

Ćwiczenie 1.

Emil Siatka, Marek Swakoń, Patryk Górski, Piotr Pich

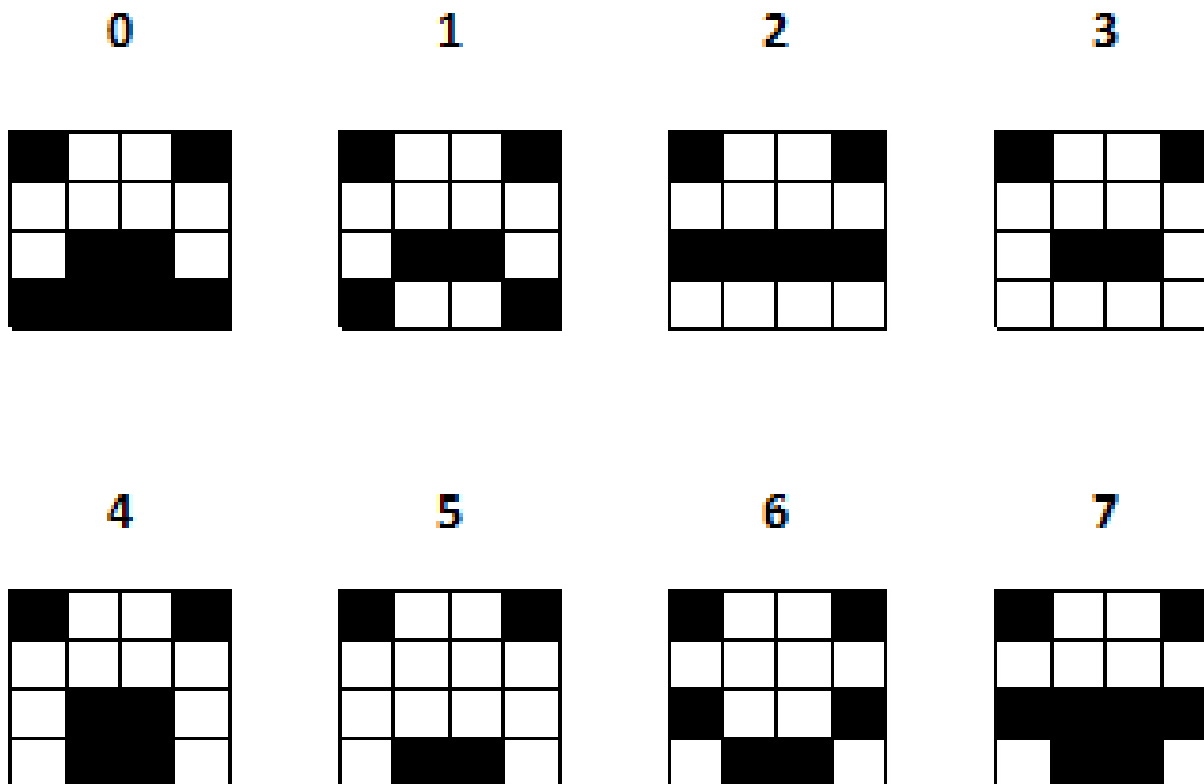
Data oddania: 03.04.2025

Spis treści

1	Wprowadzenie	1
2	Minimalizacja funkcji zdaniowych	2
3	Schemat układu	5
4	Stanowisko testujące	8
5	Podsumowanie oraz wnioski	8

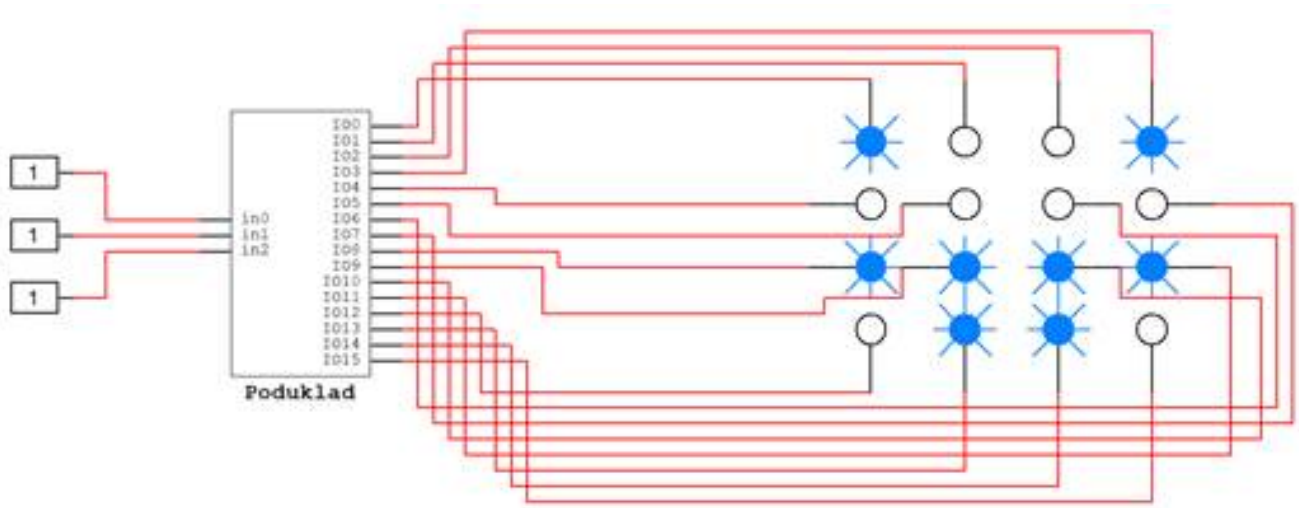
1 Wprowadzenie

Celem ćwiczenia jest zaprojektowanie bazującego na bramkach NAND układu kombinacyjnego realizującego transkoder trzybitowej liczby binarnej na układ graficzny emotikony wyświetlanej na szesnastu punktach, zgodnie z rysunkiem 1.



Rysunek 1: Docelowe wyniki działania transkodera

Interfejs układu będzie wyglądał wówczas identycznie do podukładu na rysunku 2.



Rysunek 2: Interfejs projektowanego układu

Projektowany układ przedstawić możemy jako układ którego sygnałami wyjściowymi jest 16 trójargumentowych funkcji zdaniowych P_0, P_1, \dots, P_{15} (gdzie każdy punkt wyświetlacza odpowiada wynikowi jednej funkcji) przyjmujących poniższe wartości:

Tabela 1: Tabela prawdy dla funkcji P_0, P_1, \dots, P_{15}

A	B	C		P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
0	0	0		1	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1
0	0	1		1	0	0	1	0	0	0	0	0	1	1	0	1	0	0	1
0	1	0		1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0
0	1	1		1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0
1	0	0		1	0	0	1	0	0	0	0	0	1	1	0	0	1	1	0
1	0	1		1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0
1	1	0		1	0	0	1	0	0	0	0	1	0	0	1	0	1	1	0
1	1	1		1	0	0	1	0	0	0	0	1	1	1	1	0	1	1	0

Aby zaprojektować układ, pozostaje zatem zminimalizować funkcje P_0, P_1, \dots, P_{15} (np. metodą tablic Karnaugh), przepisać zredukowaną formę wyłącznie przy użyciu bramek NAND i zbudować układ oraz stanowisko testowe w programie Multisim.

2 Minimalizacja funkcji zdaniowych

Z Tabeli 1. zauważyć można, że:

$$P_0 = P_3 = 1 \quad (1)$$

Oraz:

$$P_1 = P_2 = P_4 = P_5 = P_6 = P_7 = 0 \quad (2)$$

Dodatkowo:

- $P_8 = P_{11}$
- $P_9 = P_{10}$
- $P_{12} = P_{15}$
- $P_{13} = P_{14}$

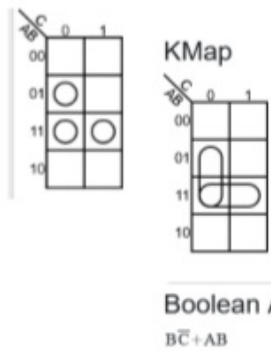
Do minimalizacji pozostaje nam zatem 6 różnych funkcji zdaniowych (w tym dwie funkcje stałe). Poniższe rysunki przedstawiają odpowiednie przekształcenia oraz tablice Karnaugh dla istotnych funkcji.

$$P_8 = P_{11} = B\bar{C} \cup AB$$

$$AB = \neg(\text{NAND}(A, B)) = \text{NAND}(\text{NAND}(A, B), \text{NAND}(A, B))$$

$$B\bar{C} = B \cap (\text{NAND}(C, C)) = \neg(\text{NAND}(B, \text{NAND}(C, C))) =$$

$$= \text{NAND}(\text{NAND}(B, \text{NAND}(C, C)), \text{NAND}(B, \text{NAND}(C, C)))$$

$$AB \cup B\bar{C} = \text{NAND}(\text{NAND}(AB, AB), \text{NAND}(B\bar{C}, B\bar{C}))$$


Boolean Algebra
 $B\bar{C} + AB$

Rysunek 3: Przekształcenia dla funkcji P_8 i P_{11}

$$P_9 = P_{10} = \bar{A} \cup \bar{B}\bar{C} \cup BC$$

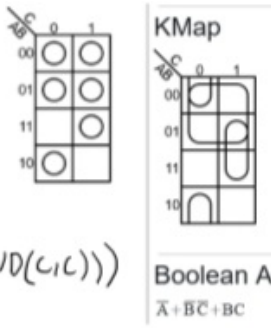
$$\bar{A} = \text{NAND}(A, A)$$

$$\bar{B}\bar{C} = \text{NAND}(B, B) \cap \text{NAND}(C, C) = \neg(\text{NAND}(\text{NAND}(B, B), \text{NAND}(C, C)))$$

$$= \text{NAND}(\text{NAND}(\text{NAND}(B, B), \text{NAND}(C, C)), \text{NAND}(\text{NAND}(B, B), \text{NAND}(C, C)))$$

$$BC = \text{NAND}(\text{NAND}(B, C), \text{NAND}(B, C))$$

$$\bar{A} \cup \bar{B}\bar{C} = \text{NAND}(\text{NAND}(A, A), \text{NAND}(\bar{B}\bar{C}, \bar{B}\bar{C}))$$

$$\bar{A} \cup \bar{B}\bar{C} \cup BC = \text{NAND}(\text{NAND}(\bar{A} \cup \bar{B}\bar{C}, \bar{A} \cup \bar{B}\bar{C}), \text{NAND}(BC, BC))$$


Boolean Algebra
 $\bar{A} + \bar{B}\bar{C} + BC$

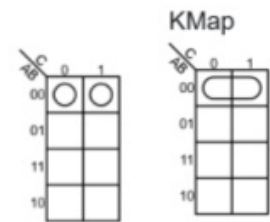
Rysunek 4: Przekształcenia dla funkcji P_9 i P_{10}

$$P_{12} = P_{15} = \bar{A} \bar{B}$$

$$\bar{A} \bar{B} = \text{NAND}(A, A) \cap \text{NAND}(B, B) =$$

$$= \neg(\text{NAND}(\text{NAND}(A, A), \text{NAND}(B, B))) =$$

$$= \text{NAND}(\text{NAND}(\text{NAND}(A, A), \text{NAND}(B, B)), \text{NAND}(\text{NAND}(A, A), \text{NAND}(B, B)))$$



Boolean Algebra
 $\bar{A} \bar{B}$

Rysunek 5: Przekształcenia dla funkcji P_{12} i P_{15}

$$P_{13} = P_{14} = A + \bar{B} \bar{C}$$

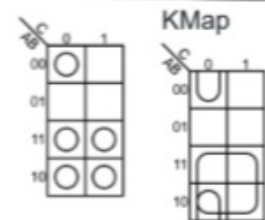
$$\bar{B} \bar{C} = \text{NAND}(B, B) \cap \text{NAND}(C, C) =$$

$$= \neg(\text{NAND}(\text{NAND}(B, B), \text{NAND}(C, C))) =$$

$$= \text{NAND}(\text{NAND}(\text{NAND}(B, B), \text{NAND}(C, C)), \text{NAND}(\text{NAND}(B, B), \text{NAND}(C, C)))$$

$$A + \bar{B} \bar{C} = \text{NAND}(\text{NAND}(A, A), \text{NAND}(\bar{B} \bar{C}, \bar{B} \bar{C})) =$$

$$= \text{NAND}(\text{NAND}(A, A), \text{NAND}(\text{NAND}(B, B), \text{NAND}(C, C)))$$



Boolean Algebra
 $A + \bar{B} \bar{C}$

Rysunek 6: Przekształcenia dla funkcji P_{13} i P_{14}

const 1

$$1 = A \cup \bar{A} = A \cup (\text{NAND}(A, A)) = \text{NAND}(\text{NAND}(A, A), \text{NAND}(\text{NAND}(A, A), \text{NAND}(A, A)))$$

const 0

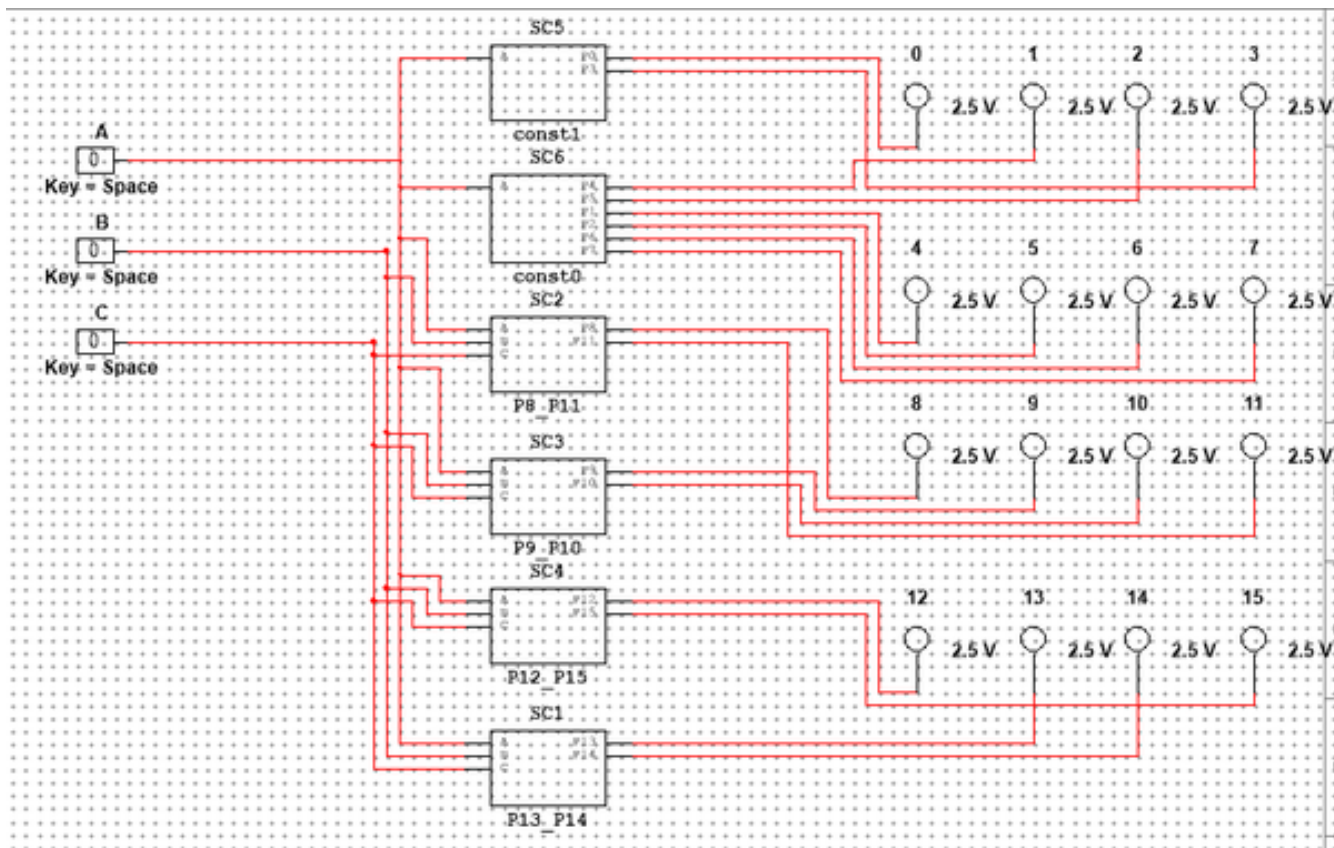
$$0 = A \cap \bar{A} = A \cap (\text{NAND}(A, A)) = \text{NAND}(\text{NAND}(A, A), \text{NAND}(\text{NAND}(A, A), \text{NAND}(A, A)))$$

Rysunek 7: Przekształcenia dla funkcji stałych

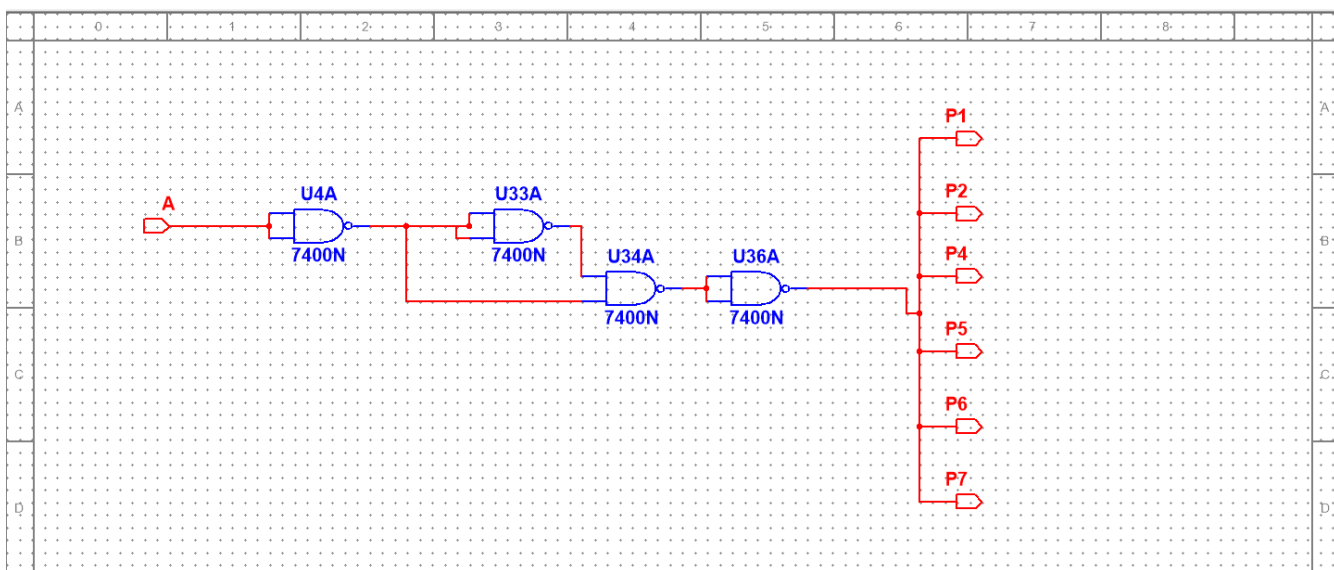
Jak można zaobserwować na rysunkach 3-7 zredukowane formy, przy użyciu odpowiednich praw logiki, przekształciliśmy na wzory wykorzystujące wyłącznie bramki NAND, gotowe do zastosowania w budowie finalnego układu.

3 Schemat układu

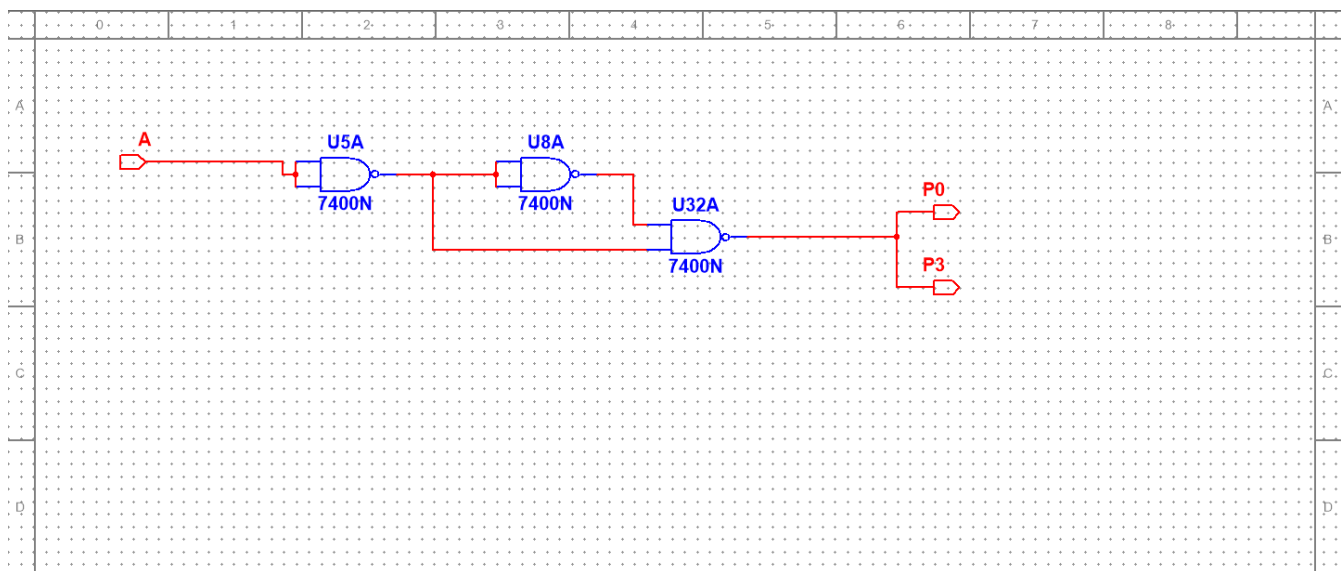
Korzystając z wyprowadzonych na rysunkach 3-7 wzorów na sygnały wyjściowe, byliśmy gotowi aby stworzyć schemat układu w programie Multisim.



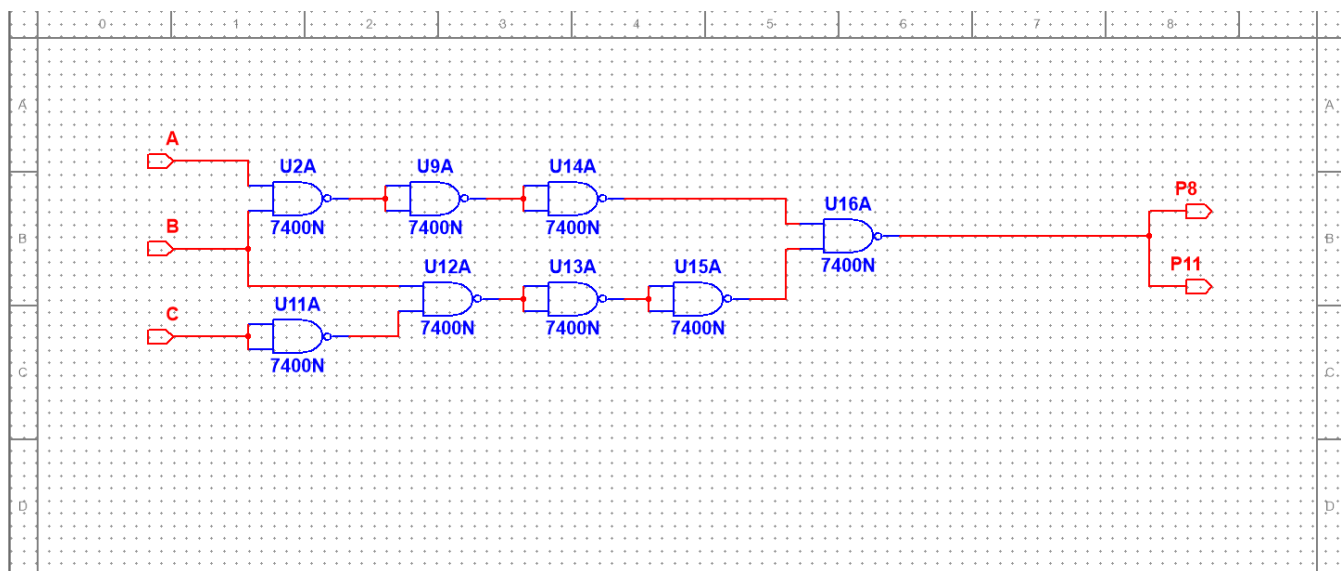
Rysunek 8: Schemat układu z podziałem na bramki w programie Multisim



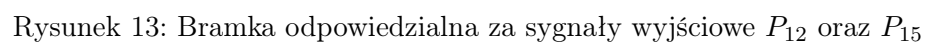
Rysunek 9: Bramka odpowiedzialna za sygnały wyjściowe P_1 do P_7

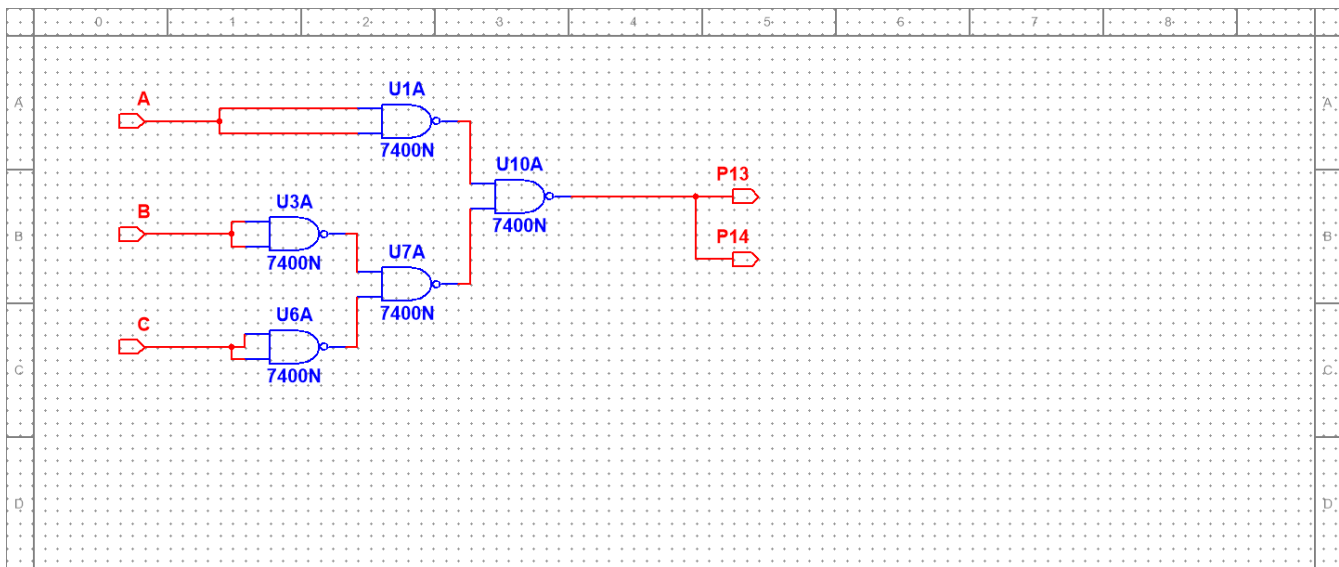


Rysunek 10: Bramka odpowiedzialna za sygnały wyjściowe P_0 oraz P_3



Rysunek 11: Bramka odpowiedzialna za sygnały wyjściowe P_8 oraz P_{11}





Rysunek 14: Bramka odpowiedzialna za sygnały wyjściowe P_{13} oraz P_{14}

4 Stanowisko testujące

Układ testujący został zaprojektowany w celu automatycznej weryfikacji poprawności działania głównego układu transkodera. Jego zadaniem jest porównywanie wyników generowanych przez właściwy układ z wartościami referencyjnymi, uzyskanymi z równolegle działającego testera. Składa się on z następujących elementów:

- Generator słów (XWG1) – generuje wszystkie możliwe kombinacje sygnałów wejściowych A, B, C (od 000 do 111). Sygnały te są jednocześnie podawane do testowanego układu transkodera oraz do układu testującego, co zapewnia synchroniczne porównanie wyników.
- Tester (subcircuit o nazwie „Tester”) – zbudowany został z 8 sekcji, z których każda odpowiada jednej z możliwych kombinacji wejść A, B, C. Każda sekcja zawiera osobny zestaw bramek NAND, które na podstawie ustalonej logiki generują oczekiwane wyjście dla danej kombinacji. Wyjścia z poszczególnych sekcji są następnie łączone przy użyciu bramek AND oraz OR w taki sposób, aby zapewnić, że w danym momencie aktywna jest tylko jedna sekcja – dokładnie ta, która odpowiada aktualnemu stanowi wejść. Dzięki temu można w pełni dynamicznie odwzorować prawidłowe działanie transkodera bez potrzeby ręcznej zmiany konfiguracji.
- Bramki XOR (7486) – do każdego z 16 wyjść podłączona jest bramka XOR, która porównuje sygnał wyjściowy z transkodera z odpowiednim wyjściem z układu testującego. Jeśli wartości się różnią, oznacza to błąd – wtedy zapala się odpowiednia dioda LED.
- XLA1 (Logic Analyzer) – pełni rolę pomocniczą i umożliwia obserwację przebiegów logicznych w czasie rzeczywistym. Pozwala to na dokładniejszą analizę pracy całego układu oraz identyfikację ewentualnych problemów.
- Wyświetlacz siedmiosegmentowy (HEX) – połączony z wejściami A, B i C za pośrednictwem układu dekodera 7447N, wyświetla aktualną kombinację testowaną przez układ (w postaci liczby binarnej od 000 do 111), co umożliwia łatwe śledzenie, który przypadek jest aktualnie sprawdzany.

Dzięki zastosowaniu generatora słów i automatycznego testera możliwe jest szybkie i powtarzalne testowanie wszystkich możliwych wejść bez ingerencji użytkownika. Układ reaguje natychmiastowo na każdą nieprawidłowość, co znacznie przyspiesza proces weryfikacji poprawności działania transkodera.

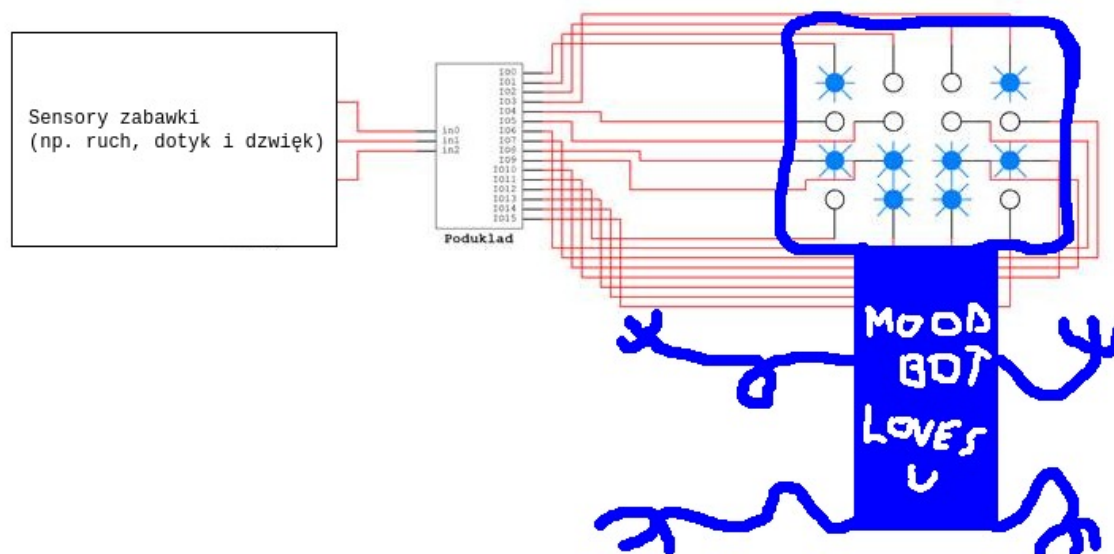
5 Podsumowanie oraz wnioski

Kluczowym etapem ćwiczenia była minimalizacja funkcji logicznych przy użyciu tablic Karnaugh, a następnie przekształcenie wszystkich wyrażeń do formy realizowalnej wyłącznie bramkami NAND, stosując prawa logiki i przekształcenia algebraiczne. Wykonanie tego etapu w pierwszej kolejności uprościło znacznie stworzenie schematu układu,

umożliwiając zbudowanie w programie Multisim i przetestowanie jego finalnej wersji bez konieczności tworzenia bardziej złożonych prototypów. Słabą stroną tego podejścia jest brak możliwości wykrycia błędów na wczesnym etapie projektowania, potencjalnym rozwiązaniem tego problemu byłyby np. programistyczne testy logiki poszczególnych bramek, co z kolei dodatkowo skomplikowałoby proces weryfikacji działania.

Praktyczne zastosowania zaprojektowanego układu obejmują:

- Systemy wyświetlania prostych ikon w urządzeniach o niskiej rozdzielczości (np. wyświetlacze LED).
- Urządzenia mające charakter informacji publicznej, gdzie przełamanie bariery językowej jest porządane, np. znaki ostrzegawcze, komunikaty o opóźnieniach, jakości powietrza itp.
- Interfejsy użytkownika w układach embedded, np. w elektronice użytkowej lub zabawkowej.



Rysunek 15: Przykładowe zastosowanie układu

Układ stanowi przykład efektywnego wykorzystania podstawowych elementów logicznych do realizacji czytelnych funkcji wizualnych, co może znaleźć zastosowanie w projektach wymagających prostoty i niskiego poboru mocy.