

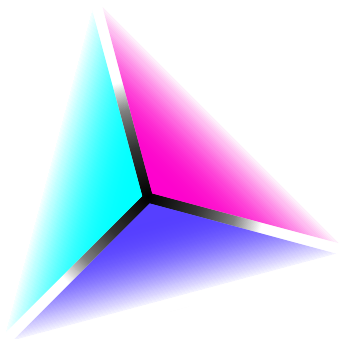
Struktura a architektura počítačů

Katedra číslicového návrhu
Fakulta informačních technologií
České vysoké učení technické

© Hana Kubátová, 2021

Typické obvody v číslicovém počítači

BI-SAP, březen 2021



Obsah

- Logické obvody kombinační a sekvenční, často použité v číslicovém počítači
- Sčítačka, pulsčítačka, registr, čítač
- (Metoda minimalizace Quinne-McCluskey)

Některé obrázky jsou převzaty z doporučené literatury:

- Gajski, D. D.: Principles of Digital Design. Prentice-Hall International, Inc. 1997

a z výukových materiálů doc. Pluháčka

Často používané obvody

Majorita: nabývá hodnotu 1, když **většina** proměnných je rovna 1

Majorita ze 3 ... M_3 (tzn. alespoň dvě proměnné jsou jedničkové)

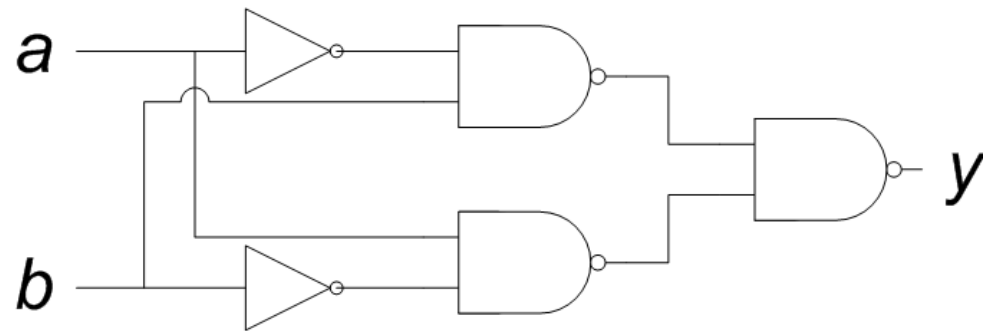
$$M_3(a, b, c) = a.b.\bar{c} + a.\bar{b}.c + \bar{a}.b.c + a.b.c$$

Úpravami, algebraicky nebo v mapě dostaneme:

$$M_3(a, b, c) = a.b + a.c + b.c$$

co vám to připomíná?

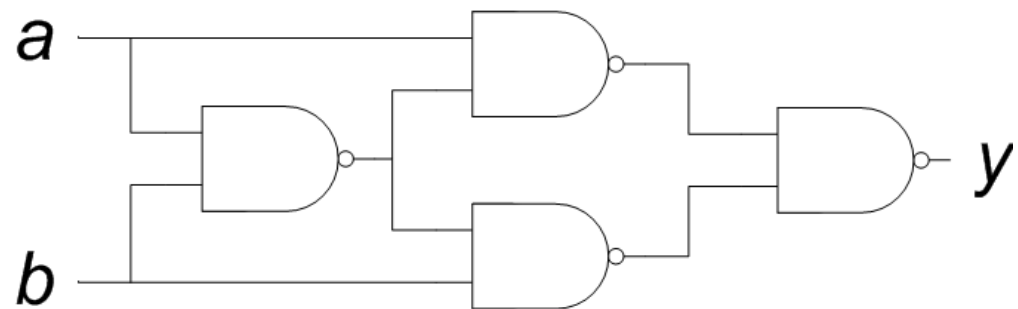
XOR



$$a \oplus b = a\bar{b} + \bar{a}b$$

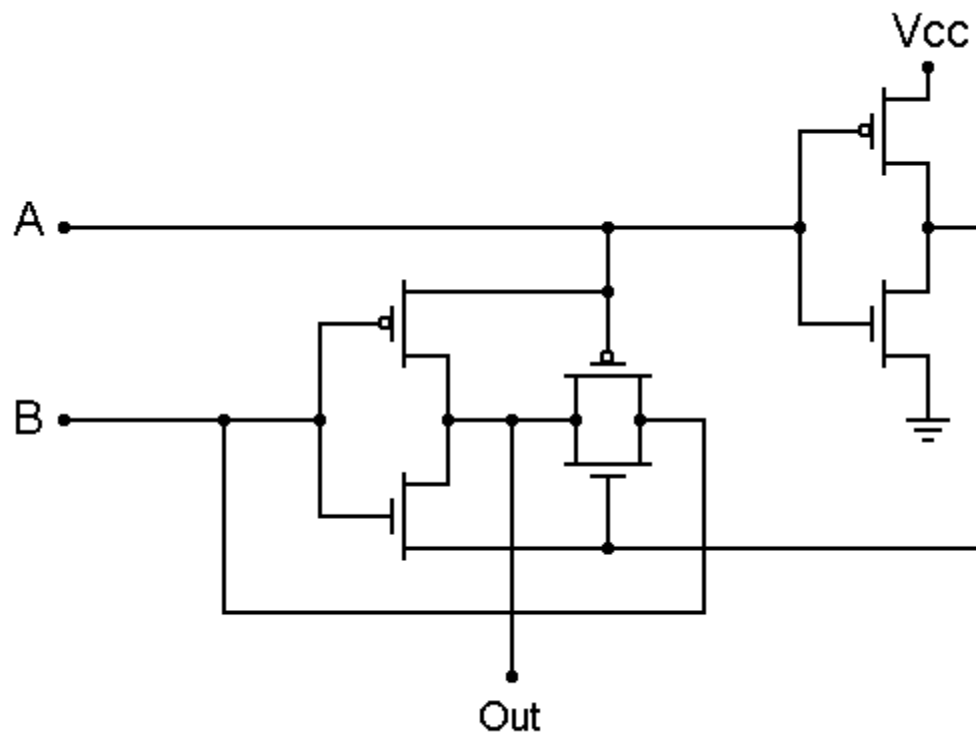
$$a\bar{b} + 0 + \bar{a}b + 0 = a\bar{b} + a\bar{a} + \bar{a}b + \bar{b}b =$$

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



oblíbené zapojení XORu: 4x NAND tedy 4x4 transistory = 16, ale v CMOS lze lépe:

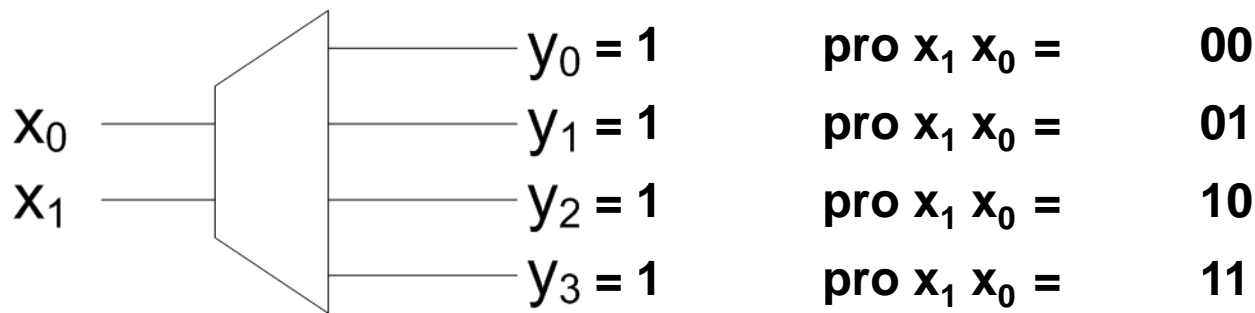
XOR - CMOS



Jen 6 transistorů místo 16

Dekodéry

Dekodér: binární kód \rightarrow 1 z N, zde 1 ze 4



4 výstupní funkce, pravdivostní tabulky:

x_1	x_0	y_0	y_1	y_2	y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Booleovské výrazy:

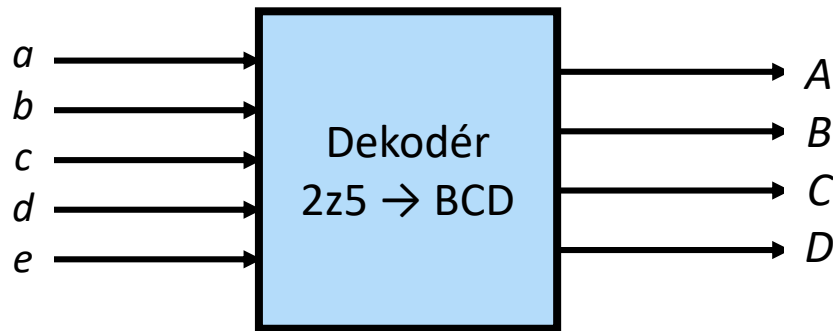
$$y_0 = \overline{x_1} \cdot \overline{x_0}$$

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

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

$$y_3 = x_1 \cdot x_0$$

Dekodéry ... obecně, jak navrhnout

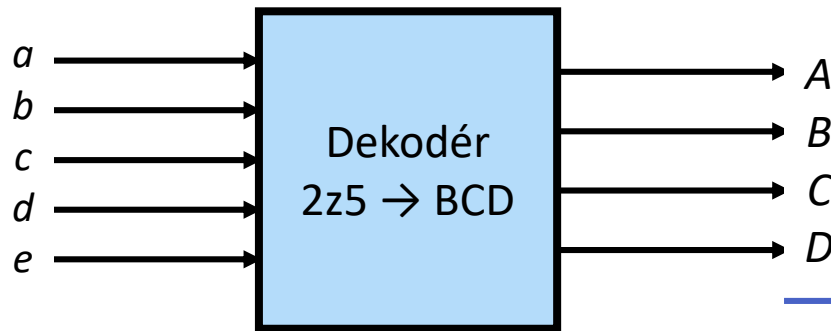


$$\begin{aligned}
 A &= \bar{e}b + ea + ec \\
 B &= \bar{e}c + ea \\
 C &= \bar{e}d + ea \\
 D &= eb + ec
 \end{aligned}$$

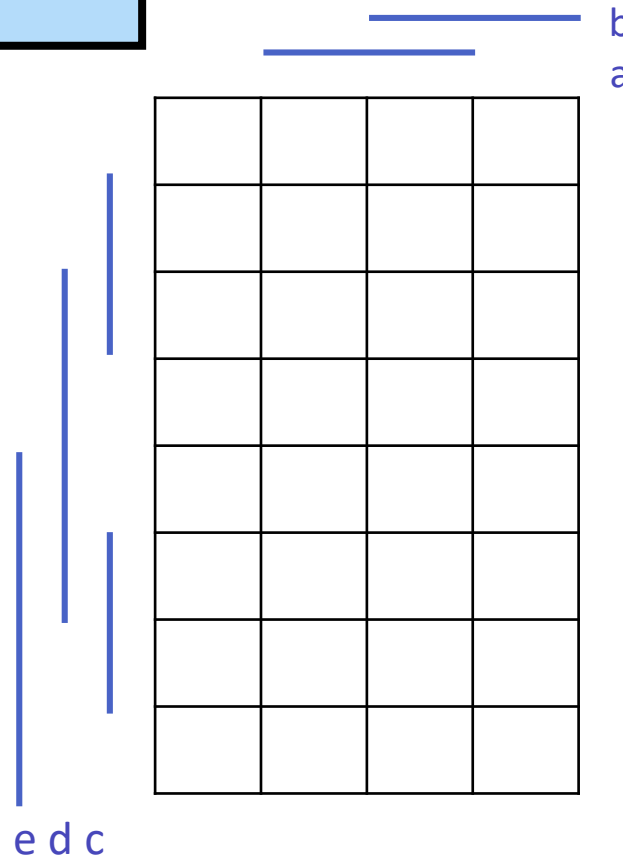
Skupinová
minimalizace

Desítkové číslo	Kód BCD <i>DCBA</i>	Kód 2z5 typ 74210 <i>edcba</i>	Kód 2z5 typ 84210 <i>edcba</i>
0	0000	11000	10100
1	0001	00011	00011
2	0010	00101	00101
3	0011	00110	00110
4	0100	01001	01001
5	0101	01010	01010
6	0110	01100	01100
7	0111	10001	11000
8	1000	10010	10001
9	1001	10100	10010

Nástin postupu řešení:



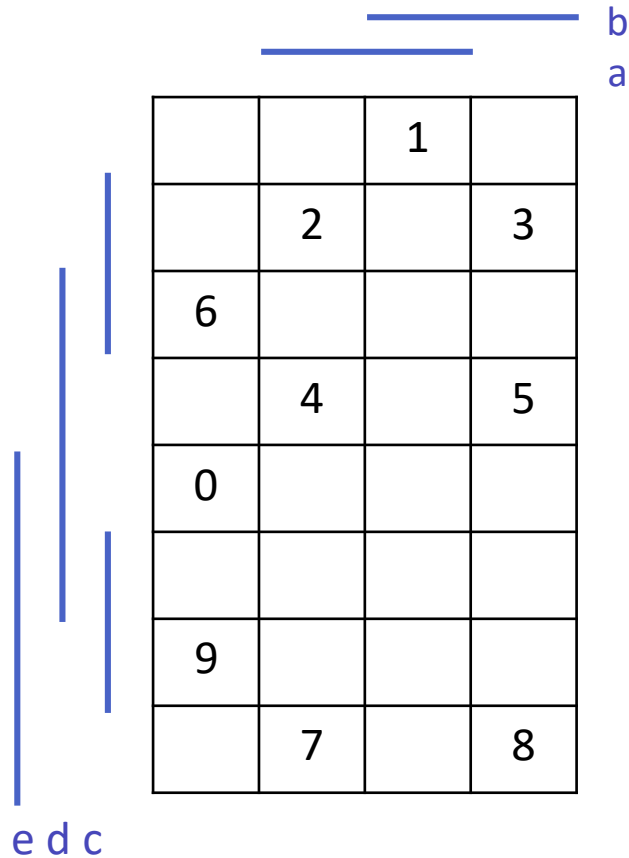
5 vstupních proměnných a 4 výstupní funkce
→ mapa pro 5 proměnných



edcba	DCBA
11000	0000
00011	0001
00101	0010
00110	0011
01001	0100
01010	0101
01100	0110
10001	0111
10010	1000
10100	1001

Definiční obor

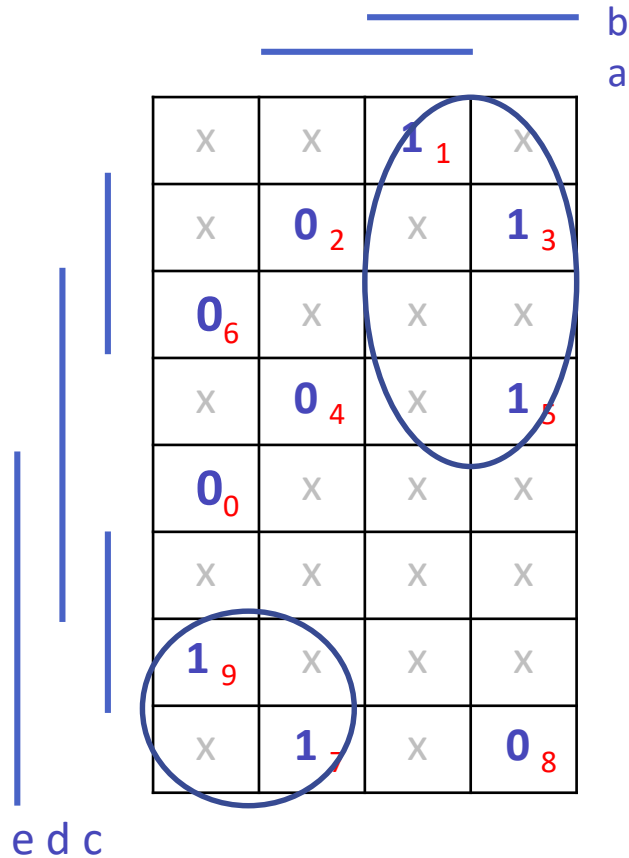
označení 1 až 9 je symbolické,
není to stavový index



	edcba	DCBA
0	11000	0000
1	00011	0001
2	00101	0010
3	00110	0011
4	01001	0100
5	01010	0101
6	01100	0110
7	10001	0111
8	10010	1000
9	10100	1001

definiční obor je kód 2 z 5, tzn. hledáme
místa v mapě, kde jsou právě 2 vstupní
proměnné jedničkové (podle tabulky)

Využití neurčených stavů



	edcba	DCBA
0	11000	0000
1	00011	0001
2	00101	0010
3	00110	0011
4	01001	0100
5	01010	0101
6	01100	0110
7	10001	0111
8	10010	1000
9	10100	1001

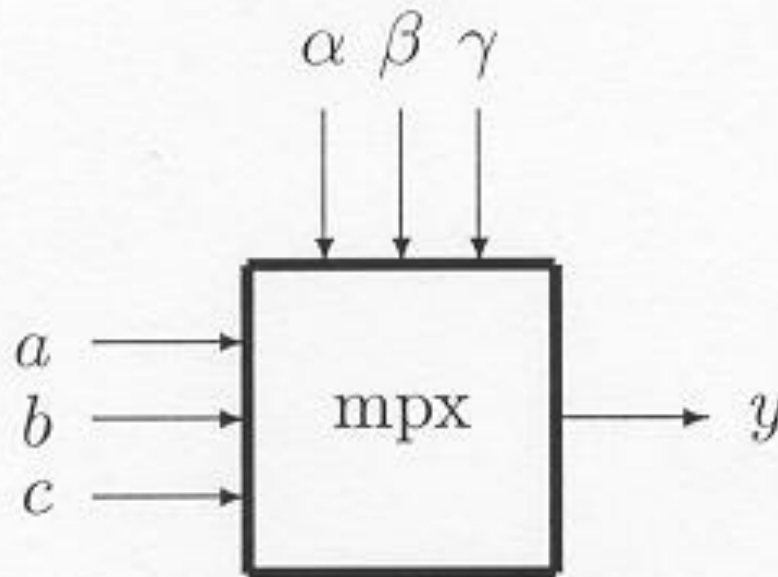
Z mapy získáme MNDF pro A:

$$A = \bar{e}b + \bar{b}\bar{d}e$$

a dtto pro všechny další výstupy

Multiplexor - princip

řízení (výběr, selekce ... *někdy název selektor*) toho, který ze vstupů se propojí na výstup



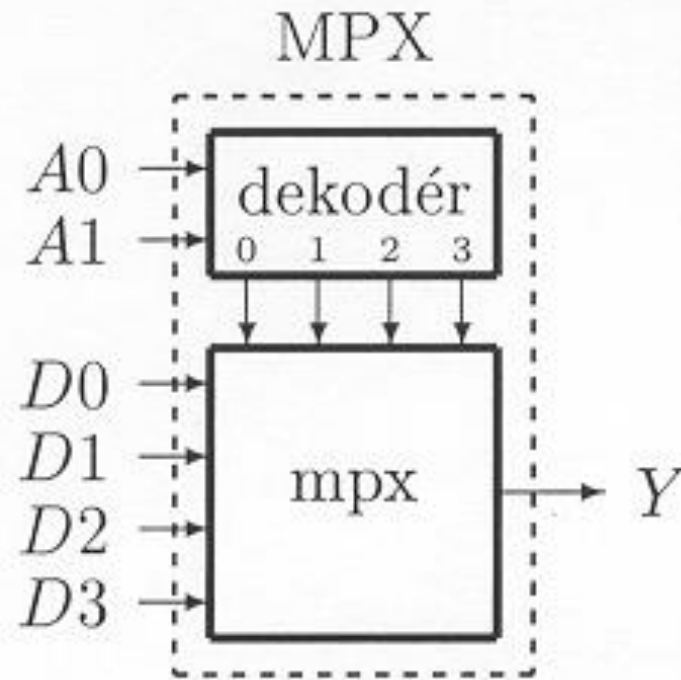
α	β	γ	y
0	0	0	0
1	0	0	a
0	1	0	b
0	0	1	c
jinak			?

$$y = a.\alpha + b.\beta + c.\gamma$$

Multiplexor - funkce

řídící vstupy, též adresa

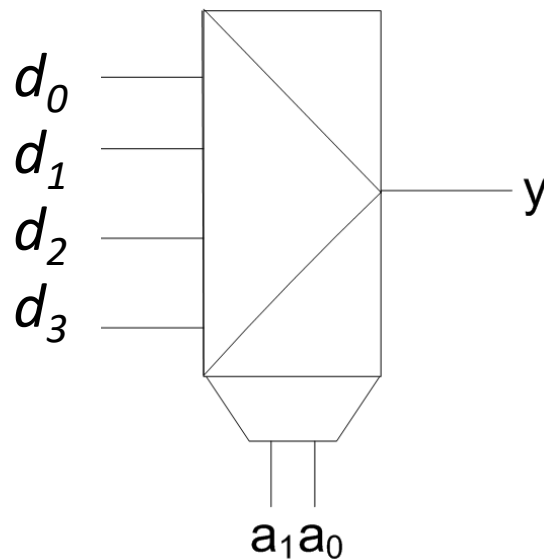
„adresa“		
A1	A0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3



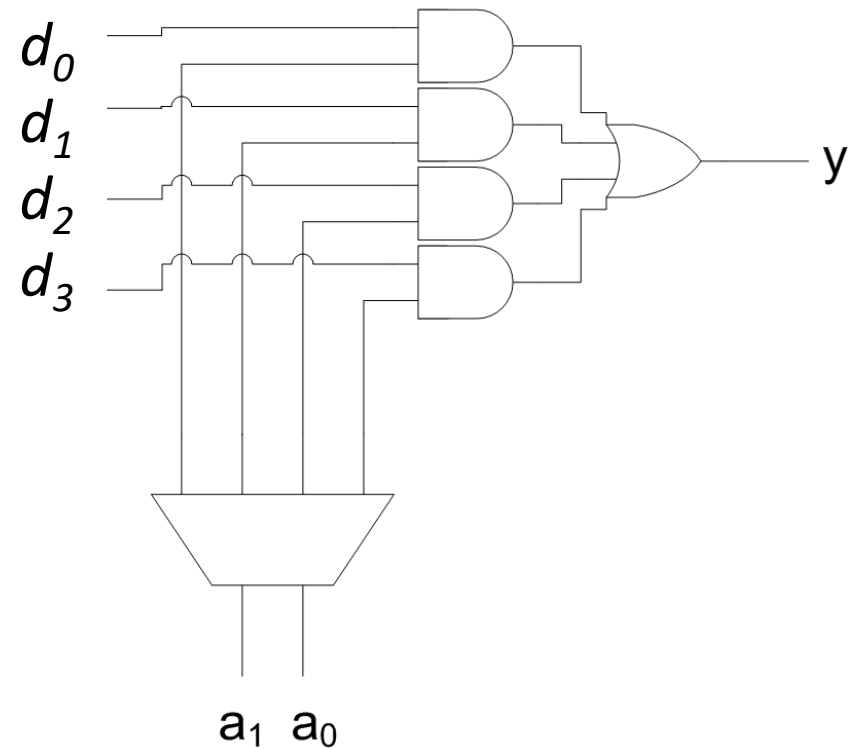
$$Y = \overline{A_1}.\overline{A_0}.D_0 + \overline{A_1}.A_0.D_1 + A_1.\overline{A_0}.D_2 + A_1.A_0.D_3$$

Multiplexor - realizace

Multiplexor:

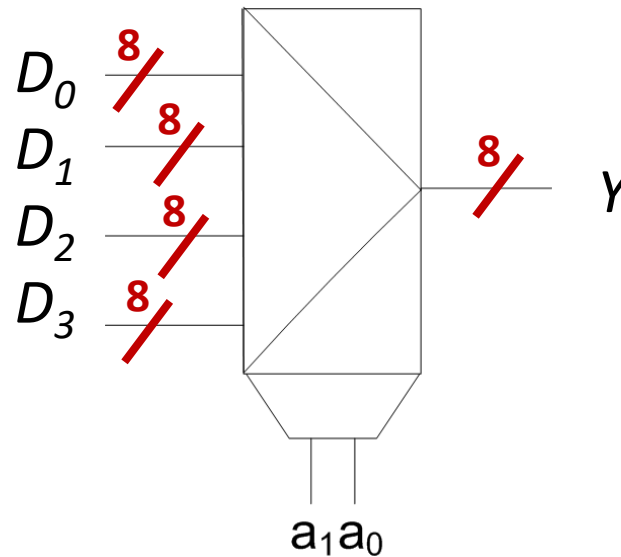


Realizace:



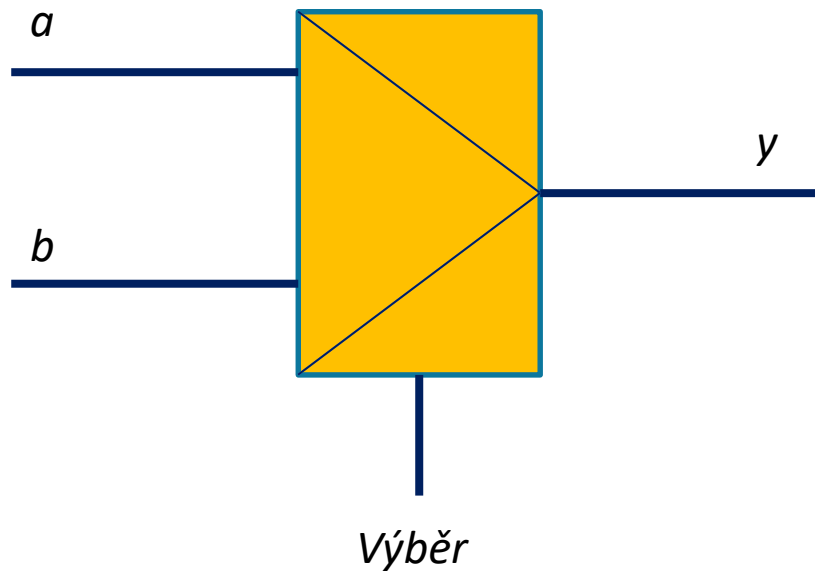
Multiplexor

Multiplexor pro širší výběr (sběrnice ... jeden z osmibitových vstupů se podle aktuální adresy připojuje na osmibitový výstup)



Multiplexor

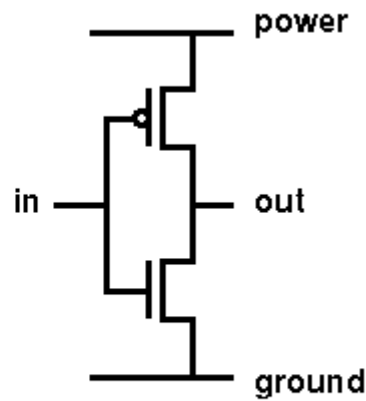
- v CMOS je 2-vstupový MUX realizován pomocí přenosových hradel, tzn. jen 6 transistorů



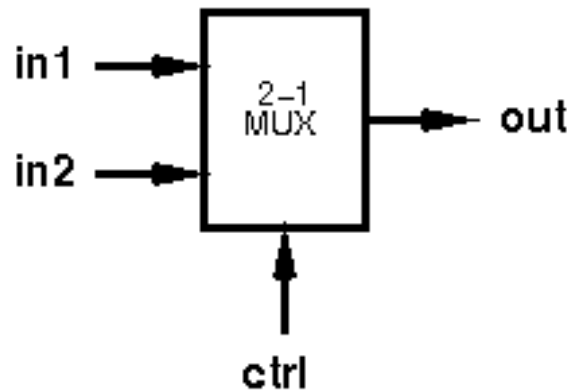
NOT



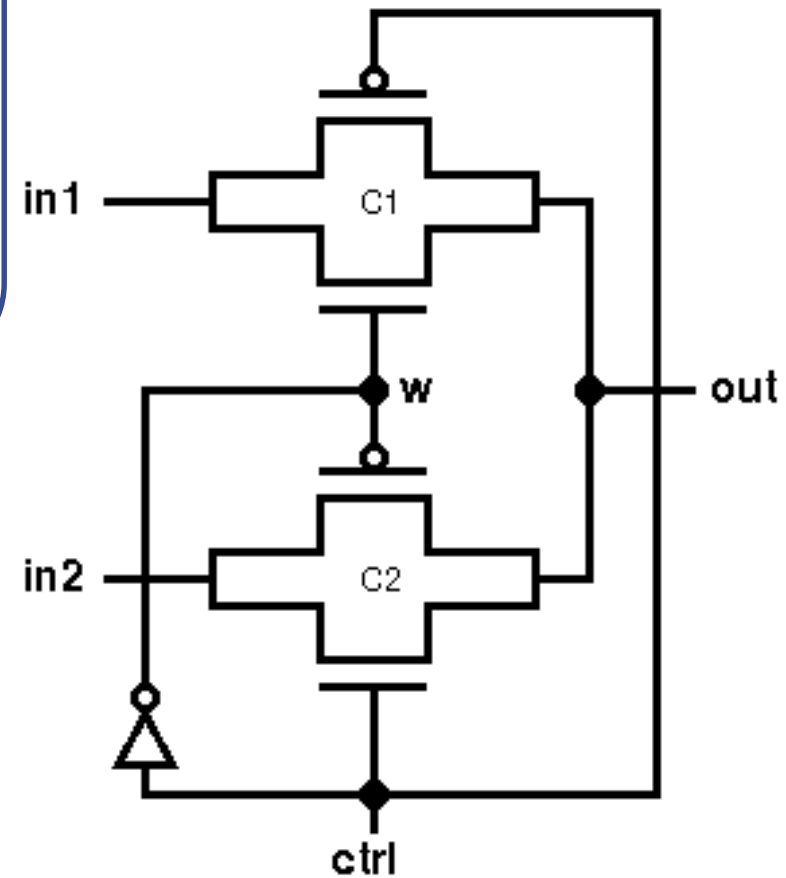
Gate representation



Switch-level model



Gate representation



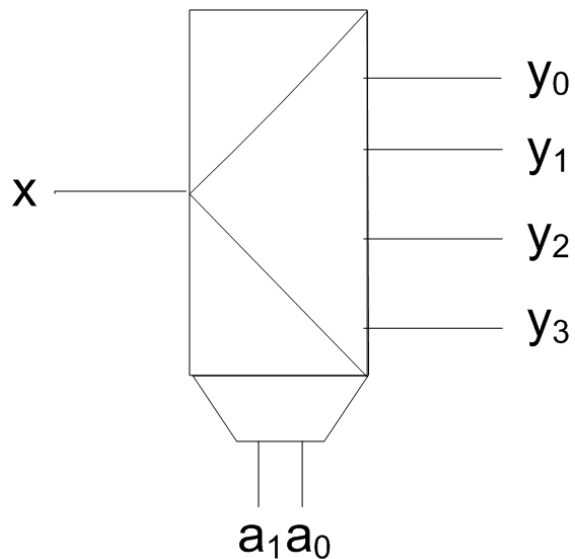
Switch-level model

Příklady, popis zapojení a simulace:

<http://www.indiabix.com/electronics-circuits/>

Demultiplexor

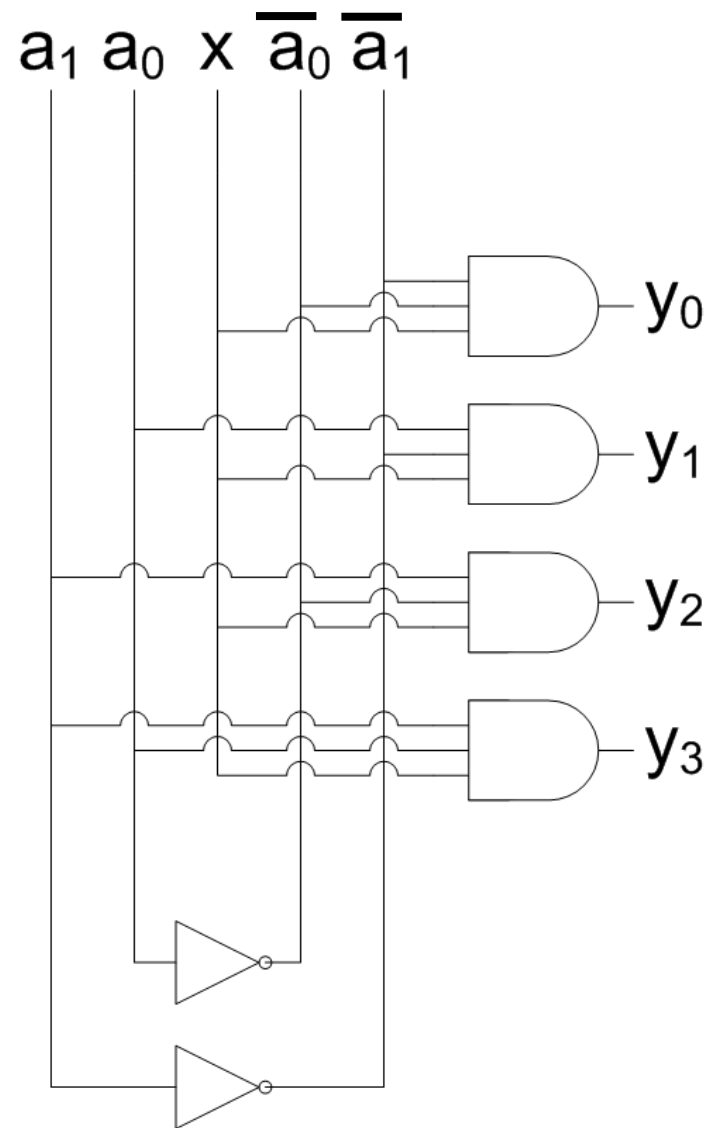
Demultiplexor:



Pravdivostní tabulka:

a_1	a_0	x	y_0	y_1	y_2	y_3
0	0	0	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	0
1	0	0	0	0	0	0
1	0	1	0	0	1	0
1	1	0	0	0	0	0
1	1	1	0	0	0	1

Realizace demultiplexoru



Sčítáčka - opakování

a	b	p	q	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$s = \bar{a}\bar{b}p + \bar{a}b\bar{p} + a\bar{b}\bar{p} + abp$$

$$q = \bar{a}bp + a\bar{b}p + ab\bar{p} + abp$$

$$(= ap + bp + ab)$$

Poloviční sčítačka: *half-adder*

Úplná binární sčítačka: $s = \bar{a}\bar{b}p + \bar{a}b\bar{p} + a\bar{b}\bar{p} + abp$

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

$$q = ap + bp + ab = M_3(a, b, p)$$

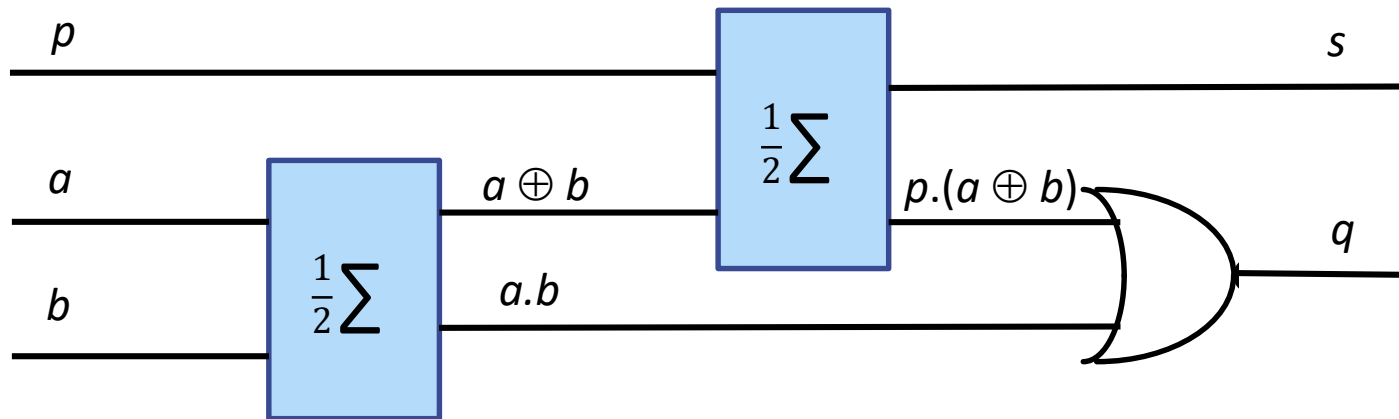
Poloviční sčítačka (*např. není přenos*):

$$s = a \oplus b$$

$$q = a.b$$

a	b	q	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Sčítací pomocí pulsčítaček

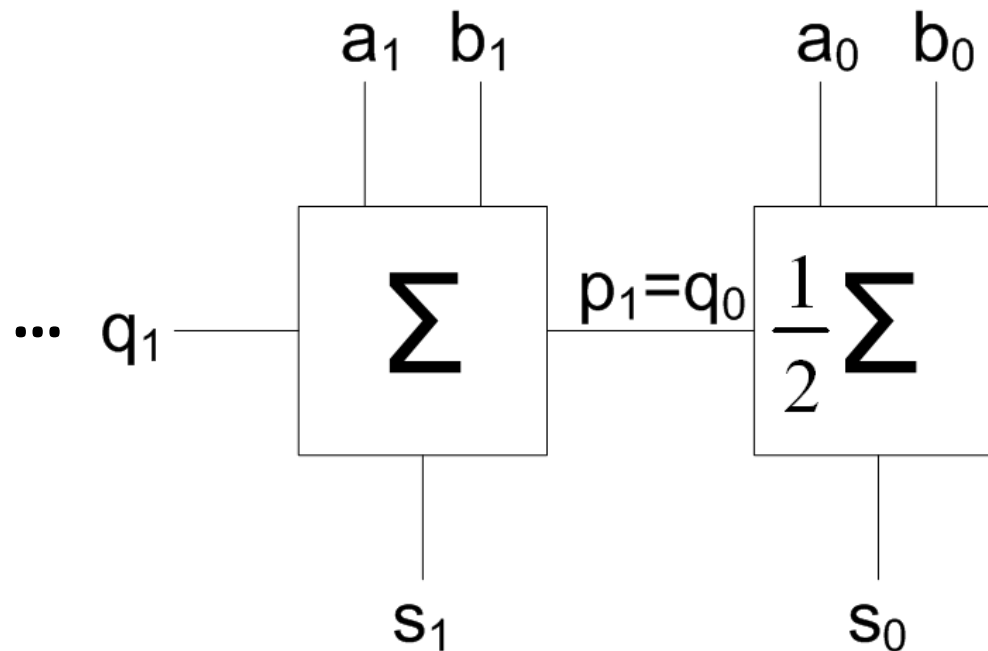


$$\begin{aligned}
 q &= ab + p.(a \oplus b) = ab + p(a\bar{b} + \bar{a}b) = ab + a\bar{b}p + \bar{a}bp \\
 &= a(b + \bar{b}p) + \bar{a}bp = ab + ap + \bar{a}bp \\
 &= b(a + \bar{a}p) + ap = ab + bp + ap
 \end{aligned}$$



2x absorpce negace, nebo mapa

Paralelní sčítačka



$$S = A + B$$

$$A = \dots a_2 a_1 a_0$$

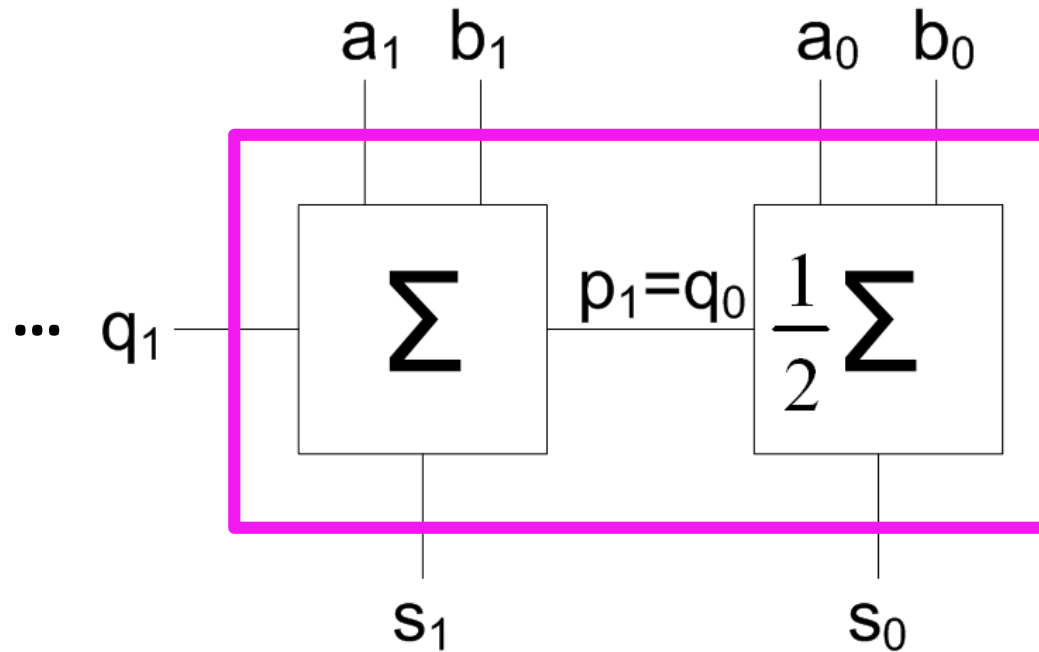
$$B = \dots b_2 b_1 b_0$$

Paralelní sčítačka

$$S = A + B$$

$$A = \dots a_2 a_1 a_0$$

$$B = \dots b_2 b_1 b_0$$

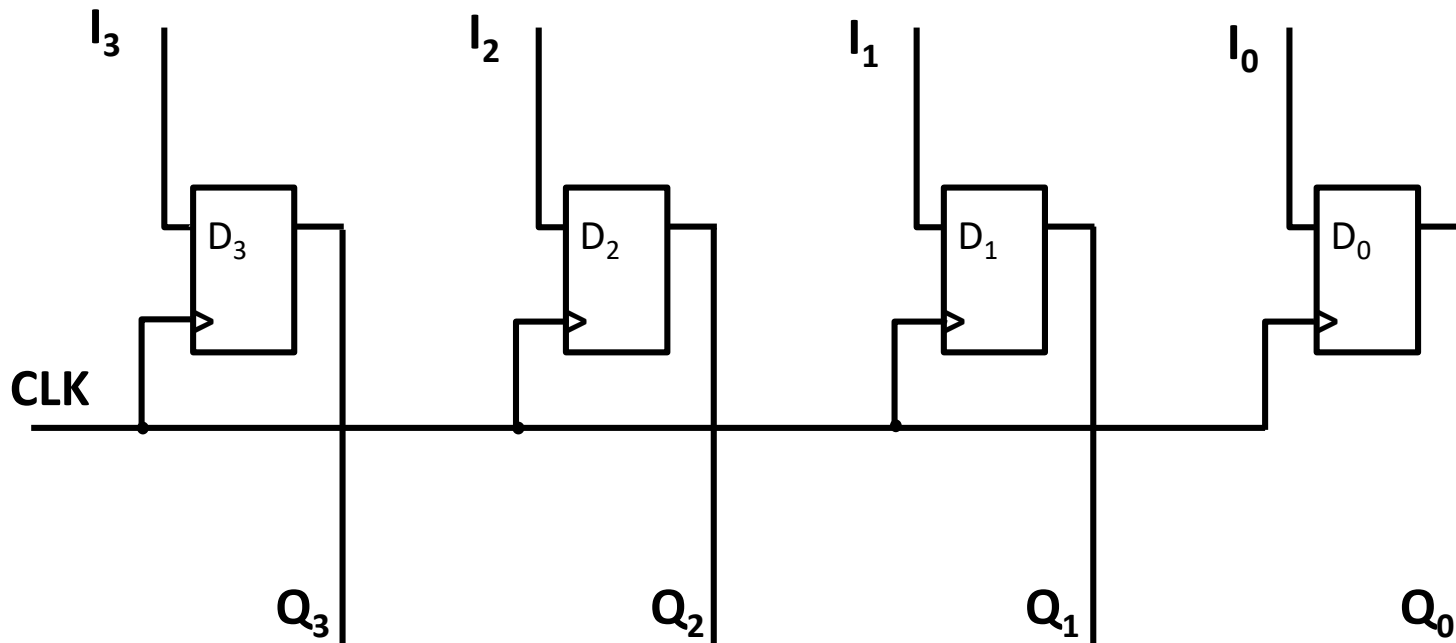
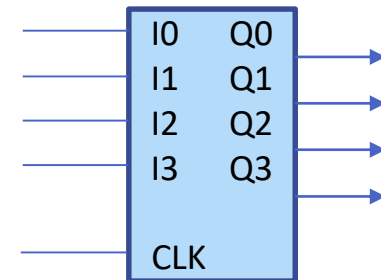


Ize i „najednou“, tzn. navrhnout dvouúrovňový obvod se 4 vstupy a 3 výstupy, a pro 4 bitovou sčítačku LO s 8 vstupy a 5 výstupy

Registry

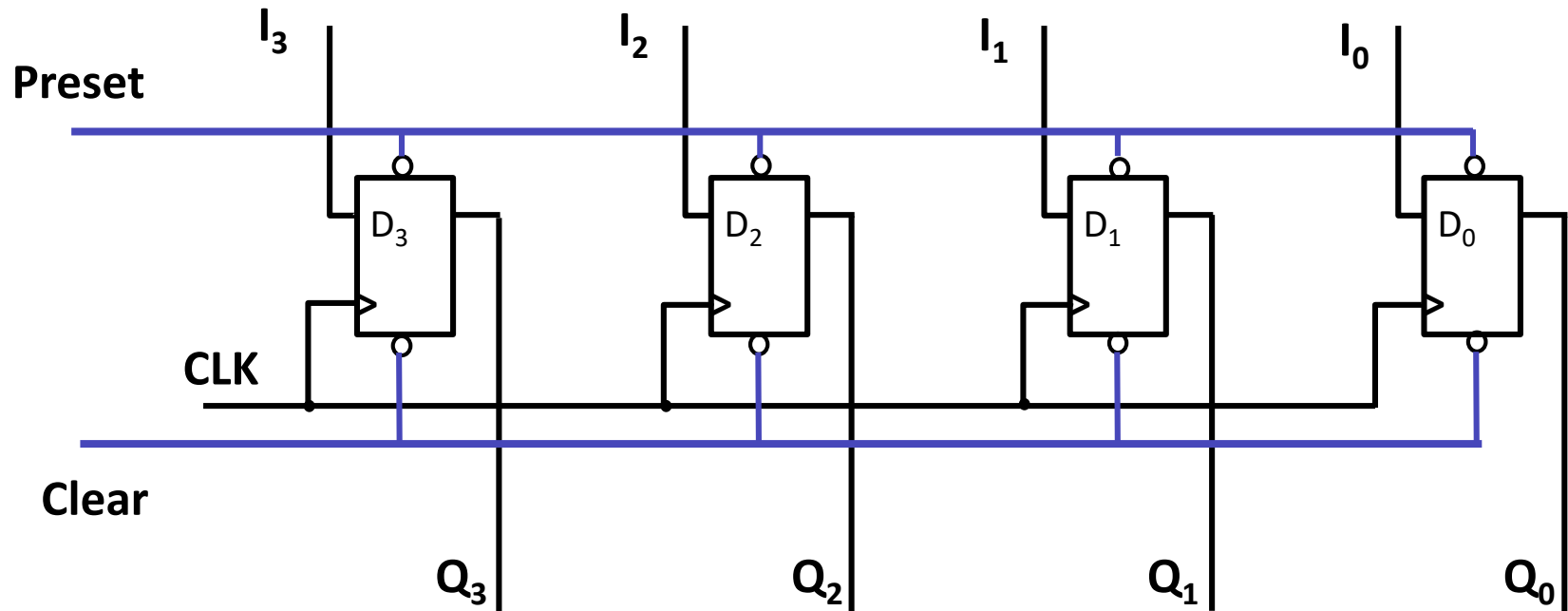
n klopných obvodů řízených
společným hodinovým signálem

zde příklad 4 bitového registru:



Otázka: jak dlouho se udrží informace v tomto registru?

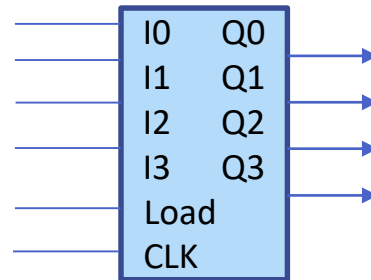
Registr s nastavením a nulováním



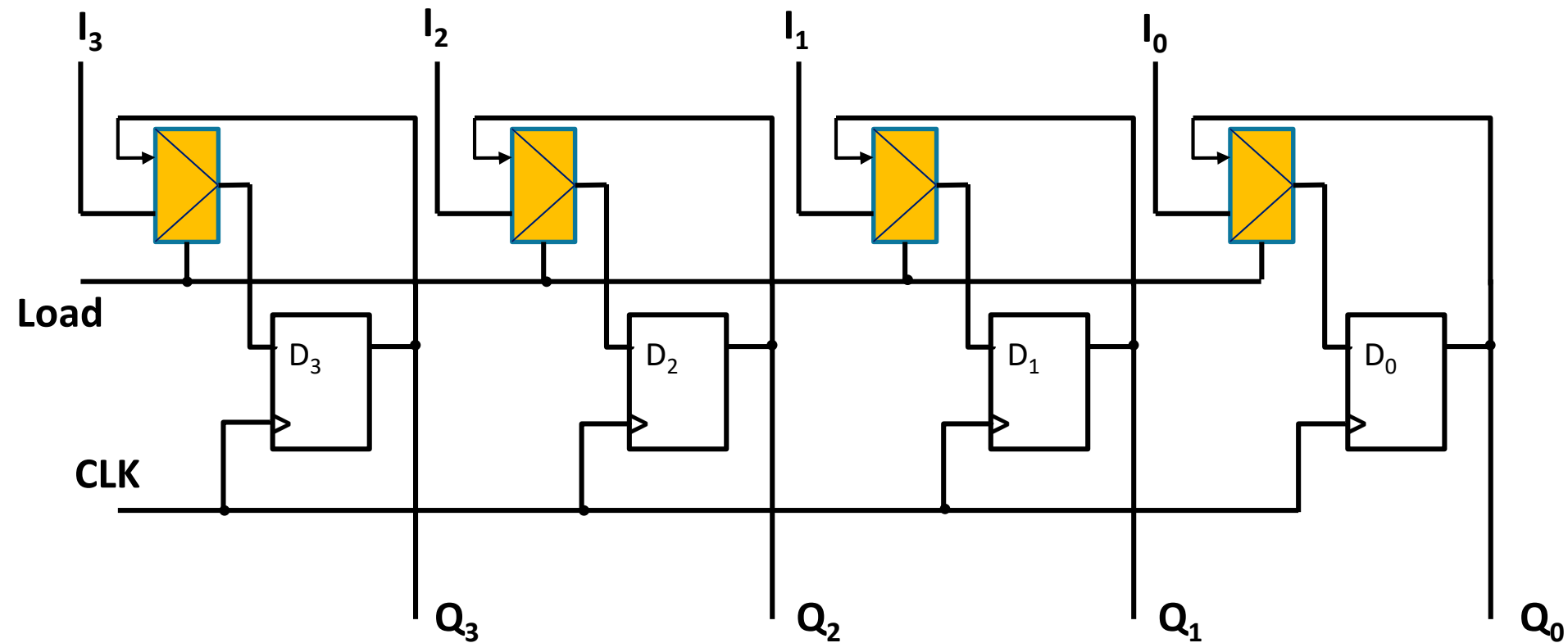
Poznámka: nastavení („preset“) na 1 nebo nulování („clear“) je asynchronní a má přednost před vstupy I_3 až I_0

Registr s řízením zápisu

„paralelní load“

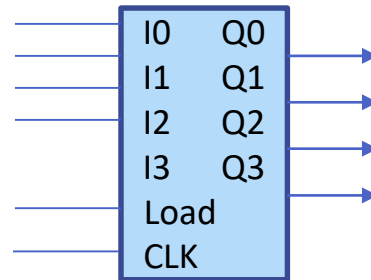


	Load=0	Load=1
Q_0	Q_0	I_0
Q_1	Q_1	I_1
Q_2	Q_2	I_2
Q_3	Q_3	I_3

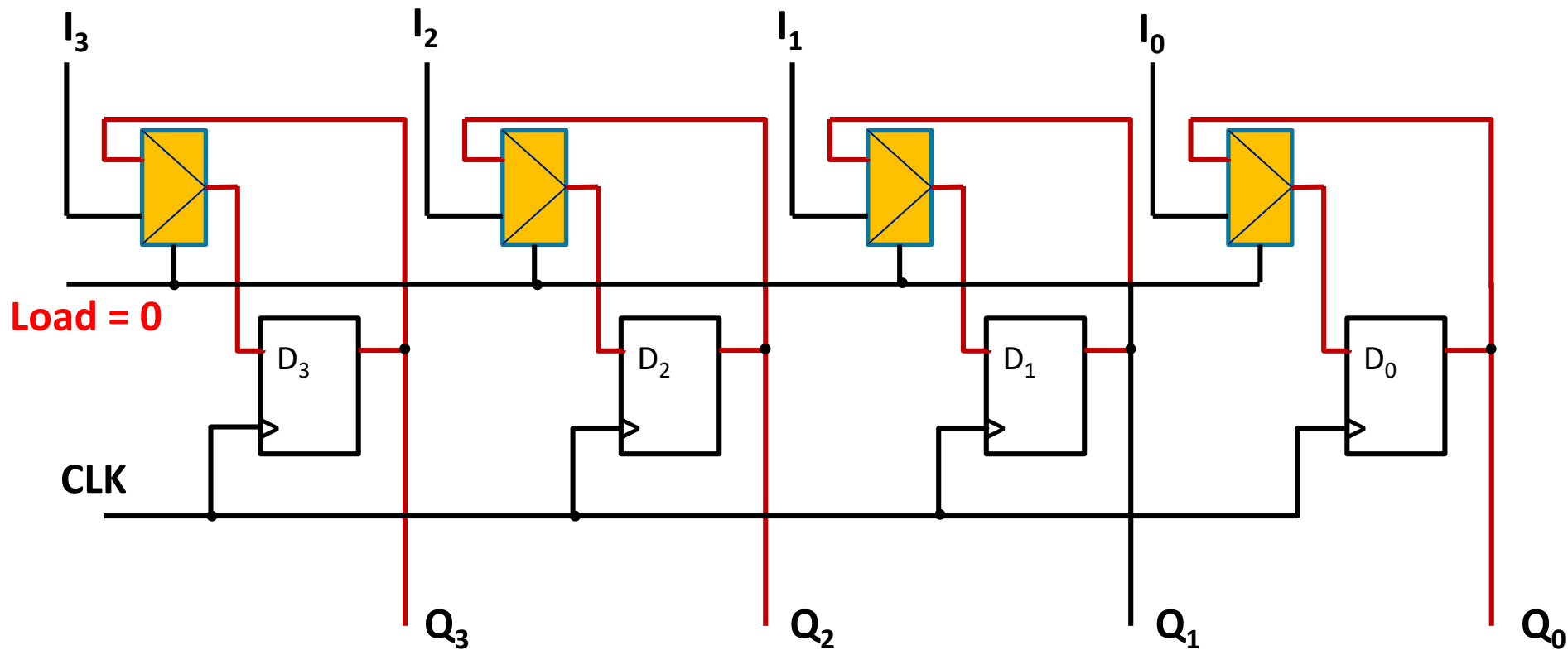


Registr s řízením zápisu

„paralelní load“

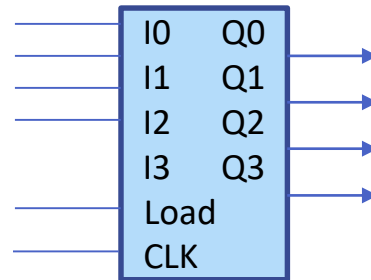


	Load=0	Load=1
Q_0	Q_0	I_0
Q_1	Q_1	I_1
Q_2	Q_2	I_2
Q_3	Q_3	I_3

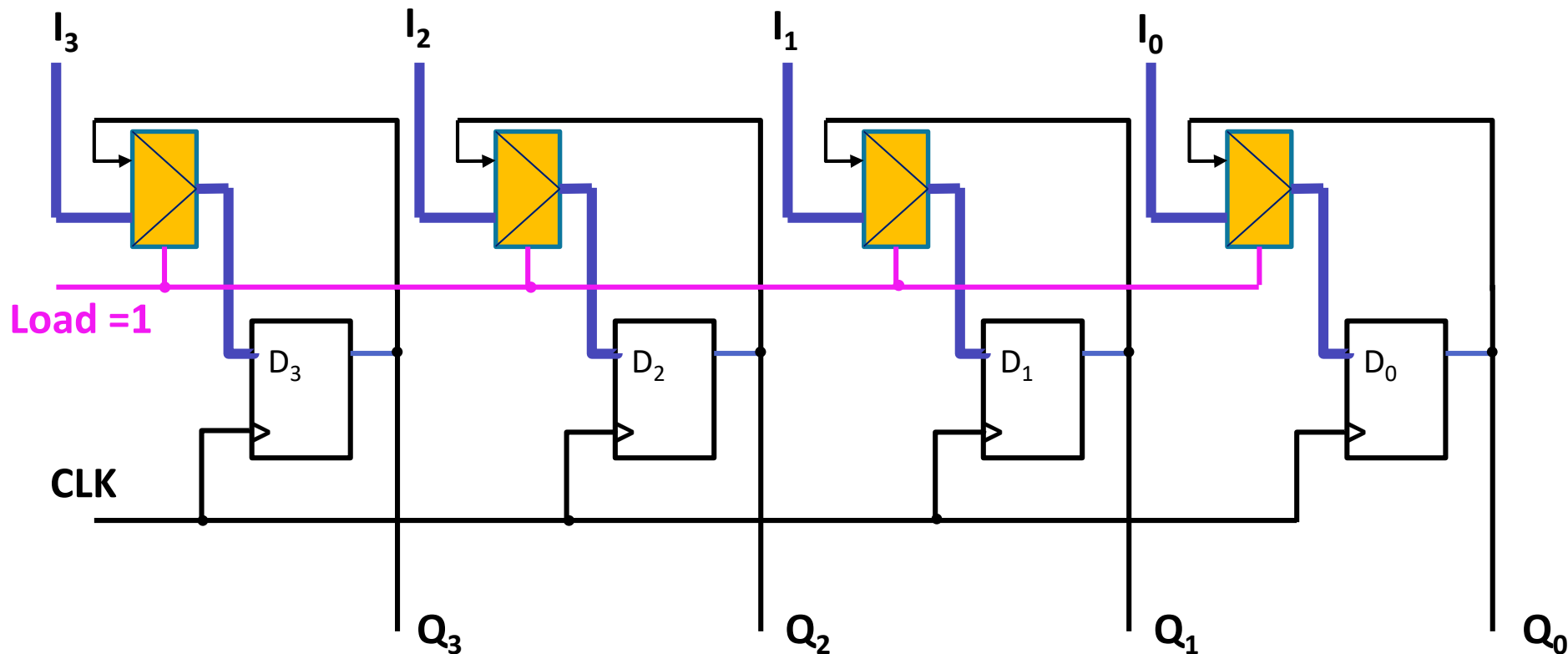


Registr s řízením zápisu

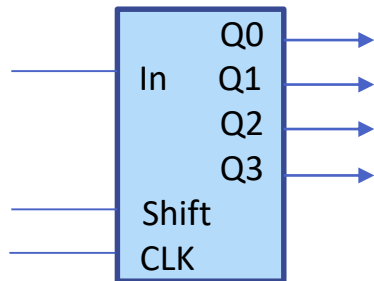
„paralelní load“



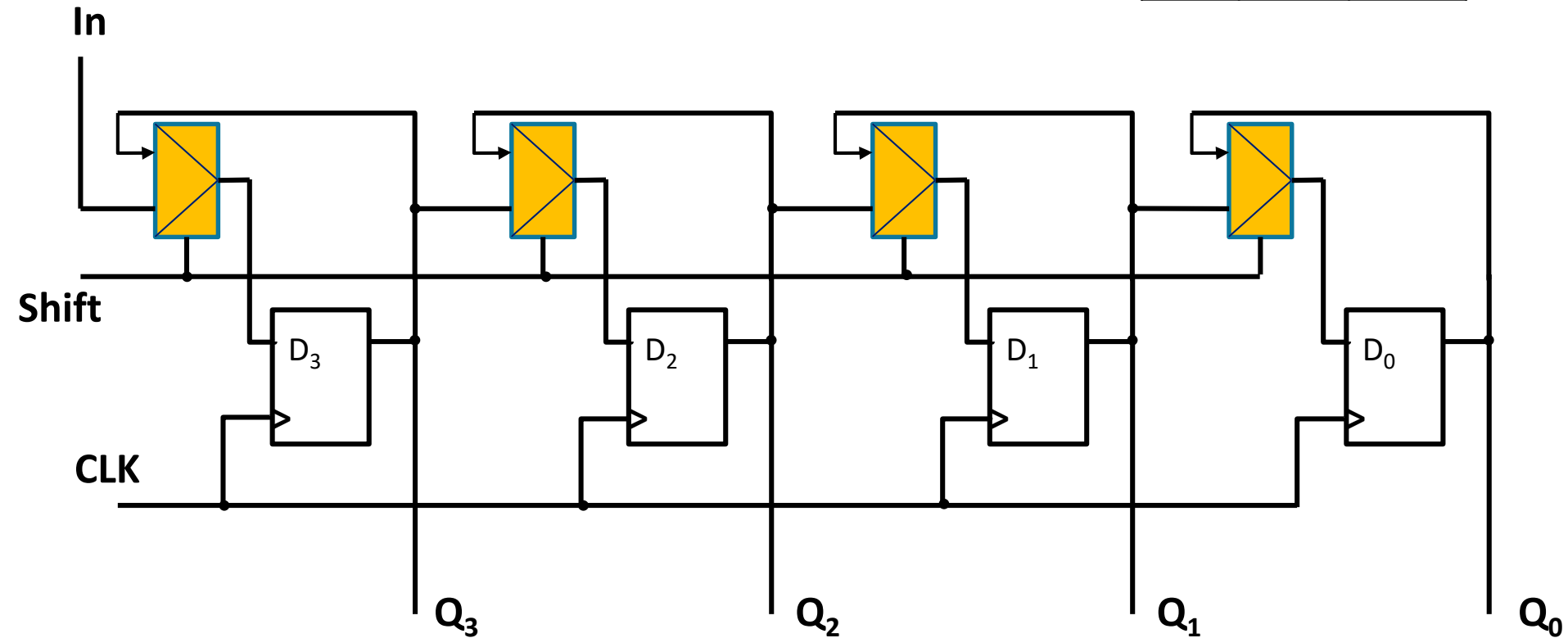
	Load=0	Load=1
Q_0	Q_0	I_0
Q_1	Q_1	I_1
Q_2	Q_2	I_2
Q_3	Q_3	I_3



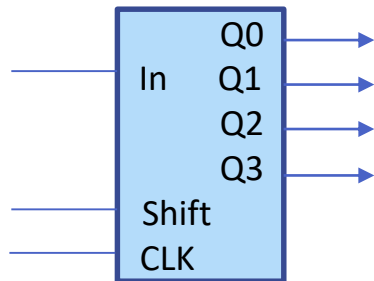
Posuvný registr



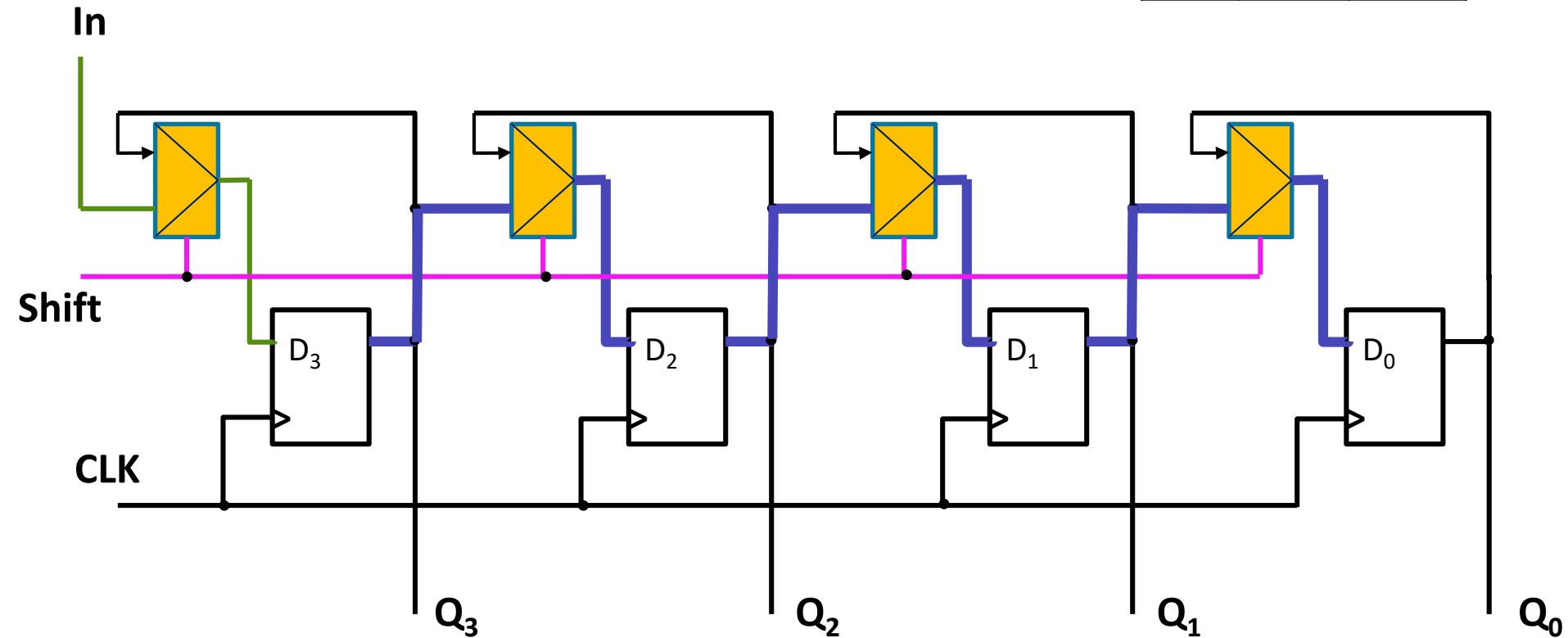
Shift	0	1
Q_0	Q_0	Q_1
Q_1	Q_1	Q_2
Q_2	Q_2	Q_3
Q_3	Q_3	In



Posuvný registr

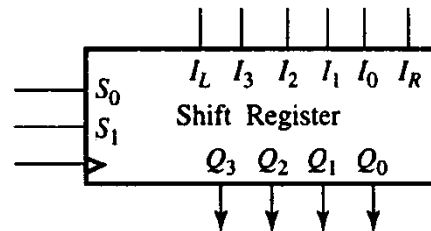


Shift	0	1
Q_0	Q_0	Q_1
Q_1	Q_1	Q_2
Q_2	Q_2	Q_3
Q_3	Q_3	In



Vícefunkční registr

Řízení zápisu
a směru posuvu:



PRESENT STATE

 S_1 S_0

0 0

0 1

1 0

1 1

OPERATION

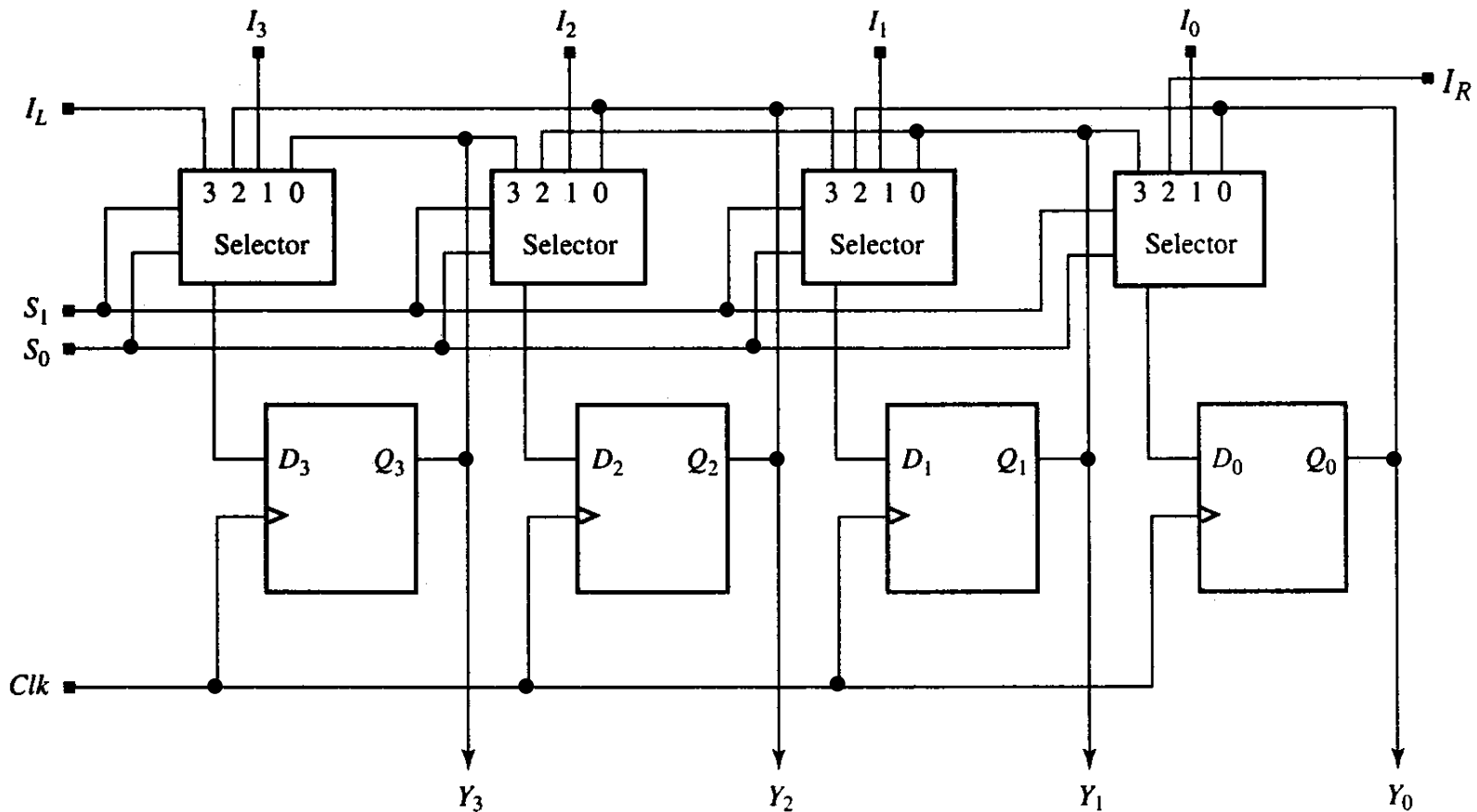
No change

Load input

Shift left

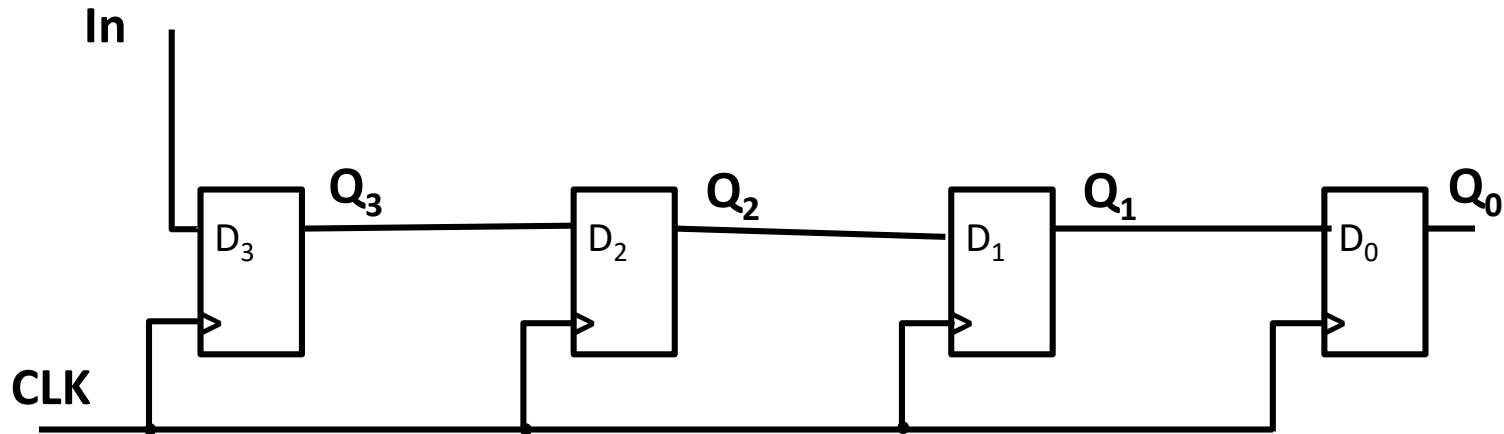
Shift right

NEXT STATE

 Q_3 Q_2 Q_1 Q_0 Q_3 Q_2 Q_1 Q_0 I_3 I_2 I_1 I_0 Q_2 Q_1 Q_0 I_R I_L Q_3 Q_2 Q_1 

Posuvný registr zjednodušený

- a v procesoru téměř nepoužitelný*



Čítače

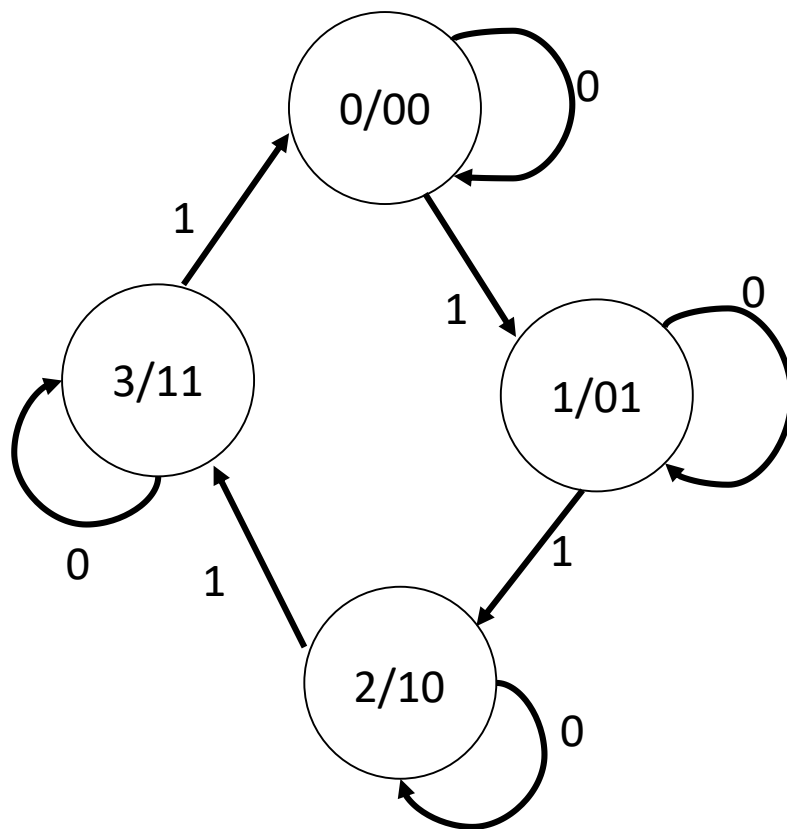
- Speciální typ registru, který v sobě zahrnuje funkci inkrementu (dekrementu) – může čítat nahoru a/nebo dolů
- Jsou tzv. úplné a neúplné čítače:
 - úplné čítače $M(\text{modulo}) 2^n$ - čítají do 4, 8, 16, 32,
 - neúplné např. do 10, 60, 80, 9, ...
- Obvykle čítají v binárním kódu
- Ale jsou čítače i v jiných kódech
 - v 1 z N
 - v Grayově kódu (aby se měnila jen jedna vnitřní proměnná)
- Jsou čítače synchronní i asynchronní

Příklad: čítač M4

Navrhněte čítač M4 se vstupem E (enable counting = povolení čítání, „počítáme“ jedničky), synchronní, v binárním kódu

typ **Moore**, graf a tabulka přechodů a výstupů:

Q/E	0	1	Y
0	0	1	00
1	1	2	01
2	2	3	10
3	3	0	11



vnitřní stavy, zde 0,1,2,3 zakódujeme podle požadovaného výstupu (00, 01, 10, 11)
ušetříme logiku pro generování výstupů,
(výstupní funkci δ)

Příklad: řešení

$\begin{array}{c} E \\ \hline q_1 q_0 \end{array}$	0	1
00	00	01
01	01	10
10	10	11
11	11	00

$\begin{array}{c} q_1 \\ \hline q_0 \end{array}$	0	1	1	0
E	1	0	0	1

$$D_{q_0} = q_0 \bar{E} + \bar{q}_0 E = q_0 \oplus E$$

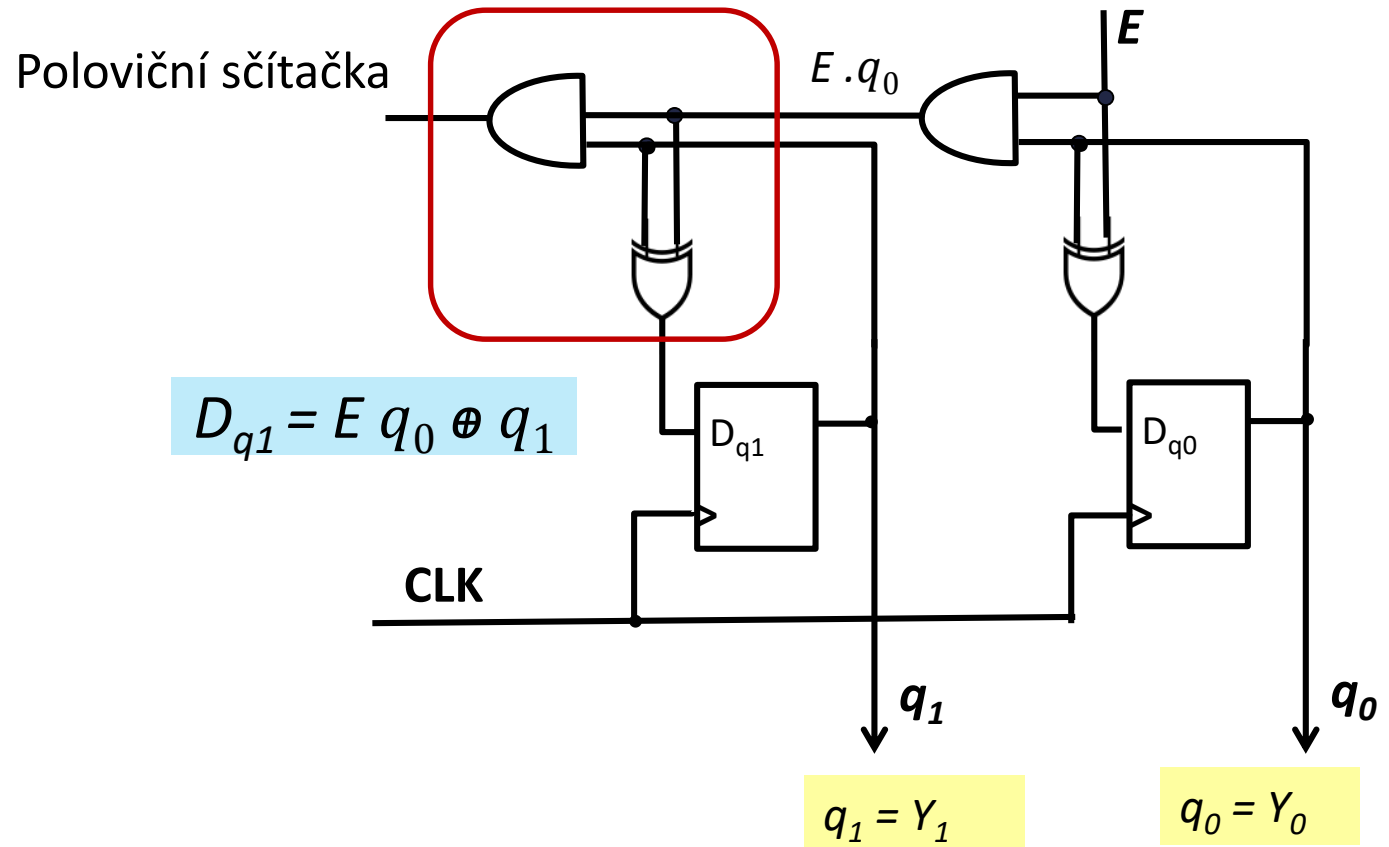
$\begin{array}{c} q_1 \\ \hline q_0 \end{array}$	0	0	1	1
E	0	1	0	1

$$\begin{aligned} D_{q_1} &= \bar{q}_0 q_1 + q_1 \bar{E} + q_0 \bar{q}_1 E = \\ &= q_1 (\bar{q}_0 + \bar{E}) + \bar{q}_1 (q_0 \cdot E) = \\ &= q_1 \cdot \overline{q_0 E} + \bar{q}_1 (q_0 \cdot E) = \\ &= E \cdot q_0 \oplus q_1 \end{aligned}$$

Výstupy: $Y_0 = q_0$ $Y_1 = q_1$

Realizace

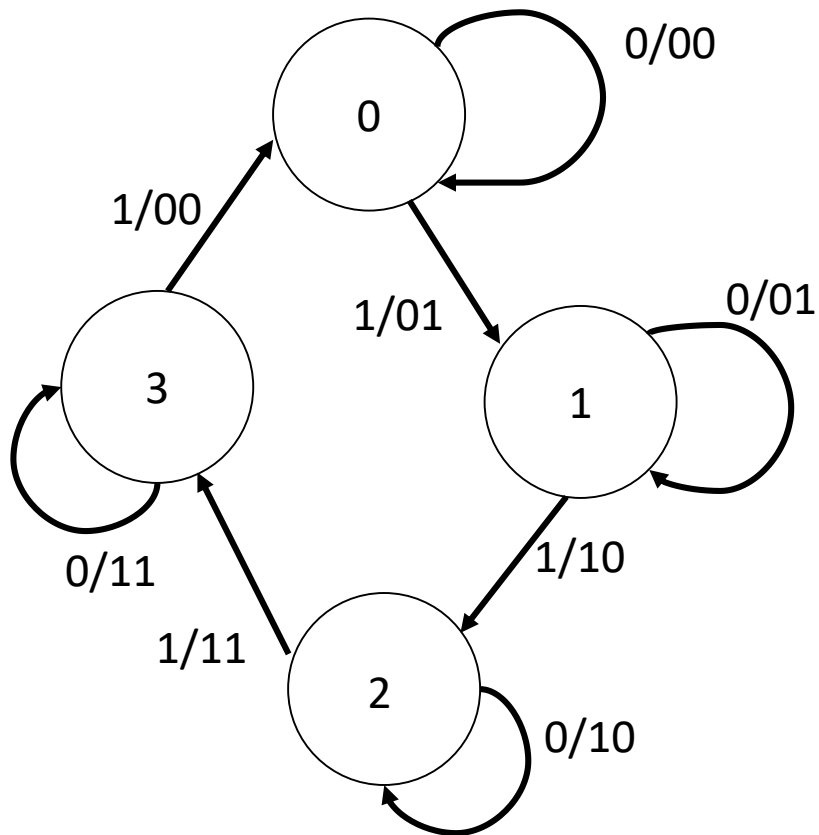
$$D_{q_0} = q_0 \overline{E} + \overline{q_0} E = q_0 \oplus E$$



Výstup je vidět až po přechodu do následného stavu Q_t
Tedy z až po aktivní hraně CLK

M4, Mealy

graf a tabulka přechodů a výstupů



Q/E	0	1	0	1
0	0	1	00	01
1	1	2	01	10
2	2	3	10	11
3	3	0	11	00

Realizace

Q/E	0	1	0	1
00	00	01	00	01
01	01	10	01	10
10	10	11	10	11
11	11	00	11	00

Zakódování (vnitřních proměnné a , b)
zvolím tak, aby odpovídalo požadovanému
kódu výstupů

0	1	1	0
1	0	0	1

Diagram illustrating a 2D array structure with dimensions b (width) and a (height). The array contains the following values:

0	1	1	0
1	0	0	1

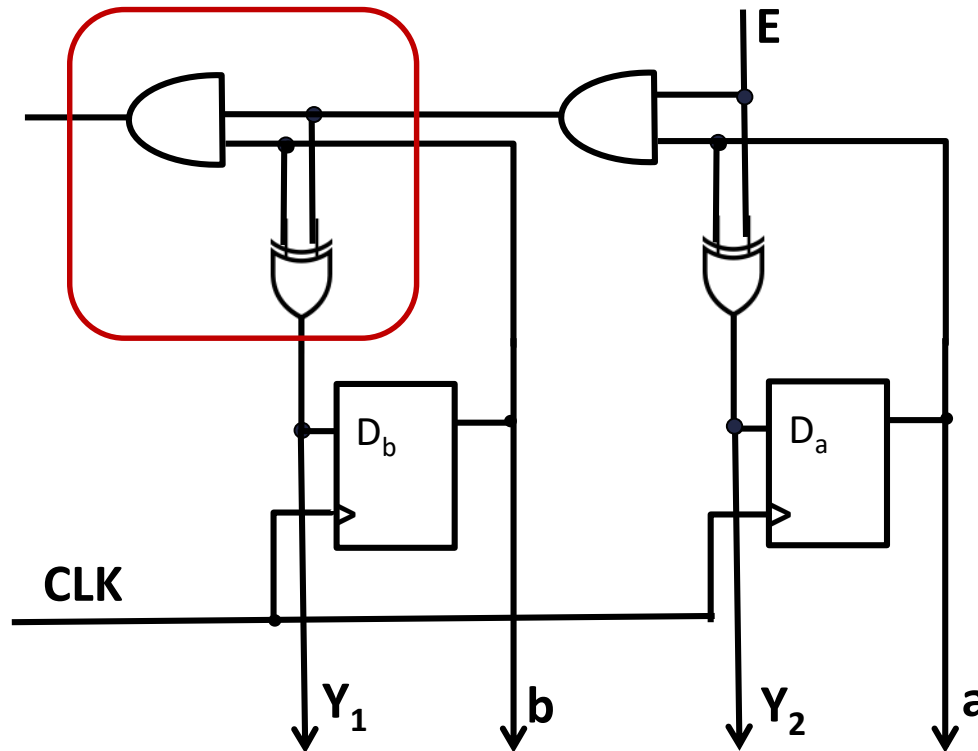
$$Y_0 = \overline{a}E + a\overline{E} = a \oplus E = D_a$$

A diagram showing a 2x4 grid of cells. The top row contains the values 0, 0, 1, 1. The bottom row contains the values 0, 1, 0, 1. To the left of the grid is a vertical line labeled 'E' below it. Above the grid are two horizontal lines: one spanning the first two columns and another spanning the last two columns. To the right of the grid are the labels 'b' and 'a' stacked vertically.

0	0	1	1
0	1	0	1

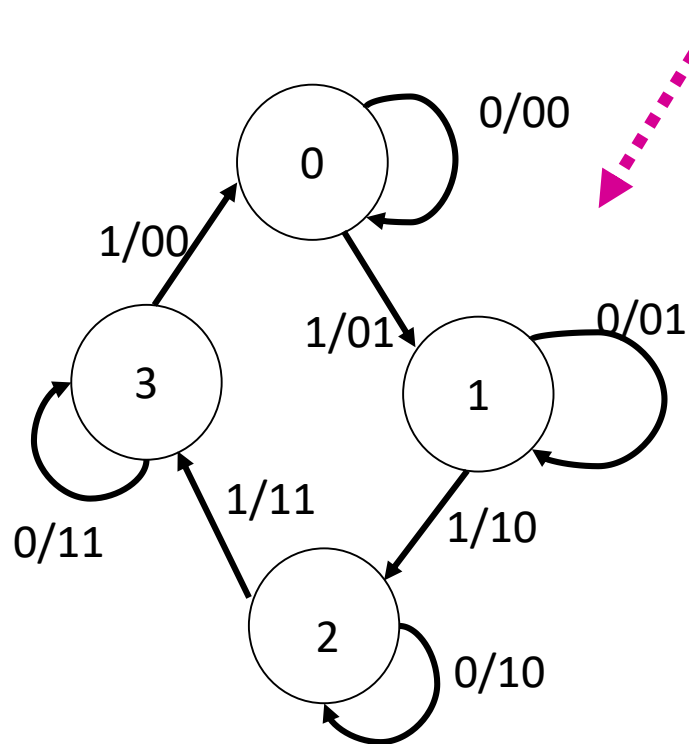
$$Y_1 = D_b = \overline{a}b + b\overline{E} + a\overline{b}E$$

Realizace

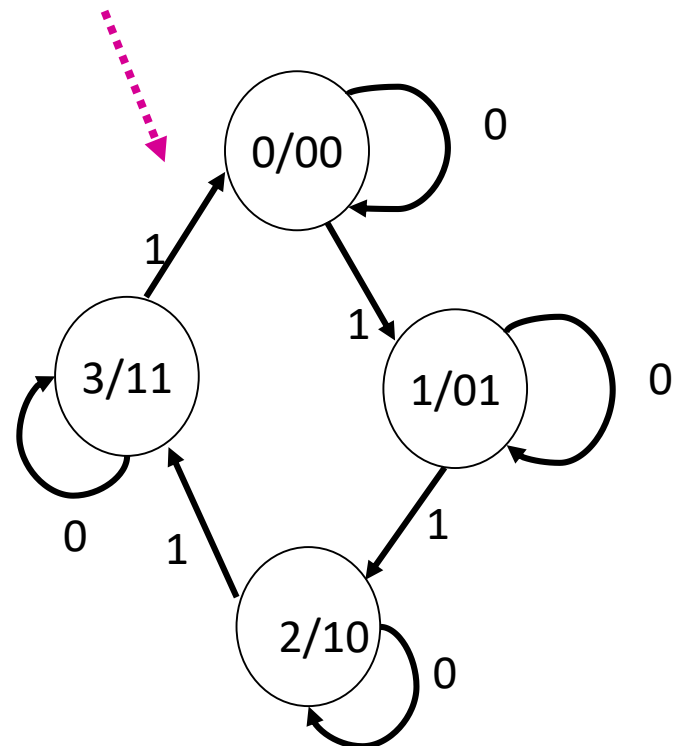


Výstup musí být vidět dříve, tzn. současně s následným stavem Q_t a ne s výchozím stavem Q_{t-1} , tedy je odvozen ze vstupů a ne z výstupů

rozdíl Mealy x Moore

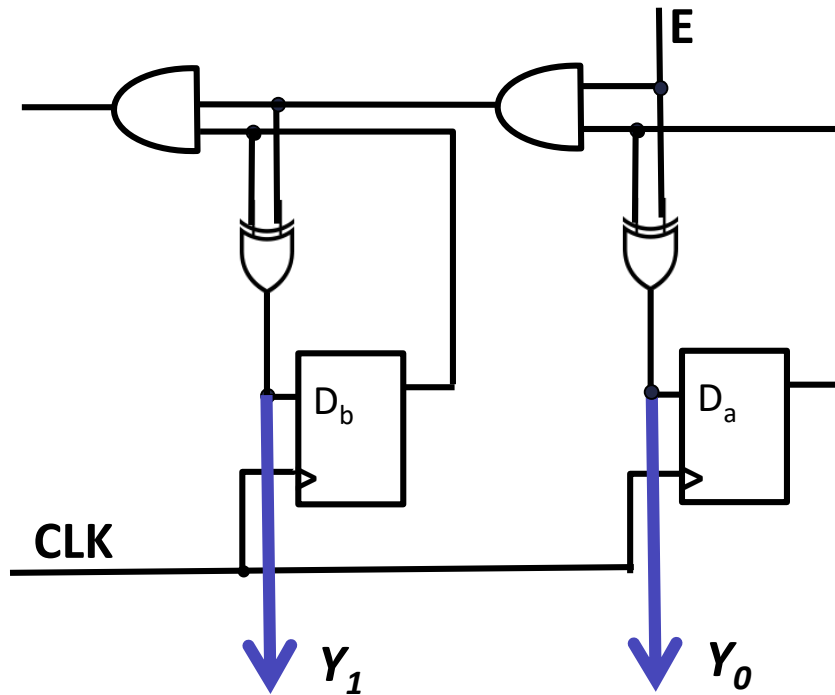


Q/E	0	1	0	1
0	0	1	00	01
1	1	2	01	10
2	2	3	10	11
3	3	0	11	00



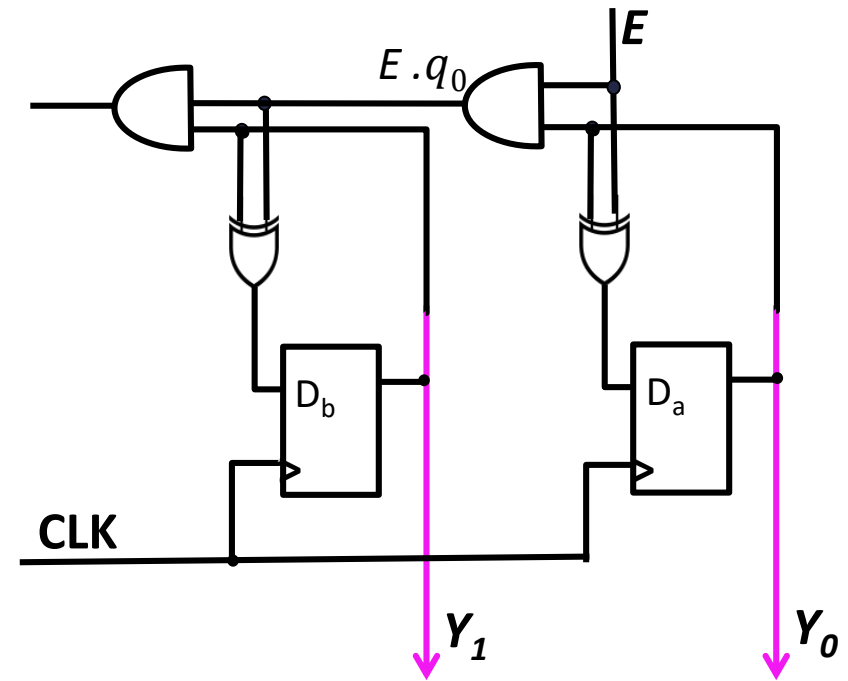
Q/E	0	1	Y
0	0	1	00
1	1	2	01
2	2	3	10
3	3	0	11

Realizace



Mealy

výstup Y ze **vstupu** D-KO
změny E se projeví na výstupu „hned“



Moore

výstup Y vyveden z **výstupu** D-KO
výstupy se změní až po aktivní hraně CLK

Čítače neúplné, vratné

- Příklad1: navrhnete čítač M5 v binárním kódu.
- Příklad2: navrhnete vratný čítač M4 v Grayově kódu, pro vstup $D=0$ čítá nahoru, pro $D=1$ dolů.
- Otázka: Jak bude vypadat čítač typu Mealy?

Řešení příkladů na prosemináři a v praktických laboratorních úlohách.

Algoritmus Quine–McCluskey

http://crc.stanford.edu/users/ejm/McCluskey_Edward.html

https://en.wikipedia.org/wiki/Quine%E2%80%93McCluskey_algorithm

- nalezení všech přímých implikantů ... algoritmicky, nalezení všech možností
- pomocí **tabulky pokrytí** nalezení minimální normální formy (nalezení podstatných implikantů a optimalizace pokrytí zbylých mintermů)
- použitelné pro více vstupních proměnných
- příklad

Příklad

- Najděte MNDF funkce:

$$f(a,b,c,d) = \sum_1 (1, 4, 7, 8, 9, 10, 11, 12, 14, 15)$$

Obvyklý postup: nalezneme všechny přímé implikanty a pak vybereme optimální pokrytí.

Mapa je vhodná do max. 5 vstupních proměnných, ale i zde je někdy problém

Následuje systematické řešení, hrubá síla:

Řešení - postup

1. jedničkové stavy zapíšeme dvojkově do tabulky:

$$f(a,b,c,d) = \sum_1 (1,4,7,8,9, 10,11,12,14,15)$$

<i>dcba</i>	<i>s pozn.</i>
0 0 0 1	1
0 1 0 0	4
0 1 1 1	7
1 0 0 0	8
1 0 0 1	9
1 0 1 0	10
1 0 1 1	11
1 1 0 0	12
1 1 1 0	14
1 1 1 1	15

Řešení - postup

2. seřadíme do skupin podle počtu jedniček:

<i>dcba</i>	<i>s pozn.</i>
0 0 0 1	1 *
0 1 0 0	4 *
0 1 1 1	7 *
1 0 0 0	8 *
1 0 0 1	9 *
1 0 1 0	10 *
1 0 1 1	11 *
1 1 0 0	12 *
1 1 1 0	14 *
1 1 1 1	15 *



<i>dcba</i>	<i>s pozn.</i>
0 0 0 1	1
0 1 0 0	4
1 0 0 0	8

1 0 0 1	9
1 0 1 0	10
1 1 0 0	12

0 1 1 1	7
1 0 1 1	11
1 1 1 0	14

1 1 1 1	15

označení, že se položka zapsala do další tabulky

Řešení - postup

3. nalezneme všechny dvojice sousedních stavů (liší se o jednu 1, hledáme jen v sousedních skupinách):

<i>dcba</i>	<i>s pozn.</i>
0 0 0 1	1 *
0 1 0 0	4 *
1 0 0 0	8 *

1 0 0 1	9 *
1 0 1 0	10 *
1 1 0 0	12 *

0 1 1 1	7 *
1 0 1 1	11 *
1 1 1 0	14 *

1 1 1 1	15 *



- 0 0 1	1,9
- 1 0 0	4,12
1 0 0 -	8,9
1 0 - 0	8,10
1 - 0 0	8,12
1 0 - 1	9,11
1 0 1 -	10,11
1 - 1 0	10,14
1 1 - 0	12,14
- 1 1 1	7,15
1 - 1 1	11,15
1 1 1 -	14,15


popis konkrétní dvojice
(podkrychle) pomocí
mintermů

* ... označení, že se položka zapsala do další tabulky

Řešení - postup

4. nalezneme všechny čtveřice sousedních stavů (opět se liší jen o jednu 1, hledáme jen v sousedních skupinách):

- 0 0 1	1,9		
- 1 0 0	4,12		
1 0 0 -	8,9 *		
1 0 - 0	8,10 *		
1 - 0 0	8,12 *		
1 0 - 1	9,11 *		
1 0 1 -	10,11 *		
1 - 1 0	10,14 *		
1 1 - 0	12,14 *		
- 1 1 1	7,15		
1 - 1 1	11,15 *		
1 1 1 -	14,15 *		



1 0 - -	8,9,10,11		
1 0 - -	8,10,9,11		
1 - - 0	8,10,12,14		
1 - - 0	8,12,10,14		
1 - 1 -	10,11,14,15		
1 - 1 -	10,14,11,15		

* ... označení, že se položka zapsala do další tabulky

Řešení - postup

5. pokud již nejde dál slučovat (hledat větší krychle), označíme si všechny neohvězdičkové stavy (= přímé implikanty) A až F

<i>dcba</i>	<i>s, s pozn.</i>	<i>dcba</i>	<i>s, s, s, s pozn.</i>
- 0 0 1	1,9 F	1 0 - -	8,9,10,11 C
- 1 0 0	4,12 E	1 0 - -	8,10,9,11
1 0 0 -	8,9 *	1 - - 0	8,10,12,14 B
1 0 - 0	8,10 *	1 - - 0	8,12,10,14
1 - 0 0	8,12 *		
		1 - 1 -	10,11,14,15 A
1 0 - 1	9,11 *	1 - 1 -	10,14,11,15
1 0 1 -	10,11 *		
1 - 1 0	10,14 *		
1 1 - 0	12,14 *		
- 1 1 1	7,15 D		
1 - 1 1	11,15 *		
1 1 1 -	14,15 *		

Řešení - postup

6. Tabulka pokrytí:

sloupce: všechny mintermy (stavové indexy pro zadané „1“)

řádky: všechny přímé implikanty

obsah: * označuje které mintermy implikant pokrývá

		1	4	7	8	9	10	11	12	14	15
A	bd						*	*		*	*
B	$\bar{a}d$				*		*		*	*	
C	$\bar{c}d$				*	*	*	*			
D	abc			*							*
E	$\bar{a}\bar{b}c$		*						*		
F	$a\bar{b}\bar{c}$	*				*					
poznámky		↑	↑	↑		×			×		×

6. Tabulka pokrytí: postup

		1	4	7	8	9	10	11	12	14	15
A	bd						*	*		*	*
B	$\bar{a}d$				*		*		*	*	
C	$\bar{c}d$				*	*	*	*			
D	abc			*							*
E	$\bar{a}\bar{b}c$		*						*		
F	$a\bar{b}\bar{c}$	*				*					
poznámky		↑	↑	↑		×			×		×

a) hledáme sloupce s jednou * = podstatné implikanty (pokrývají 1, 4, 7)

$$f(a, b, c, d) = a\bar{b}\bar{c} + \bar{a}\bar{b}c + abc + \dots$$

b) v řádcích vyškrtneme již pokryté sloupce (mintermy) ... x

7. Výběr z přímých implikantů

		1	4	7	8	9	10	11	12	14	15
A	bd						*	*		*	*
B	$\bar{a}d$				*		*		*	*	
C	$\bar{c}d$				*	*	*	*			
D	abc			*							*
E	$\bar{a}\bar{b}c$		*						*		
F	$a\bar{b}\bar{c}$	*				*					
poznámky		↑	↑	↑		×			×		×

$$f(a, b, c, d) = a\bar{b}\bar{c} + \bar{a}\bar{b}c + abc + \dots$$

jen překreslení

		8	10	11	14
A	bd		*	*	*
B	$\bar{a}d$	*	*		*
C	$\bar{c}d$	*	*	*	

dominantní sloupec => bude pokrytý při všech výběrech implikantů

Výsledky, tabulka pokrytí

		1	4	7	8	9	10	11	12	14	15
A	bd						*	*		*	*
B	$\bar{a}d$				*		*		*	*	
C	$\bar{c}d$				*	*	*	*			
D	abc			*							*
E	$\bar{a}\bar{b}c$		*						*		
F	$a\bar{b}\bar{c}$	*				*					
poznámky		↑	↑	↑		×			×		×

podstatné implikanty

$$f(a, b, c, d) = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + abc + bd + \bar{a}d$$

$$f(a, b, c, d) = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + abc + bd + \bar{c}d$$

$$f(a, b, c, d) = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + abc + \bar{a}d + \bar{c}d$$

výběr přímých implikantů

		8		11	14
A	bd			*	*
B	$\bar{a}d$	*			*
C	$\bar{c}d$	*		*	

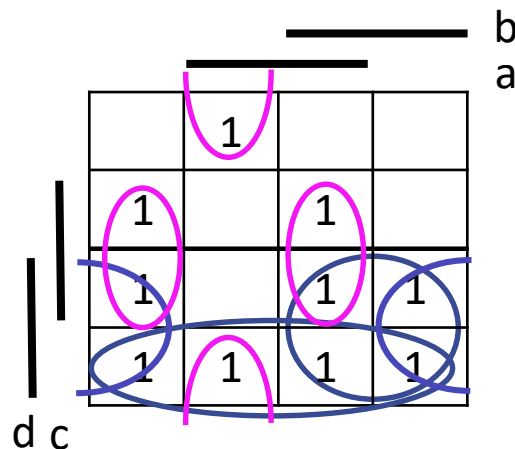
Musíme vybrat libovolné dva ze tří implikantů
 $bd, \bar{a}d, \bar{c}d$

Totéž v mapě

Najděte MNDF funkce:

$$f(a,b,c,d) = \sum_1 (1, 4, 7, 8, 9, 10, 11, 12, 14, 15)$$

Opět nalezneme všechny přímé implikanty:



$bd, \bar{a}d, \bar{c}d$
 $abc, \bar{a}\bar{b}c, a\bar{b}\bar{c}$

Ale může být problém s nalezením optimálního pokrytí
 → lze využít Tabulku pokrytí i zde