**Logičko projektovanje (akreditacija 2013)**

***spisak tema***

**I Uvod (2)**

**1. Signali (funkcije) i njihovi matematički modeli**

**2. Logičke funkcije**

Neka je  binarna relacija iz skupa X u skup Y. Ako za svaki element x X postoji jedinstveni element y iz Y tako da vazi xy tada je funkcijaiz X u Y.

Funkcijom je pravilo kojim se povezuje svaki element x izskupa X sa jedinstvenim elementom y Izskupa Y i zapisuje se y=(x). Simbolicki predstavljeno: X →Y.

Skup X je domen funkcije, a skup Y kodomen funkcije.

Logicke funkcije mogu da budu jednoizlazne i viseizlazne.

Viseizlazna funkcija predstavlja sistem funkcija :

 = (0,…..,q-1), gde je q>1.

Kod nekih prakticnih realizacija, viseizlazna funkcija moze se zameniti ekvivalentnom jednoizlaznom.

Funkciju f moguce je potpuno predstaviti vektorom:

F=[f(0), f(1), f(2), f(3), f(4), f(5), f(6), f(7), f(8)]T

**3. Prekidačka teorija i logičko projektovanje / Bulova algebra**

Bool-ova algebra su algebarske strukture koje ujedinjuju osnovne karakteristike koje su zajednicke zalogicke operacije OR,AND,NOT.

DvoelementarnaBulova algebra:

Struktura B, Ʌ, V,-, gde je B={0,1} I operacije Ʌ, V, -, predstavljaju logicke operacije AND I OR I negaciju (komplement), predstavlja dvoelementarnu Bulovu algebru.

1 *a ∧ b* = *b ∧ a*, *a ∨ b* = *b ∨ a*, komutativnost,  
2 (*a ∧ b*) *∧ c* = *a ∧*(*b ∧ c*), (*a ∨ b*) *∨ c* = *a ∨*(*b ∨ c*), asocijativnost,  
3 *a∧*(*b∨c*) = (*a∧b*)*∨*(*a∧c*), *a∨*(*b∧c*) = (*a∨b*)*∧*(*a∨c*), distributivnost,  
4 *a ∧ a* = *a*, *a ∨ a* =, idempotentnost,  
5 *a* = *a*, involucija,  
6 (*a ∧ b*) = *a ∨ b*, (*a ∨ b*) = *a ∧ b*, de Morganovi zakoni,  
7 *a ∧ a* = 0, *a ∨ a* = 1, *a ∧* 1 = *a*, *a ∨* 0 = *a*, *a ∧* 0 = 0, *a ∨* 1 = 1, 1 = 0,  
0 = 1

**4. Booleove reprezentacije logičkih funkcija**

Diskretne funkcije, posto imaju konacne domene (skup vrednosti po x) se obicno definisu tabelama kojena levoj strain sadrze elemente domena, a na desnoj sadrze vrednosti funkcije. U slucaju prekidackih funkcija, ove tabele su tabele istinitosti, a vrednosti funkcije su predstavljene kroz vektore istinitosti.

Velicina tabela eksponencijalno raste sa povecanjem broja promenljivih.

Smanjene oblike ovih funkcija mozemo dobiti koriscenjem specificnih svojstava ovih prekidackih funkcija.

Kubovi

Posto prekidacke funkcije mogu da uzmu 2 vrednosti, nije neophodno da se prikaze kompletna tabela istinitosti niti vektor istinitosti.Dovoljno je oznaciti mesta gde data funkcija uzima vrednosti 0 ili 1 i pretpostaviti da u ostalim tackama domena, definicija funkcije ima drugu vrednost 1 ili 0 respektivno.

Na ovaj nacin funkcija se predstavlja preko 0 polja ili 1 polja. U funkciji, pojavljivanje nekih kombinacija ulaznih promenljivih se retko ocekuje, pa vrednosti funkcije u ovim tackama nisu definisane. Ove tacke se oznacavaju kao “don’t care”.

Dijagrami i mape

Prekidacke funkcije sa malim brojem promenljivih (do 5 – 6 ) se obicno predstavljaju graficki razlicitim dijagramima ili mapama. Siroko rasprostranjene metode su Veitch dijagrami i Karnoove mape. Kod ovog nacina podaci su poredjani na razlicite nacine.Kod Veich dijagrama koristi se leksikografski redosled, a kod Karnoovih mapa grejov kod.

Ostali nacini predstavljanja:

- Hiperkubovi

- Suma proizvoda i proizvod suma(PDNF I PKNF)

Tabular representations of switching functions can be easily converted to their analytical representations, meaning that the function is presented as a formula written in terms of some basic expressions.

-Pozicionalna kub notacija

- Faktorizovani izrazi

**5. Definicija i osnovni pojmovi**

Funkcionalni razvoji su analiticka reprezentacija prekidackih funcija kada se ove funkcije uzimaju sa bulovom algebrom kao osnovnom algebarskom strukturom.

Osnovni pojmovi :

Shannon Expansion Rule

Reed-Muller Expansion Rules

Fast Algorithms for Calculation of RM-expressions

Negative Davio Expression

Fixed Polarity Reed-Muller Expressions

Algebraic Structures for Reed-Muller Expressions

Interpretation of Reed-Muller Expressions

Kronecker Expressions

Generalized bit-level expressions

Word-Level Expressions

Arithmetic expressions

Calculation of Arithmetic Spectrum

Applications of ARs

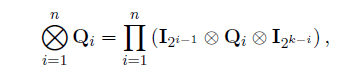
Walsh Expressions

**6. Brzi algoritmi za odredjivanje i optimizaciju fukncionalnih razvoja**

**Brzi algoritmi za izračunavanje RM-izraza**

Korišćenjem svojstava kronekerovog proizvoda može se pokazati da matrica koja predstavlja kronekerov proizvod dve matrice može da se dobije običnim proizvodom proređenih matrica.

Kronekerov proizvod matrica 2x2



gde je Ir identičkamatrica.

For n = 2, R(2) = R(1) ⊗ R(1) = C1\*C2

Each of the matrices C1 and C2 determines a step in the fast algorithm for calculation of the Reed-Muller coefficients. The non-zero elements in the i-th row of the matrix point out the values which should be added modulo 2 (EXOR) to calculate the i-th Reed-Muller coefficient. When in a row there is a single non-zero (1) element the value pointed is forwarded to the output.

**7. Algbraske strukture za proučavanje funkcionalnih razvoja**



Ova slika definise algebarske strukture koje se mogu koristiti za proucavanje

Reed Muller-ovih razvoja.Mozemo koristiti algebru, i to klasicnu Bool-ovu

algebru sa 2 elementa, ili Gibsovu algebru. U ovom slucaju, sabiranje se tretira

kao EXOR, mnozenje kao konvoluciono mnozenje. Cak, RM razvoji se mogu

proucavati i u vektorskim prostorima funkcije na konacnim diadickim grupama

u polja.

**8. Spektralna interpretacija funkcionalnih razvoja**

U drugim razvojima koeficijenti su elementi spektra koji se dobija primenom transformacionih matrica i dekompozicionih pravila.

**Razvoji na nivou reci** su uopstenje ili generalizacija razvoja na nivou bitova. U razvojima na nivou reci, koeficijenti su clanovi skupa racionalnih brojeva Q, ili uopstenije skupa kompleksnih brojeva.

**Aritmeticki razvoj**

Verovatno najprostiji primer razvoja na nivou reci je aritmeticki razvoj koji moze da se posmatra kao celobrojni deo RM pozitivne polarnisti. Aritmeticki razvoj se dobija zamenom Bool-ovih operacija odgovarajucim aritmeticim operacijama.

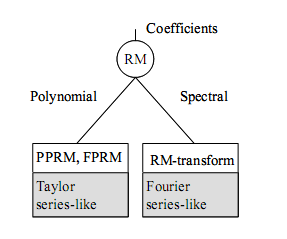
Znaci, aritmetiski razvoj ima istu formu kao i pozitivan RM razvoj samo sto se koriste aritmeticke operacije umesto bulovih.Vazna primena razvoja na nivou reci je predstavljanje viseizlaznih funkcija.

**Walsh-ovi razvoji**

Walshove funkcije su u originalu koriscene za resavanje ravnomernekonvergencije u odredjivanju realnih promenljivih u na intervalu 0 – 1. Diskretne Walshove funkcije mogu se posmatrati kao nezavisan set diskretnih funkcija ciji grafici imaju slicne oblike Walshovim funkcijama.Prikaz u vidu matrice je prost nacin za definisanje diskretnih Walsh funkcija.

**ALTERNATIVNO**

Ova slikapokazuje da RM razvojimogu da se prikazu kao polinomni razvoji, kao Tejlorovi redovi u klasicnoj matematickoj analizi, i u tom slucaju se RM koeficijenti mogu smatrati kaoanalogija Furijeovim redovima.Najcesce, ovi razvoji mogu da se izvedu iz diskretnih walshovih redova, koji jesu Furijeovi redovi.

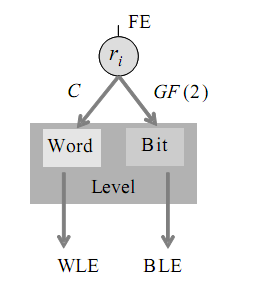


**9. Razvoji na nivou reči i primene**

Svi ranije pomenuti razvoji su na nivou bitova, zato sto su koeficijenti 0 i 1-ce. U slucaju SOP-a, koeficijenti su vrednost funkcije u datoj tacki, a u drugim razvojima koeficijenti su elementi spektra koji se dobija primenom transformacionih matrica i dekompozicionih pravila.

Razvoji na nivou reci su uopstenje ili generalizacija razvoja na nivou bitova.

U razvojima na nivou reci koeficijenti su gledani kao elementi polja raconalnih brojeva Q ili uopstenije elementi skupa kompleksnih bojeva C.

Pretpostavlja se da se logicke vrednosti za varijavle I vrednosti funcija predstavljaju kao realni ili komplekcni brojevi 0,1.

Ova slika ilustruje prosirivanje teorije funkcionalnih razvoja prekidackih funkcija

**Razvoji na nivou reci** su uopstenje ili generalizacija razvoja na nivou bitova. U razvojima na nivou reci, koeficijenti su clanovi skupa racionalnih brojeva Q, ili uopstenije skupa kompleksnih brojeva.

**Primena:**

1- Vazna primena razvoja na nivou reci je predstavljanje viseizlaznih funkcija. Upredstavljanju viseizlazne funkcije razvojima na nivou bitova, svaki izraz treba biti predstavljen odvojenim polinomom. Znacajna prednost AR-a je sto je moguce predstaviti viseizlazne funkcije jednim polinomom za celobrojne ekvivalentne funkcije.

2- Aritmeticki izrazi su korisni i kod dizajna aritmetickih kola.

3- Aritmeticki polinomi su se isto pokazali korisnim u testiranju logickih kola

4- Kao jos jedan koristan primer razvoja na nivou reci navode se Walshovi izrazi

**10. Sinteza logičkih mreža na osnovu funkcionalnih razvoja.**

U zavisnosti od funkcionalnog razvoja se koriste razliciti dijagrami odlucivanja. Dijagrami imaju svoje osobine i karakteristike kao sto su velicina dubina sirina. Sve ove osobine uticu na projetovanje logickih mreza. Logicke mreze mogu da se redukuju ako se redukuju njihovi dijagrami odlucivanja. Ako se redukuje broj neterminalnih cvorova u dijagramu smanjice se broj modula koji se koristi u logickog mrezi. Takodje i dubina utice na kasnjenje logicke mreze pa se tako moze i kasnjenje smanjiti. A dubina i sirina definisu povrsinu logicke mreze,tako da se redukcijom dubine i sirine moze uticati na povrsinu logicke mreze.

(ISTI ODGOVOR NA 15)

**III Dijagrami odlučivanja (4)**

**11. Binarni dijagrami odlučivanja**

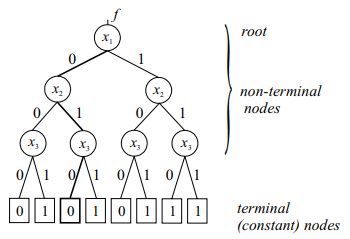
Dijagrami odlučivanja su strukture podataka koje se koriste za efikasno prikazivanje diskretnih funkcija kao što su prekidačke ili viševrednosne logičke funkcije.

Jedan način prikazivanja je stablo odlučivanja (binarno stablo odlučivanja) **DT (BDT).**

**\***((*Binary)*  Decision tree)

BDT predstavlja grafičku reprezentaciju Šenonovog razvoja.

Logička funkcija f(x1,x2,x3)=x1x2 v x3

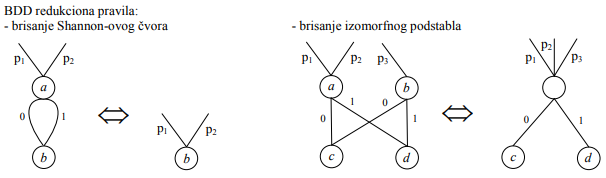
****

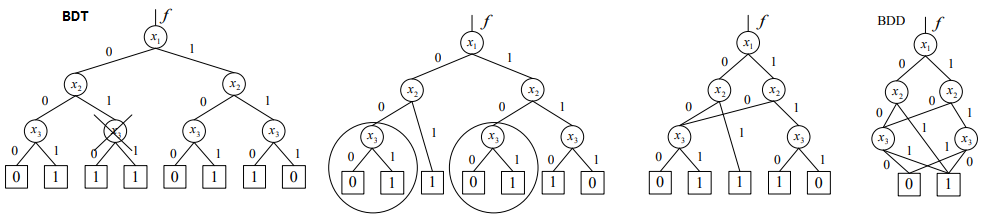
U korenu BDT je f, na dnu su terminalni čvorovi, koji imaju konstantnu vrednost, između su neterminalni čvorovi koji imaju dva kraka.

Uprošćavanjem BDT dolazimo do binarnih dijagrama odlučivanja, gde će se u zavisnoti od redosleda promenljivih dobiti različiti BDD za istu funkciju.

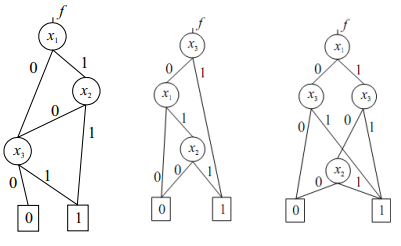
**BDD**(pravila uprošćavanja)

1. **Brisanje čvora – 2 grane se slikaju u isti čvor**
2. **Brisanje stabla – 2 čvora se slikaju u isti par terminalnih čvorova**





Različiti BDD za logičkufunkciju: f(x1,x2,x3)=x1x2 v x3:



**DEFINICIJA**

*Uređeni BDD je korenski usmereni graf koji se sastoji od dve vrste čvorova, terminalnih i neterminalnih. Neterminalni čvorovi imaju 2 potomka (0 i 1) i jednog roditelja. Svaki terminalni čvor ima konačnu vrednost 0 ili 1, koja nije ista za svaki terminalni čvor (vrednost u kvadratićima). Indeksi neterminalnih čvorova, krećući se od korenskog ka terminalnim čvorovima, su dati u rastućem redosledu (X1...Xn).*

**12. Dijagrami odlučivanja na nivou bitova i reci**

U zavisnoti od razvoja funkcije, mogu se koristiti dijagrami odlucivanja na osnovu bitova I na osnovu reči. Kod dijagrama na osnovu bitova, kao koeficijenti se javljaju samo 0 i 1 (FDD i SDD dijagrami).

Za BDD koriste se bdd pravila:

1. **Brisanje cvora - 2 grane se slikaju u isti čvor**
2. **Brisanje stable - 2 čvora se slikaju u isti par terminalnih čvorova**

Za FDD (pozitivan I negativan Reed Muller razvoj) se koriste ZBDD pravila (potiskivanje nula).

Na nivou reci koeficijenti nisu samo 0 i 1, vec i drugi brojevi. Kod ovih razvoja se koriste Aritmeticki I Wolshovi razvoji. Dijagrami su: ACDD, WDD dijagrami. Kod ovih dijagrama se koriste generalisana BDD pravila.

**13. Osnovne karaktetristike dijagrama odlučivanja i njihova optimizacija**

**Osnovne karakteristike dijagrama odlucivanja:**

1) Veličina dijagrama - definiše se kao broj neterminalnih čvorova zadijagrame na nivou bitova, ili kao suma neterminalnih i konstantnih čvorova zadijagrame na niovu reči

2) Dubina dijagrama - definiše se brojem nivoa

3) Širina dijagrama - maksimalni broj čvorova po nivou

4) Broj staza iz korenskog čvora koji vode u nenulte terminalne čvorove

Broj neterminalnih čvorova u dijagramu odgovara broju elementarnih modal u odgovarajućoj logičkoj mreži.Kada se vrši računanje preko dijagrama mora se primeniti neki obrazac za računanje za svaki neterminalni čvor. Zato je redukcija neterminalnih čvorova glavni cilj u optimizovanju dijagrama odlučivanja.Takođe kašnjenje kroz mrežu je proporcionalno dubini dijagrama odlučivanja koja zajedno sa širinom određuje područje koje mreža zauzima.

Grane u dijagramu definišu međuveze u mreži. Zato na mestima gde su određeni parametri mreže važni, bitno je poznavati karakteristike dijagrama.

**14. Spektralna interpretacija dijagrama odlučivanja**

Ako je datoj funkciji f dodeljeno stablo odlučivanja Šenonovim ekspanzionim pravilom, tada dobijamo Binarni Dijagram Odlučivanja (BDD) ili VišeTerminalni Binarni Dijagram Odlučivanja (MTBDD) zavismo od ranga funkcija koje su

predstavljene.

Međutim, ako prvo konvertujemo funkciju u spektar spektralnom transformacijom koja je određena dekompozicionim pravilima koja su primenjena u čvorovima stabla odlučivanja, i onda tako dobijenom spektru Sf dodelimo stablo odlučivanja, onda dobijamo "Spectral transform decision tree" (STDT). Kod STDT-a vrednosti konstantnih čvorova su spektralni koeficijenti a nazivi grana su definisani dekompozicionim pravilima tako da daju inverznu transformaciju za određivanje vrednosti funkcije iz spektralnih koeficijenata.

Zato kod STDT-a ako pratimo oznake na granama polazeći od konstantnih čvorova, čitanje funkcije vršimo računajući inverznu ekspanziju funkcije.

Ako zamenimo oznake grana onima koje koristimo kod Šenonovih čvorova, iz STDT možemo pročitati spektar funkcije f.Isti pristup koristimo u suprotnomsmeru za određivanje spektralnih transformacija BDT-ova i MTBDT-ova.

Glavna osobina uređenih BDD je da svaka prekidačka funkcija ima jedinstvenu reprezentaciju kao redukovani BDD, gde su suvišni čvorovi i grane uklonjeni.

Šenonov razvoj prekidačke funkcije:

f=if0xif1

Rekurzijom Šenonovog razvoja kao rezultat dobija se kompletna disjunktivna normalna forma (KDNF).

Rekurzijom Šenonovog razvojau grafičkoj reprezentaciji dobićemo BDT za datu funkciju f.

**15. Projektovanje logičkih kola preko dijagrama odlučivanja**

U zavisnosti od funkcionalnog razvoja se koriste razliciti dijagrami odlucivanja. Dijagrami imaju svoje osobine i karakteristike kao sto su velicina dubina sirina. Sve ove osobine uticu na projetovanje logickih mreza. Logicke mreze mogu da se redukuju ako se redukuju njihovi dijagrami odlucivanja. Ako se redukuje broj neterminalnih cvorova u dijagramu smanjice se broj modula koji se koristi u logickog mrezi. Takodje i dubina utice na kasnjenje logicke mreze pa se tako moze i kasnjenje smanjiti. A dubina i sirina definisu povrsinu logicke mreze,tako da se redukcijom dubine i sirine moze uticati na povrsinu logicke mreze.

**16. Izračunavanja preko dijagrama odlučivanja**

U knjizi imate na strani 117, primer 4.19 preko koga je opisano izračunavanja Walsh-ovog spektra preko MTBDD-a. Za ovo pitanje je dovoljno da se na primeru (ne mora da bude baš onaj u knjizi) objasni kako se vrši izračunavanje preko dijagrama odlučivanja. Imate i jedan zadatak na računskim vežbama na tu temu.

**IV Klasifikacija prekidačkih funkcija (4)**

**17. NPN, SD i LP-klasifikacija**

**NPN klasifikacija**

Klasifikaciona pravila:

1. Negacija ulaznih promenljivih

2. Permutacija ulaznih promenljivih

3. Negacija izlaza

Ne moramo uvekda koristimo sva pravila, pa tako dobijamo podsetove: N, P, NP, NPN.

Kazemo da dve funkcije f1 i f2 istog broja ulaznih promenljivih pripadaju istoj klasi ili su ekvivalentne ako moguda se svedu jedna na drugu klasifikacionim pravilima koje dozvoljava data klasifikacija.

**SD klasifikacija**

Klasifiacija koja je jaca od NPN se definise po uslovima samodualnih funkcija i prema tome zove SD klasifikacija prekidackih funkcija.

Funkcije pripadaju istoj SD klasi ako samodualizacione funkcije pripadaju istoj NPN klasi.

**LP klasifikacija**

NPN i SD klasifikacija su pravljene za AND-OR sintezu, sto znaci da je reprezentacija funkcija kroz AND-OR izraze.LP klasifikacija je pravljena za AND-EXOR kola.Kada se realizuje mreza AND - OR kolima, potrebno je vise mnozenja, dok kod realizacije AND-EXOR kolima treba manje mnozenja znaci da su LP efikasnije.

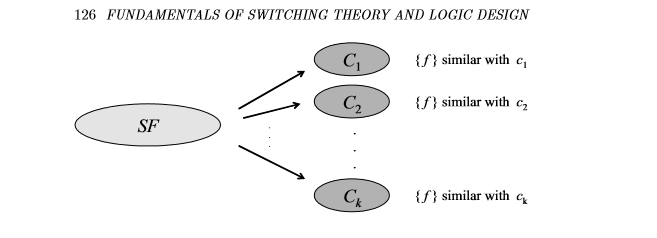
Kod ove klasifikacije, osim transformisanja promenljivih, mozemo da transformisemo i konstante.

**18. Primene klasifikacije u sintezi logičkih mreža**

Klasifikacija prekidackih funkcija je jedan od najvaznijih problema u prekidackoj teoriji i blisko je povezana sa njihovim realizacijama. Motiv- smanjenje broja razlicitih mreza u modularnoj sintezi logickih mreza.

Za n promenljivih postoji 2^2^n razlicitih funkcija, sto podrazumeva isti broj razlicitih mreza za njihovu realizaciju. Mnoge funkcije su u nekom smislu povezane.

Zadatak klasifikacije je u particiji skupa SF (skup svih prekidackih funkcija za dati broj promenljivih n) u klase Ci funkcija koje su medjusobno slicne u odnosu na neke odgovarajuce kriterijume klasifikacije. Funkcije koje pripadaju istoj klasi mogu se svesti jedne na druge primenom operacija izvrsenih u klasifikaciji(pravila klasifikacije).



Dve primene klasifikacije su:

1)Realizacija po prototipovima koji pretpostavlja dizajn slicnih kola za funkcije unutar iste klase;

2)Standardizacija metoda za testiranje logickih mreza.

**19. Univerzalni logički moduli**

Klasifikacija prekidackih funkcija ima kao jedan cilj redukciju broja razlicitih mreza za realizaciju funcija datog broja promenljivih. Buduci razvoj ove ideje vodi do univerzalnih logickih modula ULM definisanih kao logicke mreze koje mogu da realizuju bilo koju od 2^2^n logickih funkcija za n promenljivih. Podrazumeva se da su na ulazu ULM-ova dostupne konstante 1 i 0 , kao i menjanje polariteta. ULM mora da ima neke kontrolne ulaze kako bi mogli da odaberemo realizaciju odredjene funkcije.Cena univerzalnosti ovakvog modula jeste veci broj ulaza od broja ulaznih promenljivih funkcije. Na primer ULM sa 3 promenljive ima 5 ulaza od kojih su 2 kontrolna.

Jedna od mana ULM-a je ta sto za male prekidacke mreze moraju da se potrose citavi moduli ako bi se realizovala neka elementarna funkcija koja je mogla biti realizovana sa par najprostijih logickih kola.

Takodje koriscenje citavih modula moze da poveca broj nivoa koji se koriste, samim tim i kasnjenje kroz mrezu.

**20. Sinteza sa makro ćelijama**

Sinteza univerzalnim celijskim poljima moze da se posmatra kao sinteza sa eksplicitno zatatom bibliotekom celija. Razvijena su 2 glavna pristupa kod "poluproizvoljnog" projektovanja:

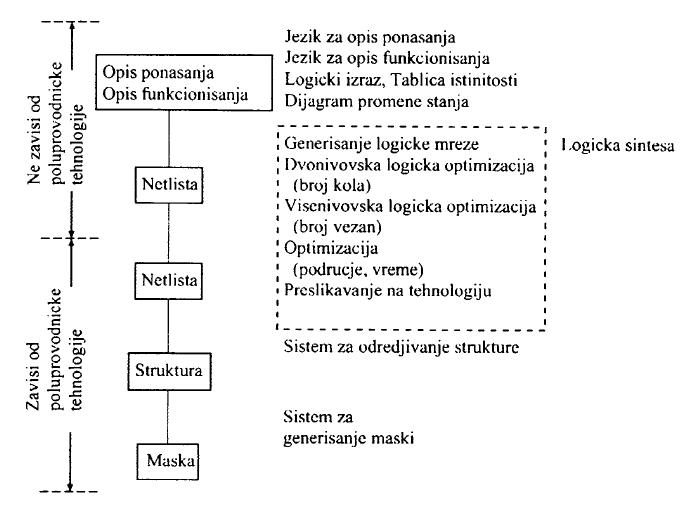
1. Celijski bazirani dizajn koji se moze definisati kao dizajn celijama i makrocelijama

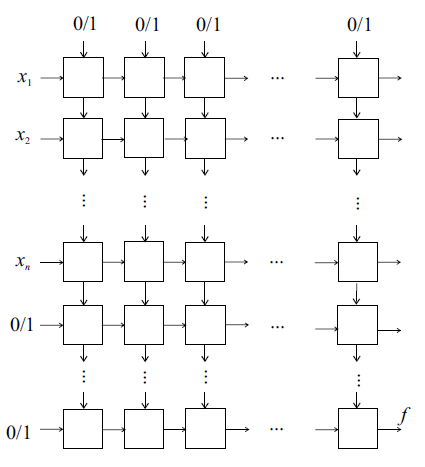
2. Dizajn baziran na poljima gde se razlikuju predifused i prewired dizajn

Kod dizajna makro celijama koristimo racunarske programe "generatore modula" koji prave makro celije (optimizovane do nekog novoa) logickih podmreza.Makro celije se dobijaju povezivanjem i distribuiranjem funkcionalnih celija automatski iz logickih izraza. Obicno se postavlja set restrikcija na nivou pocetnih logickih izraza. Ove restrikcije se uglavnom postuju na nivou celije i za potrebe performanse celije.Logicke celije koje ispunjavaju ovakve restrikcije tj. zahteve formiraju virtuelnu biblioteku.Procedura projektovanja se zatim sastoji od manipulacije mrezom sve dok se ne postignu zeljene performanse.

**21. CAD sistemi za realizaciju prekidačkih funkcija**

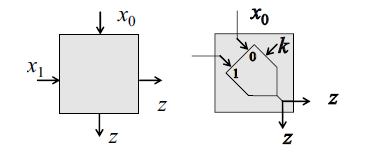
Projektovanje višenivovskih logičkih mreža je složen proces i u cilju njihove sinteze za kratko vreme koristi se automatsko projektovanje korišćenjem računara. Computer-aided design (CAD) sistem je sistem koji pruža podršku pri realizaciji, analizi i optimizaciji različitih dizajna. Međutim za razliku od minimizacije dvonivoskih mreža, ne postoje utvrđeni algoritami za minimizaciju višenivovskih mreža. Često korišćen sistem za proejktovanje je LSI sistem odnosno „silikon kompajler“ i obično se sastoji iz dva dela. Prvi deo je logička sinteza gde se funkcionalni opis tranformiše u logičku mrežu. Funkcionalni opis je zadat pomoću posebnih jezika za projektovanje kao što je VHDL, logičkim izrazima, tablicama istinitosti, tabelama prelaza itd. Logičke veze su opisane listom veua između logičkih elemenata koje se nazivaju netlistama. Prvi deo je nezavisan od poluprovodničke tehnologije a drugi deo je genersanje strukture kola i tranformisanje netlista u maske za LSI. Najpre se specifikacija opisana jezikom visokog nivoa prevodi u dvonivoske logičke mreže koja se kasnije uprošćava. Kada se zada i skup logičkih elemenata koji se koriste (biblioteka) vrši se preslikavanje na tehnologiju.





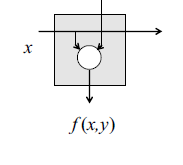
**22. Univerzalana ćelijska polja**

Mreze, koje se cesce nazivaju modularne mreze i koje se sastoje od identicnih modula sa prostim medjuvezama su cesta resenja.Takvo resenje se zove univerzalno celijsko polje i to je planarna mreza koja se sastoji od kola iz nekoliko razlicitih klasa koja se isporucuju u standardnom obliku gde su veze izmedju njih svedene samo na linkove izmedju susednih modula.



*Dvodimenzionalno jednoulazno univerzalno ćelijsko polje.*

*Struktura ćelije za realizaciju 2D mreže.*



Univerzalna celijska polja mozemo klasifikovati po razlicitim kriterijumima. Zavisno od broja medjuveza izmedju celija, polja se mogu podeliti na polja sa 1 vezom i sa vise medjuveza.Ako nema povratnih veza izmedju celija, onda su polja unilateralna, u suprotnom bilateralna. Iterativna celijska polja se sastoje od identicnih celija.

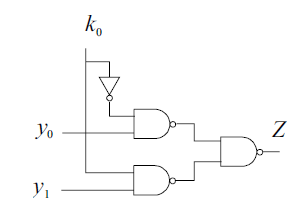
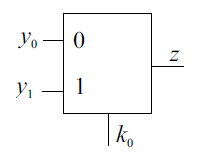
Postoje takodje dvodimenzionalna i trodimenzionalna polja.

Najcesce, polja realizuju DNF,KNF ili RM razvoj.

**23. NP-kompletni moduli, multiplekseri i demultiplekseri**

Za n promenljivih postoji različitih funkcija odnosno logičkih mreža i mogu se spram njihovih osobina rasporediti u klase pri čemu važi da su dve funkcije f1 i f2 od n promenljivih u istoj klasi ako mogu da se svedu jedna na drugu primenom dozvoljenih pravila. Ukoliko su ta pravila negacija (N) i permutacija (P) promenljivih onda govorimo o NP-klasifikaciji **a NP-kompletan modu**l predstavlja modul koji može da realizuje sve funkcije za dat broj promenljivh uko mu se negiraju ili permutuju ulazi podataka. **Multiplekser** je standardno kolo koje se može posmatrati kao višeulazni prekidač koji u zavisnosti od vrednosti kontrolnih ulaza propušta jedan od ulaza podataka na izlaz i predstavlja NP-kompletan modul. Za broj ulaza podataka, n, kontrolnih signala mora biti k = (najmanji ceo broj veći od k, pri čemu je broj ulaza podataka n = 2k ) i ta vrednost se smatra veličinom multipleksera.

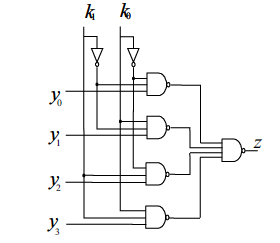
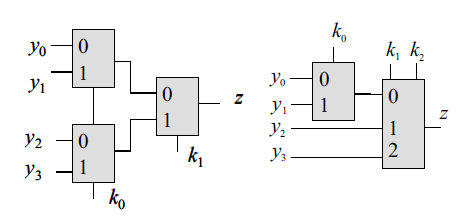
Realizacija MUX 2x1 upotrebom NI kola:

 odnosno simbolično: 

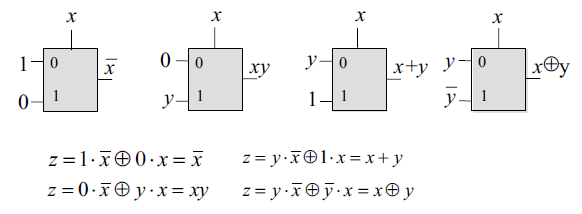
Može se uočiti da je multiplekser 2x1 kolo koje realizuje Šenonov razvoj po promenljivoj k što se primećuje i iz jednačine MUX-a :



MUX date veličine se može izraziti odnosno realizovati mrežom (razgranate strukture, „tree“ ) multiplekserima manje veličine:



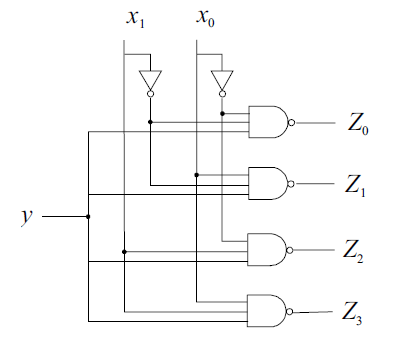
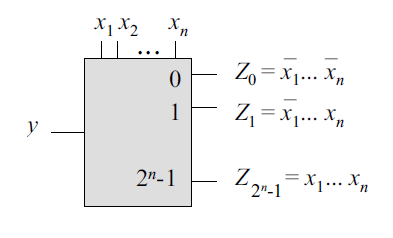
*MUX 4x1 realizovan: preko mux 2x1; preko mux 3x1 i mux 2x1; NI kolima.*



*Različite f-je realizovane multipleserom primenom NP klasifikacionih pravila.*

Kolo kojim je realizovana obrnuta operacija od one koju izvršava mux se zove **demultiplekser**. Kod DMUX jedan ulaz je usmeren ka 2n mogućih izlaza pri čemu je n broj kontrolnih ulaza. Mux mreža može se pretvoriti u ekvivalentnu DMUX mrežu dodavanjem logičkih ILI kola.

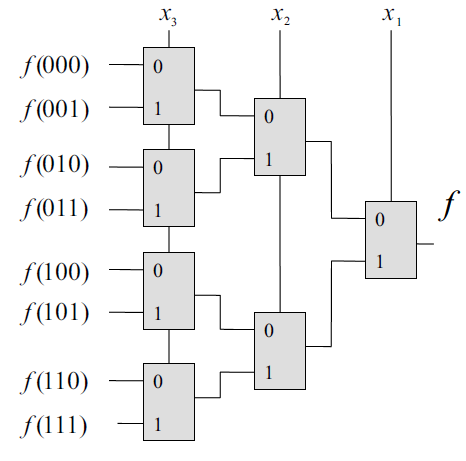
Realizacija demultipleksera 1x2 NI kolima:

 . Simbol DMUX:

**24. Sinteza i optimizacija logičkih mreža sa multiplekserima i demultiplekserima (FALI ZA DMUX)**

Pri sintezi logičkih kola upotrebom MUX-a javlja se problem koji se može opisati uvođenjem pojma cene. Cena je utoliko manja, za dat skup multipleksera kojima se može formirati mux veće dimenzije u odnosu na dimenzije onih koji su dati, ukoliko se koristi manji broj datih multipleksera. Takođe, cena je manja ukoliko je manje propagaciono kašnjenje formirane mreže. Optimalna sinteza je sinteza sa što manjom cenom. Problem je i projektovati mrežu koja realizuje funkciju f od n promenljivih korišćenjem multipleksera sa k kontrolnih ulaza pri čemu je k manje od n. U tom slučaju ulazi podataka mogu biti neki od 2k mogućih kofaktora funkcije f .

Pri čemu su funkcije odgovarajući kofaktori i predstavljaju permutaciju standardnih kofaktora dobijenih primenom Šenonovog razvoja. Da bi se odredile funkcije što manje složenosti obično se primenjuje rekurzivno Šenonov razvoj što vodi do razgranate strukture po nivoima mux-a. Ako primenimo Šenonov razvoj na svih n promenljivih dobićemo kompletno grananje pri čemu su ulazi podataka vektor istinitosti funkcije koju realizujemo:

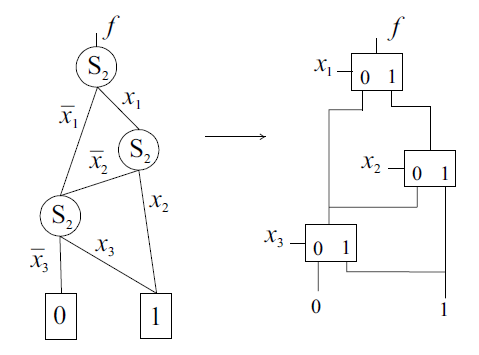
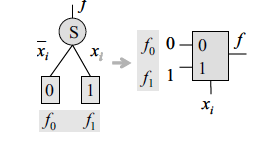


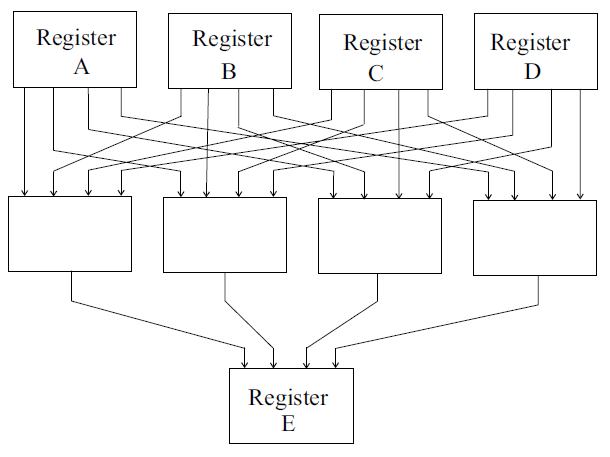
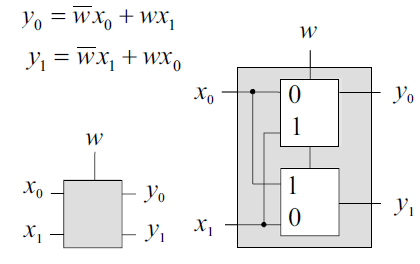
Optimizacija mreže obično podrazumeva smanjenje broja korišćenih mux-a.

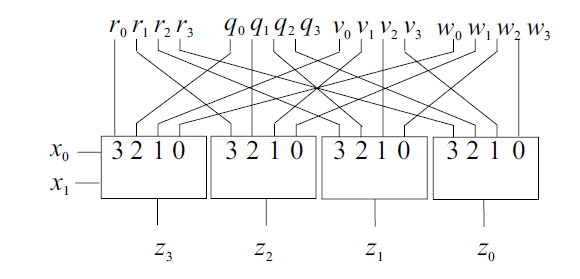
Spektralni metod koji garantuje minimalnu mrežu koristi Volšovu transformaciju da odredi kofaktore f-ja. Pored razgranate strukture ostoji i struktura serijskog povezivanja mux-a pri čemu su kontrolni ulazi Rid –Milerov razvoj pozitivne polarnosti.

**25. Dijagrami odlučivanja i multiplekserske mreže**

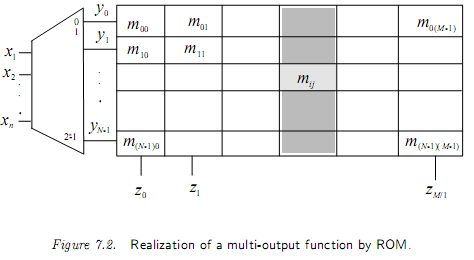
Postoji direktna veza između multipleksera i dijagrama odlučivanja koji su tipa BDD budući da su neterminalni čvorovi u BDD-u definisani kao grafička reprezentacija Šenonovog razvoja čiju realizaciju predstavlja multiplekser. Svaki neterminalni čvor datog BDD-a uključujući i koren treba uameniti 2x1 multiplekserom.

Multiplekserske mreže se često koriste kod povezianja registara u računaru i za formiranje proizvoljne 4bitne reči od 4 različitih reči uzimajući odgovarajuće bitove iz svake reči:

Takodje česta mreža je i „Cross-bar“ mreža za permutaciju ulaza za date izlaze i obrnuto:

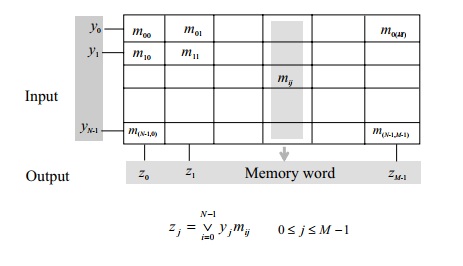


**V Realizacija programabilnim logičkim strukturama (6)**



**26. Realizacija sa ROM,**

ROM-ovi mogu da se koriste za realizaciju viseizlaznih funcija za M izlaza ako memorijska celija mi,j sadrzi i-tu vrednost j-tog izlaza funcije izrazene u DNF.

U ovom slucaju ulazi yi, i = 0,..., 2n −1 se odredjuju kao izlazi adresnog dekodera: Dekoder, znaci za svaku kombinaciju ulaznih promenljivih aktivira se po jedan ulaz y, i cela vrsta gde se on nalazi preslikava na izlaz.Ocigledno da se za realizaciju ROM-ovima koristi kompletna disjunktivna forma.z-ovi su trenutne vrednosti izlaznih funcija.Svi z-ovi zajedno formiraju izlaznu memorijsku rec.

Na primer evo kako bi izgledao trobitni sabirac,prikazan najpre vektoroma istinitosti:

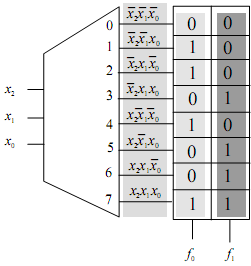
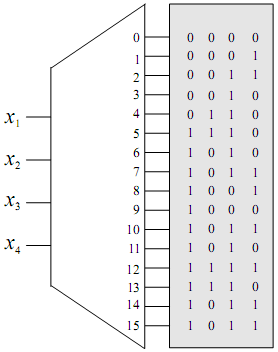
F0 = [0, 1, 1, 0, 1, 0, 0, 1]T

F1 = [0, 0, 0, 1, 0, 1, 1, 1]T .

Ova dvoizlazna funkcija moze se realizovati pomocu ROM-a kao na slici.

Ocigledno je da se u ovoj realizaciji koristi potpuna DNF:

f = x1x2x3f(0) ∨ x1x2x3f(1) ∨ x1x2x3f(2) ∨ x1x2x3f(3) ∨x1x2x3f(4) ∨ x1x2x3f(5) ∨ x1x2x3f(6) ∨ x1x2x3f(7).



A ovakokonvertor BCD to Grayjeuredjajciji je ulaz 4b BCD broj I izlaz 4b Gray code broj. Ovo se mozeopisatisledecimskupom f-ja:

f1 = x1 + x2x4 + x2x3

f2 = x2x3

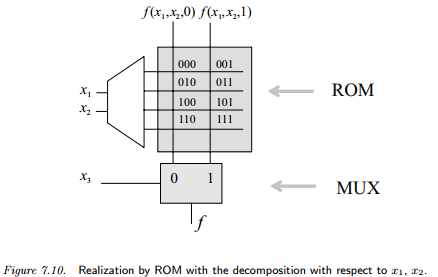
f3 = x2 + x3

f4 = x1x2x3x4 + x2x3x4 + x1x4 + x2x3x4

**27. Optimizacija u sintezi sa ROM**

**Realizacije ROM-ova u dva nivoa adresiranja**

Pri realizaciji ROM-ovima, cuva se kompletni vektor istinitosti.Medjutim, kod prakticnih primena, moze se desiti da data funkcija ne moze da se smesti u standardne dimenzije ROM-a, ili slobodni prostor ne dozvoljava da se postavi potrebni ROM.Tada se optimizacija vrsi tako sto se adresiranje vrsi u dva nivoa.Set promenljivih  se deli na dva podseta:  tako da se jedan dovodi na adresni dekoder, a drugi na mrezu multipleksera.



Adresiranje u dva nivoa ima sledece prednosti:

1. Smanjuje se velicina ROM-a redukovanjem broja vrsta

2. Optimizuje se adresni dekoder tako sto ga menjamo optimizovanom mrezom multipleksera.

3. Selekcija optimizovane mreze multipleksera na izlazu.

Karakteristike realizacije sa ROM-ovima

**Mane:**

1. Funkcija je data vektorom istinitosti i zato nema minimizacije u smislu produktnih clanova u SOP izrazima.

2. Zbog toga ROM-ovi su neefikasni u slucajevima kada funkcije imaju mnogo 0 ili 1 vrednosti.

**Prednosti:**

1. ROM realizacije su korisne u slucajevima kada funkcije koje treba da se realizuju opisane istinitosnim tablicama koje se direktno skladiste.

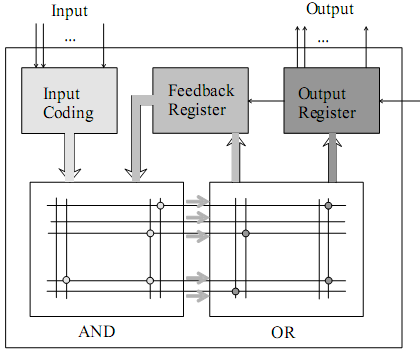
2.Takva realizacija je efikasna kod f-ja koje imaju mnogo produktnihclanova u SOP formama.Na primer funkcije za opis aritmetickih kola.

3. Efikasne su kada je potrebno cesto menjanje logike mreze.Primer su konvertori kodova.Lakse cemo samo da upisemo u ROM i zavrsimo posao nego da realizujemo mrezu ponovo za drugi problem.

**28. PLA,**

**Realizacija sa PAL(Programable Logic Arrays)**

Najjednostavniji oblik PLA sastoji se od 2 matricna elementa,gde prvi primenjuje uslove proizvoda izabranih promenljivih, a drugi implementira sume odabranih proizvoda



PLA-ovi sa ulaznim kodiranjem mogu se posmatrati kao structure memorije sa adresiranjem, preko poveznaih ili prevedenih funkcija. Onda je ulaza dresa,a izlazi su vrednosti funkcija za ulazne parameter specificirane od strane adrese.

1. AND mreza koja realizuje logice AND operacije i generise proizvode(implikante) ulaznih promenjivih

2. OR matrica ili alternativno EXOR koja vrsi sabiranje tih implikanata

3. Ulazni registar za ubacivanje podataka u AND matricu

4. Izlazni registar za prebacivanje izlaza iz OR matrice na ulaz AND matrice

5.Feedback (povratni) registar za povezivanje izlaza OR (EXOR)-matrice na ulaz AND matrice

PLA ze koristi za realizaciju prekidackih funkcija u DNF obliku, i kao sto vidimo feedback registri omogucavaju realizaciju sekvencijalnih mreza (automata).

Realizacija mreze se vrsi uspostavljanjem veze izmedju horizontalnih i vertikalnih linija u AND i OR(EXOR) matricama.

Ova metodologija dizajna se zove personalizacija PLA i izvodi se rasporedjivanjem veza izmedju AND i OR matrica

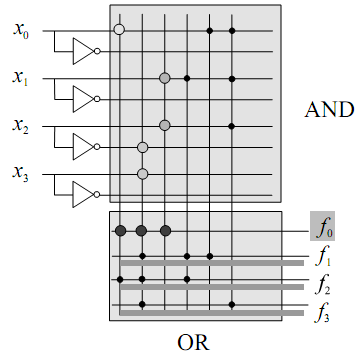
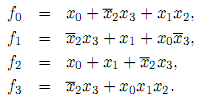
Kod nekih PLA veze u AND mrezi se prave Sotki diodama, a u OR mrezi tranzistorima.

Pored PLA, postoje naprave sa ogranicenom programibilnoscu.Na primer PAL naprave imaju programibilnu AND mrezu, ali se veze izmedju proizvoda ulaznih promenljivih i specificnih OR kola kratkospajaju.Broj tih izlaza sa proizvodima koji predstavljaju ulaze u OR kola je obicno ogranicen na 2,4,8 i 16. Za razliku od PLA kod PAL nije moguce podela proizvoda ulaznih promenljivih (implikanata).

**Realizacija sa PLA**

Realizacija f-je prebacivanja moze se izvrsiti na 3 nacina:

1.funkcionalni dizajn



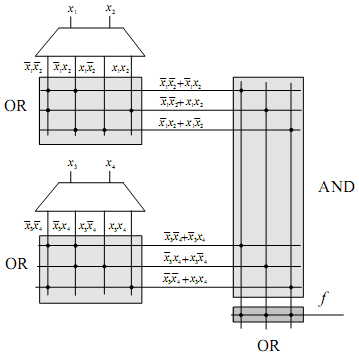
2.topoloski dizajn

3.fizicki dizajn

**29. Optimizacija PLA**

Kompleksnost PLA odredjuje se brojem ulaza i podrucjem koje zauzima.Tako da se optimizacija vrsi kroz redukovanje broja ulaza, kada je to moguce,posto ovo rezultira takodje smanjenje broja izlaza i podrucja.Ipak, redukcija broja ulaza koja redukuje broj vrsta u AND matrici ne mora uvek da znaci redukovanje broja kolona AND matrice.Postoje primeri gde redukcija broja uzlaza povecava broj implikanata.Redukcija implikanata se vrsi kroz standardne metode minimizacije DNF-ova.

U prakticnim primenama je vaznije smanjenje broja ulaza od smanjenja implikanata za smenjenje povrsine PLA.

Adresiranje PLA u dva nivoa

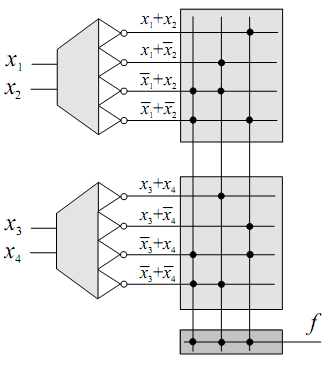
Optimizacija PLA moze da se postigne adresiranjem u 2 nivoa.Kod ovog pristupa, set ulaznih promenljivih se deli na podsetove i pomocna matrica D generise minterme(proizvode ul. prom.) u zavisnosti od promenljivih za svaki od podsetova.Izlaz D matrice je ulaz AND mreze,a izlazi AND mreze se vode naravno na ulaze OR mreze.Matrica D moze da se realizuje adresnim dekoderom ili moze da se zameni drugom OR matricom cime dobijamo

"OR-AND-OR" PLA.



Preklapanje PLA

Kod prakticnih implementacija AND i OR matrice u PLA su obicno retke posto se radi logicka minimizacija.Ova nepopunjenost moze da se iskoristi u optimizacionoj metodi za PLA koja se zove preklapanje, i sluzi za redukovanje prostora koji zauzima PLA, kao i za redukovanje kapacitivnosti linija, sto kao rezultat daje brza kola.Tehnika se sastoji u permutaciji kolona i vrsta ili oba cime se dobija maksimalan broj kolona i vrsta koje se mogu implementirati u istoj koloni, vrsti ili fizickoj mrezi.Na ovaj nacin PLA se deli na nekoliko AND i OR matrica. Deljenje je moguce kada su izlazi proizvoda (implikanata) za razlicite izlaze disjunktni.



Prethodna slika prikazuje realizaciju sa podeljenom AND mrezom na 2 dela za funkciju f koja je definisana gore u tekstu.Primetimo da su ove dve mreze disjunktne.

1. Prosto preklapanje - gde par ulaza deli istu kolonu ili vrstu respektivno.Pretpostavlja se da su ulazne i izlazne linije na gornjim ili donjim delovima kolona pa tako nema preseka izmedju preklopljenih linija. Najcesce ulazne i izlazne linije se preklapaju u AND i OR matrici respektivno zbog elektricnih i razloga fizicke realizacije.

2.Visestruko preklapanje -je generalnija tehnika gde su izlazne linije preklopljene sto je vise moguce kako bi se smanjio broj kolona i vrsta u AND i OR matrici.Ovaj metod smanjuje podrucje(povrsinu). Ipak, rutiranje ulaznih i izlaznih linija je komplikovaniji posao, pa moze da zahteva silikonske medjuslojeve kod realizacije.Zato se primenjuje samo kod velikih sistema gde su takvi slojevi vec potrebni zbog drugih stvari.

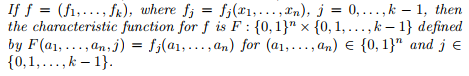
**30. Karakteristične funkcije i viseznačna logika u optimizaciji PLA**

Viseznacne funkcije se definisu kao mapiranje



gde je , n brojpromnljivih I k brojizlaza.

Karakteristicne funkcije (def):



Posto se svaka funkcija sa vise izlaznih f-ja moze izraziti kao binarna f-ja multiple-valued ulaza, minimizacija poslednjeg vodi do minimizacije prvog.

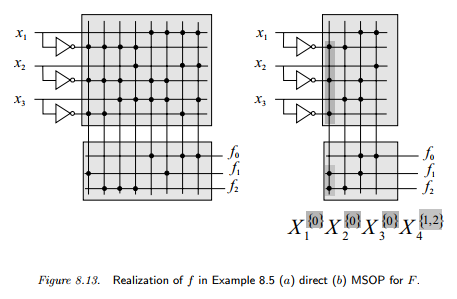
Moze se pokazati da u mrezina 2 nivoa izvedeno iz minimalnog SOP za F, broj AND kruzenja bice minimalni. Medjutim broj veza u mrezi nije uvek minimalan. Smanjenje AND matrica je glavnicilj PLA dizajna, min SOP-ovi za karakteristicne f-je F se mogu koristiti za dizajniranje PLA sa smanjenim nizom(redom).

Primer: SOP izraz f-je F je:



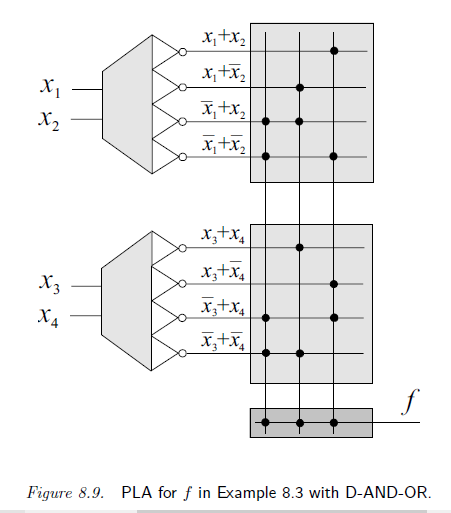
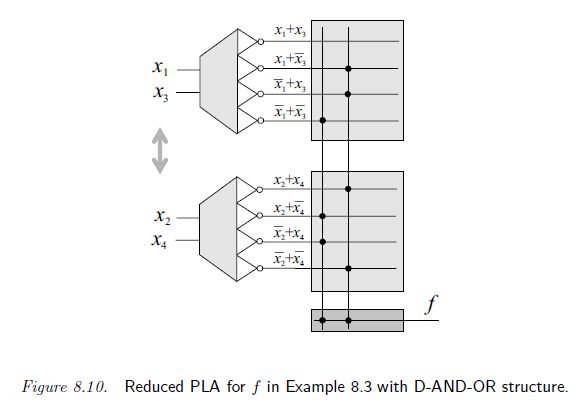
A min SOPove f-je je: (njom se smanjuje broj kolona)

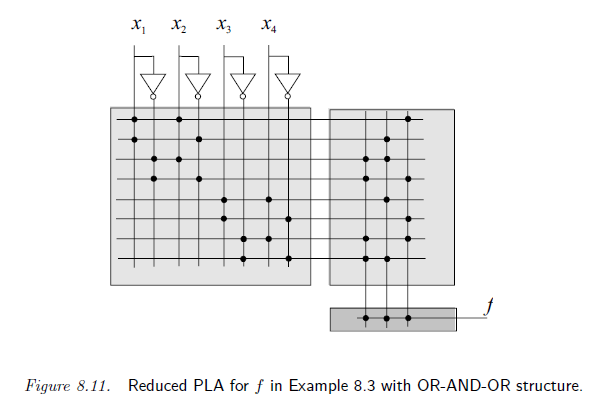
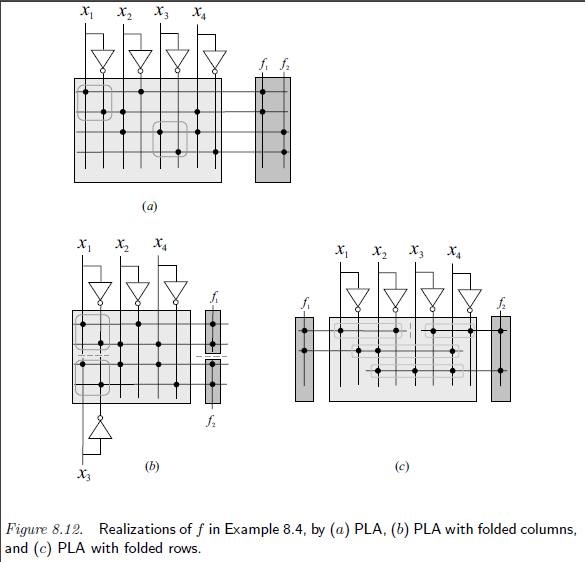




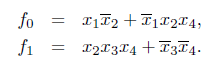
**31. Savijanje PLA,**

U praktičnim izvođenjima, I i ILI matrice u PLA su obično retke (proređene), s obzirom na to da se izvršava logička minimizacija. Ova proređenost može biti iskorišćena sa tehnikom optimizacije pod nazivom **savijanje PLA**, da bi se smanjio polje koje zauzima PLA, kao i broj linija koje proizvode brža kola. Tehnika se sastoji iz nalaženja permutacija kolona i vrsta, ili oboje, koja proizvodi maksimalni set kolona i vrsta koji mogu biti implementirani u istoj koloni, respektivno vrsti, fizičkog polja. Na ovaj način, PLA je podeljen na nekoliko I i ILI matrica.



1. Jednostavnosavijanjekada par ulaza ili izlaza dele istu kolonu ili vrstu, respektivno. Pretpostavlja se da su ulazne i izlazne linije na ili gornjoj ili donjoj strani kolona, tako da ne preseka između linija. Najčešće, ulazne i izlazne linije su savijene u I i ILI matrici, respektivno, zbog električnih i fizičkih ograničenje.
2. Višestruka savijanja je generalnija tehnika gde su ulazne I izlazne linije savijene najviše moguće da bi se minimizovao broj kolona, respektivno vrsta u I i ILI matrici. Ova metoda smanjuje polje. Međutim, uvođenje ulaznih I izlaznih linija je komplikovanije, I jos jedan metalni ili polisilikonski sloj je možda potreban. Stoga, višestruko savijanje je efikasno kada je PLA komponenta velikog sistema gde su već potrebni višemetalni i polisilikonski slojevi.
3. Dvodelno savijanje je poseban primer jednostavnog savijanja gde se prekid kolone na dva dela u istoj koloni mora desiti na istom horizontalnom nivou ili u I ili u ILI matrici.
4. Ograničeno savijanje je zabranjeno savijanje gde neka ograničenja kao štoje red. Umesto linija su data I prilagođena sadrugim savijanjem.

Razmotrite 2 funkcije od 4 promenljive.



**32. FPGA, vrste i klasifikacija, način programiranja**

Field Programmable LogicArrays (FPGA) omogucavaju veliki stepen integracije i programabilnost od strane korisnika. Pri poredjenju sa PLA na primer, imamo sledece standardne mogucnosti

1. moze se raditi realizacija u vise nivoa

2. Broj ulaza u kolo je manji

3. Kompaktniji su nego realizacije u 2 nivoa

FPGA mozemo da posmatramo kao programabilne logicke cipove koji se sastoje od logickih blokova, gde svaki blok moze da realizuje set logickih funkcija, i od programabilnih medjuveza i prekida izmedju blokova. Nesto kao Celijsko polje koje programiranjem mozemo da prilagodimo kako nama odgovara. Kompleksnost FPGA se obicno odredjuje poredjenjem njihovih celija koje mogu da se sastoje od:

1. para tranzistora

2. prostih logickih kola sa nekoliko ulaza

3. multipleksera

4. Lookup tabela

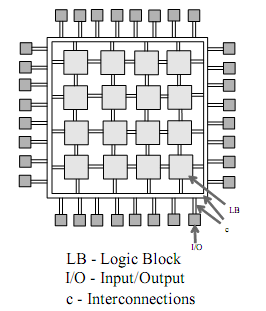
5. AND - OR struktura

**33. Diskusija osobina različitih klasa FPGA**

U ovom smislu, FPGA su klasirani u fino zrnaste i u grubo zrnaste, a podela je izvrsena na osnovu broja:

1. Ekvivalentnih NAND kola

2. Tranzistora

3. Normalizovanih podrucja koja se definisu kao odnos podrucja zauzetog logickim blokom i totalnog podrucja FPGA

4. Ulaza i izlaza

Velicina logickih blokova znacajno utice na performanse FPGA. Naime kod vecih blokova lakse je rutiranje, dok je manja iskoriscenost i sporiji su.

Pri rutiranju, medjuveze se uspostavljaju povezivanjem delovima linija u FPGA programabilnim "prekidacima". Broj segmenata definise gustinu elemenata.

Mali broj segmenata znaci smanjenje mogucnosti za njihovo slucajno povezivanje, dok veci broj segmenata podrazumeva da cevecina ostati neiskoriscenocime se gubi koristan prostor.

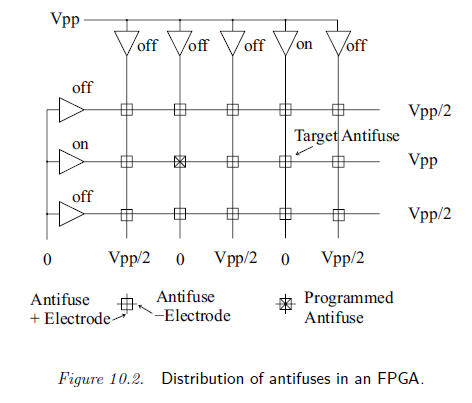
Manji segmenti zahtevaju vise prekidača sto uzrokuje kasnjenje, ali veliki segmenti zauzimaju prostor i takodje uzrokuju kasnjenje.

U zavisnosti od nacina programiranja FPGA se mogu podeliti na hard i soft programabilne FPGA. U prvom slucaju programiranje se vrsi kroz povezivanje segmenata medjuveza antifuse-ovima gde se otvorena kola pretvaraju u kratke spojeve odgovarajucim impulsom ili naponom. Druga klasa se vezuje za FPGA koji se sastoje od nizova memorijskih elemenata nazvanih "lookup tabele" u koje se programira i snima informacija o konfiguraciji modula, i medjuvezama.

FPGA koji su bazirani na lookup tabelama se programiraju na isti nacin kao i memorijski cipovi gde se rec konfiguracionih podataka upisuje na adresirani segment niza. Svaki bit u memoriji niza kontrolise specificni konekcioni element (medjuvezu). Nekoliko ovakvih elemenata koji se istovremeno programiraju moze da bude najvise sirine memorijske reci, i programiranje se vrsi dovodjenjem napona 0-5 volti u sekvenci specificnoj za dati FPGA.

FPGA bazirani na antifuse metodi se programiraju kombinovanjem digitalnih kontrolnih, i analognih signala veceg napona.Svaki antifuse se programira odvojeno u vremenskim intervalima, redom dok se ne odradi ceo niz njih. Niz antifuse-ova moze se posmatrati kao skup vertikalnih i horizontalnih zica sa antifuse-om kod svakog preklapanja ili presecanja zica.

Antifuse ima 3 stanja off-state, on-state, i off-on transitionstate. U off stanju, antifuse se sastoji od neprovodnog materijala. Primenom impulsa postaje tecan i pravi se kompleksna mesavina metala i silicijuma, i to je stanje off - on. U zavrsnom stanju antifuse postaje provodni kristal sa malom otpornoscu.Proces programiranja je ireverzibilan.



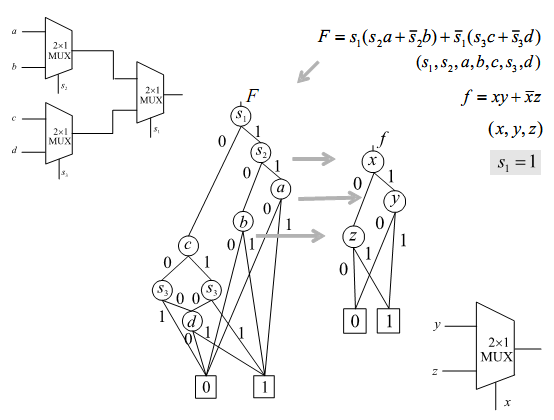
**34. Relizacije sa FPGA**

Dizajniranje pomocu FPGA moze da se posmatra kao prilagodjavanje programabilnih logickih modula od kojih se sastoje kako bi oni realizovali funkcije koje se od njih traze.

U slucaju Antifuse FPGA cesto se uzme za pocetak da svi programabilni moduli mogu da realizuju isti tip jednoizlazne funkcije nazvane "modul funkcija". Zato, modul funkcija predstavlja opis logickog bloka. Klaster funkcija je funkcija koja opisuje deo mreze. Zadatak je odrediti da li klaster funkcija moze da se implementira preko modul funkcije F, kada nam je zadata funkcija f neke mreze koja treba da se realizuje, projektovanje se sastoji od nalazenja ekvivalentne logičke mreže sa minimalnim brojem logickih kola i minimalnim kasnjenjem duz staza, sto se svodi na personalizaciju modul funkcije F. Personalizacija podrazumeva definisanje nekih parametara modul funkcije na konkretne vrednosti.

**35. Preslikavanje dijagrama odlučivanja na FPGA**

Na primer ako je modul funkcija F nekog FPGA multiplekser.Tada identicno mozemo iz redukovanog BDD-a da realizujemo mrezu pomocu takvih modul funkcija kao sto smo vrsili realizaciju multiplekserima na osnovu BDD-ova.



**VI Testiranje logičkih mreža (4)**

**36. Definicije i modeli grešaka u logičkim mrežama**

Iako su logicke mreze danas veoma pouzdane u vidu davanja netacnih izlaza u bilo kom trenutku, testiranje logickih kola je jedno od najvecih problema u ovoj oblasti. Greske se mogu javljati kao posledica pojave kratkih spojeva, pokvarene linije,pogresne vrednosti napona…Greske u logickim mrezama se mogu klasifikovati kao:

1 *Blage greske* su posledica prolaznih efekata kao sto sun a primer elektricna buka koja dolazi iz izvora struje,ili kosmicki zraci koji uzrokuju izbijanje jona na cipu.

2 *Jake greske* koje su trajne I mogu se javiti kao posledica mehanickih efekata kao sto su vibracija,dugotrajna korozija,migracija metala..

Blage greske se teze detektuju I najcesce se njima bave razliciti softveri kao i celokupne tehnike dizajniranja. Testiranje logickih mreza se najcesce bavi jakim greskama.

***Ekvivalnetne greske:*** Ukoliko su e1 I e2 dve greske u logickoj mrezi N. oznacavaju izlaze N u prisustvu gresaka e1 odnosno e2.Greske e1 I e2 su ekvivalentne ukoliko je . Procedura za oredjivanje ekvivalentnih gresaka u zadatom kolu se cove kolaps gresaka.Obicno se smatra da je pre testiranja uredjaja procedura kolaps gresaka vec izvedena.

***Visestruke greske:*** Istovremena pojava pojedinacnih gresaka

***Greske koje je nemoguce detektovati:*** Ukoliko za gresku e1 izlaz DUT-a je ,gde je f izlaz DUT-a bez greske ,tada je greska e1 greska koju je nemoguce detektovati.Takve greske se najcesce nazivaju i redundantnim greskama.

**37. Automatsko generisanje testova**

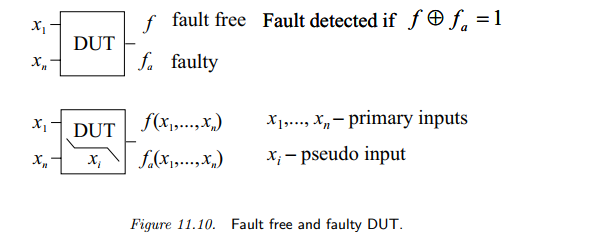
Test za detekciju greske u zadatom DUT-u se sastoji od zadavanja ulaznih vrednosti koje prouzrokuju izlaz suprotan vrednosti izlaza DUT-a bez greske. Kompletan test se sastoji od sekvenci koje su u mogucnosti da detektuju sve pretpostavljene greske za zadatu mrezu .Jasno je da test svih mogucih kombinacija ulaznih vrednosti je kompletan,ali I jako nepraktican za veliko n. Minimalan test je jako tesko naci,tako da je problem nalazenja gresaka koje je moguce detektovati za zadatu ulaznu sekvencu cesto razmatran.

Postoji vise razlicitih metoda za automatsko generisanje testova koje se mogu klasifikovati kao:

1 Algebarske metode koje se baziraju na manipulaciji algebarskih izraza koji opisuju funkciju realizovanu izabranim DUT-om.

2 Topoloske metode koje su povezane sa topologijom DUT-a, u vezi su sa povezivanjem kola, medjusobnim spojevima.

**38. Booleova diferenca i primene u otkrivanju grešaka**



Osnovni princip primene bulove diference u testiranju kombinacionih mreza je dat na slici. Ako je izlaz DUT-a bez greske f i izlaz sa greskom ejf , tada je detektovana greska za f ⊕fej = 1.Za DUT bez greske f je funkcija primarnih ulaza npr. *f* = *f*(*x*1*, . . . , xn*).Prisustvo zaglavljene greske na liniji q uzrokuje liniju da se ponasa kao

lazan ulaz xq.

Izlaz DUT-a postaje *feq* = *F* (*x*1*, . . . , xn, xq*).

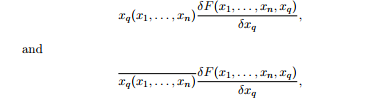
Od razlicitih mogucnosti za n primarnih ulaza za testiranje koristimo one koji omogucuju:

1 Nadrazivanje greske-vrednost na liniji q je suprotna od one koja je izazvala gresku.

2 Propagaciju greske-koja omogucava da prisustvo greske utice na izlaz tj. produkuje izlaz suprotan vrednosti DUT-a bez greske.

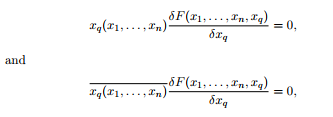
Test na zaglavljenje na greski se moze odrediti koriscenjem sledece observacije:

Zadata n-torka (a1,….an) ulaznih vrednosti je za test sekvenca za detekciju zaglavljenja na greski =0 tj =1 ako je minterm istinit u funkcijama



Prvi deo ili omogucava vrednost suprotnu onoj koja je izazvana greskom.Drugi deo osigurava da ce greska propagirati do izlaza, xq je esencijalna varijabla F.Test se sastoji od sekvenci koje zadovoljavaju prethodne uslove.

Stoga ako



Greske zaglavljene na 0 tj 1 se ne mogu detektovati,odnosno mi ne mozemo ni specificirati da li je linija zaglavljena na toj vrednosti kao rezultat greske ili da li greska ne utice na izlaz.

**39. Lako testablne mreže, njihovo projektovanje i optimizacija**

Kod dinamicke redundancije sistem se testira sa izabranom odgovarajucom frekvencijom i vremenski intervali su odredjeni tako da se greska moze detektovati i ispraviti zamenom odgovarajucih modula u razumno kratkom periodu.Za takve sisteme je veoma pogodno dizajnirati kola koja se mogu lako testirati.Takva kola se nazivaju lako testabilna kola.

DUT-uredjaj koji se testira

DUT mora da poseduje primarne ulaze i izlaze kojima se lako pristupa.Test se obavlja primenom odgovarajucih test sekvenci.

Lako testibilne mreze imaju sledece osobine:

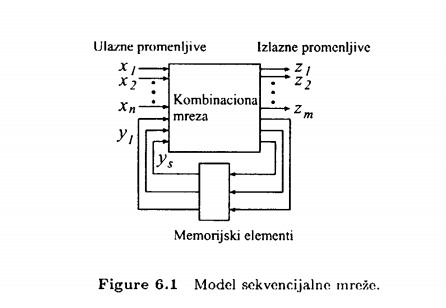
1.Set test sekvenci treba da bude sto kraci

2.Mreza ne sme biti staticki redundantna

3.Procedura izrade testa mora biti jednostavna I ,ako je moguce,deo metode za dizajn.

4.Test sekvenca treba da bude laka za generisanje

5.Pozeljno je da test odredi (sa odredjenom preciznoscu) lokaciju greske.

**VII Sekvencijalne logičke mreže (2)**

**40. Definicije i osnovni pojmovi**

Sekvencijlne mreze su logicke mreze sa memorijom.

Primer je mreza sa jednim ulazom I jednim izlazom koja na izlazu daje 1 ako I samo ako se na ulaz tri puta uzastopno dovede 1.

Kod kombinacionih mreza izlaz je zavisio iskljucivo od vrednosti ulaznih promenljivih u posmatranom trenutku ,medjutim kod sek. mreza vrednost izlaza zavisi ne samo od trenutnih vrednosti na ulazu vec i od prethodnih vrednosti ulaza.

Sekvencijalne mreze se predstavljaju dijagramima stanja I tablicama stanja.

U sinhronim sekvencijalnim mrezama taktni impuls sinhronizuje promene promenljivih stanja. Asinhrone sek. mreze ne poseduju taktni signal.

Memorijski elementi koji se koriste u sek. mrezama su najcesce lecevi ili flip-flopovi.

**41. Modeli sekvencijalnih mreža**

Ponašanje sekvencijalnih mreža je opisano matematičkim modelom koji se zove konačni automat, i može biti predstavljan tablicom stanja ili dijagramom stanja.

Bavićamo se samo konačnim i determinističkim automatima (za svaku kombinaciju tekućeg stanja i ulaza, tačno je definisano sledeće stanje).

Deterministički konačni automat je definisan uređenjom petorkom M=(Σ, Q, Z, f, g) gde je:

1) Σ - konačan neprazan skup ulaznih simbola σ1,..., σl, koji se takođe zove i ulazna azbuka,

2) Q - konačan neprazan skup stanja q1,...,qn, azbuka stanja,

3) Z - konačan neprazan skup izlaznih simbola z1,...,zm, izlazna azbuka,

4) f - funkcija stanja,

5) g - funkcija izlaza.

Postoje 2 osnovna tipa determinističkih konačnih automata Milijev i Murov automat

Deterministički konačni automat je Milijev ako

1) je funkcija stanja f : Q × Σ → Q i

2) funkcija izlaza g : Q × Σ → Z.

Deterministički konačni automat je Murov ako

1) je funkcija stanja f : Q × Σ → Q i

2) funkcija izlaza g : Q → Z.

**42. Načini predstavljanja sekvencijalnih mreža**

Ponašanje sekvencijalnih mreža je opisano matematičkim modelom koji se zove konačni automat, bavićemo se samo determinističkim konačnim automatima, i može biti predstavljan tablicom stanja ili dijagramom stanja.

Tabela stanja

Da bi se definisao deterministički konačni automat mora biti dat skup stanja, ulazni simboli, izlazni simboli i funkcije izlaza i prelaza stanja. Ove informacije mogu biti predstavljene adekvatno u formi tabele stanja.

Primeri:

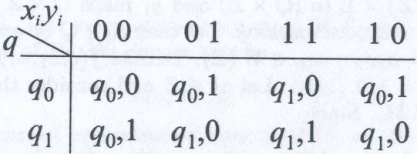


Tabela stanja binarnog sabirača realizovanog kao Milijev automat

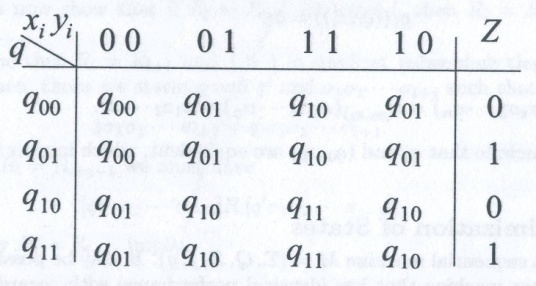
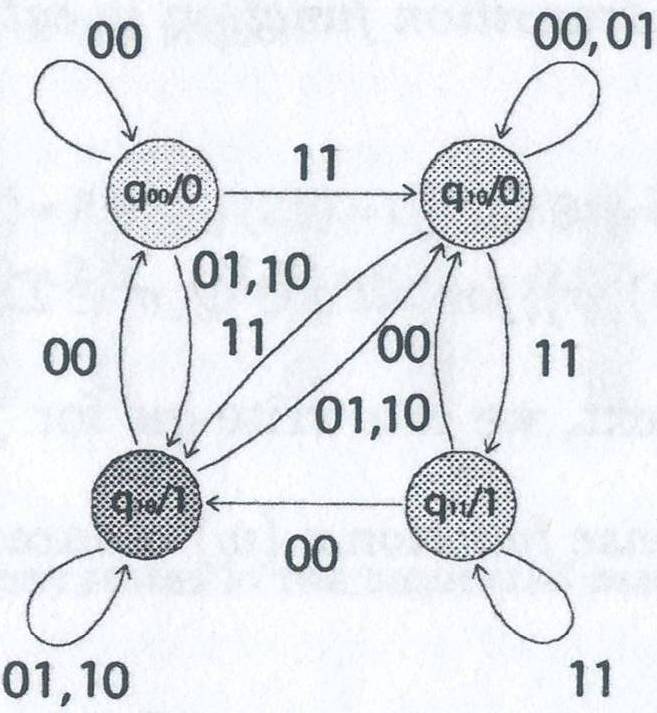
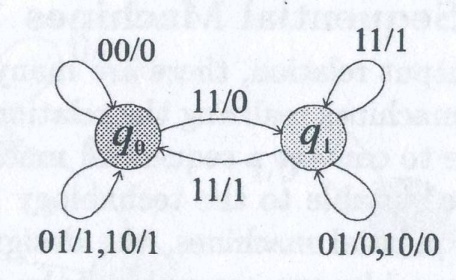


Tabela stanja binarnog sabirača realizovanog kao Murov automat

Dijagram stanja

Dijagrami stanja su korisni za vizuelizaciju različitih svojstava konačnih automata.Ako je neka od funkcija koja definiše konačni automat, funkcija stanja ili funkcija izlaza nepotpuno definisana, tada je i konačni automat takođe nepotpuno definisan.

Primeri:



Dijagram stanja binarnog sabirača realizovanog kao Milijev automat

Tabela stanja binarnog sabirača realizovanog kao Murov automat

**43. Kodiranje stanja i mimimizacija sekvencijalnih mreža**

Važan zadatak je naći binarno kodiranje za ulaze, izlaze i stanja takvo da je realizacija binarne reprezentacije jednostavna koliko god je to moguće. Kodiranje stanja je predstavljanje svakog od stanja određenom jedinstvenom binarnom kombinacijom. Prilikom kodiranja potrebno je voditi računa da se za kodiranje koristi minimalan mogući broj bitova kojim se može kodirati odgovarajući broj stanja konačnog automata.

Sa tačke gledišta projektovanja, najvažniji zadatak je minimizacija sekvencijalne mreže. Određena funkcija može biti realizovana na beskonačan broj načina različitim automatima, a zadatak je naći automat sa minimalnim brojem stanja.

Dva stanja konačnog automata su ekvivalentna ako polazeći od bilo kog stanja za istu ulaznu reč generiše potpuno isti izlaz. Tokom minimizacije, želimo da identifikujemo ekvivalentna stanja i eliminišemo suvišna.

Razmotrimo konačni automat M=(Σ, Q, Z, f, g). Možda je moguće naći drugi automat koji ima istu funkcionalnost u odnosu na odziv izlaza na ulaze, ali koji ima manji broj stanja. Od velikog je značaja da možemo da nađemo "najmanji" automat. Za potpuno definisani automat postoji jedinstveno rešenje za ovaj problem i algoritam za njegovo nalaženje.

//Ne znam da li treba ovaj algoritam ima pojmova za koje nisam siguran kako se prevode i poduži je a Radmanović je rekao da svaki odgovor mora da može da se napiše za 15 minuta

**44. Dekompozicija sekvencijalnih mreža**

Važan zadatak je hijerarhijska dekompozicija kompleksnih konačnih automata na manje subautomate koji mogu biti analizirani i optimizovani posebno.

SERIJSKA DEKOMPOZICIJA KONAČNOG AUTOMATA

Definicija (Serijska konekcija konačnih automata)

Serijska konekcija dva konačna automata M1=(Q1, Σ1, Z1, f1, g1) i M2=(Q2, Σ2, Z2, f2, g2) gde Σ2=Z1 je konačni automat M=(Q1×Q2,Σ1,Z2,f,g), gde su funkcija stanja f i funkcija izlaza g

f((q,p),σ)=(f1(q,σ),f2(p,g1(q,σ))),

g((q,p),σ)=g2(p,g1(q,σ)).

Konačni automat M1-M2, koji je serijska konekcija M1 i M2, je serijska dekompozicija datog konačnog automata M ako i samo ako M1-M2 realizuje M. Dekompozicija je netrivijalna akko je ukupan broj stanja u M1 i M2 manji nego broj stanja u M.

PARALELNA DEKOMPOZICIJA KONAČNOG AUTOMATA

Definicija (Paralelna konekcija konačnih automata)

Paralelna konekcija dva konačna automata M1=(Σ1, Q1, Z1, f1, g1) i M2=(Σ2, Q2, Z2, f2, g2) je konačni automat M=M1|M2=(Σ1 × Σ2,Q1 × Q2, Z1 × Z2, f, g) gde su funkcije stanja f i izlaza g

f((q1, q2), (x1, x2))=(f1(q1, x1), f2(q2, x2)),

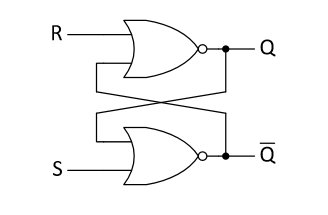
g((q1, q2), (x1, x2))=(g1(q1, x1), g2(q2,x2)).

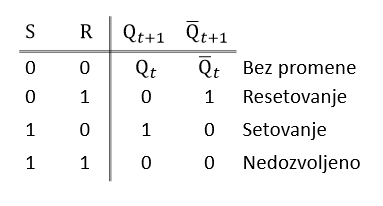
Konačni automat M1|M2 je paralelna dekompozicija konačnog automata M, akko M1|M2 realizuje M. Dekompozicija je netrivijalna akko je ukupan broj stanja u M1 i M2 manji nego broj stanja u M.

**VIII Realizacija sekvencijalnih mreža (2)**

**45. Elementrani automati i njhova primena u realizaciji sekvencijanih mreža**

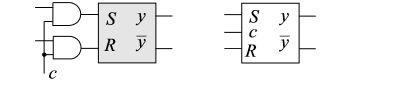
Glavna razlika između kombinacionih i sekvencijalnih mreža postojanje memorijskih elemenata koji pamte osnovne informacije o vrednostima prethodnih ulaza. Prvo ćemo ukratko razmotriti osnovne memorijske elemente koji se koriste u te svrhe. Oni se takođe zovu elementarni automati a u hardveru se realizuju kao flip-flopovi. Njihova osnovna karakteristika je da ostaju u uspostavljenom stanju i nakon izostanka pobudnih (ulaznih) signala koji su ih u to stanje doveli. U ovom stanju mogu ostati neograničeno dugo, sve do pojave novih ulaznih signala. Oni imaju 2 stabilna stanja (0 i 1) koja služe za pamćenje jednog bita u računaru. Najjednostavniji flip-flopovi su lečevi, a različiti flip-flopovi sa dodatnim karakteristikama se često projektuju na bazi lečeva. Među najčešće korišćenim su SR leč, D leč, SR flip-flop, D flip-flop, JK flip-flop, T flip-flop.

**SR LEC**



**Taktovani SR lec**

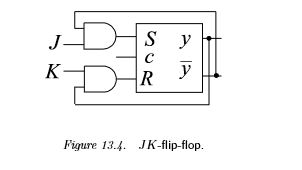
Jeste SR flip-flop



Menja stanje kad je za c=1,odnosno sa rastucom ivicom taktnog impulsa.

Postoje neke restrikcije uvezi sa koriscenjem SR flip-flopa, a odnose se na trajanje intervala izmedju dva taktna ciklusa I na trajanje taktnog impulsa. Ako je trajanje taktnog impulsa (c=1) prekratko, ulaz u lec nestane pre nego sto promeni stanje, takodje ako traje I suvise dugo, lec cepromeniti stanje dva puta u toku jednog takta.

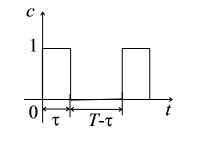
**JK flip-flop**



Razlika u odnosusa SR flip-flop sto je dozvoljeno da istovremeno budu aktivna oba ulaza, usled cega se trenutno stanje komplementira.

U praksi se flip-flopovi implementiraju kao MS(master-slave) ili kao flip-flopovi sa ivicnim okidanjem.

***MS***

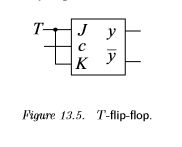
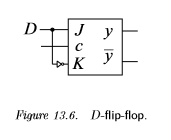
Ponasanje MS flip-flopova kontrolisano je rastucom I opadajucom ivicom taktnog impulse. Rastuca ivica izoluje slejv od mastera I cita ulaznu informaciju u master. Opadajuca ivica izoluje J i K ulaz od mastera i prenese informaciju u slejv.

***SA IVICNIM OKIDANJEM***

Samo jedna ivica-ili rastuca ili opadajuca uslovljava da flip-flop odgovori na ulaz i odmah odvezuje ulaz sa flip-flopa do sledece ivice takta-ili rastuce ili opadajuce.

**T flip-flop**

J i K ulazi su povezani u jedinstveni T ulaz.



**D flip-flop**

Bez obzira na promene ulaza D, izlaz Q se menja samo jedanput u toku svake periode taktnog signala - u trenutku rastuće ivice takta.

**46. Sekvencijalne mreže sa taktnim impulsima**

*Algoritam za sintezu sekvencijalne mreze sa taktnim impulsima*

1.Izvuci tabelu stanja iz opisa funkcije sekvencijalne mreze

2.Pojednostaviti uvek kada je moguce

3.Kodirati stanja

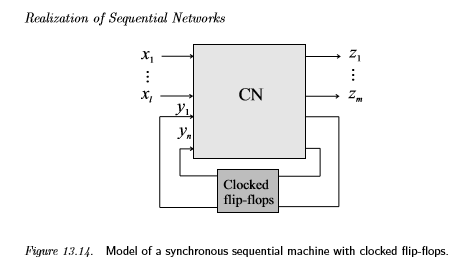
4.Izvuci funkciju stanja f I funkciju izlaza g

5.Minimizovati ove funkcije

6.Izabrati flip-flopove

7.Izvuci pobudnu funkciju ulaza flip-flopova

8.Realizovati kombinacionu mrezu funkcije pobude kao I funkcije izlaza sekvencijalne mreze.



Problem koji treba uzeti u obzir prilikom realizacije sekvencijalnih mreza je propagaciono kasnjenje taktnih impulse za sve flip-flopove u mrezi

**47. Sekvencijane mreže sa impulsnim načinom rada**

Sekvencijane mreže sa impulsnim načinom rada su sinhrone, ali nisu taktovane.

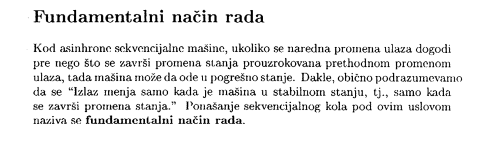
Sekvencijalna mreza radi u impusnom modu ako su zadovoljeni sledeci zahtevi:

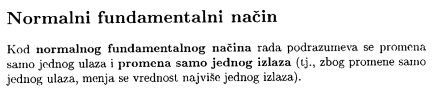
1. Ulazni signali su impulsi trajanja tau dovoljnog za dozvolu promene stanja svih flip-flopova u mrezi.
2. Impulsi su primenjeni na bilo kojem, ali samo na jednom ulazu.
3. Stanja se menjaju odgovorom na pojavu impulsa na ulazu i svaki impuls uzrokuje samo jednu promenu stanja. U praksi ovaj zahtev je ostvaren koriscenjem impulsa malih sirina

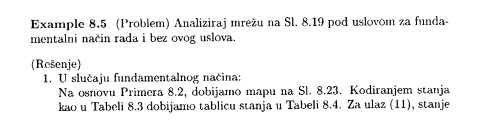
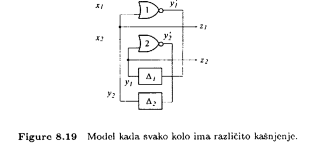
U slucaju Milijevih masina izlazi su impulsi. U Murovim masinama oni mogu biti i nivoi signala cija je vrednost odredjena u intervalima izmedju impulsa.

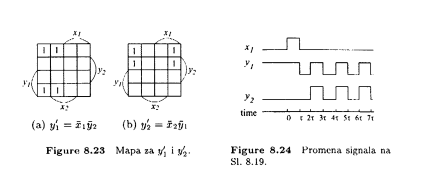
Sekvencijalne mreze impulsnog moda su uglavnom vezane za krug posebne namene koji nisu delovi vecih sistema. Klasicni primeri su prodavnice automobila i masine za sakupljanje putarine. U definisanju funkcija stanja potrebno je da svaki izraz proizvoda sadrzi ulaznu promenljivu,jer nema drugih impulsa- taktni impulsi se ne koriste.

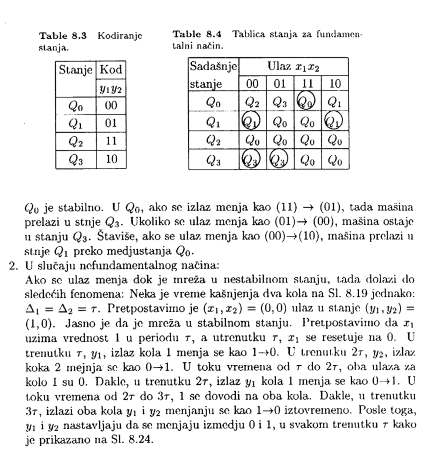
**48. Sekvencijalne mreže sa osnovnim načinom rada**

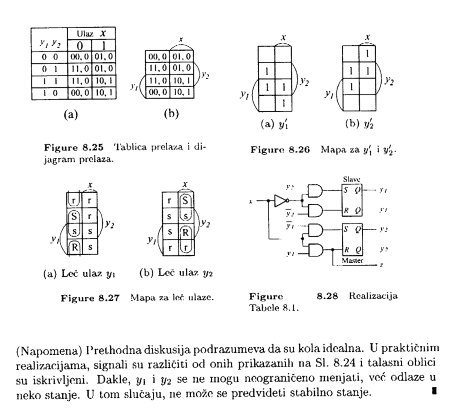












**49. Projektovanje asinhronih sekencijalnih mreža**

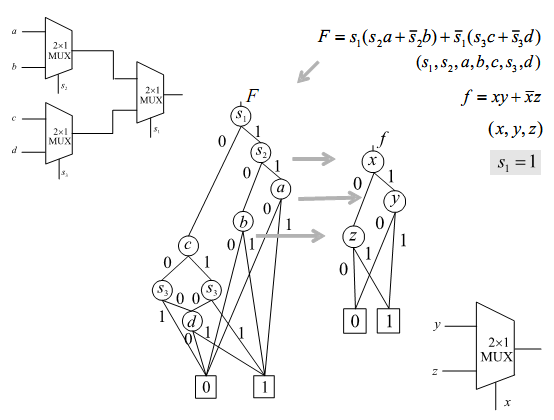
Primer na strani 162. u knjizi autora Sasao.

**Sinteza sa FPGA**

Dizajniranje pomocu FPGA moze da se posmatra kao prilagodjavanje programabilnih logickih modula od kojih se sastoje kako bi oni realizovali funkcije koje se od njih traze.

U slucaju Antifuse FPGA cesto se uzme za pocetak da svi programabilni moduli mogu da realizuju isti tip jednoizlazne funkcije nazvane "modul funkcija". Zato, modul funkcija predstavlja opis logickog bloka.Klaster funkcija je funkcija koja opisuje deo mreze.Zadatak je odrediti da li klaster funkcija moze da se implementira preko modul funkcije F, kada nam je zadata funkcija f neke mreze koja treba da se realizuje, projektovanje se sastoji od nalazenja ekvivalentne logicke mreze sa minimalnim brojem logickih kola i minimalnim kasnjenjem duz staza, sto se svodi na personalizaciju modul funkcije F.Personalizacija podrazumeva definisanje nekih parametara modul funkcije na konkretne vrednosti.

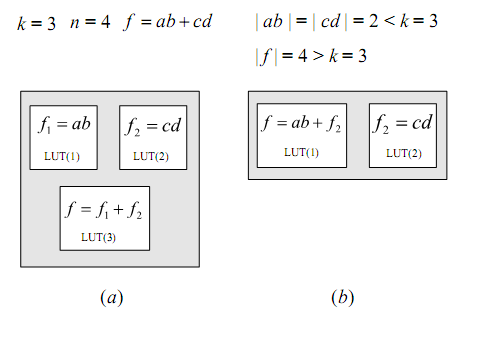
Na primer ako je modul funkcija F nekog FPGA multiplekser.Tada identicno mozemo iz redukovanog BDD-a da realizujemo mrezu pomocu takvih modul funkcija kao sto smo vrsili realizaciju multiplekserima na osnovu BDD-ova.



Sinteza sa LUT FPGA

Podrazumeva se da ako FPGA ima k ulaza i m izlaza moze da realizuje m funkcija sa k ulaznih promenljivih.

Prva stvar kod ubacivanja odredjene funkcije u LUT FPGA sa k ulaza /look up tabeli je dekomponovanje funkcije na osnovne podfunkcije sa ne vise od k promenljivih.Obicno je pogodno da se uzmu funkcije sa 2 promenljive kako bi se postigla finija granulacija.

Evo kako se realizuje KNF na 2 razlicita nacina:

Data funkcija od n promenljivih se prezentuje preko KNF koji ima r proizvoda Pi sa qi=|Pi| literala.Zadatak je dodeliti f look up tabeli FPGA sa  ulaza po tabeli koriscenjem minimalnog broja look up tabela(LUT-ova).

Testiranje logickih mreza

Iako su danasnja kola izuzetno pouzdana u pogledu verovatnoce da pojedini izlaz bude nekorektan zbog specificnosti situacije,testiranje logickih mreza je jedan od najvecih problema u ovom podrucju.Greske mogu da se jave zbog vise razloga, npr. kratkospajanje, kidanje veza, itd.Greske mogu da se podele na:

* Lake greske(Soft errors) koje su posledica okoline, kao na primer elektricne smetnje iz gradske mreze,kosmicki zraci koji mogu da naelektrisu jone na cipu(jesteeeeeeeeeeeee :D )
* Teske greske(pogadjate Hard errors) koje su stalne i koje se mogu desiti usled mehanickih potresa, korozije, menjanja osobina metala itd.

Lake greske je tesko otkriti, i one se obradjuju softverom.Testiranje logickih mreza se uglavnom odnosi na hard greske.

Na logickom nivou mi posmatramo uticaj greske na ponasanje celokupnog kola.Postoje razliciti modeli gresaka, ipak “zaglavljivanje na 0/1 “ je model greske koji se najvise srece.Dobro odredjeni testovi za ovakve greske mogu da otkriju i gomilu drugih. Pri resavanju zaglavljivanja moramo da uzmemo u obzir sve slucajeve.I kada su ulazi podeseni na 0/1 i kada su izlazi podeseni na 0/1.Ako jedna linija povezuje vise kola, broj gresaka je veci od 2.

Broj gresaka je 2c, gde je c broj kola u DUT(Device Under Testing).Ipak, broj gresaka koje se razmatraju moze se drasticno smanjiti uocavanjem “ekvivalentnih gresaka”.A evo i sta su te greske:

Ako imamo 2 greske e1 i e2 na izlazu logicke mreze N, i neka su fe1 i fe2 izazi mreze N u prisustvu gresaka e1 i e2 respektivno.Greske e1 i e2 su evivalentne akko vazi fe1=fe2.

* proces smanjivanja ekvivalentnih gresaka – “collapse of errors”
* pojavljivanje iste greske na vise mesta – “multiple errors”

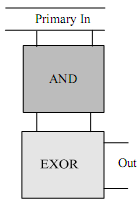
Ako je i u prisustvu greske izlaz mreze N isti kao da greska nije prisutna f=fe1, onda se takva greska naziva “undetectable error”.Cesto se naziva i redundantna greska zato sto je obicno posledica redundantnosti u kolu.

Test za detekciju gresaka u datoj mrezi se sastoji od dodeljivanja takvih vrednosti na ulazu gde ce izlaz odstupati od pravilnih vrednosti( kombinacije ulaza za koje se detektuje greska na izlazu).”Kompletan test” se sastoji od sekvenci koje bi otkrile sve “pretpostavljene greske” na mrezi.Jasno je da je najkompletniji test ako bi smo isprobali sve kombinacije,sto je neprakticno za veliki broj ulaza.Minimalan test je tesko odrediti.Postoji razlicite metode za automatsko generisanje testova koje se mogu podeliti na:

1. Algebarske koje se sastoje u manilupaciji sa algebarskim izrazima opisujuci funciju koju realizuje mreza
2. Topoloske koje se odnose na na rad sa kolima i njihovim povezivanjem, dakle topologijom DUT-a.

Lako testabilne mreze

Pouzdanost sistema se moze postici uvodjenjem redundantnosti koja moze biti dinamicka i staticka.Staticka podrazumeva da sistem sadrzi redundantne delove koji u slucaju greske koriguju izlaz.U slucaju dinamicke sistem se testira odabranom frekvencijom i u vremenskim intervalima koji se tako postavljaju da se greska moze uociti, i modul zameniti odgovarajucim u kratkom roku.Zato za takve sisteme potrebno je da budu projektovani tako da se mogu lako testirati.Zato su i dobili naziv lako testabilne mreze.Lako testabilna mreza treba da ispuni sledece zahteve:

1. Sto manji broj testova
2. Mreza sme biti staticki redundantna, posto se moze desiti da se greska ne otkrije primenom test sekvenci
3. Proces realizacije test mreze treba da bude prost i po mogucstvu deo procesa projektovanja
4. Da se test sekvence lako generisu, a rezultati dobijeni njihovom primenom lako obradjuju
5. Pozeljno je da test sa odredjenom preciznoscu ukazuje na lokaciju greske

Lako testabilne mreze mogu da se **realizuju** pomocu izraza u pozitivnom Davio razvoju(pozitivan RM).Podsetimo se da se ovaj razvoj sastoji od primene AND i OR logicih kola.

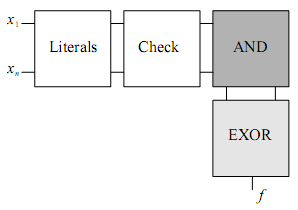
Podrazumeva se da:

1. Moze da nastane greska na izlazima AND kola usled zaglavljivanja
2. Ako se greska desi kod EXOR kola onda ovo kolo moze da realizuje bilo koju od preostalih 15 funkcija sa 2 promenljive

Realizacija sa RM razvojima fixne polarnosti

Mane pozitivnog RM-a:

1. Serijsko povezivanje EXOR kola moze da uzrokuje veliko kasnjenje
2. Za mnoge funkcije PRM razvoj moze da stvori veliki broj proizvoda i veliki broj AND kola
3. Visestuke greske se ne mogu detektovati



RM razvoj fixne polarnosti zahteva manji broj AND kola, i zato se bolji od PRM-a.

Posto se kod RM-a fixne polarnosti koriste i pozitivne i negativne kombinacije promenljivih,zvacemo ih literali.Deo gde se generisu literali se sastoji od 2 dvoulazna EXOR kola ciji su ulazi xi i c, gde je c kontrolni ulaz za sva ova EXOR kola.Deo nazvan Check se sastoji od dodatnih 2 AND ili OR kola sa dostupnim izlazima oznacenim ao AND(A), AND(B),OR(A) ili OR(B) respektivno.Kola oznacena sa A se koriste za testiranje pozitivnih literala, koji su njihovi ulazi.B kola testiraju negativne literale.AND deo se koristi za generisanje proizvoda kod fix RM razvoja.Poboljsanje je u smanjenom broju proizvoda, i u mestno serijske veze EXOR kola, EXOR mreza moze da se realizuje kao stablo, sto smanjuje kasnjenje i cini mrezu mnogo brzom.

**Sekvencijalne mreze**

Izlaz sekvencijalne mreze zavisi od trenutnog stanja i ulaza,tj. od istorije mreze.Od kombinacionih mreza se razlikuju po tome sto sadrze memorijske elemente koji sluze za snimanje prethodnog stanja sistema.Ponasanje sekvencijalne mreze se opisuje matematickim modelom “sekvencijalnom masinom”, “masinom konacnih stanja” ili “konacnim automatom”(sve ovo su sinonimi), a moze se predstaviti tabelom stanja ili dijagramom stanja.

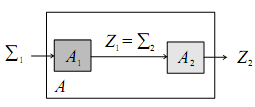
Ponasanje sekvencijalnih masina se odredjuje dvema funkcijama: funkcijom stanja koja odredjuje naredno stanje sekvencijalne masine, i funkcijom izlaza.

Modeli sekvencijalnih mreza su Murov i Milijev automat.

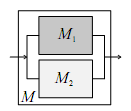
Kodiranje stanja podrazumeva biranje kodnog sistema i upotrebljavanje za kodiranje razlicitih stanja, pri cemu razliciti kodovi mogu da daju vise ili manje efikasnu realizaciju.

Minimizacija mreza podrazumeva koriscenje manjeg broja stanja.

Postoje i nepotpuno definisane masine.Na primer kada se masina nalazi u odredjenom stanju, onda neki ulaz(nebitno, neka promena na ulazu) ne moze da se aktivira, jer njegovim aktiviranjem nije specificirano sta ce biti na izlazu, niti u koje ce naredno stanje automat da ide.

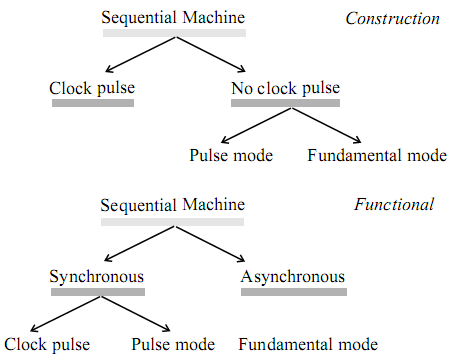
Dekompozicija se obicno vrsi ako automat ima mnogo stanja.Znamo da se velike mreze dele na module kada je upravljanje njima komplikovano, i onda ih delimo da manje module koje posebno odrzavamo i optimizujemo.Postoje serijska i paralelna dekompozicija.Kod serijske dekompozicije broj ulaza druge masine je isti kao broj izlaza prve.

Sekvencijalna masina M1-M2 koja je serijska konecija M1 i M2 se serijska dekompozicija date sekvencijalne masine M akko M1-M2 realizuje M.



Takodje, sekvencijalna masina M1|M2 je paralelna dekompozicija sekvencijalne masine M akko M1|M2 realizuje M .Dekompozicija je netrivijalna ako je totalni broj stanja u M1 i M2 manji od broja stanja u M

U odnosu na nacin rada, sekvencijalne mreze se mogu podeliti ovako:

Ako se prelazi kod sekvencijalnih mreza odvijaju u ritmu clock impulsa, onda su takve mreze taktovane.Sekvencijalne masine naravno mogu raditi bez takve eksterne sinhronizacije.Dalje sek. masine mogu raditi u pulse mode ili fundamental mode.Sa funkcionalne tacke gledista, obe, i klokovane i pulse mode mreze su sinhrone sekvencijalne mreze.One koje rade u fundamental modu su asinhrone.