Laboratorio di Architettura degli Elaboratori

Davide Quaglia
Dipartimento di Informatica – Università di Verona

Precisazioni su alcuni aspetti dell'Assembly

Come ottenere l'indirizzo di memoria di una variabile

Data una variabile `messaggio` dichiarata nella sezione `.data` e' possibile ottenere il suo indirizzo (ad es. per stamparla); a tal fine le seguenti due righe di codice sono equivalenti

```
movl $somma, %eax
leal somma, %eax
```

Indirizzamento indicizzato

Di seguito sono riportati due modi equivalenti per recuperare un valore dalla memoria dato un indirizzo di base e un indice.

Modo 1:

```
movl 8(%esp), %eax
```

Modo 2:

```
movl %esp, %ecx
movl $8, %edx
movl (%ecx, %edx), %eax
```

Utilizzo dei registri nella divisione

Nella divisione (DIV e IDIV) se il divisore e' a 16 o 32 bit, il dividendo e' contenuto nella coppia di registri DX:AX oppure EDX:EAX rispettivamente. Se il divisore è una word, il valore ottenuto concatenando il contenuto di DX e AX viene diviso per l'operando (i 16 bit più significativi del dividendo devono essere memorizzati nel registro DX), il quoziente viene memorizzato nel registro AX e il resto in DX. Se il divisore è un long, il valore ottenuto concatenando il contenuto di EDX e EAX viene diviso per l'operando (i 32 bit più significativi del dividendo devono essere memorizzati nel registro EDX), il quoziente viene memorizzato nel registro EAX e il resto in EDX.

ATTENZIONE: occorre azzerare DX (EDX) prima della moltiplicazione con divisore a 16 (32) bit se il dividendo sta tutto in 16 (32) bit.

Cicli FOR in Assembly

Di seguito e' riportato un esempio di ciclo FOR che ripete un blocco di istruzioni per 10 volte

```
     movl $10, %ecx
inizio_ciclo:
     .... # corpo del ciclo FOR
     loop inizio_ciclo
```

```
     .... # istruzioni successive
```

Lettura di un numero da tastiera

Di seguito e' riportato lo pseudo-codice per lettura di un numero da tastiera. Si assume che dopo l'ultima cifra venga premuto il tasto INVIO. Le variabili `acc` e `valore` sono di tipo `long` mentre la variabile `lettera` e' di tipo byte. Il numero viene letto un carattere alla volta utilizzando l'apposito servizio del sistema operativo.

```
     acc = 0
inizio:
     # lettura del carattere mediante Sistema Operativo
     lettera = codice ASCII del carattere letto
     cmp $10, lettera  # controllo la pressione di INVIO
     je fine
     valore = lettera - 48
     acc = acc * 10
     acc = acc + valore
     jmp inizio
fine:
     ... # ora nella variabile acc c'è il numero letto
```

NOTA: il precedente frammento di codice potrebbe essere usato in una funzione `ktoi` (keyboard to integer) che legge da tastiera un numero delimitato da INVIO e restituisce nel registro %EAX il corrispondente valore intero.

Conversione di una stringa numerica in un numero

Di seguito e' riportato lo pseudo-codice per la conversione di una stringa numerica (ad es. introdotta come parametro in linea di comando) in un numero. Si assume che dopo l'ultima cifra sia presente il codice ASCII NULL (0). Le variabili `acc` e `valore` sono di tipo `long` mentre la variabile `lettera` e' di tipo byte. Tale frammento di codice potrebbe essere usato in una funzione `atoi` (ascii to integer) che prende nel registro %ECX l'indirizzo della prima lettera della stringa e restituisce in %EAX il risultato numerico.

```
     acc = 0
     %edx = 0
inizio:
     movb (%ecx, %edx), lettera
     cmp $0, lettera  # controllo di fine stringa
     je fine
     valore = lettera - 48
     acc = acc * 10
     acc = acc + valore
     %edx = %edx + 1
     jmp inizio
fine:
     movl acc, %eax    # ora in %eax c'è il numero letto
```