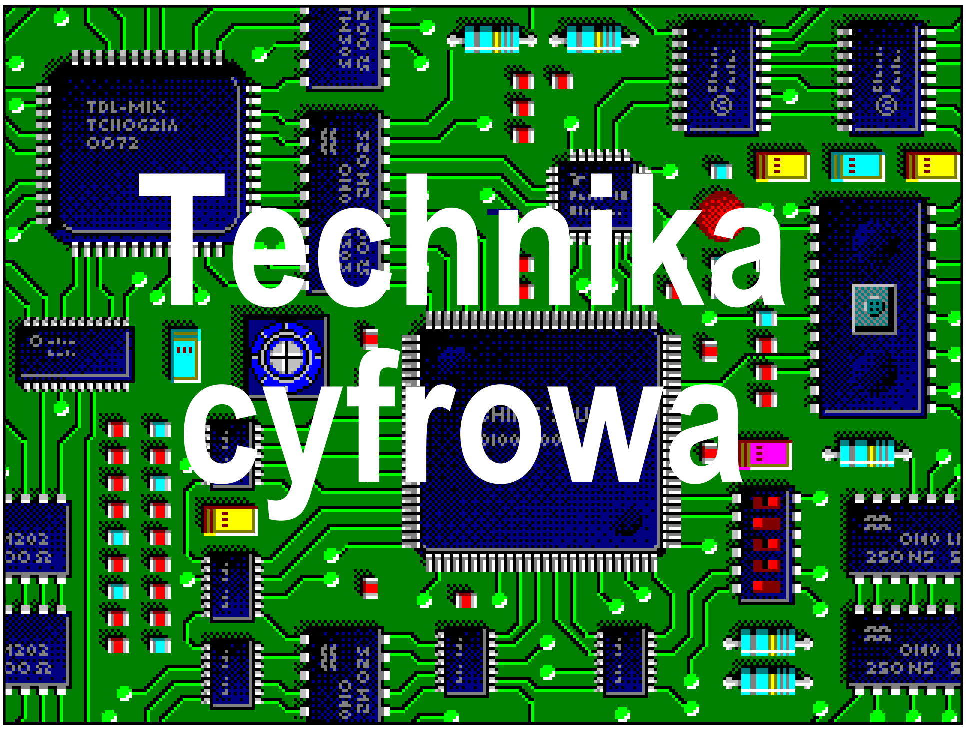



# Technika cyfrowa

Dr inż. Janusz Dudziak

Literatura:

1. J. Kalisz *Podstawy elektroniki cyfrowej*, WKŁ
2. M. Molski *Wstęp do techniki cyfrowej* WKŁ
3. B. Wilkinson: *Układy cyfrowe*, WKŁ
4. J. Pieńkos, J. Turczyński: *Układy scalone TTL w systemach cyfrowych*, WKŁ
5. W. Traczyk: *Układy cyfrowe. Podstawy teoretyczne i metody syntezy*, WNT
6. J. Piecha*: Elementy i układy cyfrowe*, PWN
7. W. Majewski: *Układy logiczne,* WNT

# pojęcia i definicje podstawowe

## technika cyfrowa

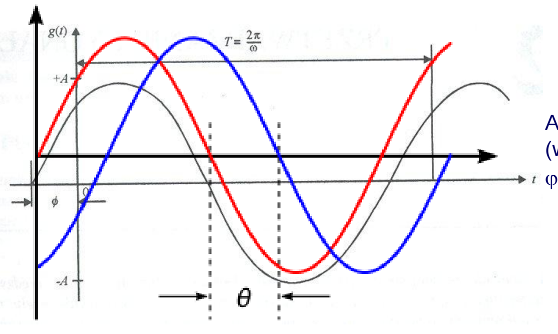
**Definicje**

**Wartość dyskretna** - wartość **nieciągła** lub pojedynczą.

W matematyce – przeciwieństwo ciągłości, np. funkcja *f ( x ) = [ x ]* (przyporządkowująca każdej liczbie rzeczywistej jej wartość całkowitą) ma dziedzinę **ciągłą**, a zbiór wartości **dyskretny**.

## sygnał

- **sygnał** to abstrakcyjny model dowolnej mierzalnej wielkości zmieniającej się w czasie, generowanej przez zjawiska fizyczne lub systemy. Tak jak wszystkie zjawiska może być opisany za pomocą aparatu matematycznego, np. poprzez podanie pewnej funkcji zależnej od czasu. Ponieważ sygnał niesie informację o naturze badanych zjawisk lub systemów, w niektórych dziedzinach nauk jest on traktowany jak nośnik informacji. Sygnał oznacza zatem przepływ strumienia informacji, przy czym przepływ może odbywać się w jednym lub w wielu wymiarach.




$$g(t) = A\sin(\omega t + \varphi)$$

A - amplituda, ω - pulsacja (w radianach na sekundę)
φ - faza początkowa

## sygnał analogowy

- **sygnał analogowy** - sygnał, który może przyjmować dowolną wartość z ciągłego przedziału (nieskończonego lub ograniczonego zakresem zmienności). Jego wartości mogą zostać określone w każdej chwili czasu, dzięki funkcji matematycznej opisującej dany sygnał. chwilowej przebiegu pierwotnego

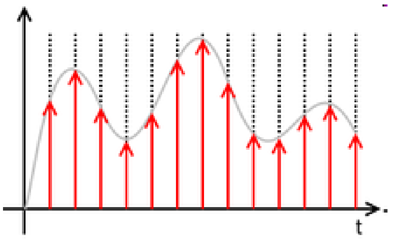

- przykład - napięcie wytworzone przez przetwornik będące odpowiednikiem chwilowej wartości ciśnienia

## sygnał dyskretny

- **Sygnał dyskretny** – model wielkości zmiennej, która jest określona tylko w dyskretnych chwilach czasu. Najczęściej jest to sygnał powstały poprzez próbkowanie sygnału ciągłego.

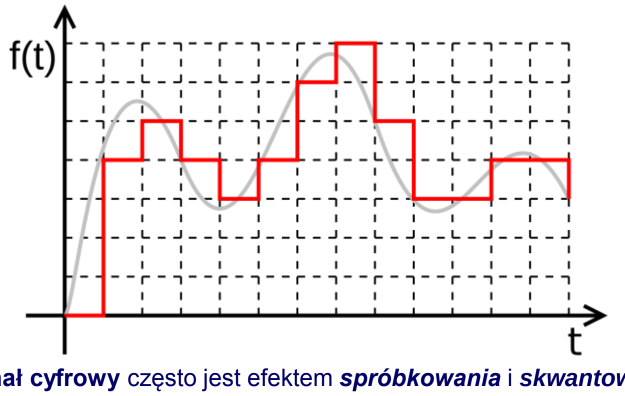
W odróżnieniu od sygnału ciągłego, sygnał dyskretny nie jest funkcją zdefiniowaną dla ciągłego przedziału argumentów, lecz ciągiem liczbowym. Każda wartość ciągu nazywa się próbką (ang. *sample*).

W odróżnieniu od **sygnału cyfrowego**, poszczególne próbki sygnału dyskretnego analogowego mogą przyjmować dowolne wartości z nieograniczonego lub ograniczonego zbioru.




## sygnał cyfrowy

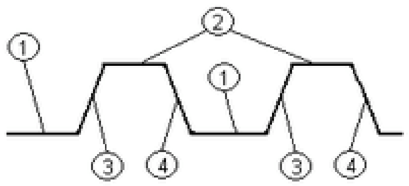
**Sygnał cyfrowy** – sygnał, którego dziedzina i zbiór wartości są dyskretne.




**Sygnał cyfrowy** często jest efektem ***spróbkowania*** i ***skwantowania*** sygnału pierwotnie ciągłego (analogowego, np. sygnał na wyjściu komparatora napięcia kontrolującego pewien proces w określonych chwilach)

## sygnał binarny

**Sygnał cyfrowy** , którego zbiór wartości jest **dwuelementowy**.



1) Poziom niski, 2) Poziom wysoki,
3) Zbocze narastające,
4) Zbocze opadające

# System pozycyjny

## Definicja

***System pozycyjny*** definiuje:

- wektor wag
  **W**=$\{w_{k-1},\ldots,w_1,w_0,\ldots,w_{-m}\}$

odpowiadający poszczególnym pozycjom liczby

- określony dla każdej pozycji zbiór dozwolonych cyfr
  $\mathbf{D}^i=\{d^i_{p-1},\ldots,d^i_1,d^i_0\}$.

W systemie pozycyjnym wartością liczby **X**= $\{x_{k-1},\ldots,x_1,x_0,\ldots,x_{-m}\}$ gdzie $x_i \subset D^i$ jest X=**X**·**W.**

# System pozycyjny

W systemie **stałobazowym** każda waga $w_i$ jest potęgą stałej całkowitej $\beta$>=2 zwanej podstawą (bazą) sytemu.

Wartość liczby w systemie stałobazowym oblicza się ze wzoru

$$X = \Sigma x_i \beta^i$$

**W**=$\{w_{k-1},\ldots,w_1,w_0,\ldots,w_{-m}\}$

## informacja cyfrowa

*system dziesiętny*

Zbiór liczb dozwolonych jest dziesięcioelementowy D={0,1,2,3,4,5,6,7,8,9}.

Wektor wag W=$\{\ldots\ 10^3,10^2,10^1,1,\ 10^{-1},10^{-2},10^{-2}\ \ldots\}$

przykład:

$83678,15 = 8*10^4+ 3*10^3 +6*10^2+ 7*10^1+ 8*10^0+1*10^{-1}+ 5*10^{-2}$

*system dwójkowy*

Zbiór liczb dozwolonych jest dwuelementowy D={0,1}.

Wektor wag W=$\{\ldots\ 2^3,2^2,2^1,1,\ 2^{-1},2^{-2},2^{-2}\ \ldots\}$

przykład:

$10110.1 = 1*2^4+ 0*2^3 +1*2^2+ 1*2^1+ 0*2^0+1*2^{-1}$

## informacja cyfrowa

$b_{n-1}b_{n-2}\ldots b_2b_1b_0$

naturalny kod binarny (NKB)

$$\sum_{i=0}^{i=n-1} b_i \cdot 2^i$$

| NKB | dziesiętnie |
|---:|---:|
| 1 | 1 |
| 10 | 2 |
| 100 | 4 |
| 1000 | 8 |
| 10000 | 16 |
| 100000 | 32 |
| 1000000 | 64 |
| 10000000 | 128 |
| 100000000 | 256 |
| 1000000000 | 512 |
| 10000000000 | 1024 |

## sygnał

Współcześnie telekomunikacja i elektronika powszechnego użytku prawie całkowicie zostały zdominowane przez cyfrowe przetwarzanie sygnałów, które jest powtarzalne, bardziej niezawodne i tańsze od przetwarzania analogowego

## sygnał

Twierdzenie **Kotielnikowa-Shannona** o próbkowaniu

1. Sygnał o skończonym paśmie i skończonej energii, nie zawierający składowych widma o częstotliwości przekraczającej *W* Hz może być jednoznacznie opisany za pomocą próbek wziętych w punktach czasu odległych o 1/2W sekund.
2. Sygnał o skończonym paśmie i skończonej energii, nie zawierający składowych widma o częstotliwości przekraczającej *W* Hz może być jednoznacznie odtworzony z próbek wziętych w punktach czasu odległych o 1/2W sekund.

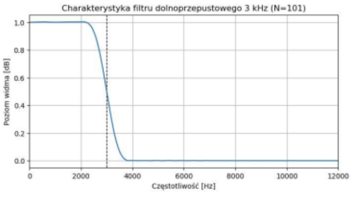
Kryterium Nyquista - $F_p > 2 f_{\max}$

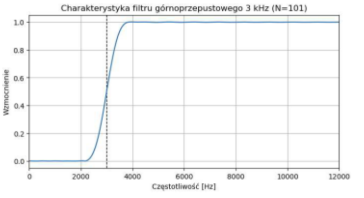
| System | $f_{max}$ | $F_p$ |
|---|---|---|
| telefonia | 3,4kHz | 8kHz |
| CD | 20kHz | 44,1kHz |
| TV cyfrowa | 6MHz | 13,5MHz |

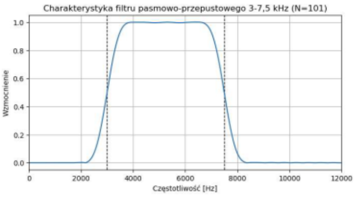
## sygnał

- **Cyfrowe przetwarzanie sygnałów** (**CPS**; *ang. Digital Signal Processing, DSP*) - dziedzina nauki i techniki zajmująca się sygnałami w postaci cyfrowej i metodami przetwarzania takich sygnałów. Cyfrowe przetwarzanie sygnałów i analogowe przetwarzanie sygnałów są gałęziami nadrzędnej dyscypliny: przetwarzania sygnałów. W ramach CPS wskazać można takie obszary jak: cyfrowe przetwarzanie dźwięku, cyfrowe przetwarzanie obrazów oraz przetwarzanie mowy.
- Pierwszym etapem cyfrowego przetwarzania sygnałów jest zazwyczaj konwersja sygnału z postaci analogowej na cyfrową za pomocą przetwornika analogowo-cyfrowego. Często sygnał przetworzony cyfrowo jest sygnałem wejściowym dla układu analogowego - wymaga to zastosowania przetwornika cyfrowo-analogowego.
- Algorytmy cyfrowego przetwarzania sygnałów są niekiedy realizowane przez specjalizowane urządzenia komputerowe, które korzystają ze specjalizowanych procesorów sygnałowych *(ang. Digital Signal Processor, DSP)*. Pozwalają one na przetwarzanie sygnałów w czasie rzeczywistym *(ang. real time signal processing)*.

## NP. filtracja cyfrowa

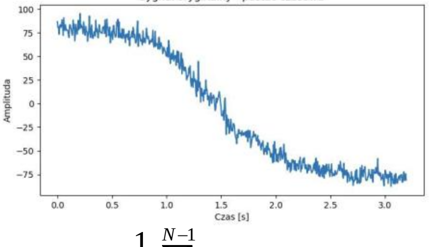


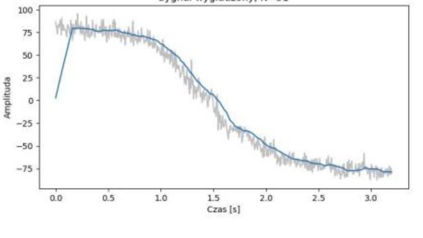




## filtracja cyfrowa

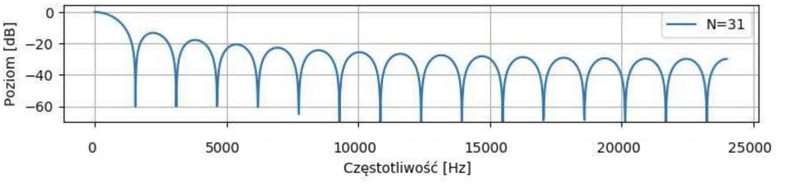
Mamy sygnał:



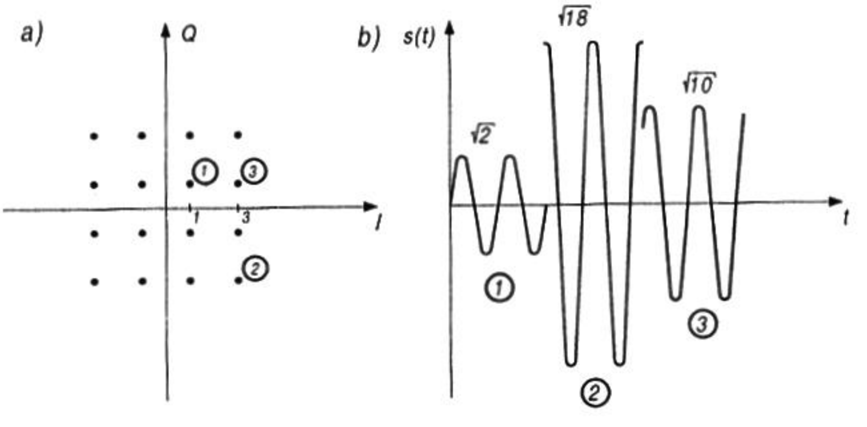

$$y[n] = \frac{1}{N}\sum_{i=0}^{N-1} x[n-i]$$ Filtr średniej ruchomej



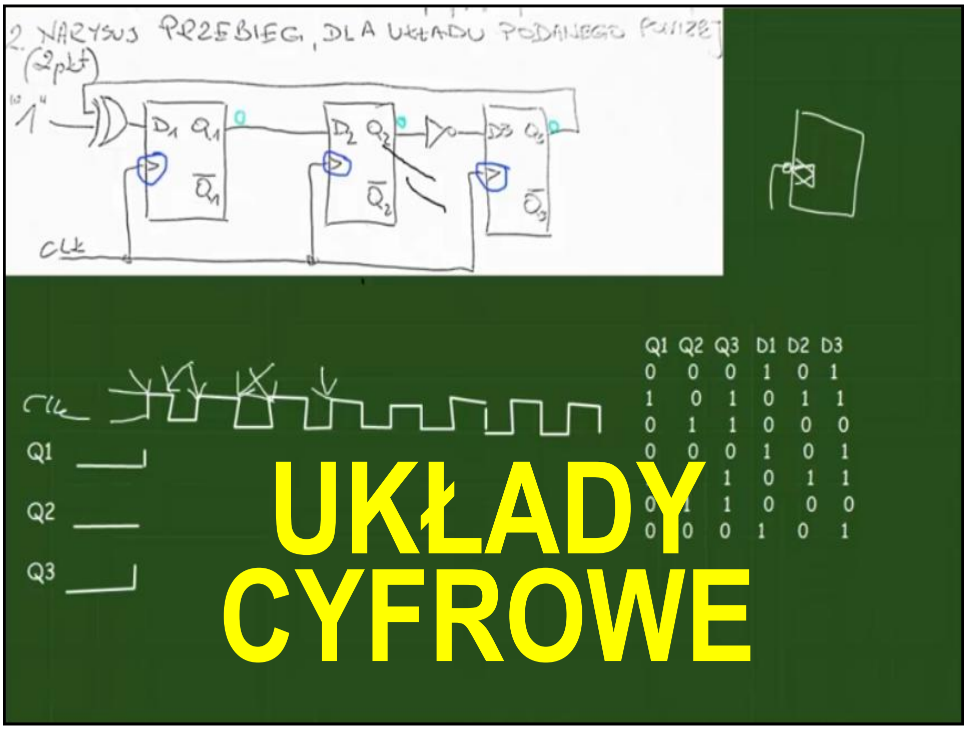
## filtracja cyfrowa




## Przykład 2. Transmisja sygnału cyfrowego w paśmie telefonicznym. Modulacja QAM.




16-Quadrature AM




## układy cyfrowe

**Układy cyfrowe** to rodzaj układów elektronicznych w których sygnały napięciowe przyjmują tylko określoną liczbę poziomów. Najczęściej (choć nie zawsze) liczba poziomów jest równa dwa, a poziomom przypisywane są stany 0 i 1.

Zalety układów cyfrowych:

- Możliwość bezstratnego kodowania i przesyłania informacji – jest to coś, czego w układach analogowych operujących na nieskończonej liczbie poziomów napięć nie sposób zrealizować.
- Zapis i przechowywanie informacji cyfrowej jest prostsze.
- Mniejsza wrażliwość na zakłócenia elektryczne.
- Możliwość tworzenia układów programowalnych

Wady układów cyfrowych:

- Są skomplikowane zarówno na poziomie elektrycznym, jak i logicznym
- Chociaż są bardziej odporne na zakłócenia, to wykrywanie przekłamań stanów logicznych, np. pojawienie się stanu 0 zamiast spodziewanego 1, wymaga dodatkowych zabezpieczeń. Jeszcze większy problem stanowi ewentualne odtworzenie oryginalnej informacji.

## układ cyfrowy


Sygnał WE: X = <$x$1, ... $xn$>, sygnał WY: Y = <y1, ... $ym$>
***Sygnał binarny*** – sygnał cyfrowy o dwuelementowym zbiorze wartości.
***Dwie wartości sygnału binarnego*** –„*logiczne zero*" oraz „*logiczne jeden*".
Reprezentacja fizyczna 0 / 1 (sygnał elektryczny) może być różna. Np.:

- zakresy napięć w TTL (i technologiach nowszych):
  „0"→ 0 ÷ 0.8V
  „1"→ 3 ÷ 5V
- standard transmisji szeregowej:
  „0"→ + 5 ÷ 15V
  „1"→ - 5 ÷ -15V
- automatyka przemysłowa:
  - sygnały prądowe (mniej podatne na zakłócenia)

## układ cyfrowy

**Układy logiczne to dział *techniki cyfrowej*, w której układy cyfrowe konstruowane są na poziomie *bramek logicznych i przerzutników*.**

**Kombinacyjne**
(przełączające, bez pamięci)

stan wyjść układu zależy wyłącznie od aktualnego stanu wejść, a nie od historii zmian stanów

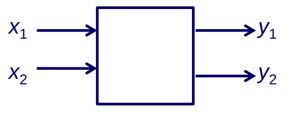
**Sekwencyjne**
(z pamięcią)

odpowiedź układu zależy od historii otrzymanych pobudzeń, układ „pamięta" stany wejść w chwilach poprzednich. Inaczej: dla jednego i tego samego sygnału WE układ może odpowiadać różnie, w zależności od tego, jakie były pobudzenia wcześniejsze.

## Kombinacyjne układy cyfrowe

- Podstawowa metoda opisu pracy układu kombinacyjnego – **tabela prawdy**
- Dla *n* wejść tabela prawdy ma $2^n$ wierszy, zwykle wypisywanych w porządku kolejnych wektorów w NKB.

Np. dla $n = m = 2$:




| $x_1\ x_2$ | $y_1\ y_2$ |
|---|---|
| 0 0 | 00 |
| 0 1 | 01 |
| 1 0 | 01 |
| 1 1 | 10 |

- Pracy układu sekwencyjnego tak opisać nie można.

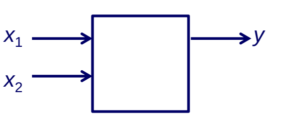
## Definicje podstawowych operacji

- NEAGACJA

| $x$ | $\bar{x}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

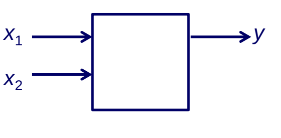
- SUMA LOGICZNA (OR)

| $x_1\ x_2$ | y |
|---|---|
| 0 0 | 0 |
| 0 1 | 1 |
| 1 0 | 1 |
| 1 1 | 1 |




- ILOCZYN LOGICZNY (AND).

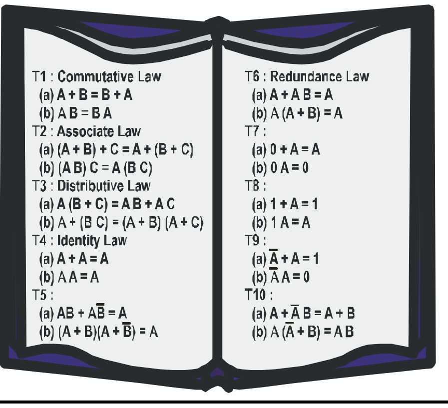
| $x_1\ x_2$ | y |
|---|---|
| 0 0 | 0 |
| 0 1 | 0 |
| 1 0 | 0 |
| 1 1 | 1 |

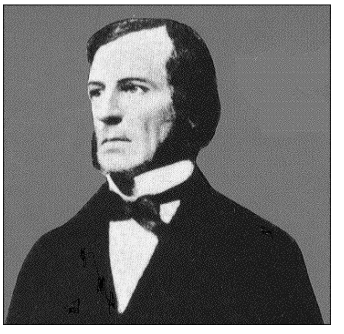


## algebra Boole'a

Do opisu działania układów logicznych zastosowano pierwotnie klasyczny rachunek zdań.

1847 - *Analiza matematyczna logiki* George'a Boole'a,



## Algebra Boole'a

Za historycznie pierwszą prace dotyczącą teorii sieci przełączających uważana jest praca Shannona *A Symbolic Analysis of Relay and Switching Circuits* (1938)

Ideą było opracowanie systemu matematycznego pozwalającego na systematyczny opis i analizę układów przełączających.

Shannon zastosował klasyczny rachunek zdań.

Istnieje system algebraiczny, który zawiera aparat formalny do badania i syntezy sieci przełączających, równoważny rachunkowi zdań. To ***algebra Boole'a***

## Algebra Boole'a
## opis aksjomatyczny

Algebrą Boole'a jest system:

BOOL= <***B*, *0,1*, ¯ , +, ·**>

gdzie

***B*** jest zbiorem i ma co najmniej dwa różne elementy

0 i 1 są wyróżnionymi stałymi nazywanymi zerem i jedynka boolowską

¯ jest działaniem jednoargumentowym nazywanym dopełnieniem (negacją)

+, · są działaniami dwuargumentowymi (dodawaniem i mnożeniem)

i dla dowolnych a, b i c $\in$ ***B*** spełnione są następujące aksjomaty

1. (a + b) + c = a + ( b + c) *łączność operacji* +
2. (a · b) · c = a · ( b · c) *łączność operacji* ·
3. a + b = b + a *przemienność operacji* +
4. a · b = b · a *przemienność operacji* ·
5. (∃!0)a + 0 = 0 + a = a *istnienie elementu neutralnego operacji* +
6. (∃!1)a · 1 = 1 · a = a *istnienie elementu neutralnego operacji* ·
7. a + (b · c) = (a+b) · (a+c) *rozdzielność* + *względem* ·
8. a · (b + c) = (a · b) + (a · c) *rozdzielność* · *względem* +
9. $a + \overline{a} = 1$ i $a \cdot \overline{a} = 0$ *istnienie dopełnienia*

## Algebra Boole'a
### wyrażenia boolowskie

**Definicja**. Wyrażenie boolowskie generowane przez zmienne $x_1,...,x_n$ jest określone w następujący sposób:

(1) 0 i 1 są wyrażeniami boolowskimi

(2) jeżeli $x_i$ jest zmienną to $x_i$ jest wyrażeniem boolowskim dla i =1,...,n

(3) jeżeli A jest wyrażeniem boolowskim to $\overline{A}$ jest wyrażeniem boolowskim

(4) jeżeli A i B są wyrażeniami boolowskimi to A · B jest wyrażeniem boolowskim

(5) jeżeli A i B są wyrażeniami boolowskimi to A + B jest wyrażeniem boolowskim

Przykład. Zbadajmy, czy $A = x_1 \cdot (\overline{x_2 + \overline{x_3}})$ jest wyrażeniem boolowskim.

# Algebra Boole'a
## przekształcenia wyrażeń boolowskich

$$F = x*\overline{y}*z + x*\overline{y}*\overline{z} = x*\overline{y}*\underbrace{(z+\overline{z})}_{1} = x*\overline{y}$$

$$F = \overline{x}*y + \overline{z}*(\overline{x}*y + z) = \overline{x}*y + \overline{z}*\overline{x}*y + \underbrace{\overline{z}*z}_{0} =$$
$$= \overline{x}*y*\underbrace{(1+\overline{z})}_{1} = \overline{x}*y$$

$$F = x*y + y*\overline{z} + \overline{x}*\overline{z} = x*y + y*\overline{z}*\underbrace{(x+\overline{x})}_{1} + \overline{x}*\overline{z} =$$
$$= x*y + x*y*\overline{z} + \overline{x}*y*\overline{z} + \overline{x}*\overline{z} =$$
$$= x*y*\underbrace{(1+\overline{z})}_{1} + \overline{x}*\overline{z}*\underbrace{(y+1)}_{1} = x*y + \overline{x}*\overline{z}$$

# Algebra Boole'a
## prawa de Morgana i przekształcenia wyrażeń boolowskich

Wychodząc z aksjomatów algebry Boole'a można udowodnić prawdziwość wyrażeń, zwanych **prawami deMorgana**

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

# Algebra Boole'a
## prawa de Morgana i przekształcenia wyrażeń boolowskich

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

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

Przekształcimy wyrażenie $\overline{x\oplus y}$

$$x\oplus y \overset{df}{=} \bar{x}\cdot y + x\cdot\bar{y}$$

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

Prawo de Morgana

Prawo de Morgana

Aksjomat 8

Aksjomat 9

Prawo de Morgana

# Algebra Boole'a
## prawa de Morgana i przekształcenia wyrażeń boolowskich

Znajdź najprostszą postać funkcji.

$$F = x\cdot z + \bar{x}\cdot\bar{y}\cdot z + y\cdot z$$

$$F = \bar{x} + x\cdot y + x\cdot\bar{z} + x\cdot\bar{y}\cdot z$$

Przekształć, wykorzystując prawa de Morgana, poniższe funkcje do postaci zawierającej jedynie operatory sumy i negacji.

$$F = \bar{x}\cdot\bar{y} + \bar{x}\cdot z + \bar{y}\cdot z$$

$$F = (y+\bar{z})\cdot(x+y)\cdot(\bar{y}+z)$$

## Algebra Boole'a
### prawa de Morgana i przekształcenia wyrażeń boolowskich

Znajdź najprostszą postać funkcji.

$$F = x \cdot z + \bar{x} \cdot \bar{y} \cdot z + y \cdot z = z(x + \bar{x} \cdot \bar{y} + y) =$$

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

## Algebra Boole'a
### prawa de Morgana i przekształcenia wyrażeń boolowskich

Znajdź najprostszą postać funkcji.

$$F = \bar{x} + x \cdot y + x \cdot \bar{z} + x \cdot \bar{y} \cdot z \quad = \bar{x} + x \cdot (y + \bar{z} + \bar{y} \cdot z)$$

$$= \bar{x} + x \cdot (y + \bar{z} + \overline{y + \bar{z}}) = \bar{x} + x \quad = 1$$

# Algebra Boole'a
## prawa de Morgana i przekształcenia wyrażeń boolowskich

Przekształć, wykorzystując prawa de Morgana, poniższe funkcje do postaci zawierającej jedynie operatory sumy i negacji.

$$F = \bar{x}\cdot\bar{y} + \bar{x}\cdot z + \bar{y}\cdot z = \overline{x+y} + \overline{x+\bar{z}} + \overline{y+\bar{z}}$$

$$F = (y+\bar{z})\cdot(x+y)\cdot(\bar{y}+z) \quad = \overline{\overline{(y+\bar{z})\cdot(x+y)\cdot(\bar{y}+z)}}$$

$$= \overline{\overline{(y+\bar{z})} + \overline{(x+y)} + \overline{(\bar{y}+z)}}$$

# Algebra Boole'a
## zasada dualizmu

Jeśli w dowolnej tożsamości algebry Boole'a zastąpimy:

- działania: „+" przez „•" i „•" przez „+" oraz
- stałe: 0 na 1 i 1 na 0

to otrzymamy także prawdziwą tożsamość.

# Algebra Boole'a
## funkcje boolowskie

**Definicja**. *Funkcją boolowską* lub *funkcją przełączającą* $f(x_1,\ldots,x_n)$ nazywamy każde odwzorowanie

$$f : D^n \to \{0,1\}$$

gdzie dziedzina funkcji $D^n \subseteq \{0,1\}^n = \{0,1\}\times\{0,1\}\times \ldots \times \{0,1\}$

Ponieważ zbiór $\{0,1\}^n$ ma $2^n$ elementów a zbiór $\{0,1\}$ dwa elementy to
**Twierdzenie**. Istnieje $2^{2^n}$ funkcji przełączających n zmiennych

Dla funkcji boolowskich zdefiniowane są operacje $+$, $\cdot$ , oraz $\overline{\phantom{x}}$ .

$D^n = \{0,1\}^n \Leftrightarrow f$ nazywamy *funkcją zupełną*.
$D^n \subset \{0,1\}^n \Leftrightarrow f$ nazywamy *funkcją niezupełną*.
W praktyce funkcje niezupełne występują często, np. gdy niektóre kombinacje wejściowe w pracy układu nie pojawiają się. Fakt ten wykorzystuje się przy minimalizacji (upraszczaniu) układu cyfrowego.

# Algebra Boole'a
## funkcje boolowskie

Funkcję boolowską można przedstawić w formie tzw *tablicy prawdy*

**Przykład 1**.
Mamy funkcję $f(x_1, x_2, x_3)$ zdefiniowaną poniżej

| $x_1$ | $x_2$ | $x_3$ | f |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

# Algebra Boole'a
## funkcje boolowskie

Funkcję boolowską można przedstawić w formie tzw *tablicy prawdy*

**Przykład 2.** Tablica prawdy ***funkcji niezupełnej*** .

| $x_1$ | $x_2$ | $x_3$ | f |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | - |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | - |
| 1 | 1 | 0 | - |
| 1 | 1 | 1 | 1 |

# Algebra Boole'a
## funkcje boolowskie

- W praktyce ***funkcje boolowskie*** reprezentowane są przez ***wyrażenia boolowskie***.
- Zwykle istnieje wiele wyrażeń boolowskich reprezentujących tę samą funkcję. Jeżeli $x_1,\ldots,x_n$ są zmiennymi boolowskimi to wyrażenie boolowskie będące konkatenacją wszystkich zmiennych lub ich negacji nosi nazwę ***iloczynu elementarnego***. (np. dla n=3 $x_1 \cdot x_2 \cdot x_3$, $x_1 \cdot x_2 \cdot x_3$)
- Podobnie wyrażenie boolowskie będące sumą wszystkich zmiennych lub ich negacji nosi nazwę ***sumy elementarnej***. (np. dla n=3 $x_1 + x_2 + x_3$, $x_1 + x_2 + x_3$)
- Jedna wyróżniona forma zapisu wyrażenia nosi nazwę ***postaci kanonicznej***. Istnieją dwie równoważne postaci kanoniczne***: kanoniczna postać sumy***
  - $f(x_1,\ldots,x_n) = \Sigma_{i \in I_1} P_i$

  oraz ***kanoniczna postać iloczynu***:
  - $f(x_1,\ldots,x_n) = \Pi_{i \in I_0} S_i$
- gdzie $I_1$ i $I_0$ są zbiorami tych indeksów, dla których $f(x_1,\ldots,x_n)$ przyjmuje odpowiednio wartość 1 lub wartość 0, $P_i$ są iloczynami elementarnymi a $S_i$ elementarnymi sumami.

# Algebra Boole'a

## funkcje boolowskie

*Przykład*

| $x_1$ | $x_2$ | $x_3$ | f |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

*Funkcję boolowską* przedstawioną wyżej opisuje odpowiadające jej wyrażenie kanoniczne sumy.

$f(x,y,z)=0\cdot\overline{x}_1\overline{x}_2\overline{x}_3 + 0\cdot\overline{x}_1\overline{x}_2x_3 + 1\cdot\overline{x}_1x_2\overline{x}_3 + 0\cdot\overline{x}_1x_2x_3 + 1\cdot x_1\overline{x}_2\overline{x}_3 + 1\cdot x_1\overline{x}_2x_3 + 0\cdot x_1x_2\overline{x}_3$
$+ 1\cdot x_1x_2x_3$

$f(x,y,z)=\overline{x}_1x_2\overline{x}_3 + x_1\overline{x}_2\overline{x}_3 + x_1\overline{x}_2x_3 + x_1x_2x_3$

# Algebra Boole'a

## funkcje boolowskie

*Przykład*

| $x_1$ | $x_2$ | $x_3$ | f |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

lub równoważne mu wyrażenie kanoniczne iloczynu

$f(x,y,z)=(x_1+x_2+x_3)(x_1+x_2+\overline{x}_3)(x_1+\overline{x}_2+\overline{x}_3)(\overline{x}_1+\overline{x}_2+x_3)$

# Algebra Boole'a

## funkcje boolowskie

*Przedstawiony sposób konstruowania* wyrażenia kanonicznego sumy uzasadnia twierdzenie Shannona

Tw.

Dowolną funkcje boolowską $f(x_1,\ldots,x_n)$ można przedstawić w postaci:

$$
\begin{aligned}
f(x_1, x_2, \ldots, x_n) = {} & f(1,1,1,\ldots,1)x_1 x_2 \ldots x_n + \\
& + f(0,1,1,\ldots,1)\overline{x_1} x_2 \ldots x_n + \\
& + f(1,0,1,\ldots,1)x_1 \overline{x_2} \ldots x_n + \\
& + f(0,0,1,\ldots,1)\overline{x_1}\,\overline{x_2} \ldots x_n + \\
& + \ldots \\
& + f(0,0,0,\ldots,0)\overline{x_1}\,\overline{x_2} \ldots \overline{x_n}
\end{aligned}
$$

# Algebra Boole'a

## funkcje boolowskie

Podobnie jest dla wyrażenia kanonicznego iloczynu

Dowolną funkcje boolowską f($x_1,\ldots,x_n$) można przedstawić w postaci:

$$
\begin{aligned}
f(x_1, x_2, \ldots, x_n) = {} & [f(1,1,1,\ldots,1) + \overline{x_1} + \overline{x_2} + \ldots + \overline{x_n}] \\
& \cdot [f(0,1,1,\ldots,1) + x_1 + \overline{x_2} + \ldots + \overline{x_n}] \cdot \\
& \cdot [f(1,0,1,\ldots,1) + \overline{x_1} + x_2 + \ldots + \overline{x_n}] \cdot \\
& \cdot [f(0,1,1,\ldots,1) + x_1 + x_2 + \ldots + \overline{x_n}] \cdot \\
& \cdot \ldots \\
& \cdot [f(0,0,0,\ldots,0) + x_1 + x_2 + \ldots + x_n]
\end{aligned}
$$

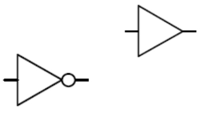
# Algebra Boole'a
## funkcje boolowskie

- Użyte operacje *sumy, iloczynu i negacji* stanowią ***system funkcjonalnie pełny,*** to znaczy, że dla każdej funkcji logicznej można utworzyć wyrażenie ją reprezentujące zapisane za pomocą tylko tych operacji.
- Istnieją inne systemy funkcjonalnie pełne, np.
  - operacja NAND
  - operacja NOR

# Algebra Boole'a
## funkcje boolowskie

Funkcje jednej zmiennej:

| | |
|---|---|
| *f(x)=0* | *Stała zero* |
| *f(x)=x* | *powtórzenie* |
| *f(x)=x* | *negacja* |
| *f(x)=1* | *Stała jeden* |



# Algebra Boole'a
## funkcje boolowskie

funkcje dwóch zmiennych

| $x_1$: 0 0 1 1 / $x_2$: 0 1 0 1 | | Wzór | Oznaczenie | Nazwa | Symbol | Uwagi |
|---|---|---|---|---|---|---|
| $f_0$ | 0 0 0 0 | 0 | – | Stała zero | – | – |
| $f_1$ | 0 0 0 1 | $x_1 \cdot x_2$ | $x_1 \cdot x_2$ | Iloczyn | (1) | AND, koniunkcja |
| $f_2$ | 0 0 1 0 | $x_1 \cdot \overline{x_2}$ | $x_1 \Delta x_2$ | Zakaz przez $x_2$ | – | – |
| $f_3$ | 0 0 1 1 | $x_1$ | $x_1$ | Zmienna $x_1$ | – | – |
| $f_4$ | 0 1 0 0 | $\overline{x_1} \cdot x_2$ | $x_2 \Delta x_1$ | Zakaz przez $x_1$ | – | – |
| $f_5$ | 0 1 0 1 | $x_2$ | $x_2$ | Zmienna $x_2$ | – | – |
| $f_6$ | 0 1 1 0 | $x_1 \cdot \overline{x_2} + \overline{x_1} \cdot x_2$ | $x_1 \oplus x_2$ | Różnica symetryczna | (2) | EX-OR , nierównoważność, suma modulo 2 |
| $f_7$ | 0 1 1 1 | $x_1 + x_2$ | $x_1 + x_2$ | Suma | (3) | OR, alternatywa |

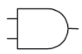
# Algebra Boole'a
## funkcje boolowskie

funkcje dwóch zmiennych

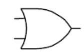
| | | | | | | |
|---|---|---|---|---|---|---|
| $f_8$ | 1 0 0 0 | $\overline{x_1} \cdot \overline{x_2}$ | $\overline{x_1 + x_2}$ | Negacja sumy | (4) | Funkcja Pierce'a, NOR |
| $f_9$ | 1 0 0 1 | $x_1 \cdot x_2 + \overline{x_1} \cdot \overline{x_2}$ | $x_1 \equiv x_2$, $x_1 \odot x_2$ | Równoważność | (5) | EX-NOR |
| $f_{10}$ | 1 0 1 0 | $\overline{x_2}$ | $\overline{x_2}$ | Negacja $x_2$ | (j.w.) | – |
| $f_{11}$ | 1 0 1 1 | $x_1 + \overline{x_2}$ | $x_2 \rightarrow x_1$ | Implikacja | – | – |
| $f_{12}$ | 1 1 0 0 | $\overline{x_1}$ | $\overline{x_1}$ | Negacja $x_1$ | (j.w.) | – |
| $f_{13}$ | 1 1 0 1 | $\overline{x_1} + x_2$ | $x_1 \rightarrow x_2$ | Implikacja | – | – |
| $f_{14}$ | 1 1 1 0 | $\overline{x_1} + \overline{x_2}$ | $\overline{x_1 \cdot x_2}$ | Negacja iloczynu | (6) | Funkcja Sheffera, NAND |
| $f_{15}$ | 1 1 1 1 | 1 | – | Stała jeden | – | – |

# Symbole bramek logicznych

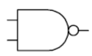
(1) Bramka AND:



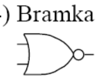
(3) Bramka OR:



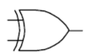
(6) Bramka NAND:



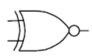
(4) Bramka NOR:



(2) Bramka EX-OR:

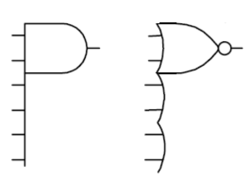


(5) Bramka EX-NOR:



| $x_1$ | $x_2$ | $x_1 \oplus x_2$ | $x_1 \odot x_2$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

# Funktory wielowejściowe

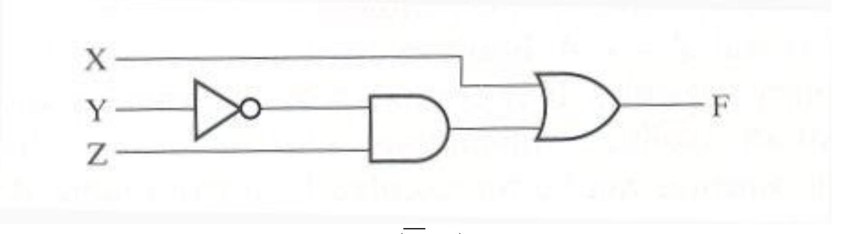


- Wielowejściowe symbole dla sumy modulo 2 (EX-OR) stosuje się rzadko.

## Realizacja techniczna - bramki i układy logiczne

Układy zbudowane z bramek logicznych noszą nazwę układów (sieci) logicznych

Przykład:




$$F = X + (\overline{Y} \cdot Z)$$

## Realizacja techniczna - bramki i układy logiczne

Przykład: $F = x * z + \overline{x} * \overline{y} * z + y * z$