Paginación MMU

Programación de Sistemas Operativos

David Alejandro González Márquez

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Armar un limitado MMU

¿Quien administra la memoria que usamos para administrar la memoria?

Armar un limitado MMU

¿Quien administra la memoria que usamos para administrar la memoria?

En el contexto del TP este problema lo solucionaremos de forma limitada:

Armar un limitado MMU

¿Quien administra la memoria que usamos para administrar la memoria?

En el contexto del TP este problema lo solucionaremos de forma limitada:

- Debemos contruir funciones que nos permitan pedir memoria.

Armar un limitado MMU

¿Quien administra la memoria que usamos para administrar la memoria?

En el contexto del TP este problema lo solucionaremos de forma limitada:

- Debemos contruir funciones que nos permitan pedir memoria.
- Nos limitaremos a pedir memoria, pero nunca liberarla.

Armar un limitado MMU

¿Quien administra la memoria que usamos para administrar la memoria?

En el contexto del TP este problema lo solucionaremos de forma limitada:

- Debemos contruir funciones que nos permitan pedir memoria.
- Nos limitaremos a pedir memoria, pero nunca liberarla.
- Nuestro sistema perderá memoria continuamente hasta que colapse.

Armar un limitado MMU

¿Quien administra la memoria que usamos para administrar la memoria?

En el contexto del TP este problema lo solucionaremos de forma limitada:

- Debemos contruir funciones que nos permitan pedir memoria.
- Nos limitaremos a pedir memoria, pero nunca liberarla.
- Nuestro sistema perderá memoria continuamente hasta que colapse.

Solución:

```
unsigned int proxima_pagina_libre;

void mmu_init() {
    proxima_pagina_libre = INICIO_DE_PAGINAS_LIBRES;
}

unsigned int mmu_nextFreeTaskPage() {
    unsigned int pagina_libre = proxima_pagina_libre;
    proxima_pagina_libre += PAGE_SIZE;
    return pagina_libre;
}
```

Mapear y desmapear paginas

Para administrar la memoria de una tarea debemos construir funciones que nos permitan mapear y desmapear páginas sobre un esquema de paginación.

Mapear y desmapear paginas

Para administrar la memoria de una tarea debemos construir funciones que nos permitan mapear y desmapear páginas sobre un esquema de paginación.

- `void mmu_mapPage(uint32_t cr3, uint32_t virtual, uint32_t phy)`
  Mapea en el esquema de paginación dado por `cr3`, la dirección `virtual` a la dirección física phy.

Mapear y desmapear paginas

Para administrar la memoria de una tarea debemos construir funciones que nos permitan mapear y desmapear páginas sobre un esquema de paginación.

- `void mmu_mapPage(uint32_t cr3, uint32_t virtual, uint32_t phy)`
  Mapea en el esquema de paginación dado por `cr3`, la dirección `virtual` a la dirección física phy.
- `void mmu_unmapPage(uint32_t cr3, uint32_t virtual)`
  Desmapea en el esquema de paginación dado por `cr3`, la dirección `virtual`.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.
4. Obtener la page `table` de la PDE.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.
4. Obtener la page `table` de la PDE.
5. Usando el puntero al comienzo de la page `table` y el campo `tableIdx` obtener la PTE.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.
4. Obtener la page `table` de la PDE.
5. Usando el puntero al comienzo de la page `table` y el campo `tableIdx` obtener la PTE.
6. Completar la PTE con el marco de página que se busca mapear.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.
4. Obtener la page `table` de la PDE.
5. Usando el puntero al comienzo de la page `table` y el campo `tableIdx` obtener la PTE.
6. Completar la PTE con el marco de página que se busca mapear.
7. Completar los atributos en la PDE y PTE.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.
4. Obtener la page `table` de la PDE.
5. Usando el puntero al comienzo de la page `table` y el campo `tableIdx` obtener la PTE.
6. Completar la PTE con el marco de página que se busca mapear.
7. Completar los atributos en la PDE y PTE.
8. Llamar a la función `tlbflush()`.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.
4. Obtener la page `table` de la PDE.
5. Usando el puntero al comienzo de la page `table` y el campo `tableIdx` obtener la PTE.
6. Completar la PTE con el marco de página que se busca mapear.
7. Completar los atributos en la PDE y PTE.
8. Llamar a la función `tlbflush()`.

Considerar que los atributos en la PDE y PTE deben ser un parámetro de la función.

Pasos para mapear una página

1. Dividir la dirección a mapear en `directoryIdx`, `tableIdx` y `offset`.
2. Usando el parámetro `cr3`, calcular la dirección de la PDE.
3. Si el bit de `present` de la PDE es `0`. Entonces pedir una nueva página para la page `table`, completarla con ceros y completar la PDE.
4. Obtener la page `table` de la PDE.
5. Usando el puntero al comienzo de la page `table` y el campo `tableIdx` obtener la PTE.
6. Completar la PTE con el marco de página que se busca mapear.
7. Completar los atributos en la PDE y PTE.
8. Llamar a la función `tlbflush()`.

Considerar que los atributos en la PDE y PTE deben ser un parámetro de la función.

La función `tlbflush()` invalida todas las entras de la TLB (`translation lookaside buffer`)

Seudocódigo para mapear una página

```
mmu_mapPage(cr3, virtual, phy)

    directoryIdx = virtual >> 22
    tableIdx = (virtual >> 12) & 0x3FF

    PDE = cr3[directoryIdx]

    if (PDE.present != 1):
            newPT = mmu_nextFreeKernelPage()

            for(i = 0; i < 1024; ++i)
                    newPT[i] = 0

            cr3[directoryIdx] = newPT | PAG_US | PAG_RW | PAG_P

    PT = (cr3[directoryIdx] &~ 0xFFF)
    PT[tableIdx] = (phy &~ 0xFFF) | PAG_US | PAG_RW | PAG_P
    tlbflush()
```

Tareas

- Las tareas utilizarán un esquema similar al del kernel.
- Tendrán además mapeado el código de la tarea.
- Este código deberá ser copiado y mapeado.

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.
3. Construir un esquema de paginación con Identity Mapping para los primeros 4MB.

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.
3. Construir un esquema de paginación con Identity Mapping para los primeros 4MB.
4. Identificar el código de la tarea que debe ser copiado desde el kernel (src).

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.
3. Construir un esquema de paginación con Identity Mapping para los primeros 4MB.
4. Identificar el código de la tarea que debe ser copiado desde el kernel (src).
5. Identificar la posición de memoria donde copiar el código (dst).

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.
3. Construir un esquema de paginación con Identity Mapping para los primeros 4MB.
4. Identificar el código de la tarea que debe ser copiado desde el kernel (src).
5. Identificar la posición de memoria donde copiar el código (dst).
6. Mapear de ser necesario la posición destino o fuente del código.

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.
3. Construir un esquema de paginación con Identity Mapping para los primeros 4MB.
4. Identificar el código de la tarea que debe ser copiado desde el kernel (src).
5. Identificar la posición de memoria donde copiar el código (dst).
6. Mapear de ser necesario la posición destino o fuente del código.
7. Copiar las dos paginas de la tarea.

Pasos para construir el esquema de paginación de una tarea

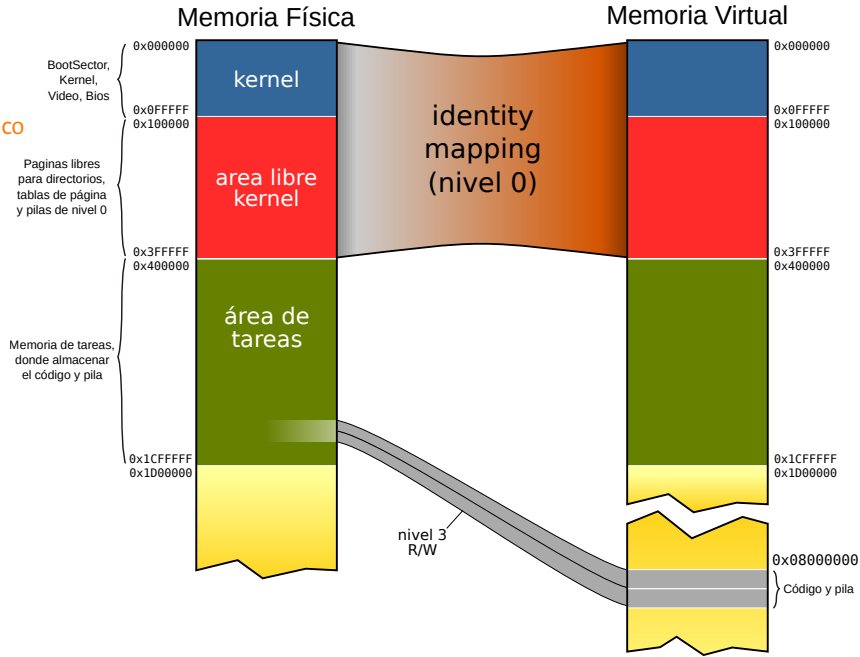
1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.
3. Construir un esquema de paginación con Identity Mapping para los primeros 4MB.
4. Identificar el código de la tarea que debe ser copiado desde el kernel (src).
5. Identificar la posición de memoria donde copiar el código (dst).
6. Mapear de ser necesario la posición destino o fuente del código.
7. Copiar las dos paginas de la tarea.
8. Mapear la tarea copiada, al nuevo esquema de paginación que se esta construyendo.

Pasos para construir el esquema de paginación de una tarea

1. Solicitar una pagina libre para el PD.
2. Solicitar una pagina libre para el PT.
3. Construir un esquema de paginación con Identity Mapping para los primeros 4MB.
4. Identificar el código de la tarea que debe ser copiado desde el kernel (src).
5. Identificar la posición de memoria donde copiar el código (dst).
6. Mapear de ser necesario la posición destino o fuente del código.
7. Copiar las dos paginas de la tarea.
8. Mapear la tarea copiada, al nuevo esquema de paginación que se esta construyendo.
9. Desmapear de ser necesario las paginas mapeadas para poder copiar.

Paginación en una tarea

Ejemplo Trabajo Práctico



Bibliografía: Fuentes y material adicional

- Convenciones de llamados a función en x86:
  `https://en.wikipedia.org/wiki/X86_calling_conventions`
- Notas sobre System V ABI:
  `https://wiki.osdev.org/System_V_ABI`
- Documentación de NASM:
  `https://nasm.us/doc/`
- Artículo sobre el flag `-pie`:
  `https://eklitzke.org/position-independent-executables`
- Documentación de System V ABI:
  `https://uclibc.org/docs/psABI-x86_64.pdf`
- Manuales de Intel:
  `https://software.intel.com/en-us/articles/intel-sdm`

¡Gracias!

Recuerden leer los comentarios al final de este video por aclaraciones o fe de erratas.