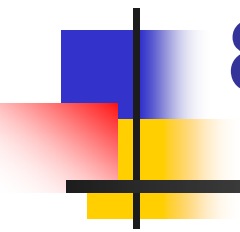


8. Programirljivi moduli

Sadržaj predavanja

- **koncept programirljivih modula**
- permanentna memorija
- programirljivo logičko polje
- poluprogramirljivo logičko polje
- složeni programirljivi moduli
- programirljivo polje logičkih blokova

Koncept programirljivih modula

- *programirljivi* moduli
  ~ "programirljive naprave", PLD
  (engl. Programmable Logic Devices):
  - ostvarivanje složenije funkcije koja *nije* unaprijed određena
    ~ moduli opće namjene
  - mogućnost *naknadnog* "programiranja"
    ~ konfiguriranje sklopa u smislu određivanja "izvana opazivog ponašanja":
    - u posebnim uređajima
    - unutar uređaja u kojem se modul koristi

Koncept programirljivih modula

- struktura složenija (puno "logike"), ali *nije* fiksirana:
  - logički sklopovi ("vrata", engl. gates)
    ili skupovi logičkih sklopova (~ "logički blokovi")
  - tvornički izvedeni kontakti ili *programirljive sklopke*:
    - različita povezivanja logičkih sklopova
    - konfiguriranje skupova logičkih sklopova
      unutar modula
  - osnovna struktura
    ~ dvodimenzijsko polje dekoder-koder:
    permanentna memorija

Koncept programirljivih modula

- podjela programirljivih modula:
  - jednostavni PLD (engl. Simple PLD, SPLD):
    - programirljivo logičko polje, PLA
    - poluprogramirljivo logičko polje, PAL
  - složeni PLD (engl. Complex PLD, CPLD)
    ~ *više programirljivo povezanih* SPLD u modulu
  - programirljiva polja logičkih blokova
    (engl. Field Programmable Gate Arrays, FPGA)
    ~ *veliki broj* programirljivo povezanih *programirljivih* logičkih blokova

Sadržaj predavanja

- koncept programirljivih modula
- **permanentna memorija**
  - **funkcionalnost i struktura**
  - **tehnologija izvedbe**
  - **karakteristične primjene**
  - **generiranje Booleovih funkcija**
- programirljivo logičko polje
- poluprogramirljivo logičko polje
- složeni programirljivi moduli
- programirljivo polje logičkih blokova

Permanentna memorija

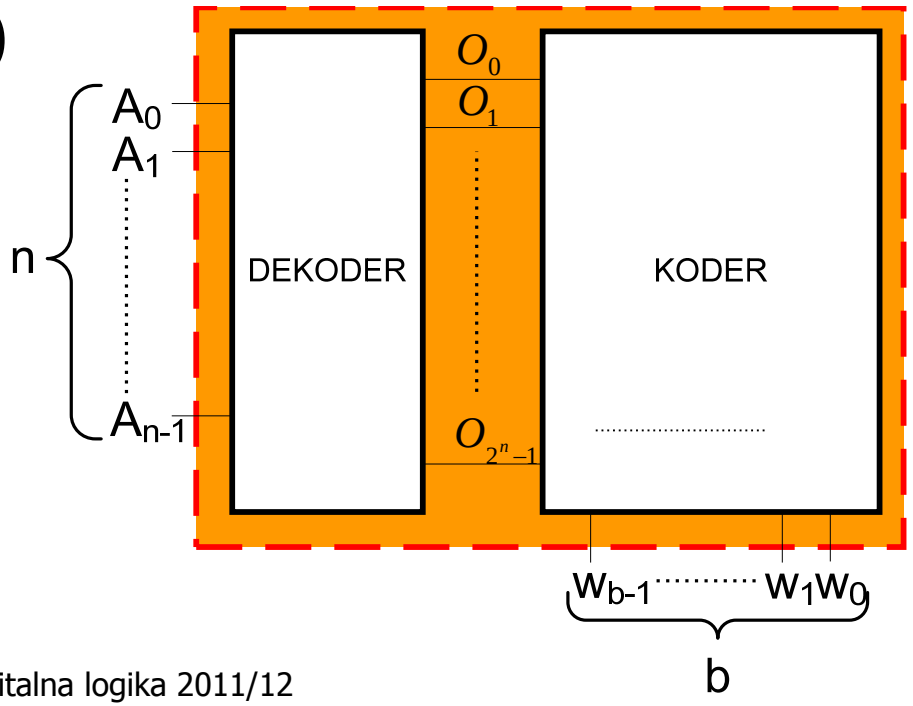
- funkcijski pogled
  ~ sklop s *permanentno* upisanim sadržajem:
  memorija
  - jedan upis (obično pri proizvodnji), ostalo čitanje
    ~ ispisna memorija:
    "samo-se-čita", ROM (engl. Read Only Memory)
  - može i više upisa, ali *zanemarivo malo*
    u odnosu na broj čitanja
- izvedba
  ~ *kombinacijski* sklop
  - podatak upisan nekom vrstom "ožičenja"
  - mogućnost "programiranja"

Permanentna memorija

- karakteristična struktura
  ~ *dva* polja:
  - ulazno ili *dekodersko* polje:
    - generiranje potrebnog broja internih adresnih linija
    - potpuno adresiranje: "1-od-2^n"
    - dekoder
      ~ I sklopovi na izlazima $\rightarrow$ *I polje*
  - izlazno ili *kodersko* polje:
    - generiranje bitova adresirane "riječi"
      ~ aktiviranje željenih izlaza:
      "kodiranje" pojedinih simbola
    - koder
      ~ ILI sklopovi na izlazima $\rightarrow$ *ILI polje*
      (izlaz = podatak1 ILI podatak2 ILI …)

Permanentna memorija

- karakteristična struktura:
  - *dva* polja
  - broj "memorijskih riječi" = 2^n
  - broj bitova/riječ = b
  - kapacitet: W = 2^n x b
- programiranje (*kodera*!)
  ~ upis uzorka 1 i 0
  ∀ memorijsku riječ

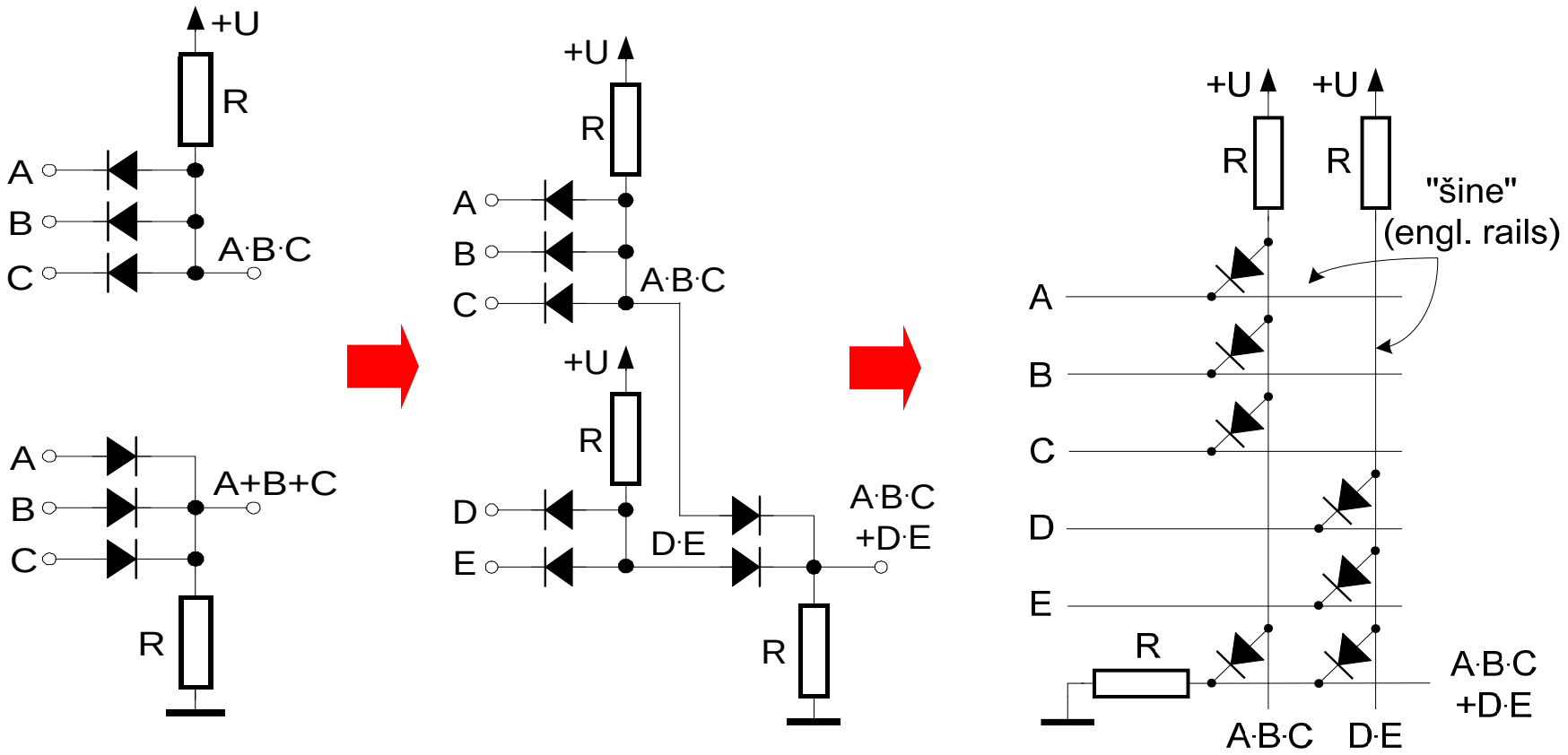


Izvedbe permanentne memorije

- karakteristična struktura s *dva* polja
  ~ izvorno *diodna matrica*
  - osnovni logički sklopovi ostvareni *diodnim mrežama*
  - struktura tipa funkcije drugog reda (suma minterma)
    ~ oblik ILI-I
  - električka funkcija dioda
    ~ onemogućiti *povratno* djelovanje
    s drugih "šina" (engl. rails)
  - suvremene strukture
    ~ *poopćenja* diodne matrice

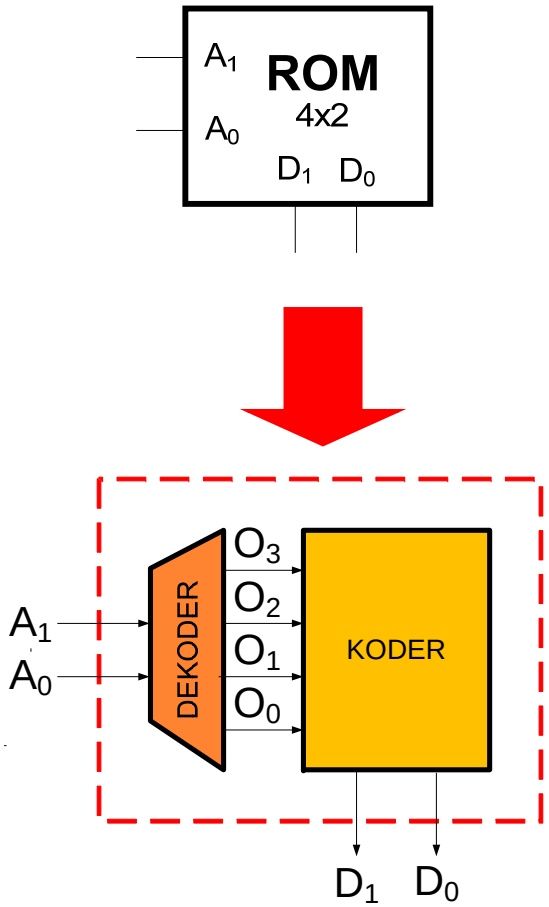
Izvedbe permanentne memorije

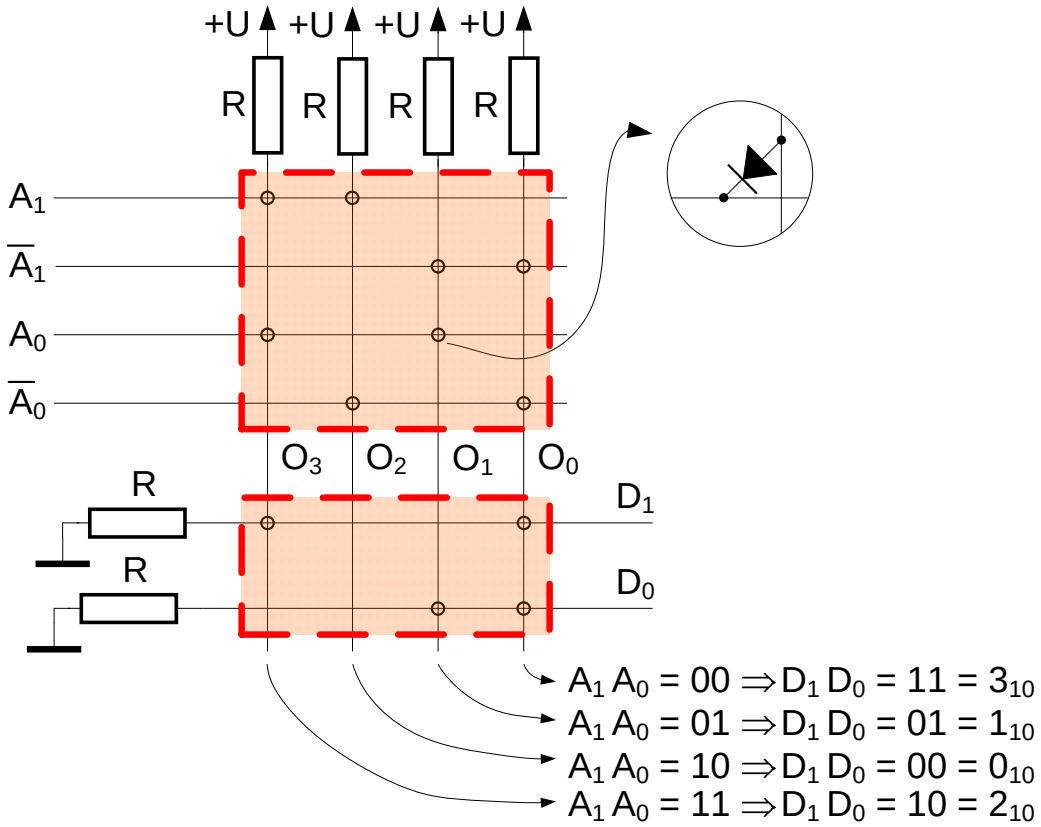
- izvedba diodne matrice iz diodnih sklopova I i ILI



Izvedbe permanentne memorije

Primjer: Izverdba ROMa s 4 2-bitne riječi (ROM 4x2) diodnom matricom





Izvedbe permanentne memorije

Zadatak: temeljeći sa na izvedbi permanentne memorije diodnom mrežom nacrtati:

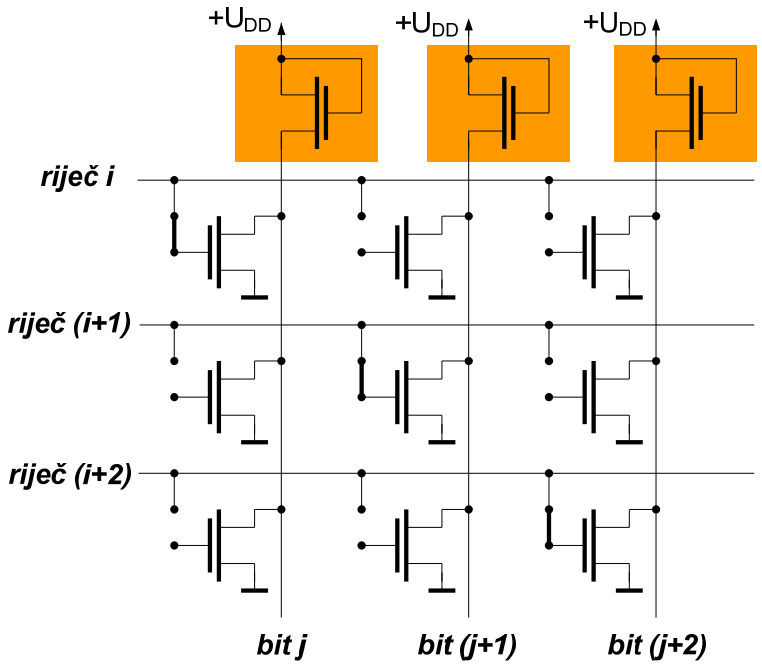
- sklop 1-bitnog potpunog zbrajala
- sklop 1-bitnog potpunog odbijala
- sklop 1-bitnog potpunog zbrajala/odbijala
  (uputa: predvidjeti upravljačku varijablu K za odabir zbrajanja (K = 0), odnosno oduzimanja (K = 1))
- na raspolaganju su varijable i komplementi

Izvedbe permanentne memorije

- izvedbe ~ tehnologija:
  - bez mogućnosti programiranja, ROM
  - s mogućnošću jednokratnog programiranja, PROM (engl. Programmable ROM)
  - s mogućnošću *višekratnog* programiranja i brisanja UV svjetlom, EPROM (engl. Erasable PROM)
    ~ kućište sa staklenim prozorčićem
  - s mogućnošću višekratnog programiranja i brisanja *električkim* putem, EAROM (engl. Electrically Alterable ROM), EEPROM (engl. Electrically EPROM)

Izvedbe permanentne memorije

- "klasična" permanentna memorija, ROM:
  - uobičajena tehnologija
    ~ MOSFET
  - programiranje u proizvodnji:
    ~ zadnja se maska
    izrađuje po narudžbi
    i sadrži potrebne veze
  - t_a ~ 100 ns



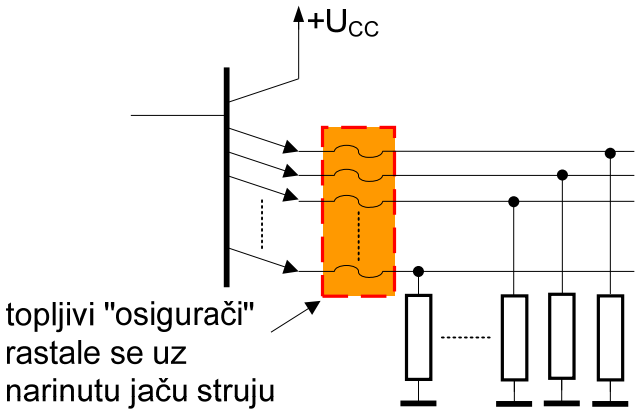
Izvedbe permanentne memorije

Zadatak: temeljeći sa na izvedbi permanentne memorije MOSFET tranzistorima nacrtati:

- sklop 1-bitnog potpunog zbrajala
- sklop 1-bitnog potpunog oduzimala
- sklop 1-bitnog potpunog zbrajala/oduzimala
  (uputa: predvidjeti upravljačku varijablu K za odabir zbrajanja (K = 0), odnosno oduzimanja (K = 1))

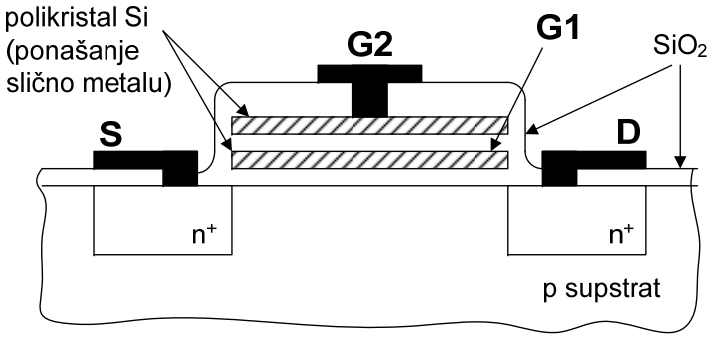
Izvedbe permanentne memorije

- s mogućnošću *jednokratnog* programiranja, PROM:
  - bipolarna tehnologija, tipično TTL
    ~ višeemiterski tranzistor
  - za male serije
    ~ programiranje "na licu mjesta" (engl. in-the-field)
  - programiranje
    ~ jačom strujom (U_B >>)
  - t_a ~ 30÷50 ns



Izvedbe permanentne memorije

- s mogućnošću *višekratnog* programiranja i brisanja *UV svjetlom*, EPROM:
  - tehnologija MOSFET
    ~ posebna izvedba NMOS tranzistora "s lebdećom elektrodom", FAMOS (engl. Floating-gate Avalanche Injection MOS)
  - programiranje
    ~ U_{G2D} ~ 25 V
    prodor elektrona u G_1 *lavinskim probojem*
  - t_a ~ 200 ns

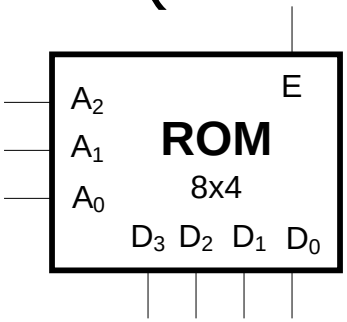


Izvedbe permanentne memorije

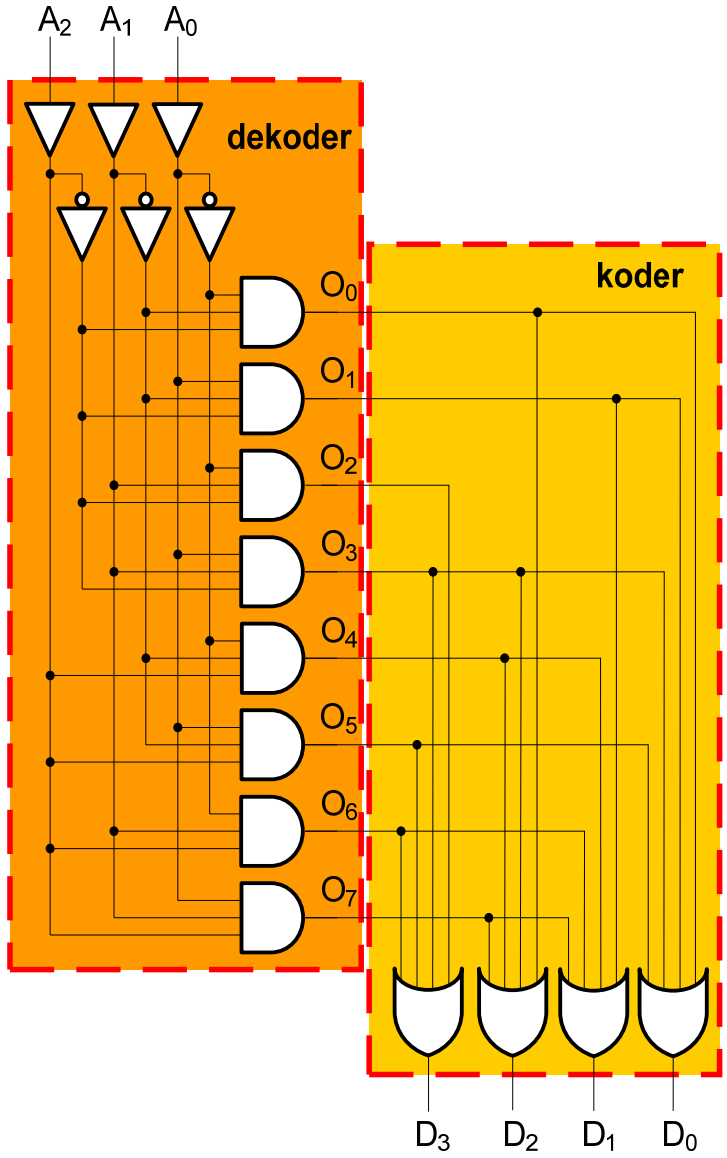
- s mogućnošću *višekratnog* programiranja i brisanja *električkim putem*, EAROM, EEPROM:
  - izbjeći probleme EPROM
    ~ dugo brisanje cijelog sadržaja u posebnom uređaju
  - smanjen razmak G_1 i D
    ~ upisivanje i brisanje podatka *tuneliranjem*
    (upis: U_{G2D} ~ 10 V, brisanje: U_{G2D} ~ -10 V)
  - t_a ~ 250 ns

Izvedbe permanentne memorije

Primjer: ROM s 8 4-bitnih riječi (ROM 8x4)

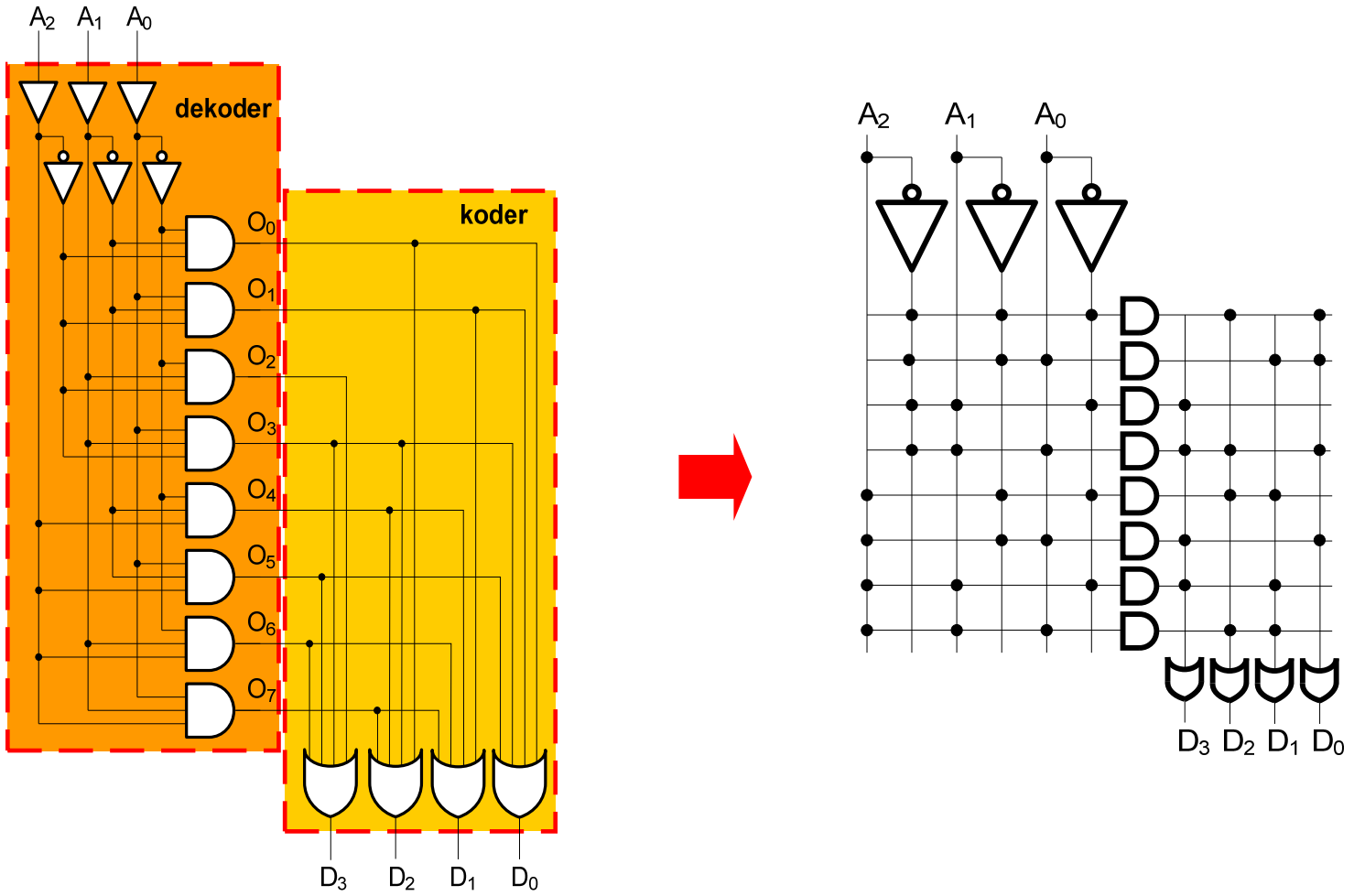


riječ	A_2	A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	1	0	1
1	0	0	1	0	0	1	1
2	0	1	0	1	0	0	0
3	0	1	1	1	1	0	1
4	1	0	0	0	1	1	0
5	1	0	1	1	0	0	1
6	1	1	0	1	0	1	0
7	1	1	1	0	1	1	0



Izvedbe permanentne memorije

- polje = matrica
  ~ *matrični prikaz*



Karakteristične primjene

- primjena ROM:
  - pohranjivanje značajnih podataka važnih za rad cjelokupnog digitalnog sustava (npr. računalo)
  - pohranjivanje sustavskih programa
  - upravljačka memorija kod *mikroprogramiranja*
    ~ posebna izvedba upravljačke jedinice procesora
  - pretvorba koda
    ~ naročito generatori znakova za rasterske prikaze (zasloni, matrični pisači)
  - aritmetički sklopovi:
    ~ izvedbe tablica posebnih funkcija (npr. trigonometrijske)
- *problem*
  ~ ROM je *sporiji*, jer ima više razina logike

Karakteristične primjene

Zadatak: permantnom memorijom ostvariti slijedeće pretvornike koda:

- BCD u 2421
- BCD u 7-segmentni
- koji su parametri (broj riječi x broj bitova/riječ) potrebnih permanentnih memorija?

Karakteristične primjene

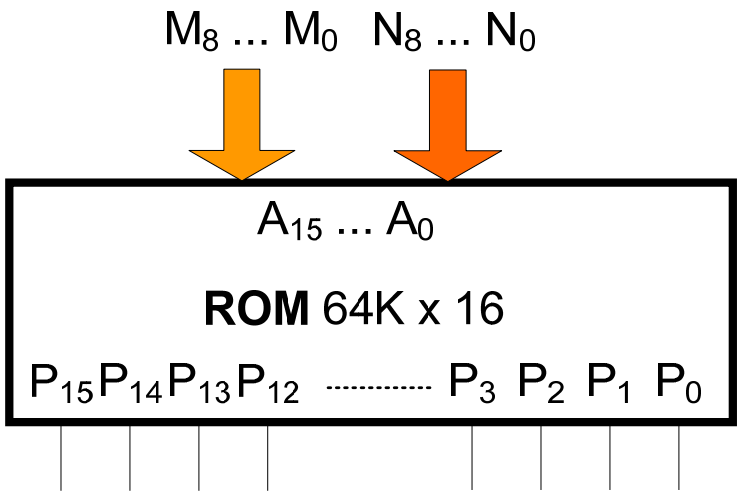
Primjer: sklop za množenje 8-bitnih brojeva

- tablica množenja ugrađena u ROM
  ~ *pregledna tablica* (engl. Look-Up Table, LUT)
- efikasnija izvedba:
  - *kombinacija* izvjesnog broja ROMova značajno manjeg kapaciteta (parcijalni produkti) i zbrajala
  - veća kašnjenja!

Karakteristične primjene

- množenje 8-bitnih brojeva
  ~ potreban kapacitet *prevelik*:

$$C = (8+8)\cdot 2^{8+8} = 2^4 \cdot 2^{16} = 2^{20} = 1Mbit$$



Karakteristične primjene

- "rastavljanje" multiplikanda i multiplikatora:

$$M = (m_7 m_6 m_5 m_4) \cdot 2^4 + (m_3 m_2 m_1 m_0) = m \cdot 2^4 + \Delta m$$

$$N = (n_7 n_6 n_5 n_4) \cdot 2^4 + (n_3 n_2 n_1 n_0) = n \cdot 2^4 + \Delta n$$

$$P = M \cdot N$$

$$= \left(m \cdot 2^4 + \Delta m\right) \cdot \left(n \cdot 2^4 + \Delta n\right) =$$

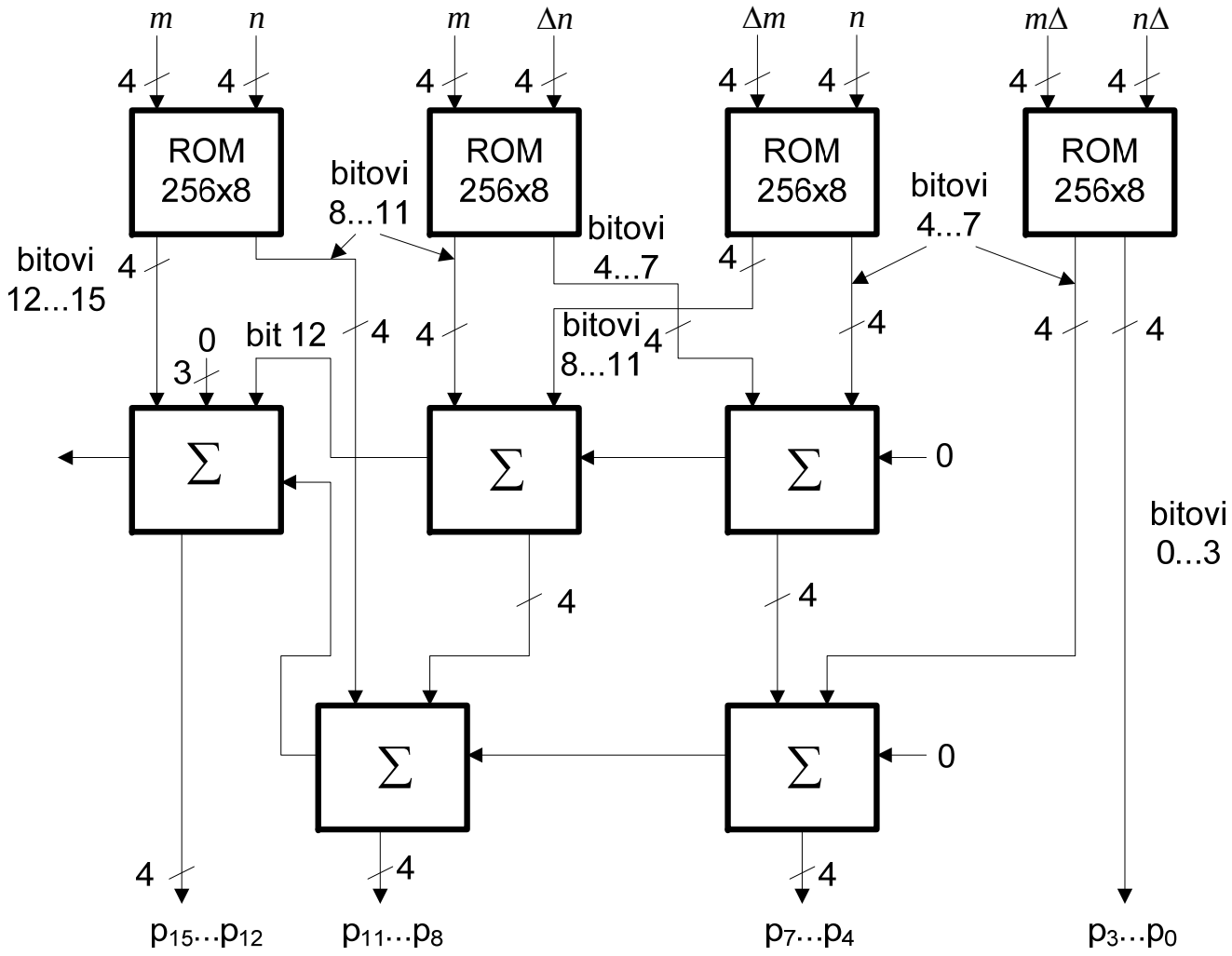
$$= \left(m \cdot n\right) \cdot 2^8 + \left(\Delta m \cdot n + m \cdot \Delta n\right) \cdot 2^4 + \Delta m \cdot \Delta n$$

- dovoljan puno manji kapacitet:

$$C' = \left(4+4\right) \cdot 2^{4+4} = 2Kbit; C_{ukupni} = 4 \cdot C' = 8Kbit$$

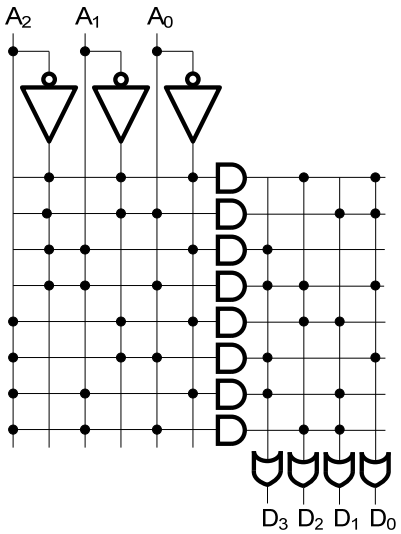
Karakteristične primjene

- sklop zasnovan na kompoziciji manjih ROMova:



Generiranje Booleovih funkcija

- interpretacija podataka "pohranjenih" u ROM: *logičke funkcije* (više njih!)
  - svaki izlaz ~ jedna logička funkcija
  - sve funkcije ~ višeizlazna funkcija
  - ROM ~ generator funkcija



$$f_i(A_2, A_1, A_0) = D_i \quad 0 \le i \le 3$$

$$f_0 = \sum m(0,1,3,5)$$

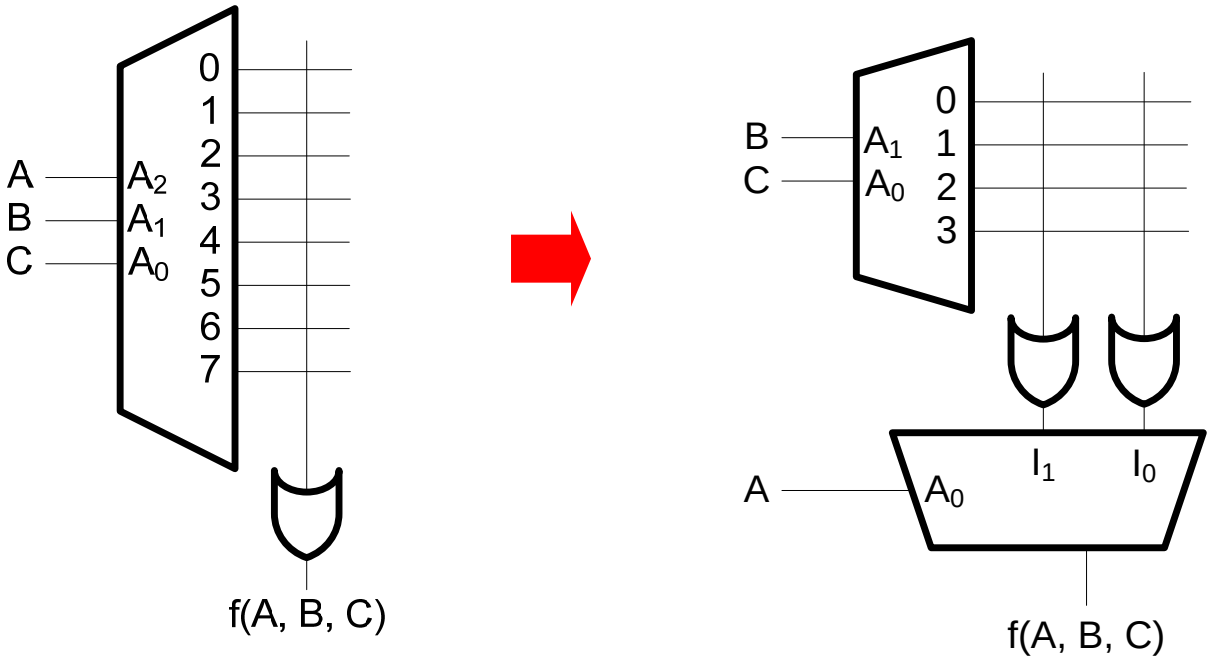
$$f_1 = \sum m(1,4,6,7)$$

$$f_2 = \sum m(0,3,4,7)$$

$$f_3 = \sum m(2,3,5,6)$$

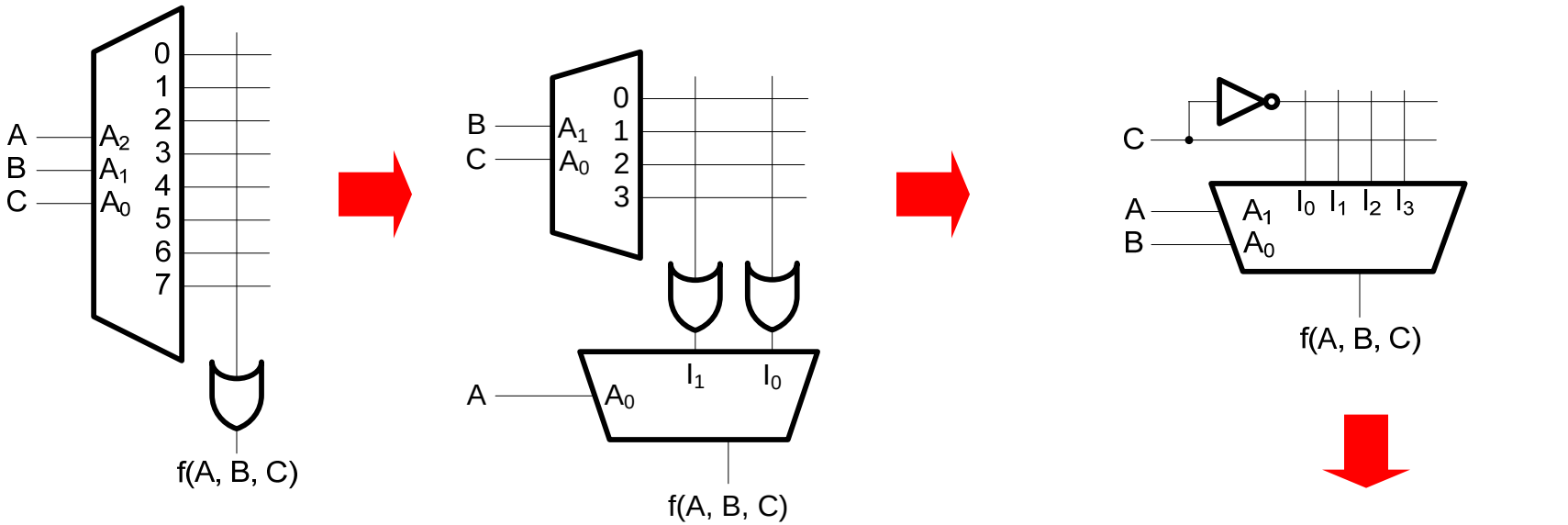
Generiranje Booleovih funkcija

- zapažanje
  ~ sklopovski povoljnije koristiti *raspodijeljeni dekoder* (engl. split decoder)



Generiranje Booleovih funkcija

- razvoj ostvarivanja funkcija raspodijeljenim dekodiranjem ROMa
  ~ ostvarivanje funkcija multipleksorom!



- simulacija permanentne memorije:

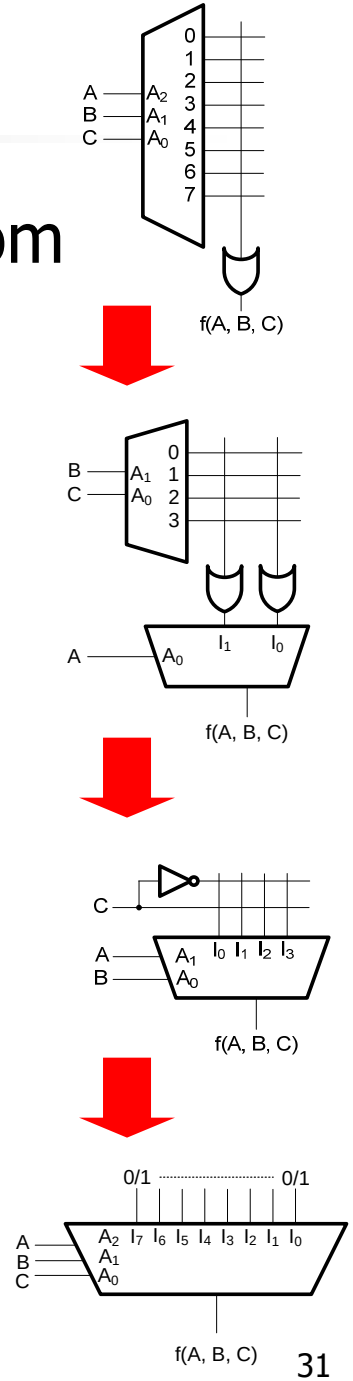
Generiranje Booleovih funkcija

- ostvarivanja funkcija ROMom i multipleksorom ~ *Shannonova ekspanzija*:

$$f(A,B,C) = f(0,B,C)\cdot\overline{A} + f(1,B,C)\cdot A$$

$$f(A,B,C) = f(0,0,C)\cdot\overline{A}\cdot\overline{B} + f(0,1,C)\cdot\overline{A}\cdot B + f(1,0,C)\cdot A\cdot\overline{B} + f(1,1,C)\cdot A\cdot B$$

$$f(A,B,C) = f(0,0,0)\cdot\overline{A}\cdot\overline{B}\cdot\overline{C} + f(0,0,1)\cdot\overline{A}\cdot\overline{B}\cdot C + f(0,1,0)\cdot\overline{A}\cdot B\cdot\overline{C} + f(0,1,1)\cdot\overline{A}\cdot B\cdot C + f(1,0,0)\cdot A\cdot\overline{B}\cdot\overline{C} + f(1,0,1)\cdot A\cdot\overline{B}\cdot C + f(1,1,0)\cdot A\cdot B\cdot\overline{C} + f(1,1,1)\cdot A\cdot B\cdot C$$



Generiranje Booleovih funkcija

- parcijalne funkcije kod Shannonova ekspanzije u kojima je neki od literala fiksiran (0 ili 1)
  ~ *kofaktori*, *rezidui* (ostaci), *rezidualne* funkcije
  npr. $\varphi_3(C), \varphi_2(C), \varphi_1(C), \varphi_0(C)$

$$f(A,B,C) = f(0,0,C)\cdot\overline{A}\cdot\overline{B} + f(0,1,C)\cdot\overline{A}\cdot B + f(1,0,C)\cdot A\cdot\overline{B} + f(1,1,C)\cdot A\cdot B$$

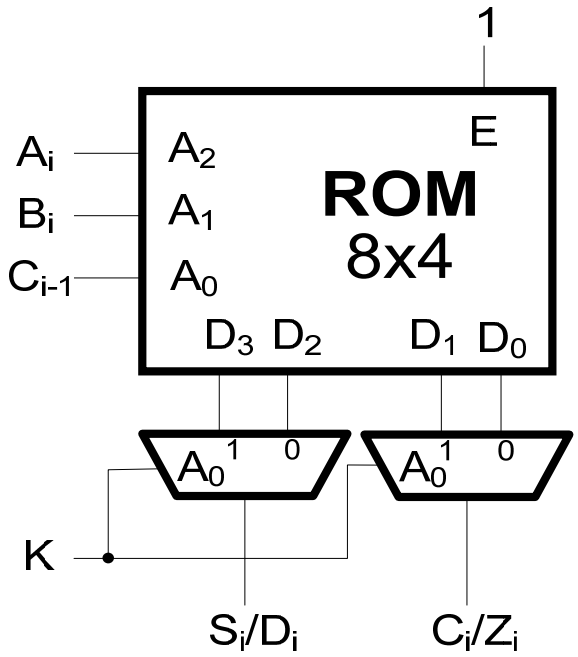
$$\varphi_0(C) = f(0,0,C)$$
$$\varphi_1(C) = f(0,1,C)$$
$$\varphi_2(C) = f(1,0,C)$$
$$\varphi_3(C) = f(1,1,C)$$

Generiranje Booleovih funkcija

Primjer : potpuno zbrajalo/odbijalo

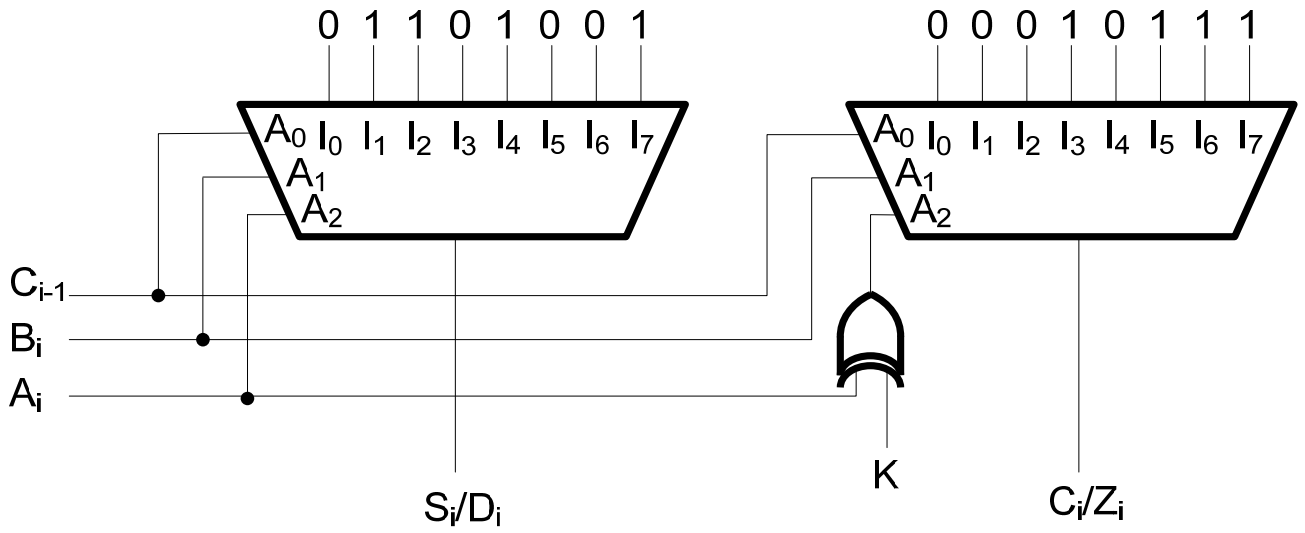
- izvedba s ROMom 8x4 i 2 MUXa 2/1
  - K=0: zbrajalo, K=1: odbijalo
  - uočiti: $S_i = D_i$, simetrija C_i i Z_i

A_2	A_1	A_0	D_2	D_0	D_3	D_1
A_i	B_i	C_{i-1}	S_i	C_i	D_i	Z_i
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	1	1
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	1	0	0	1	0	0
1	1	1	1	1	1	1



Generiranje Booleovih funkcija

- izvedba s 2 MUXa 8/1 u funkciji ROMova
  - K=0: zbrajalo, K=1: odbijalo



Sadržaj predavanja

- koncept programirljivih modula
- permanentna memorija
- **programirljivo logičko polje**
  - **funkcionalnost i struktura**
  - **tehnologija izvedbe**
  - **generiranje Booleovih funkcija**
- poluprogramirljivo logičko polje
- složeni programirljivi moduli
- programirljivo polje logičkih blokova

Programirljivo logičko polje

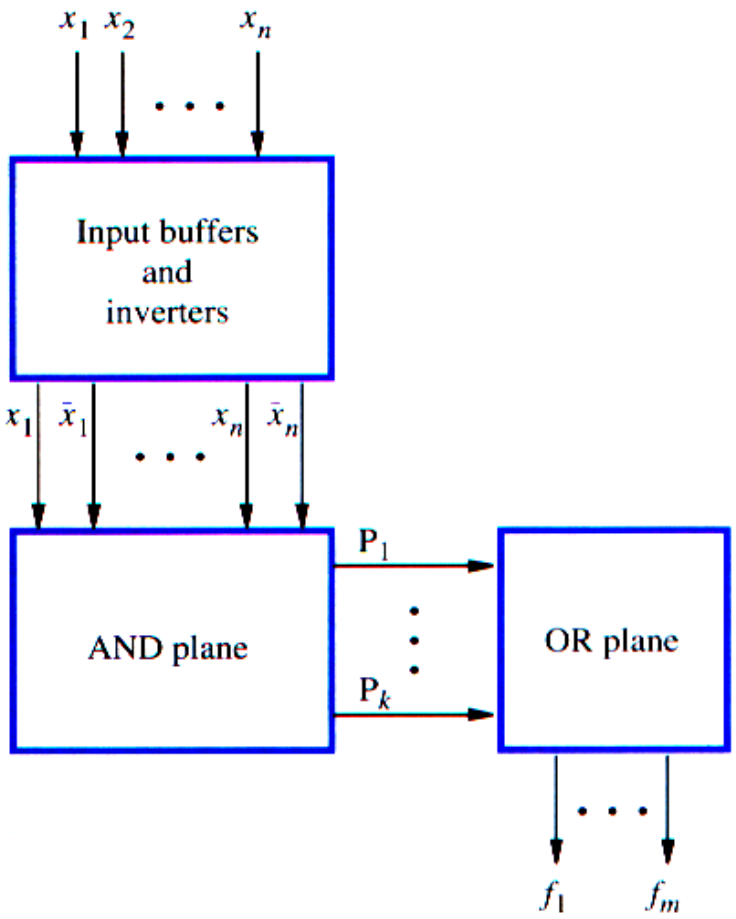
- zapažanje
  ~ pri generiranju funkcija je spojno polje kodera ROMa "slabo popunjeno"
  - obično se *ne* koriste *sve* kombinacije ulaza
    ~ *nepotpuno specificirane funkcije*
  - ostvariti uštedu
    ~ puno manji broj riječi od onog omogućenog dekoderom ($W << 2^n$)
  - relativno mali broj raspoloživih kombinacija ulaza
    ~ obuhvatiti upravo one potrebne!
    - programiranje i dekodera
      ~ *nepotpuno* dekodiranje

Programirljivo logičko polje

- *programirljivo logičko polje*, PLA
  (engl. Programmable Logic Array)
  ~ posebna logička struktura s *oba programirljiva* polja:
  - efikasno i fleksibilno rješenje
  - generirane funkcije
    ~ u obliku *sume produkata*
  - ograničenje ostvarivosti funkcija
    ~ veličina I polja (broj P_i):
    - smanjenje broja riječi
      ~ *minimizacija*!
  - višeizlazna funkcija:
    - posebni postupak minimizacije
    - što više P_i *zajedničkih* za što veći broj funkcija

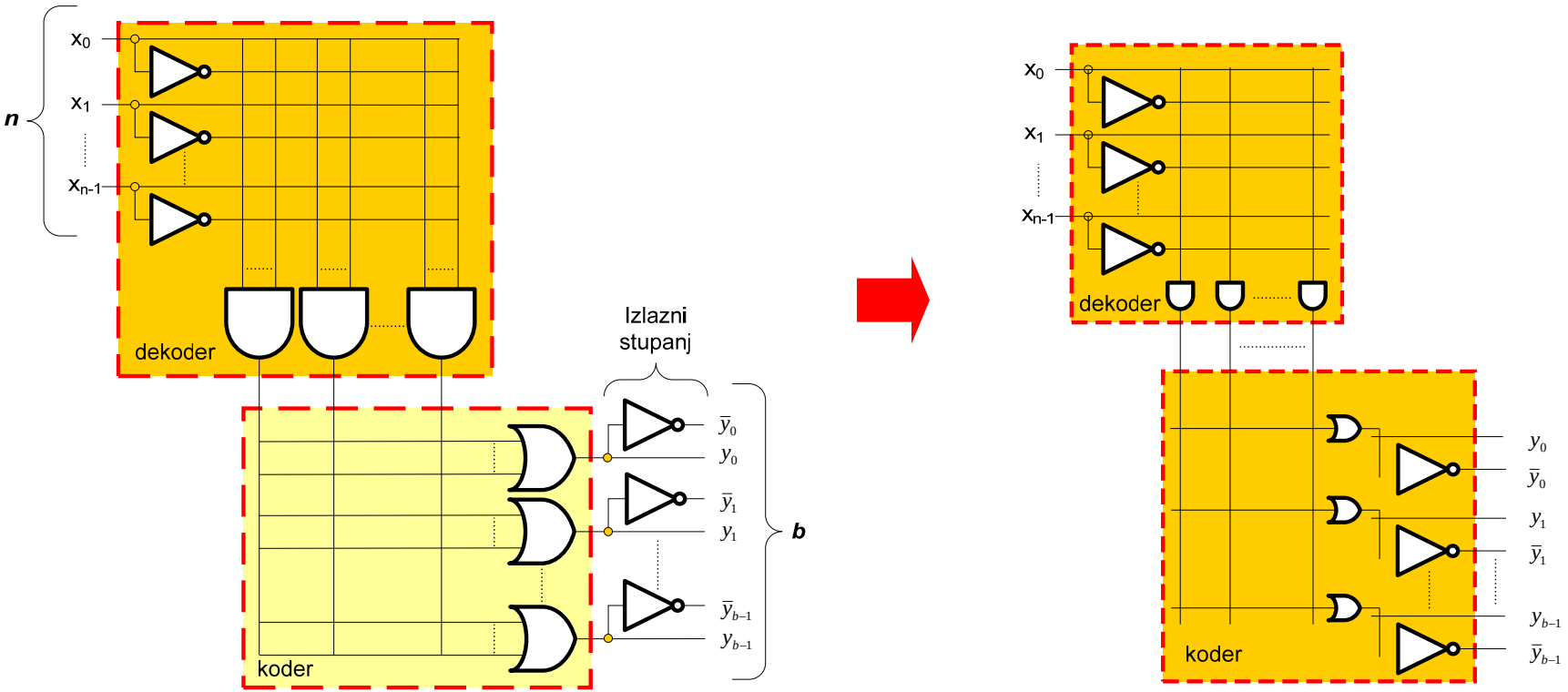
Programirljivo logičko polje

- izvedba PLA:
  - odvojni sklopovi na ulazu ~ generiranje $x_i, \overline{x_i}, \forall i$
  - tipični parametri: n x k x m = 16 x 32 x 8



Programirljivo logičko polje

- struktura PLA:
  - izlaz
    ~ funkcija i *komplement* (upravljani EX-ILI)

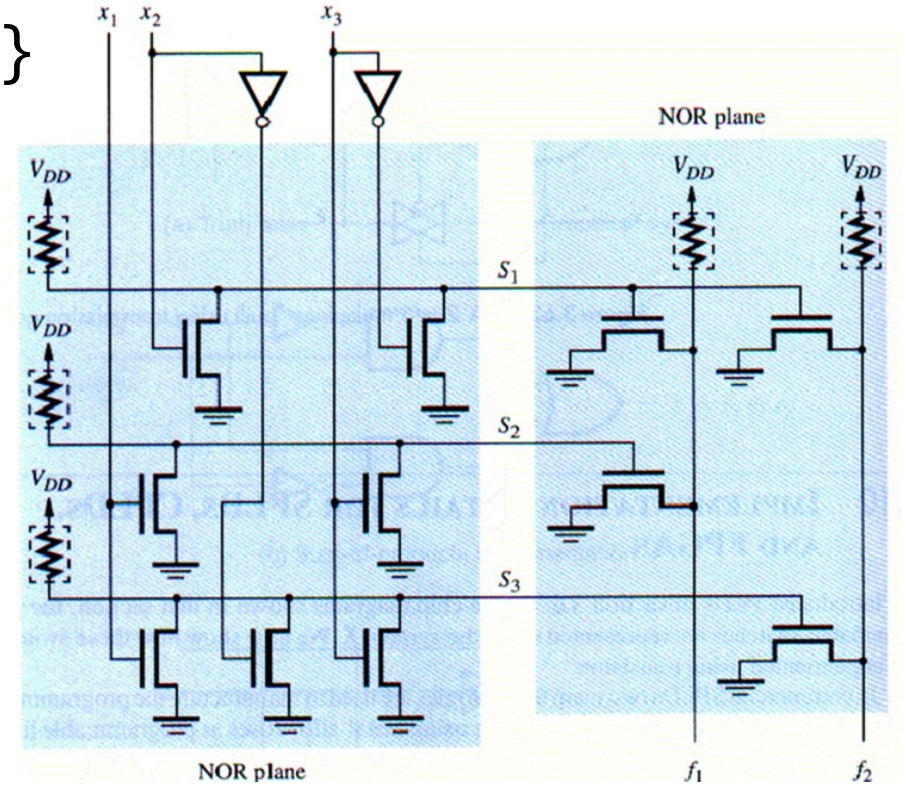


Izvedbe programirljivog logičkog polja

Primjer: izvedba PLA na razini tranzistora

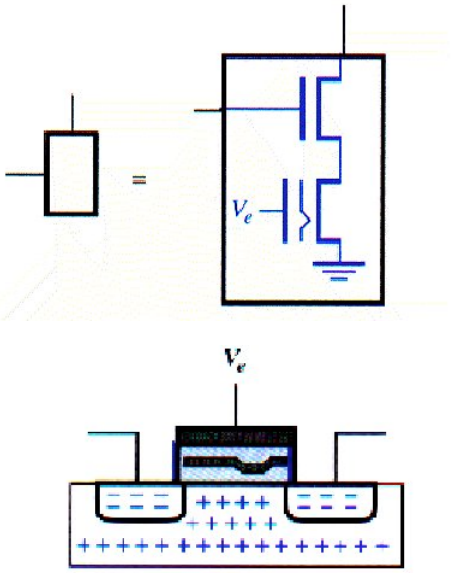
- programiranje u tvornici
- implementacija dvorazinske funkcije tipa NILI-NILI (NILI ~ paralelni NMOSovi)
- višeizlazna funkcija: {f_1, f_2}

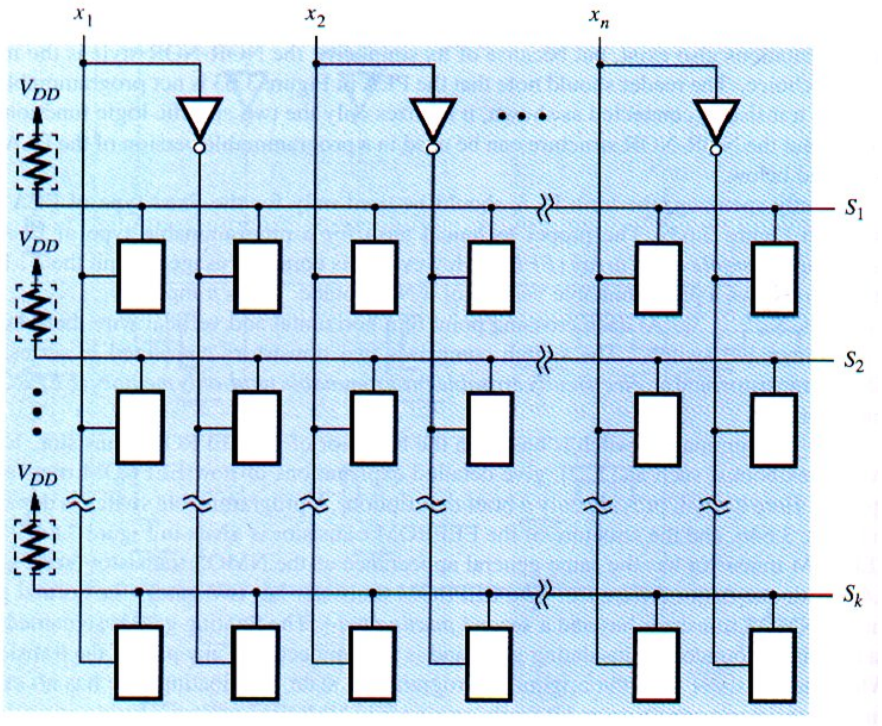
$$f_1 = \overline{S_1 + S_2}$$
$$= \overline{S_1} \cdot \overline{S_2}$$
$$= \left(x_2 + \overline{x_3}\right) \cdot \left(x_1 + x_3\right)$$
$$f_2 = \overline{S_1 + S_3}$$
$$= \overline{S_1} \cdot \overline{S_3}$$
$$= \left(x_2 + \overline{x_3}\right) \cdot \left(x_1 + \overline{x_2} + x_3\right)$$



Izvedbe programirljivog logičkog polja

- izvedba FPLA (engl. Field-Programmable Logic Array):
  - (višekratno) programiranje "na licu mjesta"
  - NMOS Tr + programirljiva sklopka
  - programirljiva sklopka ~ EEPROM





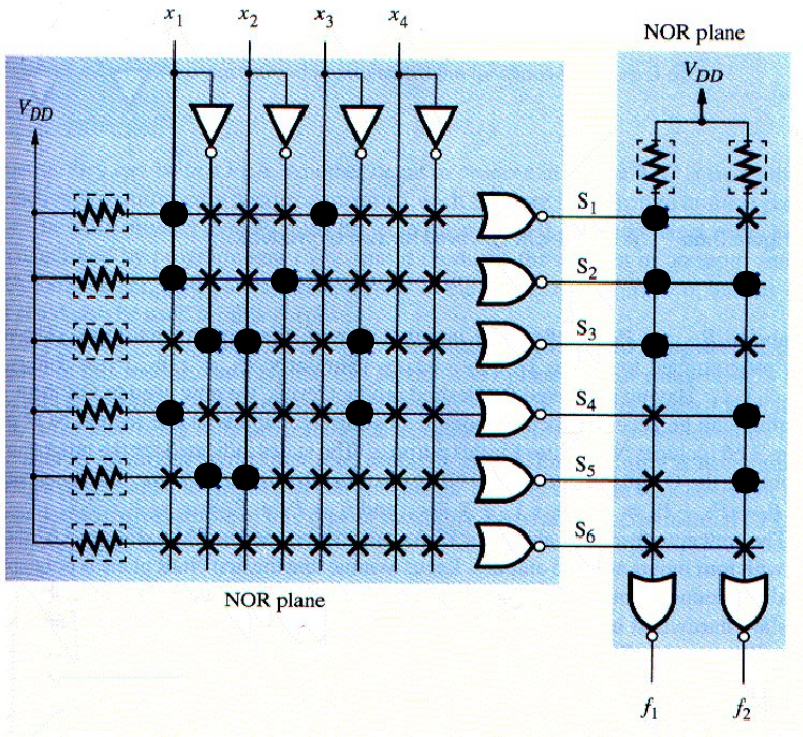
Izvedbe programirljivog logičkog polja

Primjer:ostvarenje funkcija s FPLA

●: neprogramirana sklopka

x: programirana, *nema* spoja

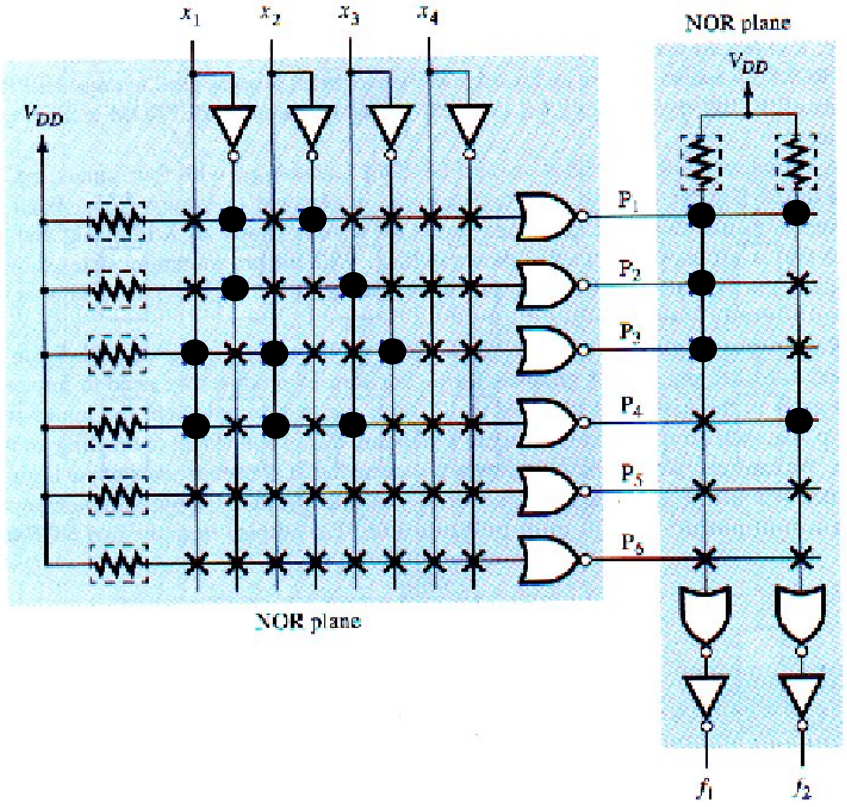
$$f_1 = \overline{S_1 + S_2 + S_3}$$
$$= \overline{S_1} \cdot \overline{S_2} \cdot \overline{S_3}$$
$$= (x_1 + x_3) \cdot (x_1 + \overline{x_2}) \cdot (\overline{x_1} + x_2 + \overline{x_3})$$
$$f_2 = \overline{S_2 + S_4 + S_5}$$
$$= \overline{S_2} \cdot \overline{S_4} \cdot \overline{S_5}$$
$$= (x_1 + \overline{x_3}) \cdot (\overline{x_1} + x_2) \cdot (x_1 + \overline{x_2})$$



Izvedbe programirljivog logičkog polja

- FPLA: ostvarenje dvorazinske funkcije tipa ILI-I
  - invertirati ulaze (x1, …, x4)
  - invertirati izlaz (f1, f2)

$$f_1 = P_1 + P_2 + P_3$$
$$= x_1 \cdot x_2 + x_1 \cdot \overline{x_3} + \overline{x_1} \cdot \overline{x_2} \cdot x_3$$
$$f_2 = P_1 + P_4$$
$$= x_1 \cdot x_2 + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$$

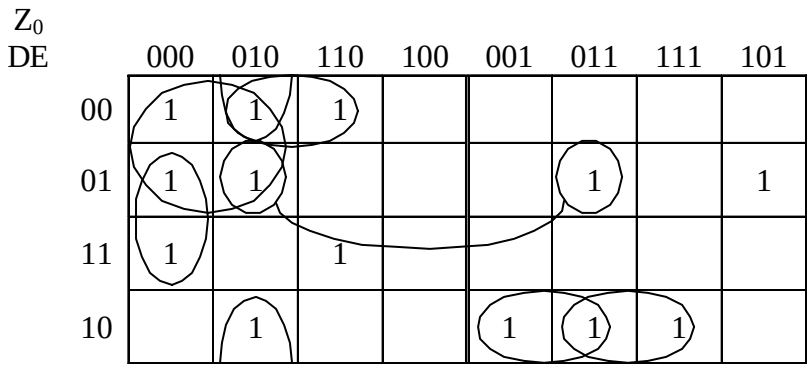


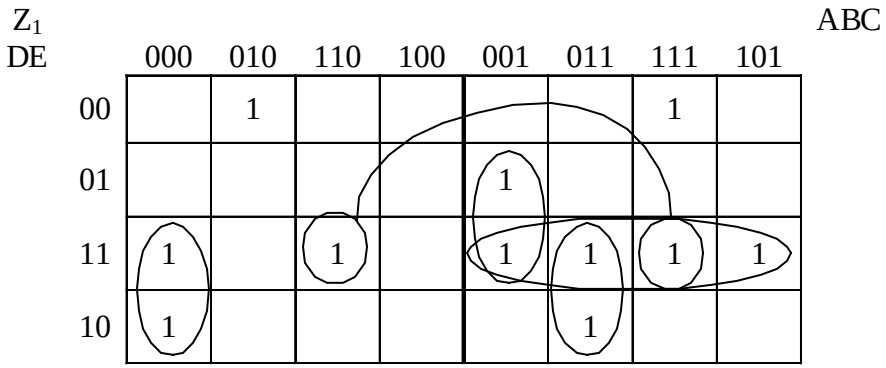
Generiranje Booleovih funkcija

Primjer: generiranje logičkih funkcija s PLA

$$Z_0 = \sum(0,1,3,6,8,9,10,13,14,21,24,27,30)$$

$$Z_1 = \sum(2,3,5,7,8,14,15,23,27,28,31)$$





$$Z_0 = \overline{A}\,\overline{C}\,\overline{D} + B\overline{C}\,\overline{D}\,\overline{E} + \overline{A}B\overline{C}\,\overline{E} + \overline{A}\,\overline{B}\,\overline{C}E + \overline{A}B\overline{D}E + \overline{A}CD\overline{E} + BCD\overline{E}$$
$$+ AB\overline{C}DE + A\overline{B}C\overline{D}E$$

$$Z_1 = CDE + \overline{A}\,\overline{B}CE + \overline{A}BCD + \overline{A}\,\overline{B}\,\overline{C}D + ABDE + \overline{A}B\overline{C}\,\overline{D}\,\overline{E} + ABC\overline{D}\,\overline{E}$$

Generiranje Booleovih funkcija

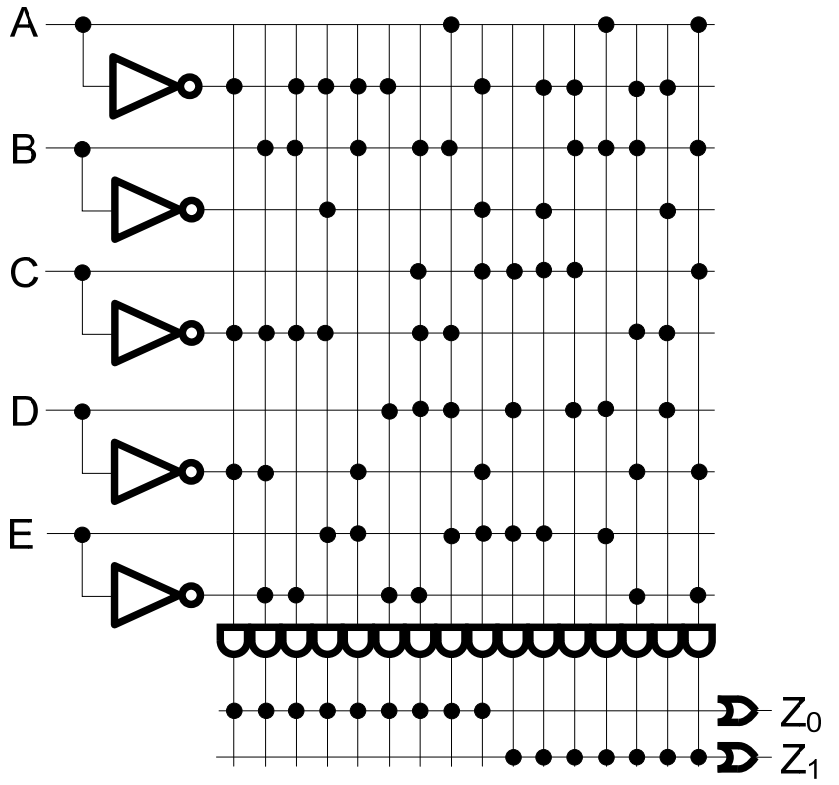
- matrični prikaz

$$Z_0 = \sum(0,1,3,6,8,9,10,13,14,21,24,27,30)$$

$$Z_1 = \sum(2,3,5,7,8,14,15,23,27,28,31)$$

$$Z_0 = \overline{A}\,\overline{C}\,\overline{D} + B\overline{C}\,\overline{D}\,\overline{E} + \overline{A}B\overline{C}\,\overline{E} + \overline{A}\,\overline{B}\,\overline{C}E + \overline{A}B\overline{D}E + \overline{A}C\overline{D}E + BC\overline{D}E + AB\overline{C}DE + A\overline{B}C\overline{D}E$$

$$Z_1 = CDE + \overline{A}\,\overline{B}CE + \overline{A}BCD + \overline{A}\,\overline{B}\,\overline{C}D + ABDE + \overline{A}B\overline{C}\,\overline{D}\,\overline{E} + ABC\overline{D}\,\overline{E}$$



Generiranje Booleovih funkcija

Zadatak :

- primjerenim sklopom PLA ostvariti:
  - potpuno zbrajalo
  - potpuno zbrajalo/odbijalo
  - BCD zbrajalo
- primjenom K-tablica i metode QuineMcCluskey sklopom PLA ostvariti višeizlaznu funkciju:
  - $f_1(A,B,C,D) = \sum m(0,1,2,3,6,7)$

    $f_2(A,B,C,D) = \sum m(0,1,6,7,14,15)$

    $f_3(A,B,C,D) = \sum m(0,1,2,3,8,9)$

  - $f_1(A,B,C,D,E) = \sum m(0,1,2,3,6,7,20,21,26,27,28)$

    $f_2(A,B,C,D,E) = \sum m(0,1,6,7,14,15,16,17,19,20,24,27)$

    $f_3(A,B,C,D,E) = \sum m(0,1,2,3,8,9,16,20,26,28,30)$

Sadržaj predavanja

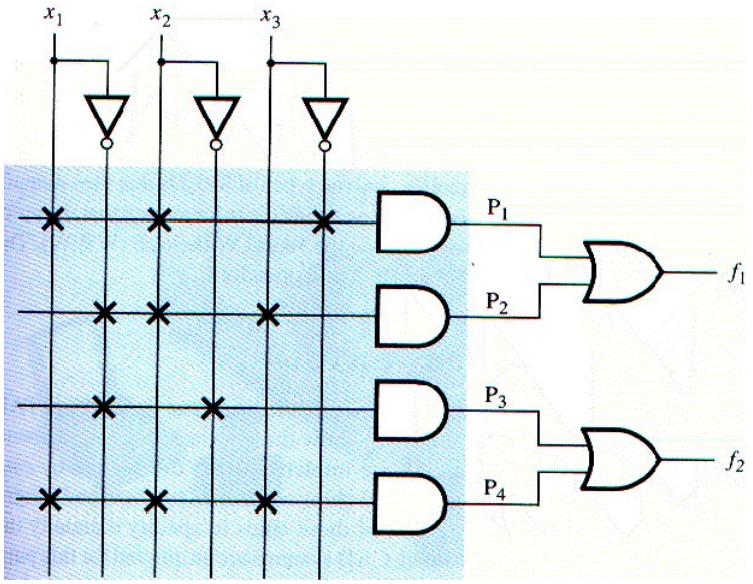
- koncept programirljivih modula
- permanentna memorija
- programirljivo logičko polje
- **poluprogramirljivo logičko polje**
  - **funkcionalnost i struktura**
  - **generiranje Booleovih funkcija**
- složeni programirljivi moduli
- programirljivo polje logičkih blokova

Poluprogramirljivo logičko polje

- problemi s PLA:
  - poteškoće s programirljivim sklopkama:
    - teško ih je ispravno proizvesti
    - redukcija brzine rada sklopa ostvarenog s PLA
  - ograničiti mogućnost programiranja na *samo jedno* polje:
    - jednostavnije za proizvodnju
    - jeftinije
    - bolje performanse
    - smanjena fleksibilnost

Poluprogramirljivo logičko polje

- *poluprogramirljivo polje*, PAL
  (engl. Programmable Array Logic)
  ~ jako popularno rješenje:



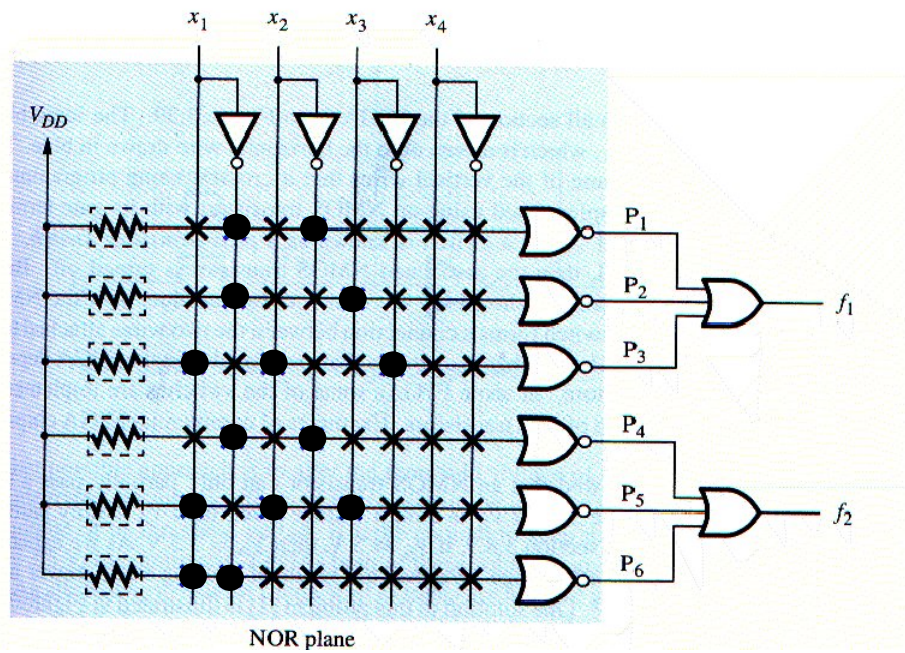
  - programira se samo I polje
  - komercijalne izvedbe:
    ~ 1000 programirljivih sklopki
  - programiranje CAD sustavom
  - kompenziranje reducirane funkcionalnosti
    ~ proizvodnja u širokom rasponu veličina
    (broj ulaza i izlaza, broj ulaza u ILI sklopove)

Generiranje Booleovih funkcija

Primjer: ostvarenje funkcije s PAL

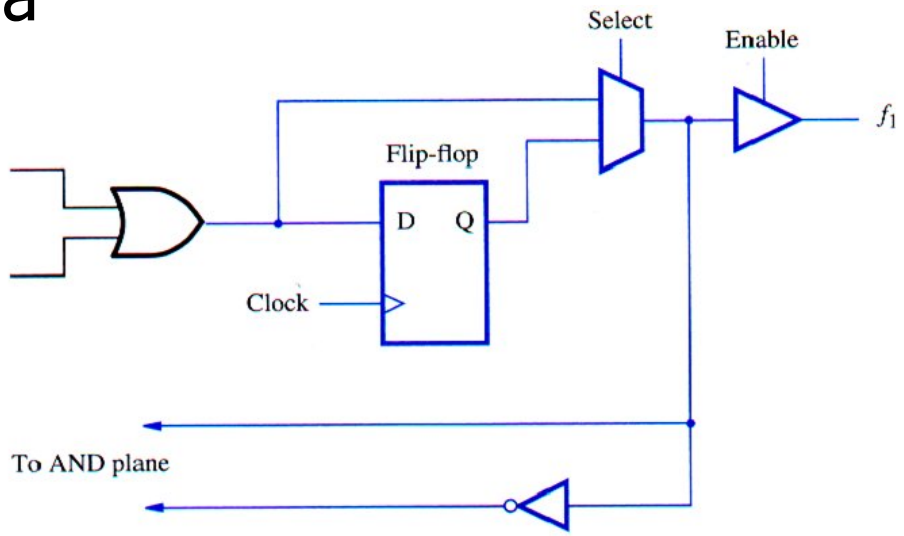
- *nema* višestrukog korištenja P_i
  (npr. $P_1 = P_4 = x_1 \cdot x_2$)
  ~ *ne koristi* se minimiziranje višeizlaznih funkcija!!!
- nekorišteni P_i programira se da daje 0:
  npr. $P_6 = x_1 \cdot \overline{x_1} = 0$

$$f_1 = P_1 + P_2 + P_3$$
$$= x_1 \cdot x_2 + x_1 \cdot \overline{x_3} + \overline{x_1} \cdot \overline{x_2} \cdot x_3$$
$$f_2 = P_4 + P_5$$
$$= x_1 \cdot x_2 + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$$



Generiranje Booleovih funkcija

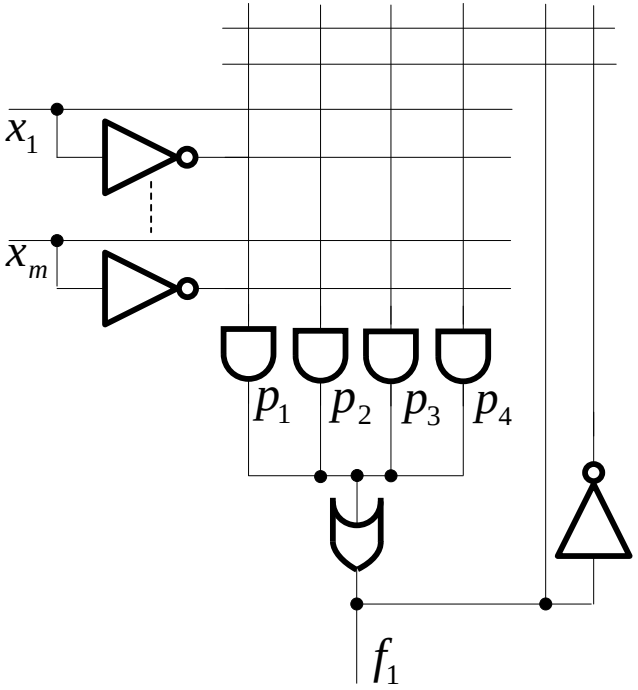
- povećanje fleksibilnosti
  ~ *makroćelija* (engl. macrocell):
  - dodatno sklopovlje na izlazu ILI sklopa
  - povratna veza na I polje
    ~ ostvarivanja složenijih (~ *višerazinskih*) sklopova



Generiranje Booleovih funkcija

Primjer: potpuno zbrajalo/odbijalo ostvareno makroćelijom

- K=0: zbrajalo, K=1: odbijalo
- uočiti: $S_i = D_i$, simetrija C_i i Z_i
- primitivna makroćelija

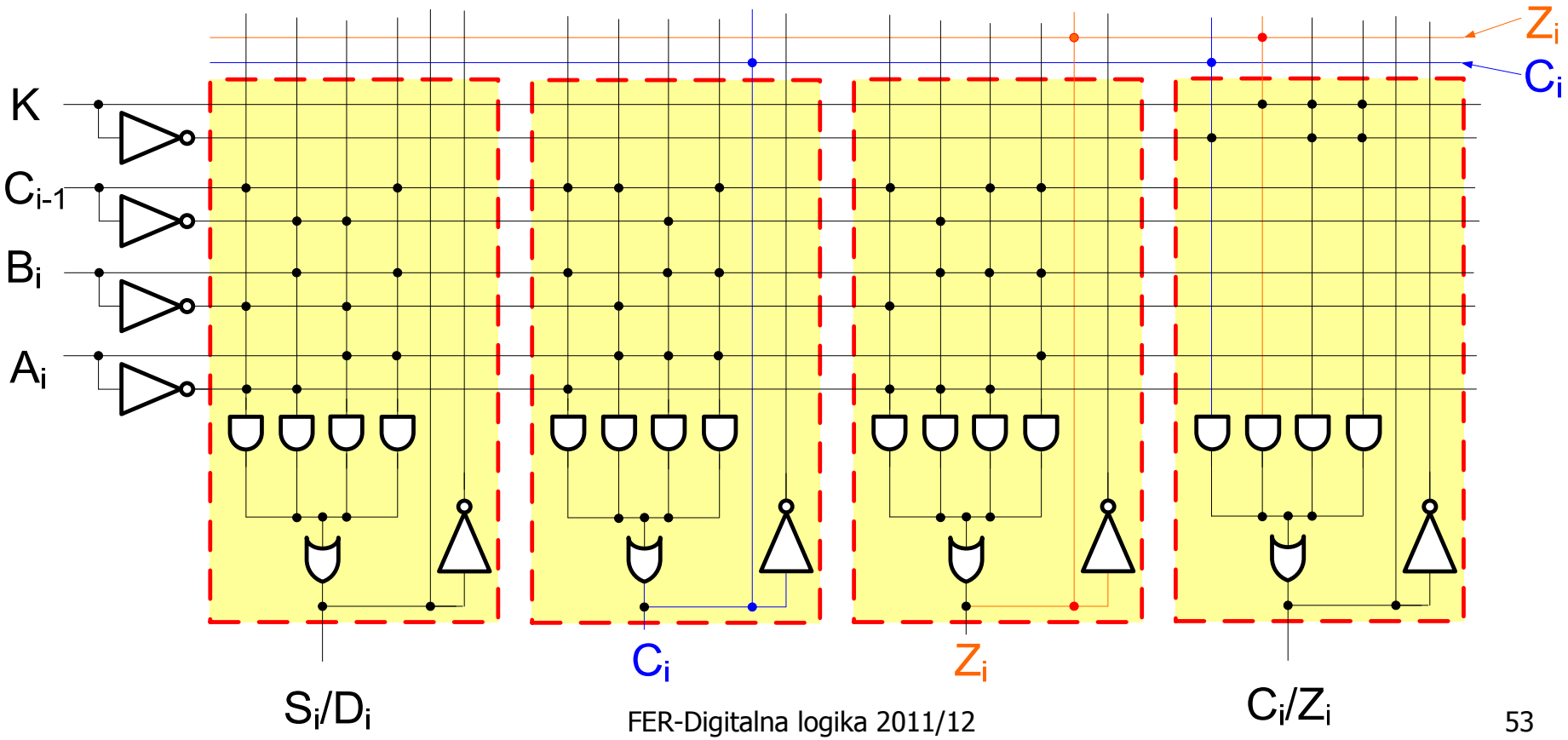


A_i	B_i	C_{i-1}	S_i	C_i	D_i	Z_i
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	1	1
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	1	0	0	1	0	0
1	1	1	1	1	1	1

Generiranje Booleovih funkcija

- potrebne su *četiri* makroćelije (zašto?)

A_i	B_i	C_{i-1}	S_i	C_i	D_i	Z_i
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	1	1
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	1	0	0	1	0	0
1	1	1	1	1	1	1



Sadržaj predavanja

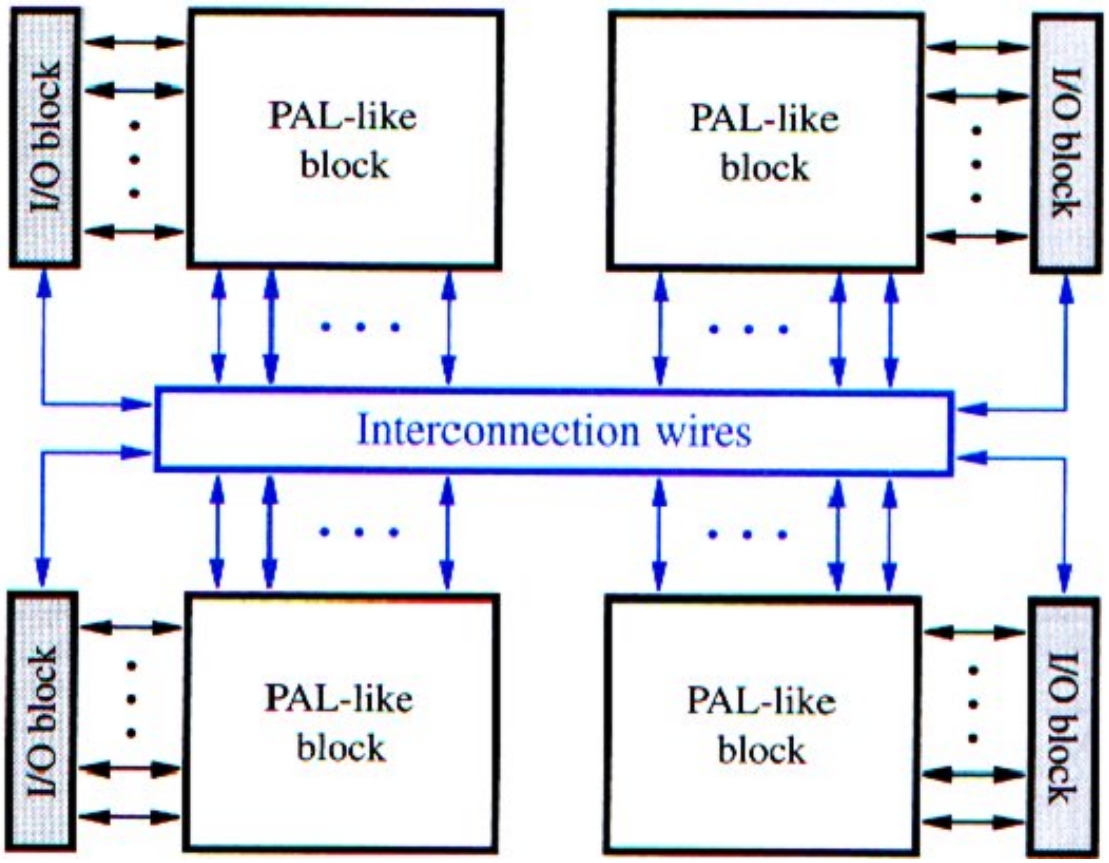
- koncept programirljivih modula
- permanentna memorija
- programirljivo logičko polje
- poluprogramirljivo logičko polje
- **složeni programirljivi moduli**
- programirljivo polje logičkih blokova

Složeni programirljivi moduli

- programirljivi moduli:
  - SPLD (= PLA, PAL)
    ~ skromne dimenzije sklopa
  - složeni PLD, CPLD:
    - *više* blokova sa sklopovljem (~ SPLD)
    - mogućnost internog povezivanja blokova
    - tipične dimenzije:
      - 2÷100 "PALu sličnih blokova "
      - 16 makroćelija u " PALu sličnom bloku "
      - 5÷20 ulaza u ILI sklopove
      - EX-ILI za programirljivo komplementiranje izlaza
      - izlaz iz makroćelije
        ~ sklop s tri stanja

Složeni programirljivi moduli

- struktura CPLD:

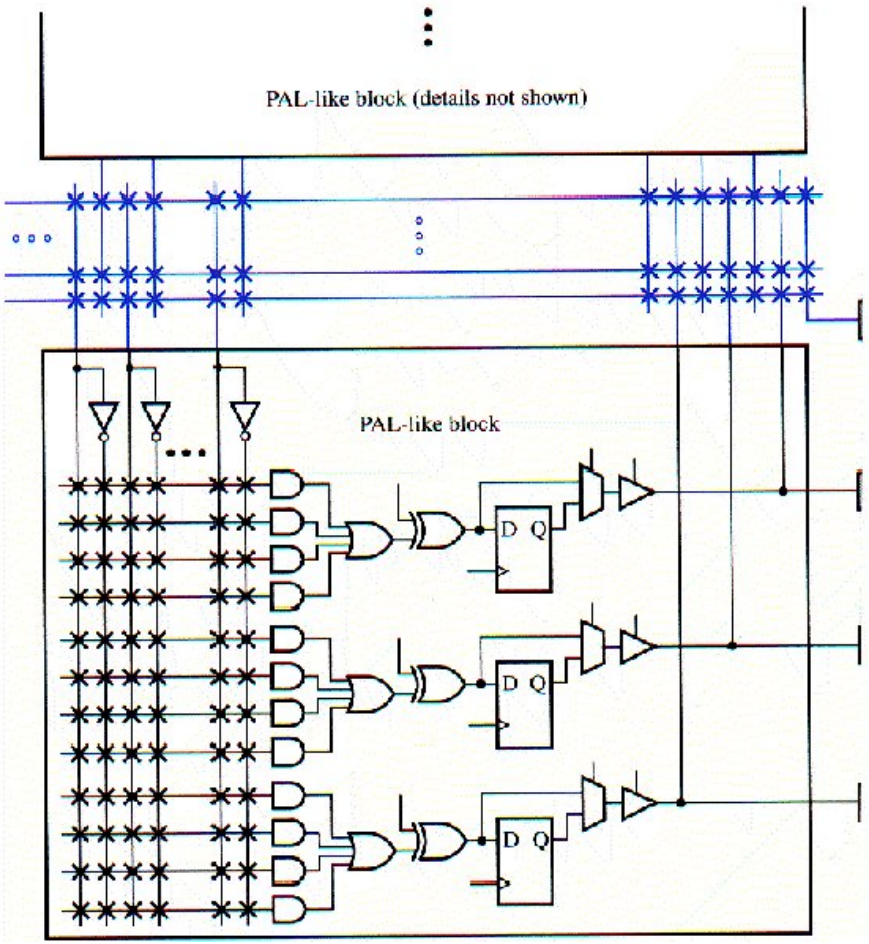


Složeni programirljivi moduli

- struktura CPLD:
  - *PALu slični blokovi* (engl. PAL-like blocks)
    ~ ostvarenje primitivnije funkcije
  - *UI blokovi*
    ~ sučelje za svaki PALu slični blok
  - *povezno ožičenje* (engl. interconnection wires):
    - programirljive sklopke za povezivanje PALu sličnih blokova
    - broj programirljivih sklopki:
      ~ pažljiva procjena! (fleksibilnost-efikasnost)
    - vertikalne linije
      ~ ulazi u makroćeliju i izlazi iz nje
    - problem:
      ~ ako je izvod IC korišten kao izlaz, pripadna je makroćelija neupotrebljiva

Složeni programirljivi moduli

- struktura PALu sličnog bloka:



Složeni programirljivi moduli

- programiranje CPLD
  ~ tipično *u sustavu u kojem se koriste* (engl. In-System Programming, ISP)
  - mehanički razlozi:
    ~ IC s velikim brojem (~100÷200) krhkih i savitljivih izvoda
  - *podnožja* (engl. sockets) su sumjerljivo skupa
  - posebni *konektor* povezan na sve CPLD u sustavu
    ~ *JTAG pristup* (engl. JTAG port, Joint Test Action Group), standard IEEE
  - *stalno* (engl. nonvolatile) programiranje
  - primjena CPLD u ~50% slučajeva korištenja PLD

Sadržaj predavanja

- koncept programirljivih modula
- permanentna memorija
- programirljivo logičko polje
- poluprogramirljivo logičko polje
- složeni programirljivi moduli
- **programirljivo polje logičkih blokova**
  - **struktura i logički blokovi**
  - **generiranje Booleovih funkcija**

Programirljivo polje logičkih blokova

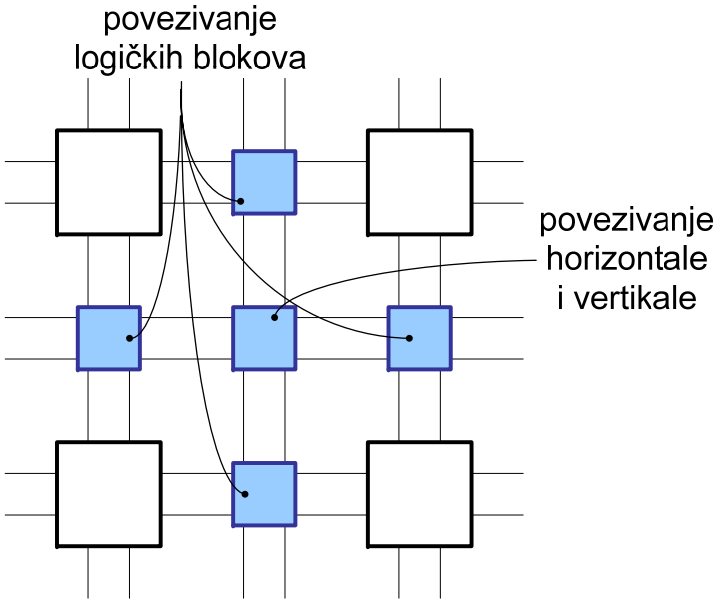
- mjera složenosti digitalnog sklopa
  ~ broj *ekvivalentnih sklopova* (engl. equivalent gates): ukupni broj NI sklopova s 2 ulaza koji bi bili potrebni za njegovo ostvarenje
- SPLD/CPLD podržavaju ostvarenja relativno *jednostavnijih* sklopova
  - makroćelije SPLD/CPLD: ~20
    npr. PAL s 8 makroćelija: ~160
    CPLD s 1000 makroćelija: ~20.000
  - smanjenje troškova i povećanja performansi
    $\Rightarrow$ sklop sa *što manjim* brojem IC

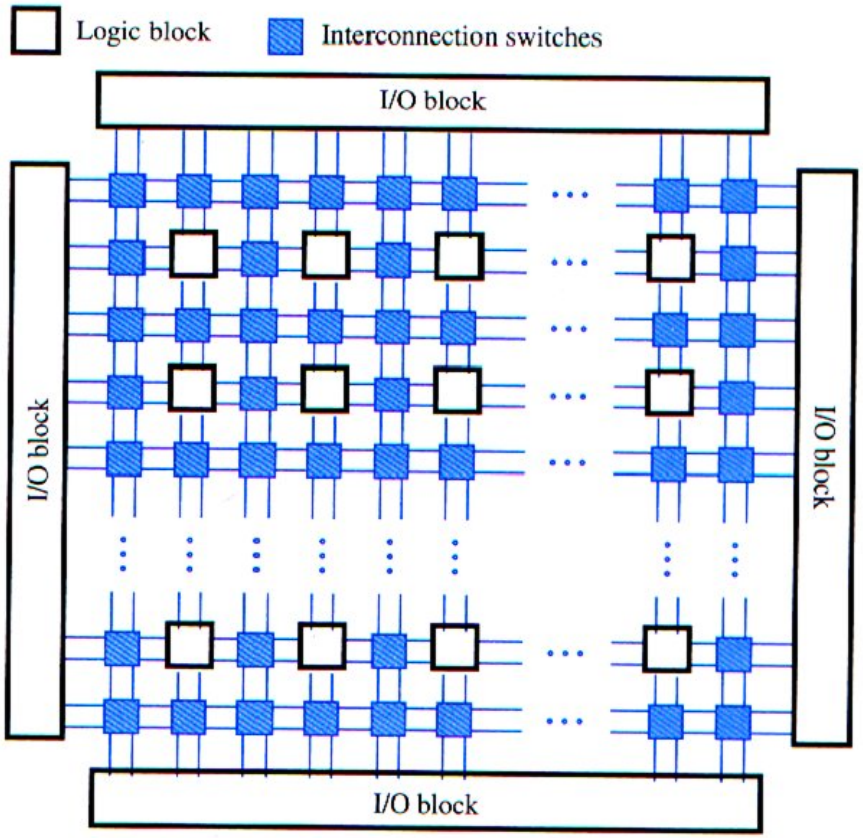
Programirljivo polje logičkih blokova

- *programirljivo polje logičkih blokova* (engl. Field Programmable Gate Array, FPGA)
  - PLD za ostvarivanje *relativno velikih* digitalnih sklopova ($\geq$ 100.000 ekvivalentnih sklopova)
  - *logički blokovi* (engl. logic blocks, LB) umjesto I/ILI polja
  - tipična struktura FPGA:
    - LB organizirani u *dvodimenzijsko* polje
    - *UI blokovi* za sučelje (s izvodima IC)
    - vodovi i programirljive sklopke
      ~ za međusobno povezivanje LB:
      horizontalni i vertikalni
      kanali za usmjeravanje (engl. routing channels)
      između redaka i stupaca LBova

Programirljivo polje logičkih blokova

- općenita struktura FPGA:
  - bijela polja ~ logički blokovi (LBovi)
  - plava polja ~ programirljive sklopke





Programirljivo polje logičkih blokova

- općenita struktura *logičkog bloka*:
  - više ulaza, jedan izlaz
    ~ ostvaruje *jednostavnu* Booleovu funkciju
  - više tipova, najuobičajeniji koristi preglednu tablicu (LUT) s memorijskim ćelijama:
    - ostvarivanje funkcije primjerenim MUX
      ~ izvedba multipleksorskim stablom
    - MUX u funkciji permanentne memorije!
    - tipično:
      LUT s 4/5 ulaza $\rightarrow$ 16/32 ćelija
      + neko dodatno sklopovlje

Generiranje Booleovih funkcija

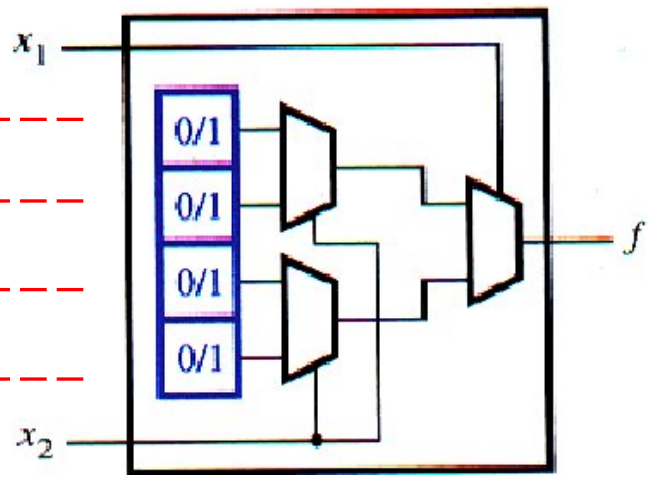
- struktura logičkog bloka s 2 ulaza

  2 ulaza $\rightarrow$ 2-ulazni MUX: f = f(x_1, x_2)

  y_i: sadržaj memorijskih ćelija

$$f = \overline{x_1} \cdot \left(\overline{x_2} \cdot y_0 + x_2 \cdot y_1\right) + x_1 \cdot \left(\overline{x_2} \cdot y_2 + x_2 \cdot y_3\right)$$
$$= \overline{x_1} \cdot \overline{x_2} \cdot y_0 + \overline{x_1} \cdot x_2 \cdot y_1 + x_1 \cdot \overline{x_2} \cdot y_2 + x_1 \cdot x_2 \cdot y_3$$

x_1	x_2	f
0	0	y_0
0	1	y_1
1	0	y_2
1	1	y_3

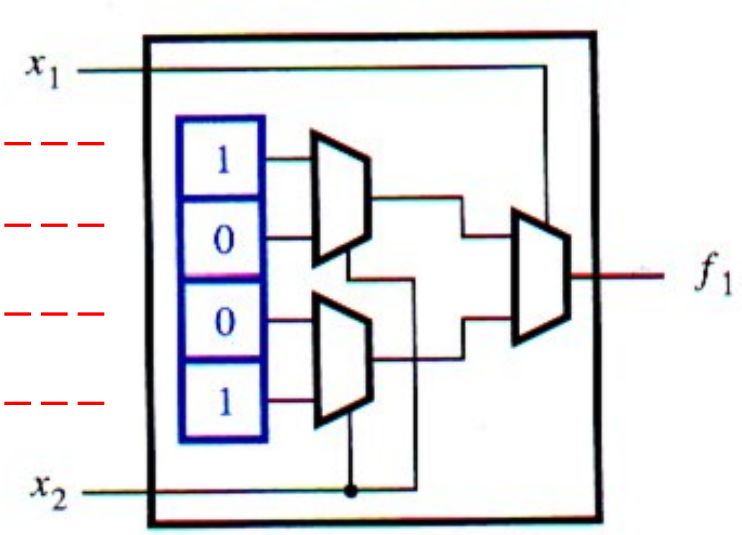


Generiranje Booleovih funkcija

Primjer : logički blok s 2 ulaza

sadržaj ćelija $\underline{y}$ = <1, 0, 0, 1>

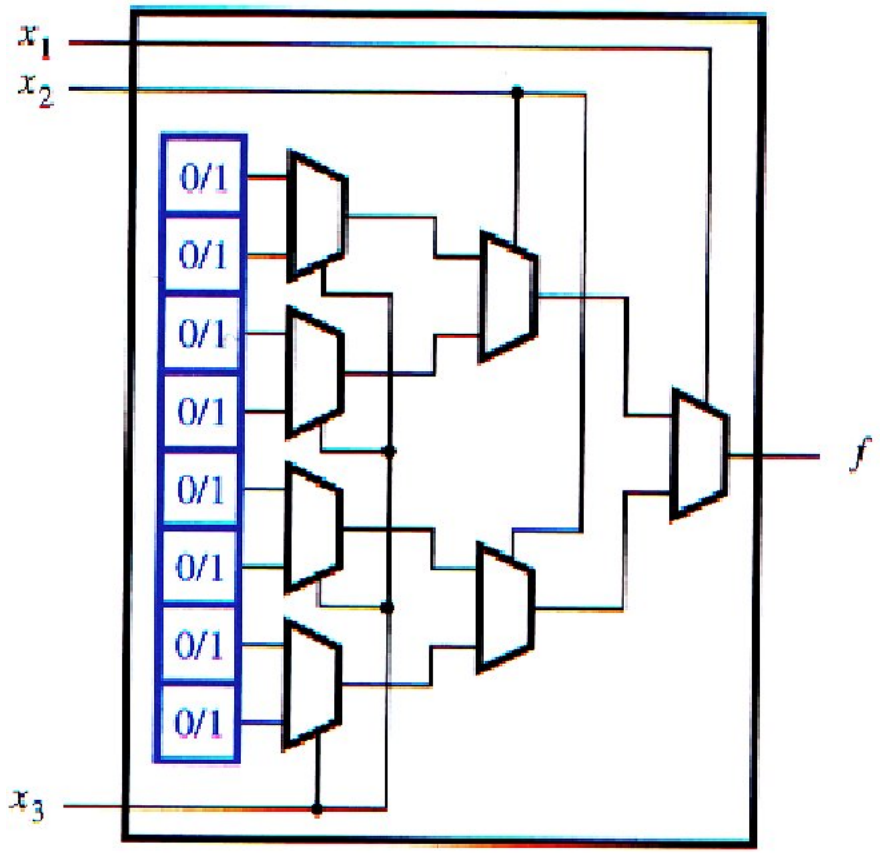
x_1	x_2	f
0	0	1
0	1	0
1	0	0
1	1	1



$$f_1 = \overline{x_1} \cdot \overline{x_2} + x_1 \cdot x_2$$

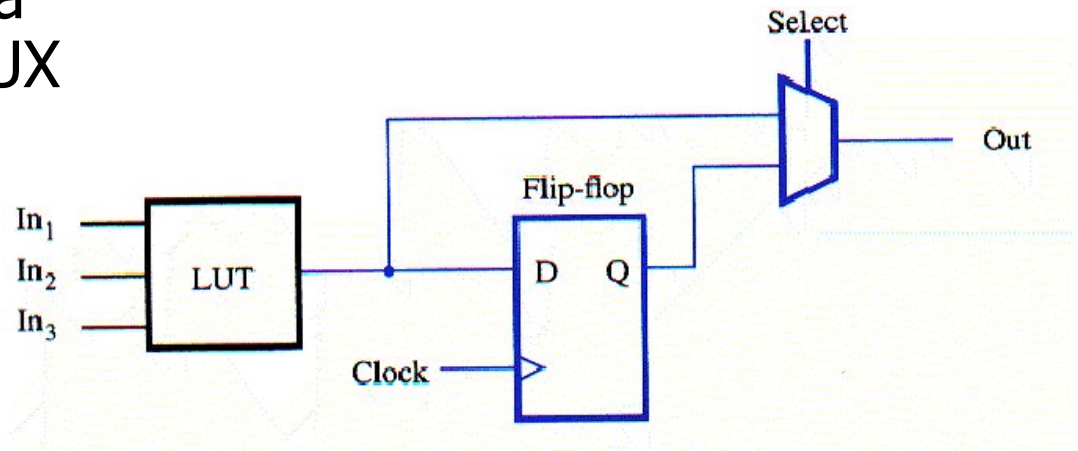
Generiranje Booleovih funkcija

- struktura logičkog bloka s 3 ulaza
  ~ 8 memorijskih ćelija



Generiranje Booleovih funkcija

- "dodatna logika" u logičkom bloku
  ~ makroćelije:
  - element za pamćenje
    (D bistabil: memorira 1 bit)
  - odabir izlaza
    ~ izlazni MUX



$$Out = \overline{Select} \cdot f + Select \cdot f^{n-1}$$

$$f = f(In_1, In_2, In_3)$$

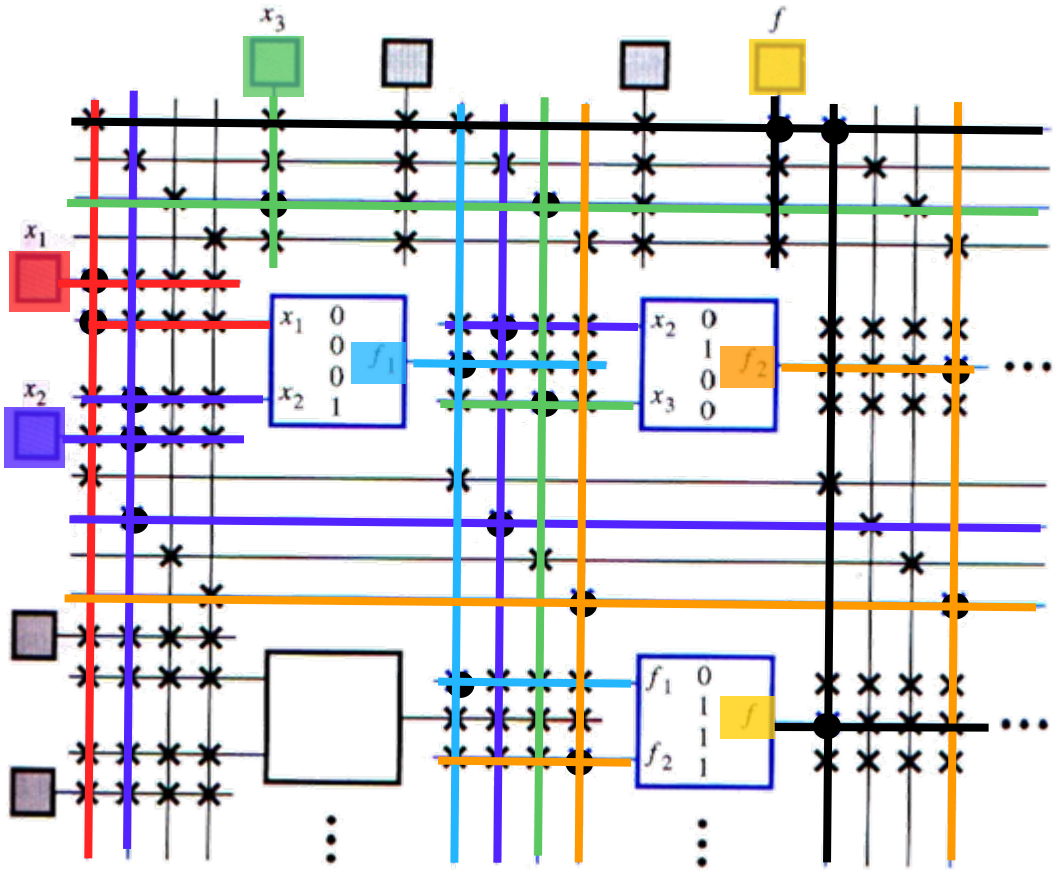
Generiranje Booleovih funkcija

- *programiranje* FPGA
  ~ također ISP:
  - memorijske ćelije LUT:
    ~ *nestalne* (engl. volatile):
    (EA)ROM za pohranjivanje sadržaja LUT
  - automatsko *punjenje* (engl. loading)
    prilikom uključivanja uređaja

Generiranje Booleovih funkcija

Primjer: (dio) programiranog FPGA

$f(x_1, x_2, x_3) = ?$



$$f_1 = x_1 \cdot x_2$$

$$f_2 = \overline{x_2} \cdot x_3$$

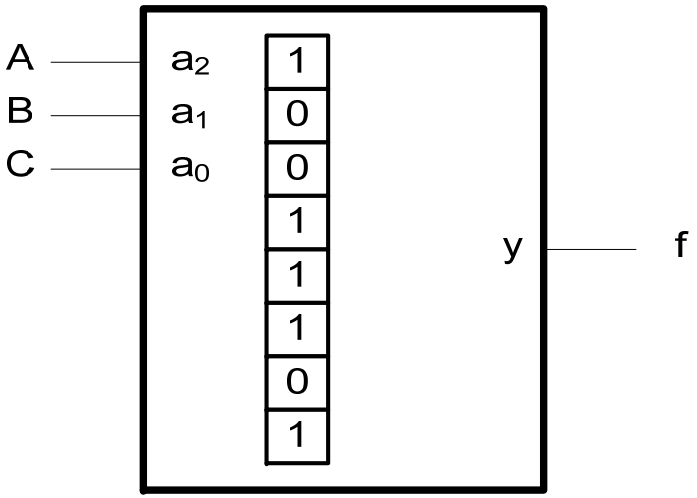
$$f = f_1 + f_2$$

$$= x_1 \cdot x_2 + \overline{x_2} \cdot x_3$$

Generiranje Booleovih funkcija

Primjer: $f(A,B,C) = B \cdot C + \overline{B} \cdot \overline{C} + A \cdot C$

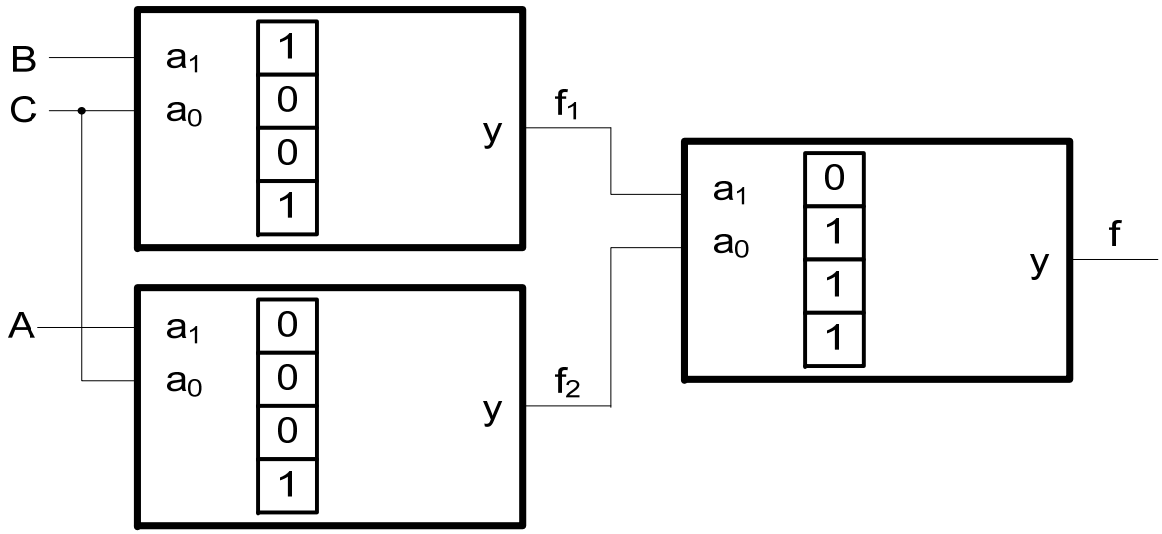
- ostvarenje LUTom s 3 ulaza
  ~ standardno "programiranje" multipleksora



Generiranje Booleovih funkcija

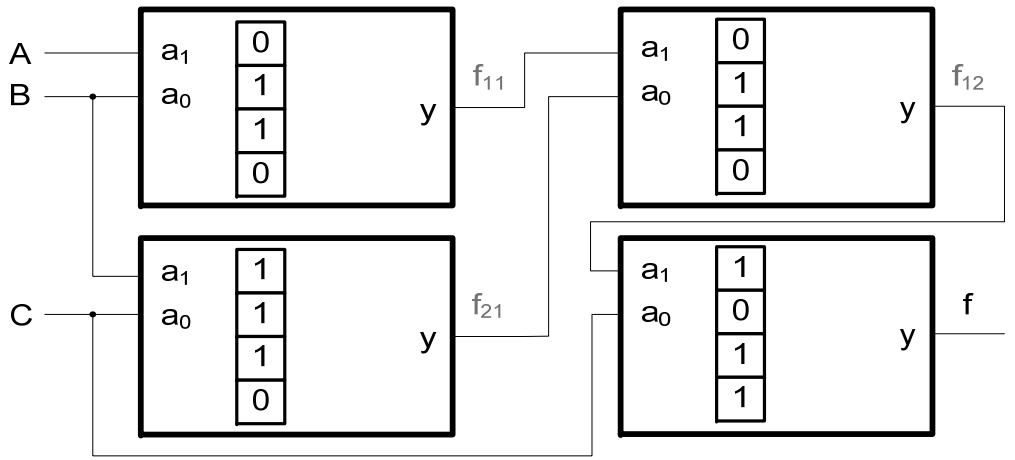
- ostvarenje LUTovima s 2 ulaza
  ~ kaskadiranje multipleksora

$$f(A,B,C) = B \cdot C + \overline{B} \cdot \overline{C} + A \cdot C$$
$$= (B \cdot C + \overline{B} \cdot \overline{C}) + A \cdot C$$
$$= f_1 + f_2$$



Generiranje Booleovih funkcija

Primjer : f = ?



$$f_{11}(A,B) = A \oplus B$$

$$f_{21}(B,C) = \overline{BC}$$

$$f_{12}(f_{11}, f_{21}) = f_{11} \oplus f_{21} = A \oplus B \oplus \overline{BC}$$

$$f(f_{12}, C) = f_{12} + \overline{C} = A \oplus B \oplus \overline{BC} + \overline{C} = A \oplus B \oplus (\overline{B} + \overline{C}) + \overline{C}$$

$$f = A + BC + \overline{B}\,\overline{C}$$

Literatura

U. Peruško, V. Glavinić: *Digitalni sustavi*, Poglavlje 7: Standardni kombinacijski moduli.

- koncept programirljivih modula: str. 276
- permanentna memorija: str. 267-275
- programirljivo logičko polje: str. 276-281
- poluprogramirljivo logičko polje: str. 281-282
- složeni programirljivi moduli: str. 283-284
- programirljivo polje logičkih blokova: str. 285-286

Zadaci za vježbu (1)

U. Peruško, V. Glavinić: *Digitalni sustavi*, Poglavlje 7: Standardni kombinacijski moduli.

- permanentna memorija: 7.27-7.29
- programirljivo logičko polje: 7.2, 7.30-7.32, 7.34
- programirljivo polje logičkih blokova: 7.33

Zadaci za vježbu (2)

M. Čupić: *Digitalna elektronika i digitalna logika. Zbirka riješenih zadataka*, Cjelina 5: Standardni kombinacijski moduli. Cjelina 6: Standardni programirljivi moduli.

- permanentna memorija:
  - riješeni zadaci: 5.12b-5.15
- programirljivo logičko polje
  - riješeni zadaci: 6.1-6.4, 6.7, 6.13
  - zadaci za vježbu: 1-3
- poluprogramirljivo logičko polje
  - riješeni zadaci: 6.5-6.6
- programirljivo polje logičkih blokova
  - riješeni zadaci: 6.8-6.11