Sławomir Kulesza

Technika cyfrowa
Projektowanie automatów synchronicznych

Wykład dla studentów III roku Informatyki

Wersja 2.0, 20/12/2012

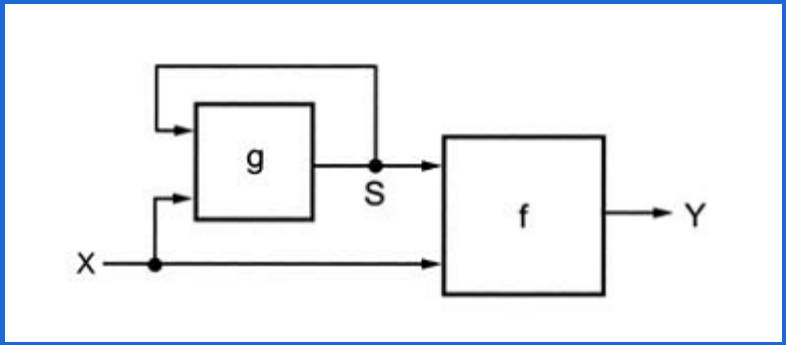
Automaty skończone

- **Układy sekwencyjne nazywane są także automatami skończonymi (Finite State Machine – FSM). Istnieją dwa formalne modele FSM:**

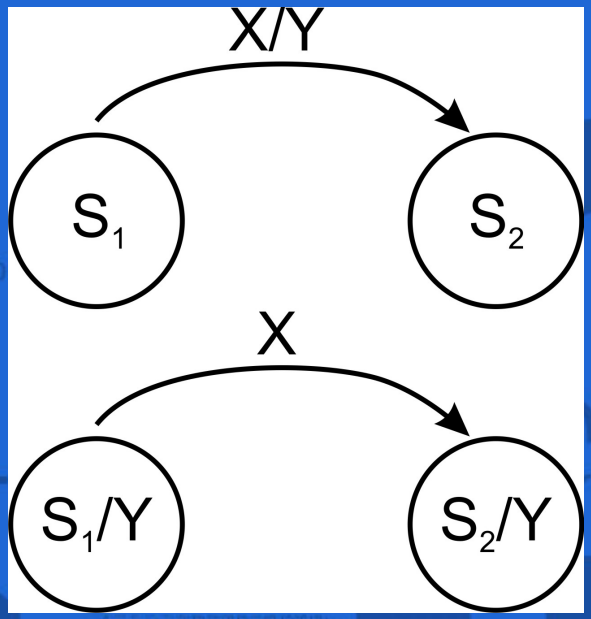
- **Model Moore'a**
  - **Wyjście jest tylko funkcją stanu**
  - **Wyjście opisywane razem ze stanem.**

- **Model Mealy'ego**
  - **Wyjście jest funkcją wejścia i stanu**
  - **Wyjście opisywane nad strzałką przejścia.**

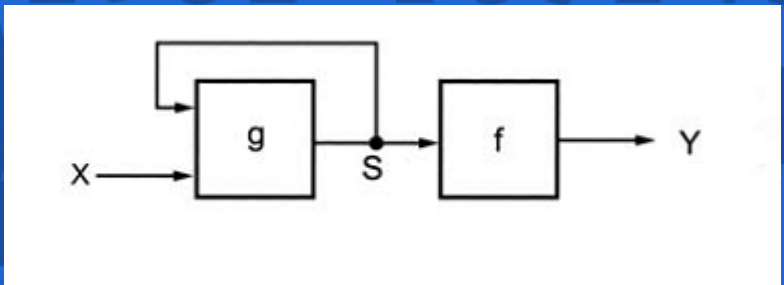
Automat Mealy'ego



Funkcja wyjść: $Y^t = f(S^t, X^t)$



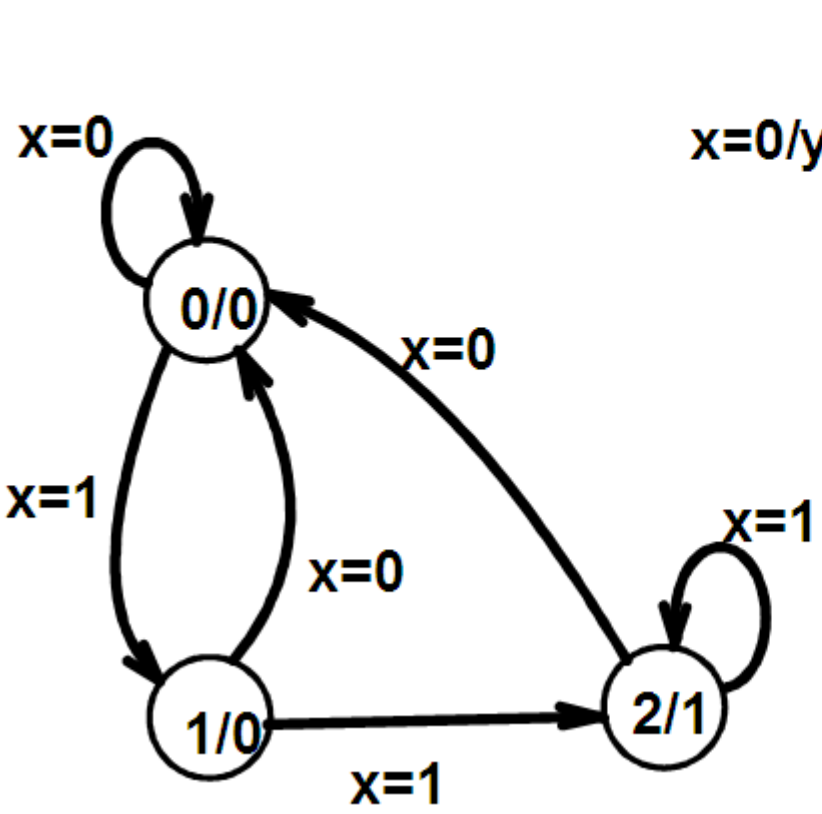
Automat Moore'a

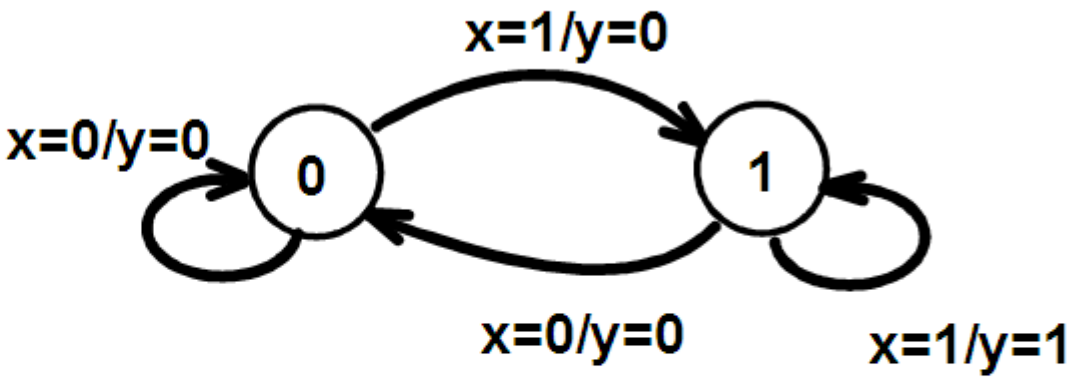


Funkcja wyjść: $Y^t = f(S^t)$

Diagramy stanów a. Mealy'ego i Moore'a

- **Diagram stanów a. Mealy'ego mapuje wejścia i stany wewn. na wyjścia**





- **Diagram stanów a. Moore'a mapuje stany wewn. na wyjścia**

Tablice przejść i wyjść automatów Mealy'ego i Moore'a

- **Tablica stanów a. Moore’a**

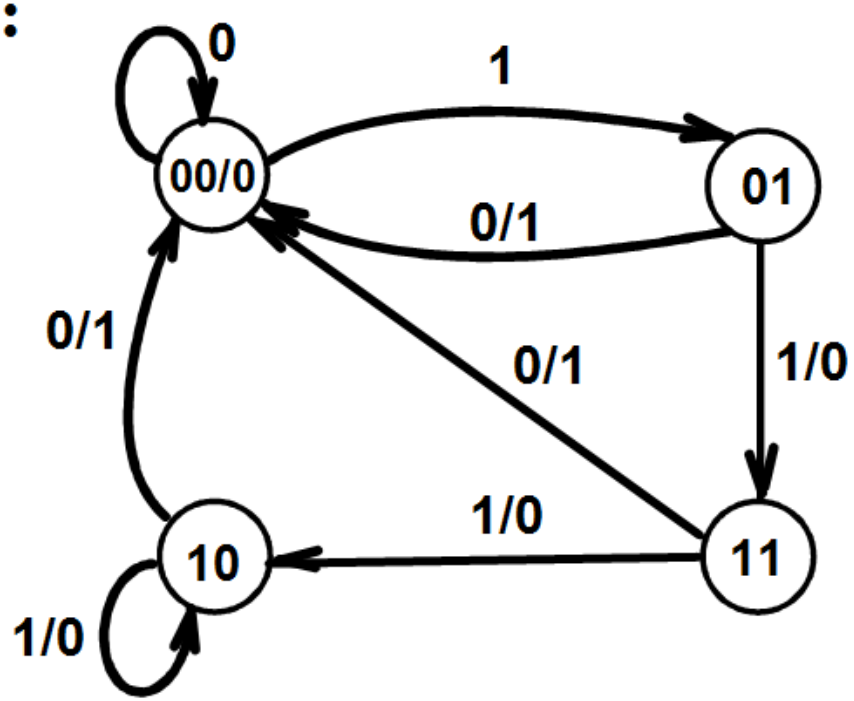
Present State	Next State x=0	Next State x=1	Output
0	0	1	0
1	0	2	0
2	0	2	1

- **Tablica stanów a. Mealy’ego**

Present State	Next State x=0	Next State x=1	Output x=0	Output x=1
0	0	1	0	0
1	0	1	0	1

Modele mieszane

- **W rzeczywistych układach spotyka się mieszany opis automatów:**
- **Stan 00: Moore**
- **Stany 01, 10, 11: Mealy**
- **Uproszczony opis wyjścia**



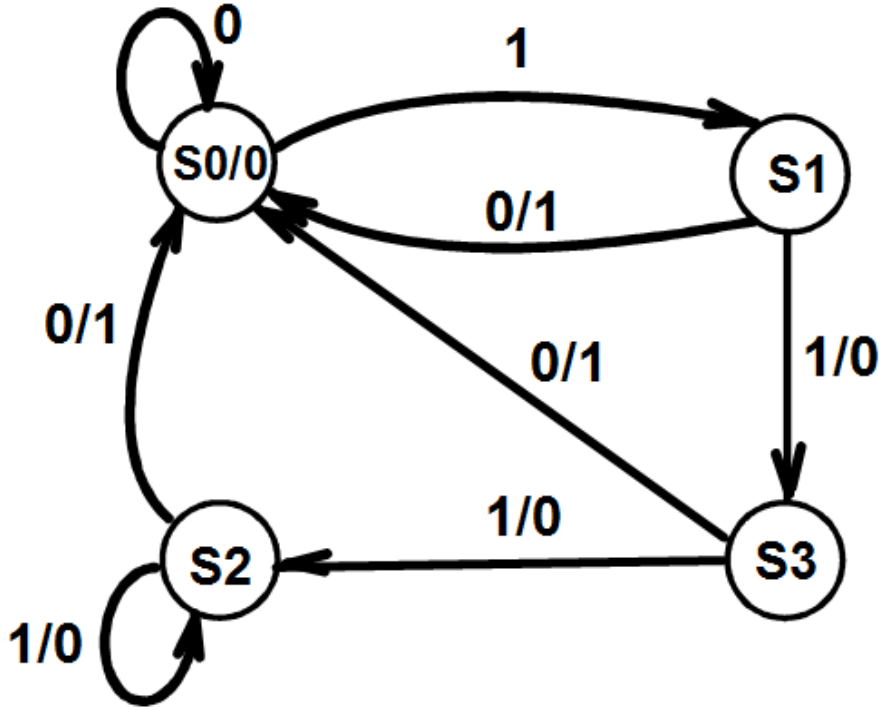
Stany równoważne układu

- **Dwa stany układu są równoważne, gdy w odpowiedzi na dowolne słowo wejściowe generują identyczne słowo wyjściowe.**
- **Inaczej, dwa stany układu są równoważne, gdy pod wpływem tego samego wejścia generują identyczne wyjście i jednocześnie ich następny stan dla dowolnego wejścia jest taki sam.**

Stany równoważne układu

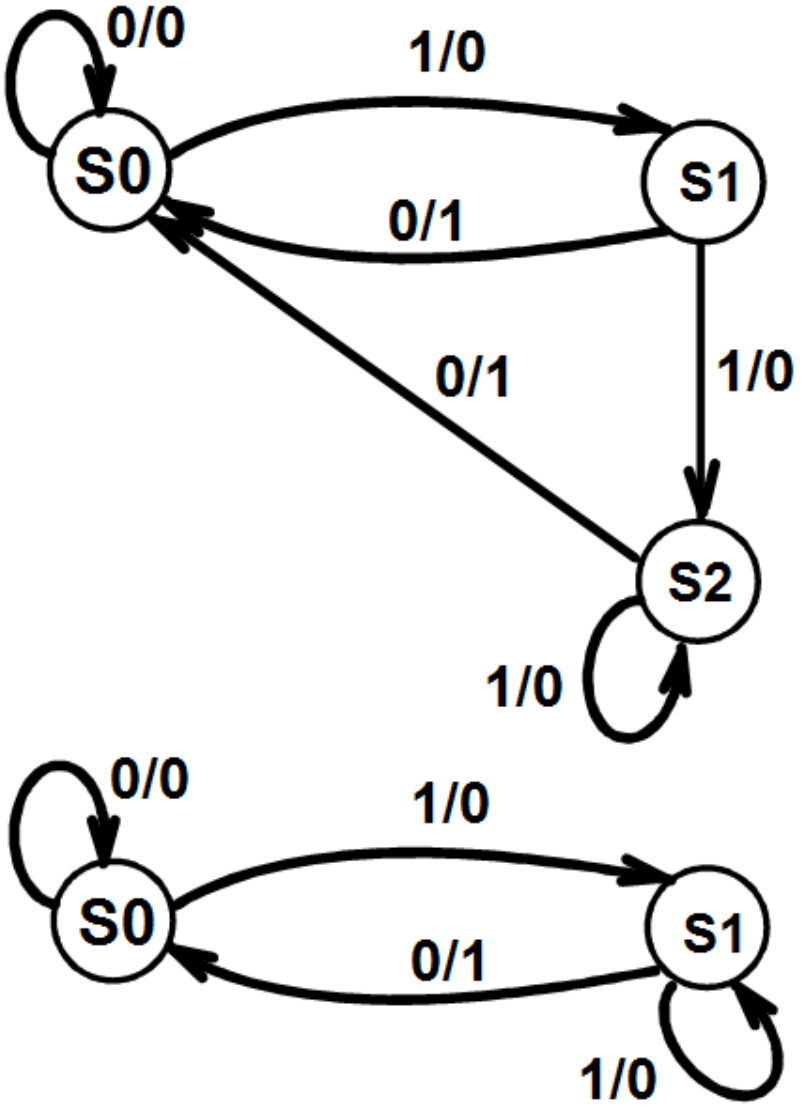
Stany S3 i S2 są równoważne, bo:

- **Generują identyczne wyjście dla tego samego wejścia**
- **Generują identyczny stan następny dla tego samego wejścia**



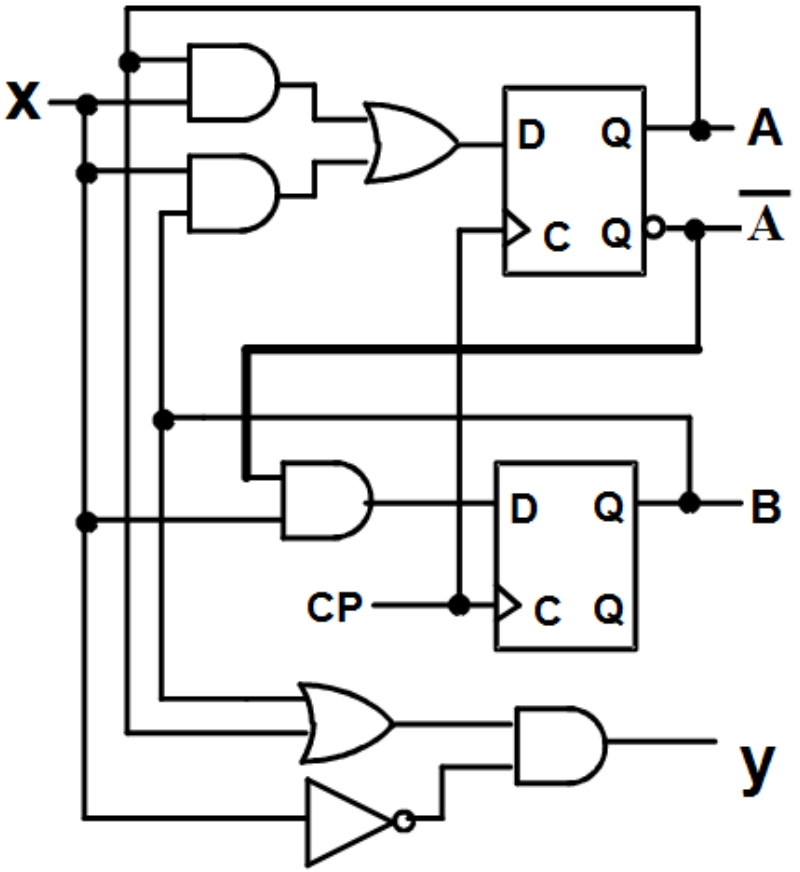
Redukcja stanów równoważnych

- **Stany S3 i S2 można zredukować**
- **Na nowym diagramie widać, że także stany S1 i S2 są równoważne**



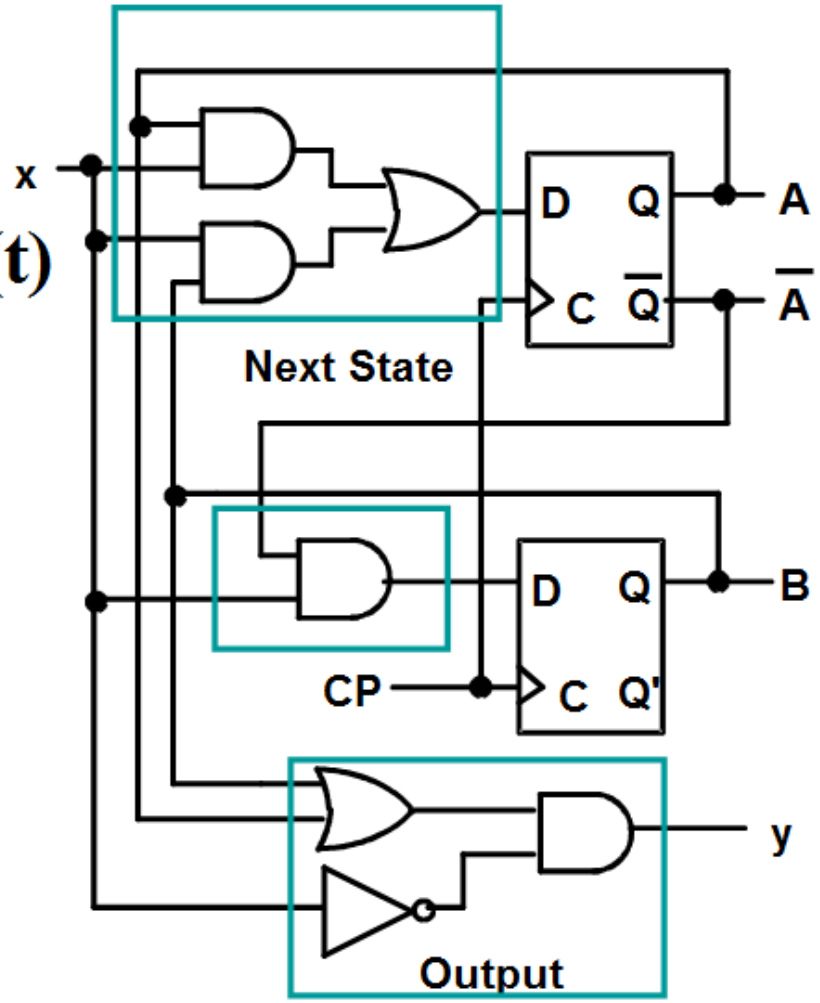
Analiza układów sekwencyjnych

- **Input:** x(t)
- **Output:** y(t)
- **State:** (A(t), B(t))
- **Jaka jest postać funkcji wyjścia?**
- **Jaka jest postać funkcji następnego stanu?**



Analiza układów sekwencyjnych

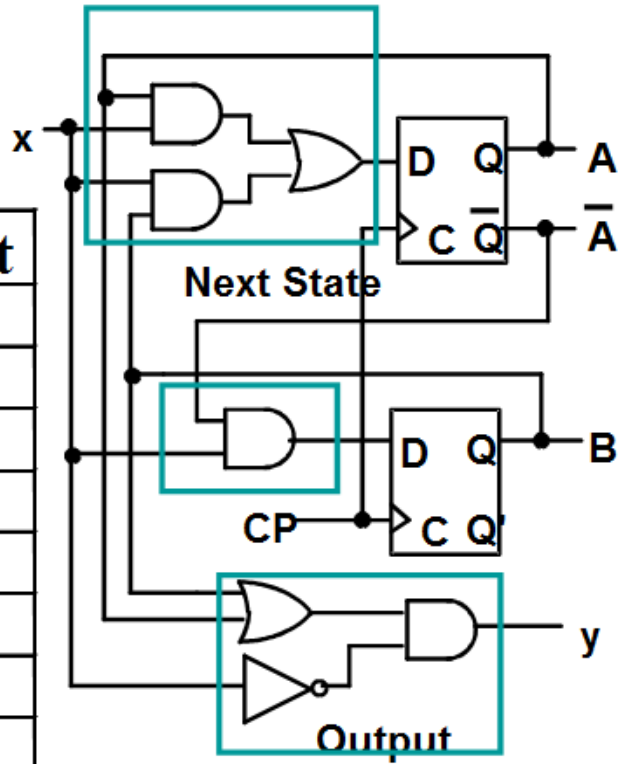
- **Postaci funkcji przełączających:**
  - $A(t+1) = A(t)x(t) + B(t)x(t)$
  - $B(t+1) = \overline{A}(t)x(t)$
  - $y(t) = \overline{x}(t)(B(t) + A(t))$



Deskryptory układów sekwencyjnych
Tablica przejść

- $A(t+1) = A(t)\,x(t) + B(t)\,x(t)$
- $B(t+1) = \bar{A}(t)\,x(t)$
- $y(t) = \bar{x}(t)\,(B(t) + A(t))$

Present State		Input	Next State		Output
A(t)	B(t)	x(t)	A(t+1)	B(t+1)	y(t)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0



Automaty skończone

- **Układy sekwencyjne nazywane są także automatami skończonymi (Finite State Machine – FSM). Istnieją dwa formalne modele FSM:**

- **Model Moore'a**
  - **Wyjście jest tylko funkcją stanu**
  - **Wyjście opisywane razem ze stanem.**

- **Model Mealy'ego**
  - **Wyjście jest funkcją wejścia i stanu**
  - **Wyjście opisywane nad strzałką przejścia.**

Efektywne algorytmy

Z twierdzenia Gödla wynika, że formalizmy matematyczne są niezupełne → istnieją twierdzenia, których nie można w ramach danego formalizmu dowieść → nie można zautomatyzować procesu dowodzenia matematycznego oraz procesu obliczeniowego → istnieją twierdzenia, których nie da się dowieść algorytmicznie.

Maszyna Turinga jest „testerem” algorytmów → algorytm wykonywany przez MT jest wykonywany w ogóle.

Automaty skończone są prostszymi wersjami MT, pozbawionymi nieskończonej pamięci. Nie mogą wykonywać np. nieskończonych rekurencji (mnożenie dwóch dowolnych liczb).

Automaty synchroniczne i asynchroniczne

Automat asynchroniczny nie posiada wejścia zegarowego (synchronizującego), stan wewnętrzny (i odpowiadające mu wyjście) zmienia się wraz ze zmianą sygnału wejściowego → **Mealy**

Automat synchroniczny posiada wejście zegarowe (synchronizujące) wyznaczające chwile, w których może się zmienić stan wewnętrzny. W pozostałym czasie automat jest niewrażliwy na zmiany stanu wejściowego → **Moore**

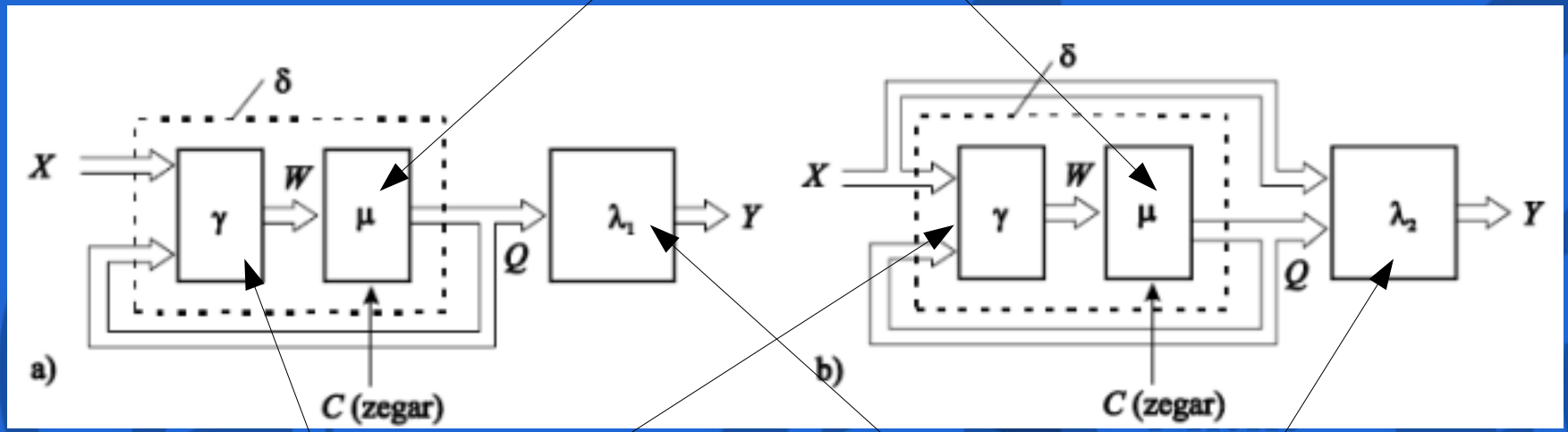
Każdy układ asynchroniczny można traktować jak synchroniczny z nieskończenie wysoką częstotliwością taktowania lub .

Ogólna struktura automatu

Blok wzbudzeń
(synchroniczny/asynchroniczny)

Moore

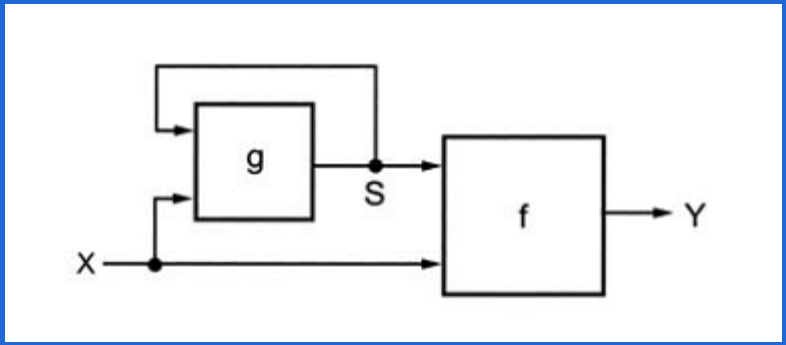
Mealy



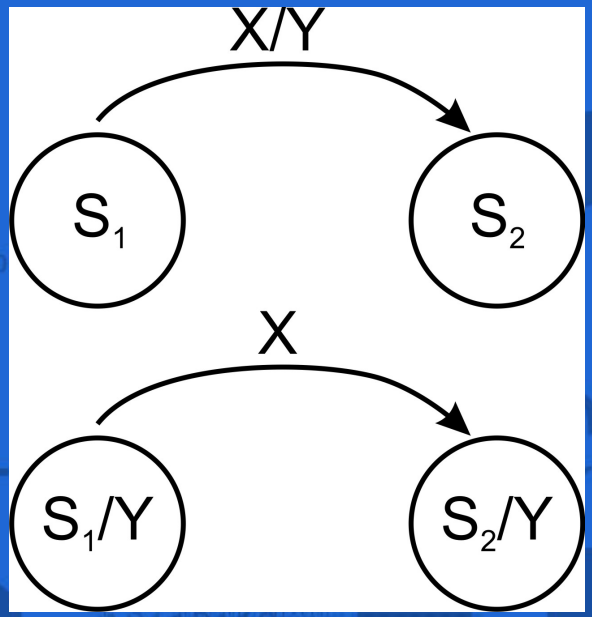
Blok wejściowy
(kombinacyjny, asynchroniczny)

Blok wyjściowy
(kombinacyjny, asynchroniczny)

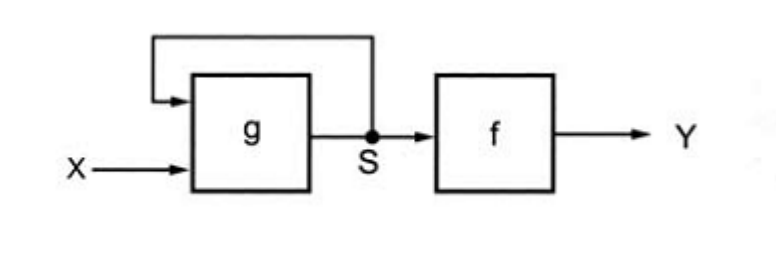
Automat Mealy'ego



Funkcja wyjść: $Y^t = f(S^t, X^t)$



Automat Moore'a

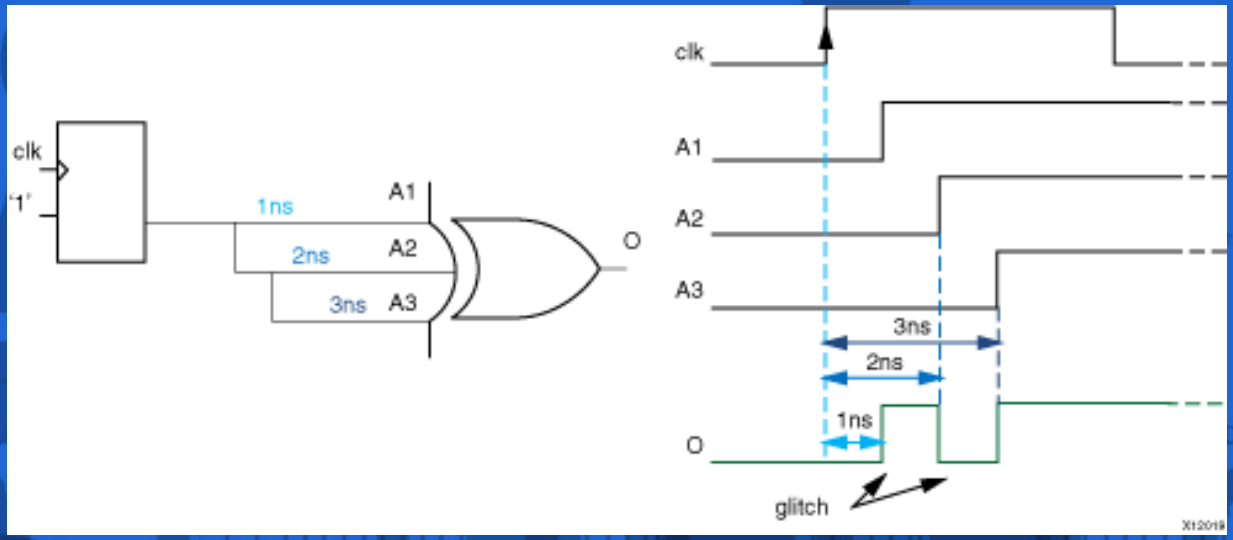


Funkcja wyjść: $Y^t = f(S^t)$

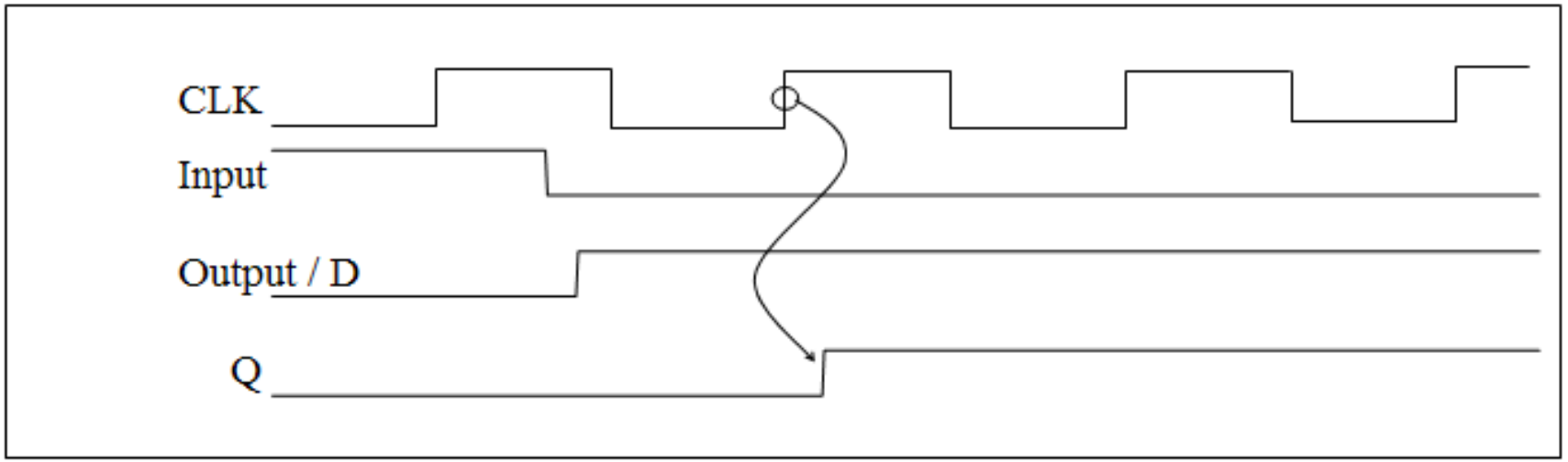
Mealy vs. Moore

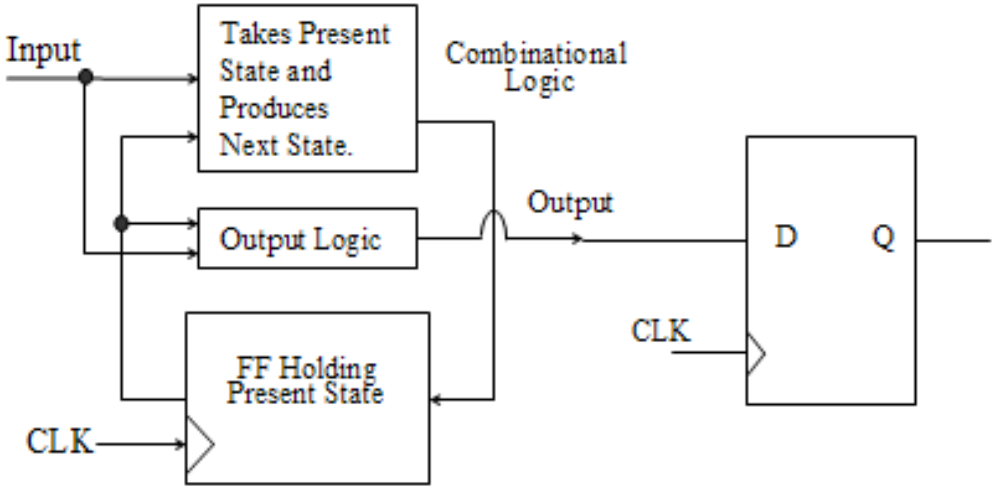
Automaty Mealy'ego są prostsze

- Mniejsza liczba stanów, prostszy układ
- Szybsza reakcja na zmiany wejścia – brak oczekiwania na sygnał zegara
- Asynchroniczne wyjścia i sprzężenia zwrotne niosą zagrożenia – wyścigi krytyczne i niekrytyczne (races), niestabilność sygnałów wejściowych przenosi się na sygnał wyjściowy (feedthrough loops), czas ustalania się sygnałów (glitching)



Automaty Mealy'ego



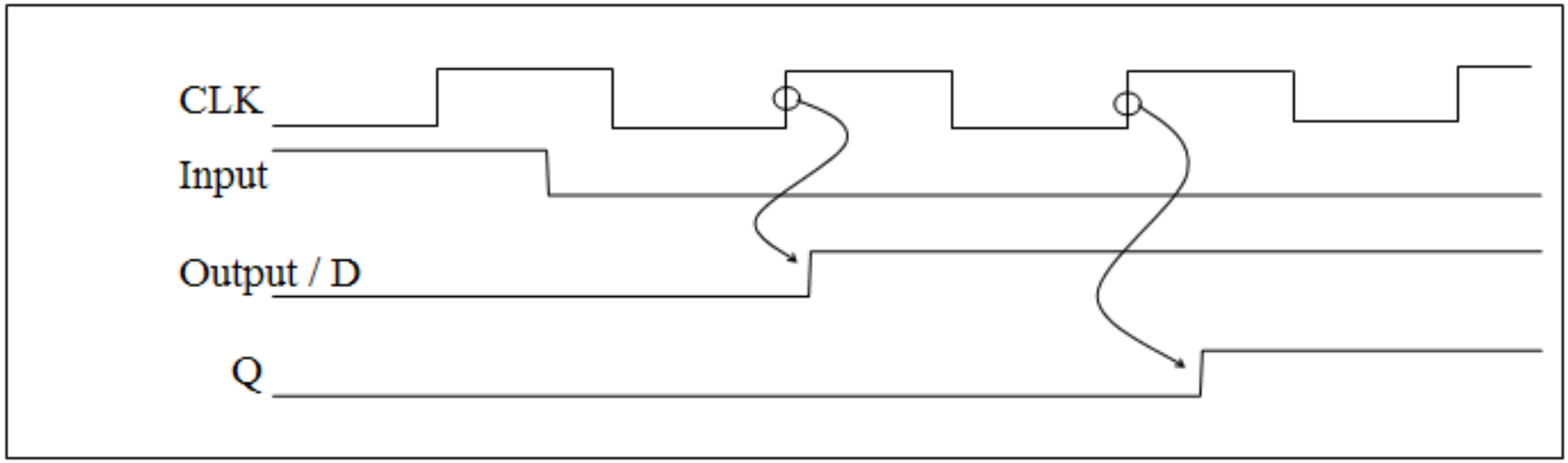


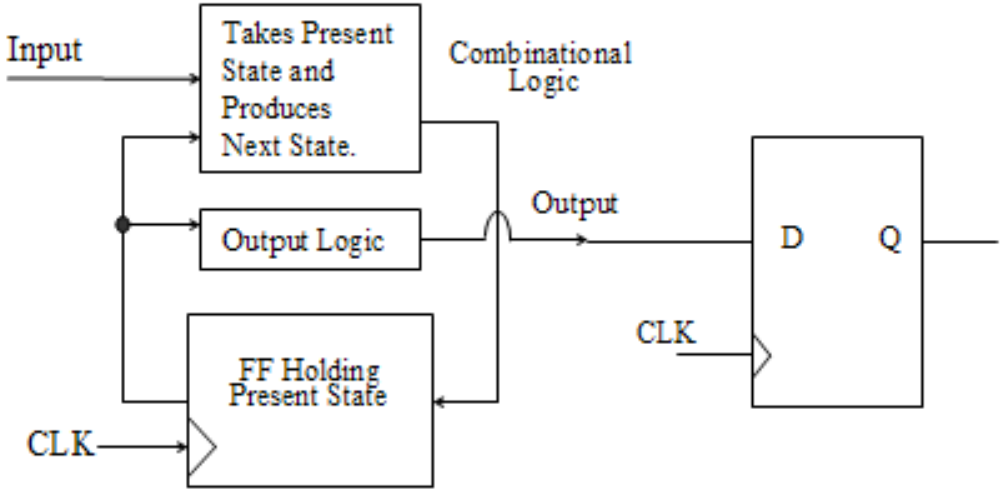
Mealy vs. Moore

Automaty Moore'a są bezpieczniejsze

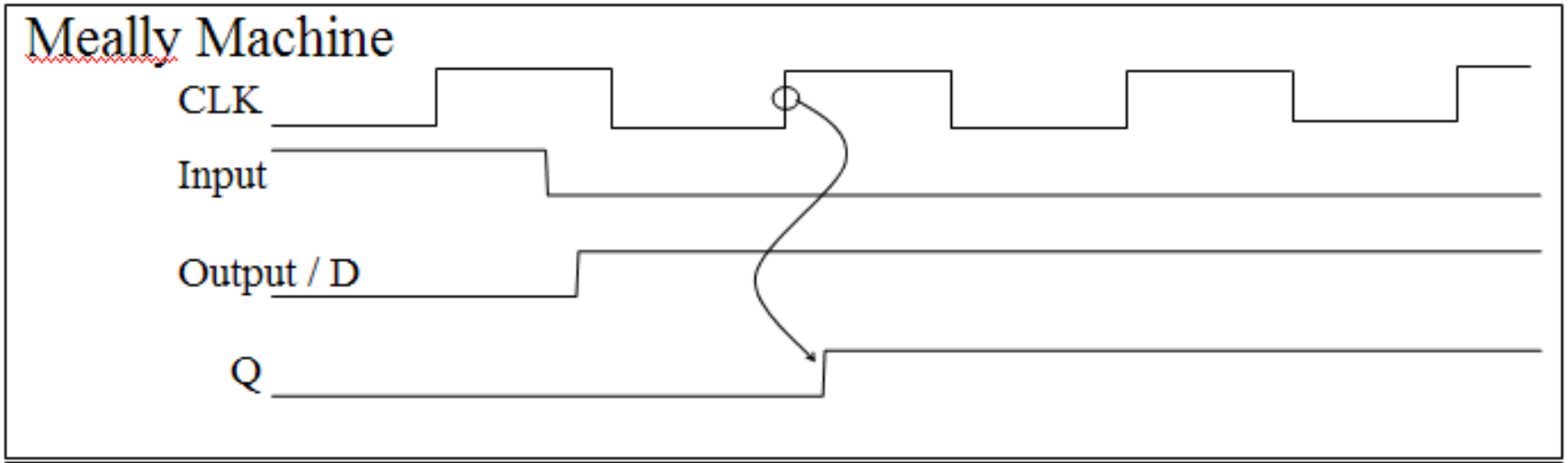
- Pewniejsze działanie
- Zmiana wyjścia zsynchronizowana z sygnałem zegarowym (z opóźnieniem 1 taktu)
- Wolniejsze działanie
- Przerwanie ścieżki sygnału od wejścia do wyjścia układu
- Zwykle bardziej rozbudowana część wyjściowa
- Większa liczba stanów i przerzutników

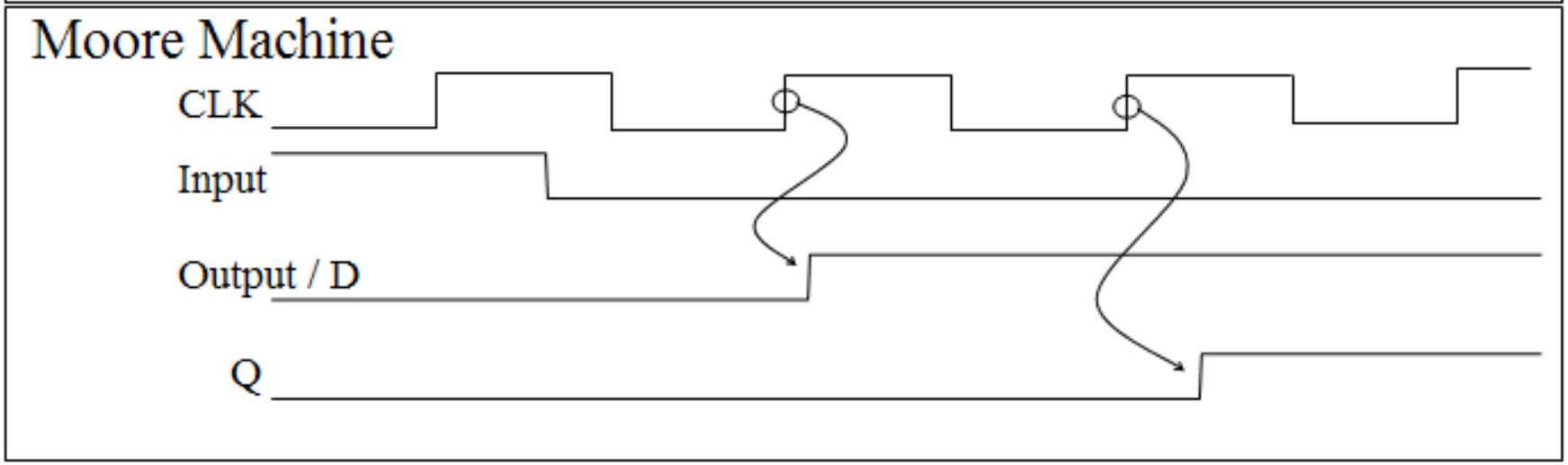
Automaty Moore'a





Mealy vs. Moore





Procedura projektowania automatów synchronicznych

1. Diagram stanów (graf przejść)
2. Minimalizacja liczby stanów wewnętrznych
3. Tablica przejść-wyjść
4. Kodowanie stanów wewnętrznych
5. Tablica wzbudzeń przerzutników
6. Funkcje wzbudzeń przerzutników
   - zminimalizuj układ korzystając z map Karnaugh
7. Funkcje wyjść
8. Schemat układu

Potencjalne problemy projektowania FSM

1. Nadmiarowość kodowania stanów wewnętrznych – niewłaściwa inicjalizacja

2. Nadmiarowość kodowania stanów wewnętrznych – możliwe pętle w obrębie stanów nieużywanych

3. Nadmiarowość kodowania stanów wewnętrznych – konieczność sprowadzenia do stanu właściwego

4. Jednoczesna zmiana co najmniej dwóch bitów stanu wewnętrznego – wyścigi krytyczne i niekrytyczne (a. asynchroniczne).

Kodowanie stanów układu

- **Każdemu z m-stanów układu należy przypisać unikalny identyfikator**
- **Min. długość identyfikatora n wynosi:**

$$n = \text{ceil}(\log_2 m)$$

- **Wygodnie jest czasami opisywać stany słowami dłuższymi niż n-bitowe, pozostaje wówczas (2^n-m)-stanów nieużywanych.**

Kodowanie stanów FSM

Dla m-stanów wewnętrznych FSM i n-bitów kodu ($m \geq n \geq \log_2 m$) liczba możliwych kodów wynosi:

$$N_K = \frac{2^n!}{(2^n - m)!}$$

Ex.: n = 3, m = 4, N_K = 1680

Przykłady kodowania stanów

1. Kodowanie proste (porządkowe):
   000, 001, 010, 011, ...

2. Kodowanie w kodzie Graya:
   000, 010, 011, 010, ...

3. Kodowanie losowe (random):
   010, 111, 011, 101, ...

4. Kodowanie z pojedynczym gorącym bitem (one-hot):
   0001, 0100, 0010, 0001

Kodowanie z pojedynczym gorącym bitem

1. n-stanów kodowanych przy pomocy n-przerzutników, np.:

Porządkowe → 00, 01, 10, 11
One-hot → 0001, 0010, 0100, 1000

2. Łatwa kontrola poprawności działania bloku przerzutników

3. Skomplikowany diagram stanów → złożony układ

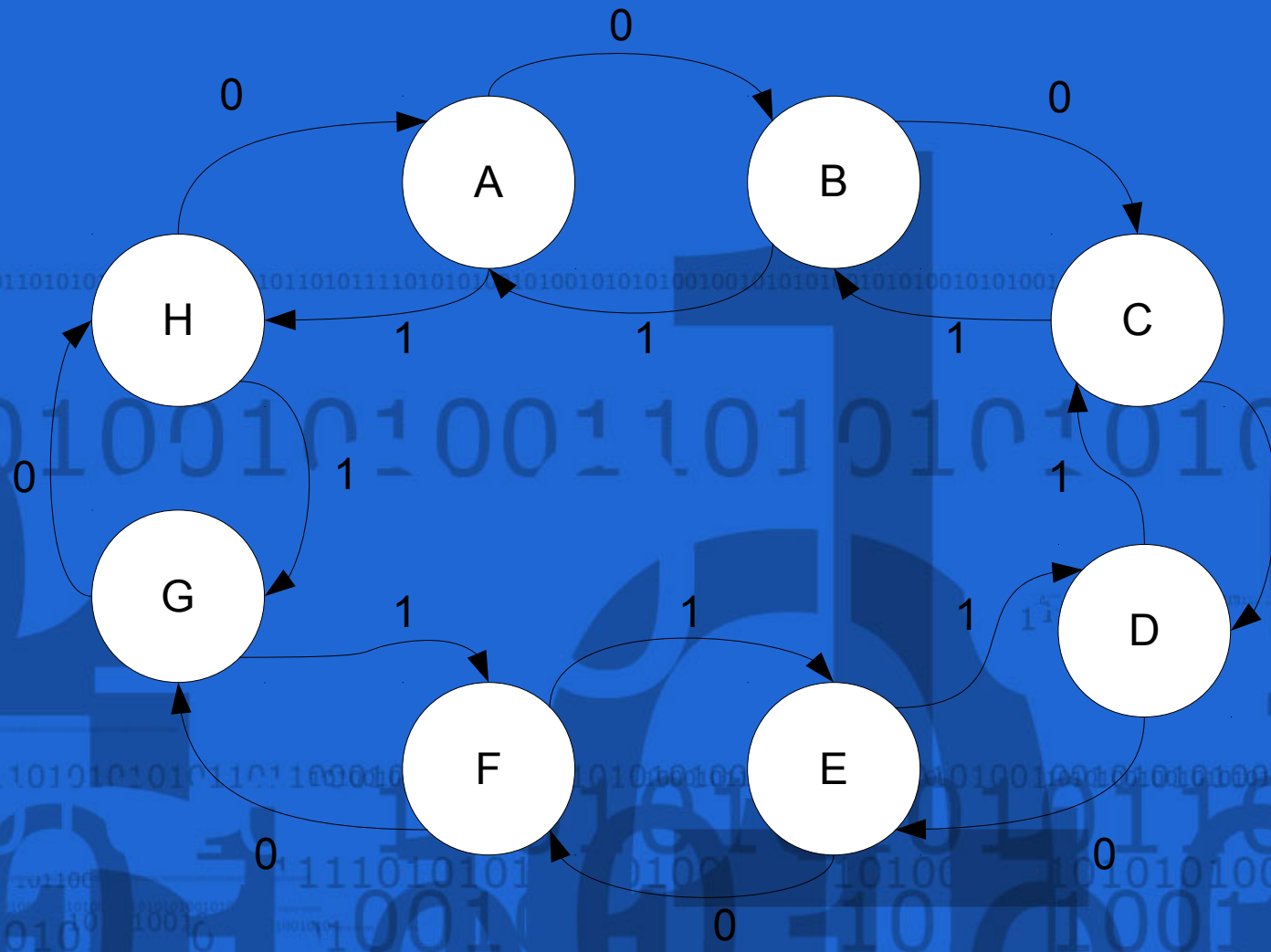
4. Rozbicie układu na 2 podukłady daje (n+m)-przerzutników zamiast (n*m).

Kodowanie z pojedynczym gorącym bitem

- Zwykle układ działa szybciej. FSM z dużą liczbą silnie zakodowanych stanów spowalniają swoje działanie.
- Prostota projektu
- Prostota modyfikacji działania
- Łatwa kontrola działania
- Łatwe diagnozowanie usterek

3-bitowy licznik binarny ze zmianą kierunku zliczania (0 FWD/1 REV)

Diagram stanów:



3-bitowy licznik binarny ze zmianą kierunku zliczania (0 FWD/1 REV)

Tablica przejść-wyjść (Moore):

Q	X=0	X=1	Y
A	B	H	000
B	C	A	001
C	D	B	010
D	E	C	011
E	F	D	100
F	G	E	101
G	H	F	110
H	A	G	111

3-bitowy licznik binarny ze zmianą kierunku zliczania (0 FWD/1 REV)

Tablica przejść-wyjść (Mealy):

Q	Q^+		Y^+	
	X=0	X=1	X=0	X=1
A	B	H	001	111
B	C	A	010	000
C	D	B	011	001
D	E	C	100	010
E	F	D	101	011
F	G	E	110	100
G	H	F	111	101
H	A	G	000	110

3-bitowy licznik binarny ze zmianą kierunku zliczania (0 FWD/1 REV)

Kodowanie stanów:

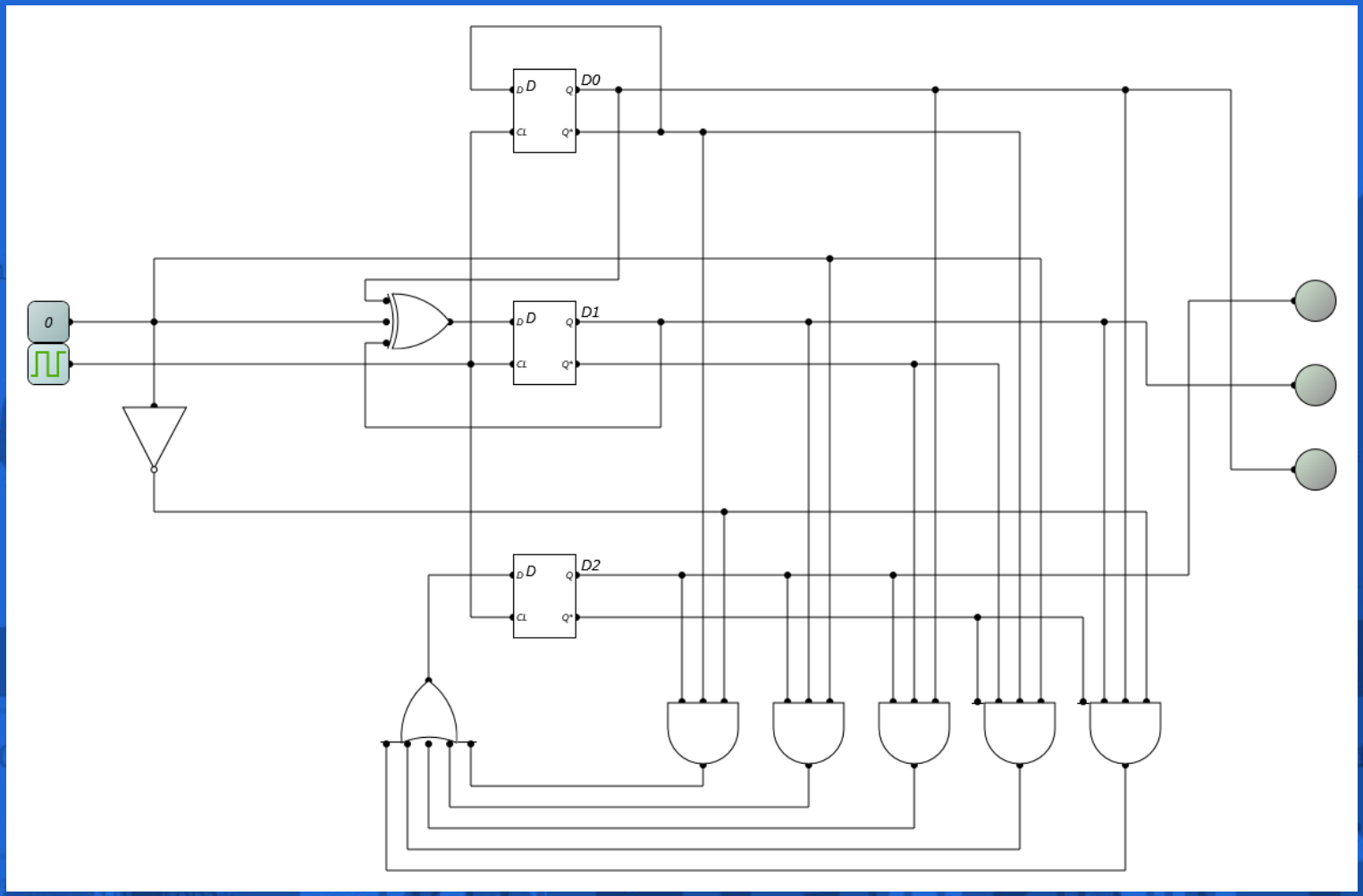
1. Kodowanie proste – brak bloku wyjść, ale większa wrażliwość na zakłócenia stanów wewnętrznych

2. Kodowanie w kodzie Graya – większa stabilność układu, ale dodatkowo logika wyjściowa

3. Kodowanie z pojedynczym 'gorącym bitem' (One-Hot Assignment) – największa nadmiarowość, ale możliwość kontroli działania

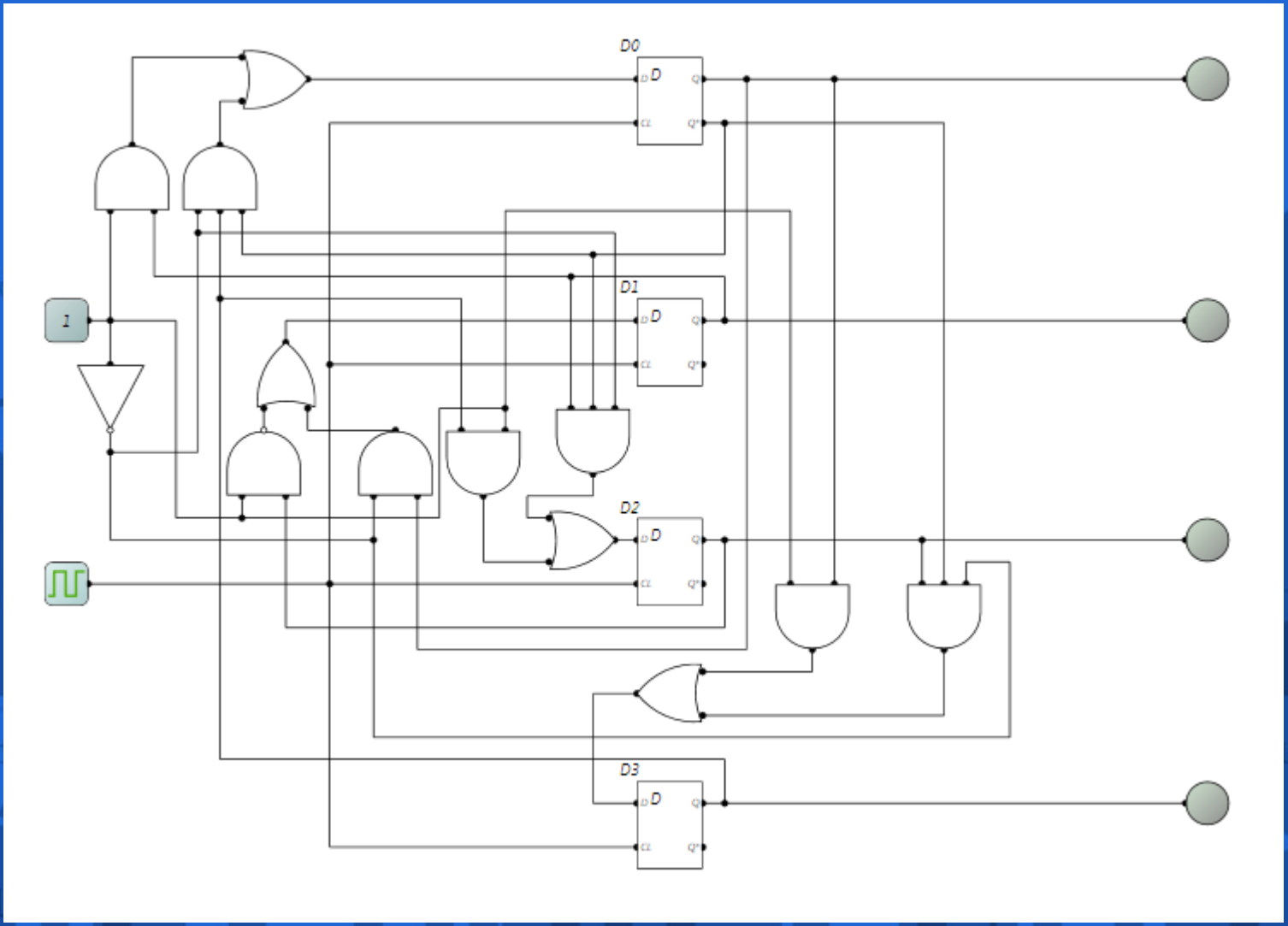
3-bitowy licznik binarny ze zmianą kierunku zliczania (0 FWD/1 REV)

Schemat układu (proste kodowanie stanów):



3-bitowy licznik binarny ze zmianą kierunku zliczania (0 FWD/1 REV)

Schemat układu (one-hot):



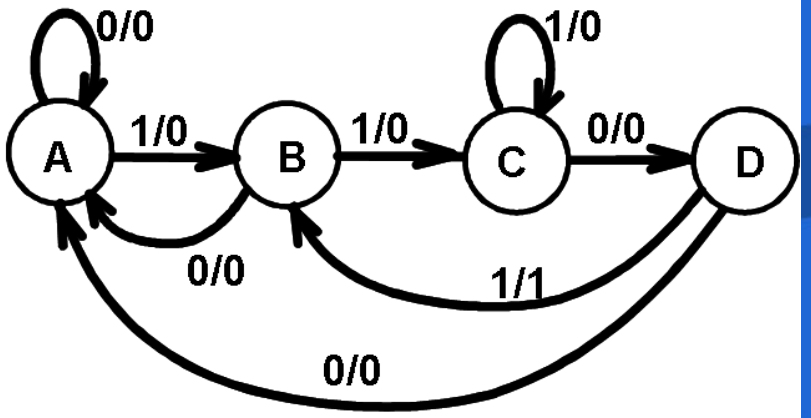
Detektor sekwencji '1101'

- **Przykład: detektor ciągu 1101**
  - **Zauważmy, że np. ciąg 1111101 zawiera 1101, zaś podciągiem jest np. "11".**
- **Zatem, detektor musi pamiętać wystąpienie dwóch jedynek przed nadejściem kolejnego znaku.**
- **Ponadto, ciąg 1101101 zawiera 1101 jako podciąg początkowy i końcowy nałożone na siebie, tj. 1101101 lub 1101101.**
- **Wspólnym znakiem obu podciągów jest środkowa 1, 1101101.**
- **Ciąg 1101 musi być wykrywany za każdym razem, gdy pojawi się w ciągu wejściowym.**

Detektor sekwencji '1101' – Mealy

- **Kompletna tabela stanów.**

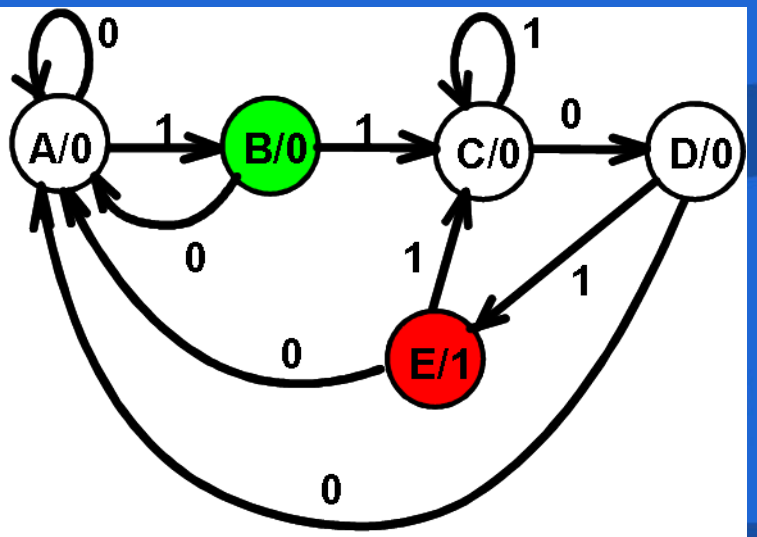
Present State	Next State x=0	Next State x=1	Output x=0	Output x=1
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	A	B	0	1



- **Jak wyglądałby diagram i tabela dla modelu Moore'a?**

Detektor sekwencji '1101' - Moore
„Moore is more”

Present State	Next State x=0	Next State x=1	Output y
A	A	B	0
B	A	C	0
C	D	C	0
D	A	E	0
E	A	C	1



Przypisanie stanów – przykład 1

Stan bieżący	Stan kolejny		Wyjście	
	x=0	x=1	x=0	x=1
A	A	B	0	0
B	A	B	0	1

- **Jakie są możliwości opisania stanów układu słowami binarnymi przy minimalnej liczbie bitów?**
  - **2 sposoby:** $A = 0, B = 1$ **lub** $A = 1, B = 0$

Przypisanie stanów – przykład 2

Stan bieżący	Stan kolejny x=0	Stan kolejny x=1	Wyjście x=0	Wyjście x=1
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	A	B	0	1

- **Ilość możliwych przypisań stanów:**
  - **4! = 24**
- **Czy sposób przypisania wpływa na koszt układu?**

Przypisanie stanów – przykład 2

- **Przypisanie porządkowe:**

  A = 0 0, B = 0 1, C = 1 0, D = 1 1

- **Wynikowa tablica stanów:**

Present State	Next State x = 0	Next State x = 1	Output x = 0	Output x = 1
0 0	0 0	0 1	0	0
0 1	0 0	1 0	0	0
1 0	1 1	1 0	0	0
1 1	0 0	0 1	0	1

Przypisanie stanów – przykład 2

- **Przypisanie w kodzie Graya:**

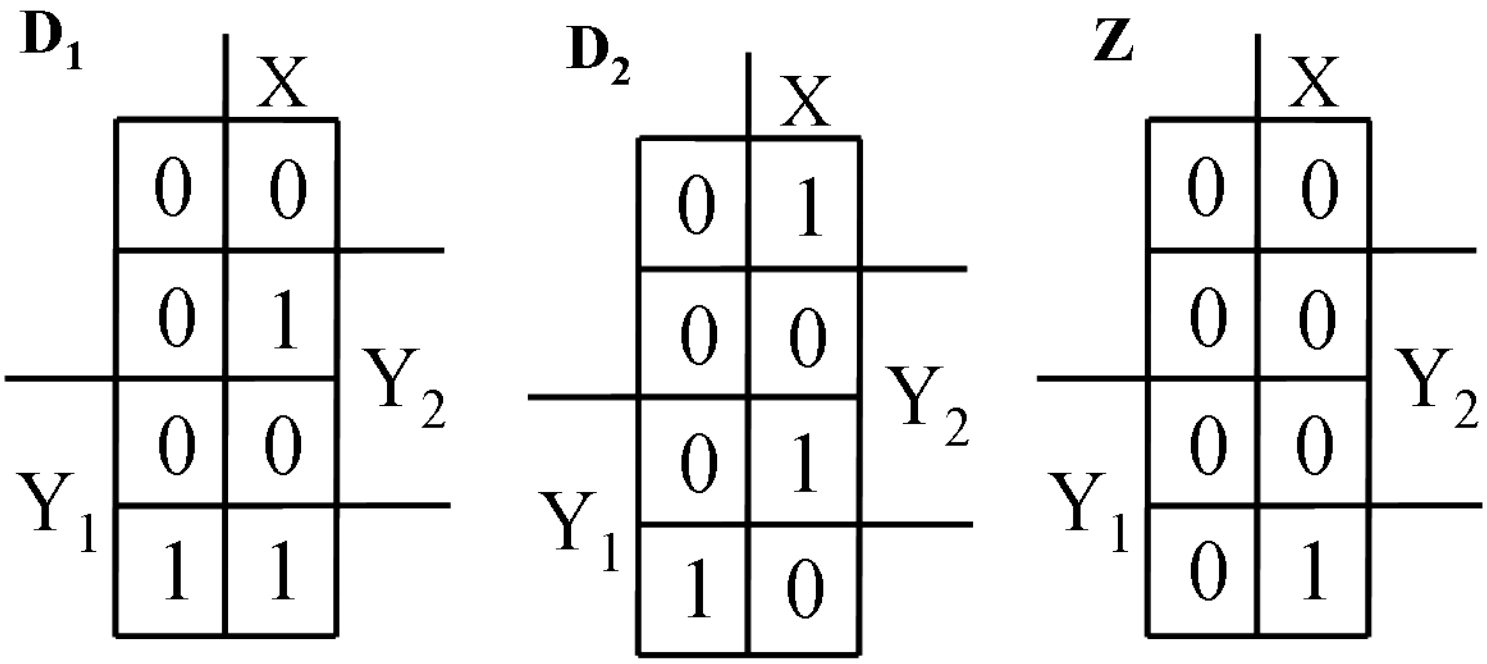
  A = 0 0, B = 0 1, C = 1 1, D = 1 0

- **Wynikowa tablica stanów:**

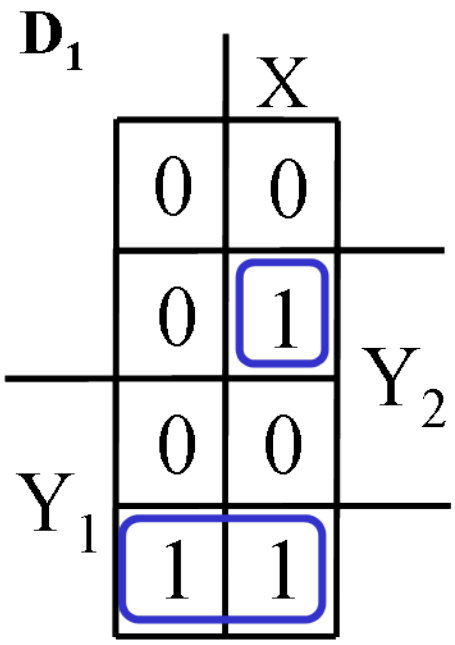
Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
0 0	0 0	0 1	0	0
0 1	0 0	1 1	0	0
1 1	1 0	1 1	0	0
1 0	0 0	0 1	0	1

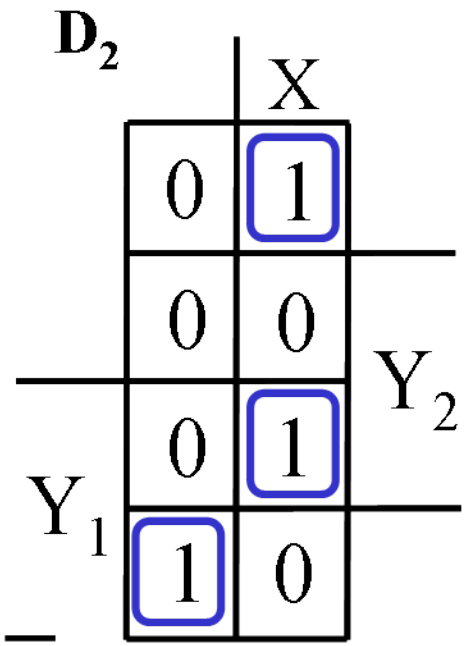
Poszukiwanie równań wej/wyj przerzutników (przykład 2)

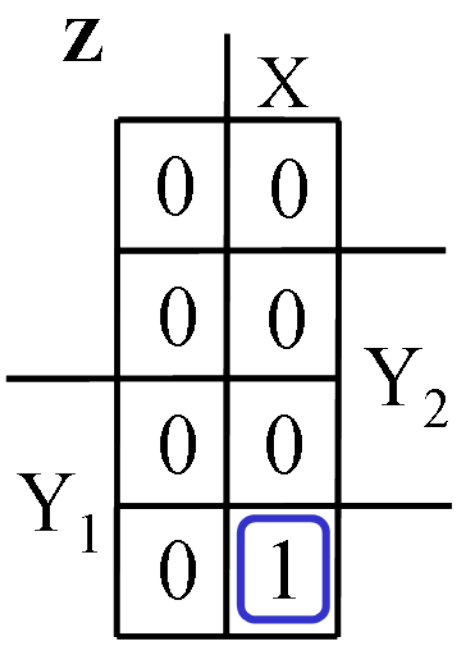
- **Przypisanie porządkowe, przerzutniki D**
- **Mapy Karnaugha dla D_1, D_2 i Z:**



Optymalizacja przypisania porządkowego (przykład 2)







$$D_1 = Y_1\bar{Y}_2 + X\bar{Y}_1Y_2$$

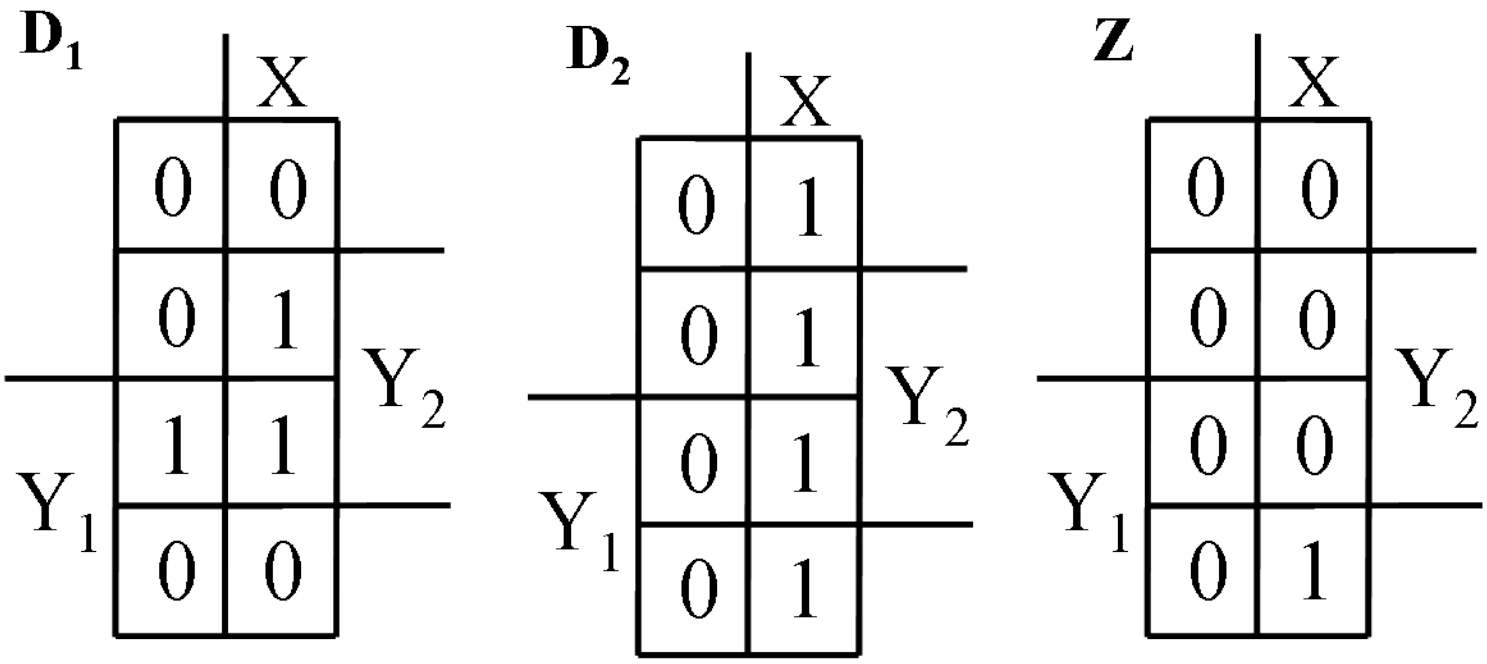
$$D_2 = \bar{X}Y_1\bar{Y}_2 + X\bar{Y}_1\bar{Y}_2 + XY_1Y_2$$

$$Z = XY_1\bar{Y}_2$$

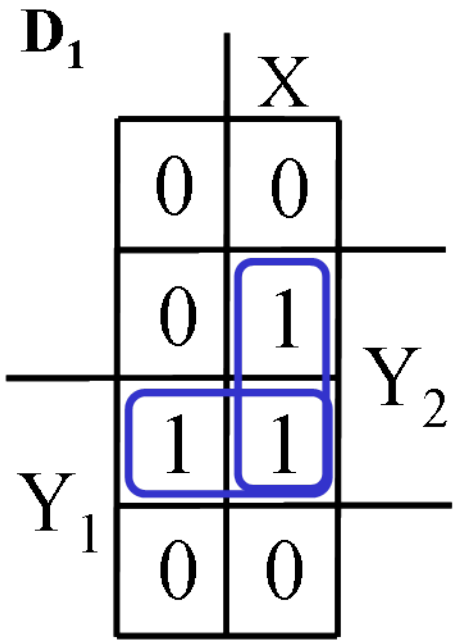
Gate Input Cost = 22

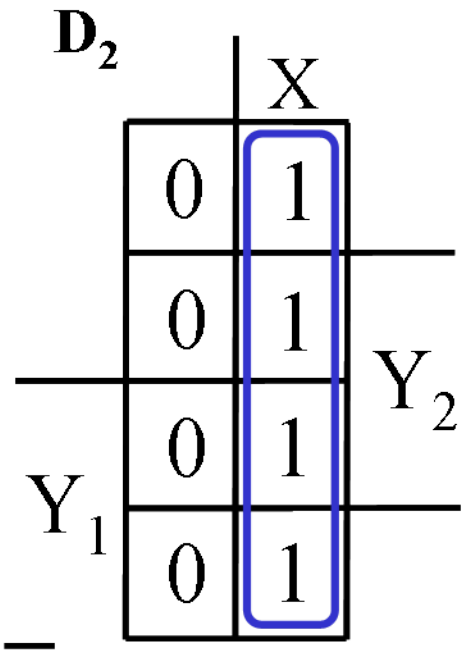
Poszukiwanie równań wej/wyj przerzutników (przykład 2)

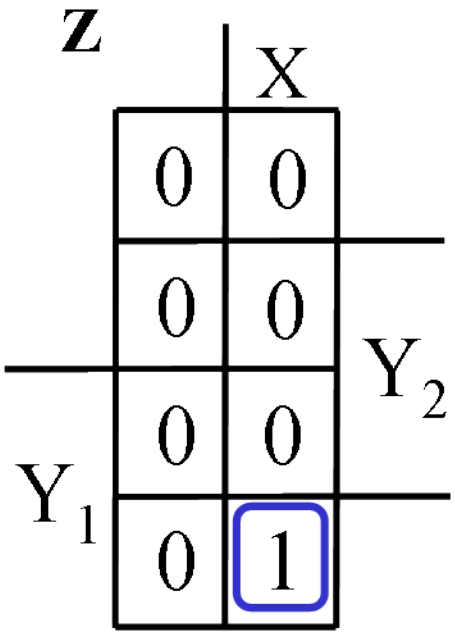
- **Przypisanie w kodzie Graya, przerzutniki D**
- **Mapy Karnaugha dla D_1, D_2 i Z:**



Optymalizacja przypisania w kodzie Graya (przykład 2)







$$D_1 = Y_1 Y_2 + X\bar{Y}_2$$

$$D_2 = X$$

$$Z = XY_1\bar{Y}_2$$

Gate Input Cost = 9

Przypisanie „1 przerzutnik na stan” (One-Hot Assignment)

- **Przykładowe przypisanie dla 4 stanów:**

$$(Y_3, Y_2, Y_1, Y_0) = 0001,\ 0010,\ 0100,\ 1000.$$

- **W równaniach wystarczy wypisać tylko zmienną przyjmującą dla stanu wartość 1, np. stan 0001 jest opisany przez Y_0 zamiast $\overline{Y_3}\,\overline{Y_2}\,\overline{Y_1}\,Y_0$, gdyż wszystkie kody z 0 lub co najmniej dwiema 1 posiadają wartości „don’t care” w następnym stanie.**
- **Logika kombinacyjna może być prostsza, ale koszt przerzutników wyższy – końcowy koszt trudny do oszacowania.**

Przypisanie „1 przerzutnik na stan” (One-Hot Assignment)

- **One-Hot Assignment:**

  A = 0001, B = 0010, C = 0100, D = 1000

Wynikowa tablica stanów:

Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
0001	0001	0010	0	0
0010	0001	0100	0	0
0100	1000	0100	0	0
1000	0001	0010	0	1

Optymalizacja „1 przerzutnik na stan” (One-Hot Assignment)

- **Równania na podstawie położeń 1 w kolumnie następnego stanu tabeli stanów:**

$$D_0 = \overline{X}(Y_0 + Y_1 + Y_3) \text{ or } X\,\overline{Y}_2$$

$$D_1 = X(Y_0 + Y_3)$$

$$D_2 = X(Y_1 + Y_2) \text{ or } X(\overline{Y_0 + Y_3})$$

$$D_3 = \overline{X}\,Y_2$$

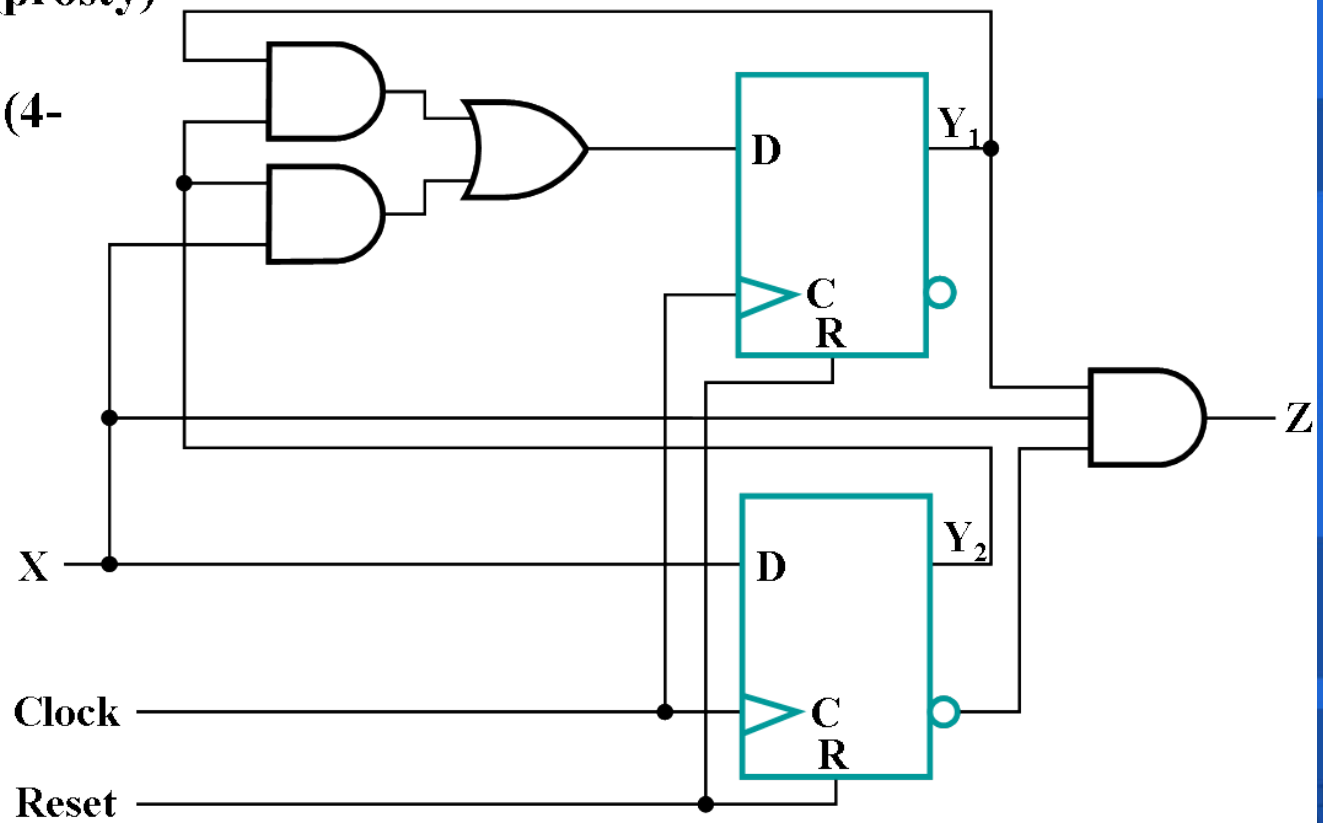
$$Z = XY_3$$

Gate Input Cost = 15

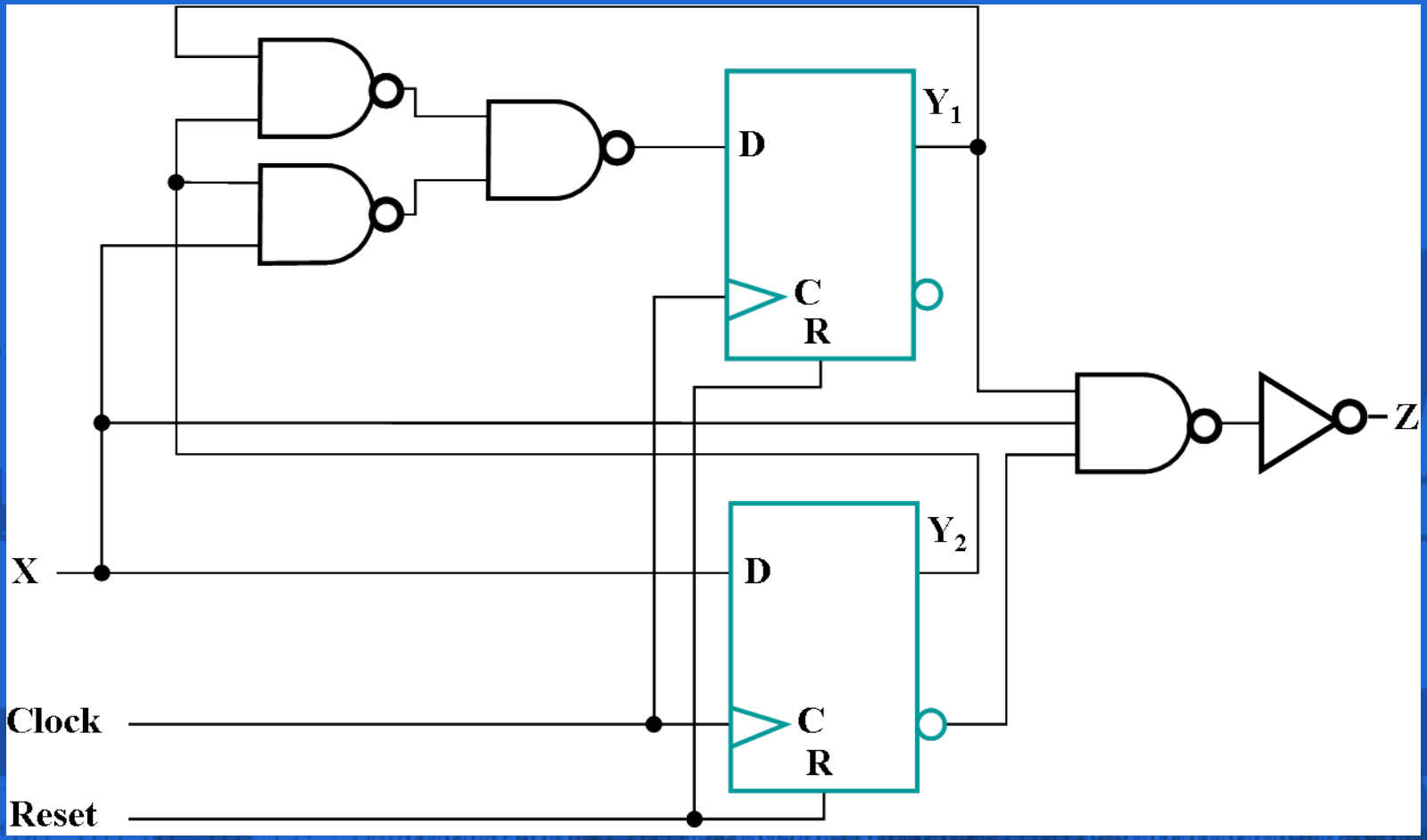
- **Pośredni koszt logiki kombinacyjnej plus koszt przerzutników.**

Realizacja „1 przerzutnik na stan” (One-Hot Assignment)

- **Dostępne elementy:**
  - **Przerzutnik D (prosty) z resetem**
  - **Bramki NAND (4-wejściowe) z inwerterami**
- **Układ pierwotny:**



Końcowa postać układu

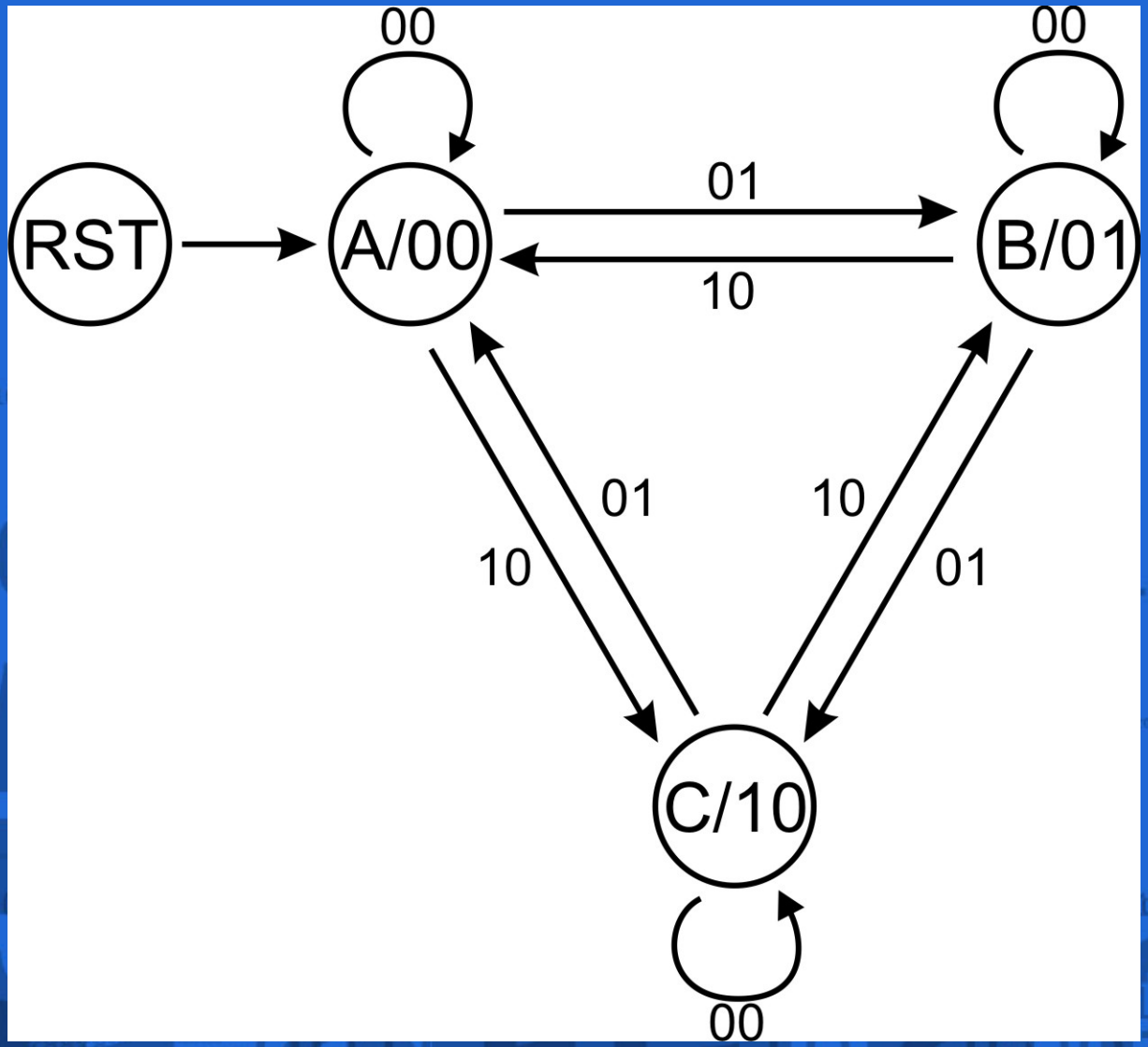


Sekwencyjny akumulator modulo 3

- **Cel: projekt 2-bitowego, sekwencyjnego akumulatora modulo 3**
- **Definicje:**
  - **Sumator modulo *n* – sumator, który wynik przedstawia jako resztę z dzielenia modulo *n***
    - **$(2 + 2) \bmod 3 = 1$**
  - **Akumulator – układ kumulujący słowa pojawiające się na wejściu przez dodanie bieżącej wartości wejścia do bieżącego wyniku na wyjściu; wyjście początkowo równe jest 0.**
- **Oznaczenie zmiennych: suma skumulowana (Y_1,Y_0), wejście (X_1,X_0), wyjście (Z_1,Z_0)**

Sekwencyjny akumulator modulo 3

Diagram stanów - Moore



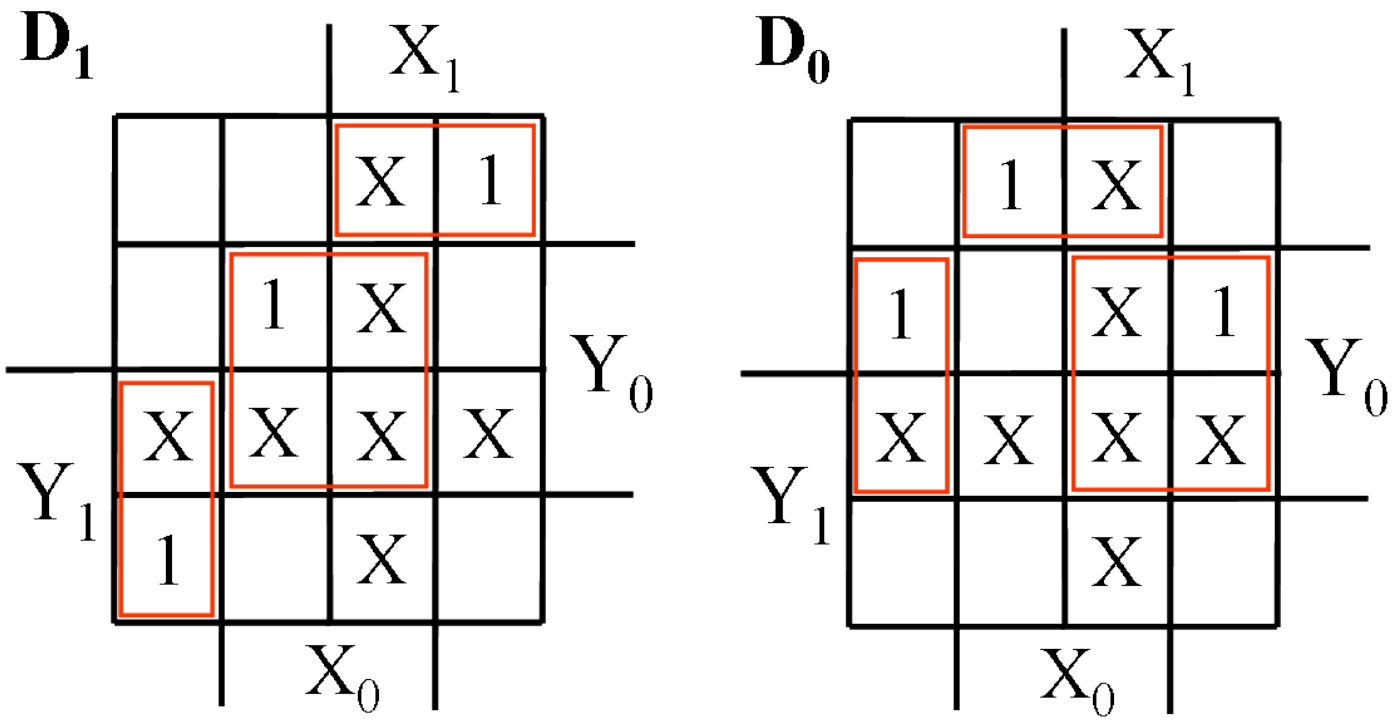
Sekwencyjny akumulator modulo 3
Tablica stanów

Y_1Y_0 \ X_1X_0	00	01	11	10	Z_1Z_0
	$Y_1(t+1)$, $Y_0(t+1)$	$Y_1(t+1)$, $Y_0(t+1)$	$Y_1(t+1)$, $Y_0(t+1)$	$Y_1(t+1)$, $Y_0(t+1)$	
A (00)	00	01	X	10	00
B (01)	01	10	X	00	01
- (11)	X	X	X	X	11
C (10)	10	00	X	01	10

- Przypisanie stanów: $(Y_1,Y_0) = (Z_1,Z_0)$
- Opisanie stanów w kodzie Graya

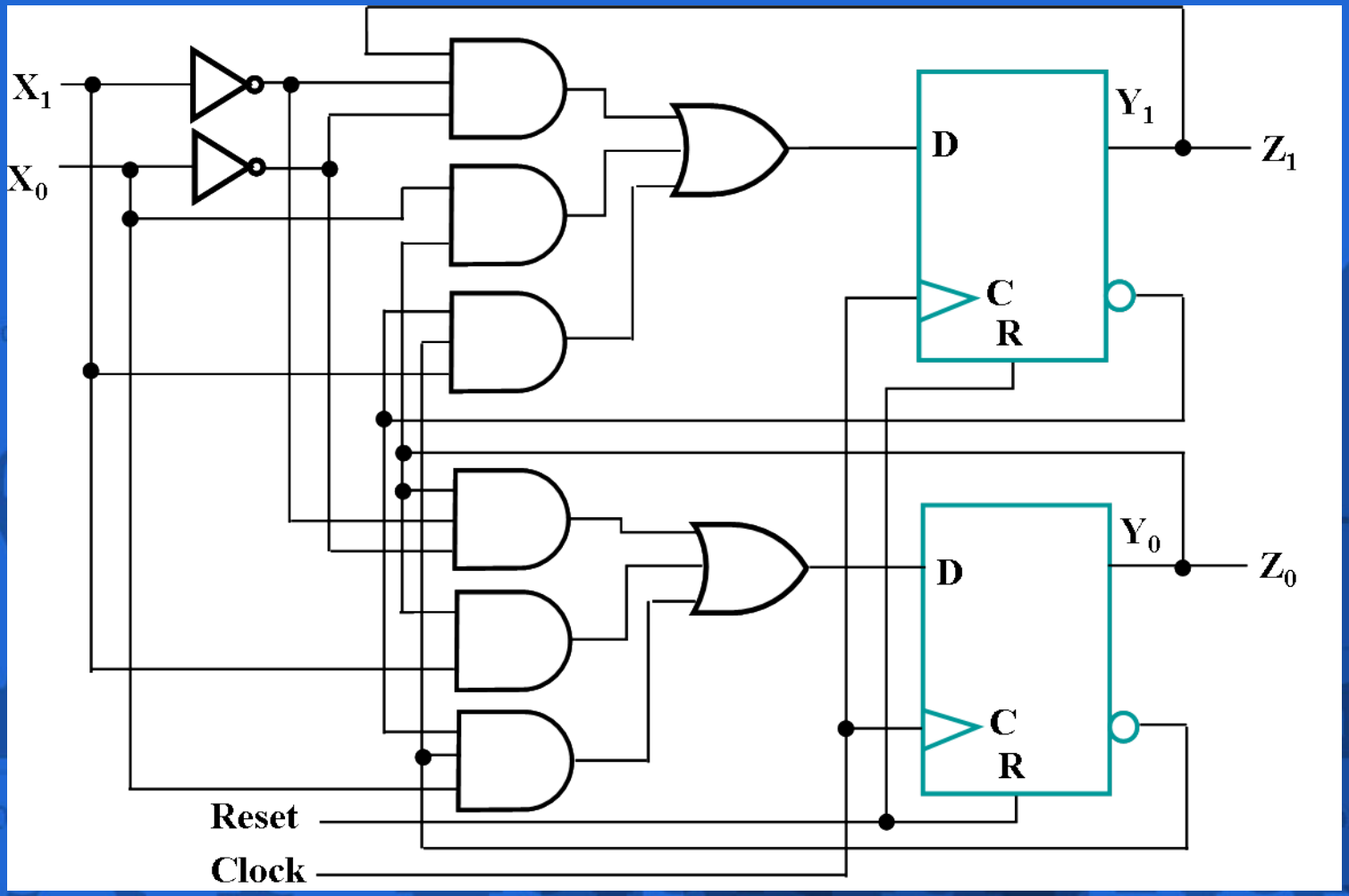
Sekwencyjny akumulator modulo 3
Równania wejściowe przerzutników D

- **Równania wejściowe przerzutników D:**



- $D_1 = \overline{X}_1\overline{X}_0Y_1\overline{Y}_0 + \overline{X}_1X_0\overline{Y}_1Y_0 + X_1\overline{X}_0\overline{Y}_1\overline{Y}_0 = \overline{X}_1\overline{X}_0Y_1 + X_0Y_0 + X_1\overline{Y}_1\overline{Y}_0$
- $D_0 = \overline{X}_1X_0\overline{Y}_1Y_0 + \overline{X}_1X_0\overline{Y}_1\overline{Y}_0 + X_1\overline{X}_0\overline{Y}_1Y_0 = \overline{X}_1\overline{X}_0Y_0 + X_1Y_0 + X_0\overline{Y}_1\overline{Y}_0$

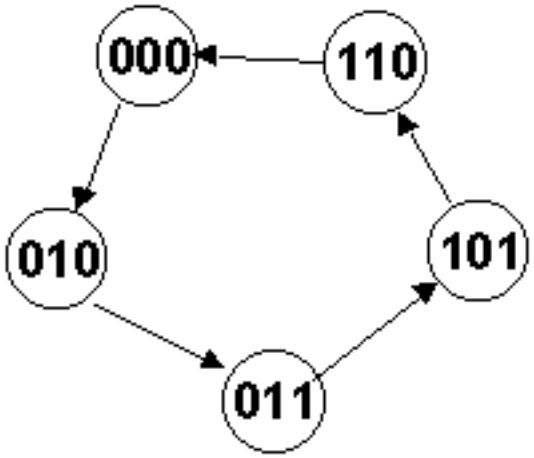
Sekwencyjny akumulator modulo 3
Schemat układu



Automaty niepełne

Należy z ostrożnością projektować automaty, w których liczba stanów wewnętrznych jest mniejsza niż 2^n, gdyż rodzą one problemy z właściwą inicjalizacją.

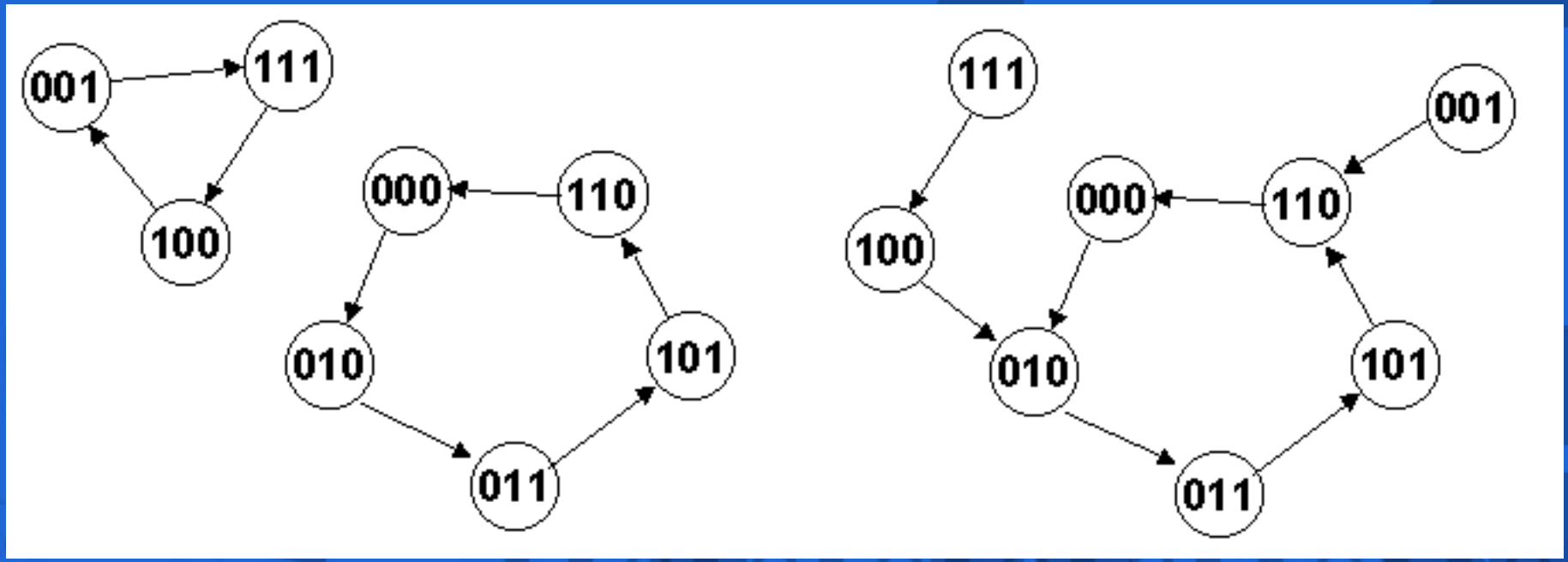
Licznik niepełny



Present State			Next State		
C	B	A	C+	B+	A+
0	0	0	0	1	0
0	0	1	–	–	–
0	1	0	0	1	1
0	1	1	1	0	1
1	0	0	–	–	–
1	0	1	1	1	0
1	1	0	0	0	0
1	1	1	–	–	–

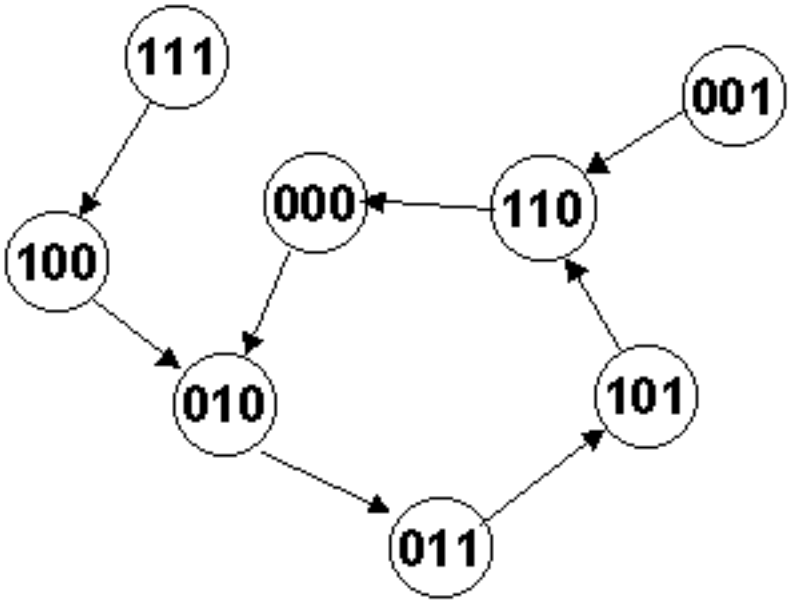
Problem niewłaściwej inicjalizacji

Należy zadbać, aby układ nie zapętlił się lub nie wpadł w niedozwolone stany wewnętrzne przy niewłaściwej inicjalizacji.



Automaty samostartujące

Taki projekt zapewnia niewrażliwość na niewłaściwą inicjalizację, aczkolwiek może ograniczyć elastyczność minimalizacji.
Rozwiązanie alternatywne (niesamostartujące) – asynchroniczne ustawianie automatu w stanie początkowym.



Present State			Next State		
C	B	A	C+	B+	A+
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	1	1	0	0