Insper

Sistemas Hardware-Software

Aula 20– Sinais POSIX 1

2017 – Engenharia

Igor Montagner, Fábio Ayres

Aulas passadas

- SO alterna quais processos são rodados em cada CPU
  - Pode ocorrer por tempo limite
  - Ou por conta de operações de entrada/saída

- Chamadas de sistema `fork` e `exec`
  - Isolamento de memória: cada processo tem sua cópia das variáveis do programa

Hoje

- Revisão de fork e exec
- Sinais
- `wait e waitpid`

Criação de processos

Criamos processos usando a chamada de sistema *fork*

```
pid_t fork();
```

O fork cria um clone do processo atual e retorna duas vezes

No processo original (pai) `fork` retorna o pid do filho

O pid do pai é obtido chamando

```
pid_t getpid();
```

No processo filho `fork` retorna o valor 0. O pid do filho é obtido usando

```
pid_t getpid();
```

O pid do pai pode ser obtido usando a chamada

```
pid_t getppid();
```

Syscall exec

Exercício: Por que o código abaixo (`exemplo2-exec`) não imprime "Fim do exec!" após executar o programa "ls"?

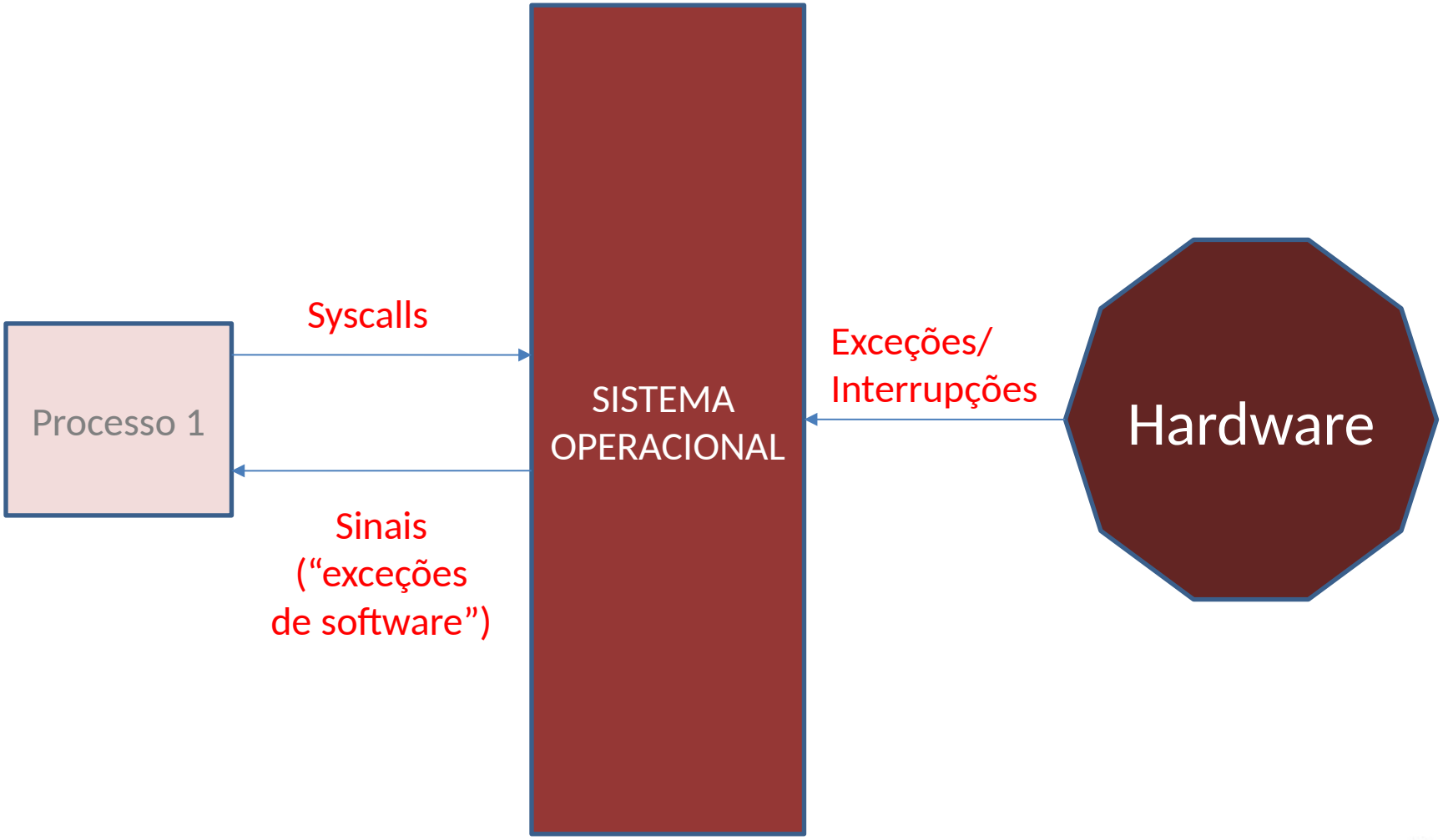
```c
#include <unistd.h>
#include <stdio.h>

int main() {
    char prog[] = "ls";
    // a lista de argumentos sempre começa com o nome do
    // programa e termina com NULL
    char *args[] = {"ls", "-l", "-a", NULL};

    execvp(prog, args);
    printf("Fim do exec!\n");

    return 0;
}
```

Interação do SO com seus processos



Insper

(Alguns) Sinais POSIX

Signal	Default Action	Description
SIGABRT	Terminate (core dump)	Process abort signal
SIGALRM	Terminate	Alarm clock
SIGCHLD	Ignore	Child process terminated, stopped, or continued.
SIGFPE	Terminate (core dump)	Erroneous arithmetic operation.
SIGILL	Terminate (core dump)	Illegal instruction.
SIGINT	Terminate	Terminal interrupt signal. (Ctrl+C)
SIGKILL	Terminate	Kill (cannot be caught or ignored).
SIGTERM	Terminate	Termination signal.

Exemplos de usos de sinais

- Ctrl+C envia um sinal SIGINT para o processo.
    - Ele pode ser capturado e fazer com que o programa feche conexões e arquivos abertos, por exemplo.

- O sinal SIGSTOP (SIGTSTP) é usado para deixar um processo em background. Ele fica parado até ser resumido por SIGCONT

- O sinal SIGKILL interrompe um processo imediatamente. Ele não pode ser ignorado.

Enviando um sinal

- Kernel detectou um evento de sistema, tal como uma divisão-por-zero (SIGFPE) ou término de um processo filho (SIGCHLD)

- Outro processo invocou a chamada de sistema **`kill`** para explicitamente pedir ao kernel que envie um sinal ao processo destinatário.

Recebendo um sinal

O kernel força o processo destinatário a reagir de alguma forma à entrega do sinal. O destinatário pode:

- Ignorar o sinal (não faz nada)
- Terminar o processo (opcional: core dump)
- Capturar o sinal e executar, como usuário, um signal handler (próxima aula!)

Agora

- Handout de hoje: espera pelo término de processos usando wait e waitpid
- Envio de sinais para outros processos

Insper

www.insper.edu.br