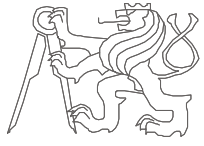
Logické obvody

Kombinační a sekvenční stavební bloky

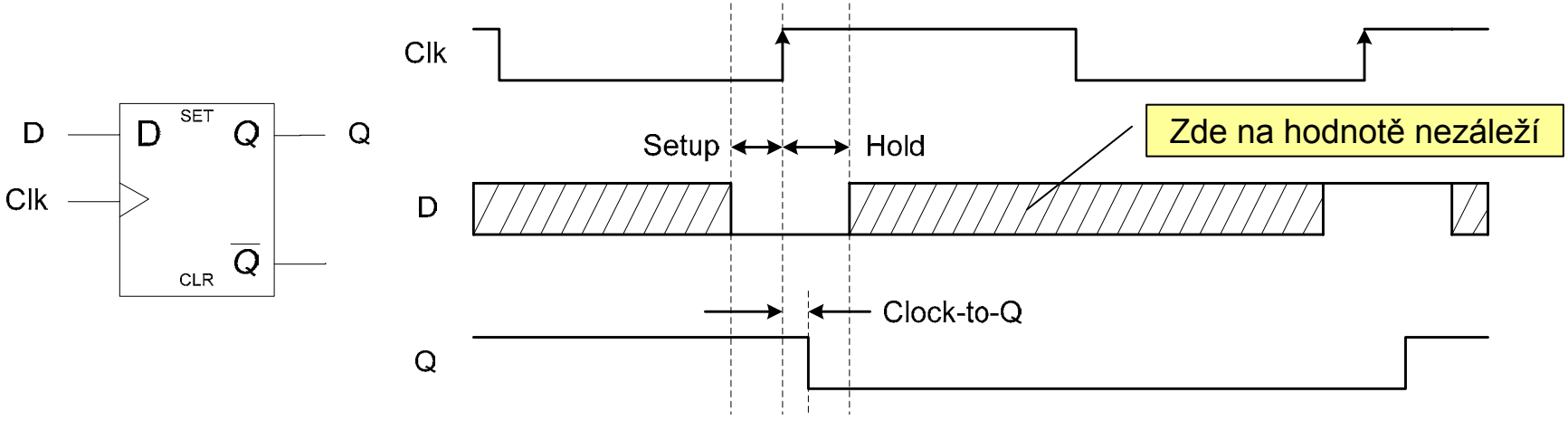


České vysoké učení technické Fakulta elektrotechnická

Časování – výpočet maximální hodinové frekvence

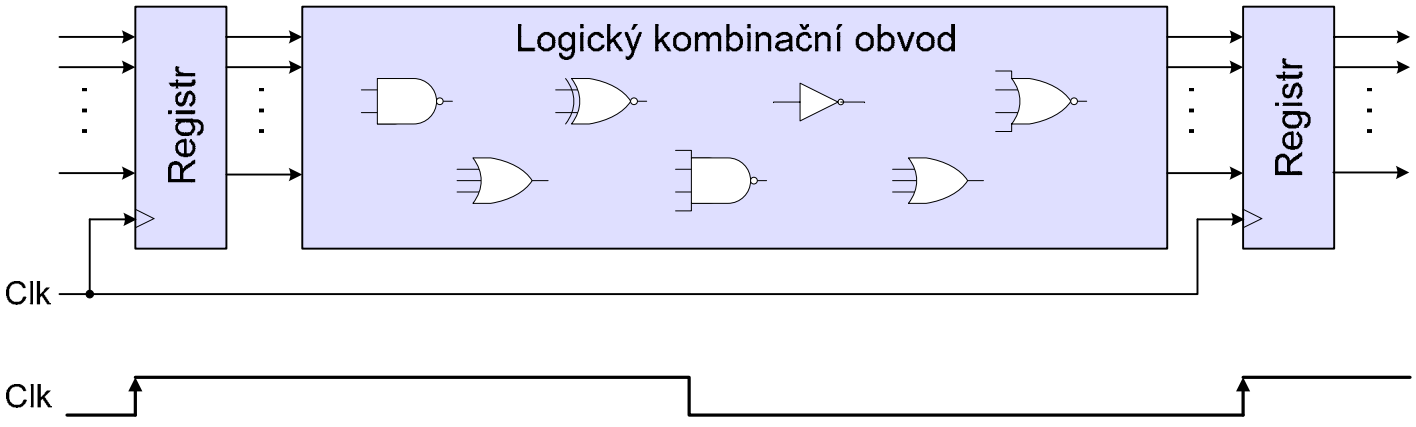
- Ovlivněno:
  - Technologií
  - Typy hradel
  - Počtem vstupů u hradel
  - Zatížením výstupů hradel (větvením)
  - Typem klopných obvodů
  - Délkou propojovacích vodičů (na plošném spoji,…)
  - Vzájemnou polohou vodičů (kvalita návrhu plošného spoje)
  - Rozmístěním součástek
  - Počtem zemnících a napájecích vrstev
  - Způsobem rozvodu napájení
  - Rozmístěním blokovacích kondezátorů
  - Dalšími vlivy …..

Časování klopného obvodu



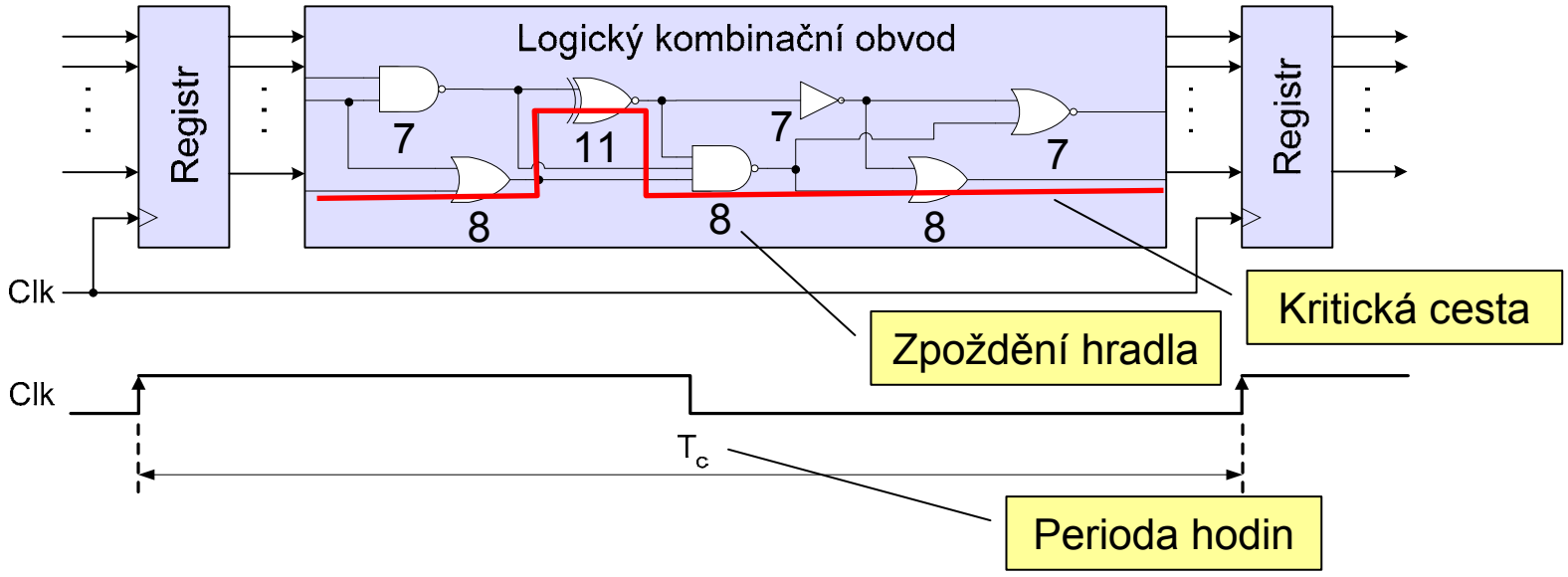
- Předstih (Setup Time) – Vstup D musí být stabilní (ustálený) před aktivní (zde náběžnou) hranou hodinového signálu
- Přesah (Hold Time) – Vstup D musí zůstat stabilní (ustálený) po aktivní (zde náběžné) hraně hodinového signálu
- Zpoždění (Clock-to-Q Time) výstupu Q po aktivní (zde náběžné) hraně hodinového signálu

Maximální hodinová frekvence



- Všechny klopné obvody jsou řízeny stejným hodinovým signálem
- Kombinační logické bloky:
  - Vstupy jsou aktualizovány při každém taktu hodin
  - Všechny výstupy musí být stabilní před dalším taktem

Kritická cesta a perioda hodin

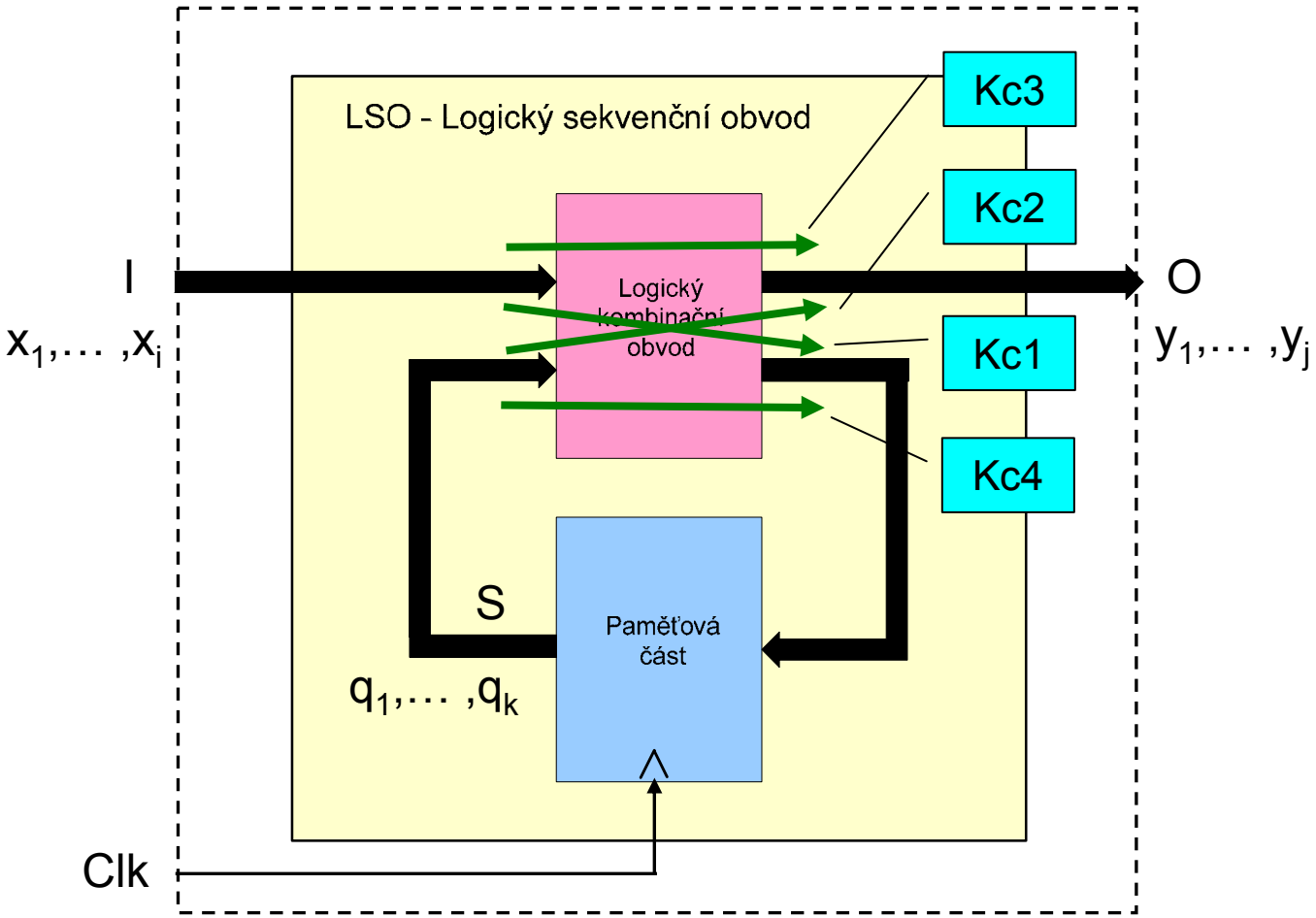


- Kritická cesta: nejpomalejší cesta mezi libovolným z registrů (klop.obvodů)
- Minimální perioda hodin je funkcí kritické cesty
- Perioda T_{cmin} musí být větší než:

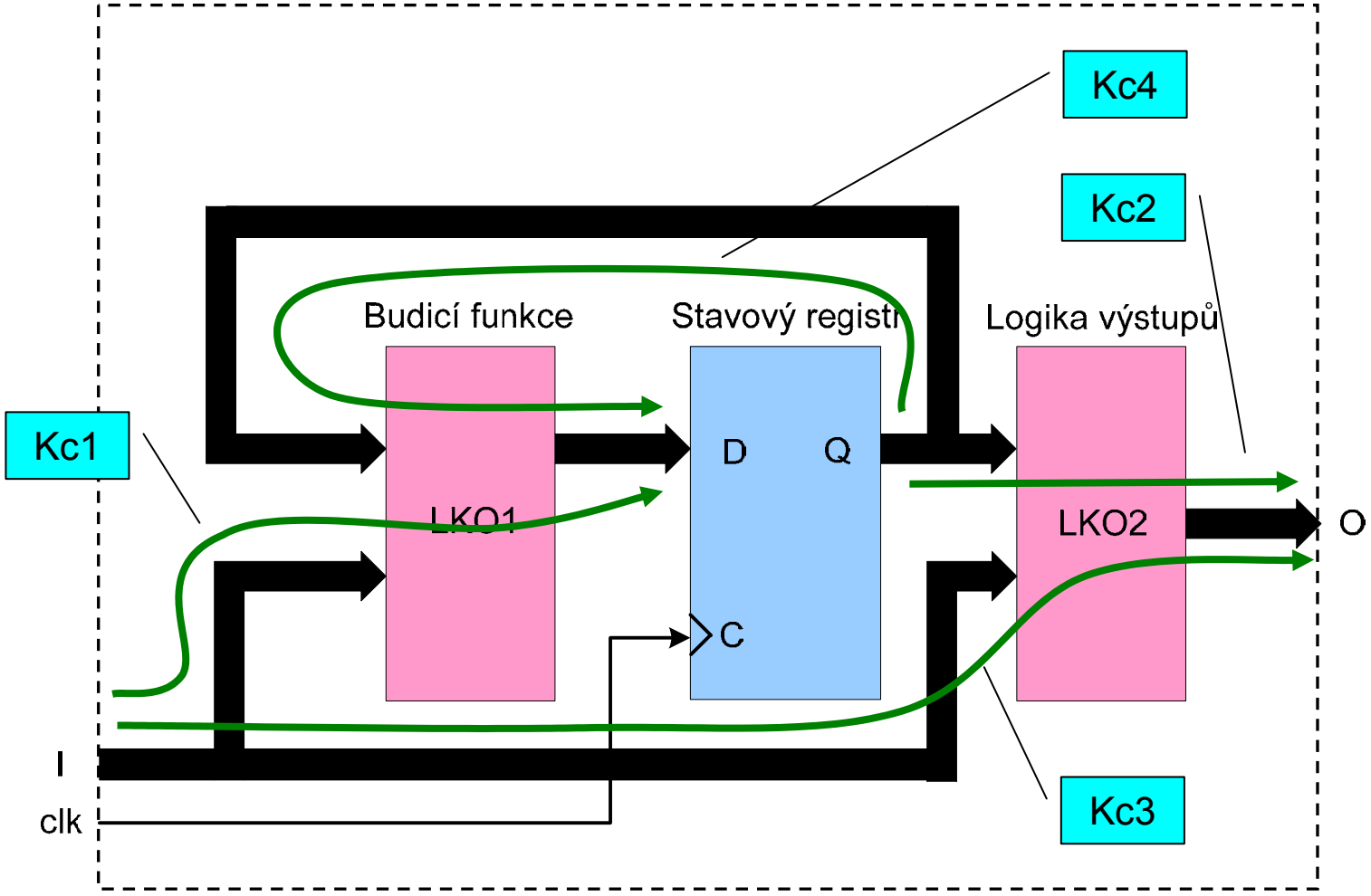
$$T_{c\min} \geq Clock - to - Q + „Nejpomalejsi\ cesta\ kombinacni\ casti“ + Setup$$

- Musí být splněny požadavky na stabilitu vstupů a výstupů

Kritická cesta (Kc)



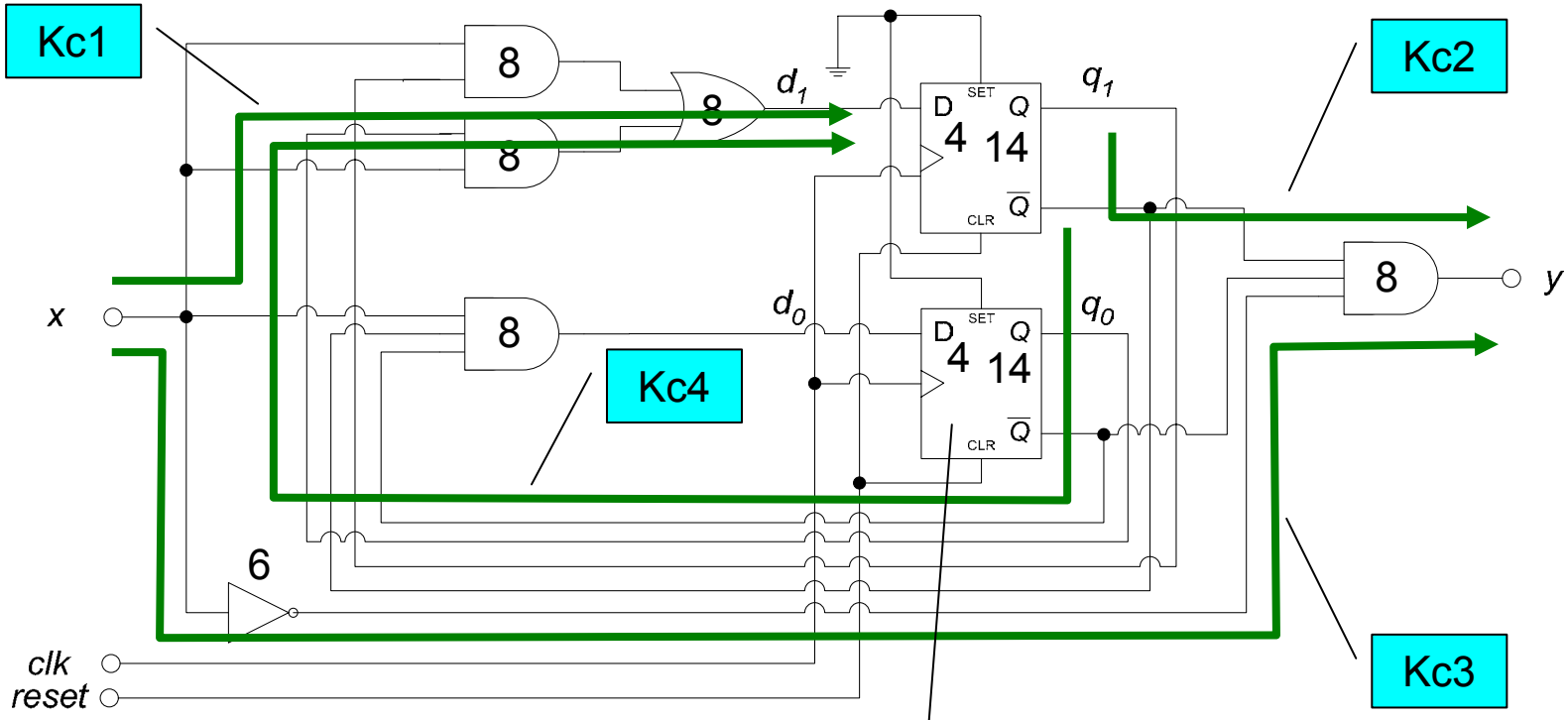
Kritická cesta (Kc)



Hodinová frekvence = $1/Kc_{i\,max}$

- Kc1 – ze vstupů I na vstup stavového registru
  - Nestabilita vstupů + zpoždění v LKO1 + Setup (předstih)
- Kc2 – z výstupu stavového registru na výstup O
  - Clock-to-Q + zpoždění v LKO2 + požadavek na stabilitu výstupů
- Kc3 – ze vstupů I na výstupy O
  - Nestabilita vstupů + zpoždění v LKO2 + požadavek na stabilitu výstupů
- Kc4 – z výstupu stavového registru na jeho vstupy
  - Clock-to-Q + zpoždění v LKO1 + Setup (předstih)

Detektor posloupnosti bitů ‘110’ (FSA typu Mealy)

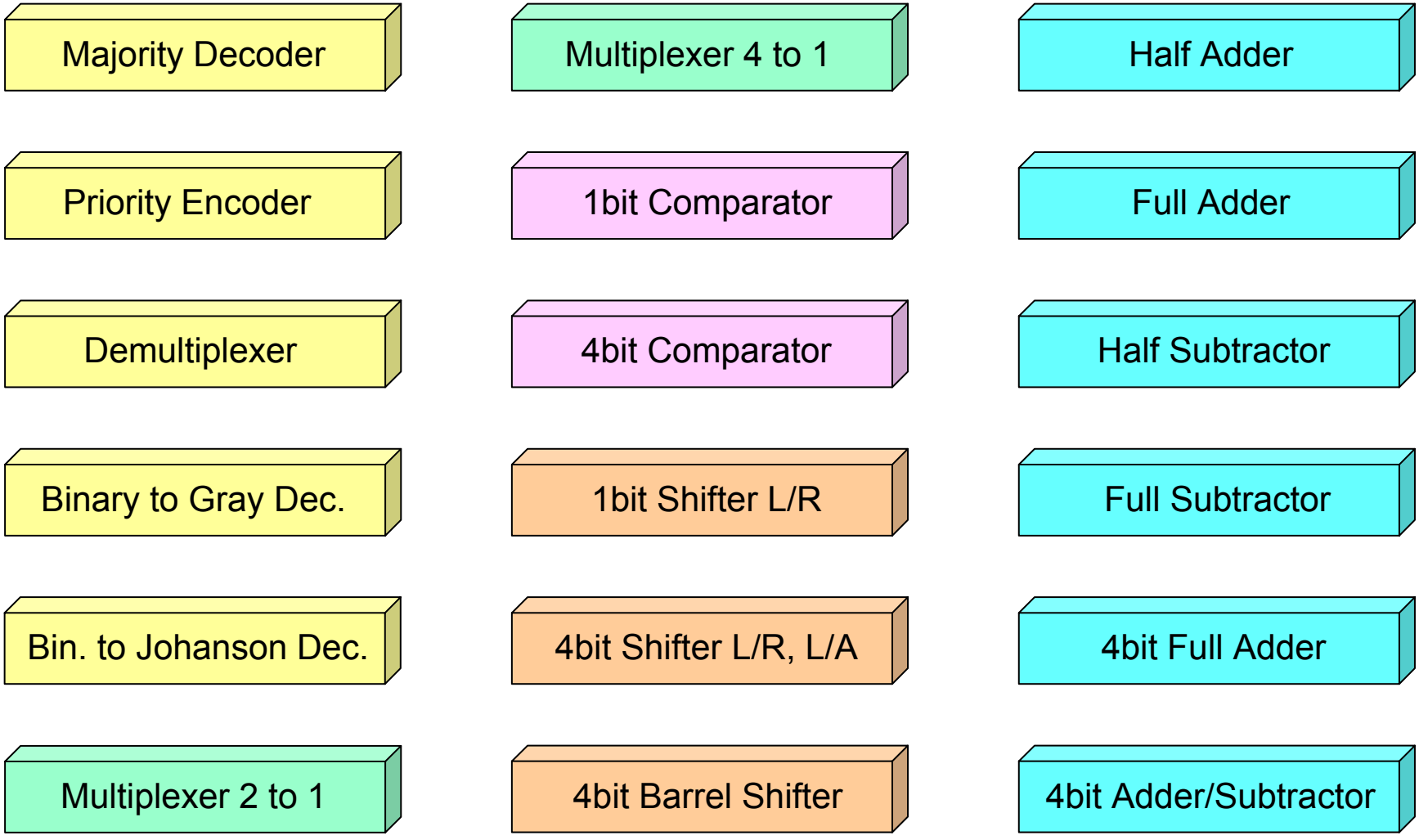


- Kc1 = 8 + 8 + 4 [ns]
- Kc2 = 14 + 8 [ns]
- Kc3 = 6 + 8 [ns]
- Kc4 = 14 + 8 + 8 + 4 [ns]

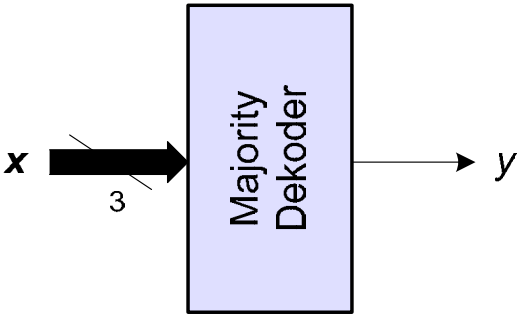
$$\begin{cases} TClock-to-Q = 14\,ns \\ Setup = 4\,ns \end{cases}$$

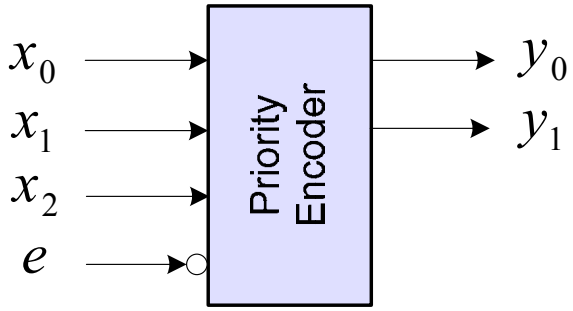
$$\Rightarrow \quad f_{\max} = 1/T_{c\min} = 1/34\,ns = 29{,}4\,MHz$$

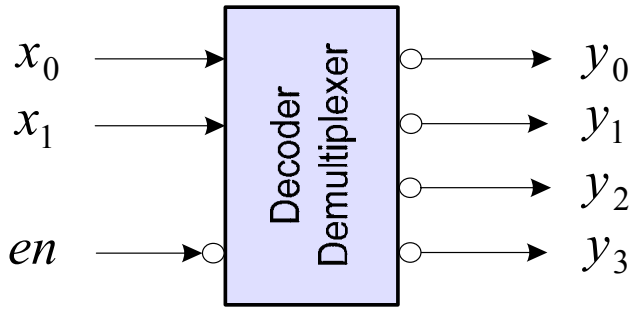
Přehled kombinačních bloků dle kategorie

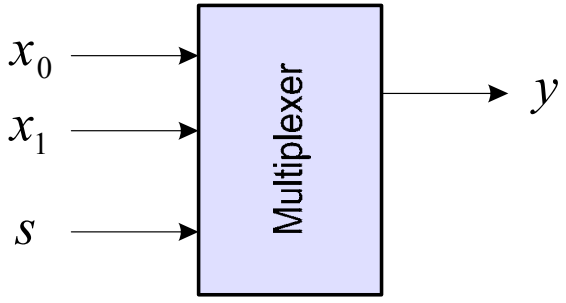


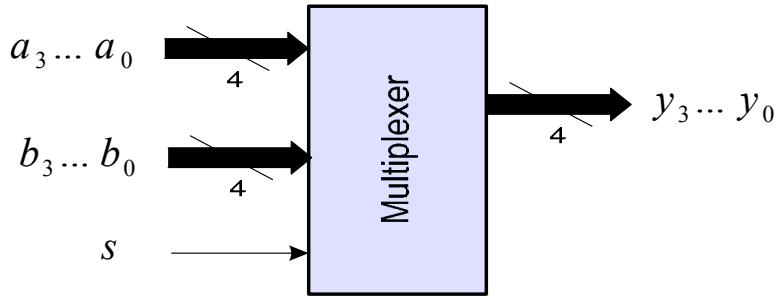
Přehled kombinačních bloků

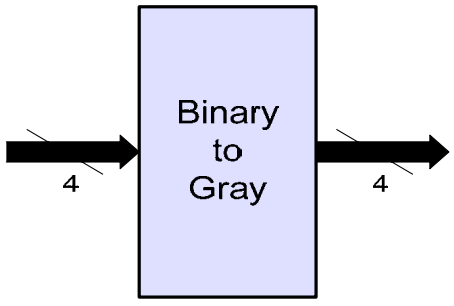


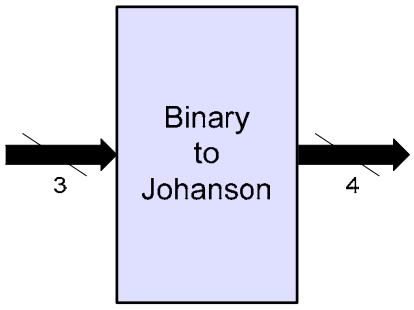




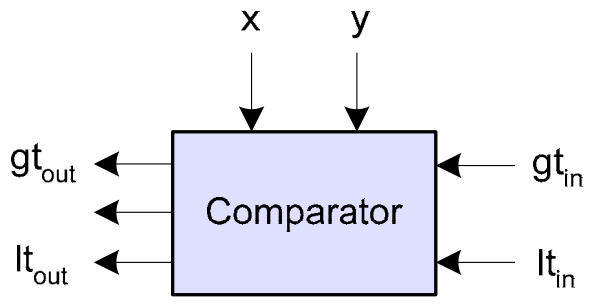


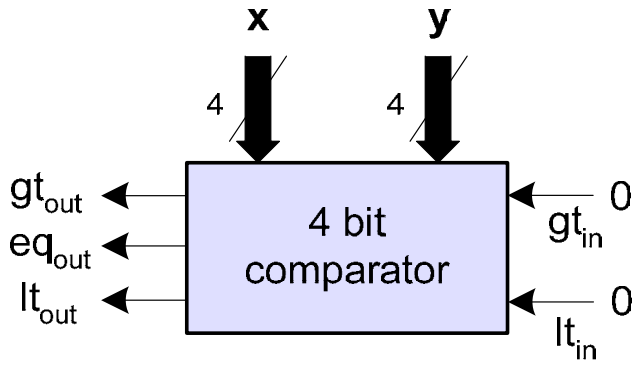


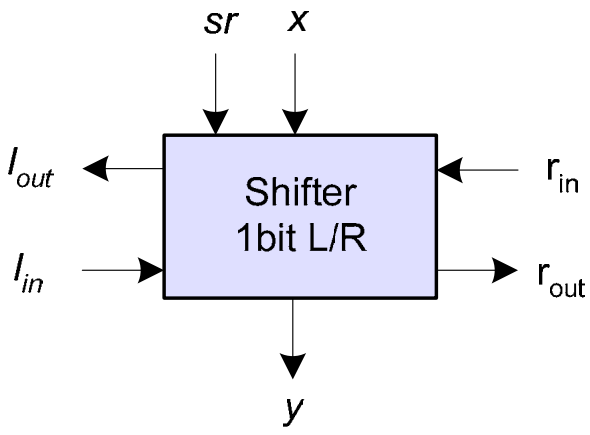


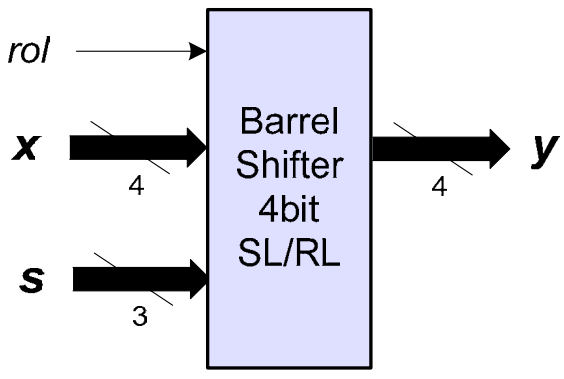


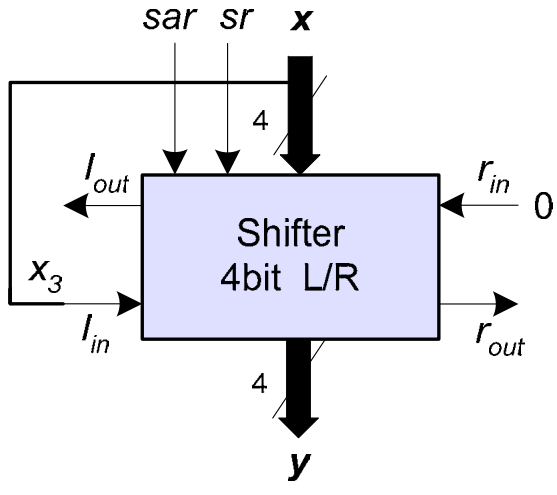
Přehled kombinačních bloků



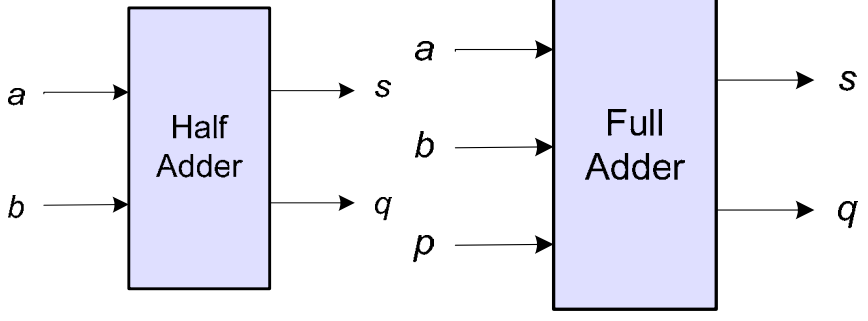


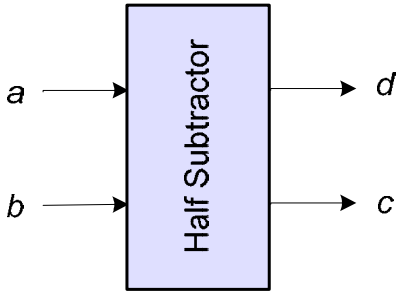


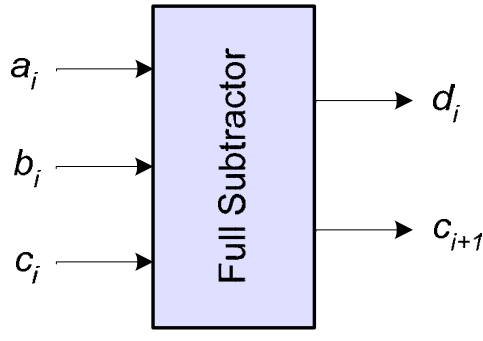


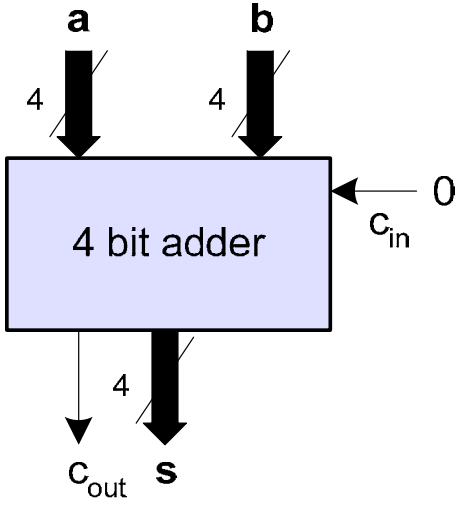


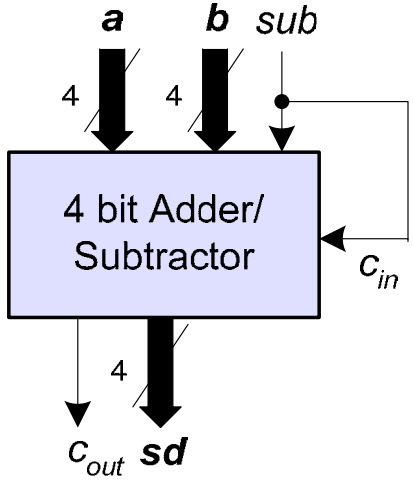
Přehled kombinačních bloků





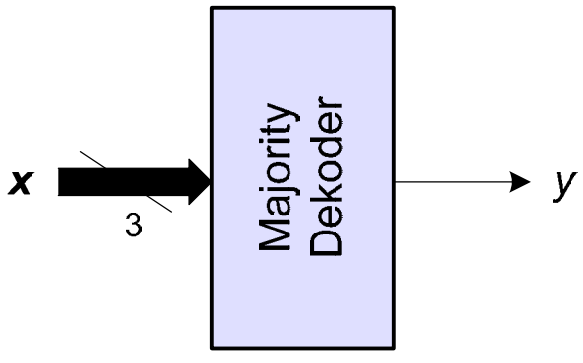






Majoritní dekodér

- Majorita
  - Nabývá hodnoty 1, když většina vstupních proměnných je rovna 1
  - Majorita ze 3 – tj. 2 nebo 3 vstupní proměnné mají hodnotu 1

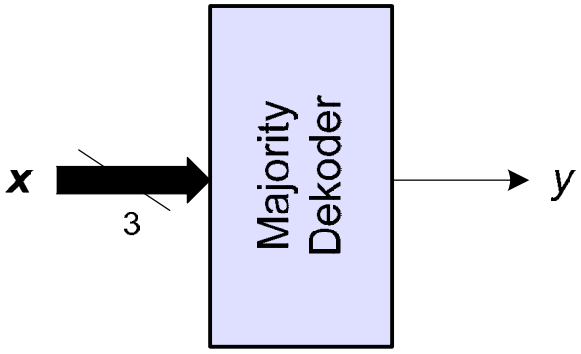


Majorita ze 3

D	x_2	x_1	x_0	y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

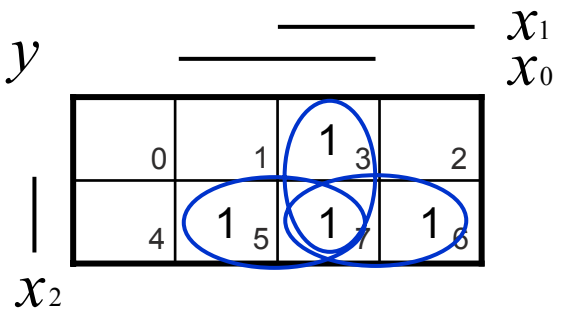
$$y = \sum \mathrm{m}(3,5,6,7) = \overline{x_2}\, x_1\, x_0 + x_2\, \overline{x_1}\, x_0 + x_2\, x_1\, \overline{x_0} + x_2\, x_1\, x_0$$

Majoritní dekodér



Majorita ze 3

D	x_2	x_1	x_0	y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

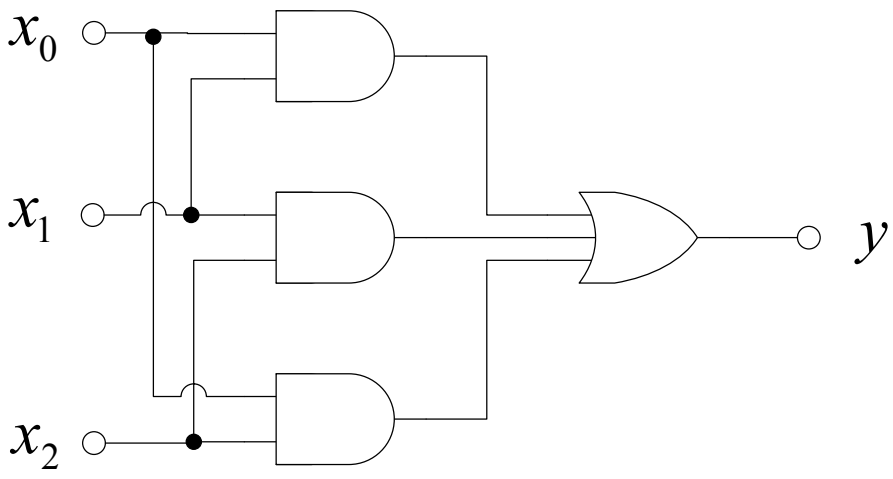


$$y = x_1 x_0 + x_2 x_1 + x_2 x_0$$

Majoritní dekodér

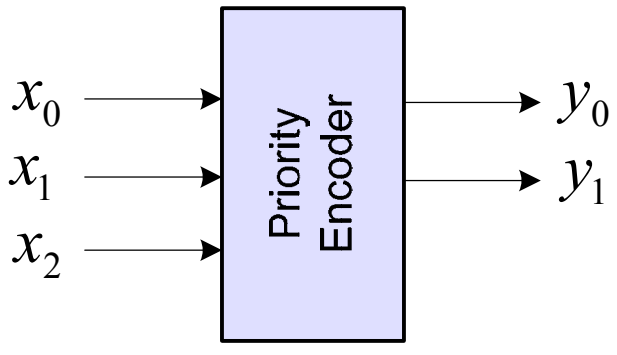
Realizace

$$y = x_1 x_0 + x_2 x_1 + x_2 x_0$$



Prioritní enkodér

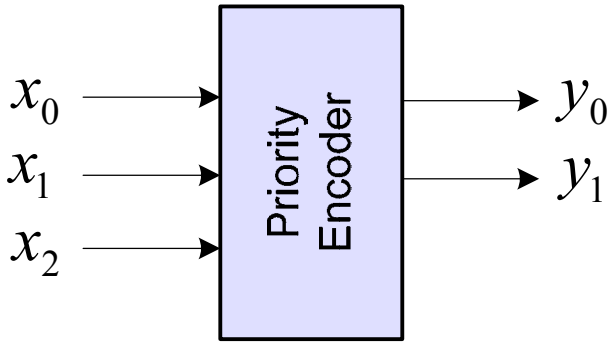
- Kóduje stav n vstupů do určeného kódu (např. binárního) na výstupu
- Libovolný počet vstupů x_i prioritního enkodéru může současně nabývat hodnoty 1. Prioritní enkodér na výstupech y_k vyšle vždy pouze kód aktivního vstupu (tj. $x_i = 1$) s nejvyšší prioritou (zde nejvyšší prioritu má x_0)
- Použití – systém přerušení v počítači (interrupt system), …



Prioritní enkodér

D_i	x_2	x_1	x_0	y_1	y_0	D_o
0	0	0	0	0	0	0
1	0	0	1	0	1	1
2	0	1	0	1	0	2
3	0	1	1	0	1	1
4	1	0	0	1	1	3
5	1	0	1	0	1	1
6	1	1	0	1	0	2
7	1	1	1	0	1	1

Prioritní enkodér



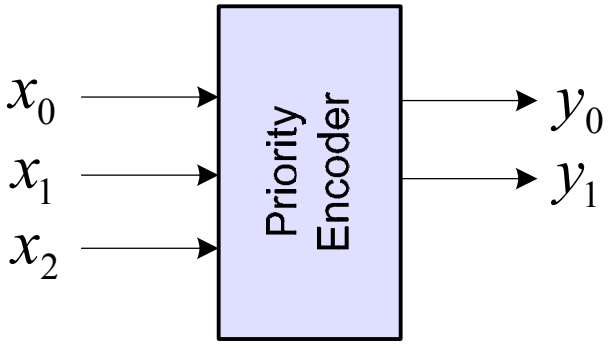
Prioritní enkodér

D_i	x_2	x_1	x_0	y_1	y_0	D_o
0	0	0	0	0	0	0
1	0	0	1	0	1	1
2	0	1	0	1	0	2
3	0	1	1	0	1	1
4	1	0	0	1	1	3
5	1	0	1	0	1	1
6	1	1	0	1	0	2
7	1	1	1	0	1	1

$$y_0 = \sum \mathrm{m}(1,3,4,5,7) = \overline{x_2}\,\overline{x_1}\,x_0 + \overline{x_2}\,x_1\,x_0 + x_2\,\overline{x_1}\,\overline{x_0} + x_2\,\overline{x_1}\,x_0 + x_2\,x_1\,x_0$$

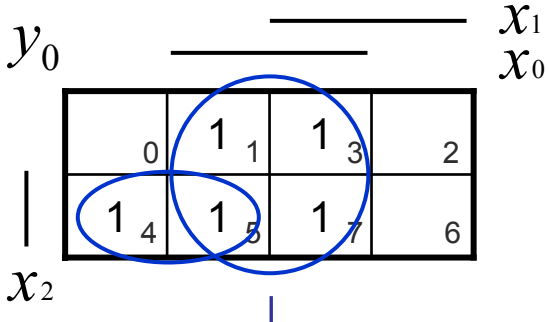
$$y_1 = \sum \mathrm{m}(2,4,6) = \overline{x_2}\,x_1\,\overline{x_0} + x_2\,\overline{x_1}\,\overline{x_0} + x_2\,x_1\,\overline{x_0}$$

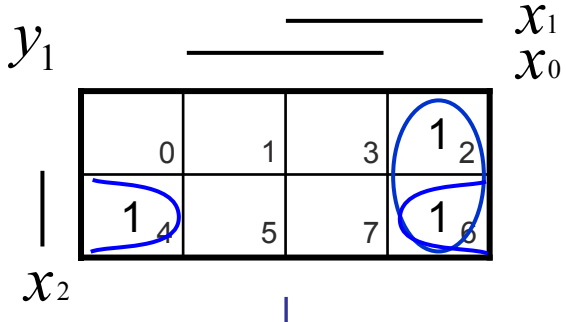
Prioritní enkodér



Prioritní enkodér

D_i	x_2	x_1	x_0	y_1	y_0	D_o
0	0	0	0	0	0	0
1	0	0	1	0	1	1
2	0	1	0	1	0	2
3	0	1	1	0	1	1
4	1	0	0	1	1	3
5	1	0	1	0	1	1
6	1	1	0	1	0	2
7	1	1	1	0	1	1



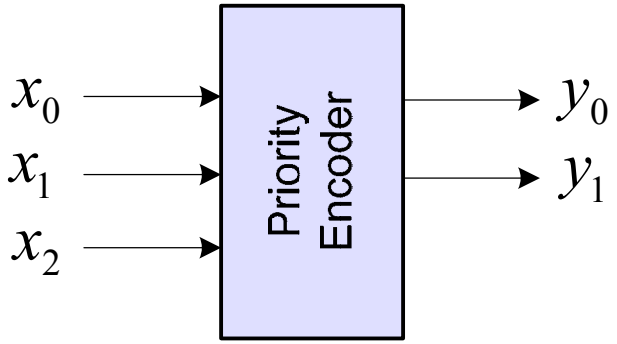


$$y_0 = x_0 + x_2 \overline{x_1}$$

$$y_1 = x_2 \overline{x_0} + x_1 \overline{x_0}$$

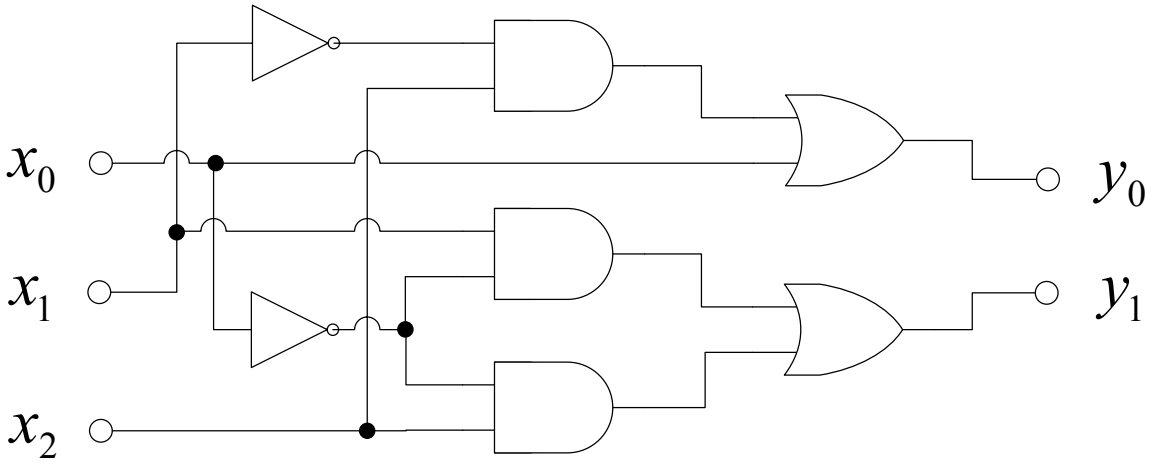
Prioritní enkodér

Realizace

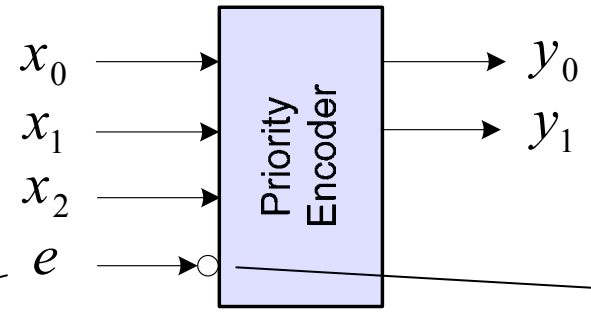


$$y_0 = x_0 + x_2 \overline{x_1}$$

$$y_1 = x_2 \overline{x_0} + x_1 \overline{x_0}$$



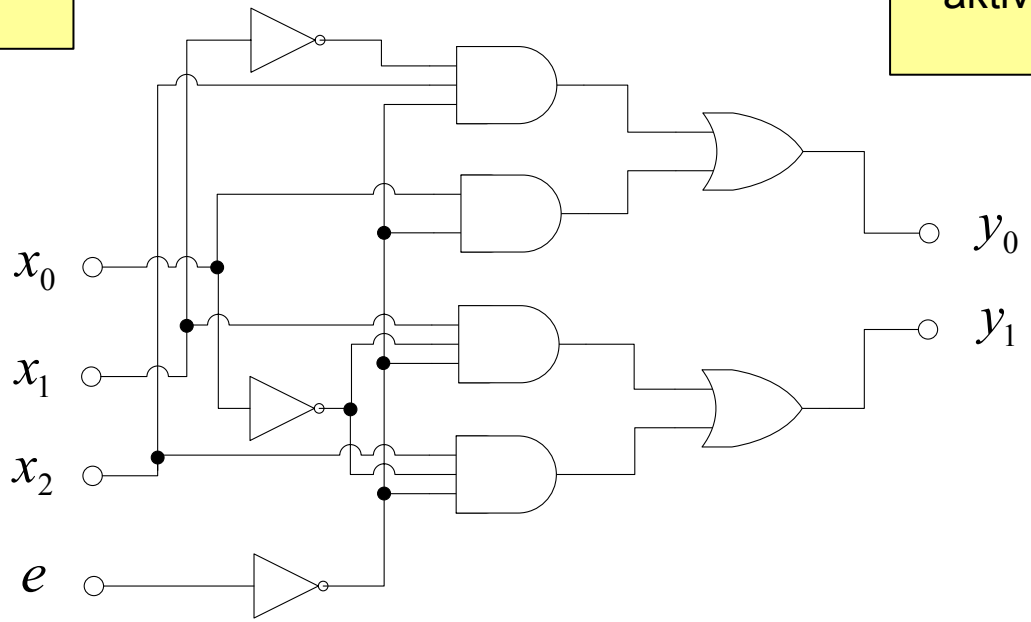
Prioritní enkodér



Realizace

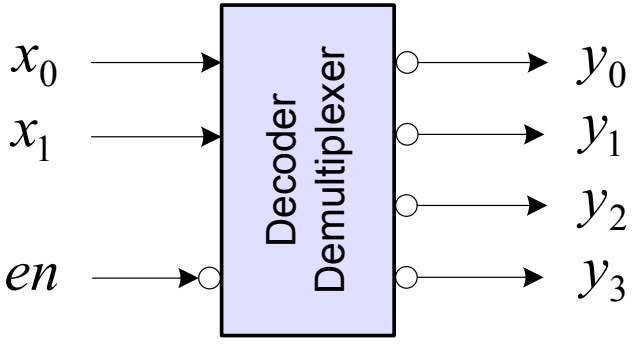
Blokování výstupu
e = 1

Kroužek značí, že aktivní stav je pro
e = 0



Dekodér/Demultiplexer

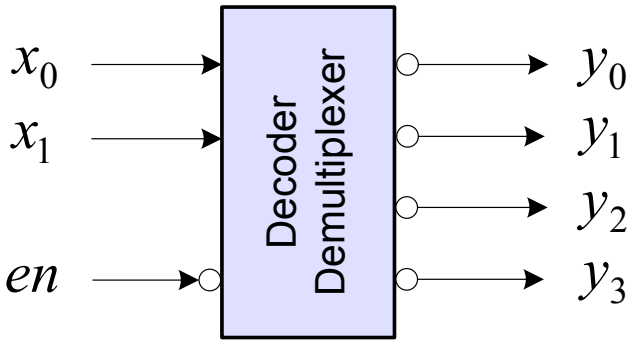
- Dekóduje kód na vstupu (např. binární) na kód 1 z n na výstupu
- Typicky je výstupní kód aktivní v 0
- Použití – dekodér adresových bloků v počítači, …



Dekodér/Demultiplexer

D_i	x_1	x_0	Y_3	y_2	y_1	y_0
0	0	0	0	0	0	1
1	0	1	0	0	1	0
2	1	0	0	1	0	0
3	1	1	1	0	0	0

Dekodér/Demultiplexer



Dekodér/Demultiplexer

D_i	x_1	x_0	Y_3	y_2	y_1	y_0
0	0	0	0	0	0	1
1	0	1	0	0	1	0
2	1	0	0	1	0	0
3	1	1	1	0	0	0

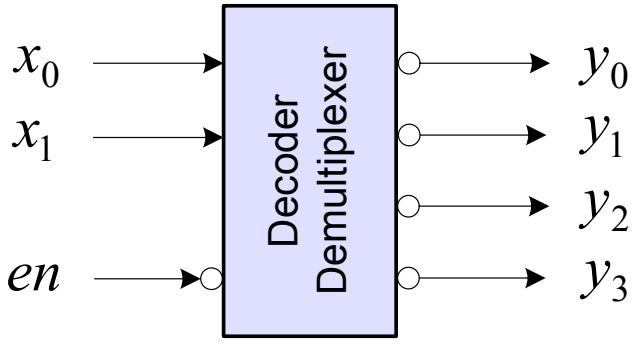
$$\overline{y_0} = \overline{x_1}\,\overline{x_0}$$

$$\overline{y_1} = \overline{x_1}\,x_0$$

$$\overline{y_2} = x_1\,\overline{x_0}$$

$$\overline{y_3} = x_1\,x_0$$

Dekodér/Demultiplexer



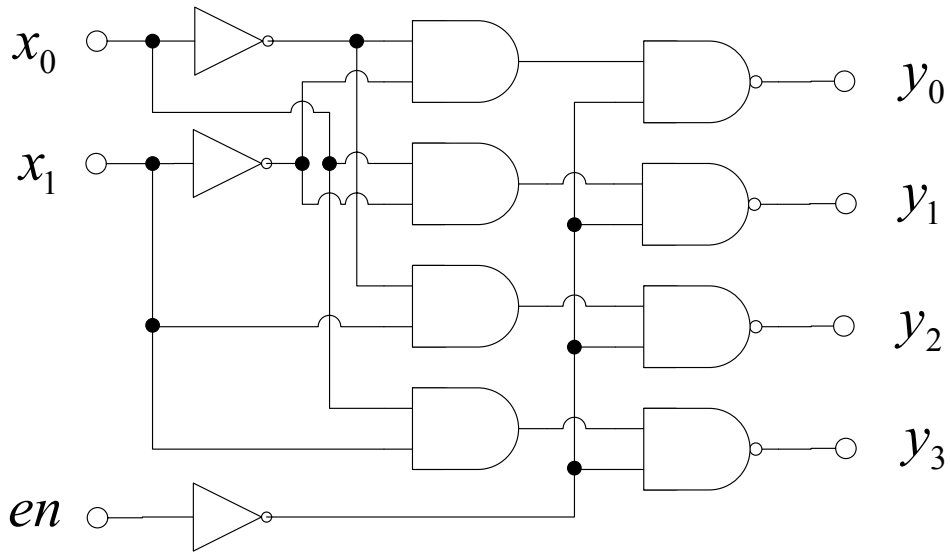
Realizace

$$\overline{y_0} = \overline{\overline{x_1}\,\overline{x_0}}$$

$$\overline{y_1} = \overline{x_1\,\overline{x_0}}$$

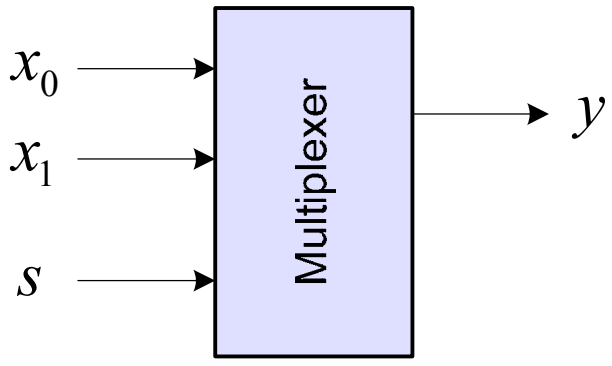
$$\overline{y_2} = \overline{\overline{x_1}\,x_0}$$

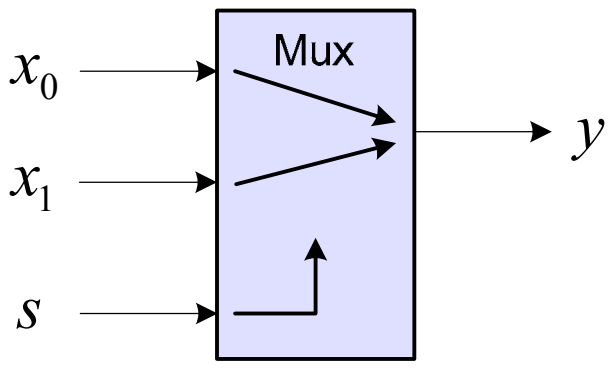
$$\overline{y_3} = \overline{x_1\,x_0}$$



Multiplexer

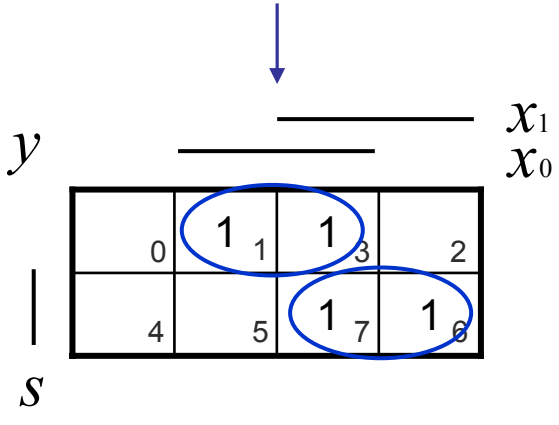
- n – vstupový multiplexer je číslicový přepínač n – vstupů na jeden výstup





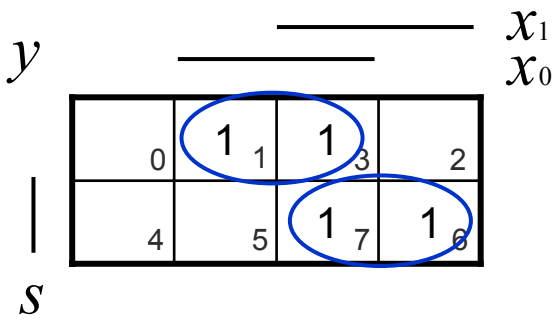
Multiplexer 2 na 1

D_i	s	x_1	x_0	y
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

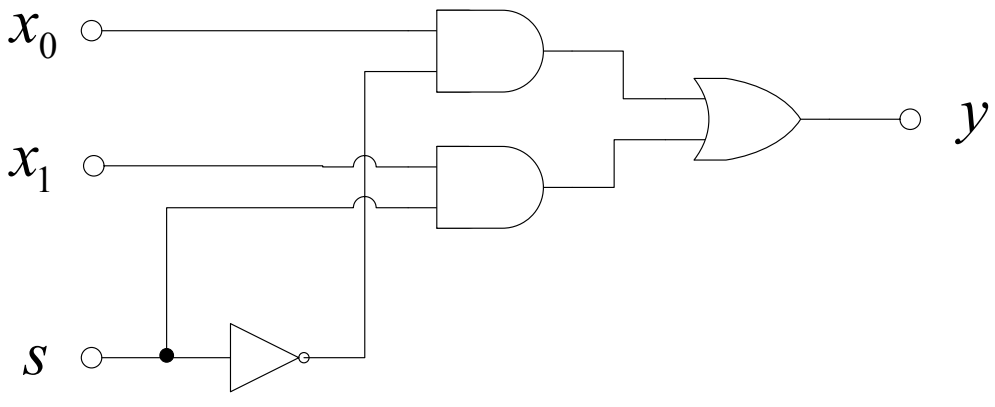


Multiplexer

Realizace



$$y = \overline{s}\, x_0 + s\, x_1$$

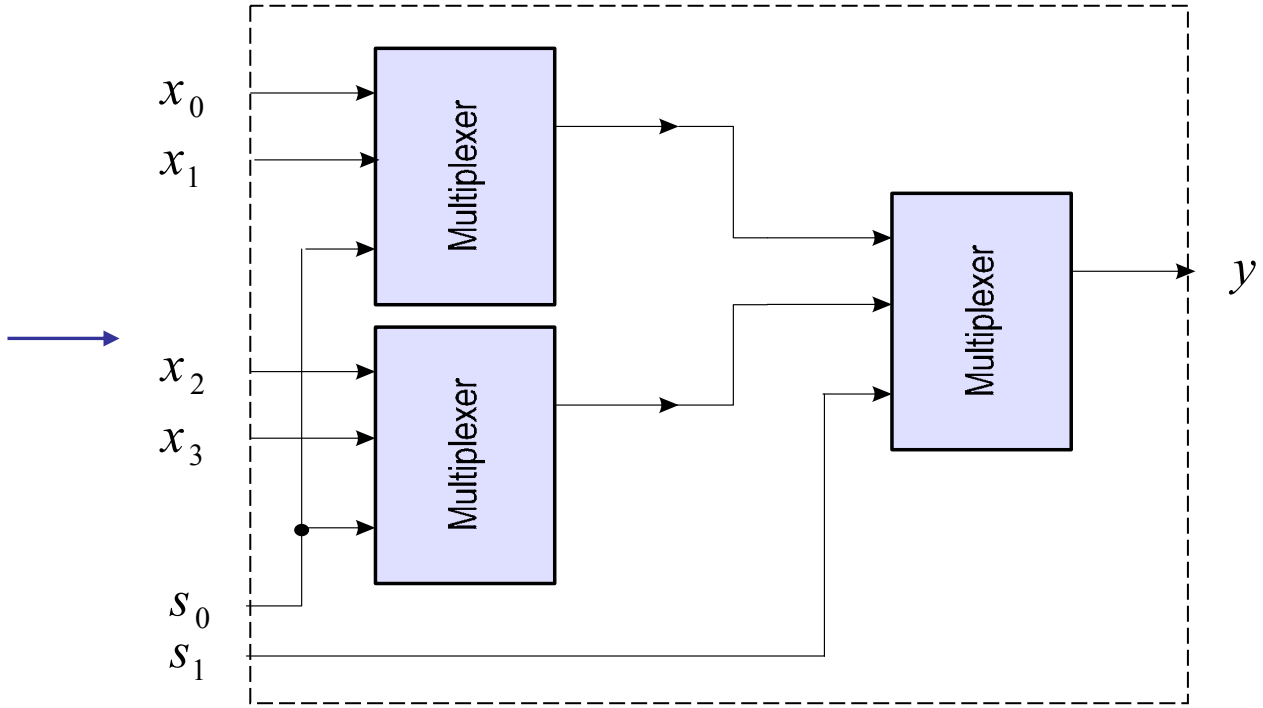


Multiplexer

- Multiplexer 4 na 1 ze tří multiplexerů 2 na 1

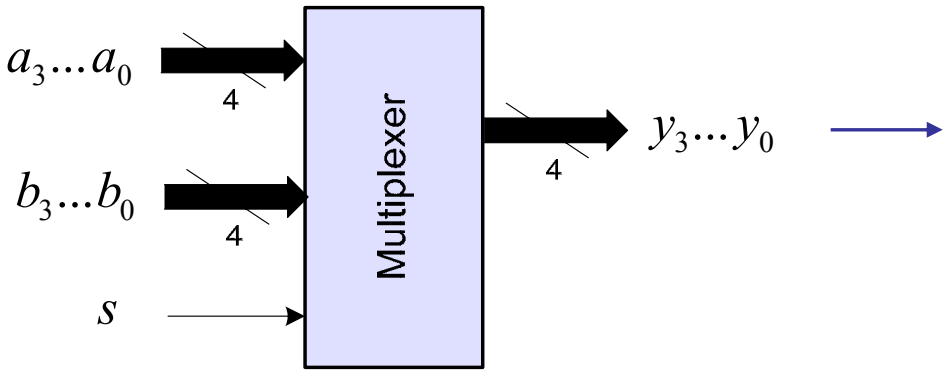
Multiplexer 4 na 1

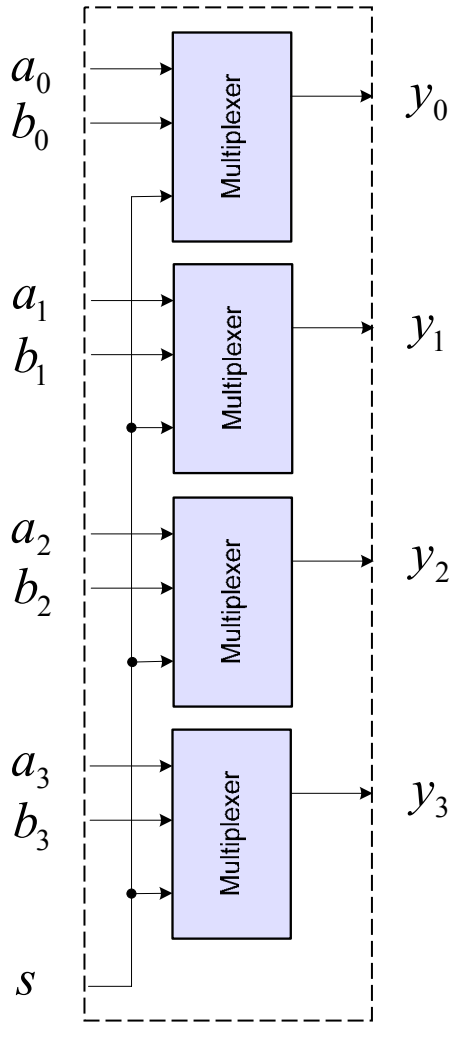
D_i	s_1	s_0	y
0	0	0	x_0
1	0	1	x_1
2	1	0	x_2
3	1	1	x_3



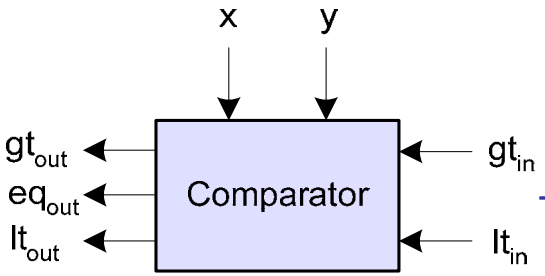
Multiplexer

- Multiplexer 4 x 2 na 1 z multiplexerů 2 na 1





Komparátor (1bit Comparator)



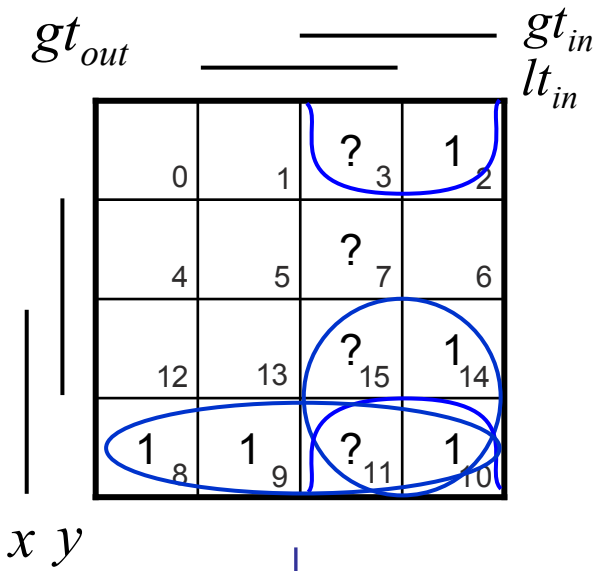
$$\begin{cases} gt_{out} = (x > y)\ \text{OR}\left[(x = y)\ \text{AND}\ (gt_{in} = 1)\right] \\ eq_{out} = (x = y)\ \text{AND}\ (gt_{in} = 0)\ \text{AND}\ (lt_{in} = 0) \\ lt_{out} = (x < y)\ \text{OR}\left[(x = y)\ \text{AND}\ (lt_{in} = 1)\right] \end{cases}$$

Komparátor

D_i	x	y	gt_{in}	lt_{in}	gt_{out}	eq_{out}	lt_{out}
0	0	0	0	0	0	1	0
1	0	0	0	1	0	0	1
2	0	0	1	0	1	0	0
3	0	0	1	1			
4	0	1	0	0	0	0	1
5	0	1	0	1	0	0	1
6	0	1	1	0	0	0	1
7	0	1	1	1			
8	1	0	0	0	1	0	0
9	1	0	0	1	1	0	0
10	1	0	1	0	1	0	0
11	1	0	1	1			
12	1	1	0	0	0	1	0
13	1	1	0	1	0	0	1
14	1	1	1	0	1	0	0
15	1	1	1	1			

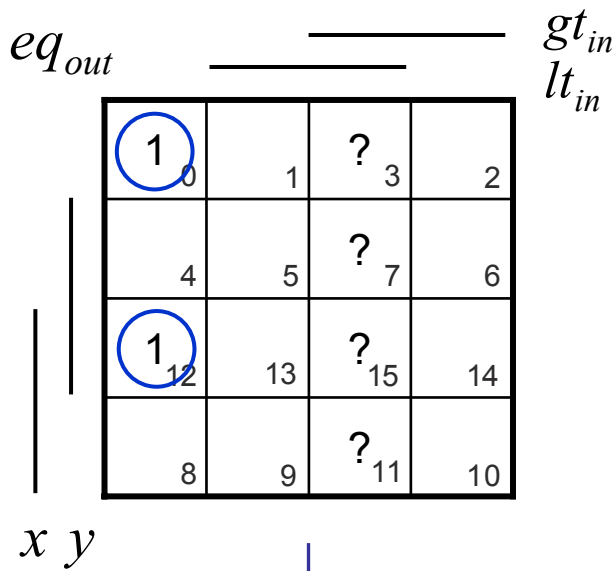
Nemůže nastat, doplníme pro co nejlepší minimalizaci

Komparátor (1bit Comparator)



$$gt_{out} = x.\overline{y} + x.gt_{in} + \overline{y}.gt_{in}$$

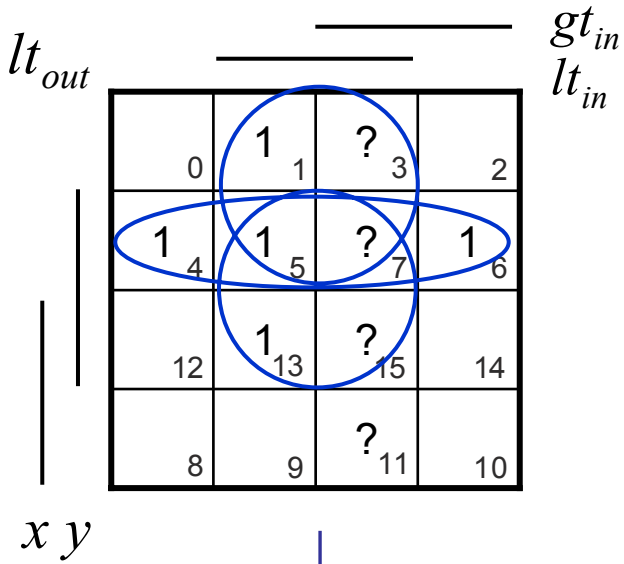
Greater Than



$$eq_{out} = \overline{x}.\overline{y}.\overline{gt_{in}}.\overline{lt_{in}} + x.y.\overline{gt_{in}}.\overline{lt_{in}}$$

Equal

Komparátor (1bit Comparator)

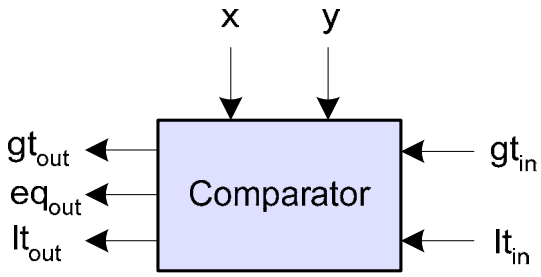


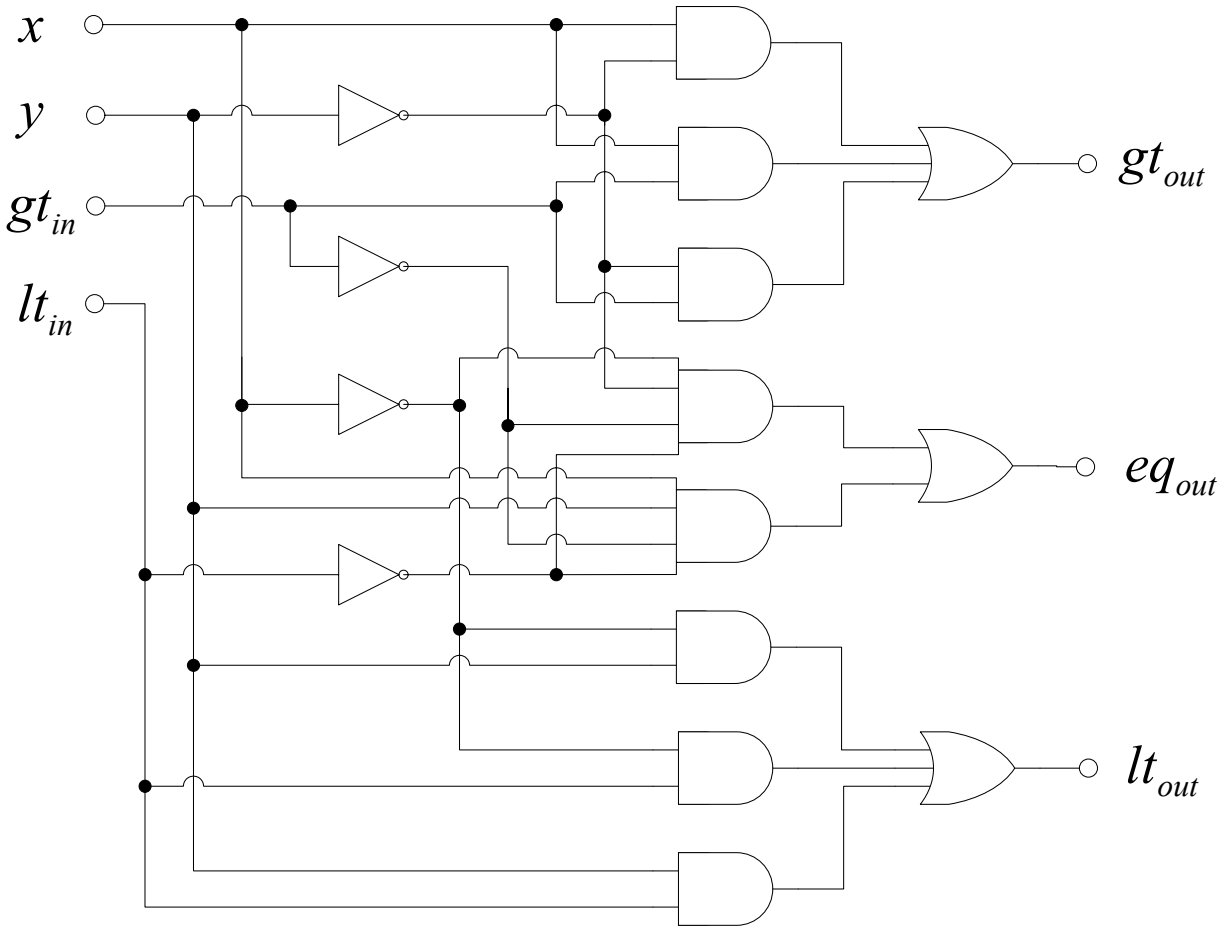
$$lt_{out} = \overline{x}.y + \overline{x}.lt_{in} + y.lt_{in}$$

Less Than

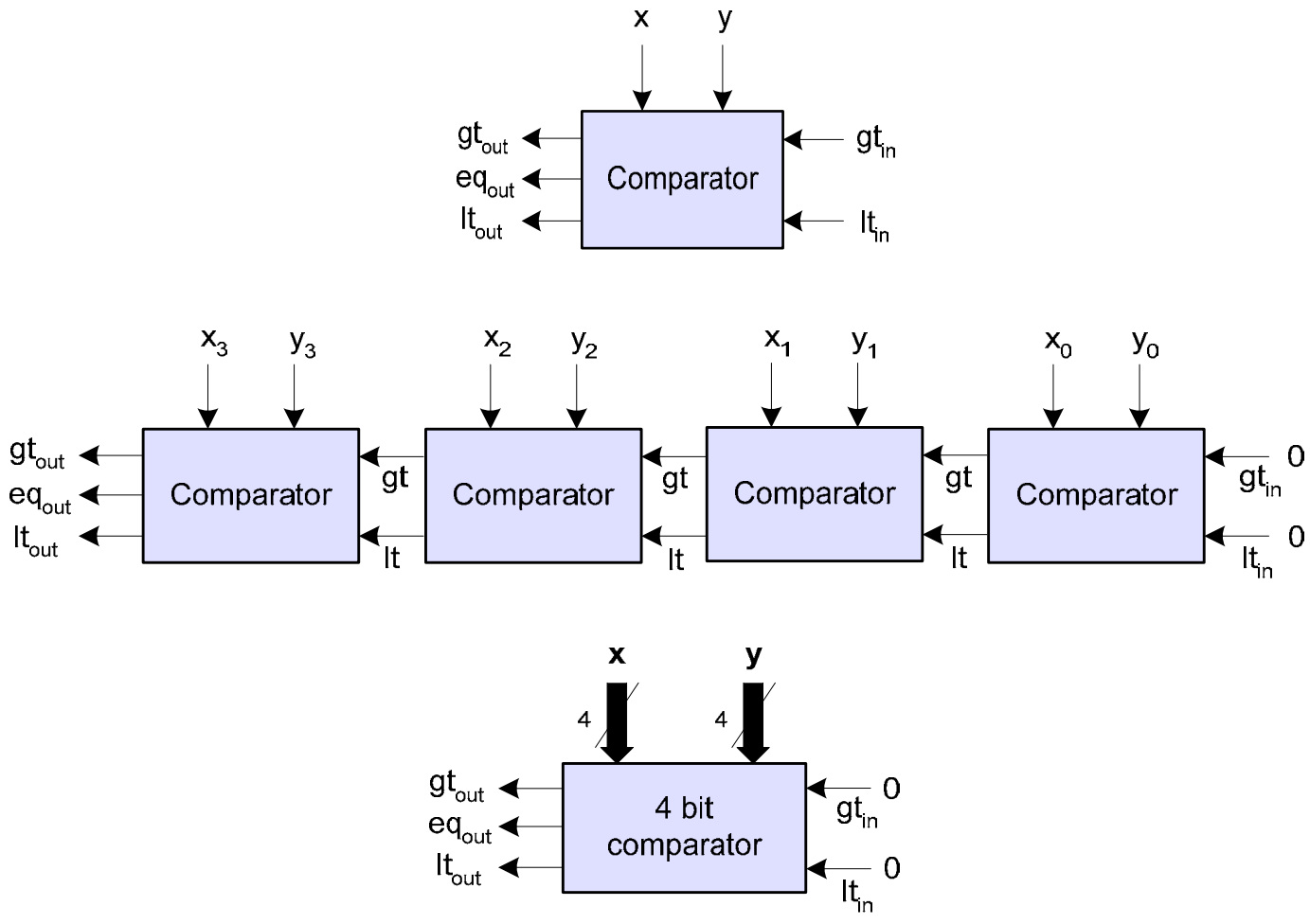
Komparátor (1bit Comparator)

Realizace



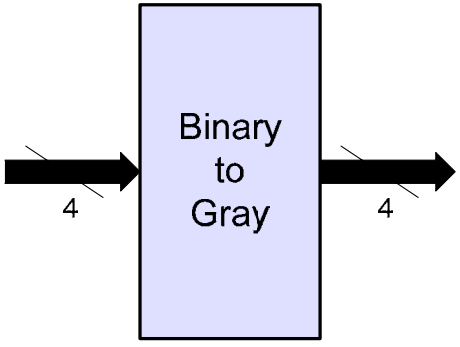


Komparátor (4bit Comparator)



Převodník kódu (Code Converter) – Binary to Gray

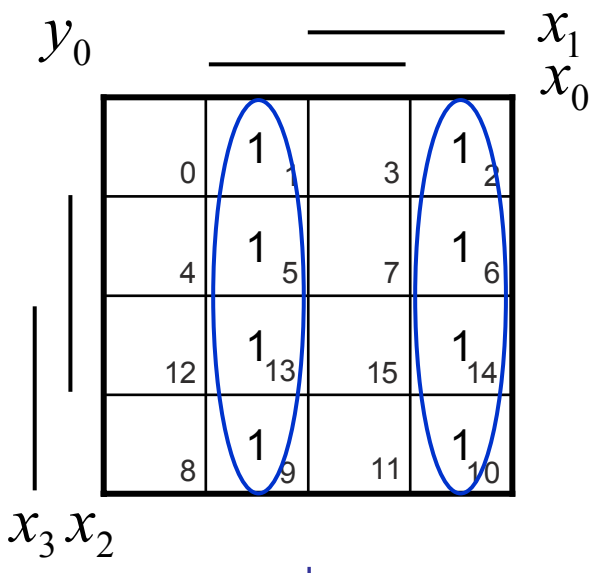
- Binární kód na Grayův kód (sousední kombinace se liší pouze v jednom bitu)



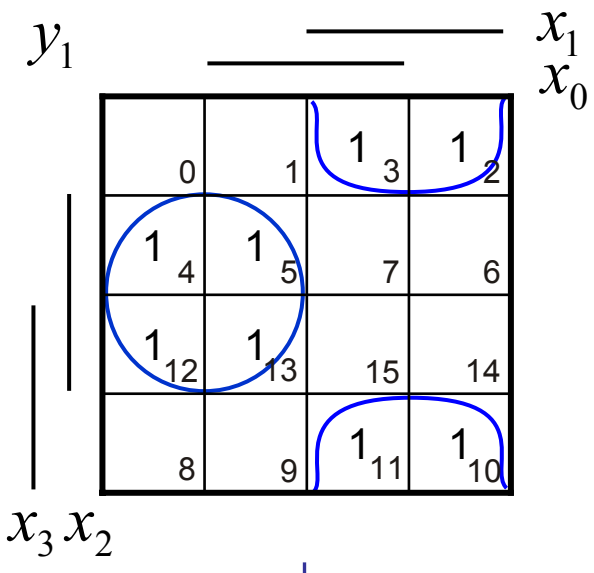
Binární na Grayův kód

D_i	x_3	x_2	x_1	x_0	y_3	y_2	y_1	y_0
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

Převodník kódu (Code Converter) – Binary to Gray

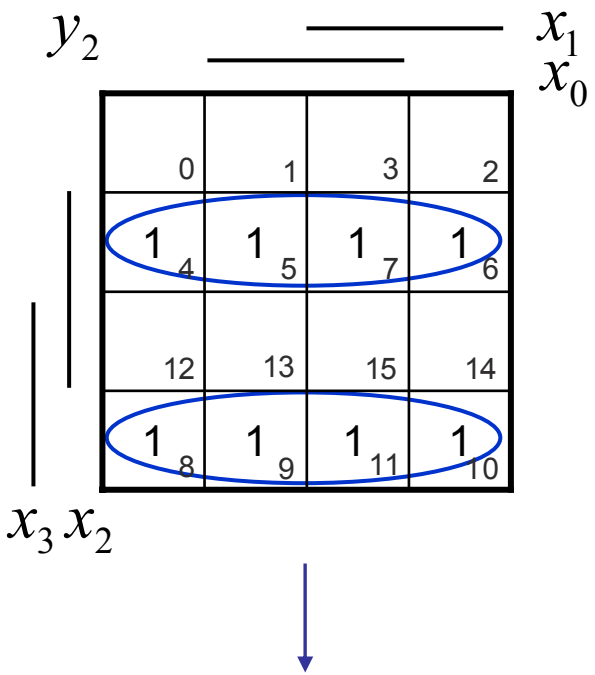


$$y_0 = \overline{x_1}\, x_0 + x_1\, \overline{x_0} = x_1 \oplus x_0$$

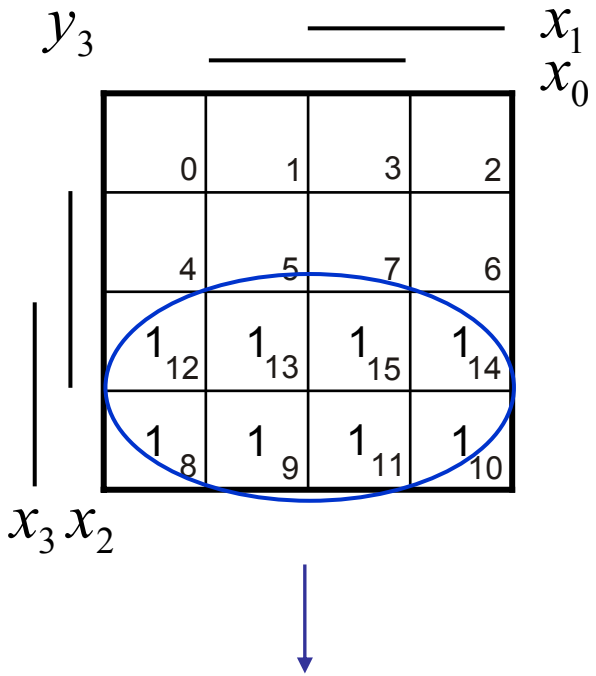


$$y_1 = \overline{x_2}\, x_1 + x_2\, \overline{x_1} = x_2 \oplus x_1$$

Převodník kódu (Code Converter) – Binary to Gray



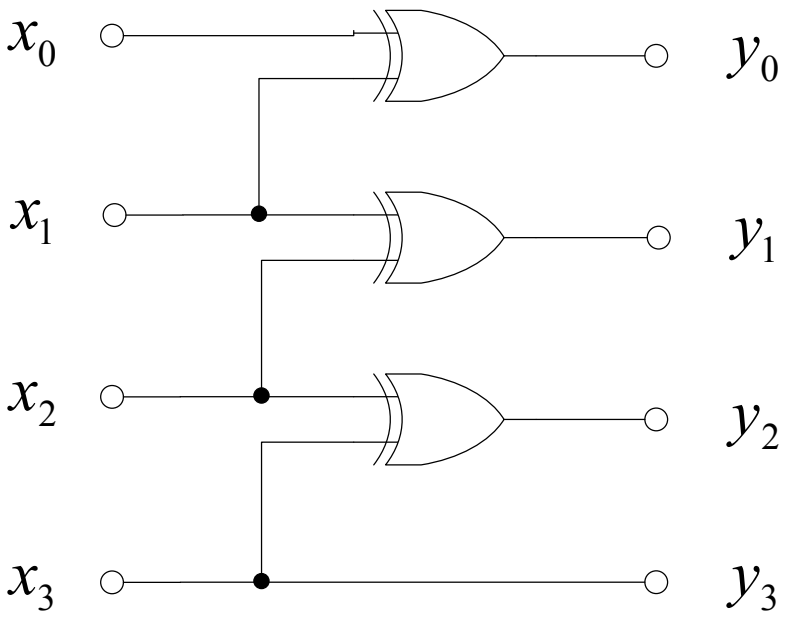
$$y_2 = \overline{x_3}\, x_2 + x_3\, \overline{x_2} = x_3 \oplus x_2$$



$$y_3 = x_3$$

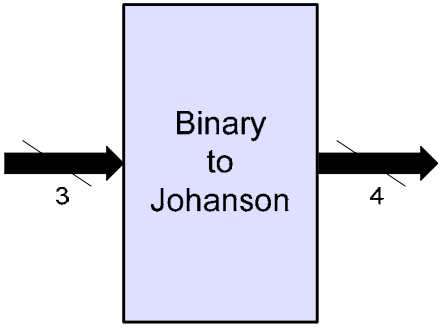
Převodník kódu (Code Converter) – Binary to Gray

Realizace



Převodník kódu (Code Converter) – Binary to Johanson

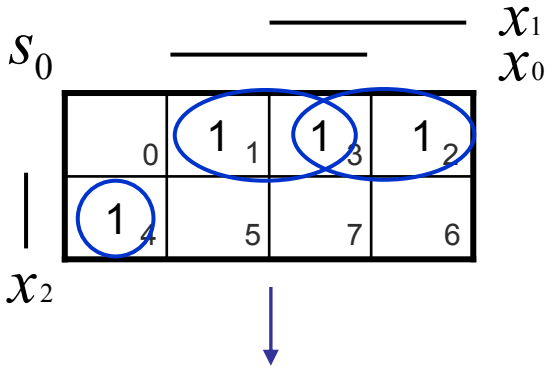
- Binární kód na Johansonův kód (sousední kombinace se liší pouze v jednom bitu)



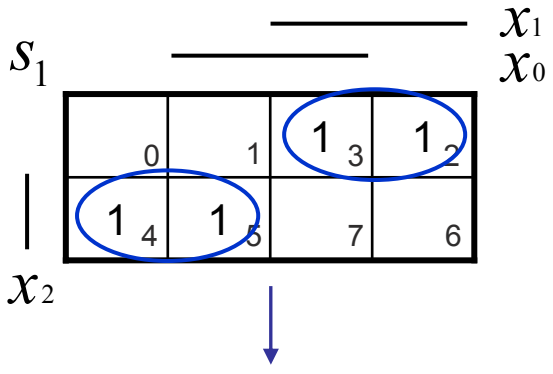
Binární na Johansonův kód

D_i	x_2	x_1	x_0	s_3	s_2	s_1	s_0
0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	1
2	0	1	0	0	0	1	1
3	0	1	1	0	1	1	1
4	1	0	0	1	1	1	1
5	1	0	1	1	1	1	0
6	1	1	0	1	1	0	0
7	1	1	1	1	0	0	0

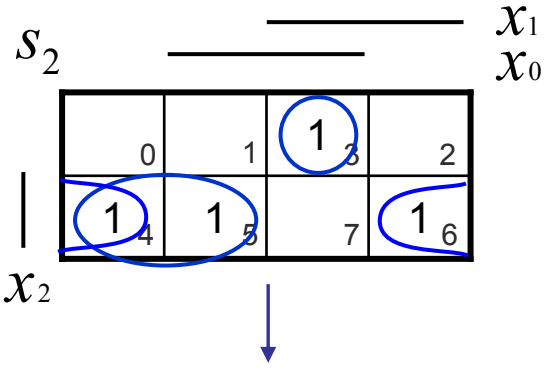
Převodník kódu (Code Converter) – Binary to Johanson



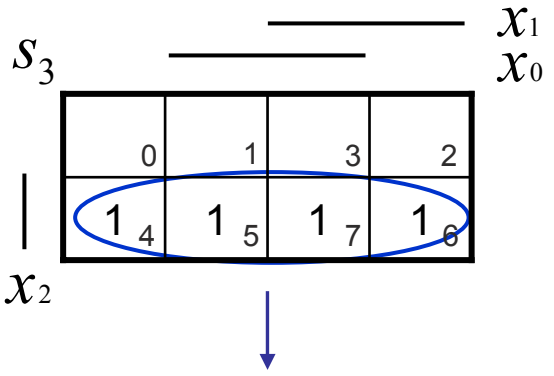
$$s_0 = \overline{x_2}\,x_1 + \overline{x_2}\,x_0 + x_2\,\overline{x_1}\,\overline{x_0}$$



$$s_1 = \overline{x_2}\,x_1 + x_2\,\overline{x_1} = x_2 \oplus x_1$$



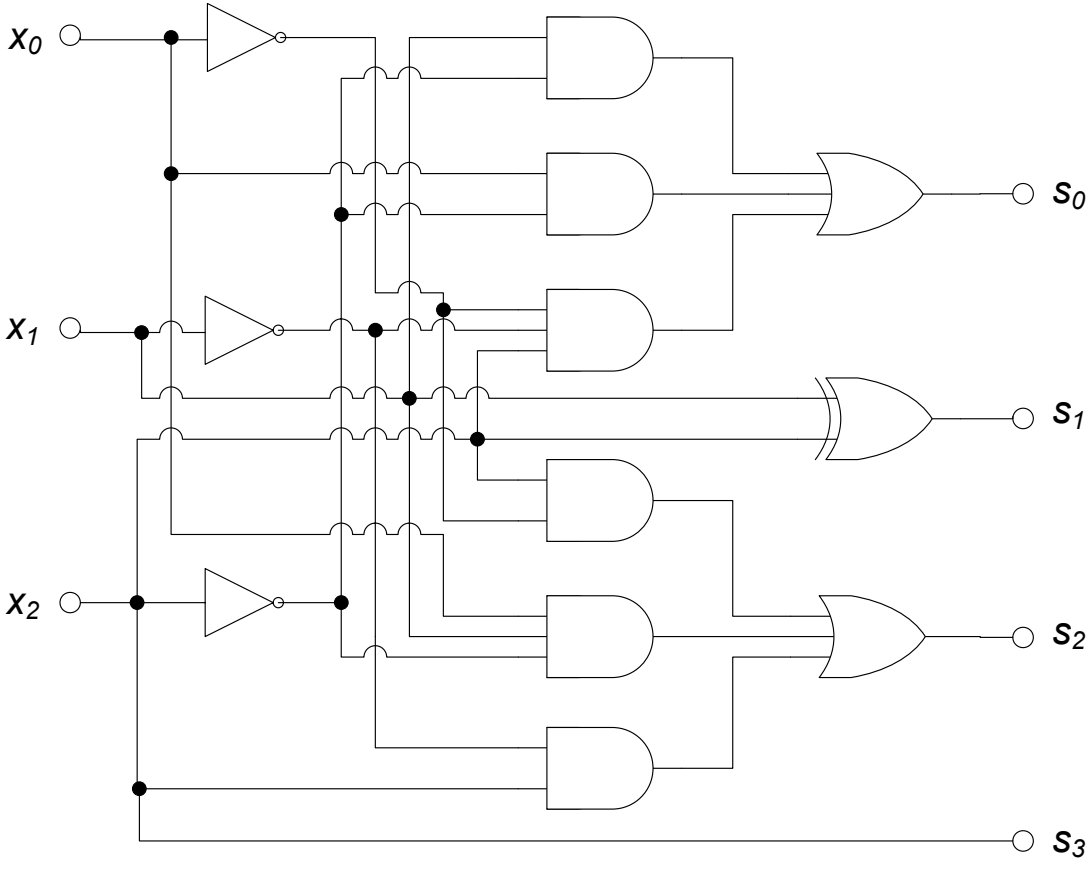
$$s_2 = x_2\,\overline{x_0} + \overline{x_2}\,x_1\,x_0 + x_2\,\overline{x_1}$$



$$s_3 = x_2$$

Převodník kódu (Code Converter) – Binary to Johanson

Realizace

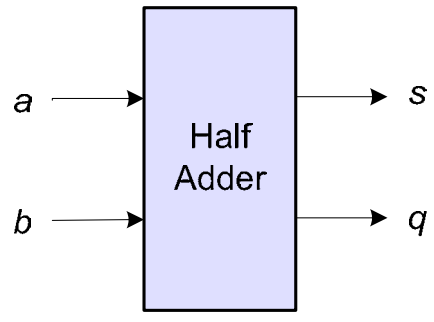


Půlsčítačka (Half Adder)

$$s = a + b$$

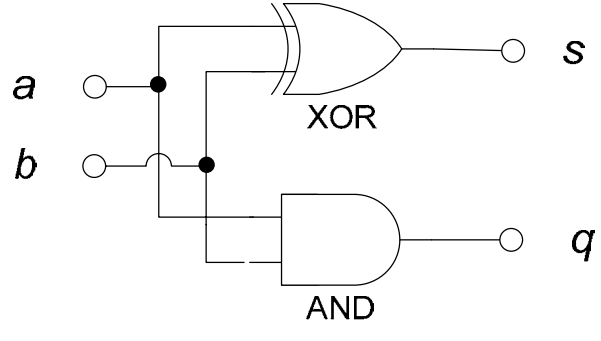
D	a	b	q	s
0	0	0	0	0
1	0	1	0	1
2	1	0	0	1
3	1	1	1	0

Realizace



$$s = \sum \mathrm{m}(1, 2) = \bar{a}b + a\bar{b} = a \oplus b$$

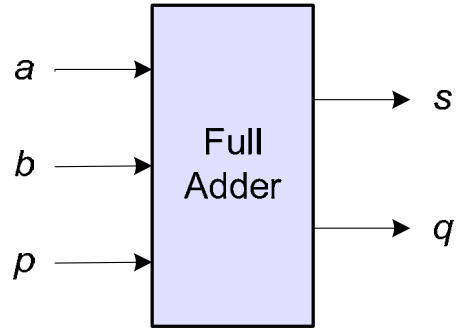
$$q = \sum \mathrm{m}(3) = ab$$



Sčítačka (Full Adder)

$$s = a + b + p$$

D	a	b	p	q	s
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1



$$s = \sum \mathrm{m}(1,\ 2,\ 4,\ 7) = \bar{a}.\bar{b}.p + \bar{a}.b.\bar{p} + a.\bar{b}.\bar{p} + a.b.p$$

$$q = \sum \mathrm{m}(3,\ 5,\ 6,\ 7) = \bar{a}.b.p + a.\bar{b}.p + a.b.\bar{p} + a.b.p$$

Sčítačka (Full Adder)

Úprava logického výrazu

$$s = \bar{a}.\bar{b}.p + \bar{a}.b.\bar{p} + a.\bar{b}.\bar{p} + a.b.p = \bar{p}(\bar{a}\,b + a\,\bar{b}) + p(\bar{a}\,\bar{b} + a\,b) =$$

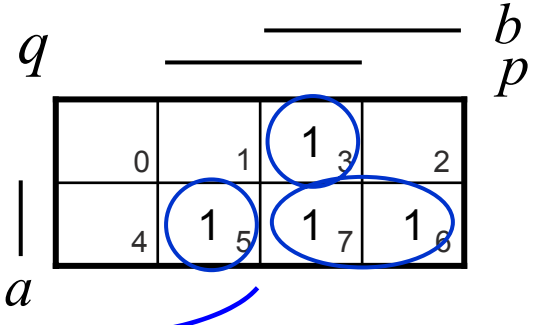
$$= \bar{p}(\bar{a}\,b + a\,\bar{b}) + p\,\overline{(\bar{a}\,b + a\,\bar{b})} = p \oplus (a \oplus b)$$

XOR, XNOR, XNOR = XOR

Half Adder s

Vhodná minimalizace z K-mapy

$$q = \bar{a}.b.p + a.\bar{b}.p + a.b.\bar{p} + a.b.p$$



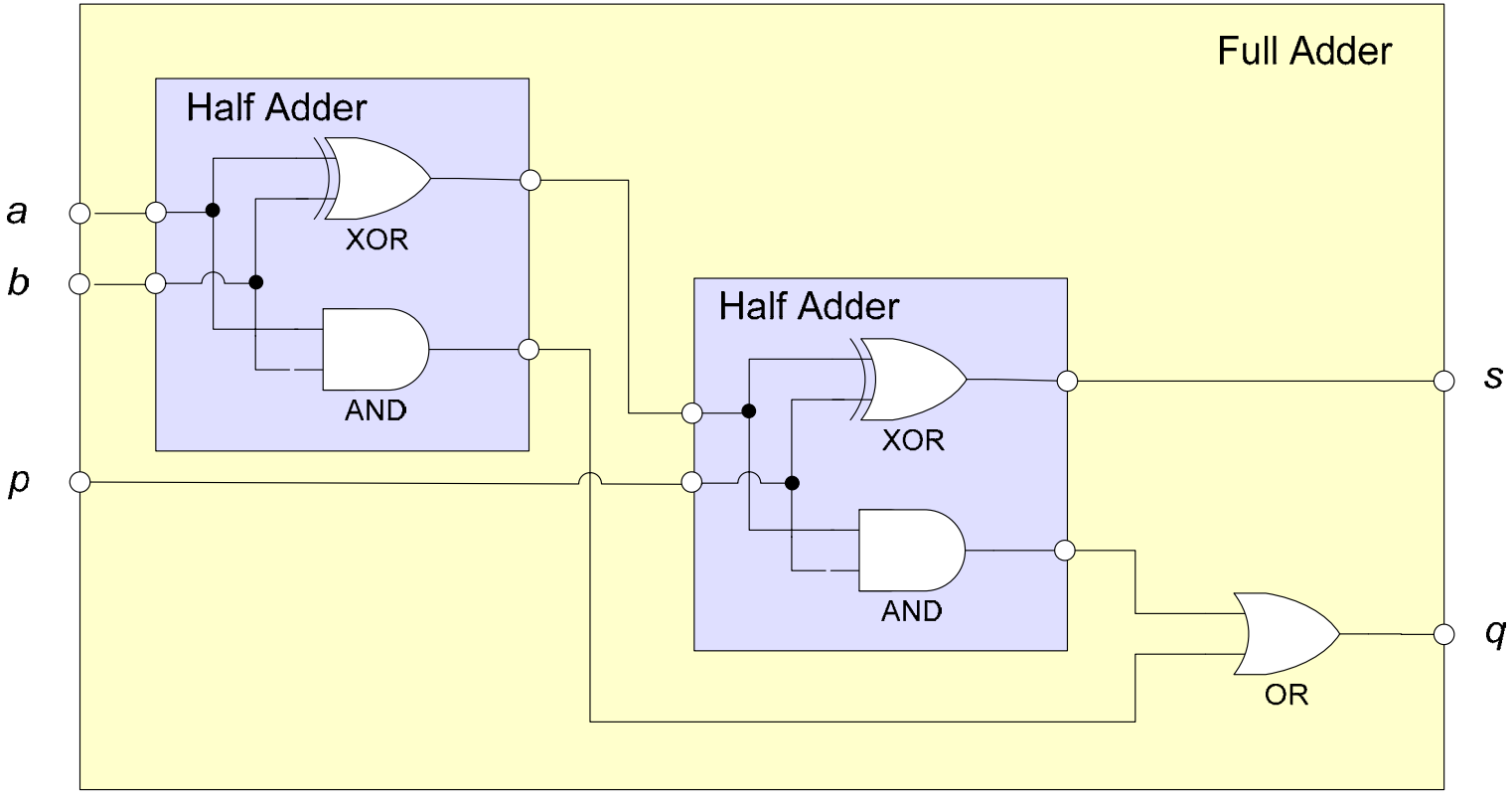
$$q = ab + p\bar{a}b + pa\bar{b} = ab + p(\bar{a}b + a\bar{b}) =$$

$$= ab + p(a \oplus b)$$

Half Adder q

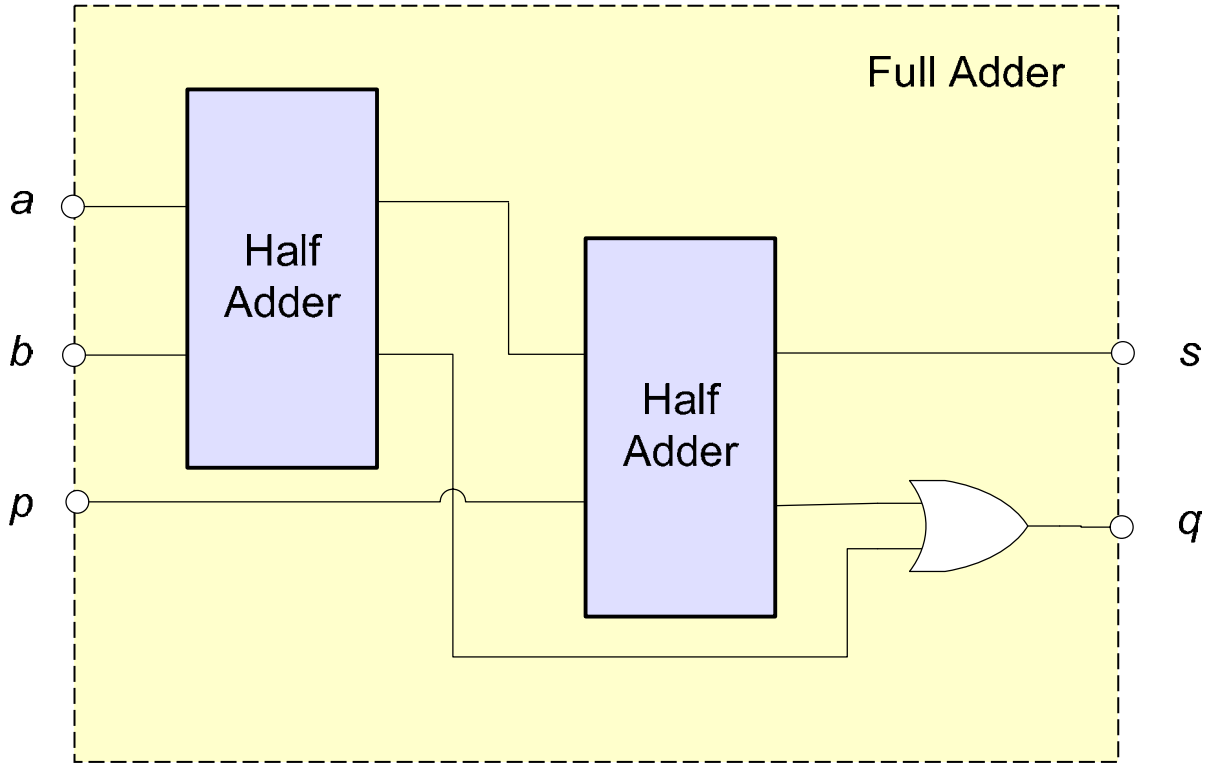
Sčítačka (Full Adder)

Realizace

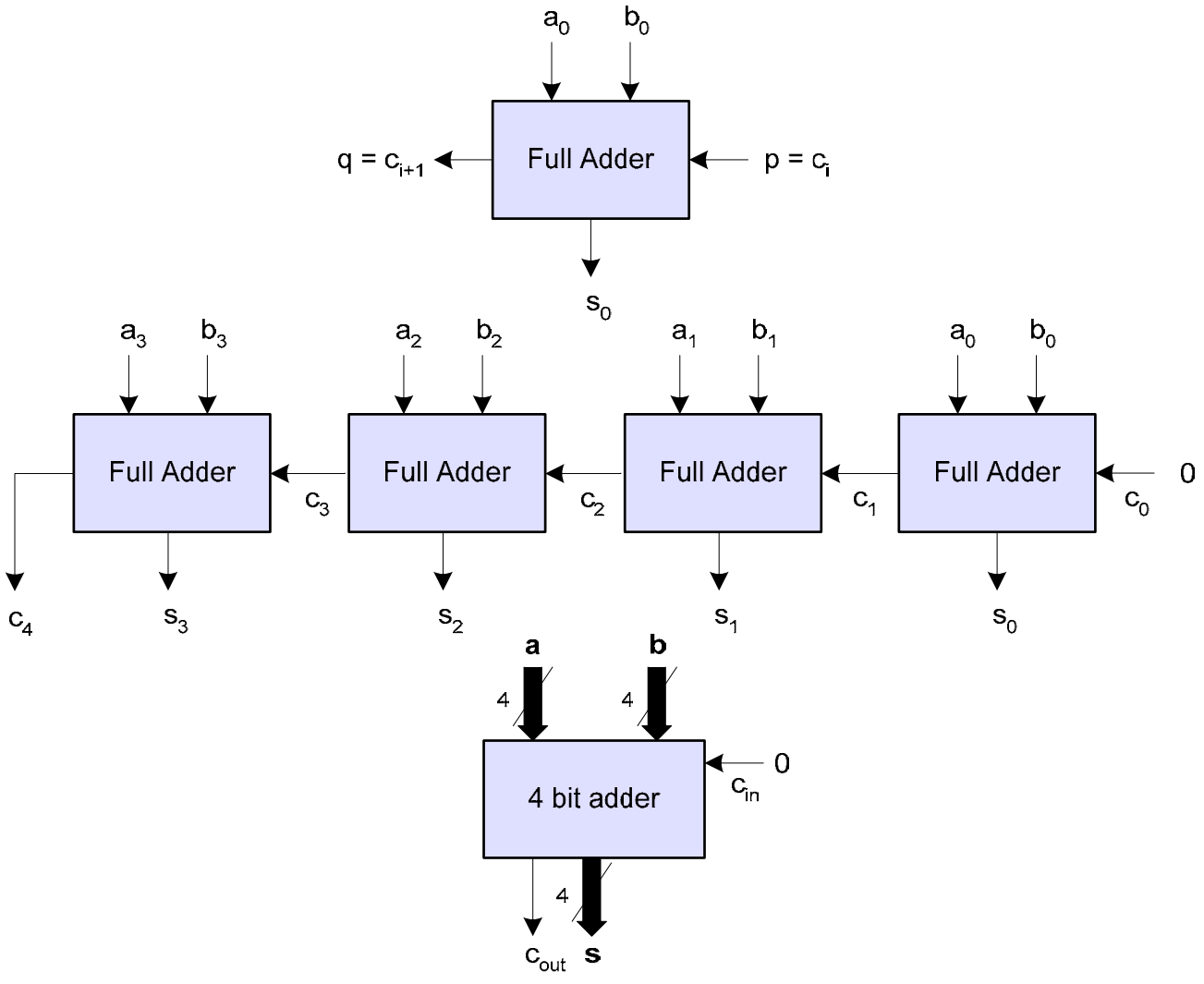


Sčítačka (Full Adder)

Realizace



Sčítačka (4bit Adder)

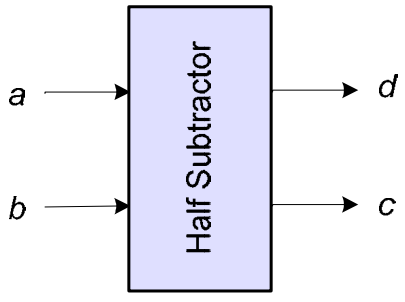


Půlodčítačka (Half Subtractor)

$$d = a - b$$

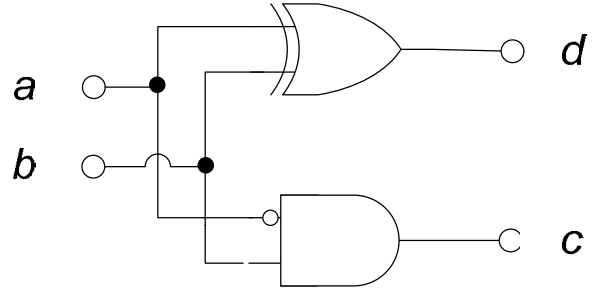
D	a	b	c	d
0	0	0	0	0
1	0	1	1	1
2	1	0	0	1
3	1	1	0	0

Realizace



$$d = \sum \mathrm{m}(1, 2) = \bar{a}b + a\bar{b} = a \oplus b$$

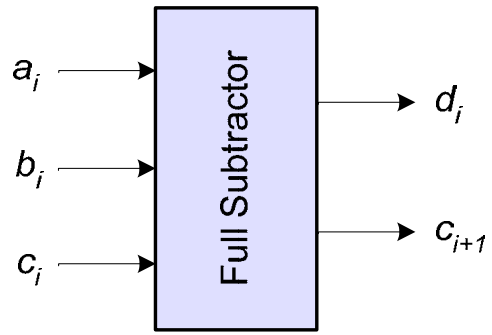
$$c = \sum \mathrm{m}(1) = \bar{a}\,b$$



Odčítačka (Full Subtractor)

$$d_i = a_i - b_i - c_i$$

D	a_i	b_i	c_i	c_{i+1}	d_i
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	1



$$d_i = \sum \mathrm{m}(1,\ 2,\ 4,\ 7) = \overline{a_i}.\overline{b_i}.c_i + \overline{a_i}.b_i.\overline{c_i} + a_i.\overline{b_i}.\overline{c_i} + a_i.b_i.c_i$$

$$c_{i+1} = \sum \mathrm{m}(1,\ 2,\ 3,\ 7) = \overline{a_i}.\overline{b_i}.c_i + \overline{a_i}.b_i.\overline{c_i} + \overline{a_i}.b_i.c_i + a_i.b_i.c_i$$

Odčítačka (Full Subtractor)

Úprava logického výrazu

$$d_i = \overline{a_i}.\overline{b_i}.c_i + \overline{a_i}.b_i.\overline{c_i} + a_i.\overline{b_i}.\overline{c_i} + a_i.b_i.c_i = \overline{c_i}\,(\overline{a_i}\,b_i + a_i\,\overline{b_i}) + c_i\,(\overline{a_i}\,\overline{b_i} + a_i\,b_i) =$$

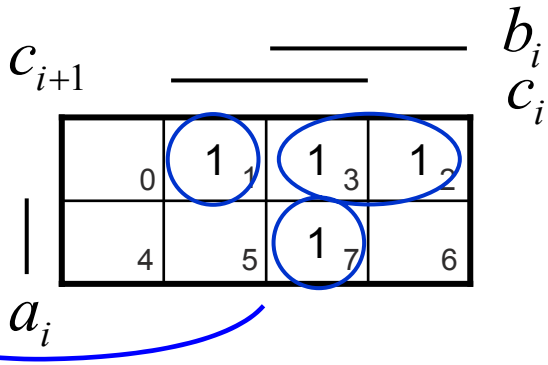
$$= \overline{c_i}\,(\overline{a_i}\,b_i + a_i\,\overline{b_i}) + c_i\,\overline{(\overline{a_i}\,b_i + a_i\,\overline{b_i})} = c_i \oplus (a_i \oplus b_i)$$

XOR, XNOR, XNOR = XOR

Half Subtractor d

Vhodná minimalizace z K-mapy

$$c_{i+1} = \overline{a_i}.\overline{b_i}.c_i + \overline{a_i}.b_i.\overline{c_i} + \overline{a_i}.b_i.c_i + a_i.b_i.c_i$$



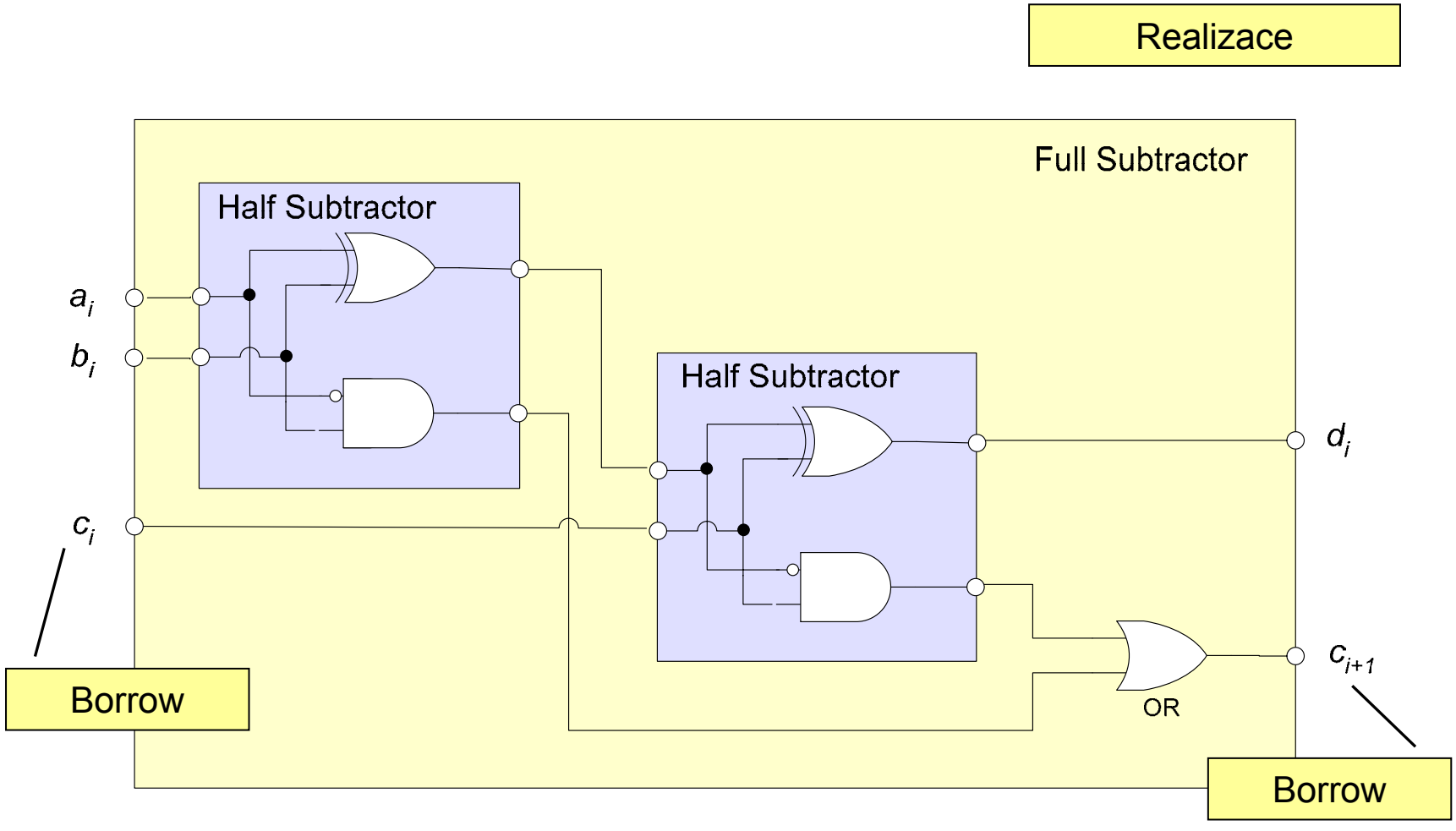
$$c_{i+1} = \overline{a_i}\,b_i + c_i\,\overline{a_i}\,\overline{b_i} + c_i\,a_i\,b_i = \overline{a_i}\,b_i + c_i\,(\overline{a_i}\,\overline{b_i} + a_i\,b_i) =$$

$$= \overline{a_i}\,b_i + c_i\,\overline{(a_i \oplus b_i)}$$

XNOR

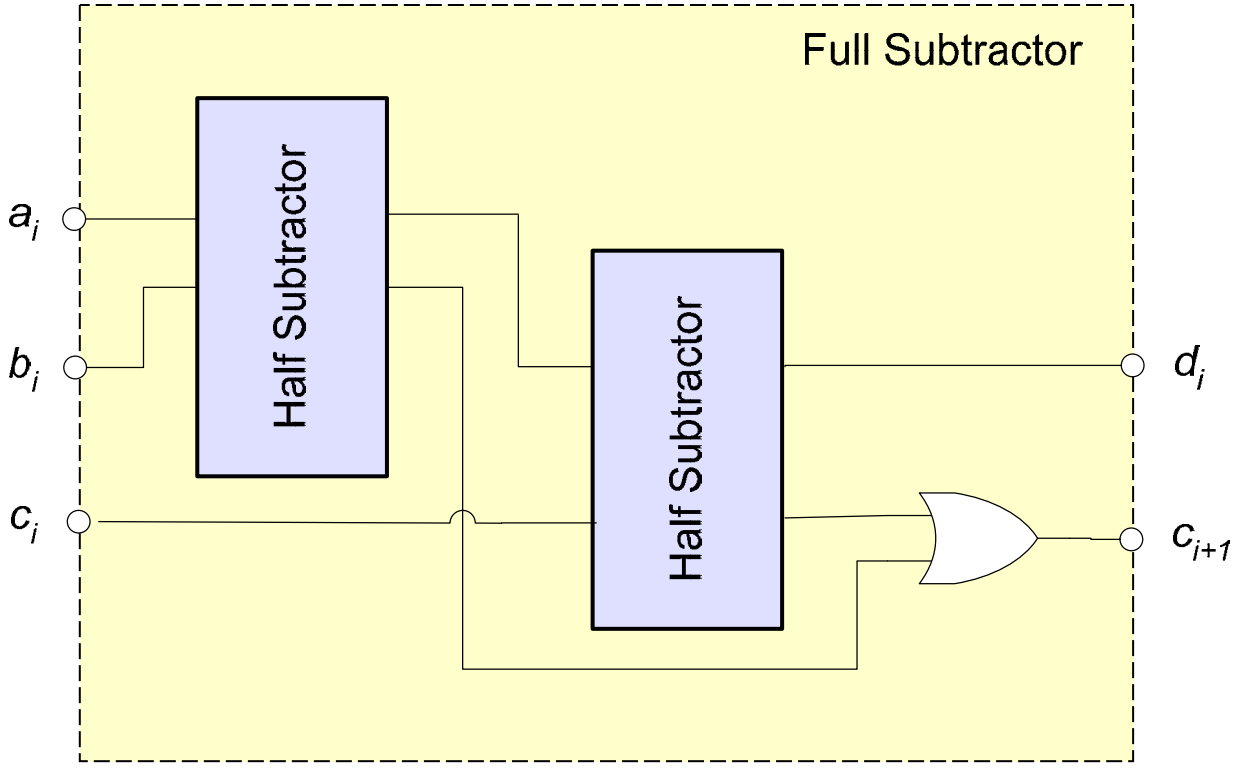
Half Subtractor c

Odčítačka (Full Subtractor)



Odčítačka (Full Subtractor)

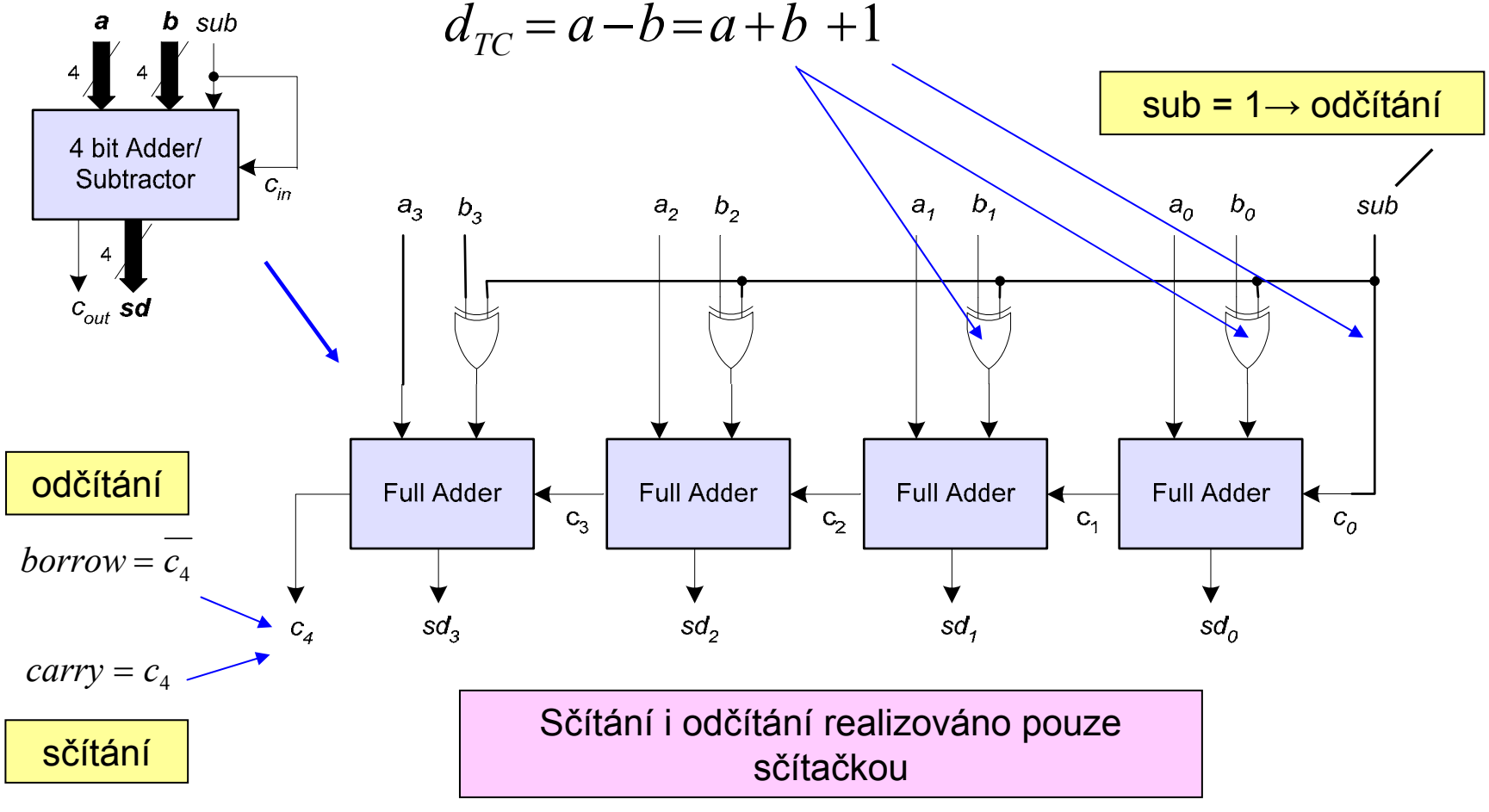
Realizace



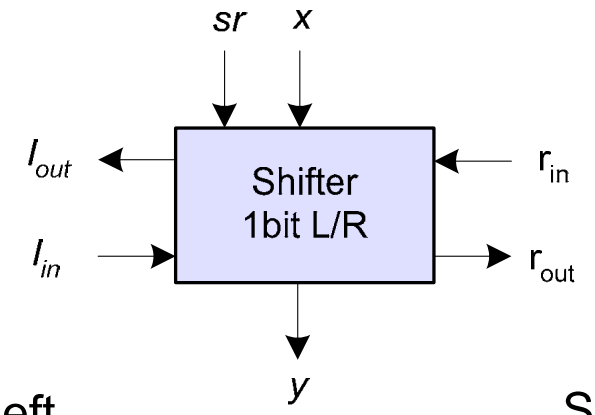
Sčítačka/odčítačka (4bit Adder/Subtractor)

- Pro čísla kódovaná v dvojkovém doplňku (Two's Complement) platí:

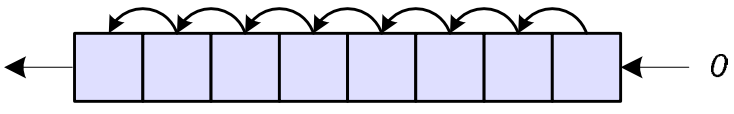
$$d_{TC} = a - b = a + \overline{b} + 1$$



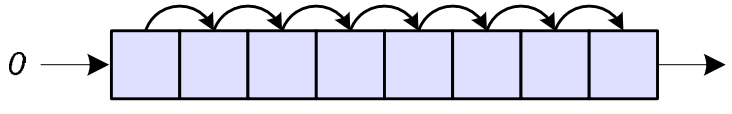
Shifter (1bit Left/Right)



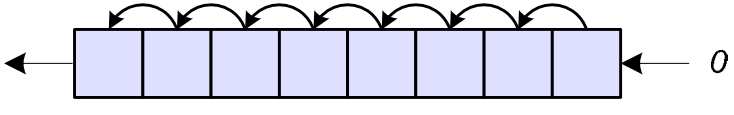
Shift Logical Left



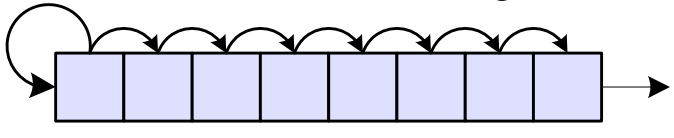
Shift Logical Right



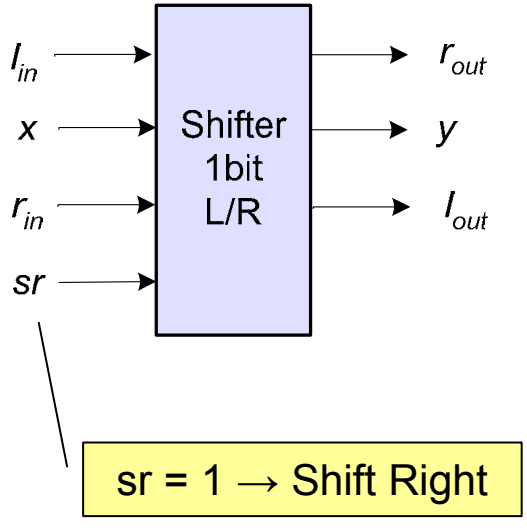
Shift Arithmetic Left



Shift Arithmetic Right



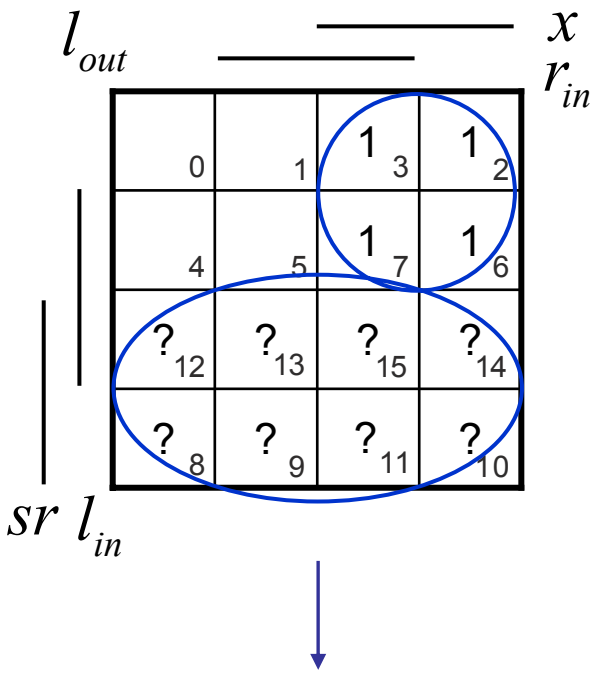
Shifter (1bit Left/Right)



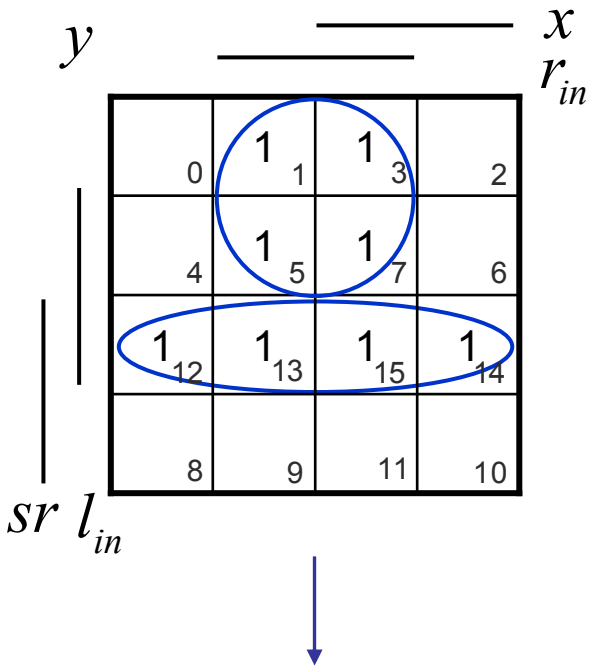
Posun binárního řádu (o 1 bit)

D_i	sr	l_{in}	x	r_{in}	l_{out}	y	r_{out}
0	0	0	0	0	0	0	-
1	0	0	0	1	0	1	-
2	0	0	1	0	1	0	-
3	0	0	1	1	1	1	-
4	0	1	0	0	0	0	-
5	0	1	0	1	0	1	-
6	0	1	1	0	1	0	-
7	0	1	1	1	1	1	-
8	1	0	0	0	-	0	0
9	1	0	0	1	-	0	0
10	1	0	1	0	-	0	1
11	1	0	1	1	-	0	1
12	1	1	0	0	-	1	0
13	1	1	0	1	-	1	0
14	1	1	1	0	-	1	1
15	1	1	1	1	-	1	1

Shifter (1bit Left/Right)

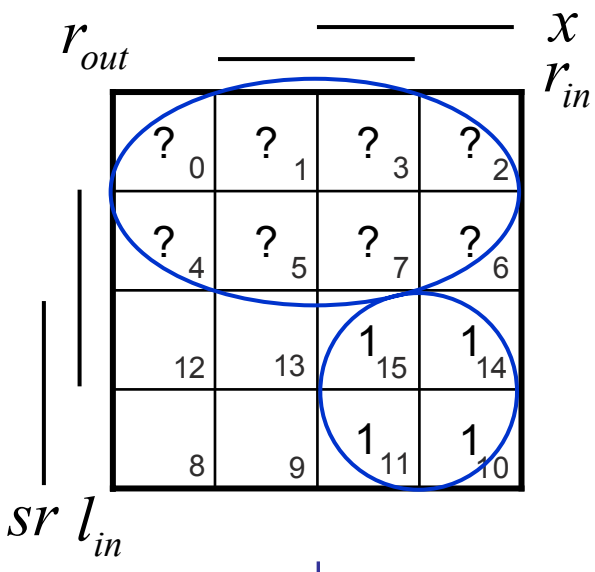


$$l_{out} = sr + \overline{sr}.x$$



$$y = sr.l_{in} + \overline{sr}.r_{in}$$

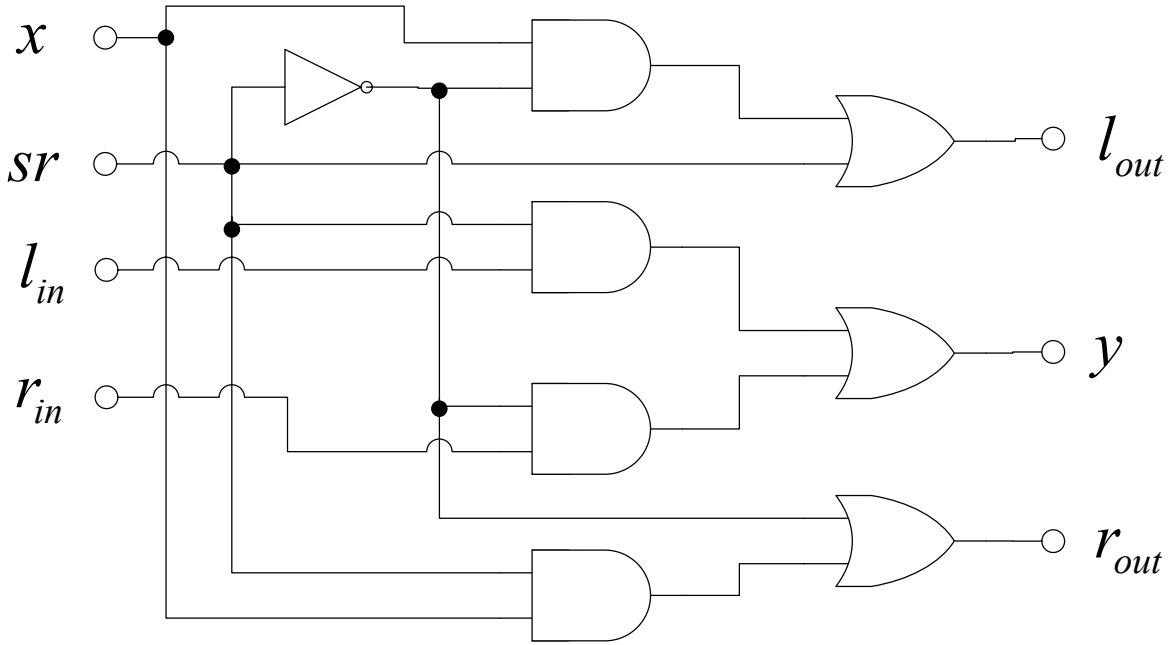
Shifter (1bit Left/Right)



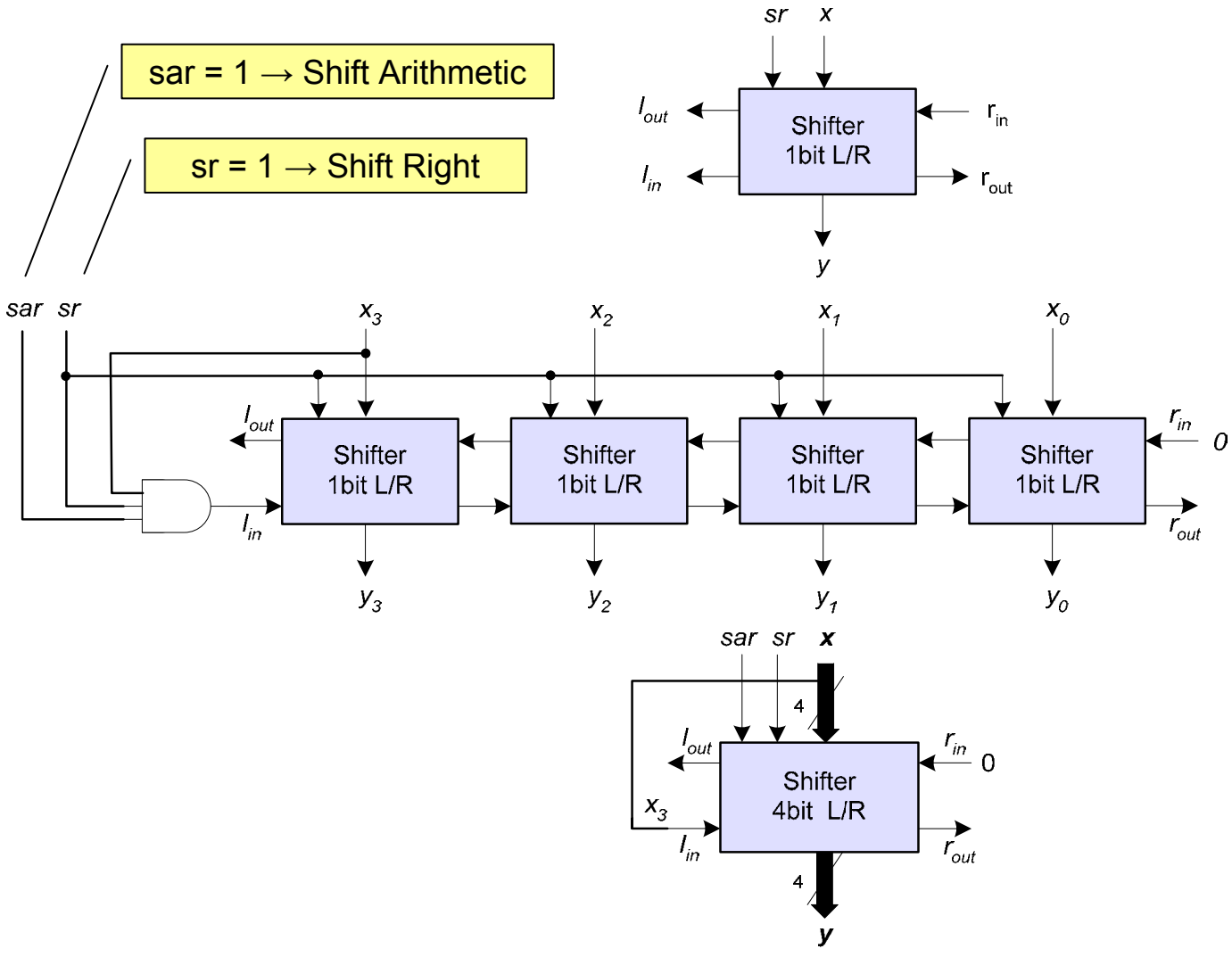
$$r_{out} = \overline{sr} + sr.x$$

Shifter (1bit Left/Right)

Realizace

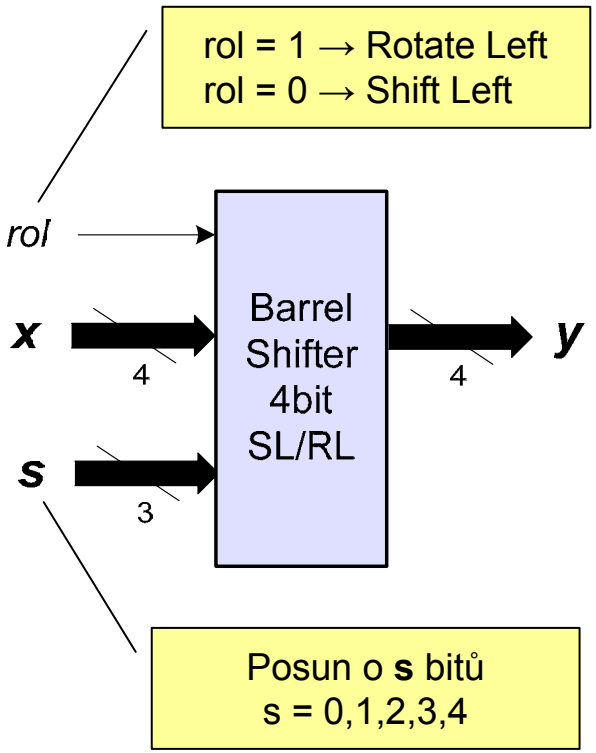


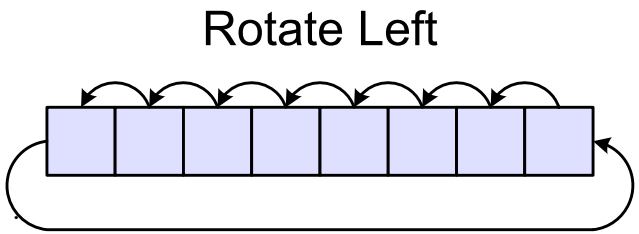
4bit Shifter (1bit Left/Right, Logical/Arithmetic)

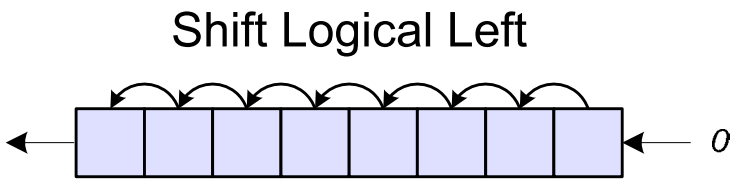


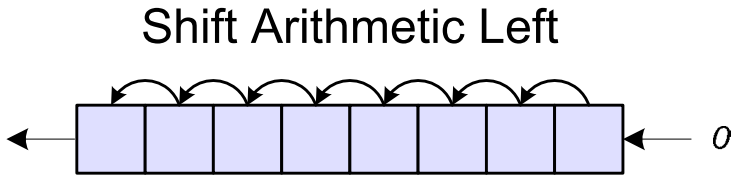
Barrel Shifter (4bit, Shift Left/Rotate Left)

- Posun vlevo nebo rotace vlevo o s bitů (kombinační obvod)
- Zde ukázka posunu vlevo, podobně vpravo (pozor vpravo se liší logický a aritmetický posun)

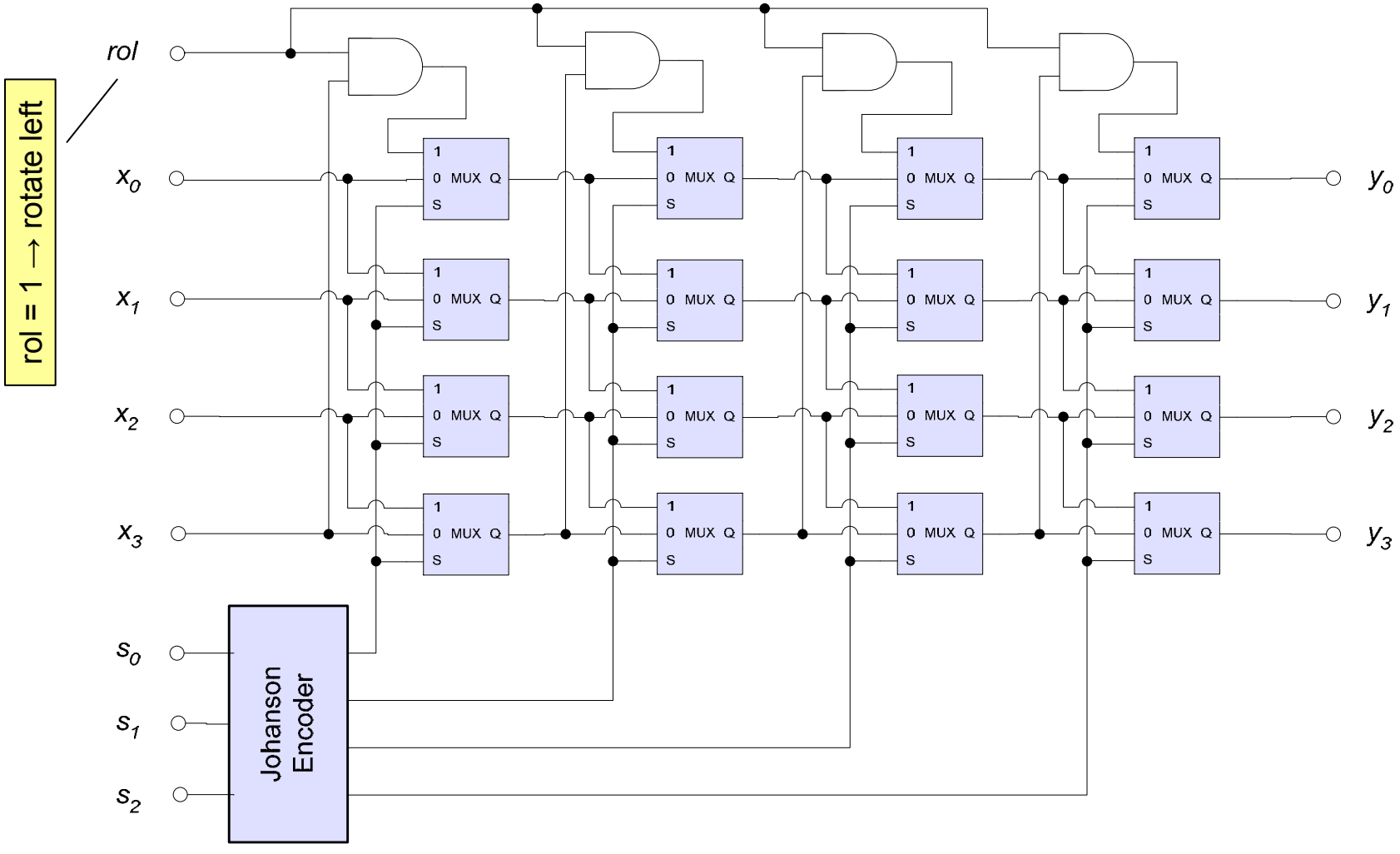








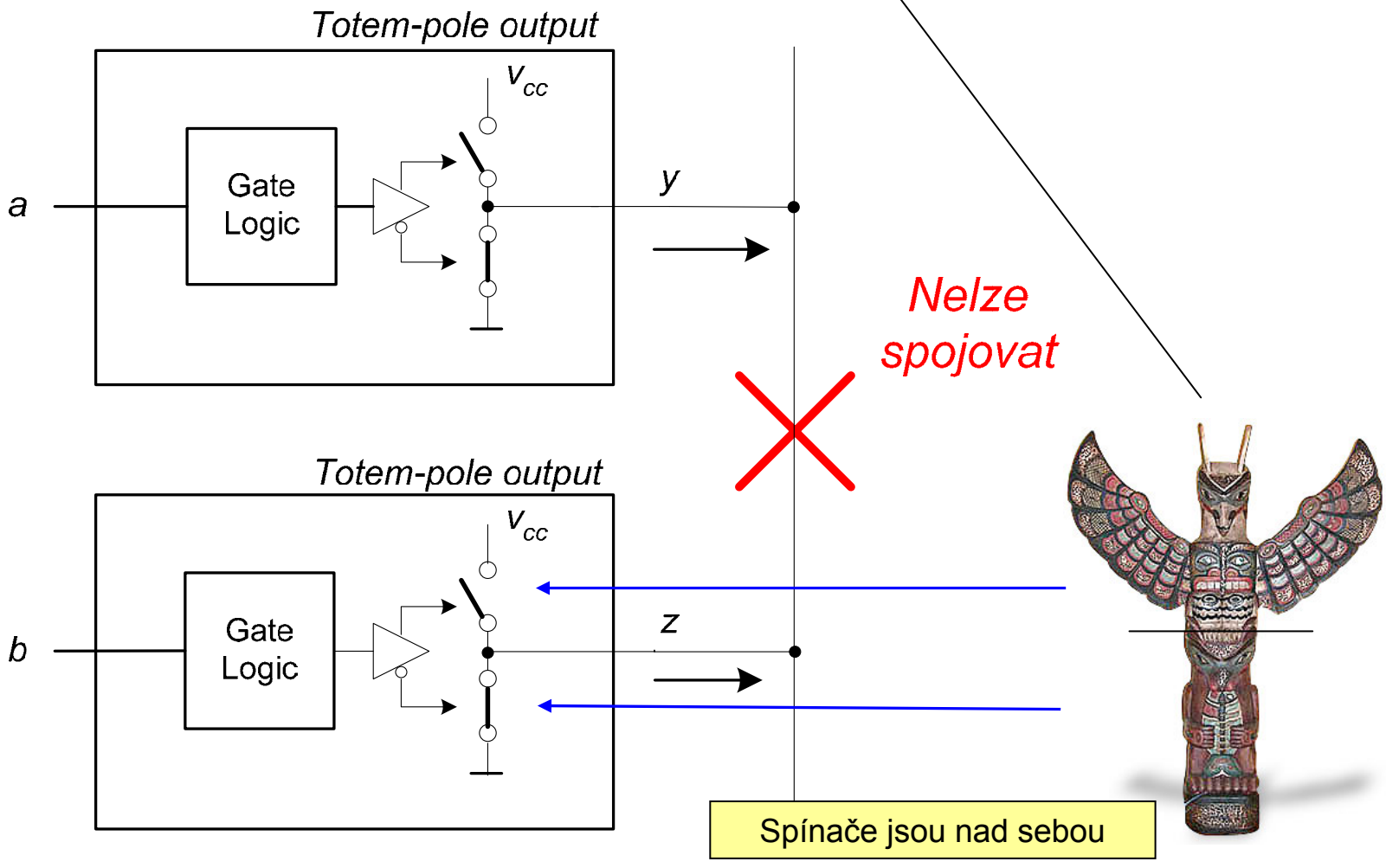
Barrel Shifter (4bit, Shift Left/Rotate Left)



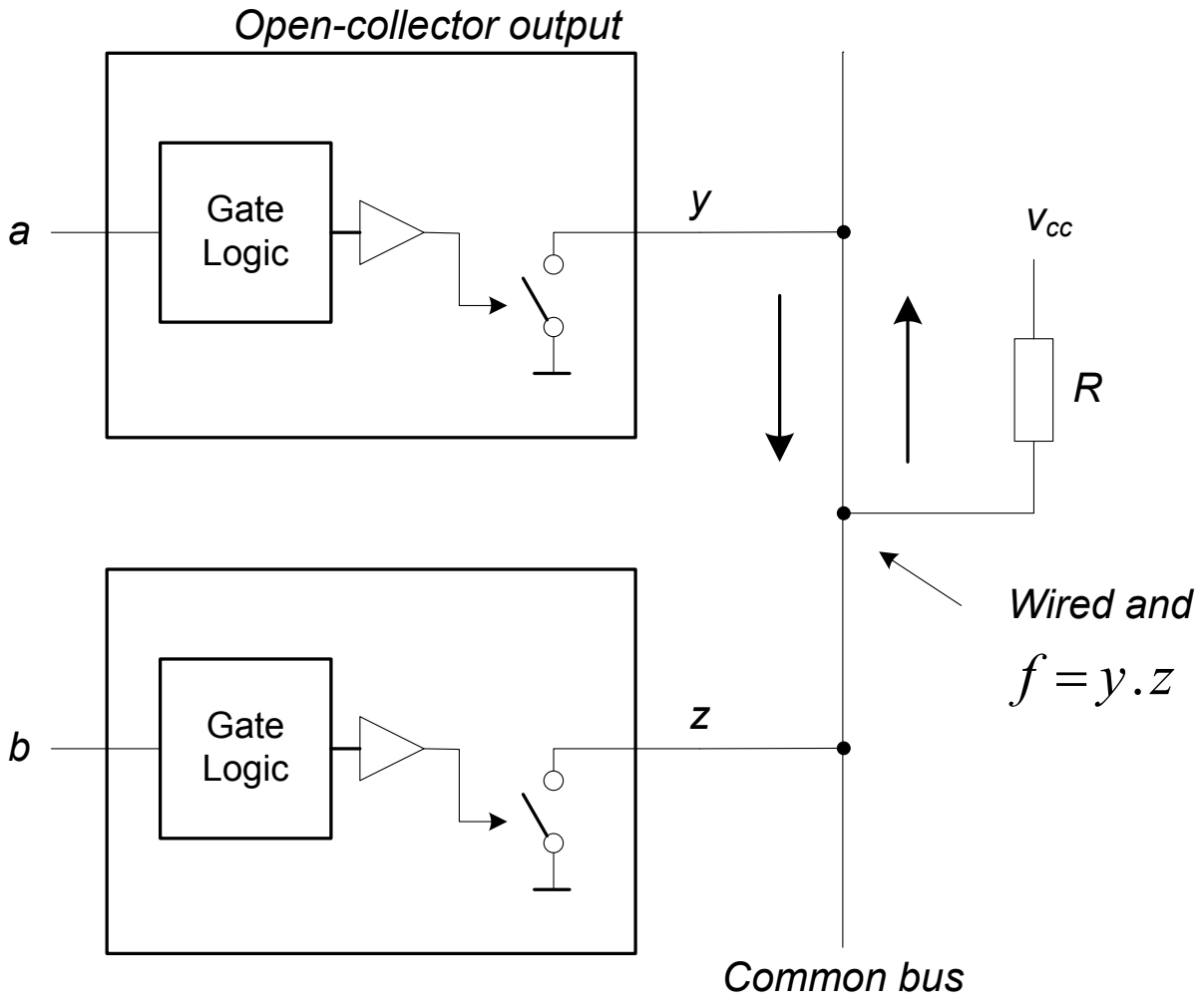
Typy výstupů logických členů

- Standardní výstup – Totem-pole output (Push-Pull)
  - Dvoustavový výstup
  - Na výstupu vždy hodnota 0 nebo 1
  - Výstupy nelze navzájem spojovat
- Otevřený kolektor – Open-collector output (OC)
  - Na výstupu pouze spodní spínač
  - Výstupy lze spojit, nutný upínací odpor na V_{cc}
  - Montážní součin – Wired-AND
- Třístavový výstup – Tri-state output (TS)
  - Na výstupu hodnoty 0, 1, Z (Z = vysoká impedance-odpojeno)
  - Výstupy lze spojovat
  - Řízení výstupních členů musí zajistit, že pouze jeden vysílač není v Z

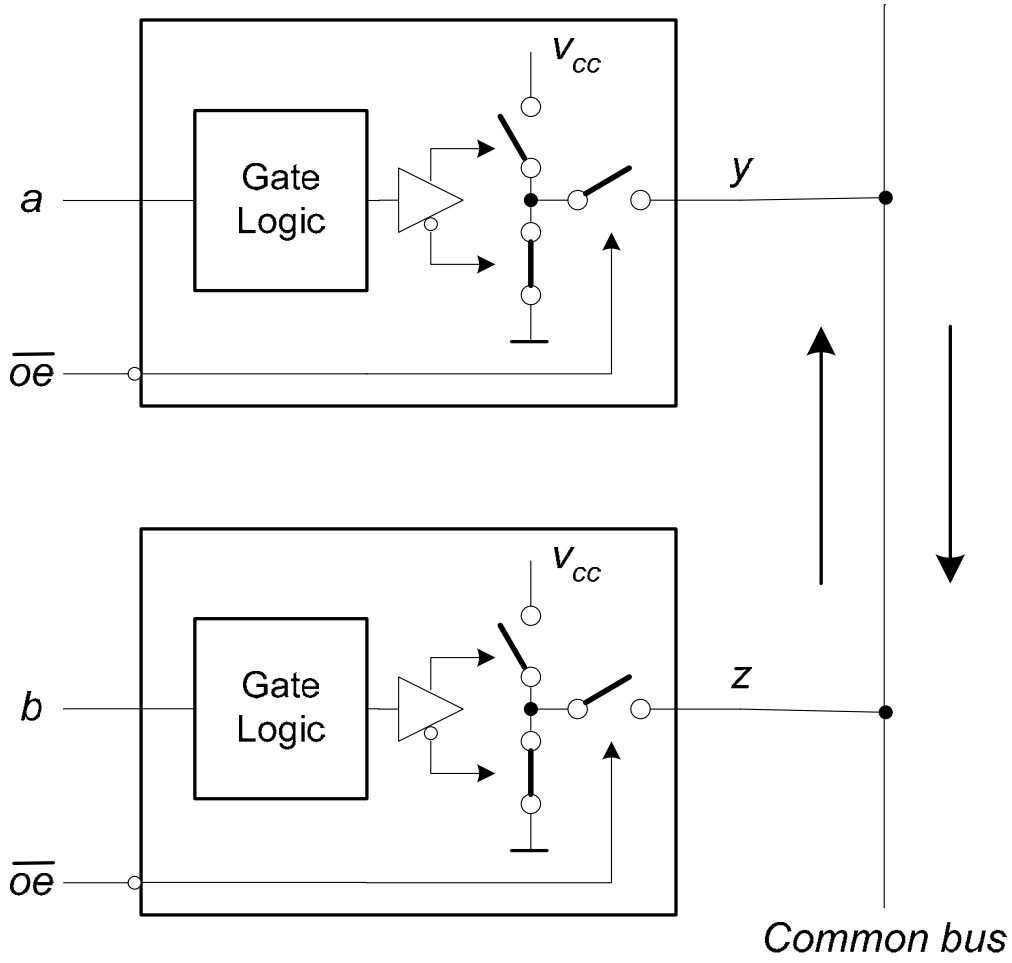
Dvoustavový výstup (Totem-pole output)



Otevřený kolektor (Open-collector output, OC)

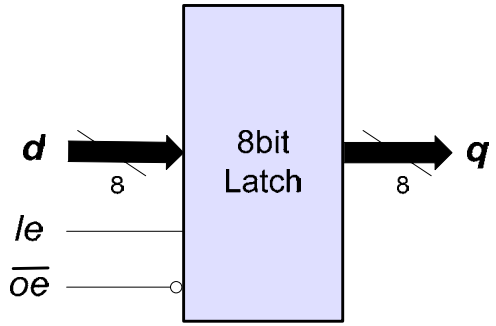


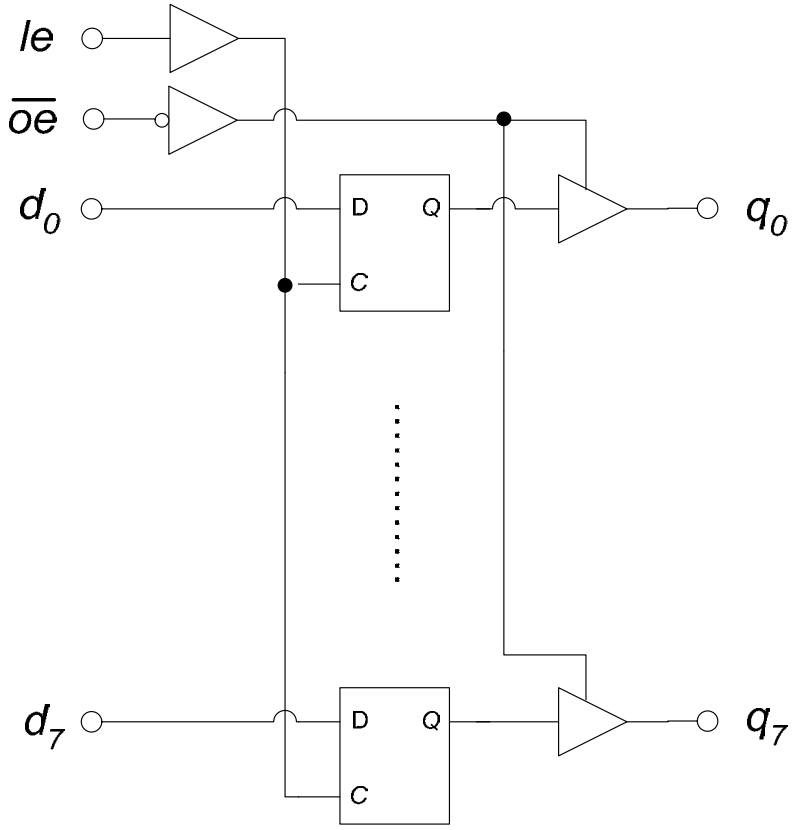
Třístavový výstup (Tri-state output, TS)



8bitový záchytný registr s třístavovým výstupem (Latch, TS)

- 8-bit D–type latch with tri-state outputs (TS – output)





D Latch with TS output

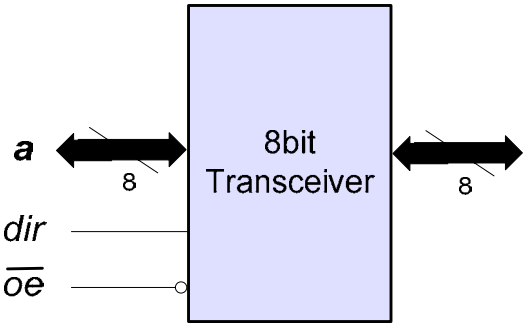
$\overline{OE}$	LE	D^i	Q^{i+1}
0	1	0	0
0	1	1	1
0	1	X	Q^i
1	X	X	Z

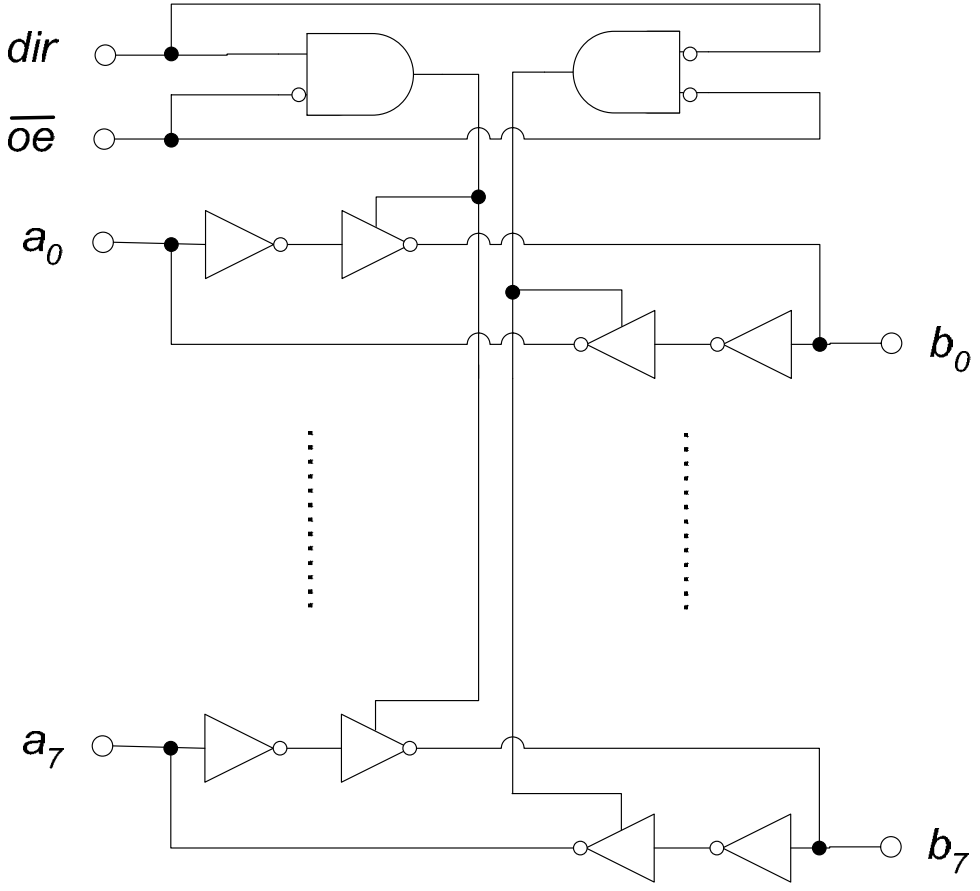
X – nezáleží

Z – odpojeno

8bitový obousměrný budič sběrnice (Transceiver)

- 8-bit bus transceiver with tri-state outputs





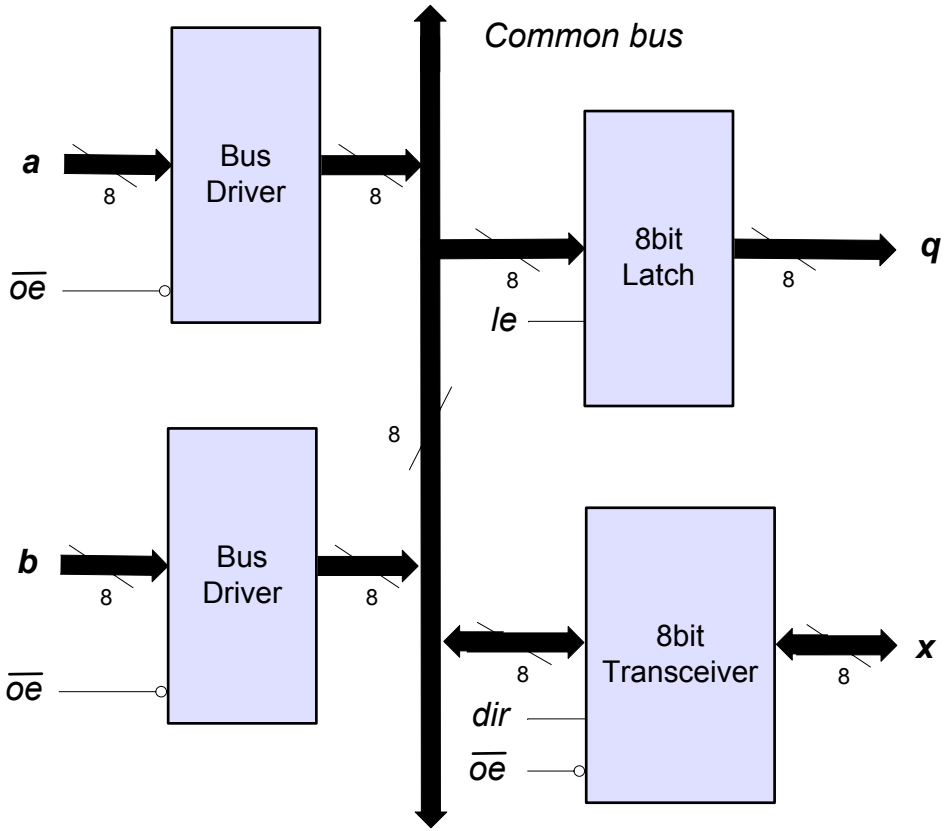
Bus Transceiver

$\overline{OE}$	DIR	A port	B port
0	0	A → B	Z
0	1	Z	B → A
1	X	Z	Z

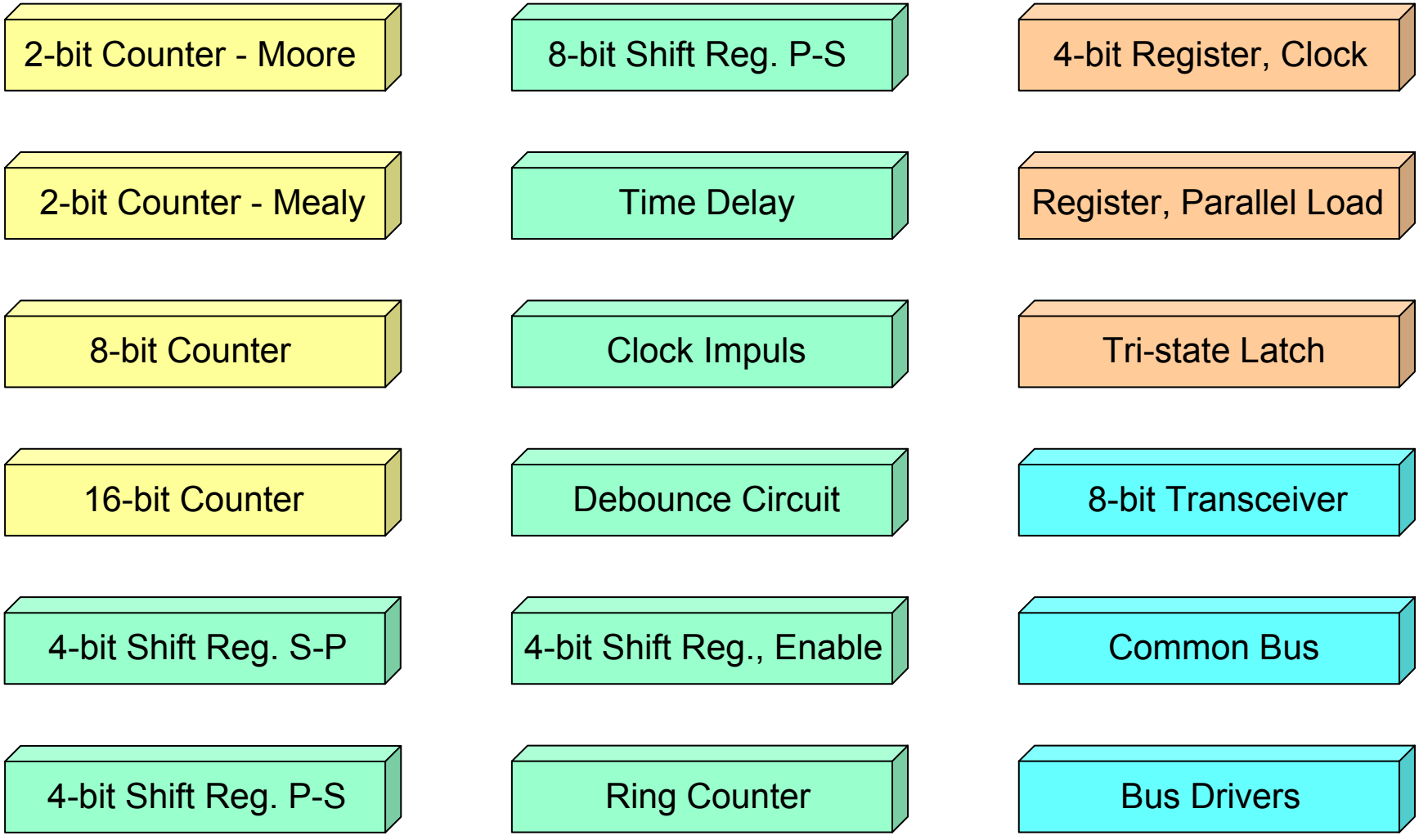
Z – odpojeno

Společná sběrnice (Bus, Common Bus)

- Použití: propojení komunikujících bloků počítače
- Sběrnice jednosměrná nebo obousměrná
- Sběrnice s třístavovými budiči nebo s budiči s otevřeným kolektorem



Přehled sekvenčních bloků a budičů dle kategorie



Čítače (Counter)

- Speciální typ registru – zahrnuje funkce přičítaní nebo odčítání

- Čítače čítají (odčítají) modulo M
  - Úplné čítače čítají modulo 2^n (tj. čítají do 4, 8, 16,……)
  - Neúplné čítače čítají např. do 5, 10, 13, 55, 80, ….

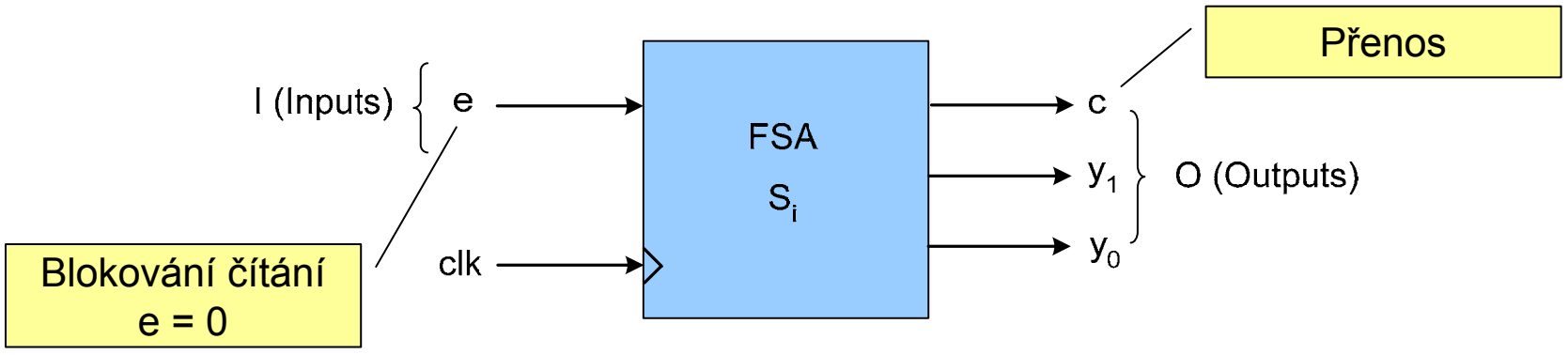
- Čítače čítají:
  - v binární kódu
  - v Grayově kódu (mění se vždy jen jedna stavová proměnná)
  - v Johansonově kódu (mění se vždy jen jedna stavová proměnná)
  - v dalších kódech

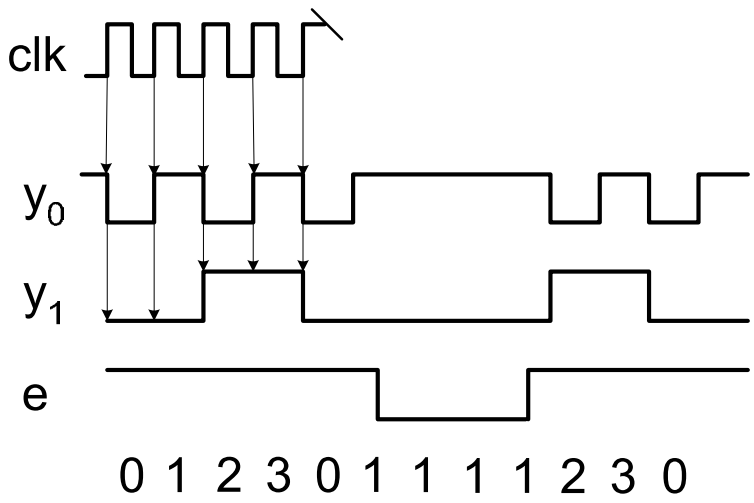
- Čítače jsou:
  - Synchronní – stavové klopné obvody mají společné hodiny
  - Asynchronní – výstup jednoho klopného obvodu tvoří hodiny následujícího klopného obvodu

Synchronní 2bitový binární čítač s blokováním

Moore

Navrhněte synchronní konečný automat (FSA – Finite State Automaton, **Moore**) typu čítač. Čítač čítá v binárním kódu, je 2bitový a má blokování čítání. V zapojení použijte půlsčítačku (Half Adder). Automat navrhněte s asynchronním nulováním.





Synchronní 2bitový binární čítač s blokováním

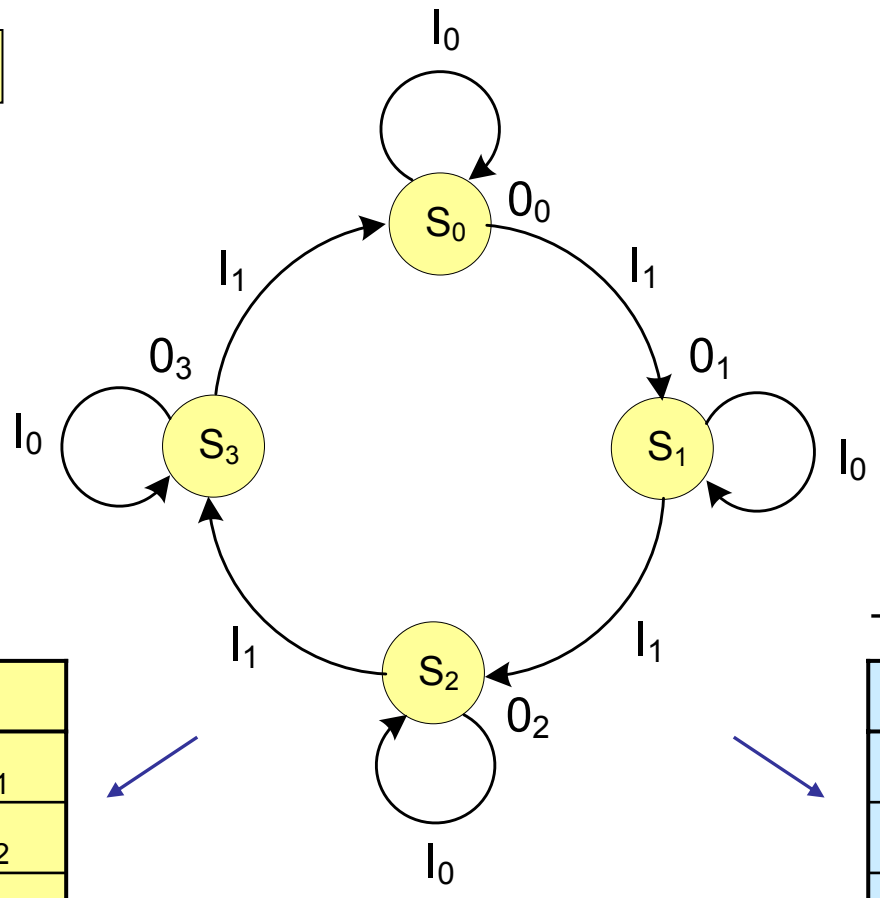
Moore

Stavový diagram

I – Vstupy (Inputs)

O – Výstupy (Outputs)

S_i – i-tý stav



Tabulka přechodů

S_i	I_0	I_1
S_0	S_0	S_1
S_1	S_1	S_2
S_2	S_2	S_3
S_3	S_3	S_0

Tabulka výstupů

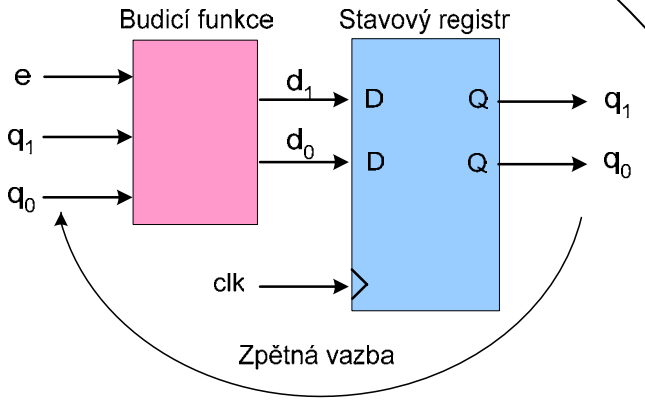
S_i	O_i
S_0	O_0
S_1	O_1
S_2	O_2
S_3	O_3

Synchronní 2bitový binární čítač s blokováním

Moore

Tabulka přechodů

S_i	I_0	I_1
S_0	S_0	S_1
S_1	S_1	S_2
S_2	S_2	S_3
S_3	S_3	S_0



Kódování stavů

S_i	e	q_1	q_0	d_1	d_0	S_{i+1}
S_0	0	0	0	0	0	S_0
S_1	0	0	1	0	1	S_1
S_2	0	1	0	1	0	S_2
S_3	0	1	1	1	1	S_3
S_0	1	0	0	0	1	S_1
S_1	1	0	1	1	0	S_2
S_2	1	1	0	1	1	S_3
S_3	1	1	1	0	0	S_0

Synchronní 2bitový binární čítač s blokováním

Moore

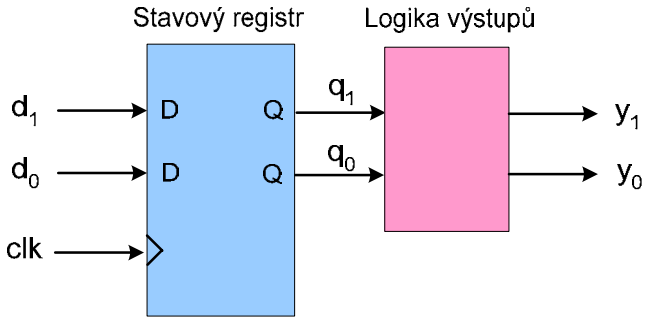
Tabulka výstupů

S_i	O_i
S_0	O_0
S_1	O_1
S_2	O_2
S_3	O_3

S_i O

Kódování výstupů

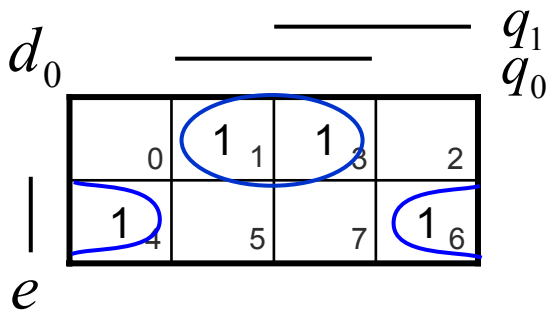
S_i	q_1	q_0	y_1	y_0	c	O_i
S_0	0	0	0	0	0	O_0
S_1	0	1	0	1	0	O_1
S_2	1	0	1	0	0	O_2
S_3	1	1	1	1	1	O_3



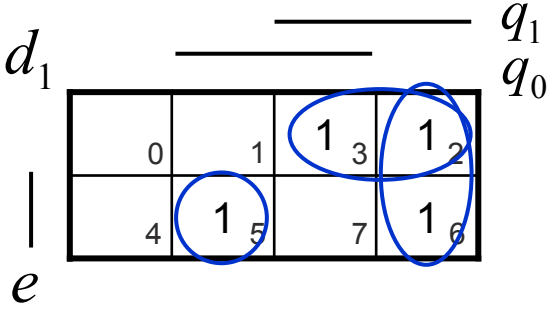
Synchronní 2bitový binární čítač s blokováním

Moore

Minimalizace



$$d_0 = e\overline{q_0} + \overline{e}q_0 = e \oplus q_0$$



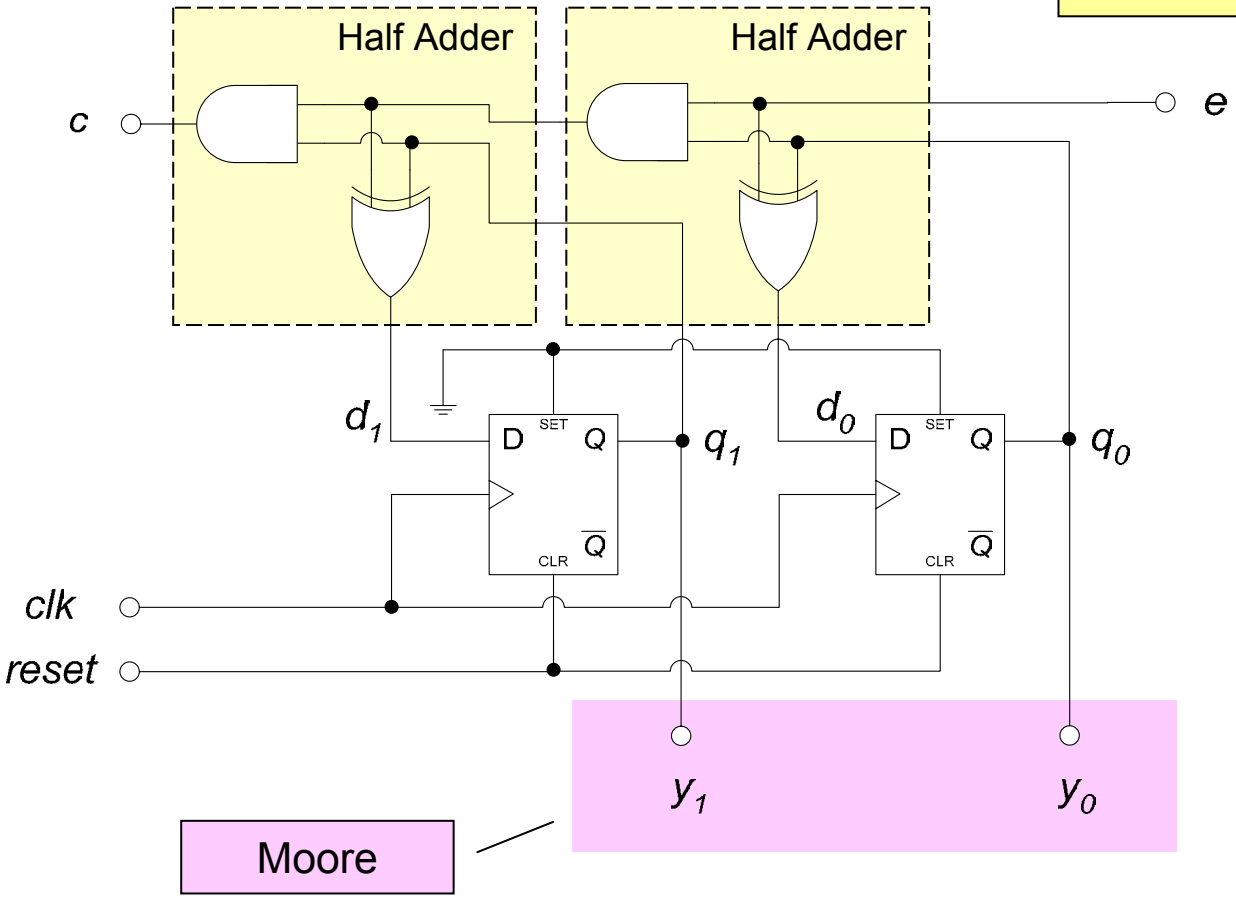
$$d_1 = q_1\overline{q_0} + \overline{e}q_1 + e\overline{q_1}q_0 =$$
$$= q_1(\overline{q_0} + \overline{e}) + e\overline{q_1}q_0 =$$
$$= q_1(\overline{eq_0}) + \overline{q_1}(eq_0) =$$
$$= q_1 \oplus (eq_0)$$

$$y_0 = q_0 \qquad y_1 = q_1 \qquad c = e\,q_1\,q_0$$

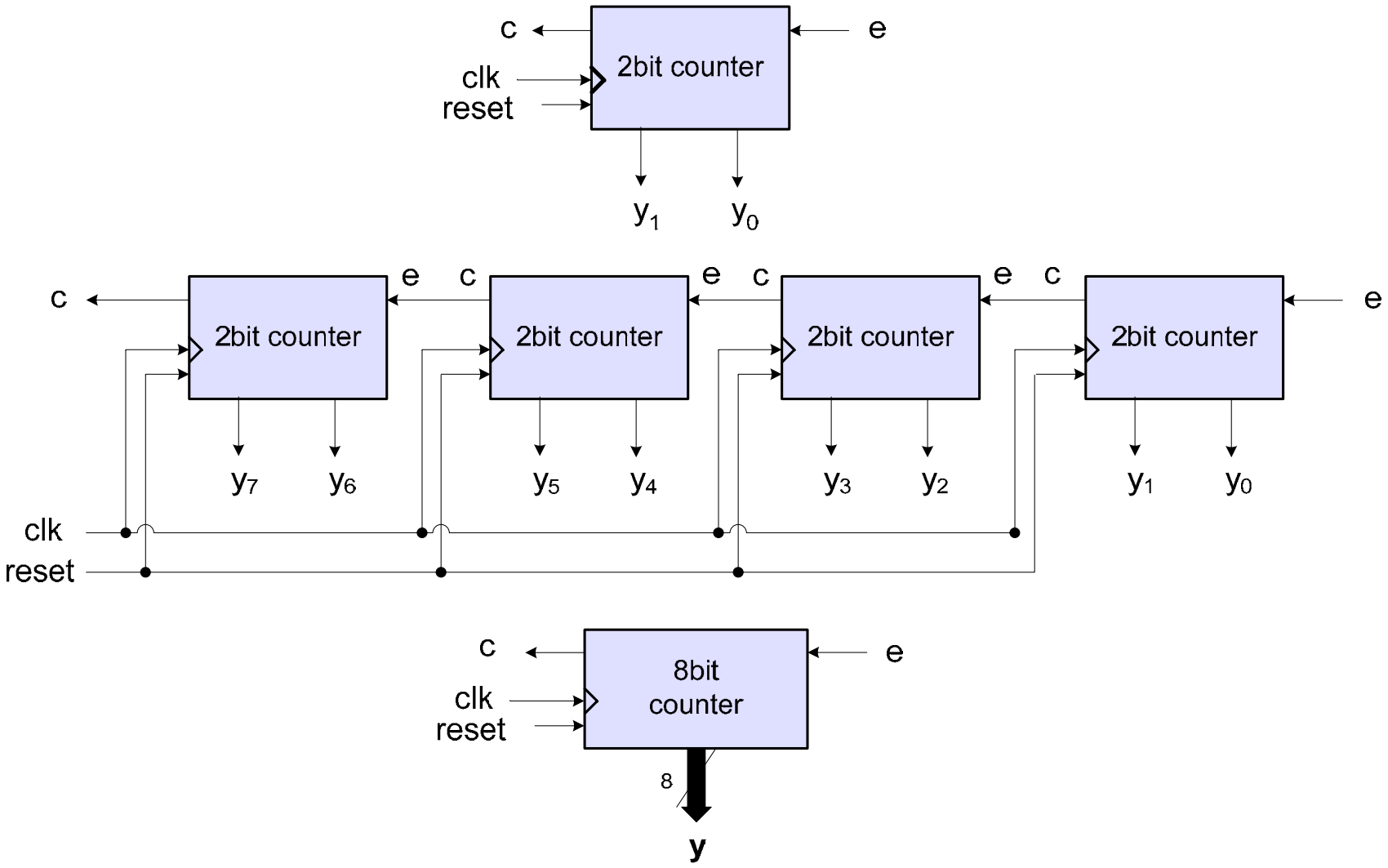
Synchronní 2bitový binární čítač s blokováním

Moore

Realizace



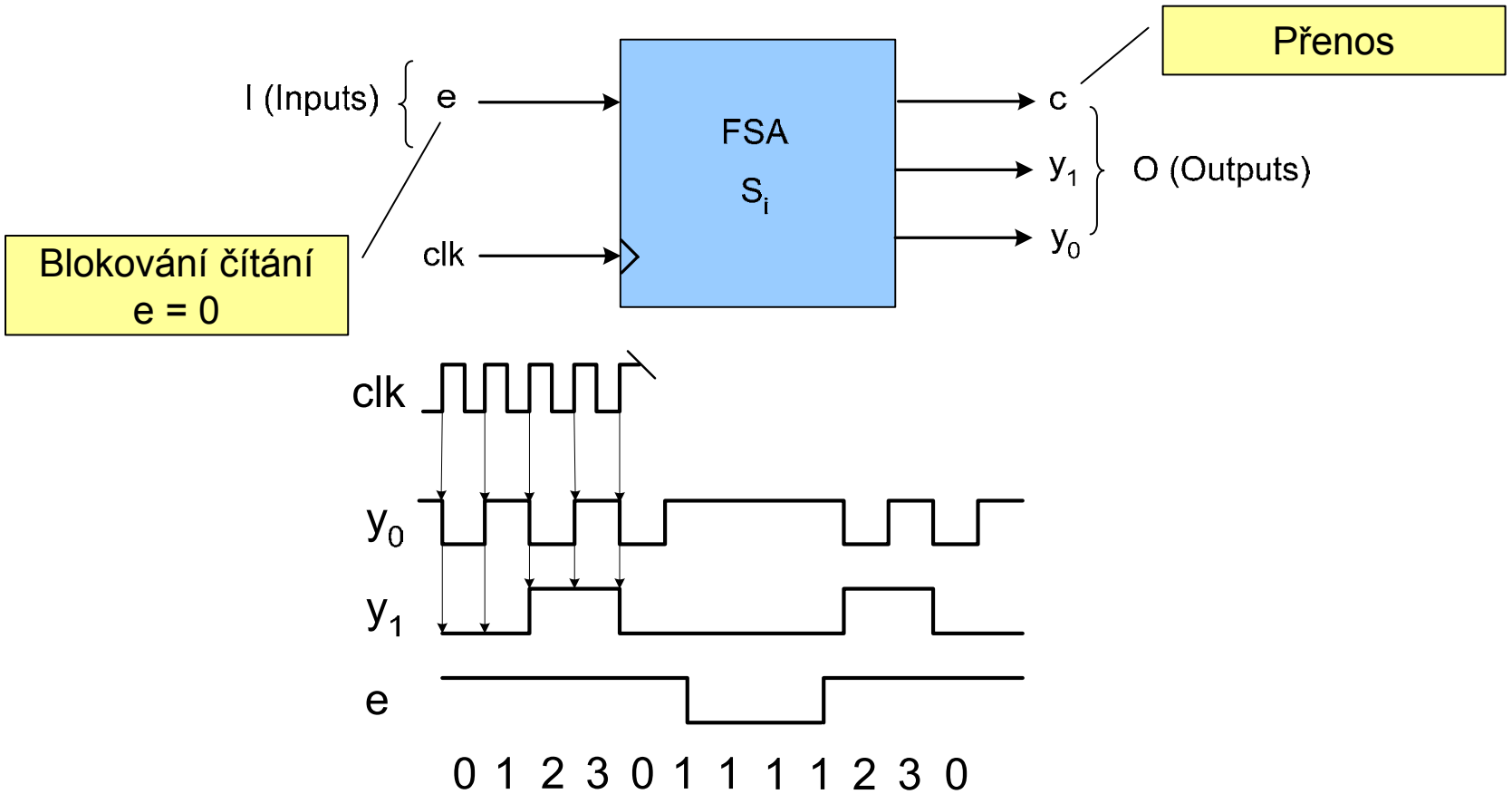
Synchronní 8bitový binární čítač s blokováním



Synchronní 2bitový binární čítač s blokováním

Mealy

Navrhněte synchronní konečný automat (FSA – Finite State Automaton, **Mealy**) typu čítač. Čítač čítá v binárním kódu, je 2bitový a má blokování čítaní. V zapojení použijte půlsčítačku (Half Adder). Automat navrhněte s asynchronním nulováním.



Synchronní 2bitový binární čítač s blokováním

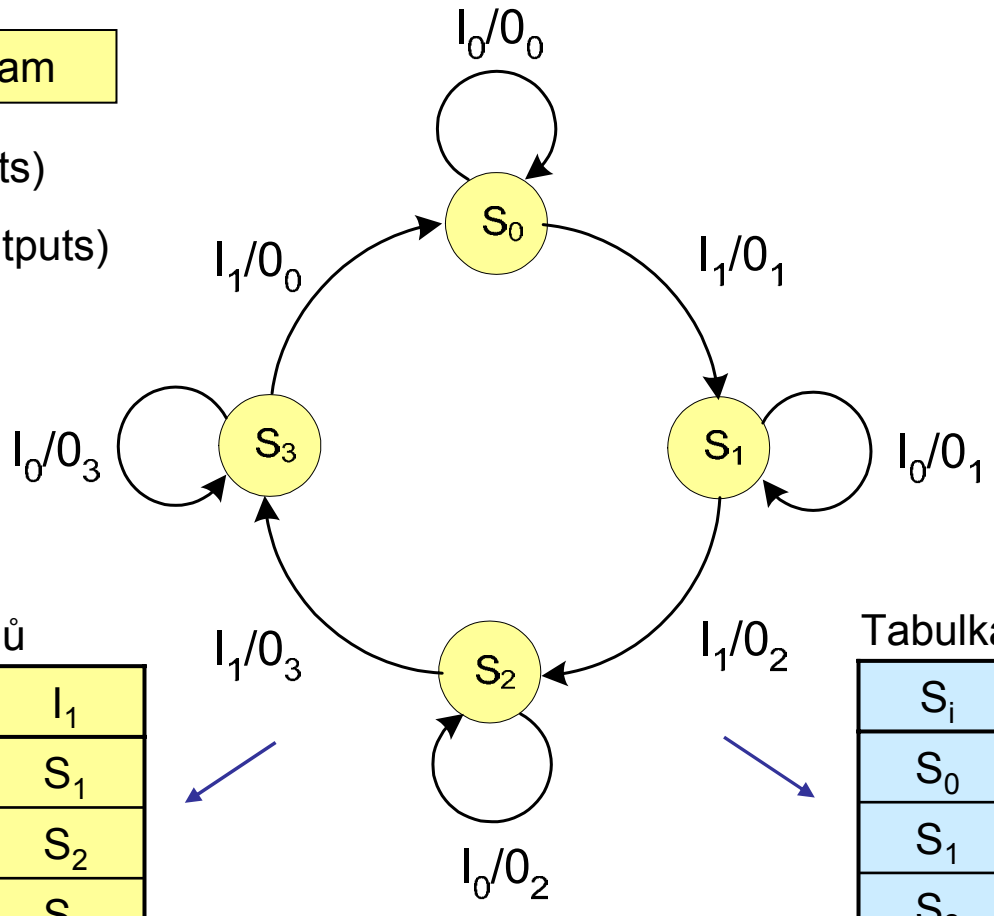
Mealy

Stavový diagram

I – Vstupy (Inputs)

O – Výstupy (Outputs)

S_i – i-tý stav



Tabulka přechodů

S_i	I_0	I_1
S_0	S_0	S_1
S_1	S_1	S_2
S_2	S_2	S_3
S_3	S_3	S_0

Tabulka výstupů

S_i	I_0	I_1
S_0	O_0	O_1
S_1	O_1	O_2
S_2	O_2	O_3
S_3	O_3	O_0

Synchronní 2bitový binární čítač s blokováním

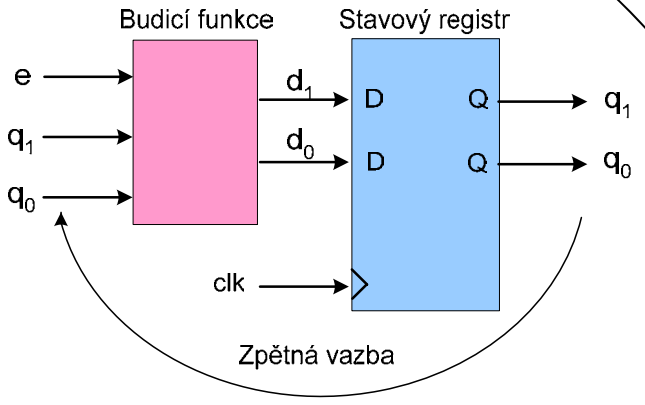
Mealy

Tabulka přechodů

S_i	I_0	I_1
S_0	S_0	S_1
S_1	S_1	S_2
S_2	S_2	S_3
S_3	S_3	S_0

Kódování stavů

S_i	e	q_1	q_0	d_1	d_0	S_{i+1}
S_0	0	0	0	0	0	S_0
S_1	0	0	1	0	1	S_1
S_2	0	1	0	1	0	S_2
S_3	0	1	1	1	1	S_3
S_0	1	0	0	0	1	S_1
S_1	1	0	1	1	0	S_2
S_2	1	1	0	1	1	S_3
S_3	1	1	1	0	0	S_0

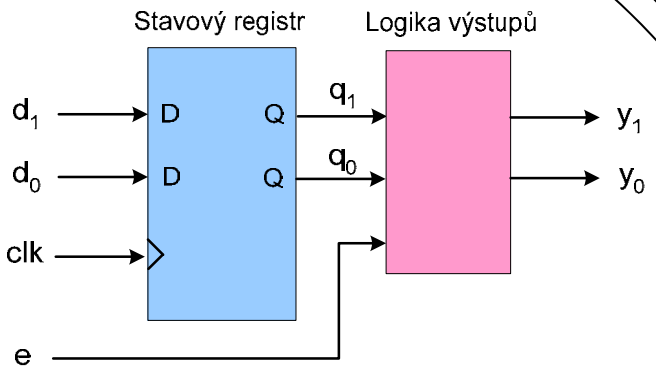


Synchronní 2bitový binární čítač s blokováním

Mealy

Tabulka výstupů

S_i	I_0	I_1
S_0	O_0	O_1
S_1	O_1	O_2
S_2	O_2	O_3
S_3	O_3	O_0



S_i O_i

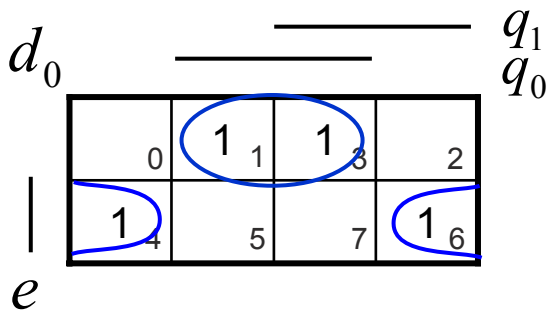
Kódování výstupů

S_i	e	q_1	q_0	y_1	y_0	O_i
S_0	0	0	0	0	0	O_0
$S_!$	0	0	1	0	1	$O_!$
S_2	0	1	0	1	0	O_2
S_3	0	1	1	1	1	O_3
S_0	1	0	0	0	1	O_1
$S_!$	1	0	1	1	0	O_2
S_2	1	1	0	1	1	O_3
S_3	1	1	1	0	0	O_0

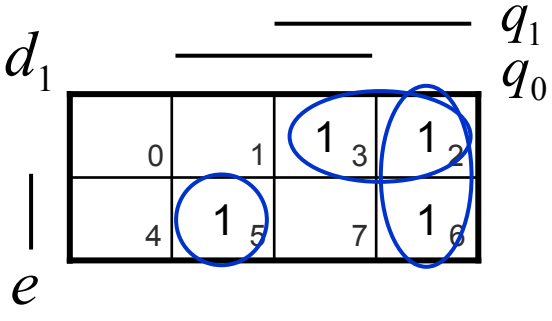
Synchronní 2bitový binární čítač s blokováním

Mealy

Minimalizace



$$d_0 = e\overline{q_0} + \overline{e}q_0 = e \oplus q_0$$



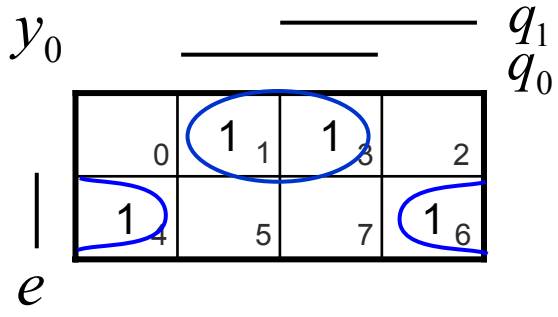
$$d_1 = q_1\overline{q_0} + \overline{e}q_1 + e\overline{q_1}q_0 =$$
$$= q_1(\overline{q_0} + \overline{e}) + e\overline{q_1}q_0 =$$
$$= q_1(\overline{eq_0}) + \overline{q_1}(eq_0) =$$
$$= q_1 \oplus (eq_0)$$

$$c = eq_1q_0$$

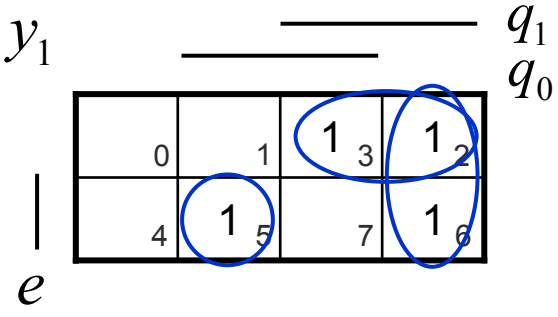
Synchronní 2bitový binární čítač s blokováním

Mealy

Minimalizace



$$y_0 = e\overline{q_0} + \overline{e}q_0 = e \oplus q_0$$



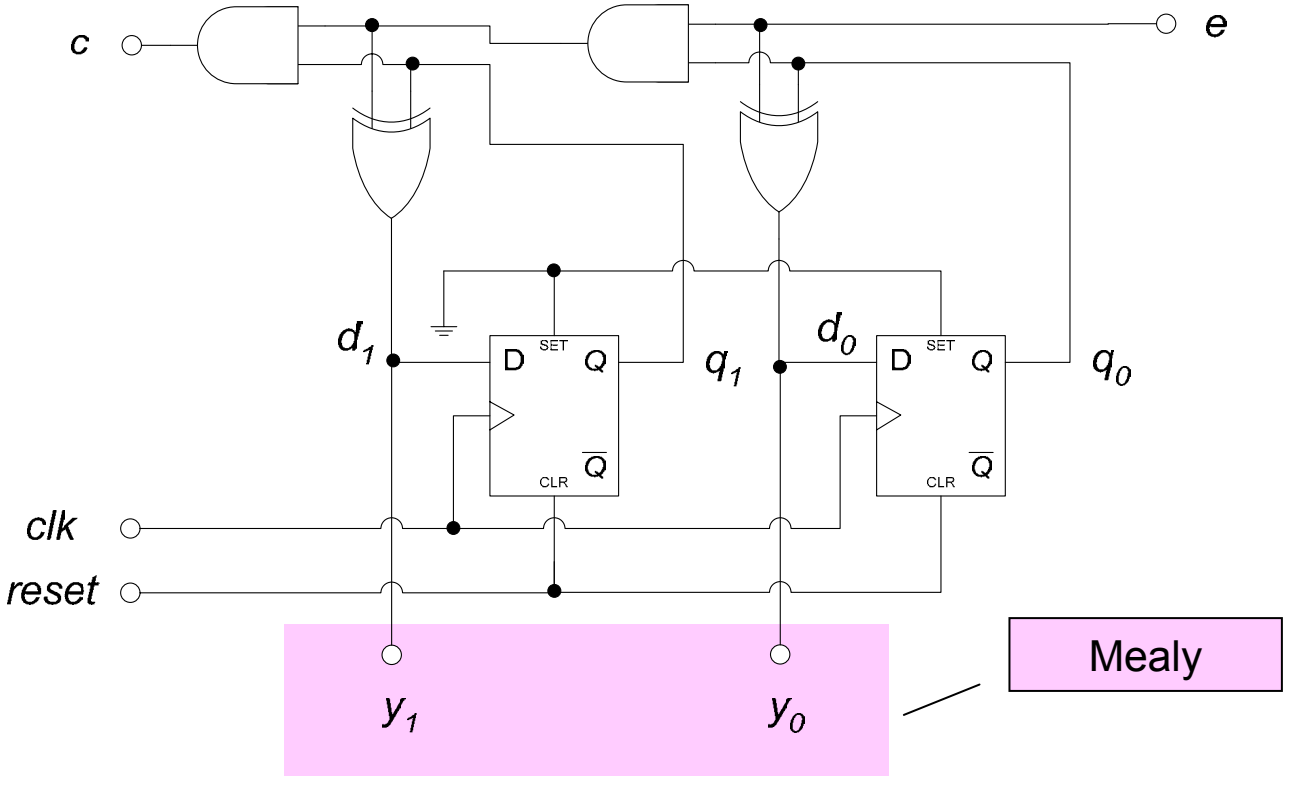
$$y_1 = q_1\overline{q_0} + \overline{e}q_1 + e\overline{q_1}q_0 =$$
$$= q_1(\overline{q_0} + \overline{e}) + e\overline{q_1}q_0 =$$
$$= q_1\overline{(eq_0)} + \overline{q_1}(eq_0) =$$
$$= q_1 \oplus (eq_0)$$

$$c = eq_1q_0$$

Synchronní 2bitový binární čítač s blokováním

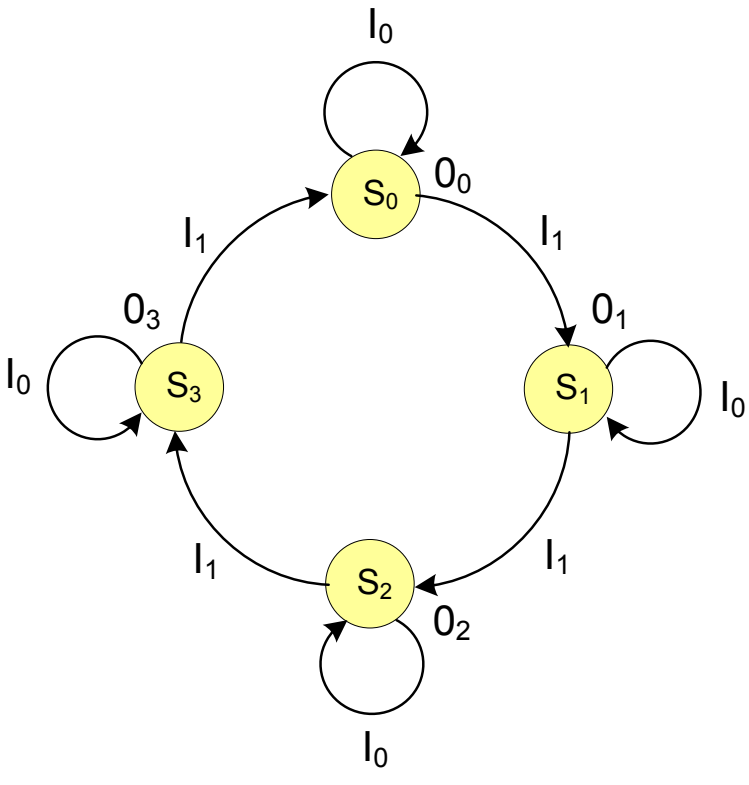
Mealy

Realizace



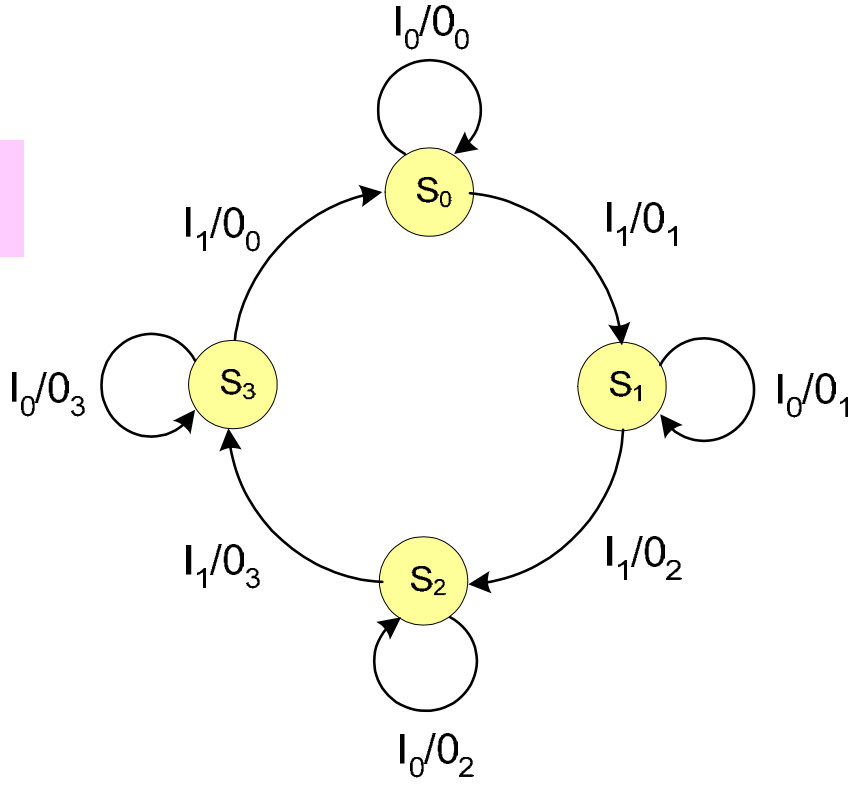
Synchronní 2bitový binární čítač s blokováním (porovnání)

Moore



≠

Mealy



Synchronní 2bitový binární čítač s blokováním (porovnání)

Moore

Tabulka přechodů

S_i	I_0	I_1
S_0	S_0	S_1
S_1	S_1	S_2
S_2	S_2	S_3
S_3	S_3	S_0

Tabulka výstupů

S_i	O_i
S_0	O_0
S_1	O_1
S_2	O_2
S_3	O_3

$\equiv$

$\neq$

Mealy

Tabulka přechodů

S_i	I_0	I_1
S_0	S_0	S_1
S_1	S_1	S_2
S_2	S_2	S_3
S_3	S_3	S_0

Tabulka výstupů

S_i	I_0	I_1
S_0	O_0	O_1
S_1	O_1	O_2
S_2	O_2	O_3
S_3	O_3	O_0

Synchronní 2bitový binární čítač s blokováním (porovnání)

Moore

S_i S_{i+1}

Kódování stavů

S_i	e	q_1	q_0	d_1	d_0	S_{i+1}
S_0	0	0	0	0	0	S_0
S_1	0	0	1	0	1	S_1
S_2	0	1	0	1	0	S_2
S_3	0	1	1	1	1	S_3
S_0	1	0	0	0	1	S_1
S_1	1	0	1	1	0	S_2
S_2	1	1	0	1	1	S_3
S_3	1	1	1	0	0	S_0

≡

Mealy

S_i S_{i+1}

Kódování stavů

S_i	e	q_1	q_0	d_1	d_0	S_{i+1}
S_0	0	0	0	0	0	S_0
S_1	0	0	1	0	1	S_1
S_2	0	1	0	1	0	S_2
S_3	0	1	1	1	1	S_3
S_0	1	0	0	0	1	S_1
S_1	1	0	1	1	0	S_2
S_2	1	1	0	1	1	S_3
S_3	1	1	1	0	0	S_0

Synchronní 2bitový binární čítač s blokováním (porovnání)

Moore

S_i O

Kódování výstupů

S_i	q_1	q_0	y_1	y_0	c	O_i
S_0	0	0	0	0	0	O_0
$S_!$	0	1	0	1	0	O_1
S_2	1	0	1	0	0	O_2
S_3	1	1	1	1	1	O_3

$\neq$

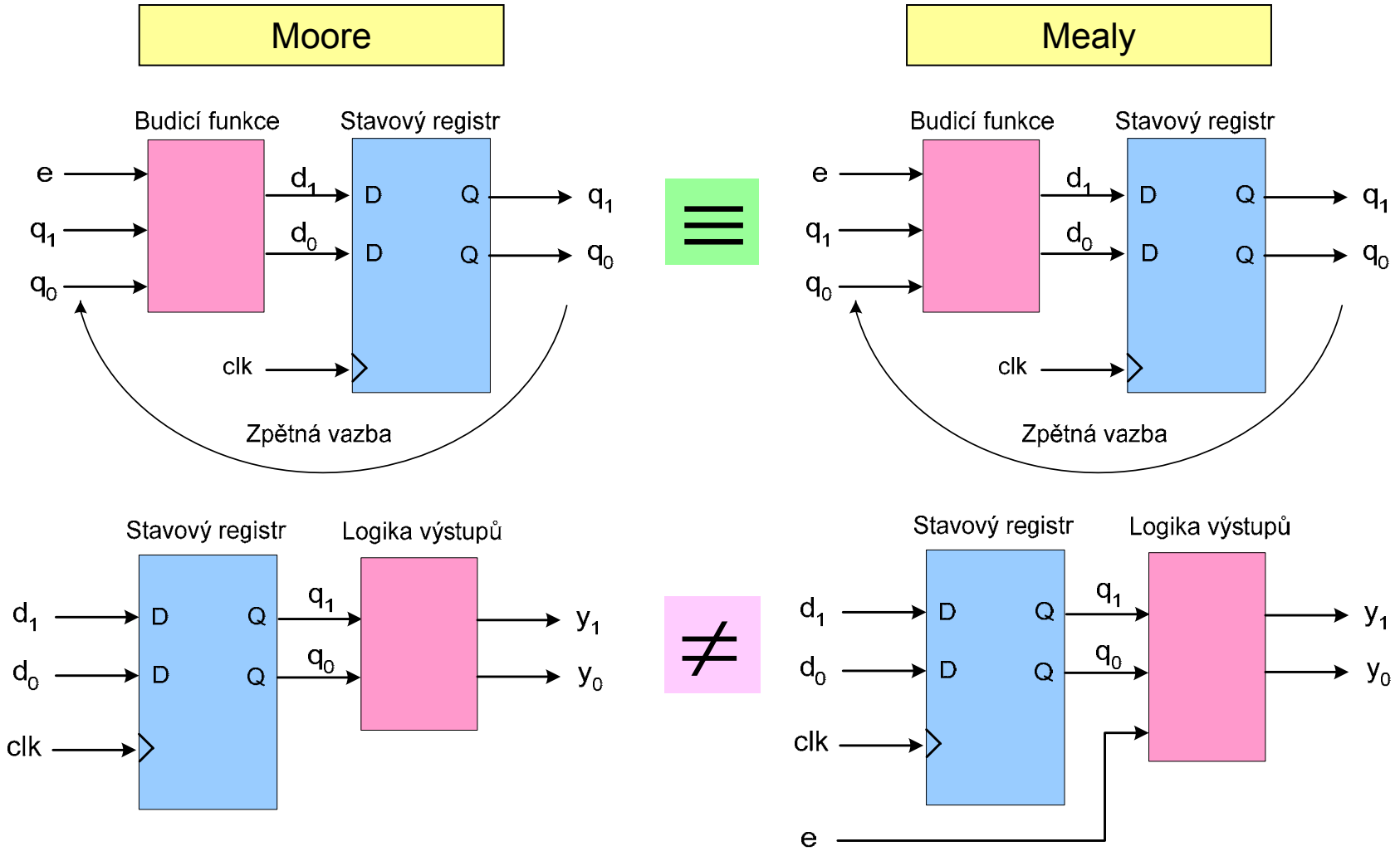
Mealy

S_i O_i

Kódování výstupů

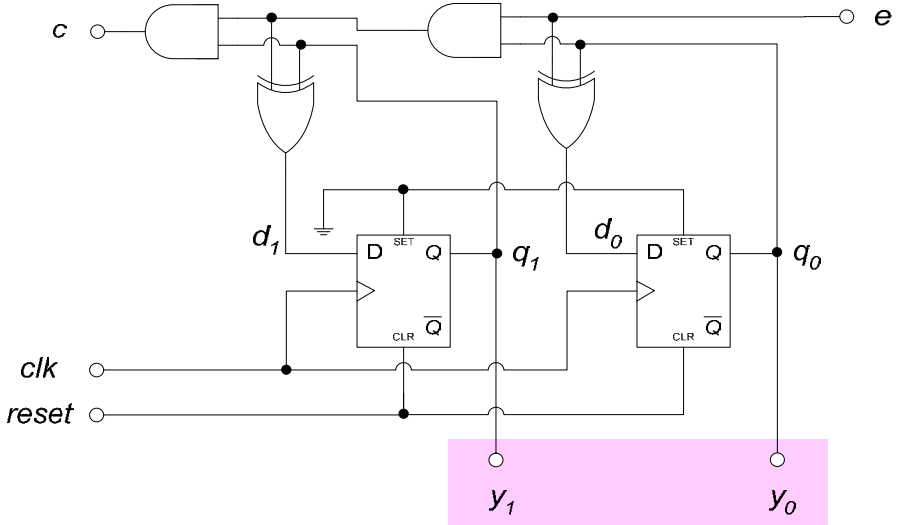
S_i	e	q_1	q_0	y_1	y_0	O_i
S_0	0	0	0	0	0	O_0
$S_!$	0	0	1	0	1	$O_!$
S_2	0	1	0	1	0	O_2
S_3	0	1	1	1	1	O_3
S_0	1	0	0	0	1	O_1
$S_!$	1	0	1	1	0	O_2
S_2	1	1	0	1	1	O_3
S_3	1	1	1	0	0	O_0

Synchronní 2bitový binární čítač s blokováním (porovnání)



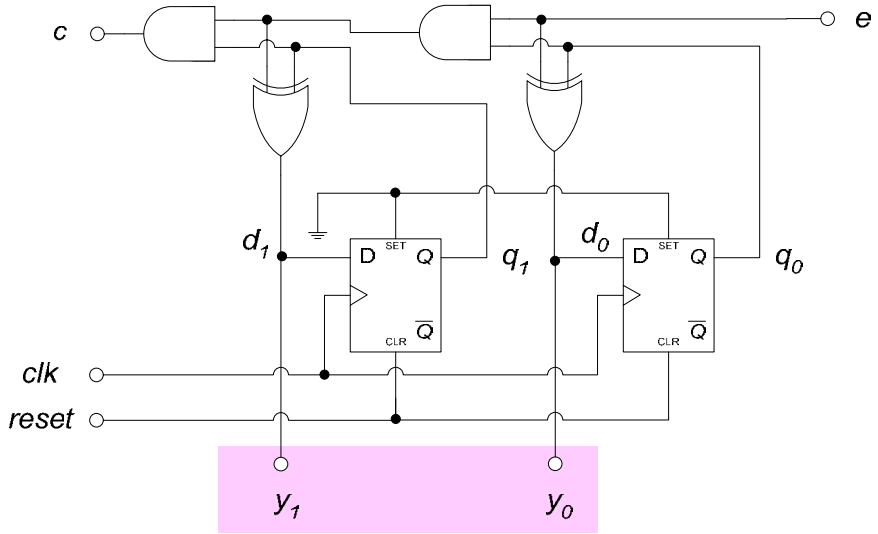
Synchronní 2bitový binární čítač s blokováním (porovnání)

Moore



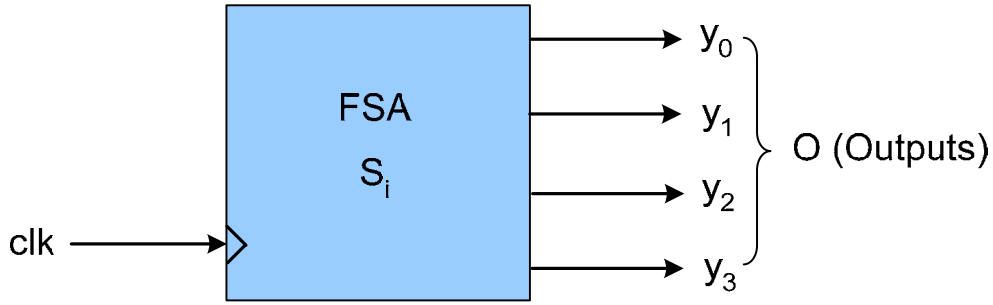
≠

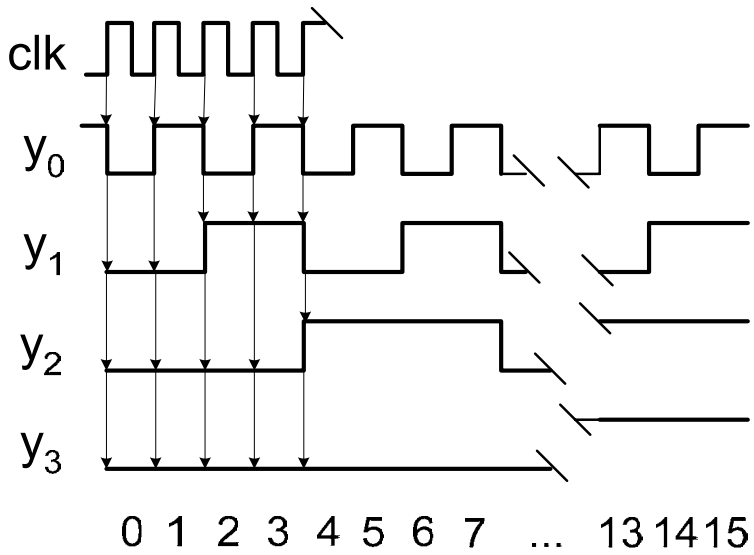
Mealy



Synchronní 4bitový binární čítač

Navrhněte synchronní konečný automat (FSA – Finite State Automaton) typu čítač. Čítač čítá v binárním kódu a je 4bitový. V zapojení použijte půlsčítačku (Half Adder). Automat navrhněte s asynchronním nulováním.





Synchronní 4bitový binární čítač

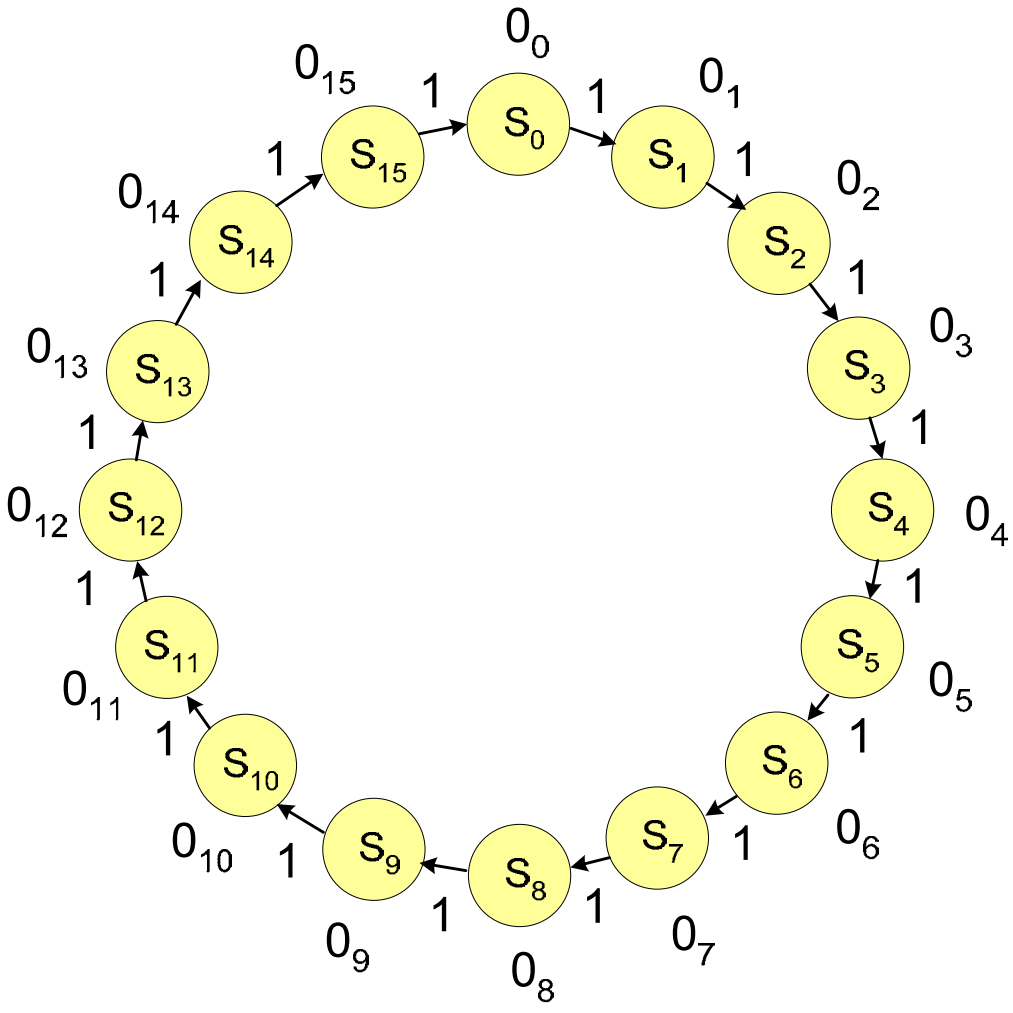
Moore

Stavový diagram

I – Vstupy (Inputs)

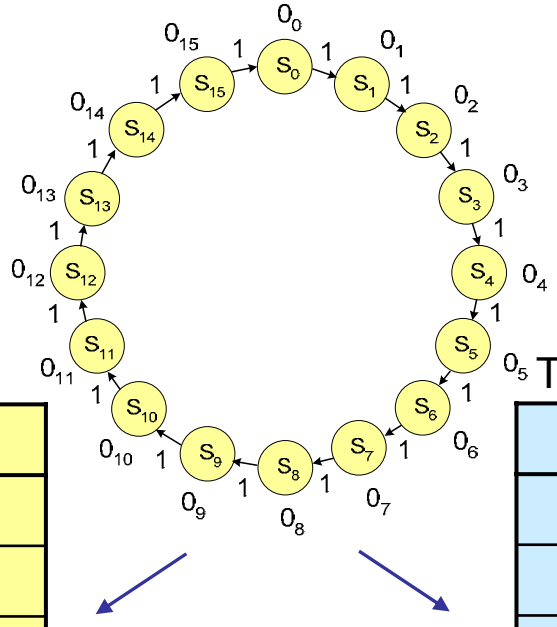
O – Výstupy (Outputs)

S_i – i-tý stav



Synchronní 4bitový binární čítač

Moore



Tabulka přechodů

S_i	1	S_i	1
S_0	S_1	S_8	S_9
S_1	S_2	S_9	S_{10}
S_2	S_3	S_{10}	S_{11}
S_3	S_4	S_{11}	S_{12}
S_4	S_5	S_{12}	S_{13}
S_5	S_6	S_{13}	S_{14}
S_6	S_7	S_{14}	S_{15}
S_7	S_8	S_{15}	S_0

Tabulka výstupů

S_i	O_i	S_i	O_i
S_0	O_0	S_8	O_8
S_1	O_1	S_9	O_9
S_2	O_2	S_{10}	O_{10}
S_3	O_3	S_{11}	O_{11}
S_4	O_4	S_{12}	O_{12}
S_5	O_5	S_{13}	O_{13}
S_6	O_6	S_{14}	O_{14}
S_7	O_7	S_{15}	O_{15}

Synchronní 4bitový binární čítač

Moore

Kódování stavů

S_i	q_3	q_2	q_1	q_0	d_3	d_2	d_1	d_0	S_{i+1}
0	0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	0	1	0	2
2	0	0	1	0	0	0	1	1	3
3	0	0	1	1	0	1	0	0	4
4	0	1	0	0	0	1	0	1	5
5	0	1	0	1	0	1	1	0	6
6	0	1	1	0	0	1	1	1	7
7	0	1	1	1	1	0	0	0	8
8	1	0	0	0	1	0	0	1	9
9	1	0	0	1	1	0	1	0	10
10	1	0	1	0	1	0	1	1	11
11	1	0	1	1	1	1	0	0	12
12	1	1	0	0	1	1	0	1	13
13	1	1	0	1	1	1	1	0	14
14	1	1	1	0	1	1	1	1	15
15	1	1	1	1	0	0	0	0	0

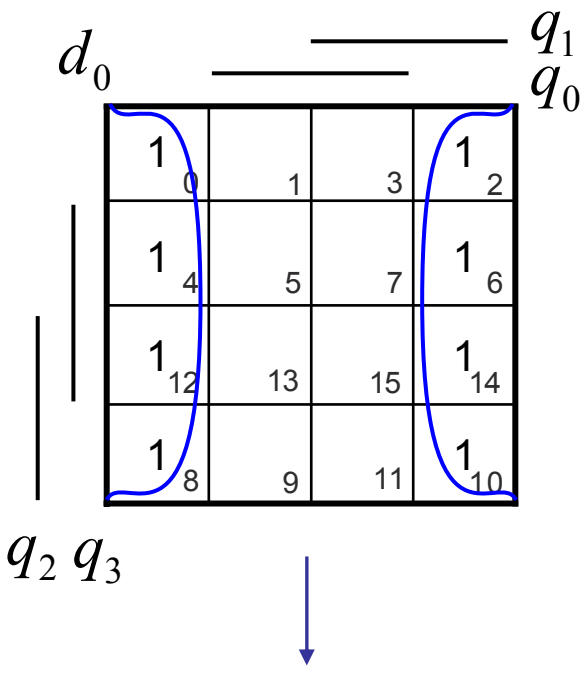
Kódování výstupů

S_i	q_3	q_2	q_1	q_0	y_3	y_2	y_1	y_0	O_i
0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	1
2	0	0	1	0	0	0	1	0	2
3	0	0	1	1	0	0	1	1	3
4	0	1	0	0	0	1	0	0	4
5	0	1	0	1	0	1	0	1	5
6	0	1	1	0	0	1	1	0	6
7	0	1	1	1	0	1	1	1	7
8	1	0	0	0	1	0	0	0	8
9	1	0	0	1	1	0	0	1	9
10	1	0	1	0	1	0	1	0	10
11	1	0	1	1	1	0	1	1	11
12	1	1	0	0	1	1	0	0	12
13	1	1	0	1	1	1	0	1	13
14	1	1	1	0	1	1	1	0	14
15	1	1	1	1	1	1	1	1	15

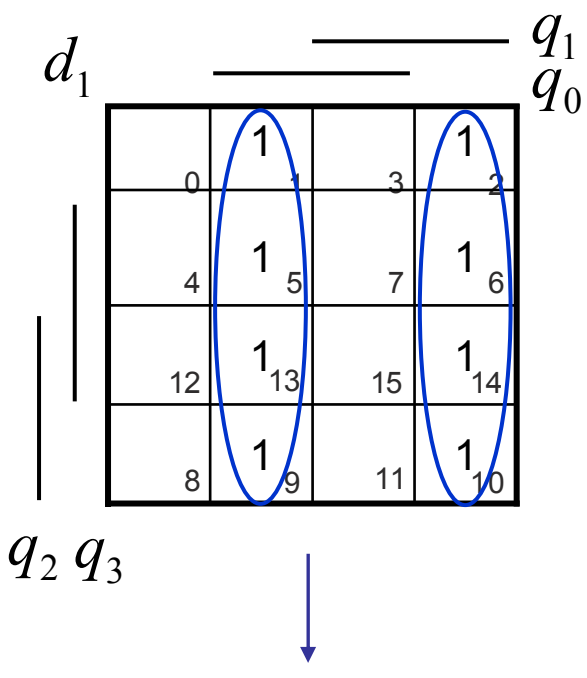
Synchronní 4bitový binární čítač

Moore

Minimalizace

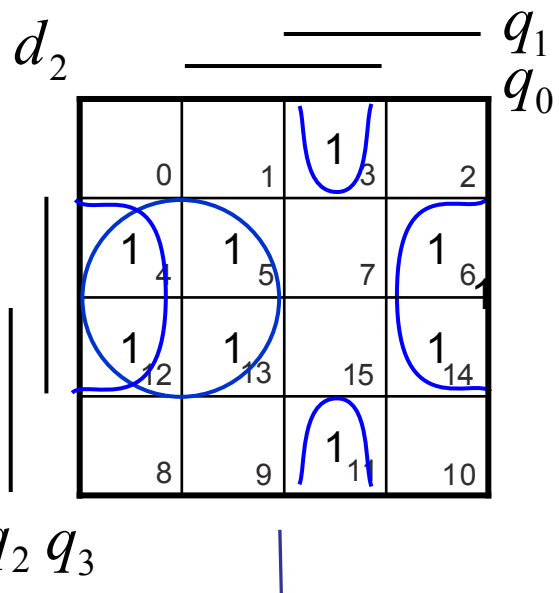


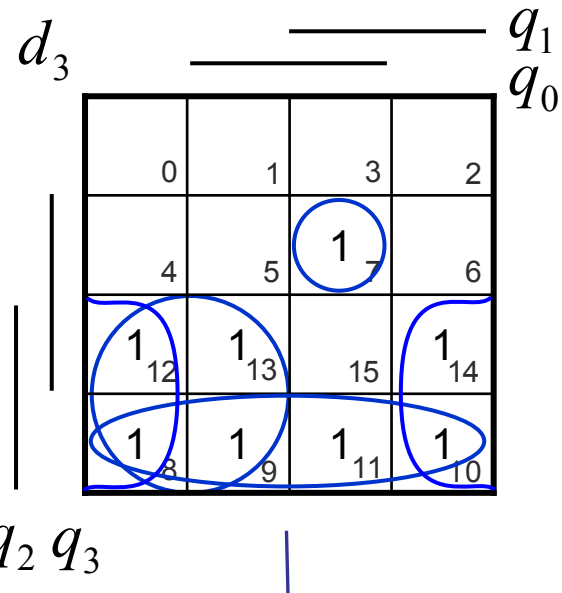
$$d_0 = \overline{q_0} = q_0 \oplus 1$$



$$d_1 = \overline{q_1}\, q_0 + q_1\, \overline{q_0} =$$
$$= q_1 \oplus q_0$$

Synchronní 4bitový binární čítač





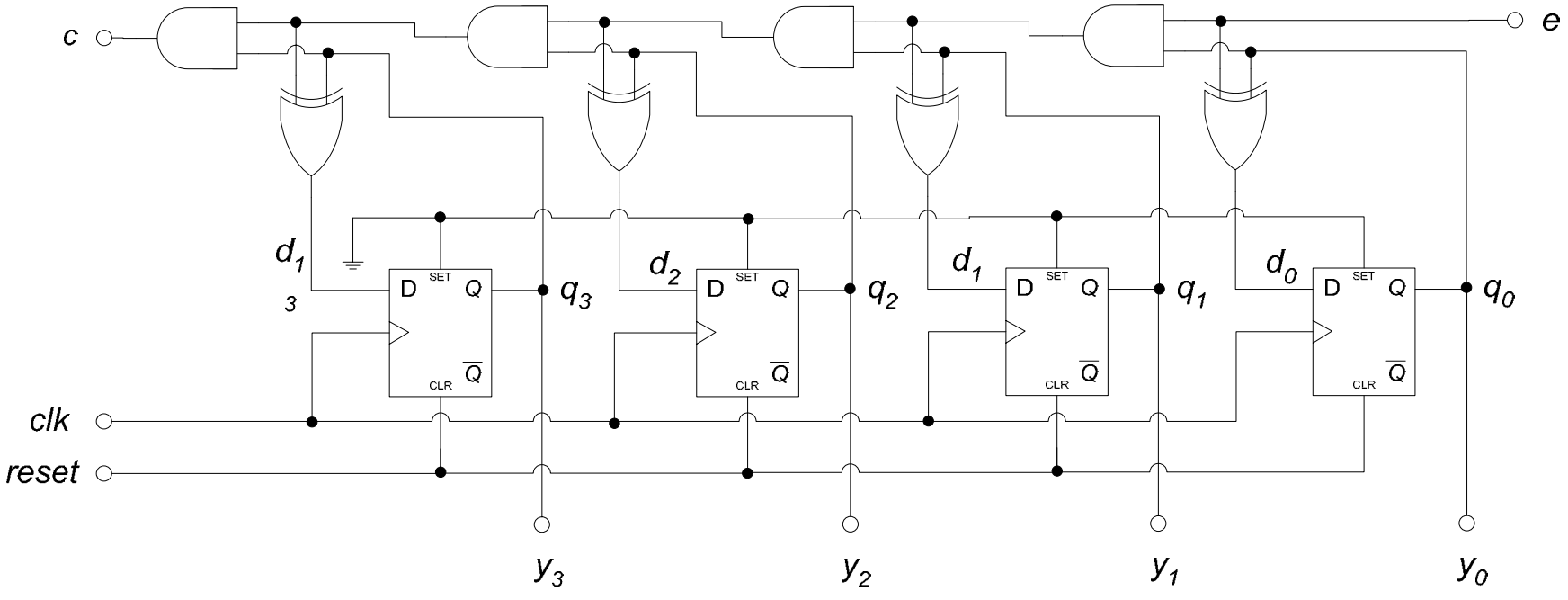
$$d_2 = q_2\overline{q_0} + q_2\overline{q_1} + \overline{q_2}\,q_1\,q_0 =$$
$$= q_2(\overline{q_0} + \overline{q_1}) + \overline{q_2}\,q_1\,q_0 =$$
$$= q_2\overline{(q_1\,q_0)} + \overline{q_2}(q_1\,q_0) =$$
$$= q_2 \oplus (q_1\,q_0)$$

$$d_3 = q_3\overline{q_2} + q_3\overline{q_1} + q_3\overline{q_0} + \overline{q_2}\,q_2\,q_1\,q_0 =$$
$$= q_3(\overline{q_2} + \overline{q_1} + \overline{q_0}) + \overline{q_3}(q_2\,q_1\,q_0) =$$
$$= q_3\overline{(q_2\,q_1\,q_0)} + \overline{q_3}(q_2\,q_1\,q_0) =$$
$$= q_3 \oplus (q_2\,q_1\,q_0)$$

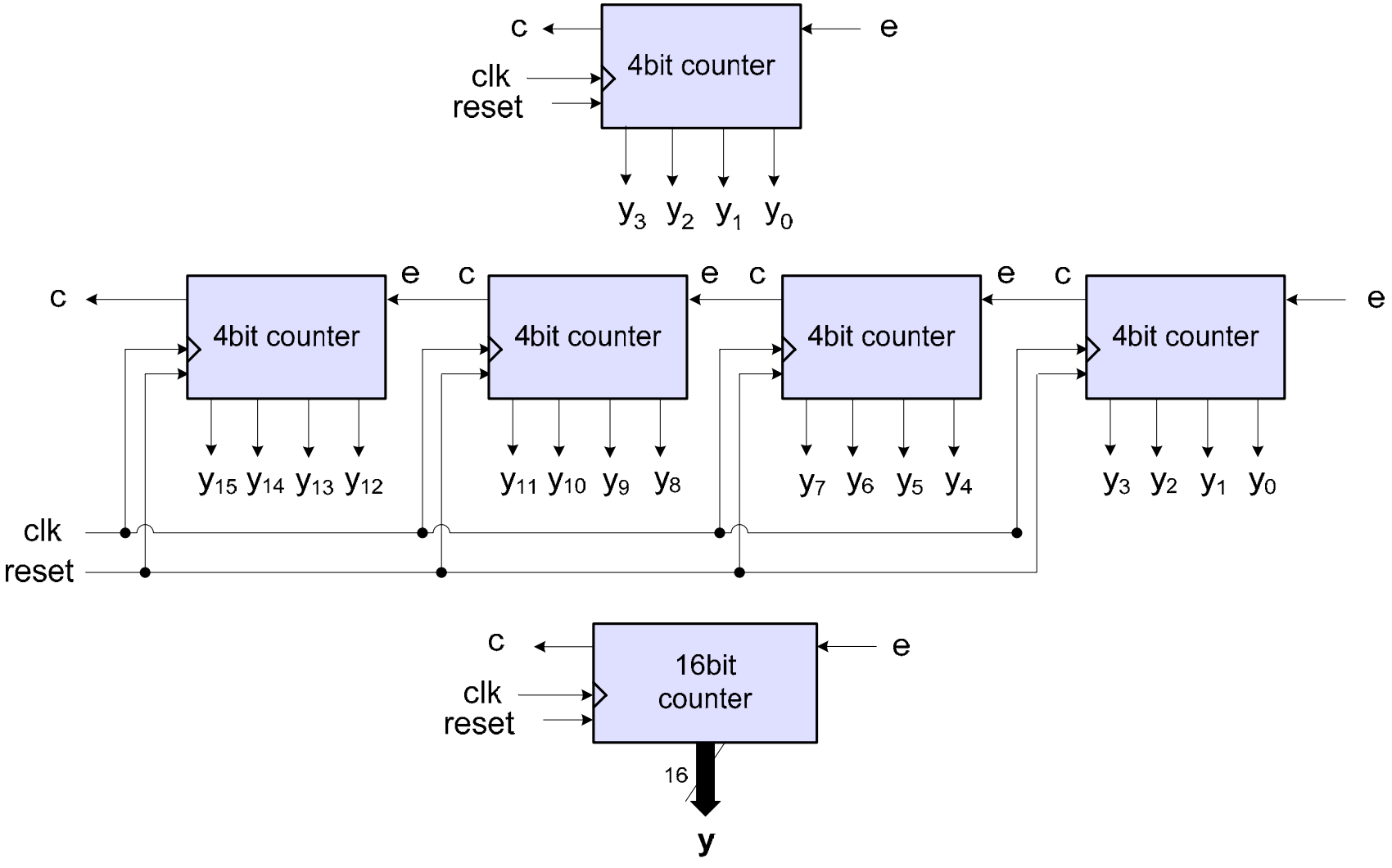
Synchronní 4bitový binární čítač

Moore

Realizace



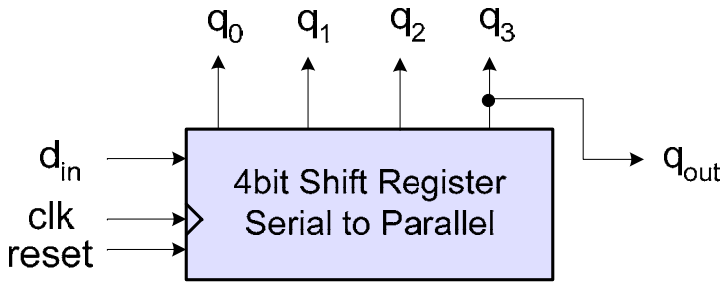
Synchronní 16bitový binární čítač

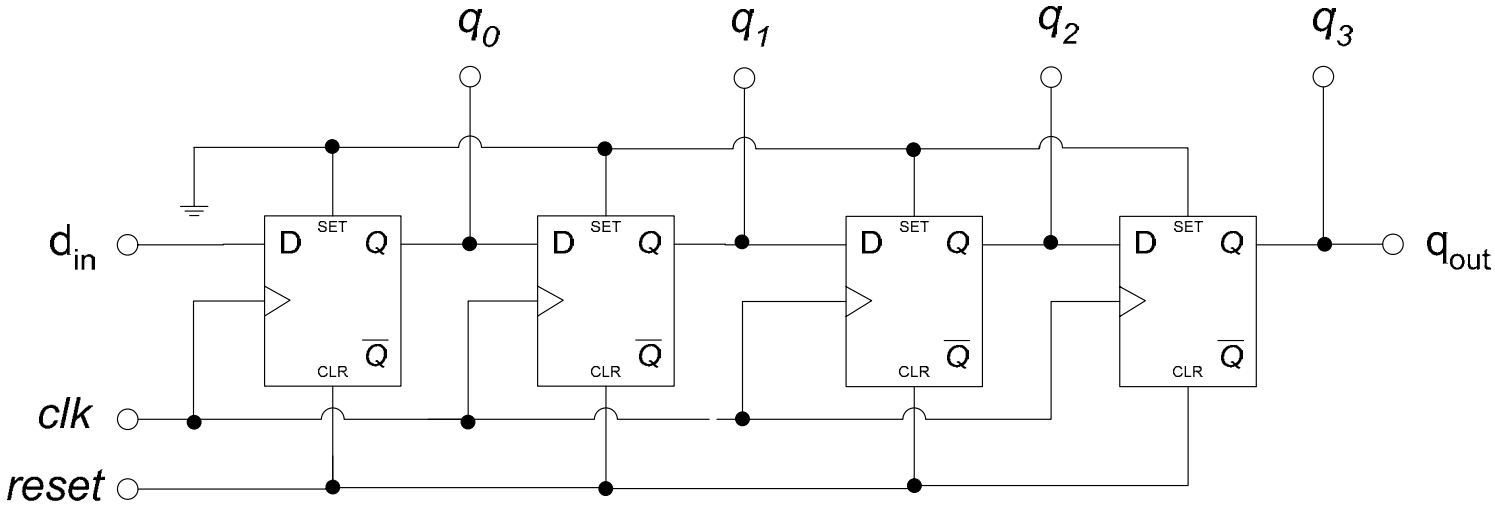


Posuvný registr (Shift Register)

- Použití:
  - Převod sériové informace na paralelní
    - Sériová komunikace - příjem
  - Převod paralelní informace na sériovou
    - Sériová komunikace – vysílání
  - Definované zpoždění signálu
  - Vícefázové hodiny pro řízení sekvenčních obvodů
    - Kruhový čítač (Ring Counter)
  - Převod hladinového signálu na impuls
    - Clock pulse circuit
  - Potlačení zákmitů mechanických tlačítek a spínačů
    - Debounce circuit
  - Další použití …

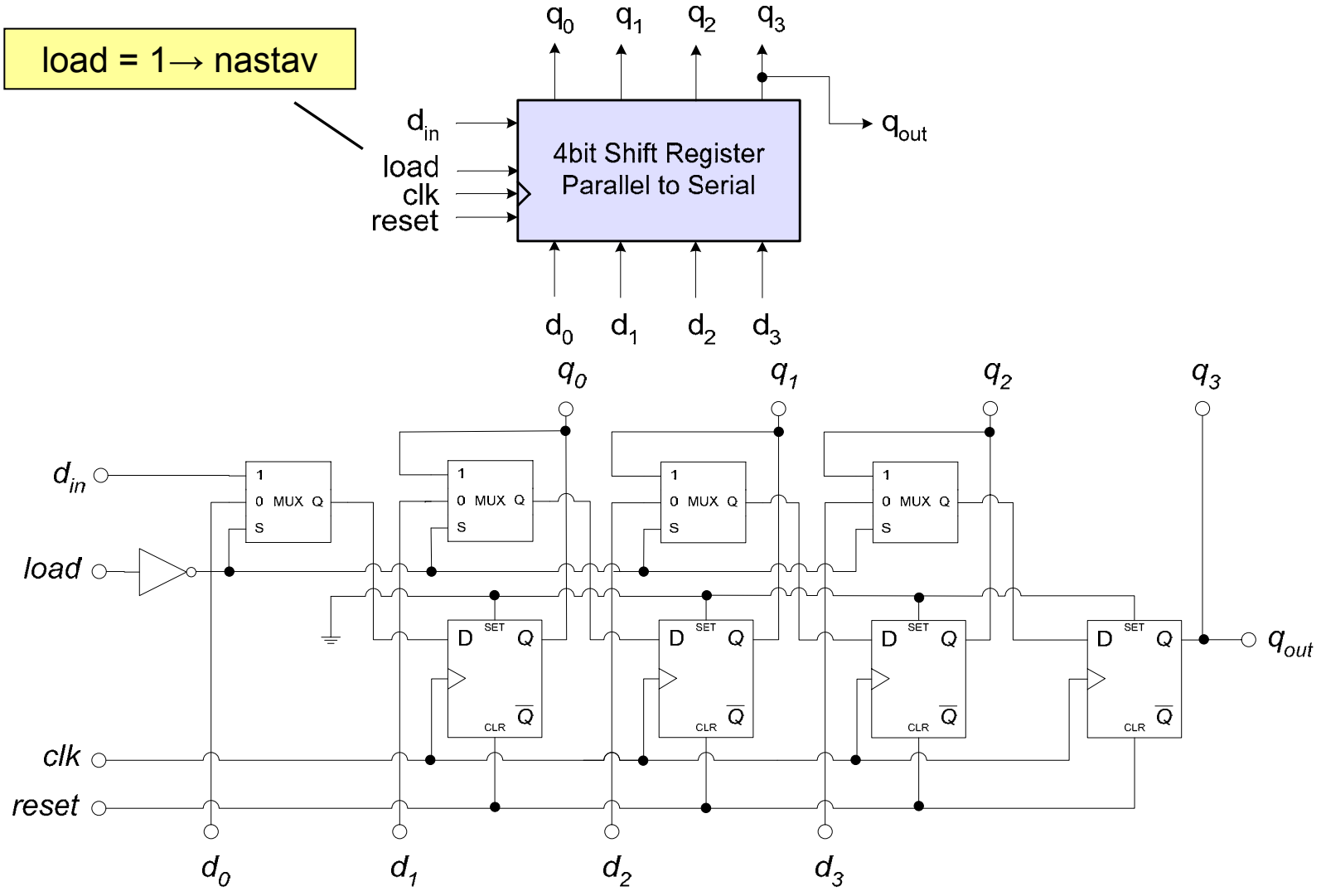
Posuvný registr (4bit Shift Register, Serial to Parallel)



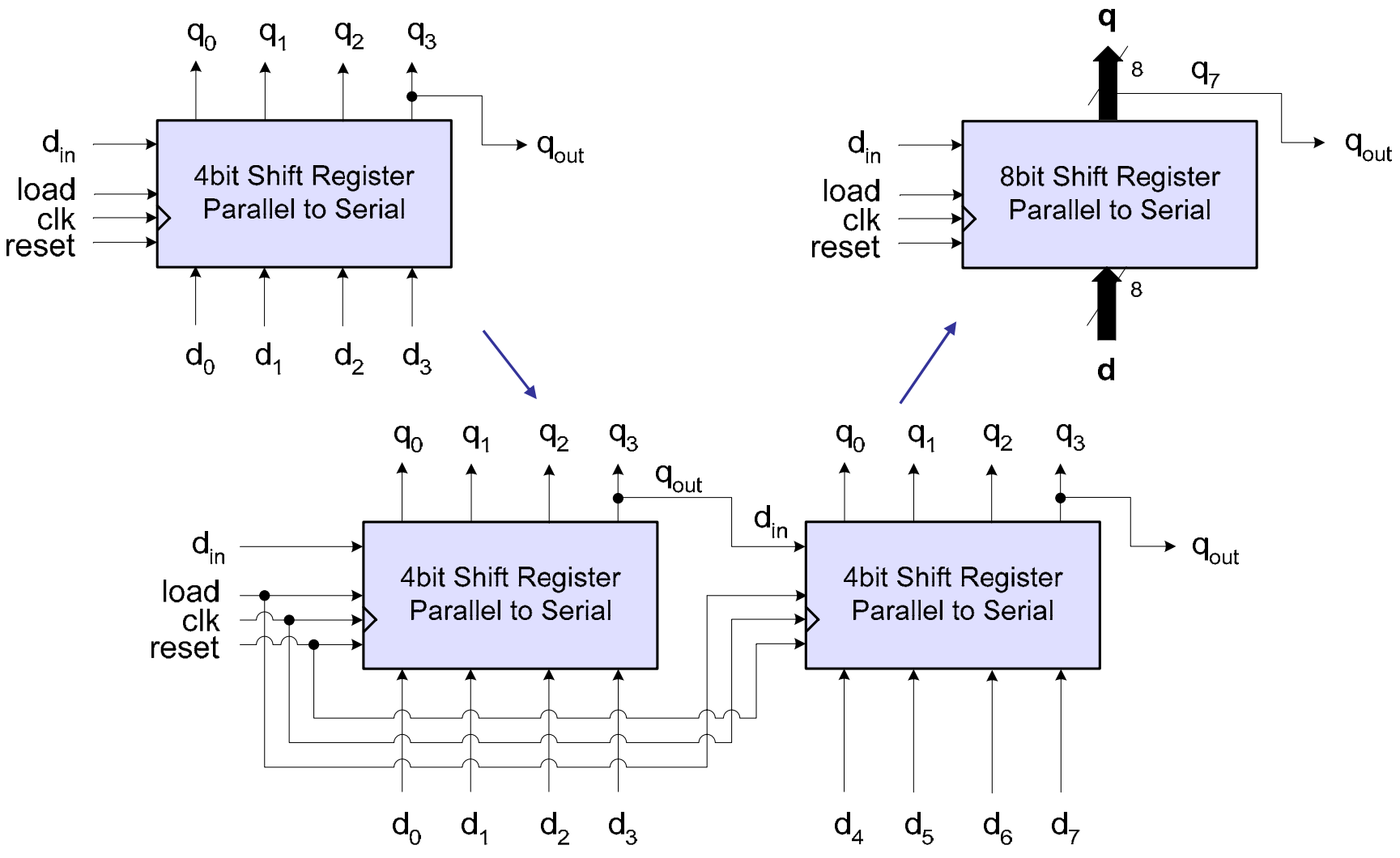


Posuvný registr (4bit Shift Register, Parallel to Serial)

load = 1→ nastav

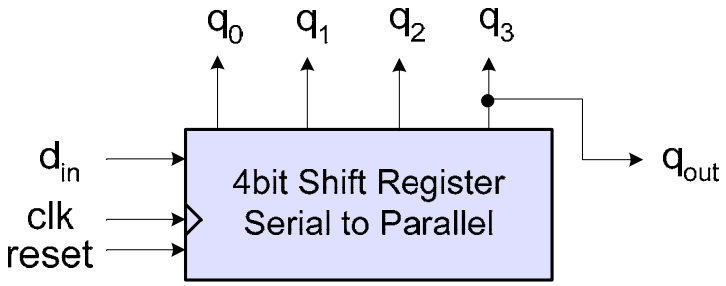


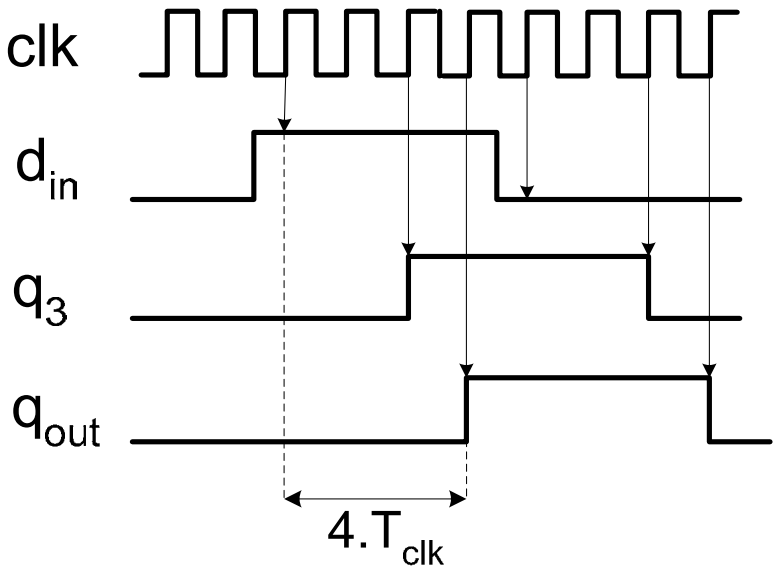
Posuvný registr (8bit Shift Register, Serial to Parallel)



Posuvný registr (4bit Shift Register, Serial to Parallel)

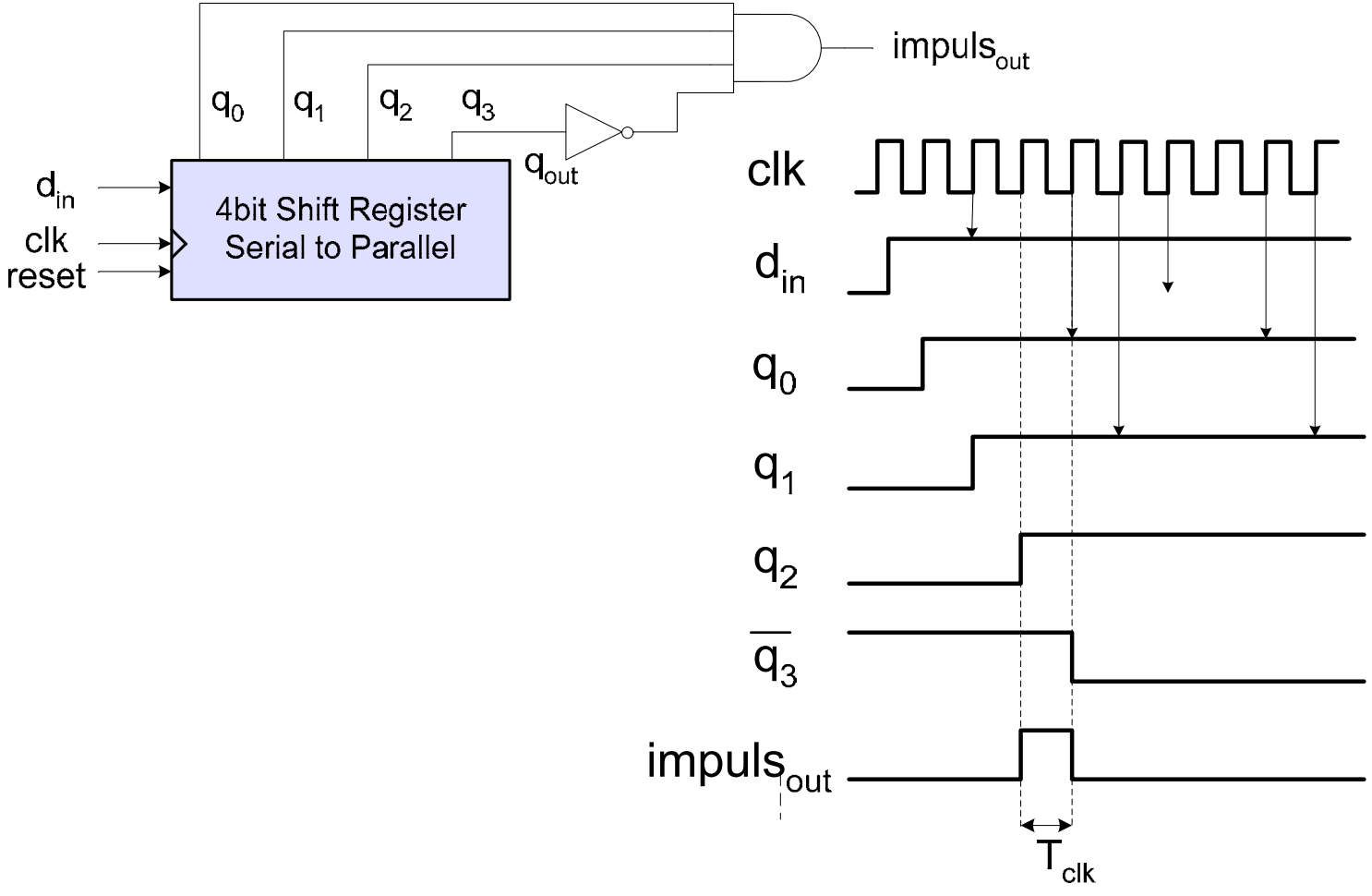
- Definované zpoždění signálu





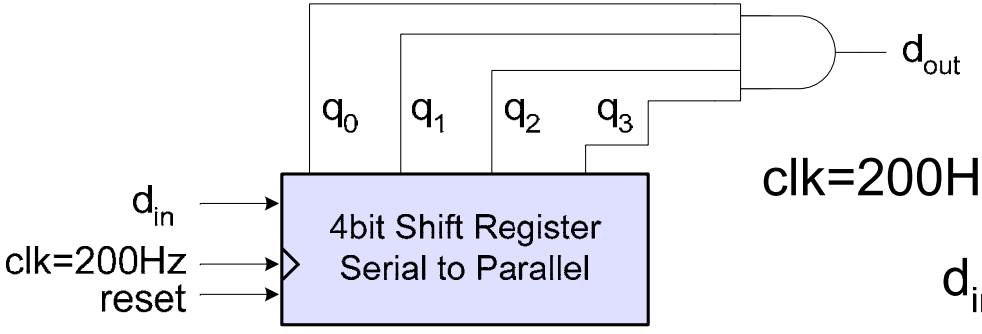
Posuvný registr (4bit Shift Register, Serial to Parallel)

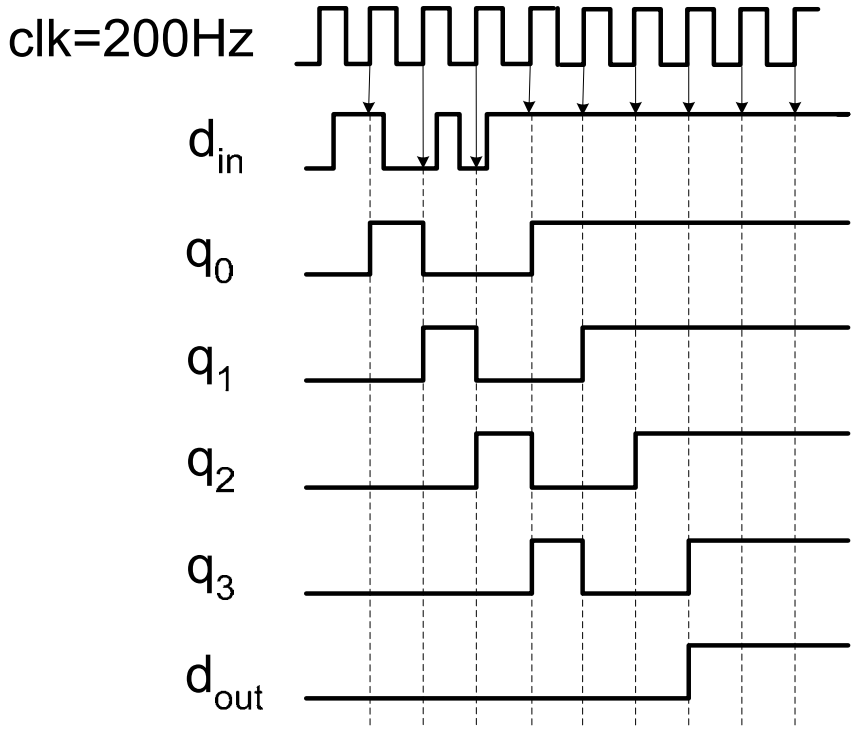
- Převod hladinového signálu na impuls (testování hardware)



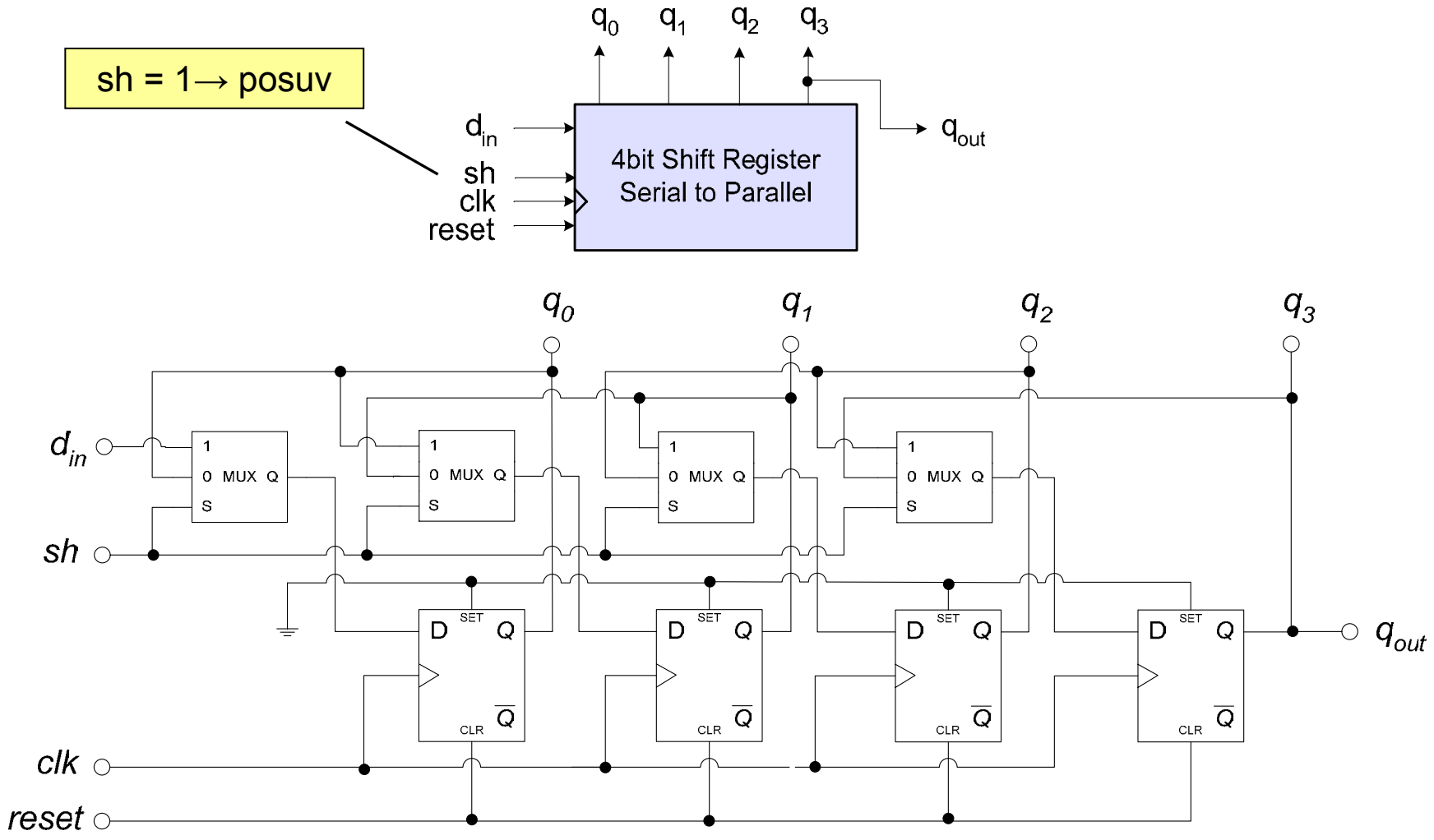
Posuvný registr (4bit Shift Register, Serial to Parallel)

- Potlačení zákmitů mechanických tlačítek a spínačů (Debounce Circuit)



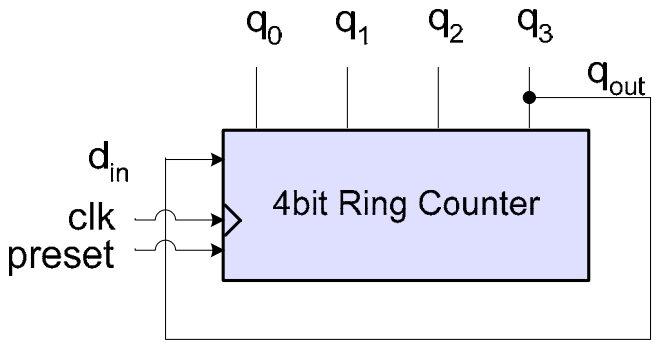


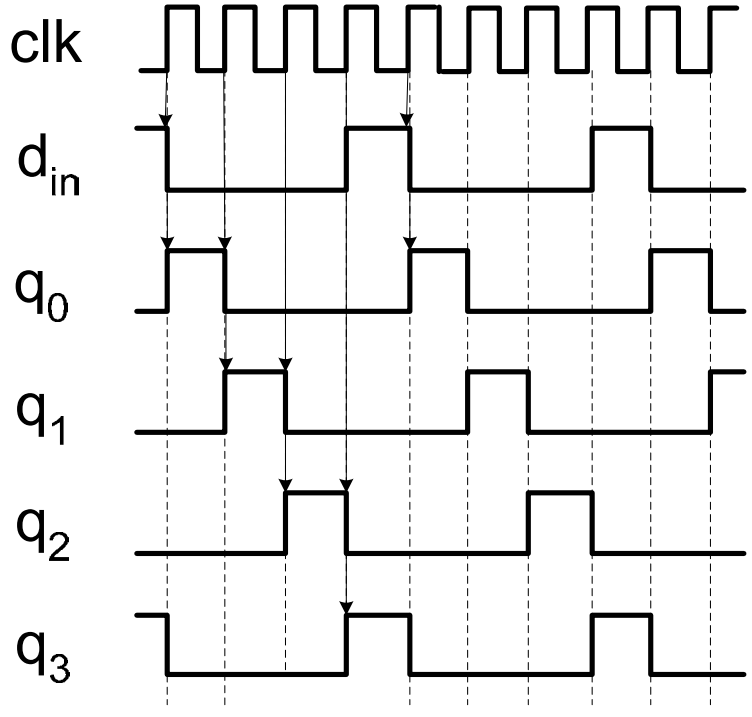
Posuvný registr (4bit Shift Register, Shift Enable)



Kruhový čítač (4bit Ring Counter)

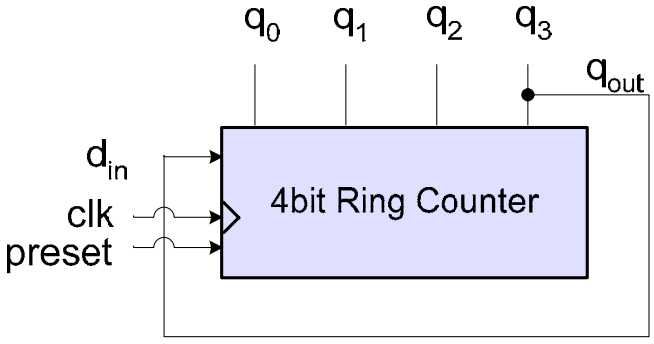
- Vícefázové hodiny pro řízení sekvenčních obvodů

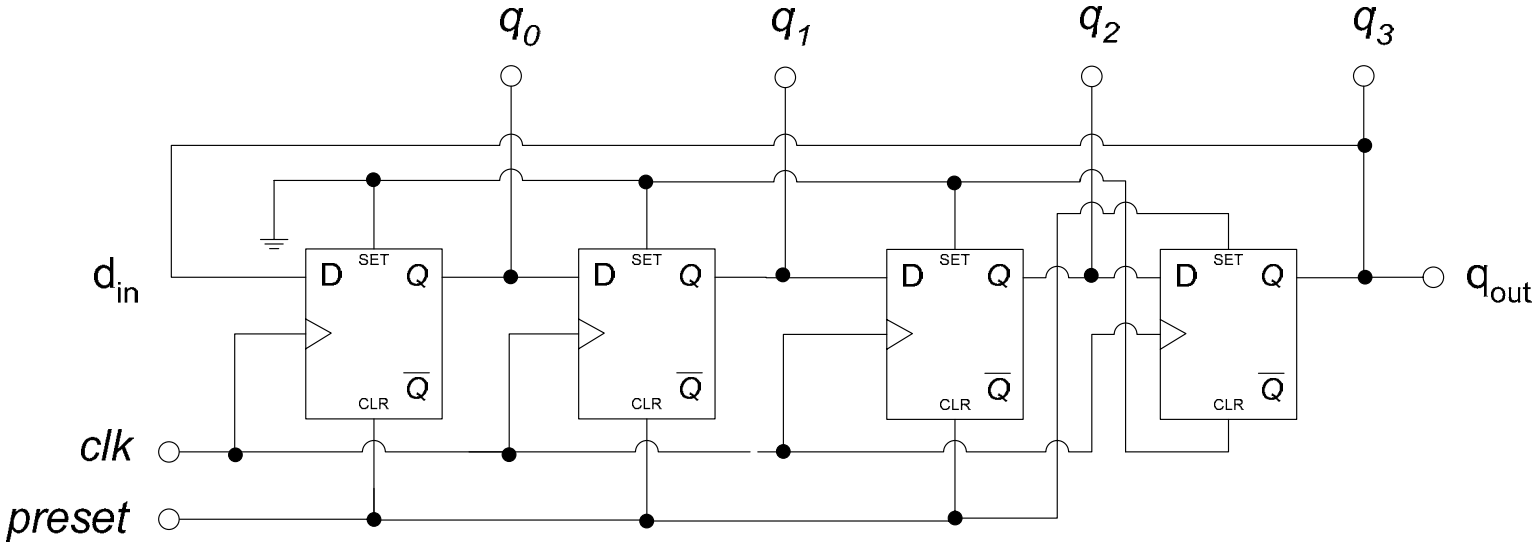




Kruhový čítač (4bit Ring Counter)

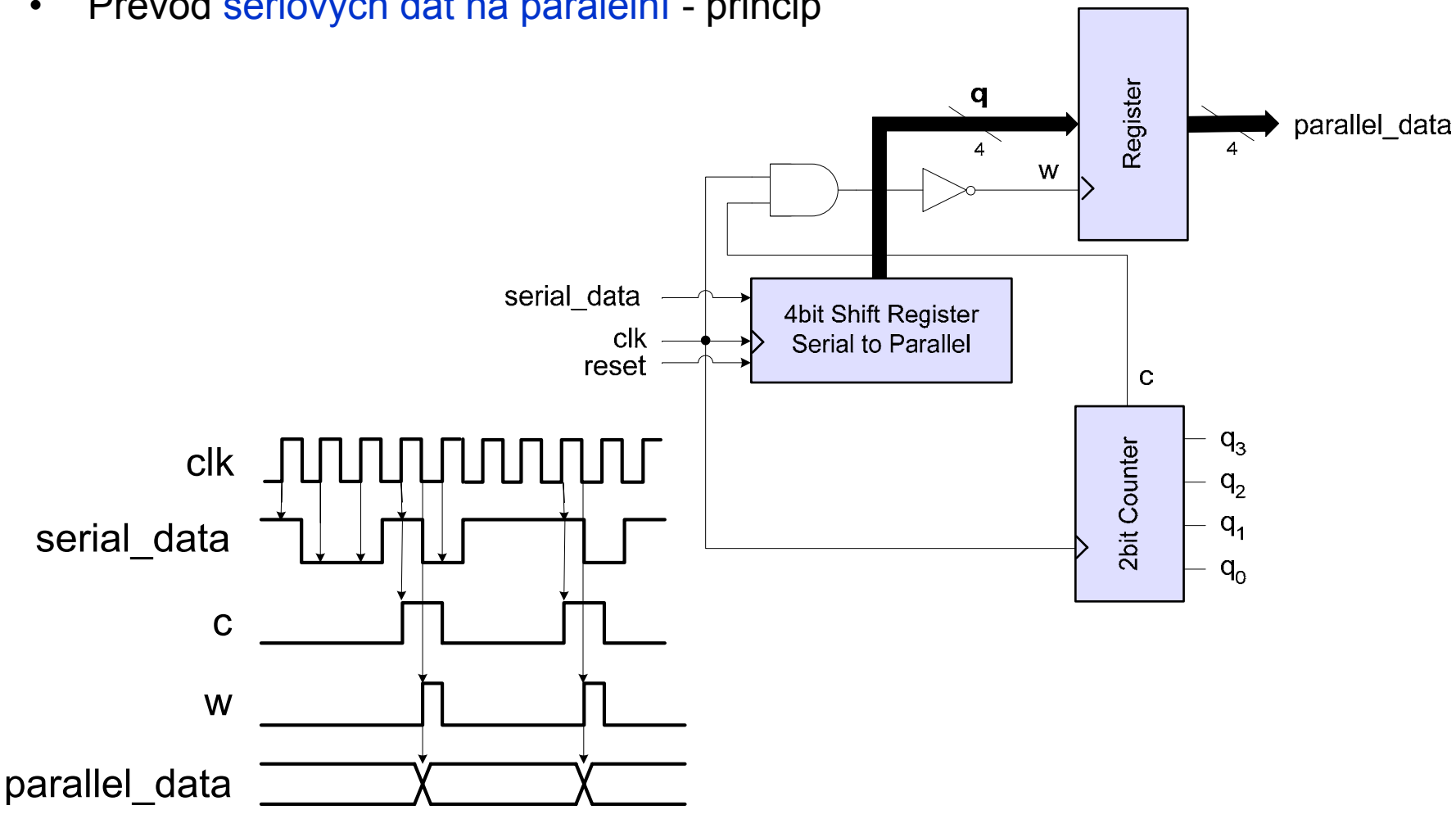
- Vícefázové hodiny pro řízení sekvenčních obvodů



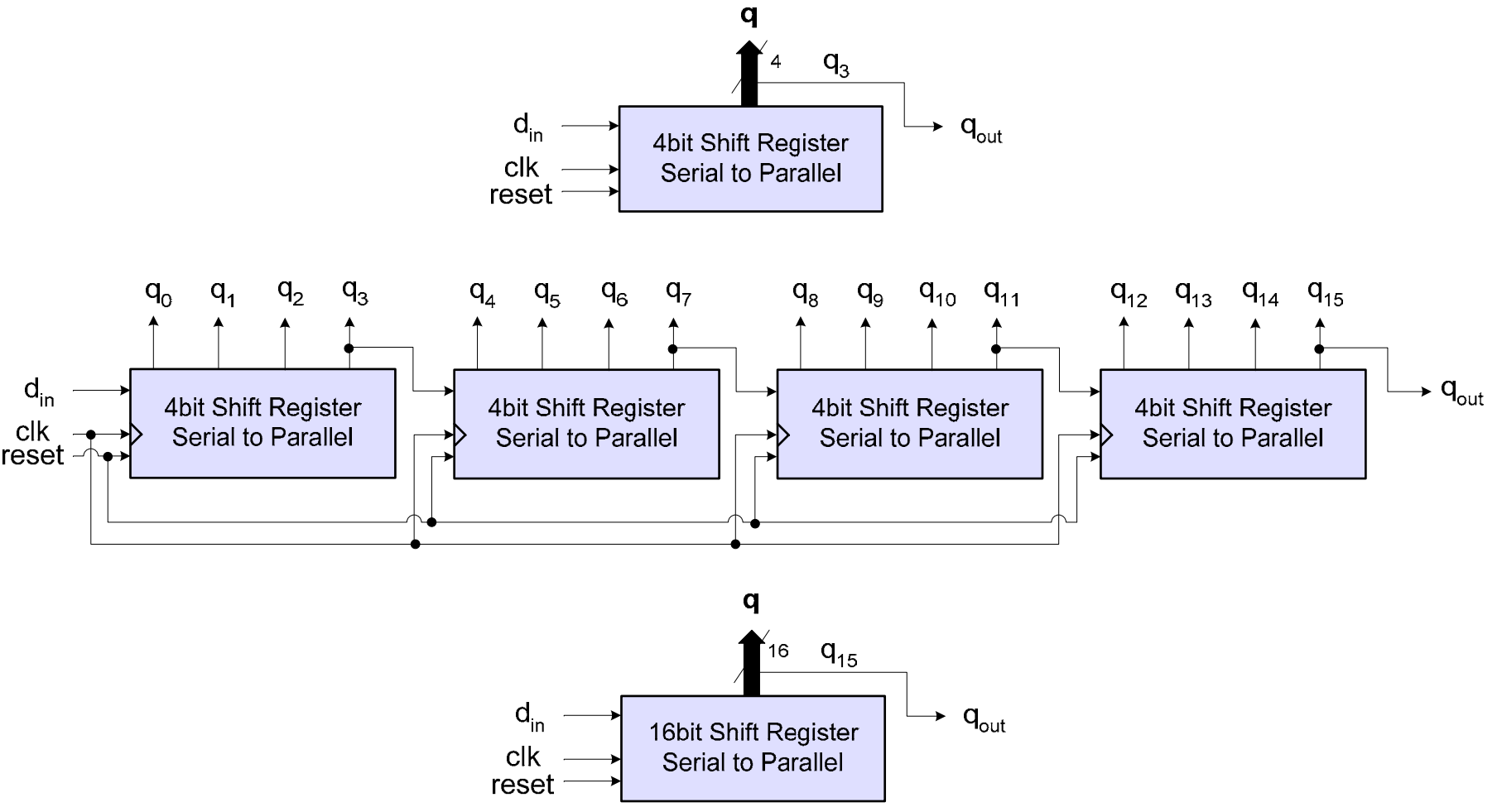


Posuvný registr (4bit Shift Register, Serial to Parallel)

- Převod sériových dat na paralelní - princip

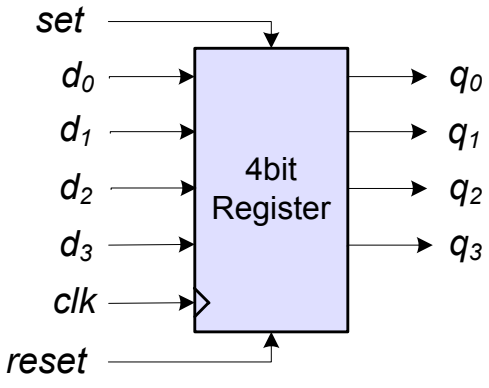


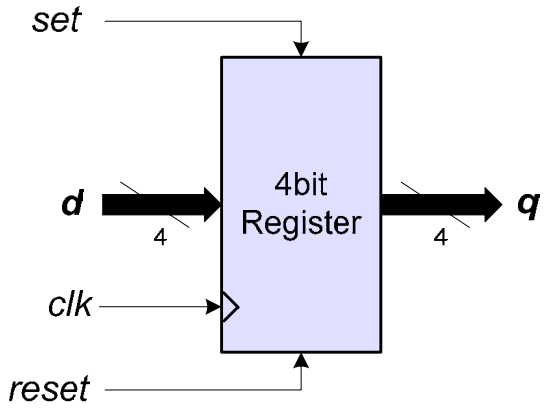
Posuvný registr (16bit Shift Register, Serial to Parallel)

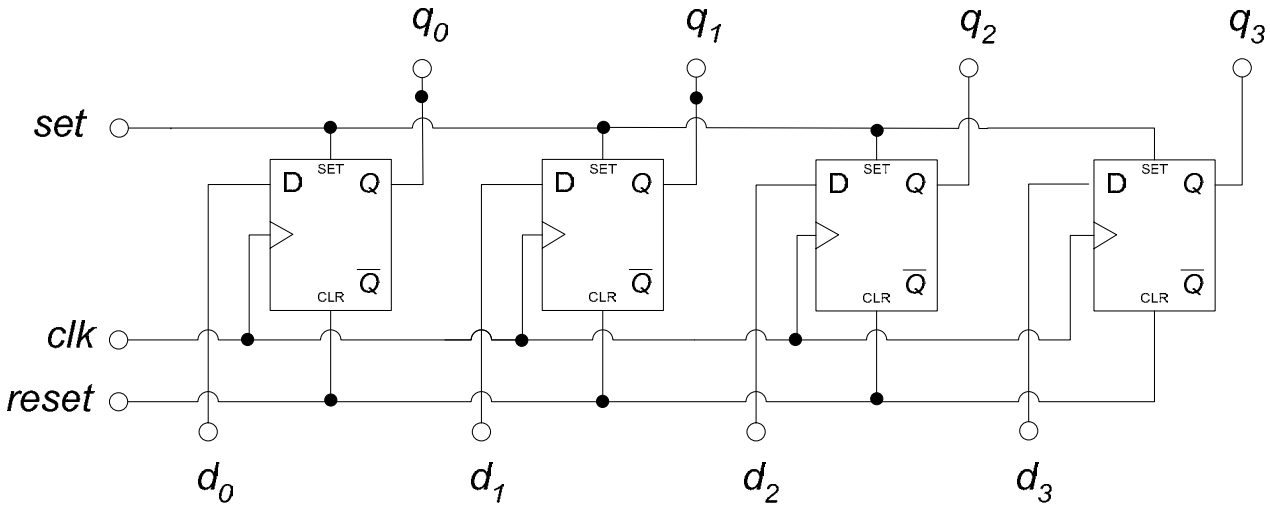


Registr (4bit Register)

- n – klopných obvodů řízených společným hodinovým signálem

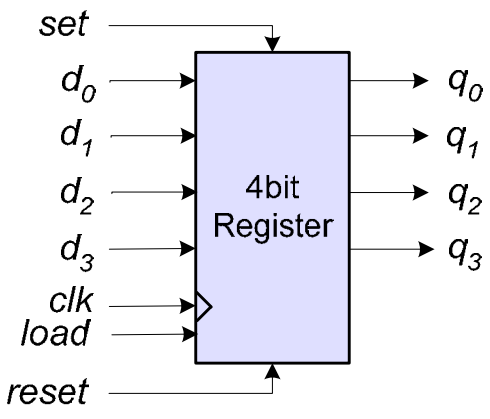


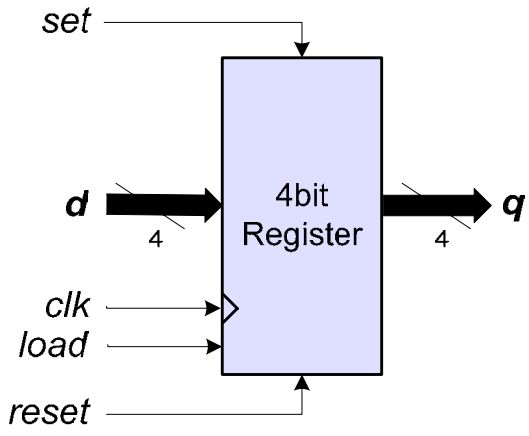


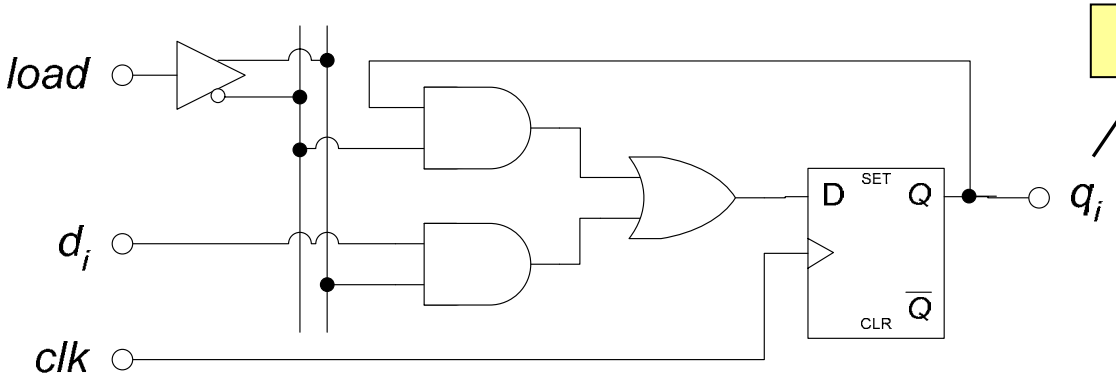


Registr (4bit Register, Parallel Load)

- n – klopných obvodů řízených společným hodinovým signálem
- Zápis do registru i při trvale běžících hodinách signálem load = 1

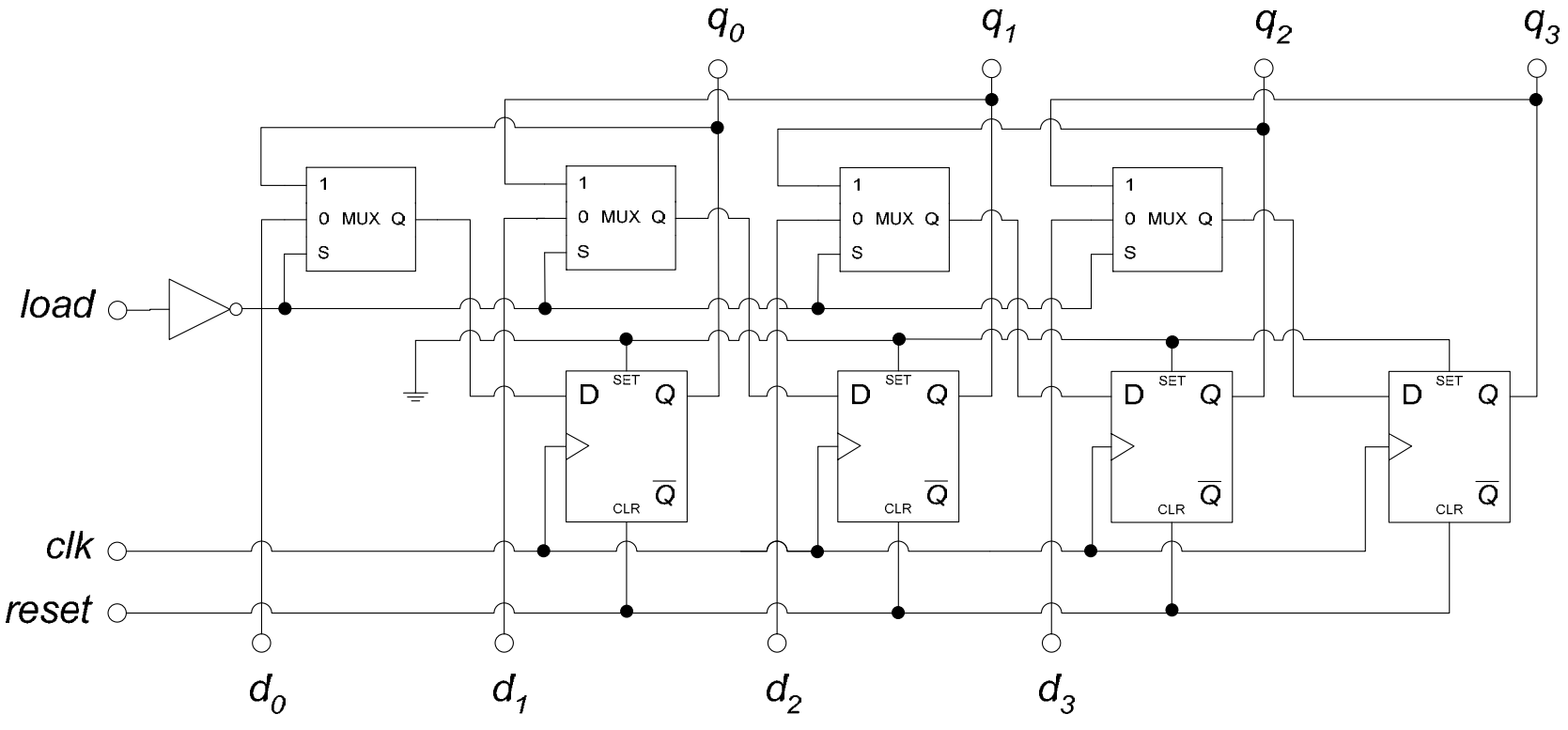






Registr (4bit Register, Parallel Load)

- n – klopných obvodů řízených společným hodinovým signálem
- Zápis do registru i při trvale běžících hodinách signálem load = 1



Hazardy

- Co je hazard v logických obvodech
- Příčiny vzniku hazardu
- Nalezení hazardu
- Kdy hazard ovlivní činnost logických obvodů?
- Poznámka: zde se zabýváme jen statickým hazardem.
  - Existují ještě hazardy dynamické (souvisejí se statickými)

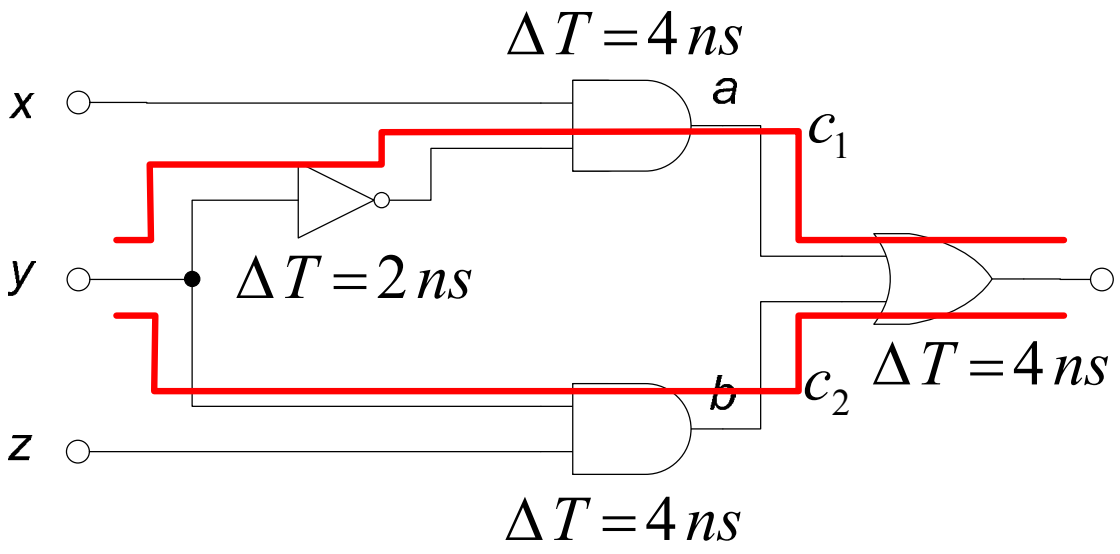
Příčiny vzniku hazardu

- Hazard je krátká neočekávaná změna výstupního signálu (glitch), která není matematickým výstupem logické funkce

- Signál ze vstupu logického obvodu se šíří na výstup různými cestami, které se někdy rozdělí a pak zase spojí. Signál se různými cestami vlivem časového zpoždění na hradlech a vodičích šíří různou dobu. V místě opětovného spojení má signál z různých cest různý časový posun.

- Statický hazard – výstup logického obvodu má být trvale v 0 nebo 1 (má být statický), místo toho se objeví krátký impuls do opačné úrovně.
  - 0-1-0 … statický hazard v 0
  - 1-0-1 … statický hazard v 1

Statický hazard v úrovni 1



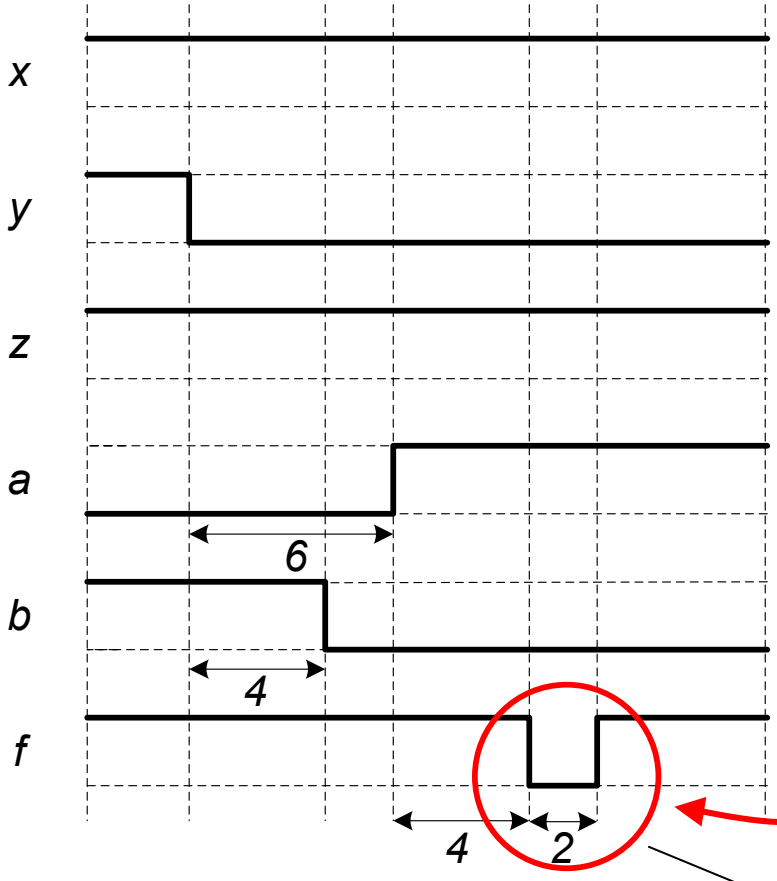
$$f = x\,\overline{y} + y\,z$$

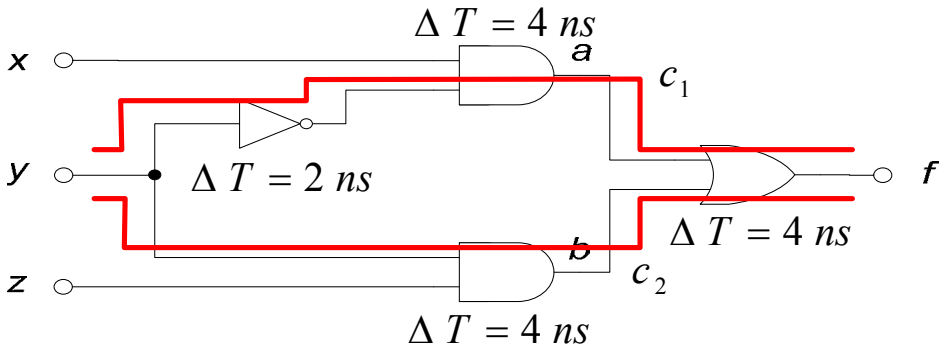
$$\Delta T_{c1} = 2 + 4 + 4 = 10\ ns$$

$$\Delta T_{c2} = 4 + 4 = 8\,ns$$

$$\Delta T_H = \Delta T_{c1} - \Delta T_{c2} = 2\,ns$$

Statický hazard v úrovni 1





$$\text{Pro: } x=1, z=1 \rightarrow$$

$$\rightarrow f = x\overline{y} + yz = 1$$

!! hazard

Skutečnost

Má být

Kdy hazardy vadí?

- Hazardy v kombinačních obvodech nejsou kritické – výstup kombinačního obvodu se po určité (krátké) době vždy ustálí ve správné hodnotě

- Hazardy v sekvenčních obvodech mohou uvést klopné obvody do nesprávného stavu a tím nastavit celý sekvenční obvod (konečný automat) do nevratného kritického stavu !!!

- Řešení:
  - Hazard-free design
  - Synchronní návrh a správný výpočet maximální povolené hodinové (synchronizační) frekvence

Logické obvody
Kombinační a sekvenční stavební bloky

KONEC

České vysoké učení technické Fakulta elektrotechnická