Czterowejściowa komórka PAL

- technologia CMOS

Opracowali:

Krzysztof Boroń

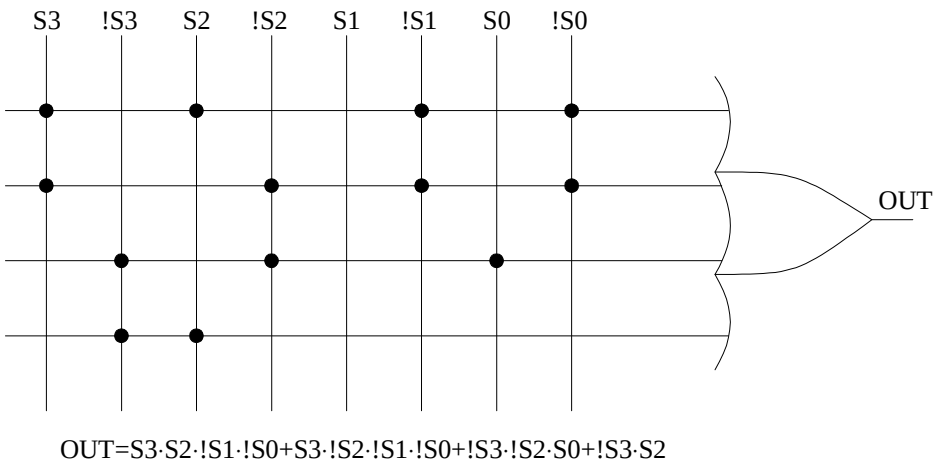
Grzegorz Bywalec

Kraków 20.I.2003

Naszym zadaniem było stworzenie projektu komórki PAL16V8. Komórka w odróżnieniu od pierwowzoru miała posiadać cztery wejścia. Układ miał mieć dwie wersje: z wyjściem kombinacyjnym oraz rejestrowym. Projekt ma charakter dydaktyczny – zastosowana technologia nie pozwala na implementację matrycy połączeniowej w pełnym wymiarze tego słowa. W naszym układzie przyjęliśmy że programowanie będzie się odbywać na drodze umieszczania w odpowiednich punktach matrycy przelotek zwierających odpowiednie linie sygnałowe.

Komórka PAL.
Komórka PAL przez długi czas była najpopularniejszą strukturą programowalną. Składała się z następujących bloków:

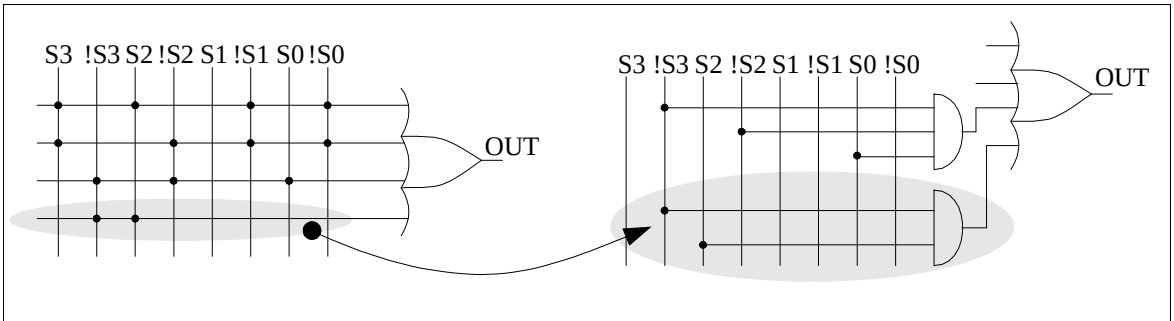
- Buforów wejściowych sterujących matrycą połączeniową – bufory dostarczały na matrycę równocześnie sygnał prosty i zanegowany. W naszym projekcie pomimo zastosowania tylko czterech wejść matryca ma osiem wejściowych linii sygnałowych – cztery dla sygnałów prostych oraz cztery dla ich zanegowanych odpowiedników
- Programowalnej matrycy połączeniowej odpowiedzialnej za realizację zadanej funkcji logicznej
- Współpracującego z matrycą układu bramek AND i OR realizujących zadaną funkcję
- Bloku wyjściowego który w zależności od wersji może być inwerterem lub przerzutnikiem.



Rysunek 1 Schemat komórki logicznej PAL. Pominięto bufory-inwertery wejściowe oraz układ wyjściowy.

Na powyższym rysunku przedstawiono schemat omawianego układu. Pominięto na nim bufory wejściowe sterujące pionowymi liniami sygnałowymi. Na schemacie tym nie występują również bramki AND a jedynie jedna wielowejściowa bramka OR. W rzeczywistości poziome linie podłączone do wejść bramki OR symbolizują aż dwa elementy układu. Pierwszy i chyba najważniejszy jest “ukryty” w punktach przecięcia się linii poziomych z pionowymi. W tych punktach znajduje się programowalny punkt umożliwiający ich łączenie. Drugim elementem kryjącym się pod postacią poziomej linii sygnałowej jest bramka AND realizująca iloczyn wszystkich sygnałów podłączonych do danej linii. Taka postać schematu sugeruje że mamy do czynienia z iloczynem na drucie – jeżeli na wszystkich “przyłączonych” liniach panuje stan wysoki wówczas na wejście bramki OR zostanie podany stan wysoki a tym samym na wyjściu tej bramki również pojawi się stan wysoki.
Ponieważ nasz układ miał być zrealizowany z wykorzystaniem technologii CMOS nie możliwa była bezpośrednia implementacja “iloczynu na drucie”. Zamiast tego musieliśmy zastosować, tym razem jawne, wielowejściowe bramki AND zrealizowane w sposób charakterystyczny dla technologii CMOS. Proszę zauważyć że “iloczyn na drucie” pozwala nie tylko na programowanie układu ale również na zmianę jego budowy – dysponujemy tylu-wejściowymi bramkami AND jakich w danym

przypadku potrzebujemy (jeżeli wykonamy tylko dwa połączenia na matrycy do danego wejścia bramki OR realizujemy tym samym dwuwejściową bramkę AND). W naszej realizacji nie mamy takiej możliwości. Wielkość bramek AND (ilość wejść) jest zależna od ich fizycznej budowy. Przeprowadźmy prostą analizę w celu ustalenia możliwie optymalnych rozmiarów bramek AND i ich liczby. Będziemy się tu kierować głównie chęcią stworzenia uniwersalnej struktury o możliwie małych wymiarach.

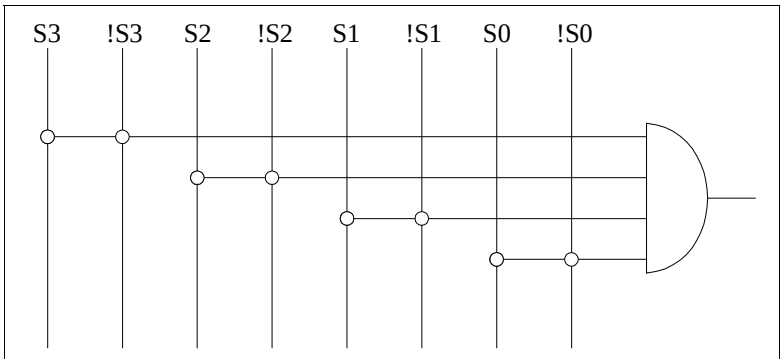


Rysunek 2 Rozwinięcie schematu matrycy PAL. Iloczyn "na drucie" faktycznie realizowany jest przez wielowejściowe bramki AND.

Przedstawiona na rysunku 1 wersja pozwala teoretycznie realizować ośmiowejściowe bramki AND – bramka taka realizowałaby funkcję:

$$OUT = S3 \cdot !S3 \cdot S2 \cdot !S2 \cdot S1 \cdot !S1 \cdot S0 \cdot !S0$$

Oczywiście wynikiem będzie zawsze logiczne zero. Żadna z bramek logicznie zaprojektowanego (zaprogramowanego) układu nie będzie realizowała równocześnie iloczynu sygnału i jego negacji. Wobec tego na pojedynczą bramkę AND możemy wprowadzić tylko cztery z pośród ośmiu sygnałów. Każde z wejść jest przypisane konkretnemu sygnałowi wejściowemu i może na nie być podany ten sygnał albo jego negacja.



Rysunek 3 Schemat proponowanej realizacji matrycy połączeniowej. Każde z wejść bramek jest przypisane konkretnemu sygnałowi wejściowemu i może być zwarte z jego prosta lub zanegowana reprezentacją.

Do rozpatrzenia pozostaje jeszcze problem liczby bramek. Czterobitowy sygnał wejściowy może przyjmować jeden z szesnastu możliwych stanów. Ponieważ realizacja układu w którym każdemu stanowi wejściowemu odpowiada jedynka na wejściu jest cokolwiek wątpliwa (taki układ można zrealizować znacznie skromniejszymi środkami niż układem PAL) toteż za maksymalną liczbę stanów w których na wyjściu panuje jedynka należałoby przyjąć piętnaście stanów. Ponieważ bramka AND odpowiada jednej czterobitowej kombinacji sygnału wejściowego wówczas należało by zastosować ich właśnie piętnaście. Spójrzmy na tablicę Karnaugha.

S3S2 \ S1S0	00	01	11	10
00	1	1	1	1
01	1	1	1	0
11	1	1	1	1
10	1	1	1	1

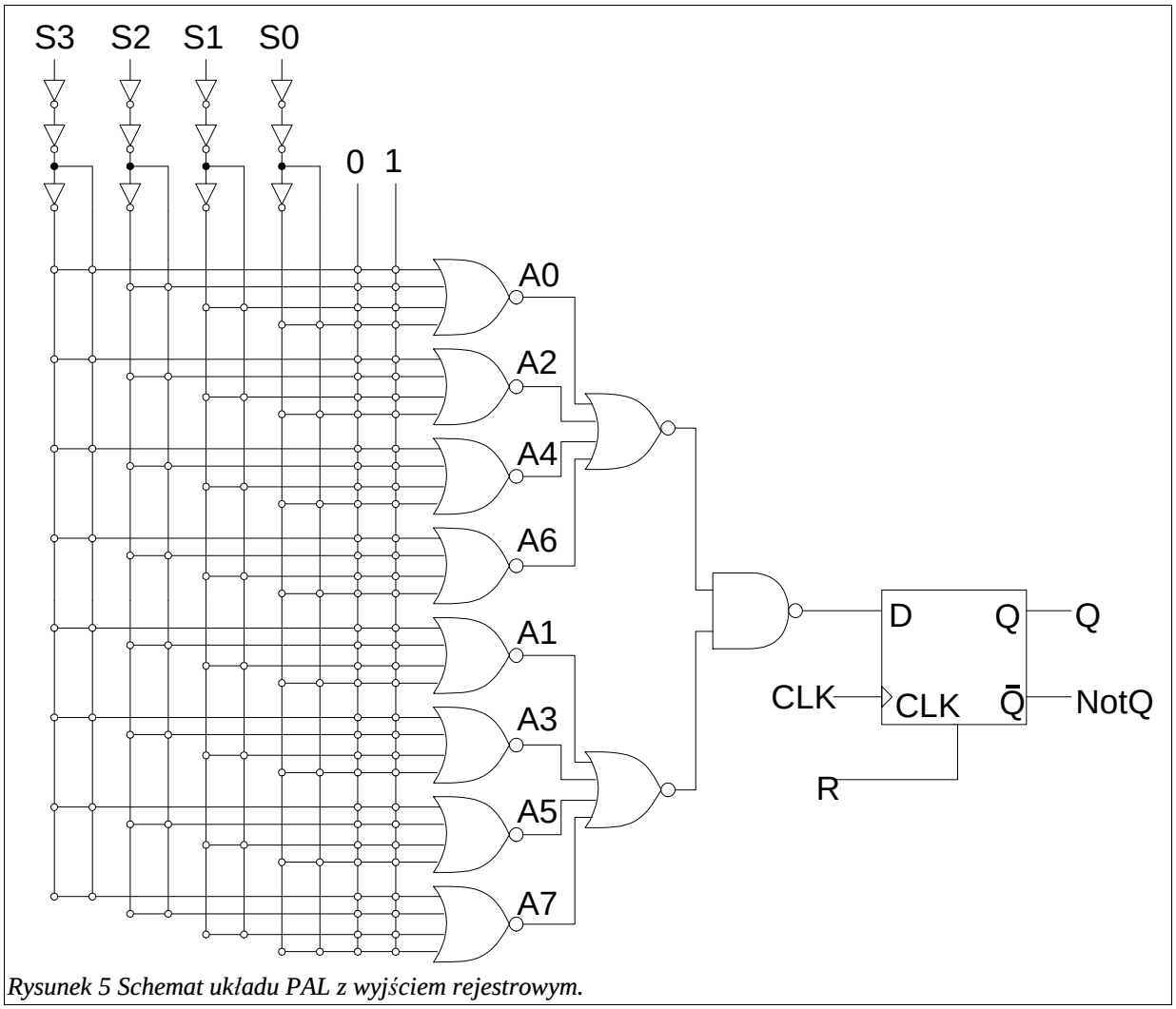
S3S2 \ S1S0	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	0	1	0	1
10	1	0	1	0

Rysunek 4 Dwa przypadki funkcji realizowanych z wykorzystaniem układu PAL.

Niezależnie od tego która kombinacja odpowiada zeru tak zrealizowany układ byłby wielce nie optymalny. Jak łatwo zauważyć do realizacji dowolnej takiej funkcji potrzebujemy w zasadzie po jednej bramce czterowejściowej, trójwejściowej i dwuwejściowej oraz ewentualnego inwertera. Wprawdzie to też nie jest realizacja optymalna ale generalnie wystarczą trzy bramki AND i ewentualny inwerter. Kluczem do dalszej minimalizacji układu okazuje się takie zaprojektowanie jego struktury aby było możliwe zastosowanie bramek AND o liczbie wejść mniejszej niż cztery! Ponieważ jak już wspomniano bramki fizycznie będą czterowejściowe toteż jedynym sposobem na osiągnięcie zamierzonego celu jest "eliminacja" zbędnych wejść poprzez podanie na nie stanu logicznej jedynki co oznacza zwarcie do szyny zasilającej. W ten sposób matryca połączeniowa rozbudowała się do dziewięciu sygnałów - czterech par związanych z sygnałami wejściowymi oraz linii V_{CC} niezbędnej do eliminacji nie wykorzystywanych wejść bramek.
W dalszym ciągu nie wiemy jednak ile powinno być bramek. Popatrzmy jeszcze raz na tablicę Karnaugha i zastanówmy się jeszcze raz nad najgorszym możliwym przypadkiem. Z najgorszym wykorzystaniem zasobów układu mamy do czynienia gdy pojedynczej bramce AND odpowiada tylko jedna kombinacja stanów (wykorzystujemy wszystkie cztery wejścia). Do takiego wykorzystania układu jesteśmy przymuszeni gdy kombinacje odpowiadające jedynce nie sąsiadują ze sobą (brak możliwości minimalizacji układu). Rozważania nasze prowadzą w efekcie do czterobitowego kodera parzystości (lub nieparzystości). Tablica Karnaugha odpowiadająca temu układowi jest przedstawiona na rysunku 4. Jak widać mamy maksymalnie osiem izolowanych od siebie stanów w których na wyjściu układu panuje logiczna jedynka. "Przesunięcie" dowolnej jedynki prowadzi do powstania grupy w obrębie której możemy już zastosować minimalizację i redukując liczbę stosowanych bramek do siedmiu (czterech czterowejściowych) i trzech trójwejściowych. Dodanie do układu dziewiątej jedynki również prowadzi do utworzenia grupy pozwalającej na minimalizacje układu, tym razem do czterech bramek czterowejściowych oraz czterech trójwejściowych. Z powyższego rozważania płyną dwa ważne wnioski: maksymalna niezbędna liczba bramek AND wynosi osiem oraz że w pewnych przypadkach może się okazać konieczna eliminacja pewnych bramek z układu. Aby dana bramka (AND) nie oddziaływała na układ na jej wyjściu musi panować logiczne zero co najłatwiej osiągnąć podając na wejście również logiczne zero czyli zwierając wejście do szyny masy. W efekcie otrzymujemy końcową postać matrycy połączeniowej: cztery pary sygnałowe, jedna linia Vcc do eliminacji zbędnych wejść oraz linia masy do eliminacji zbędnych bramek. Razem dziesięć linii.

Powyższe rozważania doprowadziły nas do sformułowania obrazu układu który wprawdzie jest możliwy do zrealizowania ale być może nie jest jeszcze najlepszy. Przypomnijmy, układ składa się obecnie z ośmiu czterowejściowych bramek AND które sterują ośmiowejściową bramką OR. Bramek tych nie da się zrealizować bezpośrednio w układzie CMOS gdzie podstawowymi elementami (poza inwerterem) są bramki NAND oraz NOR. Bramkę AND mogliśmy zrealizować stosując bramkę NOR i zamieniając funkcję linii sygnałowych na matrycy połączeniowej (sygnał zanegowany dla bramki AND staje się sygnałem prostym w przypadku zastosowania w jej miejsce

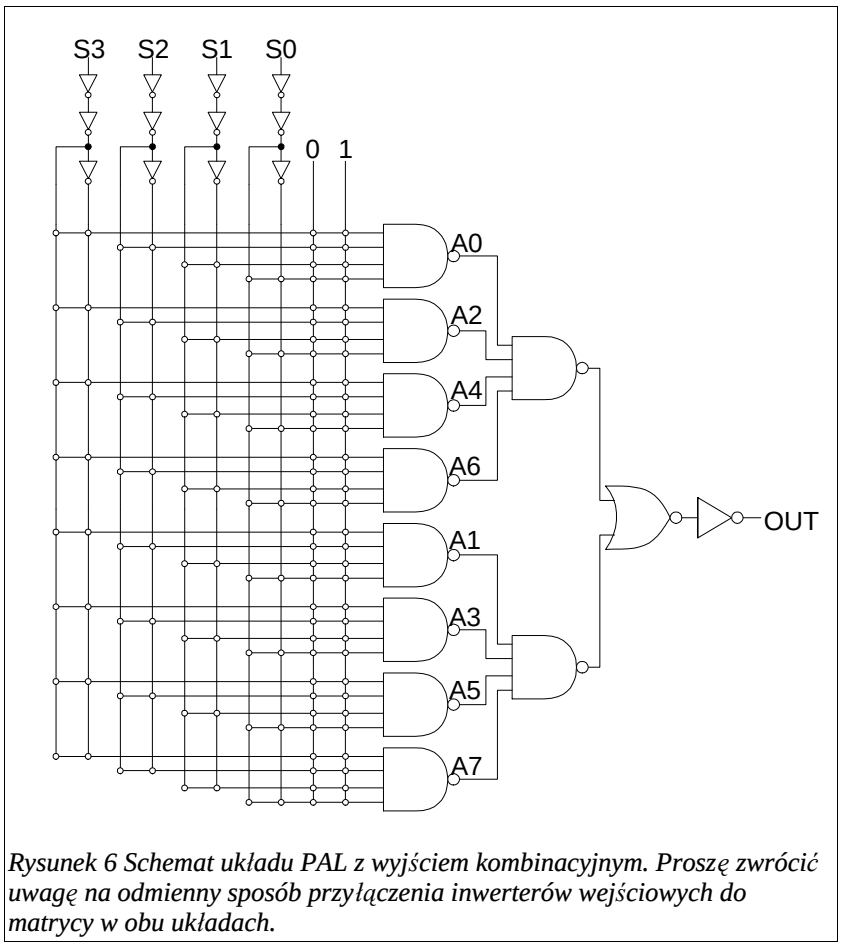
bramki NOR). Pozostaje bramka OR dla której w tej realizacji nie mielibyśmy już żadnej alternatywy. Ponieważ jednak przyjeliśmy że nie będziemy stosować bramek o liczbie wejść większej od czterech toteż bramkę OR musimy zrealizować z kaskady trzech bramek - dwóch czterowejściowych oraz jednej dwuwejściowej. Wówczas bramkę OR zrealizujemy stosując czterowejściowe bramki NOR których sygnał wymnożymy w bramce NAND! Taka realizacja jest odpowiednia dla układu sekwencyjnego. Wówczas dostępny sygnał możemy podać wprost na przerzutnik. Nieco inaczej sytuacja wygląda w przypadku układu kombinacyjnego. Bramka NANA sterowała by wówczas bezpośrednio wyjściem układu. Wydaje się że lepiej w tej roli sprawowałby się bufor zapewniając symetryczne charakterystyki wyjściowe. Jeszcze lepszym rozwiązaniem byłby inwerter (zajmowałby mniejszą powierzchnię). Okazuje się że dla układu kombinacyjnego lepsza byłaby logika z wyjściem zanegowanym. Zadaną funkcję dałoby się zrealizować stosując kaskadę czterowejściowych bramek NAND oraz pojedynczej ośmiowejściowej bramki NAND!



Rysunek 5 Schemat układu PAL z wyjściem rejestrowym.

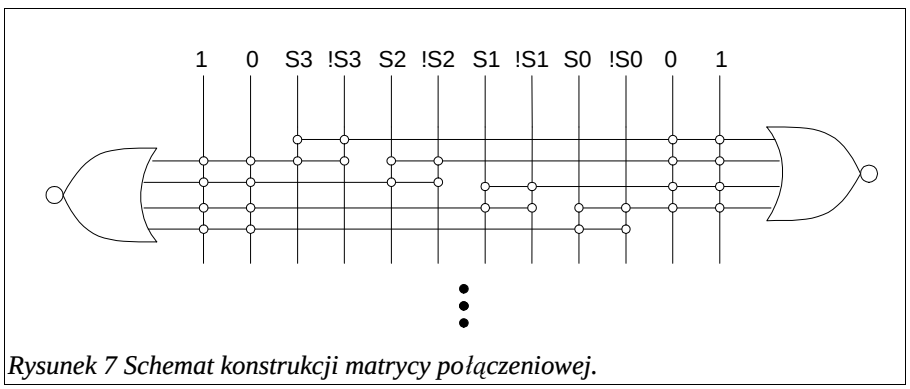
Ostatecznie mamy dwa schematy realizowanych układów -jeden z wyjściem kombinacyjnym oraz drugi z wyjściem rejestrowym. W obu przypadkach największy obszar zajmie matryca połączeniowa. Proste przeniesienie układu ze schematu na krzem jest zasadniczo możliwe ale prowadzi do dużego marnotrawstwa powierzchni. Popatrzmy na plan matrycy. O wielkości matrycy decyduje obszar niezbędny do zrealizowania połączeń do wejść każdej z bramek. Wymiar pionowy tego obszaru jest limitowany przez odległości pomiędzy poszczególnymi kanałami tranzystorów budujących daną bramkę czyli w zasadzie przez rozmiar samej bramki. Wymiar poziomy został określony przez szerokość ścieżek sygnałowych oraz narzucony przez technologię minimalny odstęp. Przypisany każdej z bramek obszar matrycy wykorzystany jest tylko w stopniu nieco większym niż 50%. Istnieje prosta metoda zmniejszenia powierzchni matrycy. Należy bramki

umieścić po jej obu stronach. Konieczne jest przesunięcie obu kolumn bramek o taką wielkość aby możliwe było bezkolizyjne poprowadzenie połączeń oraz powiększenie ilości sygnałów na matrycy o dalsze dwa - jeszcze jedną linię V_{CC} i GND gdyż dotychczasowe mogły być wykorzystane tylko przez jedna kolumnę bramek. Otrzymujemy redukcję powierzchni około 40%.



Rysunek 6 Schemat układu PAL z wyjściem kombinacyjnym. Proszę zwrócić uwagę na odmienny sposób przyłączenia inwerterów wejściowych do matrycy w obu układach.

Obie wersje układów mają taki sam rozkład logiczny sygnałów pomimo stosowania w obrębie matrycy odmiennej logiki (w układzie rejestrowym zero na matrycy oznacza stan wysoki, natomiast w układzie kombinacyjnym zeru odpowiada stan niski). Z punktu widzenia wejść i wyjść oba układy są równoważne i posługują się logiką dodatnią - jedynce logicznej odpowiada podanie na wejście napięcia zasilania lub też obecność na wyjściu sygnału o takim napięciu. Sterowanie matrycami w obu układach zostało tak zrealizowane aby poszczególne punkty matryc w obu układach sobie odpowiadały. Tak więc w obu układach realizacja danej funkcji logicznej prowadzi do takiego samego schematu połączeń!



Rysunek 7 Schemat konstrukcji matrycy połączeniowej.

Programowanie układu.

Aby zaprogramować układ należy stworzyć mapę połączeń. Punktem wyjścia jest zapisanie tablicy

Karnaugha dzięki której szybko można otrzymać równanie opisujące funkcję. Równanie powinno być zapisane w postaci sumy iloczynów (postać ta jest naturalną dla naszego układu). Tworząc równanie możemy równocześnie minimalizować funkcję. Należy pamiętać że ze względu na konstrukcję fizyczną układu równanie może zawierać do ośmiu iloczynów! Otrzymanie bardziej złożonego równania oznacza że jest ono nie optymalne - układ da się jeszcze zminimalizować. Następnie porządkujemy równanie - grupujemy iloczyny pod względem ilości występujących w nich sygnałów i przypisując je poszczególnym bramką (służy do tego tabelka nr 2). Tabelka posłuży do wypełnienia mapy połączeń odwzorowującej budowę matrycy połączeniowej. Każda biała kratka odpowiada na matrycy punktowi w którym możemy umieścić przelotkę tworząc tym samym połączenie. Każde z wejść bramki możemy połączyć z jednym i tylko jednym z pośród czterech sygnałów. Są to: 0, 1, Sx lub !Sx.
Sx to sygnał z wejścia układu, x oznacza numer wejścia (od 3 do 0). Sygnał !Sx to zanegowany sygnał Sx. Każde z czterech wejść bramki przypisane jest konkretnemu wejściu układu (IN21 - wejście pierwsze bramki numer 2 przypisane jest wejściu numer 1 układu, czyli można na nie podać następujące sygnały: S1, !S1, 0 lub 1. Na wejście IN53 można natomiast podać sygnały: S3, !S3, 0 lub 1). Połączenie danego wejścia z 1 pozwala na eliminacje zbędnych wejść danej bramki np: realizując iloczyn S1*!S3 musimy dokonać połączenia INy0 - 1, INy1 - S1, INy2 - 1, INy3 - !S3 (y to numer wykorzystanej bramki). Podłączenie na wszystkie wejścia bramki jedynki powoduje podanie 1 logicznej na bramkę OR i w efekcie uniemożliwi pracę układu (na wyjściu zawsze będzie panował stan 1 logicznej. Możliwość podania na bramkę stanu 0 wykorzystujemy do jej eliminacji. Generalnie możliwość ta służy eliminacji bramek zbędnych. Wówczas wszystkie wejścia łączymy z szyną 0 (generalnie wystarczy połączyć tylko jedno z wejść, wszak mamy do czynienia z bramką AND).
Przeanalizujmy to na przykładzie.Przykładową funkcję mamy opisaną za pomocą tabelki 1. Dla dziewięciu kombinacji występuje jedynka. Oznacza to że do zrealizowania funkcji niezbędne będzie chociaż minimalne jej zminimalizowanie. Popatrzmy na tablice Karnaugha:

S3S2 \ S1S0	00	01	11	10
00	1	0	1	1
01	0	1	0	0
11	1	1	0	0
10	1	1	0	1

Lp.	S3S2S1S0	OUT
0	0000	1
1	0001	0
2	0010	1
3	0011	1
4	0100	0
5	0101	1
6	0110	0
7	0111	0
8	1000	1
9	1001	1
10	1010	1
11	1011	0
12	1100	1
13	1101	1
14	1110	0
15	1111	0

Niech funkcja po minimalizacji wygląda następująco:

S3S2 \ S1S0	00	01	11	10
00	1	0	1	1
01	0	1	0	0
11	1	1	0	0
10	1	1	0	1

W drugiej tablicy zaznaczono pogrupowane kombinacje. Nie jest to najlepsza minimalizacja ale chodzi tu tylko o przykład. Utworzone na podstawie powyższej tablicy równanie wygląda następująco:

$$OUT = !S_3 \cdot !S_2 \cdot !S_1 \cdot !S_0 + S_1 \cdot !S_3 \cdot !S_2 + !S_3 \cdot !S_1 \cdot S_2 \cdot S_0 + !S_1 \cdot S_3 + S_1 \cdot S_3 \cdot !S_2 \cdot !S_0$$

Mamy pięć iloczynów. Porządkujemy je przypisując je kolejnym bramką AND:

$AND0 = !S_3 \cdot !S_2 \cdot !S_1 \cdot !S_0$

$AND1 = !S_3 \cdot S_2 \cdot !S_1 \cdot S_0$

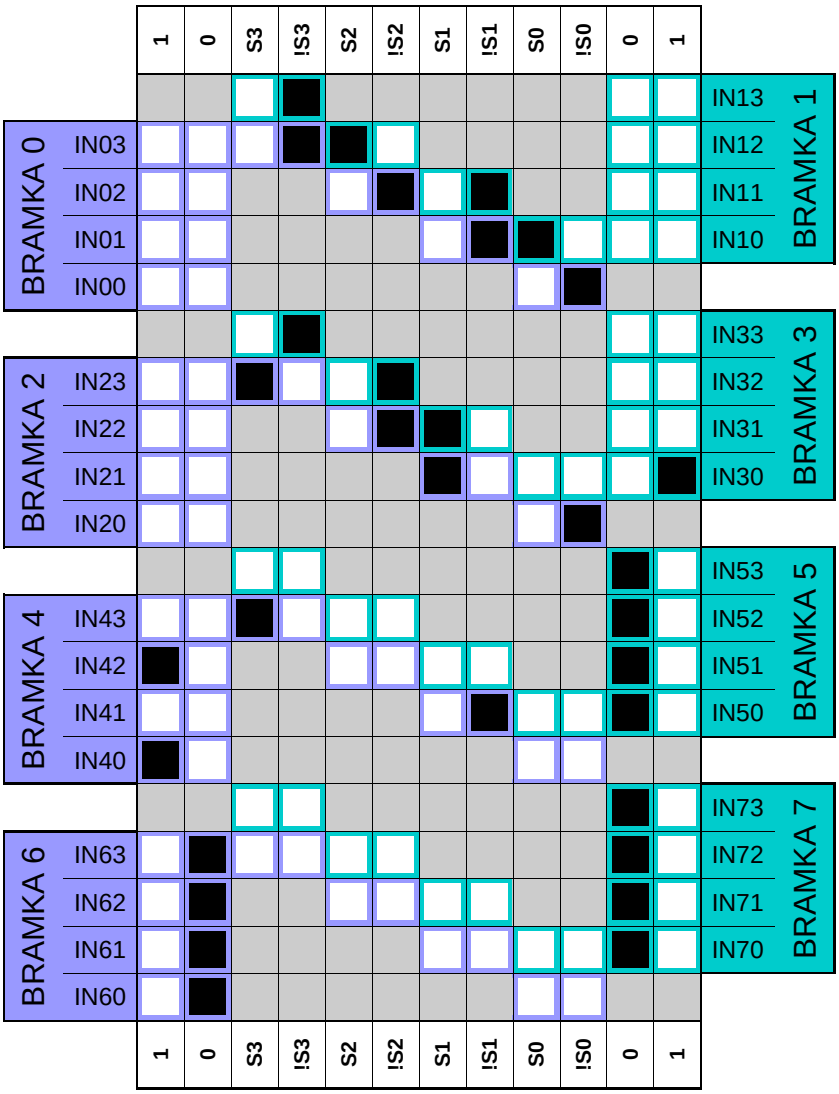
$AND2 = S_3 \cdot !S_2 \cdot S_1 \cdot !S_0$

$AND3 = !S_3 \cdot !S_2 \cdot S_1$ - *S0* nieaktywne

$AND4 = S_3 \cdot !S_1$ - *S2* oraz *S0* nieaktywne

Bramki AND5, AND6 oraz AND7 nieaktywne.

Przypomnijmy że niewykorzystane wejścia aktywnych bramek (np. S0 bramki AND3) podłączamy do szyny logicznej jedynki, natomiast wejścia niewykorzystanych bramek (AND5, AND6 i AND7) podłączamy do szyny logicznego zera. Poniżej zamieszczono mapę połączeń wypełnioną na podstawie powyższych równań:



Należy zwrócić uwagę na numerację bramek w matryce. Zasadniczo jest ona umowna i ma przede wszystkim na celu ułatwienie procesu programowania.
Po "zaprogramowaniu" układu i jego ekstrakcji należy przeprowadzić symulację układu. W wygenerowanej przez program ekstraktujący netliście należy odszukać numery węzłów przypisane sygnałom:

-wejściowym S0...S3
-sterującym RST oraz CLK (w przypadku układu sekwencyjnego)
-wyjściowym Q i notQ (układ sekwencyjny) lub OUT (układ kombinacyjny)

Pomocniczo można monitorować sygnały A0...A7. Są to sygnały wyjściowe bramek "AND".

PROGRAMATOR :-)

1. Tabela

S3S2 \ S1S0	00	01	11	10
00				
01				
11				
10				

2. Równanie

3. Mapa połączeń