Università degli Studi di Milano - Corso Architettura degli elaboratori II – Prof. Borghese
Prima prova in itinere 27.03.2023 – Versione B

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

Il compito è diviso in due sezioni: la prima sezione a risposta multiple e la seconda sezione a risposte aperte. Per raggiungere la sufficienza in questa prova **occorre raggiungere almeno 6,5 punti** con le risposte ai quesiti della prima sezione.

SEZIONE 1 (Punteggio massimo: 10 punti. Per ogni risposta corretta +1 punto, per ogni riposta sbagliata, -0,25 punti, per ogni risposta mancante 0 punti).

Barrare la risposta esatta tra le 5 risposte possibili: A, B, C, D, E

Data la CPU di Figura 1 e il seguente frammento di codice:

```
0x00000400 or $t5, $s2, $s1
0x00000404 sw $t1, 8($t0)
0x00000408 addi $s4, $t4, 8
0x0000040C add $s1, $t1, $t2
0x00000410 lw $t2, 32($s4)
```

Supporre che l'istruione di `or` sia in fase di WriteBack e rispondere alle seguenti domande:

1. Cosa passa nel cammino a?
   A. 13
   B. 17
   C. $17 + 18*2^{11}$
   D. Non si può determinare
   E. Il codice operativo

2. Cosa passa nel cammino b?
   A. 8
   B. 32
   C. 0x42C
   D. 0x50C
   E. 0x41C

3. Cosa passa nel cammino c?
   A. Il contenuto del registro $t4
   B. Il contenuto del registro $s4
   C. 8
   D. 32
   E. Non si può determinare

Data la CPU di Figura 2 e il seguente frammento di codice:

```
0x00000400 and $t5, $s1, $s1
0x00000404 sw $t1, 8($t0)
0x00000408 add $s4, $t5, $s2
0x0000040C addi $s1, $t1, 20
0x00000410 lw $t2, 8($s4)
```

Supporre che l'istruione di `and` sia in fase di WriteBack e rispondere alle seguenti domande:

4. Cosa passa nel cammino a?
   A. Il contenuto del registro $s1
   B. Il contenuto del registro $t5
   C. Il contenuto del registro $s2
   D. Il contenuto del registro $s4
   E. Non si può sapere

5. Cosa passa nel cammino b?
   A. 32
   B. 8
   C. 0x44C

D. Il contenuto del registro $s2
E. Il contenuto del registro $t5

6. Cosa passa nel cammino c?
   A. Il contenuto del registro $s1
   B. Il contenuto del registro $t1
   C. 16
   D. Il contenuto del registro $t5
   E. 20

7. Cosa è un interrupt?
   A. Un'interruzione del ciclo di esecuzione della CPU
   B. Una richiesta di attenzione della CPU
   C. Un segnale aggiuntivo per l'UC
   D. Un errore che si verifica nella CPU
   E. Un evento non previsto che deve essere gestito dalla CPU e dal Sistema Operativo

8. Le CPU multiple-issue
   A. Sono sinonimo di CPU SIMD
   B. Sono CPU che eseguono in parallelo più istruzioni non in pipeline
   C. Sono CPU che eseguono in parallelo più istruzioni in pipeline
   D. Sono CPU che riorganizzano il codice per evitare stalli
   E. Sono CPU che sono in grado di gestire più di un interrupt alla volta

9. Il flush di un'isrtuzione in una pipeline
   A. E' il blocco di una pipeline per un ciclo di clock
   B. Si verifica quando vengono messi a 0 i segnali di controllo di un'istruzione in una determinata fase e l'istruzione verrà poi fatta ripartire da quella fase con i segnali di controllo attivi.
   C. Si verifica quando un'istruzione viene eliminata dalla pipeline.
   D. Si verifica quandoviene fatto il reset del registro IF/ID.
   E. Si verifica quando viene fatto il reset del PC.

10. Una pipeline aumenta la velocità di esecuzione
   A. Di un fattore pari al numero di cammini paralleli
   B. Di un fattore pari al numero di stadi.
   C. A seconda del tipo di ISA che si ha a disposizione.
   D. Sono se si può operare l'over-clocking
   E. Se il mix di istruzioni lo consente.

SEZIONE 2 (punteggio massimo 25 punti)

1. [8] Modificare la pipeline in Figura 1 perché diventi una pipeline superscalare. Spiegare la ragione e lo scopo di **tutte le modifiche** più rilevanti da apportare ai diversi stadi. Che differenza c'è tra pipeline super-scalare e pipeline dotata di VLIW? Quali sono i vantaggi e gli svantaggi di un approccio rispetto all'altro. Qual è il migliore e perché? Descrivere come funzionano le seguenti tecniche e dire se sono tecniche principalmente **software** o **hardware** e perchè. In alcuni casi la risposta corretta può essere entrambi gli approcci. Identificare quali sono i **punti forti** ed i **punti deboli**.
   a) Issue
   b) Parallelizzazione dell'esecuzione
   c) Pipeline superscalari
   d) Esecuzione fuori ordine
   e) Predizione dei salti
   f) Branch predicion buffer
   g) Speculazione
   h) Reservation station
   i) Buffer di riordino
   j) Ridenominazione dei registri

2. [2] Data la CPU N. 2, quando è in esecuzione il seguente segmento di codice:

```
0x00000400 or $s5, $t2, $t1
0x00000404 lw $t1, 8($s0)
0x00000408 or $s4, $t5, $t1
0x0000040C addi $t1, $t1, 20
0x00000410 sw $t2, 16($s1)
0x00000414 sub $t2, $t0, $t2
```

Supporre che l'istruzione di `or` sia in fase di WriteBack e sottolineare quali linee trasportano segnali utili. Identificare tutti gli hazard.

3. [4] Cosa sono gli interrupt e le eccezioni? Come vengono gestiti dalle architetture Intel e dalle architeture MIPS/ARM? Specificare gli elementi della CPU MIPS che sono dedicati alla gestione delle eccezioni e cosa contengono. Spiegare come la CPU in Figura 2 gestisce un'eccezione di "Overflow" (aggiungere eventuali cammini mancanti). Cosa si intende per mascheramento degli interrupt? Viene praticato nei MIPS? Come vengono gestite le eccezioni e gli interrupt dai sistemi operativi sul MIPS? Scrivere uno scheletro di funzione assembler per gestire un'eccezione.

4. [4] Riscrivere il codice seguente in modo che sia eseguito nel minor tempo possibile su una pipeline con 4 cammini di esecuzione di cui 3 general purpose (in grado di eseguire tutte le istruzioni) e 1 dedicato solo alle istruzioni di memoria. Qual è lo speed-up? (rapporto tra le prestazioni prima della modifica, sul codice sequenziale, e dopo la modifica). Applicare lo srotolamento dei cicli per un numero massimo di 8 iterazioni del ciclo. Si supponga di avere a disposizione un numero di registri interni di pipeline sufficientemente grande e che si possa applicare la ridenominazione dei registri.

```
Ciclo:    lw   $t0, 0($s1)          # M[s1] -> t0
          addu $t0, $t0, $s2        # t0 = t0 + s2
          sw   $t0, 0($s1)          # M[s1] <- t0
          addi $s1, $s1, -4         # next element
          bne  $s1, $zero, Ciclo
          or   $s6, $s7, $s5
```

Il codice assembler corrisponde al seguente codice C:

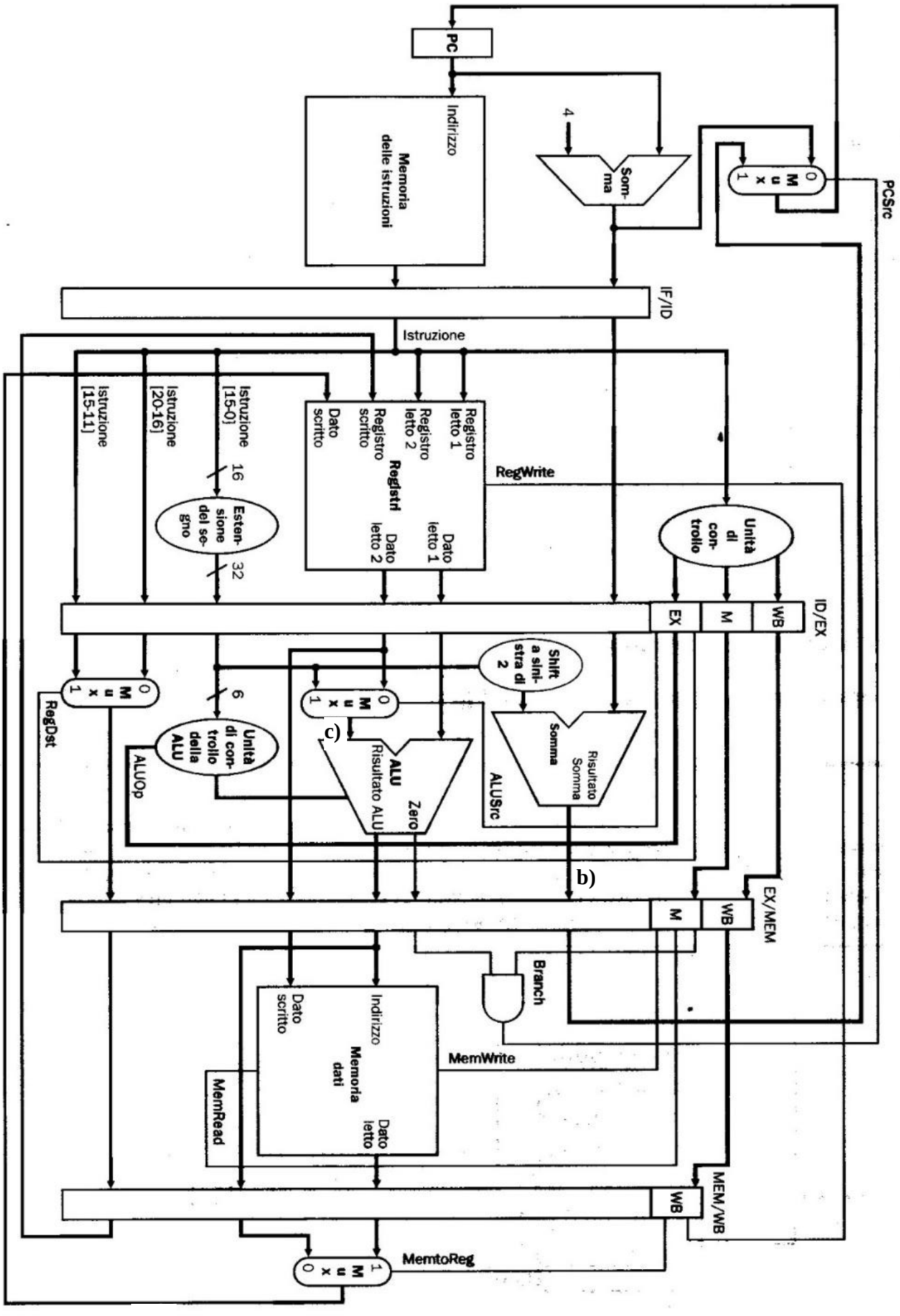
```
Ciclo:    v[s1] = t0 + v[s1];
          s1--;
          if (s1 != 0) goto Ciclo
```

5. [2] Come si implementa l'esecuzione vettoriale in una pipeline super-scalare? Mostrarlo modificando un cammino di esecuzione di una pipeline non dotata di esecuzione vettoriale.

6. [3] Modificare la pipeline di Figura 2, in modo che gestisca correttamente l'hazard generato da un'istruzione aritmetico-logica sequita da un'istruzione di salto condizionato. Fare un esempio.

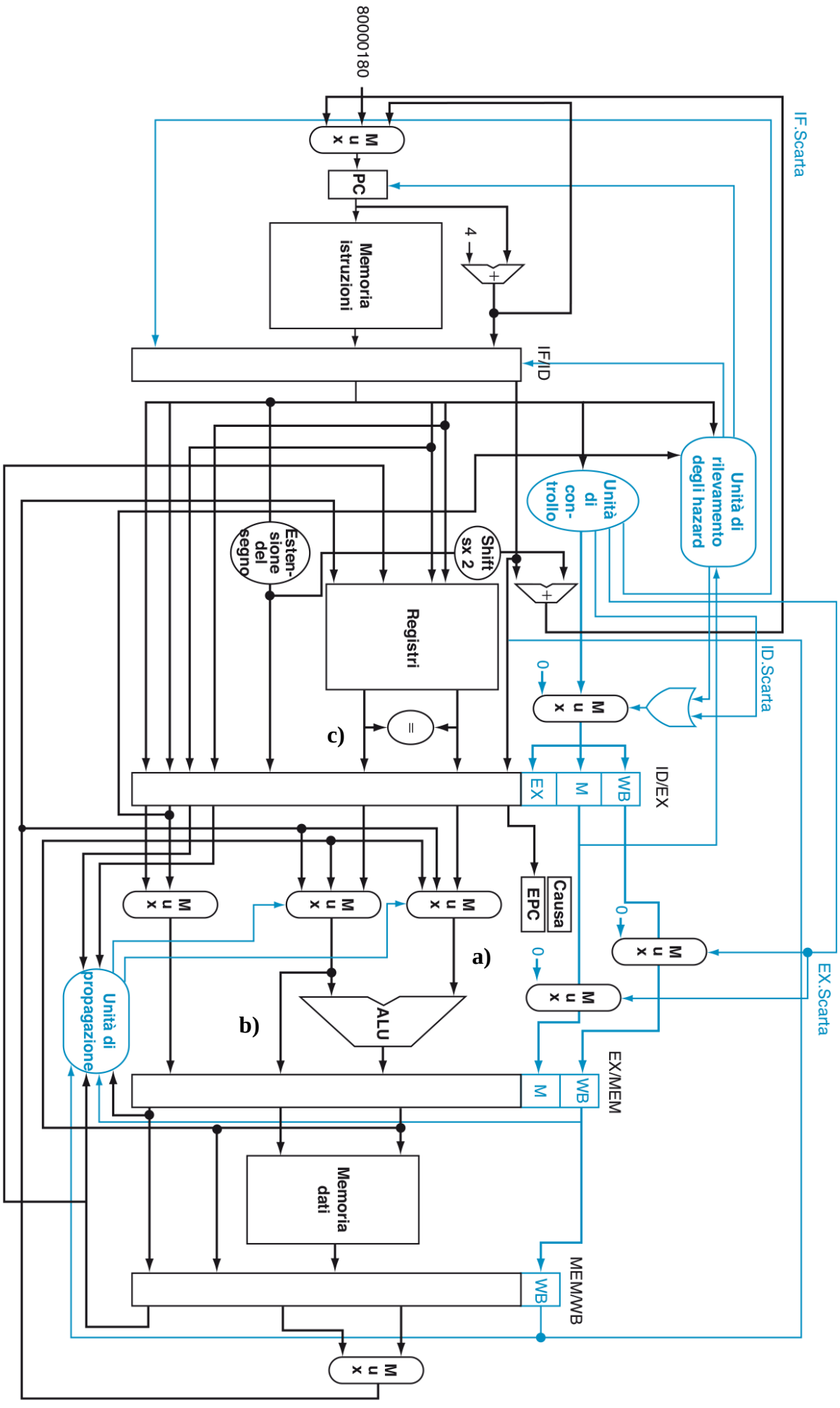
7. [2] Spiegare chiaramente cosa si intende per stallo e cosa si intende per flush di una pipeline.

Figura 1


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

a)

Figura 2



I registri dell'architettura MIPS

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)