X: monitor, adattatore grafico e frequenza dot-clock

Quando si vuole configurare XFree86 nelle versioni 3.* e qualcosa va storto, oppure non si riesce a ottenere quello che si vuole esattamente attraverso uno dei vari programmi già descritti nel capitolo precedente, può essere necessario mettere mano alle sezioni '**`Monitor`**', '**`Device`**' e '**`Screen`**' del file '`/etc/X11/XF86Config`'. Tra tutte, la sezione '**`Monitor`**' è la più difficile per il principiante, a causa delle direttive '**`Modeline`**' o '**`Mode`**', in cui si devono indicare una serie di numeri più o meno oscuri.

In questo capitolo si mostra in che modo calcolare i valori delle modalità video. Una scelta impropria di questi valori, potrebbe causare problemi, fino ad arrivare al danneggiamento del monitor. Si prega di intervenire con prudenza ed eventualmente anche di leggere *XFree86 Video Timings HOWTO* di Eric S. Raymond.

Autorilevamento con Read-edid

Read-edid [1] è un piccolo sistema di programmi in grado di scandire l'adattatore grafico e il monitor, allo scopo di ottenere le informazioni necessarie a configurare correttamente programmi come XFree86. Si compone precisamente di '**`get-edid`**' e di '**`parse-edid`**'.

Il programma '**`get-edid`**' esegue la scansione dell'adattatore grafico e attraverso di questo anche del monitor. Il risultato della scansione è un file binario emesso attraverso lo standard output, mentre attraverso lo standard error si ottengono altre informazioni diagnostiche. Per esempio, si può ignorare temporaneamente il risultato emesso dallo standard output per osservare tali notizie diagnostiche:

```
# get-edid 1> /dev/null [Invio]
```

Si potrebbe ottenere un risultato simile a quello seguente:

```
get-edid: get-edid version 1.4.1

        Performing real mode VBE call
        Interrupt 0x10 ax=0x4f00 bx=0x0 cx=0x0
        Function supported
```

```
        Call successful

        VBE version 102
        VBE string at 0xc098d "S3 Incorporated. Trio64V+"

VBE/DDC service about to be called
        Report DDC capabilities

        Performing real mode VBE call
        Interrupt 0x10 ax=0x4f15 bx=0x0 cx=0x0
        Function supported
        Call successful

        Monitor and video card combination supports DDC1 transfers
        Monitor and video card combination does not support DDC2 transfers
        0 seconds per 128 byte EDID block transfer
        Screen is blanked during DDC transfer

Reading next EDID block

VBE/DDC service about to be called
        Read EDID

        Performing real mode VBE call
        Interrupt 0x10 ax=0x4f15 bx=0x1 cx=0x0
        Function supported
        Call successful
```

In questo caso è stato individuato un vecchio adattatore grafico «Trio64V+».

Per poter leggere il risultato emesso attraverso lo standard error, si usa '**`parse-edid`**':

```
# get-edid 2> /dev/null | parse-edid [Invio]
```

Ecco quello che si potrebbe ottenere:

```
        # EDID version 1 revision 0
Section "Monitor"
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        Identifier "PHL:0012"
        VendorName "PHL"
        ModelName "PHL:0012"
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # DPMS capabilities: Active off:yes  Suspend:yes  Standby:yes

        Mode    "640x400"       # vfreq 70.072Hz, hfreq 31.462kHz
                DotClock        25.170000
                HTimings        640 656 752 800
                VTimings        400 412 414 449
                Flags   "+HSync" "-VSync"
        EndMode
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
        # Block type: 2:0 3:0
EndSection
```

In pratica, con questo risultato si può compilare la sezione '**`Monitor`**' del file di configurazione di XFree86.

Autorilevamento con un servente XFree86 versione 3.*

«

Quando non si conoscono tutte le caratteristiche del proprio adattatore grafico, è possibile utilizzare un servente X con l'opzione '**`-probeonly`**' per vedere cosa questo riesce a determinare da solo. Alcuni parametri sono sensibili al carico del sistema, per cui, questo tipo di prova deve essere fatto quando non si effettuano altre attività.

Qui si sta facendo riferimento all'uso di un servente XFree86 versione 3.*. Pertanto, il file di configurazione di partenza che viene proposto, rispecchia la sintassi relativa a quelle versioni e non può funzionare per una versione 4.*.

È il caso di utilizzare un servente più o meno generico, per esempio quello per gli adattatori SVGA ('**`XF86_SVGA`**'), che deve essere stato installato. Per stimolare l'autorilevamento, è necessario che le voci corrispondenti non siano presenti nel file di configurazione '`/etc/X11/XF86Config`' (o siano commentate). Un file come quello seguente, dove le sezioni '**`Monitor`**', '**`Device`**' e '**`Screen`**' sono quasi

vuote, dovrebbe andare bene per cominciare lo studio del proprio adattatore grafico:

```
Section "Files"
    RgbPath     "/etc/X11/rgb"
    FontPath    "/usr/lib/X11/fonts/misc/"
    FontPath    "/usr/lib/X11/fonts/Type1/"
    FontPath    "/usr/lib/X11/fonts/Speedo/"
    FontPath    "/usr/lib/X11/fonts/75dpi/"
    FontPath    "/usr/lib/X11/fonts/100dpi/"
EndSection

Section "ServerFlags"
    # DontZap
    # DontZoom
EndSection

Section "Keyboard"
    Protocol    "Standard"
    AutoRepeat  500 5
    Xkbkeycodes "xfree86"
    XkbTypes    "default"
    XkbCompat   "default"
    XkbSymbols  "en_US(pc102)+it"
    XkbGeometry "pc"
EndSection

Section "Pointer"
    Protocol    "microsoft"
    Device      "/dev/mouse"
    Emulate3Buttons
    Emulate3Timeout    50
EndSection

Section "Monitor"
    Identifier  "Monitor generico"
EndSection

Section "Device"
    Identifier  "SuperVGA"
EndSection

Section "Screen"
    Driver      "svga"
    Device      "SuperVGA"
    Monitor     "Monitor generico"
    Subsection "Display"
        Modes       "640x400" "640x480" "640x480.28" "800x600"
    EndSubsection
EndSection
```

Si avvia quindi '**`X`**', come utente '**`root`**', con l'opzione '**`-probeonly`**', salvando lo standard output e lo standard error in un file ('**`X`**' è un collegamento simbolico al file binario del servente grafico prescelto).

Purtroppo, è necessario tenere in considerazione che questo tipo di prove può modificare l'aspetto dei caratteri sullo schermo, o bloccarlo del tutto. Per cui, se non si hanno alternative, si rischia di dover riavviare il sistema.

```
# X -probeonly > /tmp/x.tmp 2>&1 [Invio]
```

Se tutto è andato bene, il file '`/tmp/x.tmp`' generato dal comando, dovrebbe contenere un risultato simile a quello seguente, che viene sezionato per descriverlo in dettaglio.

```
XFree86 Version 3.3.2 / X Window System
(protocol Version 11, revision 0, vendor release 6300)
Release Date: March 2 1998
        If the server is older than 6-12 months, or if your card is newer
        than the above date, look for a newer version before reporting
        problems.  (see http://www.XFree86.Org/FAQ)
Operating System: Linux 2.0.34 i686 [ELF]
```

La parte iniziale presenta la versione del servente e del sistema operativo utilizzato.

```
Configured drivers:
  SVGA: server for SVGA graphics adaptors (Patchlevel 0):
      NV1, STG2000, RIVA128, ET4000, ET4000W32, ET4000W32i,
      ET4000W32i_rev_b, ET4000W32i_rev_c, ET4000W32p, ET4000W32p_rev_a,
…
      s3_svga, ct65520, ct65525, ct65530, ct65535, ct65540, ct65545,
      ct65546, ct65548, ct65550, ct65554, ct65555, ct68554, ct64200,
      ct64300, generic
```

Segue quindi l'indicazione del tipo di servente grafico avviato (SVGA) e l'elenco di tutti i nomi degli adattatori grafici gestibili con questo.

```
(using VT number 7)
```

Il servente grafico utilizzerebbe (se avviato normalmente) il posto della console virtuale numero sette.

```
XF86Config: /usr/lib/X11/XF86Config
```

È stata letta la configurazione del file '`/usr/lib/X11/XF86Config`' (in questo caso si tratta di un collegamento simbolico a '`/etc/X11/XF86Config`').

Dopo questo punto segue un elenco di informazioni, in parte definite all'interno del file di configurazione e in parte determinate in modo automatico.

```
(**) stands for supplied, (--) stands for probed/default values
```

Le informazioni fornite attraverso il file di configurazione sono prefissate dal simbolo '**`(**)`**', mentre quelle predefinite o determinate dall'interrogazione dell'adattatore grafico, sono prefissate dal simbolo '**`(--)`**'.

```
(**) XKB: keycodes: "xfree86"
(**) XKB: types: "default"
(**) XKB: compat: "default"
(**) XKB: symbols: "en_US(pc102)+it"
(**) XKB: geometry: "pc"
(**) Mouse: type: microsoft, device: /dev/mouse, baudrate: 1200
(**) Mouse: buttons: 3, 3 button emulation (timeout: 50ms)
(**) SVGA: Graphics device ID: "SuperVGA"
(**) SVGA: Monitor ID: "Monitor generico"
(**) FontPath set to "/usr/lib/X11/fonts/misc/,…
```

Dato l'esempio proposto, le informazioni sulla tastiera, il mouse e i percorsi dei tipi di carattere, sono prelevati dal file di configurazione. In particolare, si osserva che da quel file, sono state prese in considerazione la sezione '**`Device`**' denominata '**`SuperVGA`**' e la sezione '**`Monitor`**' denominata '**`Monitor generico`**'.

```
(--) SVGA: PCI: S3 ViRGE/DX or /GX rev 1, Memory @ 0xe0000000
(--) SVGA: S3V: ViRGE/DXGX rev 1, Linear FB @ 0xe0000000
(--) SVGA: Detected S3 ViRGE/DXGX
(--) SVGA: using driver for chipset "s3_virge"
```

L'adattatore grafico è una scheda S3 ViRGE/DXGX, per la quale verrebbe utilizzato il driver '**`s3_virge`**'. Tuttavia, data la circostanza, converrebbe utilizzare un servente grafico differente per questo adattatore; precisamente '**`XF86_S3V`**'.

```
(--) SVGA: videoram:  4096k
(--) SVGA: Ramdac speed: 170 MHz
(--) SVGA: Detected current MCLK value of 42.955 MHz
(--) SVGA: chipset:  s3_virge
(--) SVGA: videoram: 4096k
(**) SVGA: Using 8 bpp, Depth 8, Color weight: 666
(--) SVGA: Maximum allowed dot-clock: 170.000 MHz
```

Seguono altre informazioni molto importanti, come la quantità di memoria video e la frequenza massima di dot-clock. Si osservi in particolare la profondità di colori indicata: 8 bit/pixel (8 bit per punto). L'informazione è preceduta dal simbolo '**`(**)`**' perché il tipo di servente grafico permette la gestione di un massimo di 8 bit/pixel (256 colori), per cui è questo il valore fissato, benché l'adattatore grafico permetta ben altri livelli di profondità.

Dot-clock

Una delle informazioni più delicate dell'adattatore grafico è la frequenza del cosiddetto ***dot-clock***. Il significato di questo parametro viene descritto più avanti, tuttavia è bene sapere subito che si può manifestare in modi differenti.

Nell'esempio mostrato, appare l'indicazione di un livello massimo.

```
(--) SVGA: Maximum allowed dot-clock: 170.000 MHz
```

In altre situazioni, può essere fornita una o più righe con un elenco di valori di dot-clock, come nell'esempio seguente:

```
(--) xxx: clocks:  25.0  28.0  40.0   0.0  50.0  77.0  36.0  45.0
(--) xxx: clocks: 130.0 120.0  80.0  31.0 110.0  65.0  75.0  94.0
```


In questo secondo caso, è necessario indicare la direttiva '**`Clocks`**' nella sezione '**`Device`**' del file '`/etc/X11/XF86Config`', come nell'esempio seguente:

```
Section "Device"
    …
    Clocks    25.0  28.0  40.0   0.0  50.0  77.0  36.0  45.0
    Clocks   130.0 120.0  80.0  31.0 110.0  65.0  75.0  94.0
EndSection
```

Quando invece la frequenza di dot-clock viene indicata solo come valore massimo (come nel caso dell'adattatore S3 ViRGE), non serve indicare alcuna direttiva '**`Clocks`**'.

Un po' di teoria

Alla base della costruzione dell'immagine da parte dell'adattatore grafico, sta la frequenza di dot-clock, ovvero la frequenza a cui ogni punto che la compone viene emesso. Questa è espressa in megahertz (MHz) e a volte deve essere selezionata da un elenco (quando si deve utilizzare la direttiva '**`Clocks`**'), altre volte può essere programmata liberamente, purché non venga superato il limite massimo.

In linea di massima, l'adattatore grafico VGA elementare tradizionale, ha una frequenza di dot-clock di 25175 MHz.

Chi lavora con l'informatica potrebbe essere portato a confondersi. In questo caso, megahertz significa esattamente milioni di hertz. Per cui, 25175 MHz sono esattamente pari a 25175000 Hz. Così, kilohertz rappresenta migliaia di hertz, per cui, per esempio, 31,5 kHz corrispondono a 31500 Hz.

A parità di condizioni, al crescere della risoluzione deve crescere la frequenza di dot-clock. Leggendo il contenuto standard di una vecchia versione 3.* del file '`/etc/X11/XF86Config`', si conoscono i valori minimi delle frequenze di dot-clock per le risoluzioni più comuni. Qui vengono riportate nella tabella u50.15.

Tabella u50.15. Frequenze minime di dot-clock in base alla risoluzione.

Risoluzione	Frequenza di dot-clock minima
640×480	25,175 MHz
800×600	36 MHz
1024×768 interlacciato	44,9 MHz
1024×768	65 MHz
1152×864 interlacciato	65 MHz
1152×864	92 MHz
1280×1024 interlacciato	80 MHz
1280×1024	110 MHz
1600×1200	162 MHz
1800×1440	230 MHz

Ampiezza di banda del monitor

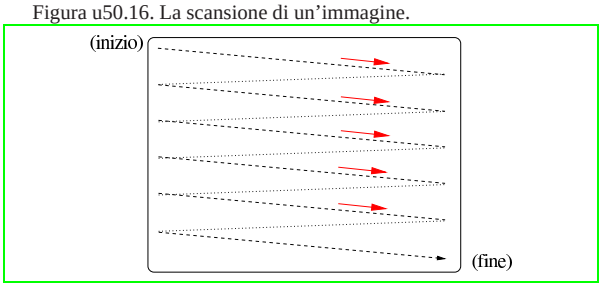
L'ampiezza di banda del monitor, o *bandwidth*, rappresenta la frequenza massima del segnale video che il monitor è in grado di gestire. Frequenze superiori vengono semplicemente filtrate, diventando particolari visivi non più percettibili.

In linea di principio, la frequenza di dot-clock utilizzata nell'adattatore grafico dovrebbe essere inferiore o uguale al valore massimo della frequenza del segnale video gestibile con il monitor, cioè al valore dell'ampiezza di banda.

Scomposizione e scansione dell'immagine sul monitor

L'immagine che appare sullo schermo di un monitor può essere descritta, in modo semplificato, come l'insieme di una serie di righe, composte a loro volta da punti. La prima forma di rappresentazione di un'immagine di origine elettronica è stata quella del tubo a raggi catodici e da questo tipo di tecnologia derivano le soluzioni adottate per la sua composizione.

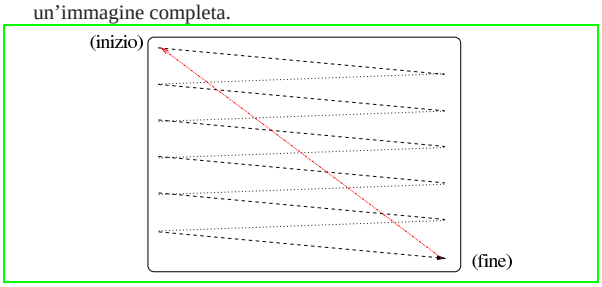
Figura u50.16. La scansione di un'immagine.



Le righe di un'immagine video vengono disegnate da un «pennello» ideale, che inizia la sua scansione in una posizione dello schermo in alto a sinistra, muovendosi verso destra e ricominciando sempre dal lato sinistro della riga successiva. Giunto alla fine dello schermo, riprende dalla posizione superiore sinistra.

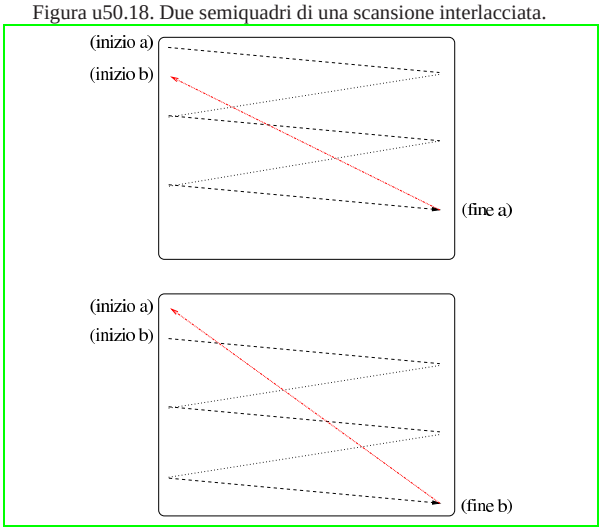
Il pennello di scansione, una volta che ha terminato una riga, prima di poter riprendere con la riga successiva, deve avere il tempo necessario per posizionarsi all'inizio di questa. Nello stesso modo, quando il pennello di scansione giunge alla fine dell'ultima riga, deve avere il tempo di ritornare all'inizio dello schermo, cioè nella posizione estrema in alto a sinistra.

Figura u50.17. Il ritorno all'inizio dopo la scansione di un'immagine completa.



Un'immagine completa è un ***quadro***, o *frame*, ma un quadro potrebbe essere ottenuto con un'unica scansione, dall'inizio alla fine dello schermo, oppure dalla somma di due ***semiquadri***. In questo ultimo caso si usa la tecnica dell'interlacciamento, in cui le righe dei due semiquadri si affiancano senza accavallarsi. La figura u50.18 mostra il caso di un quadro composto da un numero di righe pari.

Figura u50.18. Due semiquadri di una scansione interlacciata.

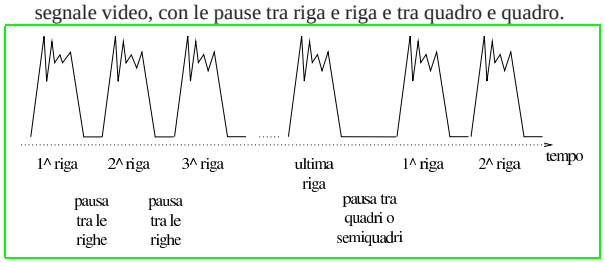


L'interlacciamento è nato come un metodo per ridurre lo sfarfallio dell'immagine nel sistema televisivo tradizionale. Per esempio, in Europa i quadri si susseguono a una frequenza di 25 Hz, un valore troppo basso perché l'occhio umano non si accorga dello sfarfallio. Così, attraverso l'interlacciamento, le immagini trasmesse vengono scomposte in due parti, visualizzate in sequenza a una frequenza di 50 Hz, considerata accettabile per quel tipo di utilizzo, anche se questo può comunque provocare strani effetti alla percezione dei particolari.

In generale, a parità di frequenza di quadro, è preferibile un'immagine interlacciata per ridurre l'effetto dello sfarfallio.

Da quanto affermato si può intendere che l'immagine video sia prodotta come una sequenza lineare di punti e di pause, necessarie al ritorno all'inizio di una riga successiva, di un quadro o di un semiquadro successivo.

Figura u50.19. Rappresentazione schematica dello scorrere del segnale video, con le pause tra riga e riga e tra quadro e quadro.



Il monitor su cui si visualizza il segnale video, deve avere un modo per sapere quando inizia un quadro e quando inizia ogni riga. Le pause necessarie al ritorno del pennello di scansione, vengono usate per sincronizzare la scansione stessa.

Frequenza, durata e lunghezza

La frequenza di dot-clock è una sorta di orologio che scandisce il tempo del segnale video. Un ciclo di questa frequenza rappresenta un punto dell'immagine. Questa frequenza si esprime in megahertz, per cui, una frequenza di dot-clock di 25,175 indica che in un secondo possono essere visualizzati 25 175 000 punti (si deve tenere presente che si tratta di valori teorici).

Seguendo questo ragionamento, le «misure» dell'immagine potrebbero essere valutate in quantità di dot-clock.

In tutto si utilizzano tre tipi di unità di misura per ciò che riguarda la composizione delle immagini: frequenze, riferite ai cicli di scansione delle righe e dei quadri; durate, riferite alle pause tra le righe e tra i quadri; lunghezze, pari alla traduzione di questi valori in unità di dot-clock.

Ricapitolando quanto già esposto nella sezione precedente, l'immagine video è composta da quadri che a loro volta si scompongono in righe. Le righe vengono scandite a una certa frequenza, definita come ***frequenza orizzontale***, e così anche i quadri, ***frequenza di quadro***. Queste frequenze possono essere tradotte in «lunghezze», riferite a unità di dot-clock. Per esempio, la frequenza orizzontale di 31,5 kHz (31 500 Hz), misurata con un dot-clock di 25,175 MHz, si traduce in una lunghezza di riga pari a 799,2 punti (25 175 000 / 31 500 = 799,2).

Quando si valutano grandezze riferite alla scansione verticale dell'immagine, per esempio la frequenza di quadro, si utilizzano lunghezze riferite al numero di righe. Continuando l'esempio precedente, se si aggiunge che la frequenza verticale è di 60 Hz, si determina che un quadro è composto da circa 419 583 dot-clock, pari a circa 525 righe.

Come già affermato, anche lo scorrere del tempo può essere valutato in unità dot-clock. Per esempio, un intervallo di tempo di 3,8 μs (microsecondi, ovvero milionesimi di secondo) è lungo 95,6 dot-clock (25 175 000 * 0,000 003 8 = 95,6).

Definizioni, concetti ed equazioni

La documentazione di XFree86 utilizza alcune definizioni che conviene elencare e chiarire. Le sigle indicate fanno volutamente riferimento a quelle utilizzate nel *XFree86 Video Timings HOWTO*.

- ***Frequenza di sincronizzazione orizzontale***, *Horizontal sync frequency*, HSF

La frequenza di scansione orizzontale delle righe sullo schermo. Generalmente si tratta di una grandezza espressa in kilohertz.

- ***Frequenza di sincronizzazione verticale***, *Vertical sync frequency*, VSF

  La frequenza di scansione verticale dei semiquadri, o dei quadri, sullo schermo. Quando l'immagine viene composta attraverso semiquadri interlacciati, si tratta della frequenza di scansione dei semiquadri stessi, altrimenti si tratta della stessa frequenza di scansione dei quadri interi.

- ***Frequenza di quadro***, ***frequenza di frame***, *Vertical refresh rate*, RR

  La frequenza di scansione verticale dei quadri interi (*frame*).

- ***Frequenza di dot-clock***, *Dot-clock frequency*, DCF

  La frequenza di dot-clock, espressa generalmente in megahertz.

- ***Ampiezza di banda video***, *Video bandwidth*, VB

  La frequenza massima per il segnale video accettato dal monitor. Questo valore dovrebbe essere maggiore o uguale a quello del dot-clock, ma anche valori inferiori a questo permettono ugualmente di vedere qualcosa. In generale, il livello minimo dell'ampiezza di banda deve essere almeno superiore alla metà della frequenza di dot-clock.

- ***Ampiezza orizzontale***, *Horizontal frame length*, HFL

  Si tratta della lunghezza totale di una riga, espressa in dot-clock. Questa dimensione deve includere la parte visibile e la pausa prima dell'inizio della riga successiva.

- ***Ampiezza verticale***, *Vertical frame length*, VFL

  Si tratta dell'altezza totale di un quadro intero, espressa in righe. Questa dimensione deve includere la parte visibile e la pausa prima dell'inizio del quadro successivo.

- ***Risoluzione orizzontale***, *Horizontal resolution*, HR

  La risoluzione orizzontale, espressa in punti o dot-clock, della parte visibile dell'immagine. Per definizione, si tratta di un valore inferiore dell'ampiezza orizzontale (HFL).

- ***Risoluzione verticale***, *Vertical resolution*, VR

  La risoluzione verticale, espressa in righe, della parte visibile dell'immagine. Per definizione, si tratta di un valore inferiore dell'ampiezza verticale (VFL).

Alcune equazioni elementari possono aiutare a collegare i vari pezzi del mosaico.

- HSF = DCF / HFL

  La frequenza di scansione orizzontale equivale alla frequenza di dot-clock divisa per la lunghezza completa della riga (espressa in dot-clock).

- RR = DCF / (HFL * VFL)

  La frequenza di scansione di un quadro intero equivale alla frequenza di dot-clock divisa per il prodotto della lunghezza completa della riga e il numero complessivo delle righe.

  La frequenza di sincronizzazione verticale è pari al doppio di RR quando si utilizzano semiquadri interlacciati, altrimenti corrisponde al valore di RR.

- DCF = RR * HFL * VFL

  Derivata dalla precedente. La frequenza di dot-clock si ottiene con il prodotto della frequenza di scansione di un quadro intero, la lunghezza completa della riga e il numero complessivo delle righe.

Multipli di otto e rapporto 3/4

Una particolarità comune dei valori che riguardano la risoluzione di un'immagine, è l'essere un multiplo di otto. Se si osserva, valori come 640×480, 800×600, 1024×768,... sono numeri divisibili per otto, senza lasciare alcun resto.

Un gran numero di adattatori grafici accetta determinati tipi di valori solo se sono multipli di otto. Per questo, in generale, per tutte le «lunghezze» orizzontali, quindi ciò che si esprime in punti o in dot-clock e riguarda la riga, si deve avere l'accortezza di usare multipli di otto. Questo particolare viene chiarito meglio in seguito.

Data la tradizione televisiva, il formato più comune della visualizzazione su monitor è 3/4, cioè la risoluzione verticale (il numero delle righe visibili) è il 75 % rispetto alla risoluzione orizzontale (il numero di punti visibili per riga). Questa regola non è obbligatoria. L'unico vincolo sono i multipli di otto per le grandezze che riguardano la scansione orizzontale.

Utilizzo della memoria video

L'immagine che appare sullo schermo di un monitor viene generata all'interno di una matrice contenuta in una memoria, che poi viene scandita nel modo che è stato spiegato. All'interno di questa memoria si deve conservare solo la parte di immagine visibile effettivamente, escludendo le pause inserite per facilitare il compito del pennello di scansione del monitor.

La memoria disponibile pone un limite alla risoluzione massima e alla ***profondità*** dell'immagine. A seconda del numero di colori o di sfumature che si vogliono rappresentare, deve essere impiegato un numero più o meno grande di bit per ogni punto dell'immagine. Se ***n*** è il numero di bit messo a disposizione per ogni punto, il numero di colori o sfumature disponibile è di 2^n. Nello stesso modo, conoscendo la memoria disponibile e la risoluzione che si vuole ottenere, si determina quanti siano i colori ottenibili per ogni punto.

Per esempio, se si dispone di 1 Mibyte, pari a 1048576 byte, cioè 8388608 bit, volendo ottenere una risoluzione (visibile) di 800×600 punti, si ottiene che per ogni punto sarebbero disponibili 17 bit (8388608 / (800 * 600)).

Tuttavia, di solito, il numero di bit che può essere utilizzato per definire la profondità di un'immagine è limitato a valori ben precisi: 2 bit (bianco/nero), 4 bit (16 colori), 8 bit (256 colori), 16 bit (64 Kicolori), 24 bit (16 Micolori), 32 bit (4 Micolori),...

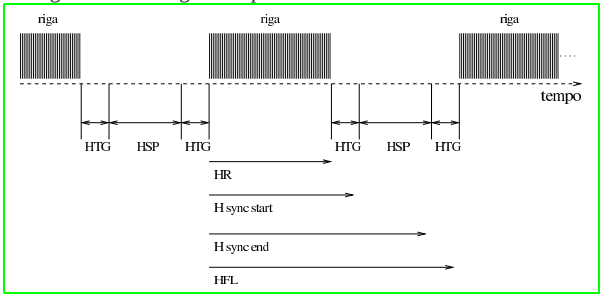
Si osservi che in alcune circostanze, vale anche per la profondità di colori la regola del multiplo di otto; per esempio, di solito si ha a che fare con profondità da 8 bit, 16 bit, 24 bit e 32 bit.

Impulsi di sincronismo

Fino a questo momento sono state descritte le immagini video come qualcosa formato da righe visibili, collegate tra loro da delle pause, a formare dei quadri (o dei semiquadri), i quali si collegano tra loro con delle pause più grandi. Quando si è accennato ai concetti di ampiezza orizzontale e verticale, si è sottolineato il fatto che queste grandezze devono includere anche le pause relative.

Ma le pause da sole non bastano. Al loro interno si inseriscono degli impulsi di sincronismo, senza i quali queste non sarebbero riconosciute dal monitor.

Figura u50.20. Righe e impulsi di sincronismo.



L'impulso di sincronismo orizzontale ha una durata che può variare da monitor a monitor, ma in ogni caso si esprime in un'unità di tempo che resta costante al variare della frequenza dot-clock. Generalmente sono accettabili valori compresi tra 3,5 μs e 4,0 μs (microsecondi). La figura u50.20 mostra schematicamente gli elementi che compongono una riga completa: la parte visibile, definita come risoluzione orizzontale (HR), un tempo di guardia precedente all'impulso di sincronismo (HTG), l'impulso di sincronismo (HSP), un tempo di guardia finale. Quindi ricomincia un'altra riga.

Il tempo di guardia iniziale e finale è importante come l'impulso di sincronismo, tuttavia viene definito normalmente in modo approssimativo, salvo aggiustamenti successivi. In generale, un tempo di guardia medio di 30 dot-clock dovrebbe andare bene. È importante osservare subito che di solito il tempo di guardia iniziale e finale non sono simmetrici.

In maniera analoga funziona il sincronismo verticale. Si ha un tempo di guardia iniziale (VTG, *Vertical time guard*), un impulso di sincronismo verticale (VSP) e un tempo di guardia finale. L'impulso di sincronismo dovrebbe oscillare tra i 50 μs e i 300 μs (microsecondi).

Tradurre i valori in unità dot-clock e in quantità di righe

La definizione dei vari elementi che compongono l'immagine deve essere fatta attraverso due unità di misura uniformi: dot-clock per ciò che riguarda la scansione orizzontale e righe per la scansione verticale.

Si è accennato al fatto che il tempo di guardia orizzontale può aggirarsi attorno a un valore di 30 dot-clock, senza bisogno di fare altri calcoli, mentre il problema si pone per trasformare il tempo dell'impulso di sincronismo in dot-clock. Basta moltiplicare la frequenza di dot-clock per il tempo. La frequenza è espressa in hertz e il tempo in secondi.

Lunghezza in dot-clock = DCF * tempo

Per riprendere un esempio già fatto, se si utilizza una frequenza di dot-clock di 25,175 MHz e si vuole misurare un intervallo di 3,8 μs, si ottiene una lunghezza di 95,6 dot-clock (25 175 000 * 0,000 003 8).

Il vero problema, quando si fa riferimento a grandezze orizzontali, è il fatto che queste devono essere espresse in multipli di otto. Molte approssimazioni nei calcoli relativi, che per il momento non sono ancora state mostrate, derivano da questa esigenza.

Il tempo di guardia verticale, a seconda del tipo di monitor utilizzato, potrebbe essere assente del tutto, oppure potrebbe essere richiesto un massimo di tre righe. Eventualmente, un tempo di guardia maggiore del necessario, non può essere dannoso.

Il calcolo della lunghezza dell'impulso di sincronismo verticale, in termini di righe, è un po' più complesso. Uno dei modi possibili è quello di definire prima la lunghezza in dot-clock e quindi di convertirla in righe, dividendo questo valore per la lunghezza complessiva della riga.

Lunghezza VSP = (DCF * tempo) / HFL

Riprendendo l'esempio precedente, aggiungendo che una riga ha la lunghezza complessiva di 800 dot-clock, volendo calcolare un impulso di sincronismo verticale di 64 μs circa, si ottengono due righe ((25 175 000 * 0,000 064) / 800).

Configurazione della sezione «Monitor» di «XF86Config»

Quanto descritto fino a questo momento serve per chiarire il significato delle direttive contenute nella sezione '**`Monitor`**' del file di configurazione di XFree86: '`/etc/X11/XF86Config`'. Viene proposto un esempio:

```
Section "Monitor"
    Identifier  "Monitor generico"
    HorizSync   31.5, 35.15
    VertRefresh 50-70

    # 640x400 @ 70 Hz, 31.5 kHz hsync
    Modeline "640x400"     25.175 640  664  760  800   400  409  411  450

    # 640x480 @ 60 Hz, 31.5 kHz hsync
    Modeline "640x480"     25.175 640  664  760  800   480  491  493  525

    # 800x600 @ 56 Hz, 35.15 kHz hsync
    Modeline "800x600"     36     800  824  896 1024   600  601  603  625
EndSection
```

Si deve osservare che ogni direttiva '**`Modeline`**', o la sua equivalente '**`Mode`**', contiene tutte le informazioni necessarie sul funzionamento del monitor in corrispondenza a quella particolare modalità. Questo significa che le direttive '**`HorizSync`**' e '**`VertRefresh`**' sono solo un'informazione aggiuntiva che serve a permettere un controllo incrociato. Per essere più precisi, il file '`/etc/X11/XF86Config`' potrebbe contenere informazioni su molte modalità di visualizzazione, che vengono selezionate in base alle limitazioni poste dalle direttive '**`HorizSync`**' e '**`VertRefresh`**'.

Scomposizione delle informazioni

La direttiva '**`Modeline`**' contiene una serie di notizie che è necessario distinguere per poterne conoscere il significato.

```
Modeline nome freq_dot_clock informazioni_scansione_orizz informazioni_scansione_ve
```

In particolare, le informazioni sulla scansione orizzontale e quelle sulla scansione verticale sono una coppia di quattro numeri distinti (otto in tutto).

```
#          nome          dot          scansione           scansione
#        modalità      clock        orizzontale          verticale
#                                 ------------------   ------------------
Modeline "640x480"     25.175  640  664  760  800   480  491  493  525
```

Le opzioni sono costituite da parole chiave che possono apparire in coda, in presenza di occasioni particolari, secondo quanto descritto nella documentazione del servente grafico che si utilizza.

È bene ripetere che la direttiva '**`Modeline`**' potrebbe essere sostituita con ***Mode***, una specie di sottosezione molto più leggibile.

```
Mode nome
    DotClock     frequenza_dot_clock
    HTimings     informazioni_scansione_orizzontale
    VTimings     informazioni_scansione_verticale
    [Flags       opzioni…]
EndMode
```

Segue l'esempio già mostrato sopra.

```
Mode "640x480"
    DotClock    25.175
    HTimings    640  664  760  800
    VTimings    480  491  493  525
EndMode
```

Scansione orizzontale

I quattro valori indicati nella direttiva '**`HTimings`**', o quelli che appaiono subito dopo la frequenza di dot-clock nella direttiva '**`Modeline`**', rappresentano i tempi della scansione orizzontale, espressi in unità di dot-clock.

```
risoluzione_orizzontale  inizio_sinc  fine_sinc  ampiezza_orizzontale
```

In pratica, seguendo l'esempio già mostrato, «640 664 760 800» indica che: la risoluzione orizzontale è di 640 punti, o dot-clock, l'impulso di sincronismo orizzontale inizia in corrispondenza del 664-esimo dot-clock e termina con il 760-esimo dot-clock, infine la lunghezza complessiva della riga è di 800 punti.

Con qualche conto si scopre che la frequenza orizzontale necessaria per la scansione con questa modalità è di 31,5 kHz

(25175000 / 800) e che la durata dell'impulso di sincronismo è di 3,8 μs ((760 − 664) / 25175000).

La cosa più importante da osservare è che tutti i valori sono divisibili per otto.

Scansione verticale

I quattro valori indicati nella direttiva '**VTimings**', o gli ultimi quattro valori della direttiva '**Modeline**', rappresentano i tempi della scansione verticale, espressi in quantità di righe.

risoluzione_verticale inizio_sinc fine_sinc ampiezza_verticale

In pratica, seguendo l'esempio già mostrato, «480 491 493 525» indica che: la risoluzione verticale è di 480 righe, l'impulso di sincronismo verticale inizia in corrispondenza della 491-esima riga (ideale) e termina con la 493-esima, infine l'altezza complessiva del quadro è di 525 righe.

Con qualche conto si scopre che la frequenza verticale (del quadro intero) necessaria per la scansione con questa modalità è di 70 Hz (25175000 / (800 * 525)) e che la durata dell'impulso di sincronismo è di 64 μs ((493 − 491) * 800 / 25175000).

La frequenza dei semiquadri è doppia, quando si utilizza una modalità interlacciata. Questo va tenuto in considerazione, perché è la frequenza dei semiquadri quella che viene presa in considerazione nella direttiva '**VertRefresh**'.

Interlacciamento

La predisposizione di una modalità interlacciata richiede solo due particolarità: che il numero complessivo delle righe (VFL) sia in numero dispari e che si aggiunga alla fine l'opzione '**Interlace**'.

```
# 1024x768 @ 87 Hz interlaced, 35.5 kHz hsync
Modeline "1024x768"  44.9 1024 1048 1208 1264   768  776  784  817 Interlace

# 1152x864 @ 89 Hz interlaced, 44 kHz hsync
ModeLine "1152x864"   65  1152 1168 1384 1480   864  865  875  985 Interlace

# 1280x1024 @ 87 Hz interlaced, 51 kHz hsync
Modeline "1280x1024"  80  1280 1296 1512 1568  1024 1025 1037 1165 Interlace
```

I valori riferiti alla scansione verticale si riferiscono sempre al quadro intero, per cui, la frequenza di sincronizzazione verticale risulta doppia rispetto alla frequenza di quadro (*refresh rate* o *frame rate*).

A questo si può aggiungere che la durata dell'impulso di sincronismo verticale dovrebbe essere doppia (o quasi) rispetto a quella necessaria in caso di scansione normale (non interlacciata).

Adattamento delle configurazioni predefinite

Il file di configurazione di XFree86, '`/etc/X11/XF86Config`', offre molti esempi validi di configurazione del monitor, ma non tutti i casi possibili e immaginabili. Uno degli elementi che può creare disturbo è proprio la frequenza di dot-clock.

È già stato spiegato che il servente grafico, usato con l'opzione '**-probeonly**', può dare tante informazioni utili sull'adattatore grafico utilizzato. Tra le altre cose, dovrebbe essere in grado di informare sulle frequenze di dot-clock disponibili. Quello che si vede dall'esempio è l'informazione sul dot-clock di un elaboratore portatile Zenith (Z*Star 433VL), ottenuto da un servente XFree86 versione 3.*.

```
(--) VGA16: clocks: 28.32 28.32 28.32 28.32
```

Potrebbe nascere un problema se si tratta di frequenze fisse che non corrispondono ad alcuna modalità predefinita del file di configurazione; proprio come nel caso dell'esempio.

Intuitivamente, si può cercare di adattare una modalità che abbia una frequenza di dot-clock abbastanza vicina. Osservando il file di configurazione predefinito si possono trovare queste due modalità.


```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     25.175 640  664  760  800   480  491  493  525

# 640x480 @ 72 Hz, 36.5 kHz hsync
Modeline "640x480"     31.5   640  680  720  864   480  488  491  521
```

Anche se non si conosce nulla delle caratteristiche del monitor (in questo caso è quello del portatile, un LCD) si può azzardare l'idea che delle frequenze orizzontali e verticali comprese tra i valori di questi esempi, non dovrebbero creare problemi (la frequenza orizzontale di 31,5 kHz è quella più bassa in assoluto rispetto a tutte le modalità predefinite). Si procede per tentativi.

Evidentemente, dagli esempi proposti, ci si accontenta di una risoluzione di 640×480 punti, quindi questi valori sono noti. Inoltre, si può decidere di mantenere le stesse frequenze di sincronizzazione verticale e orizzontale dell'esempio già visto che utilizzava una frequenza di dot-clock leggermente più bassa. Così facendo, la pausa tra una riga e l'altra dovrebbe aumentare, come probabilmente anche la pausa tra un quadro e l'altro.

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  ???  ???  ???   480  ???  ???  ???
```

Conoscendo la frequenza di scansione orizzontale, si calcola la dimensione complessiva della riga: 28320000 / 31500 = 899, ma questo numero deve essere divisibile per otto, così si sceglie il valore 896. Nello stesso modo si calcola il numero di righe complessivo che compone un quadro: (28320000 / 896) / 60 = 526,78, ma si sceglie di approssimare per difetto (al massimo, la frequenza verticale diviene leggermente più alta di 60 Hz).

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  ???  ???  896   480  ???  ???  526
```

Adesso è la volta di determinare la durata dell'impulso di sincronismo orizzontale. Dovrebbe essere di circa 4 μs: 28320000 * 0,000004 = 113 dot-clock. Il problema adesso è quello di trovare qualcosa di soddisfacente che sia divisibile per otto.

((896 − 640) − 113) / 2 = 71,5

640 + 71 = 711; il valore più vicino che sia divisibile per otto è 712.

712 + 113 = 825; il valore più vicino che sia divisibile per otto è 824.

896 − 824 = 72, che rende il tempo di guardia perfettamente simmetrico (è stato solo un caso).

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  712  824  896   480  ???  ???  526
```

Restano i dati sulla durata dell'impulso di sincronismo verticale. Dal momento che la differenza rispetto all'esempio di riferimento non è molto grande, si può provare un po' a occhio, salvo verificare con la calcolatrice.

```
# 640x480 @ 60 Hz, 31.5 kHz hsync
Modeline "640x480"     28.32  640  712  824  896   480  491  494  526
```

Con questi valori, l'impulso di sincronismo dura 95 μs ((494 − 491) * 896 / 28320000), perfettamente accettabile.

Volendo verificare la frequenza orizzontale e verticale per sicurezza, si ottengono 31,58 kHz e 60,08 Hz, valori leggermente differenti rispetto a quelli di partenza, ma sicuramente tollerabili.

Rifiniture

I valori che si calcolano a tavolino, non possono essere sempre perfetti al primo colpo. Se tutto va bene, può capitare che l'immagine appaia un po' troppo spostata rispetto al centro dello schermo. Di certo si possono utilizzare i controlli del monitor per spostarla, ma a volte non conviene esagerare, dovendo trovare un compromesso tra la visualizzazione di schermate a caratteri e l'uso di X.

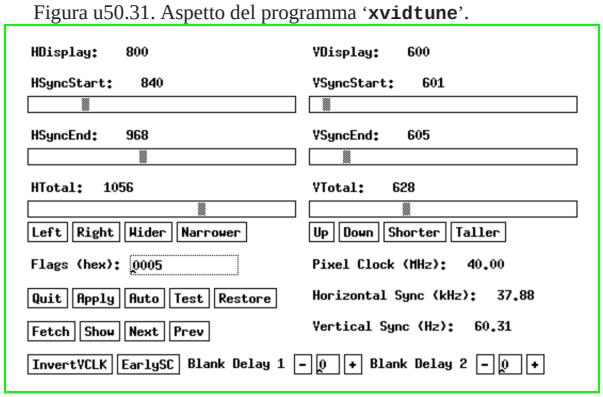
Quello che bisogna fare è ritoccare le dimensioni degli impulsi di sincronismo, oltre a cercare la loro collocazione ideale. Per questo però, viene in aiuto un programma apposito, che permette di verificare al volo valori differenti. Si tratta di '**xvidtune**'.

Utilizzo di «xvidtune»

Il programma '**xvidtune**'[2] permette di verificare la configurazione delle modalità video utilizzabili attraverso il servente X attivo. Generalmente viene avviato senza opzioni, ottenendo un funzionamento interattivo:

```
xvidtune [opzioni]
```

Figura u50.31. Aspetto del programma '**xvidtune**'.



Come si può osservare dalla figura, i controlli dal lato sinistro riguardano la scansione orizzontale, mentre quelli del lato destro quella verticale. In basso a destra si può tenere sotto controllo il valore della frequenza di dot-clock (*pixel clock*), della frequenza di sincronizzazione orizzontale e verticale.

Al posto di utilizzare le barre si scorrimento, si possono selezionare i pulsanti grafici corrispondenti all'azione che si vuole ottenere: LEFT dovrebbe spostare l'immagine a sinistra, RIGHT a destra, WIDER dovrebbe allargare l'immagine, e NARROWER dovrebbe restringerla.

Per verificare l'effetto delle modifiche, basta selezionare il pulsante grafico TEST.

I pulsanti grafici NEXT e PREV permettono di passare alla modalità grafica successiva (quella che si otterrebbe con la combinazione [*Ctrl Alt* +]) e precedente ([*Ctrl Alt* -]).

Altri programmi affini

- *modeline(1)* [3]

  Si tratta di un programma che calcola i valori da associare alla direttiva '**Modeline**', oppure '**Mode**', dei programmi che usano la grafica e devono sapere come gestire la scansione dell'immagine.

Riferimenti

- Eric S. Raymond, *XFree86 Video Timings HOWTO*
  http://www.linux.org/docs/ldp/howto/HOWTO-INDEX/howtos.html

[1] **Read-edid** software libero per la maggior parte GNU GPL

[2] **X** MIT più altre licenze per porzioni particolari di codice

[3] **modeline** GNU GPL