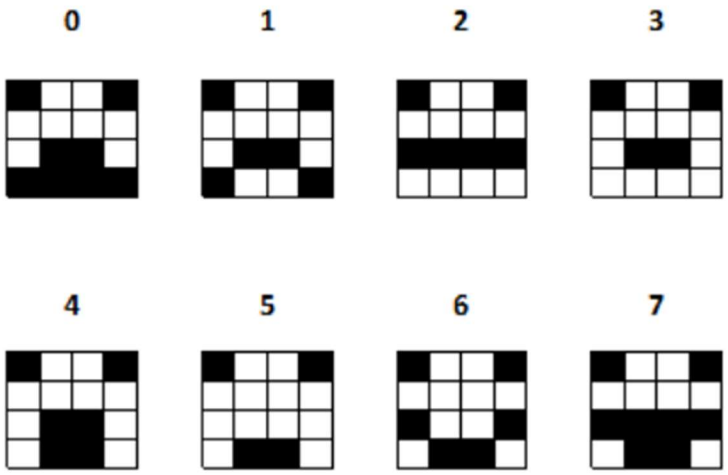
Technika cyfrowa
Sprawozdanie z ćwiczenia nr 1

Rozmus Dariusz, Fabia Jakub,
Smołka Tomasz, Czyż Bartosz, Zając Kacper

1. Treść zadania

Celem ćwiczenia było zaprojektowanie układu cyfrowego pełniącego funkcję dekodera trzybitowej liczby binarnej na układ graficzny przedstawiający emotikony, zgodnie z przedstawionym poniżej rysunkiem.



Rysunek 1: Możliwe do uzyskania warianty emotikon

Zgodnie z wymaganiami, do realizacji układu należało wykorzystać wyłącznie bramki logiczne NAND.

2. Idea rozwiązania

W naszym projekcie wykorzystaliśmy transkoder, który posiada 3 wejścia i 16 wyjść. Wejścia odpowiadają cyfrom binarnej liczby trzybitowej, którą konwertuje na daną emotikonę. Wyjścia są przypisane do konkretnych kratek wyświetlacza dla emotikon.

Wyświetlacz został zrealizowany przy pomocy lampek elektrycznych, gdzie każda kratka odpowiada osobnej lampce. Zapalona lampka reprezentuje kolor czarny, natomiast zgaszona – kolor biały, zgodnie z emotikonami przedstawionymi na Rysunku 1.

Zgodnie z treścią polecenia, do realizacji układu wykorzystaliśmy wyłącznie bramki logiczne *NAND*.

W celu przetestowania działania układu zbudowaliśmy również dodatkowy układ testowy. Wykorzystuje on generator słów oraz wyświetlacz emotikon, działający na tej samej zasadzie co dla układu głównego (dekodera).

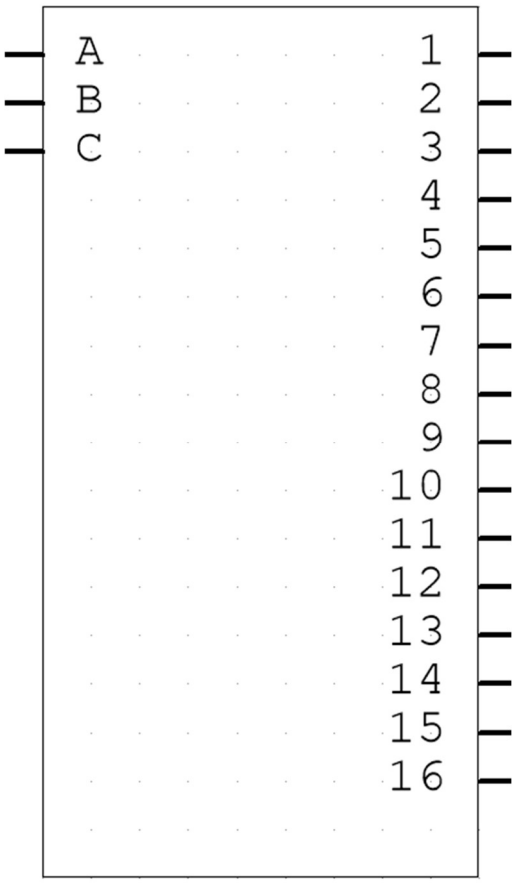
3. Schemat ogólny układu głównego

Oznaczmy odpowiednie kratki wyświetlacza emotikon następującymi symbolami:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Rysunek 2: Oznaczenie poszczególnych kratek wyświetlacza dla emotikon

Wówczas schemat ogólny układu głównego prezentuje się następująco:



Rysunek 3: Schemat wejść i wyjść dla układu głównego/dekodera

Gdzie:

- *A*, *B*, *C* – wejścia odpowiadające cyfrom trzybitowej liczby binarnej, która ma zostać zdekodowana na emotikonę,
- *1*, *2*, *3*, …, *15*, *16* – wejścia odpowiadające konkretnym kratkom wyświetlacza.

4. Tablica wejść i wyjść dla układu głównego

dec.	C	B	A	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	0	1	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1
1	0	0	1	1	0	0	1	0	0	0	0	0	1	1	0	1	0	0	1
2	0	1	0	1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0
3	0	1	1	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0
4	1	0	0	1	0	0	1	0	0	0	0	0	1	1	0	0	1	1	0
5	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0
6	1	1	0	1	0	0	1	0	0	0	0	1	0	0	1	0	1	1	0
7	1	1	1	1	0	0	1	0	0	0	0	1	1	1	1	0	1	1	0

Tabela 1: Tablica wejść i wyjść (prawdy) dla układu głównego

Spostrzeżenia:

- Każda kolumna w tabeli ma swój duplikat, np. 1 i 4, 10 i 11, 13 i 16 itd. Jest to wynikiem symetrii każdego rozważanego wariantu emotikony (**Rysunek 1)**. Z tego powodu przy implementacji układu wystarczy kierować się tylko jedną połową wyświetlacza.
- Kratki wyświetlacza 2, 5 i 6 są zawsze białe, tak samo jak ich symetryczne odbicia 3, 7 i 8
- Kratka wyświetlacza 1 jest zawsze czarna, tak samo kratka 4

Wniosek:

- Wystarczy zaprojektować podukłady dla kratek wyświetlacza 1, 9, 10, 13 i 14 (lub 4, 11, 12, 15 i 16, jeśli rozważamy prawą połowę wyświetlacza).

5. Tablice Karnaugh

Przy projektowaniu układu skorzystano z pomocniczych podukładów zawierających bramki logiczne, które odpowiadają za odpowiednie przejście między stanami. Aby określić, jak powinny zostać zbudowane, stworzono tablice Karnaugh na podstawie Tabeli 1, a następnie na ich podstawie wyznaczono funkcje logiczne. Funkcje te zostały przekształcone za pomocą praw logicznych, tak aby używały tylko operatora *NAND*.

5.1. Prawa logiczne

$$\overline{AB} = \overline{A} + \overline{B}$$

$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

$$\overline{\cup A_i} = \cap \overline{A_i}$$

$$\overline{\cap A_i} = \cup \overline{A_i}$$

$$\overline{\overline{A}} = A$$

Wykorzystano tabele Karnaugh oraz wymienione wzory do wyznaczenia funkcji logicznych dla wyjść podukładów: 1, 9, 10, 13, 14 (liczby te odpowiadają danym kratkom wyświetlacza emotikon, zgodnie z Rysunkiem 2).

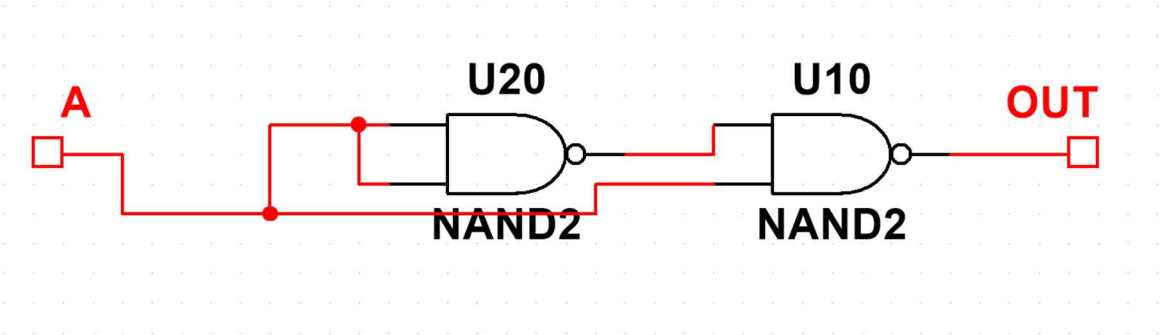
5.2. Wyjście ‘1’

A \ CB	00	01	11	10
0	1	1	1	1
1	1	1	1	1

Tabela 2: Tablica Karnaugh dla wyjścia ‘1’

W przypadku wyjścia ‘**1**’, funkcja logiczna powinna zawsze zwracać wartość logiczną 1:

$$\boxed{1} = 1$$



Rysunek 4: Implementacja wyjścia ‘1’

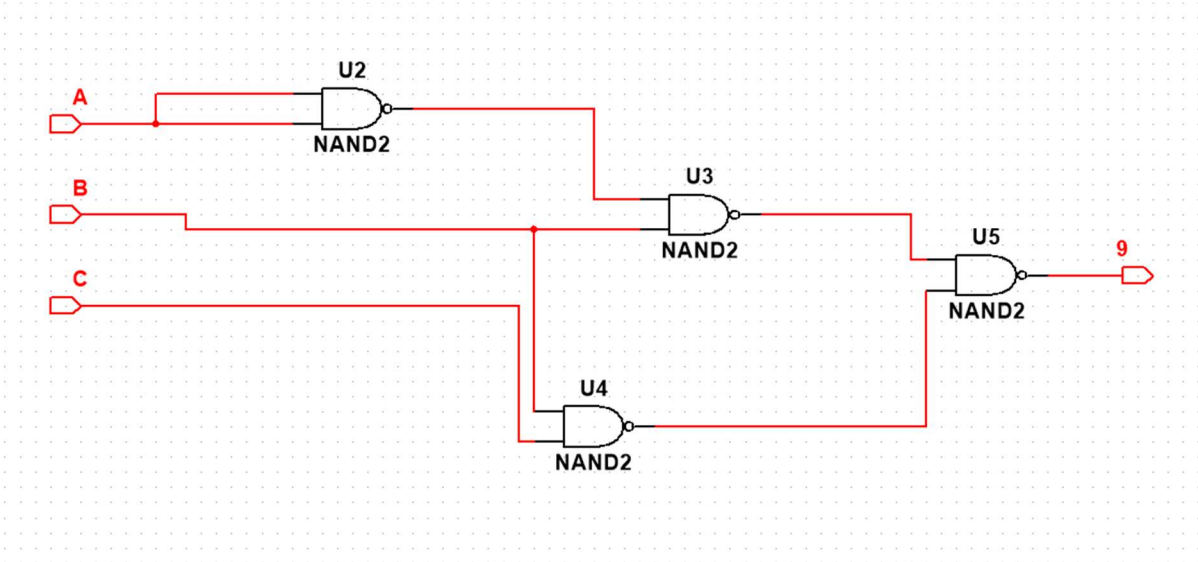
5.3. Wyjście ‘9’

A \ CB	00	01	11	10
0	0	1	1	0
1	0	0	1	0

Tabela 3: Tablica Karnaugh dla wyjścia ‘9’

$$\boxed{9} = B\,\overline{A} + C\,B$$

$$\boxed{9} = \overline{\overline{B\,\overline{A} + C\,B}} = \overline{B\,\overline{A} \cdot \overline{C\,B}}$$



Rysunek 5: Implementacja wyjścia ‘9’

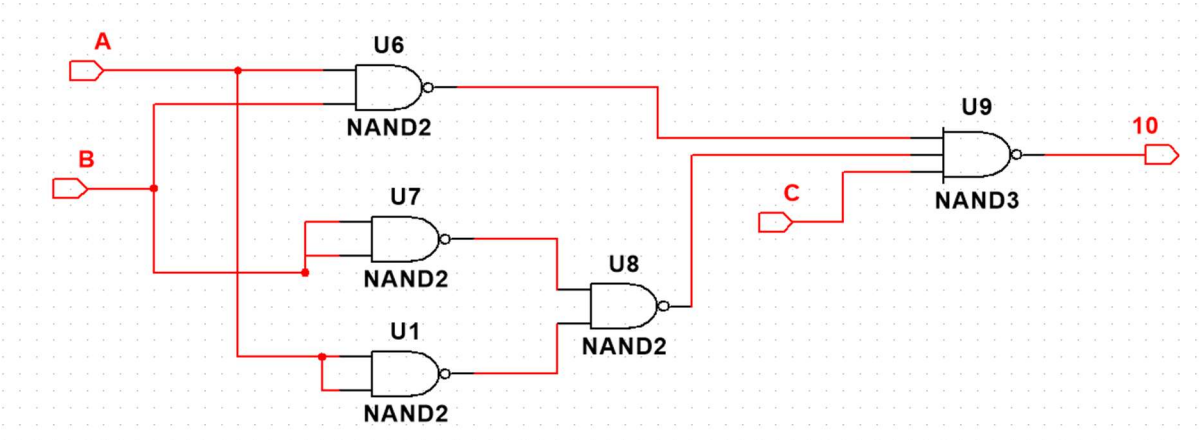
5.4. Wyjście ‘10’

A \ CB	00	01	11	10
0	1	1	0	1
1	1	1	1	0

Tabela 4: Tablica Karnaugh dla wyjścia ‘10’

$$\boxed{10} = \overline{C} + A\,B + \overline{B}\,\overline{A}$$

$$\boxed{10} = \overline{\overline{C} + A\,B + \overline{B}\,\overline{A}} = \overline{\overline{A\,B} \cdot \overline{\overline{B}\,\overline{A}} \cdot C}$$



Rysunek 6: Implementacja wyjścia ‘10’

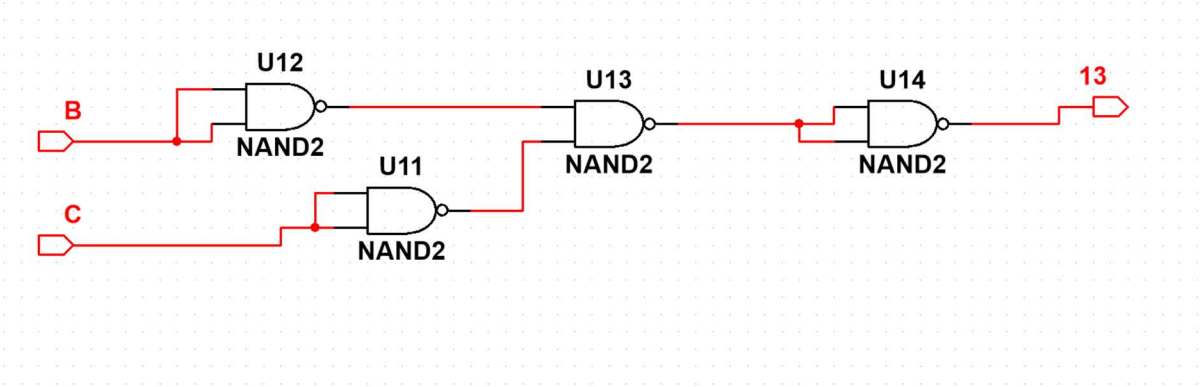
5.5. Wyjście ‘13’

A \ CB	00	01	11	10
0	1	0	0	0
1	1	0	0	0

Tabela 5: Tablica Karnaugh dla wyjścia ‘13’

$$\boxed{13} = \overline{C}\,\overline{B}$$

$$\boxed{13} = \overline{\overline{C}\,\overline{B}}$$



Rysunek 7: Implementacja wyjścia ‘13’

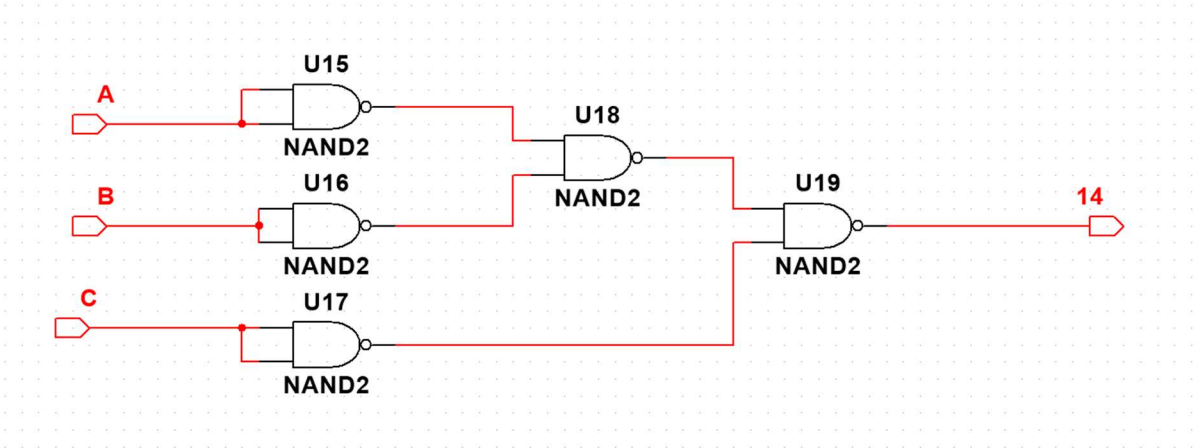
5.6. Wyjście ‘14’

A \ CB	00	01	11	10
0	1	0	1	1
1	0	0	1	1

Tabela 6: Tablica Karnaugh dla wyjścia ‘14’

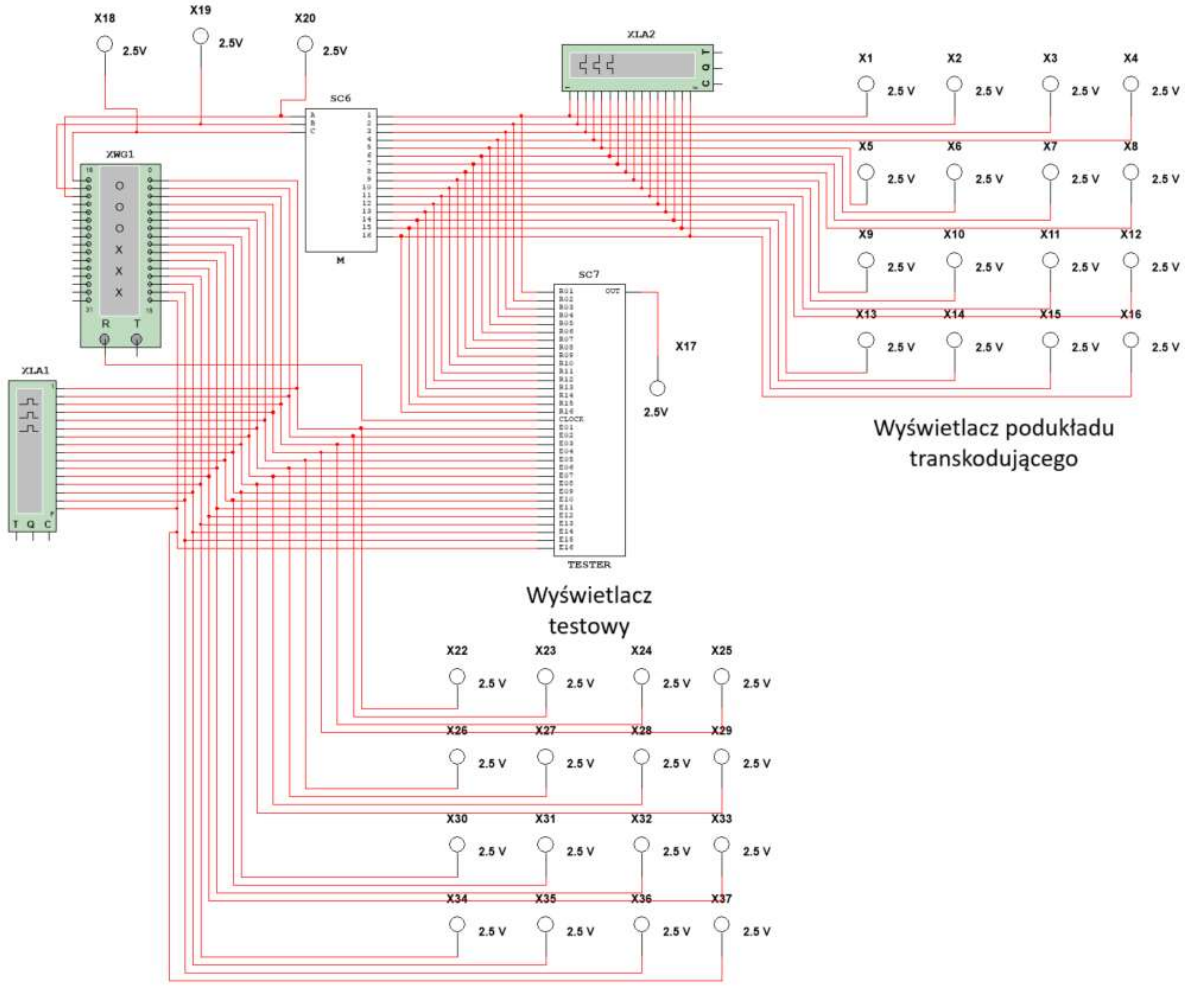
$$\boxed{14} = C + \overline{B}\,\overline{A}$$

$$\boxed{14} = \overline{\overline{C + \overline{B}\,\overline{A}}} = \overline{\overline{\overline{A}\,\overline{B}} \cdot \overline{C}}$$

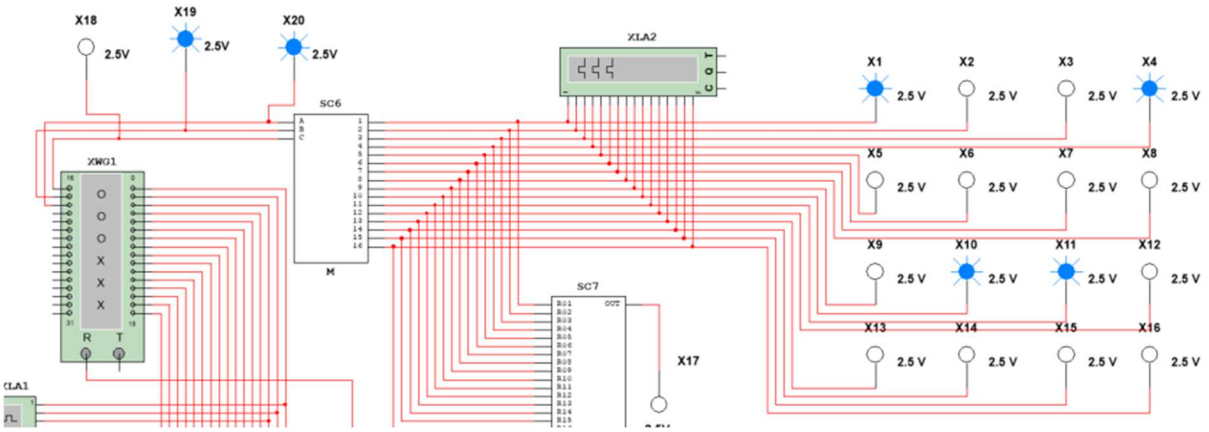


Rysunek 8: Implementacja wyjścia ‘14’

6. Implementacja układu w programie Multisim

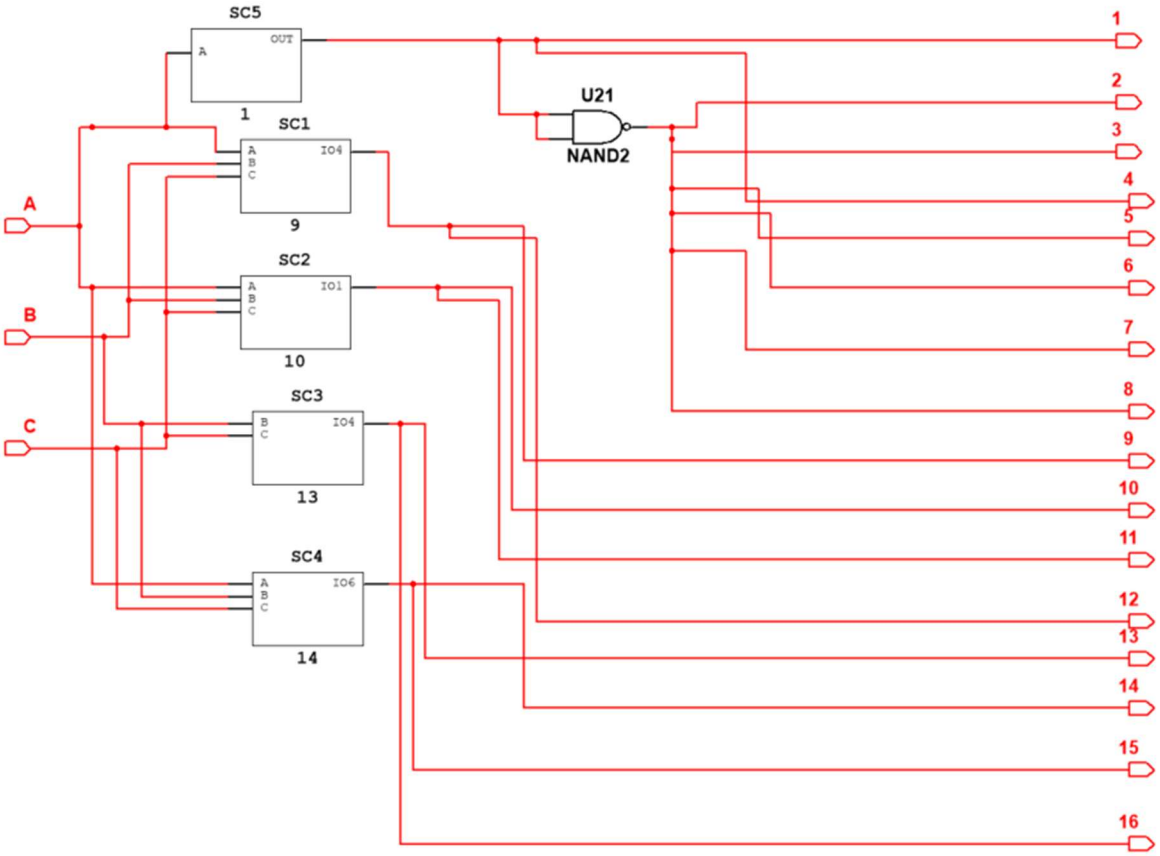


Rysunek 9: Implementacja układu głównego



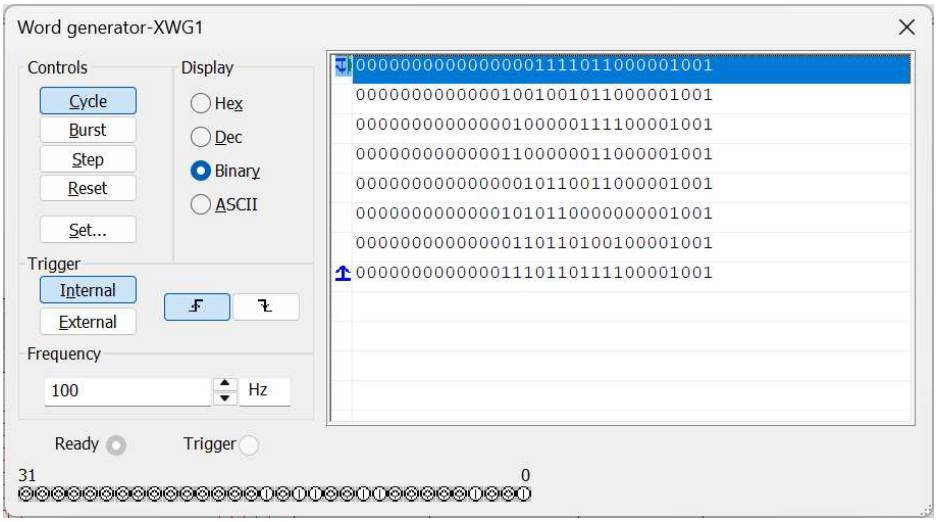
Rysunek 10: Przykład działania układu, konwersja kodu binarnego na emotikon

6.1. Podukład transkodujący



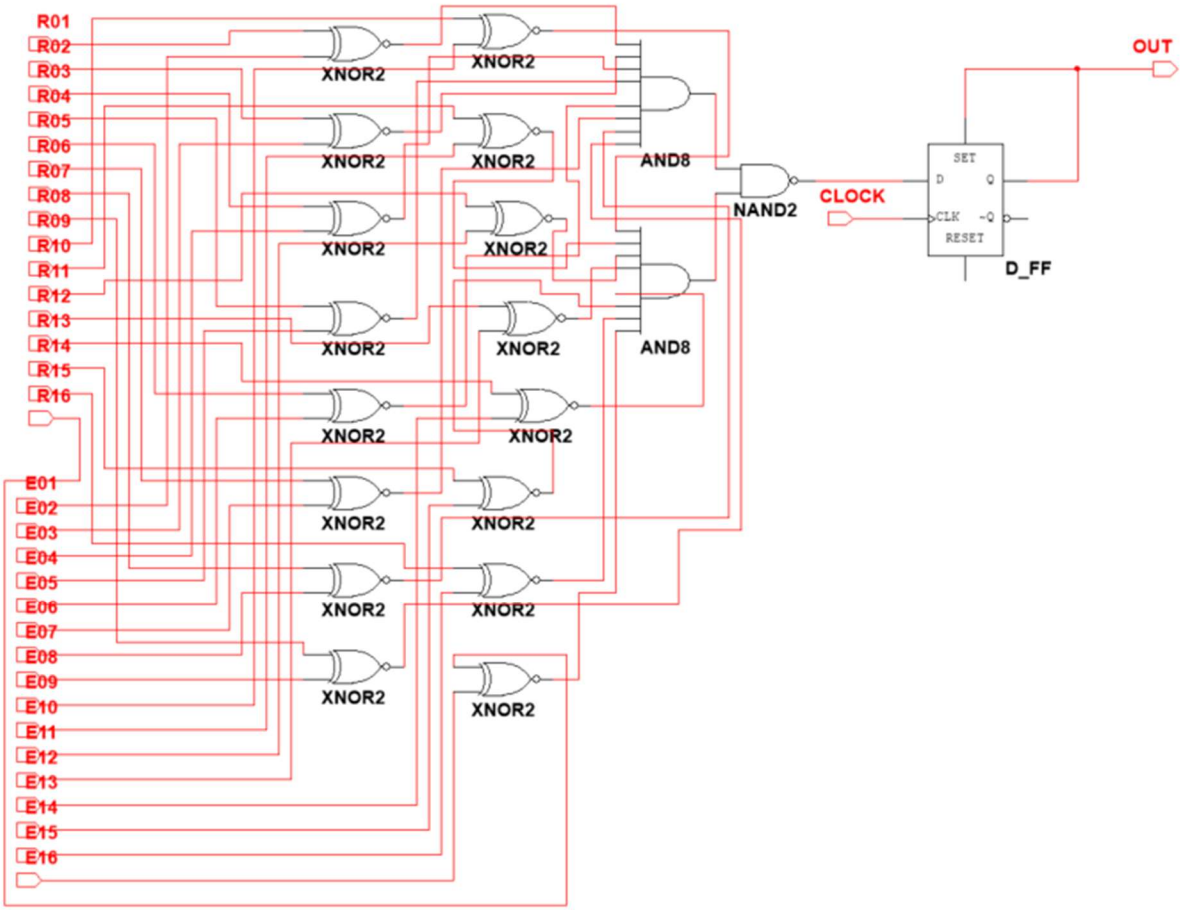
Rysunek 11: Implementacja podukładu transkodującego

6.2. Ustawienia generatora słów

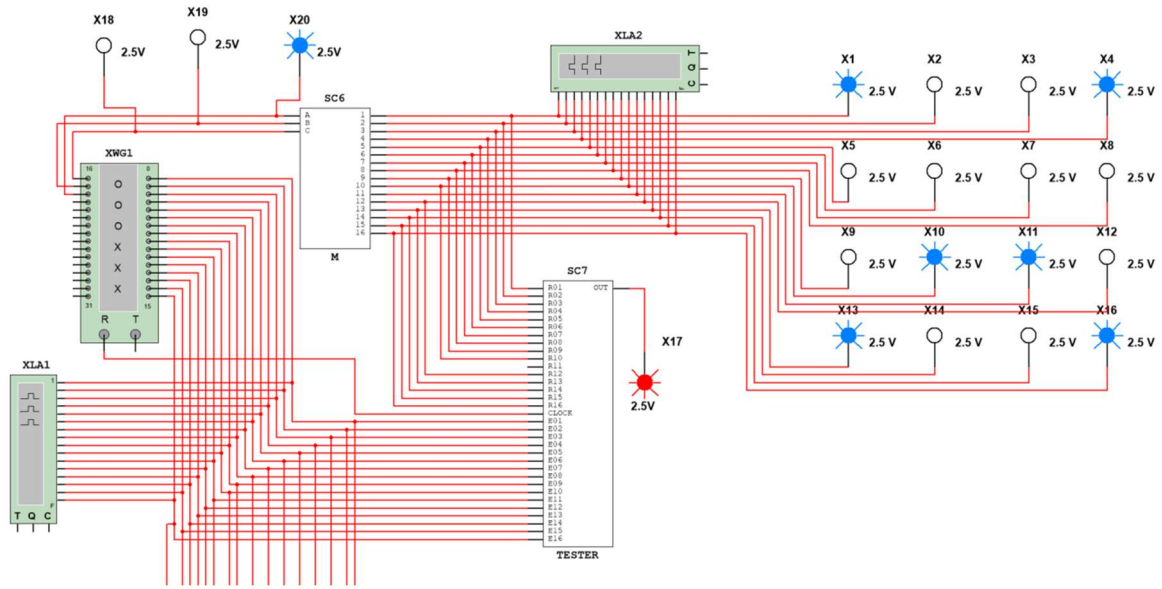


Rysunek 12: Ustawienia generatora słów

6.3. Podukład testujący



Rysunek 12: Implementacja podukładu testującego



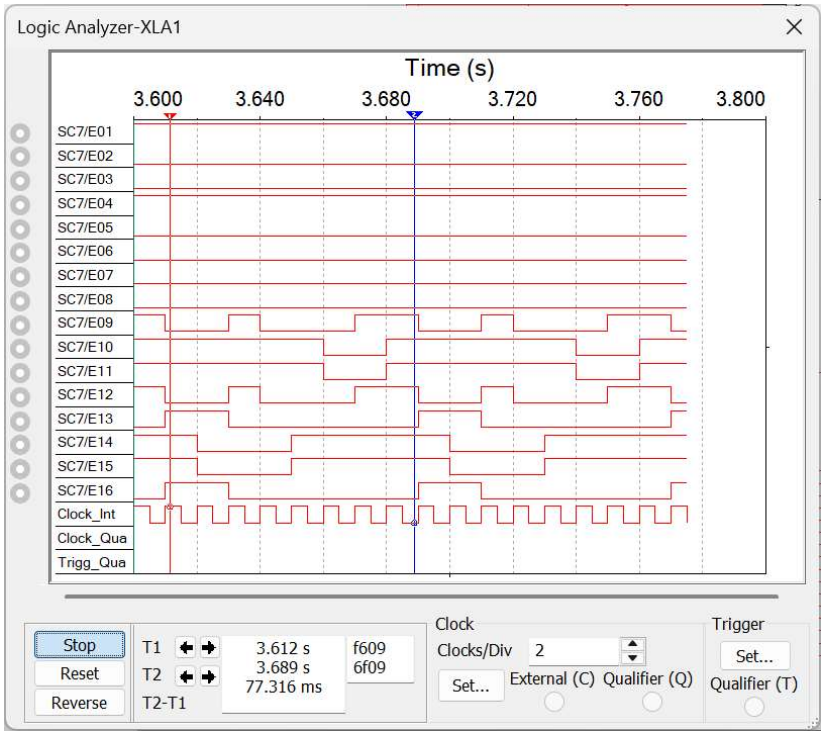
Rysunek 13: Układ z jednym odpiętym połączeniem. Wyjście podukładu testowego wskazuje błąd.

6.4. Wyniki analizatorów logicznych

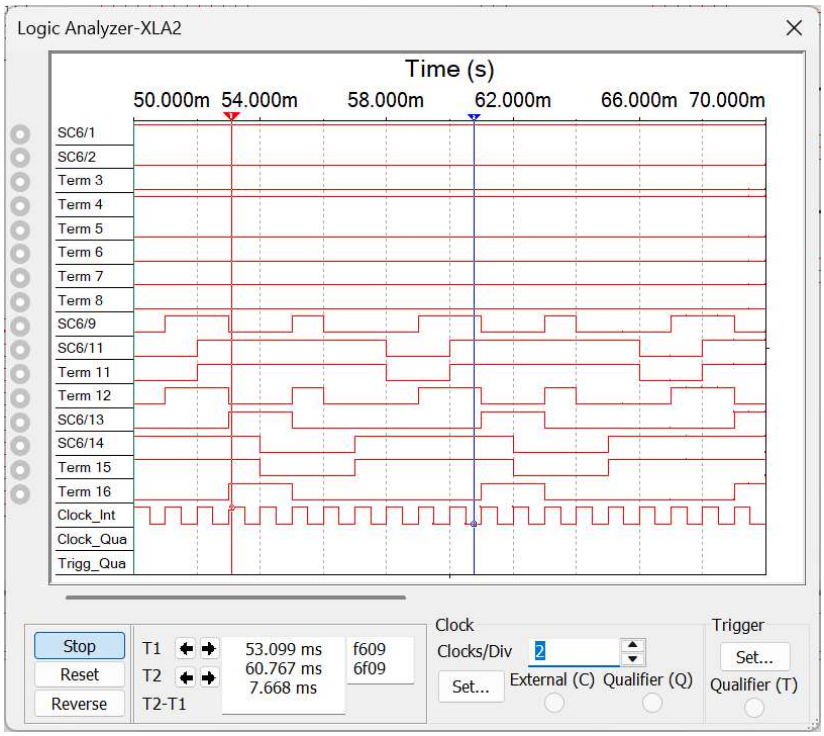
Do układu podpięte są 2 analizatory logiczne:

- XLA1 do wyjścia generatora słów – wynik oczekiwany,

- XLA2 do wyjścia podukładu transkodującego.



Rysunek 14: Wynik analizatora XLA1 (oczekiwany)

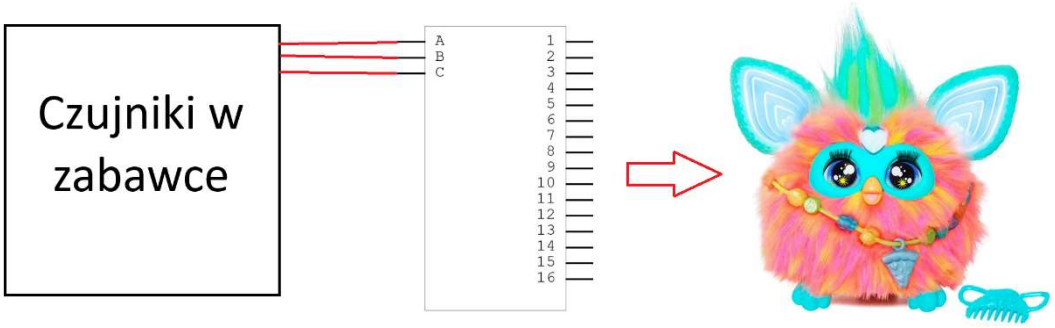


Rysunek 15: Wynik analizatora XLA2 (transkodowany)

7. Zastosowania

Taki układ można wykorzystać w:

- **Zabawkach** – zabawka może za pomocą tego układu reagować na działanie (np. uśmiechać się na pogłaskanie, śmiać na łaskotanie itp.)



Rysunek 16: Zastosowanie układu w zabawce

- **Wskaźnikach diagnostycznych** – Układ można wykorzystać w elektronice użytkowej lub urządzeniach AGD, gdzie zamiast kodu działania/błędu na wyświetlaczu 7 segmentowego można pokazywać status przez emotikony.

8. Wnioski

Zaprojektowany układ poprawnie realizuje funkcję dekodera trzybitowej liczby binarnej na graficzny układ emotikon, zgodnie z założeniami ćwiczenia. Implementacja oparta wyłącznie na bramkach NAND potwierdziła możliwość budowania złożonych układów logicznych z użyciem jednego typu bramki. Zastosowanie tablic Karnaugh pozwoliło na efektywną minimalizację funkcji logicznych, a symetria wyświetlacza umożliwiła uproszczenie konstrukcji układu.

Symulacje w programie Multisim potwierdziły poprawność działania projektu – zarówno w warunkach prawidłowych, jak i w przypadku wystąpienia błędów, co umożliwiło skuteczną weryfikację układu testowego. Ćwiczenie potwierdziło przydatność cyfrowych metod projektowych w tworzeniu prostych, interaktywnych rozwiązań wizualnych.