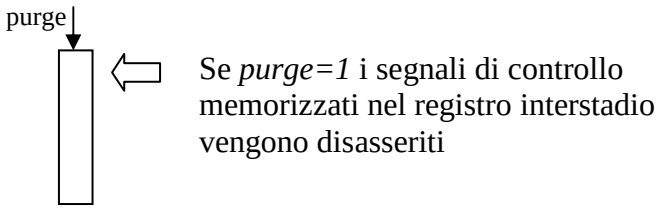
CALCOLATORI ELETTRONICI B – 12 settembre 2008

NOME:	**COGNOME:**	**MATR:**
Scrivere chiaramente in caratteri maiuscoli a stampa		

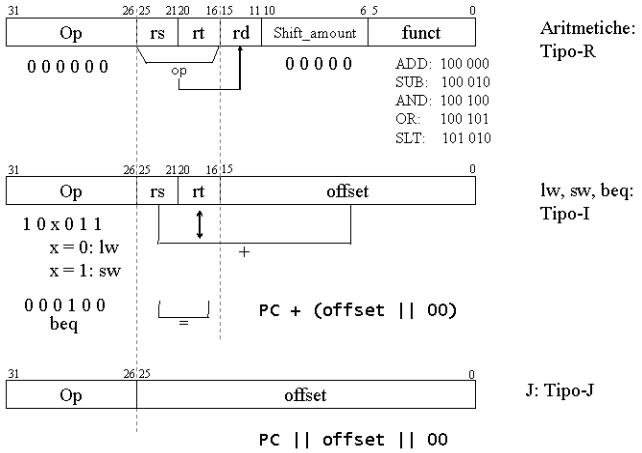
1. Si chiede di illustrare l'implementazione, nel contesto di un processore con pipeline a 4 stadi (F, D, E, W), della sola seguente istruzione di salto condizionato:

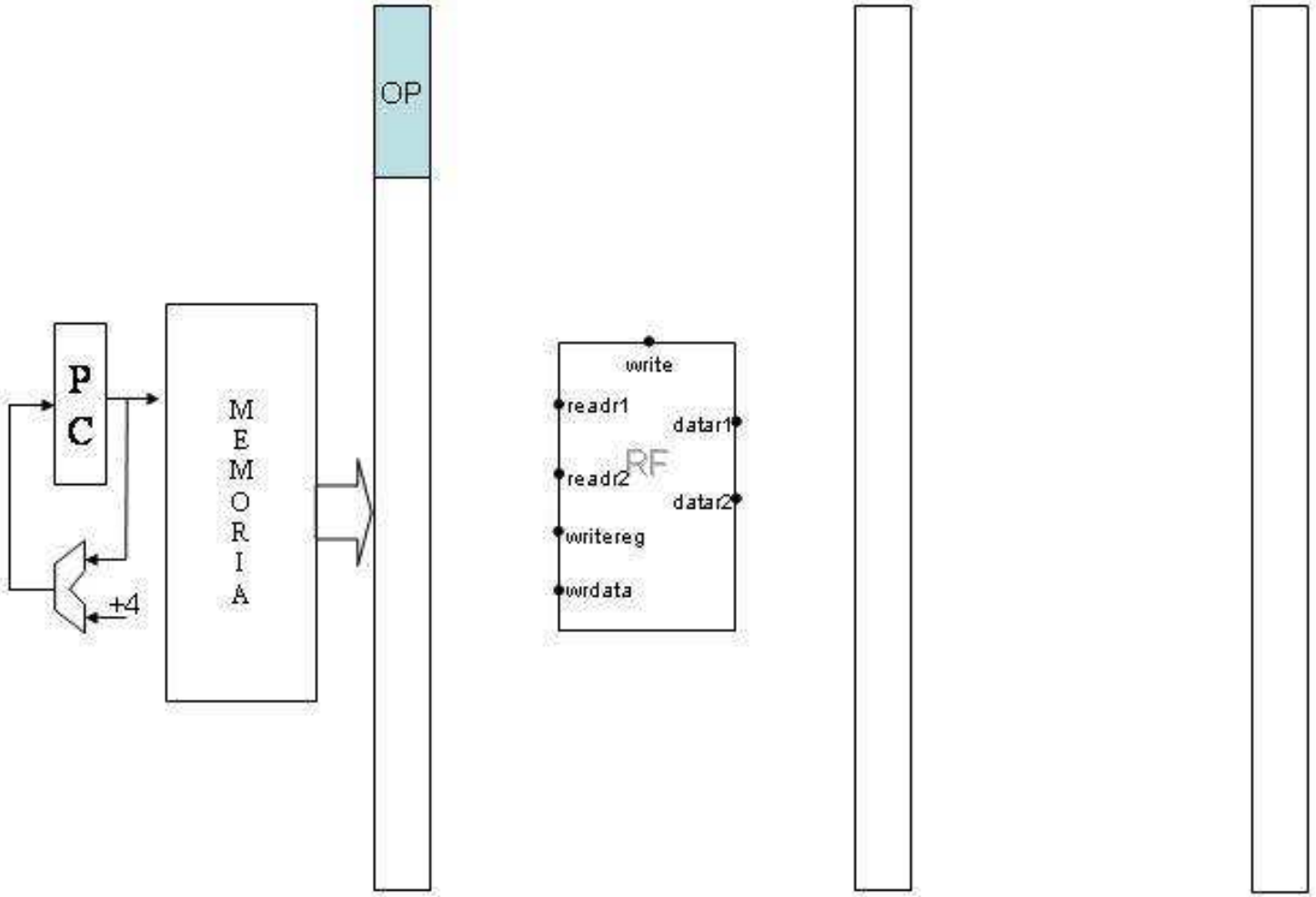
   bzero offset(r1), r2 // IF M[r1+offset]=0 THEN PC←r2

   che salta all'indirizzo indicato nel registro r2 se la locazione di memoria di indirizzo r1+offset (cfr. le istruzioni lw e sw) contiene il valore 0.
   Per la predizione dei salti, il processore utilizza la semplice tecnica di predizione di "salto non effettuato" (carica le istruzioni successive al salto e, nel caso in cui il salto sia effettuato, le elimina).
   Nella pagina seguente è riportato uno schema incompleto di datapath, in cui il Register File lavora secondo le modalità usuali. Si chiede di:
   1) riportare il formato della nuova istruzione macchina (specificando anche i campi destinati rispettivamente a r1 e r2);
   2) completare il datapath, introducendo le opportune unità funzionali e i relativi collegamenti con i registri interstadio. Le unità funzionali che possono essere inserite sono memoria dati, ALU, multiplexer e porte logiche (AND, OR, NOR, …) con qualunque numero di ingressi.
   Si ricordi di includere nel datapath la logica per la gestione della predizione del salto. A questo scopo, si assuma che i registri interstadio possano disporre di un segnale di ingresso "purge" che, se affermato, disasserisce tutti i segnali di controllo creando una bolla:



   3) disegnare nel datapath l'unità di controllo (che può essere schematizzata con un ovale) e i relativi collegamenti;
   4) disegnare nel datapath i collegamenti dei segnali di controllo negli stadi D, E, W;
   5) indicare la specifica dell'unità di controllo (solo per l'istruzione *bzero*). [6]




OP
P
C
+4
MEMORIA
write
readr1
datar1
RF
readr2
datar2
writereg
wrdata

2. Nel contesto della pipeline a 4 stadi del precedente esercizio, si consideri la gestione delle eccezioni dovute alla presenza di una istruzione indefinita (ovvero, il cui codice operativo non appartiene all'insieme previsto nell'Instruction Set Architecture). Si supponga che il processore, al verificarsi di una eccezione di questo tipo, debba:
- scrivere in un registro chiamato EPC il valore del program counter
- saltare all'indirizzo (prefissato) della routine di gestione dell'eccezione.

Mediante uno schema da disegnare di seguito, si estenda il datapath ottenuto nel precedente esercizio limitatamente ai primi due stadi (fetch e execute), includendo la logica di gestione delle eccezioni dovute ad istruzioni indefinite. [4]

3. Si consideri il processore con pipeline a quattro stadi che implementa l'istruzione *bzero* descritta nell'esercizio 1 ed utilizza la logica di gestione delle eccezioni per istruzioni indefinite individuata nell'esercizio 2. Si consideri il seguente frammento di codice:

I_1: bzero 20(r1), r2
I_2: <istruzione indefinita>

Supponendo che l'istruzione I_1 effettui il salto, quale problema si presenta? Suggerire una possibile soluzione implementativa per risolvere questo problema. [3]

4. Si consideri il seguente frammento di codice MIPS:

```
sub   $s1, $s2, $s1
sw    $s1, 40($s2)
lw    $s1, 20($s1)
lw    $s2, 40($s1)
add   $s1, $s2, $s2
```

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:
- 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.

Nei diagrammi, si chiede di indicare il numero di cicli di penalità. [4]

5. Si consideri un processore MIPS, implementato tramite pipeline a 5 stadi, che disponga di una cache primaria distinta per i dati e le istruzioni (il processore non dispone invece di cache secondaria). La cache presenta le seguenti caratteristiche:
- numero di cicli di clock richiesti in caso di successo (hit): 1
- percentuale di successo (hit rate): 90% per le istruzioni, 70% per i dati
- penalità di fallimento in scrittura: 5 cicli di clock
- penalità di fallimento in lettura: 10 cicli di clock

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

lw:	30 %
sw:	10 %
Tipo-R:	30 %
beq:	20 %
j:	10 %

Si supponga inoltre che:
- il 40% delle istruzioni Tipo-R siano seguite da istruzioni che ne utilizzano il risultato (la metà lo utilizzano nello stadio E, l'altra metà nello stadio M);
- il 40% delle istruzioni lw siano seguite da istruzioni che ne utilizzano il risultato, tra le quali:
  - > il 40% sono istruzioni Tipo-R
  - > il 20% sw che utilizzano il risultato solo per il calcolo dell'indirizzo
  - > il 10% sw che utilizzano il risultato solo per immagazzinarlo in memoria
  - > il 10% sw che utilizzano il risultato sia per il calcolo dell'indirizzo sia per immagazzinarlo in memoria
  - > il rimanente 20% sono istruzioni lw che ne utilizzano il risultato

Tenendo conto dei miss di cache e delle criticità sui dati, si calcoli il CPI (numero medio di cicli di clock per istruzione) ottenuto nei due casi seguenti:
- si dispone di un'unità di propagazione solo verso lo stadio E
- si dispone di un'unità di propagazione verso lo stadio E ed una verso lo stadio M. [4]

6. Si consideri un sistema di memoria virtuale con le seguenti caratteristiche:
- indirizzi virtuali di 40 bit
- dimensione delle pagine: 16 KB
- memoria fisica indirizzata: 4 GB
- bit di controllo utilizzati per ciascuna riga della tabella delle pagine: 6

Si calcoli la dimensione della tabella delle pagine per ciascun processo, assumendo che le pagine virtuali siano tutte utilizzate. [2]

7. Si descriva la tecnica di predizione dinamica dei salti che utilizza BTB (Branch Table Buffer), illustrandone le funzioni, i vantaggi e sinteticamente la realizzazione. [3]

8. E’ dato un bus asincrono che collega un processore P e diversi dispositivi slave. Il bus è costituito da *n* linee dati, *m* linee indirizzi e dalle linee di controllo seguenti:

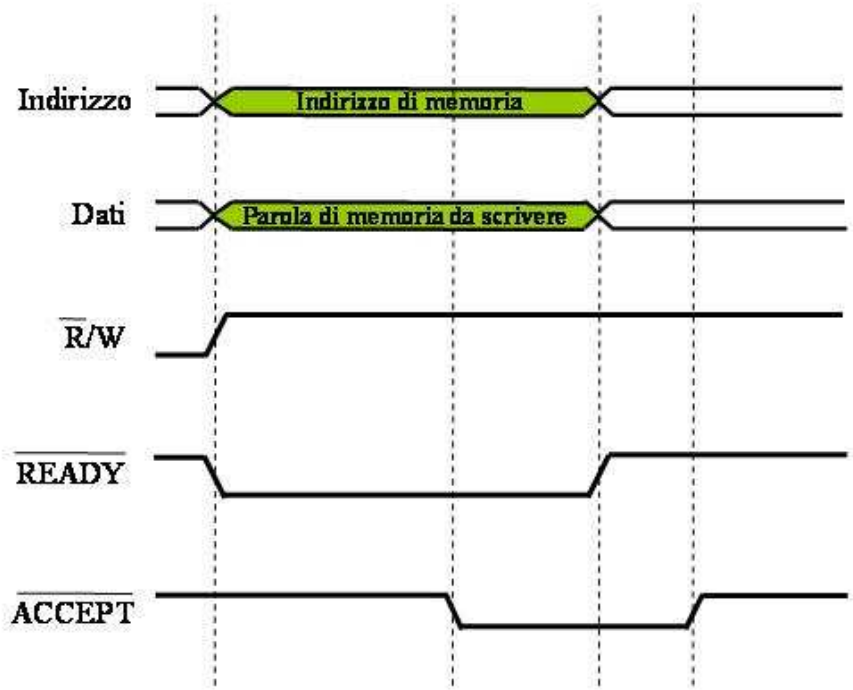
$\overline{R}$/W: utilizzato dal processore P per segnalare una richiesta di lettura (se basso) o di scrittura (se alto).

$\overline{\text{READY}}$: utilizzato dal processore P per segnalare una richiesta di trasferimento.

$\overline{\text{ACCEPT}}$: utilizzato dal dispositivo indirizzato per segnalare il completamento del trasferimento richiesto.

I segnali di controllo READY e ACCEPT sono attivi a livello basso.

La figura seguente riporta l’evoluzione temporale di un’operazione di trasferimento di una parola dal processore P al dispositivo slave (scrittura).



Si chiede di:

a) Mostrare in un diagramma temporale come può avvenire un’operazione di lettura (dal dispositivo slave al processore P), illustrando le relazioni tra i segnali.
b) Specificare le macchine a stati finiti che controllano l’esecuzione, sia per il master sia per lo slave, del protocollo di handshaking in lettura di cui al punto a). [6]