Sistemas Operativos

Sincronización por hardware

Curso 2021

Facultad de Ingeniería, UDELAR

Agenda

1. Introducción

2. Sincronización por hardware

Introducción

Sincronización por hardware

- Las soluciones eficientes al problema de la región crítica requieren asistencia de hardware y por lo tanto también del sistema operativo.
- La abstracción fundamental es el lock que protege las regiones críticas.
- Los procesos deben adquirir un lock antes de entrar a la región crítica y liberarlo al salir. A veces se les llama mutex.
- A lo largo del curso veremos varias formas de implementar esta primitiva básica.

Estructura de un proceso

- La estructura general de un proceso que usa una locks es:

```
loop
   Get                                  ▷ Obtengo lock
        Sección crítica;
   Release                               ▷ Libero lock
   Otras tareas;
end loop
```

Sincronización por hardware

Sistema monoprocesador

- En un sistema monoprocesador para asegurar el acceso con mutua exclusión a una región crítica alcanza con deshabilitar las interrupciones
- Pero si la región crítica dura mucho tiempo se pueden perder interrupciones

```
loop
   CLI                                  ▷ Obtengo lock
        Sección crítica;
   STI                                   ▷ Libero lock
   Otras tareas;
end loop
```

Sistema multiprocesador

- En un multiprocesador no es suficiente con deshabilitar las interrupciones en todos los procesadores
- Para no tener que hacer esto el hardware proporciona instrucciones de sincronización que se pueden usar para implementar locks.

Test and set

- Permite chequear el contenido de una variable global y modificarlo en forma atómica (como si fuera una región crítica)

```
function TestAndSet(var)
   ret := var;
   var := True;
   return ret;
end function
```

Región crítica con test and set

- La solución a la región crítica es más sencilla que Dekker o Peterson pero igual requiere busy waiting.

```
lock := False;  ▷ inicialización (una vez)

repeat
   while TestAndSet(lock) do
   end while
   Región crítica;
   lock := False;
   Otras tareas;
until False
```

Swap

- Intercambia el valor de dos variables en forma atómica.

```
procedure Swap(a, b)
    tmp := a;
    a := b;
    b := tmp;
end procedure
```

Región crítica con swap

- La solución a la región crítica con swap también requiere busy waiting.

```
lock := False;  ▷ inicialización (una vez)

repeat
   key := True;                ▷ variable local
   while key do
      Swap(lock, key);
   end while
   Región crítica;
   lock := False;
   Otras tareas;
until False
```