ISTVÁN MATIJEVICS

DIGITALNA TEHNIKA

skripta

VIŠA TEHNIČKA ŠKOLA
SUBOTICA

2003

1. UVOD

1.1 UVODNE REČI

U matematici simboliku su definisali i uveli francuski matematičari Francois Viete (1540-1603) i René Descartes (1596-1650) u 16. i 17. veku. U logici (logička rasuđivanja, logički iskazi) bilo je nekoliko pokušaja za definisanje simbolike, ali na kraju matematičar Gottfried Wilhelm Leibniz (1646-1716) je rešio uspešno taj zadatak.

$$F = \int_a^b f(x)dx$$

$$x + \sqrt{y(2a - y)} = a \cdot Arc \cdot \cos\frac{a - y}{a}$$

$$\frac{\partial F}{\partial x}(X - x) + \left(\frac{\partial F}{\partial y}\right)(Y - y) = 0$$

Aristotel je već u starom veku počeo izraditi logiku, a Leibniz je pokušao algebraizaciju te logike, približavajući se Boole-evoj algebri koja se danas veoma upešno koristi u tehnici.

De Morgan i Hamilton su dali značajan doprinos u razvoju matematičke logike, ali najuspešniji matematičar ipak je bio englez George Boole (1815-1864), ko je osnivač savremene matematičke logike (današnji naziv te grane matematike je Boole-ova algebra).

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

$$\overline{y} = \left(x_1 x_2 + x_3 \overline{x}_4 \left(x_1 + \overline{x}_2\right)\right)$$

$$(A \cdot B) + C = (A + B) \cdot (B + C)$$

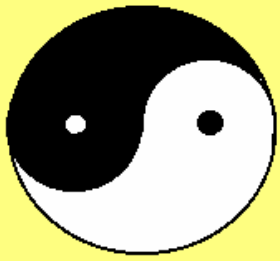
Boole, a i njegovi sledbenici su isključivo razvijali ovu algebru tako, kao da je ona sama sebi svrha, interesujući se pri tome malo za mogućnost primene dostignutih rezultata u matematici. Tadašnji stepen razvoja tehnike nije omogućila primenu ovu granu matematike u tehnici, tj. u rešavanju prekidačkih problema.

Samo u 20. veku, zahvaljujući pre svega radovima Claude Shannon-a Boole-ova algebra, tj. algebra logike je našla i svoju primenu u tehnici.

Oznake, tj. matematička simbolika, koja se koristi u praktičnoj, tehnički orijentisanoj Boole-ovoj algebri nije identična s oznakama, koje se koriste u teorijskoj matematici. Postoji, međutim, između odgovarajućih oznaka jednoznačna korespondencija, i mogućnost transkripcije je veoma laka.

Pošto često koristi naše razmišljanje dvoznačnu strukturu (hladno-toplo, malo-veliko, istinito-neistinito itd.), a to je binarno, ili bistabilno, pomoću Boole-ove algebre je moguće to razmišljanje i opisati.

I Ching (Ji Džing): Stari Kinezi su već pre 5000 godina napisali Knjigu promene, gde su korišćeni principi Jang-a i Jin-a. Osnovni polariteti u kojima se ispoljavaju Jang i Jin jesu: muško-žensko, aktivno-pasivno, inicijativa-održavanje, ekstroverzija-introverzija, napredovanje- povlačenje itd. Kombinacijom Jang-a i Jin-a su dobili ukupno 64 heksagrama, koji su služili za proricanje



Relacije u Boole-ovoj algebri se definišu iskazima. Iskaz je sud, koji ima smisla i za koji važe sledeći principi:

- princip kontradikcije, tj. svaki sud ima najviše jednu od osobina istinitosti ili neistinitosti, što znači da nema suda, koji bi bio i istinit i neistinit i
- princip isključenja trećeg, prema kome svaki sud ima bar jednu od osobina istinitosti ili neistinitosti, tj. nema suda, koji ne bi bio ni istinit ni neistinit.

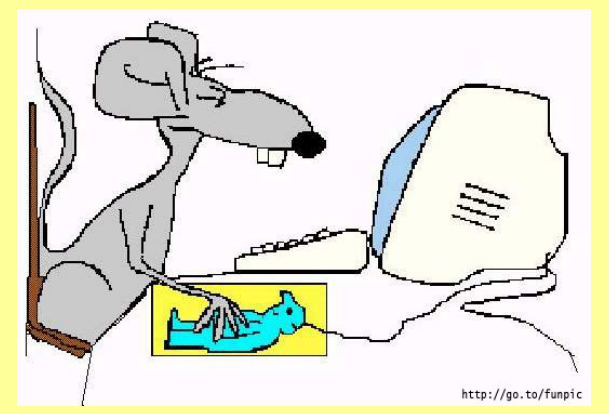
Vrednost istinitog suda označavaćemo u daljim izlaganjima sa »**1**« (jedinica), a neistinitog sa »**0**« (nula). Često se koristi i notacija za istinitog **H** (High) a za neistinitog **L** (Low).

Sudovi, koje izričemo, odnose se po pravilu na nekakav pojam, koji će u simbolici figurisati kao binarna algebarska veličina. Ova se veličina, po dogovoru, najčešće obeležava velikim slovima s početka ili s kraja abecede.

A,B,C,D....

....X,Y,Z

Pored primene u elektrotehnici (prekidači, releji, upravljački uređaji, računari itd.) digitalna tehnika našla je primenu i u pneumatici (postoje pneumatički elementi), a još i u nekim drugim granama nauke.

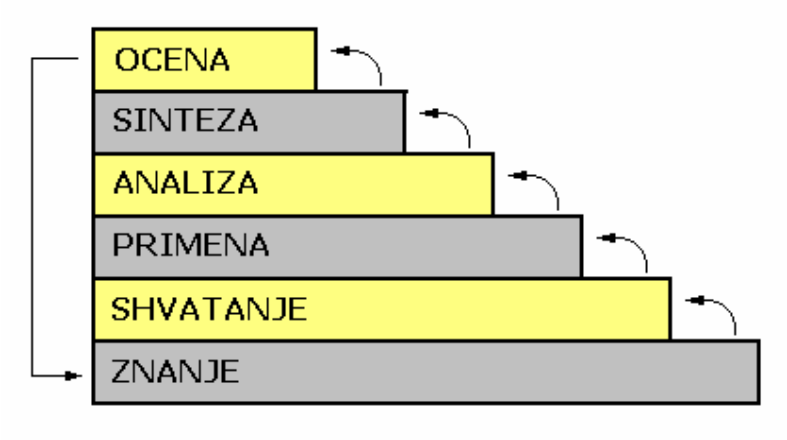


☺ ☹ ☺

1.2 ALGORITAMSKE STRUKTURE

1.2.1 OBRAZOVNI PROCES

Proces obrazovanja možemo podeliti u šest aktivnosti, koji aktivnosti su međusobno povezani, kao što je to prikazano na slici 1.1.



Slika 1.1: Proces obrazovanja

- **ZNANJE:** Znanje čini određeni skup činjenica vezanih za određene pojmove. Te pojmove reprezentuje zajednička terminologija. Prikupljanje znanja (sticanje znanja) je prvi korak, koji prethodi ostalim fazama edukacije, a predstavlja njihovu podlogu.

- **SHVATANJE:** Pojmove, činjenice i terminologiju treba razumevati, prvi korak za to je shvatanje. Važno je definisati odnose između pojmova, kao i uočavanje prelaska sa jedne predstave na drugu. Treba generisati objašnjenja vezanih za pojmove i za njihove odnose i veze. Faza shvatanja služi za određivanje skupa apstrakcija (generalizacije) određene oblasti.

- **PRIMENA:** Inžinjerske discipline vide opravdanje svog postojanja u procesu edukacije u ovoj fazi. Vremenom pokupljeno stanje u obliku apstraktnih i uopštenih postupaka u ovoj fazi su primenjene na konkretni slučaj.

- **ANALIZA:** U ovoj fazi potrebno je realizovati identifikaciju sastavnih delova elemenata, i odrediti veze među njima koje nisu eksplicitno iskazane.

- **SINTEZA:** U ovoj fazi se vrši kombinovanje elemenata i delova. Kao rezultati sinteze stvaraju se ranije nepoznate strukture. Rezultat ove faze je proizvod.

- **OCENA:** Drugi naziv ove faze je još i EVALUACIJA. To je zadnja faza edukacije, gde će se dobiti podloge za kvalitativnu i kvantitativnu ocenu vrednosti metoda, procesa i produkata.

Veoma je važno, da ocenjivanje prolazi kroz sve gore navedene faze, odnosno moramo, kao inženjeri, ili budući inženjeri da usvajamo pristup da ocenu dajemo tek onda, kada smo višestruko verifikovali sve korake u procesu edukacije.

1.2.2 OSNOVNI PRINCIPI REŠAVANJA PROBLEMA

Pošto će studenti koji slušaju predavanja posle diplomiranja postati inženjeri, moramo prvo saznati šta znači reč inženjer. Reč inženjer dolazi iz latiskog termina

IN GENIOSUS,

što znači »u nepoznatom«.

Prema tome inženjer se kreće u nepoznatom. Inženjer mora stalno da se kreće u nepoznatom, neistraženom ili nerealizovanom. Inženjertvo je primenjena disciplina, stvara sa ograničenjima od strane okruženja.

U inženjerskoj praksi moramo izolovati određeni broj koraka, u tome iskustvo pomaže, da bi se problem, koji opisuje nepoznato doveo u domen rešivog. Upoznavanje problema, znači identifikacija smanjuje polazne neodređenosti.

U identifikaciji prikupljamo informacije o problemu na različite načine:

- merenje,
- opažanje,
- razumevanje zadatka,
- analiza sličnih rešenja itd.

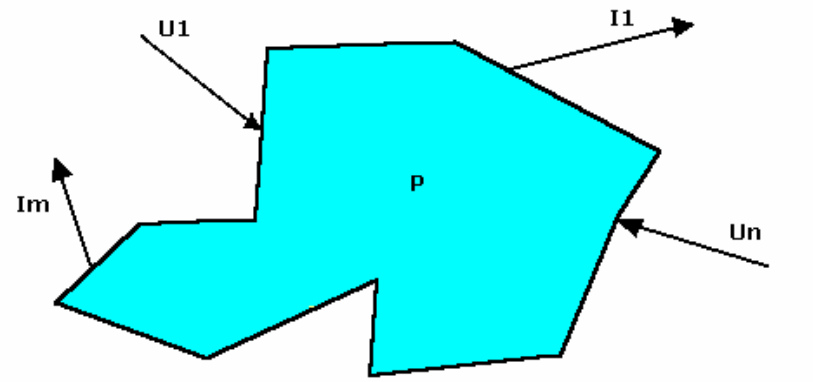
Pošto su ovi načini prikupljanja informacija posredne metode imaćemo poteškoća (na primer neadekvatni merni instrumenti). Bilo koja greška, netačni podaci u ovoj fazi su veoma značajni faktori za daljnji rad, za uspeh.

U ovoj prvoj fazi, u identifikaciji inženjeru stoje na raspolaganju:

- znanje,
- iskustvo i
- tehnička sredstva

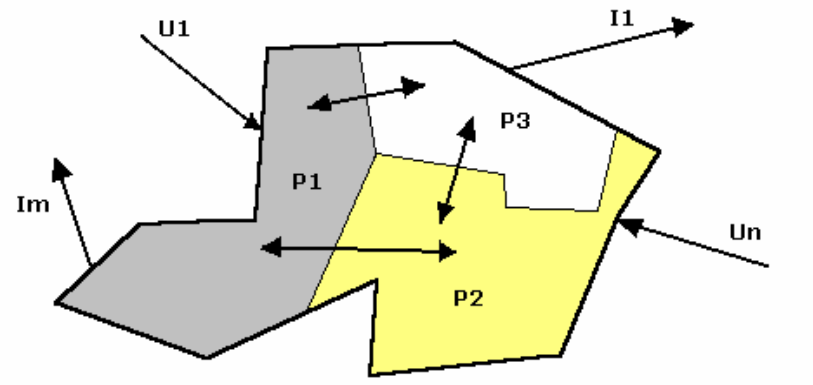
Posebno, a i zajedno nedostatak u bilo kojoj grupi činilaca bitno utiče na kvalitet identifikacije.

Pošto su problemi obično »veliki« (slika 1.2), pokušamo ih razlučivati, razdeliti na manje celine, t.j. izvršiti dekompoziciju (razlaganje).



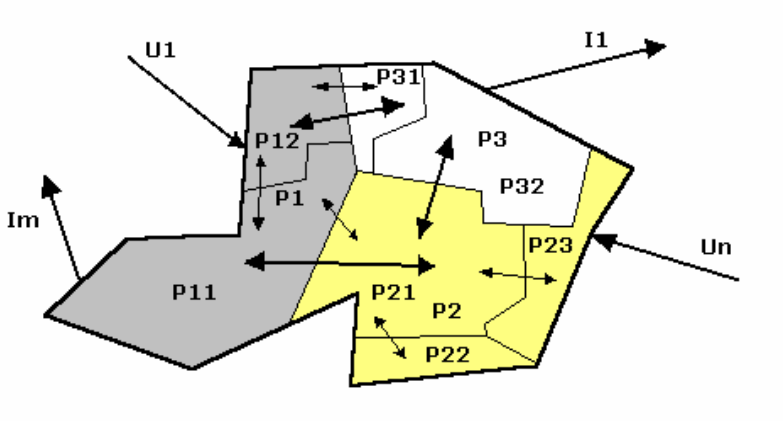
Slika 1.2: prikazivanje problema sa **n** ulaza i **m** izlaza

U ovoj fazi treba veliki problem, celinu razdeliti na nekoliko međusobno spregnutih pod-problema, delova. Složeni problemi uvek imaju unutrašnju strukturu. Problem je složen, ako ima unutrašnju strukturu. Dekompozicija je uvek hijerarhijska i zahteva dublje znanje, iskustvo i tehnička sredstva nego identifikacija. Na slici 1.3 je prikazana metoda dekompozicije.



Slika 1.3: metoda dekompozicije problema

Dekompoziciju treba nastaviti dalje, dok je to moguće, odnosno dok broj nivoa razlaganja ne promeni originalno značenje problema. Sledeći korak dekompozicije je na slici 1.4.



Slika 1.4: drugi nivo dekompozicije

Posle dekompozije na drugom nivou, ako postoji mogućnost na trećem, četvrtom itd., treba uraditi. Kraj dekompozicije je postignut, ako kod razlaganja dođe do nivoa elementarnog problema, tj. do problema, koji se relativno lako može rešiti

☺ ☹ ☺

1.2.3 ALGORITAM

Posle dekompozicije sledi faza, kada je potrebno naći postupak rešavanja, drugim rečima naći algoritam. Algoritam će dati konačnu proceduru za rešavanje celog problema.

U praksi dva osnovna principa poznajemo za određivanje strukture algoritma:

- odozgo prema dole (Top-down),
- odozdo prema gore (Bottom-up) i
- kombinacija dva principa.

U algoritmu »odozgo prema dole« kretanje je od opšteg ka pojedinačnom. U početku detalji nisu interesantni, isključivo su bitne veze većih celina (slika 1.3), posle realizacije zadataka na ovom nivou se ulazi u podprobleme drugog nivoa (slika 1.4), zatim na sledeći niži nivo itd. Na nižim nivoima složenosti su sve jednostavnije. Na poslednjem nivou dekompozicije su elementarni problemi, gde smo stigli do rešenja problema.

U algoritmu »odozdo prema gore« kretanje ide od pojedinačnog ka opštem. Pošto su poznati podproblemi (nakon dekompozicije), moguće je rešavanje istih paralelno. Cilj je formiranje skup gradivnih elemenata, a na osnovu tih elemenata rešenja problema.

Često se mogu kombinovati metode »odozgo prema dole« i »odozdo prema gore«. Pošto ne postoji precizno definisan način, kad koji princip da koristimo, inženjer uvek dodaje i intuiciju u rešenjima problema.

Način rešavanja problema »odozgo prema dole« ima prednost nad metodom rešavanja problema »odozdo prema gore« u tome, što prilikom celokupne realizacije vidi rešenje problema kao celinu. Isto je važno, da eventualno napravljena greška ne izlazi iz odgovarajućeg nivoa.

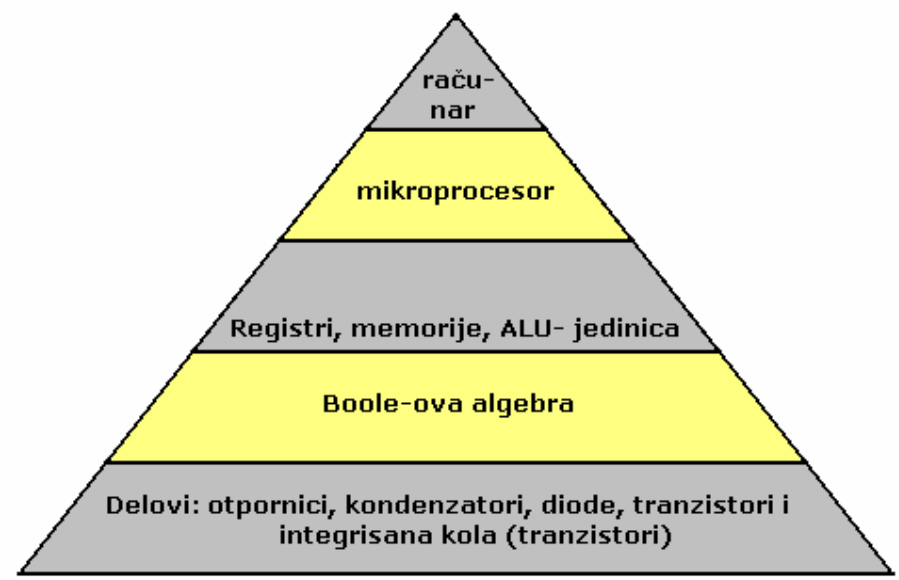
Nakon izrade algoritma sledi implementacija istog zadatim tehničkim sredstvima. Sledeća faza je testiranje funkcionalnosti. Nakon uspešnog testa treba integrisati tehnička sredsva u radno okruženje. Pri tom veoma je važno uraditi obuku korisnika.

Gore opisani postupak za rešavanje problema ako je uspešno izveden, mora da bude korektno dokumentovan.

Operativnu eksploataciju sistema poboljšava i korektno organizovano održavanje.

1.2.4 MESTO DIGITALNE TEHNIKE U TEHNICI

Na slici 1.5 je prikazan hierarhijski model strukture računara.



Slika 1.5: hierarhijski model računara, mesto digitalne tehnike

Na slici se vidi, da pomoću zakona Boole-ove algebre od osnovnih elemenata (otpornici, tranzistori itd.) možemo projektovati funkcionalne sklopove (registre, memorije itd.), od kojih ćemo sastaviti hardver mikroprocesora a zatim i računara.

2. BISTABILNA STANJA I LOGIČKI ISKAZI

2.1 LOGIČKO RAZMIŠLJANJE

Naše logičko razmišljanje ima binarnu strukturu, odnosno naši iskazi, ili sudovi uvek imaju isključivo dva stanja:

- istinito (1 – H = High) ili
- neistinito (0 – L = Low).

Nadalje, bilo koji naš iskaz ne može istovremeno da poprimi oba stanja, a mora da primi jedno od njih.

Postoji mala grupa iskaza, gde su neki iskazi neizvesni i dvosmisleni, sa takvim iskazima u ovoj skripti nećemo raditi. Na primer, iz filozofije je poznat primer sledeći:

- Svi Grci lažu, ja sam Grk.

Isto ne analiziramo one iskaze, gde zbog nedovoljne informisanosti može da dođe do situacije u iskazima »uzdržavanja od davanja izjave«.

PRIMER 2.1:

Ovaj primer nije iz tehnike, ali preko njega lako se može razumeti logika analize takvih i sličnih primera. Analiziraćemo, šta će se desiti sa osobom, ako ima, odnosno ako nema kišobran i ako izađe na slobodan prostor i pri tome kiša pada ili ne pada. Osoba će pokisnuti, ondosno neće, u zavisnosti od okolnosti koje mogu za kišu biti dvojake i za prisustvo kišobrana takođe dvojake. Iz toga proizilazi da sam iskaz može da bude dvojak: osoba pokisne ili ne.

Elementi u iskazu su:

- osoba,
- kiša i
- kišobran.

Pošto sva tri elementa mogu imati dva stanja (**da** i **ne**), dobićemo, da je ukupan broj varijacija $N = 2^3 = 8$. Od tih 8 sudova neki će biti tačni, neki netačni, ali svaki će biti određen (Tabela 2.1).

Tabela 2.1: primer 2.1				
Redni br.	**Pada li kiša?**	**Ima li kišobran?**	**Osoba pokisne?**	**Iskaz tačan?**
0.	**ne**	**ne**	**ne**	**da**
1.	ne	ne	da	ne
2.	**ne**	**da**	**ne**	**da**
3.	ne	da	da	ne
4.	da	ne	ne	ne
5.	**da**	**ne**	**da**	**da**
6.	**da**	**da**	**ne**	**da**
7.	da	da	da	ne

Istina, da je tabela pregledna, tačna i precizna, ali formula (obrazac) za gornji primer bi bio kraći. U sledećim poglavljima će se dati način, kako se može naći obrazac za takve zadatke.

PRIMER 2.2:

U ovom tehničkom primeru će se razmotriti rad prese pri presovanju nekog radnog predmeta. Presa sme da radi sa pritiskom na pedalu (polugu), ali sa uslovom, da je prethodno odstranjen iz kalupa izrađen otpresak, ako je spuštena zaštitna rešetka i ako je postavljen u kalup novi materijal (tabela 2.2).

Tabela 2.2: primer 2.2					
r.b.	pedala	rešetka	novi materijal	otpresak odstranjen	presa radi
0.	ne	ne	ne	ne	ne
1.	ne	ne	ne	da	ne
2.	ne	ne	da	ne	ne
3.	ne	ne	da	da	ne
4.	ne	da	ne	ne	ne
5.	ne	da	ne	da	ne
6.	ne	da	da	ne	ne
7.	ne	da	da	da	ne
8.	da	ne	ne	ne	ne
9.	da	ne	ne	da	ne
10.	da	ne	da	ne	ne
11.	da	ne	da	da	ne
12.	da	da	ne	ne	ne
13.	da	da	ne	da	ne
14.	da	da	da	ne	ne
15.	da	da	da	da	da

3. NUMERIČKI SISTEMI

3.1 UVOD

U inženjerskoj praksi veličine predstavljamo sa brojevima. Najčešće koristimo dva skupa brojnih sistema:

- nepozicioni (aditivno-supstraktivni) i
- pozicioni (aditivno-multiplikativni).

Primer za nepozicioni brojni sistem je pisanje brojeva sa "rimskim brojevima".

PRIMER 3.1:

Broj 129 sa rimskim brojevima je:

CXXIX = 100 + 10 + 10 + (10 – 1) = 129.

I = 1
V= 5
X = 10
L = 50
C = 100
D = 500
M = 1000

Kod pozicionih brojnih sistema vrednost broja se iskazuje kao kombinacija vrednosti cifre (znamenke) i težinskog faktora definisanog pozicijom cifre u okviru broja.

Pozicioni brojni sistem čine (tabela 3.1):
b baza (osnova) i
$C = \{C_0, C_1, C_2, ...C_n\}$ skup cifara (znamenki, brojki).

Baza b je strogo veća od vrednosti bilo koje znamenke brojnog sistema (b > C_i i ε[0,n]).

Tabela 3.1: brojni sistemi

baza	skup	naziv
2	C={0,1}	binarni
3	C={0,1,2}	trinarni
4	C={0,1,2,3}	kvaternarni
8	C={0,1,2,3,4,5,6,7}	oktalni
10	C={0,1,2,3,4,5,6,7,8,9}	dekadski
16	C={0,1,2,3,4,5,6,7,8, 9,A,B,C,D,E,F}	heksadeci-malni

Brojevi mogu biti:

- prirodne,
- cele,
- racionalne,
- iracionalne,
- kompleksne itd.

U nekim tehničkim sistemima, kao i u računarima operacije se svode na dovođenje bistabilnih komponenata u određeno stanje (0 ili 1). Danas se u upotrebi nalazi više numeričkih sistema, ali se kod digitalnih uređaja isključivo primenjuje binarni sistem, i to zbog svoje jednostavnosti i jednoznačnosti. Naime, pomoću binarnog sistema je moguće jednoznačno odrediti dva stanja, a u upotrebi su samo dva znaka - **0** i **1**.

Pomoću binarnih brojeva u binarnom sistemu je moguće iskazati bilo koji broj nekog drugog sistema.

3.2 DEKADNI BROJNI SISTEM

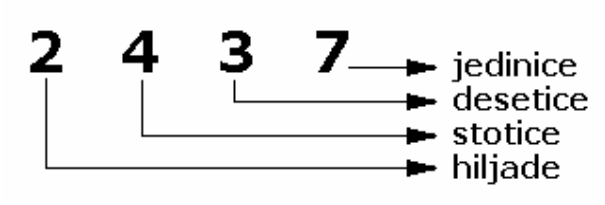
Dekadni sistem je najčešće primenjivani numerički sistem u svetu. Osnova ovog sistema je **10**, što znači da se sa deset cifara, **0, 1, 2, 3, 4, 5, 6, 7, 8** i **9** mogu napisati svi brojevi.

☺ ☹ ☺

PRIMER 3.2:

Brojne veličine u dekadnom sistemu se prikazuju u nizu, gde se cifre pišu s leva na desno: 2437.

Najznačajnije cifre su na levoj strani, da bi im vrednost opadala udesno. Svaka cifra ima svoju težinsku vrednost, usvojenu po dogovoru (slika 3.1):



Slika 3.1: primer za dekadni broj

Broj 2437 bi se mogao zapisati i na nekoliko sledećih načina:

$2437 = 2000 + 400 + 30 + 7$ ili

$2437 = 2 * 1000 + 4 * 100 + 3 * 10 + 7 * 1$ ili

$2437 = 2*10^3 + 4*10^2 + 3*10^1 + 7*10^0$

Ovaj način zapisa dekadnog broja nije praktičan, ali se može uočiti princip stvaranja dekadnog sistema :

$10^0,\ 10^1,\ 10^2,\ 10^3, \ldots$

U opštem slučaju, svaki dekadni broj se može zapisati pomoću težinskih vrednosti i cifara:

$$N= \ldots d*10^3 + c*10^2 + b*10^1 + a*10^0;$$

gde su:

- 0, 1, 2 i 3 stepeni broja 10 težinske vrednosti,
- **a, b, c i d** su cifre.

Kraći zapis dekadnog broja, koji je usvojen, izgleda ovako : **dcba** .

Mada je dekadni sistem najčešće primenjivan sistem za računanje, umesto njega se može koristiti i bilo koji drugi sistem. Međutim, dekadni sistem poseduje izvesnu prednost u odnosu na druge sisteme, jer su aritmetičke operacije znatno uprošćene.
Pomoću dekadnog sistema moguće je predstaviti i brojeve čija je apsolutna vrednost u intervalu između **0** i **1**, odnosno **decimalne brojeve**. Zbog toga se uvodi decimalni zarez ili tačka, a koriste se i stepeni broja **10** sa negativnim eksponentima.

PRIMER 3.3:

$$35.625 = 3*10^{1} + 5*10^{0} + 6*10^{-1} + 2*10^{-2} + 5*10^{-3}$$

Mada kod aritmetičkih operacija dekadni sistem ima znatnu prednost pred drugim brojnim sistemima, on nije prihvaćen u računarskoj tehnici. Naime, kod elektronskih brojnih sistema, brojevi se predstavljaju različitim potencijalnim nivoima. Kada bi se koristio dekadni sistem, morala bi se projektovati takva kola, koja bi jednoznačno definisala deset različitih nivoa za deset cifara. Ovakva kola bi bilo moguće konstruisati, ali bi ovakva konstrukcija bila previše komplikovana, i bilo bi više izvora grešaka. Zbog toga se u računarskim sistemima upotrebljava **binarni sistem**.

3.3 BINARNI BROJNI SISTEM

Binarni brojni sistem ima za osnovu broj 2, tako da se svi binarni brojevi mogu napisati pomoću cifara **0** i **1**. Ovaj brojni sistem je za čoveka vrlo nepraktičan, ali je ipak najvažniji sistem u svetu elektronike. Razlog za uvođenje ovog sistema u upotrebu je strogo tehnološke prirode, jer su sa stanovišta proizvodnje elektronskih sklopova, najjeftiniji, najsigurniji i najpouzdaniji oni uređaji i sklopovi, koji razlikuju dva stanja; npr. ima li napona ili nema, postoji li magnetno polje ili ne, bez naponskog impulsa - sa naponskim impulsom, itd. Pri tome je mogućnost generisanja greške svedena na minimum, jer je daleko lakše konstatovati prisustvo ili odsustvo napona, nego meriti vrednost tog napona bez greške.

Princip stvaranja binarnog sistema je sličan stvaranju dekadnog sistema - težinske vrednosti su određene položajem i iznose:

2^0, 2^1, 2^2, itd. , a cifre su 0 i 1.

Bilo koji broj se može prikazati u binarnom sistemu, u sledećem obliku :

$$N = ...+ n*2^3 + m*2^2 + l*2^1 + k*2^0 , + p*2^{-1} + q*2^{-2} + r*2^{-3} + ... ,$$

skraćeni zapis broja N je **nmlk.pqr**, gde su stepeni broja 2 težinske vrednosti, a **n, m, l, k, p, q, r** cifre **0** ili **1**.

PRIMER 3.4:

Broj 37 u dekadnom sistemu se binarno zapisuje kao 100101.

$100101 = 1 * 2^5 + 0 * 2^4 + 0 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2$

3.3.1 PRETVARANJE DEKADNOG BROJA U BINARNI

Broj napisan u dekadnom zapisu se u binarni broj pretvara deljenjem na sledeći način: dekadni broj se deli sa dva. Ako je broj deljiv sa dva upisuje se nula, a ako je broj neparan, zapisuje se jedinica. Rezultat se dalje deli i na isti način se zapisuju nule i jedinice. Ovaj postupak se ponavlja dok rezultat deljenja ne bude nula.

PRIMER 3.5:

Pretvoriti broj 298_{10} u binarni.

298 : 2 = 149	(0)	**0**
149 : 2 = 74	(1)	**1**0
74 : 2 = 37	(0)	**0**10
37 : 2 = 18	(1)	**1**010
18 : 2 = 9	(0)	**0**1010
9 : 2 = 4	(1)	**1**01010
4 : 2 = 2	(0)	**0**101010
2 : 2 = 1	(0)	**0**0101010
1 : 2 = 0	(1)	**1**00101010

Dekadni broj 298 se u binarnom sistemu predstavlja kao 100101010, tj. $298_{10} = 100101010_2$.

Decimalni racionalni broj se u binarni pretvara metodom množenja - decimalni broj se množi sa dva. Ako je proizvod množenja veći od jedan, tada se jedinica zapisuje iza zareza, a ako nije, zapisuje se nula. Ovaj postupak se ponavlja dok se ne dobije tražena tačnost.

PRIMER 3.6:

Pretvoriti broj 0.738 u binarni.

0.738 * 2 = 1.476	0.**1**
0.476 * 2 = 0.952	0.1**0**
0.952 * 2 = 1.904	0.10**1**
0.904 * 2 = 1.808	0.101**1**
0.808 * 2 = 1.616	0.1011**1**
0.616 * 2 = 1.232	0.10111**1**
0.617 * 2 = 0.464	0.1011110

Broj 0.738 se u binarnom obliku može zapisati kao 0.1011110, odnosno $0.738_{10} = 0.1011110_2$.

3.3.2 PRETVARANJE BINARNOG BROJA U DEKADNI

Pretvaranje binarnog broja u dekadni je prilično jednostavan postupak, i svodi se na izračunavanje sume težinskih vrednosti koje su zastupljene u celom broju (tabela 3.2).

Tabela 3.2: binarne vrednosti	
Binarno mesto	**Vrednost**
	$2^{10} = 1024$
levo od tačke	$2^{9} = 512$
	$2^{8} = 256$
	$2^{7} = 128$
	$2^{6} = 64$
	$2^{5} = 32$
	$2^{4} = 16$
	$2^{3} = 8$
	$2^{2} = 4$
	$2^{1} = 2$
	$2^{0} = 1$
desno od tačke	$2^{-1} = 0.5$
	$2^{-2} = 0.25$
	$2^{-3} = 0.125$
	$2^{-4} = 0.0625$
	$2^{-5} = 0.03125$
	$2^{-6} = 0.015625$
	$2^{-7} = 0.0078125$
	$2^{-8} = 0.00390625$
	$2^{-9} = 0.001953125$
	$2^{-10} = 0.0009765625$

PRIMER 3.7:

Pretvoriti binarni broj 10010111 u dekadni.

$$10010111 = 1*2^{7} + 0*2^{6} + 0*2^{5} + 1*2^{4} + 0*2^{3} + 1*2^{2} + 1*2^{1} + 1*2^{0}$$

$$= 128 + 0 + 0 + 16 + 0 + 4 + 2 + 1$$

$$10010111_2 = 151_{10}$$

3.3.3 ARITMETIKA BINARNIH BROJEVA

3.3.3.1 Sabiranje brojeva u binarnom sistemu

Sabiranje binarnih brojeva slično je sabiranju brojeva u dekadnom sistemu. Kod ovakvog sabiranja postoje sledeće kombinacije, odnosno pravila :

$0 + 0 = 0$; $0 + 1 = 1$; $1 + 0 = 1$; $1 + 1 = 0$, prenos 1

Broj koji se prenosi dodaje se sledećoj koloni višeg reda.

PRIMER 3.4:

Sabrati brojeve 11_{10} i 14_{10} u binarnom sistemu.

$11 + 14 = 25$,

$11_{10} = 01011_2$, $14_{10} = 01110_2$,

prenos	1	1	1		
	0	1	0	1	1
+	0	1	1	1	0
rezultat	1	1	0	0	1

3.3.3.2 Oduzimanje binarnih brojeva

Oduzimanje binarnih brojeva je moguće izvesti na isti način kao i u dekadnom sistemu.

PRIMER 3.9:

Oduzeti brojeve 12_{10} i 7_{10} u binarnom sistemu.

12_{10}	$12_{10} = 01100_2$	01100
$\underline{-7_{10}}$	$7_{10} = 00111_2$	$\underline{- 00111}$
5_{10}		00101

Međutim, pošto elektronski računari imaju, zbog ekonomičnosti, ugrađen sklop samo za sabiranje, oduzimanje se mora rešiti ovim sklopom. Oduzimanje je moguće rešiti preko sabiranja uvođenjem **drugog komplementa**. Drugi komplement broja se dobija na sledeći način :

- svaka cifra broja se menja u suprotnu vrednost, odnosno jedinice prelaze nule i obrnuto. Ovako se dobija takozvani **prvi komplement**.
- dodavanjem jedinice prvom komplementu dobija se **drugi komplement**.

PRIMER 3.10:

Odrediti drugi komplement broja 011001_2 (25_{10}).

broj 25 : 011001
prvi komplement : 100110
\+ 1

drugi komplement : 100111

Drugi komplement broja **011001** je **100111**.

Drugi komplement nekog binarnog broja je definisan kao broj, koji sabran sa originalnim brojem daje nulu i **prenos** (carry) jedan.

PRIMER 3.11:

Sabrati broj 011001 sa svojim drugim komplementom 100111.

011001
\+ 100111
1 000000 rezultat
prenos (carry)

Drugi komplement nekog broja smatraćemo njegovom negativnom vrednošću, pa je na taj način moguće realizovati oduzimanje putem sabiranja.

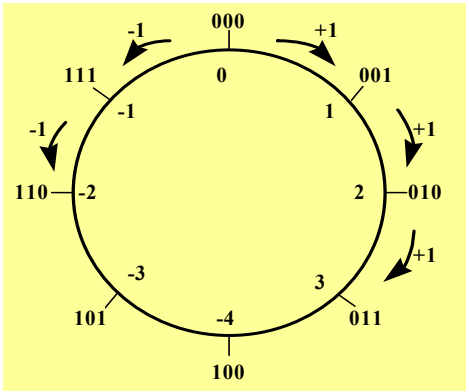
Negativne brojeve je moguće, pored drugog komplementa, predstaviti i pomoću predznaka i broja; jedinica na najznačajnijem mestu predstavlja negativan broj, a nula pozitivan.

U sledećoj tabeli (Tabela 3.3) su predstavljeni četvorobitni binarni brojevi sa svojim drugim komplementom i dekadnim ekvivalentnim brojem.

Tabela 3.3: binarni broj, dekadni ekvivalent i drugi komplement

BINARNI BROJ	DEKADNI EKVIVALENT	DRUGI KOMPLEMENT	DEKADNI EKVIVALENT
0000	0	0000	0
0001	1	1111	-1
0010	2	1110	-2
0011	3	1101	-3
0100	4	1100	-4
0101	5	1011	-5
0110	6	1010	-6
0111	7	1001	-7

Pomoću kruga je moguće predstaviti brojeve u intervalu od –max. do max., na primer od -4 do 3 u trobitnom obliku (Slika 3.2).



Slika 3.2: Trobitni binarni brojevi

Oduzimanje pomoću drugog komplementa je lako izvršiti: umanjeniku se dodaje drugi komplement umanjioca, i dobijeni algebarski zbir je tražena razlika. Poslednji prenos se odbacuje.

3.3.3.3 Množenje binarnih brojeva

Množenje binarnih brojeva je vrlo slično množenju dekadnih brojeva. Tu važe sledeća pravila :

0 * 0 = 0; 0 * 1 = 0; 1 * 0 = 0; 1 * 1 = 1;

PRIMER 3.12:

Izvršiti množenje brojeva 7 i 5 u binarnom sistemu.

$7 = 0111_2$
$5 = 0101_2$

7*5 = 35
111 * 101
111
000
111

100011 = 35

Množenje u elektronskim računarima se najčešće realizuje višestrukim sabiranjem.

3.3.3.4. Deljenje binarnih brojeva

Deljenje binarnih brojeva je istovetno kao i u dekadnom sistemu.

PRIMER 3.13:

Izvršiti deljenje brojeva 24 i 6 u binarnom sistemu.

$24 = 011000_2$ $6 = 000110_2$

11000 : 110 = 100
- 110
0000
-000
0000

Kao što je rečeno, računari poseduju sklop za sabiranje, te se tako i deljenje realizuje pomoću više uzastopnih oduzimanja, odnosno (pošto je oduzimanje moguće realizovati pomoću sabiranja) sabiranja.

3.3.4 BINARNI BROJEVI U TEHNICI

U tabeli 3.4 prikazani su prirodni binarni brojevi. Na primer, ako se želi napraviti pretvarač pomeranja (kada se upravlja nekom mašinom, brojevi, koji će značiti npr. put, koga treba da pređe radni organ neke mašine) koji na izlazu daje prirodni binarni kod, onda u slučaju prelaza 0 na jedan daje tačan rezultat, dok kod prelaza 1 na 2 za jedno kratko vreme može da se pojavi kod broja 3, jer nije moguće realizovati takav mehanički, ili elektronski sistem, gde će doći apsolutno u istom trenutku do menjanja binarne vrednosti. Zbog ove činjenice u tehnici za realizaciju takvih davača se koristi na primer Gray-ov kod (Tabela 3.2, desna kolona), gde kod svakog prelaza sa jedne vrednosti na drugu do promene se dolazi uvek isključivo na jednoj poziciji.

Tabela 3.4: binarni kodovi

decimalni ekvivalent	naziv koda	
	prirodni binarni	Gray-ov kod
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

Umesto prirodnog binarnog koda u tehnici za detekciju puta se koristi reflektirani binarni kod, za koji je drugi naziv Gray-ov kod. Karakteristika ovog binarnog koda je, da se, pri prelasku s jednog broja na drugi broj, koji mu je susedan (na primer pri prelasku sa 6 na 5 ili 7), menja se samo jedna kolona.

Gray-ov kod je binarni ciklički kod. Kod prirodnog binarnog koda cifre istog ranga u nizu, koje predstavljaju redosled brojeva,ređaju se po prirodnom poretku i svaka od njih se ponavlja saglasno težini odgovarajućeg ranga. Kod Gray-ovog koda se nizovi ponavljaju naizmenično u prirodnom i inverznom poretku. Simetričnost ponavljanja pojedinih cifara je sledeća:

- u odnosu na osu između 7 i 8 su poslednje tri cifre simetrično raspoređene,
- u odnosu na osu između 3 i 4, 7 i 8 i 11 i 12 simetrične su poslednje dve cifre i
- u odnosu na ose između 1 i 2, 3 i 4, 5 i 6, 7 i 8, 9 i 10, 11 i 12 i 13 i 14 su samo poslednje cifre simetrične.

U tabeli 3.4 se vidi to, ne samo numerički, već i grafički, pri čemu **1** prikazujemo crtom, dok te crte za **0** nema.

Decimalni ekvivalent bilo kojeg broja u Gray-ovom kodu možemo izračunati, ako ciframa dodajemo s desna na levo sledeće težinske vrednosti:

$$\mathbf{1, 3, 7, 15,.....}\, 2^{n+1} - 1 \mathbf{....,}$$

a proizvodima koji nisu nule dati alternativne predznake. Pri tome n odgovara rangu 0, 1, 2 itd.

PRIMER 3.14:

Pretvoriti **1 1 0 1** (Gray-ov kod) u decimalni ekvivalent.

```
   1  1   0   1
+15 -7  +0  +1  = 9
```

PRIMER 3.15:

Pretvoriti **1 0 1 1** (Gray-ov kod) u decimalni ekvivalent.

```
   1   0   1   1
+15  +0  -3  +1  = 13
```

Ako je broj dat u prirodnom binarnom kodu, onda pretvaranje u Gray-ov kod ide tako, što će cifra biti zapisana, ako je ispred te cifre **0**, odnosno cifra će biti komplementovana ako je ispred nje **1**.

PRIMER 3.16:

Broj 10011101000 (prirodni binarni kod) pretvoriti u Gray-ov ekvivalent:

prirodni binarni kod 1 0 0 1 1 1 0 1 0 0 0

reflektirani binarni (ili Gray-ov) kod 1 1 0 1 0 0 1 1 1 0 0.

PRIMER 3.17:

Broj 1101 (prirodni binarni kod) pretvoriti u Gray-ov ekvivalent:

prirodni binarni kod 1 1 0 1

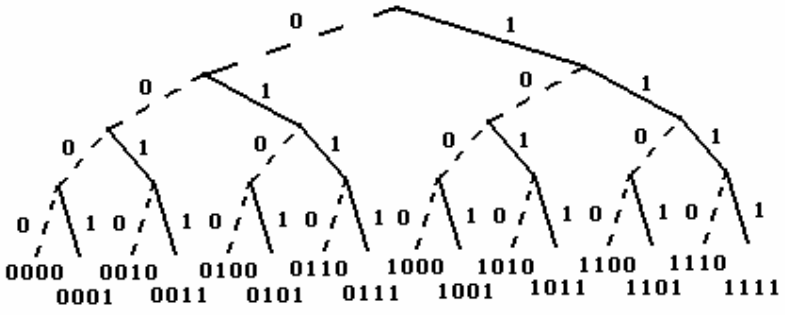
reflektirani binarni (ili Gray-ov) kod 1 0 1 1.

Na slici 3.3 i 3.4 se prikazuje još jedan način formiranja prirodnog, odnosno reflektiranog binarnog (Gray-ovog) koda.

Ako se pri svakom grananju držimo reda:

(0 1) (0 1) (0 1)

dobijamo brojeve u prirodnom binarnom kodu (Slika 3.3).

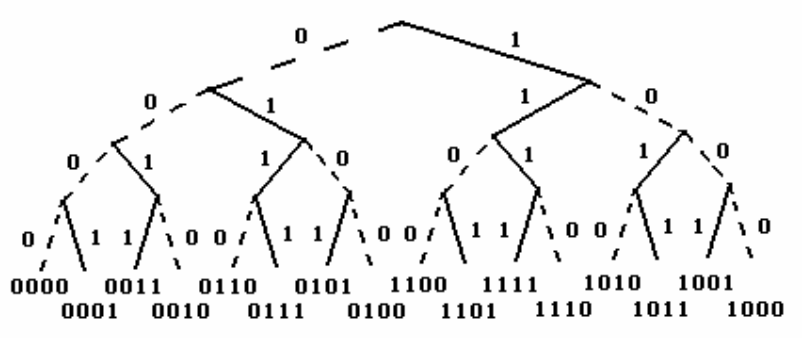


Slika 3.3: prirodni binarni kod

Ako se pri svakom grananju držimo reda :

(0 1) (1 0) (0 1)

dobijamo brojeve u Gray-ovom, tj. reflektiranom binarnom kodu (slika 3.4).



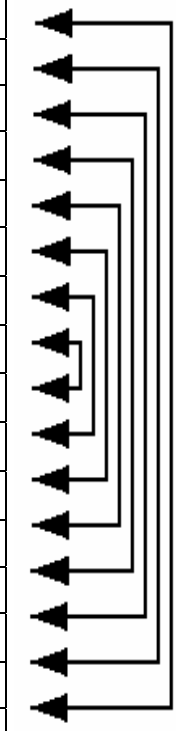
Slika 3.3: Gray-ov kod

4. LOGIČKE OSNOVE RADA DIGITALNIH SISTEMA

4.1 FUNKCIJE ALGEBRE LOGIKE

U algebri logike između dve logičke promenljive (A i B) možemo definisati ukupno $2^{2^2} = 16$ kombinacija (odnosno funkcija), kao što je to prikazano u Tabeli 4.1. Može se uočiti, da F_{15} je prvi komplement od F_0, F_{14} od F_1, ... i F_8 od F_7 (Slika 4.1).

		3	2	1	0
A	2^1	**1**	**1**	**0**	**0**
B	2^0	**1**	**0**	**1**	**0**
0.	F_0^2	**0**	**0**	**0**	**0**
1.	F_1^2	**0**	**0**	**0**	**1**
2.	F_2^2	**0**	**0**	**1**	**0**
3.	F_3^2	**0**	**0**	**1**	**1**
4.	F_4^2	**0**	**1**	**0**	**0**
5.	F_5^2	**0**	**1**	**0**	**1**
6.	F_6^2	**0**	**1**	**1**	**0**
7.	F_7^2	**0**	**1**	**1**	**1**
8.	F_8^2	**1**	**0**	**0**	**0**
9.	F_9^2	**1**	**0**	**0**	**1**
10.	F_{10}^2	**1**	**0**	**1**	**0**
11.	F_{11}^2	**1**	**0**	**1**	**1**
12.	F_{12}^2	**1**	**1**	**0**	**0**
13.	F_{13}^2	**1**	**1**	**0**	**1**
14.	F_{14}^2	**1**	**1**	**1**	**0**
15.	F_{15}^2	**1**	**1**	**1**	**1**

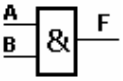
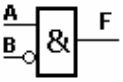
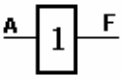
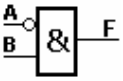
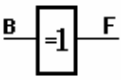
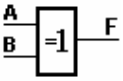
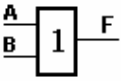
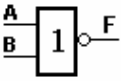
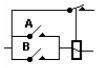
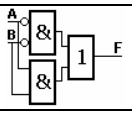
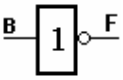
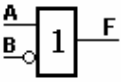
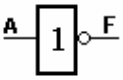
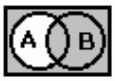
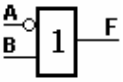


Slika 4.1: simetrija logičkih funkcija sa dve promenljive

U tabeli su i funkcije sa jednom promenljivom: F_0, F_3, F_5, F_{10}, F_{12} i F_{15}.

U daljem će se analizirati funkcije sa dve promenljive.

Tabela 4.1: kombinacije između dve promenljive

r. b.	A / B	0011 / 0101	funkcija F=f(AB)	naziv funkcije	predstavljanje funkcije: Wenn-ov dijagram	logički simbol	prekidačka šema
0.	F_0	0000	0	nula (apsolutna negacija)		nema	
1.	F_1	0001	$A \cdot B$	konjunkcija (I funkcija)			
2.	F_2	0010	$A \cdot \overline{B}$	inhibicija			
3.	F_3	0011	A	identitet za A			
4.	F_4	0100	$\overline{A} \cdot B$	inhibicija			
5.	F_5	0101	B	identitet za B			
6.	F_6	0110	$\overline{A} \cdot B + A \cdot \overline{B}$	antivalencija (ekskluzivno ILI)			
7.	F_7	0111	$A + B$	disjunkcija (logičko ILI)			
8.	F_8	1000	$\overline{(A + B)}$	Pierce funkcija (NILI)			
9.	F_9	1001	$\overline{A} \cdot \overline{B} + A \cdot B$	ekvivalencija			
10.	F_{10}	1010	$\overline{B}$	negacija za B			
11.	F_{11}	1011	$A + \overline{B}$	implikacija			
12.	F_{12}	1100	$\overline{A}$	negacija za A			
13.	F_{13}	1101	$\overline{A} + B$	implikacija			
14.	F_{14}	1110	$\overline{A \cdot B}$	Šeferova funkcija (NI)			
15.	F_{15}	1111	1	jedinica,apsolutna afirmacija		nema	

U algebri logike postoje tri osnovne operacije. Dve su binarne, a jedna je unarna.

Osnovne binarne operacije su:

- konjunkcija (I funkcija, logičko moženje) i
- disjunkcija (ILI funkcija, logičko sabiranje).

Osnovna unarna operacija je:

- negacija (poricanje).

Za binarne operacije potrebno je imati barem dve binarne algebarske veličine (binarni argurgemt), za logičku negaciju samu jednu.

Rezultat svake pojedinačne binarne operacije ili niza operacija ponovo je binarna veličina (0 ili 1). Pošto binarni argumenti mogu imati vrednost 0 ili 1, rezultat operacije sa binarnim argumentima jeste funkcija, jer zavisi od kombinacije vrednosti argumenata. Ova se funkcija može obeležiti na način uobičajen u matematici:

$$F = f(A, B, C, ...) \qquad (4.1)$$

gde su:

A,B,C,.... -ulazne veličine,
F -izlazna veličina.

Matematičku funkciju (4.1) možemo i simbolično prikazati blok šemom (slika 4.2).

Slika 4.2: simbolični prikaz funkcije $F = f(A, B, C, ...)$

4.1.1 I FUNKCIJA – KONJUNKCIJA, LOGIČKO MNOŽENJE

Ako su **A** i **B** dva suda, onda se pod sudom »**A** i **B**« podrazumeva tvrđenje da je sud »**A** i **B**« istinit samo ako je i sud **A** i sud **B** istinit. Znači **I** funkcija, odnosno konjunkcija je istinita samo ako su binarni argumenti **A** i **B** istiniti.

Nazivi za ovu funkciju su:

- **I** funkcija,
- konjunkcija i
- logičko množenje.

Koriste se simboli:

- AB ,
- $A \cdot B$,
- $A \,\&\, B$,
- $A \wedge B$.

Funkciju možemo prikazati pomoću tablice istine, vremenskog dijagrama, logičkog simbola, prekidačkog simbola i dijagrama toka (slika 4.3).

I funkcija: F = A B

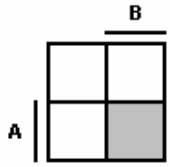
TABLICA ISTINE

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

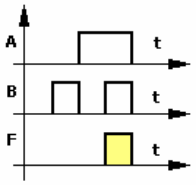
DIJAGRAMI

VENN

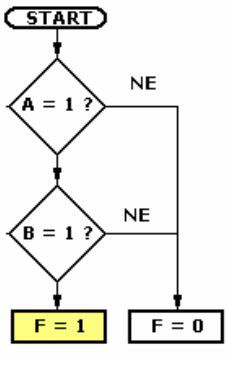
VEITCH



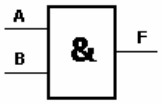
VREMENSKI



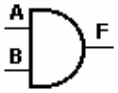
DIJAGRAM TOKA



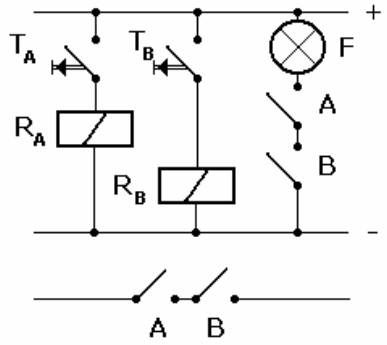
JUS SIMBOL



AMERIČ-KI SIM.



PREKIDAČKA REALIZACIJA



Slika 4.3: logička I funkcija

U ovom primeru ima N = 2 elemenata, koji mogu imati 2 vrednosti (**0** i **1**), i broj mogućih varijacija je $V = 2^N = 2^2 = 4$.

Konjunkcija između **A** i **B** (**I** funkcija) može se veoma slikovito predstaviti električnim prekidačima (slika 4.3). Kontakti **A** i **B** su iskazi (binarni argumenti). Ako je kontakt otvoren (**0**) odnosno zatvoren (**1**), tada je i strujni krug otvoren, odnosno zatvoren.

Sa tasterima (T_A, T_B), relejima (R_A, R_B), sijalicom (F) i napajanjem može se sastaviti strujni krug (slika 4.3), gde odgovarajući taster aktivira relej, a kontakt releja priključuje, odnosno isključuje napon.

U inženjerskoj praksi cilj je ostvariti što preglednij šemu uređaja. Za to služi šema sa logičkim simbolima. Ovi simboli omogućuju veliku preglednost u složenijim logičkim šemama. Sam simbol još ne prejudicira vrstu tehničkog sredstva (tehničkog rešenja), pomoću koga se željena logička funkcija treba ostvariti.

U primeru za logičku **I** funkciju smo uzeli dva iskaza, međutim to nije uvek slučaj. I funkcija može da ima minimum dva iskaza, a maksimalni broj iskaza nije ograničen.

PRIMER 4.1:

Na primer, u primeru 2.2 (poglavlje 2.1) **I** funkcija ima četiri ulaza (slika 4.4).

Slika 4.4: **I** funkcija sa 4 ulaza (primer 2.2)

Suštinu logike **I** funkcije možemo pratiti i na dijagramu toka (slika 4.3).

U tabeli 4.2 su dati zakoni relacijskih operacija **I** funkcije.

Tabela 4.2: zakoni **I** funkcije	
$A \cdot 1 = A$	
$A \cdot 0 = 0$	
$A \cdot A = A$	
$A \cdot \overline{A} = 0$	
$A \cdot B = B \cdot A$	komutativnost
$A \cdot (B \cdot C) = (A \cdot B) \cdot C = A \cdot B \cdot C$	asocijativnost
$\overline{A} \cdot 1 = \overline{A}$	
$\overline{A} \cdot 0 = 0$	
$\overline{A} \cdot \overline{A} = \overline{A}$	
$\overline{A} \cdot A = 0$	
$0 \cdot 0 = 0$	
$0 \cdot 1 = 0$	
$1 \cdot 0 = 0$	
$1 \cdot 1 = 1$	

Treba napomenuti, da u opštem slučaju **I** funkciju možemo proširiti i na proizvoljan broj promenljivih, tj. važi da je:

$$F = X_1 \cdot X_2 \cdot .....X_n \qquad (4.2)$$

gde su:

- F -funkcija,
- X_n -ulaz i
- n -indeks ulaza.

Isto tako je moguće **I** funkciju primeniti i za bajtove, kao i za reči. Ova primena se koristi u programiranju, na pr. na najnižem nivou programiranja, u asembleru su definisane neke logičke operacije, među ostalog i **I** funkcija. U nekim višim programskim jezicima takođe srećemo kompajlerske ili interpreterske funkcije koje realizuju **I** operaciju.

PRIMER 4.2

Pomoću ANA B instrukcije odrediti I funkciju između sadržaja A registara (akumulatora) i B registara. Sadržaj A je $A = C3_{16}$, a sadržaj B je $B = 8F_{16}$.

$$A = C3_{16} = 11000011_2$$
$$B = 8F_{16} = 10001111_2$$
$$\overline{10000011_2} = 83_{16}$$

4.1.2 ILI FUNKCIJA, DISJUNKCIJA, LOGIČKO SABIRANJE

Ako su **A** i **B** dva suda, onda se pod sudom »**A** ili **B**« podrazumeva tvrđenje, da važi bilo sud **A**, bilo sud **B**, bilo istovremeno oba. Znači izlaz je jedinica (**1**), ako je ili **A** jedinica (**1**) ili **B** jedinica (**1**), ili su oba jedinice.

Nazivi za ovu funkciju su:

- **ILI** funkcija,
- disjunkcija,
- logičko sabiranje i
- inkluzivno **ILI**.

Koriste se simboli:

- $A + B$,
- $A \vee B$.

Funkcija je istinita, **1** je, onda ako je istinito **A**, ili ako je istinito **B**, ili ako su **A** i **B** istinita. Funkciju možemo prikazati pomoću tablice istine, vremenskog dijagrama, logičkog simbola, prekidačkog simbola i dijagrma toka (slika 4.5).

ILI funkcija: F = A+B

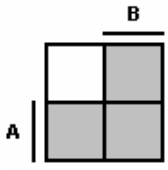
TABLICA ISTINE

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

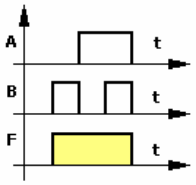
DIJAGRAMI

VENN

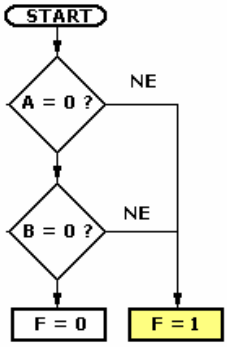
VEITCH



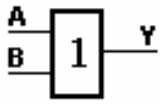
VREMENSKI



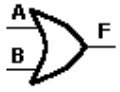
DIJAGRAM TOKA



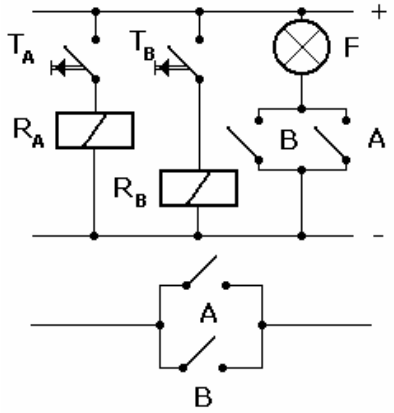
JUS SIMBOL



AMERIČ-KI SIM.



PREKIDAČKA REALIZACIJA



Slika 4.5: logička **ILI** funkcija

U tabeli 4.3 su dati zakoni relacijskih operacija **ILI** funkcije.

Tabela 4.3: zakoni ILI funkcije	
$A+1=1$	
$A+0=A$	
$A+A=A$	
$A+\overline{A}=1$	
$A+B=B+A$	komutativnost
$A+(B+C)=(A+B)+C=A+B+C$	asocijativnost
$\overline{A}+1=1$	
$\overline{A}+0=\overline{A}$	
$\overline{A}+\overline{A}=\overline{A}$	
$\overline{A}+A=1$	
$0+0=0$	
$0+1=1$	
$1+0=1$	
$1+1=1$	

Treba napomenuti, da u opštem slučaju **ILI** funkciju možemo proširiti i na proizvoljan broj promenljivih:

$$F = X_1 + X_2 + ...X_n \qquad (4.3)$$

gde su:

- F -funkcija,
- X_n -ulaz i
- n -indeks ulaza.

Isto tako je moguće ILI funkciju primeniti za bajtove kao i za reči.

☺ ☹ ☺

PRIMER 4.3

Pomoću ORA B instrukcije odrediti ILI funkciju između sadržaja A registra (akumulatora) i B registara. Sadržaj A je $A = C3_{16}$, a sadržaj B je $B = 8F_{16}$.

$$A = C3_{16} = 11000011_2$$
$$B = 8F_{16} = 10001111_2$$
$$\overline{11001111_2} = CF_{16}$$

☺ ☹ ☺

4.1.3 NEGACIJA (INVERZIJA)

Negacija suda (operacija inverzije) ne vrši se bezuslovno nad nizom veličina, već se može vršiti i nad jednom veličinom. Veličina može da bude ili prosta promenljiva, ili je funkcija izvesnog broja drugih binarnih promenljivih.

Nazivi za ovu funkciju su:

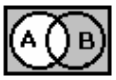
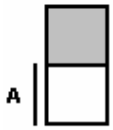
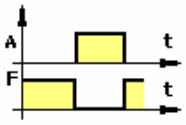
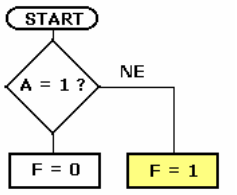
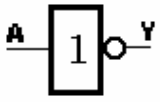
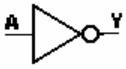
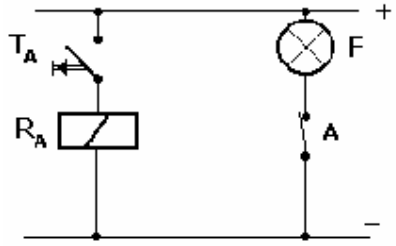
- negacija,
- inverzija.

Koriste se simboli:

- $\overline{A}$ i
- $-A$.

Funkcija je istinita (**F=1**), onda ako je **A** neistinito (**A=0**). Funkciju možemo prikazati pomoću tablice istine, vremenskog dijagrama, logičkog simbola, prekidačkog simbola i dijagrama toka (slika 4.6).

NE funkcija: $F = \overline{A}$

TABLICA ISTINE			DIJAGRAMI			
			VENN	VEITCH	VREMENSKI	DIJAGRAM TOKA
A	**F**					
0	**1**					
1	**0**					
JUS SIMBOL			AMERIČ-KI SIM.	PREKIDAČKA REALIZACIJA		
						

Slika 4.6: logička **NE** funkcija

Na slici 4.6 je prikazana i prekidačka šema, gde se vidi, da sijalica svetli, kada taster T_A nije aktivan, i obratno, čim se taster T_A aktivira, sijalica se gasi. Releji imaju mirni kontakt, koji takođe deluju na prethodnom principu.

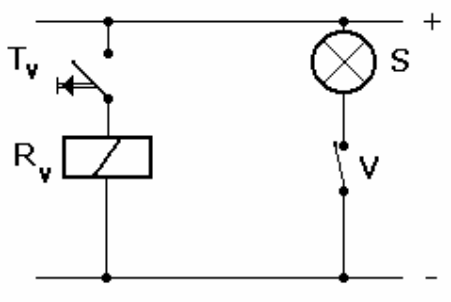
PRIMER 4.4:

Projektovati za hladnjak (frižider) osvetljenje, kada su vrata otvorena, sijalica u frižideru svetli, a kada su vrata zatvorena sijalica se gasi.

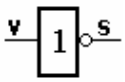
Ovde promenljiva **V** (vrata) je jedinica (**1**), dok su vrata otvorena, a u suprotnom je nula (**0**). Događaj otvoreno-zatvoreno možemo detektovati sa tasterom (krajnjim prekidačem) T_V, a sijalicu ćemo označiti sa **S**. Tablica istine je na slici 4.7, a kompletna prekidačka šema je na slici 4.8, odnosno logički simbol na slici 4.9.

redni broj	V	T_V	S
0.	0	0	1
1.	1	1	0

Slika 4.7: tablica istine osvetljenja unutrašnjosti frižidera



Slika 4.8: prekidačka šema rešenja



Slika 4.9: logički simbol

4.1.4 NI FUNKCIJA

Ako se operacija negacije (inverzije) primeni na **I** funkciju dobija se **NI** funkcija (skraćenica od **NE-I** kombinacije). Tehnički se realizuje jednostavno, univerzalno se može koristiti, pa zbog toga ima široku primenu u gradnji elektronskih sklopova.

Logički obrazac za **NI** funkciju je:

$$F = \overline{A \cdot B} \qquad (4.4)$$

Koriste se simboli:

- $\overline{AB}$,
- $\overline{A \cdot B}$,
- $\overline{A \,\&\, B}$,
- $\overline{A \wedge B}$.

NI funkcija: $F = \overline{A \cdot B}$

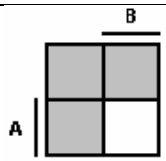
TABLICA ISTINE

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

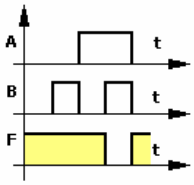
DIJAGRAMI

VENN

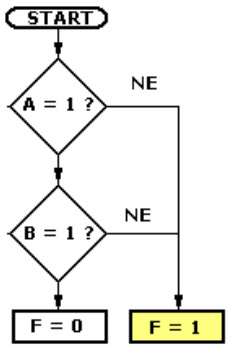
VEITCH



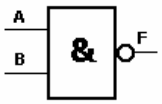
VREMENSKI



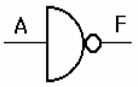
DIJAGRAM TOKA



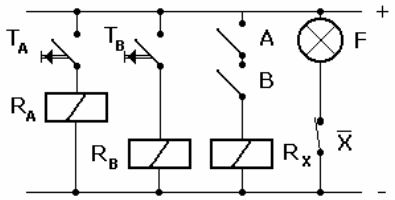
JUS SIMBOL



AMERIČ-KI SIM.



PREKIDAČKA REALIZACIJA



Slika 4.10: NI funkcija

Funkcija je neistinita **0** (nula), onda ako je istinito **A**, i ako je istinito **B**, ili ako su **A** i **B** istinita. Funkciju možemo prikazati pomoću tablice istine, vremenskog dijagrama, logičkog simbola, prekidačkog simbola i dijagrma toka (slika 4.10).

Iz tablice istine **NI** fukcije (slika 4.10) se vidi da su vrednosti **F** za pojedine redove suprotne vrednostima **I** funkcije (slika 4.3).

U tabeli 4.4 su dati zakoni operacja **NI** funkcije.

Tabela 4.4: zakoni **NI** funkcije	
$\overline{A \cdot 1} = \overline{A}$	
$\overline{A \cdot 0} = 1$	
$\overline{A \cdot A} = \overline{A}$	
$\overline{A \cdot \overline{A}} = 1$	
$\overline{A \cdot B} = \overline{B \cdot A}$	komutativnost
$\overline{A \cdot (B \cdot C)} = \overline{(A \cdot B) \cdot C} = \overline{A \cdot B \cdot C}$	asocijativnost
$\overline{\overline{A} \cdot 1} = A$	
$\overline{\overline{A} \cdot 0} = 1$	
$\overline{\overline{A} \cdot \overline{A}} = A$	
$\overline{\overline{A} \cdot A} = 1$	
$\overline{0 \cdot 0} = 1$	
$\overline{0 \cdot 1} = 1$	
$\overline{1 \cdot 0} = 1$	
$\overline{1 \cdot 1} = 0$	

Treba napomenuti, da **NI** funkciju možemo proširiti na proizvoljan broj promenljivih:

$$F = \overline{X_1 \cdot X_2 \cdot .....X_n} \qquad (4.5)$$

4.1.5 NILI FUNKCIJA

Ako se operacija negacije (inverzije) primeni na **ILI** funkciju dobija se **NILI** funkcija (skraćenica od **NE-ILI** kombinacije). Tehnički se realizuje jednostavno, univerzalno se može koristiti, pa zbog toga ima široku primenu u gradnji elektronskih sklopova.

Logički obrazac za **NILI** funkciju je:

$$F = \overline{A + B} \qquad (4.6)$$

NILI funkcija: $F = \overline{A + B}$

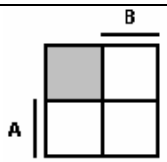
TABLICA ISTINE

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

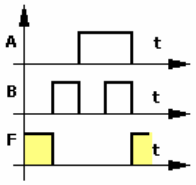
DIJAGRAMI

VENN

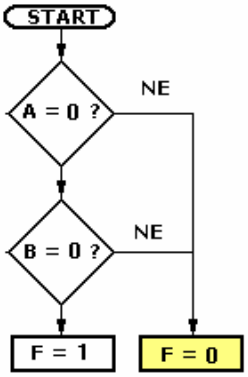
VEITCH



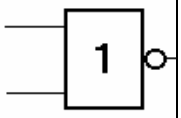
VREMENSKI



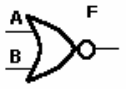
DIJAGRAM TOKA



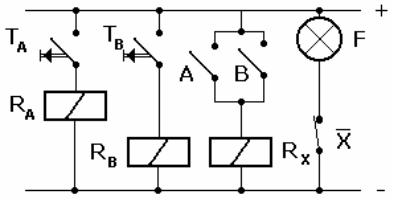
JUS SIMBOL



AMERIČ-KI SIM.



PREKIDAČKA REALIZACIJA



Slika 4.11: **NILI** funkcija

Funkcija je neistinita, (**0**), onda ako je istinito **A**, i ako je istinito **B**, ili ako su **A** i **B** istinita. Funkciju možemo prikazati pomoću tablice istine, vremenskog dijagrama, logičkog simbola, prekidačkog simbola i dijagrama toka (slika 4.11).

Iz tablice istine **NILI** fukcije (slika 4.11) se vidi da su vrednosti **F** za pojedine redove suprotne vrednostima **ILI** funkcije (slika 4.5).

Treba napomenuti, da NILI funkciju možemo proširiti na proizvoljan broj promenljivih:

$$F = \overline{X_1 + X_2 + ...X_n} \qquad (4.7)$$

U tabeli 4.5 su dati zakoni operacja ILI funkcije.

Tabela 4.5: zakoni NILI funkcije	
$\overline{A+1}=0$	
$\overline{A+0}=\overline{A}$	
$\overline{A+A}=\overline{A}$	
$\overline{A+\overline{A}}=0$	
$\overline{A+B}=\overline{B+A}$	komutativnost
$\overline{A+(B+C)}=\overline{(A+B)+C}=\overline{A+B+C}$	asocijativnost
$\overline{\overline{A}+1}=0$	
$\overline{\overline{A}+0}=A$	
$\overline{\overline{A}+\overline{A}}=A$	
$\overline{\overline{A}+A}=0$	
$\overline{0+0}=1$	
$\overline{0+1}=0$	
$\overline{1+0}=0$	
$\overline{1+1}=0$	

5. OSNOVNI ZAKONI BOOLE-OVE ALGEBRE

5.1 UVOD

Kao što smo već videli (4. poglavlje) pomoću tri osnovne logičke funkcije (**I**, **ILI** i **NE**) mogućno je dobiti sve moguće logičke funkcije (4. poglavlje, tabela br. 4.2).

5.2 ZAKONI ALGEBRE LOGIKE

5.2.1 OSNOVNI ZAKONI

Za operaciju logičke **I** funkcije kao i za **ILI** funkciju imamo četiri zakona:

1. **Zakon komutacije:**

$$A \cdot B = B \cdot A \qquad (5.1)$$
$$A + B = B + A \qquad (5.2)$$

2. **Zakon distribucije:**

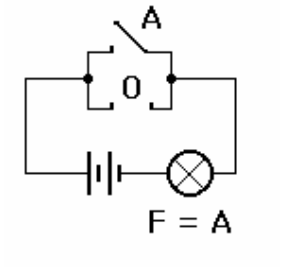
$$(A + B) \cdot C = A \cdot C + B \cdot C \qquad (5.3)$$
$$(A \cdot B) \cdot C = (A + C) \cdot (B + C) \qquad (5.4)$$

3. **Uticaj nule i jedinice:**

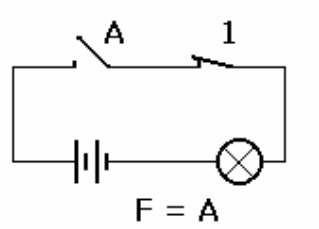
$$A + 0 = A \quad \text{(slika 5.1)} \qquad (5.5)$$
$$A \cdot 1 = A \quad \text{(slika 5.2)} \qquad (5.6)$$

4. **Uticaj komplmenta:**

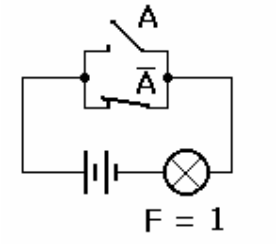
$$A + \overline{A} = A \quad \text{(slika 5.3)} \qquad (5.7)$$
$$A \cdot \overline{A} = 0 \quad \text{(slika 5.4)} \qquad (5.8)$$



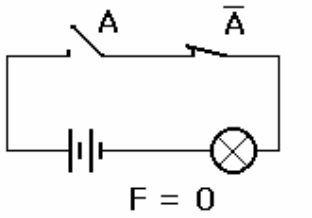
Slika 5.1: uticaj nule



Slika 5.2: uticaj jedinice



Slika 5.3: uticaj komplementa kod **ILI**



Slika 5.4: uticaj komplementa kod **I**

Zakoni 1, 3 i 4 su očigledni, prikazani su na slikama 5.1, 5.2, 5.3 i 5.4, a za dokazivanje 2 zakona koristimo tablicu istine (slika 5.5 i slika 5.6).

r.b.	A	B	C	$(A+B)\cdot C$	$A\cdot C+B\cdot C$
0.	**0**	**0**	**0**	**0**	**0**
1.	**0**	**0**	**1**	**0**	**0**
2.	**0**	**1**	**0**	**0**	**0**
3.	**0**	**1**	**1**	**1**	**1**
4.	**1**	**0**	**0**	**0**	**0**
5.	**1**	**0**	**1**	**1**	**1**
6.	**1**	**1**	**0**	**0**	**0**
7.	**1**	**1**	**1**	**1**	**1**

Slika 5.5: dokazivanje zakona distribucije pomoću tablice istine

r.b.	A	B	C	$(A\cdot B)+C$	$(A+C)\cdot(B+C)$
0.	**0**	**0**	**0**	**0**	**0**
1.	**0**	**0**	**1**	**1**	**1**
2.	**0**	**1**	**0**	**0**	**0**
3.	**0**	**1**	**1**	**1**	**1**
4.	**1**	**0**	**0**	**0**	**0**
5.	**1**	**0**	**1**	**1**	**1**
6.	**1**	**1**	**0**	**1**	**1**
7.	**1**	**1**	**1**	**1**	**1**

Slika 5.6: dokazivanje zakona distribucije pomoću tablice istine

Na osnovu četiri prethodna osnovna zakona se baziraju sledeći zakoni:

- **Princip dualnosti:** uz svaki logički iskaz može se dati njemu pripadajući dualni iskaz putem sledećih koraka:

  a) međusobno zameniti znake » **.** « (logičko množenje) i » + « (logičko sabiranje),
  b) međusobno zameniti **0** (nulu) i **1** (jedinicu) i
  c) umesto svake promenljive staviti njenu negiranu vrednost.

PRIMER 5.1:

za $A=1$ je $\overline{A}=0$.

PRIMER 5.2:

za $C = A \cdot \overline{B} + \overline{A} \cdot B$ je $\overline{C} = (\overline{\overline{A}} + \overline{\overline{B}}) \cdot (\overline{\overline{A}} + \overline{B}) = (\overline{\overline{A}} + B) \cdot (A + \overline{B})$.

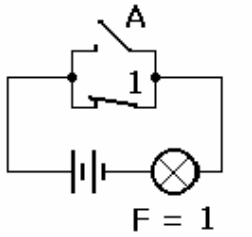
Pri tome je $\overline{0} = 1$ i $\overline{1} = 0$.

- **Stav o tautologiji:**

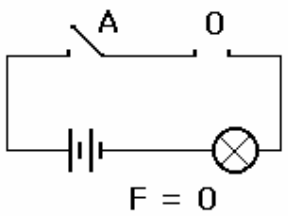
$$A + A = A \qquad (5.9)$$
$$A \cdot A = A \qquad (5.10)$$

- **Stav o uticaju jedinice (1) i nule (0):**

$$A + 1 = 1 \quad \text{(slika 5.7)} \qquad (5.11)$$
$$A \cdot 0 = 0 \quad \text{(slika 5.8)} \qquad (5.12)$$



Slika 5.7: uticaj jedinice (**1**)



Slika 5.8: uticaj nule (**0**)

- **Zakon apsorbcije:**

$$A + A \cdot B = A. \qquad (5.13)$$

Dokaz teoreme: $A + A \cdot B = A \cdot (1 + B) = A \cdot 1 = A$.

$$A \cdot (A + B) = A. \qquad (5.14)$$

Dokaz teoreme: $A \cdot (A + B) = A \cdot A + A \cdot B = A + A \cdot B = A$.

- **Zakon dvostruke negacije:**

Prema ovom zakonu, ako negiramo negiranu vrednost promenljive, opet ćemo dobiti originalnu vrednost promenljive.

$$\overline{(\overline{A})} = \overline{\overline{A}} = A. \qquad (5.15)$$

Isto tako, za nulu (0) i jedinicu (1) važi:

$$\overline{\overline{0}} = 0 \text{ i } \overline{\overline{1}} = 1 .$$

Sa trostrukom negacijom menjamo vrednost promenljive:

$$\overline{\overline{\overline{1}}} = \overline{1} = 0 .$$

Ovakav zakon imamo i u govoru, na primer »nije neispravan« znači da je ispravan.

- **De Morganovi zakoni:**

$$\overline{A \cdot B} = \overline{A} + \overline{B} \qquad (5.16)$$
$$\overline{A + B} = \overline{A} \cdot \overline{B} \qquad (5.17)$$

Gornji zakoni (5.16) i (5.17) su veoma važni zakoni, lako možemo koristeći njih pretvoriti I, ILI, NI, NILI i NE funkcije u neke druge funkcije.

Treba napomenuti, da ova dva zakona važe ne samo za dve promenljive **A** i **B**, nego i za proizvoljan broj istih.

Dokazati De Morganov zakon $\overline{A \cdot B} = \overline{A} + \overline{B}$ možemo sa tablicom istine (slika 5.9). Pošto je vrednost 4. kolone $\overline{A \cdot B}$ jednaka vrednosti sedme kolone $\overline{A} + \overline{B}$, zakon $\overline{A \cdot B} = \overline{A} + \overline{B}$ je dokazan.

r.b.	A	B	$A \cdot B$	$\overline{A \cdot B}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
	1.	**2.**	**3.**	**4.**	**5.**	**6.**	**7.**
0.	**0**	**0**	**0**	**1**	**1**	**1**	**1**
1.	**0**	**1**	**0**	**1**	**1**	**0**	**1**
2.	**1**	**0**	**0**	**1**	**0**	**1**	**1**
3.	**1**	**1**	**1**	**0**	**0**	**0**	**0**

Slika 5.9: dokaz De Morgan-ove teoreme $\overline{A \cdot B} = \overline{A} + \overline{B}$

Dokazati De Morganov zakon $\overline{A+B}=\overline{A}\cdot\overline{B}$ možemo sa tablicom istine (slika 5.10). Pošto je vrednost 5. kolone $\overline{A}\cdot\overline{B}$ jednaka vrednosti sedme kolone $\overline{A+B}$, zakon $\overline{A+B}=\overline{A}\cdot\overline{B}$ je dokazan.

r.b.	A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$	$A+B$	$\overline{A+B}$
	1.	**2.**	**3.**	**4.**	**5.**	**6.**	**7.**
0.	**0**	**0**	**1**	**1**	**1**	**0**	**1**
1.	**0**	**1**	**1**	**0**	**0**	**1**	**0**
2.	**1**	**0**	**0**	**1**	**0**	**1**	**0**
3.	**1**	**1**	**0**	**0**	**0**	**1**	**0**

Slika 5.10: dokaz De Morgan-ove teoreme $\overline{A+B}=\overline{A}\cdot\overline{B}$

De Morgan-ove teoreme možemo dokazati matematički, ali možemo i preko primera iz drugog poglavlja (primer 2.2).

☺ ☹ ☺

PRIMER 5.3:

Indirektno možemo na ovom primeru dokazati de Morgan-ove teoreme. Presa može da radi (**F**), ako su ispunjena četiri uslova, a to su:

- (**A**) pedala za pokretanje pritisnuta,
- (**B**) prethodno izrađeni otpresak izvađen iz kalupa,
- (**C**) postavljen je novi materijal u kalup i
- (**D**) ako je zaštitna rešetka spuštena.

Već smo videli, da **I** logičko kolo realizuje funkciju:

$$F = A\cdot B\cdot C\cdot D \qquad (5.18)$$

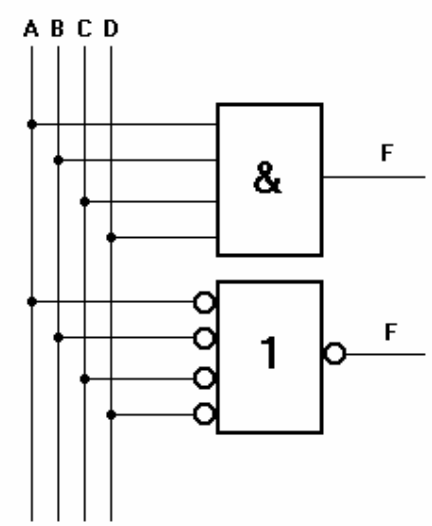
Ako negiramo funkciju i primenimo De Morgan-ov zakon, dobijamo izraz:

$$\overline{F}=\overline{A\cdot B\cdot C\cdot D}=\overline{A}+\overline{B}+\overline{C}+\overline{D}\,. \qquad (5.19)$$

Analizirajući dobijenu funkciju, možemo da kažemo sledeće: presa se ne sme uključiti ($\overline{F}$), ako:
($\overline{A}$) pedala za uključivanje prese nije pritisnuta ili
($\overline{B}$) prethodno izrađeni otpresak nije izvađen iz kalupa ili
($\overline{C}$) nije postavljen novi materijal u kalup ili
($\overline{D}$) zaštitna rešetka nije spuštena.

Na slici 5.11 je dato originalno rešenje zadatka sa **I** kolom, a dato i rešenje sa **NILI** kolom, gde su i ulazi invertovani. U ovom slučaju (5.19) ima oblik:

$$F = \overline{\overline{A} + \overline{B} + \overline{C} + \overline{D}}\,. \qquad (5.20)$$



Slika 5.11: rešenje zadatka sa **I** i sa **NILI** kolima

☺ ☹ ☺

Posle izloženog uočimo sledeća četiri izraza:

1. $X \cdot 1 = X$, jedan (**1**) je neutralni element za logičko množenje,
2. $X + 0 = X$, nula (**0**) je neutralni element za logičko sabiranje,
3. $X + 1 = 1$, (jedan je nul-element za logičko sabiranje, i
4. $X \cdot 0 = 0$, (nula je nul-lement za logičko mnženje.

Tablični prikaz ovih relacija je u tabeli 5.1.

Tabela 5.1: tablični prikaz relacija 1 do 4		
operacija	jedan (**1**)	nula (**0**)
logičko sabiranje	nul-element	neutralni-element
logičko množenje	neutralni-element	nul-element

Pod neutralnim se podrazumeva takav element, koji ne menja funkciju, tj. ne utiče na nju. Nul-element ja pak takav, koji funkciji nameće svoju vrednost, anulirajući pri tome stanje ostalih elemenata.

Prethodni zakoni prekidačke logike su u tabeli 5.2, gde se vidi zakon dualnosti.

Tabela 5.2: zakoni prekidačke logike

br.	zakon	br.	zakon	napomena
1a	$\overline{0}=1$	1b	$\overline{1}=0$	
2a	$\overline{\overline{A}}=A$	2b	$\overline{\overline{\overline{A}}}=\overline{A}$	
3a	$A+1=1$	3b	$A\cdot 0=0$	nultost
4a	$A+0=A$	4b	$A\cdot 1=A$	neutralnost
5a	$A+A=A$	5b	$A\cdot A=A$	idenpotentost
6a	$A+\overline{A}=1$	6b	$A\cdot\overline{A}=0$	uticaj komplementa
7a	$A+B=B+A$	7b	$A\cdot B=B\cdot A$	komutativnost
8a	$A+B+C=A+(B+C)=(A+B)+C$	8b	$A\cdot B\cdot C=A\cdot(B\cdot C)=(A\cdot B)\cdot C$	asocijativnost
9a	$A\cdot B+A\cdot C=A\cdot(B+C)$	9b	$(A+B)\cdot(A+C)=A+B\cdot C$	distributivnost
10a	$A+A\cdot B=A$	10b	$A\cdot(A+B)=A$	apsorbcija
11a	$A\cdot(\overline{A}+B)=A\cdot B$	11b	$A+\overline{A}\cdot B=A+B$	
12a	$(A+B)\cdot(A+\overline{B})=A$	12b	$A\cdot B+A\cdot\overline{B}=A$	
13a	$\overline{A+B}=\overline{A}\cdot\overline{B}$	13b	$\overline{A\cdot B}=\overline{A}+\overline{B}$	De Morgan
14	$\overline{f}(X,Y,Z,...+,\cdot)=f(\overline{X},\overline{Y},\overline{Z},...,\cdot,+)$			Shannon

5.2.2 MINTERMI I MAKSTERMI

A, B, C,....N su nezavisne promenljive u Boole-ovoj algebri. Mintermima nazivamo sve one članove-proizvode, koji sadrže sve promenljive. Ako je ukupan broj promenljivi **n**, imaćemo 2^n takvih članova:

$$
\begin{aligned}
m_0 &= \overline{A}\cdot\overline{B}\cdot\overline{C}.....\overline{N}\\
m_1 &= \overline{A}\cdot\overline{B}\cdot\overline{C}.....N\\
&.....\\
&.....\\
m_{2^n-1} &= A\cdot B\cdot C....N
\end{aligned}
\qquad (5.21)
$$

PRIMER 5.4:

Treba odrediti sve minterme za promenljive **A**, **B**, **C** i **D** (n=4).

$$
\begin{aligned}
m_0 &= \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D} = 0\cdot 0\cdot 0\cdot 0\\
m_1 &= \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot D = 0\cdot 0\cdot 0\cdot 1\\
m_2 &= \overline{A}\cdot\overline{B}\cdot C\cdot\overline{D} = 0\cdot 0\cdot 1\cdot 0\\
m_3 &= \overline{A}\cdot\overline{B}\cdot C\cdot D = 0\cdot 0\cdot 1\cdot 1\\
m_4 &= \overline{A}\cdot B\cdot\overline{C}\cdot\overline{D} = 0\cdot 1\cdot 0\cdot 0\\
m_5 &= \overline{A}\cdot B\cdot\overline{C}\cdot D = 0\cdot 1\cdot 0\cdot 1\\
m_6 &= \overline{A}\cdot B\cdot C\cdot\overline{D} = 0\cdot 1\cdot 1\cdot 0\\
m_7 &= \overline{A}\cdot B\cdot C\cdot D = 0\cdot 1\cdot 1\cdot 1\\
m_8 &= A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D} = 1\cdot 0\cdot 0\cdot 0\\
m_9 &= A\cdot\overline{B}\cdot\overline{C}\cdot D = 1\cdot 0\cdot 0\cdot 1\\
m_{10} &= A\cdot\overline{B}\cdot C\cdot\overline{D} = 1\cdot 0\cdot 1\cdot 0\\
m_{11} &= A\cdot\overline{B}\cdot C\cdot D = 1\cdot 0\cdot 1\cdot 1\\
m_{12} &= A\cdot B\cdot\overline{C}\cdot\overline{D} = 1\cdot 1\cdot 0\cdot 0\\
m_{13} &= A\cdot B\cdot\overline{C}\cdot D = 1\cdot 1\cdot 0\cdot 1\\
m_{14} &= A\cdot B\cdot C\cdot\overline{D} = 1\cdot 1\cdot 1\cdot 0\\
m_{15} &= A\cdot B\cdot C\cdot D = 1\cdot 1\cdot 1\cdot 1
\end{aligned}
\qquad (5.22)
$$

Indeks iza slova **m** je redni broj minterma, a dobija se tako što se umesto afirmacije odgovarajuće promenljive stavi jedan (**1**) a umesto negacije odgovarajuće promenljive nula (**0**). Kombinacija jedinica i nula daje binarni broj, koji predstavlja indeks minterma.

Pošto u svakom mintermu imamo proizvod svih promenljivih, zovemo ih još i potpunom ili perfektnom konjukcijom.

Ako primenimo De Morgan-ovog zakona na minterme dobija se :

$$\begin{aligned}
&\overline{m}_0 = A + B + C + ... + J + K + N = M_{2^N=1} \\
&\overline{m}_1 = A + B + C + ... + J + K + \overline{N} = M_{2^n-2} \\
&\overline{m}_2 = A + B + C + ... + J + \overline{K} + N = M_{2^n-3} \\
&\overline{m}_3 = ... \\
&..... \\
&\overline{m}_{2^n-1} = \overline{A} + \overline{B} + \overline{C} + ... + \overline{J} + \overline{K} + \overline{N} = M_0
\end{aligned} \qquad (5.23)$$

Izrazi M_i su makstermi, ili ih zovemo potpunom (perfektnom) disjunkcijom. Oni sadrže sve promenljive u disjunktivnom obliku. Veza između minterma i maksterma je:

$$m_i = M_{2^n-i-1} \qquad (5.24)$$

gde je:

$i = 0,1,2,3,....2^n - 1$.

STAV 1:

Zbir svih minterma je jedinica (**1**), odnosno $\sum_{i=0}^{i=2^n-1} m_i = 1$.

Susedni mintermi se razlikuju samo u pogledu vrednosti jedne promenljive. Ta jedna promenljiva se javlja u jednom mintermu u afirmaciji, dok u drugom mintermu je u negaciji. Pri tome se putem postupka

$$A \cdot B + A \cdot \overline{B} = A \cdot (B + \overline{B}) = A \cdot 1 = A .$$

eliminiše jedna promenljiva. To se na isti način nastavlja do kraja, kada je najzad $A + \overline{A} = 1$.

STAV 2:

Proizvod svih maksterma je nula (0), odnosno $\prod_{i=0}^{i=2^n-1} M_i = 0$.

Ovaj stav sledi iz primene De Morgan-ovog stava na izraz $\sum_{i=0}^{i=2^n-1} m_i = 1$, t.j. biće $\sum_{i=0}^{i=2^n-1} m_i = \prod_{i=0}^{i=2^n-1} \overline{m_i} = \bar{1} = 0$.

Pošto ovaj izraz važi za svaki indeks **i**, važiće takđe $\prod_{i=0}^{i=2^n-1} M_i = 0$, što je i trebalo dokazati.

STAV 3:

Vrednost svih minterma je **0**, ako je jedan minterm **1**.

U svim mintermima, osim jednog minterma, barem jedan član je nula, a pošto u mintermu je proizvod, mintermi su nule.

6. IZNALAŽENJE FUNKCIJE

6.1 UVOD

U običnoj algebri ako je na primer dat izraz $y = 3 \cdot x^3 - 5 \cdot x^2 + x$ veoma je jednostavno za razne vrednosti nezavisne promenljive x izračunati vrednost funkcije y. Posle toga možemo sastaviti odgovarajuću tablicu parova x – y. Iz ove tablice možemo nacrtati i graf date funkcije.

Znatno je teže na osnovu postojećeg grafa (ili tablice) formirati analitički izraz $y = f(x)$.

6.2 GENEZA LOGIČKE FUNKCIJE

U algebri logike imamo sličnu, analognu situaciju, kao u algebri. Lako je na osnovu datog izraza $Y = f(A,B,C,.....Z)$ sačiniti tablicu istine (tablicu stanja). Zadatak je teži, ako treba rešiti neki problem upravljanja, u kome treba da se realizuju neki uslovi, karakteristični za dati slučaj.

Analizom postavljenih uslova se formira tablica istine na osnovu koje treba realizovati neko upravljačko kolo. Za realizaciju treba odrediti analitički izraz tražene logičke funkcije polazeći od tablice istine. Treba iznaći onu funkciju, koja zadovoljava date uslove, t.j. datu tablicu istine. Kako se iznalaženje funkcije svodi na njeno stvaranje, odnosno rađanje, postupak se zove geneza (postanak) funkcije.

PRIMER 6.1:

U jednom odboru (žiriju), koji treba da donese odluku po nekom pitanju, imamo tri člana: **A**, **B** i **C**. Odluka je punovažna, ako većina glasa za istu. Ovo će se desiti, ako bilo koja dvojica (**A** i **B**, **A** i **C**, odnosno **B** i **C**), ili pak trojica (**A** i **B** i **C**) glasati za. U tablici istine glas ćemo označiti sa jedinicom, protiv sa nulom.

Treba napomenuti, da težinski faktori promenljive su svi jedinice, a ne kao kod prirodnog binarnog broja 2^n. Tablica istine glasanja žirija je na slici 6.1.

Tabela 6.1: tablica istine glasanja žirija

	A	**B**	**C**	**F**
tež. fakt. ➔	**1**	**1**	**1**	
0.	**0**	**0**	**0**	**0**
1.	**0**	**0**	**1**	**0**
2.	**0**	**1**	**0**	**0**
3.	**0**	**1**	**1**	**1**
4.	**1**	**0**	**0**	**0**
5.	**1**	**0**	**1**	**1**
6.	**1**	**1**	**0**	**1**
7.	**1**	**1**	**1**	**1**

Pošto je ispunjena tablica istine, treba sad dobiti na osnovu te tablice funkciju u obliku $Y = f(A,B,C)$, a kad već postoji izraz crta se šema uređaja za glasanje.

Svaka logička funkcija sa n nezavisno promenljivih može napisati ili kao zbir minterma ili kao proizvod maksterma. U tabeli 6.2 su prikazani mintermi i makstermi za dati zadatak.

Tabela 6.2: mintermi i makstermi za primer 6.1						
r.b.	A	B	C	F	mintermi	makstermi
0.	**0**	**0**	**0**	**0**	$m_0 = \overline{A} \cdot \overline{B} \cdot \overline{C}$	$M_7 = A + B + C$
1.	**0**	**0**	**1**	**0**	$m_1 = \overline{A} \cdot \overline{B} \cdot C$	$M_6 = A + B + \overline{C}$
2.	**0**	**1**	**0**	**0**	$m_2 = \overline{A} \cdot B \cdot \overline{C}$	$M_5 = A + \overline{B} + C$
3.	**0**	**1**	**1**	**1**	$m_3 = \overline{A} \cdot B \cdot C$	$M_4 = A + \overline{B} + \overline{C}$
4.	**1**	**0**	**0**	**0**	$m_4 = A \cdot \overline{B} \cdot \overline{C}$	$M_3 = \overline{A} + B + C$
5.	**1**	**0**	**1**	**1**	$m_5 = A \cdot \overline{B} \cdot C$	$M_2 = \overline{A} + B + \overline{C}$
6.	**1**	**1**	**0**	**1**	$m_6 = A \cdot B \cdot \overline{C}$	$M_1 = \overline{A} + \overline{B} + C$
7.	**1**	**1**	**1**	**1**	$m_7 = A \cdot B \cdot C$	$M_0 = \overline{A} + \overline{B} + \overline{C}$

Iz tablice treba ispisati minterme, gde je funkcija 1, logička funkcija se dobija kao zbir (disjunkcija – logičko sabiranje) svih ovih minterma:

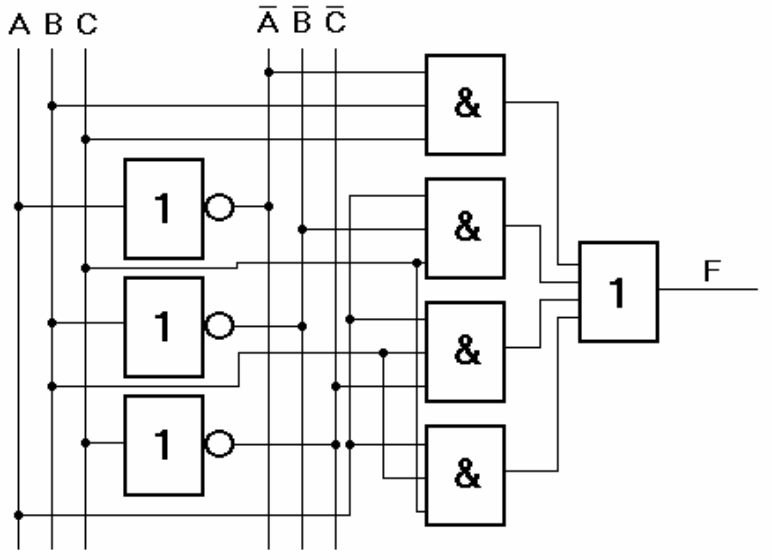
$$F = m_3 + m_5 + m_6 + m_7 = \sum_{i=0}^{2^n-1} m_i ; F = 1 \qquad (6.1)$$

Uzimajući ove minterme funkcija je:

$$F = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} + A \cdot B \cdot C \qquad (6.2)$$

Ovaj oblik funkcije, koji se dobija putem logičkog zbira svih minterma zove se **potpuna** (perfektna, kanonična odnosno savršena) **disjunktivna normalna forma**, i koristi se skraćenica **PDNF**.

Na slici 6.1 je logička šema rešenja zadatka.



Slika 6.1: logička šema primera 6.1

Funkciju možemo dobiti i pomoću maksterma, ako se uzimaju oni, za koje važi da je $F = 0$. U ovom slučaju funkcija je proizvod ovih maxterma:

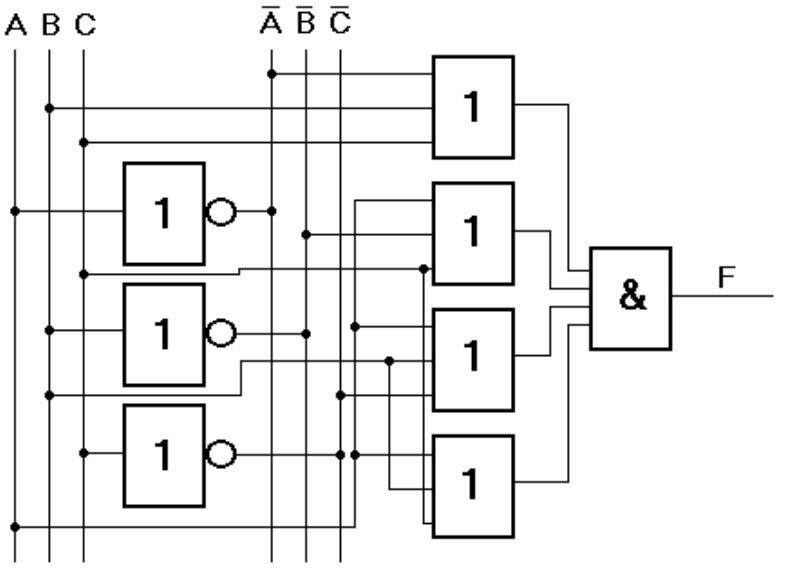
$$F = M_7 \cdot M_6 \cdot M_5 \cdot M_3 = \prod_{i=0}^{2^n-1} M_i ; F = 0 \qquad (6.3)$$

Uvrštavajući odgovarajuće maksterme dobićemo funkciju:

$$F = (A + B + C) \cdot (A + B + \overline{C}) \cdot (A + \overline{B} + C) \cdot (\overline{A} + B + C) \qquad (6.4)$$

Ovaj oblik funkcije, koji je dobijen logičkim množenjem maksterma, zove se **potpuna** (perfektna) **konjunktivna normalna forma**, za koju koristimo skraćenicu **PKNF**.

Na slici 6.2 je logička šema rešenja zadatka.



Slika 6.2: logička šema primera 6.1

Izrazi za funkciju **F** po postupku **PDNF** i **PKNF** formalno nisu isti, ali realizuju isti zadatak. Ako se primene poznati zakoni algebre logike, tj. ako se primeni minimizacija dobija se isti izraz za jedan (6.2), odnosno drugi (6.4) oblik funkcije.

☺ ☹ ☺

6.2.1 POSTUPAK ZA GENEZU FUNKCIJE POMOĆU PDNF

Ako postoji ispunjena tablica istine, postupak iznalaženja funkcije je sledeći:

- u tablici se uoče one redovi u kojima je $F = 1$,
- u tim redovima formiramo konjunkcije (logičke proizvode) svih binarnih promenljivih i
- formira se disjunkcija (logičko sabiranje) zbir tih proizvoda.

One promenljive, čija je vrednost u tablici istine jedinica zapisuje se u afirmaciji, a promenljive, koje u datoj vrsti imaju vrednost nula zapisujemo u negaciji. Na ovaj način se dobija zbir minterma, pri čemu svaki minterm sadrži sve ulazne promenljive. Svaka se ulazna promenljiva javlja u svakom mintermu samo jednom, ili u afirmaciji, ili u negaciji. Broj minterma je jednak broju onih vrsta iz tablice istine, gde je $F = 1$.

☺ ☹ ☺

6.2.2 POSTUPAK ZA GENEZU FUNKCIJE POMOĆU PKNF

Ovaj postupak je na neki način «slika u ogledalu» prethodnog posupka (**PDNF**). Ovde se posmatraju redovi, gde su $F = 0$.

U tim redovima se obrazuje disjunkcija (logički zbir) svih binarnih promenljivih a zatim se obrazuje konjunkcija (logički proizvod) tih maksterma. One promenljive, koje imaju u tablici istine u posmatranom redu vrednost nulu zapisuju se u afirmaciji, a one sa vrednošću jedan se zapisuju u negaciji. Na ovkav način dobijamo proizvod maksterma, gde svaki maksterm će sadržavat sve nezavisno-promenljive (ulazne promenljive) samo jednom ili u afirmaciji ili u negaciji.

Kod datog zadatka sve jedno dali je za iznalaženje funkcije korišćen postupak PDNF ili PKNF, posle minimizacije funkcija konačan rezultat će biti isti.

7. ANALIZA I SINTEZA LOGIČKIH MREŽA

7.1 UVOD

Kod analize logičkih mreža data je logička (ili prekidačka) mreža i traži se fukcija koja opisuje ovu mrežu.

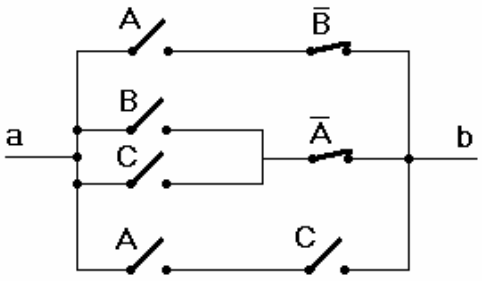
Kod sinteze logičkih mreža je situacija suprotna, treba projektovati logičku (ili prekidačku) mrežu na osnovu postavljenih uslova upravljanja, tehničke realizacije elemenata i ostalih uslova potrebnih za pouzdan rad mreža.

7.2 ANALIZA LOGIČKIH MREŽA

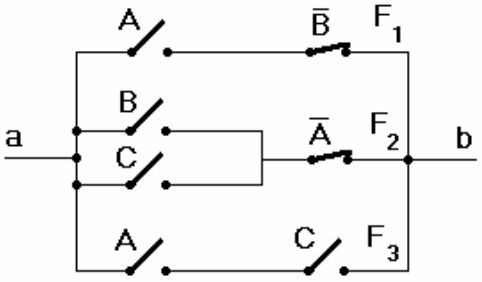
Kod analize logičkih (prekidačkih) mreža znači iz postojće mreže (šeme) treba odrediti funkciju, koja opisuje mrežu. Za rešenje problema pre svega moraju biti poznate logičke (prekidačke) funkcije pojedinih logičkih (prekidačkih) elemenata u mreži. Zbog toga treba izvršiti označavanje raznih priključaka mreže nekim simbolima. Ako je mreža jednostavna, onda se direktno mogu napisati izrazi za definisanje izlaznih funkcija.

PRIMER 7.1:

Data je prekidačka mreža upravljačkog kola (slika 7.1), naći logičku funkciju mreže. Možemo grane mreže označiti sa simbolima F_1, F_2 i F_3 (slika 7.2).



Slika 7.1: prekidačka mreža



Slika 7.2: prekidačka mreža sa oznakama

Pošto su grane paralelne tražena funkcija je:

$$F_{ab} = F_1 + F_2 + F_3 \qquad (7.1)$$

Oznake F_1, F_2 i F_3 su logičke funkcije pojedinih grana kola:

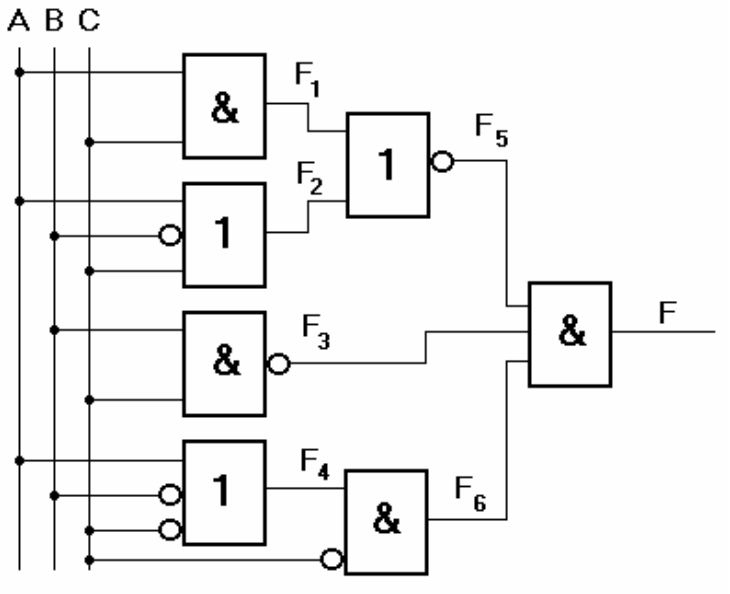
$$\begin{aligned} F_1 &= A \cdot \overline{B} \\ F_2 &= (B + C) \cdot \overline{A} \\ F_3 &= A \cdot C \end{aligned} \qquad (7.2)$$

Zamenom F_1, F_2 i F_3 (7.2) u jednačini (7.1) dobijamo funkciju, koja predstavlja logičku funkciju date prekidačke mreže:

$$F_{ab} = A \cdot \overline{B} + (B + C) \cdot \overline{A} + A \cdot C \qquad (7.3)$$

PRIMER 7.2:

Data je simbolička šema logičke mreže (slika 7.3), treba napisati logičku funkciju mreže.



Slika 7.3: simbolička šema mreže

Izlazi pojedinih logičkih kola sa šeme su:

$$\begin{aligned} F_1 &= A \cdot C \\ F_2 &= A + \overline{B} + C \\ F_3 &= \overline{B \cdot C} \\ F_4 &= A + \overline{B} + \overline{C} \\ F_5 &= \overline{F_1 + F_2} \\ F_6 &= F_4 \cdot \overline{C} \end{aligned} \qquad (7.4)$$

Izlazna funkcija je:

$$F = F_3 \cdot F_5 \cdot F_6 \qquad (7.5)$$

Zamenom funkcija F_1, F_2, F_3 i F_4 u (7.5):

$$\begin{aligned} F &= \overline{B \cdot C} \cdot \overline{F_1 + F_2} \cdot (A + \overline{B} + \overline{C}) \cdot \overline{C} = \\ &= \overline{B \cdot C} \cdot \overline{A \cdot C + A + \overline{B} + C} \cdot (A + \overline{B} + \overline{C}) \cdot \overline{C} \end{aligned} \qquad (7.6)$$

7.3 SINTEZA LOGIČKIH MREŽA

7.3.1 UVOD

Sinteza logičko-prekidačkih mreža je projektovanje mreža, i sastoji se u sagledavanju mogućnosti realizacije prekidačkih mreža na osnovu postavljenih uslova upravljanja, kao i tehničke realizacije elemenata. Za razmatrenje su i kod projektovanja uslovi, koji su potrebni za nesmetani, pouzdan rad upravljačkih sistema.

Sledeći koraci su obavezni kod sinteze mreže:

- prvo se zadaje projektni zadatak, gde su definisani zahtevi prema upravljačkom uređaju,
- na osnovu projektnog zadatka se određuje broj ulaznih i izlaznih promenljivih,
- određuje se stanja izlaznih promenljivih u tablici,
- iznalaženje funkcije se izvodi na osnovu tablice istine za pojedine izlaze,
- na osnovu tako dobijenih funkcija pristupa se crtanju mreže i
- na kraju realizaciji prekidačke mreže.

U daljem ćemo se kod sinteze mreže ograničiti samo na iznalaženje logičko-prekidačke mreže koja ostvaruje datu funkciju, a ne analiziramo vrstu elemenata (tehničku realizaciju) od kojih se mreža može sastaviti.

Bez obzira na to, dali je funkcija data u disjunktivnoj ili konjunktivnoj normalnoj formi, ona se uvek može ostvariti pomoću paralelno i/ili redno vezanih prekidačkih elemenata.

Kod logičkih sklopova, mreža koja ostvaruje normalnu formu jedne funkcije je takozvana dvostepena mreža. Prvi stepen se sastoji od više logičkih **I** ili **ILI** elemenata čiji si izlazi vezani na ulaze drugog stepena koji se sastoji od jednog **I** ili **ILI** kola.

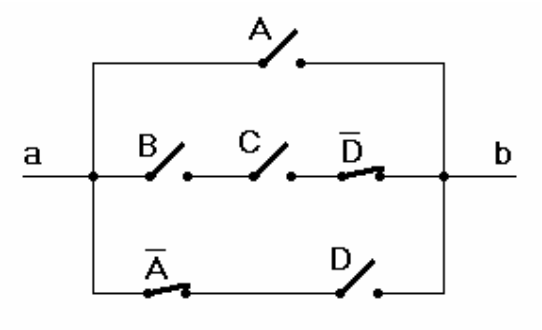
7.3.2 SINTEZA LOGIČKIH MREŽA POMOĆU SLOBODNO-IZABRANIH LOGIČKIH ELEMENATA

PRIMER 7.3:

Treba projektovati prekidačku mrežu, ako je data funkcija:

$$F = A + B \cdot C \cdot \overline{D} + \overline{A} \cdot D \qquad (7.7)$$

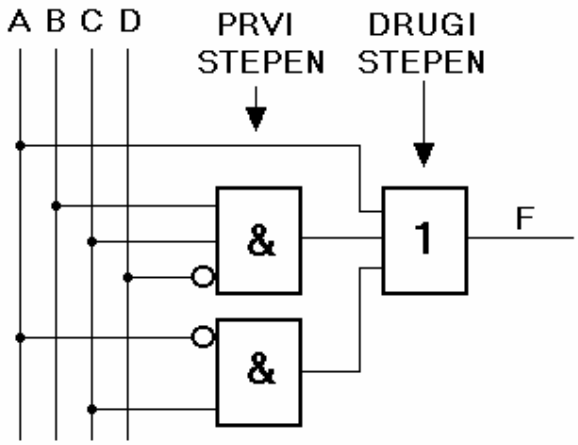
(7.7) je disjunktivna normalna forma, ovu funkciju realizuje paralelna veza redno vezanih prekidačkih elemenata (slika 7.4).



Slika 7.4: rešenje zadatka iz primer 7.3

PRIMER 7.4:

Realizovati funkciju (7.7) (PRIMER 7.3) pomoću logičkih sklopova. Simbolička šema rešenja je na slici 7.5.



Slika 7.5: simbolička šema iz primera 7.4 sa logičkim sklopovima

7.3.3 SINTEZA LOGIČKIH MREŽA POMOĆU *NI* ILI *NILI* ELEMENATA

U primerima 7.2 i 7.4 korišćena su različita logička kola. Neki ulazi su bili direktni, neki invertovani, negde je kolo imalo 2, negde tri ili više ulaza. Ponekad je teško naći odgovarajuće kolo za realizaciju date funkcije, pa zbog toga trebalo bi razmišljati o tome, da se primenjuje samo jedna vrsta kola. Lakše je projektovanje upravljačkog sistema sa istim elementima, a kod održavanja i servisiranja je potrebno na lageru imati samo jednu vrstu kola.

Često se koriste **NI**, ili **NILI** kola u takvim rešenjima. **NI** kolo realizuje **I** funkciju i negaciju, odnosno **NILI** kolo **ILI** funkciju i negaciju, a osim toga već smo videli, da pomoću De Morgan-ovog zakona bilo koju funkciju možemo pretvoriti u **NI** ili **NILI** oblik (poglavlje 5.2.1).

Prethodno videli smo i to, da smo sa dvostrukom negacijom dobili originalnu funkciju (poglavlje 5.2.1), odnosno, da se vrednost funkcije neće promeneti ako se izvrši dvostruka negacija. Posle dvostruke negacije dobićemo novi oblik funkcije.

Interesantno je, da De Morgan-ov zakon praktično od **I** funkcije napravi **ILI** (u stvari **NILI**, ali je to **ILI** i negacija), a od **ILI** funkcije **I** (**NI**, **I** i negaciju).

PRIMER 7.5:

Realizovati sledeću funkciju sa **NILI** elementima:

$$F = A \cdot B \cdot C \cdot D \tag{7.8}$$

Prema De Morgan-ovom zakonu, primenom dvostruke negacije dobijamo:

$$F = \overline{\overline{A \cdot B \cdot C \cdot D}} = \overline{\overline{A} + \overline{B} + \overline{C} + \overline{D}} \tag{7.9}$$

(7.9) predstavlja **NILI** funkciju.

PRIMER 7.6:

Na isti način možemo funkciju disjunktivnog oblika pretvoriti u NILI oblik. U zadatku treba naći NILI oblik za funkciju iz primera 7.3. (7.7):

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

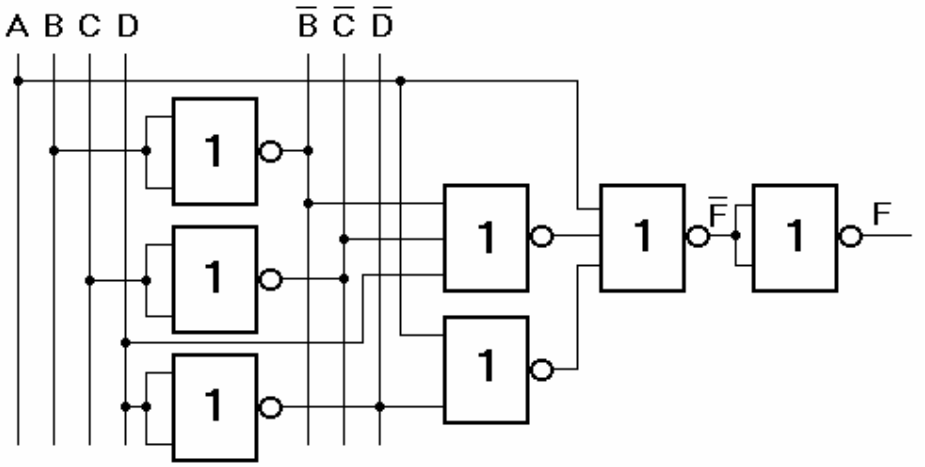
Primenom dvostruke negacije na pojedine minterme možemo dobiti NILI oblik sabiraka:

$$F = \overline{\overline{A}} + \overline{\overline{B \cdot C \cdot \overline{D}}} + \overline{\overline{\overline{A} \cdot D}} = A + \overline{\overline{B} + \overline{C} + D} + \overline{A + \overline{D}} \qquad (7.10)$$

Očigledno je, da oblik funkcije (7.10) još nije **NILI**, jer sadrži **ILI** funkciju, pa zbog toga primenom dvostruke negacije na celu jednačinu dobijamo **NILI** oblik cele funkcije:

$$F = \overline{\overline{A + \overline{\overline{B} + \overline{C} + D}} + \overline{A + \overline{D}}} \qquad (7.11)$$

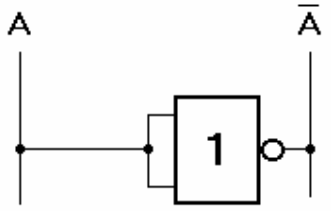
(7.11) ima još jednu dodatnu negaciju. Realizacija dobijene funkcije je prikazana na slici 7.6.



Slika 7.6: realizacija funkcije iz primera 7.3 sa **NILI** elementima

Na slici 7.6 su isključivo korišćena **NILI** kola, tako i za negaciju ulaznih promenljivih i za izlaz **F**. Pošto je NILI kolo minimalno sa dva ulaza, ulazi su spojeni u slučaju negacije (negacija je unarna funkcija). Na ovakav način od **NILI** kola može se napraviti kolo za negaciju, a preko tablice istine (slika 7.7) možemo to i dokazati.

r.b.	A	B	NILI	NE
0.	0	0	1	1
1.	0	1	0	-
2.	1	0	0	-
3.	1	1	0	0

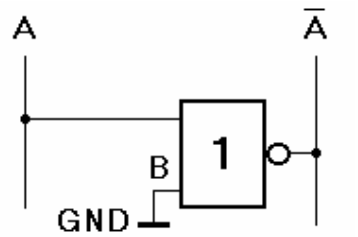


Slika 7.7: korišćenje **NILI** kola za negaciju

Negacija je unarna funkcija, pa zbog toga redovi 1. i 2. u tablici istine sa slike 7.7 ne postoje.

Drugo rešenje za pretvaranje **NILI** kola u kolo za negaciju je, ako od dva ulaza na jedan fiksno spajamo logičku nulu (**0**). Na slici 7.8 je data tablica istine **NILI** kola sa dva ulaza, kao i šema, vidi se, da ako je na ulazu **B** konstantna nula (masa-**GND**) samo 0. i 2. redovi su aktuelni.

r.b.	A	B	NILI	NE
0.	0	0	1	1
1.	0	1	0	-
2.	1	0	0	0
3.	1	1	0	-



Slika 7.8: korišćenje **NILI** kola za negaciju, ulaz **B** uvek **0** (masa)

U integrisanim kolima, u čipovima često imamo više istih elemenata. Na primer u **TTL** (Transistor Transistor Logic) seriji kolo sa oznakom SN 7400 ima 4 **NI** kola sa dva ulaza. Čip ima 14 nožica (pinova), dve za napajanje, a pošto svako kolo pored dva ulaza ima jedan izlaz, ukupno tri izvoda, četiri kola puta 3 izvoda su 12, i ovako ima ukupno 14 nožica čip.

PRIMER 7.7:

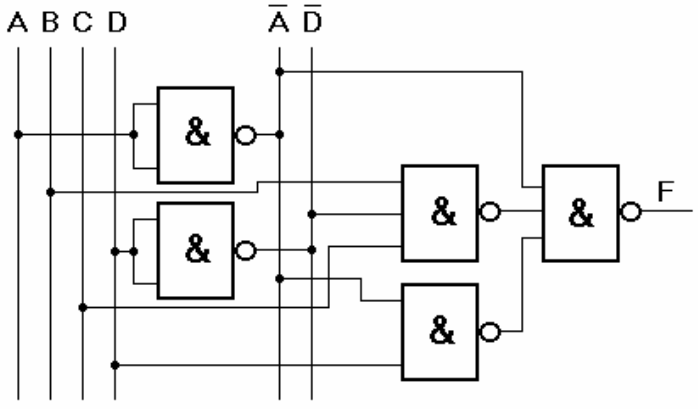
U ovom primeru istu funkciju (7.7) iz primera 7.3 realizujemo sa **NI** elementima:

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

Primenom dvostruke negacije ne celu jednačinu dobijamo:

$$F = \overline{\overline{A + B \cdot C \cdot \overline{D} + \overline{A} \cdot D}} = \overline{\overline{A} \cdot \overline{B \cdot C \cdot \overline{D}} \cdot \overline{AD}} \qquad (7.12)$$

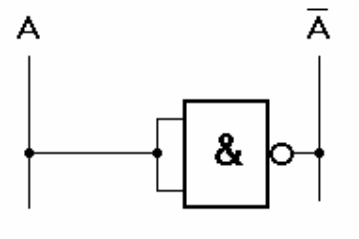
(7.12) je očigledno funkcija sa **NI** elementima (i negacijama), šema te funkcije je prikazana na slici 7.9.



Slika 7.9: realizacija funkcije iz primera 7.3 sa **NI** elementima

Na slici 7.9 su isključivo korišćena **NI** kola, tako i za negaciju ulaznih promenljivih i za izlaz **F**. Pošto je **NI** kolo minimalno sa dva ulaza, ulazi su spojeni u slučaju negacije (negacija je unarna funkcija). Na ovakav način od **NI** kola može se napraviti kolo za negaciju, a preko tablice istine (slika 7.10) možemo to i dokazati.

r.b.	A	B	NI	NE
0.	0	0	1	1
1.	0	1	1	-
2.	1	0	1	-
3.	1	1	0	0

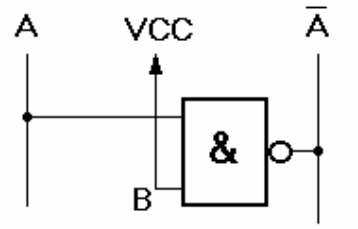


Slika 7.10: korišćenje **NI** kola za negaciju

Negacija je unarna funkcija, pa zbog toga redovi 1 i 2 u tablici istine sa slike 7.9 ne postoje.

Drugo rešenje za pretvaranje **NI** kola u kolo za negaciju je, ako od dva ulaza na jedan fiksno spajamo logičku jedinicu (**1**). Na slici 7.11 je data tablica istine **NI** kola sa dva ulaza, kao i šema, vidi se, da ako je na ulazu **B** konstantna jedinica (napajanje-**VCC**) samo 1. i 3. redovi su aktuelni.

r.b.	A	B	NI	NE
0.	0	0	1	-
1.	0	1	1	1
2.	1	0	1	-
3.	1	1	0	0



Slika 7.11: korišćenje **NI** kola za negaciju, ulaz **B** uvek 1 (V_{cc} , napajanje)

8. MINIMIZACIJA LOGIČKO PREKIDAČKIH KOLA

8.1 UVOD

Projektovanje digitalnih upravljačkih sistema možemo realizovati na dva načina:

- intuitivno i
- na osnovu tablice istine.

Ako se koristi tablica istine, onda su dobijene logičke funkcije:

- disjunktivnog normalnog ili
- konjunktivnog normalnog

oblika.

U oba slučaja (intuitivno ili na osnovu tablice istine realizovano projektovanje) dobijene logičke funkcije obično nisu minimalnog oblika. Ovo znači, da uprvljački sistemi nisu ekonomični (veći broj elemenata, veća cena, veća potrošnja, povećani troškovi održavanja i servisiranja itd.).

Zbog gore navedenih problema treba težiti ka tome, da se koristi što manji broj elemenata (kola). U literaturi ima više metoda minimizacije, koje ne daju uvek isti rezultat, ponekad su u suštini iste, ponekad ne daju apsolutni minimum. U ovom pogavlju anliziramo najpoznatije metode minimizacije.

8.2 MINIMIZACIJA METODOM ALGEBARSKIH TRANSFORMACIJA

Do minimalnog oblika logičke funkcije može se doći najednostavnije sa algebarskim transformacijama. Ova metoda je efikasna u slučaju, ako je broj nezavisnih promenljivih mali, i ako je originalna funkcija jednostavna. Treba reći, da je metoda intuitivna, prema tome nije sistematska.

Kod algebarske minimizacije treba pronaći najjednostavniji oblik ekvivalentnih algebarskih jedačina, koje omogućavaju realizaciju date funkcije sa najmanjim brojem elemenata.

Funkcija minimalnog oblika ima sledeće karakteristike:

- ne može se iz logičke funkcije izostaviti ni jedan minterm (ili maksterm) i
- ne može se iz logičke funkcije izostaviti bilo koja promenljiva.

Odnosno logička funkcija ne sadrži ni jedan redundantni term (minterm ili maksterm) ni redundantnu promenljivu, a da se pri tom vrednost funkcije ne promeni. Nakon algebarske minimizacije broj oznaka koje reprezentuju nezavisne promenljive kao i broj disjunktivnih i konjunktivnih operacija je minimalan.

☹ ☺ ☹

PRIMER 8.1:

Minimizirati sledeću funkciju algebarskom metodom:

$$F = A \cdot \overline{B} + C + \overline{A} \cdot \overline{C} \cdot D + B \cdot \overline{C} \cdot D \qquad (8.1)$$

Iz trećeg i četvrtog člana funkcije možemo izvući $\overline{C}$:

$$F = A \cdot \overline{B} + C + \overline{A} \cdot \overline{C} \cdot D + B \cdot \overline{C} \cdot D = A \cdot \overline{B} + C + \overline{C} \cdot (\overline{A} \cdot D + B \cdot D) \quad (8.2)$$

U tabeli 5.2 (poglavlje 5.) je zakon 11b, koji glasi:

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

a koji možemo napisati sa promenljivama x i y:

$$x + \cdot\overline{x} \cdot y = x + z\,,$$

uzimajući za:

- $x = C$ i
- $y = \overline{A} \cdot D + B \cdot D$.

Zadnja dva člana funkcije (8.2) se mogu pojednostaviti:

$$F = A \cdot \overline{B} + C + \overline{A} \cdot D + B \cdot D \qquad (8.3)$$

U jednačini (8.3) iz trećeg i četvrtog člana izvucimo promenljivu D:

$$F = A \cdot \overline{B} + C + D \cdot (\overline{A} + B) \qquad (8.4)$$

Na zadnji član funkcije (8.4) prmenimo De Morgan-ov zakon (tabela 5.2):

$$\overline{A} + B = \overline{A \cdot \overline{B}} \ ,$$

pa je funkcija:

$$F = A \cdot \overline{B} + C + D \cdot (\overline{A \cdot \overline{B}}) \qquad (8.5)$$

U tabeli 5.2 (poglavlje 5.) je zakon 11b, koji glasi:

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

a koji možemo napisati sa promenljivama x i y:

$$x + \cdot \overline{x} \cdot y = x + z \ ,$$

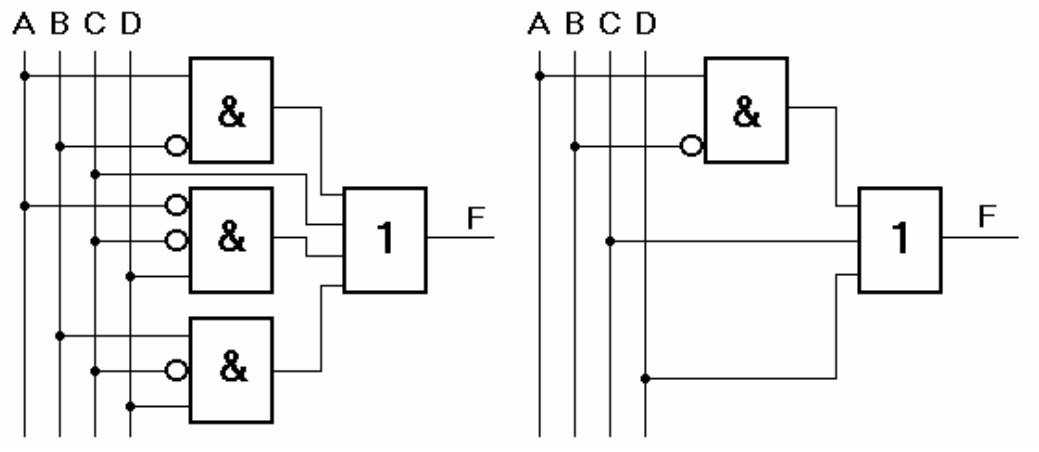
uzimajući za:

- $x = A \cdot \overline{B}$ i
- $y = D$.

Prvi i zadnji članovi funkcije (8.5) se mogu pojednostaviti:

$$F = A \cdot \overline{B} + C + D \qquad (8.6)$$

Dobili smo minimalni oblik (8.6) originlne funkcije (8.1). Na slici 8.1 je data realizacija originalne i minimizirane funkcije.



Slika 8.1: logička šema originalne (8.1) iminimizirane (8.6) funkcije

Ispravnost minimizacije možemo dokazati pomoću tablice istine (tabela 8.1).

Tabela 8.1: tablica istine funkcija (8.1) i (8.6)

r.b.	A	B	C	D	$F = A \cdot \overline{B} + C + \overline{A} \cdot \overline{C} \cdot D + B \cdot \overline{C} \cdot D$	$F = A \cdot \overline{B} + C + D$
0.	0	0	0	0	0	0
1.	0	0	0	1	1	1
2.	0	0	1	0	1	1
3.	0	0	1	1	1	1
4.	0	1	0	0	0	0
5.	0	1	0	1	1	1
6.	0	1	1	0	1	1
7.	0	1	1	1	1	1
8.	1	0	0	0	1	1
9.	1	0	0	1	1	1
10.	1	0	1	0	1	1
11.	1	0	1	1	1	1
12.	1	1	0	0	0	0
13.	1	1	0	1	1	1
14.	1	1	1	0	1	1
15.	1	1	1	1	1	1

☹ ☺ ☹

8.3 KARNAUGH-OVA METODA MINIMIZACIJE

Ova metoda spada u grupu grafičke minimizacije što znači da logičku funkciju moramo prikazati u grafičkom obliku. Mada teoretski može se koristiti za minimizaciju logičkih funkcija sa bilo koliko ulaznih promenljivih, u praksi najpogodnija je za minimizaciju funkcije sa tri ili četiri promenljive, retko i sa pet.

Na osnovu disjunktivne normalne forme logičke funkcije formiramo takazvanu Karnaugh-ovu kartu. Praktično tablicu istine prikazujemo jedan prema jedan u ovoj formi. Ako smatramo da u tablici istine radimo sa «vektorom», onda kod Karnough-ove tablice isto to prikazujemu u obliku «matrice».

Ako logička funkcija ima **n** promenljivih, onda Karnaugh-ova karta sadrži 2^n polja (kao tablica istine 2^n redova). Svako polje u tablici predstavlja minterm, a u tablici između svaka susednih polja razlikuje se uvek samo jedan znak, tj, za jednu logičku vrednost (u tablici istine nije tako!).

Polja mogu biti označena na različite načine od kojih su neki:

- minterm ➔ m_5,
- promenljive ➔ $\overline{A} \cdot B \cdot \overline{C} \cdot D$,
- binarni brojevi ➔ 0101 i
- decimalni broj ➔ 5.

☹ ☺ ☹

PRIMER 8.2:

Prikazati Karnaugh-ovu tablicu za slučaj da se imaju četiri ulazne promenljive, a da pri tom funkcija ima jedinice za svaki mogući minterm.

a) algebarski oblik funkcije je:

$$F = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot \overline{B} \cdot C \cdot D + \overline{A} \cdot B \cdot \overline{C} \cdot \overline{D} + \\ + \overline{A} \cdot B \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot C \cdot D + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} \cdot D + \\ + A \cdot \overline{B} \cdot C \cdot \overline{D} + A \cdot \overline{B} \cdot C \cdot D + A \cdot B \cdot \overline{C} \cdot \overline{D} + A \cdot B \cdot \overline{C} \cdot D + A \cdot B \cdot C \cdot \overline{D} + \\ + A \cdot B \cdot C \cdot D \qquad (8.7)$$

b) prikaz funkcije preko minterma sa binarnim brojevima je:

$$F = 0000 + 0001 + 0010 + 0011 + 0100 + 0101 + 0110 + 0111 + \\ + 1000 + 1001 + 1010 + 1011 + 1100 + 1101 + 1110 + 1111 \qquad (8.8)$$

c) prikaz funkcije pomoću skupa indeksa:

$$F = \overset{4}{\Sigma}(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15) \qquad (8.9)$$

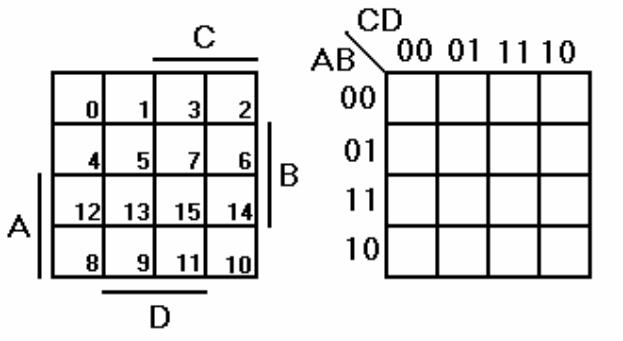
d) tablica istine funkcije je:

Tabela 8.2: tablica istine funkcije iz primera 8.2

r.b.	A	B	C	D	F
0.	0	0	0	0	1
1.	0	0	0	1	1
2.	0	0	1	0	1
3.	0	0	1	1	1
4.	0	1	0	0	1
5.	0	1	0	1	1
6.	0	1	1	0	1
7.	0	1	1	1	1
8.	1	0	0	0	1
9.	1	0	0	1	1
10.	1	0	1	0	1
11.	1	0	1	1	1
12.	1	1	0	0	1
13.	1	1	0	1	1
14.	1	1	1	0	1
15.	1	1	1	1	1

e) Karnaugh-ova karta:

Karta za datu logičku funkciju ima ukupno $p = 2^n = 2^4 = 16$ polja, to znači, da se karta sastoji od mreže pravougaonika sa 16 polja (slika 8.2).



Slika 8.2: Krnaugh-ova tabla funkcije sa 4 ulazne promenljive

Veza između kanoničnih termova (minterma i maksterma) i pojedinih pravougaonika izvodi se na sledeći način:

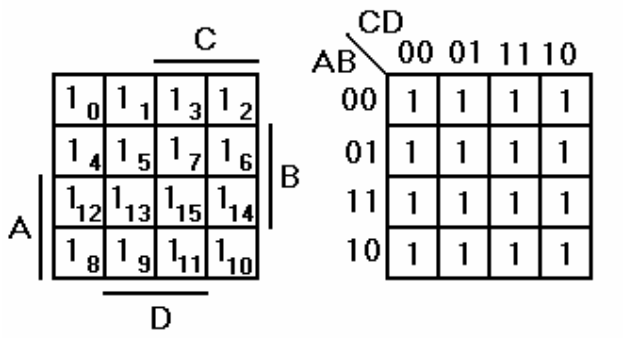
- vrste i kolone Karnaugh-ove karte binarno se kodiraju.

Odabiranje koda ide tako, da se binarne kombinacije susednih pravougaonika razlikuju samo za jedan znak. Tako se vertikalno, gde su promenljive **A** i **B** nanose redom binarne oznake: **00**, **01**, **11** i **10** odozgo naniže. Horizontalno, gde su promenljive **C** i **D** nanose se oznake: **00**, **01**, **11** i **10** s leva na desno.

Ovakvim rasporedom binarnih oznaka postignuto je to da se susedna polja razlikuju samo za jedan znak. Na primer, polja **7** i **15** predstavljaju minterme $m_7 = \overline{A} \cdot B \cdot C \cdot D$ i $m_{15} = A \cdot B \cdot C \cdot D$, koji se razlikuju u logičkoj vrednosti označene sa **A**.

Zadata logička funkcija se u Karnaugh-ovoj tablici predstavlja tako, da u odgovarajuća polja zadatih minterma upisujemo 1 (jedinicu) a u ostala polja 0 (nulu), ili ih uopšte ne obeležavamo.

Rešenje zadatka je na slici 8.3.



Slika 8.3: Karnaugh-ova tabela primera 8.2

☹ ☺ ☹

PRIMER 8.3:

Funkciju $F(A,B,C,D) = \sum^{4}(1,3,4,5,8,14,15)$ predstaviti Karnaugh-ovom karticom.

a) algebarski oblik je:

$$F = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot D + \overline{A} \cdot B \cdot \overline{C} \cdot \overline{D} + \\ + \overline{A} \cdot B \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + A \cdot B \cdot C \cdot \overline{D} + A \cdot B \cdot C \cdot D \qquad (8.10)$$

b) mintermi sa binarnim brojevima su:

$$F = 0001 + 0011 + 0100 + 0101 + \\ + 1000 + 1110 + 1111 \qquad (8.11)$$

c) prikaz pomoću skupa indeksa:

$$F(A,B,C,D) = \sum^{4}(1,3,4,5,8,14,15) \qquad (8.12)$$

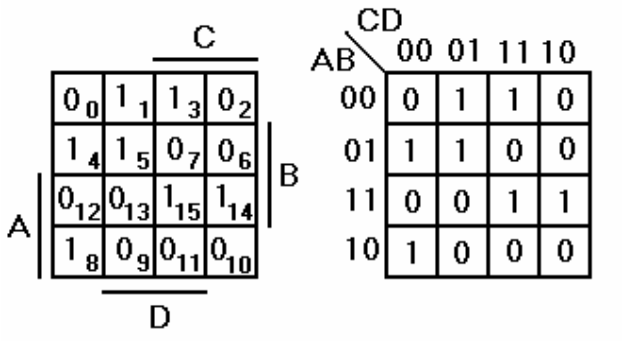
d) tablica istine je:

Tabela 8.3: tablica istine funkcije iz primera 8.3

r.b.	A	B	C	D	F
0.	0	0	0	0	0
1.	0	0	0	1	1
2.	0	0	1	0	0
3.	0	0	1	1	1
4.	0	1	0	0	1
5.	0	1	0	1	1
6.	0	1	1	0	0
7.	0	1	1	1	0
8.	1	0	0	0	1
9.	1	0	0	1	0
10.	1	0	1	0	0
11.	1	0	1	1	0
12.	1	1	0	0	0
13.	1	1	0	1	0
14.	1	1	1	0	1
15.	1	1	1	1	1

e) Karnaugh-ova karta:

Karta za datu logičku funkcuju ima ukupno $p = 2^n = 2^4 = 16$ polja, to znači, da se karta sastoji od mreže pravougaonika sa 16 polja (slika 8.3).



Slika 8.4: Krnaugh-ova tabla za primer 8.3

☹ ☺ ☹

PRIMER 8.4:

Funkciju $F(A,B,C) = \overset{3}{\Sigma}(3,4,6,7)$ predstaviti Karnaugh-ovom karticom.

f) algebarski oblik je:

$$F = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot B \cdot C \qquad (8.13)$$

g) mintermi sa binarnim brojevima:

$$F = 011 + 100 + 011 + 111 \qquad (8.14)$$

h) prikaz pomoću skupa indeksa:

$$F(A,B,C) = \overset{3}{\Sigma}(3,4,6,7) \qquad (8.15)$$

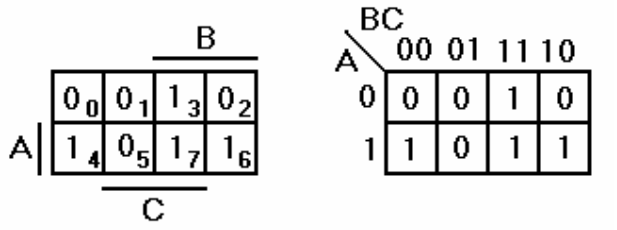
i) tablica istine je:

Tabela 8.4: tablica istine funkcije iz primera 8.4

r.b.	A	B	C	F
0.	0	0	0	0
1.	0	0	1	0
2.	0	1	0	0
3.	0	1	1	1
4.	1	0	0	1
5.	1	0	1	0
6.	1	1	0	1
7.	1	1	1	1

j) Karnaugh-ova karta:

Karta za datu logičku funkcuju ima ukupno $p = 2^n = 2^3 = 8$ polja, to znači, da se karta sastoji od mreže pravougaonika sa 8 polja (slika 8.5).



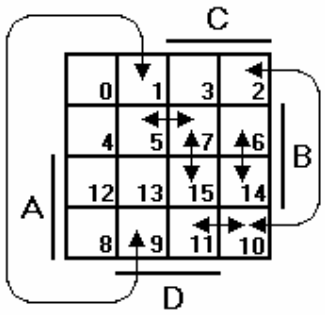
Slika 8.5: Krnaugh-ova tabla za primer 8.4

☹ ☺ ☹

8.3.1 POSTUPAK KARNAUGH-OVE MINIMIZACIJE

Postupak minimizacije pomoću ove metode je veoma brz, pregledan i efikasan. Na Karnaugh-ovoj karti susedni termovi uvek se međusobno razlikuju samo za vrednost jedne promenljive.

Na slici 8.6 je dat Karnaugh-ov dijagram sa četiri promenljive. Na karti su označene neki susedni mintermi (ne svi). Vidi se, da su mintermi na ivicama karte susedne sa mintermima, koji su na suprotnim ivicama.



Slika 8.6: neki susedni mintermi u Karnaugh-ovoj karti

Postupak minimizacije se svodi na traženje, zapažanje karakterističnih konfiguracija simbola 1 (jedinice) u karti. Jedinice treba prethodno upisati u kartu na osnovu zadatih podataka (na primer iz tablice istine). Grupisanje jedinica dovode do eliminacije jedne ili više promenljivih.

Karakteristične konfiguracije simbola 1 (jedinica) se nalaze tako da se uzimaju u obzir samo susedne 1 (jedinice), od njih se zaokružuje 1, 2, 4, 8, ... 2^k , gde je k ceo broj i manji od broja ulaznih prmenljivih. Pri stvaranju konfiguracija, zaokružene jedinice mogu se združiti sa susednim još nezaokruženim jedinicama. Ovo se može izvoditi sve dok postoji i jedna slobodna susedna jedinica.

☹ ☺ ☹

PRIMER 8.5:

Funkcije $F(A,B,C)=\overset{3}{\Sigma}(1,5)$, $G(A,B,C)=\overset{3}{\Sigma}(1,3)$ i $H(A,B,C)=\overset{3}{\Sigma}(0,2)$ minimizirati Karnaugh-ovim karticama.

a) algebarski oblici su:

$$F=\overline{A}\cdot\overline{B}\cdot C+A\cdot\overline{B}\cdot C,$$
$$G=\overline{A}\cdot\overline{B}\cdot C+\overline{A}\cdot B\cdot C \quad \text{i}$$
$$H=\overline{A}\cdot\overline{B}\cdot\overline{C}+\overline{A}\cdot B\cdot\overline{C} \qquad (8.16)$$

b) mintermi sa binarnim brojevima su:

$$F = 001 + 101$$
$$G = 001 + 011$$
$$H = 001 + 010 \qquad (8.17)$$

c) pomoću skupa indeksa:

$$F(A,B,C) = \overset{3}{\Sigma}(1,5)$$
$$G(A,B,C) = \overset{3}{\Sigma}(1,3)$$
$$H(A,B,C) = \overset{3}{\Sigma}(0,2) \qquad (8.18)$$

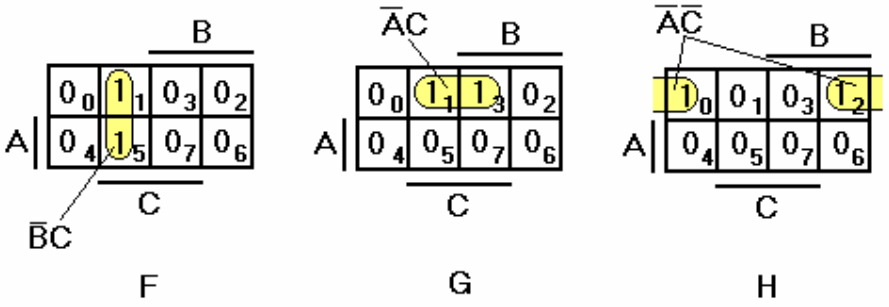
d) tablica istine:

Tabela 8.5: tablica istine funkcije iz primera 8.4

r.b.	A	B	C	F	G	H
0.	0	0	0	0	0	1
1.	0	0	1	1	1	0
2.	0	1	0	0	0	1
3.	0	1	1	0	1	0
4.	1	0	0	0	0	0
5.	1	0	1	1	0	0
6.	1	1	0	0	0	0
7.	1	1	1	0	0	0

e) Karnaugh-ove karte:

Karte za date logičke funkcuje imaju ukupno $p = 2^n = 2^3 = 8$ polja, to znači, da se karte sastoje od mreže pravougaonika sa 8 polja (slika 8.6).



Slika 8.6: Krnaugh-ove table sa minimizacijama za primer 8.5

f) minimizirane funkcije su:

$$F = \overline{B} \cdot C\,,$$
$$G = \overline{A} \cdot C \text{ i}$$
$$H = \overline{A} \cdot \overline{C} \qquad (8.19)$$

☹ ☺ ☹

PRIMER 8.6:

Funkcije $F(A,B,C) = \overset{3}{\Sigma}(1,3,5,7)$, $G(A,B,C) = \overset{3}{\Sigma}(0,1,2,3)$ i $H(A,B,C) = \overset{3}{\Sigma}(0,2,4,6)$ minimizirati Karnaugh-ovim karticama.

a) algebarski oblici su:

$$F = \overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot C\,,$$
$$G = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C \qquad \text{i}$$
$$H = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C} + A \cdot B \cdot \overline{C} \qquad (8.20)$$

b) mintermi sa binarnim brojevima su:

$$F = 001 + 011 + 101 + 111$$
$$G = 000 + 001 + 010 + 011$$
$$H = 000 + 010 + 100 + 110 \qquad (8.21)$$

c) prikaz pomoću skupa indeksa:

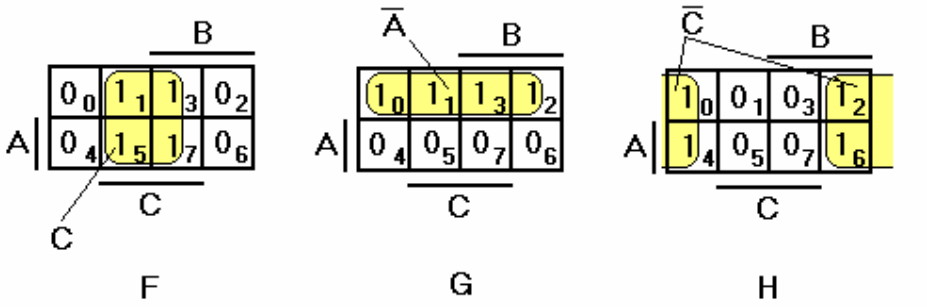
$$F(A,B,C) = \overset{3}{\Sigma}(1,3,5,7)$$
$$G(A,B,C) = \overset{3}{\Sigma}(0,1,2,3)$$
$$H(A,B,C) = \overset{3}{\Sigma}(0,2,4,6) \qquad (8.22)$$

d) tablica istine:

Tabela 8.6: tablica istine funkcije iz primera 8.4						
r.b.	A	B	C	F	G	H
0.	0	0	0	0	1	1
1.	0	0	1	1	1	0
2.	0	1	0	0	1	1
3.	0	1	1	1	1	0
4.	1	0	0	0	0	1
5.	1	0	1	1	0	0
6.	1	1	0	0	0	1
7.	1	1	1	1	0	0

e) Karnaugh-ove karte:

Karte za date logičke funkcije imaju ukupno $p = 2^n = 2^3 = 8$ polja, to znači, da karte se sastoje od mreže pravougaonika sa 8 polja (slika 8.8).



Slika 8.8: Krnaugh-ove table sa minimizacijama za primer 8.6

f) minimizirane funkcije su:

$$F = C,$$
$$G = \overline{A} \text{ i}$$
$$H = \overline{C} \qquad (8.23)$$

☹ ☺ ☹

PRIMER 8.7:

Funkcije $F(A,B,C,D) = \overset{4}{\Sigma}(5,13)$, $G(A,B,C,D) = \overset{4}{\Sigma}(4,5,6,7,12,13,14,15)$ i $H(A,B,C,D) = \overset{4}{\Sigma}(1,2,4,5,11,12,15)$ minimizirati Karnaugh-ovim karticama.

a) algebarski oblici su:

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

$$G = \overline{A} \cdot B \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot C \cdot D + \\ + A \cdot B \cdot \overline{C} \cdot \overline{D} + A \cdot B \cdot \overline{C} \cdot D + A \cdot B \cdot C \cdot \overline{D} + A \cdot B \cdot C \cdot D$$ i

$$H = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot D + \\ + A \cdot \overline{B} \cdot C \cdot D + A \cdot B \cdot \overline{C} \cdot D + A \cdot B \cdot C \cdot D \qquad (8.24)$$

b) mintermi sa binarnim brojevima su:

$$F = 0101 + 1101$$
$$G = 0100 + 0101 + 0110 + 0111 + 1100 + 1101 + 1110 + 1111$$
$$H = 0001 + 0100 + 0100 + 0101 + 1011 + 1100 + 1111 \qquad (8.25)$$

c) prikaz pomoću skupa indeksa:

$$F(A,B,C,D) = \overset{4}{\Sigma}(5,13)$$
$$G(A,B,C,D) = \overset{4}{\Sigma}(4,5,6,7,12,13,14,15)$$
$$H(A,B,C,D) = \overset{4}{\Sigma}(1,2,4,5,11,12,15) \qquad (8.26)$$

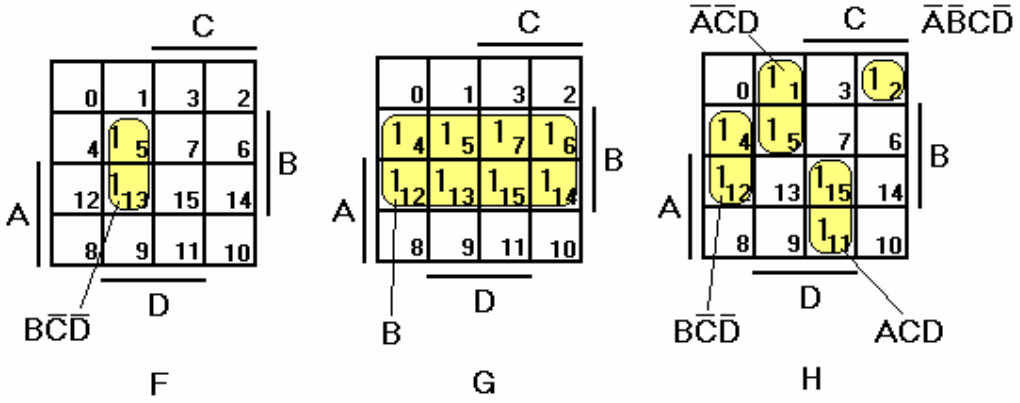
d) tablica istine:

Tabela 8.7: tablica istine funkcije iz primera 8.4

r.b.	A	B	C	D	F	G	H
0.	0	0	0	0	0	0	0
1.	0	0	0	1	0	0	1
2.	0	0	1	0	0	0	1
3.	0	0	1	1	0	0	0
4.	0	1	0	0	0	1	1
5.	0	1	0	1	1	1	1
6.	0	1	1	0	0	1	0
7.	0	1	1	1	0	1	0
8.	1	0	0	0	0	0	0
9.	1	0	0	1	0	0	0
10.	1	0	1	0	0	0	0
11.	1	0	1	1	0	0	1
12.	1	1	0	0	0	1	1
13.	1	1	0	1	1	1	0
14.	1	1	1	0	0	1	0
15.	1	1	1	1	0	1	1

e) Karnaugh-ove karte:

Karte za date logičke funkcije imaju ukupno $p = 2^n = 2^4 = 16$ polja, to znači, da se karte sastoje od mreže pravougaonika sa 16 polja (slika 8.9).



Slika 8.9: Karnaugh-ove table sa minimizacijama

f) minimizirane funkcije su:

$$F = B \cdot \overline{C} \cdot \overline{D},$$

$$G = B \text{ i}$$

$$H = B \cdot \overline{C} \cdot \overline{D} + A \cdot C \cdot D + \overline{A} \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} \qquad (8.27)$$

☹ ☺ ☹

PRIMER 8.8:

Funkcije: $F(A,B,C,D) = \sum^{4}(1,3,5,6,7,8,9,10,11,14,15)$,

$G(A,B,C,D) = \sum^{4}(1,3,6,9,11,12,13,14,15)$ i $H(A,B,C,D) = \sum^{4}(0,2,3,7,8,10,11,15)$

minimizirati Karnaugh-ovim karticama.

a) algebarski oblici su:

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

,

$$G = \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot D + \overline{A}\cdot\overline{B}\cdot C\cdot D + \overline{A}\cdot B\cdot C\cdot\overline{D} + A\cdot\overline{B}\cdot\overline{C}\cdot D +$$
$$+ A\cdot\overline{B}\cdot C\cdot D + A\cdot B\cdot\overline{C}\cdot\overline{D} + A\cdot B\cdot\overline{C}\cdot D + A\cdot B\cdot C\cdot\overline{D} + A\cdot B\cdot C\cdot D$$

i

$$H = \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D} + \overline{A}\cdot\overline{B}\cdot C\cdot\overline{D} + \overline{A}\cdot\overline{B}\cdot C\cdot D + \overline{A}\cdot B\cdot C\cdot D +$$
$$+ A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D} + A\cdot\overline{B}\cdot C\cdot\overline{D} + A\cdot\overline{B}\cdot C\cdot D + A\cdot B\cdot C\cdot D \qquad (8.28)$$

b) mintermi sa binarnim brojevima su:

$$F = 0001 + 0011 + 0101 + 0110 + 0111 + 1000 + 1001 + 1010 + 1101 + 1110 + 1111$$
$$G = 0001 + 0011 + 0110 + 1001 + 1011 + 1100 + 1101 + 1110 + 1111$$
$$H = 0000 + 0010 + 0011 + 0111 + 1000 + 1010 + 1011 + 1111 \qquad (8.29)$$

c) prikaz pomoću skupa indeksa:

$$F(A,B,C,D) = \sum^{4}(1,3,5,6,7,8,9,10,11,14,15)$$
$$G(A,B,C,D) = \sum^{4}(1,3,6,9,11,12,13,14,15)$$
$$H(A,B,C,D) = \sum^{4}(0,2,3,7,8,10,11,15) \qquad (8.30)$$

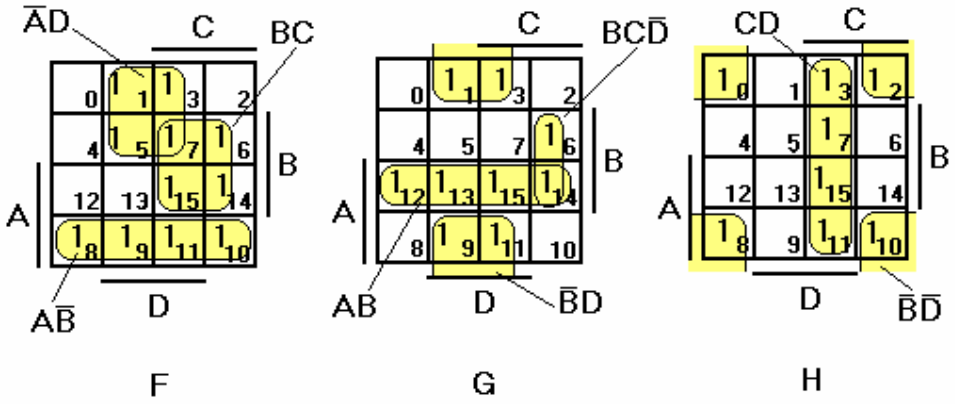
d) tablica istine:

Tabela 8.7: tablica istine funkcije iz primera 8.4

r.b.	A	B	C	D	F	G	H
0.	0	0	0	0	0	0	1
1.	0	0	0	1	1	1	0
2.	0	0	1	0	0	0	1
3.	0	0	1	1	1	1	1
4.	0	1	0	0	0	0	0
5.	0	1	0	1	1	0	0
6.	0	1	1	0	1	1	0
7.	0	1	1	1	1	0	1
8.	1	0	0	0	1	0	1
9.	1	0	0	1	1	1	0
10.	1	0	1	0	1	0	1
11.	1	0	1	1	1	1	1
12.	1	1	0	0	0	1	0
13.	1	1	0	1	0	1	0
14.	1	1	1	0	1	1	0
15.	1	1	1	1	1	1	1

e) Karnaugh-ove karte:

Karte za date logičke funkcuje imaju ukupno $p = 2^n = 2^4 = 16$ polja, to znači, da se karte sastoje od mreže pravougaonika sa 16 polja (slika 8.10).



Slika 8.9: Karnaugh-ove table sa minimizacijama

f) minimizirane funkcije su:

$$F = A \cdot \overline{B} + \overline{A} \cdot D + B \cdot C,$$
$$G = A \cdot B + \overline{B} \cdot D + B \cdot C \cdot \overline{D} \text{ i}$$
$$H = \overline{B} \cdot \overline{D} + C \cdot D \qquad (8.31)$$

☹ ☺ ☹

8.4 QUINE-OVA METODA MINIMIZACIJE

Quine-ova metoda minimizacije se bazira na primeni relacije:

$$A \cdot B + A \cdot \overline{B} = A \qquad (8.32)$$

Kod ove minimizacije, koja je pogodna i za minimizaciju funkcija sa 5 ili više promenljivih, obično polazimo od funkcije disjunktivnog normalnog oblika. Disjunktivnu normalnu formu dobijamo na osnovu tablice istine. Kod ove metode uvek polazimo od kanoničke forme (mintermi se zapisuju u formi koja sadrži sve promenljive – kanoničan term).

PRIMER 8.9:

Minimizirati funkciju $F(A,B,C,D) = \sum^{4}(0,4,8,12,15)$ pomoću Quine-ove metode.

Prvi korak kod ove minimizacije je formiranje tablice sa mintermima. Minterme treba redom uneti u kolonu tabele za minimizaciju. Svaki minterm ima redni broj, koji određuje mesto minterma u tabeli.

Prva tabela za ovaj primer data je na slici 8.10.

1. tabela		
i	mintermi	
0	$m_0 = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$	√
4	$m_4 = \overline{A} \cdot B \cdot \overline{C} \cdot \overline{D}$	√
8	$m_8 = A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$	√
12	$m_{12} = A \cdot B \cdot \overline{C} \cdot \overline{D}$	√
15	$m_{15} = A \cdot B \cdot C \cdot D$	

Slika 8.10: 1. polazna tabela za Quine-ovu minimizaciju

Na osnovu podataka iz prve tablice možemo formirati drugu tablicu (slika 8.11). Upoređivanjem svakog reda sa svakim redom treba pronaći one parove minterma koji se međusobno razlikuju samo za stanje jedne promenljive. Od datih parova se formira novi red u narednoj tabeli tako da se ispušta ona promenljiva koja se razlikuje u paru minterma. Indeksi redova sada označavaju redove od kojih je novi izraz proizašao. Svaki minterm kome smo našli «par» označimo sa kvačicom («√»). U tablici znak «-» (minus) na mestu promenljive ukazuje na to, da ta promenljiva više nije u funkciji.

2. tabela		
i,j		
0,4	$\overline{A}\cdot -\cdot \overline{C}\cdot \overline{D}$	√
0,8	$-\cdot \overline{B}\cdot \overline{C}\cdot \overline{D}$	√
4,12	$-\cdot B\cdot \overline{C}\cdot \overline{D}$	√
8,12	$A\cdot -\cdot \overline{C}\cdot \overline{D}$	√

Slika 8.11: 2. tabela kod Quine-ove minimizacije

Postupak ovim nije završen, već treba ponoviti prethodne korake u tablici br. 2, odnosno sa ovde nađenim novim parovima formirati 3. tabelu (slika 8.12).

3. tabela		
i,j,k,l		
0,4,8,12	$-\cdot -\cdot \overline{C}\cdot \overline{D}$	
0,8,4,12	$-\cdot -\cdot \overline{C}\cdot \overline{D}$	

Slika 8.12: 3. tablica kod Quine-ove minimizacije

Gore opisani postupak treba ponavljati sve dotle dok se ne iscrpe sve mogućnosti takvog eliminisanja pojedinih promenljivih. U ovom primeru dalja minimizacija nije moguća, jer u koloni tablice 3 ne postoji ni jedan red u kome bi se stanje samo jedne promenljive razlikovalo u odnosu na neki drugi red, pa je postupak završen.

U tablicama 1, 2 i 3 neki redovi nisu označeni sa znakom «√ » (kvačica), izrazi u ovim redovima su **prosti implikanti**, koji će činiti minimiziranu funkciju:

$$F = A\cdot B\cdot C\cdot D + \overline{C}\cdot \overline{D} \qquad (8.33)$$

Dobijeni izraz daje pojednostavljeniji oblik originalne funkcije, ali u većini slučajeva ne predstavlja i njenu minimalnu formu. Ovu funkciju je moguće i dalje minimizirati, ako između prostih implikanata nađemo onu najmanju grupu, koja daje funkciju u iredundantnom (neredundantnom) obliku.

Odabiranje parova pri ovoj minimizaciji olakšava upotreba tablica prostih implikanata u kojoj se može dobro uočiti veza između kanoničkih termova i prostih implikanata. Ova tabela ima toliko kolona koliki je broj kanoničkih termova, i onoliko redova koliki je broj nađenih prostih implikanata. Izgled ove tabele je dat na slici 8.13.

Tabela kanoničkih minterma i prostih implikanata						
Prosti implikanti		m_0	m_4	m_8	m_{12}	m_{15}
		$\overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D}$	$\overline{A}\cdot\overline{B}\cdot\overline{C}\cdot D$	$\overline{A}\cdot\overline{B}\cdot C\cdot\overline{D}$	$\overline{A}\cdot B\cdot\overline{C}\cdot\overline{D}$	$\overline{A}\cdot B\cdot C\cdot D$
15	$A\cdot B\cdot C\cdot D$					ⓧ
0,4,8,12	$-\cdot-\cdot\overline{C}\cdot\overline{D}$	ⓧ	ⓧ	ⓧ	ⓧ	
0,8,4,12	$-\cdot-\cdot\overline{C}\cdot\overline{D}$	x	x	x	x	

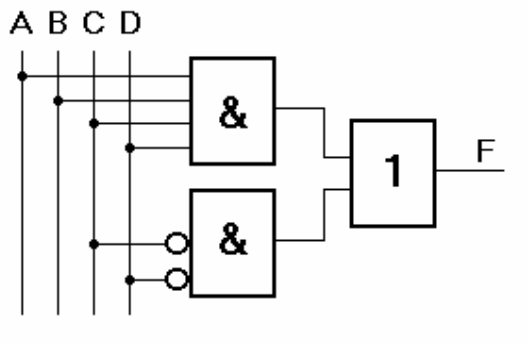
Slika 8.13: Tablica kanoničkih minterma i prostih implikanata

Propratimo ponaosob svaki red tablice (Slika 8.13) i označimo sa **x** svaki kvadratić čiji je kanonički term učestvovao u stvaranju implikanata dotičnog reda. Ako između kolona nađemo takve da u njima postoji samo jedna oznaka **x** (na primer kolona sa indeksom 15) tada je pripadajući red neophodan za realizaciju i prosti implikant koji označava taj red se naziva **bitni implikant**. Implikante minimalne forme označavamo sada tako, da su i originalni termovi normalne forme budu zastupljeni, drugim rečima svaki odabrani implikant mora imati bar jedan znak **x** u koloni. Zaokružimo sada bitne implikante i napišemo na onovu toga minimalni oblik funkcije:

$$F = A\cdot B\cdot C\cdot D + \overline{C}\cdot\overline{D} \qquad (8.34)$$

Na osnovu zaokruženih znakova ⓧ se vidi da je svaka kolona zastupljena.

Simbolička šema minimizirane funkcije je data na slici 8.14:



Slika 8.14: simbolička šema funkcije posle minimizacije

☹ ☺ ☹

PRIMER 8.9:

Minimizirati funkciju $F(A,B,C,D) = \sum^{4}(0,1,3,7,10,13,15)$ pomoću Quine-ove metode.

Radi iznalaženja prostih implikanata sastavimo tabelu od minterma (slika 8.15 i slika 8.16).

1. tabela		
i	mintermi	
0	$m_0 = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$	√
1	$m_1 = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D$	√
3	$m_3 = \overline{A} \cdot \overline{B} \cdot C \cdot D$	√
7	$m_7 = \overline{A} \cdot B \cdot C \cdot D$	√
10	$m_{10} = A \cdot \overline{B} \cdot C \cdot \overline{D}$	
13	$m_{13} = A \cdot B \cdot \overline{C} \cdot D$	√
15	$m_{15} = A \cdot B \cdot C \cdot D$	√

Slika 8.15: 1. tablica kod Quine-ove minimizacije

2. tabela		
i,j		
0,1	$\overline{A} \cdot \overline{B} \cdot \overline{C} \cdot -$	
1,3	$A \cdot \overline{B} \cdot - \cdot D$	
3,7	$\overline{A} \cdot - \cdot C \cdot D$	
7,15	$- \cdot B \cdot C \cdot D$	
13,15	$A \cdot B \cdot - \cdot D$	

Slika 8.15: 2. tablica kod Quine-ove minimizacije

Sastavimo tabelu prostih implikanata kao što je to prikazano na slici 8.17.

Prosti implikanti		m_0	m_1	m_3	m_7	m_{10}	m_{13}	m_{15}
10	$A \cdot \overline{B} \cdot C \cdot \overline{D}$					⊗		
0,1	$\overline{A} \cdot \overline{B} \cdot \overline{C} \cdot -$	⊗	⊗					
1,3	$\overline{A} \cdot \overline{B} \cdot - \cdot D$		x	x				
3,7	$\overline{A} \cdot - \cdot C \cdot D$			⊗	⊗			
7,15	$- \cdot B \cdot C \cdot D$				x			x
13,15	$A \cdot B \cdot - \cdot D$						⊗	⊗

Slika 8.17: Tabela kanoničkih minterma i prostih implikanata

Nakon zaokruživanja bitnih implikanata možemo napisati minimiziranu funkciju:

$$F = A \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot C \cdot D + A \cdot B \cdot D \qquad (8.35)$$

☹ ☺ ☹

9. LOGIČKA KOLA SA VIŠE IZLAZA

9.1 UVOD

Analizu, sintezu i minimizaciju logičko-prekidačkih kola napravili smo sa više ulaza i jednim izlazom. Realna upravljačka kola su međutim sa više ulaza i izlaza. U ovom poglavlju analiziramo takva kola.

☹ ☺ ☹

9.2 LOGIČKA KOLA SA VIŠE ULAZA I SA VIŠE IZLAZA

Na slici 9.1 je prikazana blok šema logičkog kola sa više ulaza i više izlaza.

Slika 9.1: blok šema logičkog kola sa više ulaza i više izlaza.

Izlazne funkcije logičkog kola sa slike 9.1 su:

$$\begin{aligned} X &= F_1(A,B,C,.....T) \\ Y &= F_2(A,B,C,.....T) \\ &. \\ &. \\ &. \\ W &= F_n(A,B,C,.....T) \end{aligned} \qquad (9.1)$$

Bilo koje logičko kolo sa više izlaza može se dobiti sintezom više logičkih kola sa jednim izlazom. Pojedinačno svaki izlaz logičkog kola sa više izlaza može biti predstavljen logičkom funkcijom. Prema tome, logičko kolo sa više izlaza ima onoliki broj logičkih funkcija koliki je broj izlaza.

Ako minimiziramo svaku logičku funkciju ponaosob, nije sigurno da će se dobiti i minimalan broj elemenata za realizaciju sistema sa više izlaza. Zbog toga je efikasnije sprovesti minimalizaciju paralelno za sve logičke funkcije sistema. Često u većini slučajeva postoje zajednički implikanti pojedinih logičkih funkcija, koje ako se iskoriste, zajednički daju mogućnost za dalju minimizaciju sistema. Zajednički implikanti logičkog kola se mogu odrediti Karnaugh-ovom ili Quine-ovom metodom minimizacije.

☹ ☺ ☹

PRIMER 9.1:

Logičko kolo sa četiri ulaza i sa dva izlaza opisana je sa izlaznim funkcijama:

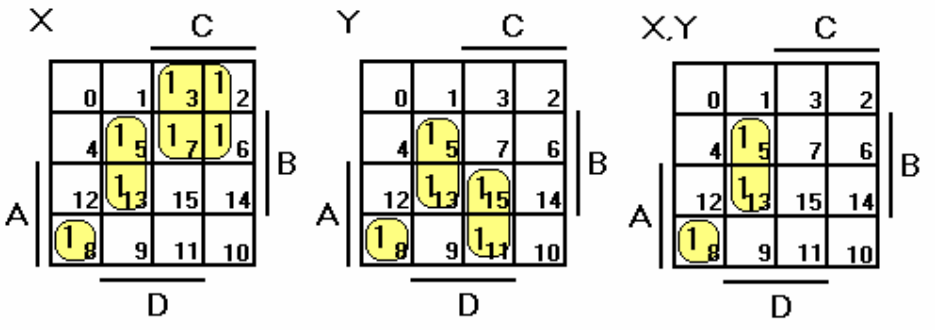
$$X(A,B,C,D) = \sum^{4}(2,3,5,6,7,8,13)$$
$$Y(A,B,C,D) = \sum^{4}(5,8,11,13,15) \qquad (9.2)$$

a) tablica istine logičkog kola sa više izlaza je data na slici 9.2.

Tablica istine						
	A	B	C	D	X	Y
0.	0	0	0	0	0	0
1.	0	0	0	1	0	0
2.	0	0	1	0	1	0
3.	0	0	1	1	1	0
4.	0	1	0	0	0	0
5.	0	1	0	1	1	1
6.	0	1	1	0	1	0
7.	0	1	1	1	1	0
8.	1	0	0	0	1	1
9.	1	0	0	1	0	0
10.	1	0	1	0	0	0
11.	1	0	1	1	0	1
12.	1	1	0	0	0	0
13.	1	1	0	1	1	1
14.	1	1	1	0	0	0
15.	1	1	1	1	0	1

Slika 9.2: tablica istine

b) minimizacija funkcije X i Y Karnaugh-ovom metodom je prikazana na slici 9.3.



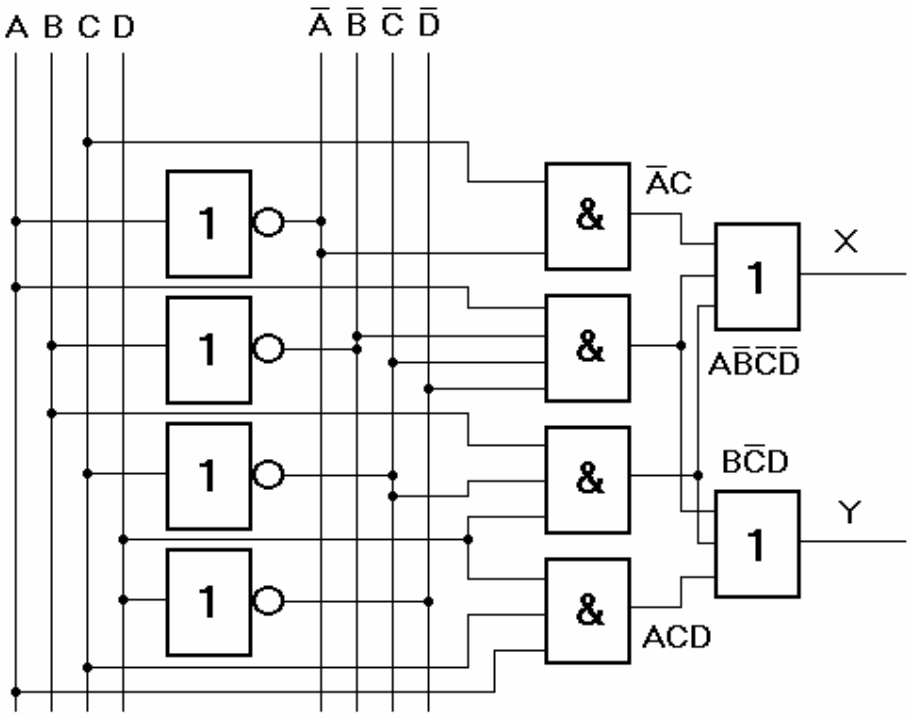
Slika 9.3: minimizacija funkcije X i Y i zajednički implikanti

Funkcije nakon minimizacije su:

$$X = \overline{A} \cdot C + B \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$$
$$Y = A \cdot C \cdot D + B \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} \qquad (9.3)$$

Očigledno je, da su u obe funkcije (X i Y) zadnji ($B \cdot \overline{C} \cdot D$) i predzadnji ($A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$) implikanti isti, pa je dovoljno jednom realizovati zajedničke implikante za obe funkcije.

c) šema logičkog kola sa četiri ulaza i sa dva izlaza ja data na slici 9.4.



Slika 9.4: realizacija logičkog kola sa četiri ulaza i sa dva izlaza

☹ ☺ ☹

PRIMER 9.2:

Logičko kolo sa četiri ulaza i sa tri izlaza opisana je sa izlaznim funkcijama:

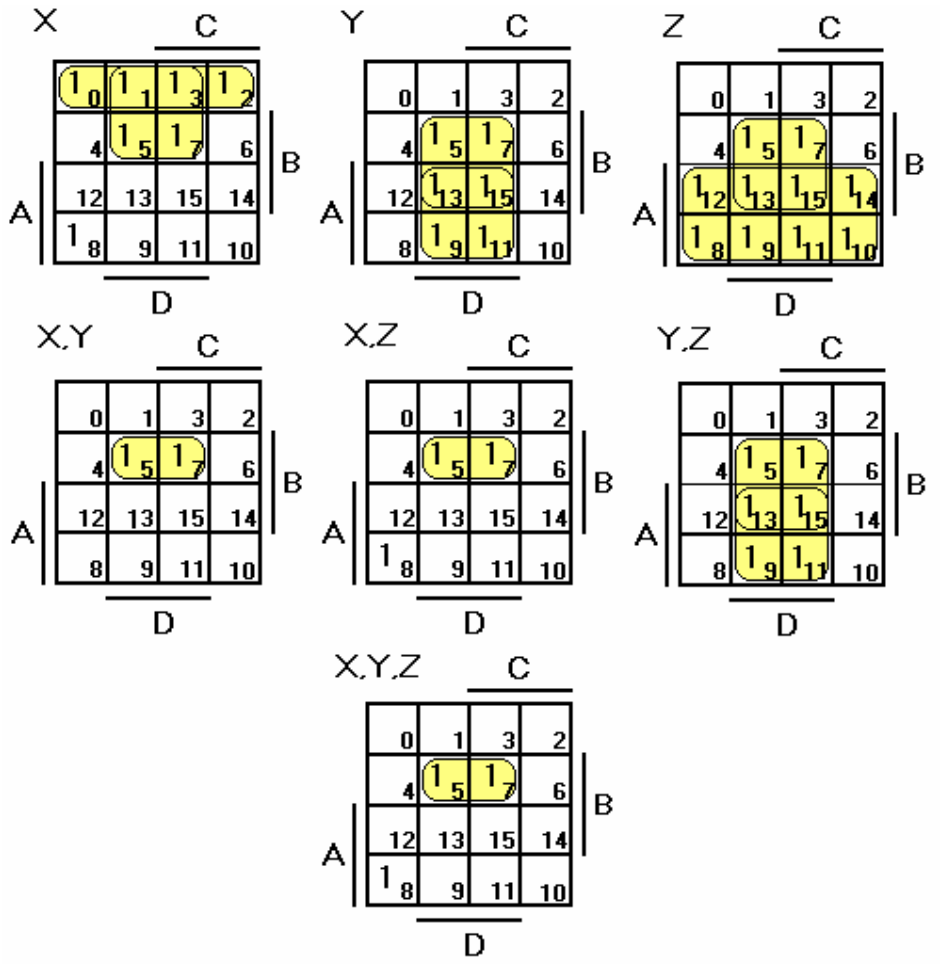
$$X(A,B,C,D) = \sum^{4}(0,1,2,3,5,7)$$
$$Y(A,B,C,D) = \sum^{4}(5,7,9,11,13,15) \qquad (9.4)$$
$$Z(A,B,C,D) = \sum^{4}(5,7,8,9,10,11,12,13,14,15)$$

d) tablica istine logičkog kola sa više izlaza je na slici 9.5.

Tablica istine							
	A	B	C	D	X	Y	Z
0.	0	0	0	0	1	0	0
1.	0	0	0	1	1	0	0
2.	0	0	1	0	1	0	0
3.	0	0	1	1	1	0	0
4.	0	1	0	0	0	0	0
5.	0	1	0	1	1	1	1
6.	0	1	1	0	0	0	0
7.	0	1	1	1	0	1	1
8.	1	0	0	0	0	0	1
9.	1	0	0	1	0	1	1
10.	1	0	1	0	0	0	1
11.	1	0	1	1	0	1	1
12.	1	1	0	0	0	0	1
13.	1	1	0	1	0	1	1
14.	1	1	1	0	0	0	1
15.	1	1	1	1	0	1	1

Slika 9.5: tablica istine

e) minimizacija funkcije X i Y Karnaugh-ovom metodom je prikazana na slici 9.6.



Slika 9.6: minimizacija funkcije X,Y i Z i zajednički implikanti

Pomoću Quine-ove tablice možemo nadalje minimizirati dobijene funkcije (slika 9.7)

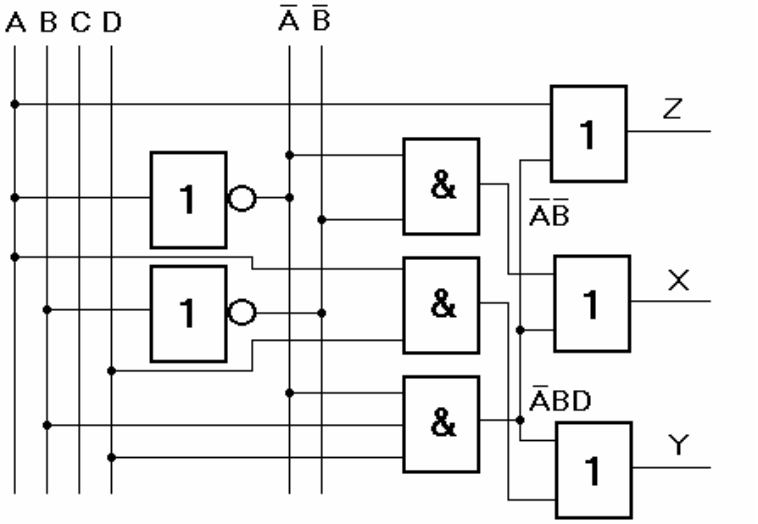
prosti i zajednički implikanti			X						Y						Z									
			0	1	2	3	5	7	5	7	9	11	13	15	5	7	8	9	10	11	12	13	14	15
a	X	$\overline{A}\cdot\overline{B}$	•	•	•	•																		
b		$\overline{A}\cdot D$		o		o	o	o																
c	Y	$B\cdot D$							o	o			o	o										
d		$A\cdot D$									•	•	•	•										
e	Z	A															•	•	•	•	•	•	•	•
f		$B\cdot D$													o	o						o		o
g	X,Y	$\overline{A}\cdot B\cdot D$					o	o	o	o														
h	X,Z	$\overline{A}\cdot B\cdot D$					o	o							o	o								
i	Y,Z	$B\cdot D$							o	o	o		o	o	o	o						o		o
j		$A\cdot D$									o	o	o	o				o		o		o		o
k	X,Y,Z	$\overline{A}\cdot B\cdot D$					•	•	•	•					•	•								

Slika 9.7: Quine-ova tablica minimizacije

Na osnovu bitnih implikanata označenih sa znakom «●» dobijamo minimalni oblik logičke funkcije upravljačkog uređaja:

$$\left.\begin{aligned} X &= \overline{A}\cdot\overline{B}+ \\ Y &= A\cdot D+ \\ Z &= A+ \end{aligned}\right\} \quad \overline{A}\cdot B\cdot D \qquad (9.5).$$

Simbolička šema sistema je prikazana na slici 9.8.



Slika 9.8: simbolička šema sistema

10. STANDARDNE KOMBINACIONE MREŽE

10.1 UVOD

Kombinacione mreže koristimo u svim digitalnim sistemima, u najjednostavnijim digitalnim uređajima isto tako, kao i u najsavršenijim računarima. U digitalnim sistemima neke funkcije su iste, specifične i standardizovane. Pojedini tipovi mreža (moduli) dobili su naziv prema funkciji koju obavljaju:

- dekoder,
- koder,
- multiplekser,
- demultiplekser,
- generator parnosti itd.

Analizu, sintezu i projektovanje ovih kombinacionih mreža radimo na isti način, kao štoje to prikazano u poglavlju 7.

10.2 DEKODERI

Dekoderi su kombinacione mreže sa više ulaza i sa više izlaza (9. poglavlje). U ovim mrežama svaka dozvoljena kombinacija ulaznih promenljivih aktivira poseban izlaz. Dekoderi mogu biti:

- potpuni – za **n** ulaznih promenljivih postoji 2^n izlaznih funkcija i
- nepotpuni – za **n** ulaznih promenljivih broj izlaznih funkcija manji od 2^n, određene kombinacije ulaznih promenljivih ne mogu pojaviti.

DEFINICIJA:

Dekoder je kombinaciona mreža, koja realizuje skup prekidačkih funkcija:

$$\begin{aligned} D_0 &= \overline{X_1} \cdot \overline{X_2} \cdot .....\overline{X_n} \cdot E, \\ D_1 &= \overline{X_1} \cdot \overline{X_2} \cdot .....X_n \cdot E, \\ &. \\ &. \\ D_{2^n-1} &= X_1 \cdot X_2 \cdot ...X_n \cdot E \end{aligned} \qquad (10.1)$$

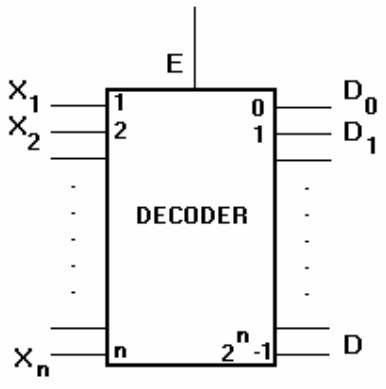
gde su:

$X_1, X_2, ...X_n$ - ulazi,

E - signal blokiranja (ENABLE),

$D_0, D_1, ...D_{2^n-1}$ - izlazni signali.

Kada je signal blokiranja nula (E = 0), tada su svi izlazni signali nule ($D_i = 0$), a kada je signal blokiranja jedinica (E = 1) onda samo jedan od izlaznih signala D_i ima vrednost jedinicu (1). Koji će to izlazni signal biti, jednoznačno je određno vrednostima ulaznih signala X_i, jer X_i predstavljaju potpune proizvode **n** prmenljivih. Grafički simbol dekodera (10.1) je dat na slici 10.1.



Slika 10.1: grafički simbol dekodera (10.1)

10.2.1 POTPUNI DEKODERI

Drugi naziv za potpuni dekoder je još i binarni dekoder, pošto su ulazne promenljive binarno kodirani brojevi, a kao što smo već videli, za svaku kombinaciju ulaznih promenljivih postoji jedan i samo jedan aktivan izlaz iz mreže.

PRIMER 10.1:

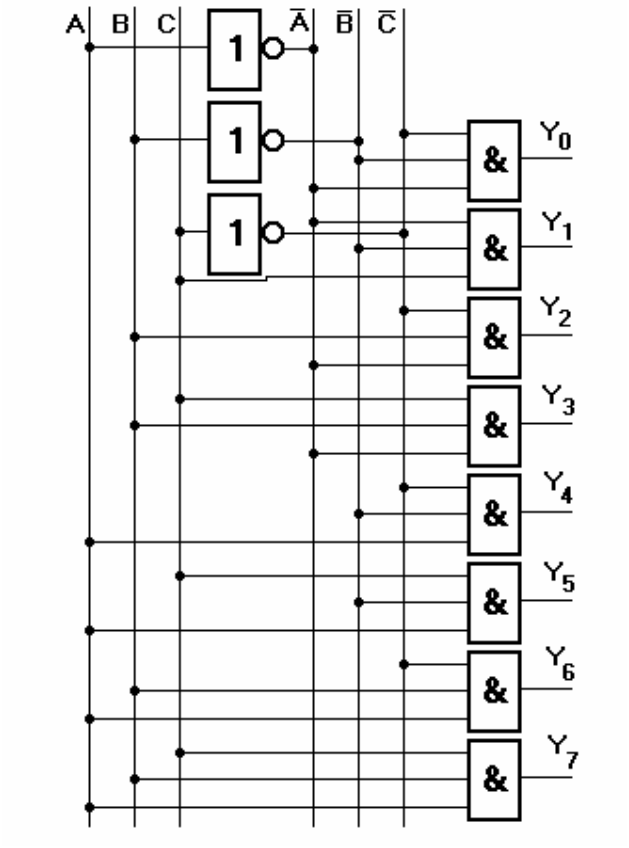
Projektovati dekoder sa n = 3 ulaza, ako su ulazi A,B i C.

Ako je broj ulaza n = 3, onda je ukupan broj kombinacija ulaznih promenljivih $2^3 = 8$. Tablica istine dekodera je data na slici 10.2

r.b.	A	B	C	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0.	0	0	0	0	0	0	0	0	0	0	1
1.	0	0	1	0	0	0	0	0	0	1	0
2.	0	1	0	0	0	0	0	0	1	0	0
3.	0	1	1	0	0	0	0	1	0	0	0
4.	1	0	0	0	0	0	1	0	0	0	0
5.	1	0	1	0	0	1	0	0	0	0	0
6.	1	1	0	0	1	0	0	0	0	0	0
7.	1	1	1	1	0	0	0	0	0	0	

Slika 10.2: tablica istine dekodera sa 3 ulaza iz primera 10.1

Iz tablice istine se vidi, da svaka izlazna funkcija ima samo po jedan član logičkog proizvoda, tako da se mreža može relizovati korišćenjem **I** kola i invertora, a minimizacija mreže je nemoguća. Logička mreža projektovanog dekodera je data na slici 10.3.

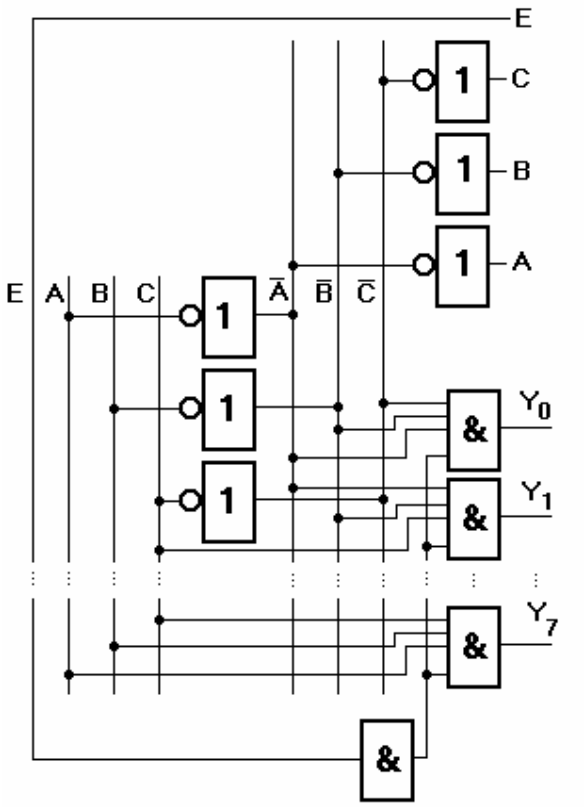


Slika 10.3: simbolička šema dekodera iz primera 10.1

Drugi nazivi za dekoder prikazan na slici su još:

- 3/8 dekoder ili
- 1 od 8 dekoder.

Pošto je teško obezbediti istovremeno menjanje ulaznih signala, na izlazima ovog kola će se često javljati pogrešan kod. Zbog ovog razloga, i zbog velikog opterećenja ulaza praktična realizacija ovog kola je sa ulaznim invertorima i sa **E** (ENABLE) sinhronizacionom signalom (slika 10.4). Signal dozvole **E** treba držati na nultom nivou za vreme dok ulazne promenljive menjaju vrednost, a na nivou **1** kada su ulazne promenljve stabilne.



Slika 10.4: simbolička šema dekodera 3/8 sa signalom dozvole **E** iz primera 10.1

PRIMER 10.2:

Na slici 10.6 je data simbolička šema 3/8 dekodera iz serije **TTL**, sa oznakom 74HC138. Ovde su korišćena umesto **I NI** kola za realizaciju pojedinih funkcija, pa zbog toga su izlazi neaktivni, kada su jedinice, a samo jedan izlaz je aktivan, kad je nula na izlazu (tablica istine je na slici 10.5, a kolo na slici 10.6).

r.b.	A_2	A_1	A_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0.	0	0	0	1	1	1	1	1	1	1	0
1.	0	0	1	1	1	1	1	1	1	0	1
2.	0	1	0	1	1	1	1	1	0	1	1
3.	0	1	1	1	1	1	1	0	1	1	1
4.	1	0	0	1	1	1	0	1	1	1	1
5.	1	0	1	1	1	0	1	1	1	1	1
6.	1	1	0	1	0	1	1	1	1	1	1
7.	1	1	1	0	1	1	1	1	1	1	1

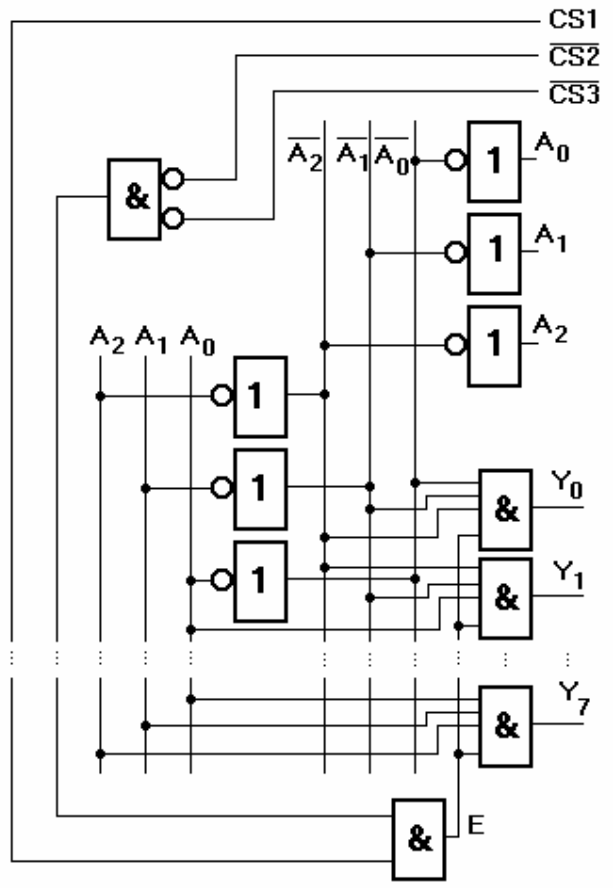
Slika 10.5: tablica istine dekodera 74HC138 3/8 iz primera 10.2

Vidi se, da i kod ovog kola svaki ulaz obezbeđuje jedinično ulazno opterećenje. Ako je signal E = 0, onda su svi izlazi dekodera neaktivni ($Y_0 = Y_1 = ... = Y_7 = 1$), bez obzira na to, koja je kombinacija ulaznih promenljivih prisutna.

Na slici 10.6 se vidi, da signal dozvole (**E**) je izlaz iz kombinacione mreže. Ova mreža je opisana jednačinom (10.2):

$$E = CH1 \cdot \overline{CH2} \cdot \overline{CH3} \,. \qquad (10.2)$$

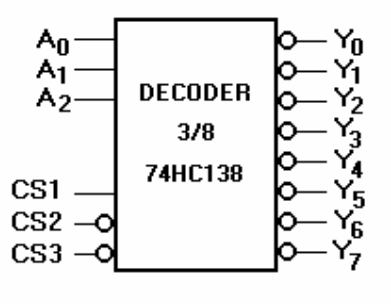
gde su:
E – ENABLE (signal dozvole),
CHx - CHIP SELECT (selekcija čipa) i
x - index ulaza.



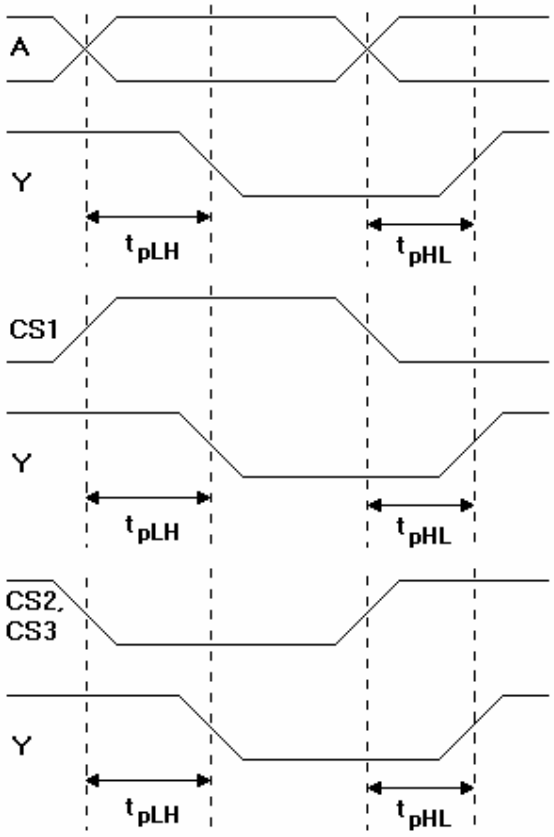
Slika 10.6: simbolička šema dekodera 3/8 74HC138 3/8 iz primera 10.2

U šemama digitalnih mreža za dekoder se koristi simbol, na kom su naznačeni ulazni, izlazni i kontrolni signali (slika 10.7).

Svako kolo realizuje kašnjenje. Kašnjenje utiče na brzinu rada sistema. Kod dekodera za bilo koji izlaz se definiše kašnjenje u odnosu na promenu ulaznih promenljivih, i isto tako i u odnosu na CS signale (slika 10.8).



Slika 10.7: logički simbol dekodera 3/8 74HC138

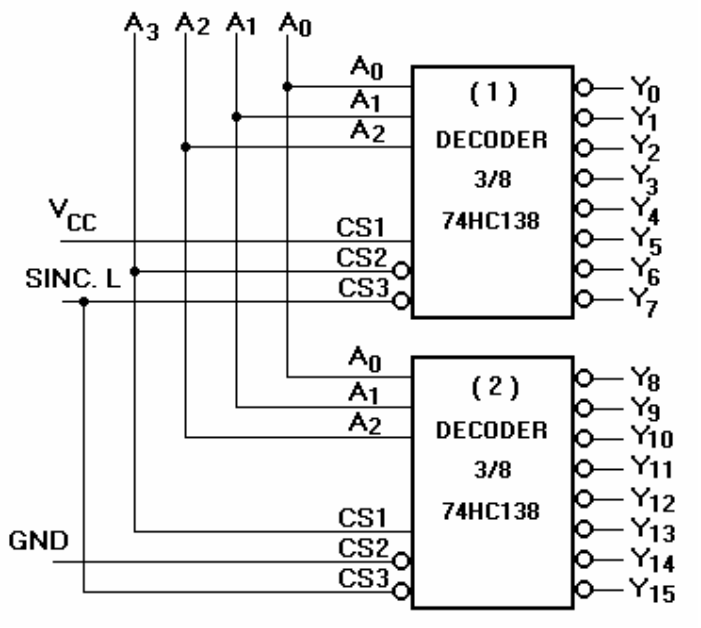


Slika 10.8: vremenski dijagrami kašnjenja dekodera 3/8 74HC138

☺ ☹ ☺

PRIMER 10.3:

Međusobnim povezivanjem dva 3/8 dekodera možemo realizovati jedan 4/16 dekoder (slika 10.9).



Slika 10.9: dekoder 4/16 realizovan dekoderima 3/8

Očigledno je, da će ulazni signal A_3 odrediti, da li će kolo (1) ili kolo (2) raditi, a da pri tom i dalje postoji signal dozvole, sad sa oznakom SINC.L.

Dekodovanje binarnih brojeva sa praktično neograničenim brojem cifara je moguće kaskadnim vezivanjem više dekodera.

☺ ☹ ☺

10.2.2 NEPOTPUNI DEKODERI

Kod nepotpunih dekodera određene kombinacije ulaznih promenljivih se ne mogu pojaviti na ulazu kombinacione mreže. Princip realizacije nepotpunog dekodera je isti kao i za potpuni, samo što će realizacije izlaznih funkcija za nepostojeće kombinacije biće izostavljene.

☺ ☹ ☺

PRIMER 10.4:

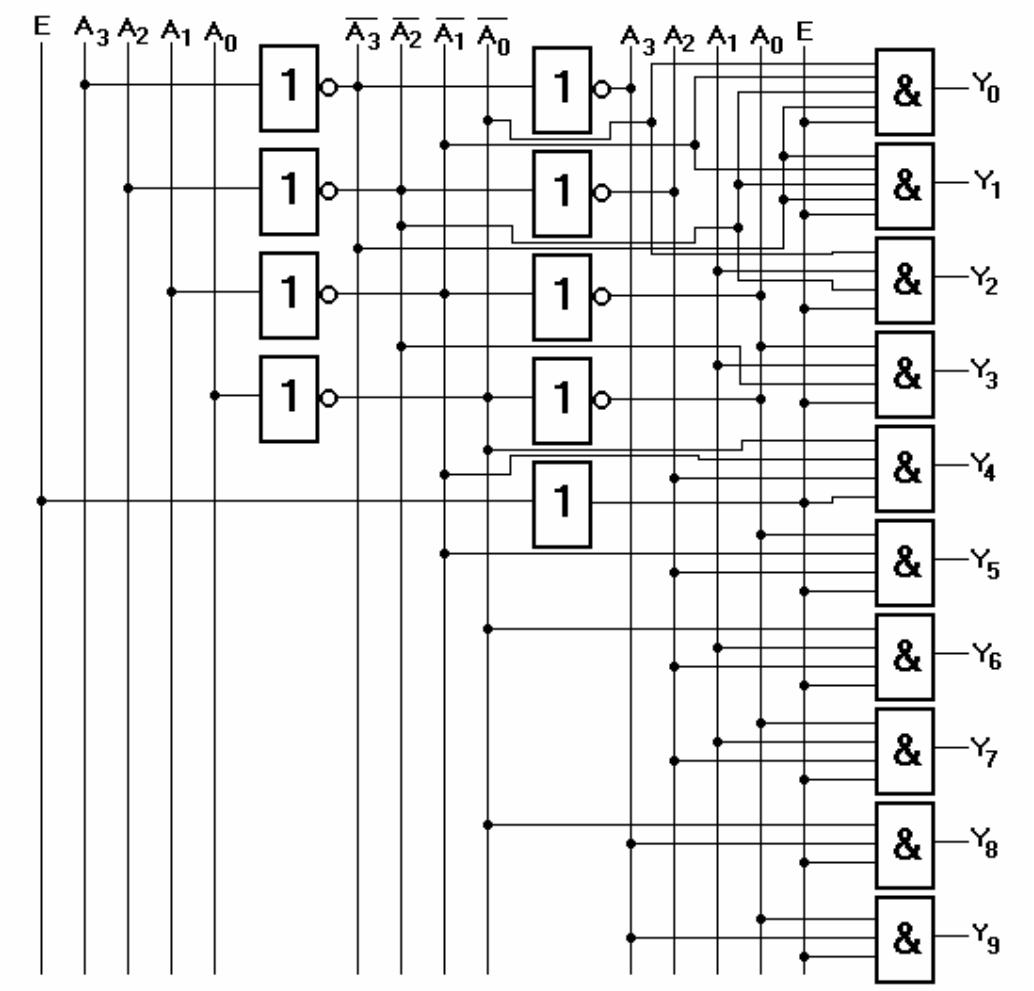
BCD dekoder dekoduje binarno kodovanu decimalnu cifru. Kombinaciona tabela **BCD** dekodera je data na slici 10.10, gde su binarne (ulazne promenljive) A_3, A_2, A_1 i A_0, odnosno izlazi decimalni brojevi (Y_0 do Y_9). Treba napomenuti, da kombinacije koje dekoduju logičke proiyvode 10 do 15 ne mogu da se pojave u **BCD** kodu, a to znači da se kombinaciona mreža može minimizirati. Na slici 10.11 je data logička šema minimiziranog **BCD** dekodera.

A_3	A_2	A_1	A_0	Y_9	Y_8	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_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	1	0
0	0	1	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0
1	0	1	0	X	X	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X	X

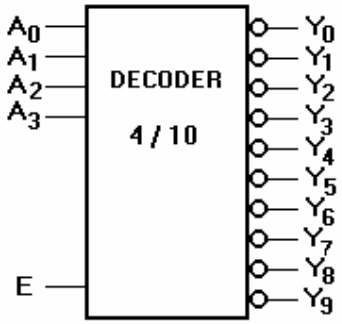
Slika 1.10: kombinaciona tabela **BCD** dekodera

Integrisani **BCD** dekoderi imaju signal dozvole (**E**). Simbol dekodera je na slici 10.12. Za označavanje **BCD** dekodera često se koriste još i izrazi:

- 1 od 10 dekoder,
- 4/10 dekoder i
- BCD/DC (binarno kodovane decimalne u decimalne cifre).



Slika 10.11: logička šema minimiziranog **BCD** dekodera



Slika 10.12: simbolička šema **BCD** dekodera

10.3 KODERI

U digitalnim sistemima obrada podataka je moguća samo onda, ako su informacije kodovane u digitalnu formu, tj. Ako su date u obliku nule i jedinice. Na primer, aktiviranje tastera znači, da odgovarajuća kombinaciona mreža generiše kombinaciju nule i jedinica koja odgovara tom tasteru. Logička mreža, koja realizuje dati zadatak je **koder** (encoder).

Koderi su:

- potpuni, ili binarni, ako je broj ulaza 2^n i broj izlaza **n**, odnosno
- nepotpuni, ako ima **n** izlaza, a broj ulaza je manji od 2^n.

Koder na izlazima generiše izlazni kod, koji može da bude:

- prirodni binarni i
- kod koji je zadat tabelom, gde svakom signalu (karakteru) odgovara određena kombinacija nula i jedinica.

10.3.1 POTPUNI DEKODERI

Na osnovu tablice istine kodera možemo napraviti sintezu kombinacione mreže.

PRIMER 10.5:

Na slici 10.13 je data tablica istine trobitnog kodera, koji ima 8 ulaza. Na osnovu tabele možemo napisati logičke funkcije izlaza:

$$\begin{aligned} Y_0 &= A_1 + A_3 + A_5 + A_7 \\ Y_1 &= A_2 + A_3 + A_6 + A_7 \\ Y_2 &= A_4 + A_5 + A_6 + A_7 \end{aligned} \qquad (10.3)$$

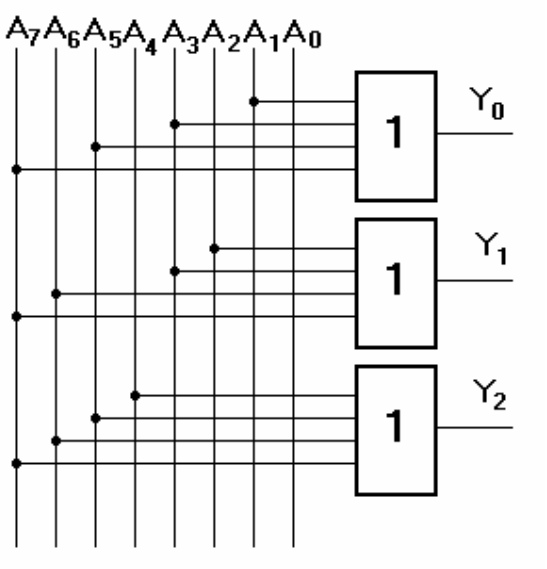
Za oznaku kodera često se koriste još i izrazi:

- trobitni koder i
- koder 8/3.

r.b.	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Y_2	Y_1	Y_0
0.	0	0	0	0	0	0	0	1	0	0	0
1.	0	0	0	0	0	0	1	0	0	0	1
2.	0	0	0	0	0	1	0	0	0	1	0
3.	0	0	0	0	1	0	0	0	0	1	1
4.	0	0	0	1	0	0	0	0	1	0	0
5.	0	0	1	0	0	0	0	0	1	0	1
6.	0	1	0	0	0	0	0	0	1	1	0
7.	1	0	0	0	0	0	0		1	1	1

Slika 10.13: tablica istine trobitnog kodera

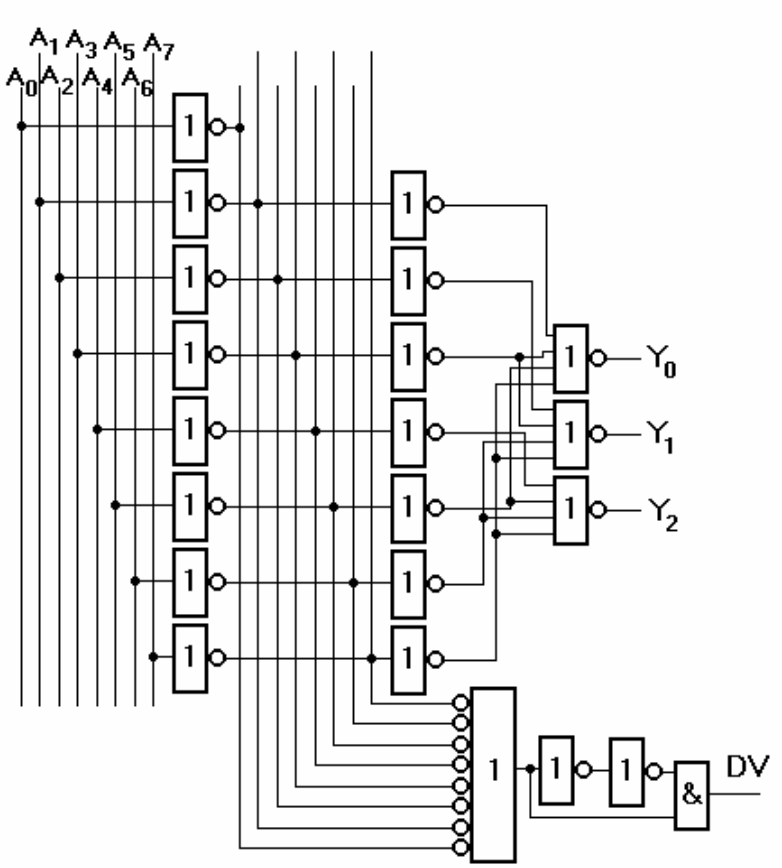
Simbolička šema trobitnog kodera je na slici 10.14.



Slika 10.14: simbolička šema trobitnog kodera (8/3)

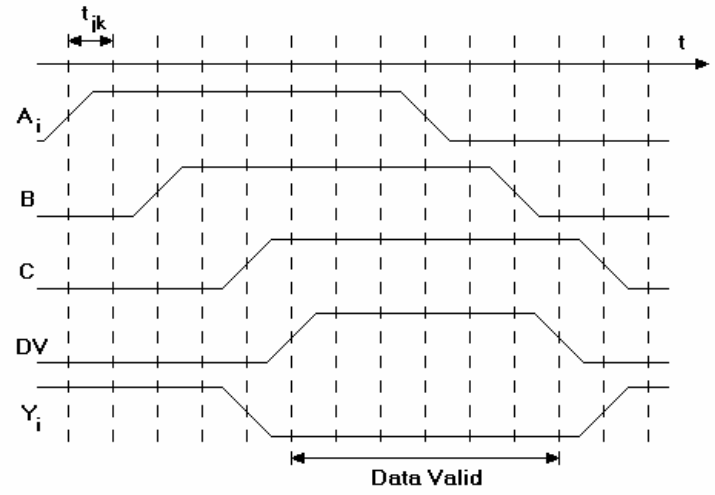
Funkcija kodera je suprotna funkciji dekodera. Na ulazu je aktivan samo jedan signal od ukupno 2^n signala, a na izlazu kola je binarni broj od **n** bita. Broj **ILI** kola kod kodera $2^n / n$ je **n**, dok svako izlazno kolo ima 2^{n-1} ulaza.

Logička mreža kodera (slika10.14) ne može da radi ispravno, iz dva razloga. Signal A_0 praktično nije uključen u koder, to znači, da su izlazne kombinacije kodera uvek iste, ako je $A_0 = 1$, a i onda, ako ni jedan A_i signal nije aktivan. Drugi razlog za neispravan rad je tehničkog karaktera, vreme propagacije kod svakog ILI kola se razlikuje, pa zbog toga postoji vremenski period, kad izlazni kod ne odgovara ulaznom kodu. Zbog ovih razloga kod kodera je dodata još jedna izlazna linija za sinhronizaciju **DV** (Data Valid = podaci važe) (slika 10.15).

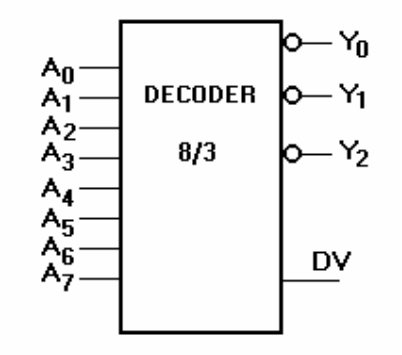


Slika 10.15: koder 8/3 sa izlaznim sinhronizacionim signalom

Ulazni invertori obezbeđuju faktor opterećenja **1** (slika 10.15). Invertori između **NILI** kola i **I** kola od **B** signala realizuju **C** signal , koji signal kasni u odnosu na **B** signal. **B** i **C** pomoću **I** kola daju **DV** signal (slika 10.16).



Slika 10.16: vremenski dijagram koder 8/3



Slika 10.17: simbolčka šema kodera 8/3

10.3.2 NEPOTPUNI KODERI

Kao što smo već objasnili, sinteza nepotpunog kodera je identična sintezi potpunog kodera, sa razlikom da je u slučaju **n** ulaza broj izlaza manji od 2^n.

PRIMER 10.6:

Na slici 10.18 je data tablica istine **DC/BCD** kodera. Iz tablice se vidi, da koder ima 10 ulaza i 4 izlaza.

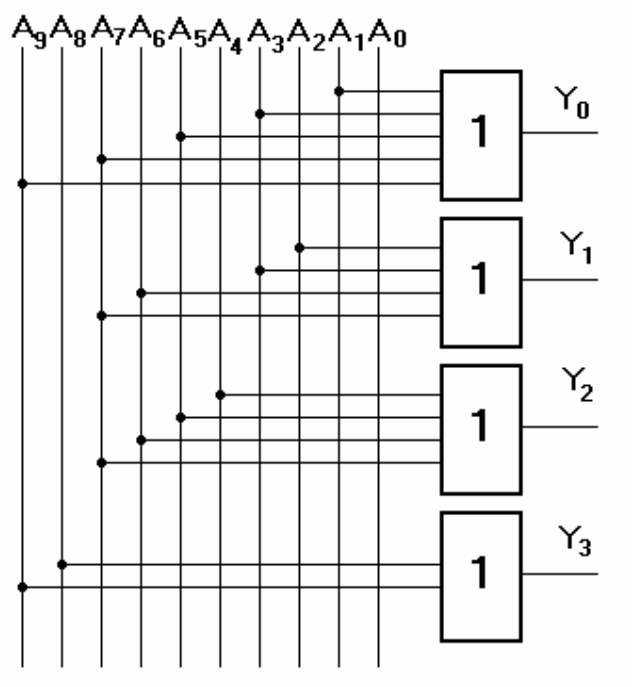
rb.	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0.	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1.	0	0	0	0	0	0	0	0	1	0	0	0	0	1
2.	0	0	0	0	0	0	0	1	0	0	0	0	1	0
3.	0	0	0	0	0	0	1	0	0	0	0	0	1	1
4.	0	0	0	0	0	1	0	0	0	0	0	1	0	0
5.	0	0	0	0	1	0	0	0	0	0	0	1	0	1
6.	0	0	0	1	0	0	0	0	0	0	0	1	1	0
7.	0	0	1	0	0	0	0	0	0	0	0	1	1	1
8.	0	1	0	0	0	0	0	0	0	0	1	0	0	0
9.	1	0	0	0	0	0	0	0	0	0	1	0	0	1

Slika 10.18: tablicaistine **DC/BCD** kodera

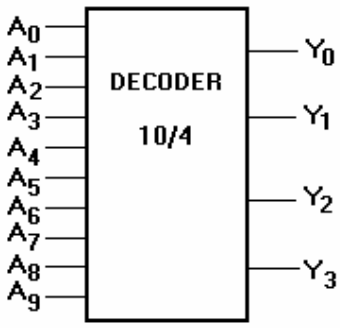
Iz tablice istine **DC/BCD** kodera možemo ispisati funkcije izlaza Y_i:

$$\begin{aligned} Y_0 &= A_1 + A_3 + A_5 + A_7 + A_9 \\ Y_1 &= A_2 + A_3 + A_6 + A_7 \\ Y_2 &= A_4 + A_5 + A_6 + A_7 \\ Y_3 &= A_8 + A_9 \end{aligned} \qquad (10.4)$$

Na osnovu jednačina (10.4) možemo realizovati mrežu kodera (slika 10.19).



Slika 10.19: mreža **DC/BCD** kodera



Slika 10.20: simbolička šem kodera **DC/BCD**

10.4 KONVERTORI KODA

Često treba pretvoriti informaciju iz jednog kodnog sistema u drugi. Mreže, koje realizuju ovaj zadatak nazivaju se konvertori koda. Konvertor koda se može realizovati, ili kao kaskadna veza dekodera i kodera, ili jednostavnije, minimizacijom funkcije konverzije koda.

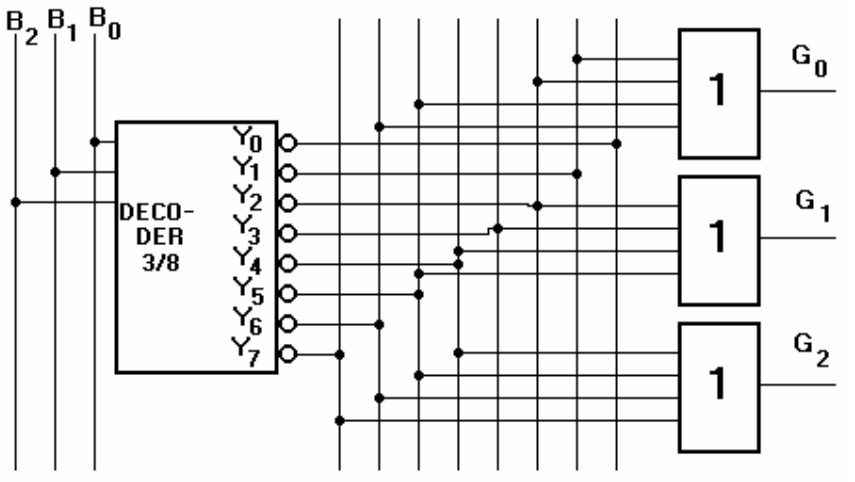
PRIMER 10.7:

Za pretvaranje binarnog koda u Grayov kod prvo formiramu tablicu istine (tabelu konverzije). Ako su kodovi trobitni, onda ulazni binarni kod je kodovan sa B_2, B_1, B_0, a izlazni Grayov kod je kodovan sa G_2, G_1, G_0 (slika 10.21).

broj	B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0	0
1	0	0	1	0	0	1
2	0	1	0	0	1	1
3	0	1	1	0	1	0
4	1	0	0	1	1	0
5	1	0	1	1	1	1
6	1	1	0	1	0	1
7	1	1	1	1	0	0

Slika 10.21: tablica istine konvertora koda iz binarnog u Grayov

Kod prvog rešenja (slika 10.22) dekoder je običan 3/8 dekoder, a drugi stepen je Grayov koder 8/3 .



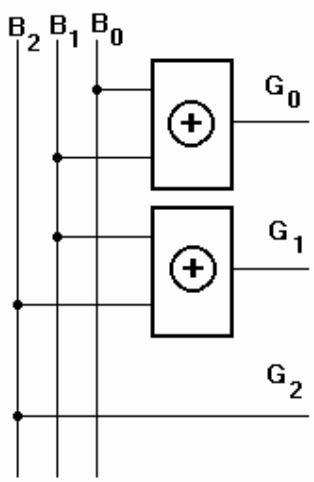
Slika 10.22: konvertor binarnog koda u Grayov kod sa dekoderom 3/8

Možemo i jednostavnije realizovati konvertor koda B/G, ako minimiziramo funkcije izlaza:

$$G_0 = \overline{B_2} \cdot \overline{B_1} \cdot B_0 + \overline{B_2} \cdot B_1 \cdot \overline{B_0} + B_2 \cdot \overline{B_1} \cdot B_0 + B_2 \cdot B_1 \cdot \overline{B_0} = B_1 \cdot \overline{B_0} + \overline{B_1} \cdot B_0 = B_1 \oplus B_0$$
$$G_1 = \overline{B_2} \cdot B_1 \cdot \overline{B_0} + \overline{B_2} \cdot B_1 \cdot B_0 + B_2 \cdot \overline{B_1} \cdot \overline{B_0} + B_2 \cdot \overline{B_1} \cdot B_0 = B_2 \cdot \overline{B_1} + \overline{B_2} \cdot B_1 = B_2 \oplus B_1$$
$$G_2 = B_2 \cdot \overline{B_1} \cdot \overline{B_0} + B_2 \cdot \overline{B_1} \cdot B_0 + B_2 \cdot B_1 \cdot \overline{B_0} + B_2 \cdot B_1 \cdot B_0 = B_2$$

(10.5)

Simbolička šema mreže (10.5) je na slici 10.23.

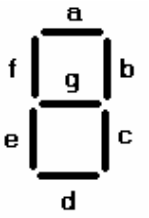


Slika 10.23: konvertor binarnog koda u Grayov kod sa ekskluzivno ILI elementima

☺ ☹ ☺

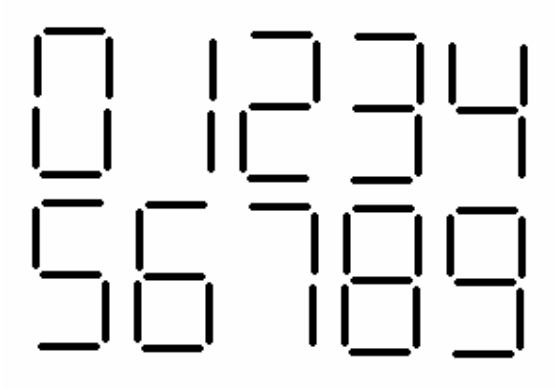
PRIMER 10.7:

Za prikazivanje cifara u digitalnim uređajima često koristimo sedmosegmentni pokazivač (display), slika 10.24.



Slika 10.24: označavanje segmenata kod sedmosegmentnog pokazivača

Oblik cifara kod sedmosegmentnog pokazivača je prikazan na slici 10.25



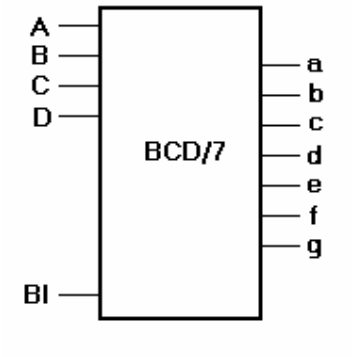
Slika 10.25: brojevi kod sedmosegmentnog pokazivača

Na osnovu slike 10.25 možemo ispuniti tablicu istine konvertora **BCD/7**, sa dodatnim signalom **BI** (Blanking Input), za deaktiviranje segmenta (slika 10.26).

cifra	BI	D	C	B	A	a	b	C	d	e	f	g
X	0	X	X	X	X	0	0	0	0	0	0	0
0	1	0	0	0	0	1	1	1	1	1	1	0
1	1	0	0	0	1	0	1	1	0	0	0	0
2	1	0	0	1	0	1	1	0	1	1	0	1
3	1	0	0	1	1	1	1	1	1	0	0	1
4	1	0	1	0	0	0	1	1	0	0	1	1
5	1	0	1	0	1	1	0	1	1	0	1	1
6	1	0	1	1	0	1	0	1	1	1	1	1
7	1	0	1	1	1	1	1	1	0	0	0	0
8	1	1	0	0	0	1	1	1	1	1	1	1
9	1	1	0	0	1	1	1	1	1	0	1	1

Slika 10.26: tablica istine konvertora koda **BCD/7**

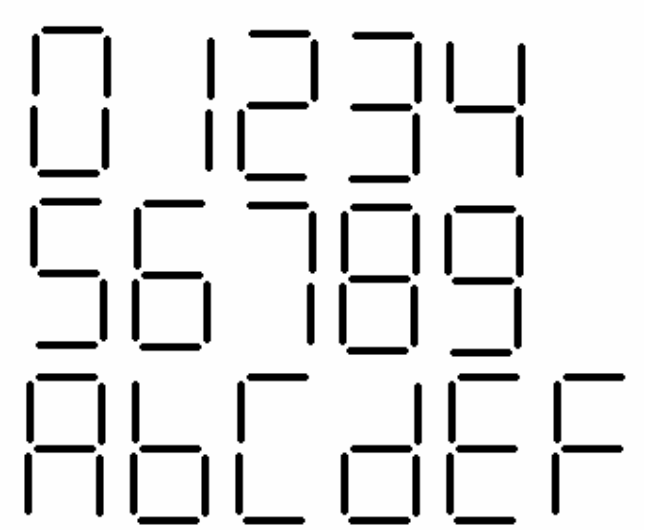
Simbolička šema **BCD/7** konvertora koda je data na slici 10.27.



Slika 10.27: simbolička šema **BCD/7** konvertora koda

PRIMER 10.8:

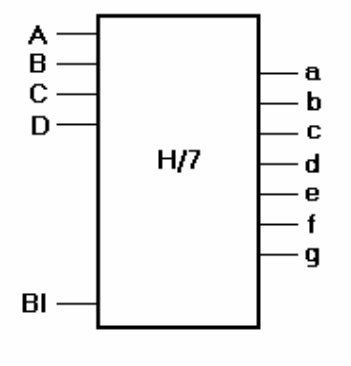
Postoji sličan konvertor koda kao **BCD/7**, ali ovaj se zove **H/7** konvertor, jer pretvara heksadecimalni kod u cifre **0-9** i slova **A, b, C, d, E** i **F**. Cifre i slova konvertora je prikazana na slici 10.28, tablica istine konvertora **H/7** je prikazana na slici 10.29, a simbolička šema **H/7** konvertora je data na slici 10.30.



Slika 10.28: brojevi i slova kod sedmosegmentnog pokazivača

cifra	BI	D	C	B	A	a	b	c	d	e	f	g
X	0	X	X	X	X	0	0	0	0	0	0	0
0	1	0	0	0	0	1	1	1	1	1	1	0
1	1	0	0	0	1	0	1	1	0	0	0	0
2	1	0	0	1	0	1	1	0	1	1	0	1
3	1	0	0	1	1	1	1	1	1	0	0	1
4	1	0	1	0	0	0	1	1	0	0	1	1
5	1	0	1	0	1	1	0	1	1	0	1	1
6	1	0	1	1	0	1	0	1	1	1	1	1
7	1	0	1	1	1	1	1	1	0	0	0	0
8	1	1	0	0	0	1	1	1	1	1	1	1
9	1	1	0	0	1	1	1	1	1	0	1	1
A	1	1	0	1	0	1	1	1	0	1	1	1
b	1	1	0	1	1	0	0	1	1	1	1	1
C	1	1	1	0	0	1	0	0	1	1	1	0
d	1	1	1	0	1	0	1	1	1	1	0	1
E	1	1	1	1	0	1	0	0	1	1	1	1
F	1	1	1	1	1	1	0	0	0	1	1	1

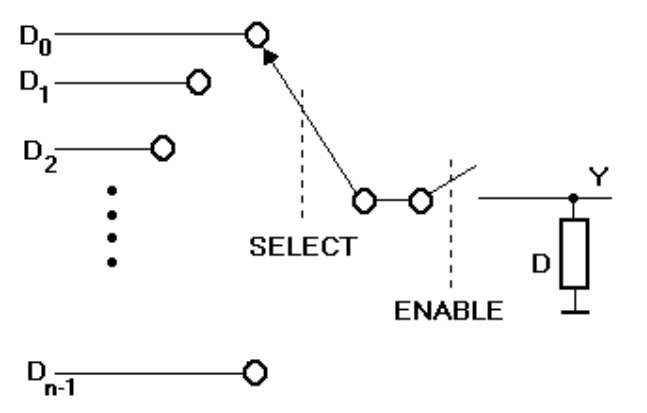
Slika 10.29: tablica istine konvertora koda **H/7**



Slika 10.30: simbolička šema **BCD/7** konvertora koda

10.5 MULTIPLEKSERI

Multiplekser je digitalni višepoložajni prekidač (slika 10.31).

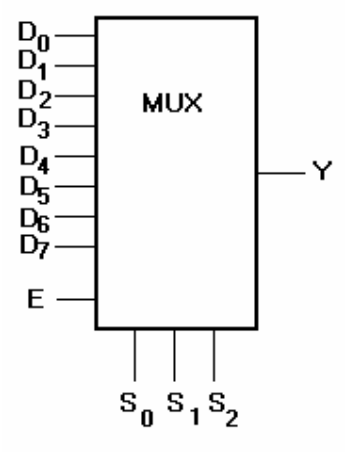


Slika 10.31: funkcionalna šema multipleksera

Jedan od **n** ulaza **SELECT** (selekcija) signal priključuje se na izlaz **Y**. Ulazni signal može da se pojavi na izlazu **Y** samo tada, ako je signal dozvole (**ENABLE**) jedinica (1).

PRIMER 10.9:

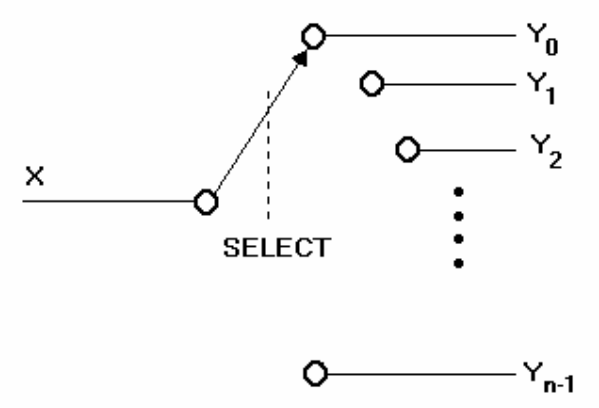
Na slici 10.32 je data simbolička šema multipleksera sa 8 ulaza.



Slika 10.32: simbolička šema multipleksera sa 8 ulaza

10.6 DEMULTIPLEKSERI

Demultiplekser je digitalni višepoložajni prekidač (slika 10.33).



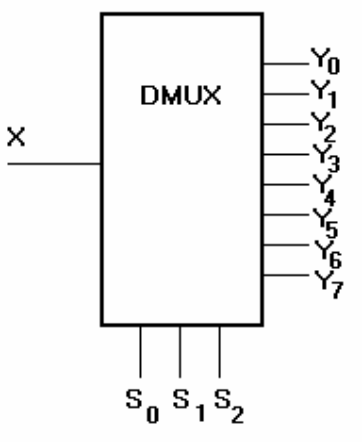
Slika 10.33: funkcionalna šema multipleksera

Digitalni ulazni signal **X** će se pojaviti na Y_i-tom izlazu. Indeks **i** će odrediti upravljački signal **SELECT**.

☺ ☹ ☺

PRIMER 10.10:

Na slici 10.34 je data simbolička šema multipleksera sa 8 ulaza.



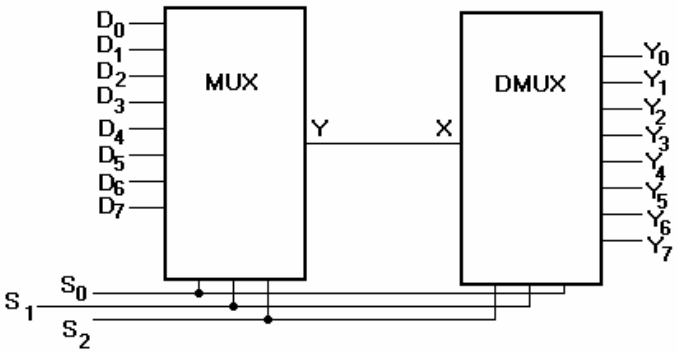
Slika 10.34: simbolička šema demultipleksera sa 8 izlaza

10.7 PRENOS DIGITALNIH INFORMACIJA KORIŠĆENJEM MULTIPLEKSERA I DEMULTIPLEKSERA

Prenos informacije pomoću multipleksera i demultipleksera omogućavanja korišćenje redukovanog broja prenosnih linija. Ako je broj linija $n = 2^m$, onda je potrebno imati svega $m+1$ linija za prenos informacije.

PRIMER 10.11:

Na slici 10.35 je data šema prenosa 8 digitalnih podataka sa multipekserom i demultiplekserom.



Slika 10.35: prenos digitalne informacije putem multipleksera i demultipleksera

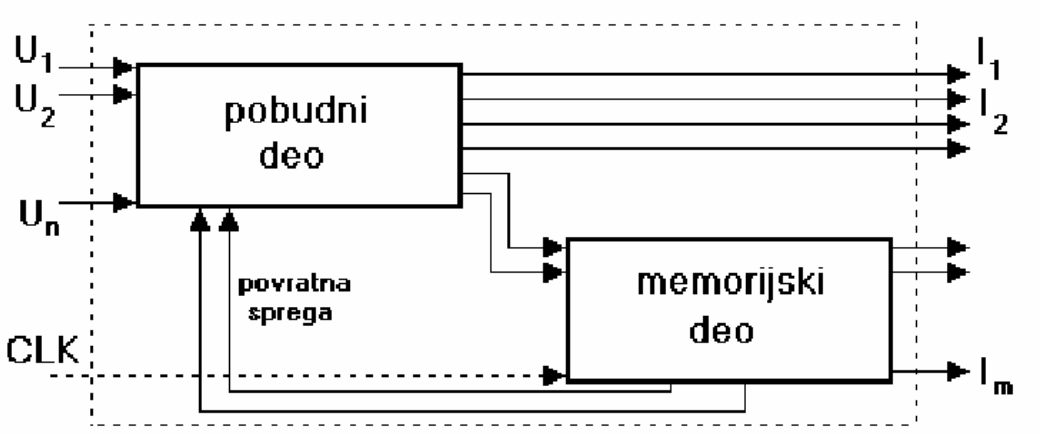
11. SEKVENCIJALNE MREŽE

11.1 UVOD

Sekvencijalna mreža (ili mreža sa memorijom) je logička mreža kod koje izlazi zavise:

- od trenutne vrednosi ulaznih promenljivih (ulaza) i
- od stanja, u kom se mreža momentalno nalazi.

Blok šema sekvencijalne mreže je data na slici 11.1.



Slika 11.1: blok šema sekvencijalne mreže

Na slici blokovi imaju sledeće zadatke:

- memorijski deo – pamti informaciju o stanju mreže i
- pobudni deo – obezbeđuje promenu stanja.

☺ ☹ ☺

11.2 SINHRONE I ASINHRONE SEKVENCIJALNE MREŽE

Sekvencijalne mreže mogu biti:

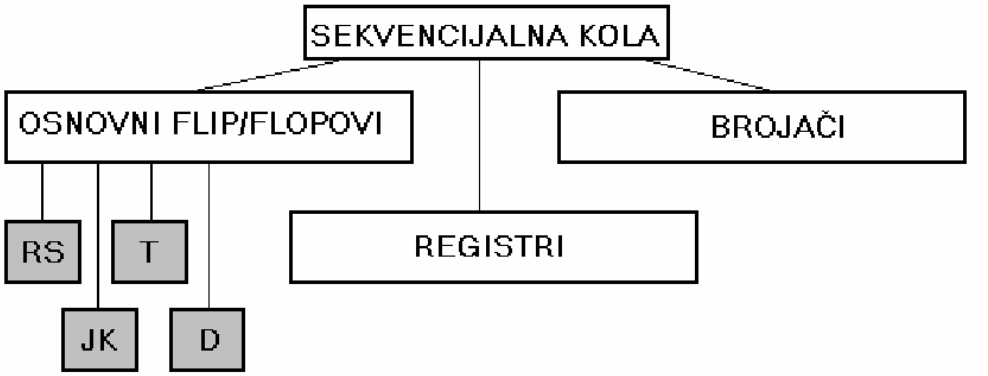
- **sinhrone mreže** – ako postoji sinhronizacioni signal **CLK** (Clock) koji »diktira« trenutak prelaza stanja mreže, i
- **asinhrone mreže** – ako ponašanje mreže određuju promene vrednosti ulaznih signala, a te promene se mogu desiti u bilo kom vremenskim trenutcima.

☺ ☹ ☺

11.3 MEMORIJSKI ELEMENTI SEKVENCIJALNE MREŽE

Elementarna memorijska ćelija sekvencijalne mreže je »flip-flop« koji može da čuva binarnu informaciju u obliko bit-a (**0** ili **1**), a ovu informaciju menja pod određenim uslovima.

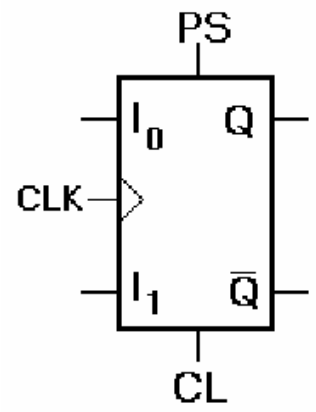
Sekvencijalne mreže možemo deliti na način, kako je to prikazano na slici 11.2.



Slika 11.2: sekvencijalne mreže

11.4 OSNOVNI FLIP-FLOPOVI

Opšta šema flip-flopa sa dva ulaza je data na slici 11.3.



Slika 11.3: opšta šema flip-flopa sa dva ulaza

Korišćene oznake su sledeće:

- I_0, I_1 - ulazi (pobudni ulazi),
- $Q, \overline{Q}$ - izlazi (stanje flip-flopa),
- CLK – sinhronizacija (kontrolni signal),
- PS (PRESET) – početno stanje flip-flopa (Q=1), (kontrolni signal) i
- CL (CLEAR) – početno stanje flip-flopa (Q=0), (kontrolni signal).

☺ ☹ ☺

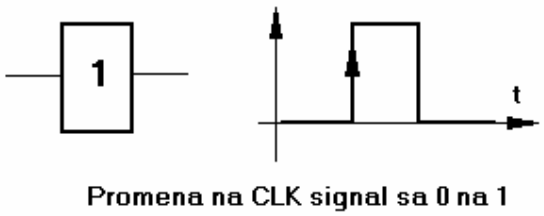
11.4.1 NAČINI SINHRONIZACIJE FLIP-FLOPOVA

Sinhronizacija flip-flopa može da bude na:

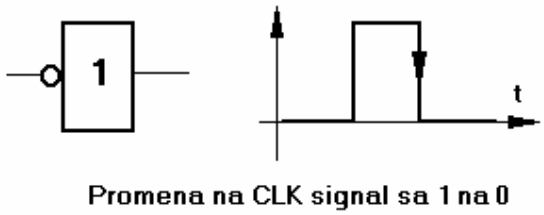
- okidanje sa ivicom i
- okidanje sa nivoom .

Ivično okidanje može da bude sa:

- pozitivnom ivicom (slika 11.4) i
- negativnom ivicom (slika 11.5).



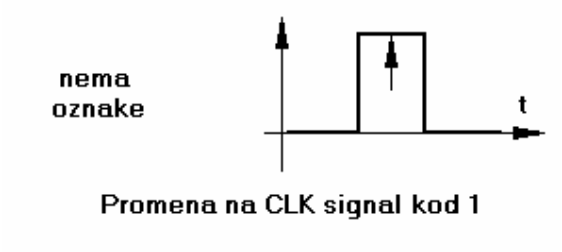
Slika 11.4: pozitivnom ivicom okidan flip-flop i vremenski dijagram



Slika 11.5: negativnom ivicom okidan flip-flop i vremenski dijagram

Nivoom okidani flip-flop može da bude sa:

- pozitivnim nivoom okidan flip-flop (slika 11.6) i
- negativnim nivoom okidan flip-flop (slika 11.7).



Slika 11.6: pozitivnom nivoom okidan flip-flop i vremenski dijagram

Slika 11.7: negativnom nivoom okidan flip-flop i vremenski dijagram

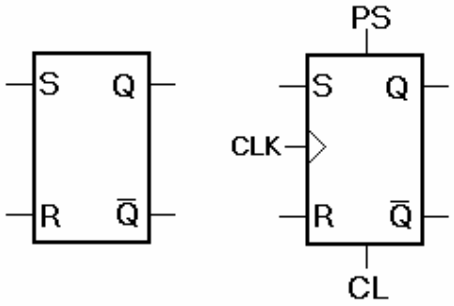
Pošto kod sekvencijalnih mreža vreme preko **CLK** sinhronizacionog signala karakrteriše ponašanje mreže moramo definisati stanje mreže:

- Q_t - sadašnje stanje (stanje pre pojave sinhronizacionog signala), i
- Q_{t+1} - buduće stanje (stanje posle sinhronizacionog signala).

☺ ☹ ☺

11.4.2 R/S (RESET-SET) FLIP-FLOP

Simbolička šema asinhronog i sinhronog **R/S** flip-flopa je data na slici 11.8.



Slika 11.8: simbolička šema asinhronog i sinhronog **R/S** flip-flopa

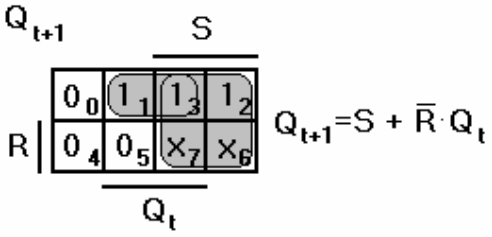
Funkcionalna tabela data na slici 111.9. opisuje rad **R/S** flip-flopa. Kod **R/S** flip-flopa imamo dva nedozvoljena stanja.

		Sadašnje stanje	Buduće stanje	Opis
R	S	Qt	Qt+1	
0	0	0	0	HOLD
0	0	1	1	HOLD
0	1	0	1	SET
0	1	1	1	SET
1	0	0	0	RESET
1	0	1	0	RESET
1	1	0	X	nedozvoljeno
1	1	1	X	nedozvoljeno

Slika 11.9: funkcionalna tabela **R/S** flip-flopa

Rad **R/S** flip-flopa možemo i analitički opisati ako minimiziramo funkciju (slika 11.10):

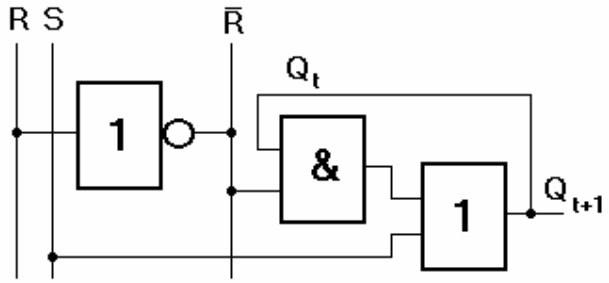
$$Q_{t+1} = S + \overline{R} \cdot Q_t \qquad (11.1)$$



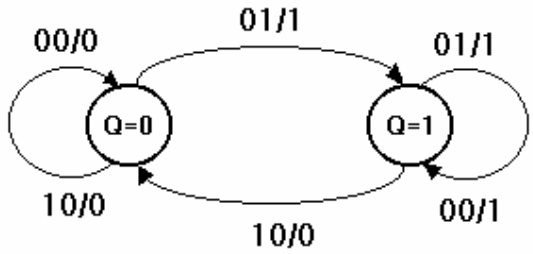
Slika 11.10: minimizacija funkcije **R/S** flip-flopa pomoću Karnaugh-ove table

Q_{t+1} i Q_t oznake koristimo za označavanje istog izlaza flip-flopa, ali indeks **t +1** je vremenski trenutak posle promene stanja (buduće stanje), dok indeks **t** se odnosi na stanje memorije pre promene stanja (sadašnje stanje). **R/S** flip-flop ima dva ulaza (**R** i **S**), međutim na ponašanje kola utiče još i sadašnje stanje, tj. Stanje u kom je trenutno mreža, pa tablica istine praktično ima **3** ulaza (R, S i Q_t), odnosno ukupno $n = 2^3 = 8$ kombinacija (slika 11.9).

Na osnovu jednačine **R/S** flip-flopa možemo nacrtati simboličku šemu asinhronog **R/S** flip-flopa (slika 11.11) i dati graf kola (slika 11.12).



Slika 11.11: simbolička šema asinhronog **R/S** flip-flopa



Slika 11.12: dijagram (graf) stanja **R/S** flip-flopa (**RS/Q**)

Tabelu pobude **R/S** flip-flopa možemo ispunniti na osnnovu podataka iz funkcionalne tabele (slika 11.13).

Q_t	Q_{t+1}	R	S
0	0	X	0
0	1	0	1
1	0	1	0
1	1	0	X

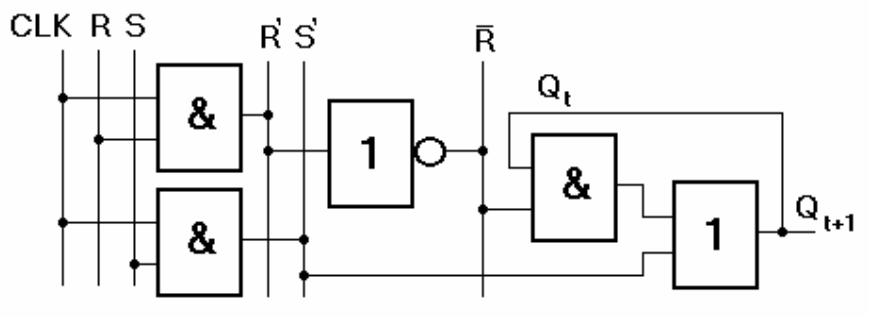
Slika 11.13: tabela pobude **R/S** flip-flopa

Zakoni funkcionisanja **R/S** flip-flopa se mogu zadati na pet načina:

- funkcionalnom tabelom,
- analitičkim jednačinama,
- dijagramom stanja,
- simboličkom šemom i
- tabelom pobude.

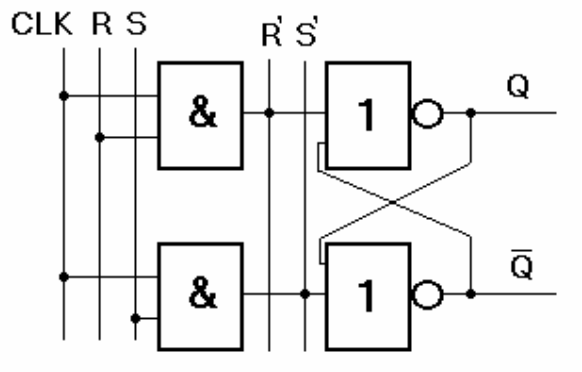
Praktično svi ovi opisi ponašanja su isti i iz jednog funkcionisanja možemo dobiti bilo koji drugi.

Jednostavno možemo dobiti od asinronog **R/S** flip-flopa (slika 11.11) sinhroni **R/S** flip-flop, ako dodajemo **CLK** signal za sinhronizaciju i dva **I** kola (slika 11.14). Do promene stanja kola će doći samo tada, kada se pojavi sinhronizacioni signal (**CLK = 1**).



Slika 11.14: sinhroni **R/S** flip-flop

Na slici 11.15 je dato tehničko rešenje sinhronog R/S flip-flopa sa NILI elementima, i sa **I** elemetima za sinhronizaciju.

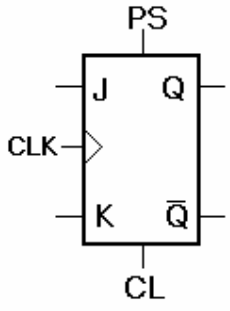


Slika 11.15: sinhroni R/S flip-flop sa NILI elementima

☺ ☹ ☺

11.4.3 J/K FLIP-FLOP

Simbolička šema sinhronog **J/K** flip-flopa je data na slici 11.16. Mada teoretski možemo napraviti i asinhroni J/K flip-flop, međutom to ne može da funkcioniše.



Slika 11.16: simbolička šema sinhronog **J/K** flip-flopa

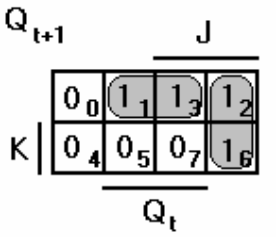
Funkcionalna tabela koja opisuje rad **J/K** flip-flopa, data je na slici 11.17. Dok kod **R/S** flip-flopa imamo dva nedozvoljena stanja (11.9), ovde **JK=1** je dozvoljeno, ako je **Q=1**, i onda će izlaz biti **Q=0** i obrnuto..

		Sadašnje stanje	Buduće stanje	opis
K	J	Qt	Qt+1	
0	0	0	0	HOLD
0	0	1	1	HOLD
0	1	0	1	SET
0	1	1	1	SET
1	0	0	0	RESET
1	0	1	0	RESET
1	1	0	1	TOGGLE
1	1	1	0	TOGGLE

Slika 11.17: funkcionalna tabela **J/K** flip-flopa

Rad **J/K** flip-flopa možemo i analitički opisati ako minimiziramo funkciju (slika 11.18):

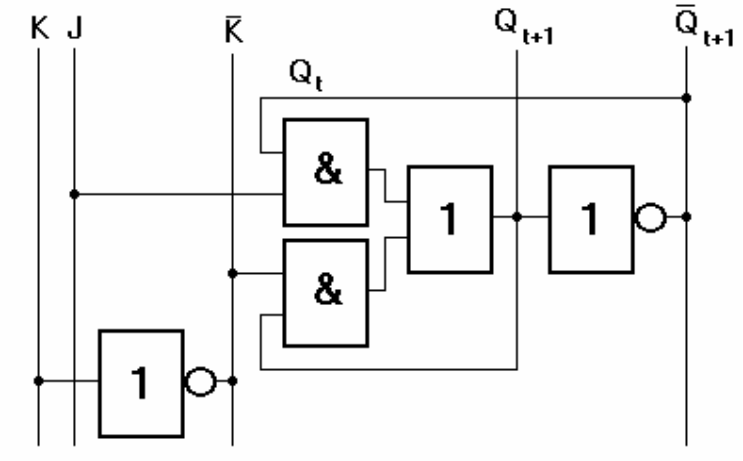
$$Q_{t+1} = J \cdot \overline{Q_t} + \overline{K} \cdot Q_t \qquad (11.2)$$



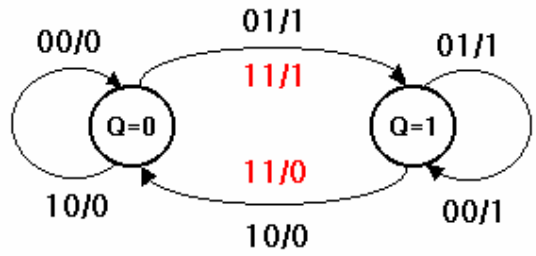
Slika 11.18: minimizacija funkcije **J/K** flip-flopa pomoću Karnaugh-ove table

Q_{t+1} i Q_t oznake koristimo za označavanje istog izlaza flip-flopa, ali indeks **t +1** je vremenski trenutak posle promene stanja (buduće stanje), dok indeks **t** se odnosi na stanje memorije pre promene stanja (sadašnje stanje). **J/K** flip-flop ima dva ulaza (**K** i **J**), međutim na ponašanje kola utiče još i sadašnje stanje, u kom stanju je mreža, pa tablica istine praktično ima **3** ulaza (K, J i Q_t), odnosno ukupno $n = 2^3 = 8$ kombinacija (slika 11.17).

Na osnovu jednačine **J/K** flip-flopa možemo nacrtati simboličku šemu asinhronog **J/K** flip-flopa (slika 11.19) i dati graf kola (slika 11.20).



Slika 11.19: simbolička šema asinhronog **J/K** flip-flopa



Slika 11.20: dijagram (graf) stanja **J/K** flip-flopa (**KJ/Q**)

Kombinacije **11/1** i **11/0** (**KJ/Q**) praktično dovode kolo u režim oscilacija, gde frekvencija oscilacija zavisi od tehničkih parametara kola. Drugačije rečeno, u stanju J=K=1 ne postoji stabilno stanje. Zbog ove činjenice **J/K** flip-flop će ispravno raditi samo sa sinhronizacionim signalom (**CLK**).

Tabelu pobude **J/K** flip-flopa možemo ispuniti na osnovu podataka iz funkcionalne tabele (slika 11.21).

Q_t	Q_{t+1}	K	J
0	0	X	0
0	1	X	1
1	0	1	X
1	1	0	X

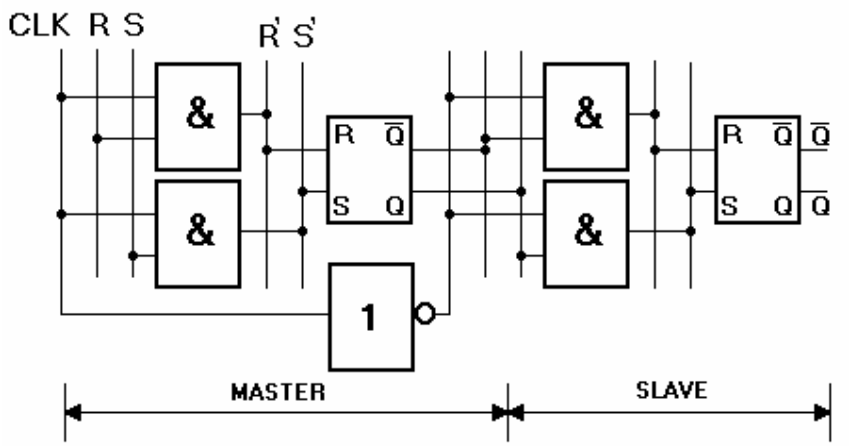
Slika 11.21: tabela pobude **J/K** flip-flopa

Zakoni funkcionisanja **J/K** flip-flopa su zadati na pet načina:

- funkcionalnom tabelom,
- analitičkom jednačinom,
- dijagramom stanja,
- simboličkom šemom i
- tabelom pobude.

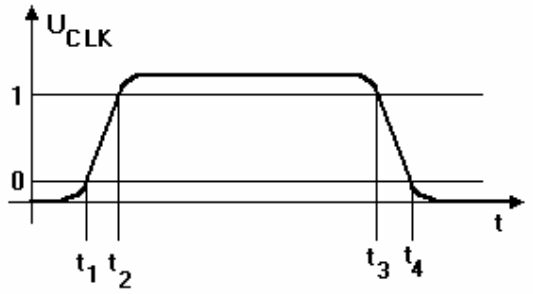
Praktično svi ovi opisi ponašanja su isti i iz jednog funkcionisanja možemo dobiti bilo koji drugi.

Praktična realizacija **J/K** flip-flopa je u obliku **MS** (MASTER/SLAVE). Rešenje je sa ivičnim okidanjem. Simbolička šema kola je data na slici 11.22.



Slika 11.22: **MS J/K** flip-flop

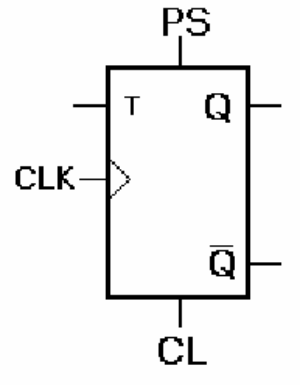
Na slici 11.23 možemo pratiti rad **MS** flip-flopa. Dok je **CLK** signal (U_{CLK}) nula (**0**) nije dozvoljena nikakva promena kod MASTER stepena flip-flopa. Prelaz sinhronizacionog sigala $CLK = 0 \rightarrow 1$ (pozitivno okidanje) informacija preko R i S ulaza će biti upisana u MASTER stepen, a istovremeno zbog $\overline{CLK}$ signala SLAVE stepen je zatvoren za bilo kakvu promenu. Sve dok je CLK=1, sadržaj MASTER stepena se menja zavisno od ulaza. Kod promene sinhronizacionog signala $CLK = 1 \rightarrow 0$ (negativno okidanje) ulazi MASTER stepena su zatvoreni, a istovremeno ulazi (R i S) u SLAVE stepenu su otvoreni, pa informacija iz MASTER dela je prepisana u SLAVE stepen.



Slika 11.23: vremenski dijagram **CLK** signala kod **MS J/K** flip-flopa

11.4.4 T (TOGGLE) FLIP-FLOP

Šematski prikaz **T** flip-flopa je dat na slici 11.24.



Slika 11.24: šematski prikaz **T** flip-flopa

Funkcionalna tabela data na slici 11.25opisuje rad **T** flip-flopa.

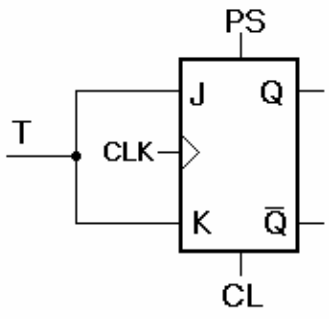
Ulaz	Sadašnje stanje	Buduće stanje	Opis
T	Q_t	Q_{t+1}	
0	0	0	HOLD
1	0	1	TOGGLE
0	1	1	HOLD
1	1	0	TOGGLE

Slika 11.25: funkcionalna tabela **T** flip-flopa

Rad **T** flip-flopa možemo i analitički opisati:

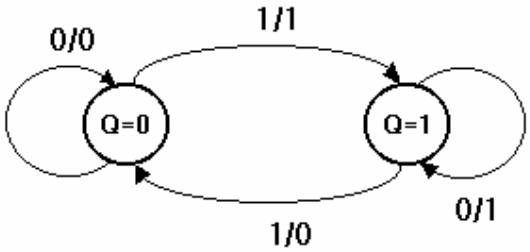
$$Q_{t+1} = T \cdot \overline{Q_t} + \overline{T} \cdot Q_t \qquad (11.3)$$

Iz **J/K** flip-flopa možemo napraviti **T** flip-flop, tako, što spojimo **J** i **K** ulaze. U ovom slučaju moguće kombinacije **J/K** flip-flopa (slika 11.17) su samo one, za koje je **J=K**. Simbolička šema **T** flip-flopa je data na slici 11.26.



Slika 11.26: pretvarnje **J/K** flip-flopa u **T** flip-flop

Graf kola (dijagram stanja) je na slici 11.27.



Slika 11.27: dijagram stanja **T** flip-flopa

Tabelu pobude **T** flip-flopa možemo ispuniti iz funkcionalne tabele (slika 11.28).

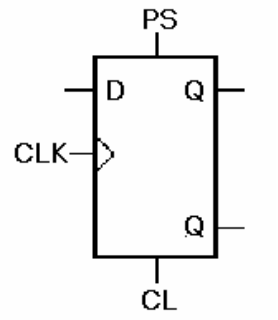
Q_t	Q_{t+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Slika 11.28: tabela pobude T flip-flopa

☺ ☹ ☺

11.4.5 D (DELAY) FLIP-FLOP

Šematski prikaz **D** flip-flopa je dat na slici 11.29.



Slika 11.29: šematski prikaz **D** flip-flopa

Funkcionalna tabela koja opisuje rad **D** flip-flopa, je data na slici 11.30.

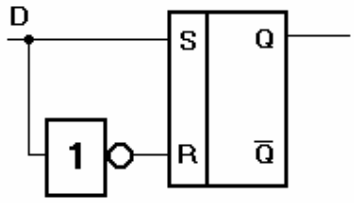
Ulaz	Sadašnje stanje	Buduće stanje	Opis
D	Q_t	Q_{t+1}	
0	0	0	HOLD
1	0	1	TOGGLE
0	1	0	TOGGLE
1	1	1	HOLD

Slika 11.30: funkcionalna tabela **D** flip-flopa

Rad **D** flip-flopa možemo i analitički opisati:

$$Q_{t+1} = D \qquad (11.4)$$

Iz **R/S** flip-flopa možemo napraviti **D** flip-flop, tako, što spojimo **R** i **S** ulaze sa invertorom. U ovom slučaju moguće kombinacije **R/S** flip-flopa (slika 11.9) su samo one, za koje su $J = \overline{K}$ i $\overline{J} = K$. Simbolička šema **D** flip-flopa je data na slici 11.31.



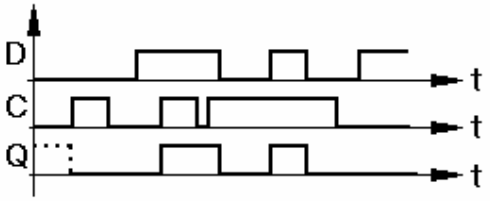
Slika 11.31: pretvarnje **R/S** flip-flopa u **D** flip-flop

Tabelu pobude **D** flip-flopa možemo ispuniti iz funkcionalne tabele (slika 11.32).

Q_{t+1}	D
0	0
1	1

Slika 11.32: tabela pobude **D** flip-flopa

D flip-flop nije memorija, u određenim vremenskim intervalima prihvata (ili ne prihvata) ulazni podatak i drži na izlazu do sledeće promene. Rad kola možemo analizirati na vremenskom dijagramu (slika 11.33). Dok je sinhronizacioni signal **C** nula (**0**), promene na ulazu **D** nemaju nikakav uticaj, a kada je jedinica (**1**) svaka promena ulaza će se pojaviti na izlazu **Q**.



Slika 11.33: vremenski dijagram rada **D** flip-flopa

11.5 TEHNIČKI PARAMETRI SINHRONIH MREŽA

11.5.1 UVOD

Sinhronu logičku mrežu možemo opisati sa:

- skupom stanja,
- skupom ulaza,
- uslovima prelaza stanja i
- ponašanjem u prelaznom režimu rada.

11.5.2 NAČINI ZADAVANJA SINHRONIH MREŽA

Sinhrone mreže se mogu zadati:

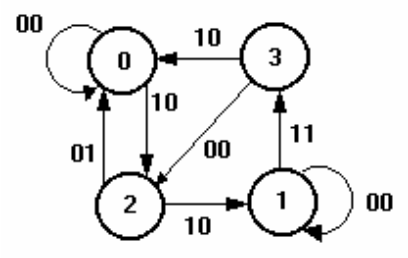
- neformalno (opisno) i
- formalno.

Formalno se zadaju sinhrone mreže:

- algebarskim jednačinama, koje opisuju prelaz stanja automata, na primer:

$$A_{t+1} = (\overline{A} \cdot B + A \cdot \overline{B} + B \cdot C) \cdot \overline{x} + A \cdot C \cdot (x + y)$$
$$B_{t+1} = (\overline{B} \cdot C + A \cdot B \cdot C + B \cdot \overline{C}) \cdot \overline{y} + A \cdot C \cdot x$$
$$C_{t+1} = \overline{A} \cdot x + \overline{B} \cdot y + B \cdot \overline{C}$$

- grafom (dijagramom) prelaza stanja:



Slika 11.34: graf prelaza stanja mreže

- tabelom prelaza stanja:

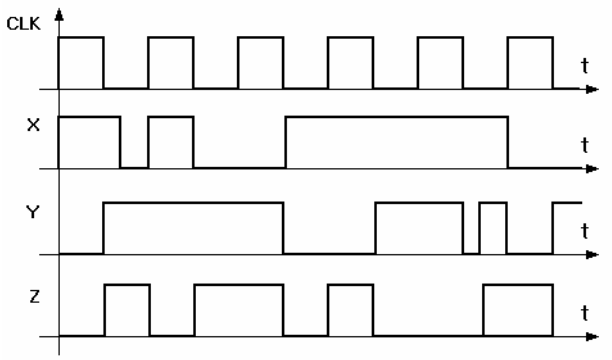
Sadašnje stanje	Ulaz x	Buduće stanje	Izlaz y
0	0	0	0
0	1	2	0
1	0	1	0
1	1	3	1
2	0	0	1
2	1	1	0
3	0	2	0
3	1	0	0

Slika 11.35: tabela prelaza stanja mreže

- sekvencom stanja:

0,0,0,1,1,2,1,3,4,5,6,5,7,6,4,8,9,3,5,7,8,9,9

- vremenskim dijagramima pojedinih signala (ulaza, izlaza itd.)



Slika 11.36: vremenski dijagram pojedinih signala

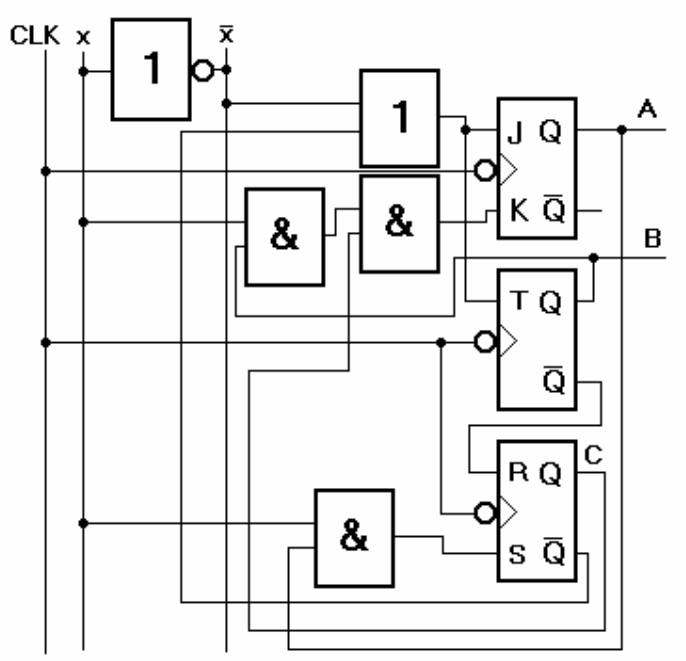
☺ ☹ ☺

11.5.3 ANALIZA SINHRONIH MREŽA

Analiza mreže se zasniva na postojećoj logičkoj šemi. Analiza mreže podrazumeva postupak određivanja ponašanja mreže čija je logička šema data.

PRIMER 11.1:

Na slici 11.37 je data simbolička šema sinhrone mreže.



Slika 11.37: simbolička šema sinhrone mreže

- pošto šema sadrži 3 flip-flopa, imamo i tri promenljivih stanja, a to su **A**, **B** i **C**,
- broj ulaza je jedan, i označavamo ga sa **x**,
- imamo dva izlaza, **A** i **B**,
- analitički izraz koji opisuje rad automata je:

$$A_{t+1} = J_A \cdot \overline{A} + \overline{K}_A \cdot A \qquad J_A = \overline{C} + \overline{x} \qquad K_A = \overline{x} \cdot B \cdot C$$

$$\begin{aligned} A_{t+1} &= \overline{C} \cdot \overline{A} + \overline{x} \cdot \overline{A} + (x \cdot B \cdot C) \cdot A = \\ &= \overline{A} \cdot \overline{C} + \overline{A} \cdot \overline{x} + A \cdot \overline{x} + A \cdot \overline{B} + A \cdot \overline{C} = \\ &= \overline{C} \cdot (A + \overline{A}) + \overline{x} \cdot (A + \overline{A}) + A \cdot \overline{B} = \overline{C} + \overline{x} + A \cdot \overline{B} \end{aligned}$$

$$B_{t+1} = T_B \cdot \overline{B} + \overline{T_B} \cdot B \qquad T_B = \overline{x} + \overline{C}$$

$$B_{t+1} = \overline{x} \cdot \overline{B} + \overline{C} \cdot \overline{B} + \overline{\overline{x} + \overline{C}} \cdot B = \overline{x} \cdot \overline{B} + \overline{C} \cdot \overline{B} + x \cdot C \cdot B$$

$$C_{t+1} = S_C + \overline{R_C} \cdot C \qquad S_C = x \cdot A \qquad R_C = \overline{B}$$

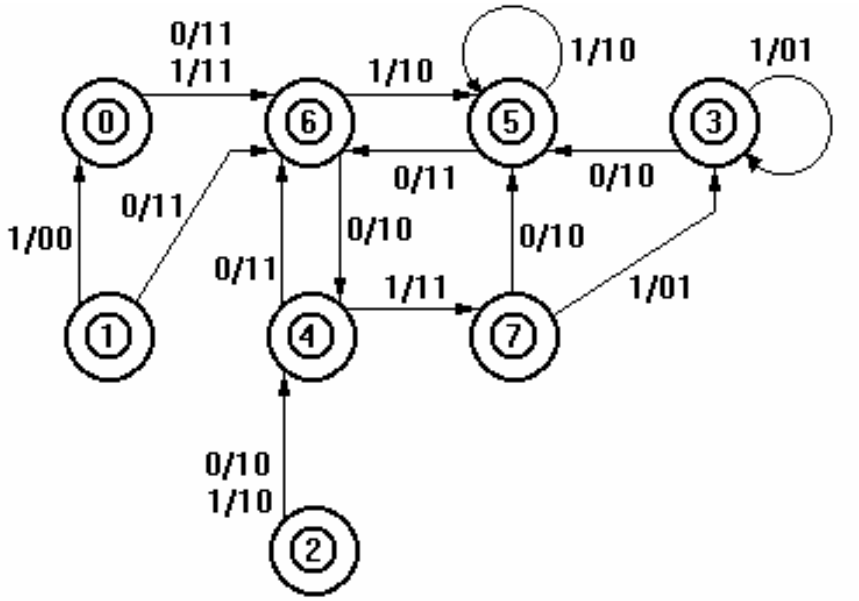
$$C_{t+1} = x \cdot A + B \cdot C$$

- Sada formiramo tablicu prelaza (slika 11.38):

	A	B	C	x	A	B	C
0	0	0	0	0	1	1	0
	0	0	0	1	1	1	0
1	0	0	1	0	1	1	0
	0	0	1	1	0	0	0
2	0	1	0	0	1	0	0
	0	1	0	1	1	0	0
3	0	1	1	0	1	0	1
	0	1	1	1	0	1	1
4	1	0	0	0	1	1	0
	1	0	0	1	1	1	1
5	1	0	1	0	1	1	0
	1	0	1	1	1	0	1
6	1	1	0	0	1	0	0
	1	1	0	1	1	0	1
7	1	1	1	0	1	0	1
	1	1	1	1	0	1	1

Slika 11.38: graf stanja mreže

- iz tabele prelaza formiramo graf stanja (slika 11.39):



Slika 11.39: graf stanja mreže

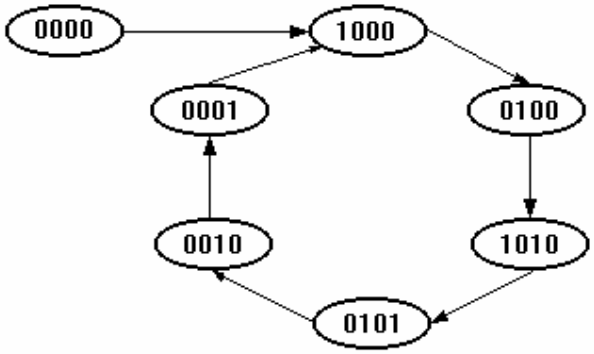
☺ ☹ ☺

11.5.4 SINTEZA SINHRONIH MREŽA

Kod sinteze mreže treba projekovati neku sinhronu mrežu.

PRIMER 11.2:

Dat je dijgram stanja neke mreže (slika 11.40).



Slika 11.40: dijagram stanja mreže

Izvršiti sintezu mreže uz primenu **J/K** plip-flopova. Pošto **J/K** flip-flopovima želimo realizovati mrežu, treba poznavati tabelu pobude te memorijske ćelije (slika 11.21).

Q_t	Q_{t+1}	K	J
0	0	X	0
0	1	X	1
1	0	1	X
1	1	0	X

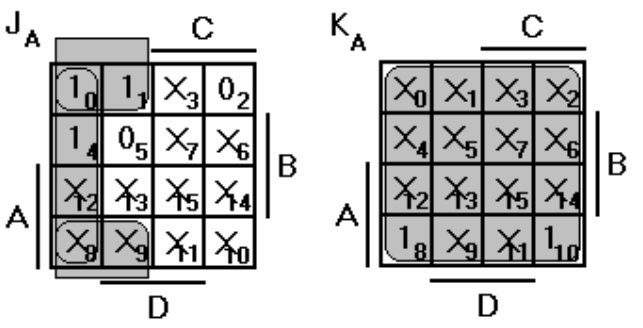
Slika 11.41: tabela pobude **J/K** flip-flopa

U prvom koraku treba ispuniti tabelu prelaza mreže (slika 11.42), podaci su uzeti sa dijagrama prelaza stanja i na snovu tabele pobude.

ABCD	ABCD	J	K	J	K	J	K	J	K
0000	1000	1	X	0	X	0	X	0	X
1000	0100	X	1	1	X	0	X	0	X
0100	1010	1	X	X	1	1	X	0	X
1010	0101	X	1	1	X	X	1	1	X
0101	0010	0	X	X	1	1	X	X	1
0010	0001	0	X	0	X	X	1	1	X
0001	1000	1	X	0	X	0	X	X	1

Slika 11.42: tabela prelaza stanja mreže

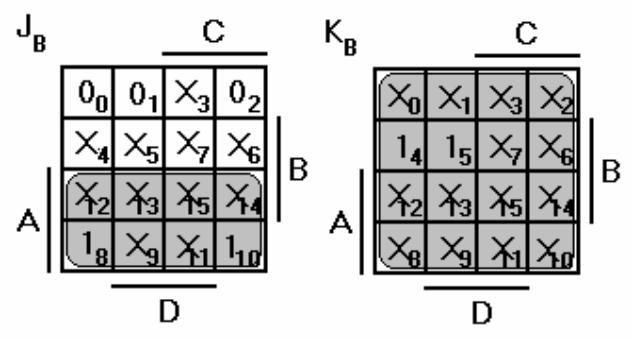
Za svaku memoriju, i kod svake memorije posebno za J i K ulaze treba odrediti analitički izraz. Za ovaj postupak koristimo Karnaugh-ovu kartu minimizacije (slike od 11.43 do .11.46).



Slika 11.43: minimizacija funkcija J i K ulaza prve memorije

Analitički izrazi **A** J/K flip-flopa su:

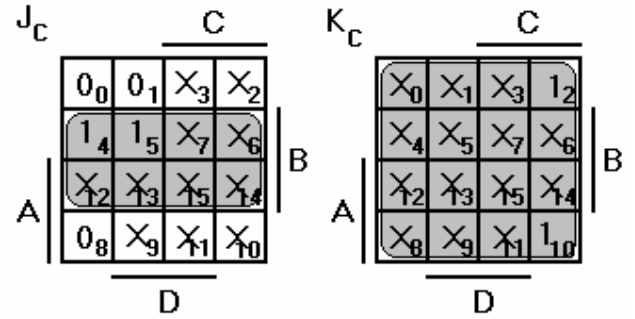
$$J_A = \overline{C} \cdot \overline{D} + \overline{B} \cdot \overline{C} \qquad K_A = 1$$



Slika 11.44: minimizacija funkcija J i K ulaza druge memorije

Analitički izrazi **B J/K** flip-flopa su:

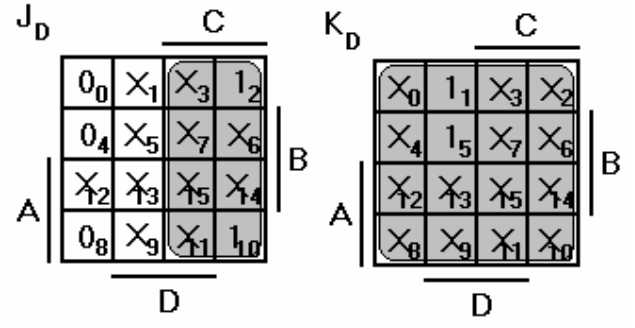
$$J_B = A \qquad K_B = 1$$



Slika 11.45: minimizacija funkcija J i K ulaza treće memorije

Analitički izrazi **C J/K** flip-flopa su:

$$J_C = B \qquad K_C = 1$$

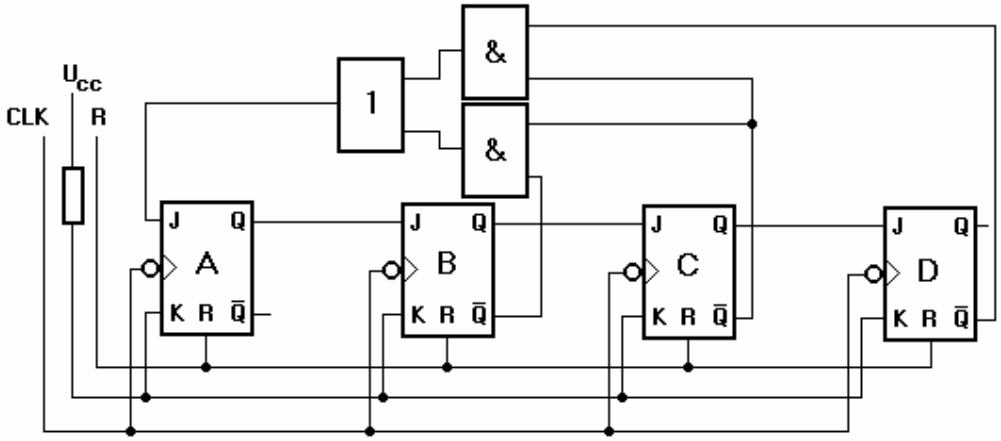


Slika 11.46: minimizacija funkcija J i K ulaza četvrte memorije

Analitički izrazi **D J/K** flip-flopa su:

$$J_D = C \qquad K_D = 1$$

Rešenje zadatka je dato na slici 11.47.



Slika 11.47: rešenje zadatka sa **J/K** plip-flopovima

☺ ☹ ☺

11.5 REGISTRI

11.5.1 UVOD

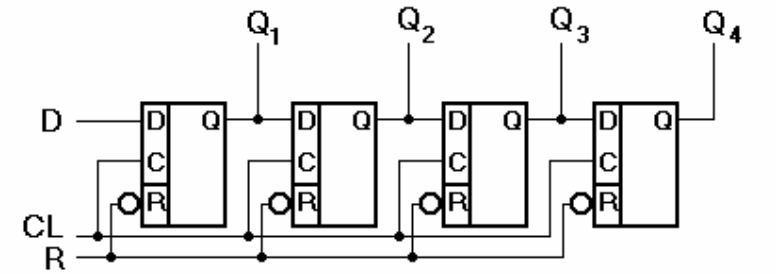
Pomoću registara možemo privremeno memorisati digitalnu informaciju. Registar se sastoji od nekoliko memorijskih ćelija, ove memorijske ćelije mogu biti **R/S**, **J/K**, **T** i **D** memorije (poglavlje 11.4), odnosno kombinacije ovih. Kad od nekoliko elemenata realizujemo višebitni registar, postoji uslov, i za paralelni rad, a to je zajednički taktni impuls.

Registar može da bude:
stacionarni i
pomerački.

Pomerački registar može da pomeri podatak:
levo,
desno i
kombinovano levo/desno.

PRIMER 11.3:

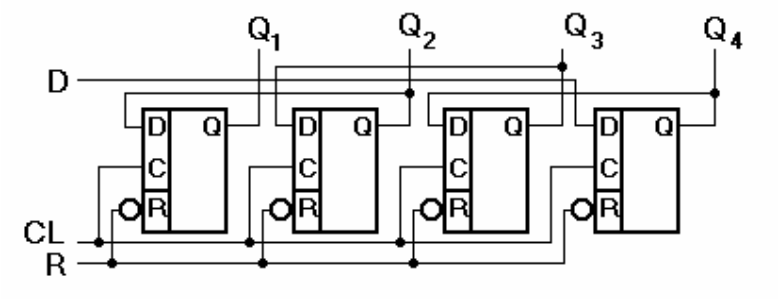
Na slici 11.48 je data šema za pomerački registar, koji će pomeriti upisani podatak nadesno.



Slika 11.48: pomerački registar nadesno

PRIMER 11.4:

Na slici 11.49 je data šema za pomerački registar, koji će pomeriti upisani podatak nalevo.



Slika 11.49: pomerački registar nalevo

11.6 BROJAČI

11.6.1 UVOD

Brojač ima zadatak da registruje broj događaja.

Na više načina možemo grupisati brojače, jedan je od njih po sinhronizaciji:

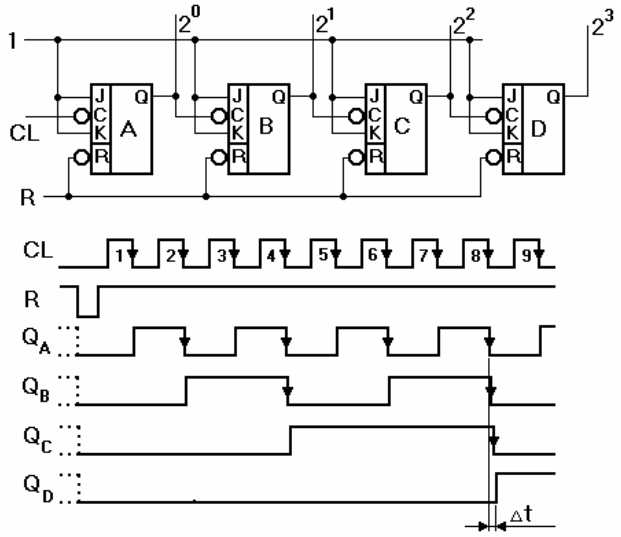
- asinhroni i
- sinhroni brojač.

Po kodnom sistemu razlikujemo brojače:
binarni,
Gray-ov,
BCD, itd.

Prema smeru brojanja razlikujemo brojače koje rade:
napred (inkrementiranje),
nazad (dekrementiranje) i
kombinovane (napred/nazad).

PRIMER 11.5:

Na slici 11.50 je dat asinhroni brojač i njegov vremenski dijagram. Očigledno je, da će kod asinhronog brojača izlaz memorijske ćelije odrediti stanje sledeće memorijske ćelije, pa zbog toga kašnjenja pojedinih kola posle određenog broja memorijskih elemenata proizvode kašnjenje Δt, pa kolo na izlazima menja stanje asinhrono.



Slika 11.50: asinhroni brojač

12. LITERATURA

1. Grunder Darko: Uvod u mikroprocesore, Tehnička knjiga, Zagreb.
2. Tešić Spasoje, Vasiljević Dragan: Osnovi elektronike, Gros knjiga, Beograd, 1995.
3. Lazić Borivoj, Urošević Zoran: Zbirka rešenih zadataka iz logičkog projektovanja digitalnih sistema, Nauka, Beograd, 1991.
4. Tešić Spasoje, Vasiljević Dragan: Zbirka zadataka iz digitalne elektronike, Naučna knjiga, Beograd, 1992.
5. Perišić Branko: Osnovi računarstva – Metodička zbirka zadataka I, , Stylos, Novi Sad, 1997.
6. Dirner Aleksandar, Vidaković Gavro: Osnovi logičko – prekidačkih kola, VTŠ, Subotica, 1981.

SADRŽAJ