FSO 16-09-2018

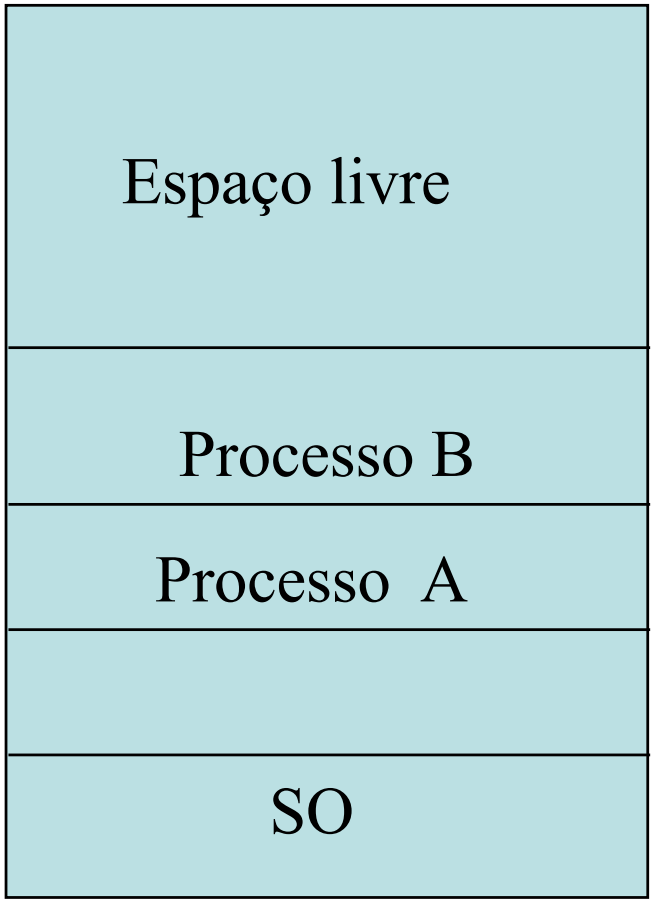
Aula 03

Organização geral de um SO – execução direta controlada

- capítulos 2 e 6 do livro OSTEP

Um sistema multi-programado executando vários processos

Na RAM estão carregados vários processos



Mecanismos Hardware Indispensáveis ao SO

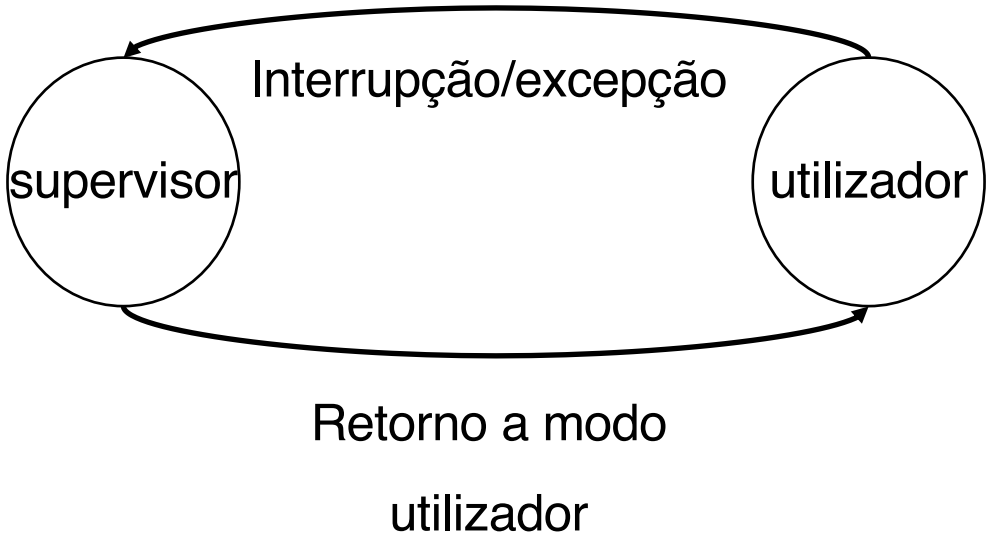
- Modo utilizador/supervisor e interrupções
- Protecção do espaço de I/O
- Protecção de memória
- Protecção do CPU

Dois modos de operação

- A partilha de recursos exige que um programa incorrecto não prejudique outros programas/utilizadores do sistema
- O hardware tem de suportar pelo menos dois modos de operação do CPU.
  1. *Modo utilizador* – CPU está a executar por conta de um utilizador.
  2. *Modo supervisor* (*kernel mode* or *system mode*) – execução de código do SO

Modo utilizador/supervisor

- *bit de modo* indica o modo corrente: supervisor (0) ou utilizador (1).
- Quando ocorre uma interrupção hardware ou uma excepção o CPU muda para modo supervisor.



Instruções privilegiadas só podem ser executadas em modo supervidor.

Protecção de I/O

- Todas as instruções de I/O são privilegiadas.
- É preciso garantir que um programa utilizador nunca execute o seu código em modo supervisor (podia por exemplo alterar o vector de interrupções).

SO como “interrupt handler”

- Interrupções hardware

- Serviços do sistema invocados através de interrupções de software.
  - Instrução máquina *int XX* no Pentium

Interacção CPU-Periféricos

- Os periféricos e o CPU I/O executam em paralelo
- Cada controlador supervisiona um tipo de periférico.
- Cada controlador tem um “buffer”.
- O CPU transfere dados de/para a RAM para/de os buffers dos controladores
- A acção de I/O faz-se entre o buffer do controlador e o periférico.
- O controlador informa o CPU que acabou a transferência através de uma *interrupção*.

O SO é “interrupt-driven”

- Quando ocorre uma interrupção o controlo é transferido para uma *rotina de serviço*, a maior parte das vezes através do *vector de interrupções*.
- A execução continua no ponto onde se encontrava quando a interrupção ocorre
- Um *trap* (excepção) é uma interrupção associada à execução da instrução corrente (quer por erro quer por vontade deliberada)

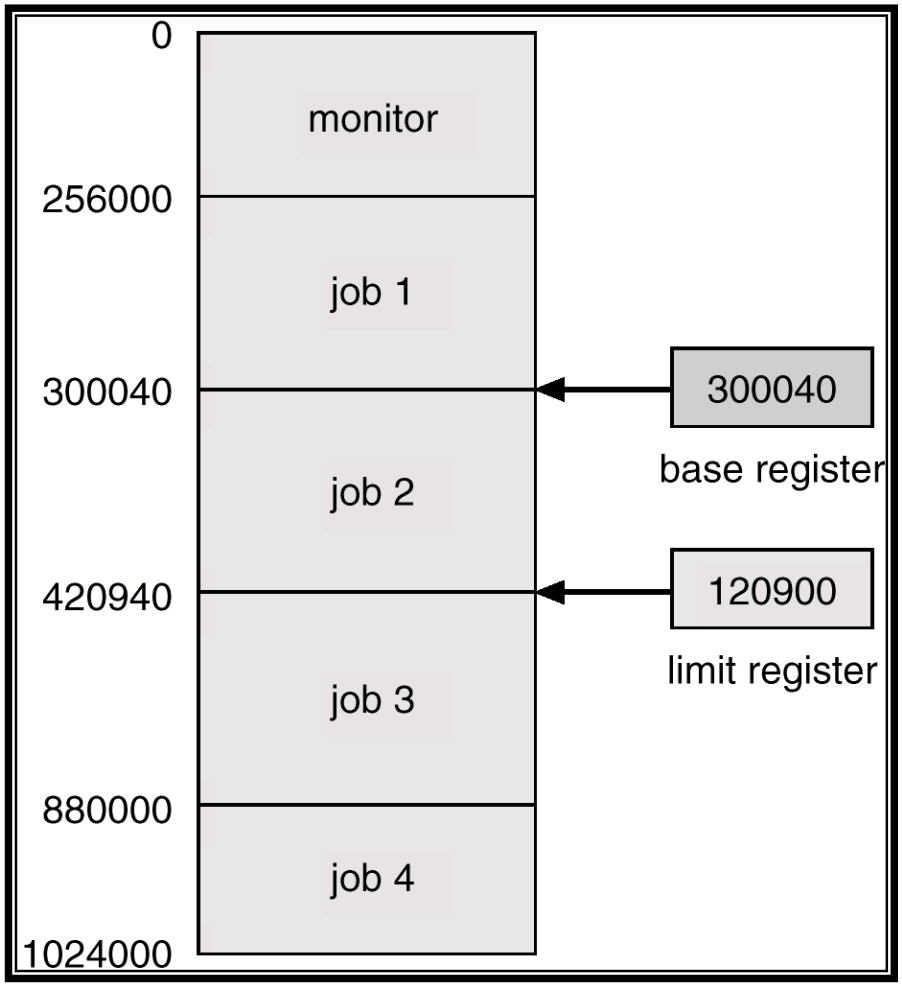
Tratamento de Interrupções

- O estado do CPU (registos+PC) é salvo.
- Determina-se que tipo de interrupção ocorreu:
  - *Polling*
  - *Interrupções vectorizadas*
- Cada tipo de interrupção provoca a chamada de um conjunto de código diferente

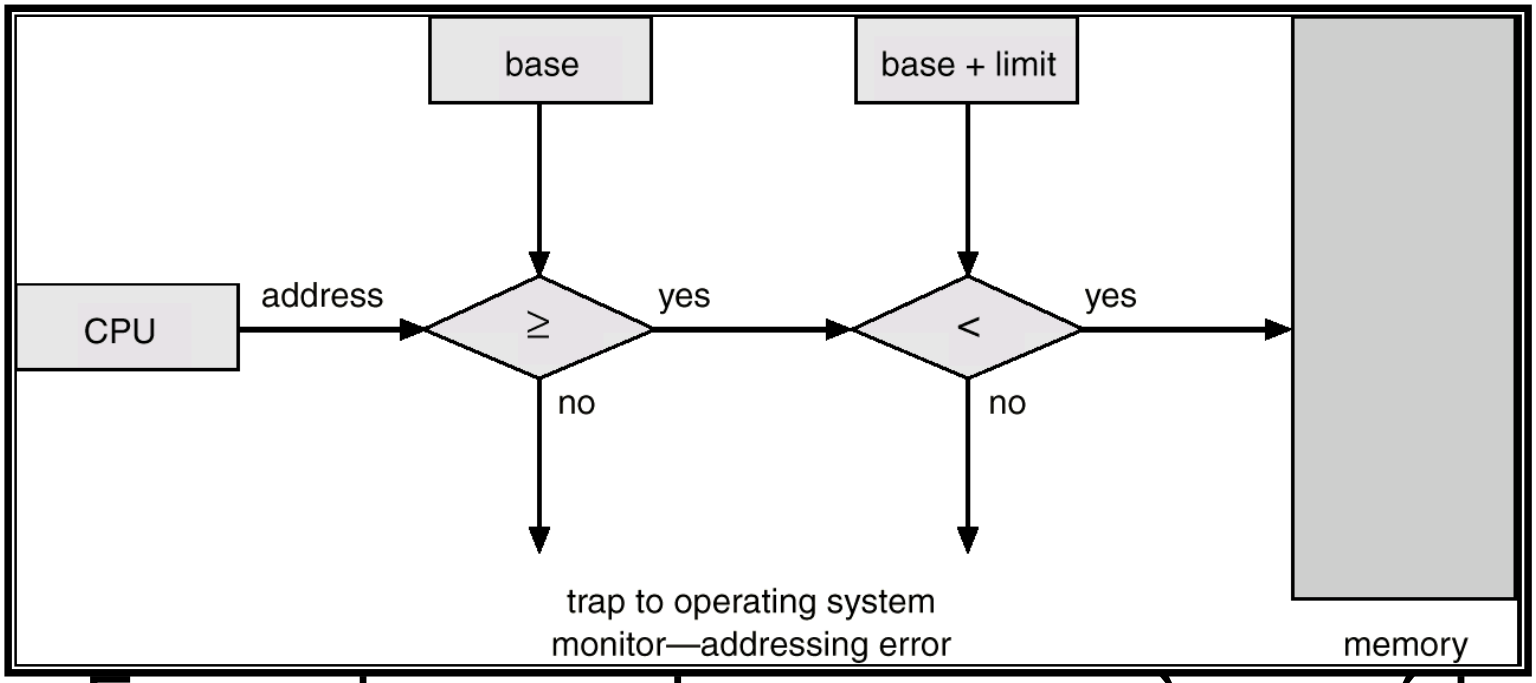
Protecção de Memória

- É necessário pelo menos proteger o vector de interrupções e as rotinas de serviço
- Nos sistemas modernos a gestão da memória física é baseada em páginas
- No modelo mais primitivo e mais básico, dois registos determinam a faixa legal de endereços:
  - **Registo Base**
  - **Registo Limite**
- A memória fora da zona é protegida.

Registos base e limite



Protecção de memória



- Em modo supervisor o acesso à memória é total.
- As instruções para alterar os registos *base* e *limite* são privilegiadas.

Protecção do CPU

- *Temporizador* – interrompe o processamento ao fim de um tempo pré-determinado para assegurar que o SO nunca perde o controlo
  - Decrementado a cada “tick”do relógio.
  - Qaundo chega a 0 lança uma interrupção.

.

- *Load-timer* é uma instrução privilegiada.

Suporte de uma chamada ao sistema

- Monolítico
  - Carrega os parâmetros em registos
  - Faz uma interrupção por software
  - O fluxo de execução entra pelo código de sistema
  - Não se comuta de processo