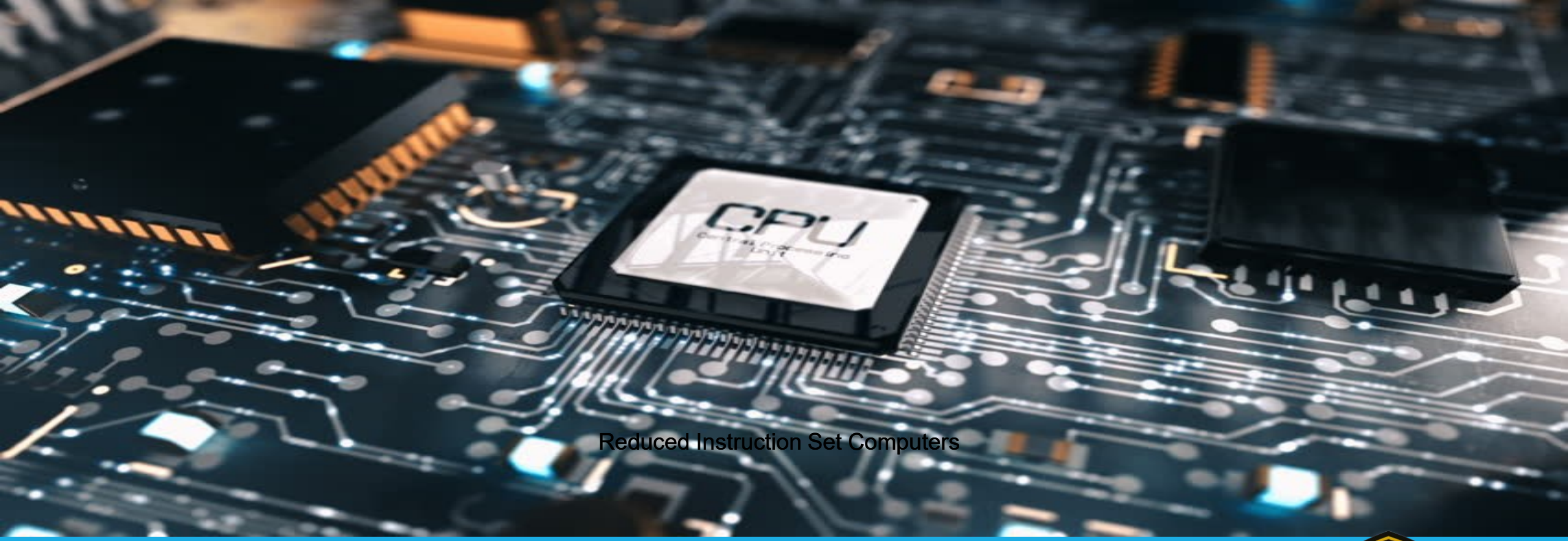


Reduced Instruction Set Computers

Arsitektur dan Organisasi Komputer
COM 60011

Bab #15 – Reduced Instruction Set Computers (RISC)
Complex Instruction Set Computers (CISC)

Tabel 15.1
Karakteristik dari Beberapa CISC, RISC, dan Prosesor Superscalar

	Complex Instruction Set (CISC)Computer			Reduced Instruction Set (RISC) Computer	
Characteristic	IBM 370/168	VAX 11/780	Intel 80486	SPARC	MIPS R4000
Year developed	1973	1978	1989	1987	1991
Number of instructions	208	303	235	69	94
Instruction size (bytes)	2–6	2–57	1–11	4	4
Addressing modes	4	22	11	1	1
Number of general-purpose registers	16	16	8	40 - 520	32
Control memory size (kbits)	420	480	246	—	—
Cache size (kB)	64	64	8	32	128

(Tabel dapat ditemukan di halaman 538 di buku teks.)

Tabel 15.1
Karakteristik dari Beberapa CISC, RISC, dan Prosesor Superscalar

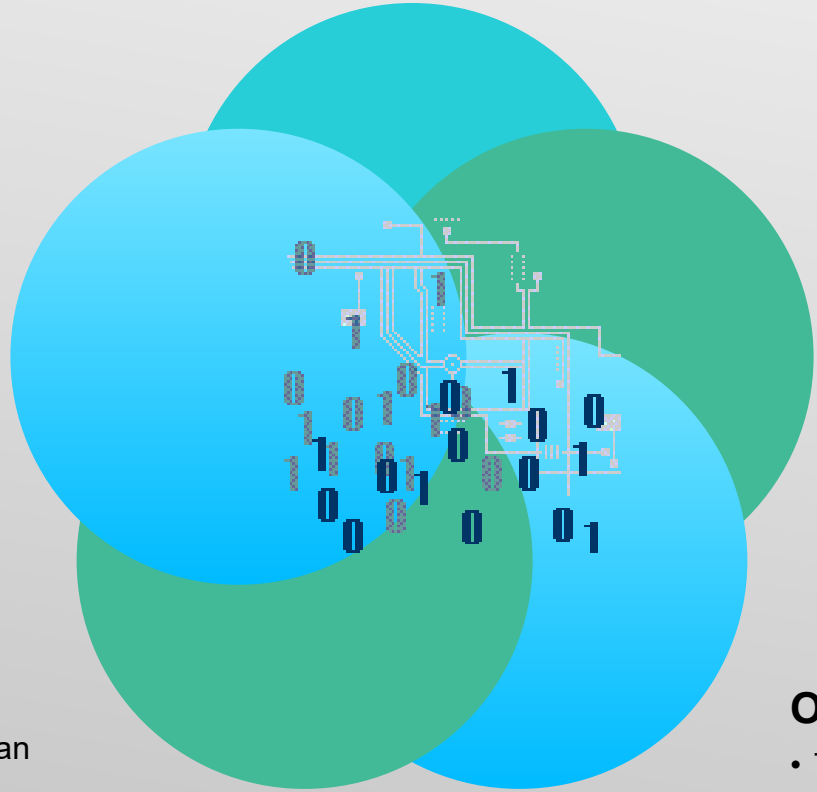
	Superscalar		
Characteristic	PowerPC	Ultra SPARC	MIPS R10000
Year developed	1993	1996	1996
Number of instructions	225		
Instruction size (bytes)	4	4	4
Addressing modes	2	1	1
Number of general-purpose registers	32	40 - 520	32
Control memory size (kbits)	—	—	—
Cache size (kB)	16-32	32	64

(Tabel dapat ditemukan di halaman 538 di buku teks.)

Karakteristik Eksekusi Instruksi

High-level languages (HLLs)

- Izinkan programmer untuk mengekspresikan algoritme secara lebih ringkas
- Izinkan compiler untuk mengurus detail yang tidak penting dalam ekspresi algoritma programmer
- Seringkali mendukung penggunaan pemrograman terstruktur secara alami dan/atau desain berorientasi objek



Execution sequencing

- Menentukan organisasi kontrol dan saluran pipa

Semantic gap

- Perbedaan antara operasi yang disediakan dalam HLL dan yang disediakan dalam arsitektur komputer

Operands used

- Jenis operan dan frekuensi penggunaannya menentukan organisasi memori untuk menyimpannya dan mode mengatasinya untuk mengaksesnya

Operations performed

- Tentukan fungsi yang akan dilakukan oleh prosesor dan interaksinya dengan memori

Tabel 15.2
Tertimbang Frekuensi Dinamis Relatif Operasi HLL [PATT82a]

	Dynamic Occurrence		Machine-Instruction Weighted		Memory-Reference Weighted	
	Pascal	C	Pascal	C	Pascal	C
ASSIGN	45%	38%	13%	13%	14%	15%
LOOP	5%	3%	42%	32%	33%	26%
CALL	15%	12%	31%	33%	44%	45%
IF	29%	43%	11%	21%	7%	13%
GOTO	—	3%	—	—	—	—
OTHER	6%	1%	3%	1%	2%	1%

(Tabel dapat ditemukan di halaman 540 di buku teks.)

Tabel 15.3
Dinamis Persentase Operand

	Pascal	C	Average
Integer Constant	16%	23%	20%
Scalar Variable	58%	53%	55%
Array/Structure	26%	24%	25%

Tabel 15.4
Argumen Prosedur dan Variabel Skalar Lokal

Percentage of Executed Procedure Calls With	Compiler, Interpreter, and Typesetter	Small Nonnumeric Programs
>3 arguments	0–7%	0–5%
>5 arguments	0–3%	0%
>8 words of arguments and local scalars	1–20%	0–6%
>12 words of arguments and local scalars	1–6%	0–3%

(Tabel dapat ditemukan di halaman 541 di buku teks.)

Implikasi

HLL paling baik dapat didukung dengan mengoptimalkan kinerja fitur yang paling memakan waktu dari program HLL tipikal

Tiga elemen yang menjadi ciri arsitektur RISC:

Gunakan register dalam jumlah besar atau gunakan kompiler untuk mengoptimalkan penggunaan register

Perhatian yang cermat perlu diberikan pada desain pipa instruksi

Instruksi harus memiliki biaya yang dapat diprediksi dan konsisten dengan implementasi berkinerja tinggi

Penggunaan Register Besar Mengajukan

Solusi Perangkat Lunak

Membutuhkan kompiler untuk mengalokasikan register

Alokasikan berdasarkan variabel yang paling banyak digunakan dalam waktu tertentu

Membutuhkan program yang canggih analisis

Solusi Perangkat Keras

Lebih banyak register

Dengan demikian lebih banyak variabel akan berada di register

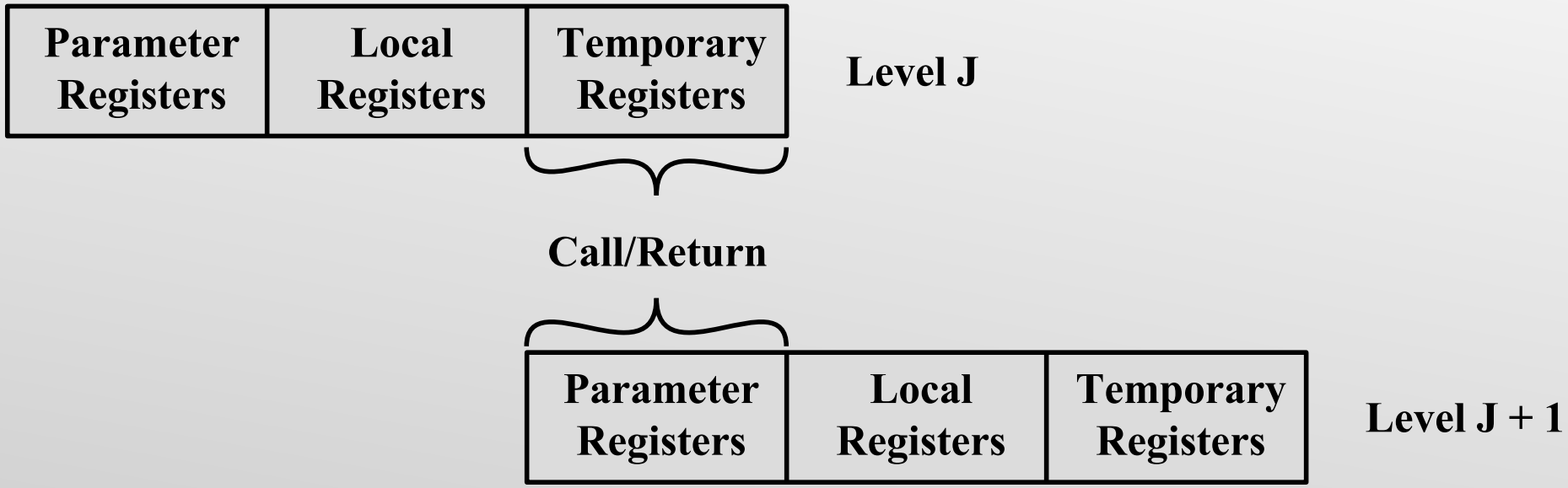


Figure 15.1 Overlapping Register Windows

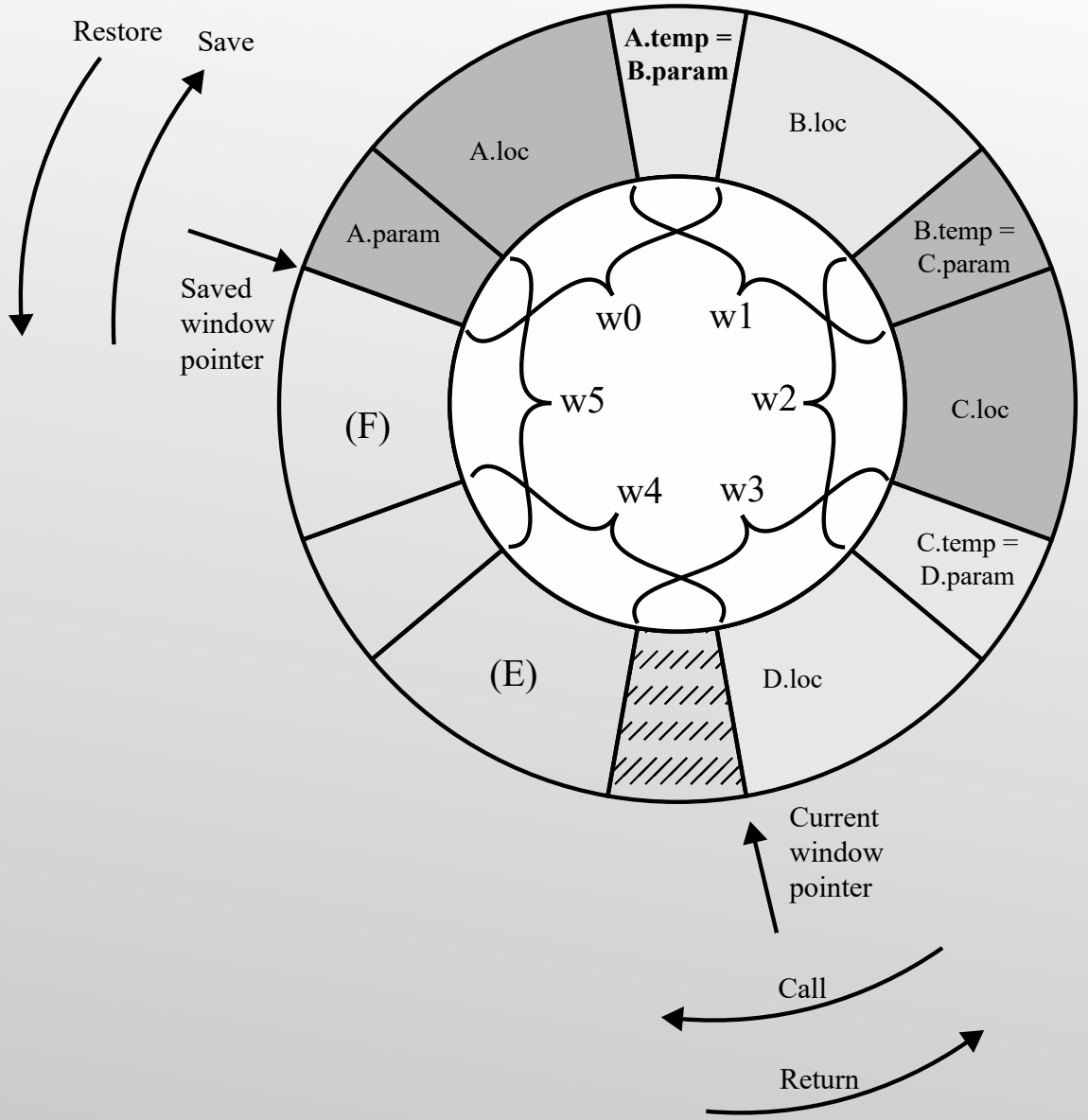


Figure 15.2 Circular-Buffer Organization of Overlapped Windows

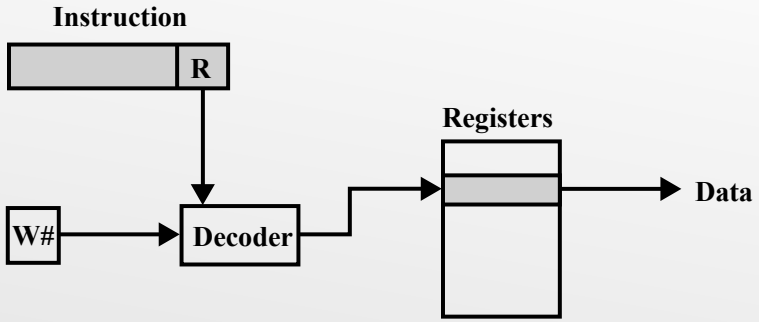
Variabel Global

- **Variabel yang dideklarasikan sebagai global dalam HLL dapat diberi lokasi memori oleh compiler dan semua instruksi mesin yang mereferensikan variabel ini akan menggunakan operan referensi memori**
  - Namun, untuk variabel global yang sering diakses, skema ini tidak efisien
- Alternatifnya adalah memasukkan satu set register global ke dalam prosesor
  - Register-register ini akan tetap jumlahnya dan tersedia untuk semua prosedur
  - Skema penomoran terpadu dapat digunakan untuk menyederhanakan format instruksi
- Ada peningkatan beban perangkat keras untuk mengakomodasi perpecahan dalam pengalamatan register
- Selain itu, linker harus memutuskan variabel global mana yang harus ditetapkan ke register

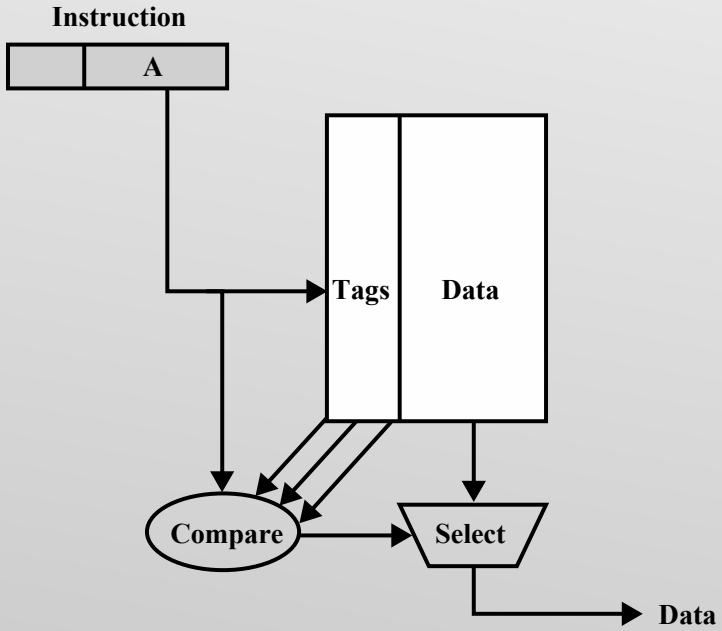
Tabel 15.5
Karakteristik dari Large-Register-File dan Organisasi Cache

Large Register File	Cache
All local scalars	Recently-used local scalars
Individual variables	Blocks of memory
Compiler-assigned global variables	Recently-used global variables
Save/Restore based on procedure nesting depth	Save/Restore based on cache replacement algorithm
Register addressing	Memory addressing
Multiple operands addressed and accessed in one cycle	One operand addressed and accessed per cycle

(Tabel dapat ditemukan di halaman 546 di buku teks.)

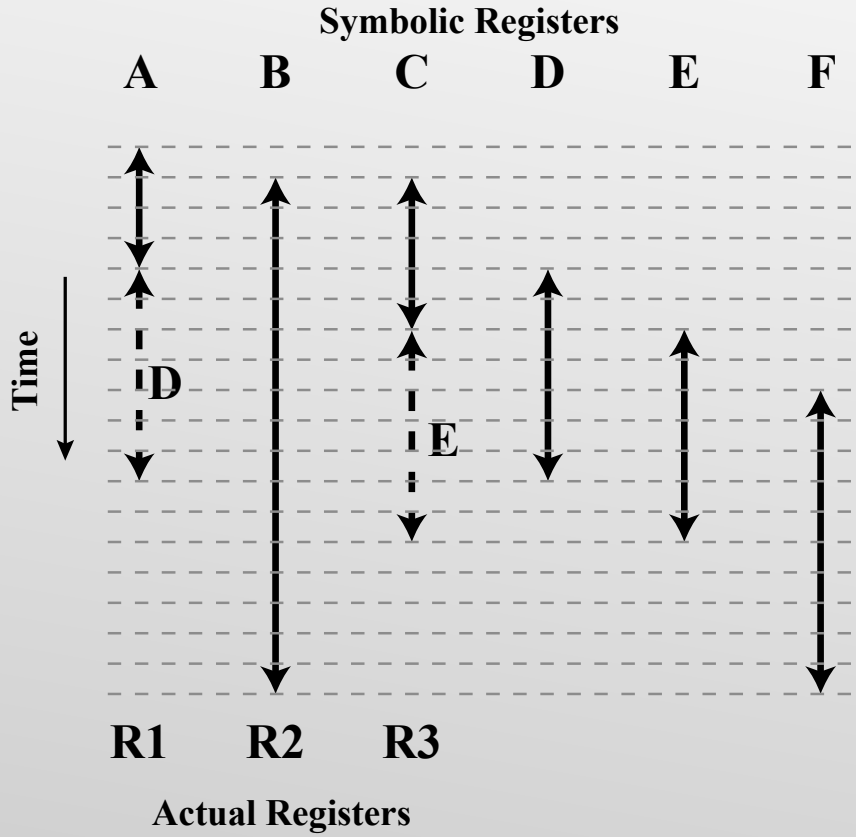


(a) Windows-based register file

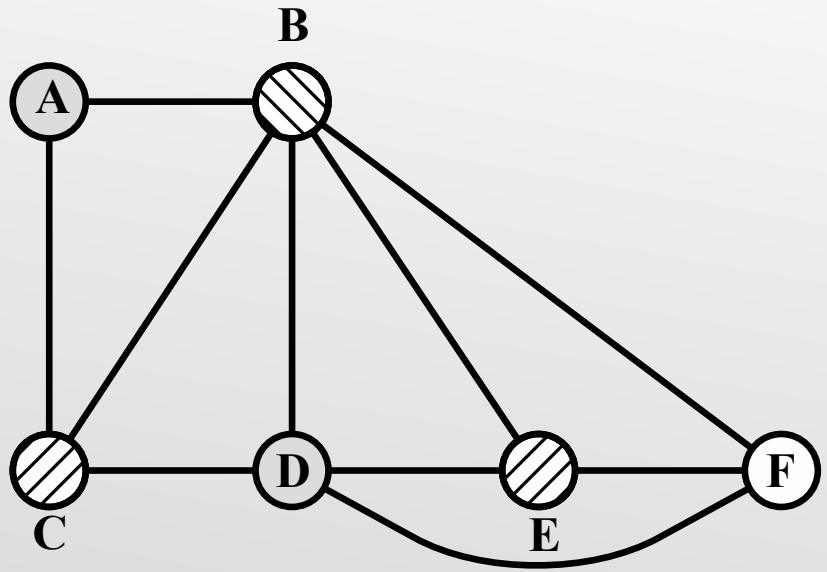


(b) Cache

Figure 15.3 Referencing a Scalar



(a) Time sequence of active use of registers



(b) Register interference graph

Figure 15.4 Graph Coloring Approach

Mengapa CISC?

(Complex Instruction Set Computers)

Ada kecenderungan set instruksi yang lebih kaya yang mencakup jumlah instruksi yang lebih besar dan lebih kompleks

Dua alasan utama untuk tren ini:

- Keinginan untuk menyederhanakan penyusun
- Keinginan untuk meningkatkan kinerja

Ada dua keuntungan untuk program yang lebih kecil:

- Program ini membutuhkan lebih sedikit memori
- Harus meningkatkan kinerja
  - Lebih sedikit instruksi berarti lebih sedikit byte instruksi yang akan diambil
  - Dalam lingkungan paging, program yang lebih kecil menempati lebih sedikit halaman, mengurangi kesalahan halaman
  - Lebih banyak instruksi masuk ke dalam cache

Tabel 15.6
Kode Ukuran Relatif dengan RISC I

	[PATT82a] 11 C Programs	[KATE83] 12 C Programs	[HEAT84] 5 C Programs
RISC I	1.0	1.0	1.0
VAX-11/780	0.8	0.67	
M68000	0.9		0.9
Z8002	1.2		1.12
PDP-11/70	0.9	0.71	

(Tabel dapat ditemukan di halaman 550 di buku teks.)

Karakteristik Arsitektur Set Instruksi Berkurang

One machine instruction per machine cycle	• *Siklus mesin --- waktu yang diperlukan untuk mengambil dua operan dari register, melakukan operasi ALU, dan menyimpan hasilnya dalam register*
Register-to-register operations	• Only simple LOAD and STORE operations accessing memory • Ini menyederhanakan kumpulan instruksi dan oleh karena itu unit kontrol
Simple addressing modes	• Menyederhanakan kumpulan instruksi dan unit kontrol
Simple instruction formats	• Umumnya hanya satu atau beberapa format yang digunakan • Panjang instruksi diperbaiki dan diratakan pada batas kata • Opcode decoding dan register operan accessing dapat terjadi secara bersamaan

8	16	16	16
Add	B	C	A

Memory to memory

I = 56, D= 96, M = 152

8	4	16		
Load	RB	B		
Load	RC	B		
Add	RA	RB	RC	
Store	RA	A		

Register to memory

I = 104, D = 96, M = 200

(a) A □ B + C

8	16	16	16
Add	B	C	A
Add	A	C	B
Sub	B	D	D

Memory to memory

I = 168, D= 288, M = 456

8	4	4	4
Add	RA	RB	RC
Add	RB	RA	RC
Sub	RD	RD	RB

Register to register

I = 60, D = 0, M = 60

(b) A □ B + C; B □ A + C; D □ D – B

I = number of bytes occupied by executed instructions
D = number of bytes occupied by data
M = total memory traffic = I + D

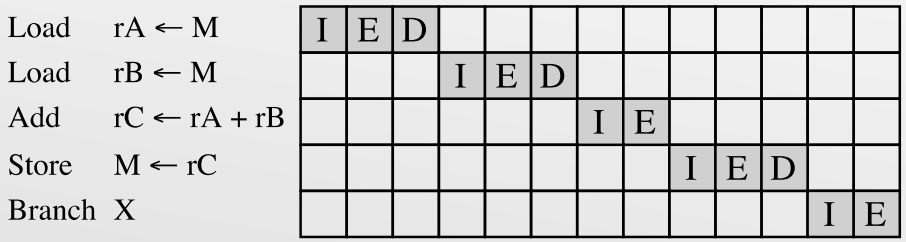
Figure 15.5 Two Comparisons of Register-to-Register and Memory-to-Memory Approaches

Tabel 15.7
Karakteristik dari Beberapa Prosesor

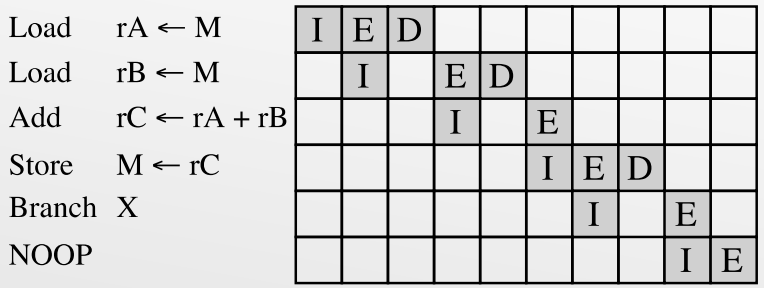
Processor	Number of instruc-tion sizes	Max instruc-tion size in bytes	Number of addressing modes	Indirect addressing	Load/store combined with arithmetic	Max number of memory operands	Unaligned addressing allowed	Max Number of MMU uses	Number of bits for integer register specifier	Number of bits for FP register specifier
AMD29000	1	4	1	no	no	1	no	1	8	3 [a]
MIPS R2000	1	4	1	no	no	1	no	1	5	4
SPARC	1	4	2	no	no	1	no	1	5	4
MC88000	1	4	3	no	no	1	no	1	5	4
HP PA	1	4	10 [a]	no	no	1	no	1	5	4
IBM RT/PC	2 [a]	4	1	no	no	1	no	1	4 [a]	3 [a]
IBM RS/6000	1	4	4	no	no	1	yes	1	5	5
Intel i860	1	4	4	no	no	1	no	1	5	4
IBM 3090	4	8	2 [b]	no [b]	yes	2	yes	4	4	2
Intel 80486	12	12	15	no [b]	yes	2	yes	4	3	3
NSC 32016	21	21	23	yes	yes	2	yes	4	3	3
MC68040	11	22	44	yes	yes	2	yes	8	4	3
VAX	56	56	22	yes	yes	6	yes	24	4	0
Clipper	4 [a]	8 [a]	9 [a]	no	no	1	0	2	4 [a]	3 [a]
Intel 80960	2 [a]	8 [a]	9 [a]	no	no	1	yes [a]	—	5	3 [a]

a RISC that does not conform to this characteristic.
b CISC that does not conform to this characteristic.

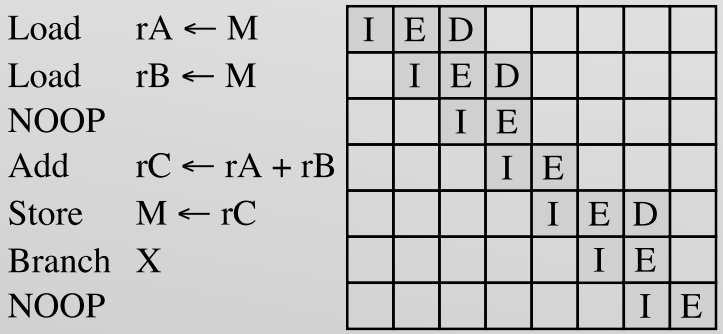
(Tabel dapat ditemukan di halaman 554 di buku teks.)



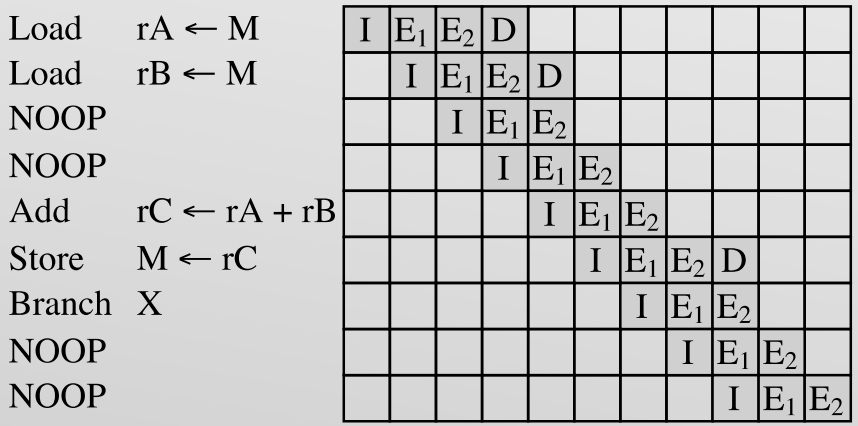
(a) Sequential execution



(b) Two-stage pipelined timing



(c) Three-stage pipelined timing



(d) Four-stage pipelined timing

Figure 15.6 The Effects of Pipelining

Optimasi Pipelining

Cabang tertunda

- Tidak berlaku sampai setelah pelaksanaan instruksi berikut
- Instruksi berikut ini adalah slot penundaan

Beban Tertunda

- Register untuk menjadi target dikunci oleh prosesor
- Lanjutkan eksekusi aliran instruksi sampai register diperlukan
- Diam sampai dimuat selesai
- Menata ulang instruksi dapat memungkinkan pekerjaan yang bermanfaat sementara Memuat

Ulangi Membuka gulungan

- Replikasi body of loop beberapa kali
- Iterasi loop lebih sedikit
- Mengurangi overhead loop
- Meningkatkan paralelisme instruksi
- Peningkatan register, data cache, atau lokalitas TLB

Tabel 15.8
Normal Dan Cabang Tertunda

Address	Normal Branch		Delayed Branch		Optimized Delayed Branch	
100	LOAD	X, rA	LOAD	X, rA	LOAD	X, rA
101	ADD	1, rA	ADD	1, rA	JUMP	105
102	JUMP	105	JUMP	106	ADD	1, rA
103	ADD	rA, rB	NOOP		ADD	rA, rB
104	SUB	rC, rB	ADD	rA, rB	SUB	rC, rB
105	STORE	rA, Z	SUB	rC, rB	STORE	rA, Z
106			STORE	rA, Z		

(Tabel dapat ditemukan di halaman 557 di buku teks.)

Time

	1	2	3	4	5	6	7	8
100 LOAD X, rA	I	E	D					
101 ADD 1, rA		I		E				
102 JUMP 105				I	E			
103 ADD rA, rB					I	E		
105 STORE rA, Z						I	E	D

(a) Traditional Pipeline

	1	2	3	4	5	6	7	8
100 LOAD X, rA	I	E	D					
101 ADD 1, rA		I		E				
102 JUMP 106				I	E			
103 NOOP					I	E		
106 STORE rA, Z						I	E	D

(b) RISC Pipeline with Inserted NOOP

	1	2	3	4	5	6
100 LOAD X, Ar	I	E	D			
101 JUMP 105		I	E			
102 ADD 1, rA			I	E		
105 STORE rA, Z				I	E	D

(c) Reversed Instructions

Figure 15.7 Use of the Delayed Branch

```
do i=2, n-1
     a[i] = a[i] + a[i-1] * a[i+1]
end do
```

(a) original loop

```
do i=2, n-2, 2
     a[i] = a[i] + a[i-1] * a[i+i]
     a[i+1] = a[i+1] + a[i] * a[i+2]
end do

if (mod(n-2,2) = i) then
     a[n-1] = a[n-1] + a[n-2] * a[n]
end if
```

(b) loop unrolled twice

Figure 15.8 Loop unrolling

MIPS R4000

Salah satu set chip RISC pertama yang tersedia secara komersial dikembangkan oleh MIPS Technology Inc.

Terinspirasi oleh sistem eksperimental yang dikembangkan di Stanford

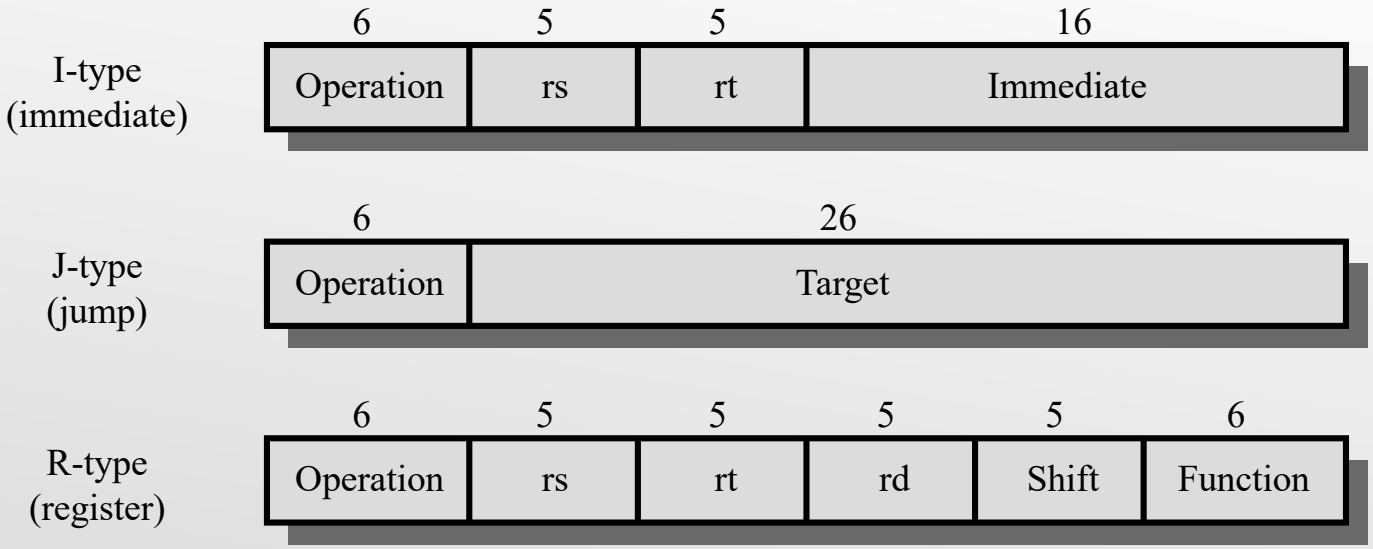
Memiliki arsitektur dan kumpulan instruksi yang sama dari desain MIPS sebelumnya (R2000 dan R3000)

Menggunakan 64 bit untuk semua jalur data internal dan eksternal dan untuk alamat, register, dan ALU

Dipartisi menjadi dua bagian, satu berisi CPU dan yang lain berisi koprosesor untuk manajemen memori

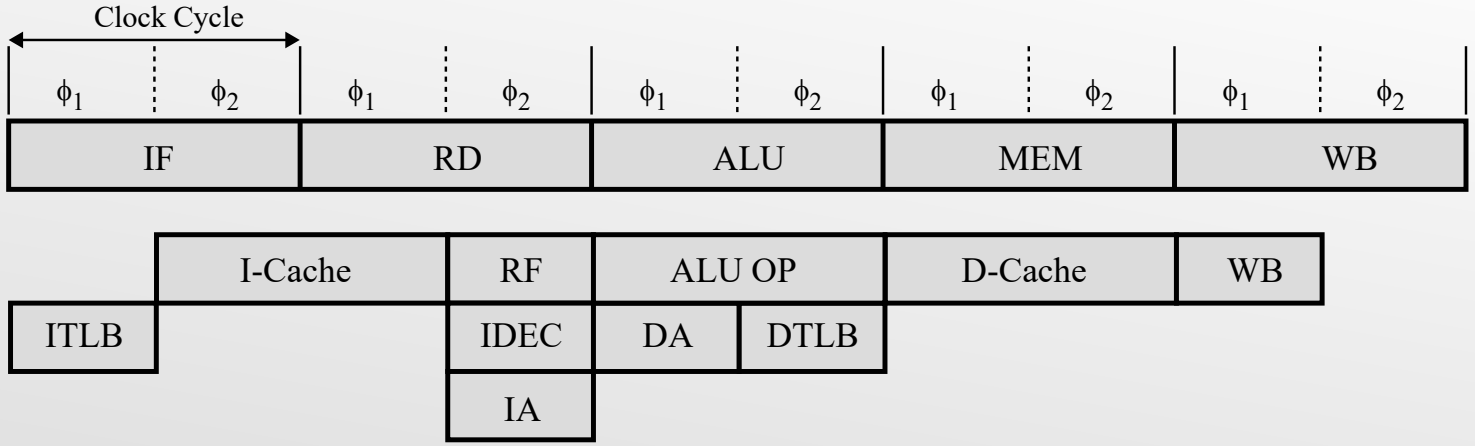
Mendukung tiga puluh dua register 64-bit

Menyediakan hingga 128 Kbytes cache berkecepatan tinggi, masing-masing setengah untuk instruksi dan data

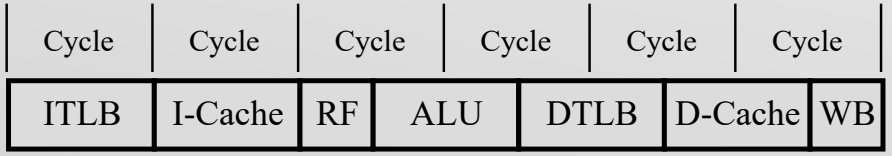


Operation	Operation code
rs	Source register specifier
rt	Source/destination register specifier
Immediate	Immediate, branch, or address displacement
Target	Jump target address
rd	Destination register specifier
Shift	Shift amount
Function	ALU/shift function specifier

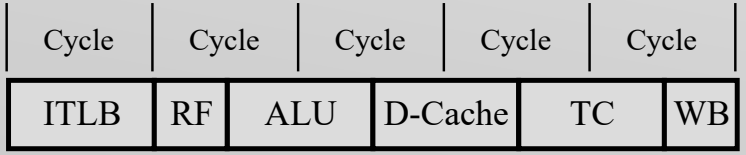
Figure 15.9 MIPS Instruction Formats



(a) Detailed R3000 pipeline



(b) Modified R3000 pipeline with reduced latencies



(c) Optimized R3000 pipeline with parallel TLB and cache accesses

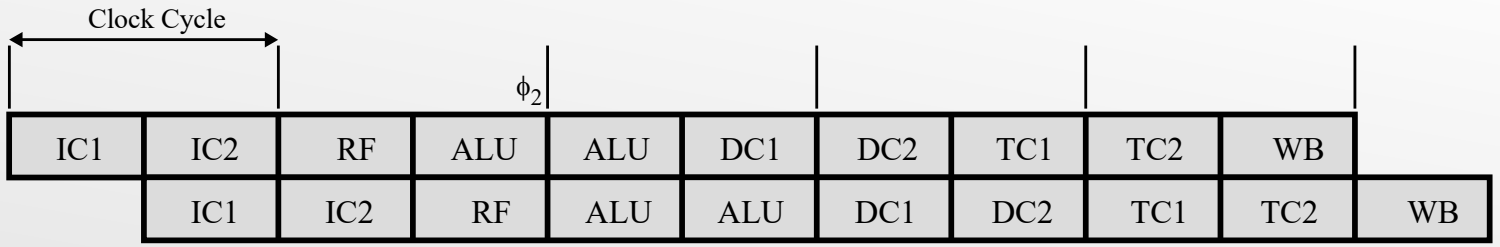
IF	=	Instruction fetch
RD	=	Read
MEM	=	Memory access
WB	=	Write back to register file
I-Cache	=	Instruction cache access
RF	=	Fetch operand from register
D-Cache	=	Data cache access
ITLB	=	Instruction address translation
IDEC	=	Instruction decode
IA	=	Compute instruction address
DA	=	Calculate data virtual address
DTLB	=	Data address translation
TC	=	Data cache tag check

Figure 15.10 Enhancing the R3000 Pipeline

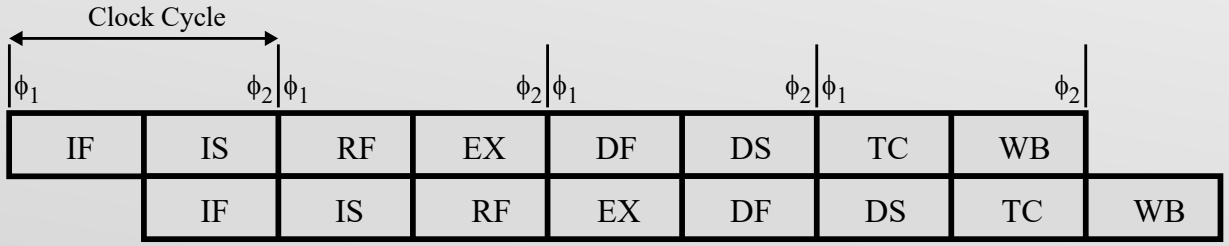
Tabel 15.9
Tahapan Pipa R3000

Pipeline Stage	Phase	Function
IF	$\phi 1$	Using the TLB, translate an instruction virtual address to a physical address (after a branching decision).
IF	$\phi 2$	Send the physical address to the instruction address.
RD	$\phi 1$	Return instruction from instruction cache. Compare tags and validity of fetched instruction.
RD	$\phi 2$	Decode instruction. Read register file. If branch, calculate branch target address.
ALU	$\phi 1 + \phi 2$	If register-to-register operation, the arithmetic or logical operation is performed.
ALU	$\phi 1$	If a branch, decide whether the branch is to be taken or not. If a memory reference (load or store), calculate data virtual address.
ALU	$\phi 2$	If a memory reference, translate data virtual address to physical using TLB.
MEM	$\phi 1$	If a memory reference, send physical address to data cache.
MEM	$\phi 2$	If a memory reference, return data from data cache, and check tags.
WB	$\phi 1$	Write to register file.

(Tabel dapat ditemukan di halaman 563 di buku teks.)



(a) Superpipelined implmentation of the optimized R3000 pipeline



(b) R4000 pipeline

IF = Instruction fetch first half
IS = Instruction fetch second half
RF = Fetch operands from register
EX = Instruction execute
IC = Instruction cache
DC = Data cache
DF = Data cache first half
DS = Data cache second half
TC = Tag check
WB = Write back to register file

Figure 15.11 Theoretical R3000 and Actual R4000 Superpipelines

Tahapan Pipa R4000

Pengambilan instruksi paruh pertama

Alamat virtual disajikan ke cache instruksi dan buffer tepi tampilan terjemahan

Instruksi mengambil paruh kedua

Cache instruksi mengeluarkan instruksi dan TLB menghasilkan alamat fisik

Daftarkan file

Salah satu dari tiga aktivitas dapat terjadi:

- Instruksi diterjemahkan dan pemeriksaan dibuat untuk kondisi interlock
- Pemeriksaan tag cache instruksi dilakukan
- Operand diambil dari file register

Periksa tag

Pemeriksaan tag cache dilakukan untuk pemuatan dan penyimpanan

Eksekusi instruksi

Salah satu dari tiga aktivitas dapat terjadi:

- Jika operasi register-to-register ALU melakukan operasi tersebut
- Jika memuat atau menyimpan data alamat virtual dihitung
- Jika cabang alamat virtual target cabang dihitung dan operasi cabang diperiksa

Cache data terlebih dahulu

Alamat virtual disajikan ke cache data dan TLB

Cache data kedua

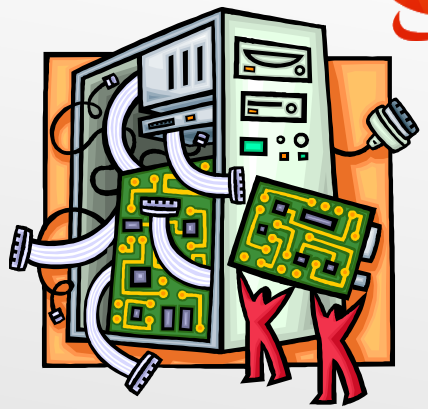
TLB menghasilkan alamat fisik dan cache data mengeluarkan data

Menulis kembali

Hasil instruksi ditulis kembali ke file register

SPARC

Arsitektur Prosesor yang Dapat Diskalakan



- Arsitektur yang ditentukan oleh Sun Microsystems
- Sun melisensikan arsitektur tersebut kepada vendor lain untuk memproduksi mesin yang kompatibel dengan SPARC
- Terinspirasi oleh mesin Berkeley RISC 1, dan set instruksi serta organisasi registernya didasarkan pada model Berkeley RISC

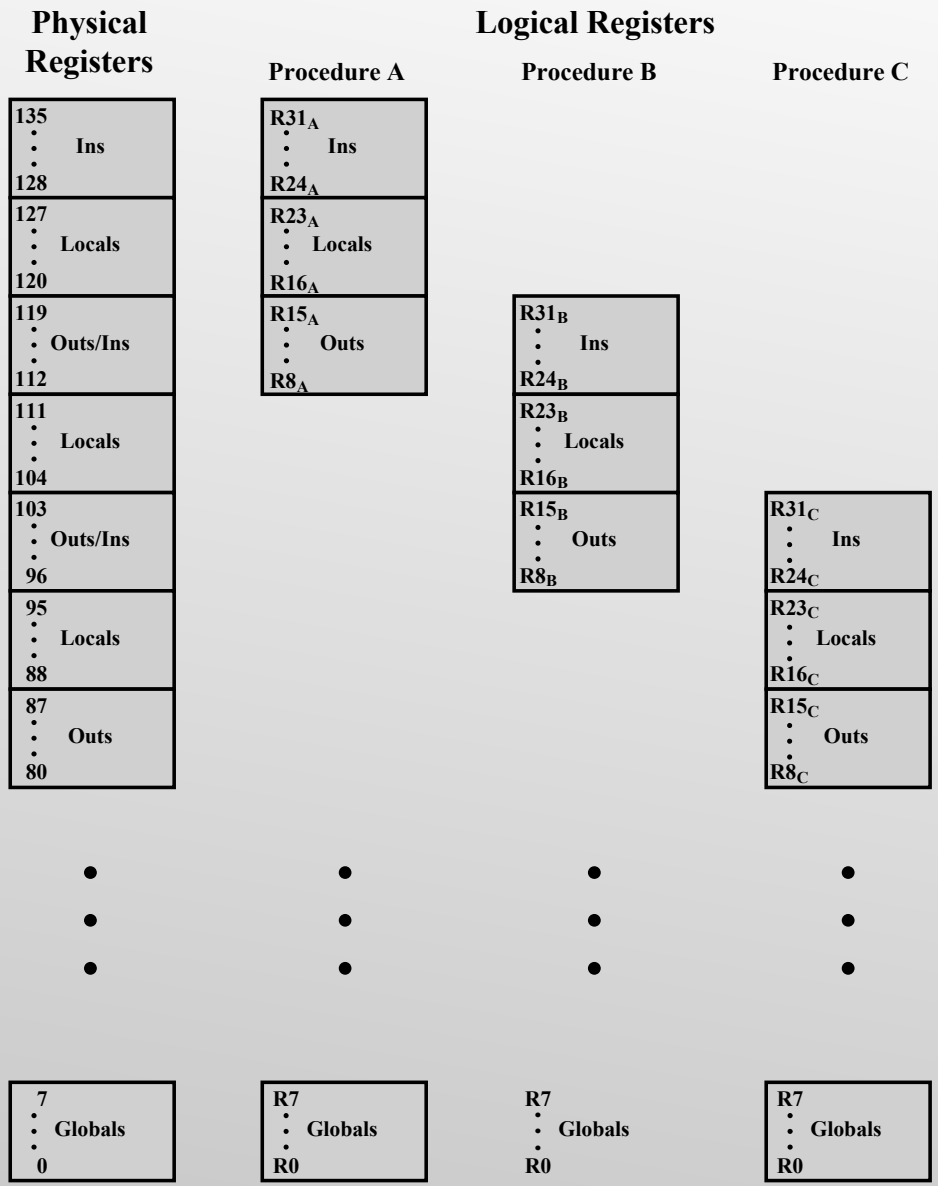


Figure 15.12 SPARC Register Window Layout with Three Procedures

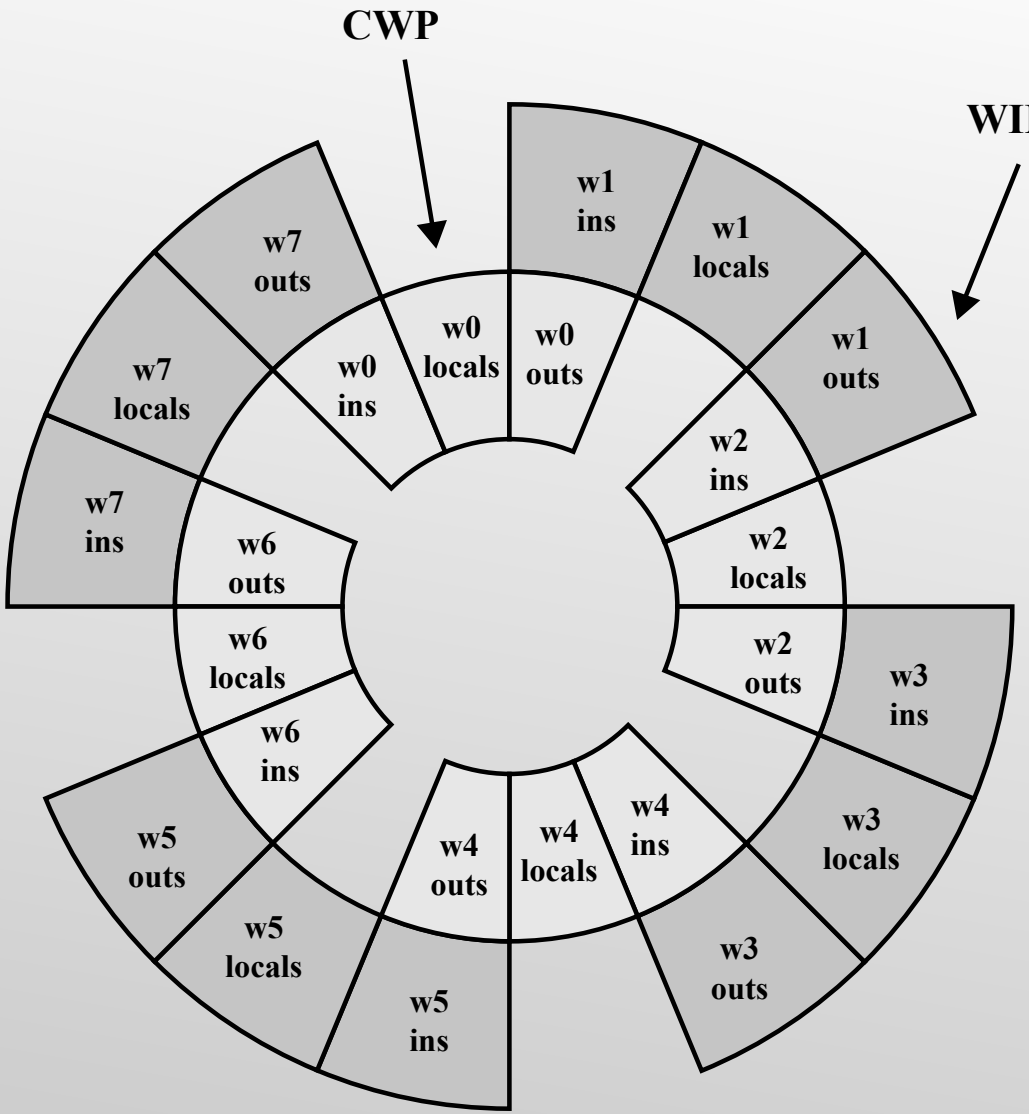


Figure 15.13 Eight Register Windows Forming a Circular Stack in SPARC

Meja 15.10
Mensintesis Mode Pengalamatan Lainnya dengan Mode Pengalamatan SPARC

Instruction Type	Addressing Mode	Algorithm	SPARC Equivalent
Register-to-register	Immediate	operand = A	S2
Load, store	Direct	EA = A	R_0 + S2
Register-to-register	Register	EA = R	R_{S1}, R_{S2}
Load, store	Register Indirect	EA = (R)	R_{S1} + 0
Load, store	Displacement	EA = (R) + A	R_{S1} + S2

S2 = either a register operand or a 13-bit immediate operand

(Tabel dapat ditemukan di halaman 568 di buku teks.)

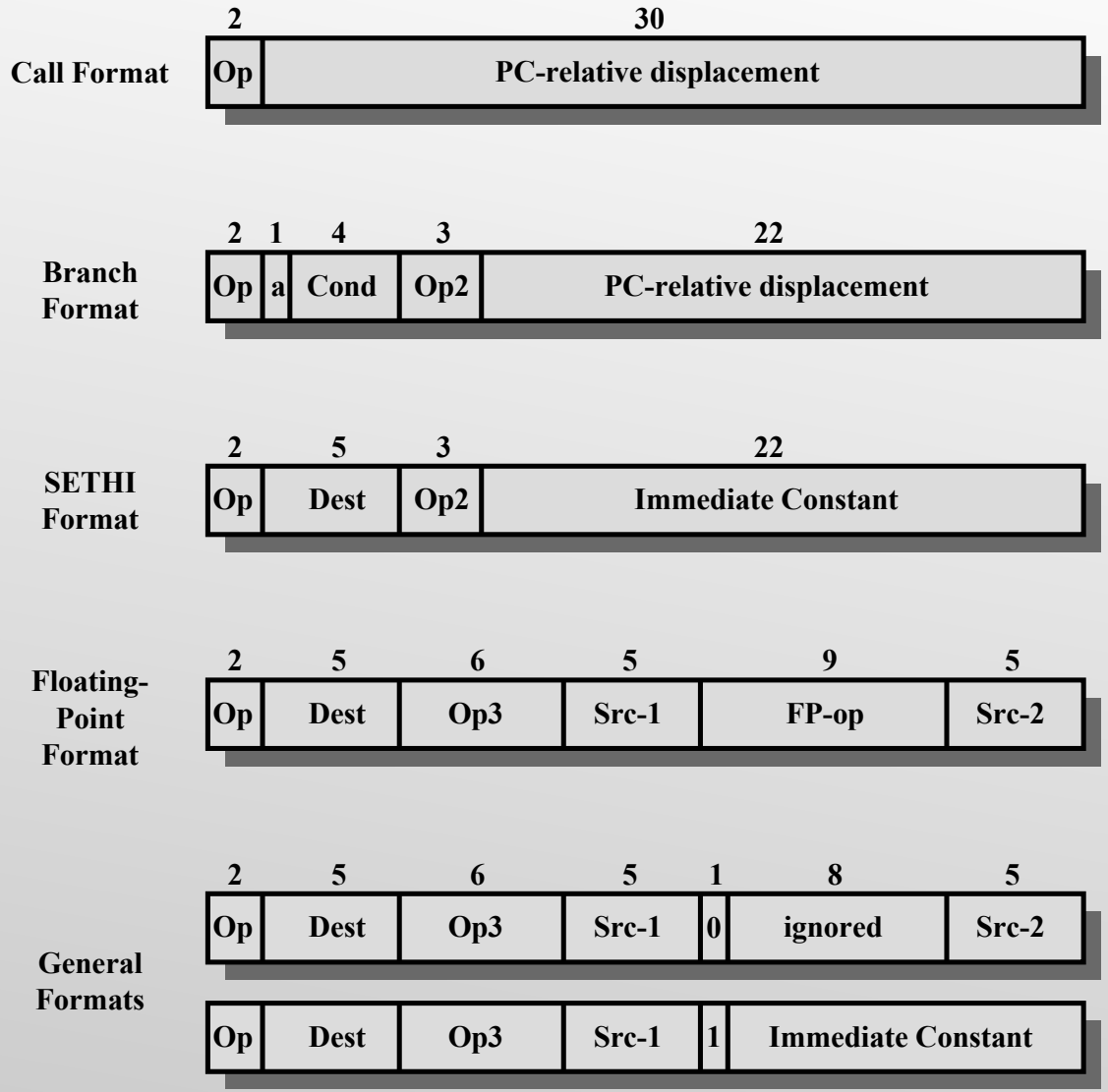


Figure 15.14 SPARC Instruction Formats

Kontroversi RISC versus CISC

Kuantitatif

Cmembandingkan ukuran dan eksekusi program kecepatan program pada mesin RISC dan CISC yang menggunakan teknologi yang sebanding

Kualitatif

Examine masalah dukungan bahasa tingkat tinggi dan penggunaan real estate VLSI

Masalah dengan perbandingan:

Tidak ada pasangan RISC dan CISC mesin yang sebanding dalam biaya siklus hidup, tingkat teknologi, kompleksitas gerbang, kecanggihan kompiler, dukungan sistem operasi, dll.

Tidak ada rangkaian pengujian yang pasti program ada

Sulit untuk memisahkan efek perangkat keras dari efek yang lebih lengkap

Kebanyakan perbandingan dilakukan pada "mainan" daripada produk komersial

Sebagian besar perangkat komersial diiklankan karena RISC memiliki campuran karakteristik RISC dan CISC

Ringkasan

Bab 15

Komputer Set Instruksi Berkurang (RISC)

Karakteristik pelaksanaan instruksi
- Operasi
- Operand
- Panggilan prosedur
- Implikasi

Penggunaan file register yang besar
- Daftarkan jendela
- Variabel global
- File register besar versus cache

Pengurangan arsitektur set instruksi
- Karakteristik RISC
- CISC versus karakteristik RISC

Pipelining RISC
- Pipelining dengan instruksi biasa
- Optimalisasi pipelining

MIPS R4000
- Set instruksi
- Pipa instruksi

SPARC
- Set register SPARC
- Set instruksi
- Format instruksi

Optimasi register berbasis kompiler

Kontroversi RISC versus CISC