



Arytmetyka komputerów

wer. 10 z drobnymi modyfikacjami!

Wojciech Myszka

2023-10-16 18:06:12 +0200



Politechnika Wroclawska

Liczby binarne

Liczby dwójkowe nie są wcale nowym wynalazkiem:

- ▶ Pierwsze wzmianki pochodzą z Indii, z 5–2 w. pne.
- ▶ W Chinach księga I Ching składa się z 64 heksagramów i gdzieś tak w XI w. ponumerowano je (od 0 do 63) z użyciem yin jako 0, yang jako 1.
- ▶ System opisał bardzo dokładnie matematyk, Goetfryd Leibnitz; patrz na przykład: <http://books.google.pl/books?id=Fuk8AAAAcAAJ&printsec=frontcover#v=onepage&q&f=false>. Inspirował się I Ching.



System dwójkowy

System dwójkowy to pozycyjny system zapisu liczb używający jako bazy (podstawy) liczby 2.

1. Dwie cyfry: 0 i 1.
2. W systemie dziesiętnym (przy podstawie 10) cyfr jest 10: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
3. W systemie ósemkowym (przy podstawie 8) cyfr jest osiem: 0, 1, 2, 3, 4, 5, 6, 7.
4. W systemie szesnastkowym (przy podstawie 16) cyfr jest 16: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Przykłady:

10	2	16
10	1010	A
100	1100100	64
123,75	1111011,11	7B,C



Czemu system szesnastkowy jest ważny?

- ▶ Każda cyfra dwójkowa to jeden bit (od **binary digit**). W skrócie b (małe b).
- ▶ Osiem bitów to jeden bajt. W skrócie B (wielkie B).
- ▶ Komputery zwykle używają parzystych wielokrotności bajtów do zapisu liczb (dwa, cztery albo osiem).
- ▶ Każda cyfra szesnastkowa to cztery bity, zatem jeden bajt to dwie cyfry szesnastkowe.
- ▶ Stosunkowo łatwo zapamiętać binarny wygląd wszystkich cyfr szesnastkowych. . .



Duże liczby I

W układzie dziesiętnym używamy przedrostków:

- ▶ kilo (10^3), mega (10^6), giga (10^9),... do oznaczania dużych liczb lub
- ▶ mili (10^{-3}), mikro (10^{-6}), nano (10^{-9}),... do zapisu małych liczb

2^{10} to 1024 i przez analogię zaczęto nazywać to „kilo” (1024 bajtów to 1 „kilo-bajt”).

Ale nie jest to prawidłowe!

W końcu ustandaryzowano nazwy (norma IEC 60027-2:1998):

kibi	Ki	2^{10}
mebi	Mi	2^{20}
gibi	Gi	2^{30}
tebi	Ti	2^{40}

pebi	Pi	2^{50}
eksbi	Ei	2^{60}
zebi	Zi	2^{70}
jobi	Yi	2^{80}



Konwersje

Liczby całkowite

Liczby całkowite:

Liczbę dzielimy przez dwa zapisując reszty z dzielenia:

10 |



Konwersje

Liczby całkowite

Liczby całkowite:

Liczbę dzielimy przez dwa zapisując reszty z dzielenia:

$$\begin{array}{r|l} 10 & \\ 5 & 0 \end{array}$$



Konwersje

Liczby całkowite

Liczby całkowite:

Liczbę dzielimy przez dwa zapisując reszty z dzielenia:

$$\begin{array}{r|l} 10 & \\ 5 & 0 \\ 2 & 1 \end{array}$$



Konwersje

Liczby całkowite

Liczby całkowite:

Liczbę dzielimy przez dwa zapisując reszty z dzielenia:

$$\begin{array}{r|l} 10 & \\ 5 & 0 \\ 2 & 1 \\ 1 & 0 \end{array}$$



Konwersje

Liczby całkowite

Liczby całkowite:

Liczbę dzielimy przez dwa zapisując reszty z dzielenia:

10		
5		0
2		1
1		0
0		1



Konwersje

Liczby całkowite

Liczby całkowite:

Liczbę dzielimy przez dwa zapisując reszty z dzielenia:

10		
5		0
2		1
1		0
0		1

Reszty z dzielenia zapisujemy „od końca” otrzymując 1010



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

| ,33



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

$$\begin{array}{r|l} & ,33 \\ 0 & ,66 \end{array}$$



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28
0		,56



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28
0		,56
1		,12



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28
0		,56
1		,12
0		,24



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28
0		,56
1		,12
0		,24
0		,48



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28
0		,56
1		,12
0		,24
0		,48
0		,96



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

	,33
0	,66
1	,32
0	,64
1	,28
0	,56
1	,12
0	,24
0	,48
0	,96
1	,92



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28
0		,56
1		,12
0		,24
0		,48
0		,96
1		,92
1		,84



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

	,33
0	,66
1	,32
0	,64
1	,28
0	,56
1	,12
0	,24
0	,48
0	,96
1	,92
1	,84
1	,68



Konwersje

Ułamki dziesiętne

Ułamki dziesiętne:

Liczbę mnożymy przez dwa i zapisujemy część całkowitą:

		,33
0		,66
1		,32
0		,64
1		,28
0		,56
1		,12
0		,24
0		,48
0		,96
1		,92
1		,84
1		,68

$0,010101000111_2 = 0,329833984375_{10}$



Konwersje

Dwójkowy do dziesiętneho

Do zrobienia w domu!



Procesor

Operacje logiczne

Iloczyn $Y = A \cap B$

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



Procesor

Operacje logiczne

Suma $Y = A \cup B$

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



Procesor

Operacje logiczne

Różnica symetryczna $Y = A \oplus B$

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



Procesor

Operacje logiczne

Iloczyn $Y = A \cap B$

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

Różnica symetryczna $Y = A \oplus B$

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

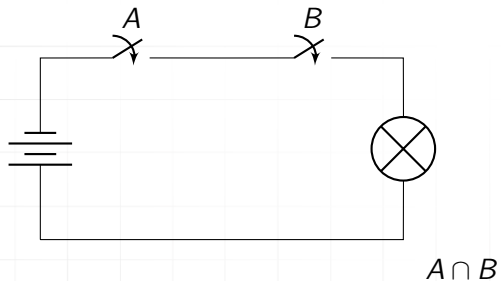
Suma $Y = A \cup B$

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



Operacje logiczne

Iloczyn logiczny

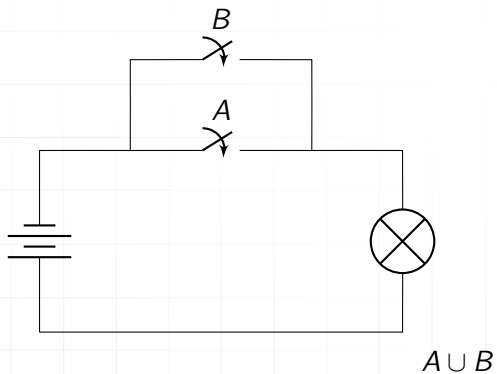


AND	0	1
0	0	0
1	0	1



Operacje logiczne

Suma logiczna



OR	0	1
0	0	1
1	1	1



Operacje logiczne

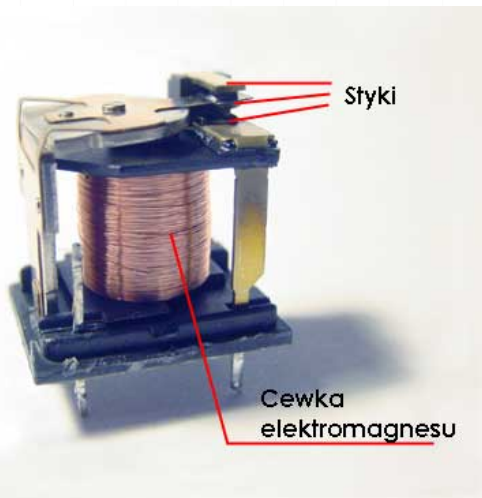
XOR

XOR	0	1
0	0	1
1	1	0

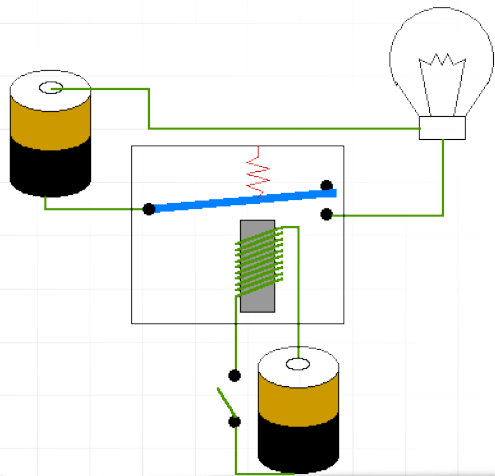
$$(\bar{A} \cap B) \cup (A \cap \bar{B})$$



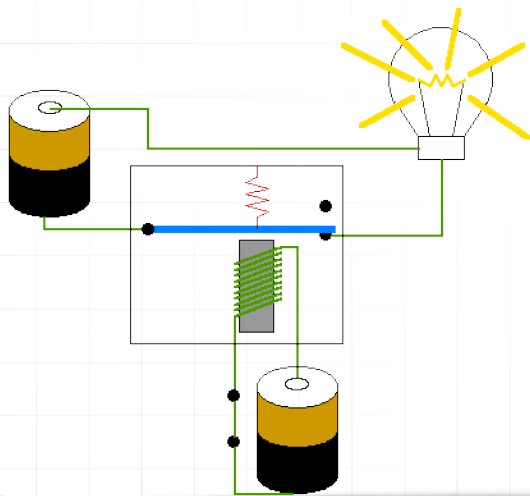
Przełącznik



Przełącznik

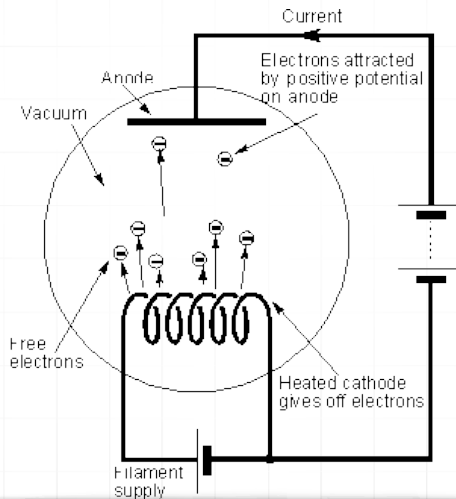


Przełącznik



Lampy elektronowe

Dioda



Rysunek: Schemat działania diody próżniowej



Lampy elektronowe

Dioda

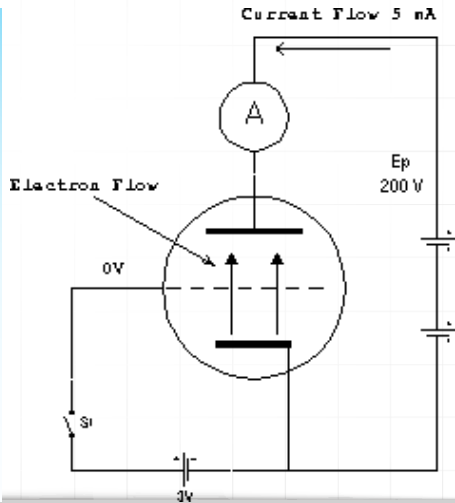
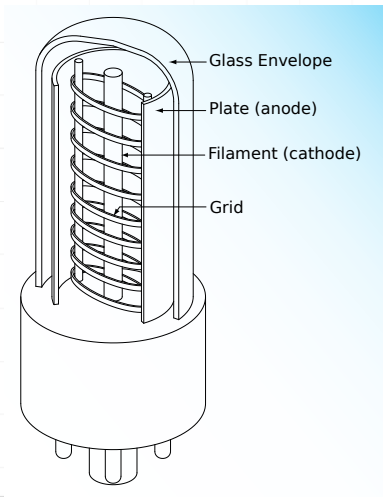


Rysunek: Rzeczywisty wygląd diody półprzewodnikowej DY87 używanej w pierwszych telewizorach do uzyskania wysokiego stałego napięcia do zasilania kineskopu



Lampy elektronowe

Trioda



Rysunek: Schemat działania triody



Binarne operacje arytmetyczne

Oto „tablice prawdy” dla dodawania i mnożenia:

1. Suma:

- ▶ $0 + 0 = 0$
- ▶ $0 + 1 = 1$
- ▶ $1 + 0 = 1$
- ▶ $1 + 1 = 10$



Binarne operacje arytmetyczne

Oto „tablice prawdy” dla dodawania i mnożenia:

1. Suma:

- ▶ $0 + 0 = 0$
- ▶ $0 + 1 = 1$
- ▶ $1 + 0 = 1$
- ▶ $1 + 1 = 10$

2. Iloczyn

- ▶ $0 * 0 = 0$
- ▶ $0 * 1 = 0$
- ▶ $1 * 0 = 0$
- ▶ $1 * 1 = 1$



Procesor

Jak procesor dodaje?

1. „Półsumator”
2. Suma dwu bitów ($Y = X_1 + X_2$)
3. Przeniesienie (C_{out})
4. „Tabela prawdy”

X_1	X_2	Y	C_{out}
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$Y = X_1 \oplus X_2$$

$$C_{out} = X_1 \cap X_2$$



Procesor

Jak procesor dodaje — sumator

C_{in}	X_1	X_2	Y	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$Y = C_{in} \oplus (X_1 \oplus X_2)$$

$$C_{out} = (X_1 \cap X_2) \cup (C_{in} \cap (X_1 \oplus X_2))$$



Operacje logiczne

Zastosowanie XOR

Znak

		■		
		■		
■	■	■	■	■
		■		
		■		

Kursor

■	■	■	■	■
■	■	■	■	■
■	■	■	■	■
■	■	■	■	■
■	■	■	■	■

Wynik



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik

1				



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik

1	1			



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik

1	1	0		



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik

1	1	0	1	



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik

1	1	0	1	1



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik

1	1	0	1	1



Operacje logiczne

Zastosowanie XOR

Znak

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Kursor

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Wynik

1	1	0	1	1
1	1	0	1	1
0	0	0	0	0
1	1	0	1	1
1	1	0	1	1



Operacje logiczne

Zastosowanie XOR

Znak

		■		
		■		
■	■	■	■	■
		■		
		■		

Kursor

■	■	■	■	■
■	■	■	■	■
■	■	■	■	■
■	■	■	■	■
■	■	■	■	■

Wynik

■	■		■	■
■	■		■	■
■	■		■	■
■	■		■	■



XOR — Patent nonsense

Method for dynamically viewing image elements stored in a random access

Patent number: 4197590

Filing date: Jan 19, 1978

Issue date: Apr 8, 1980

Inventors: Josef S. Sukonick, Greg J. Tilden

Assignees: NuGraphics, Inc.

Primary Examiner: Thomas M. Heckler



Zapis liczb w komputerze

- ▶ Jedna cyfra dwójkowa to **bit**: **B**inary **digiT**. Jest to ten sam bit co w przypadku pomiaru ilości informacji.
- ▶ Bity grupowane są po osiem; osiem bitów to **bajt**
- ▶ 00000000 to 0 (zero)
- ▶ 11111111 to 255 ($1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 1 * 2^3 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0$
 $= 2^8 - 1$)
- ▶ bajty bywają grupowane po
 - ▶ dwa (komputer szesnastobitowy)
 - ▶ cztery — trzydziestodwubitowy
 - ▶ osiem — sześćdziesięcioczworo-bitowy



Liczby ujemne?

1. W systemie dziesiętnym: $+3$ albo 3 , a ujemne: -3
2. W dwójkowym teoretycznie: $+00000011$ albo -00000011
3. Ale jak zapisać znaki $+$ i $-$?
4. Wariant najprostszy $3 \rightarrow 00000011$
5. Wariant najprostszy $-3 \rightarrow 10000011$
6. Jak wygodnie prowadzić obliczenia na liczbach całkowitych (dodatnich i ujemnych)?



Liczby ujemne

Tablica odejmowania:

–	0	1
0	0	1
1	1	0

(Zakładamy, że operujemy na liczbach czterobitowych!)

$$0011 - 1 = 0010$$

$$0010 - 1 = 0001$$

$$0001 - 1 = 0000$$

$$0000 - 1 = 1111$$

Zatem -1 to 1111 (czterobitowo!)



Liczby ujemne

Dokonajmy prostego sprawdzenia:

$$5 + (-1)$$

$$0 \ 1 \ 0 \ 1$$

$$\underline{1 \ 1 \ 1 \ 1}$$



Liczby ujemne

Dokonajmy prostego sprawdzenia:

$$\begin{array}{r} 5 + (-1) \\ 0 \ 1 \ 0 \ 1 \\ 1 \ 1 \ 1 \ 1 \\ \hline 1 \ 0 \ 1 \ 0 \ 0 \end{array}$$



Dygresja

Liczby dziesiętne, dwucyfrowe:

$$\begin{array}{r} 3 \ 3 \\ 9 \ 9 \\ \hline \end{array}$$



Dygresja

Liczby dziesiętne, dwucyfrowe:

$$\begin{array}{r} 33 \\ 99 \\ \hline 132 \end{array}$$



Negacja liczby

Mnemotechniczny algorytm negacji jest bardzo prosty: „negujemy” (przełączamy) wszystkie bity i powstałą liczbę zwiększamy o 1:

1 to 0001

negacje: 1110

zwiększenie o 1: 1111

2 to 0010

negacja: 1101

zwiększenie o 1: 1110

sprawdzenie $5 + (-2)$

$$\begin{array}{rcccc} 0 & 1 & 0 & 1 & \\ 1 & 1 & 1 & 0 & \\ \hline 1 & 0 & 0 & 1 & 1 \end{array}$$



Mnożenie?

Ćwiczenie/Zadanie domowe

Założmy, że nasz komputer jest 5-bitowy (4 bity + znak?)

$$\begin{array}{r} 00101 \\ * 11110 \\ \hline \end{array}$$

