

SYNTEZA KOMBINACYJNYCH UKŁADÓW STEROWANIA

SPIIS TREŚCI

Synteza układów kombinacyjnych	1
Etapy syntezy układu kombinacyjnego	1
Metody opisu układów kombinacyjnych	2
Metody minimalizacji wyrażeń logicznych	6
Realizacja układu	12
Ocena kosztu układu	15
Ćwiczenie laboratoryjne	16
Stanowisko laboratoryjne	16
Zadania do wykonania	19
Pytania kontrolne	20
Literatura	20

SYNTEZA UKŁADÓW KOMBINACYJNYCH

Układy cyfrowe stanowią grupę układów elektronicznych przeznaczonych do przetwarzania sygnałów dyskretnych, przyjmujących najczęściej dwa poziomy logiczne oznaczane jako 0 i 1. Ich podstawowymi elementami są bramki logiczne realizujące funkcje algebry Boole'a.

Ze względu na sposób przetwarzania informacji układy logiczne dzieli się na układy kombinacyjne oraz układy sekwencyjne.

- W układach kombinacyjnych wartości sygnałów wyjściowych są jednoznacznie określone przez aktualne wartości sygnałów wejściowych. Oznacza to, że układy te nie posiadają elementów pamięci, a każdej kombinacji stanów wejściowych odpowiada jedna, ściśle określona kombinacja stanów wyjściowych.
- W przeciwieństwie do nich układy sekwencyjne zawierają elementy pamięci, dlatego wartości sygnałów wyjściowych zależą nie tylko od bieżących wartości sygnałów wejściowych, lecz również od poprzedniego stanu układu.

Układy kombinacyjne znajdują zastosowanie w realizacji funkcji logicznych wykorzystywanych między innymi w układach dekodów, koderów, multiplexerów, demultiplexerów, komparatorów, sumatorów oraz blokach decyzyjnych cyfrowych systemów sterowania.

ETAPY SYNTEZY UKŁADU KOMBINACYJNEGO

Projektowanie układów kombinacyjnych, określane jako synteza układów kombinacyjnych, polega na wyznaczeniu zależności logicznych pomiędzy sygnałami wejściowymi i wyjściowymi oraz opracowaniu schematu logicznego realizującego wymaganą funkcję. Celem syntezy jest uzyskanie poprawnego funkcjonalnie rozwiązania przy możliwie najmniejszej złożoności sprzętowej.

Proces syntezy układu kombinacyjnego można przedstawić w postaci następujących etapów:

1. **Analiza i opis zadania** – określenie działania projektowanego układu oraz wymagań funkcjonalnych. Wyznaczenie sygnałów wejściowych $x_1, x_2, \dots, x_n$ oraz sygnałów wyjściowych $y_1, y_2, \dots, y_m$, jeżeli nie zostały jednoznacznie określone w treści zadania. Określenie funkcji logicznych opisujących zależności pomiędzy sygnałami wejściowymi i wyjściowymi, najczęściej w postaci tablicy prawdy, tabeli stanów lub równań logicznych.
2. **Minimalizacja funkcji logicznych** w celu uzyskania ich możliwie najprostszej postaci, np. z wykorzystaniem algebry Boole'a, siatki Karnaugh'a.
3. **Realizacja układu logicznego** poprzez opracowanie schematu z wykorzystaniem odpowiednich bramek logicznych lub układów scalonych.
4. **Ocena i optymalizacja rozwiązania** obejmująca analizę złożoności układu, liczby zastosowanych elementów, liczby wejść bramek, opóźnień propagacji oraz – w razie potrzeby – modyfikację schematu.

METODY OPISU UKŁADÓW KOMBINACYJNYCH

Pierwszym etapem syntezy układu kombinacyjnego jest analiza treści zadania oraz określenie funkcji, jaką ma realizować projektowany układ. Na tym etapie identyfikuje się wymagania funkcjonalne oraz ustala zależności pomiędzy sygnałami wejściowymi i wyjściowymi.

Układ kombinacyjny może być opisany na kilka równoważnych sposobów. Wybór metody opisu zależy od etapu projektowania oraz stopnia szczegółowości analizowanego zagadnienia. Najczęściej stosowane metody przedstawiono poniżej.

- **Opis słowny** – określa zasadę działania układu oraz warunki, w których poszczególne sygnały wyjściowe przyjmują określone wartości logiczne.
- **Tablica prawdy** – przedstawia wszystkie możliwe kombinacje wartości sygnałów wejściowych oraz odpowiadające im wartości sygnałów wyjściowych.
- **Funkcja logiczna (boolowska)** – algebraiczny opis zależności pomiędzy sygnałami wejściowymi i wyjściowymi. Funkcja może być przedstawiona w postaci:
 - kanonicznej sumy iloczynów (SOP, *Sum of Products*),
 - kanonicznego iloczynu sum (POS, *Product of Sums*).

W praktyce proces syntezy polega na przechodzeniu pomiędzy kolejnymi sposobami opisu. Najczęściej punktem wyjścia jest opis słowny, na podstawie którego wyznacza się tablicę prawdy, następnie funkcje logiczne, a ostatecznie opracowuje schemat realizujący wymagane zależności.

Opis słowny

Opis słowny stanowi najczęściej punkt wyjścia procesu syntezy układu kombinacyjnego. Określa on wymagane działanie projektowanego układu, opisując zależności pomiędzy sygnałami wejściowymi i wyjściowymi. Na jego podstawie projektant wyznacza zmienne wejściowe i wyjściowe oraz opracowuje tablicę prawdy lub funkcje logiczne.

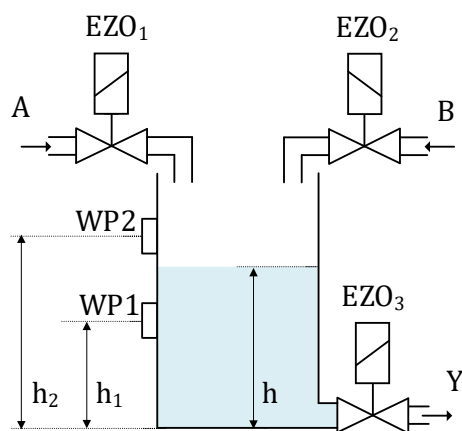
Dodatkowe założenia projektowe

Opis słowny nie zawsze zawiera jednak wszystkie informacje niezbędne do jednoznacznego zaprojektowania układu. Opis ten otrzymujemy najczęściej od zlecniodawcy, co prowadzi do tego, że specyfikacja może być niepełna, zawierać niejednoznaczności lub pomijać sytuacje wyjątkowe, które mogą wystąpić podczas eksploatacji. Zadaniem projektanta jest zidentyfikowanie takich przypadków oraz uzupełnienie opisu o odpowiednie założenia projektowe. Opis funkcjonalny powinien jednoznacznie określać warunki, dla których sygnały wyjściowe przyjmują wartość logiczną 0 lub 1.

Do najczęściej przyjmowanych założeń projektowych należą:

- określenie reakcji układu na stany niepoprawne lub nieosiągalne,
- określenie sposobu postępowania w przypadku uszkodzenia czujników lub elementów wykonawczych,
- zdefiniowanie stanu bezpiecznego (fail-safe), do którego układ powinien przejść w przypadku wykrycia awarii,
- określenie priorytetów działania, jeżeli jednocześnie spełnionych jest kilka warunków sterowania,
- zdefiniowanie stanów początkowych oraz warunków uruchomienia i zatrzymania układu,
- uwzględnienie właściwości elementów wykonawczych, takich jak sposób sterowania (aktywny stan wysoki lub niski), czas działania czy wzajemne blokady.

Przyjęte założenia powinny zostać jednoznacznie określone przed rozpoczęciem syntezy funkcji logicznych, ponieważ mają bezpośredni wpływ na postać tablicy prawdy oraz otrzymane funkcje logiczne.



Rys. 1. Układ regulacji pompy

Przykładowo, podczas projektowania układu sterowania pompą napełniającą zbiornik z wykorzystaniem dwóch czujników poziomu cieczy (Rys. 1) może wystąpić sytuacja, w której czujnik górnego poziomu (WP2) sygnalizuje obecność cieczy, natomiast czujnik dolnego poziomu

(WP1) pozostaje nieaktywny. Taki stan jest fizycznie niemożliwy podczas prawidłowej pracy i może świadczyć o uszkodzeniu jednego z czujników. W takiej sytuacji projektant powinien określić sposób działania układu, m.in.:

- nie uwzględniać stanów awaryjnych,
- przełączyć układ do stanu bezpiecznego, np. wyłączyć pompę,
- wygenerować sygnał alarmowy informujący o wykryciu nieprawidłowości,
- rozróżniać rodzaje awarii i sygnalizować je za pomocą odrębnych sygnałów wyjściowych.

Tablica prawdy

Tablica prawdy stanowi podstawową metodę opisu działania układu kombinacyjnego. Przedstawia ona zależność pomiędzy wszystkimi możliwymi kombinacjami wartości sygnałów wejściowych a odpowiadającymi im wartościami sygnałów wyjściowych. Każdy wiersz tablicy odpowiada jednej kombinacji stanów wejściowych, dla której określane są wartości wszystkich sygnałów wyjściowych.

Tablicę prawdy można utworzyć według następującej procedury:

- 1

Określ liczbę sygnałów wejściowych x oraz sygnałów wyjściowych y oraz utwórz kolumny im odpowiadające.

Wartości sygnałów wyjściowych są wyznaczone na podstawie analizy opisu słownego oraz przyjętych założeń projektowych.
- 2

Wyznacz liczbę wszystkich kombinacji sygnałów wejściowych

Liczba wierszy tablicy prawdy zależy od liczby sygnałów wejściowych i wynosi:

$$N = 2^n$$

gdzie: N – liczba kombinacji stanów wejściowych,
 n – liczba sygnałów wejściowych.

- 3

Wypełnij kolumny sygnałów wejściowych, zapisując wszystkie możliwe kombinacje stanów logicznych w kolejności odpowiadającej rosnącym wartościom liczby binarnej

Kolejność kombinacji wejściowych w tablicy prawdy jest zazwyczaj zgodna z rosnącą wartością liczby binarnej utworzonej przez sygnały wejściowe, rozpoczynając od kombinacji 00...0 i kończąc na 11...1

Do przejścia z zapisu binarnego na dziesiętny stosuje się wzór, w którym każda cyfra binarna jest mnożona przez odpowiednią potęgę liczby 2, zależną od jej położenia. Najmniej znaczący bit ma numer 0, a kolejne bity są numerowane rosnąco w kierunku lewej strony.

$$D = \sum_{i=0}^{n-1} b \cdot 2^i$$

gdzie: D – wartość liczby w systemie dziesiętnym,
 b – wartość i -tego bitu (0 lub 1), i – numer pozycji liczony od prawej strony (najmniej znaczący bit ma numer 000).

Przykład

$$\begin{aligned} 10 &= 1 \cdot 2^1 + 0 \cdot 2^0 = 2 + 0 = 2 \\ 1011 &= 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = \\ &= 8 + 0 + 2 + 1 = 11 \end{aligned}$$

- 4

Wyznacz wartości sygnałów wyjściowych dla każdej kombinacji sygnałów wejściowych na podstawie opisu słownego oraz przyjętych założeń projektowych.

Każdej kombinacji sygnałów wejściowych należy przypisać jednoznacznie określone wartości wszystkich wyjść układu. Jeżeli opis zadania przewiduje występowanie stanów nieosiągalnych lub nieistotnych z punktu widzenia działania układu, mogą one zostać oznaczone jako **stany obojętne**, co umożliwi uzyskanie prostszej postaci funkcji logicznych na etapie minimalizacji.

Tab. 1. Kolejność kombinacji sygnałów wejściowych (2 wejścia)

x_1	x_2	Wartość dziesiętna
0	0	0
0	1	1
1	0	2
1	1	3

$\downarrow$ $\downarrow$
 2^1 2^0

Tab. 2. Kolejność kombinacji sygnałów wejściowych (4 wejścia)

x_1	x_2	x_3	x_4	Wartość dziesiętna
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$
 2^3 2^2 2^1 2^0

Postać kanoniczna sumy

Postać kanoniczna sumy jest zapisem funkcji logicznej w postaci **sumy iloczynów (mintermów)**. Każdy iloczyn zawiera **wszystkie zmienne wejściowe**, występujące w postaci prostej lub zanegowanej.

Postać kanoniczną sumy wyznacza się bezpośrednio z tablicy prawdy, uwzględniając wyłącznie te wiersze, dla których wartość funkcji wynosi **1**. Dla każdego takiego wiersza tworzy się jeden iloczyn, stosując następującą zasadę:

- zmienne o wartości **1** zapisuje się w postaci niezanegowanej,
- zmienne o wartości **0** zapisuje się w postaci zanegowanej.

Ostateczna funkcja jest sumą wszystkich otrzymanych iloczynów.

Postać kanoniczna iloczynu

Postać kanoniczna iloczynu jest zapisem funkcji logicznej w postaci **iloczynu sum (makstermów)**. Każda suma zawiera **wszystkie zmienne wejściowe**, występujące w postaci prostej lub zanegowanej.

Postać kanoniczną iloczynu wyznacza się bezpośrednio z tablicy prawdy, uwzględniając wyłącznie te wiersze, dla których wartość funkcji wynosi **0**. Dla każdego takiego wiersza tworzy się jedną sumę, stosując następującą zasadę:

- zmienne o wartości **0** zapisuje się w postaci niezanegowanej,
- zmienne o wartości **1** zapisuje się w postaci zanegowanej.

Ostateczna funkcja jest iloczynem wszystkich otrzymanych sum.

Przykład

Tab. 3. Przykładowa
tablica prawdy
dla 3 zmiennych

x_1	x_2	x_3	y
0	0	0	0
0	0	1	-
0	1	0	0
0	1	1	1
1	0	0	-
1	0	1	1
1	1	0	1
1	1	1	0

Postać kanoniczna sumy:

Uwzględniamy $y = 1$, czyli 011, 101, 110:

$$y = \overline{x_1} \cdot x_2 \cdot x_3 + x_1 \cdot \overline{x_2} \cdot x_3 + x_1 \cdot x_2 \cdot \overline{x_3}$$

Postać kanoniczna iloczynu:

Każda suma pełna odpowiada jednej linii w tablicy prawdy, dla której funkcja przyjmuje wartość zero, czyli 000, 010, 111

$$y = (x_1 + x_2 + x_3) \cdot (x_1 + \overline{x_2} + x_3) \cdot (\overline{x_1} + \overline{x_2} + \overline{x_3})$$

Każdą funkcję logiczną można jednoznacznie przedstawić zarówno w kanonicznej postaci sumy (SOP), jak i kanonicznej postaci iloczynu (POS). Postacie te są równoważne i mogą zostać wyznaczone bezpośrednio na podstawie tablicy prawdy. Należy jednak zauważyć, że w większości przypadków nie są to postacie minimalne, dlatego przed realizacją sprzętową funkcje logiczne poddaje się procesowi minimalizacji.

METODY MINIMALIZACJI WYRAŻEŃ LOGICZNYCH

Minimalizacja wyrażeń logicznych jest ważnym etapem projektowania układów cyfrowych, ponieważ pozwala zmniejszyć liczbę bramek logicznych, a tym samym obniżyć koszt, zużycie energii oraz złożoność układu.

- Do najprostszych metod należy minimalizacja z wykorzystaniem **praw algebry Boole’a**. Polega ona na ręcznym przekształcaniu funkcji logicznych za pomocą takich praw jak rozdzielność, absorpcja czy prawa de Morgana. Metoda ta jest łatwa do zastosowania dla prostych funkcji, jednak dla bardziej złożonych wyrażeń staje się czasochłonna i nie zawsze prowadzi do rozwiązania minimalnego.
- Popularną metodą graficzną jest **metoda tablic Karnaugh**. Funkcję logiczną przedstawia się w postaci specjalnej siatki, w której grupuje się sąsiednie jedynki lub zera. Umożliwia ona szybkie wyznaczenie postaci minimalnej funkcji dla niewielkiej liczby zmiennych, dlatego jest powszechnie wykorzystywana w nauczaniu oraz przy projektowaniu prostych układów cyfrowych.
- Dla funkcji o większej liczbie zmiennych stosuje się **metodę Quine’a-McCluskey’a**. Jest to metoda algorytmiczna pozwalająca w sposób systematyczny znaleźć minimalną postać funkcji logicznej. Stanowi ona komputerowy odpowiednik map Karnaugh’a i jest wykorzystywana w programach wspomagających projektowanie układów cyfrowych.

- Wśród mniej popularnych metod można wymienić metodę Kazakowa, metodę Tablic Niezgodności (TN) oraz metodę bezpośredniego przeszukiwania (BP).
 - **Metoda Kazakowa** ma głównie znaczenie dydaktyczne i jest rzadko stosowana w praktyce.
 - **Metoda TN** znajduje zastosowanie przede wszystkim w minimalizacji automatów sekwencyjnych poprzez analizę stanów zgodnych i niezgodnych.
 - Z kolei **metoda BP** polega na sprawdzeniu wszystkich możliwych rozwiązań i wybraniu najlepszego, jednak ze względu na bardzo dużą złożoność obliczeniową wykorzystuje się ją jedynie dla niewielkich problemów.
- Obecnie w profesjonalnych narzędziach do syntezy układów cyfrowych najczęściej stosowane są **zaawansowane algorytmy komputerowe**, takie jak Espresso, które pozwalają efektywnie minimalizować bardzo złożone funkcje logiczne. Mimo to podstawowe metody, takie jak algebra Boole’a, mapy Karnaugh’a oraz metoda Quine’a-McCluskey’a, nadal pozostają najważniejszymi narzędziami wykorzystywanymi w edukacji oraz analizie układów cyfrowych.

Prawa Algebry Boole’a

Podstawowe znaczenie w automatyce mają następujące prawa:

Prawo przemienności	$A + B = B + A$	$A \cdot B = B \cdot A$
Prawo łączności	$(A + B) + C = A + (B + C)$	$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
Prawo rozdzielności	$A + B \cdot C = (A + B) \cdot (A + C)$	$A \cdot (B + C) = A \cdot B + A \cdot C$
Prawo pochłaniania	$A + A \cdot B = A$	$A \cdot (A + B) = A$
Prawo podwójnej negacji	$\bar{\bar{A}} = A$	
Prawa de Morgana	$\overline{A + B} = \bar{A} \cdot \bar{B}$	$\overline{A \cdot B} = \bar{A} + \bar{B}$
Prawo dominacji	$A + 1 = 1$	$A \cdot 0 = 0$
Prawo tożsamości	$A + 0 = A$	$A \cdot 1 = A$
Prawo idempotentności (powtórzenia)	$A + A = A$	$A \cdot A = A$
Prawo komplementarności (dopełnienia)	$A + \bar{A} = 1$	$A \cdot \bar{A} = 0$

Sprawdźmy, czy postaci kanoniczne sumy i iloczynu uzyskane dla prostszego przykładu są równoważne wykorzystując prawa algebry Boole'a:

Przykład

Tab. 4. Przykładowa tablica prawdy dla układu z 2 zmiennymi

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Postać kanoniczna sumy: $y = \overline{x}_1 \cdot x_2 + x_1 \cdot \overline{x}_2$

Postać kanoniczna iloczynu: $y = (x_1 + x_2)(\overline{x}_1 + \overline{x}_2)$

Sprawdzenie używając algebry Boole'a:

Wychodzimy z postaci kanonicznej iloczynu.

Korzystamy z prawa rozdzielności:

$$y = (x_1 + x_2)(\overline{x}_1 + \overline{x}_2) = x_1 \cdot \overline{x}_1 + x_1 \cdot \overline{x}_2 + x_2 \cdot \overline{x}_1 + x_2 \cdot \overline{x}_2$$

następnie z prawa komplementarności:

$$y = 0 + x_1 \cdot \overline{x}_2 + \overline{x}_1 \cdot x_2 + 0$$

Tym samym dochodzimy do postaci kanonicznej sumy.

Zaprezentowany układ to bramka XOR.

Uwaga! W przypadku występowania stanów nieoznaczonych różne postaci funkcji logicznej mogą różnić się wartościami dla tych stanów, zachowując jednocześnie poprawność działania układu dla wszystkich stanów określonych.

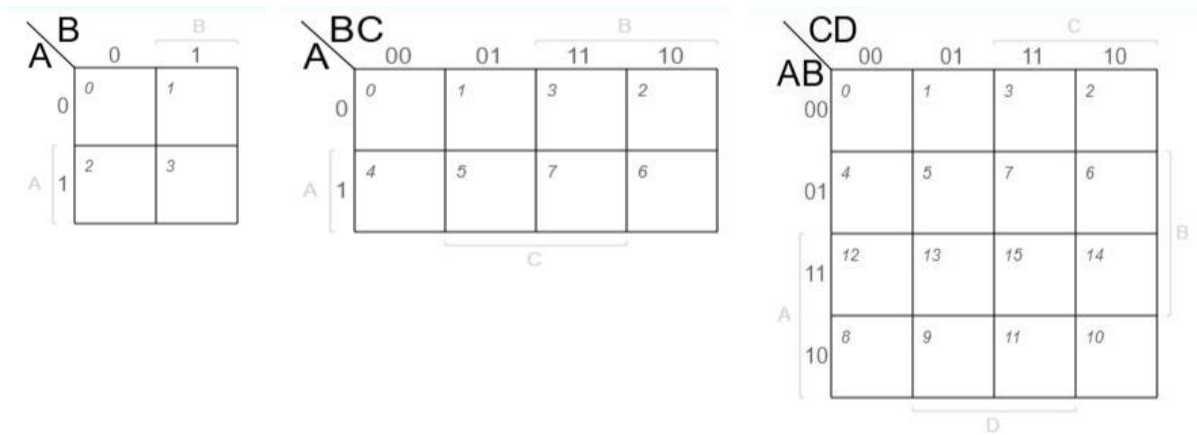
Tablica Karnaugh

Tablica Karnaugh jest szczególną postacią tablicy prawdy, w której wartości funkcji logicznej przedstawiono w postaci odpowiednio uporządkowanej siatki pól. Metoda tablic Karnaugh jest graficzną metodą minimalizacji funkcji logicznych, wykorzystującą zdolność człowieka do rozpoznawania wzorców i zależności geometrycznych.

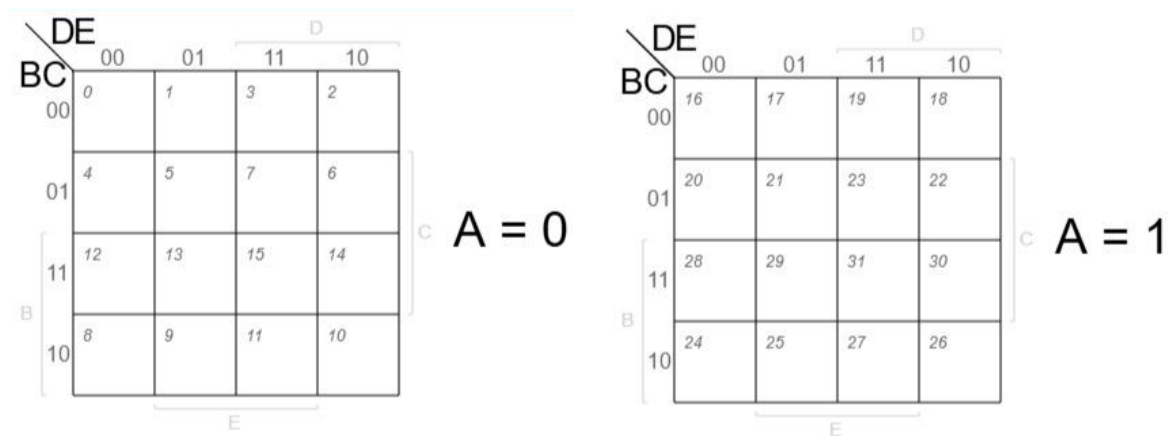
Mapa Karnaugh, znana także jako mapa K, została wynaleziona w 1953 roku przez Maurice'a Karnaugh, fizyka i matematyka pracującego w laboratoriach Bell.

Podstawą konstrukcji tablic Karnaugh jest **kod Graya**, którego właściwością jest to, że dwa sąsiednie słowa kodowe różnią się wartością tylko jednej pozycji bitowej. Dzięki takiemu uporządkowaniu zmiennych w wierszach i kolumnach tablicy sąsiednie pola odpowiadają kombinacjom wejściowym różniącym się tylko jedną zmienną.

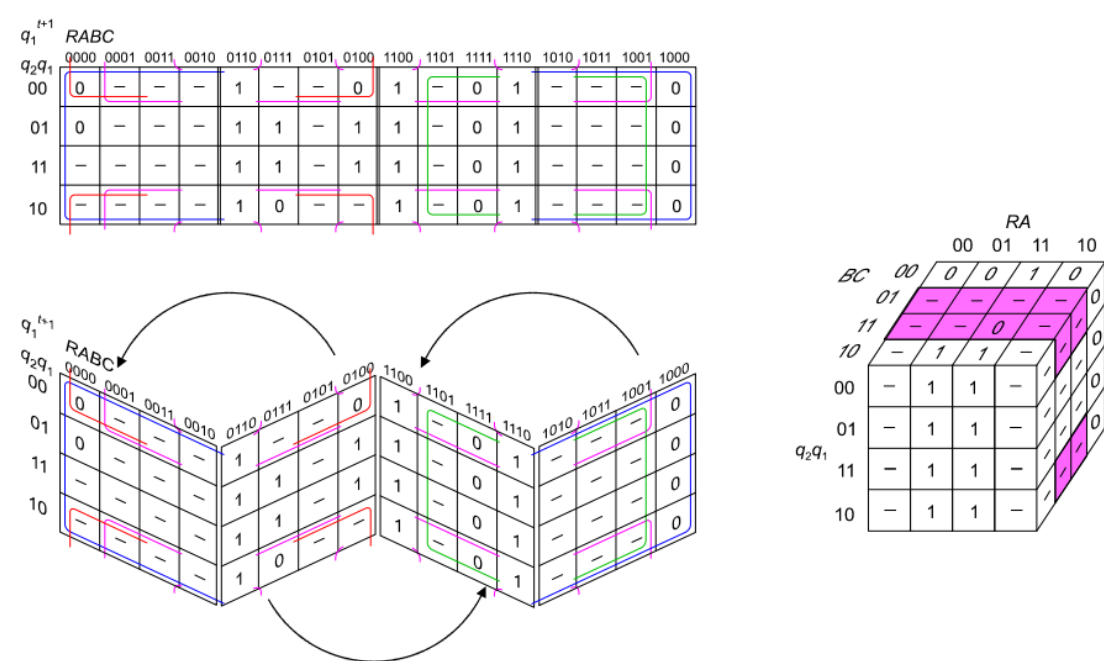
Ta właściwość umożliwia wykorzystanie **prawa sklejan** podczas grupowania sąsiednich pól zawierających wartości 1 (lub wartości 0 w przypadku minimalizacji w postaci iloczynu). W wyniku łączenia odpowiednich grup pól możliwe jest usunięcie zmiennych, które nie wpływają na wartość funkcji, a tym samym uzyskanie jej postaci minimalnej.



Rys. 2. Tablice Karnaugh dla 2, 3 i 4 zmiennych [2].



Rys. 3. Tablica Karnaugh dla 5 zmiennych [2].



Rys. 4. Tablica Karnaugh dla 6 zmiennych z jej trójwymiarową reprezentacją [3].

Tablice Karnaugh dla dużej liczby zmiennych

Tablice Karnaugh najczęściej stosuje się dla funkcji zawierających od dwóch do czterech zmiennych (Rys. 2), ponieważ wraz ze wzrostem liczby zmiennych liczba pól tablicy rośnie wykładniczo. Dla funkcji o większej liczbie zmiennych nie jest możliwe wygodne przedstawienie wszystkich pól wraz z zachowaniem relacji sąsiedztwa na jednej płaszczyźnie (Rys. 4). W takim przypadku tablicę Karnaugh przedstawia się jako kilka połączonych tablic czterozmiennych 4×4 , zachowując odpowiednie zależności wynikające z kodu Graya.

Minimalizacja funkcji logicznych

Minimalizacja funkcji logicznych jest etapem syntezy układów kombinacyjnych, którego celem jest uzyskanie możliwie najprostszej postaci funkcji przy zachowaniu jej pierwotnego działania. Uproszczenie funkcji pozwala zmniejszyć liczbę elementów logicznych potrzebnych do realizacji układu, ograniczyć jego złożoność oraz zmniejszyć koszt wykonania.

Metoda tablic Karnaugh wykorzystuje graficzną interpretację praw algebry Boole'a, w szczególności prawo sklejania. Minimalizacja funkcji polega na łączeniu sąsiednich pól tablicy Karnaugh zawierających jednakowe wartości logiczne:

- **grupowanie pól zawierających wartość 1** prowadzi do uzyskania minimalnej postaci sumacyjnej (SOP – suma iloczynów),
- **grupowanie pól zawierających wartość 0** prowadzi do uzyskania minimalnej postaci iloczynowej (POS – iloczyn sum).

Pola zawierające wartości nieokreślone, oznaczane symbolem „–” lub „X”, mogą zostać dołączone do grup jedynek albo grup zer, w zależności od tego, która możliwość prowadzi do prostszej postaci funkcji.

Zasady tworzenia grup pól

Podczas minimalizacji należy przestrzegać następujących zasad:

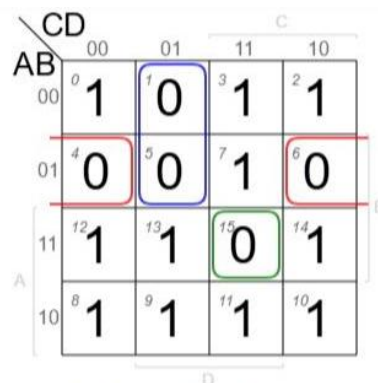
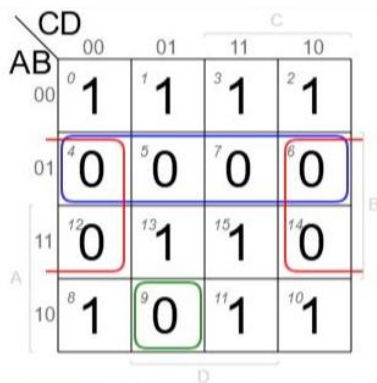
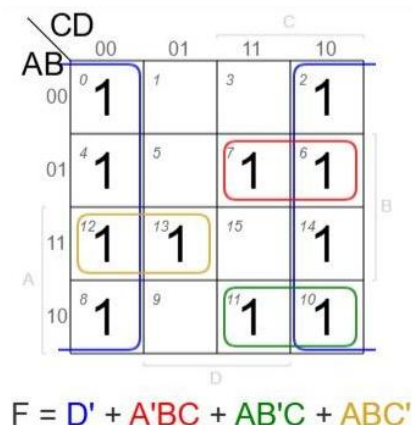
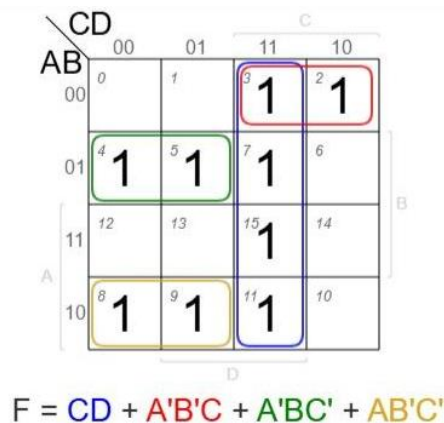
1. Grupowane mogą być wyłącznie pola sąsiadujące, czyli takie, których kombinacje zmiennych wejściowych różnią się tylko jedną zmienną.
2. Liczba pól w grupie musi być potęgą liczby 2: tj. 1, 2, 4, 8, 16, 32, ... 2^n
3. Grupy powinny obejmować **możliwie największą liczbę pól**, ponieważ większa grupa pozwala na usunięcie większej liczby zmiennych i uzyskanie prostszej funkcji.
4. Należy dążyć do utworzenia **jak najmniejszej liczby grup**, które obejmują wszystkie wymagane pola.
5. Dopuszczalne jest nakładanie się grup, jeżeli prowadzi to do uzyskania prostszej postaci funkcji.
6. Przeciwnieległe krawędzie tablicy Karnaugh traktuje się jako sąsiadujące, ponieważ odpowiadają one kombinacjom zmiennych różniącym się tylko jedną wartością.

Zapis funkcji po minimalizacji

Po zakończeniu grupowania:

- dla grup zawierających **jedynki** pozostawia się tylko te zmienne, które w całej grupie mają stałą wartość:
 - wartość **1** → zmienna niezanegowana,
 - wartość **0** → zmienna zanegowana.
- dla grup zawierających **zera** otrzymuje się postać iloczynu sum. W tym przypadku:
 - wartość **0** → zmienna niezanegowana,
 - wartość **1** → zmienna zanegowana.

Nie wykonuje się jednocześnie grupowania jedynek i zer. Wybór sposobu minimalizacji zależy od tego, która postać funkcji jest korzystniejsza z punktu widzenia dalszej realizacji układu, tj. wymagająca mniejszej liczby bramek logicznych.



Rys. 5. Przykłady grupowania wyrażeń sąsiednich. Wygenerowano w madformath.com [2]. Na Tablicach znacznik ' oznacza negację.

REALIZACJA UKŁADU

Po wyznaczeniu i minimalizacji funkcji logicznej kolejnym etapem syntezy układu kombinacyjnego jest opracowanie jego schematu realizacyjnego. Układy kombinacyjne mogą być wykonywane z wykorzystaniem różnych elementów realizujących podstawowe operacje logiczne.

Schemat logiczny

Układy kombinacyjne realizuje się najczęściej za pomocą **bramek logicznych**, czyli elementów elektronicznych wykonujących określone operacje logiczne na sygnałach wejściowych i generujących odpowiedni sygnał wyjściowy.

Podstawowe bramki logiczne oraz odpowiadające im funkcje logiczne przedstawiono w tabeli (Tab. 5).

W symbolach graficznych bramek logicznych:

- **kółko umieszczone na wyjściu bramki** oznacza negację realizowanej funkcji,
- **kółko umieszczone na wejściu bramki** oznacza zanegowany sygnał wejściowy.

Przykładowo:

- bramka AND z zanegowanym wyjściem realizuje funkcję NAND,
- bramka OR z zanegowanym wyjściem realizuje funkcję NOR.

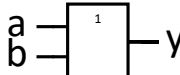
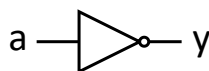
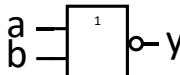
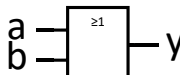
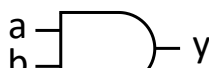
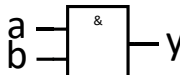
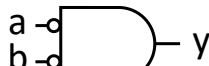
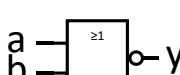
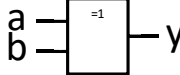
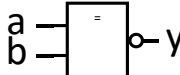
Schemat logiczny przedstawia połączenia pomiędzy bramkami realizującymi określoną funkcję logiczną. Jest on bezpośrednim odwzorowaniem zminimalizowanego wyrażenia boolowskiego.

Przykładowo:

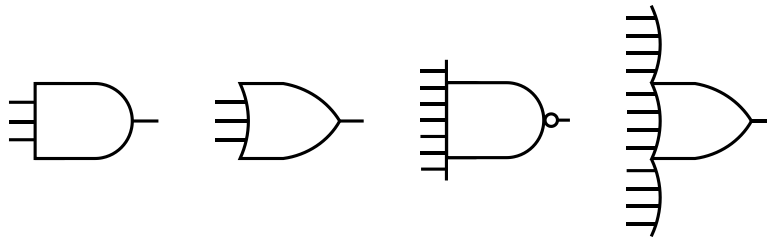
- minimalizacja poprzez grupowanie jedynek prowadzi do realizacji funkcji w postaci sumy iloczynów (SOP),
- minimalizacja poprzez grupowanie zer prowadzi do realizacji funkcji w postaci iloczynu sum (POS).

Obie postacie opisują tę samą funkcję logiczną, ale mogą prowadzić do różnych schematów realizacyjnych.

Tab. 5. Podstawowe bramki logiczne.

Nazwa bramki	Symbol wg ANSI	Symbol wg IEC	Funkcja logiczna	Tablica prawdy															
bufor			$y = a$	<table><tr><th>a</th><th>y</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	a	y	0	0	1	1									
a	y																		
0	0																		
1	1																		
NOT (negacja) (inverter)			$y = \bar{a}$	<table><tr><th>a</th><th>y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	a	y	0	1	1	0									
a	y																		
0	1																		
1	0																		
OR (suma)			$y = a + b$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	y	0	0	0	0	1	1	1	0	1	1	1	1
a	b	y																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	1																	
AND (iloczyn)			$y = a \cdot b$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	y	0	0	0	0	1	0	1	0	0	1	1	1
a	b	y																	
0	0	0																	
0	1	0																	
1	0	0																	
1	1	1																	
NOR (negacja sumy)	 		$y = \overline{a + b}$ $y = \bar{a} \cdot \bar{b}$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	y	0	0	1	0	1	0	1	0	0	1	1	0
a	b	y																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	0																	
NAND (negacja iloczynu)	 		$y = \overline{a \cdot b}$ $y = \bar{a} + \bar{b}$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	y	0	0	1	0	1	1	1	0	1	1	1	0
a	b	y																	
0	0	1																	
0	1	1																	
1	0	1																	
1	1	0																	
XOR			$y = a \cdot \bar{b} + \bar{a} \cdot b$ $y = a \oplus b$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	y	0	0	0	0	1	1	1	0	1	1	1	0
a	b	y																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	0																	
XNOR			$y = \bar{a} \cdot \bar{b} + a \cdot b$ $y = \overline{a \oplus b}$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	y	0	0	1	0	1	0	1	0	0	1	1	1
a	b	y																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	1																	

Symbol graficzny bramki o większej liczbie wejść przedstawia się poprzez dodanie odpowiedniej liczby linii wejściowych, bez zmiany obrysu graficznego symbolu bramki.

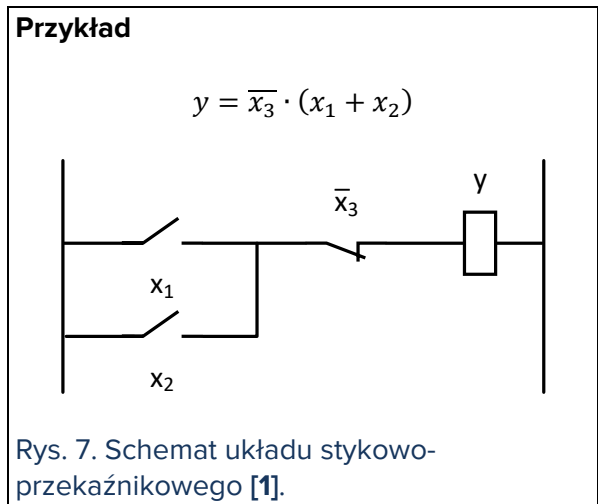


Rys. 6. Symbol graficzny bramki o zwiększonej ilości wejść [4].

Schemat stykowo-przełącznikowy

Funkcje logiczne mogą być również realizowane w technice stykowo-przełącznikowej. W tym przypadku:

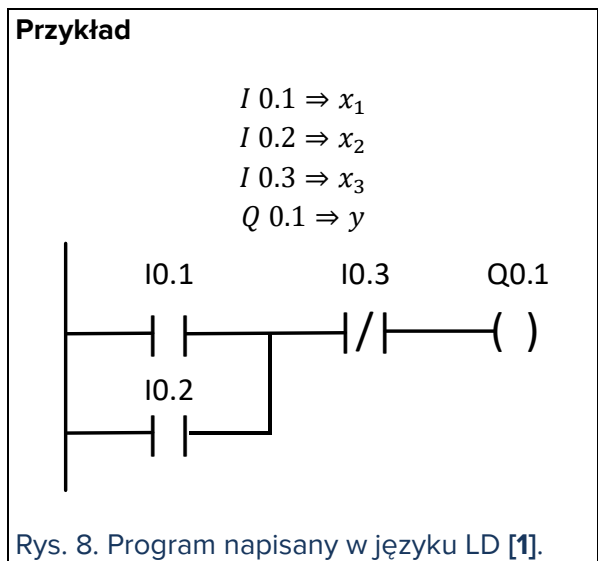
- połączenie szeregowe styków odpowiada operacji **AND**,
- połączenie równoległe styków odpowiada operacji **OR**,
- styk normalnie zamknięty realizuje operację negacji (**NOT**).



Realizacja w sterowniku PLC

Program w języku **LD (Ladder Diagram)** stanowi cyfrowy odpowiednik schematu stykowo-przełącznikowego:

- styki reprezentują sygnały wejściowe,
- cewki reprezentują sygnały wyjściowe,
- połączenia szeregowe i równoległe odwzorowują operacje logiczne AND i OR.



OCENA KOSZTU UKŁADU

Ostatnim etapem projektowania układu kombinacyjnego jest ocena kosztu jego realizacji. Celem tego etapu jest sprawdzenie, czy otrzymany schemat zapewnia korzystną realizację pod względem liczby zastosowanych elementów, ich rodzaju oraz złożoności połączeń.

Koszt układu może być oceniany na podstawie:

- liczby wykorzystanych bramek logicznych,
- liczby wejść zastosowanych bramek,
- liczby elementów dodatkowych (np. inwerterów),
- liczby zastosowanych układów scalonych,
- złożoności połączeń pomiędzy elementami.

Należy pamiętać, że minimalna postać algebraiczna funkcji nie zawsze prowadzi do najtańszej realizacji sprzętowej. Ostateczny koszt zależy również od dostępności i sposobu wykonania elementów logicznych.

Przykładowo, w przypadku realizacji układu w technologii TTL, zastąpienie bramek NOT bramkami NAND może nie przynieść zmniejszenia kosztu układu. Wynika to z faktu, że standardowe układy scalone TTL zawierające dwuwejściowe bramki NAND są często wykonywane jako układy wielobramkowe (np. cztery bramki NAND w jednej obudowie). Wykorzystanie dodatkowej bramki NAND tylko w celu realizacji negacji może więc nie zmniejszyć liczby użytych układów scalonych.

Po przeprowadzeniu analizy kosztu możliwa jest korekta wcześniejszego rozwiązania, np. wybór innej postaci minimalnej funkcji lub zastosowanie innego zestawu bramek, jeżeli prowadzi to do prostszej i tańszej realizacji.

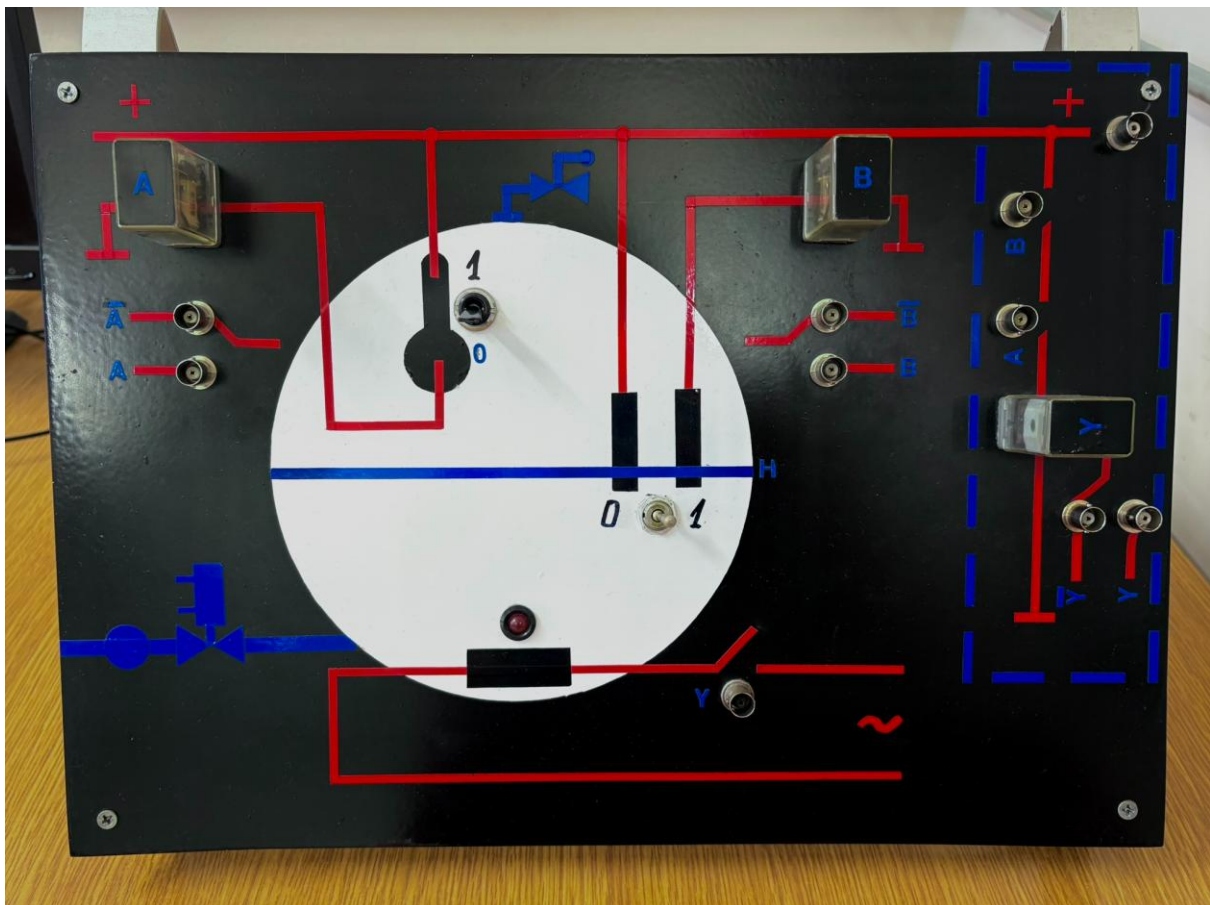
ĆWICZENIE LABORATORYJNE

STANOWISKO LABORATORYJNE

Ćwiczenie A

Walczak jest to element konstrukcyjny kotła parowego w postaci cylindrycznego zbiornika ciśnieniowego, w którym gromadzi się woda oraz wytwarzana para wodna. Pełni funkcję przestrzeni pośredniej pomiędzy układem zasilania wodą a odbiorem pary, umożliwiając oddzielenie fazy ciekłej od gazowej oraz stabilizację procesu parowania. W walczaku utrzymywany jest określony poziom wody, a jego parametry pracy (temperatura) są monitorowane ze względu na bezpieczeństwo eksploatacji urządzenia.

W przypadku modelu laboratoryjnego kotła parowego walczak pełni funkcję zbiornika, w którym podgrzewana jest woda za pomocą grzałki elektrycznej w celu wytworzenia pary, a kontrola poziomu wody i temperatury pozwala zabezpieczyć układ przed pracą w niebezpiecznych warunkach.



Rys. 9. Stanowisko A - model procesu wytwarzania pary (walczaka).

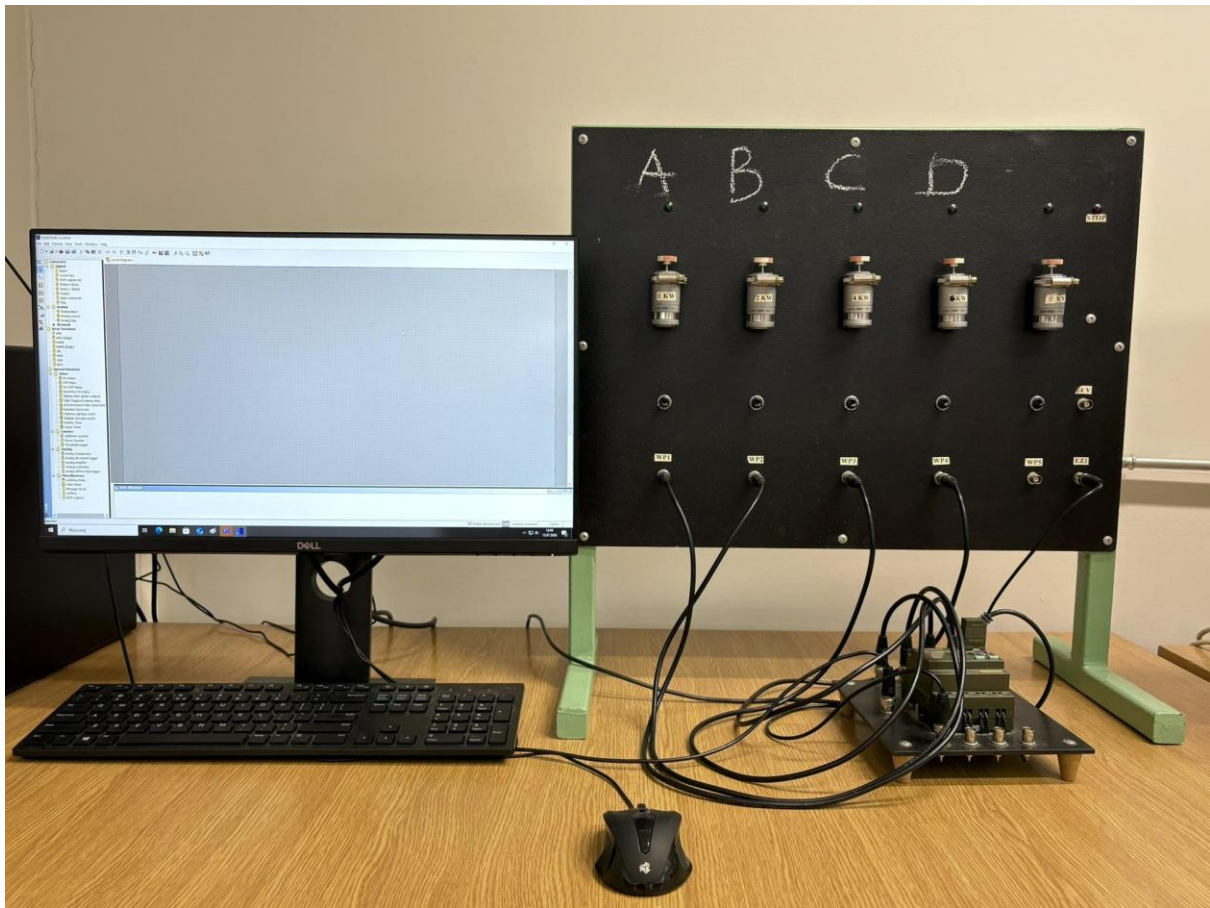
W układzie wykorzystano dwa czujniki działające na różnych zasadach fizycznych, co wpływa na sposób interpretacji ich sygnałów wyjściowych.

- Czujnik temperatury (A) jest elementem progowym, który zmienia stan wyjścia po osiągnięciu przez mierzony parametr określonej wartości granicznej. Oznacza to, że sygnał z czujnika jest zależny od przekroczenia ustalonego poziomu temperatury, a nie od wartości ciągłej mierzonego parametru.
- Czujnik poziomu wody (B) wykorzystuje natomiast właściwości przewodzące wody i działa na zasadzie zamykania obwodu elektrycznego. W momencie, gdy poziom wody osiągnie wysokość elektrody pomiarowej, woda tworzy połączenie elektryczne pomiędzy elektrodą a elementem odniesienia, powodując zmianę stanu sygnału wejściowego. Brak kontaktu elektrody z wodą oznacza natomiast przerwę w obwodzie.

Pomimo zastosowania w modelu takich samych przekaźników stykowych, sygnały z obu czujników należy interpretować odmiennie ze względu na różną zasadę ich działania. Z tego względu przed zaprojektowaniem układu regulacji konieczne jest jednoznaczne określenie znaczenia stanów logicznych przypisanych do poszczególnych wejść.

Ćwiczenie B

Drugie stanowisko przedstawia układ zasilania, w którym zestaw silników elektrycznych pełni funkcję modelowego obciążenia sieci elektroenergetycznej.



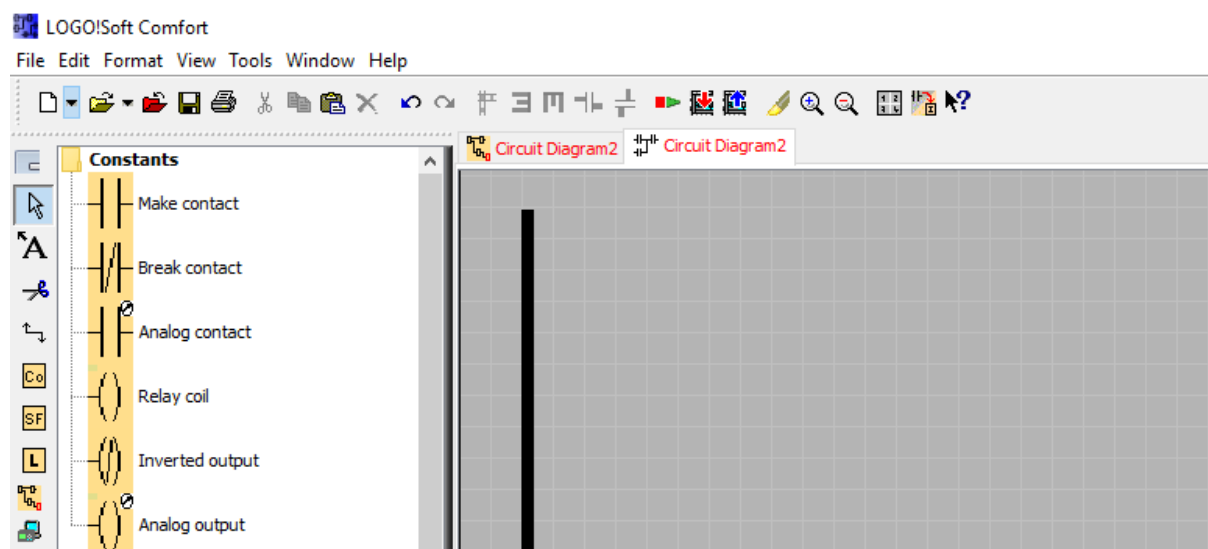
Rys. 10. Stanowisko B - stanowisko z silnikami.

Poszczególne silniki reprezentują odbiorniki o różnych mocach znamionowych, a ich załączanie za pomocą sterownika PLC umożliwia zmianę całkowitego obciążenia układu. Zadaniem zaprogramowanego układu sterowania PLC jest monitorowanie sumarycznego obciążenia oraz odłączenie zasilania odbiorników po przekroczeniu określonej wartości granicznej, symulując działanie zabezpieczenia przeciążeniowego.

W języku LD stosowanym w programie **LOGO!Soft Comfort** podstawowymi elementami są styki wejściowe oraz cewki wyjściowe:

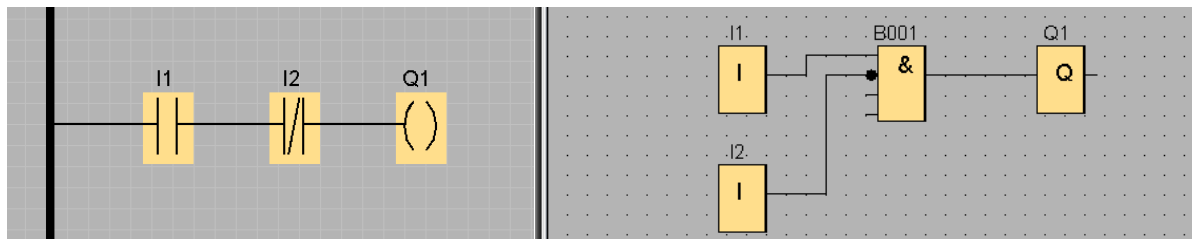
- Styk normalnie otwarty (NO, make contact) - przewodzi sygnał, gdy przypisana zmienna logiczna przyjmuje wartość **1**. Przykład zastosowania: przycisk START.
- Styk normalnie zamknięty (NC, break contact) - przewodzi sygnał, gdy przypisana zmienna logiczna przyjmuje wartość **0**. Przykład zastosowania: przycisk STOP, sygnał zabezpieczenia awaryjnego.
- Cewka (relay coil) jest końcowym elementem szeregu programu i określa stan przypisanego wyjścia sterownika. Jeżeli warunki logiczne po lewej stronie cewki są spełnione, przypisane wyjście przyjmuje wartość logiczną 1.

Elementy programu dodaje się poprzez przeciągnięcie odpowiednich bloków z panelu elementów znajdującego się po lewej stronie okna programu na obszar edycji schematu. Następnie wszystkie elementy należy połączyć zgodnie z projektowaną funkcją logiczną. Zasilanie szeregu programu realizowane jest poprzez podłączenie pierwszego elementu do pionowej, grubej czarnej linii znajdującej się po lewej stronie obszaru roboczego.



Rys. 11. Zrzut ekranu z oknem programu Logo!Soft Comfort.

W programie LOGO!Soft Comfort możliwe jest tworzenie programów sterujących w dwóch podstawowych językach graficznych: LD (Ladder Diagram) oraz FBD (Function Block Diagram). W języku LD program tworzony jest w postaci schematu stykowo-cewkowego, natomiast w FBD funkcja logiczna realizowana jest za pomocą połączonych bloków funkcyjnych.



Rys. 12. Realizacja układu w języku LAD i FBD.

ZADANIA DO WYKONANIA

Ćwiczenie A

Opis słowny: Celem zadania jest zaprojektowanie kombinacyjnego układu sterowania dla modelu kotła parowego, który zapewnia bezpieczną i efektywną pracę grzejnika elektrycznego w zależności od parametrów fizycznych procesu. Walczak pełni funkcję zbiornika pośredniego, w którym woda jest podgrzewana w celu wytworzenia pary. Główne zadania układu sterowania to:

- Utrzymanie bezpiecznego poziomu wody, aby nie dopuścić do odstonięcia i przepalenia grzałek. Uwaga. Elektroda przewodzi prąd wtedy, gdy jest zanurzona.
- Regulacja temperatury (ciśnienia) wody w celu wykrywania warunków mogących prowadzić do nadmiernego wzrostu ciśnienia w walczaku.

Wykonanie ćwiczenia i zakres sprawozdania:

1. Sformułuj założenia projektowe dotyczące sygnałów wejściowych (T, temperatura oraz H, poziom wody) oraz wyjściowych (G, grzanie).

○ T:	stan „0” -	stan „1” -
○ H:	stan „0” -	stan „1” -
○ G:	stan „0” -	stan „1” -
2. Przedstaw tablicę prawdy dla wszystkich kombinacji sygnałów wejściowych (T, H).
3. Przygotuj tablicę Karnaugh’a w celu weryfikacji minimalizacji funkcji.
4. Wyznacz równanie logiczne opisujące sterowanie walczakiem.
5. Opracuj schemat elektroniczny układu stykowo-przełącznikowego.
6. Sprawdź działanie układu we wszystkich kombinacjach.

Ćwiczenie B

Opis słowny: Dla układu zasilania czterech silników elektrycznych o mocach odpowiednio: $x_1 = 1\text{ kW}$, $x_2 = 2\text{ kW}$, $x_3 = 4\text{ kW}$ oraz $x_4 = 6\text{ kW}$, zaprojektuj układ logiczny sterujący bezpiecznikiem y. Bezpiecznik ma odłączać zasilanie, gdy spełniony zostanie poniższy warunek (rozważ dwa oddzielne warianty):

- a) łączna moc załączonych silników przekroczy 7 kW ($P > 7\text{ kW}$),
- b) łączna moc załączonych silników będzie równa lub przekroczy 7 kW ($P \geq 7\text{ kW}$).

Wykonanie ćwiczenia i zakres sprawozdania:

1. Przygotuj założenia działania układu:
Sygnały wejściowe opisujące stan silników, x_1, x_2, x_3, x_4 :
stan „___” = silnik włączony,
stan „___” = silnik wyłączony.

Sygnał wyjściowy:
stan „___” = praca normalna
stan „___” = zadziałanie bezpiecznika (odłączenie zasilania),
2. Przedstaw tablicę prawdy uwzględniającą wszystkie kombinacje stanów wejściowych.
3. Wykonaj minimalizację funkcji logicznej za pomocą tablicy Karnaugh
4. Wyznacz zminimalizowane równanie logiczne w postaci sumy iloczynów.
5. Opracuj schemat elektroniczny układu stykowo-przełącznikowego i program w języku drabinkowym LD (Ladder Diagram) w programie Logo Soft Comfort realizujący zaprojektowaną funkcję sterowania.
6. Zweryfikuj działanie zaprojektowanego układu poprzez sprawdzenie poprawności logicznej dla wybranych stanów obciążenia sieci.

PYTANIA KONTROLNE

1. Czym różni się układ kombinacyjny od sekwencyjnego?
2. Czym różni się układ kombinacyjny od bramek logicznych?
3. Co jest pierwszym etapem syntezy układu kombinacyjnego?
4. Omów kod dwójkowy.
5. Przeanalizuj kolejne kolumny tablicy prawdy i odpowiedz na pytanie: co ile wierszy zmienia się wartość poszczególnych zmiennych wejściowych?
6. Na czym polega minimalizacja funkcji logicznych?
7. Scharakteryzuj bramki logiczne.

LITERATURA

- [1] Mikulczyński T. (red.), Laboratorium podstaw automatyki i automatyzacji, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2005, 235 s. [Dostęp online](#)
- [2] MadForMath, Karnaugh Map Solver, [narzędzie online]. [Dostęp online](#)
- [3] Politechnika Łódzka, [Temat 2 – materiał dydaktyczny], Wydział Fizyki Technicznej, Informatyki i Matematyki Stosowanej, Politechnika Łódzka. [Dostęp online](#)
- [4] Zakład Techniki Cyfrowej ISŁ WEL WAT, LAB1: Synteza układów kombinacyjnych, Laboratorium Układów Cyfrowych, Wojskowa Akademia Techniczna, Warszawa, 2025. [Dostęp online](#)