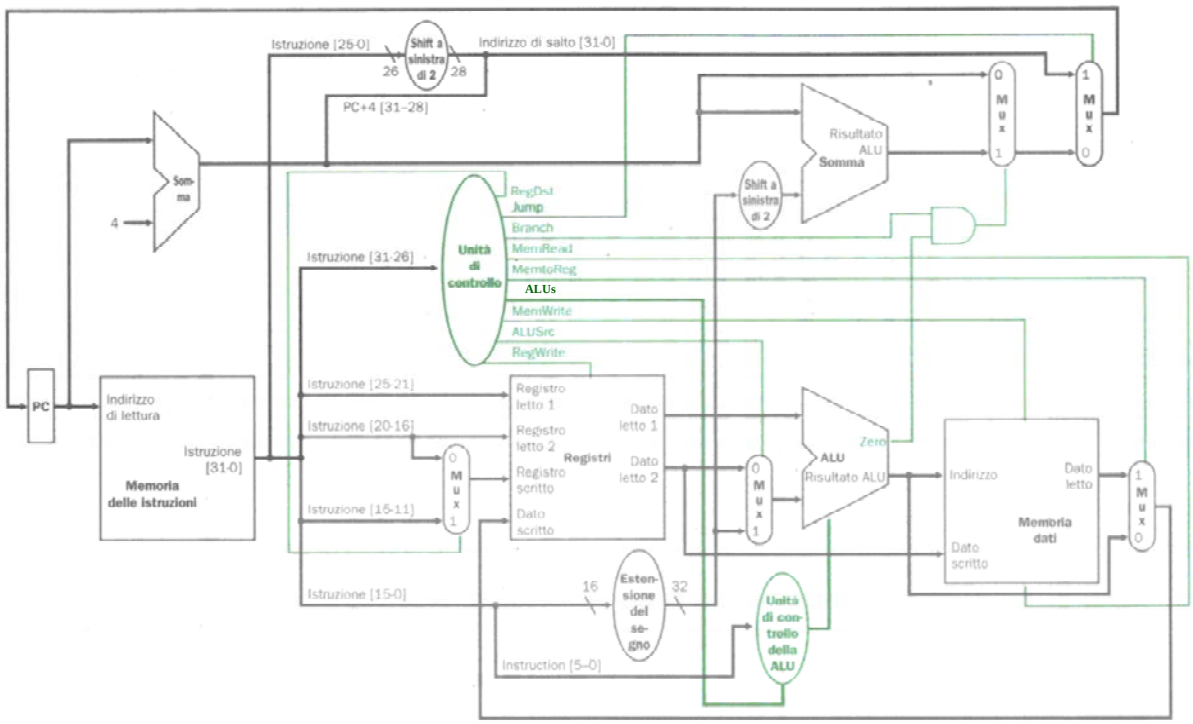
Università degli Studi di Milano - Corso Architettura degli elaboratori e delle reti – Turno 1 – Prof. Borghese

A.A. 2006-2007 – Terza prova in itinere – 22 Maggio 2007

Nome e Cognome dello studente:

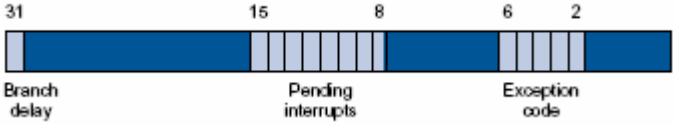


1) Data la CPU disegnata sopra, scrivere la funzione logica che controlla i salti (condizionati ed incondizionati); cosa riceve in input, cosa riceve in output, quale funzione implementa? E' una funzione combinatoria o sequenziale? Perché? [3]

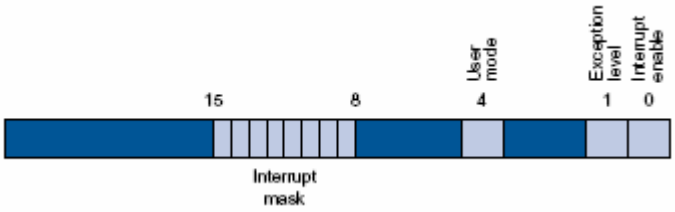
2) Cosa si intende per superpipeline? Cosa si intende per pipeline superscalari? Cosa si intende per reorder buffer? A cosa servono le reservation stations? Cosa si intende per flush di una pipeline? Cosa si intende per hazard strutturale? Cosa è uno stallo di una pipeline? E' corretto affermare che una pipeline aumenta la velocità di esecuzione di un'istruzione e perché? Cosa si intende per esecuzione speculativa? Cos'è un branch prediction buffer? [7]

3) Come vengono gestite le interruzione vettorializzate? Come vengono gestite mediante il registro causa? Vengono riportati qui sotto i registri causa e stato del MIPS. Scrivere una procedura Assembly che gestisca correttamente un interrupt di livello 2 inviando un messaggio a schermo contenente l'indirizzo dell'istruzione che ha generato l'interrupt. Scrivere una procedura Assembly che gestisca correttamente un'eccezione di breakpoint inviando il codice di stato, e l'indirizzo dell'eccezione ad una procedura che li stampa a monitor [7].

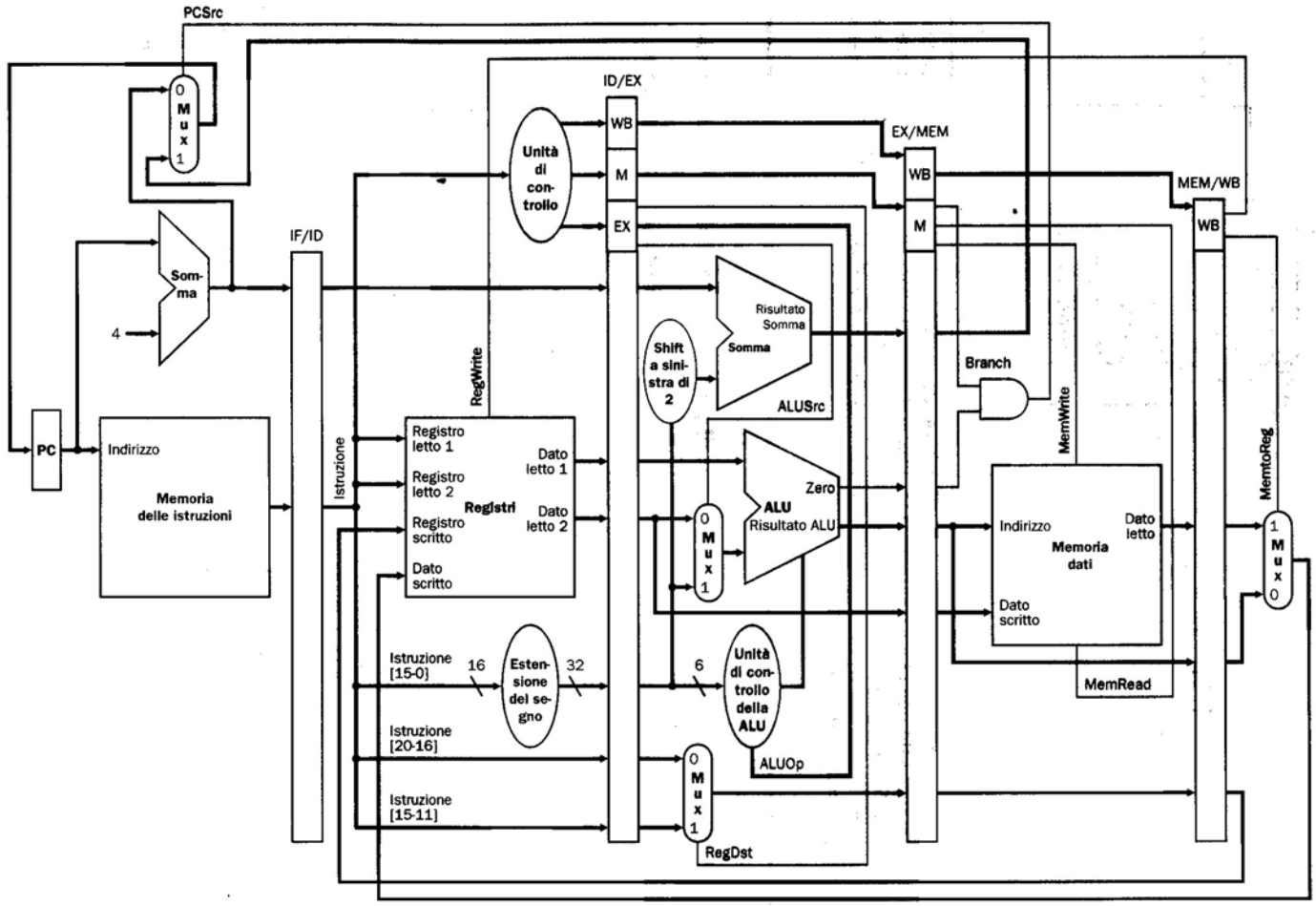
Registro causa



Registro stato

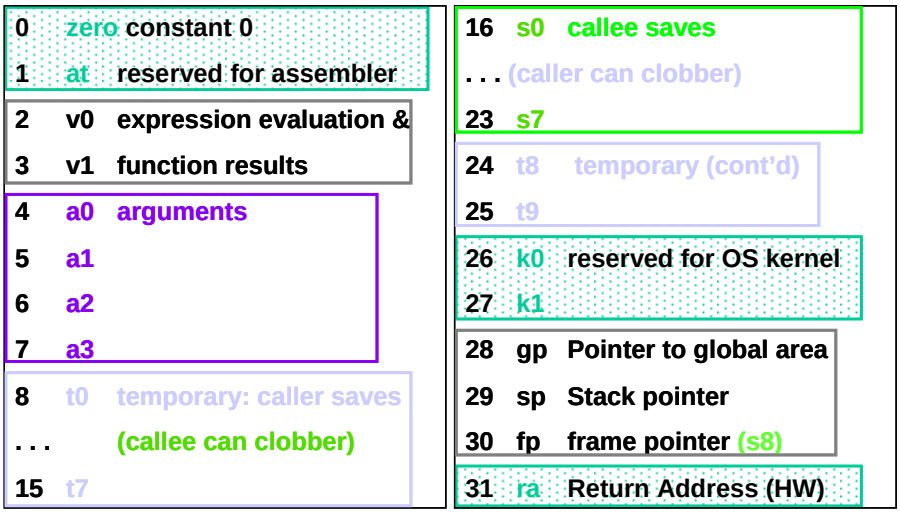


4) Data la CPU disegnata sotto [19]:



1) Scrivere il contenuto di tutti i registri (parte master di PC + parte master dei registri di pipeline) durante l'esecuzione di questo frammento di codice [6]:
```
400: lw $t1, 20($t2)
404: beq $t0, $t2, 20
408: add $s0, $s1, $s2
412: or $s3, $s4, $s5
416: and $s6, $s0, $s0
```
Quando l'istruzione `and` è in fase di fetch.
2) Ci sono hazard in questo frammento di codice (motivare la risposta)? [2]
3) Modificare la CPU disegnata sopra in modo da potere gestire gli hazard presenti in questo frammento di codice [6]:
```
4000: add $s0, $s1, $s2
4004: addi $t0, $s0, 42
4008: or $t1, $t0, $s0
```
4) Scrivere le funzioni logiche implementate dall'unità di controllo che gestisce la criticità riportata nel seguente segmento di codice [3]:
```
800: lw $s1, 0($s2)
804: add $t0, $t2, $s1
808: add $t1, $t2, $s1
```
5) Dimensionare correttamente tutti i registri presenti nella CPU [2].

Number	Name	Cause of exception
0	Int	interrupt (hardware)
4	AdEL	address error exception (load or instruction fetch)
5	AdES	address error exception (store)
6	IBE	bus error on instruction fetch
7	DBE	bus error on data load or store
8	Sys	syscall exception
9	Bp	breakpoint exception
10	RI	reserved instruction exception
11	CpU	coprocessor unimplemented
12	Ov	arithmetic overflow exception
13	Tr	trap
15	FPE	floating point



Codici operativi: lw – 35, sw – 43, addi – 8, beq - 4.. Campo funct: add – 32, or – 35, and -36.

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.