Introducción Implementación de MMU Recomendaciones útiles

Paginación - MMU Organización del Computador II

Facundo Linari

¹Departamento de Computación Facultad de Ciencias Exactas y Naturales

22/10/2019

¿Qué hicimos hasta ahora?

- Armamos un esquema de paginación con identity mapping.
 - ► Rango: 0x00000000 a 0x003FFFFF (Primeros 4Mb)
- ¿Cómo podemos comprobarlo?
 - ▶ info tab

¿Qué vamos a hacer?

Implementar una limitada Memory Management Unit

- Funciones para obtener una página de memoria física libre para utilizar.
- 2. Funciones para mappear y desmappear páginas.
- Funciones para construir los esquemas de paginación de las tareas.
 - Cada tarea tiene su page directory.

Obtención de páginas físicas libres

- Cómo podemos administrar qué páginas físicas tenemos libres de manera sencilla?
 - Vamos a tener un entero para cada área que guarde la dirección de una página física libre.
 - Cada vez que querramos una nueva página física libre, la obtenemos desde el entero correspondiente y luego incrementamos al mismo.
 - Necesitamos las funciones para pedir las páginas y 2 enteros globales.

Implementación

```
unsigned int proxima_pagina_libre_kernel;
unsigned int proxima_pagina_libre_task;
void mmu_init() {
  proxima_pagina_libre_kernel = INICIO_PAGINAS_LIBRES_KERNI
  proxima_pagina_libre_task = INICIO_PAGINAS_LIBRES_TASK;
unsigned int mmu_nextFreeTaskPage() {
  unsigned int pagina_libre = proxima_pagina_libre_task;
  proxima_pagina_libre_task += PAGE_SIZE;
  return pagina_libre;
```

Mappeo de páginas

Tenemos que hacer una función que mappee páginas de una dirección virtual a otra fisica.

- Necesitamos:
 - La dirección en donde se encuentra el page directory donde vamos a hacer el mappeo.
 - La dirección física.
- Esto se implementa en:

Procedimiento para mappear una página

- 1. Descomponemos la dirección virtual en índice del page directory y en índice del page table.
- Utilizar la dirección del page directory y el índice del page directory para encontrar el page directory entry asociado al índice.
 - Hay que chequar que la pagle table a la que referencia el pde exista.
 - Si no existe, hay que crearla y settear correctamente (bits de propiedades) la **pde** para que pueda ser accedida posteriormente.
- Utilizar el índice del page table para obtener la page table entry correspondiente.
- 4. Completar la pte para que referencie a la dirección física.
- 5. Ejecutar tlbflush() para invalidar la cache de traducciones.

Desmappeo de páginas

Tenemos que hacer una función que desmappee una dirección virtual.

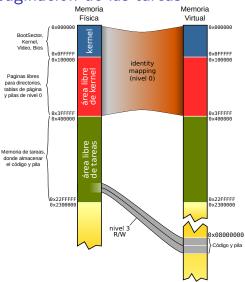
Necesitamos:

- La dirección virtual.
- La dirección en donde se encuentra el page directory donde vamos a hacer el desmappeo.
- Esto se implementa en:

Procedimiento para desmappear una página

- 1. Descomponemos la dirección virtual en índice del page directory y en índice del page table.
- Utilizar la dirección del page directory y el índice del page directory para encontrar el page directory entry asociado al índice.
- Utilizar el índice del page table para obtener la page table entry correspondiente.
- 4. Establecer el bit de presente en 0.

Esquema de paginación de las tareas



Construcción de esquema de paginación de las tareas

- Necesitamos pedir una página física libre para el directorio de páginas.
- 2. Tenemos que armar un esquema de **indentity mapping** de los primeros 4Mb.
 - Los **pde** y **pte** deben estar setteados como supervisor.
- 3. Copiar el código de la tarea a su posición en el mapa.
- 4. Mappear la dirección virtual del código de la tarea a su posición física donde fue copiada.
- 5. Mappear las demás páginas que pertenezcan a la tarea según el enunciado.
 - Los pde y pte (código y páginas de la tarea) deben estar setteados como user.

Alguna recomendaciones útiles

- Pueden modificar las funciones anteriores parametrizando distintos atributos a settear.
- Pueden comprobar que el mappeo es correcto setteando el CR3 actual con el page directory que crearon.
- Utilicen info tab para ver si el mappeo es correcto.
- Macros útiles:

```
#define PDE_INDEX(virtual) virtual >> 22
#define PTE_INDEX(virtual) ???
#define PG READ_WRITE ???
#define PG USER ???
#define PG PRESENT 0x00000001
```