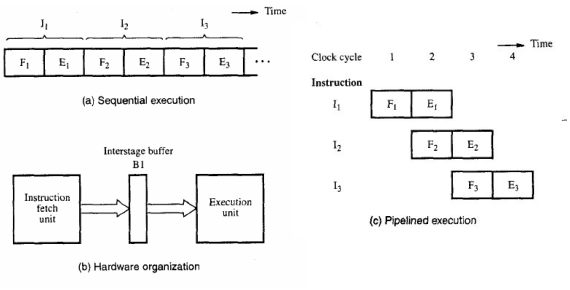
Protočnost (Preklapanje)

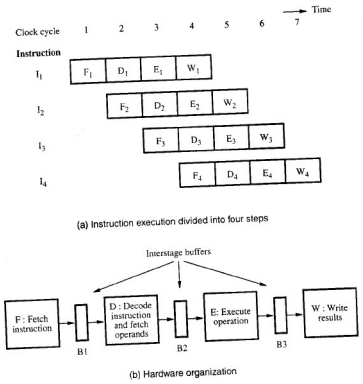
Vladimir Filipović
vladaf@matf.bg.ac.yu

Osnovni pojmovi



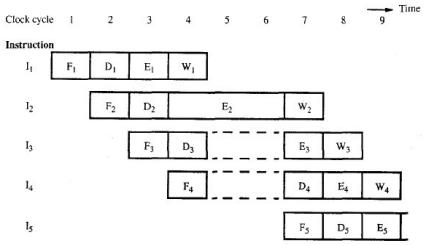
Osnovna ideja protočnosti instrukcija

Osnovni pojmovi



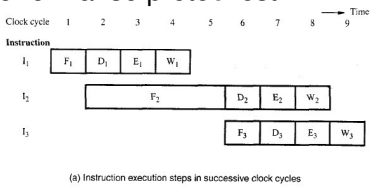
Protočnost sa četiri nivoa

Performanse protočnosti

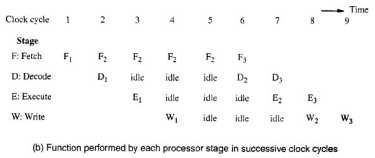


Efekti izvršavanja operacije koja zahteva više od jednog ciklusa časovnika

Performanse protočnosti



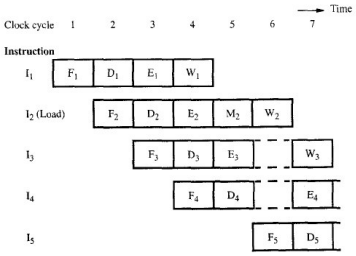
(a) Instruction execution steps in successive clock cycles



(b) Function performed by each processor stage in successive clock cycles

Zaustavljanje protočnosti koje je prouzrokovano promašajem keša u F_2

Performanse protočnosti



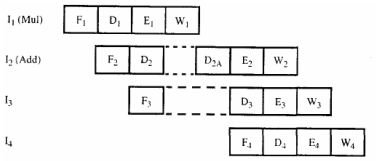
Load X(R1),R2

Efekat instrukcije Load na vremensko sekvenciranje protočnosti

Rizici podataka

$A \leftarrow 3 + A$
$B \leftarrow 4 \times A$

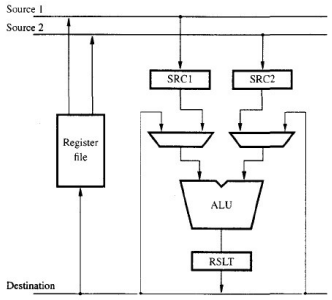
$A \leftarrow 5 \times C$
$B \leftarrow 20 + C$



Protočnost je zaustavljena zbog zavisnosti podataka između D_2 i W_1

```
Mul   R2,R3,R4
Add   R5,R4,R6
```

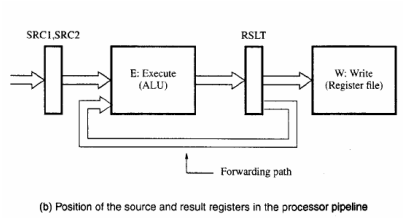

Prosleđivanje operatora



(a) Datapath

Prosleđivanje operatora kod procesora sa protočnošću

Prosleđivanje operatora



(b) Position of the source and result registers in the processor pipeline

Prosleđivanje operatora kod procesora sa protočnošću

Rukovanje sa rizicima podataka u softveru

```
I1:  Mul   R2,R3,R4
     NOP
     NOP
I2:  Add   R5,R4,R6
```

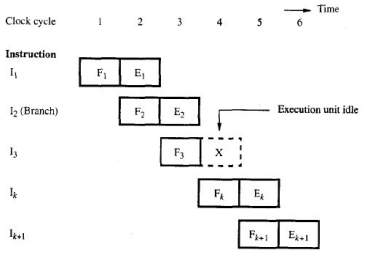

Bočni efekti

```
Add            R1,R3
AddWithCarry   R2,R4
```

$R4 \leftarrow [R2] + [R4] + carry$

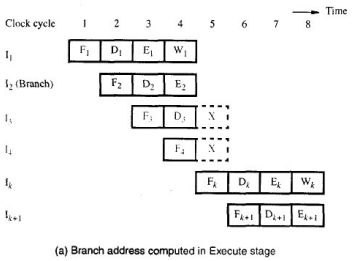
Rizici instrukcija

Bezuslovni skokovi



Prazan ciklus koji prouzrokuje instrukcija skoka

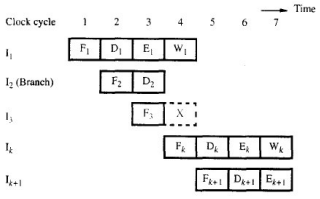
Bezuslovni skokovi



(a) Branch address computed in Execute stage

Vremensko sekvenciranje za skokove

Bezuslovni skokovi

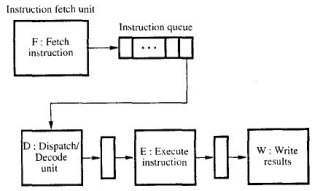


(b) Branch address computed in Decode stage

Vremensko sekvenciranje za skokove

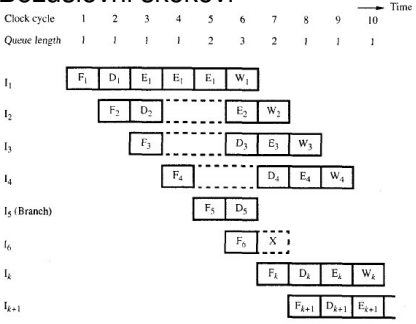
Bezuslovni skokovi

Red instrukcija i pred-dohvatanje



Korišćenje reda instrukcija u hardverskoj organizaciji procesora sa protočnošću

Bezuslovni skokovi



Vremensko sekvenciranje skoka pri postojanju reda instrukcija.
Ciljna adresa skoka se izračunava u fazi dekodiranja instrukcije.

Uslovni skokovi i predviđanje skoka

LOOP	Shift_left	R1
	Decrement	R2
	Branch=0	LOOP
NEXT	Add	R1,R3

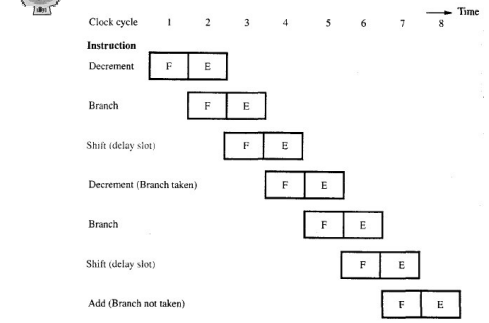
(a) Original program loop

LOOP	Decrement	R2
	Branch=0	LOOP
	Shift_left	R1
NEXT	Add	R1,R3

(b) Reordered instructions

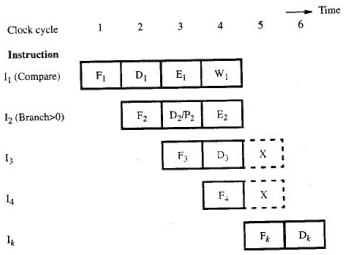
Preuređenje redosleda instrukcija radi kašnjenja skoka.

Uslovni skokovi i predviđanje skoka



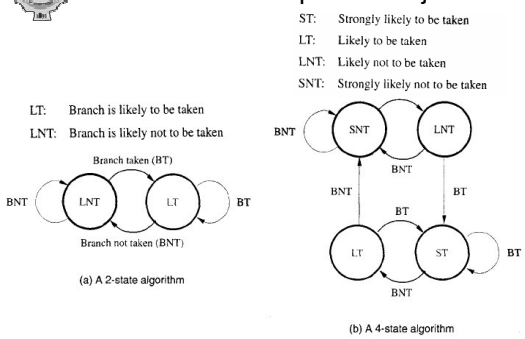
Vremenska sekvenca izvršavanja pokazuje da se slot za kašnjenje puni tokom dve poslednje faze prolaska kroz petlju koju obrazuju preuređene instrukcije

Uslovni skokovi i predviđanje skoka



Vremenska sekvenca izvršavanja kada je odluka o skoku nekorektno predviđena I to da se skok ne izvršava

Uslovni skokovi i predviđanje skoka



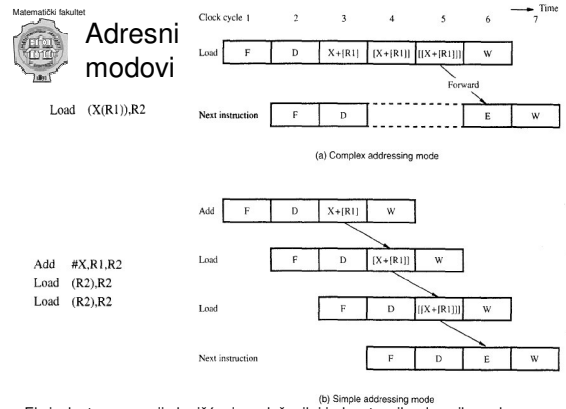
(a) A 2-state algorithm

(b) A 4-state algorithm

Reprezentacija algoritma za predviđanje skoka pomoću konačnog automata

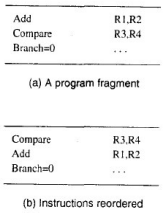
Uticaj skupova instrukcija

Adresni modovi



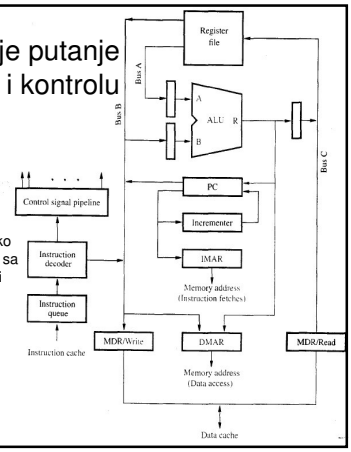
Ekvivalentne operacije korišćenjem složenih i jednostavnih adresnih modova

Uslovni kodovi



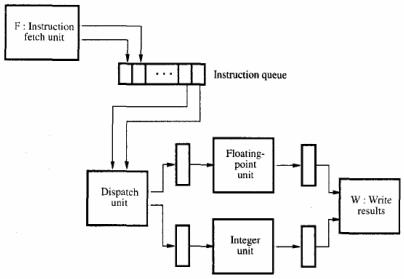
Preuređivanje redosleda instrukcija

Razmatranje putanje za podatke i kontrolu



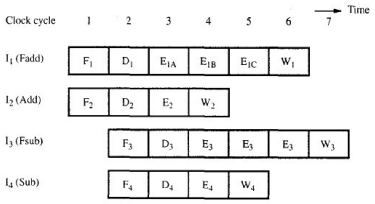
Modifikovanje putanje podataka tako da odgovara protočnom izvršavanju sa baferima za međufaze kod ulaza i izlaza ALU

Superskalarne operacije



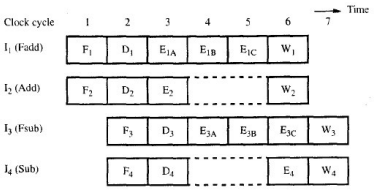
Procesor sa dve jedinice za izvršavanje

Superskalarne operacije



Primer toka izvršavanja instrukcija na prethodno prikazanom procesoru, uz pretpostavku da nije došlo do nikakvog rizika

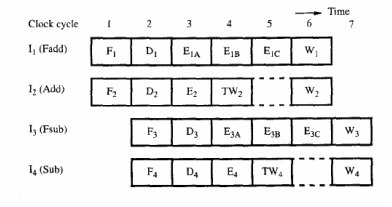
Kompletiranje izvršavanja



(a) Delayed write

Kompletiranje izvršavanja instrukcije u redosledu programiranja

Kompletiranje izvršavanja



(b) Using temporary registers

Kompletiranje izvršavanja instrukcije u redosledu programiranja

SPARC

Instruction		Description
ADD	R5, R6, R7	Integer add: R7 ← [R5] + [R6]
ADDcc	R2, R3, R5	R5 ← [R2] + [R3], set condition code flags
SUB	R5, Imm, R7	Integer subtract: R7 ← [R5] − Imm(sign-extended)
AND	R3, Imm, R5	Bitwise AND: R5 ← [R3] AND Imm(sign-extended)
XOR	R3, R4, R5	Bitwise Exclusive OR: R5 ← [R3] XOR [R4]
FADDq	F4, F12, F16	Floating-point add, quad precision: F12 ← [F4] + [F12]
FSUBs	F2, F5, F7	Floating-point subtract, single precision: F7 ← [F2] − [F5]
FDIVs	F5, F10, F18	Floating-point divide, single precision, F18 ← [F5]/[F10]
LDSW	R3, R5, R7	R7 ← 32-bit word at [R3] + [R5] sign extended to a 64-bit value
LDX	R3, R5, R7	R7 ← 64-bit extended word at [R3] + [R5]
LDUB	R4, Imm, R5	Load unsigned byte from memory location [R4] + Imm, the byte is loaded into the least significant 8 bits of register R5, and all higher-order bits are filled with 0s
STW	R3, R6, R12	Store word from register R3 into memory location [R6] + [R12]
LDF	R5, R6, F3	Load a 32-bit word at address [R5] + [R6] into floating-point register F3
LDDF	R5, R6, F8	Load doubleword (two 32-bit words) at address [R5] + [R6] into floating-point registers F8 and F9
STF	F14, R6, Imm	Store word from floating-register F14 into memory location [R6] + Imm
BLE	icc, Label	Test the icc flags and branch to Label if less than or equal to zero
BZ,pn	xcc, Label	Test the xcc flags and branch to Label if equal to zero, branch is predicted not taken
BGT,a,pt	icc, Label	Test the 32-bit integer condition codes and branch to Label if greater than zero, set annul bit, branch is predicted taken
FBNE,pn	Label	Test floating-point status flags and branch if not equal, the annul bit is set to zero, and the branch is predicted not taken

Primeri SPARC instrukcija

SPARC

```
                LDX     R3, 0, R6       Load number of items in the list.
                OR      R0, R0, R4      R4 to be used as offset in the list
                OR      R0, R0, R7      Clear R7 to be used as accumulator.
LOOPSTART       LDX     R3, R4, R5      Load list item into R5.
                ADD     R5, R7, R7      Add number to accumulator.
                ADD     R4, 8, R4       Point to the next entry.
                SUBcc   R6, 1, R6       Decrement R6 and set condition flags.
                BG      xec, LOOPSTART  Loop if more items in the list.
NEXT            ...
```

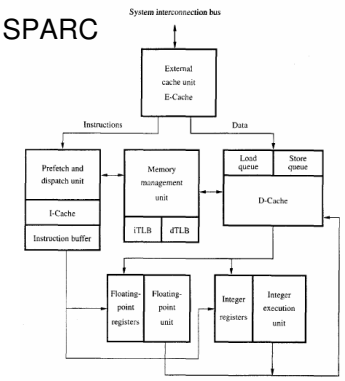
(a) Desired program loop

```
                LDX     R3, 0, R6
                OR      R0, R0, R4
                OR      R0, R0, R7
LOOPSTART       LDX     R3, R4, R5
                ADD     R4, 8, R4
                SUBcc   R6, 1, R6
                BG,pt   xec, LOOPSTART  Predicted taken, Annul bit = 0
                ADD     R5, R7, R7
NEXT            ...
```

(b) Instructions reorganized to use the delay slot

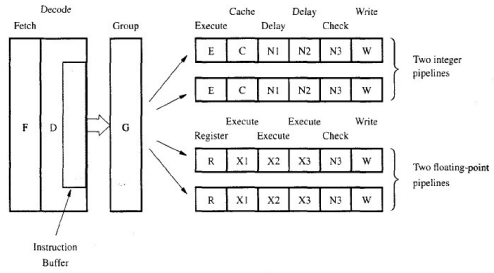
Petlja za sabiranje koja prikazuje korišćenje kašnjenja pri skoku i predviđanja skoka

SPARC



Glavni gradivni blokovi procesora UltraSPARC II

SPARC



Organizacija protočnosti kod procesora UltraSPARC II

SPARC

```
                ADDcc   R3, R4, R7      R7 ← [R3] + [R4],
                                        Set condition codes
                BRZ,a   Label           Branch if zero, set Annul bit to 1
                FCMP    F1, F5          FP: Compare [F2] and [F5]
                FADD    F2, F3, F6      FP: F6 ← [F2] + [F3]
                FMOVs   F3, F4          Move single precision operand from F3 to F4
                :
Label           FSUB    F2, F3, F6      FP: F6 ← [F2] - [F3]
                LDSW    R3, R4, R7      Load single word at location [R3] + [R4] into R7
                :
```

(a) Program fragment

```
ADDcc   R3, R4, R7
BRZ,a   Label
FCMP    F1, F5
FSUB    F2, F3, F6
```

(b) Instruction grouping, branch taken

```
ADDcc   R3, R4, R7
BRZ,a   Label
FCMP    R1, R5
FADD    R2, R3, R6
```

(c) Instruction grouping, branch not taken

Primeri grupisanja instrukcija

SPARC

```
ADD     R3, R5, R6      G   E   C   N1  N2  N3  W
LDSW    R4, R7, R6          G   E   C   N1  N2  N3  W
```

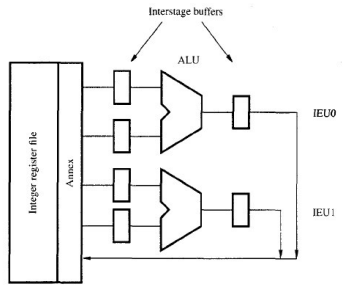
(a) Instructions with common destination

```
MOVRZ   R1, R6, R7      G   E   C   N1  N2  N3  W
OR      R7, R8, R9              G   E   C   N1  N2  N3  W
```

(b) Delay caused by MOVR instruction

Kašnjenja u raspoređivanju zbog rizika

SPARC



Jedinica za izvršavanje nad celobrojnim podacima

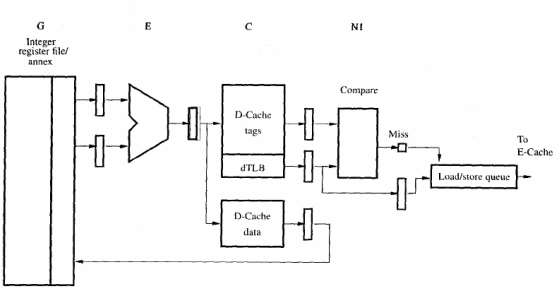
SPARC

I_1 (Icc)	G	E	C
I_2 (BRcc)	G	E	C
I_3	G	E	C
I_4	G	E	C
I_5		G	E
I_6		G	E
I_7		G	E
I_8		G	E
I_9			G
I_{10}			G
I_{11}			G
I_{12}			G

↑ Abort

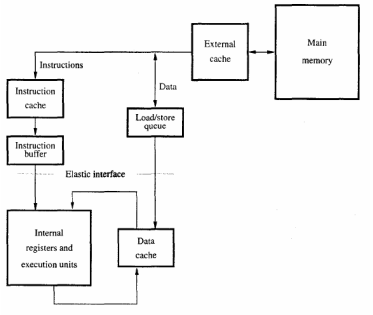
Vremensko sekvenciranje za najgori slučaj nekorektno predviđenog skoka

SPARC



Jedinica za učitavanje i čuvanje

SPARC



Tok izvršavanja

Efekti rizika instrukcija

$$T = \frac{N \times S}{R}$$

$$P_s = R/S$$

$P_p = R = 500$ MIPS (million instructions per second)

$$\delta_{miss} = ((1 - h_i) + d(1 - h_d)) \times M_p$$

$$\delta_{miss} = (0.05 + 0.3 \times 0.1) \times 17 = 1.36 \text{ cycles}$$

$$P_p = \frac{R}{T_t} = \frac{R}{1 + \delta_{miss}} = 0.42R$$

$$P_s = \frac{R}{4 + \delta_{miss}} = 0.19R$$

$$\delta_{miss} = ((1 - h_i) + d(1 - h_d)) \times (h_s \times M_s + (1 - h_s) \times M_p) = 0.46 \text{ cycle}$$

Zadaci

-
-
-