Prova pratica di Calcolatori Elettronici (nucleo v6.*)

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

2 luglio 2014

1. Una *barriera* è un meccanismo di sincronizzazione tra processi che funziona nel modo seguente: la barriera è normalmente chiusa; se un processo arriva alla barriera e la trova chiusa, si blocca; la barriera si apre solo quando sono arrivati tutti i processi, che a quel punto si sbloccano; una volta aperta e sbloccati tutti i processi, la barriera si richiude e il meccanismo si ripete.

   Vogliamo realizzare il meccanismo della barriera nel nucleo, ma solo tra i processi che si sono preventivamente registrati. I processi possono registrarsi e deregistarsi per una barriera in qualunque momento. La deregistrazione di un processo può causare l'apertura di una barriera, se tutti gli altri processi registrati erano già arrivati.

   Per realizzare il meccanismo aggiungiamo le seguenti primitive:

   - `void reg()` (da realizzare): registra il processo corrente sulla barriera. È un errore se un processo tenta di registrarsi due volte.
   - `void dereg()` (da realizzare): deregistra il processo corrente sulla barriera. È un errore se un processo tenta di deregistrarsi senza essere registrato.
   - `void barrier()` (da realizzare): fa giungere il processo corrente alla barriera. È un errore se un processo invoca questa primitiva senza essersi registrato.

   Le primitive abortiscono il processo chiamante in caso di errore e tengono conto della priorità tra i processi.

   Modificare i file `sistema.cpp` e `sistema.s` in modo da realizzare le primitive appena descritte. Può essere necessario definire nuove strutture dati e aggiungere campi al descrittore di processo.