Programowanie struktur cyfrowych

Podstawowe zagadnienia
Logika binarna i bramki

dr Aleksander Lamża

Uniwersytet J. Kochanowskiego w Kielcach
Uniwersytet Śląski w Katowicach

aleksander.lamza@us.edu.pl

Zawartość

Celem tego wykładu jest przypomnienie i przedstawienie podstawowych zagadnień związanych z logiką binarną i elementami układów cyfrowych, takimi jak bramki, dekodery, multipleksery, przerzutniki itp.

Są to zagadnienia niezbędne do zabawy cyfrowymi układami programowalnymi.

Tematy poruszone w cyklu prezentacji:

- System dwójkowy (binarny)
- Logika binarna
- Bramki logiczne
- Cyfrowe układy kombinacyjne
- Cyfrowe układy sekwencyjne

System dwójkowy

Mam nadzieję, że wszyscy dobrze pamiętają **system dwójkowy**.

Dla przypomnienia – jest to system pozycyjny (podobnie jak system dziesiętny), w którym podstawą jest 2:

$$L = a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + \ldots + a_1 \cdot 2^1 + a_0 \cdot 2^0$$

Mamy więc dwie cyfry: 0 i 1.

Przykładowa liczba

$$1101_2$$

ma wartość dziesiętną

$$L = \mathbf{1} \cdot 2^3 + \mathbf{1} \cdot 2^2 + \mathbf{0} \cdot 2^1 + \mathbf{1} \cdot 2^0$$

$$8 + 4 + 0 + 1$$

$$L = \mathbf{13}$$

Algebra Boole'a

W tej chwili bardziej niż arytmetyka binarna interesuje nas sprawa **algebry Boole'a**. Właściwie interesuje nas tylko niewielki wycinek algebry Boole'a dotyczący funkcji logicznych mających zastosowanie w elektronice cyfrowej i informatyce.

Mowa o **negacji**, **sumie** i **iloczynie**.

Jak wspomniałem, są to funkcje logiczne. Z logiką są związane dwa pojęcia: **prawda** i **fałsz**.
W dalszych rozważaniach będziemy stosować symbole **0** i **1** (cyfry systemu dwójkowego) odpowiadające fałszowi (0) i prawdzie (1).

Logika binarna

Zacznijmy od **negacji**. Sprawa jest prosta:

$$\mathbf{Y = \overline{a}}$$

W tabeli prawdy wygląda to tak:

a	**Y**
0	**1**
1	**0**

Negacja to logiczne „nie”.

Negację można oznaczyć również w inny sposób:
~a, **¬**a, **!**a, **–**a

Logika binarna

Teraz kolej na **sumę logiczną**:

Y = a + b

Tabela prawdy:

a	b	Y
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

Suma to logiczne „lub” – jest równe 1, jeśli a **lub** b jest równe 1.

Sumę można oznaczyć również w inny sposób:
a **∨** b, a **∪** b, a **|** b

Logika binarna

Na koniec **iloczyn logiczny**:

$$\mathbf{Y = a \cdot b}$$

Tabela prawdy:

a	b	Y
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

Iloczyn to logiczne „i” – jest równy 1, jeśli a **i** b jest równe 1.

Iloczyn można oznaczyć również w inny sposób:
a **∧** b, a **∩** b, a **&** b

Logika binarna

Omówione funkcje pozwalają na zrealizowanie dowolnych operacji logicznych i arytmetycznych na wartościach binarnych.

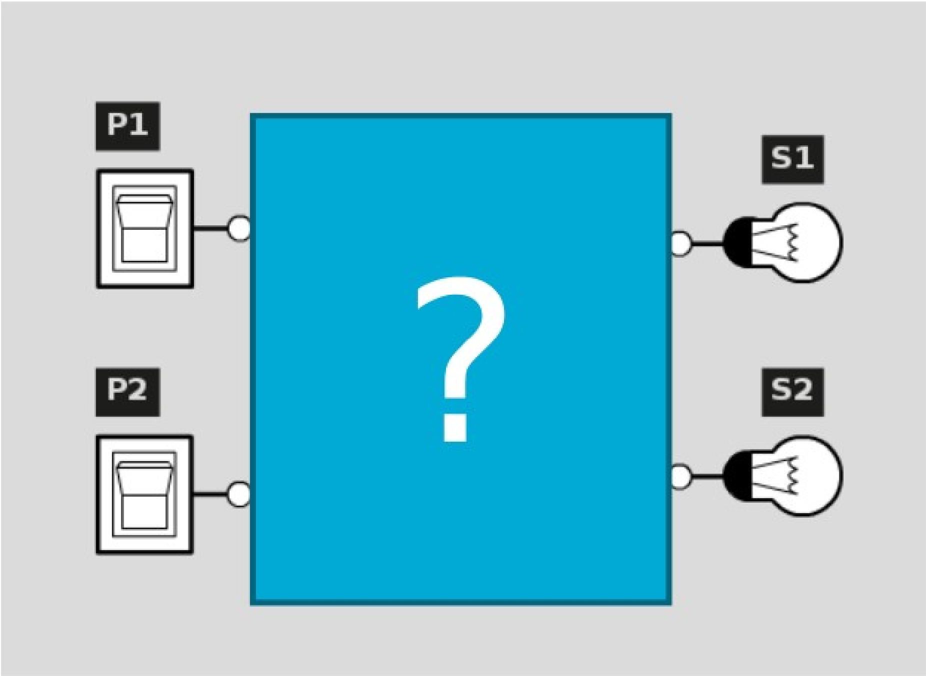
Bardzo często w praktyce będziemy się spotykać z sytuacją, w której znamy zasadę działania i musimy na jej podstawie zbudować działający w ten sposób układ logiczny.

Pierwszym krokiem zawsze jest określenie funkcji logicznej, która ma być realizowana.

Najlepiej będzie to sprawdzić na jakimś prostym przykładzie...

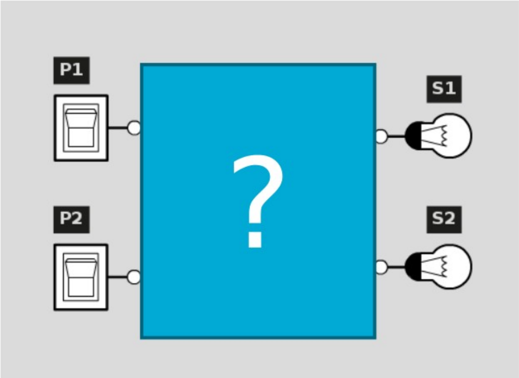
Przykład funkcji

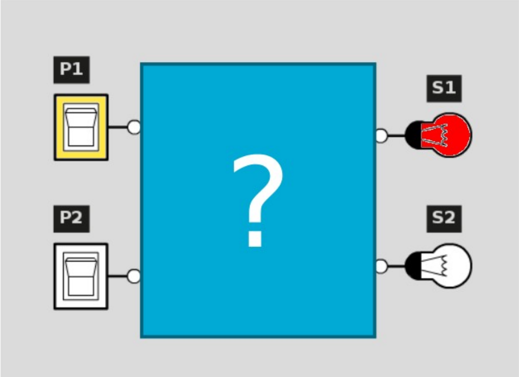
Powiedzmy, że mamy dwa przyciski, którymi chcemy sterować dwoma światłami.

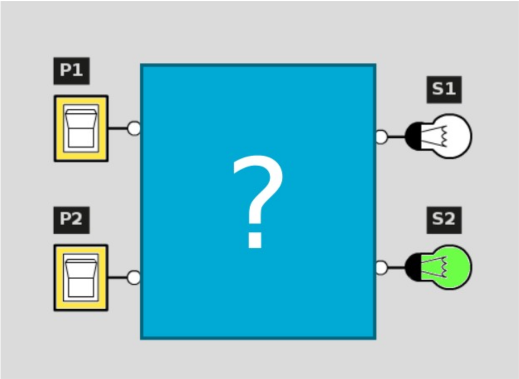


Pierwszy przycisk P1 służy do aktywowania urządzenia (jeżeli jest wyłączony, żadne światło nie świeci).
Jeżeli P1 jest włączony, w zależności od stanu P2 zapala się pierwsze światło S1 (P2 wyłączony) lub drugie S2 (P2 włączony).

Przykład funkcji







Mamy więc dwie główne możliwości:

- S1 świeci, jeśli P1 jest włączony i P2 wyłączony,
- S2 świeci, jeśli P1 jest włączony i P2 włączony.

Zbudujmy tabelę prawdy dla obu świateł. Ponieważ mamy dwa przyciski, możliwości ich włączenia i wyłączenia są cztery:

P1	P2	S1	S2
0	0	**0**	**0**
0	1	**0**	**0**
1	0	**1**	**0**
1	1	**0**	**1**

Teraz już z górki...
Na podstawie tabeli musimy sformułować funkcje logiczne.

Przykład funkcji

P1	P2	S1	S2
0	0	**0**	**0**
0	1	**0**	**0**
1	0	**1**	**0**
1	1	**0**	**1**

Zacznijmy od zielonego (S2). Sprawdzamy, dla jakiego stanu przycisków światło S2 świeci:

S2 = 1 dla P1 = 1 **i** P2 = 1

Ponieważ spójnik „i” to iloczyn logiczny, możemy zapisać:

$S2 = P1 \cdot P2$

Teraz czerwone (S1):

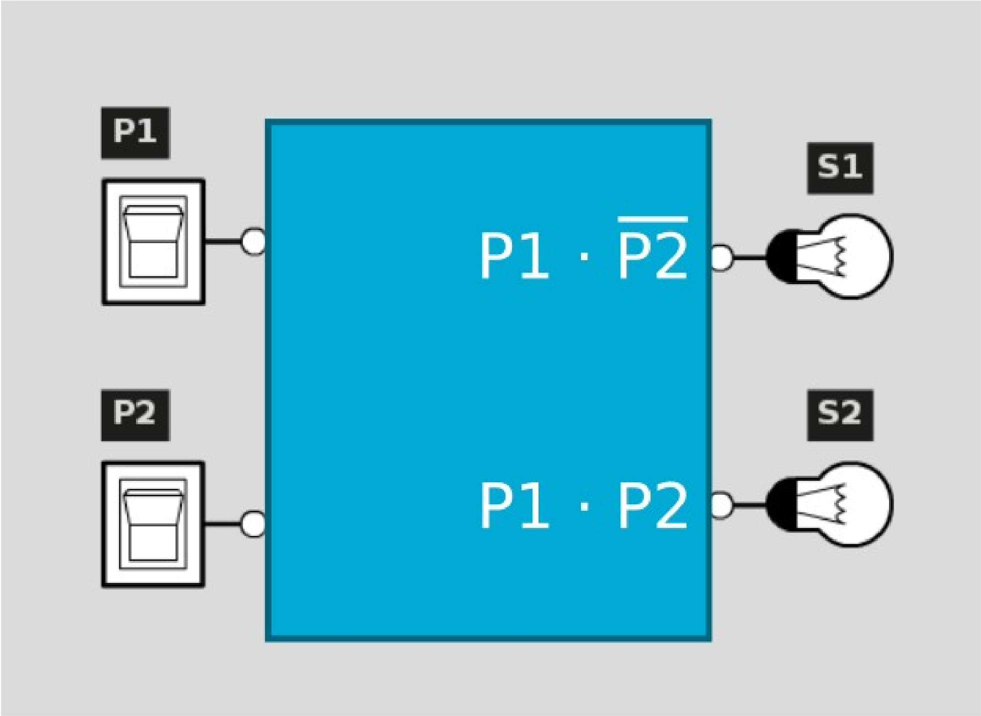
S1 = 1 dla P1 = 1 **i** P2 = 0

Aby w iloczynie uwzględnić P2 = 0, musimy je zanegować:

$S1 = P1 \cdot \overline{P2}$

Przykład funkcji

Funkcje określone! Nie było to takie trudne.



A co gdybyśmy chcieli zbudować taki układ?
Nie będziemy jednak wchodzić w kwestie czysto techniczne i konstruktorskie. Zastanowimy się, jak realizuje się funkcje logiczne (i nie tylko) w rzeczywistych, działających urządzeniach.

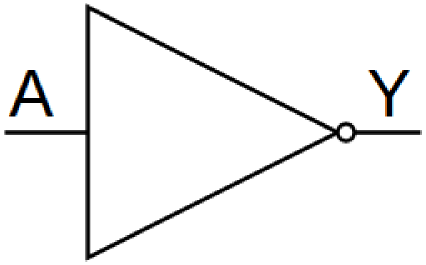
Bramki logiczne

W elektronice cyfrowej podstawowymi elementami są **bramki**.
Z bramek buduje się bardziej złożone układy cyfrowe, takie jak dekodery, sumatory, multipleksery czy przerzutniki. Z nich z kolei buduje się jeszcze bardziej złożone układy, na procesorach i układach programowalnych kończąc.

Ale po kolei – zacznijmy od podstaw.
Na kolejnych slajdach opiszę główne bramki stosowane w elektronice cyfrowej. Są to:

- negacja (**NOT**),
- iloczyn (**AND**) i zanegowany iloczyn (**NAND**),
- suma (**OR**) i zanegowana suma (**NOR**),
- alternatywa (**XOR**) i zanegowana alternatywa (**XNOR**).

Bramka NOT (negacja, inwerter)

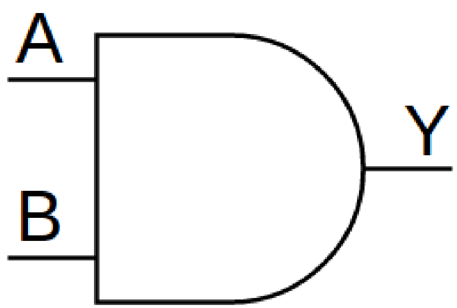


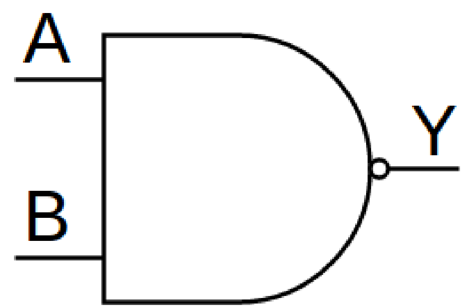
Dla przypomnienia tabela prawdy:

A	Y
0	**1**
1	**0**

Bramki logiczne

Bramka AND (iloczyn) i NAND (zanegowany iloczyn)



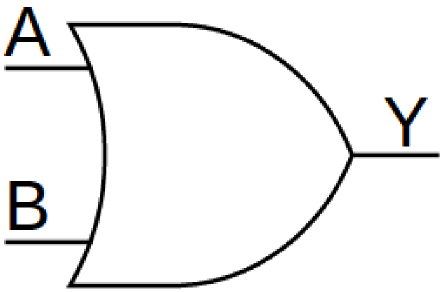


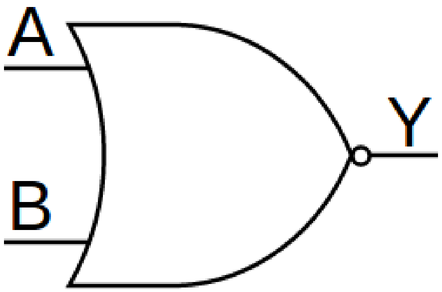
Tabele prawdy:

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

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

Bramka OR (suma) i NOR (zanegowana suma)



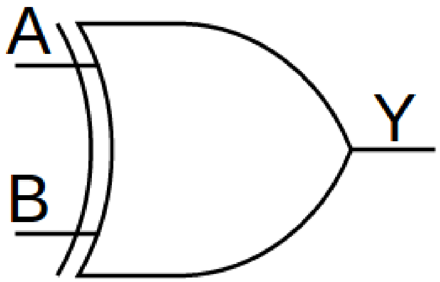


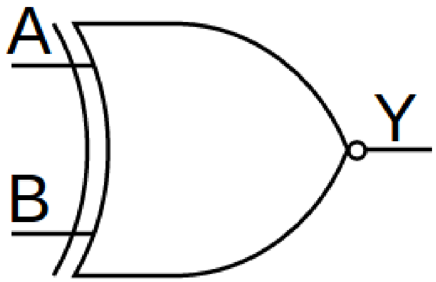
Tabele prawdy:

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

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

Bramka XOR (alternatywa) i XNOR (zanegowana alternatywa)





Tabele prawdy:

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

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

Bramki logiczne

Zwróćcie uwagę, że przy omawianiu funkcji algebry Boole'a nie wspominałem o **alternatywie**. Dlaczego?

Alternatywę da się wyrazić za pomocą podstawowych funkcji (negacja, suma, iloczyn). Jest na to kilka sposobów.

Spróbujcie to zrobić sami. Zastosujcie metodę, którą wykorzystałem przy określaniu funkcji dla świateł i przycisków.

Spróbujcie też narysować schemat układu realizującego wymyśloną funkcję.

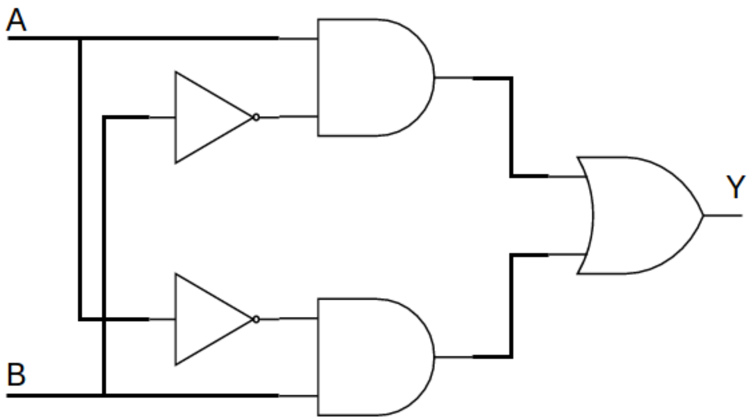
Bramki logiczne

Funkcję realizującą alternatywę można odczytać z tabeli prawdy:

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

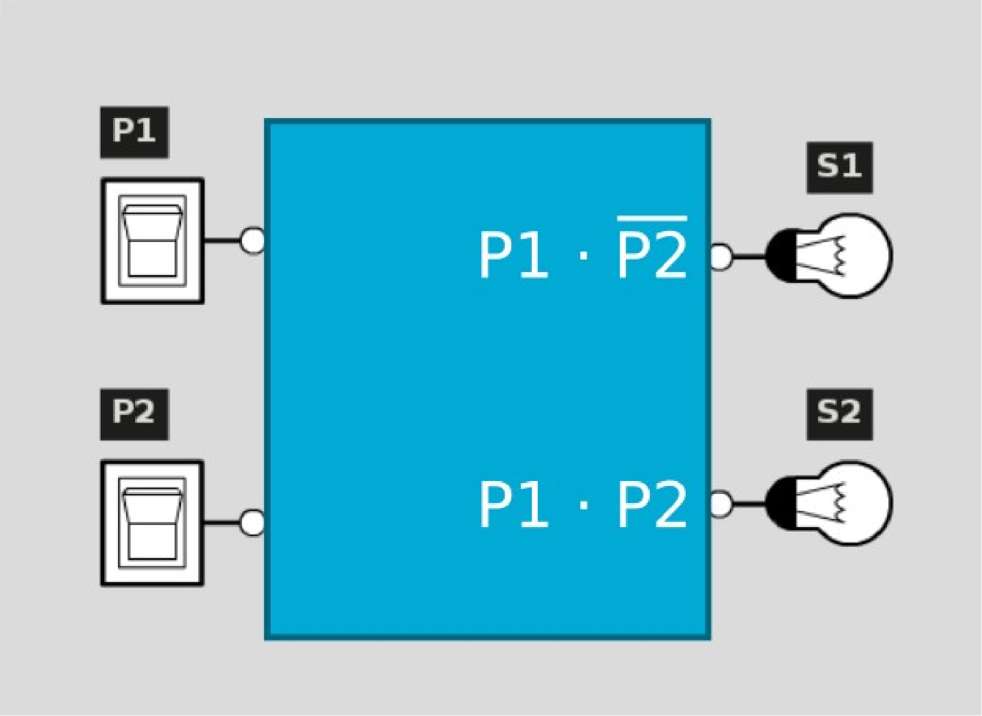
$$Y = A \cdot \overline{B} + \overline{A} \cdot B$$

Schemat wygląda tak:

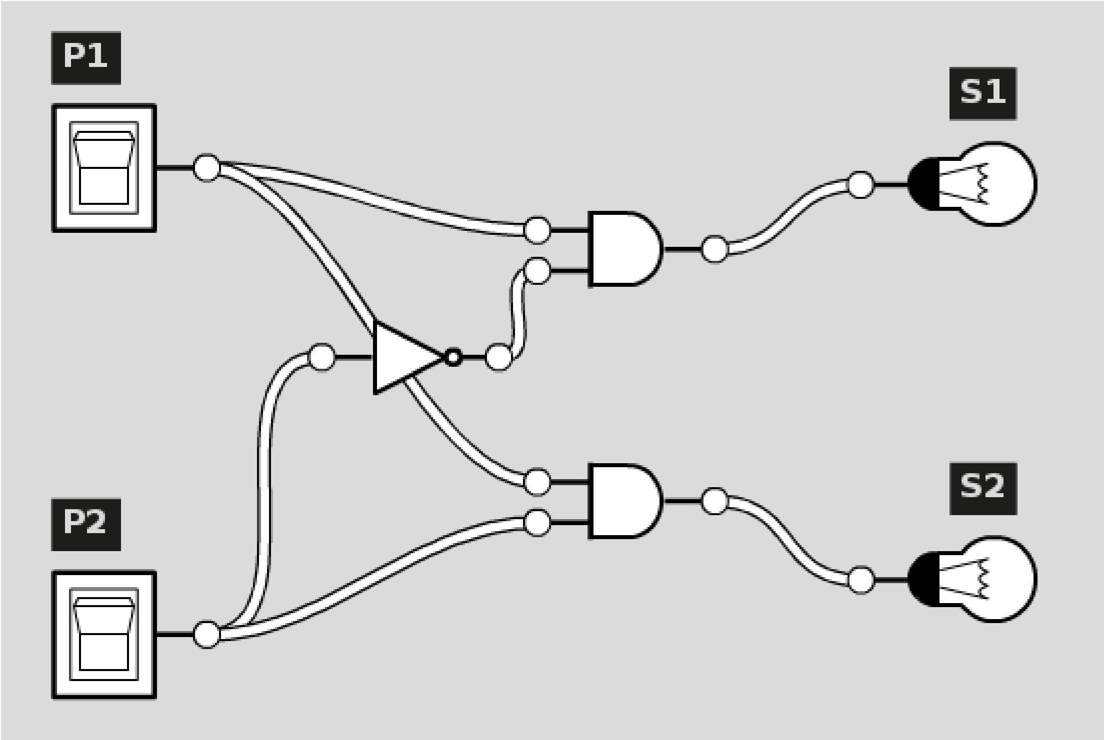


Przy okazji, symbol alternatywy to $\oplus$, a zanegowanej to $\odot$.

Skoro jesteśmy przy ćwiczeniach, może narysowalibyście schemat urządzenia, które analizowaliśmy wcześniej?

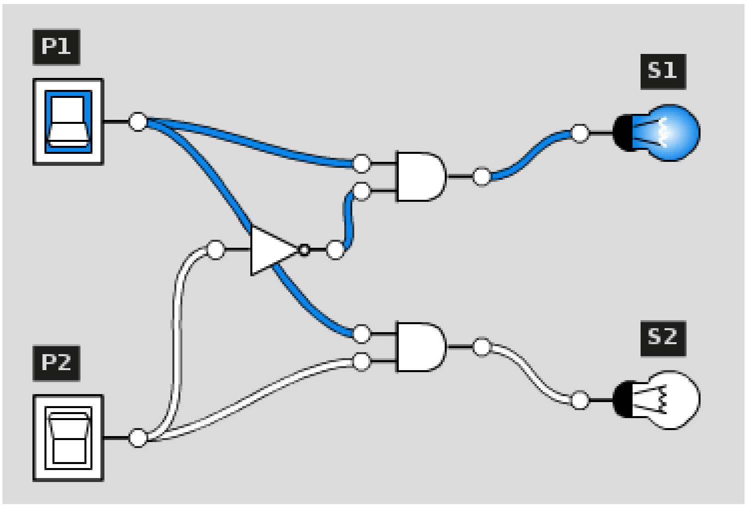


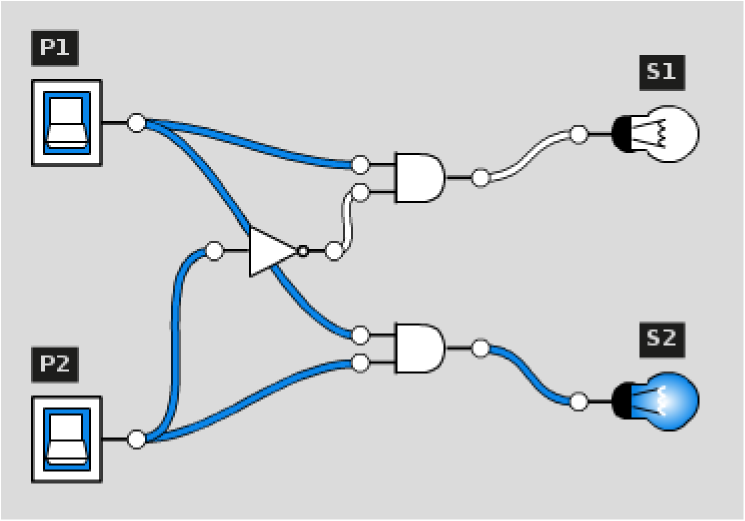
Oto rozwiązanie:



Skorzystałem tu z fajnego narzędzia do zabawy bramkami i innymi układami cyfrowymi: https://logic.ly.

A to symulacja działania. Doskonale widać, gdzie są „jedynki" (stan wysoki, włączenie), a gdzie „zera" (stan niski, wyłączenie).





No dobrze, zabawę pojedynczymi bramkami uważam za zakończoną. Przyszedł czas na ciekawsze sprawy...