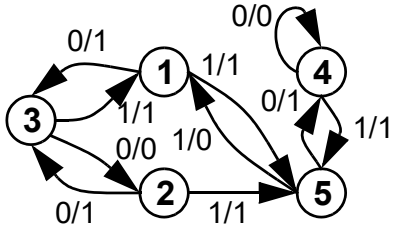
Sünkroonsete loogikaskeemide optimeerimine

- Oleku-põhised mudelid
  - siirde diagrammid või tabelid (FSM)
  - ilmutatud olek / ilmutamata pindala ja viide
  - olekute arvu minimeerimine
- Struktuursed mudelid
  - sünkroonsed loogikavõrkgraafid
  - ilmutamata olek / ilmutatud pindala ja viide
  - skeemi optimeerimine
- Sarnane mitmetasemeliste loogikafunktsioonide minimeerimisega
- Esitusviis – modifitseeritud loogikavõrkgraaf
  - mäluelemendid / registrid

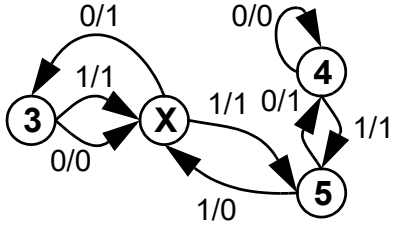
Automaadi minimeerimine

- **Automaadi minimeerimine – olekute arvu vähendamine**
- **Ekvivalentsed olekud – võrdsete sisendjadade korral väljundjadad katuvad**




```
I: 001001100011...
O: 101101010111...
S: 1325445132315...
```

"must kast #1" == "must kast #2"




```
I: 001001100011...
O: 101101010111...
S: X3X5445X3X3X5...
```

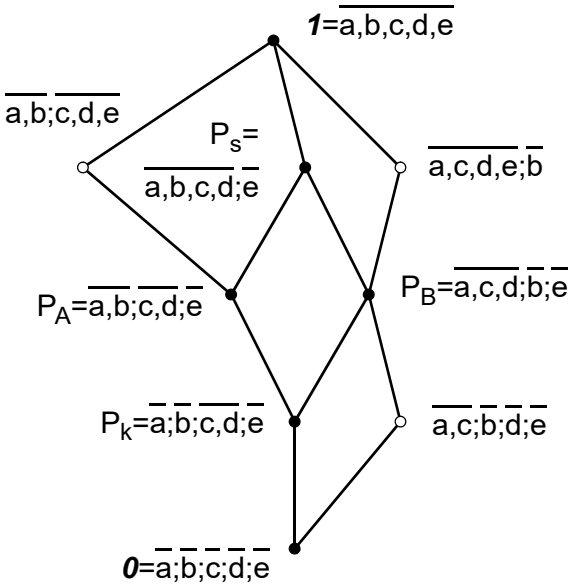
- **Intuitiivne lähenemine**
  - **kaks olekut on ekvivalentsed, kui nende siirded on identsed (ei pruugi alati töötada!)**
- **Süstemaatiline lähenemine**
  - **olekud 1 & 2 on ekvivalentsed**
  - **leida selline olekute hulga tükeldus, et ekvivalentsed olekud oleksid samas tükelduste plokis (ekvivalentsiklassis)**
  - **olekute hulk S={1,2,3,4,5}, tükeldus P={ $\overline{1,2}$; $\overline{3}$; $\overline{4}$; $\overline{5}$ } (vt. asendusomadusega tükeldus)**

Algebraline struktuuriteooria – Tükeldused

- Vastavuse φ erijuht, lähte- ja sihthulk langevad kokku - $D(\varphi) = R(\varphi) = A$
  - Tähistus – $R \subseteq A \times A$
  - Ekvivalentsisuhe R on refleksiivne, sümmeetriline ja transitiivne
  - Elemendi $a \in A$ ekvivalentsiklass ekvivalentsisuhtes R – $K(a) = \{ b \mid \langle a,b \rangle \in R \}$
- Ekvivalentsisuhe genereerib tükelduse (partition) P hulgal A
  Tükeldus P koosneb ekvivalentsiklassidest K_i, i=1,...,n
  $P = \{ K_1, K_2, ..., K_n \}$, kus $K_i \neq \varnothing$, i=1,...,n;
  $K_i \cap K_j = \varnothing$, i,j=1,...,n, $i \neq j$;
  $\cup K_i = A$
  - 0-tükeldus (nulltükeldus) koosneb 1-elemendilistest ekvivalentsi klassidest
  - 1-tükelduses (ühiktükelduses) on ainult üks ekvivalentsiklass
- $B_P(a)$ – tükelduse P ekvivalentsiklass (plokk) K_i, millesse kuulub a
  - $K_i = B_P(a) \Rightarrow a \in K_i$
- $a_1 \equiv a_2$ (P) – elemendid a_1 ja a_2 kuuluvad samasse tükelduse P plokki K_i
  siis ja ainult siis, kui $B_P(a_1) = B_P(a_2)$
- Tükelduste kasutamine – dekompositsioon ja kompositsioon, olekute kodeerimine
  - automaadi tükeldamine kaheks või enamaks automaadiks
  - kahe või enama automaadi ühendamine

Tükeldus – operatsioonid

- Hulk: A={a,b,c,d,e}, tükeldused: P_A={ $\overline{a,b}$; $\overline{c,d}$; $\overline{e}$ } ja P_B={ $\overline{a,c,d}$; $\overline{b}$; $\overline{e}$ }
  - P_A={ {a,b}, {c,d}, {e} } ja P_B={ {a,c,d}, {b}, {e} }
- Korrutis: $P_1 \bullet P_2$: $(a_1 \equiv a_2\ (P_1 \bullet P_2)) \Leftrightarrow (a_1 \equiv a_2\ (P_1)\ \&\ a_1 \equiv a_2\ (P_2))$
  - $P_k = P_A \bullet P_B$ = { $\overline{a,b}$; $\overline{c,d}$; $\overline{e}$ }•{ $\overline{a,c,d}$; $\overline{b}$; $\overline{e}$ } = { $\overline{a}$; $\overline{b}$; $\overline{c,d}$; $\overline{e}$ }
- Summa: $P_1 + P_2$: $(a_1 \equiv a_2\ (P_1 + P_2)) \Leftrightarrow (a_1 \equiv a_2\ (P_1)\ V\ a_1 \equiv a_2\ (P_2))$
  - $P_s = P_A + P_B$ = { $\overline{a,b}$; $\overline{c,d}$; $\overline{e}$ }+{ $\overline{a,c,d}$; $\overline{b}$; $\overline{e}$ } = { $\overline{a,b,c,d}$; $\overline{e}$ }
- Võrdlus:
  $P_1 \leq P_2 \Leftrightarrow P_1 \bullet P_2 = P_1 \Leftrightarrow [\ \forall\ K_i \in P_1\ (\ \exists\ K_j \in P_2\ [K_i \subseteq K_j]\ )\ ]$
  - $P_A \leq P_s$ – { $\overline{a,b}$; $\overline{c,d}$; $\overline{e}$ }•{ $\overline{a,b,c,d}$; $\overline{e}$ } = { $\overline{a,b}$; $\overline{c,d}$; $\overline{e}$ }
  - P_A ja P_B ei ole võrreldavad
- Tükelduste võre
  - Tehted ~~ asukoht võrel
  - Võrreldavus ~~ asukoht võrel
- Null-tükeldus (*0*) – iga element on omaette plokis
- Ühik-tükeldus (*1*) – kõik elemendid on samas plokis



Asendusomadus (S.P.)

- Olekute grupid, mis teistest ei sõltu
- Automaadi M=(S,I,O,δ,λ) olekute hulga tükeldusel P on *asendusomadus* (substitution property, S.P.) siis ja ainult siis, kui
  s $\equiv$ t(P) järeldub, et ka δ(s,i) $\equiv$ δ(t,i)(P) $\forall$ i$\in$I
  - kui lähteolekud kuuluvad ühte tükelduse plokki, siis ka järgmised olekud peavad kuuluma ühte plokki (ei pruugi kokku langeda lähteolekute plokiga) kõikide sisendkombinatsioonide korral
  - tükeldus P - kasutusel ka tähistus π
- Asendusomadustega tükeldused:
  - P_A = {$\overline{1,2,3}$; $\overline{4,5,6}$}
  - P_B = {$\overline{1,6}$; $\overline{2,5}$; $\overline{3,4}$}
- 0-tükeldus ja 1-tükeldus on S.P. tükeldused

x	s_t	s_{t+1}	y
0	1	4	0
1		3	
0	2	6	0
1		3	
0	3	5	0
1		2	
0	4	2	1
1		5	
0	5	1	0
1		4	
0	6	3	0
1		4	

Tükelduste kasutamine kodeerimisel

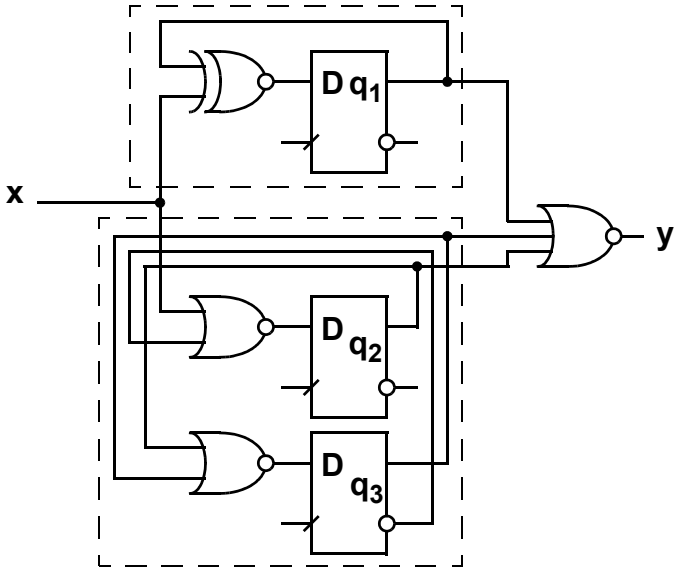
- Tükeldustel baseeruv lähenemine
  - automaadi olekute hulga tükeldamine selliselt, et tükelduste plokid omaksid võimalikult väikest "vastastikust sõltuvust"
- Leitakse automaadi asendusomadusega (S.P.) tükeldused
  - väljundeid ei vaadelda
  - iga tükeldus vastab sõltumatule osale
  - teostatakse paralleelne dekompositsioon
- Näide
  - S.P. tükeldused - $P_1 = \overline{1,2,3}\ \overline{4,5,6}$, $P_2 = \overline{1,6}\ \overline{2,5}\ \overline{3,4}$
  - Dekompositsiooniks vajalik, et $P_1 \cdot P_2 = P_0$
    - vajadusel geneeritakse täiendavaid tükeldusi
  - Iga tükeldus (kui automaat) kodeeritakse eraldi:
    - P_1-> 1-{1,2,3} 0-{4,5,6} & P_2-> 10-{1,6} 01-{2,5} 00-{3,4}
    - Olekute koodid: 1->110, 2->101, 3->100, 4->000, 5->001, 6->010

x	s_t	s_{t+1}	y
0	1	4	0
1		3	
0	2	6	0
1		3	
0	3	5	0
1		2	
0	4	2	1
1		5	
0	5	1	0
1		4	
0	6	3	0
1		4	

Tükeldused, kodeering ja realisatsioon

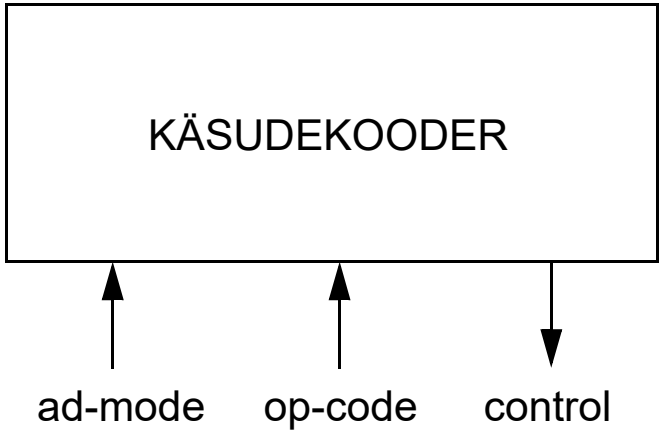
- Olekute koodid: 1->110, 2->101, 3->100, 4->000, 5->001, 6->010
- Tükeldused: $P_1=\{\overline{1,2,3};\ \overline{4,5,6}\}$, $P_2=\{\overline{1,6};\ \overline{2,3,4,5}\}$, $P_3=\{\overline{2,5};\ \overline{1,3,4,6}\}$
  - kaheplokilised tükeldused: 1 - parempoolne plokk, 0 - vasakpoolne plokk
- $P_1 \bullet P_2 \bullet P_3=\{\overline{1};\overline{2};\overline{3};\overline{4};\overline{5};\overline{6}\}=0$ & $P_1+P_2+P_3=\{\overline{1,2,3,4,5,6}\}=1$

- **Automaatide võrk - kaks paralleelselt töötavat automaati**
  - $d_1=q_1x+\overline{q_1}\,\overline{x} = x\text{nor}(q_1,x)$
  - $d_2=q_3\overline{x} = \text{nor}(\overline{q_3},x)$
  - $d_3=\overline{q_2}\,\overline{q_3} = \text{nor}(q_2,q_3)$
  - $y=q_1q_2q_3 = \text{nor}(q_1,q_2,q_3)$
  - Tükeldused:
    - 1. automaat: $P_A=\{\overline{1,2,3};\ \overline{4,5,6}\}$ $(=P_1)$
    - 2. automaat: $P_B=\{\overline{1,6};\ \overline{2,5};\ \overline{3,4}\}$ $(=P_2 \bullet P_3)$



Sümboolne minimeerimine ja kodeerimine

- Sümbolite tabelite minimeerimine
- Kahend- ja MV-loogika funktsioonide minimeerimise edasiarendus
  - Rakendused – kodeerimine
    - operatsioonikoodid, automaadi olekud, jne.
  - Lahendatavad probleemid
    - sisendite, väljundite & kombineeritud kodeerimine
- Näide



ad-mode	op-code	control
INDEX	AND	CNTA
INDEX	OR	CNTA
INDEX	JMP	CNTA
INDEX	ADD	CNTA
DIR	AND	CNTB
DIR	OR	CNTB
DIR	JMP	CNTC
DIR	ADD	CNTC
IND	AND	CNTB
IND	OR	CNTD
IND	JMP	CNTD
IND	ADD	CNTC

Näide – sisendite kodeerimine

ad-mode	op-code	control
INDEX	AND	CNTA
INDEX	OR	CNTA
INDEX	JMP	CNTA
INDEX	ADD	CNTA
DIR	AND	CNTB
DIR	OR	CNTB
DIR	JMP	CNTC
DIR	ADD	CNTC
IND	AND	CNTB
IND	OR	CNTD
IND	JMP	CNTD
IND	ADD	CNTC

1-hot / MV	
100 1000	1000
100 0100	1000
100 0010	1000
100 0001	1000
010 1000	0100
010 0100	0100
010 0010	0010
010 0001	0010
001 1000	0100
001 0100	0001
001 0010	0001
001 0001	0010

1-hot / MV	
100 1111	1000
010 1100	0100
001 1000	0100
010 0011	0010
001 0001	0010
001 0110	0001

ad-md.	op-code	ctrl
INDEX	AND,OR,JMP,ADD	CNTA
DIR	AND,OR	CNTB
IND	AND	CNTB
DIR	JMP,ADD	CNTC
IND	ADD	CNTC
IND	OR,JMP	CNTD

Sisendite kodeerimine – lahendamine

- Minimaalse sümbolkatte teisendamine minimaalseks binaarkatteks
- Sümbolimplikantide vastavus kahendimplikantidega
- Liitimplikandid
  - vastavad sümbolid tuleb kodeerida selliselt, et vastav superkuup ei sisalda teisi sümboleid
- Sümbolid asendada kattes koodidega

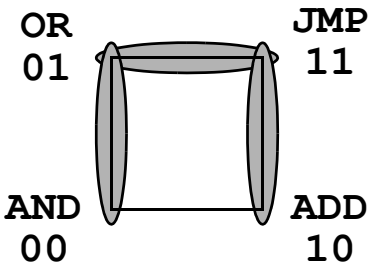
Liitliteraalid

AND,OR,JMP,ADD

AND,OR

JMP,ADD

OR,JMP



AND	00
OR	01
JMP	11
ADD	10

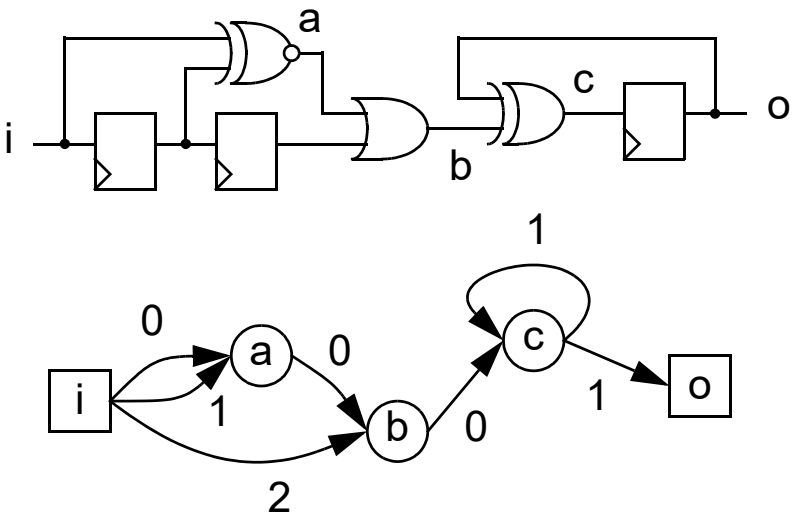
INDEX	00
DIR	01
IND	11

Kodeeritud kate

ad.	op.	ctrl
00	--	1000
01	0-	0100
11	00	0100
01	1-	0010
11	10	0010
11	-1	0001

Automaadi skeemi minimeerimine

- Loogikavõrkgraaf (logic network)
  - omavahel ühendatud loogikafunktsioonid
  - kombineeritud struktuurne/käitumuslik mudel
- Sünkroonne loogikavõrkgraaf
  - sünkroonne viide
    - sõlmed == võrrandid == s/v, loogikalülid
    - servad == sõltuvused == ahelad
    - kaalud == sükroonne viide == registrid
  - Pindala ennustuse minimeerimine
    - literaalide arv / funktsioonide/loogikalülide arv
    - arvestada tuleb viite piiranguid
  - Suurima viite minimeerimine
    - tee sügavus / loogikalülide mudelid / olulised teed
    - arvestada tuleb pindala (võimsustarbe) piiranguid
  - Testitavuse maksimaliseerimine ja/või võimsustarbe minimeerimine

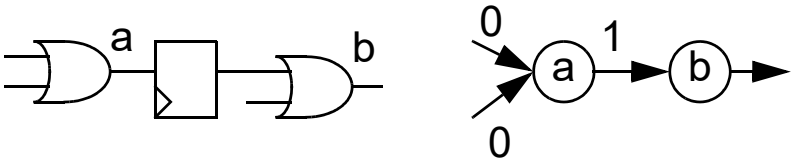


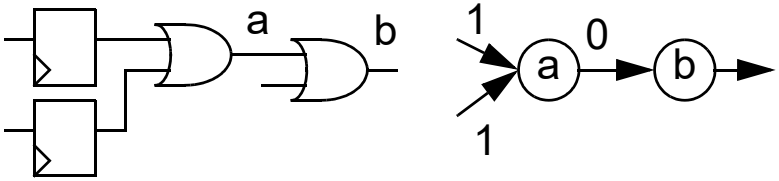
Optimeerimine

- Optimeeritakse ainult kombinatoorset loogikat
  - loogika minimeerimine
    - kahe- ja/või mitme-tasemeline minimeerimine
- Optimeeritakse ainult mäluelementide / registrite asukohti
  - *retiming* (ümberajastamine)
- Optimeeritakse tervet skeemi
  - sisendi/väljundi ajastuse muutmine
    - sünkroonsed teisendused – algebralised / Boole'i
- Optimeerimis-strateegiad
  - Samm-sammuline parendamine
    - teisendused (transformations) võrkgraafil
  - Funktsionaalsus ei tohi muutuda
  - Erinevad meetodid
    - erinevad teisenduste variandid / erinevad teisenduste rakendamise järjekorrad

Retiming (ümberajastamine) – optimeerimise etapid

- Registrid eraldatakse kombinatoorsest osast
- Kombinatsioonskeemide minimeerimine
  - avaldiste modifitseerimine
  - graafi struktuuri modifitseerimine
- Registrid ühendatakse tagasi
- Registrite asukohti muudetakse
  - kombinatoorne osa ei muutu
  - muutuvad kriitilised teed
- Võrkgraafi struktuur ei muutu
  - muutuvad kaalud
  - graafi struktuur ei muutu





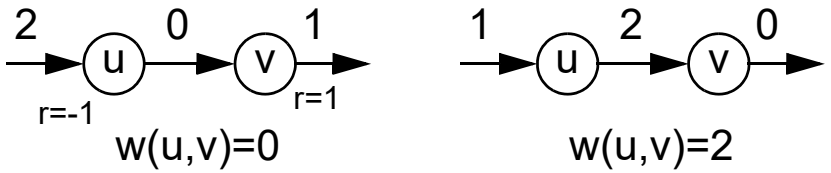
Retiming

- Globaalne optimeerimistehnika
  - mõjutab pindala – registrite arv muutub
  - mõjutab viidet – teede pikkused registrite vahel muutuvad
  - lahendatav polünomaalse ajaga
- Eeldused
  - sõlme viide on konstantne
    - väljundite arvust sõltuvat viite komponenti ei arvestata
  - graafi topoloogia on invariantne
    - ei teostata loogika teisendusi
  - sünkroonne realisatsioon
    - tsüklitel on positiivsed kaalud
    - servadel on mitte-negatiivsed kaalud
  - arvestatakse ainult topoloogilisi teid
    - “false path” analüüsi ei teostata

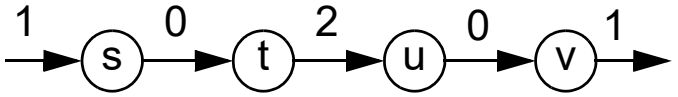
Retiming – definitsioone ja omadusi

- **Sõlme retiming**
  - täisarvuline teisendus, register nihutatakse väljundist sisendisse
- **Skeemi retiming**
  - sõlmede retiming'ute vektor
- **Ekvivalentsete skeemide kogum**
  - originaalne skeem + retiming vektorid

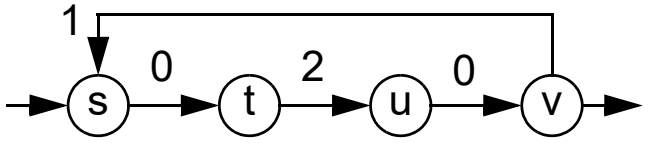
serva retiming



- **Definitsioonid**
  - $w(v_i,v_j)$ – serva (v_i,v_j) kaal
  - $(v_i,...,v_j)$ – tee sõlmest v_i sõlmeni v_j
  - $d(v_i,...,v_j)$ – tee $(v_i,...,v_j)$ viide (sõlmest v_i sõlmeni v_j)
- **Omadused**
  - serva retiming: $\overline{w}_{ij} = w_{ij}+r_j-r_i$
  - tee retiming: $\overline{w}(v_i,...,v_j) = w(v_i,...,v_j)+r_j-r_i$
  - tsüklite kaalud on invariantsed



tee w(s,t,u,v)=2



tsükkel w(tsükkel)=3

Legaalne retiming

- **Taktsignaali periood ϕ**

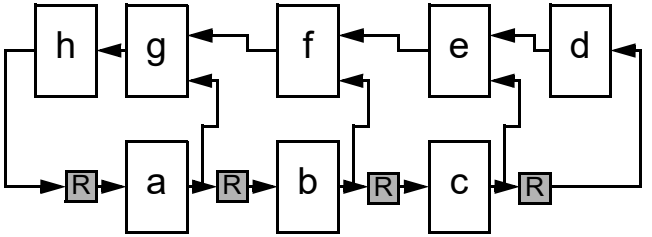
- **Retiming'u vektor peab tagama:**
  - **ühegi serva kaal ei ole negatiivne:** $\overline{w}_{ij} = w_{ij}+r_j-r_i \geq 0 \ \forall i,j$
  - **iga tee $(v_i,...,v_j)$, mille puhul $d(v_i,...,v_j) \geq \phi$, omab vähemalt ühte registrit:**

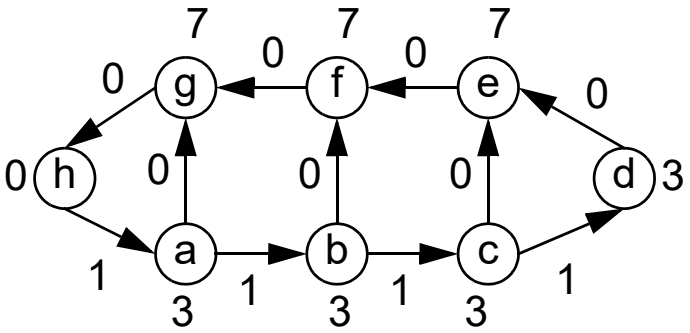
    $\overline{w}(v_i,...,v_j) = w(v_i,...,v_j)+r_j-r_i \geq 1 \ \forall i,j$

- **Originaalne graaf ei oma tsükleid negatiivse kaaluga, seega ka uus graaf ei oma tsükleid negatiivse kaaluga**

Retiming – optimeerimisülesanne

- **Lühim registrite tee – $W(v_i,v_j)=\min w(v_i,...,v_j)$ [kõik teed v_i ja v_j vahel]**
- **Kriitiline viide – $D(v_i,v_j)=\max d(v_i,...,v_j)$ [kõik teed v_i ja v_j vahel kaaluga $W(v_i,v_j)$]**
- **Leida selline minimaalne taktsignaali periood ϕ, et eksisteeriks retiming vektor, mille puhul**
- $r_i - r_j \leq w_{ij} \quad \forall (v_i,v_j) \in E$
- $r_i - r_j \leq W(v_i,v_j)-1 \quad \forall v_i,v_j \mid D(v_i,v_j) > \phi$





sõlmed: a & e
teed: (a,b,c,e) & (a,b,c,d,e)
W(a,e)=2 & D(a,e)=16

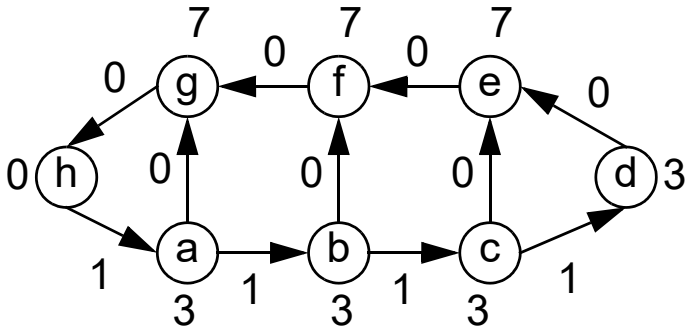
Relaktsioonil põhinev algoritm

- Otsib suure viitega teid
- Lühendab selliseid teid otsmiste registri “koomale tõmbamise” abil
  - mõni teine teed võib muutuda liiga pikaks
  - eriti need teed, mille “saba” liikus
- Kasutab iteratiivset lähenemist

- Iga sõlme jaoks leitakse andmete valmisoleku aeg (data ready time)
  - koguviide registrist alates
- Iteratiivne lähenemine
  - leiab sõlmed, mille andmete valmisoleku aeg > ϕ
  - sellistel sõlmedel teostatakse retiming 1 võrra (register väljundist sisendisse)
- Omadused
  - legaalne retiming leitakse |N| iteratsiooniga (kui üldse leidub)

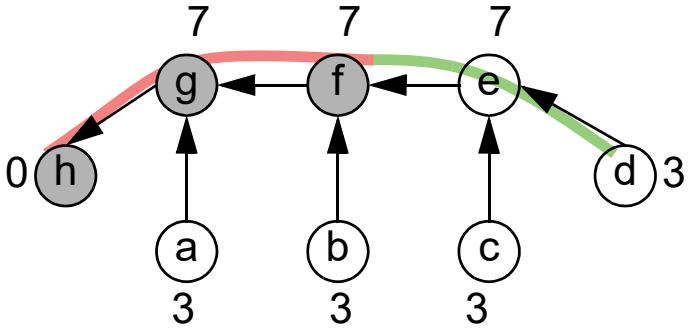
Retiming – näide

- Ülemine gaaf – originaalne skeem
- Alumine graaf – registrid eemaldatud
  - kombinatsioonskeem



- ϕ = 13

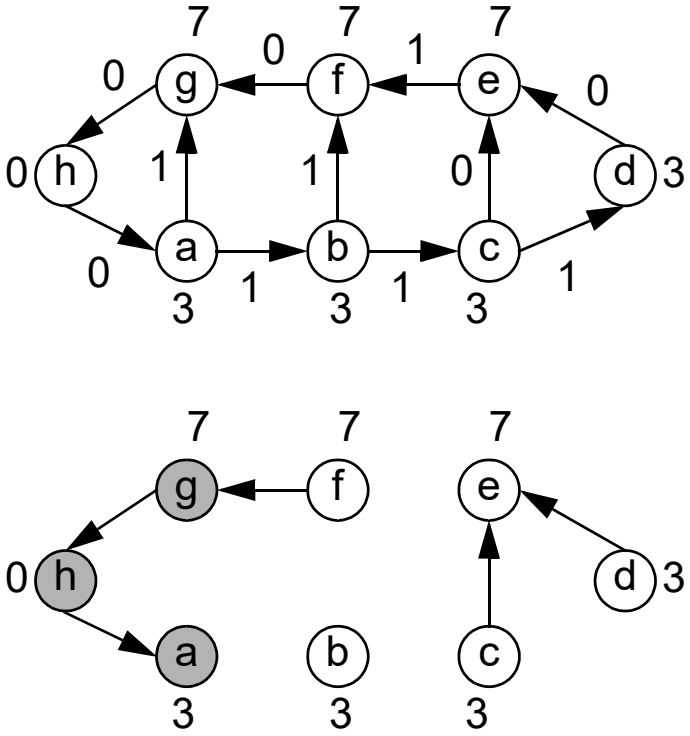
- andmete valmisoleku ajad:
  - t_a=3, t_b=3, t_c=3, t_d=3, t_e=10, t_f=<u>17</u>, t_g=<u>24</u>, t_h=<u>24</u>



- sõlmedele f, g ja h tehakse retiming 1 võrra

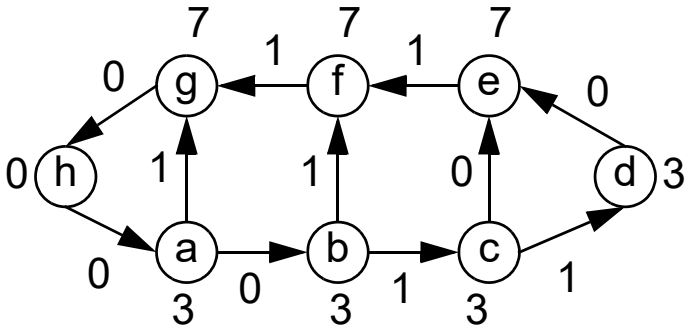
Retiming – näide (järg)

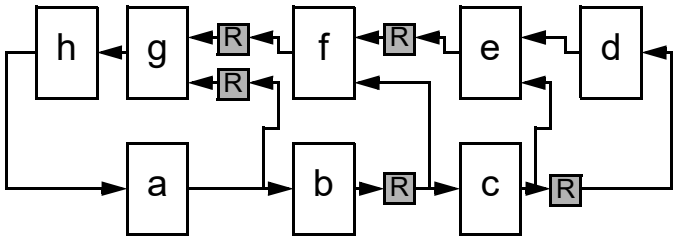
- andmete valmisoleku ajad:
  - t_a=<u>17</u>, t_b=3, t_c=3, t_d=3, t_e=10, t_f=7, t_g=<u>14</u>, t_h=<u>14</u>
- sõlmedele a, g ja h tehakse retiming 1 võrra

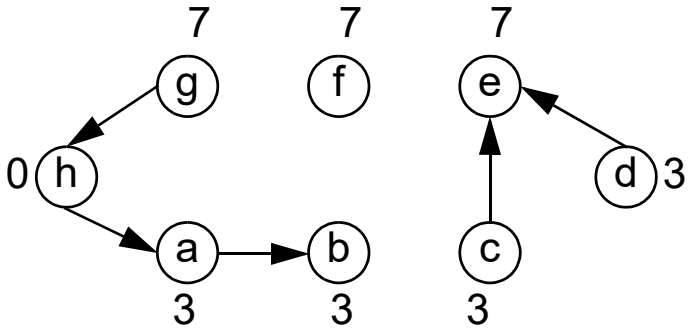


Retiming – näide (järg)

- **andmete valmisoleku ajad:**
  - t_a=10, t_b=13, t_c=3, t_d=3, t_e=10, t_f=7, t_g=7, t_h=7
- **lõpp –** $\forall\ t_i \leq \phi$

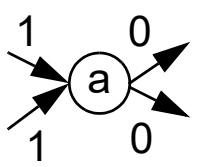


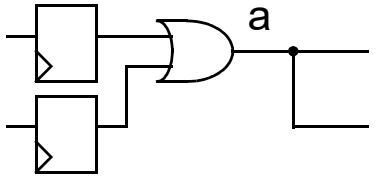




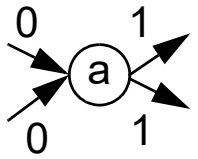
Retiming – rakendatavus

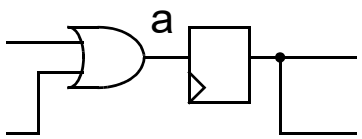
- **Registrite arvu minimeerimine**
  - **sünkroonse viite nihutamine sisenditest väljundisse**





- **Kontroll-osa**
  - **abstrakne automaadi mudel peidetud võrkgraafi sisse**





- **Andme-osa**
  - **loogikafunktsioonid**
  - **mäluelemendid / registrid**
- **Mitte ainult loogikaskeemid vaid ka kõrgemad abstraktsiooni tasemed**
  - **sõlmed – funktsioonid**
  - **sünkroonsed viited – nt. kriitilised ressursid (mälud)**

Digitaalsüsteemide testimise alused

Funktsioonide teisendused – kahendmeetodid

- Kasutavad fakti, et osa sisendmuutujaid ja/või võrgugraafi sõlmi ei mõjuta osasid funktsioone
  - toob sisse täiendavad määramatused
  - võimalik kasutada lokaalset kahendminimeerimist

- *Juhitavus* (controllability)
  - sisendkombinatsioonid, mida ei esine võrgu sisendis (ümbritsevast keskkonnast tingituna)
- *Jälgitavus* (observability)
  - sisendkombinatsioonid, mille korral väljund ei ole vaadeldav keskkonna poolt
  - suhteline iga väljundi jaoks
- Seotud funktsioonide testitavusega

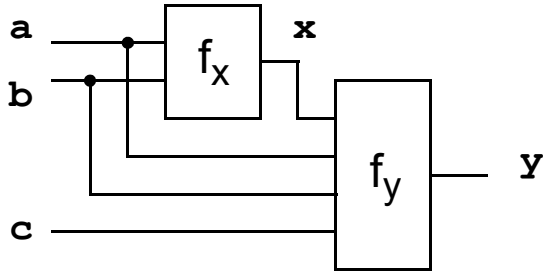
Kahendmeetodid – sisemised määramatused

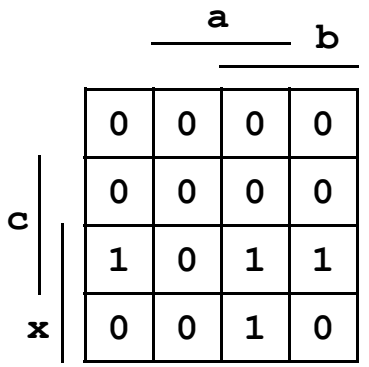
- Võrgu sise-ehitusest sõltuvad
  - Juhitavus – kombinatsioonid, mida ei esine alamvõrgu sisendis
  - Jälgitavus – kombinatsioonid, mille korral alamvõrgu väljundid ei ole vaadeldavad
- Näide #1
  - $x = \overline{a} + b$; $y = a\ b\ x + \overline{a}\ c\ x$;
  - juhitavus – y sisendis ei saa esineda kombinatsiooni $a\ \overline{b}\ x + \overline{a}\ \overline{x} + b\ \overline{x}$
  - minimeerides – $y = a\ x + \overline{a}\ c$
- Näide #2
  - $v = \overline{a}\ d + b\ d + \overline{c}\ d + a\overline{e}$; (dekompositsioon) $j = \overline{a} + b + \overline{c}$; $v = jd + a\overline{e}$;
  - $a=0 \rightarrow j=1$: seega on võimatu kombinatsioon: $a=0\ \&\ j=0 \rightarrow v=$'-'
  - minimeerides – $v = jd + a\overline{e}$ (alati ei muutu lihtsamaks)

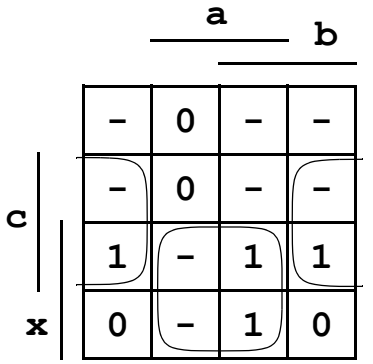
$x = \bar{a} + b; \quad y = abx + \bar{a}cx;$

- $x=0 \rightarrow a=1 \;\&\; b=0$
- $x=1 \rightarrow a=0 \vee b=1$
- Võimatud kombinatsioonid:
  - $x=0 \;\&\; !(a=1 \;\&\; b=0) \rightarrow x=0 \;\&\; (a=0 \vee b=1)$
  - $x=1 \;\&\; !(a=0 \vee b=1) \rightarrow x=1 \;\&\; a=1 \;\&\; b=0$
- Määramatused:
  - $x=0 \;\&\; a=0; \quad x=0 \;\&\; b=1; \quad x=1 \;\&\; a=1 \;\&\; b=0$
- Tulemus

  $y = \bar{a}c + ax$





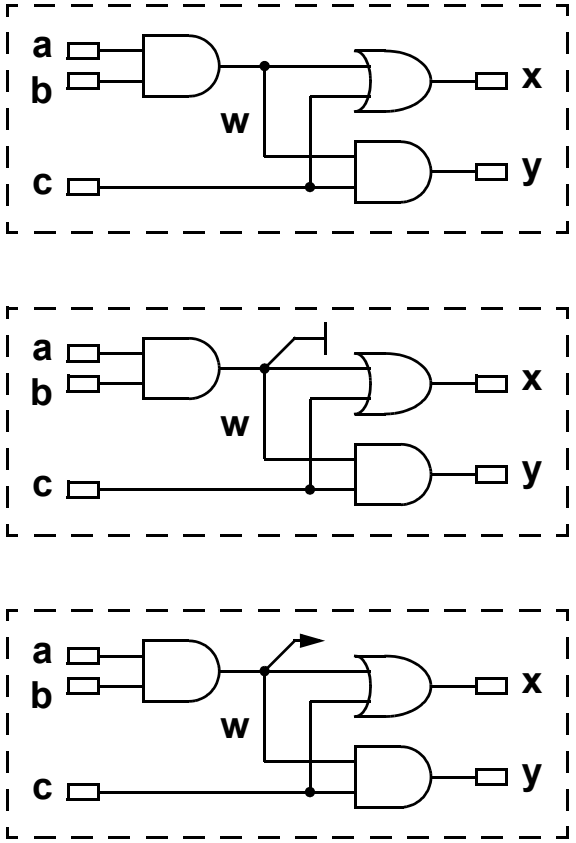


Testitavuse alused

- Rikke mudel – mingi ahela *lühis* 0 või 1-ga (stuck-at-0/stuck-at-1)
- Ahela *w* kontroll lühisele 0-ga
  - sisendkombinatsiooniga seatakse vastav ahel 1-ks
  - võrreldakse väljundeid – vigase skeemi väljund on erinev soovitust
- Ahela *w* kontroll lühisele 1-ga
  - sisuliselt sama, kuid jälgitav ahel seatakse 0-ks
- Ahel *w* peab olema juhitav ja jälgitav
  - testitavus sõltub skeemi struktuurist
- Süntees testitavust silmas pidades
  - võimalikult suur osa sisemisi ahelaid peaks olema jälgitavad ja juhitavad
  - liiasuste eemaldamine mitmetasemelisel loogikafunktsioonide minimeerimisel

Näide

- Skeem: w=ab; x=w+c; y=wc;
- Lühis 0-ga
  w=0; → x=c; y=0;
- Lühis 1-ga
  w=1; → x=1; y=c;
- Kas w on jälgitav?
  - Millised väljundid sõltuvad w-st?
- Kas w on juhitav?
  - Kas leidub sisendkombinatsioon, mis lubab w-le seada soovitud väärtuse?



- **Jälgitavus - Boole'i diferentsiaal - $\partial f/\partial x_i = f_{xi} \oplus f_{x'i}$**
  - **Kas väljund x (x=w+c) sõltub w-st? Kas väljund y (y=wc) sõltub w-st?**
  - **Kofaktorid -- $x_w=1$; $x_{w'}=c$; $y_w=c$; $y_{w'}=0$;**
  - **$\partial x/\partial w = x_w \oplus x_{w'} = 1 \oplus c = c'$ (sõltub siis, kui c==0)**
  - **$\partial y/\partial w = y_w \oplus y_{w'} = c \oplus 0 = c$ (sõltub siis, kui c==1)**
  - **Nii x kui ka y sõltuvad w-st, kuid erinevatel c väärtustel → w on jälgitav väljunditel x ja y (teatud mööndustega)**

- **Juhitavus - soovitud väärtuse seadmine w-l**
  - **w=a b**
  - **lühis 0-ga kontrollimiseks → w==1 → a==1 ja b==1**
  - **lühis 1-ga kontrollimiseks → w==0 → a==0 või b==0**
  - **BDD'd (või muud otsustus diagrammid) sobivad selleks suurepäraselt**

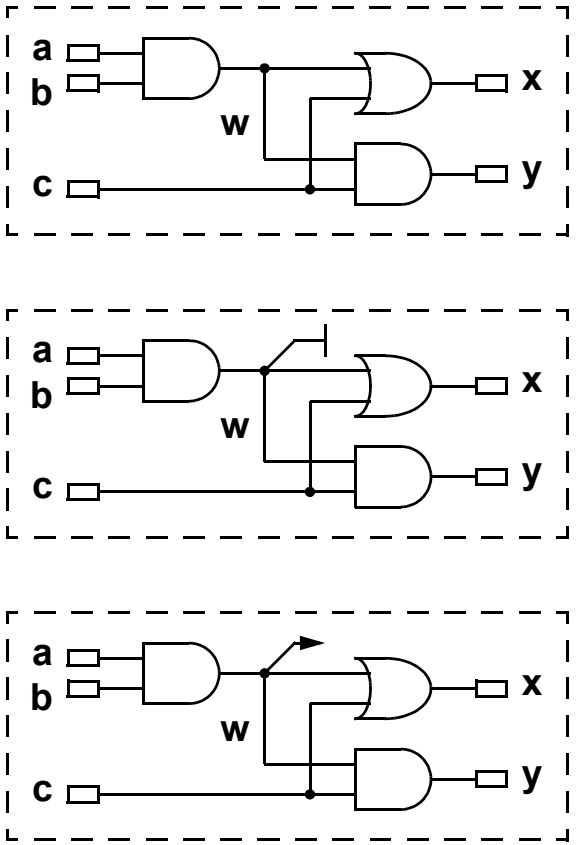
- **Ahel *w* peab olema jälgitav ja juhitav**
  - **Peab leiduma ühisosa jälgitavust ja juhitavust määravate sisend-kombinatsioonide vahel, vastasel korral pole mõni riketest määratav**
  - **Funktsionaalne test – ainult töö õigsuse kontroll, rike ei pruugi olla määratav**
  - **Diagnostika – konkreetse rikke (või isegi mitme rikke) täpne määramine**

- **Konsensus ($C^x_w = x_w \cdot x_{w'}$) – milline osa ei sõltu w-st**
  - $C^x_w = x_w \cdot x_{w'} = 1 \cdot c = c$; $C^y_w = y_w \cdot y_{w'} = c \cdot 0 = 0$;

- ***Häiritus* (perturbation)**
  - Lühis 0-ga - $\delta_{w'} = w \cdot (\partial x/\partial w) = x \oplus x_{w'}$
    - x – $\delta^x_{w'} = w \cdot (\partial x/\partial w) = x \oplus x_{w'} = w \cdot c'$
    - y – $\delta^y_{w'} = w \cdot (\partial y/\partial w) = y \oplus y_{w'} = w \cdot c$
  - Lühis 1-ga - $\delta_w = w' \cdot (\partial x/\partial w) = x' \oplus x_w$
    - x – $\delta^x_{w'} = w' \cdot (\partial x/\partial w) = x' \oplus x_w = w' \cdot c'$
    - y – $\delta^y_{w'} = w' \cdot (\partial y/\partial w) = y' \oplus y_w = w' \cdot c$

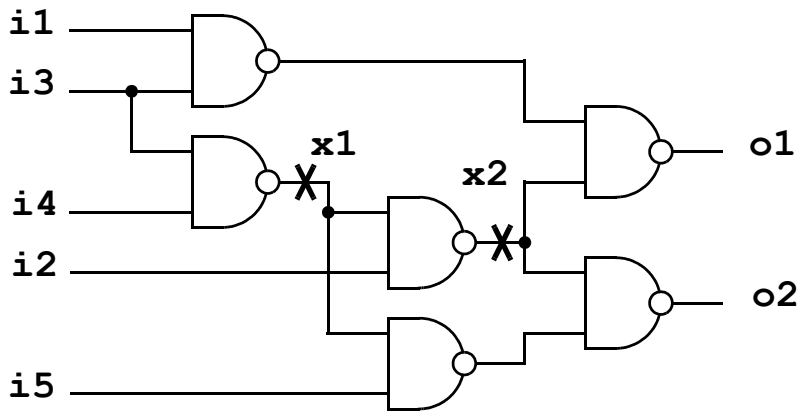
Näide – vektorid

- **Lühis 0-ga**
  - $\delta^{x}_{w'} = w \cdot c'$ & $\delta^{y}_{w'} = w \cdot c$
  - **sisend - abc == 110**
    - **väljund x - 0 (peab olema 1)**
    - **väljund y - 0 (peab olema 0)**
  - **sisend - abc == 111**
    - **väljund x - 1 (peab olema 1)**
    - **väljund y - 0 (peab olema 1)**
- **Lühis 1-ga**
  - $\delta^{x}_{w'} = w' \cdot c'$ & $\delta^{y}_{w'} = w' \cdot c$
  - **sisend - abc == 000**
    - **väljund x - 1 (peab olema 0)**
    - **väljund y - 0 (peab olema 0)**
  - **sisend - abc == 001**
    - **väljund x - 1 (peab olema 1)**
    - **väljund y - 1 (peab olema 0)**



Näide #2

- x1 = (i3 & i4)'
- o1 = ((i1 & i3)' & (i2 & x1)')'
- o2 = ((i5 & x1)' & (i2 & x1)')'



- x2 = (i2 & (i3 & i4)')'
- o1 = ((i1 & i3)' & x2)'
- o2 = (x2 & (i5 & (i3 & i4)')')'

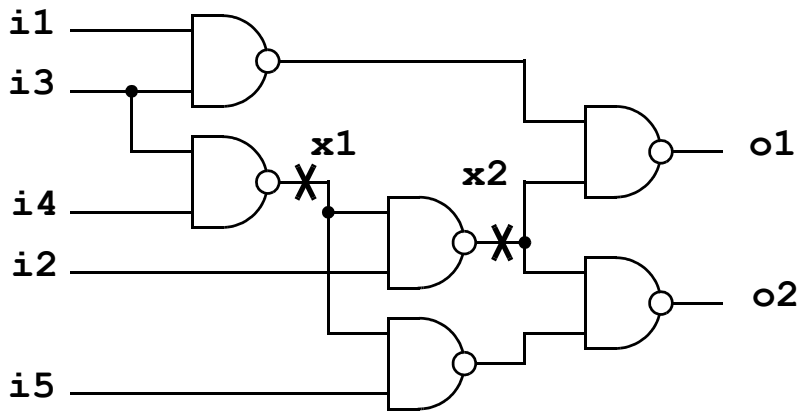
- Lühis 0-ga - $\delta^{x}_{w'} = w \cdot (\partial x/\partial w) = x \oplus x_{w'}$ Lühis 1-ga - $\delta^{x}_{w} = w' \cdot (\partial x/\partial w) = x' \oplus x_{w}$

- $\delta^{o1}_{\overline{x1}} = x1 \cdot (\partial o1/\partial x1)$; $\partial o1/\partial x1 = o1_{x1} \oplus o1_{x1'}$; $o1_{\overline{x1}} = i1\ i3$; $o1_{x1} = i1\ i3 + i2$
- $\partial o1/\partial x1 = (i1\ i3) \oplus (i1\ i3 + i2) = (i1\ i3) (i1\ i3 + i2)' + (i1\ i3)' (i1\ i3 + i2)$
- $\partial o1/\partial x1 = i1\ i3\ i2' (i1' + i3') + (i1' + i3') (i1\ i3 + i2) = i1'\ i2 + i2\ i3'$
- $\delta^{o1}_{\overline{x1}} = x1 \cdot (\partial o1/\partial x1) = (i3\ i4)' (i1'\ i2 + i2\ i3') = (i3' + i4') (i1'\ i2 + i2\ i3') = i2\ i3' + i1'\ i2\ i4'$

Näide #2

- x1 = (i3 & i4)'
- o1 = ((i1 & i3)' & (i2 & x1)')'
- o2 = ((i5 & x1)' & (i2 & x1)')'

- x2 = (i2 & (i3 & i4)')'
- o1 = ((i1 & i3)' & x2)'
- o2 = (x2 & (i5 & (i3 & i4)')')'



- Lühis 0-ga - $\delta^{x}_{w'} = w \cdot (\partial x/\partial w) = x \oplus x_{w'}$ Lühis 1-ga - $\delta^{x}_{w} = w' \cdot (\partial x/\partial w) = x' \oplus x_{w}$

- $\delta^{o1}_{\overline{x1}} = x1 \cdot (\partial o1/\partial x1) = i2\ \overline{i3} + \overline{i1}\ i2\ \overline{i4}$; $\delta^{o1}_{x1} = \overline{x1} \cdot (\partial o1/\partial x1) = \overline{i1}\ i2\ i3\ i4$
- $\delta^{o2}_{\overline{x1}} = x1 \cdot (\partial o2/\partial x1) = i2\ \overline{i3} + i2\ \overline{i4} + \overline{i3}\ i5 + \overline{i4}\ i5$; $\delta^{o2}_{x1} = \overline{x1} \cdot (\partial o2/\partial x1) = i2\ i3\ i4 + i3\ i4\ i5$
- $\delta^{o1}_{\overline{x2}} = x2 \cdot (\partial o1/\partial x2) = \overline{i1}\ \overline{i2} + \overline{i2}\ \overline{i3} + \overline{i1}\ i3\ i4$; $\delta^{o1}_{x2} = \overline{x2} \cdot (\partial o1/\partial x2) = \overline{i1}\ i2\ \overline{i4} + i2\ \overline{i3}\ \overline{i4}$
- $\delta^{o2}_{\overline{x2}} = x2 \cdot (\partial o2/\partial x2) = \overline{i2}\ \overline{i5} + i3\ i4$; $\delta^{o2}_{x2} = \overline{x2} \cdot (\partial o2/\partial x2) = i2\ \overline{i3}\ \overline{i5} + i2\ \overline{i4}\ \overline{i5}$

Näide #2 – vektorid

- $\delta^{o1}_{\overline{x1}} = x1 \cdot (\partial o1/\partial x1) = i2\ \overline{i3} + \overline{i1}\ i2\ \overline{i4}$;
- $\delta^{o2}_{\overline{x1}} = x1 \cdot (\partial o2/\partial x1) = i2\ \overline{i3} + i2\ \overline{i4} + \overline{i3}\ i5 + \overline{i4}\ i5$;
- $\delta^{o1}_{\overline{x2}} = x2 \cdot (\partial o1/\partial x2) = \overline{i1}\ \overline{i2} + \overline{i2}\ \overline{i3} + \overline{i1}\ i3\ i4$;
- $\delta^{o2}_{\overline{x2}} = x2 \cdot (\partial o2/\partial x2) = \overline{i2}\ \overline{i5} + i3\ i4$;

$\delta^{o1}_{x1} = \overline{x1} \cdot (\partial o1/\partial x1) = \overline{i1}\ i2\ i3\ i4$

$\delta^{o2}_{x1} = \overline{x1} \cdot (\partial o2/\partial x1) = i2\ i3\ i4 + i3\ i4\ i5$

$\delta^{o1}_{x2} = \overline{x2} \cdot (\partial o1/\partial x2) = \overline{i1}\ i2\ \overline{i4} + i2\ \overline{i3}\ \overline{i4}$

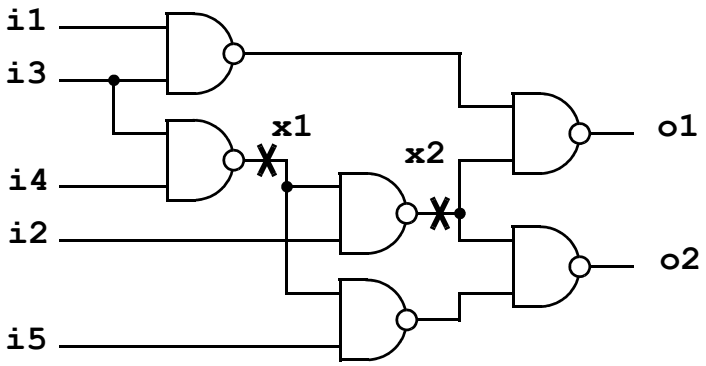
$\delta^{o2}_{x2} = \overline{x2} \cdot (\partial o2/\partial x2) = i2\ \overline{i3}\ \overline{i5} + i2\ \overline{i4}\ \overline{i5}$

i1...5	x1	o1	o2
-10--	1/0	1/0	1/0
01-0-	1/0	1/0	1/0
0111-	0/1	0/1	0/1

i1...5	x1	o1	o2
-10--	1/0	1/0	1/0
-1-0-	1/0	?/?	1/0
--0-1	1/0	?/0	1/0
---01	1/0	?/?	1/0
-111-	0/1	?/1	0/1
--111	0/1	?/?	0/1

i1...5	x2	o1	o2
00---	1/0	0/1	?/1
-00--	1/0	0/1	?/1
0-11-	1/0	0/1	0/1
01-0-	0/1	1/0	1/?
-100-	0/1	1/0	1/?

i1...5	x2	o1	o2
-0--0	1/0	?/1	0/1
--11-	1/0	?/1	0/1
-10-0	0/1	1/0	1/0
-1-00	0/1	1/?	1/0



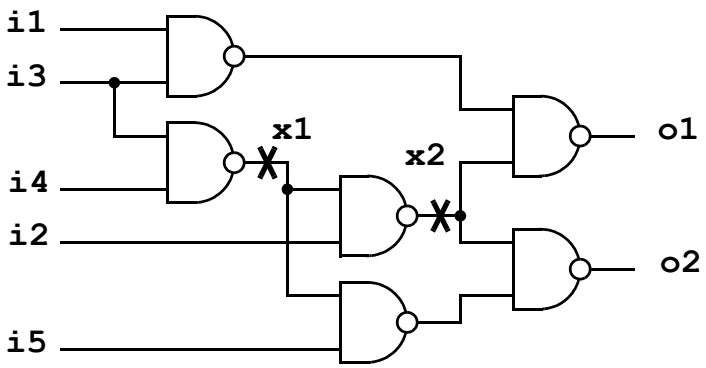
Näide #2 – vektorite pakkimine

i1...5	x1	o1	o2
-10--	1/0	1/0	1/0
01-0-	1/0	1/0	1/0
0111-	0/1	0/1	0/1

i1...5	x1	o1	o2
-10--	1/0	1/0	1/0
-1-0-	1/0	?/?	1/0
--0-1	1/0	?/0	1/0
---01	1/0	?/?	1/0
-111-	0/1	?/1	0/1
--111	0/1	?/?	0/1

i1...5	x2	o1	o2
00---	1/0	0/1	?/1
-00--	1/0	0/1	?/1
0-11-	1/0	0/1	0/1
01-0-	0/1	1/0	1/?
-100-	0/1	1/0	1/?

i1...5	x2	o1	o2
-0--0	1/0	?/1	0/1
--11-	1/0	?/1	0/1
-10-0	0/1	1/0	1/0
-1-00	0/1	1/?	1/0



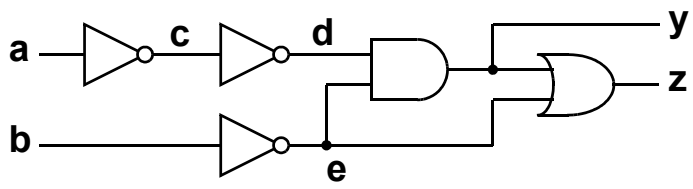
i1...5	0 x1 1	0 x2 1
01000	* .	. *
0111-	. *	* .

- **Testimine – vähim arv (osaliselt) kattuvaid vektoreid, et katta võimalikult palju rikkeid**
  - **Veel üks katte leidmise ülesanne!**
- **Diagnostika – unikaalsed vektorid rikete identifitseerimiseks**
  - **Ei leidu...**

Viite minimeerimine ja testimine

Väär topoloogiline kriitiline tee - näide

- Kõikidel loogikalülidel ühikviide
- Kõik sisendid valmis ajahetkel 0
- Pikim topoloogiline tee
  - (a,c,d,y,z) – viide 4
- Tõeline kriitiline tee
  - (a,c,d,y) – viide 3

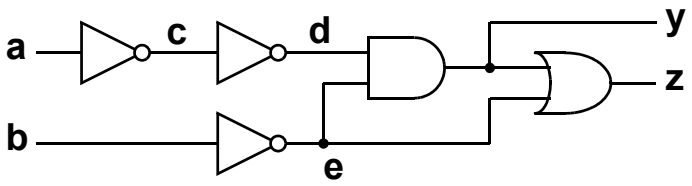


Tundlik kriitiline tee

- Sündmus levib algusest lõpuni
- Mitte-tundlikud kriitilised teed on väärad (ja neid võib ignoreerida)

Dünaamiline tundlikkuse määramine

- Tee – $P = (v_0, v_1, \ldots , v_m)$
- Sündmus levib mööda teed, kui $\partial f_{xi}/\partial x_{i-1} = 1 \quad \forall i=1,2,...m$
  - Kõrvalsisendid (side-inputs) - sisendid, mis pole teel
  - Boole'i diferentsiaal on funktsioon kõrval-sisenditest (mille väärtused võivad muutuda)
  - Boole'i diferentsiaal peab olema tõene ajal, mil sündmus levib
- Tee - (a,c,d,y,z)
  - $\partial f_y/\partial d = e = 1$ ajahetkel 2
  - $\partial f_z/\partial y = e' = 1$ ajahetkel 3
  - ei ole dünaamiliselt tundlik, sest e stabiliseerub ajahetkel 1



- Alternatiiv – staatiline tundlikkuse määramine
  - Lihtsustatud mudel – ajalisi tingimusi Boole'i diferentsiaali väärtuse jaoks ei eksisteeri
  - Oht – liiga väikeste viidete ennustamine

Automaatide testitavuse alused

- Loogikafunktsioonid
  - sisendid seatakse soovitud väärtustele
  - väljundeid võrreldakse eeldatud väärtustega
- Mäluelemendid
  - sisendid seatakse soovitud väärtustele
  - väljundeid võrreldakse eeldatud väärtustega
- Probleem!
  - mäluelementide sisendid/väljundid pole üldjuhul otseselt seatavad/nähtavad
  - kaudne seadmine ja võrdlemine

Seade- ja kontrolljada

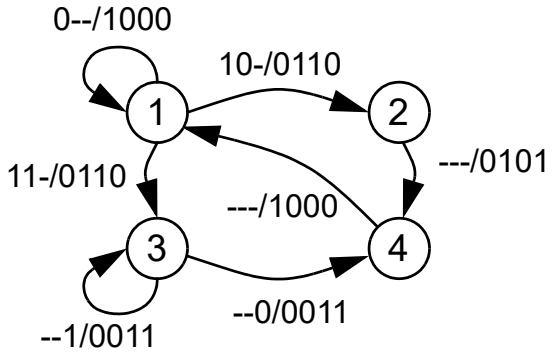
- Automaadi seadejada
  - sisendkombinatsioonide jada, mis viiks soovitud olekusse
    - lähteolekust soovitud olekusse
    - suvalisest olekust soovitud olekusse
  - võimalik lahendus
    - suvalisest olekust lähteolekusse
    - lähteolekust soovitud olekusse

- Automaadi kontrolljada
  - sisendkombinatsioonide jada, mis viitaks üheselt, et automaat oli mingis kindlas olekus (läbis mingit kindlat olekut)
    - eri olekud võivad genereerida sarnaseid väljundsignaale
    - eristatavad väljundkombinatsioonide jadad

Seadejada – näide

- Soovitud olek – 2
  - 1 -> 2 : 1 takt, sisend “10-”
  - 1 -> 1 : 0 takti
  - 2 -> 1 : 2 takti, sisendjada “---”, “---”
  - 3 -> 1 : 2 takti, sisendjada “--0”, “---”
  - 4 -> 1 : 1 takt, sisendjada “---”
- Ühepikkused jadad vajalikud (ootab mõne takti olekus 1)
  - 1 -> 1 : 2 takti, sisendjada “0--”, “0--”
  - 4 -> 1 : 2 takti, sisendjada “---”, “0--”
- Ühepikkuste sisendjadade ühisosa, pluss soovitud olekusse minekuks vajalik jada: “0-0”, “0--”, “10-”
  - 1->1->1->2, 2->4->1->2, 3->4->1->2, 4->1->1->2

I	S^t	S^{t+1}	O
0 - -	1	1	1 0 0 0
1 0 -		2	0 1 1 0
1 1 -		3	0 1 1 0
- - -	2	4	0 1 0 1
- - 0	3	4	0 0 1 1
- - 1		3	0 0 1 1
- - -	4	1	1 0 0 0



Sisse-ehitatud testitavus

- Lisavahendid mäluelementide otseseks seadmiseks
  - nihkeregistrid
  - spetsiaalsed testjadade sisendid ja väljundid
- Kasutusel nii kontroll-osa kui ka andme-osa puhul
- Täiendav riistvara
  - suureneb pindala
  - suureneb viide
  - suureneb voolutarve
  - ka täiendavaid osi tuleks testida
- Ainult osa mäluelemente on otseselt seatavad ja kontrollitavad
- BIST – Built-In Self-Test