Architektura komputerów
Wykład 2

Jan Kazimirski

Elementy techniki cyfrowej

Plan wykładu

- Algebra Boole'a
- Podstawowe układy cyfrowe – bramki
- Układy kombinacyjne
- Układy sekwencyjne

Algebra Boole'a

- Stosowana do projektowania i analizy cyfrowych układów logicznych
- Nazwa pochodzi od nazwiska brytyjskiego matematyka – George Boole'a, który zaproponował podstawowe zasady tej algebry
- Wykorzystanie w technice cyfrowej – Claude Shannon (1938)

Algebra Boole'a c.d.

- Posiada przynajmniej 2 elementy (oznaczymy je jako 0 i 1).
- Posiada zdefiniowane operacje
  - Sumy (suma logiczna) '+'
  - Iloczynu (iloczyn logiczny) '*'
  - Negacja '-' lub „~'

Suma logiczna

- Operator sumy logicznej (OR)
- Operator 2-argumentowy
- Tabela prawdy:

p	q	p+q
0	0	0
0	1	1
1	0	1
1	1	1

Iloczyn logiczny

- Operator iloczynu logicznego
- Operator 2-argumentowy
- Tabela prawdy:

p	q	p*q
0	0	0
0	1	0
1	0	0
1	1	1

Negacja

- Operator negacji
- Operator 1-argumentowy
- Tabela prawdy:

p	~p
0	1
1	0

Funkcja XOR

- Różnica symetryczna
- Operator 2-argumentowy
- Tablica prawdy:

p	q	p^q
0	0	0
0	1	1
1	0	1
1	1	0

Aksjomaty algebry Boole'a

- Element neutralny: $x+0=x$, $x*0=0$
- Uzupełnienie: $x+(\sim x)=1$, $x*(\sim x)=0$
- Przemienność: $x+y=y+x$, $x*y=y*x$
- Łączność: $(x+y)+z=x+(y+z)$, $(x*y)*z=x*(y*z)$
- Rozdzielność: $x*(y+z)=(x*y)+(x*z)$ $x+(y*z)=(x+y)*(x+z)$

Własności algebry Boole'a

- x+x=x, x*x=x
- x+1=1, x*0=0
- Prawa de Morgana:
  ~(x+y)=(~x)*(~y), ~(x*y)=(~x)+(~y)
- x+(x*y) = x, x*(x+y)=x
- ~(~x)=x

Wyrażenia i funkcje logiczne

- **Wyrażenie logiczne** – ciąg zer, jedynek i symboli logicznych rozdzielonych operatorami

$$(x * 1)+(x*y)*(\sim(x*z))$$

- **Funkcja logiczna** – funkcja zmiennych logicznych

$$F(x,y,z)$$

Tabela prawdy

a	b	c	F=a * b * c
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Funkcje logiczne c.d.

- Dla danej liczby argumentów można stworzyć nieskończenie wiele wyrażeń logicznych
- Dla danej liczby argumentów (N) istnieje określona liczba funkcji logicznych

  N=2 - 16 funkcji
  N=3 - 256 funkcji
  N=4 - 65536 funkcji

Zapis funkcji logicznej

1. Odpowiednie wyrażenie logiczne
2. Tabela prawdy
3. Określenie zbiorów argumentów dla których wartość funkcji wynosi 1 lub 0

Ćwiczenie

Zapisać funkcję logiczną opisywaną wyrażeniem **f(x,y,z) = (x+y)*(~(y*z))** w postaci tabeli prawdy oraz w postaci zbioru wartości x,y,z, dla których wartość funkcji wynosi 1.

Funkcje logiczne 2 zmiennych

0. Zero – zawsze zwraca zero
1. Logiczne NOR – (a+b)'
2. Zakaz a*b'
3. Not b – niezależnie od a zwraca b'
4. Zakaz b*a'
5. Not a – niezależnie od b zwraca a'
6. XOR – a różne od b
7. NAND – (a*b)'
8. AND
9. NOR wyłączające (NOT XOR)
10. Kopia a – zwraca zawsze a
11. Implikacja – z b wynika a – a+b'
12. Kopia b – zwraca zawsze b
13. Implikacja – z a wynika b – b+a'
14. OR
15. Jeden – zawsze zwraca jedynkę

Formy kanoniczne

- Funkcja logiczna może być przedstawiona w postaci wyrażenia logicznego na nieskończenie wiele sposobów.
- W celu standaryzacji wprowadzono formy kanoniczne funkcji logicznej:
  - Suma termów minimalnych
  - Iloczyn termów maksymalnych

Formy kanoniczne c.d.

- Suma termów minimalnych
  - Term minimalny – zmienna lub iloczyn kilku zmiennych wejściowych
  - Dla zmiennych a i b: a, b, a', b', ab, a'b, ab', a'b'
  - Kanoniczną postać funkcji - suma termów minimalnych.
- Iloczyn termów maksymalnych
  - Term maksymalny – suma wszystkich zmiennych wejściowych (zanegowanych lub nie)
  - Kanoniczną postać funkcji - iloczyn termów maksymalnych
  - Np. (a+b+c)*(a'+b+c)*(a+b'+c)

Minimalizacja zapisu funkcji logicznej

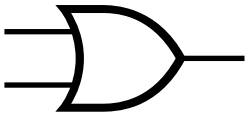
- Problem często spotykany w technice cyfrowej
- Dana funkcja logiczna (forma kanoniczna lub tabela prawdy). Należy zapisać ją jak najprościej
- Metody minimalizacji:
  - Metoda algebraiczna
  - Metoda map Karnaugha
  - Algorytm Quina - McCluskeya

Podstawowe układy cyfrowe – bramki

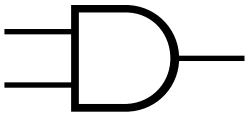
- Wykorzystywane do realizacji funkcji przetwarzania w komputerze.
- Realizują podstawowe operacje logiczne.
- Zbudowane są z nich bardziej zaawansowane elementy cyfrowe

Podstawowe bramki logiczne

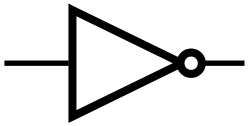
- Bramka OR F= a+b



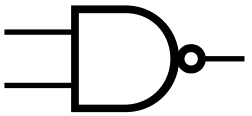
- Bramka AND F= a*b



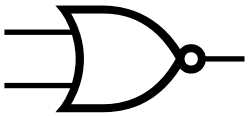
- Bramka NOT F=a'



- Bramka NAND F=(a*b)'

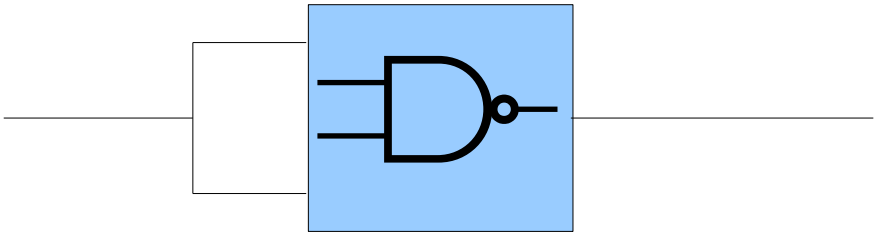


- Bramka NOR F=(a+b)'

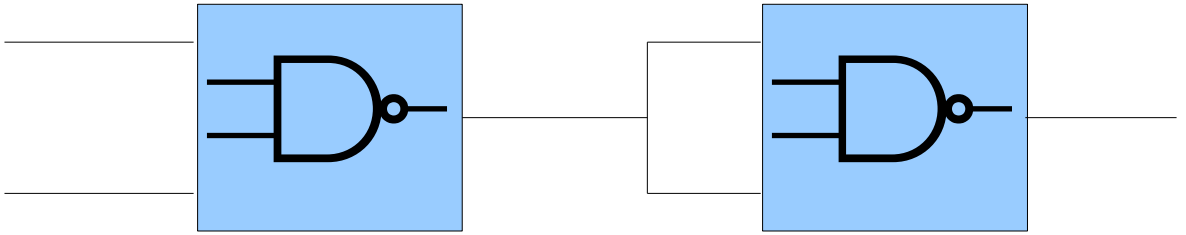


Realizacja funkcji logicznych za pomocą bramki NAND

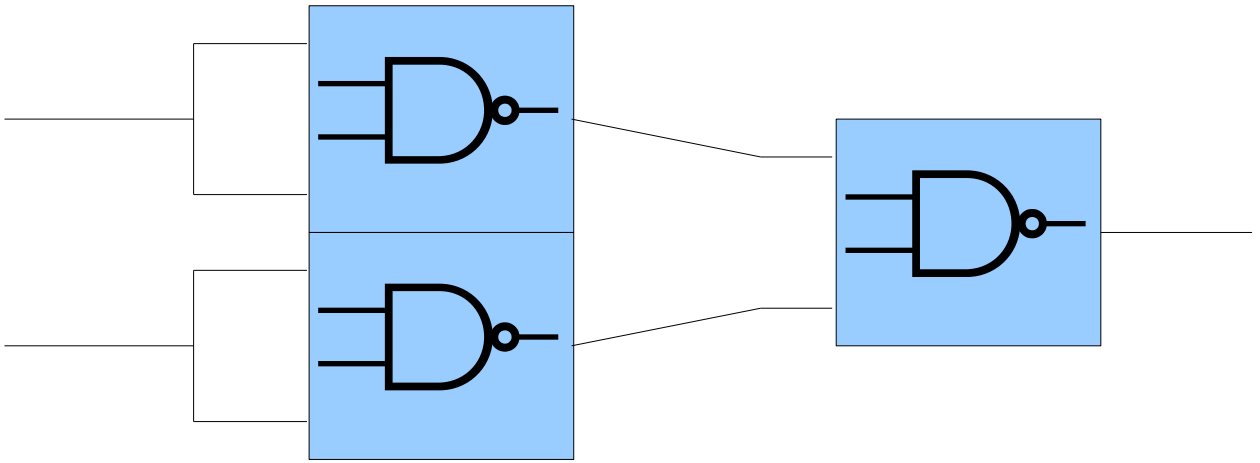
- Funkcja NOT



- Funkcja AND



- Funkcja OR



Ćwiczenie

Za pomocą tabeli prawdy pokazać, że układy pokazane na poprzednim slajdzie realizują funkcje NOT, AND i OR

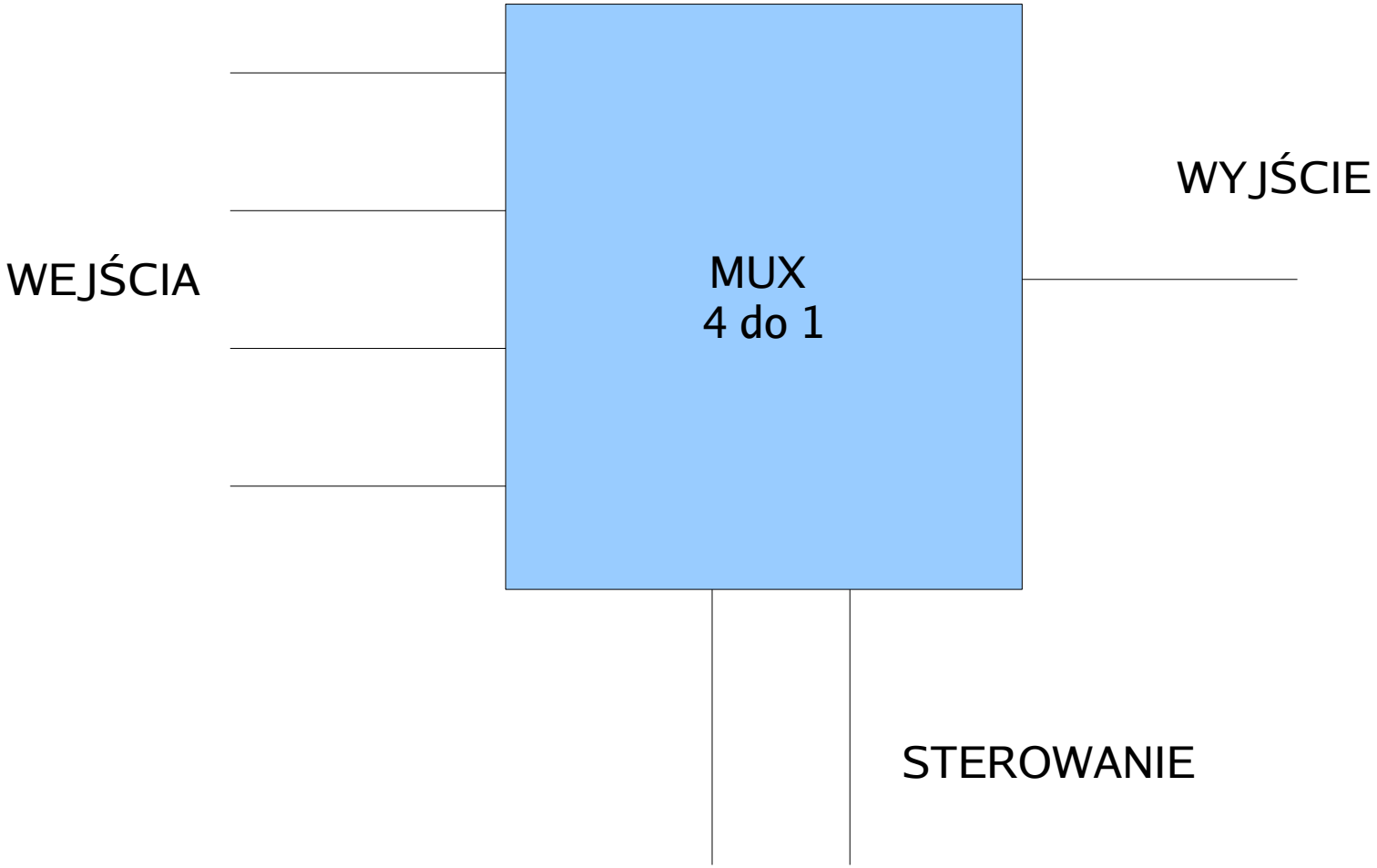
Układy kombinacyjne

- Zbiór wzajemnie połączonych bramek
- Zawiera N wejść i M wyjść binarnych
- Stan wyjść zależy tylko i wyłącznie od stanu wejść
- Opis układu:
  - Tablica prawdy
  - Schemat graficzny połączeń bramek
  - Funkcja Boole'a

Multiplekser

- Układ posiadający wiele wejść (D0...Dn), jedno wyjście F, oraz linie sterujące (S0..Sm).
- Na wyjście dostarczany jest sygnał z jednego wybranego wejścia
- Wartość linii sterujących decyduje które wejście zostanie powielone na wyjściu.
- Używany do sterowania przepływem sygnałów i danych

Multiplexer



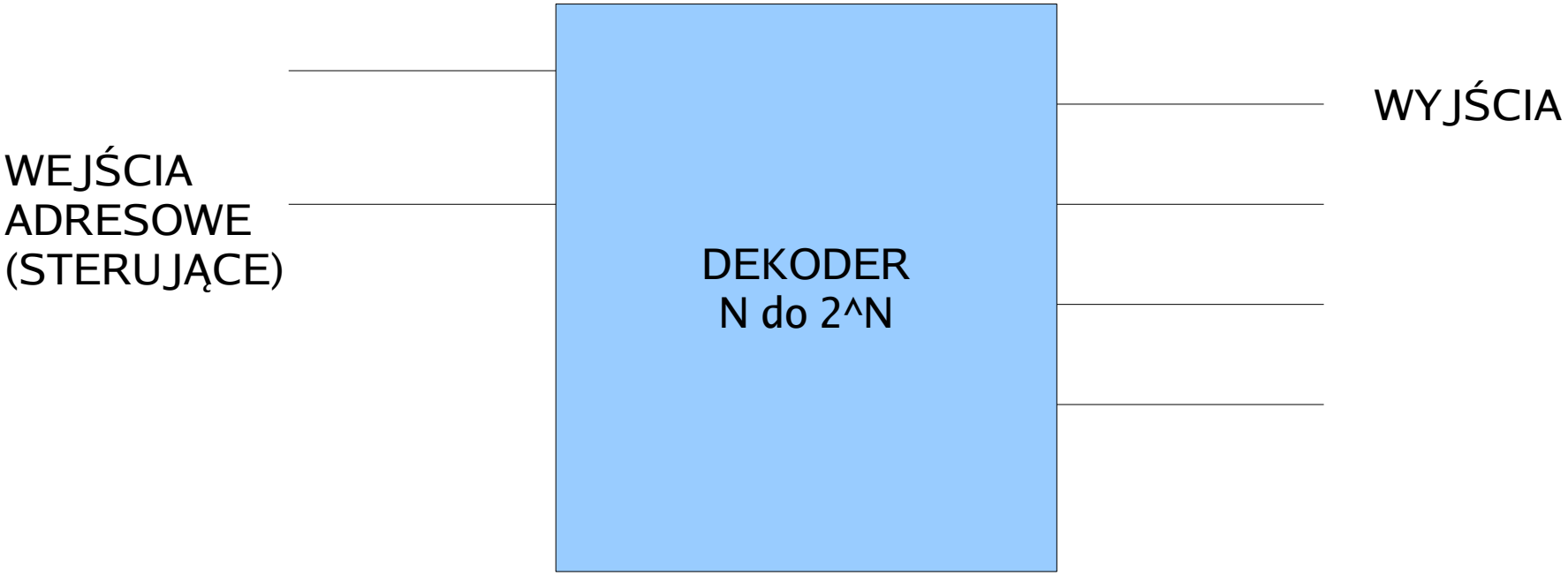
Multiplexer

S0	S1	F
0	0	D0
0	1	D1
1	0	D2
1	1	D3

Dekoder

- Posiada N wejść sterujących i 2^N wyjść.
- Zależnie od kombinacji stanów wejść sygnał wysyłany jest jedno określone wyjście
- Stosowany do sterowania wyświetlaczami LED, dekodowania adresów pamięci lub instrukcji maszynowych

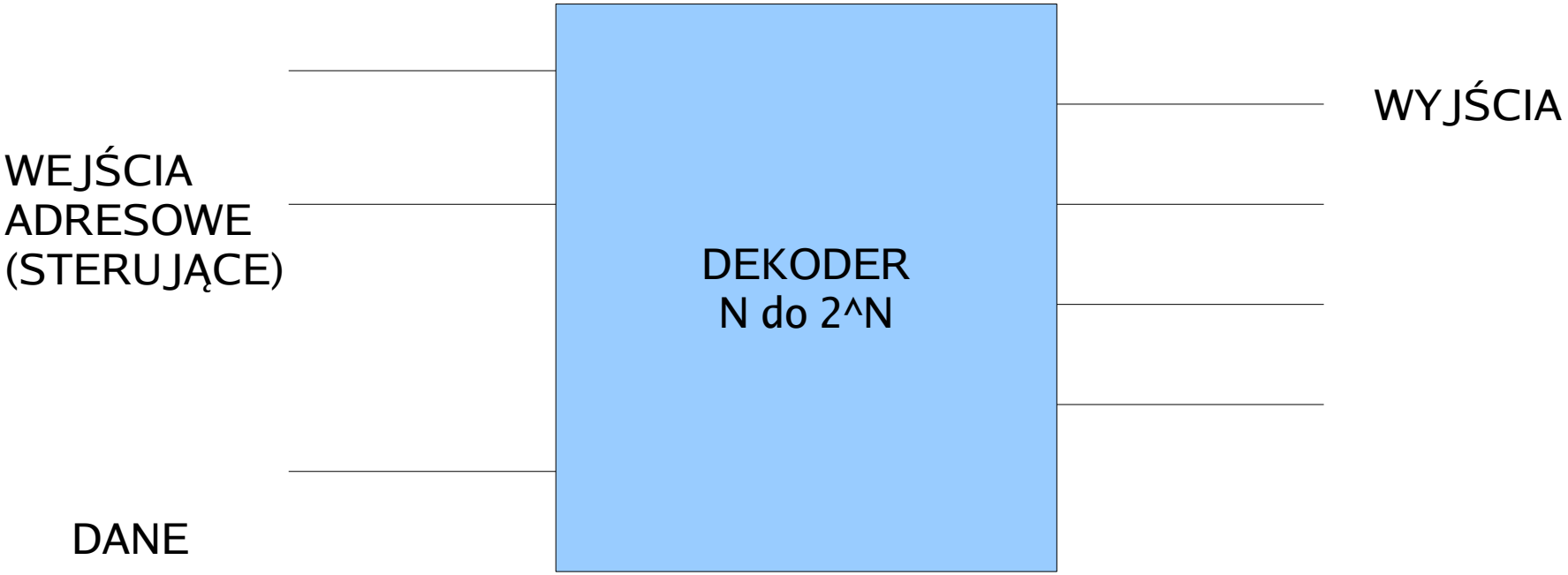
Dekoder



Demultiplexer

- Dekoder z dodatkowym wejściem danych
- Zależnie od kombinacji stanów sygnałów sterujących, dane przekazywane są do wybranego wyjścia
- Linia danych może być również stosowana do aktywacji układu (np. przez impuls zegarowy)

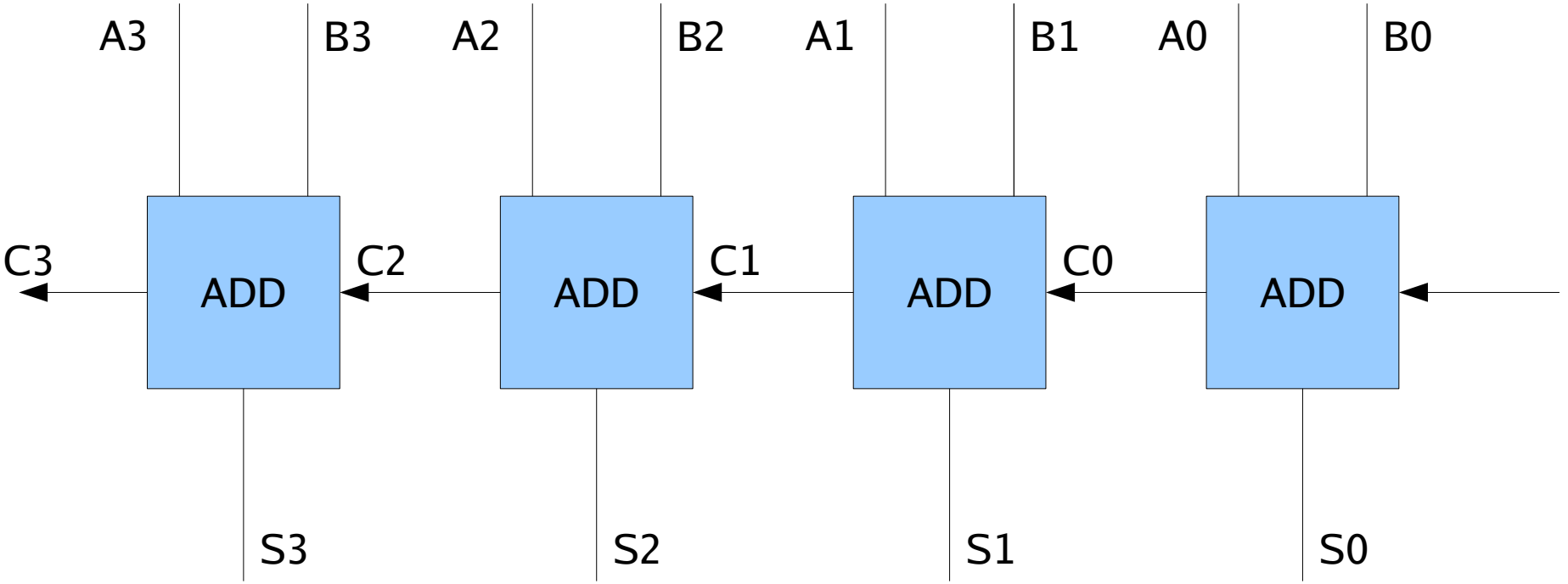
Demultiplexer



Półsumator i Sumator

- Półsumator
  - Dwie linie wejściowe
  - Dwie linie wyjściowe – suma, oraz przeniesienie
- Sumator
  - Do linii wejściowych dodana jeszcze jedna (suma 3-bitowa) – przeniesienie z poprzedniej sumy bitowej

Sumator 4-bitowy



Układy sekwencyjne

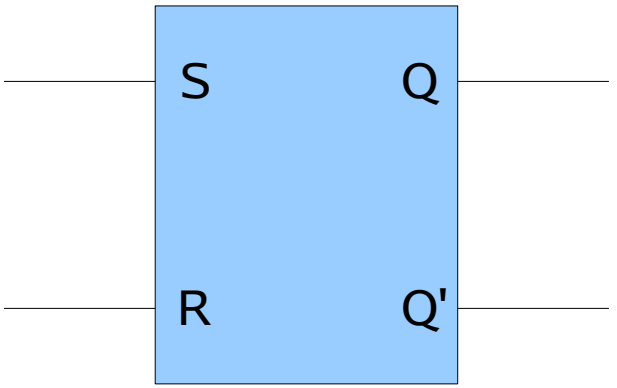
- Układy „z pamięcią” - układy których aktualny stan zależy od stanu wejść oraz poprzedniego stanu układu.
- Przykłady:
  - Przerzutniki
  - Rejestry
  - Liczniki

Przerzutniki

- Układy dwustabilne – mogą utrzymywać się w jednym z dwóch stanów.
- Sygnał wejściowy zmienia stan przerzutnika
- Przykład „komórki pamięci”
- Zwykle mają dwa wyjścia – Q i Q' (zanegowane Q)

Przerzutnik S/R

S	R	Q(t+1)
0	0	Q(T)
0	1	0
1	0	1
1	1	?



Przerzutnik D

- Układ opóźniający
- Ma tylko jedno wejście danych oraz wejście sygnału zegarowego
- Układ „zapamiętuje” bit danych z wejścia
- Stosowany np. do realizacji rejestrów lub liczników

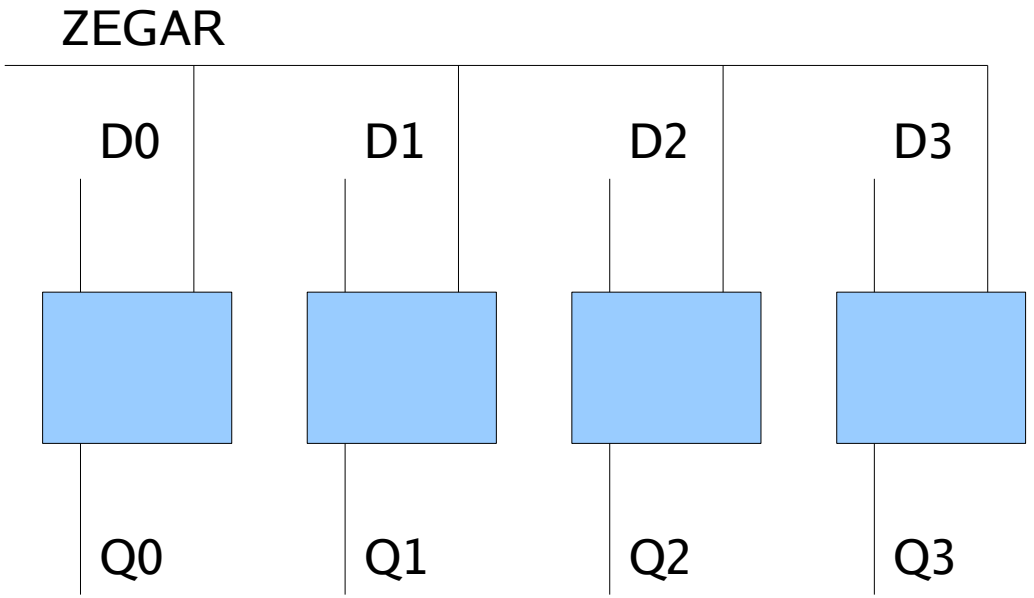
Przerzutnik J/K

- Podobny do przerzutnika S/R
- Dwa wejścia informacyjne J,K
- Sygnał zegarowy (praca synchroniczna)
- Może mieć asynchroniczne wejścia R,S
- Wyjścia Q i Q'
- Wejście K – kasujące, J - „jedynkujące”
- Stany wysokie J,K – negowanie Q

Rejestry

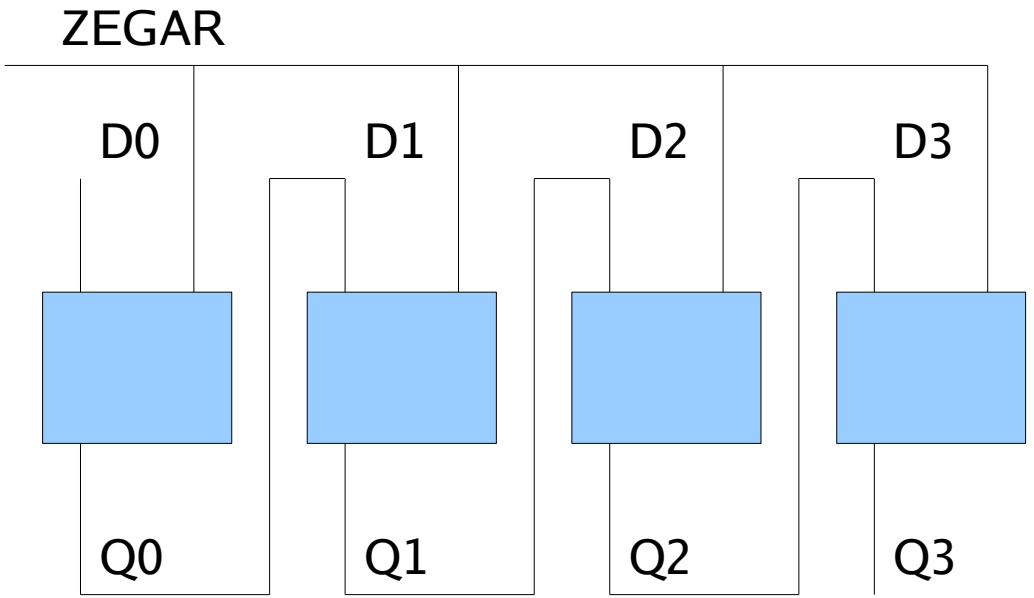
- Służą do przechowywania danych we wnętrzu procesora
- Jedno- lub wielobitowe
- Typy rejestrów:
  - Równoległe
  - Przesuwne

Rejestr równoległy



4-bitowy rejestr równoległy zrealizowany za pomocą przerzutników typu D

Rejestr przesuwny



4-bitowy rejestr przesuwny zrealizowany za pomocą przerzutników typu D

Liczniki

- Rejestr, którego zawartość może być zwiększana o jeden
- Po osiągnięciu maksymalnej wartości licznik jest zerowany
- Przykład: licznik rozkazów (PC) w procesorze

Podsumowanie

- W technice cyfrowej wykorzystuje się algebrę Boole'a i operatory logiczne
- Podstawowym elementem techniki cyfrowej są bramki
- Z bramek budowane są bardziej złożone elementy cyfrowe – układy kombinacyjne i sekwencyjne