Lezione 8

Gestione della memoria centrale

Sistemi operativi

8 maggio 2012

Marco Cesati

System Programming Research Group
Università degli Studi di Roma Tor Vergata

Di cosa parliamo in questa lezione?

La gestione della memoria centrale

1. Schemi di indirizzamento della memoria centrale
2. La segmentazione
3. La paginazione
4. Linux su Intel IA-32

La memoria centrale

La memoria centrale è un componente essenziale per un calcolatore elettronico:

- È costituita dai chip di memoria dinamica (RAM)
- Contiene il codice ed i dati dei processi attivi nel sistema
- Contiene tutte le strutture di dati del nucleo del SO

La memoria centrale è una risorsa condivisa disponibile in quantità limitata

- Deve essere gestita e amministrata dal SO

La modalità con cui un SO gestisce la memoria centrale dipende essenzialmente dalle caratteristiche dell'architettura hardware del calcolatore

Gran parte delle architetture hardware moderne utilizza lo stesso schema di base per la memoria centrale

Architettura della memoria centrale

Memory Management Unit (MMU)

Componente hardware del calcolatore elettronico che ha il compito di arbitrare, gestire e realizzare i trasferimenti dei dati da e verso la memoria primaria

Poiché la memoria primaria è costituita una gerarchia di livelli di memoria, la MMU deve contenere circuiti per

- controllare il funzionamento della memoria statica (cache)
- gestire le transazioni da e verso la memoria centrale (memoria dinamica, RAM)

Inoltre la MMU offre meccanismi essenziali per la realizzazione di SO multiprogrammati:

- astrazione degli spazi di indirizzamento dei processi
- protezione degli accessi alle regioni di memoria
- memoria virtuale

Indirizzi logici, fisici e di I/O

Si consideri una determinata cella di memoria: ad essa sono associati differenti tipi di indirizzi

Indirizzo logico

L'indirizzo che deve essere scritto in un registro della CPU o nel codice operativo di una istruzione macchina

Indirizzo fisico

L'indirizzo che deve essere posto sul bus di comunicazione tra CPU e RAM

Indirizzo di I/O

L'indirizzo che deve essere posto sul bus di sistema per avviare un trasferimento diretto tra RAM e periferica di I/O (DMA)

Indirizzi virtuali

- In tutte le architetture dei microprocessori moderni, gli indirizzi logici sono diversi dagli indirizzi fisici
- La trasformazione tra indirizzi logici e fisici è realizzata automaticamente dalla MMU

- L'associazione (mapping) tra indirizzi logici e fisici è in genere
  - dinamicamente modificabile
  - diversa da processo a processo
- In questi casi l'indirizzo logico è anche chiamato *indirizzo virtuale*

Spazi di indirizzamento

- Lo spazio di indirizzamento fisico della memoria centrale è costituito dall'insieme di tutti gli indirizzi fisici che potenzialmente identificano celle di memoria RAM
  - La dimensione dello spazio di indirizzamento fisico dipende dalla quantità di RAM installabile nel sistema
  - Non fa parte delle caratteristiche architetturali
- Lo spazio di indirizzamento logico (o virtuale) è costituito dall'insieme di tutti gli indirizzi logici che possono essere utilizzati come indentificatore di una cella di memoria in un registro della CPU oppure nel codice operativo di una istruzione macchina
  - È una caratteristica architetturale del calcolatore
  - Generalmente se un indirizzo logico è costituito da n bit, lo spazio di indirizzamento logico ha dimensione 2^n
  - La dimensione dell'indirizzo logico coincide con la dimensione dei registri della CPU (ad es., 32 o 64 bit)

Indirizzi logici e fisici possono avere dimensioni differenti? ***Sì!***

Vantaggi degli indirizzi virtuali

Due processi eseguono due programmi differenti:

Senza indirizzi virtuali	Con indirizzi virtuali
I due processi debbono utilizzare indirizzi logici differenti perché condividono lo spazio di indirizzamento logico	I due processi possono utilizzare gli stessi indirizzi logici, quindi compilazione e caricamento dei programmi sono più facili
È difficile impedire che un processo acceda ad un dato dell'altro processo	È facile realizzare meccanismi di protezione degli accessi
Il SO deve assegnare immediatamente a ciascun processo tutta la memoria necessaria per l'intera esecuzione	Il SO può creare dinamicamente l'associazione tra indirizzo virtuale e fisico e quindi ritardare l'assegnazione della memoria

Trasformazione di indirizzi virtuali in indirizzi fisici

- In linea di principio a ciascun indirizzo virtuale potrebbe essere associato un diverso indirizzo fisico
  - Questa associazione deve essere memorizzata dal SO utilizzando una parte della memoria centrale
  - Nel caso peggiore per ciascun indirizzo di memoria logico (o fisico) identificante un singolo byte si dovrebbero "sprecare" diversi byte di memoria per registrare il corrispondente indirizzo fisico (o logico)
- L'associazione tra i due tipi di indirizzo viene realizzata per gruppi di dimensione minima di centinaia o migliaia di byte
- Esistono sostanzialmente due schemi generali per realizzare questa associazione:
  - La segmentazione
  - La paginazione

Segmentazione della memoria

La *segmentazione* della memoria centrale è un meccanismo basato sul concetto di *segmento*

Segmento

Insieme di indirizzi logici contigui caratterizzato da:

- un indirizzo fisico di base
- una lunghezza (numero di celle/indirizzi nel segmento)
- attributi e diritti di accesso comuni alle celle del segmento

La memoria centrale occupata da un processo può essere suddivisa in modo naturale in diversi segmenti, ad esempio:

- Il segmento di codice contenente le istruzioni macchina da eseguire, con soli diritti di esecuzione
- Il segmento di dati contenente i dati del programma, con diritti di lettura e scrittura
- Il segmento di stack contenente lo stack User Mode
- Un segmento per lo heap (allocazione dinamica)
- Per ciascuna libreria, un segmento di codice ed uno di dati

Gli indirizzi logici con la segmentazione

Se l'architettura utilizza il meccanismo di segmentazione della memoria, il formato di un indirizzo logico è il seguente:

Segmento	Spiazzamento

- "Segmento" rappresenta un indice o altro identificatore per un determinato segmento di memoria
- "Spiazzamento" rappresenta la posizione (offset) della cella richiesta all'interno del segmento

Gli indirizzi logici con segmentazione non sono necessariamente semplici valori numerici

Nel linguaggio C le operazioni di differenza tra puntatori sono ben definite soltanto per puntatori allo stesso oggetto o vettore!

Esempio di segmentazione

Un processo esegue un programma costituito da 10 KiB di istruzioni macchina, 64 KiB di dati e 24 KiB di stack

Nel suo PCB è presente la seguente *tabella dei segmenti*:

# seg.	base	limite	flag	
0	`0x00001000`	10239	Exec	codice
1	`0x00004000`	65535	Read+Write	dati
2	`0x00100000`	24575	Read+Write	stack
3	`0x0001a000`	8191	Read+Write+Exec	heap
	⋮			

A cosa corrisponde l'indirizzo fisico `0x00002010` *?*

È nel segmento #0 all'offset `0x1010`, quindi corrisponde all'istruzione macchina all'indirizzo logico `<0; 4112>`

A cosa corrisponde l'indirizzo logico `<3; 12240>` *?*

Non è un indirizzo logico valido!

Svantaggi della segmentazione

Quali sono i principali svantaggi della segmentazione?

- A ciascun segmento deve corrispondere una zona contigua di memoria fisica
  - Il SO deve assegnare a ciascun processo tutta la memoria fisica necessaria per l'esecuzione, anche se il processo non la utilizzerà realmente
  - Possibile frammentazione: la memoria fisica ancora libera è suddivisa in tante porzioni ciascuna delle quali troppo piccola per contenere un segmento di un nuovo processo
- Lo spazio degli indirizzi logici di un processo non è "connesso" ed uniforme
  - L'indirizzo logico è suddiviso in due parti che debbono essere trattate, anche a livello di codice macchina, in modo differente

La paginazione

La *paginazione* della memoria centrale è un meccanismo basato sul concetto di *pagina*

Pagina (page)

Gruppo di indirizzi logici contigui di lunghezza prefissata con un insieme di attributi e diritti d'accesso comuni

Per estensione "pagina di memoria" indica anche un blocco di dati di lunghezza prefissata in memoria primaria o secondaria

Qual è la principale differenza con la segmentazione?

- I segmenti hanno lunghezza variabile mentre le pagine hanno lunghezza costante e prefissata
- La lunghezza prefissata permette di semplificare e rendere più efficienti i circuiti della MMU
  - La lunghezza è sempre una potenza di due!

Lunghezza di una pagina: tra 512 byte e alcuni megabyte

Gli indirizzi fisici con la paginazione

Si consideri una architettura con pagine di dimensione $N = 2^n$

- La memoria fisica è partizionata in blocchi di N indirizzi fisici chiamati *page frame* o *pagine fisiche*
  - In ciascuna page frame è memorizzabile il contenuto di una ed una sola pagina di memoria
- L'indirizzo fisico iniziale di ciascuna page frame è un multiplo di N:
  - Gli n bit meno significativi sono zero
  - I restanti bit sono chiamati numero di pagina fisica (PFN, Page Frame Number)
- Formato di un generico indirizzo fisico:

PFN	Spiazzamento

  - Ad esempio, se $N = 4096$, l'indirizzo fisico `0x00031104` corrisponde alla cella in posizione `0x104` $= 260$ della page frame `0x31` $= 49$

Gli indirizzi logici con la paginazione

A differenza della segmentazione, gli indirizzi logici utilizzati con la paginazione costituiscono uno spazio di indirizzamento connesso ed uniforme

Il formato dell'indirizzo logico è:

VPN	Spiazzamento

- Spiazzamento: posizione relativa della cella nella pagina
  - È costituito dagli n bit meno significativi
- VPN (Virtual Page Number): numero di indice della pagina
  - È costituito dai restanti bit più significativi

Il meccanismo di traduzione da indirizzi logici a indirizzi fisici trasforma un VPN in un PFN e lascia invariati i bit di spiazzamento

Esempi di traduzione con la paginazione

Esempio #1:

Dimensione pagina:	$N = 4096$ byte ($n = 12$ bit)
Dimensione degli indirizzi logici:	32 bit (4 GiB)
Dimensione degli indirizzi fisici:	30 bit (1 GiB)
Dimensione dei VPN:	$32 - 12 = 20$ bit
Dimensione dei PFN:	$30 - 12 = 18$ bit

Esempio #2:

Dimensione pagina:	$N = 4096$ byte ($n = 12$ bit)
Dimensione degli indirizzi logici:	32 bit (4 GiB)
Dimensione degli indirizzi fisici:	36 bit (64 GiB)
Dimensione dei VPN:	$32 - 12 = 20$ bit
Dimensione dei PFN:	$36 - 12 = 24$ bit

Cosa comportano indirizzi logici più piccoli degli indirizzi fisici?

- Un singolo processo può indirizzare soltanto un sottoinsieme della memoria fisica
- Non è una buona soluzione, ma è spesso utilizzata per mantenere la compatibilità architetturale (ad es. in IA-32)

La tabella di paginazione

Il cuore del meccanismo di paginazione è una struttura di dati che memorizza le associazioni tra indirizzi logici e fisici

Nella sua forma più semplice è una *tabella di paginazione* (*page table*):

- Definita per ciascun processo e collegata al PCB
- Codificante l'indirizzo logico ed il corrispondente indirizzo fisico per ciascuna pagina di memoria utilizzata dal processo
- Per ciascuna pagina memorizza anche attributi e diritti di accesso

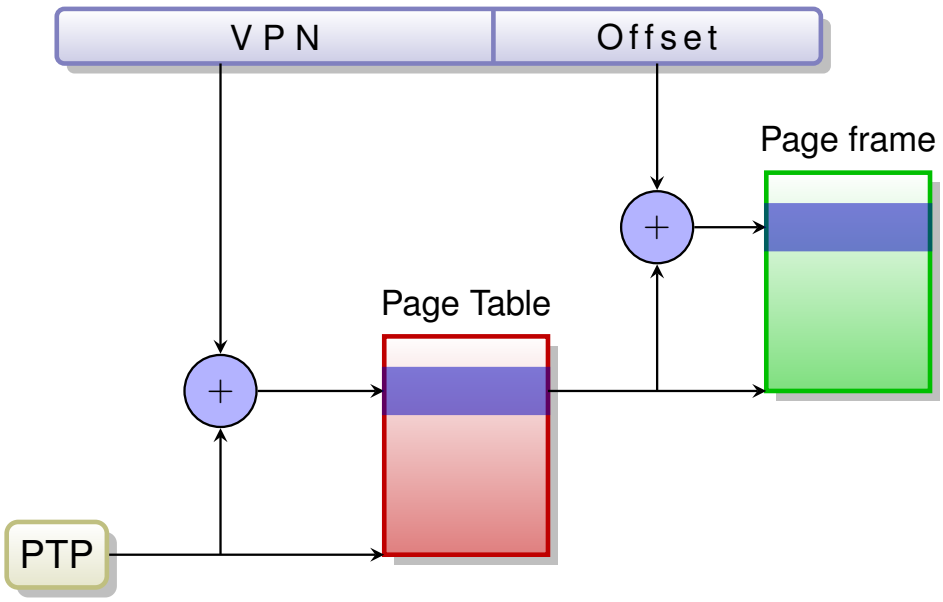
Esistono due tipi principali di tabelle di paginazione:

- La tabella di tipo diretto è specifica per un singolo processo, e memorizza l'indirizzo fisico associato a ciascun indirizzo logico utilizzato dal processo
- La tabella di tipo inverso è comune per tutti i processi, e memorizza l'indirizzo logico associato a ciascun indirizzo fisico in uso nel sistema

Il registro PTP

- La tabella di paginazione è memorizzata in memoria centrale
- Nei sistemi multiprocessore, ciascuna CPU può utilizzare una tabella di paginazione diversa
- Per poter trasformare un indirizzo logico in un indirizzo fisico la MMU deve leggere il contenuto della tabella di paginazione
- Pertanto ogni CPU con paginazione contiene un registro speciale:
  - Chiamato genericamente PTP (Page Table Pointer) o PTR (Page Table Register)
  - Contenente l'indirizzo fisico della tabella di paginazione
- Se l'architettura utilizza tabelle di paginazione dirette ed il SO è multiprogrammato, ciascun processo utilizza una propria tabella di paginazione
  - il contenuto del PTP deve essere modificato ad ogni cambio di contesto

Esempio di tabella di paginazione diretta



Paginazione gerarchica

Lo schema di paginazione diretta finora presentato ha un grave difetto: la tabella di paginazione contiene una voce per ciascuna pagina di memoria ed è diversa per ciascun processo

Esempio: con indirizzi logici da 32 bit, pagine da 4096 byte, PFN e flag codificati in 32 bit, ciascun processo occupa per la sola tabella di paginazione

$$2^{32}/2^{12} \times 4 = 4 \text{ MiB, ossia } 1024 \text{ pagine}$$

In un sistema con 100 processi attivi e 2 GiB di RAM, le tabelle di paginazione occuperebbero oltre il 12% memoria disponibile

Per porre rimedio a questo problema generalmente si hanno tabelle di paginazione dirette gerarchiche

- Sono allocate solo le tabelle "interne" che mappano blocchi di indirizzi logici almeno parzialmente utilizzati
- Le tabelle di paginazione non sono contigue in memoria fisica, riducendo i rischi di frammentazione della memoria

Paginazione diretta a due livelli

Il più semplice esempio di paginazione diretta gerarchica consiste in tabelle di paginazione organizzate su due livelli:

- Una tabella *esterna* o *di primo livello* contiene riferimenti a tabelle di paginazione *interne* o *di secondo livello*
- Le tabelle interne contengono i PFN degli indirizzi logici utilizzati dal processo

In uno schema a due livelli il VPN è costituito da due indici:

- La parte più significativa è l'indice della tabella interna entro la tabella esterna
- La parte meno significativa è l'indice della pagina entro la tabella interna

Ad esempio, se $N = 4096$ e il VPN è costituito da 20 bit divisi in due parti uguali:

- La tabella esterna ha $2^{10} = 1024$ voci
- Ciascuna tabella interna ha $2^{10} = 1024$ voci e mappa una porzione di $1024 \times 4096 = 4$ MiB di indirizzi logici
- Se una voce occupa 4 byte, ciascuna tabella occupa 4 KiB

Paginazione diretta a tre o quattro livelli

Nelle architetture a 64 bit due livelli di tabelle di paginazione non sono più sufficienti

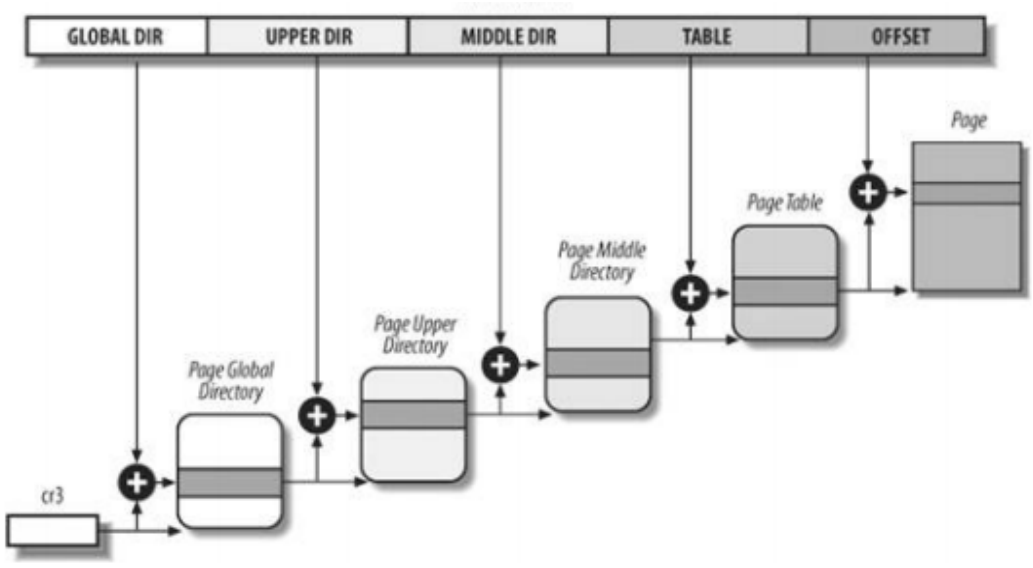
Con indirizzi logici da 64 bit, pagine da 8192 byte, voci di tabella da 8 byte, tabelle interne lunghe 4 pagine, la tabella esterna dovrebbe avere $2^{64-13-12} = 2^{39}$ voci (4 TiB)

La soluzione più diffusa consiste nell'aumentare il numero di livelli di tabelle di paginazione

Con tre livelli, indirizzi logici da 64 bit, pagine da 8192 byte, voci di tabella da 8 byte, tabelle interne lunghe 1 pagina, la tabella esterna ha $2^{64-13-20} = 2^{31}$ voci (16 GiB)

Con quattro livelli, indirizzi logici da 64 bit, pagine da 8192 byte, voci di tabella da 8 byte, tabelle interne lunghe 4 pagine, la tabella esterna ha $2^{64-13-36} = 2^{15}$ voci (256 KiB)

Esempio di paginazione a quattro livelli



Source: Bovet, Cesati, Understanding the Linux kernel, 3rd ed., 2005, OReilly

Translation Lookaside Buffer

Svantaggio principale della paginazione gerarchica: per ogni accesso ad una cella di memoria richiesto dal processo sono necessari diversi accessi in RAM

Ad esempio con una paginazione a tre livelli: per ciascun accesso in RAM si deve accedere a tre voci in tabelle di paginazione per ricavare l'indirizzo fisico

Translation Lookaside Buffer (TLB)

Meccanismo hardware o (parzialmente) software che memorizza in memoria statica veloce gli indirizzi fisici corrispondenti agli ultimi o più frequenti indirizzi logici utilizzati dalla CPU

Nelle architetture moderne questo meccanismo è estremamente importante per ottenere prestazioni significative

I processori più recenti hanno TLB organizzate con più livelli gerarchici, come le memorie cache

Tabella di paginazione inversa

Nei sistemi in cui lo spazio di indirizzamento logico è molto più grande dello spazio di indirizzamento fisico diventa conveniente utilizzare le tabelle di paginazione inverse

Una tabella di paginazione inversa

- è in comune per tutto il sistema
- contiene per ciascun indirizzo fisico (page frame) l'indirizzo logico corrispondente

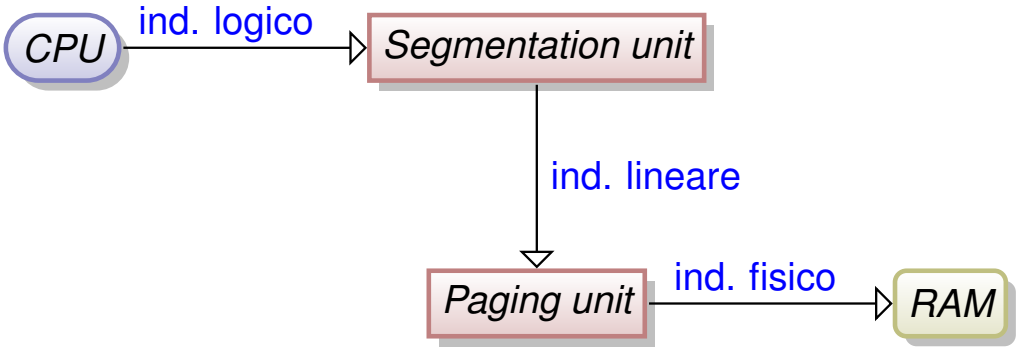
Ad esempio, in un sistema con 64 GiB di RAM e pagine da 8192 byte è sufficiente una tabella di paginazione inversa avente $2^{36-13} = 2^{23}$ voci

Quali problemi debbono essere risolti con questo approccio?

- Lo stesso indirizzo logico è utilizzato da più processi
  - Si aggiunge il PID agli indirizzi logici memorizzati
- La risoluzione di un indirizzo logico comporta la scansione dell'intera tabella
  - La tabella inversa è implementata come una struttura "hash": PID e VPN sono applicati ad una funzione che dà l'indice della voce su cui è attestata la lista di collisione

Segmentazione in Intel IA-32

L'architettura Intel IA-32 usa sia la segmentazione che la paginazione



- I segmenti sono contenuti in tabelle di descrittori di segmento globali (GDT) o locali per processo (LDT)
- Ogni processo può accedere a sei segmenti contemporaneamente tramite sei registri speciali che contengono indici entro le tabelle GDT o LDT

Segmentazione in Linux su Intel IA-32

Per essere il più possibile portabile su differenti architetture, il kernel Linux sfrutta essenzialmente soltanto il meccanismo della paginazione

Sono definiti sei segmenti "universali":

- segmento di codice per KM, indirizzi lineari tra `0xc0000000` e `0xffffffff`
- segmento di dati per KM, indirizzi lineari tra `0xc0000000` e `0xffffffff`
- segmento di codice per UM, indirizzi lineari tra 0 e `0xbfffffff`
- segmento di dati per UM, indirizzi lineari tra 0 e `0xbfffffff`
- segmento per lo stato del processo (Task State Segment, TSS, richiesto dall'hardware)
- segmento per la tabella locale dei descrittori LDT

I segmenti codice e dati si sovrappongono, quindi gli indirizzi logici di codice e dati producono indirizzi lineari nello stesso spazio

Paginazione in Intel IA-32

Ciascun processo nell'architettura Intel IA-32 può utilizzare indirizzi fisici a 32 bit oppure a 36 bit

Con indirizzi fisici a 32 bit:

- Le pagine hanno una dimensione prefissata di 4 KiB (due livelli di paginazione) oppure di 4 MiB (un solo livello)
- Con due livelli di paginazione le tabelle interne ed esterne hanno 1024 voci

Con indirizzi fisici a 36 bit (PAE, Physical Address Extension):

- Le pagine hanno dimensione prefissa di 4 KiB (tre livelli di paginazione) oppure di 2 MiB (due livelli)
- Con tre livelli di paginazione le tabelle di paginazione includono 4 (esterna), 512 e 512 voci

L'indirizzo della tabella di paginazione esterna è memorizzato nel registro speciale `cr3`

Paginazione in Linux su IA-32

Linux adotta uno schema di paginazione a quattro livelli

Lo schema generale viene adattato alle caratteristiche hardware dell'architettura

- In IA-32 senza PAE (indirizzi fisici da 32 bit), le quattro tabelle di paginazione si riducono a due
  - In pratica si considerano le tabelle eliminate come composte da una singola voce corrispondente a zero bit dell'indirizzo logico
- In IA-32 con PAE (indirizzi fisici da 36 bit), le quattro tabelle si riducono a tre
- Per ogni cambio di contesto, il contenuto del registro `cr3` viene salvato nel TSS del processo uscente, poi viene ripristinato il valore letto dal TSS del processo entrante
- Le tabelle di paginazione dei processi differiscono essenzialmente soltanto per le voci corrispondenti agli indirizzi logici inferiori a 3 GiB
  - Tutti i processi condividono gli spazi di indirizzamento logico e fisico quando eseguono in Kernel Mode