CALCOLATORI ELETTRONICI A – 8 aprile 2009

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

1. Utilizzando la green card tradurre in codice macchina le seguenti istruzioni assembly MIPS:

```
E1:   sw $t0, 16($t1)
      addi $t1, $t1, 4
      addi $t0, $t0, -1
      bne $t0, $zero, E1
```

[3]

Soluzione

101011 01001 01000 0000000000010000
001000 01001 01001 0000000000000100
001000 01000 01000 1111111111111111
000101 01000 00000 1111111111111100 (nb: l'offset è -4)

2. Si traduca la seguente pseudoistruzione in una o più istruzioni in assembly MIPS disponibili direttamente nell'ISA:

```
bge  $s1, $s2, L1
```

[2]

Soluzione

```
slt $at, $s1, $s2
beq $at, $zero, $L1
```

3. Scrivere una procedura in assembler MIPS corrispondente alla seguente funzione ricorsiva espressa in linguaggio C. Si utilizzino le note convenzioni sui registri. [5]

```
int F(int a, int b){
        if(a>b || a<10) return a;
          else return F(b-a, 2a);
        }
```

Soluzione

```
F:      addi $sp, $sp, -4
        sw $ra, 0($sp)

        slt $t0, $a1, $a0
        bne $t0, $zero, IF
        slti $t0, $a0, 10
        bne $t0, $zero, IF

        add $t0, $zero, $a0
        sub $a0, $a1, $a0
        add $a1, $t0, $t0
        jal F
        j FINE

IF:     add $v0, $a0, $zero

FINE:   lw $ra, 0($sp)
        addi $sp, $sp, 4
        jr $ra
```

4. Illustrare la funzione del registro frame pointer (`$fp`) nella gestione delle procedure. [2]

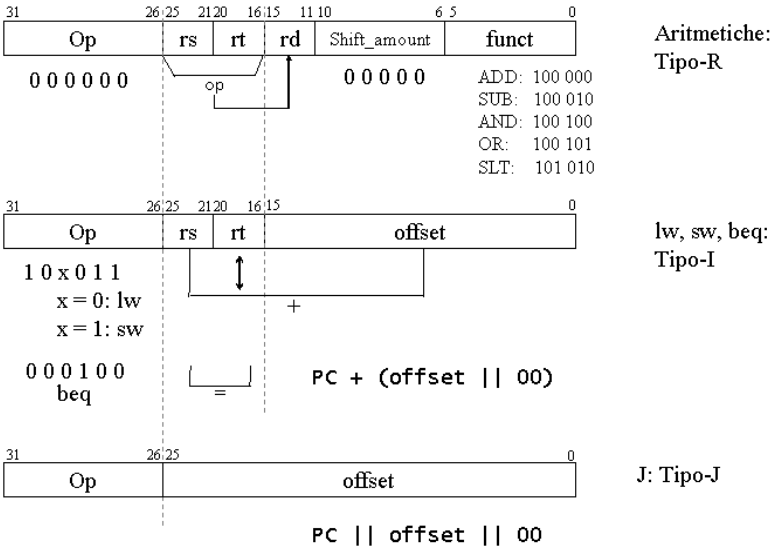
5. Si considerino, mostrati nelle figure alla pagina seguente, il datapath ed il diagramma a stati finiti che specifica l'unità di controllo secondo la tecnica a multiciclo relativamente alle istruzioni MIPS *lw*, *sw*, *beq*, *j* ed alle istruzioni *Tipo-R.*
Si vuole implementare la nuova istruzione

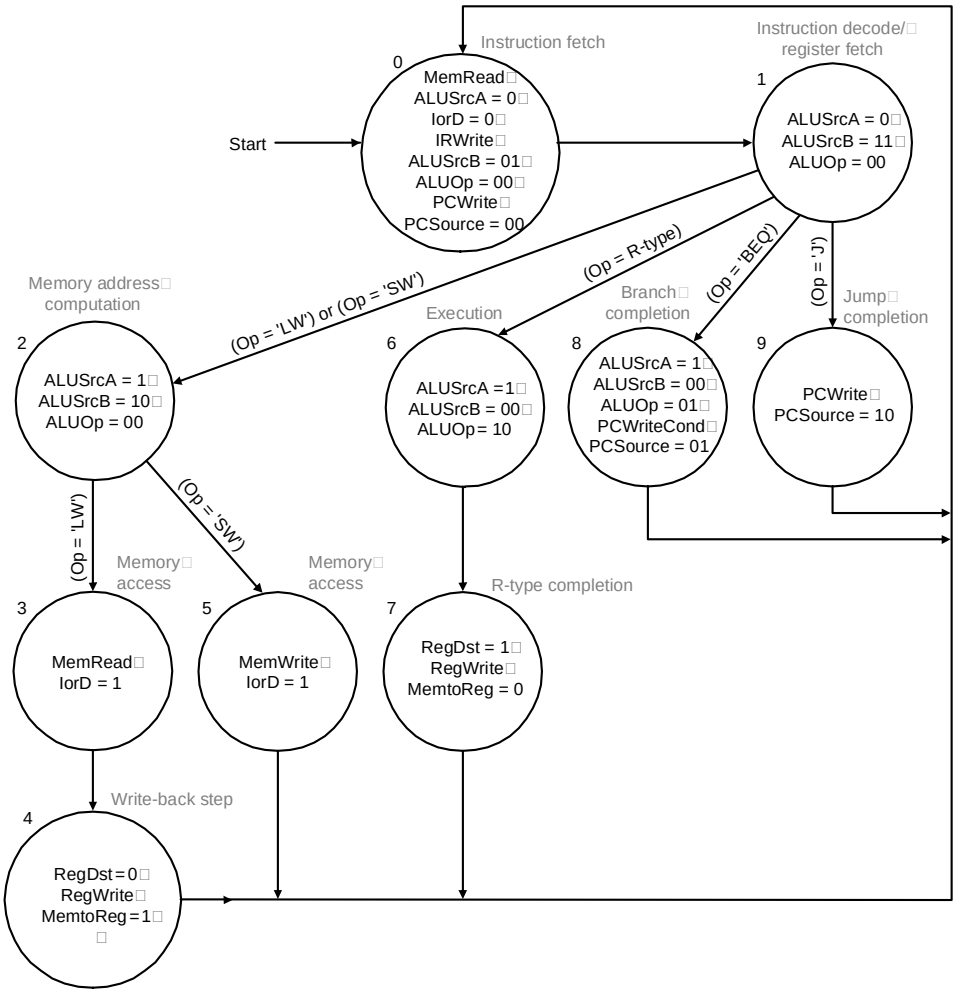
EXCHANGE r1, r2, offset

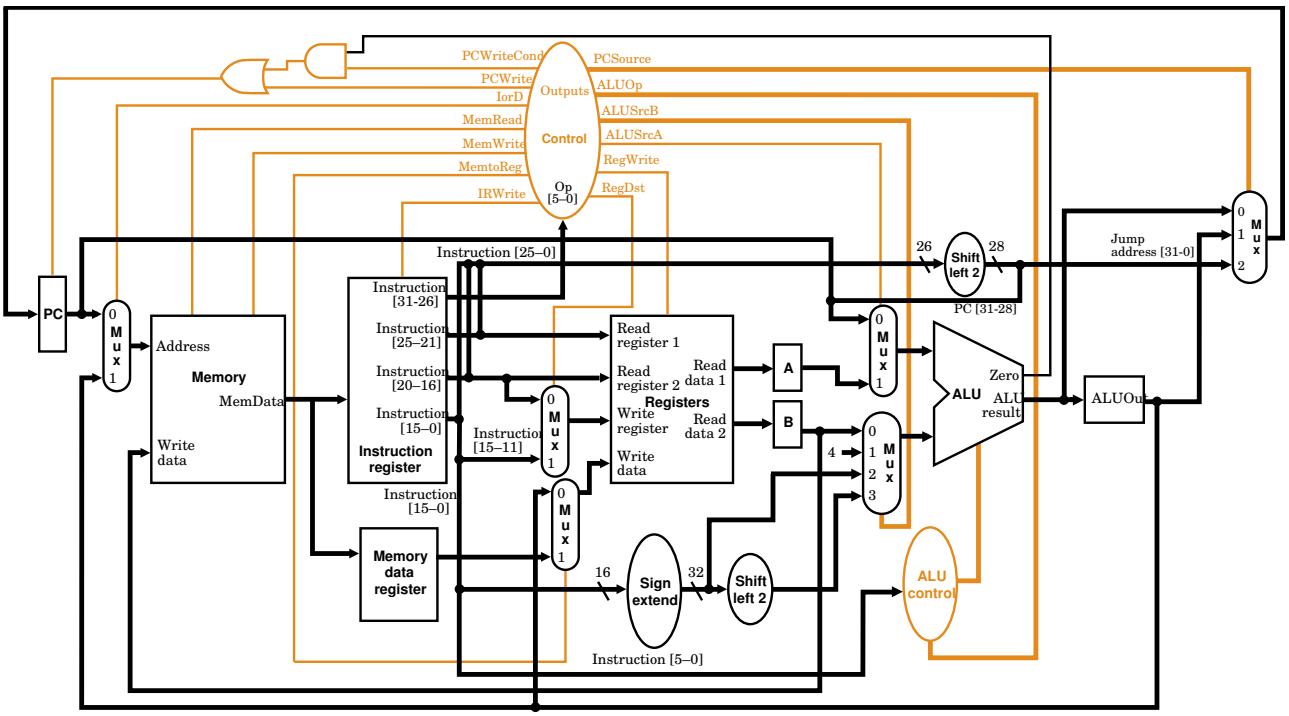
che scambia la parola di memoria di indirizzo r1+offset con la parola di memoria di indirizzo r2+offset, ovvero: M[r1+offset] ↔ M[r2+offset]

Ricordando i tre formati di codifica delle istruzioni (riportati di seguito) si chiede di:
- riportare il formato della nuova istruzione macchina (specificando anche i campi destinati rispettivamente a *r1* e *r2*);
- riportare, nella corrispondente figura, le modifiche necessarie al datapath;
- estendere il diagramma degli stati per implementare la nuova istruzione. [6]

Promemoria formati delle istruzioni:




Start
Instruction fetch
0
MemRead
ALUSrcA = 0
IorD = 0
IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00
Instruction decode/
register fetch
1
ALUSrcA = 0
ALUSrcB = 11
ALUOp = 00
(Op = 'LW') or (Op = 'SW')
(Op = R-type)
(Op = 'BEQ')
(Op = 'J')
Memory address
computation
2
ALUSrcA = 1
ALUSrcB = 10
ALUOp = 00
Execution
6
ALUSrcA =1
ALUSrcB = 00
ALUOp= 10
Branch
completion
8
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 01
PCWriteCond
PCSource = 01
Jump
completion
9
PCWrite
PCSource = 10
(Op = 'LW')
(Op = 'SW')
Memory
access
3
MemRead
IorD = 1
Memory
access
5
MemWrite
IorD = 1
R-type completion
7
RegDst = 1
RegWrite
MemtoReg = 0
Write-back step
4
RegDst = 0
RegWrite
MemtoReg = 1


PCWriteCond
PCWrite
IorD
MemRead
MemWrite
MemtoReg
IRWrite
Outputs
Control
Op
[5–0]
PCSource
ALUOp
ALUSrcB
ALUSrcA
RegWrite
RegDst
PC
Address
Memory
MemData
Write
data
Instruction
[31-26]
Instruction
[25–21]
Instruction
[20–16]
Instruction
[15–0]
Instruction
register
Instruction
[15–11]
Instruction
[15–0]
Memory
data
register
Read
register 1
Read
register 2
Registers
Write
register
Write
data
Read
data 1
Read
data 2
A
B
4
ALU
Zero
ALU
result
ALUOut
Instruction [25–0]
26
Shift
left 2
28
PC [31-28]
Jump
address [31-0]
16
Sign
extend
32
Shift
left 2
ALU
control
Instruction [5–0]

Soluzione

Si usa il formato I (c'è la necessità di memorizzare due registri e un offset) in cui r1 e r2 corrispondono ai registri rs e rt.

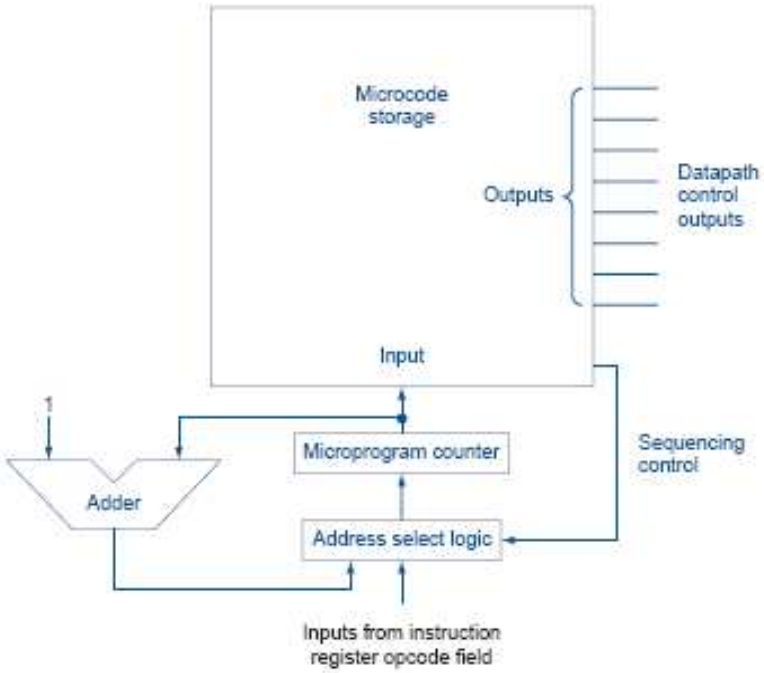
L'istruzione *exchange* è simile, nelle sue operazioni iniziali, alla *lw* ma a differenza di questa non comporta alcuna scrittura nei registri, bensì due letture e due scritture in memoria. I passi necessari per l'esecuzione sono i seguenti (a seconda della specifica soluzione implementativa adottata, i passi possono essere effettuati in diversi ordini, un passo può essere ripetuto e più passi possono essere effettuati in parallelo):

- Calcolo dell'indirizzo rs+offset
- Lettura del valore da M[rs+offset] ad un registro temporaneo
- Calcolo dell'indirizzo rt+offset
- Lettura del valore da M[rt+offset] ad un registro temporaneo
- Scrittura in M[rt+offset] del valore del registro temporaneo contenente M[rs+offset]
- Scrittura in M[rs+offset] del valore del registro temporaneo contenente M[rt+offset]

Una soluzione è quella di condividere gli stati della *lw* fino al calcolo di rs+offset, leggere il valore di M[rs+offset] mettendolo in un registro temporaneo MDR2 (oltre che MDR) controllato con un segnale di scrittura in modo da non perderne il valore, quindi calcolare l'indirizzo rt+ offset, leggere in MDR M[rt+offset] e scriverlo direttamente in M[rs+offset] per non perdere il valore, quindi completare l'istruzione scrivendo MDR2 in M[rt+offset].

Vedere la soluzione dettagliata all'esercizio 5 (fare attenzione agli stati in cui vengono calcolati gli indirizzi per accedere alla memoria!).

6. Si consideri il seguente schema, che si riferisce ad una unità di controllo del processore microprogrammata. Per ciascuna delle seguenti affermazioni, si dica se è vera o falsa, motivando le risposte:
   - il contatore di microprogramma (microprogram counter) viene aggiornato quando viene aggiornato il program counter;
   - il contatore di microprogramma viene aggiornato più volte rispetto al program counter;
   - il contatore di microprogramma viene aggiornato ad ogni ciclo di clock. [3]



7. Si consideri una memoria cache a corrispondenza diretta. Motivando brevemente le risposte, si dica come influiscono sul tempo di hit (T_{hit}), tempo di miss (T_{miss}) e frequenza di miss (f_{miss}) della stessa memoria cache:
- un eventuale aumento del grado di associatività della stessa memoria cache;
- l'introduzione di un'ulteriore cache di livello 2. [4]

8. Si consideri la nota implementazione dell'unità di controllo secondo la tecnica multiciclo relativamente alle istruzioni MIPS *lw*, *sw*, *beq*, *j* e *TIPO-R*. Si assuma il seguente carico di lavoro:

Tipo-R:	40%
sw:	20%
lw:	30%

Si dispone di una cache primaria con T_{hit} = 1 ciclo di clock, f_{miss} per le istruzioni pari al 2%, f_{miss} per i dati pari al 4% e di una memoria DRAM con un tempo di accesso pari a 100 cicli di clock.

a) Determinare quale sarebbe l'incremento delle prestazioni aggiungendo una cache secondaria con tempo di accesso di 10 cicli di clock e f_{miss}=20% per le istruzioni e 40% per i dati.

b) Determinare quale sarebbe, rispetto al caso a), la variazione delle prestazioni aggiungendo una cache di livello 3 con tempo di accesso di 20 cicli di clock e f_{miss}=50% per i dati e f_{miss}=50% per le istruzioni. [5]

Soluzione

Innanzitutto si calcola CPI_{ideale} = 0.3*5+ (0.4+0.2)*4 + (1-0.3-0.4-0.2)*3 = 4.2

Seguendo gli esercizi 3-4 sulla memoria cache esposti a lezione si risolve il punto a. In particolare:

Con cache L1 risulta:
$CPI = CPI_{ideale}$ + 0.02*100+0.5*0.04*100 = 4.2+4=8.2

Con cache L1+L2 risulta:
$CPI = CPI_{ideale}$ + 0.02*[0.8*10+0.2*110]+0.5*0.04*[0.6*10+0.4*110] = 4.2 + 1.6 = 5.8
(NB: se si suppone che a seguito di un miss sia in L1 sia in L2 la memoria DRAM fornisca il dato direttamente al processore senza passare per le cache, allora è accettabile usare il valore 100 al posto di 110 – si ottiene 5.68 al posto di 5.8)

Per il punto a) si ottiene quindi un incremento di prestazioni pari a 8.2/5.8= 1.41

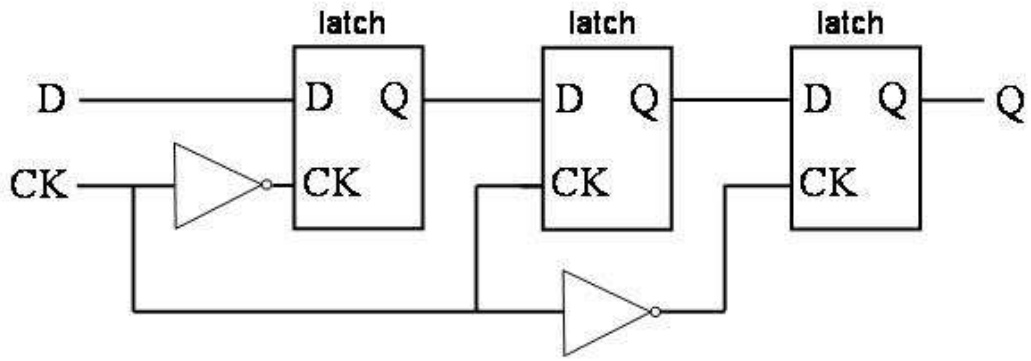
Punto b:
Occorre calcolare CPI nel caso di tre cache. Risulta:
$CPI = CPI_{ideale}$ + 0.02*[0.8*10+0.2*(0.5*30+0.5*130)] + 0.5*0.04*[0.6*10+0.4*(0.5*30+0.5*130)]
= 4.2 + 1.24 = 5.44
In pratica si considerano le seguenti penalità per le istruzioni e i dati:

miss L1 (prob. 2%) hit L2 (prob. 80%):	10 cicli
miss L1 (prob. 2%) miss L2 (prob. 20%) hit L3 (prob. 50%):	30 cicli
miss L1 (prob. 2%) miss L2 (prob. 20%) miss L3 (prob. 50%):	130 cicli

(con una ipotesi alternativa analoga a quella considerata al punto a, si potevano usare i valori 10, 20 e 100 cicli rispettivamente, ottenendo 5.2)

Si ottiene quindi un incremento di prestazioni pari a 5.8/5.44=1.066

9. Si descriva il comportamento del seguente circuito (ingressi: D e CK; uscita: Q). [2]



Soluzione

I primi due latch costituiscono un Flip-flop master slave di tipo D, attivo sul fronte di salita.
Il circuito si può quindi vedere come serie di un FF di tipo D e di un latch di tipo D. Il comportamento globale è quello di campionare D sul fronte di salita, rendendolo però disponibile in uscita Q solo sul successivo fronte di discesa (ovvero, ritardando l'uscita di mezzo ciclo rispetto al fronte di campionamento).