OSNOVE RAČUNALNIŠKE ARHITEKTURE II

8 Cevovodna realizacija CPE

- Pri CPE, zgrajeni na običajni način, zaradi zaporednega izvajanja korakov, ki sestavljajo ukaz, težko dosežemo *CPI* manjši od 3 ali 4.

- Število ukazov, ki jih CPE izvede v eni sekundi

$$\uparrow MIPS = \frac{\uparrow f_{CPE}}{\downarrow CPI \cdot 10^6}$$

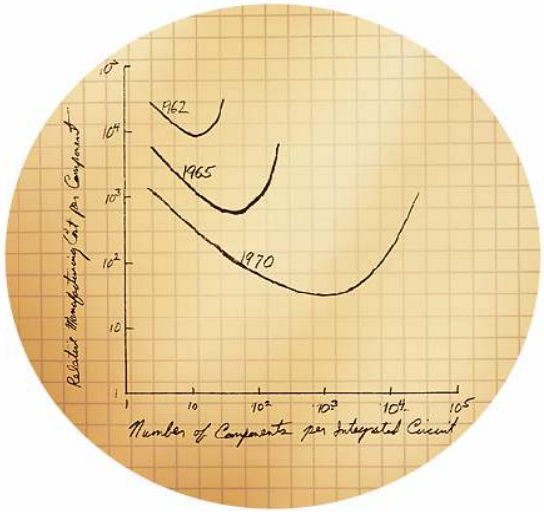
lahko povečamo:

- z uporabo hitrejših elementov (povečanje f_{CPE})
- z uporabo večjega števila elementov (zmanjšanje *CPI*)

- Uporaba hitrejših elementov ne omogoča velikega povečanja hitrosti, s seboj pa prinaša še druge probleme.
- Če želimo učinkovito povečati hitrost CPE, moramo CPE narediti tako, da paralelno izvaja več funkcij, to pa pomeni povečanje števila logičnih elementov.
- Največkrat uporaljana rešitev paralelnega izvajanja več funkcij je izvedba CPE v obliki **cevovoda (pipeline).**

- Revija Electronic Magazine je leta 1965 objavila članek Gordona E. Moora, v katerem je napovedal, da se bo število tranzistorjev, ki so jih proizvajalci sposobni izdelati na čipu, podvojilo vsako leto.
- Leta 1975 je napoved popravil, da se bo število tranzistorjev podvojilo na vsaki dve leti.
- Kar je bilo takrat mišljeno kot izkustveno pravilo in naj bi veljalo naslednjih nekaj let, velja še danes in je poznano kot Moorov zakon.

Moore's Law



In 1965, Gordon Moore sketched out his prediction of the pace of silicon technology. Decades later, Moore's Law remains true, driven largely by Intel's unparalleled silicon expertise.

According to Moore's Law, the number of transistors on a chip roughly doubles every two years. As a result the scale gets smaller and smaller. For decades, Intel has met this formidable challenge through investments in technology and manufacturing resulting in the unparalleled silicon expertise that has made Moore's Law a reality.

- Gordon E. Moore je danes častni predsenik Intela, v letu 1968 pa je bil soustanovitelj in izvršni podpredsednik Intela.
- Pri isti tehnologiji se je v zadnjih 20 letih najvišja hitrost logičnih elementov povečala za približno 10-krat.
- V istem času se je največje število elementov na enem čipu povečalo za približno 500 do celo 5000-krat pri pomnilniških čipih.

Cevovodna CPE

Microprocessor	Year of Introduction	Transistors
4004	1971	2,300
8008	1972	2,500
8080	1974	4,500
8086	1978	29,000
Intel286	1982	134,000
Intel386™ processor	1985	275,000
Intel486™ processor	1989	1,200,000
Intel® Pentium® processor	1993	3,100,000
Intel® Pentium® II processor	1997	7,500,000
Intel® Pentium® III processor	1999	9,500,000
Intel® Pentium® 4 processor	2000	42,000,000
Intel® Itanium® processor	2001	25,000,000
Intel® Itanium® 2 processor	2003	220,000,000
Intel® Itanium® 2 processor (9MB cache)	2004	592,000,000

8.1 Cevovodna CPE

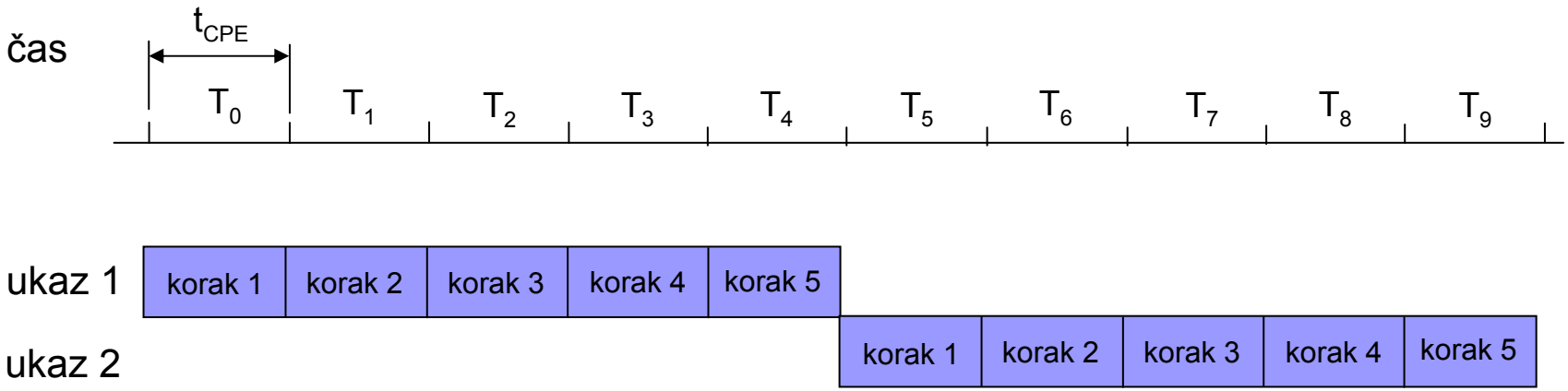
- Je realizacija CPE, kjer se hkrati izvršuje več ukazov, tako da se posamezni koraki izvrše-vanja ukazov prekrivajo.
- V cevovodni CPE se ukazi izvršujejo podobno tekočemu traku v proizvodnji (npr. avtomobilov).
- Izvrševanje ukaza se razdeli na manjše **podoperacije**, za vsako je potreben samo del celotnega časa, ki je potreben za izvršitev ukaza.

- CPE je razdeljena na **stopnje** ali **segmente cevovoda**, ki jih je toliko kot podoperacij v ukazu.
- Vsako podoperacijo izvrši določena stopnja ali segment cevovoda.
- Stopnje so med seboj povezane, ukazi na eni strani vstopajo, potujejo skozi stopnje, v katerih se izvršujejo posamezne podoperacije ukazov in na drugi strani izstopajo.
- V cevovodu je hkrati v izvrševanju toliko ukazov, kot je stopenj.

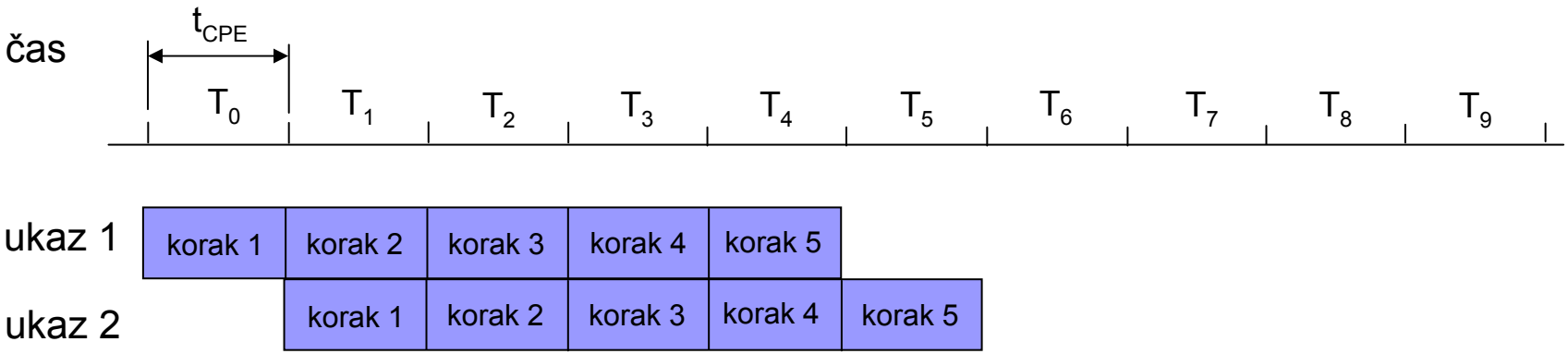
- Zmogljivost cevovodne CPE je določena s hitrostjo izstopanja ukazov iz cevovoda.
- Ker so stopnje med seboj povezane, se mora premik ukaza iz ene stopnje v drugo izvršiti pri vseh hkrati.
- Premik se običajno izvede vsako urino periodo.
- Trajanje urine periode t_{CPE} zato ne more biti krajše kot je čas, ki ga potrebuje za izvedbo podoperacije najpočasnejša stopnja cevovoda.

Izvrševanje ukazov pri necevovodni in cevovodni CPE

Necevovodna CPE



Cevovodna CPE



- Pri razvoju cevovodne CPE se stremi za tem, da izvrševanje vseh podoperacij traja približno enako dolgo.
- V tem primeru je cevovodna CPE uravnotežena.
- Pri idealno uravnoteženi CPE z *N* stopnjami ali segmenti je zmogljivost *N*-krat večja kot pri necevovodni CPE.
- Vsak posamezen ukaz se ne izvrši nič hitreje, se pa v cevovodu hkrati izvršuje *N* ukazov.

- Na izhodu cevovoda dobimo v enakem času *N*-krat več izvršenih ukazov kot pri necevovodni CPE.
- Število ukazov, ki jih cevovodna CPE izvede v določenem času, se zveča zaradi:
  - Manjšega števila urinih period za izvedbo enega ukaza (*CPI*)
  - Krajše urine periode (t_{CPE})
- Povprečno število urinih period na ukaz (*CPI*) je v idealnem primeru *N*-krat manjše kot pri necevovodni CPE.

- Trajanje izvrševanja posameznega ukaza (latenca) pa je enako $N \times t_{CPE}$, torej pri enaki urini periodi enako kot pri necevovodni CPE.
- Za krajšanje urine periode pri uvedbi cevovoda ni veliko možnosti:
  - Potrebno bi bilo razdeliti ukaze na več zelo enostavnih operacij
  - Možnosti za razdelitev ukazov na podoperacije pa ni prav veliko
  - Več možnosti je pri kompleksnih ukazih (CISC), kot pri enostavnih (RISC)

- Zaradi zapletenih ukazov pa je pri CISC računalnikih težje narediti cevovodno CPE

- Pogosto se t_{CPE} celo podaljša:
  - Vse podoperacije se morajo izvršiti v enakem času
  - Rezultat podoperacije, ki jo izvrši neka stopnja, je treba shraniti v vmesni register, da ga lahko v naslednji urini periodi uporabi naslednja stopnja
  - Večje število logičnih elementov (časovni zamik urinega signala zaradi različno dolgih poti do registrov in ff)

- Ali bi pri dovolj velikem številu stopenj *N* lahko naredili poljubno hitro CPE (*N-krat* hitrejšo)? Ne.
- Med delovanjem cevovoda prihaja do **cevovodnih nevarnosti** (pipeline hazards) zaradi katerih se mora cevovod ustaviti in počakati, da nevarnost mine.
- Razlikujemo tri vrste cevovodnih nevarnosti:
  - **Strukturne nevarnosti** - kadar več stopenj cevovoda v isti urini periodi potrebuje isto enoto

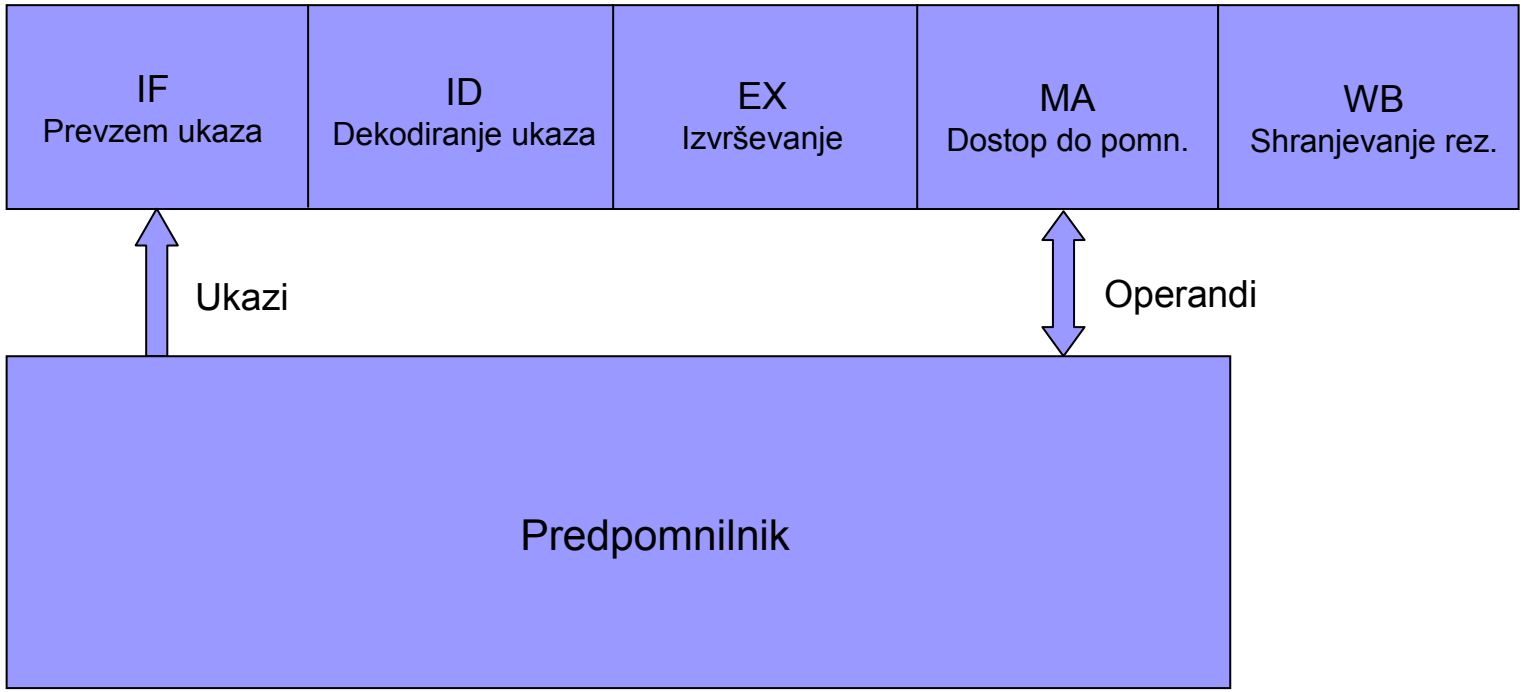
- **Podatkovne nevarnosti** - kadar nek ukaz potrebuje rezultat predhodnega ukaza, ki pa še ni končan
- **Kontrolne nevarnosti** - pri ukazih, ki spreminjajo vrednost PC (kontrolni ukazi: skoki, klici, …)

- Zaradi cevovodnih nevarnosti se mora cevovod ustavljati, zato povečanje hitrosti ni *N-kratno.*
- Z večanjem števila stopenj *N* se cevovodne nevarnosti pojavljajo pogosteje in cevovod ni več tako učinkovit.
- Danes so vsi zmogljivejši procesorji narejeni kot cevovodni.

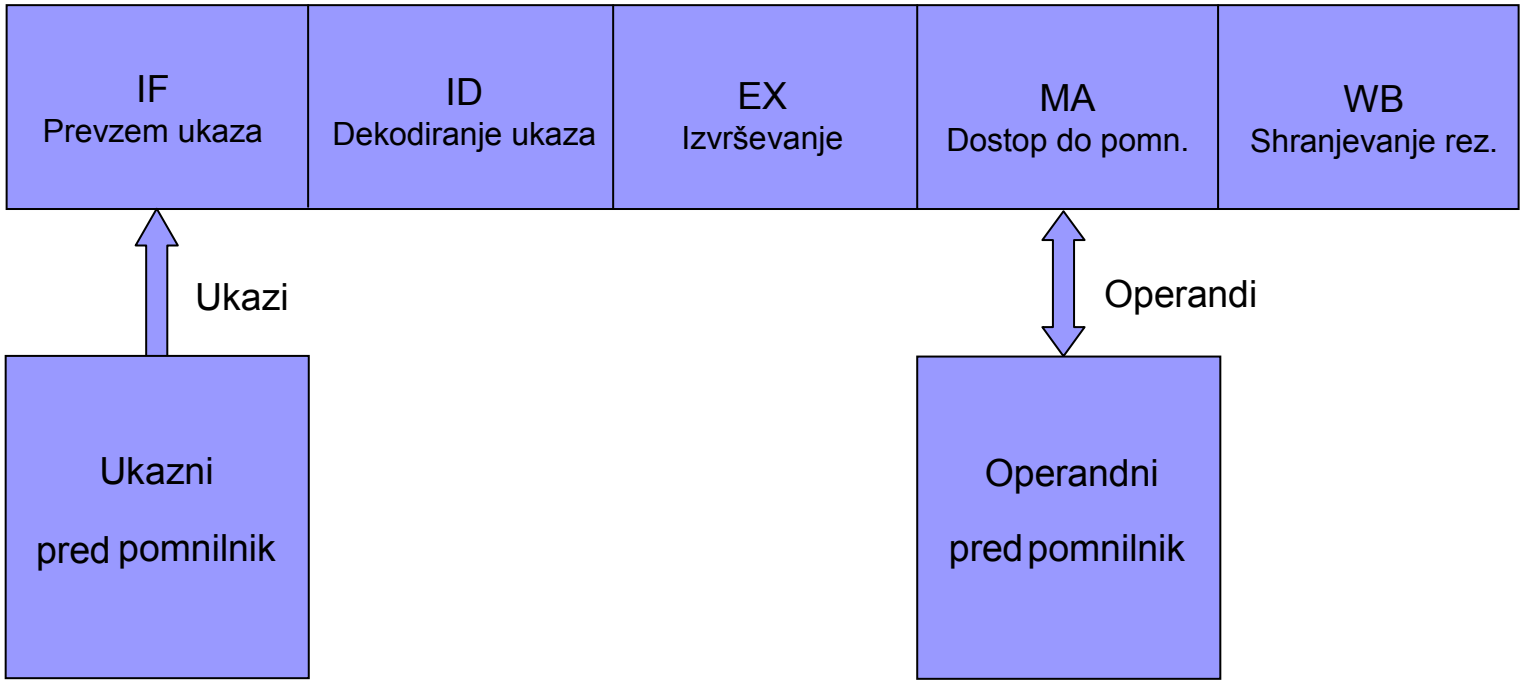
8.2 Primer petstopenjske cevovodne CPE

- Osnova naj bo izvajanje ukazov v petih korakih, kot smo ga opisali v prejšnjem poglavju.
- Izvrševanje ukaza razdelimo na pet podoperacij v skladu s koraki iz prejšnjega poglavja, CPE pa v pet stopenj oziroma segmentov:
  - Stopnja IF (Instruction Fetch) - prevzem ukaza
  - Stopnja ID (Instruction Decode) - dekodiranje ukaza in dostop do registrov
  - Stopnja EX (Execute) - izvrševanje operacije

- Stopnja MA (Memory Access) - dostop do pomnilnika
- Stopnja WB (Write Back) - shranjevanje rezultata



- Stopnja MA (Memory Access) - dostop do pomnilnika
- Stopnja WB (Write Back) - shranjevanje rezultata



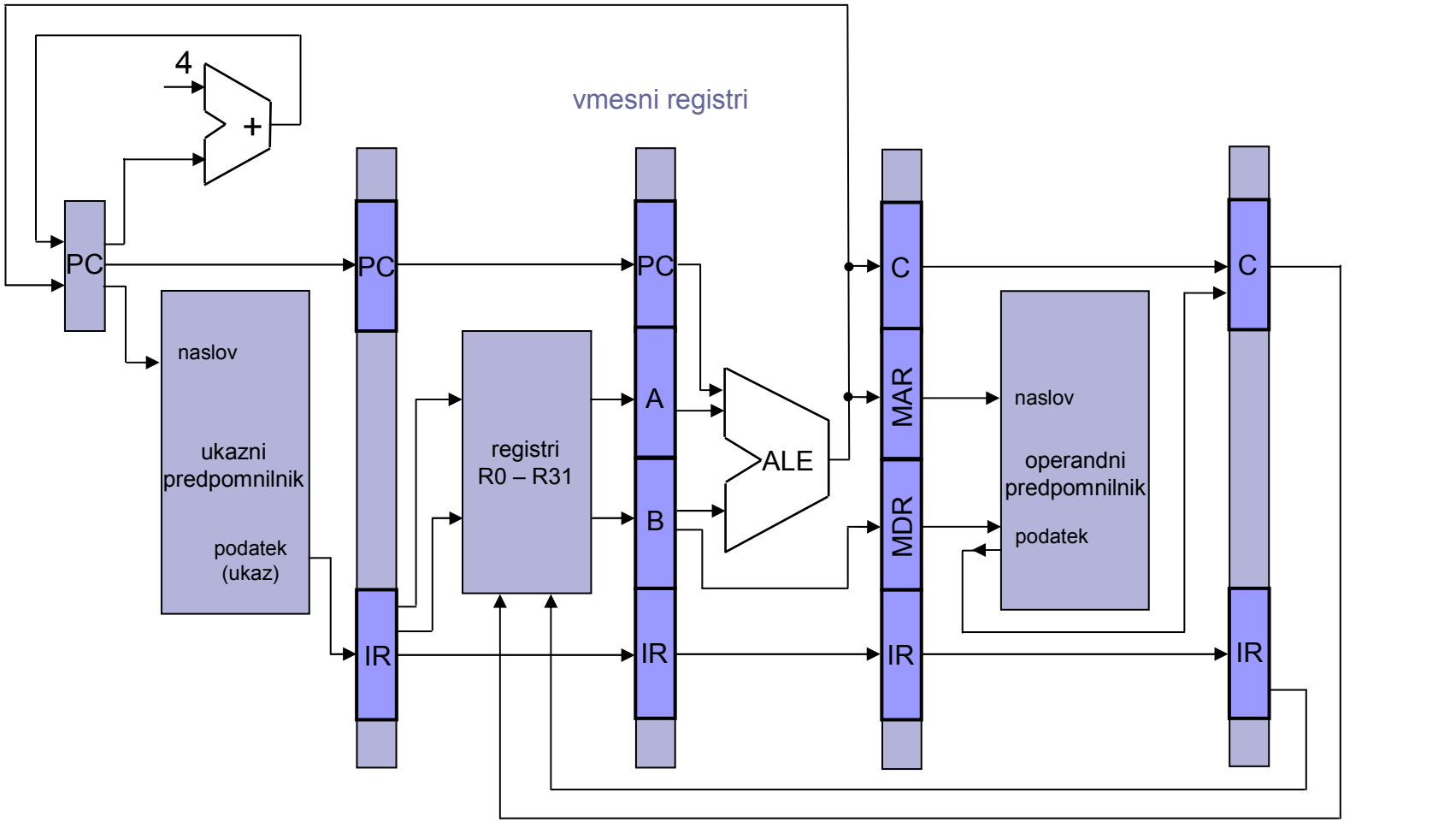
- Vsaka stopnja cevovoda mora izvršiti svojo podoperacijo v eni urini periodi.
- V stopnjah IF in MA lahko pride do hkratnega dostopa do pomnilnika (v isti urini periodi) - strukturna nevarnost.
- Predpomnilnik moramo zato razdeliti v ukazni in operandni predpomnilnik (Harvardska arhitektura).

- V stopnji IF se dostop do ukaznega predpomnilnika opravi vsako urino periodo, pri necevovodni CPE pa samo na vsakih pet urinih period.
- Hitrost prenosa informacij med predpomnilnikom in CPE mora biti zato petkrat večja kot pri necevovodni CPE.
- Pri načrtovanju cevovodne CPE je treba paziti, da od neke enote (register, ALE, …) ne zahtevamo, da bi v določeni urini periodi morala delati dve različni operaciji.

Zgradba in delovanje 5-stopenjske cevovodne CPE

- Cevovod ima 5 stopenj, med njimi so vmesni registri, v katere se shranijo rezultati podoperacij vsake stopnje, ki jih potrebujejo druge stopnje.
- V stopnji IF se prebere ukaz, ki se prenese v ukazni register in za 4 poveča vsebina programskega števca PC(ukazi so dolgi 4 bajte).
- Programski števec je potrebno povečati v stopnji IF zato, ker se vsako urino periodo prevzame nov ukaz.

Zgradba 5-stopenjske cevovodne CPE



- Naslov ukaza, ki se izvaja (vsebina PC), se shranjuje v vmesne registre, ker je pri kontrolnih ukazih potreben v stopnji EX.
- Pri kontrolnih ukazih se v PC namreč vpiše nova vrednost (skočni naslov), ki se izračuna v stopnji EX.
- Vsaka stopnja izvršuje drug ukaz, zato je treba v vmesne registre IR med vsemi stopnjami vedno shranjevati tudi ukaz, ki se v stopnji IF vsako urino periodo prebere iz ukaznega predpomnilnika.

- V tabeli so v vsaki urini periodi napisane stopnje, ki v tej urini periodi delujejo pri določenem ukazu.
- Stopnje, ki v določeni urini periodi niso napisane, v tej urini periodi ne delujejo.

urine periode	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
ukaz 1	IF_1	ID_1	EX_1	MA_1	WB_1				
ukaz 2		IF_2	ID_2	EX_2	MA_2	WB_2			
ukaz 3			IF_3	ID_3	EX_3	MA_3	WB_3		
ukaz 4				IF_4	ID_4	EX_4	MA_4	WB_4	
ukaz 5					IF_5	ID_5	EX_5	MA_5	WB_5

- Vsako urino periodo se prevzame nov ukaz in potuje skozi stopnje cevovoda.
- Če se cevovod ne bi ustavljal zaradi cevovodnih nevarnosti, bi bila njegova zmogljivost 5-krat večja kot pri necevovodni CPE.

Cevovodne nevarnosti

- Do cevovodnih nevarnosti pride, kadar so zaporedni ukazi, ki so istočasno v izvajanju v cevovodu, med seboj odvisni.
- Če pride do cevovodne nevarnosti, se mora cevovod ustaviti in čakati, dokler nevarnost ne mine.
- V cevovod mora biti zato vgrajena logika za ugotavljanje cevovodnih nevarnosti.

- Cevovodne nevarnosti se lahko odpravi tudi programsko, z vstavljanjem ukazov NOP za ukazi, ki lahko povzročijo cevovodno nevarnost.
- Vstavljanje NOP ukazov upočasni delovanje CPE in onemogoča združljivost programov, zato se danes ne uporablja pogosto.
- Logika za ugotavljanje cevovodnih nevarnosti pa omogoča, da lahko odpravimo del čakanja, ki ga povzročijo cevovodne nevarnosti.

Strukturne nevarnosti (structural hazards)

- Strukturna nevarnost nastane, kadar dva ukaza v isti urini periodi potrebujeta isto enoto, ki je sposobna izvajati samo eno podoperacijo naenkrat.
- Primeri strukturnih nevarnosti:
  - Hkratni dostop do predpomnilnika v stopnjah IF in MA
  - Pisanje v dva programsko dostopna registra hkrati
  - Izvajanje operacij, ki trajajo več urinih period, v stopnji EX

- Strukturne nevarnosti se da odpraviti tako, da se vgradi dovolj veliko število enot.
- Stopnja EX se pri operacijah, ki trajajo več urinih period, lahko zgradi v obliki cevovoda.
- Pri večini računalnikov se dovoli, da občasno pride do strukturnih nevarnosti, ker je upočasnitev precej manjša kot pa zaradi podatkovnih in kontrolnih nevarnosti.

Podatkovne nevarnosti (data hazards)

- Do podatkovne nevarnosti pride, kadar nek ukaz še obdeluje operand, ki ga že potrebuje naslednji ukaz.
- Operand, ki ga ukaz potrebuje, torej še ni na voljo, zato bi bilo bolj pravilno poimenovanje operandne nevarnosti.
- Primer zaporedja ukazov, pri katerem pride do podatkovne nevarnosti:

```
ukaz 1
ADD  R3, R1, #10      R3 ← R1 + 10   (ukaz 2)
ADD  R5, R3, #12      R5 ← R3 + 12   (ukaz 3)
ukaz 4
ukaz 5
```

- Ukaz 3 uporablja kot izvorni operand register R3, ki ga ukaz 2 izračuna v stopnji EX, shrani v register R3 pa šele v stopnji WB.
- Pravilna vsebina v registru R3 je zato na voljo šele tri urine periode za tem, ko jo ukaz 3 potrebuje v stopnji ID.

- Če podatkovne nevarnosti ne odpravimo, bo rezultat tretjega ukaza v registru R5 napačen.
- Enostavna rešitev je, da se stopnja ID pri tretjem ukazu ustavi in čaka 3 urine periode.
- To pomeni, da se ustavi tudi stopnja IF pred ID (sicer bi izgubili ukaz, ki je v njej), stopnje EX, MA in WB pa delujejo naprej, da se drugi ukaz konča in se tako odpravi vzrok za nevarnost.

- Tako delovanje imenujemo cevovodna zaklenitev, če pa se ustavijo vse stopnje cevovoda je to cevovodna zaustavitev.

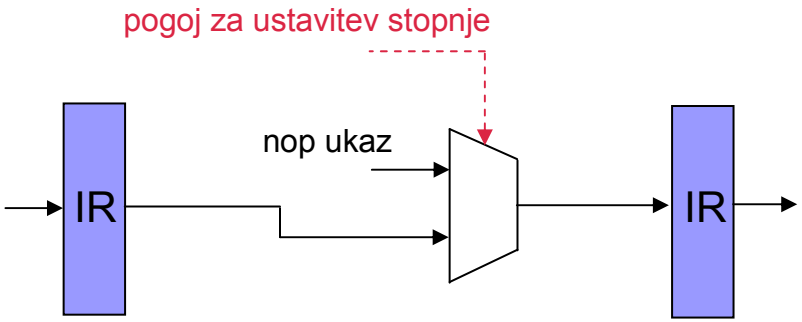
urine periode	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}
ukaz 1	IF	ID	EX	MA	WB							
ADD R3, R1, #10		IF	ID	EX	MA	WB						
ADD R5, R3, #12			IF	O	O	O	ID	EX	MA	WB		
ukaz 4							IF	ID	EX	MA	WB	
ukaz 5								IF	ID	EX	MA	WB

O mehurček - podoperacija, ki ničesar ne spremeni

- Logika za cevovodno zaklenitev mora biti vgrajena v vsako stopnjo cevovoda, ki lahko zaradi uporabe napačnih podatkov povzroči napako.
  - V vsaki urini periodi mora ugotoviti, ali v določeni stopnji obstaja cevovodna nevarnost
  - Če nevarnost obstaja, se stopnja ustavi, prav tako pa tudi vse stopnje pred njo.
  - Stopnje za njo delujejo naprej, vendar iz ustavljene stopnje ne dobijo novega ukaza, zato lahko izvajajo le podoperacije, ki ne spremenijo ničesar

- To imenujemo pri cevovodu vstavljanje mehurčka (bubble), ki ne spremeni ničesar.
- Ta mehurček potuje po stopnjah naprej in pri tem ne opravi nobenega dela.
- Mehurček je lahko ukaz NOP, ki ne naredi ničesar in ga logika za zaklepanje cevovoda vstavi v ustrezno stopnjo (v vmesni register IR)
- To omogoča, da se prvi ukaz izvrši do konca in s tem nevarnost mine.

Vstavljanje ukaza NOP (mehurčka)



- Naslednji diagram prikazuje uporabo (delovanje) stopenj cevovoda pri posameznih ukazih

ukaz 1
ADD R3, R1, #10 R3 ← R1 + 10 ukaz 2
ADD R5, R3, #12 R5 ← R3 + 12 ukaz 3
ukaz 4
ukaz 5

urine periode	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}
stopnja IF	u1	u2	u3				u4	u5				
stopnja ID		u1	u2				u3	u4	u5			
stopnja EX			u1	u2	O	O	O	u3	u4	u5		
stopnja MA				u1	u2	O	O	O	u3	u4	u5	
stopnja WB					u1	u2	O	O	O	u3	u4	u5

ustavljene stopnje

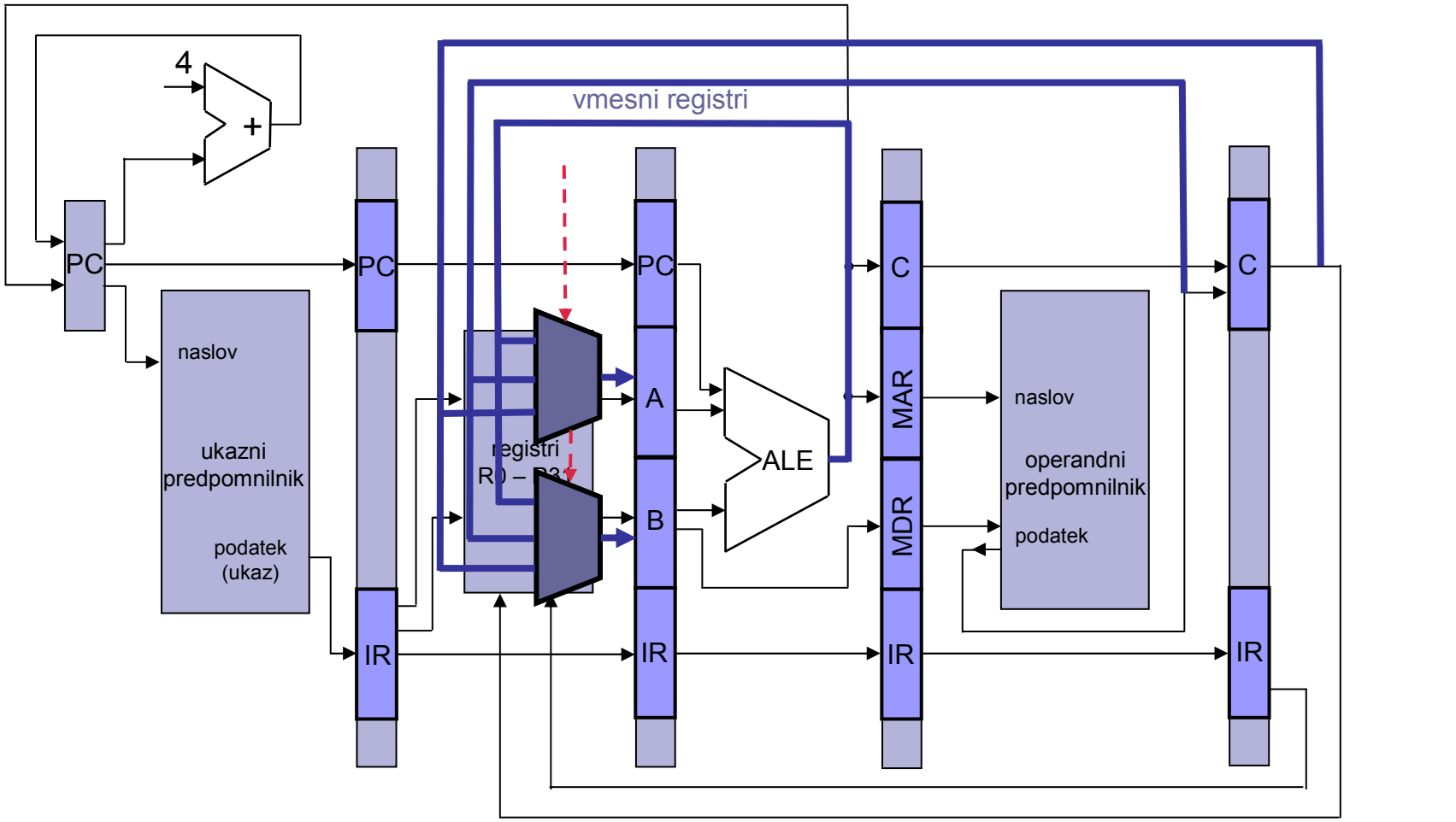
O mehurček - podoperacija, ki ničesar ne spremeni

Premoščanje (bypassing ali data forwarding)

- Premoščanje je dokaj enostavna strojna rešitev, s katero lahko v primeru podatkovnih nevarnosti odpravimo zaklenitev cevovoda za tri urine periode.
- Pri izvajanju našega zaporedja ukazov je rezultat drugega ukaza na voljo že v stopnji EX.
- Če rezultat ukaza iz stopnje EX prenesemo v stopnjo ID, ustavljanje tretjega ukaza ni potrebno.

- Pri podatkovnih nevarnostih, ki nastopijo pri drugačnih zaporedjih ukazov, je premoščanje v stopnjo ID lahko potrebno ne samo iz stopnje EX, temveč tudi iz stopnje MA ali WB.

Zgradba 5-stopenjske cevovodne CPE s premoščanjem



- Če operanda ni v nobeni stopnji cevovoda, potem ni mogoče narediti premoščanja.
- Tak primer je branje operanda iz pomnilnika z ukazom LOAD (v stopnji MA), če ta operand potrebuje naslednji ukaz (v stopnji ID).
- V takem primeru je potrebno:
  - Počakati na operand iz pomnilnika ali
  - spremeniti vrstni red izvajanja ukazov - cevovodno razvrščanje, če je to možno. Cevovodno razvrščanje opravlja prevajalnik.

Kontrolne nevarnosti (control hazards)

- Do kontrolne nevarnosti pride pri vseh ukazih, ki spreminjajo vrednost PC drugače kot po pravilu PC←PC+1.
- To so vsi kontrolni ukazi, kamor spadajo ukazi za brezpogojne in pogojne skoke, klice in vrnitve.
- V cevovodu se klici in vrnitve izvajajo enako kot brezpogojni skoki, zato poenostavljeno govorimo kar o skokih.

- Kadar stopnja EX povzroči spremembo vsebine PC (skočni naslov), sta ukaza, ki sledita skočnemu ukazu v stopnjah IF in ID, neveljavna in se ne smeta izvesti.
- Primer zaporedja ukazov s skočnim ukazom:

ukaz 1 - pogojni skok na ukaz 5

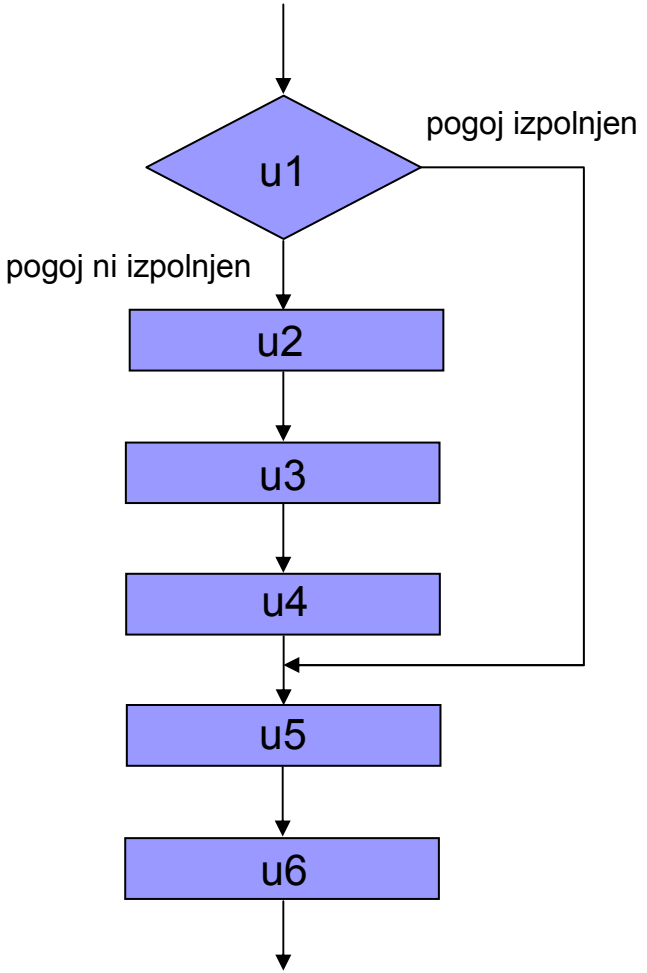
ukaz 2

ukaz 3

ukaz 4

ukaz 5

ukaz 6



ukaz 1 - pogojni skok na ukaz 5

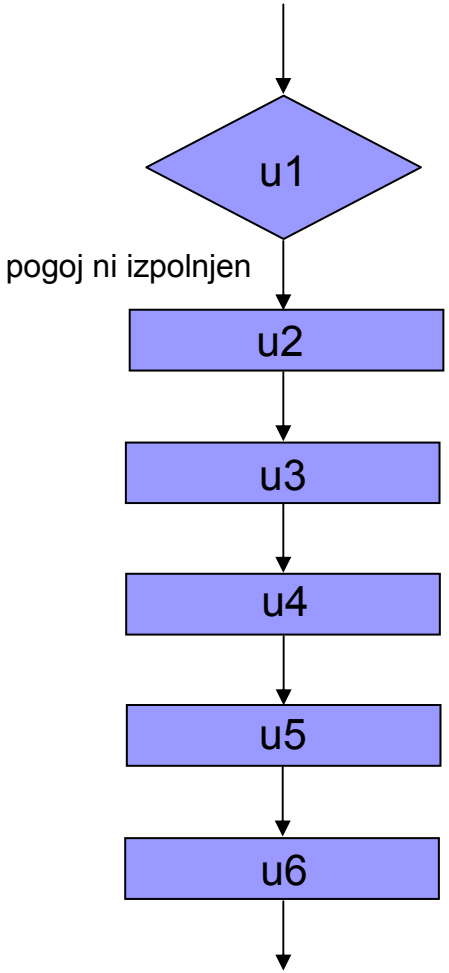
ukaz 2

ukaz 3

ukaz 4

ukaz 5

ukaz 6



ukaz 1 - pogojni skok na ukaz 5

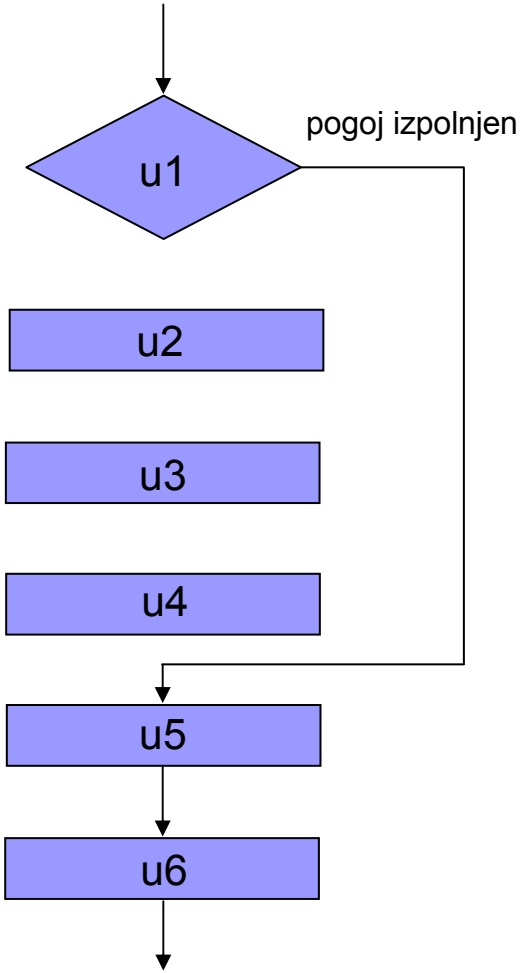
ukaz 2

ukaz 3

ukaz 4

ukaz 5

ukaz 6



- Če je pogoj pri skočnem ukazu izpolnjen, se ukaza 2 in 3, ki sledita skoku, ne smeta izvesti.

urine periode	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
ukaz 1	IF_1	ID_1	EX_1	MA_1	WB_1				
ukaz 2		~~IF_2~~	~~ID_2~~						
ukaz 3			~~IF_3~~						
ukaz 5				IF_5	ID_5	EX_5	MA_5	WB_5	
ukaz 6					IF_6	ID_6	EX_6	MA_6	WB_6

- To je kontrolna nevarnost.

- Najenostavneje se kontrolna nevarnost reši z vstavitvijo dveh mehurčkov, kar je enako čakanju dve urini periodi.

urine periode	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
ukaz 1	IF_1	ID_1	EX_1	MA_1	WB_1				
ukaz 2		O	O	O	O	O			
ukaz 3			O	O	O	O	O		
ukaz 5				IF_5	ID_5	EX_5	MA_5	WB_5	
ukaz 6					IF_6	ID_6	EX_6	MA_6	WB_6

- Taka rešitev se imenuje skočna zakasnitev (branch delay).

- Zaradi skočnih zakasnitev se CPI precej poveča, zato je treba najti boljše rešitve.
- Preverjanje pogoja za skok se mora izvajati čim bližje prvi stopnji cevovoda, enako velja za izračun skočnega naslova (v našem primeru v 3. stopnji, zato zakasnitev 2 urini periodi).
- Dve vrsti rešitev z napovedovanjem (predikcijo) izpolnitve pogoja za skok:
  - Statična predikcija
  - Dinamična predikcija

Statična predikcija z zakasnjenimi skoki

- Prevajalnik ali procesor skuša vnaprej napovedati izid preverjanja pogoja za skok.
- Med izvrševanjem programa se ta napoved ne more več spreminjati, zato ime statična predikcija.
- Primeri statične predikcije:
  - Pri predpostavki, da pogoj ne bo izpolnjen, se prevzameta in izvršujeta ukaza za skočnim ukazom. Če predpostavka drži, sta ukaza prava in se izvršita do konca, sicer se zavržeta.

- Rešitev pri procesorju PowerPC:
  - Če je odmik v skočnem ukazu negativen (skok nazaj⇒zanka), procesor predpostavi, da bo pogoj izpolnjen.
  - Če je odmik pozitiven, predpostavi, da pogoj ne bo izpolnjen in se skok ne bo izvršil.

- Pri rešitvi z zakasnjenimi skoki cevovod deluje tako, da se ne glede na pogoj vedno izvršijo vsi prevzeti ukazi in čakanje ni potrebno.

- Za ukaze, ki so v programu takoj za skočnim ukazom, pravimo da so v skočnih režah.

- Število skočnih rež je enako številu stopenj, ki so v cevovodu pred stopnjo, ki vpiše skočni naslov v PC (v našem primeru imamo dve skočni reži).
- Ukazi, ki so za skočnim ukazom v skočnih režah, se izvršijo tudi pri izpolnjenem pogoju za skok.
- Zato je treba v skočne reže vstaviti take ukaze, ki se v programu izvršijo ne glede na izid skoka in tudi ne vplivajo na skočni pogoj.
- Ime zakasnjen skok izhaja iz tega, ker je videti kot da so skočni ukazi zakasnjeni za toliko, kot je število skočnih rež.

Dinamična predikcija in skočni predpomnilnik

- Napoved izida preverjanja pogoja se med izvrševanjem programa spreminja v skladu z obnašanjem skočnega ukaza.

- Prediktorska tabela je najpreprostejša vrsta dinamične predikcije.
  - Prediktorska tabela je majhen pomnilnik v katerem so shranjeni izidi preverjanja pogoja pri predhodnih skokih.
  - Do pomnilnika se dostopa s spodnjimi biti naslova na katerem je skočni ukaz in se za vsak izvršen skočni ukaz vpiše izid skoka

- Pri prevzemu skočnega ukaza v stopnji IF se iz prediktorske tabele prebere rezultat pogoja pri zadnjem izvajanju tega skočnega ukaza in uporabi za napoved izida skoka

- Vrste prediktorskih tabel:
  - Enobitni prediktorji
  - Dvo ali večbitni prediktorji
  - Korelacijski prediktorji

- Predikcija je koristna, če imamo hkrati znan tudi skočni naslov.
- Rešitev je skočni predpomnilnik, v katerem so shranjeni podatki o zadnjih skokih, pri katerih je bil pogoj za skok izpolnjen.
- Skočni predpomnilnik in prediktorske tabele so del skočne enote (branch unit), ki v procesorjih z več funkcijskimi enotami običajno računa tudi skočni naslov.

8.3 Kako doseči CPI < 1

- S cevovodno CPE in z odpravljanjem cevovodnih nevarnosti lahko dosežemo CPI, ki je blizu 1.
- Če želimo CPI zmanjšati pod 1, moramo v vsaki urini periodi prevzeti več ukazov (in jih tudi izvesti).
- Take procesorje označujemo z izrazom več-izstavitveni procesorji in jih delimo na dve vrsti:
  - Superskalarni procesorji
  - VLIW procesorji

- Razlika med obema rešitvama je v načinu prevzemanja in izstavljanja ukazov.
- Izstavljanje ukaza imenujemo prehod ukaza iz stopnje ID v stopnjo EX.
- Ukaza, ki je bil izstavljen, ni več mogoče preprosto izničiti oziroma zavreči.
- Prevzemanje in izstavljanje ukazov sta ločeni operaciji, ki se lahko izvajata statično ali dinamično.

- Ukazi se pri prevzemanju iz pomnilnika lahko berejo v različnem vrstnem redu, kar imenujemo razvrščanje ukazov (scheduling).
  - Statično razvrščanje:
    - Procesor prevzema ukaze v takem vrstnem redu, kot so v programu
    - Spreminjanje vrstnega reda ukazov, če je to možno, opravlja prevajalnik
  - Dinamično razvrščanje:
    - Procesor lahko prevzema ukaze v drugačnem vrstnem redu kot so v programu

- Kadar je neka enota prosta, procesor išče ukaze, ki bi jih bilo možno poslati v izvrševanje
- Pri tem je nujno uporabljati tudi dinamično predikcijo skokov
- Ker predikcija ni nujno vedno pravilna, ni zanesljivo ali bo tako prevzet in izveden ukaz potreben.
- Dinamično razvrščanje skupaj z dinamično predikcijo skokov se zato imenuje tudi špekulativno izvrševanje (speculative execution)

- Podobno velja tudi za izstavljanje ukazov (issue)

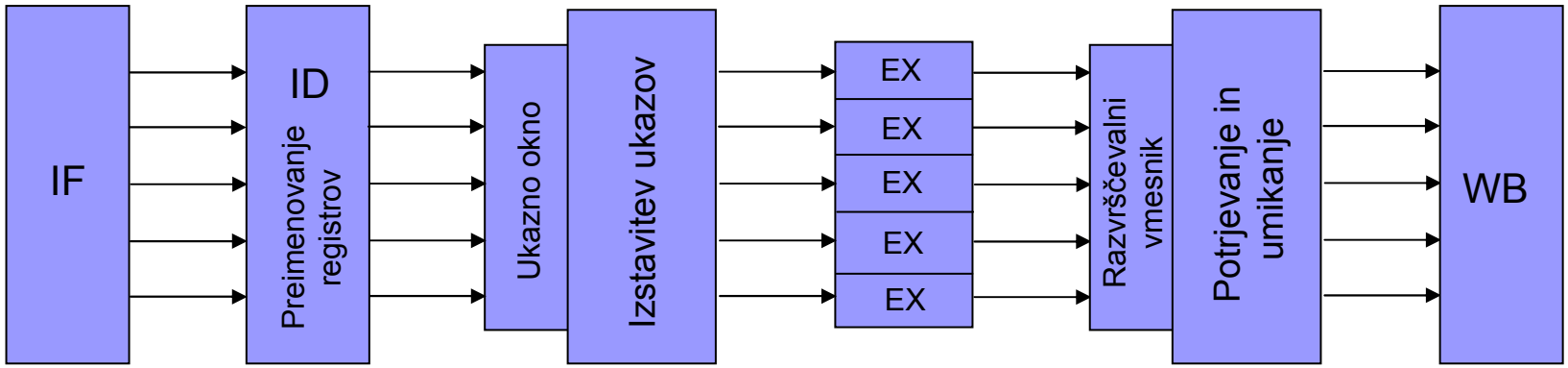
- Statično izstavljanje:
  - Vrstni red izvrševanja ukazov (njihovega izstavljanja v stopnjo EX) je določen s prevzemanjem ukazov.
  - Procesor vrstnega reda ne spreminja.

- Dinamično izstavljanje:
  - Vrstni red izstavljanja prevzetih ukazov določa logika v procesorju
  - Procesor pregleduje ukaze, ki so naprej od trenutne točke izvrševanja in išče take, ki niso odvisni od ukazov, ki se trenutno izvršujejo.
  - Če tak ukaz najde, ga izstavi v izvrševanje

- Pri takem špekulativnem izvrševanju ukazov se rezultati ukazov ne smejo shraniti v programsko dostopne registre ali pomnilnik, dokler ni jasno ali je bila predikcija skokov po kateri so bili ukazi prevzeti, pravilna.

- Pri špekulativnem izvrševanju ukazov se ukazi lahko izvršujejo v drugačnem vrstnem redu, shranjevanje rezultatov pa je vedno v enakem vrstnem redu, kot je določen z ukazi v programu.

- Superskalarni procesor je cevovodni procesor, ki je sposoben hkrati prevzemati, dekodirati izvrševati in shranjevati več ukazov.
- Pri superskalarnih procesorjih procesor dinamično določa vrstni red prevzemanja in izstavljanja ukazov.
- Istočasno delovanje zahteva dodatne vmesnike in dodatne stopnje za potrjevanje (zavezovanje) in umikanje rezultatov.



Peonostavljena shema superskalarnega procesorja,
ki ima za osnovo petstopenjski cevovod

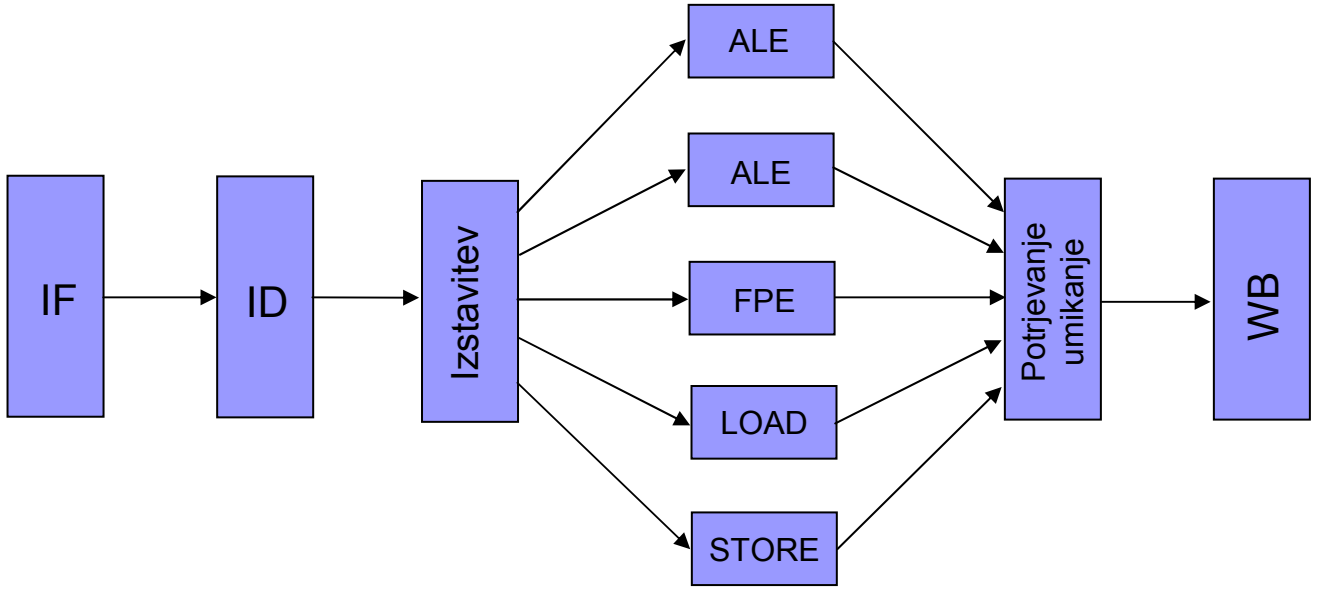
- Ena od funkcijskih enot v stopnji EX je stopnja MA (funkcijska enota LOAD/STORE ali ločeni funkcijski enoti LOAD in STORE).

- VLIW (Very Long Instruction Word) procesorji izvršujejo dolge ukaze, ki so sestavljeni iz več običajnih strojnih ukazov, ki jih procesor lahko paralelno izvršuje v različnih funkcijskih enotah.
- V dolgem ukazu izvršuje vsaka funkcijska enota svoj ukaz.

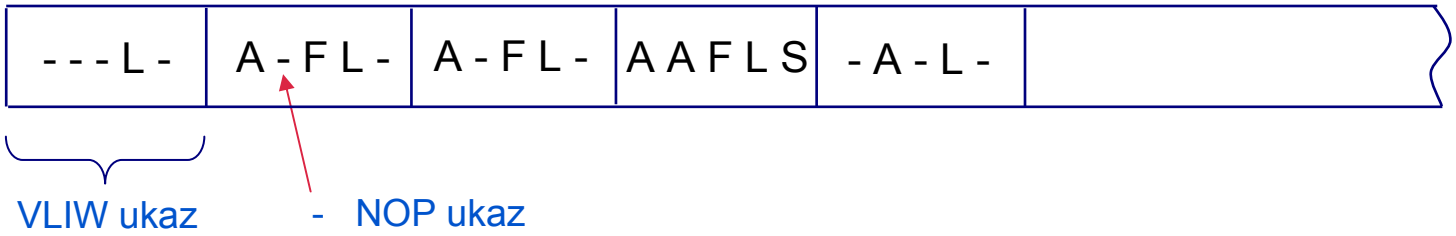
VLIW ukaz sestavljajo ukazi za posamezne funkcijske enote

- Prevajalnik v programu išče med seboj neodvisne ukaze za funkcijske enote in z njimi sestavlja dolge ukaze.
- Če ne najde dovolj ukazov za vse funkcijske enote, da enoti, za katero ni našel ukaza, ukaz NOP.

Cevovodna CPE - VLIW procesor



Zaporedje VLIW ukazov



Superskalarni procesorji Intel Pentium

- Koncept superskalarnega procesorja se običajno povezuje z RISC procesorji, vendar se enaki principi lahko uporabijo tudi pri CISC procesorjih.
- Primer takega procesorja je Intel Pentium.
  - Procesor Intel80486 (1989) je bil še običajni CISC procesor brez superskalarnih elementov.
  - Intel Pentium (1993) je prvi, ki je imel superskalarno zasnovo, z dvema cevovodnima izvršilnima enotama (prvič MMX ukazi - SIMD operacije nad 64-bitnimi pakiranimi celoštevilčnimi operandi)

Družina procesorjev P6 (1995 - 1999)

- Pentium Pro (1995) je imel že šest funkcijskih enot in je lahko izvrševal povprečno tri ukaze v urini periodi:
  - Dve celoštevilski ALE
  - Enota za operaracije v plavajoči vejici
  - Enota za skoke
  - Enoti LOAD in STORE
- Pentium II (MMX ukazi)
- Pentium II Xeon
- Celeron
- Pentium III (SSE ukazi - SIMD operacije nad pakiranimi 128-bitnimi operandi v plavajoči vejici z enojno natančnostjo)
- Pentium III Xeon

- Naslednji Pentium procesorji imajo še izboljšano in razširjeno superskalarno zasnovo
  - Pentium 4 (2000 $\Rightarrow$ 2006)
  - Xeon (2001 $\Rightarrow$ 2006)
  - Pentium M (2003 $\Rightarrow$)
  - Pentium Extreme Edition (2005 $\Rightarrow$ 2007)
  - Pentium D (2007 $\Rightarrow$)
  - Core Duo in Core Solo (2006 $\Rightarrow$)
  - Xeon Processor 5100 (2006 $\Rightarrow$)
  - Core 2 (2006 $\Rightarrow$)

Superskalarni procesor Intel Pentium 4

- Procesor prevzema strojne CISC ukaze iz (64 bajtov hkrati) predpomnilnika L2 v enakem vrstnem redu kot so v programu.
- Vsak ukaz se prevede v enega do štiri 118-bitne RISC ukaze, ki jih pri Intelu imenujejo mikro-operacije (μ-op).
- Procesor te mikro-operacije izvaja kot superskalarni cevovodni procesor - špekulativno izvrševanje.

- Procesor rezultate izvršenih mikro-operacij potrjuje in shranjuje v programsko dostopne registre v takem vrstnem redu kot so v programu.
- Pentium 4 ima 20-stopenjski cevovod (Pentium 5-stopenjski), zato je zakasnitev pri napačni napovedi izida skočnih ukazov zelo velika.
- Procesor ima zato izpopolnjeno enoto za napovedovanje skokov s 4-bitno prediktorsko tabelo.

- Ukazi, ki bi se prevedli v več kot štiri mikro-operacije, se v mikroprogramski kontrolni enoti (mikroprogramski ROM) prevedejo v zaporedje mikro-operacij.

Cevovodna CPE - superskalarni procesor Intel Pentium 4