ELEKTRONIKA

Viera Stopjaková (viera.stopjakova@stuba.sk)
Ústav elektroniky a fotoniky
FEI STU

Kombinačné obvody

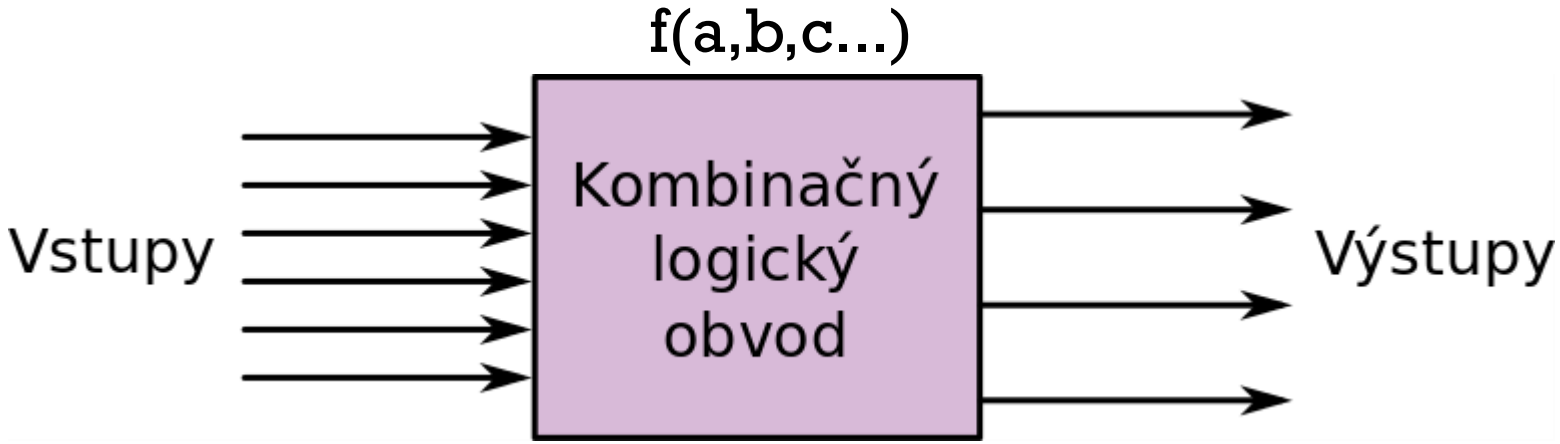
Prednáška 9

Obsah

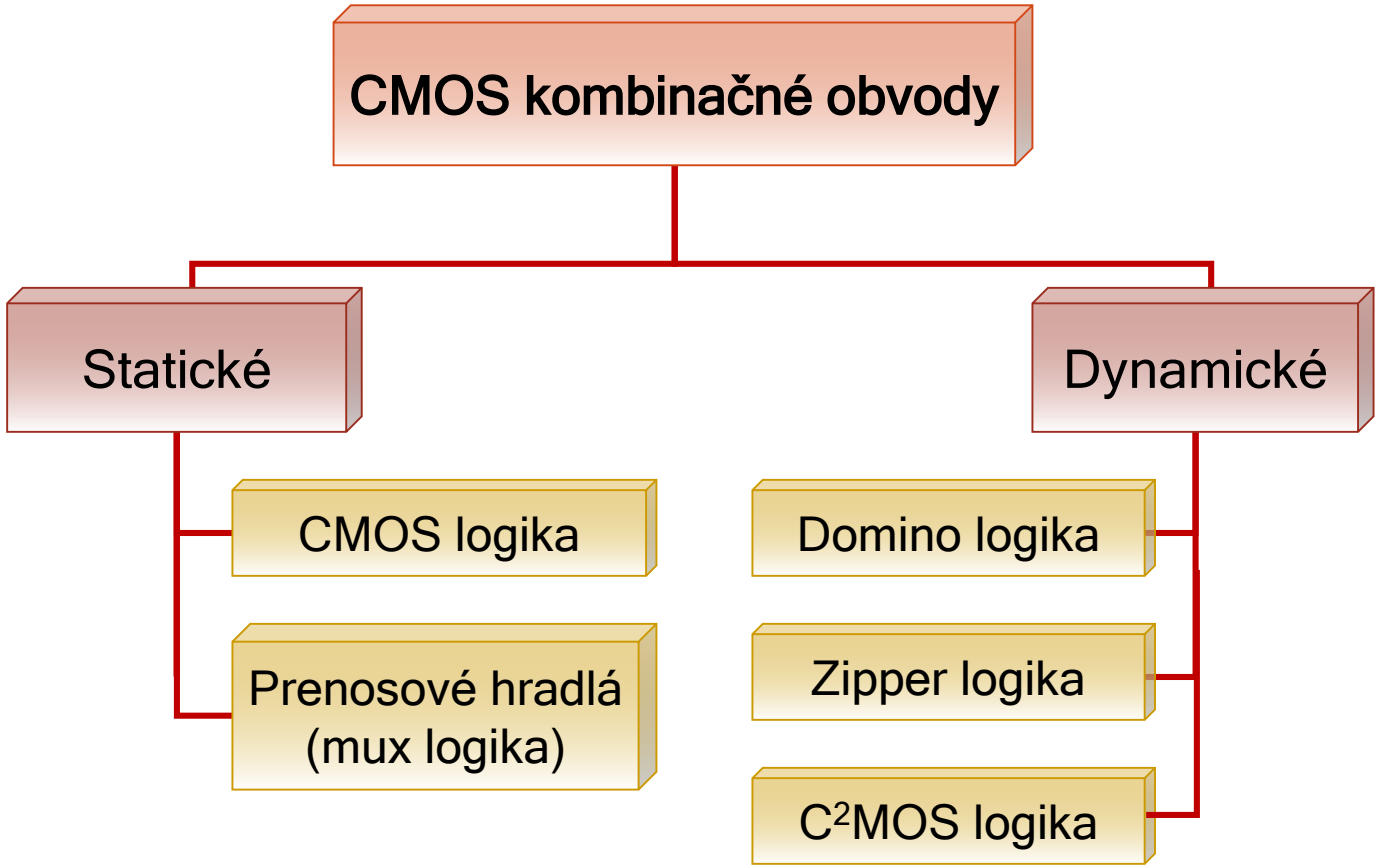
- **Kombinačné logické obvody a ich realizácia**
  - Charakteristika a rozdelenie KO
  - Najpoužívanejšie KO
- Parametre kombinačných obvodov
  - Logický zisk
  - Hazardy

Charakteristika KO

- Obvody zložené zo základných a komplexných logických členov
  - výstupné dáta **závisia** len od vstupných premenných (dát)
  - obvod je opísateľný Boolovskou funkciou alebo pravdivostnou tabuľkou
  - časovo nezávislá funkcia



Rozdelenie KO (v CMOS)

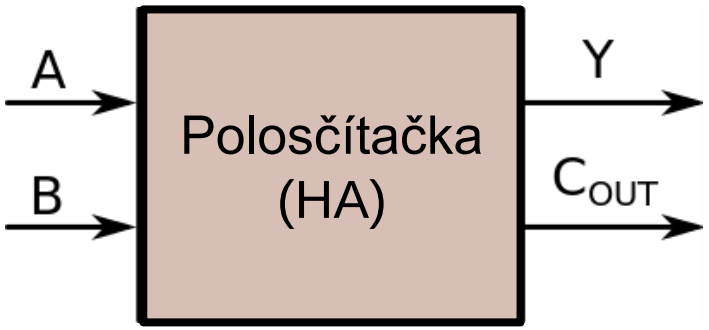


Kombinačné obvody

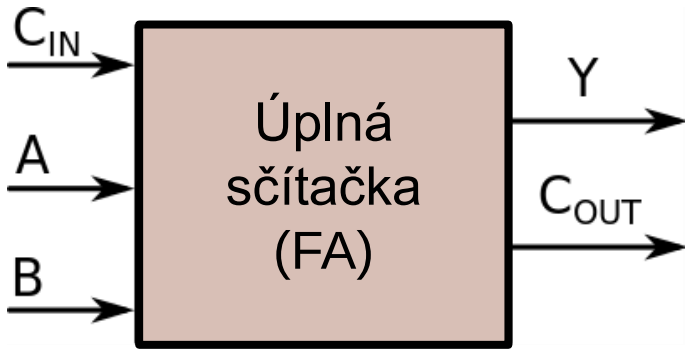
- Najpoužívanejšie kombinačné obvody
  - Aritmetická sčítačka / odčítačka
  - Multiplexor / Demultiplexor
  - Selektor
  - Komparátor
  - Kóder / Dekóder
  - Boothova bunka

Aritmetická sčítačka

- Aritmeticky sčíta dve čísla v (priamom) binárnom kóde
  - Niekoľko topológií a architektúr
- **Polosčítačka (Half Adder - HA)**
  - neuvažuje **vstupný** prenos z vyššieho rádu



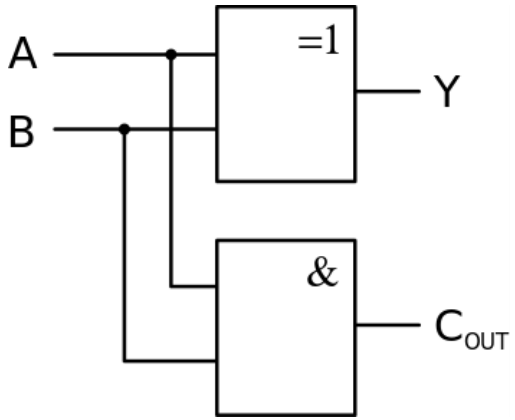
- **Úplná sčítačka (Full Adder - FA)**
  - uvažuje vstupný prenos

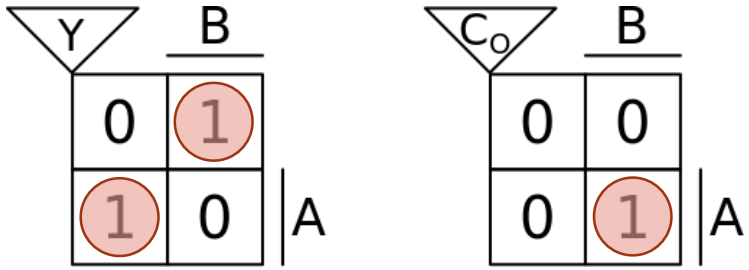


Aritmetická sčítačka

- Navrhnite 1-bitovú aritmetickú **polosčítačku** (HA)

		2^1	2^0
A	**B**	**C_{OUT}**	**Y**
0	0	**0**	**0**
0	1	**0**	**1**
1	0	**0**	**1**
1	1	**1**	**0**





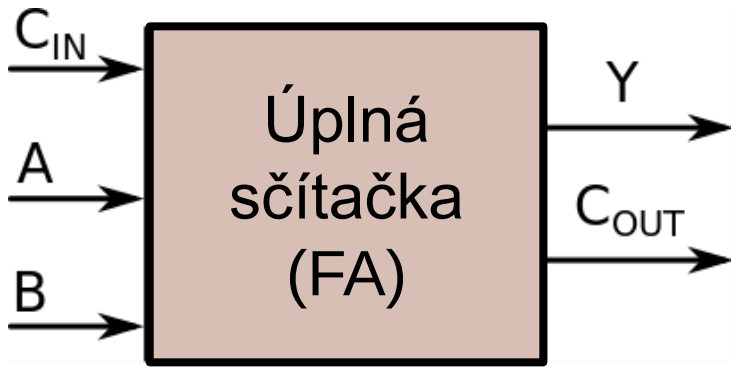
$$Y = \overline{A}.B + A.\overline{B} = A \oplus B$$

$$C_{OUT} = A.B$$

Aritmetická sčítačka

- Navrhnite 1-bitovú aritmetickú **úplnú sčítačku** (FA)

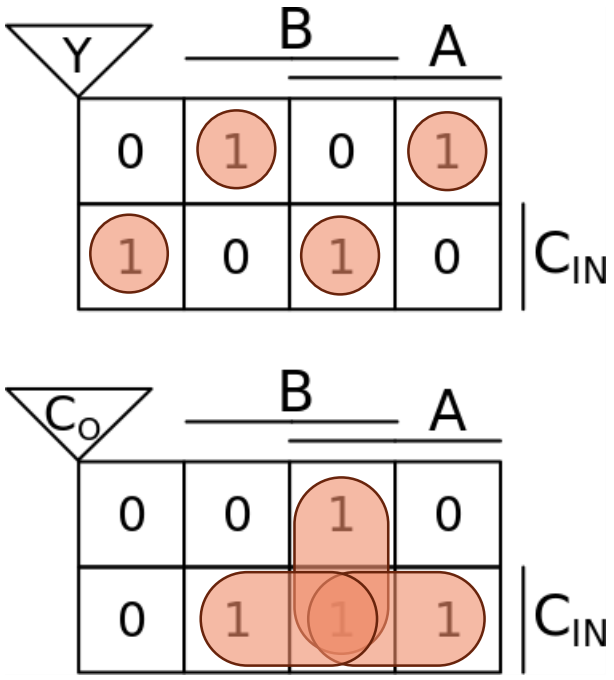
			2^1	2^0
C_{IN}	A	B	C_{OUT}	Y
0	0	0	**0**	**0**
0	0	1	**0**	**1**
0	1	0	**0**	**1**
0	1	1	**1**	**0**
1	0	0	**0**	**1**
1	0	1	**1**	**0**
1	1	0	**1**	**0**
1	1	1	**1**	**1**



Aritmetická sčítačka

- Navrhnite 1-bitovú aritmetickú **úplnú sčítačku** (FA)

			2^1	2^0
C_{IN}	A	B	C_{OUT}	Y
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**



$$Y = A \oplus B \oplus C_{IN}$$

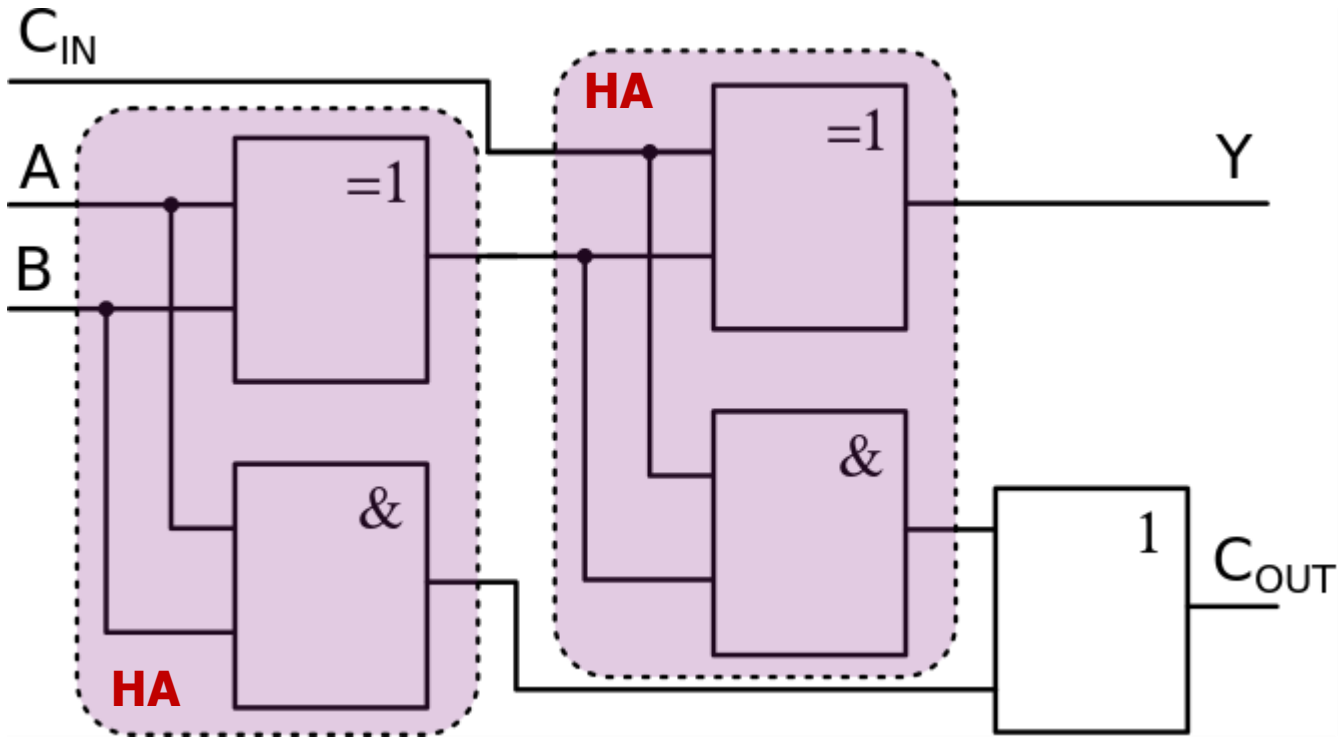
$$C_{OUT} = A.B + C_{IN}.A + C_{IN}.B$$

Aritmetická sčítačka

- Navrhnite 1-bitovú aritmetickú **úplnú sčítačku** (FA)

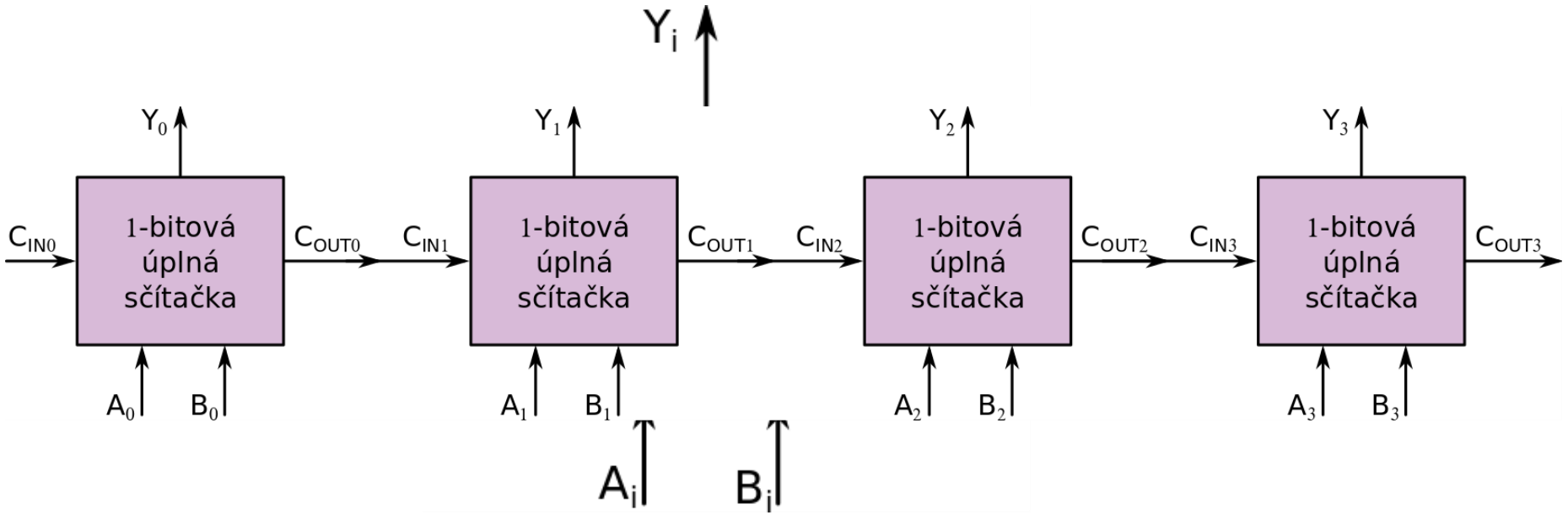
$$Y = A \oplus B \oplus C_{IN} = (A \oplus B) \oplus C_{IN}$$

$$C_{OUT} = A.B + C_{IN}.A + C_{IN}.B = A.B + C_{IN}.(A \oplus B)$$



Aritmetická sčítačka

- **Paralelná 4-bitová sčítačka/odčítačka** – najjednoduchší prípad
  - Zložená z identických segmentov
  - Posúvanie prenosu z predchádzajúceho rádu
  - Jednoduchá implementácia
  - Neoptimálne parametre – šírenie prenosu do vyššieho rádu



Rôzne typy sčítačiek

- **Carry look-ahead** sčítačka
- Carry skip, Carry select, Carry save sčítačka
- Pre-Fix sčítačka
- Pipeline paralelná sčítačka
- Multi-operandová sčítačka
- Manchesterská sčítačka

Rôzne parametre: spotreba, zložitosť, výkon, rýchlosť...

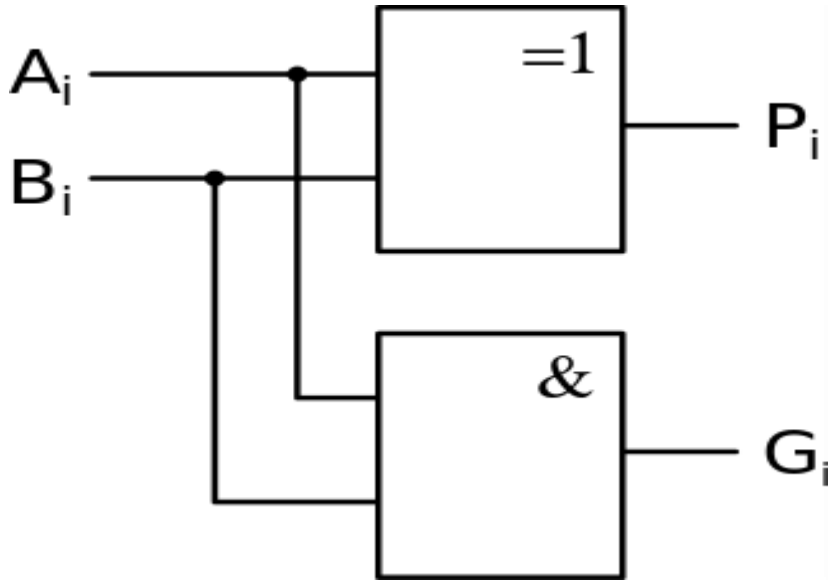
Carry look-ahead (CLA)

- Pred samotným sčítaním vypočíta propagáciu (šírenie) prenosu do vyššieho rádu na základe vstupných dát
  - Redukovaný čas potrebný na výpočet prenosu
  - Vyššia zložitosť a spotreba obvodu (hlavne pri viacbitovom slove)

- Zložená z troch častí:
  - **Propagate / Generate** generátor
  - **Look-Ahead Carry** generátor
  - **Sumátor**

Carry look-ahead (CLA)

- **Propagate/Generate generátor** buď vygeneruje prenos do vyššieho rádu alebo ho len prenesie z nižšieho rádu
  - Krok je vykonaný na všetkých vstupných bitoch naraz, oneskorenie tvorí teda len jedno hradlo



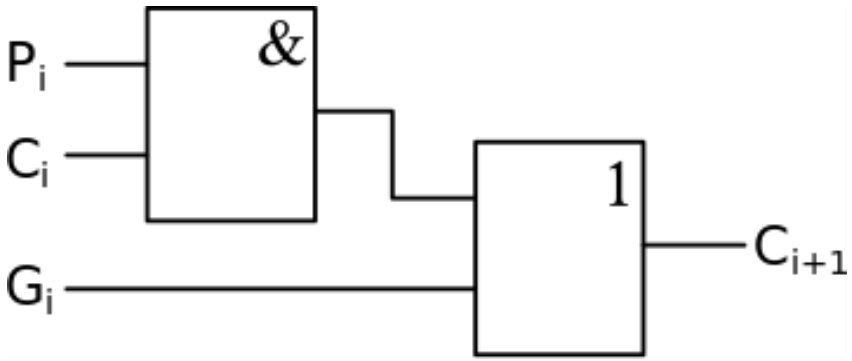
$$P_i = A_i \oplus B_i$$

$$G_i = A_i B_i$$

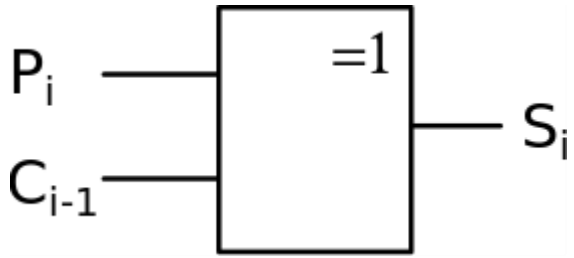
Carry look-ahead (CLA)

- **Look-Ahead Carry** a **Sumátor** moduly pracujú na základe signálov z Propagate/Generate modulu a prenosu z nižšieho rádu

$$C_{i+1} = G_i + P_i C_i$$



$$S_i = P_i \oplus C_{i-1}$$



Carry look-ahead (CLA)

- **4-bitová CLA sčítačka**

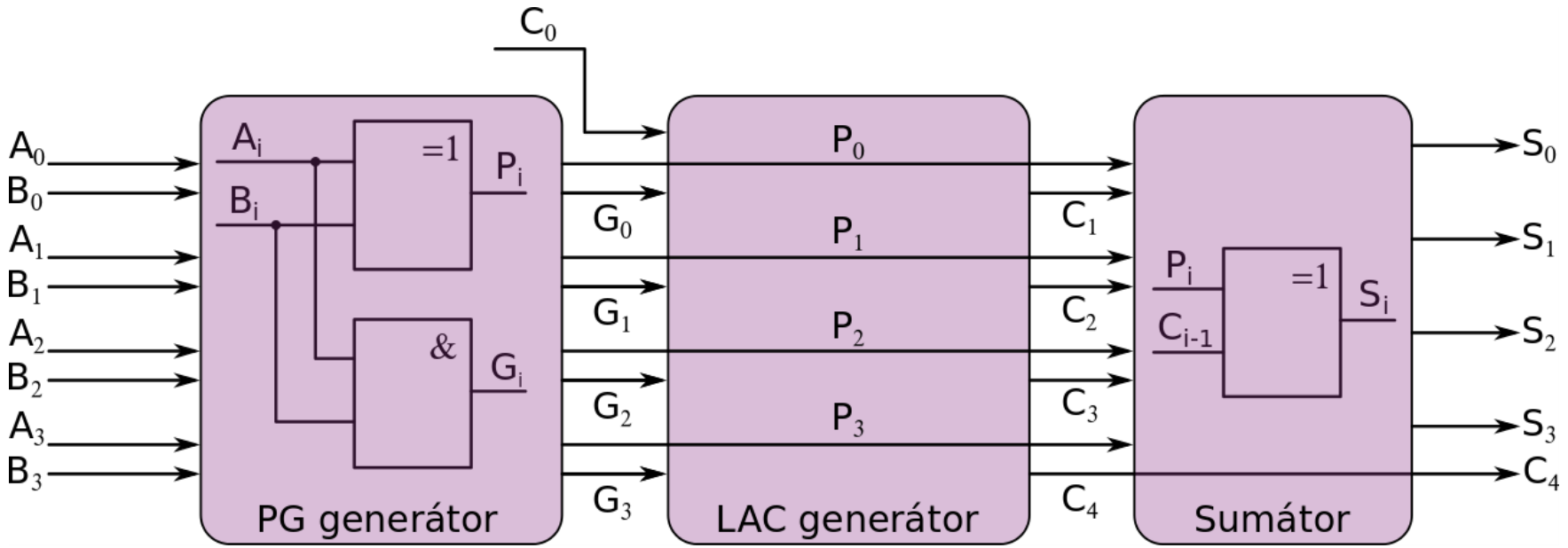
$$C_{i+1} = G_i + P_i C_i$$

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 (G_0 + P_0 G_0) = G_1 + P_1 G_0 + P_1 P_0 C_0$$

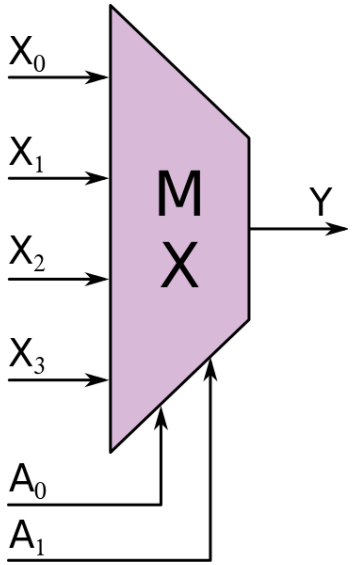
$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

$$C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$$

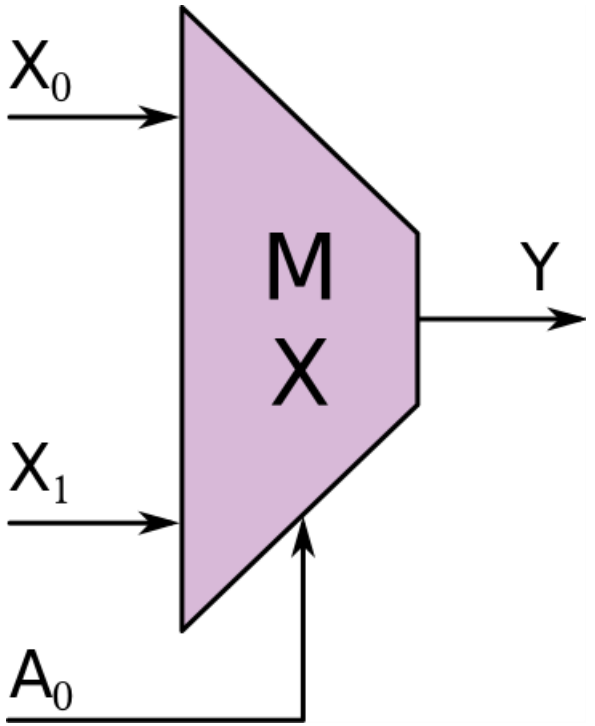


Multiplexor

- Prenáša dáta z vybraného vstupného kanála na výstup
- Dátové vstupné kanály (X_0 a X_1)
- Adresné (selektovacie) vstupy (A_0)
- Jeden dátový výstup (Y)



MUX 1 zo 4

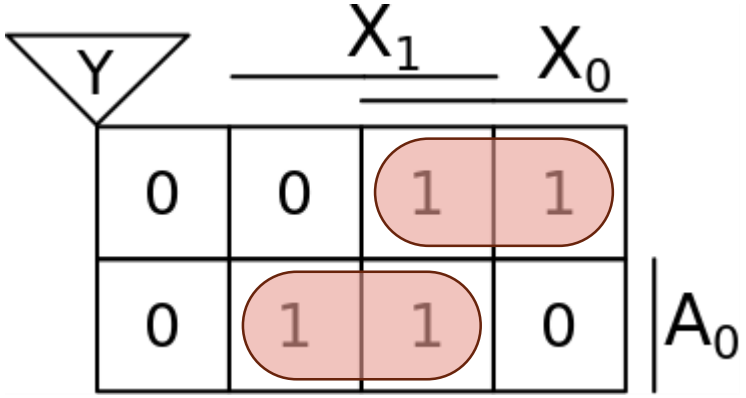


MUX 1 z 2

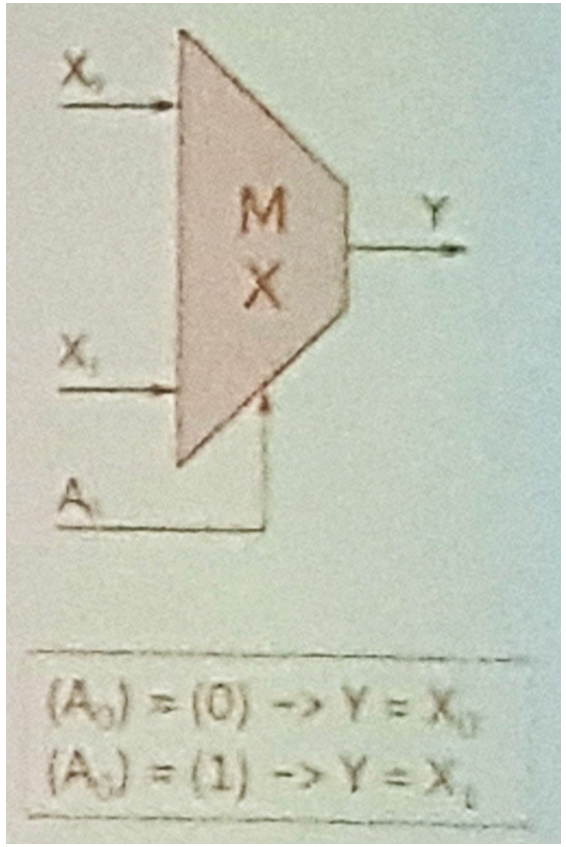
Multiplexor

- Navrhnite **multiplexor** s dvomi dátovými vstupmi (MUX 1 z 2)
  - A_0 je selektovací vstup

A_0	X_0	X_1	Y
0	0	0	**0**
0	0	1	**0**
0	1	0	**1**
0	1	1	**1**
1	0	0	**0**
1	0	1	**1**
1	1	0	**0**
1	1	1	**1**



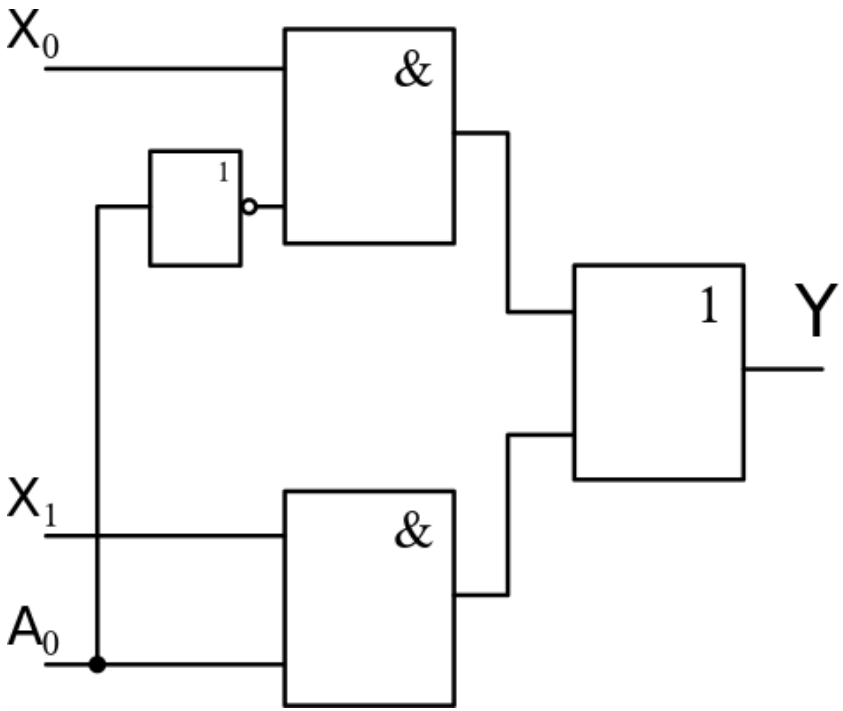
$$Y = A_0.X_1 + \overline{A_0}.X_0$$

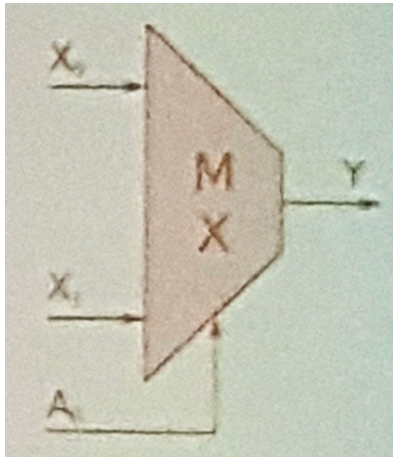


Multiplexor

- Navrhnite **multiplexor** s dvomi dátovými vstupmi (MUX 1 z 2)
  - A_0 je selektovací vstup

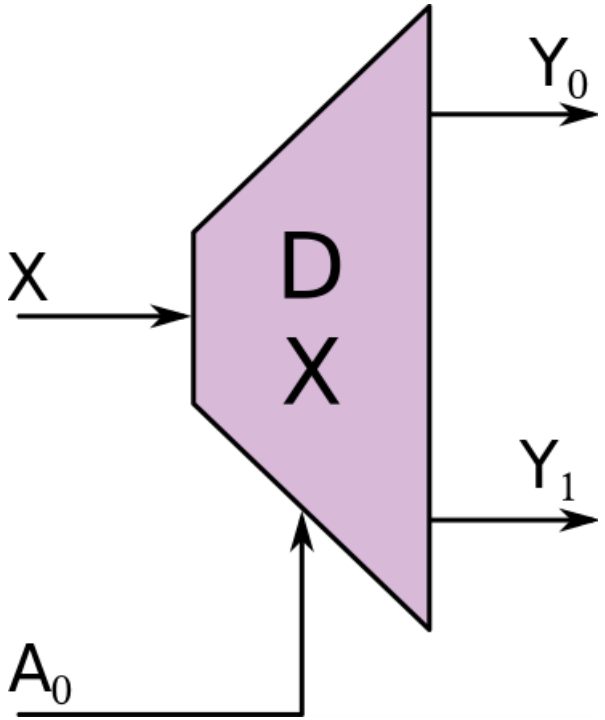
$$Y = A_0.X_1 + \overline{A_0}.X_0$$





Demultiplexor

- Opačná funkcia multiplexora
- Prenáša vstupné dáta na jeden vybraný výstup
- Jeden dátový vstupný kanál (X)
- Adresné (selektovacie) vstupy (A_0)
- Dátové výstupné kanály (Y_0,Y_1)



Demultiplexor

- Navrhnite **demultiplexor** so štyrmi výstupnými kanálmi
  - na výstupe neaktívneho kanála je LOG 0

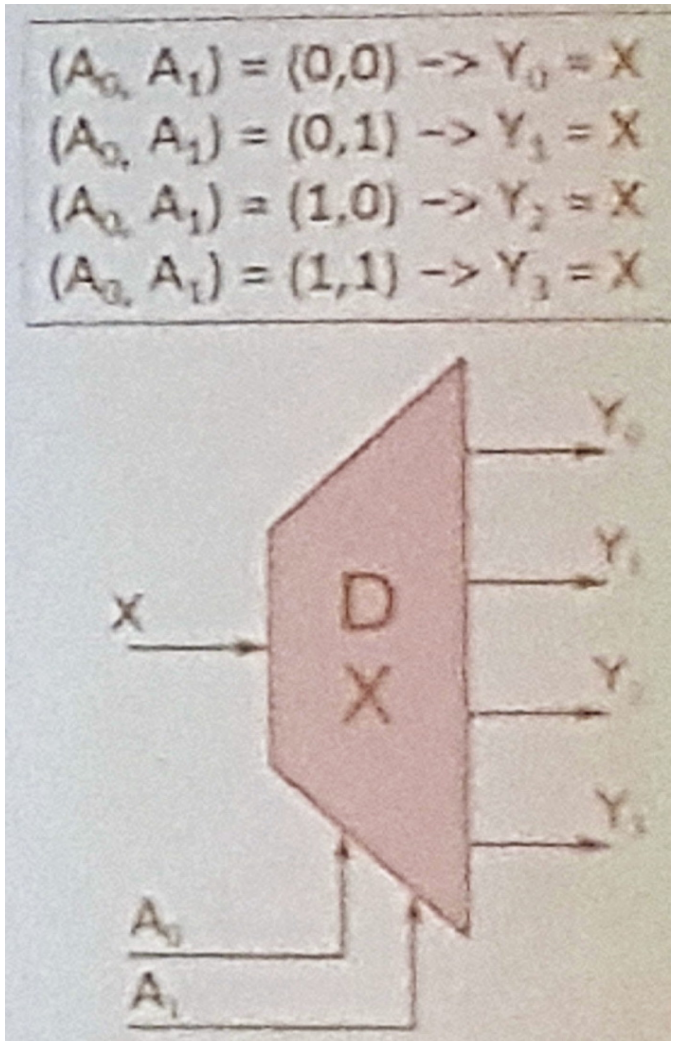
A_0	A_1	X	Y_0	Y_1	Y_2	Y_3
0	0	0	**0**	**0**	**0**	**0**
0	0	1	**1**	**0**	**0**	**0**
0	1	0	**0**	**0**	**0**	**0**
0	1	1	**0**	**1**	**0**	**0**
1	0	0	**0**	**0**	**0**	**0**
1	0	1	**0**	**0**	**1**	**0**
1	1	0	**0**	**0**	**0**	**0**
1	1	1	**0**	**0**	**0**	**1**

$$Y_0 = \overline{A_0}\,\overline{A_1}X$$

$$Y_1 = \overline{A_0}A_1X$$

$$Y_2 = A_0\overline{A_1}X$$

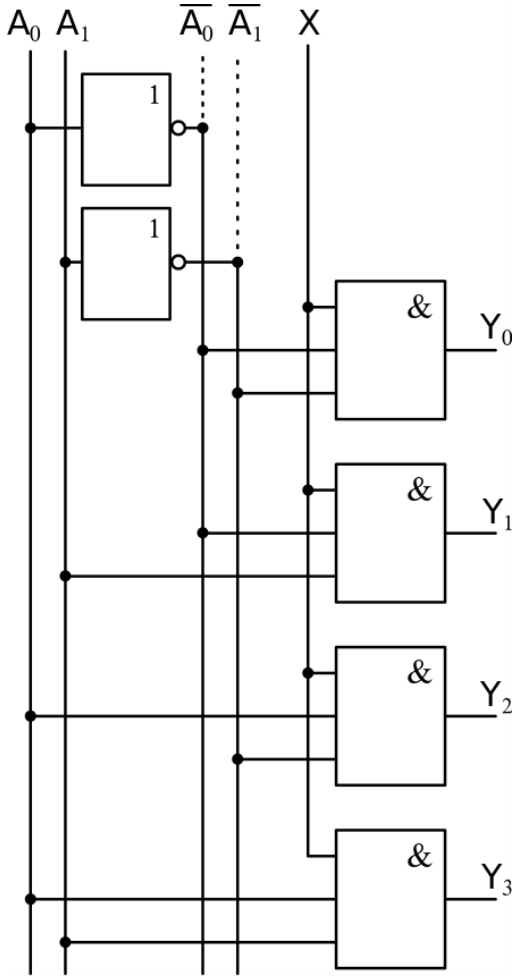
$$Y_3 = A_0A_1X$$



Demultiplexor

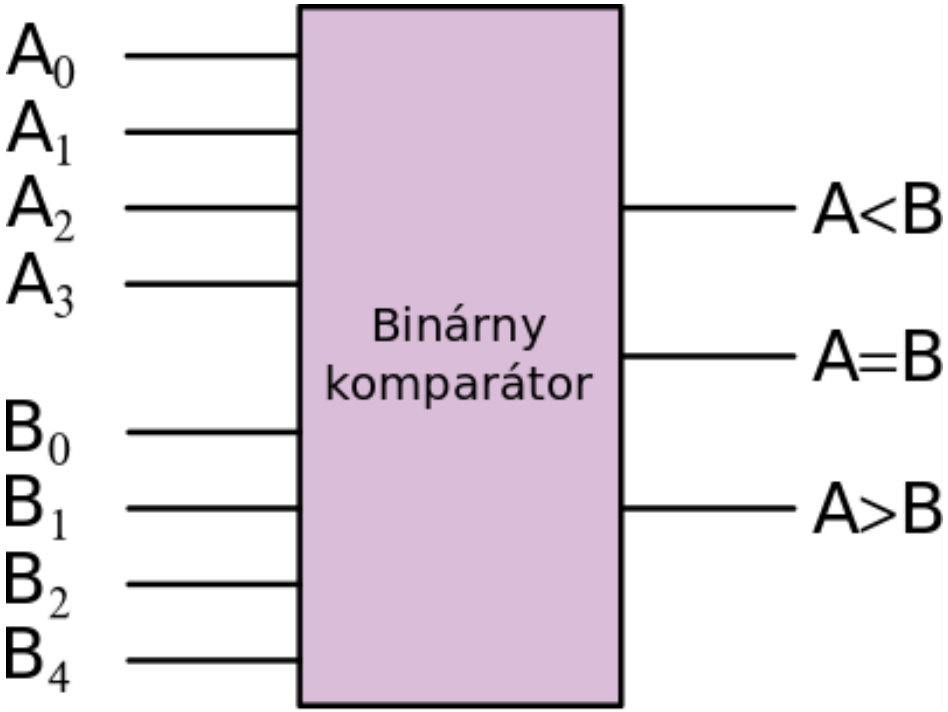
- Navrhnite **demultiplexor** so štyrmi výstupnými kanálmi
  - na výstupe neaktívneho kanála je LOG 0

$$Y_0 = \overline{A_0}\,\overline{A_1}X$$
$$Y_1 = \overline{A_0}A_1X$$
$$Y_2 = A_0\overline{A_1}X$$
$$Y_3 = A_0A_1X$$



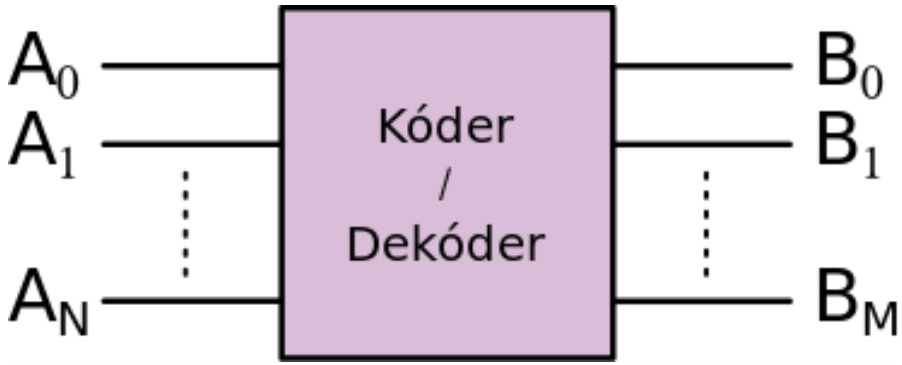
Binárny komparátor

- Paralelný obvod na porovnanie dvoch slov
- Porovnáva jednotlivé bity vstupných čísel
- Výstupom je logická hodnota na príslušnom výstupe obvodu



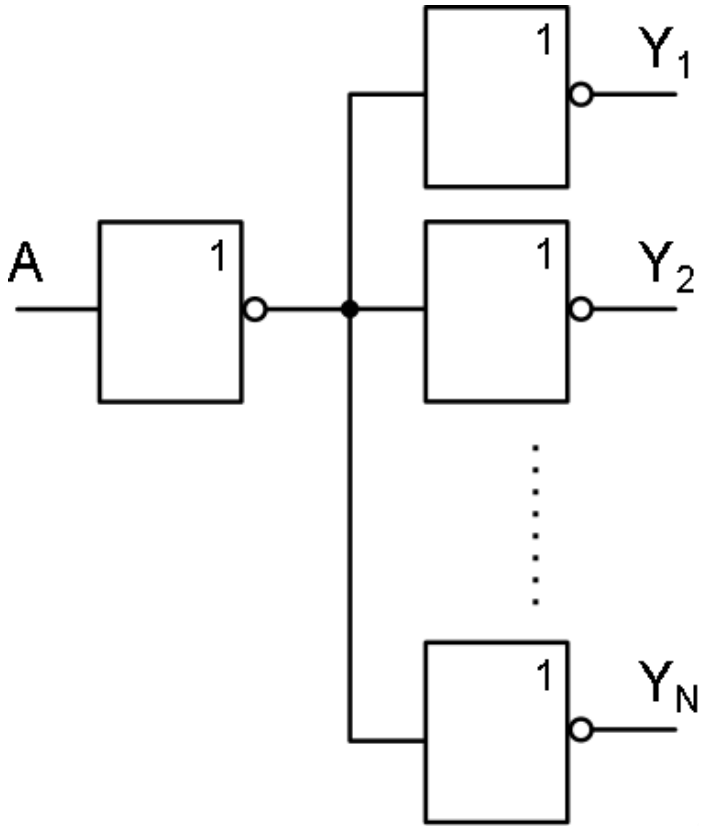
Kóder / Dekóder

- **Obvod na prevod rôznych kódov**
  - Priamy, Inverzný, Doplnkový binárny kód
  - BCD kód – Binary Coded Decimal
  - Grayov kód
  - Johnsonov kód
  - Adresné dekódery v RAM



Vlastnosti KO – Logický zisk

- **Logický zisk (Fanout)**
  - Parameter určujúci **zaťažiteľnosť** logického hradla (počet členov na výstupe)
  - Statický a dynamický logický zisk
  - Max. počet pripojených hradiel na výstupe hradla pri dodržaní dynamických parametrov
  - Parazitná kapacita a odpor vstupu hradiel
  - zvyčajne 10



Vlastnosti KO – Hazardy

- **Hazardy v logických obvodoch**
  - Vznikajú kvôli nedokonalostiam a parazitným vplyvom v reálnych obvodoch (časovanie, oneskorenie...)
  - Dochádza k nežiadúcej zmene hodnoty výstupu na krátky moment
    - zvyčajne pri zmene oboch premenných na vstupe toho istého hradla

- **Statické** hazardy
- **Dynamické** hazardy
- **Funkčné** hazardy

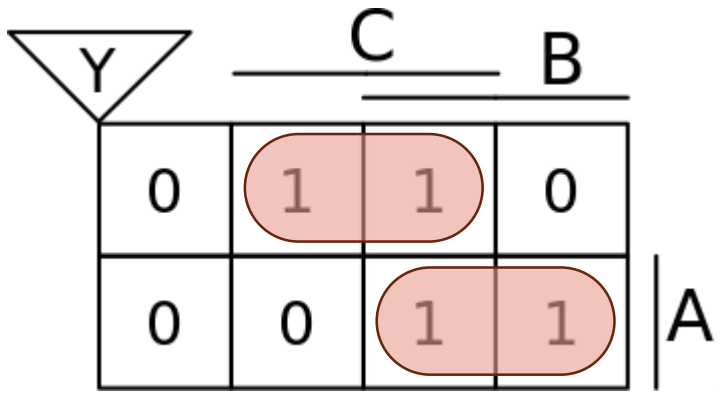
- **Statické hazardy**
  - Hazard statickej 1
    - Keď signál nežiaduco zmení hodnotu na LOG 0 (má byť 1)
  - Hazard statickej 0
    - Keď signál nežiaduco zmení hodnotu na LOG 1 (má byť 0)

- Odstrániteľné pomocou pridanej logiky
  - Redudantné slučky v Karnaughovej mape
  - Kompromis medzi zložitosťou/optimalizáciou obvodu a počtom hazardov
  - tzv. Huffmanova metóda

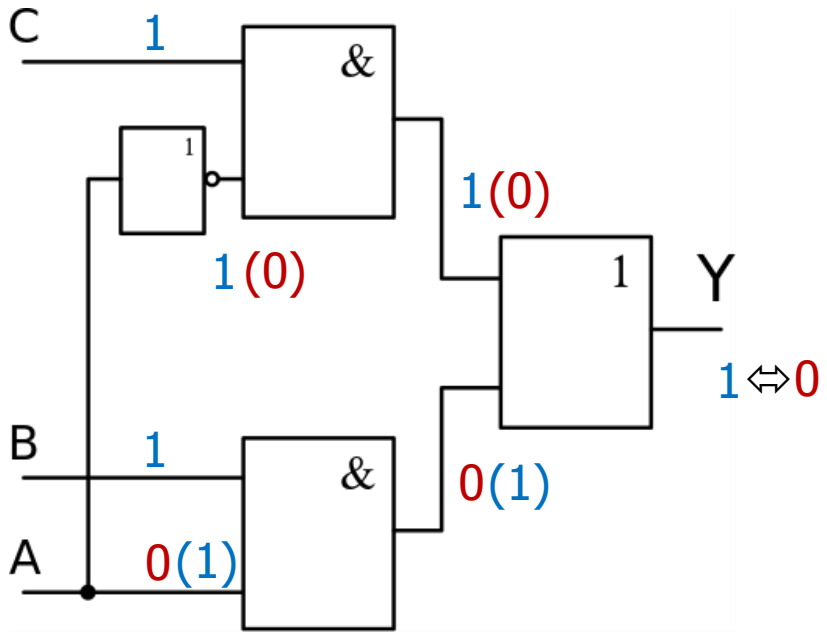
2x

Hazardy

- **Príklad statického hazardu -** statická 1



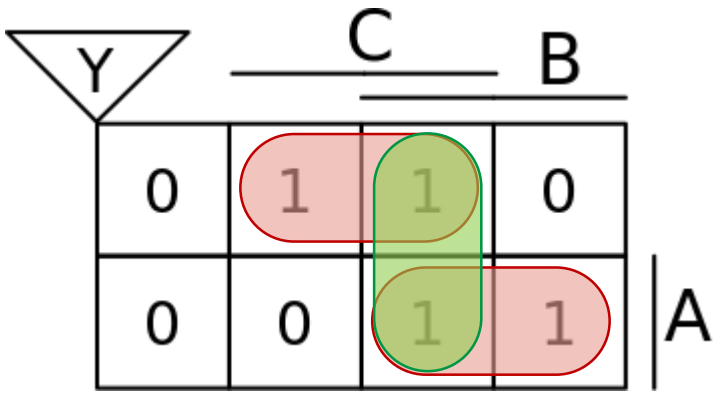
$$Y = AB + \overline{A}C$$



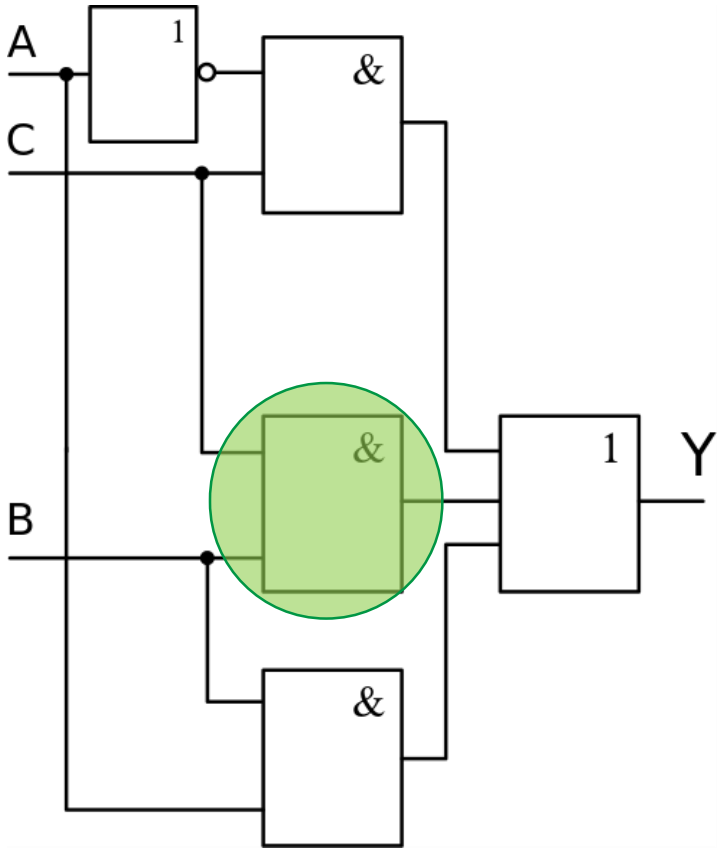
Pri prechode vstupov z 011 do 111, výstup Y neosláva v LOG 1
→ vzniká **statický hazard!**

Hazardy

- **Príklad statického hazardu -** statická 1



$$Y = AB + \overline{A}C + BC$$



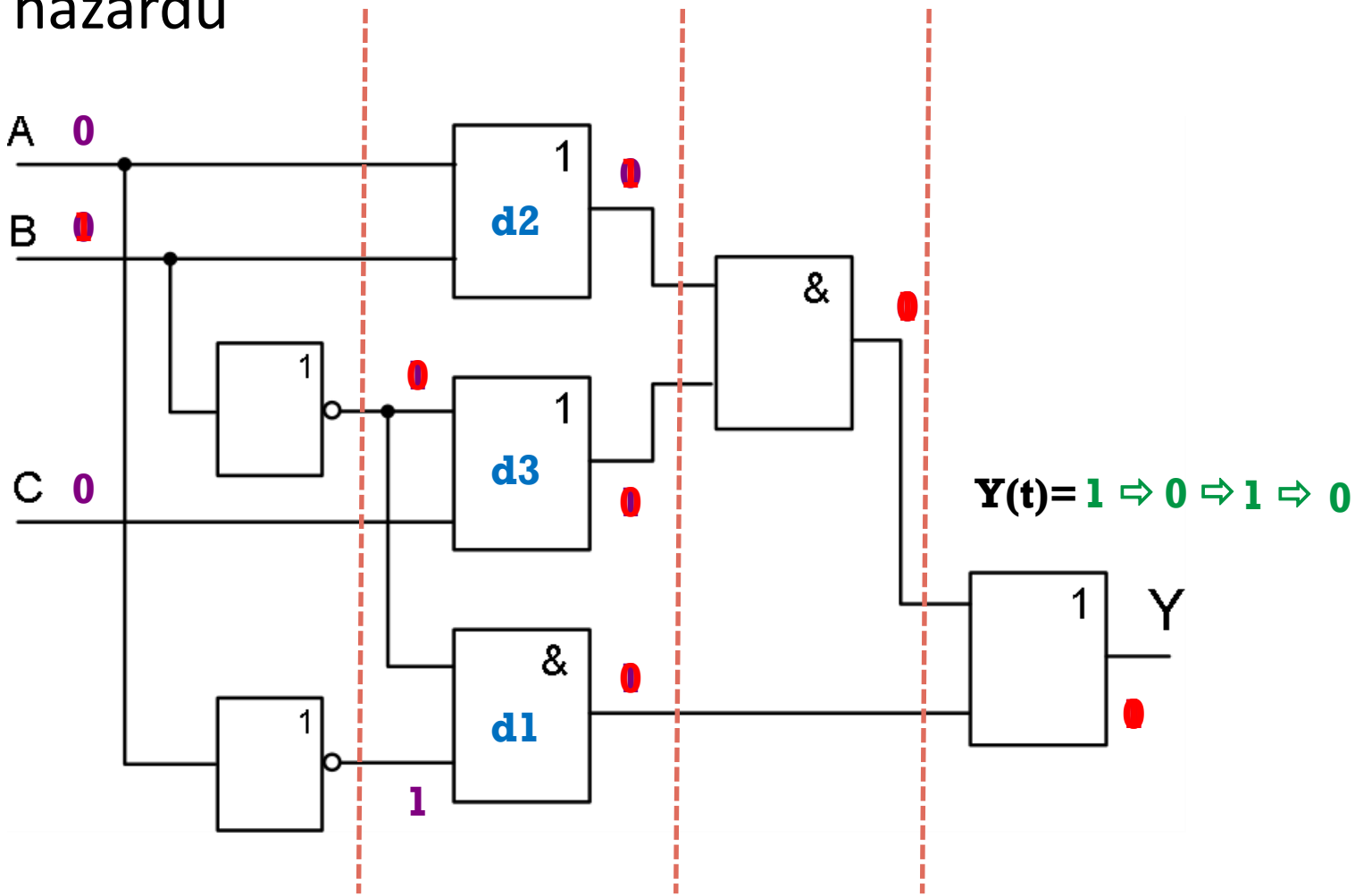
- Redudantná slučka/hradlo
- Výstup je podržaný v jednotke pri prechode na vstupoch z 011 do 111

Hazardy

- **Dynamické hazardy**
  - Viacnásobná zmena výstupu pred ustálením, pri jednej zmene vstupu
  - Nastávajú ak existuje viac ciest zo vstupu na výstup (vetvenie obvodu) a signálové vetvy majú rôzne oneskorenie (často ☹)
  - Na odstránenie dynamických hazardov treba odstrániť **všetky** statické hazardy, čo nie je vždy možné / ekonomické

Hazardy

- **Príklad** dynamického hazardu



* Oneskorenie hradiel: d1 << d2 << d3

Ďakujem za pozornosť.