Prova pratica di Calcolatori Elettronici (nucleo v6.*)

C.d.L. in Ingegneria Informatica, Ordinamento DM 270

1 febbraio 2012

1. Vogliamo aggiungere la schedulazione di tipo *Round Robin* al nucleo. In questo tipo di schedulazione ogni processo può restare continuamente in esecuzione soltanto per certo tempo massimo, dopo di che deve cedere il processore ad un altro processo (il primo in coda pronti). Nel cedere il processore, il processo si reinserisce in coda pronti come ultimo. Poichè nel nucleo abbiamo anche le priorità, prevediamo che il processo si inserisca come ultimo tra quelli che hanno la sua stessa priorità (ma prima di quelli con priorità inferiore). In questo modo i processi di uguale priorità vengono eseguiti a rotazione.

   Per realizzare questo tipo di schedulazione aggiungiamo al descrittore di processo un campo `int quanto`, che rappresenta il numero di tick del timer che il processo ha ancora a disposizione prima di dover cedere il processore. Il numero massimo di tick per i quali ciascun processo può restare in modo continuato in esecuzione è contenuto nella costante `MAX_QUANTO`. Ogni volta che il driver del timer va in esecuzione, decrementa il campo `quanto` del processo attualmente in esecuzione e provvede ad eseguire le azioni necessarie ad implementare quanto sopra illustrato.

   Aggiungiamo infine le seguenti primitive:

   - `void abilita_rr()` (da realizzare): abilita la schedulazione Round Robin. Il driver del timer deve eseguire quanto sopra specificato solo se la schedulazione Round Robin è abilitata.
   - `void disabilita_rr()` (da realizzare): Disabilita la schedulazione round robin. Se questa era abilitata provvede anche a riportare a `MAX_QUANTO` i campi `quanto` di tutti i processi esistenti nel sistema.

   Modificare i file `sistema.cpp` e `sistema.s` in modo da realizzare le primitive e il codice mancante.