Logičke osnove obrade podataka

Algebra logike

Algebru logike možemo definisati kao strukturu

$$\{S; \wedge, \vee, \neg\}$$

gde je $S = \{\top, \bot\}$, $\wedge$ binarna operacija konjunkcije, $\vee$ binarna operacija disjunkcije i $\neg$ unarna operacija negacije.

Formule algebre logike se nazivaju i **logički izrazi**.

Oznaka u matematici	$\bot$	$\top$	$\neg$	$\wedge$	$\vee$
Oznaka u računarstvu	0	1	$_$	$\cdot$	$+$

Neki od osnovnih zakona i identiteta Bulove algebre vezani za operacije negacije, konjunkcije i disjunkcije koji se koriste u digitalnoj logici su:

Osnovni zakoni

$A \cdot B = B \cdot A$	$A + B = B + A$	Zakon komutacije
$(A \cdot B) \cdot C = A \cdot (B \cdot C)$	$(A + B) + C = A + (B + C)$	Zakon asocijacije
$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$	$A + (B \cdot C) = (A + B) \cdot (A + C)$	Zakon distribucije
$1 \cdot A = A$	$0 + A = A$	Neutralni element
$A \cdot \overline{A} = 0$	$A + \overline{A} = 1$	Inverzni element

Identiteti i pravila

$A \cdot 0 = 0$	$A + 1 = 1$	
$A \cdot A = A$	$A + A = A$	
$\overline{\overline{A}} = A$		Dvostruka negacija
$(A \cdot B) \cdot C = A \cdot B \cdot C$	$(A + B) + C = A + B + C$	Brisanje zagrada
$\overline{A \cdot B} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A} \cdot \overline{B}$	De Morganove teoreme

Logičke funkcije

Označimo sa A_1, A_2, ..., A_n logičke promenljive. Svaka funkcija

$$f : (A_1, A_2, ..., A_n) \longrightarrow \{0, 1\}$$

se naziva **logička funkcija**.

Argument	Vrednost		Naziv	Oznaka
A	0	1		
Funkcija				
f_{10}	0	0	nula funkcija	
f_{11}	0	1	identitet	
f_{12}	1	0	negacija	$\neg$A
f_{13}	1	1	jedinična funkcija	

Tabela 1: Logičke funkcije sa jednim argumentom

Argument	Vrednost				Naziv	Oznaka
A	0	0	1	1		
B	0	1	0	1		
Funkcija						
f_{20}	0	0	0	0	nula funkcija	
f_{21}	0	0	0	1	konjunkcija	A$\wedge$B
f_{22}	0	0	1	0	negacija implikacije od A ka B	A$\wedge\neg$B = $\neg$(A$\Rightarrow$B)
f_{23}	0	0	1	1	prva projekcija	
f_{24}	0	1	0	0	negacija implikacije od B ka A	$\neg$A$\wedge$B = $\neg$(B$\Rightarrow$A)
f_{25}	0	1	0	1	druga projekcija	
f_{26}	0	1	1	0	ekskluzivna disjunkcija	(A$\wedge\neg$B)$\vee$($\neg$A$\wedge$B)
f_{27}	0	1	1	1	disjunkcija	A$\vee$B
f_{28}	1	0	0	0	Pirsova (Lukašiewičeva) funk.	A$\downarrow$B
f_{29}	1	0	0	1	ekvivalencija	(A$\Leftrightarrow$B)
f_{2A}	1	0	1	0	negacija druge projekcije	
f_{2B}	1	0	1	1	implikacija od B na A	B$\Rightarrow$A
f_{2C}	1	1	0	0	negacija prve projekcije	
f_{2D}	1	1	0	1	implikacija od A na B	A$\Rightarrow$B
f_{2E}	1	1	1	0	Šeferova funkcija	A$\uparrow$B
f_{2F}	1	1	1	1	Jedinična funkcija	

Tabela 2: Logičke funkcije sa dva argumenta

Pun sistem funkcija

Skup F={f_1, f_2, ..., f_n} funkcija se naziva **pun sistem funkcija** ako se proizvoljna funkcija algebre logike može predstaviti pomoću funkcija iz ovog skupa.

F je *minimalan* ako skup $G = F - \{f_i\}$ (za neko i) ne predstavlja pun sistem funkcija.

Neki od minimalnih punih sistema funkcija:

1. $F = \{-, \cdot\}$.
2. $F = \{-, +\}$.
3. F={ $\uparrow$ }:
   - $\overline{A}$= (A $\uparrow$ A)
   - A $\cdot$ B = (A $\uparrow$ B) $\uparrow$ (A $\uparrow$ B)
   - A + B = (A $\uparrow$ A) $\uparrow$ (B $\uparrow$ B)
4. F={ $\downarrow$ }:
   - $\overline{A}$= (A $\downarrow$ A)
   - A $\cdot$ B = (A $\downarrow$ A) $\downarrow$ (B $\downarrow$ B)
   - A + B = (A $\downarrow$ B) $\downarrow$ (A $\downarrow$ B)

Savršene normalne forme funkcija

Primer: neka se od skupa promenljivih {A, B, C} grade logički izrazi.

- Elementarne konjunkcije su: $A \cdot B$, $A \cdot C$, $\overline{A} \cdot B$, $\overline{A} \cdot C$, ...
- Elementarne disjunkcije su: $A + B$, $A + C$, $\overline{A} + B$, $\overline{A} + C$, ...
- Savršeno elementarne konjunkcije su: $A \cdot B \cdot C$, $A \cdot B \cdot \overline{C}$, $A \cdot \overline{B} \cdot C$, $A \cdot \overline{B} \cdot \overline{C}$, ...
- Savršeno elementarne disjunkcije su: $A + B + C$, $A + B + \overline{C}$, $A + \overline{B} + C$, $A + \overline{B} + \overline{C}$, ...
- Konjunktivne forme su $(A + B) \cdot (A + C)$, $(A + B) \cdot (\overline{A} + \overline{C})$,
- Disjunktivne forme su $A \cdot B + A \cdot C$, $A \cdot B + \overline{A} \cdot \overline{C}$, ...
- Savršene konjunktivne normalne forme su $(A + B + C) \cdot (A + B + \overline{C})$, $(A + B + C) \cdot (\overline{A} + \overline{B} + \overline{C})$, ...
- Savršene disjunktivne normalne forme su $(A \cdot B \cdot C) + (A \cdot B \cdot \overline{C})$, $(A \cdot B \cdot C) + (\overline{A} \cdot \overline{B} \cdot \overline{C})$, ...

Funkcija je zapisana u obliku **savršeno konjunktivne normalne forme** (skraćeno SKNF) ako je logički izraz koji je predstavlja zapisan kao savršeno konjunktivna normalna forma pri čemu svaka od savršenih elementarnih konjunkcija koja ulazi u njen sastav sadrži sve promenljive koje su argumenti funkcije.

Funkcija je zapisana u obliku **savršeno disjunktivne normalne forme** (skraćeno SDNF) ako je logički izraz koji je predstavlja zapisan kao savršeno disjunktivna normalna forma pri čemu svaka od savršenih elementarnih disjunkcija koja ulazi u njen sastav sadrži sve promenljive koje su argumenti funkcije.

Primer: neka je funkcija F zadata preko tabele istinitosnih vrednosti predstavljene u tabeli 3.

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Tabela 3: Logička funkcija sa tri promenljive

SDNF:

$F = \overline{A} \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot B \cdot \overline{C}$

SKNF:

$F = \overline{(\overline{A} \cdot \overline{B} \cdot \overline{C})} \cdot \overline{(\overline{A} \cdot \overline{B} \cdot C)} \cdot \overline{(A \cdot \overline{B} \cdot \overline{C})} \cdot \overline{(A \cdot \overline{B} \cdot C)} \cdot \overline{(A \cdot B \cdot C)}$

Primenom generalizovane De Morganove teoreme:

$\overline{(X \cdot Y \cdot Z)} = \overline{X} + \overline{Y} + \overline{Z}$

dobija se SKNF:

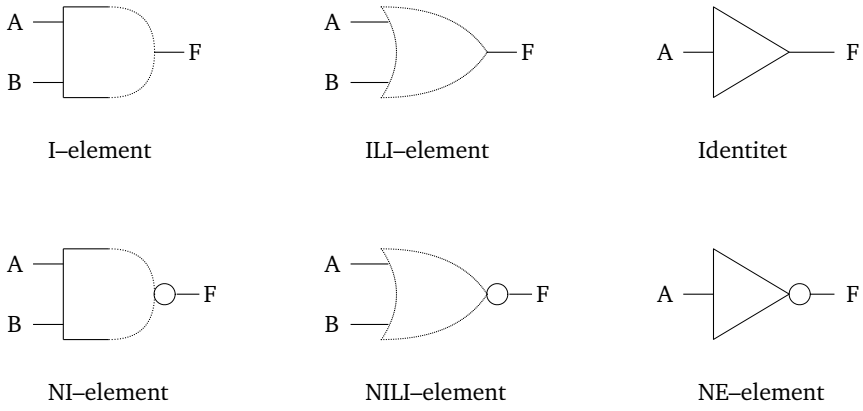
$$\begin{aligned} F &= (\overline{\overline{A}} + \overline{\overline{B}} + \overline{\overline{C}}) \cdot (\overline{\overline{A}} + \overline{\overline{B}} + \overline{C}) \cdot (\overline{\overline{A}} + \overline{\overline{B}} + \overline{C}) \cdot (\overline{\overline{A}} + \overline{\overline{B}} + \overline{C}) \cdot (\overline{\overline{A}} + \overline{B} + \overline{C}) \\ &= (A + B + C) \cdot (A + B + \overline{C}) \cdot (\overline{A} + B + C) \cdot (\overline{A} + B + \overline{C}) \cdot (\overline{A} + \overline{B} + \overline{C}) \end{aligned}$$

Logički elementi

- su fizički objekti koji implementiraju neku od funkcija algebre logike
- predstavljaju osnovne komponente svih elektronskih kola računara
- argumenti funkcija odgovaraju ulaznim veličinama logičkih elemenata, a vrednosti funkcija njihovim izlaznim veličinama.

Funkcija algebre logike	Naziv logičkog elementa	Naziv na engleskom
Negacija	NE–element	NOT–element
Konjunkcija	I–element	AND–element
Disjunkcija	ILI–element	OR–element
Šeferova funkcija	NI–element	NAND–element
Pirsova funkcija	NILI–element	NOR–element

Tabela 4: Osnovne logičke funkcije i logički elementi



Slika 1: Osnovni logički elementi



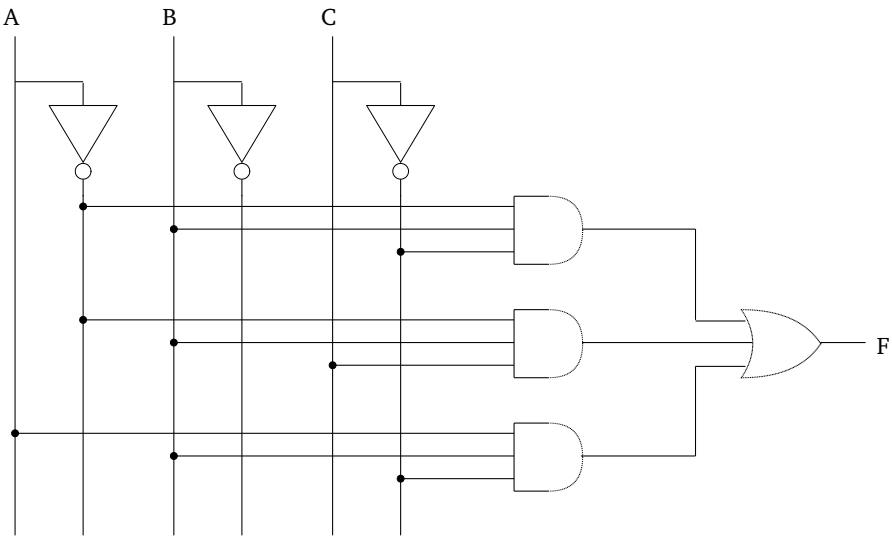
a) I–element sa četiri ulaza b) Isti signal na svim ulazima

Slika 2: Logički elementi (nastavak)

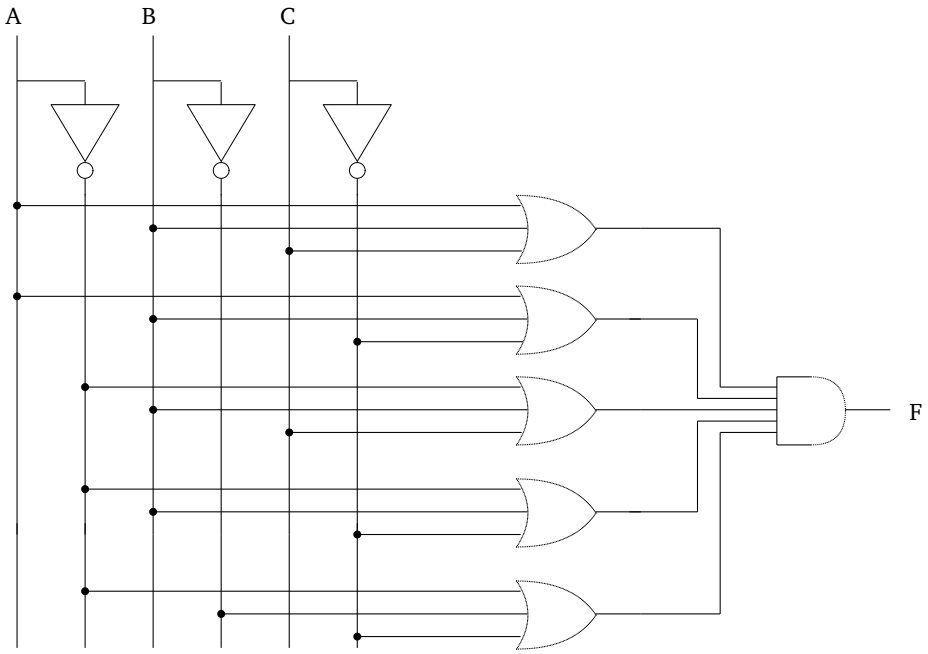
Minimizacija logičkih funkcija

Postoje tri načina za minimizaciju funkcija:

1. Algebarske transformacije
2. Karnoove (*Karnaugh*) mape
3. Tablična metoda Kvin-MekKlaskog (*Quine-McKluskey*)



Slika 3: Implementacija SDNF funkcije iz tabele 3



Slika 4: Implementacija SKNF funkcije iz tabele 3

Algebarske transformacije

Algebarske transformacije uključuju primenu osnovnih zakona i identiteta na logički izraz. Na primer, na SDNF funkcije prikazane u tabeli 3

$F = \overline{A}B\overline{C} + \overline{A}BC + AB\overline{C}$

se mogu primeniti sledeće transformacije:

1. Identitet `1 + A = 1`. Kako SDNF funkcije ima vrednost 1 to mu se vrednost neće promeniti ako dodamo izraz $\overline{A}B\overline{C}$

   $F = \overline{A}B\overline{C} + \overline{A}BC + AB\overline{C} + \overline{A}B\overline{C}$

2. Zakon distribucije:

   $F = \overline{A}B\cdot(\overline{C} + C) + (A + \overline{A})\cdot B\overline{C}$

3. Zakon o inverznom elementu ($A + \overline{A} = 1$) i identitetu ($A\cdot 1 = A$)

   $F = \overline{A}B + B\overline{C}$

4. Zakon distribucije:

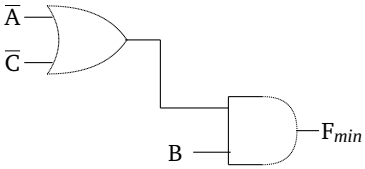
   $F_{min} = B\cdot(\overline{A} + \overline{C})$

Za prethodnu funkciju F:

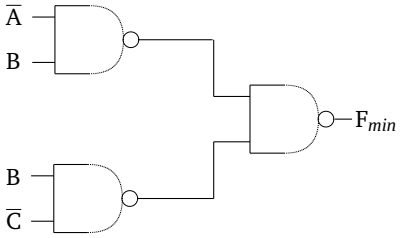
$F_{min} = B\cdot(\overline{A} + \overline{C}) = (\overline{A}B) + (B\overline{C}) = \overline{\overline{(\overline{A}B) + (B\overline{C})}}$

Primenom De Morganove teoreme dobija se oblik sastavljen od tri NI izraza

$F_{NI-min} = \overline{\overline{(\overline{A}B)}\cdot \overline{(B\overline{C})}}$



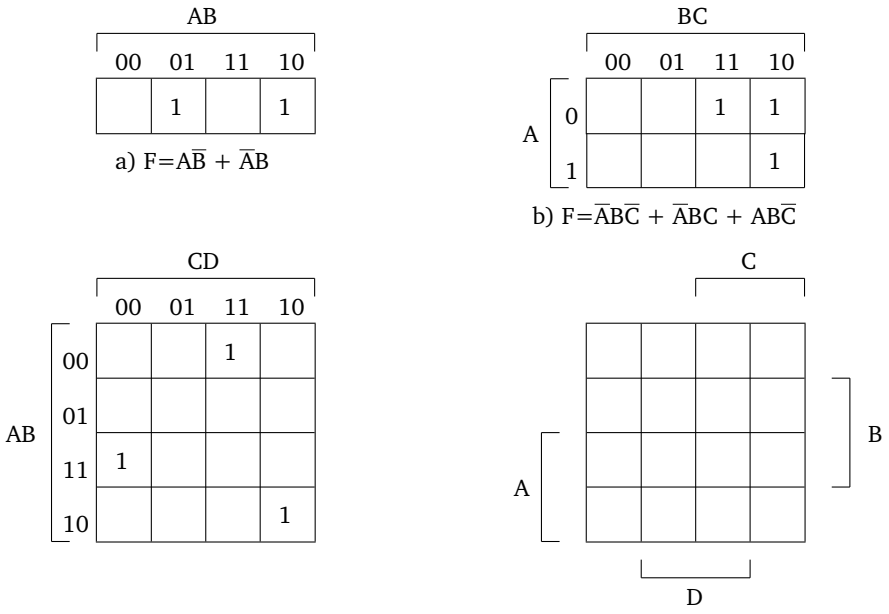
Slika 5: Implementacija funkcije F_{min}



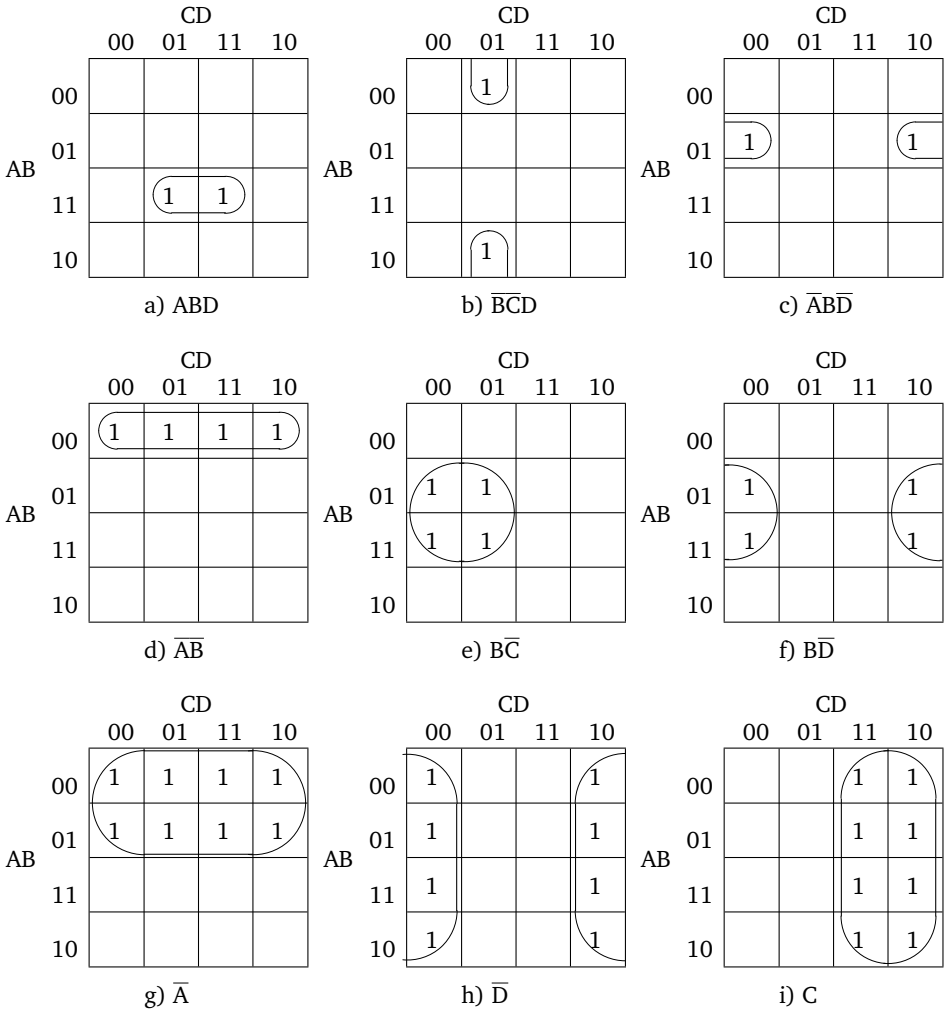
Slika 6: NI implementacija funkcije F_{min}

Karnoove mape

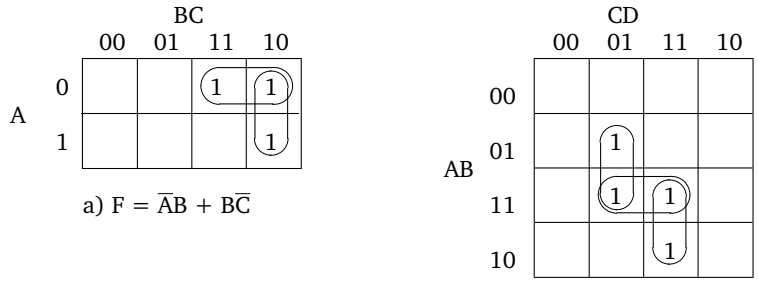
Minimizacija logičkih funkcija sa malim brojem (oko 4–6) promenljivih.
Koristi se SDNF funkcije.



Slika 7: Upotreba Karnoovih mapa za prikaz logičkih funkcija



Slika 8: Označavanje grupa kod Karnoovih mapa

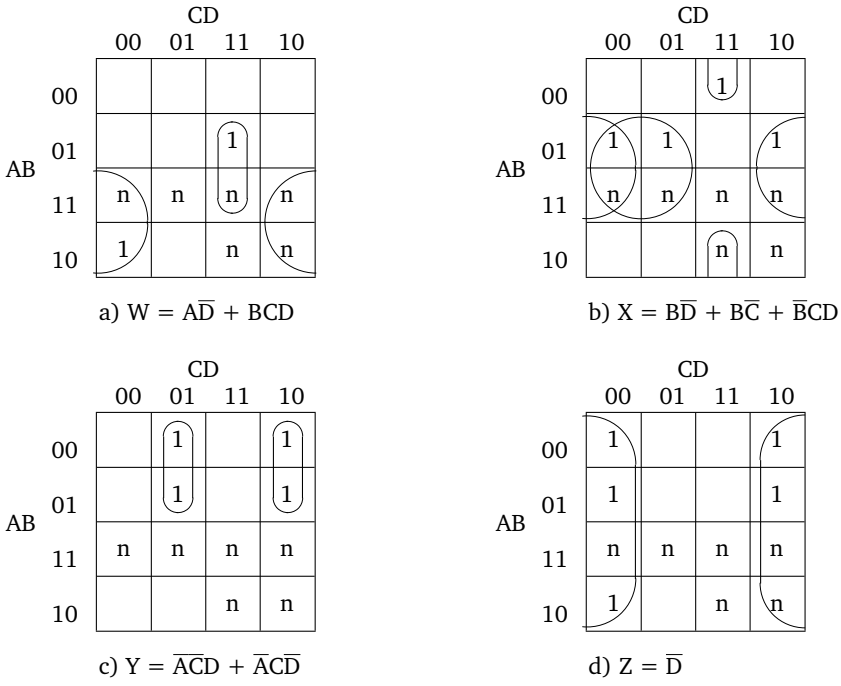


Slika 9: Preklapanje grupa kod Karnoovih mapa

Primer: nalaženje logičke funkcije koja realizuje sabiranje pakovane dekadne cifre sa jedinicom. Cifra je kodirana u kodu 8421 kao 4-bitni binarni broj. Kombinacije koje se ne koriste u kodu 8421 (1010–1111) su označene kao “nebitno“.

Ulaz					Izlaz				
Dekadna	kod 8421				Dekadna	kod 8421			
cifra	A	B	C	D	cifra	W	X	Y	Z
0	0	0	0	0	1	0	0	0	1
1	0	0	0	1	2	0	0	1	0
2	0	0	1	0	3	0	0	1	1
3	0	0	1	1	4	0	1	0	0
4	0	1	0	0	5	0	1	0	1
5	0	1	0	1	6	0	1	1	0
6	0	1	1	0	7	0	1	1	1
7	0	1	1	1	8	1	0	0	0
8	1	0	0	0	9	1	0	0	1
9	1	0	0	1	0	0	0	0	0
n	1	0	1	0		n	n	n	n
n	1	0	1	1		n	n	n	n
n	1	1	0	0		n	n	n	n
n	1	1	0	1		n	n	n	n
n	1	1	1	0		n	n	n	n
n	1	1	1	1		n	n	n	n

Tabela 5: Tabela istinitosnih vrednosti za jednocifreni BCD sabirač



Slika 10: Karnoove mape za BCD jednocifreni sabirač

Metoda Kvin-Mekklaskog

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1

A	B	C	D	F
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Tabela 6: Funkcija F na koju se primenjuje metoda Kvin–MekKlaskog

Na osnovu tabele, SDNF funkcije F je

$F = \overline{A}\overline{B}\overline{C}D + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}CD + AB\overline{C}\overline{D} + AB\overline{C}D + ABCD$

U prvoj fazi minimizacije se konstruiše tabela u kojoj svaki red odgovara jednoj od konjunkcija iz SDNF. Redovi u tabeli će biti grupisani prema broju komplementiranih promenljivih. Prva grupa sadrži term iz SDNF koji ne sadrži ni jednu komplementiranu promenljivu, druga grupa sadrži sve terme koji imaju jednu, itd.

Term	A	B	C	D	
$\overline{A}\overline{B}\overline{C}D$	0	0	0	1	✓
$\overline{A}B\overline{C}D$	0	1	0	1	✓
$\overline{A}BC\overline{D}$	0	1	1	0	✓
$AB\overline{C}\overline{D}$	1	1	0	0	✓
$\overline{A}BCD$	0	1	1	1	✓
$A\overline{B}CD$	1	0	1	1	✓
$AB\overline{C}D$	1	1	0	1	✓
$ABCD$	1	1	1	1	✓

Tabela 7: I faza primene metode Kvin–MekKlaskog na funkciju iz tabele 6 (1/3)

Sledeći korak je pronalaženje svih parova terma koji se medjusobno razlikuju samo za jednu jedinicu, tj. svih parova koji se razlikuju samo za vrednost jedne promenljive.

Kad god se termi upare označićemo (npr. sa ✓) vrstu svakog od njih, kombinovati pronadjeni par eliminacijom promenljive koja se razlikuje i dobijeni term priključiti novoj tabeli. Na primer, termi $\overline{A}BC\overline{D}$ i $\overline{A}B\overline{C}\overline{D}$ se kombinuju i daju term $\overline{A}C\overline{D}$. Postupak se nastavlja sve dok se početna tabela u potpunosti ne iscrpi. Redovi u tabeli formiranoj od novodobijenih terma su grupisani na isti način. Dobijena tabela je ulaz u naredni korak. U narednom koraku se tabela obradjuje na isti način kao i tabela u prethodnom koraku.

Term	A	B	C	D	
$\overline{A}C\overline{D}$	0		0	1	
$\overline{A}BC$	0	1	1		
$\overline{A}BD$	0	1		1	✓
$AB\overline{C}$	1	1	0		
$B\overline{C}D$		1	0	1	✓
ABD	1	1		1	✓
ACD	1		1	1	
BCD		1	1	1	✓

Tabela 8: I faza primene metode Kvin–MekKlaskog na funkciju iz tabele 6 (2/3)

U opštem slučaju, proces se nastavlja sve dok se ne dobije tabela u kojoj ne mogu da se pronadju upareni termi. U ovom primeru to je slučaj sa trećom tabelom koja sadrži samo term BD.

Term	A	B	C	D	
BD		1		1	

Tabela 9: I faza primene metode Kvin–MekKlaskog na funkciju iz tabele 6 (3/3)

	$\overline{A}\overline{B}\overline{C}D$	$\overline{A}B\overline{C}D$	$\overline{A}BC\overline{D}$	$\overline{A}BCD$	$A\overline{B}CD$	$AB\overline{C}\overline{D}$	$AB\overline{C}D$	$ABCD$
BD		X		X			X	X
$\overline{A}\overline{C}D$	(X)	[X]						
$\overline{A}BC$			(X)	[X]				
$AB\overline{C}$						(X)	[X]	
ACD					(X)			[X]

Tabela 10: II faza primene metode Kvin–MekKlaskog za funkciju F iz tabele 6

Postupak rada u drugoj fazi je sledeći:

1. Od preostalih terma konstruiše se tabela čiji redovi odgovaraju termima koji nisu eliminisani, dok kolone predstavljaju terme iz SDNF funkcije koja je bila ulaz u proces minimizacije.
2. Sa X se označi svaka pozicija u tabeli koja se nalazi na preseku vrste i kolone ako važi da term koji označava vrstu predstavlja deo terma koji označava kolonu.
3. Ako kolona sadrži samo jednu vrednost X, tada se to X zaokruži.
4. U svim redovima koji sadrže neko od zaokruženih X-eva oko svih ostalih X-eva se opiše kvadrat.
5. Ako svaka od kolona sadrži ili zaokruženo X ili X upisano u kvadrat tada je postupak odredjivanja minimalne forme funkcije završen. Minimalni oblik čine termi koji označavaju one redove koji sadrže X koje je označeno.
6. U slučaju kada postoji kolona koja ne sadrži niti zaokruženo X niti X upisano u kvadrat tada treba sprovesti dodatni postupak koji se sastoji u dodavanju redova sve dok ne budu obuhvaćene sve kolone.

U ovom slučaju, minimalni oblik funkcije F je

$F = AB\overline{C} + ACD + \overline{A}BC + \overline{A}\overline{C}D$

Kombinatorne mreže

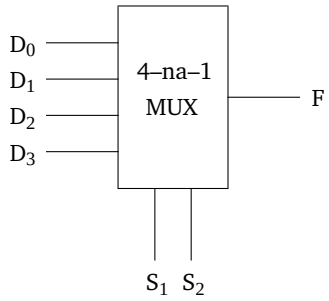
Kombinatorne mreže predstavljaju skup medjusobno povezanih logičkih elemenata čiji je izlaz u nekom vremenskom trenutku funkcija koja zavisi samo od vrednosti ulaza u tom istom vremenskom trenutku.

Kombinatorne mreže se u literaturi nazivaju još i *kombinatorna kola*. U opštem slučaju, kombinatorne mreže se sastoje od n binarnih ulaza i m binarnih izlaza i mogu biti definisane na tri načina:

1. Preko tabele istinitosnih vrednosti koja sadrži kombinacije za svaku od 2^n mogućih ulaznih vrednosti.
2. Preko povezanog skupa grafičkih simbola.
3. Preko logičkih funkcija koje izražavaju vezu izmedju ulaznih i izlaznih promenljivih (signala).

Multipleksori

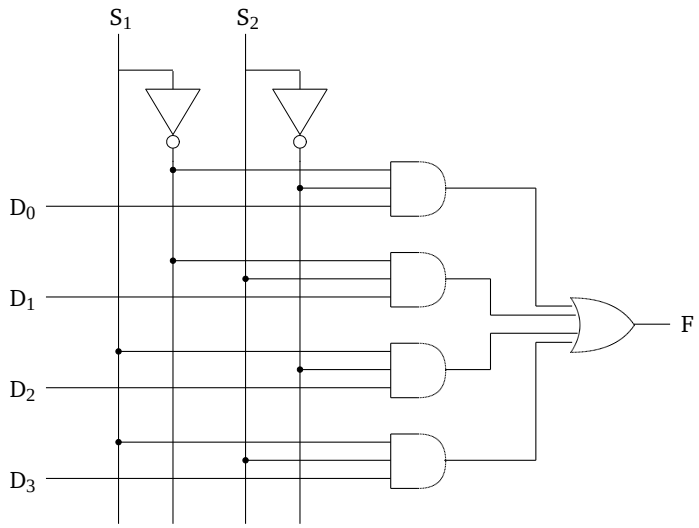
Multipleksori su kombinatorne mreže koje povezuju više ulaznih signala u jedan izlazni signal. U bilo kom trenutku bira se tačno jedan od ulaznih signala i propušta na izlaz.



Slika 11: Predstavljanje multipleksora 4–na–1

S_1	S_2	F
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Tabela 11: Istinitosna tabela za multipleksor 4–na–1



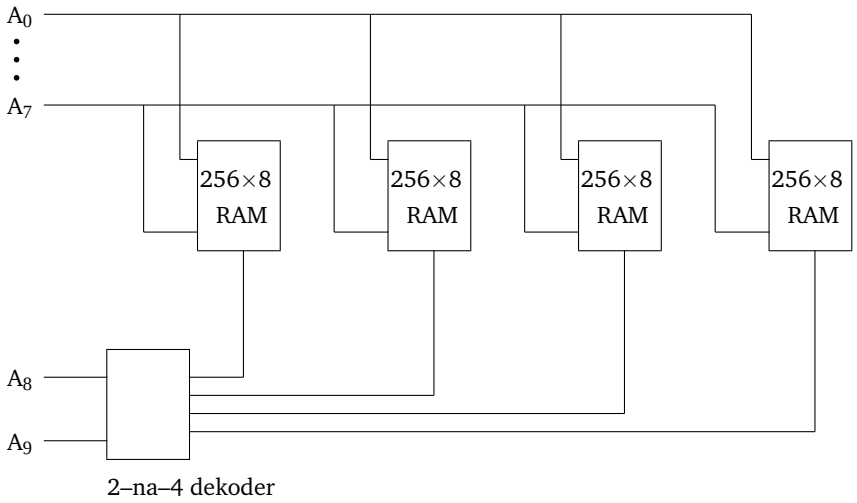
Slika 12: Implementacija multipleksora 4–na–1

Dekoderi

Dekoderi predstavljaju kombinatorne mreže sa više izlaza pri čemu je istovremeno aktivan samo jedan izlaz koji prepoznaje (dekodira) vrednost signala na ulazu. U opštem slučaju, dekoder ima n ulaza i najviše 2^n izlaza.

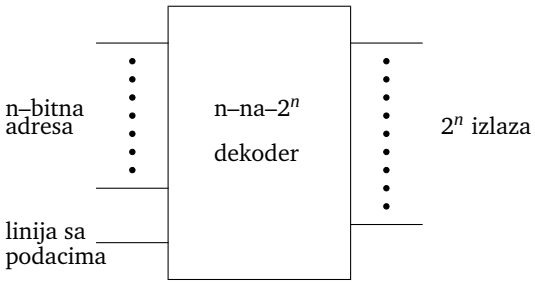
Primer 1: dekodiranje adresa. Pretpostavimo da želimo da konstruišemo memoriju veličine 1KB korišćenjem četiri 256×8-bitnih RAM čipova.

Adrese	Čip
0000–00FF	0
0100–01FF	1
0200–02FF	2
0300–03FF	3



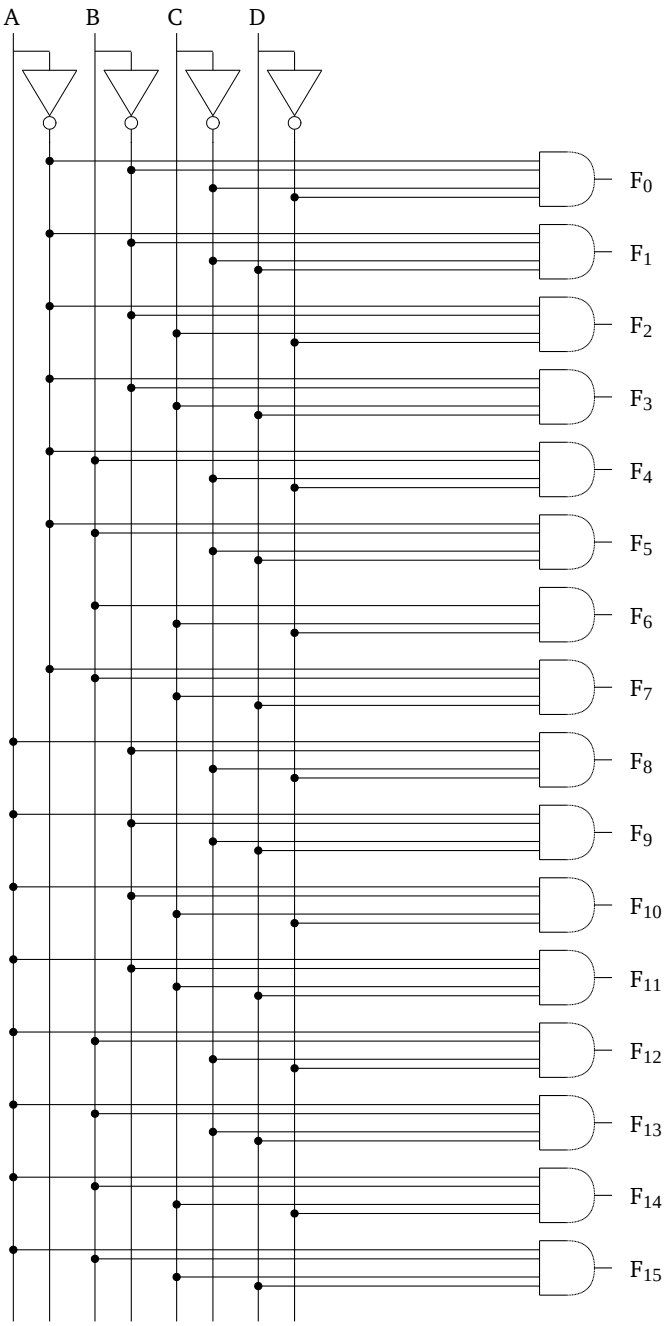
Slika 13: Dekoder adresa

Primer 2: dekoder kao demultipleksor. Ako se u dekoder uvedu dodatne ulazne linije on tada može da posluži kao demultipleksor. Demultipleksor vrši inverznu funkciju multipleksora, tj. razvodi jedan ulaz na više izlaza.



Slika 14: Implementacija demultipleksora pomoću dekodera

Primer 3: dekodiranje BCD cifre. Pretpostavka je da želimo da konstruišemo dekoder koji koji vrši dekodiranje (prevodjenje) binarno kodirane dekadne cifre iz koda 8421 u dekadnu vrednost. Dekoder će imati 4 ulaza i 16 izlaza.



Slika 15: Dekoder dekadnih cifara iz BCD zapisa

Ulaz				Izlaz															
				Dekadne cigre										Greška					
A	B	C	D	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

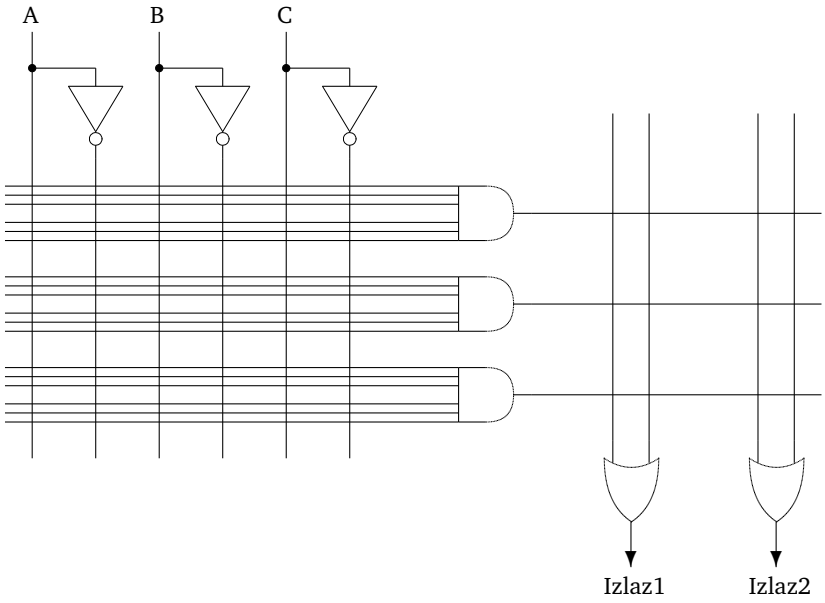
$F_0 = \overline{A}\overline{B}\overline{C}\overline{D}$
$F_1 = \overline{A}\overline{B}\overline{C}D$
$F_2 = \overline{A}\overline{B}C\overline{D}$
$F_3 = \overline{A}\overline{B}CD$
$F_4 = \overline{A}B\overline{C}\overline{D}$
$F_5 = \overline{A}B\overline{C}D$
$F_6 = \overline{A}BC\overline{D}$
$F_7 = \overline{A}BCD$
$F_8 = A\overline{B}\overline{C}\overline{D}$
$F_9 = A\overline{B}\overline{C}D$
$F_{10} = A\overline{B}C\overline{D}$
$F_{11} = A\overline{B}CD$
$F_{12} = AB\overline{C}\overline{D}$
$F_{13} = AB\overline{C}D$
$F_{16} = ABC\overline{D}$
$F_{15} = ABCD$

Tabela 12: Zahtevi za dekoder dekadnih cifara iz BCD zapisa

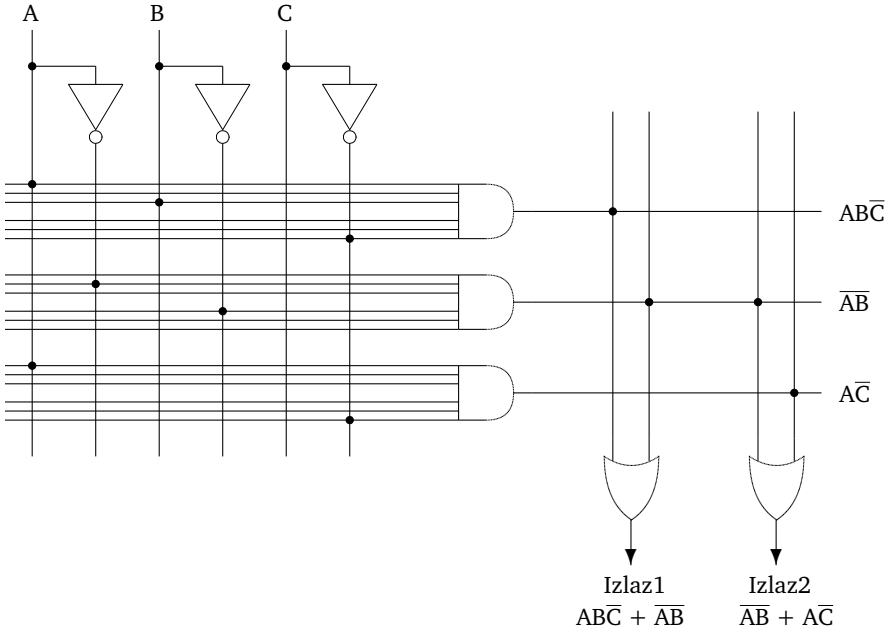
Programibilni niz logičkih elemenata

Programibilni niz logičkih elemenata (eng. *PLA, Programmable Logic Array*) je čip opšte namene koji jednostavno može da se prilagodi raznim specifičnim potrebama.

Upotreba PLA je zasnovana na činjenici da se svaka logička funkcija može izraziti u obliku SDNF.



Slika 16: Izgled PLA sa tri ulaza i dva izlaza



Slika 17: Dizajniranje veza u PLA sa tri ulaza i dva izlaza

Sabirači

Sabirači su kombinatorne mreže koje se koriste pri realizaciji sabiranja. Sabirači mogu biti *binarni* i *dekadni*.

Binarni **polusabirač** je kombinatorna mreža koja vrši sabiranje jednocifrenih binarnih brojeva.

A	B	C	P
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

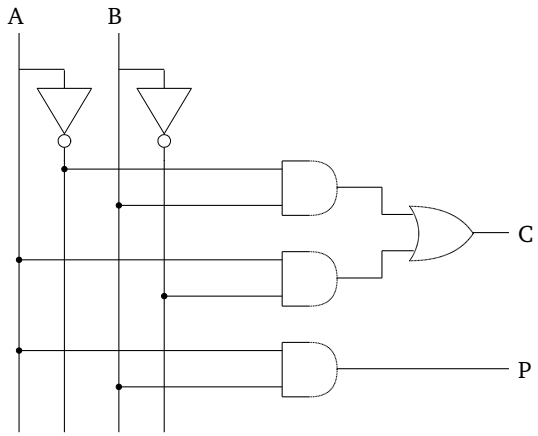
Tabela 13: Istinitosne vrednosti za binarni polusabirač

P_p	A	B	C	P_t
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

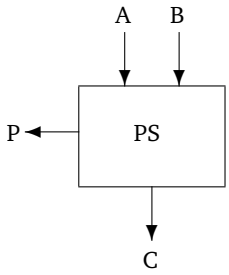
Tabela 14: Istinitosne vrednosti za binarni sabirač

Na osnovu tabele se lako formira SDNF za funkcije C i P:

$C = \overline{A}B + A\overline{B}$
$P = AB$



Slika 18: Implementacija polusabirača



Slika 19: Oznaka za polusabirač

Kombinatorna mreža koja realizuje sabiranje dve binarne cifre višecifrenog binarnog broja uz uzimanje u obzir prenosa sa prethodnog mesta se naziva (pun) sabirač.

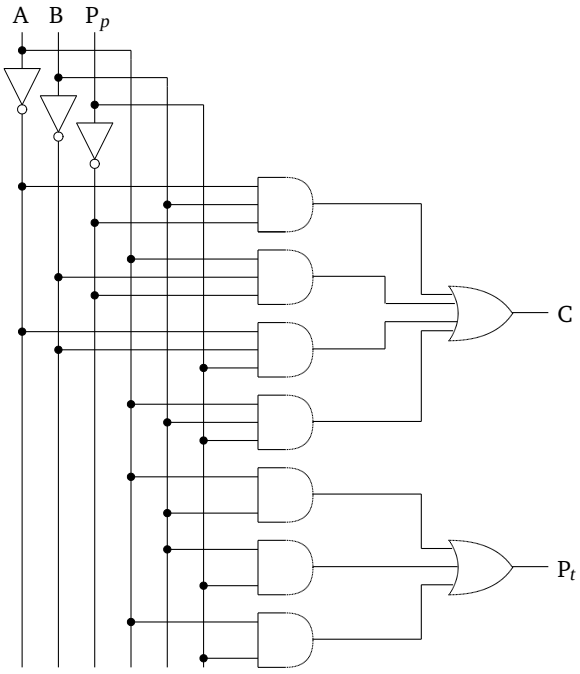
Tabela istinitosnih vrednosti sabirača je prikazana na prethodnoj strani.

Na osnovu tabele se formira SDNF za funkcije C i P_t:

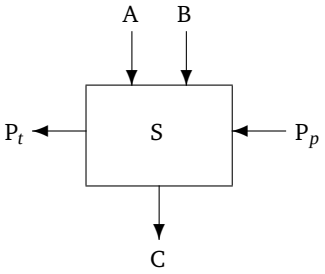
$$\begin{aligned} C &= \overline{A}B\overline{P_p} + A\overline{B}\overline{P_p} + \overline{A}\overline{B}P_p + ABP_p \\ P_t &= AB\overline{P_p} + \overline{A}BP_p + A\overline{B}P_p + ABP_p \end{aligned}$$

SDNF za P_t može da se minimizuje jednostavnom algebarskom transformacijom:

$$\begin{aligned} P_t &= AB\overline{P_p} + \overline{A}BP_p + A\overline{B}P_p + ABP_p + ABP_p + ABP_p \\ &= AB(\overline{P_p} + P_p) + BP_p(\overline{A} + A) + AP_p(\overline{B} + B) \\ &= AB + BP_p + AP_p \end{aligned}$$



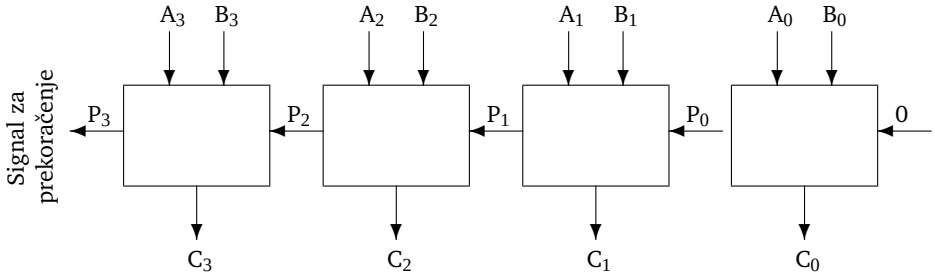
Slika 20: Implementacija sabirača



Slika 21: Oznaka za sabirač

Višecifreni sabirači

Sabiranje višecifrenih binarnih brojeva se realizuje pomoću jednog sabirača za svako binarno mesto. U ovom slučaju se zbog akumulacije kašnjenja javlja značajno kašnjenje izmedju bita najmanje i najveće težine kod sabirača sa većim brojem mesta.



Slika 22: 4-bitni sabirač

Tehnikom nazvanom *odredjivanje budućih prenosa* (eng. *carry lookahead*) se kašnjenje najvećim delom eliminiše. Ova tehnika se primenjuje na sledeći način:

Označimo sa P_i prenos koji se dobija sabiranjem binarnih cifara na poziciji i i koji se prenosi u sabirač koji sabira cifre na poziciji $i+1$. Cilj je da definišemo P_i pomoću logičkog izraza koji ne zavisi od P_j, i$\leq$j. U tabeli 14, uzimimo da je $P_p = P_{i-1}$ i $P_t = P_i$. Takodje očigledno važi $P_{-1} = 0$. Tada važe sledeće jednakosti:

$P_0 = A_0B_0$
$P_1 = A_1B_1 + (A_1 + B_1)P_0$

Zamenom prve jednakosti u drugoj dobija se

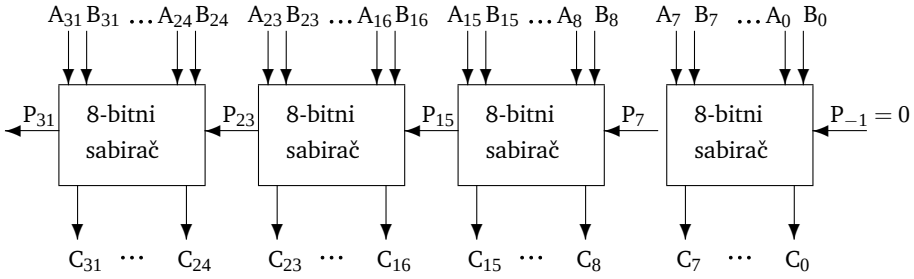
$P_1 = A_1B_1 + A_1A_0B_0 + B_1A_0B_0$

Koristeći istu proceduru dobija se

$P_2 = A_2B_2 + A_2A_1B_1 + A_2A_1A_0B_0 + A_2B_1A_0B_0 + B_2A_1B_1 + B_2A_1A_0B_0 + B_2B_1A_0B_0$

Na sličan način se dobijaju vrednosti P_i za proizvoljno velike sabirače.

Ovakav pristup je praktično neprimenljiv za dugačke brojeve zbog glomaznosti dobijenih logičkih izraza. Zbog toga se u realnim situacijama odredjivanje budućih prenosa radi za sabirače koji sabiraju od 4 do 8 bitova istovremeno.



Slika 23: Konstrukcija 32-bitnog pomoću 8-bitnih sabirača

Sekvencijalne mreže

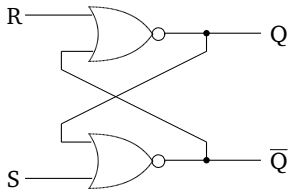
Sekvencijalne mreže predstavljaju skup povezanih logičkih elemenata čiji je izlaz u nekom vremenskom trenutku funkcija koja zavisi od tekućeg stanja elemenata mreže i vrednosti ulaza u tom istom vremenskom trenutku.

Flip-flop elementi

Flip-flop je najjednostavniji oblik sekvencijalne mreže. Svi flip-flop elementi poseduju sledeće dve osobine:

1. Flip-flop je bistabilni element. Poseduje dva stabilna stanja i u slučaju nepostojanja ulaza ostaje u trenutnom stanju. Na taj način flip-flop može da memorišadržaj jednog bita.
2. Flip-flop ima dva izlaza koji su medjusobno komplementarni. Ovi izlazi se obično označavaju sa Q i $\overline{Q}$.

R-S element

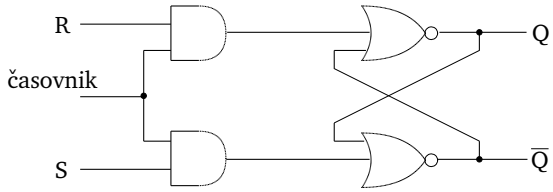


Slika 24: R–S flip-flop element

Karakteristična tabela			Uprošćen oblik		
Ulaz RS	Tekuće stanje Q_n	Naredno stanje Q_{n+1}	R	S	Q_{n+1}
00	0	0	0	0	Q_n
00	1	1	1	0	0
10	0	0	0	1	1
10	1	0	1	1	—
01	0	1			
01	1	1			
11	0	—			
11	1	—			

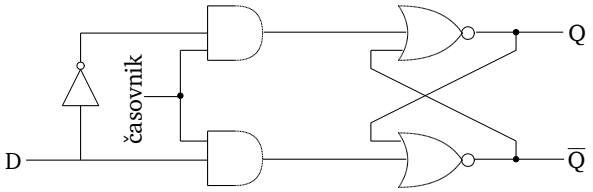
Tabela 15: Karakteristična tabela R-S flip flop elementa

R–S flip-flop sa časovnikom za sinhronizaciju



Slika 25: R-S flip-flop sa časovnikom sa sinhronizaciju

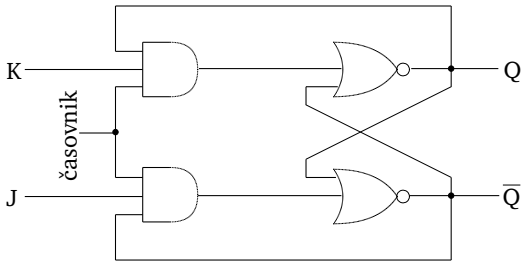
D flip-flop element



D	Q_{n+1}
0	0
1	1

Slika 26: D flip-flop sa časovnikom sa sinhronizaciju

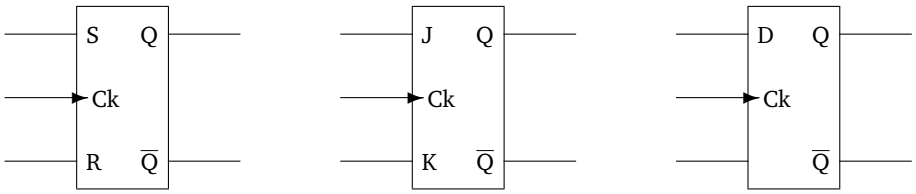
J–K flip-flop



J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

Slika 27: JK flip-flop sa časovnikom sa sinhronizaciju

1. J=0, K=0: izlaz je nepromenjen, stanje elementa je stabilno.
2. J=1, K=0: izlaz se postavlja na 1, bez obzira na prethodnu vrednost.
3. J=0, K=1: izlaz se postavlja na 0, bez obzira na prethodnu vrednost.
4. J=1, K=1: prethodna vrednost izlaza se invertuje. Npr. ako je prethodno Q bilo 0 sada postaje 1 i obratno.



R–S flip-flop J–K flip-flop D flip-flop

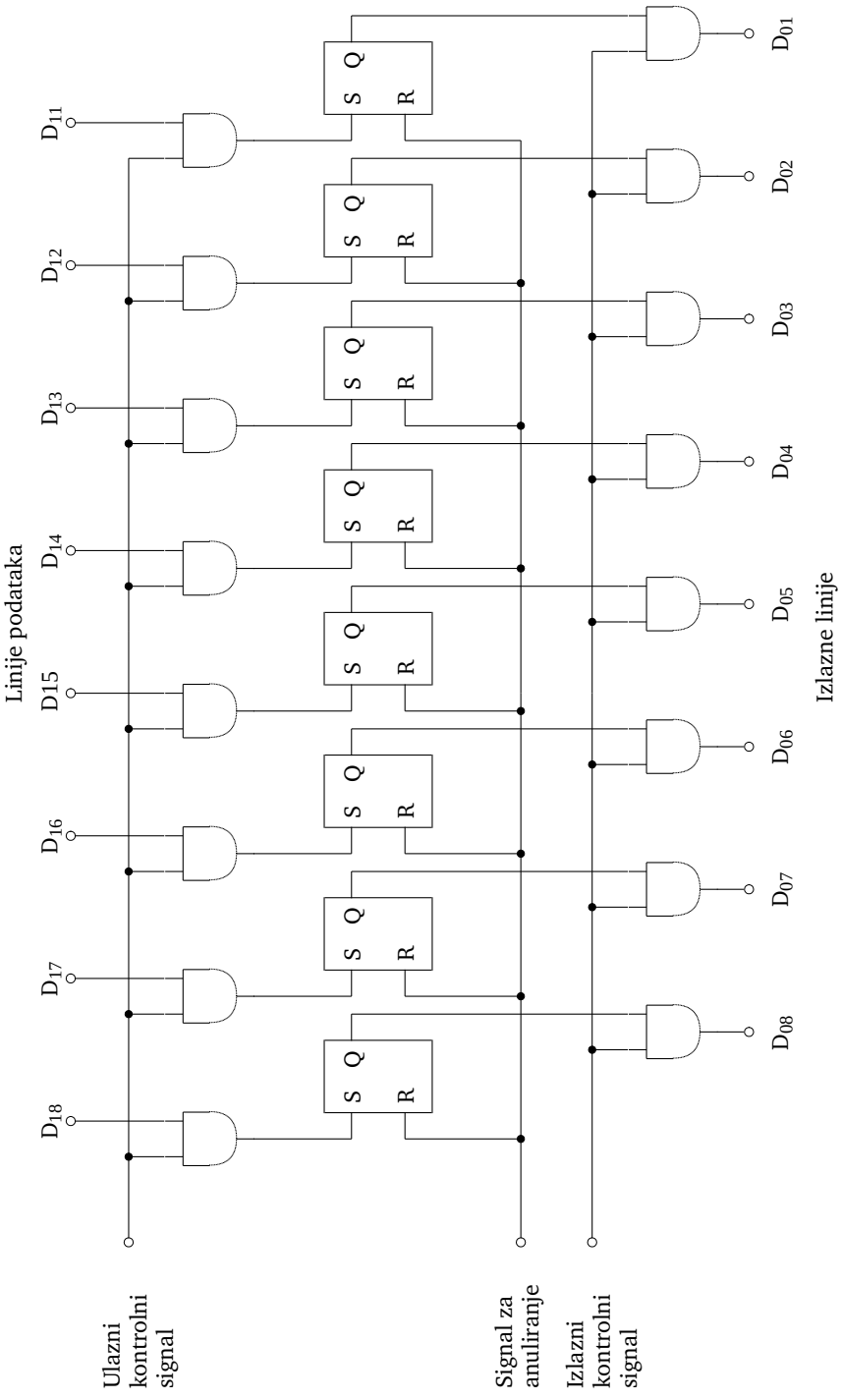
Slika 28: Oznake za flip-flop elemente u složenijim mrežama

Registri

Registri su digitalna kola koja se koriste unutar CPU-a za čuvanje jednog ili više bitova podataka. Dva osnovna tipa registara koja se najčešće koriste su paralelni i pomerački registri. Obe vrste se realizuju u obliku sekvencijalnih mreža u čijoj izgradnji se koriste flip-flop elementi.

Paralelni registri

Paralelni registar se sastoji od skupa 1-bitnih memorijskih jedinica čiji sadržaj može istovremeno da se čita/upisuje.



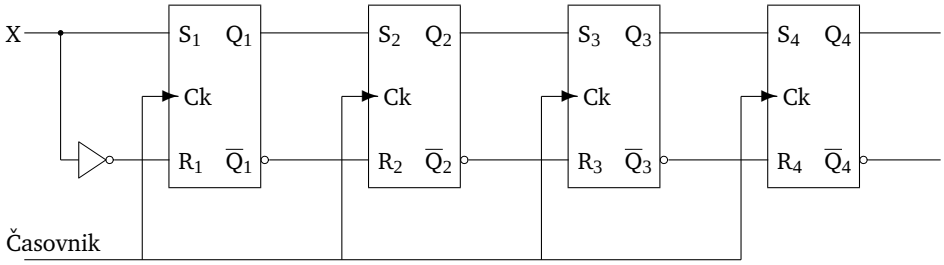
Slika 29: 8-bitni paralelni registar

Pomerački registri

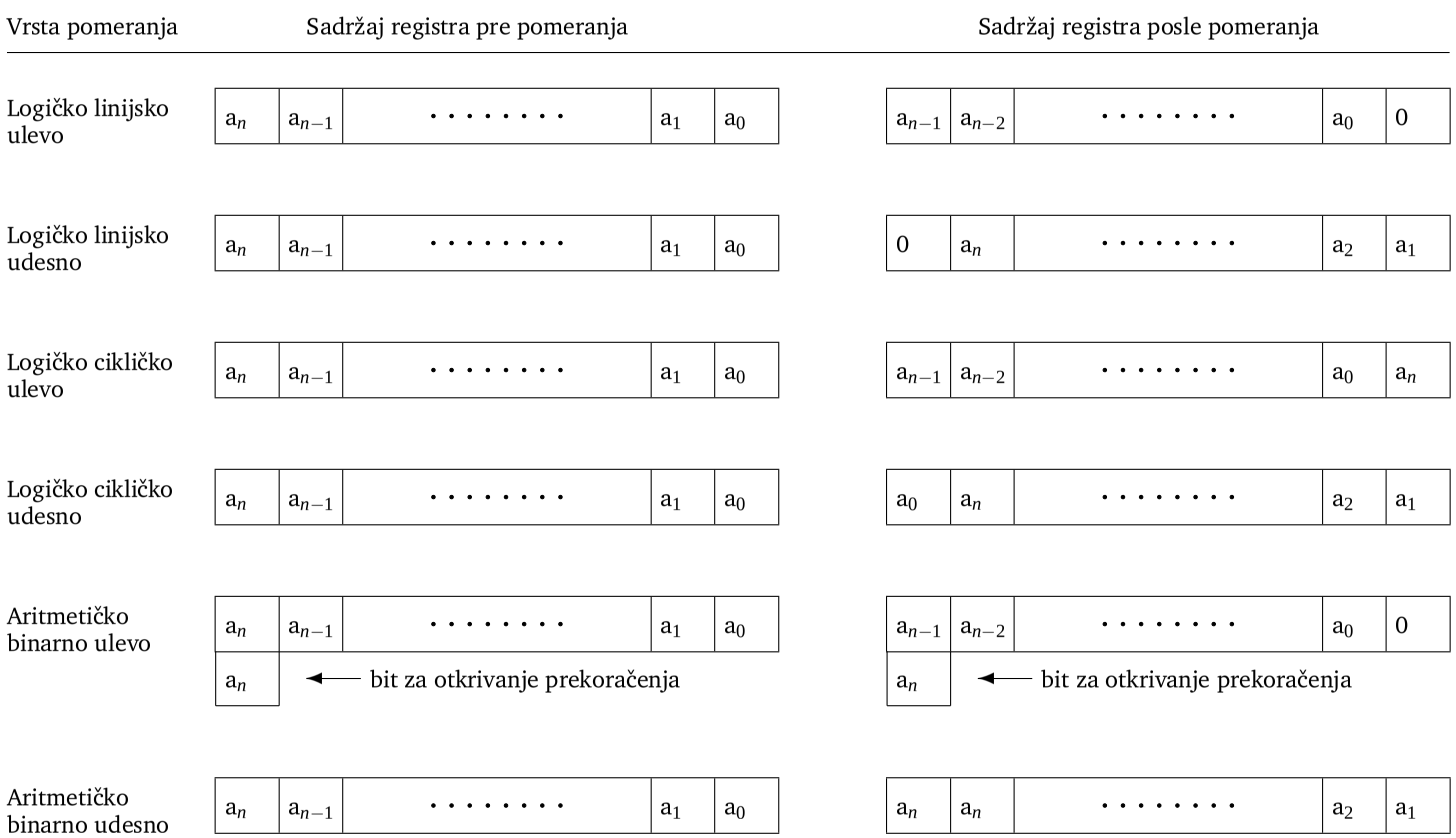
Pomerački registri prihvataju i prenose informacije serijski. Sadržaj registra se može pomerati na dva načina:

1. Logički, pri čemu se sadržaj registra posmatra kao binarna reč. Logičko pomeranje može biti linijsko ili cikličko.

2. Aritmetički, pri čemu se sadržaj registra posmatra kao broj, i pri pomeranju se vodi računa o znaku. Aritmetičko pomeranje može biti binarno (kada je sadržaj binarni broj) ili dekadno (kada je sadržaj BCD broj). Pri aritmetičkom pomeranju BCD brojeva istovremeno se pomeraju grupe od po 4 bita.



Slika 30: 4-bitni pomerački registar (logičko linijsko pomeranje)



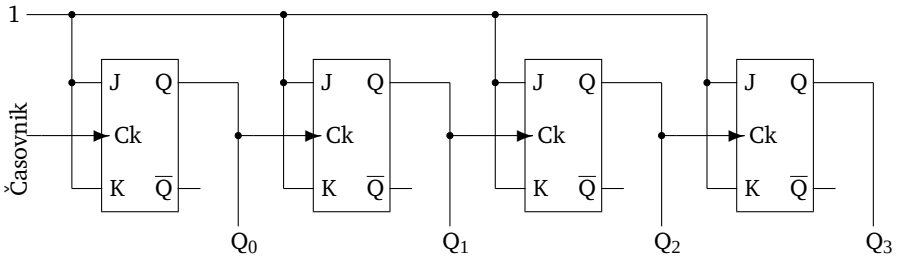
Slika 31: Vrste pomeranja sadržaja registra

Brojači

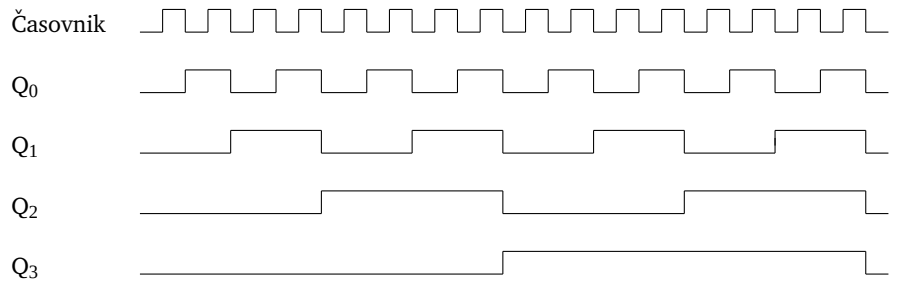
Brojači su registri koji imaju mogućnost jednostavnog uvećanja ili umanjenja vrednosti za 1. Vrednost brojača se uvećava (ili umanjuje) po modulu jednakom kapacitetu registra. Maksimalna vrednost koju brojač od n flip-flopova može da registruje je 2^n-1. Kada se brojač koji ima maksimalnu vrednost poveća za 1, njegova vrednost se menja na 0.

Asinhroni brojači

Kod asinhronih brojača se ulazni signal prenosi kroz brojač u obliku talasa.



a) Sekvencijalna mreža



b) Vremenski dijagram

Slika 32: 4-bitni asinhroni brojač

Sinhroni brojači

Kod sinhronih brojača se izlazi svih flip-flop elemenata koji ih sačinjavaju menjaju istovremeno.

Primer – 3-bitni binarni brojač. U implementaciji će biti korišćena tri J–K flip-flop elementa, označena sa *A*, *B*, *C*. Neka element *A* predstavlja bit najveće, a element *C* bit najmanje težine. Označimo J i K ulaze u svaki od elemenata sa odgovarajućim indeksom; J_a, K_a, J_b, K_b, J_c i K_c; izlaze i njihove komplemente označimo prema oznakama samih brojača: A, $\overline{A}$, B, $\overline{B}$, C i $\overline{C}$.

Q_n	J	K	Q_{n+1}
0	0	n	0
0	1	n	1
1	n	1	0
1	n	0	1

Na osnovu karakteristične tabele vidi se da brojač može da ima sledeće vrednosti:

A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Odavde se vidi da ako je vrednost brojača A=0, B=0, C=0 tada u narednom uvećanju vrednosti brojača treba zadržati vrednosti A=0, B=0 i promeniti vrednost C na 1. Iz modifikacije karakteristične tabele se vidi da je za održavanje izlaza na nuli potrebno da bude J=0 dok je vrednost K

A	B	C	J_a	K_a	J_b	K_b	J_c	K_c
0	0	0	0	n	0	n	1	n
0	0	1	0	n	1	n	n	1
0	1	0	0	n	n	0	1	n
0	1	1	1	n	n	1	n	1
1	0	0	n	0	0	n	1	n
1	0	1	n	0	1	n	n	1
1	1	0	n	0	n	0	1	n
1	1	1	n	1	n	1	n	1

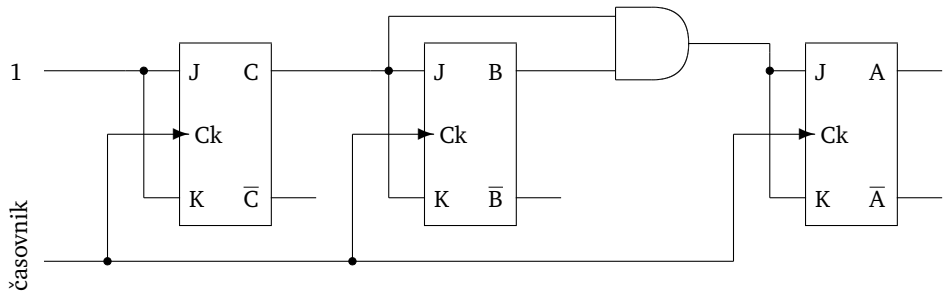
Tabela 16: Tabela istinitosnih vrednosti 3-bitnog sinhronog brojača

nebitna. Slično, za promenu vrednosti sa 0 na 1 (za flip-flop C) potrebno je da bude J=1 dok je vrednost K nebitna. Time su odredjene odgovarajuće vrednosti za J_a, K_a, J_b, K_b, J_c i K_c u tom redu tabele istinitosnih vrednosti. Na sličan način se formiraju i ostali redovi. Dobijena tabela istinitosnih vrednosti je prikazana u tabeli 16.

Naredni korak predstavlja minimizacija funkcija koje predstavljaju ulazne veličine u flip-flop elemente. Npr. pomoću Karnoovih mapa dobija se

$J_a=BC \quad K_a=BC \quad J_b=C \quad K_b=C \quad J_c=1 \quad K_c=1$

Na osnovu minimizovanih vrednosti se konstruiše sinhroni brojač čiji je izgled prikazan na slici 33.



Slika 33: 3-bitni sinhroni brojač