Pracownia fizyczna i elektroniczna

Wykład 4:
Cyfrowe układy scalone I

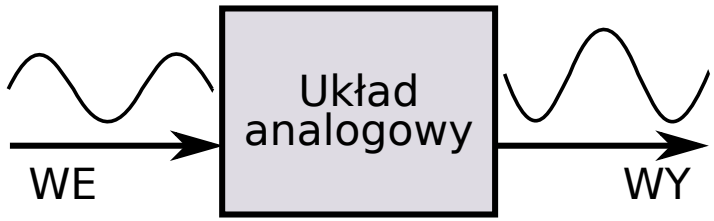
Kontakt: mkuich@fuw.edu.pl

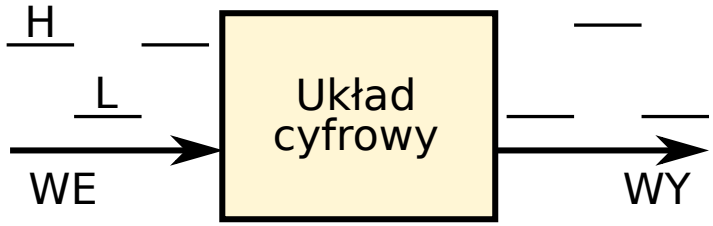
Materiały: Pracownia fizyczna i elektroniczna (1100-2F23,1100-2BF21)

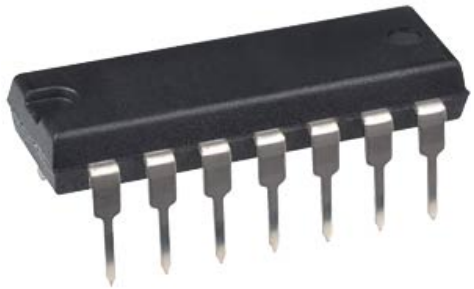
- kurs na platformie Kampus

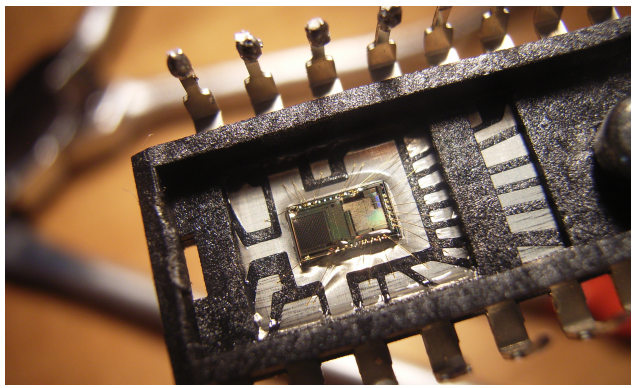
Analogowe i cyfrowe układy scalone

- **Układy analogowe** - przystosowane do przetwarzania napięć (lub prądów), których wartości zawierają się w pewnym przedziale wartości
- **Układy cyfrowe** - służą do przetwarzania sygnałów o dwóch wielkościach napięć (ewentualnie prądów): wysokiej (**H** - high) i niskiej (**L** - low)
- **Układ scalony** (ang. *integrated circuit*) stanowi fizycznie wykonany mikrominiaturowy układ elektroniczny, którego część lub wszystkie elementy i ich połączenia są wytworzone we wspólnym procesie technologicznym, wewnątrz lub na powierzchni wspólnego podłoża
  $\rightarrow$ funkcja układu scalonego określona w trakcie konstruowania i produkcji



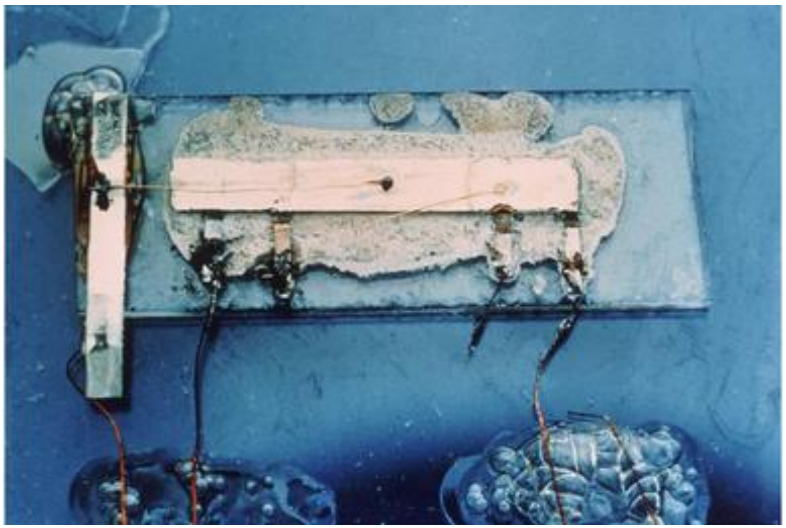


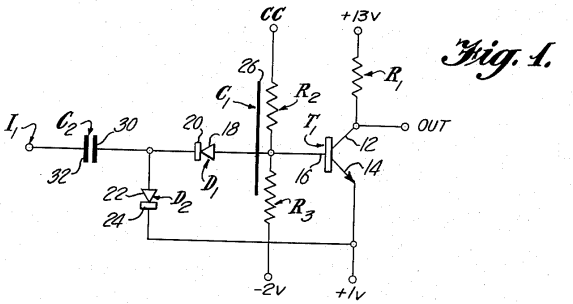




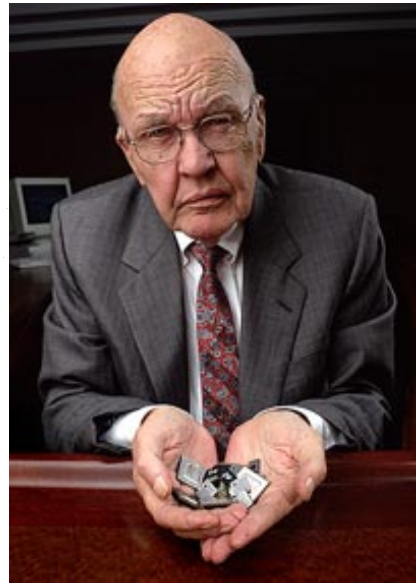
Pierwsze układy scalone

- 1958 r. - pierwszy (hybrydowy) układ scalony - multiwibrator astabilny - opatentowany przez Jack'a Kilby'ego





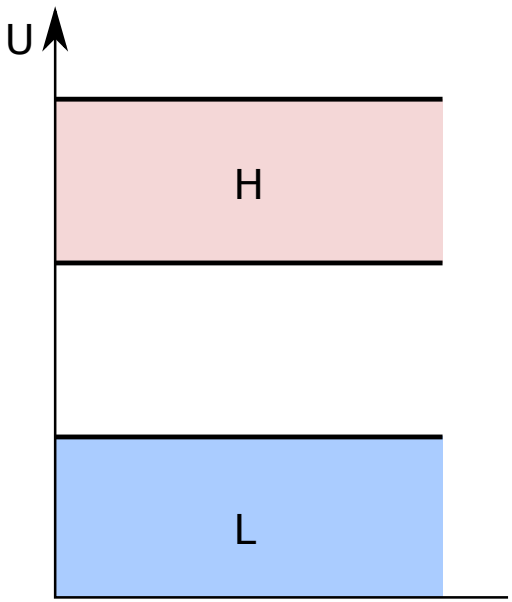
→ 1/2 Nagrody Nobla w 2000 r. za: „Za jego wkład w wynalezienie układu scalonego"



- 1959 r. - pierwszy (monolityczny) układ scalony opatentowany przez Roberta Noyce'a
  → opracował planarną technologię tworzenia układów scalonych wykorzystywaną do dziś

Stany w układach cyfrowych

- Pracę takich układów cyfrowych (układów logicznych) opisuje się za pomocą **dwuwartościowej algebry Boole'a** (logiki matematycznej)
- Poziomom napięć H i L przyporządkowuje się **wartości logiczne**: 1 (prawda) oraz 0 (fałsz):
  - przyporządkowanie H $\rightarrow$ 1 oraz L $\rightarrow$ 0 nazywa się **logiką dodatnią**
  - przyporządkowanie H $\rightarrow$ 0 oraz L $\rightarrow$ 1 nazywa się **logiką ujemną**

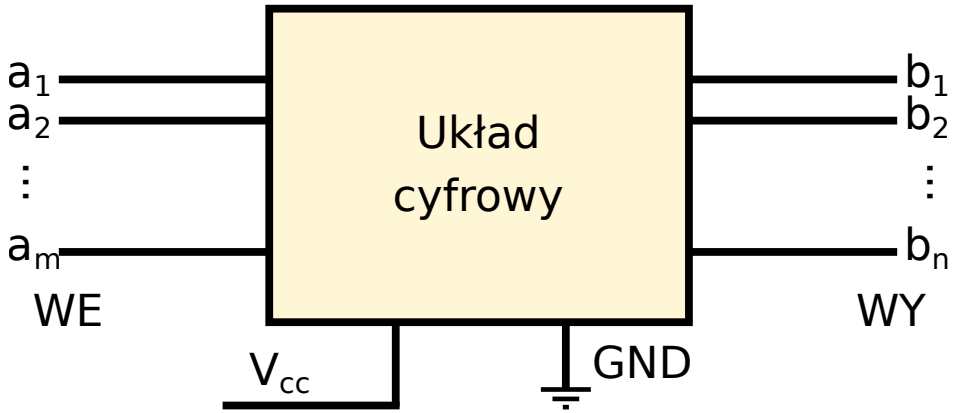


- Stany (sygnały) w układach cyfrowych nie mają ściśle określonych wartości napięcia
- Wartości logiczne przyporządkowuje się przedziałom napięć oddzielonych przerwami, np.:

stan	TTL	CMOS
H (1)	2.0 - 5.0 V	3.5 - 5.0 V
L (0)	0.0 - 0.8 V	0.0 - 1.5 V

- Jeżeli napięcie przyjmie wartość z zakresu przerwy to stan układu jest nieokreślony

Układ cyfrowy



- Aby układ cyfrowy działał poprawnie musimy podłaczyć go do zewnętrznego zasilania (V_{cc}) oraz poziom odniesienia (GND)
- W układach logicznych na każdym z wejść/wyjść może występować stan 0 lub 1 będący jednostką informacji zwaną **bitem**, a wektory $a = (a_1, a_2, ..., a_m)$ i $b = (b_1, b_2, ..., b_n)$ stanowią **słowa logiczne**; słowo ośmiobitowe nazywamy **bajtem**
- Słowa bitowe są interpretowane przez układ cyfrowy, jako określony kod (dwójkowy, Gray'a, itp.)
- W naturalnym kodzie binarnym (dwójkowym) każde słowo logiczne jest interpretowane jako pewna liczba, np. czterobitowe słowo (1101) w kodzie dziesiętnym jest liczbą:

$$1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 13$$

Klasyfikacja układów logicznych ze względu na sposób przetwarzania informacji

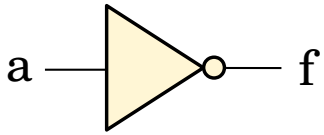
Układy kombinacyjne	Układy sekwencyjne
Stan wyjść jest jednoznacznie określony przez stan wejść układu: $b = f(a)$	Stan wyjść zależy od stanu wejść oraz od poprzednich stanów układu: $b(t_n) = f[a(t_1), a(t_2), ..., a(t_n)]$
Przykłady układów:	
• Bramki logiczne • Bloki funkcjonalne - multipleksery, demultipleksery - konwertery kodów - bloki arytmetyczne	• Przerzutniki • Liczniki • Rejestry • ...

Układy kombinacyjne

Bramki logiczne I

NOT (negacja):

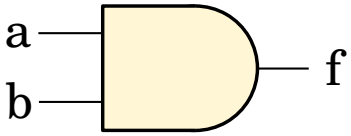
$f = \bar{a}$



a	f
0	1
1	0

AND (iloczyn):

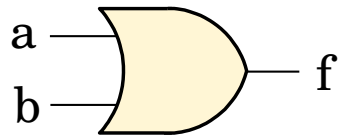
$f = a \cdot b$



a	b	f
0	0	0
0	1	0
1	0	0
1	1	1

OR (suma):

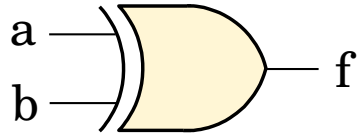
$f = a + b$



a	b	f
0	0	0
0	1	1
1	0	1
1	1	1

EXOR (Exclusive OR, wyłączna suma logiczna):

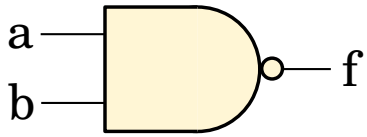
$f = a \oplus b$



a	b	f
0	0	0
0	1	1
1	0	1
1	1	0

NAND (NOT-AND, negacja iloczynu):

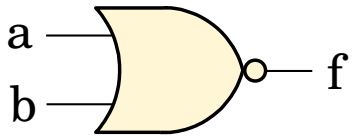
$f = \overline{a \cdot b}$



a	b	f
0	0	1
0	1	1
1	0	1
1	1	0

NOR (NOT-OR, negacja sumy):

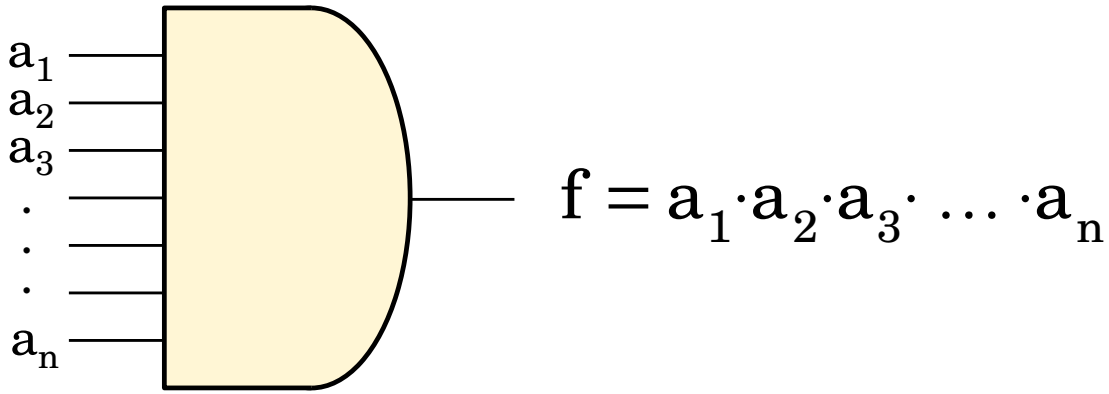
$f = \overline{a + b}$



a	b	f
0	0	1
0	1	0
1	0	0
1	1	0

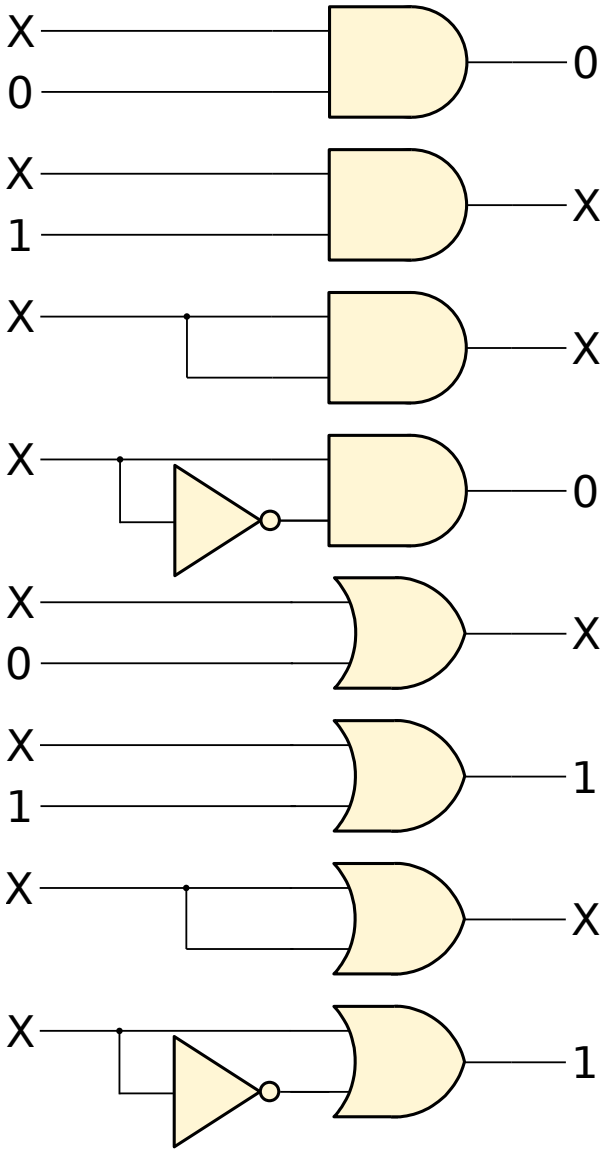
Bramki logiczne II

Oprócz bramek dwuwejściowych stosowane są również bramki wielowejściowe.
Stan na wyjściu bramki wielowejściowej jest funkcją wszystkich stanów na jej wejściu, np. wielowejściowej bramka AND‘:



- Wartość logiczna 1 pojawia się na wyjściu jedynie wówczas, gdy stan logiczny wszystkich wejść wynosi 1
- W innych przypadkach $f = 0$
- Bramka taka bywa nazywana **układem koincydencyjnym**

Podstawowe twierdzenia i tożsamości algebry Boole'a



Prawo przemienności:

$$x + y = y + x$$
$$x \cdot y = y \cdot x$$

Prawo łączności:

$$x + (y + z) = (x + y) + z = x + y + z$$
$$x \cdot (y \cdot z) = (x \cdot y) \cdot z = x \cdot y \cdot z$$

Prawo rozdzielności:

$$x \cdot (y + z) = x \cdot y + x \cdot z$$
$$(x + y) \cdot (w + z) = x \cdot w + y \cdot w + x \cdot z + y \cdot z$$

Prawa de Morgana:

$$\overline{x + y} = \overline{x} \cdot \overline{y}$$
$$\overline{x \cdot y} = \overline{x} + \overline{y}$$

Inne przydatne tożsamości:

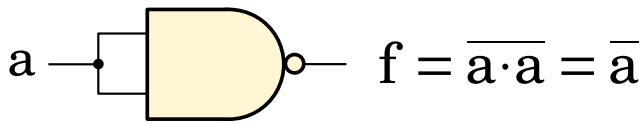
$$\overline{\overline{x}} = x$$
$$x + x \cdot y = x$$
$$x + \overline{x} \cdot y = x + y$$
$$x \cdot y + \overline{x} \cdot y = y$$
$$(x + y) \cdot (\overline{x} + y) = y$$

Bramki logiczne III

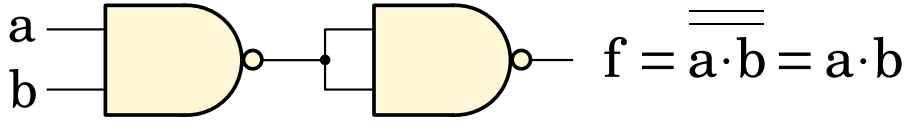
- Najbardziej uniwersalnymi bramkami są bramki NAND i NOR.

Używając tyko bramek NAND lub tylko bramek NOR
można zbudować układ realizujący dowolną funkcję logiczną

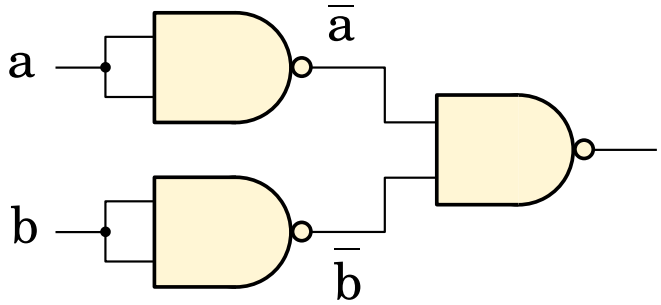
NOT



AND



OR



Korzystamy z prawa de Morgana:

$$\overline{a+b} = \overline{a} \cdot \overline{b}$$

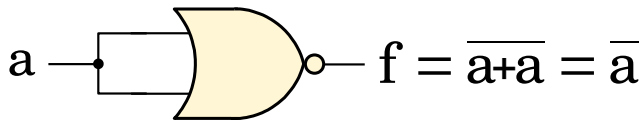
Zadanie:

Przedstaw przykłady realizacji podstawowych funkcji logicznych (NOT, OR, AND) przy użyciu bramek NOR

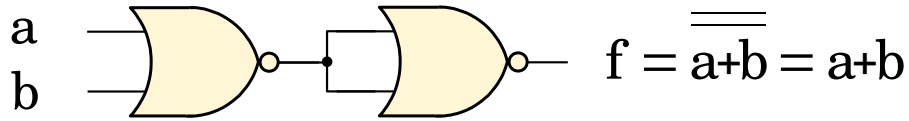
Zadanie:

Przedstaw przykłady realizacji podstawowych funkcji logicznych (NOT, OR, AND) przy użyciu bramek NOR

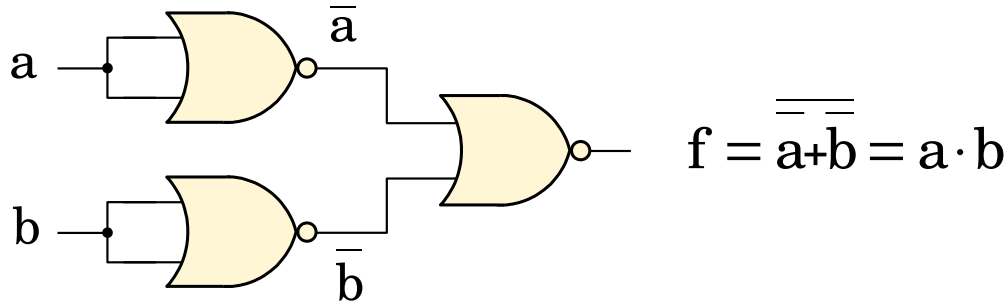
NOT



OR



AND



Tablice Karnaugha

- **Projektowanie układu kombinacyjnego** polega na zbudowaniu układu realizującego zadaną funkcję logiczną i zawierającego jak najmniejszą liczbę elementów, połączeń oraz wyprowadzeń
- Do minimalizacji prostych układów można stosować **tablice Karnaugha**

a \ b	0	1
0	f(0,0)	f(0,1)
1	f(1,0)	f(1,1)

Przykład dla pewnej funkcji logicznej

a	b	f
0	0	1
0	1	1
1	0	0
1	1	0

a \ b	0	1
0	1	1
1	0	0

lub

korzystając z kodu Gray'a:

ab	00	01	11	10
	1	1	0	0

$$f(a,b) = \overline{a}$$

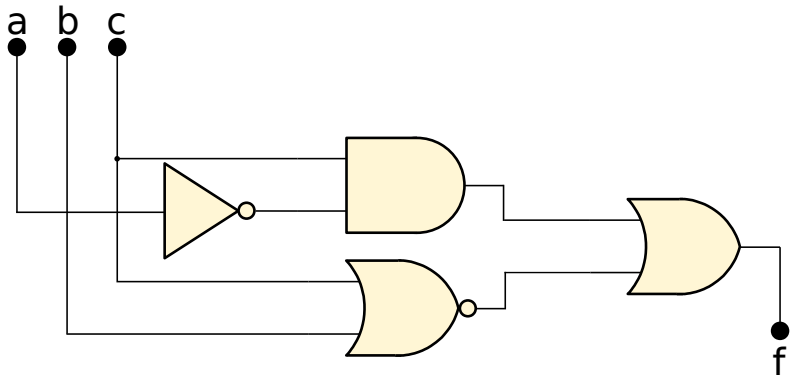
Tablice Karnaugha - przykład z 3 sygnałami

a	b	c	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

ab \ c	0	1
00	1	1
01	0	1
11	0	0
10	1	0

$$f(a,b,c) = \overline{a} \cdot c + \overline{b} \cdot \overline{c}$$

$$f(a,b,c) = \overline{a} \cdot c + \overline{b + c}$$

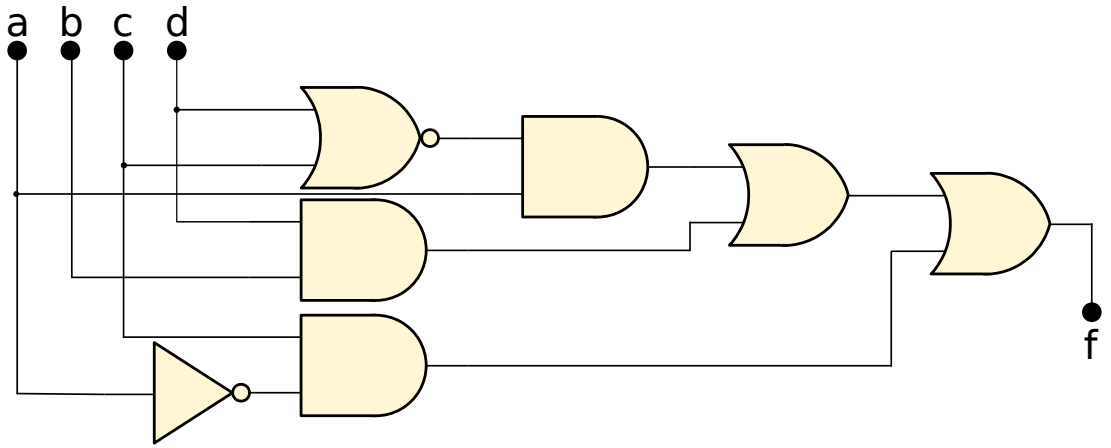


Tablice Karnaugha - przykład z 4 sygnałami

ab \ cd	00	01	11	10
00	0	0	1	1
01	0	1	1	1
11	1	1	1	0
10	1	0	0	0

$$f(a,b,c,d) = (a \cdot \overline{c} \cdot \overline{d}) + (b \cdot d) + (\overline{a} \cdot c)$$

$$f(a,b,c,d) = (a \cdot (\overline{c+d})) + (b \cdot d) + (\overline{a} \cdot c)$$

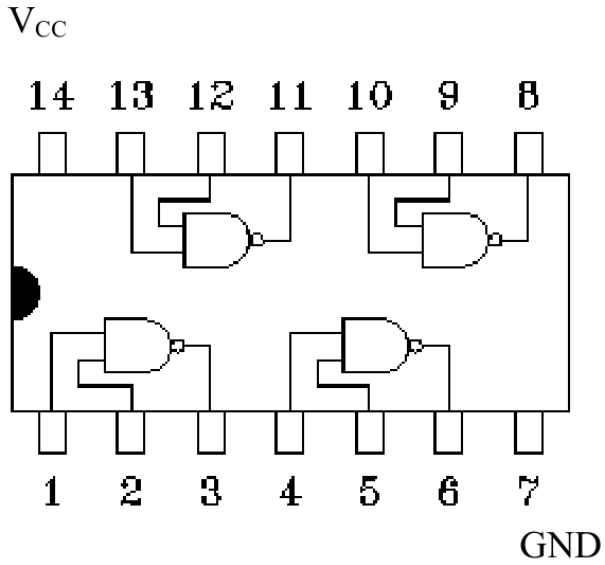


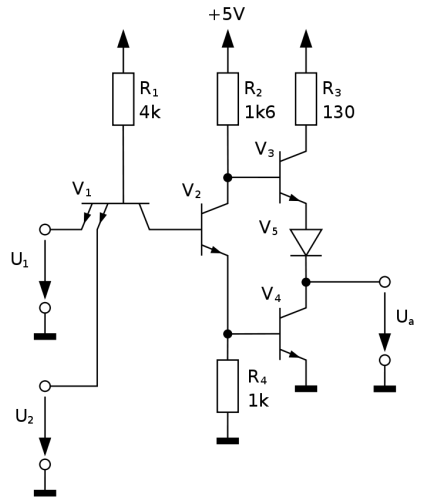
Bramki logiczne w praktyce

- Ze względu na stosowane technologie, bramki logiczne tworzą tzw. rodziny, np.: **TTL** (**Transistor – Transistor Logic**), ECL (Emitter - Coupled Logic), CMOS (Complementary MOS)

Charakterystyki układów TTL:

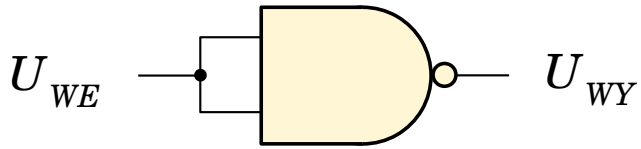
- układy pracują w logice dodatniej
- wartości napięć między wejściami/wyjściami a GND określają poziomy logiczne
- logicznemu zeru (L – stan niski) odpowiada napięcie z przedziału:
  0 – 0.8 V (sygnały wejściowe)
  0 – 0.4 V (sygnały wyjściowe)
- logicznej jedynce (H – stan wysoki) odpowiada napięcie z zakresu:
  2.0 – 5 V (sygnały wejściowe)
  2.7 – 5 V (sygnały wyjściowe)
- wejście bramki niepodłączone do niczego znajduje się w stanie logicznym 1
- układy zasila się napięciem +5 V





Czas propagacji sygnału przez bramkę

- Odpowiedź na wyjściu bramki następuje po pewnym, charakterystycznym dla danego układu czasie od momentu zmiany sygnałów wejściowych (czas propagacji sygnału przez bramkę)



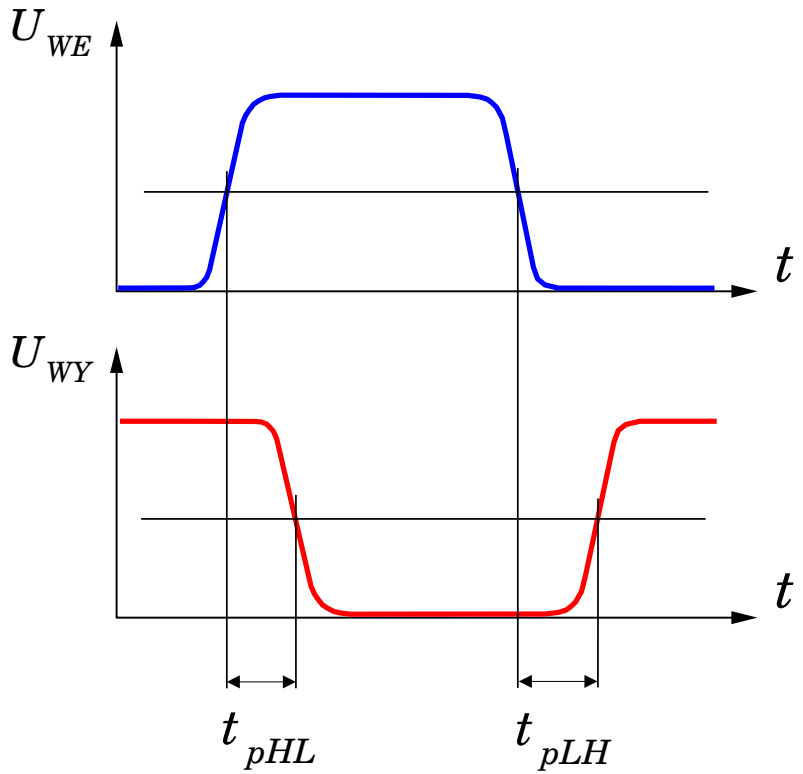
Średni czas propagacji

$$t_p = \frac{t_{pHL} + t_{pLH}}{2}$$

Czasy przełączania bramki UCY7400:

$t_{pHL} \approx 15$ ns

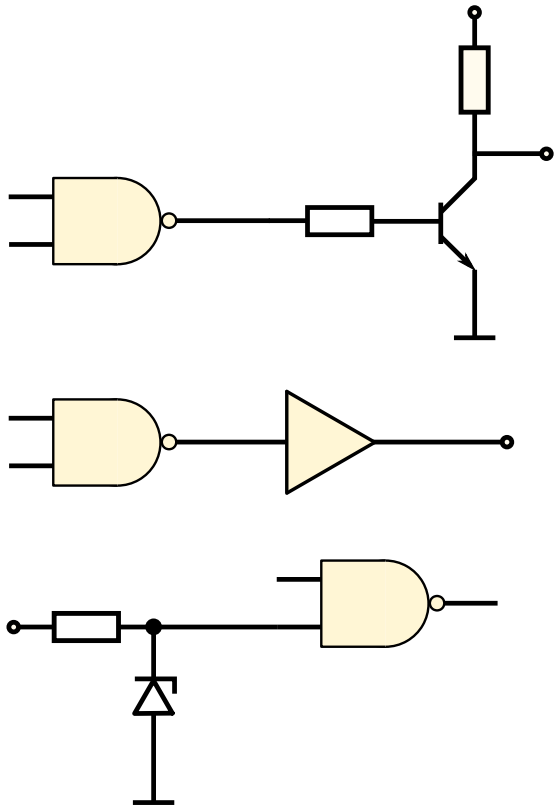
$t_{pLH} \approx 22$ ns



Ograniczenie obciążenia wyjścia bramki logicznej

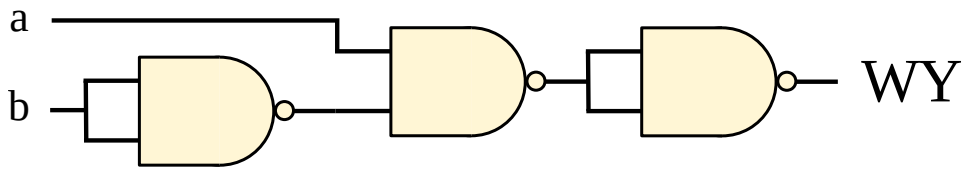
- Każdy układ cyfrowy ma określoną obciążalność, czyli liczbę mówiącą ile wejść cyfrowych może być podłączonych do danego wyjścia lub jaki największy prąd może przepłynąć przez wyjście
- W przypadku, gdy układ cyfrowy ma sterować innym układem należy posłużyć się dodatkowymi układami wspierającymi, np.:

- wzmacniaczem tranzystorowym
- tzw. *driverem* lub *buforem* - wzmacniaczem znajdującym się w rodzinie cyfrowych układów scalonych zwiększającym obciążalność wyjścia bramki
- Gdy do układu cyfrowego wprowadza się sygnał sterujący z zewnątrz należy zadbać o zachowanie standardowych napięć i odpowiednich polaryzacji, np. stosując zabezpieczenie diodą Zenera

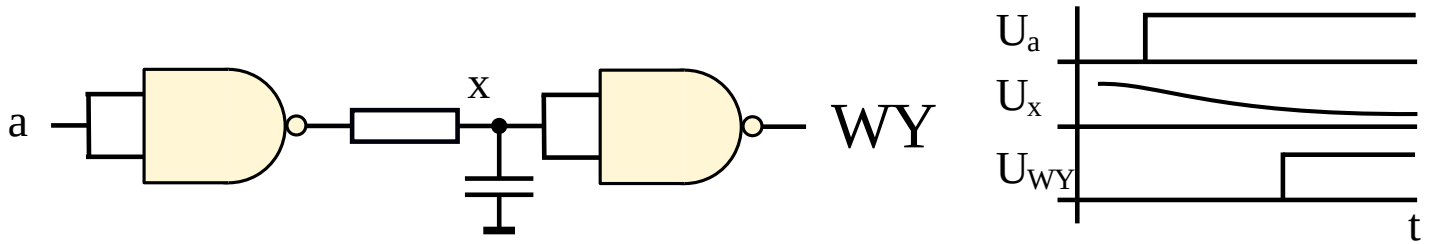


Przydatne układy oparte na bramkach logicznych

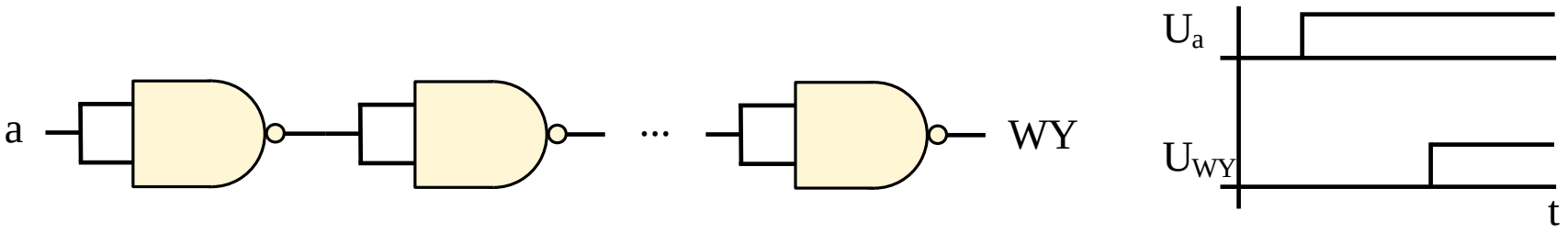
- Układ antykoincydencyjny: $f = a \cdot \overline{b}$



- Układ opóźniający, gdzie opóźnienie jest proporcjonalne do stałej czasowej RC

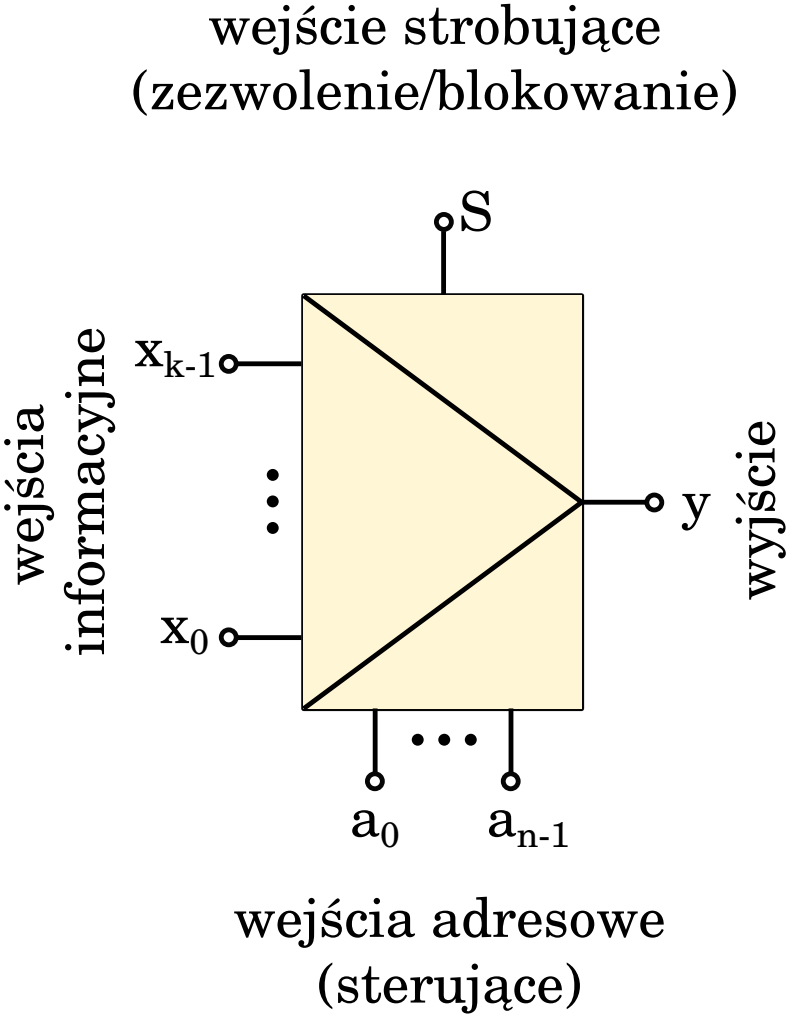


- Układ opóźniający, gdzie opóźnienie jest proporcjonalne liczby bramek

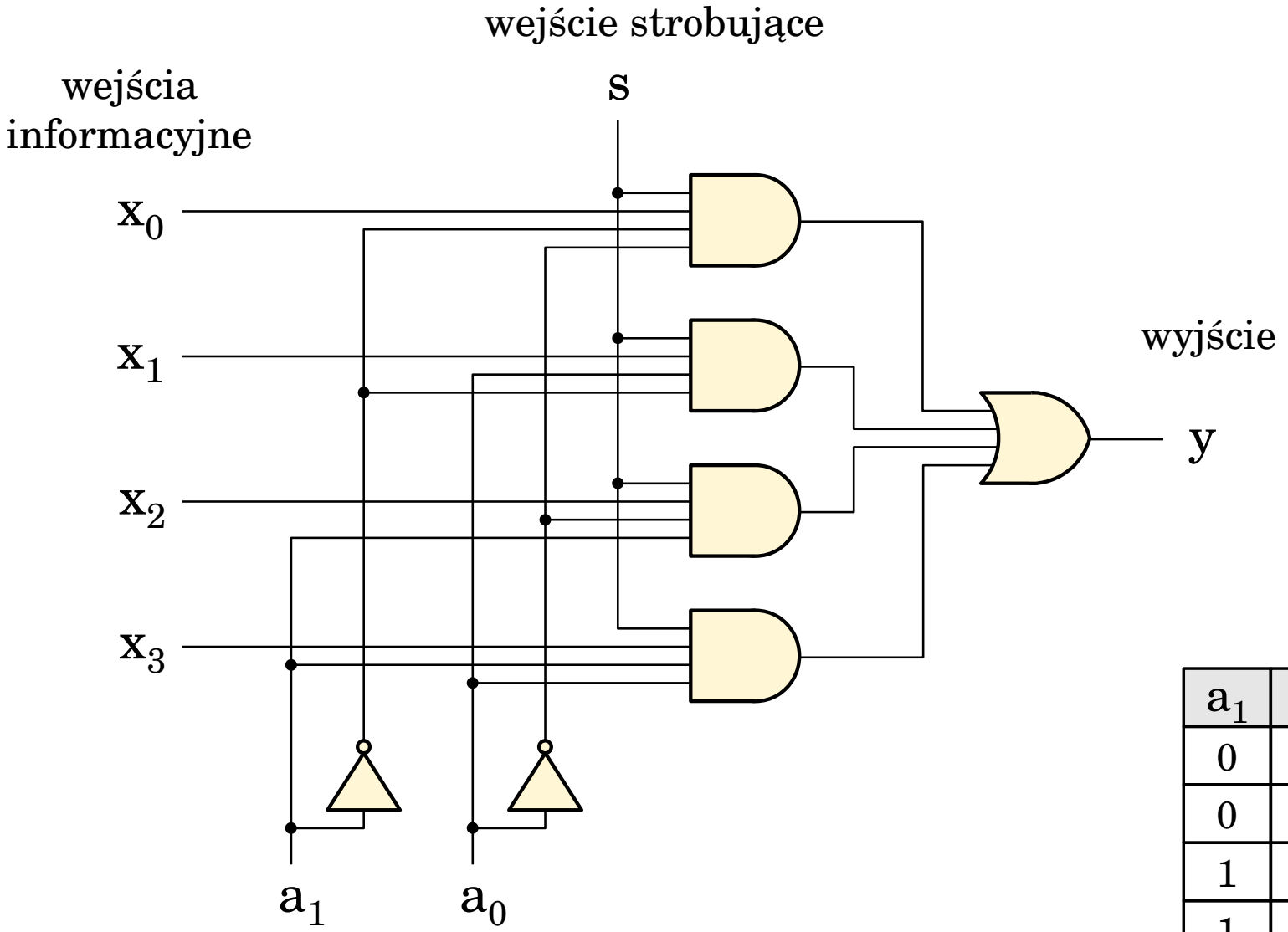


Multiplekser

- Multiplekser to logiczny układ kombinacyjny realizujący przepływ informacji między wieloma wejściami i jednym wyjściem.
- Wybór wejścia przesyłającego informację jest określony przez podanie jego adresu (numeru) na wejściach adresowych
- Przepływ informacji z wejścia na wyjście jest możliwy tylko wtedy, gdy wejście strobujące znajduje się w stanie logicznym 1
- Zastosowanie: **mikrokontrolery** (gdy np. chcemy skorzystać ze znacznie większej liczby portów, niż jest dostępna), reklamy świetlne



Przykład multipleksera z czterokanałowym wejściem

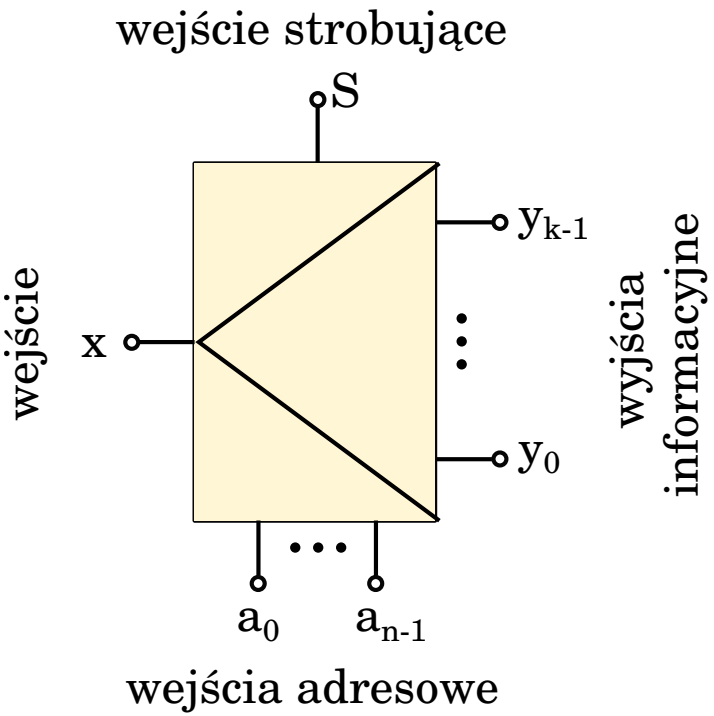


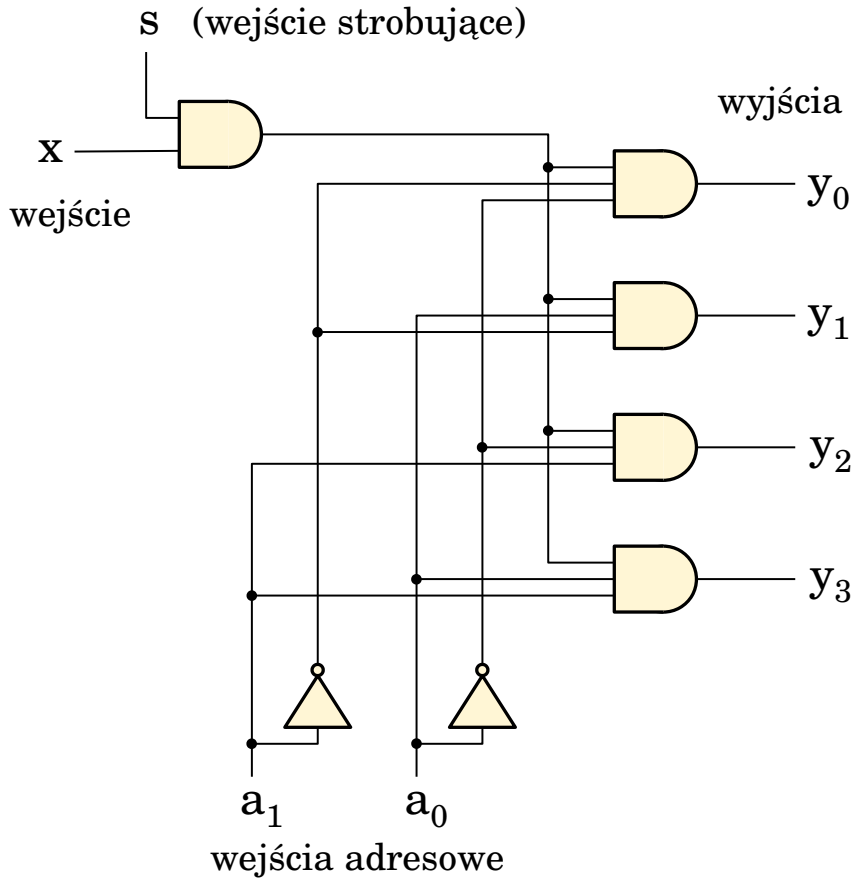
a_1	a_0	y
0	0	x_0
0	1	x_1
1	0	x_2
1	1	x_3

gdy s = 1

Demultiplekser

- Demultiplekser przekazuje dane z jednego wejścia na selektywnie wybrane adresem wyjście



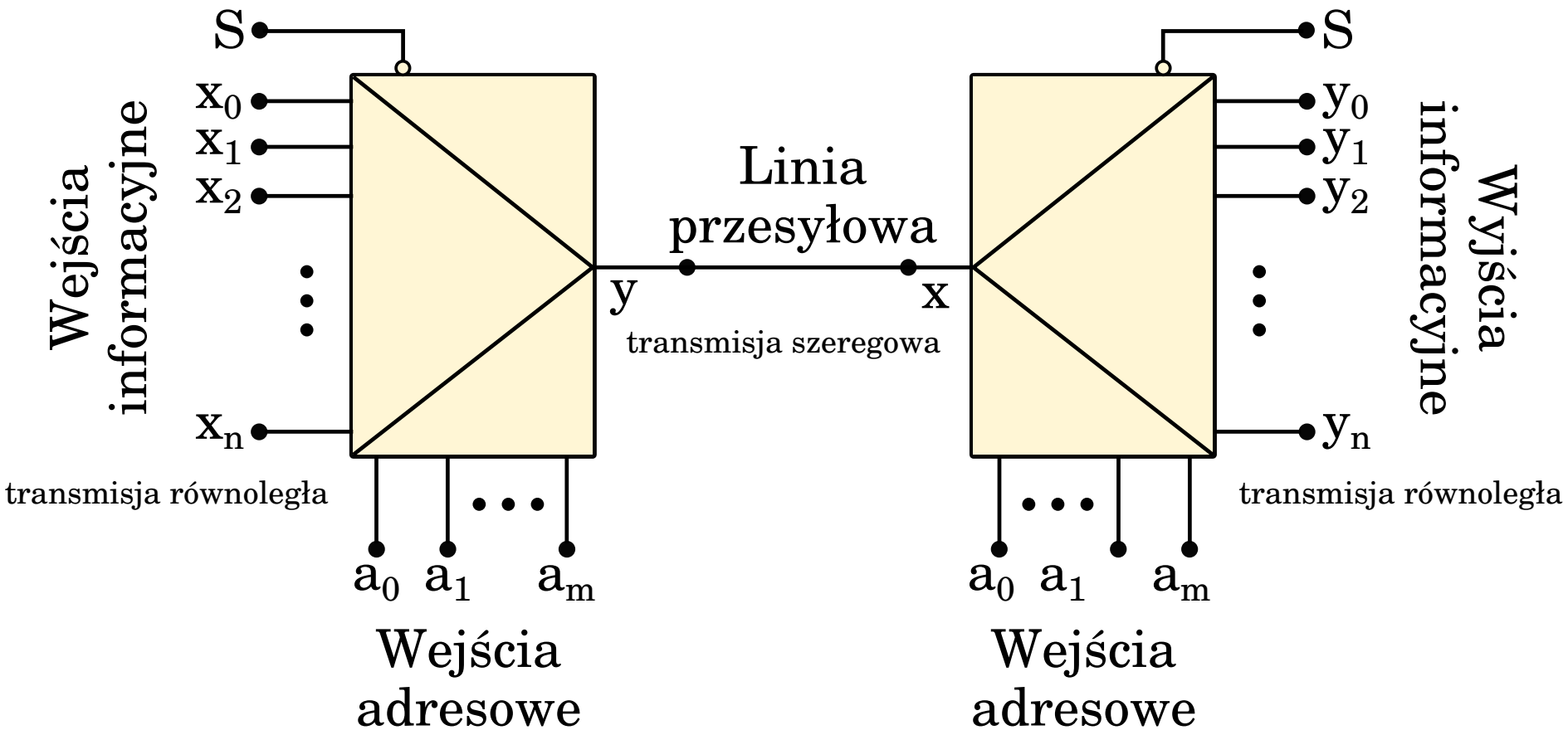


a_1	a_0	y_3	y_2	y_1	y_0
0	0	0	0	0	x
0	1	0	0	x	0
1	0	0	x	0	0
1	1	x	0	0	0

gdy s = 1

Przykład demultipleksera z czterema wyjściami

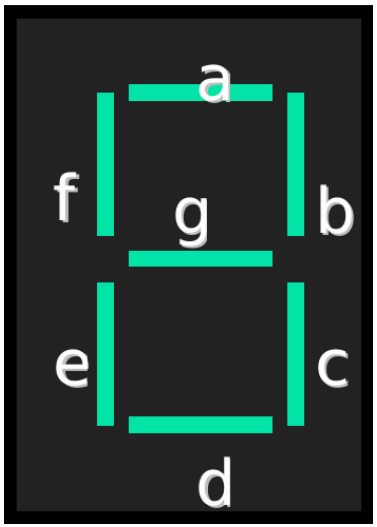
Schemat multiplekserowego przesyłu informacji



- Ekonomizacja przesyłu informacji
  - Multiplekser i demultiplekser zazwyczaj są połączone w jeden element wyposażenia, który jest po prostu nazywany multiplekserem

Forma informacji i rodzaje kodów

- Każda **informacja cyfrowa** może być przedstawiona jako słowo logiczne, czyli **określona kombinacja bitów** $\rightarrow$ **kod**
- Najczęściej spotykanym sposobem kodowania informacji cyfrowej jest **naturalny kod binarny** - pozycyjny system liczbowy, którego podstawą jest liczba 2, a do zapisu liczb potrzebne są tylko dwie cyfry: 1 i 0 (prawda i fałsz)
- **Kod siedmiosegmentowego wyświetlacza cyfr** - w siedmiobitowym słowie binarnym każdy bit odpowiadał jednemu z segmentów wyświetlacza



CYFRA	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1

Inne kody

- **Kod Gray'a** (**kod refleksyjny**) - bezwagowy zero-jedynkowy kod niepozycyjny
  - kodem Graya długości n jest ciąg wszystkich 2^n różnych ciągów n cyfr $\{0,1\}$, ustawionych tak, że dwa kolejne ciągi cyfr różnią się dokładnie jedną z nich

2 – bitowy
00
01
11
10

3 – bitowy
000
001
011
010
110
111
101
100

- **Kod „1 z N"** (**kod pierścieniowy**) - bezwagowy zero-jedynkowy kod pozycyjny
  - w słowie logicznym tylko jeden z bitów przyjmuje wartość 1 (pozostałe bity 0), pozycja jedynki determinuje zakodowaną wartość

„1 z 4"
0001
0010
0100
1000

„1 z 8"
00000001
00000010
00000100
00001000
00010000
00100000
01000000
10000000

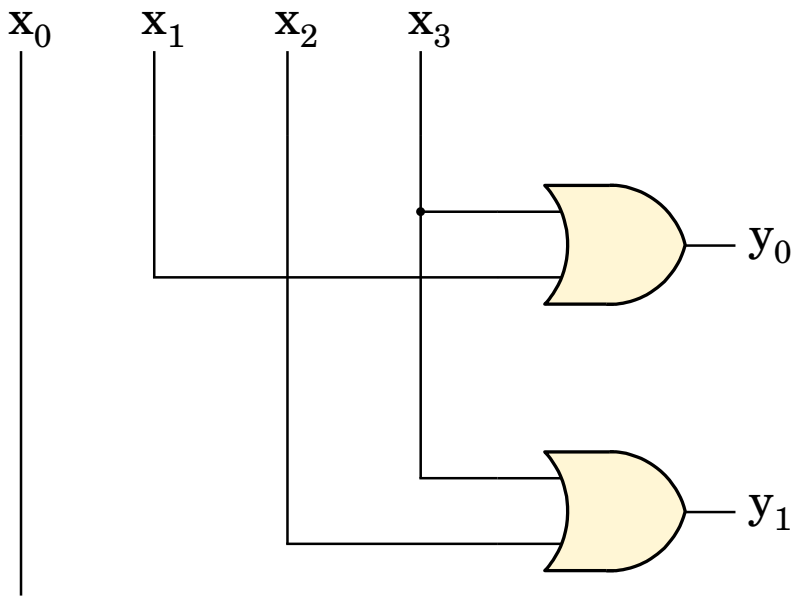
Podsumowanie wybranych kodów dla 16 elementów

naturalny binarny	dziesiętny	Graya	Kod “1 z N”
0000	0	0000	0000000000000001
0001	1	0001	0000000000000010
0010	2	0011	0000000000000100
0011	3	0010	0000000000001000
0100	4	0110	0000000000010000
0101	5	0111	0000000000100000
0110	6	0101	0000000001000000
0111	7	0100	0000000010000000
1000	8	1100	0000000100000000
1001	9	1101	0000001000000000
1010	10	1111	0000010000000000
1011	11	1110	0000100000000000
1100	12	1010	0001000000000000
1101	13	1011	0010000000000000
1110	14	1001	0100000000000000
1111	15	1000	1000000000000000

Konwertery kodów

- **Koder** (enkoder) - układ cyfrowy, który przekształca kod „1 z N" na określony kod wyjściowy (najczęściej naturalny kod binarny)
  - koder „1 z N"→kod binarny posiada n wejść oraz $k = \log_2(n)$ wyjść (czyli $n = 2^k$)
  - koder priorytetowy - jest to układ kombinacyjny, w którym kodem wejściowym jest kod „x z n" oraz posiada ustalone priorytety poszczególnych wejść
- **Dekoder** - układ cyfrowy który przekształca określony kod wejściowy (np. naturalny binarny) na kod wyjściowy „1 z N"
  - ma więc N wyjść, przy czym każdemu ze słów wejściowych jest przyporządkowany sygnał aktywny pojawiający się tylko na jednym z N wyjść
- **Transkoder** (translator) - układ cyfrowy realizujący konwersję dwóch dowolnych kodów z których żaden nie jest kodem „1 z N"
  - typowym przykładem takiego układu jest układ zamieniający naturalny kod binarny na kod wyświetlacza siedmiosegmentowego

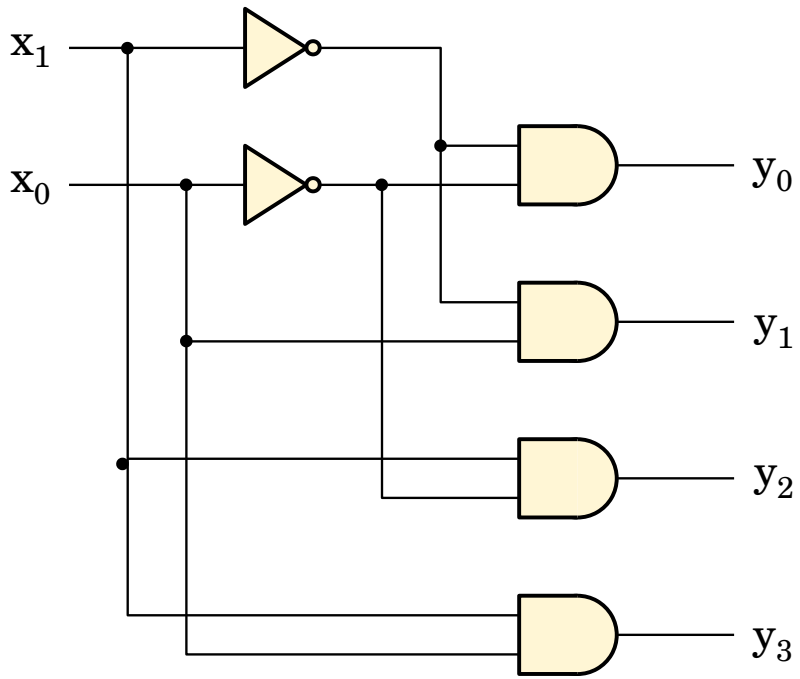
Przykład kodera



x_3	x_2	x_1	x_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Koder realizujący konwersję kodu „1 z 4” na dwubitowy naturalny kod binarny

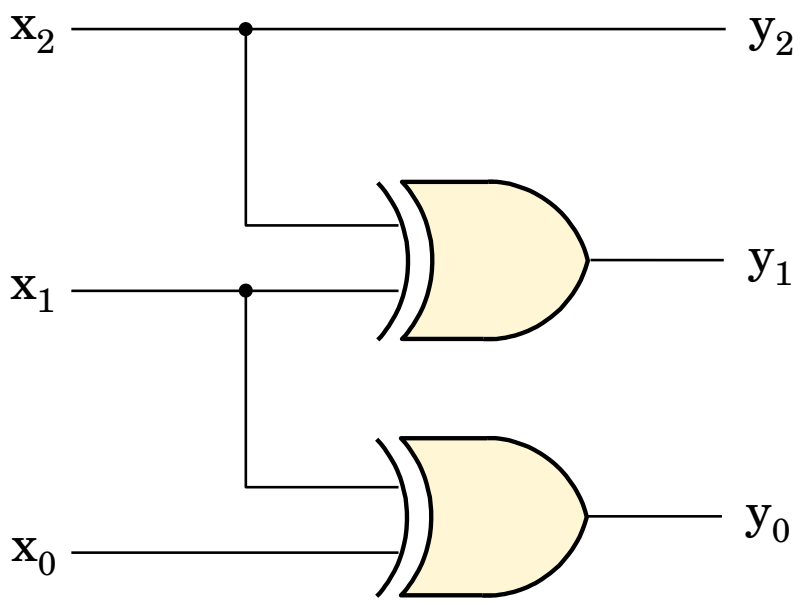
Przykład dekodera



x_1	x_0	y_3	y_2	y_1	y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Dekoder realizujący konwersję dwubitowego naturalnego kodu binarnego na kod „1 z 4”

Przykład transkodera

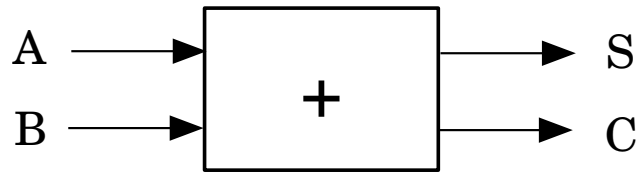


x_2	x_1	x_0	y_2	y_1	y_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Transkoder realizujący konwersję naturalnego kodu binarnego na kod Graya (kody 3 - bitowe)

Sumator

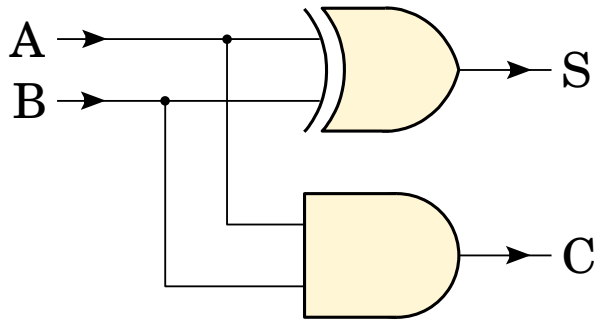
- **Sumatory** - podstawowe układy arytmetyczne, realizujące dodawanie liczb przedstawionych w zapisie binarnym
- **Półsumatory** - układy dodające dwie liczby jednobitowe A i B
  - w wyniku sumowania powstaje liczba dwubitowa której elementami są suma (S) i przeniesienie (C)
  - młodszy bit wyniku wyprowadza się na wyjście S, starszy bit - na wyjście C



B	A	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$S = A \oplus B$$

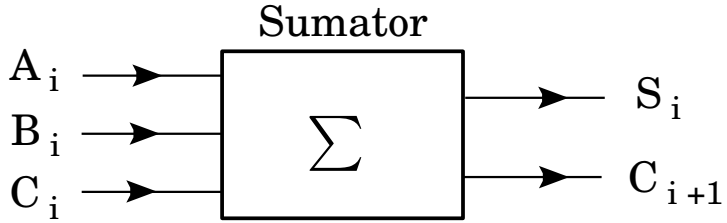
$$C = A \cdot B$$



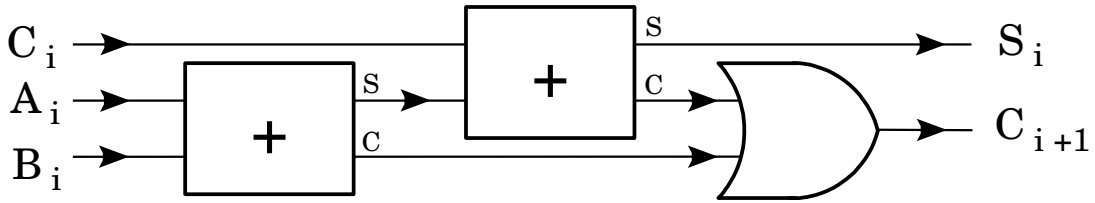
Pełny sumator (jednobitowy)

- W przypadku dodawania liczb o większej liczbie bitów półsumator można zastosować tylko na najmłodszej pozycji, na wszystkich pozostałych trzeba dodawać nie dwa, ale trzy bity uwzględniając przeniesienie z poprzedniej pozycji
- Pełny sumator zawiera trzy wejścia A_i, B_i, C_i oraz dwa wyjścia S_i, C_{i+1}.

B_i	A_i	C_i	C_{i+1}	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



Schemat sumatora zbudowanego z półsumatorów

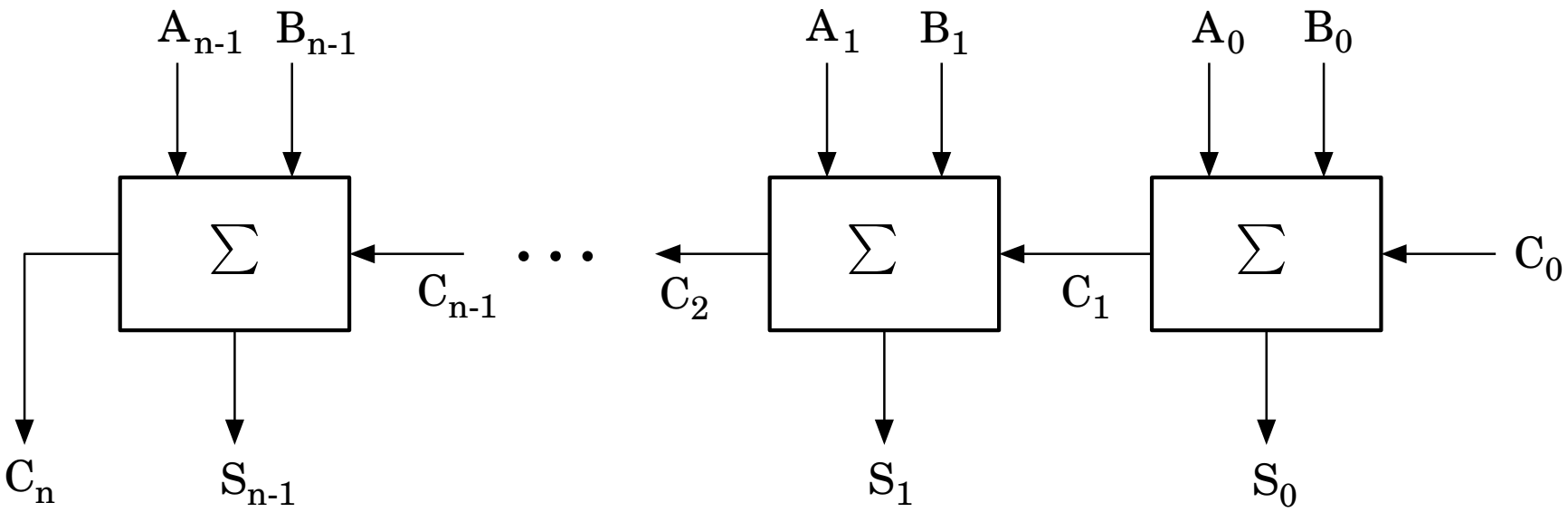


$$S_i = A_i \oplus B_i \oplus C_i$$

$$C_{i+1} = A_i \cdot B_i + A_i \cdot C_i + B_i \cdot C_i$$

Sumator wielobitowy

- W celu dodawania liczb wielobitowych sumatory jednobitowe łączy się w zespoły
- W najprostszym przypadku sumatory łączy się szeregowo (wyjście przeniesienia łączy się z wejściem przeniesienia bloku następnego)



Sumator n−bitowy z przeniesieniami szeregowymi

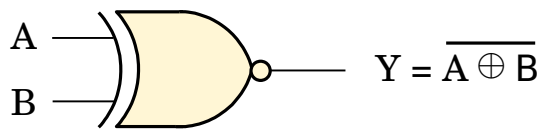
np.:

$$\begin{array}{r} 0110 \\ +\ 1100 \\ \hline 10010 \end{array}$$

Komparator

- **Komparator cyfrowy** - to układ cyfrowy służący do porównywania dwu lub więcej liczb binarnych
- Najważniejsze kryteria porównawcze to:
  - $A = B$
  - $A > B$
  - $A < B$

  układ sprawdzający wszystkie trzy relacje nazywa się **komparatorem uniwersalnym**
- Kryterium równości dwóch liczb binarnych jest identyczność wszystkich bitów
- W przypadku dwóch liczb jednobitowych A i B, informację o tym uzyskać można za pomocą funkcji **negacja EXOR**:

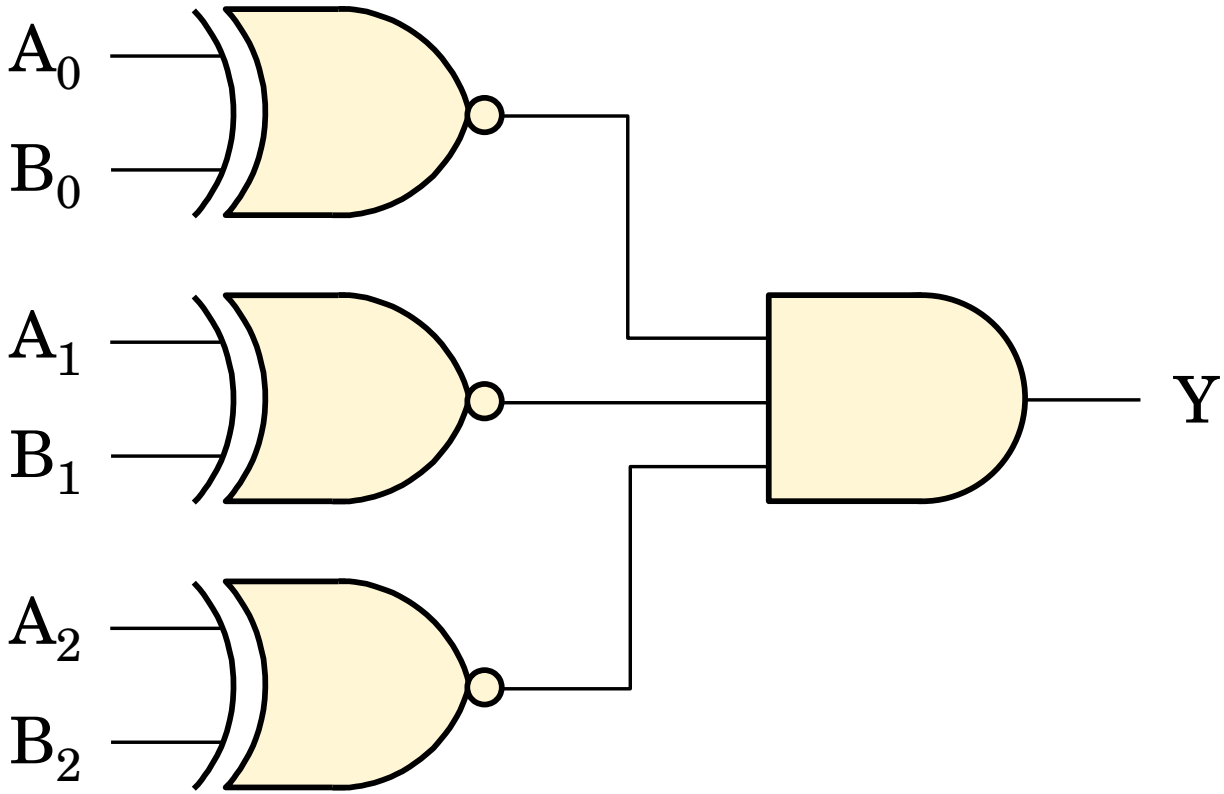


Wartość 1 na wyjściu sygnalizuje równość A = B

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Komparator równoległy

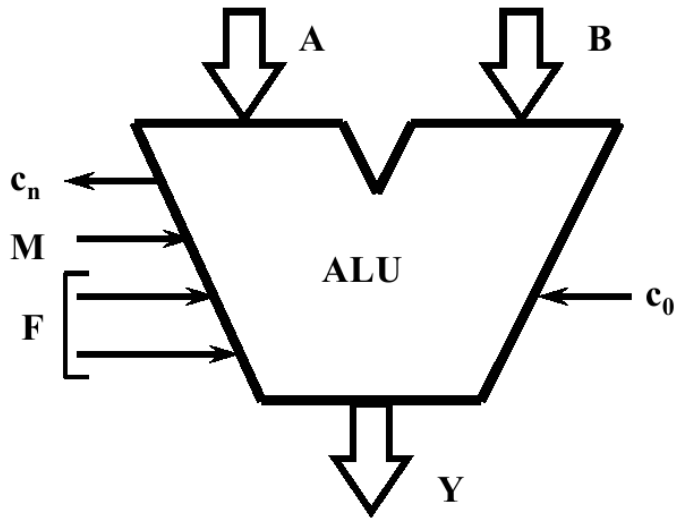
- **Komparator równoległy** to taki układ, na którego wejścia podawane są jednocześnie wszystkie bity porównywanych liczb
- Przykład komparatora równoległego 3-bitowego

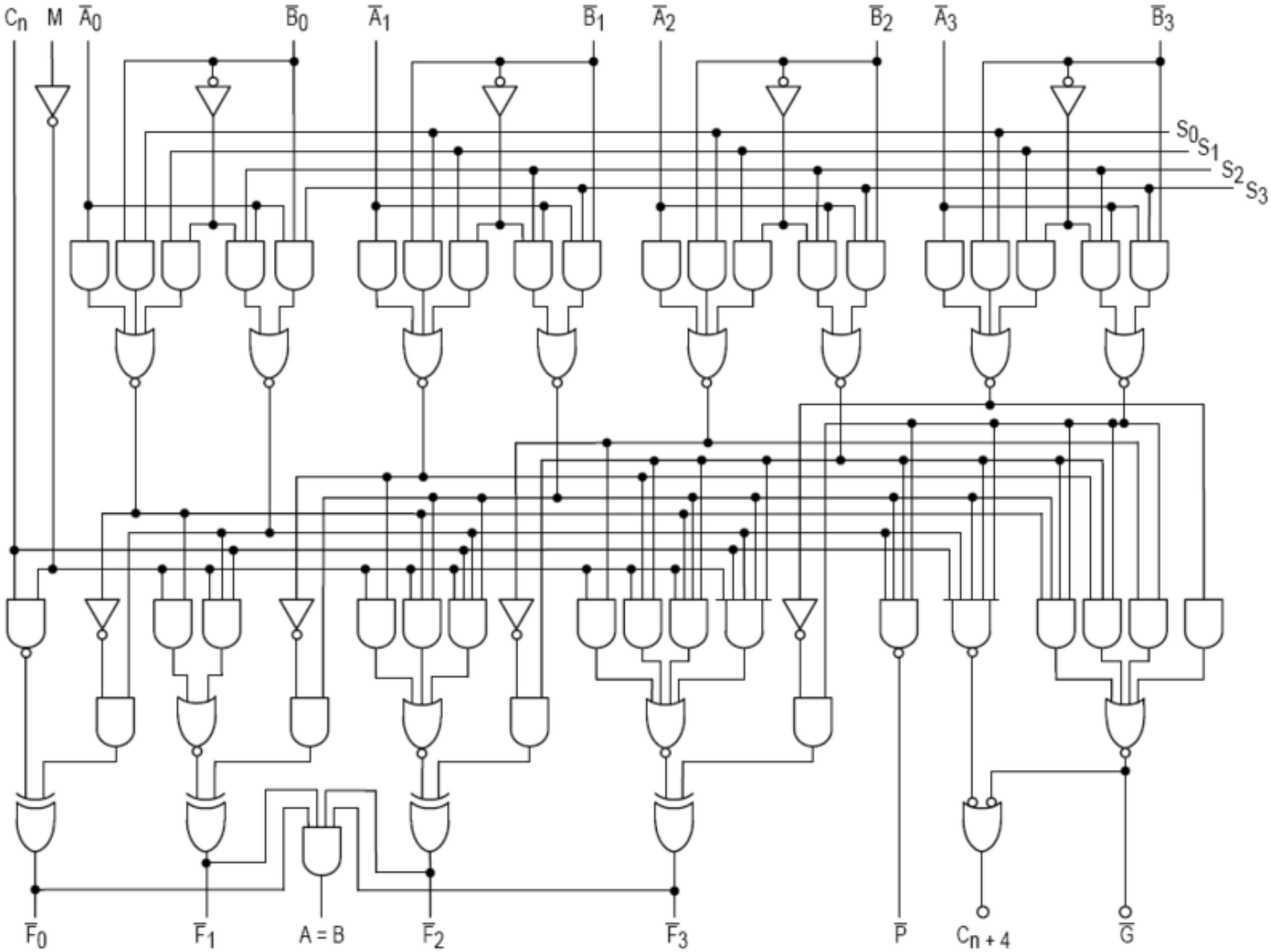


$Y = 1$
tylko gdy:
$A_0 = B_0$
$A_1 = B_1$
$A_2 = B_2$
czyli $A = B$

Jednostki arytmetyczno-logiczne (ALU)

- ALU (ang. *Arithmetic Logic Unit*) - jest uniwersalnym układem cyfrowym przeznaczonym do wykonywania operacji arytmetycznych i logicznych
- ALU jest podstawowym blokiem centralnej jednostki obliczeniowej komputera
- Typowe ALU ma dwa wejścia odpowiadające parze argumentów i jedno wyjście na wynik, a operacje jakie prowadzi to:
  - AND, OR, NOT, XOR,
  - sumowanie słów logicznych
  - przesunięcia bitowe o jeden bit, stałą liczbę bitów, czasem też o zmienną liczbę
  - często też: odejmowanie, negacja liczby, dodawanie z przeniesieniem, zwiększanie/zmniejszanie o 1 mnożenie dzielenie/modulo




Cn
M
A0
B0
A1
B1
A2
B2
A3
B3
S0
S1
S2
S3
F0
F1
A = B
F2
F3
P
Cn + 4
G