Architettura dei Calcolatori a.a. 2016/17
Prova di Laboratorio - assembly MIPS: conta_uni
27 Giugno 2017

Introduzione

Lo scopo di questa prova di laboratorio è lo sviluppo di un semplice programma nel linguaggio assembly del processore MIPS. Non è richiesta una particolare base di conoscenze algoritmiche, ma semplicemente un minimo di dimestichezza con la programmazione assembly.

Istruzioni

Cominciate facendo il login sulla macchina del laboratorio che vi è stata assegnata. Per il login occorre usare matricola e password dello *student portal*. Nel disco Z: troverete una cartella (`mips`) contenente il simulatore Mars. Tutto il vostro codice (sia esso costituito da un singolo file, o da file multipli) andrà salvato nella cartella "`mips`" da creare sul drive `H:` .

Create un file `student-info.txt` con incluso il vostro nome e cognome e numero di matricola nella cartella "`mips`". Per maggior sicurezza, includete anche nome, cognome e matricola come commento, in testa ad ogni file sorgente. Alla fine della prova, i file saranno prelevati automaticamente dalla directory. Tutto quello che lascerete nella cartella `mips` sarà utilizzato per la valutazione. Salvare i vostri file altrove, o non indicare nome e cognome, porterà inevitabilmente all'annullamento della vostra prova. *Tutti i file all'esterno della cartella verranno cancellati automaticamente!!!*

Le specifiche

Dovete scrivere un programma assembly che legge un numero da tastiera utilizzando la system call per la lettura di interi, e ne stampa il numero di 1 contenuto **nella sua rappresentazione binaria** utilizzando una funzione (`count_ones`), che riceve in input in $a0 il puntatore all'intero e ritorna in $v0 il conteggio degli "1", che sarà successivamente stampato dal main. La chiamata della funzione dovrà rispettare le convenzioni per il salvataggio dei registri. Utilizzate le system call MARS per l'I/O e la terminazione del main. Per esempio, in numero di "1" nel numero 17892, che in binario è 0100010111100100 (con un po' di zeri davanti), è 7.

È fondamentale che il programma sia in grado di accettare in input anche numeri negativi, calcolandone correttamente la somma delle cifre.

Suggerimenti

Il seguente è un output di esempio:

```
Dammi un numero (-2^31<= x <2^31):
1234567
il numero di 1 nella rappresentazione binaria del numero è 11
-- program is finished running --

Reset: reset completed.

Dammi un numero (-2^31<= x <2^31):
-17892
il numero di 1 nella rappresentazione binaria del numero è 7
-- program is finished running --
```

Valutazione

Scrivere un programma funzionante, che faccia uso di una funzione come richiesto nelle specifiche e che segua le convenzioni di salvataggio dei registri è strettamente necessario per essere ammessi a sostenere l'orale. In quella sede, si entrerà nel dettaglio della struttura del codice, dando una valutazione migliore a soluzioni "pulite" e ben commentate.