

PPU power up state

From Nesdev wiki

In March 2008, Blargg reverse-engineered the power-up/reset state and behavior of the NES PPU, NTSC version.

Initial Register Values

Register	At Power	After Reset
PPUCTRL (\$2000)	0000 0000	0000 0000
PPUMASK (\$2001)	0000 0000	0000 0000
PPUSTATUS (\$2002)	+0+x xxxx	U??x xxxx
OAMADDR (\$2003)	\$00	unchanged ¹
\$2005 / \$2006 latch	cleared	cleared
PPUSCROLL (\$2005)	\$0000	\$0000
PPUADDR (\$2006)	\$0000	unchanged
PPUDATA (\$2007) read buffer	\$00	\$00
odd frame	no	no
OAM	pattern	pattern
NT RAM (external, in Control Deck)	mostly \$FF	unchanged
CHR RAM (external, in Game Pak)	unspecified pattern	unchanged

? = unknown, x = irrelevant, + = often set, U = unchanged

- The PPU comes out of power and reset at the top of the picture. See: PPU rendering.
- Writes to the following registers are ignored if earlier than ~29658 CPU clocks after reset: PPUCTRL, PPUMASK, PPUSCROLL, PPUADDR. This also means that the PPUSCROLL/PPUADDR latch will not toggle. The other registers work immediately: PPUSTATUS, OAMADDR, OAMDATA (\$2004), PPUDATA, and OAMDMA (\$4014).
 - There is an internal reset signal that clears PPUCTRL, PPUMASK, PPUSCROLL, PPUADDR, the PPUSCROLL/PPUADDR latch, and the PPUDATA read buffer. (Clearing PPUSCROLL and PPUADDR corresponds to clearing the VRAM address latch (T) and the fine X scroll. Note that the VRAM address itself (V) is *not* cleared.) This reset signal is set on reset and cleared at the end of VBlank, by the same signal that clears the VBlank, sprite 0, and overflow flags. Attempting to write to a register while it is being cleared has no effect, which explains why writes are "ignored" after reset.
- If the NES is powered on after having been off for less than 20 seconds, register writes are ignored as if it were a reset, and register starting values differ: PPUSTATUS = \$80 (VBlank flag set), OAMADDR = \$2F or \$01, and PPUADDR = \$0001.
- The VBL flag (PPUSTATUS bit 7) is random at power, and unchanged by reset. It is next set around 27384, then around 57165.
- Preliminary testing on a PAL NES shows that writes are ignored until ~33132 CPU clocks after power and reset, 9 clocks less than 311 scanlines. It is conjectured that the first VBL flag setting will be close to 241 * 341/3.2 cycles (241 PAL scanlines); further testing by nocash has confirmed this.
- It is known that after power and reset, it is as if the APU's \$4017 were written 10 clocks before the first code starts executing. This delay is probably the same source of the 9 clock difference in the times for PPU writes being ignored. The cause is likely the reset sequence of the 2A03, when it reads the reset vector.

- ¹: Although OAMADDR is unchanged by reset, it is changed during rendering and cleared at the end of normal rendering, so you should assume its contents will be random.
- During the warmup state, the PPU outputs a solid color screen, usually grey, but it depends on the hardware.
- The Reset button on the Control Deck resets the PPU only on the front-loading NES (NES-001). On top-loaders (Famicom, NES-101), the Reset button resets only the CPU.
- Nametable memory in the Control Deck and CHR RAM in the Game Pak are unchanged by Reset, as are the CHR bank settings of most mappers. Their contents are unspecified at power on.

Best practice

The easiest way to make sure that 29658 cycles have passed, and the way used by commercial NES games, involves a pair of loops like this in your init code:

```

    bit PPUSTATUS ; clear the VBL flag if it was set at reset time
vwait1:
    bit PPUSTATUS
    bpl vwait1      ; at this point, about 27384 cycles have passed
vwait2:
    bit PPUSTATUS
    bpl vwait2      ; at this point, about 57165 cycles have passed

```

Due to the \$2002 race condition, alignment between the CPU and PPU clocks at reset may cause the NES to miss an occasional VBL flag setting, but the only consequence of this is that your program will take one frame longer to start up. You might want to do various other initialization, such as getting the mapper and RAM into a known state, between the two loops.

See also

- CPU power up state

References

- Confirmation by nocash (<http://forums.nesdev.com/viewtopic.php?p=99837#p99837>)
- Notes on reset color (<http://forums.nesdev.com/viewtopic.php?f=9&t=4279>)
- PPU warmup testing by Blargg (<http://forums.nesdev.com/viewtopic.php?f=2&t=3958>)

Retrieved from "https://wiki.nesdev.com/w/index.php?title=PPU_power_up_state&oldid=11010"

-
- This page was last modified on 28 May 2015, at 14:00.