Projekt czterobitowego licznika Fibonacciego

Bartosz Kaganiec
Julia Demitraszek
Adrian Krawczyk
Damian Chłus

14 maja 2025

Spis treści

1 Opis zadania projektowego

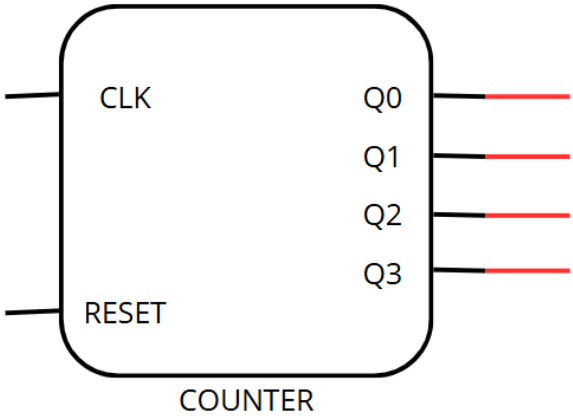
Celem zadania jest zaprojektowanie czterobitowego licznika działającego zgodnie z ciągiem Fibonacciego:

$$0 \to 1 \to 2 \to 3 \to 5 \to 8 \to 13 \to 0 \to \ldots$$

Licznik będzie wykorzystywał jeden typ przerzutnika (w tym projekcie D) oraz dowolne bramki logiczne. Przejścia między stanami przedstawiono w postaci kodów binarnych:

$$0000 \to 0001 \to 0010 \to 0011 \to 0101 \to 1000 \to 1101 \to 0000$$

Można przedstawić ogólny schemat układu jako:



Rysunek 1: Bardzo ogólne przedstawienie układu

Każdy stan licznika reprezentuje jedną liczbę ciągu, a jego wartość wyświetlana jest na wyświetlaczu siedmiosegmentowym za pośrednictwem dekodera. Funkcje wejściowe przerzutników wyznaczane są na podstawie poniższej sekwencji (Tabela 1), a ich realizacja może być zoptymalizowana za pomocą map Karnaugha.

2 Tabele Karnaugha i Funkcje Logiczne

2.1 Tabela Przejść Stanów

Podstawą do wyznaczenia funkcji logicznych dla układu jest tabela przejść stanów.

Tabela 1: Tabela przejść stanów dla licznika Fibonacciego

Stan bieżący (T_m)		Wyjścia				Następny stan				Stan następny (T_{m+1})	
Dec	Bin	Q_3	Q_2	Q_1	Q_0	Q'_3	Q'_2	Q'_1	Q'_0	Bin	Dec
0	0000	0	0	0	0	0	0	0	1	0001	1
1	0001	0	0	0	1	0	0	1	0	0010	2
2	0010	0	0	1	0	0	0	1	1	0011	3
3	0011	0	0	1	1	0	1	0	1	0101	5
5	0101	0	1	0	1	1	0	0	0	1000	8
8	1000	1	0	0	0	1	1	0	1	1101	13
13	1101	1	1	0	1	0	0	0	0	0000	0

2.2 Mapy Karnaugha, Minimalizacja i Realizacja Funkcji

Poniżej przedstawiono mapy Karnaugha dla funkcji logicznych wejść przerzutników D: $D_0 = Q'_0, D_1 = Q'_1, D_2 = Q'_2, D_3 = Q'_3$.

Dla każdego sygnału pokazano dwie mapy: początkową z zaznaczonymi wymaganymi wartościami '1' i '0' (zgodnie z Tabelą 1) oraz warunkami obojętnymi ('X') dla stanów nieużywanych, oraz finalną z przypisanymi wartościami '0' lub '1' do 'X' w celu uzyskania minimalnej funkcji logicznej.

Kolorem czerwonym oznaczono tło grupy $\overline{Q}_0$, kolorem zielonym tło grupy Q_1, a kolorem pomarańczowym tło pól należących do obu grup jednocześnie. Pod każdą parą map znajduje się zminimalizowana funkcja logiczna oraz schemat jej realizacji za pomocą bramek.

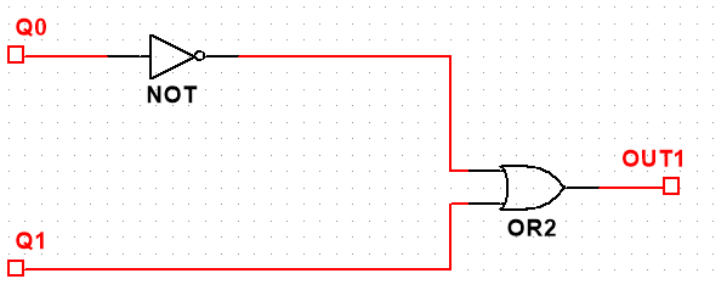
Tabela dla wyjścia Q'0

Q_1Q_0 / Q_3Q_2	00	01	11	10
00	1	0	1	1
01	X	0	X	X
11	X	0	X	X
10	1	X	X	X

$\Longrightarrow$

Q_1Q_0 / Q_3Q_2	00	01	11	10
00	1	0	1	1
01	1	0	1	1
11	1	0	1	1
10	1	0	1	1

$$Q'0 = \overline{Q}_0 + Q_1$$



Rysunek 2: Schemat logiczny realizujący funkcję $Q'0$

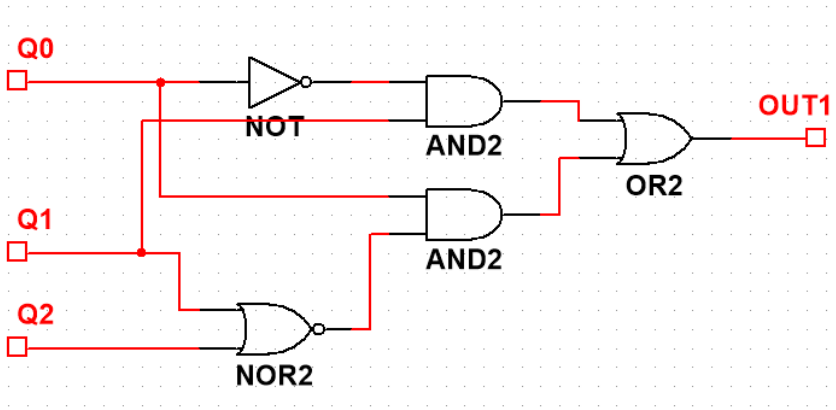
Tabela dla wyjścia Q'1

Q_1Q_0 / Q_3Q2	00	01	11	10
00	0	1	0	1
01	X	0	X	X
11	X	0	X	X
10	0	X	X	X

$\Longrightarrow$

Q_1Q_0 / Q_3Q2	00	01	11	10
00	0	1	0	1
01	0	0	0	1
11	0	0	0	1
10	0	1	0	1

$$Q'1 = \overline{Q}_2\overline{Q}_1Q_0 + Q_1\overline{Q}_0 = \overline{(Q_2 + Q_1)} \cdot Q_0 + Q_1 \cdot \overline{Q}_0$$



Rysunek 3: Schemat logiczny realizujący funkcję $Q'1$

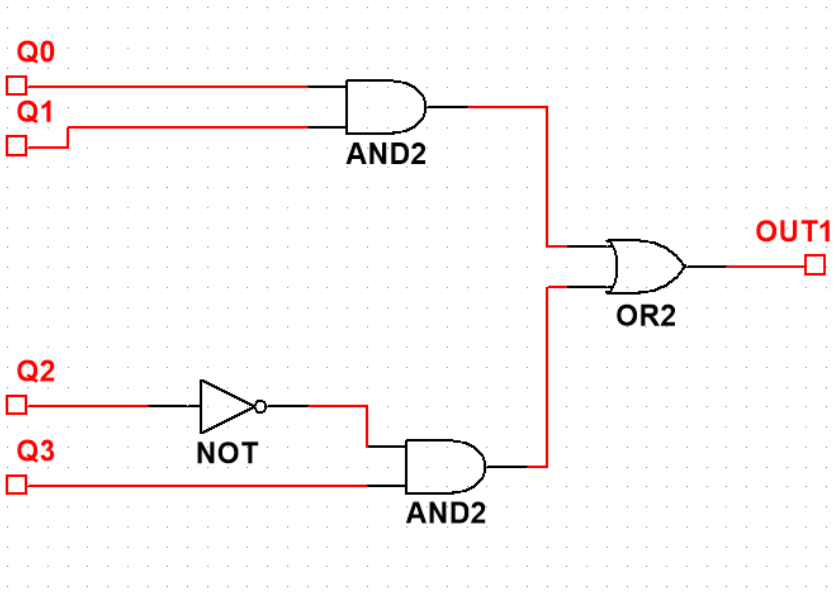
Tabela dla wyjścia Q'2

Q_1Q_0 / Q_3Q2	00	01	11	10
00	0	0	1	0
01	X	0	X	X
11	X	0	X	X
10	1	X	X	X

$\Longrightarrow$

Q_1Q_0 / Q_3Q2	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	0	1	0
10	1	1	1	1

$$Q'2 = Q_3\overline{Q}_2 + Q_1Q_0$$



Rysunek 4: Schemat logiczny realizujący funkcję $Q'2$

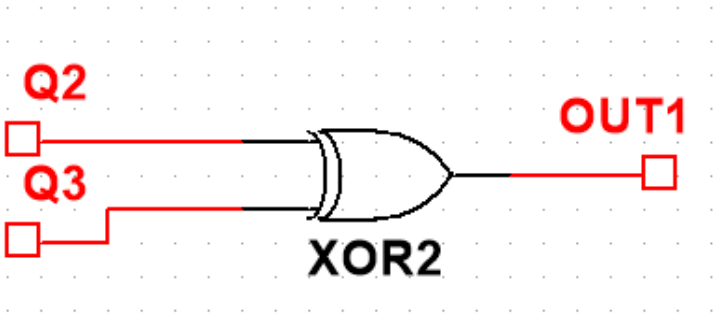
Tabela dla wyjścia Q'3

Q_1Q_0 / Q_3Q2	00	01	11	10
00	0	0	0	0
01	X	1	X	X
11	X	0	X	X
10	1	X	X	X

$\Longrightarrow$

Q_1Q_0 / Q_3Q2	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$$Q'_3 = Q_3\overline{Q}_2 + \overline{Q}_3Q_2 = Q_3 \oplus Q_2$$

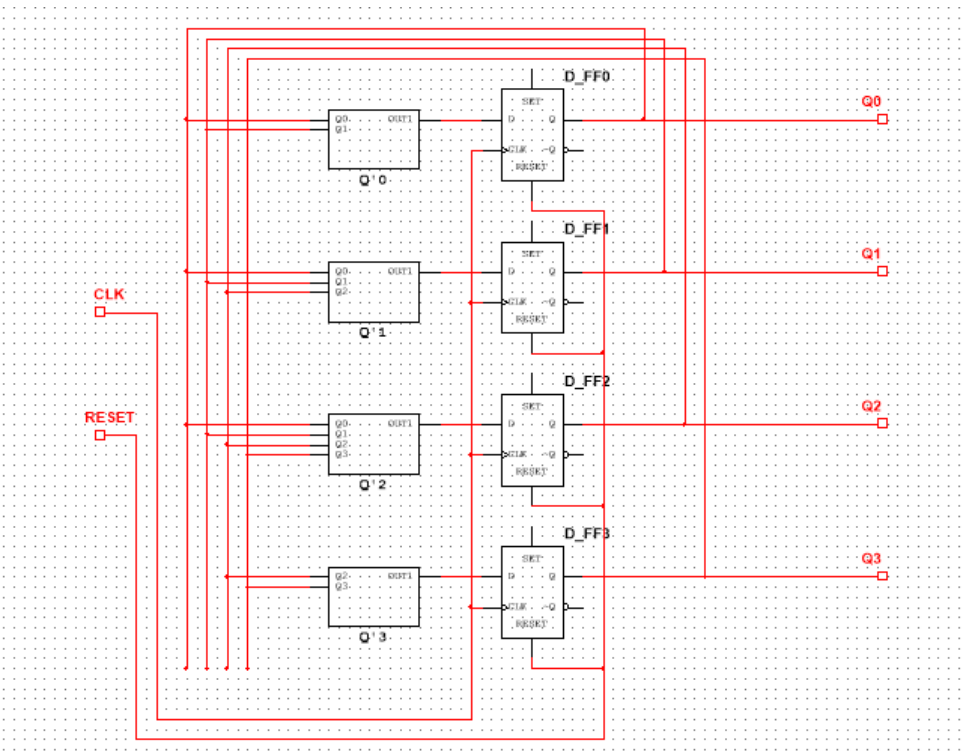


Rysunek 5: Schemat logiczny realizujący funkcję $Q'3$

3 Schemat automatu i wyjaśnienie działania

Przedstawiony układ jest 4-bitowym automatem synchronicznym (Rysunek 6).

- **Rejestry Stanu:** Cztery przerzutniki typu D (D_FF0 do D_FF3) przechowują aktualny 4-bitowy stan automatu (wyjścia Q0 do Q3).
- **Logika Następnego Stanu:** Bloki realizujące funkcje logiczne D_0 do D_3 (przedstawione na Rysunkach 2-5), wyprowadzone w Sekcji 2.2. Na podstawie bieżącego stanu (Q0-Q3) obliczają wartości wejściowe dla przerzutników w następnym cyklu zegarowym.
- **Synchronizacja:** Wspólny sygnał zegarowy CLK synchronizuje zmianę stanu we wszystkich przerzutnikach jednocześnie (na zboczu narastającym lub opadającym, w zależności od typu przerzutnika).
- **Reset:** Sygnał RESET (asynchroniczny) wymusza ustawienie stanu początkowego 0000 we wszystkich przerzutnikach, niezależnie od sygnału zegarowego.
- **Wyjścia:** Wyjścia Q0 do Q3 reprezentują bieżący stan automatu, czyli wartość liczby binarnej z sekwencji Fibonacciego.



Rysunek 6: Schemat automatu sekwencyjnego licznika Fibonacciego

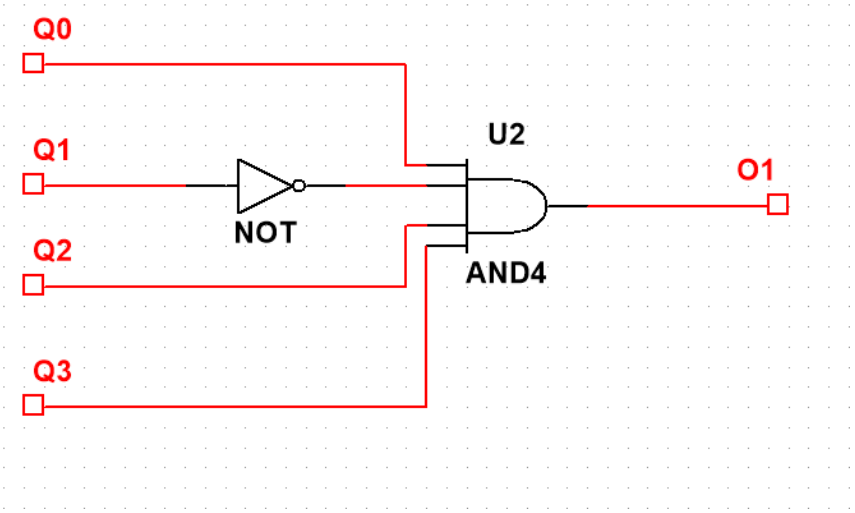
4 Projekt sterownika wyświetlacza 7-segmentowego

4.1 Struktura i zasada działania sterownika

Sterownik wyświetlacza 7-segmentowego został zaprojektowany w celu wizualizacji aktualnego stanu licznika Fibonacciego przy użyciu komponentu Hex Display w środowisku Multisim. Wyświetlacz ten posiada cztery wejścia binarne i są używane dwa takie wyświetlacze.

4.2 Komponent detekcji liczby 13

Centralnym elementem sterownika jest układ "13 Checker", którego zadaniem jest wykrycie, czy aktualna wartość licznika wynosi 13 (binarnie 1101).



Rysunek 7: Schemat układu detekcji liczby 13

Układ posiada cztery wejścia (IO1–IO4) podłączone do wyjść licznika (Q0–Q3) oraz wyjście (OUT) sygnalizujące wykrycie liczby 13. W przypadku pojawienia się na wejściach kombinacji binarnej 1101, układ generuje sygnał aktywny, który przełącza wyświetlacze w tryb specjalny.

4.3 Obsługa wyświetlania liczb

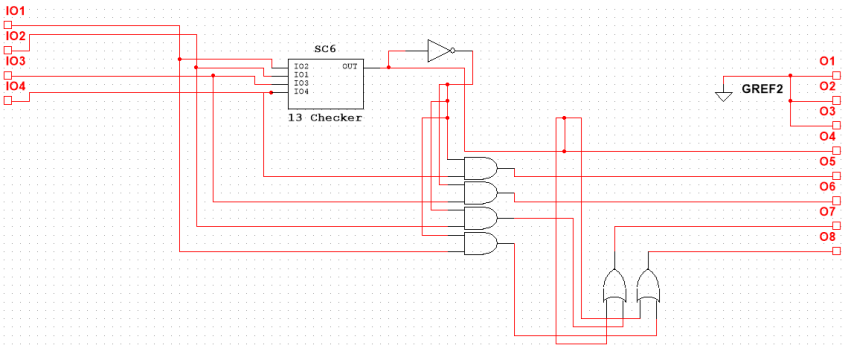
System działa według następującej zasady:

- Dla wartości 0–9: liczby są bezpośrednio podawane na wejścia Hex Display i wyświetlane standardowo (tylko prawy wyświetlacz jest aktywny, lewy pokazuje 0 lub jest wygaszony).
- Dla wartości 13: dwie cyfry, 1 i 3, są wyświetlane osobno na dwóch osobnych Hex Display (lewy wyświetla 1, prawy wyświetla 3).

4.4 Implementacja układu sterującego

Układ sterujący wyświetlaczem został zrealizowany przy użyciu:

- 4 wejść przeznaczonych na bity liczby (z licznika Q0-Q3).
- Układu 13 Checker do detekcji wartości 13.
- Zestawu bramek logicznych AND i OR do multipleksowania sygnałów w zależności od trybu wyświetlania (liczby 0-9 vs liczba 13).
- 8 wyjść do sterowania dwoma wyświetlaczami Hex Display (po 4 bity na wyświetlacz).



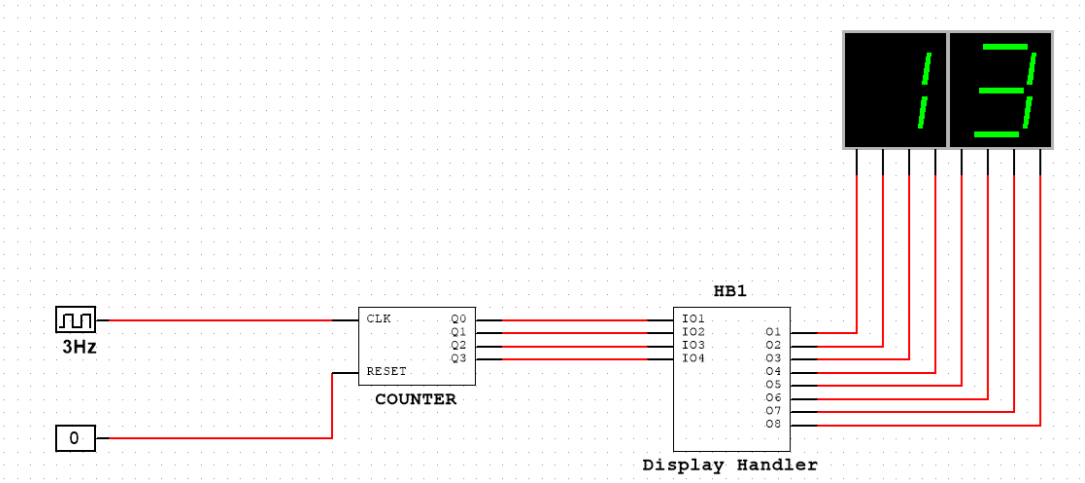
Rysunek 8: Schemat układu sterującego wyświetlaczem

5 Schemat całościowy systemu

Kompletny schemat systemu przedstawiono na Rysunku 9. Integruje on kluczowe bloki funkcjonalne:

- **Generator zegarowy (DIGITAL_CLOCK):** Dostarcza sygnał taktujący (CLK).
- **Licznik Fibonacciego (Counter):** Blok realizujący automat opisany w Sekcji 3. Generuje 4-bitową sekwencję (0, 1, 2, 3, 5, 8, 13).
- **Sterownik Wyświetlacza (Display Handler):** Blok opisany w Sekcji 4. Przetwarza wyjście licznika na sygnały dla wyświetlaczy, obsługując specjalnie liczbę 13.
- **Wyświetlacze 7-segmentowe (Hex Display):** Dwa moduły do wizualizacji aktualnej liczby.
- **Sygnał Reset:** Umożliwia powrót licznika do stanu początkowego (0).

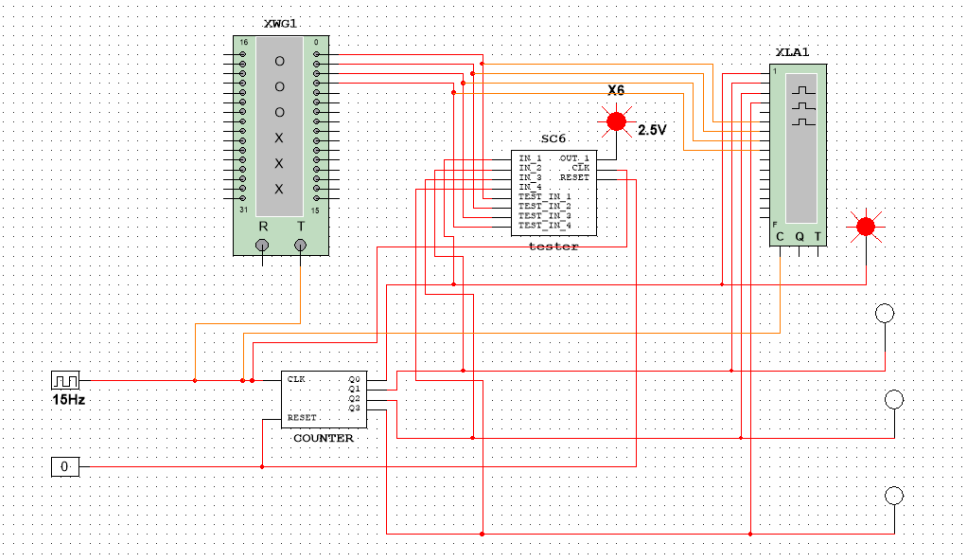
System działa synchronicznie: sygnał zegarowy CLK taktuje blok licznika (Counter). Stan licznika (jego wyjścia Q0-Q3) jest podawany na wejście sterownika wyświetlacza (Display Handler), który generuje odpowiednie kody dla dwóch wyświetlaczy Hex Display. Sygnał Reset inicjalizuje licznik do stanu 0000.



Rysunek 9: Schemat całościowego układu licznika Fibonacciego ze sterownikiem i wyświetlaczami

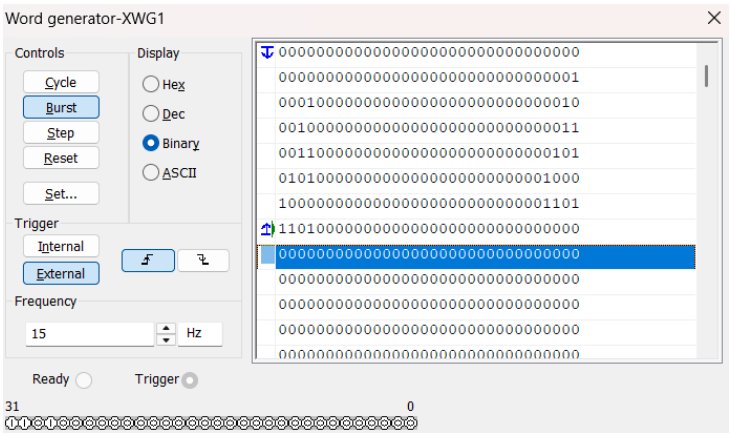
6 Testowanie układu

Testowanie poprawności działania układu odbywa się poprzez zaimplementowane stanowisko testowe. Generator słów przyjmuje external clocka oraz ma na wyjściu oczekiwane sygnały wyjścia, które później porównujemy z faktycznymi wyjściami naszego układu za pomocą testera.



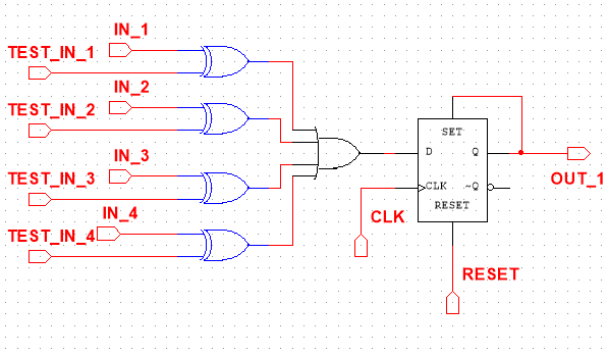
Rysunek 10: Zdjęcie układu testowego.

Sekwencje generatora słów na czterech ostatnich bitach mają ustawione oczekiwane wartości (kolejne wyrazy ciągu Fibonacciego w systemie binarnym).



Rysunek 11: Zdjęcie generatora słów.

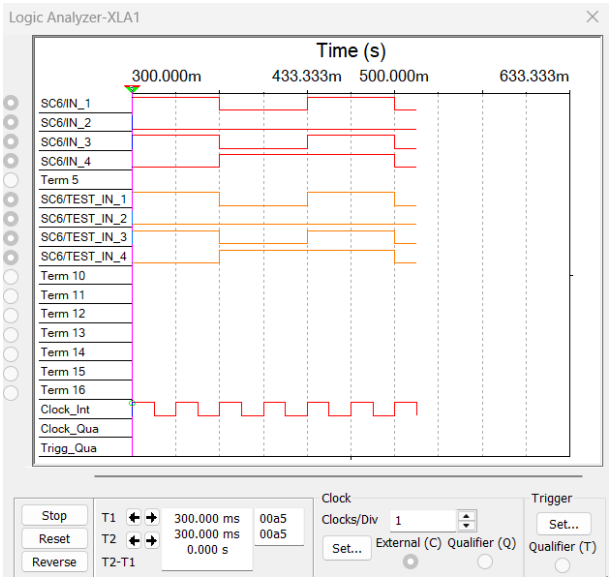
Tester opiera się na czterech bramkach XOR idących do bramki OR z czterema wejściami. Do każdego XOR wchodzi odpowiednio in_i – wejście faktyczne z układu oraz in_test_i – wejście z word generatora. Każda niezgodność zostaje przekazana do przerzutnika D.



Rysunek 12: Zdjęcie testera.

W przerzutniku D, dzięki połączeniu wyjścia Q i wejścia SET, przy wykryciu błędu próbnik zaświeci się na czerwono i nie zgaśnie do momentu ręcznego zresetowania układu.

Wyjścia z układu, jak i wyjścia z word generatora, trafiają do analizatora stanów logicznych (logic analyzera) z internal clockiem ustawionym na dwukrotną częstotliwość naszego oryginalnego zegara, co pozwala na szczegółową analizę przebiegów i skuteczne zweryfikowanie wyników.



Rysunek 13: Zdjęcie analizatora logicznego.

7 Alternatywna Implementacja z Wykorzystaniem Sumatora

Innym sposobem realizacji licznika Fibonacciego jest wykorzystanie jego definicji rekurencyjnej $F_n = F_{n-1} + F_{n-2}$. Schemat takiej implementacji przedstawiono na Rysunku 18.

Układ wykorzystuje dwa 4-bitowe rejestry, oznaczone jako SC3 (`c1`) i SC2 (`c0`), oraz 4-bitowy sumator U1 (74LS83D). Zasada działania jest następująca:

- Rejestr `c1` przechowuje aktualną liczbę Fibonacciego (F_n)
- Rejestr `c0` przechowuje poprzednią liczbę (F_{n-1})

- Sumator oblicza wartość $F_n + F_{n-1}$, która będzie kolejną liczbą Fibonacciego (F_{n+1})

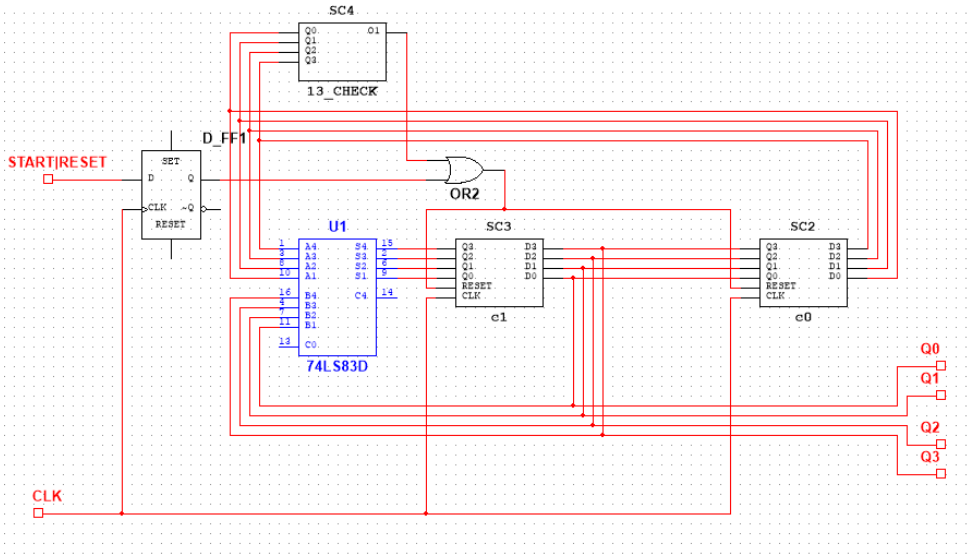
W każdym takcie zegara CLK:

1. Sumator 74LS83D oblicza sumę wyjść rejestrów `c1` i `c0`
2. Wynik sumowania jest przekazywany na wejścia danych (D0-D3) rejestru `c1`
3. Aktualna wartość rejestru `c1` jest przekazywana na wejścia danych rejestru `c0`
4. Rejestry zapamiętują nowe wartości z nadejściem następnego zbocza zegarowego

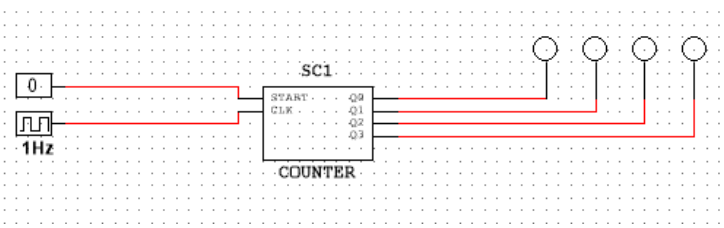
Aktualna wartość ciągu Fibonacciego jest dostępna na wyjściach Q0-Q3 rejestru `c1`. Układ SC4 (`13_CHECK`) monitoruje wyjście rejestru `c1` i wykrywa wartość 13 (1101). Po wykryciu tej wartości generuje sygnał resetujący, który poprzez bramkę OR2 może resetować licznik.

Zresetowanie układu (poprzez sygnał START|RESET) inicjalizuje rejestr `c1` do wartości 0001 ($F_1 = 1$), a rejestr `c0` do wartości 0000 ($F_0 = 0$). To ustawienie zapewnia poprawne rozpoczęcie sekwencji Fibonacciego: 1, 1, 2, 3, 5, 8, 13, a następnie powrót do początku.

Do uruchomienia układu należy nacisnąć i zwolnić przycisk START|RESET, co ustawia początkowe wartości w rejestrach i rozpoczyna generowanie sekwencji Fibonacciego.



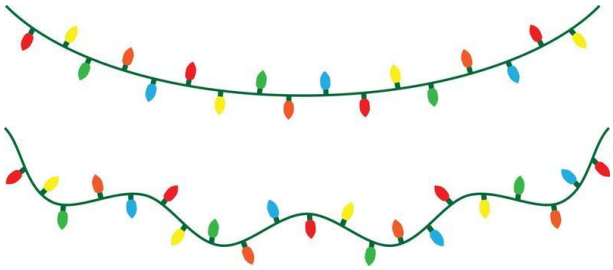
Rysunek 14: Schemat blokowy alternatywnej implementacji licznika Fibonacciego z użyciem sumatora



Rysunek 15: Schemat blokowy całości rozwiązania (komponent COUNTER wykonuje całą logikę zadania)

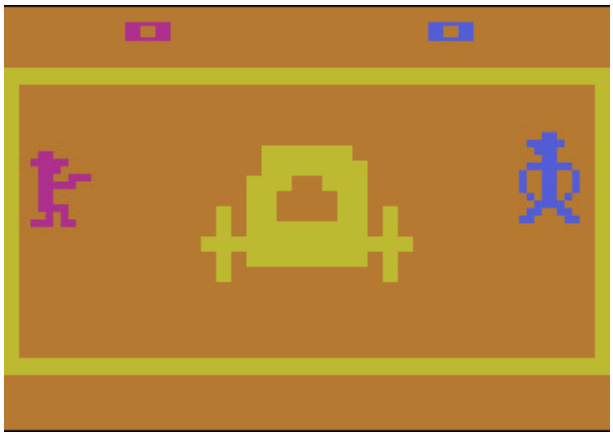
8 Zastosowania

1. **Generator prostych efektów wizualnych:** Nieliniowa sekwencja zliczania (0, 1, 2, 3, 5, 8, 13) może być wykorzystana do sterowania prostymi efektami w celach artystycznych lub rozrywkowych. Wyjścia licznika (Q0-Q3) mogą sterować jasnością lub kolorem diod LED. Zmiany następujące zgodnie z ciągiem Fibonacciego mogą tworzyć bardziej organiczne lub mniej przewidywalne wzory świetlne niż liniowe narastanie/opadanie.

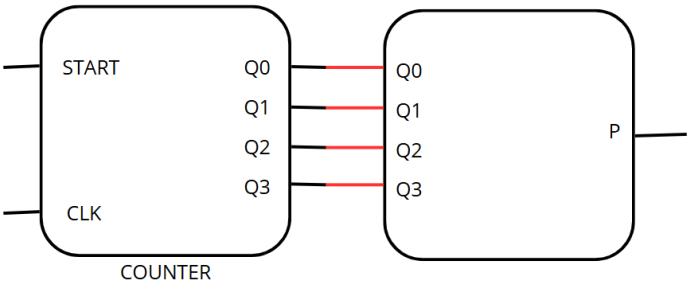


Rysunek 16: Bardzo ogólne przedstawienie układu

2. **Element gier lub zabawek interaktywnych:** Nietypowa sekwencja może wprowadzić element zaskoczenia lub stanowić podstawę mechaniki w prostych grach stworzonych na układzie FPGA lub mikrokontrolerze. W grze zręcznościowej licznik mógłby odmierzać czas do pojawienia się przeszkody lub pocisku z broni przeciwnika. Odstępy czasowe byłyby proporcjonalne do kolejnym liczb Fibonacciego (np. 1, 2, 3, 5... jednostek czasu), co czyniłoby rozgrywkę mniej przewidywalną. Może też służyć jako generator pseudolosowych zdarzeń opartej na wartości licznika.



Rysunek 17: Bardzo ogólne przedstawienie układu



Rysunek 18: Bardzo uproszczony schemat w jaki sposób mógł być podłączony nasz układ aby umożliwić powyższe zastosowanie, w komponencie drugim sygnał P byłby 1 gdyby jakaś przeszkoda/bonus miałby się pojawić w grze

9 Wnioski

Zaprojektowano i opisano czterobitowy licznik Fibonacciego, generujący wymaganą sekwencję stanów (0, 1, 2, 3, 5, 8, 13), powracający do stanu 0.
Podstawą projektu była tabela przejść stanów (Tabela 1), na podstawie której wyznaczono zminimalizowane funkcje logiczne wejść przerzutników D (D_0, D_1, D_2, D_3) przy użyciu map Karnaugha (Sekcja 2.2).

Zastosowane metody projektowania automatów sekwencyjnych okazały się skuteczne.
Poprawność minimalizacji funkcji oraz ich implementacje za pomocą bramek logicznych przedstawiono bezpośrednio pod mapami Karnaugha.

Zaprojektowano również dedykowany sterownik wyświetlacza 7-segmentowego (Sekcja 4), umożliwiający poprawną wizualizację wszystkich stanów licznika, w tym dwucyfrowej liczby 13.
Kompletny system zintegrowano i przedstawiono na schemacie całościowym (Rysunek 9).

Omówiono również alternatywną, bardziej bezpośrednią metodę realizacji licznika z wykorzystaniem sumatora i rejestrów (Sekcja 7).
Układ stanowi praktyczną ilustrację syntezy niestandardowych liczników synchronicznych oraz projektowania interfejsów do wyświetlaczy.