Synteza strukturalna automatów Moore'a i Mealy

Formalna definicja automatu:

$A = < Z, Q, Y, \Phi, \Psi, q_0 >$

Z – alfabet wejściowy

Q – zbiór stanów wewnętrznych

Y – alfabet wyjściowy

Φ – funkcja przejść

$q(t+1) = \Phi\,(q(t), z(t))$

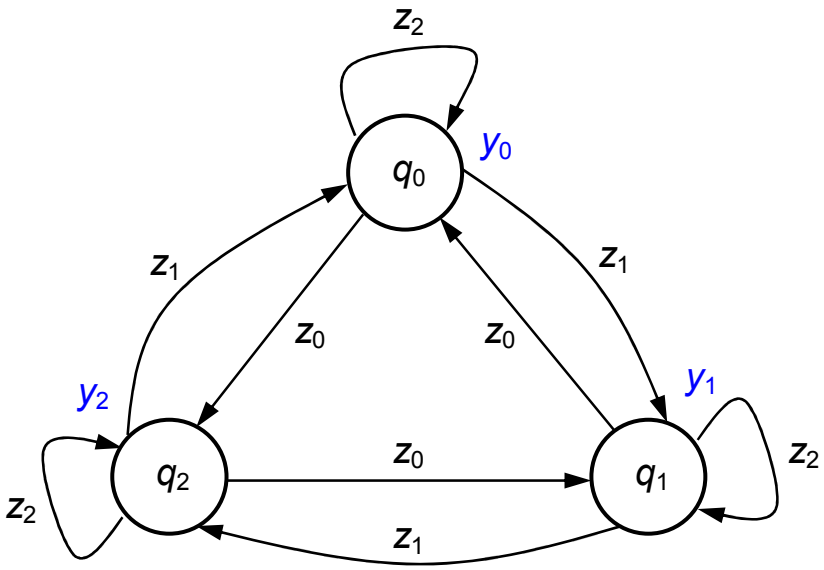
Ψ – funkcja wyjść

automat Moore'a $y = \Psi(q)$
automat Mealy $y = \Psi(q, z)$

q_0 – stan początkowy

Przykład

Załóżmy, że mamy następujący graf automatu Moore'a:



Celem syntezy strukturalnej jest stworzenie schematu układu elektronicznego odzwierciedlającego działanie automatu. Pokazana zostanie synteza układu synchronicznego, w którym wejścia zegarowe wszystkich przerzutników są połączone. Oznacza to, że zmiany stanu przerzutników następują równocześnie, można więc mówić o umownych chwilach t (przed aktywnym zboczem zegara) oraz $t+1$ (po zboczu zegara), wspólnych dla wszystkich przerzutników.

Układ elektroniczny operuje na sygnałach dwustanowych (bitach) wobec tego symbole (z_i, q_i, y_i), występujące w opisie automatu, należy odpowiednio przystosować do układu elektronicznego czyli po prostu zakodować.

1. Kodowanie sygnałów wejściowych

W rozpatrywanym automacie mamy trzy sygnały wejściowe z_0, z_1 i z_2. Minimalna liczba bitów potrzebna do ich zakodowania wynosi 2, bity te oznaczymy jako Z_0 i Z_1. Przykładowe kodowanie przedstawiono w tabeli poniżej:

Tab. 1

	Z_1	Z_0
z_0	0	0
z_1	0	1
z_2	1	0

Możliwe są również inne sposoby kodowania, na przykład omówione dalej kodowanie typu H1 (*hot one*), w którym liczba bitów jest równa liczbie sygnałów. Wybór kodowania może wynikać ze sposobu podawania danych wejściowych.

2. Kodowanie sygnałów wyjściowych

Kodowanie to jest analogiczne do kodowania sygnałów wejściowych. Wybór kodowania zależy od sposobu wykorzystania wyjść. Do zakodowania trzech sygnałów wyjściowych wystarczą dwa bity Y_0 i Y_1. W poniższej tabeli pokazane jest przykładowe kodowanie sygnałów wyjściowych:

Tab. 2

	Y_1	Y_0
y_0	0	0
y_1	0	1
y_2	1	0

3. Kodowanie stanów

Elementem pamięciowym umożliwiającym realizację stanów automatu jest przerzutnik. W układzie logicznym stan automatu będzie reprezentowany przez kombinację stanów przerzutników. Dla trzech stanów konieczne będzie użycie dwóch przerzutników oznaczonych jako Q_0 i Q_1. Symbol Q_i oznacza jednocześnie i-ty przerzutnik oraz wyjście tego przerzutnika. W tabeli poniżej przedstawiono przykładowy sposób kodowania stanów:

Tab. 3

	Q_1	Q_0
q_0	0	0
q_1	0	1
q_2	1	0

W tym przypadku stany zakodowane zostały jako kolejne liczby binarne (Q_1Q_0 odpowiada indeksowi stanu q_i). Niekiedy korzystniejsze może być użycie kodu Graya, trudno jednak wskazać ogólną zasadę, zależy to od układu przejść między stanami automatu. Stosowane są również inne sposoby kodowania (np. omówione dalej kodowanie H1).

4. Kodowanie tabeli przejść

Na podstawie przejść między stanami pokazanych w grafie oraz tabel kodowania dla wejść i stanów można utworzyć zakodowaną tabelę przejść:

Tab. 4

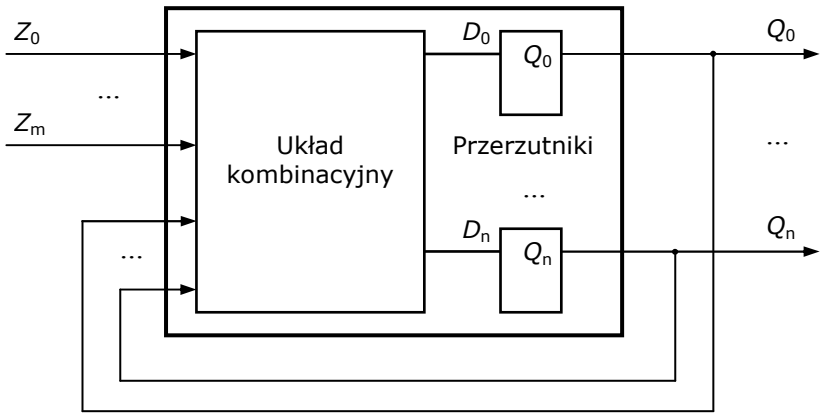
t		t+1
q_i	z_i	q_i
q_0	z_0	q_2
q_0	z_1	q_1
q_0	z_2	q_0
q_1	z_0	q_0
q_1	z_1	q_2
q_1	z_2	q_1
q_2	z_0	q_1
q_2	z_1	q_0
q_2	z_2	q_2

⇒

t				t+1	
Q_1	Q_0	Z_1	Z_0	Q_1	Q_0
0	0	0	0	1	0
0	0	0	1	0	1
0	0	1	0	0	0
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	0	1
1	0	0	0	0	1
1	0	0	1	0	0
1	0	1	0	1	0

5. Synteza funkcji przejść

Synteza funkcji przejść polega na określeniu pobudzeń (funkcji logicznych) dla wejść przerzutników. Sprowadza się to do syntezy układu kombinacyjnego pokazanego na poniższym rysunku. Wejściami układu są sygnały Z_i oraz stany przerzutników Q_i. Liczba wyjść układu jest równa liczbie przerzutników D lub T, dla przerzutników JK jest dwukrotnie większa.



Przy syntezie układu kombinacyjnego sterującego wejściami przerzutników niezbędna jest tabela wzbudzeń przerzutnika, która pokazuje jaki powinien być stan logiczny wejścia (wejść) przerzutnika aby uzyskać żądaną zmianę stanu. Poniżej przedstawiono połączone tabele wzbudzeń dla przerzutników D, T i JK. Symbol Ø oznacza dowolną wartość logiczną (stan wejścia przerzutnika nieistotny):

Tab. 5

$Q(t)$	$Q(t+1)$	$D(t)$	$T(t)$	$J(t)$	$K(t)$
0	0	0	0	0	Ø
0	1	1	1	1	Ø
1	0	0	1	Ø	1
1	1	1	0	Ø	0

Na przykład drugi wiersz w tabeli 5 pokazuje, że aby uzyskać zmianę stanu $Q(t) = 0 \rightarrow Q(t+1) = 1$ na wejścia przerzutników D i T należy podać sygnał 1 a na wejścia przerzutnika JK 1Ø . Proces syntezy zostanie pokazany jednocześnie dla trzech typów przerzutników (zazwyczaj wykonywany jest on tylko dla zadanego lub wybranego typu przerzutnika).

Korzystając z tabel wzbudzeń przerzutników oraz zakodowanej tabeli przejść można uzyskać tabelę pobudzeń dla wejść przerzutników. Na podstawie pary $Q(t) \rightarrow Q(t+1)$ i tabeli wzbudzeń określany jest stan wejścia przerzutnika.

Na przykład pierwszy wiersz tabeli 6 pokazuje, że przerzutnik Q_1 powinien przejść ze stanu 0 w chwili t do stanu 1 w chwili $t+1$. Wobec tego na podstawie tabeli 5 (drugi wiersz) dla przerzutnika D wpiszemy $D_1 = 1$, dla przerzutnika T $T_1 = 1$ a dla przerzutnika JK $J_1 = 1$ i $K_1 = Ø$.

Tab. 6

t				$t+1$			D_1	D_0		T_1	T_0		J_1	K_1	J_0	K_0
Q_1	Q_0	Z_1	Z_0	Q_1	Q_0											
0	0	0	0	1	0		1	0		1	0		1	Ø	0	Ø
0	0	0	1	0	1		0	1		0	1		0	Ø	1	Ø
0	0	1	0	0	0		0	0		0	0		0	Ø	0	Ø
0	1	0	0	0	0		0	0		0	1		0	Ø	Ø	1
0	1	0	1	1	0		1	0		1	1		1	Ø	Ø	1
0	1	1	0	0	1		0	1		0	0		0	Ø	Ø	0
1	0	0	0	0	1		0	1		1	1		Ø	1	1	Ø
1	0	0	1	0	0		0	0		1	0		Ø	1	0	Ø
1	0	1	0	1	0		1	0		0	0		Ø	0	0	Ø

Tabela 6 to zwykła tabela prawdy (zmienne wejściowe: Q_1, Q_0 w chwili t oraz Z_1 i Z_0). Minimalizacja funkcji dla D_1 i D_0 za pomocą siatek Karnaugha wygląda następująco:

Tab. 7 D_1

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00	1		Ø	
01		1	Ø	
11	Ø	Ø	Ø	Ø
10			Ø	1

Tab. 8 D_0

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00		1	Ø	
01			Ø	1
11	Ø	Ø	Ø	Ø
10	1		Ø	

Można zauważyć, że w tabeli przejść (tabela 6) nie występują kombinacje $Z_1Z_0 = 11$ oraz $Q_1Q_0 = 11$, dlatego w siatkach dla tych kombinacji można wstawić znak Ø (wartość dowolna) i wykorzystać to przy minimalizacji. Po zaznaczeniu optymalnych obszarów w siatkach

Karnaugha można zapisać następujące funkcje dla wejść przerzutników:

$$D_1 = \overline{Q_1}\,\overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + Q_0 Z_0 + Q_1 Z_1 \qquad D_0 = Q_1\,\overline{Z_1}\,\overline{Z_0} + \overline{Q_1}\,\overline{Q_0}\,Z_0 + Q_0 Z_1$$

Funkcje logiczne można również określić z postaci kanonicznej, ale wymaga to dalszych przekształceń a postać końcowa może nie być minimalna (jeśli nie są używane wszystkie kombinacje sygnałów wejściowych). W naszym przypadku funkcja D_1 byłaby następująca:

$$D_1 = \overline{Q_1}\,\overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + \overline{Q_1}\,Q_0\,\overline{Z_1}\,Z_0 + Q_1\,\overline{Q_0}\,Z_1\,\overline{Z_0}$$

Analogicznie można przeprowadzić minimalizację z wykorzystaniem siatek Karnaugha dla przerzutników typu T:

Tab. 9 T_1

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00	1		Ø	
01		1	Ø	
11	Ø	Ø	Ø	Ø
10	1	1	Ø	

$$T_1 = \overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + Q_1\,\overline{Z_1} + Q_0 Z_0$$

Tab. 10 T_0

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00		1	Ø	
01	1	1	Ø	
11	Ø	Ø	Ø	Ø
10	1		Ø	

$$T_0 = Q_1\,\overline{Z_1}\,\overline{Z_0} + Q_0\,\overline{Z_1} + \overline{Q_1}\,Z_0$$

W przypadku przerzutników JK siatek Karnaugha będzie oczywiście dwa razy więcej:

Tab. 11 J_1

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00	1		Ø	
01		1	Ø	
11	Ø	Ø	Ø	Ø
10	Ø	Ø	Ø	Ø

$$J_1 = \overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + Q_0 Z_0$$

Tab. 12 K_1

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00	Ø	Ø	Ø	Ø
01	Ø	Ø	Ø	Ø
11	Ø	Ø	Ø	Ø
10	1	1	Ø	

$$K_1 = \overline{Z_1}$$

Tab. 13 J_0

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00		1	Ø	
01	Ø	Ø	Ø	Ø
11	Ø	Ø	Ø	Ø
10	1		Ø	

$$J_0 = Q_1\,\overline{Z_1}\,\overline{Z_0} + \overline{Q_1}\,Z_0$$

Tab. 14 K_0

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00	Ø	Ø	Ø	Ø
01	1	1	Ø	
11	Ø	Ø	Ø	Ø
10	Ø	Ø	Ø	Ø

$$K_0 = \overline{Z_1} + Z_0$$

6. Synteza sygnałów wyjściowych

Z zasady działania automatu Moore'a wynika, że wyjście zależy tylko od stanu automatu (wyjście jest funkcją stanów). Na podstawie kodowania wyjść i stanów (tabele 2 i 3) oraz powiązania wyjść ze stanami (według grafu automatu) można utworzyć następującą tabelę:

Tab. 15

q	y
q_0	y_0
q_1	y_1
q_2	y_2

$\Rightarrow$

Q_1	Q_0	Y_1	Y_0
0	0	0	0
0	1	0	1
1	0	1	0

Po minimalizacji z wykorzystaniem siatek Karnaugha:

Tab. 16 Y_1

Q_1 \ Q_0	0	1
0		
1	1	Ø

$Y_1 = Q_1$

Tab. 17 Y_0

Q_1 \ Q_0	0	1
0		1
1		Ø

$Y_0 = Q_0$

Zwróćmy uwagę, że nawet dla tak prostych funkcji nie warto rezygnować z siatek Karnaugha. Funkcje logiczne dla Y_0 i Y_1, zapisane wprost z postaci kanonicznej (sumy iloczynów odpowiadających wierszom tabeli 15 w których $Y_i = 1$), byłyby następujące:

$$Y_1 = Q_1 \overline{Q_0} \qquad Y_0 = \overline{Q_1}\, Q_0$$

Widać, że dla obu funkcji nie udało się uzyskać postaci minimalnej, chociaż w kolumnach Y_1 i Y_0 w tabeli 15 są tylko pojedyncze jedynki (sugerowałoby to brak możliwości minimalizacji). Byłaby to prawda, ale tylko wtedy, gdyby tabela zawierała wszystkie możliwe kombinacje wartości zmiennych Q_1 i Q_0 (w tym przypadku brak kombinacji $Q_1Q_0 = 11$).

Dla porównania, poniżej pokazane są wyniki syntezy automatu Moore'a dla innego kodowania stanów (kod Graya), pokazanego w tabeli 18. Jak widać, nie ma znaczących różnic jeśli chodzi o złożoność funkcji logicznych.

Tab. 18

	Q_1	Q_0
q_0	0	0
q_1	0	1
q_2	1	1

$$D_1 = \overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + \overline{Q_1}\,Q_0\,Z_0 + Q_1\,Z_1 \qquad D_0 = Q_1\,\overline{Z_0} + \overline{Q_0}\,\overline{Z_1} + Q_0\,Z_1 + \overline{Q_1}\,Z_0$$

$$T_1 = \overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + Q_1\,\overline{Z_1} + Q_0\,Z_0 \qquad T_0 = \overline{Q_1}\,\overline{Z_1}\,\overline{Z_0} + \overline{Q_0}\,\overline{Z_1} + Q_1\,Z_0$$

$$J_1 = \overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + Q_0\,Z_0 \qquad K_1 = \overline{Z_1}$$

$$J_0 = \overline{Z_1} \qquad J_0 = \overline{Q_1}\,\overline{Z_1}\,\overline{Z_0} + Q_1\,Z_0$$

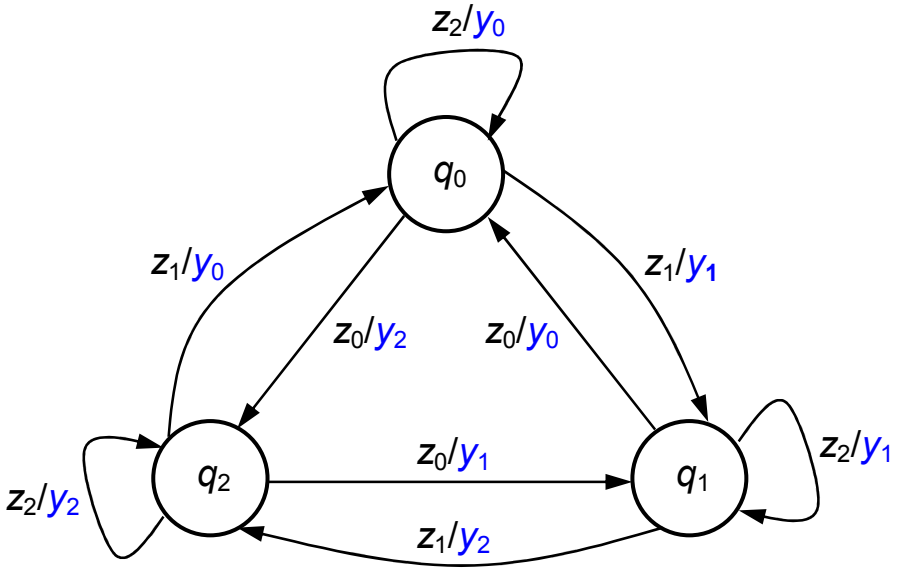
$$Y_1 = Q_1 \qquad Y_0 = Q_0$$

7. Synteza automatu Mealy

Synteza funkcji przejść automatu Mealy jest identyczna jak dla automatu Moore'a, różnica występuje tylko dla sygnałów wyjściowych. W automacie Moore $y = f(q)$ co oznacza, że wyjścia uzależnione są tylko od stanów. W automacie Mealy $y = f(q, z)$ a więc wyjścia zależą zarówno od stanów jak i wejść.

Przykład

Rozpatrzmy automat Mealy o identycznej funkcji przejść jak poprzednio i zmodyfikowanej funkcji wyjść:



Przyjmując takie samo kodowanie wszystkich sygnałów jak dla automatu Moore'a otrzymujemy następującą tabelę dla funkcji wyjść:

Tab. 19

q_i	z_i	y_i
q_0	z_0	y_2
q_0	z_1	y_1
q_0	z_2	y_0
q_1	z_0	y_0
q_1	z_1	y_2
q_1	z_2	y_1
q_2	z_0	y_1
q_2	z_1	y_0
q_2	z_2	y_2

⇒

Q_1	Q_0	Z_1	Z_0	Y_1	Y_0
0	0	0	0	1	0
0	0	0	1	0	1
0	0	1	0	0	0
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	0	1
1	0	0	0	0	1
1	0	0	1	0	0
1	0	1	0	1	0

Podobnie jak w przypadku funkcji przejść dla kombinacji $Z_1Z_0 = 11$ oraz $Q_1Q_0 = 11$ używany jest znak Ø (wartość dowolna). Minimalizacja funkcji wyjść przy użyciu siatek Karnaugha wygląda następująco:

Tab. 20 Y_1

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00	1		Ø	
01		1	Ø	
11	Ø	Ø	Ø	Ø
10			Ø	1

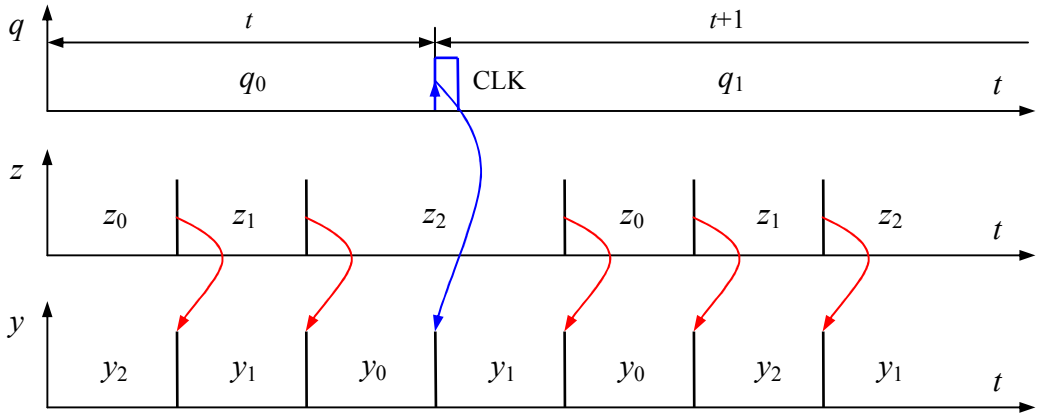
$$Y_1 = \overline{Q_1}\,\overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + Q_0 Z_0 + Q_1 Z_1$$

Tab. 21 Y_0

Q_1Q_0 \ Z_1Z_0	00	01	11	10
00		1	Ø	
01			Ø	1
11	Ø	Ø	Ø	Ø
10	1		Ø	

$$Y_0 = Q_1\,\overline{Z_1}\,\overline{Z_0} + \overline{Q_1}\,\overline{Q_0}\,Z_0 + Q_0 Z_1$$

Warto zwrócić uwagę, że w automacie Mealy wyjścia mogą zmieniać się asynchronicznie, to znaczy bez impulsu zegarowego, wynika to wprost z definicji funkcji wyjść. W automacie Moore wyjścia zmieniają się zawsze synchronicznie ponieważ $y = f(q)$ a zmiana stanów q jest synchroniczna. W automacie Mealy $y = f(q, z)$ a więc nawet przy ustalonym stanie q możliwa jest zmiana wyjść, będąca rezultatem zmiany na wejściach. Ilustrują to pokazane niżej wykresy czasowe:



Większość zmian wyjść pokazanych na wykresach odbywa się asynchronicznie i jest efektem zmiany wejść, przy ustalonym stanie automatu (zaznaczone na czerwono). Można zaobserwować również jedną zmianę wyjścia $y_0 \rightarrow y_1$ (zaznaczoną na niebiesko) przy ustalonym wejściu z_2, wynikającą ze zmiany stanu automatu $q_0 \rightarrow q_1$ pod wpływem impulsu zegarowego.

8. Metoda kodowania H1

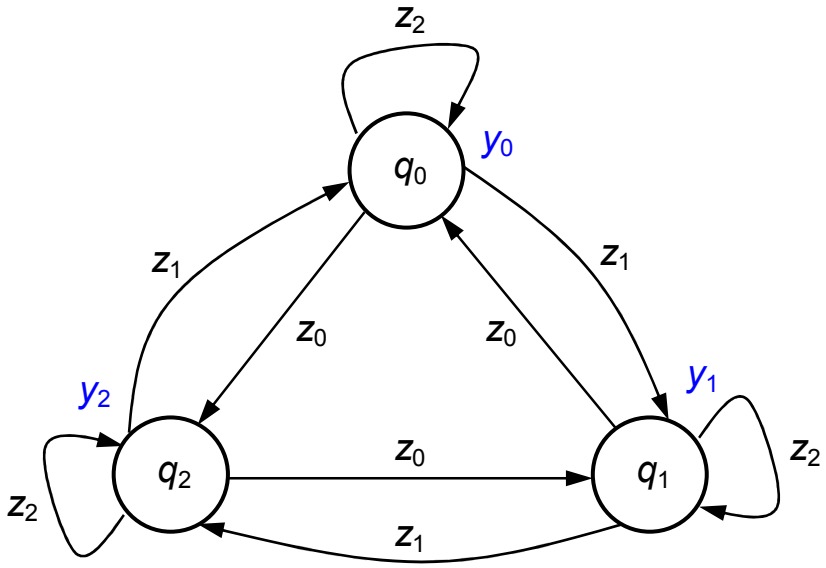
Metoda H1 (*hot one*) polega na takim zakodowaniu stanów na przerzutnikach, że każdy stan reprezentowany jest przez kombinację, w której aktywny jest tylko jeden przerzutnik. Tabela kodowania stanów zgodnie z tą ideą wygląda następująco:

Tab. 22

	Q_2	Q_1	Q_0
q_0	0	0	1
q_1	0	1	0
q_2	1	0	0

Do syntezy automatu przy takim kodowaniu można użyć przedstawionej wcześniej ogólnej metody. Jednak w przypadku użycia przerzutników D można skorzystać z metody

uproszczonej, przedstawionej poniżej. Rozpatrzmy raz jeszcze nasz automat Moore'a:



Z grafu automatu wynika, że przejście do stanu q_0 w chwili t+1 nastąpi wtedy, gdy w chwili t będzie spełniony jeden z następujących warunków:

- automat jest w stanie q_0 i zostanie podany sygnał z_2,
- automat jest w stanie q_1 i zostanie podany sygnał z_0,
- automat jest w stanie q_2 i zostanie podany sygnał z_1.

Inaczej mówiąc, warunki przejścia do stanu q_0 są odzwierciedlone w grafie przez wszystkie łuki dochodzące do stanu q_0. Można to zapisać następująco:

$$q_0(t+1) = q_0(t)\, z_2(t) + q_1(t)\, z_0(t) + q_2(t)\, z_1(t) \qquad (1)$$

W równaniu (1) można zastąpić symbole q_i stanami przerzutników Q_i (pamiętając, że w syntezie używany jest przerzutnik D) wykorzystując następujące zależności:

- ponieważ w każdym stanie tylko jeden przerzutnik jest aktywny więc na przykład dla stanu q_0, zapis po zakodowaniu czyli $\overline{Q_2}\,\overline{Q_1}\,Q_0$, można uprościć do jednego symbolu Q_0, podobnie dla pozostałych stanów,
- z zasady działania przerzutnika D wynika, że $Q(t+1) = D(t)$.

Uwzględniając powyższe zależności równanie (1) można zapisać jako:

$$D_0 = Q_0\, z_2 + Q_1\, z_0 + Q_2\, z_1 \qquad (2)$$

W wyrażeniu (2) sygnały wejściowe z_i nie są jeszcze zakodowane. Zgodnie z tabelą kodowania wejść (tabela 1) możemy zapisać:

Tab. 23

	Z_1	Z_0
z_0	0	0
z_1	0	1
z_2	1	0

$z_0 = \overline{Z_1}\,\overline{Z_0}$

$z_1 = \overline{Z_1}\,Z_0$

$z_2 = Z_1\,\overline{Z_0}$

Po wstawieniu zakodowanych sygnałów wejściowych z_i do wyrażenia (2) otrzymamy:

$$D_0 = Q_0 Z_1 \overline{Z_0} + Q_1 \overline{Z_1}\,\overline{Z_0} + Q_2 \overline{Z_1} Z_0 \tag{3}$$

Po przeprowadzeniu podobnych operacji dla stanów q_1 i q_2 uzyskujemy następujące wyrażenia:

$$q_1(t+1) = q_0(t)\, z_1(t) + q_1(t)\, z_2(t) + q_2(t)\, z_0(t) \tag{4}$$
$$q_2(t+1) = q_0(t)\, z_0(t) + q_1(t)\, z_1(t) + q_2(t)\, z_2(t) \tag{5}$$

Po przekodowaniu:

$$D_1 = Q_0 \overline{Z_1} Z_0 + Q_1 Z_1 \overline{Z_0} + Q_2 \overline{Z_1}\,\overline{Z_0} \tag{6}$$
$$D_2 = Q_0 \overline{Z_1}\,\overline{Z_0} + Q_1 \overline{Z_1} Z_0 + Q_2 Z_1 \overline{Z_0} \tag{7}$$

Uzyskane zostały w ten sposób wyrażenia potrzebne do stworzenia schematu logicznego funkcji przejść automatu. Należy pamiętać, że po resecie automat powinien znaleźć się w stanie początkowym q_0, co oznacza, że układ resetujący musi ustawić przerzutnik Q_0 na 1 a pozostałe przerzutniki na 0.

Kodowanie typu H1 może być użyte również dla wejść i wyjść automatu. Dla omawianego przykładu kodowanie wejść wyglądałoby następująco:

Tab. 24

	Z_2	Z_1	Z_0
z_0	0	0	1
z_1	0	1	0
z_2	1	0	0

Uzasadnieniem dla takiego kodowania może być sposób podawania danych wejściowych. Na przykład trzy różne sygnały wejściowe mogą pochodzić bezpośrednio z przełącznika trzypozycyjnego. Przy kodowaniu wejść na dwóch bitach potrzebny byłby dodatkowy układ enkodera, służący do zakodowania numeru wyróżnionego wejścia jako liczby dwubitowej. Wadą kodowania H1 dla wejść jest zwiększenie liczby zmiennych w syntezie funkcji przejść (dla automatu Mealy również wyjść).

Dla funkcji wyjść kodowanie H1 dałoby następującą tabelę:

Tab. 25

	Y_2	Y_1	Y_0
y_0	0	0	1
y_1	0	1	0
y_2	1	0	0

Kodowanie H1 dla wyjść daje możliwość wykrycia uszkodzenia elementu sygnalizacyjnego (np. diody LED) lub wykonawczego (np. przekaźnika) podłączonego do linii wyjściowej. Ponieważ kombinacja $Y_2Y_1Y_0 = 000$ nie jest używana, jej pojawienie się na wyjściach sygnalizuje problem.

Takiej możliwości nie ma w przypadku kodowania wyjść na dwóch bitach. Wtedy kombinacja $Y_1Y_0 = 00$ może oznaczać zarówno prawidłowe wyjście y_0 jak też wyjście y_1 przy uszkodzeniu na linii Y_1 (lub y_2 przy uszkodzonej linii Y_2).

9. Testowanie automatu

W docelowym zastosowaniu automat jest "czarną skrzynką", której zachowanie odzwierciedlają zmiany na wyjściach. Nie należy jednak rozpoczynać testowania automatu od sprawdzenia funkcji wyjść (lub ograniczać tylko do funkcji wyjść).

Po pierwsze, pełny test automatu, zawężony do obserwacji wyjść, wymagałby zaprojektowania takich ciągów testowych sygnałów wejściowych, które zagwarantowałyby sprawdzenie wszystkich przejść między stanami co w ogólnym przypadku nie jest proste.

Po drugie, funkcja wyjść dla automatów Mealy i Moore'a jest uzależniona od stanów. Oznacza to, że nieprawidłowe zmiany na wyjściach mogą wynikać zarówno z problemu w samej funkcji wyjść jak też z błędów w funkcji przejść (nieprawidłowe zmiany stanów dają w konsekwencji błędne zmiany wyjść), nie można tego rozstrzygnąć bez obserwacji stanów.

Poprawne podejście do testowania automatu wymaga sprawdzenia w pierwszej kolejności funkcji przejść (i usunięcia ewentualnych problemów). Należy zadawać odpowiednie wejścia i obserwować stany wszystkich przerzutników (na podstawie tabeli kodowania stanów można określić w jakim stanie q_i znajduje się automat). Trzeba zweryfikować wszystkie przejścia według grafu lub tabeli przejść. Dla naszego automatu Moore'a (ze względu na jego regularną strukturę) sekwencja testowa jest wyjątkowo prosta:

$z_0\ z_2\ z_0\ z_2\ z_0\ z_2\ z_1\ z_1\ z_1$

Startując ze stanu q_0 można w tym przypadku wykonać pełny test, przechodząc każde przejście tylko raz. W większości przypadków konieczne będą wielokrotne przejścia lub kolejne sekwencje testowe po sprowadzenie automatu do stanu początkowego.

Po wykryciu pierwszego błędnego przejścia należy przeprowadzić analizę układu logicznego i usunąć problem. Testowanie dalszych przejść jest stratą czasu (usunięcie problemu z jednym przejściem może wpłynąć na inne przejścia - zlikwidować lub przeciwnie, ujawnić kolejne błędy). Po ewentualnych zmianach w układzie logicznym zawsze i tak trzeba sprawdzić wszystkie przejścia.

Dopiero przy prawidłowo działającej funkcji przejść można wykonać sprawdzenie funkcji wyjść. Poszukiwanie przyczyn ewentualnych błędów można wtedy zawęzić do układu logicznego bezpośrednio związanego z funkcją wyjść.

Przykład

Rozpoczynamy test naszego automatu Moore'a sprowadzając go do stanu początkowego q_0 i podając sygnał wejściowy z_0. Załóżmy, że już pierwsze przejście jest nieprawidłowe (zamiast przejść do q_2 automat pozostaje w stanie q_0):

t				$t+1$	
Q_1	Q_0	Z_1	Z_0	Q_1	Q_0
0	0	0	0	1	0
				0	0

$\Rightarrow$

t		$t+1$
q_0	z_0	q_2
q_0	z_0	$\boldsymbol{q_0}$

Przejście oczekiwane

Przejście obserwowane

Porównując obserwowany stan $Q_1Q_0 = 00$ ze stanem oczekiwanym $Q_1Q_0 = 10$ można stwierdzić nieprawidłowe zachowanie przerzutnika Q_1 (przerzutnik Q_0 pozostając w stanie 0 zachowuje się poprawnie, nie wymaga więc interwencji).

Aby ustalić przyczynę problemu, trzeba sprawdzić poziom logiczny na wejściu D_1 przerzutnika. Podczas sprawdzania automat musi być w stanie bezpośrednio poprzedzającym błędne przejście (w tym przypadku $Q_1Q_0 = 00$). Możliwe są tutaj dwie sytuacje:

- $D_1 = 1$ czyli prawidłowy poziom logiczny na wejściu. Oznacza to problem z samym przerzutnikiem (brak zasilania, źle podłączony sygnał zegarowy, aktywny sygnał Reset lub uszkodzony przerzutnik),
- $D_1 = 0$ oznacza problem w układzie logicznym sterującym wejściem przerzutnika.

W drugim (częściej występującym) przypadku należy ustalić, czy problem tkwi w połączonym układzie czy też wynika z błędów popełnionych podczas syntezy funkcji przejść. W pierwszym kroku trzeba zweryfikować fragment schematu odpowiadający za sterowanie wejściem D_1 (analiza zerojedynkowa dla kombinacji sygnałów $Q_1Q_0 = 00$ i $Z_1Z_0 = 00$):

- jeśli ze schematu wynika prawidłowa wartość $D_1 = 1$ to znaczy, że testowany układ nie jest zgodny ze schematem. Przyczyną może być błędne połączenie lub problem techniczny (brak zasilania, uszkodzona bramka),
- jeśli analiza daje wartość $D_1 = 0$ to oznacza, że schemat jest błędny i nie zapewnia spodziewanego przejścia między stanami.

W drugim przypadku należy sprawdzić jaką wartość daje funkcja logiczna dla kombinacji $Q_1Q_0 = 00$ i $Z_1Z_0 = 00$, w naszym przypadku:

$$D_1 = \overline{Q_1}\,\overline{Q_0}\,\overline{Z_1}\,\overline{Z_0} + Q_0\,Z_0 + Q_1\,Z_1 = \overline{0}\,\overline{0}\,\overline{0}\,\overline{0} + 0\,0 + 0\,0 = 1111 + 0 + 0 = 1$$

Jeśli wartość wynikająca z funkcji jest prawidłowa, to błąd został popełniony przy przekształceniach funkcji do schematu. W przeciwnym razie juz funkcja logiczna nie zapewnia odpowiedniego sterowania wejściem D_1 i trzeba wyjaśnić, czy wynika to z błędu w syntezie funkcji czy też z nieprawidłowej tabeli prawdy. W siatce Karnaugha należy sprawdzić zawartość komórki odpowiadającej kombinacji $Q_1Q_0 = 00$ i $Z_1Z_0 = 00$:

- jeśli $D_1 = 1$ to błąd wystąpił na etapie minimalizacji lub przekształceń funkcji (nie jest ona zgodna z tabelą prawdy),
- jeśli $D_1 = 0$ to tabela prawdy dla D_1 jest niepoprawna a błąd musiał zostać popełniony przy tworzeniu funkcji przejść (na podstawie tabeli przejść i funkcji wzbudzeń przerzutnika). Możliwe również, że sama tabela przejść jest niepoprawna (błąd przy jej tworzeniu na podstawie grafu lub opisu słownego albo błąd przy kodowaniu stanów).

W przypadku, gdy błędne przejście między stanami wynika z nieprawidłowych zmian więcej niż jednego przerzutnika, trzeba powtórzyć opisane wyżej kroki dla kolejnych przerzutników.