



Sistemi Operativi¹

Mattia Monga

Dip. di Informatica e Comunicazione
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

a.a. 2010/11

¹ © 2011 M. Monga. Creative Commons Attribuzione-Condividi allo stesso modo 2.5 Italia License.
<http://creativecommons.org/licenses/by-sa/2.5/it/>. Immagini tratte da [?] e da Wikipedia.



Lezione II: Introduzione laboratorio



Informazioni sul corso

- 6 (Bruschi) + 4 (Monga) ore di lezione settimanali (12 crediti)
- Lezioni di teoria e in laboratorio
- Esame:
 - Scritto con domande a risposta multipla + orale
 - Prova pratica per la parte di laboratorio
- Libro di testo: *Operating System Design and Implementation*, di Tanenbaum e Woodhull, III ed.
- <http://homes.dico.unimi.it/sisop/>



Things A Computer Scientist Rarely Talks About

“When I talk about computer science as a possible basis for insights about God, of course I’m not thinking about God as a super-smart intellect surrounded by large clusters of ultrafast Linux workstations and great search engines. That’s the user’s point of view.” [Donald E. Knuth]



Il sistema operativo



Sistemi Operativi
Bruschi Monga

Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.
Il ruolo del s.o.
Astrazioni

Cos'è un sistema operativo

Un insieme di programmi che:

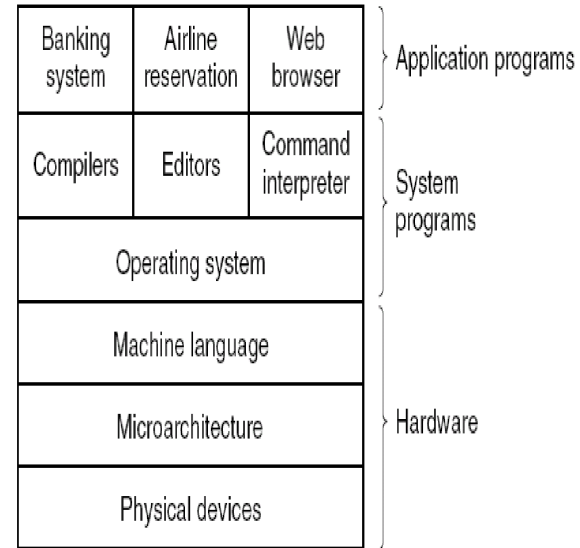
- Gestisce in modo ottimale le risorse di un calcolatore;
- Facilita a programmatori ed utenti finali l'uso della sottostante macchina hardware

The onion model



Sistemi Operativi
Bruschi Monga

Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.
Il ruolo del s.o.
Astrazioni



Kernel/User mode



Sistemi Operativi
Bruschi Monga

Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.
Il ruolo del s.o.
Astrazioni

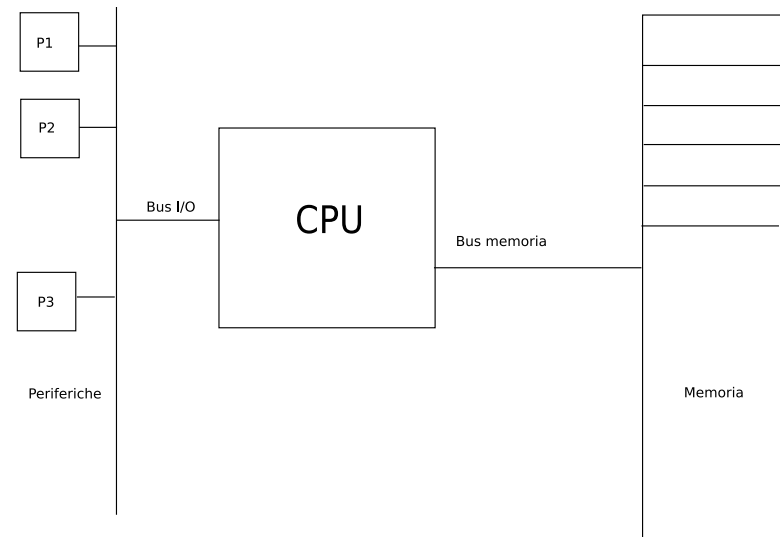
- Il s.o. è l'unico programma che esegue con il totale controllo delle risorse hardware (kernel mode).
- Gli altri programmi si appoggiano unicamente sui servizi del s.o. e la loro esecuzione è gestita e controllata dal s.o. (user mode)
- In molti processori questa separazione è imposta via hardware

La macchina di Von Neumann



Sistemi Operativi
Bruschi Monga

Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.
Il ruolo del s.o.
Astrazioni





Sistemi Operativi

Bruschi Monga

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

- Registri a 32 bit
 - EAX, EBX, ECX, EDX,
 - ESI, EDI,
 - EBP, ESP,
 - EIP, EFLAGS
- Registri a 16 bit:
 - CS, DS, SS,
 - ES, FS, GS
- Real e Protected mode



Sistemi Operativi

Bruschi Monga

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

- Si possono indirizzare direttamente porzioni di 8 bit, 1 byte ($AX = AH+AL$, $EAX = 16bit+AX$)
- Programmable Interrupt Controller (PIC): i8259 compatibile



Sistemi Operativi

Bruschi Monga

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

I processori moderni hanno modalità di funzionamento in cui sono permesse operazioni diverse (*ring*), p.es. indirizzare tutta la memoria. i386 permette 4 ring diversi, di cui normalmente vengono usati solo 2 (Minix ne usa 3):

- 1 kernel (supervisor) mode
- 2 user mode



Sistemi Operativi

Bruschi Monga

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

	Real mode	32-bit Protected mode
Protezioni hw	no	sí
Spazio di indirizzamento	2^{20}	2^{32}

- Real mode: memoria max 2^{20} byte, indirizzo ottenuto con due registri a 16 ($SS:OFFSET$)
*indirizzo = 16 * selettore + offset*
 - ci sono piú modi riferirsi allo stesso indirizzo: 07C0:0000 e 0000:7C00 sono la stessa locazione fisica.
 - A20 gate
- Protected mode: il segmento è stabilito da un descrittore (che può essere cambiato solo in kernel mode)



- NASM, <http://nasm.sourceforge.org>
- PC Assembly Language, by Paul A. Carter <http://www.drpaulcarter.com/pcasm/>
- Minix usa un altro assembler (x86) <http://www.users.csbsju.edu/%7Ecburch/cs/portfolio/x86.pdf>

```

1  mov eax, 3 ; eax = 3
2  mov bx, ax ; bx = ax
3  add eax, 4 ; eax = eax + 4
4  add al, ah ; al = al + ah
5  L8:db "A" ; *L8 = 'A'
6  mov al, [L8] ; al = *L8
    
```



Gli assembleri x86 si distinguono per la famiglia sintattica

Intel (nasm)	AT&T (as86, gas)
mov ebx, eax	movl %eax, %ebx
mov eax, 42	movl \$42, %eax
mov [ebx], eax	movl %eax, 0(%ebx)
mov [ebx+4], eax	movl %eax, 4(%ebx)
mov byte [ebx], al	movb %eax, 0(%ebx)
call eax	call *%eax



Qemu <http://fabrice.bellard.free.fr/qemu> PC (x86 or x86_64 processor)

- i440FX host PCI bridge and PIIX3 PCI to ISA bridge
- Cirrus CLGD 5446 PCI VGA card
- PS/2 mouse and keyboard
- 2 PCI IDE interfaces with hard disk and CD-ROM support
- Floppy disk
- NE2000 PCI network adapters
- Serial ports
- PCI UHCI USB controller and a virtual USB hub.



Ogni periferica è dotata di un controller. Il controller avrà registri che conservano lo stato della periferica. Come accedere (leggere o scrivere) al contenuto dei registri?

- 1 Spazi di indirizzamento separati chiamati port. Vi si accede con istruzioni particolari:
 - **out port, eax**
 - **in eax, port**
- 2 Memory-mapped I/O, lo spazio di indirizzamento è unico
 - **mov [address], eax**
 - **mov eax, [address]**

Sequenza di boot



Sistemi Operativi

Bruschi Monga

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

Cosa succede quando si accende un PC?

- 1 Inizia l'esecuzione del programma contenuto nel firmware (BIOS)
- 2 Il BIOS carica il programma contenuto nel **boot sector**
- 3 Il programma di boot carica il sistema operativo
- 4 A questo punto il controllo della macchina è affidato al s.o., a cui dovranno essere richiesti i caricamenti di altri programmi

56

Programming the iron



Sistemi Operativi

Bruschi Monga

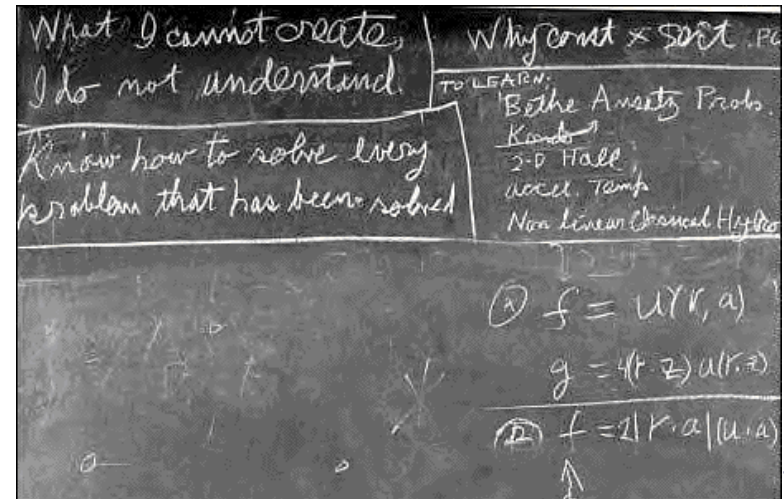
Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

What I cannot create I do not understand. [R. Feynman]



7

Programming the iron



Sistemi Operativi

Bruschi Monga

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

```
1 ;Copyright (C) 2008, 2009 by Mattia Monga <mattia.monga@unimi.it>
2 bits 16 ; 16 bit real mode
3 org 0x7C00 ; origine indirizzo 0000:7C00
4
5 start:
6 mov ax, 0xb800 ; text video memory
7 mov ds, ax ; ds non accessibile direttamente
8 mov bx, 10
9 write:
10 cmp bx, 0
11 jz end
12 mov byte [ds:bx], 'm' ; indirizzamento relativo a ds
13 mov byte [ds:bx+1], 0x0F ; attrib = white on black
14 sub bx, 2
15 jmp write
16 end:
17 hlt
18
19 times 510-($-$$) db 0 ; 0-padding
20 dw 0xAA55
```

58

Interruzioni



Sistemi Operativi

Bruschi Monga

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

Un'interruzione (*interrupt request (IRQ)*) è un segnale (tipicamente generato da una periferica, ma non solo) che viene notificato alla CPU. La CPU, secondo le politiche programmate nel PIC, risponderà all'interruzione eseguendo il codice del gestore dell'interruzione (*interrupt handler*). Dal punto di vista del programmatore la generazione di un'IRQ è analoga ad una chiamata di procedura, ma:

- Il codice è completamente disaccoppiato, potenzialmente in uno spazio di indirizzamento diverso (permette le protezioni)
- Non occorre conoscere l'indirizzo della procedura
- La tempistica dell'esecuzione è affidata alla CPU

59

BIOS (1/3)



DICo

Sistemi Operativi

Bruschi Monga

```
1 ;Copyright (C) 2008 by Mattia Monga <mattia.monga@unimi.it>
2 bits 16 ; 16 bit real mode
3 org 0x7C00 ; origine indirizzo 0000:7C00
4
5 start:
6     cld ; clears direction flag (index regs incremented)
7     mov si, boot
8     call message
9 working:
10    mov si, work
11    call message
12
13    call waitenter
14    jmp working
```

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

60

BIOS (2/3)



DICo

Sistemi Operativi

Bruschi Monga

```
1 message:
2     lodsb ; carica un byte da [DS:SI] in AL e inc SI
3     cmp al, 0
4     jz done
5     mov ah, 0x0E ; write char to screen in text mode
6     mov bx, 0 ; BH page number BL foreground color
7     int 0x10 ; write AL to screen (BIOS)
8     jmp message
9 done: ret
10
11 boot: db "Loading unuseful system..." , 10, 13, 0
12 work: db "I've done my unuseful stuff!" , 10, 13, 0
13 cont: db "Hit ENTER to continue..." , 10, 13, 0
14 wow: db "Great! Hello world!" , 10, 13, 0
```

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

61

BIOS (3/3)



DICo

Sistemi Operativi

Bruschi Monga

```
1 waitenter: mov si, cont
2             call message
3             mov ah, 0
4             int 0x16 ; Wait for keypress (BIOS)
5             cmp al, 'm'
6             jz egg
7             cmp al, 'b'
8             jz basic
9             cmp al, 13
10            jnz waitenter
11            ret
12 egg: mov si, wow
13         call message
14         jmp waitenter
15 basic: int 0x18 ; basic (BIOS)
16         hlt
17
18         times 510-($-$$) db 0
19         dw 0xAA55
```

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

62

Cos'è un sistema operativo



DICo

Sistemi Operativi

Bruschi Monga

Sistema Operativo

Un s.o. è un programma che rende conveniente l'uso dello hardware

- fornendo astrazioni che semplificano l'uso delle periferiche e della memoria
- gestendo opportunamente le risorse fra tutte le attività in corso

Concetti generali

La macchina fisica

Hardware
Concetti di base
Perché un s.o.

Il ruolo del s.o.
Astrazioni

63



Le principali sono:

- System call
- Memoria virtuale
- Processo
- File
- Shell