XFree86 Video Timings HOWTO 中文版

作者: Eric S. Raymond <esr@thyrsus.com>
譯者: 彭迅 <pxun@163.net>
4.4版,2000年3月13日翻譯完成日期: 2000年4月1日

本文描述了如何爲特定顯示卡/顯示器的組合編寫XFree86下的模式行.XFree86發行版現在包含了很好的工具,可以配置大多數的標準組合;本文的主要用途在於爲高性能顯示器或非常特殊的硬件手工調整模式行.它也可以幫助你使用kvideogen來生成模式行,或用xvidtune來修改不太適合你的顯示器的某種標準模式.

目錄

1 免責聲明

使用在這裡的材料,必須自行承擔風險.當設置顯示器超出廠商技術規范時,可能會同時傷害顯示器和你自己.閱讀12 (過度驅動你的顯示器)獲取詳細的警告信息.由於過度驅動顯示器而造成對你或你的顯示器的傷害是你的問題.

本文的最新版本可以在*Linux*文件計劃 <http://metalab.unc.edu/LDP>網頁找到.

請直接把評論,批評和改進建議發往*esr@snark.thyrsus.com.*請不要把尋求解決特殊顯示器問題的電子郵件發來,這樣做隻會耗盡我的時間並令你失望 — 關於這個主題我所知道的全都已經在這裡了.

譯注:若發現任何翻譯欠妥之處,請通知譯者*pxun@163.net.*但譯者對於因譯文或原文內容所造成的任何損失一概不負任何責任.

2 緒論

XFree86服務器允許用户配置其顯示子系統,從而鼓勵最優化地使用已有的硬件.本文意圖幫你學會怎樣生成你自己的時序值以最優化地使用你的顯示卡和顯示器.

我們將提出一種使系統運轉的方法,然後向你展示怎樣由基本設置開始逐步試驗,直到調出符合你口味的設置.

如果你已經有了一種基本可以工作的模式(特別是如果某種預定義的VESA模式能穩定地顯示,但另一種模式圖象向右或向左移位,或者圖象太小或太大)你可以直接閱讀這一章15 (解決圖象問題).這會幫助你修改時序值以達到特殊的效果.

如果安裝完後第一次進入X時顯示混亂,不要假定你需要對所有的模式設置進行調整;也許大部分模式行初始設置是正確的,隻是碰巧默認值是不適合你的硬件的一行.你應該按'CTRL'-'ALT'-'小鍵盤+'在所有已經安裝的模式間循環切換.如果其中一些模式看起來正常,嘗試除640x480的一行外注釋掉其他所有模式行,檢查這種模式是否工作.如果可以,那麼去掉其他幾種模式的注釋符,例如以顯示器能處理的頻率運行的某一800x600模式和某一1024x768模式.

更多的幫助正在著手進行.在剛剛發行的XFree86 4.0中許多驅動程序模塊支持DDC,即VESA的Display Data Channel(顯示數據通道)功能.當DDC激活後,顯示器告訴XFree86它能支持什麼模式行.因而你使用4.0和任何最近生產的顯示器的話很可能完全不必進行任何設置.

3 自動計算工具

如果你有一台支持即插即用規范的比較新的顯示器(1996年或以後的),就有機會使用read-edid程序來獲取顯示器的統計數據表並爲你算出模式行.請瀏覽 <http://altern.org/vii/programs/linux/read-edid/>.

從XFree86 3.2開始,XFree86提供一個**XF86Setup**(1)程序,能很簡單地交互式生成一種可以工作的顯示器模式,用不著直接對付麻煩的顯示時序值.因而在大多數情況下你不需要算出一種基本的顯示器模式.不幸的是,**XF86Setup**(1)有一些限制;它隻能識別最高1280x1024的標準顯示模式.如果你有一台性能非常高的顯示器,可以支持1600x1200或更高分辨率,仍需自行計算基本顯示器模式.

有一個KDE的工具叫*KVideoGen* <http://without.netpedia.net/kvideogen/>,能從顯示器和顯示卡的基本統計數據表中計算出模式行.我試驗過用它生成模式行,但沒有實際嘗試它們.注意它的Horizontal Refresh Rate(水平刷新率)和Vertical Refresh Rate(垂直刷新率)參數與我們下面描述的行頻(HSF)和場頻(VSF)含義是相同的.Horizontal Sync Pulse(行同步脈沖)值看來指以毫秒計算的一個同步脈沖的寬度,該工具假定上升沿HGT1和下降沿HGT2的值固定.如果你不知道行同步脈沖的值,使用缺省值比較安全.

最近版本的XFree86提供了一個工具叫**xvidtune**(1).你很可能發現在測試和調整顯示器模式時它相當有用.開始它會出現一段令人生畏的關於不正確使用可能導致結果的警告.如果你仔細地研究本文並學會xvidtune對話框漂亮的數字背後代表的東西,你將能有效地充滿信心地使用xvidtune.

如果你有**xvidtune**(1),就能快速地測試新的模式,不用修改X配置文件,甚至不用重啓動X服務器.另外,XFree86允許你在Xconfig定義的不同模式之間用熱鍵切換(查閱XFree.man獲得詳細資料).使用這種能力可以節省你的大量工作!當你要測試一種新模式時,給它一個唯一的模式標記並把它增加到熱鍵切換清單的末尾.留下一種已知正常的模式作爲默認值,當測試的模式不工作時可以退回去.

在接近本文結尾的地方,我們包含了一個'modeplot'腳本(script),你可以用來產生各種可選模式的模擬曲線圖.它不直接幫助你生成模式行,但它可以幫你更好地理解定義的參數之間的關系.

4 顯示器怎樣工作

了解顯示器怎樣工作對於理解在Xconfig文件不同的域應填入什麼值是不可缺少的.那些值被XFree86服務器用於對顯示器進行最底層控制.

顯示器用可以認爲是一連串的光柵點來生成一幅圖象.這些點從左到右排列形成行.這些行從上到下排列形成圖象.當顯示器內部三原色各發一束電子打在這些點上,就發出了光.爲了使電子束以相等的時間間隔打在每個點上,電子束以一種恆定圖案在屏幕上從左到右掃描,稱爲光柵.

我們說"可以認爲是一連串的光柵點"因爲這些光柵點並不實際對應物理熒光點.物理熒光點比光柵點更小 — 它們必須這樣,否則顯示器會有嚴重的水紋圖樣效果.光柵點其實是模擬驅動信號的採樣,顯示爲按格柵排列的點隻是因爲信號中的峰值和谷值是規則而精細地間隔的.

掃描圖案起始於屏幕的左上角,在屏幕上沿直線向右,移動時一直略微"下傾"(下傾坡度太小以至不可覺察).然後電子束返回屏幕的左邊,開始於新的一行.新的一行象第一行一樣從左到右掃描.這圖案不斷地重復直到顯示器最底一行掃描完.然後電子束從顯示器的左下角(往返掃描若幹次)移到右上角,該圖案又再次開始.

這種模式有一個變種叫隔行掃描:在第一個半幀僅僅每兩行中的一行被掃描,其余的行在第二個半幀才被掃描.

電子束在顯示器左上角開始掃描是一幀的開始.當電子束從顯示器右下角再次到達左上角時這一幀就結束了.一個幀是由電子束在顯示器上從上到下掃描過的所有的行構成的.

如果在掃描幀時電子束總是發出的,顯示器上所有的點都會被照亮.沿著屏幕的邊緣就不會有黑邊.在屏幕的邊緣圖象就會變形,因爲電子束很難控制那裡.爲了減少變形,沿著屏幕邊緣的點不會被電子束照亮(因爲電子束被關閉了),而且即使有電子束發出並指向它們仍不會照亮.這樣,顯示器的可視區域就減少了.

另一件要理解的重要事情是,當可視區域中沒有點被掃描時電子束會怎麼樣.電子束照在顯示器的左右邊緣的時間用於把電子束從右邊緣移回左邊緣.電子束照在顯示器的上下邊緣的時間用於把電子束從顯示器的右下角移回左上角.

顯示卡產生信號,使顯示器(依照想得到的顏色)發出電子束打在每個點上產生圖象.通過產生一個叫行同步脈沖的信號,顯示卡也控制何時顯示器把電子束從右邊移回左邊.在每一行結束時產生一個行同步脈沖.在每行結束時出現一個行同步脈沖.顯示卡還產生一個場同步脈沖通知顯示器把電子束移到屏幕左上角.在接近每一幀結束時產生一個場同步脈沖.

在行同步和場同步脈沖之前與之後,顯示器都需要有一段短的時間以讓電子束的位置能穩定.如果電子束不能穩定,圖象也將不穩定.

要了解更多信息,請瀏覽電視與顯示器偏轉系統 <http://fribble.cie.rpi.edu/~repairfaq/REPAIR/F_deflfaq.html>.

在下一章,我們將回來用定義,公式和范例來討論這些基本原理以幫助你使用它們.

5 了解關於顯示器和顯示卡的基本情況

在修改一項Xconfig條目前,有一些基本情況你需要了解.它們是:

- 顯示器的行頻和場頻選項
- 顯示器的頻寬
- 顯示卡的驅動時鐘頻率,或"點時鐘"

5.1 顯示器同步頻率

行頻就是每秒顯示器能水平掃描的行數;它是關於顯示器的一個最重要的統計數值.場頻是每秒顯示器電子束能在垂直方向來回移動的次數.

同步頻率通常列在顯示器手冊的技術規格頁中.場頻值典型地以Hz(每秒周期數)標定,行頻以KHz(每秒千周數)標定. 通常場頻的范圍在50到150Hz之間,行頻在31到135KHz之間.

如果你有一台多頻同步顯示器,其頻率會給出一個范圍.某些顯示器,尤其是低檔型號,有多個固定的頻率.它們也能進行配置,但你的可選項會嚴格受限於顯示器本身的技術特性.爲最佳分辨率配對選擇最高的頻率.並且要小心 — 試圖對一台固定頻率顯示器設置高於其設計規格的頻率會很容易地損壞它.

本指南的早期版本在過度驅動多頻同步顯示器方面相當積極,驅使它們超過其名義上的最高場頻以獲得更好的性能.那以後其他人向我們指出了更多理由告誡這種做法;我們將在下面章節12 (過度驅動你的顯示器)中論及.

5.2 顯示器的頻寬

顯示器的頻寬應該包括在手冊的技術規格頁中.如果沒有,看看顯示器的最高額定分辨率.作爲一個經驗法則,這裡指出怎樣把它們轉化爲頻寬的估計值(從而轉化爲你可以使用的點時鐘的大致上限):

640x480	25
800x600	36
1024x768	65
1024x768隔行	45
1280x1024	110
1600x1200	185

順便說一句,這張表格沒什麼神秘的;這些值隻是在標準XFree86模式數據庫中每種分辨率的最低點時鐘而已(最後的一個例外,是我用插值法補上的).實際上你的顯示器頻寬很可能高於它在最高分辨率下要求的最小值,所以不要擔心去嘗試一個高若幹MHz的點時鐘.

也請注意對約65MHz以下的點時鐘來說,頻寬很少是一項有影響的因素.對於一張SVGA顯示卡和大多數高分辨率顯示器而言,你怎麼也不會達到顯示器頻寬的限值.下列各項是例子:

型號	頻寬
NEC 4D	75MHz
Nano 907a	50MHz
Nano 9080i	60MHz
Mitsubishi HL6615	110MHz
Mitsubishi Diamond Scan	100MHz
IDEK MF-5117	65MHz
IOCOMM Thinksync-17 CM-7126	136MHz
HP D1188A	100MHz
Philips SC-17AS	110MHz
Swan SW617	85MHz
Viewsonic 21PS	185MHz
PanaSync/Pro P21	220MHz

甚至低檔顯示器在其額定分辨率通常也不會太受限於頻寬.NEC Multisync II是一個很好的例子 — 按照其技術規格甚至不能顯示800x600.它隻能顯示800x560.對這樣的低分辨率你不需要高的點時鐘或大的頻寬;或許你可以做到最好的是32MHz或36MHz,兩者離該顯示器的額定頻寬30MHz仍不太遠.

在這兩個驅動頻率下,你的屏幕圖象也許沒有理應達到的那麼清晰,但圖象質量顯然可以接受.如果NEC Multisync II頻寬能高於36MHz當然更好.但對普通任務象文本編輯而言這不是很關鍵的,隻要差別不要顯著到造成嚴重圖象變形即可(如果這樣你的眼睛會立刻告訴你的).

5.3 顯示卡的點時鐘

顯示卡手冊的技術規格頁通常會告訴你該卡的最大點時鐘(那是指每秒鐘顯示卡能寫到屏幕上的像素總個數).

如果手冊沒有這項信息,X服務器可以爲你查出.最近版本的X服務器全都支持 –probeonly 選項,可以顯示出這項信息然後退出,用不著眞正啓動X或改變顯示模式.

果你的X沒有 –probeonly 選項也不要擔心.即使X鎖住了你的顯示器,它仍會把一行關於時鐘和其它錯誤的信息顯示在標準輸出上.如果你把這條信息重定向到一個文件,它會被保存下來,即使你不得不重啓動以回到控制台.

探查的結果或啓動信息看上去應該類似以下示例之一:

如果你使用XFree86: (譯注:以下爲屏幕輸出,—-下爲屏幕輸出的翻譯和說明)

```
Xconfig: /usr/X11R6/lib/X11/Xconfig
         --------------------------
            X配置文件所在路徑

(**) stands for supplied, (--) stands for probed/default values
            --------                  ---------------------
            (**)代表提供值              (--)代表檢測值/缺省值

(**) Mouse: type: MouseMan, device: /dev/ttyS1, baudrate: 9600
                  --------          ----------            ----
                  鼠標類型          鼠標設備文件名        與串口通訊速率

Warning: The directory "/usr/andrew/X11fonts" does not exist.
         Entry deleted from font path.
(**) FontPath set to "/usr/lib/X11/fonts/misc/,/usr/lib/X11/fonts/75dpi/"
                      --------------------------------------------------
                                     字體路徑

(--) S3: card type: 386/486 localbus
                    ----------------
                     顯示卡總線類型

(--) S3: chipset:   924
                    ---
    芯片集 --- 這是確切的芯片型號;一種86C911之前的型號

(--) S3: chipset driver: s3_generic
                         ----------
                       芯片集驅動程序

(--) S3: videoram:  1024k
                    -----
         顯示卡上幀緩沖內存(俗稱顯存)的大小

(**) S3: clocks:  25.00  28.00  40.00   3.00  50.00  77.00  36.00  45.00
(**) S3: clocks:   0.00   0.00  79.00  31.00  94.00  65.00  75.00  71.00
                  ------------------------------------------------------
                              以MHz計算的可用的驅動頻率
```

```
(--) S3: Maximum allowed dot-clock: 110MHz
                                    ------
                                     頻寬

(**) S3: Mode "1024x768": mode clock =  79.000, clock used =  79.000
                                        ------                ------
                                            1024x768下模式時鐘

(--) S3: Virtual resolution set to 1024x768
                                   --------
                                虛擬屏幕分辨率

(--) S3: Using a banksize of 64k, line width of 1024
(--) S3: Pixmap cache:
(--) S3: Using 2 128-pixel 4 64-pixel and 8 32-pixel slots
(--) S3: Using 8 pages of 768x255 for font caching
```

如果你使用SGCS或X/Inside X:

```
WGA: 86C911 (mem: 1024k clocks: 25 28 40 3 50 77 36 45 0 0 79 31 94 65 75 71)
---  ------       -----         ------------------------------------------
 |     |            |                 以MHz計算的可用驅動頻率
 |     |            +--顯示卡上幀緩沖內存的大小
 |     +--芯片型號
 +--X服務器類型
```

注意:(如果可能的話)在機器卸下負荷後才進行檢測.因爲X是一個應用程序,它的時序循環會與磁盤活動沖突,造成上述的數值不準確.檢測幾次觀察數值是否穩定;如果不穩定,關閉一些進程直到數值穩定.鼠標守護進程,如果有的話,特別可能讓你失敗(鼠標守護進程對Linux用户而言是gpm,對SVr4用户而言是mousemgr).

爲了避免時鐘檢測的不準確性,你應該寫下時序值並把它們作爲時鐘屬性值放到你的Xconfig中 — 這樣抑制了時序循環並給予X一份可以嘗試的時鐘值的準確清單.使用上面例子的數據:

```
wga
        Clocks  25 28 40 3 50 77 36 45 0 0 79 31 94 65 75 71
```

在負荷高度變化的系統上,這也許可以幫你避免神秘的X啓動失敗.有可能X啓動時,因系統的負荷而獲得錯誤的時序值,於是不能在其配置數據庫中找到一項匹配的點時鐘 — 或找到錯誤的一項!

5.4 這些基本的統計數據控制什麼

顯示器的同步頻率范圍,與顯示卡的點時鐘一起,決定了可用的極限分辨率.但由驅動程序決定對硬件的潛力利用到什麼程度.一套出眾的硬件組合卻沒有同樣能令人滿意的設備驅動程序是對金錢的一種浪費.另一方面,有一個萬能的驅動程序但硬件能力相對較弱的話,你就能盡可能地挖掘硬件的潛力.這是XFree86的設計主導思想.

你應該把你用的點時鐘與顯示器的頻寬相匹配.然而,這是有很大彈性的 — 一些顯示器能超出頻寬標稱值30%運行.這裡的風險在於超出顯示器的額定場頻;我們會在下面詳細討論它們.

解頻寬使你能在各種可能的配置中作出更明智的選擇.它會影響顯示器的圖象質量(尤其是微小細節的清晰度).

6 對基本技術規格的解釋

這一章解釋上述技術規格的含義,和其它一些你需要了解的東西.首先是定義.接著在括弧中的是我們作計算時指代它的變量名.

行頻**(HSF)**

每秒水平掃描次數(見上述).

場頻**(VSF)**

每秒垂直掃描次數(見上述).重要性在於是刷新率的上限.

點時鐘**(DCF)**

更正式的表述,'驅動時鐘頻率';顯示卡的晶振或壓控振盪器的頻率 — 每秒它能發出的最大點數.

頻寬**(VB)**

你能饋入到顯示器視頻輸入且顯示內容仍可辨別時的最高頻率.如果你的顯示卡產生交替亮滅的圖案,它的最低頻率是點時鐘(DCF)的一半,所以理論上頻寬從 DCF/2 開始才有意義.然而,爲了在顯示器圖象中微小細節有可以接受的清晰的顯示,你不能讓它低於最高點時鐘太多,高於就更好.

幀長**(HFL,VFL)**

水平幀長(HFL)是顯示器電子槍掃描包括不可見的左右邊界在內的水平一行的點數.垂直幀長(VFL)是包括不可見的上下邊界在內的整個圖象的掃描行數.

屏幕刷新率**(RR)**

每秒你的屏幕重畫的次數(這也叫"幀率").頻率越高越好,因爲可以減少閃爍.60Hz就可以了,VESA標準的72Hz就更好.

這樣算出它的值

```
RR = DCF / (HFL * VFL)
```

注意分母中的乘積與顯示器的可視分辨率不一樣,一般稍微大點.我們將在下面講到細節.

通常隔行模式指出的頻率(象87Hz隔行)實際上是半幀率:對典型的顯示器整個屏幕看上去大概有那樣的閃爍頻率,但每一單行隻有一半的次數刷新過.

爲了計算的方便我們把隔行掃描以其全幀率(刷新率)計算,也就是43.5Hz.隔行模式的圖象質量要好於相同全幀率的逐行模式,但明顯差於全幀率等於其半幀率的逐行模式.

6.1 關於頻寬

顯示器制造商喜歡以高頻寬作宣傳因爲頻寬限制了屏幕的亮度和色彩變化的銳度.高頻寬意味著可以看到更小的細節.

顯示器用電子信號在你眼前呈現出圖象.這些信號一旦由數字形式轉變爲模擬形式,總是以波形形式輸入顯示器.它們可以看作由很多簡單波形組合而成,每一個簡單波形都有一個固定頻率,很多是在MHz范圍,例如,20MHz,40MHz,或甚至70MHz.顯示器的頻寬實際上是指它能不失眞地顯示的頻率最高的模擬信號.

我們來說,頻寬主要作爲一個能使用的最高點時鐘的近似截止點.

6.2 同步頻率和刷新率:

顯示器上的每一水平掃描行隻是掃描一幀長度的可視部分.在任何時刻屏幕上實際隻有一個點在亮,但在足夠快的刷新率下你眼睛的影像余輝使你能"看"到整個圖象.

下面是一些幫助理解的示意圖:

```
 ----------------------
|                      |      行頻是顯示
|->->->->->->->->->->  |      器電子束每
|                     )|      秒能描繪該
|<-----<-----<-----<---|      圖案的次數
|                      |
|                      |
|                      |
|                      |
|______________________|
 ----------------------
|        ^             |      場頻是顯示
|       ^ |            |      器電子束每
|       | v            |      秒能描繪該
|       ^ |            |      圖案的次數
|       | |            |
|       ^ |            |
|       | v            |
|       ^ |            |
|_______|_v____________|
```

記住實際的光柵掃描是靠得很緊的Z字形圖案;換言之,電子束從左到右同時從上到下移動.

現在你可以知道點時鐘和幀大小與刷新率的關系了.根據定義,一赫茲(Hz)指每秒一個周期.因此,如果水平幀長是HFL而垂直幀長是VFL,那麼覆蓋整個屏幕要有 (HFL * VFL) 個點.因爲根據定義顯示卡每秒發出DCF個點,所以很顯然顯示器的電子槍從左到右並從上到下往復掃過屏幕每秒 DCF /(HFL * VFL) 次.這就是你的屏幕刷新率,因爲它就是指你的屏幕每秒被更新(從而被刷新)的次數!

你必須理解這個概念,才能按適合你需要的某種方式更改分辨率設置防止閃爍.

對那些理解圖象優於理解文字的人,這裡有一幅圖:

```
        RR                                      VB
         |   min HSF                     max HSF |
         |    |             R1        R2  |      |
max VSF -+----|------------/----------/---|------+----- max VSF
         |    |:::::::::::/::::::::::/:::::\     |
         |    \::::::::::/::::::::::/:::::::\    |
         |     |::::::::/::::::::::/:::::::::|   |
         |     |:::::::/::::::::::/::::::::::\   |
         |     \::::::/::::::::::/::::::::::::\  |
         |      \::::/::::::::::/::::::::::::::| |
         |       |::/::::::::::/:::::::::::::::| |
         |        \/::::::::::/:::::::::::::::::\|
         |        /\:::::::::/:::::::::::::::::::|
         |       /  \:::::::/::::::::::::::::::::|\
         |      /    |:::::/:::::::::::::::::::::| |
         |     /      \::::/::::::::::::::::::::::| \
min VSF -+----/-------\--/-----------------------|--\--- min VSF
         |   /         \/                        |   \
         +--/----------/\------------------------+----\- DCF
           R1        R2  \                       |     \
                          min HSF                |    max HSF
                                                 VB
```

這是一幅普通的顯示器模式圖.圖上x軸顯示點時鐘(DCF),y軸代表刷新率(RR).圖上的填充區域描繪出了顯示器的特性:區域內的每一點都是一種可能的顯示模式.

標記爲'R1'和'R2'的直線代表一個固定的分辨率(例如640x480);它們畫出了怎樣通過點時鐘和刷新率的很多不同組合來實現一個分辨率.R2線代表了比R1更高的分辨率.

允許區域的上下邊界是代表場頻限值的水平線.頻寬是點時鐘的上限,因此由一條在右邊限制了特性區域的垂直線代表.

在16 (繪制顯示器特性)你能找到一個程序幫你爲自己的顯示器描繪與此類似的圖(但有X圖形界面,更好看).那一章還討論了有趣的部分;由行頻的限制而造成的邊界的引出.

7 配置系統時的權衡

用另外的方式看看我們在上文得出的公式

```
DCF = RR * HFL * VFL
```

換言之,你的點時鐘是固定的.你能根據這個點時鐘更改刷新率,水平分辨率或垂直分辨率.如果三項中的一項增大了,其它的一項或兩項必須減少.

注意,刷新率不能大於顯示器的最大場頻.因而,對於任何在一個特定點時鐘的特定顯示器,有一個不能強制低於的最小的幀長乘積.

在選擇你的設置時,記住:如果你把刷新率(RR)設置得太低,你的屏幕就會閃爍.讓它保持在60Hz以上.72Hz是VESA人機工程學標準.120Hz是熒光的閃爍頻率;如果你對此比較敏感,就要保持其高於此值.

閃爍會使眼睛很疲勞,即使人類眼睛有適應性而且各人對它的忍耐力有很大的差異.如果以90度視角面對著顯示器,使用黑色的背景和顏色對比鮮明的前景,低到中等亮度,至少45Hz你*才*會覺得舒適.

行下面的嚴格測試:用`xterm -bg white -fg black`打開一個純白背景和黑色前景的xterm,使它擴大到覆蓋整個可視區域.現在把顯示器的亮度調到最大值的3/4,臉從顯示器轉開.試著從旁窺視顯示器(發揮敏感度更高的外圍視網膜細胞作用).如果你沒有感到任何閃爍或閃爍程度是可容忍的,那麼這個刷新率對你是適宜的.否則你最好設置一個更高的刷新率,因爲這種半可視的閃爍會迅速使你眼睛疲勞並讓你覺得頭疼,即使屏幕在常規視角看上去是正常的.

對隔行模式而言,閃爍的情況依賴於更多的因素,例如當前的垂直分辨率和實際屏幕內容.所以直接試驗一下吧.可是你不會希望低於約85Hz半幀率的.

所以讓我們認爲你已經選了一個可接受的最低刷新率.在選擇水平幀長(HFL)和垂直幀長(VFL)時,你將有一些機動的空間.

8 存儲器需求

可用的幀緩沖內存會限制在彩色或灰度級顯示器上能達到的分辨率.在隻有黑白兩色而其間沒有灰度深淺變化的顯示器上它大概沒有什麼影響.

對256色顯示而言,每個可視點需要1字節顯示內存.這個字節包含了決定該點怎樣由紅綠藍組合生成的信息.要知道需要的顯存數量,用每行可視點數乘以可視行數.對顯示分辨率1024x768而言, $1024 * 768 = 786432$,就是顯示器的可視點數.在每點1字節的情況下,這也是顯示卡需要的顯示內存字節數.

因此,顯示內存需求一般就是 (HR * VR) / 1024 K字節並向上舍入(此例中精確值是768K).如果你的顯存多於實際需求,就有額外的顯存用於虛擬屏幕滾動.

然而,如果顯示卡上隻有512K顯存,那麼就不能使用該分辨率.即使你有好的顯示器,沒有足夠的顯示內存,還是不能發揮顯示器的潛力.另一方面,如果你的SVGA顯示卡有1兆顯存,但你的顯示器至多能顯示800x600,那麼無論如何你也達不到高分辨率(查閱13 (使用隔行模式)找一個可能的補救方法).

如果你的顯存多於需求請不要擔心;XFree86會通過允許你滾動屏幕可視區來利用這些顯存(查閱Xconfig文檔關於虛擬屏幕大小的參數部分).也請記住一塊512K顯存的顯示卡實際安裝的顯存不是512,000字節,而是 $512 * 1024 = 524{,}288$ 字節.

如果你正在用一塊S3顯示卡運行X/Inside,並且樂意忍受16色(每象素4位),可以在Xconfig中設置色彩深度爲4,有效地把顯示卡能處理的分辨率加倍.例如,正常能運行1024x768x256的S3顯示卡,你能使其運行在4位顏色深度的1280x1024x16下.

9 計算幀的大小

警告:這種方法是爲多頻同步顯示器研究出來的.它或許也能用於固定頻率顯示器,但不保証一定可以!

開始時通過點時鐘(DCF)除以你可用的最高行頻(HSF)來得到一個水平幀長.

例如;假設你有一塊Sigma Legend SVGA卡,點時鐘是65MHz,你的顯示器行頻是55KHz.那麼(DCF / HSF)得到的數字是1181(65MHz = 65000KHz ; 65000 / 55 = 1181).

現在是我們巫術的第一步.你要把這個數字舍入爲最接近的8的倍數.對於SVGA和S3顯示卡使用的VGA硬件控制器這步必須進行;它使用一個8位寄存器,再左移3位,因此實際是一個11位數字.其它類型的顯示卡

如ATI 8514/A也許沒有這個要求,但我們不知道,而且這個修正不會有危害.所以把合用的水平幀長值向下舍入爲1176.

個數字(DCF / HSF 舍入爲8的倍數)是你可用的最小水平幀長(HFL).通過設置同步脈沖產生更低的行頻(HSF)你能獲得更長的水平幀長(HFL)(從而,屏幕上可有更多的水平點).但代價是更低的刷新率和更易覺察的閃爍.

作爲一個經驗法則,水平幀總長的80%可用於水平掃描線的可視部分 — 水平分辨率(概略而言這允許了邊緣和後掠時間 — 這是電子束從屏幕的右邊緣移到下條光柵線的左邊緣需要的時間).在這個例子裡水平分辨率是944.

現在,要獲得4:3的正常屏幕高寬比,把你的垂直分辨率設爲你剛才算出的水平分辨率的3/4.對這個例子而言是708.要算出你實際的垂直幀長(VFL),把它乘以1.05得到743.

4:3不是技術上的規定;如果一個不同的比率能最好地利用你的屏幕實際尺寸的話,你完全可以使用.但4:3使從對角線長度算出幀高和幀寬變得很方便,你隻要用對角線乘以0.8得到寬,乘以0.6得到高.

所以, HFL = 1176 而 VFL = 743 .用65MHz除以兩者的乘積得到了一個不錯的,健康的74.4Hz刷新率.好極了!優於VESA標準!啓動X時你得到944x708,多於預期的800x600.完全不賴!

你甚至能更進一步地改進刷新率,達到接近76Hz,因爲顯示器常常以高於額定大約2KHz的頻率進行行同步而且稍微降低垂直幀長(VFL)(換言之,在上例中垂直分辨率小於944的75%).但在嘗試這種"過度驅動"的機動之前,如果你眞的要進行,確認顯示器電子槍場頻能達到76Hz.(例如,流行的NEC 4D不行.它的場頻(VSF)隻能達到75Hz).(查閱12 (過度驅動你的顯示器)獲得這個問題更全面的討論.)

迄今爲止,上述內容大部分是簡單的算術和關於光柵顯示器的基本事實.幾乎完全沒有任何巫術!

10 巫術和同步脈沖

好,現在你已爲你選的點時鐘算出了 HFL / VFL ,找出了可接受的刷新率,檢查了你有足夠的顯示內存.現在是眞正的巫術 — 你要知道什麼時候在哪裡放置同步脈沖.

同步脈沖實際上控制了顯示器的行頻和場頻.你從技術規格表裡查出的行頻(HSF)和場頻(VSF)是名義上的,近似的同步頻率最大值.來自顯示卡的同步脈沖信號告訴顯示器實際以多快頻率運行.

回憶起上面的兩幅圖嗎?光柵掃描一幀需要的時間隻有部分用於顯示可視圖象(即屏幕分辨率).

10.1 行同步:

根據之前的定義,水平掃描一行有HFL個點.讓我們把可視點數目(屏幕水平分辨率)稱爲HR.因此根據定義顯然 HR < HFL .具體地,讓我們假設兩者同時開始,如下所示:

```
|--- -- -- -- -- -- -- -- -- -- -- -- --
|_ _ _ _ _ _ _ _ _ _ _ _ _              |
|_______________________|_______________|_____
0                       ^               ^      單位:點數
                        |   ^       ^   |
                        HR  |       |  HFL
                        |   |<----->|   |
                        |<->|  HSP  |<->|
                        HGT1         HGT2
```

現在,我們想要在圖示長度爲HSP的位置,也就是在可視區域末點和整個幀末點之間的位置放置一個同步脈沖.爲什麼要這樣?因爲如果我們能這樣放置的話,那麼屏幕圖象就不會偏移向左或右.它會在屏幕上期望的位置,方形地覆蓋顯示器的可視區域.

此外,在同步脈沖的每一邊我們都要約30個點作爲"防護時間".這由HGT1和HGT2表示.在一個典型的配置中HGT1不等於HGT2,但如果你正在從零建立一個配置,就要從兩者相等開始你的試驗(換言之,從同步脈沖居中開始試驗).

同步脈沖放置錯誤的癥狀是圖象在屏幕上偏移,一條邊過分地寬,另一邊圖象在屏幕邊緣回繞過來,產生一條白色邊緣和一條"疊影"帶.一個位置不適當的場同步脈沖會使圖象象一台錯誤調整了場同步控制的電視機一樣滾動(實際上,這是相同的現象).

幸運的話,你的顯示器同步脈沖寬度會在技術規格文檔中.如果沒有,眞正的巫術開始了...

在這一部分,你將不得不反復試驗.但大多數時候,我們可以安全地假定一個同步脈沖長度大約是3.5到4.0微秒.

具體起見,我們認爲行同步脈沖(HSP)是3.8微秒(順便說一句,這是一個不錯的試驗起始值).

現在,用上述的65MHz時序值,我們知道行同步脈沖(HSP)等於247個點(= 65 * 10**6 * 3.8 * 10^-6) [記得M=10^6, micro=10^-6]

一些人喜歡引用他們的水平幀參數而不是點數寬度作爲時序.你也許會看見下列術語:

活躍時間 **(HAT)**
: 相當於HR,但按時間單位(通常用微秒)計算. HAT * DCF = HR.

消隱時間 **(HBT)**
: 相當於(HFL - HR),但按時間單位(通常用微秒)計算. HBT * DCF = (HFL - HR).

上升沿 **(HFP)**
: 這就是HGT1.

同步時間
: 這就是HSP.

下降沿 **(HBP)**
: 這就是HGT2.

10.2 場同步:

回到上圖,我們怎樣能按圖示放置這247個點呢?

用我們的例子,水平分辨率(HR)是944而水平幀長(HFL)是1176.兩者之差是 $1176 - 944 = 232 < 247$!顯然這裡我們不得不做些調整.我們怎麼做呢?

首先把1176提高到1184,把944降低到936.現在差是 $1184 - 936 = 248$.嗯,接近些了.

下一步,我們不用3.8而用3.5計算行同步脈沖(HSP);那麼,我們得出 $65 * 3.5 = 227$.看起來更好了.但248不比227高多少.通常在水平分辨率(HR)和同步脈沖(SP)的開始點之間要有約30個點,同步脈沖(SP)的結束點和水平幀長(HFL)之間也一樣.而且它們必須是8的倍數!我們被卡住了嗎?

不.讓我們這麼做, $936 \% 8 = 0$,而 $(936 + 32)$

但現在我們有潛在的麻煩了,同步脈沖不再位於h和H的正中.幸虧用計算器我們發現 1232 - 32 = 1200 也是8的倍數,而且 (1232 - 32) - 968 = 232 ,相應於使用3.57微秒長的同步脈沖,仍然合理.

外, 936 / 1232   0.76或76%,仍然離80

此外,使用當前的水平幀長,我們要求顯示器同步於52.7KHz(= 65MHz / 1232),在其承受范圍之內.沒問題.

用前面提過的經驗法則, 936 * 75% = 702 ,這是我們新的垂直分辨率. 702 * 1.05 = 737 ,我們新的垂直幀長.

屏幕刷新率= 65MHz / (737 * 1232) = 71.6Hz .依然很好.

畫出的場同步脈沖布局圖是類似的:

```
|___ __ __ __ __ __ __ __ __ __ __ __ __
|_ _ _ _ _ _ _ _ _ _ _ _               |
|______________________|_______________|_____
0                     VR              VFL      單位:點數
                       ^   ^       ^
                       |   |       |
                       |<->|<----->|
                        VGT    VSP
```

我們在垂直可視區剛結束時開始同步脈沖.VGT是同步脈沖的垂直防護時間.大部分顯示器VGT爲0(沒有防護時間)時都沒問題.我們在這個例子裡就那樣設定.少數需要兩或三個點的防護時間,加上這個通常沒有害處.

回到本例:根據幀長的定義,一個垂直點時鐘是跟蹤完整的一個水平幀的時間,因此在我們的例子裡,它是 1232 / 65MHz = 18.95us .

經驗告訴我們一個場同步脈沖應該在50us和300us范圍之間.讓我們用150us做一個例子,並換算成8個垂直點時鐘(150us / 18.95us   8).

一些人喜歡引用他們的垂直幀參數而不是點數寬度作爲時序.你也許會看見下列術語:

活躍時間 **(VAT)**

相當於VR,但以毫秒爲單位. VAT * VSF = VR .

消隱時間 **(VBT)**

相當於 (VFL - VR) ,但以毫秒爲單位. VBT * VSF = (VFL - VR) .

上升沿 **(VFP)**

這就是VGT.

同步時間

這就是VSP.

下降沿 **(VBP)**

這是在場同步脈沖後的又一個防護時間.它通常是零.

11 匯總以上討論

Xconfig文件的視頻模式表包含多行數字,每一行都是與X服務器的某一種模式對應的一份完整的技術規格.各個域歸類爲四段,名稱段,時鐘頻率段,水平段和垂直段.

稱段包含一個域,是該行其余內容要説明的顯示模式的名稱.該名稱在Xconfig文件中圖形驅動程序設置段的"Modes"行中被引用.如果前面一行的名稱與當前行一樣,名稱域可以忽略.

點時鐘段僅包含顯示模式行的點時鐘(我們前面稱作DCF)域.該域中的數字指出用於生成隨後各段中數字的點時鐘值.

水平段由四個域組成,指出顯示器上每一水平行如何生成.本段的第一個域包含每行被照亮形成圖象的點數(我們前面叫作HR).本段的第二個域(SH1)指出行同步脈沖在哪個點開始.第三個域(SH2)指出行同步脈沖在哪個點結束.第四個域指出水平幀總長(HFL).

垂直段也包含四個域.第一個域包含在顯示器上看到的可視行數(VR).第二個域(SV1)指出場同步脈沖在哪行開始.第三個域(SV2)指出場同步脈沖在哪行結束.第四個域包含垂直幀總長(VFL).

例子:

```
#模式名稱     時鐘        水平時序             垂直時序

"752x564"    40     752 784  944 1088  564 567 569 611
             44.5   752 792  976 1240  564 567 570 600
```

(注意:以前的X11R5不支持帶小數的點時鐘值.)

對Xconfig而言,剛才提及的所有數字 — 行中發亮的點數,把發亮點與同步脈沖的起始處分隔開的點數,代表脈沖持續時間的點數,和在同步脈沖結束處之後的點數 — 加起來生成每行的點數.水平點數必須能被8整除.

水平值示例: 800 864 1024 1088

示例行中首先是發亮點數(800),跟著是同步脈沖起始點位置(864),跟著是同步脈沖結束點位置(1024),跟著是水平行最後一點的位置(1088).

再次提醒注意,所有水平值(800,864,1024,和1088)要能被8整除!垂直值不需要這樣.

顯示器從上到下的行數構成了幀.行是一幀的基本時序信號.許多行組成了圖象.在發亮的最後一行顯示了之後和場同步脈沖產生之前,有若幹行的延遲.然後同步脈沖會持續若幹行,最後本幀的末尾幾行將被生成,以作爲脈沖之後必須的延遲.特定顯示模式運行時要指定的數字值會以一種與下述例子類似的方式被輸入.

垂直值示例: 600 603 609 630

該例子指出在顯示器上有600可視行,場同步脈沖開始於第603行,結束於第609行,以及總計有630行.

注意垂直數字值不需要能被8整除!

讓我們回到原先討論的例子.綜上所述,現在我們要做的就是把我們的結果按下面格式寫到Xconfig中:

```
<名稱>   DCF     HR  SH1 SH2   HFL   VR  SV1 SV2 VFL
```

其中SH1是行同步脈沖的開始點而SH2是結束點;同樣地,SV1是場同步脈沖的開始點而SV2是結束點.

要輸入這些值,回憶一下上面關於巫術和同步脈沖的討論.SH1是行同步脈沖上升沿的開始點;因此, SH1 = HR + HGT1 .SH2是下降沿的結束點;因此, SH2 = SH1 + HSP .同樣地, SV1 = VR + VGT (隻是VGT通常爲0)而 SV2 = SV1 + VSP .

```
# 名稱     時鐘        水平時序             垂直時序        標志
 936x702   65       936 968 1200 1232   702 702 710 737
```

不需要特殊標志;因爲這是一種逐行模式.現在我們眞正地做完了.

12 過度驅動你的顯示器

如果顯示器是固定頻率的,你絕對不應試圖超越其掃描頻率.那樣做的話你會損害你的硬件.你也應知道過度驅動一台多頻同步顯示器可能會有潛在的不易覺察的問題.

相反地,象素時鐘高於顯示器的最大頻寬一般是無害的.超過了額定最大同步頻率時會有問題.某些新型顯示器也許有保護電路,發出危險的掃描頻率時關閉顯示器,但不要依賴它.尤其是一些舊的多頻同步顯示器(象Multisync II),隻使用一個水平變壓器.這類顯示器對過度驅動沒有作太多的保護.在你必須有高壓調節電路(在固定頻率顯示器中可以沒有)的同時,它不會覆蓋每個可能的頻率范圍,尤其是在便宜的型號中.這不僅意味着電路中有更多損耗,它也會使屏幕熒光粉更快老化,和使顯示器發射出高於規格的輻射(包括X射線).

然而,這裡最成問題的是顯示輸出驅動器的轉換速率(視頻信號的陡峭度),它通常獨立於實際的象素頻率,但與顯示卡的最大象素頻率相關(如果你的顯示卡廠商關心這個問題的話).

所以要小心...

13 使用隔行模式

(這一章很大程度上歸功於David Kastrup <dak@pool.informatik.rwth-aachen.de>)

如果顯示器場同步電路能穩定地支持一個固定的點時鐘的話,一台隔行顯示器的閃爍要比一台逐行顯示器較不顯著.就是因爲這個所以首先發明的是隔行模式.

隔行模式得到壞名聲是因爲在相同場頻即VSF(在廣告中經常給出的參數)下劣於對應的逐行模式.但在相同行頻下隔行模式明顯較爲優越,而行頻通常是你的顯示器/顯示卡決定性的限制所在.

在固定刷新率(或半幀率,或場頻VSF)下,隔行顯示器會更爲閃爍:一台90Hz的隔行顯示器會次於一台90Hz的逐行顯示器.然而,它隻需要一半的頻寬和一半的行頻.如果你在相同的點時鐘和相同掃描率下和逐行模式比較,它有顯著的優越性:45Hz的逐行模式是無法忍受的.在90Hz的隔行模式下,我的Multisync 3D顯示器(在1024x768)已經工作了幾年,感覺非常滿意.我猜想需要至少70Hz的逐行顯示器才能同樣舒適.

盡管如此,你要注意少數要點:隻在高分辨率下使用隔行模式,因而交替變亮的掃描線是彼此靠緊的.你也許要與同步脈沖的寬度和位置打交道以獲得最穩定的掃描線位置.如果掃描線是交替明暗的,交錯的圖象會跳出來.我有一個應用程序選了這樣一個點圖案作爲菜單背景(XCept,幸運的是據我所知沒有其它應用程序是這樣).爲了使用XCept,我切換到800x600,因爲不這樣的話它眞的傷害我的眼睛.

爲了相同原因,使用至少100dpi字體,或其它水平筆畫至少爲兩條掃描線寬的字體(對於高分辨率,別的字體都沒有意義).

當然了,當你的硬件在相近的刷新率下支持逐行模式的話就不要使用隔行模式.

可是,如果你發現在某些分辨率下你把顯示器或顯示卡推到了它們的上限,仍得到令人不滿的閃爍或水沖蝕般(超出頻寬)的圖象,你也許想要嘗試在相同分辨率下使用隔行模式.當然如果你的顯示器場頻(VSF)已經接近其極限這也是沒用的.

設置隔行模式很容易:象逐行模式一樣進行.隻需多考慮兩個事項:垂直行總數(在模式行的最後一個數)需要是一個奇數,以及當你指定"Interlace"標志時,顯示器的實際垂直幀率加倍.如果你指定的模式看來象45Hz模式但有"Interlace"標志,你的顯示器必須支持90Hz幀率.

舉一個例子,這是1024x768隔行模式下我的模式行:我的Multisync 3D能支持高至90Hz場頻和38KHz行頻.

```
Modeline "1024x768" 45 1024 1048 1208 1248 768 768 776 807 Interlace
```

在該模式下兩個限制都恰當地基本用盡了.指定相同模式,隻是沒有"Interlace"標志,仍然幾乎達到顯示器水平能力的極限(嚴格來說,稍微低於場頻的下限),但圖象產生了無法忍受的閃爍.

基本設置規則:如果你設置一種模式,小於顯示器垂直能力的一半,使垂直行總數爲奇數並加入"Interlace"標志.大多數情況下顯示質量將會極大改進.

除非你使用逐行模式,否則用盡你的顯示器的技術規格限制(技術規格中列出的場頻一般低於顯示器最大值約30%或更多)並手動設置一種隔行模式(或許用稍微高些的分辨率)會帶來較好的效果,但我不能對此承諾.

14 問與答

問:你舉的例子不是標準的屏幕分辨率,我可以用它嗎?

答:爲什麼不行呢?你沒有理由一定使用640x480,800x600,或甚至1024x768.XFree86服務器讓你在配置硬件時有很大自由.通常要進行兩到三次嘗試以找出正確的配置.要得到的最重要的東西是在合理大小的可視區域下的高刷新率,而不是以閃爍得眼睛流淚爲代價的高分辨率!

問:給定65MHz點時鐘和55KHz行頻(HSF)時這是唯一的分辨率嗎?

答:絕對不是!你被鼓勵遵循常規的步驟並進行反復試驗來找出眞正符合你的喜好的設置.這樣試驗會有許多樂趣.大多數設置也許隻是給你討厭的雜亂視頻信號,但實際上一台新式多頻同步顯示器通常是不容易損壞的.可是在長期使用一種模式前請確認你的顯示器能支持該模式的幀率.

小心固定頻率顯示器!這種試驗可以很快地使它們損壞.確認你對它們進行的每次試驗都用了有效的刷新率.

問:你剛提及兩種標準分辨率.在Xconfig中,有很多可用的標準分辨率,你能告訴我在修改時序時是否有什麼要點嗎?

答:絕對有!以在當前的Xconfig中列出的"標準"模式640x480爲例.它使用25MHz驅動頻率,幀長是800和525=>刷新率約爲59.5Hz.不太壞.但對很多SVGA板卡來說28MHz是一個通常可使用的驅動頻率.如果我們以其驅動640x480,按照我們上面討論的步驟,你可得到幀長爲812(向下舍入爲808)和505.現在刷新率升到68Hz,是一個對原標準模式相當有意義的改進.

問:你能對迄今爲止我們已討論的內容作個總結嗎?

答:簡而言之:

1. 對於任何固定的驅動頻率,提高最大分辨率會招致刷新率降低的代價從而帶來更明顯的閃爍.
2. 如果很想用高分辨率而你的顯示器也支持,設法獲得一塊提供匹配的點時鐘(DCF)的SVGA卡.越高越好!

15 解決圖象問題

好,你已經獲得了X配置的值.你把它們放人Xconfig並加上一個測試模式的標注.你啓動X,熱鍵切換到新模式,...而圖象看來不正常.你該怎麼辦?這裡是一份清單,列出常見的視頻圖象變形以及如何進行糾正.

(糾正這些較小的變形是**xvidtune**(1)眞正閃亮之處.)

通過改變同步脈沖時序可以移動圖象.通過改變幀長可以按比例縮放圖象(你必須移動同步脈沖以保持圖象在對應的相同位置,否則縮放也會移動圖象).這裡是一些更詳細的方法:

水平和垂直位置是獨立的.換言之,水平移動圖象不會影響垂直位置,反之亦然.可是,縮放就不一樣了.雖然改變水平尺寸不會改變垂直尺寸,反之亦然,但兩個方向總的改變是受限制的.特別是圖象在兩個方向都太大時,你大概不得不調到一個更高的點時鐘來糾正.因爲這樣提升了可用的分辨率,它幾乎不成爲一個問題了!

15.1 圖象向左或向右移位

要糾正它,移動行同步脈沖.換言之,對定義了行同步脈沖上升沿和下降沿的水平時序段中間兩個數字進行增減(按8的倍數).

如果圖象偏左(右邊界太大,你想向右移動圖象)減少數字.如果圖象偏右(左邊界太大,你想向左移動圖象)增大同步脈沖.

15.2 圖象向上或向下移位

要糾正它,移動場同步脈沖.換言之,對定義了場同步脈沖上升沿和下降沿的垂直時序段中間兩個數字進行增減.

如果圖象偏上(下邊界太大,你想向下移動圖象)減少數字.如果圖象偏下(上邊界太大,你想向上移動圖象)增大數字.

15.3 圖象在水平和垂直方向都太大

切換到一個更高的顯示時鐘速度.如果在你的時鐘設置文件中有多個模式,很可能一個低速模式被錯誤地激活了.

15.4 圖象在水平方向太寬(太窄)

要糾正它,增大(減少)水平幀長.換言之,改變第一個時序段的第四個數字.爲了避免移動圖象,也把同步脈沖(第二和第三個數字)移動一半的點數,以保持圖象在對應的相同位置.

15.5 圖象在垂直方向太深(太淺)

要糾正它,增大(減少)垂直幀長.換言之,改變第二個時序段的第四個數字.爲了避免移動圖象,也把同步脈沖(第二和第三個數字)移動一半的點數,以保持圖象在對應的相同位置.

任何不能通過這些技巧的組合來解決的變形很可能是犯了一些更基本錯誤的跡象,如計算錯誤或點時鐘高於顯示器能處理的范圍.

最後,記住增大任一個幀長會降低你的刷新率,反之亦然.

時候你可以通過調整顯示器上的屏幕控制來糾正小的變形.缺點是如果你爲了糾正圖形模式問題而調整得離中性(出廠)設定太遠的話,也許在文本模式下顯示器圖象會變得古怪.改好你的模式行是更好的方法.

16 繪制顯示器特性

要繪制一份顯示器模式圖,你將需要gnuplot軟件包(一個類UNIX操作系統下的繪圖語言自由軟件)和modeplot工具.它是一個輸入命令行選項來繪制顯示器特性圖的shell/gnuplot腳本.

這裡是modeplot的一份拷貝:

```
#!/bin/sh
#
# modeplot --- 生成顯示器可使用的X模式圖
#
# 執行'modeplot -?'獲知控制選項.
#

# 顯示器描述.頻寬以MHz爲單位,行頻以KHz爲單位,而場頻以Hz爲單位.
TITLE="Viewsonic 21PS"
BANDWIDTH=185
MINHSF=31
MAXHSF=85
MINVSF=50
MAXVSF=160
ASPECT="4/3"
vesa=72.5       # VESA建議的最小刷新率

while [ "$1" != "" ]
do
        case $1 in
        -t) TITLE="$2"; shift;;
        -b) BANDWIDTH="$2"; shift;;
        -h) MINHSF="$2" MAXHSF="$3"; shift; shift;;
        -v) MINVSF="$2" MAXVSF="$3"; shift; shift;;
        -a) ASPECT="$2"; shift;;
        -g) GNUOPTS="$2"; shift;;
        -?) cat <<EOF
modeplot control switches:

-t "<description>"      name of monitor            defaults to "Viewsonic 21PS"
-b <nn>                 bandwidth in MHz           defaults to 185
-h <min> <max>          min & max HSF (KHz)        defaults to 31 85
-v <min> <max>          min & max VSF (Hz)         defaults to 50 160
-a <aspect ratio>       aspect ratio               defaults to 4/3
-g "<options>"          pass options to gnuplot
```

```
The -b, -h and -v options are required, -a, -t, -g optional. You can use -g to pass a device type to gnuplot so that (for example) modeplot's output can be redirected to a printer. See gnuplot(1) for details.

The modeplot tool was created by Eric S. Raymond <esr@thyrsus.com> based on analysis and scratch code by Martin Lottermoser <Martin.Lottermoser@mch.sni.de>

This is modeplot $Revision: 1.13 $
EOF
                exit;;
        esac
        shift

done

gnuplot $GNUOPTS <<EOF
set title "$TITLE Mode Plot"

# 有魔力的數值.不幸的是特性圖對它們的改變相當敏感,而對某些顯示器它們不能代表實際值.我們需
要糾
# 正這些值以獲得更接近的模式圖.這些值是查找ModeDB數據庫中大量的值而得到的.
F1 = 1.30       # 轉換水平分辨率到幀寬的倍數
F2 = 1.05       # 轉換垂直分辨率到幀高的倍數

# 函數定義(乘以1.0強迫進行實數運算)
ac = (1.0*$ASPECT)*F1/F2
refresh(hsync, dcf) = ac * (hsync**2)/(1.0*dcf)
dotclock(hsync, rr) = ac * (hsync**2)/(1.0*rr)
resolution(hv, dcf) = dcf * (10**6)/(hv * F1 * F2)

# 在坐標軸上放置標記
set xlabel 'DCF (MHz)'
set ylabel 'RR (Hz)' 6  # 在Y軸右邊放置它

# 生成圖形
set grid
set label "VB" at $BANDWIDTH+1, ($MAXVSF + $MINVSF) / 2 left
set arrow from $BANDWIDTH, $MINVSF to $BANDWIDTH, $MAXVSF nohead
set label "max VSF" at 1, $MAXVSF-1.5
set arrow from 0, $MAXVSF to $BANDWIDTH, $MAXVSF nohead
set label "min VSF" at 1, $MINVSF-1.5
set arrow from 0, $MINVSF to $BANDWIDTH, $MINVSF nohead
set label "min HSF" at dotclock($MINHSF, $MAXVSF+17), $MAXVSF + 17 right
set label "max HSF" at dotclock($MAXHSF, $MAXVSF+17), $MAXVSF + 17 right
set label "VESA $vesa" at 1, $vesa-1.5
set arrow from 0, $vesa to $BANDWIDTH, $vesa nohead # style -1
plot [dcf=0:1.1*$BANDWIDTH] [$MINVSF-10:$MAXVSF+20] \
  refresh($MINHSF, dcf) notitle with lines 1, \
  refresh($MAXHSF, dcf) notitle with lines 1, \
```

```
  resolution(640*480,   dcf) title "640x480  " with points 2, \
  resolution(800*600,   dcf) title "800x600  " with points 3, \
  resolution(1024*768,  dcf) title "1024x768 " with points 4, \
  resolution(1280*1024, dcf) title "1280x1024" with points 5, \
  resolution(1600*1280, dcf) title "1600x1200" with points 6

pause 9999
EOF
```

一旦知道了modeplot和gnuplot軟件包已經在適當位置,你需要下列顯示器特性:

- 頻寬(VB)
- 行頻范圍(HSF)
- 場頻范圍(VSF)

繪制程序需要進行一些不完全精確的簡化假定.因此得出的圖隻能是一個粗略描述.這些假定是:

1. 所有分辨率有單一的固定屏幕高寬比 AR = HR / VR .標準分辨率有 AR = 4/3 或 AR = 5/4 .modeplot程序默認假定4/3,但你可以替換它.
2. 對計算的模式,水平和垂直幀長分別是水平和垂直分辨率的固定倍數:

```
HFL = F1 * HR
VFL = F2 * VR
```

作爲一份粗略的指南,讓 F1 = 1.30 而 F2 = 1.05 (查閱9 ()"計算幀的大小").

現在處理一個特殊的同步頻率,行頻(HSF).根據剛才提出的假定,其每個值對時鐘頻率DCF而言已經決定刷新率RR,也就是說,對行頻(HSF)每個值有一個函數RR(DCF).這可以依下列各項得出.

刷新率等於時鐘頻率除以幀長的乘積:

```
RR = DCF / (HFL * VFL)          (*)
```

另一方面,水平幀長等於時鐘率除以行頻:

```
HFL = DCF / HSF                 (**)
```

根據上述兩個假定VFL可以化爲HFL:

```
VFL = F2 * VR
    = F2 * (HR / AR)
    = (F2 / F1) * HFL / AR      (***)
```

插入(**)和(***)到(*)我們得到:

```
RR = DCF / ((F2 / F1) * HFL**2 / AR)
   = (F1 / F2) * AR * DCF * (HSF / DCF)**2
   = (F1 / F2) * AR * HSF**2 / DCF
```

對於固定的HSF,F1,F2和AR,在我們的圖中這是一條雙曲線.對行頻最小值和最大值畫兩條這樣的曲線我們就得到了允許區域的兩條邊界.

穿過特性區域的直線代表特定的分辨率.這是基於(*)和第二個假定:

```
RR = DCF / (HFL * VFL) = DCF / (F1 * HR * F2 * VR)
```

通過爲某人感興趣的所有分辨率畫這樣的直線,他能立刻記住顯示器特性范圍内分辨率,時鐘頻率和刷新率之間可能的關系.注意這些線不依賴於顯示器的特性,但確實依賴於第二個假定.

`modeplot`工具提供給你容易的途徑繪制特性圖.運行`modeplot -?`看它的控制選項.一個典型調用類似這樣:

```
modeplot -t "Swan SW617" -b 85 -v 50 90 -h 31 58
```

-b選項指定頻寬;-v和-h設置行頻和場頻范圍.

當閱讀`modeplot`的輸出時,始終記住它隻給出近似的描述.例如,它忽視了由同步脈沖寬度的最小要求造成的水平幀長(HFL)的限制,而且它的精確度受限於兩個假定的精確度.因此它不是在11 (匯總以上討論)中出現的詳細計算(涉及某些巫術)的代替品.無論如何,對於什麼模式是可能的和有哪些相關權衡,它應該給了你更好的感覺.

17 致謝

本文最初原形的作者是Chin Fang <fangchin@leland.stanford.edu>.

Eric S. Raymond <esr@snark.thyrsus.com> 在努力理解它的基礎上重新加工,重新組織和整體改寫了Chin Fang的原作.在這個過程中,他並入了由Bob Crosson <crosson@cam.nist.gov> 寫的另一篇HOWTO的大部分内容.

隔行模式資料主要作者是David Kastrup <dak@pool.informatik.rwth-aachen.de>.

Nicholas Bodley <nbodley@alumni.princeton.edu> 對顯示器如何工作的章節進行了更正和闡明.

Payne Freret <payne@freret.org> 更正了關於顯示器設計的一些技術上的小錯誤.

Martin Lottermoser <Martin.Lottermoser@mch.sni.de> 貢獻了用gnuplot繪制模式圖的想法,進行了`modeplot`背後的數學分析.發布的`modeplot`是由ESR根據Martin的原始gbuplot代碼爲本文的情況重新設計和歸納出來的.