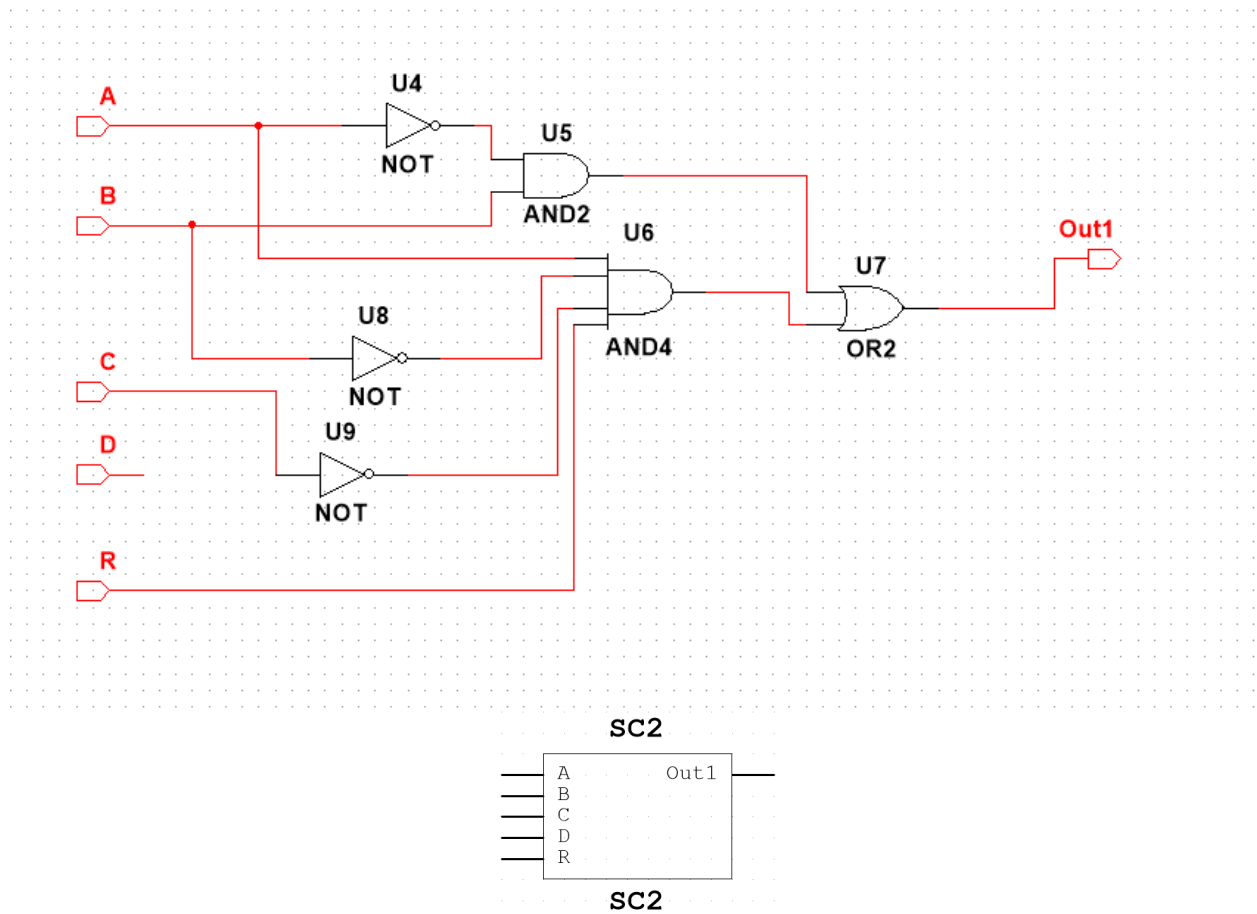
$$\bar{C}D + C\bar{D}$$

5 Implementacja w programie Multisim



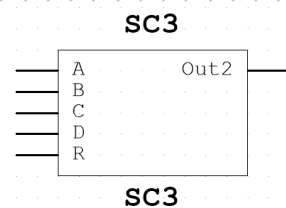
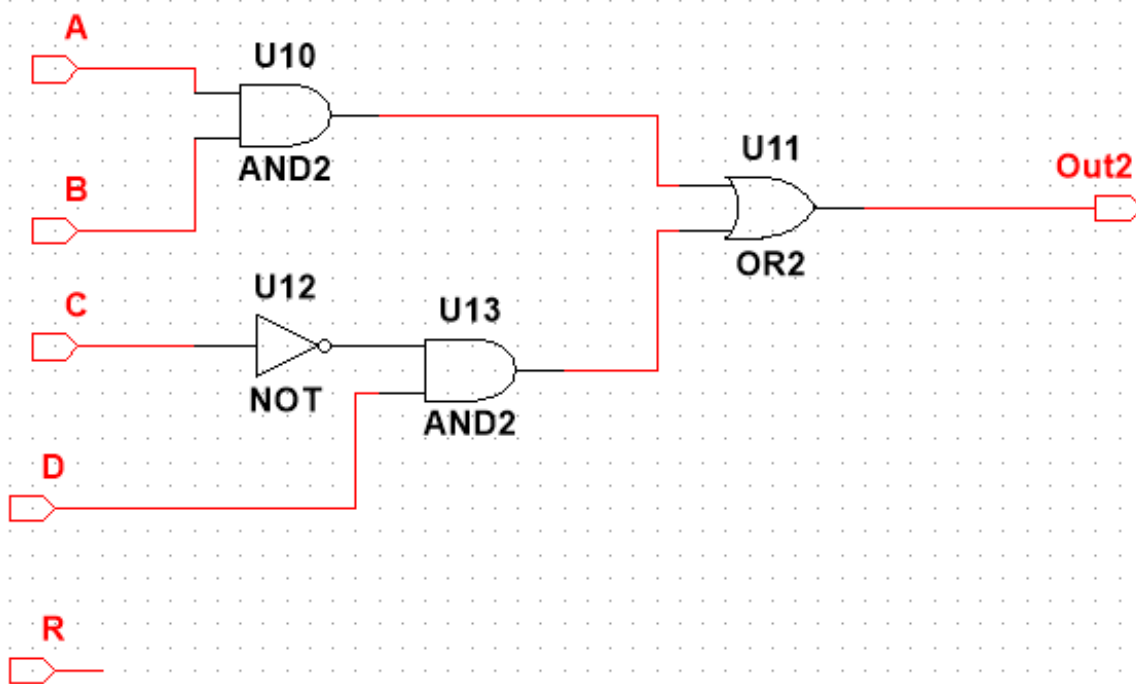
Rysunek 6: Implementacja podukładu SC1

$$\bar{A} + B + \bar{C}R$$



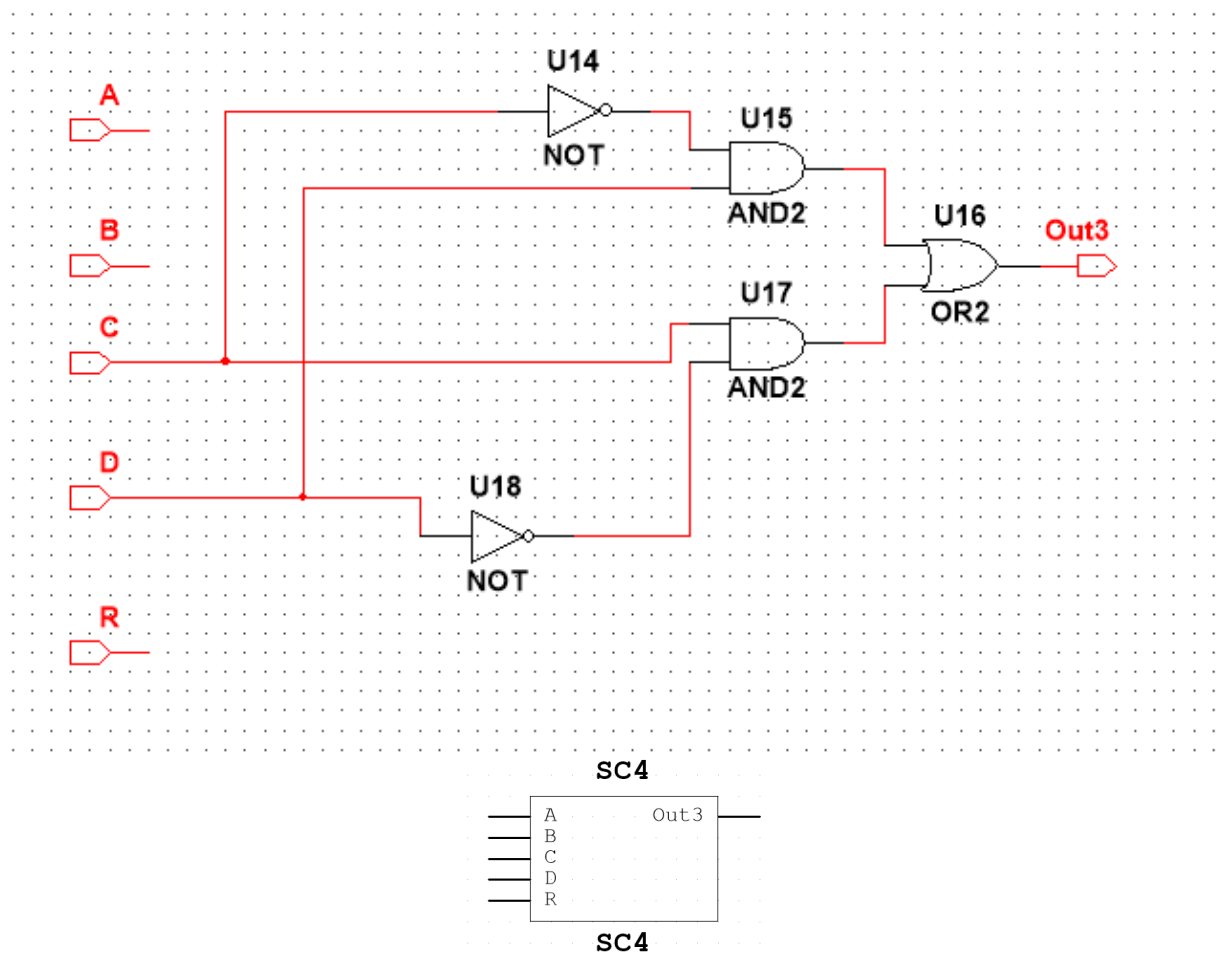
Rysunek 7: Implementacja podukładu SC2

$$\overline{A}B + A\overline{B}C\overline{D}R$$

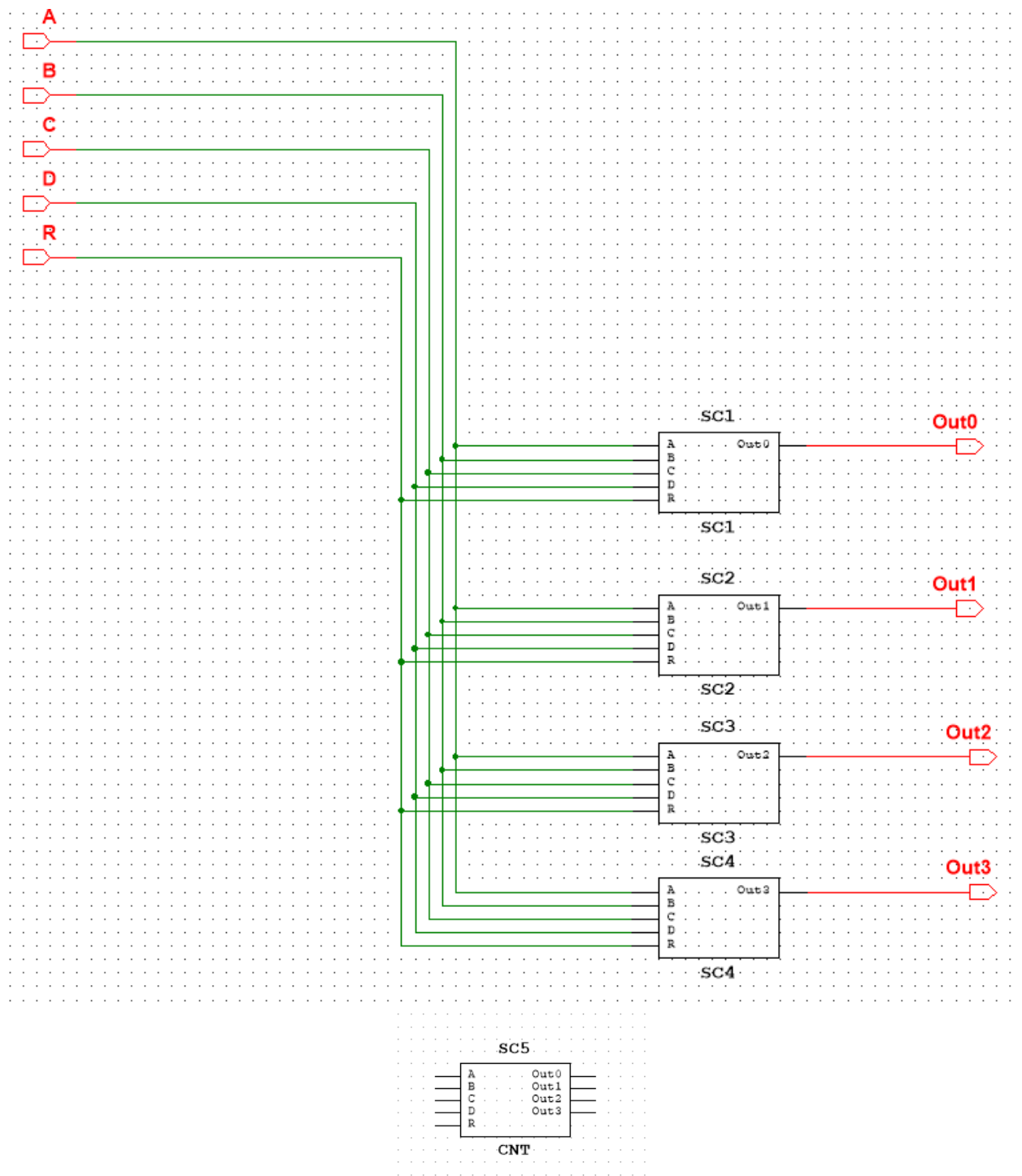


Rysunek 8: Implementacja podukładu SC3

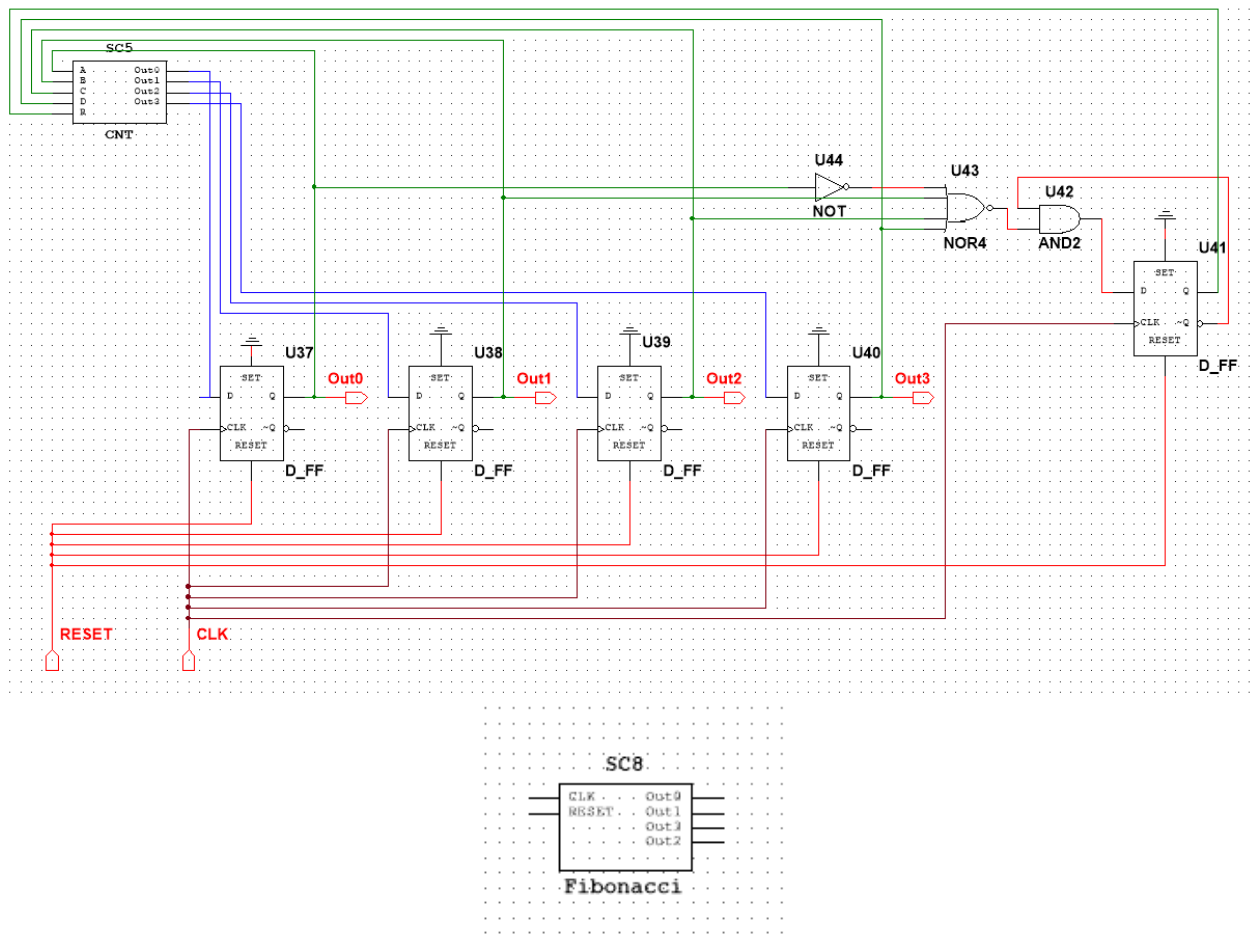
$$AB + D\bar{C}$$



Rysunek 9: Implementacja podukładu SC4
 $\overline{C}D + C\overline{D}$



Rysunek 10: Implementacja podukładu CNT



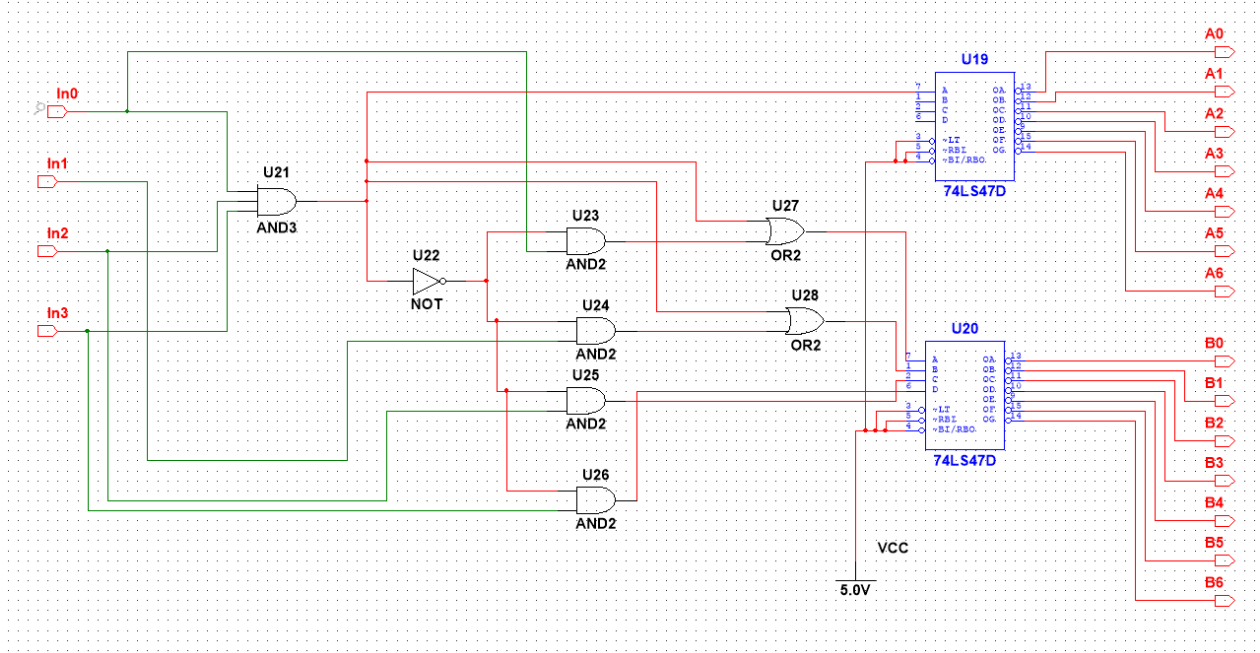
Rysunek 11: Implementacja podukładu Fibonacci

D	C	B	A	Q	D_o
0	0	0	0	X	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	X	0
0	0	1	1	X	0
0	1	0	1	X	0
1	0	0	0	X	0
1	1	0	1	X	0

Tabela 1: Tabela prawdy dla układu bramek U44, U43, U42

CLK	D	SET	RESET	Q_{n+1}
↑	0	0	0	0
↑	1	0	0	1
X	X	0	0	Q_n
X	X	1	0	1
X	X	0	1	0
X	X	1	1	X

Tabela 2: Tabela prawdy dla przerzutnika D



Rysunek 12: Implementacja transkodera

In3	In2	In1	In0	A1	B1	C1	D1	A2
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0
0	0	1	0	0	0	1	0	0
0	0	1	1	0	0	1	1	0
0	1	0	1	0	1	0	1	0
1	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	1	1

Tabela 3: Tabela prawdy dla układu bramek U21, U22... U28

Wejścia układu U20 to A1, B1, C1, D1, natomiast wejście układu U19 to A2

$$A_1 = \overline{\text{In0}} \cdot \overline{\text{In2}} \cdot \overline{\text{In3}} \cdot \text{In0} + \text{In0} \cdot \text{In2} \cdot \text{In3}$$

$$B_1 = \overline{\text{In0}} \cdot \overline{\text{In2}} \cdot \overline{\text{In3}} \cdot \text{In1} + \text{In0} \cdot \text{In2} \cdot \text{In3}$$

$$C_1 = \overline{\text{In0}} \cdot \overline{\text{In2}} \cdot \overline{\text{In3}} \cdot \text{In2}$$

$$D_1 = \overline{\text{In0}} \cdot \overline{\text{In2}} \cdot \overline{\text{In3}} \cdot \text{In3}$$

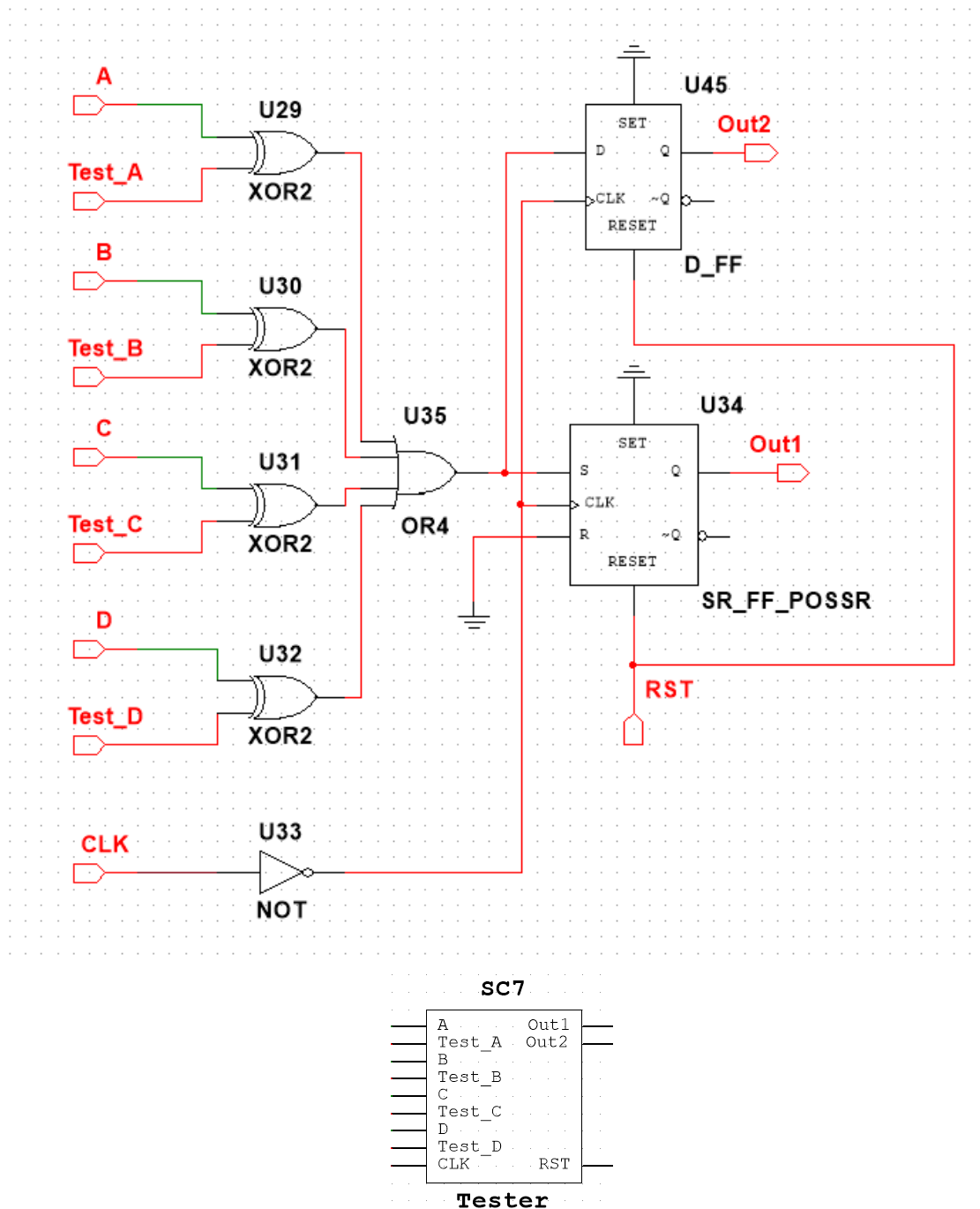
$$A_2 = \text{In0} \cdot \text{In2} \cdot \text{In3}$$

In3	In2	In1	In0	A0	A1	A2	A3	A4	A5	A6	B0	B1	B2	B3	B4	B5	B6
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1
0	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	0	1
0	0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	0
0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	0	1	0
1	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0
1	1	0	1	0	1	1	1	1	0	1	1	0	0	0	0	1	0

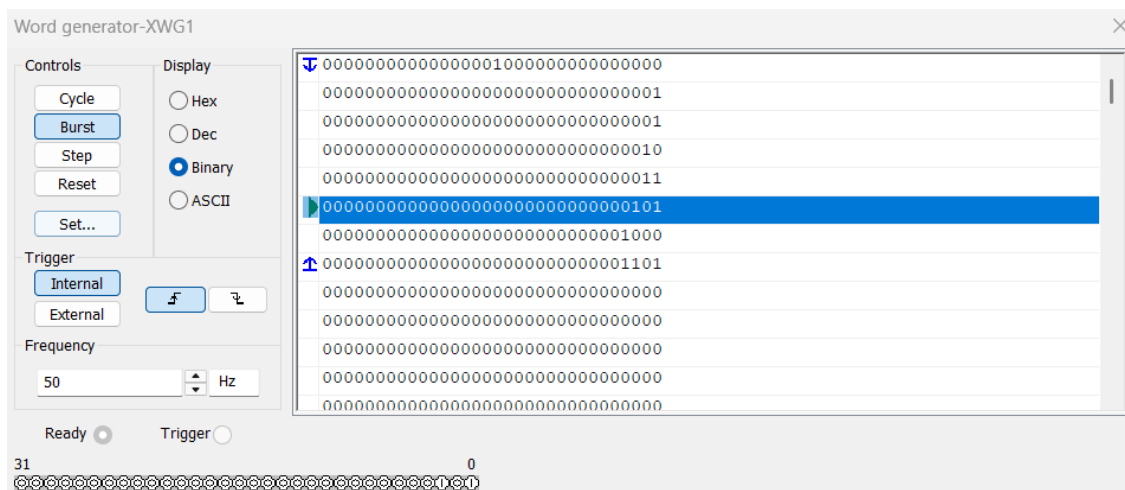
Tabela 4: Tabela prawdy dla całego transkodera

6 Układ testujący

Układ testujący analizuje dane wyjściowe pochodzące z naszego układu i porównuje je z przewidywanymi kolejnymi liczbami, które powinny się pojawić w naszym wyświetlaczu 7-segmentowym.



Rysunek 13: Schemat układu testującego



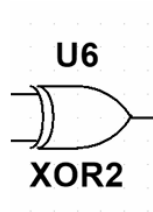
Rysunek 14: Schemat układu testującego

W generatorze słowa są postaci: 0000000000000000

- 0 – Bit resetujący
- 000000000000 – Bity nieużywane
- 0000 – Bity testujące poprawność

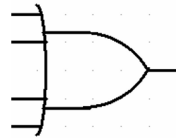
Układ testujący najpierw porównuje bit otrzymany z podukładu Fibonacci z bitem przewidywanym przez XWG-1. Wykorzystaliśmy tutaj bramki XOR ($\sim (A \Leftrightarrow B)$).

A	B	O
0	0	0
0	1	1
1	0	1
1	1	0



Rysunek 15: Tabela prawdy dla bramki XOR

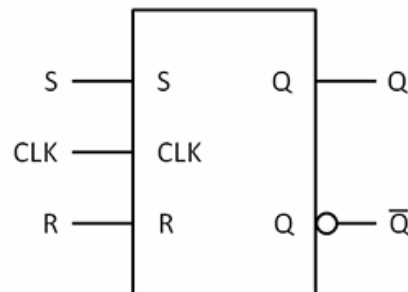
Następnie wyjście z bramek XOR przekazujemy do bramki OR:



A	B	C	D	O
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Tabela 5: Tabela prawdy i symbol dla bramki 4-wejściowej OR

Sygnal błędu jeśli błąd wystąpił w jakimkolwiek teście rejestrowany jest przez przerzutnik synchroniczny RS (dodatkowe wejście CLK) i zapalana jest dioda X1.

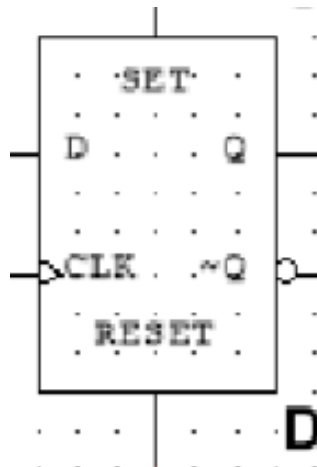


Rysunek 16: Przerzutnik RS

CLK	S	R	Q_{n+1}	$\neg Q_{n+1}$
↑	0	0	Q_n	$\neg Q_n$
↑	0	1	0	1
↑	1	0	1	0
↑	1	1	X	X
X	X	X	Q_n	$\neg Q_n$

Tabela 6: Tabela prawdy przerzutnika RS

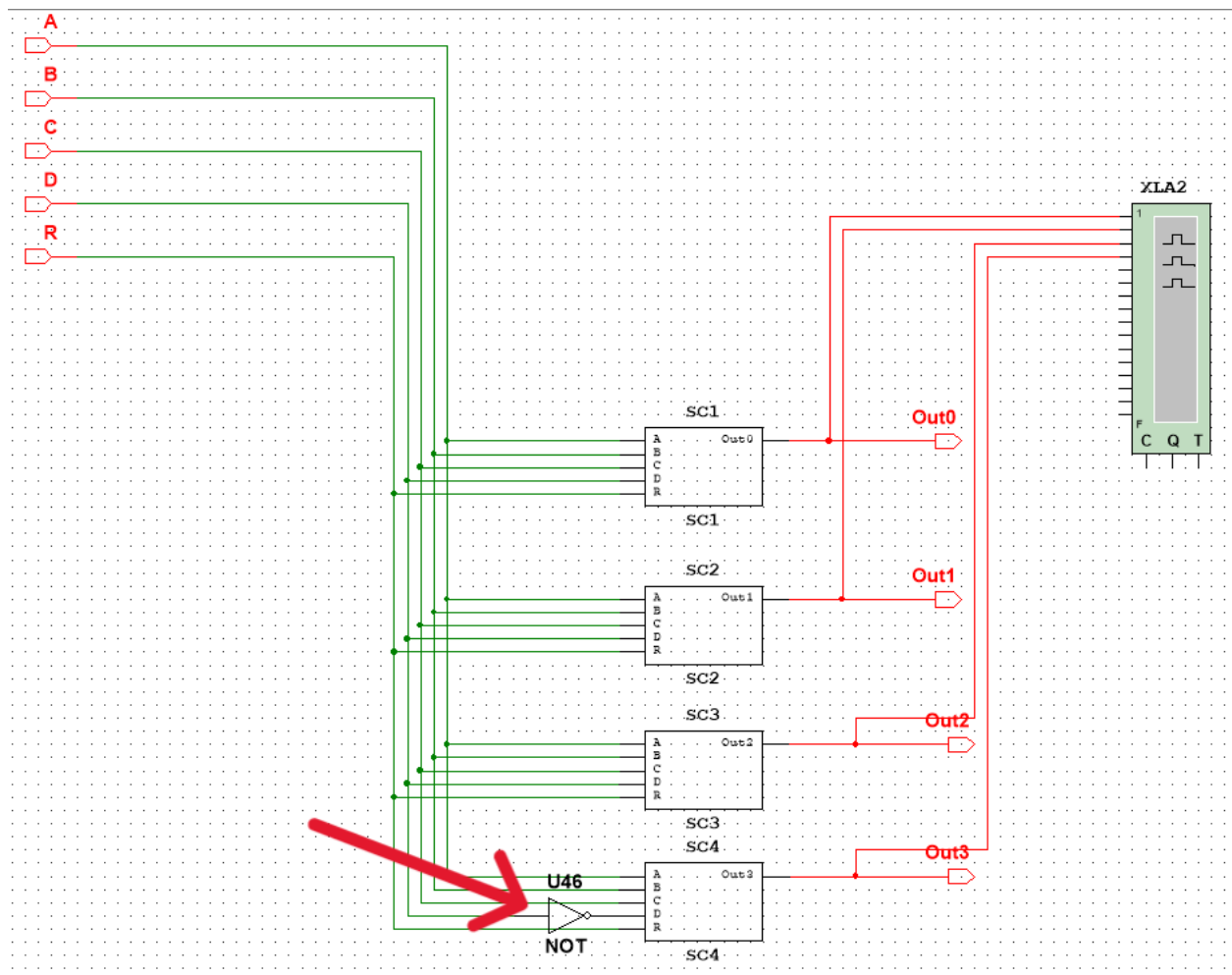
Natomiast sygnał błędu jeśli błąd wystąpił w danym teście rejestrowany jest przez przerzutnik synchroniczny D i zapalana jest dioda X2:



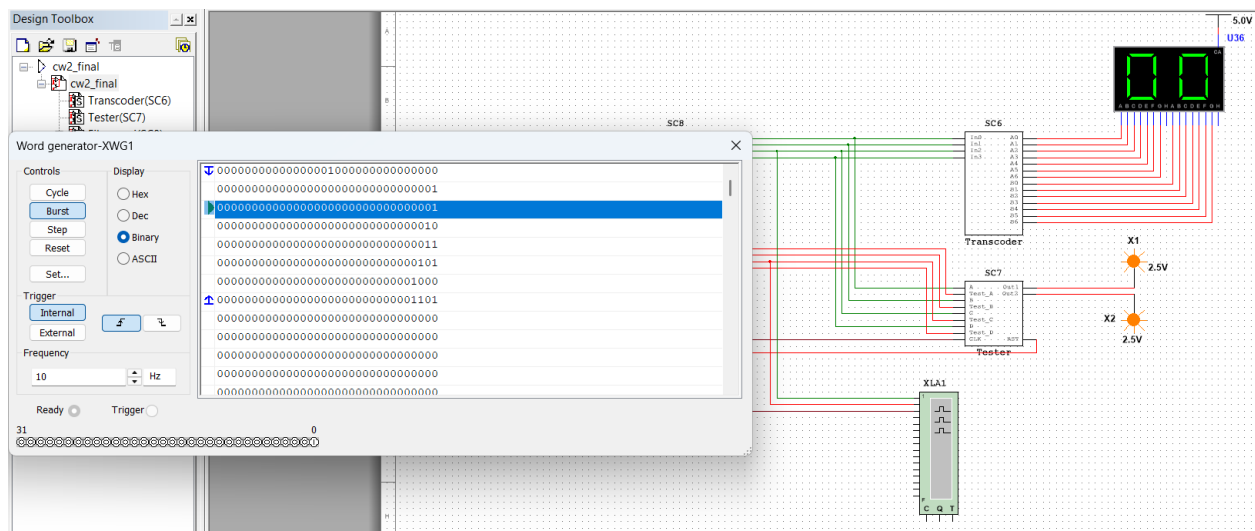
Rysunek 17: Przerzutnik D

CLK	D	SET	RESET	Q_{n+1}
\uparrow	0	0	0	0
\uparrow	1	0	0	1
X	X	0	0	Q_n
X	X	1	0	1
X	X	0	1	0
X	X	1	1	X

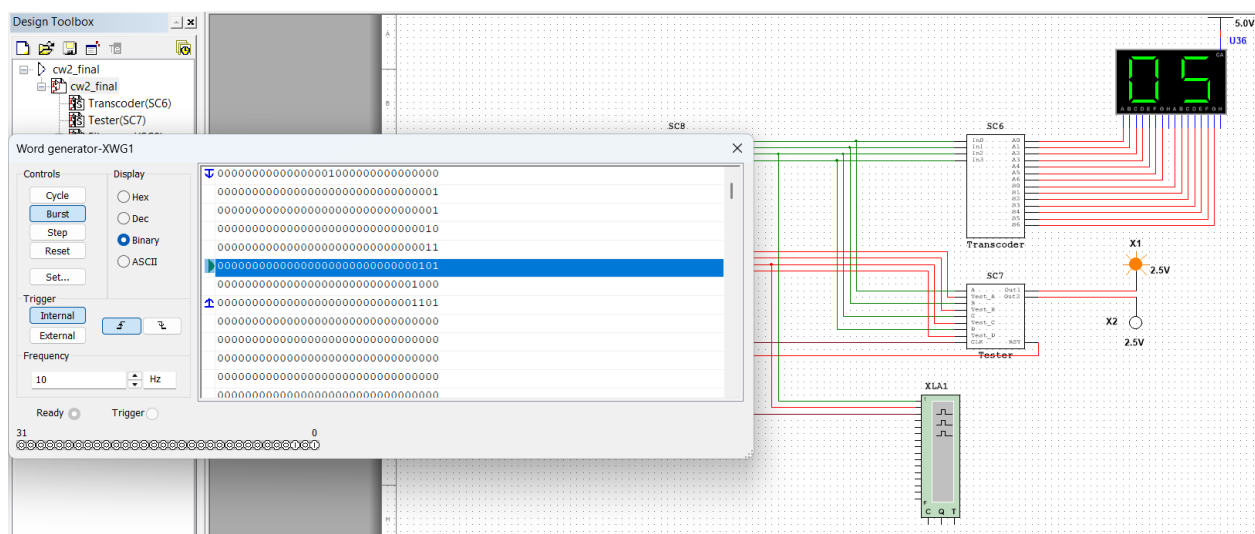
Tabela 7: Tabela prawdy dla przerzutnika D



Rysunek 18: Uszkodzony układ

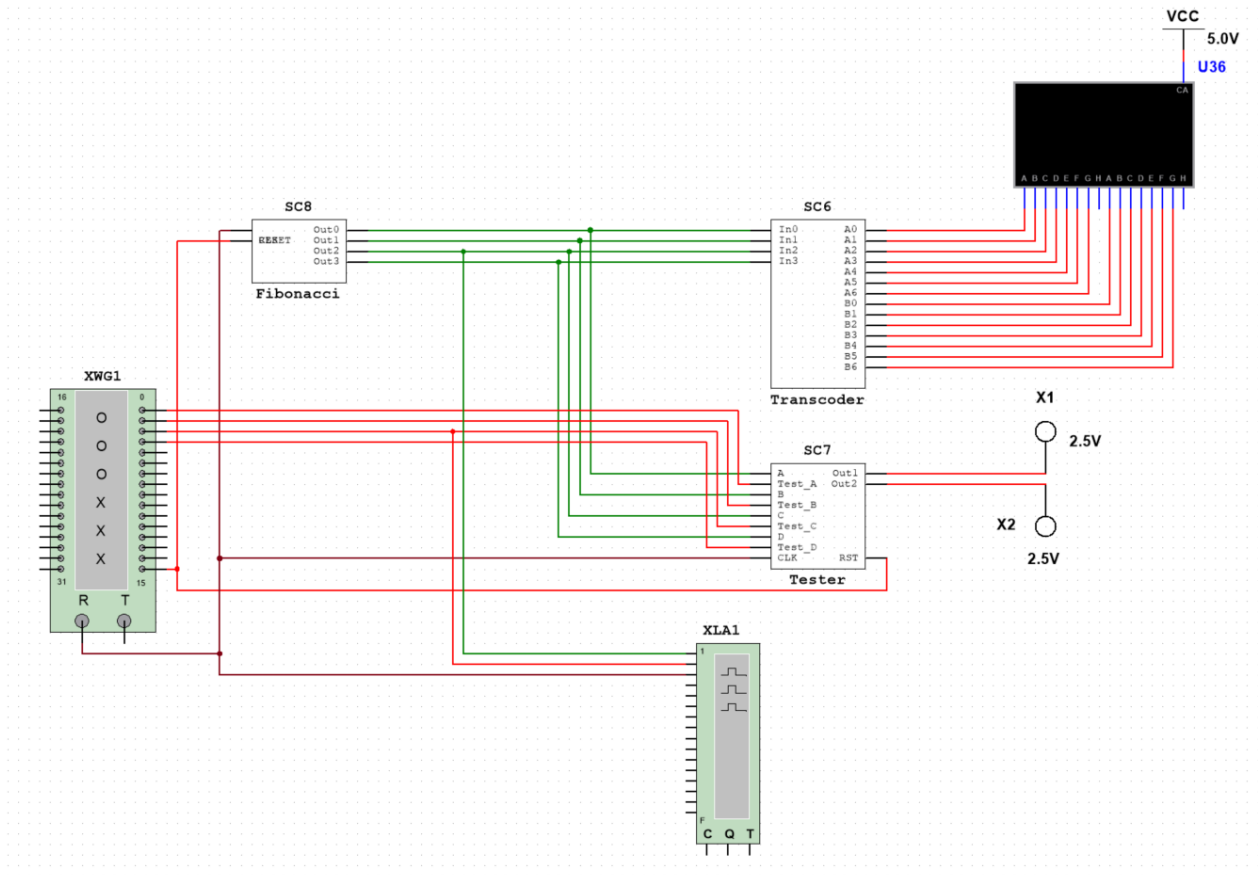


Rysunek 19: Testowanie uszkodzonego układu 1

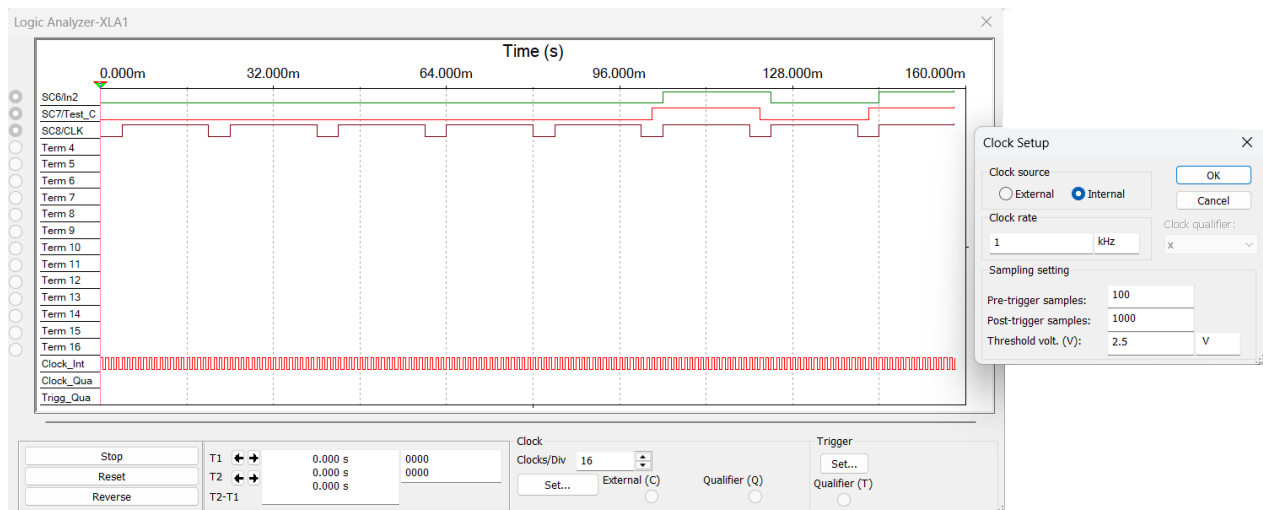


Rysunek 20: Testowanie uszkodzonego układu 1

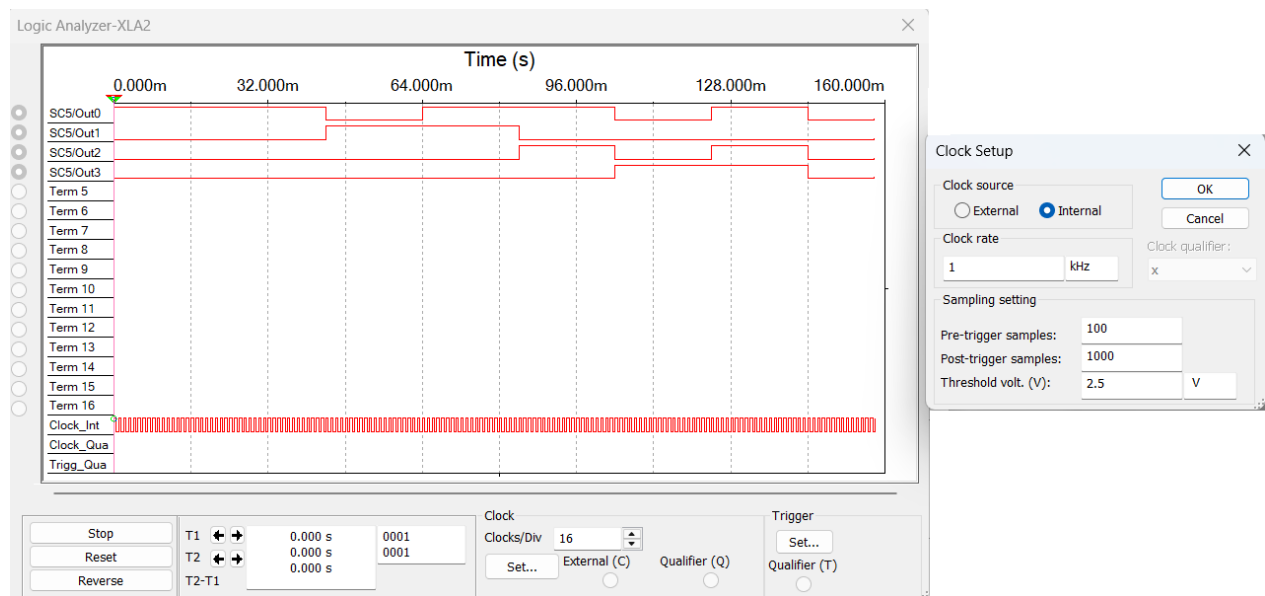
7 Schemat całego układu



Rysunek 21: Schemat całego układu



Rysunek 22: Wykres analizatora XLA1



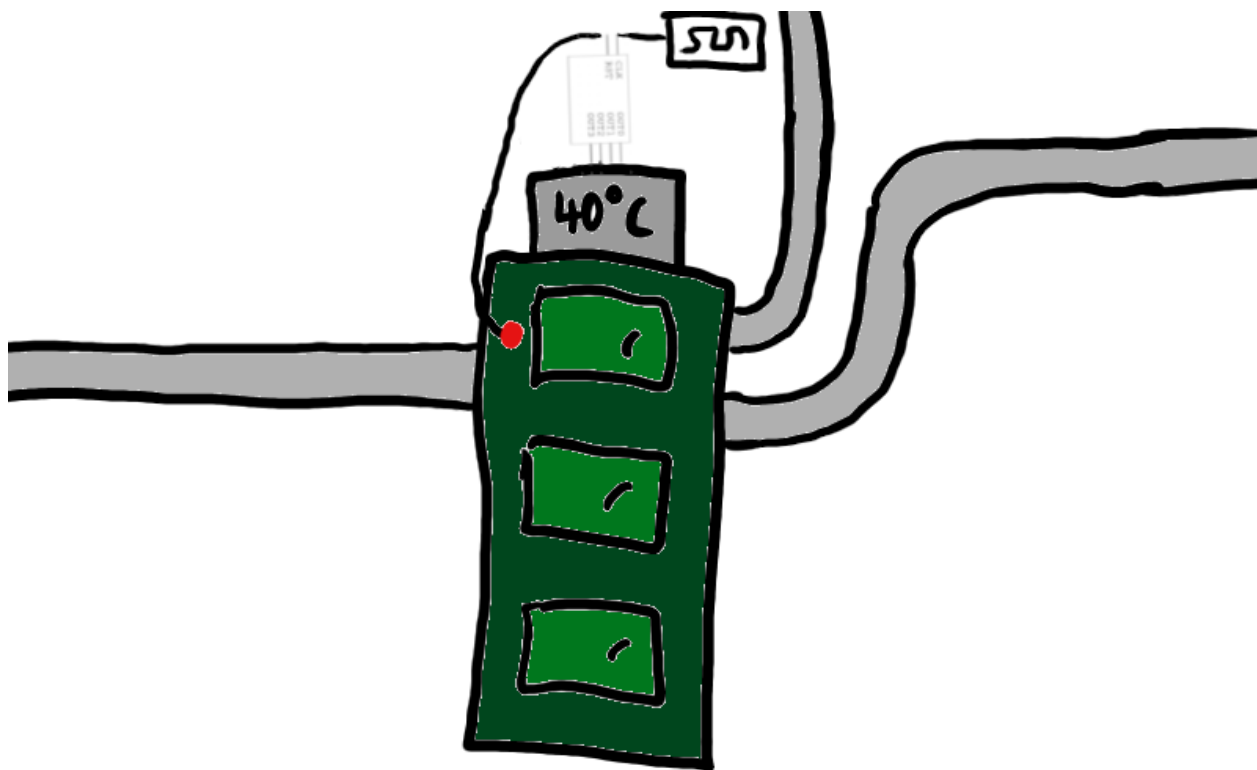
Rysunek 23: Wykres analizatora XLA2 dla nieuszkodzonego podukładu CNT

8 Przykłady zastosowania

Poniżej znajdują się przykładowe zastosowania naszego układu.

8.1 Automatyczny termostat

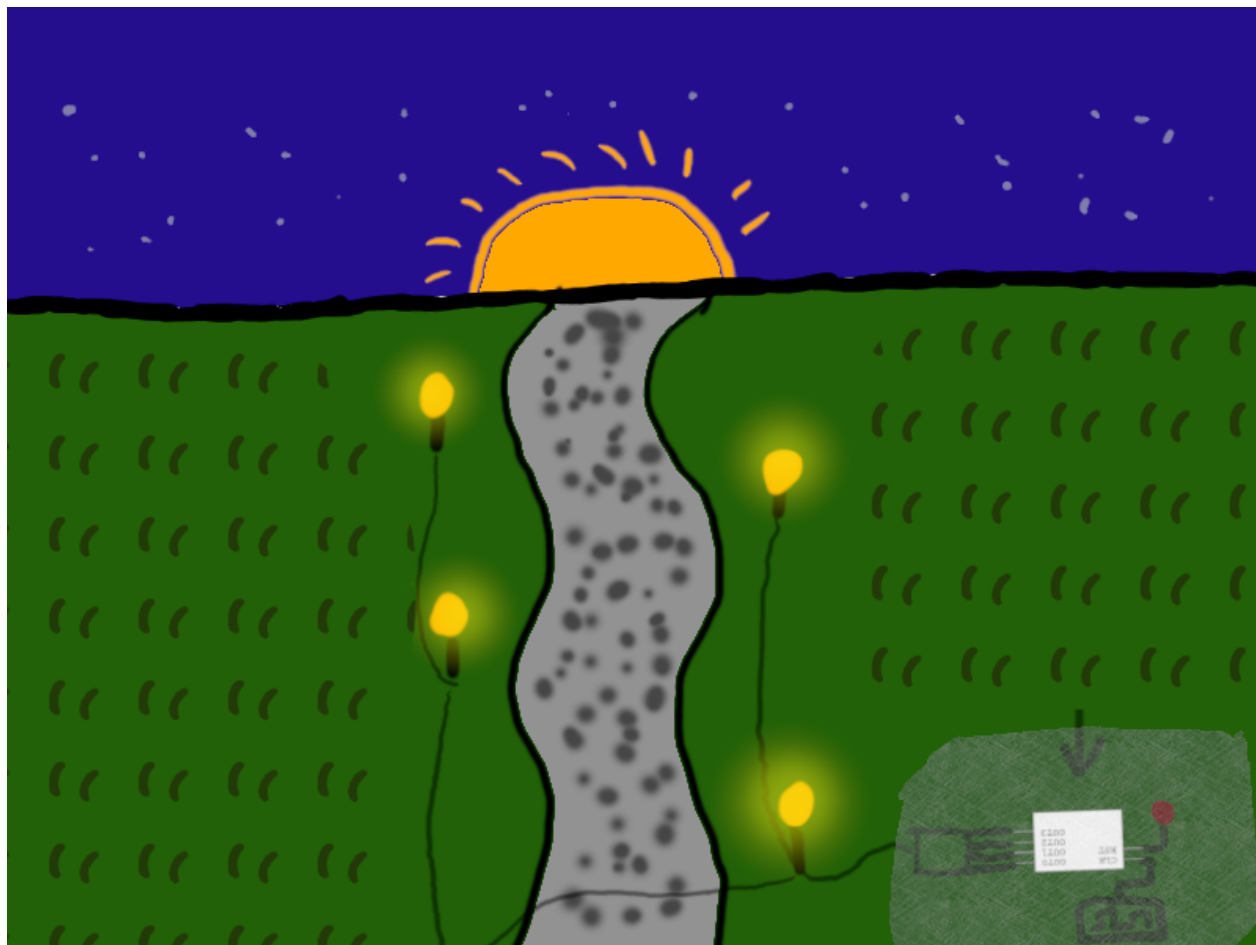
Układ możemy zastosować do sterowania temperaturą panującą w domu w ciągu dnia. W nocy oraz gdy nikogo nie ma w domu temperatura może być niższa, natomiast przed wstawaniem oraz w ciągu dnia temperatura może zostać podwyższona w celu zapewnienia domownikom komfortu.



Rysunek 24: Termostat

8.2 Oświetlenie przydomowe

Możemy zastosować układ do sterowania oświetleniem obok domu dostosowując je do wschodów i zachodów słońca. W ten sposób możemy zapewnić nastrojowe oświetlenie w ciągu wieczoru, minimalne w ciągu nocy, a w ciągu dnia zrezygnować z niego całkowicie.



Rysunek 25: Oświetlenie przydomowe

9 Wnioski

W trakcie realizacji projektu wykorzystanie narzędzi do analizy tablic Karnaugh znacząco ułatwiło optymalizację funkcji logicznych.

Zdecydowaliśmy się wprowadzić modyfikację polegającą na rozszerzeniu zbioru wejść — poprzez dodanie jednej z jedynek do ciągu — co rozszerzyło liczbę wymaganych wejść do pięciu.

W projekcie kluczową rolę odegrały przerzutniki typu D oraz RS, jednak można go przekształcić do realizacji za pomocą samych bramek logicznych co upraszczałoby układ bez zmiany jego funkcjonalności jednak znacząco zmniejszając jego czytelność i przejrzystość. Zaletą takiego podejścia byłaby możliwość fizycznego odwzorowania układu z użyciem popularnych komponentów, co zwiększa jego dostępność i praktyczność.

Co można było zrobić inaczej?

Przy projektowaniu układu testującego zaczęliśmy od zaprojektowania go z użyciem jednego przerzutnika typu D, jednak dodaliśmy jeszcze przerzutnik RS aby osobno skutecznie reprezentować sytuację wystąpienia błędu w danym kroku zegara oraz wystąpienia go kiedykolwiek wcześniej.