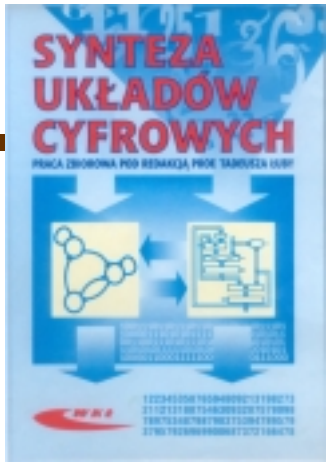
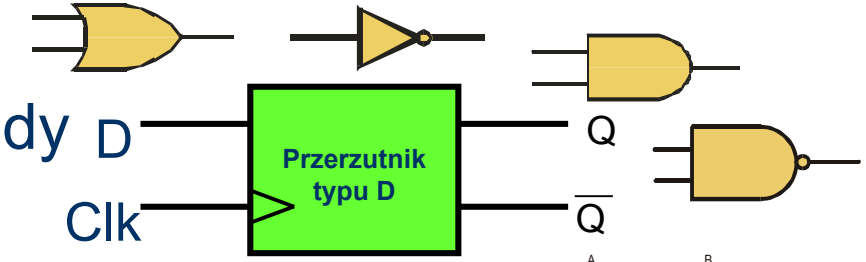
Układy cyfrowe



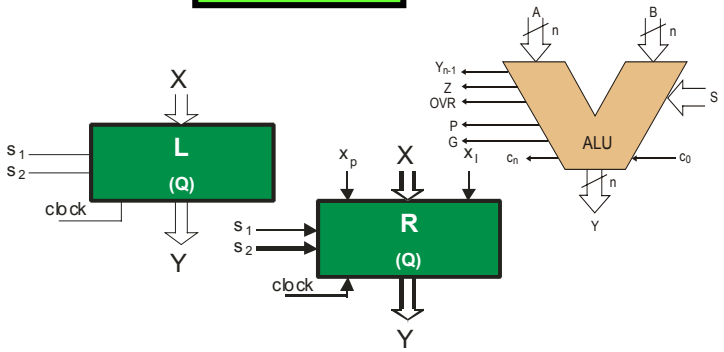
...konstruowane są w różnych technologiach i na różnych poziomach opisu.

Poziomy opisu:

1) Bramki i elementarne układy pamięciowe (przerzutniki)



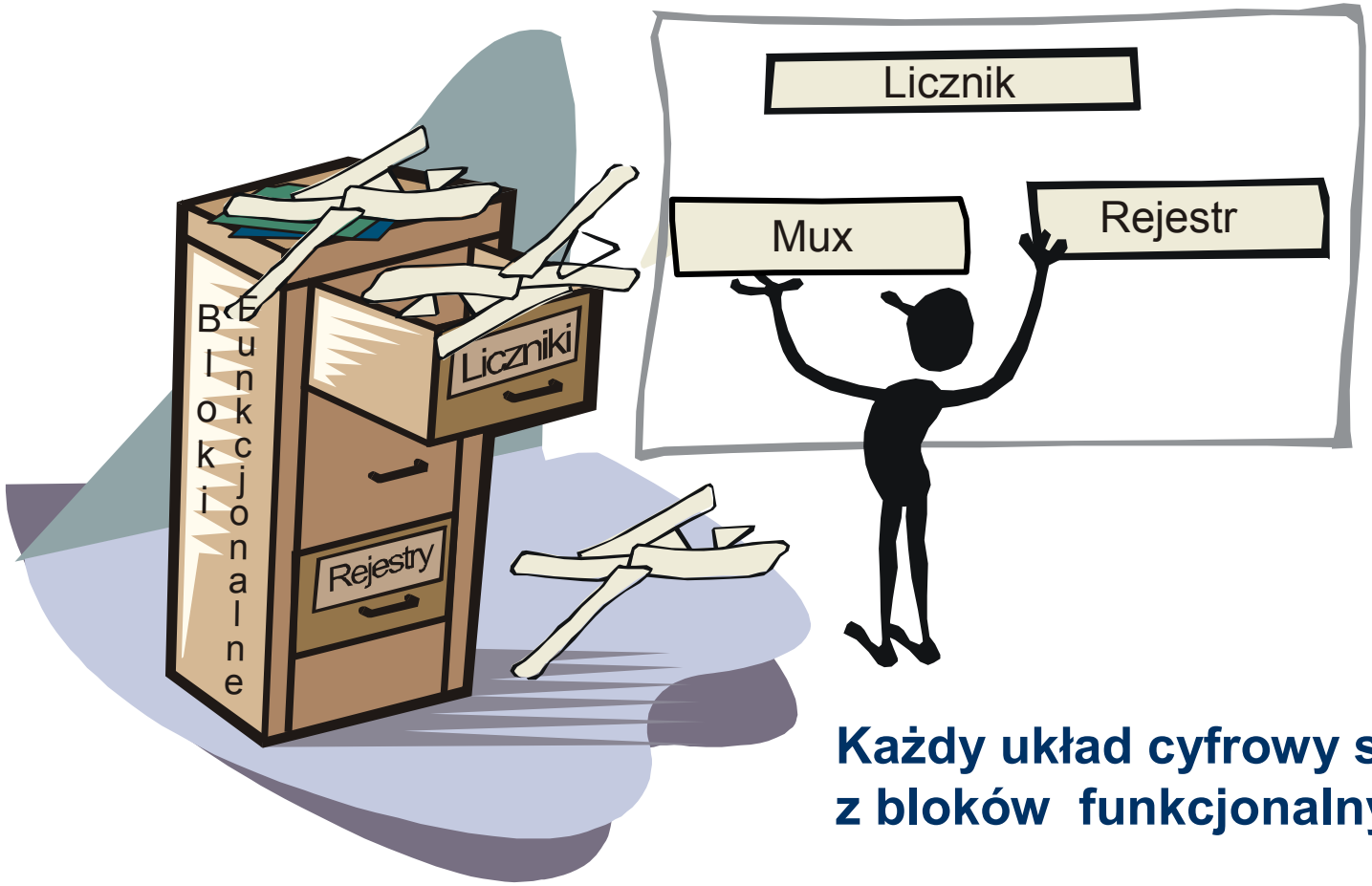
2) Bloki funkcjonalne: układy arytmetyczne (sumatory), liczniki, rejestry.



Tworzą one nowe elementy konstrukcyjne, z których buduje się złożone układy cyfrowe o różnorodnych zastosowaniach: układy przetwarzania sygnałów, układy sterowania, specjalizowane procesory, układy kryptograficzne

I T P W ZPT

Synteza strukturalna układów cyfrowych



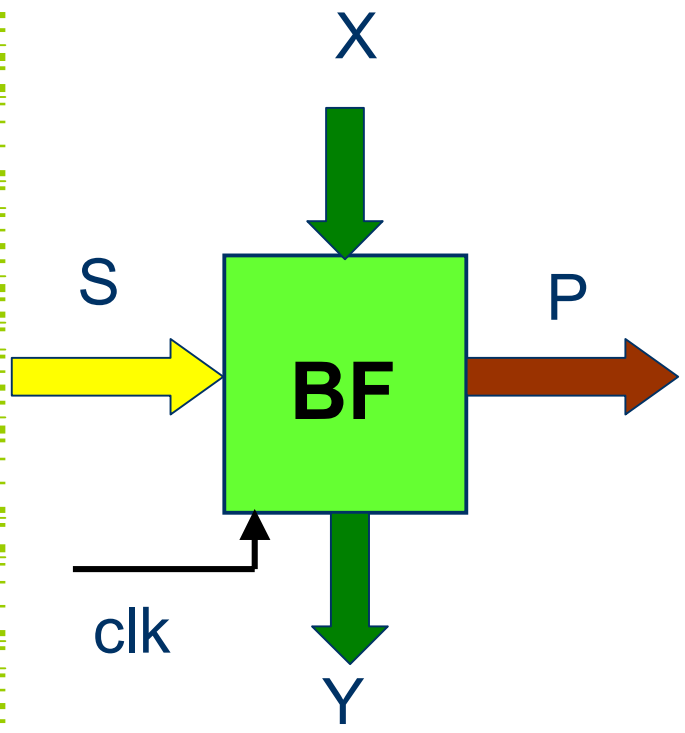
Każdy układ cyfrowy składamy z bloków funkcjonalnych

Bloki funkcjonalne stanowią wyposażenie bibliotek komputerowych systemów projektowania

I T P W ZPT

Blok funkcjonalny

…specjalizowany układ cyfrowy:



***X*, (*Y*)** – wejścia (wyjścia) sygnałów reprezentujących dane wejściowe i wyjściowe

S – wejścia sterujące,

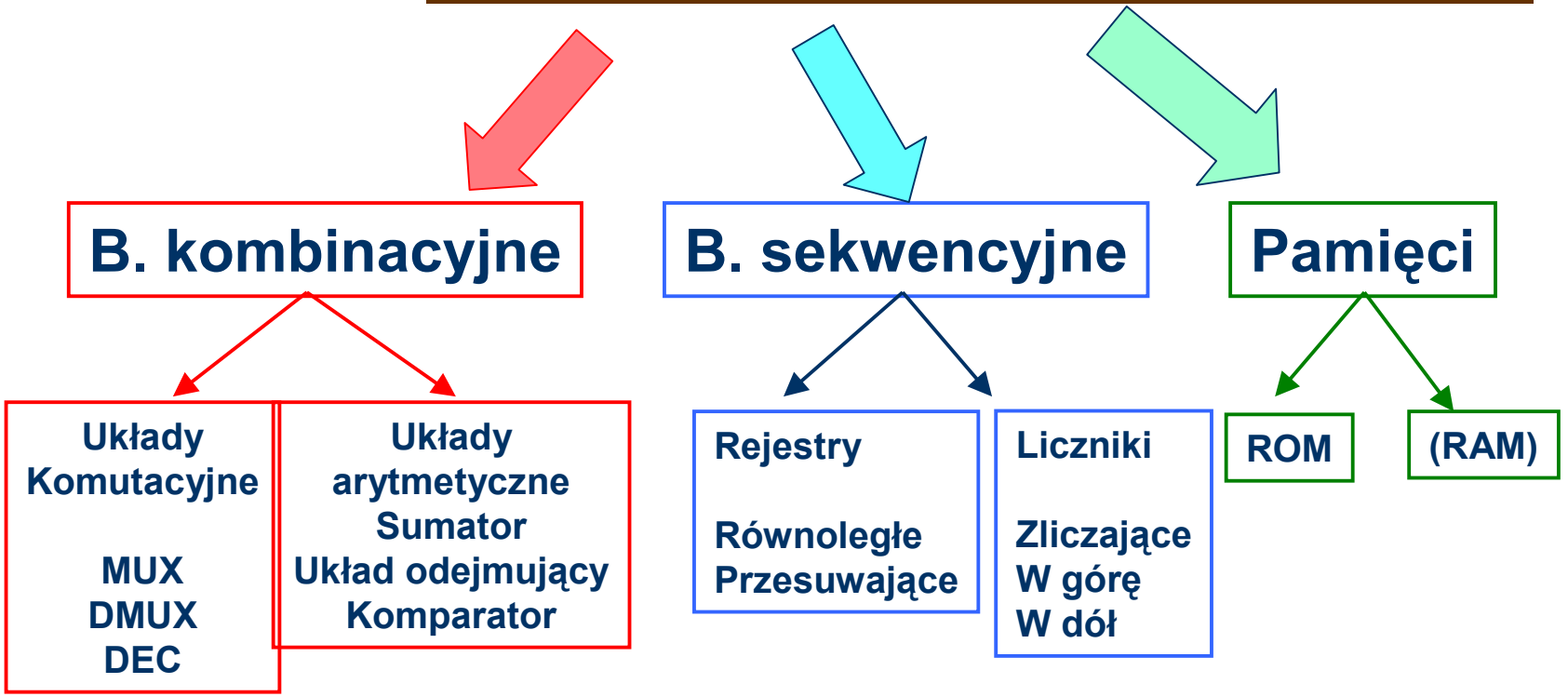
P – wyjścia predykatowe,

clk – wejście zegarowe

Bloki funkcjonalne stanowią wyposażenie bibliotek komputerowych systemów projektowania

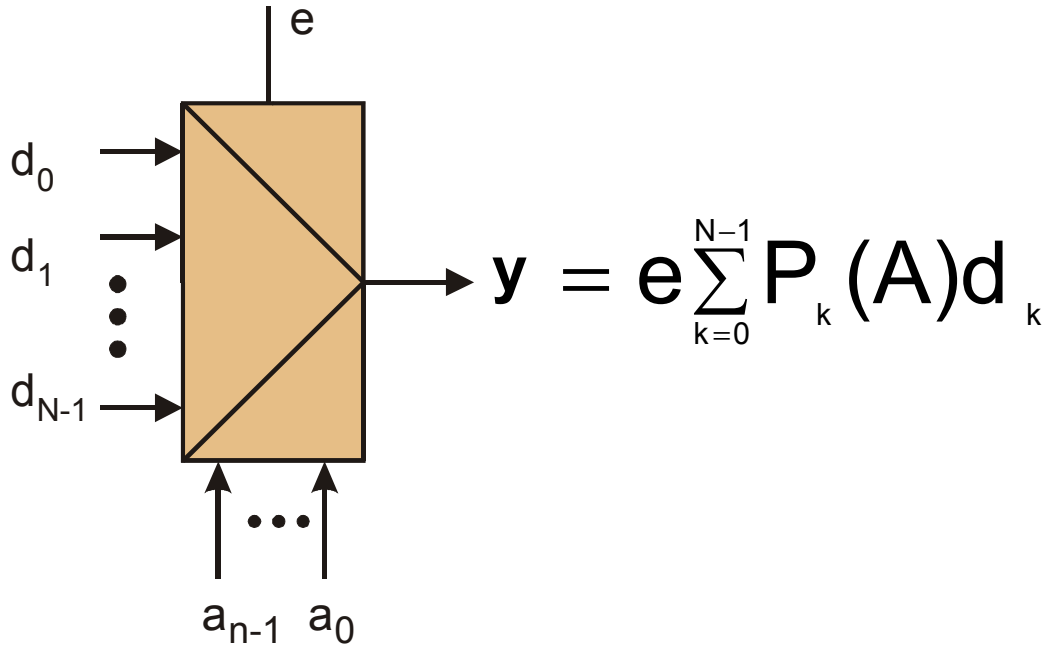
I T P W ZPT

Bloki funkcjonalne



Multiplekser (MUX)

$N = 2^n$



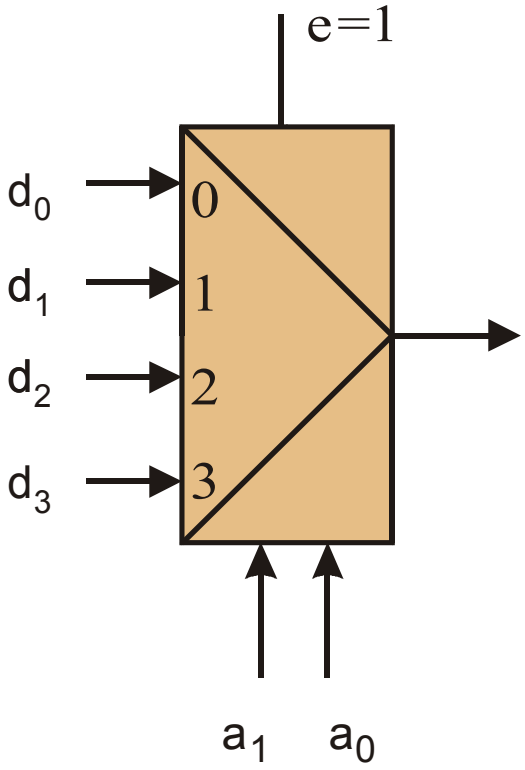
$$y = e\sum_{k=0}^{N-1} P_k(A)d_k$$

gdzie $P_k(A)$ oznacza pełny iloczyn zmiennych $a_{n-1},...,a_0$, prostych lub zanegowanych, zgodnie z reprezentacją binarną liczby k = L(A).

I T P W

ZPT

Multipleksery



Dla *n* = 1 (MUX 2 : 1):

$$y = \bar{a}d_0 + ad_1$$

dla n = 2 (MUX 4 : 1):

$$y = \bar{a}_1\bar{a}_0d_0 + \bar{a}_1a_0d_1 + a_1\bar{a}_0d_2 + a_1a_0d_3$$

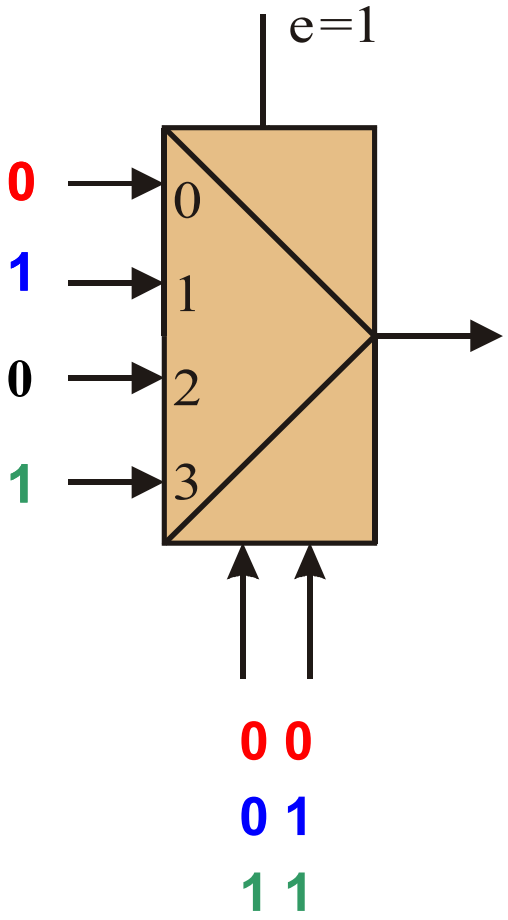
dla *n* = 3 (MUX 8 : 1):

$$y = \bar{a}_2\bar{a}_1\bar{a}_0d_0 + \bar{a}_2\bar{a}_1a_0d_1 + \bar{a}_2a_1\bar{a}_0d_2 + \bar{a}_2a_1a_0d_3 + \\ + a_2\bar{a}_1\bar{a}_0d_4 + a_2\bar{a}_1a_0d_5 + a_2a_1\bar{a}_0d_6 + a_2a_1a_0d_7$$

I T P W

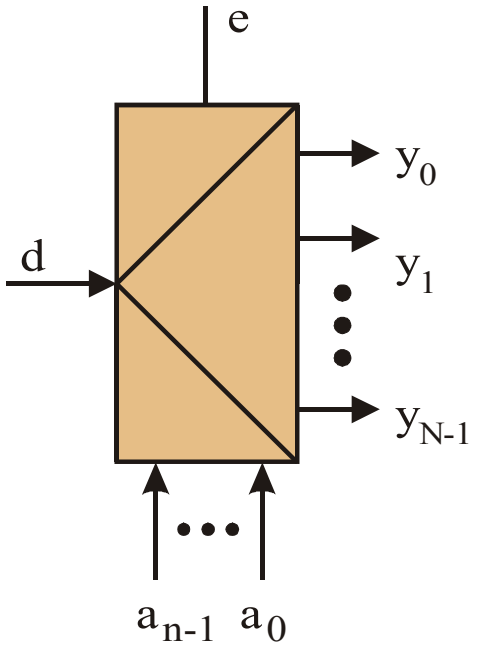
ZPT

Multiplekser jako przełącznik



$$y = \bar{a}_1\bar{a}_0d_0 + \bar{a}_1a_0d_1 + a_1\bar{a}_0d_2 + a_1a_0d_3$$

Demultiplekser



$$y_k = eP_k(A)d$$

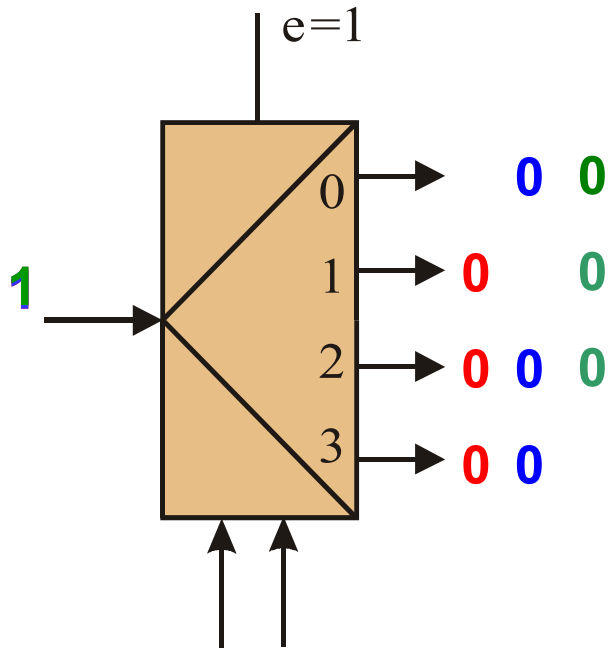
$N = 2^n$

gdzie $P_k(A)$ oznacza pełny iloczyn zmiennych $a_{n-1},...,a_0$, prostych lub zanegowanych, zgodnie z reprezentacją binarną liczby k = L(A).

I T P W

ZPT

Demultiplekser jako przełącznik



$$y_0 = \overline{a}_1 \overline{a}_0 d$$

$$y_1 = \overline{a}_1 a_0 d$$

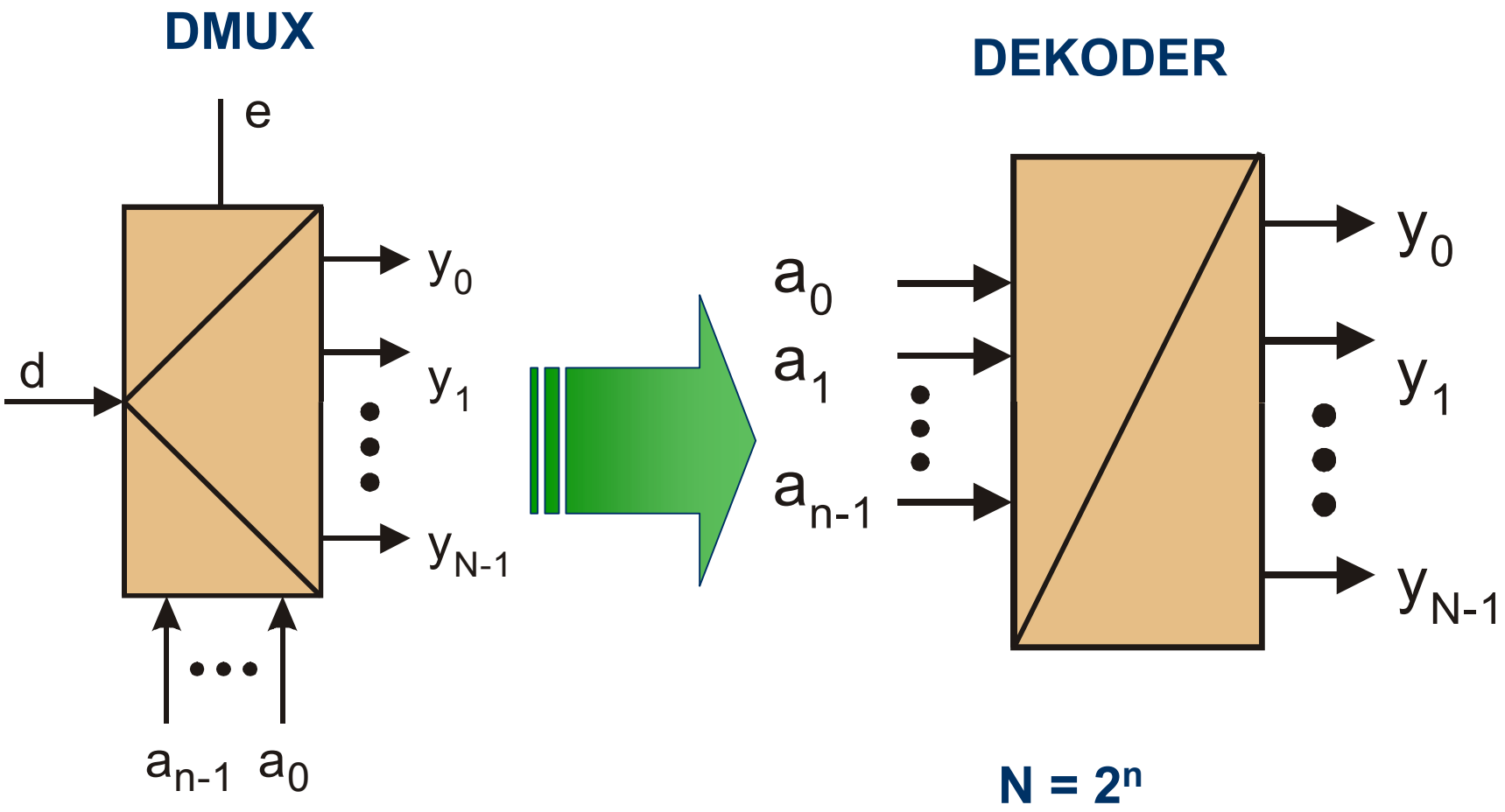
$$y_2 = a_1 \overline{a}_0 d$$

$$y_3 = a_1 a_0 d$$

I T P W

ZPT

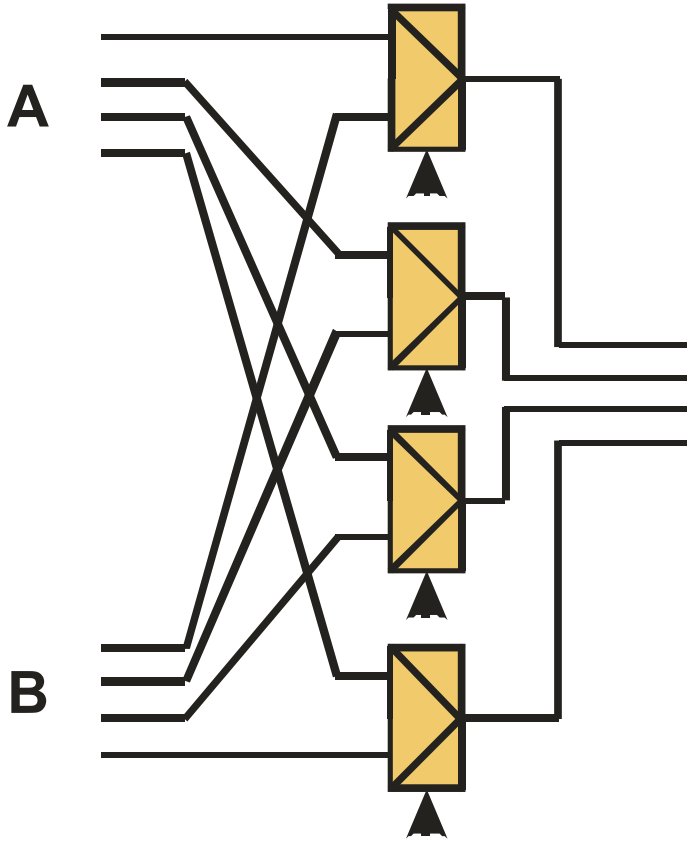
Dekoder

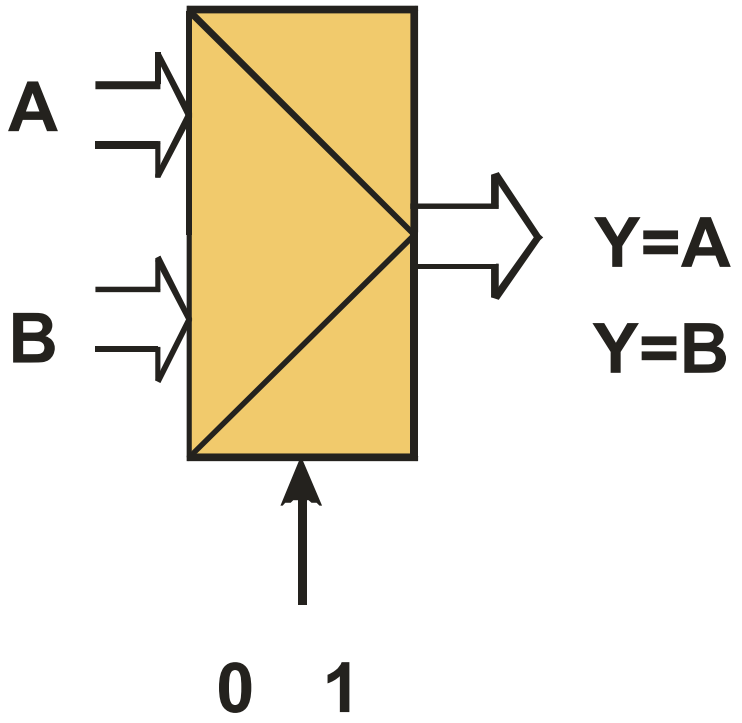


$N = 2^n$

Multipleksery grupowe

MUX-y i DMUX-y można przystosować do przełączania (komutacji) sygnałów wielobitowych (grupowych)

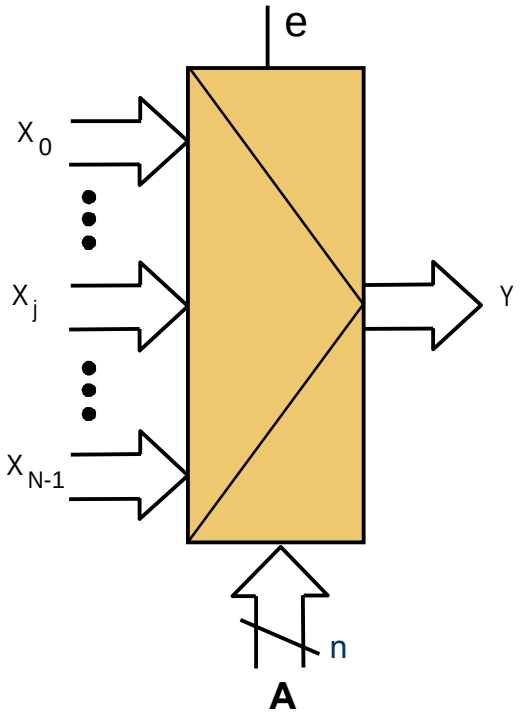




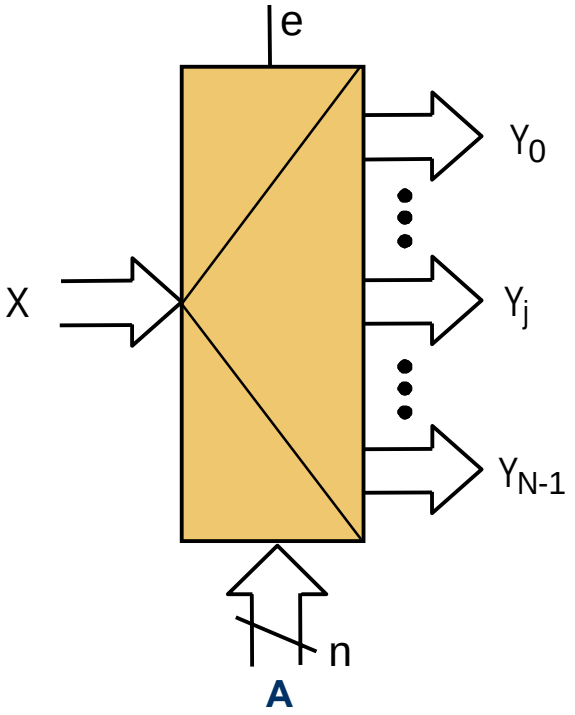
I T P W

ZPT

Bloki komutacyjne



Multiplekser służy do wybierania jednego z wielu słów wejściowych i przesyłania go na wyjście. Na wyjściu Y pojawia się słowo wejściowe wskazane adresem A (wg naturalnego kodu binarnego).

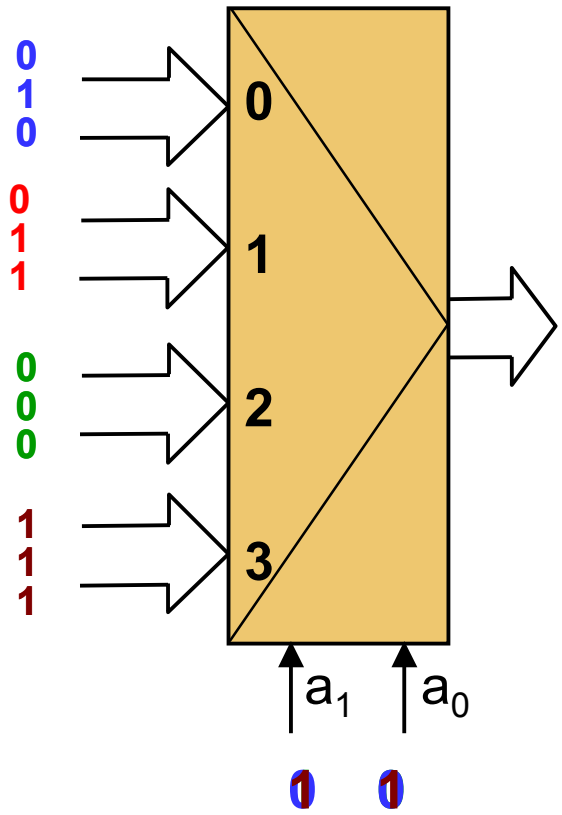


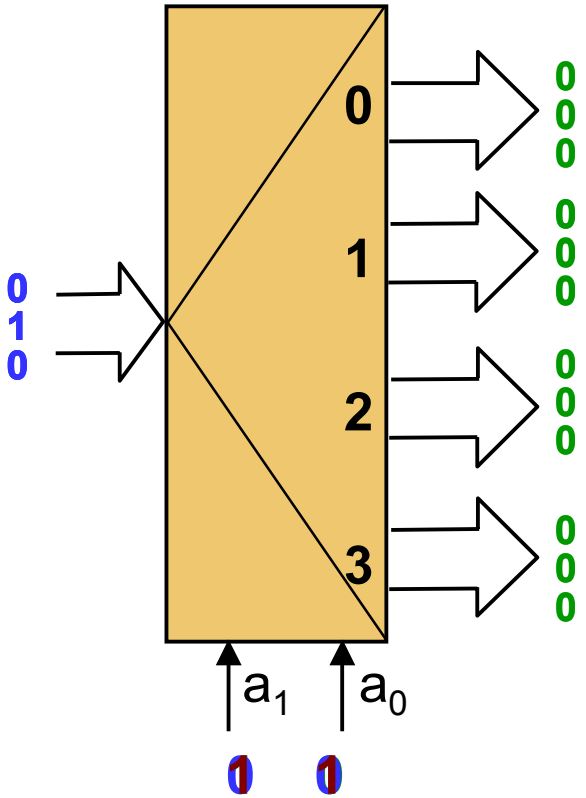
Demultiplekser służy do przesyłania słowa X wejściowego na jedno z wielu wyjść; numer tego wyjścia jest równy aktualnej wartości adresu.

I T P W

ZPT

Bloki komutacyjne



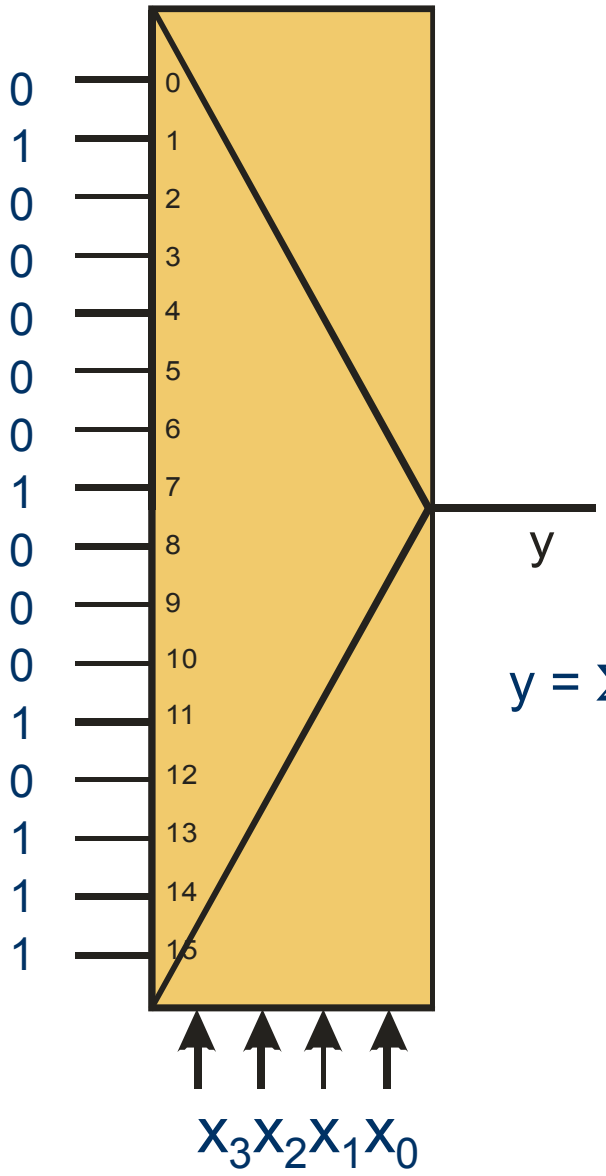


Najważniejsze zastosowanie

I T P W

ZPT

Inne zastosowania…



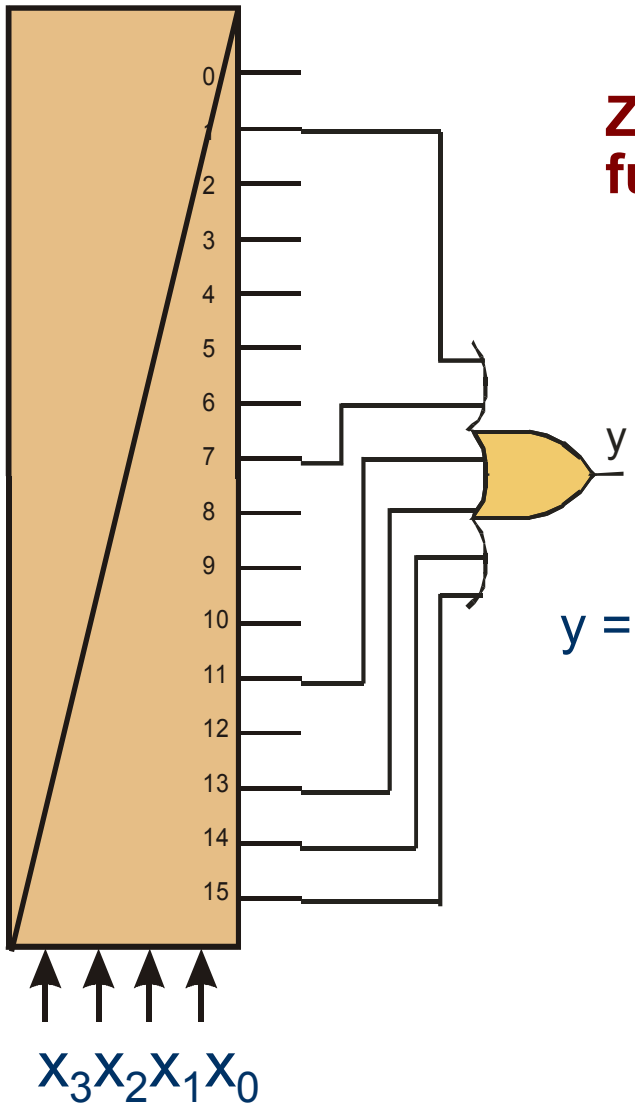
Zastosowanie MUX do realizacji funkcji boolowskich

$y = \Sigma(1,7,11,13,14,15)$

I T P W

ZPT

Inne zastosowania…



Zastosowanie dekodera do realizacji funkcji boolowskich

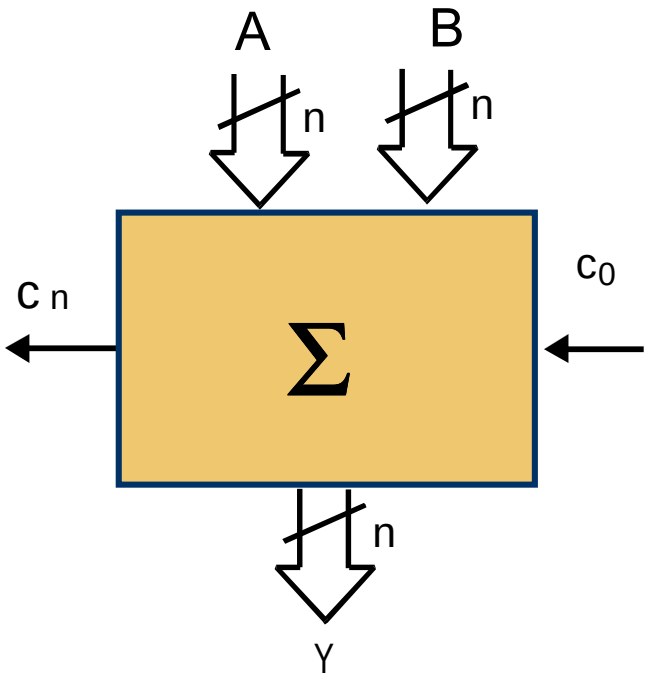
$y = \Sigma(1,7,11,13,14,15)$

… należy odłożyć do kosza!

I T P W ZPT

Sumatory

Sumator – podstawowy BF powszechnie stosowany w technice DSP



Inne układy arytmetyczne:
układy odejmowania
układy mnożące
układy dzielenia

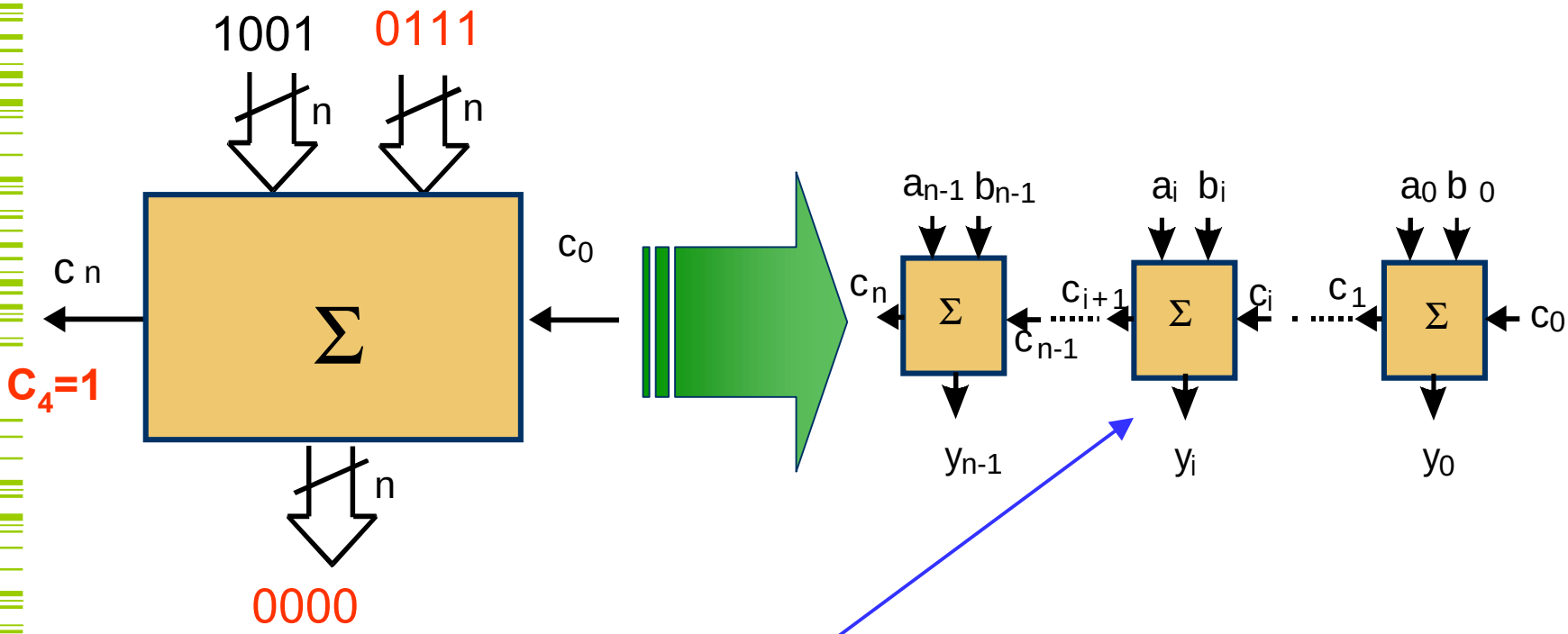
...są budowane z sumatorów

I T P W

ZPT

Najprostszy sumator

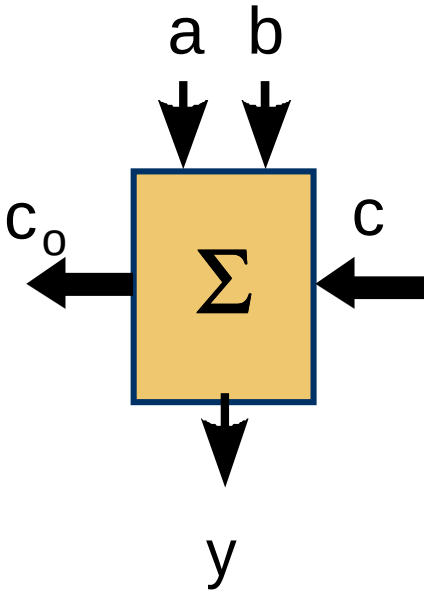
Kaskadowy – *ripple carry adder*



Jak jest zbudowane pojedyncze ogniwo?

I T P W ZPT

Funkcje logiczne sumatora



a	b	c	c_o	y
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

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

$$y = cab \vee c\bar{a}\bar{b} \vee \bar{c}\bar{a}b \vee \bar{c}a\bar{b}$$

$$= c\left(\overline{a \oplus b}\right) \vee \bar{c}(a \oplus b)$$

$$y = c \oplus a \oplus b$$

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

$$y_i = a_i \oplus b_i \oplus c_i$$

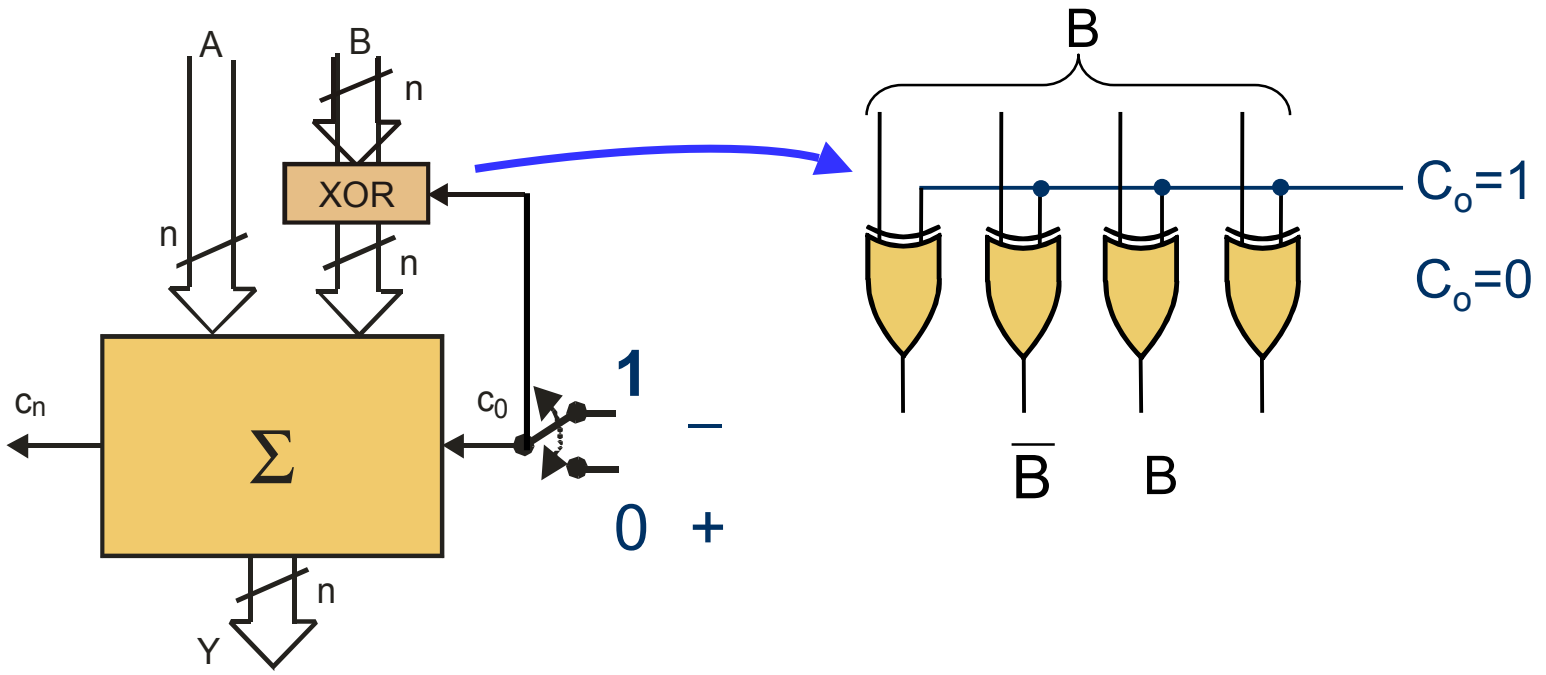
$$c_{i+1} = a_i b_i \vee c_i(a_i \vee b_i)$$

$$c_o = ab \vee c(a \vee b) = ab \vee c(a \oplus b)$$

I T P W ZPT

Sumator/układ odejmujący

Jak z sumatora zbudować układ odejmujący?



Reprezentacje liczb – NKB/U2

A = $< a_{n-1}, \ldots, a_j, \ldots, a_0 >$ gdzie $a_j \in \{0,1\}$

NKB:

$$A_D = L(A_{NKB}) = \sum_{j=0}^{n-1} a_j 2^j$$

U2:

$$A_D = L(A_{U2}) = -a_{n-1} \cdot 2^{n-1} + \sum_{j=0}^{n-2} a_j 2^j$$

I T P W

ZPT

Kod U2

$\mathbf{A}_{U2} = \langle a_{n-1}, \ldots, a_j, \ldots, a_0 \rangle$, gdzie $a_j \in \{0,1\}$

$$A_D = L(A_{U2}) = -a_{n-1} \cdot 2^{n-1} + \sum_{j=0}^{n-2} a_j 2^j$$

Bit a_{n-1} można interpretować jako bit znaku.
Jeśli $a_{n-1} = 0$, to liczba jest dodatnia;
jeśli $a_{n-1} = 1$ to liczba jest ujemna; pozostałe bity stanowią uzupełnienie (różnicę) wartości liczby do najwyższej potęgi liczby 2

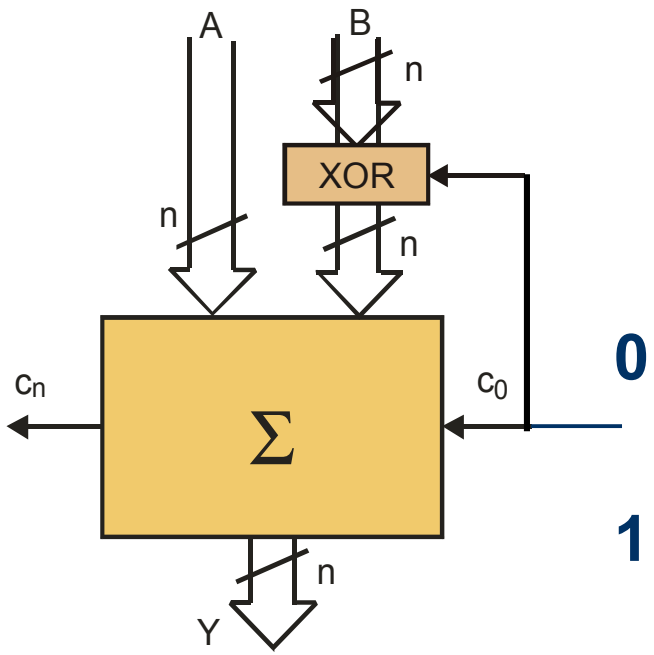
$$\langle 0101 \rangle\big|_{U2} = +5\big|_D; \quad \langle 1011 \rangle\big|_{U2} = -5\big|_D$$

Zakres: $-2^{n-1} \le \boldsymbol{A}_D \le 2^{n-1} - 1$

I T P W

ZPT

Sumator/układ odejmujący



$$Y = A - B = A + (-B|_{U2})$$

$$-B|_{U2} = \overline{B} + 1 = B \oplus 1 + 1$$

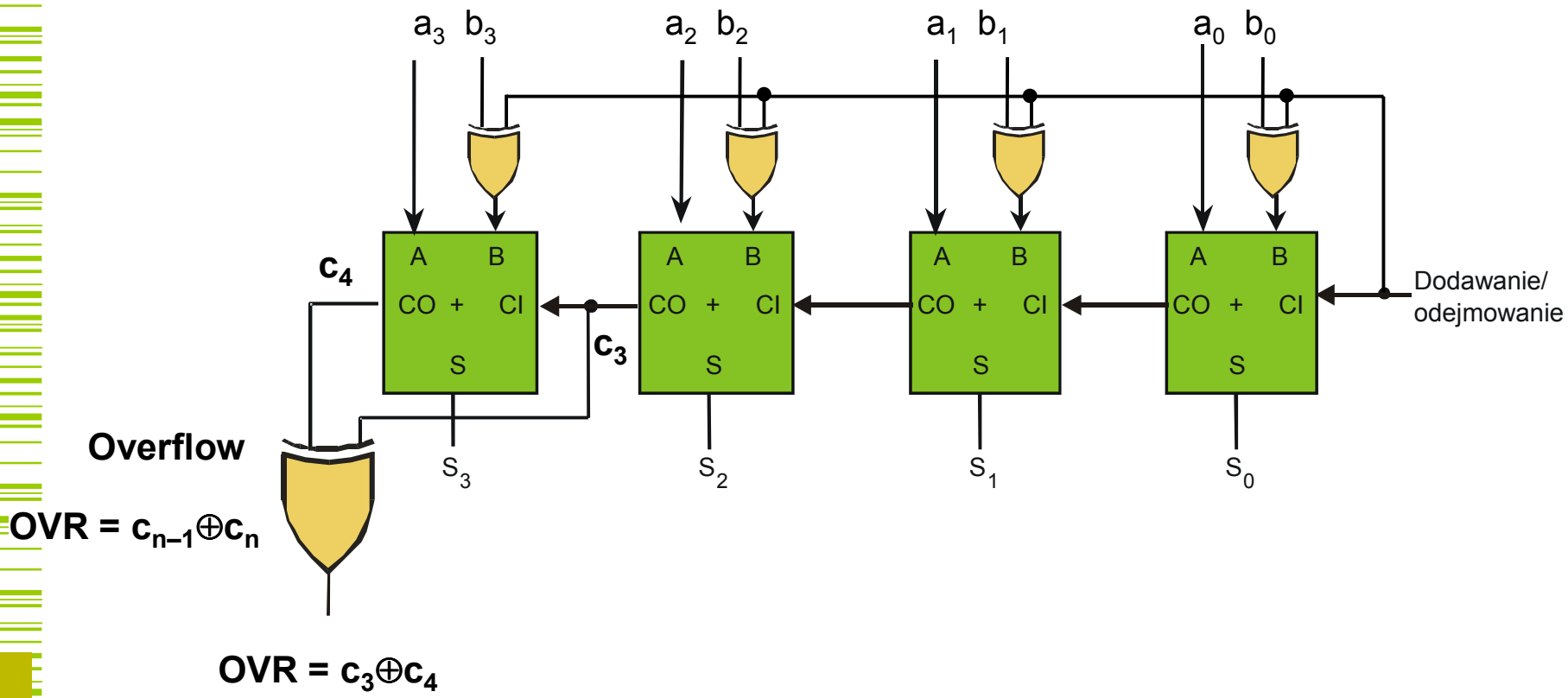
Dla $c_0 = 0$

$$Y = A + B \oplus 0 + 0 = A + B$$

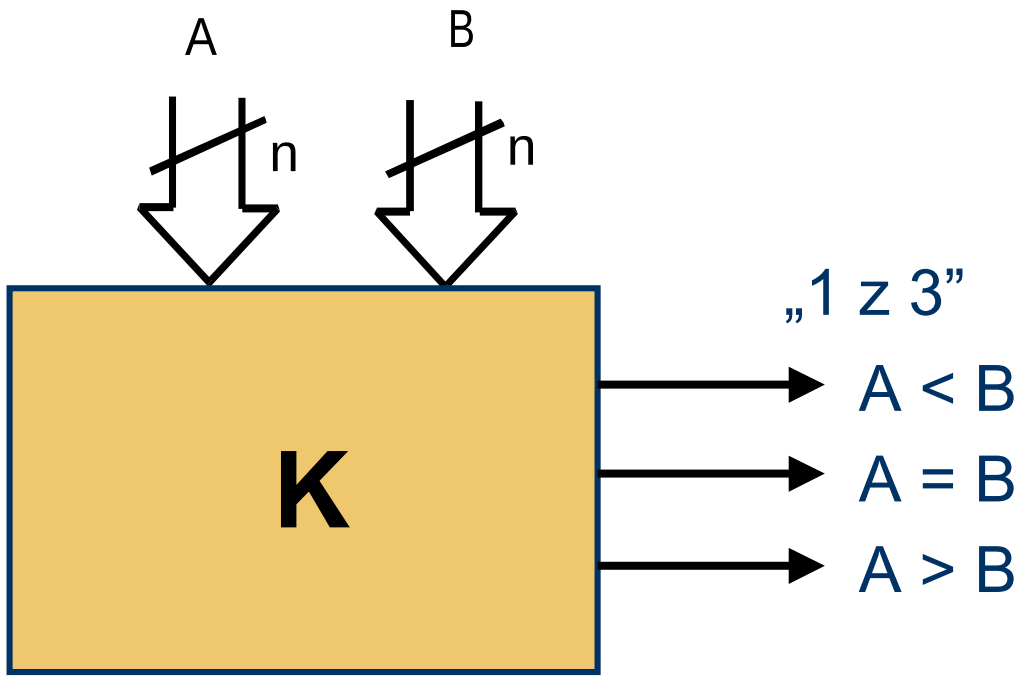
Dla $c_0 = 1$

$$Y = A + \overline{B} + 1 = A - B$$

Sumator/układ odejmujący



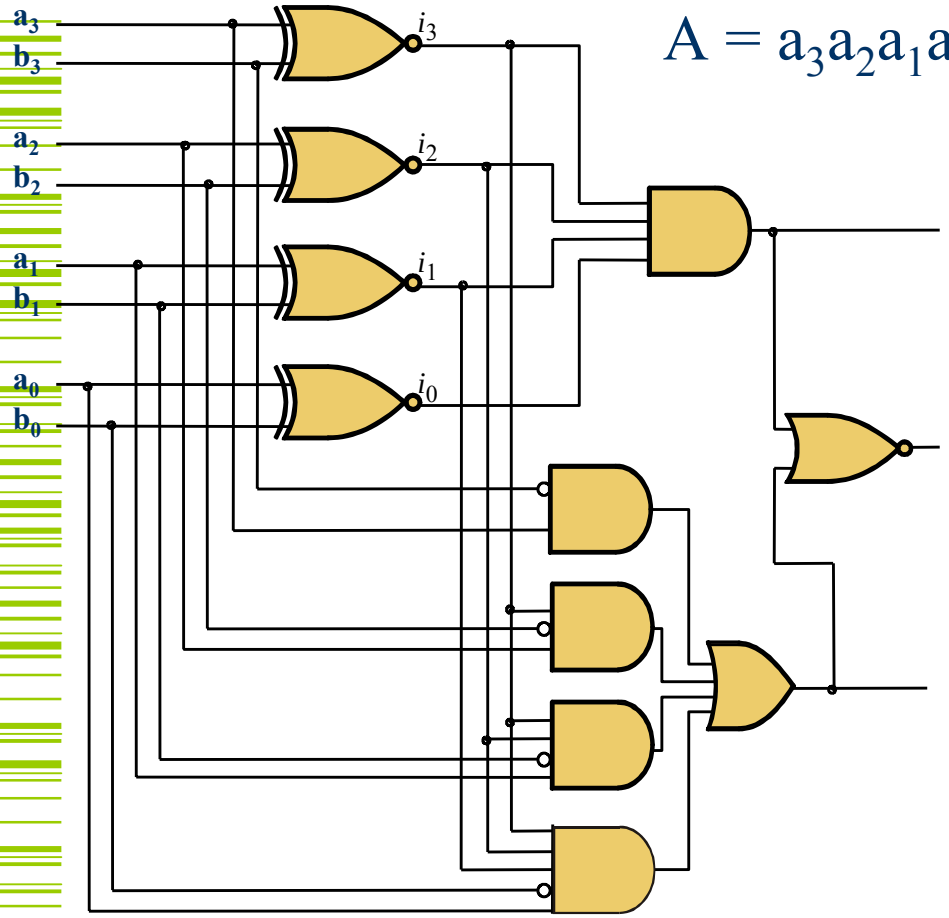
Komparator



I
T
P
W

ZPT

Komparator



$$A = a_3a_2a_1a_0 \quad B = b_3b_2b_1b_0 \quad i_k = \overline{a_k \oplus b_k}$$

$$A\ eq\ B = i_3i_2i_1i_0$$

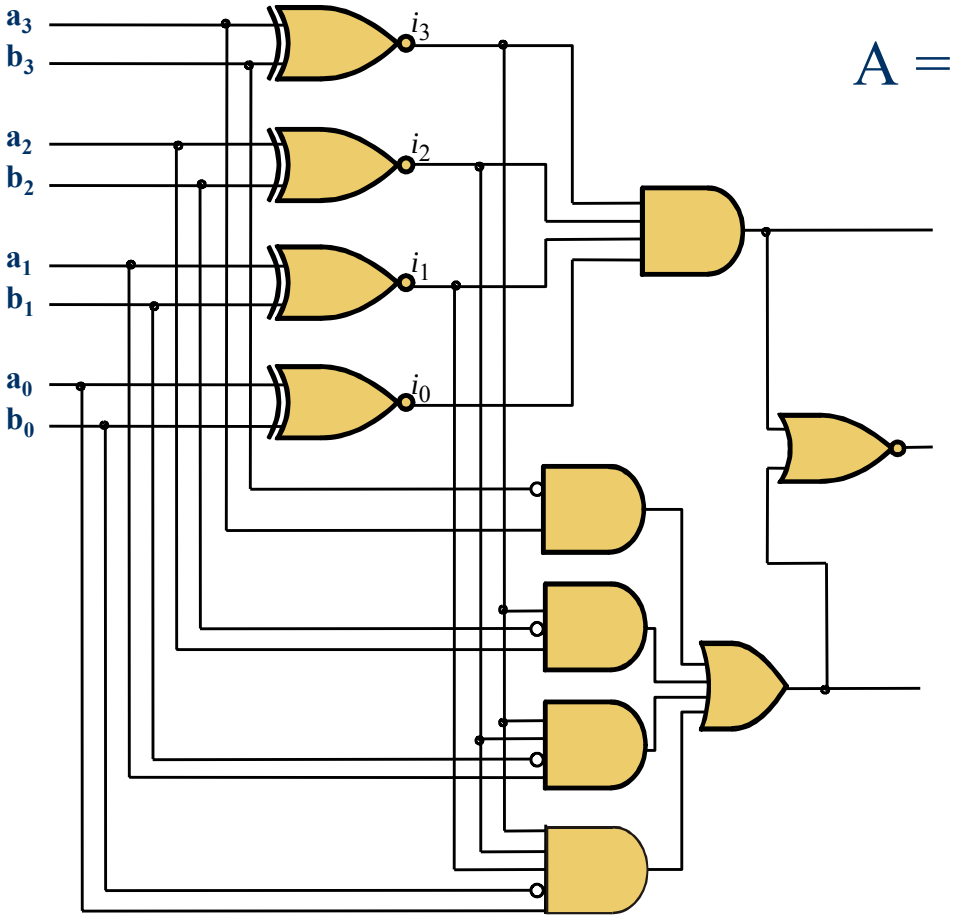
$$A < B = \overline{A\ eq\ B + A\ gt\ B}$$

$$A > B = a_3\bar{b}_3 + i_3a_2\bar{b}_2 + i_3i_2a_1\bar{b}_1 + i_3i_2i_1a_0\bar{b}_0$$

$a_k \neq b_k \Rightarrow$ $A < B$, gdy $a_k = 0$, $b_k = 1$

$A > B$, gdy $a_k = 1$, $b_k = 0$

Komparator



$A = a_3a_2a_1a_0 \quad B = b_3b_2b_1b_0$

A = B

A < B

A > B

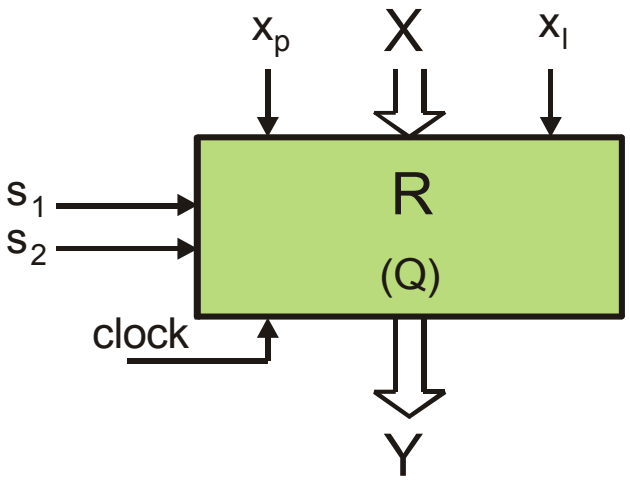
Sekwencyjne bloki funkcjonalne

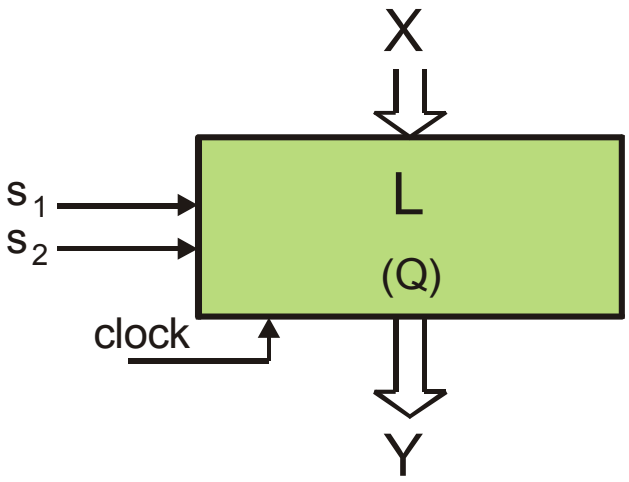
Y := X LOAD
Y := Y HOLD

Rejestry

Liczniki

Y := <0...0> RESET (CLEAR)





Y := SHR(x_p, Y)
Y := SHL(Y, x_l)

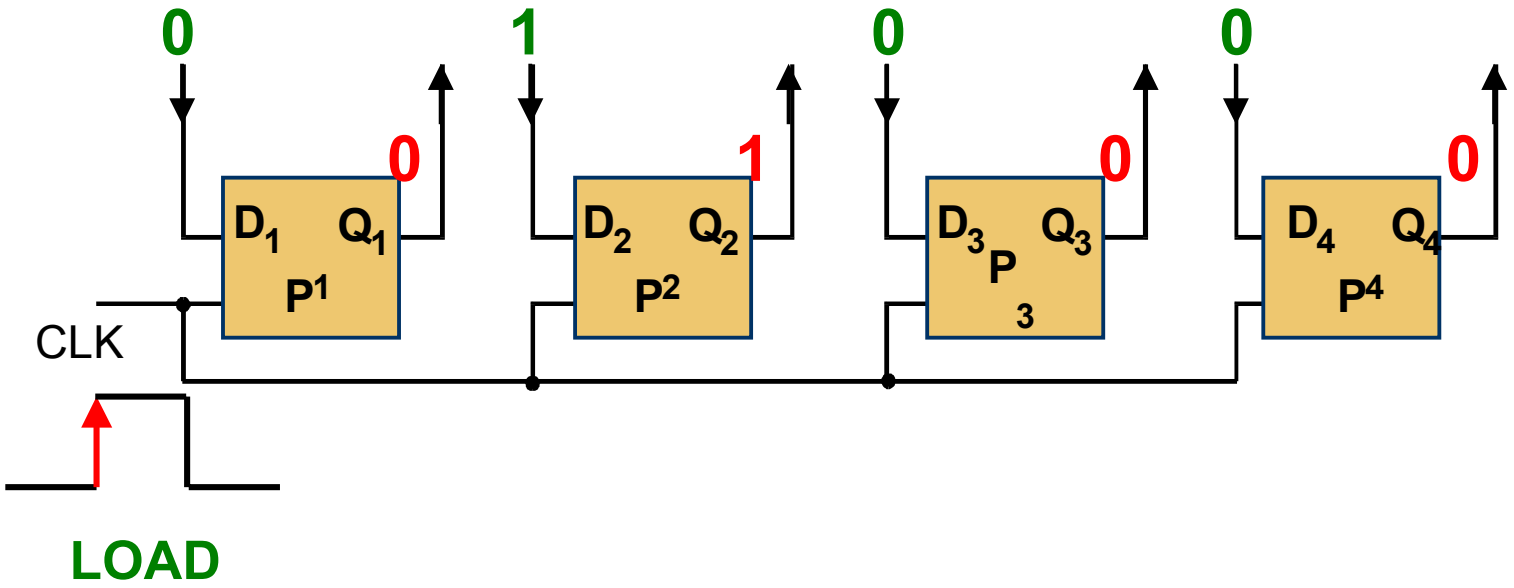
Y := Y + 1 = INC(Y)
Y := Y – 1 = DEC(Y)

I T P W

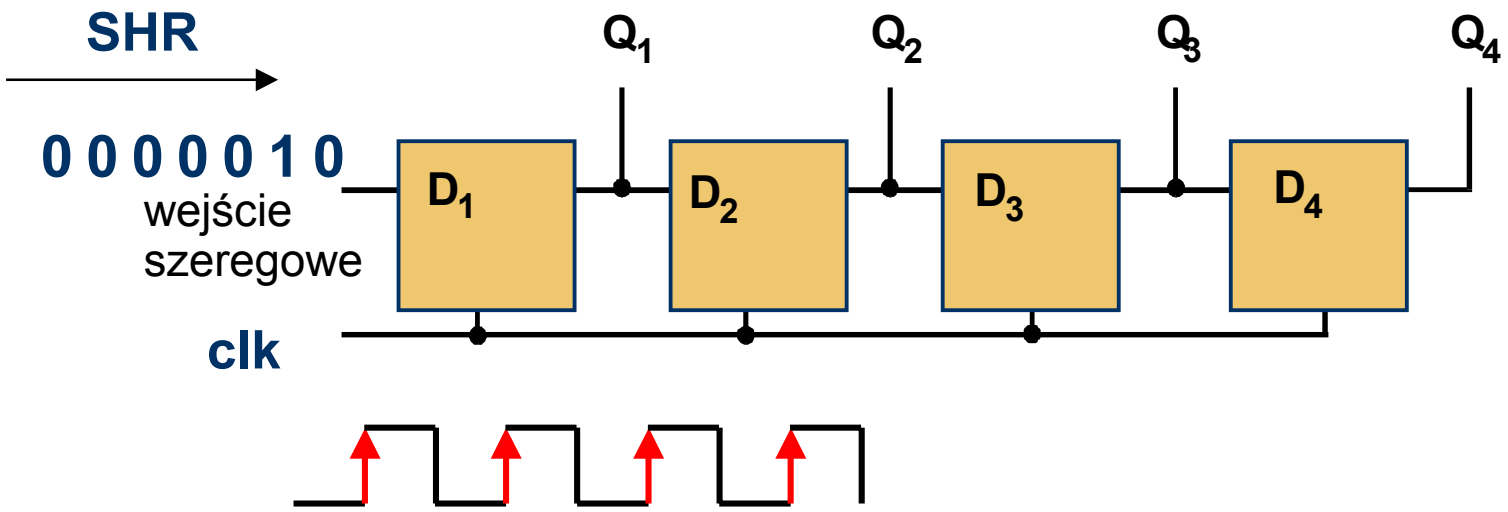
ZPT

Prosty rejestr

Rejestr zbudowany z przerzutników
– ładowanie (load) i pamiętanie



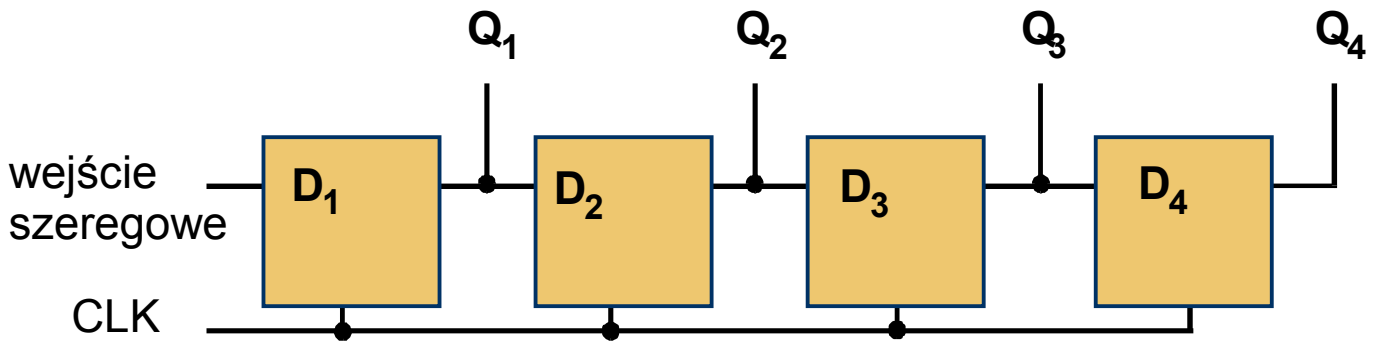
Rejestr przesuwający



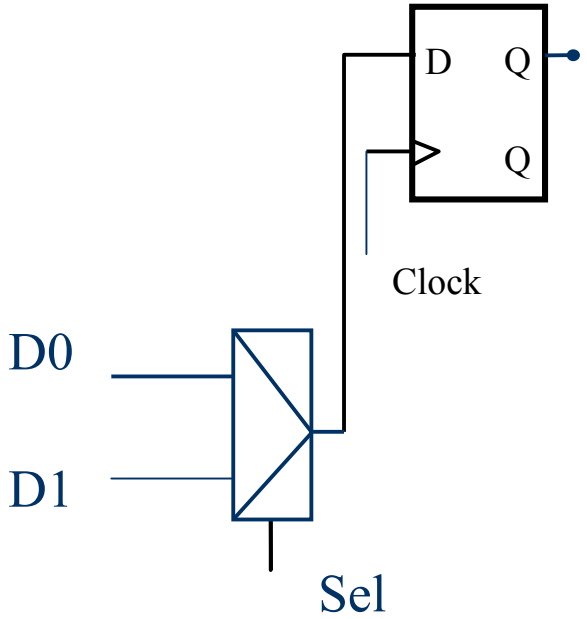
WE	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0
1	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1
0	0	0	0	0

I T P W ZPT

Rejestr przesuwający

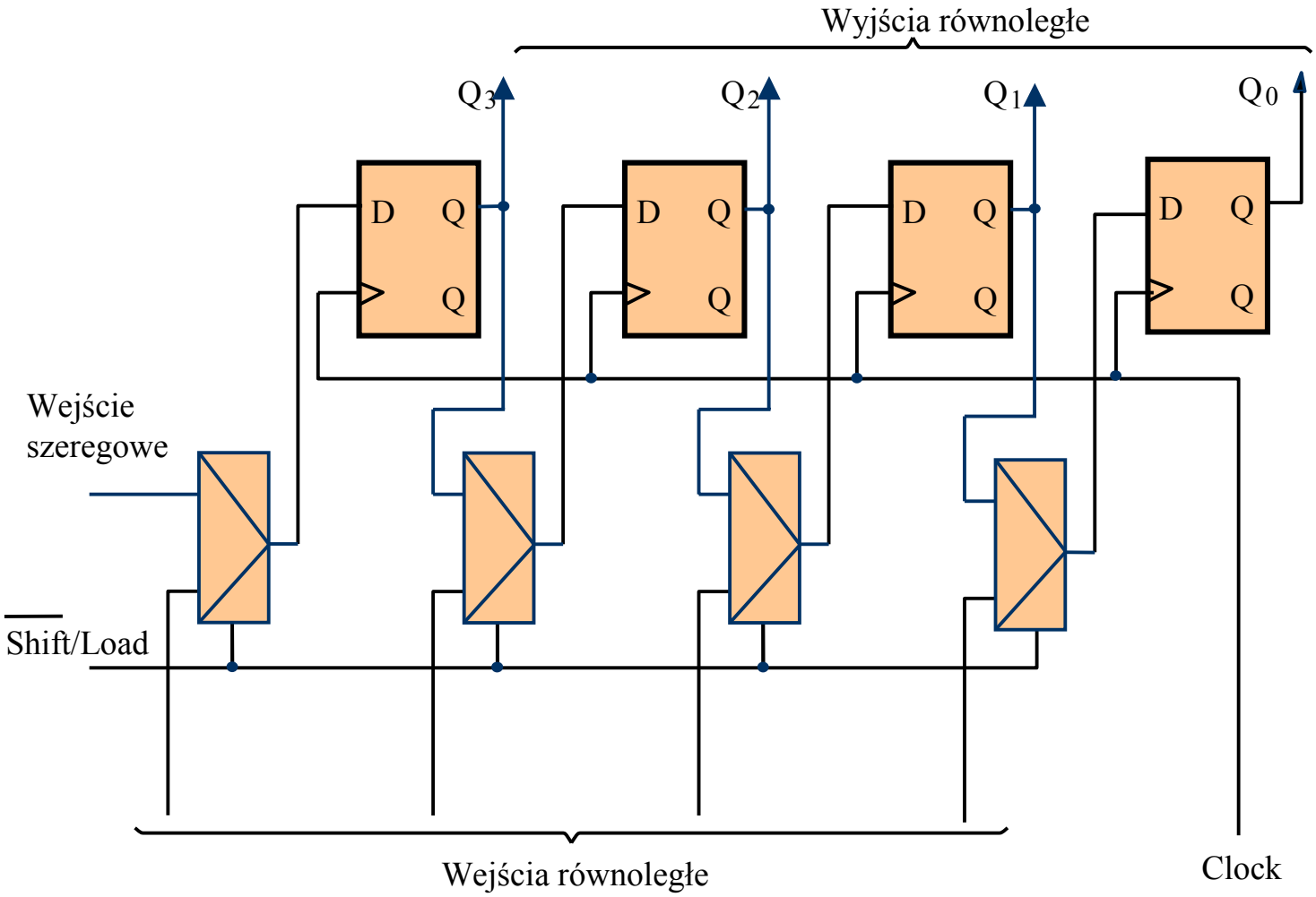


Łatwo można zbudować rejestr, w którym obie funkcje (ładowanie, przesuwanie) wykonywane byłyby w jednym układzie



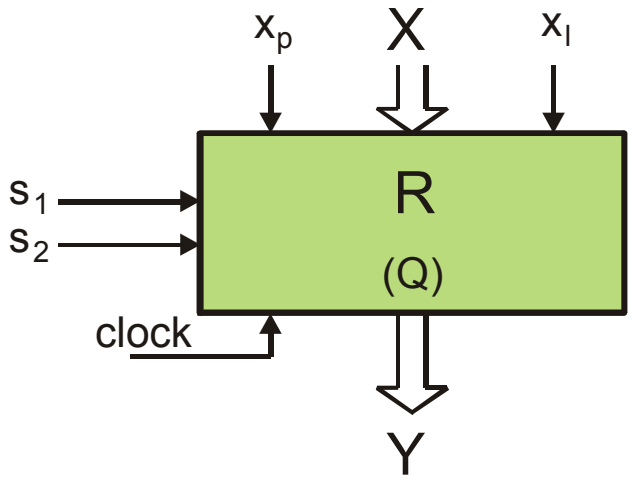
I T P W ZPT

Rejestr przesuwający z wpisem równoległym



Rejestr uniwersalny

Y := X LOAD
Y := Y HOLD



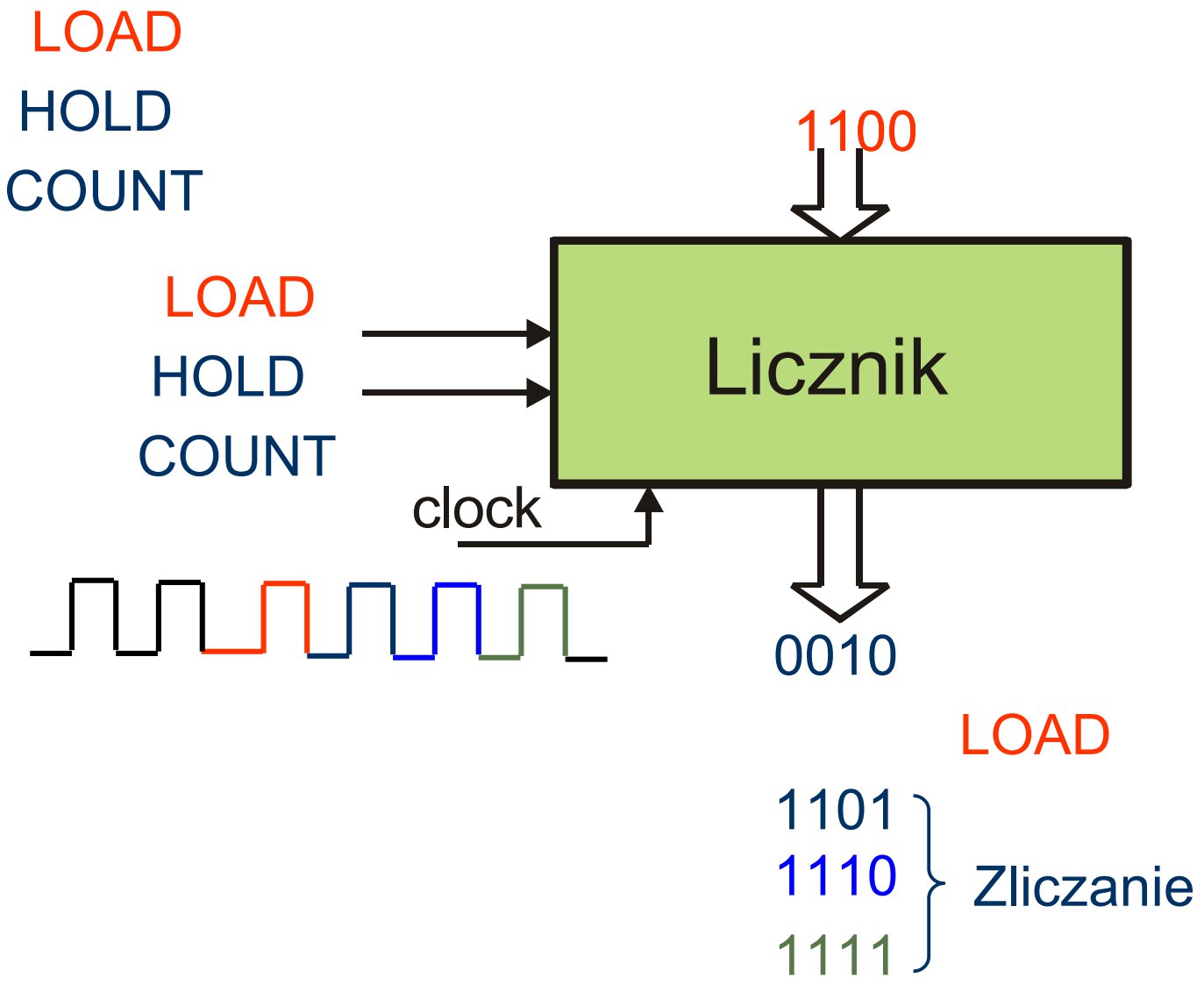
Y := $SHR(x_p, Y)$
Y := $SHL(Y, x_l)$

Y := <0...0> RESET
(CLEAR)

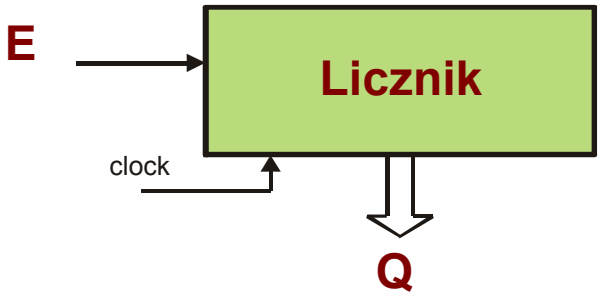
I T P W

ZPT

Mikrooperacje licznika



Przykład licznika z wejściem Enable



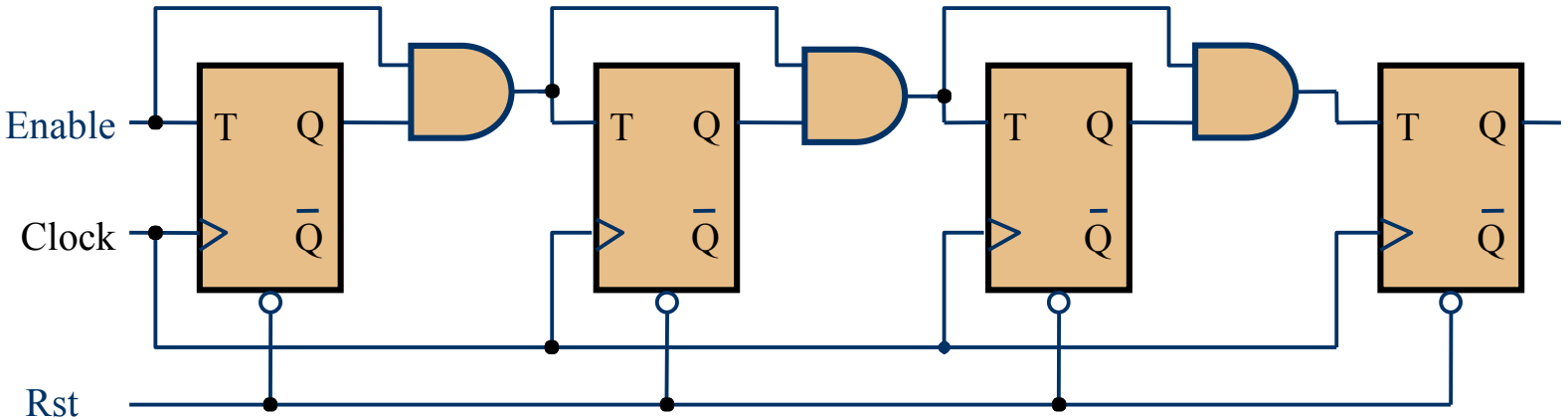
A \ E	0	1
A_0	A_0	A_1
A_1	A_1	A_2
A_2	A_2	A_3
A_3	A_3	A_4
A_4	A	A_5
⋮		
A_{14}	A_{14}	A_{15}
A_{15}	A_{15}	A_0

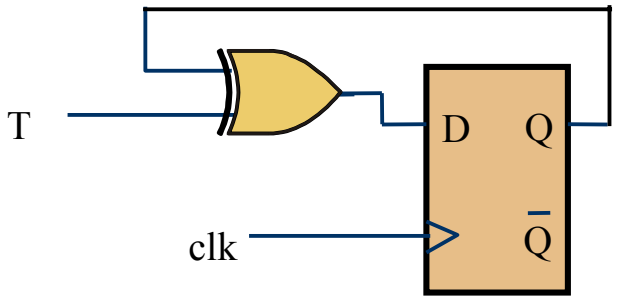
$$T_0 = E$$
$$T_1 = EQ_0$$
$$T_2 = EQ_0Q_1 = T_1Q_1$$
$$T_3 = EQ_0Q_1Q_2 = T_2Q_2$$

I T P W

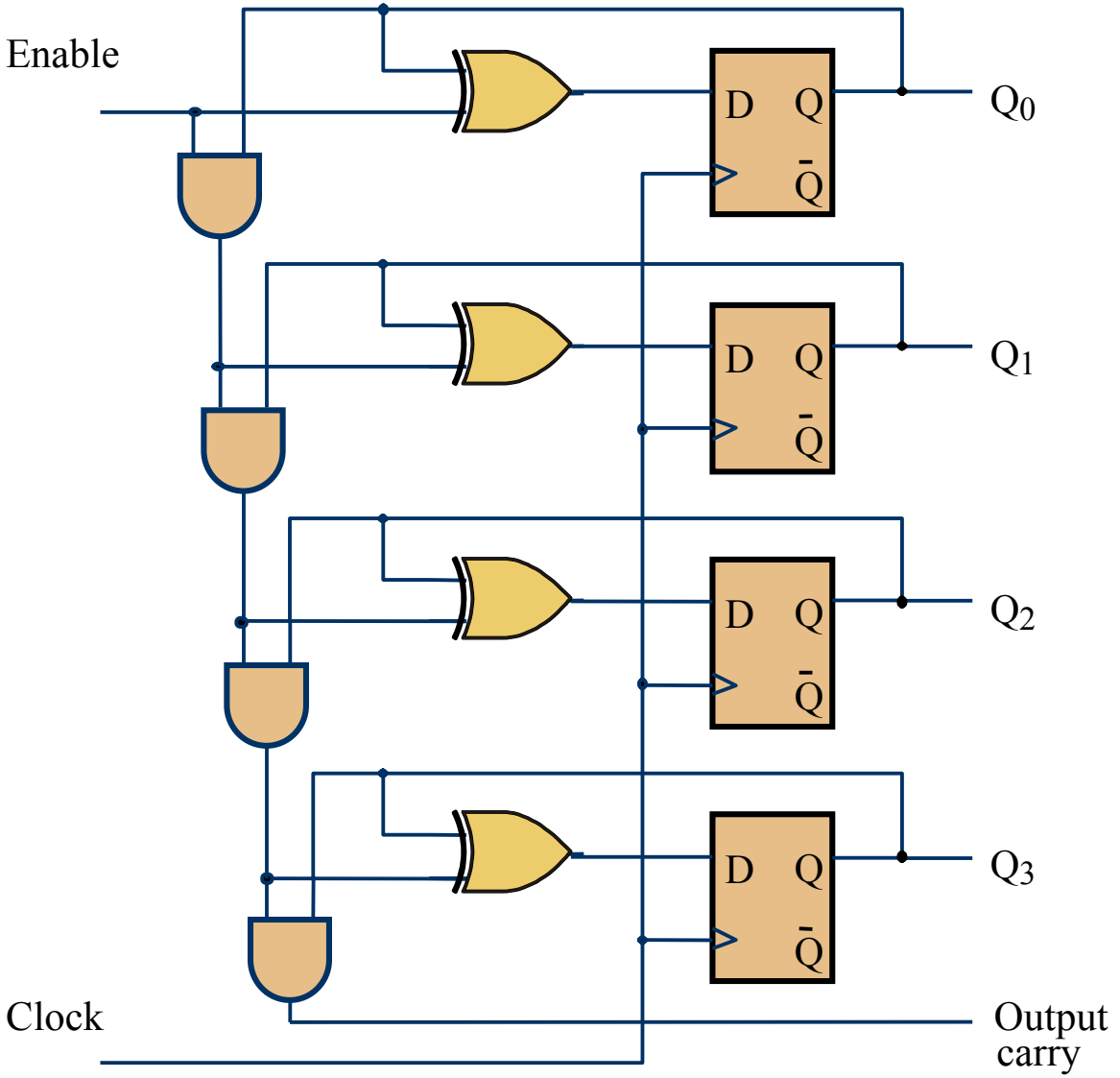
ZPT

Licznik w górę

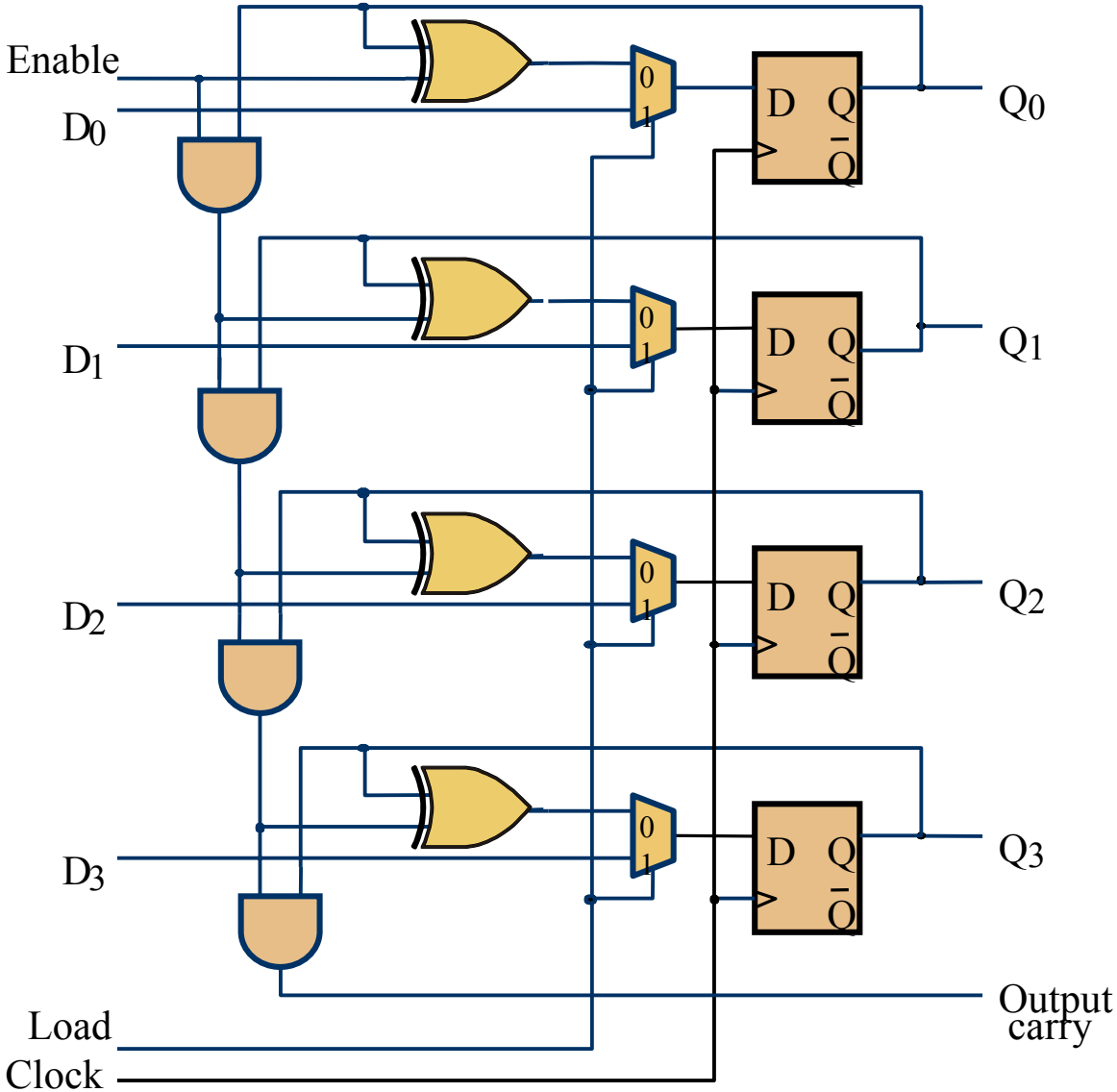




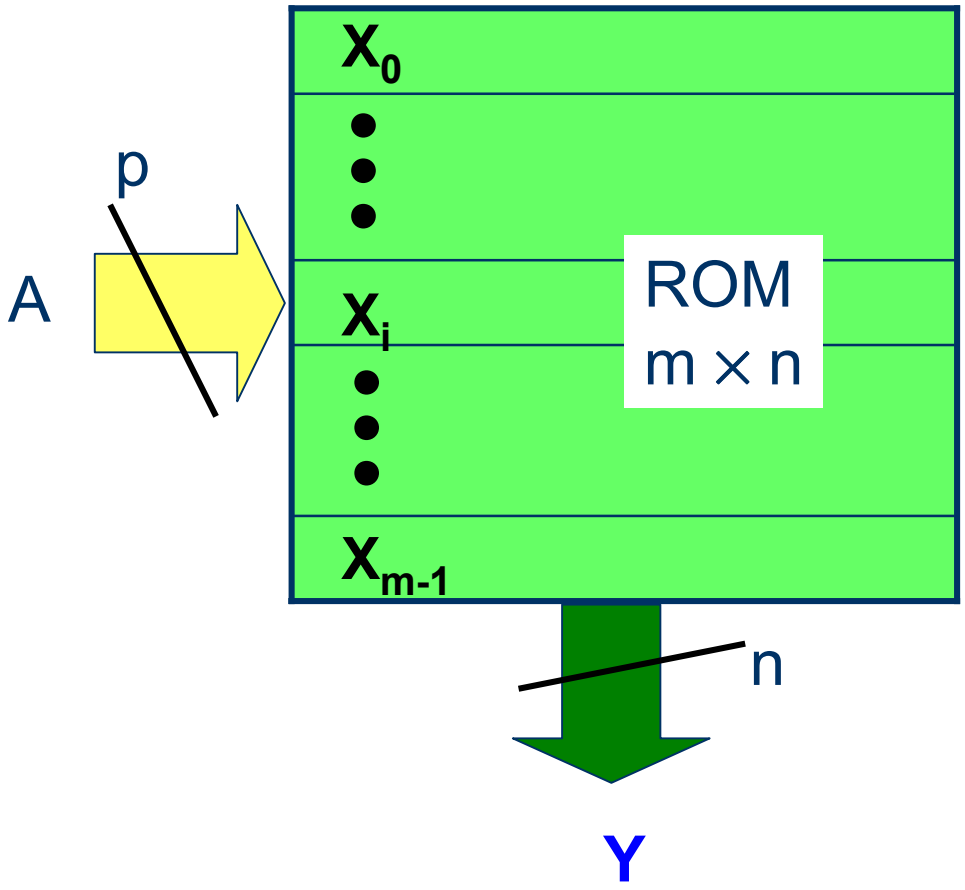
Licznik z przerzutnikami D



Licznik z wpisywaniem równoległym



Pamięci typu ROM



ROM – uniwersalny układ kombinacyjny

Pamięci typu ROM