PPU Player

breaknes/BreaksPPU/PPUPlayer at main · emu-russia/breaknes (github.com)

What is it

Essentially a PPU emulator

Architecture

```
┌──────────────┐
│  PPU Regs    │
│  Dump        │──────┐
└──────────────┘      │
                      ↓
              ┌──────────────────┐        ┌──────────────────────┐
              │ PPU Player (managed │◄──────►│ PPUSim (native part) │
              │        UI)         │        │                      │
              └──────────────────┘        └──────────────────────┘
                      ↑
┌──────────────┐      │
│  .nes ROM    │──────┘
└──────────────┘
```
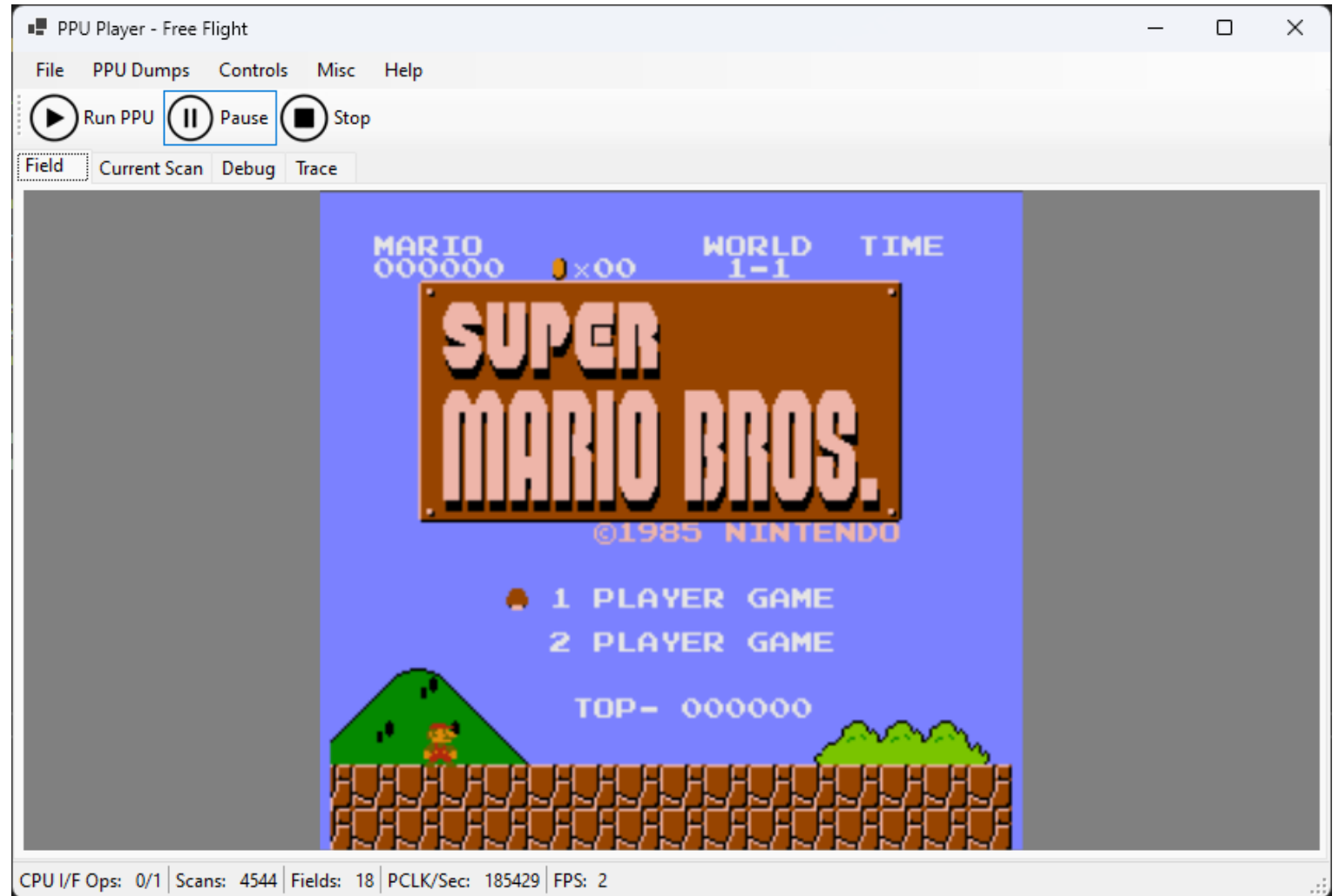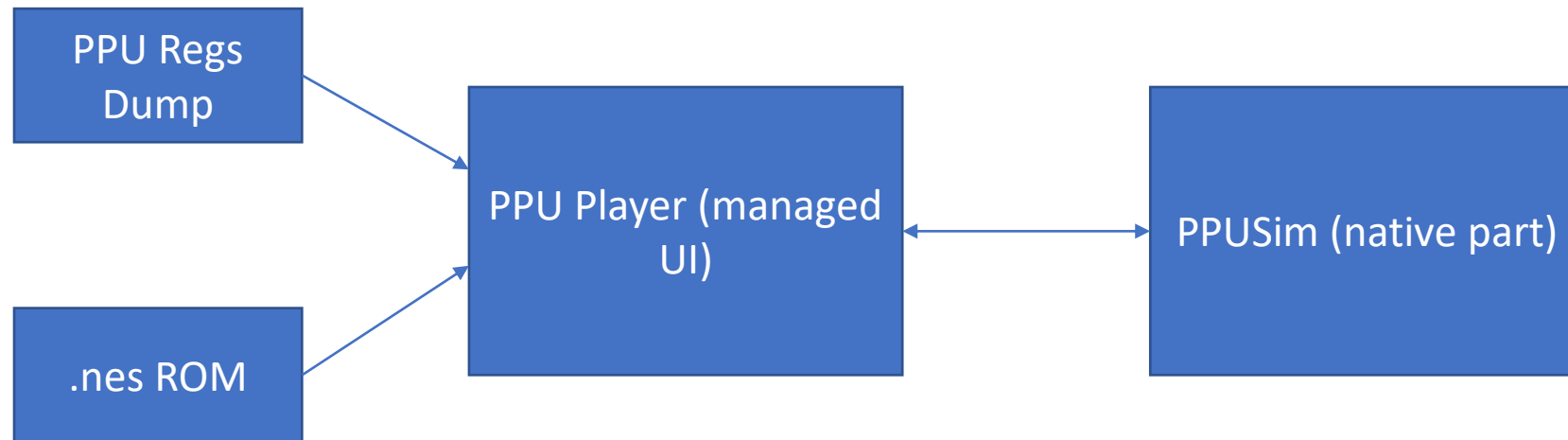
A little more to make it clearer

A special version of the Nintendulator, which collects writes to the PPU registers in a dump (.bin file)

dump of writes into registers

ppu_dump.bin

PPUPlayer looks at the PCLK and writes to the PPU register when the moment comes

C# app

Image screen;
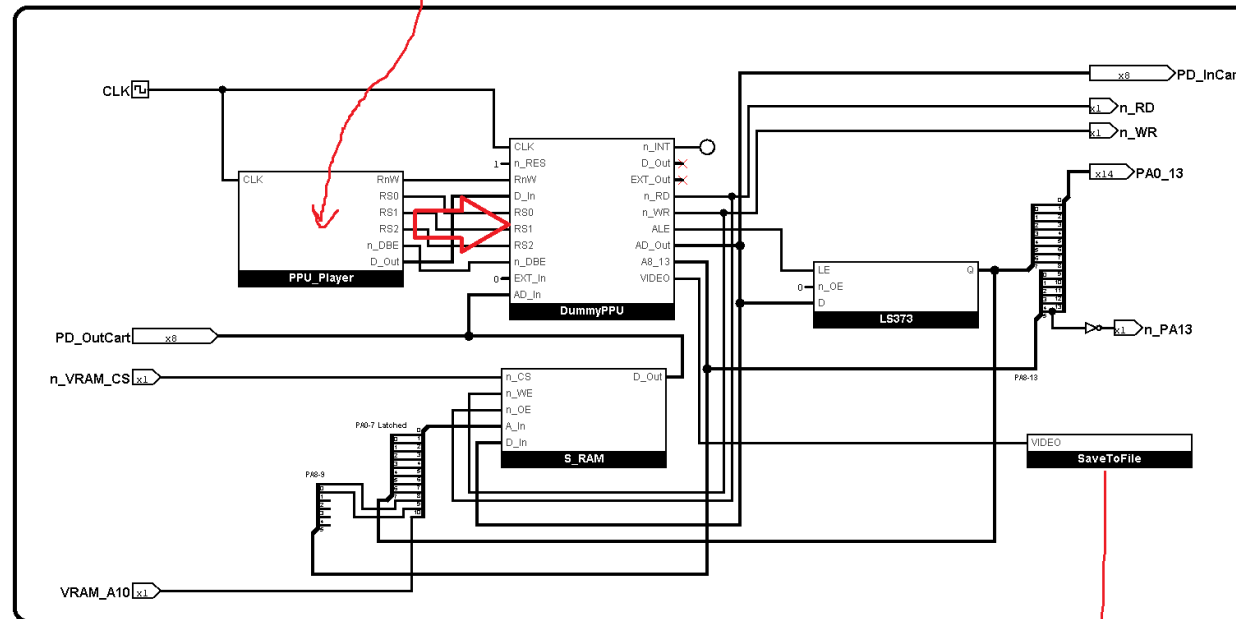
It sits in a .DLL
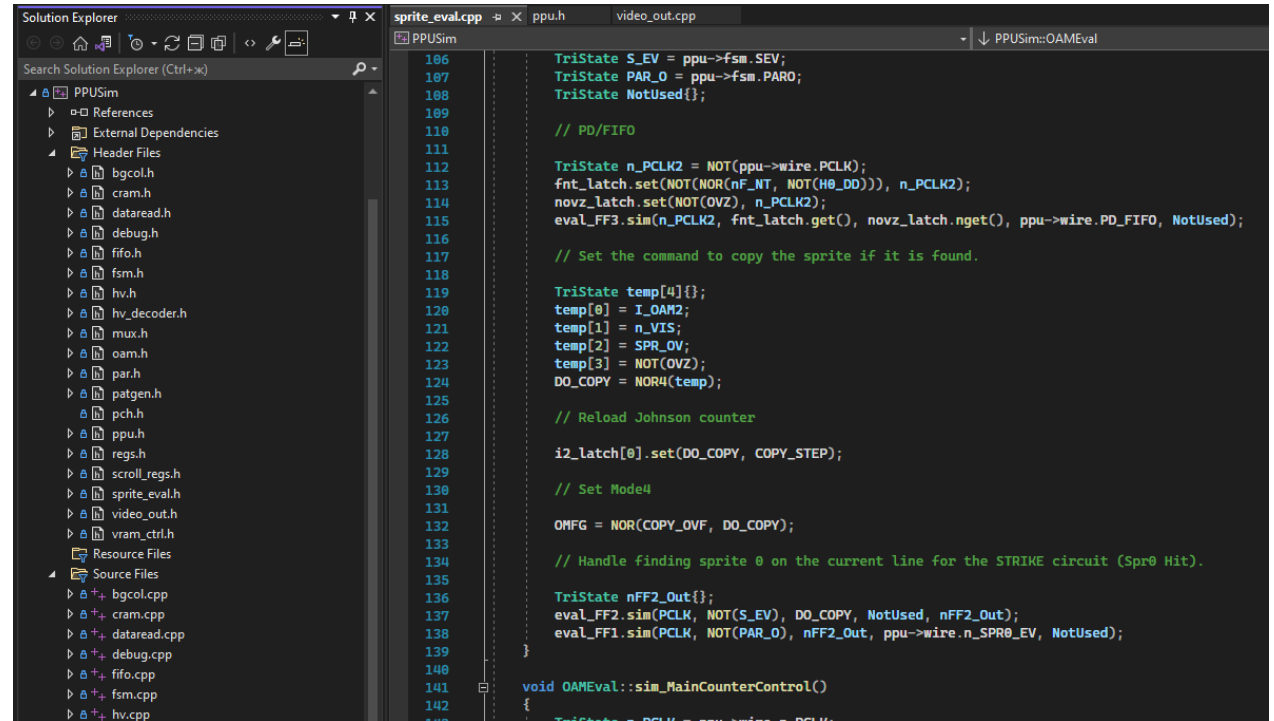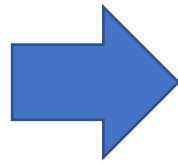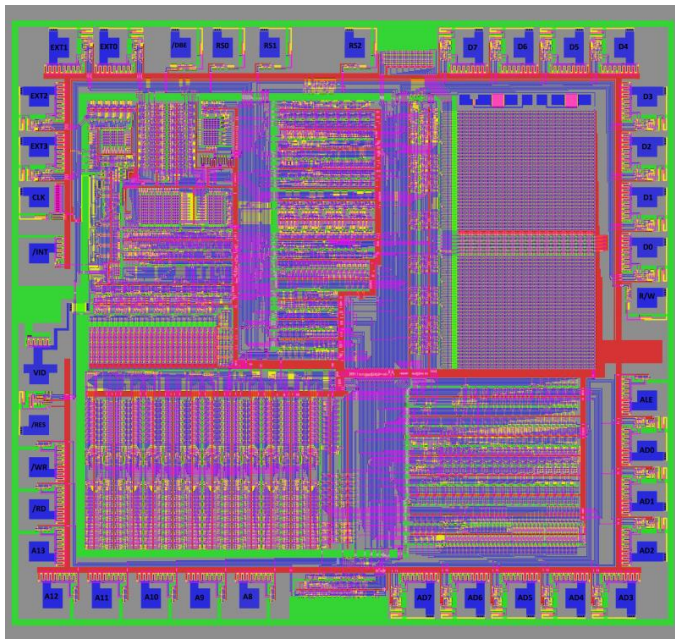
Samples are sampled.
As it is sampled for 1 frame, it is output as Image.

Somewhere in here are the rest of the contacts, but that's not so interesting anymore...

It's a mess too, I can't make out how the tracks go.

not required

bomberman.nes
(or any other on Mapper 0)

PPUSim

- A full-fledged C++ PPU simulator at the logic gate level

.nes ROM image

- Simple dumps are supported, based on mapper 0 (NROM)
- Bomberman, Super Mario etc.

RegDump

- Dump of access to PPU registers on the CPU side
- Array of records in simple format

```
#pragma pack(push, 1)
struct PPULogEntry
{
        uint32_t        pclkDelta;      // Delta of previous PCLK counter value at the time of accessing to the register
        uint8_t         reg;            // PPU register index (0-7) + Flag (msb - 0: write, 1: read)
        uint8_t         value;          // Written value. Not used for reading.
        uint16_t        padding;        // Not used (yet?)
};
#pragma pack(pop)
```

Free Flight Mode

- Press «Run PPU» and go
- PPU register dump or NES ROM are not required
- PPU outputs what is loaded in VRAM / CHR / OAM

Settings

Settings	— □ ✕

Board Features
OAMDecay	Keep
PPU_Revision	RP2C02G
PpuRAWMode	False
ResetPPU	False

Debug
ColorDebug	True
RenderAlwaysEnabled	False
TraceCollapseSameRows	False
TraceEnable	False
TraceFilter	CLK;/CLK;PCLK;/PCLK;RES
TraceMaxFields	2
TraceTimeScale	23

Misc
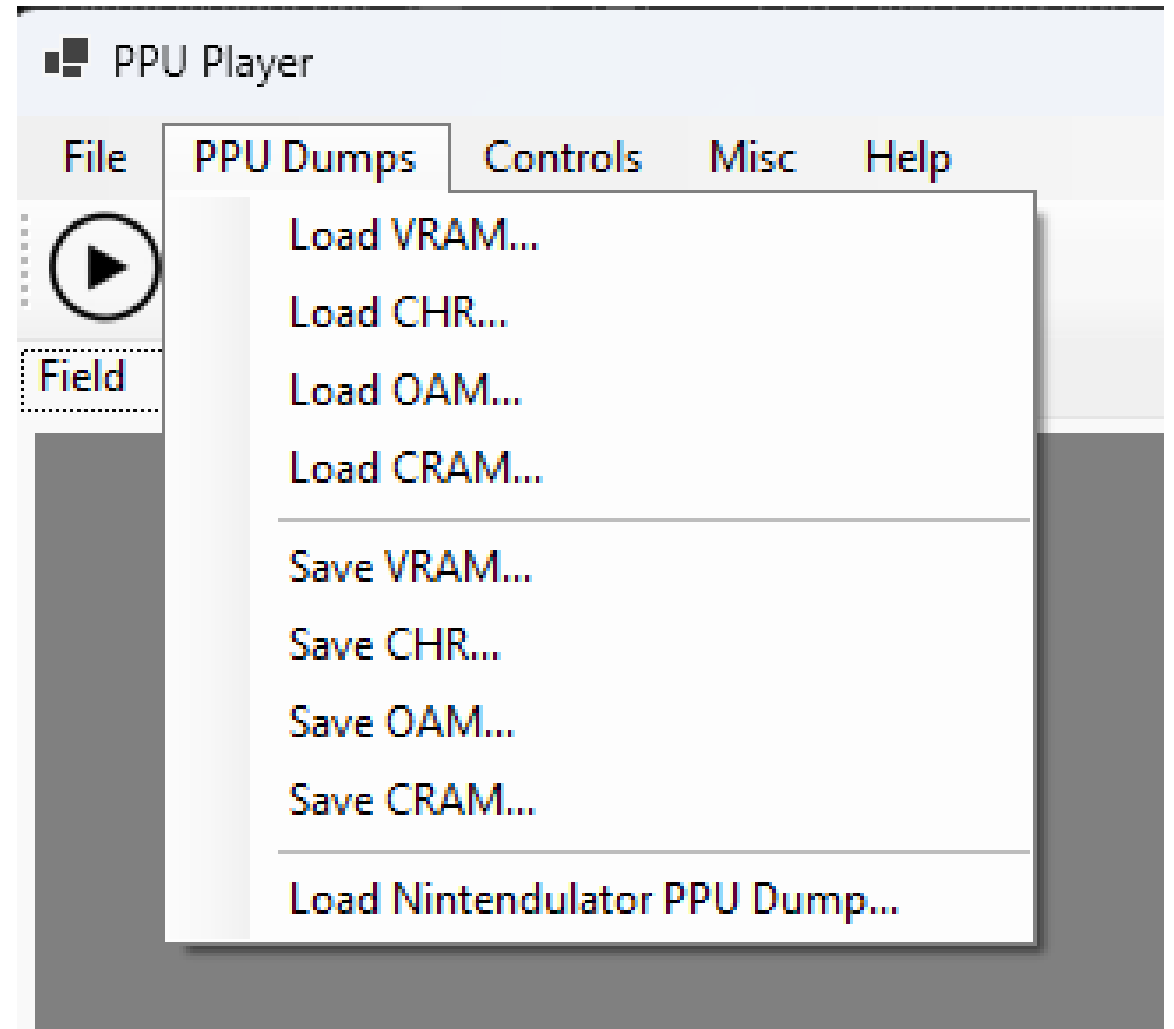|	
FreeModeVMirroring	False

ColorDebug
Show color in the VRAM/Objects viewer. The CRAM currently in use will be loaded.
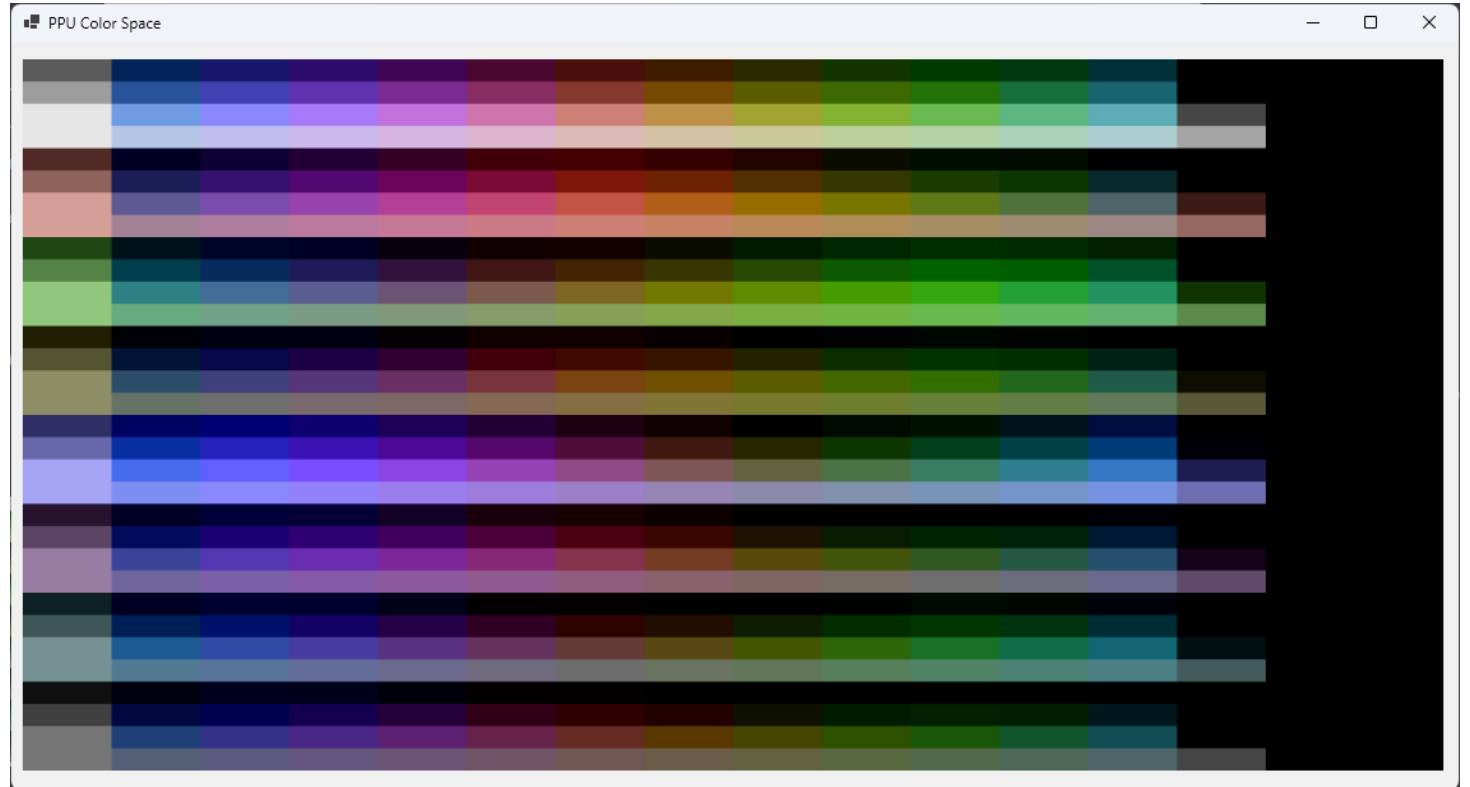
Save

Working with PPU memory dumps

- VRAM / CHR / OAM / CRAM can be loaded and saved
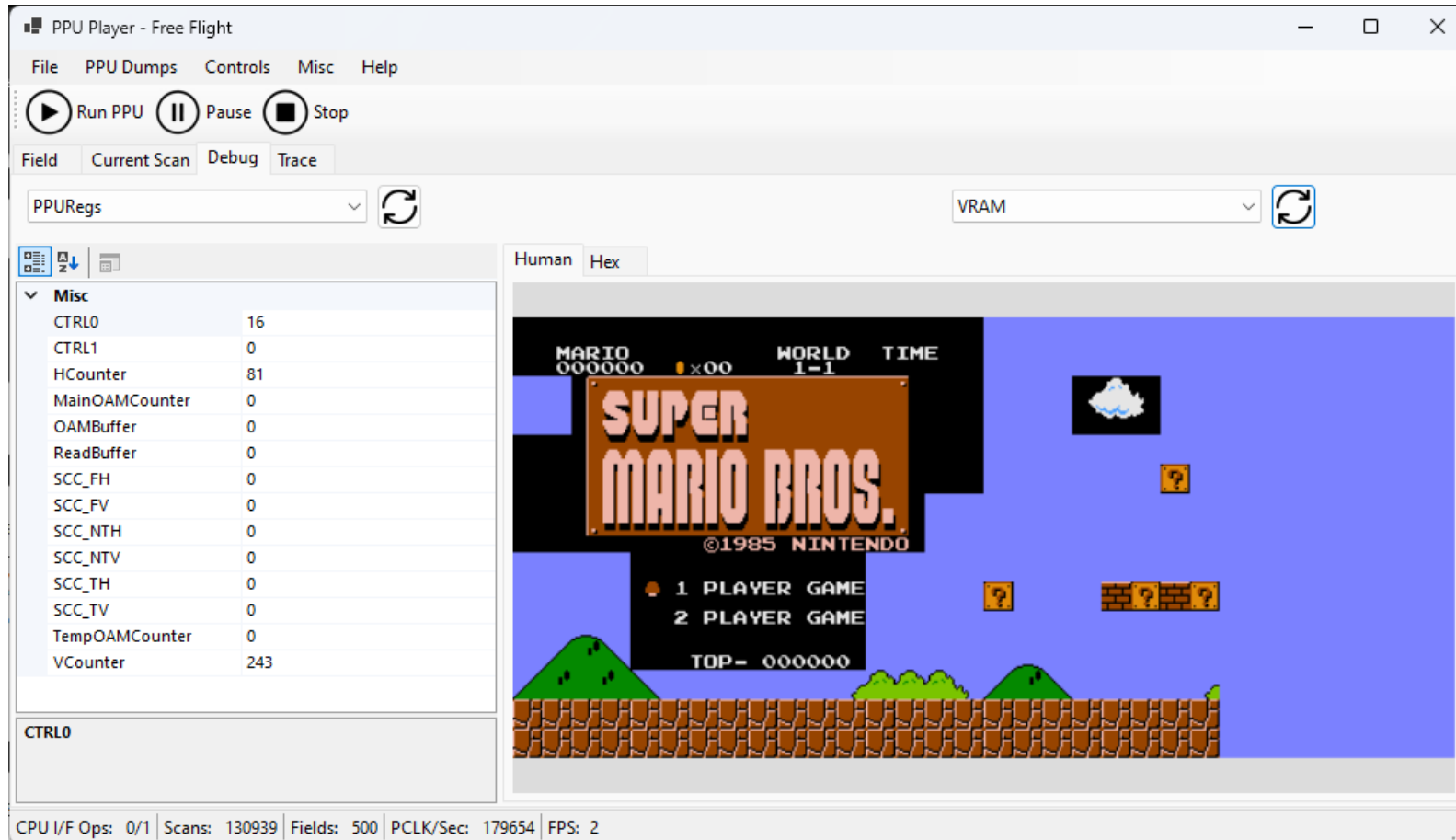- You can load a PPU memory dump from the Nintendulator

PPU Color Space

- Shows the color space ("palette") of the PPU
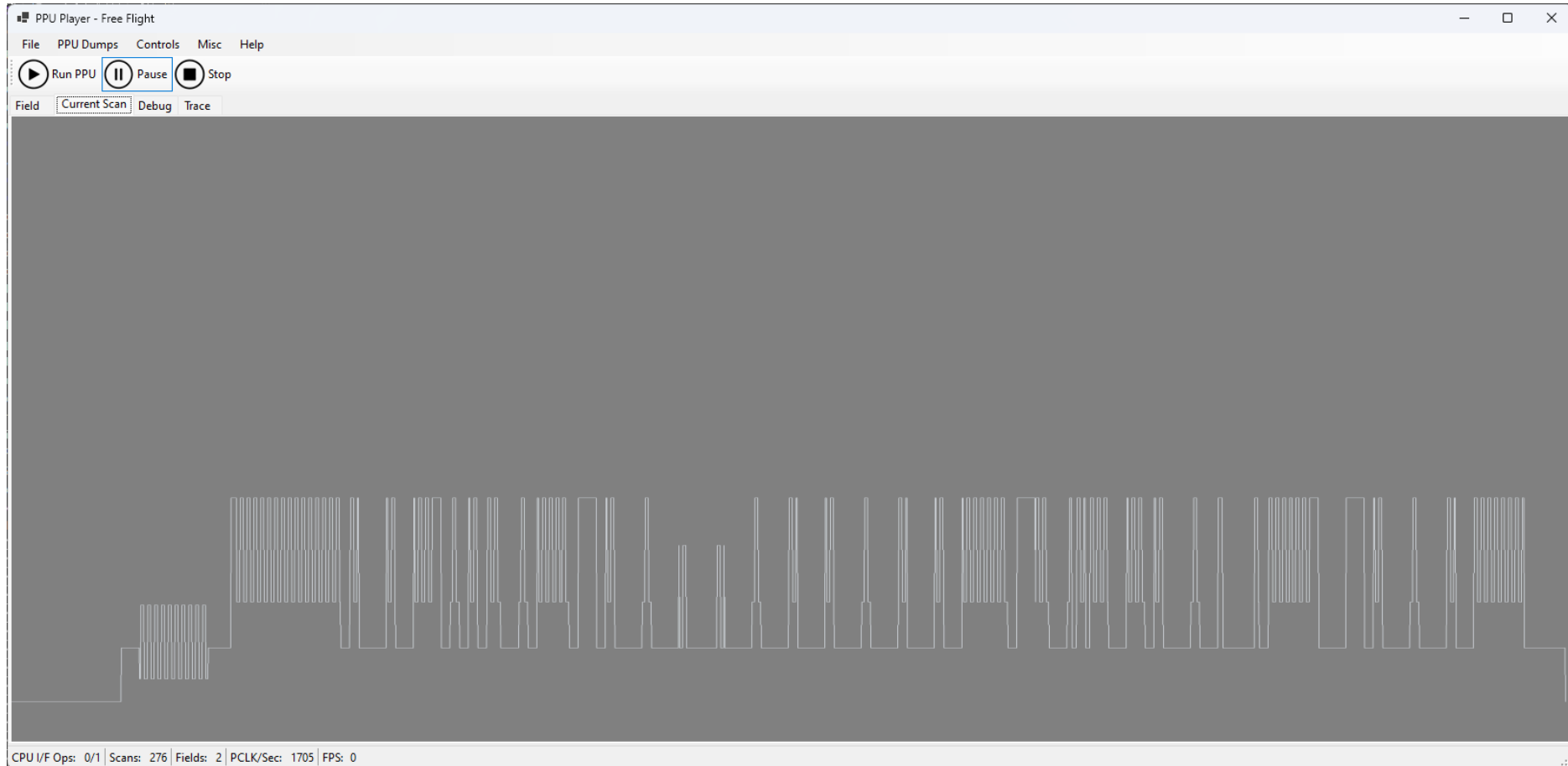
- Including Emphasis

Debugging

Debugging capabilities

- Watch the state of PPU signals
- Watch the state of the PPU registers
- Change the value of PPU registers $2000/$2001 on the fly
- Watch VRAM / CHR / OAM / Temp OAM / CRAM memory in Hex
- Watching memory in a "human" representation

Current video scanline

(*) only for composite PPUs

Questions?

- Ask @org / Discord