Report Progetto Sistemi Operativi Avanzati

Daniele Ferrarelli

1 Introduzione

Il progetto consiste in un device driver che implementa due flow di dati con diverse gestioni delle scritture. Il flow a bassa priorità viene gestito tramite deferred work. Entrambi i flow sono sincronizzati per scritture e letture. Le operazioni di scrittura avvengono nella coda del flow, mentre le operazioni di lettura avvengono in testa. Il driver supporta 128 device diversi identificati tramite il loro minor number e pone un limite al numero massimo di byte presente in ogni flow di ogni device, questo è configurabile tramite parametro del modulo.

1.1 Stato del deivce

Lo stato di ogni device viene mantenuto utilizzando una struttura che contiene al suo interno i metadati del device, come per esempio il numero di byte validi o il numero di byte che verranno aggiunti tramte deferred work.

Viene inoltre mantenuto un indice che permette di ottenere fino a che punto è avvenuta la lettura sul primo blocco del flow. Questo viene fatto poichè in caso di letture parziali di un blocco bisogna tenere traccia dei byte che sono stati già letti.

2 Scelte Implementative

Il driver per implementare queste funzionalità utilizzerà una serie di liste collegate che vengono sincronizzate in lettura e scrittura tramite mutex. Queste liste vengono mantenuto nella struttura dati che contiene i metadati del device.

2.1 Scrittura

La prima fase della scrittura consiste nel creare una serie di nodi della lista collegata separando in più nodi i dati in input al driver. Questo viene fatto per rientrare nei limite della chiamata kmalloc che viene utilizzata per allocare i buffer. Dalla prima fase della scrittura si produrrano una lista collegata con nodi di dimensione minore o uguale alla massima dimensione permessa per ogni nodo (impostata tramite parametro del modulo).

La scrittura nel device avviene secondo due metodi diversi a seconda della priorità corrente del device. Nel caso sia una scrittura a bassa priorità si utilizzerà una work queue per aggiungere i dati al flow tramite un work handler, questa work queue è single thread e privata per ogni device.

Nel caso sia una scrittura ad alta priorità si andrà ad aggiungere direttamente gli elementi alla lista collegata che implementa il flow.

Nel caso sia impostata una chiamata bloccante si utilizza la macro wait_queue_interruptible_timeout per bloccare il device fino a che si ha abbastanza spazio per scrivere o il timeout scade. Per eseguire questo controllo si utilizza la funzione can_write che andrà a controllare il numero di byte validi, il numero di byte che devono essere aggiunti tramite deferred work e il numero di byte da scrivere. Al suo interno si ottiene il lock sul flow dei dati per controllare lo device, se si può scrivere verrà mantenuto il lock per aggiornare lo stato del dispositivo, in caso contrario verrà restituito l'errore -ENOSPC. I thread vengono aggiunti ad una wait queue privata per ogni device e vengono risvegliati al momento di una lettura che potrebbe aver liberato abbastanza spazio per eseguire la scrittura.

2.2 Lettura

La lettura avviene nello stesso modo per entrambe le priorità implementate. Si andrà a prendere il lock per poi andare a scorrere la lista e prelevare il numero corretto di byte dal flow. Si utilizzerà poi copy_to_user per copiarli nel buffer di lettura. Nel caso un nodo venga completamente letto si andrà a liberare i buffer dei nodi utilizzando la chiamata kfree, altrimenti si andrà ad aggiornare un valore che indica fino a che punto è avvenuta la lettura sul blocco corrente.

Nel caso sia una chiamata bloccante si utilizzerà la macro wait_queue_interruptible_timeout per bloccare i thread su una wait queue finchè non sono disponibili abbastanza dati o il timeout scade. Al termine del timeout se non si hanno ancora abbastanza dati si andrà eseguire una lettura parziale dei dati. I thread vengono risvegliati al momento di una scrittura che andrà ad inserire abbastanza dati all'interno del flow. Stessa cosa succede nel caso di chiamate non bloccanti.

3 VFS

Tramite SYSFS vengono esposte le configurazioni dei vari device e le loro statistiche. Per fare ciò si crea la cartella /sys/hlm che contiene una serie di cartelle riguardanti tutti i minor number supportati. All'interno di queste cartelle sono presenti file che rappresentano lo stato del device (timeout, block, enabled, priority) e le statistiche per ogni device (asleep_hi, asleep_lo, bytes_lo, bytes_hi).

4 User

E' stata creata una cli per permettere di interagire con i flow del device driver. Per utilizzarla bisognerà andare ad eseguire la cli con primo argomento il path dell'file che utilizzerà il driver. Per creare file si può utilizzare lo script create_node.sh. Questo script prenderà in input il nome del file da creare ed il minor number da assegnare.