Under Construction!

Switchres - minimum dot clock

Most of this information has been gathered from GroovyMame's wiki.

Fun fact about CRTs: they receive more data than they actually display and keep those extra bits outside of the frame at the top and the bottom. This will be relevant for the next paragraph.

The dot clock. Essentially, the amount of pixels drawn per second. You can work this out by multiplying your width, your amount of horizontal lines (this is your height, but a bit larger than it) and frames per second. For example, a theoretical resolution of 320×240 at 60 FPS would be calculated by 320×262.5×60, which equals 5,040,000 pixels per second. This is typically measured in MHz (dot clock), but that's confusing so I'm just going to stick with pixels per second (p/s) for the unit and “bandwidth” for the noun.

The issue with modern graphics cards is that they have a **minimum** required bandwidth to function, anything below that will simply not render (or in some cases, lock up the hardware). Typical minimum bandwidths for these cards are 8 p/s, 12 p/s and 25 p/s. Sometimes the minimum bandwidths will differ depending on which ports are used, digital ports tend to have an even higher minimum bandwidth.

So what can we do about this? The answer is simple:

Super resolutions

CRTs are beautiful in the sense that they don't think in pixels, only lines. So essentially, what we can do is send a **BLOODY HUGE** amount of pixels per line to overcome the minimum required bandwidth for our graphics card. These are what super resolutions are.

It's typically good practice to use a *multiple* of what the native width of the original system was in order to not have pixels overlapping into other pixels when shrunk down by our CRT. However, we need to practice a bit of caution, as the transistors and other components in a CRT can only handle so much p/s before becoming overloaded and unresponsive, or possibly even damaged!

So therefore, we will let Switchres calculate the appropriate super resolution for us. It will work out the super resolution that is both a multiple of the original system's width and that satisfies the minimum required bandwidth for our card. A good starting point is 8 million p/s, increasing if that doesn't work for your hardware.

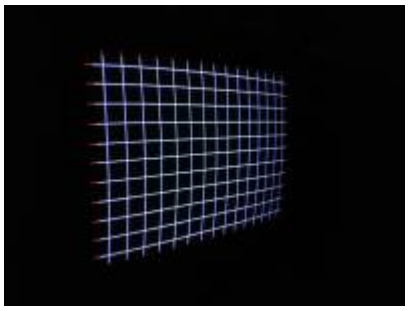
Here's an example: let's say you're playing a game that runs at a native 640×480 at 60 FPS. Our graphics card has a minimum required bandwidth of 25 Mp/s. The bandwidth of that is calculated by 640x480x60 = 18,432,000 p/s. This is close to the required 25 Mp/s but not quite. Switchres

calculates our super resolution as 1280×480, as 1280 is a *multiple* of 640 and the total bandwidth is 1280x480x60 = 36,864,000 p/s. This is well above our required minimum, and thus suited for our purposes.

Same goes for any other modeline, like for 256×224 it would calculate 512x224x60, 768x224x60 and so on until it meets the minimum required bandwidth (dot clock) set in the INI.

Super resolutions are only supported by MAME and Libretro cores.

Testing modelines



configure switchres.ini to your monitor (WinScp)

Connect to Batocera via SSH

Have a keyboard connected

Exit Emulation Station

```
/etc/init.d/S31emulationstation stop
```

or

Exit to terminal `ctrl+alt+f3`

From putty type for this to test modeline 640×480@60 with configured monitor profile

```
DISPLAY=:0 switchres 640 480 60 -i switchres.ini -s -l grid
```

You can change between 2 types of grids with Tab.

Exit pressing Escape from keyboard to try a new modeline

Go back to Es with command

```
/etc/init.d/S31emulationstation start
```

or restart with

```
reboot
```

Modelines to test your switchres.ini settings with

```
dotclock_min 0 (Default) 0-25 with decimal support 1.1, 8.3, 10.5 and so on
```

Arcade Modelines

For example the Arcade game Dottori Kun (dotrikun)

```
DISPLAY=:0 switchres 128 192 61.035156 -i switchres.ini -s -l grid
```

AMD Troubleshooting Tearfree and Freezes

If you get uneven frame-pacing and microstutters in Mame and/or Retroarch we need to disable the option **TearFree** in the driver configuration file located at `/etc/X11/xorg.conf.d`

- 20-amdgpu.conf (GCN 3, GCN 4, GCN 5, RDNA & RDNA 2) or download an already fixed 20-amdgpu.conf

```
Section "OutputClass"
     Identifier "Fix AMD Tearing"
     Driver "amdgpu"
     MatchDriver "amdgpu"
     Option "TearFree" "false"
EndSection
```

- 20-radeon.conf (TeraScale and older, GCN 1 & GCN 2) or download an already fixed 20-radeon.conf

```
Section "OutputClass"
     Identifier "Fix AMD Radeon Tearing"
     Driver "radeon"
     MatchDriver "radeon"
     Option "TearFree" "false"
EndSection
```

Don't forget to `batocera-save-overlay` and reboot.

Games freeze or EmulationStation freeze.

This has been reported by some users and myself included. It's not know yet what is the underlying reason. It could be a driver or kernel issue.

We need to add **Option “DRI” “2” & 3** in the driver configuration file located at `/etc/X11/xorg.conf.d`

- 20-amdgpu.conf (GCN 3, GCN 4, GCN 5, RDNA & RDNA 2) or download an already fixed 20-

amdgpu.conf

```
Section "OutputClass"
     Identifier "Fix AMD Tearing"
     Driver "amdgpu"
     MatchDriver "amdgpu"
     Option "TearFree" "false"
     Option "DRI" "3"
EndSection
```

- 20-radeon.conf (TeraScale and older, GCN 1 & GCN 2) or download an already fixed 20-radeon.conf

```
Section "OutputClass"
     Identifier "Fix AMD Radeon Tearing"
     Driver "radeon"
     MatchDriver "radeon"
     Option "TearFree" "false"
     Option "DRI" "2"
EndSection
```

Don't forget to `batocera-save-overlay` and reboot.

- Short explanation of expression mentioned here

TearFree is a tearing prevention option which prevents tearing by using the hardware page flipping mechanism

Microstutter short irregular frame dips

Frame-pacing uneven distribution of frames