

# CALCOLATORI ELETTRONICI B – 30 agosto 2010

**NOME:**

**COGNOME:**

**MATR:**

Scrivere chiaramente in caratteri maiuscoli a stampa

1. La seguente figura riporta uno schema di massima del datapath corrispondente all'implementazione con pipeline a 5 stadi (F: Fetch, D: Decode, E: Execute, M: Mem, W: Write-Back) di un processore MIPS. Lo schema include (almeno parzialmente) anche l'hardware per la gestione delle eccezioni dovute a overflow aritmetico: nel caso in cui il segnale di overflow in uscita dalla ALU (non riportato nello schema) sia attivo, il controllo viene passato alla procedura di gestione posta all'indirizzo (esadecimale) 80000180.

Con riferimento a tale figura, si chiede di:

- precisare la funzione dei segnali IF.Flush, ID.Flush, Ex.Flush
- precisare la funzione del registro EPC (Exception Program Counter)

Si consideri inoltre il seguente frammento di codice MIPS:

```
add    $t0, $s1, $s1
```

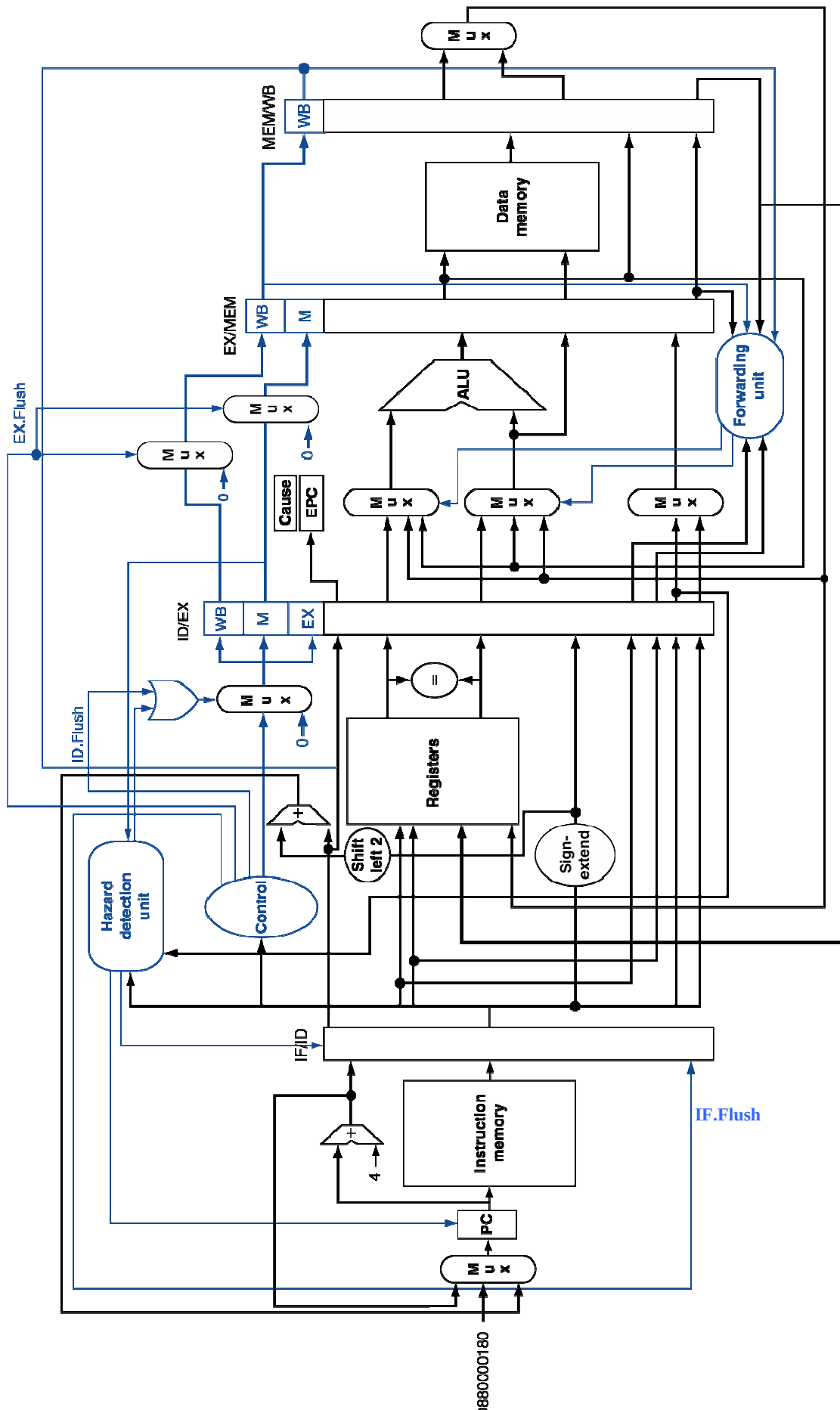
```
sub     $t0, $t0, $s1
```

```
sub     $s2, $t0, $s2
```

```
sw      $s2, 20($s1)
```

```
add     $s1, $s2, $s1
```

Si supponga che la terza istruzione (`sub $s2, $t0, $s2`) generi overflow aritmetico. Si dettagli con la maggior precisione possibile come viene gestito l'overflow (anche in relazione alle altre istruzioni presenti in pipeline) e quali segnali vengono attivati nel datapath allo scopo. [5]





2. Si consideri il seguente frammento di codice MIPS:

```
add    $s1, $t0, $s1  
sub     $s1, $t0, $s1  
lw      $s2, 20($s1)  
sw      $s2, 20($s1)  
add     $s1, $s2, $s1
```

Si consideri l'implementazione con pipeline a 5 stadi (F: Fetch, D: Decode, E: Execute, M: Mem, W: Write-Back). Si chiede di:

a) individuare in modo preciso tutte le dipendenze tra i dati

b) tracciare il diagramma temporale delle istruzioni (indicando esplicitamente le eventuali propagazioni e, per ognuna di esse, quale dato è propagato) in ognuna delle seguenti ipotesi:

- non è disponibile alcuna unità di propagazione
- è disponibile un'unità di propagazione verso lo stadio E
- è disponibile un'unità di propagazione verso lo stadio E ed una verso lo stadio M. [6]

3. Si consideri il processore dell'esercizio precedente, dotato di pipeline a 5 stadi, che disponga di un'unità di propagazione solamente verso lo stadio E (come nel secondo caso al punto b dell'esercizio precedente).

Il processore utilizza una cache primaria distinta per i dati e le istruzioni, mentre non dispone di cache secondaria. La cache, che in caso di successo consente di accedere all'istruzione o al dato in un ciclo di clock, presenta le seguenti caratteristiche:

- percentuale di successo (hit rate): 90% per le istruzioni, 80% per i dati
- penalità di fallimento (sia in scrittura che in lettura): 10 cicli di clock

Si assuma un carico di lavoro che prevede la seguente distribuzione delle istruzioni MIPS:

|            |      |
|------------|------|
| lw:        | 25 % |
| sw:        | 20 % |
| formato-R: | 35 % |
| beq:       | 15 % |
| j:         | 5 %  |

tale che:

- il 25% delle istruzioni che seguono lw utilizzano il risultato nello stadio E (ed eventualmente anche in M), il 25% utilizzano il risultato solo nello stadio M ed il rimanente 50% non ne utilizzano il risultato;
- il 10.6% delle istruzioni che seguono istruzioni di Tipo-R ne utilizzano il risultato nello stadio E, il 10.49% nello stadio M ed il rimanente 78.91% non ne utilizzano il risultato.

Trascurando le criticità sui salti ma tenendo conto dei miss di cache e delle criticità sui dati, si calcoli il CPI (numero medio di cicli di clock per istruzione) ottenuto. [3]

4. Si consideri il seguente frammento di codice assembler (ispirato al MIPS):

```
add    $t1, $s1, $s2
beq     $t1, 20($t2), Dest
add     $t3, $t4, $t5
...
Dest: lw    $t1, 24($t2)
sub     $t1, $t1, $t5
```

costituito da istruzioni MIPS ad eccezione di *beq*, che a differenza dell'usuale istruzione di salto condizionato verifica l'uguaglianza di un registro e di una locazione di memoria (in luogo dell'uguaglianza di due registri):

*beq rs, offset(rt), Dest*                      SE  $rs = M[rt + \text{offset}]$  ALLORA  $PC \leftarrow \text{Dest}$

L'implementazione del processore è realizzata con una pipeline a 5 stadi (F: Fetch, D: Decode, E: Execute, M: Mem, W: Write-Back) che per la gestione delle criticità sui dati dispone di una unità di propagazione verso E.

L'istruzione *beq* è implementata in modo che l'indirizzo di destinazione sia calcolato nello stadio D; per il confronto degli operandi si utilizza un comparatore molto veloce, in modo che l'aggiornamento del program counter possa essere effettuato nello stesso stadio in cui gli operandi sono disponibili per il confronto (ovviamente, l'aggiornamento avviene come al solito alla fine del ciclo di clock).

Assumendo di utilizzare per i salti condizionati la predizione statica demandata al compilatore ed ipotizzando che, nel caso considerato, il compilatore abbia previsto per l'istruzione *beq* che il salto venga effettuato, si chiede di:

- tracciare il diagramma temporale nel caso di previsione corretta (salto effettuato)
- tracciare il diagramma temporale nel caso di previsione errata (salto non effettuato)
- indicare in entrambi i casi l'eventuale numero di cicli di penalità

[6]

5. Si illustri sinteticamente la tecnica del salto ritardato per la gestione delle criticità sul controllo nell'implementazione con pipeline.  
Se un'istruzione di salto aggiorna il program counter nel secondo stadio della pipeline, quanti slot di ritardo occorrono? Perché? [2]

6. Si consideri un processore che genera indirizzi (virtuali) di 24 bit, dotato di un semplice TLB in grado di memorizzare soltanto 4 elementi. Si consideri in particolare un processo in esecuzione per il quale il TLB e la tabella delle pagine siano correntemente come nella figura seguente:

bit di  
validità

TLB

|   |                   |          |
|---|-------------------|----------|
| 1 | 00000000000000011 | 10000000 |
| 1 | 00000000000000010 | ?        |
| 0 | 00000000000000011 | 10000000 |
| 0 | 00000000000000010 | 10000001 |

Tabella delle pagine

bit di  
validità

|   |          |
|---|----------|
| 1 | 11111110 |
| 1 | 11111111 |
| 1 | 00000000 |
| ? | ?        |
| 1 | 00010010 |
| 0 | .        |
| 0 | .        |
| . | .        |
| . | .        |
| 0 | .        |

Si chiede di:

- specificare il numero di elementi della tabella delle pagine (motivando la risposta)
- specificare i tre elementi che, nel TLB e nella tabella delle pagine, occupano i tre campi contrassegnati da punti di domanda
- Supponendo che vengano generati i tre indirizzi virtuali seguenti:

0000000000000001100000000  
 0000000000000000000000010  
 111111111111111100000000

per ognuno di essi individuare quali miss vengono generati (miss di TLB e/o miss di pagina) e (se la pagina fisica è presente in memoria) specificare l'indirizzo fisico generato. [5]

7. Con riferimento a due ipotetici segnali REQUEST (inviato dal master allo slave) e ACK (inviato dallo slave al master) si illustri sinteticamente un'ipotetica operazione di lettura di un dato in un bus asincrono utilizzando il protocollo di handshaking, evidenziando in modo preciso il significato dei segnali e le relazioni di causa-effetto tra le loro transizioni. [3]

