Algebra Boola i podstawy systemów liczbowych.

Ćwiczenia z Teorii Układów Logicznych, dr inż. Ernest Jamro

1. System dwójkowy – reprezentacja binarna

Układy logiczne operują tylko na dwóch stanach oznaczanymi jako zero (stan napięcia bliski zeru) i jedynka (stan napięcia bliski napięciu zasilania zwykle 5V lub w nowszych układach 3.3B lub nawet 1.5V). System operujący na dwóch stanach nazywamy dwójkowym lub też binarnym.

W systemie dziesiętnym kolejne cyfry od prawej strony mają wartość kolejnych potęg 10, podobnie w dwójkowym – są to kolejne potęgi dwójki: $1(2^0)$, 2, 4, 8, 16, 32, 64, 128, $256(2^8)$ itp. Czyli np. liczba $10011010_{bin}=2+8+16+128=154_{dec}$.

W celu zamiany z systemu dziesiętnego na dwójkowy, wykonujemy na liczbie dzielenie całkowite przez 2, zapisując przy tym resztę z dzielenia i powtarzamy to aż dojdziemy do 1. Kolejne reszty to cyfry reprezentacji binarnej ułożone od najmłodszej (najmniej znaczącej) do najstarszej (łącznie z końcową jedynką). Tak więc na przykład:

Liczba	Reszta	Liczba	Reszta
500	0	260	0
250	0	130	0
125	1	65	1
62	0	32	0
31	1	16	0
15	1	8	0
7	1	4	0
3	1	2	0
1	1	1	1

$500_{dec}=111110100_{bin}$ (=4+16+32+64+128+256=500)

$260_{dec}=100000100_{bin}$.

Uwaga: wynik binarny wpisujemy odczytując reszty z dzielenia patrząc od dołu do góry.

Zapiszemy tabelę dla liczb dziesiętnych, dwójkowych i szesnastkowych:

dec	bin	hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10

Widzimy, że jedna cyfra szesnastkowa odpowiada dokładnie czterem cyfrom dwójkowym. Szesnastkowy zapis liczb binarnych jest powszechnie stosowany, gdyż zamiana jest o wiele łatwiejsza niż dla liczb dziesiętnych, a zapis jest krótszy. Na przykład jeden bajt, który składa się z ośmiu bitów, można przedstawić przy pomocy dwóch znaków od 00 do FF.

W celu zamiany bin→hex grupujemy cyfry po 4 (od najmłodszego bitu), a następnie każdej grupie przypisujemy 1 cyfrę szesnastkową (np. korzystając z tabeli).

Tak więc: $101\ 1000\ 1011\ 1011\ 0010_{bin}=58BB2_{hex}$.

Zamiana w drugą stronę wygląda analogicznie:

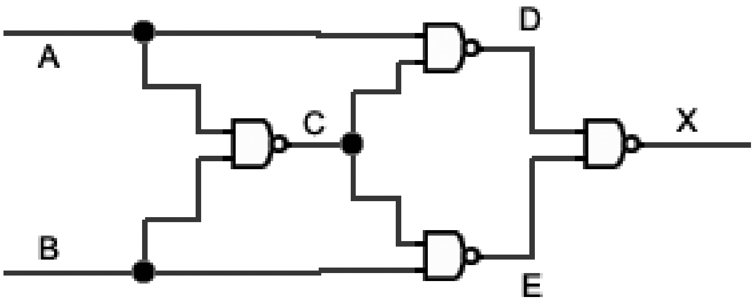
$AF8C2E_{hex}=1010\ 1111\ 1000\ 1100\ 0010\ 1110_{bin}$.

2. Bramki logiczne

Podstawą układów logicznych są bramki, realizujące pewne funkcje logiczne. Odpowiednim stanom napięć na wejściu odpowiada napięcie na wyjściu, przy czym napięcie interpretujemy jako 1, a jego brak – jako 0. (jest to tzw. logika dodatnia).
Podstawowe bramki wraz z ich symbolami:

		AND	**NAND**	**OR**	**NOR**	**XOR**		**NOT**
A	**B**	$A \cdot B$	$\overline{A \cdot B}$	$A + B$	$\overline{A + B}$	$A \oplus B$	**A**	$\overline{A}$
0	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**
0	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**
1	**0**	**0**	**1**	**1**	**0**	**1**		
1	**1**	**1**	**0**	**1**	**0**	**0**		

Przykład:
Jaką funkcję realizuje poniższy układ (wejście A,B, wyjście Y)?
Aby rozwiązać poniższe zadanie należy dodać zmienne pomocnicze C,D,E oraz sprawdzić ich stan w zależności od wszystkich możliwych kombinacji wejść.



C=NAND(A,B)
D=NAND(A,C)
E=NAND(B,C)
X=NAND(D,E)

A	B	C	D	E	X
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	0

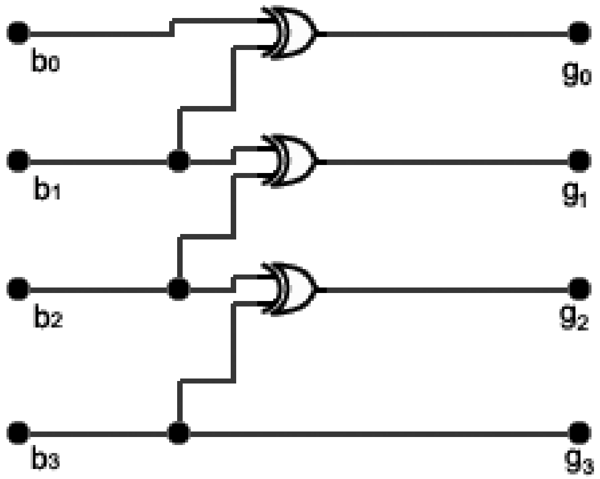
Więc po porównaniu X z A i B otrzymujemy, że **X=XOR(A,B)**, $X = A \oplus B$.

Jednym z zastosowań bramki XOR jest kontrola bitu parzystości, np. jeżeli mamy 4 bity $(a_3 a_2 a_1 a_0)_{\text{bin}}$, to $a_3 \oplus a_2 \oplus a_1 \oplus a_0 = 1$ wtedy, kiedy na bitach jest nieparzysta liczba jedynek, natomiast funkcja ta jest równa 0, jeżeli liczba jedynek (czyli bitów zapełnionych) jest parzysta. Można to łatwo sprawdzić rozpisując tabelę dla np. 4 zmiennych wejściowych.

$a_3a_2a_1a_0$	Bit parzystości XOR(a_3,a_2,a_1,a_0)		$a_3a_2a_1a_0$	Bit parzystości XOR(a_3,a_2,a_1,a_0)
0000	0		1000	1
0001	1		1001	0
0010	1		1010	0
0011	0		1011	1
0100	1		1100	0
0101	0		1101	1
0110	0		1110	1
0111	1		1111	0

Przy pomocy bramek XOR można opisać **kod Gray'a**:

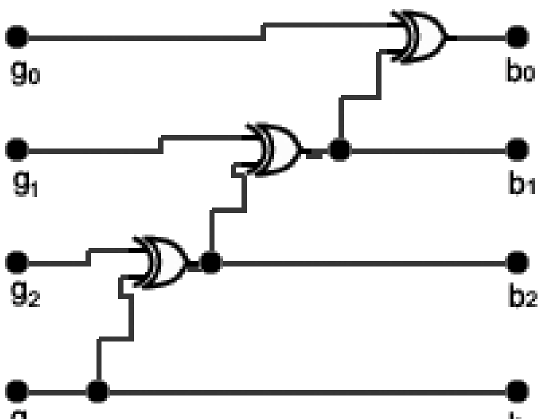
	$b_3b_2b_1b_0$	$g_3g_2g_1g_0$
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	**1001**	**1101**
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000



Układ ten zamienia kod binarny na kod Gray'
Na przykład dla liczby **1001** mamy:
$b_3=1, b_2=0, b_1=0, b_0=1$.
$g_3 = b_3 = 1$, $g_2 = b_3 \oplus b_2 = 1 \oplus 0 = 1$, $g_1 = b_2 \oplus b_1 = 0 \oplus 0 = 0$,
$g_0 = b_1 \oplus b_0 = 0 \oplus 1 = 1$, więc w kodzie Gray'a jest ona przedstawiana jako **1101**.

Jak łatwo zauważyć, następujące po sobie liczby przedstawione w kodzie Gray'a różnią się tylko jednym bitem (na przykład dla liczb 11 i 12 jest to 1**110** i 1**0**10). Znajdzie to później zastosowanie między innymi przy minimalizacji. Zazwyczaj kolejne stany bitów przedstawiamy w sposób pokazany w pierwszej kolumnie (b), bo odpowiada to numeracji w systemie dwójkowym, jednak przedstawianie ich w sposób pokazany w drugiej kolumnie (g) - czyli w kodzie Gray'a - powoduje, że zmiana stanu bitów na kolejny pociąga za sobą zmianę tylko jednego bitu, co – jak się później okaże – pozwala tworzyć układy logiczne w bardziej ekonomiczny sposób.

A to jest układ konwertujący kod Gray'a na kod binarny:



3. Algebra Boole'a

Podstawowe twierdzenia algebry Boole'a:

$$a + 0 = a \qquad a \cdot 0 = 0$$

$$a + 1 = 1 \qquad a \cdot 1 = a$$

$$a + a = a \qquad a \cdot a = a$$

$$a + \overline{a} = 1 \qquad a \cdot \overline{a} = 0$$

$$a + a \cdot b = a \cdot (1 + b) = a \cdot 1 = a \qquad a \cdot (a + b) = a + ab = a$$

$$a + \overline{a} \cdot b = (a + \overline{a})(a + b) = a + b \qquad a \cdot (\overline{a} + b) = a \cdot b$$

$$\overline{(a + b)} = \overline{a} \cdot \overline{b} \qquad \overline{(a \cdot b)} = \overline{a} + \overline{b}$$

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

Można je łatwo udowodnić korzystając z tablic prawdy dla poszczególnych funkcji. Jak widać, operacje dodawania (OR) i mnożenia (AND) logicznego podlegają takim samym prawom rozdzielności jak zwykłe dodawanie i mnożenie, ale mają też kilka nietypowych własności. Można np. udowodnić wzór na $a + \overline{a} \cdot b = a + b$:

a	b	$\overline{a}$	$\overline{a} \cdot b$	$a + \overline{a} \cdot b$
0	0	1	0	0
0	1	1	1	1
1	0	0	0	1
1	1	0	0	1

Przykład wykorzystania twierdzeń: zminimalizować funkcję $a(\overline{a} + ab + bc)$.

$$a(\overline{a} + ab + bc) = a \cdot \overline{a} + a \cdot a \cdot b + a \cdot b \cdot c = 0 + a \cdot b + a \cdot b \cdot c = a \cdot b \cdot (c + 1) = a \cdot b$$

Albo też następującą funkcję:

$$a \cdot b \cdot \overline{c} + \overline{b} \cdot \overline{c} + \overline{a \cdot b} + \overline{a} \cdot c = a \cdot b \cdot \overline{c} + \overline{b} \cdot \overline{c} + \overline{a} + \overline{b} + \overline{a} \cdot c = ab\overline{c} + \overline{a}(c + 1) + \overline{b}(\overline{c} + 1) = \overline{b} + \overline{a} + ab\overline{c} =$$

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

Ten przykład może na pierwszy rzut oka wydać się nieco zamieszany – skorzystaliśmy w nim dwukrotnie z twierdzenia, że $A + \overline{A}B = A + B$, $\overline{a} + ab\overline{c} = \overline{a} + (\overline{\overline{a}})(b\overline{c}) = \overline{a} + b\overline{c}$

Można też zająć się pierwszym przykładem, gdzie: C=NAND(A,B), D=NAND(A,C), E=NAND(B,C), X=NAND(D,E).

$$X = \overline{DE} = \overline{(\overline{AC})(\overline{BC})} = AC + BC = A(\overline{AB}) + B(\overline{AB}) = (A + B)(\overline{AB}) = (A + B)(\overline{A} + \overline{B}) =$$

$$= A\overline{A} + A\overline{B} + B\overline{A} + B\overline{B} = 0 + A\overline{B} + B\overline{A} + 0 = A\overline{B} + \overline{A}B$$

Jak widzieliśmy wcześniej funkcja ta odpowiada funkcji XOR, można więc zauważyć, że:

$$A \oplus B = A \cdot \overline{B} + B \cdot \overline{A}$$

Uproszczenia dla funkcji XOR:

$a \oplus 0 = a$

$a \oplus 1 = \bar{a}$

$a \oplus a = 0$

$a \oplus \bar{a} = 1$

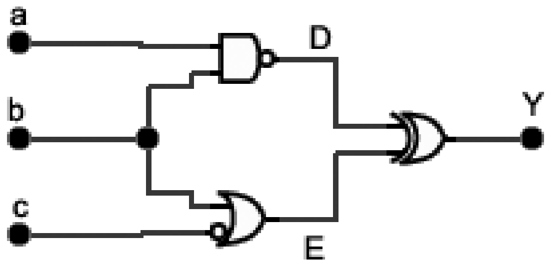
$\bar{a} \oplus b = a \cdot b + \bar{a} \cdot \bar{b}$

$\bar{a} \oplus b = \overline{a \oplus b} = a \oplus \bar{b}$

Przykład: zminimalizować $(a \oplus b) + a \cdot b$

$(a \oplus b) + a \cdot b = a\bar{b} + \bar{a}b + ab = a(b + \bar{b}) + \bar{a}b = a + \bar{a}b = a + b$

Można też zminimalizować układ zadany w formie schematu:



$Y = D \oplus E = (\overline{ab}) \oplus (b + \bar{c}) = (\overline{ab})(\overline{b + \bar{c}}) + (ab)(b + \bar{c}) = (\bar{a} + \bar{b})\bar{b}c + ab + ab\bar{c} =$

$= \bar{a}\bar{b}c + \bar{b}c + ab(1 + \bar{c}) = (\bar{a} + 1)\bar{b}c + ab = a \cdot b + \bar{b} \cdot c$

Można też rozpisać tablicę prawdy dla funkcji Y(a,b,c) zadanych na oba sposoby:

a	b	c	$\bar{c}$	D	E	Y
0	0	0	1	1	1	0
0	0	1	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	1	0
1	0	0	1	1	1	0
1	0	1	0	1	0	1
1	1	0	1	0	1	1
1	1	1	0	0	1	1

a	b	c	ab	$\bar{b}c$	$Y = ab + \bar{b}c$
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	0	1

Sprawdziliśmy więc równoważność tych wyrażeń.

Ćwiczenia nr 1 z Teorii Układów Logicznych
Prowadzący: dr inż. Ernest Jamro
Opracował: Daniel Starnowski