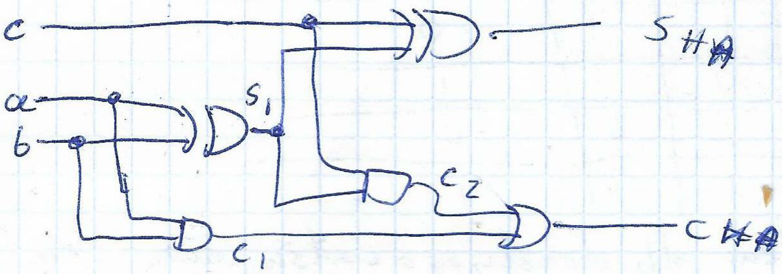
Zad 1. Sumator pełny (Karnaugh)

$s_k = a \oplus b \oplus c$

$c_k = ab + ac + bc$

Sumator pełny (Half Adder)



$S_{HA} = c \oplus s_1 = c \oplus (a \oplus b)$

$C_{HA} = c_1 + c_2 = (ab) + (s_1 c) = ab + c\,(a \oplus b)$

a	b	c	C_{HA}	$a \oplus b$	C_k	S_k	S_{HA}
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1
0	1	0	0	1	0	1	1
0	1	1	1	1	1	0	0
1	0	0	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	1	0	1	0	0
1	1	1	1	0	1	1	1

Zatem dla dowolnego wartościowania a, b, c $S_{HA} = S_k$ oraz $C_k = C_{HA}$, czyli sumatory są sobie równoważne.

Zad 2. Pokażemy, że $c_k = a_k \oplus b_k \oplus s_k$ przy sumowaniu liczb binarnych za pomocą tabeli wartości logicznych.

a_k	b_k	c_k	s_k	$a_k \oplus b_k$	$a_k \oplus b_k \oplus s_k$
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	1	1	0
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	0	1	1
1	1	0	0	0	0
1	1	1	1	0	1

Wartości s_k znamy z wykładu.

Zatem dla dowolnego wartościowania $c_k = a_k \oplus b_k \oplus s_k$

Zad 3.

$g_k = a_k b_k$ - jedna bramka AND

$p_k = a_k + b_k$ - jedna bramka OR

$s_k = a_k \oplus b_k \oplus c_k$ - jedna bramka XOR

Musimy sprawdzić ilość bramek dla każdego c_n, $n \in [1, 8]$. Stosujemy wzór

$$c_n = \sum_{i=-1}^{n-1} g_i \prod_{j=i+1}^{n-1} p_j \quad \cancel{c_0 = g_0}$$

$c_1 = (g_{-1} p_0) + g_0$ - 2 bramki

$c_2 = (g_{-1} p_0 p_1) + (g_0 p_1) + g_1$ - $2 + 1 = 3$ bramki

$c_3 = (g_{-1} p_0 p_1 p_2) + (g_0 p_1 p_2) + (g_1 p_2) + g_2$ ~~5 bramek~~ - 4 bramki

$c_4 = [(g_{-1} p_0 p_1 p_2 p_3) + (g_0 p_1 p_2 p_3) + (g_1 p_2 p_3) + (g_2 p_3) + g_3]$ - $4 + 2 + 1 + 1 = 8$ bramek

$c_5 = \cancel{c_3 p_4 + g_4}\ [(g_{-1} p_0 p_1 p_2 p_3 p_4) + (g_0 p_1 p_2 p_3 p_4) + (g_1 p_2 p_3 p_4) + (g_2 p_3 p_4) + (g_3 p_4) + g_4]$

~~$4 + 2 + 2 + 1 = 9$ bramek~~

$c_6 = [(g_{-1} p_0 p_1 p_2 p_3 p_4 p_5) + (g_0 p_1 p_2 p_3 p_4 p_5) + (g_1 p_2 p_3 p_4 p_5) + (g_2 p_3 p_4 p_5) + (g_3 p_4 p_5) + (g_4 p_5) + g_5]$

~~$4 + 3 + 2 + 2 + 1 = 13$ bramek~~

$c_7 = (g_{-1} p_0 p_1 p_2 p_3 p_4 p_5 p_6) + (g_0 p_1 p_2 p_3 p_4 p_5 p_6) + (g_1 p_2 p_3 p_4 p_5 p_6) + (g_2 p_3 p_4 p_5 p_6) + (g_3 p_4 p_5 p_6 + g_4 p_5 p_6 + g_5 p_6 + g_6)$

$c_8 = (g_{-1} p_0 p_1 p_2 p_3 p_4 p_5 p_6 p_7) + (g_0 p_1 p_2 p_3 p_4 p_5 p_6 p_7) + (g_1 p_2 p_3 p_4 p_5 p_6 p_7) + (g_2 p_3 p_4 p_5 p_6 p_7) + (g_3 p_4 p_5 p_6 p_7) + (g_4 p_5 p_6 p_7) + (g_5 p_6 p_7) + (g_6 p_7) + g_7$

~~$c_7 = 4 + 2 + 3 + 3 + 3 + 1 + 1 = 17$ bramek~~

~~$c_8 = 4 + 2 + 3 + 3 + 3 + 4 + 1 + 1 = 21$~~

~~W sumie $8 + 8 + 8 + 21 + 17 + 13 + 9 + 7 + 4 + 3 + 2$~~

$c_4 = [(g_{-1}p_0p_1p_2)p_3 + (g_0p_1p_2p_3) + (g_1p_2p_3) + (g_2p_3) + g_3]$ — $4+2+1=7$ bramek

$\cancel{(4 + 1 + 1 = 1)}$

$+1=$

$c_5 = [((g_{-1}p_0p_1p_2)p_3p_4) + ((g_0p_1p_2p_3)p_4)] + [(g_1p_2p_3p_4) + (g_2p_3p_4) + g_3p_4 + g_4]$

— $4+2+2+1 = 9$ bramek

$c_6 = [(g_{-1}p_0p_1p_2)p_3p_4p_5] + [(g_0p_1p_2p_3)p_4p_5] + [(g_1p_2p_3p_4)p_5] + [(g_2p_3p_4p_5) + (g_3p_4p_5) + (g_4p_5) + g_5]$ — $4+2+2+2+1 = 11$ bramek

$c_7 = [(g_{-1}p_0p_1p_2)(p_3p_4p_5p_6)] + [(g_0p_1p_2p_3)p_4p_5p_6] + [(g_1p_2p_3p_4)p_5p_6] + [(g_2p_3p_4p_5)p_6] + [g_3p_4p_5p_6 + g_4p_5p_6 + g_5p_6 + g_6]$ — $4+2+2+2+3+1+1 = 15$ bramek

$c_8 = [(g_{-1}p_0p_1p_2)(p_3p_4p_5p_6p_7)] + [(g_0p_1p_2p_3)(p_4p_5p_6p_7)] + [(g_1p_2p_3p_4)p_5p_6p_7] + [(g_2p_3p_4p_5)p_6p_7] + [(g_3p_4p_5p_6)p_7] + [g_4p_5p_6p_7 + g_5p_6p_7 + g_6p_7 + g_7]$

— $4+2+2+2+3+3+1+1 = 18$ bramek

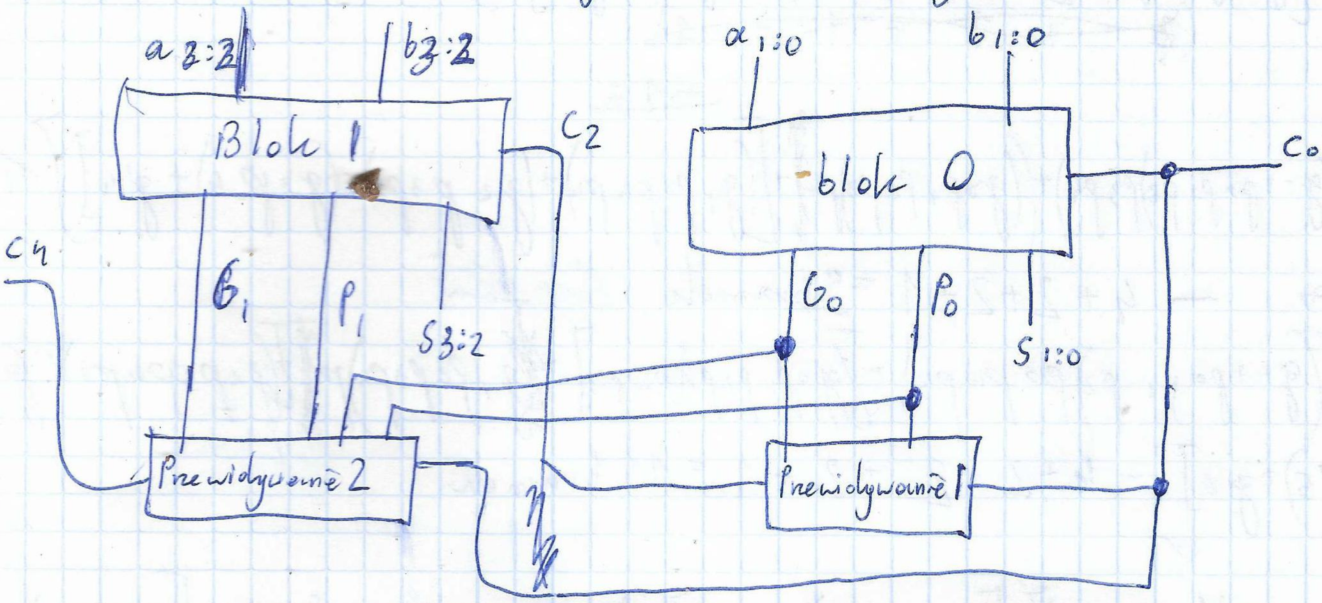
Razem $\cancel{4}$ $8+8+8+18+15+11+9+7+4+3+2 = 93$ bramki

Zad 5. Wygląd ścieżki załączony w pliku zad 5.png. ~~Zauważmy, że FA jest z~~ Jej długość trzeba policzyć analizując kolejne mijane bramki. Przejście przez bramkę AND zwiększa długość o 1~~, w każdym FA wyjście jako s zwiększa długość~~

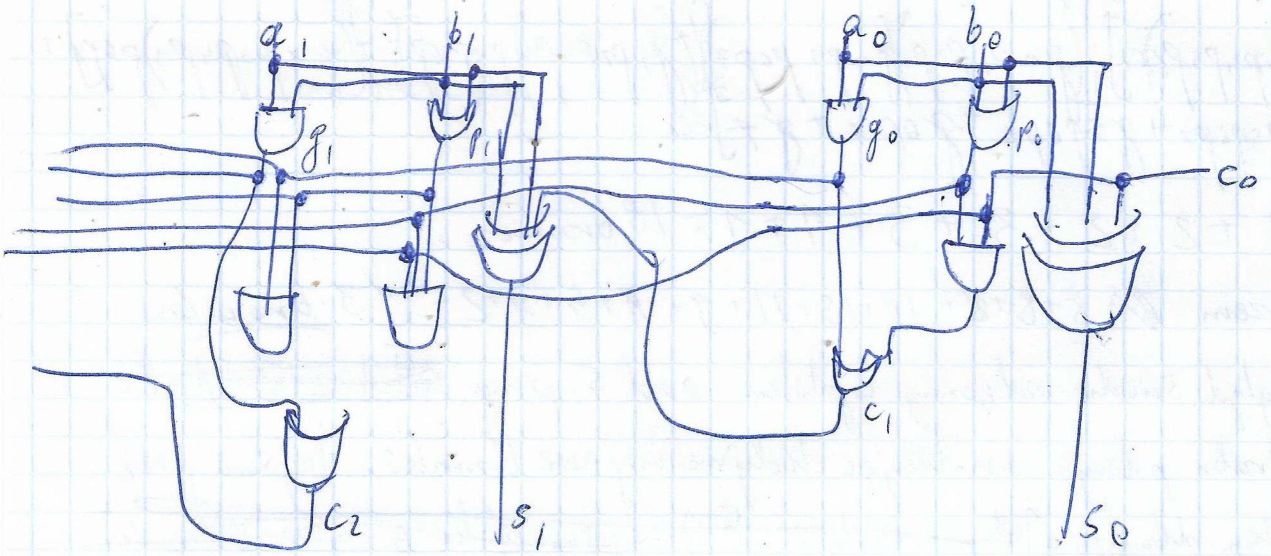
— Zauważmy, że FA jest zbudowany z HA. ~~W takim wyjście z FA jako c zwiększa długość ścieżki o 3, a jako s o 1, gdy wejście,~~ ~~do~~ Przejście przez pierwszy FA ~~zwiększa długość o 3, przez drugi o 2. Następnie wykonuj~~ Przejście przez FA zwiększa ścieżkę na kilka sposobów:

- o 1, gdy wchodzimy jako c i wychodzimy jako s
- o 2, gdy c -> c_0
- o 2, gdy a -> s.
- o 3, gdy a -> c_0

Zad 4. Sumator hierarchiczny dla liczb czterobitowych

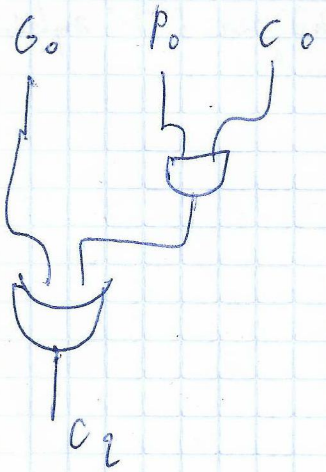


Bloki 0 i 1

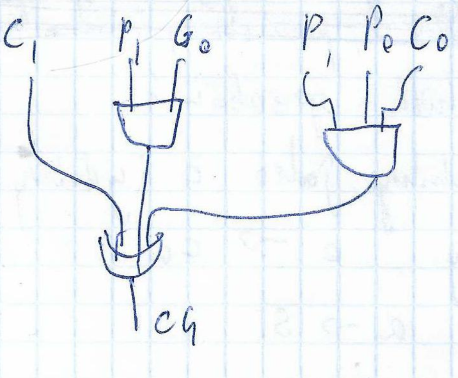


Blok przewidywania

1



2



6. Sumator szeregowy jest prostszym w budowie układem. Dla mniejszych układów logicznych ~~pe~~ zysk czasowy płynący z zastosowania sumatora hierarchicznego jest nieznaczący, gdy weźmiemy pod uwagę stopień wzrostu skomplikowania obwodu. Optymalnie może być zastosować wtedy sumator szeregowy.

7. Wartości g-d będziemy przechowywali na 4 bitach. Sprawdźmy jakie mają części w zależności od d. d podajemy na czterech bitach w, x, y, z

$g\text{-}d = [4] = a[4]$

$a_0 = g\text{-}d[0] = \bar{w}\bar{x}\bar{y}$

wx \ yz	00	01	11	10
00	1	1	0	0
01	0	0	0	0
11	–	–	–	–
10	0	0	–	–

$a_1 = g\text{-}d[1] = x\bar{y} + \bar{x}y$

wx \ yz	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	–	–	–	–
10	0	0	–	–

$a_2 = g\text{-}d[2] = y$

wx \ yz	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	–	–	–	–
10	0	0	–	–

$a_3 = g\text{-}d[3] = \bar{z}$

wx \ yz	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	–	–	–	–
10	1	0	–	–

a_0	a_1	a_2	a_3	w	x	y	z
1	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1
0	1	1	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	0	0	0	1	0	1
0	0	1	1	0	1	1	0
0	0	1	0	0	1	1	1
0	0	0	1	1	0	0	0
0	0	0	0	1	0	0	1

Układ w załączniku zad 7.sv

Zad 9.

– 24-bitowe l.s.p. bez znaku z 12 bitów cz.ul. leżą w przedziale

$[\underbrace{00\ldots0}_{12\text{ zer}},\underbrace{00\ldots0}_{12\text{ zer}};\ \underbrace{111\ldots111}_{12\text{ jedynek}},\underbrace{111\ldots1}_{12\text{ jedynek}}]$

– 24-bitowe l.s.p. ze znakiem w kodzie znak-moduł z 12-bitów cz.ul.

$[\underset{\text{znak ujemny}}{1}\ \underbrace{111\ldots1}_{11\text{ jedynek}},\underbrace{111\ldots1}_{12\text{ jedynek}};\ \underset{\text{znak dodatni}}{0}\ \underbrace{1\ldots11}_{11\text{ jedynek}},\underbrace{111\ldots11}_{12\text{ jedynek}}]$

– 24-bitowe l.s.p. ze znakiem w U2 z 12-bit. cz.ul.

$[\underset{\text{znak ujemny}}{1}\ \underbrace{000\ldots00}_{11\text{ zer}},\underbrace{00\ldots00}_{12\text{ zer}};\ \underset{\text{znak dodatni}}{0}\ \underbrace{111\ldots111}_{11\text{ jedynek}},\underbrace{111\ldots111}_{12\text{ jedynek}}]$