Tema 2 - Driver UART

Termen de predare: **joi, 7 Aprilie 2011, ora 23:00**

Obiectivele temei

- Consolidarea noţiunii de device driver;
- Citire documentaţie hardware. Urmărirea funcţionalităţii dorite în documentaţie;
- Lucrul cu întreruperi. Folosire funcţii neblocante în context întrerupere(L)/funcţii la nivele IRQL corespunzătoare(W);
- Folosirea buffere-lor, sincronizare;
- Module cu parametri;
- DPC-uri.

Enunţ

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 implementate (`open`, `read`, `write`, `close`), driver-ul trebuie să aibă suport şi pentru schimbarea parametrilor de comunicaţie cu ajutorul unei operaţii `ioctl` sau `DeviceIoControl` (`UART16550_IOCTL_SET_LINE`).

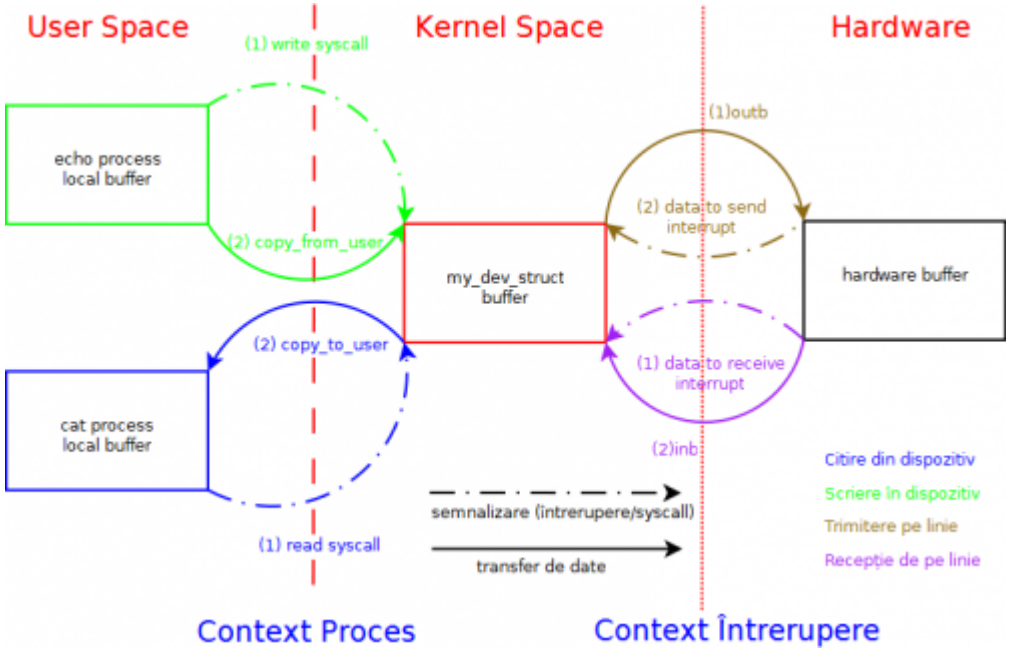
Driverul trebuie să folosească înteruperi atât pentru recepţie cât şi pentru transmisie, pentru a reduce latenţa şi timpul de utilizare a procesorului. De asemenea, apelurile `read` şi `write` trebuie să fie blocante. Temele care nu respectă aceste cerinţe nu se iau în considerare. Este indicat să folosiţi 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 puţin'` 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 puţin'` un octet (buffer-ul de write din kernel este plin şi nu se pot scrie date).

Precizări generale

- documentaţie despre portul serial găsiţi aici
- sau pe tldp
- aici mai aveţi o tabela cu registre
- datasheet 16550

Schemă buffer-e



Transferele între diversele buffer-e sunt probleme de tipul Producător-Consumator. Exemple:

- dacă se scrie dintr-un proces în dispozitiv, procesul este producător şi dispozitivul consumator; procesul se va bloca până există loc în buffer(-ul consumatorului)
- dacă se citeşte dintr-un proces de la dispozitiv, procesul este consumator şi dispozitivul consumator; procesul se va bloca până există cel puţin un element în buffer(-ul producătorului)

Precizări Linux

- driverul va fi accesat ca un device driver de tip caracter, cu funcţionări diferite în funcţie de parametrii transmişi modulului la încărcare:
  - parametrul `major` va specifica majorul cu care trebuie înregistrat dispozitivul;
  - parametrul `option` va specifica modul de funcţionare:
    - OPTION_BOTH: va înregistra şi COM1 şi COM2, cu majorul dat de parametrul `major` şi minorii 0 (pentru COM1) şi 1 (pentru COM2);
    - OPTION_COM1: va înregistra doar COM1, cu majorul `major` şi minorul 0;
    - OPTION_COM2: va înregistra doar COM2, cu majorul `major` şi minorul 1;
  - pentru a afla cum se pot pasa parametri în Linux, consultaţi Laboratorul 2 sau linkul de aici
  - **valorile implicite** sunt `major=42` şi `option=OPTION_BOTH`.
- header-ul cu definiţiile necesare pentru operaţiile speciale îl găsiţi 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 diferenţă este că trebuie să folosiţi două buffere, unul pentru read şi altul pentru write;
- pentru citirea/scrierea datelor în/din porturi nu trebuie să folosiţi funcţii amânabile (puteţi să faceţi totul din întrerupere);
- va trebui să sincronizaţi rutinele de read/write cu rutina de tratare a întreruperii pentru ca rutinele sa fie blocante; este recomandat să folosiţi sincronizare-cozi-de-aşteptare;
- pentru ca tema să poată funcţiona este nevoie ca driver-ul implicit `serial` să fie dezactivat:
  - `cat /proc/ioports | grep serial` va detecta prezenţa driver-ului implicit pe regiunile unde sunt definite COM1 şi COM2;
  - pentru a-l dezactiva trebuie recompilat kernel-ul, fie punând driver-ul `serial` ca modul, fie dezactivându-l cu totul (pe maşina virtuală este deja făcuta această modificare);
    - `Device Drivers ?> Character devices ?> Serial driver ?> 8250/16550 and compatible serial support`.
- puteţi folosi kfifo pentru buffere.

Precizări Windows

- driverul va fi accesat din user-space cu `\\.\uart0` pentru `COM1` şi `\\.\uart1` pentru `COM2`. Va exista un parametru al

modulului, care va specifica dacă se vor folosi ambele seriale sau doar câte una (similar Linux). Citirea parametrilor se va face din regiştri cu funcţia RtlQueryRegistryValues, ca în exemplul de aici;
- header-ul cu definiţiile necesare pentru operaţiile speciale îl găsiţi aici;
- pentru a putea lucra fără probleme cu porturile seriale, trebuie să dezinstalaţi ACPI-ul şi să faceţi disable pe `COM1` şi `COM2` (~~maşina virtuală pusă la dispoziţie este configurată corespunzător~~);
- o să aveţi nevoie de două buffere interne pentru fiecare port în care să puneţi datele (un buffer de read şi un buffer de write); ca să nu vă complicaţi cu sincronizări între ISR şi rutina de read folosiţi funcţiile `Interlocked?`;
- un bun punct de pornire în implementarea rutinelor de read/write este exemplul de driver de conversie upper-case de la laboratorul 5;
- pentru write, cel mai simplu e să ţineţi minte într-o listă IRP-urile şi să le serviţi atunci când vă indică întreruperea; cum nu aveţi voie să utilizaţi `IoCompleteRequest` într-o ISR, va trebui să folosiţi DPC-uri;
- puteţi apela `IoCompleteRequest` direct din rutina DPC sau vă puteţi 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 şi Windows). Testele presupun că numele modulului de kernel este `uart16550` atât pe Linux cât şi pe Windows.

Depunctări pentru probleme constatate in implementarea temei:

- -0.5 funcţionalitate incompletă
- -0.3 utilizare incorectă a device-ului
- -0.1 lock şi posibilitate de pierdere unlock
- -0.1 down şi posibilitate de pierdere up
- -0.1 disable şi posibilitate de pierdere enable
- -0.1 utilizare funcţii la niveluri IRQL necorespunzătoare
- -0.0 observaţii

- -1 pentru fiecare test pe linux
- -0.1 pentru fiecare test pe windows

De asemenea, consultaţi sfaturile şi depunctările generale.

În cazuri excepţionale (tema trece testele prin nerespectarea cerinţelor) se poate scădea mai mult decât este menţionat mai sus.

Întrebări

Pentru întrebări legate de tema puteţi consulta arhivele listei de discuţii sau puteţi trimite un e-mail (trebuie să fiţi înregistraţi).

Înainte să puneţi o întrebare verificaţi că:

1. aţi citit bine enunţul temei
2. nu este deja prezentată întrebarea pe pagina de FAQ
3. nu se poate găsi răspunsul în arhivele listei de discuţii