

SECONDA PROVA INTERMEDIA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO ORDINAMENTO DIDATTICO
14 Gennaio 2010

NOME:

COGNOME:

MATRICOLA:

ESERCIZIO 1 (11 punti)

Si consideri un calcolatore in cui la CPU lavora alla frequenza di 1 Ghz. L'esecuzione di una istruzione richiede 5 cicli di clock, 3 dei quali tengono occupato il bus di sistema. Il 90% delle istruzioni è usato dalla CPU per eseguire programmi che non contengono trasferimenti di I/O. L'ampiezza della linea dati del bus è pari a 64 bit. Si ipotizzi che il bus sia sincrono, la durata di una trasmissione sul bus corrisponda a 1 ciclo di clock e che il tempo di ciclo della memoria sia 3 ns.

1. (2 punti) Calcolare il numero di cicli di clock necessari per trasferire una parola da 64 bit, mostrando chiaramente il protocollo di lettura dati sul bus.
2. Utilizzando il risultato ottenuto al punto precedente, calcolare la massima velocità di trasferimento (in parole/s):
 - a. (2 punti) in modalità "Block Transfer DMA";
 - b. (2 punti) in modalità "Transparent DMA";
 - c. (2 punti) in modalità DMA con furto di ciclo, in cui vengono rubati 2 cicli a istruzione.
3. (3 punti) Si illustrino brevemente le diverse tecniche di arbitraggio del bus (centralizzato e distribuito), riportandone vantaggi e svantaggi.

ESERCIZIO 2 (8 punti)

Considerato un campo di 64 bit, siano dati i seguenti formati:

- a. rappresentazione di interi senza segno;
 - b. rappresentazione in virgola fissa con 20 bit di parte frazionaria;
 - c. rappresentazione in virgola mobile con mantissa frazionaria e normalizzata in segno e valore (1.M) ed esponente a 8 bit in eccesso 128.
1. (3 punti) Calcolare il minimo e il massimo valore (entrambi positivi e non nulli) nei tre casi.
 2. (3 punti) Sommare, secondo l'algoritmo dei calcolatori, i numeri $(12.5)_{10}$ e $(5.25)_{10}$, esprimendoli in virgola mobile secondo la rappresentazione c.
 3. (2 punti) Qual è il valore da attribuire all'eccesso (sempre secondo la rappresentazione c) per fare in modo che l'esponente massimo rappresentabile sia pari a 64?

ESERCIZIO 3 (6 punti)

Si progetti un addizionatore parallelo a 2 bit utilizzando un half adder ed un full adder. Gli ingressi dell'addizionatore parallelo sono due operandi A e B rappresentati su 2 bit. In uscita l'addizionatore dovrà fornire la somma (A+B, sempre su 2 bit) e il riporto (per segnalare le situazioni di overflow, es.: $11 + 11 = 10$ con riporto 1). Si mostrino chiaramente i collegamenti tra i due moduli (half e full adder) e si descrivano le problematiche relative alla propagazione del riporto, indicando schemi progettuali più efficienti.

ESERCIZIO 4 (8 punti)

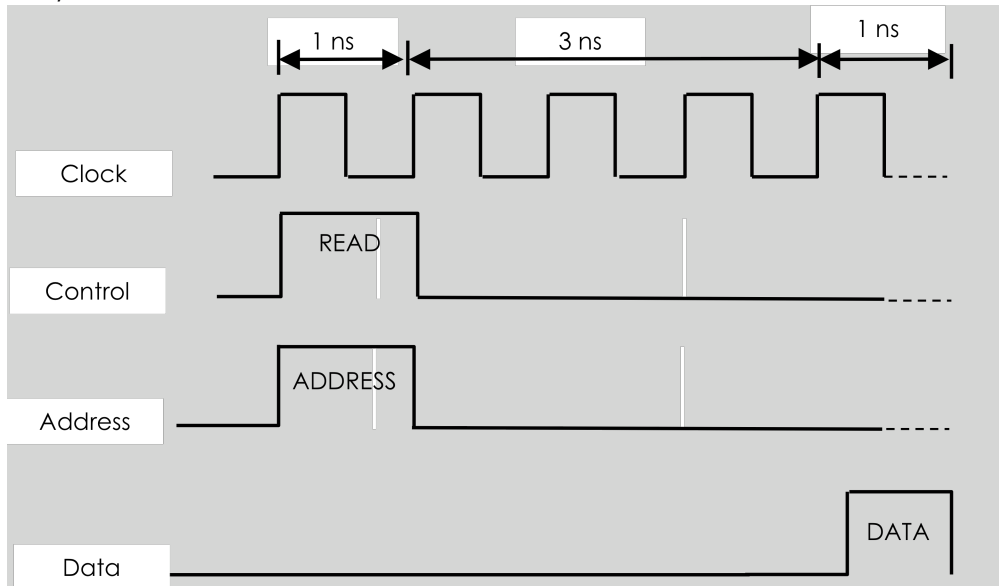
Scrivere il codice Assembly MIPS di una funzione che, dati l'indirizzo iniziale di un vettore v e la sua dimensione N (in \$4 e \$5, rispettivamente), scambi i valori $v[i]$ con $v[N-1-i]$, per $i=0, \dots, (N/2-1)$. Si utilizzi inoltre una funzione esistente "div", che esegue la divisione tra i valori contenuti in \$4 e \$5 e scrive in \$6 il risultato. Si implementi inoltre il salvataggio/ripristino del contesto (modalità "callee save"). E' dato il codice C della funzione corrispondente:

```
void inverti(int *v, int N) {
    int Nmezzi=div(N,2);
    for (int i=0 ; i<Nmezzi; i++) {
        int temp=v[i];
        v[i]=v[N-1-i];
        v[N-1-i]=temp;
    }
}
```

ESERCIZIO 1

Soluzione

1. $IR = 1\text{GHz} / (5 \text{ cicli / istr}) = 2 \times 10^8 \text{ istruzioni/s}$. Dato che una trasmissione dura un solo ciclo di clock: $T = 1/f = 1 \text{ ns}$



In totale sono necessari 5 cicli di clock per trasferire una parola (1/5 parole/ciclo).

2.

- a. In modalità "DMA block transfer" il modulo DMA diventa il master del bus quindi tutti i trasferimenti da parte della CPU vengono sospesi. Si ha:
max velocità di trasferimento =
 $2 \times 10^8 \text{ (istruzioni/s)} \times 5 \text{ (cicli/istruzione)} \times 1/5 \text{ (parole / ciclo)} = 2 \times 10^8 \text{ parole/s}$
- b. In modalità "DMA transparent" il modulo DMA può intervenire solo quando alla CPU non occorre il bus di sistema. Quindi: max velocità di trasferimento =
 $= (2 \text{ cicli/istr} \times 0.9 + 5 \text{ cicli/istr} \times 0.1) \times 1/5 \text{ (parole/ciclo)} \times 2 \times 10^8 \text{ (istruzioni/s)} =$
 $0.92 \times 10^8 \text{ parole/s}$
- c. In modalità "furto di ciclo", vengono rubati $2/7 \times 10^9$ cicli/s. Potendo trasferire 1/5 (parole / ciclo), la max velocità di trasferimento risulta:
 $2/7 \times 10^9 \text{ cicli/s} \times 1/5 \text{ (parole / ciclo)} = 0.57 \times 10^8 \text{ parole/s}$

3. Vedi lucidi del corso (cap. 7).

ESERCIZIO 2

Soluzione

1. a. Minimo: 1 Max: $2^{64}-1$.
b. Minimo: 2^{-20} Max: $2^{43}-2^{-20}$
c. Minimo: 2^{-128} Max: $2^{127}(2-2^{-55})$.

2. $(12.5)_{10} = 1100.1 = 1.1001 \cdot 2^3$
 $(5.25)_{10} = 101.01 = 1.0101 \cdot 2^2$

I due numeri si possono rappresentare nel seguente modo:

Segno	Esponente	Mantissa
0	1000011	10010000000000000000...0
0	1000010	01010000000000000000...0

Poiché il primo ha esponente maggiore del secondo ($3 > 2$) di quest'ultimo si fa scorrere la mantissa a destra di una posizione.

I due numeri da sommare sono:

$$\begin{array}{r} 1.10010 + \\ 0.10101 = \\ \hline 10.00111 \quad (*2^3) \end{array}$$

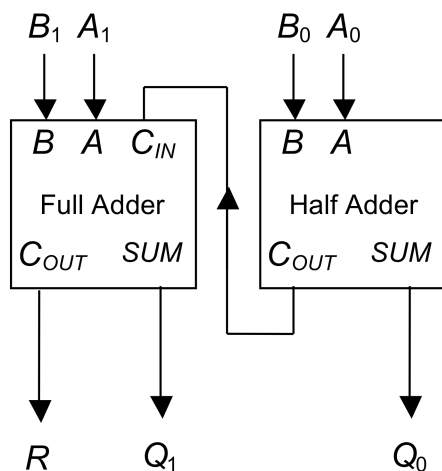
E' necessario normalizzare il risultato

Segno	Esponente	Mantissa
0	10000100	00011100000000000000...0

3. Con 8 bit si possono rappresentare 256 valori (con eccesso 0 corrisponderebbero a 0, ..., 255). Per fare in modo che il valore 64 sia il valore massimo dell'esponente, dobbiamo scegliere un valore per l'eccesso pari a $255-64 = 191$ (in modo che il range rappresentabile sia $[-191, 64]$).

ESERCIZIO 3

Soluzione



Lo schema realizzativo è indicato nella figura a fianco, in cui si vede come viene calcolata l'uscita Q_1Q_0 e il bit di riporto R , partendo dai singoli bit degli operandi A e B .

Il principale limite degli addizionatori paralleli progettati utilizzando questi componenti (half e full adder) è relativo alla propagazione del riporto. Infatti, perché l'uscita del parallel adder sia valida, occorre attendere che il riporto in uscita dall'half adder sia valido (e quindi occorre attendere che l'half adder sommi i bit meno significativi di A e B).

Si possono utilizzare schemi di progetto più efficienti come il carry look-ahead adder, in cui i riporti sono calcolati a priori utilizzando una rete logica opportuna, e quindi non è necessario attenderne la propagazione.

ESERCIZIO 4

Soluzione

$i \rightarrow \$8$; $\&v[0]+i*4 \rightarrow \$9$; $\&v[0]+(N-1-i)*4 \rightarrow \10
 $v[i] \rightarrow \$11$; $v[N-1-i] \rightarrow \$12$

copia di $\&v[0] \rightarrow \$13$; copia di $N \rightarrow \$14$

<pre>inverti: addi \$29, \$29, -32 sw \$8, 0(\$29) sw \$9, 4(\$29) sw \$10, 8(\$29) sw \$11, 12(\$29) sw \$12, 16(\$29) sw \$13, 20(\$29) sw \$14, 24(\$29) sw \$31, 28(\$29) move \$13, \$4 move \$14, \$5 move \$4, \$5 addi \$5, \$0, 2 move \$5, \$14 subi \$14, \$14, 1 move \$4, \$13 jal div move \$8, \$0</pre>	<pre>for: beq \$8, \$6, exit subi \$10, \$14, \$8 muli \$9, \$8, 4 muli \$10, \$10, 4 add \$9, \$9, \$13 add \$10, \$10, \$13 lw \$11, 0(\$9) lw \$12, 0(\$10) sw \$11, 0(\$10) sw \$12, 0(\$9) addi \$8, \$8, 1 j for exit: lw \$8, 0(\$29) lw \$9, 4(\$29) lw \$10, 8(\$29) lw \$11, 12(\$29) lw \$12, 16(\$29) lw \$13, 20(\$29) lw \$14, 24(\$29) lw \$31, 28(\$29) addi \$29, \$29, 32 jr \$31</pre>
---	---

Si noti che non è necessario salvare e ripristinare \$4 e \$5 usando lo stack, visto che i loro valori sono temporaneamente copiati in \$13 e \$14 e successivamente "ripristinati" usando l'istruzione move (quindi di fatto il corpo della procedura non ne altera il valore).