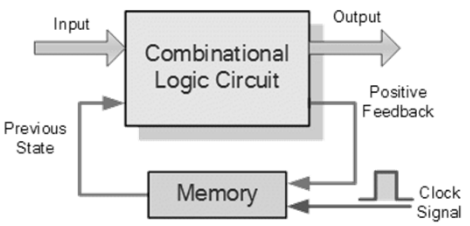
Układy sekwencyjne

Technika cyfrowa i mikroprocesorowa
dr inż. Dominik Łuczak

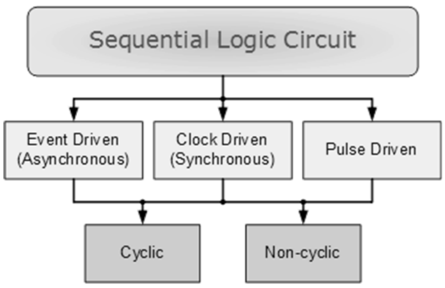
1

Sequential logic



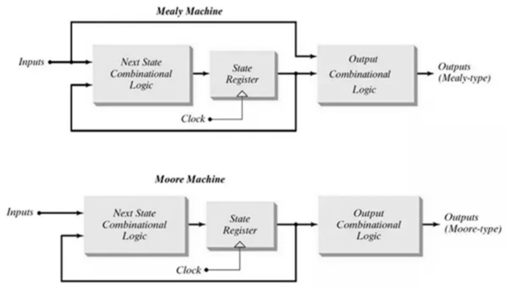
2

Poznan University of Technology
Institute of Control, Robotics and Information Engineering



3

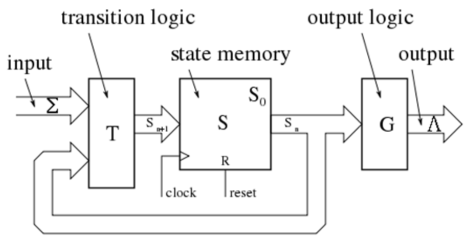
Poznan University of Technology
Institute of Control, Robotics and Information Engineering



2019-10-12 dominik.luczak@put.poznan.pl 4

4

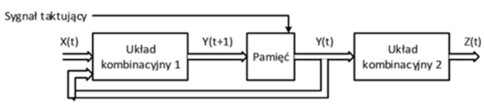
Poznan University of Technology
Institute of Control, Robotics and Information Engineering



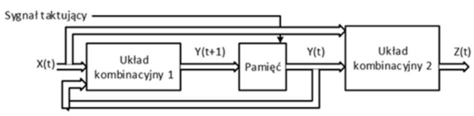
2019-10-12 dominik.luczak@put.poznan.pl 5

5

Poznan University of Technology
Institute of Control, Robotics and Information Engineering



Rys. 1 Automat Moor'a



Rys. 2 Automat Mealy'ego

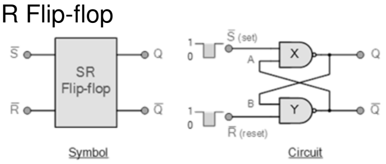
2019-10-12 dominik.luczak@put.poznan.pl 6

6

State memory

7

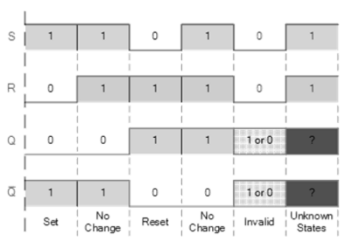
Basic SR Flip-flop



State	S	R	Q		Description
Set	1	0	0	1	Set $\overline{Q}$ » 1
	1	1	0	1	no change
Reset	0	1	1	0	Reset $\overline{Q}$ » 0
	1	1	1	0	no change
Invalid	0	0	1	1	Invalid Condition

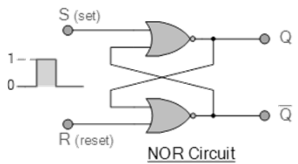
8

S-R Flip-flop Switching Diagram



9

The NOR Gate SR Flip-flop



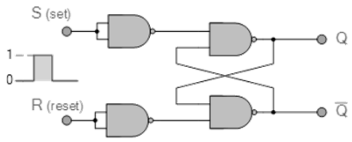
NOR Circuit

S	R	Q	$\overline{Q}$
0	0	No change	
0	1	1	0
1	0	0	1
1	1	X	X (Invalid)

10

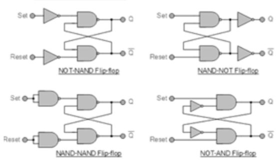
Poznan University of Technology
Institute of Control, Robotics and Information Engineering

Active HIGH Flip-flops



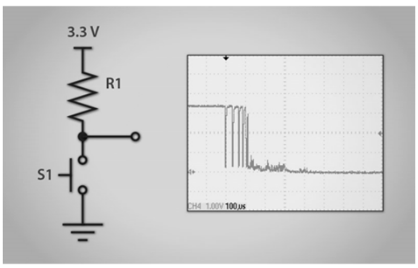
NAND Circuit

S	R	Q	$\overline{Q}$
0	0	No change	
0	1	0	1
1	0	1	0
1	1	X	X (Invalid)



11

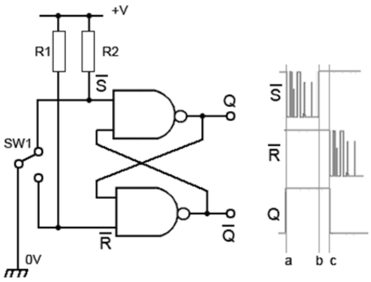
Poznan University of Technology
Institute of Control, Robotics and Information Engineering



12

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

SR Flip Flop Switch Debounce Circuit

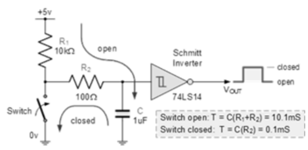


13

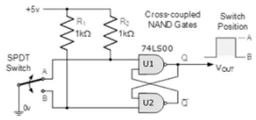
Poznan University of Technology
Institute of Control, Robotics and Information Engineering

Debounce Circuit

RC Switch Debounce Circuit



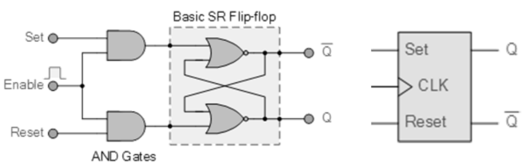
Switch Debounce with NAND Gates



14

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

Gated SR Flip-flop Circuit

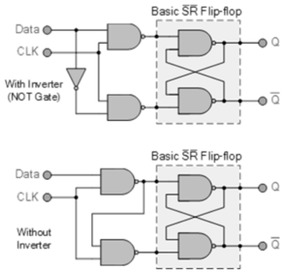


Tab. 3 Tabela przejść przerzutnika typu RS

S R	$Q^t \rightarrow Q^{t+1}$
0 0	$Q^t \rightarrow Q^{t+1}$
1 0	$Q^t \rightarrow 1$
0 1	$Q^t \rightarrow 0$
1 1	$Q^t \rightarrow$ -

15

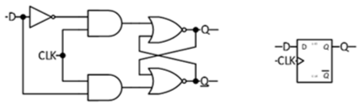
The D-type Flip-flop (NAND)



16

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

D (AND, NOR)



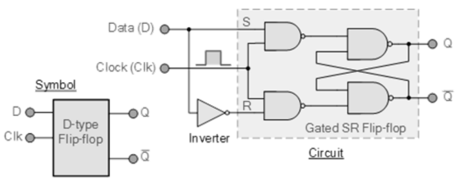
Rys. 4 Przerzutnik typu D

Tab. 4 Tabela przejść przerzutnika typu D

D	$Q^t \rightarrow Q^{t+1}$
0	$0 \rightarrow 0$
0	$1 \rightarrow 0$
1	$0 \rightarrow 1$
1	$1 \rightarrow 1$

17

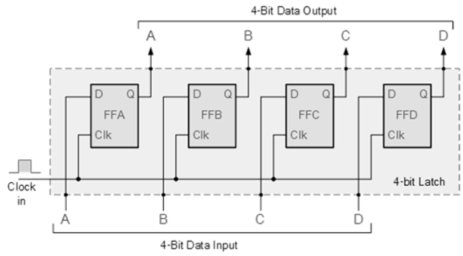
Poznan University of Technology
Institute of Control, Robotics and Information Engineering



18

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

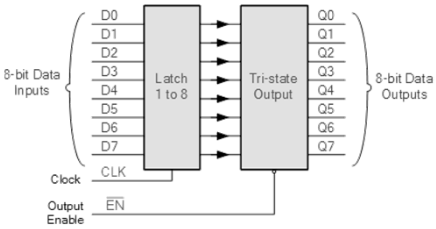
4-bit Data Latch



19

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

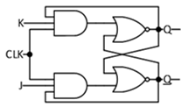
8-bit Data Latch

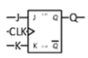


20

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

JK





Tab. 6 Skrócona tabela przejść przerzutnika typu JK

J K	$Q^t \rightarrow Q^{t+1}$
0 –	$0 \rightarrow 0$
1 –	$0 \rightarrow 1$
– 0	$1 \rightarrow 1$
– 1	$1 \rightarrow 0$

21

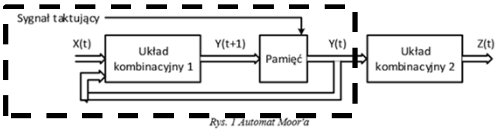
Przykład

22

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

Tab. 2 Tabela przejść automatu

S	σ =0	σ =1
0	0	3
1	4	3
2	2	5
3	0	4
4	1	2
5	1	5



Rys. 1 Automat Moor'a

23

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

Tab. 7 Realizowany automat

S	σ =0	σ =1	λ
1	1	2	0
2	2	3	1
3	3	2	1

Tab. 8 Kodowanie stanów automatu

S	s_1 s_2
0	0 0
1	0 1
2	1 0
3	1 1

Tab. 9 Realizowany automat z kodowaniem stanów

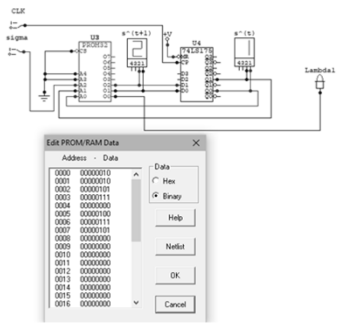
s_1 s_2	σ =0	σ =1	λ
0 0	- -	- -	-
0 1	0 1	1 0	0
1 0	1 0	1 1	1
1 1	1 1	1 0	1

Tab. 10 Zapis w pamięci stałej

Nr	A2 A1 A0	Q2 Q1 Q0
	σ s_1^t s_2^t	s_1^{t+1} s_2^{t+1} λ
0	0 0 0	- - -
1	0 0 1	0 1 0
2	0 1 0	1 0 1
3	0 1 1	1 1 1
4	1 0 0	- - -
5	1 0 1	1 0 0
6	1 1 0	1 1 1
7	1 1 1	1 0 1

24

Poznan University of Technology
Institute of Control, Robotics and Information Engineering



25

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

$$s_1^{t+1} = D_1 = s_1 + \sigma = \overline{\overline{s_1 + \sigma}} = \overline{\bar{s}_1 \cdot \bar{\sigma}} \qquad (2)$$

$$s_2^{t+1} = D_2 = s_2 \cdot \bar{\sigma} + \bar{s}_2 \cdot \sigma = \overline{\overline{s_2 \cdot \bar{\sigma} + \bar{s}_2 \cdot \sigma}} = \overline{\overline{s_2 \cdot \bar{\sigma}} \cdot \overline{\bar{s}_2 \cdot \sigma}} \qquad (3)$$

$$\lambda = s_1 \qquad (4)$$

Tab. 11 Realizowany automat z kodowaniem stanów - tablica Karnaugh

$s_1 s_2$	σ =0	σ =1	λ
0 0	- -	- -	-
0 1	0 1	1 0	0
1 1	1 1	1 0	1
1 0	1 0	1 1	1
	$D_1 D_2$	$D_1 D_2$	

26

Poznan University of Technology
Institute of Control, Robotics and Information Engineering

Tab. 12 Pomocnicza tablica Karnaugh - D1

Tablica pomocnicza D1

$s_1 s_2$	σ =0	σ =1	λ
0 0	-	-	-
0 1	0	1	0
1 1	1	1	1
1 0	1	1	1
	D_1	D_1	

Pierwszy fragment

$s_1 s_2$	σ =0	σ =1	λ
0 0	-	-	-
0 1	0	1	0
1 1	1	1	1
1 0	1	1	1
	D_1	D_1	

Drugi fragment

$s_1 s_2$	σ =0	σ =1	λ
0 0	-	-	-
0 1	0	1	0
1 1	1	1	1
1 0	1	1	1
	D_1	D_1	

Tab. 13 Pomocnicza tablica Karnaugh - D2

Tablica pomocnicza D2

$s_1 s_2$	σ =0	σ =1	λ
0 0	-	-	-
0 1	1	0	0
1 1	1	0	1
1 0	0	1	1
	D_2	D_2	

Pierwszy fragment

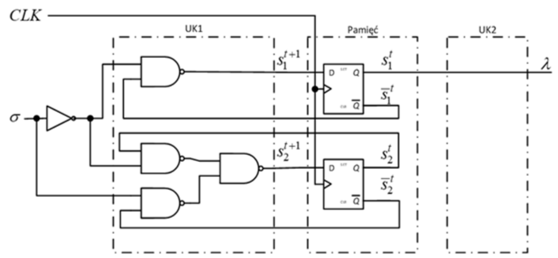
$s_1 s_2$	σ =0	σ =1	λ
0 0	-	-	-
0 1	1	0	0
1 1	1	0	1
1 0	0	1	1
	D_2	D_2	

Drugi fragment

$s_1 s_2$	σ =0	σ =1	λ
0 0	-	-	-
0 1	1	0	0
1 1	1	0	1
1 0	0	1	1
	D_2	D_2	

27

Poznan University of Technology
Institute of Control, Robotics and Information Engineering



Rys. 8 Realizacja automatu z wykorzystaniem przerzutnika typu D

28

https://www.onlinegdb.com/online_c_compiler

main.c

```c
#include <stdio.h>
#include <stdint.h>

uint8_t Q[] = {0,1,2,3,0,2,3,2};
uint8_t L[] = {0,0,1,1,0,0,1,1};
uint8_t S1=0,S2=0,sigma=0, address=0, y=0,S=0, nr=0;

int main()
{
//Combinational
for(sigma=0;sigma<=1;sigma++)
  for(S1=0;S1<=1;S1++)
    for(S2=0;S2<=1;S2++)
        {
          address = sigma<<2 | S1<<1 | S2;
          S=Q[address];
          y=L[address];
          printf("S=%d, y=%d\tNr:%d\t(Sigma,S1,S2 -> %d,%d,%d)\n",S,y,nr,sigma,S1,S2);
          nr++;
        }
//Sequential
S=1;sigma=1;
for(uint8_t clk=0;clk<10;clk++){
    address = sigma<<2 | S;
    S=Q[address];
    y=L[address];
    printf("S=%d, y=%d\tClk:%d\t(Sigma -> %d)\n",S,y,clk,sigma);
    if(clk>5)sigma=0;
}

    return 0;
}
```

```
S=0, y=0        Nr:0    (Sigma,S1,S2 -> 0,0,0)
S=1, y=0        Nr:1    (Sigma,S1,S2 -> 0,0,1)
S=2, y=1        Nr:2    (Sigma,S1,S2 -> 0,1,0)
S=3, y=1        Nr:3    (Sigma,S1,S2 -> 0,1,1)
S=0, y=0        Nr:4    (Sigma,S1,S2 -> 1,0,0)
S=2, y=0        Nr:5    (Sigma,S1,S2 -> 1,0,1)
S=3, y=1        Nr:6    (Sigma,S1,S2 -> 1,1,0)
S=2, y=1        Nr:7    (Sigma,S1,S2 -> 1,1,1)
S=2, y=0        Clk:0   (Sigma -> 1)
S=3, y=1        Clk:1   (Sigma -> 1)
S=2, y=1        Clk:2   (Sigma -> 1)
S=3, y=1        Clk:3   (Sigma -> 1)
S=2, y=1        Clk:4   (Sigma -> 1)
S=3, y=1        Clk:5   (Sigma -> 1)
S=2, y=1        Clk:6   (Sigma -> 1)
S=2, y=1        Clk:7   (Sigma -> 0)
S=2, y=1        Clk:8   (Sigma -> 0)
S=2, y=1        Clk:9   (Sigma -> 0)


...Program finished with exit code 0
Press ENTER to exit console.
```


29