



Sistemi  
Operativi

Bruschi  
Monga Re

JOS

Stack  
Memoria virtuale  
Memory  
mapping

# Sistemi Operativi<sup>1</sup>

Mattia Monga

Dip. di Informatica  
Università degli Studi di Milano, Italia  
[mattia.monga@unimi.it](mailto:mattia.monga@unimi.it)

a.a. 2013/14

<sup>1</sup> © 2008–14 M. Monga. Creative Commons Attribuzione — Condividi allo stesso modo 4.0 Internazionale. <http://creativecommons.org/licenses/by-sa/4.0/deed.it>. Immagini tratte da [2] e da Wikipedia.



Sistemi  
Operativi

Bruschi  
Monga Re

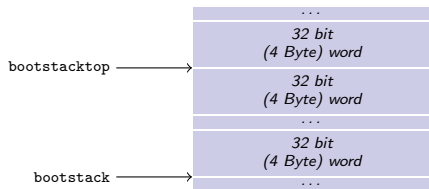
JOS

Stack

Memoria virtuale

Memory  
mapping

# Lezione XIX: Gestione della memoria in JOS



- $ESP == bootstacktop$
- $bootstacktop == bootstack + KSTKSIZE$
- Una *push sottrae* 4 Byte all'indirizzo ESP, una *pop* li *aggiunge*. (ESP è sempre divisibile per 4)
- Una *call* gestisce automaticamente il salvataggio dell'indirizzo di ritorno sullo stack, mentre EBP deve essere gestito a mano (salvandovi il vecchio ESP in modo da poter identificare facilmente il *record di attivazione* o *stack frame*)



Nei manuali x86 si parla di 3 tipologie di indirizzi

**virtuali** quando sono relativi ad un segmento: un puntatore C è un *offset*

**lineare** selettore di segmento + offset permette di calcolare un indirizzo nello spazio di indirizzamento (virtuale) lineare 0–4GB

**fisico** l'indirizzo lineare è “mappato” su un indirizzo fisico dalla MMU (che non può essere saltata!)



Segmentazione e MMU non possono essere saltati: il programmatore “vede” esclusivamente indirizzi virtuali.

- JOS configura tutti i segmenti (in `boot/boot.S` tramite la prima GDT) in `0-0xffffffff` (0-4GB), quindi il segmento può essere ignorato
- Quando serve manipolare indirizzi fisici (che **non** possono essere dereferenziati) devono essere usati *numeri* che sarà utile contrassegnare con il tipo `physaddr_t`
- Un numero che può essere dereferenziato (perché si tratta di un indirizzo virtuale) verrà contrassegnato con `uintptr_t` e per dereferenziarlo come T va interpretato come `T*`.



# Il mapping iniziale

I kernel sono generalmente caricati a un indirizzo (lineare) alto, p.es. `0xf0100000` (3,75GB), che potrebbe perfino non esistere nello spazio fisico.

- il programmatore del kernel (e il programma!) usa `0xf0100000` (virtuale)
- il boot loader carica il kernel all'indirizzo `0x00100000`
- il boot loader istruisce la MMU perché mappi `0xf0100000`  
→ `0x00100000`

Sistemi  
Operativi

Bruschi  
Monga Re

JOS

Stack  
Memoria virtuale  
Memory  
mapping

# le prime page table

- La page table 'zeresima' in boot/boot.S configura il mapping *identità*, quindi indirizzi lineari uguali a fisici.
- La prima vera page table è in kern/entrypgdir.c

lineare	fisico
0xf0000000 (KERNBASE)	0x00000000
...	...
0xf0400000	0x00400000 (4MB)
0x00000000	0x00000000
...	...
0x00400000	0x00400000 (4MB)
*	eccezione



# Macro che sostituiscono la MMU

$0xf0000000 == \text{KERNBASE} \rightarrow 0x00000000$

$0xf0100000 == \text{KERNBASE} + 1\text{MB}$

$0xf0400000 == \text{KERNBASE} + 4\text{MB} \rightarrow 0x00400000$

Alla fine del lab2 verranno mappati 256MB. Si noti che esiste una relazione semplice fra fisico e lineare: quando serve il programmatore può calcolare l'indirizzo lineare aggiungendo KERNBASE al fisico. Per farlo meglio usare KADDR (e PADDR per l'inverso) che controllano che il numero cui si applica sia sensato.

Sistemi  
Operativi

Bruschi  
Monga Re

JOS

Stack

Memoria virtuale

Memory  
mapping



# Le strutture dati per la gestione della memoria



Sistemi  
Operativi

Bruschi  
Monga Re

JOS

Stack  
Memoria virtuale  
Memory  
mapping

- 1 **struct** PageInfo \*pages; // *Physical page state array*
- 2 **static struct** PageInfo \*page\_free\_list; // *Free list of physical pages*

(Lo static garantisce che page\_free\_list sia “privata” del file kern/pmap.c. Analogamente la variabile nextfree è privata alla funzione boot\_alloc, anche se la durata del suo valore è analoga a quella di una variabile globale: si mantiene fra una chiamata e l'altra)

- 1 L'array npages viene allocata inizialmente con boot\_alloc
- 2 Viene inizializzata con page\_init; una pagina è libera se fa parte della lista collegata page\_free\_list
- 3 L'allocazione poi deve avvenire sempre con page\_alloc

Il reference count di una pagina (quante pagine virtuali vengono mappate su di essa) è aggiornato da page\_insert. Per altri usi occorre farlo a mano.



Sistemi  
Operativi

Bruschi  
Monga Re

JOS

Stack

Memoria virtuale

Memory  
mapping