|  |  |
| --- | --- |
| **Technika cyfrowa** | |
| Temat ćwiczenia:  **Minimalizacja i praktyczna realizacja złożonych funkcji logicznych** | Nr ćwiczenia:  2 |
| Wykonawca:  **Tomasz Wiewióra** |

1) Układ cyfrowy zwiększający o jeden trzybitową nieujemną liczbę całkowitą:

Aby zwiększać liczbę zapisaną binarnie należy dodawać do niej jedynkę zapisaną także binarnie. Dodawanie binarne pokazuje poniższa tabelka:

Tabela 1 Dodawanie binarne dwóch cyfr

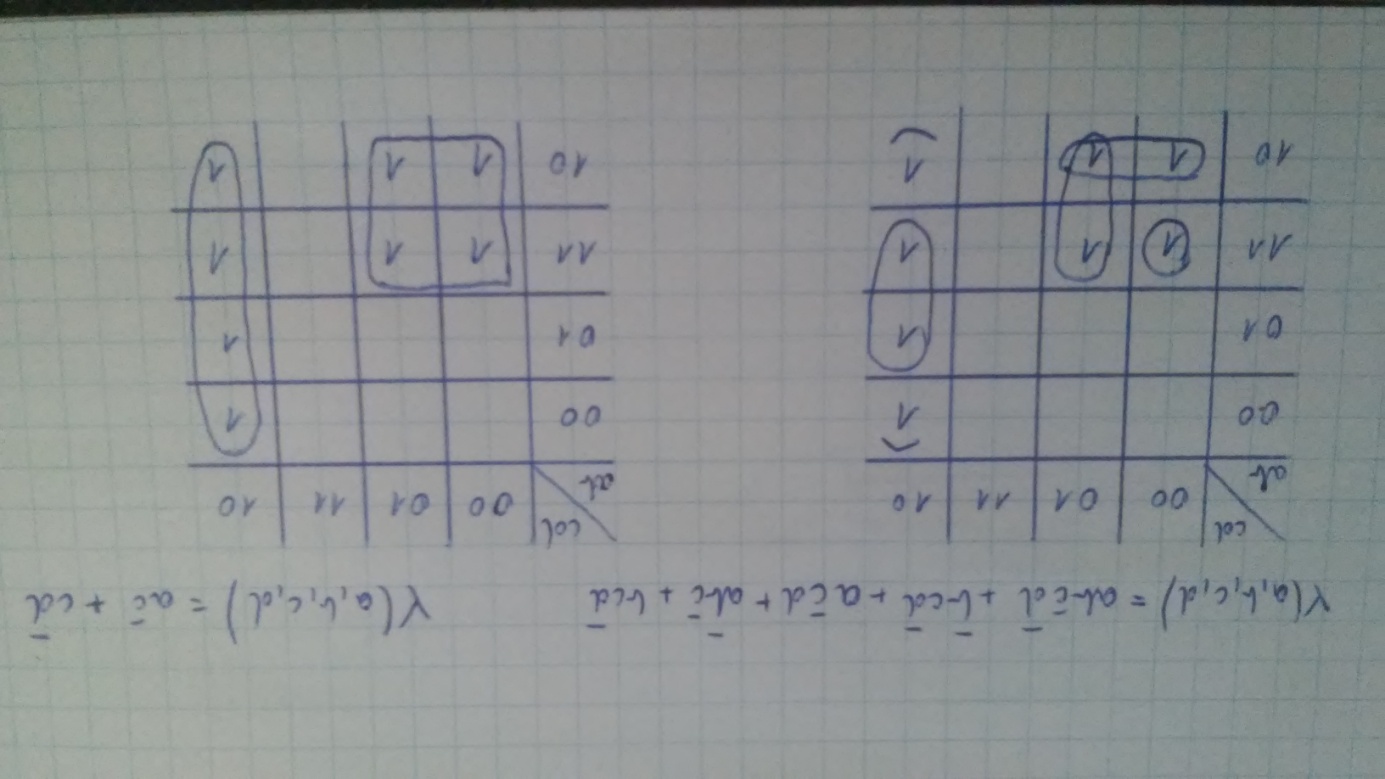
|  |  |  |
| --- | --- | --- |
| **x+y** | **Wynik** | **Przeniesienie** |
| 0+0 | 0 | 0 |
| 0+1 | 1 | 0 |
| 1+0 | 1 | 0 |
| 1+1 | 0 | 1 |



Rysunek 1 Układ cyfrowy inkrementujący liczbę 3-bitową

Teraz wystarczy tylko zauważyć, że do określenia wyniku takiego dodawania należy użyć bramki XOR, natomiast do określenia statusu przeniesienia należy użyć bramki AND.

Zbudowany przeze mnie układ składa się z generatora słów 3-bitowych, trzech diod, które świecąc się w odpowiedniej kolejności kodują zinkrementowaną liczbę z wejścia, dwóch bramek AND oraz trzech bramek XOR. Generator słów 3-bitowych generuje liczby o 0 do 7. Bramki XOR realizują dodawanie binarne. Te same sygnały, które wchodzą do bramek XOR przechodzą także przez bramki AND, aby uwzględnić ewentualną resztę dodawania w dalszych obliczeniach.

image003.png2) Minimalizowanie funkcji logicznej podanej w zadaniu oraz praktyczna jej realizacja w programie Multisim:

Rysując tablicę Karnaugh'a dla podanej w zadaniu funkcji logicznej widzimy, że nie jest ona optymalna i możemy ją w prosty sposób uprościć. Możemy o wiele prościej połączyć jedynki ze sobą w implikanty, a przez to uprościć funkcję logiczną.

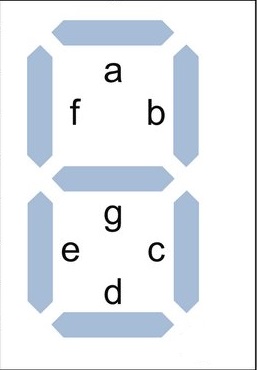
Rysunek 2 Tablica Karnaugh'a dla podanej funkcji logicznej oraz jej uproszczenie



Rysunek 3 Układ bramek wykonujących funkcję logiczną podaną w zadaniu

Do wykonania powyższego układu realizującego podaną w zadaniu funkcję logiczną użyłem dwóch bramek NOT, które negują sygnał c oraz d. Dwóch bramek AND oraz jednej bramki OR. Po skróceniu funkcji logicznej okazuje się, że sygnał na wejściu b nie ma żadnego wpływu na wyjście układu.

3) Transkoder czterobitowych cyfr (od 0 do 9) na pojedynczy wyświetlacz siedmiosegmentowy:

 Wyświetlacz siedmiosegmentowy składa się z siedmiu segmentów. Przeznaczony jest do wyświetlania cyfr dziesiętnych. Istnieją także modele, które mają ósmy segment będący kropką. Wyświetlacze siedmiosegmentowe mogą być wykonane w różnych technologiach, najczęściej jest to zestaw oddzielnych diod LED. Wyświetlacze mają wyprowadzenia sterujące oddzielnie każdym segmentem.

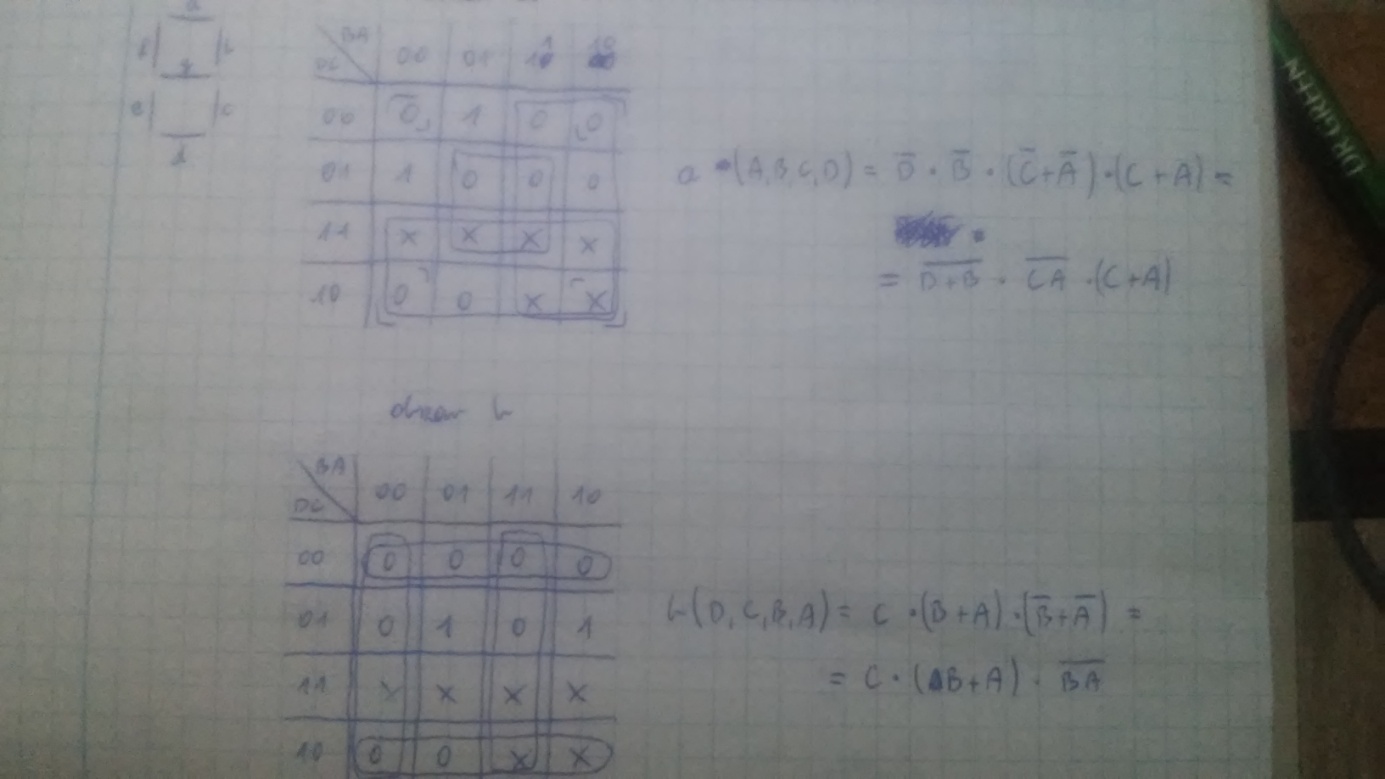
Rysunek 4 Schemat wyświetlacza 7-segmentowego z podpisanymi segmentami

Wyświetlacz 7-segmentowy ze wspólną anodą zaświeca odpowiednią diodę LED, kiedy na wejściu oznaczonym tą samą literą podamy sygnał niski, czyli logiczne "0", a gasi gdy na odpowiednim wejściu jest stan wysoki, czyli logiczne "1". Aby zakodować w ten sposób każdą cyfrę, należy każdej z cyfr przyporządkować odpowiednią sekwencję jedynek oraz zer logicznych na wejściach wyświetlacza. W moim przykładzie zastosowałem wyświetlacz 7-segmentowy ze wspólną anodą.

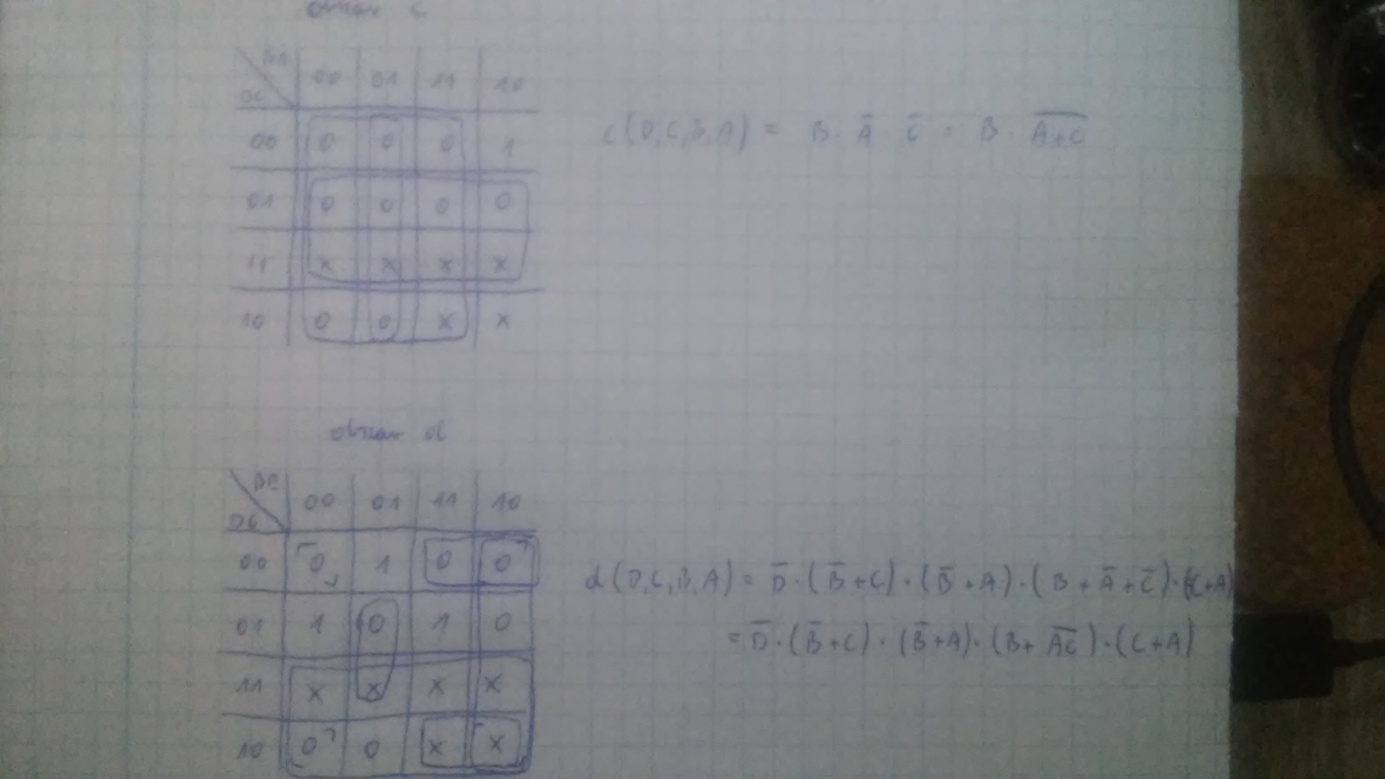
Tabela 2 Tabela definiująca sygnały wyjściowe w funkcji sygnałów wejściowych

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Wejście** | | | | **Wyjście** | | | | | | | **Cyfra** |
| **D** | **C** | **B** | **A** | **a** | **b** | **c** | **d** | **e** | **f** | **g** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | **0** |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | **1** |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | **2** |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | **3** |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | **4** |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | **5** |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | **6** |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | **7** |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **8** |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | **9** |

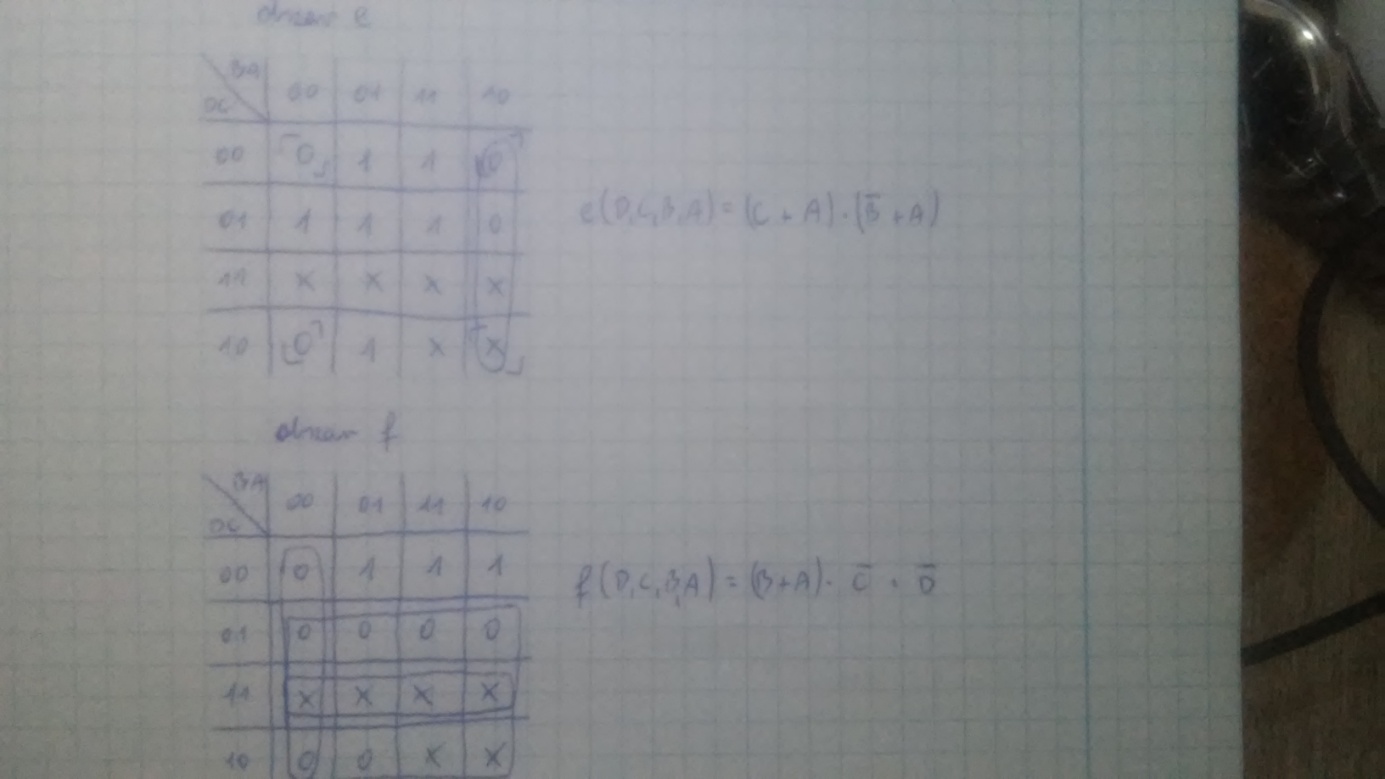
Mając tak przygotowaną tabelę można stworzyć układy bramek logicznych, które będą wykonywać odpowiednie funkcje dla każdego segmentu. Aby to wykonać korzystamy z tablic Karnaugh'a.



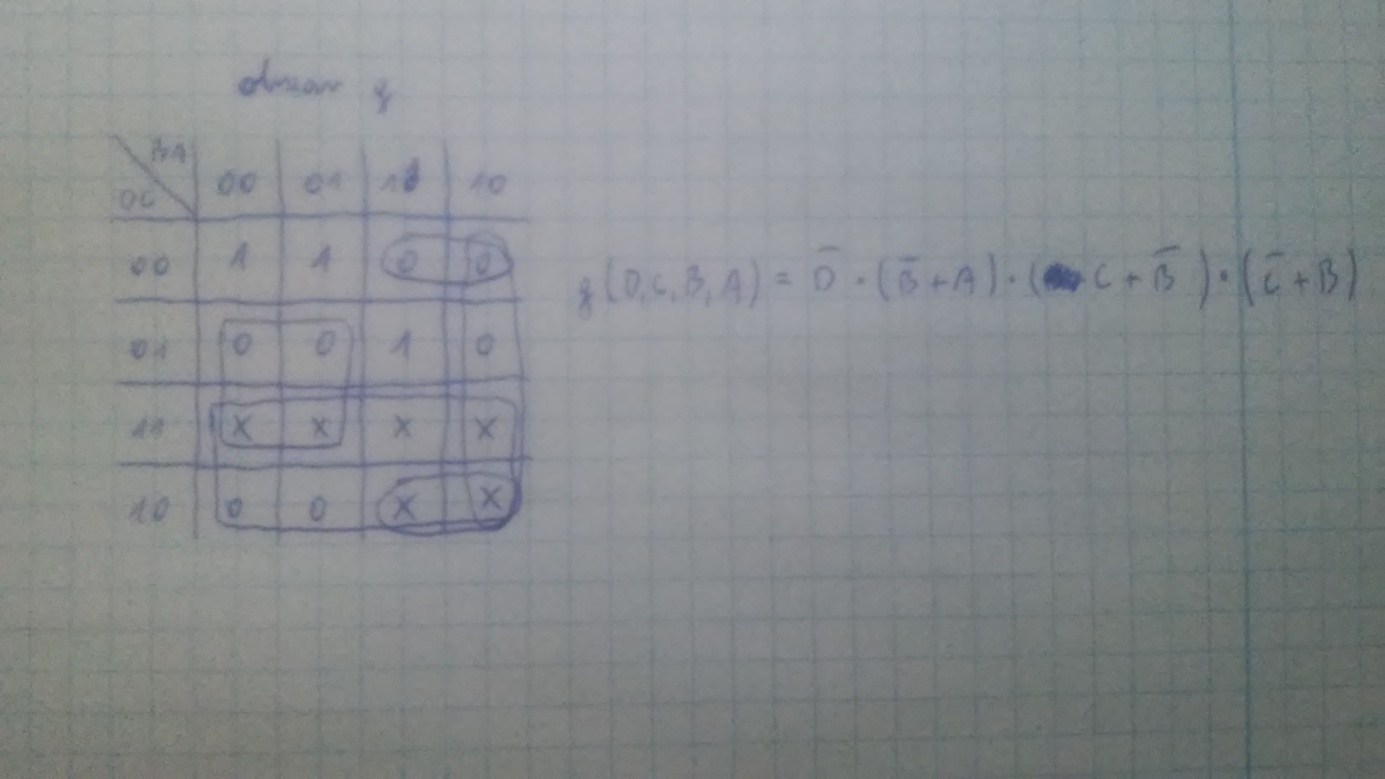
Rysunek 5 Tablice Karnaugh dla segmenów a oraz b



Rysunek 6 Tablice Karnaugh dla segmenów c oraz d



Rysunek 7 Tablice Karnaugh dla segmentów e oraz f



Rysunek 8 Tablice Karnaugh dla segmentu g

Mając gotowe funkcję logiczne, wystarczy tylko przedstawić je w programie w postaci odpowiednich bramek logicznych i zaprojektować układ w programie Multisim.



Rysunek 9 Transkoder 4-bitowych cyfr na pojedynczy wyświetlacz siedmiosegmentowy

Każdy kolor w tym schemacie odpowiada funkcji odpowiedzialnej za inny segment wyświetlacza. Na wejście układu generator słów podaje kolejne cyfry od 0 do 9 zakodowanej w 4 bitach.

4) Wnioski:

Tablica Karnaugh jest bardzo pomocnym narzędziem do minimalizowania skomplikowanych funkcji logicznych oraz do tworzenia zupełnie nowych funkcji, które po poprawnym zaimplementowaniu w postaci bramek logicznych, mogą tworzyć skomplikowane układy cyfrowe.