Technika Cyfrowa
Ćwiczenie 2

Kacper Bieniasz
Aleksander Buszek
Maciej Nowakowski
Albert Pęciak

30.04.2024

1 Temat ćwiczenia

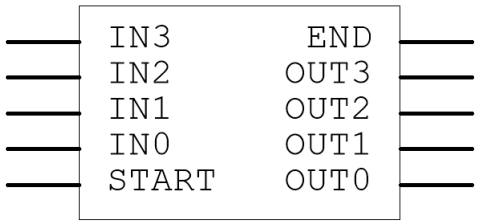
Korzystając wyłącznie z wybranych przerzutników oraz dowolnych bramek logicznych, należało zaprojektować czterobitowy układ TIMER, odmierzający ustawiany za pomocą przełączników czas (od 0 do 15).

Po wciśnięciu przycisku START, układ rozpoczyna odmierzanie czasu do tyłu. Po wyzerowaniu się licznika czasu, układ powinien się zatrzymać i włączyć alarm świetlny wykorzystujący diodę LED. Po ponownym wciśnięciu przycisku START, układ powinien wyłączyć alarm i ponownie rozpocząć odmierzanie ustawionego na przełącznikach czasu.

Aktualny wskazywany przez układ czas należało pokazywać na wyświetlaczach siedmiosegmentowych.

2 Implementacja układu

Układ timer przyjmuje na wejściu pięć bitów. Czterobitową liczbę IN oraz sygnał START rozpoczynający pracę układu. Bity wyjściowych wysyła aktualny stan timera (OUT3 - OUT0) oraz informację o zakończeniu działania układu(END).

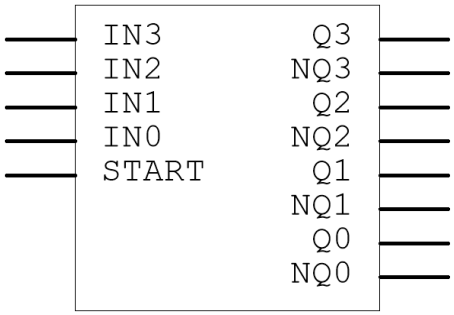


Rysunek 1: Bity IN3 - IN0 - cztero bitowa wartość początkowa odliczania; START - sygnał rozpoczęcia pracy timera; Bity OUT3 - OUT0 - aktualna wartość timera ; END - sygnał zakończenia pracy timera

Bity wejściowe są przekazywane do podukładu SETTER, który ustawia i uruchamia główną część obwodu.

2.1 Setter układu TIMER

Setter układu służy do rozpoczynania timera. Każdy z czterech podukładów po otrzymaniu sygnału z bitu START wysyła sygnał Q_n zgodny z wartością IN_n oraz sygnał o przeciwnej wartości NQ_n.



Rysunek 2: Bity wejściowe to bity przekazywane do układu TIMER; Qn oraz NQn dla n∈<0,3> sygnały nadające ustawienia początkowe przerzutnikom.

IN_n	START	Q	NQ
1	0	0	0
1	1	1	0
0	0	0	0
0	1	0	1

Tabela 1: Tabela prawdy dla każdego z czterech podukładów układu SETTER

IN_n \ $START$	0	1
0	0	0
1	0	1

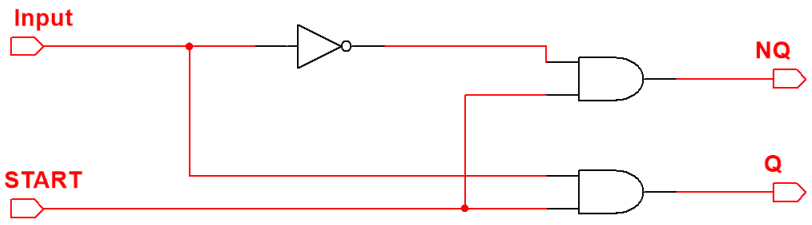
Wynik: $\boldsymbol{START + IN_n}$

Rysunek 3: Tablica Karnough dla wyjścia Q

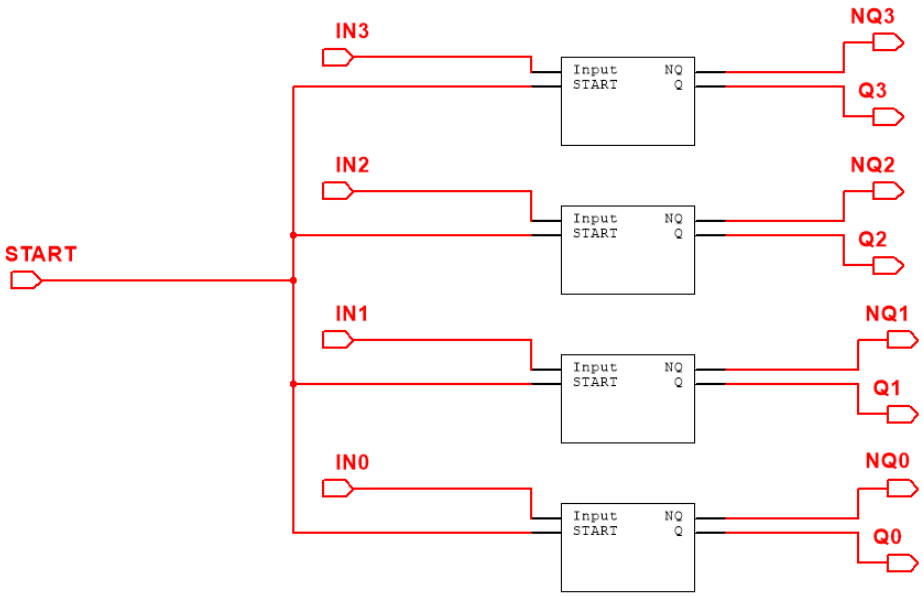
IN_n \ $START$	0	1
0	0	1
1	0	0

Wynik: $\boldsymbol{START + \overline{IN_n}}$

Rysunek 4: Tablica Karnough dla wyjścia NQ



Rysunek 5: Podukład nastawiający jeden z czterach bitów timera



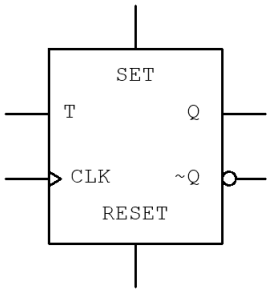
Rysunek 6: Cały układ SETTER

Bity wyjściowe przesyłane są do przerzutników, na których opiera się główny mechanizm układu TIMER. Wszystkie cztery podukłady nastawiające dany bit wyglądają identycznie.

2.2 Przechowywanie i aktualizowanie stanu timera

Przerzutniki

By generować kolejne malejące liczby potrzebujemy mechanizmu zapamiętującego wartość timera, który można łatwo zaktualizować o kolejną wartość. Z pomocą przychodzą nam przerzutniki typu T, które po otrzymaniu informacji o rozpoczęciu odczytywania (wejście CLK) zmieni swój stan jeśli na T wysłano wartość logiczną **1**. Podanie **0** na wejście T powoduje zachowanie bieżącego stanu przerzutnika. Wejścia SET i RESET służą do nadania stanu początkowego.



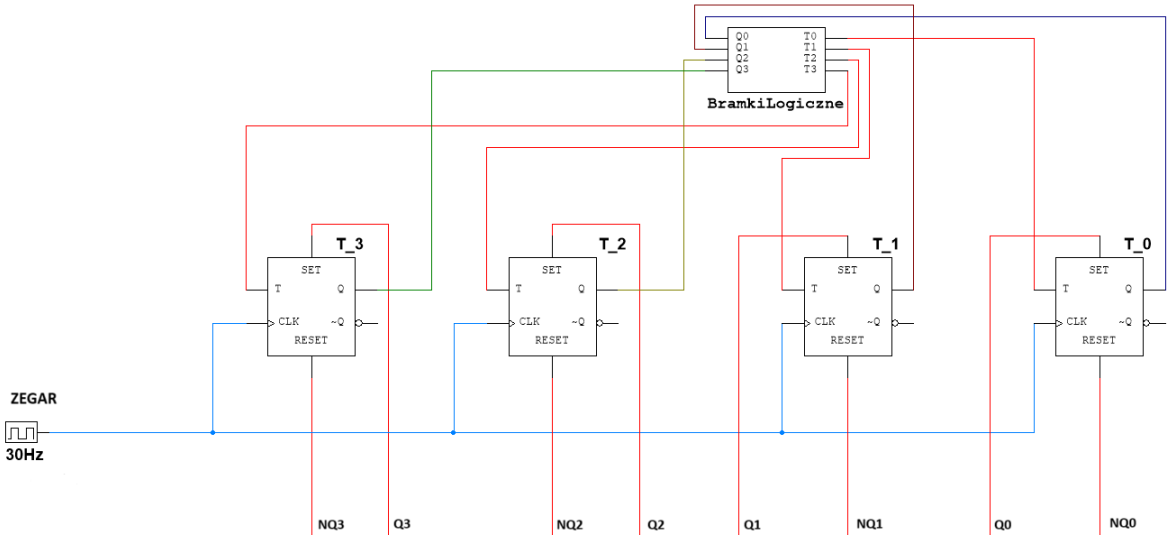
Rysunek 7: Przerzutnik typu T

T	Q_n	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Tabela 2: Tabela prawdy dla przerzutnika typu T

Przechowywanie stanu timera

Każdy z czterech bitów tworzących aktualny stan timera (liczbę czterobitową) jest określany przez przerzutniki T_3 , T_2 , T_1 i T_0, który jest zwracany na wyjściu Q by aktualizować stan przerzutnika.



Rysunek 8: Podłączenie przerzutników. Sygnały NQ3, Q3, NQ2... programujące przerzutniki przekazuje SETTER. Zegar wysyła z określoną częstotliwością sygnał o konieczności zmiany statusu.

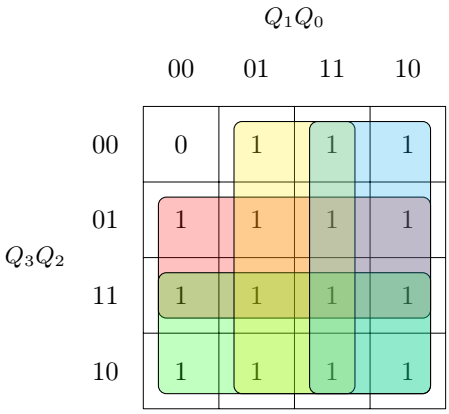
Na początku stworzyliśmy tabele dla wejść (A3, A2, A1, A0) oraz wyjść (B3, B2, B1, B0), a także bitów przełączania przerzutników (T3, T2, T1, T0). Mówi ona jakie wartości muszą być przesłane by liczbę n przerzutniki zmieniły w liczbę $n-1$. Liczbę zero zamieniamy na liczbę zero (ostatni wiersz tabeli (3)), by na wyjście układu TIMER przesyłane były wartości, które przy pomocy bramki NOR można było zamienić na sygnał zakończenia pracy timera (patrz rysunek (13)).

Decimal	A_3	A_2	A_1	A_0	Decimal	B_3	B_2	B_1	B_0	T_3	T_2	T_1	T_0
	input					expected output				flip flop reset			
15	1	1	1	1	14	1	1	1	0	0	0	0	**1**
14	1	1	1	0	13	1	1	0	1	0	0	**1**	**1**
13	1	1	0	1	12	1	1	0	0	0	0	0	**1**
12	1	1	0	0	11	1	0	1	1	0	**1**	**1**	**1**
11	1	0	1	1	10	1	0	1	0	0	0	0	**1**
10	1	0	1	0	9	1	0	0	1	0	0	**1**	**1**
9	1	0	0	1	8	1	0	0	0	0	0	0	**1**
8	1	0	0	0	7	0	1	1	1	**1**	**1**	**1**	**1**
7	0	1	1	1	6	0	1	1	0	0	0	0	**1**
6	0	1	1	0	5	0	1	0	1	0	0	**1**	**1**
5	0	1	0	1	4	0	1	0	0	0	0	0	**1**
4	0	1	0	0	3	0	0	1	1	0	**1**	**1**	**1**
3	0	0	1	1	2	0	0	1	0	0	0	0	**1**
2	0	0	1	0	1	0	0	0	1	0	0	**1**	**1**
1	0	0	0	1	0	0	0	0	0	0	0	0	**1**
0	0	0	0	0	0	0	0	0	0	0	0	0	0

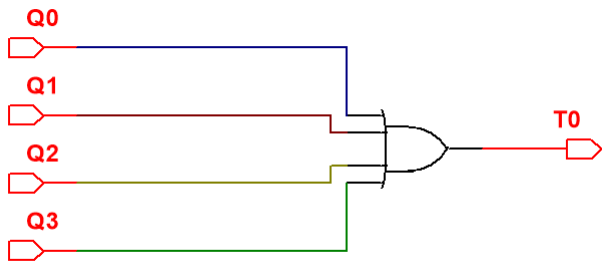
Tabela 3: Tabela prawdy

Aby zminimalizować liczbę potrzebnych funkcji skorzystaliśmy z tablic Karnough. Pozwalają one w stosunkowo łatwy sposób uprościć formę funkcji logicznych zapewniając poprawność. Dla każdego bitu wyjścia tworzymy jedną tablice. W tworzeniu siatek pamiętamy o kodzie Gray'a, aby sąsiednie bity różniły się między sobą o jeden bit.

Bit T0



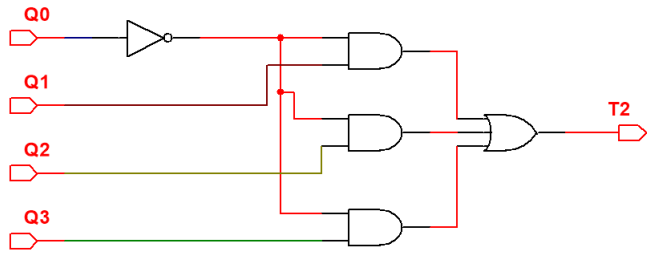
Z czerwonego otrzymujemy: Q_2
Z zielonego otrzymujemy: Q_3
Z żółtego otrzymujemy: Q_0
Z niebieskiego otrzymujemy: Q_1
Ostateczny wynik: $\boldsymbol{Q_3 + Q_2 + Q_1 + Q_0}$



Rysunek 9: Tworzenie sygnału dla przerzutnika T0

Bit T1

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

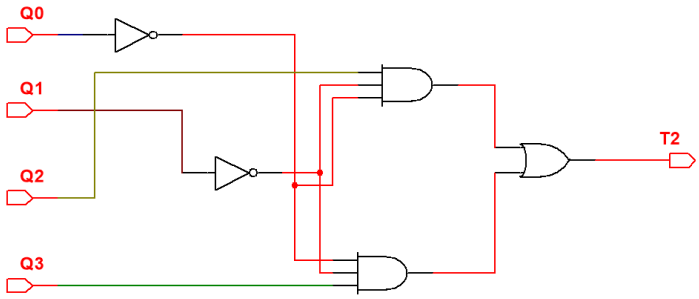


Rysunek 10: Tworzenie sygnału dla przerzutnika T1

Z czerwonego otrzymujemy: $Q_1\ \overline{Q_0}$
Z zielonego otrzymujemy: $Q_2\ \overline{Q_0}$
Z żółtego otrzymujemy: $Q_3\ \overline{Q_0}$
Ostateczny wynik: $\boldsymbol{Q_1\ \overline{Q_0} + Q_3\ \overline{Q_0} + Q_2\ \overline{Q_0}}$

Bit T2

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

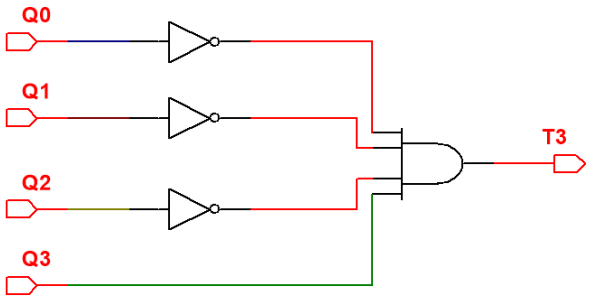


Rysunek 11: Tworzenie sygnału dla przerzutnika T2

Z czerwonego otrzymujemy: $Q_2\ \overline{Q_0}\ \overline{Q_1}$
Z zielonego otrzymujemy: $Q_3\ \overline{Q_0}\ \overline{Q_1}$
Ostateczny wynik: $\boldsymbol{Q_3\ \overline{Q_0}\ \overline{Q_1} + Q_2\ \overline{Q_0}\ \overline{Q_1}}$

Bit T3

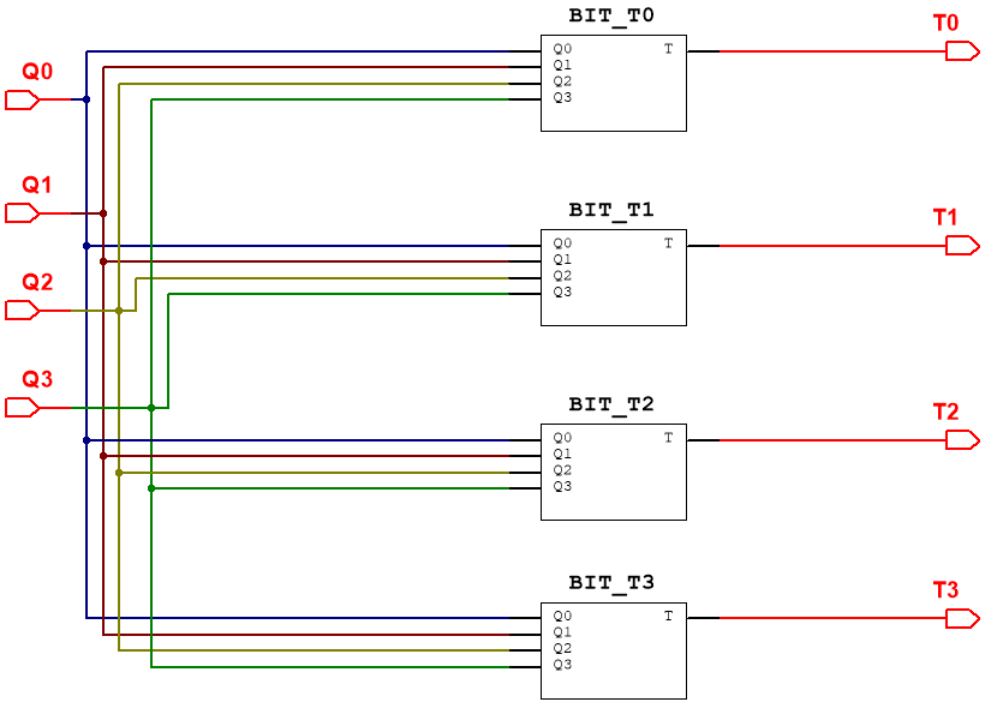
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	1	0	0	0



Rysunek 12: Tworzenie sygnału dla przerzutnika T3

Z czerwonego otrzymujemy: $Q_3\ \overline{Q_0}\ \overline{Q_1}\ \overline{Q_2}$
Ostateczny wynik: $\boldsymbol{Q_3\ \overline{Q_0}\ \overline{Q_1}\ \overline{Q_2}}$

Pełny układ bramek logicznych:



Rysunek 13: Schemat układu bramek logicznych

Sygnały z tej części układu, są przesyłane dalej do bramki, która sprawdza stan logiczny czterech bitów wyjściowych i ocenia czy TIMER zakończył działanie.

2.3 Zwracanie wyniku

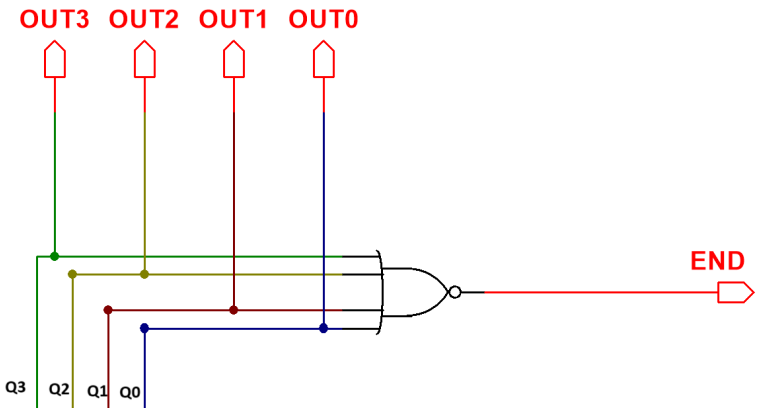
Bramka logiczna NOR zwróci wartość logiczną **1** gdy wszystkie cztery bity będą równe 0, czyli gdy timer będzie wskazywał liczbę 0. Gdy przerzutniki "dotrą"do wartości końcowej i podadzą informację dalej do układu bramek logicznych (13) otrzymają informacje by nie zmieniać sygnału (patrz ostatni wiersz tabeli (3)) i tak w koło, więc na wyjście będzie wysyłana informacja o zakończeniu testu do ponownego naciśnięcia przycisku start i zresetowaniu przerzutników.

Q3	Q2	Q1	Q0	Alarm
1	1	1	1	0
1	1	1	0	0
1	1	0	1	0
1	1	0	0	0
1	0	1	1	0
1	0	1	0	0
1	0	0	1	0
1	0	0	0	0
0	1	1	1	0
0	1	1	0	0
0	1	0	1	0
0	1	0	0	0
0	0	1	1	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Rysunek 14: Tabela prawdy układu wyjściwego

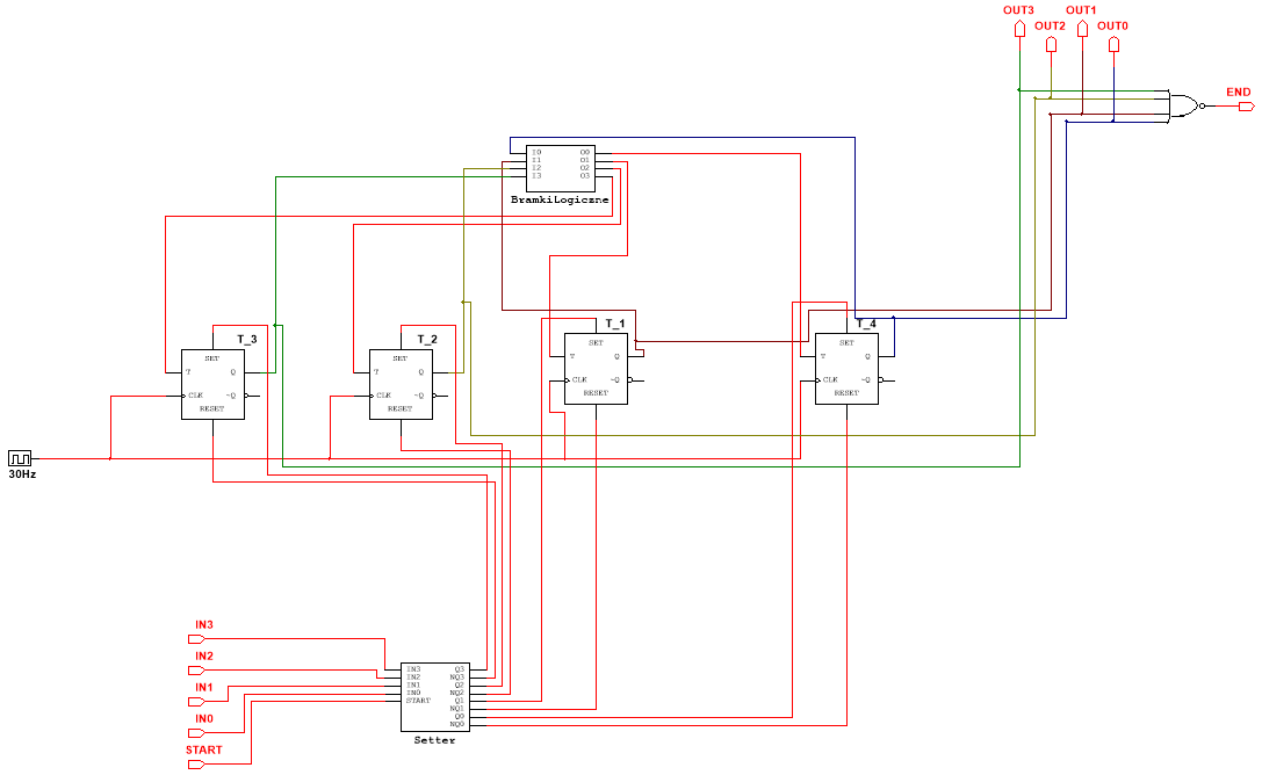
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	1	0	0	0
01	0	0	0	0
11	0	0	0	0
10	0	0	0	0

Z czerwonego otrzymujemy: $\overline{Q_3}\ \overline{Q_0}\ \overline{Q_1}\ \overline{Q_2}$
Ostateczny wynik: $\boldsymbol{\overline{Q_3}\ \overline{Q_0}\ \overline{Q_1}\ \overline{Q_2}}$

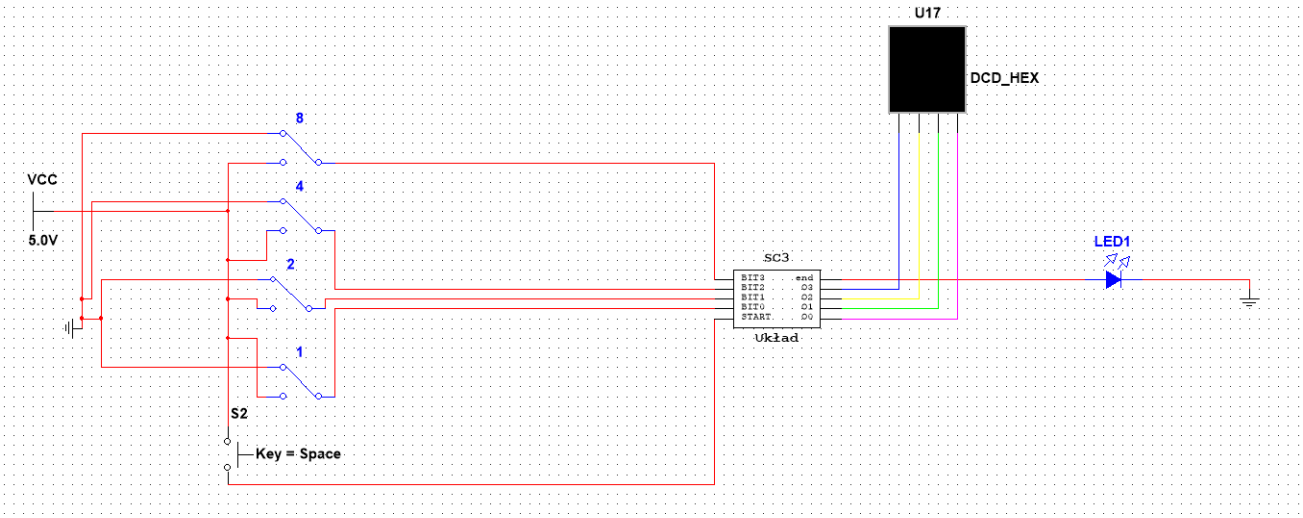


Rysunek 15: Wyjścia układu TIMER. Gdy wszystkie bity liczby wyjściowej są równe zero wysyłany jest sygnał końcowy.

Układ TIMER w całości:



Rysunek 16: Bity wejściowe przekazywane są do układu Setter, który rozpoczyna pracę przerzutników, a z przerzutników sygnały trafiają do bramki NOR i na wyjście.

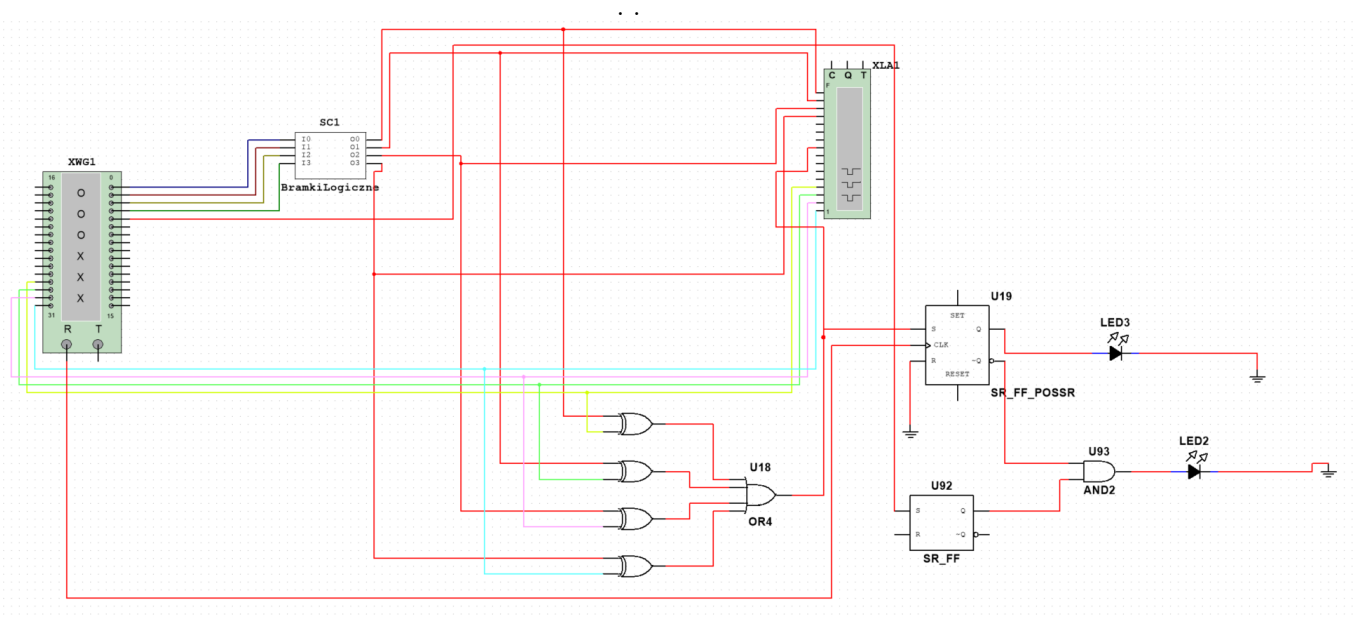


Rysunek 17: Skrzynka z układałem TIMER podłączona do zewnętrznych urządzeń.

Do układu TIMER podłączone są cztery przełączniki którymi wskazujemy startową wartość odliczania. Przycisk BUTTON startuje timer. Wyświetlacz siedmiosegmentowy wyświetla aktualny stan timera w systemie szesnastkowym, a dioda LED zapala się gdy timer zakończył swoje działanie.

3 Testowanie układu

Testować będziemy tylko jeden podukład całego układu. Testowanie układu polega na sprawdzeniu, czy bity wejściowe poprawnie zostały zamienione na bity, które otrzymuje układ przerzutników odmierzający czas. Testowany układ zamienia bity wejściowe na bity, które trafiają do układu przerzutników. Dokładny opis działania testowanego mechanizmu jest przedstawiony w rozdziale 2.2.

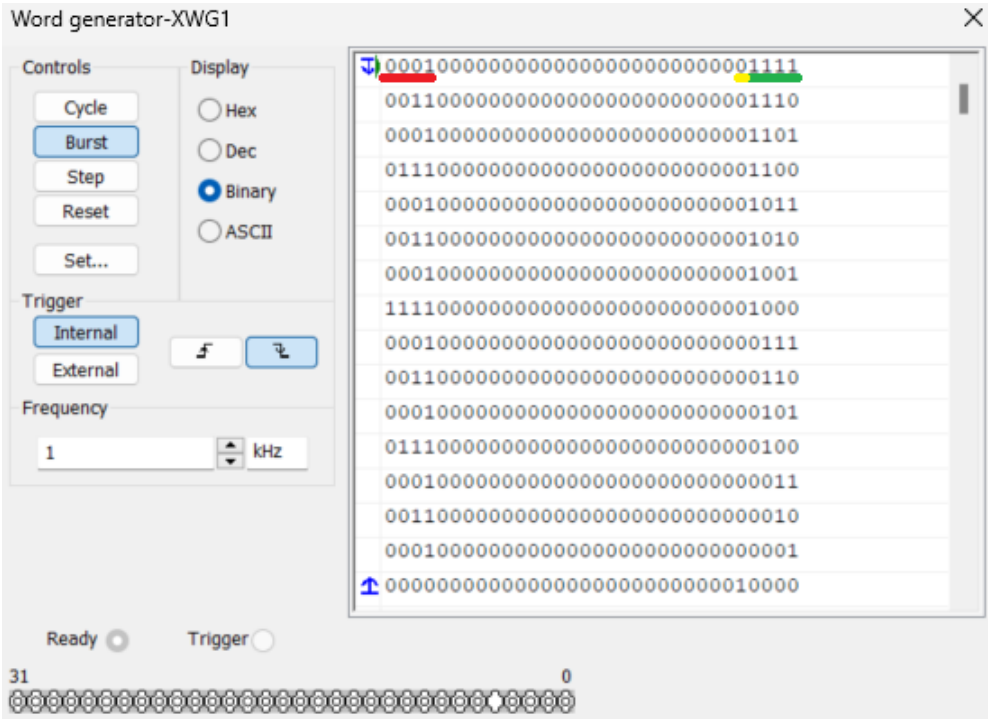


Rysunek 18: Układ testujący w całości

Opis poszczególnych komponentów znajduje się poniżej.

3.1 Generator słów

Generator słów wysyła 16 sygnałów, które zawierają bity wejściowe, wyjściowe oraz sygnał zakończenia odliczania.

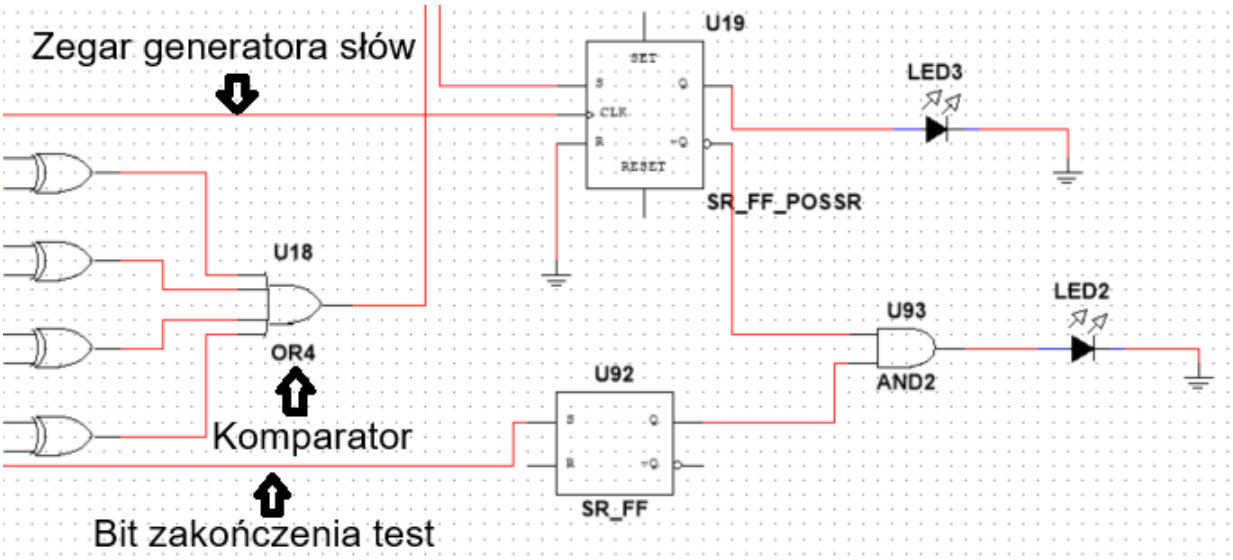


Rysunek 19: Generator bitów używanych do testowania układu

- Bity wejściowe - cztery pierwsze bity oznaczające liczby do 15 do 0 malejąco.
- Oczekiwane bity wyjściowe - cztery ostatnie bity, które są dostarczane do układu przerzutników.
- Bit końcowy - piąty bit od prawej oznaczający koniec odliczania.

3.2 Układ wyświetlający wyniki testu

Komparator porównuje wartości oczekiwane z rzeczywistymi wyjściowych bitów i wysyła sygnał do górnego przerzutnika. Przed otrzymaniem sygnału od komparatora generator słów wysyła sygnał o kolejnej liczbie. Jeśli przerzutnik otrzyma informację o błędnym wyniku testu aktualizuje sygnał wysyłany z Q (i $\overline{Q}$) co zapala górną diodę. W przeciwnym wypadku, po poprawnym zmienieniu liczb dolny przeprzutnik otrzymuje informację o zakończonym tescie i zapala diodę dolną.



Rysunek 20: Układ odpowiedzialny za porównywanie wyników

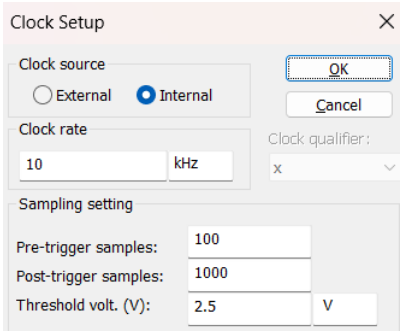
- **górny przerzutnik** w momencie otrzymania przeczącego sygnału od komparatora zapala LED3; jeśli nie ma błędu przesyła sygnał do bramki AND2
- **dolny przerzutnik** w momencie zakończenia testu wysyła do AND2 odpowiedni sygnał

Gdy układ nie wykryje błedu oraz popłynie sygnał z generatora słów, LED2 zapali sie na niebiesko sygnalizując pozytywne zakończenie testowania układu.

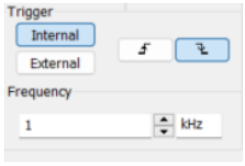
3.3 Analizator stanu logicznego

Do analizatora stanów logicznych podpięte jest 9 bitów. Górne 4 to bity wyjściowe z układu, środkowy to bit komparatora, gdy wykryje błąd to zauważymy wzorst bądź spadek na wartości tego bitu, natomiast 4 dolne bity to oczekiwane bity wyjściowe wysyłane przez generator słów.

Rysunek 21: Wskazania analizatora stanów logicznych po pozytywnym teście



Rysunek 22: Ustawienia analizatora



Rysunek 23: Ustawienia generatora słów

Przy testowaniu układu pamiętamy o twierdzeniu Nyquista. Częstotliowść generatora słów to 1kHz, częstotliwość próbkowania analizatora wynosi 10kHz, co jest zgodne z owym twierdzeniem.

3.4 Testowanie pozostałych komponentów układu odliczającego

Pozostałe układy, które należy przetestować w układzie odliczającym to jest: setter oraz komparator testujemy na tej samej zasadzie co powyżej.

4 Wnioski i zastosowanie

- Rozbicie całego układu na podukłady odpowiedzialne za konkretne czynności pozwoliło lepiej zorganizować pracę, a także ułatwiło budowę i testowanie układu.
- Przerzutniki asynchroniczne w układzie TIMER nie sprawdziły by się tak dobrze jak synchroniczne. W sytuacji gdy rejestrujemy zmianę co stałą jednostkę czasu wykorzystanie przerzutników asynchronicznych utrudniło by znacząco implementację.

Zastosowanie
Zaprojektowany przez naszą grupę licznik można wykorzystać do jeszcze większego zabezpieczenia zaproponowanego przez nas systemu szyfrowania w sprawozdaniu pierwszym. W poniższym zastosowaniu opiszemy jak wykorzystać skonstruowany licznik do zmiany klucza odszyfrowującego. Pod spodem znajdują się zmodyfikowane tabele służące do tłumaczenia wiadomości. Będą one potrzebne do wytłumaczenia wykorzystania licznika.

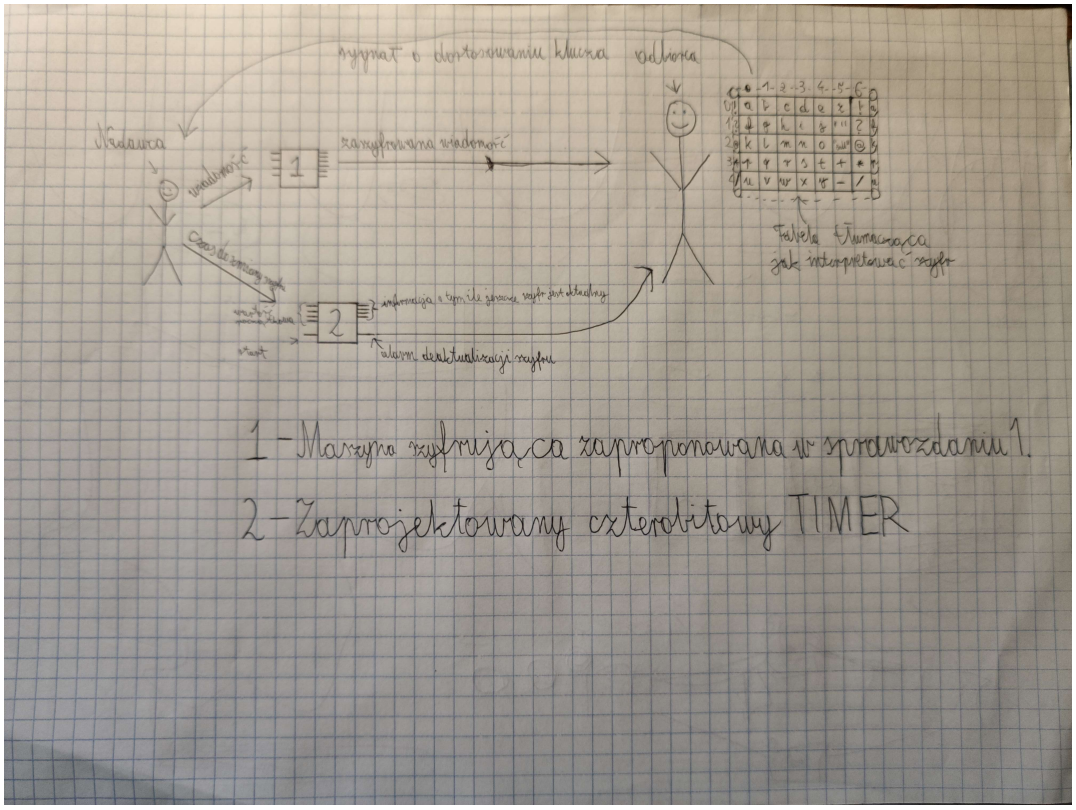
Żółta oś	**Bit B_5**	**Bit B_4**	**Bit B_3**	**Bit B_2**	**Bit B_1**	**Bit B_0**	**Czerwona oś**
i mod 7	**0**	**0**	**0**	**0**	**1**	**0**	**0**
(i+1) mod 7	**0**	**0**	**1**	**0**	**1**	**1**	**1**
(i+2) mod 7	**0**	**1**	**0**	**1**	**0**	**1**	**2**
(i+3) mod 7	**0**	**1**	**1**	**1**	**1**	**1**	**3**
(i+4) mod 7	**1**	**0**	**0**	**0**	**0**	**1**	**4**
(i+5) mod 7	**1**	**0**	**1**	**Nie dotyczy**	**Nie dotyczy**	**Nie dotyczy**	**Nie dotyczy**
(i+6) mod 7	**1**	**1**	**0**	**Nie dotyczy**	**Nie dotyczy**	**Nie dotyczy**	**Nie dotyczy**

Tabela 4: Tabela z bitami, które generują cały język.

bity czerwone / bity żółte,	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	**a**	**b**	**c**	**d**	**e**	**z**	**!**
1	**f**	**g**	**h**	**i**	**j**	**""**	**?**
2	**k**	**l**	**m**	**n**	**o**	**"endl"**	**@**
3	**p**	**q**	**r**	**s**	**t**	+	*****
4	**u**	**v**	**w**	**x**	**y**	**-**	/

Tabela 5: Rozszyfrowanie co oznaczają poszczególne konfiguracje bitów, z tabeli powyżej.

W celu zwiększenia bezpieczeństwa szyfrowania należałoby często zmieniać to jak nasze zaszyfrowane bity są interpretowane. W tym celu możnaby wykorzystać licznik, który po skończeniu odliczania wysłałby sygnał do zmiany klucza. Najłatwiejszą w implementacji zmianą klucza mogłoby być zliczanie ilości sygnałów zmiany klucza w zmiennej i pokazanej w tablicy pierwszej. Otrzymane w kolumnie Żółta oś liczby mówiłyby, do której kolumny w tabeli drugiej należałoby dospasować otrzymany sygnał. W taki sposób otrzymamy siedem różnych sposobów szyfrowania wiadomości przy bez jakiejkolwiek zmiany sposobu przesyłania wiadomości. Dodatkowo dzięki temu, że nasz licznik ma opcję ręcznego wprowadzania wartości początkowych, to możemy zapewnić nieregularność czasów zmiany szyfru. Jeśli chcielibyśmy zwiększyć bezpieczeństwo szyfrowania moglibyśmy podpiąć bity czerwone do oddzielnego licznika o innym taktowaniu zegara, co spowodwało by możliwość otrzymania aż 35 możliwych kodów i w dodatku jeszcze bardziej skomplikowałoby próbę odkrycia wzorca zmiany kodu, ponieważ opierałoby się to o dwa różne zegary. Poniżej znajduje się rozrysowana prezentacja podpięcia licznika do całego układu szyfrującego.



Rysunek 24: Szkic koncepcyjny zastosowania układu

W powyższym szkicu wykorzystania wykorzystana została metoda z tylko jednym licznikim. Wybrane to zostało ze względu na łatwiejszą prezentację sposobu podbięcia naszego podzespołu do całego układu szyfrującego. Na szkicu zastosowania tabela z kodem została przedstawiona w postaci karty z kodem owiniętej na dwóch walcach. Odbiorca wiadomości w tej sytuacji po otrzymaniu sygnału końca odliczania powinien po prostu przesunąć kartki na walcu o jedno pole w prawo oraz wysłać sygnał nadawcy informujący o dostosowaniu klucza do nowego kodu szyfrującego.