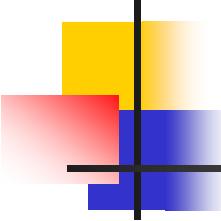


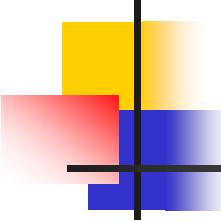
9. Programirljivi moduli

- permanentna memorija
- programirljivo logičko polje
- poluprogramirljivo logičko polje
- složene programirljive naprave
- programirljivo polje logičkih blokova



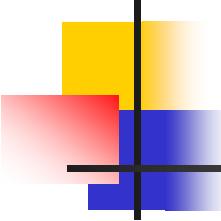
Programirljivi moduli

- *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



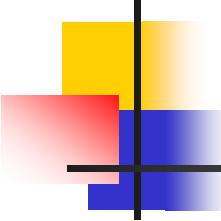
Programirljivi moduli

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



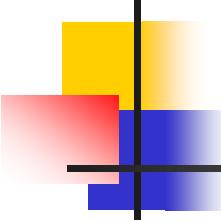
Programirljivi moduli

- 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



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"

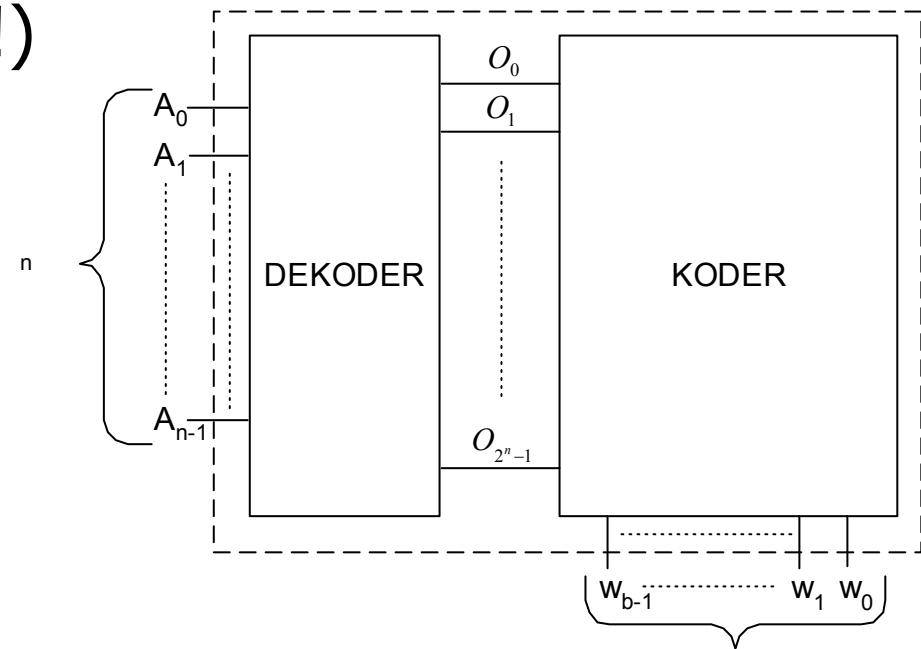


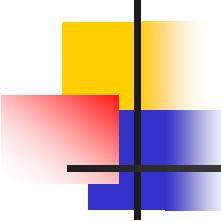
Permanentna memorija

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

Permanentna memorija

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



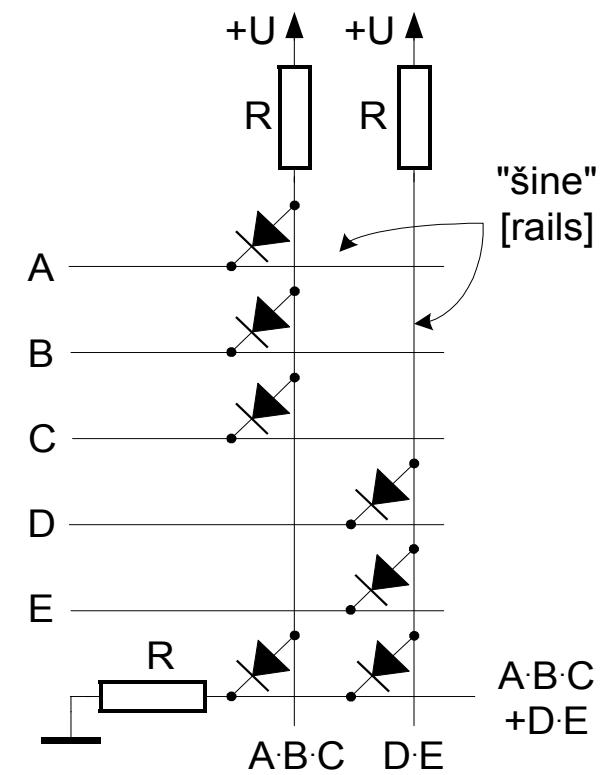
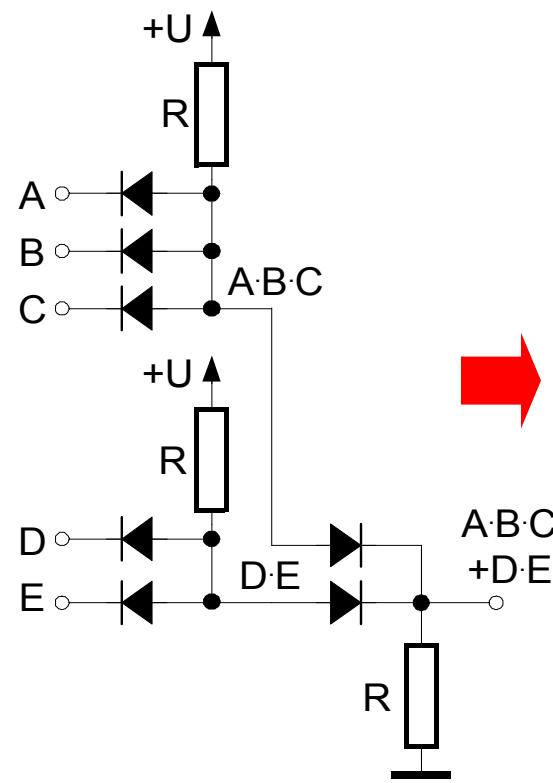
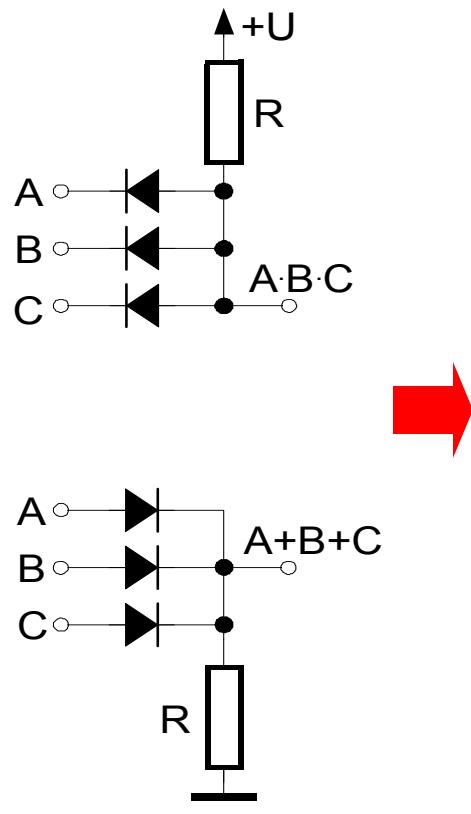


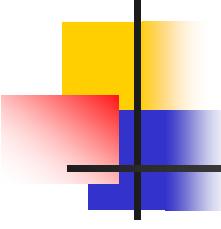
Permanentna memorija

- 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

Permanentna memorija

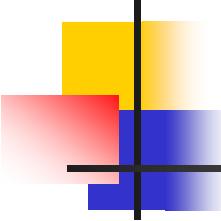
- izvorna izvedba
~ *diodna matrica*





Permanentna memorija

- 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

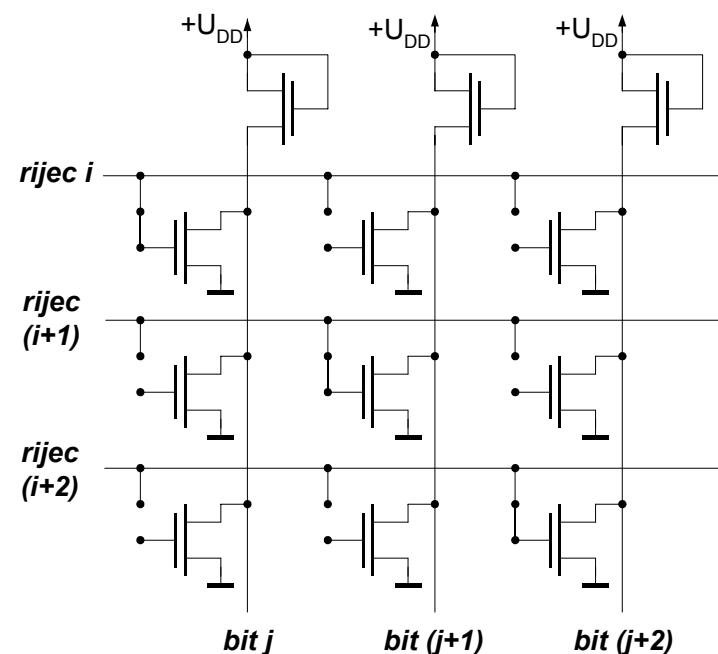


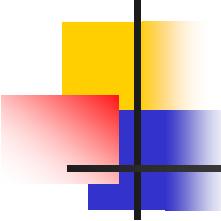
Permanentna memorija

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

Permanentna memorija

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



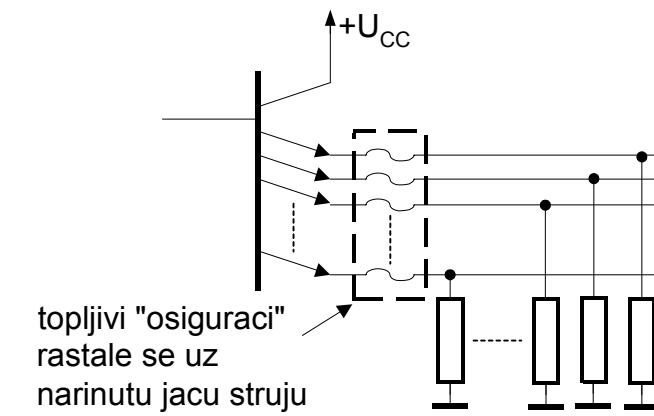


Permanentna memorija

- temeljeći sa na izvedbi permanentne memorija MOSFET tranzistorima 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 ($K = 1$))

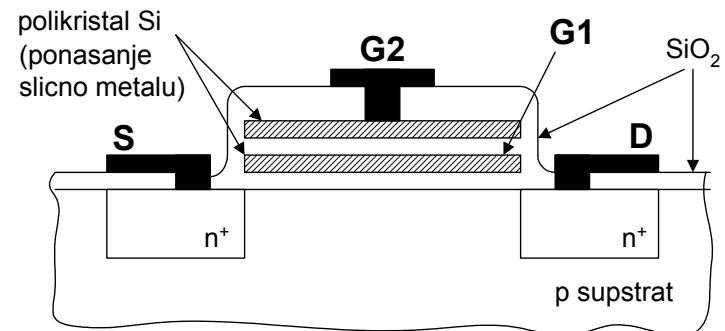
Permanentna memorija

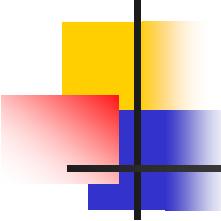
- s mogućnošću *jednokratnog* programiranja, PROM:
 - bipolarna tehnologija, tipično TTL
 - ~ višeemitorski tranzistor
 - za male serije
 - ~ programiranje "na licu mjesta" (engl. in-the-field)
 - programiranje
 - ~ jačom strujom ($U_B \gg$)
 - $t_a \sim 30 \div 50 \text{ ns}$



Permanentna memorija

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



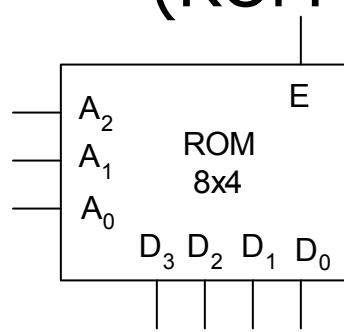


Permanentna memorija

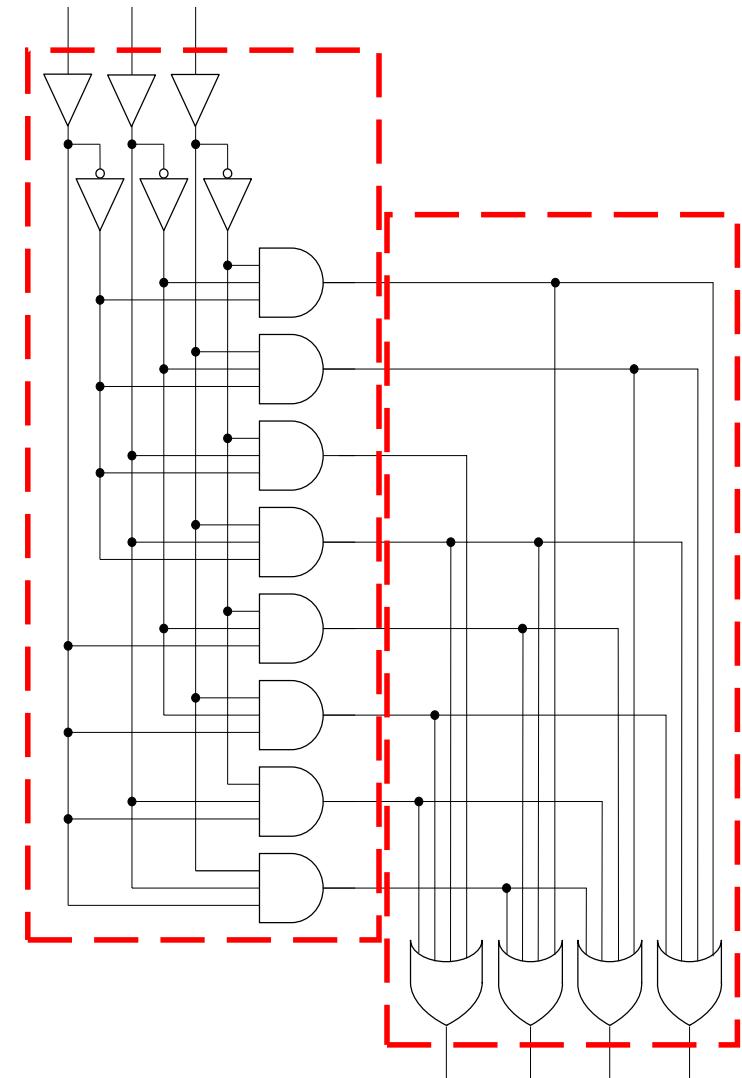
- s mogućnošću *višekratnog* programiranja i brisanja *električkim putem*, EROM, EEPROM:
 - izbjeći probleme EPROM
 - ~ dugo brisanje cijelog sadržaja u posebnom uređaju
 - smanjen razmak G_1 i D
 - ~ upisivanje i brisanje podatka *tuneliranjem*
(upis: $U_{G2D} \sim 10$ V, brisanje: $U_{G2D} \sim -10$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

Permanentna memorija

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

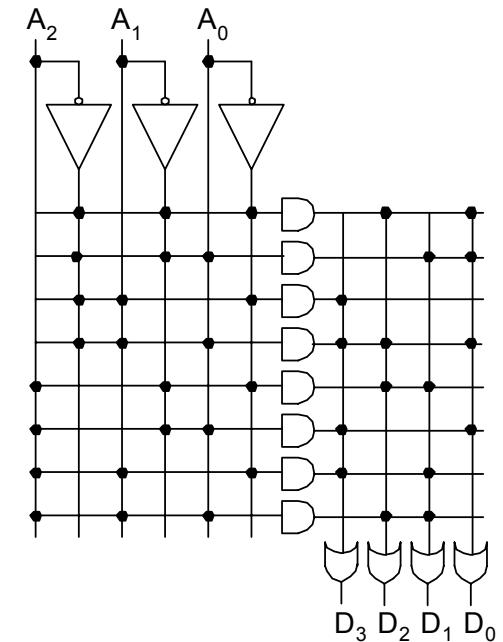
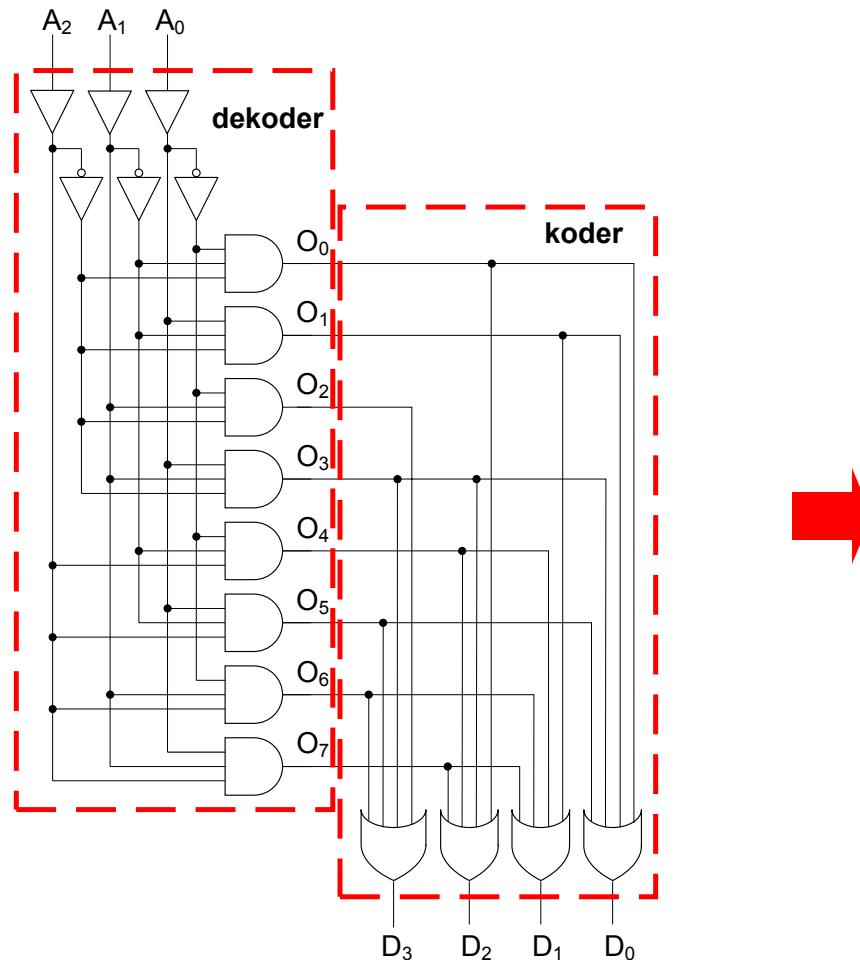


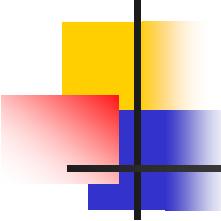
riječ	A ₂	A ₁	A ₀	D ₃	D ₂	D ₁	D ₀
0	0	0	0	0	1	0	1
1	0	0	1	0	0	1	1
2	0	1	0	1	0	0	0
3	0	1	1	1	1	0	1
4	1	0	0	0	1	1	0
5	1	0	1	1	0	0	1
6	1	1	0	1	0	1	0
7	1	1	1	0	1	1	0



Permanentna memorija

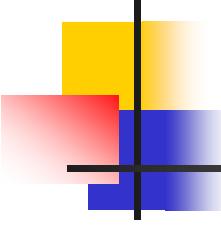
- polje = matrica
~ *matrični prikaz*





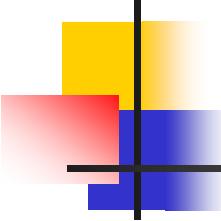
Permanentna memorija

- 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



Permanentna memorija

- permanentnom memorijom ostvariti pretvornike koda:
 - BCD u 2421
 - BCD u 7-segmentni
 - koji su parametri (broj riječi \times broj bitova/rijec)
potrebnih permanentnih memorija?



Permanentna memorija

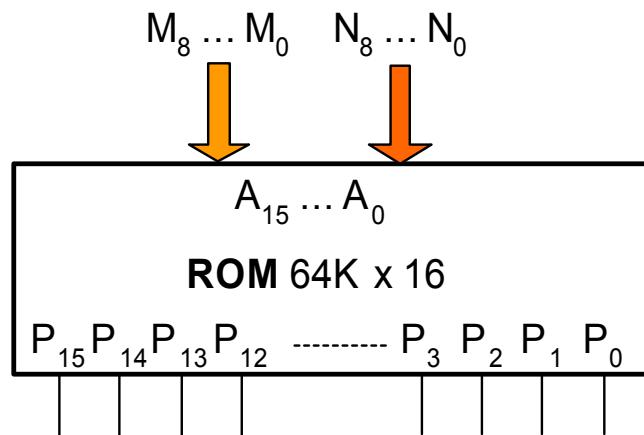
Primjer: sklop za množenje 8-bitnih brojeva

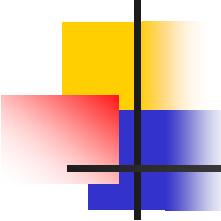
- tablica množenja ugrađena u ROM
 - ~ *pregledna tablica* (engl. Look-Up Table, LUT)
- efikasnija izvedba:
 - *kombinacija* izvjesnog broja ROMova
značajno manjeg kapaciteta (parcijalni produkti)
i zbrajala
 - veća kašnjenja!

Permanentna memorija

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

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





Permanentna memorija

- "rastavljanje" multiplikanda i multiplikatora:

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

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

$$P = M \cdot N$$

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

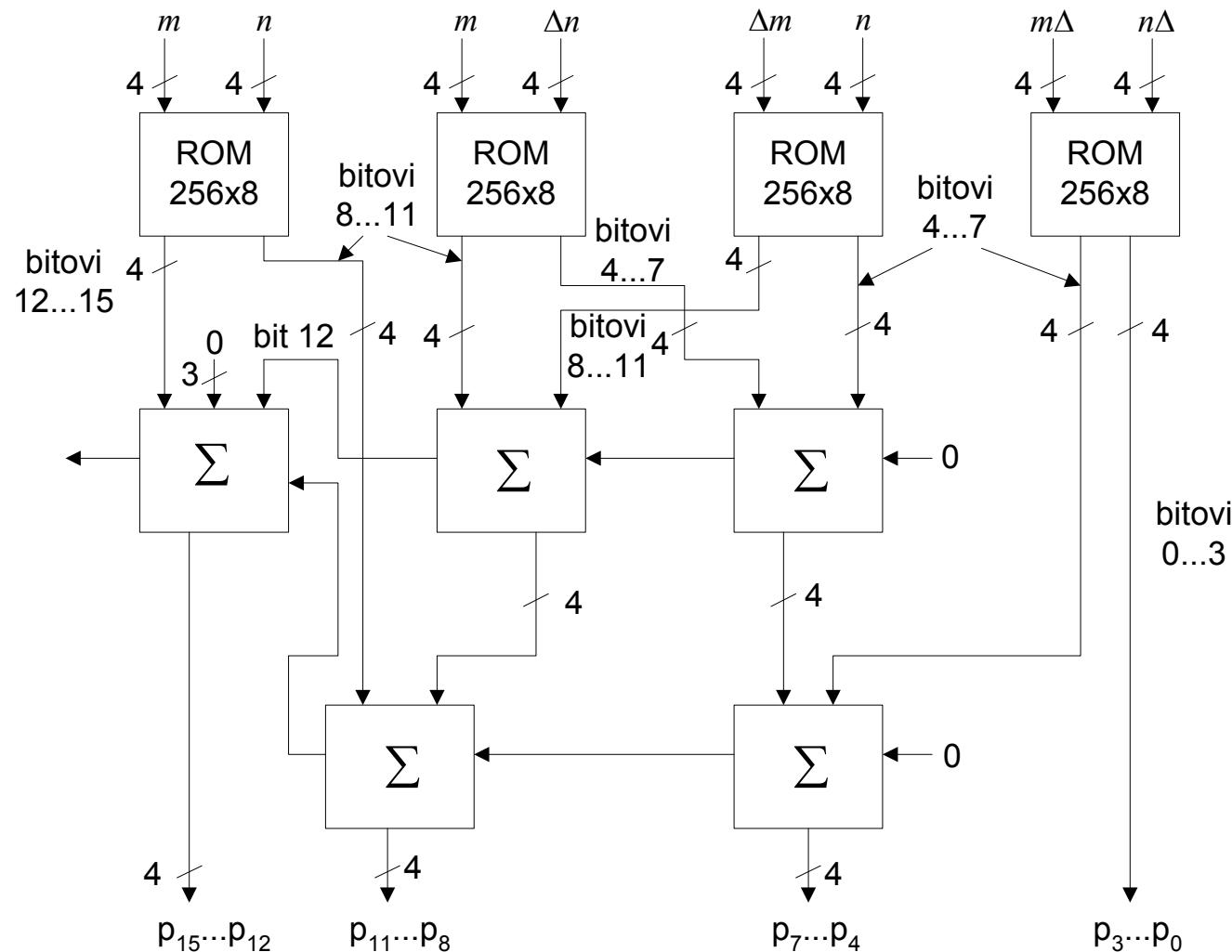
$$= (m \cdot n) \cdot 2^8 + (\Delta m \cdot n + m \cdot \Delta n) \cdot 2^4 + \Delta m \cdot \Delta n$$

- dovoljan puno manji kapacitet:

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

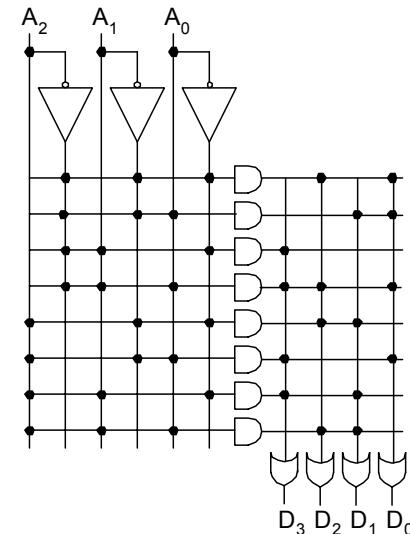
Permanentna memorija

- sklop zasnovan na kompoziciji manjih ROMova:



Permanentna memorija

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



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

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

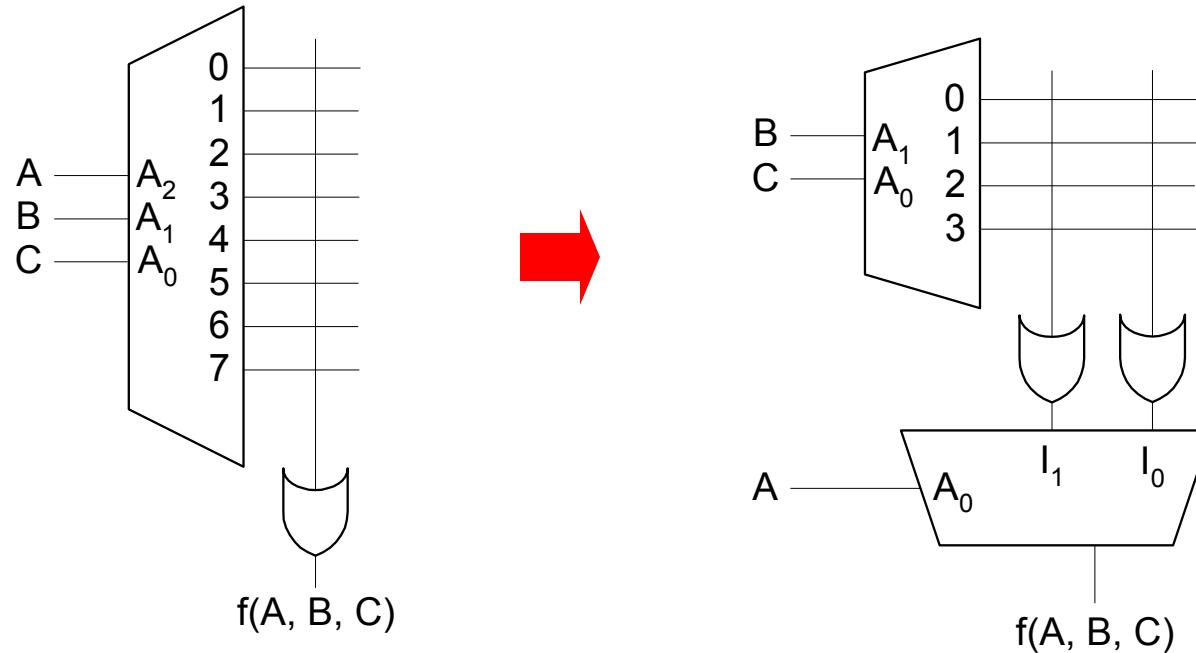
$$f_1 = \sum m(1,4,6,7)$$

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

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

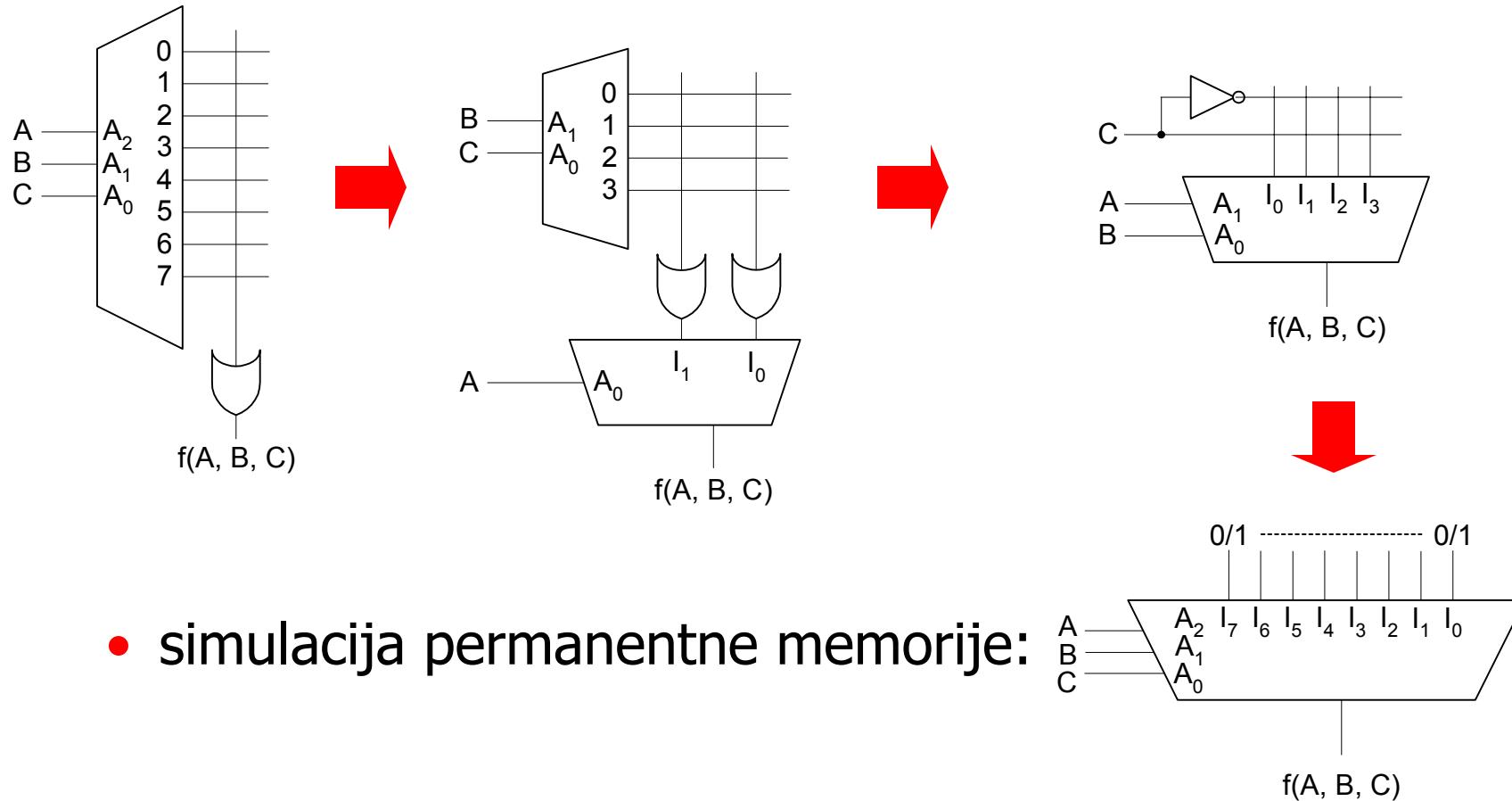
Permanentna memorija

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



Permanentna memorija

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



- simulacija permanentne memorije:

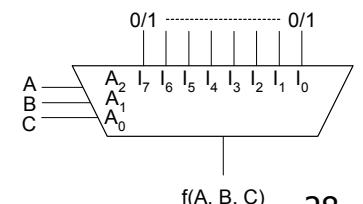
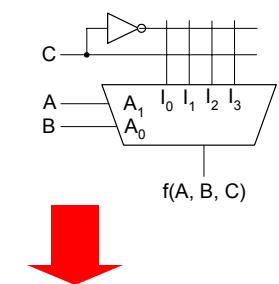
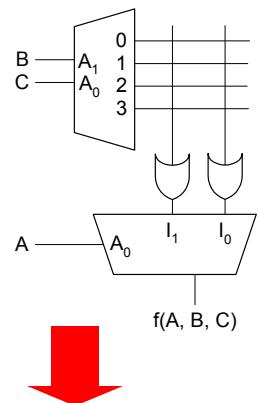
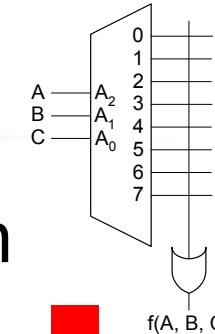
Permanentna memorija

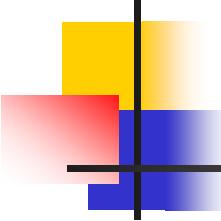
- ostvarivanja funkcija ROMom i multipleksorom
 \sim *Shannonova ekspanzija:*

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

$$\begin{aligned} f(A, B, C) &= f(0, 0, C) \cdot \bar{A} \cdot \bar{B} + f(0, 1, C) \cdot \bar{A} \cdot B \\ &\quad + f(1, 0, C) \cdot A \cdot \bar{B} + f(1, 1, C) \cdot A \cdot B \end{aligned}$$

$$\begin{aligned} f(A, B, C) &= f(0, 0, 0) \cdot \bar{A} \cdot \bar{B} \cdot \bar{C} + f(0, 0, 1) \cdot \bar{A} \cdot \bar{B} \cdot C \\ &\quad + f(0, 1, 0) \cdot \bar{A} \cdot B \cdot \bar{C} + f(0, 1, 1) \cdot \bar{A} \cdot B \cdot C \\ &\quad + f(1, 0, 0) \cdot A \cdot \bar{B} \cdot \bar{C} + f(1, 0, 1) \cdot A \cdot \bar{B} \cdot C \\ &\quad + f(1, 1, 0) \cdot A \cdot B \cdot \bar{C} + f(1, 1, 1) \cdot A \cdot B \cdot C \end{aligned}$$





Permanentna memorija

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

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

$$\varphi_0(C) = f(0, 0, C)$$

$$\varphi_1(C) = f(0, 1, C)$$

$$\varphi_2(C) = f(1, 0, C)$$

$$\varphi_3(C) = f(1, 1, C)$$

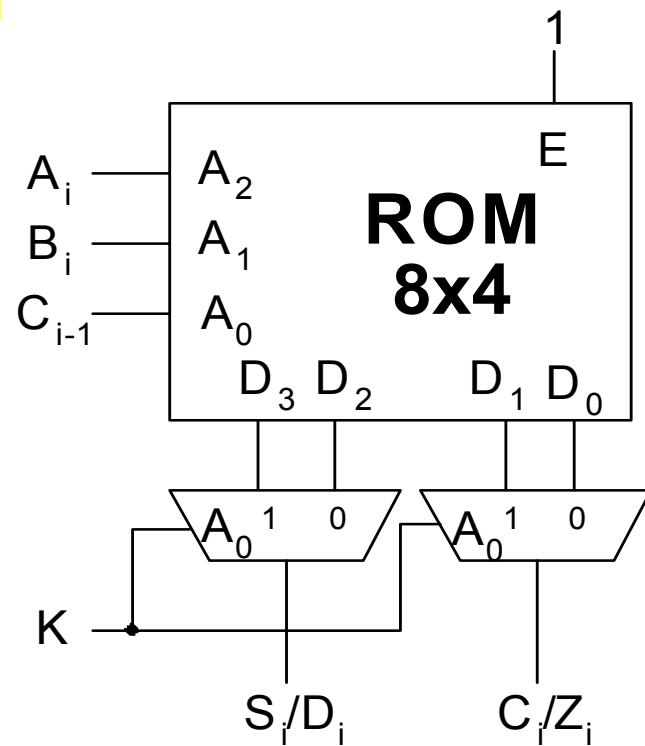
Permanentna memorija

Primjer: potpuno zbrajalo/odbijalo

- izvedba s ROMom 8x4 i 2 MUXa 2/1

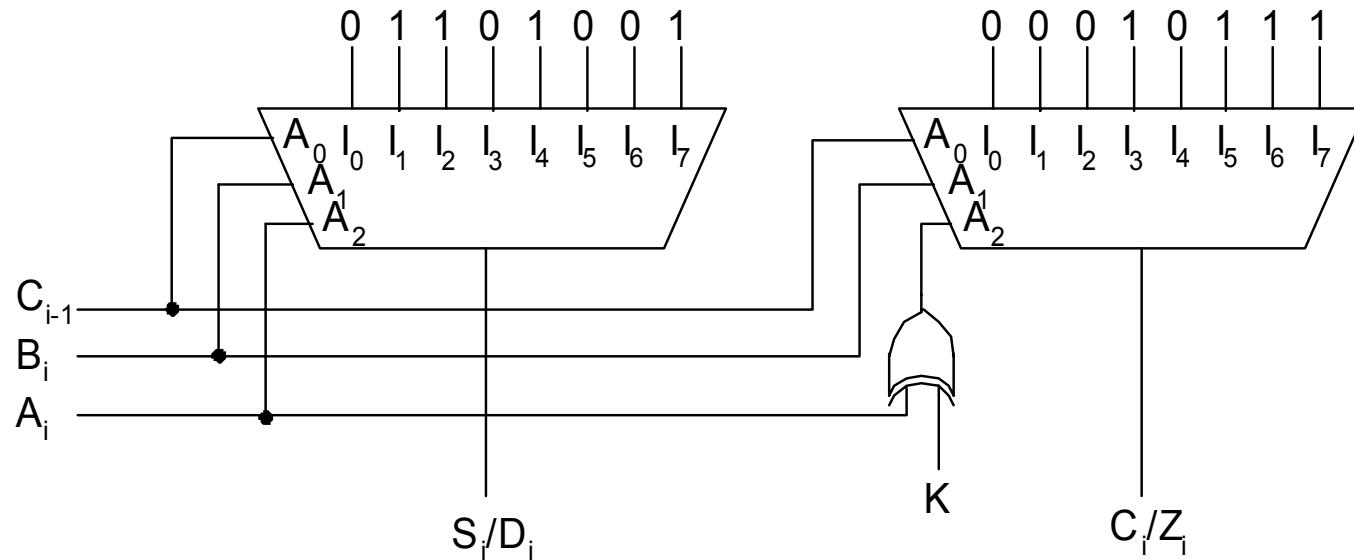
- $K=0$: zbrajalo, $K=1$: odbijalo
- uočiti: $S_i = D_i$, simetrija C_i i Z_i

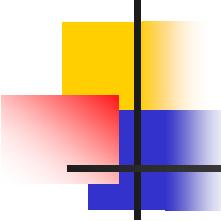
A ₂ A ₁ A ₀			D ₂ D ₀		D ₃ D ₁	
A _i	B _i	C _{i-1}	S _i	C _i	D _i	Z _i
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	1	1
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	1	0	0	1	0	0
1	1	1	1	1	1	1



Permanentna memorija

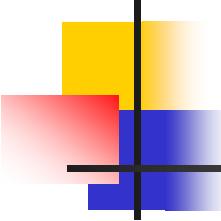
- izvedba s 2 MUXa 8/1 u funkciji ROMova
 - $K=0$: zbrajalo, $K=1$: odbijalo





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

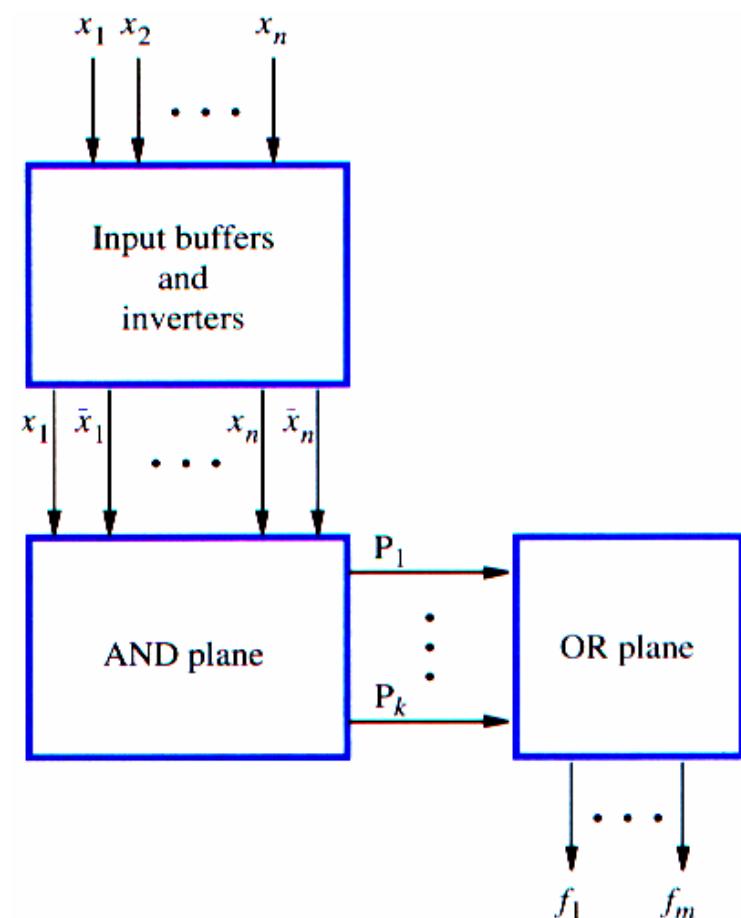


Programirljivo logičko polje

- *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 $x_i, \bar{x}_i, \forall i$
 - tipični parametri:
 $n \times k \times m = 16 \times 32 \times 8$

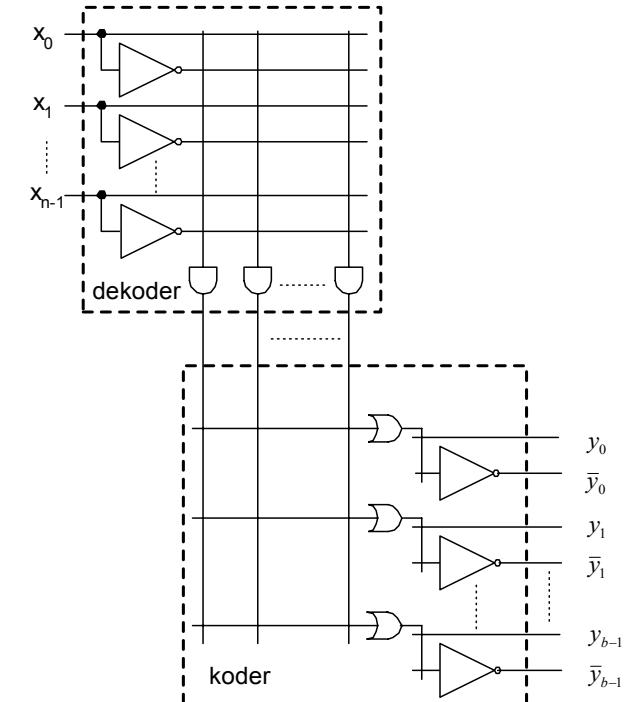
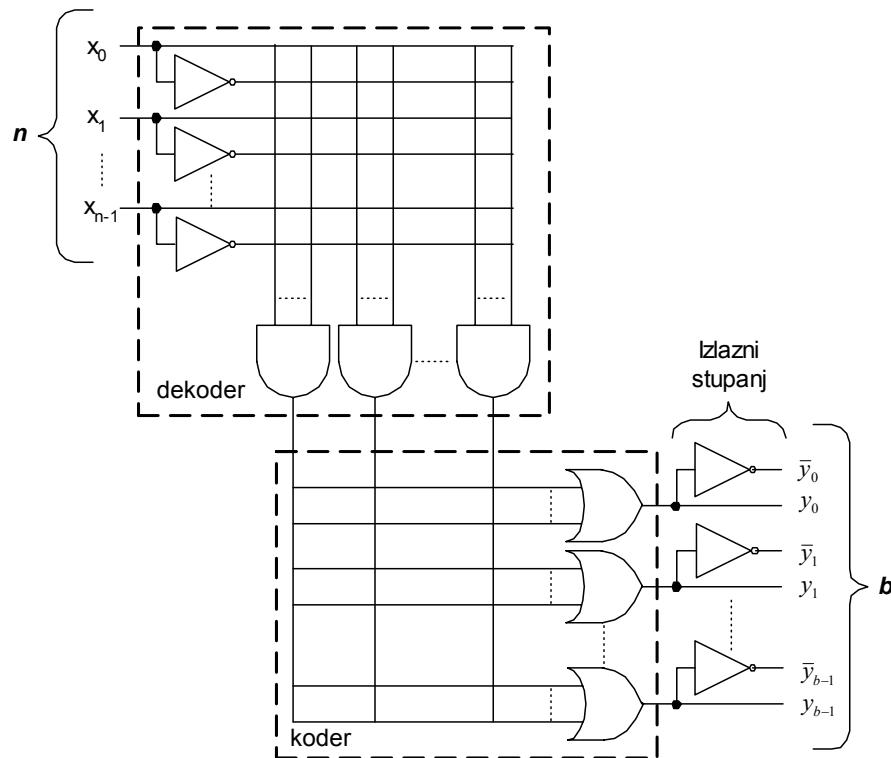


Programirljivo logičko polje

- struktura PLA:

- izlaz

~ funkcija i *komplement* (upravljeni EX-ILI)



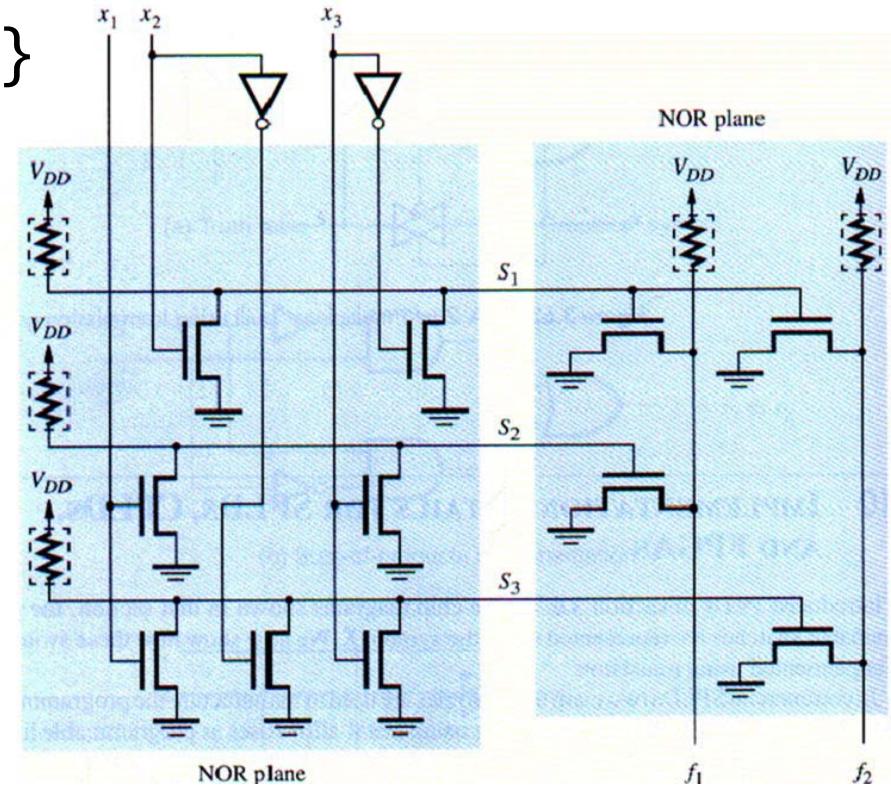
Programirljivo logičko polje

Primjer: izvedba PLA na razini tranzistora

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

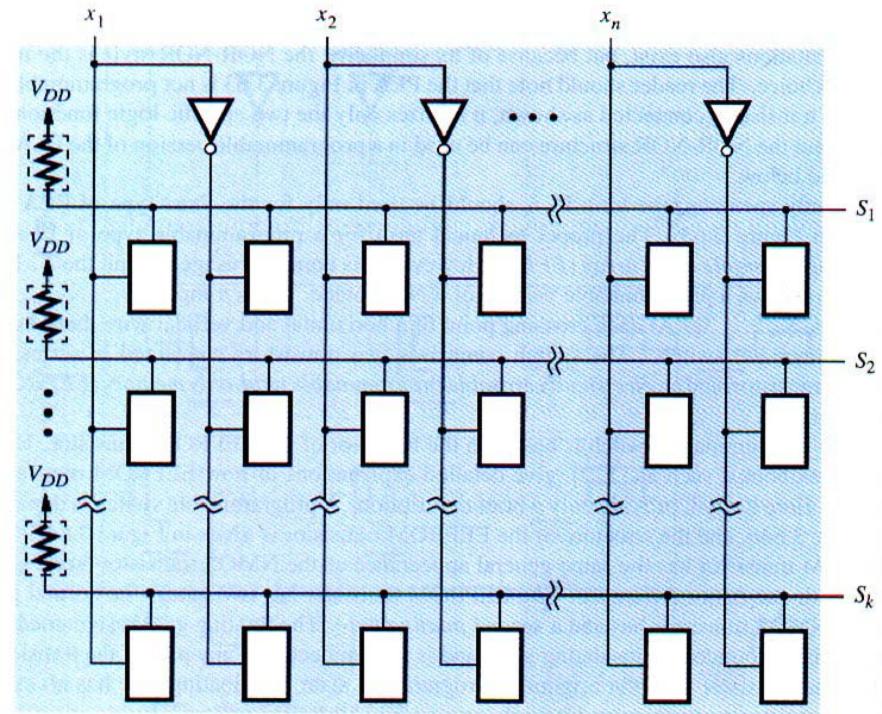
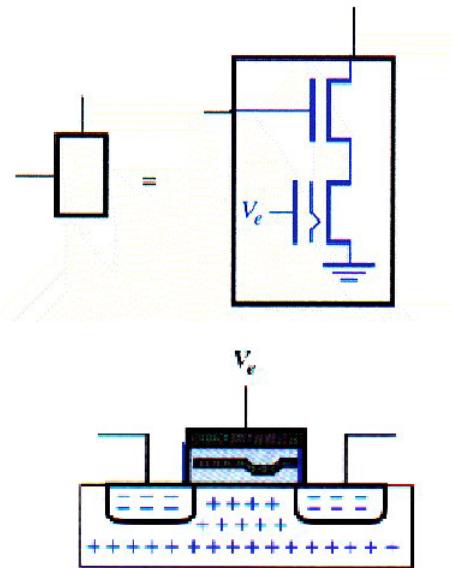
$$\begin{aligned}f_1 &= \overline{S_1 + S_2} \\&= \overline{S_1} \cdot \overline{S_2} \\&= (x_2 + \overline{x_3}) \cdot (x_1 + x_3)\end{aligned}$$

$$\begin{aligned}f_2 &= \overline{S_1 + S_3} \\&= \overline{S_1} \cdot \overline{S_3} \\&= (x_2 + \overline{x_3}) \cdot (x_1 + \overline{x_2} + x_3)\end{aligned}$$



Programirljivo logičko polje

- izvedba PLA (engl. Field-Programmable Logic Array):
 - (višekratno) programiranje "na licu mesta"
 - NMOS Tr + programirljiva sklopka
 - programirljiva sklopka \sim EEPROM



Programirljivo logičko polje

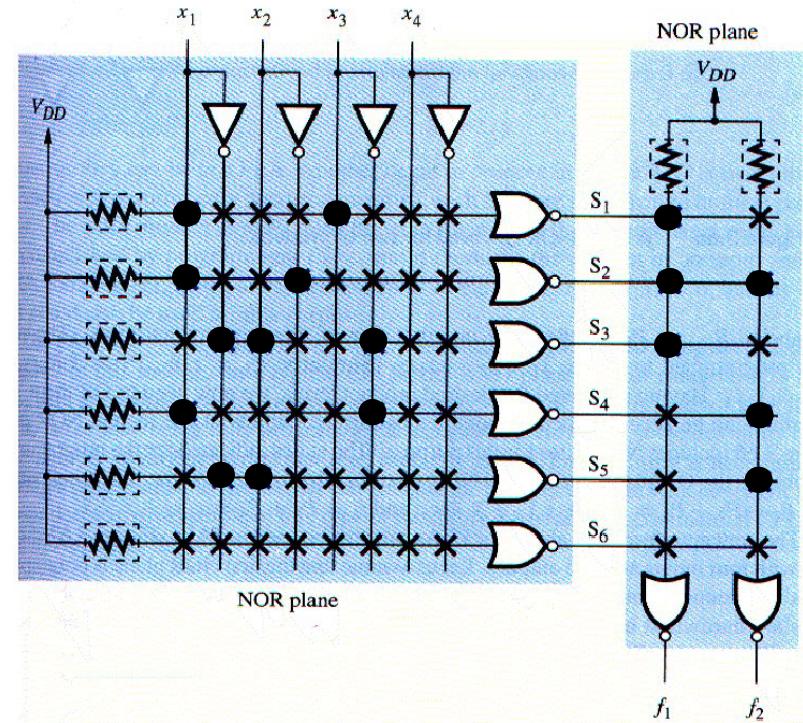
Primjer: ostvarenje funkcija s FPLA

●: neprogramirana sklopka

x: programirana, *nema* spoja

$$\begin{aligned}f_1 &= \overline{S_1 + S_2 + S_3} \\&= \overline{S_1} \cdot \overline{S_2} \cdot \overline{S_3} \\&= (x_1 + x_3) \cdot (x_1 + \overline{x_2}) \cdot (\overline{x_1} + x_2 + \overline{x_3})\end{aligned}$$

$$\begin{aligned}f_2 &= \overline{S_2 + S_4 + S_5} \\&= \overline{S_2} \cdot \overline{S_4} \cdot \overline{S_5} \\&= (x_1 + \overline{x_3}) \cdot (\overline{x_1} + x_2) \cdot (x_1 + \overline{x_2})\end{aligned}$$

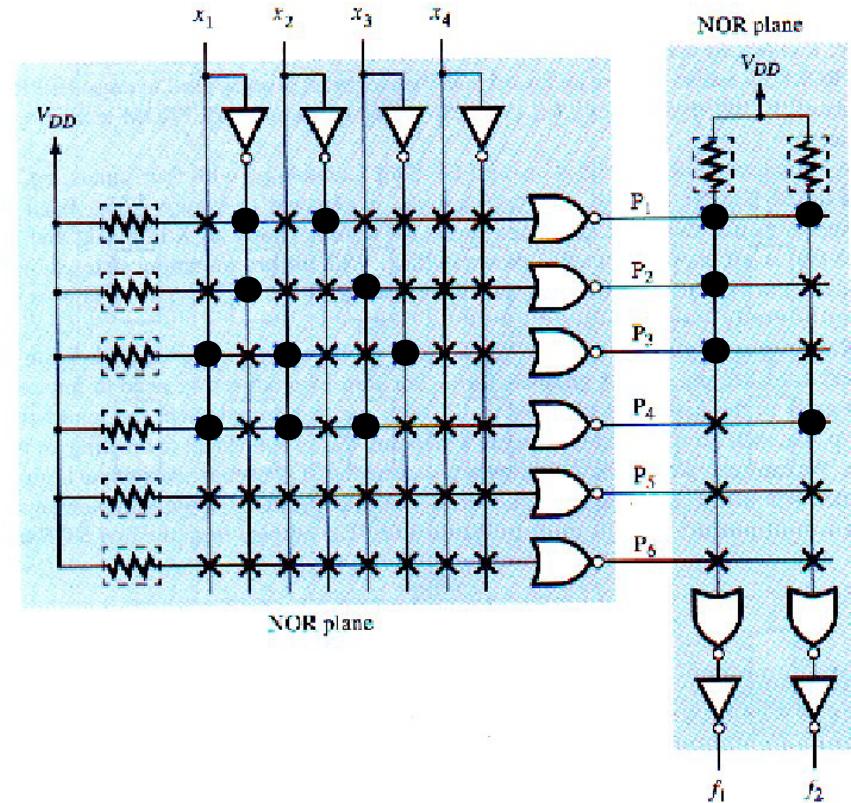


Programirljivo logičko polje

- FPLA: ostvarenje dvorazinske funkcije tipa ILI-I
 - invertirati ulaze (x_1, \dots, x_4)
 - invertirati izlaz (f_1, f_2)

$$\begin{aligned}f_1 &= P_1 + P_2 + P_3 \\&= x_1 \cdot x_2 + x_1 \cdot \overline{x_3} + \overline{x_1} \cdot \overline{x_2} \cdot x_3\end{aligned}$$

$$\begin{aligned}f_2 &= P_1 + P_4 \\&= x_1 \cdot x_2 + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}\end{aligned}$$

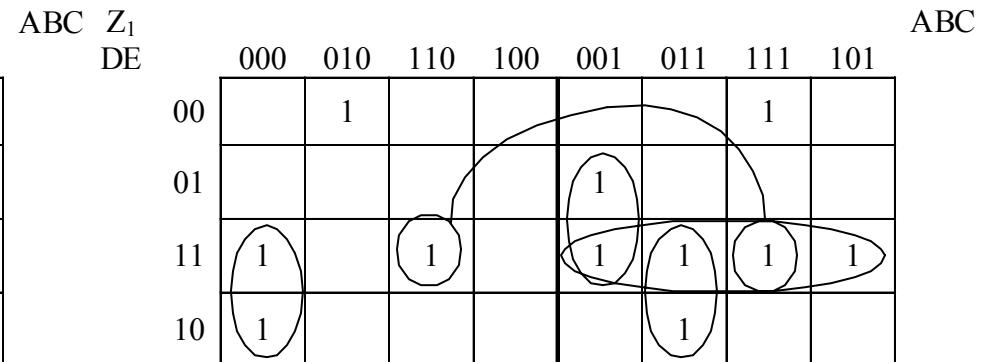
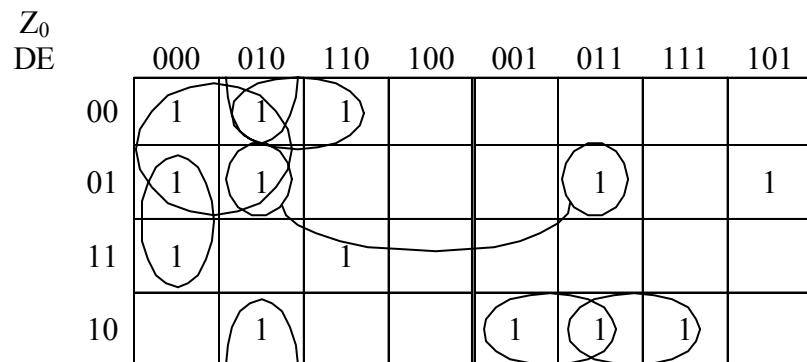


Programirljivo logičko polje

Primjer: ostvarivanje logičkih funkcija s PLA

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

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



$$\begin{aligned} Z_0 &= \overline{A}\overline{C}\overline{D} + B\overline{C}\overline{D}\overline{E} + \overline{A}\overline{B}\overline{C}\overline{E} + \overline{A}\overline{B}\overline{C}\overline{E} + \overline{A}\overline{B}\overline{D}E + \overline{A}\overline{C}\overline{D}\overline{E} + BC\overline{D}\overline{E} \\ &\quad + ABC\overline{D}E + A\overline{B}C\overline{D}E \end{aligned}$$

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

Programirljivo logičko polje

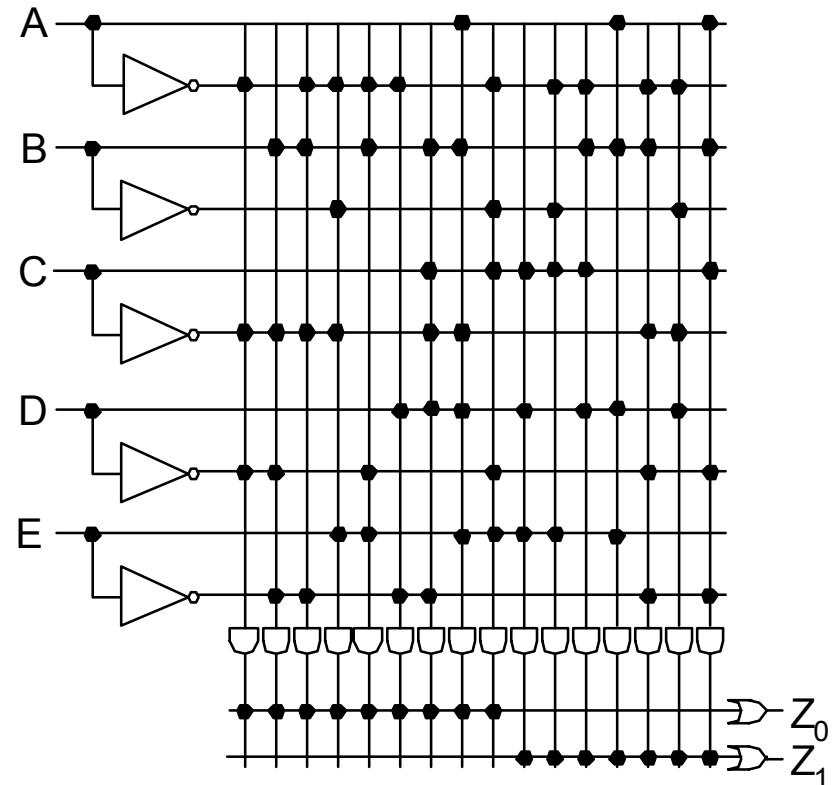
- matrični prikaz

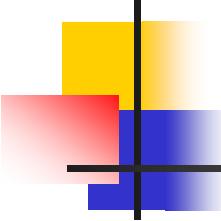
$$Z_0 = \sum(0,1,3,6,8,9,10,13,14,21,24,27,30)$$

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

$$\begin{aligned} Z_0 = & \overline{A}\overline{C}\overline{D} + B\overline{C}\overline{D}\overline{E} + \overline{A}B\overline{C}\overline{E} \\ & + \overline{A}\overline{B}\overline{C}\overline{E} + \overline{A}B\overline{D}\overline{E} + \overline{A}C\overline{D}\overline{E} \\ & + BC\overline{D} + ABC\overline{D}E + A\overline{B}C\overline{D}\overline{E} \end{aligned}$$

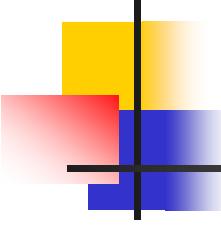
$$\begin{aligned} Z_1 = & CDE + \overline{A}\overline{B}CE + \overline{A}BCD \\ & + \overline{A}\overline{B}\overline{C}D + ABDE \\ & + \overline{A}BC\overline{D}\overline{E} + ABC\overline{D}\overline{E} \end{aligned}$$





Programirljivo logičko polje

- primjerom sklopom PLA ostvariti:
 - potpuno zbrajalo
 - potpuno zbrajalo/odbijalo
 - BCD zbrajalo
- primjenom K-tablica i metode QuineMcCluskey sklopom PLA ostvariti višeizlaznu funkciju:
 - $f_1(A, B, C, D) = \sum m(0,1,2,3,6,7)$
 $f_2(A, B, C, D) = \sum m(0,1,6,7,14,15)$
 $f_3(A, B, C, D) = \sum m(0,1,2,3,8,9)$
 - $f_1(A, B, C, D, E) = \sum m(0,1,2,3,6,7,20,21,26,27,28)$
 $f_2(A, B, C, D, E) = \sum m(0,1,6,7,14,15,16,17,19,20,24,27)$
 $f_3(A, B, C, D, E) = \sum m(0,1,2,3,8,9,16,20,26,28,30)$



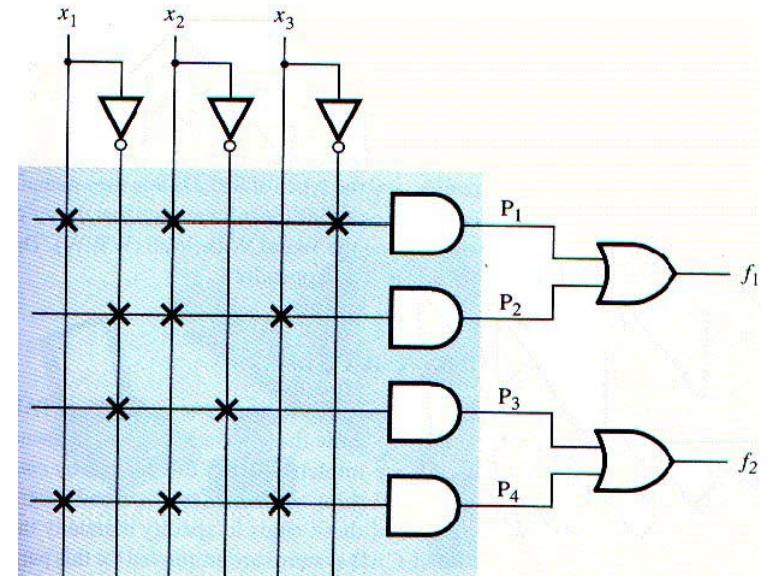
Poluprogramirljivo logičko polje

- 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)



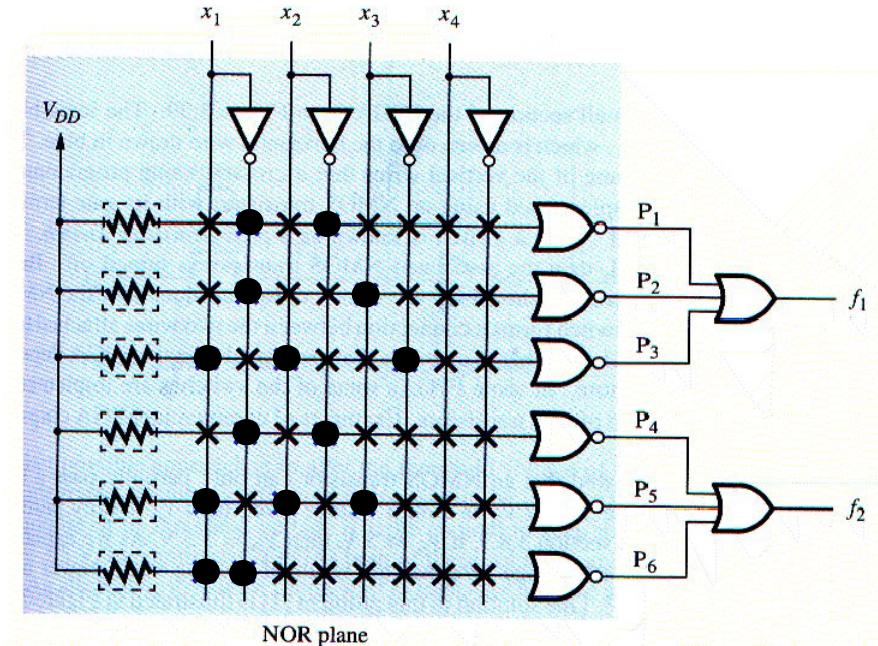
Poluprogramirljivo logičko polje

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. $P_6 = x_1 \cdot x_1 = 0$

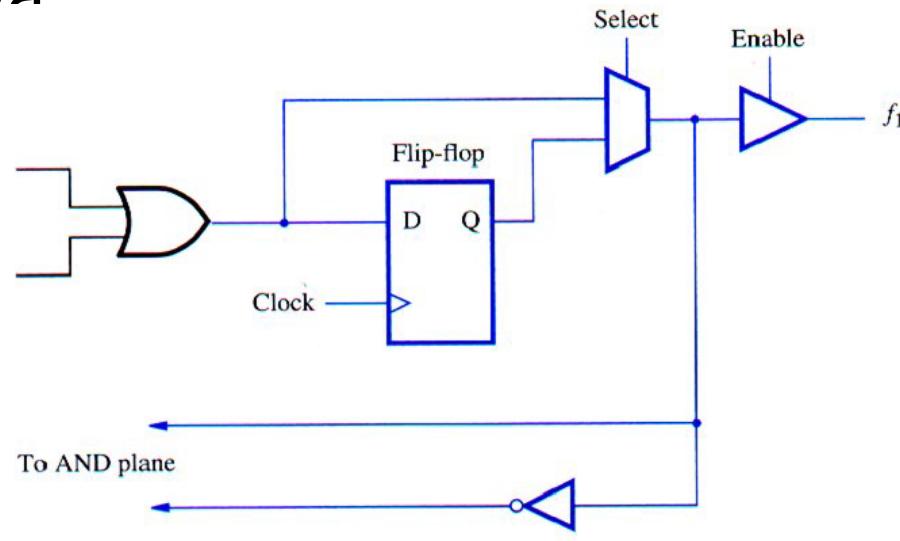
$$\begin{aligned}f_1 &= P_1 + P_2 + P_3 \\&= x_1 \cdot x_2 + x_1 \cdot \overline{x_3} + \overline{x_1} \cdot x_2 \cdot x_3\end{aligned}$$

$$\begin{aligned}f_2 &= P_4 + P_5 \\&= x_1 \cdot x_2 + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}\end{aligned}$$



Poluprogramirljivo logičko polje

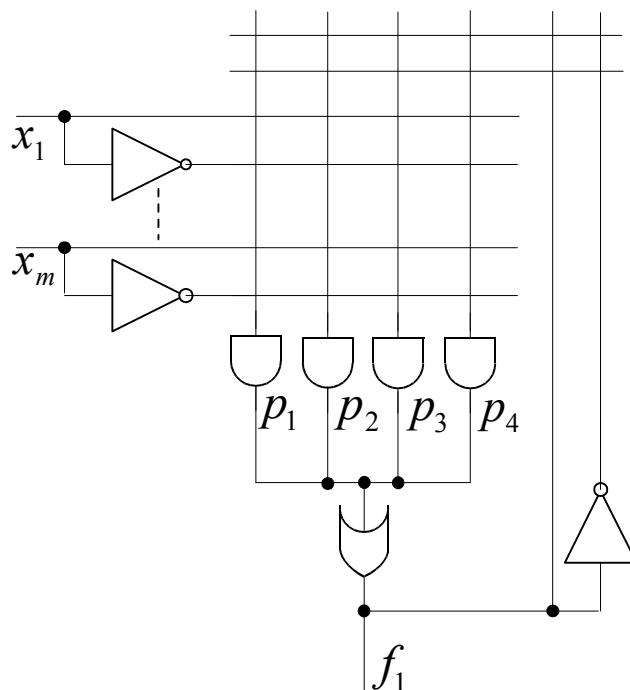
- povećanje fleksibilnosti
 - ~ *makroćelija* (engl. macrocell):
 - dodatno sklopolje na izlazu ILI sklopa
 - povratna veza na I polje
 - ~ ostvarivanja složenijih (~ višerazinskih) sklopova



Poluprogramirljivo logičko polje

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

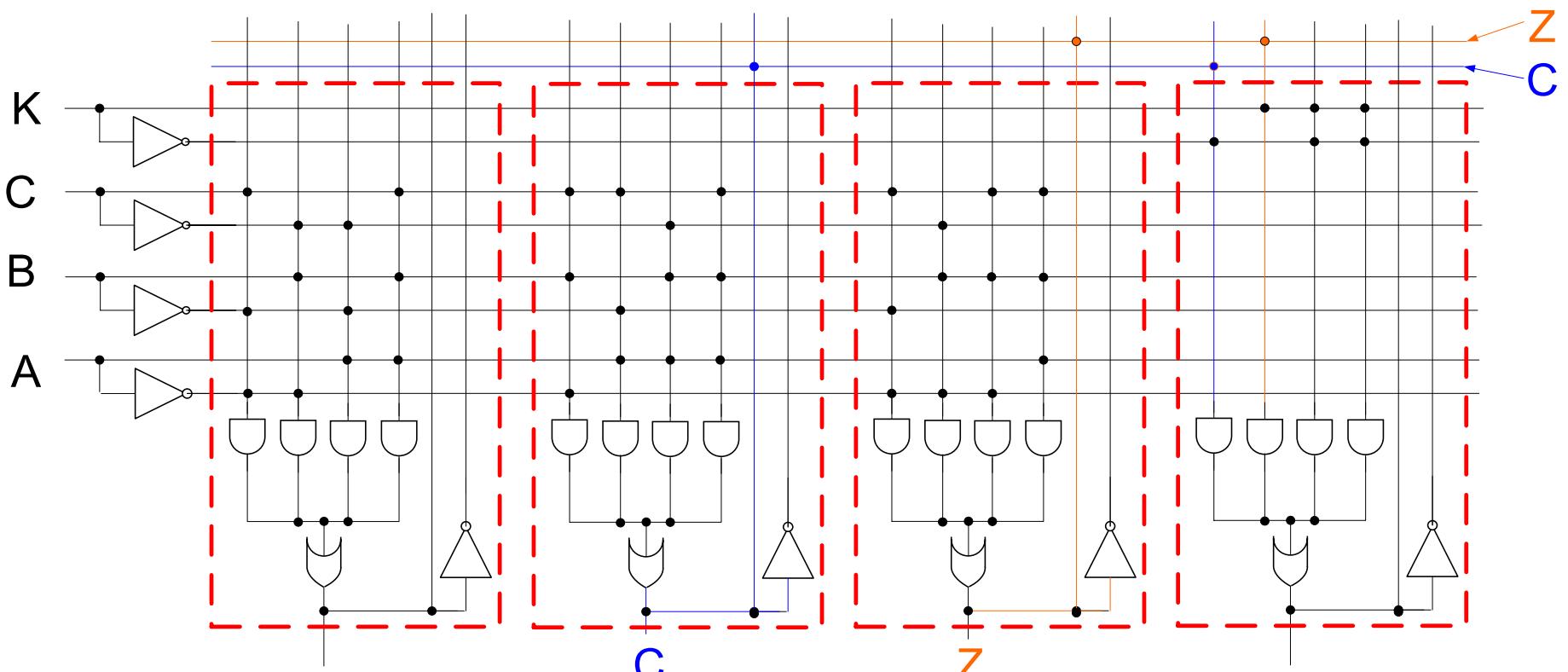


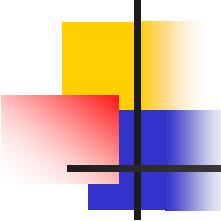
A _i	B _i	C _{i-1}	S _i	C _i	D _i	Z _i
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	1	1
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	1	0	0	1	0	0
1	1	1	1	1	1	1

Poluprogramirljivo logičko polje

- 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	0
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



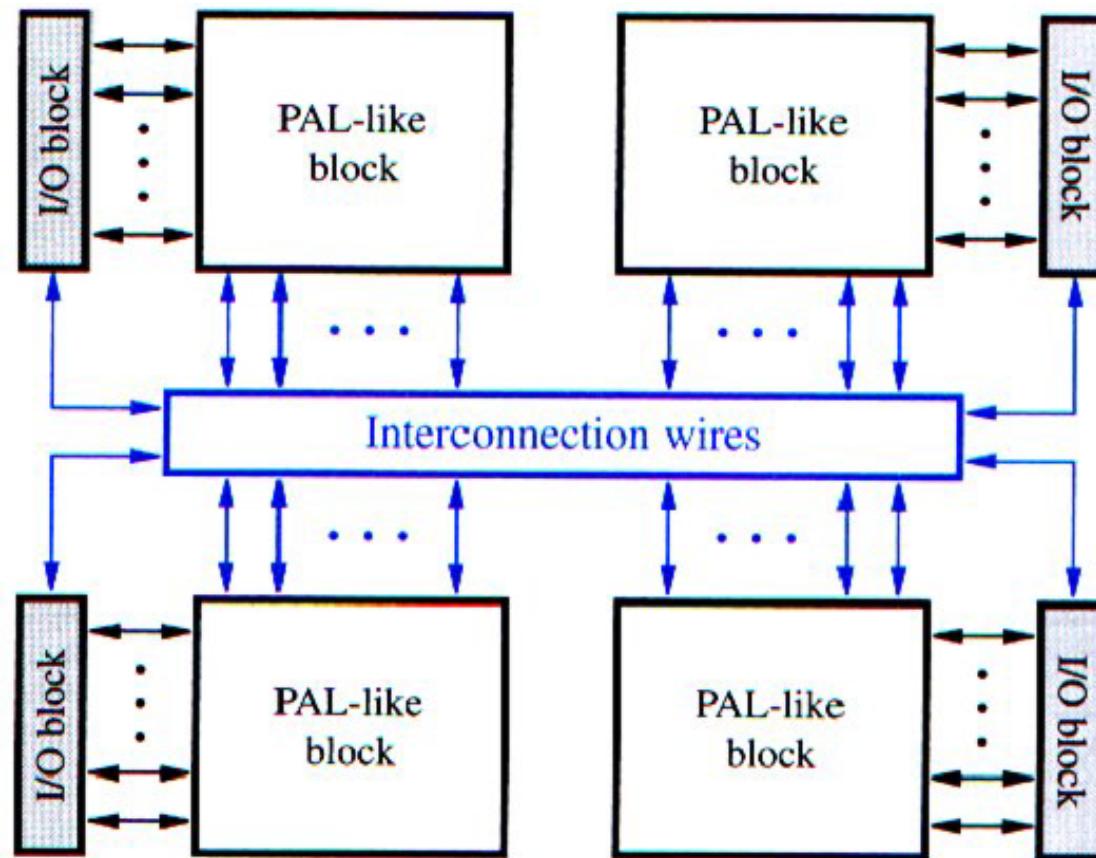


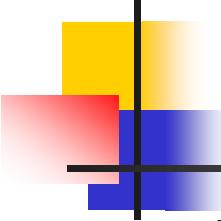
Složeni programirljivi moduli

- programirljivi moduli:
 - SPLD (= PLA, PAL)
~ skromne dimenziye sklopa
 - složeni PLD, CPLD:
 - više blokova sa sklopovljem (~ SPLD)
 - mogućnost internog povezivanja blokova
 - tipične dimenziye:
 - 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:





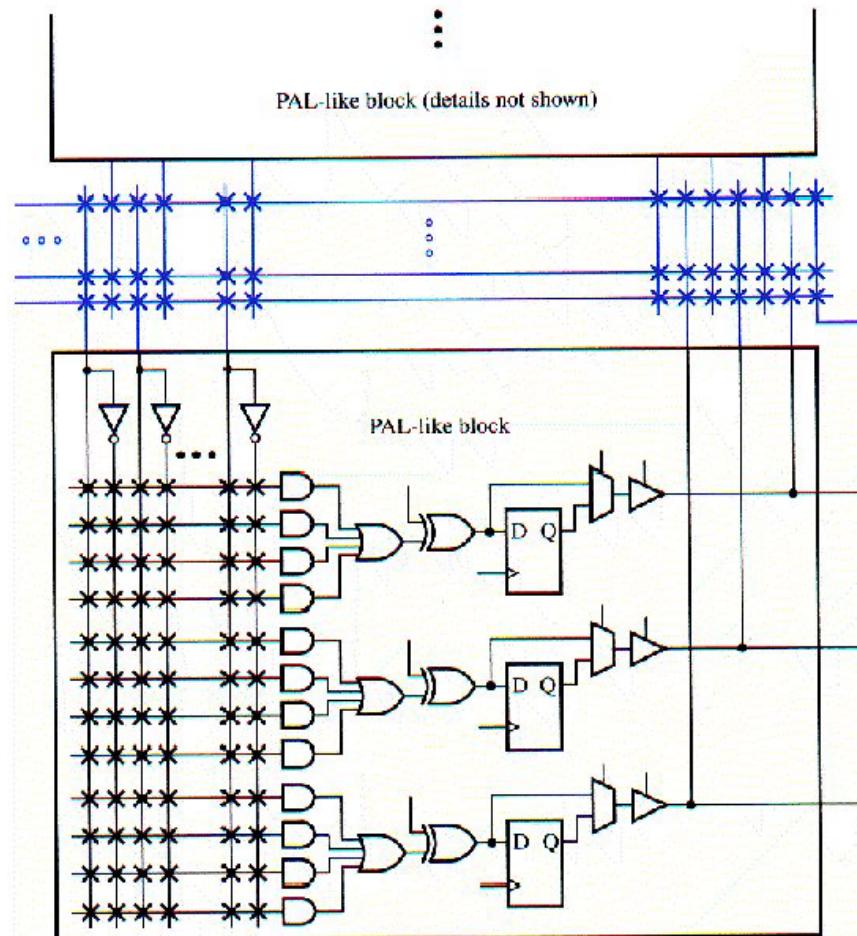
Složeni programirljivi moduli

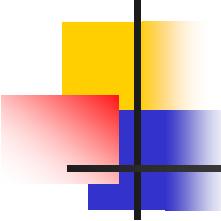
- struktura CPLD:

- *PALu slični blokovi* (engl. PAL-like blocks)
~ ostvarenje primitivnije funkcije
- *UI blokovi*
~ sučelje za svaki PALu slični blok
- *povezno ozičenje* (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

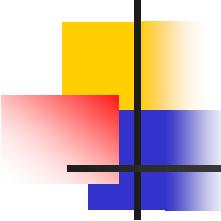
- struktura PALu sličnog bloka:





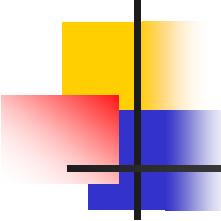
Složeni programirljivi moduli

- programiranje CPLD
 - ~ tipično *u sustavu u kojem se koriste* (engl. In-System Programming, ISP)
 - mehanički razlozi:
 - ~ IC s velikim brojem ($\sim 100 \div 200$) 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* [nonvolatile] programiranje
 - primjena CPLD u $\sim 50\%$ slučajeva korištenja PLD



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
⇒ sklop sa *što manjim* brojem IC

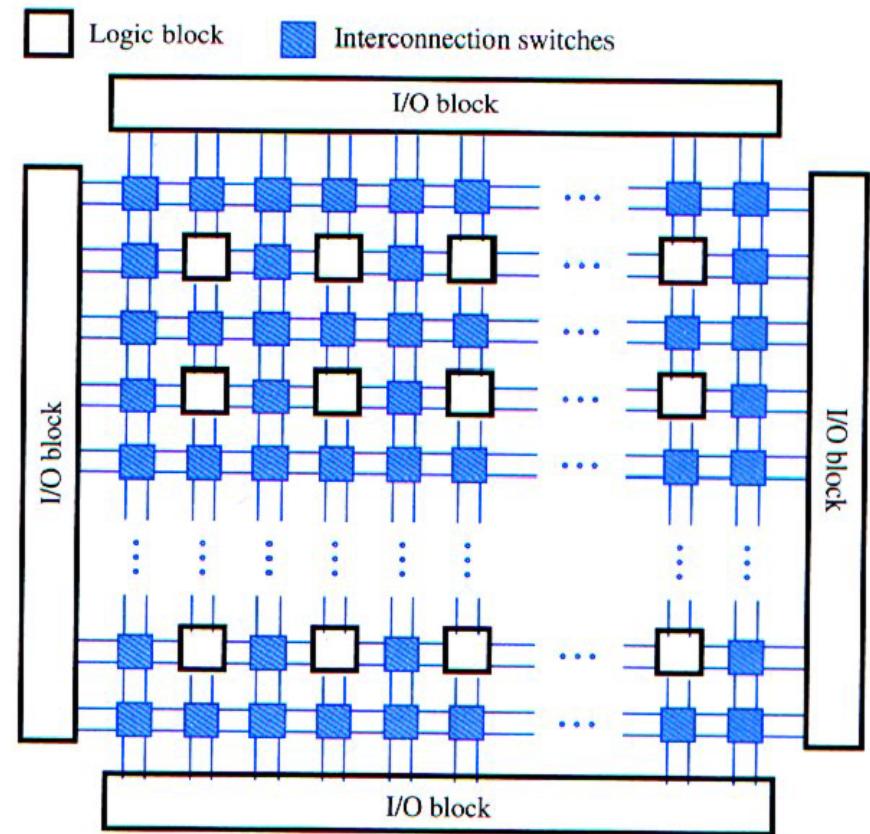
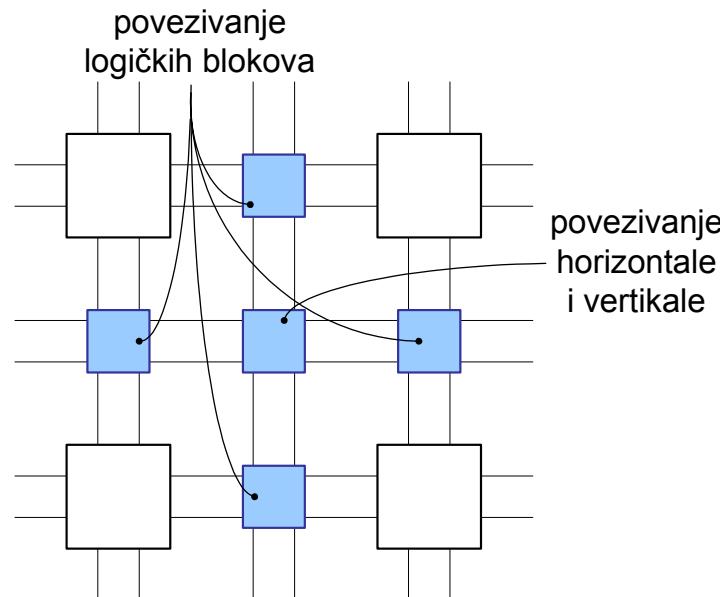


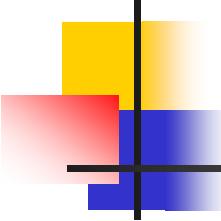
Programirljivo polje logičkih blokova

- *programirljivo polje logičkih blokova*
(engl. Field Programmable GAteway Array, FPGA)
 - PLD za ostvarivanje *relativno velikih* digitalnih sklopova (≥ 100.000 ekv. 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:
 - 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 primjerenum MUX
 - ~ izvedba multipleksorskim stablom
 - MUX u funkciji permanentne memorije!
 - tipično:
 - LUT s 4/5 ulaza → 16/32 ćelija
 - + neko dodatno sklopolje

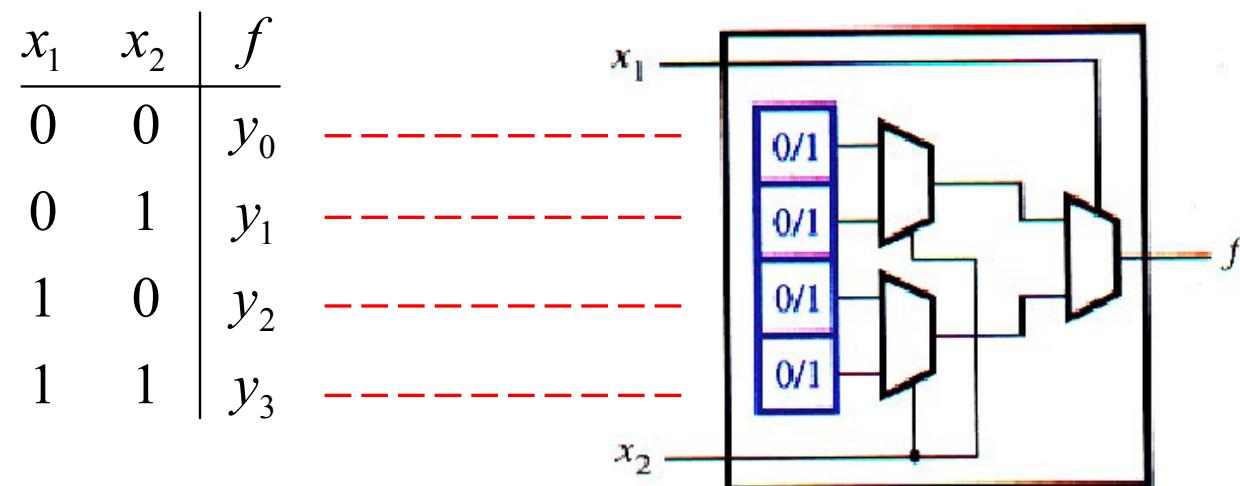
Programirljivo polje logičkih blokova

- struktura logičkog bloka s 2 ulaza

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

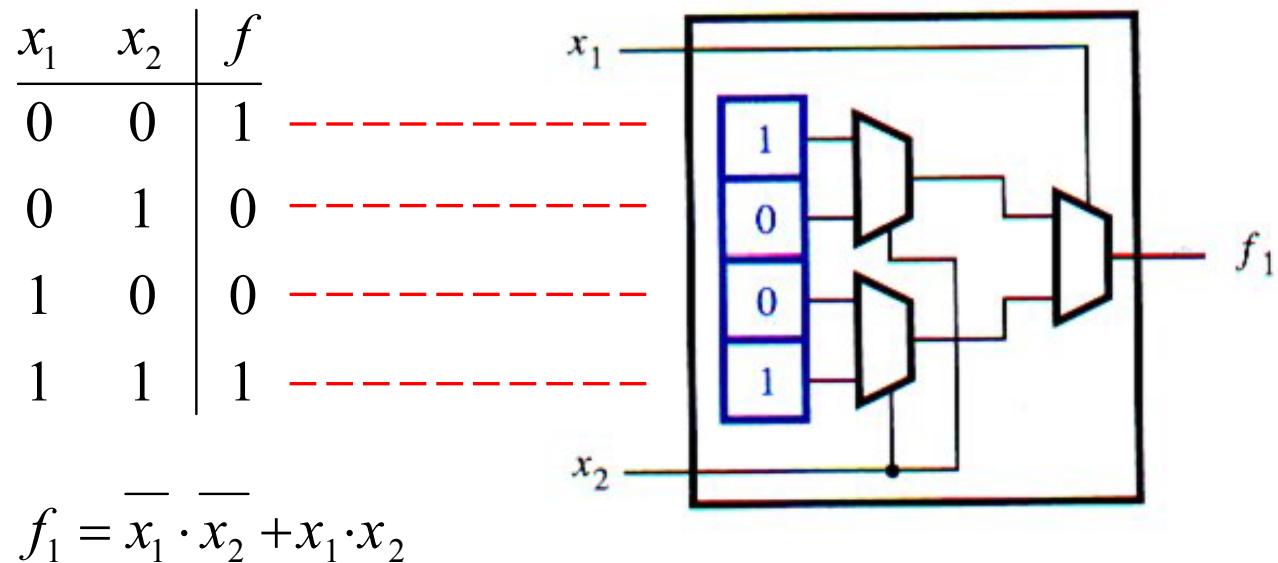
y_i : sadržaj memorijskih celija

$$\begin{aligned}f &= \overline{x_1} \cdot (\overline{x_2} \cdot y_0 + x_2 \cdot y_1) + x_1 \cdot (\overline{x_2} \cdot y_2 + x_2 \cdot y_3) \\&= \overline{x_1} \cdot \overline{x_2} \cdot y_0 + \overline{x_1} \cdot x_2 \cdot y_1 + x_1 \cdot \overline{x_2} \cdot y_2 + x_1 \cdot x_2 \cdot y_3\end{aligned}$$



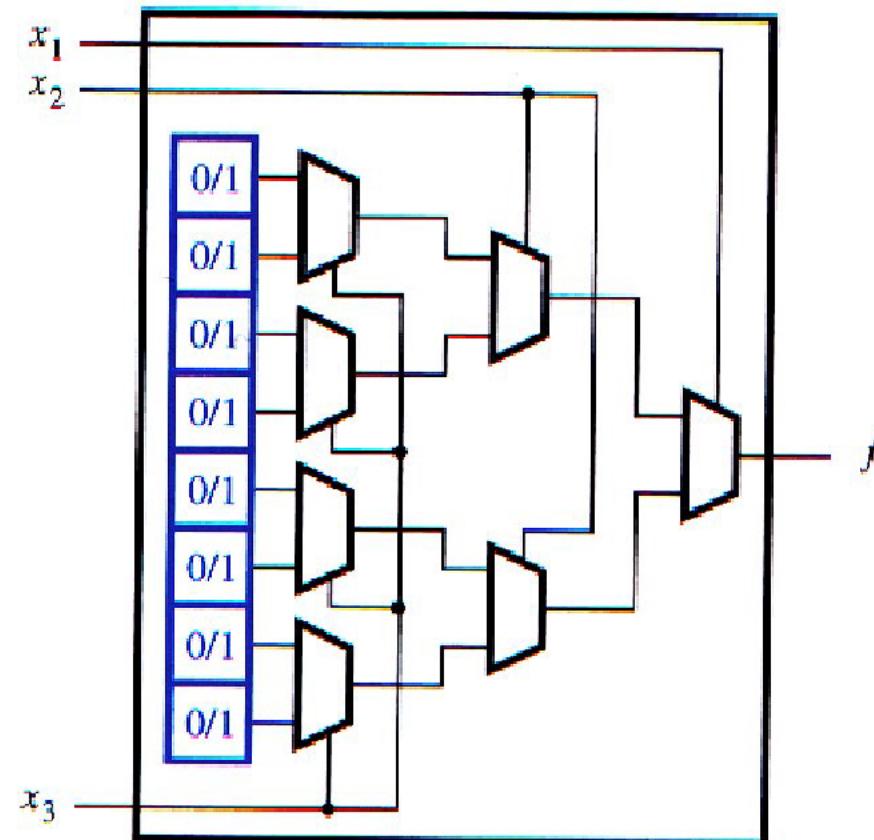
Programirljivo polje logičkih blokova

Primjer: logički blok s 2 ulaza
sadržaj ćelija $y = \langle 1, 0, 0, 1 \rangle$



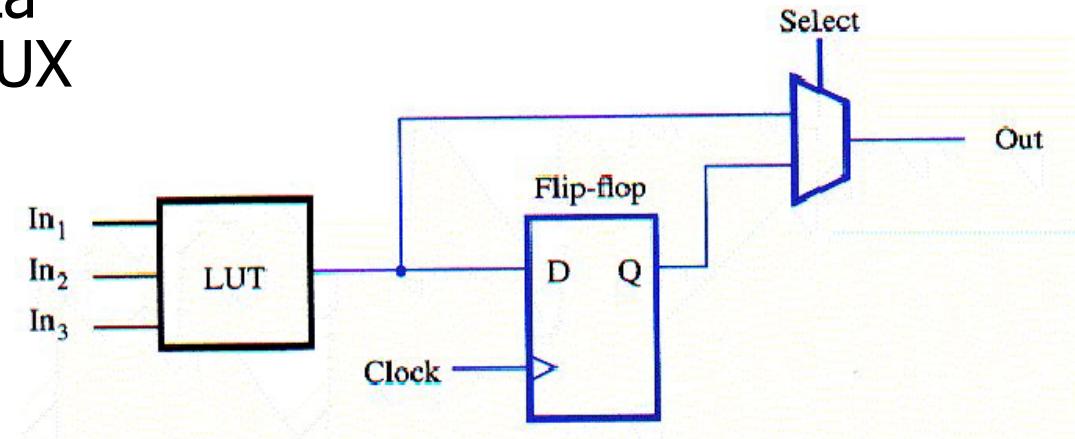
Programirljivo polje logičkih blokova

- struktura logičkog bloka s 3 ulaza
~ 8 memorijskih ćelija

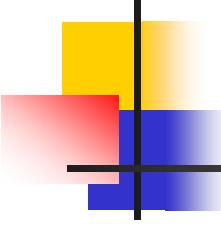


Programirljivo polje logičkih blokova

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



$$Out = \overline{Select} \cdot f + Select \cdot f^{n-1}$$
$$f = f(In_1, In_2, In_3)$$



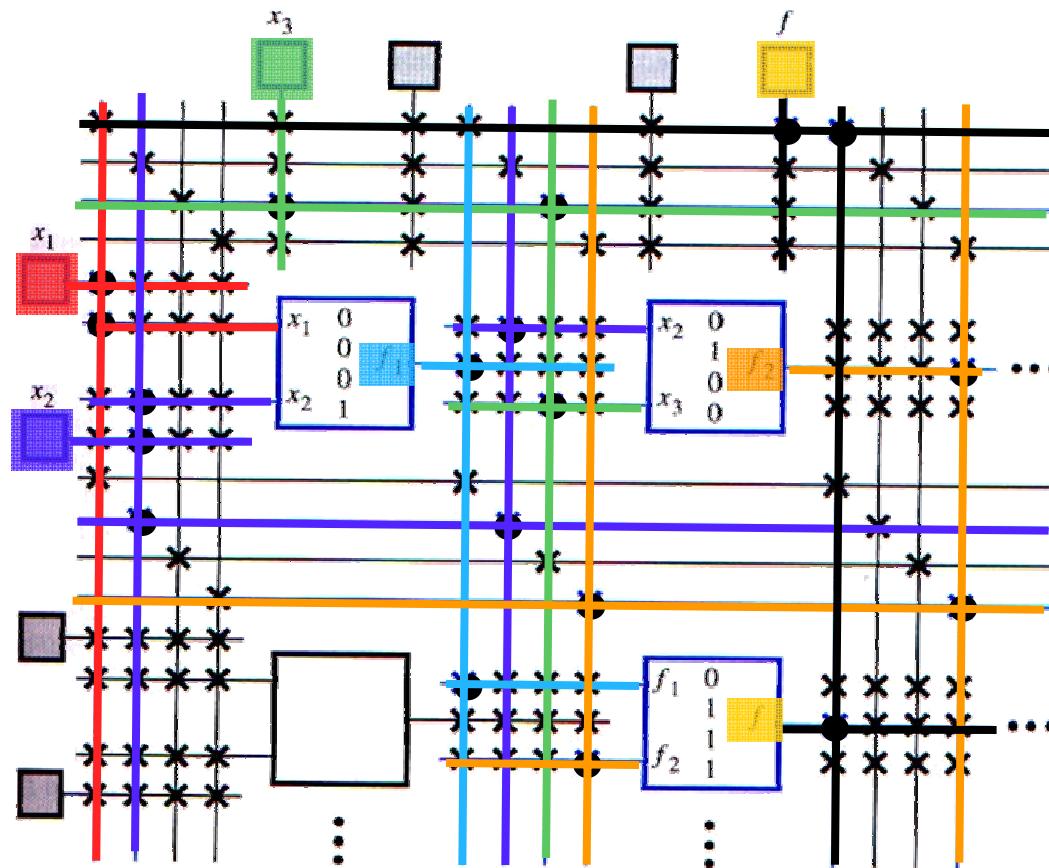
Programirljivo polje logičkih blokova

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

Programirljivo polje logičkih blokova

Primjer : (dio) programiranog FPGA

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

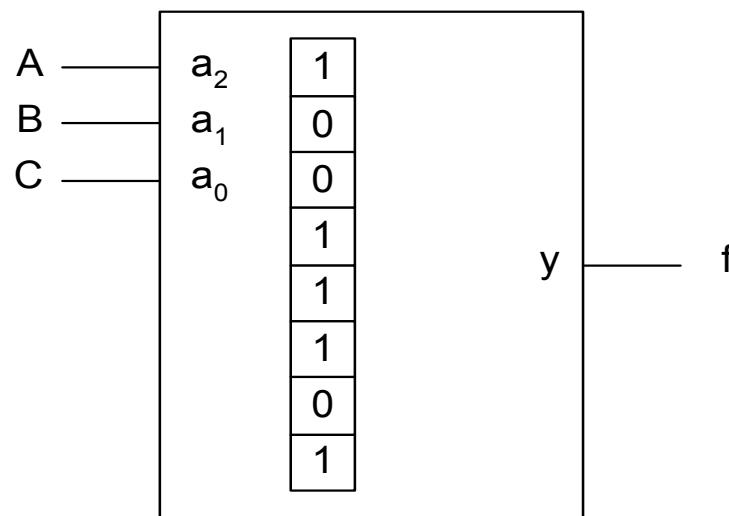


$$\begin{aligned}f_1 &= x_1 \cdot x_2 \\f_2 &= \overline{x}_2 \cdot x_3 \\f &= f_1 + f_2 \\&= x_1 \cdot x_2 + \overline{x}_2 \cdot x_3\end{aligned}$$

Programirljivo polje logičkih blokova

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

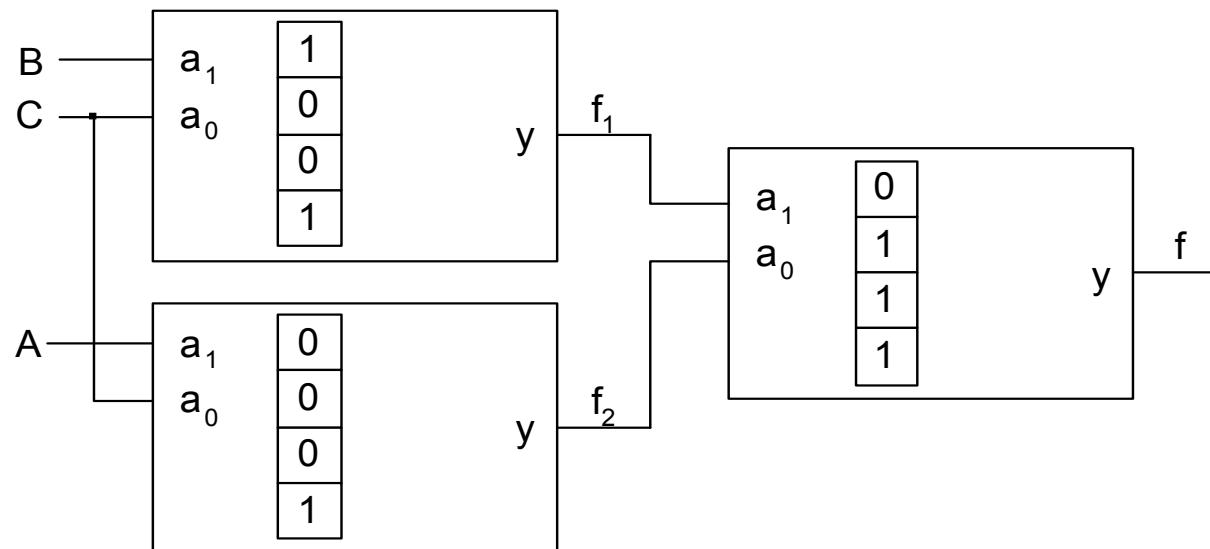
- ostvarenje LUTom s 3 ulaza
~ standardno "programiranje" multipleksora



Programirljivo polje logičkih blokova

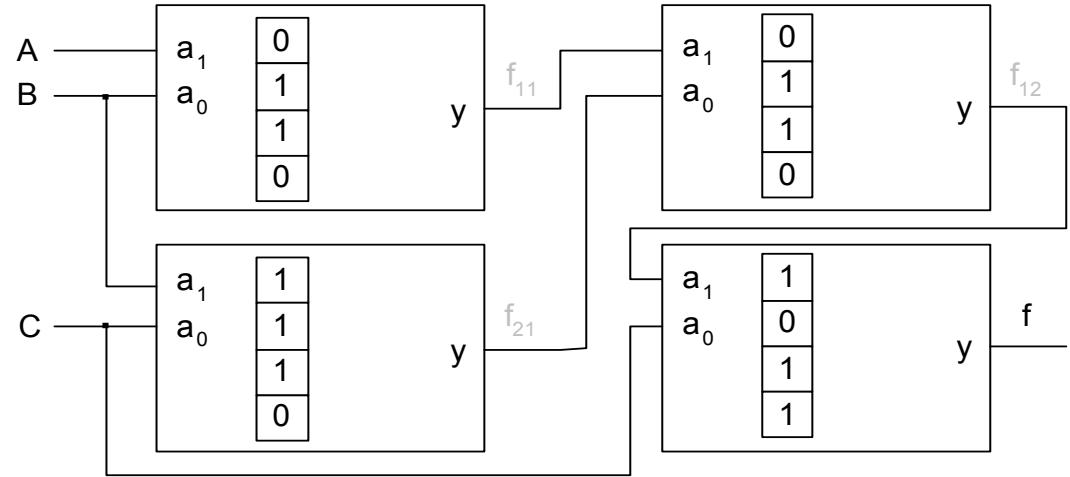
- ostvarenje LUTovima s 2 ulaza
~ kaskadiranje multipleksora

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



Programirljivo polje logičkih blokova

Primjer: $f = ?$



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

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

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

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

$$f = A + BC + \overline{BC}$$