Esame di Architetture – Canale AL – Prof. Sterbini – 1/7/16 – Compito A

Cognome e Nome: _____ Matricola: _____

Parte 1 (per chi non ha superato l'esonero)

Esercizio 1A. Si ha il dubbio che in una partita di CPU a ciclo di clock singolo (vedi sul retro) la Control Unit sia rotta, producendo il segnale di controllo **RegWrite** attivo **solo quando** NON è attivo il segnale di controllo **AluSrc**.
Si assume che RegDst sia asserito solo per le istruzioni di tipo R, che MemToReg sia asserito solo per l'istruzione lw, che AluSrc sia asserito solo per le istruzioni lw e sw e di tipo immediato.

a) Si indichino *qui sotto* quali delle istruzioni base (**lw, sw, di tipo R, beq, j, R immediate**) funzioneranno male e qual'è il comportamento anomalo in caso di CU guasta.

b) si scriva *qui sotto* un breve programma assembly MIPS che termina valorizzando il registro $s0 con il valore 1 se il processore è guasto, altrimenti con 0.

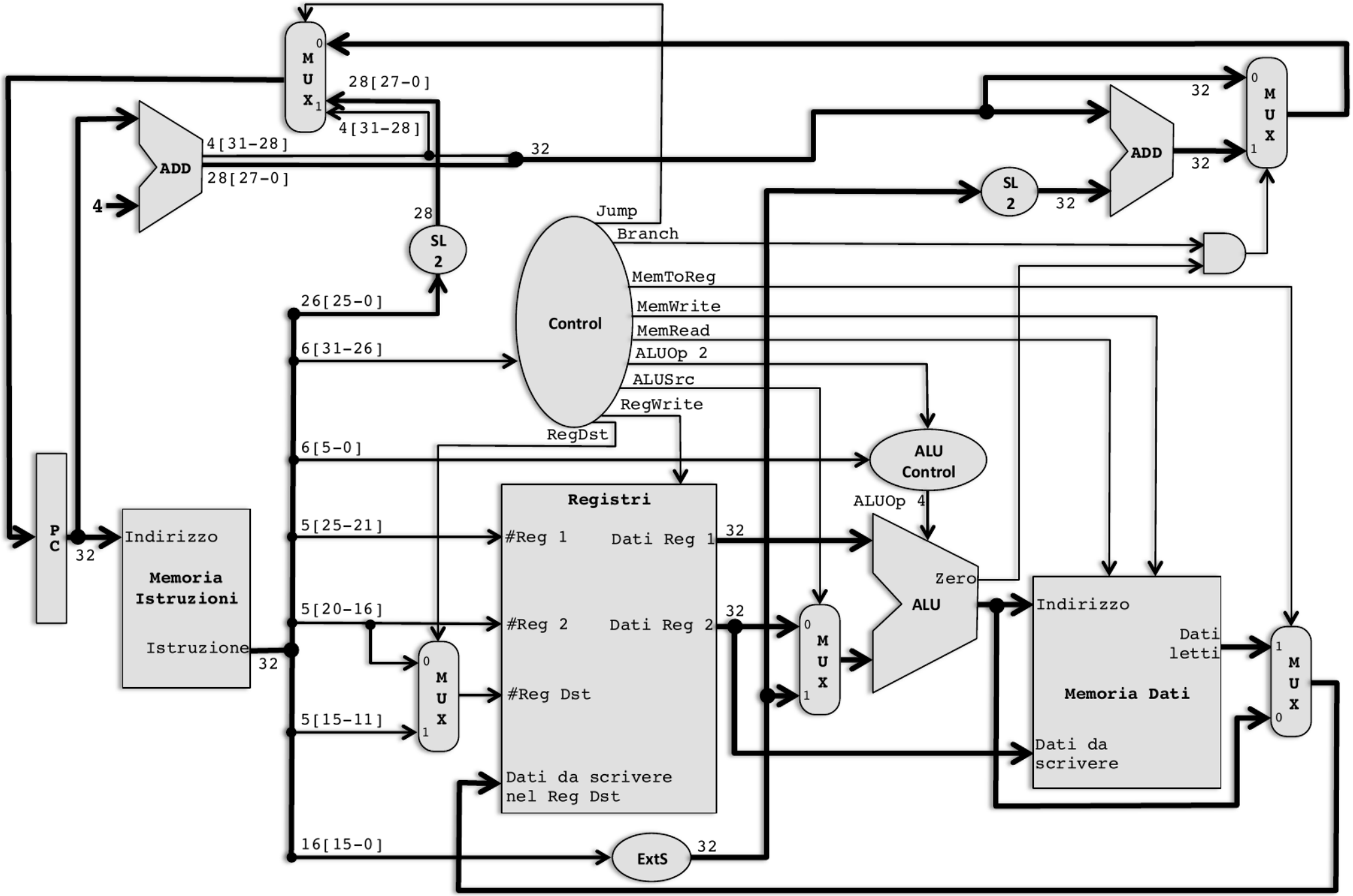
Esercizio 2A. Considerate l'architettura MIPS a ciclo singolo in figura (diagramma sul retro).
Si vuole aggiungere l'istruzione di tipo R **`min rd, rt, rs`** che salva nel registro **rd** il minimo tra i due valori contenuti nei due registri **rs** e **rt**. Nella soluzione non modificate gli ingressi/uscite o le funzionalità della ALU.

1) modificate il diagramma mostrando gli eventuali altri componenti necessari a realizzare l'istruzione

2) indicate sul diagramma tutti i segnali di controllo che la CU genera per realizzare l'istruzione

3) supponendo che l'accesso alle memorie impieghi **133ns**, l'accesso ai registri **66ns**, le operazioni dell'ALU e dei sommatori **200ns**, e ignorando gli altri ritardi di propagazione dei segnali, indicate sul diagramma la durata totale del ciclo di clock per permettere l'esecuzione anche della nuova istruzione.

Implementazione ad un ciclo di clock di MIPS (solamente le istruzioni: `add`, `sub`, `and`, `or`, `xor`, `slt`, `lw`, `sw`, `beq`, `j`)



Esame di Architetture – Canale AL – Prof. Sterbini – 1/7/16 – Compito A

Cognome e Nome: _____ Matricola: _____

Parte 2 (per tutti)

Esercizio 3A. Si consideri l'architettura MIPS con pipeline mostrata in figura (sul retro) ed il frammento di programma qui a destra che somma i valori della prima colonna, scandendo la matrice per puntatori dalla fine all'inizio e tenendo un conteggio della colonna in cui ci si trova.

```
.data
matrice:  .space 400
DIM:      .word 10  #lato matrice
.text
main:   xor  $s4, $s4, $s4     # somma=0
        lw   $s2, DIM          # col=N
        lw   $s1, DIM          # N
        mul  $s1, $s1, $s1     # N^2
        sll  $s1, $s1, 2       # 4N^2
        subi $s1, $s1, 4       # last el.
loop:   lw   $s3, matrice($s1) # x=M[i]
        subi $s2, $s2, 1       # col-1
        bgtz $s2, next         # col=0?
        add  $s4, $s4, $s3     # somma+=x
        lw   $s2, DIM          # col=N
next:   subi $s1, $s1, 4       # prox el
        bgez $s1, loop         # ancora?
end:    li   $v0, 10
        syscall
```

Si indichino qui sotto o sul codice (PASSAGGI INCLUSI):

1) tra quali istruzioni sono presenti data hazard,

2) tra quali istruzioni sono presenti control hazard,

3) quanti cicli di clock sono necessari a eseguire il programma con il forwarding

4) quanti ne sarebbero necessari se il forwarding non esistesse

5) quali sono le istruzioni contenute nei registri della pipeline durante il **13°** ciclo di clock (con FW)

WB: MEM: EXE: ID: IF:

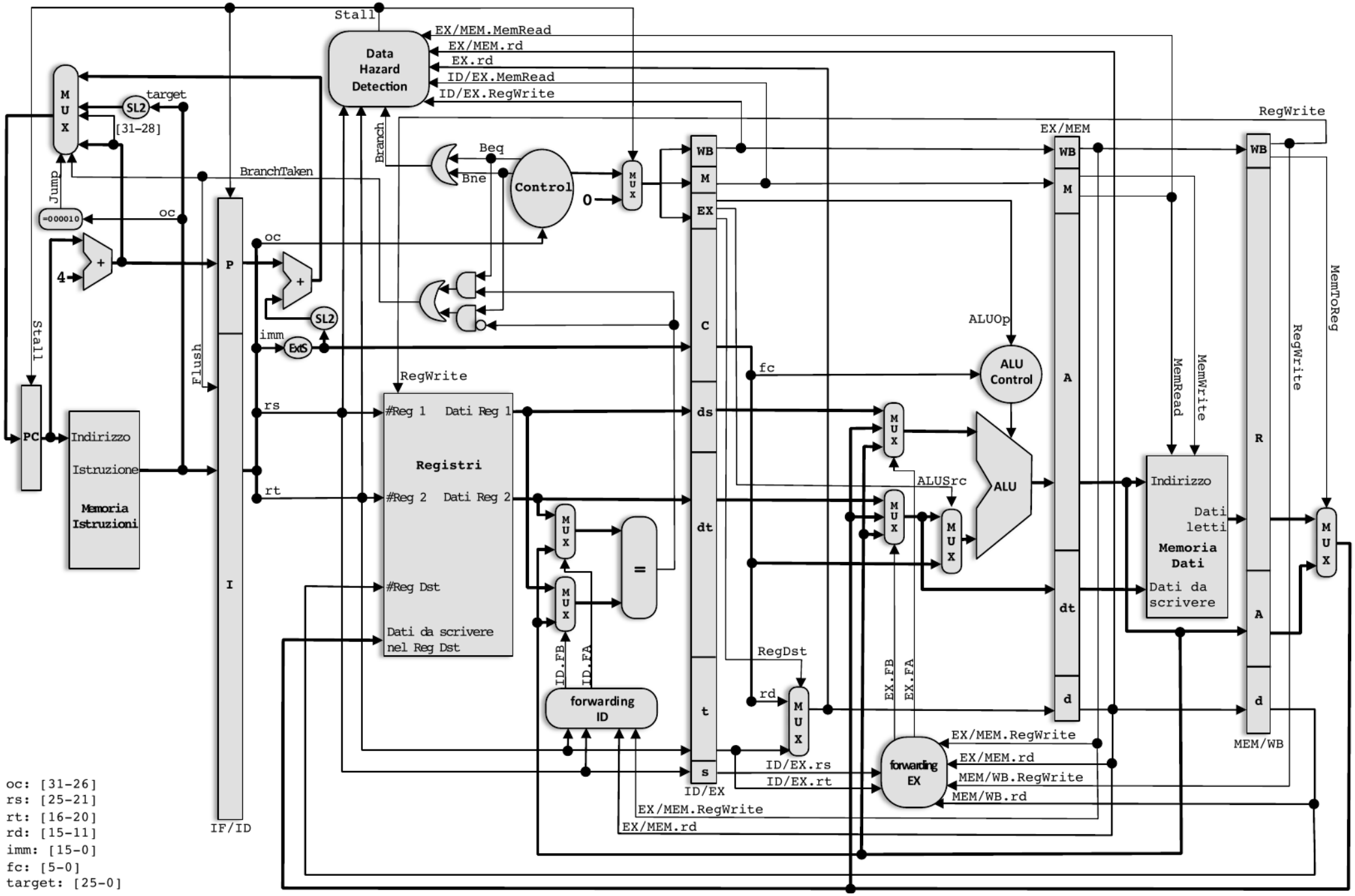
6) come riordinare le istruzioni per ridurre il numero di stalli al massimo (su foglio a parte)

7) quanti colpi di clock sarebbero necessari <u>nel codice ottimizzato</u>

8) quali sono le istruzioni contenute nei registri della pipeline durante il **13°** ciclo di clock <u>nel codice ottimizzato</u>

WB: MEM: EXE: ID: IF:

Implementazione pipeline di MIPS (solamente le istruzioni: `add`, `addi`, `sub`, `and`, `andi`, `or`, `ori`, `xor`, `xori`, `nor`, `slt`, `slti`, `lw`, `sw`, `beq`, `bne`, `j`).




```
oc: [31-26]
rs: [25-21]
rt: [16-20]
rd: [15-11]
imm: [15-0]
fc: [5-0]
target: [25-0]
```

Esame di Architetture – Canale AL – Prof. Sterbini – 1/7/16 – Compito B

Cognome e Nome: _____ Matricola: _____

Parte 1 (per chi non ha superato l'esonero)

Esercizio 1B. Si ha il dubbio che in una partita di CPU a ciclo di clock singolo (vedi sul retro) la Control Unit sia rotta, producendo il segnale di controllo **RegDst** attivo **solo quando** è attivo il segnale di controllo **AluSrc**.
Si assume che normalmente RegDst sia asserito solo per le istruzioni di tipo R, che MemToReg sia asserito solo per l'istruzione lw e che AluSrc sia asserito solo per le istruzioni lw e sw e di tipo immediato.

a) Si indichino *qui sotto* quali delle istruzioni base (**lw, sw, di tipo R, beq, j, R immediate**) funzioneranno male e qual'è il comportamento anomalo in caso di CU guasta.

b) si scriva *qui sotto* un breve programma assembly MIPS che termina valorizzando il registro $s0 con il valore 1 se il processore è guasto, altrimenti con 0.

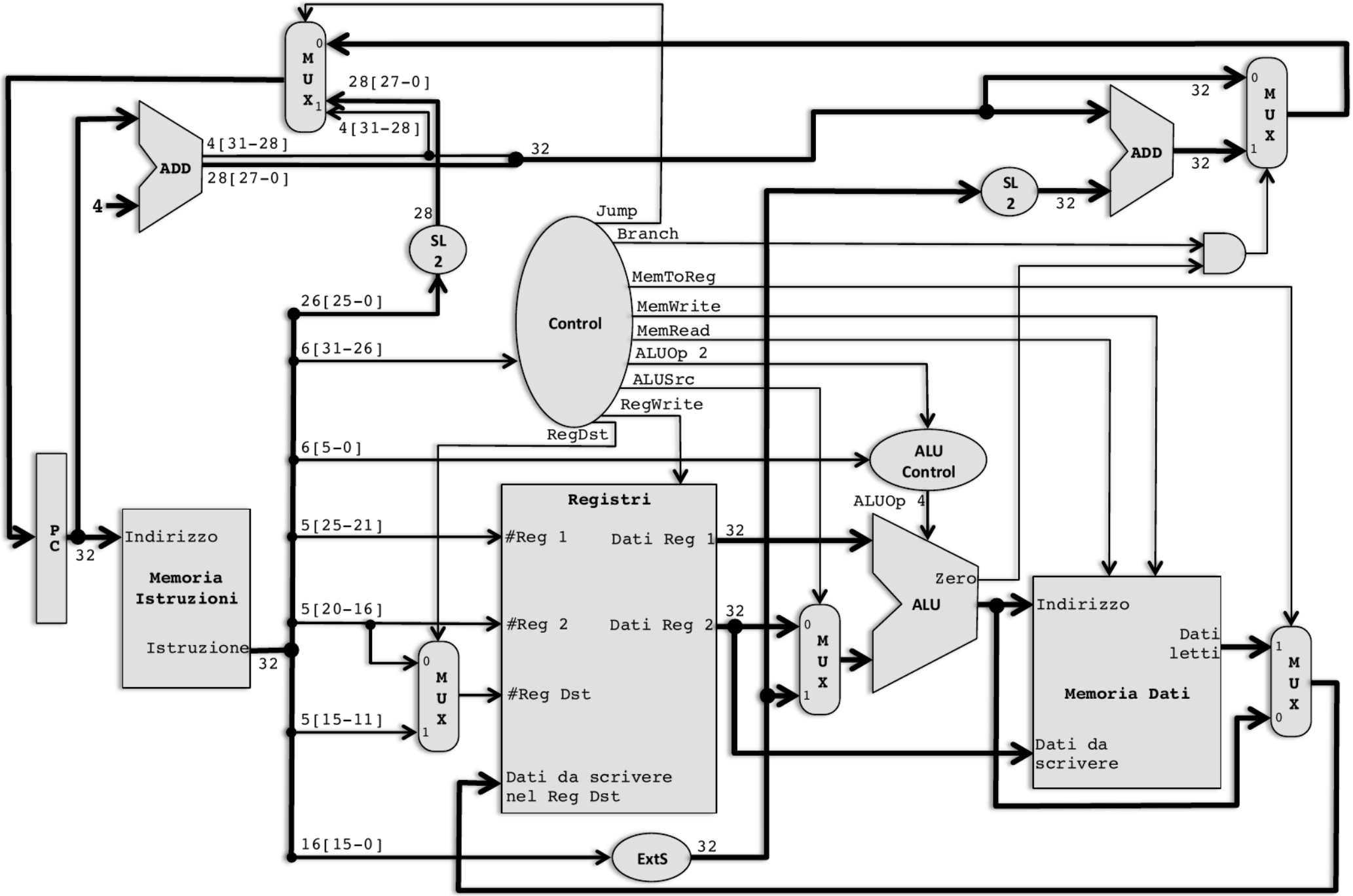
Esercizio 2B. Considerate l'architettura MIPS a ciclo singolo in figura (diagramma sul retro).
Si vuole aggiungere l'istruzione di tipo R **`b_mult rs, rt, label`** che esegue un salto alla label (relativo) se il valore del registro **rs** è un multiplo esatto del valore del registro **rt**. (non vi preoccupate del caso $rt=0)

1) modificate il diagramma mostrando gli eventuali altri componenti necessari a realizzare l'istruzione

2) indicate sul diagramma tutti i segnali di controllo che la CU genera per realizzare l'istruzione

3) supponendo che l'accesso alle memorie impieghi **33ns**, l'accesso ai registri **11ns**, le operazioni dell'ALU e dei sommatori **50ns**, e ignorando gli altri ritardi di propagazione dei segnali, indicate sul diagramma la durata totale del ciclo di clock per permettere l'esecuzione anche della nuova istruzione.

Implementazione ad un ciclo di clock di MIPS (solamente le istruzioni: `add`, `sub`, `and`, `or`, `xor`, `slt`, `lw`, `sw`, `beq`, `j`)



Esame di Architetture – Canale AL – Prof. Sterbini – 1/7/16 – Compito B

Cognome e Nome: _____ Matricola: _____

Parte 2 (per tutti)

Esercizio 3B. Si consideri l'architettura MIPS con pipeline mostrata in figura (sul retro) ed il frammento di programma qui a destra che calcola la somma degli elementi dispari della diagonale di una matrice 10x10 usando puntatori.
La matrice contiene in ordine per righe e colonne i valori ordinati da 0 a 99

```
.data
M:      .word 0, 1, 2, 3, 4, 5, . . . , 99
DIM:    .word 10
.text
main:   xor  $s0, $s0, $s0      # somma=0
        lw   $s1, DIM           # N
        addi $s2, $s1, 1        # N+1
        sll  $s2, $s2, 2        # 4(N+1)
        mul  $s1, $s1, $s1      # N^2
        sll  $s1, $s1, 2        # 4N^2
        la   $s3, matrice       # inizio di M
loop:   lw   $t0, ($s3)         # x=M[i]
        andi $t1, $t0, 1        # x%2
        beqz $t1, pari          # se pari
        add  $s0, $s0, $t0      # sum+=x
pari:   add  $s3, $s3, $s2      # prox el
        addi $s4, $s1, matrice  # fine di M
        blt  $s3, $s4, loop     #
        li   $v0, 1             # print
        move $a0, $s0           # la somma
        syscall                 #
```

Si indichino qui sotto o sul codice (PASSAGGI INCLUSI):
1) tra quali istruzioni sono presenti data hazard,

2) tra quali istruzioni sono presenti control hazard,

3) quanti cicli di clock sono necessari a eseguire il programma con il forwarding

4) quanti ne sarebbero necessari se il forwarding non esistesse

5) quali sono le istruzioni contenute nei registri della pipeline durante il **13°** ciclo di clock (con FW)

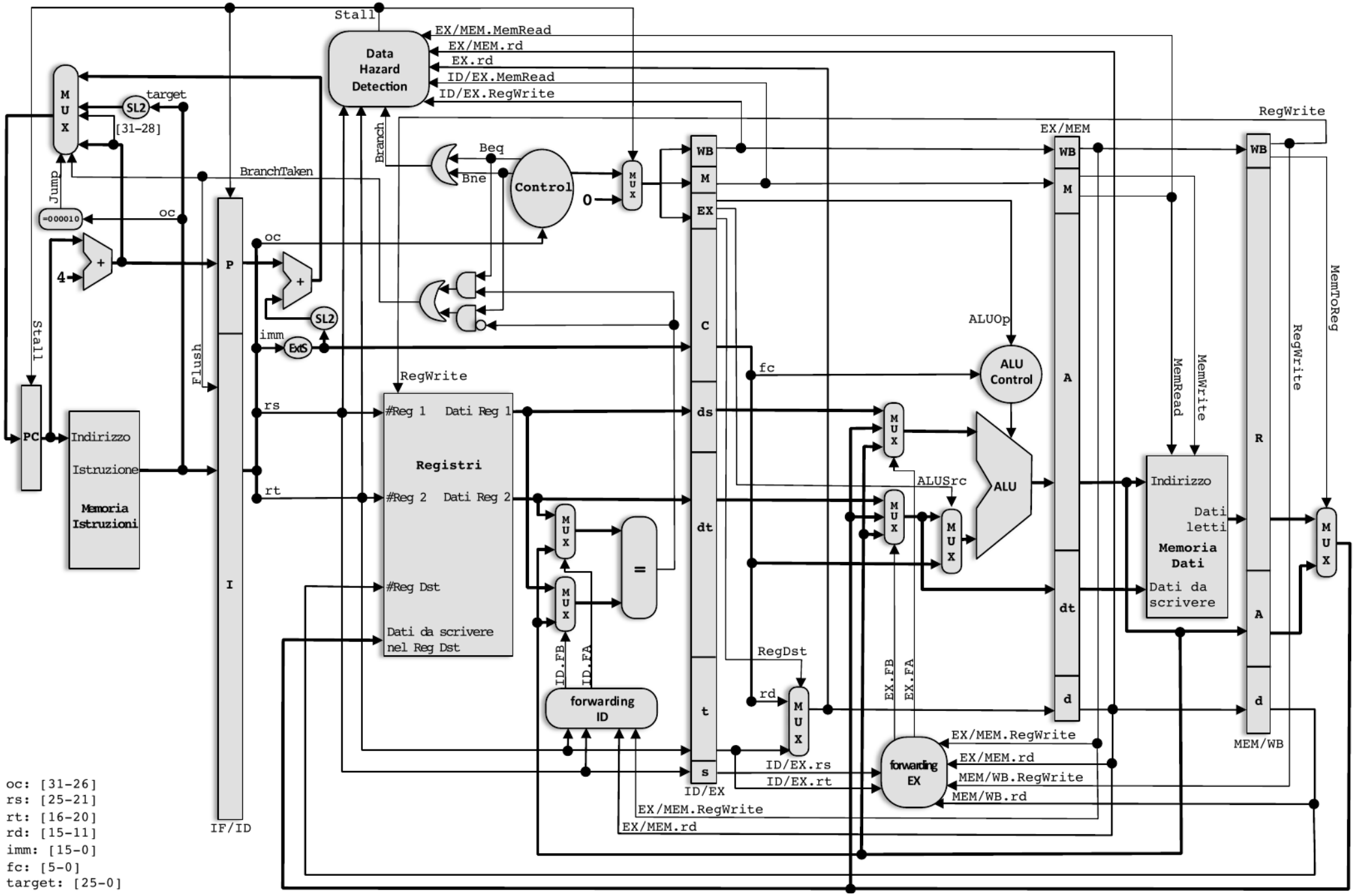
WB: MEM: EXE: ID: IF:

6) come riordinare le istruzioni per ridurre il numero di stalli al massimo (su foglio a parte)
7) quanti colpi di clock sarebbero necessari <u>nel codice ottimizzato</u>

8) quali sono le istruzioni contenute nei registri della pipeline durante il **13°** ciclo di clock <u>nel codice ottimizzato</u>

WB: MEM: EXE: ID: IF:

Implementazione pipeline di MIPS (solamente le istruzioni: `add`, `addi`, `sub`, `and`, `andi`, `or`, `ori`, `xor`, `xori`, `nor`, `slt`, `slti`, `lw`, `sw`, `beq`, `bne`, `j`).




```
oc: [31-26]
rs: [25-21]
rt: [16-20]
rd: [15-11]
imm: [15-0]
fc: [5-0]
target: [25-0]
```

1°	2°	3°

Correzioni ricevute

Esame di Architetture – Canale MZ – 1/7/16 – Compito A

Inserite qui solo la vostra matricola: _____

Esercizio 4A (14 punti).
Considerate un sistema con due livelli di cache: **CPU <=> L1 <=> L2 <=> RAM**
- L1 è una cache **2-way** set-associativa con **4 set** e blocchi grandi **4 word** e strategia di rimpiazzo **LRU**.
- L2 è una cache **1-way** set-associativa con **2 set** e blocchi grandi **32 word** e strategia di rimpiazzo **LRU**.

1) Supponendo che gli indirizzi siano da 32 bit (indirizzamento al byte) e che all'inizio nessuno dei dati sia in cache, indicate quali degli accessi in memoria più sotto sono hit o miss in ciascuna delle due cache
2) per ciascuna MISS indicate se è di tipo **Caricamento (L), Capacità (Cap) o Conflitto (Conf)**

	Address	**1120**	**531**	**129**	**1099**	**1100**	**2056**	**319**	**445**	**2060**	**368**	**131**	**318**	**130**	**528**
L1	**Block#**														
	Index														
	Tag														
	Hit/Miss														
	Tipo miss														
L2	**Block#**														
	Index														
	Tag														
	Hit/Miss														
	Tipo miss														

3) calcolate le dimensioni in bit delle due cache L1, L2 compresi i bit di controllo
Scrivete le soluzioni CON I PASSAGGI qui sotto continuando se necessario sul retro del foglio

Dimensioni L1:

Dimensioni L2:

4) assumendo che il processore vada a **4Ghz** con **3 CPI** (Clock Per Instruction), che gli accessi in memoria impieghino **25ns**, che gli hit nella cache L1 impieghino **1ns** e gli hit nella cache L2 impieghino **5ns**, calcolate il **tempo medio** per questa sequenza di accessi e **quante istruzioni** vengono svolte nel tempo calcolato.
Scrivete le soluzioni CON I PASSAGGI qui sotto continuando se necessario sul retro del foglio

Tempo totale:

Tempo medio:

Numero di istruzioni eseguite nel tempo medio:

1°	2°	3°

Correzioni ricevute

Esame di Architetture – Canale MZ – 1/7/16 – Compito B

Inserite qui solo la vostra matricola: _____

Esercizio 4B (14 punti).
Considerate un sistema con un livello di cache e VM con TLB: **CPU <=> L1 <=> MMU <=> RAM**
v
TLB <=> Page Table

- la cache è **1-way** set-associativa con **8 set** e blocchi grandi **8 word** e strategia di rimpiazzo **LRU**.
- il TLB è **2-way** set associativo con **4 set** e blocco grande 1 linea di page table e strategia di rimpiazzo **LRU**.
- la memoria fisica disponibile è di **2 sole pagine** da **1024 byte** ciascuna, con strategia di rimpiazzo **LRU**.

1) Supponendo che gli indirizzi siano da 32 bit (indirizzamento al byte) e che all'inizio nessuno dei dati sia in cache, e che la **cache agisca sugli indirizzi VIRTUALI**, indicate quali degli accessi in memoria più sotto sono hit o miss nella cache e nel TLB e quali page fault vengono generati.
2) per ciascuna MISS indicate se è di tipo **Caricamento (L), Capacità (Cap) o Conflitto (Conf)**

Address	**1120**	**531**	**129**	**1099**	**1100**	**2056**	**3391**	**445**	**2060**	**368**	**131**	**318**	**130**	**528**
#block														
tag														
index														
H/M														
Tipo miss														
#pag.														
TLB tag														
TLB index														
H/M TLB														
Tipo miss														
P.Fault?														

Corrispondenza tra #pagina v. e #pagina f. (da mettere in colonna sotto la richiesta della pagina calcolata sopra)

#p. virt.														
#p. fis.														

3) assumendo che il processore vada a **2Ghz** con **5 CPI** (Clock Per Instruction), che gli accessi in memoria impieghino **45ns**, che gli hit nella cache e in TLB impieghino **5ns**, calcolate il **tempo medio** per questa sequenza di accessi e **quante istruzioni** vengono svolte nel tempo calcolato (IGNORANDO IL TEMPO PER PAGE FAULT).
Scrivete qui sotto le soluzioni CON I PASSAGGI continuando se necessario sul retro del foglio

Tempo totale:

Tempo medio:

Numero di istruzioni eseguite nel tempo medio:

Esame di Architetture – Canale AL – Prof. Sterbini – 1/7/16 – Compito A

Cognome e Nome: _____ Matricola: _____

Parte 3 (assembler)

Esercizio 5A.

1) si realizzi la funzione **conta_bit_uguali(X,Y) RICORSIVA** che riceve due interi X ed Y maggiori di zero e ne calcola il numero di bit uguali nella stessa posizione come segue:

conta_bit_uguali(X, Y) = 0 **se X=0 oppure Y=0** **(solo quando XoY sono stati esaminati tutti)**

conta_bit_uguali(X, Y) = conta_bit_uguali(X/2, Y/2) + 1 **se (X and 1) = (Y and 1)** **(LSB uguale)**

conta_bit_uguali(X, Y) = conta_bit_uguali(X/2, Y/2) **altrimenti** **(LSB diverso)**

NOTA: gli zeri non significativi (nella parte alta del numero) non vengono confrontati

2) si realizzi il programma main che:

legge due interi X ed Y (nei test saranno sempre maggiori di zero)

ne calcola il numero di bit uguali chiamando la funzione **conta_bit_uguali(X, Y)**

lo stampa

Esempio di esecuzione ricorsiva

conta_bit_uguali(22, 20) = conta_bit_uguali(11, 10) + 1

= (conta_bit_uguali(5, 5) + 1)

= ((conta_bit_uguali(2, 2) + 1) + 1)

= (((conta_bit_uguali(1, 1) + 1) + 1) + 1)

= ((((conta_bit_uguali(0, 0) + 1) + 1) + 1) + 1)

= ((((0 + 1) + 1) + 1) + 1)

= (((1 + 1) + 1) + 1)

= ((2 + 1) + 1)

= (3 + 1)

= 4

infatti 22=<u>10110</u> **hanno 4 bit uguali nella stessa posizione (sottolineati)**

20=<u>10100</u>

oppure 44=101<u>100</u> **hanno 3 bit uguali nella stessa posizione (sottolineati)**

20= 10<u>100</u>

Una possibile versione iterativa può essere realizzata ciclando sui bit e confrontandoli

Esame di Architetture – Canale AL – Prof. Sterbini – 1/7/16 – Compito B

Cognome e Nome: _____ Matricola: _____

Parte 3 (assembler)

Esercizio 5B.

1) si realizzi la funzione **conta_bit_diversi(X,Y) RICORSIVA** che riceve due interi X ed Y maggiori di zero e ne calcola il numero di bit diversi nella stessa posizione come segue:

conta_bit_diversi(X, Y) = 0 se X=0 e Y=0 (solo quando XeY sono stati esaminati tutti)

conta_bit_diversi(X, Y) = conta_bit_diversi(X/2, Y/2) + [(X and 1) xor (Y and 1)] altrimenti

2) si realizzi il programma main che:

legge due interi X ed Y (nei test saranno sempre maggiori di zero)

ne calcola il numero di bit uguali chiamando la funzione **conta_bit_diversi(X, Y)**

lo stampa

Esempio di esecuzione ricorsiva

conta_bit_diversi(22, 20)

= conta_bit_diversi(11, 10) + (0 xor 0)

= (conta_bit_diversi(5, 5) + (1 xor 0)) + 0

= ((conta_bit_diversi(2, 2) + (1 xor 1)) + 1) + 0

= (((conta_bit_diversi(1, 1) + (0 xor 0)) + 0) + 1) + 0

= ((((conta_bit_diversi(0, 0) + (1 xor 1)) + 0) + 0) + 1) + 0

= ((((0 + 0) + 0) + 0) + 1) + 0

= (((0 + 0) + 0) + 1) + 0

= ((0 + 0) + 1) + 0

= (0 + 1) + 0

= 1 + 0

= 1

infatti 22=10110 hanno un solo bit diverso (sottolineato)

20=10100

oppure 44=101100 hanno 3 bit diversi nella stessa posizione (sottolineati)

20=010100

Una possibile versione iterativa può essere realizzata ciclando sui bit e confrontandoli

Modulo di raccolta dati - PEER-ASSESSMENT – 10-6-16 – Compito A

Studente N° (**INSERISCI LA TUA MATRICOLA**): _____

Queste risposte NON saranno usate per valutare i compiti vostri o dei colleghi.

Istruzioni - Per ogni compito da valutare:

- **NON SCRIVETE ASSOLUTAMENTE NULLA SUL COMPITO CHE STATE VALUTANDO!**
- **Copiate la matricola del compito esaminato nella prima colonna**
- **Se necessario usate questo foglio per prendere appunti (sul retro)**
- Date una valutazione **in 14esimi** analizzando solo la correttezza della risposta ovvero:
  - la presenza di eventuali errori, indicandoli nello spazio apposito (e NON sul compito!)
  - la loro relativa gravità con la seguente penalizzazione: **-1=errore grave, -0.5=non grave, -0=svista**
- **Una volta data la valutazione aggiungete una crocetta sullo spazio apposito del compito esaminato**
- Se un errore di un certo tipo è ripetuto più volte lo si consideri solo una volta
- Se dopo un errore ci sono passaggi svolti correttamente (anche se su dati errati) i passaggi corretti vanno considerati come corretti (anche se il risultato finale è errato)
- un compito radicalmente errato vale 0, una parte non svolta almeno -2

Compito N°	**Voto/14**	**Motivazione/appunti/errori individuati/peso di ciascun errore**
1°		
2°		
3°		

Modulo di raccolta dati - PEER-ASSESSMENT – 10-6-16 – Compito B

Studente N° (**INSERISCI LA TUA MATRICOLA**): _____

Queste risposte NON saranno usate per valutare i compiti vostri o dei colleghi.

Istruzioni - Per ogni compito da valutare:

- **NON SCRIVETE ASSOLUTAMENTE NULLA SUL COMPITO CHE STATE VALUTANDO!**
- **Copiate la matricola del compito esaminato nella prima colonna**
- **Se necessario usate questo foglio per prendere appunti (sul retro)**
- Date una valutazione **in 14esimi** analizzando solo la correttezza della risposta ovvero:
  - la presenza di eventuali errori, indicandoli nello spazio apposito (e NON sul compito!)
  - la loro relativa gravità con la seguente penalizzazione: **-1=errore grave, -0.5=non grave, -0=svista**
- **Una volta data la valutazione aggiungete una crocetta sullo spazio apposito del compito esaminato**
- Se un errore di un certo tipo è ripetuto più volte lo si consideri solo una volta
- Se dopo un errore ci sono passaggi svolti correttamente (anche se su dati errati) i passaggi corretti vanno considerati come corretti (anche se il risultato finale è errato)
- un compito radicalmente errato vale 0, una parte non svolta almeno -2

Compito N°	**Voto/14**	**Motivazione/appunti/errori individuati/peso di ciascun errore**
1°		
2°		
3°		