

**SECONDA PROVA INTERMEDIA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO ORDINAMENTO DIDATTICO
9 Giugno 2006**

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

ESERCIZIO 1 (10 punti)

L'ampiezza della linea dati del bus di un calcolatore è pari a 32 bit. La frequenza del clock della CPU è di 500 MHz.

- 1) (4 punti) Sapendo che un bus sincrono presenta la stessa frequenza di clock della CPU e la durata di una trasmissione sul bus impiega 2 cicli di clock e che il tempo di ciclo della memoria richiede 16 cicli di clock, illustrare chiaramente il protocollo di lettura su bus sincrono utilizzando l'opportuno grafico, indicando il tempo complessivo di trasferimento di una parola da 64 bit.
- 2) (3 punti) Le istruzioni di CPU necessarie per trasferire una parola da 64 bit su bus dati da 32 bit da una periferica alla memoria nel caso in cui i trasferimenti periferica-memoria vengano gestiti mediante IO da programma richiedono complessivamente 40 cicli di clock. Calcolare la massima velocità di trasferimento (in Byte/s) fra periferica e memoria che è possibile raggiungere effettuando i trasferimenti mediante IO da programma.
- 3) (3 punti) Indicare il fattore di incremento della velocità rispetto al caso precedente nell'ipotesi di trasferimento in DMA "block transfer".

ESERCIZIO 2 (8 punti)

Scrivere una funzione Assembler MIPS che implementi la seguente definizione **ricorsiva** di fattoriale:

`fattoriale(0) = 1.`
`fattoriale(n) = n * fattoriale(n-1).`

n è un intero non negativo, che si suppone memorizzato nel registro \$4. Il risultato va memorizzato nel registro \$5.

Ad esempio il codice MIPS potrebbe implementare la seguente funzione C:

```
int fattoriale(int n)
{
    return (n==0) ? 1 : n*fattoriale(n-1);
}
```

N.B. Eventuali soluzioni **non** ricorsive, anche se corrette, saranno valutate con punteggio nullo.

ESERCIZIO 3 (7 punti)

a) (4 punti) Si rappresentino i valori $(63.5)_{10}$ $(31.25)_{10}$ nella forma $1.M * 2^E$ e li si sommi con l'algoritmo dei calcolatori.

b) (3 punti) Si consideri ora un campo di 12 bit. La mantissa è frazionaria e rappresentata in segno e valore con normalizzazione 1.M. Con lo scopo di rappresentare i due valori al punto precedente e la loro somma, si calcoli il numero minimo di bit necessari per l'esponente, con il vincolo di rappresentazione in eccesso. A tal scopo si scelga l'eccesso massimo che permetta di rappresentare i tre valori al punto precedente.

ESERCIZIO 4 (8 punti)

La gestione della memoria di un elaboratore è operata dal sistema operativo in modo paginato. La memoria primaria è costituita da un massimo di 10 pagine. Siano dati i seguenti processi:

Processo	Tempo di arrivo	Tempo di CPU	Pagine
1	0.0	0.8	2
2	0.3	1.0	3
3	0.5	0.7	2
4	0.8	1.2	4
5	1.5	2.0	5

Mostrare, utilizzando il metodo grafico, la sequenza di esecuzione dei processi qualora si impieghi la politica di scheduling FIFO multiprogrammata. Il grafico deve essere accompagnato da una descrizione sintetica ma precisa dello stato della memoria ad ogni istante.

ESERCIZIO 1 (10 punti)

L'ampiezza della linea dati del bus di un calcolatore è pari a 32 bit. La frequenza del clock della CPU è di 500 MHz.

- 1) (4 punti) Sapendo che un bus sincrono presenta la stessa frequenza di clock della CPU e la durata di una trasmissione sul bus impiega 2 cicli di clock e che il tempo di ciclo della memoria richiede 16 cicli di clock, illustrare chiaramente il protocollo di lettura su bus sincrono utilizzando l'opportuno grafico, indicando il tempo complessivo di trasferimento di una parola da 64 bit.
- 2) (3 punti) Le istruzioni di CPU necessarie per trasferire una parola da 64 bit su bus dati da 32 bit da una periferica alla memoria nel caso in cui i trasferimenti periferica-memoria vengano gestiti mediante IO da programma richiedono complessivamente 40 cicli di clock. Calcolare la massima velocità di trasferimento (in Byte/s) fra periferica e memoria che è possibile raggiungere effettuando i trasferimenti mediante IO da programma.
- 3) (3 punti) Indicare il fattore di incremento della velocità rispetto al caso precedente nell'ipotesi di trasferimento in DMA "block transfer".

Soluzione:

- 1) La durata di un ciclo di clock è pari a $1/(500 \text{ MHz}) = 2 \text{ ns}$

La lettura su un bus sincrono avviene secondo il protocollo seguente:

- Segnale di READ sulla linea di controllo e contemporaneamente l'indirizzo della locazione in cui risiede il dato sulla linea indirizzi:

2 cicli di clock = 4 ns

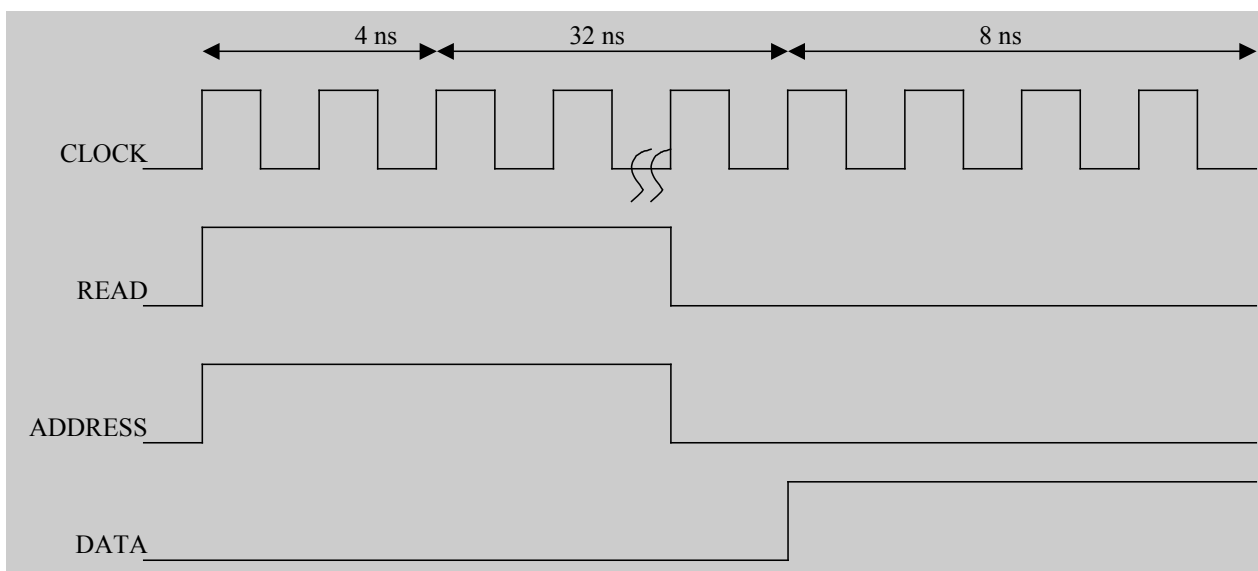
- Lettura della parola dalla memoria: **32 ns**

- Trasferimento della parola dalla memoria:

se la parola da leggere ha ampiezza pari a 64 bit

trasferimento della parola = 8 ns

Tempo totale per leggere una parola dalla memoria = (4 + 32 + 8) ns = 44 ns



- 2) Per trasferire un blocco di 64 bit occorrono 40 cicli. La massima velocità di trasferimento è dunque pari a:

$$64 \frac{\text{bit}}{\text{parola}} \cdot 500 \frac{\text{Mcicli}}{\text{sec}} \cdot \frac{1}{40 \frac{\text{cicli}}{\text{parola}}} = 800 \frac{\text{Mbit}}{\text{sec}} = 10^8 \frac{\text{Byte}}{\text{sec}}$$

- 3) In DMA "block transfer" è possibile trasferire un blocco pari alla dimensione del bus dati per ogni ciclo di clock. Poiché la parola è di 64 bit mentre il bus risulta di 32 bit, occorrono due trasferimenti per un totale di 2 cicli. Pertanto la massima velocità di trasferimento è pari a:

$$64 \frac{\text{bit}}{\text{parola}} \cdot 500 \frac{\text{Mcicli}}{\text{sec}} \cdot \frac{1}{2 \frac{\text{cicli}}{\text{parola}}} = 16 \frac{\text{Gbit}}{\text{sec}} = 2 \cdot 10^9 \frac{\text{Byte}}{\text{sec}} .$$

Usando DMA "block transfer" trasferiamo blocchi da 64 bit con velocità venti volte superiore a quella in modalità I/O programmato.

ESERCIZIO 2 (8 punti)

Scrivere una funzione Assembler MIPS che implementi la seguente definizione **ricorsiva** di fattoriale:

$$\begin{aligned} \text{fattoriale}(0) &= 1. \\ \text{fattoriale}(n) &= n * \text{fattoriale}(n-1). \end{aligned}$$

n è un intero non negativo, che si suppone memorizzato nel registro \$4. Il risultato va memorizzato nel registro \$5.

Ad esempio il codice MIPS potrebbe implementare la seguente funzione C:

```
int fattoriale(int n)
{
    return (n==0) ? 1 : n*fattoriale(n-1);
}
```

N.B. Eventuali soluzioni **non** ricorsive, anche se corrette, saranno valutate con punteggio nullo.

Soluzione.

fattoriale:	beq \$4, \$0, basestep	#if(n==0) vai a passo base
	addi \$29, \$29, -4	#spazio nello stack
	sw \$31, 0(\$29)	#salvataggio del PC
	subi \$4, \$4, 1	# $n \leftarrow n - 1$
	jal fattoriale	#calcola fattoriale(n-1)
	addi \$4, \$4, 1	# $n \leftarrow n + 1$
	mul \$5, \$4, \$5	# $\text{fattoriale}(n) \leftarrow n * \text{fattoriale}(n-1)$
	lw \$31, 0(\$29)	#ripristina il PC
	addi \$29, \$29, 4	#dealloca lo spazio nello stack
	jr \$31	#ritorna al chiamante
basestep:	addi \$5, \$0, 1	#passo base dell'algoritmo
	jr \$31	#ritorna al chiamante

ESERCIZIO 3 (7 punti)

a) (4 punti) Si rappresentino i valori $(63.5)_{10}$ e $(31.25)_{10}$ nella forma $1.M * 2^E$ e li si sommi con l'algoritmo dei calcolatori.

b) (3 punti) Si consideri ora un campo di 12 bit. La mantissa è frazionaria e rappresentata in segno e valore con normalizzazione 1.M. Con lo scopo di rappresentare i due valori al punto precedente e la loro somma, si calcoli il numero minimo di bit necessari per l'esponente, con il vincolo di rappresentazione in eccesso. A tal scopo si scelga l'eccesso massimo che permetta di rappresentare i tre valori al punto precedente.

Soluzione

a) Innanzi tutto rappresentiamo in virgola mobile i valori dati:

$$\begin{aligned}(63.5)_{10} &= 111111.1 = 1.111111 * 2^5 \\ (31.25)_{10} &= 11111.01 = 1.111101 * 2^4\end{aligned}$$

Poiché il primo ha esponente maggiore del secondo ($5 > 4$) di quest'ultimo si fa scorrere la mantissa a destra di una posizione, ovvero si moltiplica e divide per 2^1 .

Sommiamo ora le mantisse:

$$\begin{array}{r} 1.1111110 + \\ 0.1111101 = \\ \hline 10.1111011 \end{array}$$

Normalizzando la mantissa ottenuta, la somma si rappresenta:

$$1.01111011 * 2^6 \rightarrow (93.75)_{10}$$

b) Per rappresentare i tre valori esattamente, è sufficiente far sì che il massimo esponente rappresentabile sia 6, in quanto è l'esponente maggiore fra quelli dei tre valori. Il numero minimo di bit per rappresentare il valore 6 è tre, in quanto è il minimo K che soddisfa la relazione $2^K \geq 6$. Con tre bit possiamo rappresentare otto possibili valori, da 0 a 7 in eccesso 0. L'eccesso massimo che ci permette ancora di rappresentare il valore 6 è 1. Il numero *minimo* di bit per l'esponente è dunque 3 con l'esponente rappresentato in eccesso 1.

Dal punto di vista della mantissa, questa scelta ci permette di rappresentare esattamente i valori dati: si nota infatti che la lunghezza massima della mantissa richiesta è di otto bit, disponibilità rispettata dal formato scelto. La tabella seguente mostra i tre numeri rappresentati secondo il formato scelto:

S (1)	Esponente (3)			Mantissa (8 bit)							
0	1	1	0	1	1	1	1	1	1	0	0
0	1	0	1	1	1	1	1	0	0	1	0
0	1	1	1	0	1	1	1	1	0	1	1

ESERCIZIO 4 (8 punti)

La gestione della memoria di un elaboratore è operata dal sistema operativo in modo paginato. La memoria primaria è costituita da un massimo di 10 pagine. Siano dati i seguenti processi:

Processo	Tempo di arrivo	Tempo di CPU	Pagine
1	0.0	0.8	2
2	0.3	1.0	3
3	0.5	0.7	2
4	0.8	1.2	4
5	1.5	2.0	5

Mostrare, utilizzando il metodo grafico, la sequenza di esecuzione dei processi qualora si impieghi la politica di scheduling FIFO multiprogrammata. Il grafico deve essere accompagnato da una descrizione sintetica ma precisa dello stato della memoria ad ogni istante.

Soluzione.

Elenco degli eventi:

Istante 0.0. Avvio di P1. Due pagine impegnate.

Istante 0.3. Avvio di P2. Tre pagine impegnate.

Istante 0.5. Avvio di P3. Due pagine impegnate.

Istante 0.8. P4 non può essere posto in esecuzione perché non ci sono abbastanza risorse. Viene dunque posto in attesa.

Istante 1.7. P1 termina. P4 può essere allocato (si noti in locazioni non contigue).

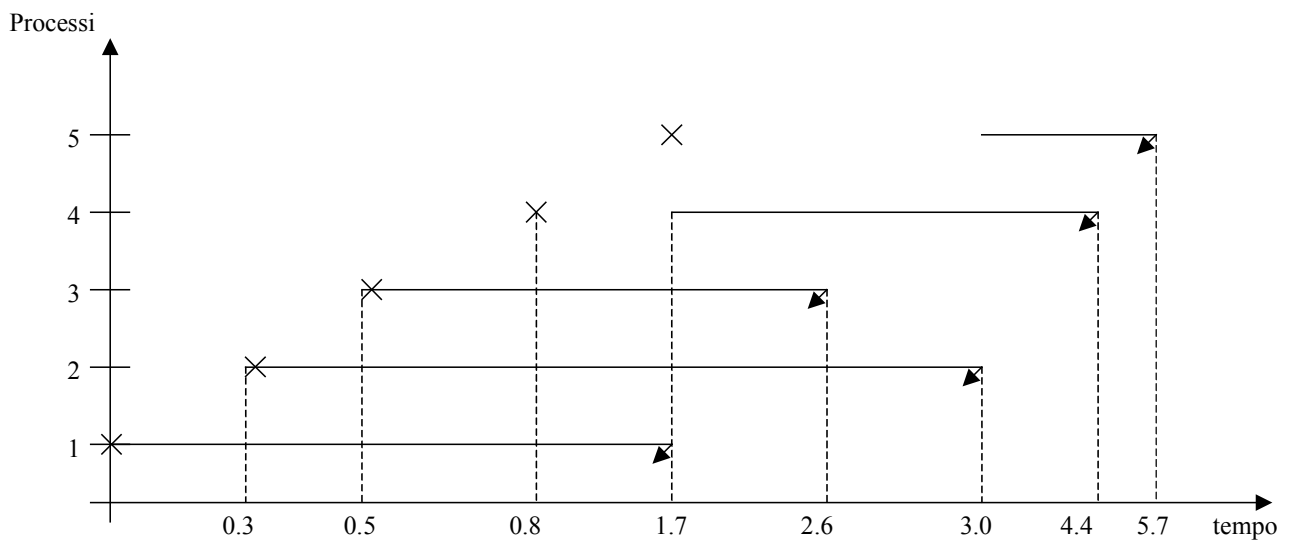
Istante 1.7. P5 non può essere posto in esecuzione perché non ci sono abbastanza risorse. Viene dunque posto in attesa.

Istante 2.6. P3 termina. Si liberano due pagine ma non ci sono ancora risorse disponibili per P5.

Istante 3.0. P2 termina. P5 può essere allocato.

Istante 4.4. P4 termina.

Istante 5.7. P5 termina.



P1
P1

P1
P1
P2
P2
P2

P1
P1
P2
P2
P2
P3
P3

P4
P4
P2
P2
P2
P2
P3
P3
P3
P4
P4

P4
P4
P2
P2
P2
P4
P4

P4
P4
P5
P5
P5
P5
P5
P4
P4