

Interruzioni

Meccanismo che permette la gestione uniforme di:

operazioni asincrone di I/O / timer la CPU configura un handler di interruzione, invia comandi e parametri e prosegue l'esecuzione. In background la *interrupt unit* legge in loop lo stato del dispositivo, e segnala un'interruzione quando ha completato l'operazione;

chiamate di sistema interruzioni software sincrone, permettono il passaggio da user a kernel mode;

eccezioni sincrone, segnalano errori (e.g. segfault e divisioni per 0).

Per l'implementazione, è necessario considerare:

interrupt vector stabilito dal SO, contiene gli indirizzi degli interrupt handler. La interrupt unit fornisce un valore che indica chi ha causato l'interrupt, da usare come indice nell'IV;

stack (kernel) separato da quello dei programmi in esecuzione, usato per salvare i registri modificati dagli handler;

masking durante l'esecuzione di interrupt con alta priorità è utile disabilitare gli interrupt minori. Si distinguono interrupt *mascherabili* e non, e handler non interrompibili ("monolitici", disabilitano gli interrupt) e thread kernel (interrompibili);

trasferimento atomico del controllo

trasparenza il codice utente non deve aver bisogno di tenere in considerazione l'esistenza degli interrupt.