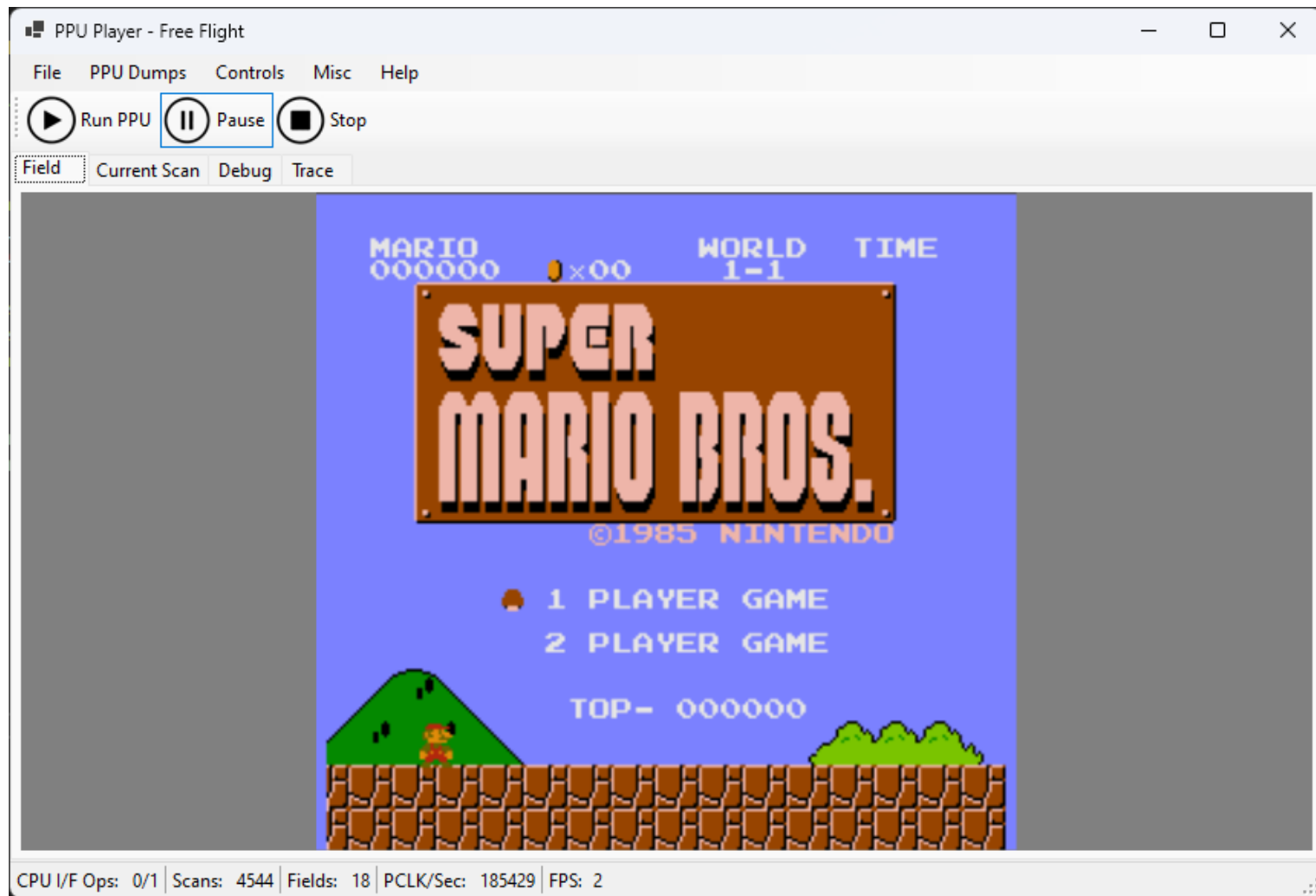


PPU Player

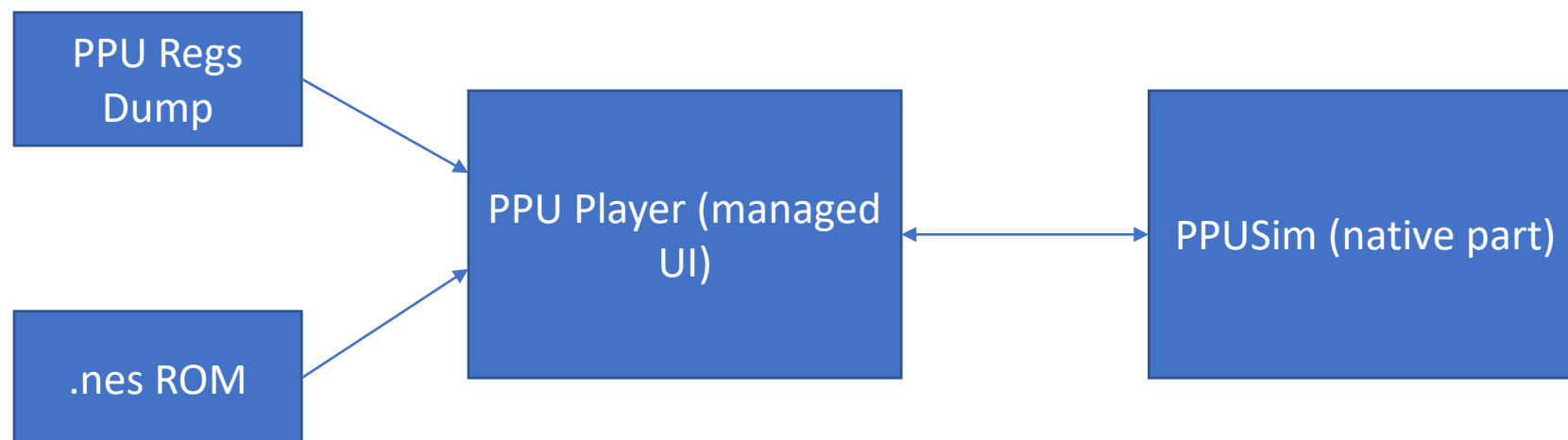
[breaknes/BreaksPPU/PPUPlayer at main · emu-russia/breaknes \(github.com\)](https://github.com/emurussia/breaknes/tree/main/breaknes/PPUPlayer)

Что это

По сути - эмулятор PPU



Архитектура



Ещё немного, чтобы было понятнее

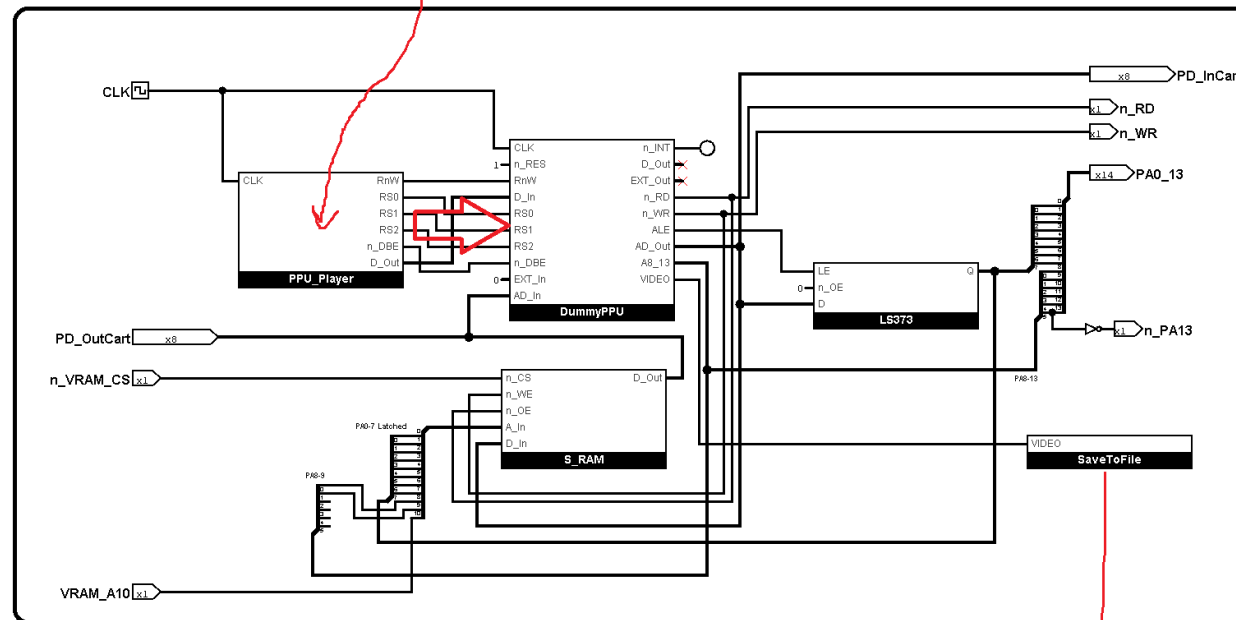
A special version of the Nintendulator, which collects writes to the PPU registers in a dump (.bin file)

dump of writes into registers
ppu_dump.bin

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

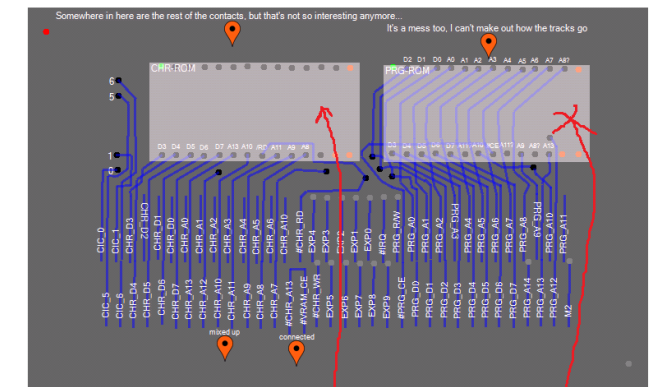
C# app

Image screen;



It sits in a .DLL

Samples are sampled.
As it is sampled for 1 frame, it is output as Image.

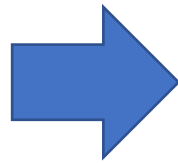
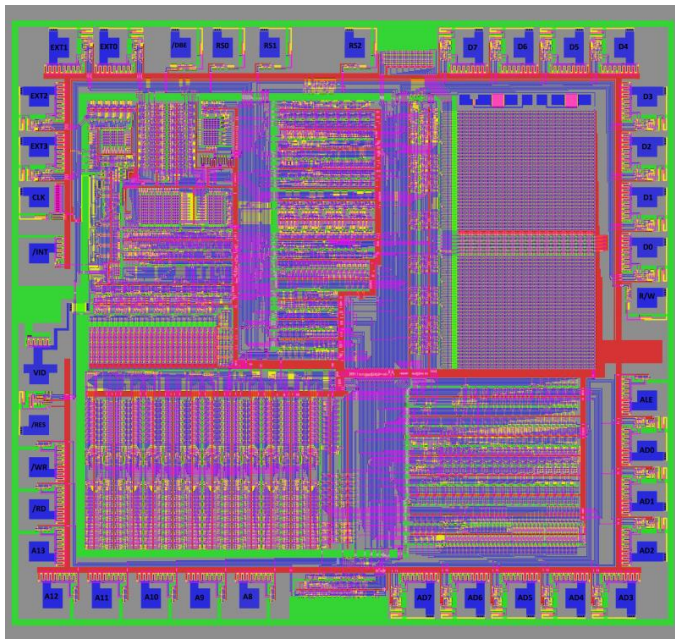


not required

bombberman.nes
(or any other on Mapper 0)

PPUSim

- Полноценный C++ симулятор PPU на уровне логических вентилей



```
106 TriState S_EV = ppu->fsm.SEV;
107 TriState PAR_0 = ppu->fsm.PAR0;
108 TriState NotUsed{};
109
110 // PD/FIFO
111
112 TriState n_PCLK2 = NOT(ppu->wire.PCLK);
113 fnt_latch.set(NOT(NOR(nF_NT, NOT(H0_DD))), n_PCLK2);
114 novz_latch.set(NOT(OVZ), n_PCLK2);
115 eval_FF3.sim(n_PCLK2, fnt_latch.get(), novz_latch.nget(), ppu->wire.PD_FIFO, NotUsed);
116
117 // Set the command to copy the sprite if it is found.
118
119 TriState temp[4]{};
120 temp[0] = I_OAM2;
121 temp[1] = n_VIS;
122 temp[2] = SPR_OV;
123 temp[3] = NOT(OVZ);
124 DO_COPY = NOR4(temp);
125
126 // Reload Johnson counter
127
128 i2_latch[0].set(DO_COPY, COPY_STEP);
129
130 // Set Mode4
131
132 OMFG = NOR(COPY_OVF, DO_COPY);
133
134 // Handle finding sprite 0 on the current line for the STRIKE circuit (Spr0 Hit).
135
136 TriState nFF2_Out{};
137 eval_FF2.sim(PCLK, NOT(S_EV), DO_COPY, NotUsed, nFF2_Out);
138 eval_FF1.sim(PCLK, NOT(PAR_0), nFF2_Out, ppu->wire.n_SPR0_EV, NotUsed);
139
140 void OAMEval::sim_MainCounterControl()
141 {
142     TriState n_PCLK = ppu->wire.n_PCLK;
```

Образ .nes ROM

- Поддерживаются простые дампы, на базе маппера 0 (NROM)
- Bomberman, Super Mario etc.

RegDump

- Дамп доступа к регистрам PPU со стороны CPU
- Массив записей простого формата

```
#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)
```

Режим «свободного полёта»

- Free Flight
- Не требует дампа регистров PPU или .nes
- PPU выводит то что загружено в VRAM / CHR / OAM

Настройки

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

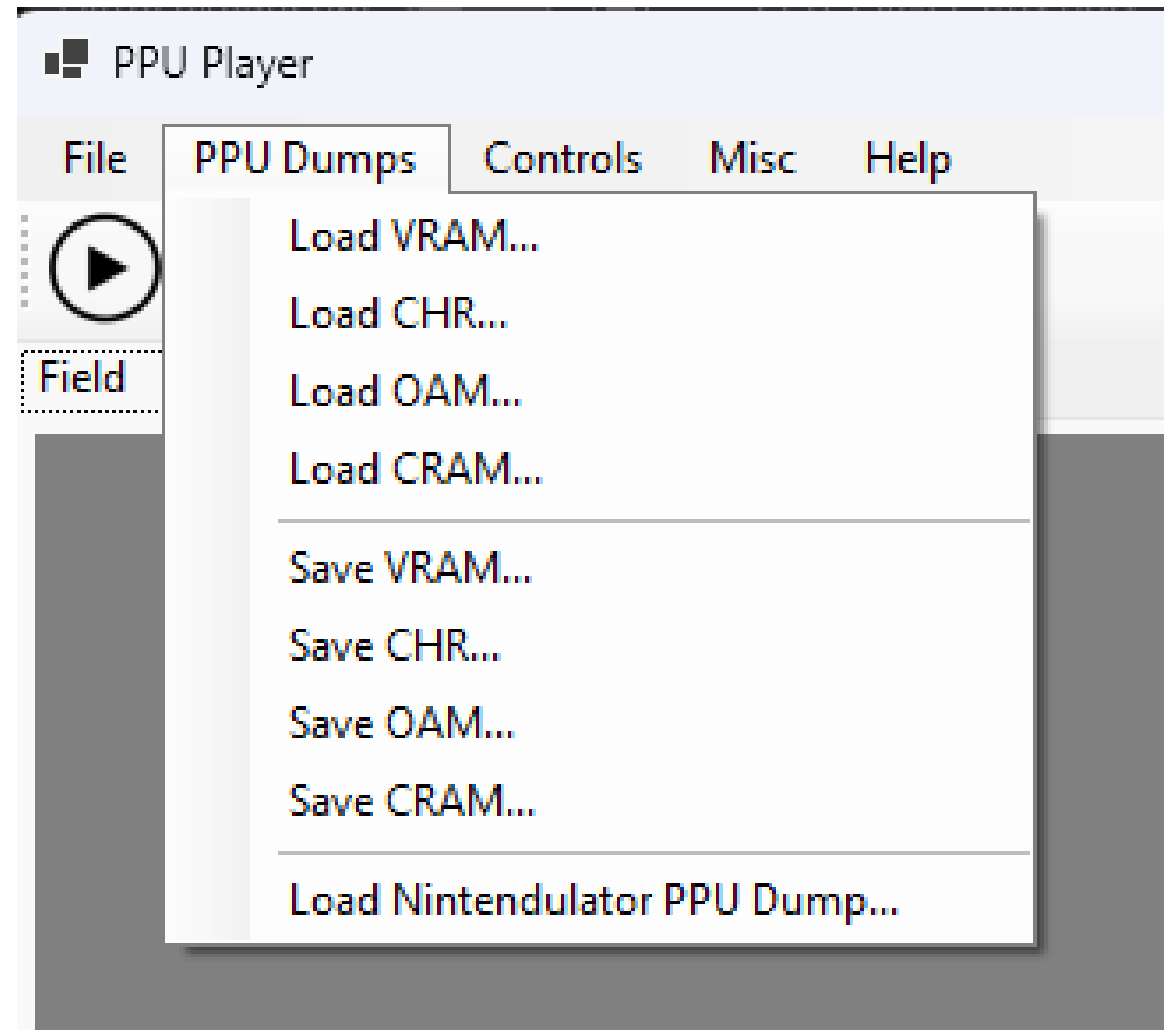
ColorDebug

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

Save

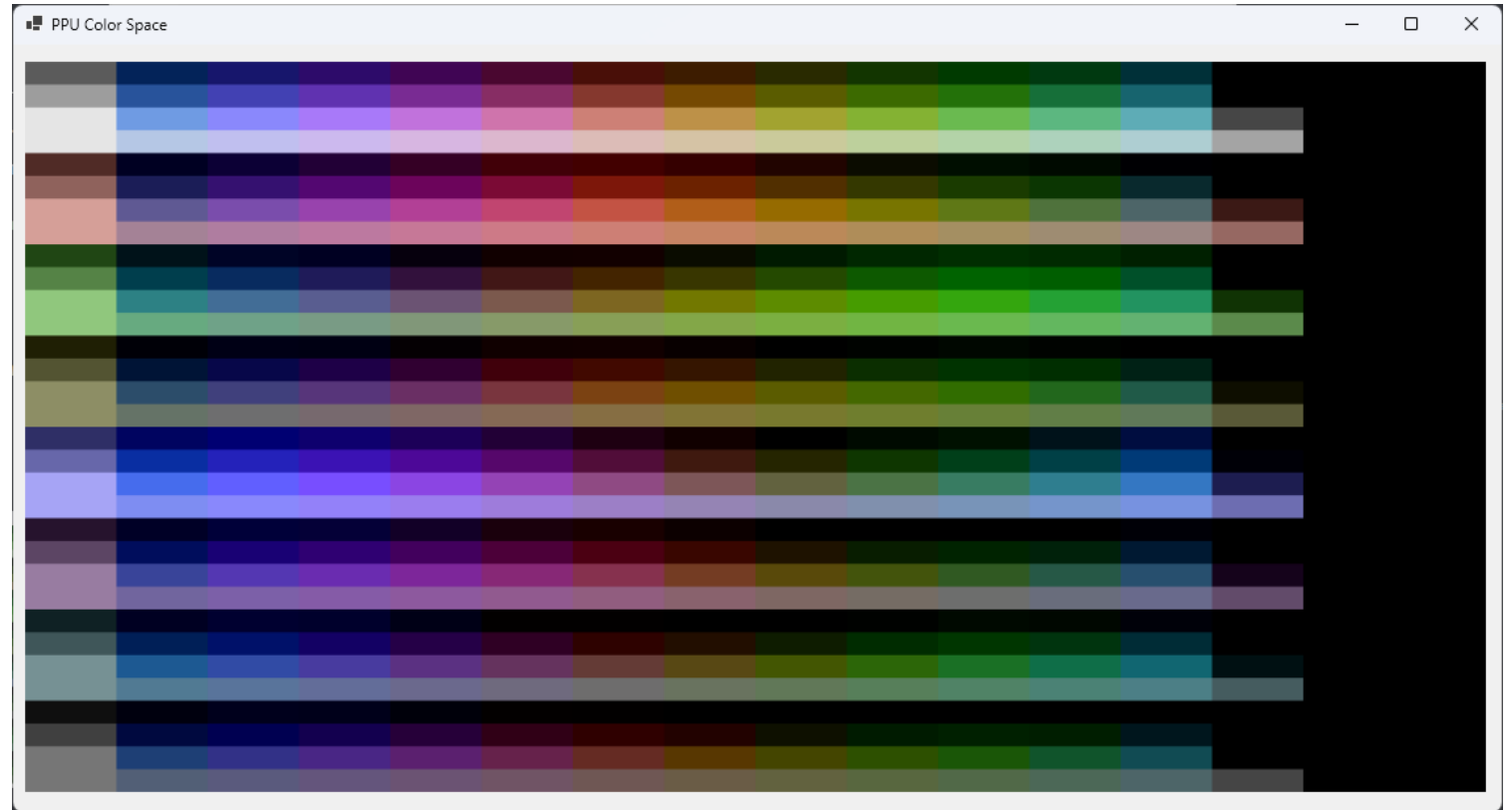
Работа с дампами памяти PPU

- Можно загружать и сохранять VRAM / CHR / OAM / CRAM
- Можно загружать дампы памяти PPU из Nintendulator

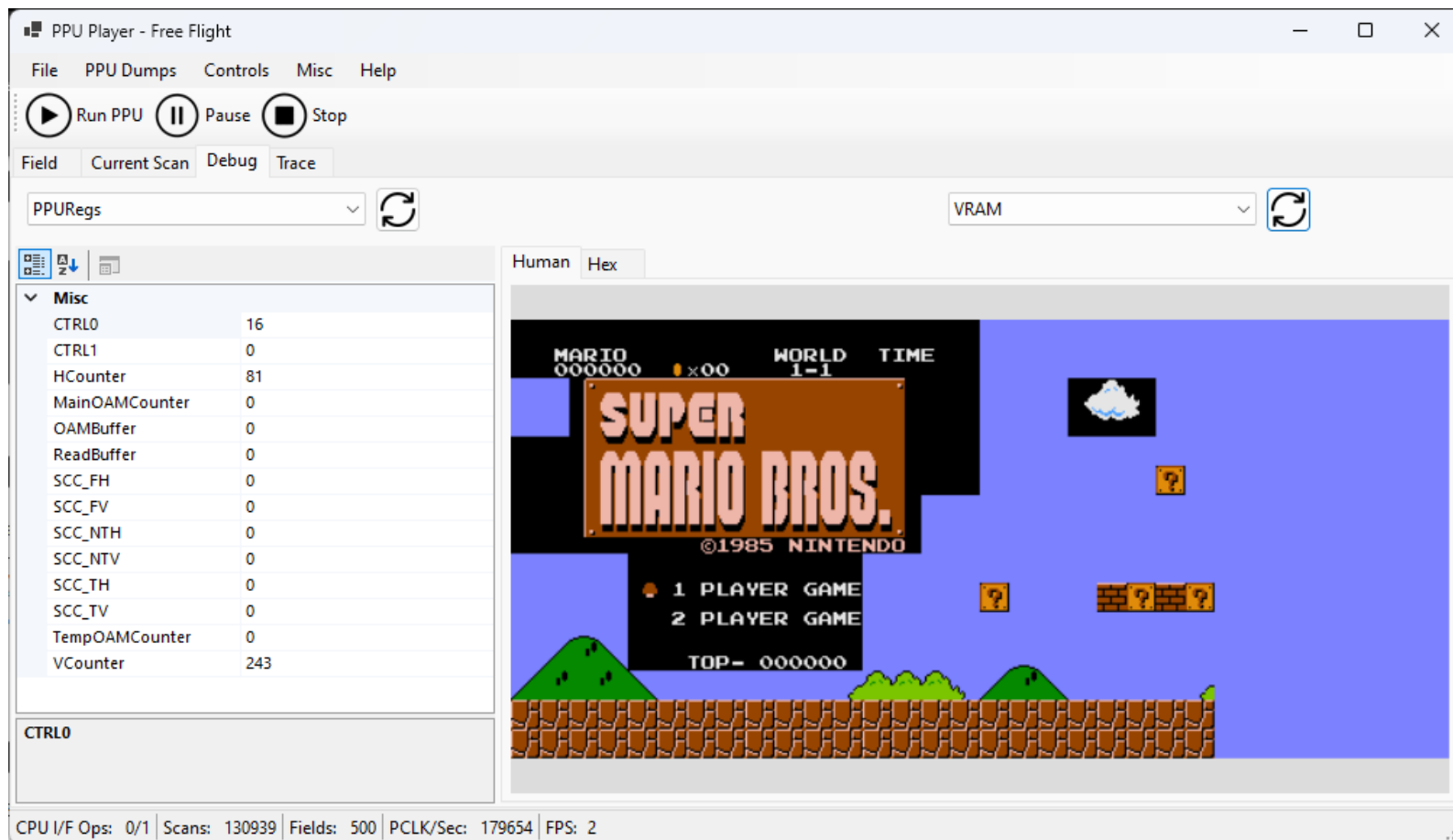


PPU Color Space

- Показывает цветовое пространство («палитру») PPU
- Включая Emphasis



Отладка



Возможности отладки

- Смотреть состояние сигналов PPU
- Смотреть состояние регистров PPU
- Менять значение регистров PPU \$2000/\$2001 на лету
- Смотреть память VRAM / CHR / OAM / Temp OAM / CRAM в Hex
- Смотреть память в «человеческом» представлении

Вопросы?

- Спрашивайте @org / Discord