



**SAPIENZA**  
UNIVERSITÀ DI ROMA

# GESTIONE FILE

Dott. Franco Liberati

# Argomenti

01

CHIAMATE A SISTEMA

02

FILE

03

GESTIONE FILE TESTUALE

04

GESTIONE FILE BINARI  
IMMAGINE



A glowing blue microchip is centered on a circuit board. The chip has a bright blue, textured surface and is surrounded by a glowing blue border. Numerous glowing blue lines and dots are scattered across the dark blue background, creating a sense of connectivity and data flow. The overall aesthetic is futuristic and technological.

**CHIAMATE A SISTEMA**



# Chiamate a sistema



Le chiamate a sistema (syscall) costituiscono l'interfaccia tra un programma in esecuzione ed il sistema operativo

Le syscall sono generalmente disponibili in forma di istruzioni in linguaggio assembler

Passaggio di parametri :

- ❖ Tramite registri
- ❖ Tabella di memoria (si passa l'indirizzo dove risiedono i parametri)
- ❖ Tramite stack (si riserva una area di memoria dove si possono collocare parametri (push) o prelevarli (pop))



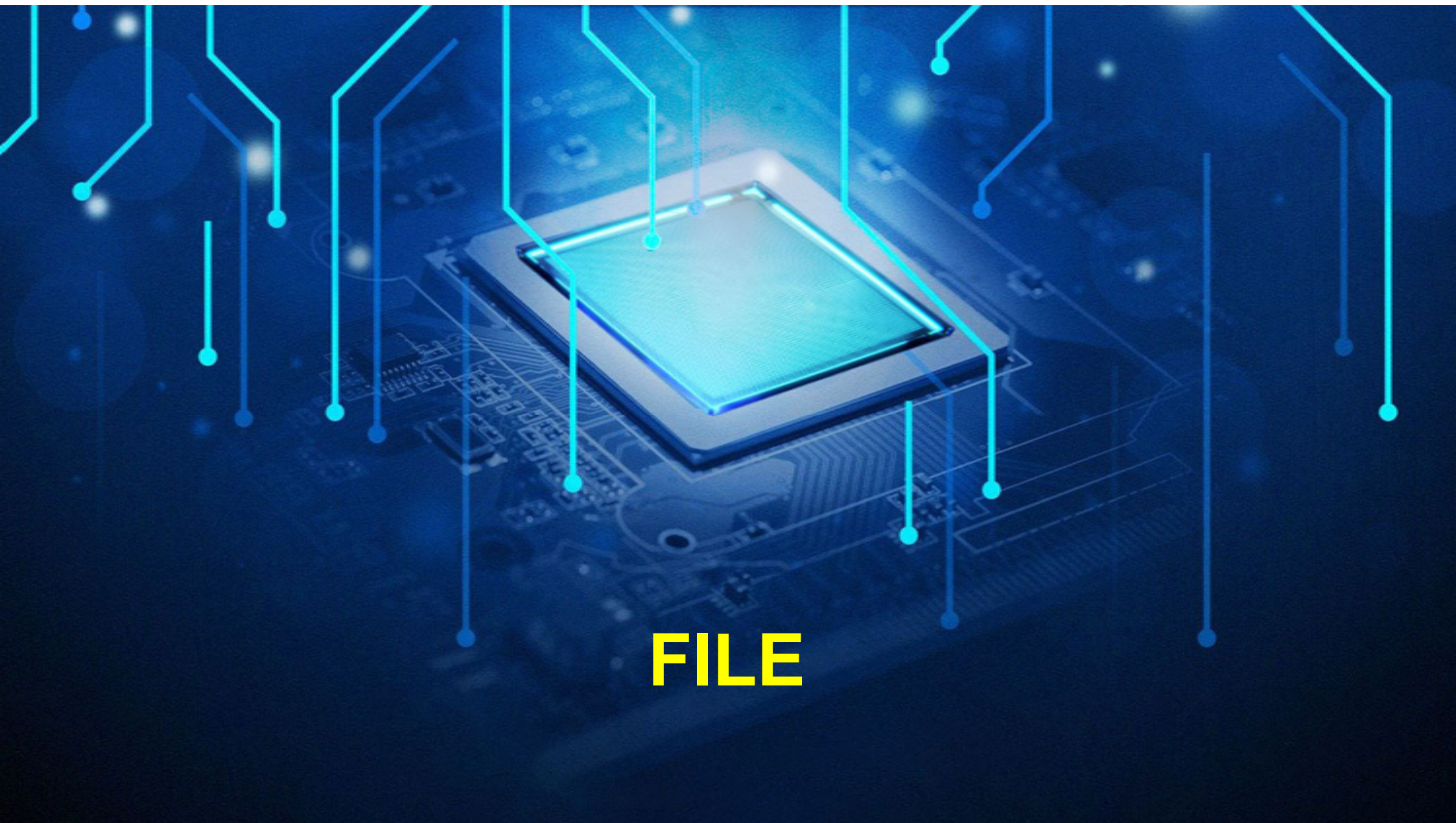
# Chiamate a sistema



Tipicamente le syscall sono classificate in categorie tra cui le principali sono

Categoria	Esempio di Syscall
Controllo di processo	Abort, Load, Execute, Wait
Manipolazione file	Create, Delete, Open, Close
Gestione dispositivi	Request, Release, Read, Write
Gestione informazioni	GetTime, SetTime
Comunicazioni	Send, Receive







# FILE



Un **file** (o documento digitale) è una collezione di dati omogenea, avente un significato ben determinato, organizzato in formato e conservata in un supporto digitale di memorizzazione

I file sono suddivisibili in **due categorie**

- ❑ **File di testo** (i valori corrispondono a codifiche testuali come UNICODE e ASCII)

- ❑ Es.: File temporaneo, File di configurazione, File di log,...

- ❑ **File binario** (i valori hanno un significato variabile a seconda della codifica)

- ❑ Es.: File audio (es.: file musicale, mp3); File immagine (es.: immagine, bmp, jpeg;); File video (es. film, mp4); File multimediale (es. audio-video-testo); File di sistema; File sorgente; File oggetto; File eseguibile;



# FILE



Operazione sui file:

- ☐ Apertura
  - ☐ Modalità solo lettura
  - ☐ Modalità solo scrittura
  - ☐ Modalità lettura e scrittura
  - ☐ Modalità append (inserisce byte alla fine del file)
- ☐ Chiusura
- ☐ Scrittura e lettura dei contenuti





**GESTIONE FILE TESTUALI**

# GESTIONE FILE TESTUALE

Un **File di testo** è una collezione di valori che corrispondono a codifiche testuali (come UNICODE e ASCII)

The screenshot displays the hexed.it web application interface. The top navigation bar includes icons for file management (Nuovo file, Apri file, Salva con nome, Annulla, Ripeti, Strumenti, Traduci, Impostazioni, Aiuto) and a search bar. The main content area is divided into three sections: 'Informazioni File', 'Ispettore dati (Little-endian)', and 'Vai a'.

**Informazioni File**

Nome File	filetesto.txt
Dim. File	469 Bytes

**Ispettore dati (Little-endian)**

Tipo	Non segnato (+)	Segnato (±)
Integer a 8-bit	109	109
Integer a 16-bit	25965	25965
Integer a 24-bit	8021357	8021357
Integer a 32-bit	2054841709	2054841709
Integer a 64-bit (+)	7306000158670087533	
Integer a 64-bit (±)	7306000158670087533	
Virg. mob. a 16-bit	1389	
Virg. mob. a 32-bit	3,2503284e+35	
Virg. mob. 64-bit	2,609876851780575e+180	
LEB128 (+)	109	
LEB128 (±)	-19	
Data/Ora MS-DOS	26/03/2041 12:43:26 Local	
Data/Ora OLE 2.0	Data non valida	
Data/Ora UNIX	11/02/2035 21:21:49 UTC	
Data/Ora Macintosh HFS	10/02/1969 22:21:49 Local	
Data/Ora Macintosh HFS+	10/02/1969 21:21:49 UTC	

**Vai a**

Indirizzo attuale: 0x00000004  
Ultimo Indirizzo: 0x000001D4  
Vai a:

**Ricerca**

Cerca:

Tipo di dati:

- ☐ Integer a 8-bit
- ☐ Integer a 16-bit
- ☐ Integer a 24-bit
- ☐ Integer a 32-bit
- ☐ Integer a 64-bit
- ☐ Virgola Mobile 16-bit
- ☐ Virgola Mobile 32-bit
- ☐ Virgola Mobile 64-bit
- ☐ LEB128
- ☐ Valore Esadecimale
- ☐ Testo

Codifiche Testo:

Distingui Maiusc./Minusc.: ☐ Maiusc./minusc. (veloce)

Ordine Byte: ☒ Little-endian ☐ Big-endian

# GESTIONE FILE TESTUALE

SERVIZI	CODICE	ARGOMENTI	RISULTATO
Open_file	13	<p><b>\$a0 = locazione della stringa relativa al nome del file che vuole essere aperto (si intende il percorso dove risiede il file)</b></p> <p><b>\$a1=modalità di apertura (flag):</b></p> <ul style="list-style-type: none"><li><b>0= sola lettura</b></li><li><b>1= sola scrittura (e creazione)</b></li><li><b>9= append (e creazione)</b></li></ul> <p><b>\$a2=mode (non va inizializzato)</b></p>	<b>\$v0= descrittore del file</b>
Close_file	16	<b>\$a0= descrittore file</b>	



# GESTIONE FILE TESTUALE

## Creazione file testuale

```
.txt
.globl main
main:
    la $a0,nome_file #apertura file (creazione)
    li $a1,1 #file aperto per la scrittura
    li $a2,0 #
    li $v0,13 #
    syscall #
    move $t0,$v0 #descrittore del file (indirizzo del file in memoria)

    move $a0,$t0 #chiusura file
    li $v0,16 #
    syscall #

    li $v0,10
    syscall

.data
nome_file: .asciiz "C:\\Pippo.txt"
```

# GESTIONE FILE TESTUALE

SERVIZI	CODICE	ARGOMENTI	RISULTATO
Read from file	14	<b>\$a0 = descrittore a file</b> <b>\$a1 = indirizzo a partire dal quale riportare la stringa letta dal documento</b> <b>\$a2 = numero di caratteri da leggere</b>	\$v0= numeri di caratteri letti (se negativo è accaduto un errore)
Write to file	15	<b>\$a0 = descrittore a file</b> <b>\$a1 = indirizzo della stringa da scrivere nel documento</b> <b>\$a2 = numero di caratteri da scrivere</b>	\$v0= numeri di caratteri scritti (se negativo è accaduto un errore)

# GESTIONE FILE TESTUALE

## Scrittura in un file testuale

```
.txt
.globl main
main:
    la $a0,nome_file      # apertura file per scrivere
    li $a1,1              #
    li $a2,0              #
    li $v0,13             #
    syscall               #
    move $t0,$v0          # memorizzazione descrittore file
    move $a0,$t0          # scrittura file
    la $a1,messaggio      # locazione dove risiede il dato da scrivere
    li $a2,5              # numeri di caratteri da scrivere
    li $v0,15             #
    syscall               #
    move $a0,$t0          # chiusura file
    li $v0,16             #
    syscall               #
    li $v0,10             # chiusura programma
    syscall               #
```

```
.data
nome_file: .asciiz "C:\\Pippo.txt"
messaggio: .asciiz "Ciao"
```



# GESTIONE FILE TESTUALE

## Lettura da un file testuale

```
.text
.globl main
main:
    la $a0,nome_file
    li $a1,0
    li $a2,0
    li $v0,13
    syscall                                #apertura file per lettura
    move $a0,$v0
    la $a1,messaggio_letto
    li $a2,100
    li $v0,14
    syscall                                #lettura messaggi su file
    la $a0,messaggio_letto
    li $v0,4
    syscall                                #stampa a video quando letto nel file
    li $v0,16
    syscall                                #chiusura file
    li $v0,10
    syscall                                #chiusura programma

.data
nome_file: .asciiz "C:\\Pippo.txt"
messaggio_letto: .space 255
```

The background of the slide features a dark blue, high-tech aesthetic. In the center, a square, glowing blue chip is mounted on a circuit board. Numerous glowing blue lines, resembling circuit traces or data paths, extend from the chip and across the frame. The overall effect is one of digital connectivity and advanced technology.

# **GESTIONE FILE BINARI IMMAGINE**



# GESTIONE FILE BINARIO

**File binario** (i valori hanno un significato variabile a seconda della codifica)

❑ File audio (es.: file musicale, mp3); File immagine (es.: immagine, bmp, jpeg;); File video (es. film, mp4); File multimediale (es. audio-video-testo); File di sistema; File sorgente; File oggetto; File eseguibile;

In un file di immagine i valori corrispondono a dei punti di colore (*pixel*)



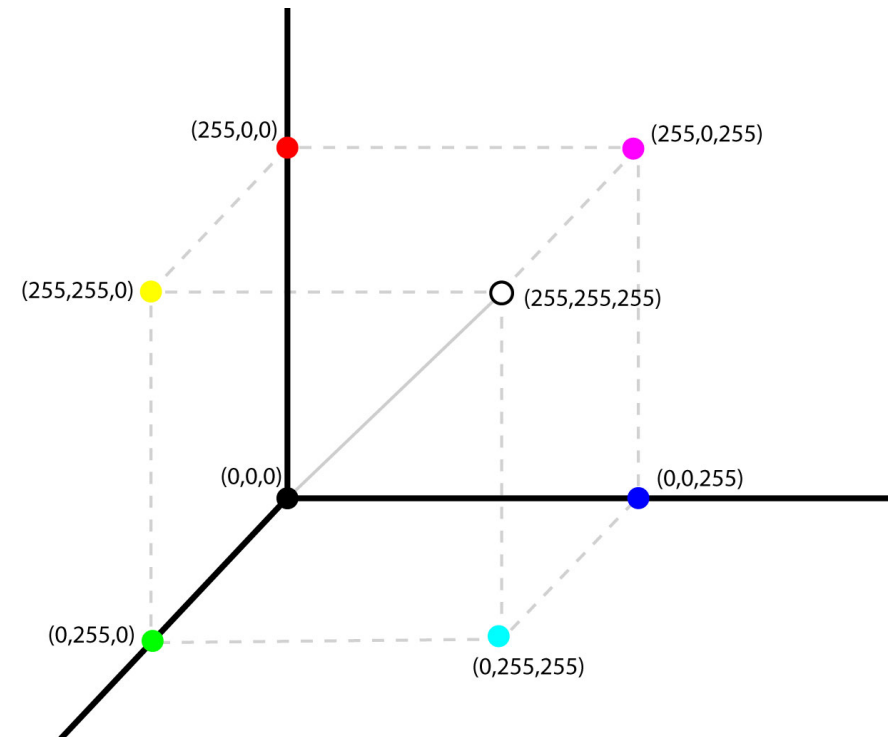
# IMMAGINE

I punti di colore (pixel) di una immagine fanno riferimento ad un **modello colore**

Il **modello colore RGB** identifica un punto mediante i principi ottici della sintesi sottrattiva.

Il punto è visto come la composizione di **tre componenti principali** (o canali): il rosso, il verde e il blu.

In termini pratici il valore che rappresenta il colore è diviso in tre campi (di uguale lunghezza) ciascuno che identifica la quantità di componente principale relativa

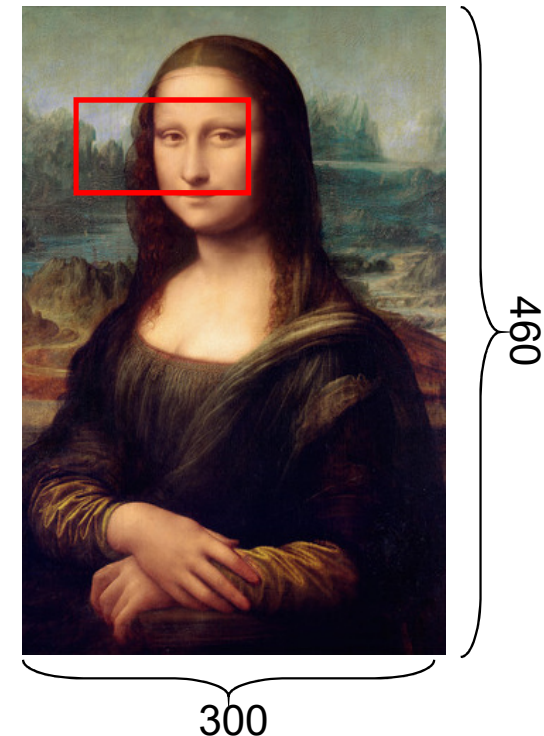


# FORMATO BITMAP

**BitMaP** è un formato proprietario proposto da Microsoft (poi aperto al pubblico) per l'elaborazione (PAINT, Microsoft Office Image Manager) e l'archiviazione di immagini digitali

Il formato consta di una intestazione e l'immagine digitale

L'immagine digitale è di tipo **raster**: una matrice di pixel descritti secondo lo standard RGB a 32bit (8 bit per il canale rosso, 8bit per il canale verde e 8bit per il canale blu).



# FORMATO BITMAP

In particolare il file BMP è strutturato in tre campi:

- ❑ **Intestazione** (HEADER) in cui sono presenti l'identificatore del formato, la grandezza del file e la posizione in cui sono archiviati i pixel

- ❑ **Descrizione immagine** (INFO HEADER) in cui sono specificate le principali informazioni sull'immagine (larghezza, altezza, numero di bit per specificare un punto di colore, dimensione file,...)

- ❑ **I punti di immagine o pixel** (PIXEL DATA) cioè i dati

- ❑ Opzionalmente tra INFOHEADER e PIXEL DATA è presente una tabella dei colori (COLOR TABLE) ma non sarà trattato in questa esposizione

**HEADER**  
(14byte)

**INFOHEADER**  
(40byte)

**COLOR TABLE**  
(da 0 a 4xnum colori in byte)

**PIXEL DATA**

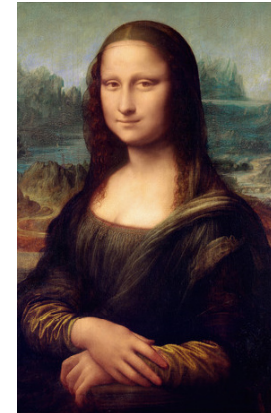


# FORMATO BITMAP

## Campo Header

### HEADER

CAMPO	DIMENSIONE	DESCRIZIONE
<i>Signature</i>	2byte	Identificatore (fisso "BM")
<i>FileSize</i>	4byte	Dimensione del file (in byte)
<i>reserved</i>	4byte	Riservato (fisso "0")
<i>DataOffset</i>	4byte	Indirizzo dove iniziano i pixel



Data View		
Add...	Hexadecimal (1 Byte)	Text (ASCII)
000000	42 4D 66 51 06 00 00 00 00 00 00 36 00 00 00 28 00	B M f Q . . . . . 6 . . . . ( .
000010	00 00 2C 01 00 00 CC 01 00 00 01 00 18 00 00 00	. . , . . . . . . . . . . . .
000020	00 00 30 51 06 00 00 00 00 00 00 00 00 00 00	. . 0 Q . . . . . . . . . . . .
000030	00 00 00 00 00 00 15 05 06 15 05 06 14 04 05 14	. . . . . . . . . . . . . . . .
000040	04 05 17 06 09 16 05 08 16 05 08 16 05 08 17 06	. . . . . . . . . . . . . . . .
000050	09 17 06 09 16 05 08 16 05 08 16 06 0D 16 06 0D	. . . . . . . . . . . . . . . .
000060	16 06 0D 18 06 0D 18 06 0D 17 05 0C 18 03 0B 17	. . . . . . . . . . . . . . . .
000070	02 0A 18 05 08 1B 07 0C 1C 08 0D 1A 05 0D 17 05	. . . . . . . . . . . . . . . .
000080	0C 18 07 0B 19 08 0B 17 06 09 19 08 0B 19 08 0B	. . . . . . . . . . . . . . . .
000090	19 08 0B 19 08 0B 18 07 0A 18 07 0A 19 08 0B 19	. . . . . . . . . . . . . . . .
0000A0	08 0C 1B 0A 15 19 08 13 18 07 10 18 07 10 17 07	. . . . . . . . . . . . . . . .
0000B0	0E 14 04 0B 14 05 09 15 06 0A 17 05 0C 1A 08 0F	. . . . . . . . . . . . . . . .

NB: i dati sono little-endian: dimensione file hex66510600 è hex00065166 cioè 414054 (cioè  $(300 \cdot 460 \cdot 3) + 54$ )



# FORMATO BITMAP

## Campo InfoHeader



INFOHEADER	40byte	
CAMPO	DIMENSIONE	DESCRIZIONE
Size	4byte	Dimensione info header ("40")
Width	4byte	Larghezza immagine in pixel
Height	4byte	Altezza immagine in pixel
Planes	2byte	Numero di immagini nel file
Bits Per Pixel	2byte	Bit per pixel : 1,4,8,16,32 <i>NB: noi utilizzeremo solo file a 32bit per pixel 8 bit per la componente di Rosso 8 bit per il Verde 8 per il Blu</i>
Compression	4byte	Tipo di compressione (0: nessuna ; 1: RLE8; 2: RLE4) <i>NB: noi utilizzeremo file non compressi</i>
ImageSize	4byte	Dimensione immagine
XpixelsPerM	4byte	Risoluzione orizzontale
YpixelsPerM	4byte	Risoluzione verticale
Colors Used	4byte	Numero dei colori usati
Important Colors	4byte	Numero di colori importanti

# FORMATO BITMAP

## Campo InfoHeader



Data View																																	
Add...	Hexadecimal (1 Byte)																Text (ASCII)																
000000	42	4D	66	51	06	00	00	00	00	00	36	00	00	00	28	00	B	M	f	Q	*	*	*	*	*	*	*	6	*	*	*	(	*
000010	00	00	2C	01	00	00	CC	01	00	00	01	00	18	00	00	00	*	*	,	*	*	*	*	*	*	*	*	*	*	*	*	*	*
000020	00	00	30	51	06	00	00	00	00	00	00	00	00	00	00	00	*	*	0	Q	*	*	*	*	*	*	*	*	*	*	*	*	*
000030	00	00	00	00	00	00	15	05	06	15	05	06	14	04	05	14	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
000040	04	05	17	06	09	16	05	08	16	05	08	16	05	08	17	06	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
000050	09	17	06	09	16	05	08	16	05	08	16	06	0D	16	06	0D	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
000060	16	06	0D	18	06	0D	18	06	0D	17	05	0C	18	03	0B	17	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
000070	02	0A	18	05	08	1B	07	0C	1C	08	0D	1A	05	0D	17	05	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
000080	0C	18	07	0B	19	08	0B	17	06	09	19	08	0B	19	08	0B	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
000090	19	08	0B	19	08	0B	18	07	0A	18	07	0A	19	08	0B	19	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
0000A0	08	0C	1B	0A	15	19	08	13	18	07	10	18	07	10	17	07	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
0000B0	0E	14	04	0B	14	05	09	15	06	0A	17	05	0C	1A	08	0F	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

DIMENSIONE FILE

00 06 51 66 : 414054

INIZIO PIXEL

00 00 00 36 : 54

LARGHEZZA

00 00 01 2C : 300

ALTEZZA

00 00 01 CC : 460



# FORMATO BITMAP



Ogni riga deve essere un multiplo di 4. Qualora la larghezza di una immagine non sia un multiplo di 4 si inseriscono dei byte di riempimento (poi esclusi in visualizzazione)

Per le esercitazioni considerare solo immagini con dimensioni delle righe multiple di 4

I punti di immagine sono scritti a partire dal pixel in basso a destra dell'immagine per arrivare al pixel in alto a sinistra.

L'organizzazione dei bit che costituiscono i pixel sono organizzati in BGR (seguono la specifica *little endiann*)





# FORMATO BITMAP



## Lettura infoHeader

```
.data
FILEBMP:.asciiz "C:\\Users\\Franco\\Desktop\\MonaLisa.bmp"
s_filedim:.asciiz "\\nDIMENSIONE FILE: "
s_lar:.asciiz "\\nLARGHEZZA IMMAGINE: "
s_alt:.asciiz "\\nALTEZZA IMMAGINE: "
headerfull:.space 54
```

```
main:      .text
          .globl main

          la $a0,FILEBMP      #apertura file (creazione)
          li $a1,0            #file aperto per la lettura
          li $a2,0
          li $v0,13
          syscall
          move $s1,$v0        #descrittore del file (indirizzo del file in memoria)

          move $a0,$s1        #acquisizione headerfull
          la $a1,headerfull
          li $a2,54
          li $v0,14
          syscall

          move $a0,$s1        #chiusura FILEIN
          li $v0,16           #chiusura file
          syscall
```

# FORMATO BITMAP

## Lettura infoHeader

```
lbu $a0,headerfull+2  #lettura dimensione file
lbu $a1,headerfull+3
lbu $a2,headerfull+4
lbu $a3,headerfull+5
jal ordina             #ordinamento in big endian
move $s0,$v1           #dimesione file
lbu $a0,headerfull+18  #lettura larghezza
lbu $a1,headerfull+19
lbu $a2,headerfull+20
lbu $a3,headerfull+21
jal ordina             #ordinamento in big endian
move $s1,$v1           #larghezza
lbu $a0,headerfull+22  #lettura altezza
lbu $a1,headerfull+23
lbu $a2,headerfull+24
lbu $a3,headerfull+25
jal ordina             #ordinamento in big endian
move $s2,$v1           #altezza
```

ordina:

```
li $v1,0
move $v1,$a3
sll $v1,$v1,8
add $v1,$v1,$a2
sll $v1,$v1,8
add $v1,$v1,$a1
sll $v1,$v1,8
add $v1,$v1,$a0
jr $ra
```



# FORMATO BITMAP



## Lettura infoHeader

```
la $a0,s_filedim
li $v0,4
syscall
move $a0,$s0
li $v0,1
syscall
```

#stampa dimensione file

```
la $a0,s_lar
li $v0,4
syscall
move $a0,$s1
li $v0,1
syscall
```

#stampa larghezza

```
la $a0,s_alt
li $v0,4
syscall
move $a0,$s2
li $v0,1
syscall
```

#stampa altezza

```
li $v0,10
Syscall
```

The background is a dark blue gradient with a subtle, grainy texture. Overlaid on this are stylized circuit board traces and components. In the top-left and bottom-left corners, there are white traces and components, including what looks like a multi-pin connector and several small rectangular chips. In the top-right and bottom-right corners, there are yellow traces and components, including a large rectangular chip and several smaller components. The word "FINE" is centered in the middle of the image in a bold, yellow, sans-serif font.

**FINE**