Overscan

From Nesdev wiki

The NES PPU always generates a 256x240 pixel picture. But there is a recurring misconception that the picture is 256x224 pixels on NTSC. This article explains how both are true.

Consumer TV monitors draw the picture slightly larger than the screen, with some of the picture hidden by the border. This is called "overscan".[1] (http://www.mastersofcinema.org/reviews/03lookingbeyond.htm)[2] (http://en.wikipedia.org/wiki/Overscan) The BBC recommends keeping a 5 percent margin inside the screen and putting all important information, such as text, within the center 90 percent of the width and height of the screen.[3] (http://www.bbc.co.uk/guidelines/dq/pdf/tv/tv_standards_london.pdf) Most TVs show slightly more than 90 percent, so make sure to keep any glitch pixels well outside this margin.

Many games appear to rely on the top 8 lines of the picture being hidden, for instance using a "negative" Y scroll to place the top of the nametable 8 lines down, but causing garbage to appear in the overscan area above it. See *Teenage Mutant Ninja Turtles* for a common example (title screen and horizontal scrolling segments).

Contents

NTSC

BT.601 (formerly CCIR 601) is a standard for digital processing of component video signals. It describes a way to sample NTSC video at exactly 13.5 MHz to produce 704x480 non-square pixels or 135/11 MHz to produce 640x480 square pixels in the "clean aperture".[1] (The commonly quoted 720x480 is slightly wider to capture signals slightly before and after the clean aperture in case of a sync shift, allowing the signal to be recentered before broadcast. There are still only 704 pixels on a scanline.) This means the signal associated with one scanline is 640/(135/11) µs = 704/13.5 µs = 52.148 microseconds long.

The NTSC color subcarrier is at 39,375,000/11 Hz = 3.5795 MHz. The NES master clock is always 6 times the color subcarrier frequency because there are 12 hues on the NES PPU, and the color generator uses a double-pumped counter to generate the hue signal. So the PPU's pixel rate is 1/4 of the master clock, or 6/4 of the color subcarrier, or 39,375,000*6/4/11 = 5.3693 million pixels per second. (This pixel rate appears to have originated in the TMS9918 VDP used in the ColecoVision. The NTSC Sega Master System VDP and Super NES PPU use this same rate, as does the Sega Genesis VDP in the 256-pixel mode that several multiplatform titles used.)

Multiplying the pixel rate by the scanline length gives 39,375,000*6/4/11*640/(135,000,000/11) = 280 pixels per scanline. The PPU puts signal in 256 of these and a border at the left and right sides. The color of this border is the same as the backdrop color (usually the value in $3F00). This makes the pixel aspect ratio on a 4:3 TV to be

240/280*4/3 = exactly 8:7, or about 1.143:1.

In NTSC video sampled at 13.5 MHz, there are 481.5 non-square pixels from the start of the horizontal sync pulse to the center of the picture. This equals 191.5 NES pixels. But there are 65 pixels from the start of hsync to the left side of the active picture, or 193 to the center. So the NES picture is in theory a pixel and a half to the right of center.

For game developers

For a 280x240 pixel picture, the 90 percent safe area is 252x216 pixels. Some CRT TVs have rounded corners; it's a good idea to keep text away from the corners if possible.

This, however, doesn't give developers a free pass to ignore glitches caused by background mirroring. Actual TVs show about 224 lines of the signal, hence the commonly reported 256x224 resolution. But the vertical position may be slightly off center. Emulators and LCD HDTVs tend to use lines 8 to 231, but some TVs have been seen to show lines 12 to 235. In fact, even some CRT SDTVs made in the 2000s show more of the bottom than the top; this may be so that tickers on cable news channels aren't cut off.

As seen in an interview with the *Zelda* team (http://forums.nesdev.com/viewtopic.php?t=5991), Nintendo's official background planning sheets had a conservative 224x192 pixel title safe area. It reserved 24 lines at the top and bottom and 16 pixels at the left and right as potential overscan. Later in the NES's commercial life, as TVs became manufactured to tighter tolerances, some developers pushed this out farther. The scaled mode of PocketNES, an NES emulator for Game Boy Advance, has 8 pixels of overscan on the left and right, 16 on the top, and 11 on the bottom. If all your graphics are visible in PocketNES, they should be visible on a TV.

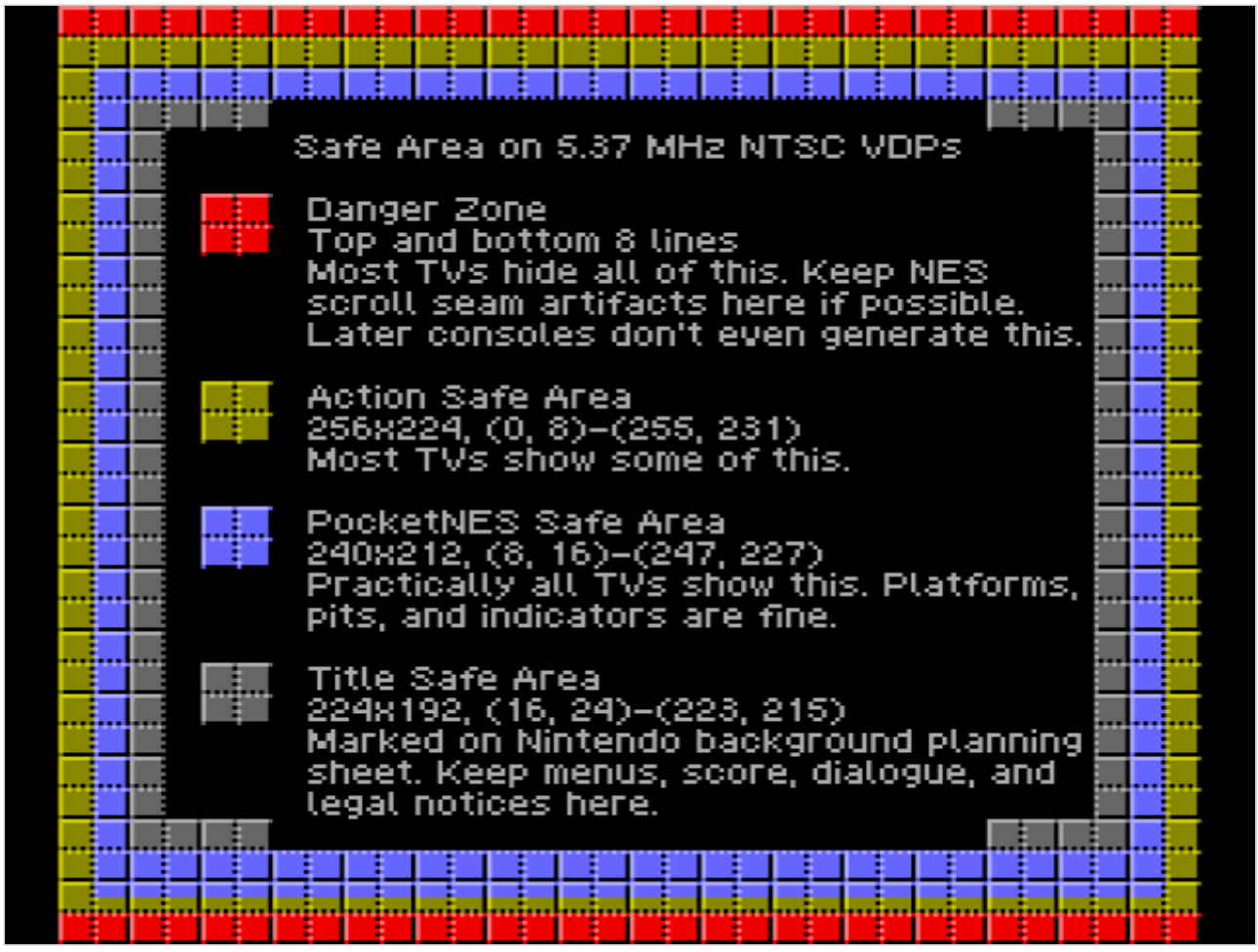

Diagram of action, PocketNES, and title safe areas within the 4:3 frame. (Transcript behind link)

You can preview a game's overscan compliance by pasting a border template image over a 256x224-pixel screenshot from an emulator:

- Border of title safe area
- Border of PocketNES safe area

Since 2007, most new TVs have been 16:9. To prepare your game for zoomed-in mode, assume a 160-line safe area surrounded by an overscan of 40 lines on the top and bottom.

For emulator developers

There are two ways to emulate the pixel aspect ratio of the NES: scale before padding and pad before scaling. The NES PPU hardware performs the padding first, adding 24 pixels of border to form a 280x240 pixel picture that can be resized to 320x240, 640x480, or 960x720 square pixels, or to 352x240 or 704x480 if your SDTV output circuit produces non-square pixels at 13.5 MHz (Rec. 601/DVD dot clock, 132/35*colorburst) or 13.423 MHz (PlayStation dot clock, 15/4*colorburst). But as a slight optimization, you can scale first (256 * 8/7 = 292) and then pad: stretch the 256x240 pixels to 292x240, 584x480, 876x720, or 1168x960 square pixels or 320x240 or 640x480 non-square pixels. Or when enlarging 4.5x to fit a 1080p screen, output 256*(1080/240)*(8/7) = 1316 pixels. Then you can emulate the overscan by drawing a TV bezel on top of the edges of the emulated picture.

Statistics

Here are some numbers of overscan lines from actual TVs, determined using a homebrew paint program on an NTSC NES + PowerPak. Ranges denote fewer hidden lines at top and bottom center (x = 120-135) than at the corners (x = 16-31 and 223-239) due to CRT curvature.

TV	Description	Mode	Top lines	Bottom lines
Vizio VX32L	LCD HDTV	Normal	8	6
Vizio VX32L	LCD HDTV	Zoom	36	35
Magnavox MS2530 C225	CRT SDTV		12-14	8-10
Sylvania 6420FF	CRT SDTV		13-15	8-11

PAL

When sampled at 14.75 MHz (768x576) or 7.375 MHz (384x288), PAL video has square pixels. The width of a scanline is 768 pixels, or 768/14750000 seconds, or 52.068 microseconds.[1]

The PAL color subcarrier is defined as 4,433,618.75 Hz. The PAL NES master clock is six times that, and the PPU generates one pixel for every 5 master clock cycles, or 5320342.5 Hz. This makes the width of a scanline 5320342.5*768/14750000 = 277 pixels, and the pixel aspect ratio 7375000/5320342.5 = (59*125000)/(165*64489/2) ≈ 1.3862:1.

The width of the picture is nearly the same as on NTSC, so we need not reconsider horizontal placement. The border is always black, and it extends into the leftmost and rightmost 2 pixels[2] and the top scanline, the one that never shows sprites.[3] However, the total picture is 288 lines tall, making the safe area roughly 260 lines tall. This means every TV shows blank bars at the top and bottom of the 240-line active area that the NES generates. So for a

PAL-only title, the developer need not concern himself with TVs cutting off the status bar, but minimizing artifacts caused by nametable mirroring and sprite pop-on at the top becomes paramount. Both the PAL NES and PAL famiclones (such as Dendy) behave this way.[4]

Emulator developers can simulate this system's pixel aspect ratio by stretching the picture to a multiple of 355×240, such as 1065×720 or 1420×960.

References

1. ↑ 1.0 1.1 Chris Pirazzi. "Programmer's Guide to Video Systems (http://lurkertech.com/lg/video-systems/)". *Lurkertech*. Accessed 2015-11-17.
2. ↑ BBS topic: Looking for PAL display that displays ALL pixels (http://forums.nesdev.com/viewtopic.php?f=2&t=6132)
3. ↑ BBS topic: 240p test suite (http://forums.nesdev.com/viewtopic.php?p=159304#p159304)
4. ↑ BBS topic: 240p test suite (http://forums.nesdev.com/viewtopic.php?p=173764#p173764)