

**SOLUZIONI DELLA SECONDA PROVA INTERMEDIA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO ORDINAMENTO DIDATTICO**

14 Giugno 2007

MOTIVARE IN MANIERA CHIARA LE SOLUZIONI PROPOSTE A CIASCUNO DEGLI ESERCIZI SVOLTI

ESERCIZIO 1 (8 punti)

Si consideri un calcolatore in cui la CPU esegue 2×10^6 istruzioni/s e l'esecuzione di una istruzione richiede 5 cicli di clock, 2 dei quali tengono occupato il bus di sistema. Si ipotizzi che l'80% dell'Instruction Rate sia usato dalla CPU per eseguire programmi che non contengono istruzioni di I/O.

1. Calcolare la massima frequenza di trasferimento dati ottenibile (espressa in parole/s) fra una periferica collegata al bus di sistema e la memoria principale nei seguenti due casi:
 - a. (3 punti) modalità "block transfer DMA" in cui una operazione di lettura/scrittura della memoria richiede 1 ciclo di clock (posso trasferire 1 parola ogni ciclo di clock);
 - b. (3 punti) modalità "transparent DMA" in cui una operazione di lettura/scrittura della memoria richiede 1 ciclo di clock (posso trasferire 1 parola ogni ciclo di clock).
2. (2 punti) Si illustrino brevemente le modalità di indirizzamento delle periferiche di I/O in un calcolatore.

Soluzione

1.
 - a. Nella modalità block transfer DMA, il controller DMA prende possesso del bus di sistema ed esegue il trasferimento dati interrompendo la CPU. Pertanto la massima frequenza di trasferimento dati ottenibile è data da:
$$2 \times 10^6 \text{ istruzioni/s} \times 5 \text{ cicli/istruzione} \times 1 \text{ parola/ciclo} = 10^7 \text{ parole/s}$$
 - b. Nella modalità transparent DMA, il controller DMA può effettuare i trasferimenti dati solo quando il bus di sistema non è occupato, senza interrompere la CPU. Ovviamente può sfruttare tutti i cicli a istruzione quando la CPU esegue istruzioni di I/O, pertanto la massima frequenza dati ottenibile è data da:
$$2 \times 10^6 \text{ istruzioni/s} \times (0.8 \times 3 \text{ cicli/istruzione} + 0.2 \times 5 \text{ cicli/istruzione}) \times 1 \text{ parola/ciclo} = 6.8 \times 10^6 \text{ parole/s}$$
2. Si veda il cap. 5 (memory mapped e isolated I/O).

ESERCIZIO 2 (8 punti)

Sia dato il seguente formato: rappresentazione in virgola mobile a 24 bit con mantissa frazionaria normalizzata in segno e valore (1.M) e esponente a 7 bit in eccesso 63.

1. (2 punti) Calcolare il minimo e il massimo valore rappresentabile;
2. (4 punti) Rappresentare i valori $(20.25)_{10}$ e $(322.625)_{10}$;
3. (2 punti) Sommare i due numeri al punto precedente con l'algoritmo dei calcolatori.

Soluzione

1. Min: 2^{-63}
Max: $2^{64}(2-2^{-16})$
2. $(20.25)_{10} = (10100.01)_2 = 1.010001 \times 2^4$
L'esponente è in eccesso 63, per cui lo rappresenteremo come $63+4=67 = (1000011)_2$.
In definitiva, la rappresentazione è 0 100011 0100010000000000 (considerando il bit di segno, i 7 bit di esponente e i restanti 16 bit di mantissa).
 $(322.625)_{10} = (101000010.101)_2 = 1.01000010101 \times 2^8$
L'esponente è in eccesso 63, per cui lo rappresenteremo come $63+8=71 = (1000111)_2$.
In definitiva, la rappresentazione è 0 100011 0100001010100000 (considerando il bit di segno, i 7 bit di esponente e i restanti 16 bit di mantissa).
3. Per sommare i 2 numeri ottenuti al punto precedente, è necessario allineare gli esponenti. Nello specifico, dovremmo shiftare di 4 posizioni verso destra il numero più piccolo, in modo da ottenere entrambi gli esponenti uguali a 8. Per quanto detto, otteniamo una diversa

rappresentazione del numero 20.25:

0 1000111 0001010001000000 (si noti in grassetto il bit implicito).

Ora che gli esponenti sono allineati, possiamo sommare le mantisse:

0.0001010001000000 +

1.0100001010100000 =

1.0101011011100000

La rappresentazione della somma sarà:

0 1000111 0101011011100000, corrispondente a $(101010110.111)_2 = (342.875)_{10}$.

ESERCIZIO 3 (8 punti)

Si scriva una funzione Assembler MIPS, chiamata `max_vettori`, che, ricevendo in ingresso l'indirizzo iniziale di tre vettori `u`, `v`, `z` (in `$4`, `$5`, `$6` rispettivamente) di dimensione `N` (in `$7`), metta in `z[i]` il massimo tra `u[i]` e `v[i]`. Si faccia uso di una funzione esistente `max` che, ricevendo due interi in `$4` e `$5`, mette in `$6` il massimo tra i due. Si abbia cura di non avere alterato i contenuti di `$4`–`$6` all'uscita dalla funzione `max_vettori`.

Soluzione.

`$8 ← i; $9-$11 ← copia di $4-$6` – vengono anche copiati nello stack

```
max_vettori:  addi $29, $29, -32
               sw $8, 0($29)
               sw $9, 4($29)
               sw $10, 8($29)
               sw $11, 12($29)
               sw $4, 16($29)
               sw $5, 20($29)
               sw $6, 24($29)
               sw $31, 28($29)
               move $9, $4
               move $10, $5
               move $11, $6
               move $8, $0
for:          beq $8, $7, exit
               lw $4, 0($9)
               lw $5, 0($10)
               jal max
               sw $6, 0($11)
               addi $9, $9, 4
               addi $10, $10, 4
               addi $11, $11, 4
               addi $8, $8, 1
               j for
exit:         lw $8, 0($29)
               lw $9, 4($29)
               lw $10, 8($29)
               lw $11, 12($29)
               lw $4, 16($29)
               lw $5, 20($29)
               lw $6, 24($29)
               lw $31, 28($29)
               addi $29, $29, 32
               jr $31
```

ESERCIZIO 4 (9 punti)

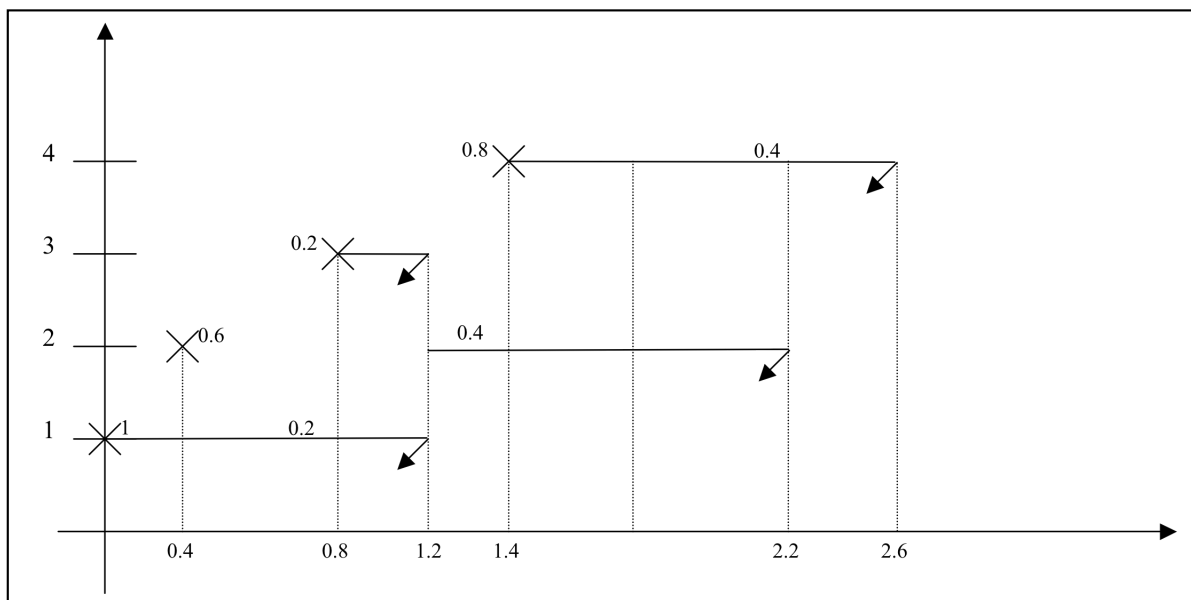
Si consideri un calcolatore in cui la memoria è partizionata in modo statico con quattro partizioni della seguente grandezza: 50K, 120K, 20K, 60K. La politica di scheduling dei processi è la FIFO multiprogrammata e l'allocazione dei processi avviene secondo la strategia Best-Fit.

Data la seguente lista di processi (si supponga che l'istante iniziale sia 0):

Job	Tempo di arrivo	Tempo di CPU	Memoria
1	0.0	1	70K
2	0.4	0.6	100K
3	0.8	0.2	45K
4	1.4	0.8	10K

- (6 punti) Mostrare la sequenza di esecuzione dei job usando il metodo grafico (tempo, job) e indicando chiaramente lo stato della memoria durante le varie fasi dell'esecuzione;
- (3 punti) Calcolare il tempo di turnaround medio e turnaround pesato medio.

Soluzione



Stato della memoria:

Istante:	0.0	0.8	1.2	1.4
50KB		(3)		
120KB	(1)	(1)	(2)	(2)
20KB				(4)
60KB				

Job	Tarrivo	Tstart	Tfinish	Turnaround	Wturn.
1	0.0	0.0	1.2	1.2	1
2	0.4	1.2	2.2	1.8	3
3	0.8	0.8	1.2	0.4	2
4	1.4	1.4	2.6	1.2	1.5
Media				1.15	1.92