

# Pouzdanost računalnih sustava

---

Predavanje 1:      Uvod

Prof.dr.sc. Vlado Sruk



Sveučilište u Zagrebu  
**Fakultet elektrotehnike i računarstva**  
Zavod za elektroniku, mikroel., računalne i inteligentne sustave



# Sadržaj

---

- diskusija o temi predmeta
- o predmetu
- motivacija
- uvod
- terminologija
- osnovni principi

# Opis predmeta

---

- Sveprisutnost računalnih sustava i oslanjanje na njihove usluge čini ih nezamjenjivim dijelom današnjeg života. Stoga je izuzetno važno smanjiti učestalost kvarova takvih sustava i njima uzrokovanih šteta na prihvatljivu razinu.
  - U okviru kolegija proučavaju se koncepti, metode i tehnike oblikovanja, kao i implementacija, analiza i vrednovanje pouzdanosti, raspoloživosti i neosjetljivosti na pogreške sklopolja i programa s ciljem razumijevanja uzroka i posebnosti zatajenja računalnih sustava.
  - Naglasak je na proučavanju pouzdanosti i neosjetljivosti na pogreške računalnih sustava, te razvijanje sposobnosti za primjenu osnovnih principa za izgradnju stvarnih sustava.
-

# I ciklus

Tj.	Datum	Opis
I.	23. veljače	<b>Uvod. Osnovni principi, primjeri i terminologija. Definicije oslonljivosti, pouzdanosti i raspoloživosti. Kvarovi, pogreške i zatajenja. (<a href="#">prs 01</a>, <a href="#">prs 01 3</a>)</b>
II.	2. ožujka	<b>Modeli kvarova i pogrešaka. Procesi zatajenja, obrada kvarova.</b>
III.	9. ožujka	<b>Testiranje digitalnih sustava. Simulacije. Projektiranje s naglaskom na testabilnost. Ugrađeno testiranje, samotestiranje.</b>
IV.	16. ožujka	<b>Teorija pouzdanosti. Metode vrednovanja pouzdanosti. Funkcija intenziteta kvara, srednje vrijeme do kvara, srednje vrijeme do popravka. Kombinatorni modeli. Blok dijagram pouzdanosti. Monte Carlo simulacija.</b>
V.	23. ožujka	<b>Modeliranje pouzdanosti, raspoloživosti i sigurnosti Markovljevim modelima. Načini zatajenja i analiza efekata.</b>

# II ciklus

VIII.	13. travnja*	<b>Tehnike poboljšanja pouzdanosti. Tehnike oblikovanja sustava neosjetljivih na pogreške. Popravljivi sustavi. Sustavi s pričuvom.</b>
IX.	20. travnja	<b>Vremenska redundancija. Otkrivanje i toleracija prijelaznih i trajnih kvarova. Informacijska redundancija.</b>
X.	27. travnja	<b>Kodovi za otkrivanje i ispravak pogrešaka. Primjena kodova.</b>
XI.	4. svibnja	<b>Redundancija programa. Modeli pogrešaka programa. N-verzijsko programiranje, blokovi oporavka.</b>

# III ciklus

XIV.	25. svibnja	<b>Modeli i procjena programske pouzdanosti. Utjecaj na ponašanje sustava.</b>
XV.	1. lipnja	<b>Neosjetljivost na pogreške u raspodijeljenim sustavima. Bizantinski model zatajenja.</b>
XVI.	8. lipnja	<b>Visoko raspoloživi računalni sustavi i usluge. Modeli održavanja.</b>
XVII.	15. lipnja	<b>Eksperimentalna analiza pouzdanosti i raspoloživosti sustava. Metodologije oblikovanja sustava. Optimalna uporaba resursa.</b>

# Motivacija

---

- postići visoku pouzdanost računalnog sustava
- kako?
  - projektiranjem
  - testiranjem i pronalaženjem potencijalnih kvarova (engl. faults)
  - dopuštanje kvarova uporabom redundancije /zalihosti/
- terminologija, definicije i modeli kvarova
- principi projektiranja sustava neosjetljivih na pogreške
- procjena, verifikacija, provjera valjanosti

# Motivacija

---

- jednostavni sustavi
  - radi / ne radi
- distribuirani sustavi – više neovisnih čvorova
  - mogućnost djelomičnih kvarova
  - kakve su mogućnosti oporavka od takvih kvarova?
  - značajno jer vjerojatnost kvara raste s brojem neovisnih dijelova
- korisnici

# Razvoj tehnika

---

- <1950
  - BRC : ponovljeno izračunavanje
  - UNIVAC : uvođenje pariteta, 2 ALU
  - uvođenje ad-hoc tehnika
- 1950
  - teoretski radovi na redundantnosti i kodiranju
    - Moore, Shannon, Hamming, von Neumann
  - uvođenje teorije vjerojatnosti
- 1960
  - ugradnja neosjetljivosti na pogreške u sustave
  - IBM 360, svemirski program
- 1970
  - oslonljivi sustavi
    - (ex) Tandem , Stratus
  - utjecaj VLSI tehnologije
- 1980
  - objedinjavane neosjetljivosti na pogreške i performanse
  - primjena u paralelnom i raspodijeljenom računarstvu
- 1990
  - visokopouzdani sustavi za rad u stvarnom vremenu
  - neosjetljivost na pogreške svojstvena pojedinim aplikacijama
- 2000
  - primjena u mrežnim sustavima
  - heterogeni sustavi

# Rast utjecaja neosjetljivosti na pogreške

---

- povećavanje brzine digitalnih sustava ⇒ stroža vremenska ograničenja
- zahtjevne okoline ⇒ sve šire područje primjena
- velike cijene popravaka ⇒ ljudski rad i ispadи sustava
- veliki sustavi ⇒ veći broj dijelova povećava šanse za pojavu kvara

# Značaj sustava neosjetljivih na pogreške

---

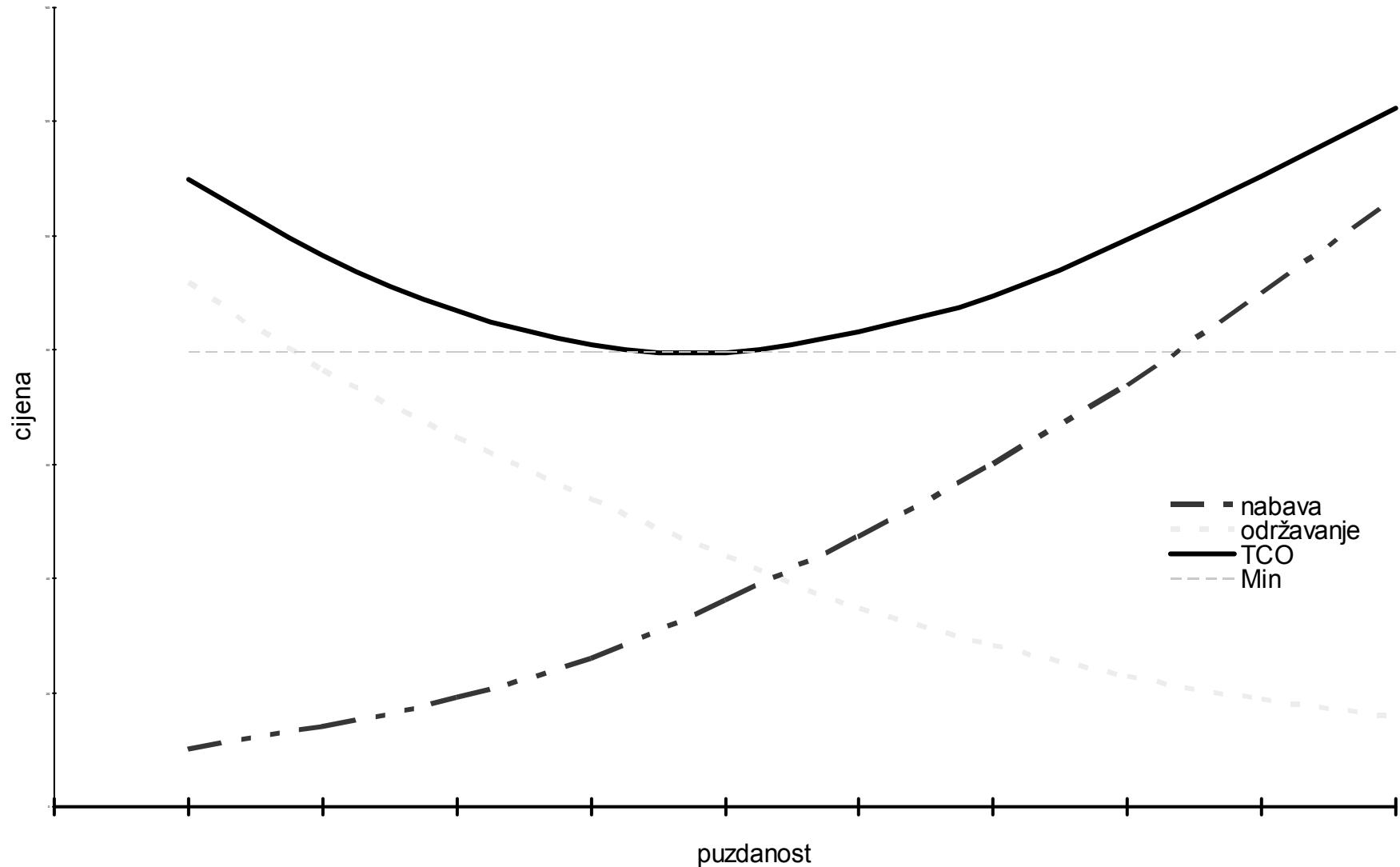
- Sustavi opće namjene
  - osobna računala: memorije s paritetom
  - radne stanice: detekcija greške (HW), povremene korektivne akcije (SW), ECC (HW), bilješke (SW)
- pouzdani sustavi
  - telefonske sustav
  - bankarski sustav
  - burze
  - zabava
- kritični i životno kritični sustavi
  - svemirske letjelice sa i bez ljudske posade
  - upravljanje letjelicama
  - nuklearni reaktori
  - sustavi za održavanje života
- pouzdani -> kritični sustavi
  - služba hitne pomoći
  - upravljački sustav semafora
  - ABS, ubrizgavanje

# Ukupna cijena posjedovanja (Total Cost of Ownership )

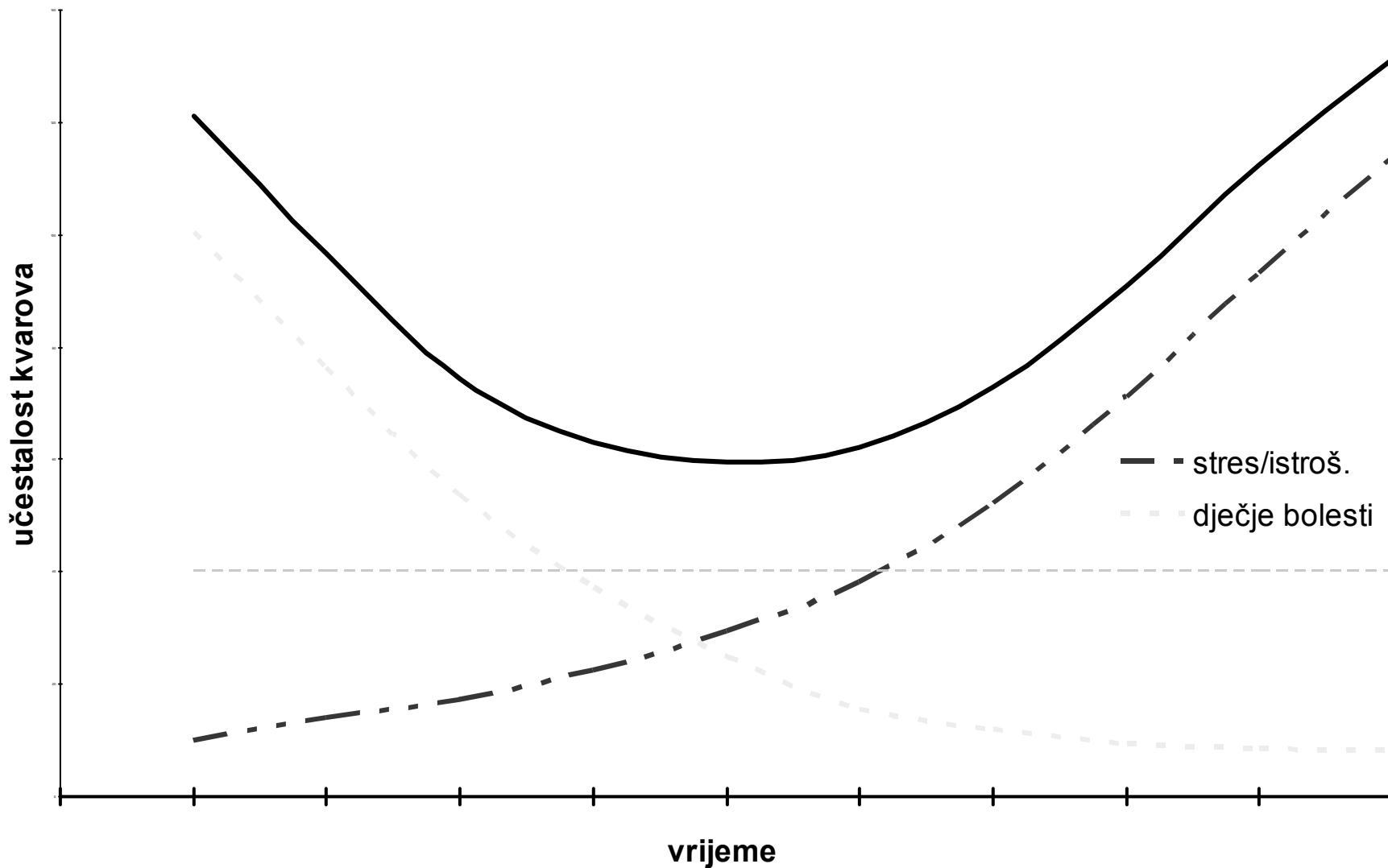
---

- mjeri se u trogodišnjem periodu
- cijena sklopoljia i programske podrške
  - uključuje početnu cijenu i cijenu nadogradnje
- cijena održavanja i podrške
- cijena posjedovanja (engl. Opportunity Costs)
  - troškovi korisničkog samoodržavanja i podešavanja sustava
    - npr. Korisnik prosječno tjedno troši 3,2 sata na rad sa sustavom  $3,2 \times \text{bruto cijena sata} \times 52 \text{ tjedna} \times 3 \text{ godine}$
    - $499,2 \text{ sati} / 160 = 3,12 \text{ mjeseci}$

# TCO/pouzdanost



# faktori koju utječu na kvarove

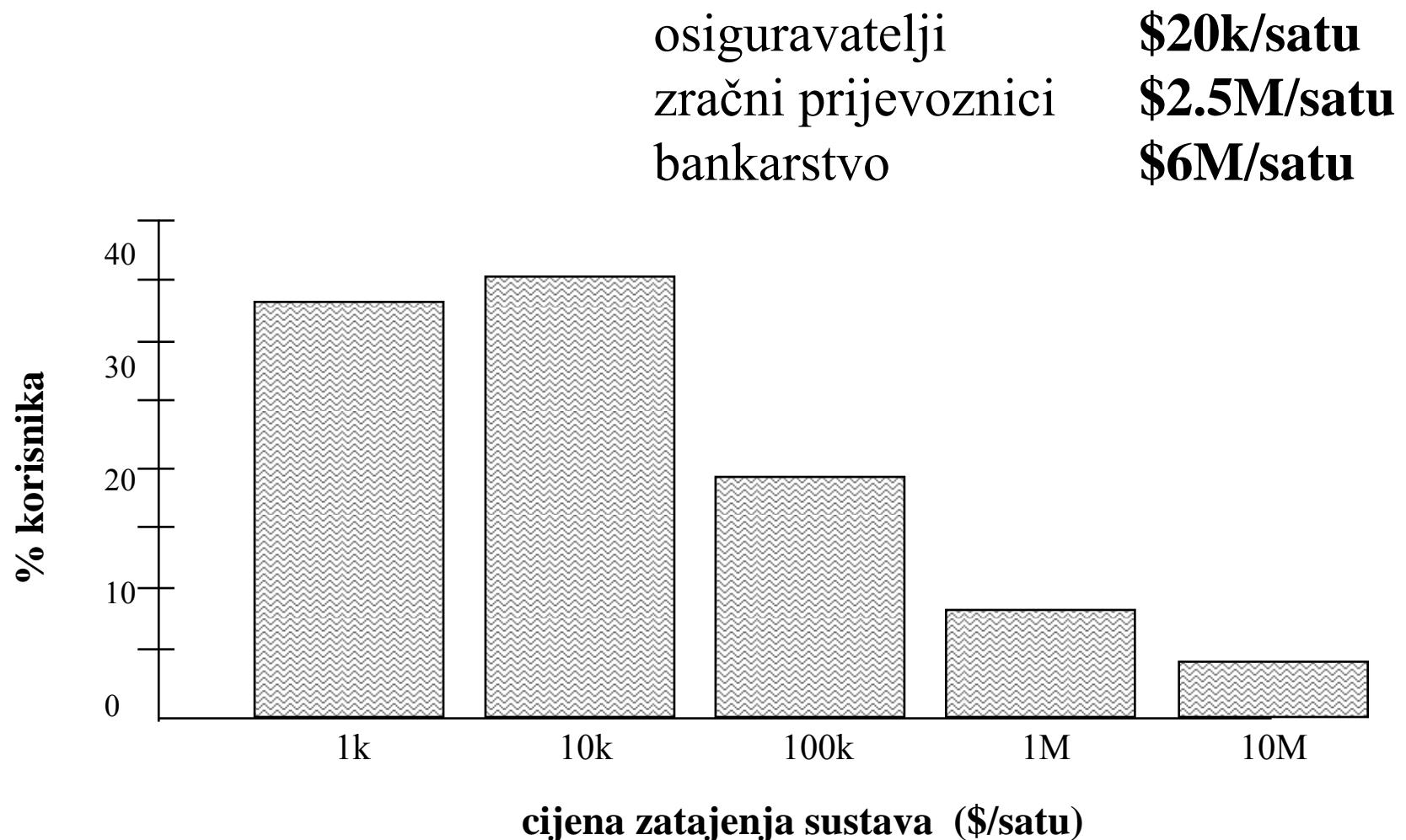


# Primjeri raspoložljivosti

raspoložljivost	9	zatajenje	primjer
90%	1	>1 mjesec	Neodržavano osobno računalo
99%	2	~4 dana	Održavano osobno računalo
99.9%	3	~9 sati	Nakupina računala (Cluster)
99.99%	4	~1 sat	Mainframe
99.999%	5	~5 minuta	Ugrađeno računalo PC platforma
99.9999%	6	~30 s	Ugrađeno računalo namjenska platforma
99.99999%	7	~3 s	Ugrađeno računalo namjenska platforma

# cijena zatajenja sustava

---



# Neosjetljivost na pogreške

---

## ■ Što je neosjetljivost na pogreške?

Sustav je *neosjetljiv na pogreške* (engl. fault-tolerant system) *ako obavlja svoje zadaće usprkos zatajenju* (engl. failure) pojedinih komponenti

## ■ ključni termini

- kvar-pogreška-zatajenje
- fault-error-failure
- performanse - raspoložljivost - pouzdanost

# Oslonljivost (Dependability)

---

- osigurava veliku vjerojatnost ponašanja sustava prema zadanim specifikacijama
- neophodna sveobuhvatna specifikacija sustava
  - funkcionalnost
  - uvjeti okoline
  - uključuje sadržajno ovisan pojam “*velike vjerojatnosti*”

# Oslonljivost ...

---

- 1980 IFIP Working Group 10.4
  - International Federation for Information Processing  
<http://www.ifip.or.at/>
- IFIP Technical Committee TC-10, "Computer Systems Technology"
  - 1991 "Dependability: Basic Concepts and Terminology"
- definicija
  - oslonljivost se definira kao pouzdanost sustava takva da se očekivanja mogu opravdati njegovim obavljanjem zadaća
  - obavljanje zadaća sustava predstavlja njegovo ponašanje viđeno od strane korisnika (čovjek ili sustav)

# Oslonljivost

---

- postoji li sustavan put za postizanje opravdanih očekivanja?
  - nema recepta ► umijeće
  - #1: poznavanje prijetnji
  - #2: umijeće sredstava za postizanje oslonljivosti
  - #3: osmisliti načine za spec. potrebne oslonljivosti
  - #4: mjerenja postignutih rezultata

# Oslonljivost Dependability

## Sustav

svojstva

pouzdanost	Reliability
raspoložljivost	Availability
sigurnost	Safety
tajnost	Confidentiality
cjelovitost	Integrity
izvršljivost	Performability
popravljivost	Maintainability
ispitljivost	Testability

sredstva

prevencija	Fault Prevention
uklanjanje	Fault Removal
neosjetljivost	Fault Tolerance
predviđanje	Fault Forecasting

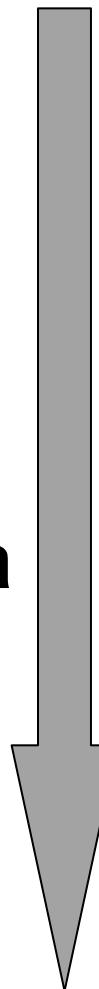
prijetnje

kvar	Faults
greška	Errors
zatajenje	Failures

# kvar, pogreška, zatajenje

---

- **kvar** - proglašeni uzročnik kvara
  - može biti prikriven neko vrijeme
- **pogreška** - dio stanja sustava odgovorno za stvaranje zastoja
  - manifestacija kvara
- **zatajenje** - sustav ne zadovoljava specifikacije
  - problem vidljiv izvan sustava



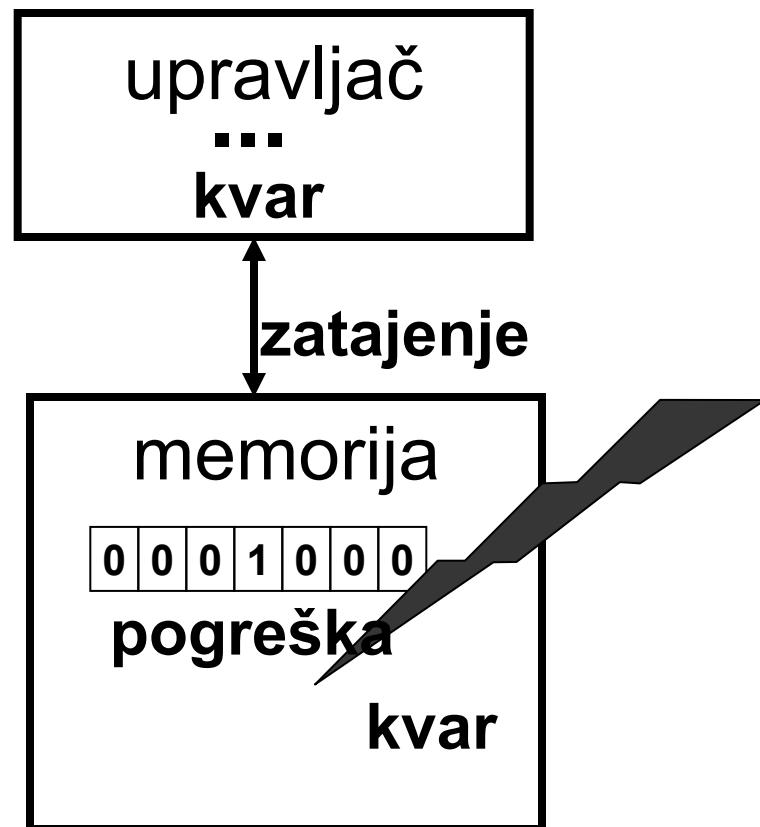
fizikalni svijet

informacija

vanjska  
pojavnost

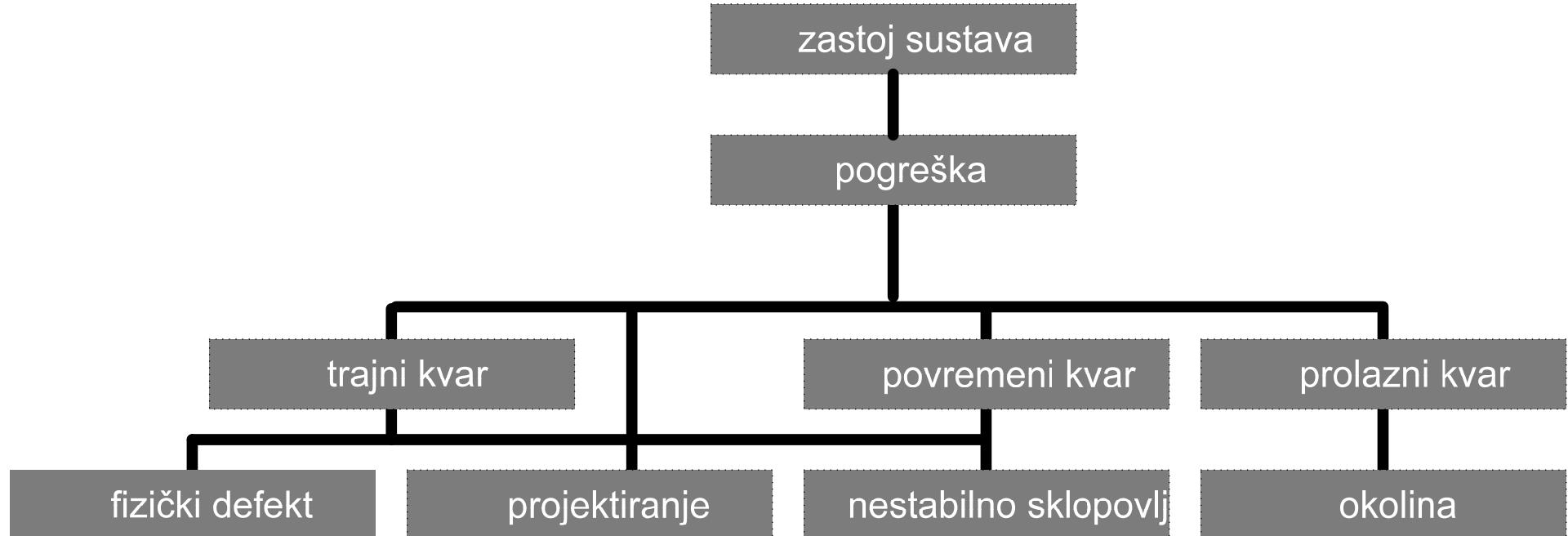
# primjer

---



# uzročno posljedična veza

---



# Tipovi kvarova

---

- fizički kvar
  - defekt sklopoljja
  - vanjske smetnje
- kvar uzrokovani ljudima
  - slučajan kvar
    - kvar operatera
    - programski kvar projektiranja
    - sklopolovski kvar projektiranja
  - namjeran kvar
    - npr. nedopušteni ulaz u sustav

# Klasifikacija kvarova

priroda		porijeklo						trajanje		oznaka
		uzrok pojave		granice sustava		vrijeme stvaranja				
Slučajni	namjerni	fizičke	ljudske	Unutar	vanjski	projektiranje	rad	postojan	prolazan	
♦		♦		♦			♦	♦		<i>fizički kvar</i>
♦		♦			♦		♦	♦		
♦		♦			♦		♦		♦	<i>prolazan kvar</i>
♦		♦		♦			♦		♦	<i>povremen kvar</i>
♦			♦	♦		♦			♦	
♦			♦	♦		♦		♦		<i>kvar u projekt.</i>
♦			♦		♦		♦		♦	<i>kvar interakcije</i>
♦		♦	♦	♦	♦	♦	♦	♦	♦	<i>zlonamjernost</i>
♦		♦	♦	♦	♦	♦	♦		♦	
♦		♦	♦	♦	♦	♦	♦	♦	♦	<i>nedopušteni ulazak</i>

# Ostvarenje oslonljivosti

---

- bez intervencije lanac zatajenja sustava se širi
- prevencija kvara – sprječava nastanje uzroka pogreške
  - kvalitetne komponente
  - komponente s ugrađenom zalihošću
  - stroge tehnike projektiranja
- uklanjanje kvara- otkrivanje i uklanjanje kvara prije nastanka pogreške
  - pronalaženje loših komponenti, programske pogreške
- predviđanje kvara – procjena vjerojatnosti javljanja ili zadržavanja kvara
- izbjegavanje kvara: prevencija kvara + uklanjanje kvara

...

---

- uvijek nije moguće ostvariti izbjegavanje svih kvarova ⇒ tolerancija
- Sustav neosjetljiv na pogreške (engl. Fault-Tolerant System)
  - sustav koji ostvaruje zadaće bez obzira na pojavu jednog ili više kvara
  - djeluje u fazi stvaranja kvara
- Otkrivanje pogreške (engl. Error detection)
  - pronalaženje pogreške
- obrada pogreške (engl. Error processing)
  - mehanizam uklanjanja pogreške (prije pojave zastoja:))
  - oporavak
  - kompenzacija

# Obilježja

---

## ■ pouzdanost $R(t)$

- uvjetna vjerojatnost da sustav radi ispravno (unutar zadane specifikacije) unutar zadanog vremenskog intervala  $[0,t]$  i u zadanim radnim uvjetima
  - MTBF - Mean Time Between Failures
  - funkcije razdiobe vjerojatnosti (npr., bathtub)

## ■ popravljivost $M(t)$

- vjerojatnost da će sustav biti popravljen unutar vremenskog intervala  $[0,t]$ 
  - MTTR – Srednje vrijeme dio popravka (engl. Mean Time To Repair)

# ... obilježja

---

- raspoložljivost  $A(t)$ 
  - vjerojatnost da će sustav raditi ispravno u trenutku t
    - popravljeni sustav
    - $EA = MTTF / (MTTF + MTTR)$
- izvršljivost  $P(l,t)$ 
  - vjerojatnost da će svojstva sustava biti iznad razine l u trenutku t
  - kvalitativno određuje degradaciju sustava
- sigurnost  $S(t)$ 
  - vjerojatnost da će sustav raditi ispravno ili da će prestati raditi bez ugrožavanja drugih sustava ili ljudi u vremenu  $[0, t]$
  - MTTCF - Mean Time To Catastrophic Failure

# primjena

---

- dugotrajni sustavi
  - Long Life Applications
- kritično izračunavanje
  - Critical Computation
- odgoda održavanja
  - Maintenance Postponent
- visokoraspoloživi sustavi
  - High Availability Systems

# Pitanja...

# Pouzdanost računalnih sustava

---

**Predavanje 2:**      **Uzroci i modeliranje kvarova**

**Prof.dr.sc. Vlado Sruk**



**Sveučilište u Zagrebu**  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroel., računalne i inteligentne sustave*



# Sadržaj

---

- Manifestacije kvarova
- Modeli kvarova
- Modeli pogrešaka
- Simulacija kvara

# Zašto modelirati kvarove?

---

- svojstva kvara: prolazna, neodređene vrijednosti  
⇒ kako projektirati testiranje ili simulaciju
- testiranja U/I funkcija su nedovoljna za proizvodnju
  - funkcionalnost u odnosu na testiranje komponenti i međusobnih spojeva
- stvarni nedostaci su brojni i teško ih je analizirati
- što je model?
  - apstrakcija koja obuhvaća ponašanje originalnog sustava
    - jednostavnost
    - vodi do ispravnih zaključaka o ponašanju sustava

# Svojstva modela kvara

---

- omogućava nedestruktivne metode ispitivanja
- upravljiv prostor analize (mogućnost provjere ekvivalentnosti i smanjenja prostora)
- omogućava predstavljanje fizičkih pojava
- služi za pronalaženje ciljeva testiranja
- nije 100% pouzdan u svim slučajevima
- omogućava analizu
- djelotvornost mjerljiva pokusima

# Stvarni nedostaci čipova

---

## ■ nedostaci proizvodnje

- manjak okvira kontakata
- parazitski tranzistori
- lom oksida
- ...

## ■ nedostaci materijala

- nedostaci
- nečistoća materijala
- ...

## ■ Vremenski ovisna zatajenja

- probaj dielektrika
- elektromigracija
- ...

## ■ Zatajenja pakiranja

- degradacija kontakata
- nehermetično pakiranje
- ...

# Nedostaci štampanih pločica

---

klase nedostataka	pojavljivanje (%)
kratki spoj	51
prekid	1
nedostatak komponenti	6
pogrešne komponente	13
zamijenjene komponente	6
savijeni vodovi	8
analogne specifikacije	5
digitalna logika	5
performanse (vrijeme)	5

# Nivoi modeliranja

nivo	opis sklopa	vrijed. signala	vrijeme	primjena
funkcija, ponašanje, RTL	HDL	0,1	granice takta	verifikacija arhitekture i funkc
logika	mreža log. sklopova, bistabila, tranz.	0,1,X,Z	kašnjena	test. i log. verifikacija
sklopka	vel. tranz., povezivanje, kapac.	0,1,X	0 kašnjena	log. verifikacija
vremenski odnosi	tehnol. tranz., povezivanje, kapac.	analog. vrijed	fina rez.	vremenska verifikacija
sklop	tehn. podaci, pov. aktivnih i pasivnih komp.	analog. vrijed., struja	kontin. vrijeme	dig. vrem., analogna verifikacija

# Manifestacije kvarova

---

## ■ kako se kvar manifestira u pogrešku?

- model kvara na nivou sklopa (Circuit-Level Fault Model)
- model kvara na nivou logike (Logical-Level Fault Model)
- model kvara na nivou sklopke (Switching-Level Fault Mode)
- model kvara na nivou sustava (System-Level Fault Model)

# Model kvara na nivou sklopa

---

- najniži u hijerarhiji pogrešaka i kvarova
  - uzrok im leži u defektu u procesu proizvodnje, ili stresu za vrijeme normalnog rada

# Model kvara na nivou logike

---

- Za analiziranje ovih kvarova koristimo kvarove na nivou sklopke koji generiraju ovaj tip kvara
- ŠIRINA logičkog kvara
  - *neovisan* događaj (lokalan, šteti samo jednoj varijabli)
  - *ovisan* događaj (uzrok mu je kvar na zajedničkoj komponenti ili se javlja zbog gustoće elemenata)
- VRIJEDNOST logičkog kvara
  - *određena* (zaglavio se u 0 ili 1)
  - *neodređena* (vrijednost skače između 0 i 1)

# Model kvara na nivou sklopke

---

- MOS tranzistori se ponašaju kao bipolarne sklopke (puštaju da struja teče u oba pravca), dok TTL tranzistori rade kao sklopke koje propuštaju struju samo u jednom pravcu
- model kvara na nivou vrata (gate-level fault model)
  - apstrakcija mehanizama fizičkih nedostataka
  - pričvršćen-na (stuck-at) - stalno u 0 ili 1
  - premošćujući (bridging) - 2 ili više signalnih linija je kratko spojeno, tvoreći pri tome logičku familiju i dodatnu *spojeni-I* ili *spojeni-ILI* funkciju
  - kratki spoj ili otvoreni krug (postoji višak ili manjak elemenata)
  - lavina grešaka (zbog 1 greške više signalnih linija je u pogrešci, npr. imaju istu funkciju)
- model kvara sklopke (switch-level fault model)
  - neophodni simulatori nižeg nivoa
  - ograničeno generiranje i procjena pokrivenosti na manje sustave

# Model kvara na nivou sustava

---

- manifestacije povremenih i prolaznih kvarova, te pogrešnog projektiranja sklopoljja i programa je puno teže odrediti no manifestacije trajnih kvarova, sami trajni kvarovi su često praćeni različitim povremenim kvarovima, dok su prolazni kvarovi često uzrokovani različitim kombinacijama lokalnih fenomena.
- kvar pri projektiranju
  - može biti lokaliziran na jednu komponentu
  - rezultat složenosti
  - nema dovoljno mogućnosti za praćenje širenja kvara od izvora do manifestacije
  - problem integracije

# primjer I

---

	prije integracije (%)	poslije integracije(%)
logičke greške	<b>52</b>	<b>21</b>
mogućnost testiranja	<b>17</b>	<b>12</b>
kompatibilnost	<b>15</b>	<b>4</b>
povreda tehnoloških pravila specifikacije	<b>6</b>	<b>25</b>
greške takta	<b>4</b>	<b>19</b>
upravljanje pogreškom	-----	<b>2</b>
šum	-----	<b>10</b>
pogreška projektiranja	-----	<b>4</b>
performanse	-----	<b>2</b>
		<b>1</b>

# Uobičajeni modeli kvarova

---

- pričvršćen-na (engl. stuck-at fault)
- otvoreni (engl. stuck-open fault)
- premošćujući (engl. bridging fault)
- kašnjenje (engl. delay fault)

# Model kvara pričvršćen—na

---

- efikasan i jednostavan
- s-a-0 ili s-a-1
- pretpostavke
  - kvar rezultira ponašanjem sklopa kao da je jedan od ulaza ili izlaza pričvršćen na 0 ili 1
  - osnovna funkcionalnost sklopa nije narušena kvarom
  - kvar je stalan
  - uobičajeno se pretpostavlja da u sklopu postoji samo jedan kvar takve vrste!!!?

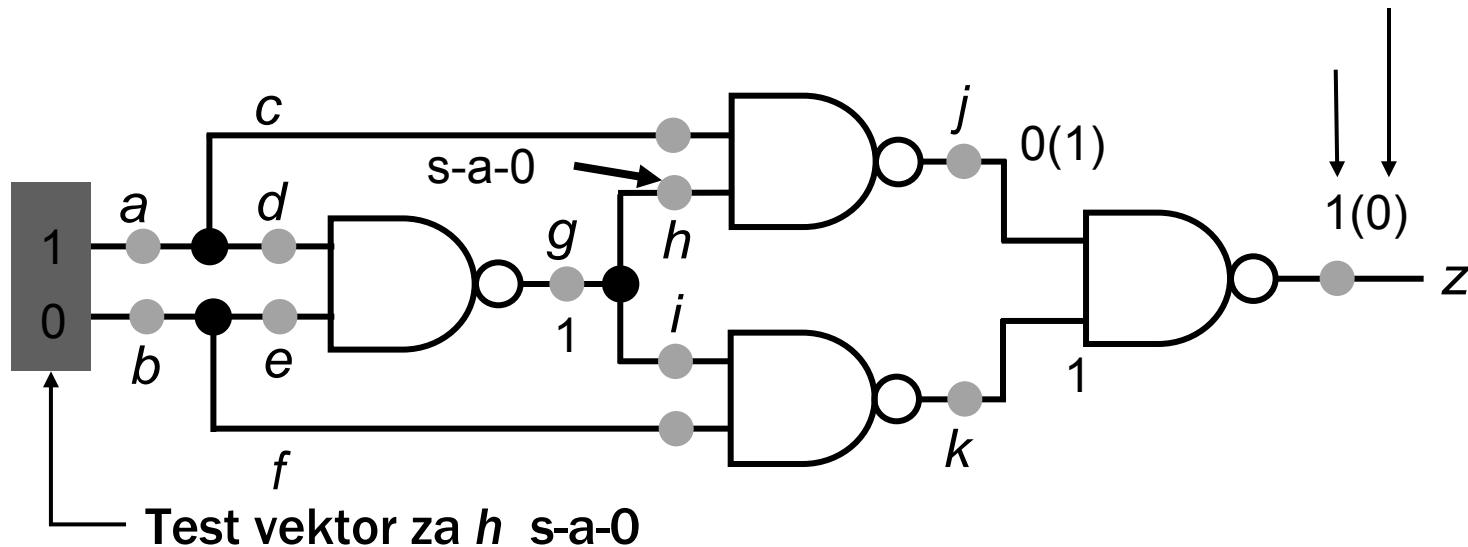
- 
- svaki čvor u sustavu može imati tri vrijednosti:
    - ispravan
    - s-a-0
    - s-a-1
  - za sklop s 100 čvorova postoji  $3^{100}$  različitih stanja za potpunu provjeru
  - uz trajanje provjere jednog slučaja od  $1\mu\text{s} \Rightarrow 1.6 \cdot 10^{34}$  godina
  - Faktor obuhvata (engl. Coverage parameter)
    - kvarovi koji su obuhvaćeni izabranim modelom kvarova

# Jednostruki kvar pričvršćen na

## ■ svojstva

- jedan kvar
- linija u kvaru trajno pričvršćena na 0 ili 1
- kvar može biti na ulazu ili izlazu

## ■ Primjer XILI sklop



---

## ■ Prednosti

- Primjenjiv na nivou logike ili modula
- Dobro razvijeni efikasni algoritmi automatiziranog generiranja uzoraka i simulacije (ATPG)
- Pokazano da pokriva 90% grešaka u CMOS-u
  - Više od 95% višestrukih grešaka
- Ostali modeli mogu se svesti na niz s-a kvarova

## ■ Nedostatak

- Ne pokriva potpuno sve defekte CMOS-a

# Test vektor

---

- Specifikacija ponašanja razmatranog sklopa
  - Vrijednosti primjenjene na ulaze sklopa
  - Očekivani izlazi
  - Npr. 1100101010/0 (destoulazni I sklop)
- Otkriva postojanje kvara ako je izlaz različit od očekivanog

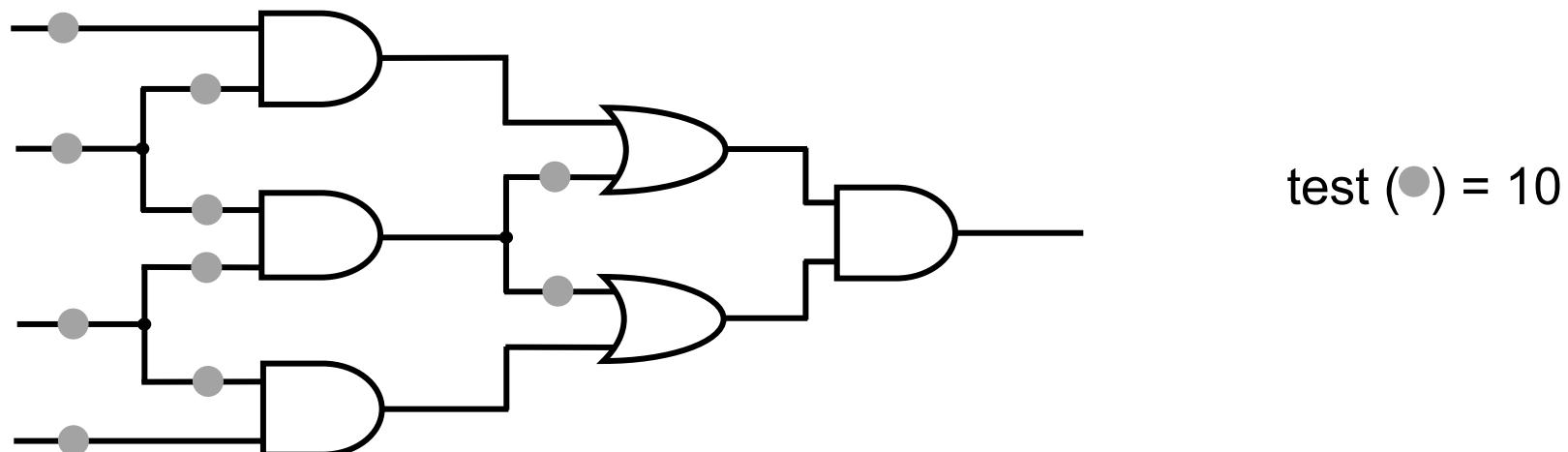
# Uvjeti otkrivanja kvara

---

- Aktiviranje kvara
  - Primjena odgovarajućeg ulaza
- Širenje efekta kvara do točke promatranja (izlaz)
  - Engl. “path sensitization”
  - Kontrola ulaza u logičke blokove
- Kvar je moguće otkriti ako postoji test koji ga otkriva

# Testne točke (checkpoints)

- ulazne i izlazne grane kombinacijskog sklopa
- skup testova koji otkrivaju sve jednostrukе (višestrukе) kvarove pričvršćen-na na svim testnim točkama kombinacijskog sklopa također otkriva i kvarove pričvršćen-na u sklopu

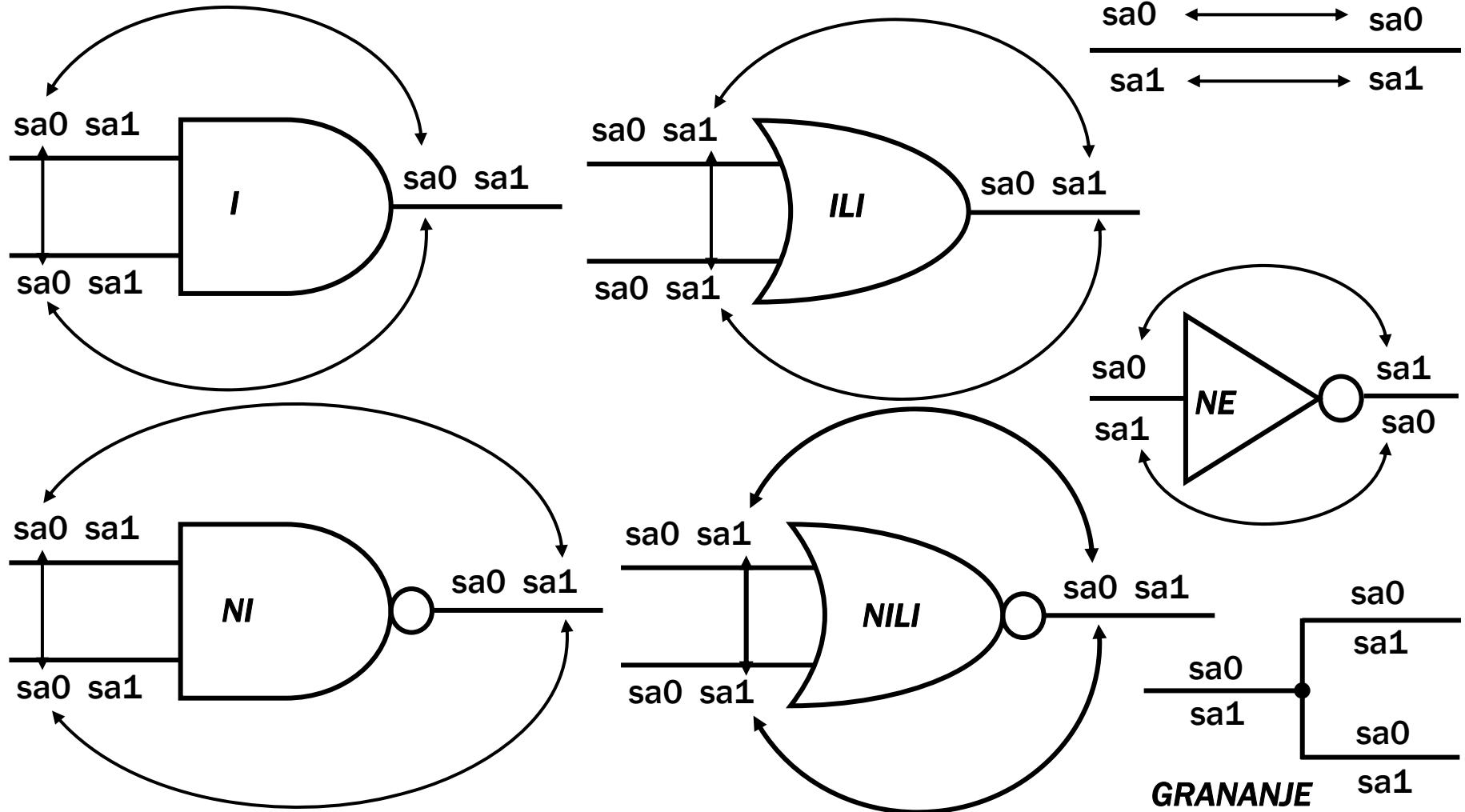


# Ekvivalentnost kvara

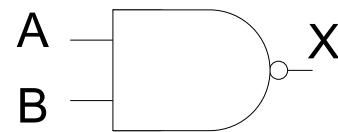
---

- broj mogućih mesta kvara = #PI + #skl. + #(izl. grana)
- kvarovi F1 i F2 su ekvivalentni ako svi testovi koji otkriju F1 ujedno otkriju i F2
- ako su kvarovi ekvivalentni tada su ekvivalentne i funkcije kvara
- sažimanje kvarova (**Fault collapsing**)
  - svi jednostruki kvarovi log. sklopova mogu se podijeliti u nepreklapajuće podskupove u kojima su svi kvarovi uzajamno ekvivalentni

# Primjeri ekvivalentnosti kvara

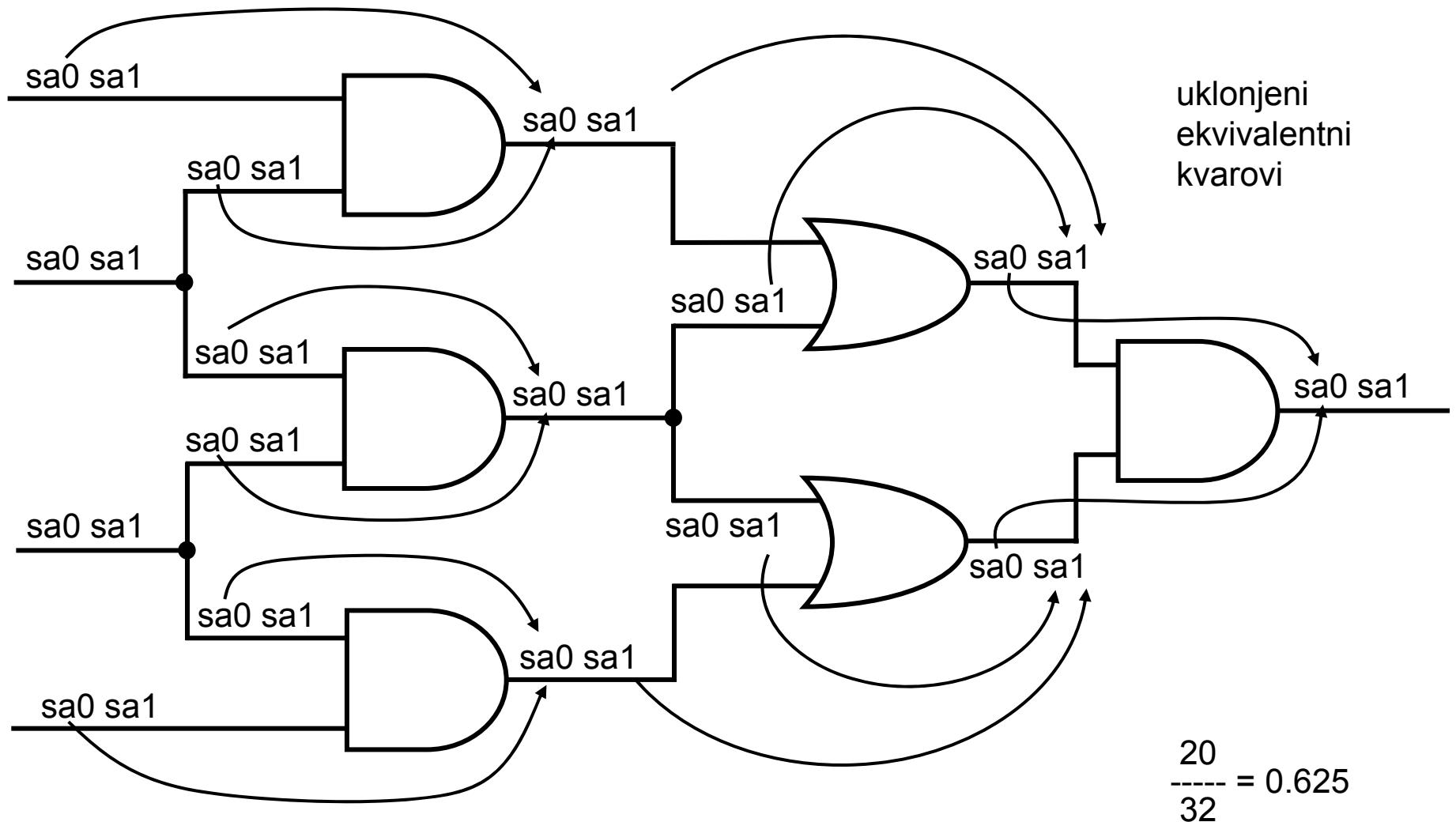


# Primjer ekvivalentnosti kvara



A	B	X	f <sub>1</sub>	f <sub>2</sub>	f <sub>3</sub>	f <sub>4</sub>	f <sub>5</sub>	f <sub>6</sub>
0	0	1	1	1	1	1	0	1
0	1	1	1	0	1	1	0	1
1	0	1	1	1	1	0	0	1
1	1	0	1	0	1	0	0	1

# Primjer ekvivalentnosti kvara

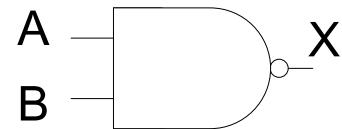


# Dominantnost kvara

---

- ako svi testovi kvara F1 otkriju drugi kvar F2, tada F2 dominira nad F1
- dominantnost sažimanja kvarova (Dominance fault collapsing)
  - ako kvar F2 dominira nad F1 tada se kvar F2 može ukloniti iz liste
  - dovoljno je promatrati ulaze
- Ako su dva kvara međusobno dominantna tada su ujedno i ekvivalentni

# Primjer dominantnosti kvara

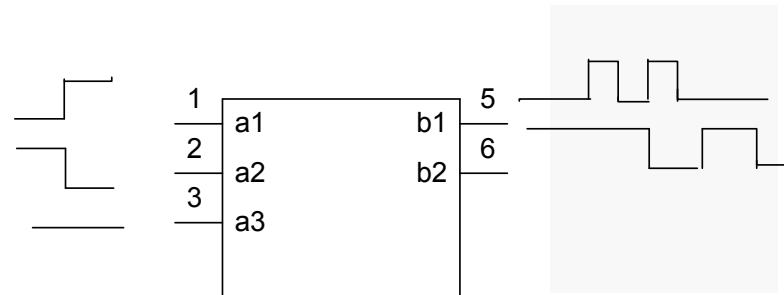


A	B	X	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$
0	0	1	1	1	1	1	0	1
0	1	1	1	0	1	1	0	1
1	0	1	1	1	1	0	0	1
1	1	0	1	0	1	0	0	1

# Kvar kašnjenja

---

- U stvarnim digitalnim sustavima neophodno je promatrati i vremena promjene signala
  - Sinkronizacija taktom – kombinacijski skloovi u stabilnom stanju tijekom 1 perioda



- Neophodno razmatrati dinamičke promjene signala
  - 110

# Model pogrešaka

---

- sredstvo klasificiranja posljedica fizičkih kvarova u sustavu
  - sa stanovišta modeliranja ne moramo ih zaključiti iz modela kvara
- Ciljevi
  - doseg loših informacija
  - doseg širenja pogreške
  - kašnjenja

- 
- efekti pogrešaka na
    - podatke
    - upravljanje
    - stanje
  - sklopovalski tipovi pogrešaka)
    - pogreške jednog bita
      - podatak, upravljanje, stanje
      - uobičajena uporaba
    - jednosmjerne pogreške (podatkovne)
    - podatkovne
    - ostale logičke pogreške

# Programski tipovi pogrešaka

---

- pogreške grananja
- nedostatak naredbe
- pogreške pokazivača

# Modeli zatajenja

---

- model sustava

- jedno ili više procesorski sustav
- izvođenje jednog ili više procesa
- interakcija procesa
  - razmjena poruka
  - distribuirani sustav
  - dijeljena memorija

- 
- zatajenja procesa ili sustava
  - klasifikacija
    - crash failure – sistem je permanentno u kvaru
    - omission failure – proces ponekad daje krive rezultate
    - timing failure – ulazi ili izlazi kasne ili dolaze prerano
    - Byzantine/Arbitrary failure – proizvoljno zlonamjerno ponašanje

# Simulacije

---

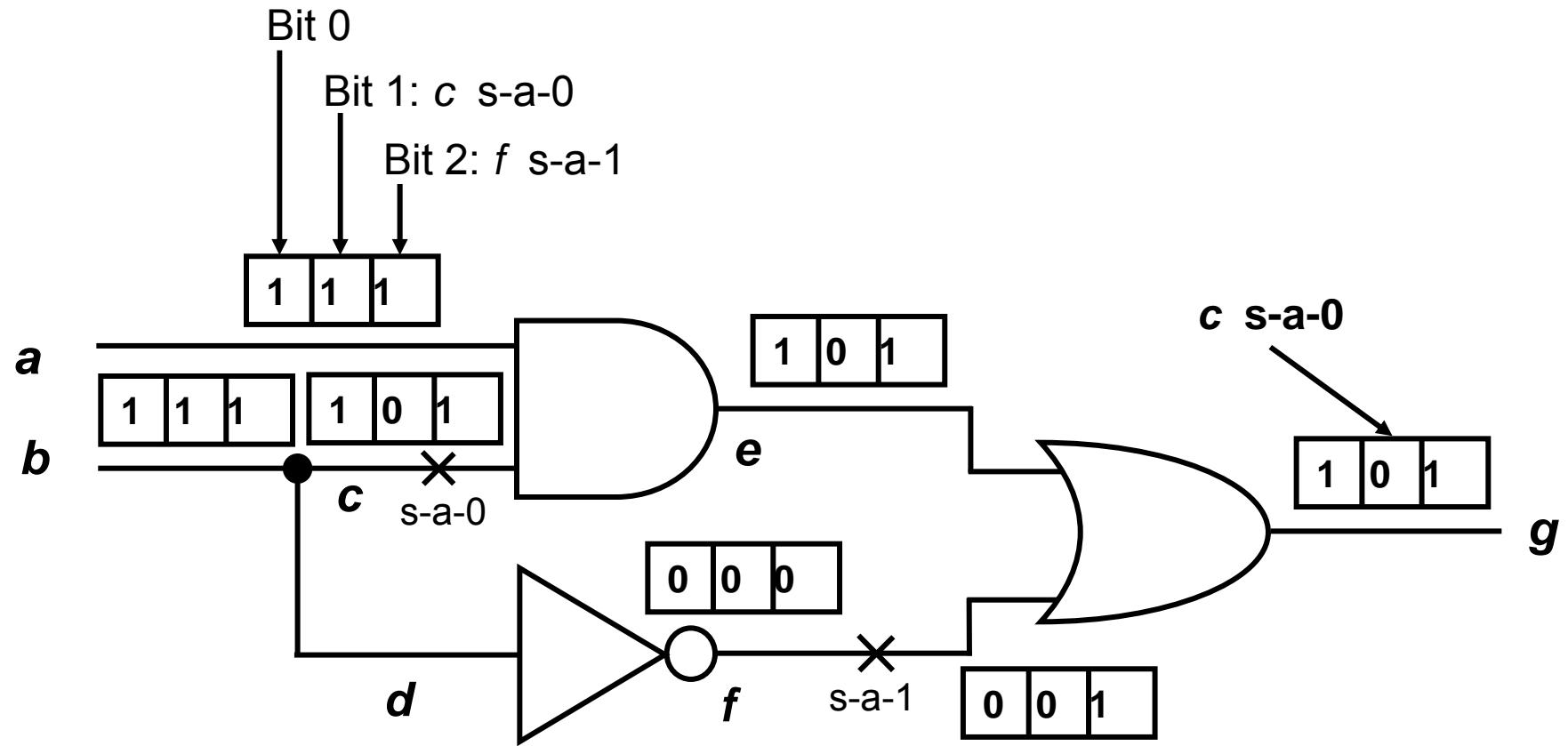
- Sekvencijalna
- Paralelne
- Diferencijalna

# Sekvencijalna simulacija

---

- Simulirati sklop bez kvara
- Ponavljam:
  - Promijeni opis sklopa uvođenjem kvara;
  - Simuliraj sve testne vektore, usporedi rezultat s ispravnim;
  - Prekini simulaciju ako je kvar otkriven.

# Paralelna simulacija



# Sažetak

---

- model kvara predstavlja pojednostavljenu aproksimaciju nedostataka koja omogućava analizu i testiranja.
- kvarovi kratkog spoja zahtijevaju specijalne testove

# Pitanja...

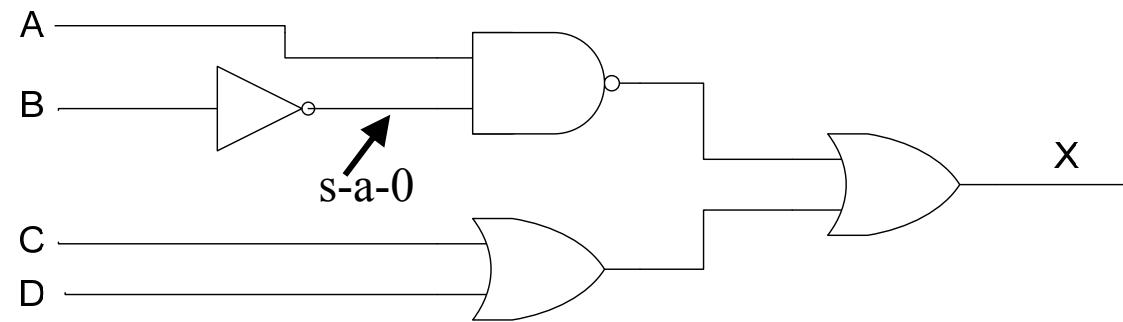
# Primjer...

---

- Procjena vremena testiranja 32 bitnog zbrajala  
ako 1 test traje 1ns

# Primjer

## ■ Mogućnost otkrivanja kvara



1		1
0		1
0		0
0		0

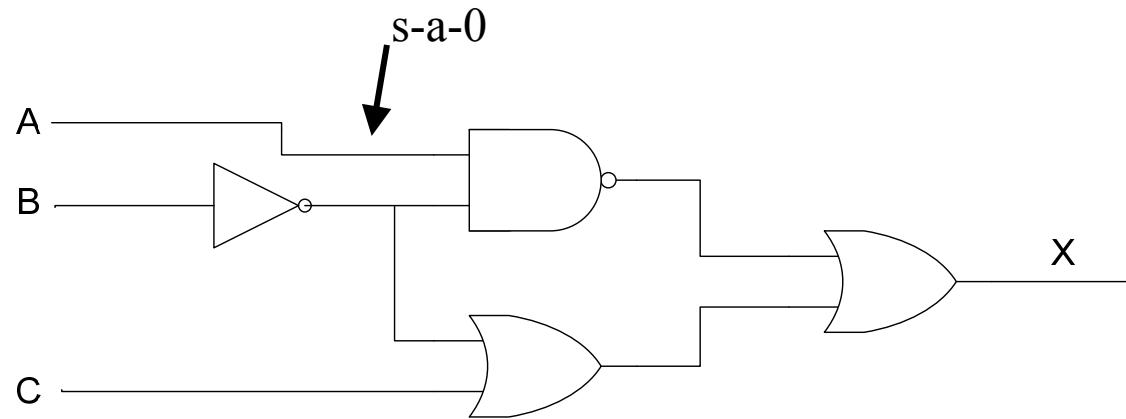
## Ispravno

$$\begin{array}{r} 1 \\ 0 \\ 0 \\ 0 \end{array} \qquad \begin{array}{r} 1 \\ \mathbf{0} \\ 0 \\ 0 \end{array} \qquad \begin{array}{r} 1 \\ 0 \\ 0 \\ 0 \end{array}$$

Kvar

# Primjer 2

## ■ Maskiranje kvara i redundancija



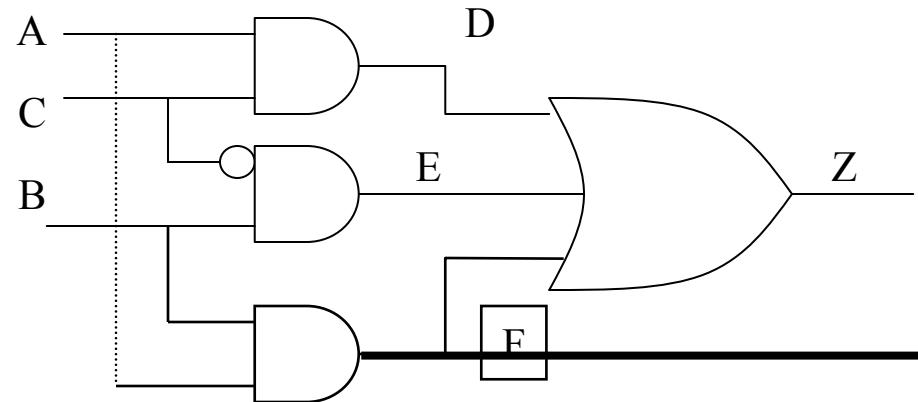
1	1	0	0
0	0	1	
X	X		

1	0	1	1
0	0	0	
X	X		

# Primjer 3

---

- $Z = AC + \overline{BC} + AB$



- Gdje je problem testiranja?

# Pouzdanost računalnih sustava

---

**Predavanje 3:      Testiranje digitalnih sustava**

**Prof.dr.sc. Vlado Sruk**



**Sveučilište u Zagrebu**  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroel., računalne i inteligentne sustave*



# Sadržaj

---

- Principi oblikovanja za testiranje
- Strukturno oblikovanje za testiranje
- Samotestiranje
- BIST
- JTAG

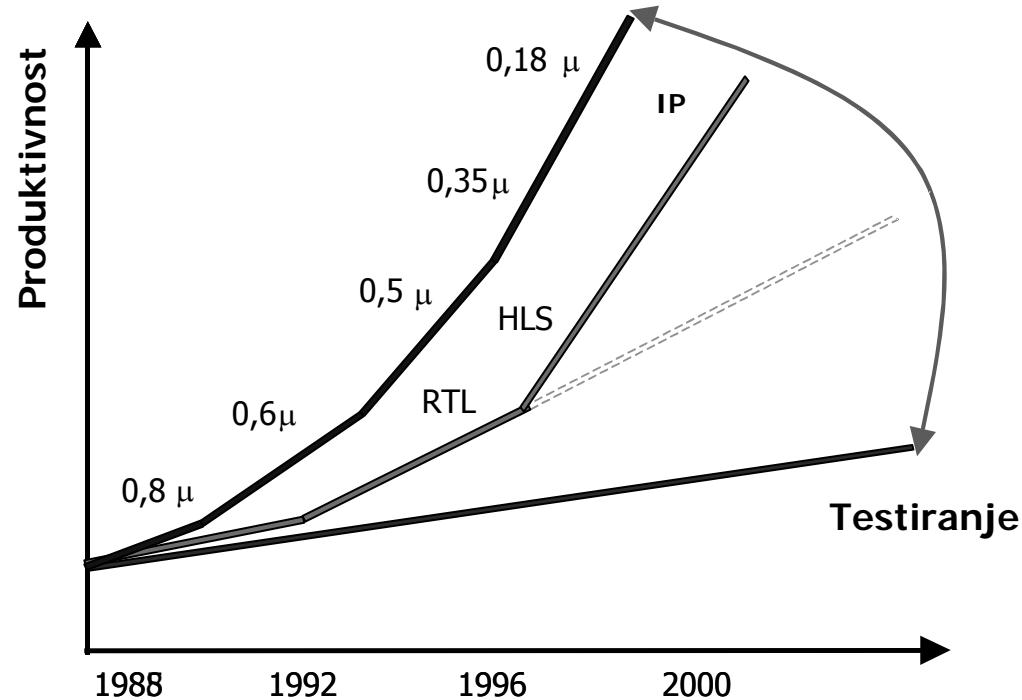
# Ciljevi testiranja

---

- Različiti ciljevi zahtijevaju različite strategije i testove, dokumentaciju i daju različite rezultate
- Pronaći i ispraviti važne pogreške
- Provjeriti sukladnost rada (interoperability) s ostalim komponentama
- Pomoći u donošenju odluke o puštanju u rad/prodaju
- Zaustaviti prerano puštanje u rad/prodaju (engl. premature product releases)
- Minimizirati troškove tehničke podrške
- Procijeniti sukladnost specifikacijama
- Sukladnost s normama (zakonske, tehničke..)
- Minimizirati rizik tužbi obzirom na sigurnost
- Definirati način sigurne uporabe
- Procjena kvalitete

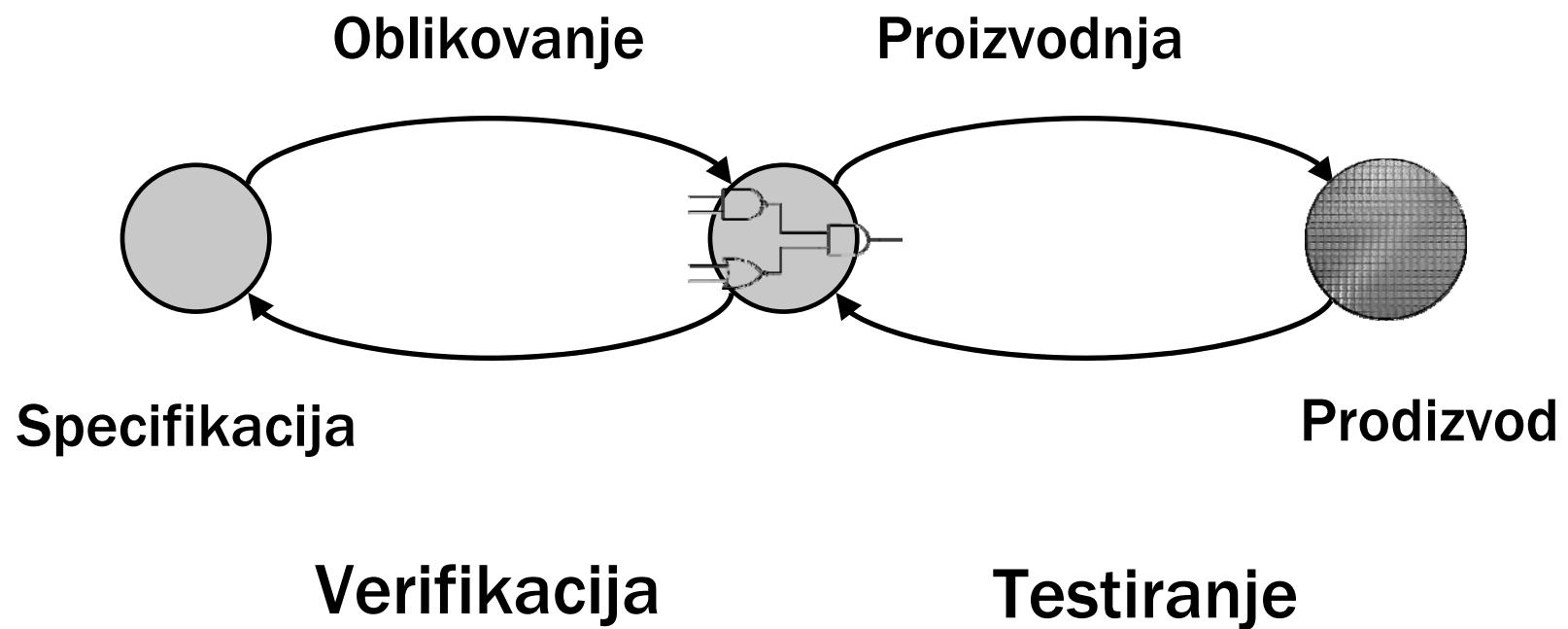
# Metode oblikovanja sustava

## ■ Kriza produktivnosti



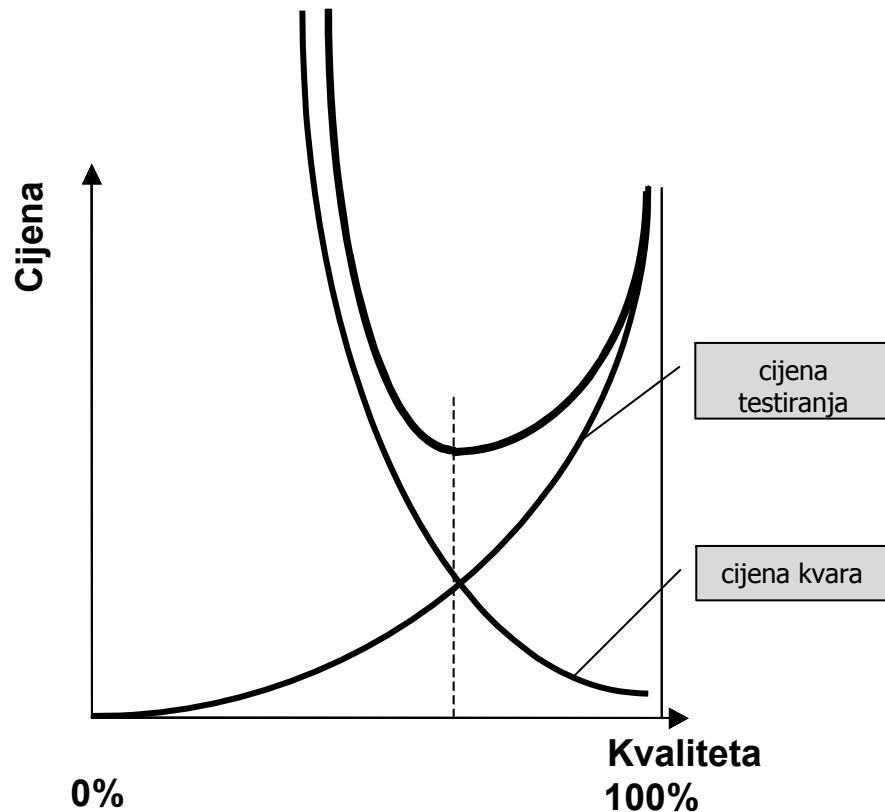
# Odnos verifikacije i testiranja

---



# Odnos testiranja i kvalitete

---



**Kako riješiti problem testiranja?**

---

# Oblikovanje za testiranje

---

- engl. Design for Test (Testability) -DFT
- “*A digital IC is testable if test patterns can be generated, applied, and evaluated in such a way as to satisfy predefined cost budget and time scale.*“

[Bennetts 1984]

- Česta definicija:
  - Oblikovanje za testiranje (*Design for test*) obuhvaća tehnike oblikovanja koje omogućavaju cjenovno efikasno generiranje i primjenu testova
- Preoblikovanje ili dodavanje sklopoljja s ciljem olakšanog provođenja testiranja
- Cilj:

$$C(\text{Oblikovanje} + \text{Testiranja}) < C(\text{oblikovanje}) + C(\text{Testiranja})$$

$$C(\text{DFT}) = (C_D + \Delta C_D) + Q(C_P + \Delta C_P)$$

$$C(\text{DFT}) + C(\text{Testiranja}') < C(\text{oblikovanje}) + C(\text{Testiranja})$$

$$C(\text{Testiranja}) = C_{TGEN} + (C_{TEST} + (1 - Y) C_{TS}) Q$$

# Metode DFT-a

---

## ■ Ad-hoc

- Omogućavanje testiranja teško dostupnih signala
- Omogućavanje inicijalizacije modula
- Izbjegavanje dizanja “loših” za testiranje (npr. povratna veza kombin. logike, velik grananja, ..)

## ■ Strukturne metode

- Dodatno sklopolje omogućava testiranje prema unaprijed određenim procedurama
- Zasnovane na ispitivanju
- samotestiranje

# Komponente oblikovanja za testiranje

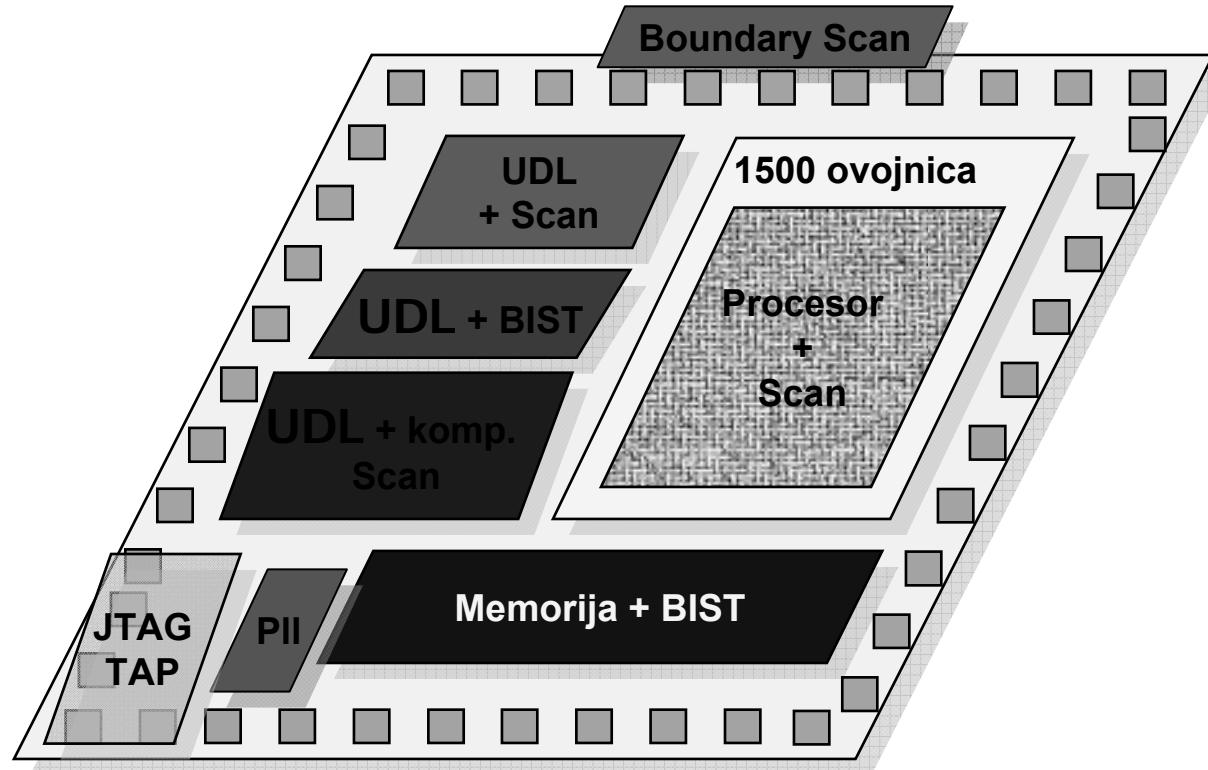
---

## ■ Standardizacija područja

- Tehnike ispitivanja (engl. Scan-based tests)
  - Korisnička logika (engl. User Defined Logic UDL)
  - Omotači gotovih jezgri (engl. core wrappers)
- Ugrađeno samotestiranje (engl. Built In Self Testing - BIST)
- Testiranje memorija
- Kontrola takta (engl. Clock Control)
- Ispitivanje granica - Boundary Scan

# Primjena komponenti oblikovanja

---



# Koncept oblikovanja za testiranje

---

## ■ Atributi

- Upravljivost
- Promatranje rezultata

## ■ Programska analiza atributa

- Analitičko određivanje za promatrani sklop
- Dodavanje testnih/pristupnih točaka u dizajn

## ■ Princip podijeli i vladaj

# Osnovna pitanja testiranja

---

## ■ Generiranje testnih uzoraka

- Iscrpno
- Slučajno
- pseudoslučajno

## ■ Primjena testnih uzoraka

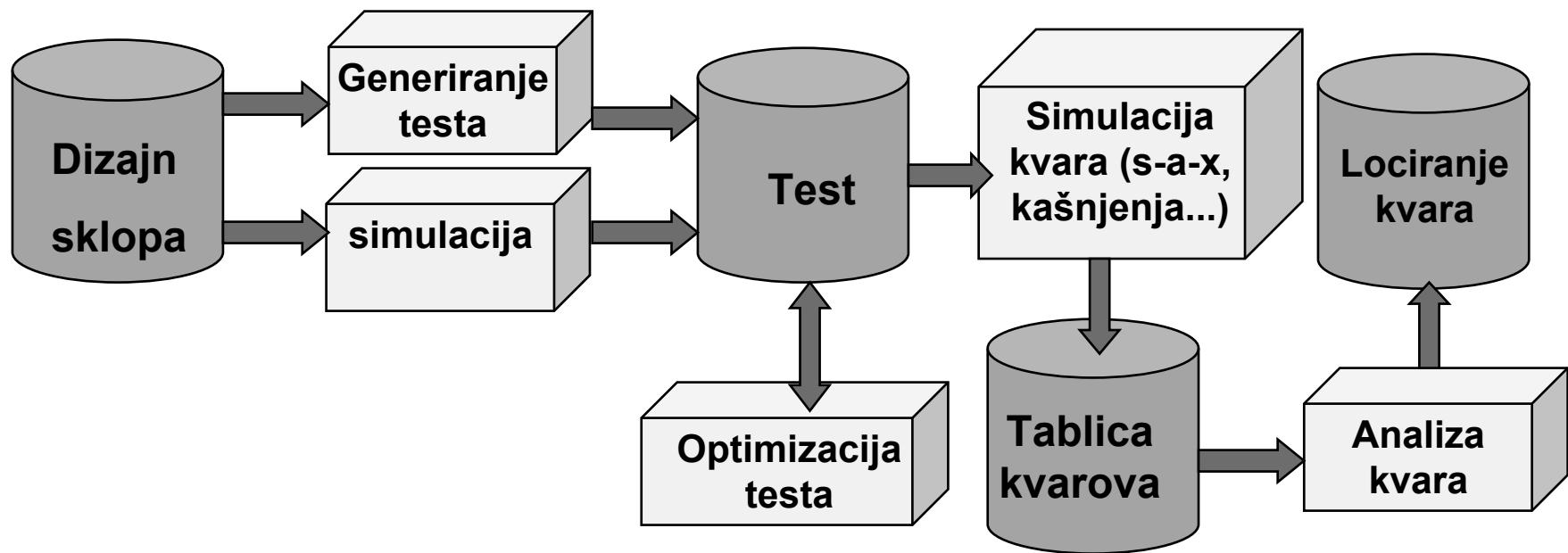
- ATE
- BIST

## ■ Obrada rezultata

- Usporedba
- Bez usporedbe - verifikacija koda

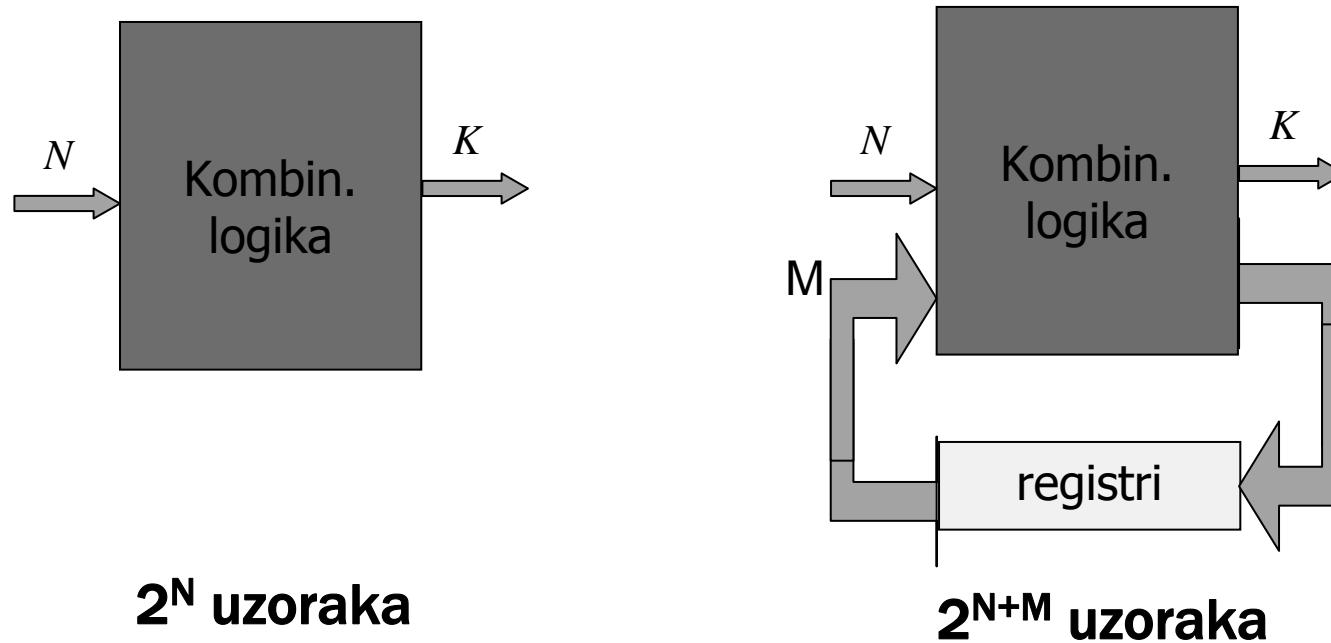
# Arhitektura sustava testiranja

---



# Složenost testiranja

- Iscrpno testiranje u složenijim sklopovima teško provedivo
- Kombinacijske i sekvencijski sklopovi



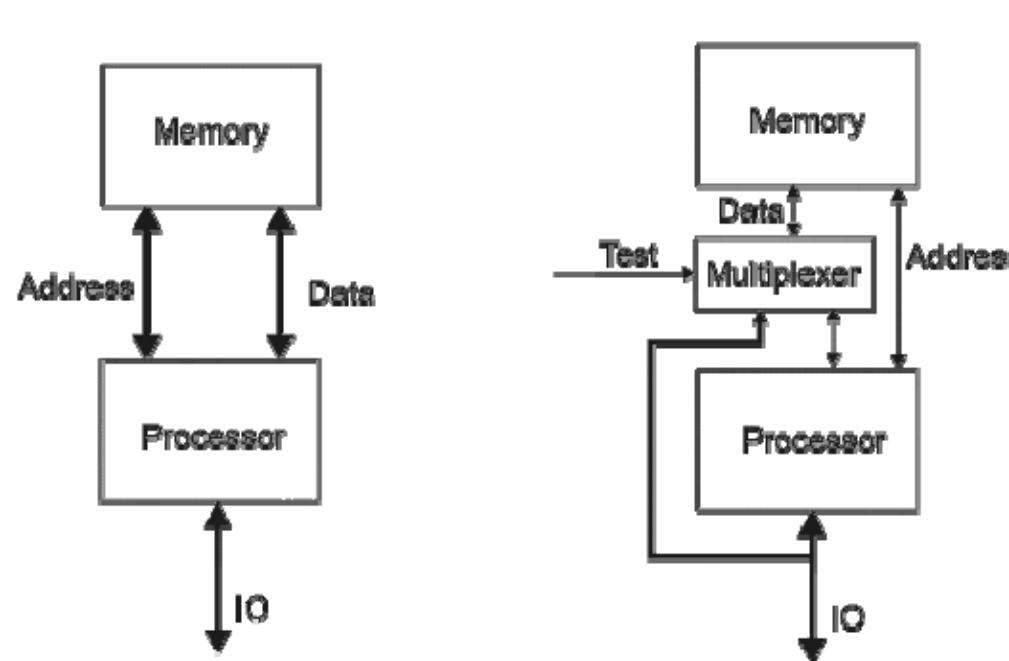
# Pravila ad-hoc tehnika

---

- Jednostavna inicijalizacija sklopova
- Omogućiti prekidanje povratnih veza
- Podjela velikih brojača na manje
- Izbjegavanje redundancije
- Odvajanje analognog i digitalnog sklopolja
- Izbjegavanje asinkrone logike
- Paziti na zahtjeve testiranja (broj pinova, smještaj, ...)

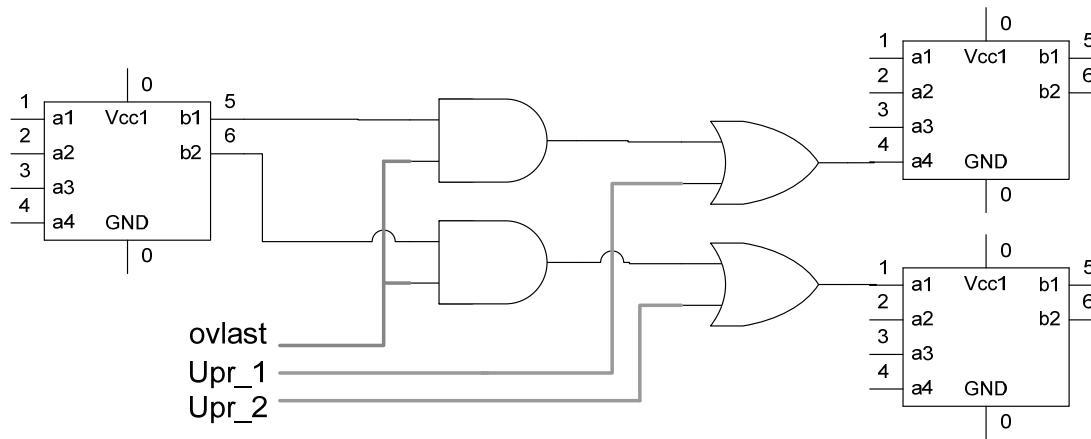
# Razdvajanje modula

- Multiplekserima
- Pogodno za ad-hoc tehnike
- Normalan i testni način rada



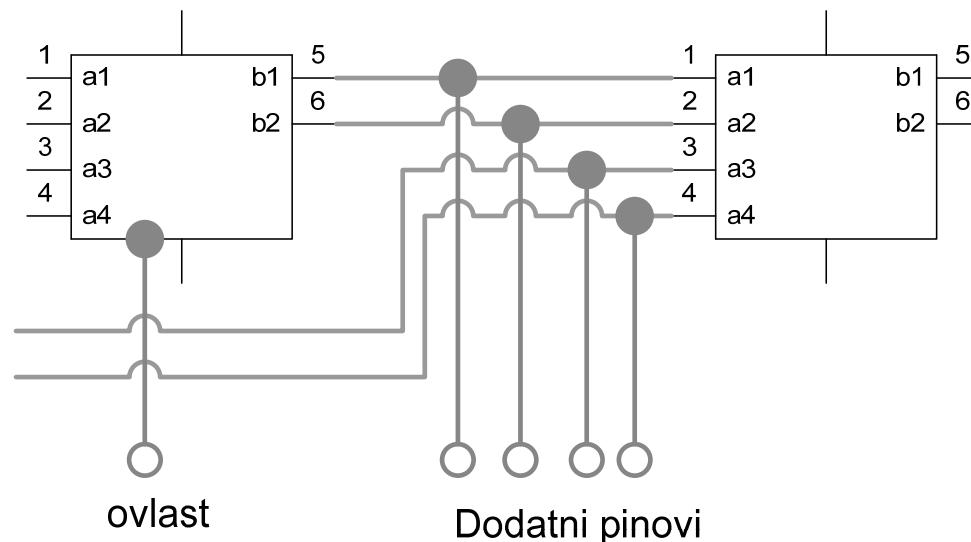
# Razdvajane modula

## ■ Linije ovlašćivanja (engl. delegating)



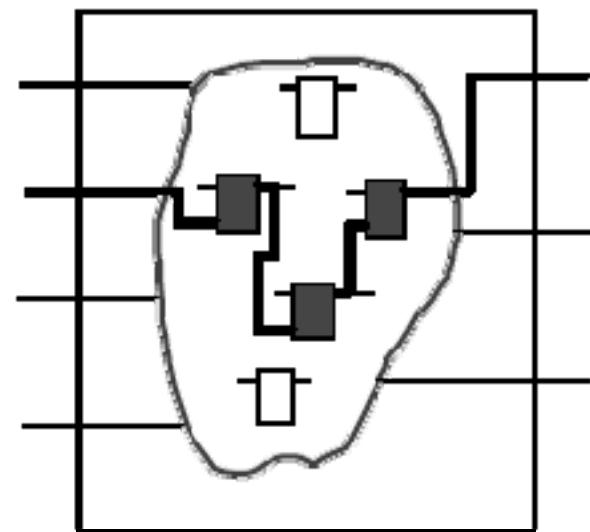
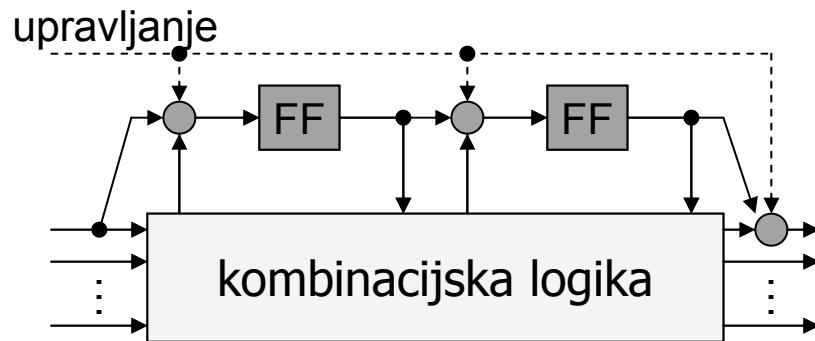
# Testne točke

- Pogodno za testiranje metodom prostirke iglica (engl. bed of nails)
- 1970 – In circuit testing



# Princip ispitivanja kombinacijskih sklopova

- Pohraniti željena stanja u bistabil
- Očitati vrijednosti iz bistabila



- Djelomično ispitivanje

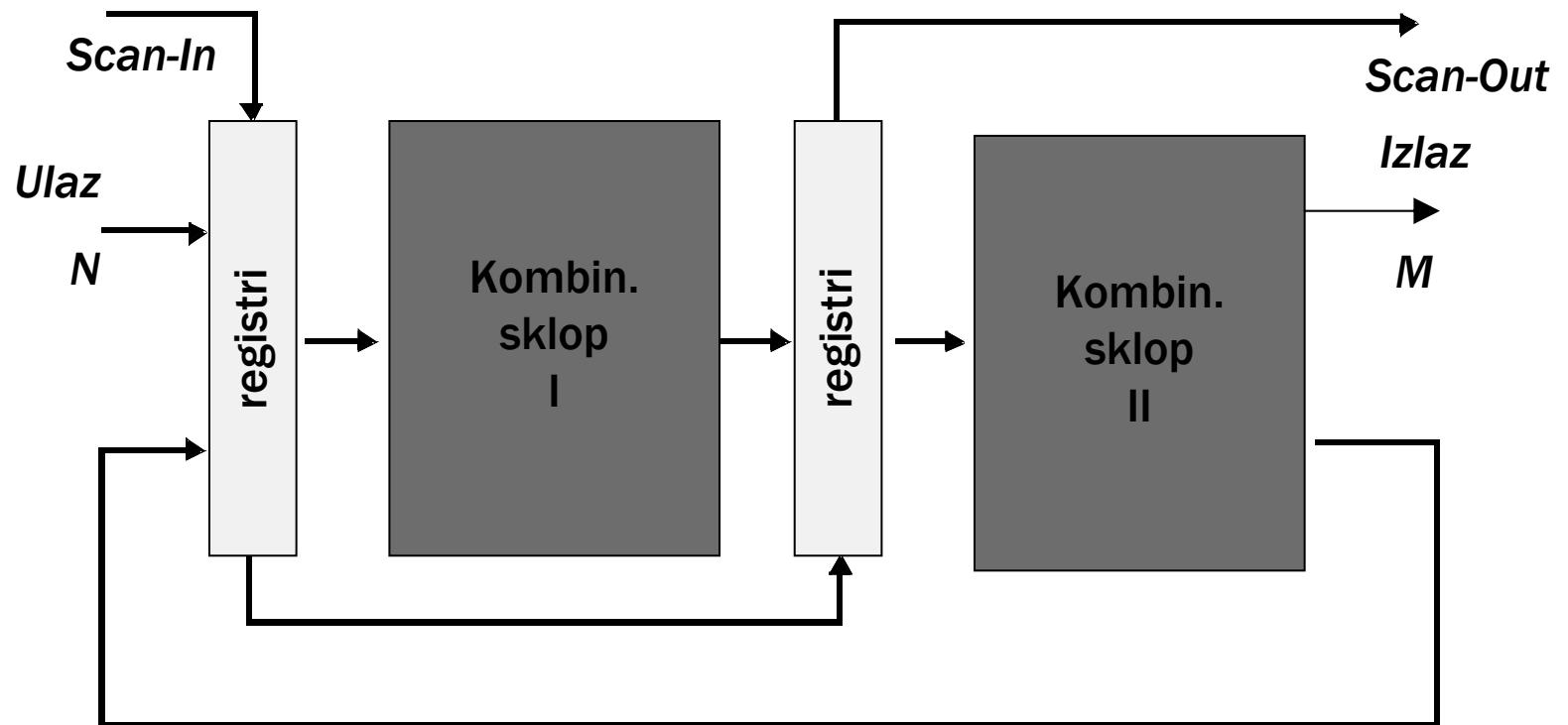
# Tehnike ispitivanja

---

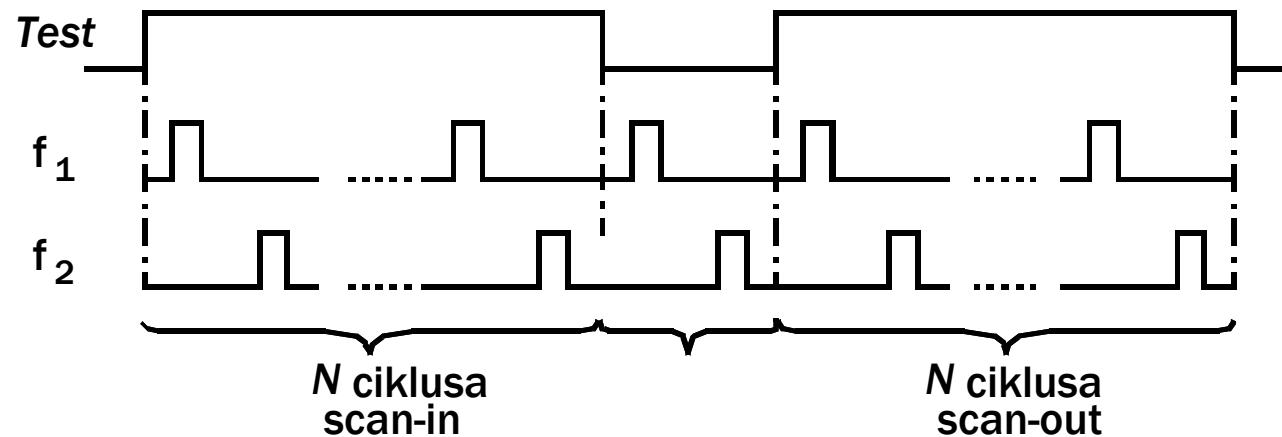
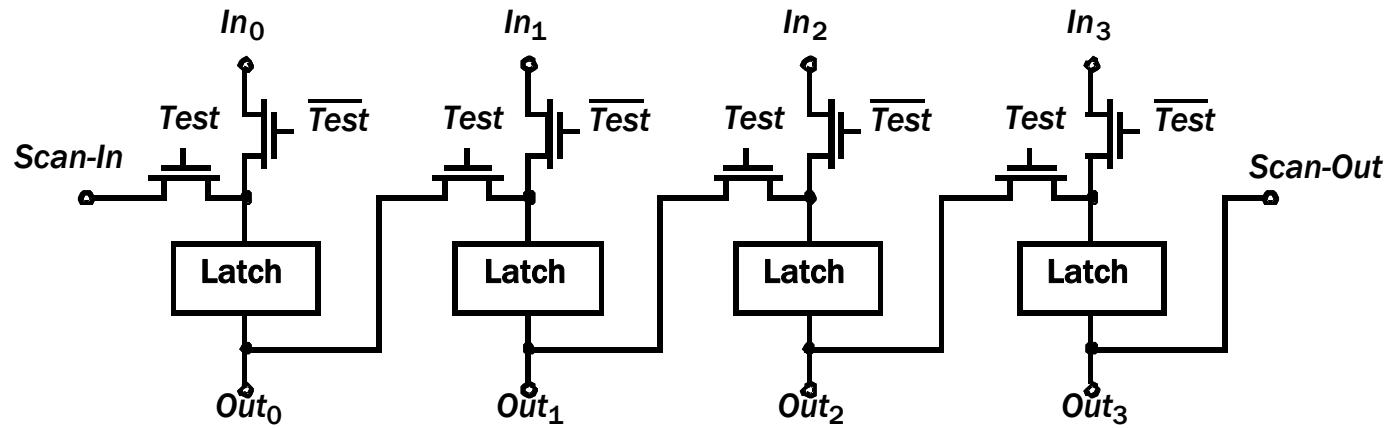
- engl. scan-based
- Osnovna ideja je upravljanje i promatranje stanja bistabila
- Promjena dizajna tako da omogućimo pristup bistabilima u fazi testiranja
  - Bistabili čine veliki posmačni registar
  - Povezivanje ulaza, izlaza i kontrole na vanjski svijet (konektori)
  - Ispitivanje omogućava postavljanje i čitanje stanja bistabila
- Osnovni pristupi:
  - Modifikacija posmačnog registra (**Shift-register modification**)
  - Ispitivanje puta (**Scan path**)
  - Level-sensitive scan design (**LSSD**)
  - Slučajan pristup registrima (**Random access**)

# Princip implementacije

---

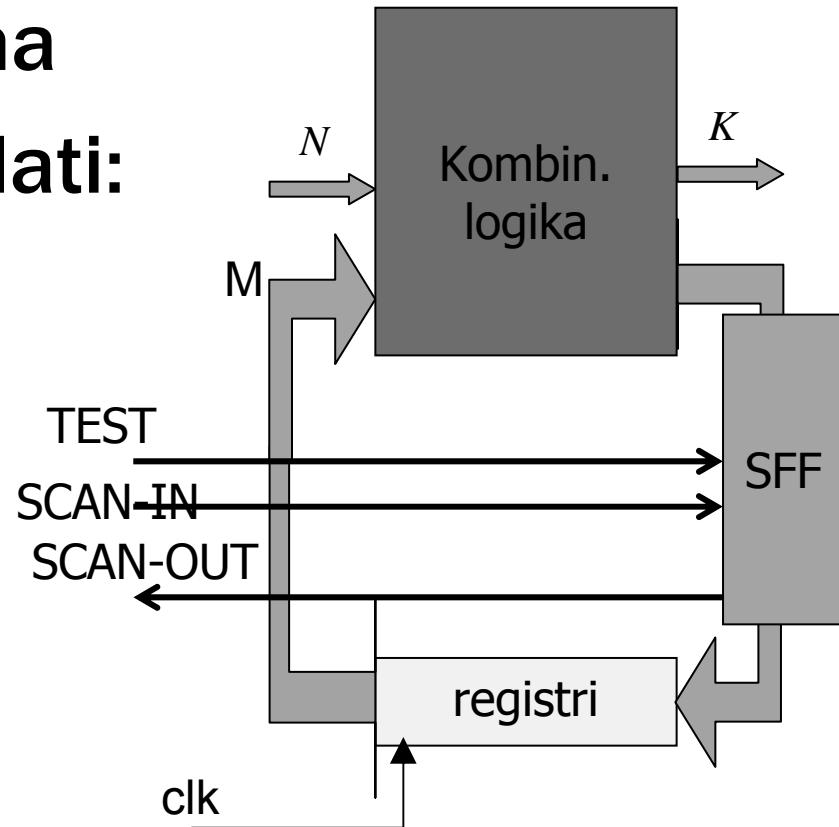


# Princip rada ispitivanja

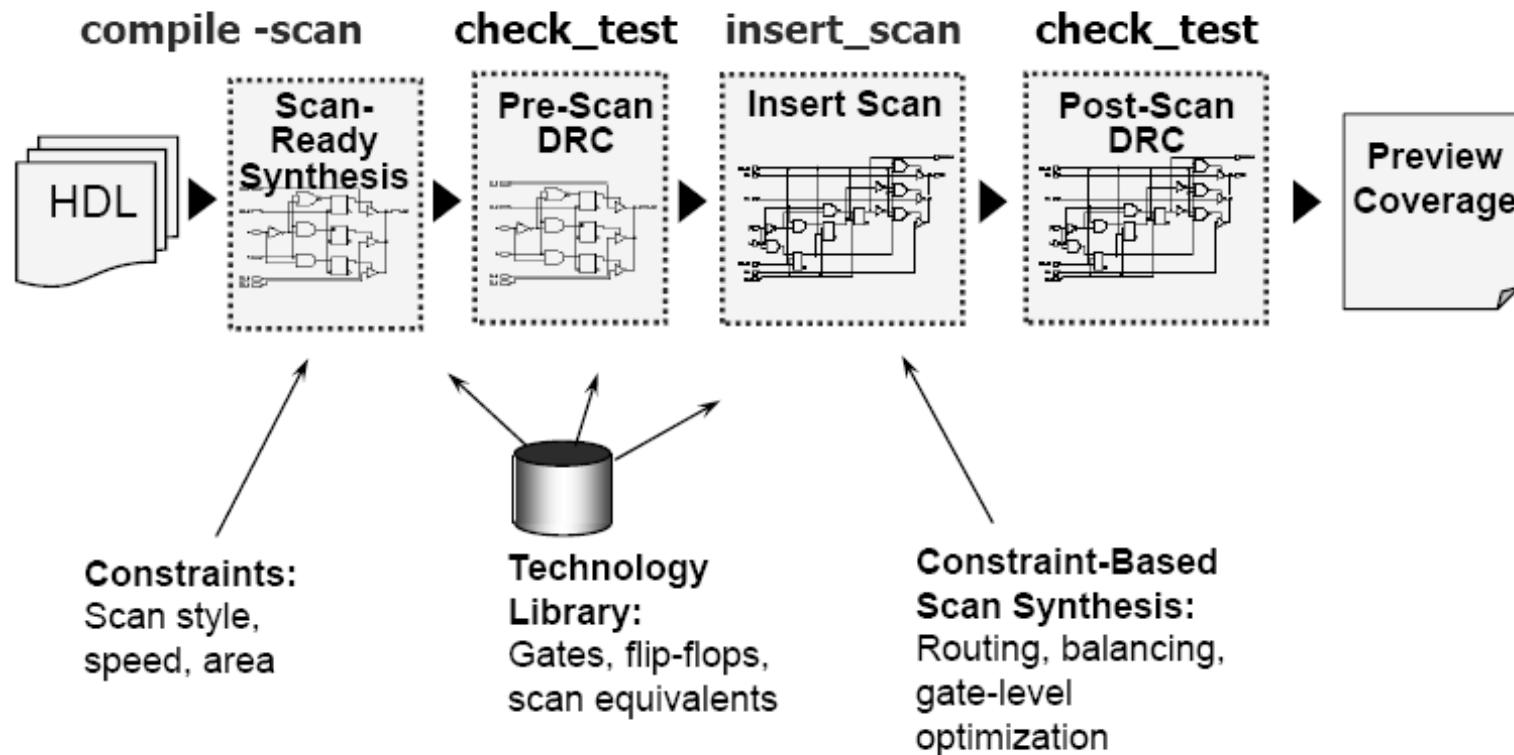


# Primjena na sekvencijske sklobove

- Problem pristupa stanjima
- Za kontrolu potrebno dodati:
  - TEST mode pin (T)
  - SCAN-IN pin (SI)
  - SCAN-OUT pin (SO)
  - bistabil i MUX ispred



# Primjer DFT metodologije



# Jednostavno testabilni sklopovi

---

- Ponavljajući kombinacijski sklopovi: Ponavljajuća logička polja (engl. Iterative Logic Arrays -ILA)
- polja identičnih struktura
  - Zbrajala, množila i sl.
  - Memorija RAM, ROM ,...
  - FPGA
  - bit-slice logika
- Svojstvo testiranja konstantni brojem uzorak bez obzira na veličinu polja - engl. **C-testable designs**

# Standard Boundary Scan

---

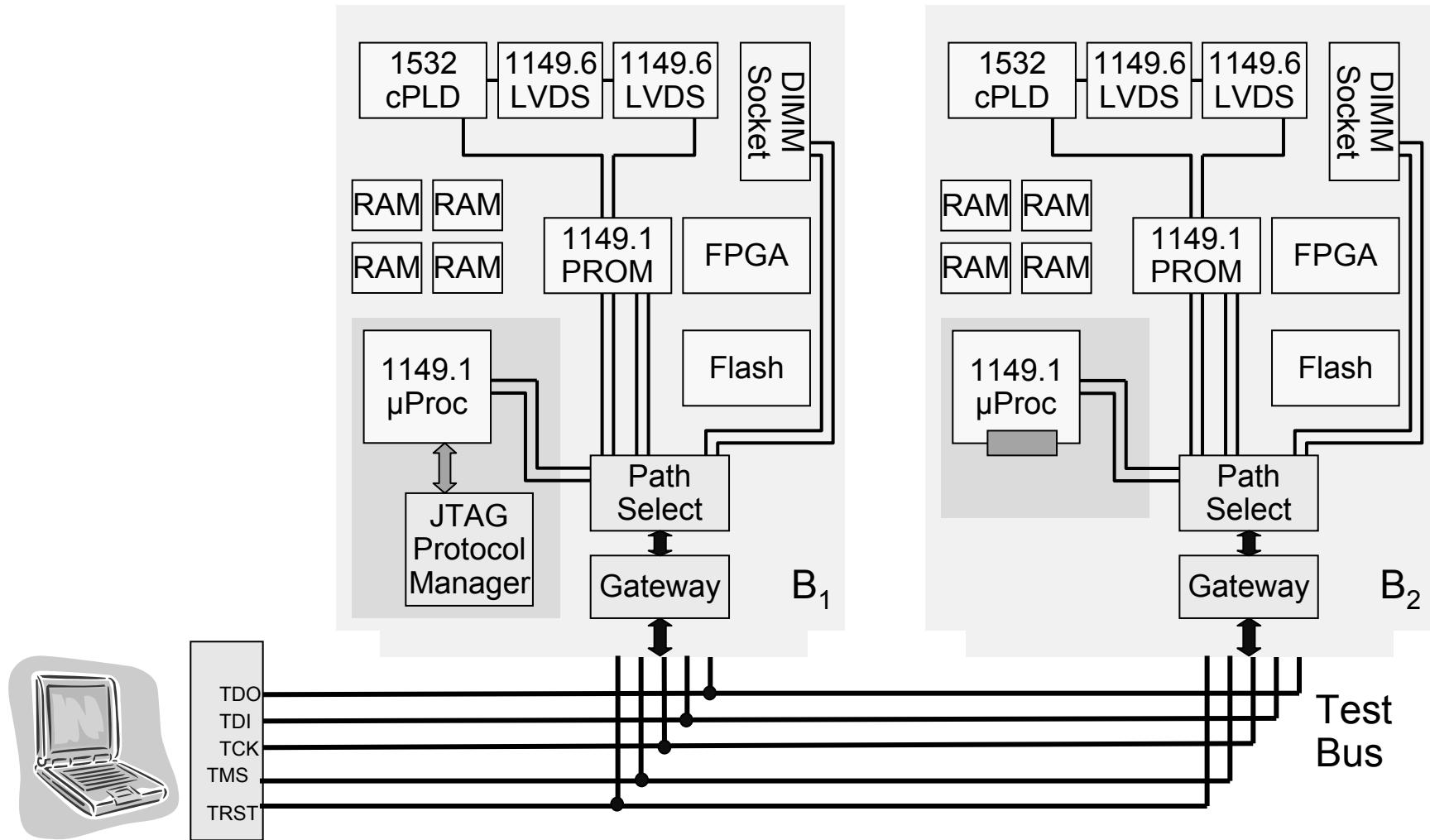
- 1985 – Joint European Test Action Group (JETAG, Philips)
- 1986 – VHSIC Element-Test & Maintenance (ETM) bus standard
  - 1988 – Joint Test Action Group (JTAG) predlaže Boundary Scan Standard
- 1990 – Boundary Scan IEEE Std. 1149.1-1990
  - Predložen jezik Boundary Scan Description Language (BSDL)

# Boundary Scan Testing BST

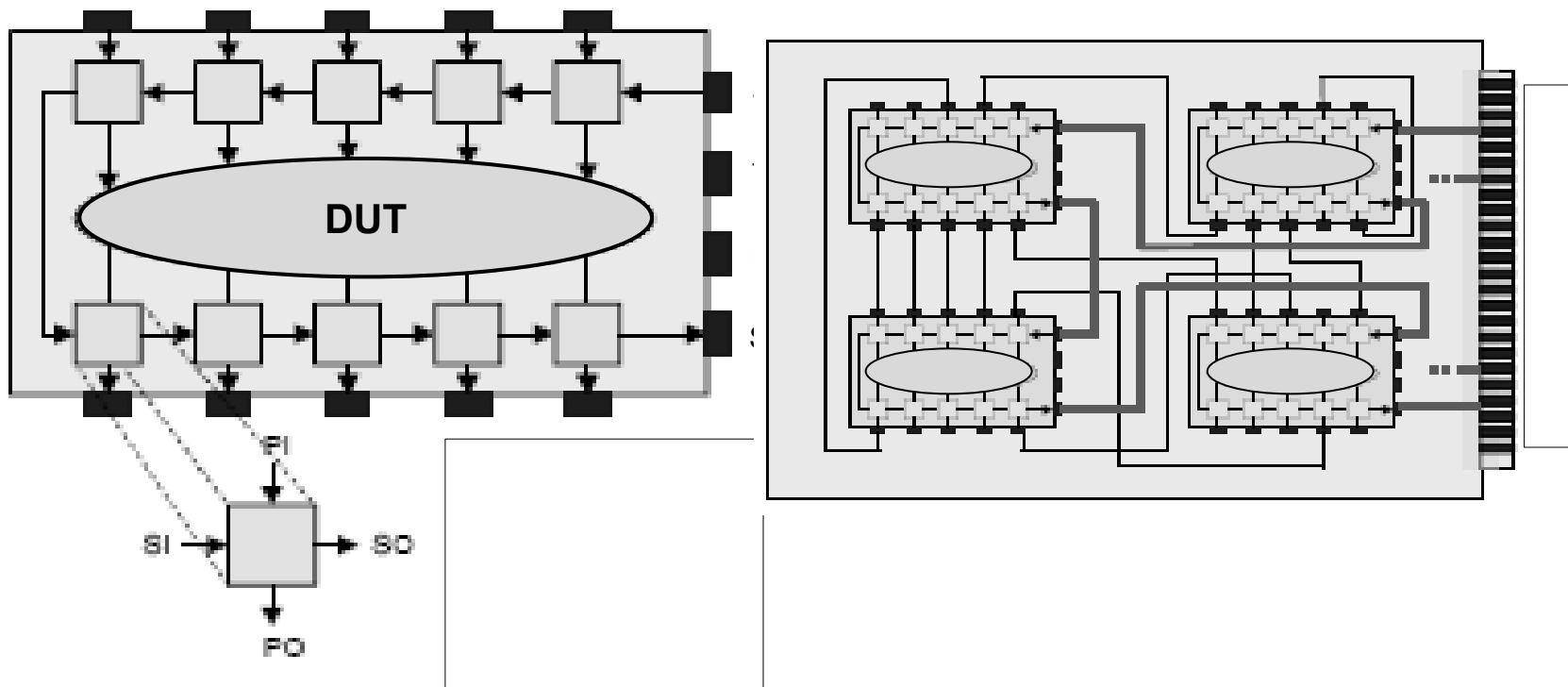
---

- Testiranje složenih gusto pakiranih štampanih pločica ili MCM-a
- Nema utjecaja na normalan rad
- Zauzima 3-20% površine!!
- Dodatni pinovi (Test Access Point TAP):
  - TDI
  - TDO
  - TCK
  - TMS
  - TRST

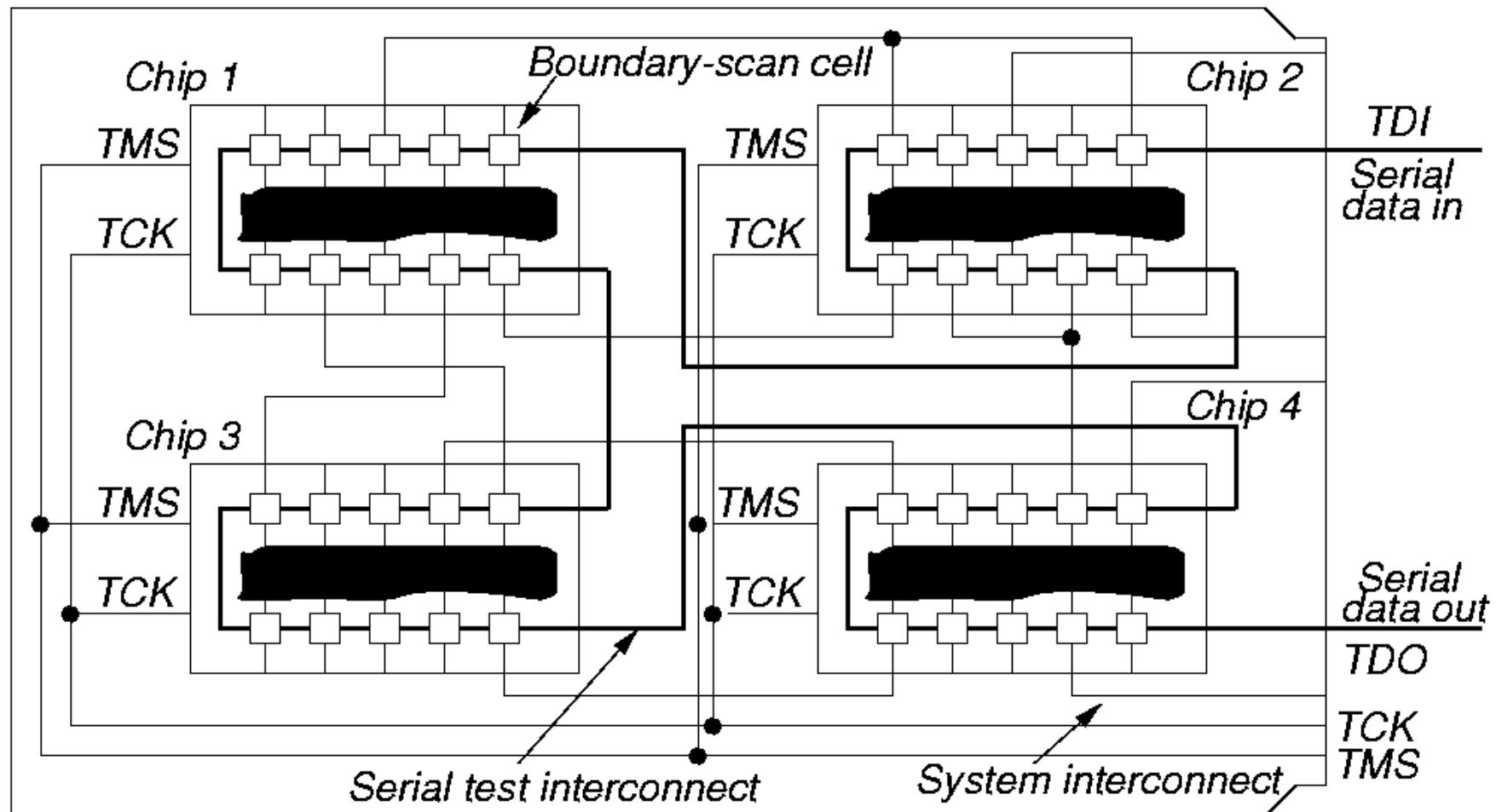
# Ideja



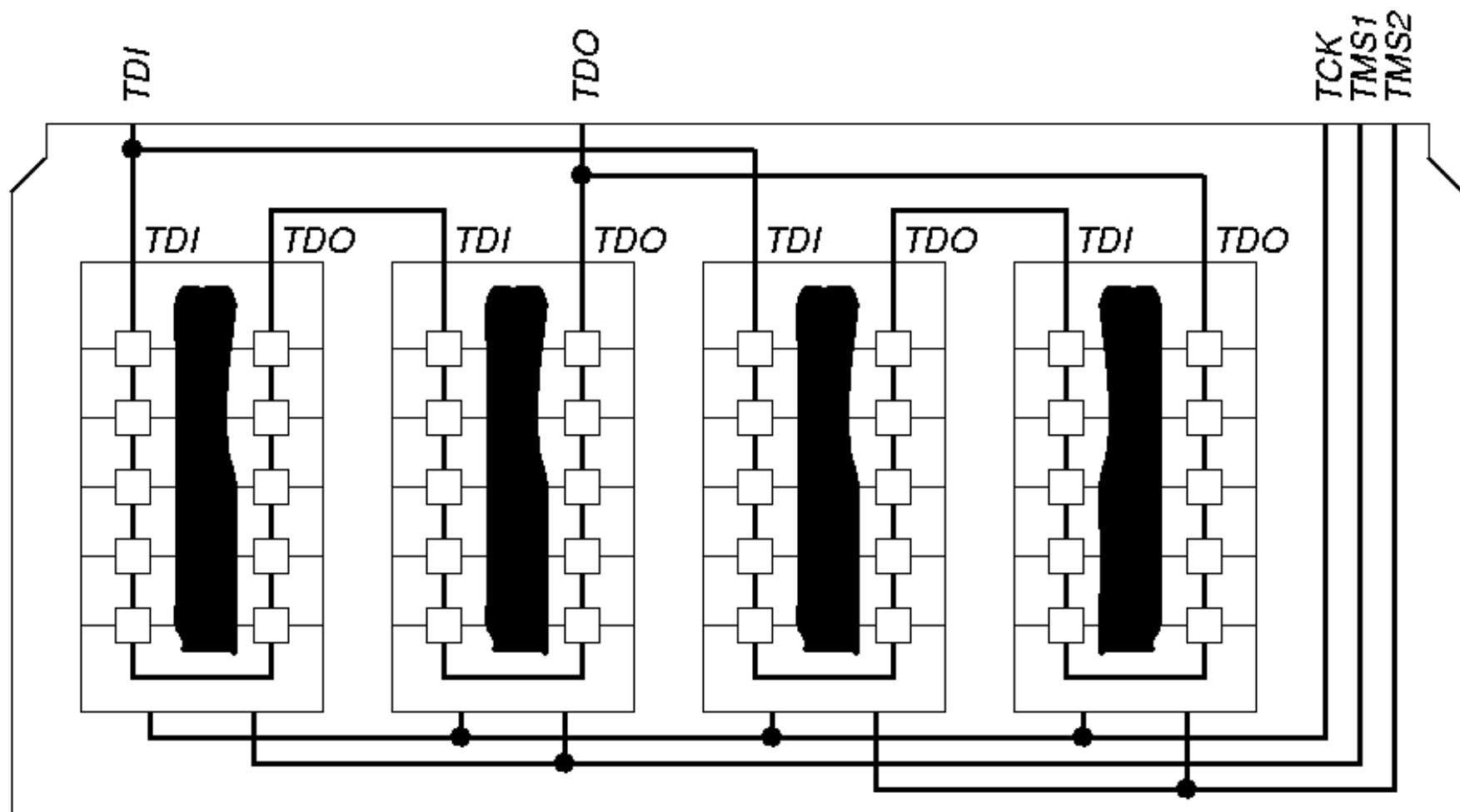
- Omogućavanje pristupa inače nedostupnim točkama sklopa
- Ulančavanje više sklopova



# Serijsko testiranje

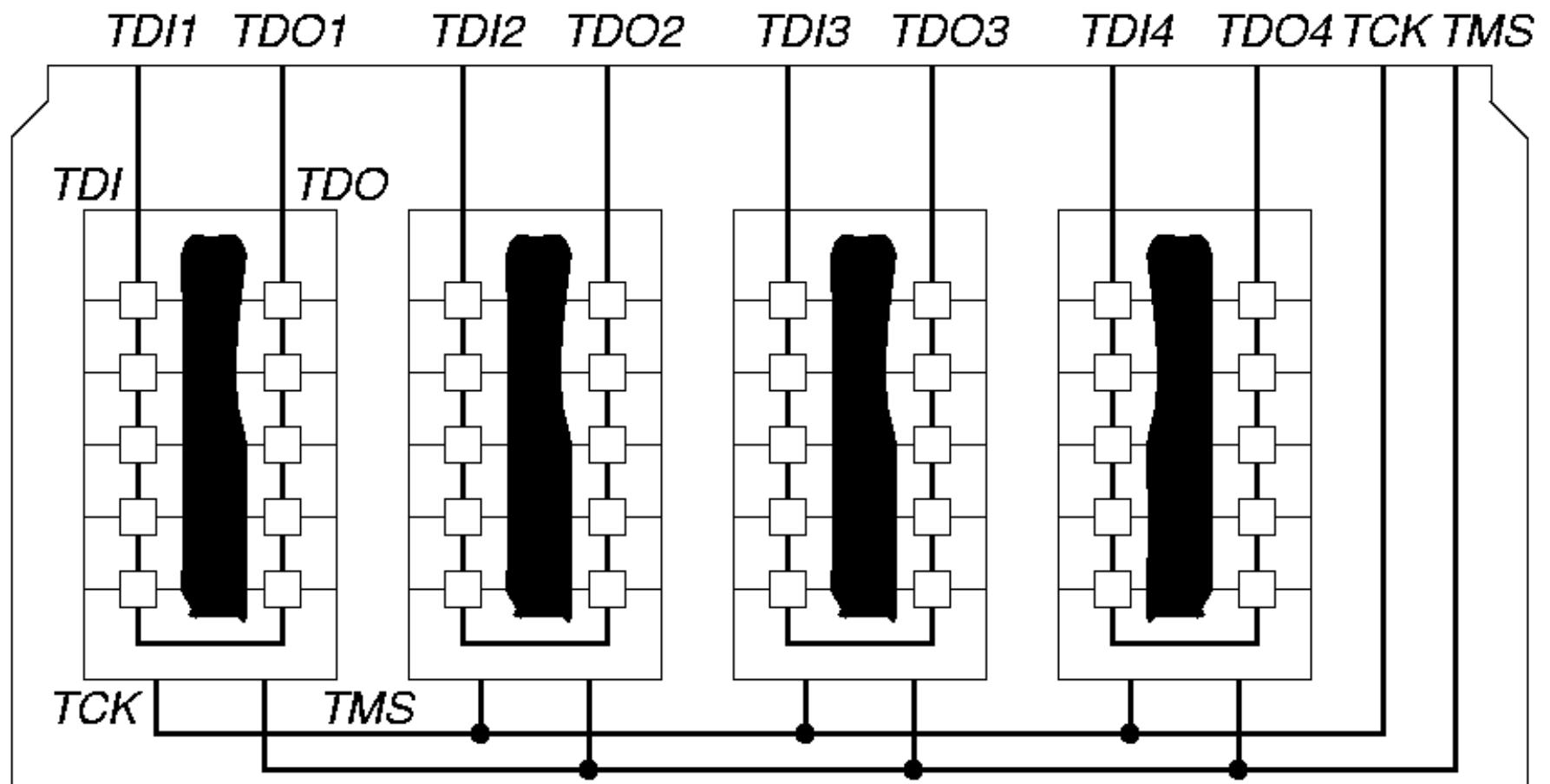


# Paralelno testiranje



# Neovisno testiranje

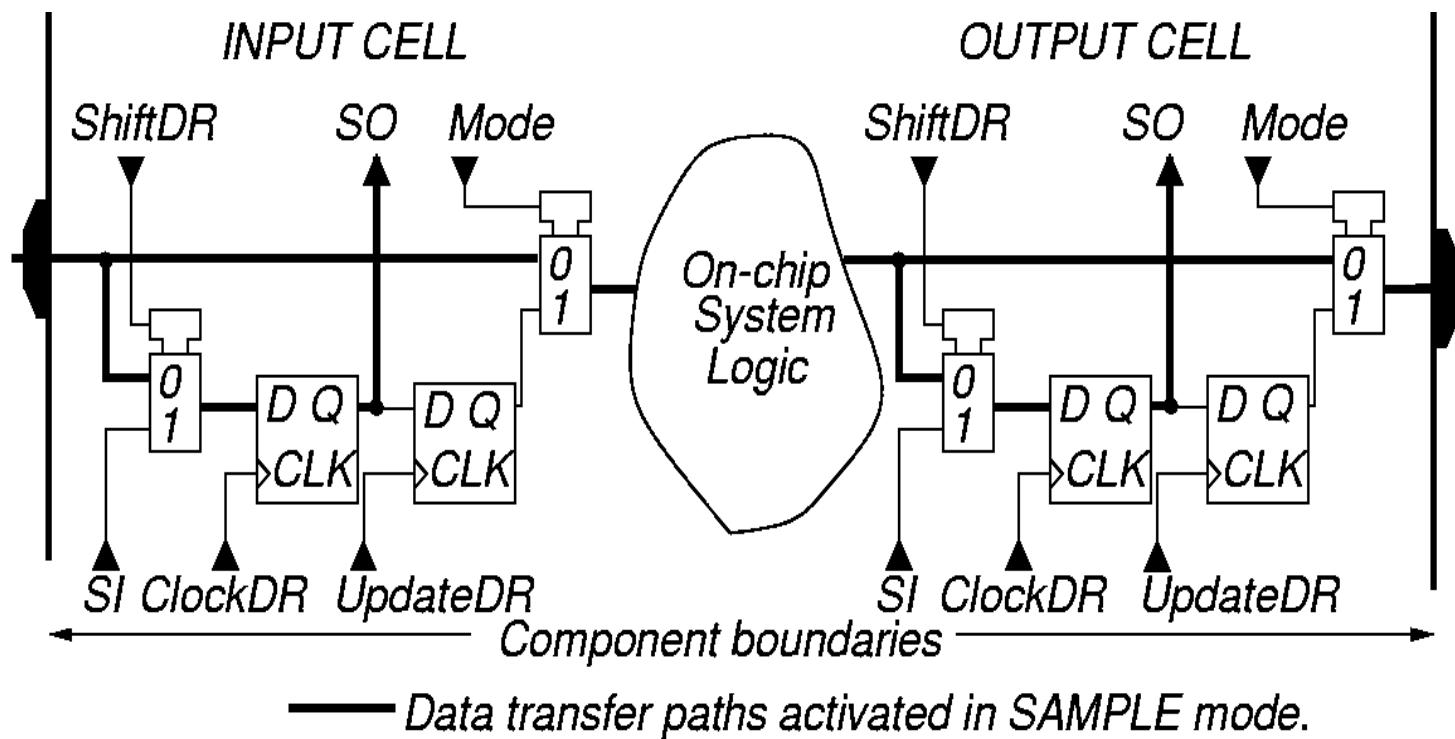
---



# Boundary Scan načini rada

## SAMPLE

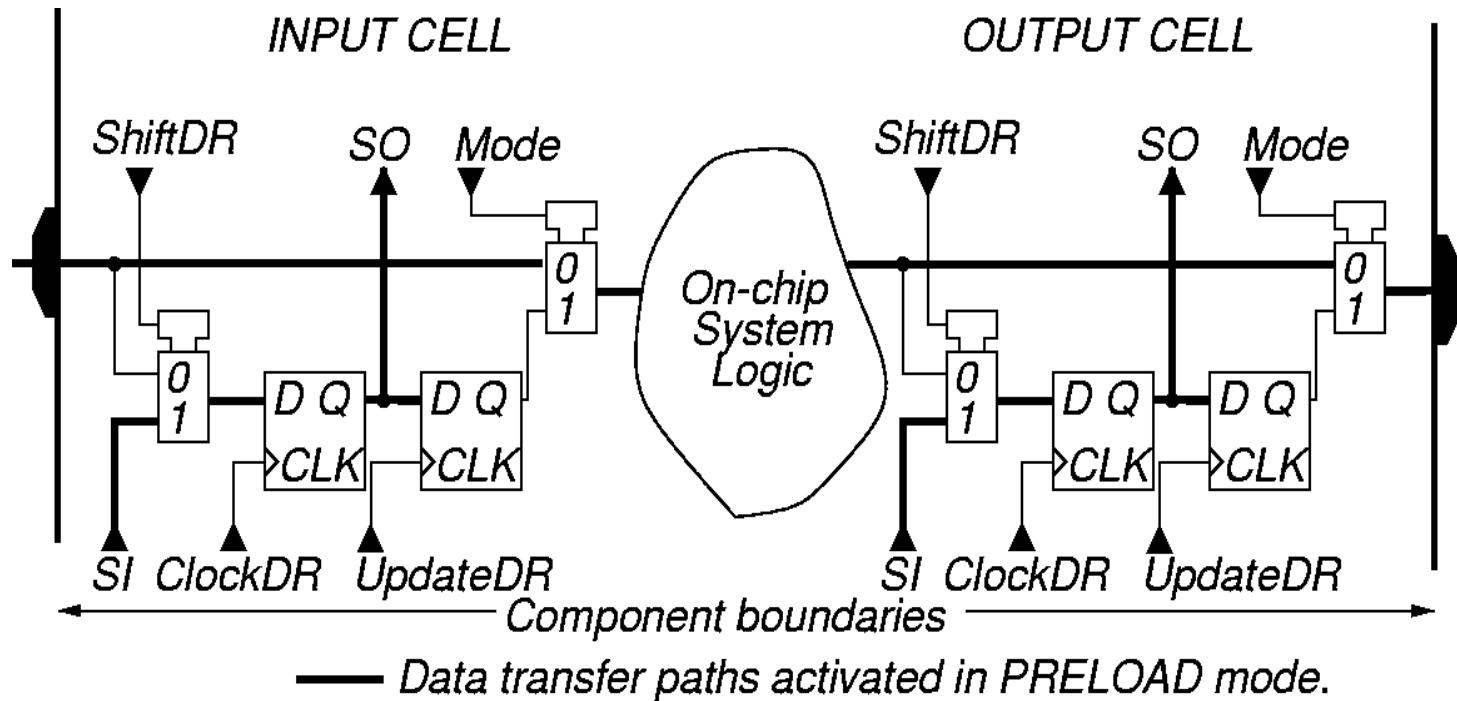
*Dohvaćanje stanja normalnih signala*



# Boundary Scan načini rada

## PRELOAD

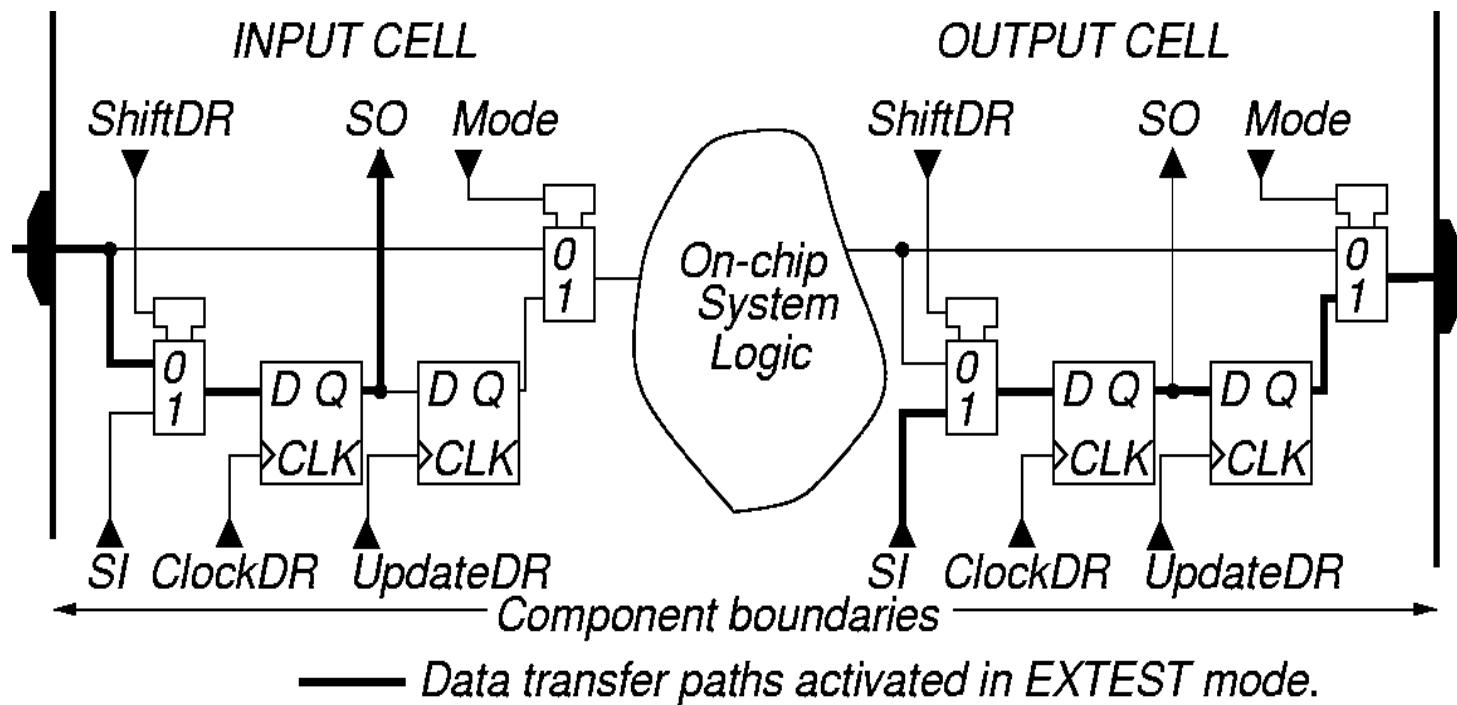
*Postavljanje podataka prije slijedećeg koraka*



# Boundary Scan načini rada

## Extest

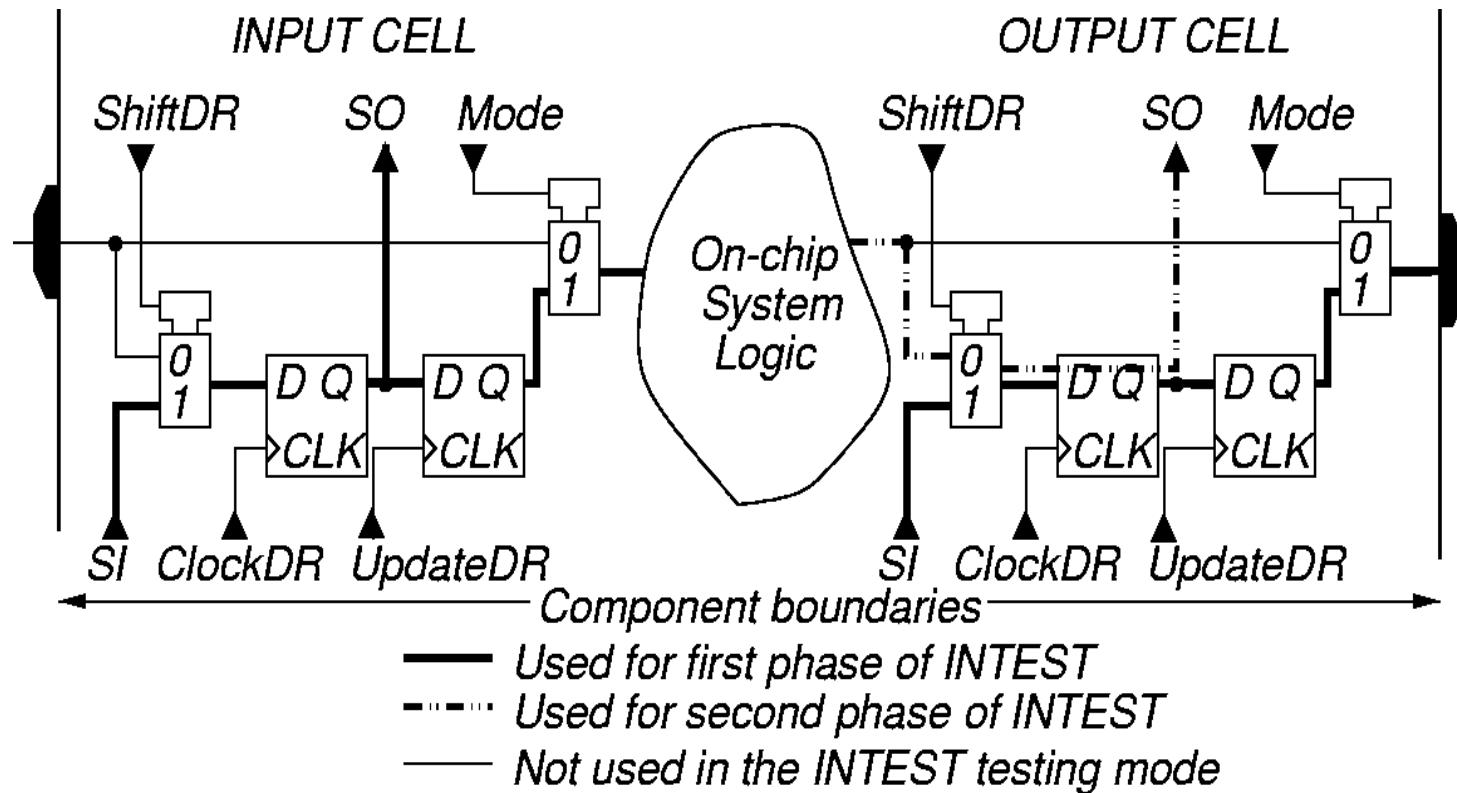
### Testiranje sklopova i veza



# Boundary Scan načini rada

## INTEST

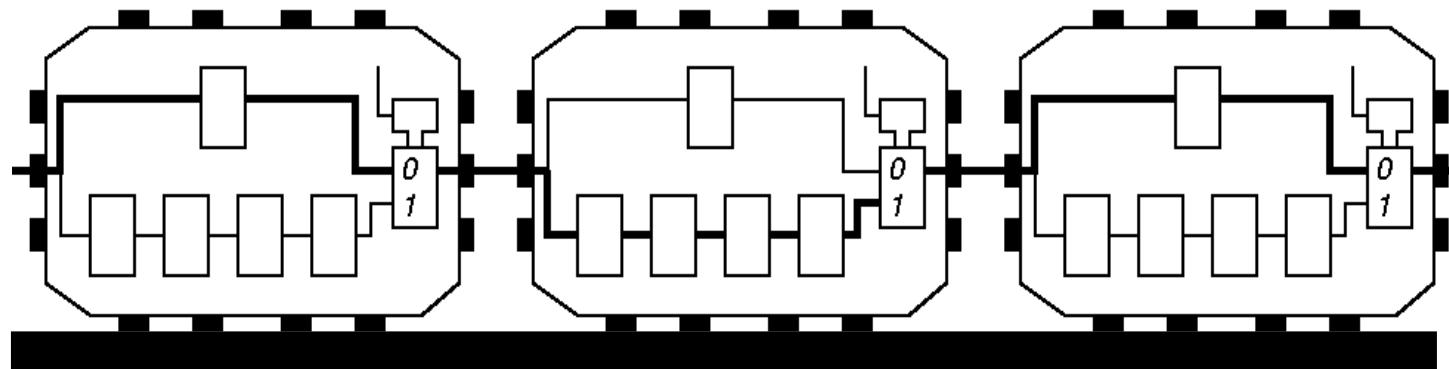
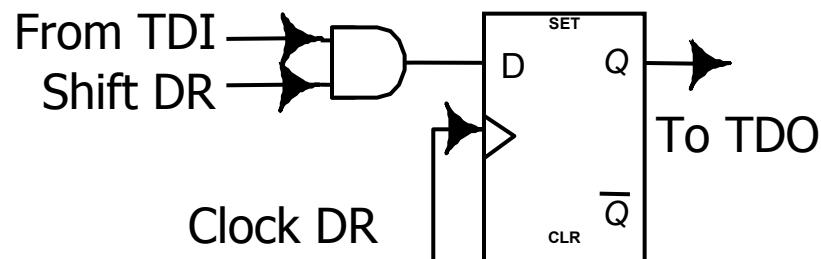
**Upis testnih podataka i očitanje stanja**



# Boundary Scan načini rada

## Bypass instruction

### Izolacija sklopa



# Samotestirajući skloovi

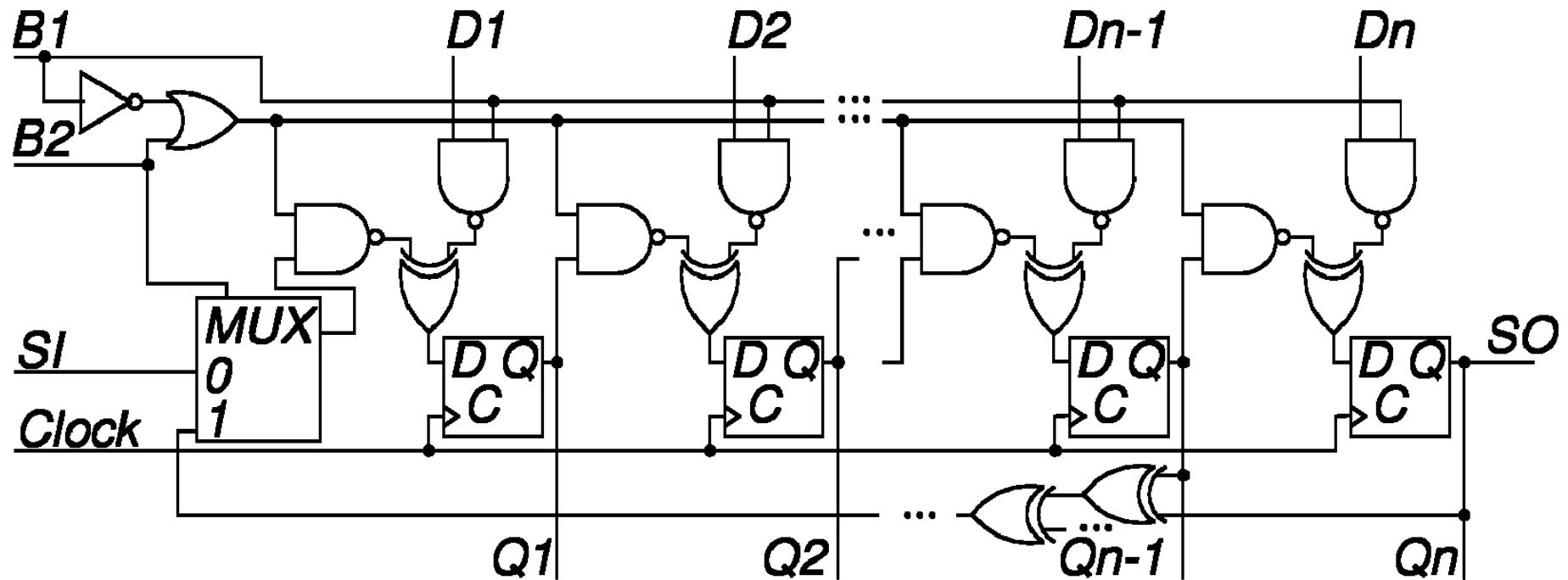
---

## ■ Osnovne tehnike:

- Promatranje bloka ugrađenom logikom
  - Built-in Logic Block Observer – BILBO
- Testiranje sindroma
- Autonomno testiranje

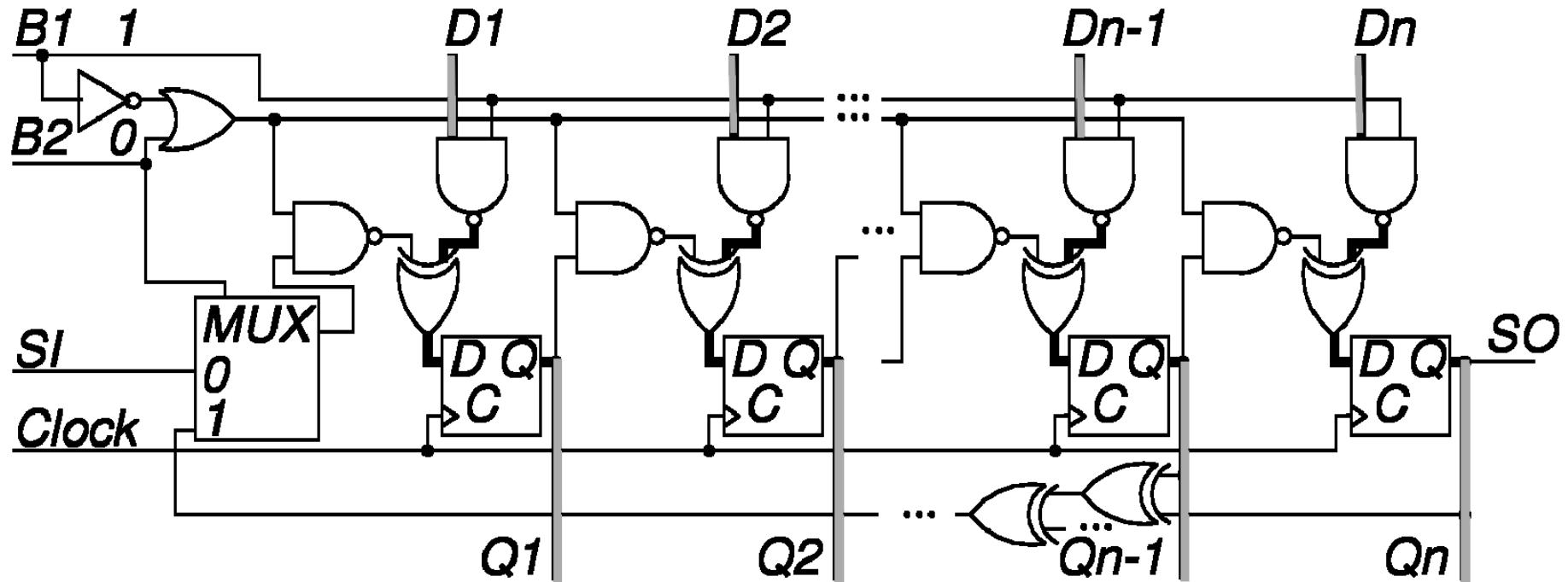
# Promatranje bloka ugrađenom logikom

- engl. Built-in Logic Block Observer
- Obuhvaća funkcionalnosti bistabila, generatora uzorka, analize odziva i lanca ispitivanja
- Struktura BILBO registra:



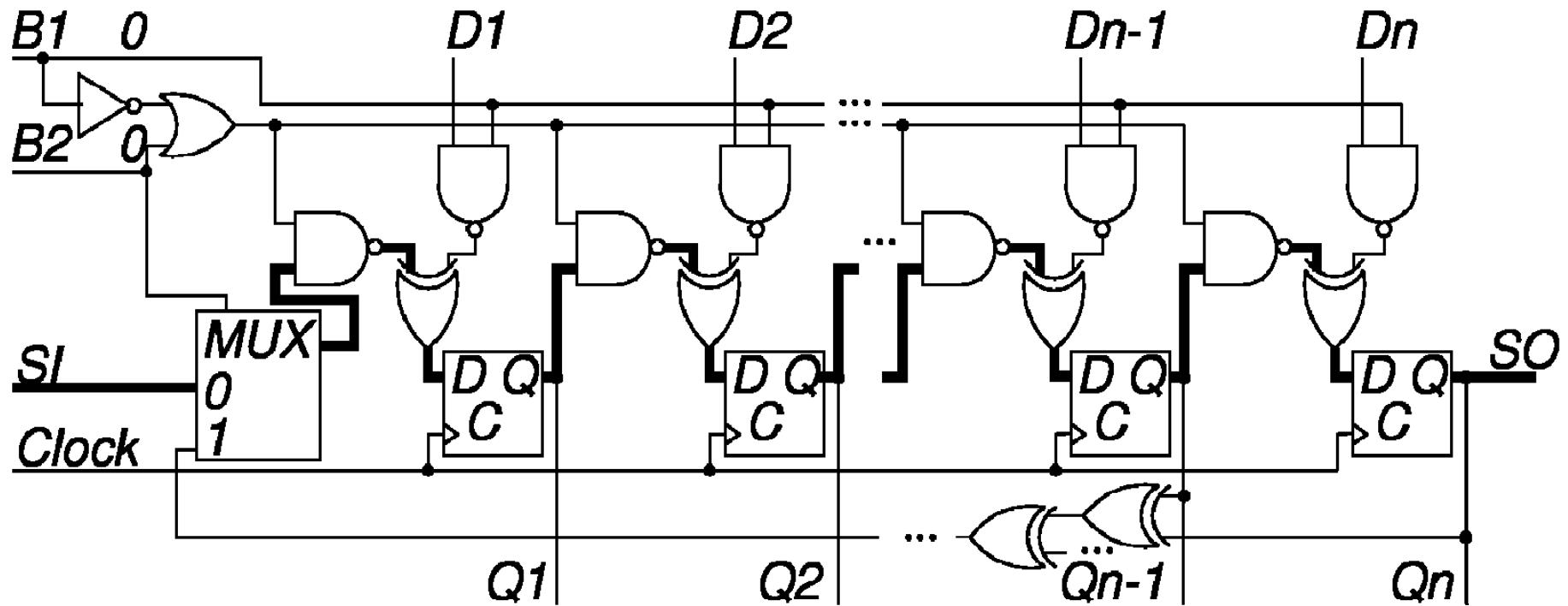
# Normalan rad - BILBO

- $B_1 B_2 = 10$



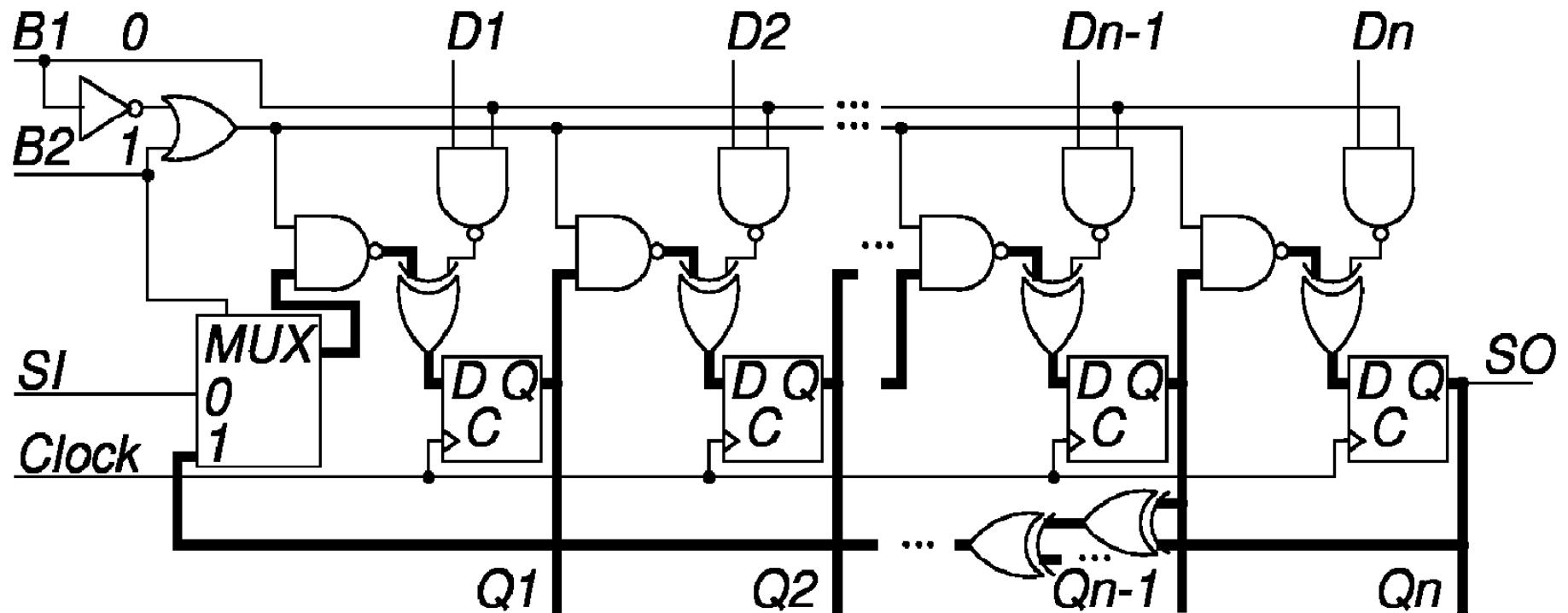
# Ispitni rad BILBO

- $B_1B_2 = 00$
- Upis podataka u bistabile/čitanje



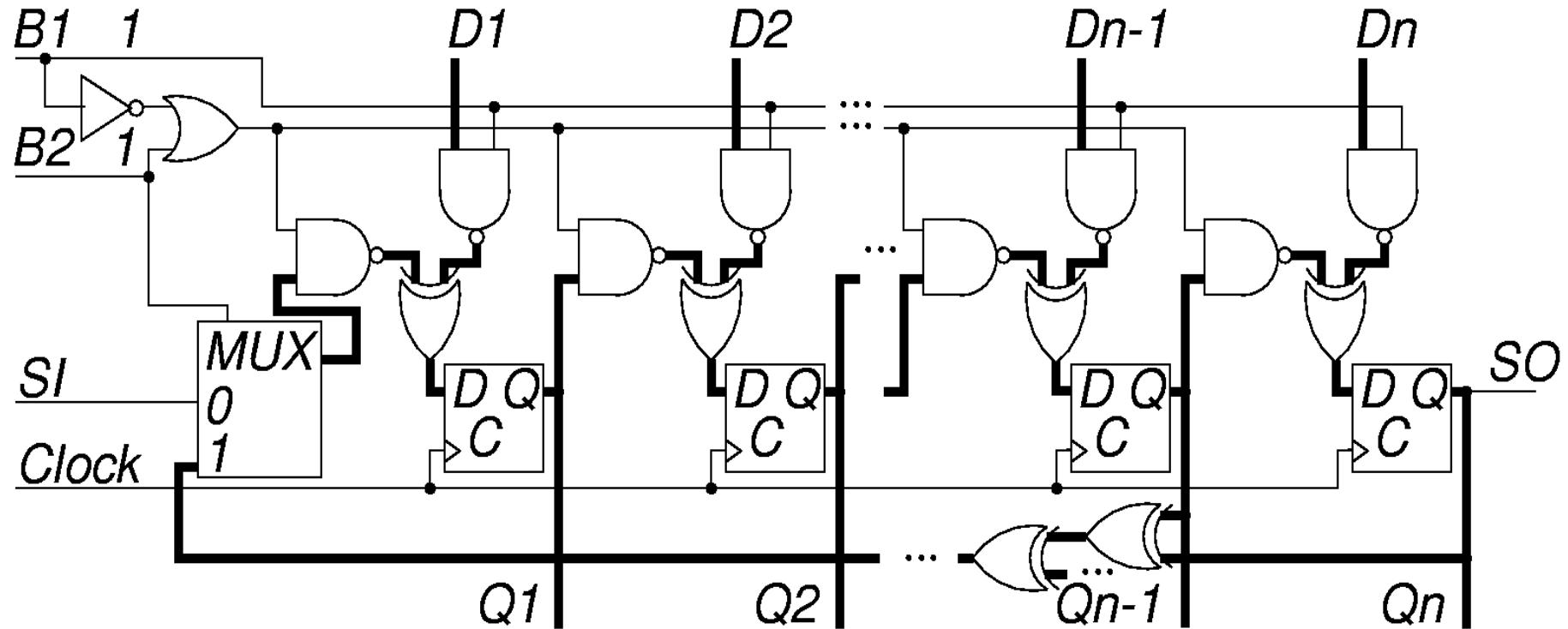
# Generiranje uzorka -BILBO

- $B_1 B_2 = 01$



# MISR način rada BILBO

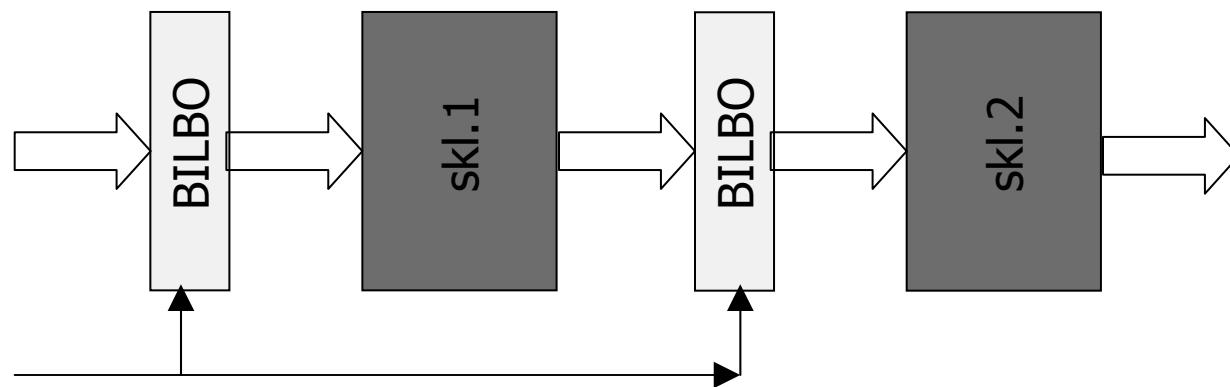
- Testiranje signature višestrukih ulaza (engl *multiple-input signature register* – MISR)
- $B_1B_2 = 11$



# Primjer

---

## ■ Kako testirati sklopove 1 i 2?



# Testne metode

---

- Iscrpno testiranje
  - Sve kombinacije ulaza
- Funkcijsko testiranje
  - Najčešće za verifikaciju
  - Mali broj algoritama za verifikaciju svih funkcija
- Strukturni vektor
  - U obzir uzima model grešaka
  - Uporaba EDA alata za automatsko generiranje

# Generator uzoraka

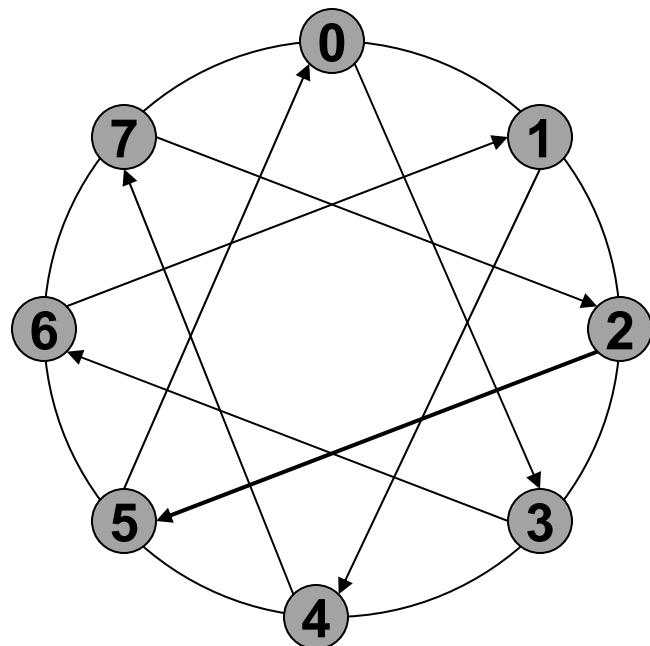
---

- Više implementacija:
- Memorija
  - RAM ili ROM s pohranjenim uzorcima
- Brojilo
- Pseudoslučajni generator uzoraka
  - Posmačni registar s povratnom vezom (engl. Feedback shift register)
  - Stanični automati (engl. Celluar automata)

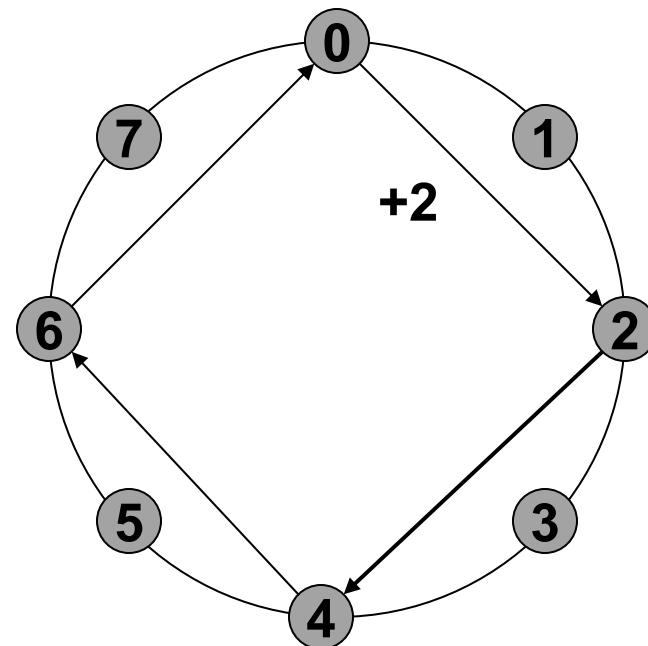
# Pseudoslučajni brojevi

---

$$X_k = X_{k-1} + 3 \text{ (modulo 8)}$$



$$X_k = X_{k-1} + 2 \text{ (modulo 8)}$$

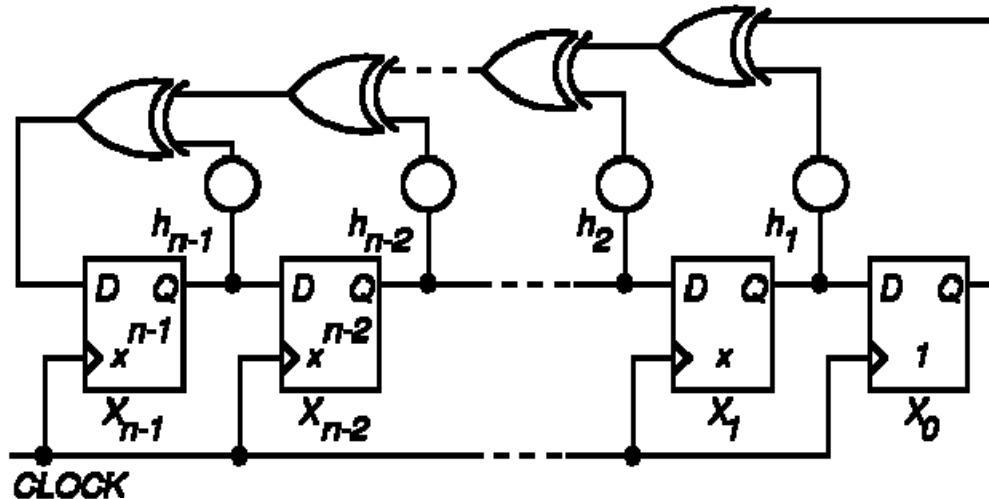


sekvenca: 0, 3, 6, 1, 4, 7, 2, 2, 5, 0 ...

sekvenca: 0, 2, 4, 6, 0, 2 ...

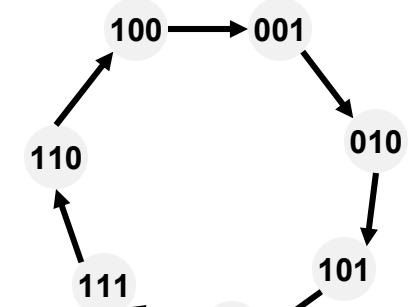
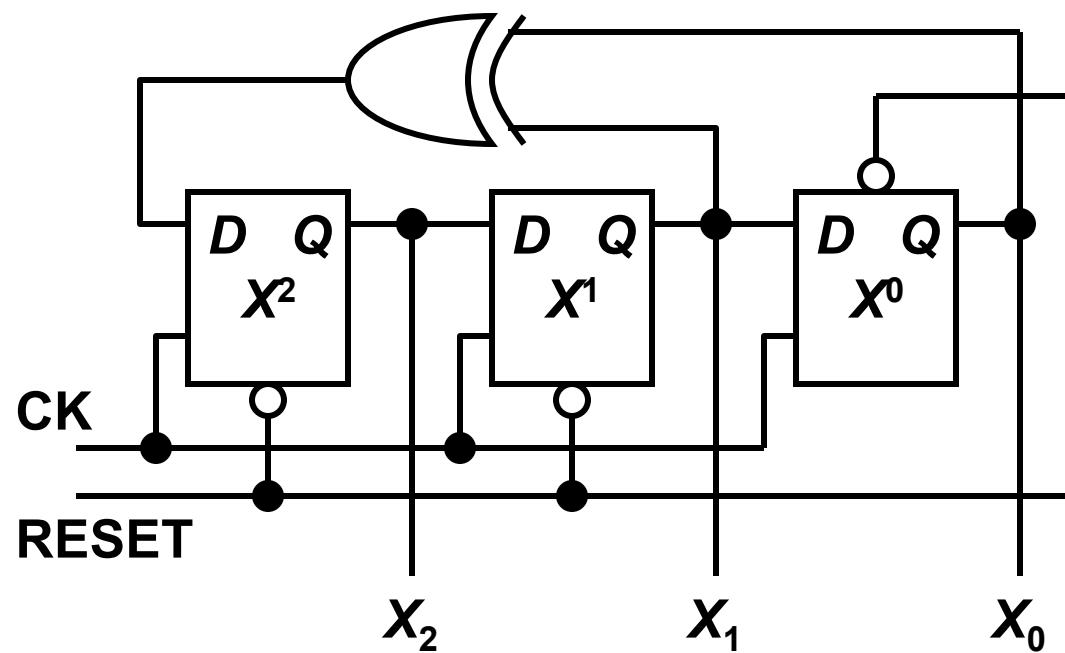
# Generator pseudoslučajnih brojeva

- engl. *Pseudo-Random Sequence Generator*
- *Linearni posmačni registar s povratnom vezom*
  - *Standard Linear Feedback Shift Register (LFSR)*
- Početno stane  $\leftrightarrow 0$
- Duža sekvenca  $\Rightarrow$  bolje pokrivanje



# Primjer: Polinom $1 + X + X^3$

---



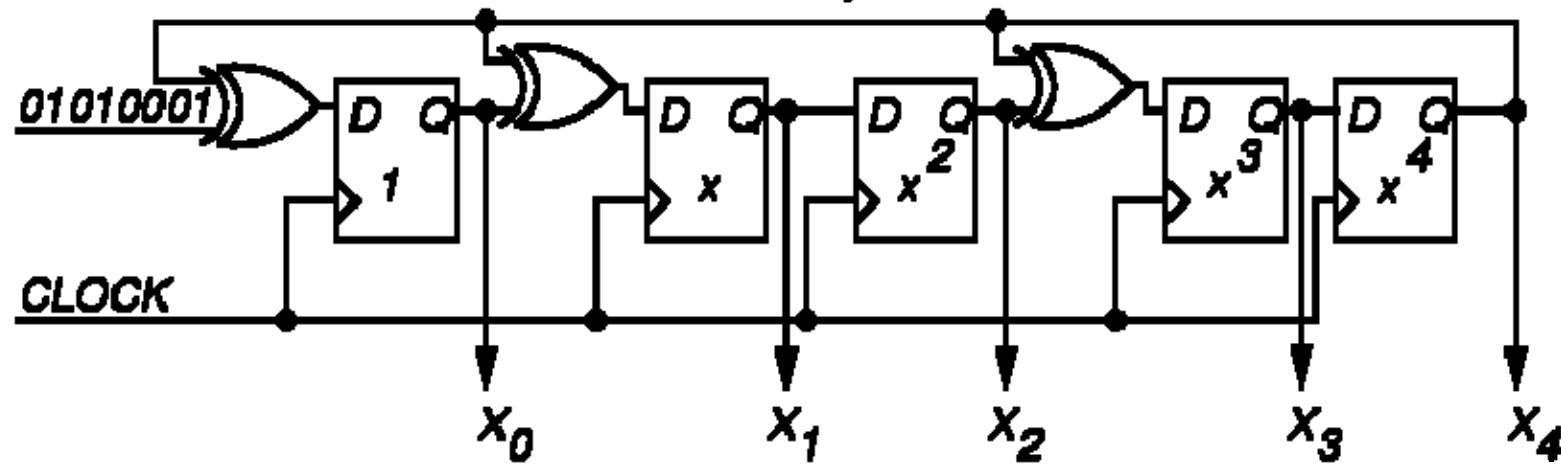
# Analiza odziva

---

- **Uporaba cikličkih kodova za zbijanje odziva (engl. response compacter)**
- ***LFSR kao generator CRC koda -CRCC***
- **Idea:**
  - Podatke s izlaza zbiti prema padfajućem slijedu koeficijenata polinoma
  - CRCC dijeli izlazni polinom s karakterističnim polinomom
  - Ostatak dijeljenja pohranjen u LFSR
  - Prije testiranja neophodno inicializirati LFSR na vrijednost sjemena
- **Nakon testiranja uspoređuje se vrijednost LFSR-a izračunatoj vrijednosti ispravnog sklopa**

# Primjer

■  $x^5 + x^3 + x + 1$



Ulagani bitovi	$x^0$	$x^1$	$x^2$	$x^3$	$x^4$
Poč. 0	0	0	0	0	0
1	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
1	0	0	0	0	1
0	1	0	0	1	0
1	1	1	0	0	1
0	1	0	1	1	0

# Izračun dijeljenjem polinoma

---

ulaz: 0 1 0 1 0 0 0 1

$$0 \cdot x^0 + 1 \cdot x^1 + 0 \cdot x^2 + 1 \cdot x^3 + 0 \cdot x^4 + 0 \cdot x^5 + 0 \cdot x^6 + 1 \cdot x^7$$

Karakteristični polinom:  $x^5+x^3+x+1$

$$\begin{array}{r} x^2 + 1 \\ \hline x^7 & + x^3 & + x \\ x^7 & + x^5 & + x^3 & + x^2 \\ \hline x^5 & & + x^2 & + x \\ x^5 & + x^3 & & + x + 1 \\ \hline x^3 & + x^2 & & + 1 \end{array}$$

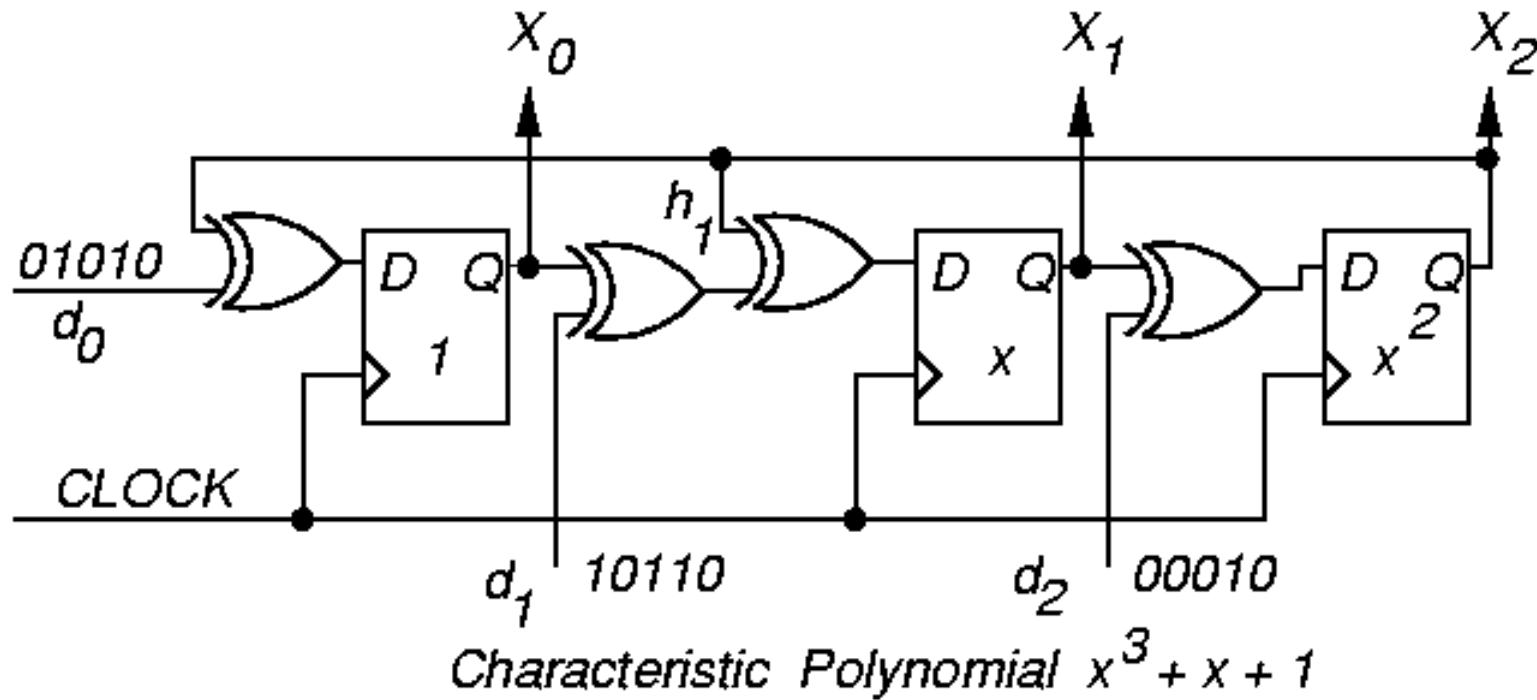
Signature:  $x^0 x^1 x^2 x^3 x^4 = 1 0 1 1 0$

# Poboljšani zbijeni odziva

---

- **Multiple-Input Signature Register (MISR)**
- **Zbijanje svih izlaza u jednom LSFR-u**
  - **Princip superpozicije**
  - **Sve odzive staviti u LSFR**
  - **Konačni ostatka je XOR suma ostataka dijeljenja polinoma s karakterističnim polinomom**

# Primjer MISR



$$\begin{bmatrix} X_0(t+1) \\ X_1(t+1) \\ X_2(t+1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_0(t) \\ X_1(t) \\ X_2(t) \end{bmatrix} + \begin{bmatrix} d_0(t) \\ d_1(t) \\ d_2(t) \end{bmatrix}$$

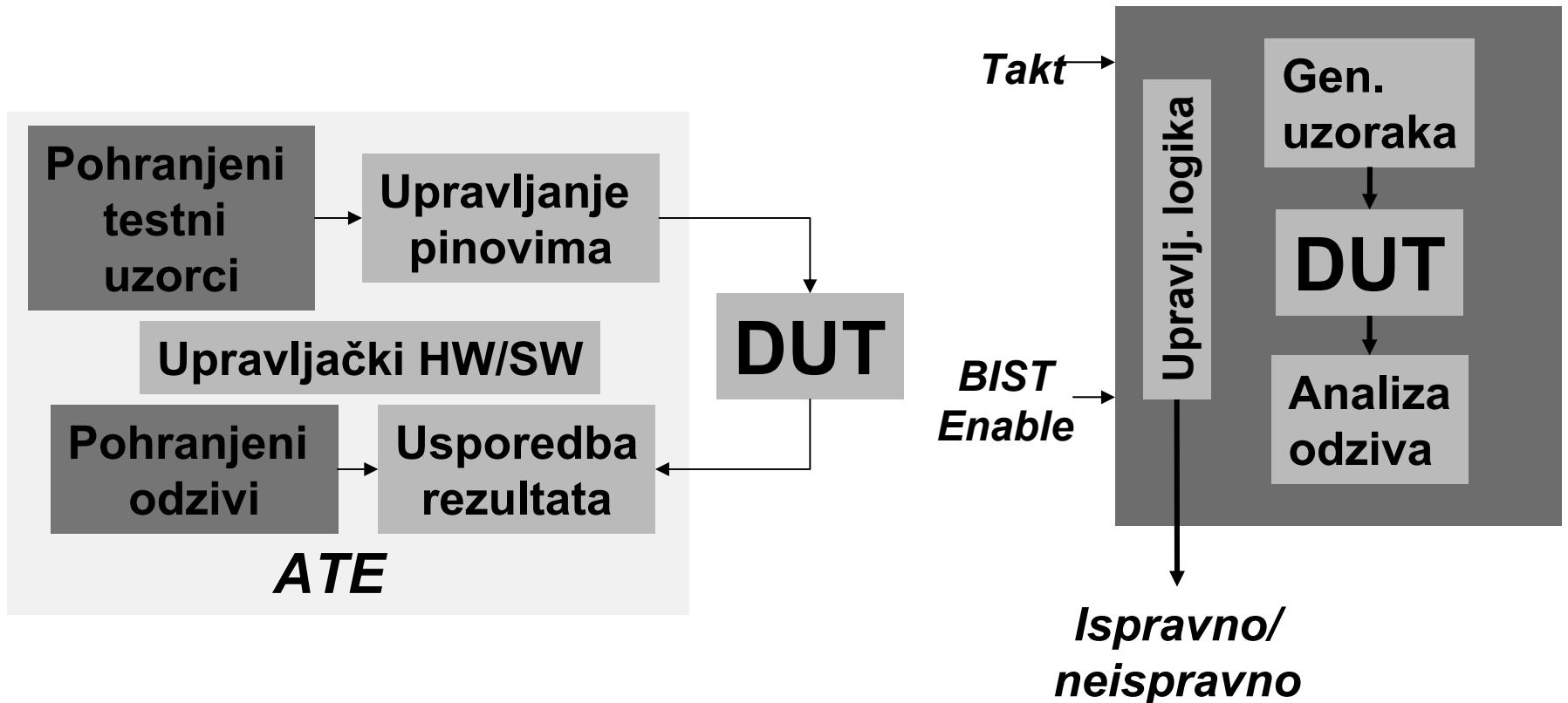
# Ugrađeno samotestiranje

---

- engl. Built-In Self-Test - BIST
- Implementacija automatiziranog testnog uređaja u samom sklopu kojeg testiramo
- Osnovni elementi:
  - Generator uzorka (eng. Pattern Generator)
    - LFSR
  - Analizator odziva (engl. Response Analysis)
    - MISR
    - Aliasing probability
  - Upravljanje ispitivanjem (engl. Test Controller)

# Struktura BIST-a

- Sklop kojeg testiramo (engl. Circuit Under Test CUT, Design Under Test DUT)



# Ugrađena samoprovjera

---

- Built-In Self Test BIST
- On-line
  - paralelno tijekom rada (concurrent)
    - Istovremeno uporabom tehnika kodiranja
  - normalan rad blokiran (nonconcurrent)
    - Samo u stanje neaktivnosti/čekanja (uvjek se može prekinuti)
- Off-line
  - Čip nije u funkciji – testni način rada
  - On-chip test pattern generators TPG
  - Output Response Analizers

# Strategija izbora

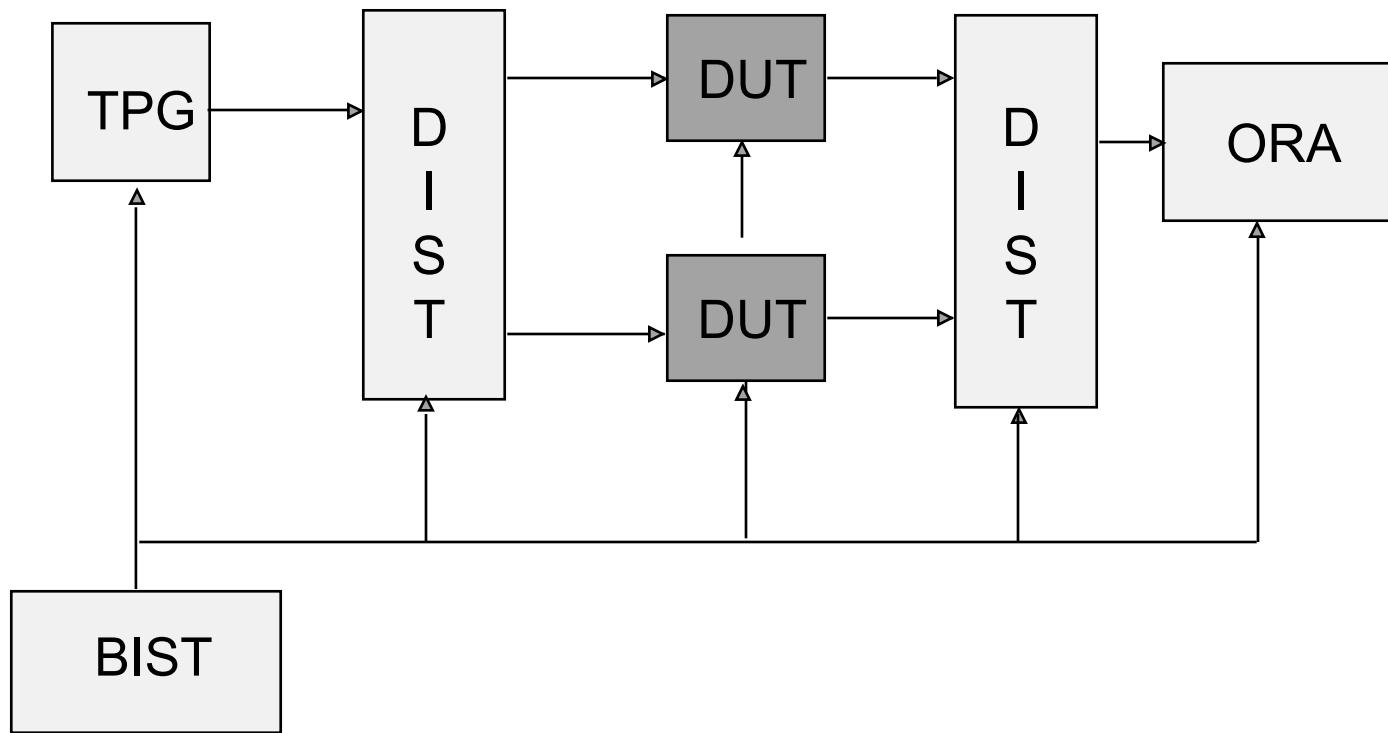
---

- Potrebni paralelizam
- Pokrivanje grešaka
- Zauzeće prostora
- Vrijeme testiranja
- Složenost
- Zahtjevi testiranaja u proizvodnji i prilikom rada
- Utjecaj na performanse
- Prednosti:
  - Niža cijena testiranja
  - Pokrivanje grešaka
  - Skraćivanje vremena
  - Cijeloživotno testiranje

# Arhitektura BIST

## ■ Poveznica sustava

### ■ DIST Distribution system for data transmission

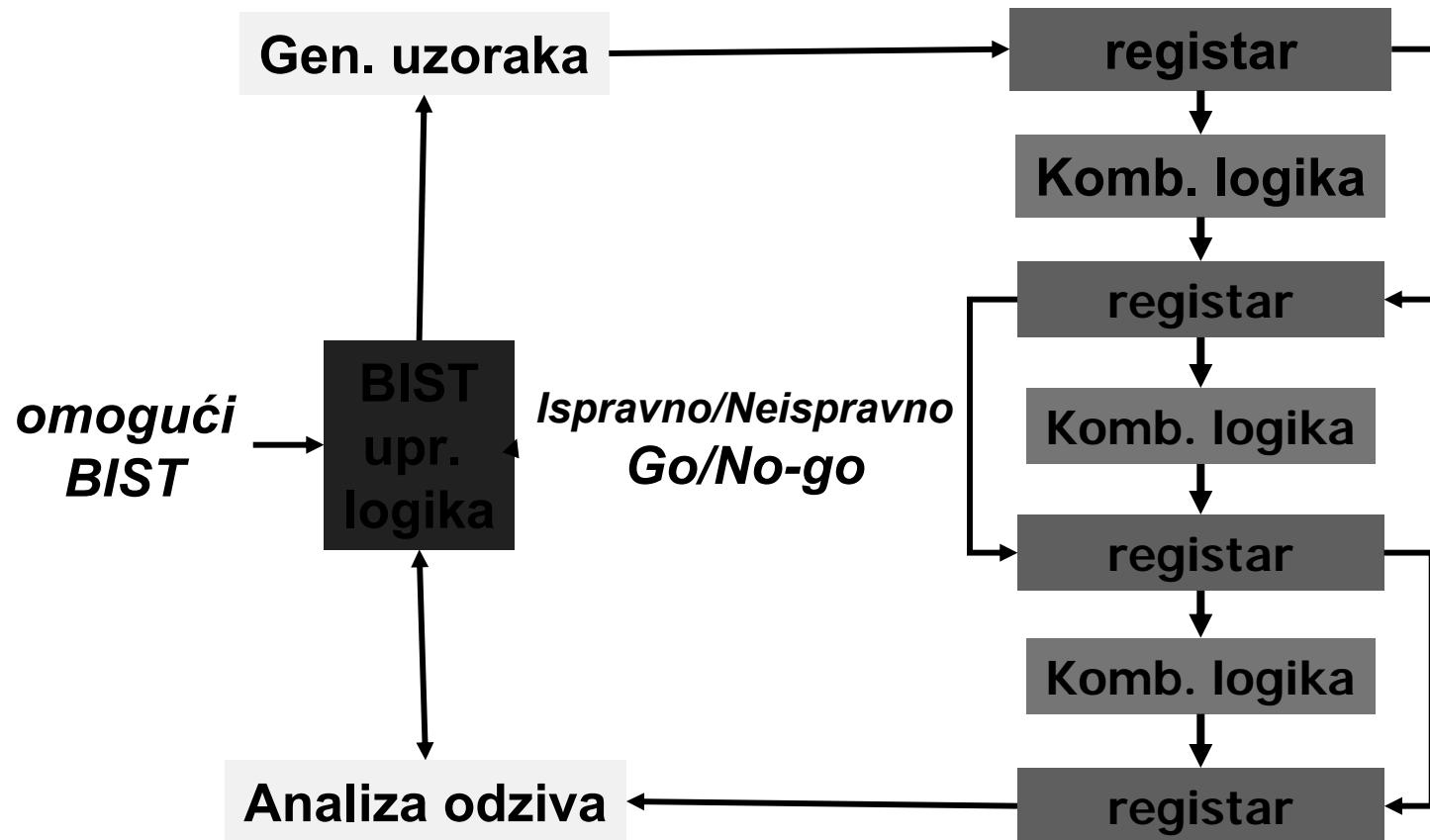


# Arhitektura BIST sklopova

---

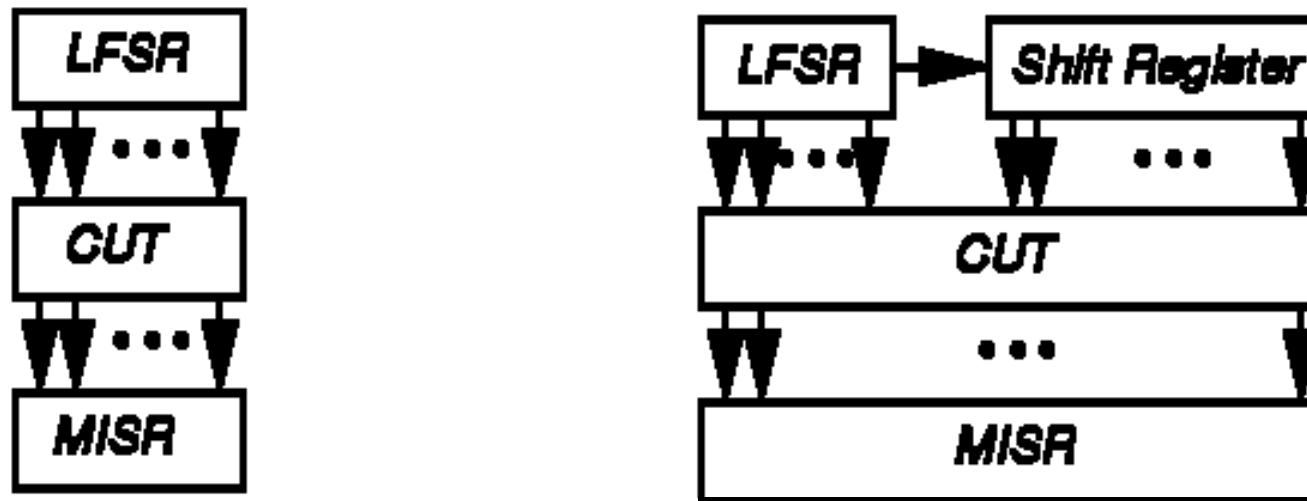
- Testiranje po jednom ispitivanju (engl. Test per scan)
- Testiranje u taktu (engl. Test per clock)
- Kružno samotestiranje (engl. Circular self-test)
- Ugrađeno samotestiranje memorija (engl. Memory BIST)

# Testiranje po jednom ispitivanju

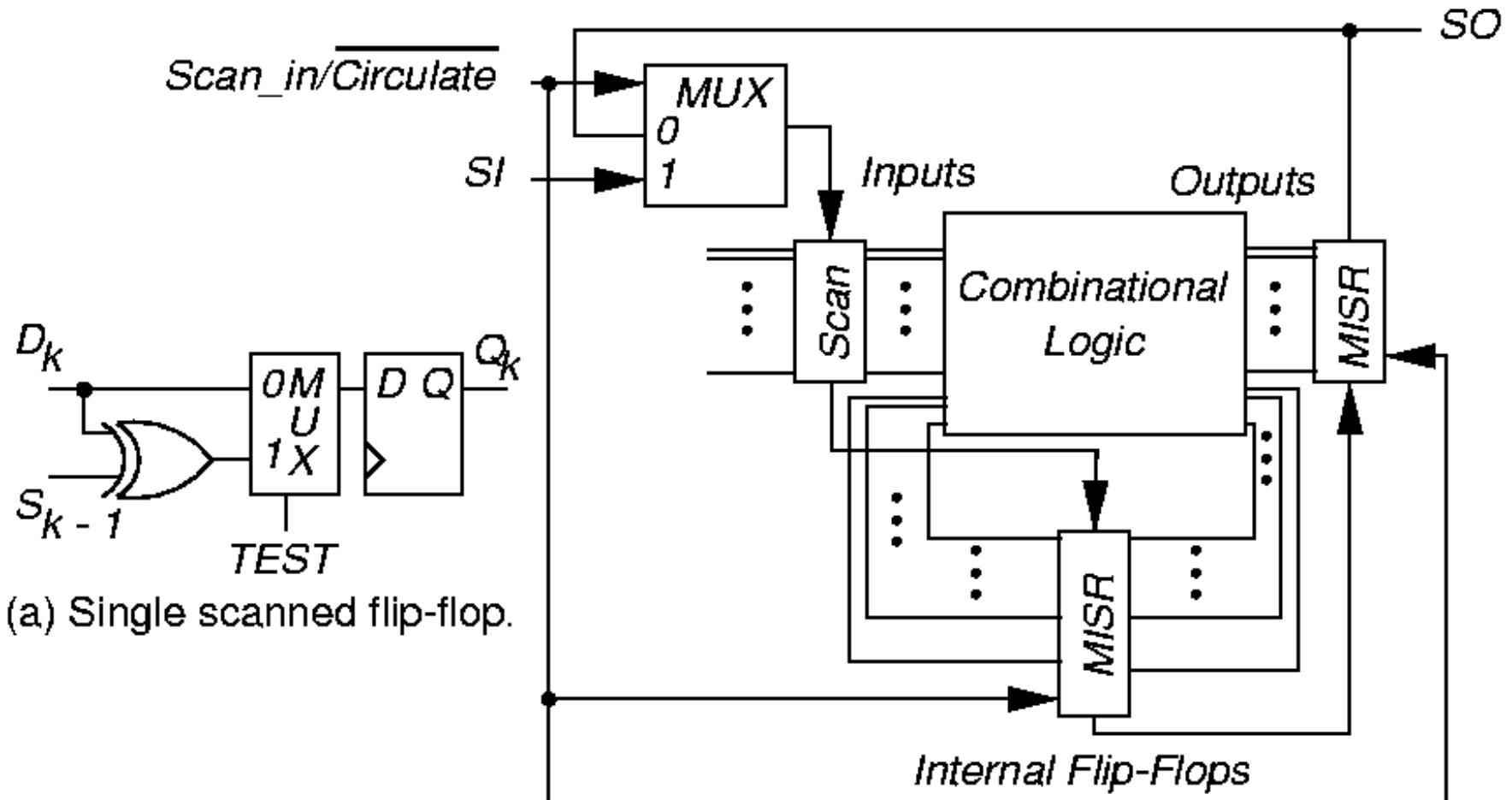


# Testiranje u taktu

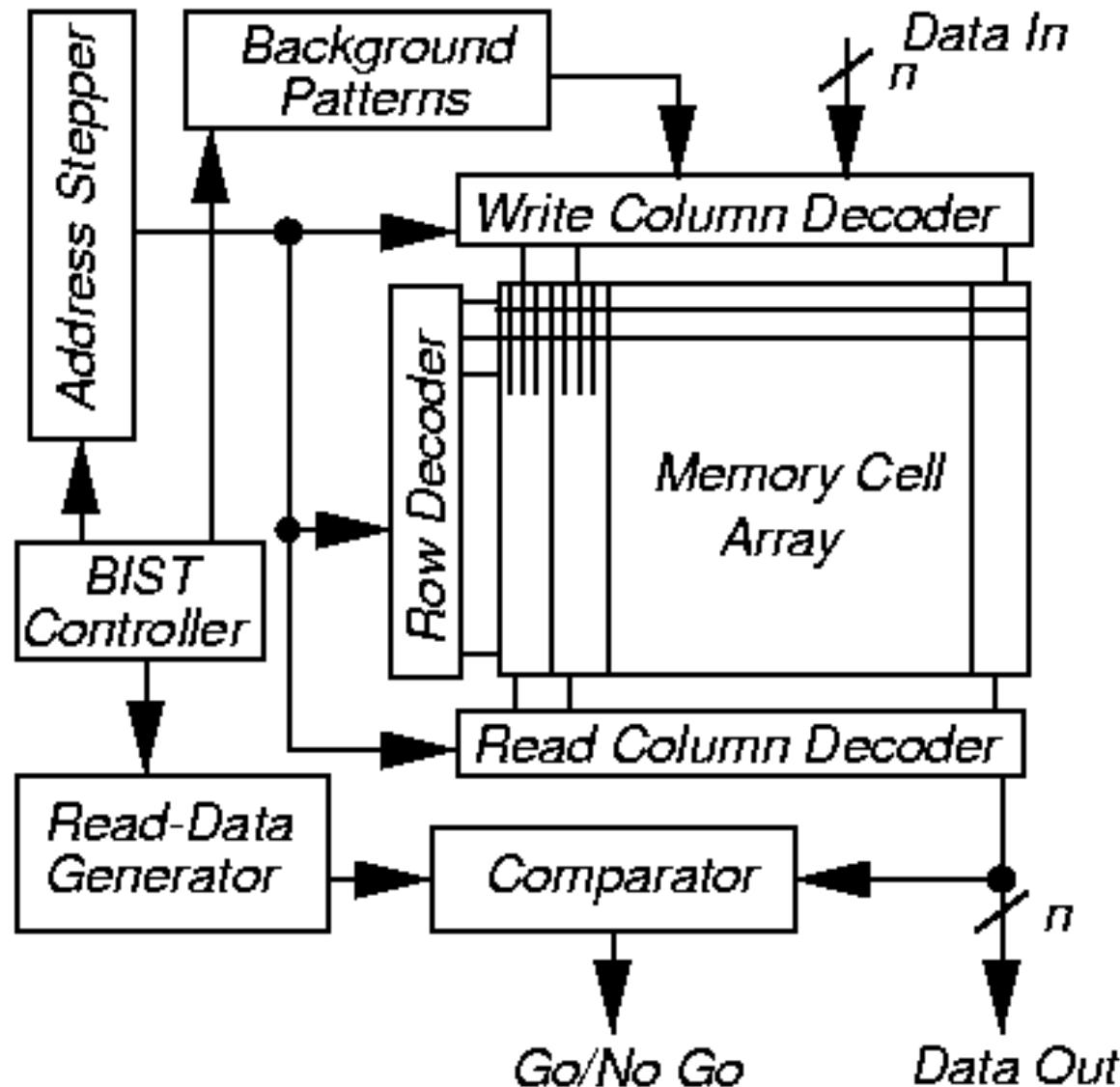
- Ispitivanje jednog skupa u svakom taktu
  - Najkraće moguće tetiranje
  - Npr. Takt 200 MHz,  $10^7$  BIST uzoraka
    - $Vrijeme testiranja = 10,000,000 / 200 \times 10^6 = 0.05 \text{ s}$



# Kružno samotestiranje

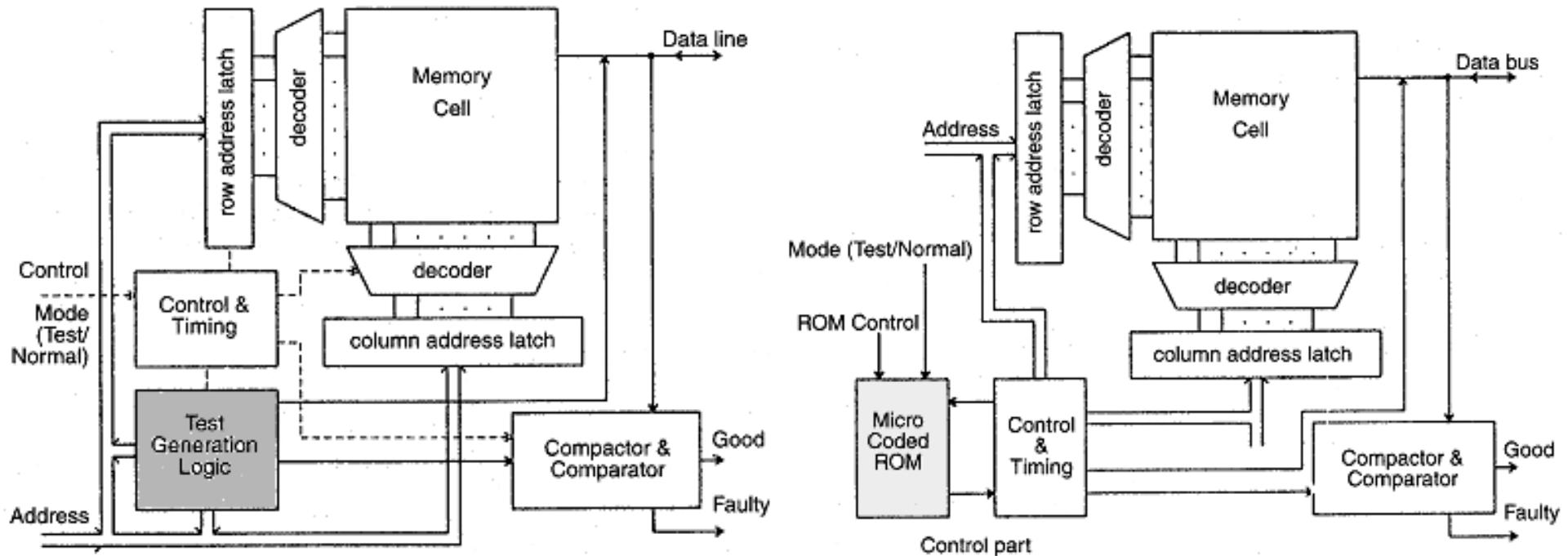


# Memory BIST

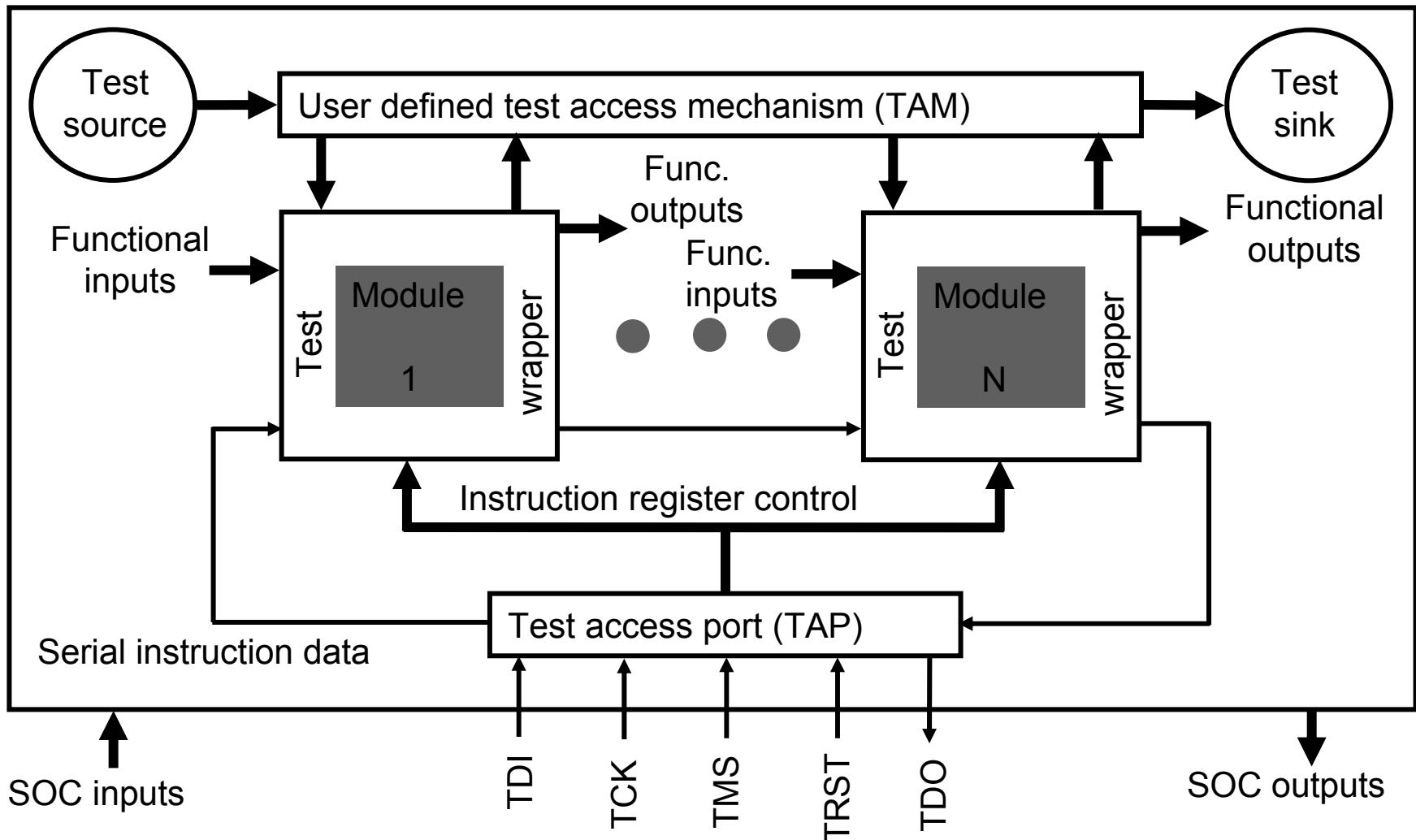


# Ugrađena samoprovjera i popravak

- Built-In Self Repair BISR
- Mb memorije koriste kompenzaciju pokvarenih celija

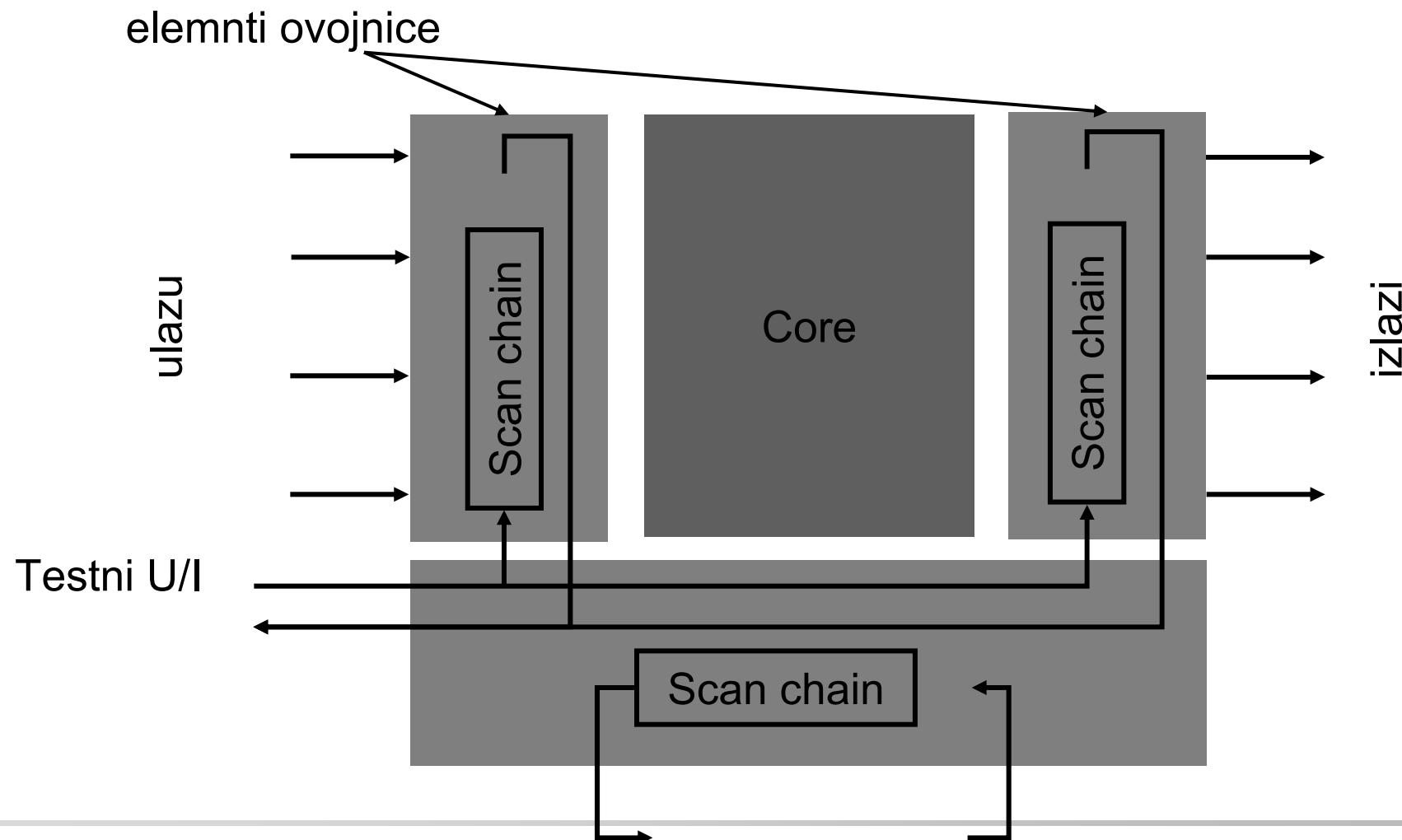


# Primjena DFT u SOC



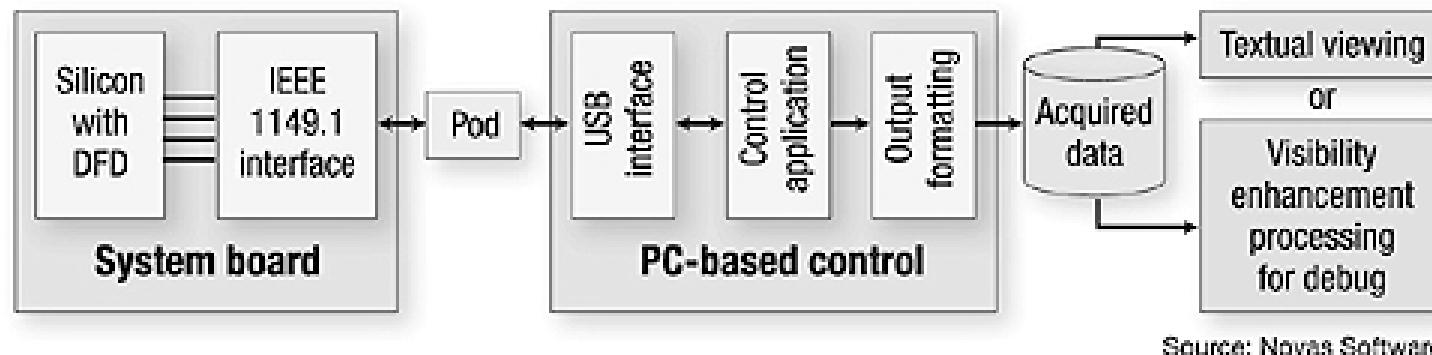
# Struktura ovojnica

## ■ engl. Wrapper



# Testiranje procesora

- Rast složenosti zahtjeva podršku ispravljanja grešaka i dijagnostike ugrađene programske podrške
- Dugotrajna podrška u mikroprocesorima
- Podrška ugrađene logike u sklopolju



# Pitanja...

# Pouzdanost računalnih sustava

---

**Predavanje 4: Metode vrednovanja pouzdanosti**

**Prof.dr.sc. Vlado Sruk**



**Sveučilište u Zagrebu**  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave*



# Sadržaj

---

- Relevantne statističke tehnike
- Metode evaluacije sustava
- Kombinatorni modeli
- Blok dijagram pouzdanosti

# Uvod

---

## ■ model

- apstrakcija koja obuhvaća ponašanje stvarnog sustava
  - jednostavnost
  - mora voditi do preciznih zaključaka o sustavu

## ■ vrednovanje:

- eksperimentalno vrednovanje
  - moguće je provesti u različitim fazama
- faza projektiranja (engl. design phase)
  - vrednovanje simulacijom
  - ubacivanje kvarova (engl. fault injection)
- kako provesti?
  - ulazni parametri
  - interpretacija rezultata

- 
- vrednovanje prototipa (engl. prototype phase)
    - kontrolirani radni uvjeti
    - fizičko ubacivanje kvarova
    - spoznaje o procesima zatajenja
    - ne određuje MTBF i raspoloživost
  - vrednovanje rada (engl. operational phase)
    - izravna mjerena pod stvarnim radnim opterećenjem
    - velika količina prikupljenih podataka
    - razumijevanje stvarnih svojstava kvarova i pogrešaka
    - ograničenost rezultata na otkrivene pogreške
  - za vjerodostojnu analizu sustava neophodna sva tri pristupa
-

# Analitički modeli

---

- procjena dizajna
- metrika za usporedbu različitih dizajna
- omogućava povratnu vezu projektantima u ranim fazama projektiranja sustava
- analiza performansi
- kvantitativna i kvalitativna analiza

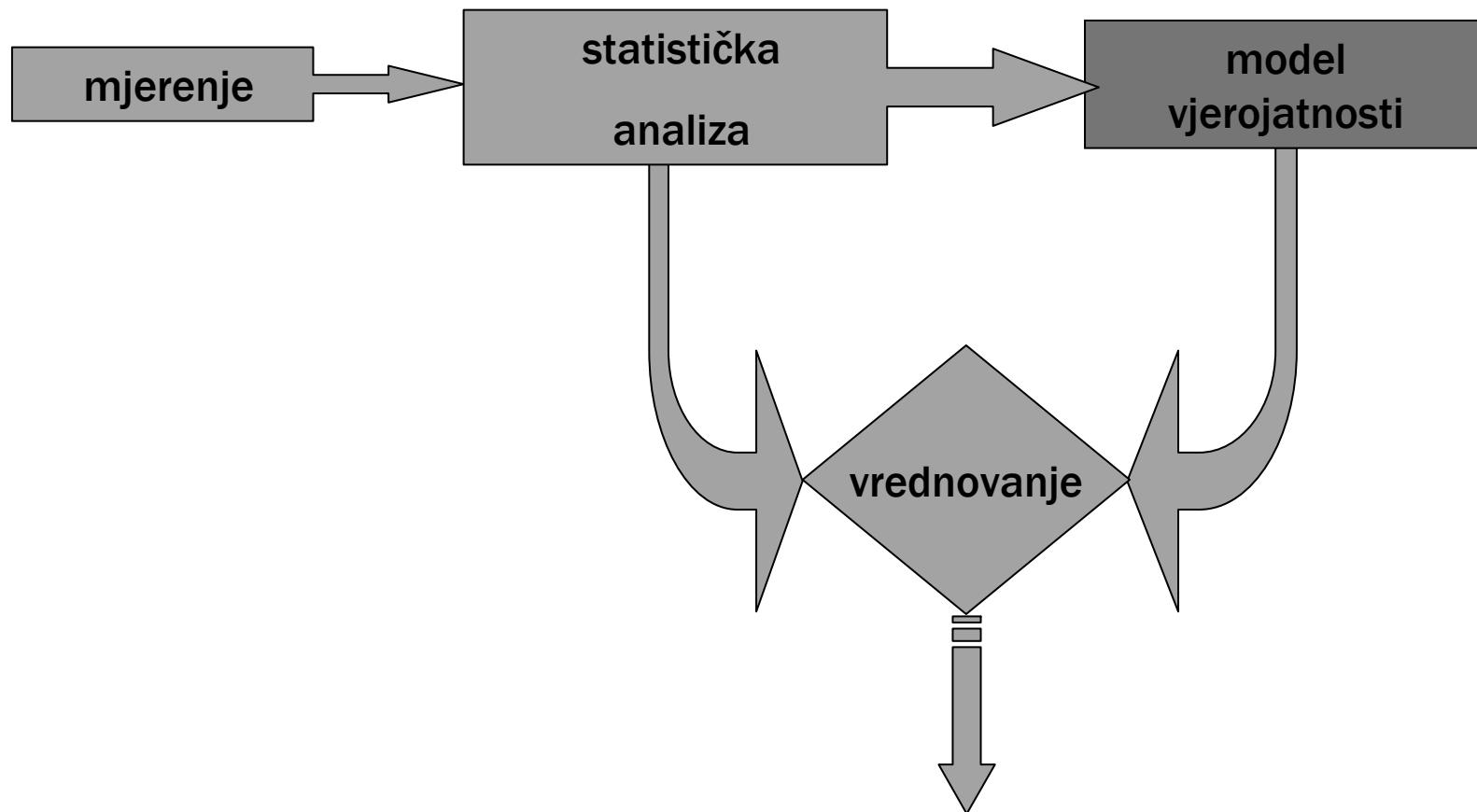
# Modeli zasnovani na vjerojatnosti

---

- teorija vjerojatnosti omogućava kvantitativnu procjenu oslonljivosti sustava (pouzdanost, raspoloživost, sigurnost...)
- slučajni događaji
- kvantitativno određivanje vjerojatnosti
  - modeli vjerojatnosti
- procjena kvantifikatora
- srednja vrijednost i varijanca
- distribucija vjerojatnosti

# Modeliranje slučajnih događaja

---



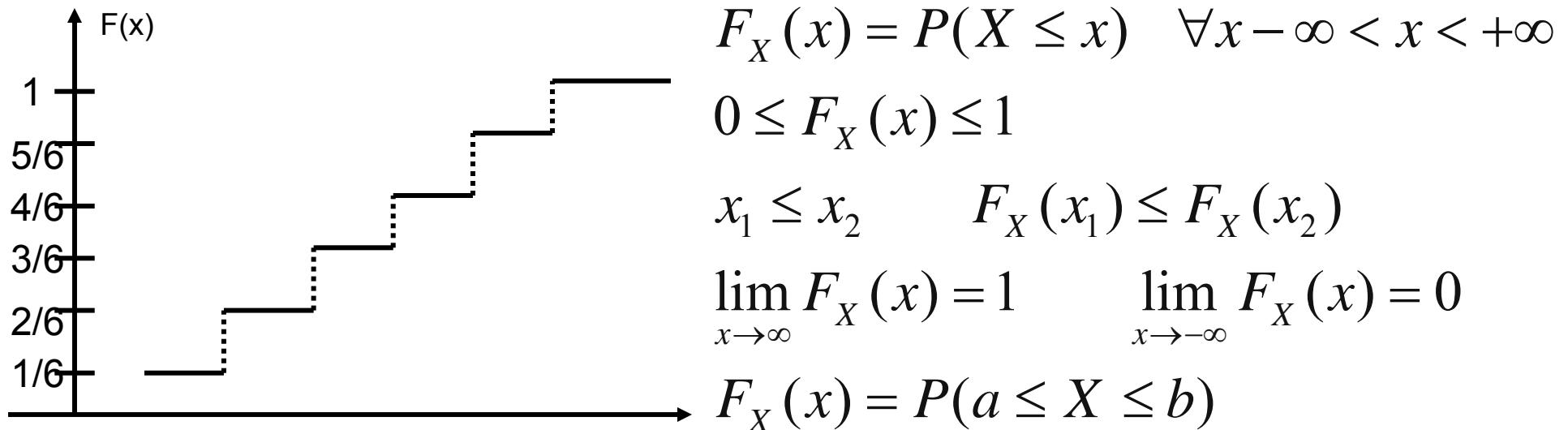
# Slučajna varijabla

---

- funkcija koja pridružuje vrijednost svakom mogućem događaju
  - diskretna
    - $X(x_1, x_2, x_3, x_4, x_5, x_6, x_7, \dots)$
  - kontinuirana

# Funkcija distribucije slučajne varijable

- (engl. Cumulative Distribution Function)
- $F(x)$  vjerojatnost  $P$  da se ostvari događaj  $X$  je jednaka  $x$  ili manja
- monotona rastuća funkcija



# Gustoća funkcije vjerojatnosti

---

## ■ engl. Probability Density Function

$$f_X(x) = \frac{dF_X(x)}{dx} \quad f_X(x) \geq 0$$

$$P(a \leq x \leq b) = \int_a^b f(x) dx$$

## ■ varijanca

$$\sigma_x^2 = E[(x - E(x))^2]$$

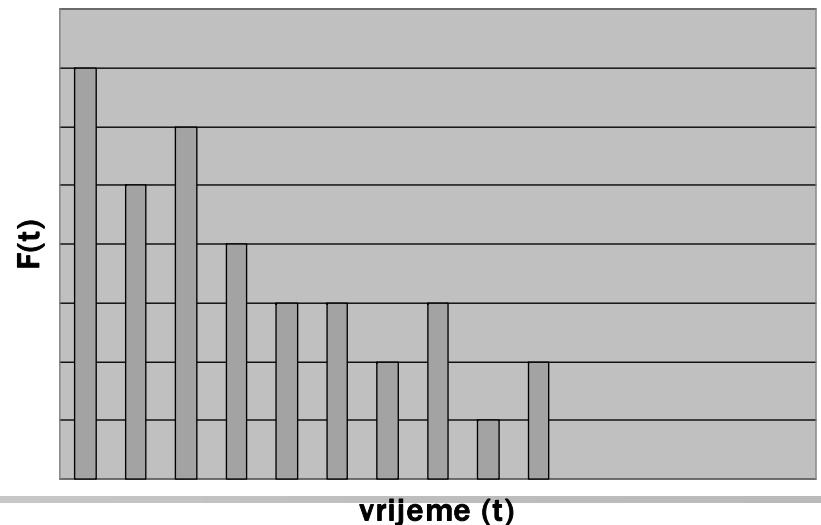
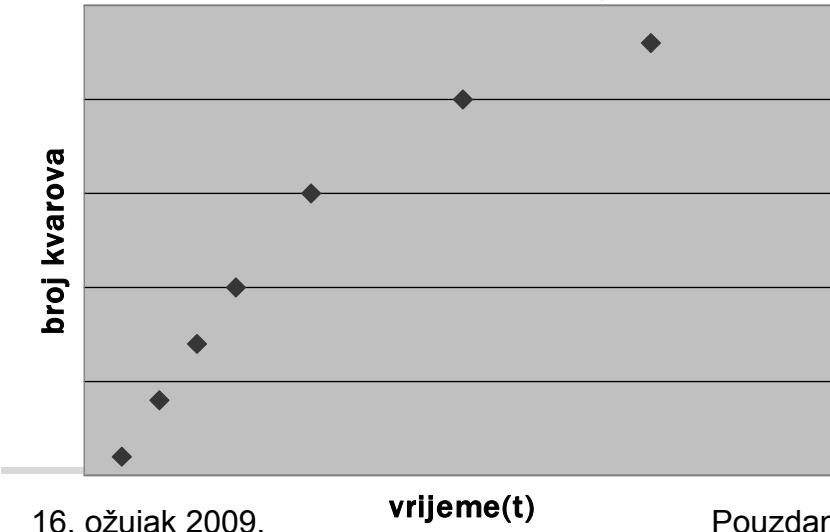
## ■ očekivanje

$$E(X) = \sum_{x_i} x_i f(x_i)$$

$$E(X) = \int_{-\infty}^{\infty} x f_X(x) dx$$

# Pouzdanost

- uvjetna vjerojatnost da sustav radi ispravno (unutar zadane specifikacije) unutar zadanog vremenskog intervala  $[0,t]$  i u zadanim radnim uvjetima
- eksperimentalno određivanje pouzdanosti
  - mjeri se vrijeme nastupanja kvara kod populacije od N identičnih sustava
    - stvarni rad
    - ubrzano starenje



# Eksperimentalna relativna gustoća kvara

---

- $N$  – broj promatralih sustava
- $n(t)$  – broj preživjelih na početku intervala
- $n(t+\Delta t)$  – broj preživjelih na kraju intervala
- $R_e(t)$  – vjerojatnost preživljavanja

$$f_e(t) = \frac{n(t) - n(t + \Delta t)}{N \Delta t}$$

$$R_e(t) = \frac{n(t)}{N}$$

$$\sum_{i=1}^m f_e(t) \Delta t = 1$$

---

# Intenzitet kvara

---

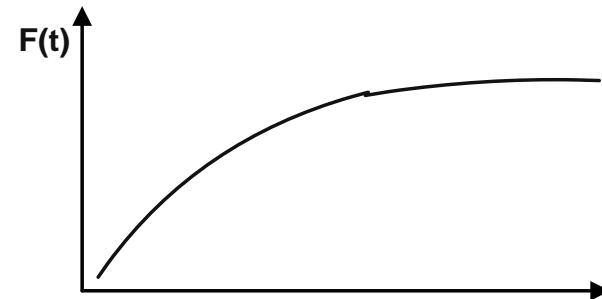
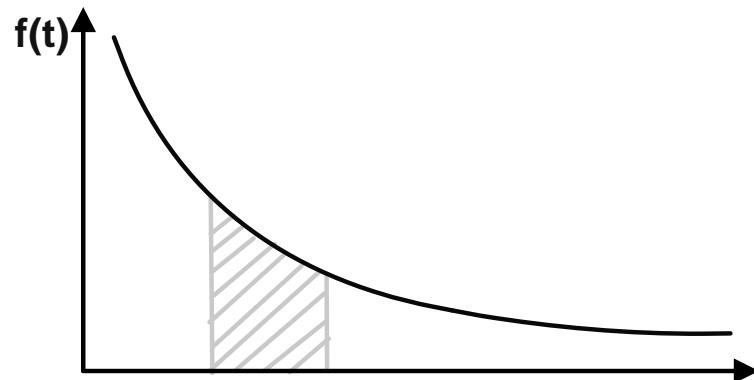
- Mjera rizika, hazard (engl. failure rate)
- $Z_e(t)$  – broj kvarova u jedinici vremena u odnosu na broj elementa na početku intervala

$$Z_e(t) = \frac{n(t) - n(t + \Delta t)}{n(t)\Delta t}$$

$$Z_e(t) = \frac{f_e(t)}{R_e(t)}$$

# Kontinuirani model

---



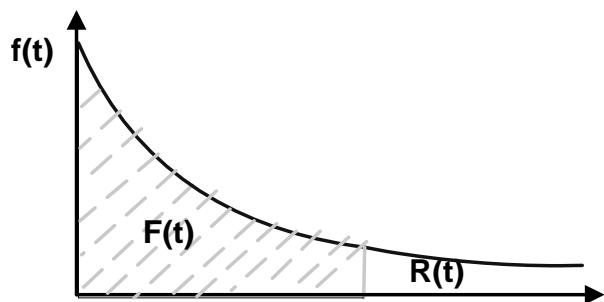
$$\int_a^b f(t)dt = 1$$

$$P(t) = P(t \leq t_k \leq t + \Delta t) = f(t)dt$$

$$F(t) = \int_0^t f(t)dt$$

# Funkcija pouzdanosti

- Vjerojatnost da ne dođe do kvara
- $R(t) = 1 - F(t) = P[T > t]$



- Mjera rizika  $z(t) = f(t)/(1 - F(t))$



# Srednje vrijeme do kvara MTTF

---

- Vrijeme očekivanja= moment prvog reda

$$\begin{aligned}MTTF &= E(t) = \int_0^t tf(t)dt \\&= \int_0^t t \frac{-dR(t)}{dt} dt = -\int_0^t t dR(t) \\&= -\left[ tR(t) \Big|_0^\infty - \int_0^t R(t)dt \right] = \int_0^\infty R(t)dt\end{aligned}$$

# Eksponencijalna distribucija

---

- $\lambda$ -učestalost kvara
- u stvarnosti je funkcija vremena

$$f(t) = \lambda e^{-\lambda t}$$

$$F(t) = 1 - e^{-\lambda t}$$

$$R(t) = e^{-\lambda t}$$

$$z(t) = \lambda$$

$$\mu(t) = 1/\lambda$$

$$\sigma(t) = 1/\lambda$$

# Weibull-ova distribucija

---

■  $\lambda$  - faktor rastezanja

■  $\alpha$  - faktor oblika

$\alpha=1$  eksp. distr.

$\alpha=2$  Raylieghova distr.

■  $\alpha < 1$  hazard pada s vremenom

■  $\alpha = 1$  hazard je konstantan

■  $\alpha > 1$  hazard raste s vremenom

■  $\mu < \sigma$  hazard pada s vremenom

■  $\mu = \sigma$  hazard je konstantan

■  $\mu > \sigma$  hazard raste s vremenom

$$f(t) = \alpha \lambda (\lambda t)^{\alpha-1} e^{-(\lambda t)^\alpha}$$

$$F(t) = 1 - e^{-(\lambda t)^\alpha}$$

$$R(t) = e^{-(\lambda t)^\alpha}$$

$$z(t) = \alpha \lambda (\lambda t)^{\alpha-1}$$

$$\Gamma(\varpi) = \int_0^{\infty} \rho^{\varpi-1} e^{-\rho} d\rho$$

$$\mu = \frac{\Gamma(\alpha+1)/\alpha}{\lambda}$$

$$\sigma = \sqrt{\frac{\Gamma(\alpha+2)}{\alpha} - \frac{\Gamma^2(\alpha+1)}{\alpha}}$$

# Geometrijska distribucija

---

## ■ iz eksponencijalne

- $e^{-\lambda} \rightarrow q$
- $t \rightarrow n$

$$f(n) = q^n - q^{(n+1)}$$

$$F(n) = 1 - q^n$$

$$R(n) = q^n$$

$$z(t) =$$

$$\mu = \frac{1}{1-q}$$

$$\sigma = \frac{\sqrt{q}}{1-q}$$

# Diskretna Weibull-ova distribucija

---

- $e^{-\lambda\alpha} \rightarrow q$
- $t \rightarrow n$

$$f(n) = q^{n^\alpha} (1 - q^{(n+1)^\alpha - n^\alpha})$$

$$F(n) = 1 - q^{n^\alpha}$$

$$R(n) = q^{n^\alpha}$$

$$z(t) = 1 - q^{(n+1)^\alpha - n^\alpha}$$

$$\mu = \sum_{k=0}^{\infty} q^{k^\alpha}$$

# Empirička formula za $\lambda$

---

$$\lambda = \pi_L \pi_Q (C_1 \pi_T \pi_V + C_2 \pi_E)$$

- $\pi_L$ : faktor učenja, zrelost tehnologije
- $\pi_Q$ : kvaliteta proizvodnog procesa, od 0.25 do 20.00
- $\pi_T$ : faktor temperature, a raspon mu je od 0. 1 do 1000  
 $\sim e^{\{-E_a/kT\}}$ ;  $E_a$  - aktivacijska energija ovisna o tehnologiji,  $k$  - Boltzmann-ova konstanta,  $T$  - temperatura
- $\pi_V$ : faktor naponskog stresa za CMOS je u rasponu od 1 do 10, a za sve ostale je 1
- $\pi_E$ : faktor okoline, koji ovisi o operativnoj okolini, a raspon mu je od 0. 38 do 220
- $C_1, C_2$ : faktor složenosti; funkcija broja log. vrata na čipu i broja pinova u pakiranju

# Procjena parametara

---

- procjena točke
- procjena maksimalne sličnosti
- metoda momenata
- analiza linearne regresije
- intervali povjerenja

# Procjena točke

---

- koristi se u analizi rezultata eksperimenata
  - svako ubacivanje kvara i pojava zatajenja neovisan eksperiment
  - $n$  – eksper.  $x_1, x_2, x_3, x_4, x_5, x_6, \dots, x_n$
  - $\{X_1, X_2, X_3, \dots, X_N\}$  slučajan uzorak od  $X$
  - kako procijeniti parametar  $\Theta$  sluč. var.  $X$ ?
  - funkcija procjene  $\Theta$   $\bar{\Theta} = \bar{\Theta}(X_1, X_2, X_3, \dots, X_N)$ 
    - $\bar{\Theta}$  -procjenitelj;  $\bar{\Theta}(X_1, X_2, X_3, \dots, X_N)$  – točka procjene  $\Theta$
  - $\Theta$  je nepristran procjenitelj E akko  $E[\bar{\Theta}] = \Theta$

# Procjena maksimalne sličnosti

---

- pogodna za procjenu parametra kod nepoznate funkcije distribucije vjerojatnosti
- zasniva se na pretpostavci da se uočeni uzorak pojavljuje najčešće od svih mogućih uzoraka
- ako je veličina uzorka premala procjenitelj ne mora biti nepristran

$\bar{x}_n$  – uočeni uzorci  $\bar{\theta}_n$  – nepoznati parametri

$P(\bar{x} | \bar{\theta})$  – vjerojatnost pojave  $\bar{x}_n$  uz param.  $\bar{\theta}$

$\bar{\theta}_{ML}$  –  $\bar{\theta}$  za koje je  $P(\bar{x} | \bar{\theta})$  maksimalna

$$P(\bar{x} | \bar{\theta}_{ML}) \geq P(\bar{x} | \bar{\theta})$$

# primjer ...

---

## ■ pretpostavka eksponencijalne raspodjele

$$\bar{\tau} = (\tau_1, \tau_2, \dots, \tau_N)$$

$$P(\bar{\tau} | \lambda)$$

$$P(\bar{\tau} | \lambda) = (\lambda e^{-\lambda \tau_1})(\lambda e^{-\lambda \tau_2}) \dots (\lambda e^{-\lambda \tau_N})$$

$$P(\bar{\tau} | \lambda) = e^{-\lambda \sum_{i=1}^N \tau_i + N \ln \lambda}$$

$$f(\lambda) = -\lambda \sum_{i=1}^N \tau_i + N \ln \lambda$$

$$\lambda_{ML} = \frac{N}{\sum_{i=1}^N \tau_i}$$

# Procjena ML za Weibullove parametre

---

- 1969 Thoman, Bain, Antle
- Newton-Raphsonova metoda

# Analiza linearne regresije

---

- grafička metoda koja se najčešće upotrebljava za provjeru podataka, tj. za usporedbu eksperimenata i računa
- transformaciji Weibullove funkcije distribucije (CDF) u linearnu funkciju
- Ako podaci odgovaraju Weibullovu razdiobi dobivamo ravnu liniju
- primjenjuje se kao pokazatelj potrebe za dubljom analizom

$$F(t) = 1 - e^{-(\lambda t)^\alpha}$$

$$1 - F(t) = e^{-(\lambda t)^\alpha}$$

$$\frac{1}{1 - F(t)} = e^{(\lambda t)^\alpha} \quad \diagup \ln$$

$$\ln \left[ \ln \frac{1}{1 - F(t)} \right] = \alpha \ln(t) + \alpha \ln(\lambda)$$

# Intervali točnosti

---

- metode daju aproksimativne vrijednosti, stoga je poželjno uvesti intervale točnosti( $0 < p < 1$ )
- p-razina poklapanja u intervalu

# Provjera izračunatih parametara

---

- Goodnes-of-Fit Tests

- Chi-square

- Slučajna varijabla k kategorija  $C_1, C_2, \dots, C_k$

- n – broj parametara slučajne varijable

- Stupanj slobode  $m = k - n - 1$

- $\alpha$  – nivo značaja

- $\chi^2_\alpha$  - tablice

- $\chi^2 \geq \chi^2_\alpha$  - pretpostavka se odbacuje

- Kolmogorov-Smirnov test

- Pogodan za eksponencijalnu razdiobu

$$\chi^2 = \sum_{i=1}^k \frac{(O_i - E_i)^2}{E_i}$$

# Primjer

a. PRIKUPLJENI PODACI

vrijeme	$O_i$	$E_i$	vrijeme	$O_i$	$E_i$
0–5	548	429.20	55–60	2	0.2639
5–10	148	219.15	60–65	1	0.1347
10–15	63	111.89	65–70	1	0.06881
15–20	35	57.13	70–75	1	0.03514
20–25	28	29.17	75–80	1	0.01794
25–30	18	14.89	80–85	1	0.009160
30–35	12	7.60	85–90	1	0.004690
35–40	6	3.88	90–95	1	0.002395
40–45	3	1.98	95–100	1	0.001215
45–50	1	1.01	100–105	1	0.000627
50–55	3	0.5178			

b. KATEGORIJE

vrijeme	$O_i$	$E_i$	$(O_i - E_i)^2/E_i$
0–5	548	429.20	32.88
5–10	148	219.15	23.10
10–15	63	111.89	21.36
15–20	35	57.13	8.57
20–25	28	29.17	0.04
25–30	18	14.89	0.64
30–35	12	7.60	2.53
35–∞	25	7.93	36.74

$$\chi^2 = 125.86$$

- Pretpostavka ekspon.  $\lambda=0.13344$
- $E_i > 5$ ,  $\alpha=0.05$
- $m=8-1-1$
- $\chi^2_{0.05} = 12,592$

# Primjer

	$O_i$	(hrs)	$O_i$	.	$O_i$	$E_i$	$(O_i - E_i)^2/E_i$
0-1	6	11-12	1	.	0-2	9	9.97
1-2	3	12-14	2	.	2-4	7	8.17
2-3	5	14-15	2	.	4-6	12	6.79
3-4	2	15-16	1	.	6-8	2	5.67
4-5	7	16-17	1	.	8-11	9	6.00
5-6	5	17-18	3	.	11-15	5	6.66
6-7	1	18-21	1	.	15-20	5	5.61
7-8	1	21-24	4	.	20-28	6	5.14
8-9	3	24-29	1	.	28-∞	5	5.13
9-10	4	29-38	3	.			0.003
10-11	2	38-75	2	.			
						Total	$\chi^2 = 7.95$

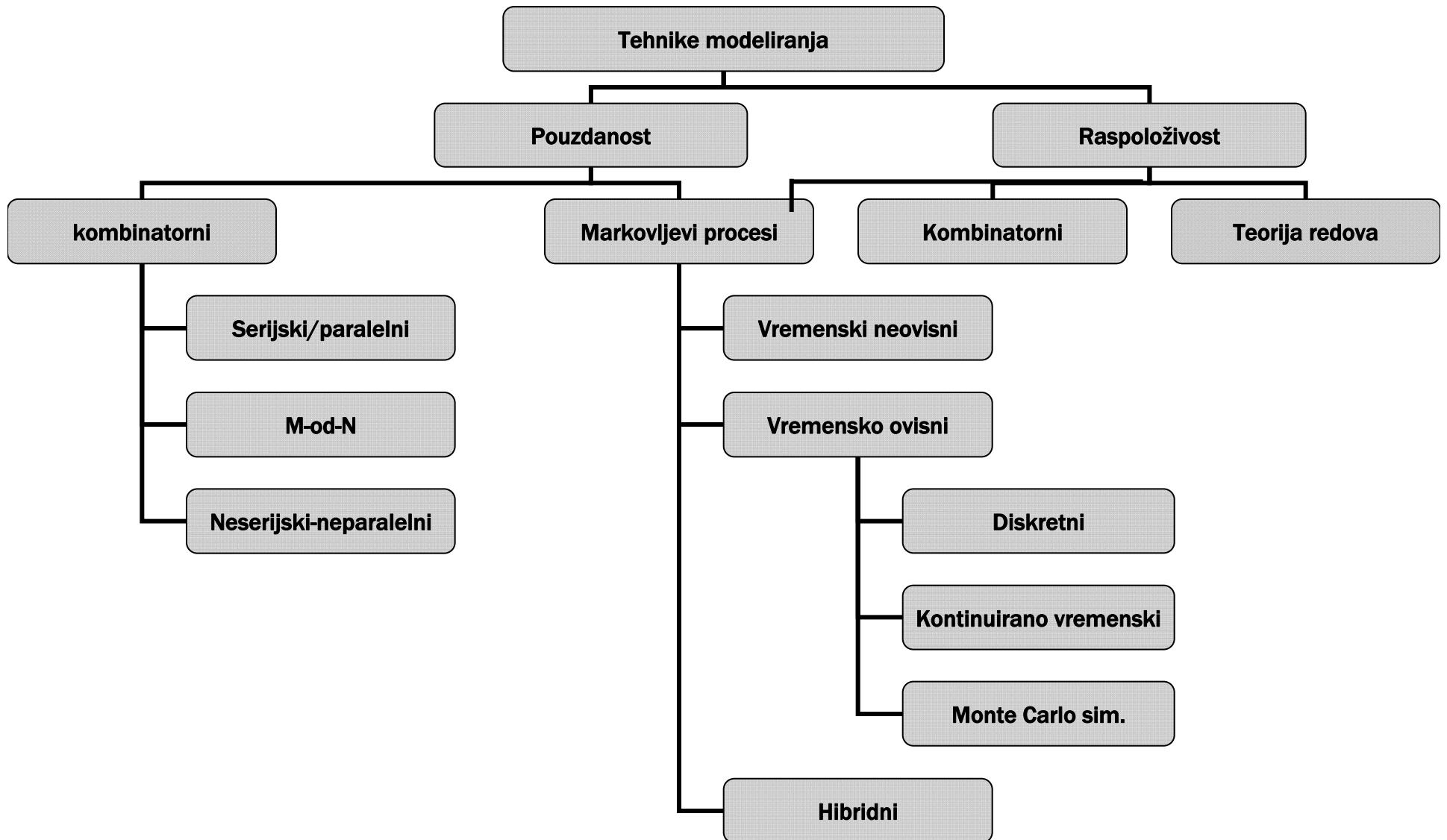
## ■ Pretpostavka Weibullove razdiobe

$$\lambda=0.0888, \alpha=0.98$$

## ■ m=9-2-1

$$■ \chi_{0.05}^2 = 12,592$$

# Tehnike modeliranja



# Kombinatorni model

---

- serijski / paralelni
- M od N
- neserijski/neparalelni

# Modeliranje pouzdanosti složenih sustava

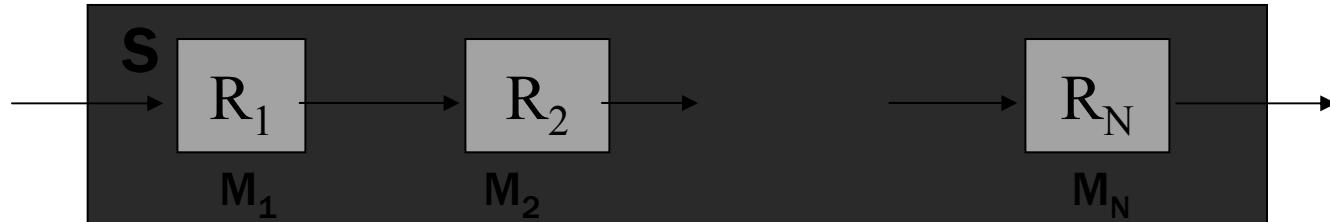
---

## ■ prepostavke

- sustava se sastoji od modula
- svaki modul ima zadanu funkciju pouzdanosti  $R(t)$
- modul koji zataji daje neispravne rezultate
- zatajenja pojedinih modula su neovisna

# Serijski spoj

- najčešća i najlošija kombinacija modula
- kvar bilo kojeg modula izaziva zatajenje sustava
- kvarovi su neovisni
- Blok dijagram pouzdanosti (engl. Reliability Block Diagram)



- pouzdanost sustava

$$R_S(t) = \prod_{i=1}^N R_i(t)$$

- za eksponencijalnu distribuciju pouzdanosti modula:

$$R_S(t) = \prod_{i=1}^N e^{-\lambda_i t} = e^{-\left(\sum_{i=1}^N \lambda_i\right)t}$$

- učestalost kvarova serijskog sustava je suma učestalosti kvarova pojedinih komponenti
  - naziva se princip broja komponenti (“*parts count*” principle)

$$\lambda_S = \sum_{i=1}^N \lambda_i$$

$$MTTF_S = \frac{1}{\lambda_S} = \frac{1}{\sum_{i=1}^N \lambda_i}$$

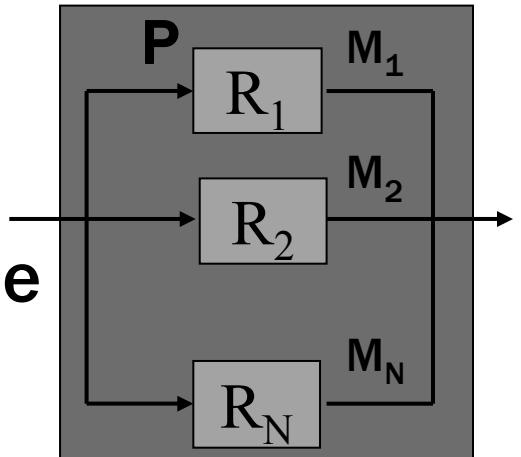
$$0 \leq E(S) \leq \min \{E(M_i)\}$$

- sustav slabiji od najslabijeg modula
- *kod ovakvih sustava nema redundantnosti*

# Paralelni spoj

## ■ pretpostavka

- sustav s rezervnim komponentama
- pokvarena komponenta se zamjenjuje ispravnom pri pojavi kvara
- jedna komponenta dovoljna za ispravan rad sustava
- svi moduli su paralelno povezani i međusobno neovisni



kvar modula  $Q_i(t) = 1 - R_i(t) = P(\bar{M}_i)$

$$\begin{aligned} \text{kvar sustava } P(\bar{M}_P) &= P\left(\bar{M}_1 \cap \bar{M}_2 \cap \dots \cap \bar{M}_N\right) \\ &= P(\bar{M}_1) \bullet P(\bar{M}_2) \bullet \dots \bullet P(\bar{M}_N) = Q_P \end{aligned}$$

$$\text{pouzdanost sustava } R_P(t) = 1 - Q_P$$

---

## ■ za eksponencijalnu distribuciju pouzdanosti modula:

$$R_P(t) = 1 - \prod_{i=1}^N Q_i(t) = 1 - \prod_{i=1}^N (1 - R_i(t)) \geq 1 - ((1 - R_i(t))) e^{-\lambda_i t}$$

$$R_P(t) = 1 - (1 - e^{-\lambda t})^n$$

$$MTTF = E(X) = \int_0^\infty \left[ 1 - (1 - e^{-\lambda t})^n \right] dt = \dots = \frac{1}{\lambda} \sum_{i=1}^N \frac{1}{i} \approx \frac{\ln N}{\lambda}$$

$$u = 1 - e^{-\lambda t} \quad dt = \frac{1}{\lambda(1-u)} du$$

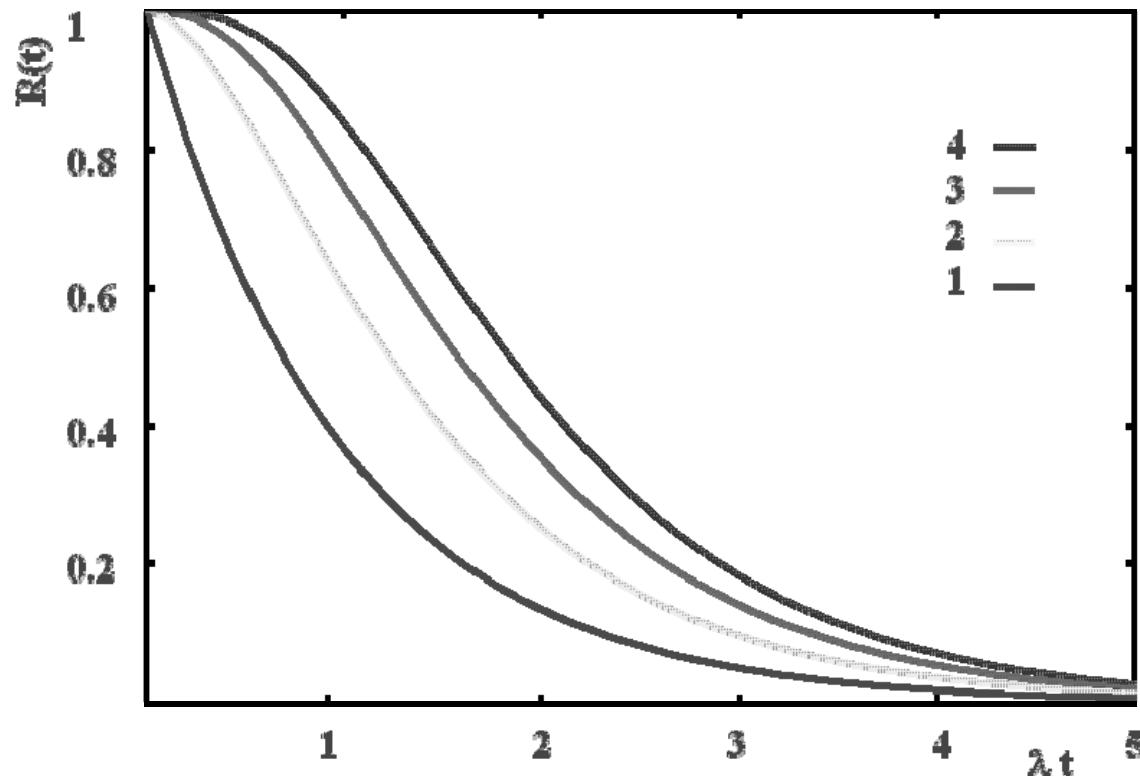
$$\begin{aligned} MTTF &= \frac{1}{\lambda} \int_0^1 \frac{1-u^N}{1-u} du = \frac{1}{\lambda} \int_0^1 \sum_{t=1}^N u^{t-1} du = \frac{1}{\lambda} \sum_{t=1}^N \frac{u^{t-1}}{t} \Big|_0^1 \\ &= \frac{1}{\lambda} \sum_{i=1}^N \frac{1}{i} \approx \frac{1}{\lambda} (\ln(N) + 0.5772) \approx \frac{\ln N}{\lambda} \end{aligned}$$

# primjer:

$$R_{2P}(t) = 1 - (1 - R(t))^2 = 1 - (1 - e^{-\lambda t})^2 = 2e^{-\lambda t} - e^{-2\lambda t}$$

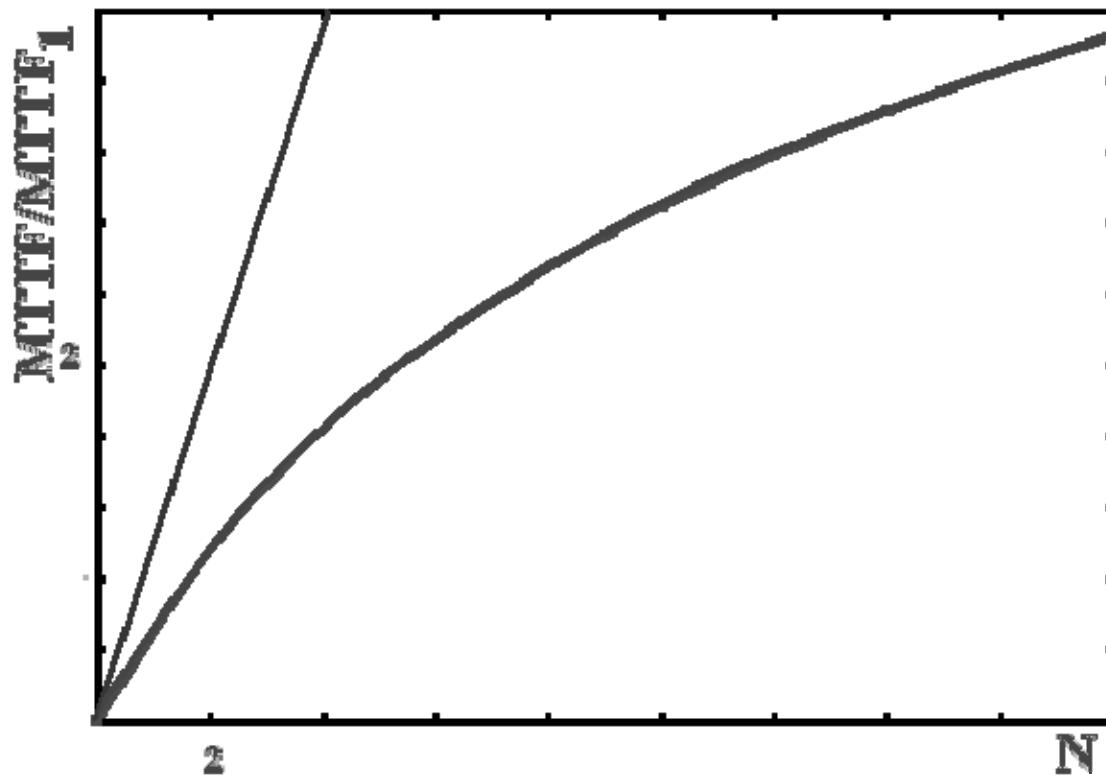
$$R_{3P}(t) = 1 - (1 - R(t))^3 = 1 - (1 - e^{-\lambda t})^3 = 3e^{-\lambda t} - 3e^{-2\lambda t} + e^{-3\lambda t}$$

$$R_{4P}(t) = 1 - (1 - R(t))^4 = 1 - (1 - e^{-\lambda t})^4 = 4e^{-\lambda t} - 6e^{-2\lambda t} + 4e^{-3\lambda t} - e^{-4\lambda t}$$

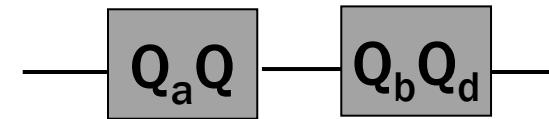
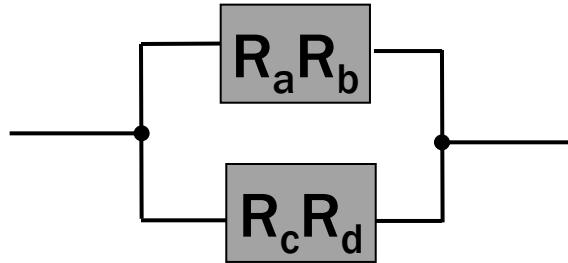
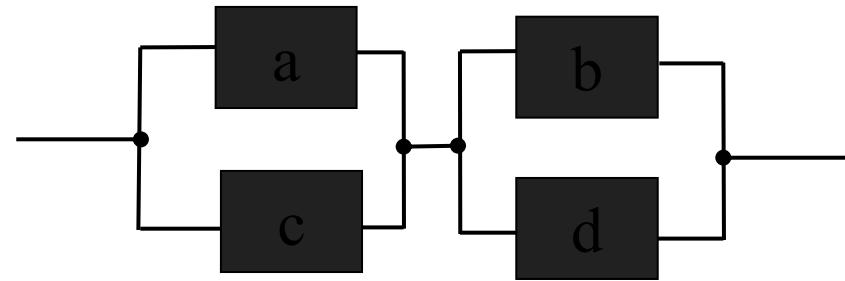
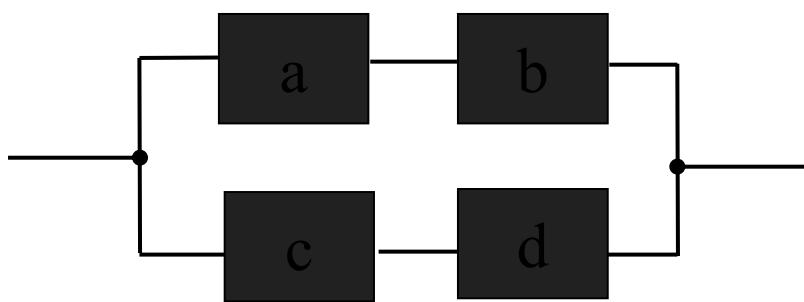


# Utjecaj broja modula na MTTF

---



# Mehanizmi povezivanja modula



kratki spoj: a,b,c,d,ac,ad,bc,bd kratki spoj: a,b,c,d,ac,bd

prekid: a,b,c,d,ab,cd

$$\begin{aligned} R_s &= 1 - (1 - R_a R_b) (1 - R_c R_d) \\ &= 2R^2 - R^4 \end{aligned}$$

prekid: a,b,c,d,ab,ad,bc,cd

$$\begin{aligned} R_0 &= [1 - (1 - R_a)][1 - (1 - R_b)][1 - (1 - R_c)] \\ &\quad [1 - (1 - R_d)] \\ &= 4R^2 - 4R^3 + R^4 \\ R_0 &> R_s \end{aligned}$$

# Pokrivanje (engl. Coverage)

---

- pretpostavka besprijekornog otkrivanja pokvarenog modula i prelaska rada na rezervni dio u složenim sustavima ?
- pokrivanje: uvjetna vjerojatnost da će se sustav oporaviti u slučaju kvara
- faktor pokrivanja: vjerojatnost pokrivanja kvara (ispravno obrađen kvar odgovarajućim mehanizmima neosjetljivosti na pogreške)

$$R_{SYS}(t) = R_1 + cR_2(1 - R_1)$$

$$R_{SYS}(t) = R \sum_{i=0}^{N-1} c^i (1 - R)^i$$

$$R_{SYS}(t) = R \left( \frac{1 - c^N (1 - R)^N}{1 - c (1 - R)} \right) = R \left( \frac{1 - c^N Q^N}{1 - c Q} \right) -$$

---

$$Q_{SYS}(t) = \prod_{i=1}^N Q_i(t) = Q_i^N(t)$$

$$N = \frac{\ln \varepsilon}{\ln Q_m}$$

$$N = \frac{\ln \left[ 1 - \frac{(1-\varepsilon)(1-cQ_m)}{R_m} \right]}{\ln(cQ_m)}$$

$$\begin{aligned}MTTF(N) &= \int_0^\infty R_m \sum_{i=0}^{N-1} c^i (1-R_m)^i dt \\&= MTTF(N-1) + \int_0^\infty R_m c^{N-1} (1-R_m)^{N-1} dt \\&= MTTF(N-1) + \int_0^\infty e^{-\lambda t} c^{N-1} (1-e^{-\lambda t})^{N-1} dt \\&= MTTF(N-1) + \frac{c^{N-1}}{N\lambda} = \frac{1}{\lambda c} \sum_{i=0}^N \frac{c^i}{i} \quad \text{doprinos N je } \frac{c^{N-1}}{N}\end{aligned}$$

---

# M-od-N model

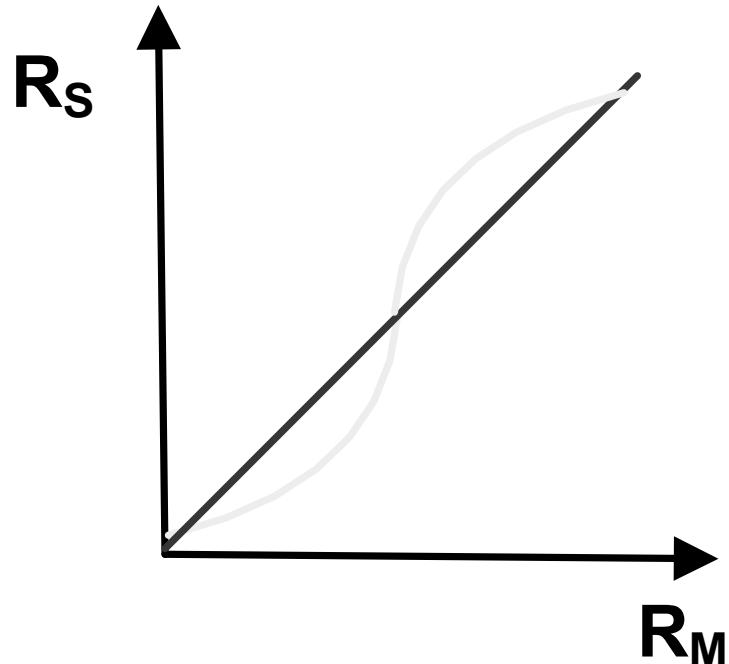
---

- poopćeni paralelni model
- za ispravan rad sustava M modula mora raditi ispravno
- sustav može tolerirati N-M kvarova

$$R_{M\text{-}od\text{-}N} = \sum_{i=0}^{N-M} \binom{N}{i} R^{N-i} (1-R)^i$$

$$R_{2\text{-}od\text{-}3} = 3R^2 - 2R^3$$

$$R_{2\text{-}od\text{-}4} = 6R^2 - 8R^3 + 3R^4$$



# Neserijsko/neparalelni modeli

---

- usložnjenje serijsko-paralelnih modela sustava
- promatraju se mreže modula i osjetljivost sustava na kratki spoj ili prekid na svakom putu, te kombinacije ovih kvarova
- pouzdanost se računa po formulama uvjetne vjerojatnosti

$$A = (A \cap B) \cup (A \cap \bar{B})$$

$$P(A) = P[(A \cap B) \cup (A \cap \bar{B})]$$

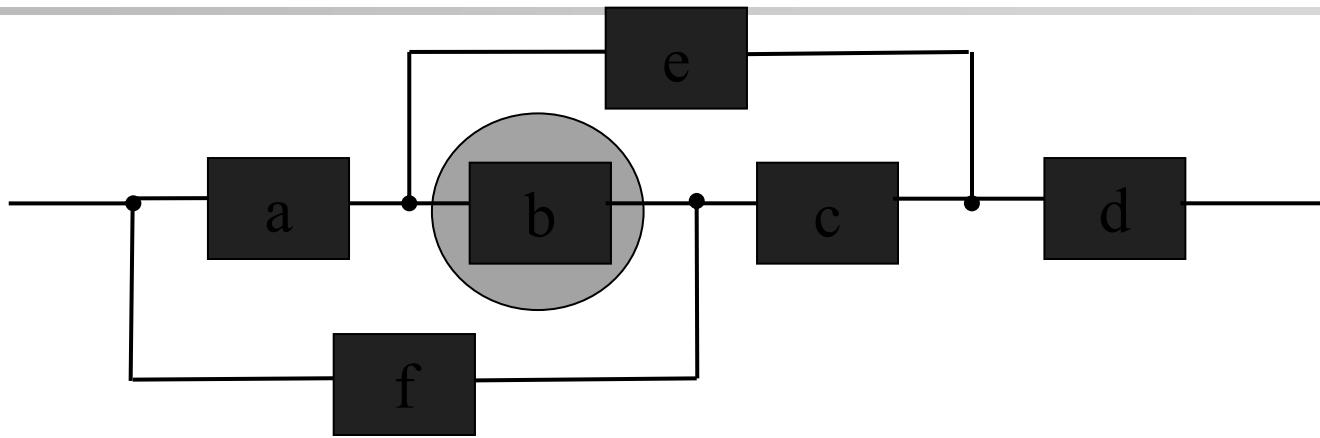
$$= P[(A \cap B)] + P[(A \cap \bar{B})]$$

$$= P(A/B)P(B) + P(A/\bar{B})P(\bar{B})$$

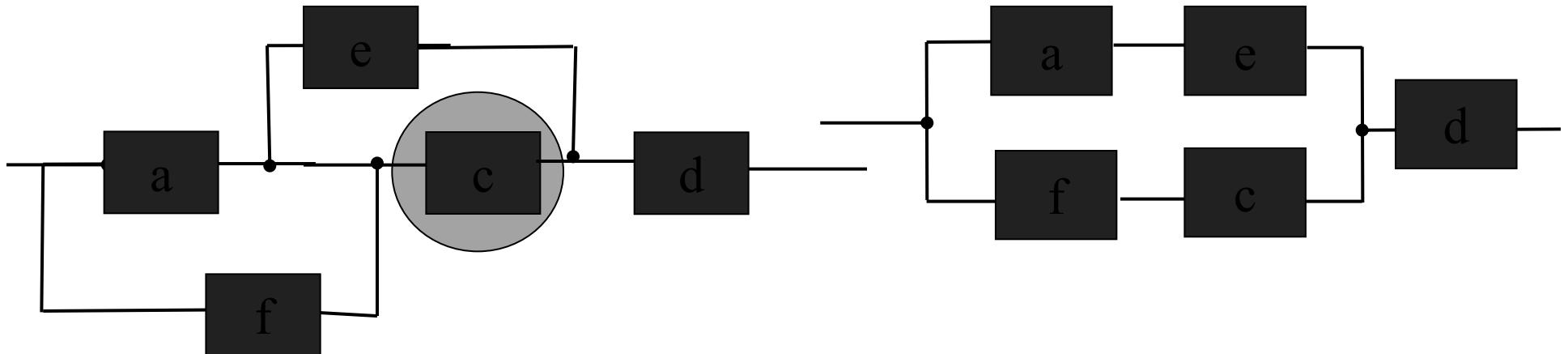
$$R_{SYS} = R_m \bullet P(sustav\_radi | m\_radi) + (1 - R_m) \bullet P(sustav\_radi | m\_u\_kvaru)$$

- pouzdanost se izvodi proširivanjem oko jednog modula

# primjer:

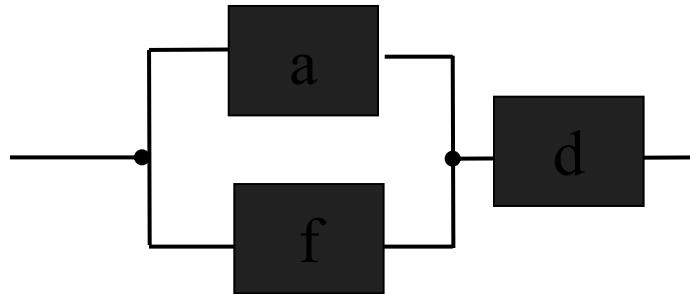


- Dijagram uspjeha



$$R_{SYS} = R_b \bullet P(S | b) + (1 - R_b) \bullet P(S | \bar{b})$$

$$= R_b \bullet P(S | b) + (1 - R_b) \bullet \left\{ R_d \left[ 1 - (1 - R_a R_e)(1 - R_f R_c) \right] \right\}$$



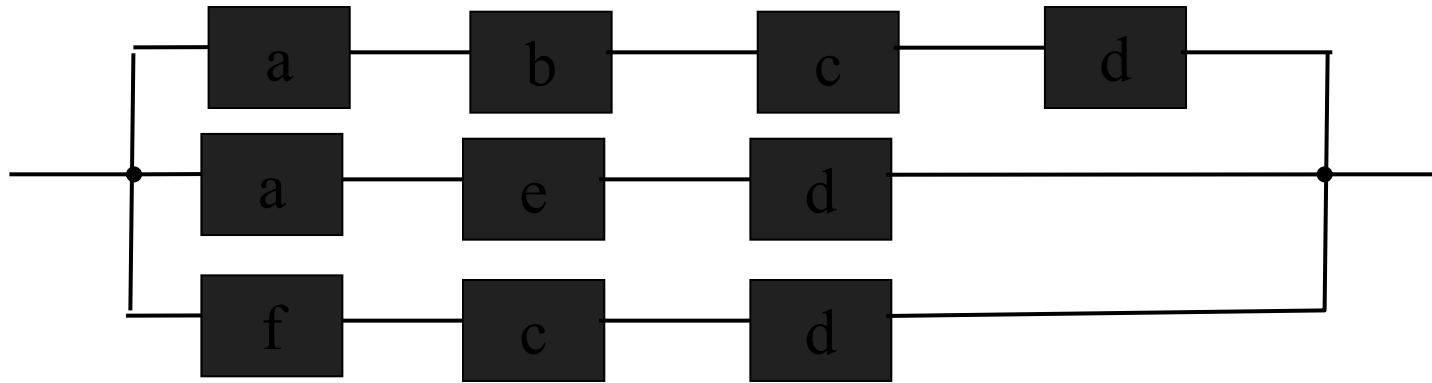
$$\begin{aligned}
 R_{SYS} &= R_b \bullet P(S | b) + (1 - R_b) \bullet P(S | \bar{b}) \\
 &= R_b \bullet P(S | b) + (1 - R_b) \bullet \left\{ R_d \left[ 1 - (1 - R_a R_e)(1 - R_f R_c) \right] \right\}
 \end{aligned}$$

$$P(S | b) = R_c \left\{ R_d \left[ 1 - (1 - R_a)(1 - R_f) \right] \right\} + (1 - R_c) \bullet (R_a R_e R_d)$$

$$R_{SYS} = R^6 - 3R^5 + R^4 + 2R^3$$

# Blok dijagrami pouzdanosti

## ■ engl. Reliability Block Diagram



$$R_{SYS} \leq 1 - \prod (1 - R_i)$$

$$R_{SYS} \leq 1 - (1 - R_a R_b R_c R_d)(1 - R_a R_e R_d)(1 - R_f R_c R_d)$$

$$R_{SYS} \leq 2R^3 + R^4 - R^6 - 2R^7 + R^{10}$$

$$R_{SYS} = R^6 - 3R^5 + R^4 + 2R^3$$

# Pitanja...

# Pouzdanost računalnih sustava

---

Predavanje 5/6: Modeliranje i poboljšanje pouzdanosti

Prof.dr.sc. Vlado Sruk



**Sveučilište u Zagrebu**  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroel., računalne i inteligentne sustave*



# Sadržaj

---

- Markovljevi modeli
- Tehnike poboljšanja pouzdanosti
- Neosjetljivost na pogreške
- Popravljni sustavi.

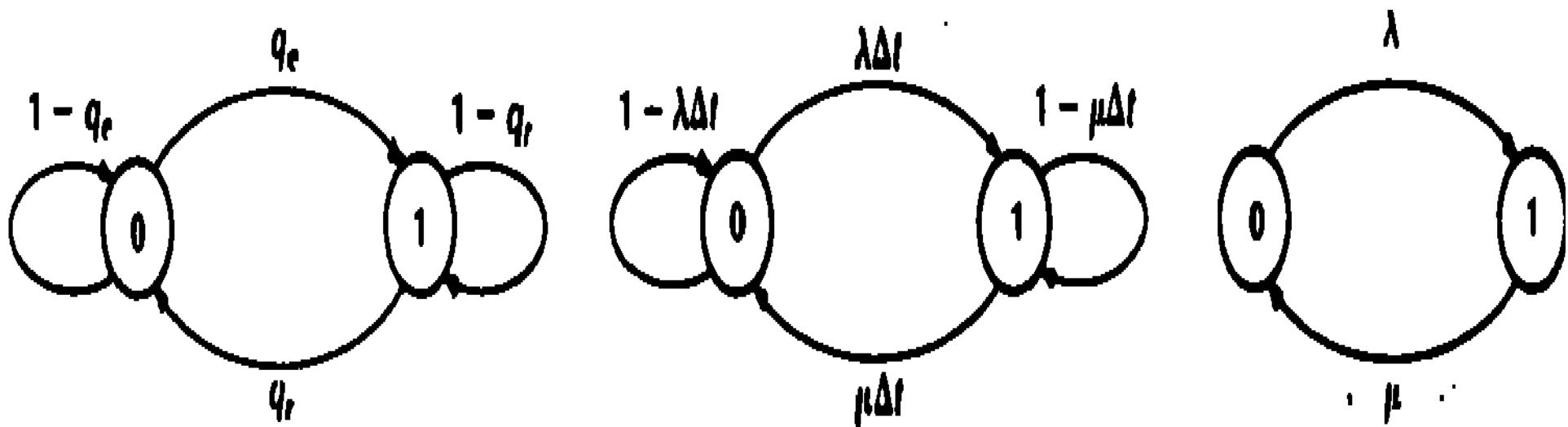
# Markovljev model

---

- posebno pogodan alat za modeliranje sustava u kojima se odvija nekoliko istodobnih procesa, kao što su kvarenje i popravljanje
- koriste se za procjenu pouzdanosti, kao i za procjenu dostupnosti
- Dva najvažnija pojma kod ovakvih modela su:
  - stanje sustava - svako stanje predstavlja različitu kombinaciju ispravnih i neispravnih modula u jednom sustavu - sustav od n komponenti posjeduje  $2^n$  stanja
  - promjene stanja - vjerojatnosti da do promjena dođe
- ovisno o tome da li se promjene stanja odvijaju u diskretnim točkama u vremenu, ili se mogu dogoditi bilo gdje na kontinuiranom vremenskom pravcu dijele se na:
  - diskretno-vremenski model
  - kontinuirano-vremenski model

# Markovljev model s dva stanja

- diskretni model
- diferencijalni model
- kontinuirano-vremenski model
  - vrijeme provedeno u nekom stanju ne utječe na distribuciju u trenutnom ili slijedećem stanju



---

$$\mathbf{P} = \begin{bmatrix} 1-q_e & q_e \\ q_r & 1-q_r \end{bmatrix}$$

$$\vec{P}(k+1) = \vec{P}(k)\mathbf{P}$$

$$[p_0(k+1), p_1(k+1)] = [p_0(k), p_1(k)] \begin{bmatrix} 1-q_e & q_e \\ q_r & 1-q_r \end{bmatrix}$$

$$p_0(k+1) = (1-q_e) p_0(k) + q_r p_1(k)$$

$$p_1(k+1) = q_e p_0(k) + (1-q_r) p_1(k)$$

$$\mathbf{P} = \begin{bmatrix} 1 - \lambda\Delta t & \lambda\Delta t \\ \mu\Delta t & 1 - \mu\Delta t \end{bmatrix}$$


---

$$[p_0(t + \Delta t), p_1(t + \Delta t)] = [p_0(t), p_1(t)] \begin{bmatrix} 1 - \lambda\Delta t & \lambda\Delta t \\ \mu\Delta t & 1 - \mu\Delta t \end{bmatrix}$$

$$p_0(t + \Delta t) = (1 - \lambda\Delta t)p_0(t) + \mu\Delta t p_1(t)$$

$$p_1(t + \Delta t) = \lambda\Delta t p_0(t) + (1 - \mu\Delta t)p_1(t)$$

/  $\Delta t$

**Vjerojatnost  
promjene stanja**

$$\frac{p_0(t + \Delta t) - p_0(t)}{\Delta t} = -\lambda p_0(t) + \mu p_1(t) = \frac{dp_0(t)}{dt} = \dot{p}_0(t)$$

$$\frac{p_1(t + \Delta t) - p_1(t)}{\Delta t} = \lambda p_0(t) - \mu p_1(t) = \frac{dp_1(t)}{dt} = \dot{p}_1(t)$$

$$[\dot{p}_0(t), \dot{p}_1(t)] = [p_0(t), p_1(t)] \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix}$$

$$L\{f(t)\} = f^x(s) = \int_0^\infty f(t)e^{-st} dt$$

$$sp_0^x(s) - p_0(0) = -\lambda p_0^x(s) + \mu p_1^x(s)s$$

---


$$p_1^x(s) - p_1(0) = \lambda p_0^x(s) - \mu p_1^x(s)$$

$$[p_0(0), p_1(0)] = [p_0^x(s), p_1^x(s)] \begin{bmatrix} s+\lambda & -\lambda \\ -\mu & s+\mu \end{bmatrix}$$


---

$$\vec{P}(0) = \vec{P}^x(s)\mathbf{A}$$

$$\vec{P}^x(s) = \vec{P}(0)\mathbf{A}^{-1}$$

$$\mathbf{A}^{-1} = \frac{kofaktor_{ji}(\mathbf{A})}{\det \mathbf{A}} = \frac{\begin{bmatrix} s+\mu & -\lambda \\ -\mu & s+\lambda \end{bmatrix}}{s^2 + \lambda s + \mu s}$$

$$\vec{P}^x(s) = [1, 0] \begin{bmatrix} \frac{s+\mu}{s^2 + \lambda s + \mu s} & \frac{-\lambda}{s^2 + \lambda s + \mu s} \\ \frac{-\mu}{s^2 + \lambda s + \mu s} & \frac{s+\lambda}{s^2 + \lambda s + \mu s} \end{bmatrix}$$

$$p_0^x(s) = \frac{s+\mu}{s^2 + \lambda s + \mu s} = \frac{\frac{\mu}{\lambda+\mu}}{\frac{s}{s+\lambda+\mu}} + \frac{\frac{\lambda}{\lambda+\mu}}{\frac{s}{s+\lambda+\mu}}$$

$$p_1^x(s) = \frac{-\lambda}{s^2 + \lambda s + \mu s} = \frac{\frac{\lambda}{\lambda+\mu}}{\frac{s}{s+\lambda+\mu}} - \frac{\frac{\lambda}{\lambda+\mu}}{\frac{s}{s+\lambda+\mu}}$$


---

$$p_0(t) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{\lambda + \mu} e^{-(\lambda + \mu)t}$$

$$p_1(t) = \frac{\lambda}{\lambda + \mu} - \frac{\lambda}{\lambda + \mu} e^{-(\lambda + \mu)t}$$

$$-\lambda p_0(t) + \mu p_1(t) = 0$$

$$\lambda p_0(t) - \mu p_1(t) = 0$$

$$p_1 = \frac{\lambda}{\mu} p_0$$

$\mathbf{p}_0(t) = \mathbf{A}(t)$

$$p_0 + \frac{\lambda}{\mu} p_0 = 1$$

$$p_0 = \frac{\mu}{\lambda + \mu}$$


---

# Monte Carlo simulacija

---

- standardna metoda složenih sustavi koje je teško analitički modelirati
- $P(n, N_A, N_B, M_A, M_B)$
- globalna inicijalizacija
  - L1:  $i=0; \#stanje$   
 $N_A=N_B=M_A=M_B=0;$   
 $n=-1; \#vrijeme$
  - L2:  $n=n+1$   
 $j=-1; \# slj. stanje$   
 $x=0; \# vjerojatnost$   
 $R=pseudo-sluč broj$
  - L3: ponavljaj  
 $j=j+1;$   
 $x=x+ P(n, NA,NB,MA,MB);$   
dok  $R < x;$
  - L4: ako  $i \neq j$   
postavi jednu od  $\{NA,NB,MA,MB\}$  na  $n+1;$   
 $i=j;$   
ako  $i \neq a$  goto L2

# Pitanja...

# Pouzdanost računalnih sustava

---

**Predavanje 7: Postupci poboljšanja pouzdanosti i neosjetljivosti na pogreške**

**Prof.dr.sc. Vlado Sruk**



**Sveučilište u Zagrebu**  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave*



# Sadržaj

---

- Tehnike oblikovanja sustava neosjetljivih na pogreške
- Popravljni sustavi - postupak oporavka od kvara
- Sklopovske tehnike poboljšanja pouzdanosti
  - pasivna sklopovska redundancija
  - aktivna sklopovska redundancija
  - hibridna sklopovska redundancija
- Vremenska redundancija
- Otkrivanje i tolerancija prijelaznih i trajnih kvarova

# POSTUPAK OPORAVKA OD KVARA

---

- Kod pojave kvara sustav može proći kroz više faza do otklanjanja kvara. Dizajniranje pouzdanog sustava mora nam omogućiti izbor faza za otklanjanje kvara i to kombiniranje nekoliko gore navedenih tehnika.
- FAZE :
  - Ograničavanje kvara
    - ograničiti područje kvara i njegov utjecaj da ne bi došlo do širenja na druga područja. Koristimo razne sklopova za detekciju kvara.
  - Detekcija kvara
    - u ovoj fazi otkriva se da se nešto neočekivano pojavilo u sustavu. Na raspolaganju imamo mnoge tehnike za detekciju kvara, ali jedan period vremena koji se zove "prikrivenost kvara" može proći prije detekcije. Tehnike detekcije kvara dijelimo u dvije grupe : off-line detekcija i on-line detekcija. Kod off-line detekcije, prilikom testiranja, sklop nije u mogućnosti izvoditi posao za koji je namijenjen. On-line detekcija nam omogućava detekciju u stvarnom vremenu, dok sklop obavlja svoj namjenski posao.
  - Dijagnosticiranje
    - ova faza je potrebna kada zakažu tehnike detekcije kvara.
  - Rekonfiguracija
    - faza koja slijedi nakon pronalaska trajnog kvara. Sustav se može rekonfiguirati tako da zamijenimo pokvarene komponente ili da ih izoliramo od ostatka sustava. Komponente mogu biti zamijenjene ili ih možemo iskljuciti i time smanjiti sposobnost sustava.

---

## ■ Oporavak

- Faza u kojoj se koriste tehnike pouzdanosti da se eliminiraju efekti kvara. Dvije osnovne tehnike koje se koriste su maskiranje kvara i ponavljanje. Tehnika maskiranja kvara prikriva efekte kvara tako da omogući redundantnoj informaciji da prevagne netočnu informaciju. U ponavljanju ponovo se izvodi operacija kod koje je nastao kvar i ona se često drugi put izvede uspješno zato jer su mnogi kvarovi koji se javljaju prijelazni i oni ne ostavljaju fizičke poremećaje na sklopovima.

## ■ Ponovno pokretanje

- Nastupa nakon uspješnog oporavka od kvara. Imamo "vruće pokretanje" - ponovni start svih operacija od točke detekcije kvara, moguć je samo ako kvar nije uzrokovao nikakvu trajnu štetu. "Toplo pokretanje" - moguće je pokretanje samo određenih procesa. "Hladno pokretanje" - kada je sustav kompletno zamijenjen jer nakon kvara nije bio u stanju ništa izvoditi.

## ■ Popravak

- Faza u kojoj se komponente koje su dijagnosticirane da su u kvaru, zamjenjuju. Zamjena može biti off-line i on-line. U off-line popravku sustav može nastaviti sa radom ako pokvarena komponenta nije nužna za obavljanje potrebnih operacija ili sustav mora biti ugašen da bi se izvršio popravak. U on-line popravku komponente mogu biti odmah zamijenjene ili sustav može raditi bez pokvarene komponente. U oba slučaja on-line popravka komponente mogu biti fizički zamijenjene, a da se ne prekida rad sustava.

## ■ Reintegracija

- Popravljeni moduli reintegriraju se u sustav.

# Sklopovske tehnike za izbjegavanje kvara

---

- povećanje pouzdanosti sustava smanjenjem vjerojatnosti pojave kvara
- koriste se u kombinaciji sa tehnikama detekcije i tolerancije kvarova
- promatrajući empiričku formulu za pojavu greške možemo izbjegavati kvar manipulirajući određenim faktorima iz formule  $\lambda = \pi_L \pi_Q (C_1 \pi_T \pi_V + C_2 \pi_E)$
- Tehnike :
  - modifikacija radne okoline
  - korištenje komponenti veće kvalitete pouzdanosti
  - stupanj integracije komponenti

# ... izbjegavanje kvara

---

- Modifikacija radne okoline
  - Dva parametra iz formule se odnose na okolinu :
  - Faktor okoline ( e ) i faktor temperature ( t ).
  - U većini slučajeva radna okolina ne ovisi o projektantu sustava i on ne može puno utjecati time na pouzdanost.
    - Npr. Temperatura u sustavu ovisi o nekoliko faktora : temperaturi zraka, prijenosu topline sa čipa na kućište i dalje u okolinu, temperaturi disipacije, i kontroliranjem tih područja možemo povećati pouzdanost.
- Korištenje komponenti veće kvalitete
  - To se odnosi na faktor kvalitete iz formule q.
  - Prema faktoru kvalitete sklopove možemo podijeliti u klase:
    - A) B klasa,  $q = 1$ , vojni standard
    - B) D klasa,  $q = 10$ , komercijalni standard - hermetički pakiran
    - C) D-1 klasa,  $q = 20$ , komercijalni standard - organski materijali

## ■ Faktor kompleksnosti

- Prvi faktor c1 odnosi se na kompleksnost integriranih krugova, a drugi c2 na kompleksnost čipa.
- Veća integracija omogućuje veću pouzdanost.

Broj vrata po čipu	Broj čipova	Vjerovatnost kvara po čipu	Vjerovatnost ukupnog kvara
4	2500	0. 0364	91. 0
100	100	0. 0364	3. 67
10000	1	1. 0112	1. 01

# sklopovske tehnike za detekciju kvara

---

- računaju na neminovnost kvara
- neophodna redundatnost sustava
- izvještavanja:
  - *kvalitativna* - specificira klase kvarova koji se mogu detektirati i može uključivati postotke detekcije za različite klase kvarova.
  - *eksplicitna* - daje vjerojatnost da je neki kvar detektiran
- Tehnike:
  - Udvostručavanje
  - Kodovi za otkrivanje pogreške
  - Samo-provjeravajući sklopovi
  - Sigurna logika (Fail-Safe Logic)
  - Budilica (Watch-dog)
  - Provjera dosljednosti i sposobnosti
  - Nadgledanje procesa

# ... detekcija kvara

---

## ■ Udvostručavanje

- Koncepcijski to je najjednostavnija tehnika detekcije.
- Problem može nastati ako u oba sklopa nastane isti kvar.

## ■ Kodovi za otkrivanje pogreške

- M-od-n kodovi, paritetni kodovi, CRC kodovi, aritmetički i ciklički kodovi.

## ■ Samo-provjeravajuća logika i sigurna logika

- Kod udvostručavanja i kodova može doći do pojave kvara u elementima za komparaciju i do kvarova u dekoderima
- Totalna samo-provjeravajuća logika (TSC)
- Djelomično samo-provjeravajuća logika (PSC)
- Sigurna logika

# ... detekcija kvara

---

- "Watch - dog"
  - Osnova vremenski sklop koji se provjerava
  - Slična je metoda i "*bus-timeouts*" – postavlja maksimalno vrijeme izvođenja operacije
- Provjera dosljednosti
  - Provjerava se smisao međurezultata i rezultata
- Provjera sposobnosti
  - Uglavnom dio operacijskog sustava, ali može biti realizirana i sklopovski
  - Pristup objektu dopušten samo korisniku s odgovarajućom dozvolom
- Nadgledanje procesa
  - Metoda nadgledanja toka
    - Detektira pogreške slijeda programa kao što su skakanje na krivu instrukciju, greške prilikom odluke kod grananja

# skl. tehnike maskirajuće redundancije

---

- tehnike detekcije kvara daju samo upozorenje o pogrešnom rezultatu, ne toleriraju kvar !!!
- maskirajuće tehnike koriste redundanciju da postignu toleranciju pogreške tako da izoliraju ili korigiraju efekte kvara prije nego što izazovu pogrešku i zatajenje
- statički oblik redundancije
- funkcija pouzdanosti postaje važan parametar tehničke uspješnosti maskiranja.
- Tehnike:
  - N-Modularna redundancija sa glasanjem
  - Kodovi za ispravljanje pogreške
  - Hammingov kod
  - Maskirajuća logika
  - Isprepletena logika
  - Kodirani konačni automat

# ... maskirajuća redundancija

---

- *N - modularna redundancija s glasanjem*
  - Udvostručavanje s komparacijom izlaza spomenuto je već prije u tehnikama detekcije kvara.
  - Ako takvom udvostručenom sustavu dodamo i treći jednaki dio imamo dovoljno redundantnih informacija da omogućimo maskiranje kvara u bilo kojem od tri modula. Ovo maskiranje obavlja se većinskim glasanjem ( 2 od 3 ) na izlazu modula.
- Model može biti proširen na n modula i zato se naziva n - modularna redundancija ( NMR ).
  - Uvjet je da n bude neparan broj da bi se izbjegla neodređena stanja izjednačenog glasanja. Troškovi NMR-a su n-puta veći od osnovnog sklopa plus troškovi sklopa za glasanje. Sklop za glasanje uzrokuje kašnjenje signala pa time još umanjuje performanse sustava. Obično omjer performanse / troškovi prevladava potrebu uporabe ove tehnike.

# ... maskirajuća redundancija

---

## ■ *Kodovi za ispravljanje pogreške*

- To je najčešće upotrebljavana tehnika maskirajuće redundancije, a najčešće se koristi Hammingov SEC kod i to u kodiranju memorija
- Niska cijena (redundancija SEC kodova je samo 10-40% ) i minimalna kašnjenja kod kodiranja i dekodiranja
- Memorije predstavljaju veliki dio digitalnog sustava, a u njima se javlja 60-70% svih kvarova takvog sustava
- Kodove koji ispravljaju višestruke greške koriste se samo u specijalnim aplikacijama jer jako podižu troškove
- Neki kodovi rade se specijalno za određene aplikacije u kojima se mogu iskoristiti njihove osobine, a nedostaci ne dolaze do izražaja. Tako je serijsko dekodiranje obično puno jeftinije od paralelnog, pa se koristi kada se podaci prenose serijski ili kad nam performanse nisu važne. Također imamo kodove koji reagiraju samo na određene aritmetičke operacije, pa se koriste za provjeru aritmetičkih procesora.

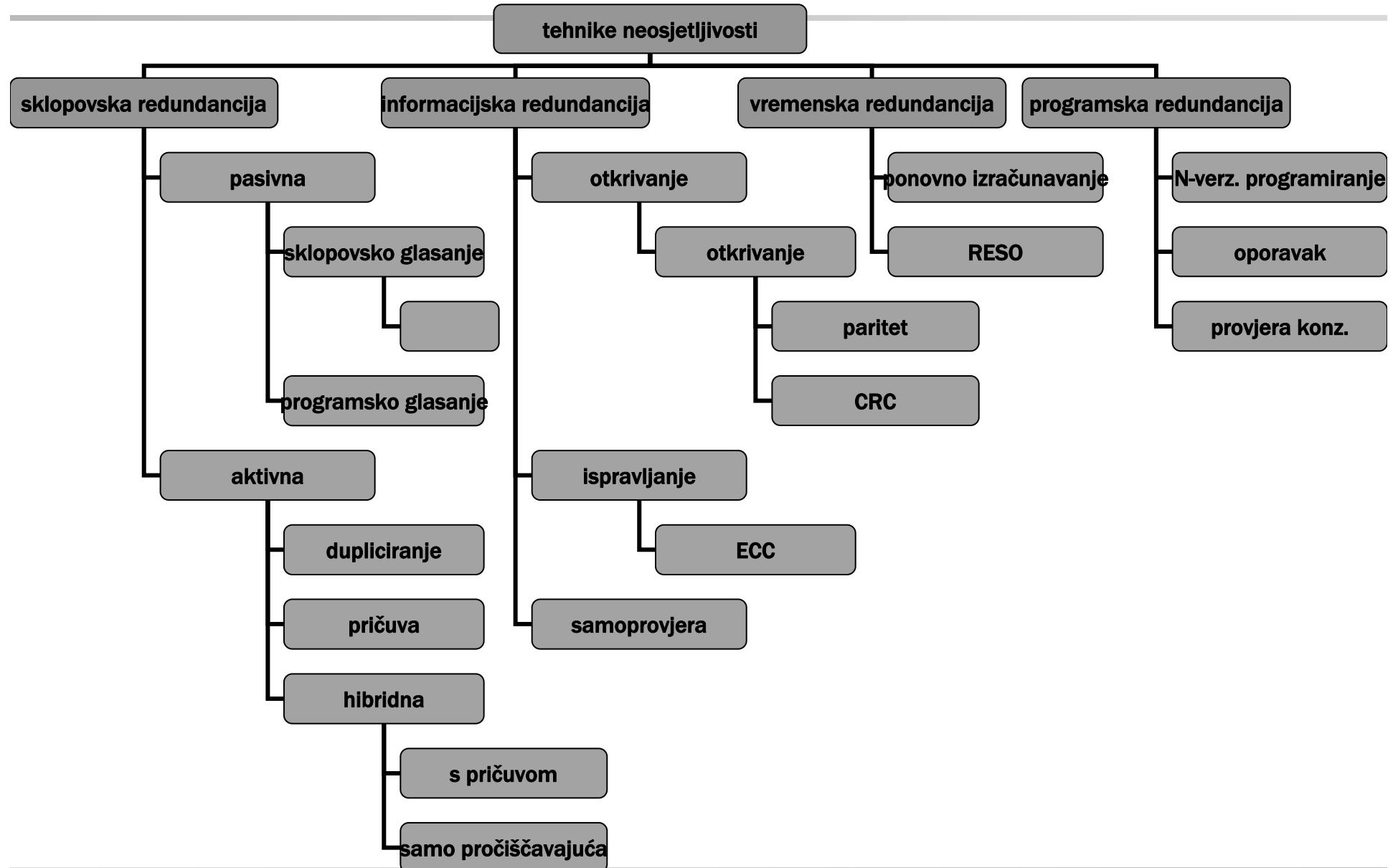
# ... maskirajuća redundancija

---

## ■ Maskirajuća logika

- Prijašnje dvije tehnike maskiranja ne uključuju maskiranje kvara na razini logičkih vrata. One štite neki skup sklopova puno kompleksniji od osnovnih sklopova.
- Tehnike maskiranja na razini logičkih vrata i stanja u konačnom automatu. Zbog svojih visokih troškova samo par tehnika se koristi u stvarnosti :
- **Isprepletena logika** : svodi se na pričvršćen-na-0/1 mjereno na izlazu, ulazu vrata ili izlazu sklopa ( kratko spajanje linije na 0 ili 1 ).
  - Kritični kvar koji uzrokuje grešku na izlazu vrata i nekritični kvar koji ne uzrokuje grešku na izlazu.
- Tolerancija kvara osigurava se korištenjem redundantnih vrata sa redundantnim ulazima.
- Veza između pojedinih slojeva vrata je "isprepletena", tako da kritičan kvar u jednom nivou vrata ostaje maskiran za ostale nivoje miješanjem krivih i dobrih signala, koji ih zamjenjuju.
- **Kodirani konačni automat:** jednostavniji od isprepletene logike jer se koristi manja redundancija i sklopovi su jednostavnije projektirani.
  - Osnovni koncept predložio je Armstrong 1961., A to je da stanja automata, koja predstavljaju varijable stanja, mogu biti kodirana u kodu za ispravljanje pogreške. Tako svaki kvar može biti maskiran ako izaziva grešku koja može biti ispravljena u konačnom automatu.

# Tehnike neosjetljivosti na pogreške



# Zalihost, redundancija

---

- dodavanje informacija, resursa ili vremena izvan potreba za normalan rad sustava
  - sklopovska redundancija
    - dodatno sklopovlje za otkrivanja kvara ili neosjetljivost na greške
  - programska redundancija
    - dodatan kod za otkrivanja kvara ili neosjetljivost na greške
  - informacijska redundancija
    - dodatno informacije npr. kodovi za otkrivanje grešaka
  - vremenska redundancija
    - dodatno vrijeme za provođenje postupaka otkrivanja kvara ili postizanje neosjetljivosti na greške

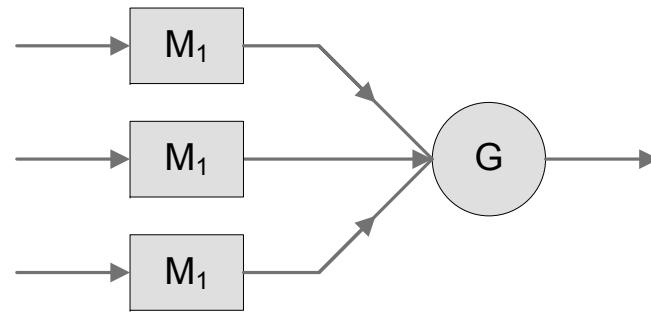
# Sklopovska redundancija

---

- klasa tehnika redundancije koja omogućava toleranciju zatajenja bez potrebe za rekonfiguracijom
- pasivna (engl. passive hardware redundancy)
  - zasniva se na mehanizmima glasanja s ciljem maskiranja pojave kvara
  - dozvoljava pojavu kvara
  - nema mehanizama otkrivanja kvara
  - nema rekonfiguracije sustava
- aktivna (engl. active hardware redundancy)
- mješovita (engl. hybrid hardware redundancy)

# Utrostručeni sustav - TMR

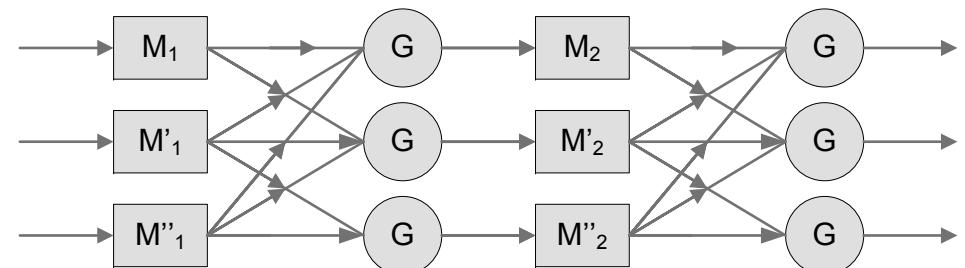
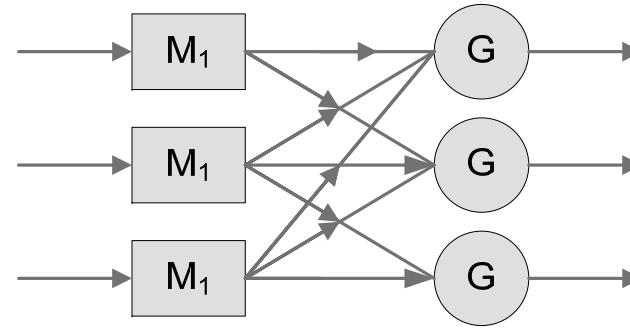
- engl. Triple Modular Redundancy
- najčešći oblik pasivne sklopovske redundancije
- kada je jedan od modula u kvaru druga dva modula prikrivaju kvar većinskim glasanjem
- moguća primjena u programskim sustavima
- problem:
  - kvar sklopa za glasanje → kvar sustava (engl. single-point of failure)



$$\begin{aligned} p_M &= p_{M1} = p_{M2} = p_{M3}, q_M = 1 - p_M \\ p_S &= p_G * (p_M p_M p_M + 3 * p_M p_M q_M) \\ &= p_G * (3p_M^2 - 2 * p_M^3) \end{aligned}$$

# TMR s utrostručenim glasačima

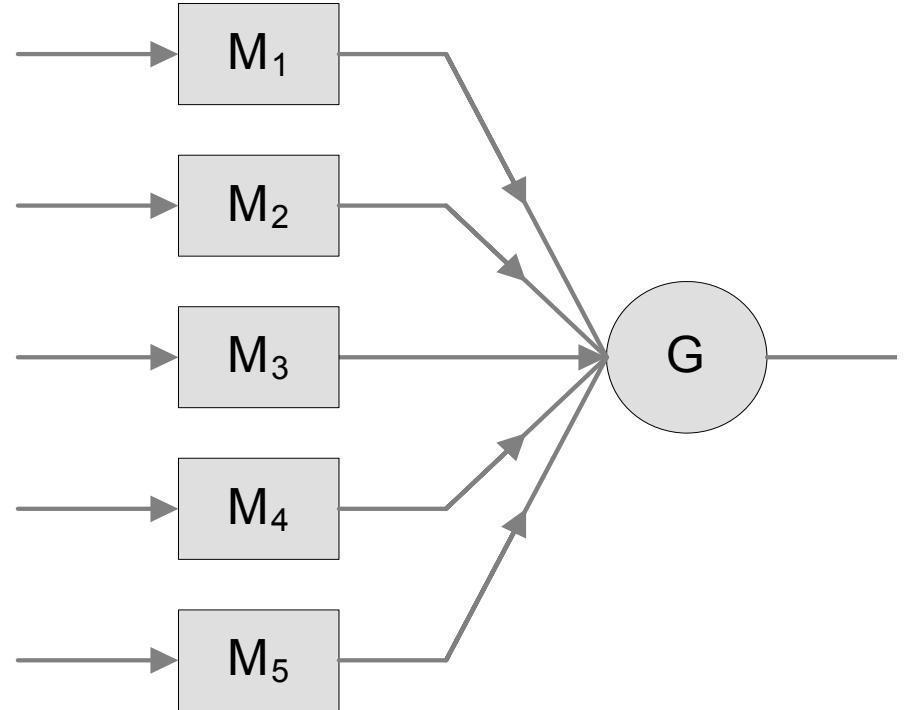
- izbjegavanje utjecaja kvara u glasanju
- kvar u sklopu za glasovanje u jednoj fazi ne utječe na izlaze u sljedećoj – restaurirajući sustav
- kako prijeći na jedan izlaz?



# N-modularna redundancija

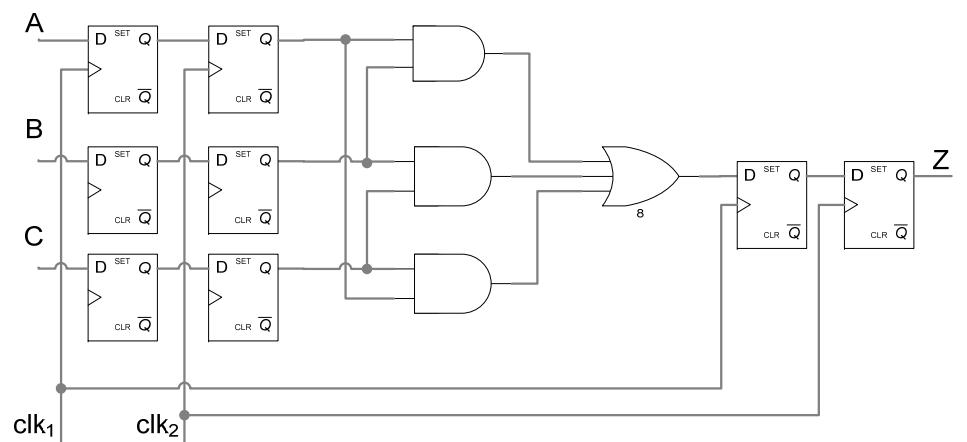
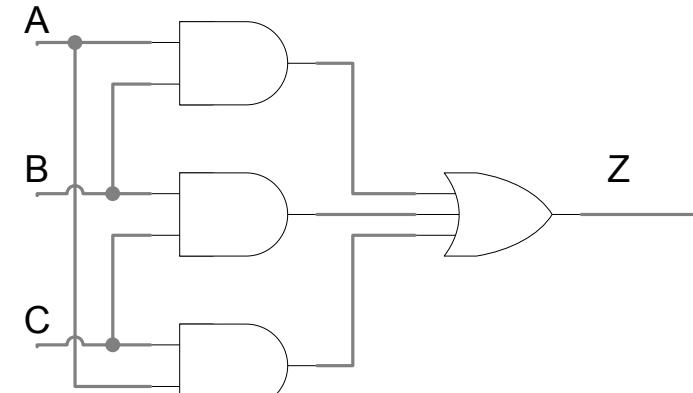
---

- poopćavanje TMR sustava
- dozvoljava veći broj kvarova
- N-neparan broj
- ograničavajući čindbenici
  - potrošnja
  - težina
  - veličina
  - cijena



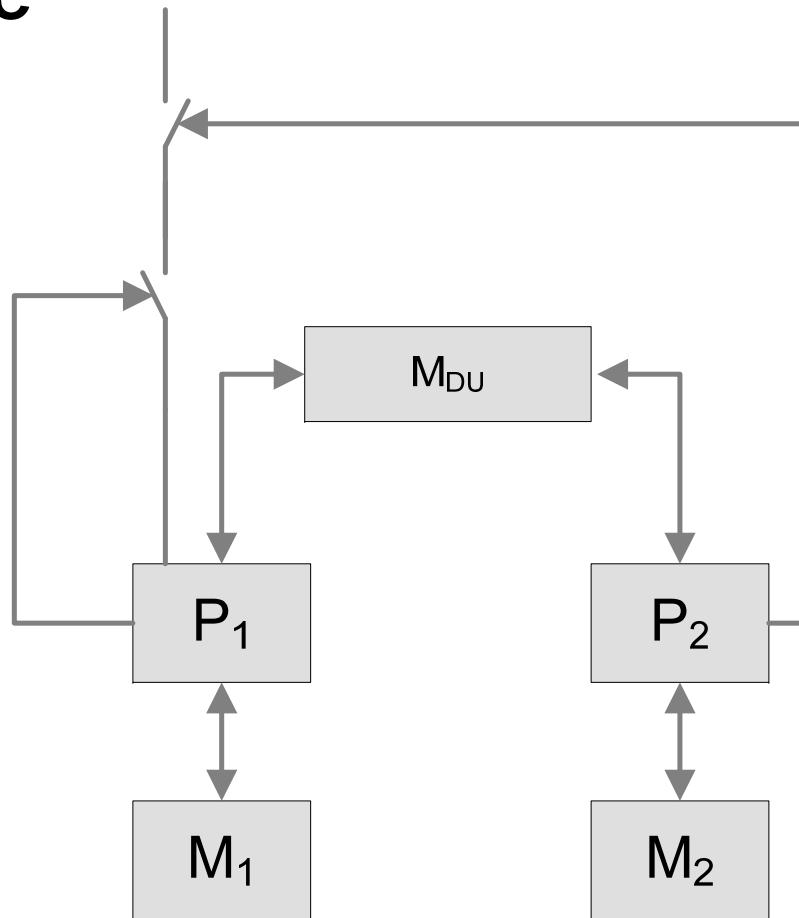
# sinkronizacija glasovanja

- jednobitni sklop za većinsko glasovanje
- vremenski pomaci pojedinih rezultata mogu rezultirati neispravnim rezultatom glasovanja
- neophodna sinkronizacija



# programsko glasanje

## ■ ako nije kritično vrijeme



# aktivna/dinamička sklopovska redundancija

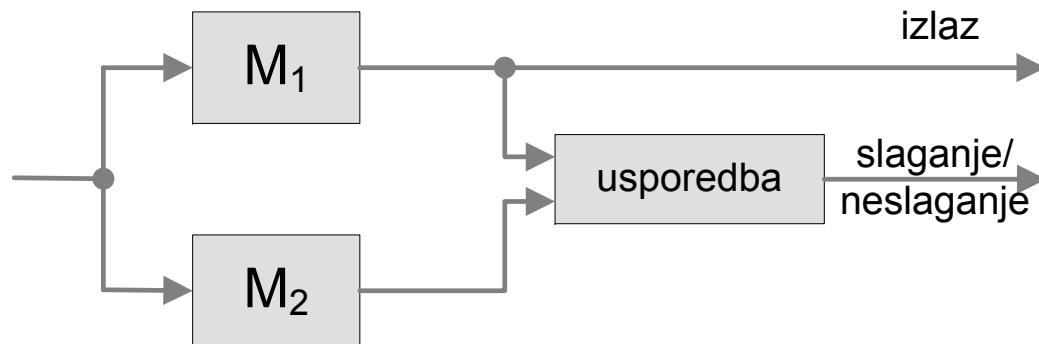
---

- detekcija
- lociranje
- oporavak
- jeftinije i prikladnije kod sustava koji dozvoljavaju kratke prekide rada

# udvostručenje s usporedbom

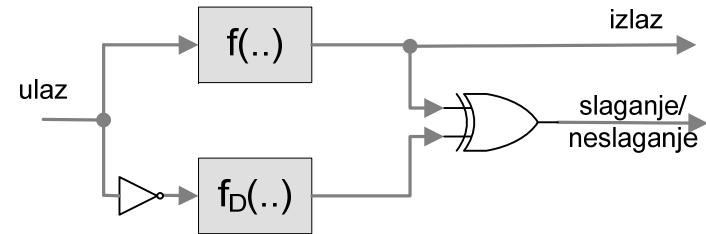
---

- moduli obavljaju istu funkciju
- problemi
  - kvar na ulazu daje uvijek krivi rezultat
  - egzaktna usporedba nije uvijek moguća!!
  - kvar komparatora
  - identični kvarovi



# Usporedba komplementarnom logikom

- moduli ne obavljaju istu funkciju
- rješava problem identičnih kvarova



$$\bar{f} = f_D(\bar{a}, \bar{b}, \bar{c}, \dots)$$

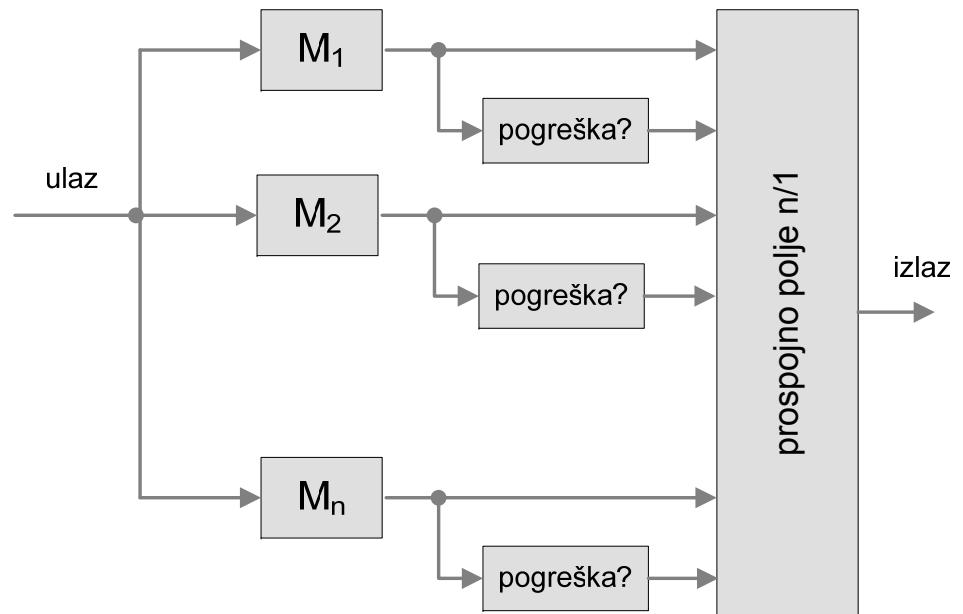
$$f(a, b, c) = ab\bar{c} + \bar{a}\bar{b}c$$

$$f_D(\bar{a}, \bar{b}, \bar{c}) = (a + b + \bar{c})(\bar{a} + \bar{b} + c)$$

$$\bar{f}(a, b, c) = (\bar{a} + \bar{b} + c)(a + b + \bar{c})$$

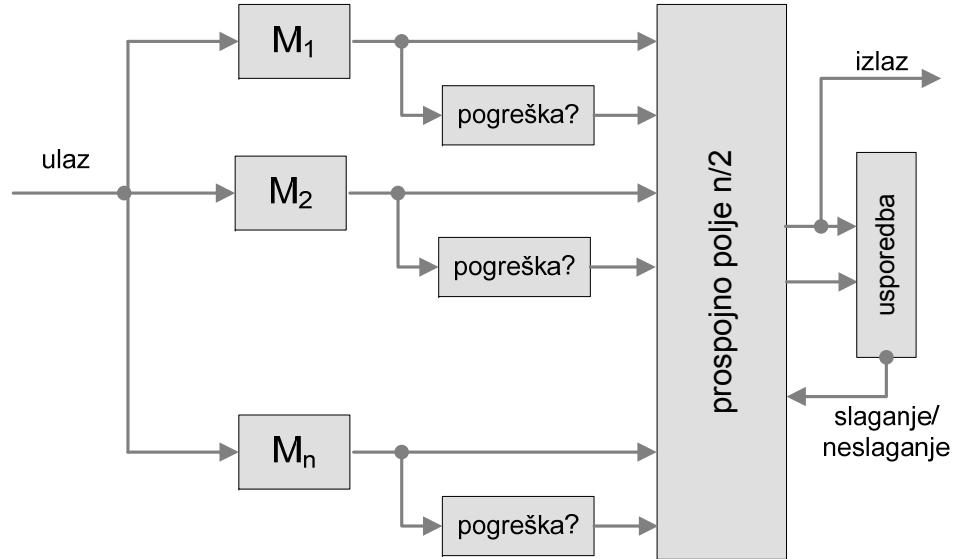
# sklopovi s pričuvom

- jedna komponenta može zamijeniti bilo koju
- jedan modul daje izlaz ostali u pričuvi
- pričuva
  - topla
    - svi moduli rade i vrijeme oporavka je kratko
  - hladna
    - ne troši energiju, ne stare komponente
- osnovna komponenta ovakvih sustava je detekcija pogreške



# udvostručavanje s pričuvom

- engl. Pair and Spare
- dva modula rade u paru cijelo vrijeme
- kod pojave pogreške sustav se rekonfigurira i zamjenjuju modul s pogreškom
- moduli se mogu čvrsto upariti



# Watch dog timer

---

- pogodno za detekciju pogreške
- princip rada: nepostojanje neke aktivnosti → postoji kvar
- vremenski sklop mora biti poništen u nekom vremenskom intervalu
- vremenski interval je funkcija sustava
- pogodno za otkrivanje kvara procesora ili preopterećenja
- moguća uporaba u sklopovskim i programskim sustavima

# dinamička redundancija

---

- omogućava rekonfiguraciju sustava u slučaju kvara, tako da se ne osjeti utjecaj kvara na rad sustava.
- U većini slučajeva rekonfiguracija omogućava odspajanje pokvarenih komponenti od sustava.
- Ako koristimo maskirajuću redundanciju kao dio dinamičke redundancije, odspajanje pokvarenih komponenti može se odgadati sve dok se ne nakupi toliko kvarova da se više ne mogu maskirati, tada započinje odspajanje.
- Rekonfiguraciju uzrokuje ili interna detekcija kvara ili detekcija pogreške na izlazu sustava. ( on-line ili off-line ).
- tehnike detekcije kvara su osnova dinamičke redundancije, jer uspješnost rekonfiguracije sustava u velikoj razini ovisi o sposobnosti detekcije kvara
  - blokiranje efekta kvara prije nego što nastupi trajni kvar
  - detekcija kvara
  - točna dijagnoza lokacije kvara

# **... dinamička redundancija**

---

- Rekonfigurabilno udvostručavanje
- Rekonfigurabilni NMR
- Čuvanje u rezervi
- Skladna degradacija
- Rekonfiguracija
- Oporavak

# ... dinamička redundancija

---

## ■ REKONFIGURABILNO UDVOSTRUČAVANJE

- Udvostručeni sustavi kod statičke redundancije nam ne omogućuju toleranciju kvara, ali uz dva povećanja udvostručenog sustava možemo doći do tolerancije
- potrebno je omogućiti sposobnost određivanja koji od modula je u kvaru ako dođe do njihovog neslaganja o rezultatu.
- sposobnost isključivanja pokvarenog modula i u isto vrijeme izbaciti sada nepotrebne elemente za komparaciju
- nastavak rada
- Osnovna koncepcija : sustav se sastoji od dva modula sa izlazima i sklopa za usporedbu. Samo jedan modul je spojen na izlaz sustava. Drugi modul također radi paralelno sa prvim, ali on nije priključen na izlaz sustava. U sklopu za usporedbu stalno se vrši provjera rezultata i kad se rezultati ne poklapaju znači da je u jednom od modula nastao kvar. Sada treba odrediti koji je to modul i isključiti ga iz rada. Postoje četiri metode sa kojima se može odrediti modul u kvaru. Prva je da pustimo program za dijagnosticiranje kvara. Druga mogućnost je ugradnja samo-provjeravajućih krugova u svaki modul. Treća mogućnost je korištenje "watch-dog" vremenskih sklopova, a zadnja metoda je korištenje vanjskog arbitra za kontrolu konfiguracije sustava. Dva procesora izvode identične test programe. Ako postoji kvar u jednom, velika je vjerojatnost da će on davati pogrešne odgovore. Specijalni upravljački sklop provjerava odgovore i odlučuje o isključivanju. Kod tih provjera može doći do problema sinkronizacije procesa, što se otklanja odgovarajućim metodama.

# ... dinamička redundancija

---

## ■ REKONFIGURABILNA N-MODULARNA REDUNDANCIJA

- Jedan od nedostataka NMR-a je da mogućnost maskiranja kvarova opada sa brojem pokvarenih modula, koji čak mogu nadglasati ispravne. Ako omogućimo da neispravni moduli budu isključeni iz sustava i ne sudjeluju više u glasanju tada se neće pojavljivati mogući problemi u radu statičke NMR-e.
- Postoje dvije metode rekonfiguracije NMR-a :
- *Hibridna redundancija* : zamjenjuje pokvarene module sa rezervnim nekorištenim modulima. To je zapravo mješavina dviju tehnika redundancije : N-modularne sa glasanjem i čuvanja u rezervi.
- *Adaptivno glasanje* : dinamički se modificira sustav glasanja kako se sustav smanjuje. Imamo i modula koji ulaze u sklop za glasanje koji ima  $n_i$  ulaza kojima se pridjeljuje faktor  $a_i$ . Odluka sklopa za glasanje zasniva se na  $\sum a_i n_i$

# ... dinamička redundancija

---

## ■ ČUVANJE U REZERVI

- U osnovi, detekcijom kvara zamjenjuju se pokvareni moduli sa rezervnim. Detekcija kvara može biti interna ili eksterna ili kombinacija oba slučaja.
- Pažnja se mora обратити на kompleksnost kod zamjene modula i na efikasnost korištenih tehnika detekcije.

## ■ SKLADNA DEGRADACIJA

- Ova tehnika koristi redundantno sklopolje kao dio normalnog rada. Postoje dvije metode :
- *Dizajnirano-u sustavu* : sklopoli potrebni za obavljanje specificirane funkcije tako projektirani da mogu nastaviti rad i u slučaju kvara uz smanjenje performansi i mogućnosti.
- *Dodano -na sustav* : sustavi sa samo jednim dodatnim procesorom. Ako neki aktivni procesor padne, samo dio kapaciteta se gubi, sustav i dalje pruža željene performanse.

# ... dinamička redundancija

---

## ■ Oporavak

- Osigurava sustav koji nam omogućuje izvršavanje procesa, bez potrebe ponovnog podizanja sustava ako dođe do kvara, uz nikakvo ili minimalno gubljenje informacija o procesu. To su uglavnom programske tehnike, ali ima nekih sklopoških elemenata. Sve tehnike su takozvane :
- "oporavak od pogreške prema natrag ",
  - Izvršenje procesa vraća na neku točku prije pojave greške. Jedna od metoda za tu tehniku je "ponovno startanje". To je najbrža i najjednostavnija metoda oporavka od pogreške. Kvar se pokušava detektirati što prije, kada se pojavi. Nakon detekcije, popravlja se sve što je potrebno.
  - Ako je kvar tranzijentan čeka se da sam nestane. U slučaju tvrdog kvara sustav se rekonfigurira. Operacije pogodjene kvarom ponovo se ispituju, s tim da se zna u kojem stanju je sustav bio neposredno prije kvara. Ako je operacija pogodjena kvarom već izazvala promjenu nekih podataka sa nemogućnošću popravka, ponovno startanje će biti neuspješno. Ova tehnika se najviše primjenjuje kod toleriranja tranzijentnih kvarova.

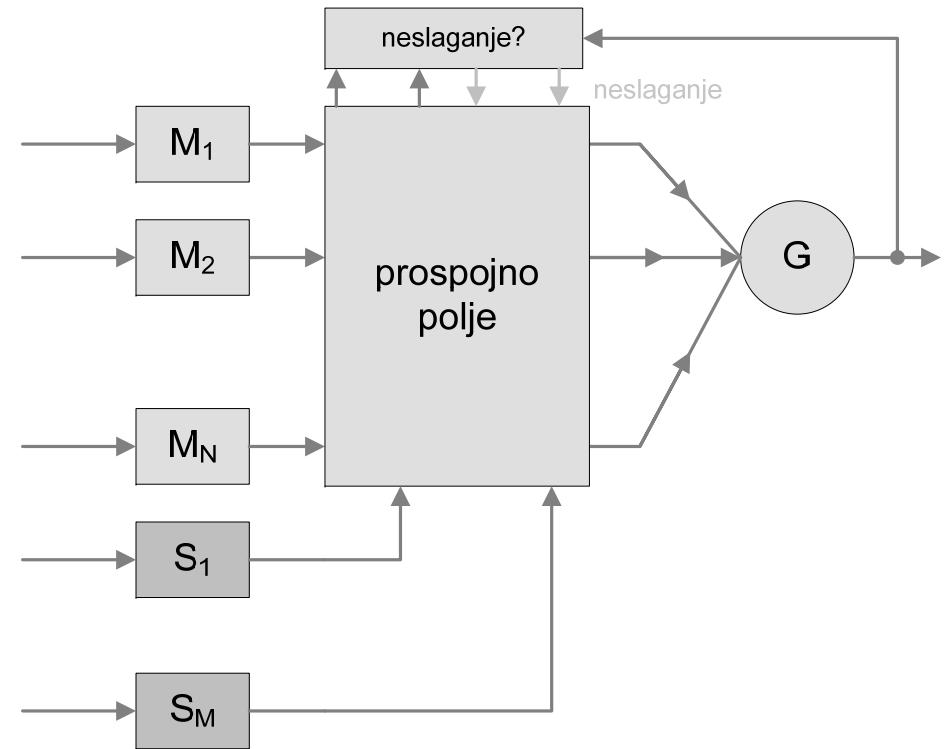
# Hibridna redundancija

---

- kombinira aktivnu i pasivnu redundanciju
  - pasivna
    - maskira kvar
  - aktivna
    - detekcija, oporavak
- uporaba u sustavima koji zahtijevaju iznimno visok stupanj pouzdanosti

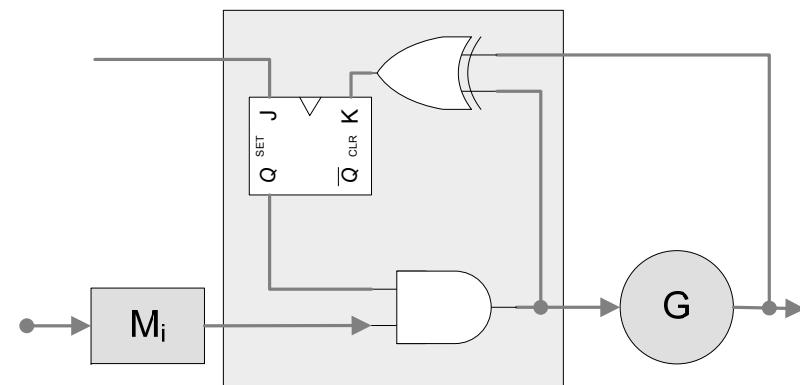
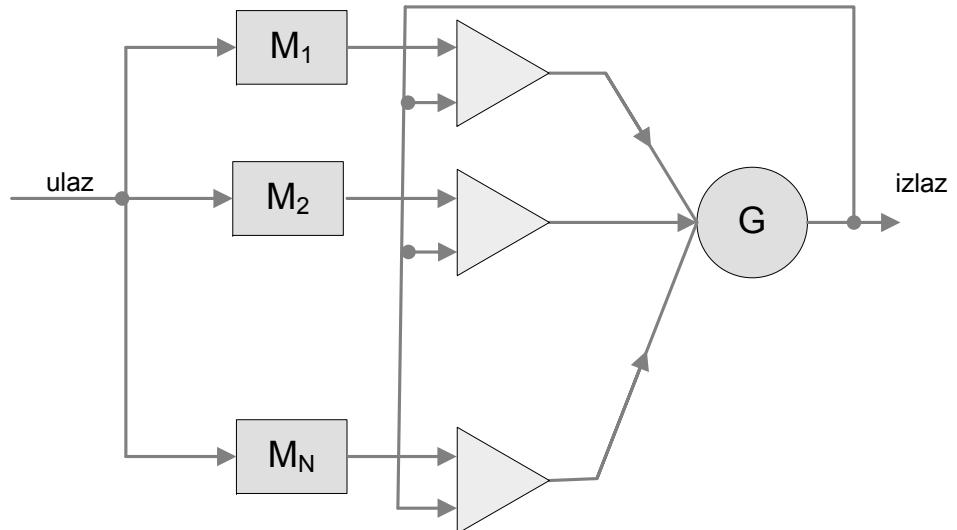
# N-modularna s pričuvom

- najčešće ostvarenje
- modul čiji izlaz se ne slaže s većinom izbacuje se iz glasanja, zamjenjuje ga pričuvni modul
- glasovanje se provodi isključivo nad aktivnim modulima
- npr. TMR sustav s jednom pričuvom (ukupno 4 modula) može tolerirati dva kvara



# Samopročišćavajuća redundancija

- engl. self-purging
- odmah isključuje modul koji je u kvaru
- svi moduli sudjeluju u glasovanju
- sklopka za isključivanje je jednostavne izvedbe
- sklop za glasovanje ?
  - threshold-gate
    - binarni ulazi
    - 1 binaran izlaz
    - uspoređuju se težinske sume ulaza s definiranim pragom glasovanja
    - težina modula u kvaru 0



# Funkcija praga

---

- engl. threshold -gate
- $x_i$  ulazi
- $z$  izlaz
- $\sum_{i=1}^n w_i x_i \geq T$  težinska suma

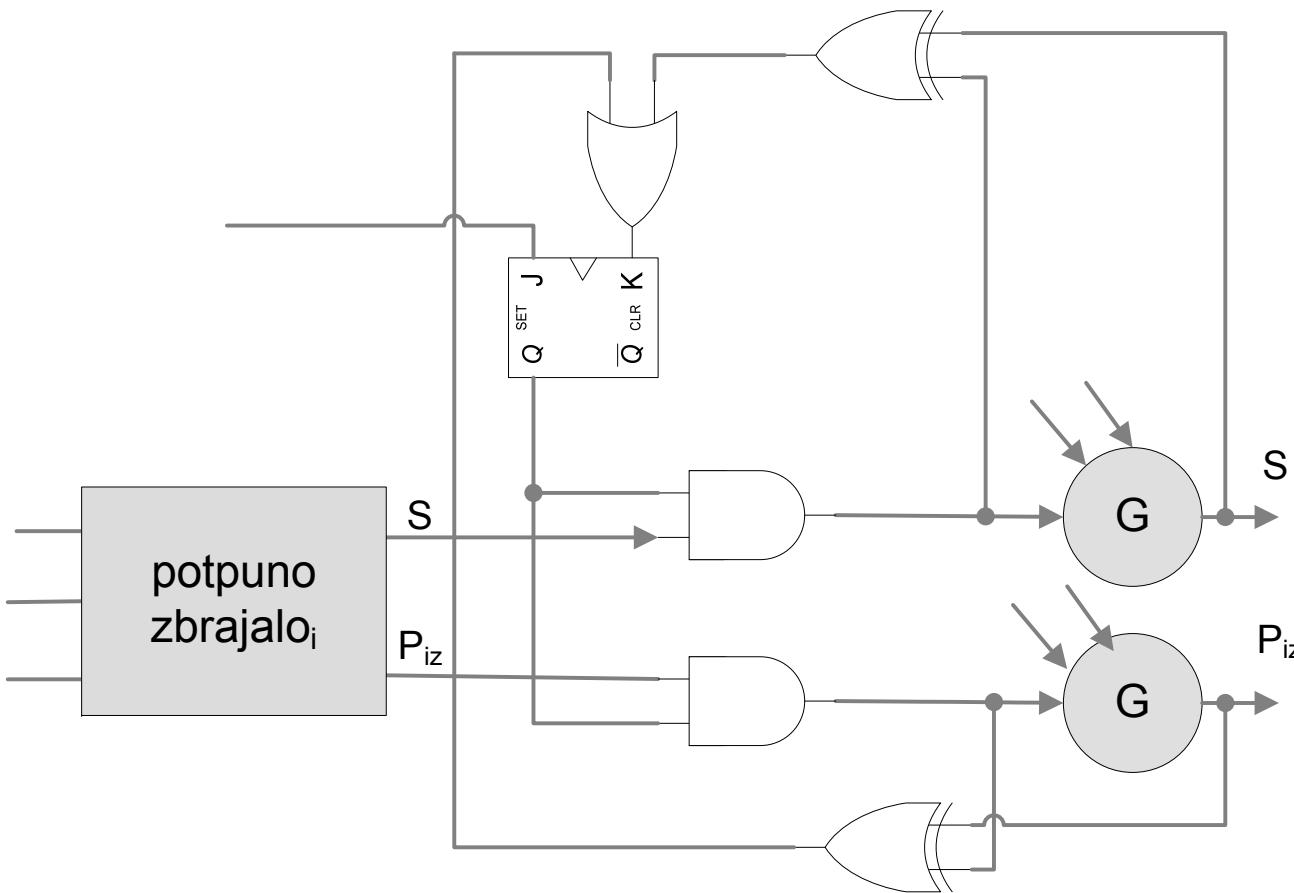
$$z = \begin{cases} 1, & \text{akko } \sum_{i=1}^n w_i x_i \geq T \\ 0, & \text{akko } \sum_{i=1}^n w_i x_i < T \end{cases}$$

- T prag
- ostvarenje sadrži  
analogne elemente
- nepraktično

$x_1$	$x_2$	$x_3$	$\sum_{i=1}^n w_i x_i$	$z$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	2	1
1	0	0	1	0
1	0	1	2	1
1	1	0	2	1
1	1	1	3	1

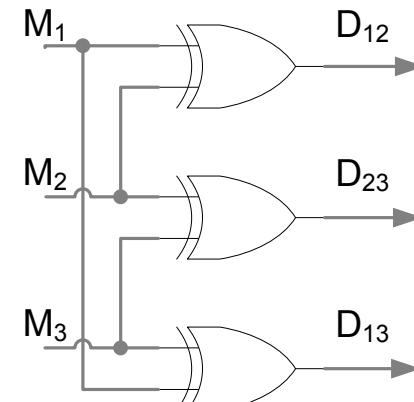
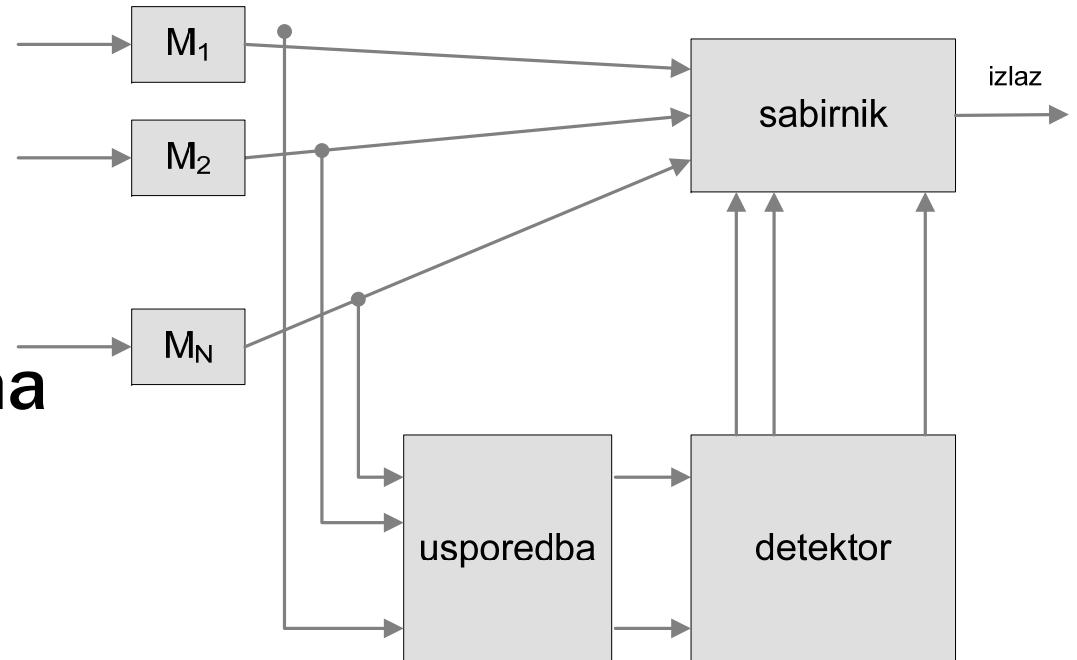
# primjer potpunog zbrajala

---



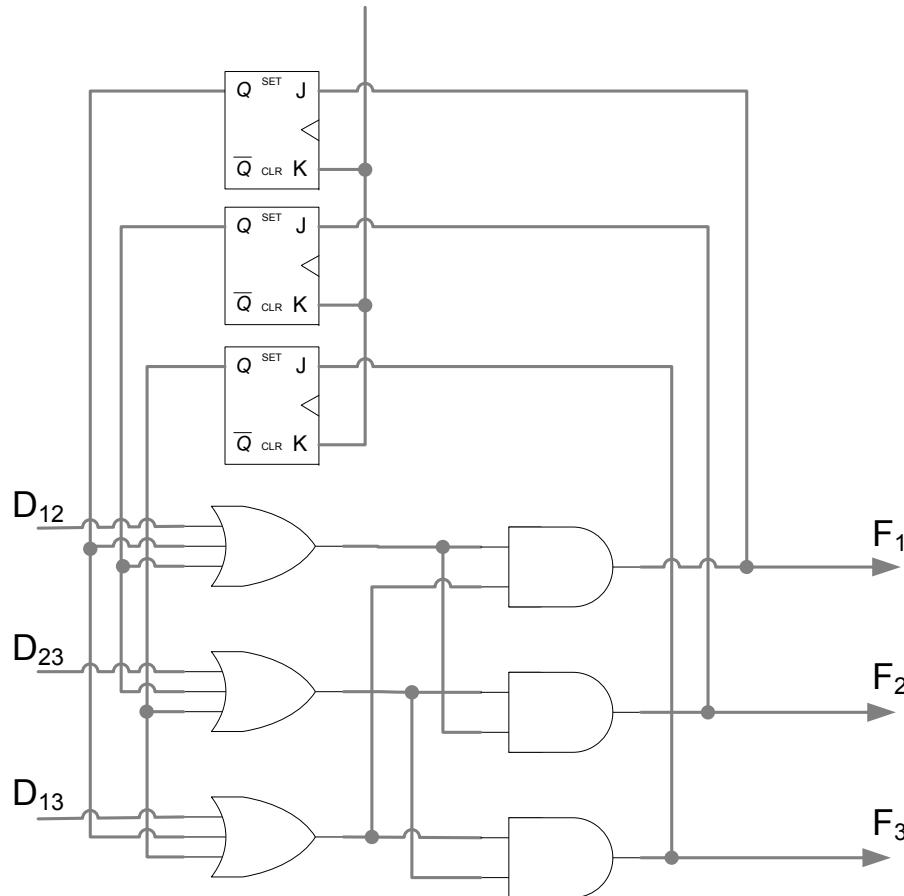
# Prosijavajuća redundancija

- engl. Shift-out
- Distribuirano glasanje
- svi moduli u funkciji
- usporedba svakog modula sa svim ostalima
- detektor određuje koji modul se ne slaže s ostalima
- sabirnik ne uzima u razmatranje rezultate modula koji se ne slažu s ostalim

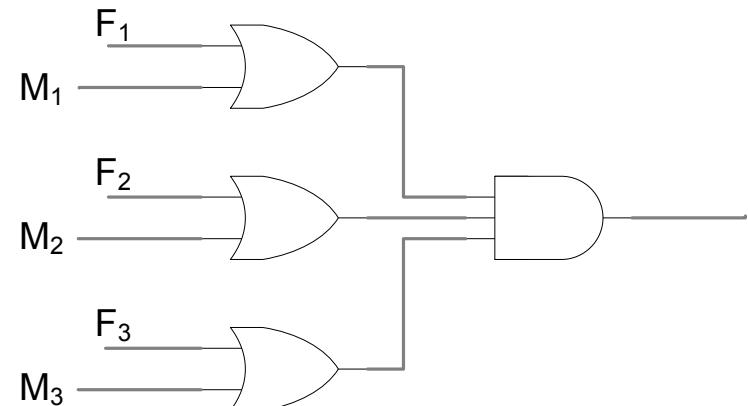


# Princip ostvarenja

## ■ detektor



## sabirnik



# Samoispitna logika

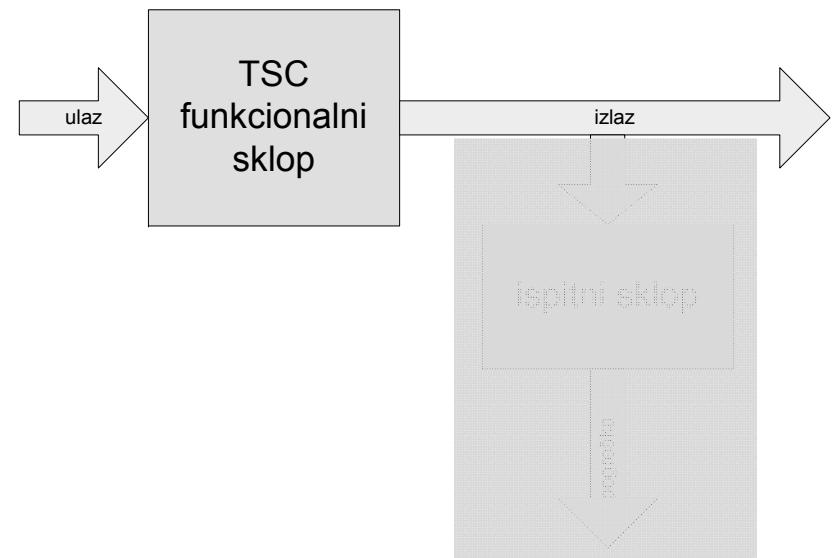
---

- može otkriti pogrešku na ulazu i unutar sklopa bez vanjskog poticaja
- ulaz i izlazi su kodne riječi
- sklop je siguran od kvarova (engl. fail-safe) ako bilo koji jednostruki kvar za *ispravnu kodnu riječ* na ulazu daje "sigurnu" kodnu riječ na izlazu
- sklop je samo-ispitni (engl. self-testing) ako postoji barem jedna ispravna kodna riječ na ulazu za koju će uz prisustvo kvara u sklopu na izlazu sklopa biti nekodnu riječ
- sklop je totalno samoispitni ako je siguran od kvarova i samo-ispitni

# arhitektura totalno-samoispitnih sklopova

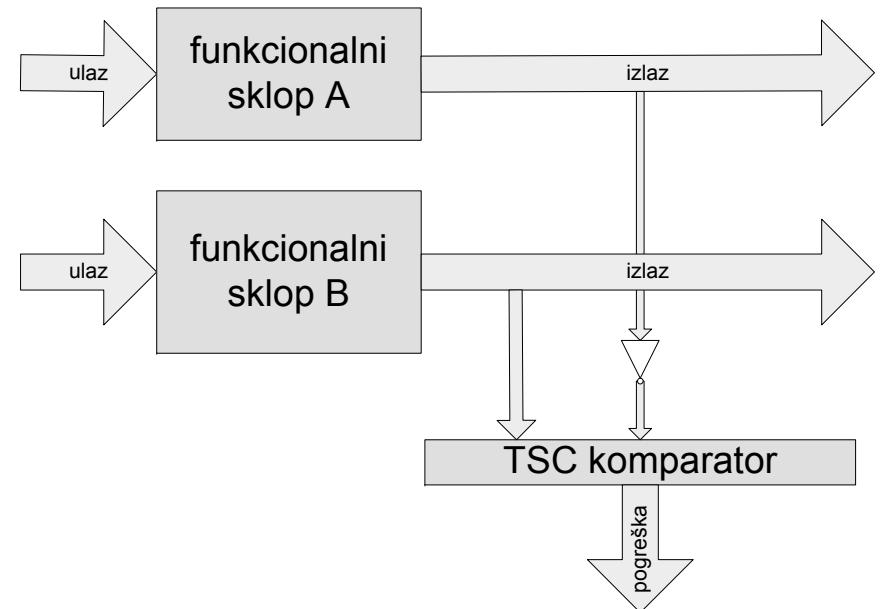
---

- oba sklopa su totalno samoispitni
- prednost pred totalno samoispitnim sklopom je garancija ispitnog sklopa da je izlaz ispravan



# TSC s udvostručavanjem

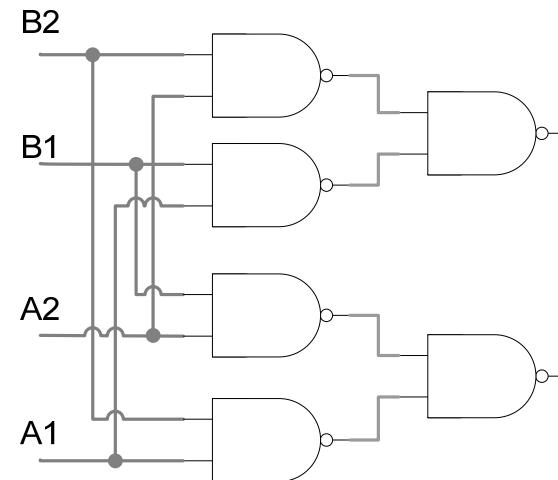
- dvije kopije iste funkcije



# dvotračni ispitni sklop

---

- engl. two-rail checker
- ulazne kodne riječi su međusobno komplementarne



**TABLE 3–15 TSC dual-rail comparator responses to stuck-at-faults**

Inputs			Normal Output	Outputs C2C1 Resulting from Single Stuck-at-1 Faults															
B2B1	A2A1	a		b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
01	01	10	11	10	11	10	10	10	10	10	11	11	10	10	00	10	10	10	11
01	10	01	11	01	01	11	11	01	01	11	01	01	01	01	01	00	01	11	01
10	01	01	01	11	11	01	01	11	11	01	01	01	01	01	01	01	00	11	01
10	10	10	10	11	10	11	10	10	10	10	11	10	10	11	00	10	10	10	11

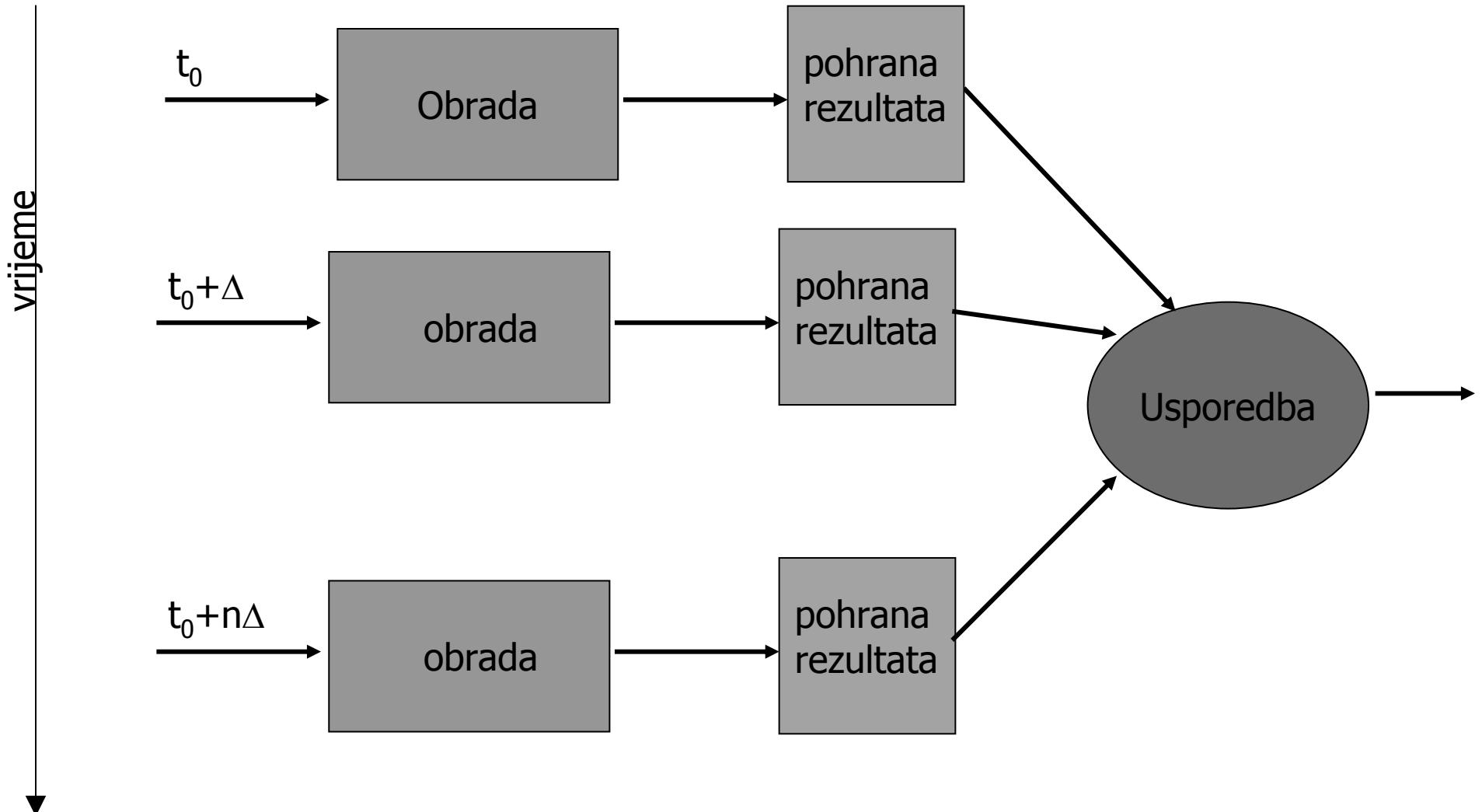
Inputs			Normal Output	Outputs C2C1 Resulting from Single Stuck-at-0 Faults															
B2B1	A2A1	a		b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
01	01	10	10	00	10	00	10	10	00	00	10	10	10	10	10	11	11	00	10
01	10	01	01	00	00	01	01	01	01	00	00	01	01	11	11	01	01	01	00
10	01	01	00	01	01	00	01	01	01	01	01	00	00	11	11	01	01	01	00
10	10	10	00	10	00	10	00	00	10	10	10	10	10	10	10	11	11	00	10

# Vremenska redundancija

---

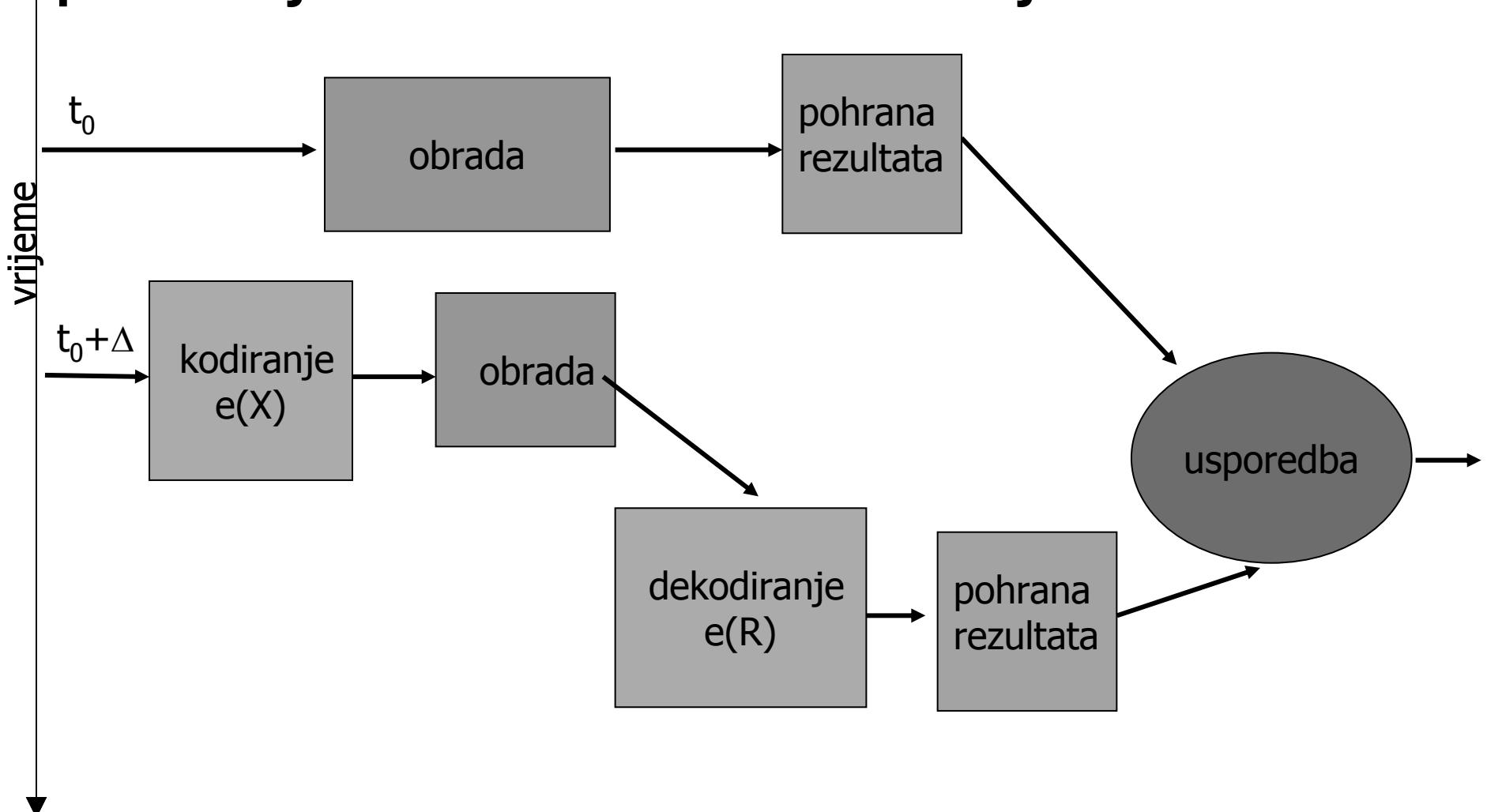
- smanjuje potrebno dodatno sklopolje na račun dodatnog vremena
- pogodna za prijelazne (tranzijentne) kvarove
- metoda: ponovljeno izračunavanje, pohrana rezultata, usporedba
- primjer: postoji detekcija pogreške i korekcija, ali se želi znati da li je kvar stalan ili prijelazni
- mora pamtitи rezultate

# Vremenska redundancija



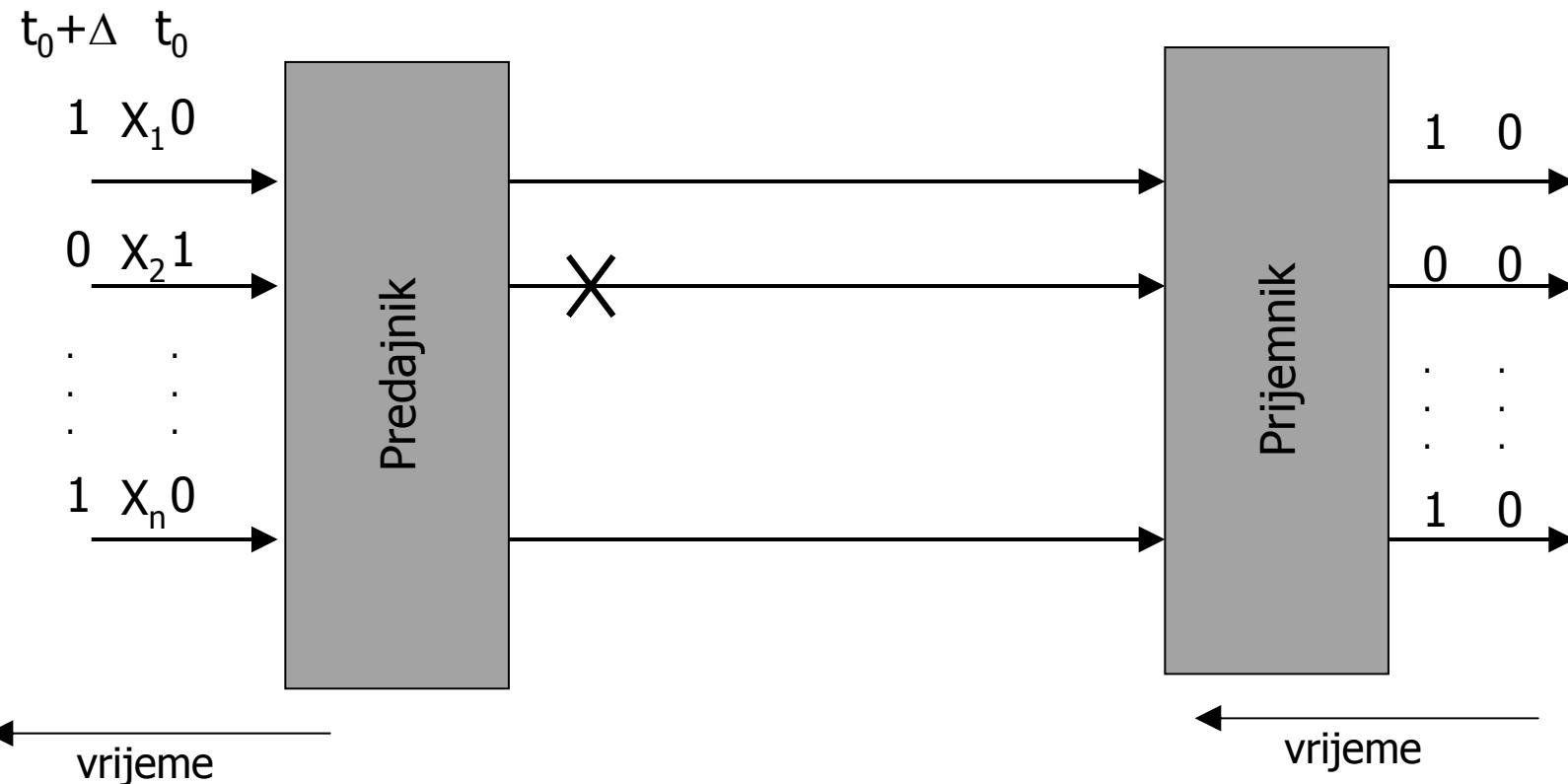
# Otkrivanje stalnih kvarova

## ■ proširenje vremenske redundancije



# Alternirajuća logika

- $t_0$  – prijenos izvornih podataka
- $t_0 + \Delta$  – prijenos komplementarnih podataka
- otkriva greške tipa zaglavljen-na na sabirnici



# primjer samo-dualnih logičkih funkcija

---

- *dualna funkcija f*

$$f_d = \overline{f(\overline{x_1}, \overline{x_2}, \dots, \overline{x_n})}$$

- *samodualna funkcija*

$$f_{sd}(x) = f(x)$$

- *vrijedi po definiciji*

$$f_{sd} = x_{n+1} f + \overline{x_{n+1}} f_d$$

- *komplementarni ulazi daju komplementarne izlaze*

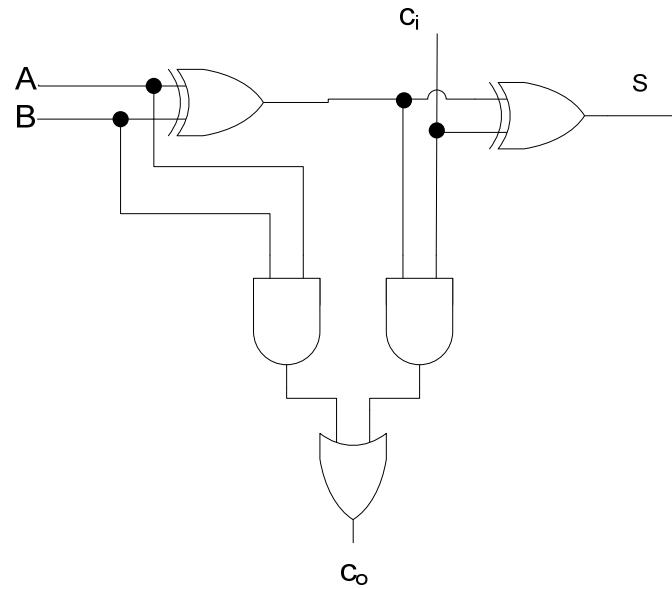
# uvjeti otkrivanja kvara alternirajuće logike

---

- otkriva kvar ako za svaki kvar najmanje jedna kombinacija daje nealternirajući izlaz
- Primjer potpuno zbrajalo
- Problem: može proteći neko vrijeme prije nego se pogodi ulazna kombinacija koja otkriva postojanje kvara

# Primjer: Potpuno zbrajalo

---



A	B	C <sub>i</sub>	S	C <sub>o</sub>
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	0

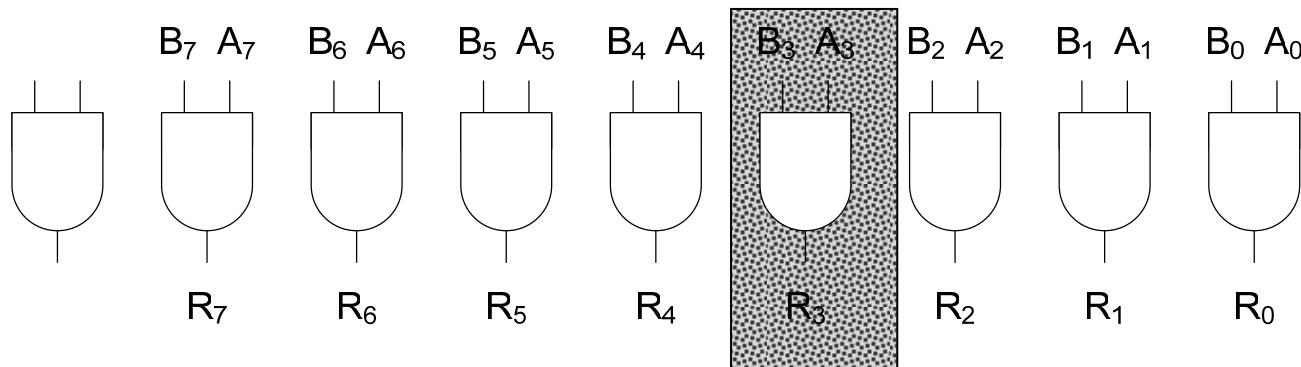
# Ponovno izračunavanje s pomaknutim operandima

---

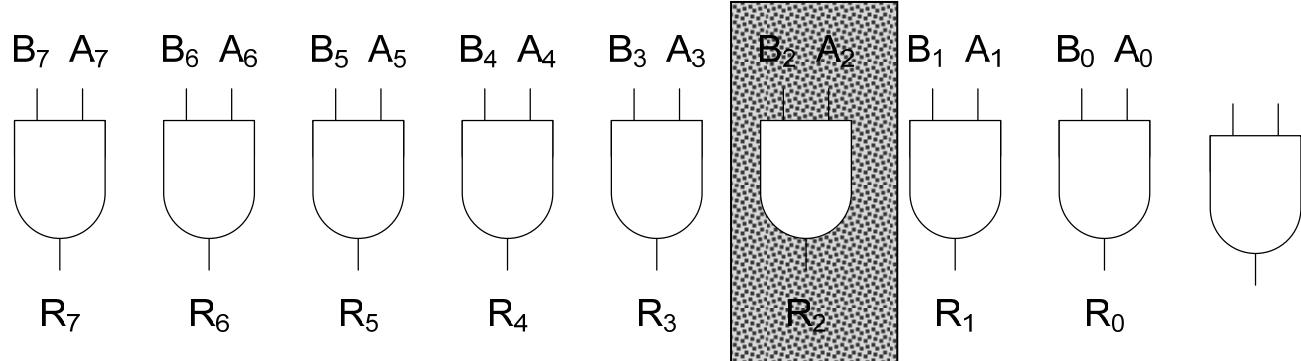
- Recomputing with Shifted Operands (RESO)
- razvijeno za ALU
- kodiranje – posmak lijevo
- dekodiranje – posmak desno
- Pogodno za bit-sliced, ripple-carry adder
  - neophodan 2-bitni posmak kod aritmetičkih operacija za detekciju pogreške
- Dodatno sklopolje:
  - 3 posmačna registra
  - registar
  - komparator
  - 2 dodatna bita u ALU

# RESO zatajenja

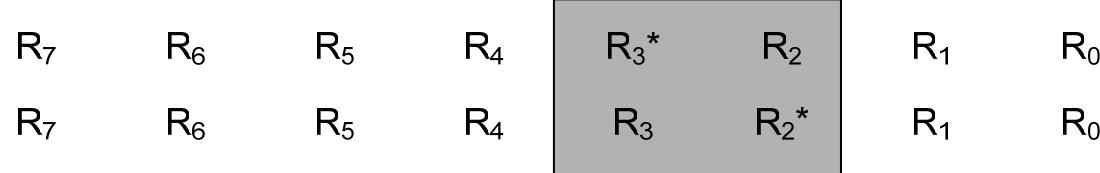
■  $t_0$



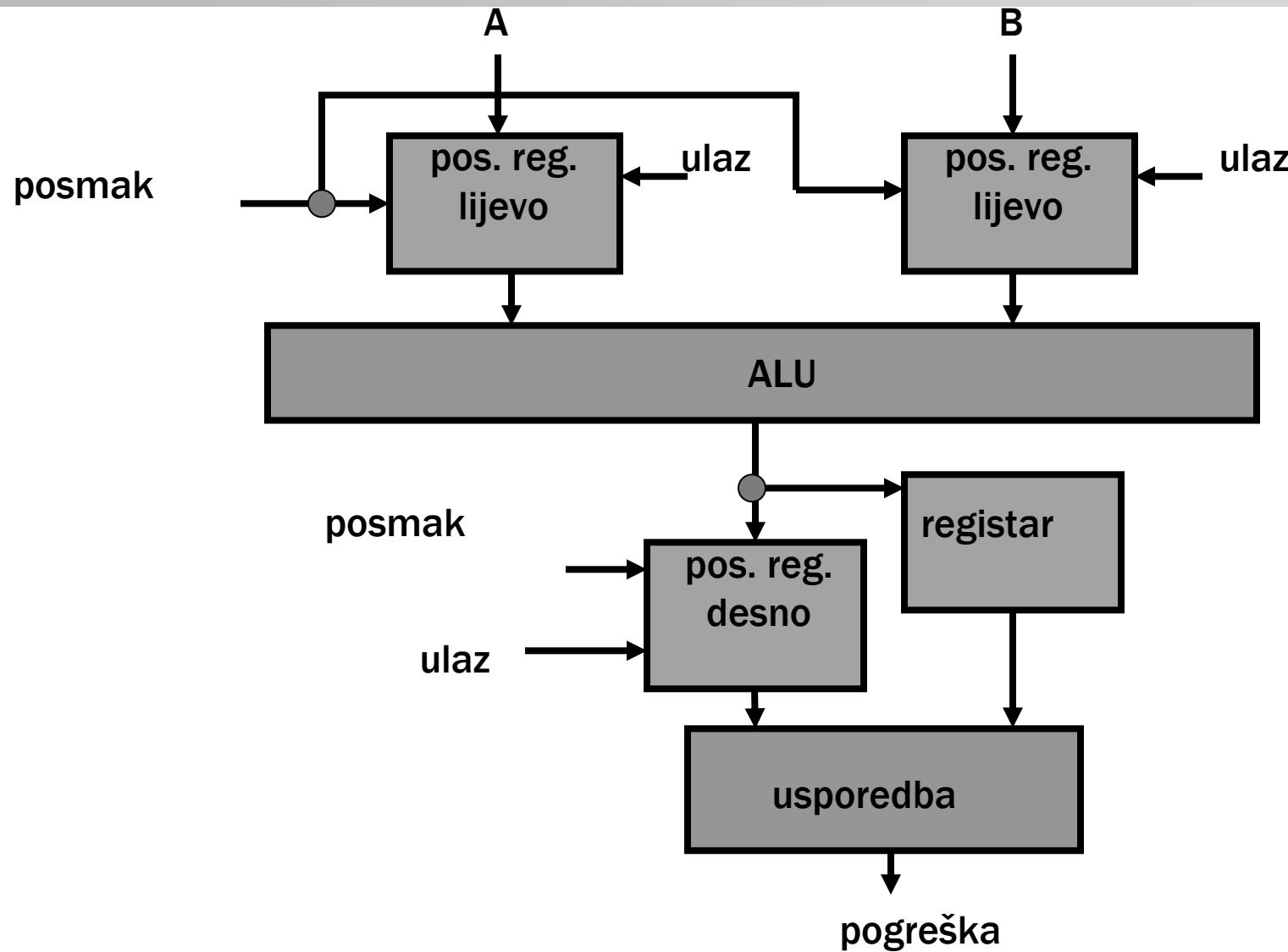
■  $t_0 + \Delta$



■ usporedba



# RESO primjer za ALU



# Problemi RESO

---

- dodatno sklopolje
- nema pokrivanja za posmake
- komparator mora biti potpuno samoispitni
- Poboljšanja uporabom kružnog posmaka
  - nisu potreban 2 dodatna bita u ALU
  - komplicira rukovanje prelivom
  - nije rašireno

# Ponovno izračunavanje s preokrenutim oper.

- engl. Recomputing with Swapped Operands (RESWO)
- Operandi zamjenjuju gornje i donje polovine kod ponovljene operacije



# RESWO analiza

---

## ■ Normalan način –

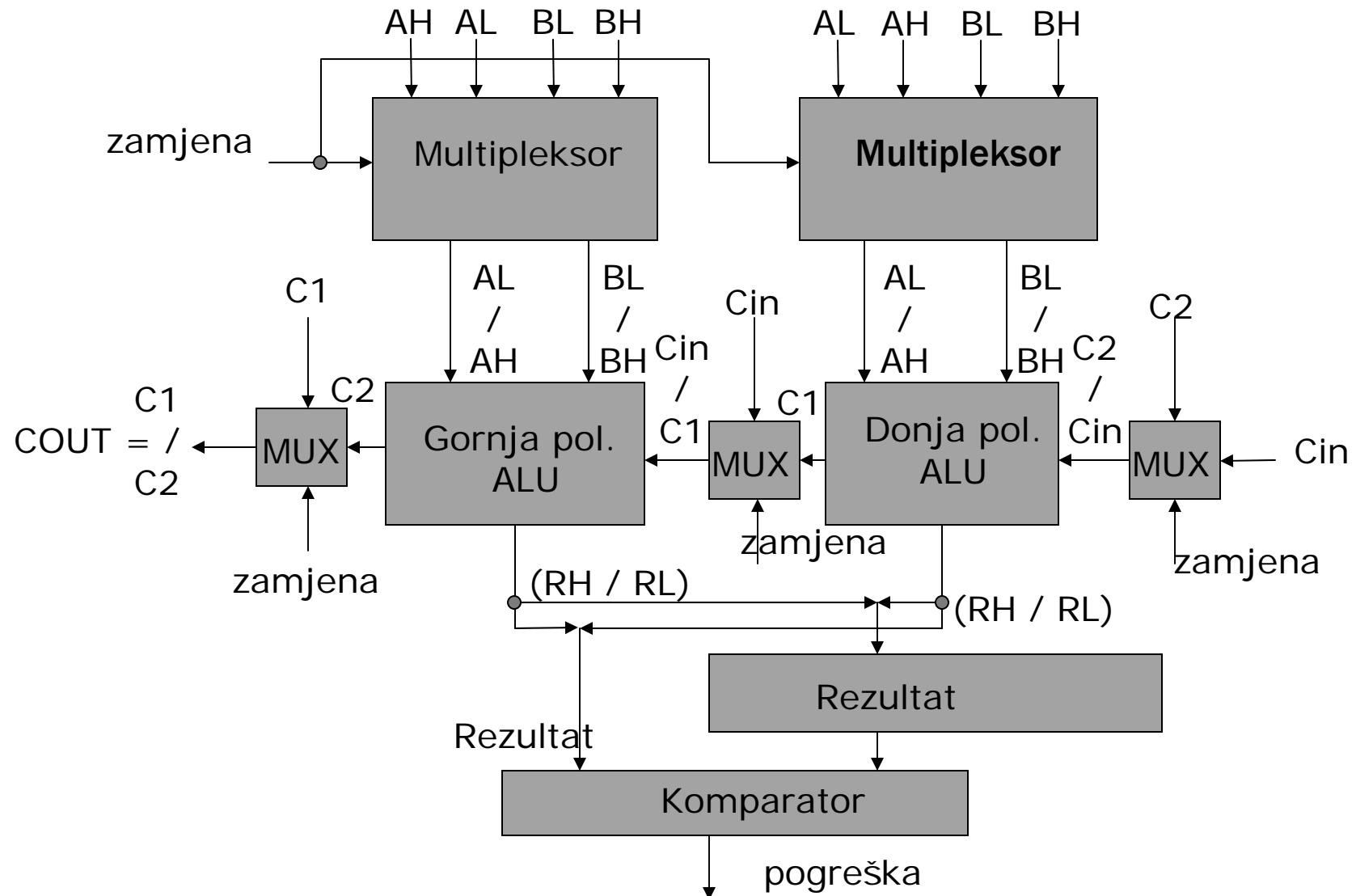
- Pogreška u bitu i utječe na sumu & preliv
- mijenja rezultat:

$$0, \pm 2^i, \pm 2^{i+1}, 2^i, \pm 2^{i+1}, \pm$$

## ■ Izračunavanje s preokrenutim operandima

- Pogreška u bitu i utječe na
- 0,  $\pm 2^{i+r+1}$ ,  $\pm 2^{i+r+2}$ ,  $\pm 2^{i+r+1}$ ,  $\pm 2^{i+r+2}$

# RESWO primjer ALU

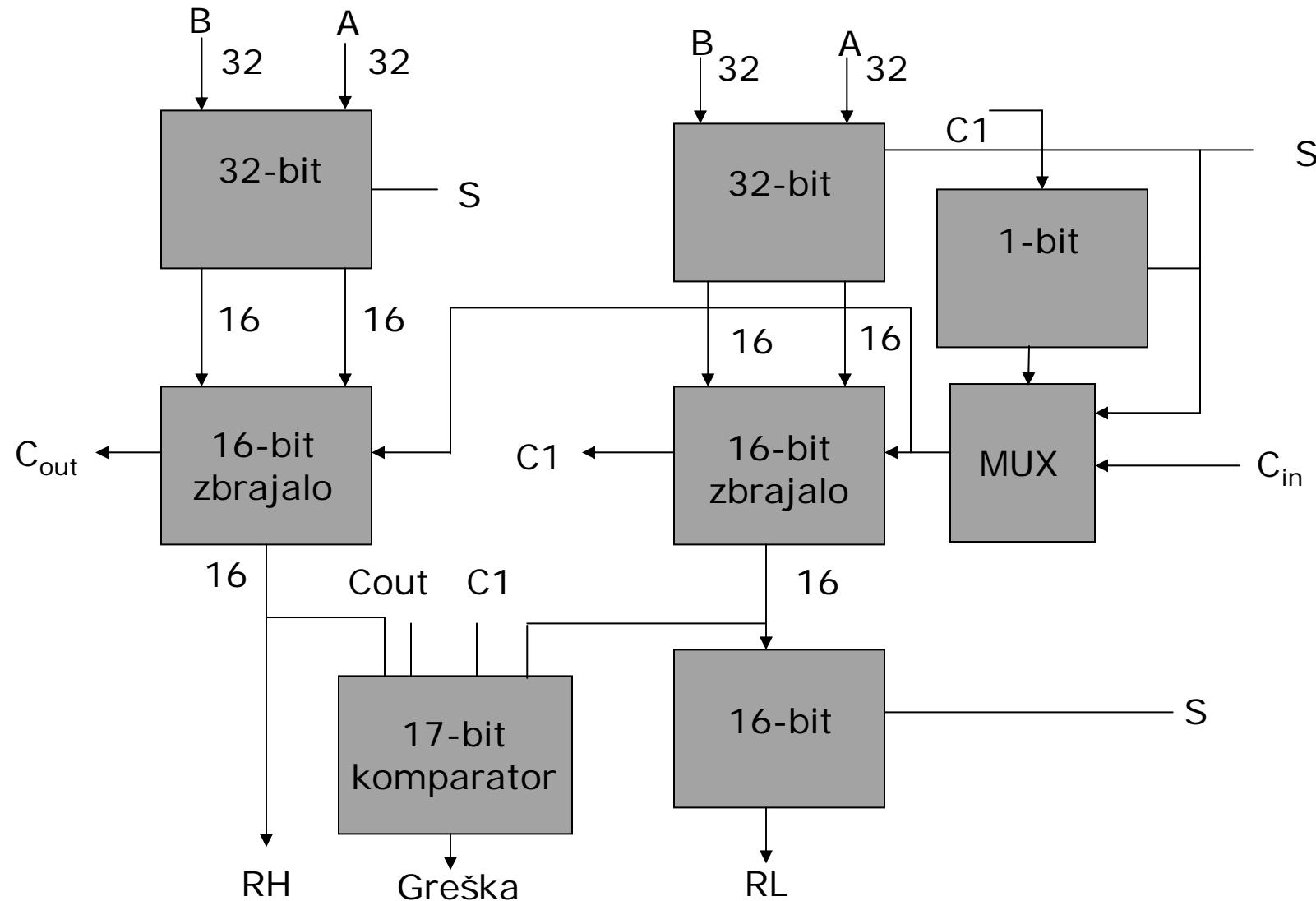


# Ponovno izračunavanje s dupliciranjem i uspor.

---

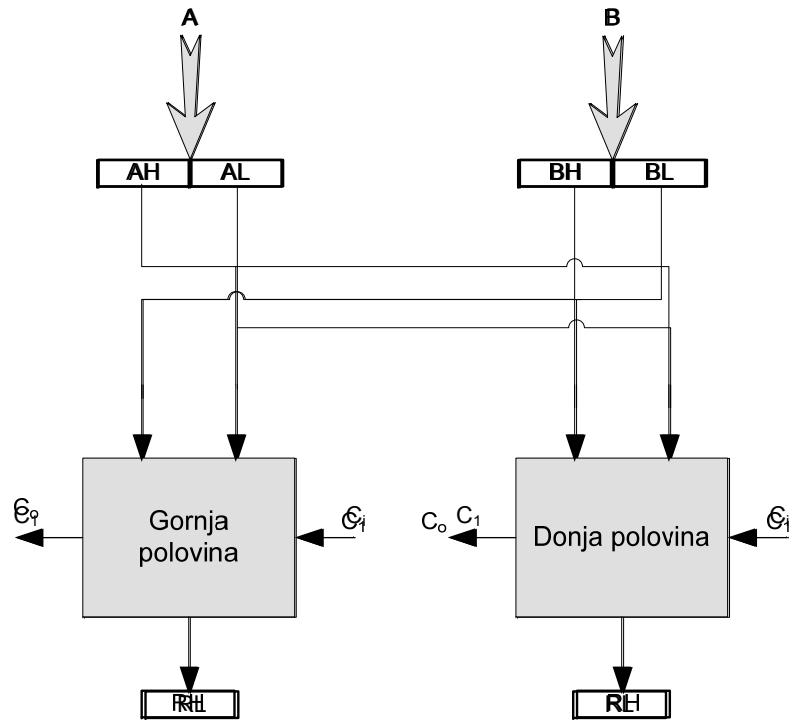
- Recomputing with Duplication with Comparison (REDWC)
- koristi vremensku i sklopošku redundanciju
- Primjer 32 bitno zbrajalo:
  - 2 16-bitna zbrajala
  - 1 izračun:
    - 2 - 16-bitna zbrajala zbrajaju donje polovine operanda
  - 2 izračun :
    - 2 - 16-bitna zbrajala zbrajaju donje gornje pol. operanda
    - koristi se preljev iz prve usporedbe
  - oba izračunavanja: 2 rezultata se uspoređuju a 1 pohranjuje

# Primjer 32-bit zbrajalo



# Primjer rada REDWC

- Isto sklopolje se upotrebljava 2 puta za usporedbu rezultata
  - Izračun donje polovice operanda
  - Izračun gornje polovice operanda
  - usporedba



- Iste mogućnosti detekcije kao i kod duplikacije s usporedbom
- Otkriva sve jednostruke kvarove u jednoj polovini zbrajala sve dok kvarovi nisu identični

# Primjer

---

- ispravljanje pogreške uporabom vremenske redundancije
- primjenjiva za logičku operacije I nad bitovima
- nije primjenjivo za aritmetičke operacije jer susjedni bitovi nisu neovisni
- Ponovno izračunavanje s dupliciranjem i usporedbom

# Pitanja...

# Pouzdanost računalnih sustava

---

## Predavanje 8: Informacijska redundancija

Prof.dr.sc. Vlado Sruk



Sveučilište u Zagrebu  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave*



# Sadržaj

---

## ■ Kodovi

# Informacijska redundancija

---

- Osnovni koncept dodavanje redundancije/zalihosti podacima
- Uporaba kodova
  - Otkrivanje pogrešaka
  - Ispravljanje pogrešaka

# Kodovi

---

- predstavljanje informacija uporabom dobro definiranih pravila
  - Binarni kod
    - jednostruka greška proizvodi valjan kodnu riječ koja više nije ispravna
- Teorija kodiranja
  - binarne kodne riječi podskup ukupnog broja svih kombinacija 0 i 1
- Kodna riječ – skup simbola koji predstavljaju određenu informaciju ovisno o odabranom kodu
- *Kodiranje*
- *Dekodiranje*

# Kodovi s otkrivanjem pogreške

---

- engl. Error Detecting Code EDC
- omogućava otkrivanje pogreške u kodnoj riječi
- otkriva pogreške u kodnoj riječi →  
rekonfiguracija

# Kodovi s ispravljanjem pogreške

---

- engl. Error Correcting Code ECC
  - omogućava određivanje ispravne kodne riječi iz riječi s pogreškom
  - maskiranje pogreške
- Ispravljanje pogreške jednog bita (engl. Single-error correcting code)
- Ispravljanje pogreške dva bita (engl. Double-error correcting code)

# Hammingova udaljenost ( $H_d$ )

---

- udaljenost dvije binarne riječi predstavlja broj broj mesta na kojima se bitovi razlikuju:
  - 0000<->0101 → 2
- udaljenost koda – minimalna Hammingova udaljenost između bilo koje dvije kodne riječi
  - = 2:
    - jednostruka pogreška → nevažeća kodna riječ
  - = 3:
    - dvostruka pogreška → nevažeća kodna riječ
    - omogućava ispravljanje jednostrukih pogrešaka → neispravna riječ ima udaljenost 1 od ispravne kodne riječi a 2 od svih ostalih

# Svojstva kodova

---

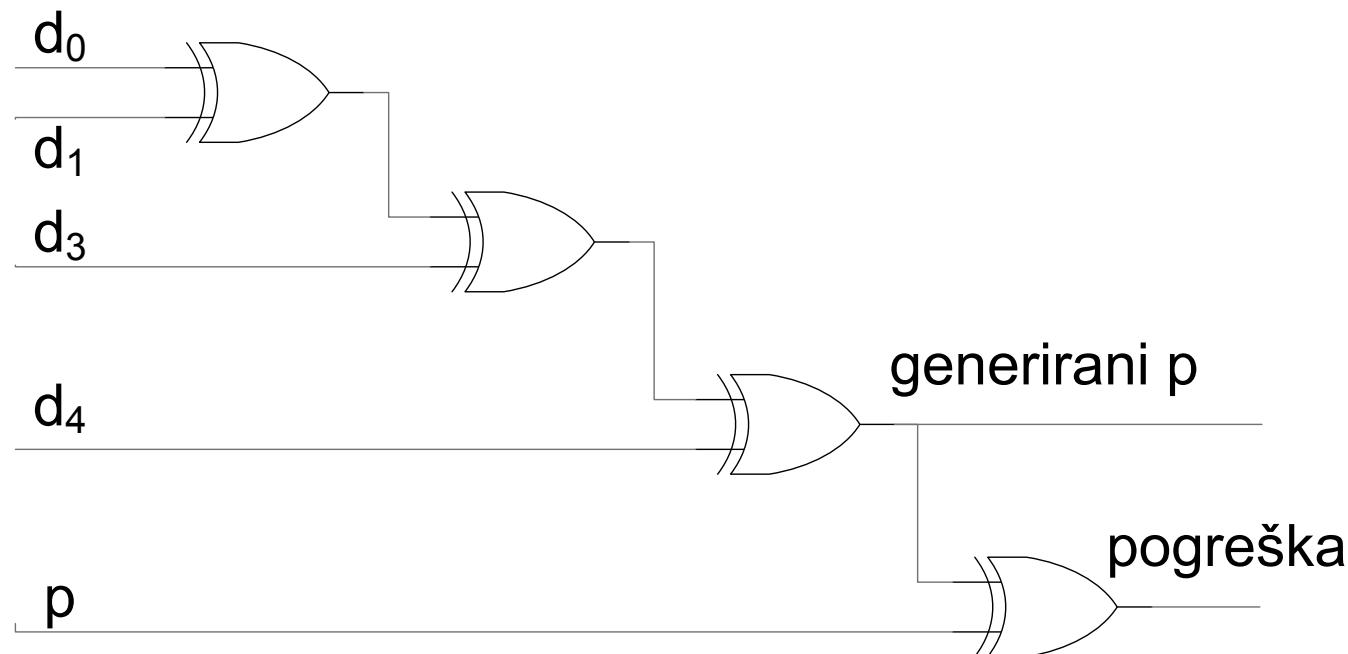
- Mogućnost ispravljanja pogrešaka
  - može ispraviti  $c$  pogrešaka i otkriti dalnjih  $d$  pogrešaka akko vrijedi  $2c+d+1 \leq H_d$
  - $H_d=2$  otkriva sve jednostruke pogreške
- Separabilni kodovi
  - za stvaranje kodne riječi originalnoj informaciji dodaju se nove informacije
    - Dekodiranje = uklanjanje dodanih informacija
  - jednostavnije dekodiranje od neseparabilnih kodova

# Paritetni kod

## ■ Parity Codes Even/Odd

## ■ Paritetni bit

- generira se pri upisu podatka
- provjerava pri čitanju



# Osnovni oblici paritetnog koda

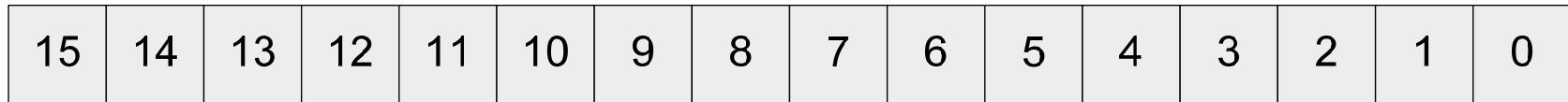
---

- paran, neparan
- Bit po riječi
- Bit po oktetu
- Bit na više čipova
- Bit po čipu
- Isprepleteni paritet
- Preklapajući paritet

# bit po riječi

---

## ■ engl. Bit-per-word



## ■ ne otkriva velik broj kvarova

- npr. riječ i paritet → 1

# bit po oktetu

---

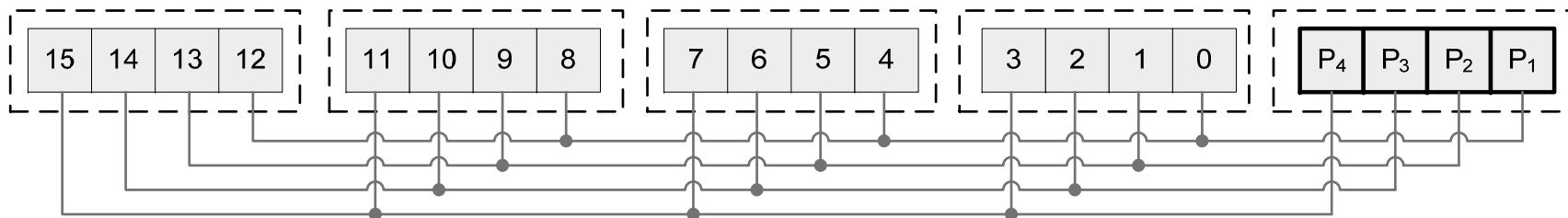
## ■ engl. Bit-per-byte



- broj bitova po paritetu treba biti paran
- pariteti pojedinih grupa naizmjenično parni i neparni
  - omogućava otkrivanje kvarova tipa sve 0 ili 1
- ne otkriva velik broj višestrukih kvarova

# bit na više čipova

## ■ engl. Bit-per-multiple-chips

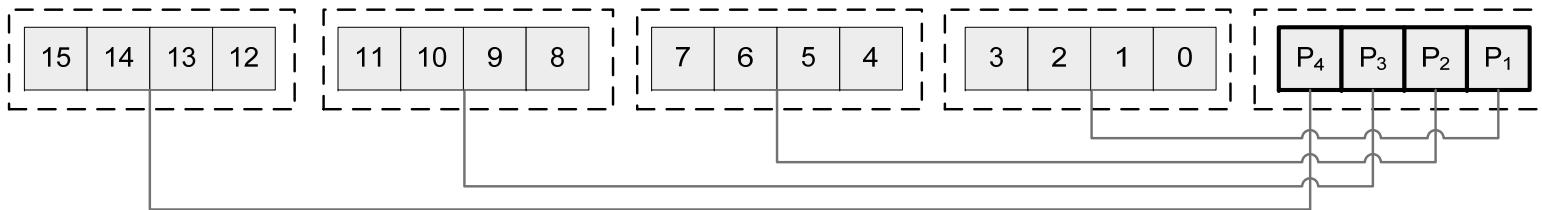


- pridružuje paritetni bit svakom čipu
- svaki bit u čipu ima svoj paritet
  - kvar jednog čipa utječe na sve grupe
  - maksimalno 1 bit u grupi

# paritetni bit po čipu

---

## ■ engl. Bit-per-chip parity

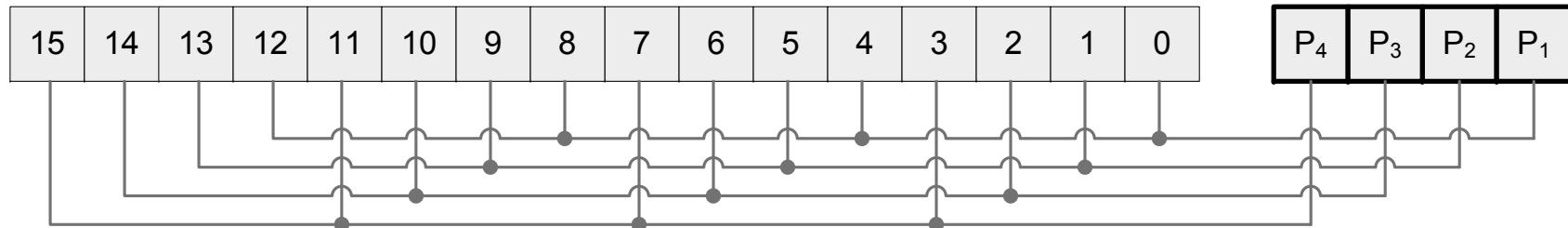


- paritetni bit za svaki od čipova
- ne otkriva višestruke kvarove ili kvar cijelog čipa

# isprepleteni paritet

## ■ engl. Interlaced Parity

- dijeli informaciju u grupe jednake veličine
- svakoj grupi pridružuje jedna paritetni bit
- neovisan o fizičkoj organizaciji
- susjedni bitovi u različitim grupama



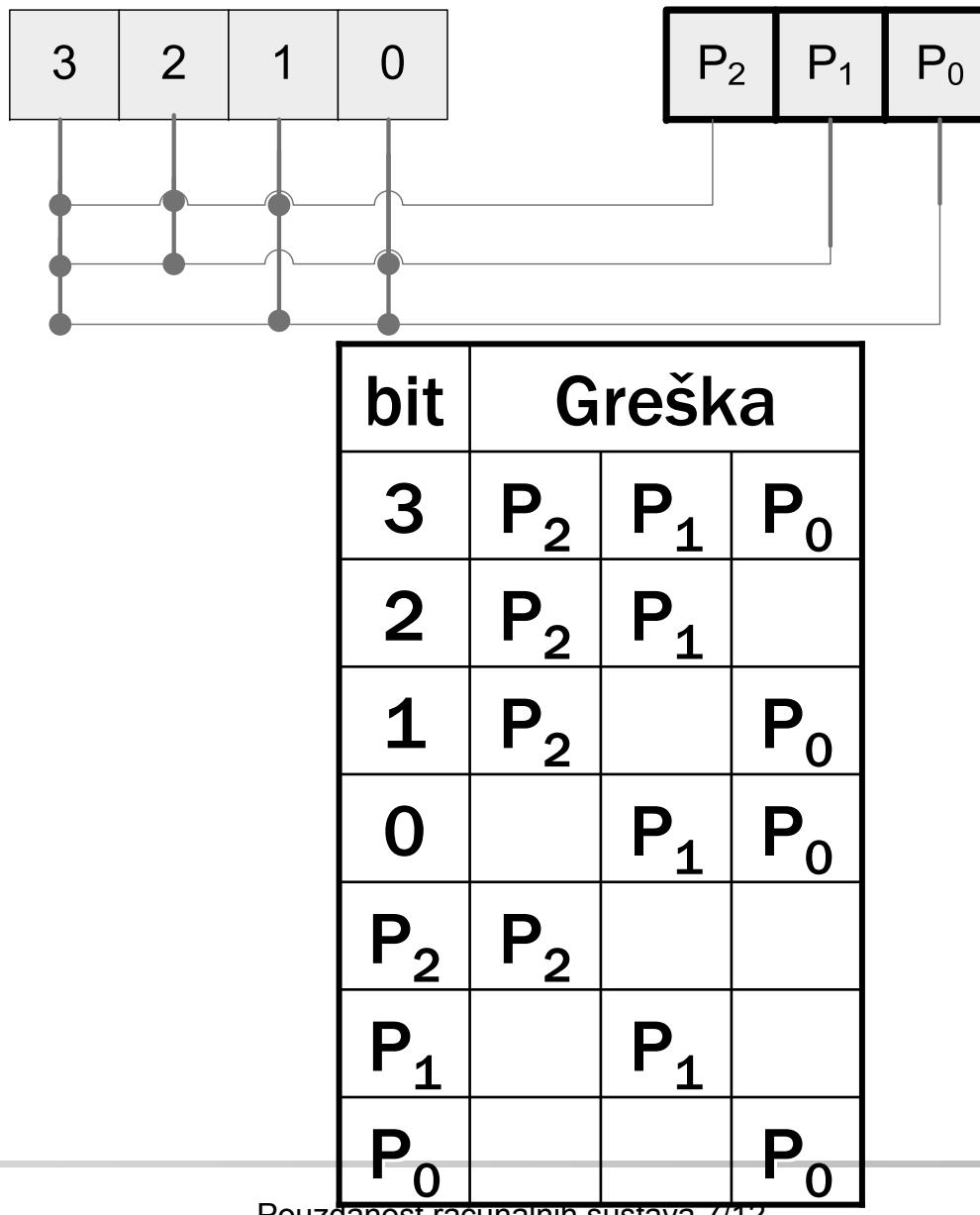
# preklapajući paritet

---

- engl. Overlapping Parity
- svaki bit informacije sudjeluje u stvaranju pariteta više grupa
  - otkriva i locira kvar
  - ispravlja pogrešku komplementiranjem bita s pogreškom
  - upotrebljava se u nekim Hammingovim kodovima
  - Za  $k$  informacijskih bitova i  $c$  bitova zaštite
    - $2^c \geq c+k+1$

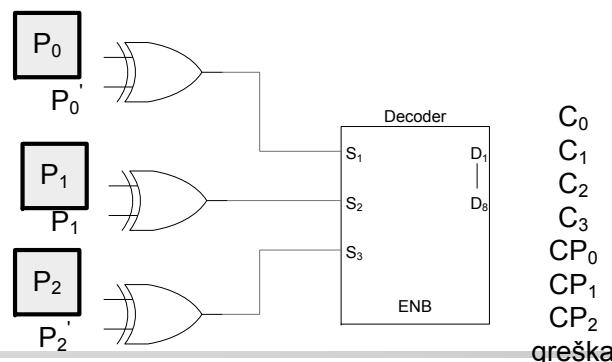
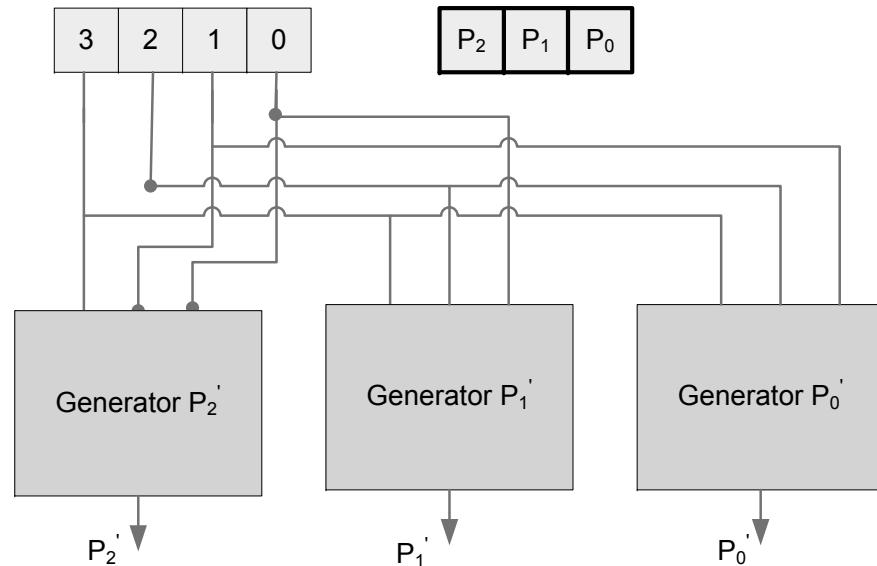
$$n=c+k$$

# Primjer preklapajućeg pariteta



# Ispravljanje pogreške

## ■ uporabom preklapajućeg pariteta



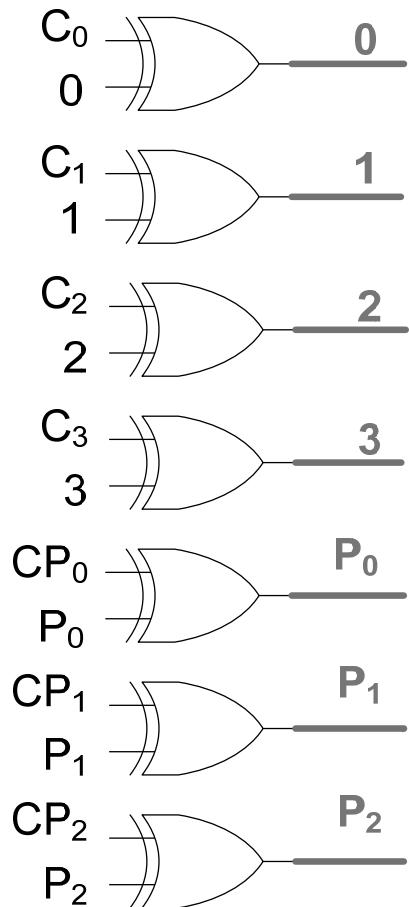
# Sindrom

---

- izlazi dekodera 3-od-8 naziva se sindrom
  - pokazuje promjene bita
- nedostatak -visoka cijena
  - 75% redundantnosti (3 /4 bita)
  - cijena pada s rastom broja informacija

# Ispravljanje pogreške

---



# Izračun broja paritetnih bitova

---

- $m$  – broj bitova koji se štite
  - $k$  – broj paritetnih bitova
  - $m + k$  – broj različitih jednostrukih pogrešaka
  - $2k$  – broj provjera pariteta
  - $m + k + 1$  – broj jedinstvenih kombinacija potrebnih za jednoznačno određivanje rezultata provjere pariteta
- ⇒ neophodan uvjet:  $2k \geq m + k + 1$

# minimalan broj paritetnih bitova

---

inf. bitova	paritet	% redund.
2	3	150
4	3	75
6	4	66.7
8	4	50
10	4	40
12	5	41.7
16	5	31.25
24	5	20.8
32	6	18.75
64	7	10.9

# Kodovi $m$ -od- $n$

- $n$  bita
- točno  $m$  jedinica u svakoj kodnoj riječi
- Jednostruka greška –
  - $m+1$  ili  $m-1$  jedinica
- Nedostatak
  - Složenost kodiranja, dekodiranja, otkrivanje
- Najjednostavnija implementacija
  - Dodavanje  $i$  bitova na  $i$  bitova originalne informacije

Inform.	3-od-6	
000	000	111
001	001	110
010	010	101
011	011	100
100	100	011
101	101	010
110	110	001
111	111	000

# svojstva koda m-od-n

---

## ■ separabilni kod

- jednostavno kodiranje i dekodiranje
- $H_d=2$
- otkrivanje jednostrukih pogrešaka
- ne mora otkriti sve dvostruke pogreške

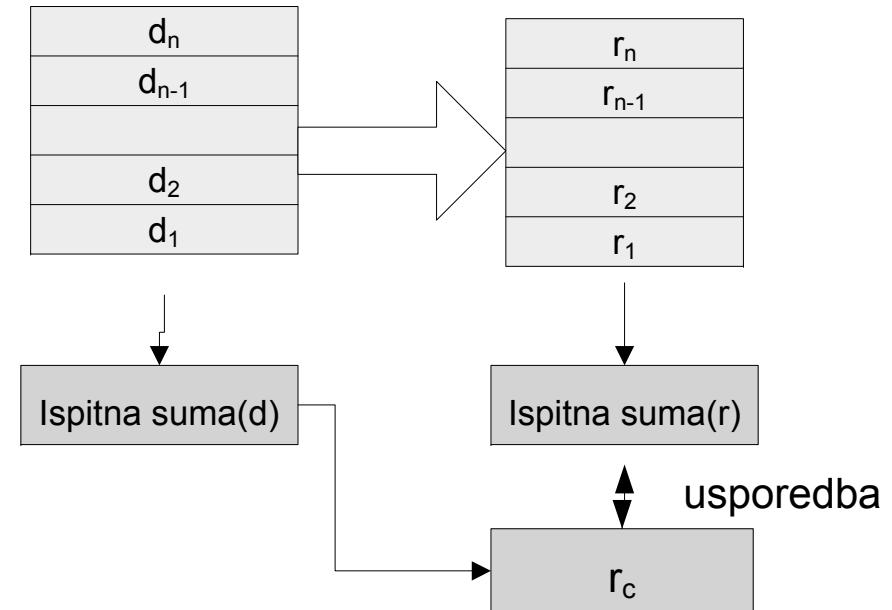
# Duplikacijski kod

---

- Potpuno duplicitiranje podatka
  - Jednostavno, velika zalihost
  - Nepovoljno
- *Komplementirana duplikacija*
  - engl. Complemented Duplication
  - Otkriva zajedničke kvarove bitova
- *Zamjena i usporedba*
  - engl. Swap and Compare
  - Dvije kopije s zamjenom gornje i donje polovine riječi

# Ispitne sume

- separabilni kod – pogodan za prijenos podataka
- Jednostruka preciznost
  - Binarno zbrajanje, ignoriran preliv, n
  - Ne detektira sve kvarove
- Dvostruka preciznost
  - $2n$  ; problem preliva?
  - Ne mogu locirati kvar



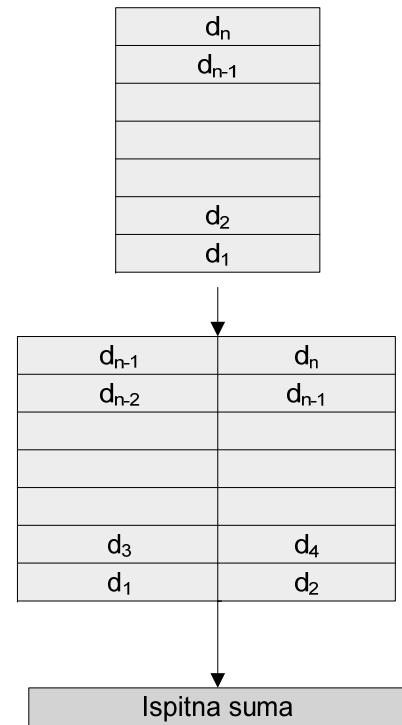
# Tipovi ispitnih suma

---

- jednostruka preciznost
  - izračun binarnim zbrajanjem svih informacija
  - ignorira preliv
  - gubitak informacija → ne detektira sve pogreške
- dvostruka preciznost
  - 2n-bitna ispitna suma
  - preliv ...
- Honeywell ispitna suma
  - spaja susjedne riječi u dvostruku riječ ( $2n$  bita)
  - ispitna suma  $2n$  bita
  - pogreška u istom bitu utječe na 2 mesta ispitne sume
- ispitna suma s ostatom - Residue checksum
  - isto kao i jednostruka preciznost ali preljev se dodaje na kraj

# Honeywell ispitna suma

---



# Ciklički kodovi

---

- Pomakom kodne riječi dobiva se nova kodna riječ
- Svojstva koda ovise o generatoru polinoma
- Generator polinoma  $G(x)$  stupnja  $\geq n-k$ 
  - $n$  – ukupan broj bitova
  - $k$  – broj bitova originalne informacije
  - $G(x)$  koeficijenti 0 ili 1 za binarne kodne riječi
- Ciklički kod dobiven generatorom polinoma  $n-k$ , naziva se  $(n, k)$  ciklički kod
- otkriva sve jednostrukе i sve višestruke uzastopne pogreške na  $n-k$  bita
- Podaci predstavljeni polinomom
- stvara se množenjem kodnog generatora polinoma s polinomom koji predstavlja podatke

# primjer

---

- 10112

$$1 * r^3 + 0 * r^2 + 1 * r^1 + 1 * r^0$$

- $r=2$ , + modulo 2 zbrajanje
- \* decimalno množenje
- kodnu riječ predstavljaju koeficijenti polinoma
  - Kodni polinom generiramo množenjem informacije u obliku polinoma s generatorom polinoma

$$v = v_0 + v_1 x + v_2 x^2 + \dots + v_{n-1} x^{n-1}$$

# primjer

---

- binarna riječ (**1101**)

$$D(x) = 1 + x + x^3$$

- Generator polinoma:

$$G(x) = 1 + x + x^3$$

- Kodni polinom:

$$v(x) = D(x) * G(x) =$$

$$(1 + x + x^3)(1 + x + x^3)$$

$$= 1 + x + x^3 + x + x^2 + x^4 + x^3 + x^4 + x^6$$

$$= 1 + 0 * x + x^2 + 0 * x^3 + 0 * x^4 + x^6$$

- Koeficijenti kodnog polinoma = (**1010001**)

# Linearni blok kod

---

## ■ engl. Linear block code

- Moguće predstaviti u matričnom obliku
- G-matrica - posmak generatora polinoma

$$G = \begin{bmatrix} g_{n-k} & \dots & g_1 & g_0 & 0 & 0 & 0 \\ 0 & g_{n-k} & \dots & g_1 & g_0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \dots & g_{n-k} & \dots & g_1 & g_0 \end{bmatrix}$$

## Primjer 2

---

$$G(x) = 1 + x + x^3$$

$$(1\ 1\ 1\ 1) = D(x) = 1 + x + x^2 + x^3$$

$$V(x) = D(x) * G(x)$$

$$= (1 + x + x^2 + x^3)(1 + x + x^3)$$

$$= 1 + x + x^3$$

$$x + x^2 + \quad + x^4$$

$$x^2 + x^3 + \quad x^5$$

$$x^3 + x^4 + \quad x^6$$

$$1+0*x+0*x^2+1*x^3+0*x^4+x^5+x^6$$

■ **V=(1001011)**

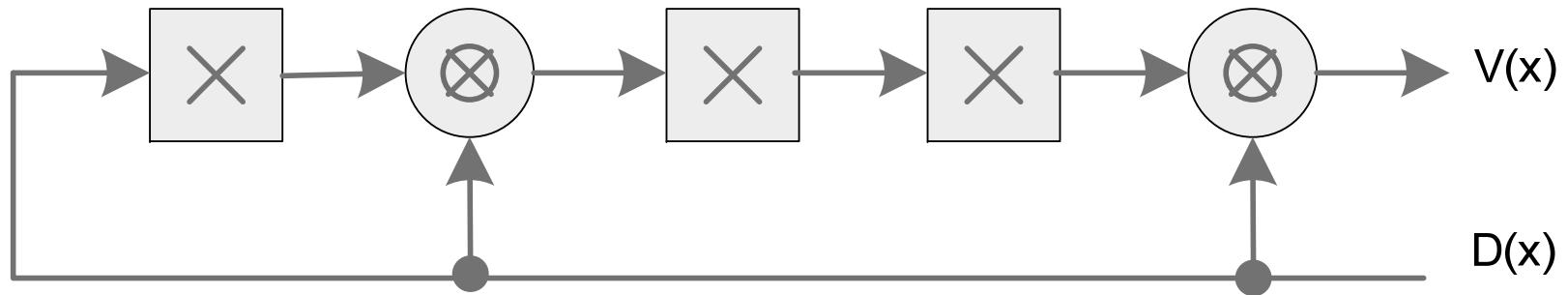
---

# ciklički kod za 4-bitu

<b>Informacija</b>	<b>D(x)</b>	<b>G(x)</b>	<b>V(x)</b>	<b>Kod</b>
0000	0	$x^2 + 1$	0	000000
0001	1	$x^2 + 1$	$x^2 + 1$	000101
0010	x	$x^2 + 1$	$x^3 + x$	001010
0011	$x + 1$	$x^2 + 1$	$x^3 + x^2 + x + 1$	001111
0100	$x^2$	$x^2 + 1$	$x^4 + x^2$	010100
0101	$x^2 + 1$	$x^2 + 1$	$x^4 + 1$	010001
0110	$x^2 + x$	$x^2 + 1$	$x^4 + x^3 + x^2 + x$	011110
0111	$x^2 + x + 1$	$x^2 + 1$	$x^4 + x^3 + x + 1$	011011
1000	$x^3$	$x^2 + 1$	$x^5 + x^3$	101000
1001	$x^3 + 1$	$x^2 + 1$	$x^5 + x^3 + x^2 + 1$	101101
1010	$x^3 + x$	$x^2 + 1$	$x^5 + x$	100010
1011	$x^3 + x + 1$	$x^2 + 1$	$x^5 + x^2 + x + 1$	100111
1100	$x^3 + x^2$	$x^2 + 1$	$x^5 + x^4 + x^3 + x^2$	111100
1101	$x^3 + x^2 + 1$	$x^2 + 1$	$x^5 + x^4 + x^3 + 1$	111001
1110	$x^3 + x^2 + x$	$x^2 + 1$	$x^5 + x^4 + x^2 + x$	110110
1111	$x^3 + x^2 + x + 1$	$x^2 + 1$	$x^5 + x^4 + x + 1$	110011

# Sklop za stvaranje cikličke kodne riječi

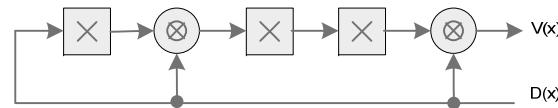
## ■ Generator kodne riječi



$$G(x) = 1 + x + x^3$$

$$D(x) = 1 + x^2 + x^3$$

# primjer rada



takt	r1	r2	r3	D(X)	V(X)
0	0	0	0		
				1	1
1	1	0	1		
				1	0
2	1	1	1		
				0	1
3	0	1	1		
				1	0
4	1	0	0		
				0	0
5	0	1	0		
				0	0
6	0	0	1		
				0	1
7	0	0	0		

# Dekodiranje

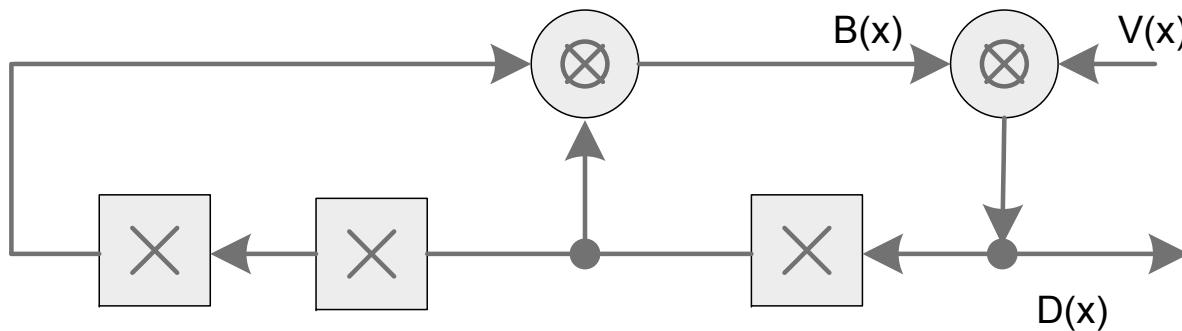
---

- Neseparabilni kod
- Provjera ispravnosti kodne riječi  $(r_0, r_1, r_2, \dots, r_{n-1})$
- Polinom:  
 $r_0 + r_1 x + r_2 x^2 + \dots r_{n-1} x^{n-1}$
- pretpostavka da je dobiven od:
  - $R(x) = D(x) G(x) + S(x)$
  - $S(x)=0$  akko je kodna riječ ispravna
- Sindrom  $S(x)$ 
  - Provjera ispravnosti  $S(x)= 0$
- Za provjeru ispravnosti:  
 $R(x) / G(x)$

# sklop za dijeljenje

---

- $B(x) = (x^3 + x) D(x)$
- $V(x)$  – kodirani podaci
- $D(x)$  – dekodirani podaci
- Modulo 2 aritmetka



# Provjera ispravnosti dijeljenja

---

$$V(x) + B(x) = D(x)$$

$$V(x) = D(x) - B(x) = D(x) - (x^3 + x)D(x)$$

$$V(x) = (x^3 + x + 1)D(x)$$

$$D(x) = \frac{V(x)}{x^3 + x + 1}$$

## ■ Primjer:

$$\begin{array}{r} 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ + 0 & - 0 & + 1 & - 1 & + 0 & - 0 & + 1 \\ \hline 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{array}$$

---

# dekodiranje

---

takt	r1	r2	r3	V(X)	B(X)	D(X)
0	0	0	0			
				1	0	1
1	0	0	1			
				0	1	1
2	0	1	1			
				1	1	0
3	1	1	0			
				0	1	1
4	1	0	1			
				0	0	0
5	0	1	0			
				0	0	0
6	1	0	0			
				1	1	0
7	0	0	0			

# primjer dekodiranja

- kodna riječ **1010001**
- kodna riječ s pogreškom  
**1011001**

takt	r1	r2	r3	V(X)	B(X)	D(X)
0	0	0	0			
				1	0	1
1	0	0	1			
				0	1	1
2	0	1	1			
				1	1	0
3	1	1	0			
				1	1	1
4	1	0	0			
				0	1	1
5	0	0	1			
				0	1	1
6	0	1	1			
				1	1	0
7	1	1	0			

# Svojstva cikličkih kodova

---

- Nisu separabilni
- Nelson & Carrole – mogućnost izgradnje i separabilnih
- Proces kodiranja ( $n, k$ )
  - $D(x)$  množiti s  $x^{n-k}$
  - Rezultat podijeliti s  $G(x) \Rightarrow R(x)$
- Odabir generatora polinoma utječe na karakteristike koda
  - Prvi i zadnji bit = 1
    - Otkriva slijedne pogreške duljine  $\leq n-k$
  - Ako je generator polinoma umnožak  $(x+1)$ 
    - Otkriva neparan broj pogrešaka
  - Ako je  $g(x) = (x+1)p(x)$  pri čemu vrijedi  $p(x)$  je stupnja  $n-k-1$  i  $n < 2^{n-k-1}$ 
    - Otkriva jednostrukе, dvostrukе, trostrukе i neparne pogreške

# Uporaba CRC kodova

---

NAZIV	$G(x)$
CRC-5 (USB token packets)	$x^5 + x^2 + 1$
CRC-12 (Telecom systems)	$x^{12} + x^{11} + x^3 + x^2 + x + 1$
CRC-16-CCITT (X25, Bluetooth)	$x^{16} + x^{12} + x^5 + 1$
CRC-32 (Ethernet)	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x + 1$
CRC-64 (ISO)	$x^{64} + x^4 + x^3 + x + 1$

# Bergerov kod

- dodaje ispitni bit svakoj riječi informacije
  - kod duljine  $n$  ima:
    - $I$  – bitova informacije
    - $k$  – ispitnih bitova
$$k = \log_2(I + 1)$$
- $$n = I + k$$

- Algoritam

1. Prebroji broj jedinica u  $I$
2. Komplementiraj i dodaj na kraj

i	k	% redund.
4	3	75.00%
8	4	50.00
16	5	31.25
32	6	18.75
64	7	10.94

# Bergerov kod za 4 bitne riječi

---

Original information	Berger code		
0000	0000	111	
0001	0001	110	
0010	0010	110	
0011	0011	101	
0100	0100	110	
0101	0101	101	
0110	0110	101	
0111	0111	100	
1000	1000	110	
1001	1001	101	
1010	1010	101	
1011	1011	100	
1100	1100	101	
1101	1101	100	
1110	1110	100	
1111	1111	011	

# Horizontalni i vertikalni paritet

---

- 0001 0
- 1001 1
- 1010 1
- 1111 1
- 0010 0      ←  
↑
- otkrivanje i ispravak jednostrukе pogreške
- Nije pogodno za višestrukе pogreške
  - Moguće otkrivanje
  - Nije moguć ispravak
    - Npr. Pogreška pariteta u retcima g i h, te stupcima k i m
    - (g,k), (g,m), (h,k), (h,m)

# Dvostruka paritetna zaštita

---

- višestruko ispitivanje pariteta :
  - zahtjev: povećati moć zaštite!
    - veći broj paritetnih ispitivanja
    - veći broj zaštitnih bitova - veća zalihost
- dvodimenzijski kod - 2D matrica
- vodoravna i okomita paritetna zaštita:
  - kodna riječ ← paritetni bit
  - cijelom bloku kodnih riječi ← paritetna riječ, BCC (engl. Block Check Character)
    - "horizontalna" (vodoravna) paritetna zaštita  
(engl. Longitudinal Redundancy Check, LRC)
    - "vertikalna" (okomita) paritetna zaštita  
(engl. Vertical Redundancy Check, VRC)
  - ispravljanje jednostrukih pogreške

# Hammingovi kodovi

---

- nezavisna paritetna ispitivanja
  - ne mogu se dobiti kombinacijom preostalih
- princip izgradnje kodne riječi:
  - "nezavisna" (ortogonalna) ispitivanja
  - "nezavisni" (ortogonalni) smještaj zaštitnih bitova
- "nezavisna" (ortogonalna) ispitivanja:
  - svaki zaštitni bit "pokriva" (= štiti) drugi podskup bitova podatka
  - svaki bit podatka zaštićen s više zaštitnih bitova

# Hammingovi kodovi

---

## ■ Hammingovi kodovi:

### ■ odnos zaštitnih i informacijskih bitova :

$$2^r \geq k + r + 1, \quad n = k + r$$

r: broj zaštitnih bitova

k: broj informacijskih bitova

n: duljina kodne riječi

### ■ obrazloženje:

■ jednostruka pogreška na jednom od n mesta

■ bez pogrešaka

# Hammingovi kodovi

- izgradnja Hammingovog koda za ispravljanje jednostrukih pogreški :
  - zaštitni bitovi na mesta koja se ne mogu dobiti kombinacijama drugih zaštitnih bitova:  $2^i$
  - zaštitni bitovi "pokrivaju" svoju poziciju
    - ~ sve pozicije čiji redni broj sadrži  $2^i$

POZICIJA	nema pogreške	1	2	3	4	5	6	7
$C_2$	0	0	0	0	1	1	1	1
$C_1$	0	0	1	1	0	0	1	1
$C_0$	0	1	0	1	0	1	0	1
		$C_0$	$C_1$	$k_1$	$C_2$	$k_2$	$k_3$	$k_4$

- zaštitni bitovi:  $C_2 \ C_1 \ C_0$

# Hammingovi kodovi

- Hammingov kod za ispravljanje jednostrukih pogreški :
  - izračunavanje zaštitnih bitova, za parni paritet

POZICIJA	nema pogreške	1	2	3	4	5	6	7
C <sub>2</sub>	0	0	0	0	1	1	1	1
C <sub>1</sub>	0	0	1	1	0	0	1	1
C <sub>0</sub>	0	1	0	1	0	1	0	1
		C <sub>0</sub>	C <sub>1</sub>	k <sub>1</sub>	C <sub>2</sub>	k <sub>2</sub>	k <sub>3</sub>	k <sub>4</sub>

$$\begin{array}{l} C_0 \oplus k_1 \oplus k_2 \oplus k_4 = 0 \\ C_1 \oplus k_1 \oplus k_3 \oplus k_4 = 0 \\ C_2 \oplus k_2 \oplus k_3 \oplus k_4 = 0 \end{array} \rightarrow \begin{array}{l} C_0 = k_1 \oplus k_2 \oplus k_4 \\ C_1 = k_1 \oplus k_3 \oplus k_4 \\ C_2 = k_2 \oplus k_3 \oplus k_4 \end{array}$$

# Hammingovi kodovi

- Hammingov kod za ispravljanje jednostrukih pogreški :
  - izračunavanje zaštitnih bitova, za neparni paritet

POZICIJA	nema pogreške	1	2	3	4	5	6	7
$C_2$	0	0	0	0	1	1	1	1
$C_1$	0	0	1	1	0	0	1	1
$C_0$	0	1	0	1	0	1	0	1
		$C_0$	$C_1$	$k_1$	$C_2$	$k_2$	$k_3$	$k_4$

$$\begin{aligned}C_0 \oplus k_1 \oplus k_2 \oplus k_4 \oplus 1 &= 0 &\rightarrow C_0 = k_1 \oplus k_2 \oplus k_4 \oplus 1 \\C_1 \oplus k_1 \oplus k_3 \oplus k_4 \oplus 1 &= 0 &\rightarrow C_1 = k_1 \oplus k_3 \oplus k_4 \oplus 1 \\C_2 \oplus k_2 \oplus k_3 \oplus k_4 \oplus 1 &= 0 &\rightarrow C_2 = k_2 \oplus k_3 \oplus k_4 \oplus 1\end{aligned}$$

# Hammingovi kodovi

- Hammingov kod za ispravljanje jednostrukih pogreški :

- Primjer: zaštita ASCII znaka A (41H)

$$k = n - r = 7 \Rightarrow r = 4$$

1	2	3	4	5	6	7	8	9	10	11
C <sub>0</sub>	C <sub>1</sub>	k <sub>1</sub>	C <sub>2</sub>	k <sub>2</sub>	k <sub>3</sub>	k <sub>4</sub>	C <sub>3</sub>	k <sub>5</sub>	k <sub>6</sub>	k <sub>7</sub>

k <sub>1</sub>	k <sub>2</sub>	k <sub>3</sub>	k <sub>4</sub>	k <sub>5</sub>	k <sub>6</sub>	k <sub>7</sub>
1	0	0	0	0	0	1

$$C_0 = k_1 \oplus k_2 \oplus k_4 \oplus k_5 \oplus k_7 \Rightarrow C_0 = 0$$

$$C_1 = k_1 \oplus k_3 \oplus k_4 \oplus k_6 \oplus k_7 \Rightarrow C_1 = 0$$

$$C_2 = k_2 \oplus k_3 \oplus k_4 \Rightarrow C_2 = 0$$

$$C_3 = k_5 \oplus k_6 \oplus k_7 \Rightarrow C_3 = 1$$

---

$$\Rightarrow \underline{0} \underline{0} 1 \underline{0} 0 0 0 \underline{1} 0 0 1$$

		k <sub>1</sub>		k <sub>2</sub>	k <sub>3</sub>	k <sub>4</sub>		k <sub>5</sub>	k <sub>6</sub>	k <sub>7</sub>
0	1	1	0	0	0	0	1	0	0	1
C <sub>0</sub>	C <sub>1</sub>		C <sub>2</sub>			C <sub>3</sub>				

$$X = C_3 C_2 C_1 C_0 = 1 0 1 0$$

$$Y = C_3' C_2' C_1' C_0' = 1 0 0 0$$

mjesto pogreške:

$$X \oplus Y = 0010_2 = 2_{10}$$

# Hammingovi kodovi

---

- Hammingov kod za ispravljanje jednostrukih pogreški:

- sindrom - uzorak zaštitnih bitova koji ukazuje na mjesto pojave pogreške

Primjer:

sindrom = 2 ~ drugi bit kodne riječi je pogrešan!

		$k_1$	$k_2$	$k_3$	$k_4$	$k_5$	$k_6$	$k_7$
0	1	1	0	0	0	1	0	1
$C_0$	$C_1$		$C_2$			$C_3$		

sindrom = 0

~ nema pogreške

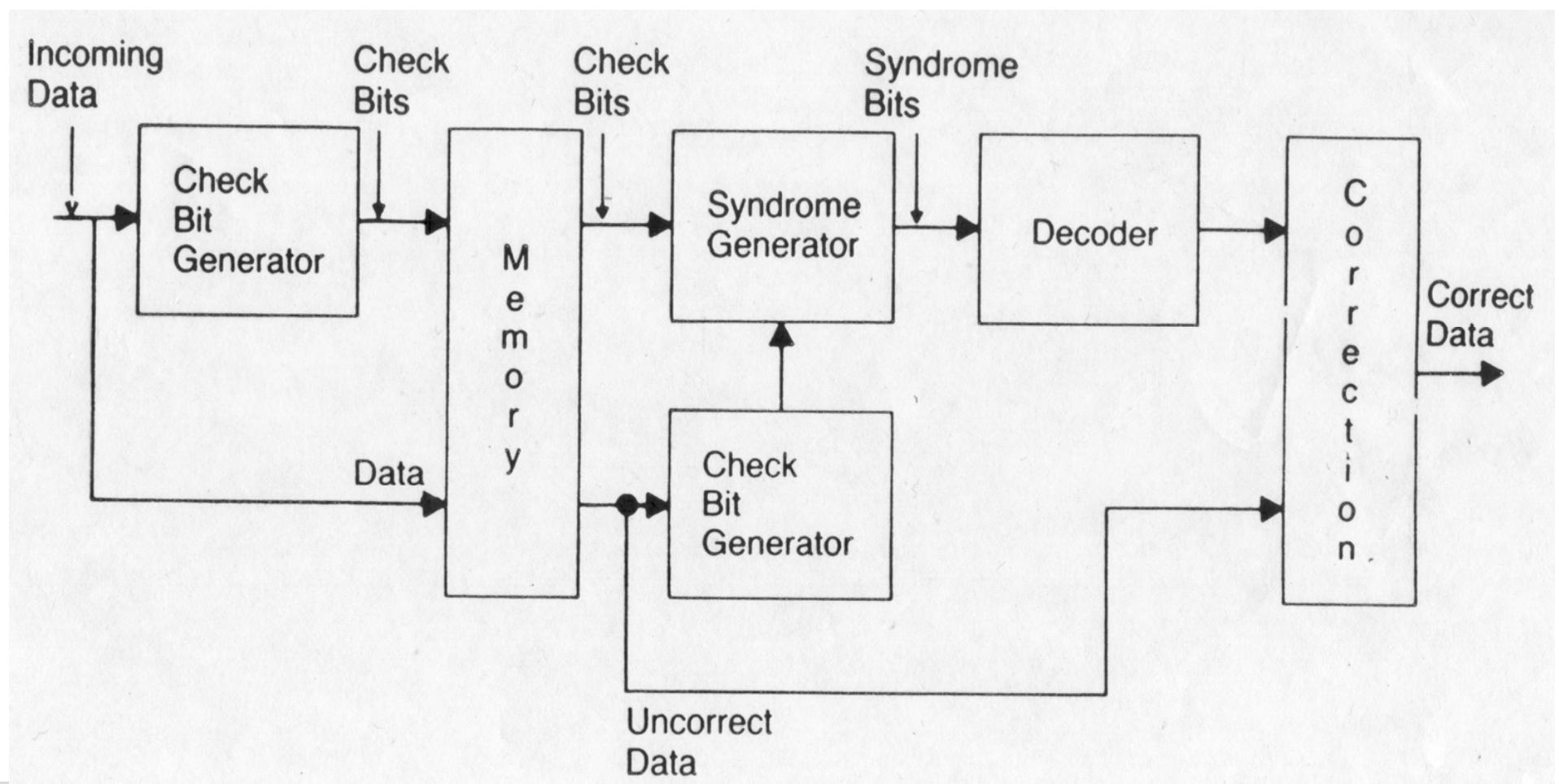
# Hammingovi kodovi

---

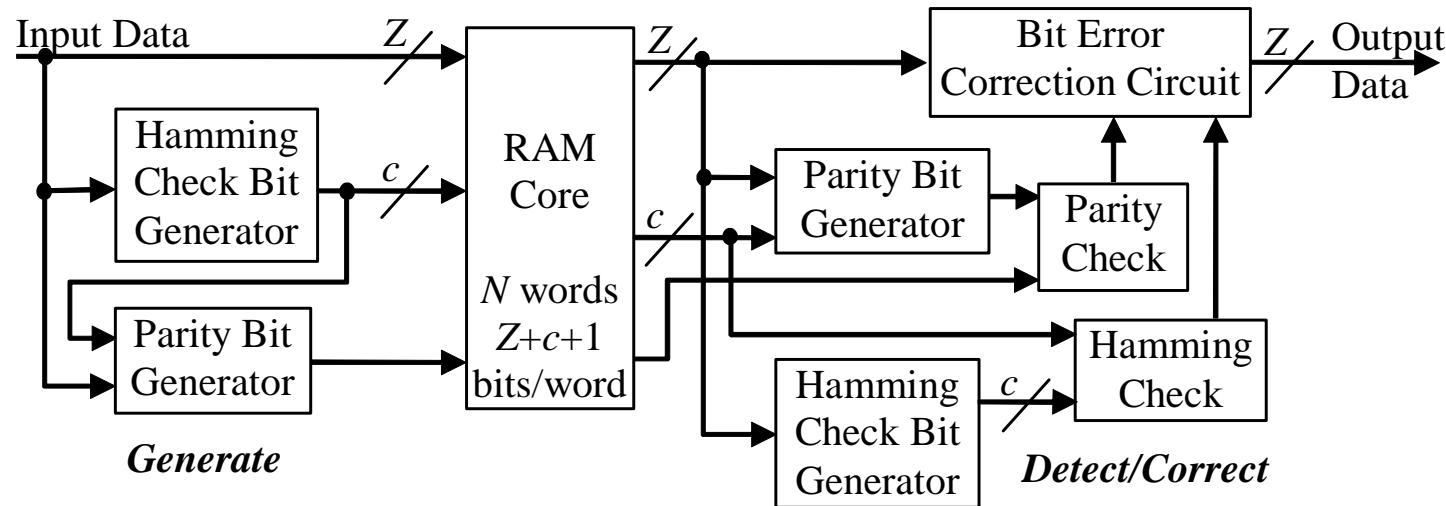
- Hammingov kod za ispravljanje jednostrukih pogreški:
  - distanca  $d = 3$
  - kod za ispravljanje "nezavisnih pogrešaka"
    - ~ rezultat djelovanja "bijelog šuma"
  - efikasan kod, jer je  $R$  mali

Podatak	Hamming (7,4)
0000	0000000
1000	1000110
0100	0100101
1100	1100011
0010	0010011
1010	1010101
0110	0110110
1110	1110000
0001	0001111
1001	1001001
0101	0101010
1101	1101100
0011	0011100
1011	1011010
0111	0111001
1111	1111111

# Primjer



# Uporaba Hammingovog ECC koda

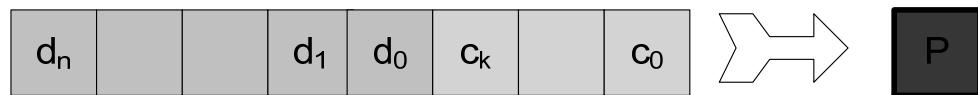


	$Z_1$	$Z_2$	$Z_3$	$Z_4$	$Z_5$	$Z_6$	$Z_7$	$Z_8$	$c_1$	$c_2$	$c_3$	$c_4$
Grupa 1	1	1	0	1	1	0	1	0	1	0	0	0
Grupa 2	1	0	1	1	0	1	1	0	0	1	0	0
Grupa 3	0	1	1	1	0	0	0	1	0	0	1	0
Grupa 4	0	0	0	0	1	1	1	1	0	0	0	1

# Modificirani Hammingov kod

---

- Ispravljanje dvostrukе greške



- Jednostruka –  $S(x)$  – ispravljanje  $P(x)$  neispravan
- dvostruka –  $P(x)$  ispravan i  $S(x)$  neispravan  $\Rightarrow$  NE ispravljati!!
  - Greška, ali se izbjegava pogrešno ispravljanje

# Odabir koda

---

- Separabilnost?
  - Dopušteno vrijeme, memorija, sklopovlje
  - Parallelnost kodiranja/dekodiranja
    - Ciklički kodovi
- Ispravljanje ili samo detekcija
- Broj bitova EDD i ECC?
  - $H_d$

# Aritmetički kodovi

---

## ■ AN kod

- množenje podatka s konstantom
- iskorištava:  $A(N+M) = AN + AM$

## ■ Residualni kodovi

- modulo zbrajanje
- $(M+N) \text{ mod } k = (M \text{ mod } k + N \text{ mod } k) \text{ mod } k$

## ■ Ispitne sume

- otkrivanje pogrešaka

# Samoprovjera

---

## ■ Self-Checking

- oblik sklopovske redundancije
- pretpostavlja kodirane ulaze i izlaze
- u slučaju kvara na izlazu isparavan rezultat ili oznaka pogreške
  - koliki je minimalan broj izlaznih bitova?
  - Razmotriti primjer rada dva identična sklopa i usporedbe rezultata njihovih izlaza

# maskirajuća logika

---

## ■ na razini logičkog sklopa

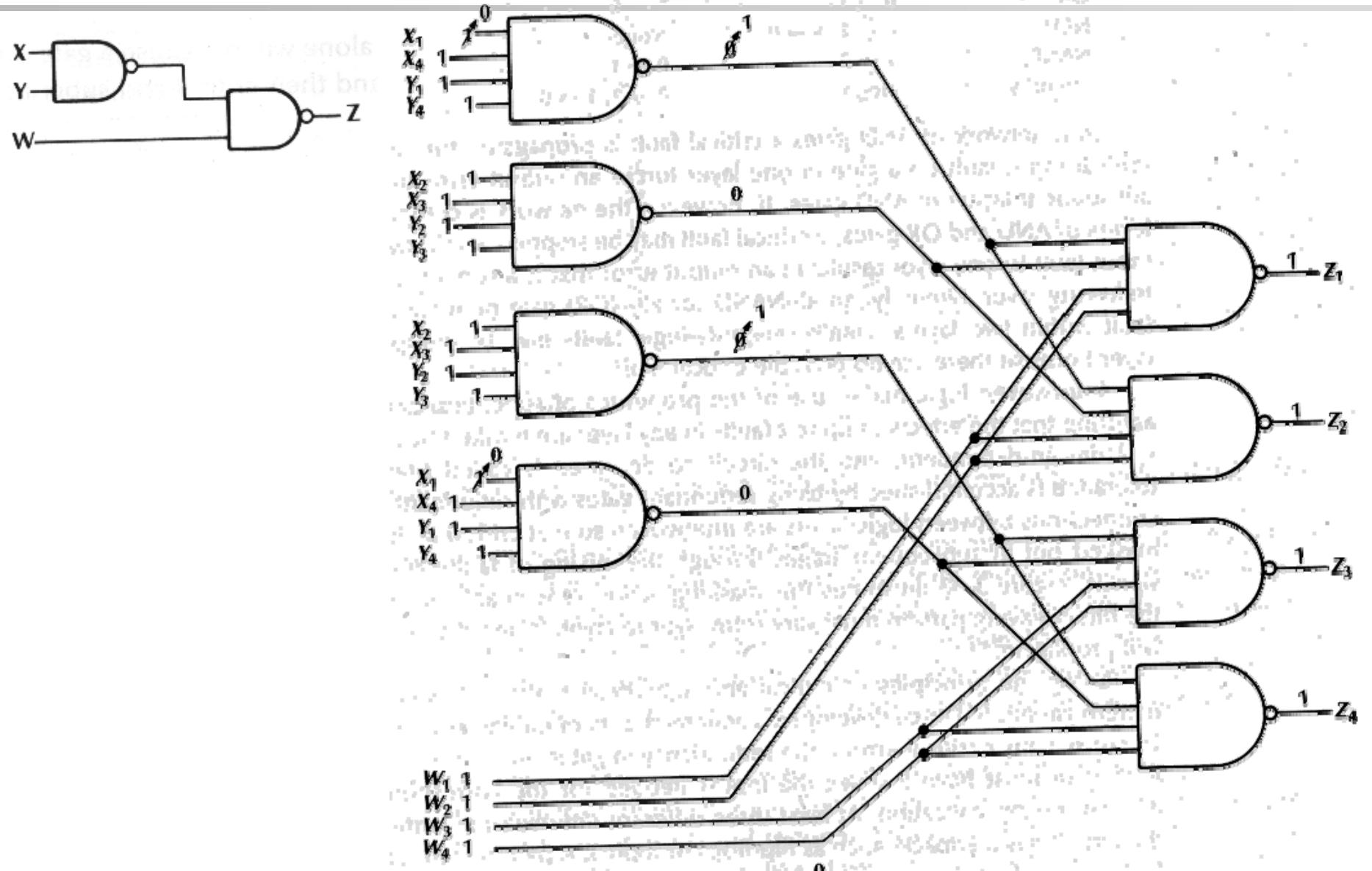
- prijašnje dvije tehnike maskiranja ne uključuju maskiranje kvara na razini logičkih vrata. One štite neki skup sklopova puno kompleksniji od osnovnih sklopova.
- tehnike maskiranja na razini logičkih vrata i stanja u konačnom automatu. Zbog svojih visokih troškova samo par tehnika se koristi u stvarnosti :

## ■ Isprepletena logika

- : svodi se na "stuck-at-", 0 ili 1 mjereno na izlazu, ulazu vrata ili izlazu sklopa ( kratko spajanje linije na 0 ili 1 ).
  - kritični kvar koji uzrokuje grešku na izlazu vrata i nekritični kvar koji ne uzrokuje grešku na izlazu.
- Tolerancija kvara osigurava se korištenjem redundantnih vrata sa redundantnim ulazima.
- Veza između pojedinih slojeva vrata je "isprepletena", tako da kritičan kvar u jednom nivou vrata ostaje maskiran za ostale nivoje miješanjem krivih i dobrih signala, koji ih zamjenjuju.

- **Kritični kvar** -sami po sebi uzrokuju vrijednost na izlazu
- **sub-kritični kvar-sami po sebi ne uzrokuju vrijednost na izlazu**
- **primjer NI sklopa**
- **Isprepletena logika koristi k i sk kvarove tako da osigura maskiranje kvara u nivoima**
- **1965 Pierce**
- **$R=(t+1)^2=B^2$**

	<b>kritični</b>	<b>subkritični</b>
I	<b>s-a-0</b>	<b>s-a-1</b>
ILI	<b>s-a-1</b>	<b>s-a-0</b>
NE	<b>s-a-0</b> <b>s-a-1</b>	
NI	<b>s-a-0</b>	<b>s-a-1</b>



### Single-Fault Tolerant

Tolerant

$$t = 1, B = 2, R = 4$$

$$g_1 = (1,2)(3,4)$$

$$g_2 = (1,4)(2,3)$$

$$g_3 = (1,3)(2,4)$$

Double-Fault Tolerant

$$t = 2, B = 3, R = 9$$

$$g_1 = (1,2,3)(4,5,6)(7,8,9)$$

$$g_2 = (1,4,7)(2,5,8)(3,6,9)$$

$$g_3 = (1,6,8)(5,7,3)(9,2,4)$$

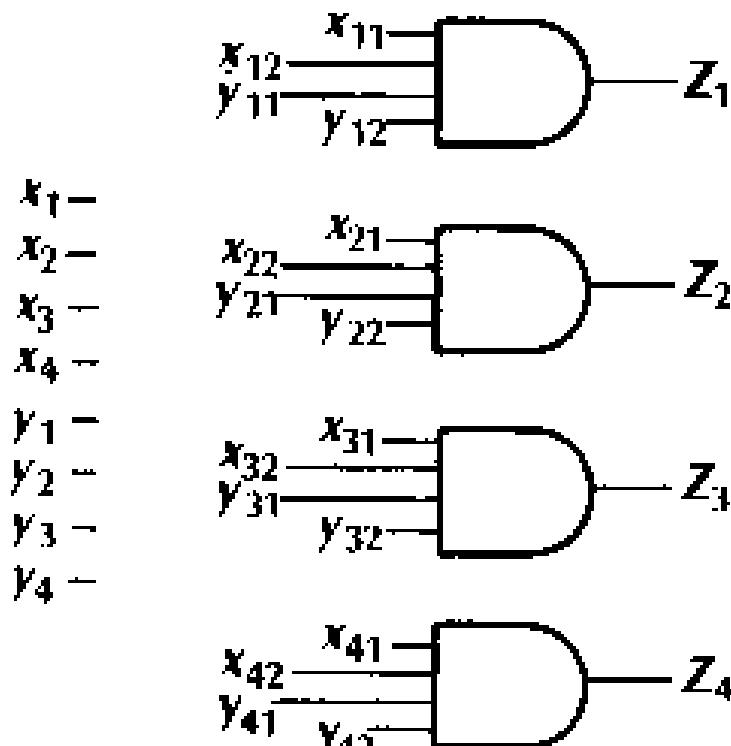
Triple-Fault Tolerant

$$t = 3, B = 4, R = 16$$

$$g_1 = (1,2,3,4)(5,6,7,8)(9,10,11,12)(13,14,15,16)$$

$$g_2 = (1,5,9,13)(2,6,10,14)(3,7,11,15)(4,8,12,16)$$

$$g_3 = (1,6,12,15)(2,5,11,16)(3,8,9,14)(4,7,10,12)$$

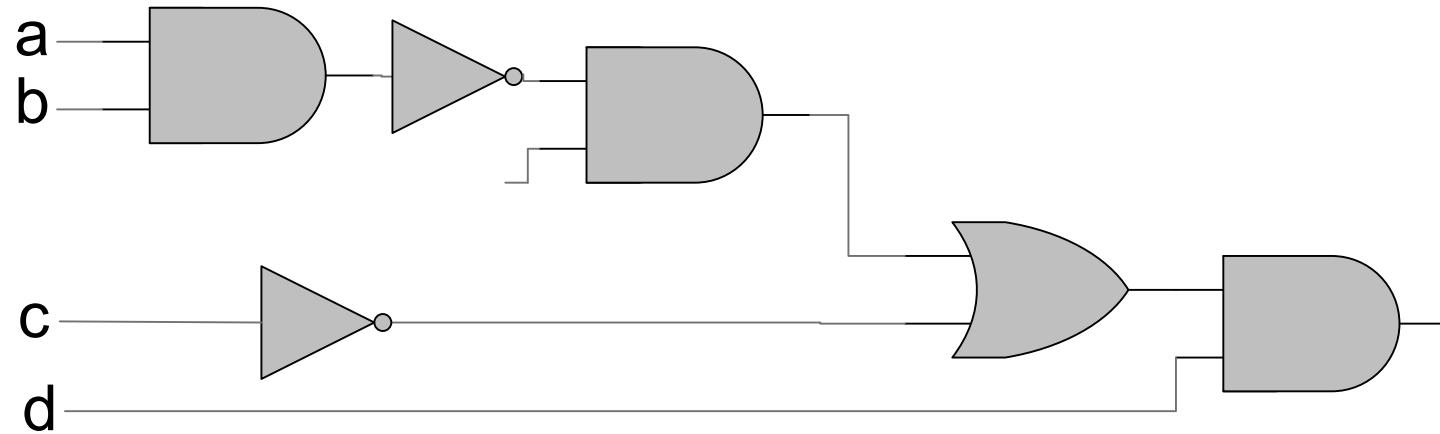
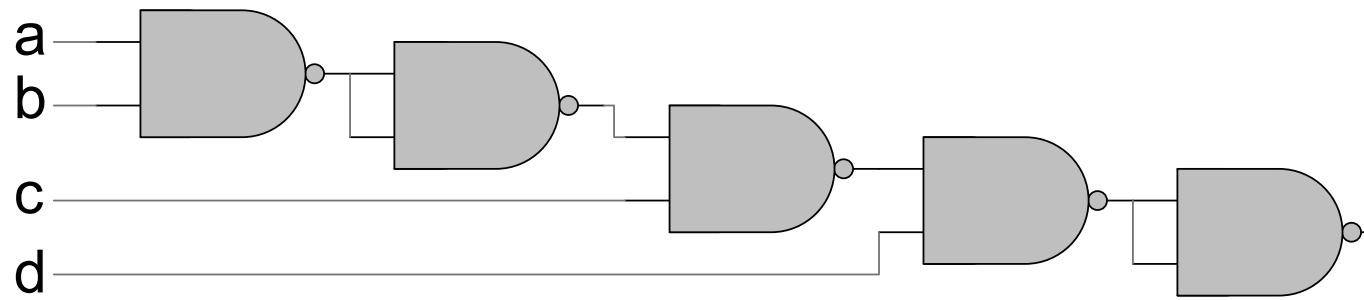


$\diagdown g_i$	$g_1$	$g_2$	$g_3$
11	1	1	1
12	2	4	3
21	1	2	2
22	2	3	4
31	3	2	1
32	4	3	3
41	3	1	2
42	4	4	4

# Primjer

---

$$f = d(\overline{ab} + \overline{c}) = \overline{(ab)}\overline{cd} = (\overline{(ab)} + \overline{c})d$$



# Pitanja...

# Pouzdanost računalnih sustava

---

## Predavanje 9: Pouzdanost programske podrške

Prof.dr.sc. Vlado Sruk



Sveučilište u Zagrebu  
**Fakultet elektrotehnike i računarstva**  
Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave



# Sadržaj

---

- Podsjetnik
- Modeli pogrešaka programa
- Redundancija programa
- N-verzijsko programiranje
- Blokovi oporavka

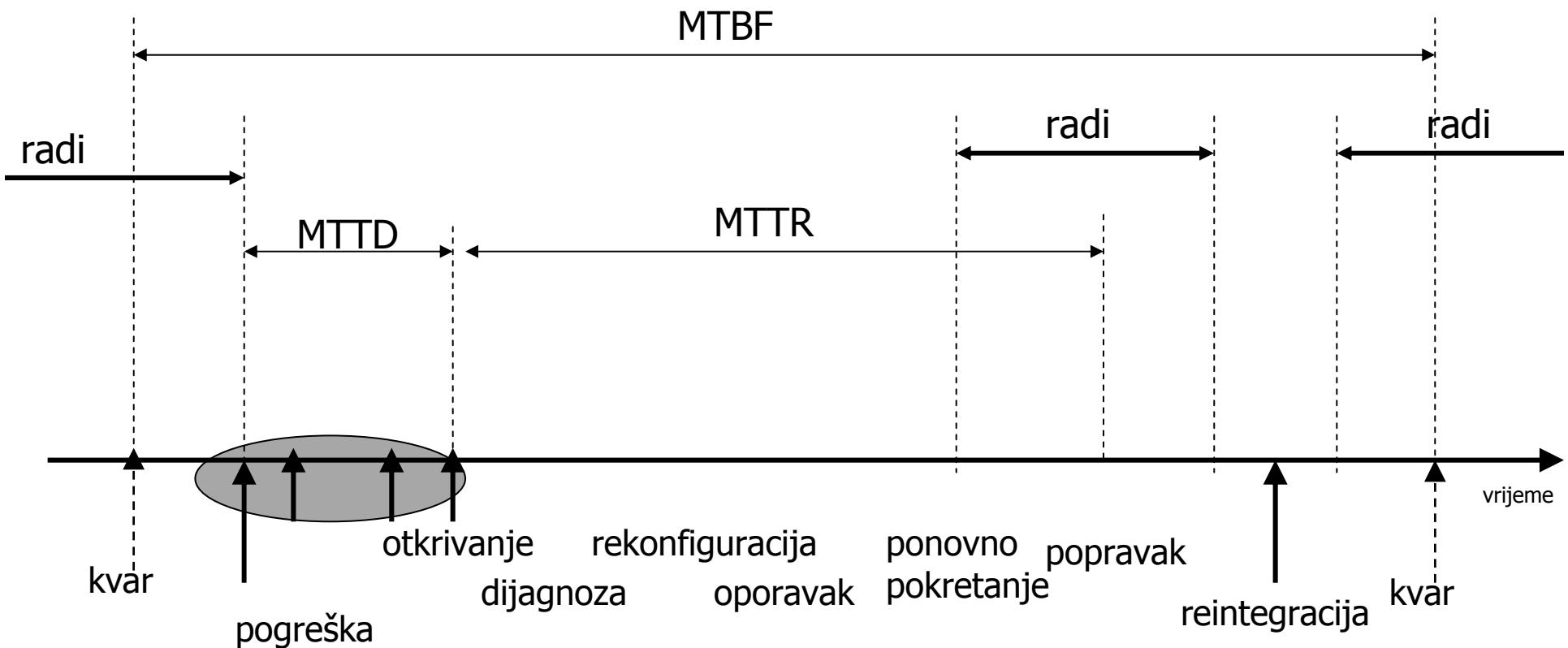
# Osnovne karakteristike programske podrške

---

- u usporedni s pouzdanošću sklopolja
  - programska pouzdanost nije funkcije proizvodnje
  - program ne degradira tijekom vremena
  - promjene okoline ne utječu na programsку podršku
- Sva zatajenja su rezultat pogrešaka oblikovanja ili pogrešaka korisnika
- Ispravljanje zatajenja moguće isključivo preoblikovanjem
  - MTTR teško odrediti – utjecaj na raspoloživost?
  - Velik dio funkcionalnosti programa može ipak biti upotrebljiv

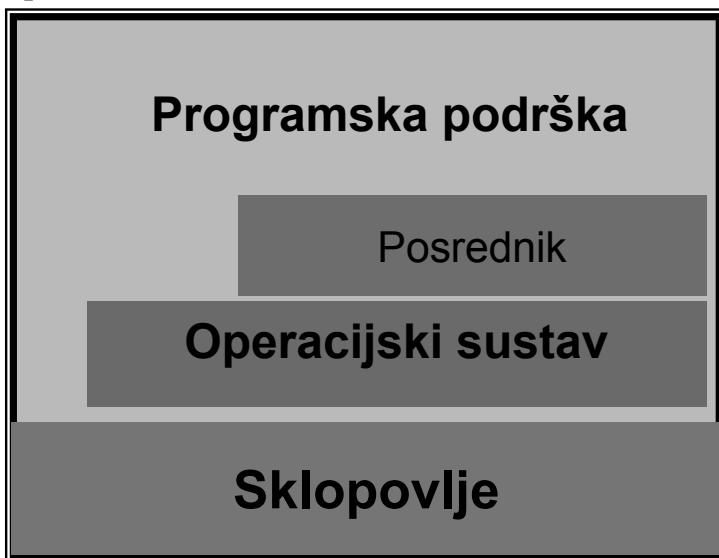
# Ciklus kvara

## ■ Više scenarija



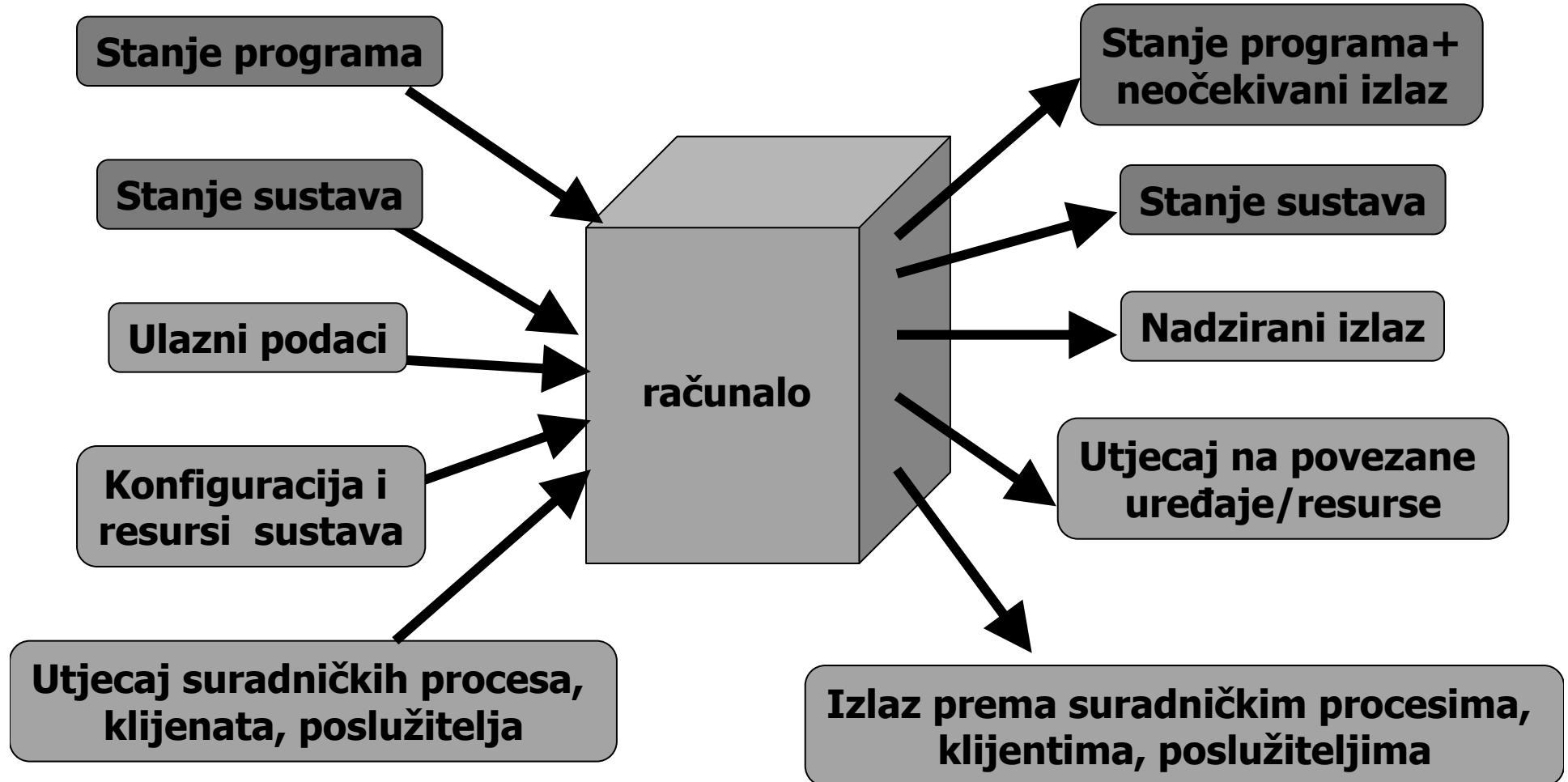
# Računalni sustavi

- Računalni sustavi kombiniraju programsku podršku i sklopolje
- Kakav je odnos programske i sklopojske pouzdanosti?



- Sklopolje:
  - ECC, N od M, pričuvna , glasanje, budilice, ...
- OS:
  - Upravljanje memorijom, otkrivanje zatajenja, podrška programskoj neosjetljivosti na pogreške
- Programska podrška:
  - Blokovi oporavka, N verzijsko programiranje, uvišestručavanje, glasanje, ...

# Uzročnici zatajenja programa



Osnovna klasifikacija kvarova programske podrške: oblikovanje, nadogradnje, promjena zahtjeva

# Zatajenja programa

---

- Error - Fault – Failure
  - Što je Bug/Neispravnost?
- Ugrađena dijagnostika
  - Umetanje provjera u kod programa
  - Negativi efekti (struktura podataka, resursi, vrijeme,...)
- Efekt “nenamjernog sljepila”
  - Ljudi: Ne vide ono što nije u centru pažnje
  - Programi: Uvijek ne vide ono što im nije naglašeno da paze
    - Precizniji i manje prilagodljivi
- Neponovljiva zatajenja
  - Razmatranje krivih uvjeta
    - nemoguće analizirati sve
- Testovi praktično ne mogu pokriti sve mogućnosti grešaka ☹

# Uporaba informacija u pouzdanost PP

---

- Specifikacije
  - Formalne, neformalne
  - za odabir, generiranje, provjeru
- Informacije oblikovanja
  - za odabir, generiranje, provjeru
- Programski kod
  - za odabir, generiranje, provjeru
- Uporaba
  - Povijest, model
- Iskustvo

# Klasifikacija programskih kvarova

---

- **Memory/Resource leak**
  - neoslobađane resursa
- **Dangling pointers**
  - uporaba pokazivača nakon oslobođanja resursa
- **Buffer overrun**
  - prepisivanje podataka
- **Race condition**
  - natjecanje za pristup resursima
- **Deadlock**
  - potpuni zastoj uzrokovan natjecanjem za više resursa
- **Timing expectations**
  - nezadovoljavanje vremenskih ograničenja
- **Transient errors**
  - nije potrebna intervencija u programu

# Klasifikacija programskih kvarova II

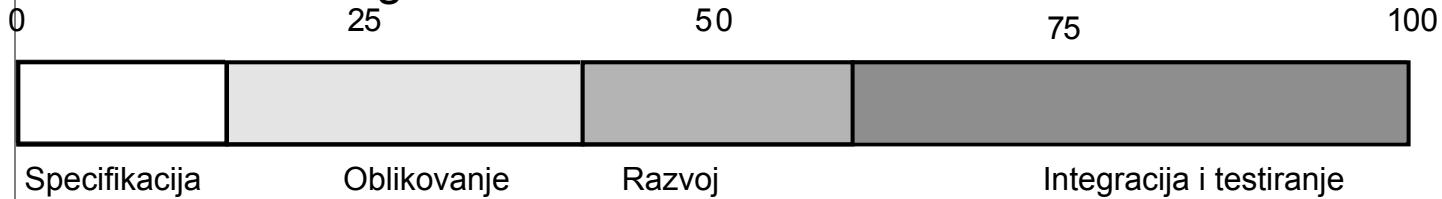
---

## Bohrbugs

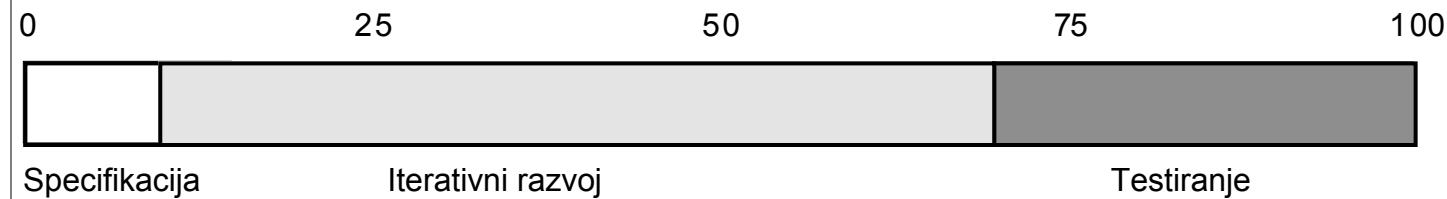
- jednostavno ponavljanje
  - uklanjanje ispravljanjem programa
- **Heisenbugs** (ime prema Heisenbergovom principu neodređenosti)
  - nestaju ili mijenjaju svojstva pri pokušaju proučavanja (teška reprodukcija)
    - uklanjanje ispravljanjem programa
- **Mandelbugs** (ime prema tvorcu fraktala Mandelbrotu)
  - nedeterminističko ponašanje uzrokuje naizgled kaotično ponašanje
- **Schrödingbugs** (ime prema fizičaru Schrödingeru)
  - složeno, teško razumljivo ponašanje za programera
    - nakon proučavanja koda ili neuobičajenim izvođenjem dolazi se do zaključka da program uopće nije trebao raditi dobro
- **Aging-bugs**
  - pojavljuju se nakon dugotrajnog izvođenja

# Model procesa programskog inženjerstva

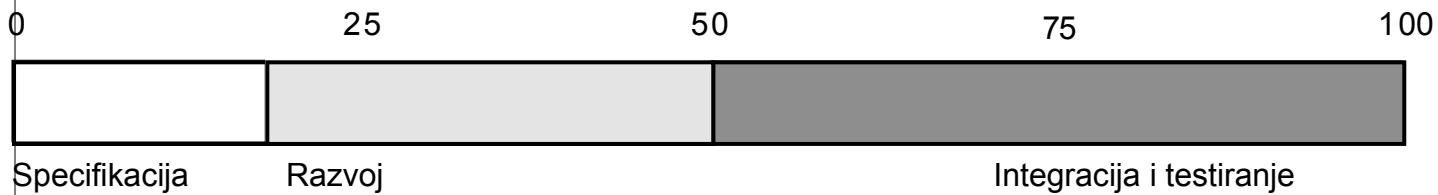
## VODOPADNI eng. Waterfall model



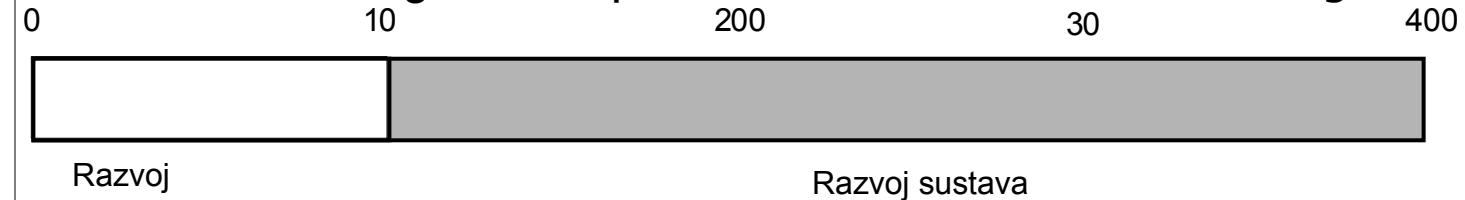
## ITERATIVNI engl. Iterative development



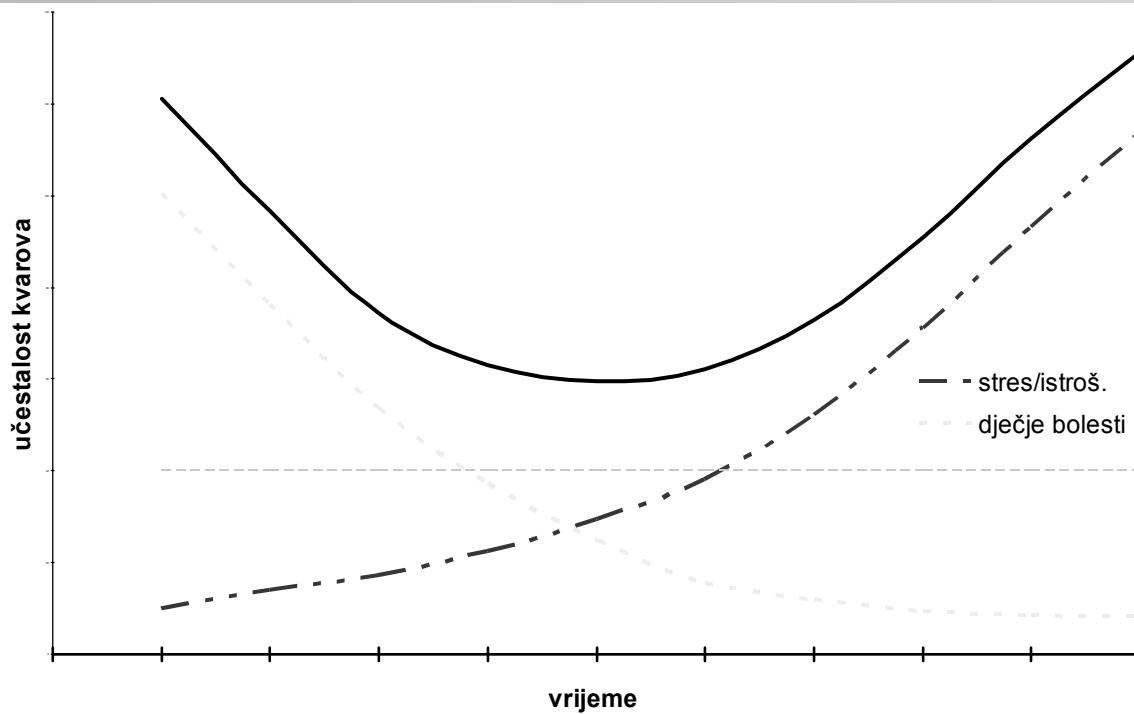
## KOMPONENTNI engl. Component-based software engineering



## DUGOVJEČNI engl. Development and evolution costs for long-lifetime syst



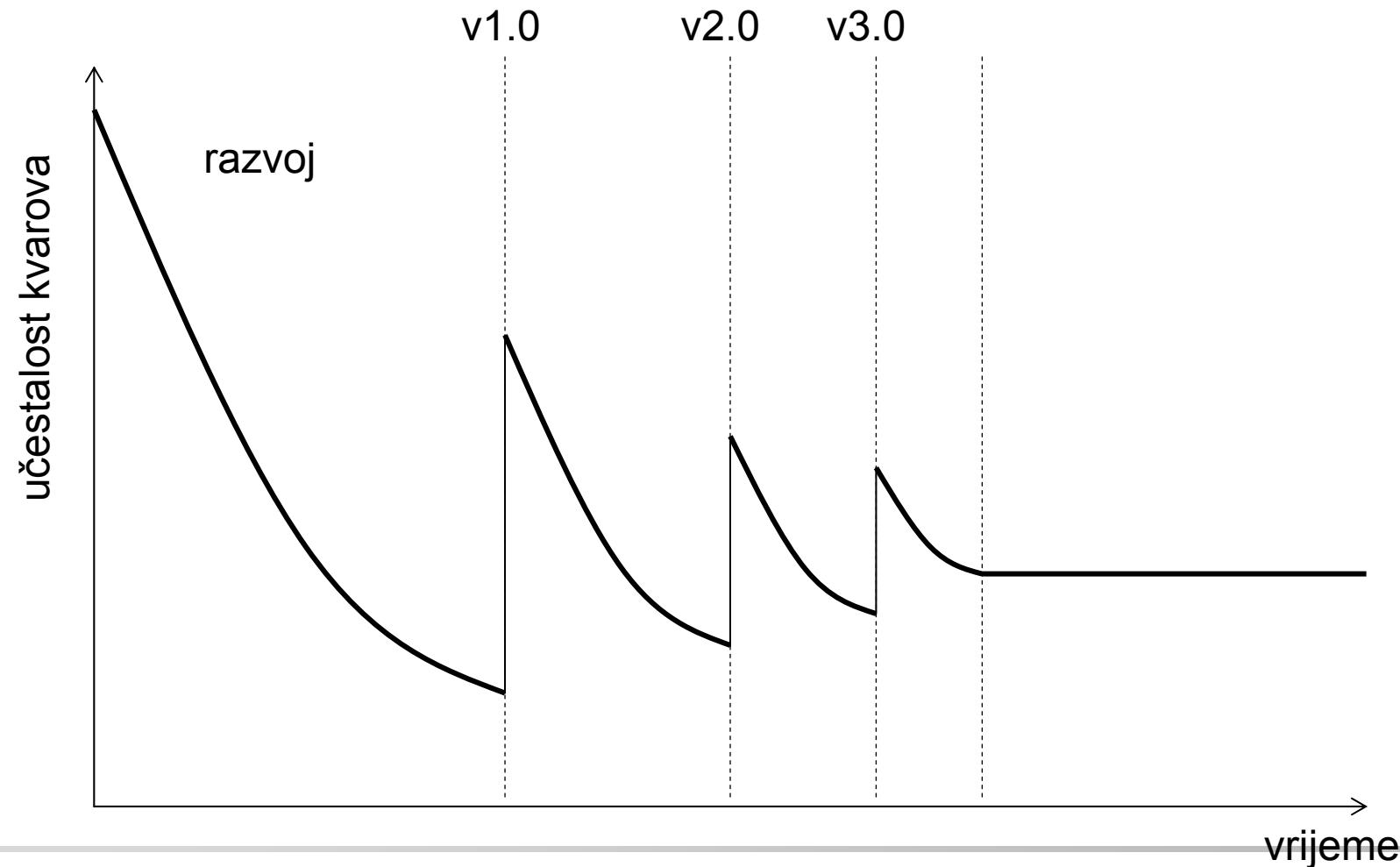
# faktori koju utječu na kvarove



- Kod programske podrške pri ispravljanju ili nadogradnji postoji mogućnost unosa novih kvarova!!!
- Kako se to odražava na pojedine faze?

# karakteristična krivulja prog. pouzdanosti

---



# Pouzdanost programske podrške

---

- Pouzdanost programske podrške je vjerojatnost ispravnog rada aplikacije u zadanoj radnoj okolini i vremenskom intervalu.
- Radna okolina:
  - sklopovlje: računalo i konfiguracija
  - programi: OS, biblioteke, ...
  - uporaba: operacijski profil
- Samo jedna od mjera kvalitete programske podrške!!!
  - performanse, održavanje, prenosivost, ...

# Definicija kvalitete prog. podrške

---

- sukladnost s izričito postavljenim zahtjevima funkcionalnosti i performansi, s primjenom dokumentiranih razvojnih standarada i implicitnim karakteristikama očekivanima profesionalnom razvoju programa
- Troškovi
- Prevencija
  - planiranje kvalitete, testiranje, školovanje
- Zatajenja - interna
  - popravci, analiza zatajenja
- Zatajenja - vanjska
  - žalbe, povrati, reklamacije, podrška, garancije

# Standardi

---

- IEEE Std. 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology
- IEEE Std 830-1998 – IEEE Recommended Practice for Software Requirements Specifications
  - definira faktore kvalitete specifikacije zahtjeva i kriterije njihove procjene
    - Kompletnost
    - Jednoznačnost
    - Korektnost
    - Konzistentnost
    - Provjerljivost
    - Promjenjivost
    - Dokazljivost Poredak važnosti
    - Poredak stabilnosti

# Definicije pouzdanosti

---

## ■ IEEE 610.12-1990

- *The ability of a system or component to perform its required functions under stated conditions for a specified period of time."*

## ■ IEEE 982.1-1988

- *Software Reliability Management is a process of optimizing the reliability of software through a program that emphasizes software error prevention, fault detection and removal, and the use of measurements to maximize reliability in light of project constraints such as resources, schedule and performance.*

## ■ Osnovne aktivnosti u pouzdanosti programske podrške

- Prevencija pogrešaka
- Otkrivanje i uklanjanje kvarova
- Mjere za poboljšanje pouzdanosti

# Primjer 3: Myrinet mrežni preklopnik

---

- brza računalna mreža za povezivanje računalnih grozdova
  - Message dropped
    - izgubljena poruka
  - Data corrupted
    - poslana poruka s neispravnim sadržajem
  - Restart
    - samostalni restart
  - Interface hung
    - pogreška sučelja
  - Computer crash
    - zatajenje lokalnog ili udaljenog čvora

# Izbjegavanje programskih kvarova

---

- engl. Fault Avoidance
- programski kvar – kvar oblikovanja
- prevencija i uklanjanje kvara
- cilj: izgradnja programske podrške bez pogrešaka (engl. zero-defect software)
- problem specifikacija programa
  - Formalne specifikacije
  - Inkrementalni razvoj
  - Statička verifikacija
  - Statističko testiranje
  - Strukturno programiranje
  - Modularno programiranje
  - Objektno orijentirano programiranje
  - Oblikovanje od vrha prema dnu
- izbjegavanje rizičnog programiranja
  - Defensive Programming

# Koraci

---

1. svi zahtjevi specificirani i analizirani formalnim metodama
  2. specifikacija pročišćena prije razvoja komponenti
  3. definiran protokol razrješavanja problema
  4. formalizirani postupci validacije i verifikacije
  5. iscrpno testiranje specifikacije, oblikovanja i koda
- u praksi razvoj programske podrške je podložan greškama te su i tehnike izbjegavanja i uklanjanja pogrešaka nesavršene ...

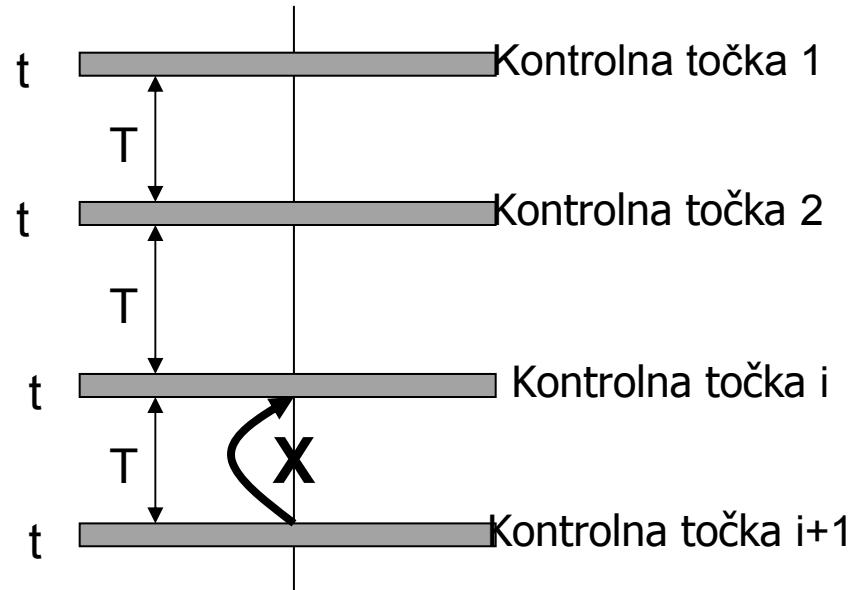
# Programska redundancija

---

- pomalo zanemarena radi visoke cijene
- Osnovne tehnike:
  - Konstrukcija algoritama
    - minimiziranje kvarova primjenom odgovarajućih algoritama
    - najraširenije u raspodijeljenim sustavima npr. transakcije
  - Različitost programiranja (engl. diverse programming)
    - neovisan razvoj programa koji se istodobno izvršavaju
    - engl. N-version programming
      1. specifikacija
      2. različite grupe razviju program (različiti alg., okolina, ...; isti test prih.)
      3. izvršavanje (u N-version execution environment) s donošenjem odluke
  - Dinamička programska redundancija
    - oporavak unaprijed (engl. Forward Error Recovery)
      - pesimistično/optimistično
    - oporavak unazad (engl. Backward Error Recovery)
      - ponovno izvođenje, kontrolne točke, oporavak

# Kontrolne točke

- provjera u pravilnim razmacima
- oporavak u slučaju kvara
- nedostatak smanjenje performansi



- Kakao detektirati kvar?

# Detekcija kvara

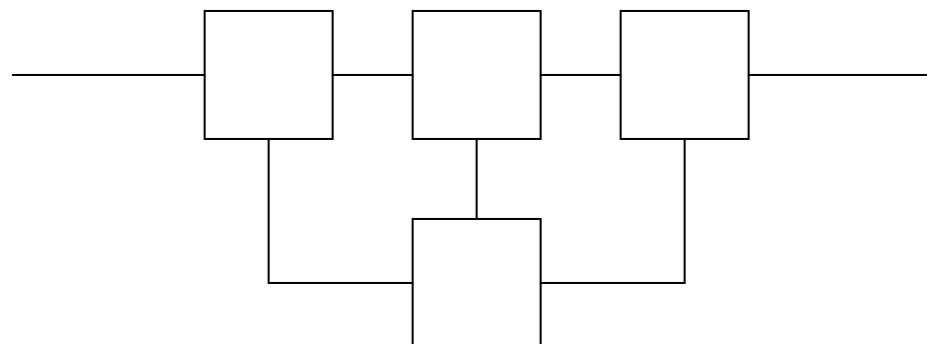
---

- Izgradnja dijagnostičkog programa
  - unosi dodatno kašnjenje
  - problem pokrivanja pogrešaka
- Udvostručeni sustav
  - usporedba

# Blokovi opravka

---

- engl. Recovery Blocks
- redundantne komponente su povezane serijski
- Testovi prihvatljivosti procjenjuju rezultat
- Pri zatajenju jednog bloka vraća se početno stanje i nastavlja sa slijedećim blokom
- Ponavlja se do uspjeha ili iscrpljivanja blokova



# Programska neosjetljivost na kvarove

---

- **engl. Software Fault Tolerance Methods**
  - izolacija programskih pogrešaka, sprečavanje zatajenja
- obrada iznimaka (engl. exception handling)
- budilica (engl. watchdog timers)
- provjere tijekom izvođenja (engl. run-time check/assertions)
- provjere prihvatljivosti (engl. acceptability checks)
- provjera (engl. reasonableness checks)
- raznovrsnost oblikovanja (engl. design diversity)
- raznovrsnost podataka (engl. data diversity)

# Pitanja...

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# Pouzdanost računalnih sustava

---

## Predavanje 10: Pouzdanost programske podrške

Prof.dr.sc. Vlado Sruk



**Sveučilište u Zagrebu**  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave*



# Sadržaj

---

- Podsjetnik
- Modeli pogrešaka programa
- Redundancija programa
- N-verzijsko programiranje
- Blokovi oporavka

# Operacijski profil

---

- engl. Operational Profile
- kvantitativno opisuje praktičnu uporabu programa
  - specifikacije klase ulaznih podataka
  - vjerojatnost njihove pojave
- određivanje operacijskog profila
  - statistički podaci o radu
  - iz specifikacije zahtjeva
    - teško razviti pouzdanu metodologiju
- ulazni podaci
  - sve što utječe na rad

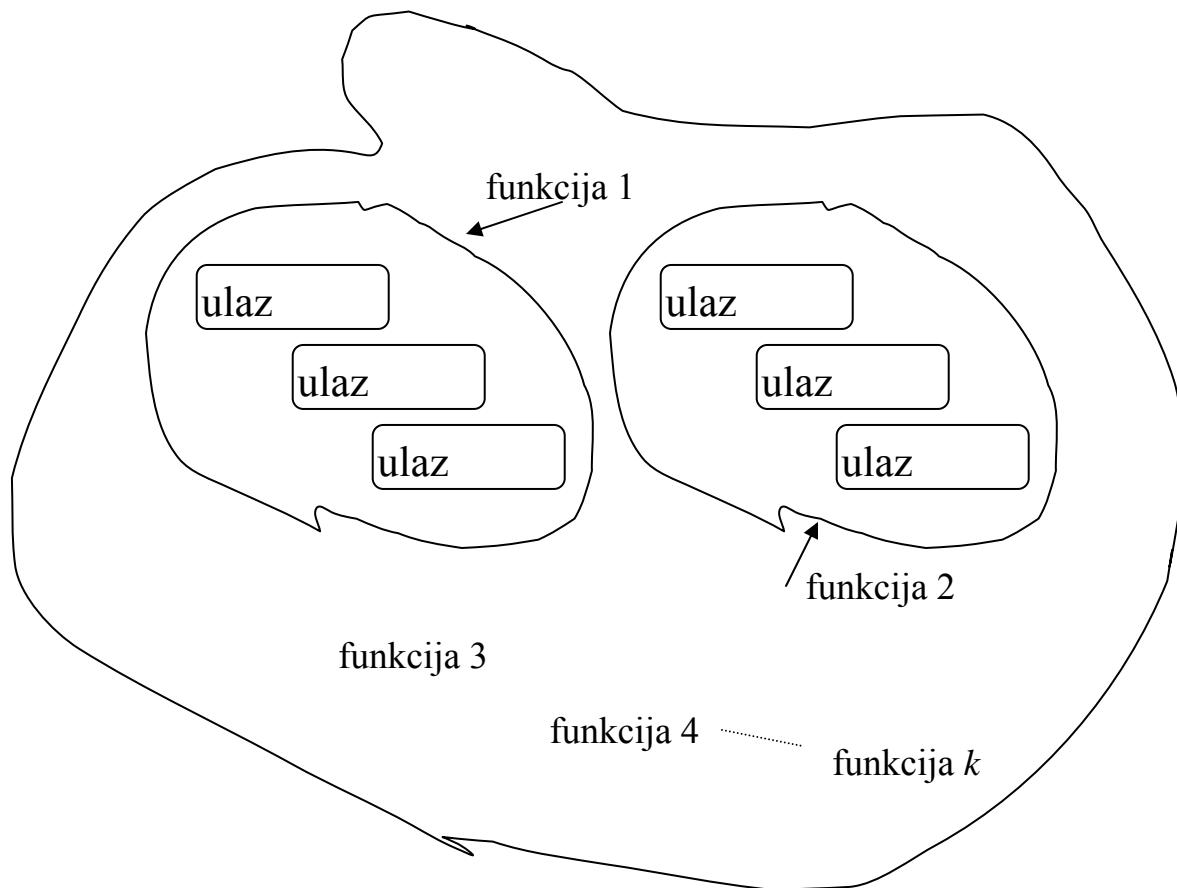
# Operacijski profil

---

- Tip izvođenja: izvođenje programa s identičnim ulaznim podacima (engl. run type)
- Funkcija: grupiranje različitih tipova izvođenja
  - u fazi analize zahtjeva use-case
- Operacija: skup tipova izvođenja jednog programa

# Primjer

---



# profil

---

## ■ Funkcijski

Funkcija	Vjerojatnost pojave
F1	0.6
F2	0.35
F3	0.05

## ■ Operacijski

Funkcija	operacija	vjerojatnost
F1	011	0.4
	012	0.1
	013	0.1
F2	021	0.05
	022	0.15
F3	031	0.15
	033	0.05

# Utjecaj načina rada

---

Način rada	funkcija	operacija	vjerojatnost
korisnički	F1	011	0.4
		012	0.1
		013	0.1
	F2	021	0.05
		022	0.15
	F3	031	0.15
		033	0.05
administacija	AF1	A011	0.4
		A012	0.1
	AF2	A021	0.5

# proces procjene pouzdanosti

---



# Procjena rizika

---

- obuhvaća
  - vjerojatnost pojave zatajenja
  - posljedice zatajenja
- važno za
  - identifikaciju kritičnih modula i procjenu potrebnog testiranja

# Modeli pouzdanosti programske podrške

---

- svojstva postojećih modela
  - veliki broj
  - grube pretpostavke
  - uska specijalizacija modela
  - zanemaruju postupak razvoja
- klasifikacija
  - modeli predviđanja -
  - modeli određivanja

# Modeli pouzdanosti programske podrške

---

- deterministički
  - Halstead
  - McCabe – ciklomatska složenost
- vjerojatnosni
  - model ubacivanja – Error Seeding
  - učestalost pogrešaka – Failure rate
  - Curve fitting
  - rast pouzdanosti – Reliability growth

# Halsteadov model

---

- parametri
- $n_1$  = broj različitih operatora
- $n_2$  = broj različitih operanada
- $N_1$  = ukupan broj operatora
- $N_2$  = ukupan broj operanada
- $N$  = duljina programa
- $V$  = volumen programa  $V=N \log(n_1+n_2)$
- $E$  = broj pogrešaka
- $I$  = broj asemblerskih instrukcija

$$E=V/3000; \quad E=A/3000$$

$$A = \left( \frac{V}{\frac{2n_2}{n_1 N_2}} \right)^{\frac{2}{3}}$$

# Modeli ubacivanja pogrešaka

---

- dvije grupe grešaka
  - izvorne
  - ubaćene
- broj nepoznatih grešaka procjenjuje se na temelju broja ubaćenih grešaka i podatak ispravljanja grešaka
- tri grupe
  - Millsov model, 1970
  - Cai's model, 1998
  - Hypergeometric distribution model, Tohma 1991

# Millsov model

---

- $N$  = broj pogrešaka
- $n_1$  = broj ubačenih pogrešaka
- $r$  = ukupan broj uklonjenih pogrešaka
- $k$  = broj uklonjenih ubačenih pogrešaka

$$P(k; N, n_1, r) = \frac{\binom{n_1}{k} \binom{N}{r-k}}{\binom{N+n_1}{r}}, \quad k = 1, 2, \dots, r$$

- $N=N_0+1$        $N_0 = \frac{n_1(r-k)}{k} - 1$

# učestalost pogrešaka

---

- zasnovane na praćenju učestalosti pogrešaka
- diskretne funkcije s diskontinuitetima u vremenu kvara
- najpoznatije
  - Jelinski and Moranda, 1972
  - Schick and Wolverton, 1978
  - Jelinski-Moranda geometrijska, Moranda 1979
  - Moranda geometrijski Poisson, Littlewood 1979
  - Negative-binomial Poisson
  - Schick-Wolverton, Sukert 1977
  - Goel and Okumoto, Goel 1979

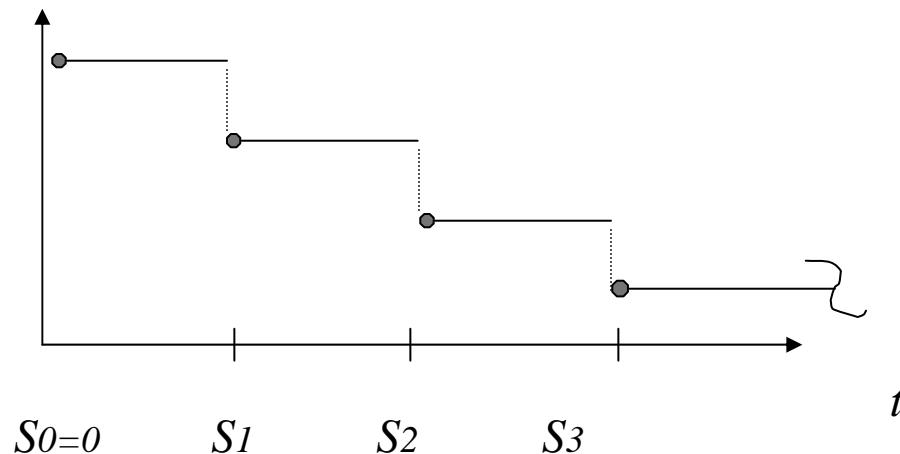
# Jelinski and Moranda Model [1972]

---

## ■ Prepostavka:

- program ima N nepoznatih pogrešaka.
- kod svakog zatajenja otklanja se kvar
- Ispravljanje je savršeno ☺
- odnos kvarova i zatajenja je konstantan

$$r_{Ti}(t - S_{i-1}) = c(N - i + 1) \quad t \geq S_{i-1}$$



# Musa-Okumoto Model

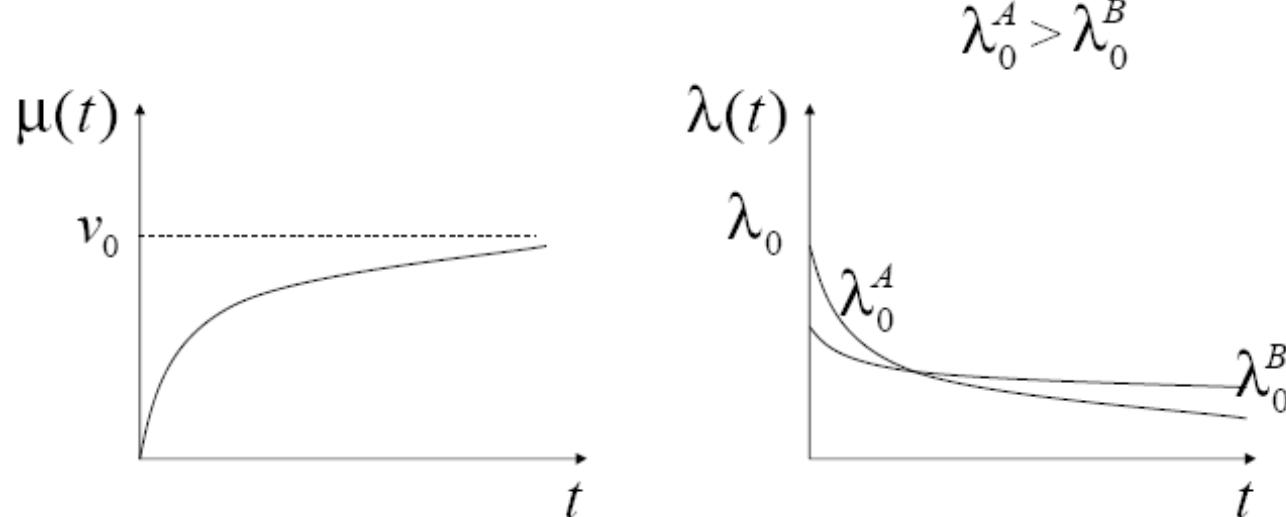
---

## ■ Pretpostavke:

- $\tau$  - vrijeme izvođenja
- $\tau'$  - vrijeme izvođenja od trenutnog vremena:
- početna učestalost kvarova:  $\lambda_0 = f K \omega_0$
- $\mu$  - prosječan broj zatajenja
- ukupan broj zatajenja:  $v_0 = \omega_0 / B$
- $B$  - faktor uklanjanja kvarova
- $\phi$  - učestalost po grešci
- $\lambda_0 / v_0 = B \phi$
- broj inherentnih pogrešaka:  $\omega_0 = \omega / \Delta I$
- broj inherentnih pogrešaka po izvornoj instrukciji :  $\omega$
- .....

# Musa-Okumoto: osnovni model

$$\lambda(\mu) = \lambda_0 \left[ 1 - \frac{\mu}{\nu_0} \right]^{-\frac{\lambda_0}{\nu_0}\tau}$$
$$R(\tau' | \tau) = e^{\{-[\nu_0 e^{-\frac{\lambda_0}{\nu_0}\tau}] [1 - e^{-\frac{\lambda_0}{\nu_0}\tau'}]\}}$$
$$\lambda(\tau) = \lambda_0 e^{-\frac{\lambda_0}{\nu_0}\tau}$$



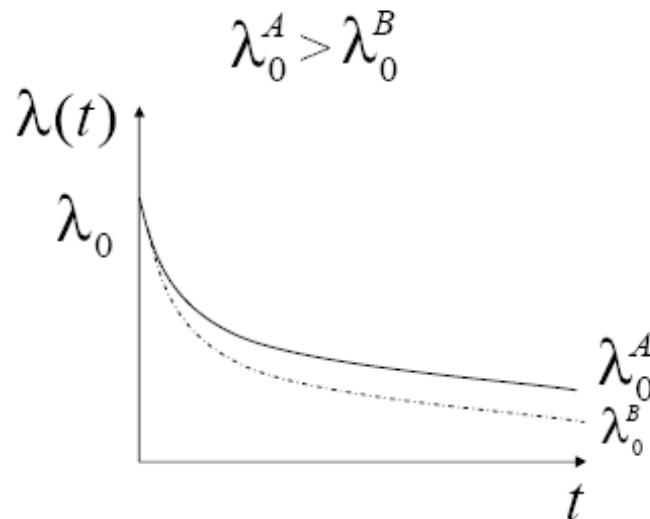
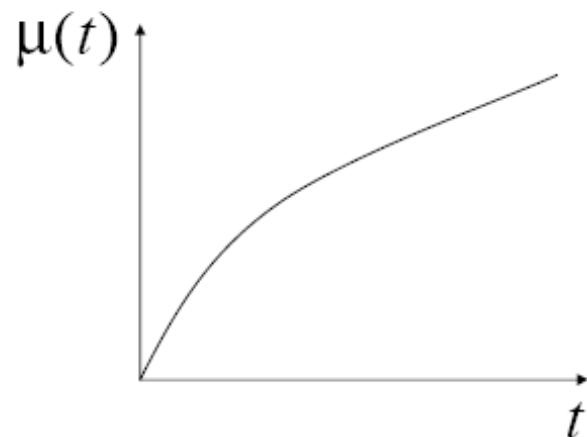
## Musa-Okumoto: Logaritamski Pissonov model

- parametra pad intenziteta kvarova nakon ispravljanja

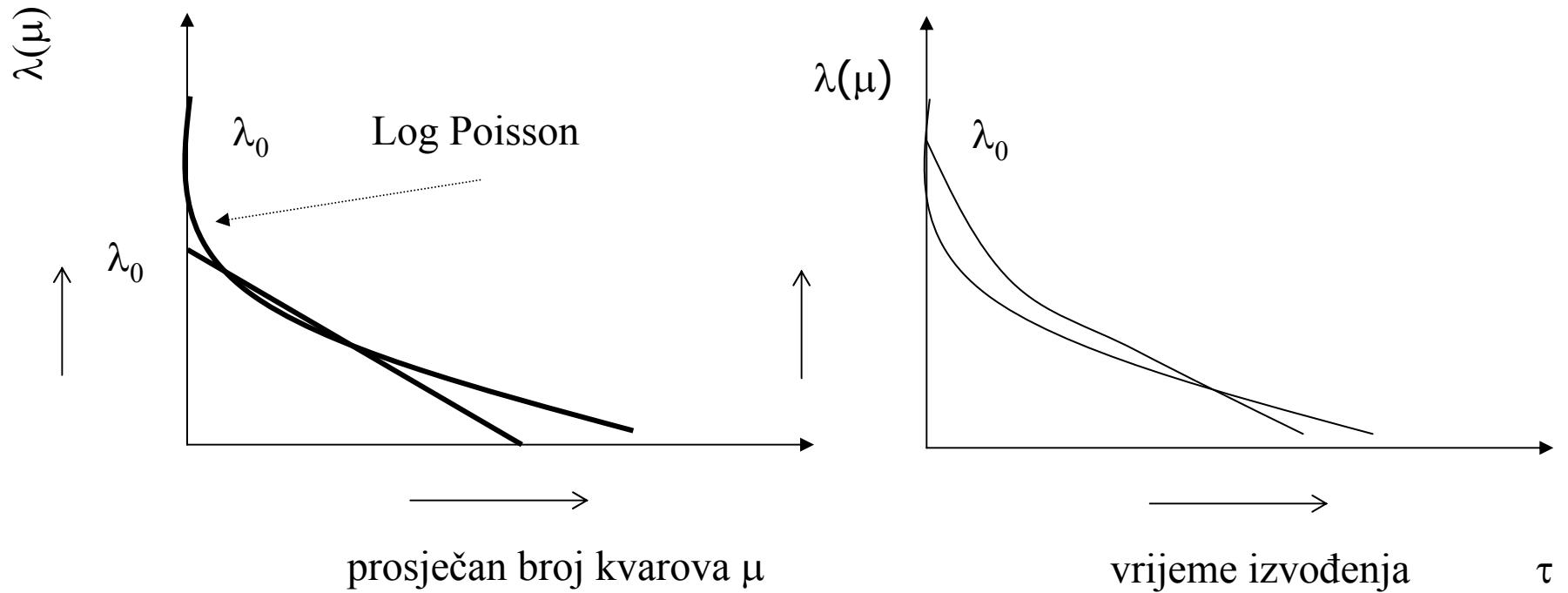
$$\lambda(\mu) = \lambda_0 e^{-\theta\mu}$$

$$\lambda(\tau) = \frac{\lambda_0}{\lambda_0 \theta \tau + 1}$$

$$R(\tau' | \tau) = \left[ \frac{\lambda_0 \theta \tau + 1}{\lambda_0 \theta (\tau' + \tau) + 1} \right]^{1/\theta}$$



# Usporedba osnovnog i log Poissonovog modela



# Odabir modela

---

- uniformni operacijski profil
  - osnovni model
- neuniformni operacijski profil
  - Logaritamski Poissonov Model

# Pitanja...

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# Pouzdanost računalnih sustava

---

**Predavanje 11: Neosjetljivost na pogreške u  
raspodijeljenim sustavima**

**Prof.dr.sc. Vlado Sruk**



**Sveučilište u Zagrebu**  
**Fakultet elektrotehnike i računarstva**  
*Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave*



# Sadržaj

---

- Raspodijeljeni sustavi
- Neosjetljivost na pogreške u RS
- Bizantinski model zatajenja.

# Raspodijeljeni sustavi

---

- osnovna svojstva:
  - Paralelne aktivnosti neovisnih komponenti
  - Komunikacija razmjenom poruka
  - Dijeljenje sredstava
  - Nema globalnog stanja
  - Nema globalnog vremenskog takta
- arhitektura
  - programska
  - sklopovska
- Lamport: “*A distributed system is one in which the failure of a computer you didn’t even know existed can render your own computer unusable.*”

# Svojstva RS

---

- osnovni elementi raspodijeljenih računalnih sustava
  - sklopovlje
  - programi
  - mrežna komunikacija
- mogu biti otporniji na kvarove
- utjecaj djelomičnog kvara na cijeli sustav?
- svojstvo prikrivenosti uvišeSTRUČAVANJA

# Neosjetljivost na pogreške

---

## ■ Kako definirati kvar RS

- nemogućnost obavljanja potpune usluge
  - djelomično zatajenje
  - mogući utjecaj na druge sustave

## ■ Sredstva za postizanje

- redundancija
  - sklopoljia
  - programa

# Pouzdana komunikacija klijent poslužitelj

---

- Osnovni model klijent poslužitelj: 2 procesa povezana komunikacijskim kanalom
- Mogući problemi:
  - klijent ne može pronaći poslužitelja
  - zahtjev klijenta ne dolazi do poslužitelja
  - poslužitelj ne obrađuje zahtjev klijenta
  - odgovor poslužitelja ne dolazi do klijenta

# Transakcije

---

- mehanizam najčešće primjenjivan u bazama podataka
  - nisu pogodne za općenitu primjenu
- omogućavaju oporavak od pogreške
- ne povećavaju postizanje raspoloživosti
  - replikacija/uvišestručavanje

# Uvišestručavanje

---

- dva pristupa
  - distribuirane transakcije za višestruke kopije podataka
    - svi poslužitelji jednaki
    - 2PC
    - pristup 1-copy serializability
  - razlikovanje primarnog poslužitelja i vruče pričuve
- da li je moguće izgraditi 2PC protokol neosjetljivim na pogreške

# uvišestručavanje kod 2PC

---

- replika podatka

- čitanje jedne kopije
  - osvježavanje svih

- gdje su kopije?

- staticki
  - dinamički

- Metoda kvoruma

- svaka kopija ima kvorum oavježavanja i čitanja
  - $Q_u + Q_r > \text{kopija}$  i  $Q_u + Q_u > \# \text{kopija}$

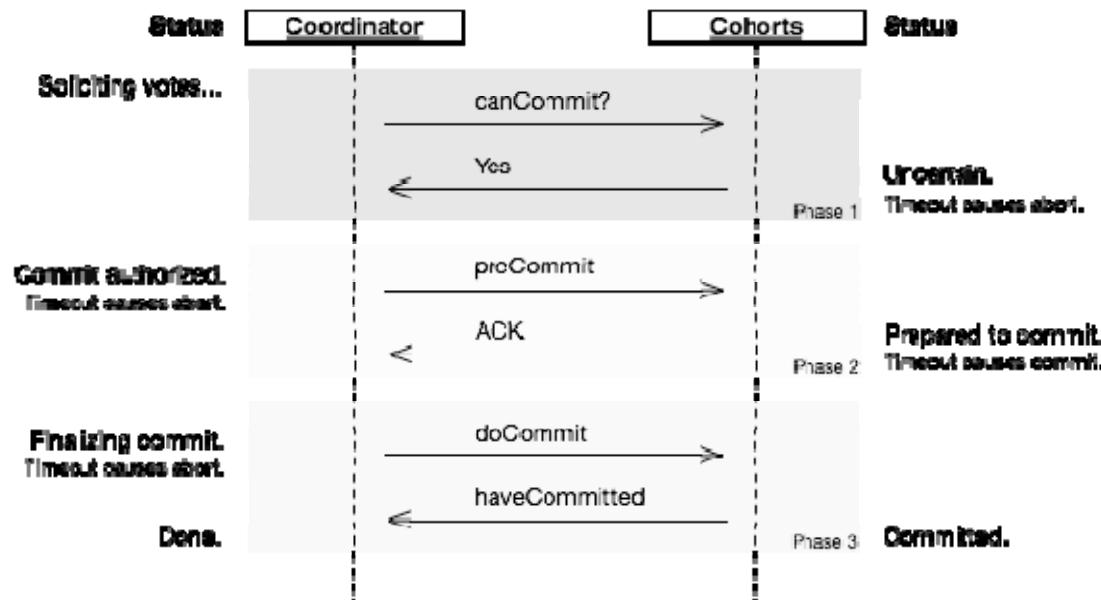
# primjer

---

- podatak uvišeestručen na čvorovima {a,b,c,d,e}
  - $Q_u = 1, Q_r = 5 \ (Q_U+Q_u > 5)$
  - $Q_u = 2, Q_r = 4$
  - $Q_u = 3, Q_r = 3$
  - $Q_u = 4, Q_r = 2$
  - $Q_u = 5, Q_r = 1 \ (\text{nije raspoloživo})$
- problem performansi

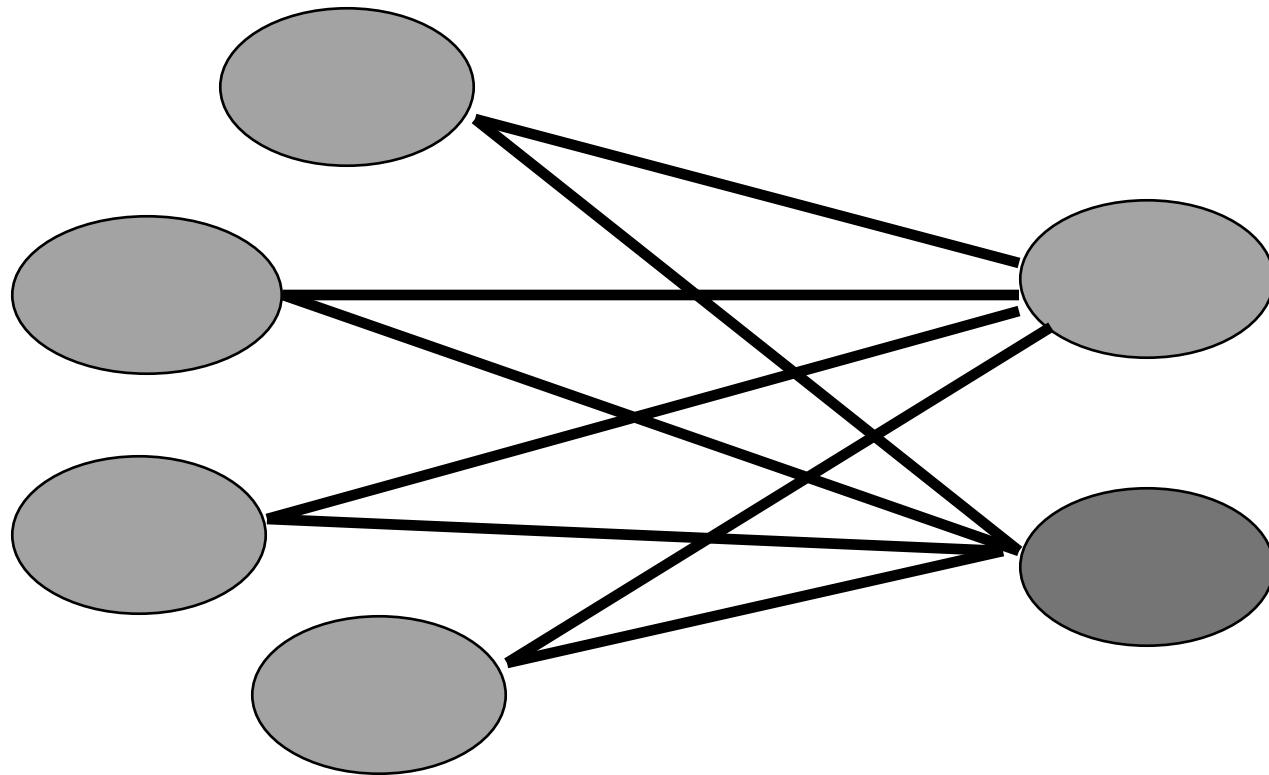
# proširenje 2PC

- 2PC ne garantira raspoloživost
- ako možemo pouzdano detektirati kvar možemo ga i tolerirati >> 3PC
  - problemi mreže nerazdvojni
  - nismo riješili problem



# uvišestručavanje poslužitelja

---



# problem blokiranja

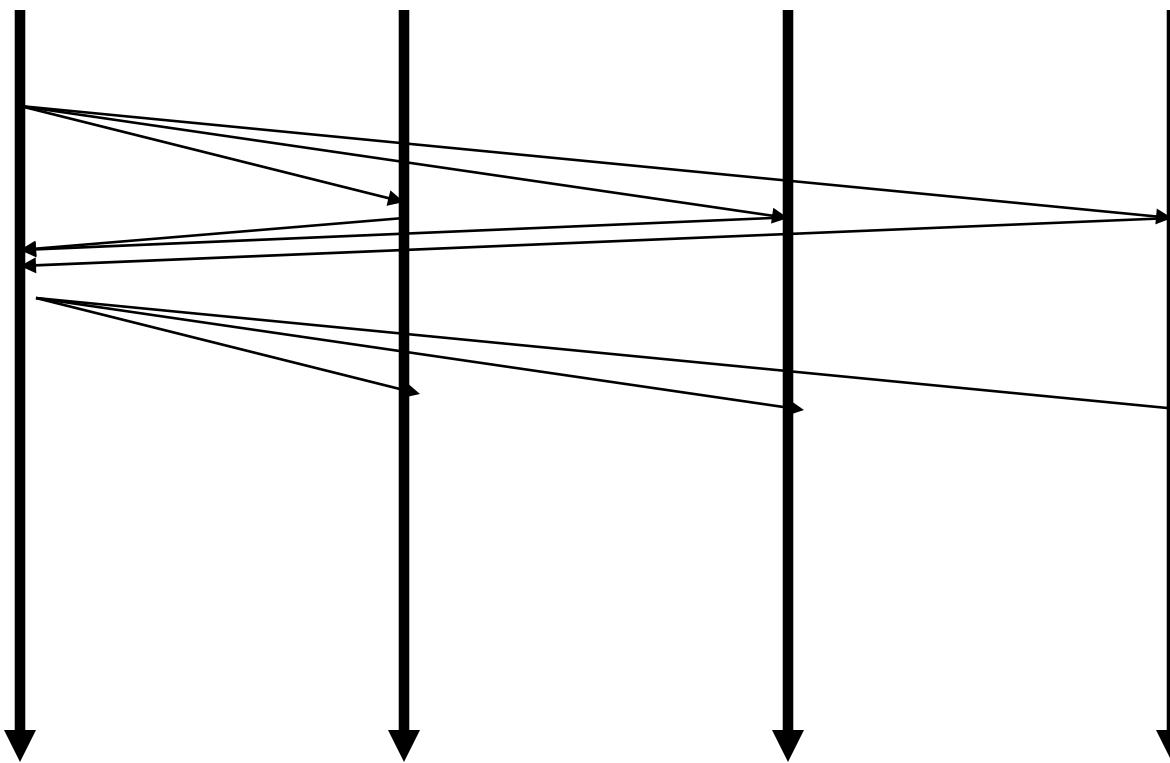
---

- Cilj: izgradnja protokola koji omogućava rad i u slučaju kvarova
- tipični protokol
  - koordinator postavlja upit o mogućnosti obavljanja akcije
  - proces odlučuje da/ne
  - koordinator prikuplja odgovore
  - koordinator obrađuje odgovore

# primjer

---

*commit?*



# problem zatajenja

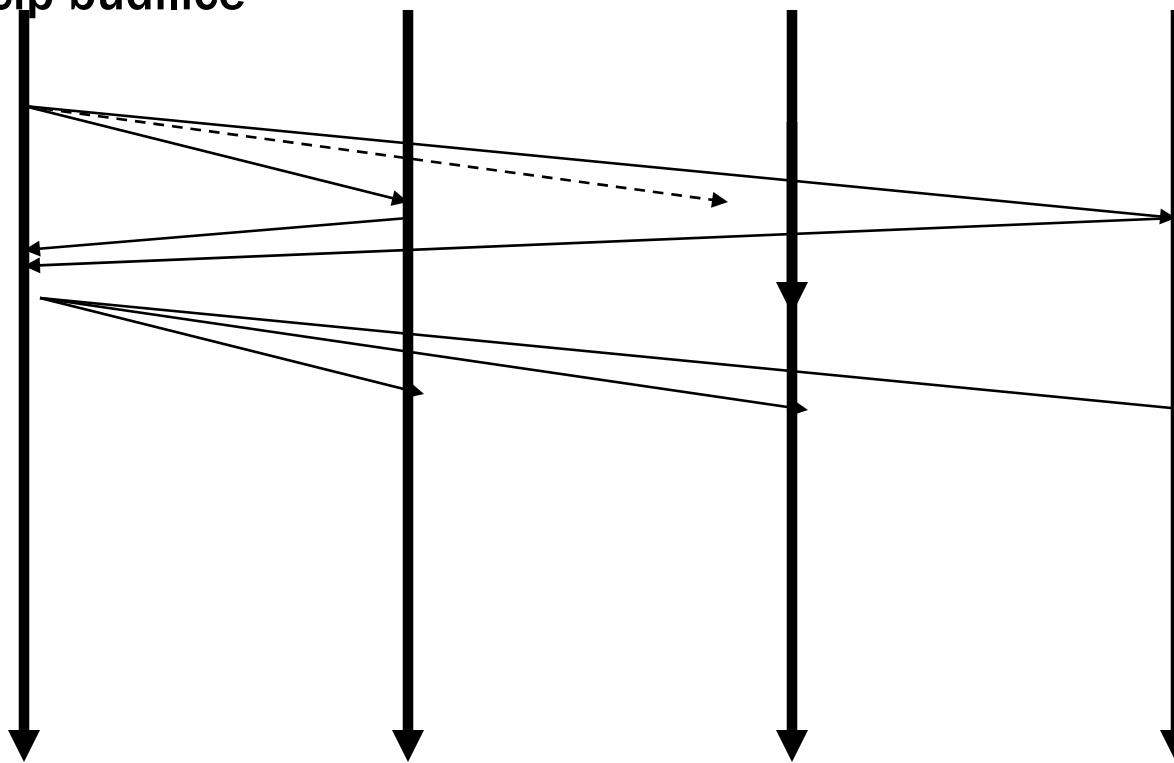
---

- mnogo načina zatajenja
- između ostalih i Bizantinske zatajenja
  - Byzantine model
    - $3t+1$  procesa za obradu t zatajenja
    - visoka cijena – rijetka uporaba

# modifikacija protokola

- ublažene pretpostavke zatajenja
  - koordinator nepouzdano otkriva pogreške

- princip budilice



- ako zataji koordinator?

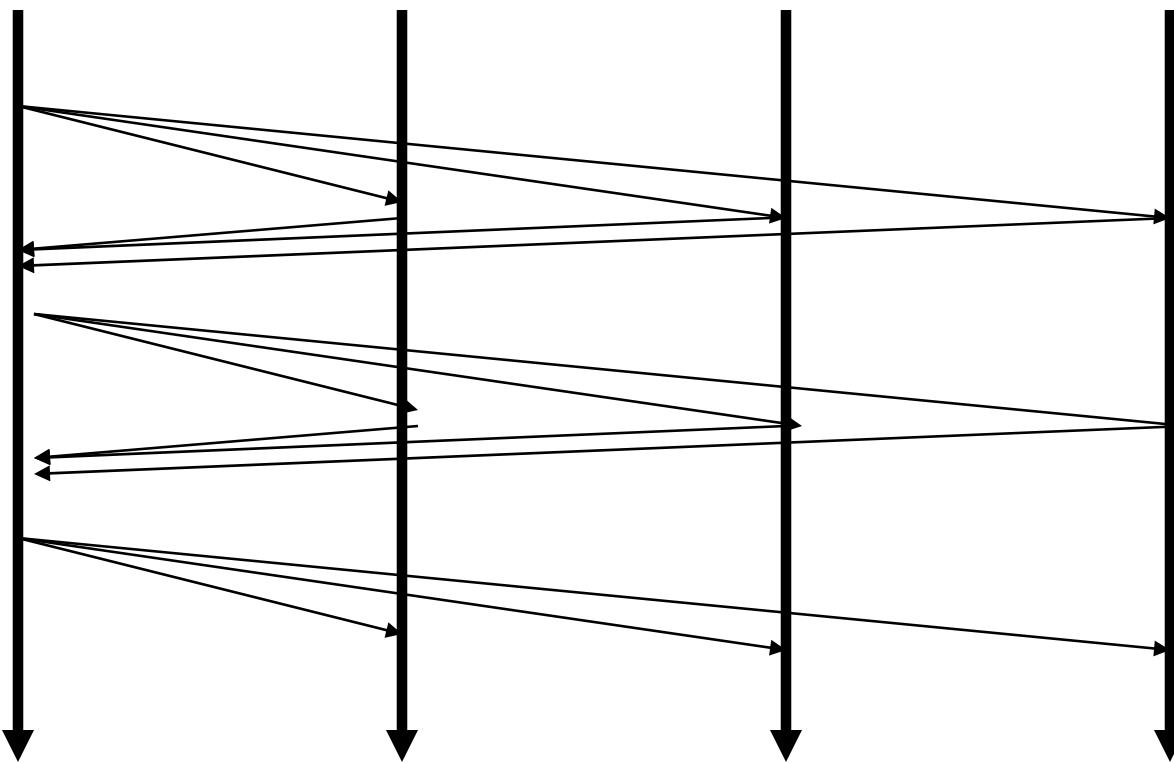
# Modificirani Three-phase commit

---

- povećanje raspoloživosti
- princip rada
  - koordinator šalje zahtjev
  - glasovanje
  - koordinator prikuplja odzive i odlučuje
  - koordinator može odustati
  - koordinator šalje zahtjev pripreme
  - svi potvrđuju

# primjer

---



# Problem dogovora u raspodijeljenim sustavima

---

- Specifičnost RS je problem postizanja dogovora kada se jednom ili više dionika ne može vjerovati
  - uporaba koordinatora ili transakcija
  - većinsko glasovanje može tolerirati  $K$  neispravnih odgovora za  $2K+1$  dionika
- Cilj raspodijeljenog sustava postizanje suglasja u što manjem broju koraka

# Primjer: problem dvije vojske

---

- Crvena vojska ima 5000 vojnika
- Plava vojska ima dvije disperzirane grupe (1 i 2) s po 3000 vojnika
  - zasebnim napadima gube ( $5000 > 3000$ ) ali zajednički mogu pobijediti
- komunikacija je nepouzdanim kanalom
- Scenario:
  - Plavi1 šalje poruku Plavi2 "Napad sutra u zoru"
  - Plavi2 odgovara "Izvrsno, napadnimo"
  - međutim Plavi1 zna da Plavi2 nema potvrdu odgovora, te šalje pouku "primljeno, sve spremno"
  - itd. razmjena poruka s potvrdama
- Plavi nikada ne mogu biti sigurni radi nepouzdane komunikacije
  - NEMOGUĆE POSTIĆI DOGOVOR

# Primjer 2: problem izdajica

---

- Pouzdana komunikacija, nepouzdani proces
- Preduvjeti:
  - 1 vojska crvenih
  - N generala plave vojske
  - M generala izdajica želi zbuniti ostale generale ( $N-M$ )
  - pouzdana komunikacija
- Kako postići ispravan dogovor?
  - pretpostavka vojnici lojalnog generala odradjuju ispravan zadatak
  - za vojнике nelojalnog generala nije poznato, no najvjerojatnije rade krivo od cilja vojske Plavih

# Algoritam

---

- Lamport za  $N=4, M=1$
- svi plavi generali šalju ostalima poruke ( $N-1$ ). (izdajice lažu)
- rezultat prikupljenih poruka daje  $\text{vector}[N]$
- svi generali šalju  $\text{vector}[N]$  ostalim  $N-1$  generalima
- svi generali ispituju dobivene poruke tražeći većinski odgovor za svakog generala
- algoritam radi uspješno jer izdajice ne mogu promijeniti poruke lojalnih generala
- minimaln broj lojalnih generala u slučaju  $M$  izdajica je  $2M+1$  t.j. ukupno  $3M+1$  generala

# Pitanja...

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---