Programowalna matryca logiczna

1. Wprowadzenie

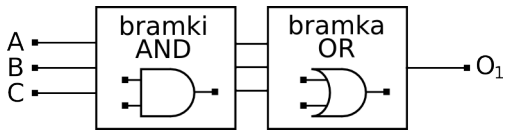
We współczesnej elektronice cyfrowej obecne są dwa trendy rozwoju [1]:

- *Specjalizowane układy scalone* ASIC (ang. *Application Specific Integrated Circuits*)– są to układy przeznaczone do określonych, ale zwykle bardzo wąskich zastosowań. Układy scalone tego typu projektowane są w celu wyprodukowania pewnej ilości egzemplarzy, których modyfikacja nie jest po produkcji możliwa. Zaletą układów ASIC jest niski koszt jednostkowy w przypadku produkowania dużych serii układów scalonych. Dodatkowo układy te są zoptymalizowane pod względem zajmowanej powierzchni krzemu i szybkości. Liczba elementów logicznych ograniczona jest do niezbędnego minimum, ścieżki łączące tranzystory są zoptymalizowane pod kątem minimalizacji poboru mocy.
- Programowalne układy logiczne PLD (ang. *Programmable Logic Devices*) – są to układy uniwersalne, które mogą zostać wykorzystane w bardzo szerokim spektrum aplikacji. Zbudowane są z programowalnych bloków elementów logicznych (kombinacyjnych i sekwencyjnych) oraz konfigurowalnych ścieżek umożliwiających łączenie bloków logiki. Funkcjonalność tych układów określana jest przez projektanta na drodze programowania połączeń pomiędzy blokami elementów logicznych. Do zapamiętania swojej konfiguracji układy PLD używają pamięci typu SRAM (ang. *Static Access Random Memory*), EEPROM (ang. *Electrically-Erasable Programmable Read-Only Memory*) lub układów bezpieczników (układy antifuse-FPGA) konfigurowanych jednorazowo za pomocą przepalania bezpieczników łączących komórki elementów logicznych oraz ścieżki. W porównaniu do układów ASIC są wolniejsze i zużywają więcej mocy.

Współcześnie dostępne technologie stosowane do produkcji układów elektroniki cyfrowej typu ASIC umożliwiają budowanie układów taktowanych zegarami o częstotliwościach rzędu kilku gigaherców. Mimo tego, wiele aplikacji przemysłowych nie wymaga taktowania układów tak szybkimi zegarami i układy typu PLD w zupełności zaspokajają zapotrzebowanie inżynierów.
Proces projektowania układów elektronicznych opartych na układach PLD umożliwia wielokrotne reprogramowanie układów połączeń – jest to główna przewaga układów PLD nad układami ASIC. Dodatkowo układy reprogramowalne produkowane w dużych seriach cenowo zbliżają się do układów ASIC. Właśnie dlatego układy PLD cieszą się dużą popularnością w praktyce inżynierskiej.

1.1.Budowa wewnętrzna układów konfigurowalnych

Najprostszą i najczęściej stosowaną strukturą PLD jest struktura złożona z matrycy bramek AND i bramki OR. przedstawiona na rys. 1.1.

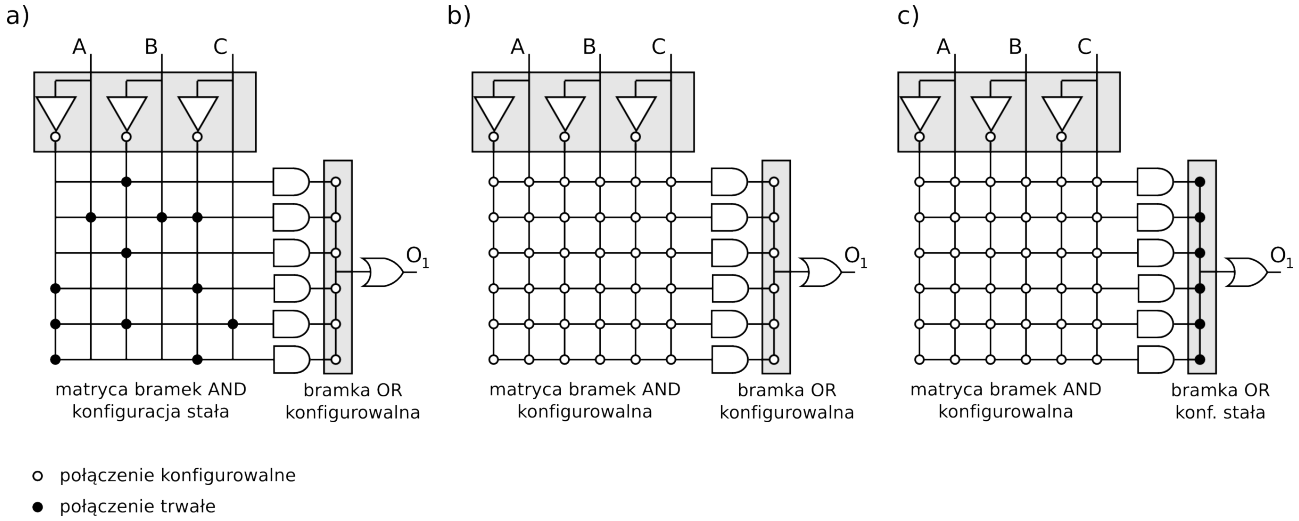


Rys. 1.1. Schemat blokowy programowalnej matrycy logicznej.

Wyróżniamy trzy typy matryc programowalnych PLD, przedstawione na rysunku 1.2:

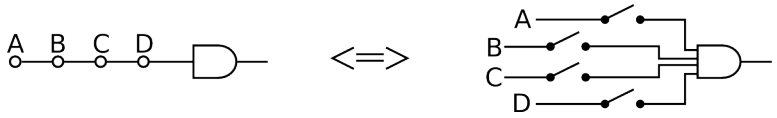
- ROM – w tym typie matrycy możliwa jest tylko konfiguracja wejść bramki OR.
- PLA (ang. *Programmable Logic Array*), w której konfigurowalna jest zarówno matryca bramek AND jak również bramek OR.
- PAL (ang. *Programmable Array Logic*), w której użytkownik ma wpływ tylko na połączenia w matrycy bramek AND.

Przykład realizacji układu przedstawionego na rys. 1.1. w architekturze ROM, PLA i PAL przedstawia rysunek 1.2.



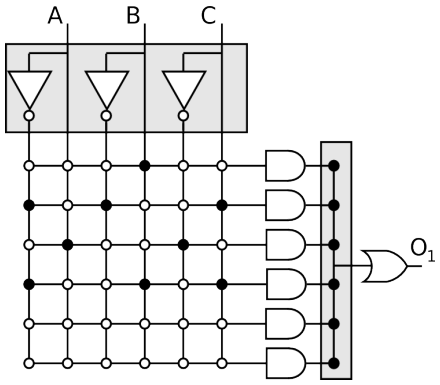
Rys. 1.2. Realizacja praktyczna matrycy bramek PLD, a) architektura ROM, b) architektura PLA, c) architektura PAL

Układ inwerterów połączony z wejściami umożliwia uwzględnienie negacji sygnałów wejściowych w programowanych funkcjach logicznych. Kilka połączeń konfigurowalnych (oznaczonych na rys. 1.2 jako puste kropki), połączonych z pojedynczą bramką AND lub OR oznacza, że dana bramka posiada kilka konfigurowalnych wejść. Jedno połączenie konfigurowalne na schemacie oznacza jedno (możliwe do wyboru przez projektanta) wejście bramki AND lub OR (rys. 1.3).



Rys. 1.3. Reprezentacja symboliczna wielowejściowej bramki konfigurowalnej oraz schemat koncepcyjny.

W ćwiczeniu wykorzystywać będziemy tylko matrycę typu PAL, dlatego w dalszej części instrukcji nie będziemy się zajmować pozostałymi architekturami. Przykładowa matryca PAL posiada jedną sześciowejściową bramkę OR, jednak liczba wejść, jak również liczba bramek OR może być inna. Dokonując odpowiednich połączeń w macierzy bramek AND, programujemy dowolną funkcję będącą sumą iloczynów sygnałów wejściowych A, B C. Przykład połączeń w macierzy bramek przedstawia rys. 1.4.



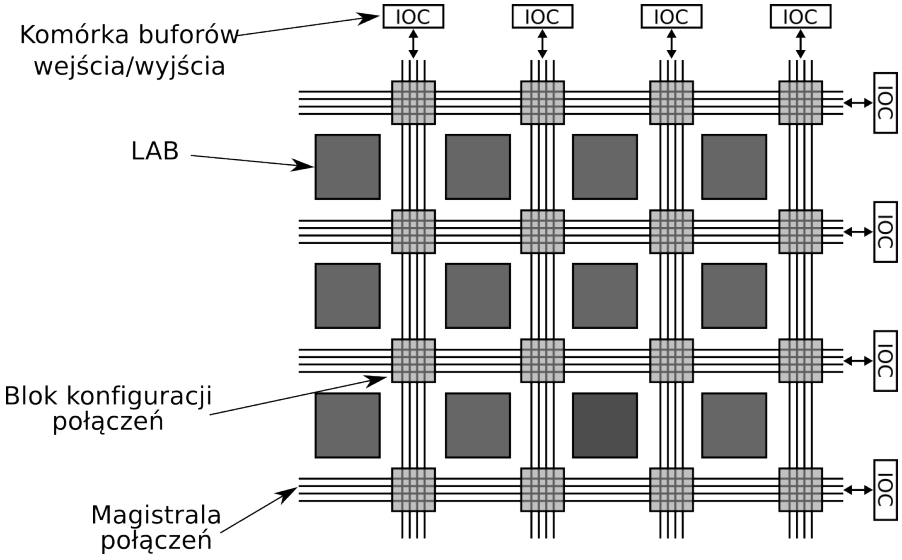
Rys. 1.4. Przykład połączeń matrycy bramek PAL.

W podanym przypadku funkcja logiczna na wyjściu bramki OR opisana jest wyrażeniem:

$$O_1 = B + \bar{A}\,\bar{B}\,C + A\,\bar{C} + \bar{A}\,B\,C$$

1.2 Zastosowania praktyczne programowalnych układów logicznych

Omówiona powyżej struktura PAL jest przykładem bloku funkcjonalnego, który wykorzystywany jest w bardziej złożonych układach programowalnych, takich jak układy FPGA (ang. *Field Programmable Gate Array*). Układy FPGA są współcześnie stosowane w procesie projektowania praktycznie we wszystkich gałęziach elektroniki – właśnie dzięki możliwości wielokrotnego programowania. Przykład struktury układu FPGA przedstawia rys. 1.5.

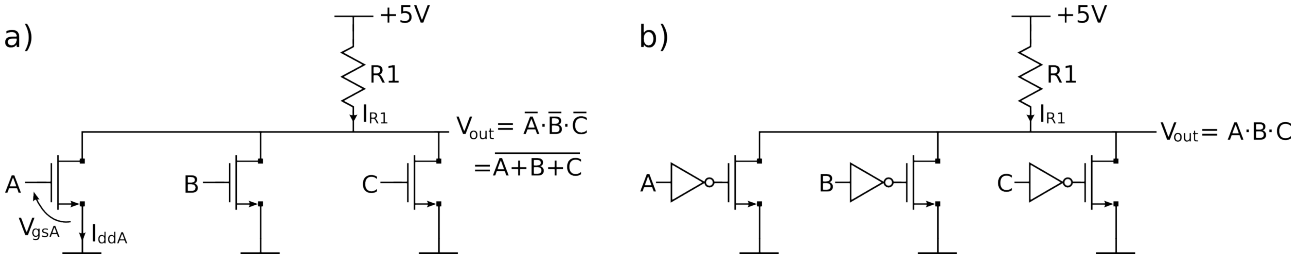


Rys. 1.5. Struktura wewnętrzna układu FPGA

Strukturę układu tworzy macierz niezależnych bloków logicznych LAB (ang. *Logic Array Block*). Każdy blok logiczny zbudowany jest z pewnej liczby elementów logicznych, każdy element może realizować niezależną funkcję logiczną. Połączenia elektryczne pomiędzy blokami są realizowane za pomocą magistrali połączeń, których topologia jest konfigurowana w blokach konfiguracji połączeń. Magistrale biegną wzdłuż całego układu, dzięki czemu łączone mogą być ze sobą bloki w dowolnych lokalizacjach. Układy FPGA wyposażone są także w konfigurowalne bufory wejścia/wyjścia (IOC) zwykle zawierające bramki trójstanowe. Umożliwia to używanie linii sygnałowych zewnętrznych w zależności od potrzeb jako linii wejściowych, wyjściowych lub dwukierunkowych.

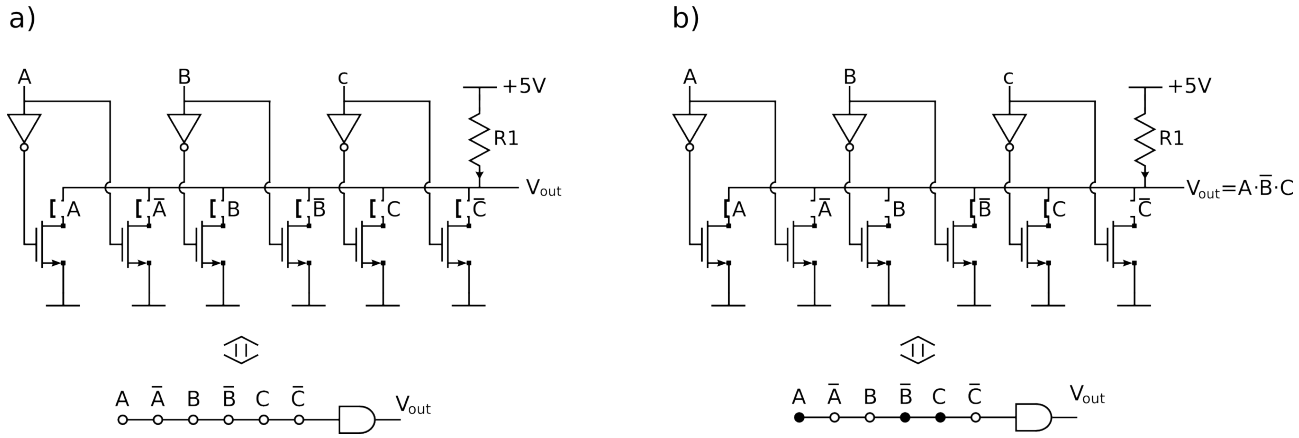
1.3. Iloczyn galwaniczny

Dla zrozumienia działania programowalnej zworkowo matrycy logicznej konieczne jest wprowadzenie pojęcia iloczynu galwanicznego (tzw. „bramka na drucie"). Przedstawione architektury cyfrowych układów konfigurowalnych zawierają w swojej strukturze bramki AND oraz bramki OR. Realizacja układowa wielowejściowej bramki wymaga zbudowania matrycy kluczy połączonych z wejściami bramki logicznej. Rozwiązanie takie wymaga zastosowania relatywnie dużej liczby tranzystorów oraz linii transmisyjnych, dlatego dla uproszczenia budowy matrycy wykorzystywanej w ćwiczeniu zastosowano bramki w postaci iloczynu galwanicznego.
Przykład realizacji bramek w postaci iloczynu galwanicznego przedstawia rysunek 1.6.



Rys. 1.6. Trzywejściowa bramka logiczna w architekturze iloczynu galwanicznego, a) bramka NOR, b) bramka AND

Wejścia logiczne stanowią bramki tranzystorów, połączonych równolegle, pracujących w układzie wspólnego źródła. W tej konfiguracji można traktować tranzystor NMOS jako sterowany klucz. Źródła tranzystorów połączone są z masą, dreny połączone są z rezystorem podciągającym R1 (tzw. rezystor pull-up). Jeżeli potencjał na bramkach (wejściach) tranzystorów wynosi ~0V, tranzystory pracują jako klucze rozwarte i prąd drenu nie płynie ($I_{ddA} = 0$), potencjał na wyjściu ustala się blisko górnego zasilania (logiczna „1") ze względu na obecność rezystora R1. Jeżeli bramka przynajmniej jednego tranzystora zostanie spolaryzowana napięciem +5V, tranzystor ten zaczyna pracować jak klucz zwarty i potencjał na wyjściu bramki ustali się blisko 0V (logiczne „0"). W ten sposób, jeżeli chociaż na jednym z wejść ustali się logiczna „1", na wyjściu układu otrzymamy logiczne „0". Układ ten działa zatem jak 3-wejściowa bramka NOR.
Jeżeli zanegujemy wejścia bramki NOR, otrzymamy bramkę AND (rys. 1.6.b.). Jeśli uzupełnimy bramkę AND w dodatkowe tranzystory oraz zworki, uzyskamy układ programowalnej bramki AND (rys. 1.7.a). Bramka taka pozwala na zaprogramowanie wybranego iloczynu z sygnałów $A, \bar{A}, B, \bar{B}, C, \bar{C}$. Przykładowo, aby zaprogramować iloczyn $A\bar{B}C$ należy połączyć zworki w sposób pokazany na rysunku 1.7.b.



Rys. 1.7. a) Programowalna bramka AND (zworki rozwarte) i jej zapis symboliczny, b) zaprogramowana zworkami funkcja logiczna $A\bar{B}C$ oraz zapis symboliczny przykładowej konfiguracji.

1.4 Przykład realizacji funkcji logicznej – logika kombinacyjna

Używając metody tablic Karnaugha, można pokazać realizację dowolnej funkcji logicznej zaimplementowanej w opisanej powyżej strukturze PAL. Rozważmy przykład 3-bitowego konwertera z kodu binarnego na kod Graya. Tabela 1. przedstawia rozważane kody wraz z reprezentacją dziesiętną.

Tabela 1. Liczby od 0 do 7 w kodzie binarnym i kodzie Gray'a

Liczba dziesiętna	Kod binarny C B A	Kod Gray'a $o_3\ o_2\ o_1$
0	0 0 0	0 0 0
1	0 0 1	0 0 1
2	0 1 0	0 1 1
3	0 1 1	0 1 0
4	1 0 0	1 1 0
5	1 0 1	1 1 1
6	1 1 0	1 0 1
7	1 1 1	1 0 0

Obliczmy funkcję logiczną dla najmłodszego bitu w kodzie Graya, czyli $\mathbf{o_1}$. Sporządzamy tablicę Karnaugha, rozdzielając zmienne wejściowe na grupy C oraz BA (rysunek 1.8).

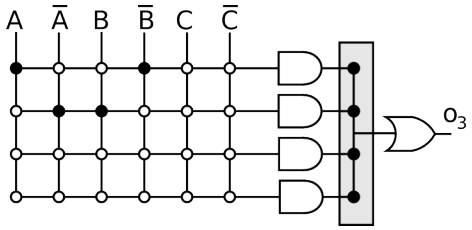
C \ BA	00	10	11	01
0	0	1	0	1
1	0	1	0	1

Rys. 1.8. Tablica Karnaugha dla najmłodszego bitu w 3-bitowym kodzie Graya.

Grupując wartości logiczne 1 w dwie grupy, możemy zapisać funkcję logiczną:

$$o_1 = A\,\bar{B} + \bar{A}\,B$$

Następnym krokiem jest wykonanie połączeń w matrycy bramki AND; połączenia te odpowiadają odpowiednio iloczynom $A\,\bar{B}$ oraz $\bar{A}\,B$, bramka OR odpowiada sumie logicznej w wyrażeniu na najmłodszy bit kodu Graya **o1** (rysunek 1.9).

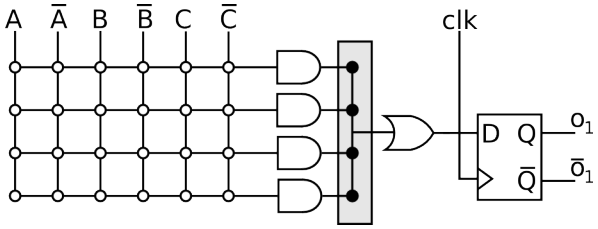


Rys. 1.9. Konfiguracja matrycy PLD dla funkcji logicznej najmłodszego bitu kodu Graya. Architektura PAL, wejścia bramki OR zwarte na stałe.

Sporządzając analogiczne tablice prawdy dla pozostałych bitów **o2** i **o3**, przy użyciu trzech matryc PLD otrzymalibyśmy pełny, 3-bitowy dekoder z kodu BCD na kod Graya.

1.5 Matryca PLD z rejestrem wyjściowym

Dla celów programowania układów logiki sekwencyjnej, struktury PLD mogą być wyposażone w rejestr (przerzutnik) na wyjściu. Przykład układu PLD z przerzutnikiem typu D przedstawia rysunek 1.10.



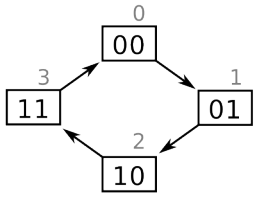
Rys. 1.10. Struktura PLD w architekturze PAL z rejestrem wyjściowym

Zastosowanie rejestru wyjściowego pozwala zapamiętać stan logiczny wyjścia matrycy bramek AND i OR. Stan logiczny zapamiętany w poprzednim cyklu zegara może również służyć jako sygnał wejściowy dla innego układu logiki kombinacyjnej lub sekwencyjnej.

1.6 Przykład realizacji funkcji logicznej – logika sekwencyjna

Przykładem zastosowania układu PLD z rejestrem wyjściowym jest licznik 2-bitowy w kodzie BCD. Licznik 2-bitowy możemy rozważać jako maszynę stanów o 4 stanach (0,1,2,3 na rysunku 1.11.a). Sporządzamy tabelę przejść przedstawiającą stan następny w funkcji stanu poprzedniego (rysunek 1.11.b).

a)



b)

Stan N o_2o_1	Stan N+1 o_2o_1
0 0	0 1
0 1	1 0
1 0	1 1
1 1	0 0

Rys. 1.11. a) Diagram przejść, b) tabela stanów licznika 2-bitowego

Używając metody tablic Karnaugha, wyznaczamy funkcje logiczne stanu następnego (N+1) w funkcji stanu poprzedniego dla wyjść $\mathbf{o_1}$ i $\mathbf{o_2}$ (rysunek 1.12).

o_2^{N+1}

o_1 \ o_2	0	1
0	0	(1)
1	(1)	0

$o_2^{N+1} = o_1\bar{o}_2 + \bar{o}_1 o_2$

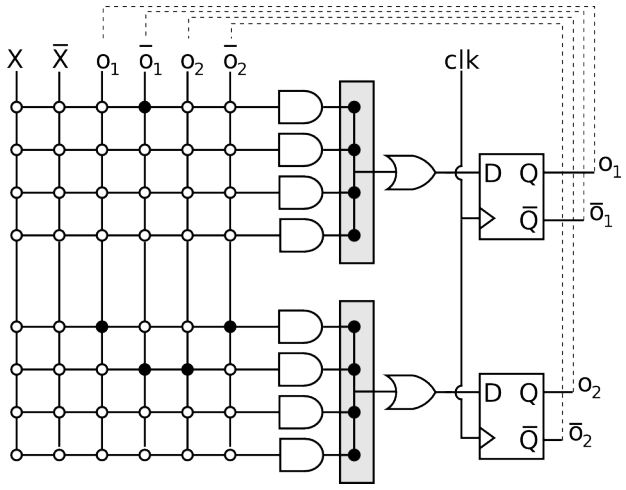
o_1^{N+1}

o_1 \ o_2	0	1
0	1	1
1	0	0

$o_1^{N+1} = \bar{o}_1$

Rys. 1.12. Tablice Karnaugha i odpowiadające im funkcje logiczne

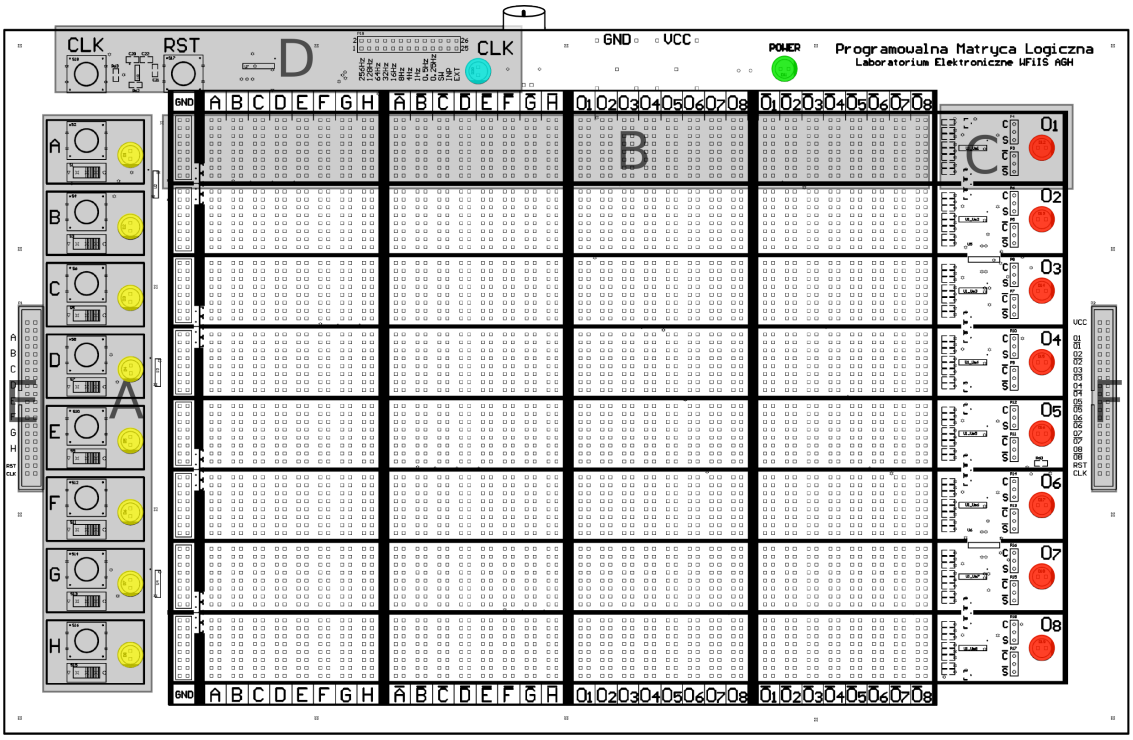
Jeżeli połączymy wyjścia przerzutników z wejściami matryc logicznych, możemy zaimplementować funkcje logiczne do układu sekwencyjnego złożonego z dwóch matryc PLD z rejestrami wyjściowymi (rysunek 1.13):



Rys. 1.13. Implementacja licznika 2-bitowego z użyciem matryc PLD.

2. Budowa programowalnej matrycy logicznej wykorzystywanej w ćwiczeniu

Matryca logiczna wykorzystywana w ćwiczeniu pozwala na programowanie układów logiki kombinacyjnej oraz sekwencyjnej za pomocą odpowiedniej kombinacji zworek. Budowa matrycy logicznej przedstawiona została na rysunku 2.1.



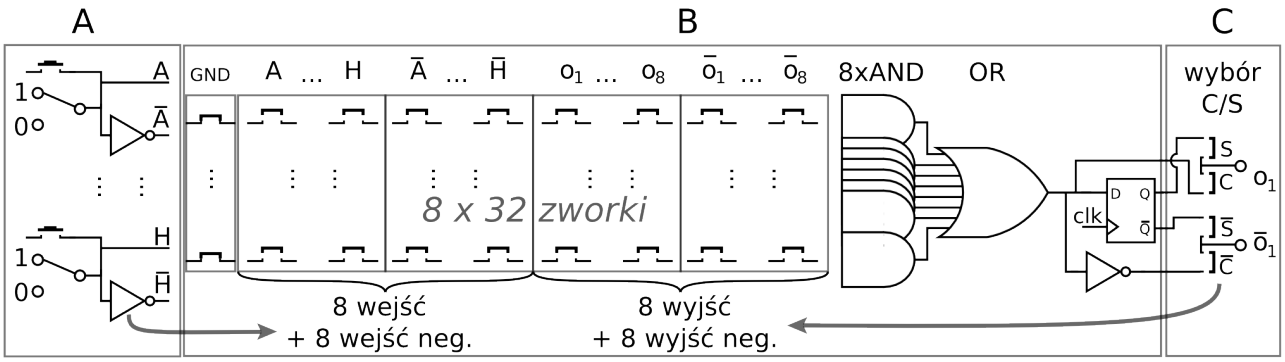
A – Grupa przełączników do ustawiania sygnałów wejściowych
B – Programowalny blok logiczny
C – Zworki wyboru wejścia kombinacyjnego (C) lub sekwencyjnego (S) wraz z diodami sygnalizującymi stan wyjścia danego bloku logicznego
D – Blok wyboru źródła sygnału zegarowego
E – Złącze doprowadzające sygnały wejściowe
F – Złącze wyprowadzające sygnały wyjściowe

Rys. 2.1. Budowa programowalnej matrycy logicznej

Sygnały wejściowe podawane są za pomocą przycisków lub przełączników (blok **A**, rys. 2.1). Osiem sygnałów wejściowych oznaczonych zostało kolejnymi literami alfabetu od „A"do „H". Stan logiczny wejść sygnalizowany jest diodami luminescencyjnymi koloru żółtego, umiejscowionymi w bezpośrednim sąsiedztwie przełączników.

Głównym elementem funkcjonalnymi matrycy są programowalne bloki logiczne. Matryca posiada 8 takich bloków, pierwszy blok został zaznaczony na rys. 2.1 jako **B**. Pojedynczy blok składa się z ośmiu 32-wejściowych bramek AND, 8-wejściowej bramki OR oraz przerzutnika typu D. Bramki AND zrealizowane zostały jako iloczyn galwaniczny (patrz punkt 1.2 instrukcji). Możliwe jest zbudowanie bramek AND czułych na sygnały wejściowe $A-H$, negacje sygnałów wejściowych $\bar{A}-\bar{H}$, wyjścia bloków logicznych $\boldsymbol{o_1}-\boldsymbol{o_8}$ oraz negacje wyjść bloków logicznych $\boldsymbol{\bar{o}_1}-\boldsymbol{\bar{o}_8}$. Bramka OR to układ scalony CD4078. Schemat programowalnego bloku logicznego (**B**) oraz zworek wyboru wyjścia sekwencyjnego lub kombinacyjnego (**C**) przedstawione zostały na rysunku 2.2.

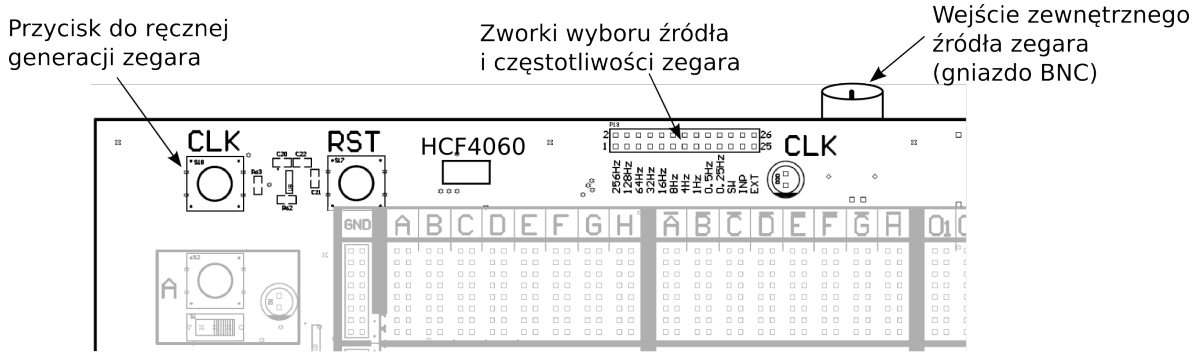
Pojedynczy rząd pinów (poziomo) stanowi jedną bramkę AND. Sygnały, na które ma być czuła pojedyncza bramka AND wybiera się poprzez wpięcie zworki w odpowiednio oznaczonej kolumnie. **Jeśli w danym bloku logicznym któraś bramka AND ma pozostać wyłączona, należy pamiętać, że wpięta powinna być w tym rzędzie zworka w kolumnie oznaczonej GND – spowoduje to wyłączenie bramki AND.**



Rys. 2.2. Blok przełączników sygnałów wejściowych (A), programowalny blok logiczny (B), zworki wyboru wyjścia kombinacyjnego lub sekwencyjnego (C).

Zworki konfiguracji wyjść w bloku **C** pozwalają wybrać, czy linie wejściowe $o_1 - o_8, \bar{o}_1 - \bar{o}_8$ w konfigurowalnych bramkach AND są połączone z wyjściami kombinacyjnymi (bezpośrednio z bramek OR) czy wyjściami sekwencyjnymi (rejestrowymi) wybranych bloków logicznych. W bloku **C**, litera „C" oznacza wyjścia kombinacyjne, „S" – wyjścia sekwencyjne. Stan logiczny wyjścia danego bloku sygnalizowany jest diodą koloru czerwonego, umiejscowioną obok zworek C/S.

Użycie przerzutników na wyjściach bloków logicznych wymaga zastosowania generatora sygnału zegarowego. Blok generacji sygnału zegarowego oznaczony jest na rys. 2.1. literą **D**. Stan logiczny sygnału zegarowego sygnalizowany jest diodą koloru niebieskiego, podpisaną „CLK". Widok szczegółowy bloku generacji zegara przedstawia rys. 2.3.



Rys. 2.3. Blok generacji sygnału zegarowego.

W programowanej zworkowo matrycy logicznej źródło impulsów zegarowych ustala pojedyncza zworka umieszczona w odpowiedniej pozycji w gnieździe znajdującym się obok napisu „CLK" i pokazanym na rysunku 2.3. Źródłem mogą być:

1) scalony generator impulsów zegarowych HCF4060. Generator scalony pozwala na wybór częstotliwości zegara w zakresie 0.25 – 256Hz, poprzez zapięcie odpowiednio oznaczonej zworki (rys. 2.3),
2) przycisk CLK – wybór tego źródła dokonuje się poprzez zapięcie zworki oznaczonej jako „SW" w bloku generacji zegara,
3) pin CLK w złączu sygnałów wejściowych **E,** wybierany poprzez zapięcie zworki w pozycji „INP",
4) Gniazdo BNC znajdujące się po północnej stronie płytki – wyboru dokonuje się poprzez zapięcie zworki w pozycji „EXT".

W blokach logicznych zastosowano przerzutniki z asynchronicznym wejściem resetującym. W bloku generacji zegara przycisk „RST" pozwala zresetować wszystkie przerzutniki na płytce.

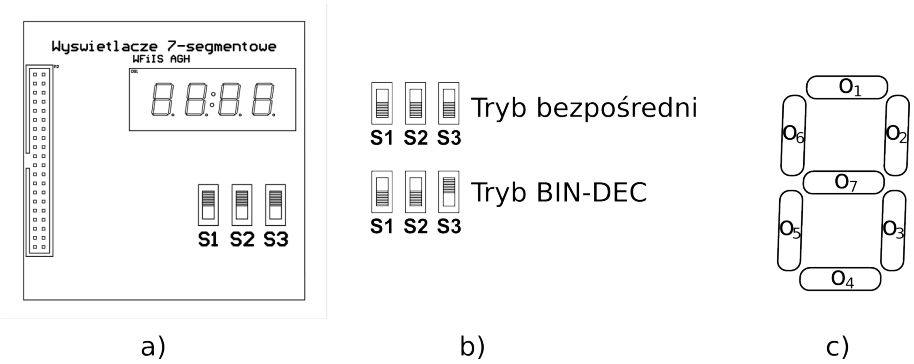
Programowalna matryca logiczna została wyposażona w dwa złącza do komunikacji z modułami zewnętrznymi. Sygnały wejściowe mogą być podawane z zewnątrz za pomocą złącza **E**, sygnały wyjściowe mogą być odczytywane poprzez złącze **F**.

3. Moduły rozszerzeń

Programowalna matryca logiczna wyposażona została w dwa moduły rozszerzeń. Moduły te mają na celu ułatwienie weryfikacji zaprojektowanych układów logicznych (sekwencyjnych i kombinacyjnych) oraz obserwację praktycznego działania budowanych układów.

3.1 Wyświetlacz 7-segmentowy 4-pozycyjny

Wyświetlacz 7-segmentowy 4-pozycyjny jest modułem, który umożliwia prezentację wyniku działania układów logicznych w postaci cyfr dziesiętnych. Moduł posiada 8 wejść, które za pomocą kabla taśmowego połączone są z wyjściami $o_1 - o_8$ matrycy logicznej poprzez złącze **F** na rysunku 2.1. Widok modułu wyświetlacza 4-ro pozycyjnego przedstawia rysunek 3.1.



Rys. 3.1. a) Widok modułu wyświetlacza 4-ro pozycyjnego, b) ustawienia przełączników i odpowiadające im tryby pracy, c) segmenty wyświetlacza odpowiadające wyjściom matrycy logicznej podczas pracy w trybie bezpośrednim

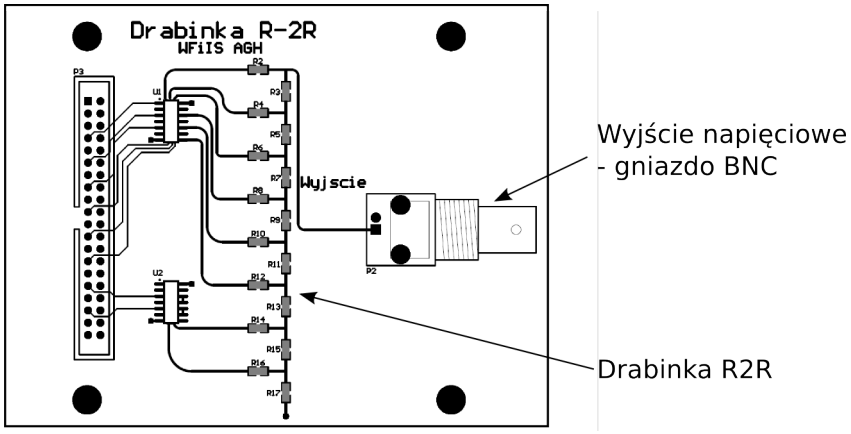
Wyświetlacz wyposażony jest w układ mikroprocesorowy (umieszczony na spodniej stronie płytki PCB), który umożliwia pracę w dwóch trybach: bezpośrednim i konwersji BIN-DEC. Wyboru trybu dokonuje się poprzez ustawienie przełączników S1 – S3 (rys. 3.1.b):

a) tryb bezpośredni – w tym trybie aktywna jest tylko pierwsza pozycja wyświetlacza, licząc od strony prawej. Wejścia modułu podłączone są bezpośrednio do segmentów wyświetlacza na pierwszej pozycji, sposób połączeń oraz oznaczenia segmentów przedstawia rysunek 3.1.c. Aby moduł wyświetlacza pracował w trybie bezpośrednim, wszystkie przełączniki S1, S2 i S3 muszą znajdować się w pozycji „0".

b) tryb BIN-DEC – tryb ten służy do obserwacji 8-bitowej liczby binarnej w formacie dziesiętnym (co odpowiada wartościom dziesiętnym 0-255). Po podłączeniu do matrycy programowalnej, wyjście o_1 odpowiada najmłodszemu, a o_8 – najstarszemu bitowi. Aby moduł wyświetlacza pracował w trybie konwersji BIN-DEC, przełączniki S1, S2 muszą znajdować się w pozycji „0", a przełącznik S3 w pozycji „1".

3.2 Przetwornik cyfrowo-analogowy R-2R

Moduł zawiera 8-bitowy przetwornik cyfrowo-analogowy. Przetwornik zrealizowany został w architekturze R-2R, zwanej drabinką rezystorową. Po połączeniu z programowalną matrycą logiczną, wartości logiczne kolejnych bitów wyjściowych $o_1 - o_8$ odpowiadają za włączanie – wyłączanie kolejnych bitów przetwornika R-2R, co pozwala uzyskać $2^8 = 256$ wartości analogowych napięcia w zakresie od 0 do 5V (przy zasilaniu układu napięciem 5V). Napięcie wyjściowe przetwornika może być mierzone poprzez gniazdo wyjściowy typu BNC. Widok modułu z przetwornikiem R-2R przedstawiony został na rysunku 3.2. Drabinka rezystorowa połączona jest z górnym i dolnym napięciem zasilania, dla kodu 0 przetwornik generuje napięcie równe 0V, dla kodu 255 napięcie równe napięciu zasilania (typowo używane jest napięcie 5V).



Rys. 3.2. Widok modułu przetwornika cyfrowo-analogowego R2R.

4. Program ćwiczenia, część I: logika kombinacyjna

4.1. Realizacja bramek logicznych

Korzystając z jednego wyjścia kombinacyjnego (np. o_1) zaprojektować w postaci **sumy iloczynów** następujące funkcje logiczne :

NOT, AND, NAND, OR, NOR, XOR, XNOR.

Funkcje te zaprogramować na matrycy logicznej, zweryfikować doświadczalnie poprawność połączeń.

Dla wybranej bramki logicznej zmierzyć czasy propagacji sygnału (przyjmując poziom połowy napięcia zasilania jako zmianę stanu logicznego) oraz czas narastania/opadania (10-90% amplitudy). Sygnał z generatora doprowadzić poprzez złącze **E**, zmiany stanu logicznego na wyjściu zaobserwować podłączając oscyloskop do odpowiedniego wyjścia poprzez złącze wyjściowe **F**.

4.2. Dekoder 1 z 8 na wyświetlacz 7-segmentowy

Zbudować układ kombinacyjny sterujący wyświetlaczem 7-segmentowym w taki sposób, aby po wciśnięciu jednego z 8 przycisków wejściowych na wyświetlaczu pojawiała się cyfra odpowiadająca kolejnemu wejściu (tzn po wciśnięciu A – 1, B – 2, C – 3 itd). Dekoder można również zmodyfikować w taki sposób, aby przy wszystkich przyciskach wejściowych zwolnionych, na wyświetlaczu pojawiało się „0”. Należy skorzystać z rys. 3.1 w celu identyfikacji połączeń wyświetlacza z wyjściami matrycy logicznej.

4.3. Dekoder kodu binarnego na wyświetlacz 7-segmentowy

Zbudować układ dekodujący liczby w kodzie binarnym na wyświetlacz 7-segmentowy prezentujący liczby w kodzie dziesiętnym. Należy użyć wejść A – D jako 4 bitów liczby binarnej oraz wyjść $o_1 - o_7$ jako sygnałów sterujących poszczególnymi segmentami wyświetlacza. Dla celów weryfikacji dekodera należy użyć modułu z 4-ro pozycyjnym wyświetlaczem 7-segmentowym. Moduł wyświetlacza należy ustawić w trybie pracy bezpośredniej, w którym wykorzystywany jest tylko wyświetlacz na pierwszej pozycji (patrz pkt. 4.1. instrukcji).

W wersji podstawowej wyświetlacz powinien prezentować liczby dziesiętne od 0 do 9, odpowiednio dla binarnych 0000 – 1001. W wersji rozszerzonej można rozbudować funkcjonalność konwertera o wyświetlanie znaków kodu szesnastkowego, czyli oprócz powyższych cyfr, również litery od A do F dla wartości binarnych od 1010 do 1111. Sposób połączenia wyświetlacza z matrycą oraz sposób ustawienia wyświetlacza w odpowiedni tryb pracy przedstawione zostały w pkt. 3.1 instrukcji. Należy zwrócić uwagę na właściwe połączenie zworek konfiguracji wyjścia sekwencyjnego/kombinacyjnego w wykorzystywanych blokach logicznych! Wszystkie wyjścia powinny zostać skonfigurowane jako wyjścia kombinacyjne!

5. *Program ćwiczenia, część II: logika sekwencyjna*

5.1. Liczniki, konwerter z kodu Gray'a na BCD

a) Zaprojektować i zbudować przy pomocy programowalnej matrycy logicznej licznik o zadanej pojemności (np. modulo 9), liczący w kodzie binarnym. Efekt działania licznika zaobserwować na diodach wyjściowych. Podłączyć moduł wyświetlacza ustawiony w trybie BIN-DEC (patrz pkt. 3.1 instrukcji) i zaobserwować efekt pracy licznika. Wszystkie wyjścia powinny zostać skonfigurowane jako wyjścia sekwencyjne!

b) Zapisać tabelę zawierającą wartości dziesiętne (kolejno od 0 do 15), odpowiadające im wartości binarne w kodzie Gray'a oraz wartości binarne w kodzie BCD. Zaprojektować i zbudować licznik 4-ro bitowy, liczący w kodzie Gray'a. Do budowy licznika wykorzystać 4 dolne bloki funkcjonalne (z wyjściami o_5-o_8). Efekt działania licznika zaobserwować na diodach wyjściowych o_5-o_8. Licznik zmontować w taki sposób, aby wyjście o_5 było najmłodszym, a o_8 najstarszym bitem.

c) Dla układu zbudowanego w punkcie b) zbudować konwerter z kodu Gray'a na kod BCD. Do montażu układu wykorzystać 4 „górne" bloki programowalnej matrycy logicznej (z wyjściami o_1-o_4). Wejścia konwertera powinny stanowić wyjścia licznika (o_5-o_8). Konwerter zmontować w taki sposób, aby wyjście o_1 było najmłodszym, a o_4 – najstarszym bitem kodu BCD. Należy zwrócić szczególną uwagę na właściwe połączenie zworek konfiguracji wyjścia sekwencyjnego/kombinacyjnego w poszczególnych blokach! Wyjścia o_1-o_4 powinny być skonfigurowane jako kombinacyjne, natomiast o_5-o_8 – jako sekwencyjne.
Podłączyć moduł wyświetlacza ustawiony w trybie BIN-DEC (patrz pkt. 3.1 instrukcji), zweryfikować poprawność projektu sprawdzając zgodność przygotowanej tabeli z wyświetlanymi liczbami dziesiętnymi tablicy efekt pracy licznika.

5.2. Licznik sterujący przetwornikiem R2R

Zaprojektować i zbudować przy pomocy programowalnej matrycy logicznej licznik w kodzie binarnym liczący pomiędzy dwoma wartościami (np. od 30 do 46). Zweryfikować poprawność projektu przy pomocy diod wyjściowych lub modułu wyświetlacza. Podłączyć moduł zewnętrzny z przetwornikeim cyfrowo-analogowym R2R.

a) W module generacji sygnału zegarowego wybrać zworkę „CLK", aby móc za pomocą przycisku CLK ręcznie inkrementować ustawienie licznika. Dla kolejnych wartości licznika wykonać za pomocą multimetru pomiar napięcia generowanego przez przetwornik.

b) Zmienić źródło zegara na generator scalony, wybrać częstotliwość 128Hz lub 256Hz, zaobserwować przebieg napięcia na oscyloskopie.

6. Przygotowanie sprawozdania

Do przygotowania sprawozdania należy użyć zamieszczonych poniżej schematów zworek – w miejscach gdzie wpinano zworki należy zaznaczyć kratkę.

6.1. Realizacja bramek logicznych
Zaznaczyć na schemacie (dołączonym na końcu instrukcji) konfiguracje zworek dla poszczególnych bramek. Podać czasy narastania i propagacji zmierzone na wyjściu wybranych bramek logicznych. Biorąc pod uwagę fakt, że bramki zrealizowane są w architekturze iloczynu galwanicznego, jak wytłumaczysz różnice w czasach narastania i opadania?

6.2. Logika kombinacyjna – dekoder 1 z 8 na wyświetlacz 7-segmentowy

W sprawozdaniu należy zamieścić wyrażenia logiczne opisujące wszystkie bity wyjściowe o_1 – o_7 w funkcji sygnałów wejściowych A – D. Załączyć schemat wykonanych połączeń. Jakiej modyfikacji należy dokonać w celu wyświetlania zera, gdy wszystkie sygnały wejściowe ustawione są na „0”?

6.3. Logika kombinacyjna – dekoder kodu binarnego na wyświetlacz 7-segmentowy
W sprawozdaniu należy zamieścić wyrażenia logiczne opisujące wszystkie bity wyjściowe $o_1 - o_7$ w funkcji sygnałów wejściowych A-D. Załączyć schemat wykonanych połączeń.

6.4. Logika sekwencyjna – liczniki, konwerter z kodu Gray'a na BCD
W sprawozdaniu należy zamieścić wyrażenia logiczne opisujące wszystkie bity wyjściowe $o_1 - o_4$ konwertera kodu Gray'a na BCD oraz wszystkie wyrażenia logiczne opisujące stan następny bitu kodu Gray'a w funkcji stanu poprzedniego. Załączyć schemat wykonanych połączeń.

6.5. Licznik sterujący przetwornikiem R2R
W sprawozdaniu umieścić wyrażenia logiczne opisujące wszystkie sygnały wyjściowe oraz schematy wykonanych połączeń. Dla zmierzonych wartości generowanych przez przetwornik cyfrowo-analogowy, porównać wartości zmierzone z teoretycznie oczekiwanymi uwzględniając używane napięcie zasilania.

7. Literatura

[1] http://www.xilinx.com/company/gettingstarted/fpgavsasic.htm#pcs
[2] Ming-Bo Lin: „Digital Systems Designs and Practices”, John Wiley & Sons (Asia) Pte Ltd, Singapore 2008
[3] John F.Wakerly: „Digital Design, Principles and Practices”, Pearson Education, wyd. 4, New Jersey, 2005
[4] U.Tietze, C.Schenk: „Układy półprzewodnikowe”, Wydawnictwa Naukowo-Techniczne, Warszawa

Schemat zworek Matrycy PLD