PODSTAWY DZIAŁANIA UKŁADÓW CYFROWYCH

1100010100010100010111001110000111000 1100010100010100010111001110000111000 1001100011

- Obecnie telekomunikacja i elektronika zostały zdominowane przez układy cyfrowe i przez cyfrowy sposób przetwarzania sygnałów.
- Cyfrowe przetwarzanie sygnałów ma następujące zalety w stosunku do analogowego sposobu przetwarzania:
  - powtarzalność,
  - większa niezawodność (np. transmisji sygnału),
  - większa możliwość ochrony danych,
  - w wielu sytuacjach jest tańsze.

Systemy liczbowe

- **System dziesiętny (decymalny)**

  Zestaw cyfr {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.

  $(7493)_D = 7 \cdot 10^3 + 4 \cdot 10^2 + 9 \cdot 10^1 + 3 \cdot 10^0$

- **System szesnastkowy (heksalny)**

  Zestaw cyfr {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}.

  $(7493)_H = 7 \cdot 16^3 + 4 \cdot 16^2 + 9 \cdot 16^1 + 3 \cdot 16^0 \rightarrow (29843)_D$

  $(74A3)_H = 7 \cdot 16^3 + 4 \cdot 16^2 + A \cdot 16^1 + 3 \cdot 16^0 \rightarrow (29859)_D$

  $(DE)_H = D \cdot 16^1 + E \cdot 16^0 \rightarrow (222)_D$

- **System dwójkowy (binarny)**

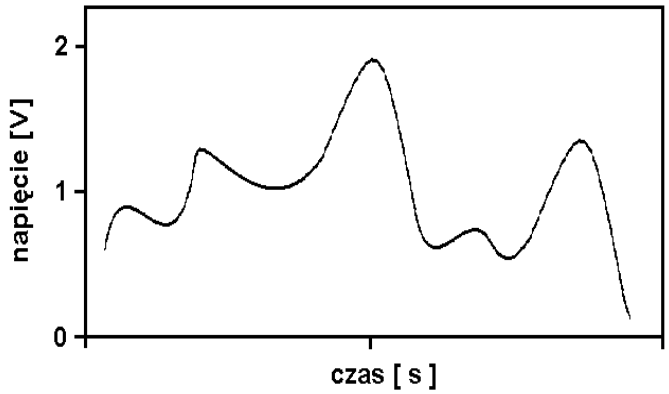
  Zestaw cyfr {0, 1}.

  $10100 = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 \rightarrow (20)_D \quad (14)_H$

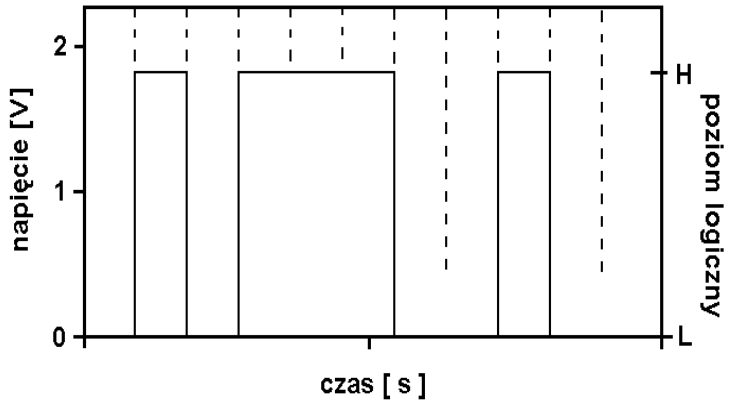
Sygnał analogowy i cyfrowy

1100010100010100010111001110000111001100010100010100010111001110000111000100110001 1

- Sygnał analogowy
  - sygnał, który może przyjmować dowolną wartość z ciągłego przedziału (nieskończonego lub ograniczonego zakresem zmienności)
  - wartości sygnału analogowego mogą zostać określone w każdej chwili czasu dzięki funkcji matematycznej opisującej dany sygnał



- Sygnał cyfrowy binarny
  - sygnał, którego dziedzina i zbór wartości są dyskretne
  - sygnału cyfrowy może zmieniać swoją wartość tylko w określonych momentach czasu i może przyjmować tylko dwie wartości



Sytuacja wyidealizowana.

100 10011000 10011100 1 1 10010

bajt

pojedynczy bit

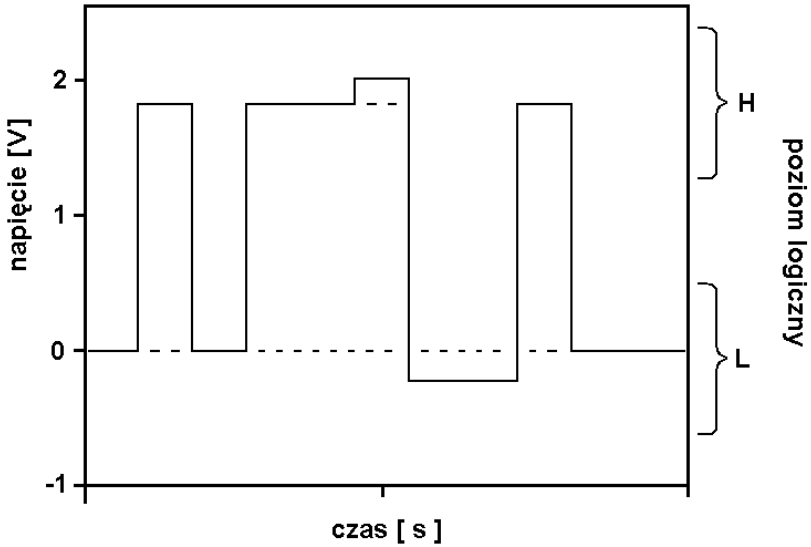
Poziomy logiczne i napięciowe

- W układach cyfrowych informacja reprezentowana jest przez dwa stany:

  stan niski (L), nazywany też poziomem logicznym niskim lub zerem (0),

  stan wysoki (H), nazywany też poziomem logicznym wysokim lub jedynką (1).

- W rzeczywistych układach należy dokładnie określić jakie wartości napięć odpowiadają poziomom logicznym H i L.



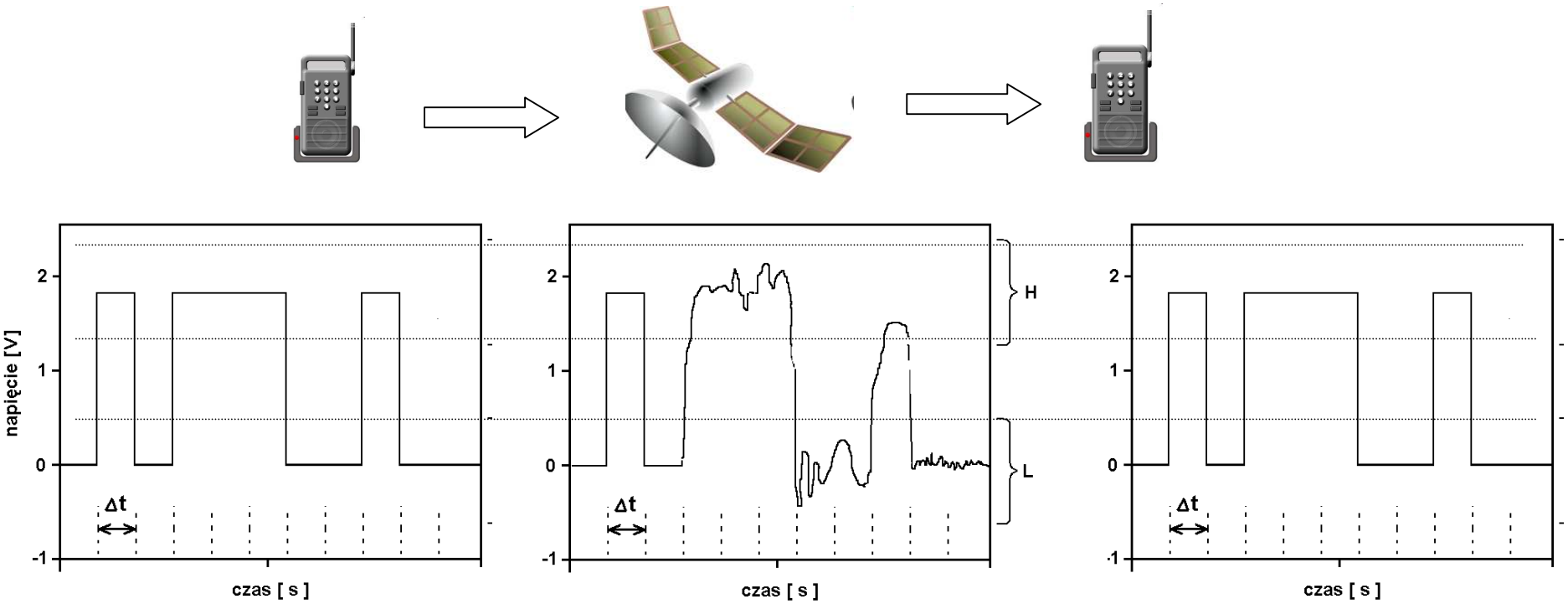
Logika prosta

Stan elektryczny (poziom logiczny)	Reprezentacja cyfrowa binarna	Zakres napięć TTL	Zakres napięć CMOS 1.8V	Zakres napięć CMOS 3.3V
H - high	1	2V – 5.5V	1V – 2V	2V – 3.5V
L - low	0	-0.5V – 0.8V	-0.3V – 0.5V	-0.3V – 1V

Logika odwrócona

Stan elektryczny (poziom logiczny)	Reprezentacja cyfrowa binarna	Zakres napięć TTL	Zakres napięć CMOS 1.8V	Zakres napięć CMOS 3.3V
H - high	0	2V – 5.5V	1V – 2V	2V – 3.5V
L - low	1	-0.5V – 0.8V	-0.3V – 0.5V	-0.3V – 1V

Odporność na zakłócenia



Sygnał oryginalny | Sygnał zakłócony | Sygnał zrekonstruowany

Współczynnik BER (bit error ratio):

$$BER = \frac{liczba\ bitów\ przekłamanych}{całkowita\ liczba\ bitów}$$

Odporność na zakłócenia

Piksel

Chcemy mieć $2^{10} = 1024$ odcienie szarości.

Reprezentacja analogowa:
1024 kolejnych wartości napięć

1 V
1023/1024 V
1022/1024 V
...
3/1024 V
2/1024 V
1/1024 V
0 V

Reprezentacja cyfrowa
1024 kolejne wartości binarne

1111111111
1111111110
1111111101
...
0000000011
0000000010
0000000001
0000000000

Logika – prawda i fałsz

1100010100010100010111001110000111000 1100010100010100010111001110000111000100110001 1

Działanie cyfrowych układów, które spotykamy na co dzień, oparte jest na prawach logiki D'Morgana i algebrze Bool'a (*materiał matematyki szkoły średniej*). Logika D'Morgana zakłada dwie wartości logiczne, prawdę i fałsz. Z tego powodu nazywana jest często **logiką dwuwartościową**. Do reprezentacji obu wartości logicznych doskonale nadaje się system binarny.

Logika prosta

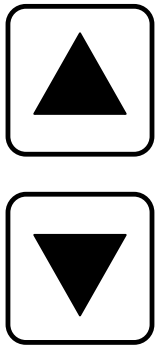
Stan elektryczny	Wartość logiczna	Reprezentacja cyfrowa binarna
H - high	True	1
L - low	False	0

Logika prosta

Stan elektryczny	Wartość logiczna	Reprezentacja cyfrowa binarna
H - high	True	1
L - low	False	0

Jest możliwe budowanie układów logicznych cyfrowych z **logiką wielowartościową**, czyli opartych np. o system trójkowy. Jednakże, synteza, analiza działania i realizacja techniczna są bardzo trudne.

Przykład 1 – prosta logika sterowania windą

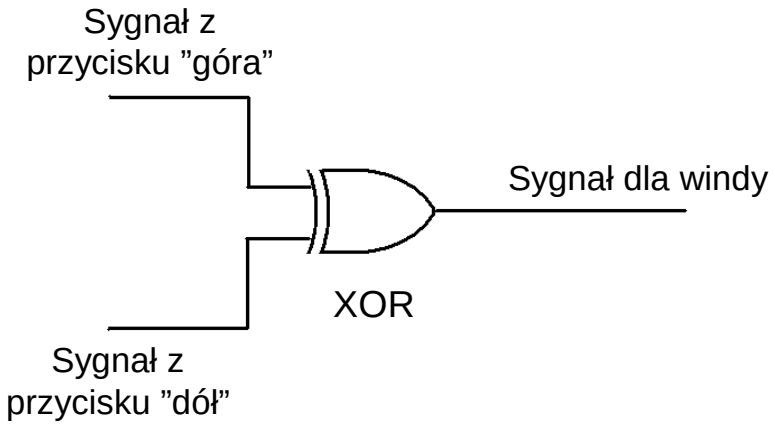


Przycisk ”góra” niewciśnięty = stan logiczny ”false”
Przycisk ”góra” wciśnięty = stan logiczny ”true”
Przycisk ”dół” niewciśnięty = stan logiczny ”false”
Przycisk ”dół” wciśnięty = stan logiczny ”true”

Winda ”jedź” = stan logiczny ”true”
Winda ”stój” = stan logiczny ”false”

Tabela prawdy

Prz. „góra”	Prz. „dół"	Winda
Niewciśnięty (False)	Niewciśnięty (False)	„stój” (False)
Wciśnięty (True)	Wciśnięty (True)	„stój” (False)
Wciśnięty (True)	Niewciśnięty (False)	„jedź” (True)
Niewciśnięty (False)	Wciśnięty (True)	„jedź” (True)



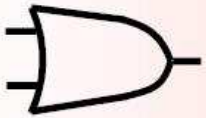
Bramki logiczne

- **rodzaje bramek (funkcje logiczne, symbole, tablice prawdy)**

- AND $\rightarrow y = x_1 \bullet x_2$

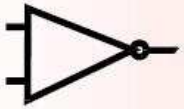
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

- OR $\rightarrow y = x_1 + x_2$



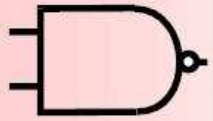
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

- NOT $\rightarrow y = \overline{x}$



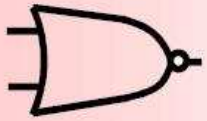
x	x_2
0	1
1	0

- NAND $\rightarrow y = \overline{x_1 \bullet x_2}$



x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

- NOR $\rightarrow y = \overline{x_1 + x_2}$



x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	0

- EXOR $\rightarrow y = x_1 \oplus x_2$

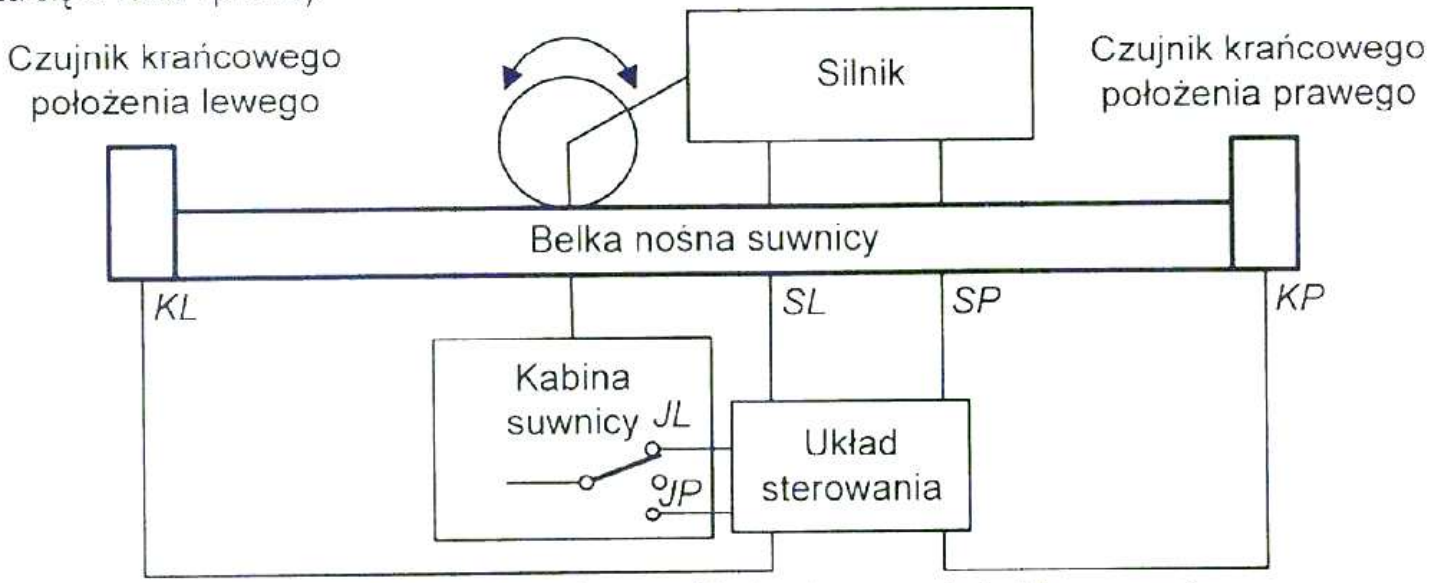
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

- **oprócz bramek dwuwejściowych dostepne są również bramki wielowejściowe**

Przykład 2 – sterowanie suwnicą

1100010100010100010111001110000111000 1100010100010100010111001110000111000100110011

Układ sterowania suwnicą posiada 4 wejścia: *JL*, *JP*, *KL* i *KP*. Wejścia *JL* i *JP* służą do określenia kierunku przesuwu suwnicy odpowiednio w lewo i prawo. Jeżeli suwnica ma się poruszać w lewo, to *1* podawana jest na wejście *JL*, jeżeli natomiast w prawo, to na wejście *JP*. Kiedy suwnica zbliży się do końca belki nośnej, na wejścia *KL* lub *KP* podawany jest stan *1*, co powinno spowodować jej zatrzymanie. Układ sterujący ma dwa wyjścia: *SL* i *SP*. Gdy *SL* = *1*, suwnica jest przesuwana w lewo, dla *SP* = 1 przemieszcza się w prawo. Gdy *SP* = *SL* = *0*, suwnica pozostaje nieruchoma. Zatem w układzie kombinacyjnym nie mogą wystąpić następujące sytuacje: *KP* = *KL* = *1* (suwnica jednocześnie dojechała do lewego i prawego krańca); *JL* = *JP* = *1* (suwnica ma jednocześnie przemieszczać się w prawo i w lewo) oraz *SL* = *SP* = *1* (suwnica równocześnie porusza się w lewo i prawo).



Schemat logiczny układu sterowania kabiną suwnicy.

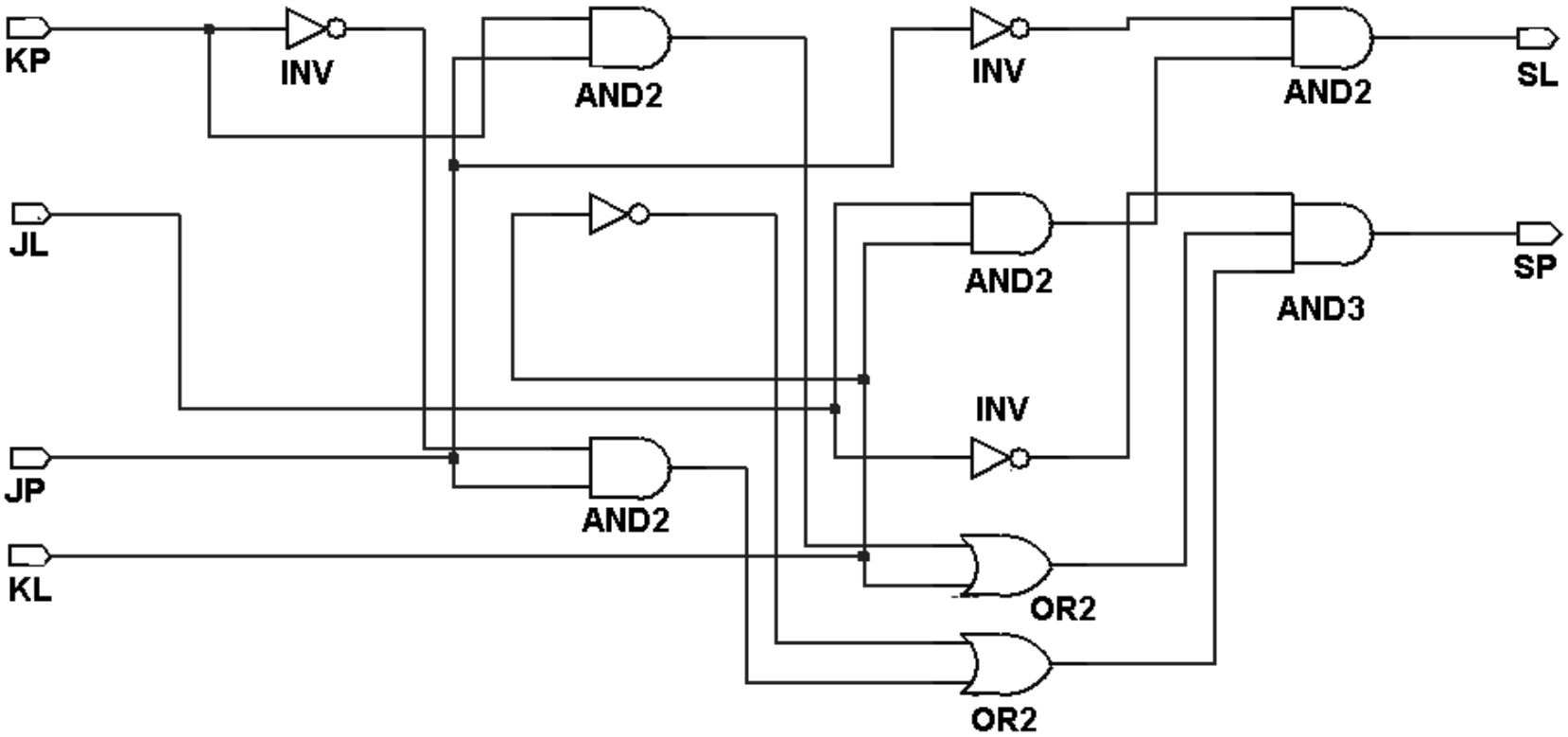
Przykład 2 – sterowanie suwnicą

Tabela prawdy układu sterowania

Wejścia				Wyjścia	
JP (z kabiny)	**JL (z kabiny)**	**KP (z czujnika)**	**KL (z czujnika)**	**SP (do silnika)**	**SL (do silnika)**
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	0	0
0	0	1	1	X	X
0	1	0	0	0	1
0	1	0	1	0	0
0	1	1	0	0	1
0	1	1	1	X	X
1	0	0	0	1	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	1	X	X
1	1	0	0	X	X
1	1	0	1	X	X
1	1	1	0	X	X
1	1	1	1	X	X

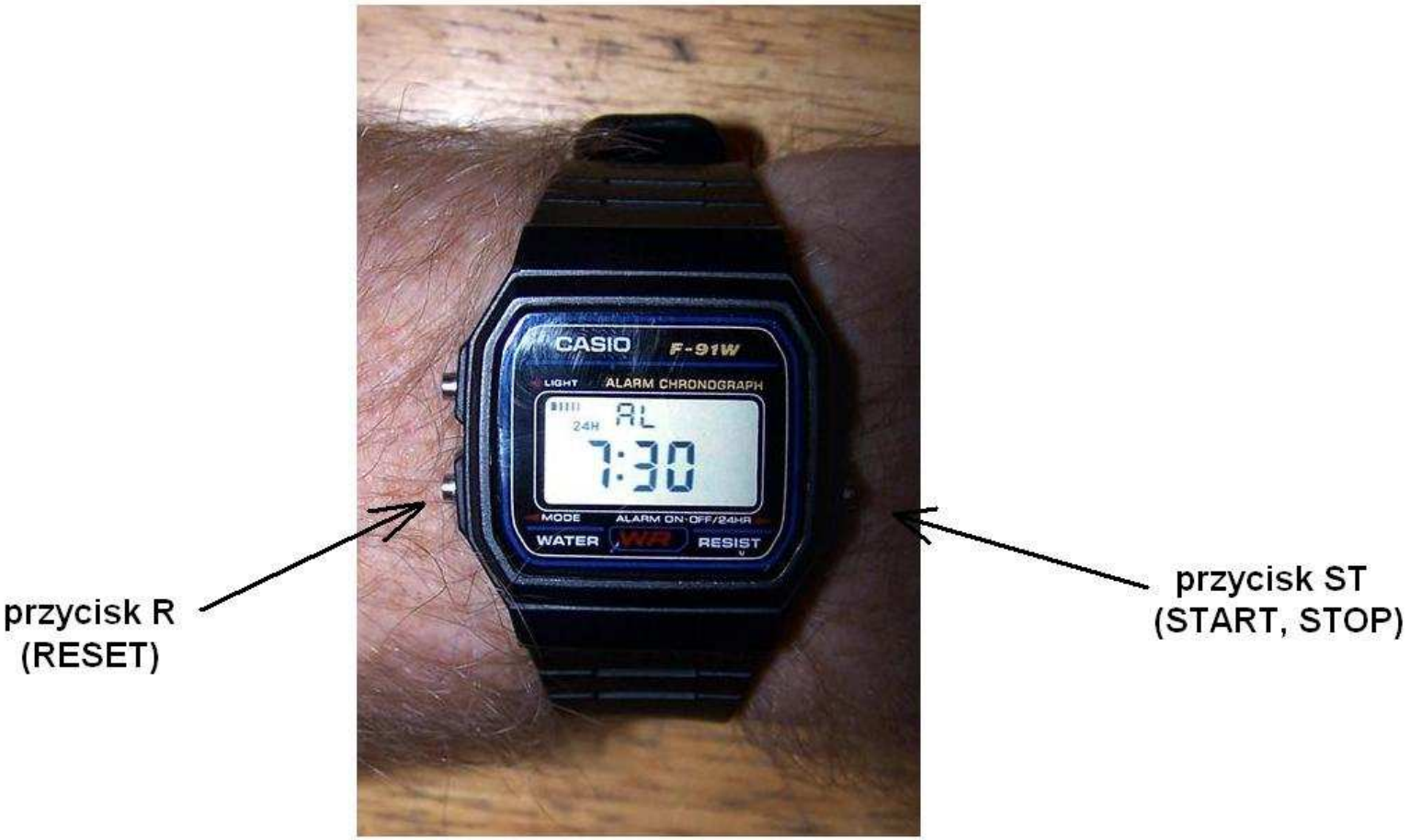
Przykład 2 – sterowanie suwnicą

- schemat układu sterowania



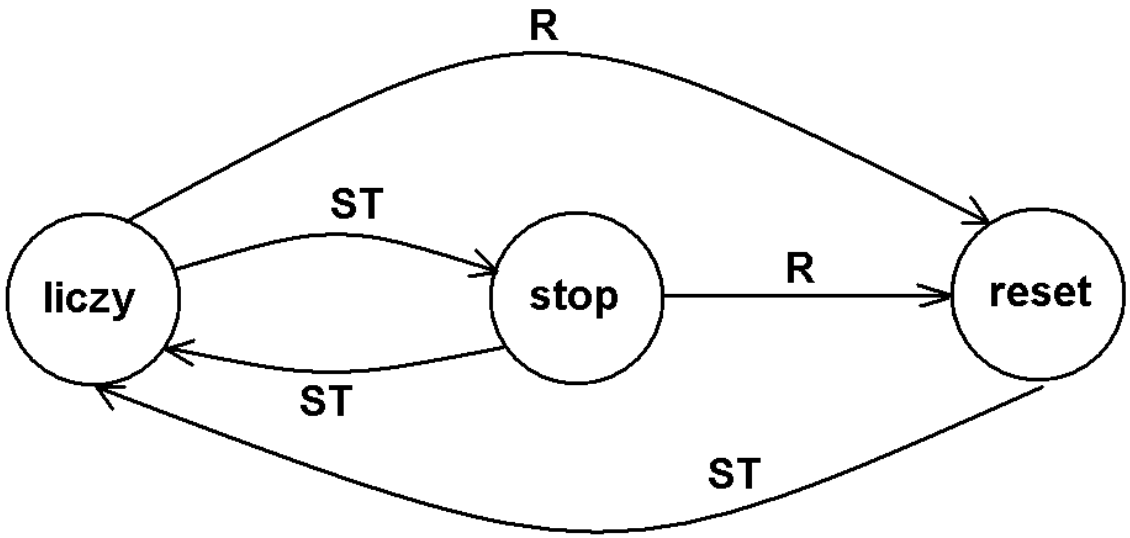
Schemat układu sterowania suwnicą.

Przykład 3 – zegar



Przykład 3 – zegar

- graf sygnałowy



Przykład 3 – zegar

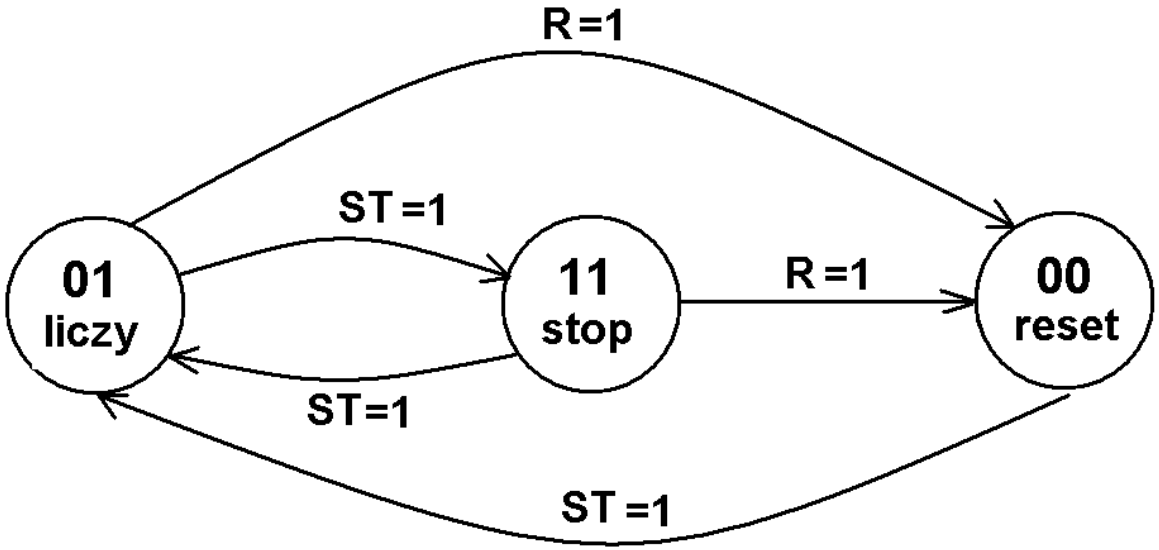
- graf sygnałowy

Zakodujmy stany układu w postaci cyfrowej:

Liczy – ”01”

Stop – ”11”

Reset – ”00”.



Działaniu przycisków przyporządkujmy cyfry:

Przycisk R wciśnięty oznacza ”1”, puszczony ”0”

Przycisk ST wciśnięty oznacza ”1”, puszczony ”0”.

Przykład 3 – zegar

- schemat logiczny