Universidade Estadual do Rio Grande do Sul

Software de Entrada e Saída

Prof. Celso Maciel da Costa
celsocostars@gmail.com

Software de E/S

- Uma das principais funções do SO é controlar os dispositivos de E/S
  - Para isso deve:
    - emitir comandos para os dispositivos
    - tratar interrupções
    - tratar erros
    - prover uma interface entre os dispositivos e o resto do sistema

Software de E/S

- Estruturação do software de E/S
  - quatro níveis
    - tratamento de interrupções
    - drivers dos dispositivos
    - software do SO independente do dispositivo
    - software ao nível do usuário

Software de E/S

- Tratamento de interrupções
  - quando uma interrupção ocorre, uma procedure específica é ativada de modo a desbloquear o driver
  - o efeito de uma interrupção é que um processo que estava previamente bloqueado esta agora apto a rodar

Drivers dos dispositivos

- Drivers dos dispositivos
  - todo código dependente do dispositivo esta em um driver do dispositivo
  - o trabalho do driver é aceitar requisições e providenciar para que sejam executadas
  - se o driver esta disponível, a requisição é processada imediatamente, caso contrário, entrará em uma fila de requisições

Drivers dos dispositivos

- Execução de uma requisição para um disco
  - determinar o posicionamento do braço no cilindro próprio
  - determinar as operações que são requeridas:
    - Ex. leitura, escrita, etc.
  - escrever as operações nos registradores da controladora

Software independente do dispositivo

- O Linux implementa o conceito de independência do dispositivo
- por exemplo, o comando

  write(f, &b, nbytes);

  que escreve no arquivo ***f*** , ***nbytes,*** armazenados no endereço ***&b***, seja executado para o arquivo ***f*** armazenado em um disco ou em um CD , disquete, etc.

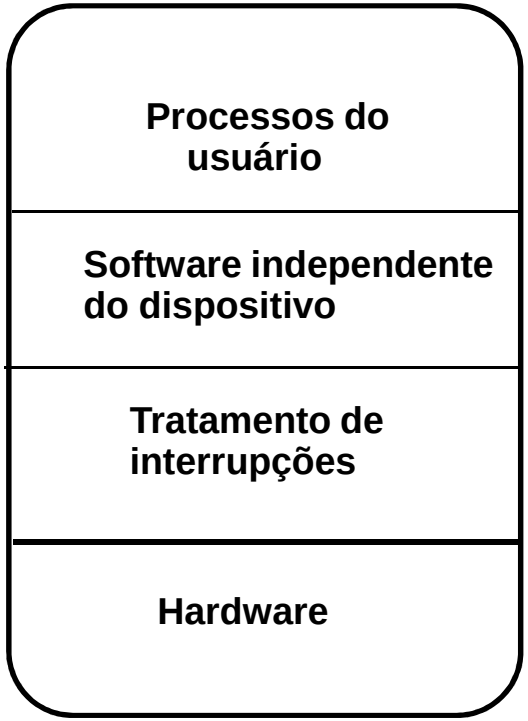
Software independente do dispositivo

- Software independente do dispositivo
  - Funções básicas
    - desenvolver as funções de E/S que são independentes do dispositivo
    - prover interface uniforme para o nível usuário
  - Funções:
    - mapeamento do nome simbólico no driver apropriado
    - interface uniforme para os drivers
    - identificação dos dispositivos
    - prover tamanho de bloco independente do dispositivo
    - proteção de acesso
    - alocação e liberação de buffers
    - tratamento de erros etc.

Software à nível do usuário

- Software à nível do usuário
  - Rotinas de biblioteca
    - read
    - write
    - open
    - close
    - etc.

Organização de um Sistema de E/S



Níveis lógicos de um sistema de entrada e saída

Exemplo de requisição

- Exemplo de requisição para um dispositivo blocado
  - Tipo da operação (read, write)
  - dispositivo: menor número
  - posição: bloco no dispositivo
  - número do processo requisitante da operação
  - Endereço dos dados no processo requisitante
  - contador: bytes a transferir
- Resposta
  - número do processo requisitante
  - status: número de bytes transferidos ou erro

Exemplo de requisição

Estrutura principal de um driver

```
message msg;  /* buffer de msg */

IO_task ()
  {
    int r, caller ;
    initialize ;
    while (true) {
          RECEIVE(any, &msg) ;
          switch (msg.type) {
                case read: r = do_read(); break ;
                case write: r = do_write()); break;
                case other: r = do_other(); break;
                default: r = error;
                }
          }
   }
```

Entrada e saída no Linux

- A nível usuário, a comunicação do programa de aplicação e o subsistema de entrada e saída é feita pelas chamadas de sistema relacionadas às operações de entrada e saída.

Entrada e saída no Linux

- As chamadas de sistema referentes as operações de entrada e saída são quase que exclusivamente tratadas pelo VFS (Virtual File System).

Entrada e saída no Linux

- O Linux implementa o conceito de independência do dispositivo:
  - A primitiva write(f, &b, nbytes) escreve no arquivo f nbyte, armazenados no endereço &b
  - O arquivo f pode estar armazenado em um disco ou em um pendrive.

Entrada e saída no Linux

- Software independente de dispositivo:
  - denominação do arquivo;
  - proteção de acesso;
  - alocação e liberação de buffers;
  - tratamento de erros.
- Driver:
  - operações de mais baixo nível, que dependem do tipo de periférico.

Entrada e saída no Linux

- Dois tipos de periféricos:
  - Orientados a caractere;
  - Orientados a bloco.
- Cada dispositivo é identificado por um maior número e um menor número. O maior número identifica o tipo de periférico e o menor número é usado para acessar o periférico específico, no caso de um disco, um inteiro entre 0 e 255

Entrada e saída no Linux

- Para um driver ser utilizado, necessita ser registrado e inicializado.

- Registrar um driver significa fazer a ligação com os arquivos necessários a sua execução. Na inicialização, são alocados os recursos necessários a execução do driver.

Entrada e saída no Linux

- Um driver pode ser compilado no kernel, neste caso é registrado quando o kernel executa as rotinas de inicialização.

- Pode ser compilado como um módulo do kernel. Neste caso, sua inicialização é realizada quando o módulo é carregado.

Entrada e saída no Linux

- O Linux identifica o final de uma operação de entrada e saída através de uma interrupção ou fazendo pooling.

Exemplo de driver de impressora no Linux (pooling)

```
void imp (char c, int lp) {
    while (!READY_STATUS(status) && count > 0)
        { count - - ; }
    if (count == 0) return (TIMEOUT) ;
    out (char, lp) ;
}
```