[OS Internals] LAB 2.1 - QUESITI

Esempi domande d'esame

1 Quesito #1 (serve OS161 - Memory???)

Si ha la seguente implementazione di `sys_write()`:

```
int sys_write(int fd, userptr_t buf_ptr, size_t size) {
    int i;
    char *p = (char *)buf_ptr;
    if (fd!=STDOUT_FILENO && fd!=STDERR_FILENO) {
        kprintf("sys_write supported only to stdout\n");
        return -1;
    }
    for (i=0; i<(int)size; i++)
        putch(p[i]);
    return (int)size;
}
```

Supponendo che standard input, output ed error siano mappati alla console (quindi non è possibile nessun loro reindirizzamento a file), si dica se è possibile sostituire la gestione di stdout/stderr mediante il ciclo for con uno o più dei frammenti di codice. Per ogni alternativa rispondere sì/no e fornire una motivazione.

2 Quesito #2

Si supponga un sistema OS/161. Descrivere la struttura **trapframe** e il suo utilizzo durante le chiamate di sistema. In particolare, indicare la funzione del registro `v0` dell'architettura MIPS.

In OS161, la **trapframe** è una struttura (`struct` del C) i cui campi sono associati ai registri dell'architettura del microprocessore MIPS; viene utilizzata per salvare le informazioni dei registri della CPU nel momento in cui nel sistema si verifichi un'**eccezione**. Essa è utilizzata, tra le altre cose, anche durante le chiamate di sistema (il microprocessore solleva un'eccezione verso il SO) e contiene - in questo caso - le informazioni di supporto utili alla loro gestione. La trapframe viene utilizzata nello specifico dal **dispatcher** delle system call integrato nel kernel che prende in esame:

1. Il campo `tf->v0`, per contenere (prima) il codice della syscall (dal μP al kernel) e dopo l'esecuzione della syscall (dal kernel al μP) il **valore di ritorno**.
2. I campi da `tf->a0` a `tf->a2` (associati ai rispettivi registri `a0-a2`) contengono gli **argomenti** della chiamata di sistema;
3. Il campo `tf->a3`, associato al relativo registro MIPS, contiene alla fine dell'esecuzione della syscall, eventuali condizione di errore.

3 Quesito #3

Si supponga un sistema OS/161. È necessario l'utilizzo di una chiamata di sistema per la scrittura su stdin (es. sys_write) per un processo utente? È possibile utilizzare `kprintf()` invece?

Motivare le risposte.

In OS161 per poter scrivere sullo stream `stdin` occorre che ci sia una system call apposita che permetta di gestire il processo utente. Come tanti altri componenti, non è implementata in OS161, va realizzata seguendo lo schema ad esempio della `write()` dell'ANSI/C. Una `kprintf()` non potrebbe essere impiegata in quanto è riservata all'utilizzo da parte del kernel.