Podstawy Elektrotechniki i Elektroniki
część 5

dr hab. inż. Stanisław Hałgas, prof. PŁ

Wprowadzenie

Wprowadzenie

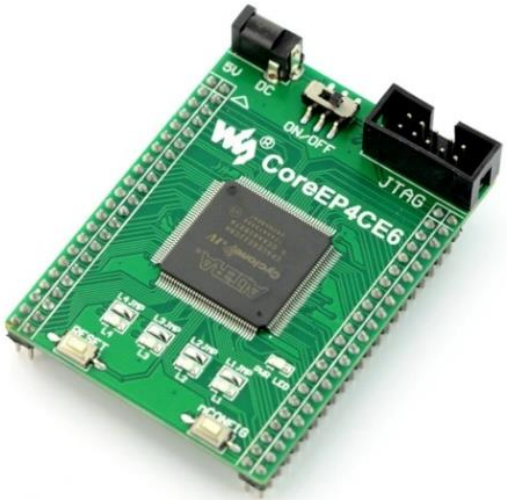
- **Układy cyfrowe – układy elektroniczne, w których sygnały napięciowe przyjmują tylko określoną liczbę poziomów z określonymi wartościami liczbowymi.**
- Jeżeli liczba poziomów napięć jest równa dwa to poziomom przypisywane są cyfry 0 i 1, wówczas układy cyfrowe realizują operacje zgodnie z **algebrą Boole'a** i nazywane są zwykle **układami logicznymi**.

Wprowadzenie

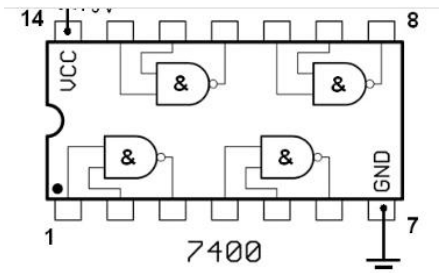
Wprowadzenie

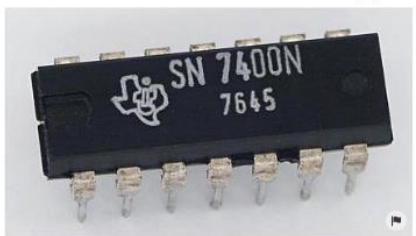
- **Obecnie układy cyfrowe budowane są w oparciu o bramki logiczne realizujące elementarne operacje: iloczyn logiczny (AND), sumę logiczną (OR), negację NOT, itp.**
- Stopień skomplikowania współczesnych układów cyfrowych sprawia, że wykonuje się je w postaci układów scalonych.
- **Ze względu na sposób przetwarzania informacji** rozróżnia się dwie główne klasy układów logicznych:
  1. **układy kombinacyjne („bez pamięci”)** – sygnały wyjściowe zależą tylko od aktualnego stanu wejść
  2. **układy sekwencyjne („z pamięcią”)** – stan wyjść zależy nie tylko od aktualnego stanu wejść, ale również od stanów poprzednich.

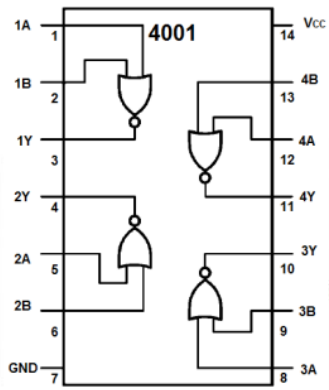
Wprowadzenie



Altera Cyclone IV EP4CE6 - płytka rozwojowa FPGA - Waveshare 6483





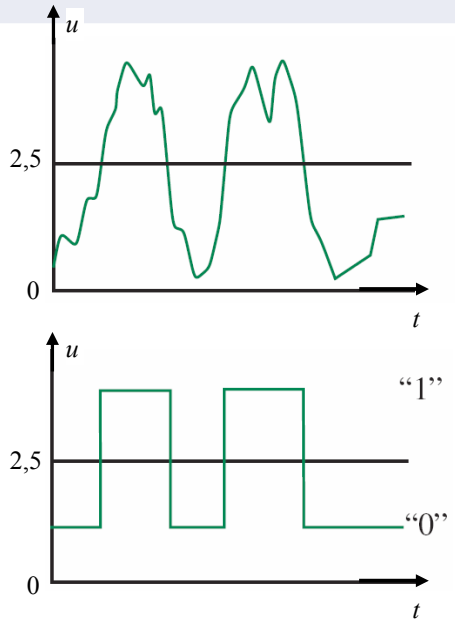


Wprowadzenie

Wprowadzenie

- **Dyskretyzacja wartości (kwantyzacja)** – podstawa cyfrowej abstrakcji (de facto zawsze operujemy na pewnych wartościach napięć)[1] – **przyporządkowanie wartościom sygnału z pewnego przedziału jednej wartości (symbolu)**.

[1] W procesie analogowo-cyfrowego przetwarzania sygnału, czyli zamiany analogowego na cyfrowy, kwantyzacja jest najczęściej etapem następującym po próbkowaniu.



Rys. 1: Dyskretyzacja wartości – przykład

Wprowadzenie

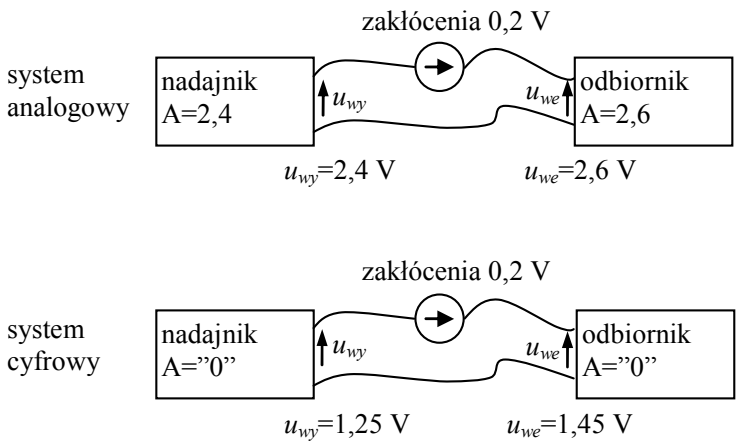
Wprowadzenie

- **Kwantyzacja (np. – zero logiczne ("0") – wartość napięcia z zakresu 0 V – 2,5 V; jedynka logiczna ("1") - 2,5 V – 5 V).**
- Przykładowo, w celu przesłania logicznego zera, wystarczy na ścieżkę sygnałową podać napięcie równe 1,25 V, a w celu transmisji logicznej jedynki napięcie o wartości 3,75 V.
- Sygnał z rys. 1 sekwencja 01010.
- **Dyskretyzacja zmniejsza dokładność reprodukcji sygnału w stosunku do transmisji analogowej. Ma jednak nad transmisją analogową ogromną przewagę przy przesyle sygnału w obecności zakłóceń.**

Wprowadzenie

Wprowadzenie

Hipotetyczna sytuacja (rys. 2) – nadawca chce przekazać wartość A odbiorcy.



Rys. 2: Porównanie transmisji analogowej i cyfrowej w obecności zakłóceń

- **Transmisja sygnału analogowego** – wartość A wynosi 2,4 V. Zakłócenia transmisji–> w odbiorniku wartość 2,6 V – **odbiornik błędnie interpretuje przesyłaną wartość** jako 2,6 V zamiast 2,4 V.
- **Transmisja sygnału cyfrowego** – wartość A jest logicznym zerem. Nadawca przekazuje tę wartość, umieszczając na ścieżce (linii) napięcie o wartości 1,25 V –> w odbiorniku napięcie 1,45 V o wartości poniżej progu 2,5 V, **odbiornik prawidłowo interpretuje przesłany sygnał jako logiczne zero**.

Wprowadzenie

Wprowadzenie

- **Dwa poziomy sygnału są wystarczające do wielu zastosowań, np. obliczenia logiczne obejmują sygnały o jednej z dwóch wartości: TRUE lub FALSE.**
- Dwupoziomowy sygnałów jest przesyłany przez pojedynczy przewód (pojedynczą linię).
- Istnieją aplikacje wymagające większej precyzji, np. **aplikacja przetwarzania sygnału dźwięku może wymagać 256 lub większej (65536=2^{16}) liczby poziomów dyskretyzacji.**
- Jedną z metod osiągnięcia większej precyzji jest **użycie kodowania** do tworzenia wielocyfrowych liczb.
- **Jeżeli każda cyfra przyjmuje jedną z dwóch wartości, cyfra taka nazywana jest cyfrą binarną lub bitem (z ang. skrót od *binary digit*).**
- *Sygnały wielobitowe są zwykle przesyłane wieloma przewodami (każda linia na jeden bit – **transmisja równoległa**) lub na drodze **multipleksowania**, czyli kolejnego przesyłania bitów jednym przewodem.*

Wprowadzenie

Wprowadzenie

- **Dwupoziomowa reprezentacja – reprezentacja binarna**.
- Praktycznie wszystkie układy cyfrowe używają reprezentacji binarnej ze względu na łatwość projektowania i produkcji.
- Poziomy w reprezentacji binarnej nazywają się różnie: **(a) TRUE lub FALSE, (b) ON lub OFF, (c) 1 lub 0, (d) HIGH lub LOW.**
- **Logika dodatnia – pozytywna** – na przykład 0 V w celu reprezentacji FALSE oraz 5 V w celu reprezentacji TRUE.
- **Logika ujemna – negatywna** przyporządkowanie przeciwne (TRUE – 0 V i FALSE – 5 V).
- **W dalszej części stosowana jest logika dodatnia.**

Wprowadzenie

Tabela 1: Reprezentacja sygnałów binarnych, x – wartość pewnej zmiennej

TRUE	FALSE
0 V	5 V
5 V	0 V
2 V	0 V
0 V	1 V
ON	OFF
$0\,\text{V} < x < 2{,}5\,\text{V}$	$2{,}5\,\text{V} < x < 5\,\text{V}$
$0\,\text{V} < x < 1\,\text{V}$	$4\,\text{V} < x < 5\,\text{V}$
$0\,\mu\text{A}$	$2\,\mu\text{A}$

Standardy poziomów logicznych

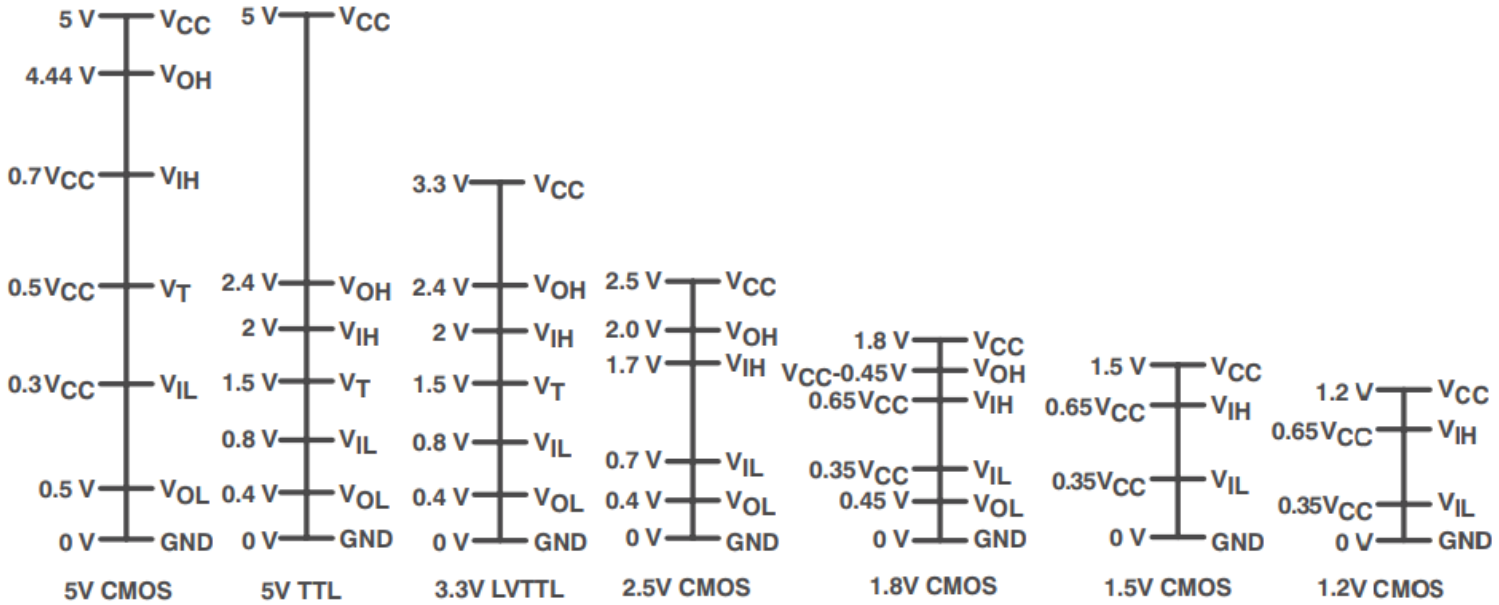
Standardy poziomów logicznych

- Istnieje **wiele sposobów reprezentowania wartości binarnych**, które różnią się nie tylko typem sygnału (prąd lub napięcie), ale także wartością sygnału (na przykład 5 V lub 4 V w celu reprezentowania logicznej jedynki).
- Aby układy cyfrowe zbudowane przez różnych producentów współpracowały ze sobą muszą być one zgodne z pewnym **standardem**.
- Reprezentacja powinna zapewniać **prawidłowe działanie w obecności pewnego poziomu zakłóceń.**

Standardy poziomów logicznych

Standardy poziomów logicznych

Wartości gwarantowane napięć odpowiadających poziomom logicznym (ang. static discipline, SD) należą do najważniejszych specyfikacji układów cyfrowych (rys. 3 [SLVA:2017]).

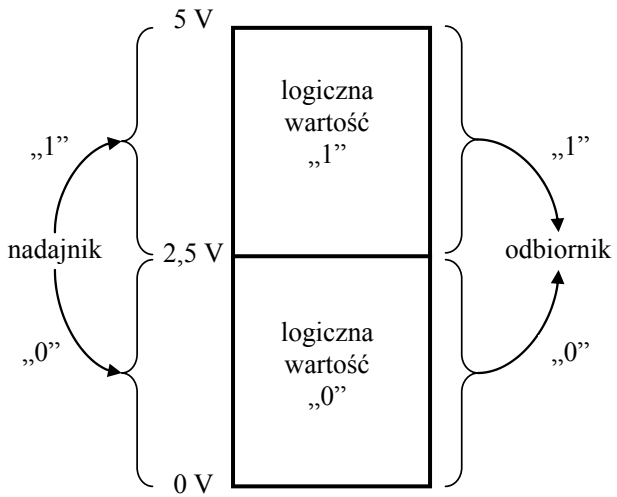


Rys. 3: Przykładowe wartości gwarantowane (standardy) napięć odpowiadających poziomom logicznym [SLVA:2017]

Standardy poziomów logicznych

Standardy poziomów logicznych

- **Specyfikacje standardowe – układy muszą prawidłowo interpretować prawidłowe (dopuszczalne) dane wejściowe (ang. valid) zgodnie z obraną reprezentacją i generować sygnały wyjściowe, które są prawidłowymi sygnałami logicznymi.**
- Najprostsza reprezentacja – podział zakresu napięć na dwa przedziały (rys. 4).
- **Problem interpretacji przez odbiornik napięcia o wartości 2,5 V na linii** – w celu eliminacji – **dodatkowy obszar (tzw. zabroniony)**, który dzieli dwa dotychczas utworzone regiony.

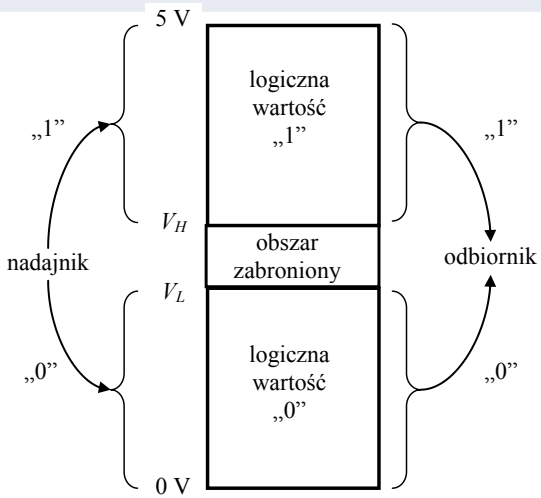


Rys. 4: Najprostsza reprezentacja standardów napięć

Standardy poziomów logicznych

Standardy poziomów logicznych

- Zachowanie odbiornika jako **nieokreślone**, gdy na linii występuje napięcie w zabronionym obszarze.
- Odbiornik – każda wartość 0 V – 2 V, umieszczona przez nadawcę, jest poprawna dla logicznego zera, a dowolna wartość 3 V – 5 V poprawna dla logicznej jedynki. Napięcia z zakresu 2 V – 3 V są nieważne (invalid).
- **próg niskiego napięcia** V_L – największa wartość napięcia dla logicznego 0
- **próg wysokiego napięcia** V_H – najmniejsza wartość napięcia dla logicznej 1.



Rys. 5: Reprezentacja standardów napięć z obszarem zabronionym

Standardy poziomów logicznych

Standardy poziomów logicznych

- **Nadajnik** może wysyłać dowolną wartość z zakresu V_H – 5 V dla ”1” oraz dowolną wartość 0 V – V_L dla ”0”. **Układ nadawczy nie może nigdy umieszczać na linii wartości w zabronionym obszarze.**
- **Odbiornik** musi interpretować dowolną wartość z zakresu V_H – 5 V jako ”1” oraz dowolną wartość 0 V – V_L jako ”0”. Zachowanie odbiornika może być nieokreślone, jeśli napięcie na linii jest z przedziału V_L – V_H, ponieważ wartości te znajdują się w zabronionym regionie.
- **Reprezentacja ta nie jest odporna na zakłócenia.**

Standardy poziomów logicznych

Standardy poziomów logicznych - odporność na zakłócenia

- **W celu osiągnięcia odporności — wprowadza się pewne ograniczenia w zakresie wartości wysyłanych.**
- Założenie: odbiornik może interpretować napięcia poniżej 2 V jako ”0”.
- Założenie: nadajnik ma ograniczone możliwości wysyłania napięć niższych niż 0,5 V jako ”0”.
- Wysyłanie napięcia o wartości 0,5 V jako ”0” sprawi, że dopiero zakłócenia przekraczające wartość 1,5 V (dodatnią) spowodują, że sygnał napięciowy wysyłany przez nadajnik będzie w zabronionym obszarze (> 2 V).

Standardy poziomów logicznych

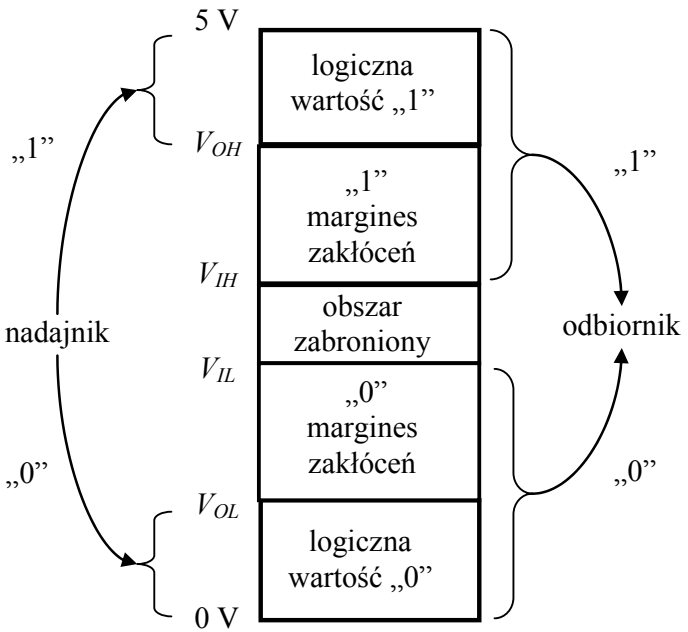
Standardy poziomów logicznych - odporność na zakłócenia

- **Logiczna jedynka**
- Założenie: odbiornik może interpretować napięcia powyżej 3 V jako ”1”.
- Założenie: nadajnik ma ograniczone możliwości wysyłania napięć powyżej 4,5 V jako ”1”.
- Tak więc (przy założeniu, że nadajnik wysyła 4,5 V) zakłócenia o wartości co najmniej 1,5 V odejmujące się od sygnału na linii mogą sprawić, że sygnał napięciowy na linii będzie w zabronionym obszarze.
- Taki wybór poziomów napięć zapewnia **odporność na zakłócenia wynoszącą 1,5 V** dla logicznej jedynki.

Standardy poziomów logicznych

Standardy poziomów logicznych - odporność na zakłócenia

- **Węższe przedziały wartości napięć w nadajniku w porównaniu z przedziałami dla odbiornika – asymetria progów napięcia wejściowego i wyjściowego.**
- Rys. 6 pokazuje zależności pomiędzy prawidłowymi poziomami napięć i sygnałami logicznymi, które są **powszechnie stosowane w układach cyfrowych**.



Rys. 6: Reprezentacja standardów napięć uwzględniająca odporność na zakłócenia

- Nadajnik ”0” – musi umieścić na linii wartość napięcia mniejszą niż V_{OL}.
- Odbiornik musi interpretować u<V_{IL} jako ”0”.
- V_{IL} **musi być większe od** V_{OL} **– margines zakłóceń**.
- Nadajnik ”1” – musi wprowadzić na linię napięcie wyjściowe większe niż V_{OH}.
- Odbiornik musi interpretować u>V_{IH} jako ”1”.
- V_{OH} **musi być większe od** V_{IH} **– margines zakłóceń**.

Standardy poziomów logicznych

Standardy poziomów logicznych - odporność na zakłócenia

- **Margines zakłóceń definiujemy zarówno dla transmisji ”1”, jak ”0”.**
- Dla logicznego zera – $NM_0 = |V_{OL} - V_{IL}|$
- Dla logicznej jedynki – $NM_1 = |V_{OH} - V_{IH}|$
- **Zakres napięć pomiędzy V_{IL}, a V_{IH} jest obszarem zabronionym. Gdy NM_1 oraz NM_0 są jednakowe – marginesy zakłóceń są symetryczne.**
- Przy projektowaniu ważna jest **maksymalizacja marginesów zakłóceń w celu osiągnięcia maksymalnej odporności**.

  Maksymalizacja NM_0 – maksymalizacja V_{IL} i minimalizacja V_{OL}.

  Maksymalizacja NM_1 – maksymalizacja V_{OH} i minimalizacja V_{IH}.

Podstawy algebry Boole'a

Algebra Boole'a – wprowadzenie

- **Binarna reprezentacja ma naturalny związek z logiką**, a zatem obwody cyfrowe są powszechnie używane do implementacji wyrażeń logicznych.
- Logiczne wyrażenie: *jeżeli (X jest TRUE) i (AND) (Y jest TRUE), to (Z jest TRUE), w przeciwnym przypadku (Z ma wartość FALSE).*
- *NOT* oznaczamy znakiem tyldy ~ albo poziomą kreską nad zmienną lub wyrażeniem, np. $\overline{B}$.
- Dla wygody używamy symbolu 1, TRUE lub HIGH wymiennie, podobnie jak 0, FALSE oraz LOW.

Tabela 2: Wybrane operacje logiczne i ich symbole

operator logiczny	symbol
AND	·
OR	+
NOT	~ ()

Podstawy algebry Boole'a

Tablice (tabele) prawdy

Tablica prawdy funkcji logicznej – wszystkie możliwe kombinacje wartości wejściowych i odpowiadające im wartości wyjściowe.

Tabela 3: Tablica prawdy dla iloczynu logicznego $Z = X \cdot Y$

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

Tabela 4: Tablica prawdy dla sumy logicznej $Z = X + Y$

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

Podstawy algebry Boole'a

Tabela 5: Tablica prawdy dla negacji logicznej $Z = \text{NOT } X$

X	Z
0	1
1	0

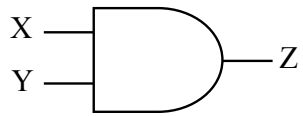
Tabela 6: Tablica prawdy dla wyrażenia logicznego $C = A + \overline{B}$

A	B	C
0	0	1
0	1	0
1	0	1
1	1	1

Podstawowe bramki logiczne

Wprowadzenie

- **Inna reprezentacja funkcji boolowskich wykorzystuje bramki logiczne**.
- Bramki zbudowane są z elementów podstawowych w postaci układów scalonych, a jedną z możliwych realizacji jest omawiana dalej **technologia wykorzystująca tranzystory MOS** (obecnie dominująca technologia wytwarzania układów scalonych).
- Oznaczenie bramki realizującej operację logiczną $Z = X$ AND Y pokazano na rys. 7.



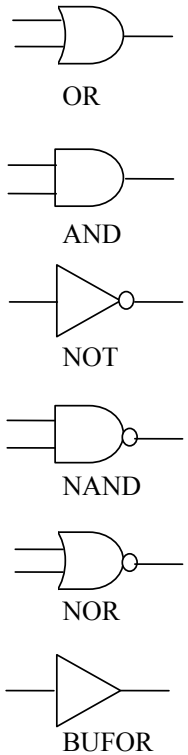
Rys. 7: Bramka AND

Podstawowe bramki logiczne

Wprowadzenie

- Wyjście bramki **układu kombinacyjnego** jest wyłącznie funkcją jej wejść.
- **Wszystkie bramki muszą zapewniać wartości gwarantowane napięć odpowiadających poziomom logicznym (ang. static discipline)** – jeżeli na wejścia podane zostaną sygnały należące do prawidłowego zakresu wejściowego, na wyjściu pojawią się sygnały należące do prawidłowego zakresu wyjściowego.
- **Układ kombinacyjny** – abstrakcyjna reprezentacja obwodu, która spełnia dwie właściwości: wyjścia bramki są funkcją wejść bramki oraz spełnione są wymagania dotyczące wartości gwarantowanych poziomów logicznych.

Podstawowe bramki logiczne



Rys. 8: Symbole wybranych bramek logicznych

Podstawowe bramki logiczne

Podstawowe bramki logiczne

- **Bramka OR** realizuje funkcję LUB wejść.
- **Bramka NOT** umieszcza na wyjściu wartość przeciwną do wartości wejściowej (negacja, wartość komplementarną, np. 0–1, 1–0). Dla wygody, często oznacza się funkcję NOT w układach logicznych za pomocą symbolu $\circ$.
- **Bramka bufora** lub bramka tożsamości kopiuje wartość wejściową na wyjście, czyli $A = A$.
- **Funkcja NAND** jest odpowiednikiem operacji AND, po której następuje operacja NOT.
- Na przykład $A = B$ NAND C jest równoważna $A = \overline{B \text{ AND } C}$. Podobnie **operacja NOR** jest równoważna operacji OR, a następnie operacji NOT.

Podstawowe bramki logiczne

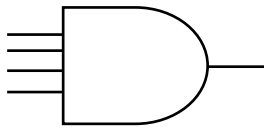
Tabela 7: Tablica prawdy dla kilku funkcji dwuwejściowych

WEJŚCIA		AND	OR	NAND	NOR
A	B	$B \cdot C$	$B + C$	$\overline{B \cdot C}$	$\overline{B + C}$
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	0

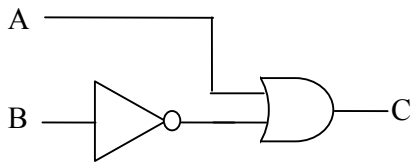
Podstawowe bramki logiczne

Podstawowe bramki logiczne

- Bramki mogą mieć wiele wejść.
- Bramki logiczne można łączyć za pomocą przewodów w celu zrealizowania bardziej złożonych funkcji logicznych (rys. 10).



Rys. 9: Czterowejściowa bramka AND

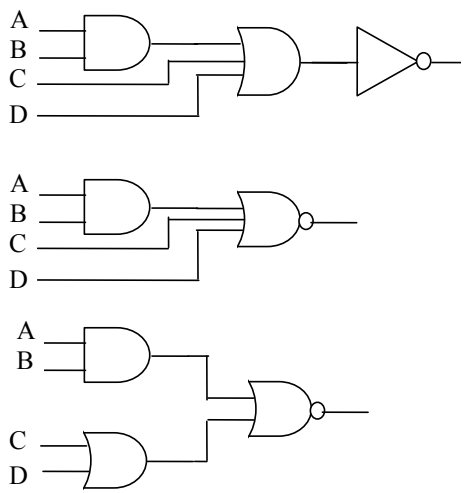


Rys. 10: Realizacja funkcji $C = A + \overline{B}$ za pomocą bramek logicznych

Podstawowe bramki logiczne

Podstawowe bramki logiczne

- Wyrażenie logiczne $\overline{AB + C + D}$ można zrealizować na kilka sposobów.
- Możemy użyć jednej bramki 2-wejściowej AND, bramki OR o trzech wejściach oraz inwertera (NOT), jak pokazano na rys. 11.
- Możemy również zastąpić parę bramek AND i NOT bramką NOR.
- Alternatywnie, przepisując wyrażenie jako $\overline{((AB) + (C + D))}$, możemy zaimplementować tę samą funkcję za pomocą bramek: AND, OR oraz NOR.



Rys. 11: Różne realizacje tej samej funkcji logicznej

Podstawowe bramki logiczne

Podstawowe bramki logiczne

- Wyrażenia logiczne można przedstawić w postaci tablic prawdy lub realizując je na poziomie bramek logicznych.
- **W jaki sposób można automatycznie wyznaczyć wyrażenie logiczne na podstawie tabeli prawdy?**
- **Standardowa (kanoniczna) forma zapisu wyrażeń logicznych – postać sumy iloczynów**.

Podstawowe bramki logiczne

Podstawowe bramki logiczne

- **Można zapisać wyrażenie logiczne w formie kanonicznej na podstawie tablicy prawdy najpierw zapisując iloczyn dla każdego wiersza w tabeli prawdy, dla którego w kolumnie wyjściowej jest logiczna jedynka, a następnie sumując otrzymane iloczyny.**
- Zmienna w iloczynach zapisywana jest w postaci podstawowej, gdy w danym wierszu występuje dla niej logiczna jedynka lub komplementarnej w przypadku przeciwnym.
- Postępując w ten sposób – $Z = \overline{X}Y + X\overline{Y} + XY$ dla tablicy prawdy pokazanej w tabeli 8.

Tabela 8: Tablica prawdy dla sumy logicznej $Z = X + Y$

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

Upraszanie wyrażeń logicznych

Upraszanie wyrażeń logicznych

- **Uproszczenie wyrażeń logicznych minimalizuje koszty realizacji.**
- Można wykazać, że wyrażenie $A + B\overline{B} + C$ (wymagające trzech bramek), jest równoważne wyrażeniu $A + C$, które wymaga jednej bramki OR (potwierdza to wynik w tablicy prawdy 9).

Tabela 9: Tablica prawdy dla dwóch funkcji logicznych $Y = A + B\overline{B} + C$, $Z = A + C$

A	B	C	$Y = A + B\overline{B} + C$	$Z = A + C$
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Upraszanie wyrażeń logicznych

Upraszanie wyrażeń logicznych

Podstawowe zasady upraszczania wyrażeń logicznych:

$A \cdot \overline{A} = 0$ $\quad$ $A \cdot A = A$

$A \cdot 0 = 0$ $\quad$ $A \cdot 1 = A$

$A + \overline{A} = 1$ $\quad$ $A + A = A$

$A + 0 = A$ $\quad$ $A + 1 = 1$

$A + \overline{A}B = A + B$ $\quad$ $A(B + C) = AB + AC$

$AB = BA$ $\quad$ $A + B = B + A$

$(AB)C = A(BC)$ $\quad$ $(A + B) + C = A + (B + C)$

Upraszanie wyrażeń logicznych

Podane reguły można wykorzystać do uproszczenia wyrażeń logicznych w celu zmniejszenia liczby bramek wymaganych do realizacji sprzętowej (ważne np. w układach FPGA).

Upraszanie wyrażeń logicznych

Upraszanie wyrażeń logicznych

Zasady te można sprawdzić, porównując tablice prawdy dla obu stron odpowiednich równań (np. tabela 10).

Tabela 10: Tablica prawdy dla dwóch funkcji logicznych $Y = A + \overline{A}B$, $Z = A + B$

A	B	$\overline{A}B$	$Y = A + \overline{A}B$	$Z = A + B$
0	0	0	0	0
0	1	1	1	1
1	0	0	1	1
1	1	0	1	1

Upraszanie wyrażeń logicznych

Prawa de Morgana

$$\overline{A \cdot B} = \overline{A} + \overline{B}, \tag{1}$$

$$\overline{A + B} = \overline{A} \cdot \overline{B} \tag{2}$$

są niezwykle użyteczne w procesie upraszczania zapisu wyrażeń logicznych (tabela 11).

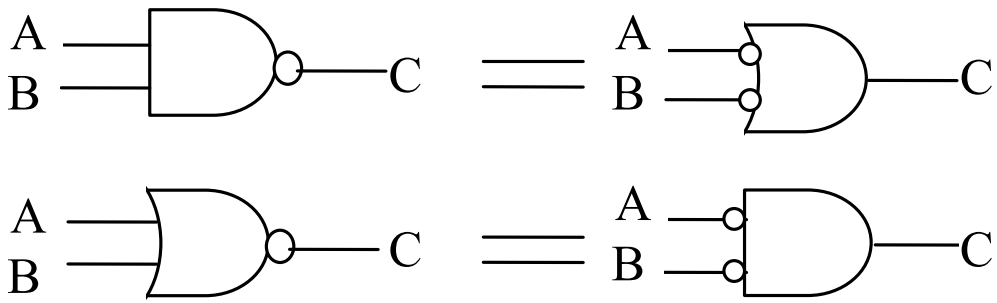
Tabela 11: Tablica prawdy dla praw De Morgana

A	B	$\overline{A}$	$\overline{B}$	$A \cdot B$	$A + B$	$\overline{A \cdot B}$	$\overline{A + B}$	$\overline{A} \cdot \overline{B}$	$\overline{A} + \overline{B}$
0	0	1	1	0	0	1	1	1	1
0	1	1	0	0	1	1	0	0	1
1	0	0	1	0	1	1	0	0	1
1	1	0	0	1	1	0	0	0	0

Upraszanie wyrażeń logicznych

Prawa de Morgana

Prawa de Morgana w kategoriach bramek logicznych – rys. 12.



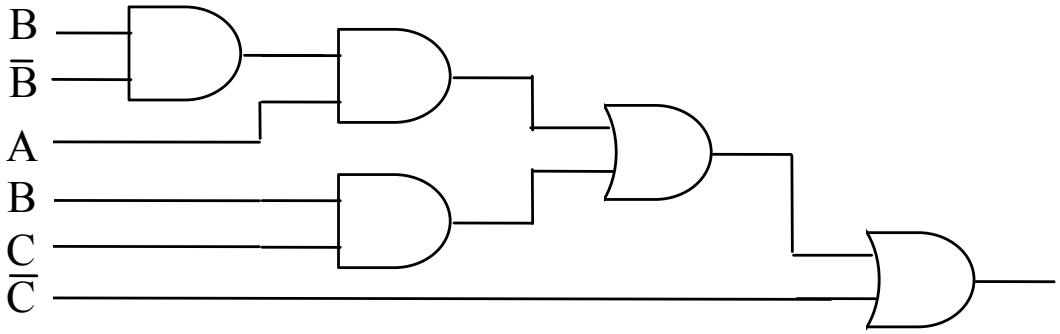
Rys. 12: Realizacja praw de Morgana za pomocą bramek logicznych

Upraszanie wyrażeń logicznych

Upraszanie wyrażeń logicznych

Bezpośrednia implementacja wyrażenia logicznego $AB\overline{B} + BC + \overline{C}$ wymaga zastosowania **pięciu bramek dwuwejściowych**, jak pokazano na rys. 13, **przy założeniu dostępności zmiennych w postaci podstawowej i komplementarnej.**

W przeciwnym razie konieczne byłoby zastosowanie jeszcze dwóch dodatkowych bramek NOT (inwerterów).

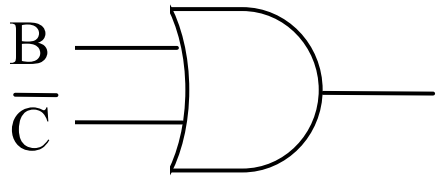


Rys. 13: Bezpośrednia realizacja sprzętowa wyrażenia $AB\overline{B} + BC + \overline{C}$

Upraszanie wyrażeń logicznych

Upraszanie wyrażeń logicznych

Uproszczenie wyrażenia $AB\overline{B} + BC + \overline{C}$ w następujący sposób:
$AB\overline{B} + BC + \overline{C} = A(B\overline{B}) + BC + \overline{C} = A0 + BC + \overline{C} = 0 + BC + \overline{C} = (0 + BC) + \overline{C} = BC + \overline{C} = B + \overline{C}$
co umożliwia implementację w postaci **jednej bramki** (rys. 14).



Rys. 14: Najprostsza realizacja sprzętowa wyrażenia $AB\overline{B} + BC + \overline{C}$

Reprezentacja liczb

Reprezentacja liczb

- **Binarna reprezentacja przyporządkowuje sygnałowi wysoki lub niski poziom logiczny.**
- Te dwie wartości mogą służyć do reprezentowania dwóch liczb na przykład 0 i 1.
- Jedna cyfra dziesiętna może reprezentować jedną z dziesięciu wartości (0, 1, 2, ..., 9), pojedyncza binarna cyfra (określana jako bit) reprezentuje jedną z dwóch wartości (0, 1). **Większe liczby są skonstruowane przez łączenie wielu cyfr.**
- Liczba dziesiętna *sdj* utworzona przez połączenie cyfr *s*, *d* i *j* ma wartość

$$s \cdot 10^2 + d \cdot 10^1 + j \cdot 10^0. \tag{3}$$

- Liczba binarna *lmn* utworzona przez łączenie liczb binarnych *l*, *m* i *n* ma wartość

$$l \cdot 2^2 + m \cdot 2^1 + n \cdot 2^0. \tag{4}$$

Reprezentacja liczb

Reprezentacja liczb

- Wartość binarnej liczby $A_n A_{n-1} \ldots A_2 A_1 A_0$ jest określona przez

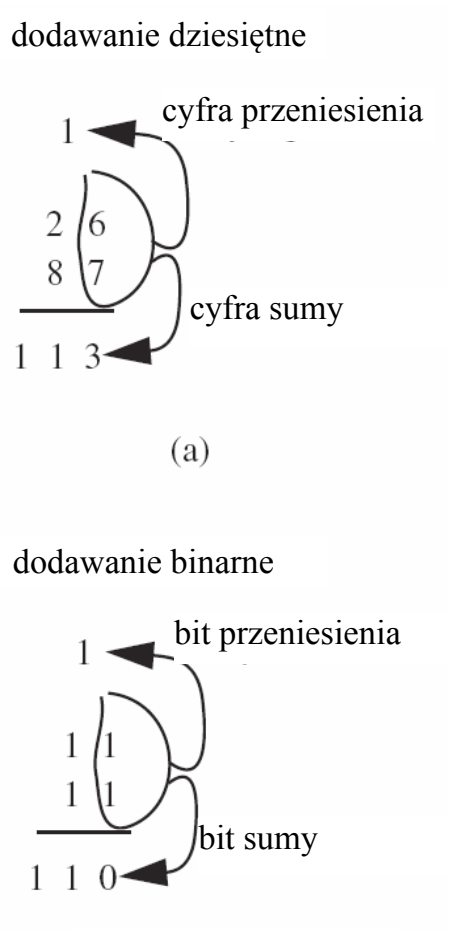
$$\sum_{i=0}^{n} A_i 2^i. \tag{5}$$

  np. liczba 110 odpowiada dziesiętnej wartości 6.

- **W celu reprezentacji liczb ujemnych można wprowadzić następującą interpretację pierwszego bitu (od lewej) i przyjąć, że 0 oznacza liczbę dodatnią, a 1 oznacza liczbę ujemną.** -jedna z wielu możliwych reprezentacji, t.zw. zapis znak - moduł (w skrócie ZM - ang. SM Signed Magnitude) najbardziej zbliżony do systemu zapisu liczb używanego przez nas, $LZM = (-1)^{bit\ znaku} \times modul\ liczby$. Zatem liczba 110 oznacza -2 $((-1)^1 * (2^1))$, a liczba 010 reprezentuje 2 $((-1)^0 * (2^1))$.

- Operacje na liczbach binarnych mogą być wykonywane w sposób analogiczny do operacji na liczbach dziesiętnych, jak to pokazano na rys. 15.

Reprezentacja liczb



Rys. 15: Dodawanie liczb dziesiętnych i binarnych

Reprezentacja liczb

Reprezentacja liczb

Uwagi:
– dodawanie cyfr generuje sumę i cyfrę przeniesienia
– dodanie pary liczb dwucyfrowych może czasami skutkować wynikiem w postaci liczby trzycyfrowej.

Realizacja sprzętowa sumatora

Sumator

- Cel: dodać parę dwóch binarnych liczb dodatnich A (zapis A_1A_0) oraz B (B_1B_0).
- **Metoda pierwsza polega na utworzeniu tablicy prawdy (tabela 12) dla operacji dodawania z przeniesieniem.**

Tabela 12: Tablica prawdy dla sumatora dwubitowego

A_1	A_0	B_1	B_0	S_2	S_1	S_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

Sumator

Sumator

- Z tabeli prawdy otrzymujemy reprezentację sum iloczynów:

$$\begin{aligned} S_0 &= \overline{A_1 A_0 B_1} B_0 + \overline{A_1 A_0} B_1 B_0 + \overline{A_1} A_0 \overline{B_1 B_0} + \overline{A_1} A_0 B_1 \overline{B_0} \\ &+ A_1 \overline{A_0 B_1} B_0 + A_1 \overline{A_0} B_1 B_0 + A_1 A_0 \overline{B_1 B_0} + A_1 A_0 B_1 \overline{B_0} \\ &= \overline{A_0} B_0 + A_0 \overline{B_0}, \end{aligned} \tag{6}$$

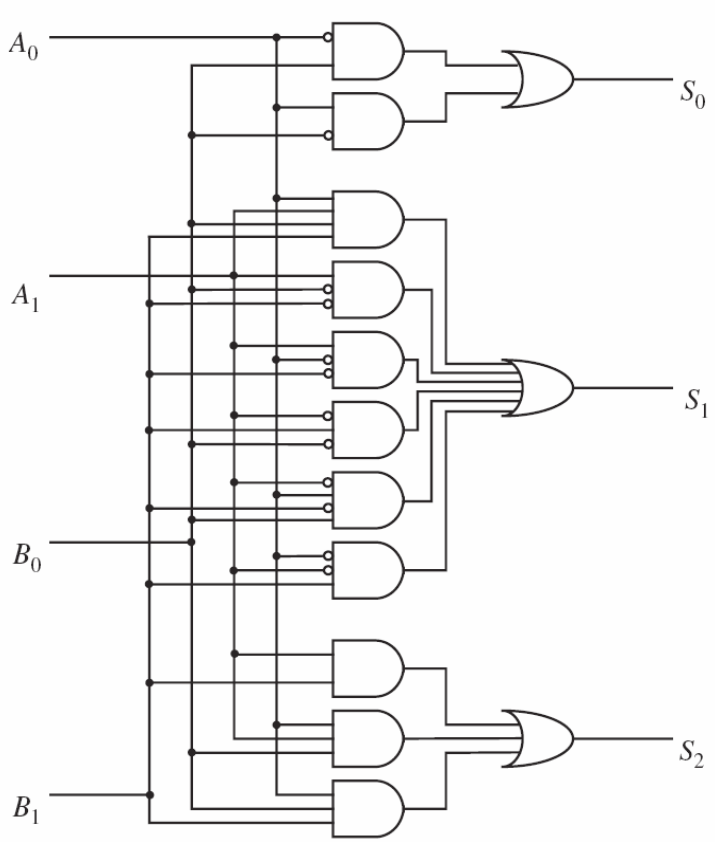
$$\begin{aligned} S_1 &= \overline{A_1 A_0} B_1 \overline{B_0} + \overline{A_1 A_0} B_1 B_0 + \overline{A_1} A_0 \overline{B_1} B_0 + \overline{A_1} A_0 B_1 \overline{B_0} \\ &+ A_1 \overline{A_0 B_1 B_0} + A_1 \overline{A_0} \overline{B_1} B_0 + A_1 A_0 \overline{B_1 B_0} + A_1 A_0 B_1 B_0 \\ &= A_1 A_0 B_1 B_0 + A_1 \overline{B_1 B_0} + A_1 \overline{A_0 B_1} + \overline{A_1} B_1 \overline{B_0} + \overline{A_1} A_0 \overline{B_1} B_0 + \overline{A_1 A_0} B_1, \end{aligned} \tag{7}$$

$$\begin{aligned} S_2 &= \overline{A_1} A_0 B_1 B_0 + A_1 \overline{A_0} B_1 \overline{B_0} + A_1 \overline{A_0} B_1 B_0 + A_1 A_0 \overline{B_1} B_0 \\ &+ A_1 A_0 B_1 \overline{B_0} + A_1 A_0 B_1 B_0 = A_1 B_1 + A_1 A_0 B_0 + A_0 B_1 B_0. \end{aligned} \tag{8}$$

Sumator

Sumator

Stąd bezpośrednia realizacja sprzętowa (rys. 16).

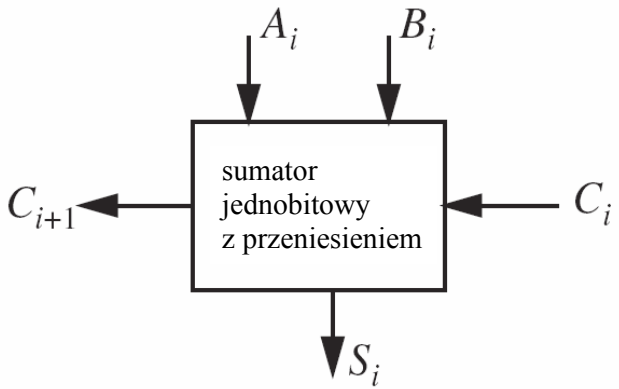


Rys. 16: Bezpośrednia realizacja sprzętowa sumatora dwubitowego

Sumator

Sumator

- **Druga metoda polega na realizacji sprzętowej sumatora jednobitowego przy wykorzystaniu tabeli prawdy, a następnie jego wykorzystanie do realizacji sprzętowej sumatora dwubitowego.**
- Jest to przykład często stosowanej techniki zwanej metodą dziel i rządź (zwyciężaj) **(ang. divide-and-conquer)**.
- **Sumatory jednobitowe (ang. full adders)** mają trzy wejścia – dwa odpowiadające dwóm cyfrom, które są dodawane (A_i oraz B_i), jedno odpowiadające bitowi przeniesienia z pozycji niższej (C_i) oraz dwa wyjścia: bit sumy S_i i bit przeniesienia na wyższą pozycję C_{i+1} (rys. 17).



Rys. 17: Symbol sumatora jednobitowego

Sumator

Sumator

Tablica prawdy jednobitowego sumatora – tabela 13.

Tabela 13: Tablica prawdy dla sumatora jednobitowego z przeniesieniem

A_i	B_i	C_i	C_{i+1}	S_i
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

Sumator

Sumator

Wyrażenie logiczne dla sumy bitowej S_i oraz bitu przeniesienia C_{i+1}:

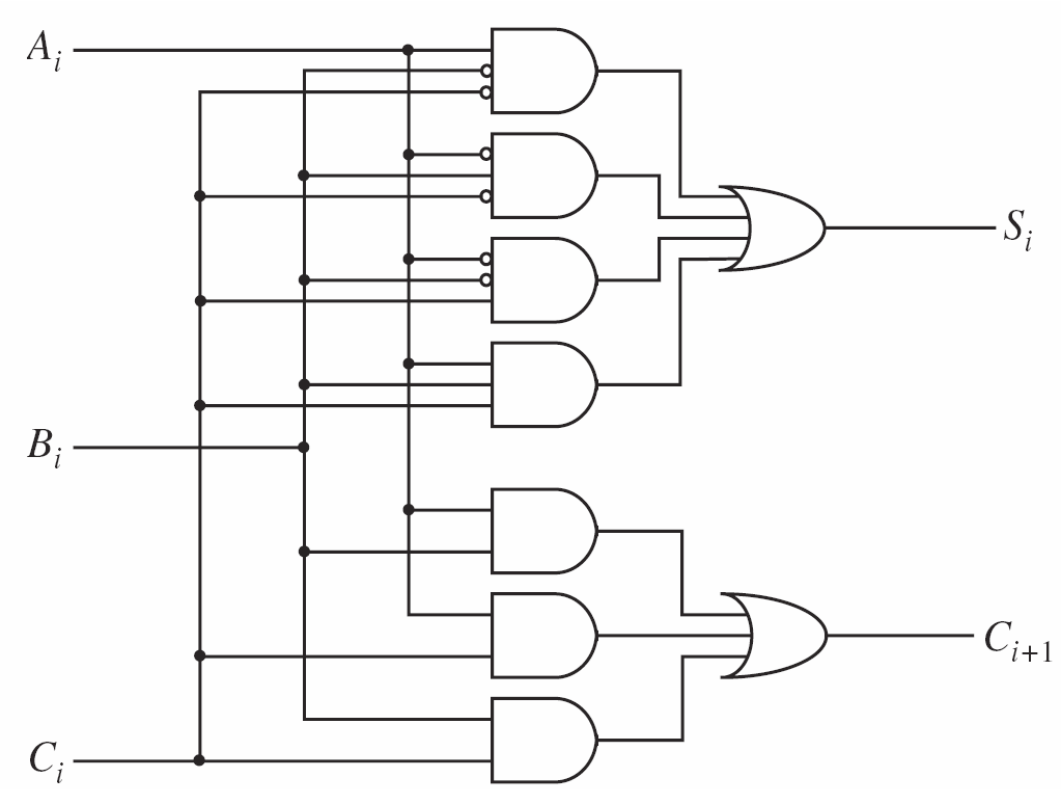
$$S_i = \overline{A_i}\,\overline{B_i}C_i + \overline{A_i}B_i\overline{C_i} + A_i\overline{B_i}\,\overline{C_i} + A_iB_iC_i, \tag{9}$$

$$C_{i+1} = \overline{A_i}B_iC_i + A_i\overline{B_i}C_i + A_iB_i\overline{C_i} + A_iB_iC_i. \tag{10}$$

Sumator

Sumator

Rys. 18 – realizacja sprzętową sumatora jednobitowego z przeniesieniem,

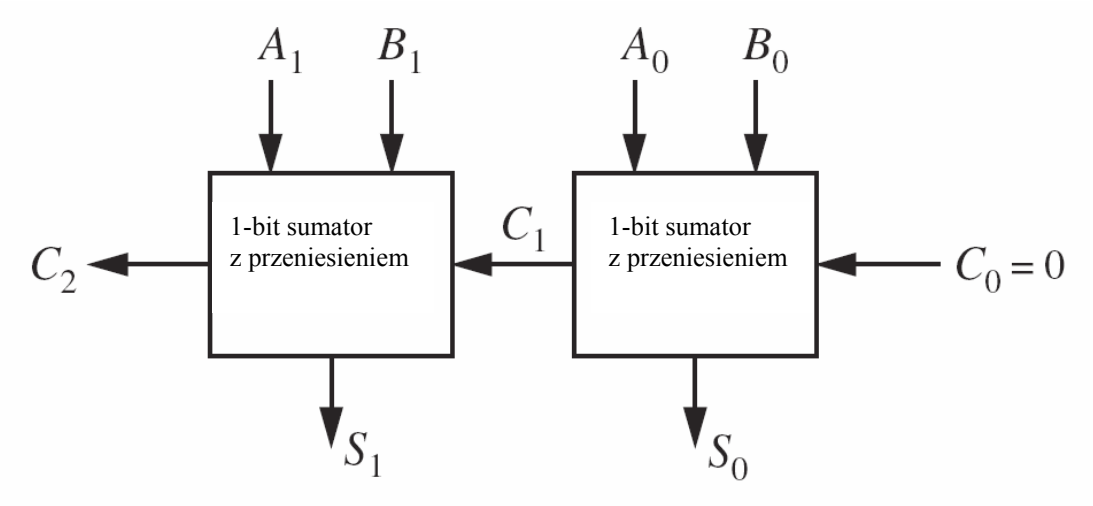


Rys. 18: Realizacja sprzętowa sumatora jednobitowego

Sumator

Sumator

Rys. 19 – schematyczna budowa sumatora dwubitowego złożonego z dwóch sumatorów jednobitowych.



Rys. 19: Schemat sumatora dwubitowego