Università degli Studi di Milano - Corso Architettura degli elaboratori e delle reti – Turno 1 – Prof. Borghese

Cognome e nome dello studente: ***Matricola:***

A.A. 2007-2008 – Appello del 22 Gennaio 2009

[2] Criteri di progettazione di cache primaria e secondaria.

[4] Multiple issue statico e dinamico. Definizioni, punti deboli e punti di forza.

[6] Facendo riferimento al coprocessore 0, riportato in uno dei fogli allegati al compito, definire il ruolo dei diversi campi dei registri più significativi per la gestione delle interruzioni. Sempre facendo riferimento ai registri del coprocessore 0, scrivere uno pseudo-programma per la gestione di un interrupt o di un'eccezione a vostra scelta.

[12] Data la CPU con pipeline rappresentata in uno dei fogli allegati al compito:

a) Definire da quali fasi è costituito il ciclo di esecuzione di un'istruzione e quando la CPU "capisce" di che istruzione si tratta. In quale fase termina l'esecuzione di un'istruzione? [2].

b) Discutere le differenza tra la CPU singolo ciclo, multi-ciclo e con pipe-line. [2]

c) Dato il seguente segmento di codice:

```
or $s0, $t1, $t2
lw $s1, 16($t2)
sub $t2, $t2, $t3
addi $t3, $s1, 64
and $s1, $s5, $s6
```

Scrivere qual è il contenuto di **tutti** i registri di pipe-line (lo stato) quando l'istruzione and $s1, $s5, $s6 si trova nella fase di fetch [5]. Non tenere conto degli hazard.

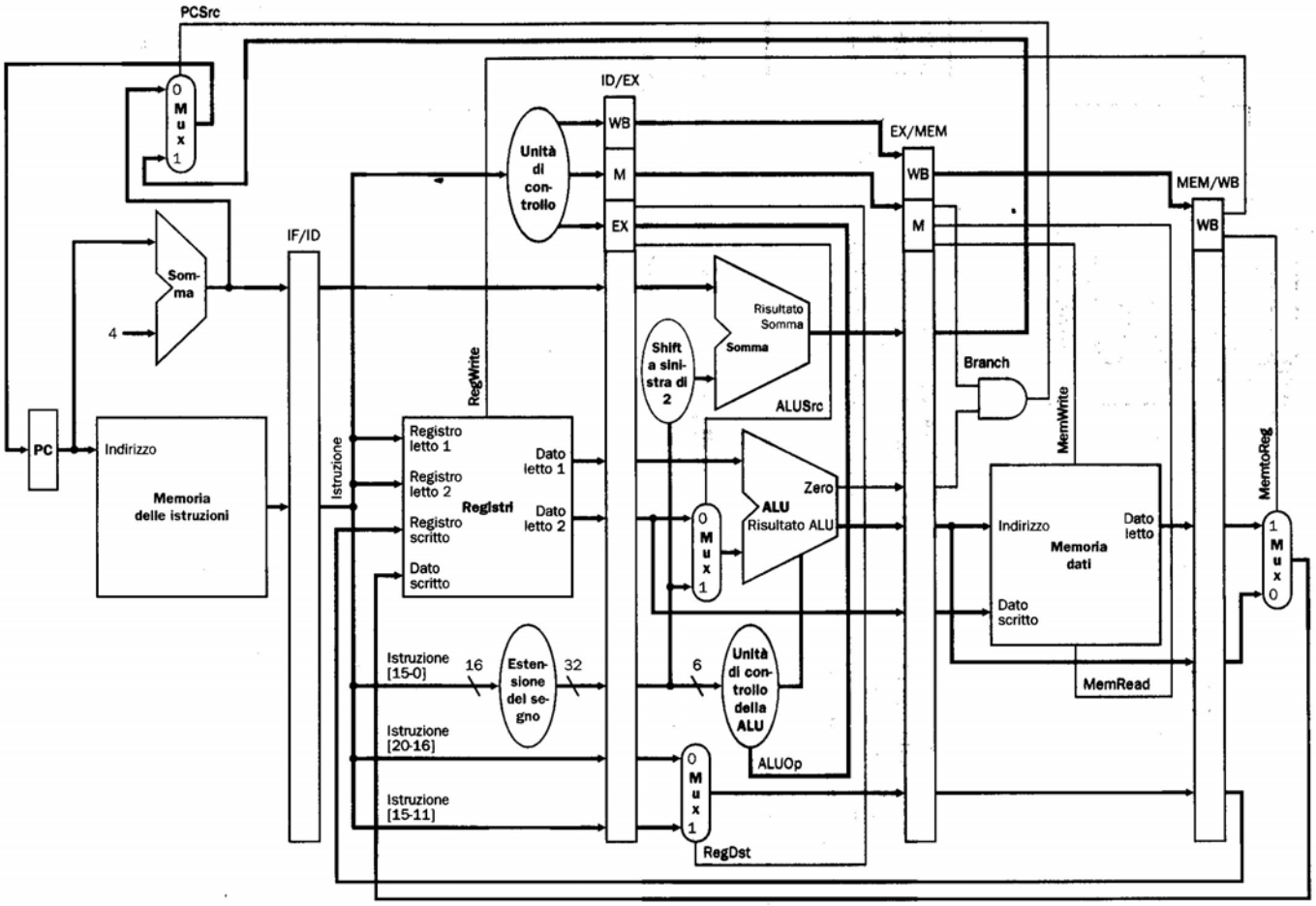
d) Definire cos'è un hazard e cos'è uno stallo. Identificare se nel frammento di codice di cui sopra ci sono hazard. Eventualmente modificare la CPU in modo tale che riesca a gestire al meglio questi hazard [4].

[5] Dimostrare che la porta NAND o la porta NOR sono porte universali. Dare la definizione di prima forma canonica. Scrivere nella prima forma canonica la seguente funzione booleana:

$$Y = ABC + !B\ !C \qquad (1)$$

Costruire il circuito associato all'equazione 1 di cui sopra e definire la lunghezza del cammino critico e la complessità circuitale.

[5] Costruire un sommatore ad anticipatore di riporto per sottrazioni e somme su 4 bit. Calcolare la complessità ed il cammino critico.


PCSrc
0
M
u
x
1
ID/EX
WB
M
EX
Unità di controllo
EX/MEM
WB
M
MEM/WB
WB
Somma
4
IF/ID
Risultato Somma
Somma
Shift a sinistra di 2
ALUSrc
Branch
RegWrite
MemWrite
MemtoReg
PC
Indirizzo
Memoria delle istruzioni
Istruzione
Registro letto 1
Registro letto 2
Registro scritto
Dato scritto
Registri
Dato letto 1
Dato letto 2
Zero
ALU
Risultato ALU
Indirizzo
Memoria dati
Dato letto
Dato scritto
MemRead
Istruzione [15-0]
16
Estensione del segno
32
6
Unità di controllo della ALU
ALUOp
Istruzione [20-16]
Istruzione [15-11]
RegDst

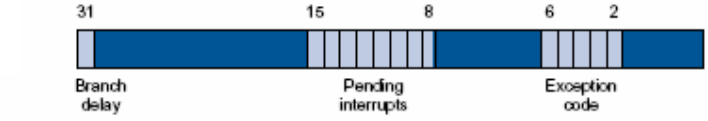
Register File

0	zero	constant 0
1	at	reserved for assembler
2	v0	expression evaluation &
3	v1	function results
4	a0	arguments
5	a1	
6	a2	
7	a3	
8	t0	temporary: caller saves
. . .		(callee can clobber)
15	t7	
16	s0	callee saves
. . .		(caller can clobber)
23	s7	
24	t8	temporary (cont'd)
25	t9	
26	k0	reserved for OS kernel
27	k1	
28	gp	Pointer to global area
29	sp	Stack pointer
30	fp	frame pointer (s8)
31	ra	Return Address (HW)

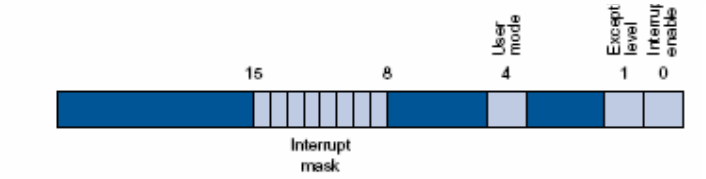
Coprocessore 0

Nome del registro	Numero del registro in coprocessore 0	Utilizzo
Bad/Addr	8	Registro contenente l'indirizzo di memoria a cui si è fatto riferimento
Count	9	Timer
Compare	11	Valore da comparare con un timer. Genera un interrupt.
Status	12	Maschera delle interruzioni e bit di abilitazione. Stato dei diversi livelli di priorità (6 HW e 2 SW).
Cause	13	Tipo dell'interruzione e bit delle interruzioni pendenti
EPC	14	Registro contenente l'indirizzo dell'istruzione che ha causato l'interruzione.

Registro causa:



Registro stato:



Codici operativi

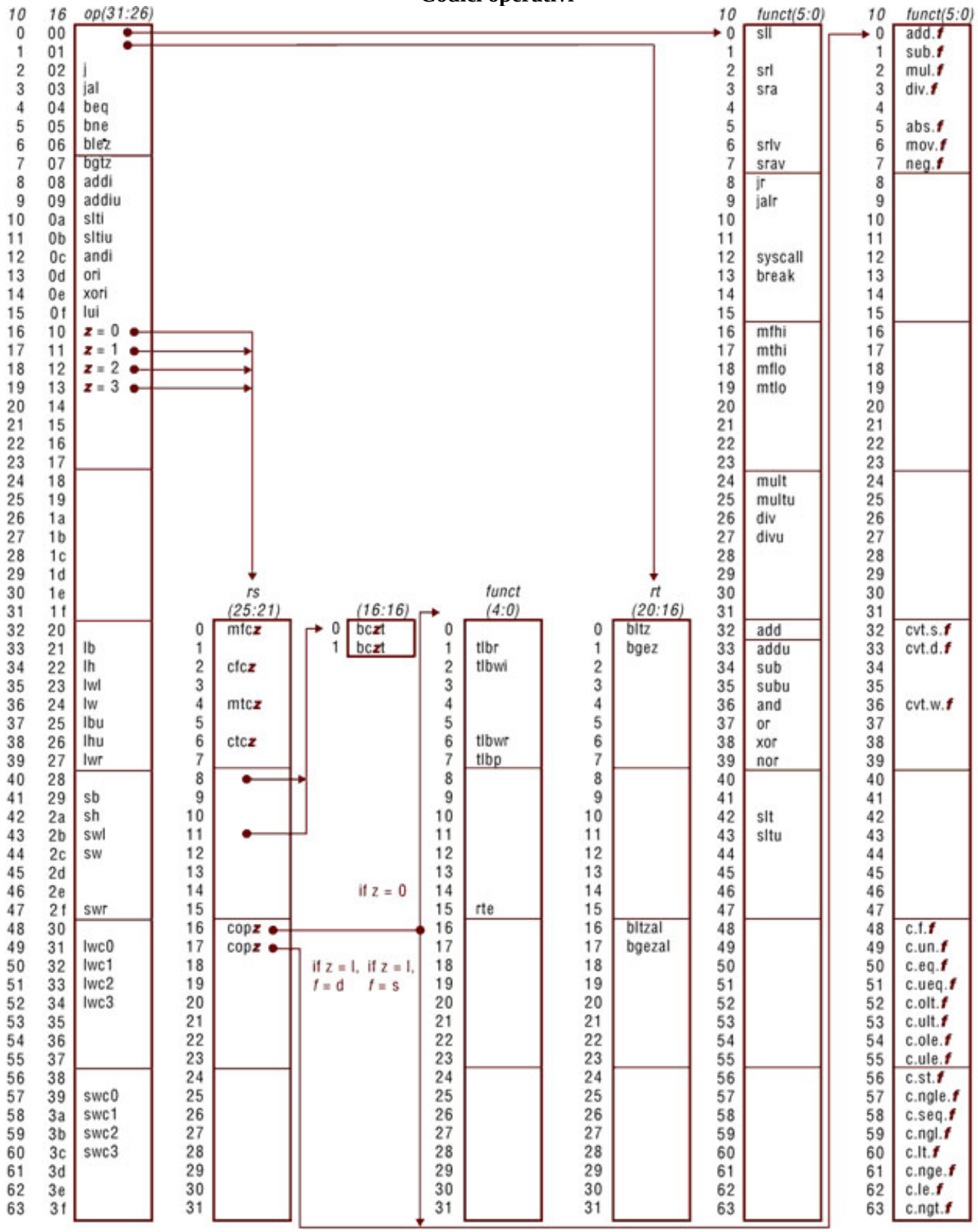


FIGURE A.19 MIPS opcode map. The values of each field are shown to its left. The first column shows the values in base 10 and the second shows base 16 for the op field (bits 31 to 26) in the third column. This op field completely specifies the MIPS operation except for 6 op values: 0, 1, 16, 17, 18, and 19. These operations are determined by other fields, identified by pointers. The last field (funct) uses "*f*" to mean "s" if rs = 16 and op = 17 or "d" if rs = 17 and op = 17. The second field (rs) uses "*z*" to mean "0", "1", "2", or "3" if op = 16, 17, 18, or 19, respectively. If rs = 16, the operation is specified elsewhere: if *z* = 0, the operations are specified in the fourth field (bits 4 to 0); if *z* = 1, then the operations are in the last field with *f* = s. If rs = 17 and *z* = 1, then the operations are in the last field with *f* = d. *(page A-54)*