

Architettura dei Calcolatori Elettronici

Esercizi di programmazione Assembly

Ogni esercizio è associato ad una difficoltà indicata tra parentesi a fianco del numero dell'esercizio stesso. Gli esercizi di difficoltà 5 e 6 sono dati come stimolo per gli studenti interessati e non sono pertanto fondamentali per il superamento dell'esame.

Esercizio 1 (1)

Dato un vettore di n unsigned word, individuare il minimo. Il vettore deve essere esplicitamente dichiarato come variabile globale; il minimo deve essere salvato all'interno di un'altra variabile globale esplicitamente dichiarata.

Esercizio 1.b (2)

Dato un vettore di n signed word, individuare il massimo. Il vettore deve essere esplicitamente dichiarato come variabile globale. Il codice deve essere organizzato come una funzione il cui passaggio di parametri (indirizzo del vettore) ed il valore di ritorno (il massimo) devono rispettare le System V ABI calling conventions.

Esercizio 2 (1)

Scrivere un programma che, dato un vettore di n longword, salvi in un secondo vettore gli elementi memorizzati nel primo in ordine inverso.

Esercizio 3 (2)

All'indirizzo 1280h è presente un vettore di 64 signed longword. Implementare in Assembly l'algoritmo [bubble sort](#) per ordinare gli elementi di questo vettore.

Esercizio 4 (2)

All'indirizzo 5555h è presente un vettore di n unsigned word. Scrivere un programma Assembly che memorizzi all'interno del registro %eax il numero di elementi pari presenti in posizione dispari all'interno del vettore (ossia, se in posizione 1 nel vettore è presente un numero pari, %eax viene incrementato, se in posizione 3 è presente un numero pari %eax viene incrementato, e così via). Il numero n di elementi del vettore può essere sia pari che dispari.

Esercizio 5 (3)

Dato un vettore di 32 unsigned byte, scandire il vettore e creare una maschera di bit all'interno del registro %eax in cui un bit viene impostato ad 1 se l'elemento corrispondente del vettore è pari, a 0 altrimenti.

Esercizio 6 (4)

Scrivere una subroutine *divisione* che implementi via software l'operazione di divisione tramite il metodo delle sottrazioni successive. La funzione opera su interi non segnati e rispetta le System V ABI calling conventions. Scrivere un programma che utilizzi questa funzione per calcolare il risultato della divisione tra due numeri specificati all'interno di due variabili globali.

Esercizio 7 (4)

Utilizzare la funzione *divisione* realizzata per l'Esercizio 6 per implementare un programma che calcoli il massimo comune divisore tra due numeri utilizzando il metodo delle divisioni successive.

Esercizio 8 (5)

Data una variabile globale 'a' di tipo word, memorizzare all'interno di un'altra variabile globale 'b' di dimensione longword la rappresentazione dello stesso dato contenuto in 'a' rappresentato mediante un codice correttore d'errore a distanza di Hamming 3. I bit di parità debbono essere inseriti in posizione corrispondente ad una potenza di 2 (1, 2, 4, 8, ...).

Esercizio 9 (6)

Data una variabile globale 'a' di tipo longword che memorizza un dato rappresentato con codice di correzione d'errore a distanza di Hamming 3, implementare: 1) una subroutine Assembly che restituisca il valore booleano 'vero' se il dato passato non ha alcun errore, 'falso' altrimenti; 2) un subroutine Assembly che restituisca la rappresentazione originale del dato passato (eliminando, quindi, i bit di parità). Scrivere un programma Assembly che utilizzi queste due subroutine per verificare la correttezza del dato e (in caso positivo) decodifichi lo stesso.

Esercizio 10 (6)

Dato un numero intero rappresentato in aritmetica segnata, scrivere un programma assembly che converta questo numero nella rappresentazione in virgola mobile IEEE 754.