



Cos'è un sistema operativo

Un insieme di programmi (*software*) che:

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

Ottiene questi obiettivi virtualizzando il dispositivo di calcolo: programmatori e utenti finali interagiscono con una macchina virtuale con caratteristiche e proprietà largamente *indipendenti* da quelle dello *hardware*.



- Sistema operativo: l'unico programma interpretato interamente dalla macchina reale;
- Applicazioni: programmi che, sfruttando il s.o., usano le risorse virtuali per fornire valore all'utente.

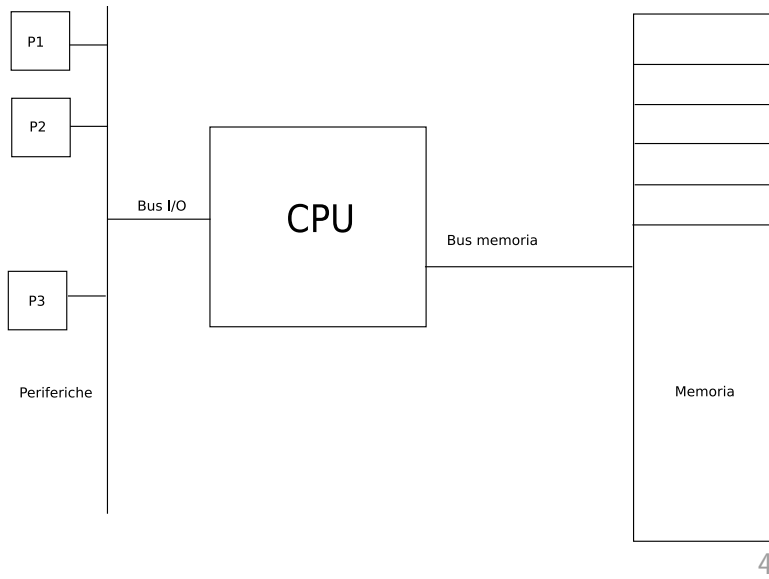


- Sistema operativo vs. *librerie*: entrambe le cose forniscono 'servizi' ai programmi, ma lo fanno in maniera *fondamentalmente* diversa. Le librerie sono a tutti gli effetti componenti di un programma, il sistema operativo costituisce il contesto di esecuzione.
- Esistono applicazioni chiamate 'macchine virtuali': java, QEmu, VMWare, VirtualBox, ... Il concetto è analogo a quello del s.o., ma a livello applicativo (con qualche eccezione).



- 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 Re
Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.

48

La macchina i386



- 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 Re
Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.

49

Linguaggio macchina



- 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 Re
Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.

50

Protezioni hardware



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 Re
Concetti generali
La macchina fisica
Hardware
Concetti di base
Perché un s.o.

51



	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 per 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.us>
- PC Assembly Language, by Paul A. Carter
<https://pacman128.github.io/pcasm/>
- Un altro assembler molto diffuso è gas
<http://www.ibm.com/developerworks/linux/library/l-gas-nasm/index.html>

```

mov    eax, 3 ; eax = 3
mov    bx, ax ; bx = ax
add    eax, 4 ; eax = eax + 4
add    al, ah ; al = al + ah
L8:db  "A"    ; *L8 = 'A'
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 (i386 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]

56



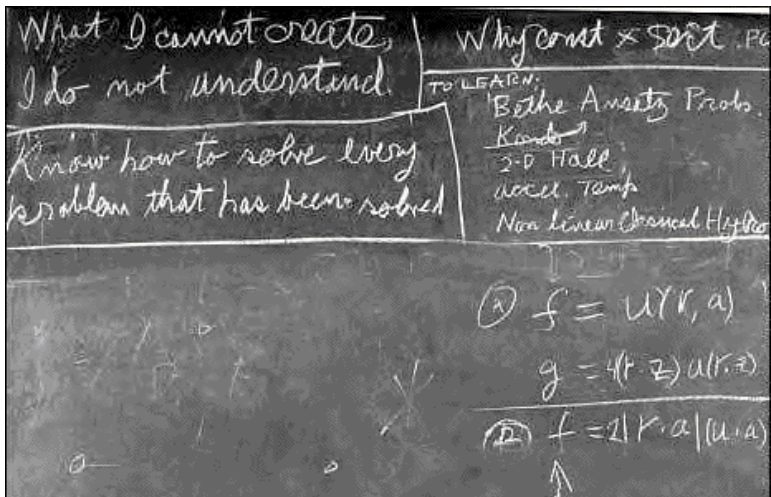
Cosa succede quando si accende un PC?

- ① Inizia l'esecuzione del 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

57



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



8



```

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

```

59



Sistemi
Operativi

**Bruschi
Monga Re**

Concetti
generali

La macchina
fisica

Hardware

Concetti di base

Perché un s.o.