Politechnika Lubelska

Katedra Automatyki i Metrologii

Laboratorium

Podstawy Automatyki

Inżynierskie Zastosowania Informatyki
w Elektrotechnice

Ćwiczenie nr **2**

Temat:

Wspomagana komputerowo synteza układów przełączających

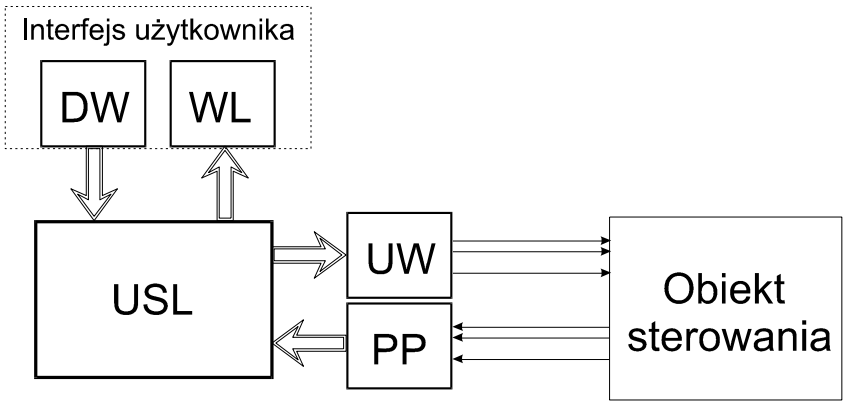
Lublin 2021

2.1. Wstęp

Automatyzację wielu czynności wykonywanych przez urządzenia przemysłowe uzyskuje się za pomocą układów sterowania logicznego (układów przełączających). Układy sterowania logicznego mają szerokie zastosowanie zarówno w automatyzacji pracy pojedynczych maszyn i zespołów (np. windy, robotów, urządzeń transportowych, sygnalizacji świetlnej, sprzętu AGD itp.) jak i w przypadku kompleksowej automatyzacji całych procesów technologicznych.

Układ sterowania logicznego składa się z następujących bloków funkcjonalnych (patrz rys. 2.1):

- zasadniczego układu sterowania (USL), realizującego algorytm sterowania logicznego. Może to być specjalizowane lub uniwersalne urządzenie techniczne operujące dwuwartościowymi (binarnymi) sygnałami, o sprzętowej realizacji algorytmu sterowania (realizacja sztywna - "zadrutowana") lub o realizacji elastycznej - programowej. Ogromną popularnością (ze względu na liczne zalety) cieszą się rozwiązania oparte na mikroprocesorowych sterownikach swobodnie programowalnych PLC.
- układów wykonawczych (UW),
- czujników i przetworników pomiarowych (PP),
- układu wprowadzania danych wejściowych (DW),
- układu sygnalizacji i wyprowadzania danych wyjściowych (WL).



Rys. 2.1. Schemat blokowy układu sterowania logicznego

Ze względu na sposób wypracowywania sygnałów wyjściowych układy przełączające dzieli się na:

- układy kombinacyjne (jednotaktowe),
- układy sekwencyjne (wielotaktowe).

2.2. Układy kombinacyjne

Układ przełączający nazywany jest kombinacyjnym (rys. 2.2), jeżeli każdemu wektorowi sygnałów wejściowych (kombinacji stanów logicznych na wejściach: x_1, x_2, x_3, ..., x_m) przyporządkowany jest jeden i tylko jeden wektor sygnałów wyjściowych (kombinacja stanów logicznych na wyjściach: y_1, y_2, y_3, ..., y_n). W układach kombinacyjnych istnieje, więc jednoznaczna zależność między zbiorem wejść i wyjść, niezależnie od czasu.

Rys. 2.2. Schemat blokowy układu kombinacyjnego

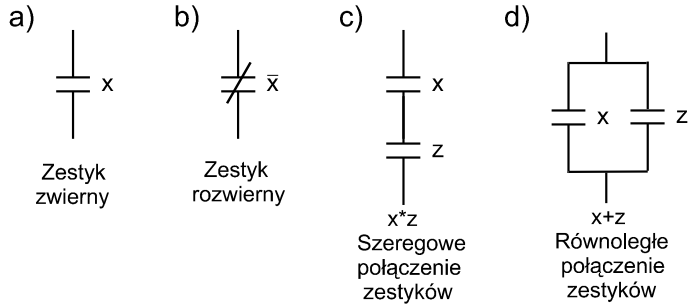
2.2.1. Struktura układu przełączającego

Teoria struktur układów przełączających opiera się na wybranych działach logiki matematycznej takich jak: rachunek zdań, rachunek zbiorów, dwuelementowa algebra Boole'a. Wykorzystywana jest tutaj tzw. logika dwuwartościowa, w której zmienne mogą przyjmować tylko dwie wartości. Oznaczone są one zwykle przez "1" i "0". Może to być zdanie wyrażające prawdę (1) lub fałsz (0). Elementy układów przełączających są elementami dwustanowymi. Każdy z elementów może znajdować się w stanie działania (1) lub niedziałania (0).

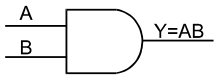
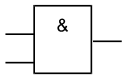
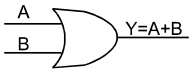
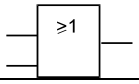
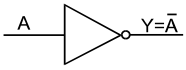
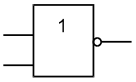
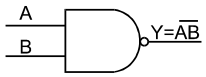
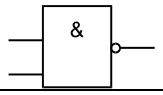
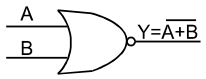
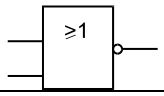
Struktura wewnętrzna każdego układu przełączającego może być przedstawiona analitycznie w postaci wyrażenia strukturalnego, przypominającego wyrażenie algebraiczne i przedstawiającego określoną dla danego układu funkcję logiczną. W przypadku układu kombinacyjnego jest to rodzina funkcji przełączających (tzw. funkcji wyjścia) zapisywanych w postaci:

$$Y_i = f_i(x_1, x_2, x_3, ..., x_m) \text{ dla i = 1,2,...n} \tag{2.1}$$

Struktura wewnętrzna układu przełączającego może być przedstawiona również graficznie - w postaci schematu opartego na elementach stykowych bądź bezstykowych. Podstawowe elementy schematu stykowego zostały przedstawione na rysunku 2.3, natomiast bezstykowe na rysunku 2.4.



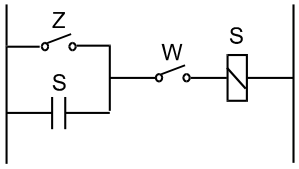
Rys. 2.3. Oznaczenia elementów stykowych (zestyków)

Operacja logiczna	Symbol 1	Symbol 2
AND (I)		
OR (LUB)		
NOT (NIE)		
NAND (NIE-I)		
NOR (NIE-LUB)		

Rys. 2.4. Oznaczenia graficzne podstawowych funkcji logicznych

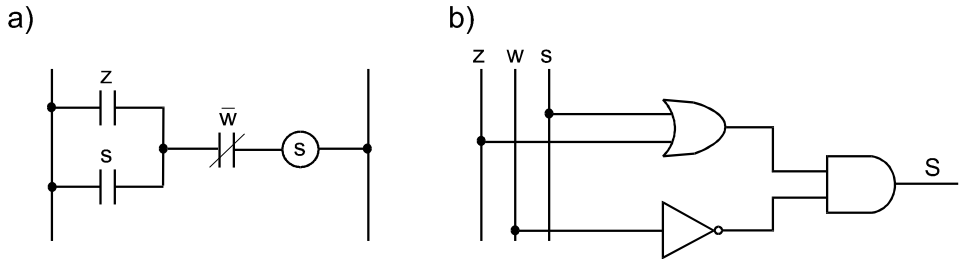
Na rys. 2.5 przedstawiono przykład układu uruchamiania stycznika S za pomocą przycisku załączającego Z. Stycznik jest wyłączany przez naciśnięci przycisku wyłączającego W. Układ taki może być opisany następującą funkcją przełączającą:

$$S = (s + z) \cdot \overline{w} \tag{2.2}$$



Rys. 2.5. Schemat elektryczny układu sterowania stycznika

Graficzne reprezentacje powyższego układu przełączającego przedstawia rysunek 2.6.

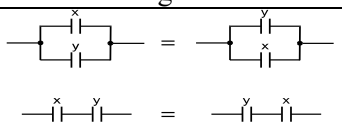
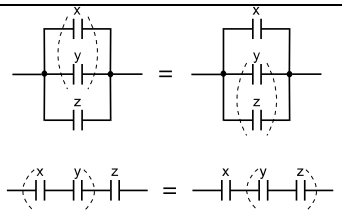
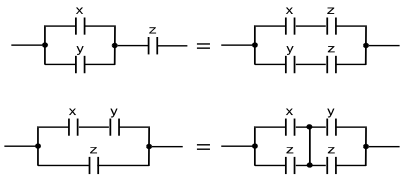
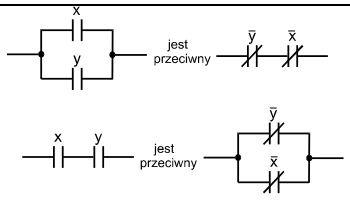


Rys. 2.6. Struktura układu przełączającego w postaci schematu opartego na elementach: a) stykowych, b) bezstykowych

2.2.2.Prawa algebry układów przełączających

Spośród wielu praw algebry Boole'a podstawowe znaczenie w zastosowaniu do teorii struktur układów przełączających mają następujące cztery prawa: przemienności, łączności, rozdzielności i De' Morgana. Przedstawienie algebraiczne i graficzne poszczególnych praw prezentuje tablica 2.1.

Tablica 2.1. Podstawowe prawa algebry Boole'a

Nazwa prawa	Postać algebraiczna	Postać graficzna
Prawo przemienności	$x + y = y + x$ $xy = yx$	
Prawo łączności	$(x + y) + z = x + (y + z)$ $(xy)z = x(yz)$	
Prawo rozdzielczości: a) mnożenia względem dodawania b) dodawania względem mnożenia	$(x + y)z = xz + yz$ $xy + z = (x + z)(y + z)$	
Prawo De' Morgana	$\overline{x + y} = \bar{y} \cdot \bar{x}$ $\overline{xy} = \bar{y} + \bar{x}$	

W teorii układów przełączających obwód lub element obwodu otwarty oznacza się zerem (**0**) a jedynką (**1**) obwód lub element obwodu zamkniętego. Na przykład szeregowe połączenie zwiernych i rozwiernych zestyków tego samego przekaźnika zawsze przerywa obwód:

$$x \cdot \bar{x} = 0 \tag{2.3}$$

Równoległe połączenie zwiernych i rozwiernych zestyków daje element schematu stale zamknięty:

$$x + \bar{x} = 1 \tag{2.4}$$

Przy szeregowym lub równoległym połączeniu kilku jednakowych zestyków układ działa tak samo jak w przypadku jednego zestyku:

$$x \cdot x \cdot x \cdot \ldots = x \tag{2.5}$$

$$x + x + x + \ldots = x \tag{2.6}$$

Dodanie do jakiegoś wyrażenia strukturalnego zera lub pomnożenie go przez jedynkę nie zmienia wartości logicznej tego wyrażenia, czyli :

$$x + 0 = x; \; x \cdot 1 = x \tag{2.7}$$

Wartość logiczna wyrażenia zmienia się w przypadku dodania do niego jedynki lub pomnożenia go przez zero, czyli:

$$x + 1 = 1; \; x \cdot 0 = 0 \tag{2.8}$$

2.2.3.Kanoniczne postacie sumy oraz iloczynu

W przypadku prostych zadań sterowania binarnego, matematyczną postać funkcji opisującej układ przełączający można napisać wprost na podstawie słownego opisu działania. W układach bardziej złożonych buduje się tablicę stanów, określającą stan elementów wyjściowych w zależności od stanów elementów wejściowych. Każdemu elementowi wejściowemu oraz wyjściowemu odpowiada jedna kolumna tej tablicy, a każdemu stanowi układu jeden wiersz. Liczba wierszy odpowiada liczbie wszystkich możliwych kombinacji stanów i dla *n*-wejść wynosi *2do potęgi n*. Przykład tablicy stanów dla 3 wejść pokazano na rys 2.7.

Stany wejść			Stan wyjścia
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	-
1	1	1	0

Rys. 2.7. Przykład tablicy stanów (tablicy prawdy) dla układu przełączającego z 3-ma wejściami i 1-nym wyjściem

Śledząc algorytm pracy projektowanego układu kombinacyjnego przypisuje się poszczególnym wierszom odpowiednie wartości wyjść (1 lub 0). Jednak może się zdarzyć, że nie dla wszystkich kombinacji sygnałów wejściowych stan wyjść jest określony lub pewne kombinacje z zasady działania układu nie mogą zaistnieć np. jednoczesne włączenie przesuwu w prawo i w lewo, itp. W takim przypadku stan wyjścia określa się mianem obojętnego i oznacza symbolem "Ɵ" lub "-".

Na podstawie wypełnionej tablicy stanów tworzone są wyrażenia strukturalne dla wyjść, które mogą się składać z:

- sumy iloczynów wejść tych wierszy, dla których sygnał wyjściowy przyjmuje wartość równą 1 (symbol sygnału wejściowego pisany jest bez negacji jeżeli przyjmuje on wartość 1 - z negacją, je-

żeli 0). Tak uzyskana postać funkcji logicznej nazywa jest kanoniczną postacią sumy (układ realizowany jest na podstawie warunków działania). Dla przykładu tablicy z rys. 2.7 wyrażenie strukturalne (funkcja logiczna) przyjmie postać: $Y = \bar{a}\bar{b}c + \bar{a}b\bar{c} + \bar{a}bc + a\bar{b}c$ (2.9)

- iloczynów sum wejść tych wierszy, dla których sygnał wyjściowy przyjmuje wartość równą 0 (symbol sygnału wejściowego pisany jest bez negacji jeżeli przyjmuje wartość 0 - z negacją, jeżeli 1). Tak uzyskana postać funkcji logicznej nazywana jest kanoniczną postacią iloczynu (układ realizowany jest na podstawie warunków niedziałania). Dla przykładu tablicy stanów z rys. 2.7 funkcja przyjmie następującą postać: $Y = (a+b+c)(\bar{a}+b+\bar{c})(\bar{a}+\bar{b}+\bar{c})$ (2.10)

Obie postacie są sobie równoważne pod względem logicznym, prowadzić mogą jednak do zróżnicowanych realizacji technicznych układu sterowania binarnego.

2.2.4.Minimalizacja funkcji logicznych

Dowolny kombinacyjny układ przełączający może być realizowany na wiele różnych sposobów. Zawsze dąży się jednak do tego by otrzymane w wyniku syntezy rozwiązanie było optymalne ze względu na koszt realizacji przy założonej niezawodności układu. Najczęściej uzyskuje się to przez minimalizację liczby elementów z zadanego zestawu, minimalizację liczby połączeń itp.

Przedstawiona w postaci kanonicznej funkcja opisująca działanie układu kombinacyjnego może być bezpośrednio zrealizowana na podstawie tej postaci. Analizując jednakże wyrażenia 2.9 i 2.10 łatwo zauważyć, że argumenty (sygnały wejściowe) występują wielokrotnie (w postaci negacji lub afirmacji) w różnych czynnikach lub składnikach. Stosując prawa algebry Boole'a postać kanoniczna funkcji może zostać zminimalizowana, tj. przekształcona do postaci, w której występuje mniejsza liczba czynników (składników) oraz wyeliminowano nadmiarowe sygnały wejściowe. Proces poszukiwania takiej postaci funkcji nazywa się minimalizacją. Przy minimalizacji wykorzystuje się zasadę sklejania:

$$x_1 \cdot x_2 + x_1 \cdot \bar{x}_2 = x_1 \quad (2.11)$$

$$(x_1 + x_2)(x_1 + \bar{x}_2) = x_1 \quad (2.12)$$

Zasada sklejania ma zastosowanie w przypadku, gdy dwa składniki (2.11) lub dwa czynniki (2.12), są "sąsiednimi", tzn. jeżeli różnią się znakiem negacji tylko na jednej pozycji.

Minimalizacja funkcji polegająca na wyszukiwaniu wyrażeń sąsiednich i stosowaniu zależności (2.11 lub 2.12) dla dużej liczby wejść jest bardzo uciążliwa. Znaczne usprawnienie minimalizacji uzyskuje się stosując jedną z wykorzystywanych w praktyce metod tablicowych: Karnaugha (czytaj Karno), lub Quine'a – Mc' Cluske'a.

W tablicy Karnaugha, stany układu przełączającego reprezentowane są przez jej kratki, przy czym tablica przy "n" stanach zawiera 2^n kratek. Każda kratka tablicy odpowiada jednej kombinacji wejść. Kod zmiennych wejściowych jest tak dobrany (kod Gray'a), żeby sąsiednie kratki różniły się wartością tylko jednej zmiennej tzn., aby możliwe było "sklejanie" wyrażeń logicznych opisanych przez kratki leżące obok siebie. Do tak opisanej w/w kodem tablicy w odpowiednie kratki wpisuje się symbole (1, 0, -), odpowiadające wartościom funkcji logicznej dla kombinacji zmiennych wejściowych przypisanych kratkom. Jeżeli w dwóch sąsiednich kratkach znajdują się wartości (0 i - lub 1 i -), to odpowiadające tym kratkom wyrażenia logiczne można skleić, co sprowadza się do wyeliminowania sygnału wejściowego z czynnika (składnika), który w ramach sklejanej grupy zmienia wartość logiczną. Zasadę minimalizacji funkcji logicznych metodą tablicy Karnaugha pokazuje rys. 2.8.

x3, x4, x5 / x1, x2	000	001	011	010	110	111	101	100
00	0	0	1	0	0	-	0	1
01	0	0	0	0	0	0	1	1
11	1	-	1	1	0	0	0	1
10	-	1	1	1	0	1	0	-

$\bar{x}_1\bar{x}_3 \quad \bar{x}_2x_4x_5 \quad \bar{x}_1x_2\bar{x}_3\bar{x}_4 \quad \bar{x}_3\bar{x}_4\bar{x}_5$

Rys. 2.8. Tablica Karnougha dla przykładowej funkcji 5-ciu zmiennych

Jeżeli zostanie wzięty pod uwagę obszar (grupa) (patrz rys. 2.8), składająca się z kratek, dla których $\bar{x}_1 x_2 x_3 \bar{x}_4 x_5$ oraz $\bar{x}_1 x_2 x_3 \bar{x}_4 \bar{x}_5$ jak widać tylko element x_5 zmienia swoją wartość logiczną. W wyniku sklejenia otrzymuje się zamiast dwóch poprzednich składników - jedno wyrażenie postaci $\bar{x}_1 x_2 x_3 \bar{x}_4$.

Minimalizacja funkcji logicznej metodą tablic Karnaugha powinna przebiegać w następujących etapach:

1. Należy podjąć decyzję czy układ będzie realizowany dla warunków działania (wtedy wybiera się grupy jedynek i niektórych stanów obojętnych) czy też dla warunków niedziałania (wybór grup zer i niektórych stanów obojętnych).
2. Wśród wybranych 0 lub 1 poszukuje się możliwości utworzenia największych grup. Jeżeli wybrana zostanie grupa dwu-kratkowa to z wyrażenia zostanie usunięte jedno wejścia (litera, zmienna wejściowa), W przypadku np. grupy ośmioro-kratkowej - cztery sygnały wejściowe itp. Im większa jest grupa połączonych kratek, tym lepszy jest efekt minimalizacji. Grupy mogą być 2^k-kratkowe, k=1, 2, 3, ... W celu wyeliminowania niepożądanego zjawiska **hazardu** grupy należy dobierać tak aby maksymalnie zachodziły na siebie. W łączonych grupach można dowolnie wykorzystywać stany obojętne.
3. Wyodrębnione w tablicy grupy opisuje się wyrażeniami strukturalnymi, będącymi albo normalną postacią sumy lub iloczynu.

2.2.5.Przykład syntezy kombinacyjnego układu przełączającego

Punktem wyjścia do syntezy (projektu) układu przełączającego jest najczęściej słowne (lub inne również mało ścisłe) sformułowanie jego logiki działania. Pomijając techniczną realizację systemu sterowania celem projektu będzie symboliczny zapis logicznych powiązań wejść z wyjściami obiektu, realizowany w sterowniku (automacie), którego modelem jest układ przełączający.

Np. postawiono sobie zadanie zaprojektowania układu sterowania binarnego równoległą pracą dwóch pomp o różnej wydajności. Pompy powinny dopełniać cieczą dwa zbiorniki, opróżniające się w nieprzewidywalny sposób wg następujących zasad:

- jeżeli poziom cieczy w jednym zbiorniku wynosi powyżej połowy a drugi zbiornik jest pełny, to powinna pracować pompa I,
- jeżeli oba zbiorniki są zapełnione powyżej połowy lub jeden mniej niż do połowy, to powinna pracować pompa II,
- jeżeli zapełnienia obu zbiorników spadną poniżej połowy, obie pompy powinny pracować.

Rozwiązanie:

Zakłada się, że czujniki poziomu zapełnienia generują sygnały logiczne 1 jeżeli przekroczone zostaną odpowiednie poziomy.

Po dokładnej analizie zadań stojących przed układem sterowania wydziela się zmienne (wejściowe i wyjściowe) i tworzy ich zestawienie z przypisaniem oznaczeń i komentarzy (tzw. tablica zmiennych).

Zbiornik I napełniony powyżej połowy		- x_1 (wejście)
Zbiornik I pełny		- x_2 (wejście)
Zbiornik II napełniony powyżej połowy		- x_3 (wejście)
Zbiornik I pełny		- x_4 (wejście)
Stan pracy pompy I	1-włączona, 0-wyłączona	- y_1 (wyjście)
Stan pracy pompy II	1-włączona, 0-wyłączona	- y_2 (wyjście)

Buduje się tablicę stanów (prawdy). Zawiera ona tyle wierszy ile kombinacji mogą mieć wejścia. Kombinacje logicznych wartości sygnałów wejściowych porządkuje się zgodnie z kodem naturalnym binarnym. W tablicy poszczególnym stanom wejść przypisuje się odpowiednie (zgodne z funkcją jaką ma spełniać układ przełączający) stany wyjść. „Nierealnym" kombinacjom wejść przypisuje się obojętne stany wyjść.

x1	x2	x3	x4	y1	y2

0	0	0	0	1	1
0	0	0	1	-	-
0	0	1	0	0	1
0	0	1	1	0	1
0	1	0	0	-	-
0	1	0	1	-	-
0	1	1	0	-	-
0	1	1	1	-	-
1	0	0	0	0	1
1	0	0	1	-	-
1	0	1	0	0	1
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	1	-	-
1	1	1	0	1	0
1	1	1	1	1	0

Rys. 2.8. Tablica stanów układu sterowania pompami

a)

x_1x_2 \ x_3x_4	00	01	11	10
00	**1**	-	0	0
01	-	-	-	-
11	0	-	**1**	**1**
10	0	-	**1**	0

b)

x_1x_2 \ x_3x_4	00	01	11	10
00	1	-	1	1
01	-	-	-	-
11	1	-	**0**	**0**
10	1	-	**0**	1

Rys. 2.9. Tablice Karnaugha dla sygnałów wyjściowych: a) dla y_1 , b) dla y_2

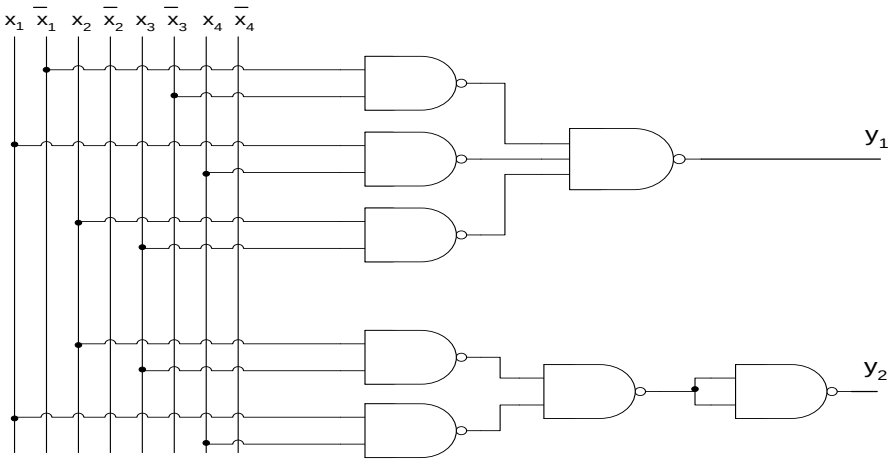
Funkcje logiczne dla poszczególnych wyjść (normalna postać sumy i iloczynu):

$$y_1 = \bar{x}_1\bar{x}_3 + x_1x_4 + x_2x_3$$
$$y_2 = (\bar{x}_2 + \bar{x}_3)(\bar{x}_1 + \bar{x}_4) \qquad (2.14)$$

Przekształcając powyższe funkcje (podwójna negacja) i stosując jedno z praw De' Morgana otrzymuje się funkcje dogodne do realizacji układu na bramkach NAND:

$$y_1 = \overline{\overline{\bar{x}_1\bar{x}_3 + x_1x_4 + x_2x_3}} = \overline{\overline{\bar{x}_1\bar{x}_3} \cdot \overline{x_1x_4} \cdot \overline{x_2x_3}}$$
$$y_2 = \overline{\overline{(\bar{x}_2 + \bar{x}_3)}\,\overline{(\bar{x}_1 + \bar{x}_4)}} = \overline{(x_2 \cdot x_3)}\,\overline{(x_1 \cdot x_4)} \qquad (2.15)$$

Ostatnim etapem syntezy jest schemat logiczny układu sterowania, którego postać graficzną przedstawiono na rys. 2.10.



Rys. 2.10. Schemat logiczny realizujący zadania sterujące pracą pomp

2.3.Układy sekwencyjne

Układami sekwencyjnymi nazywane są układy dyskretne, w których stan elementów wyjściowych jest funkcją nie tylko stanu elementów wejściowych, ale również funkcją poprzednich stanów układu. Oznacza to, iż zależność pomiędzy stanami wejść ***X*** i wyjść ***Y*** nie jest jednoznaczna:

$$Y_i^t = f_i(X^t, X^{t-1}, X^{t-2}, ...) \text{ dla i = 1,2,...n} \qquad (2.16)$$

Określonej kombinacji stanów sygnałów wejściowych (wektora wejściowego) mogą odpowiadać różne kombinacje stanów sygnałów wyjściowych (wektora wyjściowego), zależnie od stanu w jakim układ znajdował się poprzednio. Zależność aktualnego stanu od układu od jego stanu w chwili (chwilach) poprzedniej jest realizowana z pomocą elementów pamięci *Q*. Stan elementów pamięci jest nazywany stanem wewnętrznym układu przełączającego.

Poszczególne elementy układów sekwencyjnych pracują w określonej kolejności. Każdy okres, podczas którego w układzie nie zachodzi żadna zmiana jest nazywany taktem.

2.3.1.Struktura układu sekwencyjnego

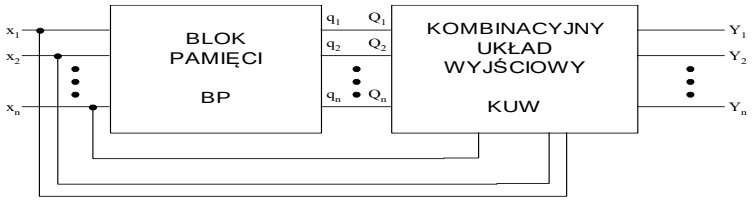
Struktura układów sekwencyjnych może być realizowana w postaci: tzw. automatu Mealy'ego (rys. 2.11), w którym sygnały wyjściowe Y_i zależą od sygnałów elementów pamięci i od niektórych sygnałów wejściowych X_i:

$$\boldsymbol{Y}_i^t = \boldsymbol{\lambda}_i(\boldsymbol{Q}^t, \boldsymbol{X}^t) \qquad (2.17)$$

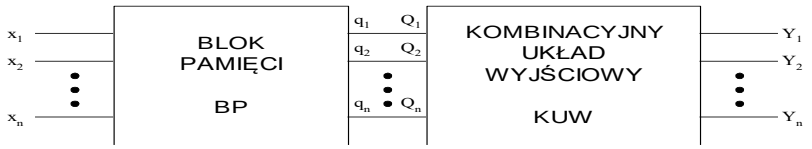
lub tzw. automatu Moore'a (rys. 2.12), w którym sygnały wyjściowe zależą tylko od sygnałów elementów pamięci:

$$\boldsymbol{Y}_i^t = \boldsymbol{\lambda}_i(\boldsymbol{Q}^t) \qquad (2.18)$$

Zależności (2.17) i (2.18) nazywane są funkcjami wyjść układu przełączającego.



Rys. 2.11. Schemat blokowy układu sekwencyjnego realizowanego w postaci automatu Mealy'ego



Rys. 2.12. Schemat blokowy układu sekwencyjnego realizowanego w postaci automatu Moore'a

W obydwu realizacjach strukturę bloku pamięci określają tzw. funkcje przejść. Umożliwiają one wyznaczenie następnego stanu wewnętrznego układu $\boldsymbol{Q}^{t+1}$ na podstawie aktualnego stanu wejść $\boldsymbol{X}^t$ i aktualnego stanu wewnętrznego $\boldsymbol{Q}^t$. Funkcje przejść można zapisać w postaci:

$$\boldsymbol{Q}^{t+1}=\boldsymbol{\delta}(\boldsymbol{Q}^t, \boldsymbol{X}^t) \tag{2.19}$$

Układy sekwencyjne mogą być realizowane jako asynchroniczne lub synchroniczne. W układach asynchronicznych zmiana stanu wewnętrznego może mieć miejsce w dowolnej chwili czasowej wyznaczonej przez zmianę jednego z sygnałów wejściowych. Układ synchroniczny przechodzi do nowego stanu (pod warunkiem zmiany jednego z sygnałów wejściowych) w dyskretnych chwilach czasu wyznaczonych przez sygnał taktujący (synchronizujący). Prędkość pracy automatu asynchronicznego jest określana jedynie wewnętrznymi właściwościami urządzenia, natomiast automatu synchronicznego – sygnałem taktującym.

Każdy układ sekwencyjny może w danej chwili znajdować się w jednym z dwu wewnętrznych stanów pracy: stabilnym (trwałym) lub niestabilnym (nietrwałym). W stanie stabilnym wyjścia wszystkich elementów systemu posiadają już wartości wynikające z ich sygnałów wejściowych i realizowanej funkcji tzn., że przebiegi przejściowe wywołane zmianą wektora wejściowego dobiegły końca. Przejście między dwoma stanami stabilnymi (wywołane zmianą wejść) odbywa się poprzez stany niestabilne. Istnienie stanu niestabilnego jest następstwem obecności opóźnień wnoszonych przez elementy składowe układu.

Zaprojektowany poprawnie pod względem logicznym układ, w rzeczywistości może pracować nieodpowiednio na skutek niedokładności swoich elementów. Spowodowane to jest zjawiskiem hazardu tzn. różnym czasem przebiegu sygnału po drogach równoległych. W układach sekwencyjnych zjawisko hazardu jest bardzo groźne, gdyż elementy pamięci mogą utrwalić przekłamane sygnały spowodowane hazardem. Hazard usuwany jest przez dodawanie tzw. grup antyhazardowych.

Gdy zmiana stanu układu sekwencyjnego wymaga równoczesnej zmiany stanu pracy dwóch lub więcej elementów pamięci pojawia się zjawisko nazywane wyścigiem. Równoczesność zmian stanu dwóch elementów jest praktycznie niemożliwa co w konsekwencji powoduje , że układ w „poszukiwaniu" stanu stabilnego „przechodzi różne drogi". Jeżeli układ w każdym przypadku osiąga ten sam stan stabilny, to taki wyścig nazywany jest wyścigiem niekrytycznym. Jeżeli układ osiąga inny stan stabilny, to taki wyścig jest nazywany wyścigiem krytycznym. W celu uniknięcia zjawiska wyścigu, układy sekwencyjne projektuje się w sposób wykluczający możliwość równoczesnej zmiany w jednym takcie stanu dwóch i więcej elementów.

2.3.2.Synteza układów sekwencyjnych

Syntezę układu sekwencyjnego rozpoczyna się od sporządzenia opisu jego działania w postaci: opisu słownego, przebiegów czasowych lub grafu (wykresu) przejść.

Następnie w zależności od przyjętej metody syntezy warunki pracy układu przedstawiane są w postaci:

- pierwotnej tablicy programu (metoda tablic programu – Huffmana),
- tablicy kolejności łączeń (metoda tablicy kolejności łączeń).

Blok pamięci układu sekwencyjnego może być realizowany w postaci układu bramek logicznych (występowanie charakterystycznych sprzężeń zwrotnych) lub w postaci układu przerzutników bistabilnych (przerzutniki J-K, S-R, D, T).

2.3.2.1.Synteza układu sekwencyjnego metodą tablic programu

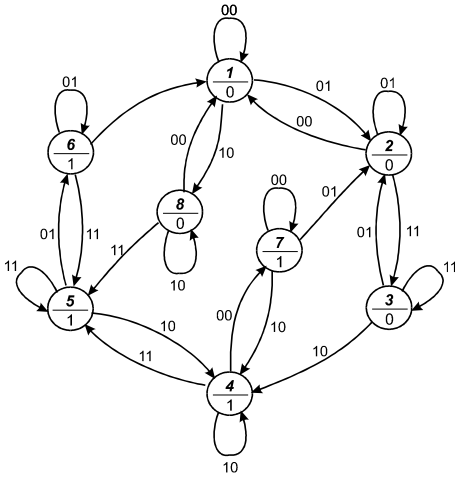
Synteza układów sekwencyjnych metodą tablic programu (Huffmana) przebiega w następujących etapach:

- sporządzenie tablicy stanów stabilnych (pierwotna tablica programu),
- uzupełnienie pierwotnej tablicy programu stanami niestabilnymi (kompletna tablica programu),
- redukcja kompletnej tablicy programu (zredukowana tablica programu),
- sporządzenie tablicy przejść (siatki przejść)
- sporządzenie tablicy stanów elementów pamięci
- sporządzenie tablicy stanów elementów wyjściowych
- określenie funkcji logicznych realizowanych przez elementy pamięci i wyjść

Przykład 1

Dokonać syntezy układu o następujących własnościach. Układ posiada dwa wejścia x_1, x_2 oraz jedno wyjście Z. Jeżeli sygnał na wejściu x_2 się zmieni, to wtedy i tylko wtedy na wyjściu układu powinien pojawić się sygnał taki jak na wejściu x_1. Jeżeli zmieni się sygnał na wejściu x_1 sygnał wyjściowy nie powinien się zmienić. Oba sygnały na wejściach nie mogą zmieniać się równocześnie.

Sporządzenie pierwotnej tablicy programu bezpośrednio na podstawie opisu słownego działania układu często nie jest zadaniem łatwym. Wtedy stosuje się środki pomocnicze, do których należy graf przejść projektowanego układu. Na grafie przejść (patrz rys. 2.13) przedstawione są wszystkie stany wewnętrzne stabilne układu („kółka" z kolejnym cyframi dziesiętnymi – numerami stanu) i możliwe przejścia między nimi (gałęzie z przypisanymi im wektorami wejściowymi wymuszającymi te przejścia). Każdy stan stabilny jest opisany odpowiadającym mu wektorem wyjściowym (wartość binarna pod numerem stanu).



Rys. 2.13. Graf przejść dla układu z przykładu 1

	x_1x_2 00	x_1x_2 01	x_1x_2 11	x_1x_2 10	Z
1	(1)				0
2		(2)			0
3			(3)		0
4				(4)	1
5			(5)		1
6		(6)			1
7	(7)				1
8				(8)	0

Rys. 2.14. Tablica programu ze stanami stabilnymi do przykładu 1

	x_1x_2 00	x_1x_2 01	x_1x_2 11	x_1x_2 10	Z
1	(1)	2	—	8	0
2	1	(2)	3	—	0
3	—	2	(3)	4	0
4	7	—	5	(4)	1
5	—	6	(5)	4	1
6	1	(6)	5	—	1
7	(7)	2	—	4	1
8	1	—	5	(8)	0

s. 2.15. Kompletna tablica programu do przykładu 1

Na podstawie grafu przejść można w łatwy sposób określić pierwotną tablicę programu (rys. 2.14). Pierwotna tablica programu zawiera tyle kolumn ile różnych wektorów wejściowych może w układzie wystąpić oraz dodatkową kolumnę dla wektora sygnałów wyjściowych. Liczba wierszy zaś odpowiada liczbie stanów stabilnych układu. Każdemu stanowi stabilnemu przypisany jest odpowiadający mu wektor wyjściowy.

W celu otrzymania kompletnej tablicy programu (rys. 2.15) pierwotną tablicę programu należy uzupełnić o stany niestabilne. Jeżeli np. układ znajduje się w stanie (1) i sygnał wejściowy x_1x_2 ulegnie zmianie 00→01 to sygnał wyjściowy nie ulegnie zmianie (Z = 0) i układ powinien przejść do stanu (2). W kratkę leżącą na przecięciu pierwszego wiersz i drugiej kolumny wstawiany jest indeks 2 (stan przejściowy – niestabilny). Jeżeli układ znajduje się w

stanie 1 to sygnał wejściowy x_1x_2 nie może ulec zmianie 00 11. W kratkę leżącą na przecięciu pierwszego wiersz i trzeciej kolumny wstawiana jest kreska - (stan zabroniony) itd.

Ponieważ liczba stanów wewnętrznych układu (liczba wierszy w pierwotnej tablicy programu) określa ilość przerzutników niezbędnych do realizacji bloku pamięci, kompletna tablica programu powinna zostać zredukowana (zminimalizowana). Redukcję przeprowadza się w dwóch krokach:

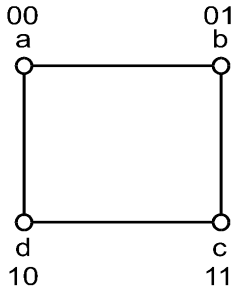
- redukcja stanów wewnętrznych równoważnych. Stany stabilne m i n są równoważne, jeżeli znajdują się w tej samej kolumnie (te same wektory wejściowe), mają jednakowe lub nie sprzeczne (np. 01 i 0-) stany sygnałów wyjściowych Z oraz mają takie same lub niesprzeczne przejścia dla dowolnej sekwencji sygnałów wejściowych (w każdej kolumnie jednakowe (lub nie sprzeczne) numery stanów niestabilnych). Jeżeli dwa stany wewnętrzne są równoważne to jeden z odpowiadających im wierszy (zwykle z większym numerem stanu stabilnego) można w tablicy skreślić.
- redukcja wierszy. Można łączyć również ze sobą takie wiersze, w których stany stabilne znajdują się w różnych kolumnach, o ile tylko inne stany w odpowiednich kolumnach tych wierszy nie są sprzeczne. Wtedy w wierszu otrzymanym w wyniku redukcji wystąpi kilka stanów stabilnych. Nie musi być tutaj przestrzegana zasada niesprzeczności stanów sygnałów wyjściowych. Jeżeli stan sygnałów wyjściowych łączonych wierszy jest taki sam (lub nie sprzeczny), to zredukowanej tablicy programu odpowiada jednokolumnowa tablica wyjść (uzyskanie automatu Moore'a). Jeżeli stan sygnałów wyjściowych łączonych wierszy jest różny to zredukowanej tablicy programu nie odpowiada już jednokolumnowa tablica wyjść (zależność od niektórych sygnałów wejściowych) i w rezultacie uzyskuje się automat Mealy'ego).

Kompletna tablica programu z przykładu 1 (rys. 2.15) nie zawiera stanów wewnętrznych równoważnych. Można więc przejść do drugiego etapu redukcji wierszy. Redukcję wierszy można przeprowadzić dwoma sposobami:

1	2
wiersz 1 z wierszem 8	wiersz 1 z wierszem 2
wiersz 2 z wierszem 3	wiersz 3 z wierszem 7
wiersz 4 z wierszem 7	wiersz 4 z wierszem 5
wiersz 5 z wierszem 6	wiersz 6 z wierszem 8

Redukcja pierwszym sposobem prowadzi do uzyskania automatu Moore'a, natomiast w drugim przypadku dochodzi się do automatu Mealy'ego (sprzeczność wyjść przy łączeniu wierszy: 3 z 7 i 6 z 8). Przedstawioną na rys. 2.16 zredukowaną tablicę programu otrzymano przeprowadzając redukcję sposobem 1.

	x_1x_2 00	x_1x_2 01	x_1x_2 11	x_1x_2 10	Z
a: 1,8	(1)	2	5	(8)	0
b: 2,3	1	(2)	(3)	4	0
c: 4,7	(7)	2	5	(4)	1
d: 5,6	1	(6)	(5)	4	1



q_1q_2	x_1x_2 00	x_1x_2 01	x_1x_2 11	x_1x_2 10
a: 00	00	01	10	00
b: 01	00	01	01	11
c: 11	11	01	10	11
d: 10	00	10	10	11

Q_1Q_2

Rys. 2.16. Zredukowana tablica pro- *Rys. 2.17.Wykres przejść dla przykł. 1* *Rys.2.18. Tablica przejść dla przykł. 1*
gramu dla przykładu 1

W celu uzyskania tablicy przejść należy wykonać kodowanie stanów wewnętrznych (wierszy zredukowanej tablicy układu) stanami elementów pamięci. Minimalną liczbę elementów pamięci potrzebnych do realizacji układu określa liczba wierszy zredukowanej tablicy programu.

W wyniku kodowania wszystkim stanom stabilnym znajdującym się w jednym wierszu można przyporządkować ten sam stan elementów pamięci q, ponieważ stany te rozróżniają różne stany elementów wejściowych. Stanom stabilnym znajdującym się w różnych wierszach przyporządkowywane są różne stany elementów q. Stany elementów pamięci należy rozmieszczać w taki sposób, aby każde przejście od stanu niestabilnego do odpowiadającego mu stanu stabilnego pociągało za sobą zmianę stanu tylko jednego elementu pamięci. Należy więc wiersze zawierające odpowiednie stany niestabilne i wiersz zawierający im odpowiadające stany stabilne zakodować sąsiednimi logicznie stanami elementów pamięci. Unika się dzięki temu zjawiska wyścigu. Przy właściwym rozmieszczeniu stanów elementów pamięci dla poszczególnych wierszy korzysta się z pomocy tzw. wykresu przejść (patrz rys. 2.17). Ilustruje on wszystkie przejścia występujące w zredukowanej tablicy programu. Węzły wykresu przejść odpowiadają wierszom tablicy programu. Między węzłami prowadzone są linie odpowiadające przejściom w tablicy. Poszczególnym węzłom przypisuje się liczby dwójkowe (reprezentujące stany elementów pamięci) w taki sposób aby węzły wzajemnie połączone różniły się w swoim opisie dwójkowym stanem tylko jednej zmiennej. Liczby dwójkowe opisujące węzły przyjmuje się do zakodowania poszczególnych wierszy zredukowanej tablicy programu i otrzymuje tablicę (siatkę) przejść (rys. 2.18). W kratki, w których znajdują się stany niestabilne wpisywany jest kod odpowiadający stanowi stabilnemu o tym samym numerze co stan niestabilny. Siatkę przejść należy rozdzielić na dwie oddzielne tablice stanów elementów pamięci (rys. 2.19).

a)

q_1q_2	x_1x_2 00	x_1x_2 01	x_1x_2 11	x_1x_2 10
00	0	0	1	0
01	0	0	0	1
11	1	0	1	1
10	0	1	1	1

Q_1

b)

q_1q_2	x_1x_2 00	x_1x_2 01	x_1x_2 11	x_1x_2 10
00	0	1	0	0
01	0	1	1	1
11	1	1	0	1
10	0	0	0	1

Q_2

Rys. 2.19. Tablice stanów elementów pamięci: a) Q_1, b) Q_2

q_1q_2	Z
00	0
01	0
11	1
10	1

Rys. 2.20. Tablice stanów dla wyjścia Z

Z tablic stanów po przeprowadzeniu minimalizacji uzyskuje się odpowiednie funkcje logiczne:

$$Q_1 = x_1q_1 + x_1x_2\overline{q_2} + \overline{x_2}q_1q_2 + x_1\overline{x_2}q_2 + x_2q_1\overline{q_2} \quad (2.20)$$

$$Q_2 = \overline{x_1}q_1q_2 + \overline{x_1}x_2\overline{q_1} + x_2\overline{q_1}q_2 + x_1\overline{x_2}q_2 + x_1\overline{x_2}q_1 + \overline{x_1}x_2q_2 + x_1\overline{q_1}q_2 \quad (2.21)$$

$$Z = q_1 \quad (2.22)$$

Przy minimalizacji funkcji logicznych należy zwracać uwagę na zjawisko hazardu. W celu jego eliminacji do funkcji opisującej element pamięci Q_2 dodano dwie grupy antyhazardowe (przerywane linie w tablicy stanów elementów pamięci Q_2).

Przykład 2

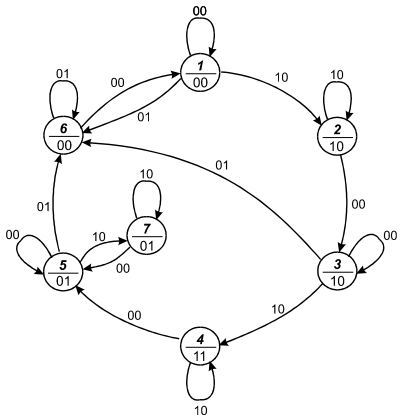
Przeprowadzić syntezę układu zdalnego załączania silnika z następującymi warunkami pracy:

- naciśnięcie przycisku startowego u_1 powinno uruchomić sygnał akustyczny Y_1,
- drugie naciśnięcie przycisku startowego u_1 powinno załączyć napięcie na stycznik Y_2 silnika,
- zwolnienia przycisku po uruchomieniu silnika powinno wyłączyć syrenę,
- naciśnięcie przycisku u_2 powinno wyłączyć silnik lub syrenę w zależności od tego które z tych urządzeń aktualnie.

Rozwiązanie:

Graf przejść projektowanego układu został przedstawiony na rys. 2.21 natomiast pierwotna tablica programu na rys. 2.22. Kompletna tablica programu nie zawiera stanów wewnętrznych równoważnych. W wyniku połączenia wierszy 1,6 i 5,7 otrzymano zredukowaną tablicę programu przedstawioną na rys. 2.23. Do realizacji układu wymagana jest pamięć 3-elementowa, ponieważ zredukowana tablica programu zawiera 5 wierszy. Wykres przejść (rys. 2.24) pokazuje , iż nie można bezpośrednio zakodować zredukowanej tablicy programu (niedopuszczalne przejście a-e). Żeby to wyeliminować wprowadzany jest dodatkowy węzeł f (dodatkowy stan). Zatem zakodowanie wierszy w

zredukowanej tablicy programu wymaga jej rozszerzenia do postaci pokazanej na rys. 2.25. Na podstawie poszerzonej zredukowanej tablicy stworzono siatkę przejść (rys. 2.26).



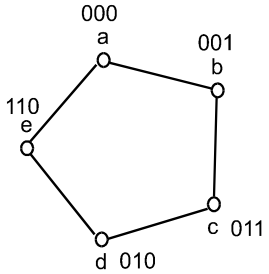
Rys. 2.21. Graf przejść dla układu z przykładu 2

	u_1u_2 00	u_1u_2 01	u_1u_2 11	u_1u_2 10	y_1y_2
1	(1)	6	—	2	00
2	3	—	—	(2)	10
3	(3)	6	—	4	10
4	5	—	—	(4)	11
5	(5)	6	—	7	01
6	1	(6)	5	—	00
7	5	—	—	(7)	01

Rys. 2.22. Kompletna tablica programu do przykładu 2

	u_1u_2 00	u_1u_2 01	u_1u_2 11	u_1u_2 10
a: 1,6	(1)	(6)	—	2
b: 2	3	—	—	(2)
c: 3	(3)	6	—	4
d: 4	5	—	—	(4)
e: 5,7	(5)	6	—	(7)

Rys. 2.23. Zredukowana tablica programu dla przykładu 2



Rys.2.24. Wykres przejść dla przykładu 2

	u_1u_2 00	u_1u_2 01	u_1u_2 11	u_1u_2 10
a: 1,6	(1)	(6)	—	2
b: 2	3	—	—	(2)
c: 3	(3)	6	—	4
d: 4	5	—	—	(4)
e: 5,7	(5)	6'	—	(7)
	—	—	—	—
	—	—	—	—
f:	—	6	—	—

Rys. 2.25. Poszerzona, zredukowana tablica programu dla przykładu 2

$q_1q_2q_3$	u_1u_2 00	u_1u_2 01	u_1u_2 11	u_1u_2 10
a: 000	000	000	—	001
b: 001	011	—	—	001
c: 011	011	000	—	010
d: 010	110	—	—	010
e: 110	110	100	—	110
111	—	—	—	—
101	—	—	—	—
f: 100	—	000	—	—

$Q_1Q_2Q_3$

Rys. 2.26. Tablica przejść dla przykładu 2

a)

$q_1q_2q_3$	u_1u_2 00	u_1u_2 01	u_1u_2 11	u_1u_2 10
000	0	0	—	0
001	0	—	—	0
011	0	0	—	0
010	1	—	—	0
110	1	1	—	1
111	—	—	—	—
101	—	—	—	—
100	—	0	—	—

Q_1

$q_1q_2q_3$	u_1u_2 00	u_1u_2 01	u_1u_2 11	u_1u_2 10
000	0	0	—	0
001	1	—	—	0
011	1	0	—	1
010	1	—	—	1
110	1	0	—	1
111	—	—	—	—
101	—	—	—	—
100	—	0	—	—

Q_2

$q_1q_2q_3$	u_1u_2 00	u_1u_2 01	u_1u_2 11	u_1u_2 10
000	0	0	—	1
001	1	—	—	1
011	1	0	—	0
010	0	—	—	0
110	0	0	—	0
111	—	—	—	—
101	—	—	—	—
100	—	0	—	—

Q_3

b)

q_1	q_2q_3 00	q_2q_3 01	q_2q_3 11	q_2q_3 10
0	0	1	1	1
1	—	—	—	0

Y_1

q_1	q_2q_3 00	q_2q_3 01	q_2q_3 11	q_2q_3 10
0	0	0	0	1
1	—	—	—	1

Y_2

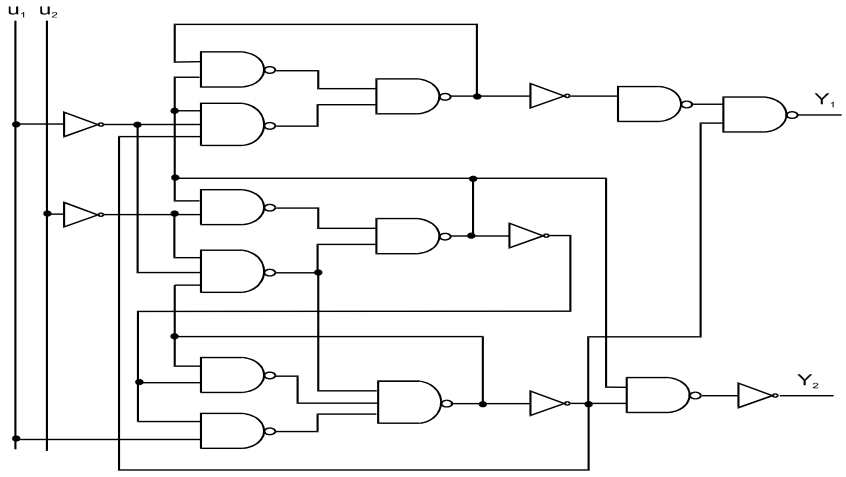
Rys. 2.27. Tablice stanów elementów: a) pamięci; b) wyjść

Z tablic stanów (rys. 2.27) otrzymywane następujące funkcje logiczne:

$$Q_1 = q_1q_2 + \overline{u_1}q_2\overline{q_3} \quad (2.23) \quad Q_2 = \overline{u_2}q_2 + \overline{u_1}\,\overline{u_2}q_3 \quad (2.24) \quad Q_3 = u_1\overline{q_2} + \overline{u_1}\,\overline{u_2}q_3 + \overline{q_2}q_3 \qquad (2.25)$$

$$Y_1 = q_3 + \overline{q_1}q_2 \quad (2.26) \quad Y_2 = q_2\overline{q_3} \qquad (2.27)$$

Zrealizowany w oparciu o funkcje (2.23 – 2.27) układ sekwencyjny przedstawiony jest na rys. 2.28.



Rys. 2.28. Schemat logiczny systemu z przykładu 2 z zastosowaniem logicznych sprzężeń zwrotnych

2.3.2.2.Synteza układu sekwencyjnego na podstawie tablicy kolejności łączeń

Metoda tablicy kolejności łączeń jest szczególnie użyteczna w przypadku sterowania, polegającego na zapewnieniu wzajemnej współzależności pracy kilku mechanizmów w sensie określonej kolejności ich włączania i wyłączania. Metodę stosuje się dla niezbyt dużej liczby elementów wejściowych i wyjściowych (do 6), nawet dla dość skomplikowanych programów pracy.

Tablica kolejności łączeń **(**TKŁ**)** zawiera bieżące stany wszystkich elementów układu przełączającego (automatu) w poszczególnych jego taktach pracy. Rys. 2.29 pokazuje przykładową TKŁ, opisującą działanie automatu 1-no wejściowego, w którym po załączeniu elementu wejściowego x mają być załączone elementy wyjściowe Y_1, Y_2, Y_3 w kolejności wzrastających indeksów, a następnie wyłączone w odwrotnej kolejności.

Poszczególne wiersze tablicy oznacza się nazwami elementów wejściowych, wyjściowych i ewentualnie elementów pośredniczących (pamięciowych) a kolumny – numerami taktów. Symbolem (+) oznacza się stany działania elementu, a symbolem (-) – stany niedziałania. Symbolami oznaczone są tylko takty, w których następują zmiany stanów elementów automatu. Dolny wiersz tablicy służy do dziesiętnego zapisu stanu układu w poszczególnych taktach pracy (tzw. stopień łączenia). W dowolnym takcie każdy element automatu znajduje się w stanie 0 (nie działa) lub w stanie 1 (działa). Stan układu w każdym takcie, można więc przedstawić w kodzie dwójkowym i odpowiadającej mu wartości dziesiętnej. Np.: w takcie 3-cim (rys. 2.29), stan układu określony jest przez stany elementów x=1; Y_1=1; Y_2=1; Y_3=0, co w zapisie dziesiętnym odpowiada liczbie 7 ($0\cdot2^3+1\cdot2^2+1\cdot2^1+1\cdot2^0=7$).

Takty			0	1	2	3	4	5	6	7	8	9	10	...
Stan elementów	x	2^0	-	+							-	+		i t d
	Y_1	2^1	-		+					-			+	
	Y_2	2^2	-			+			-					
	Y_3	2^3	-				+	-						
Stan układu (stopień łączenia)			0	1	3	7	15	7	3	1	0	1	3	...

cykl pracy układu

Rys. 2.29. Przykładowa tablica kolejności łączeń

Takt, w którym wszystkie elementy układu wracają do stanu początkowego jest zakończeniem *cyklu pracy* układu. W tablicy jak na rys.2.29 cykl pracy układu (traktowanego łącznie), obejmuje osiem taktów (od numeru 0 do numeru 7). Praca każdego elementu składa się z następujących po sobie na przemian *cykli działania* i *cykli niedziałania*. Dla każdego elementu tablicy istnieją w pewnych taktach *warunki działania*, a w innych *warunki niedziałania*. Do taktów, w których dla danego elementu istnieją warunki działania, należy dołączyć takt poprzedzający cykl działania (w tym takcie istnieje już przyczyna zadziałania elementu), natomiast wykluczyć ostatni takt cyklu działania (wtedy właśnie powstaje przyczyna niedziałania tego elementu). Analogicznie należy określić zbiór taktów dla warunków niedziałania danego elementu. Np. dla elementu Y_1 warunki jego działania istnieją w taktach 1, 2, 3, 4, 5, a warunki niedziałania w taktach 0, 6, 7.

Do tablicy wprowadza się początkowo tylko elementy wejściowe oraz wyjściowe. Zestawiona w ten sposób TKŁ, w której elementy wyjściowe odgrywają równocześnie rolę elementów pamięciowych, może wystarczyć albo nie wystarczyć do wyznaczenia odpowiednich funkcji logicznych. W drugim przypadku tablicę należy uzupełnić dodatkowymi elementami pośredniczącymi (pamięciowymi). W pierwszym przypadku mamy do czynienia z tzw. *tablicą rozwiązalną*. W takiej tablicy nie występują sprzeczności, polegające np. na tym, że w tych samych warunkach (przy takim samym stanie układu) jakiś element raz ma działać, a drugi raz (w innym takcie) nie działać. W przypadku występowania w tablicy sygnalizowanych sprzeczności, nazywa się ją *tablicą nierozwiązalną.*

TKŁ jest tablicą *rozwiązalną* jeżeli:

1. W czasie jednego cyklu pracy układu nie powtórzy się. Przykładowa tablica z rys. 2.29 jest nierozwiązalna. Wyrażony w zapisie dziesiętnym stan układu jest taki sam dla taktów 1 i 7 (równy 1), taktów 2 i 6 (równy 3) i taktów 3 i 5 (równy 7). Na przykład dla elementu Y_1 występuje jednakowy stan układu równy 3 w taktach 2 i 6 , chociaż takt 2 należy do cyklu działania, a takt 6 do cyklu niedziałania tego elementu.
2. W obrębie tego samego cyklu pracy układu, powtarzający się stan wchodzi zawsze tylko do taktów cyklu działania albo do taktów cyklu niedziałania danego elementu wyjściowego lub elementu pamięci.

W celu ułatwienia rozpoznania rozwiązalności TKŁ (mimo powtarzających się w cyklu pracy układu tych samych stanów) rozróżnia się w tablicy *takty stabilne i niestabilne*. Takty stabilne to takie, po których nie występuje zmiana stanu elementów pamięci. Mogą one trwać dowolnie długo, dopóki nie wystąpi nowa zmiana stanu elementów wejściowych. *Takty niestabilne* to takie, po których następuje zmiana stanu elementów pamięci. Trwają one krótko i po nich stan układu zmienia się samoczynnie aż do osiągnięcia stanu stabilnego. Oznacza się je znaczkiem ^, pod lub nad tablicą.

TKŁ jest więc tablicą *rozwiązalną* jeżeli:

- Nie występują powtórzenia stanu (stopnia łączenia) w cyklu pracy,
- Powtórzenia występują w *taktach stabilnych*,
- Powtórzenia występują w *taktach niestabilnych* ale wywołuje to jednakowe skutki, czyli takie same stany w taktach następnych.

Cechą charakterystyczną TKŁ jest występowanie stosunkowo niewielkiej, w porównaniu do wszystkich możliwych, liczby stanów automatu. Np. jeżeli tablica zawiera 4 elementy to liczba możliwych stanów automatu wynosi 16. Ignorując problem nierozwiązalności np. TKŁ z rys 2.29, realizujący ja automat wykorzystuje jedynie 5 stanów. Pozostałe stany automatu są stanami obojętnymi dla każdego z jego elementów. Dzięki temu struktura automatu sekwencyjnego jest zwykle dosyć prosta, nawet w przypadku realizowania skomplikowanego programu pracy.

Funkcję logiczną dowolnego elementu np., „W" TKŁ można przedstawić w *kanonicznej postaci sumy*

$$F(W) = \sum_{\alpha=1}^{i} K_{\alpha}^{1} + \sum 0^{1} \qquad (2.28)$$

lub *kanonicznej postaci iloczynu*

$$F(W) = \prod_{\beta=1}^{j} K_{\beta}^{0} * \prod 0^{0} \qquad (2.29)$$

gdzie: i – liczba taktów z warunkami działania; j – liczba taktów z warunkami niedziałania; K^1 – składniki jedynki; K^0 – czynniki zera; 0^1 – stany obojętne przyjęte za stany 1 ; 0^0 – stany obojętne przyjęte za stany 0.

W celu określenia funkcji logicznych automatu, należy dla każdego elementu wyjściowego i pamięci rozwiązalnej TKŁ zastosować *cykliczne siatki zależności* (podobnej do tablicy Karnaugha dla układów kombinacyjnych). W odpowiednie kratki siatek zależności wpisywane są obie postacie 2.28 i 2.29 wyrażenia logicznego danego elementu tablicy.

Każda *siatka zależności* przedstawia logiczny związek określonego sygnału wyjściowego lub pośredniczącego w bieżących taktach, z wartościami sygnałów wejściowych, wyjściowych i pośredniczących, w taktach bezpośrednio ich poprzedzających. Argumentami siatek (w interpretacji stykowej) są stany styków elementów układu (oznaczone małymi literami), zaś wartościami stany wzbudzeń cewek (oznaczone dużymi literami).

Od momentu sporządzenia siatek zależności metoda postępowania nie różni się od stosowanej przy syntezie układów kombinacyjnych.

Przykład 3.

Dokonać syntezy układu sekwencyjnego pracującego według tablicy kolejności łączeń jak na rysunku 2.30.

Takty			0	1	2	3	4	5	6	7	8	...
Stan elementów	X	2^0	-	+		-	+		-	+		Itd.
	Y	2^1	-		+			-			+	
Stopień łączenia			0	1	3	2	3	1	0	1	3	...
			∧ ∧						Itd.			

Rys 2.30. Nierozwiązalna tablica kolejności łączeń do przykładu 3

Istnieją powtórzenia w stanie stabilnym i niestabilnym (takt 1, 5 i 2, 4). Tablica jest nierozwiązalna. Wprowadzić należy dodatkowo jeden element pośredniczący (pamięciowy), narzucający taki jego cykl pracy, aby usunąć logiczne sprzeczności w tablicy. W konsekwencji uzyskano tablicę jak na rys. 2.31.

Takty			0	1	2	3	3’	4	5	6	6’	7	8
	X	2^0	-	+		-		+		-		+	
	Y	2^1	-	[	+		]	[	-		]		+
	Q	2^2	-		]	[	+		]	[	-		
Stopień łączenia			0	1	3	2	6	7	5	4	0	1	3

Rys. 2.31. Rozwiązywalna tablica kolejności łączeń do przykładu 3

Cykl warunków działania elementu Y obejmuje teraz takty 1, 2, 3, 3’ , cykl występowania warunków niedziałania takty 4, 5, 6, 6’ . Analogicznie dla elementu pamięciowego warunki działania występują w taktach 3, 3’, 4, 5, zaś niedziałania w taktach 1, 2, 6, 6’. Uwzględniając stany układu w tych taktach otrzymuje się wyrażenia strukturalne, które w umownym zapisie mają następującą postać:

$$F(Y)= \sum(1,2,3,6)_{x,y,q} + \sum 0^1 \quad F(Q)= \sum(2,5,6,7)_{x,y,q} + \sum 0^1 \quad (2.30)$$

$$F(Y)= \prod(0,4,5,7)_{x,y,q} \bullet \prod 0^0 \quad F(Q)= \prod(0,1,3,4)_{x,y,q} \bullet \prod 0^0 \quad (2.31)$$

Dla elementu wyjściowego i elementu pamięci cykliczne siatki zależności będą (rys. 2.32).

x\y,q	00	01	11	10
0	0	1	1	1
1	0	0	0	1

Y

x\y,q	00	01	11	10
0	0	0	0	1
1	0	1	1	1

Q

Rys. 2.32. Cykliczne siatki zależności dla przykładu 3

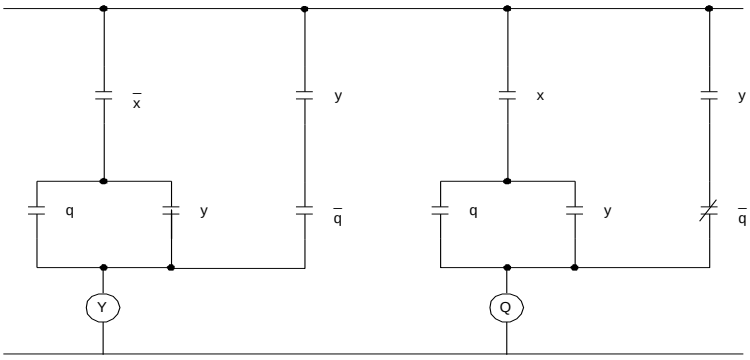
Duże litery Y, Q oznaczają stany wzbudzeń „cewek przekaźników" – elementów wyjściowych , natomiast małe litery x, y, q stany ich styków. Przeprowadzając minimalizację otrzymuje się odpowiednie wyrażenia strukturalne dla „wzbudzeń cewek" wyjścia i elementu pamięci układu. Dla prawidłowego działania układu wprowadzone zostały dodatkowe grupy (zachodzące na siebie) po to, aby wyeliminować możliwość powstawania hazardu (tzw. grupy antyhazardowe).

$$F(Y)= \bar{x}q + \bar{x}y + y\bar{q}\,;\ F(Q)= xq + xy + y\bar{q} \qquad (2.32)$$

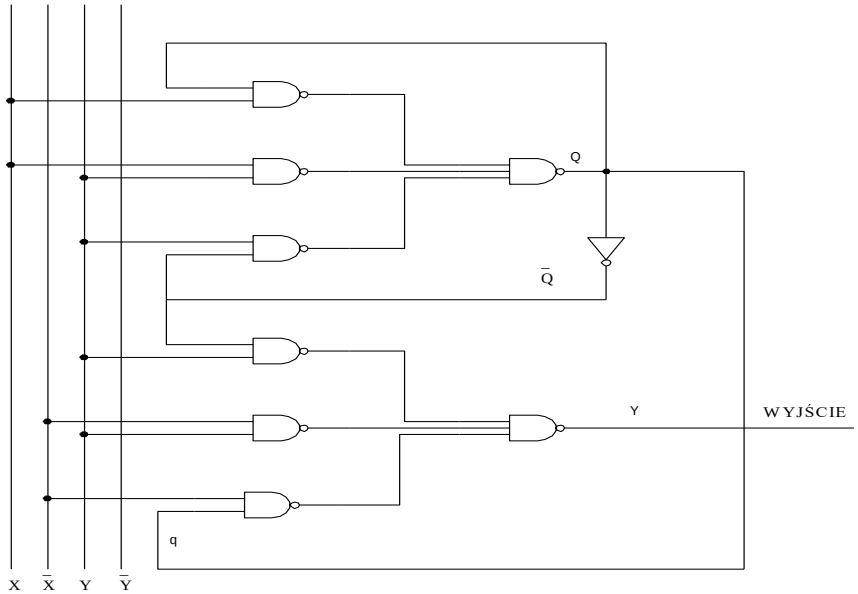
Dla całego układu wyrażenie strukturalne (warunki działania) będzie miało następującą postać:

$$F= \left(\bar{x}q + \bar{x}y + y\bar{q}\right)Y + \left(xq + xy + y\bar{q}\right)Q \qquad (2.33)$$

Odpowiednie schematy połączeń w stykowej i bezstykowej realizacji układu przełączającego przedstawione zostały na rysunkach 2.33. i 2.34.



Rys. 2.33 Układ sekwencyjny do przykładu 3 zestawiony na podstawie warunków działania – wersja stykowa



Rys. 2.34. Układ sekwencyjny do przykładu zestawiony na podstawie warunków działania – wersja bezstykowa

2.3.3.Synteza układu sekwencyjnego z zastosowaniem przerzutników

Przy syntezie układów sekwencyjnych z przerzutnikami należy najpierw wyznaczyć funkcje logiczne opisujące blok pamięci. Można przy tym zastosować dowolną metodę np. metodę tablic programu lub metodę tablic kolejności łączeń. Na podstawie wyznaczonych wyrażeń funkcji pamięci określa się funkcje wzbudzeń przerzutników, które mają być w układzie zastosowane.

Dostępne są różne typy przerzutników np.: RS, JK, T. Najpierw należy zdecydować, który typ przerzutnika będzie wykorzystywany. Tablice wzbudzeń dla wymienionych typów przerzutników prezentuje rys. 2.35.

q -> Q	r s	j k	t
0 0	- 0	0 -	0
0 1	0 1	1 -	1
1 0	1 0	- 1	1
1 1	0 -	- 0	0

Rys. 2.35. Zestawienie tablic wzbudzeń typowych przerzutników asynchronicznych

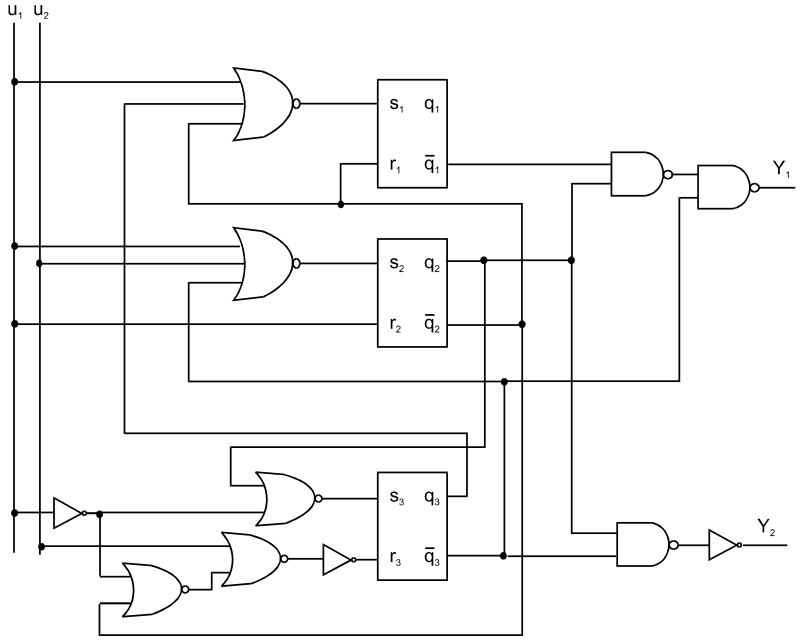
Funkcje logiczne realizowane przez odpowiednie typy przerzutników asynchronicznych są następujące:

SR: $Q = \bar{r}(s+q)$ lub $Q = s + \bar{r}q)$ lub $Q = \bar{r}q + s\bar{q}$ (2.35)

JK: $Q = \bar{k}q + j\bar{q}$ (2.36) T: $Q = \bar{T}q + T\bar{q}$ (2.37)

Najczęściej stosowanymi metodami wyznaczania funkcji wzbudzeń przerzutników są:
- Metoda wykorzystująca siatki stanów elementu pamięci i tablice wzbudzeń przerzutnika,
- metoda algebraiczna,
- metoda transformacji funkcji logicznej elementu pamięci.

Przykładową realizację układu sekwencyjnego zrealizowanego na elementach bezstykowych na elementach bezstykowych i wykorzystującego do budowy bloku pamięci przerzutniki pokazuje rys. 2.37.



Rys. 2.37. Przykładowy schemat logiczny układu sekwencyjnego, w którym blok pamięci zbudowano z przerzutników z zastosowaniem przerzutników

2.2. Instrukcja wykonania ćwiczenia nr 2

Ćwiczenie składa się z następujących części:

A - Synteza kombinacyjnych układów sterowania logicznego
B - Synteza sekwencyjnych układów sterowania logicznego
C- opracowania sprawozdania

A. Synteza kombinacyjnych układów sterowania logicznego

Synteza układów przełączających wykonywana w trakcie ćwiczenia wspomagana jest komputerowo przez programy symulacyjne.

Program ***LabView, Laboratorium Elektroniczne*** (Moduł Cyfrowy v. 1.5) firmy DEGEM w wersji dla systemu DOS i w wersji dla systemu operacyjnego Windows mogą być używane do:

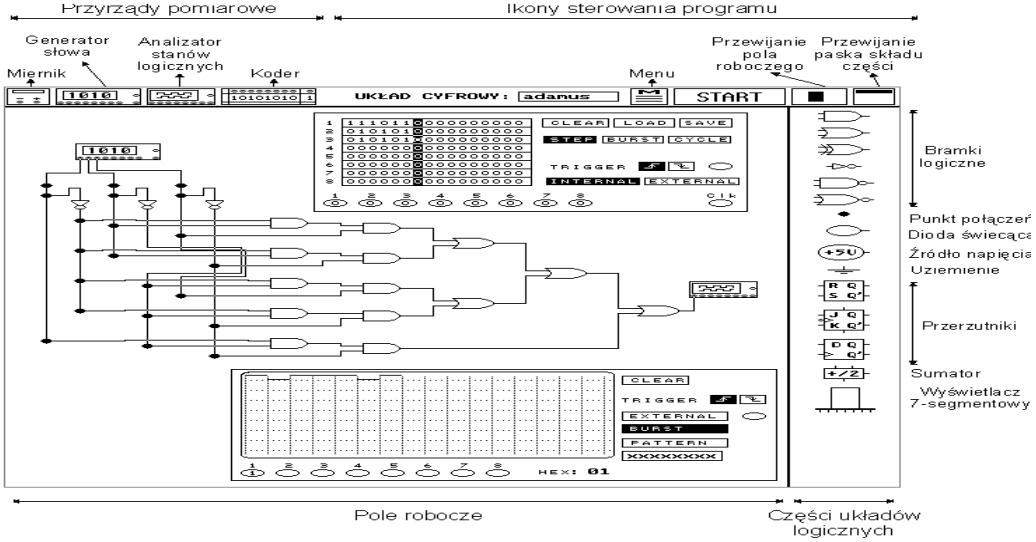
- zestawiania wirtualnego schematu układu przełączającego (też elektronicznego),
- symulowania działania układu,
- śledzenia pracy programu za pomocą symulowanych przyrządów pomiarowych,
- drukowania kopii schematu, wskazań przyrządów i listy elementów.

Moduł cyfrowy programu otwierany jest przez uruchomienie pliku ***digital.bat*** lub uruchamiany z ikony znajdującej się w odpowiedniej bibliotece na ekranie.

Widok ekranu programu przedstawiono na rys. 2.38. Menu programu składa się z następujących funkcji (najczęściej używanych):

- objaśnienia (F1) – uzyskanie informacji na temat wybranego elementu programu,
- usuwanie (F2) – usuwanie zaznaczonych elementów z pola roboczego. Zaznaczenie odbywa się z użyciem prawego klawisza myszy.
- kopia (F3) – kopiowanie znaczonego elementu (elementów) w polu roboczym,
- przesunięcie (F4) – przesuwanie zaznaczonych elementów w polu roboczym,
- makrodefinicja (F5) – łączenie zaznaczonej grupy elementów w bloki. Oznacza to tworzenie nowych części, takich jak np. wielowejściowe bramki logiczne (dostępne są tylko dwuwejściowe) lub tworzenie większych układów z mniejszych modułów,
- opis (F6),
- powiększenie (F7) - powiększenie wybranego przyrządu pomiarowego w polu roboczym,
- obrót (F8) – obrót o 90° znaczonego elementu (elementów) w polu roboczym,
- plik (F9) – wykonywanie operacji plikowych tj. zapis bieżącego układu na dysk lub odczyt wcześniej stworzonego,
- druk – drukowanie schematu, wskazań przyrządów lub listy elementów,
- wyjście – wyjście z programu.

Dostęp do poszczególnych opcji można uzyskać poprzez naciśnięcie myszką na przycisk menu i nie zwalniając go przesunąć w dół najeżdżając na wymaganą opcję. Większości opcji można ponadto użyć używając odpowiednich klawiszy funkcyjnych.



*Rys. 2.38. Ekran programu **Laboratorium Elektroniczne** z przykładową aplikacją*

W celu zaprojektowania układu należy wykonać następujące operacje:

- wybrać elementy ze składu części i umieścić w polu roboczym (metodą przeciągnij i upuść – ang. drag and drop, z użyciem lewego klawisza myszy),
- połączyć elementy ze sobą (bezpośrednie łączenie ze sobą zaczepów poszczególnych części lub pośredni poprzez punkty łączeniowe),
- opcjonalne dołączenie przyrządów pomiarowych (tj. umieszczenie przyrządu w polu roboczym, dołączenie przewodów do punktów pomiarowych, powiększenie przyrządu przez dwukrotne „kliknięcie" lub naciśnięcie F7 w celu dokładnego obejrzenia ekranu przyrządu oraz ustawienia zakresu pomiarowego),
- naciśnięcia START w celu uruchomienia układu.

Do przyrządów cyfrowych należą:

- miernik – woltomierz,
- generator słowa – źródło wejściowych sygnałów binarnych (słów) dla układu. Lewa połówka płyty czołowej generatora zawiera tabelę z słowami, którą należy wypełnić. Po uruchomieniu generatora (naciśnięcie przycisku STEP, BURST lub CYCLE), bity z każdej kolumny tabeli są przesyłane do odpowiednich końcówek przyrządu (nr końcówki = nr wiersza),
- analizator stanów logicznych – pokazuje wykres czasowy sygnałów na jego wejściach,
- konwerter (koder)– pozwala na wykonanie konwersji pomiędzy różnymi reprezentacjami funkcji logicznej: tablicą prawdy, wyrażeniem boolowskim, realizacją układową i może pełnić funkcje automatycznego narzędzia wspomagania syntezy.

Przebieg ćwiczenia dla części A

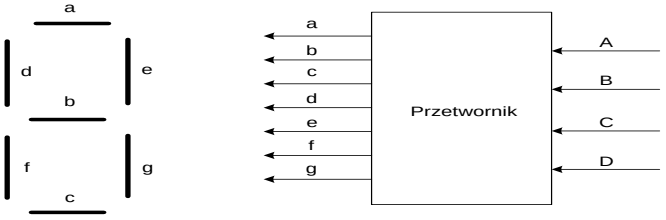
1. Sporządzić tablice stanów układu dla zadanego przez prowadzącego zadania.
2. Uprościć tablicę stanów.
3. Dokonać minimalizacji metodą tablic Karnaugha funkcji logicznych reprezentujących poszczególne wyjścia układu z rozwiązywanego zadania.
4. Na podstawie zminimalizowanych funkcji dokonać syntezy układu przełączającego, wykorzystując jedynie bramki NAND lub NOR.
5. Zamodelować otrzymane układy w programie ***Laboratorium Elektroniczne*** *lub* ***LabView*** i dokonać symulacji w celu sprawdzenia poprawności ich pracy.

Zestaw zadań do części A

Zadanie 1. Zaprojektować urządzenie do głosowania: głosuje n = 4 osób przy użyciu przycisków. Gdy liczba wciśniętych przycisków $n \geq 3$ na wyjściu układu powinien pojawić się sygnał „1".

Zadanie 2. Zaprojektować urządzenie do głosowania: głosuje n = 4 osób przy użyciu przycisków. Gdy liczba wciśniętych przycisków jest nieparzysta na wyjściu układu powinien pojawić się sygnał „1".

Zadanie 3. Zaprojektować przetwornik kodu 8421 (dziesiętnego) na kod wskaźnika 7- segmentowego. Projekt ograniczyć do dwóch wybranych segmentów wyjściowych.



Zadanie 5. Na wejście układu podawana jest liczba czterobitowa. Zbudować układ sprawdzający, czy liczba ta jest podzielna przez 3.

Zadanie 6. Zaprojektować układ sterowania grzejnikiem zawierającym jedną grzałkę. W celu zwiększenia niezawodności sterowania zastosowano trzy jednakowe czujniki temperatury, grzałka załącza się, gdy temperatura spada poniżej jednego ustalonego po-

ziomu. Układ sterujący powinien być odporny na awarię dowolnego jednego z trzech czujników, tzn. powinien wykrywać stan większości swoich trzech wejść.

Zadanie 7. Zaprojektować układ inteligentnego sterowania bramą przesuwną. Użytkownik ma do dyspozycji dwa przyciski opisane „otwórz" oraz „zamknij" (stan logiczny 1 po przyciśnięciu), przy czym brama przesuwa się tylko podczas przytrzymywania przycisku. Ponadto układ powinien reagować na sygnały z czujnika pozycji skrajnej otwartej oraz czujnika pozycji skrajnej zamkniętej (stan logiczny 1 dla pozycji skrajnej). Układ sterujący ma dwa wyjścia załączające przesuw bramy odpowiednio otwieranie i zamykanie, przy czym stan 1 nie może pojawić si jednocześnie na obu wyjściach. Rozważyć samodzielnie reakcję układu na wystąpienie niepoprawnej kombinacji sygnałów wejściowych. Opisać przyjte założenia oraz symbole wejść i wyjść.

Zadanie 8. Zaprojektować układ dokonujący konwersji trzybitowej liczby binarnej ABC na trzybitową liczbę XYZ w kodzie Graya. Każde dwie kolejne liczby zapisane w kodzie Graya różnią się dokładnie jednym bitem. Kolejne wartości binarne 000, 001, 010, 011, 100, 101, 110, 111 mają następujące reprezentacje w kodzie Graya: 000, 001, 011, 010, 110, 111, 101, 100.

Zadanie 9. Zaprojektować układ, który na dwubitowym wyjściu XY wybiera mniejszą z dwóch dwubitowych liczb binarnych AB oraz CD podanych na wejścia.

Zadanie 10. Zaprojektować układ, który przenosi sygnały z trzybitowego wejścia ABC na trzybitowe wyjście XYZ w następujący sposób: dla wejścia sterującego w stanie R = 0 układ zwraca wynik XYZ = ABC, natomiast dla R = 1 dokonuje rotacji cyklicznej w lewo, tzn. zwraca wynik XYZ = BCA.

Zadanie 11. Zaprojektować układ, który będzie obliczał sumę dwóch liczb 2-bitowych.

Zadanie 12. Zaprojektować układ, który będzie obliczał iloczyn dwóch liczb 2-bitowych.

B. Synteza sekwencyjnych układów sterowania logicznego

Synteza sekwencyjnych układów przełączających wykonywana w trakcie ćwiczenia wspomagana jest komputerowo przez program **Huffman 98**. Jest to mini narzędzie programistyczne, będące efektem pracy dyplomowej. Program wspomaga projektanta w kolejnych żmudnych czynnościach realizacji projektu układu sterowania oraz ułatwia diagnostykę logicznej poprawności syntetyzowanego układu przełączającego. Program jest bardzo łatwy w obsłudze i pomimo ograniczeń (tylko metoda tablic programu Huffmana oraz realizacja pamięci przez sprzężenia) może być cennym narzędziem, również dla dydaktyki.

B.1. Metodyka pracy z programem Huffman98

Huffman98 jest programem wspomagającym syntezę asynchronicznych układów sekwencyjnych. Produktem wyjściowym syntezy z użyciem programu Huffman98 są funkcje logiczne elementów pamięci i elementów wyjściowych oraz schemat logiczny układu (automat Moore'a w wersji bramkowej NAND & NOT).

Punktem wyjściowym syntezy z użyciem programu Huffman98 jest wypełnienie lub losowe wygenerowanie pierwotnej (kompletnej) tablicy programu PTP. W odniesieniu do PTP program posiada następujące ograniczenia:

- max liczba elementów wejściowych A, B, C,...,H wynosi 8,
- max liczba elementów wyjściowych Z1,Z2,Z3,...,Z6 wynosi 6,
- max liczba stanów ustalonych automatu (wierszy) wynosi 256

Tablica wypełniana jest przez wpisywanie w poszczególne kratki numerów stanów lub „-"(stan zabroniony), zaznaczenie stanu stabilnego odbywa się przez naciśnięcie **spacji**. Przechodzenie pomiędzy kolejnymi kratkami tablicy realizowane jest przez naciskanie kursorów na klawiaturze.

Po wypełnieniu PTP program automatycznie realizuje poszczególne etapy projektowania automatu. W konsekwencji „wciskania" lewym przyciskiem „myszy" przycisku **Dalej**, program wykonuje następujące operacje:

- redukuje liczbę wierszy PTP (tworzenie zredukowanej tablicy programu - ZTP),

- tworzy tablicę przejść (TP) między poszczególnymi stanami automatu,
- koduje tablicę przejść stanami elementów pamięci – tworzenie tablicy adresów (TA),
- buduje tablice stanów elementów pamięci oraz elementów wyjść (TK – tablice Karnaugh'a),
- tworzy zminimalizowane równania boolowskie (funkcje logiczne) dla elementów pamięci i wyjść
- wykreśla kompletny schemat logiczny układu w realizacji bramkowej.

Program umożliwia również po dokonaniu syntezy przeprowadzenie symulacji pracy zaprojektowanego automatu. Realizowane jest to przez sekwencyjne zadawanie stanów sygnałów wejściowych myszką (elementy w dolnym prawym rogu okna programu).

Przebieg ćwiczenia dla części B

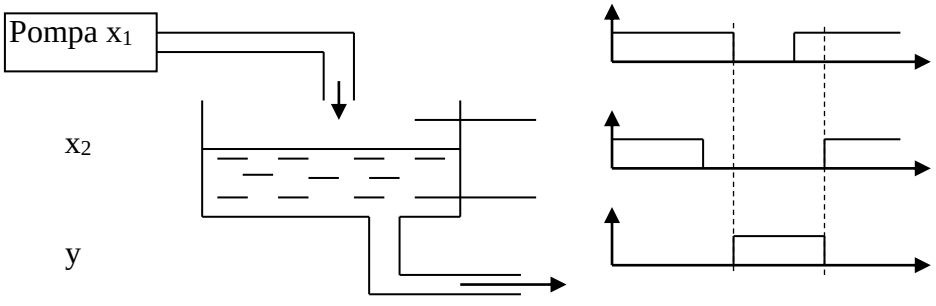
1. Utworzyć graf przejść dla zadania wybranego przez prowadzącego.
2. Wygenerować pierwotną tablicę stanów stabilnych.
3. Uzupełnić pierwotną tablicę stanów stanami niestabilnymi.
4. Zredukować kompletną tablicę.
5. Sporządzić tablicę przejść (siatki przejść).
6. Sporządzić tablicę stanów elementów pamięci.
7. Sporządzić tablicę stanów elementów wyjściowych.
8. Określić funkcje logiczne realizowane przez elementy pamięci i wyjść.
9. Na podstawie funkcji logicznych sporządzić schemat układu logicznego.
10. Wprowadzić kompletną tablicę stanów do programu Huffman98 i szczegółowo śledząc jego działanie porównać jego wyniki z efektami syntezy „ręcznej". Przeprowadzić symulację działania w programie Huffman i sprawdzić czy jest poprawna.
11. Ocenić czy „ręczna" synteza była poprawna, jeśli nie to zastanowić się gdzie popełniony został błąd.
12. Zaprojektować układ sekwencyjny z wykorzystaniem przerzutników RS.

B.2. Zestaw zadań do części B

Zadanie 1. Zaprojektować układ załączania silnika asynchronicznego do sieci za pomocą stycznika S. Załączenie przyciskiem Z, wyłączenie przyciskiem W.

Zadanie 2. Przeprowadzić syntezę układu zdalnego załączania silnika przenośnika taśmowego z następującymi warunkami pracy: naciśnięcie przycisku startowego Z powinno uruchomić sygnał akustyczny, który powinien rozlegać się tak długo, jak długo przycisk Z będzie naciśnięty. W momencie zwolnienia przycisku Z stycznik S powinien załączyć silnik. Wyłączenie układu powinno być dokonywane przyciskiem W.

Zadanie 3. Przeprowadzić syntezę układu stabilizacji cieczy w wieży ciśnień wg poniższych przebiegów czasowych. Dwie pompy Y_1 , Y_2, powinny pracować naprzemiennie a czujniki x_1 i x_2 gdy będą zanurzone w cieczy to na ich wyjściach pojawią się sygnały „1".



Zadanie 4. Zaprojektować układ sterowania silnika prądu stałego. Silnik jest sterowany trzema stycznikami: S_1 – obroty w prawo, S_2 – obroty w lewo, S_3 – hamowanie dynamiczne. Układ sterowniczy uruchamiany jest trzema przyciskami niestabilizowanymi: P (obroty w prawo), L (obroty w lewo) i H (hamownie). Dodatkowo dostępny jest sygnał z czujnika ruchu O, który ma wartość „1" gdy silnik się obraca, a wartość „0" gdy silnik pozostaje w spoczynku. Gdy silnik jest zatrzymany można przeprowadzić rozruch w dowolnym kierunku. W wyniku naciśnięcia przycisku H, powinno nastąpić najpierw wyłączenie stycznika obrotu w danym kierunku, później załączenie stycznika hamowania. W momencie całkowitego zatrzymania sil-

nika stycznik hamowania powinien się samoczynnie wyłączyć. W sytuacji gdy silnik się obraca naciśnięcie któregokolwiek przycisku rozruchu powinno być ignorowane.

Zadanie 5. Zaprojektować układ otwierania zamka szyfrowego trzema przyciskami A, B, C. Otwarcie następuje po podaniu sekwencji przyciśnięć ABB. Popełnienie jakiegokolwiek błędu powoduje uruchomienie alarmu.

Zadanie 6. Zaprojektować zamek szyfrowy otwierany dwoma przyciskami naciskanymi w sposób następujący:

a) 00, 01, 11
b) 11, 01, 10
c) 00, 01, 11, 10, 00.

Sposób w jaki były naciskane przyciski poprzednio nie ma wpływu na działanie układu.

Zadanie 7. Zaprojektować układ sterowania bramą wjazdową. Brama wyposażona jest w: silnik napędzający w dwu kierunkach (zamykanie lub otwieranie), dwa czujniki krańcowe tj. całkowitego zamknięcia lub otwarcia bramy oraz pilot z jednym przyciskiem monostabilnym. Przyciśnięcie przycisku pilota powoduje: rozpoczęcie otwierania bramy, gdy jest zamknięta i zamykania, gdy jest otwarta. W przypadku, gdy brama nie jest całkowicie zamknięta lub otwarta przyciśnięcie przycisku pilota powoduje jej otwarcie.

Zadanie 8. Zaprojektować asynchroniczny dzielnik częstotliwości przez:

a) 2,
b) 3,
c) 4,

Wypełnienie impulsów wejściowych i wyjściowych wynosi 0,5.

C. Opracowanie sprawozdania

Sprawozdanie powinno zawierać stronę tytułową ze wszystkimi danymi identyfikacyjnymi, cel i zakres, ogólny opis narzędzi programowego wspomagania syntezy wykorzystywanych w ćwiczeniu, treści zadań, etapy syntezy i jej wyniki. Do sprawozdania powinien być dołączony protokół.

Literatura

1. Siwiński J.: Układy przełączające w automatyce. WNT, W-wa 1980
2. Węgrzyn S.: podstawy automatyki. PWN, W-wa 1980
3. Wiszniewski A. (red): Teoria sterowania. Ćwiczenia laboratoryjne. Wrocław 1997
4. *Poradnik inżyniera automatyka,* WNT, W-wa 1973
5. Matuszyk M., Mazurewicz G.: *Huffman98 komputerowy program wspomagający syntezę cyfrowych automatów sekwencyjnych*, III Sympozjum Naukowe Sterowanie i Monitorowanie układów przemysłowych SM'99, Kazimierz Dolny 1999.
6. Siwiński J.: *Laboratorium teorii systemów i teorii sterowania. Systemy przełączające,* Wyd. Pol. Śląskiej, Gliwice 1980.
7. Traczyk W.: *Układy cyfrowe automatyki*, WNT, 1974.

Wzór protokołu (lekko zaciemnione pola wypełnia prowadzący)

Laboratorium Podstaw Automatyki					
Temat: Wspomagana komputerowo synteza układów przełączających					Nr: **2**
Grupa:	**Imiona i nazwiska osób:** **1.** **2.** **3.** **4.**	**Podpisy:**	**Data wykonania:**	**Termin:** **[ ] - planowy** **[ ] – odróbkowy**	**Ocena:**
Zespół:			**Data oddania:**	**Opóźnienie:**	**Dzień tygodnia:** **Godz. zajęć:**

Podsumowanie części A:

L.p.	**Etap**	**Wykonanie**		
		Poprawne	Poprawne, ale z małymi błędami	Z rażącymi błędami lub niewykonane
1.	Sporządzenie tablicy stanów układu dla zadanego przez prowadzącego zadania.			
2.	Uproszczenie tablicy stanów.			
3.	Minimalizacja metodą tablic Karnaugha funkcji logicznych reprezentujących poszczególne wyjścia układu.			
4.	Synteza układu przełączającego (opracowanie schematu połączeń).			
5.	Zamodelowanie otrzymanego układu w programie ***Laboratorium Elektroniczne*** *lub* ***LabView*** i symulacja.			
6.	Poprawność syntezy.			
Uwagi:				

Podsumowanie części B:

L.p.	**Etap**	**Wykonanie**		
		Poprawne	Poprawne, ale z małymi błędami	Z rażącymi błędami lub niewykonane
1.	Utworzenie grafu przejść dla zadania wybranego przez prowadzącego.			
2.	Utworzenie pierwotnej tablicy stanów stabilnych.			
3.	Uzupełnienie pierwotnej tablicy stanów stanami niestabilnymi.			
4.	Redukcja tablicy.			
5.	Sporządzenie tablicy przejść (siatki przejść).			
6.	Sporządzenie tablicy stanów elementów pamięci.			
7.	Sporządzenie tablicy stanów elementów wyjściowych.			
8.	Określenie funkcji logicznych realizowanych przez elementy pamięci i wyjść.			
9.	Sporządzenie schematu układu logicznego.			

10.	Symulacja w programie Huffman98 i jej poprawność.			
11.	Poprawność „ręcznej” syntezy.			
12.	Zaprojektowanie układu sekwencyjnego z wykorzystaniem przerzutników RS.			
Uwagi:				

Realizacja ćwiczenia przez studentów: