Paralelni računarski sistemi

SIMD

SIMD

- Vektorska izračunavanja mogu biti obavljena i pomoću SIMD računara:
  - SIMD računari koriste prostorni paralelizam kod obavljanja vektorskih instrukcija
  - postoji više identičnih procesnih elemenata (PE) koji jednovremeno izvršavaju istu instrukciju nad različitim elementima nekog polja.
- SIMD računar je sinhrono polje PE-ova čijim radom upravlja upravljačka jedinica (CU – control unit)
  - u jednom trenutku svi PE-ovi obavljaju jednu instrukciju nad različitim skupom ulaznih podataka
    - SIMD računarom se postiže paralelizam u obradi podataka, a ne u izvršenju instrukcija
    - SIMD se specijalno projektuju za obavljenje vektorskih izračunavanja – mašine specijalne namene koje se dodaju host računaru kao akceleratori za vektorska izračunavanja

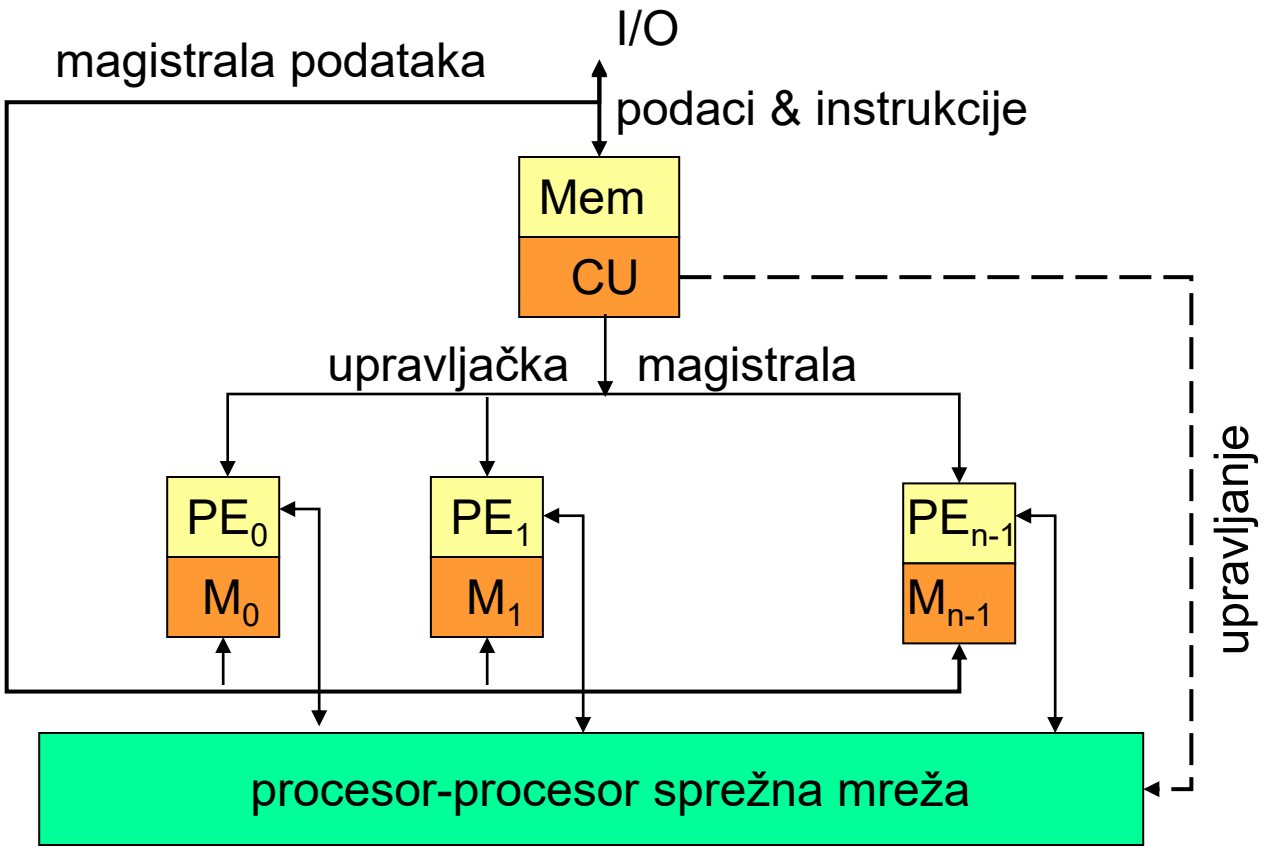
SIMD

- Javljaju se u dve osnovne arhitekturne konfiguracije:
  1. Procesorska polja – koriste RAM memorije
  2. Asocijativni procesori – koriste sadržajno adresibilne (asocijativne) memorije.

Procesorska polja

- Dve varijante:
  - sa distribuiranom memorijom
  - sa zajedničkom (deljivom) memorijom

Procesorska polja sa distribuiranom memorijom



Procesorska polja sa distribuiranom memorijom

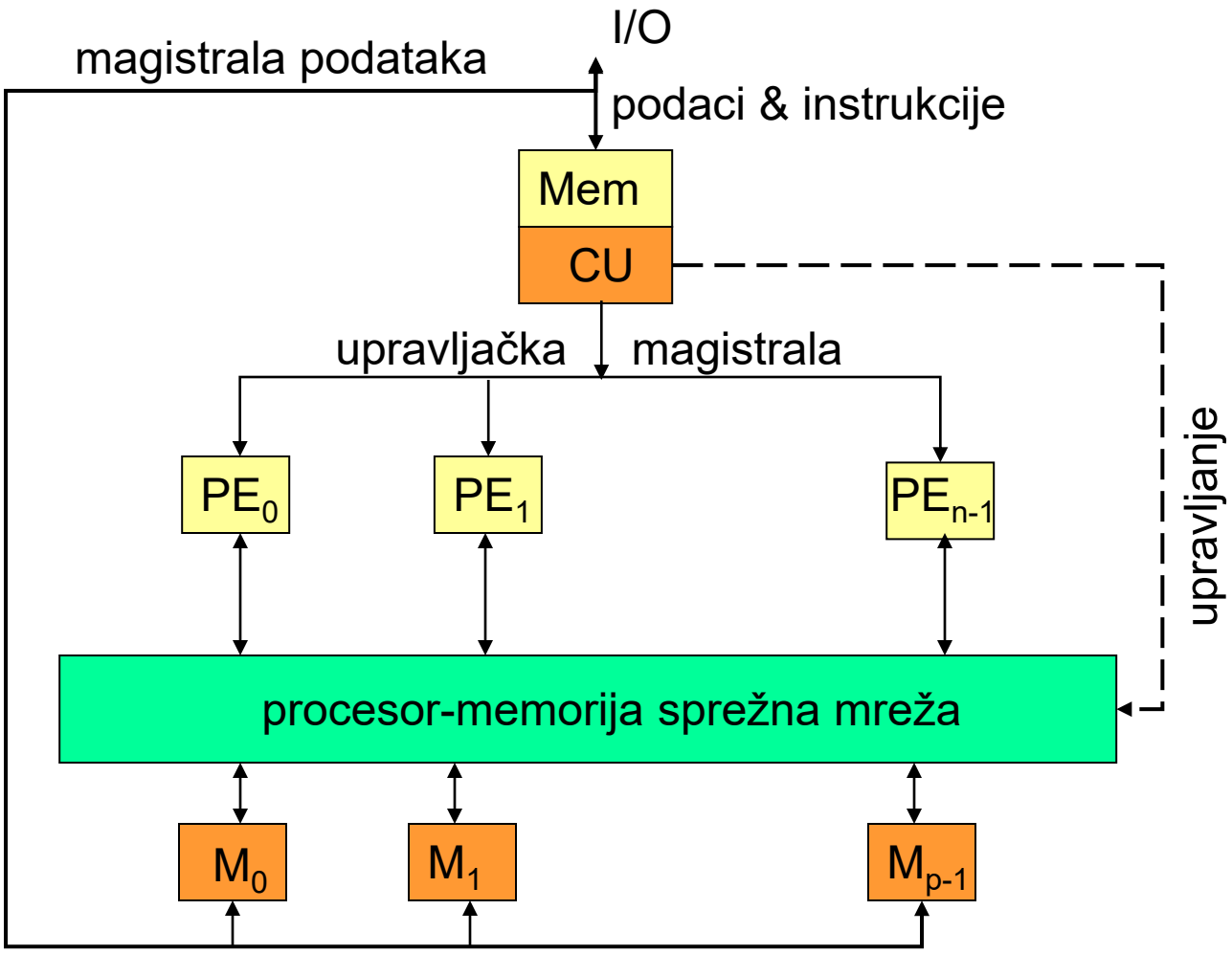
- $n=2^m$ sinhronih PE-ova čijim radom upravlja CU
  - PE je aritmetičko-logička jedinica sa pridruženim radnim registrima i lokalnom memorijom za smeštanje distribuiranih podataka
  - memorija CU služi za pamćenje programa i skalarnih podataka
    - sistemski i korisnički programi se izvršavaju pod kontrolom CU
    - korisnički programi se pune u CU iz hosta
    - CU dekodira instrukcije i odredjuje gde se izvršavaju:
      - skalarne instrukcije i instrukcije upravljačkog tipa izvršavaju se u CU
      - vektorske instrukcije se distribuiraju svim PE-ovima preko upravljačke magistrale
      - vektorski operandi se distribuiraju u lokalne memorije PE-ova preko magistrale za podatke

Procesorska polja sa distribuiranom memorijom

- Za upravljanje statusom PE koristi se maskiranje (pasiva/aktivan)
- Razmena podataka izmedju PE-ova ostvaruje se preko sprežne mreže
  - radom sprežne mreže upravlja CU preko instrukcija za rutiranje

* Procesorsko polje je spregnuto sa host računarom preko CU.
  - Host upravlja radom celog sistema
  - F-ja hosta:
    - U/I aktivnosti
    - kompajlirannje programa
    - loadovanje programa u CU
    - upravljanje resursima

Procesorska polja sa deljivom memorijom



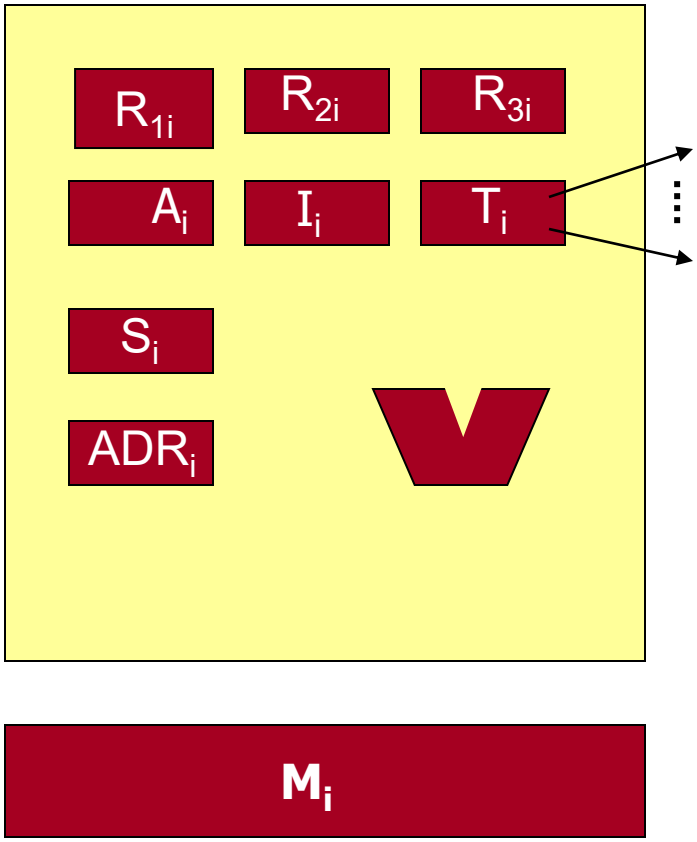
Procesorska polja sa deljivom memorijom

- Postoji $n=2^m$ PE-ova i p memorijskih modula (obično su n i p uzajamno prosti)
- Lokalne memorije su zamenjene paralelnim memorijskim modulima kojima mogu pristupati svi PE-ovi preko sprežne mreže
  - f-ja sprežne mreže je ostvarivanje veze izmedju PE i memorijskog modula za što veći broj parova procesor-memorija bez konflikata.

Procesorska polja

- Većina procesorskih polja je sa distribuiranom memorijom
  - ILLIAC IV – 64 PE
  - MasPar MP-1 od 1024 do 16384 PE-ova (32 PE na čipu)
  - Connection Machine CM-2 64K 1-bitnih PE-ova (16 PE na čipu)
  - DAP610 4K PE (64 PE na čipu)
  - BSP – 16 PE i 17 memorijskih modula (deljiva memorija)

Izgled PE sa distribuiranom memorijom



- R_{1i}, R_{2i}, R_{3i} – radni registri
- A_i – adresni registar
- I_i – indeksni registar
  - (za adresiranje lokalne memorije zajedno sa A_i)
- T_i – registar za komunikaciju spregnut sa drugim T registrima preko sprežne mreže
- S_i – statusni registar (0 Pe pasivan/ 1 PE aktivan)
- ADR_i – m-bitna adresa PE-a

Maskiranje PE

- Svaka maska deli skup PE-ova na
  - podskup aktivnih
  - podskup pasivnih
- Nekim načinima maskiranja moguće je maskirati proizvoljne podskupove PE.
  - Direktno maskiranje (ILLIAC IV)
    - omogućava maskiranje proizvoljnog podskupa
    - sadržaj registra maske odredjuje kompilator nakon prevodjenja svake instrukcije
    - CU emituje sadržaj vektora maske zajedno sa svakom instrukcijom

Maskiranje PE

- maskiranje pomoću adrese PE
  - $n=2^m$ PE, m-bitna maska
  - svaka pozicija u maski može imati vrednost 0,1,X
  - aktivni su oni PE čija se adresa poklapa sa maskom
  - PRIMER: n=8 PE, maska 1X0 ⟹ 110 i 100, aktivni
  - nije moguće maskirati proizvoljni podskup PE

✱ Maskiranje podacima – dinamičko maskiranje

- maske predstavljaju implicitni rezultat naredbe uslovnog grananja u zavisnosti od podatka koji je lokalni za PE
- Kada se naredba uslovnog grananja emituje od strane CU, svaki PE je obavlja nad različitim podacima, pa rezultat može biti različit u svakom PE.
  - ➢ maske se dobijaju naredbom WHERE oblika
  - ➢ where (uslov) do niz_nar_A elsewhere niz_naredbi_B
  - ➢ svaki PE postavlja svoj interni flag na 0 ili 1
    - emituje se niz_naredbi_A
    - komplementira se flag
    - emituje se niz_naredbi_B

✱ maskiranje po podacima se može koristiti u kombinaciji sa drugim načinima maskiranja

Primer

- Naći sve parcijalne sume S(k) prvih k komponenti vektora $V=[V_i]_{nx1}$, za svako k=0,1,...,n-1.

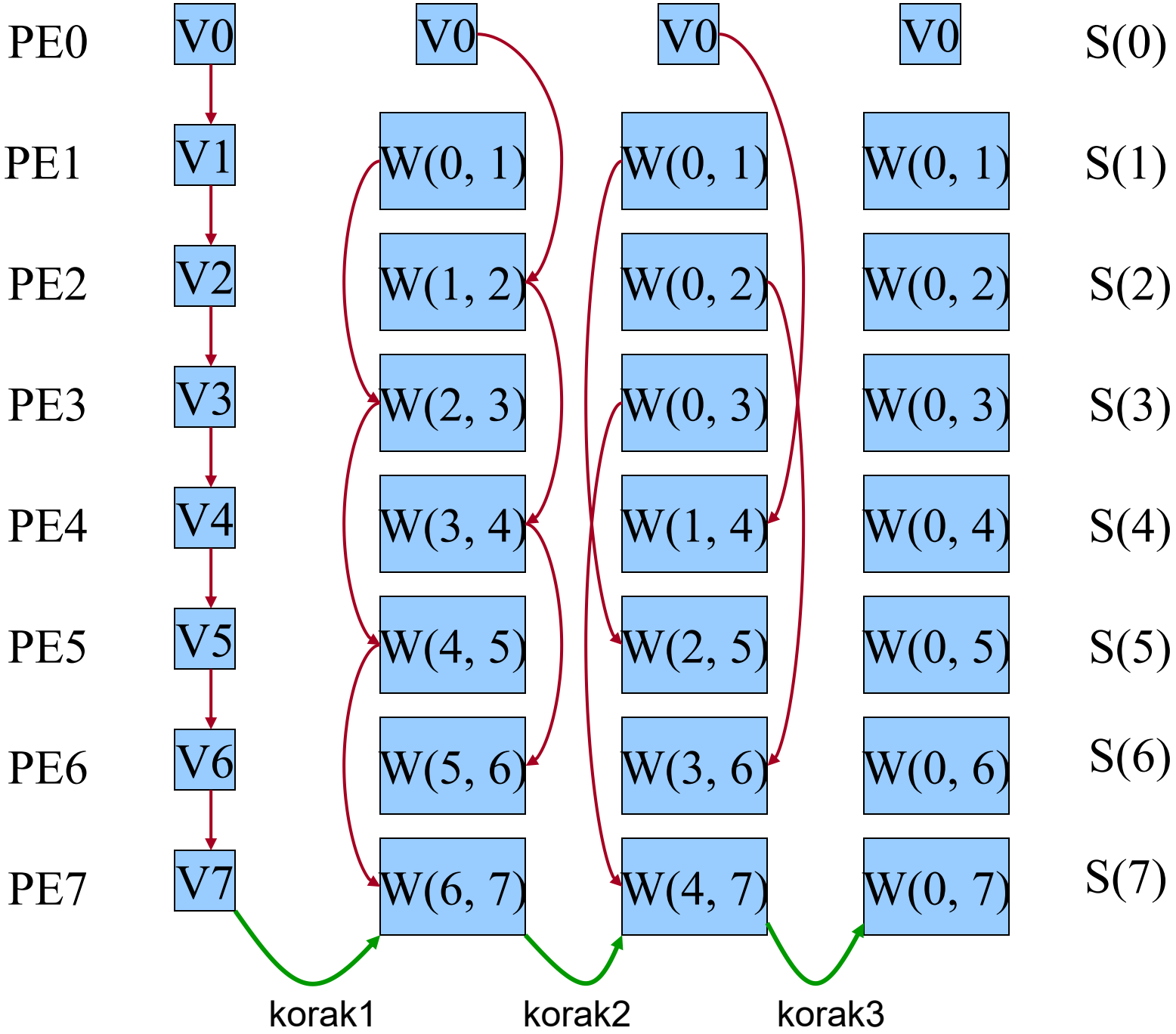
$$S(k) = \sum_{i=0}^{k} V_i, \quad k = 0,1,....,n-1$$

- Izračunavanje je moguće obaviti u $\log_2 n$ koraka ako postoji n PE-ova
  - Jedan korak podrazumeva razmenu podataka i izračunavanje (sabiranje)

Primer – nast.

- n=8, naći S(0),...., S(7)
- Inicijalno u lok. memoriji svakog PE se nalazi po jedan element vektora

1. Korak: $PE_i \rightarrow PE_{i+1}$, za i=0,1,...,6 (PE7 maskiran)
   $W_i := V_i + V_{i+1}$ (PE0 maskiran)
2. Korak: $PE_i \rightarrow PE_{i+2}$ za i=0, 1,...,5 (PE6, PE7 maskirani)
   $W_i := W_i + W_{i-2}$ za i=2,...,7 (PE0, PE1 maskirani)
3. Korak: $PE_i \rightarrow PE_{i+4}$, za i=0,1,2,3 (PE4 – PE7 maskirani) Vi
   $W_i := W_i + W_{i-4}$ za i=4,...,7 (PE0- PE3 maskirani)


PE0
PE1
PE2
PE3
PE4
PE5
PE6
PE7
V0
V1
V2
V3
V4
V5
V6
V7
V0
W(0, 1)
W(1, 2)
W(2, 3)
W(3, 4)
W(4, 5)
W(5, 6)
W(6, 7)
V0
W(0, 1)
W(0, 2)
W(0, 3)
W(1, 4)
W(2, 5)
W(3, 6)
W(4, 7)
V0
W(0, 1)
W(0, 2)
W(0, 3)
W(0, 4)
W(0, 5)
W(0, 6)
W(0, 7)
S(0)
S(1)
S(2)
S(3)
S(4)
S(5)
S(6)
S(7)
korak1
korak2
korak3

SIMD

Sprežne mreže

Sprežne mreže

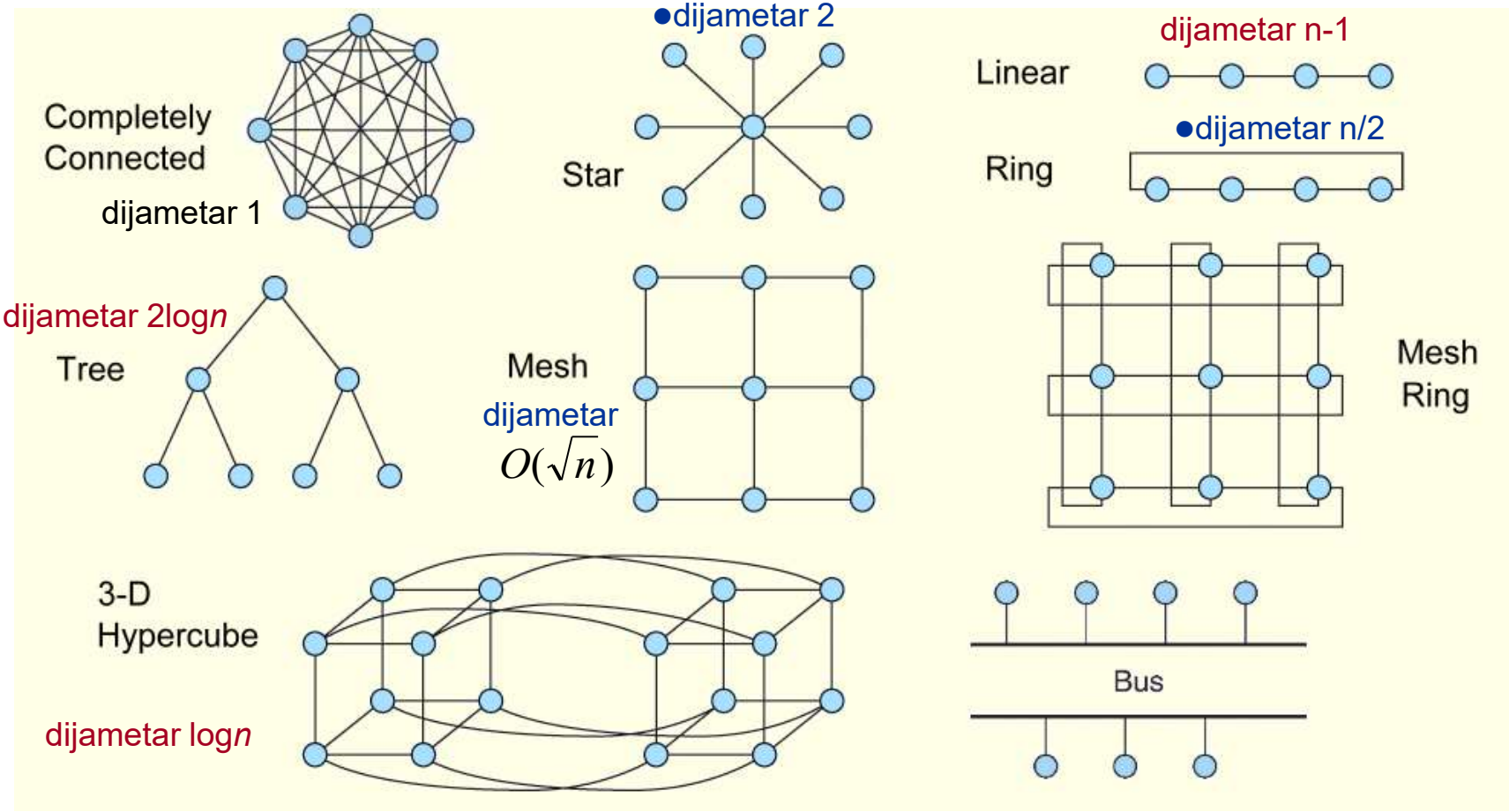
- Služe za povezivanje
  - PE i PE
  - PE i memorijskih modula
- Statičke SM – veze izmednju PE su fiksne i ne mogu se menjati u toku izvršenja programa
- Dinamičke SM – veze izmedju PE-ova nisu striktno dodeljene, već se mogu menjati u toku rada postavljenjem aktivnih mrežnih komponenti (komutacioni elementi)
  - jednostepene
  - višestepene

Parametri SM

- Dijametar SM – maksimalno minimalno rastojanje izmedju dva čvora u mreži
- proširljivost – dali se čvorovi mogu dodavati?
- Redundantnost – koloko puteva postoji izmedju svakog para čvorova
- rutiranje – koliko je komplikovano
- propusnost – koliko podataka u jedinici vremena
- latentnost – koliko je vremena potrebno da se podatak prenese od čvora do čvora

Statičke SM

- Jednodimenzionalne (linearne)
- dvodimenzionalne
- trodimenzionalne



Statičke SM

- Linearne
  - dijametar n-1
- Ring
  - dijametar n/2
- Zvezda (star)
  - dijametar 2
- Rešetka (mesh)
  - dijametar $O(\sqrt{n})$
- Stablo
  - dijametar 2log*n*
- potpuno povezana
  - dijametar 1
- Hiperkub
  - dijametar log*n*

Statičke mreže

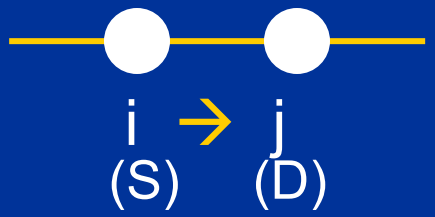
Linearno polje (1D polje)

Br. čvorova = N

Dijametar =N-1

Rutiranje - samorutirajuće

U procesu komunikacije svaki čvor zna kome da pošalje primljenu poruku



čvor i poredi svoj ID sa odredišnim ID

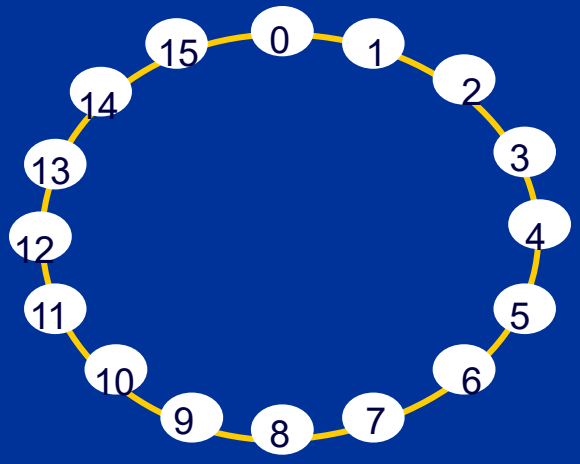
if dest.ID >, pošalji →
if dest ID <, pošalji ←

Statičke mreže

Ring

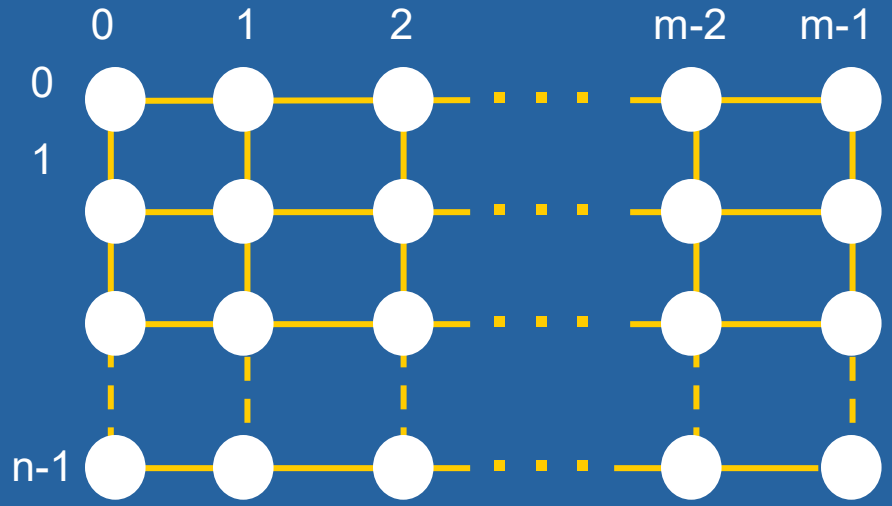
dobija se povezivanjem krajnjih čvorova u 1D polju

Br čvorova = N, dijametar = n/2



Statičke mreže

Mesh - rešetka (2D polja)



Br.čvorova N = m x n

Dijameter = m-1+n-1

Stepeni čvorova = 2 (na uglovima)
nodes) 3 (bočni)
4 (središnji)

Rutiranje

Izvor poredi prvo broj vrste svog i odredišnog čvora

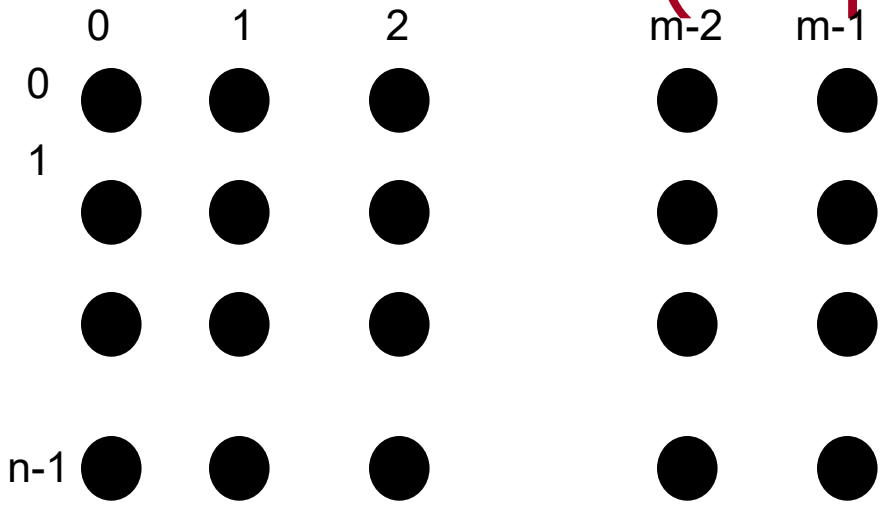
if Dest vrsta <, pošalji ↑
if Dest vrsta >, pošalji ↓

Zatim poredi redne brojeve kolona

if Dest kol >, pošalji →
if Dest kol <, pošalji ←

Statičke mreže

(2D polja)



Br.čvorova N = m x n

Dijameter = m-1+n-1

Stepeni čvorova (na uglovima) = 2
nodes) 3 (bočni)
4 (središnji)

Izvor poredi prvo broj vrste svog i odredišnog čvora

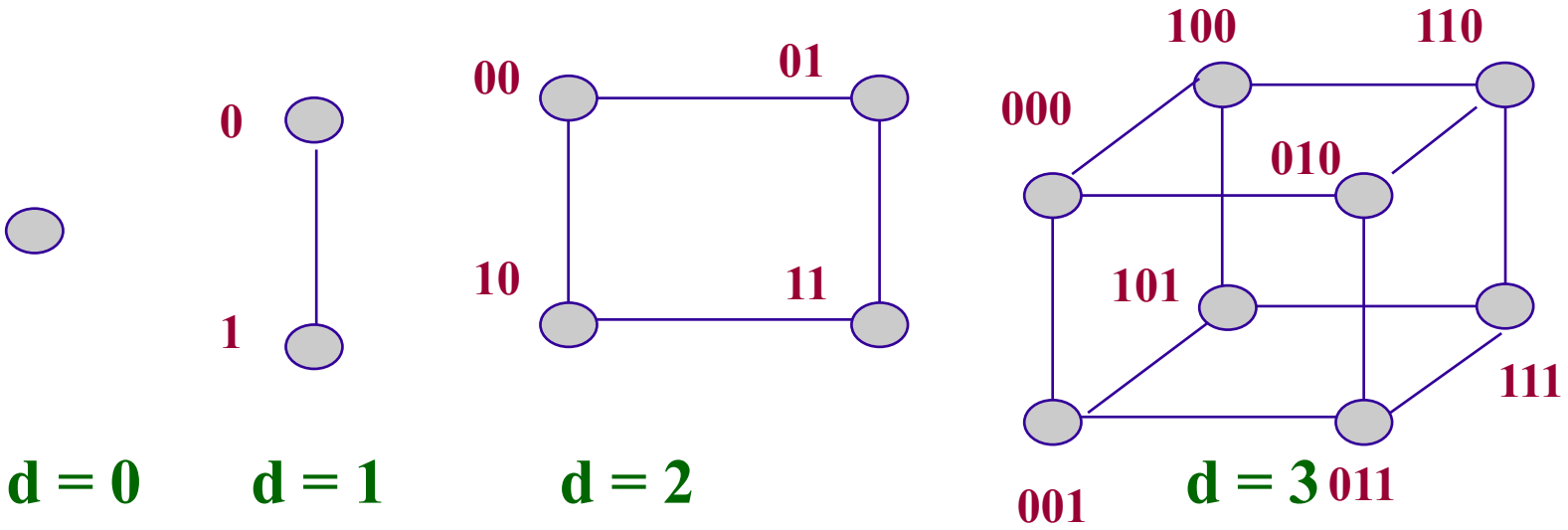
if Dest vrsta <, pošalji
if Dest vrsta >, pošalji

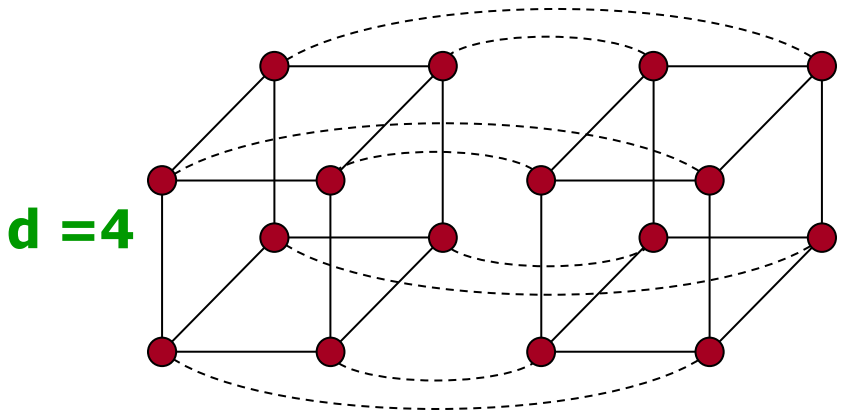
Zatim poredi redne brojeve kolona

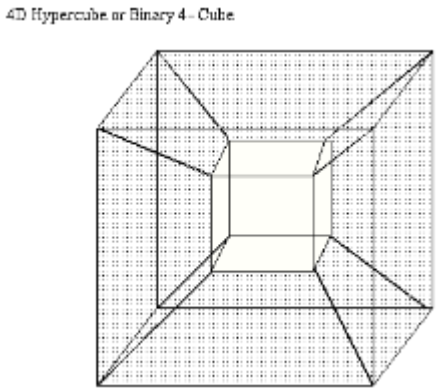
if Dest kol >, pošalji
if Dest kol <, pošalji

Hiperkub

- broj čvorova stepen 2
- Adrese čvorova 0, 1, …, 2^k-1
- Čvor *i* povezan direktno sa *k* čvorova čije se adrese razlikuju samo na jednoj bitskoj poziciji.







Rutiranje u hiperkubu

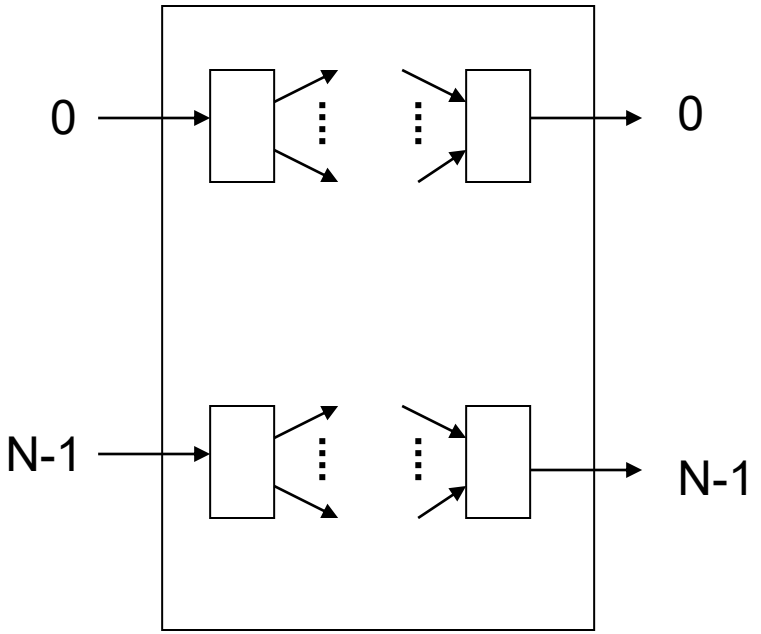
Algoritam:

- Ne ka je s(i) adresa izvora, d(i) adresa odredišta i $\sigma(i)$ adresa čvora koji je na putu od s(i) do d(i)
- Uporediti bitove adresa $d(i)$ sa $\sigma(i)$ s leva u desno
- Identifikovati prvu bit poziciju na kojoj se ove dve adrese razlikuju
- Proslediti paket susedu n(i) tako da se adrese n(i) i $\sigma(i)$ razlikuju samo na uočenoj bitskoj poziciji.

  - Npr. s(i)=(1,0,1,0), d(i)=(0,1,0,1) ide preko čvorova
  - (1,0,1,0) –> (0,0,1,0) –> (0,1,1,0) -> (0,1,0,0) -> (0,1,0,1)

Dinamičke SM - jednostepene

- Sastoji se od N ulaznih i N izlaznih jedinica (N je broj PE)
  - ulazne jedinice DMUX 1xD
  - izlazne jedinice MUX Mx1
    - 1<=D,M<=N, za D=M=N crossbar SM
  - recirkularne –podaci mogu da kruže više puta kroz mrežu da stignu od izvora do odredišta
    - broj kruženja zavisi od povezanosti (što je veće D i M veća je povezanost)



Sprežne funkcije

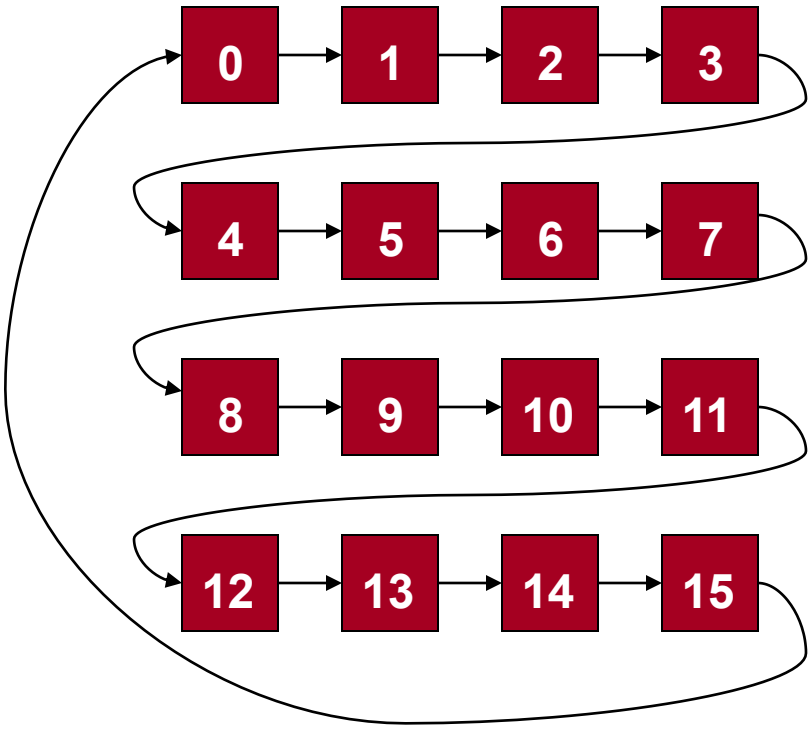
- Dinamičke SM se definišu skupom sprežnih funkcija
  - Sprežna f-ja Sf preslikava adresu X jednog PE u adrseu Sf(X) drugog PE sa kojim se ostvaruje povezivanje
    - kada se speržna f-ja primeni na adresu jedne komponente sistema, dobija se adresa druge komponente koja može direktno primiti podatak
    - Da bi se preneo podatak od jednog PE do drugog PE mora se izvršiti programirana sekvenca jedne ili više sprežnih f-ja
    - Kod SIMD sistema svi aktivni PE-ovi moraju izvršavati istu sprežnu f-ju u datom trenutku
    - Sprežne f-je su deo SIMD programa

Primeri SM: rešetka (ILLIAC IV)

- Definisana sa 4 sprežne f-je:
  - $M_{+1}(x)=(x+1) \bmod N$
  - $M_{-1}(x)=(x-1) \bmod N$
  - $M_{+r}(x)=(x+r) \bmod N$
  - $M_{-r}(x)=(x-r) \bmod N$, $r = \sqrt{N}$
  - za N=64 dobija se ILLIAC IV SM

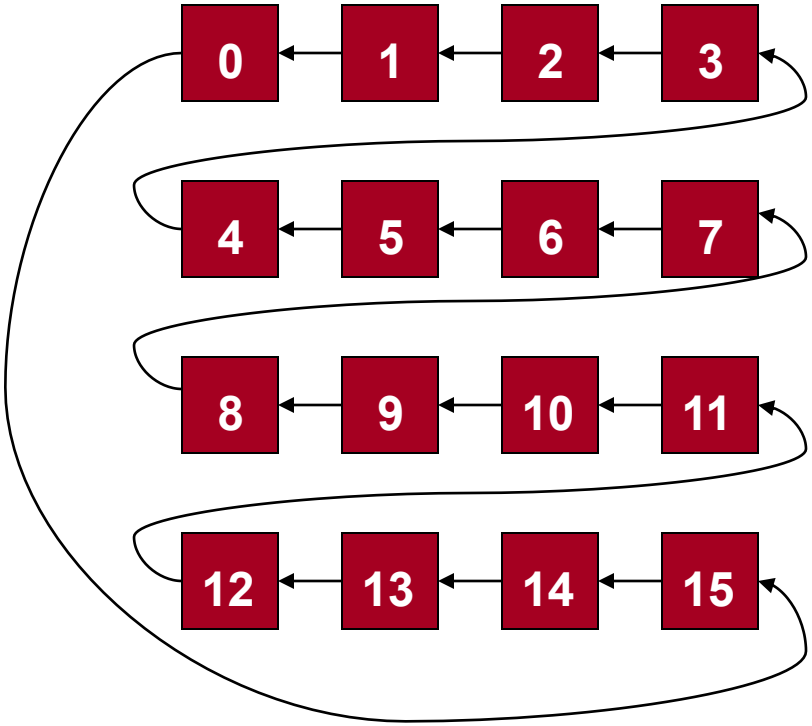
Primer: rešetka za N=16

- $M_{+1}(x)=(x+1) \bmod 16$



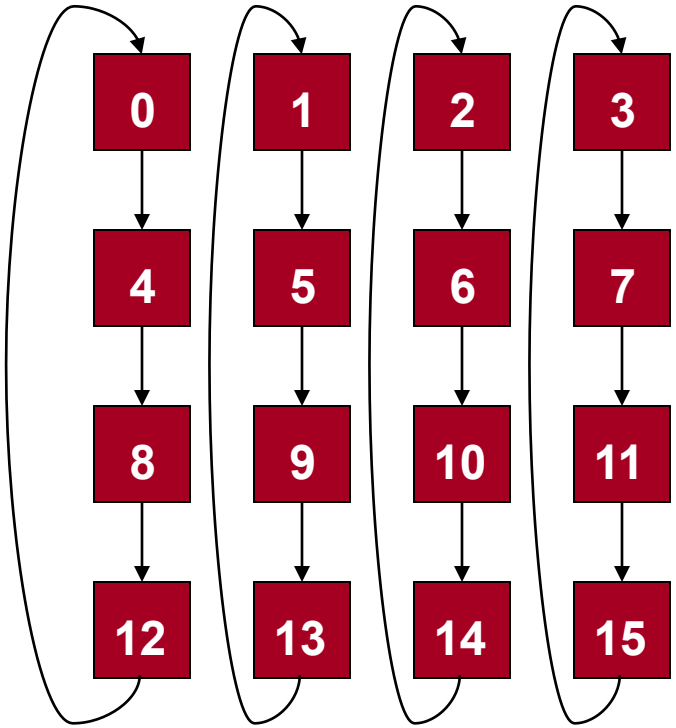
Primer: rešetka za N=16 (nast.)

- $M_{-1}(x)=(x-1) \bmod 16$



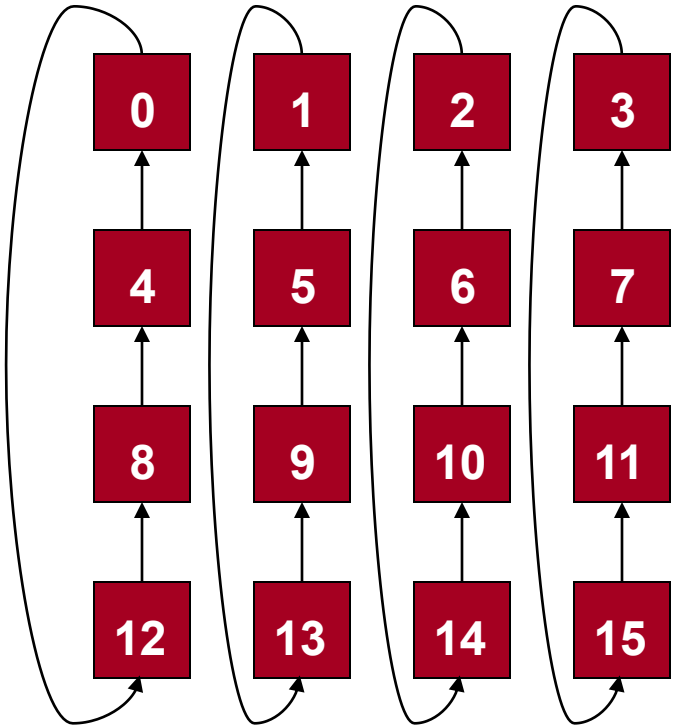
Primer: rešetka za N=16 (nast.)

- $M_{+4}(x)=(x+4)\bmod 16$,



Primer: rešetka za N=16 (nast.)

- $M_{-4}(x)=(x-4) \bmod 16$,



SM mešanje zamena (shuffle-excange)

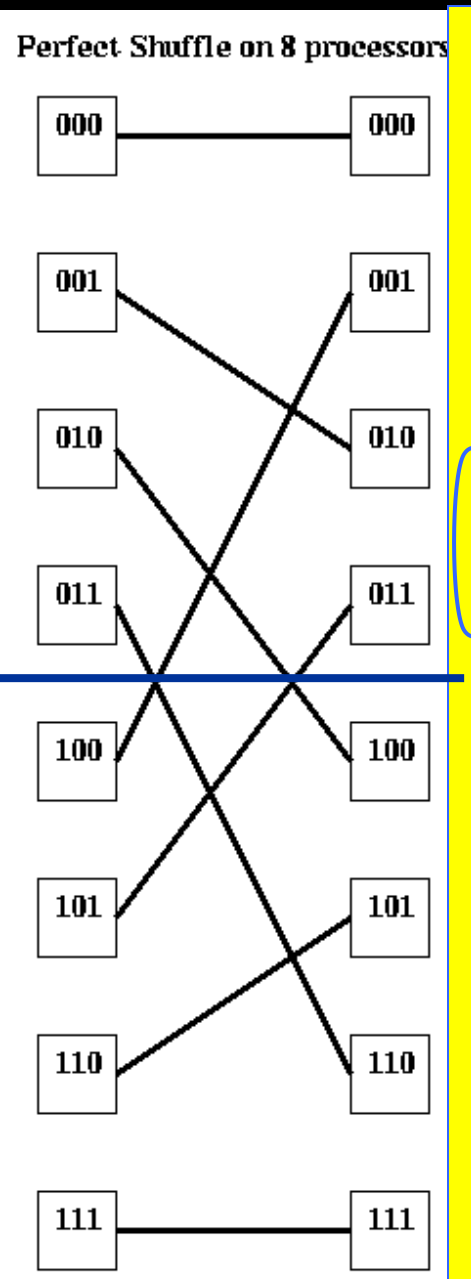
- F-ja mešanja odgovara savršenom mešanju špila karata

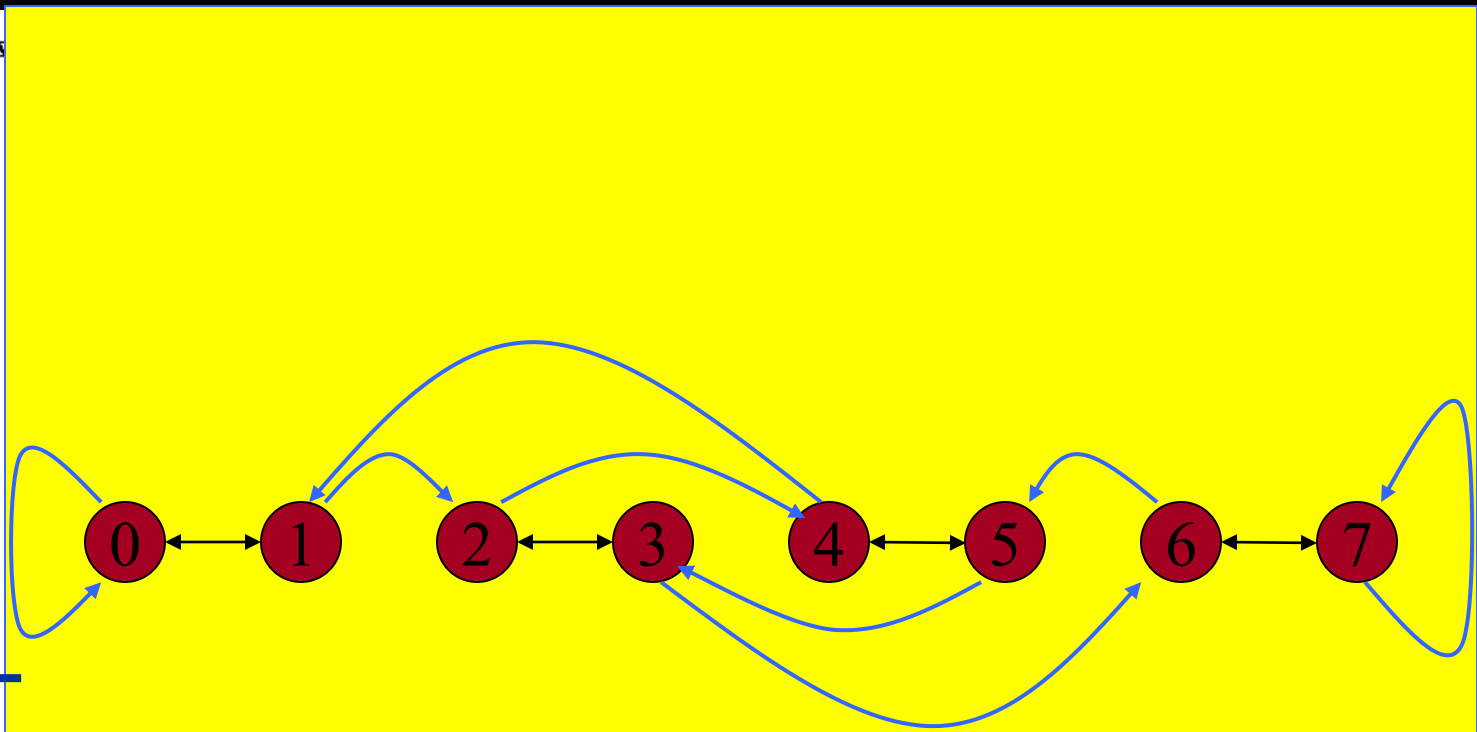
$$S(b_{m-1}, b_{m-2}, \ldots, b_1, b_0) = b_{m-2}, \ldots, b_1, b_0, b_{m-1}$$

- F-ja zamene

$$E(b_{m-1}, b_{m-2}, \ldots, b_1, b_0) = b_{m-1}, b_{m-2}, \ldots, b_1, \bar{b}_0$$

Mešanje-zamena primer





Primer – izračunavanje vrednosti polinima

$$P(n) = \sum_{i=0}^{n} a_i x^i$$

- Izračunavanje je moguće obaviti procesorskim poljem sa N=n+1 PE spregnutim SM mešanje-zamena u 2log*N* koraka:
  - dve faze:
  1. svaki PE računa $a_i x^i$
  2. svi PE vrše sumiranje dobijenih rezultata i dobijaju P(x)
     - posmatraju se sadržaji 3 registra: registar koeficijenata, Ai, registar promenljive x, registar maske, Mi

- I faza (množenje)

```
for j:=1 to logN
for i:=0 to N-1 do inparallel
if Mi=1 then
Ai:=Ai*x
endif;
x:=x*x;
shuffle(Mi);
endfor;
endfor;
```

Mi je registar maske (Mi=1 znači da je PE aktivan)

- II faza (sabiranje)

```
for j:=1 to logN
for i:=0 to N-1 do inparallel
Ti:=Ai;
shuffle(Ti);
Ai:=Ti;
exchange(Ti);
Ai:=Ai+Ti;
enfor;
endfor;
```

Ti je transportni registar čiji se sadržaj razmenjuje u komunikaciji

I faza - množenje

inicijalno

nakon I koraka

nakon II koraka

nakon III koraka

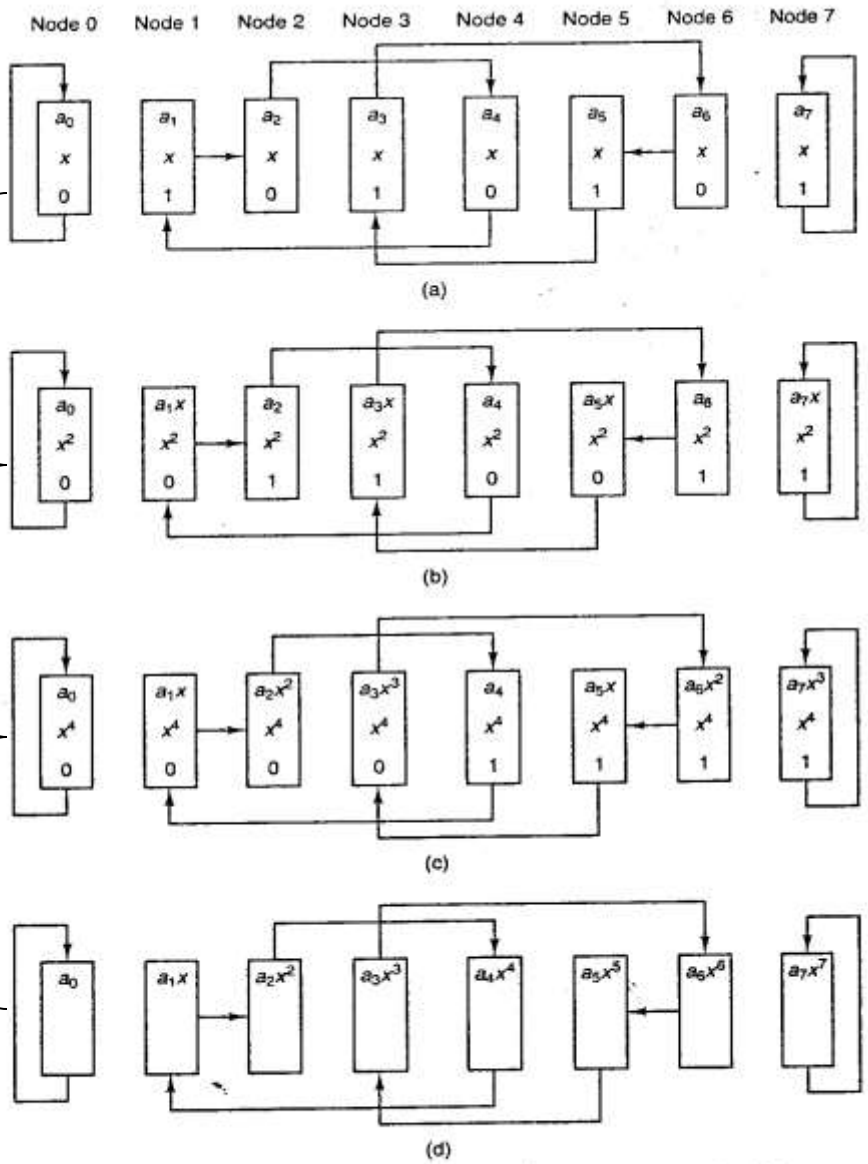


Figure 5.9 Steps for the computation of the $a_i x^i$. (a) Initial values. (b) Values after step 1. (c) Values after step 2. (d) Values after step 3.

II faza - sabiranje

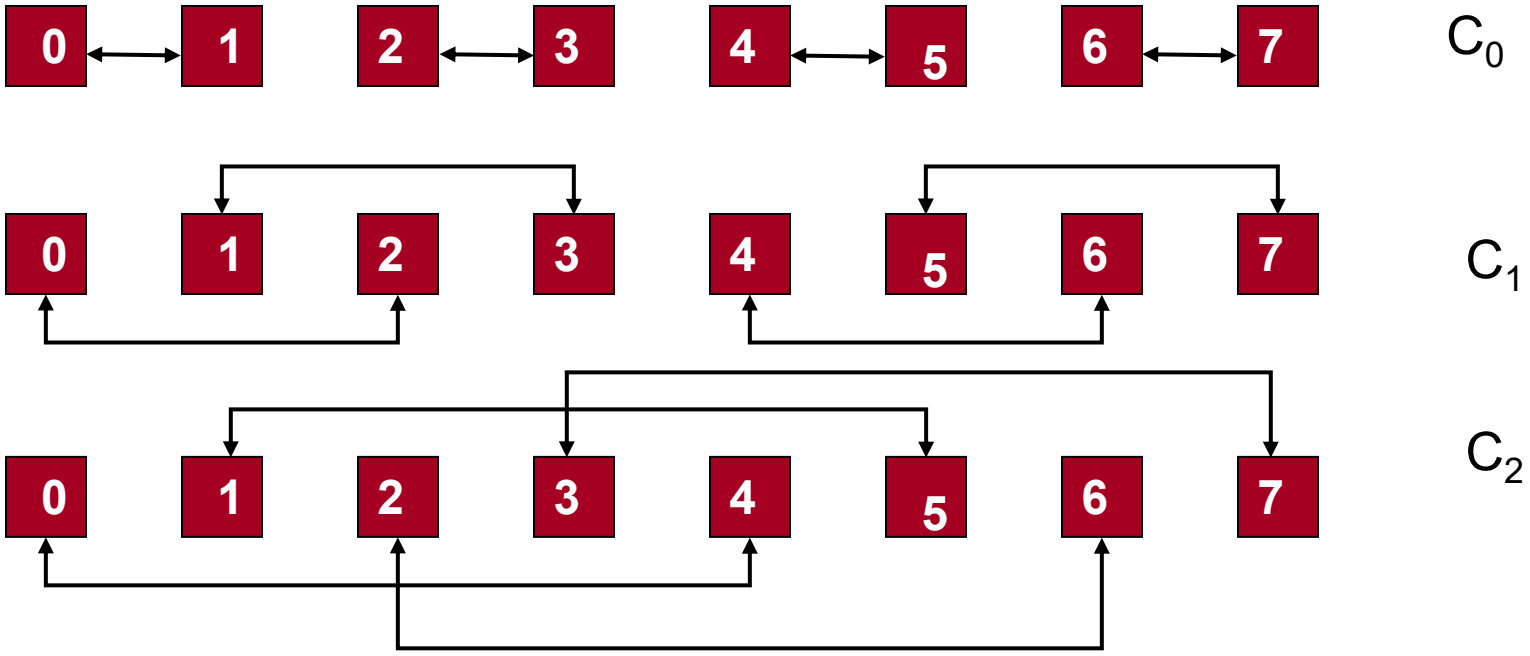
	Node 0	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7
Initial values	a_0	a_1x	a_2x^2	a_3x^3	a_4x^4	a_5x^5	a_6x^6	a_7x^7
Step 1 Shuffle	a_0	a_4x^4	a_1x	a_5x^5	a_2x^2	a_6x^6	a_3x^3	a_7x^7
Exchange	$a_0 + a_4x^4$	$a_0 + a_4x^4$	$a_1x + a_5x^5$	$a_1x + a_5x^5$	$a_2x^2 + a_6x^6$	$a_2x^2 + a_6x^6$	$a_3x^3 + a_7x^7$	$a_3x^3 + a_7x^7$
Step 2. Shuffle	$a_0 + a_4x^4$	$a_2x^2 + a_6x^6$	$a_0 + a_4x^4$	$a_2x^2 + a_6x^6$	$a_1x + a_5x^5$	$a_3x^3 + a_7x^7$	$a_1x + a_5x^5$	$a_3x^3 + a_7x^7$
	Node 0		Node 1		Node 2		Node 3	
Exchange	$a_0 + a_4x^4 + a_2x^2 + a_6x^6$		$a_0 + a_4x^4 + a_2x^2 + a_6x^6$		$a_0 + a_4x^4 + a_2x^2 + a_6x^6$		$a_0 + a_4x^4 + a_2x^2 + a_6x^6$	
	Node 4		Node 5		Node 6		Node 7	
Step 3.	$a_1x + a_5x^5 + a_3x^3 + a_7x^7$		$a_1x + a_5x^5 + a_3x^3 + a_7x^7$		$a_1x + a_5x^5 + a_3x^3 + a_7x^7$		$a_1x + a_5x^5 + a_3x^3 + a_7x^7$	
	Node 0		Node 1		Node 2		Node 3	
Shuffle.	$a_0 + a_4x^4 + a_2x^2 + a_6x^6$		$a_1x + a_5x^5 + a_3x^3 + a_7x^7$		$a_0 + a_4x^4 + a_2x^2 + a_6x^6$		$a_1x + a_5x^5 + a_3x^3 + a_7x^7$	
	Node 4		Node 5		Node 6		Node 7	
	$a_0 + a_4x^4 + a_2x^2 + a_6x^6$		$a_1x + a_5x^5 + a_3x^3 + a_7x^7$		$a_0 + a_4x^4 + a_2x^2 + a_6x^6$		$a_1x + a_5x^5 + a_3x^3 + a_7x^7$	
	Node 0				Node 1			
Exchange	$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$				$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$			
	Node 2				Node 3			
	$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$				$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$			
	Node 4				Node 5			
	$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$				$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$			
	Node 6				Node 7			
	$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$				$a_0 + a_4x^4 + a_2x^2 + a_6x^6 + a_1x + a_5x^5 + a_3x^3 + a_7x^7$			

values for k_1 and k_0; rather, the total number of nodes is defined. A two-dimensional mesh with $k_1 = k_0 = n$ is usually referred to as a *mesh* with N nodes, where $N = n^2$. For example, Figure 5.12 shows a mesh with 16 nodes. From this point forward, the term

Kub mreža

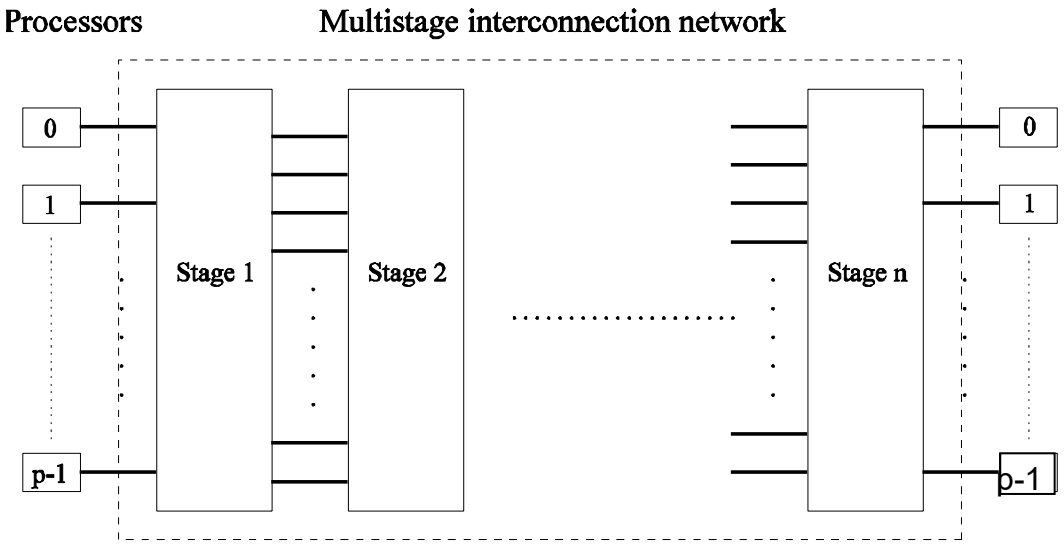
- Definisana sa $m=\log_2 N$ sprežnih f-ja

$$C_i(b_{m-1},\ldots,b_i,\ldots b_1,b_0) = b_{m-1},\ldots,\bar{b}_i,\ldots b_1,b_0,$$
$$i = 0,1,\ldots,m-1$$



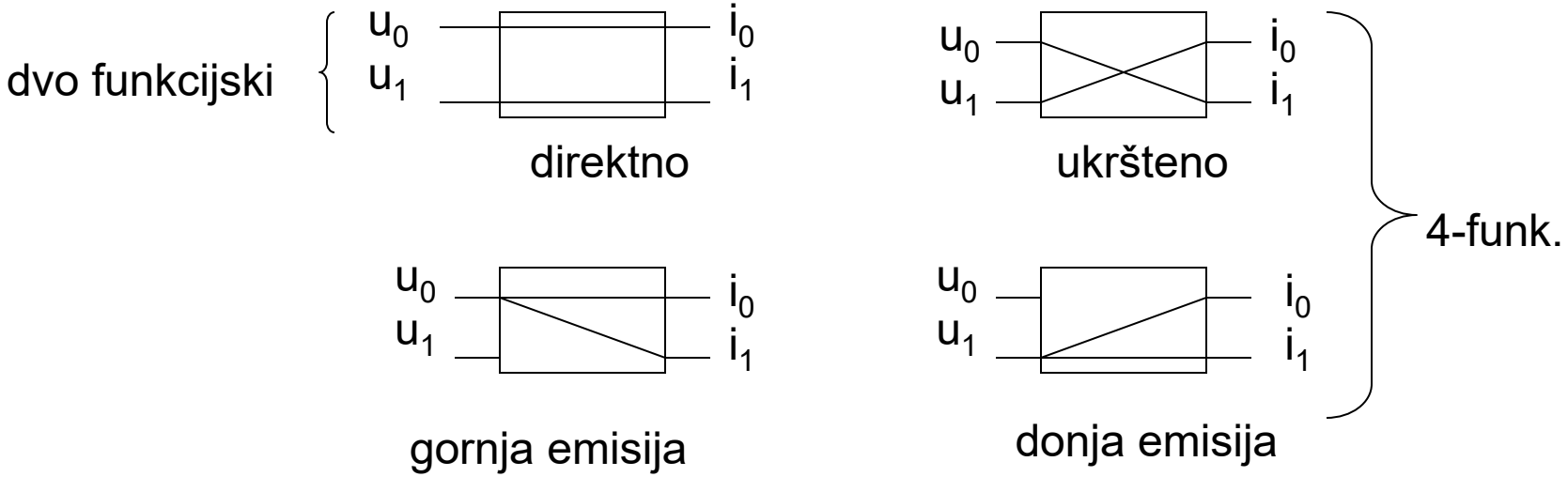
Višestepene SM

- Sastoje se od komutacionih elemenata (KE) i komunikacionih kanala
- KE čine stepene mreže
- Veze izmedju stepena su fiksne
- KE se mogu dinamički postavljati da bi se ostvarila veza izmedju željenog ulaza i izlaza
- Višestepenu SM karakteriše
  - izgled KE
  - način povezivanja susednih stepena
  - način upravljanja

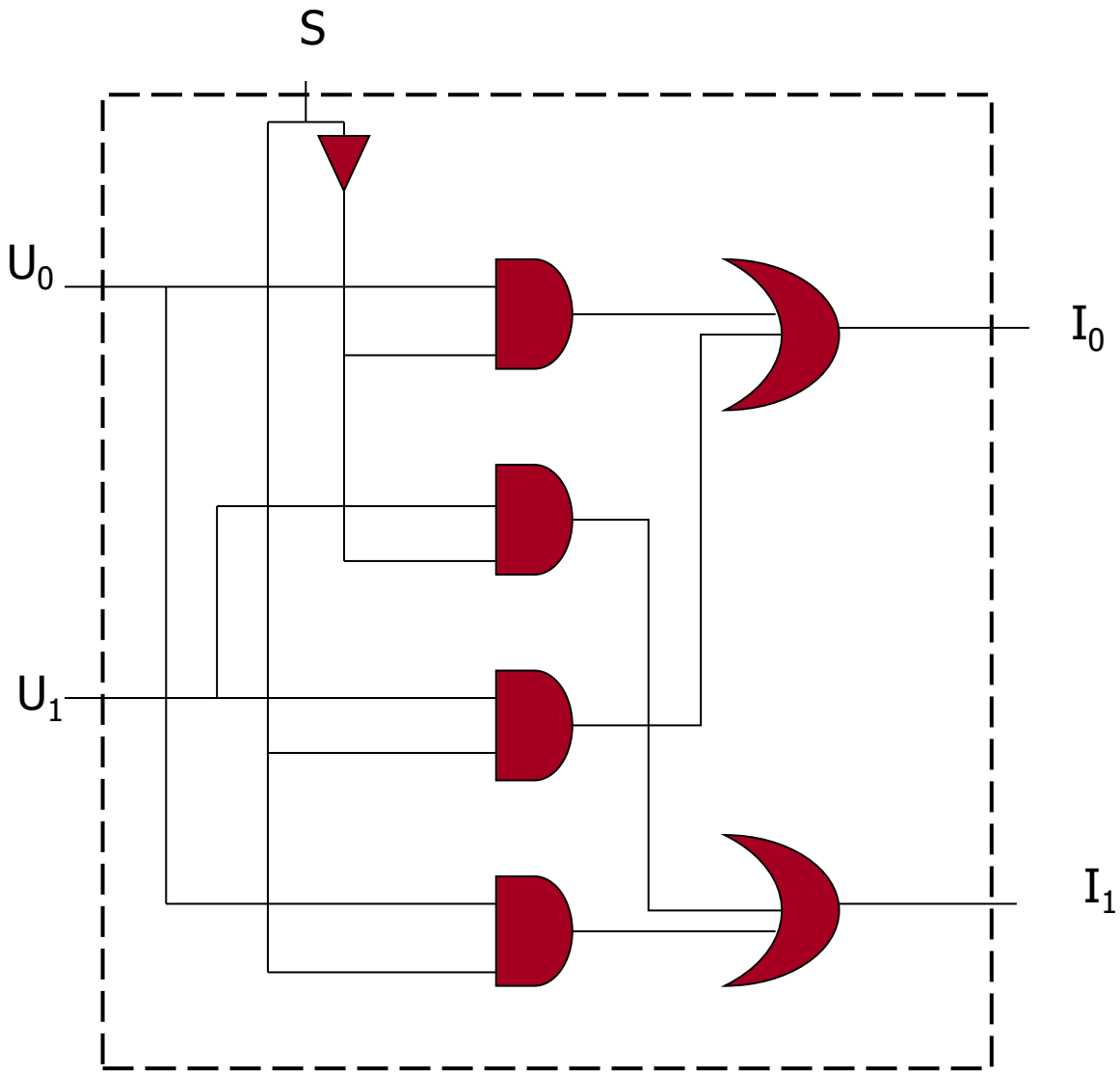


Komutacioni element

- U opštem slučaju može imati a ulaza i b izlaza
  - a, b teorijski ne moraju biti jednaki
  - najčešće je $a=b=2^k$, $k>=1$
  - sa brojem ulaza i izlaza raste složenost KE, pa je najčešće a=b=2 (tipa 2x2)
  - KE tipa 2x2
    - dvofunkcijski
    - četvorofunkcijski



Izgled dvo-funkcijskog KE



S=0, direktno
S=1, ukršteno

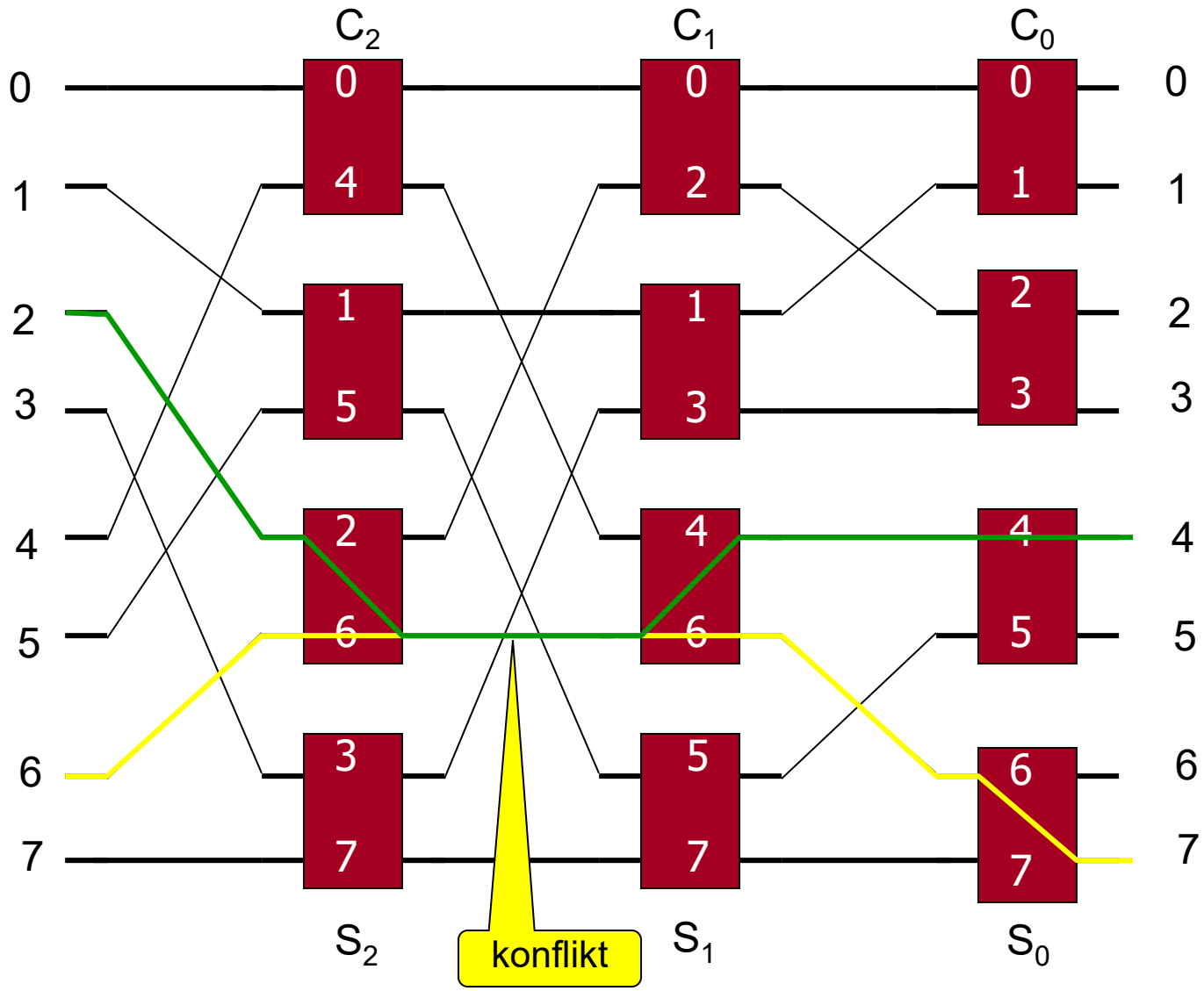
Višestepene SM

- Višestepena SM koja treba da poveže $N=2^m$ PE-ova sastoji se od $\log_2 N=m$ stepena i N/2 KE u svakom stepenu.
- U odnosu na strategiju upravljanja mogu se razlikovati mreže sa:
  - individualnim upravljanjem stepena (potrebno je m linija za upravljanje)
  - individualno upravljanje KE (potrebno je N/2*m linija za upravljanje)
  - nezavisno upravljanje na nivou grupe u jednom stepenu

Višestepena kub mreža (generalizovani kub)

- Sastoji se od $\log_2 N = m$ stepena i N/2 KE
- Svaki stepen realizuje jednu sprežnu f-ju kub mreže
  - Stepen S_0 –sprežnu f-ju C_0
  - Stepen S_1 –sprežnu f-ju C_1
  - ⋮
  - Stepen S_{m-1} –sprežnu f-ju C_{m-1}
  - Stepeni su označeni od izlaza ka ulazu sa $S_0, \ldots, S_{m-1}$

Generalizovani kub –primer N=8



Generalizovani kub - osobine

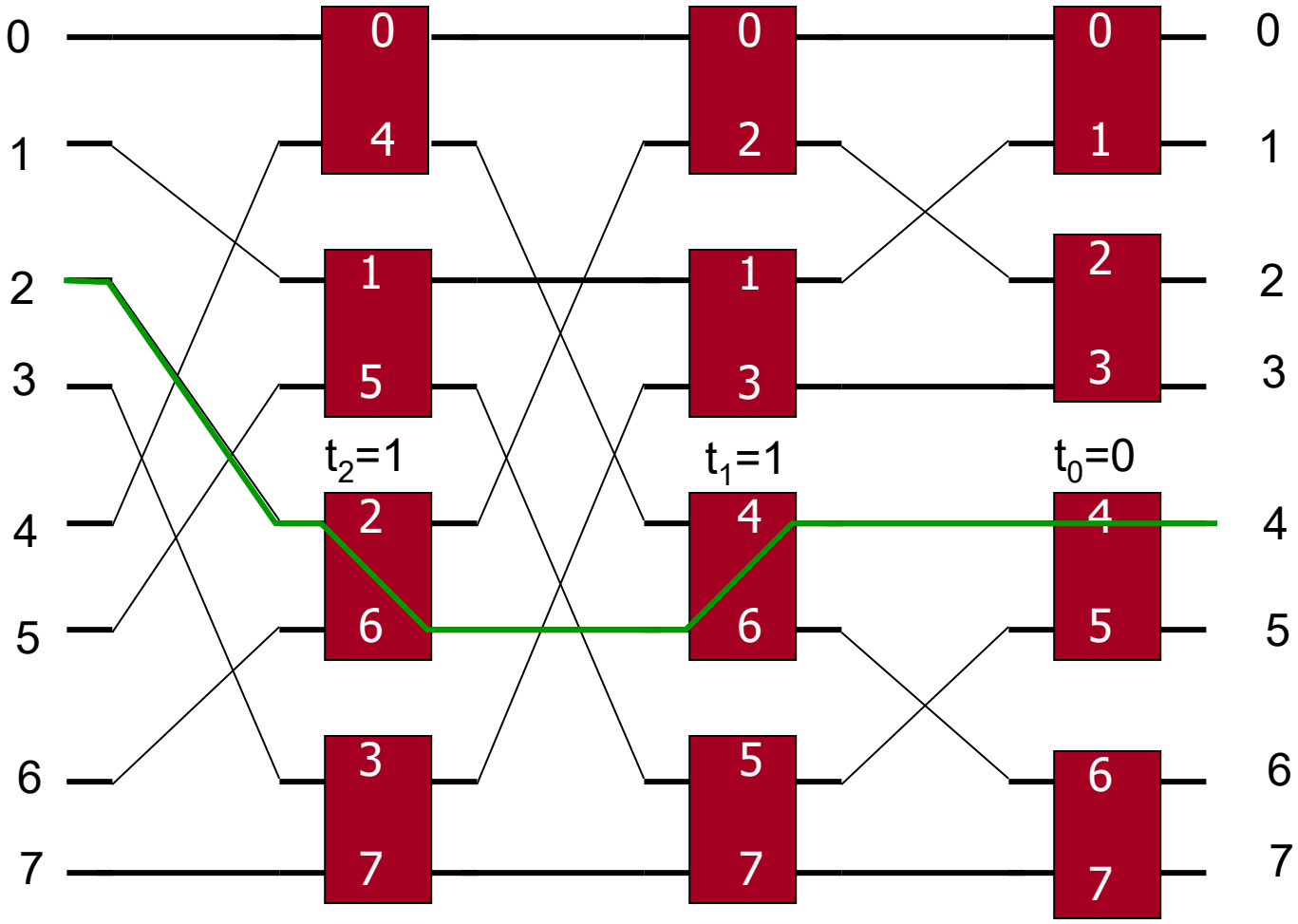
- Može povezati proizvoljni par PE
- Jedinstveni put izmedju svakog para izvor-odredište
- Blokirajuća mreža – mogu nastupiti konflikti

Generalizovani kub – odredjivanje puta poruka

- Jedan-na-jedan
  - pomoću EX-OR zaglavlja
  - pomoću odredišne adrese
- Jedan izvor u 2^k odredišta (emisija)
  - varijanta EX-OR
- Odreddjivanje puta pomoću EX-OR zaglavlja
  - poruci se dodaje zaglavlje dužine m bitova (2^m PE-ova)
    - zaglavlje se dobija primenom EX-OR operacija na binarnim adresama izvora i odredišta
    - adresa izvora $S=s_{m-1}s_{m-2}...s_1s_0$
    - adresa odredišta $D=d_{m-1}d_{m-2}...d_1d_2$
    - zaglavlje $T=S\oplus D=t_{m-1}t_{m-2}...t_1t_0$
    - da bi se odredio položaj KE (direktno ili ukršteno) u stepenu i, potrebno je ispitati t_i

$$t_i = \begin{cases} 0, & diredktno \\ 1, & ukrsteno \end{cases}$$

Odredjivanje puta - primer



S=2= 010, D=4= 100, T=010⊕100=110=$t_2t_1t_0$ (ukršteno, ukršteno, direktno)

Zašto ovo funkcioniše?

- Odgovor se krije u strukturi SM i definiciji sprežnih funkcija:
  - Ako se S i D razlikuju na i-toj bit poziciji, $s_i \oplus d_i = t_i = 1$, neophodno je primeniti sprežnu f-ju C_i da bi se ostvarilo povezivanje (preslikavanje) izvora S i odredišta D
    - Postavljanje KE u stepenu i u položaj ukršteno implementira sprežnu f-ju C_i
  - Ako se adrese S i D ne razlikuju u i-toj bit poziciji, $s_i \oplus d_i = t_i = 0$, f-ja C_i ne treba da se primeni
    - postavljanjem prekidača u položaj direktno f-ja C_i se ne implementira
- Dobra osobina ovog načina odredjivanja puta
  - na osnovu poruke sa zaglavljem moguće je odrediti odakle je poruka pristigla jer važi $S = T \oplus D$
    - korisno ako je potrebno potvrdjivanje prispelih poruka

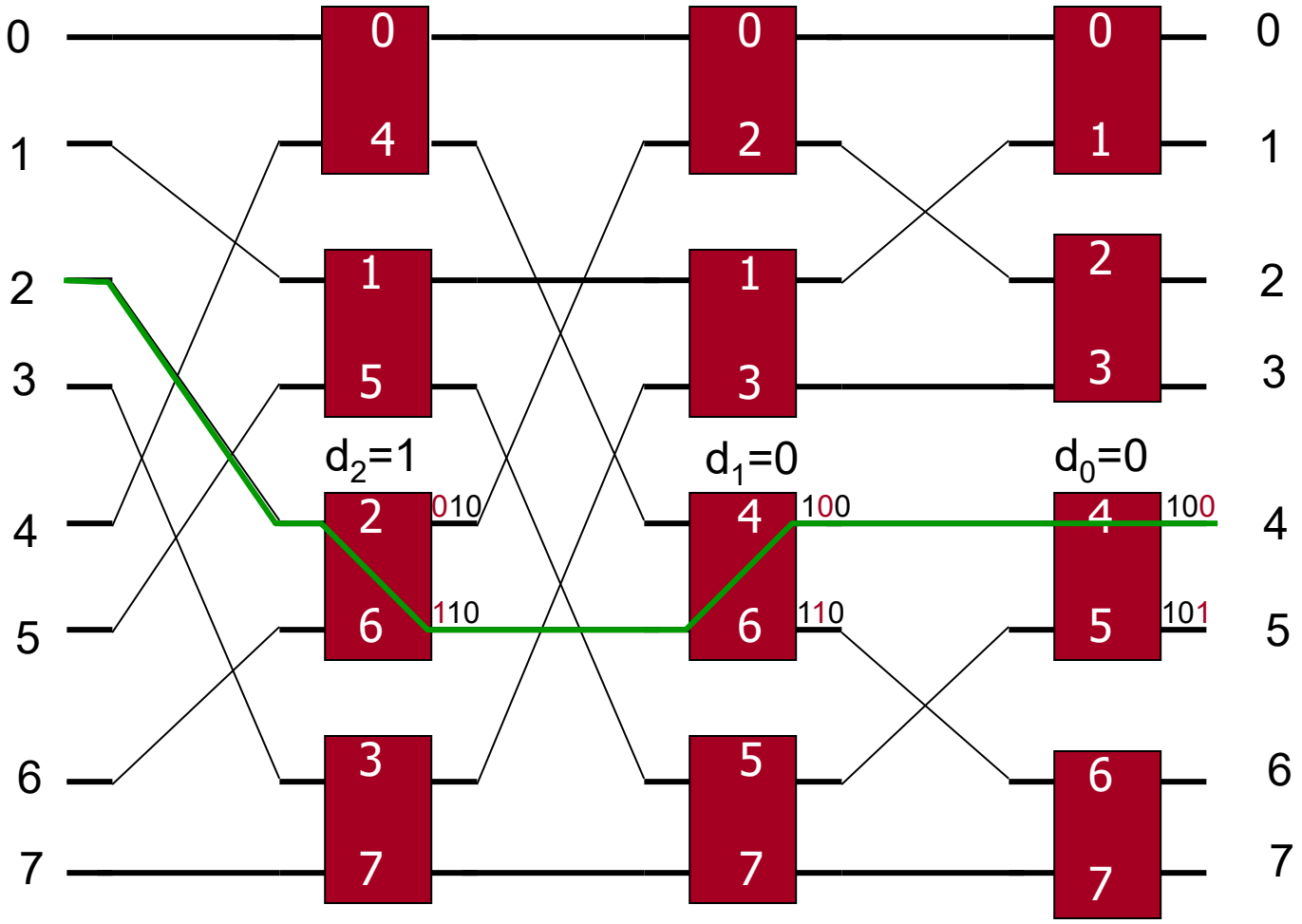
Odredjivanje puta na osnovu adrese odredišta

- Kao zaglavlje može se koristiti adresa odredišta

$$d_i = \begin{cases} 0, & gornji \quad izlaz \quad KE \\ 1, & donji \quad izlaz \quad KE \end{cases}$$

- Primer:
  - S=2= 010, D=4= 100 $\Rightarrow$ donji izlaz, gornji izlaz, gornji izlaz
  - Osobine
    - moguće je na osnovu zaglavlja utvrditi da li poruka stiže po korektnom mrežnom izlazu (poredjenjem adrese odredišta i zaglavlja poruke)
    - nije moguće odediti adresu izvora (problem ako je potrebno potvrdjivanje)

Primer – odredjivanje puta na osnovu odredišne adrese



Zašto ovo funkcioniše: gornji izlaz u stepenu i na i-toj bit poziciji uvek ima vrednost 0, a donji izlaz vrednost 1

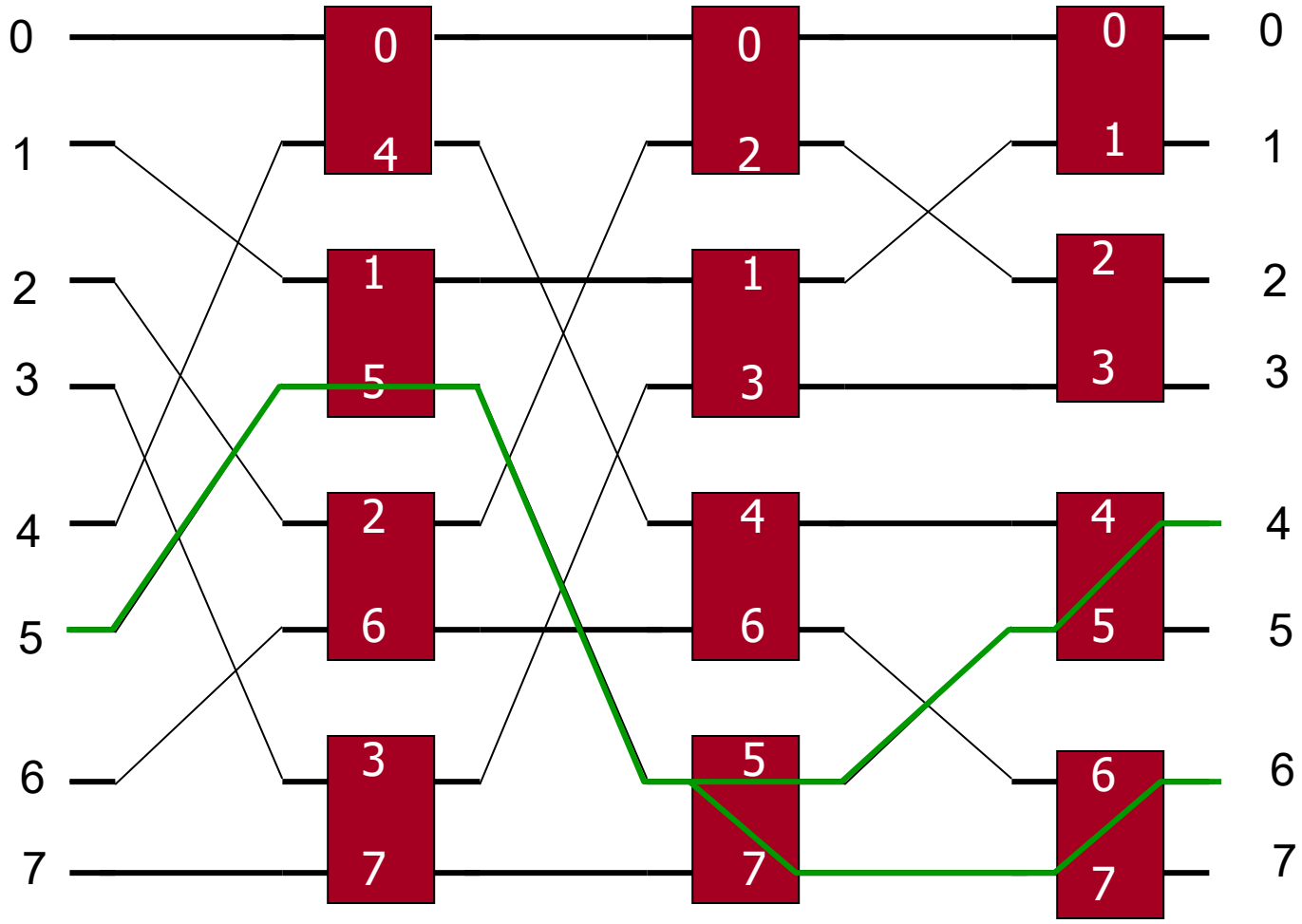
Odredjivanje puta za slučaj emisije

- Koristi se proširena šema sa EX-Or zaglavljem
  - da bi emisija bila moguća broj odredišta mora biti stepen dvojke, 2^j
  - odredišne adrese se moraju razlikovati na j pozicija, a poklapati na m-j pozicija
  - PRIMER
    - S adresa izvora, E i F adrese odredišta
    - Individualna zaglavlja za destinacije E i F, $T_E = S \oplus E$, $T_F = S \oplus F$
    - S=101 (5), E=100 (4), F=110 (6), T_E=001, T_F=011
    - Zaglavlja se slažu u svim bit pozicijama izuzev na mestu gde se razlikuju odredišne adrese
      - putevi kroz mrežu su identični u stepenu 2, granaju se (dolazi do emisije) u stepenu 1, i paralelni su na dalje (koriste isto postavljanje KE u stepenu 0)
      - Da bi se odredilo zaglavlje za slučaj emisije potrebno je poznavati informaciju o putu pre i posle grananja i tačke grananja
      - Zaglavlje za slučaj emisije odredjeno je skupom {R,B}
        - » $R = r_{m-1} \ldots r_1 r_0$ sadrži informaciju o putua, $B = b_{m-1} \ldots b_1 b_0$ informacije o tačkama grananja

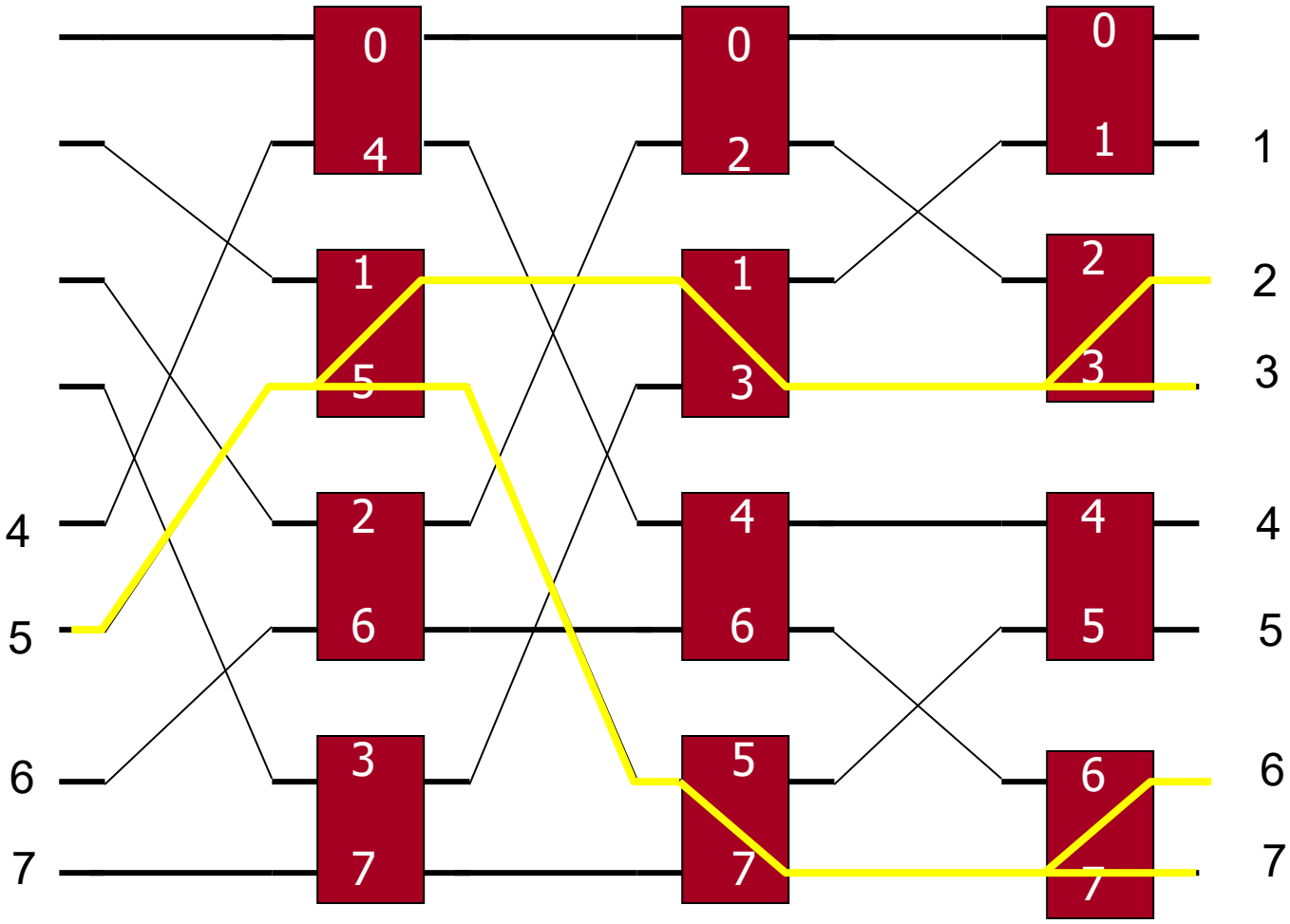
Odredjivanje puta za slučaj emisije –nast.

- I T_E i T_F sadrže informaciju o putu, pa se za R može odabrati bilo koji, npr. R= T_E.
- Da bi se odredilo $B=b_{m-1}...b_1b_0$ potrebno je naći $B=T_E \oplus T_F=E \oplus F$
  - Za S=101 (5), E=100 (4), F=110 (6), {R=001, B=E ⊕F=010}
  - Da bi se odredilo u koiji položaj se postavlja KE u stepenu i potrebno je ispitati r_i i b_i.
    - ako je b_i=1, r_i se ignoriše i obavlja se emisija (grnja ili donja)
    - b_i=0, r_i se koristi da se odredi položaj KE

Emisija iz PE_5 u PE_4 i PE_6



Emisija iz PE_5 u PE_2, PE_3, PE_6 PE_7



R=S ⊕ D2=101 ⊕010=111, B=D2 ⊕D7=010 ⊕111=101
kada se odredjuje B potrebno je potražiti ⊕odredišnih adresa koje se razlikuju na svim dozvoljenim pozicijama

Omega mreža

- Sastoji se od $\log_2 N = m$ stepena i N/2 KE
- Veze izmedju stepena su ostvarene po principu savršenog mešanja, a KE obavljaju f-ju zamene kada su postavljeni u položaj ukršteno

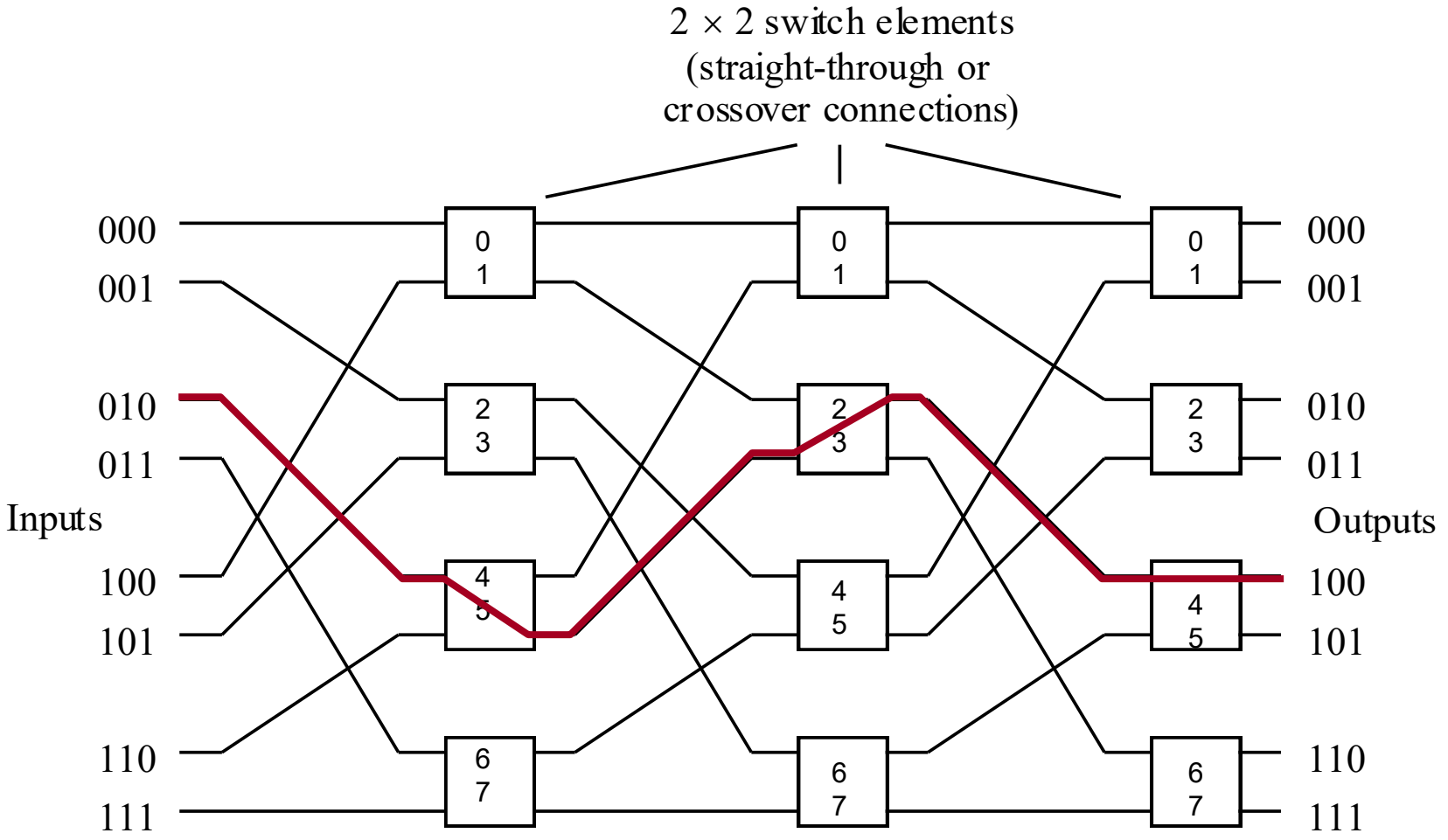
$$S(b_{m-1}, b_{m-2}, \ldots, b_1, b_0) = b_{m-2}, \ldots, b_1, b_0, b_{m-1}$$

$$E(b_{m-1}, b_{m-2}, \ldots, b_1, b_0) = b_{m-1}, b_{m-2}, \ldots, b_1, \overline{b}_0$$

- Put poruke se odredjuje na osnovu EX-OR adrese izvora i odredišta po istom principu kao kod generalizovanog kuba

Omega mreža – primer za N=8

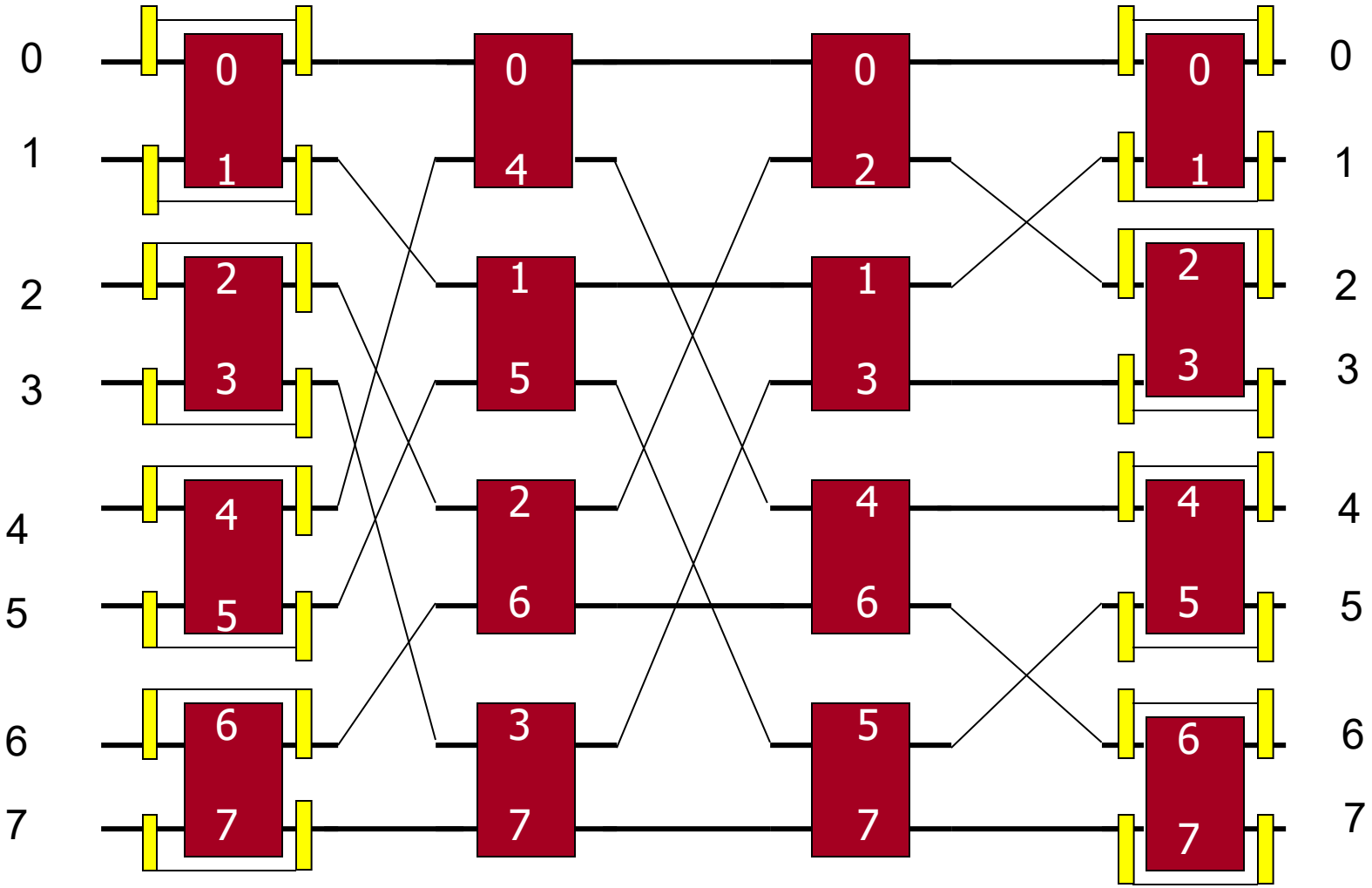
T=S ⊕D=010 ⊕100=110 (ukršteno, ukršteno, direktno)



Generalizovani kub sa ekstra stepenom(ESC)

- Ako nastupi greška na nekom kom. kanalu ili u KE, vezu izmedju odredjenog para izvor-odredište nije moguće uspostaviti (jer postoji jedinstveni put)
- Problem je moguće rešti dodavanjem ekstra stepena na ulaz mreže (stepen Sm)
  - ekstra stepen obavlja sprežnu f-ju C_0
  - potrebno je dodati i MUX i DMUX na ulaze ekstra stepena (stepena m) i poslednjeg stepena (stepena 0)
  - sada postoje dva stepena koji obavljaju spr. f-ju C_0
  - dodavanjem ekstra stepena dobijaju se dva puta izmedju svakog para izvor-odredište koji ne koriste iste KE i kom. kanale u stepenima 1,2,...,m-1
  - mreža postaje otporna na jednostruke greške

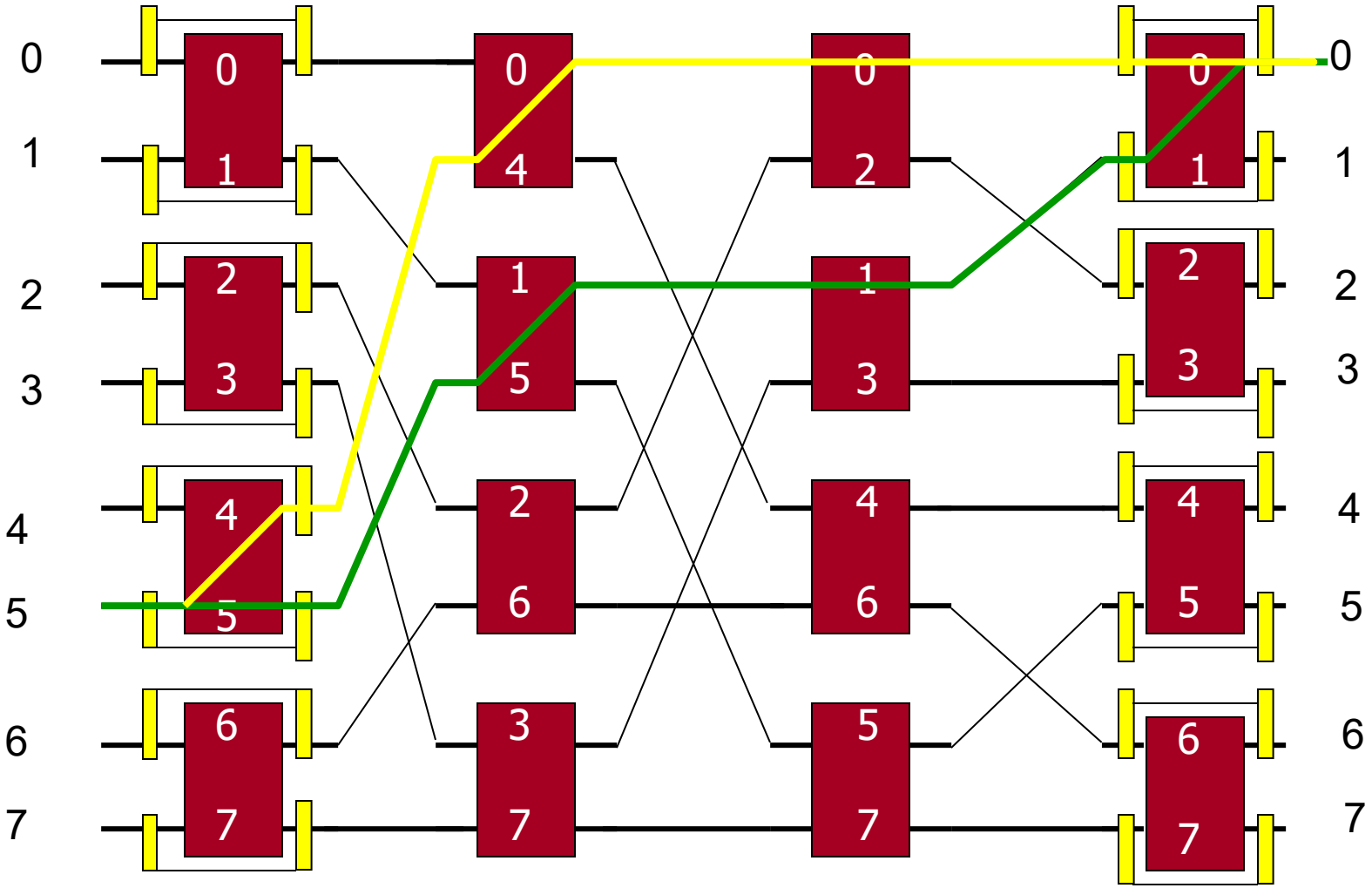
ESC



ESC

- Mreža je normalno konfigurisana da je stepen m premošćen, a stepen 0 aktivan
- analiza greške
  - mreža se testira za poznate test oblike
  - ako se detektuje jednostruka greška preduzimaju se sledeće akcije
    - ako je greška u stepenu m, stepen 0 je aktivan
    - ako je greška u stepenu 0, stepen 0 je premošćen a m aktivan
    - ako je greška u stepenu i, 0<i<m, stepeni 0 i m su aktivni
    - kada su stepeni 0 i m aktivni u mreži postoje dva disjunktna puta izmedju svakog izvora i odredišta (primarni i sekundarni)
    - Zaglavlje za primarni put $T_p = 0t_{m-1}\ldots t_1 t_0$
    - Zaglavlje za sekundarni put $T_s = 1t_{m-1}\ldots t_1 \bar{t}_0$

ESC



zeleno – primarni put, žuto – sekundarni put