Prova pratica di Calcolatori Elettronici (nucleo v6.*)

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

22 giugno 2011

1. Vogliamo aggiungere al nucleo il meccanismo delle interruzioni inter-processo. Un qualunque processo può inviare una interruzione ad un altro processo di cui conosce l'identificatore. L'interruzione setta un flag nel descrittore del processo destinatario. Ogni processo può esaminare lo stato del proprio flag con una opportuna primitiva. Se il processo destinatario è bloccato su un semaforo l'operazione di interruzione provvede anche a risvegliarlo.

   A tale scopo aggiungiamo i seguenti campi al descrittore di ogni processo:

   ```
   bool  interrupted;
   natl  blocked;
   ```

   Il campo `interrupted` è il flag di interruzione, posto a `false` alla creazione del processo. Il campo `blocked` è l'indice dell'eventuale semaforo su cui il processo è bloccato (`0xFFFFFFFF` se il processo non è bloccato su un semaforo).

   Aggiungiamo inoltre le seguenti primitive:

   - `bool sem_wait2(natl sem)`: Opera su normali semafori. Si comporta come una normale `sem_wait`, ma se il processo si deve bloccare ed è stato interrotto (flag `interrupted` settato) resetta il flag, restituisce `false` e termina. Quando termina normalmente (perché il contatore non era 0 oppure perché il processo è stato risvegliato da una `sem_signal2`) restituisce `true`. Aggiorna opportunamente i campi nel descrittore di processo.
   - `void sem_signal2(natl sem)` (già realizzata): Opera su normali semafori. Si comporta come una normale `sem_signal`, ma aggiorna opportunamente i campi nel descrittore di processo e completa il funzionamento della `sem_wait2`.
   - `bool interrupt(natl id)`: Se `id` non corrisponde ad un processo esistente restituisce `false` e termina. In tutti gli altri casi restituisce `true`. Se il processo era già stato interrotto non fa altro. Se il processo non era stato interrotto e non è bloccato su un semaforo si limita a settare l'opportuno flag. Infine, se il processo non era stato interrotto ed è bloccato su un semaforo lo sblocca. Aggiorna opportunamente i campi del descrittore di processo e del semaforo e gestisce eventuali *preemption*.
   - `bool interrupted()` (già realizzata): restituisce il valore del flag `interrupt` del processo corrente e lo resetta.

   Sono disponibili le seguenti funzioni (già realizzate):

   - `des_proc* des_p(natl id)`: restituisce un puntatore al descrittore di processo del processo di identificatore `id`. Restituisce 0 se `id` non è valido.
   - `proc_elem* elimina_da_lista(proc_elem*& testa, des_proc* p)`: rimuove dalla lista `testa` il `proc_elem` che punta al processo con descrittore puntato da `p`, se presente. Restituisce un puntatore al `proc_elem` rimosso, se trovato, altrimenti restituisce 0.

   ATTENZIONE: una primitiva che usa `salva_stato` e `carica_stato` può restituire valori al chiamante, ma per farlo deve modificare il campo del descrittore di processo che contiene il valore del registro `%RAX`.

   Modificare i file `sistema.cpp` e `sistema.s` in modo da realizzare le primitive mancanti.