Driver UART

Termen de predare: Duminica 19 Aprilie, ora 23.59 (2009-04-19)

Cuprins

- 1 Enunț
- 2 Precizări generale
- 3 Precizari Linux
- 4 Precizari Windows
- 5 Testare
- 6 Intrebari

Enunț

Să se scrie un modul de kernel care să implementeze un driver pentru portul serial (`UART16550`). Device driver-ul trebuie să suporte cele două porturi seriale standard dintr-un PC, `COM1` şi `COM2` (`0x3f8`, `0x2f8`). În afară de rutinele standard care trebuie suportate (`open`, `read`, `write`, `close`), driver-ul trebuie să suporte şi schimbarea parametrilor de comunicație cu ajutorul unei operații `ioctl` sau `DeviceIoControl` (`UART16550_IOCTL_SET_LINE`).

Driverul trebuie să folosească întreruperi atât pentru recepție cât şi pentru transmisie, pentru a reduce latența şi timpul de utilizare a procesorului. De asemenea, apelurile `read` şi `write` trebuie să fie blocante. Temele care nu respectă aceste cerințe nu se iau în considerare. Este indicat să folosiți în cadrul driver-ului un buffer de citire şi un buffer de scriere pentru fiecare port serial.

Un apel read blocant înseamnă că rutina de read apelată din user-space se va bloca până la citirea a **cel puțin** un octet (buffer-ul de read din kernel este gol şi nu se pot citi date). Un apel write blocant înseamnă că rutina de write apelată din user-space se va bloca până la scrierea a **cel puțin** un octet (buffer-ul de write din kernel este plin şi nu se pot scrie date).

Precizări generale

- documentație despre portul serial găsiți aici

Precizari Linux

- driverul va fi accesat ca un device driver de tip caracter, cu MAJORUL **42**, minorul **0** pentru `COM1` şi minorul **1** pentru `COM2`;
- header-ul cu definitiile necesare pentru operațiile speciale îl găsiți aici;
- un bun punct de pornire în implementarea rutinelor de read/write este exemplul de character device driver de conversie upper-case de la laboratorul 4; singura diferență este că trebuie să folosiți două buffere, unul pentru read şi altul pentru write;
- pentru citirea/scrierea datelor în/din porturi nu trebuie să folosiți funcții amânabile (puteți să faceți totul din întrerupere);

- va trebui să sincronizați rutinele de read/write cu rutina de tratare a întreruperii pentru ca rutinele sa fie blocante; este recomandat să folosiți cozi de așteptare.

Precizari Windows

- driverul va fi accesat din user-space cu **\\.\uart0** pentru `COM1` şi **\\.\uart1** pentru `COM2`;
- header-ul cu definițiile necesare pentru operațiile speciale îl găsiți aici;
- pentru a putea lucra fără probleme cu porturile seriale, trebuie sa dezinstalați ACPI-ul şi să faceți disable pe `COM1` şi `COM2` (maşina virtuală pusă la dispoziție este configurată corespunzător);
- o să aveți nevoie de două buffere interne pentru fiecare port în care să puneți datele (un buffer de read şi un buffer de write); ca să nu vă complicați cu sincronizări între ISR şi rutina de read folosiți funcțiile `Interlocked...`;
- un bun punct de pornire în implementarea rutinelor de read/write este exemplul de driver de conversie upper-case de la laboratorul 4;
- pentru write, cel mai simplu e să țineți minte într-o listă IRP-urile şi să le serviți atunci când vă indică întreruperea; cum nu aveți voie să utilizați `IoCompleteRequest` într-o ISR, va trebui să folosiți DPC-uri;
- puteți apela `IoCompleteRequest` direct din rutina DPC sau vă puteți sincroniza cu rutinele de read/write folosind evenimente şi apela de acolo `IoCompleteRequest`.

Testare

Pentru simplificarea procesului de corectare a temelor, dar şi pentru a reduce greşelile temelor trimise, corectarea temelor se va face automat cu ajutorul unor teste publice (linux, windows). Testele presupun că numele modului de kernel este **uart16550** atât pe Linux cât şi pe Windows.

Depunctari pentru probleme constatate in implementarea temei:

- -1.0 warning-uri la compilare
- -0.5 functionalitate incompleta
- -0.3 utilizare incorecta a device-ului
- -0.2 lipsa readme sau readme necorespunzator
- -0.1 lock si posibilitate de pierdere unlock
- -0.1 down si posibilitate de pierdere up
- -0.1 disable si posibilitate de pierdere enable
- -0.1 memory leak
- -0.1 utilizare functii la nivele IRQL necorespunzatoare
- -0.1 alte probleme constatate
- -0.0 observatii

In cazuri exceptionale (tema trecere testele prin nerespectarea cerintelor) se poate scadea mai mult decat este mentionat mai sus.

Intrebari

Pentru intrebari legate de tema puteti consulta arhivelele listei de discutii sau puteti trimite un e-mail (trebuie sa fiti inregistrati).