Prova pratica di Calcolatori Elettronici

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

2 luglio 2015

1. Siano date le seguenti dichiarazioni, contenute nel file `cc.h`:

   ```
   // [omitted]
   ```

   Realizzare in Assembler GCC le funzioni membro seguenti.

   ```
   // [omitted]
   ```

2. Nel nucleo che abbiamo studiato tutta la memoria, con l'esclusione delle pile, è condivisa tra tutti i processi. Vogliamo aggiungere un meccanismo tramite il quale solo alcuni processi, e non necessariamente tutti, possano condividere della memoria.

   Prevediamo quindi che un processo possa creare delle zone di memoria, dette `shmem`, ciascuna con un identificatore unico. I processi che vogliono accedere ad una `shmem` devono aggiungerla al proprio spazio di indirizzamento, specificandone l'identificatore. Una volta aggiunta, la `shmem` sarà disponibile contiguamente all'interno della parte utente/condivisa dello spazio di indirizzamento del processo. Un processo può aggiungere più `shmem` al proprio spazio e le diverse `shmem` non devono sovrapporsi. Non è importante che i processi che condividono una stessa `shmem` la vedano tutti allo stesso indirizzo.

   Per descrivere una `shmem` aggiungiamo al nucleo la seguente struttura dati:

   ```
   struct des_shmem {
       natl npag;
       natl first_frame_number;
   };
   ```

   Il campo `npag` contiene la dimensione (in pagine) della `shmem`. Tutti i frame che contengono la `shmem`, nell'ordine in cui devono comparire nella memoria di tutti i processi che la condividono, sono mantenuti in una lista, implementata tramite l'array `vdf`. Il numero del primo frame della lista è contenuto nel campo `first_frame_number` e l'elmento `vdf[fn]` di ogni `fn` della lista contiene il numero del frame successivo (il valore 0 termina la lista).

   Inoltre, aggiungiamo il seguente campo ai descrittori di processo:

   ```
   vaddr avail_addr;
   ```

   Questo campo contiene il primo indirizzo libero nella parte utente/condivisa del processo. Tutti gli indirizzi da `avail_addr` fino a `fin_utn_c` (escluso) sono disponibili per contenere zone di memoria condivisa.

   Aggiungiamo infine le seguenti primitive:

- `natl shmem_create(natl npag)` (tipo 0x5c, già realizzata): Crea una nuova zona di memoria condivisibile tra più processi, grande `npag` pagine, e ne restituisce l'identificatore. La primitiva si limita ad allocare i frame necessari e a inserirli in una lista. Se non vi sono frame liberi a sufficienza, restituisce 0xFFFFFFFF.
- `vaddr shmem_attach(natl id)` (tipo 0x5d, da realizzare): Permette ad un processo di aggiungere la `shmem id` al proprio spazio di indirizzamento e ne restituisce l'indirizzo di partenza. Abortisce il processo se `id` non corrisponde ad una `shmem` esistente. Restituisce 0 se non è stato possibile aggiungere la zona (perché non vi sono indirizzi liberi sufficienti a contenerla).

Modificare i file `sistema.cpp` e `sistema.S` in modo da realizzare le primitive appena descritte.