Information for Chips and Technologies Users

David Bateman (<dbateman@club-internet.fr>),
Egbert Eich (<eich@xfree86.org>)

19th July 1999

1. Introduction

With the release of XFree86 version 4.0, the Chips and Technologies driver has been extensively rewritten and contains many new features. This driver must be considered work in progress, and those users wanting stability are encouraged to use the older XFree86 3.3.x versions. However this version of the Chips and Technologies driver has many new features and bug fixes that might make users prefer to use this version. These features include

- The long standing black/blue screen problem that some people have had should be fixed.
- Hardware/Software cursor switching on the fly, that should fix many of the known hardware cursor problems.
- Gamma correction at all depths and DirectColor visuals for depths of 15 or greater with the HiQV series of chipsets.
- Supports PsuedoColor overlays on 16bpp TrueColor screens for HiQV.
- 32bpp pixmaps while using a framebuffer in 24bpp packed pixel mode.
- Heaps more acceleration.
- 1/4bpp support.
- Multihead
- Much more...

This document attempts to discuss the features of this driver, the options useful in configuring it and the known problems. Most of the Chips and Technologies chipsets are supported by this driver to some degree.

2. Supported Chips

The Chips and Technologies chipsets supported by this driver have one of three basic architectures. A basic architecture, the WinGine architecture which is a modification on this basic architecture and a completely new HiQV architecture.

2.1 Basic architecture

ct65520
(Max Ram: 1Mb, Max Dclk: 68MHz@5V)

ct65525

This chip is basically identical to the 65530. It has the same ID and is identified as a 65530 when probed. See ct65530 for details.

ct65530

This is a very similar chip to the 65520. However it additionally has the ability for mixed 5V and 3.3V operation and linear addressing of the video memory. (Max Ram: 1Mb, Max Dclk: 56MHz@3.3V, 68MHz@5V)

ct65535

This is the first chip of the ct655xx series to support fully programmable clocks. Otherwise it has the the same properties as the 65530.

ct65540

This is the first version of the of the ct655xx that was capable of supporting Hi-Color and True-Color. It also includes a fully programmable dot clock and supports all types of flat panels. (Max Ram: 1Mb, Max Dclk: 56MHz@3.3V, 68MHz@5V)

ct65545

The chip is very similar to the 65540, with the addition of H/W cursor, pop-menu acceleration, BitBLT and support of PCI Buses. PCI version also allow all the BitBLT and H/W cursor registers to be memory mapped 2Mb above the Base Address. (Max Ram: 1Mb, Max Dclk: 56MHz@3.3V,68MHz@5V)

ct65546

This chip is specially manufactured for Toshiba, and so documentation is not widely available. It is believed that this is really just a 65545 with a higher maximum dot-clock of 80MHz. (Max Ram: 1Mb?, Max Dclk: 80MHz?)

ct65548

This chip is similar to the 65545, but it also includes XRAM support and supports the higher dot clocks of the 65546. (Max Ram: 1Mb, Max Dclk: 80MHz)

2.2 WinGine architecture

ct64200

This chip, also known as the WinGine, is used in video cards for desktop systems. It often uses external DAC's and programmable clock chips to supply additional functionally. None of these are currently supported within the driver itself, so many cards will only have limited support. Linear addressing is not supported for this card in the driver. (Max Ram: 2Mb, Max Dclk: 80MHz)

ct64300

This is a more advanced version of the WinGine chip, with specification very similar to the 6554x series of chips. However there are many differences at a register level. A similar level of acceleration to the 65545 is included for this driver. (Max Ram: 2Mb, Max Dclk: 80MHz)

2.3 HiQV Architecture

ct65550

This chip includes many new features, including improved BitBLT support (24bpp color expansion, wider maximum pitch, etc), Multimedia unit (video capture, zoom video port, etc) and 24bpp uncompressed true color (i.e 32bpp mode). Also memory mapped I/O is possible on all bus configurations. (Max Ram: 2Mb, Max Dclk: 80MHz@3.3V,100MHz@5V)

ct65554

This chip is similar to the 65550 but has a 64bit memory bus as opposed to a 32bit bus. It also has higher limits on the maximum memory and pixel clocks (Max Ram:

4Mb, Max Dclk: 100MHz@3.3V)

ct65555

Similar to the 65554 but has yet higher maximum memory and pixel clocks. It also includes a new DSTN dithering scheme that improves the performance of DSTN screens. (Max Ram: 4Mb, Max Dclk: 110MHz@3.3V)

ct68554

Similar to the 65555 but also incorporates "PanelLink" drivers. This serial link allows an LCD screens to be located up to 100m from the video processor. Expect to see this chip soon in LCD desktop machines (Max Ram: 4Mb, Max Dclk: 110MHz@3.3V)

ct69000

Similar to the 65555 but incorporates 2Mbytes of SGRAM on chip. It is the first Chips and Technologies chipset where all of the registers are accessible through MMIO, rather than just the BitBlt registers. (Max Ram: 2Mb Only, Max Dclk: 130MHz@3.3V)

ct69030

Similar to the 69000 but incorporates 4Mbytes of SGRAM on chip and has faster memory and pixel clock limits. Also includes a second display channel so that the CRT can display independently of the LCD. (Max Ram: 4Mb Only, Max Dclk: 170MHz@3.3V)

3. XF86Config Options

The following options are of particular interest to the Chips and Technologies driver. It should be noted that the options are case insensitive, and that white space and "_" characters are ignored. There are therefore a wide variety of possible forms for all options. The forms given below are the preferred forms.

Options related to drivers can be present in the Screen, Device and Monitor sections and the Display subsections. The order of precedence is Display, Screen, Monitor, Device.

Option "NoAccel"

This option will disable the use of any accelerated functions. This is likely to help with some problems related to DRAM timing, high dot clocks, and bugs in accelerated functions, at the cost of performance (which will still be reasonable on VLB/PCI).

VideoRam 1024 (or another value)

This option will override the detected amount of video memory, and pretend the given amount of memory is present on the card.

Option "NoLinear"

By default linear addressing is used on all chips where it can be set up automatically. The exception is for depths of 1 or 4bpp where linear addressing is turned off by default. It is possible to turn the linear addressing off with this option. Note that H/W acceleration is only supported with linear addressing.

Option "Linear"

When the chipset is capable of linear addressing and it has been turned off by default, this option can be used to turn it back on. This is useful for the 65530 chipset where the base address of the linear framebuffer must be supplied by the user, or at depths 1 and 4bpp. Note that linear addressing at 1 and 4bpp is not guaranteed to work correctly.

MemBase 0x03b00000 (or a different address)

This sets the physical memory base address of the linear framebuffer. Typically this is probed correctly, but if you believe it to be mis-probed, this option might help. Also for non PCI machines specifying this force the linear base address to be this value, reprogramming the video processor to suit. Note that for the 65530 this is required as the base address can't be correctly probed.

Option "HWcursor"

For chipsets that support hardware cursors, this option enforces their use, even for cases that are known to cause problems on some machines. Note that it is overridden by the "`SWcursor`" option. Hardware cursors effectively speeds all graphics operations as the job of ensuring that the cursor remains on top is now given to the hardware. It also reduces the effect of cursor flashing during graphics operations.

Option "SWcursor"

This disables use of the hardware cursor provided by the chip. Try this if the cursor seems to have problems.

Option "STN"

The server is unable to differentiate between SS STN and TFT displays. This forces it to identify the display as a SS STN rather than a TFT.

Option "UseModeline"

The flat panel timings are related to the panel size and not the size of the mode specified in XF86Config. For this reason the default behaviour of the server is to use the panel timings already installed in the chip. The user can force the panel timings to be recalculated from the modeline with this option. However the panel size will still be probed.

Option "FixPanelSize"

For some machines the LCD panel size is incorrectly probed from the registers. This option forces the LCD panel size to be overridden by the modeline display sizes. This will prevent the use of a mode that is a different size than the panel. Before using this check that the server reports an incorrect panel size. This option can be used in conjunction with the option "UseModeline" to program all the panel timings using the modeline values.

Option "NoStretch"

When the size of the mode used is less than the panel size, the default behaviour of the server is to stretch the mode in an attempt to fill the screen. A "`letterbox`" effect with no stretching can be achieved using this option.

Option "LcdCenter"

When the size of the mode used is less than the panel size, the default behaviour of the server is to align the left hand edge of the display with the left hand edge of the screen. Using this option the mode can be centered in the screen. This option is reported to have problems with some machines at 16/24/32bpp, the effect of which is that the right-hand edge of the mode will be pushed off the screen.

Option "HWclocks"

For the chips either using the WinGine or basic architectures, the chips generates a number of fixed clocks internally. With the chips 65535 and later or the 64300, the default is to use the programmable clock for all clocks. It is possible to use the fixed clocks supported by the chip instead by using this option. Typically this will give you some or all of the clocks 25.175, 28.322, 31.000 and 36.000MHz. The current programmable clock will be given as the last clock in the list. On a cold-booted system this might be the appropriate value to use at the text console (see the "`TextClockFreq`" option), as many flat panels will need a dot clock different than the default to synchronise. The programmable clock makes this option obsolete and so it's use isn't recommended. It is completely ignored for HiQV chipsets.

Option "UseVclk1"

The HiQV series of chips have three programmable clocks. The first two are usually loaded with 25.175 and 28.322MHz for VGA backward compatibility, and the third is used as a fully programmable clock. On at least one system (the Inside 686 LCD/S single board computer) the third clock is unusable. This option forces the use of VClk1 as the programmable clock.

TextClockFreq 25.175

Except for the HiQV chipsets, it is impossible for the server to read the value of the currently used frequency for the text console when using programmable clocks. Therefore the server uses a default value of 25.175MHz as the text console clock. For some LCDs, in particular DSTN screens, this clock will be wrong. This allows the user to select a different clock for the server to use when returning to the text console.

Option "FPClock8" "65.0"

Option "FPClock16" "65.0" Option "FPClock24" "65.0" Option "FPClock32" "65.0"" In general the LCD panel clock should be set independently of the modelines supplied. Normally the chips BIOS set the flat panel clock correctly and so the default behaviour with HiQV chipset is to leave the flat panel clock alone, or force it to be 90% of the maximum allowable clock if the current panel clock exceeds the dotclock limitation due to a depth change. This option allows the user to force the server the reprogram the flat panel clock independently of the modeline with HiQV chipset. The four options are for 8bpp or less, 16, 24 or 32bpp LCD panel clocks, where the options above set the clocks to 65MHz.

Option "MMIO"

This has a different effect depending on the hardware on which it is used. For the 6554x machines MMIO is only used to talk to the BitBLT engine and is only usable with PCI buses. It is enabled by default for 65545 machines since the blitter can not be used otherwise. The HiQV series of chipsets must use MMIO with their BitBLT engines, and so this is enabled by default. However the 690xx chipsets can use MMIO for all communications with the video processor. So using this option on a 690xx chipset forces them to use MMIO for all communications. This only makes sense when the 690xx is on a PCI bus so that normal PIO can be disabled. (WARNING!! 690xx MMIO is untested)

Option "SuspendHack"

This option sets the centering and stretching to the BIOS default values. This can fix suspend/resume problems on some machines. It overrides the options "LcdCenter" and "NoStretch".

Option "18bitBus" (Chips 65540/45/46/48)

For 24bpp on TFT screens, the server assumes that a 24bit bus is being used. This can result in a reddish tint to 24bpp mode. This option, selects an 18 bit TFT bus. For other depths this option has no effect.

Chipset "ct65546" (or some other chip)

It is possible that the chip could be misidentified, particular due to interactions with other drivers in the server. It is possible to force the server to identify a particular chip with this option.

Option "SyncOnGreen"

Composite sync on green. Possibly useful if you wish to use an old workstation monitor. The HiQV internal RAMDAC's supports this mode of operation, but whether a particular machine does depends on the manufacturer.

DacSpeed 80.000

The server will limit the maximum dotclock to a value as specified by the manufacturer. This might make certain modes impossible to obtain with a reasonable refresh rate. Using this option the user can override the maximum dot-clock and specify any value they prefer. Use caution with this option, as driving the video processor beyond its specifications might cause damage.

Option "SetMClk" "38.000MHz"

Option "SetMClk" "38000kHz"" This option sets the internal memory clock (MCLK) registers of HiQV chipsets to 38MHz or some other value. Use caution as excess heat generated by the video processor if its specifications are exceeded might cause damage. However careful use of this option might boost performance. This option might also be used to reduce the speed of the memory clock to preserve power in modes that don't need the full speed of the memory to work correctly. This option might also be needed to reduce the speed of the memory clock with the "`Overlay`" option.

Option "RGBbits" "8"

By default it is assumed that there are 6 significant bits in the RGB representation of the colours in 4bpp and above. If the colours seem darker than they should be, perhaps your ramdac is has 8 significant bits. This option forces the server to assume that there are 8 significant bits.

Option "ShowCache"

This is a debugging option and general users have no need of it. Using this option, when the virtual desktop is scrolled away from the zero position, the pixmap cache becomes visible. This is useful to see that pixmaps, tiles, etc have been properly cached.

Option "ShadowFB"

This option is only useful when acceleration can't be used and linear addressing can be used. With this option all of the graphics are rendered into a copy of the framebuffer that is keep in the main memory of the computer, and the screen is updated from this copy. In this way the expensive operation of reading back to contents of the screen is never performed and the performance is improved. Because the rendering is all done into a virtual framebuffer acceleration can not be used.

Option "Overlay"

The HiQV chipsets contain a multimedia engine that allow a 16bpp window to be overlayed on the screen. This driver uses this capability to include a 16bpp framebuffer on top of an 8bpp framebuffer. In this way PseudoColor and TrueColor visuals can be used on the same screen. XFree86 believes that the 8bpp framebuffer is overlayed on the 16bpp framebuffer. Therefore to use this option the server must be started in either 15 or 16bpp depth. Also the maximum size of the desktop with this option is 1024x1024, as this is the largest window that the HiQV multimedia engine can display. Note that this option using the multimedia engine to its limit, and some manufacturers have set a default memory clock that will cause pixel errors with this option. If you get pixel error with this option try using the "`SetMClk`" option to slow the memory clock.

Option "ColorKey" "255"

Normally the color transparency key for the overlay is the 8bpp lookup table entry 255. This might cause troubles with some applications, and so this option allows the color transparency key to be set to some other value. Legal values are 2 to 255 inclusive.

Option "XaaNoScreenToScreenCopy",

Option "XaaNoSolidFillRect", Option "XaaNoSolidHorVertLine", Option "XaaNoMono8x8PatternFillRect", Option "XaaNoColor8x8PatternFillRect", Option "XaaNoCPUToScreenColorExpandFill", Option "XaaNoScreenToScreenColorExpandFill", Option "XaaNoImageWriteRect", Option "XaaNoImageReadRect", Option "XaaNoPixmapCache", Option "XaaNoOffscreenPixmaps" " These option individually disable the features of the XAA acceleration code that the Chips and Technologies driver uses. If you have a problem with the acceleration and these options will allow you to isolation the problem. This information will be invaluable in debugging any problems.

4. Modelines

When constructing a modeline for use with the Chips and Technologies driver you'll needed to considered several points

* Virtual Screen Size

It is the virtual screen size that determines the amount of memory used by a mode. So if you have a virtual screen size set to 1024x768 using a 800x600 at 8bpp, you use 768kB for the mode. Further to this some of the XAA acceleration requires that the display pitch is a multiple of 64 pixels. So the driver will attempt to round-up the virtual X dimension to a multiple of 64, but leave the virtual resolution untouched. This might further reduce the available memory.

* 16/24/32 Bits Per Pixel

Hi-Color and True-Color modes are implemented in the server. The clocks in the 6554x series of chips are internally divided by 2 for 16bpp and 3 for 24bpp, allowing one modeline to be used at all depths. The effect of this is that the maximum dot clock visible to the user is a half or a third of the value at 8bpp. The HiQV series of chips doesn't need to use additional clock cycles to display higher depths, and so the same modeline can be used at all depths, without needing to divide the clocks. Also 16/24/32 bpp modes will need 2 , 3 or 4 times respectively more video ram.

* Frame Acceleration

Many DSTN screens use frame acceleration to improve the performance of the screen. This can be done by using an external frame buffer, or incorporating the framebuffer at the top of video ram depending on the particular implementation. The Xserver assumes that the framebuffer, if used, will be at the top of video ram. The amount of ram required for the framebuffer will vary depending on the size of the screen, and will reduce the amount of video ram available to the modes. Typical values for the size of the framebuffer will be 61440 bytes (640x480 panel), 96000 bytes (800x600 panel) and 157287 bytes (1024x768 panel).

* H/W Acceleration

The H/W cursor will need 1kB for the 6554x and 4kb for the 65550. On the 64300 chips the H/W cursor is stored in registers and so no allowance is needed for the H/W cursor. In addition to this many graphics operations are speeded up using a "`pixmap cache`". Leaving too little memory available for the cache will only have a detrimental effect on the graphics performance.

* PseudoColor Overlay

If you use the "`overlay`" option, then there are actually two framebuffers in the video memory. An 8bpp one and a 16bpp one. The total memory requirements in this mode of operation is therefore similar to a 24bpp mode. The overlay consumes memory bandwidth, so that the maximum dotclock will be similar to a 24bpp mode.

* VESA like modes

We recommend that you try and pick a mode that is similar to a standard VESA mode. If you don't a suspend/resume or LCD/CRT switch might mess up the screen. This is a problem with the video BIOS not knowing about all the funny modes that might be selected.

* Dot Clock

For LCD screens, the lowest clock that gives acceptable contrast and flicker is usually the best one. This also gives more memory bandwidth for use in the drawing operations. Some users prefer to use clocks that are defined by their BIOS. This has the advantage that the BIOS will probably restore the clock they specified after a suspend/resume or LCD/CRT switch. For a complete discussion on the dot clock limitations, see the next section.

The driver is capable of driving both a CRT and a flat panel display. In fact the timing for the flat panel are dependent on the specification of the panel itself and are independent of the particular mode chosen. For this reason it is recommended to use one of the programs that automatically generate XF86Config files, such as "`xf86config`" or "`XF86Setup`".

However there are many older machines, particularly those with 800x600 screen or larger, that need to reprogram the panel timings. The reason for this is that the manufacturer has used the panel timings to get a standard EGA mode to work on flat panel, and these same timings don't work for an SVGA mode. For these machines the "`UseModeline`" and/or possibly the "`FixPanelSize`" option might be needed. Some machines that are known to need these options include.

```
Modeline "640x480@8bpp"         25.175  640  672  728  816   480  489  501  526
Modeline "640x480@16bpp"  25.175  640  672  728  816   480  489  501  526
Options: "UseModeline"
Tested on a Prostar 8200, (640x480, 65548, 1Mbyte)


Modeline "800x600@8bpp"         28.322  800  808  848  936   600  600  604  628
Options: "FixPanelSize", "UseModeline"
Tested on a HP OmniBook 5000CTS (800x600 TFT, 65548, 1Mbyte)


Modeline "800x600@8bpp"         30.150  800  896  960 1056   600  600  604  628
Options: "FixPanelSize", "UseModeline"
Test on a Zeos Meridan 850c (800x600 DSTN, 65545, 1Mbyte)
```

The NEC Versa 4080 just needs the "FixPanelSize" option. To the best of my knowledge no machine with a HiQV needs the "UseModeline" or "FixPanelSize" options.

5. The Full Story on Clock Limitations

There has been much confusion about exactly what the clock limitations of the Chips and Technologies chipsets are. Hence I hope that this section will clear up the misunderstandings.

In general there are two factors determining the maximum dotclock. There is the limit of the maximum dotclock the video processor can handle, and there is another limitation of the available memory bandwidth. The memory bandwidth is determined by the clock used for the video memory. For chipsets incapable of colour depths greater that 8bpp like the 65535, the dotclock limit is solely determined by the highest dotclock the video processor is capable of handling. So this limit will be either 56MHz or 68MHz for the 655xx chipsets, depending on what voltage they are driven with, or 80MHz for the 64200 WinGine machines.

The 6554x and 64300 WinGine chipsets are capable of colour depths of 16 or 24bpp. However there is no reliable way of probing the memory clock used in these chipsets, and so a conservative limit must be taken for the dotclock limit. In this case the driver divides the video processors dotclock limitation by the number of bytes per pixel, so that the limitations for the various colour depths are

	8bpp	16bpp	24bpp
64300	85	42.5	28.33
65540/65545 3.3v	56	28	18.67
65540/65545 5v	68	34	22.67
65546/65548	80	40	26.67

For a CRT or TFT screen these limitations are conservative and the user might safely override them with the "DacSpeed" option to some extent. However these numbers take no account of the extra bandwidth needed for DSTN screens.

For the HiQV series of chips, the memory clock can be successfully probed. Hence you will see a line like

```
(--) CHIPS(0): Probed memory clock of  40.090 MHz
```

in your startx log file. Note that many chips are capable of higher memory clocks than actually set by BIOS. You can use the "SetMClk" option in your XF86Config file to get a higher MClk. However some video ram, particularly EDO, might not be fast enough to handle this, resulting in drawing errors on the screen. The formula to determine the maximum usable dotclock on the HiQV series of chips is

```
Max dotclock = min(MaxDClk,  0.70  * 4 * MemoryClk / (BytesPerPixel +
                (isDSTN == TRUE ? 1 : 0)))
```

which says that there are two limits on the dotclock. One the overall maximum, and another due to the available memory bandwidth of the chip. For the memory bandwidth 4 bytes are transfered every clock cycle (Hence the 4), but after accounting for the RAS/CAS signaling only about 70% of the bandwidth is available. The whole thing is divided by the bytes per pixel, plus an extra byte if you are using a DSTN. The extra byte with DSTN screens is used for the frame buffering/acceleration in these screens. So for the various Chips and Technologies chips the maximum specifications are

	Max DClk MHz	Max Mem Clk MHz
65550 rev A 3.3v	80	38
65550 rev A 5v	110	38
65550 rev B	95	50
65554	94.5	55
65555	110	55
68554	110	55
69000	135	83
69030	170	100

Note that all of the chips except the 65550 rev A are 3.3v only. Which is the reason for the drop in the dot clock. Now the maximum memory clock is just the maximum supported by the video processor, not the maximum supported by the video memory. So the value actually used for the memory clock might be significantly less than this maximum value. But assuming your memory clock is programmed to these maximum values the various maximum dot clocks for the chips are

	------CRT/TFT-------			--------DSTN--------		
	8bpp	16bpp	24bpp	8bpp	16bpp	24bpp
65550 rev A 3.3v	80	53.2	35.47	53.2	35.47	26.6
65550 rev A 5v	106.2	53.2	35.47	53.2	35.47	26.6
65550 rev B	95	70	46.67	70	46.67	35.0
65554	94.5	77	51.33	77	51.33	38.5
65555	110	77	51.33	77	51.33	38.5
68554	110	77	51.33	77	51.33	38.5
69000	135	116.2	77.47	116.2	77.47	58.1
69030	170	140	93.33	140	93.33	70

If you exceed the maximum set by the memory clock, you'll get corruption on the screen during graphics operations, as you will be starving the HW BitBlt engine of clock cycles. If you are driving the video memory too fast (too high a MemClk) you'll get pixel corruption as the data actually written to the video memory is corrupted by driving the memory too fast. You can probably get away with exceeding the Max DClk at 8bpp on TFT's or CRT's by up to 10% or so without problems, it will just generate more heat, since the 8bpp clocks aren't limited by the available memory bandwidth.

If you find you truly can't achieve the mode you are after with the default clock limitations, look at the options "`DacSpeed`" and "`SetMClk`". Using these should give you all the capabilities you'll need in the server to get a particular mode to work. However use caution with these options, because there is no guarantee that driving the video processor beyond it capabilities won't cause damage.

6. Troubleshooting

The cursor appears as a white box, after switching modes

> There is a known bug in the H/W cursor, that sometimes causes the cursor to be redrawn as a white box, when the mode is changed. This can be fixed by moving the cursor to a different region, switching to the console and back again, or if it is too annoying the H/W cursor can be disabled by removing the "`HWcursor`" option.

The cursor hot-spot isn't at the same point as the cursor

> With modes on the 6555x machines that are stretched to fill the flat panel, the H/W cursor is not correspondingly stretched. This is a small and long-standing bug in the current server. You can avoid this by either using the "`NoStretch`" option or removing the `HWcursor`" option.

The lower part of the screen is corrupted

> Many DSTN screens use the top of video ram to implement a frame accelerator. This reduces the amount of video ram available to the modes. The server doesn't prevent the user from specifying a mode that will use this memory, it prints a warning on the console. The effect of this problem will be that the lower part of the screen will reside in the same memory as the frame accelerator and will therefore be corrupt. Try reducing the amount of memory consumed by the mode.

There is a video signal, but the screen doesn't sync.

> You are using a mode that your screen cannot handle. If it is a non-standard mode, maybe you need to tweak the timings a bit. If it is a standard mode and frequency that your screen should be able to handle, try to find different timings for a similar mode and frequency combination. For LCD modes, it is possible that your LCD panel requires different panel timings at the text console than with a graphics mode. In this case you will need the "`UseModeline`" and perhaps also the "`FixPanelSize`" options to reprogram the LCD panel timings to sensible values.

'Wavy' screen.

> Horizontal waving or jittering of the whole screen, continuously (independent from drawing operations). You are probably using a dot clock that is too high (or too low); it is also possible that there is interference with a close MCLK. Try a lower dot clock. For CRT's you can also try to tweak the mode timings; try increasing the second horizontal value somewhat.

Crash or hang after start-up (probably with a black screen).

> Try the "`NoAccel`" or one of the XAA acceleration options discussed above. Check that the BIOS settings are OK; in particular, disable caching of 0xa0000-0xaffff. Disabling hidden DRAM refresh may also help.

Hang as the first text is appearing on the screen on SVR4 machines.

> This problem has been reported under UnixWare 1.x, but not tracked down. It doesn't occur under UnixWare 2.x and only occurs on the HiQV series of chips. It might affect some other SVR4 operating systems as well. The workaround is to turn

off the use of CPU to screen acceleration with the "XaaNoCPUToScreenColorExapndFill" option.

Crash, hang, or trash on the screen after a graphics operation.

This may be related to a bug in one of the accelerated functions, or a problem with the BitBLT engine. Try the "NoAccel" or one of the XAA acceleration options discussed above. Also check the BIOS settings. It is also possible that with a high dot clock and depth on a large screen there is very little bandwidth left for using the BitBLT engine. Try reducing the clock.

Chipset is not detected.

Try forcing the chipset to a type that is most similar to what you have.

The screen is blank when starting X

One possible cause of this problem with older linux kernels is that the "APM_DISPLAY_BLANK" option didn't work correct. Either upgrade your kernel or rebuild it with the "APM_DISPLAY_BLANK" option disabled. If the problem remains, or you aren't using linux, a CRT/LCD or switch to and from the virtual console will often fix it.

Textmode is not properly restored

This has been reported on some configurations. Many laptops use the programmable clock of the 6554x chips at the console. It is not always possible to find out the setting that is used for this clock if BIOS has written the MClk after the VClk. Hence the server assumes a 25.175MHz clock at the console. This is correct for most modes, but can cause some problems. Usually this is fixed by switching between the LCD and CRT. Alternatively the user can use the "TextClockFreq" option described above to select a different clock for the text console. Another possible cause of this problem is if linux kernels are compiled with the "APM_DISPLAY_BLANK" option. As mentioned before, try disabling this option.

I can't display 640x480 on my 800x600 LCD

The problem here is that the flat panel needs timings that are related to the panel size, and not the mode size. There is no facility in the current Xservers to specify these values, and so the server attempts to read the panel size from the chip. If the user has used the "UseModeline" or "FixPanelSize" options the panel timings are derived from the mode, which can be different than the panel size. Try deleting theses options from XF86Config or using an LCD/CRT switch.

I can't get a 320x240 mode to occupy the whole 640x480 LCD

There is a bug in the 6554x's H/W cursor for modes that are doubled vertically. The lower half of the screen is not accessible. The servers solution to this problem is not to do doubling vertically. Which results in the 320x240 mode only expanded to 640x360. If this is a problem, a work around is to remove the "HWcursor" option. The server will then allow the mode to occupy the whole 640x480 LCD.

After a suspend/resume my screen is messed up

During a suspend/resume, the BIOS controls what is read and written back to the registers. If the screen is using a mode that BIOS doesn't know about, then there is no guarantee that it will be resumed correctly. For this reason a mode that is as close to VESA like as possible should be selected. It is also possible that the VGA palette can be affected by a suspend/resume. Using an 8bpp, the colour will then be displayed incorrectly. This shouldn't affect higher depths, and is fixable with a switch to the virtual console and back.

The right hand edge of the mode isn't visible on the LCD

This is usually due to a problem with the "LcdCenter" option. If this option is removed form XF86Config, then the problem might go away. Alternatively the manufacturer could have incorrectly programmed the panel size in the EGA console mode. The "FixPanelSize" can be used to force the modeline values into the panel size registers. Two machines that are known to have this problem are the "HP OmniBook 5000" and the "NEC Versa 4080".

My TFT screen has a reddish tint in 24bpp mode

For 6554x chipsets the server assumes that the TFT bus width is 24bits. If this is not true then the screen will appear to have a reddish tint. This can be fixed by using the "18BitBus" option. Note that the reverse is also true. If the "18BitBus" is used and the TFT bus width is 24bpp, then the screen will appear reddish. Note that this option only has an effect on TFT screens.

SuperProbe won't work with my chipset

At least one non-PCI bus system with a HiQV chipset has been found to require the "-no_bios" option for SuperProbe to correctly detect the chipset with the factory default BIOS settings. The server itself can correctly detect the chip in the same situation.

My 690xx machine lockups when using the "MMIO" option

The 690xx MMIO mode has been implemented entirely from the manual as I don't have the hardware to test it on. At this point no testing has been done and it is entirely possible that the "MMIO option will lockup your machine. You have been warned! However if you do try this option and are willing to debug it, I'd like to hear from you.

My TrueColor windows are corrupted when using the "Overlay" option

Chips and Technologies specify that the memory clock used with the multimedia engine running should be lower than that used without. As use of the HiQV chipsets multimedia engine was supposed to be for things like zoomed video overlays, its use was supposed to be occasional and so most machines have their memory clock set to a value that is too high for use with the "Overlay" option. So with the "Overlay" option, using the "SetMClk" option to reduce the speed of the memory clock is recommended.

I can't start X-windows with 16, 24 or 32bpp

Firstly, is your machine capable of 16/24/32bpp with the mode specified. Many LCD displays are incapable of using a 24bpp mode. Also you need at least a 65540 to use 16/24bpp and at least a 65550 for 32bpp. The amount of memory used by the mode will be doubled/tripled/quadrupled. The correct options to start the server with these modes are

```
startx -- -depth 16            5-6-5 RGB ('64K color', XGA)
startx -- -depth 15            5-5-5 RGB ('Hicolor')
startx -- -depth 24            8-8-8 RGB truecolor
```

or with the HiQV series of chips you might try

```
startx -- -depth 24 -fbbpp 32   8-8-8 RGB truecolor
```

however as XFree86 version 4.0 allows 32bpp pixmaps to be used with framebuffers operating in 24bpp, this mode of operating will cost performance for no gain in functionality.

Note that the "-bpp" option has been removed and replaced with a "-depth" and "-fbbpp" option because of the confusion between the depth and number of bits per pixel used to represent to framebuffer and the pixmaps in the screens memory.

A general problem with the server that can manifested in many way such as drawing errors, wavy screens, etc is related to the programmable clock. Many potential programmable clock register setting are unstable. However luckily there are many different clock register setting that can give the same or very similar clocks. The clock code can be fooled into giving a different and perhaps more stable clock by simply changing the clock value slightly. For example 65.00MHz might be unstable while 65.10MHz is not. So for unexplained problems not addressed above, please try to alter the clock you are using slightly, say in steps of 0.05MHz and see if the problem goes away. Alternatively, using the "`UseVClk1`" option with HiQV chips might also help.

For other screen drawing related problems, try the "`NoAccel`" or one of the XAA acceleration options discussed above. A useful trick for all laptop computers is to switch between LCD/CRT (usually with something like Fn-F5), if the screen is having problems.

If you are having driver-related problems that are not addressed by this document, or if you have found bugs in accelerated functions, you can try contacting the XFree86 team (the current driver maintainer can be reached at <dbateman@club-internet.fr> or <eich@xfree86.org>), or post in the Usenet newsgroup "*comp.windows.x.i386unix*".

7. Disclaimer

XFree86, allows the user to do damage to their hardware with software. Although the authors of this software have tried to prevent this, they disclaim all responsibility for any damage caused by the software. Use caution, if you think the Xserver is frying your screen, TURN THE COMPUTER OFF!!

8. Acknowledgement

The authors of this software wish to acknowledge the support supplied by Chips and Technologies during the development of this software.

9. Authors

`Major Contributors` (In no particular order)

- Nozomi Ytow
- Egbert Eich
- David Bateman
- Xavier Ducoin

`Contributors` (In no particular order)

- Ken Raeburn
- Shigehiro Nomura
- Marc de Courville
- Adam Sulmicki
- Jens Maurer

We also thank the many people on the net who have contributed by reporting bugs and extensively testing this server.

CONTENTS

```
$XFree86: xc/programs/Xserver/hw/xfree86/doc/sgml/chips.sgml,v 3.30 2000/03/05 16:59:11 dawes Exp $
```