RELAZIONE DEL PROGETTO DI SISTEMI OPERATIVI – Fase 2

CREATO DA ALICE BENATTI, LIBERA LONGO, GERALD MANZANO, ALBERTO SCUDERI

Difficoltà implementative

Durante lo sviluppo della fase due abbiamo riscontrato le seguenti difficoltà implementative:

- **Do_Io**:

  Durante lo sviluppo di tale funzione il gruppo ha avuto difficoltà nella comprensione dei problemi legati ad essa.

  Per motivi a noi sconosciuti, la funzione print viene considerata come una funzione di lettura anziché di scrittura, nonostante sia preferenziata la scrittura nell'ordine dei controlli dei device.

  Abbiamo provato a contattare i professori ma, purtroppo, i problemi sussistevano.

- **Device_Interrupt**:
  Esattamente come per la Do_Io, abbiamo riscontrato un problema legato alla comprensione del funzionamento dei device. Dopo diverso tempo siamo arrivati ad una presumibile comprensione ma senza riuscire a risolvere i problemi.
- **P_operation, V_operation**:
  Dopo aver convertito i print del file p2test.c in klog_print per testare le altre funzioni abbiamo comunque riscontrato dei problemi probabilmente dovuti a P_operation e V_operation, giungendo alla conclusione che con molte probabilità i problemi riscontrati nella Do_Io siano legati a ciò.

Nonostante ciò siamo riusciti a risolvere tutte le difficoltà e a **concludere la fase2**.

Scelte implementative

File

Per la fase due abbiamo scritto i seguenti file:

- **main.c:** funzione di inizializzazione.
- **scheduler.c**: ha il compito di pianificare il processo successivo da eseguire.
- **exceptions.c**: gestisce tutti i tipi di eccezioni (interrupt, syscall, timer, devices).

- **syscall.c**: Gestione delle 10 syscall e delle loro funzioni ausiliarie descritte anche nel punto seguente.
- **Memcpy.c:** file contenente la funzione memcpy descritta in seguito e delle versioni “ufficiali” prese dalle librerie di C lasciate commentate.

Funzioni ausiliare

- **BusySem**: funzione che ha il compito di cercare la key nella ASL.
- **FindPCB**: funzione che ricerca uno specifico pcb bloccato ad un semaforo.
- **Auxiliary_Terminate**: funzione ausiliaria per la Terminate_Process, col compito di terminare l’albero dei figli ricorsivamente.
- **P_operation**: P viene richiesto dal processo chiamante inserendo il valore 3 in a0, l’indirizzo fisico del semaforo da inserire in a1, e successivamente viene eseguita l’istruzione syscall.
- **V_operation**: questo servizio richiede al nucleo di eseguire un’operazione V su un semaforo.
- **memcpy**: funzione che copia uno state_t* in uno state_t.
- **Blocking_Syscall**: dato che il valore del pc deve essere incrementato di quattro per evitare un ciclo infinito utilizziamo questa funzione di blocco e richiama lo scheduler.

Syscall

- **Yield**: decide lei quale processo avviare senza richiamare lo scheduler.