



SF30th Hacking Edition : A journey into Moo

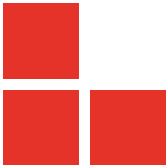


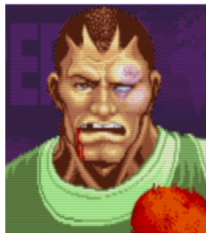
Table of Contents



- 1 Introduction
- 2 From Moo to Arcade
- 3 Play additional games
- 4 Netcode



About me



Pau Oliva Fora - @pof

Position: Senior Security Consultant

Company: IOActive

Description: I enjoy a diverse and challenging role performing penetration testing, reverse engineering and vulnerability discovery.

I only play (and care about) Super Street Fighter 2X.



Nico

Position: Reverse Engineer

Company: Synacktiv

*Description :
French offensive security
company
3 teams : pentest, reverse
engineering, development*



*RE team : focus on low level
dev, reverse, vulnerability
research / exploitation*

SF30th Anniversary Collection



- released in may 2018 on every modern platforms
- developed by Digital Eclipse and edited by Capcom

12 Street Fighter games playable offline



4 Street Fighter games playable online



Content



- Training mode
- Museum
- everything great but... only 12 games to play offline and fewer games with online mode :(

Moo Emulator



- Digital Eclipse uses a custom emulator called “Moo” in some of their games :
 - SF30th Anniversary Collection
 - SNK 40th Anniversary Collection
 - Samurai Shodown Collection (not released yet)
- Arcade emulator written from scratch, **proprietary**



MOAR Games

- Instrumentalize the emulator in order to load additional games

Netcode

- Fix SSF2X turbo speed
- Play different games and enjoy Capcom netcode (:

Why ?



- Because “Moo” is really a great emulator and some games run better than in any other emulators (2x, 3.3, etc.)
- An online mode is provided natively and works smoothly.

↑ SabreAZ 0 points · 1 year ago

↓ 30th anniversary has some of the strongest netcode I have ever seen. It was the fact that before filters, you were getting paired up with people from all around the world, regardless of connection quality. I can play people all over the US and Canada on delay zero, and get clean games. I don't know of any netcode that has such a high threshold. Not even GGPO can do this at zero delay settings

Moo ??



- By looking at the classes names extracted from the RTTI information, the symbol “Moo” appears.

```
0x140353538 .?AVMoo_Sys_StreetFighter@Moo@@@
0x140353568 .?AVGame_StreetFighter_US1@Moo@@@
0x140353640 .?AVMoo_Sys_CPS1@Moo@@@
0x140353668 .?AVGame_StreetFighterII@Moo@@@
0x1403536e0 .?AVGame_StreetFighterII_CE@Moo@@@
0x140353718 .?AVGame_StreetFighterII_HF@Moo@@@
0x140353750 .?AVMoo_Sys_CPS2@Moo@@@
0x140353778 .?AVGame_StreetFighterAlpha@Moo@@@
0x1403537b0 .?AVGame_StreetFighterAlpha2@Moo@@@
0x1403537e8 .?AVGame_StreetFighterAlpha3@Moo@@@
0x140353890 .?AVGame_SuperStreetFighterII@Moo@@@
0x140353910 .?AVGame_SuperStreetFighterIITurbo@Moo@@@
0x1403539c0 .?AVGame_SuperStreetFighterIITournamentBattle_Quad@M...
0x140353a10 .?AVGame_SuperStreetFighterIITournamentBattle@Moo@@@
0x140353ad0 .?AVMoo_Sys_CPS3@Moo@@@
0x140353af8 .?AVGame_StreetFighterIII@Moo@@@
0x140353b28 .?AVGame_StreetFighterIII_2ndImpact@Moo@@@
0x140353b68 .?AVGame_StreetFighterIII_3rdStrike@Moo@@@
0x140353ba8 .?AVMoo_CPU_Base@Moo@@@
0x140353bd0 .?AVMooSH2@Moo@@@
0x140353bf8 .?AVMoo_CPS3_Sound@Moo@@@
0x140353c28 .?AVMoo_Audio_Base@Moo@@@
0x140353c58 .?AVMoo_Audio_OKIMSM5205@Moo@@@
0x140353c88 .?AVMoo_Audio_YM2151@Moo@@@
0x140353cb8 .?AVMoo_Audio_OKIM6295@Moo@@@
0x140353ce8 .?AVMoo_Audio_QSound@Moo@@@
```

Let's google it



- If we google it, there is only one accurate occurrence, a guy that talk about Moo in Arcade1up reddit

Moo emulator



Tous Images Vidéos Shopping Actualités Plus Paramètres Outils

Environ 1 120 000 résultats (0,40 secondes)

MOO Sitio Oficial | Lo Mejor Para Tu Negocio | MOO.com

www.moo.com/

¡Crea Papelería de Empresa Única y Destaca Entre La Multitud! Hazte de notar con Luxe. El Compromiso de MOO. Printfinity. Servicio cliente de lujo. Pagos rápidos y seguros.
[Tarjetas de Visita Luxe](#) · [Postales Promocionales](#) · [Flyers promocionales](#)

MOO emulator and stock controls : Arcade1Up - Reddit

<https://www.reddit.com> > [comments](#) > [moo_emulator_a...](#) > [Traduire cette page](#)

29 mars 2019 - MOO emulator and stock controls. Has anyone else done a side-by-side comparison of games on their stock cabs versus their modified cabs? I've been doing ...

EMULATORS - there are **two three four** different emulations systems in use in - **MAME**, the "MOO" commercial emulator, RetroArch + Libreto + FBA (for Gauntlet on Rampage v1.0.1, MAME for the rest), RetroArch + Libreto + MAME2003 (for Gauntlet on Rampage v1.0.4 & v1.0.5, MAME for the rest):

- **MAME** is v0.139u1 on the 12-in-1, Centipede, Asteroids, & Rampage cabinets. Each cabinet has its own compiled build of MAME, configured to understand that cabinet's control panel layout. (See controls in the pin-out "spreadsheet" above.) If you **add a USB port**, a PC keyboard allows access all the standard MAME options - *including spinner sensitivity*. USB mouse functions as a trackball (at least on 12-in-1)
- **MOO** is a commercial emulator, and appears to be built/licensed per cabinet; **it's hard-coded to support only a specific small number of ROMs** - the games that are in the cabinet. Used in SF2, Galaga, PacMan, and Space Invaders. All future cabinets are very likely to use this same emulator.
- **RetroArch + Libreto + MAME 2003** I have near-zero familiarity with. Who can give me a quick rundown on how it works, how it differs from MAME, how it's similar, etc?
- **RetroArch + FinalBurn Alpha** I have near-zero familiarity with. Who can give me a quick rundown on how it works, how it differs from MAME, how it's similar, etc?

Arcade1Up : Cheap Arcade Cabinet (\$250)



Arcade1up PCB



Home > PCBs > Street Fighter PCB

STREET FIGHTER PCB


LA-815221026582

\$29.99 ~~\$39.99~~

Arcade1Up Printed Circuit Board is the replacement brain for your cabinet.

Easily installed

[ADD TO CART](#) [BUY IT NOW](#)





```
nico@debian ~/WIP/r2con % file MOO-Capcom-ShipMus1-SF
MOO-Capcom-ShipMus1-SF: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked, interpreter /lib/ld-musl-armhf.so.1, stripped
```

ROM strings

- The Arcade1up cabinets use the “Moo” emulator

```
0x000ca364 MOO Emulation Copyright Daniel Filner (moo@tilekiller.com)
0x000caa10 Moo_Audio_Base
0x000caad4 MooUnZipImp::GetFileAsBuffer(%s) couldn't be matched
0x000cab0c MooUnZipImp[%p]::GoToFileByIndex(%d) invalid index
0x000cbf40 Moo_Audio_YM2151
0x000cbf54 Moo_Audio_OKIM6295
0x000e1d70 Moo_Sys_CPS2::ResetNVState() finds no Default Contents
0x000e1e80 Moo_CPU_68000
0x000e1e90 Moo_Audio_QSound
0x000e40f0 Moo_CPU_Z80
```


Moo author



- Experimented developer (+30 years of experience)

Daniel Filner Contribute

Main Credits Biography Portraits

Also Known As


- Dan Filner

Game Credits

Programming/Engineering

Street Fighter: 30th Anniversary Collection (2018)	(Emulation Engineer)
Yu-Gi-Oh!: Legacy of the Duelist (2015)	(Lead Engineer)
Midway Arcade Origins (2012)	(Lead Engineer)
The Simpsons (2012)	(Lead Engineer)
Sonic: Generations (2011)	(Lead Engineer)
Build It Green: Back to the Beach (2010)	(Playground Game Engine)

Portrait



self portrait circa 2000 [add portrait]

Related Sites

- LinkedIn – professional profile [add website]

Table of Contents



- 1 Introduction
- 2 From Moo to Arcade
- 3 Play additional games
- 4 Netcode



Moo -> Mame -> EEPROM -> Arcade



Agenda

1. background
2. motivation
3. demotivation
4. results



Capcom (Digital Eclipse?) promised *arcade perfect* game play

BUT

fixing any glitch or bug causing the games to freeze or reset.





Freeze: Old Honda Throw



<https://www.youtube.com/watch?v=06xuJSVJXeE>



Reset: Sagat Tiger Knee vs CPU Gief

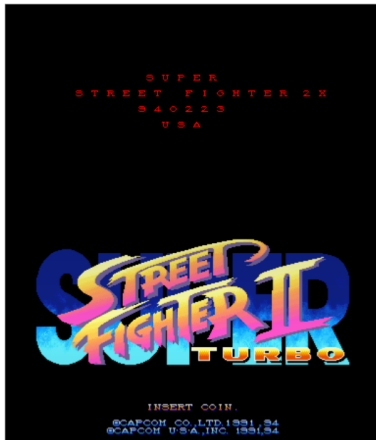
https://www.youtube.com/watch?v=_vPj8fwCLb4



ETC: 940223



USA: 940223

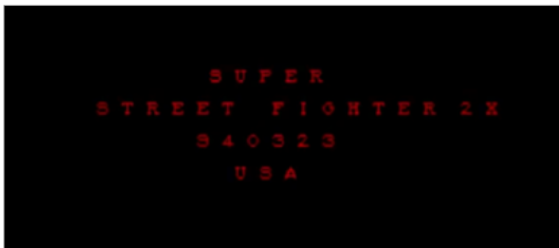


JAPAN: 940223





30th: 940323





Motivation

Possible **undumped** rom!!? :)

Write a quick & dirty PoC to extract it!

```
pol (vim)          pol (vim)          pol (bash)          pol (vim)          pol (vim)
~/bin/bash

sf30th="./30th/"
out="./out"
cps2key="0x94fa8902 0x4c77143f"

# audiocpu - sfx.01 & sfx.02
echo "Extracting audiocpu - sfx.01 & sfx.02"
dd if=${sf30th}/SuperStreetFighterIIITurbo.z80 of=${out}/sfx.01 bs=131072 count=1 skip=0
dd if=${sf30th}/SuperStreetFighterIIITurbo.z80 of=${out}/sfx.02 bs=131072 count=1 skip=1

# qsound - sfx.11m sfx.12m
echo "Extracting qsound - sfx.11m sfx.12m"
dd conv=swab <${sf30th}/SuperStreetFighterIIITurbo.qs > /tmp/SuperStreetFighterIIITurbo.qs.$$
dd if=/tmp/SuperStreetFighterIIITurbo.qs.$$ of=${out}/sfx.11m bs=2097152 count=1 skip=0
dd if=/tmp/SuperStreetFighterIIITurbo.qs.$$ of=${out}/sfx.12m bs=2097152 count=1 skip=1
rm /tmp/SuperStreetFighterIIITurbo.qs.$$

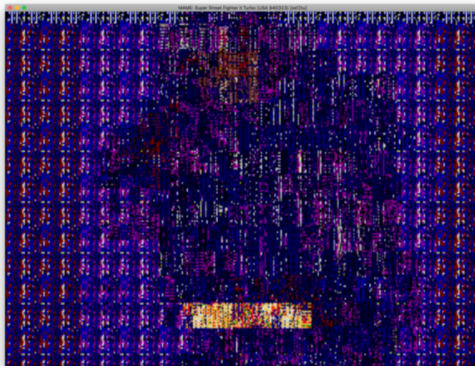
# maincpu
echo "Extracting maincpu "
dd if=${sf30th}/SuperStreetFighterIIITurbo.u.68y of=${out}/d_sfxu.03e bs=0x80000 count=1 skip=0
rahash2 -E cps2 -S "${cps2key}" ${out}/d_sfxu.03e > ${out}/sfxu.03e
dd if=${sf30th}/SuperStreetFighterIIITurbo.u.68y of=${out}/d_sfxu.04a bs=0x80000 count=1 skip=1
rahash2 -E cps2 -S "${cps2key}" ${out}/d_sfxu.04a > ${out}/sfxu.04a
dd if=${sf30th}/SuperStreetFighterIIITurbo.u.68y of=${out}/d_sfxu.05 bs=0x80000 count=1 skip=2
rahash2 -E cps2 -S "${cps2key}" ${out}/d_sfxu.05 > ${out}/sfxu.05
dd if=${sf30th}/SuperStreetFighterIIITurbo.u.68y of=${out}/d_sfxu.06b bs=0x80000 count=1 skip=3
rahash2 -E cps2 -S "${cps2key}" ${out}/d_sfxu.06b > ${out}/sfxu.06b
dd if=${sf30th}/SuperStreetFighterIIITurbo.u.68y of=${out}/d_sfxu.07a bs=0x80000 count=1 skip=4
rahash2 -E cps2 -S "${cps2key}" ${out}/d_sfxu.07a > ${out}/sfxu.07a
dd if=${sf30th}/SuperStreetFighterIIITurbo.u.68y of=${out}/d_sfxu.08 bs=0x80000 count=1 skip=5
rahash2 -E cps2 -S "${cps2key}" ${out}/d_sfxu.08 > ${out}/sfxu.08
dd if=${sf30th}/SuperStreetFighterIIITurbo.u.68y of=${out}/d_sfxu.09 bs=0x80000 count=1 skip=6
rahash2 -E cps2 -S "${cps2key}" ${out}/d_sfxu.09 > ${out}/sfxu.09

# gfx - sfx.13m sfx.15m sfx.17m sfx.19m sfx.14m sfx.16m sfx.18m sfx.20m sfx.21m sfx.23m sfx.25m sfx.27m
"extract.sh" 67L, 4669C
```



Demotivation 1

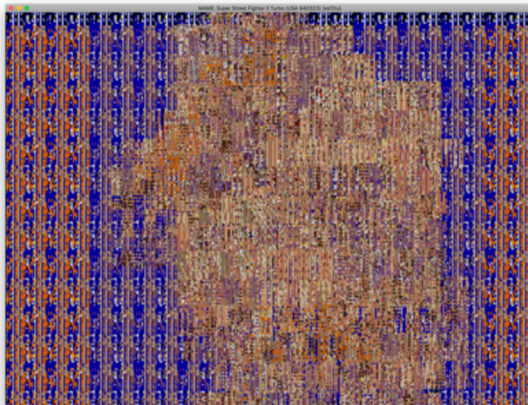
- Converting CPS2 Graphics from MAME <--> MOO was difficult





Demotivation 1

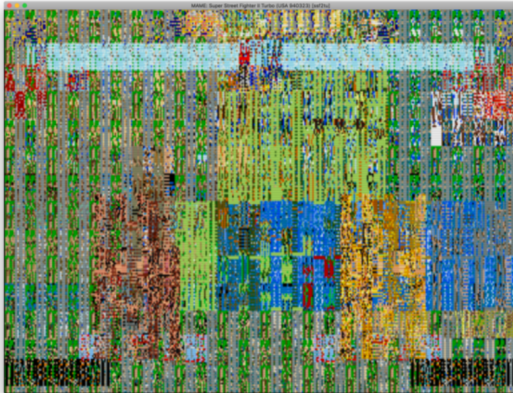
- Converting CPS2 Graphics from MAME <--> MOO was difficult





Demotivation 1

- Converting CPS2 Graphics from MAME <--> MOO was difficult





Demotivation 2

- The rom was already dumped :((((

```
ssf2t,    "Super Street Fighter II Turbo (World 940223)"
ssf2ta,   "Super Street Fighter II Turbo (Asia 940223)"
ssf2th,   "Super Street Fighter II Turbo (Hispanic 940223)"
```

```
ssf2tu, "Super Street Fighter II Turbo (USA 940323)"
```

```
ssf2tur1, "Super Street Fighter II Turbo (USA 940223)"
ssf2xj,   "Super Street Fighter II X: Grand Master Challenge (Japan 940311)"
ssf2xjr1, "Super Street Fighter II X: Grand Master Challenge (Japan 940223)"
ssf2xjr1r, "Super Street Fighter II X: Grand Master Challenge (Japan 940223 rent version)"
```



Changes

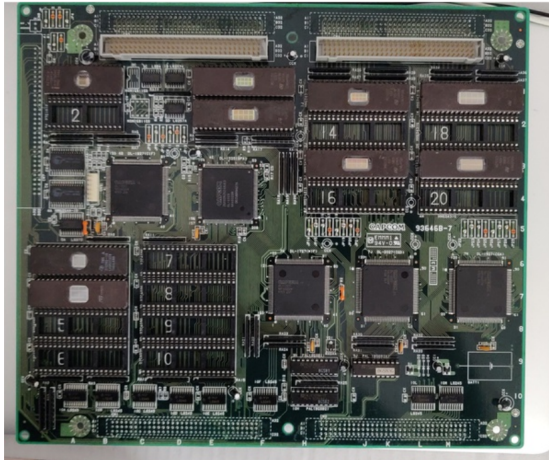




Results

Burn EEPROMs and play the game on real hardware

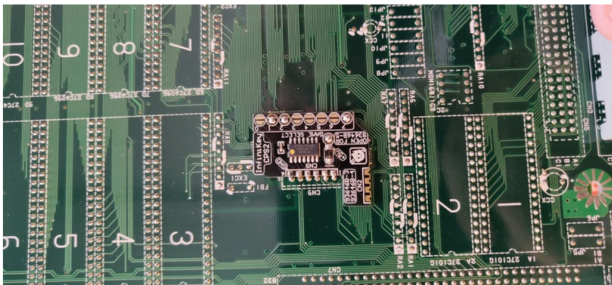
Donor B-board





CPS2 Crypto

Undamned CPS2 InfiniKey used to inject the game's key on the B board





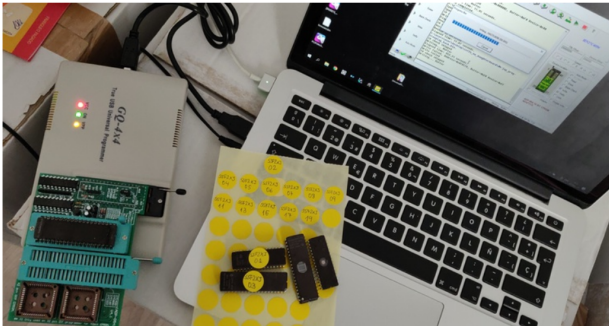
EEPROMS

- 27C1001
- 27C4096
- 27C322





Burn EEPROMs





oops...





TODO



Graphics conversion is a PITA

Table of Contents



- 1 Introduction
- 2 From Moo to Arcade
- 3 Play additional games
- 4 Netcode



Workflow when loading a game

- Init the Game object according to the chosen game. For S2HF, the following object is initialized :
 - **Game_StreetFighterII_HF : Moo_Sys_CPS1 : MooBase**
- Parse and retrieve game assets from the filesystem
- Map the GFXs using bank mappers
- Render graphics, run the 68k emulator with the maincpu rom

Game assets



```
gna@DESKTOP-RRMLEB4:/mnt/c/Program Files (x86)/Steam/steamapps/common/Street Fighter 30th Anniversary Collection/Bundle$ ls
bundleMain.mbundle          bundleStreetFighterIII_3rdStrike.mbundle
bundleStreetFighter.mbundle bundleStreetFighterII_CE.mbundle
bundleStreetFighterAlpha.mbundle bundleStreetFighterII_HF.mbundle
bundleStreetFighterAlpha2.mbundle bundleSuperStreetFighterII.mbundle
bundleStreetFighterAlpha3.mbundle bundleSuperStreetFighterIITurbo.mbundle
bundleStreetFighterII.mbundle bundleTimeline_00.mbundle
bundleStreetFighterIII.mbundle bundleTimeline_01.mbundle
bundleStreetFighterIII_2ndImpact.mbundle bundleTimeline_02.mbundle
```

Mbundle

- The game assets are located into a kind of ordered dictionary files
- These resources are neither compressed nor encrypted
- Loïc *WydD* Petit wrote a script to extract these assets ^a

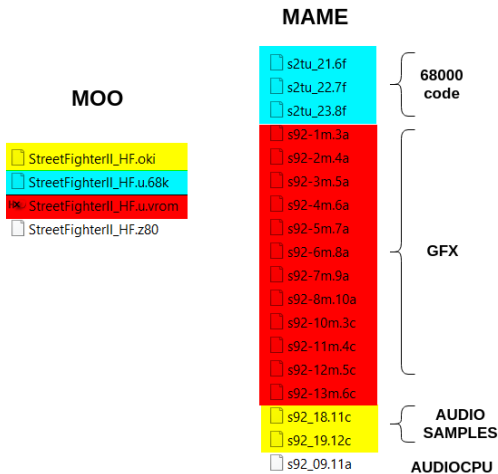
a. <https://github.com/WydD/sf30ac-extractor>



Roms

- By extracting the game assets, we can get the roms data.
- SF30th emulator do not support Mame roms, it works only with plain rom.

Street Fighter II Hyper Fighting roms



Main CPU



```
gna@DESKTOP-RRMLEB4:/mnt/c/Users/jeanvaljean/Documents/test/sf2hfu$ radiff2 -x s2tu_23.8f StreetFighterII_HF.u.68k
File size differs 524288 vs 1572864
offset 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF
0x00000000! 000000000b00e4910b00e0910b00e091 ..... 00000000000b91e4000b91e0000b91e0 .....
0x00000010! 0b00e0910b00e0910b00e0910b00e091 ..... 000b91e0000b91e0000b91e0000b91e0 .....
0x00000020! 0b00e0910b00e0910b00e0910b00e091 ..... 000b91e0000b91e0000b91e0000b91e0 .....
0x00000030! 0070754e0170754e0270754e0370754e .puN.puN.puN.puN 70004e7570014e7570024e7570034e75 p.Nup.Nup.Nup.Nu
0x00000040! 0470754e0570754e0670754e0770754e .puN.puN.puN.puN 70044e7570054e7570064e7570074e75 p.Nup.Nup.Nup.Nu
0x00000050! 0870754e0970754e0a70754e0b70754e .puN.puN.puN.puN 70084e7570094e75700a4e75700b4e75 p.Nup.Nup.Nup.Nu
0x00000060! 0b00e0910b00e0910100e0d10b00e091 ..... 000b91e0000b91e00001d1e0000b91e0 .....
0x00000070! 0b00e0910b00e0910b00e0910b00e091 ..... 000b91e0000b91e0000b91e0000b91e0 .....
0x00000080! 0b0020970b0040970b0066970b009697 ..._...@...f.... 000b9720000b9740000b9760000b9796 ..._...@...f....
0x00000090! 0b00c2970b00ea970b0006980b001a98 ..... 000b97c2000b97ea000b980000b981a .....
0x000000a0! 0b00c980b0098980b00cc980b00e898 ...L..... 000b984c000b9898000b98cc000b98e8 ...L.....
0x000000b0! 0b0026990b002a990b003c990b00e091 ..&...*...<.... 000b9926000b992a000b993c000b91e0 ..&...*...<....
0x000000c0! 2d300c003b320600fb4e02100800ba00 -0...;2...N..... 302d000c323b00064efb100200800ba 0-...;...N.....
0x000000d0! d200c2012d10dc093c325c0001010066 .....<2)...f 00d201c2102d09dc323c005c01016600 .....<2c\...f.
0x000000e0! 14007c3bffff52000061de0100615200 .;|.R...a...aR. 00143b7cffff0052610001de61000052 .;|...Ra...a..R
0x000000f0! 00609c012d0c0600dc0914667c3b0400 .-...-.....f|;.. 6000019c0c2d000609dc66143b7c0004 ~...-.....f;|..
0x00000100! 0c007c1b0200d5090061be0100603200 ..|.....a...^2 000c1b7c000209d5610001be6000032 ...|.....a...^2
0x00000110! 6d540c002d42d50900618601b84e9c66 mT...-B...a...N.f 546d000c422d09d5610001864eb8669c Tm...B...a...N.f.
0x00000120! 1a677c19010000007c192a0020002d0c .g|.....|. * . - 671a197c00010000197c002a00200c2d g...|.....|. * . -
0x00000130! 0400dc0906677c192b00200000618a01 .....g|.+. .a... 000409dc6706197c002b00206100018a .....g|.+. a...
0x00000140! 00702d10dc0940d07b3b24001a003b30 .p-...@...;$. ;0 7000102d09dc0d403b7b0024001a303b p-...@...;$. ;0
0x00000150! 0600f94ef00d000019001a001b001c00 .....N..... 00064ef80df00000019001a001b001c .....N.....
```

Main CPU



```
gna@DESKTOP-RRMLEB4:/mnt/c/Users/jeanvaljean/Documents/test/sf2hfu$ radiff2 -x s2tu_23.8f StreetFighterII_HF.u.68k
File size differs 524288 vs 1572864
  offset      0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF      0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF
0x00000000| 000000000b00e4910b00e0910b00e091 ..... 00000000000b91e4000b91e0000b91e0 .....
0x00000010| 0b00e0910b00e0910b00e0910b00e091 ..... 000b91e0000b91e0000b91e0000b91e0 .....
0x00000020| 0b00e0910b00e0910b00e0910b00e091 ..... 000b91e0000b91e0000b91e0000b91e0 .....
0x00000030| 0070754e0170754e0270754e0370754e .puN.puN.puN.puN 70004e7570014e7570024e7570034e75 p.Nup.Nup.Nup.Nu
0x00000040| 0470754e0570754e0670754e0770754e .puN.puN.puN.puN 70044e7570054e7570064e7570074e75 p.Nup.Nup.Nup.Nu
0x00000050| 0870754e0970754e0a70754e0b70754e .puN.puN.puN.puN 70084e7570094e75700a4e75700b4e75 p.Nup.Nup.Nup.Nu
0x00000060| 0b00e0910b00e0910100e0d10b00e091 ..... 000b91e0000b91e00001d1e0000b91e0 .....
0x00000070| 0b00e0910b00e0910b00e0910b00e091 ..... 000b91e0000b91e0000b91e0000b91e0 .....
0x00000080| 0b0020970b0040970b0066970b009697 ..._...@...f.... 000b9720000b9740000b9760000b9796 ..._...@...f....
0x00000090| 0b00c2970b00ea970b0006980b001a98 ..... 000b97c2000b97ea000b980000b981a .....
0x000000a0| 0b004c980b0098980b00cc980b00e898 ...L..... 000b984c000b9898000b98cc000b98e8 ...L.....
0x000000b0| 0b0026990b002a990b003c990b00e091 ..&...*...<.... 000b9926000b992a000b993c000b91e0 ..&...*...<....
0x000000c0| 2d300c003b320600fb4e02100800ba00 -0...;2...N..... 302d000c323b00064efb100200800ba 0-...;2...N.....
0x000000d0| d200c2012d10dc093c325c0001010066 .....<2>...f 00d201c2102d09dc323c005c01016600 .....<2>...\..f.
0x000000e0| 14007c3bffff52000061de0100615200 .;|.R...a...aR. 00143b7cffff0052610001de61000052 .;|...Ra...a..R
0x000000f0| 00609c012d0c0600dc0914667c3b0400 .-.....f|;.. 6000019c0c2d000609dc66143b7c0004 ~.....f;|..
0x00000100| 0c007c1b0200d5090061be0100603200 ..|.a...^2. 000c1b7c000209d5610001be6000032 ..|.a...^2
0x00000110| 6d540c002d42d50900618601b84e9c66 mT...-B...a...N.f 546d000c422d09d5610001864eb8669c Tm...B...a...N.f.
0x00000120| 1a677c19010000007c192a0020002d0c .g|.....|. * . - 671a197c00010000197c002a00200c2d g...|.....|. * . -
0x00000130| 0400dc0906677c192b00200000618a01 .....g|.+. .a. 000409dc6706197c002b00206100018a .....g|.+. a..
0x00000140| 00702d10dc0940d07b3b24001a003b30 .p-...@.{$;...;0 7000102d09dc0d403b7b0024001a303b p-...@.{$;...;0
0x00000150| 0600f94ef00d000019001a001b001c00 .....N..... 00064ef80df00000019001a001b001c .....N.....
```

■ the differences are easily visible : swap each word (2 bytes)

Mame CPS1 driver source code



```
/* B-Board 91635B-2 */
ROM_START( sf2hfu )
    ROM_REGION( CODE_SIZE, "maincpu", 0 )      /* 68000 code */
    ROM_LOAD16_WORD_SWAP( "s2tu_23.8f", 0x000000, 0x80000, CRC(89a1fc38) SHA1(aafb40fc311e318250973be8c6aa0d3f7902cb3c) )
    ROM_LOAD16_WORD_SWAP( "s2tu_22.7f", 0x080000, 0x80000, CRC(aea6e035) SHA1(ce5fe961b2c1c95d231d1235bfc03b47de489f2a) )
    ROM_LOAD16_WORD_SWAP( "s2tu_21.6f", 0x100000, 0x80000, CRC(fd200288) SHA1(3817b67ab77c7b3d4a573a63f18671bea6905e26) )
```

FIGURE 1 – <https://github.com/fesh0r/old-mame/blob/master/src/mame/drivers/cps1.c#L9618>

Audio samples (oki files)



```
gna@DESKTOP-RRMLEB4: /mnt/c/Users/jeanvaljean/Documents/test/sf2hfu$ cat s92_18.11c s92_19.12c >> rom.oki
gna@DESKTOP-RRMLEB4: /mnt/c/Users/jeanvaljean/Documents/test/sf2hfu$ sha1sum rom.oki StreetFighterII_HF.oki
995526183ffd35f92e9096500a3fe6237faaa2dd  rom.oki
995526183ffd35f92e9096500a3fe6237faaa2dd  StreetFighterII_HF.oki
```

- Just a concatenation of the oki files (for the order : check name CPS1 driver source code)

Audio CPU (z80 file)



```
gna@DESKTOP-RRMLEB4:/mnt/c/Users/jeanvaljean/Documents/test/sf2hfu$ sha1sum StreetFighterII_HF.z80 s92_09.11a
8258fcaca4ac419312531eec67079b97f471179c StreetFighterII_HF.z80
8258fcaca4ac419312531eec67079b97f471179c s92_09.11a
```

■ Identical



VROM ?

- The VROM is a ROM chip inside the game board, it contains :
 - pixel patterns, the colors and the metadata for assembling the tiles into the background and sprites

Conversion

- Convert gfx from Mame to Moo :
 - merge each files into one and reorder bytes
 - decode gfx data

GFX files [1/2]



```
sf2hfu_gfx_before_decoding.mem  s92-1m.3a
Offset (h) 00 01 02 03 04 05 06 07
00000130 FF FF FF FF FF FF FF
00000138 FF FF FF FF FF FF FF
00000140 FF FF FF FF FF FF FF
00000148 FF FF FF FF FF FF FF
00000150 FF FF FF FF FF FF FF
00000158 FF FF FF FF FF FF FF
00000160 03 63 63 03 FF FF FF FF
00000168 40 B0 F0 00 BF FF FF 1F
00000170 83 43 C3 00 D7 DF DF 0F
00000178 07 0F 0F 00 BB CF FF 07
```

```
s92-1m.3a s92-3m.5a s92-2m.4a s92-1m.3a s92-3m.5a s92-2m.4a
Offset (h) 00 01 02 03 04 05 06 07
00000000 FF FF FF FF FF FF FF
00000008 FF FF FF FF FF FF FF
00000010 FF FF FF FF FF FF FF
00000018 FF FF FF FF FF FF FF
00000020 FF FF FF FF FF FF FF
00000028 FF FF FF FF FF FF FF
00000030 FF FF FF FF FF FF FF
00000038 FF FF FF FF FF FF FF
00000040 FF FF FF FF FF FF FF
00000048 FF FF FF FF FF FF FF
00000050 FF FF FF FF FF FF FF
00000058 03 63 40 B0 83 43 07 0F
```

```
s92-1m.3a s92-3m.5a s92-2m.4a s92-1m.3a s92-3m.5a s92-2m.4a
Offset (h) 00 01 02 03 04 05 06 07
00000000 FF FF FF FF FF FF FF
00000008 FF FF FF FF FF FF FF
00000010 FF FF FF FF FF FF FF
00000018 FF FF FF FF FF FF FF
00000020 FF FF FF FF FF FF FF
00000028 FF FF FF FF FF FF FF
00000030 FF FF FF FF FF FF FF
00000038 FF FF FF FF FF FF FF
00000040 FF FF FF FF FF FF FF
00000048 FF FF FF FF FF FF FF
00000050 FF FF FF FF FF FF FF
00000058 43 03 F0 00 C3 00 0F 07
```

```
s92-1m.3a s92-3m.5a s92-2m.4a s92-1m.3a s92-3m.5a s92-4m.6a
Offset (h) 00 01 02 03 04 05 06 07
00000000 FF FF FF FF FF FF FF
00000008 FF FF FF FF FF FF FF
00000010 FF FF FF FF FF FF FF
00000018 FF FF FF FF FF FF FF
00000020 FF FF FF FF FF FF FF
00000028 FF FF FF FF FF FF FF
00000030 FF FF FF FF FF FF FF
00000038 FF FF FC 70 E3 EC DF C0
00000040 FF FF FF FF FF FF FF
00000048 FF FF FF FF FF FF FF
00000050 FF FF FF FF FF FF FF
00000058 FF FF 0F FF D7 DF 0F 07
```

```
s92-1m.3a s92-3m.5a s92-2m.4a s92-1m.3a s92-3m.5a s92-4m.6a
Offset (h) 00 01 02 03 04 05 06 07
00000000 FF FF FF FF FF FF FF
00000008 FF FF FF FF FF FF FF
00000010 FF FF FF FF FF FF FF
00000018 FF FF FF FF FF FF FF
00000020 FF FF FF FF FF FF FF
00000028 FF FF FF FF FF FF FF
00000030 FF FF FF FF FF FF FF
00000038 FF FF FD 7C EF 50 DF C0
00000040 FF FF FF FF FF FF FF
00000048 FF FF FF FF FF FF FF
00000050 FF FF FF FF FF FF FF
00000058 FF FF 1F FF D7 DF 0F 07
```

GFX files [2/2]



```
void cps_state::cps1_gfx_decode()
{
    int size = memregion("gfx")->bytes();
    int i, j, gfxsize;
    UINT8 *cps1_gfx = memregion("gfx")->base();

    gfxsize = size / 4;

    for (i = 0; i < gfxsize; i++)
    {
        UINT32 src = cps1_gfx[4 * i] + (cps1_gfx[4 * i + 1] << 8) + (cps1_gfx[4 * i + 2] << 16) + (cps1_gfx[4 * i + 3] << 24);
        UINT32 dswal = 0;

        for (j = 0; j < 8; j++)
        {
            int n = 0;
            UINT32 mask = (0xB0000000 >> j) & src;

            if (mask & 0x000000ff) n |= 1;
            if (mask & 0x0000ff00) n |= 2;
            if (mask & 0x00ff0000) n |= 4;
            if (mask & 0xff000000) n |= 8;

            dswal |= n << (j * 4);
        }
        cps1_gfx[4 * i] = dswal >> 0;
        cps1_gfx[4 * i + 1] = dswal >> 8;
        cps1_gfx[4 * i + 2] = dswal >> 16;
        cps1_gfx[4 * i + 3] = dswal >> 24;
    }
}
```

FIGURE 2 – <https://github.com/fesh0r/old-mame/blob/master/src/mame/video/cps1.c#L1720>

Mame to Moo conversion



- The Mame driver source code must be parsed
 - to know what are the audio / gfx / 68k files
 - to get the correct order when concatenating the oki files
 - to know how to reorder the 68k files

Mame to Moo conversion



```
gna@DESKTOP-RRMLEB4:/mnt/c/Users/jeanvaljean/Documents/mame2moo$ python mame2moo.py sf2hfu.zip cps1
Converting maincpu...
[+] maincpu converted
Converting gfx...
[+] gfx converted
Converting audio...
[+] audio converted
Converting audio samples...
[+] audio samples extracted
gna@DESKTOP-RRMLEB4:/mnt/c/Users/jeanvaljean/Documents/mame2moo$ sha1sum rom.* StreetFighterII_HF.* | sort
1296e423b7de50ff451b4abd5c262cced57e6cc8  rom.vrom
3175444eee225872338d6389c7604511aa81e001  StreetFighterII_HF.u.vrom
7afa980c2bd81993a177c5589498f0f8c889e719  StreetFighterII_HF.u.68k
7afa980c2bd81993a177c5589498f0f8c889e719  rom.68k
8258fcaca4ac419312531eec67079b97f471179c  StreetFighterII_HF.z80
8258fcaca4ac419312531eec67079b97f471179c  rom.z80
995526183ffd35f92e9096500a3fe6237faaa2dd  StreetFighterII_HF.oki
995526183ffd35f92e9096500a3fe6237faaa2dd  rom.oki
```

FIGURE 3 –

<https://github.com/angelkillah/MooHijack/blob/master/script/mame2moo.py>

original GFX patched in sf30th



Hijack Moo roms loading



Now that we can convert any CPS1 roms from Mame to Moo, we need to force the game to load our freshly converted roms.

Steps

- Locate the assets loading function
- Hijack the execution flow

Moo assets loading



```
;- fcn.1401481b0:
(fcn) BundleFileSystem::Set_MFData_With_Data 295
BundleFileSystem::Set_MFData_With_Data (int32_t var_80h);
; var int32_t var_20h @ rsp+0x20
; var int32_t var_38h @ rsp+0x38
; var int32_t var_60h @ rsp+0x60
; arg int32_t var_80h @ rsp+0x80
mov rax, rsp
push rdi
push r14
push r15
sub rsp, 0x60 ; ""
mov qword [rax - 0x48], 0xfffffffffffffffe
mov qword [rax + 0x10], rbx
mov qword [rax + 0x18], rbp
mov qword [rax + 0x20], rsi
mov rbx, rdx
mov rsi, rcx
call w_init_MFFileSystem_Standard
mov rbp, rax
xor edi, edi
mov r9b, 1
mov r8, rbx
lea rdx, [var_38h] ; MFS_stripped_path
; 0x38
mov rcx, rsi
call Get_MFBundle_Name
nop
lea rcx, [rsi + 0x40] ; 'e'; 64
lea rdx, [var_38h] ; MFS_stripped_path
; 0x38
call fcn.1400764d0
mov rbx, rax
test rax, rax
je 0x1401482b0
```

```
lea rdx, [rsi + 0x18] ; 24
xor r8d, r8d
mov rcx, rbp
call LoadFile
mov rsi, rax
test rax, rax
je 0x1401482b0
```

Moo assets loading



```
lea rdx, str.size ; 0x14026c684 ; "size"
mov rcx, rbx
call GetField
mov r14d, eax
lea rdx, str.offset ; 0x14026c68c ; "offset"
mov rcx, rbx
call GetField
mov ebx, eax
mov ecx, r14d
call qword sym.imp.api_ms_win_crt_heap_l1_1_0.dll_malloc ; 0x14023c5d0 ; "$\x114" ; void *malloc...
mov r15, rax
mov r8d, ebx
mov rdx, rsi
mov rcx, rbp
call MBundle_fseek
mov r9d, r14d ; size
mov r8, r15 ; data
mov rdx, rsi ; MFString
mov rcx, rbp ; MFileSystem
call GetData
mov rdx, rsi
mov rcx, rbp
call fcn.140081180
lea ecx, [rdi + 0x28] ; '(' ; 40 ; size
call fcn.new
mov qword [var_80h], rax
test rax, rax
je 0x1401482b0
```

Hooking



- To replace the loading of whatever resource, we hijack the execution flow at two different locations :
 - the result of the first call to `GetField()` : to replace the original resource size
 - the buffer filled by `GetData()` : to replace the resource data



VEH Hooking

- We modify the first byte of the instruction to hijack by an opcode that will cause an exception
- We install a vectored exception handler to catch it
- cons : no need to calculate the instructions size

Hijack assets loading



```
je sf30thanniversarycollection.7FF7CAF582B0
lea rdx,qword ptr ds:[7FF7CB07C684]
mov rcx,rbx
call <sf30thanniversarycollection.GetField>
int3
mov esi,eax
lea rdx,qword ptr ds:[7FF7CB07C68C]
mov rcx,rbx
call <sf30thanniversarycollection.GetField>
mov ebx,eax
mov ecx,r14d
call qword ptr ds:[<&malloc>]
mov r15,rax
mov r8d,ebx
mov rdx,rsi
mov rcx,rbp
call sf30thanniversarycollection.7FF7CAE91E00
mov r9d,r14d
mov r8,r15
mov rdx,rsi
mov rcx,rbp
call <sf30thanniversarycollection.GetData>
mov rdx,rsi
int3
mov ecx,ebp
```

00007FF7C807C684:"size"

00007FF7C807C68C:"offset"

VEH Handler

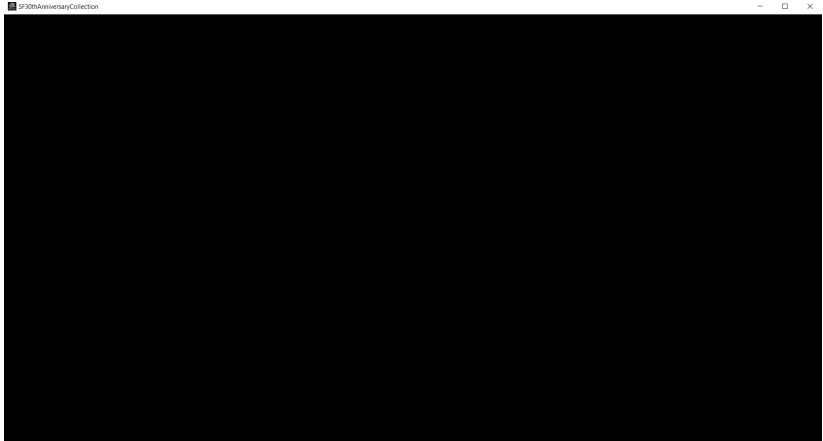
Set rax to the size
of the resource
we want to load

R10-size always
points to the
resource data, we
patch the resource
content with our
resource

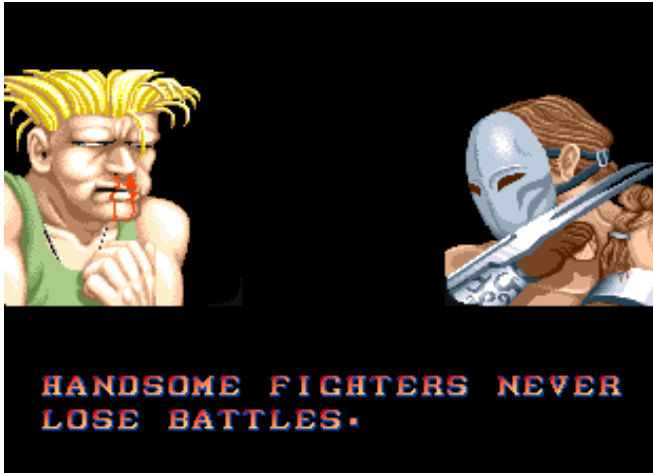
Results...



Results...



Game over ?



Load Rom function



```
(fcn) Game_StreetFighterII_HF_LoadROM 127
Game_StreetFighterIIF_LoadROM ()
; var int32_t var_20h @ rsp+0x20
; var int32_t var_20h @ rsp+0x20
; var int32_t var_50h @ rsp+0x50
push rfb
sub rsp, 0x60 ; ""
lea rdx, [SP+CP50] ; 0x14010d470
mov rfb, rcx
call Setup_CPS1_With_ROM_Info
test al, al
jne 0x14010d45f

add rsp, 0x60 ; ""
pop rfb
ret

mov rax, qword [rbx]
mov rcx, rfb
mov qword [0x14034b378], 0x20 ; "(
call qword [rax + 0x60] ; 128
mov rcx, rfb
call fcn.1401cf8e0
lea rax, [0x14020e020]
mov qword [var_20h], rax
mov qword [var_20h], rax
lea rdx, [rbx + 0x7400]
lea rax, [var_20h]
lea rcx, [rbx + 0x7500]
mov qword [var_50h], rax
lea r9, [var_20h]
lea r8, [0x1402d3660]
call Patch_CFX_Menu
mov al, 1
add rsp, 0x60 ; ""
pop rfb
ret
```

```
(fcn) Game_StreetFighterII_LoadROM 117
Game_StreetFighterII_LoadROM ()
; var int32_t var_20h @ rsp+0x20
; var int32_t var_20h @ rsp+0x20
; var int32_t var_50h @ rsp+0x50
push rfb
sub rsp, 0x60 ; ""
lea rdx, [SP+CP50] ; 0x14020c340
mov rfb, rcx
call Setup_CPS1_With_ROM_Info
test al, al
jne 0x14010d72f

add rsp, 0x60 ; ""
pop rfb
ret

mov rax, qword [rbx]
mov rcx, rfb
call qword [rax + 0x60] ; 128
mov rcx, rfb
call fcn.1401cf8e0
lea rax, [0x14020e020]
mov qword [var_20h], rax
mov qword [var_20h], rax
lea rdx, [rbx + 0x7400]
lea rax, [var_20h]
lea rcx, [rbx + 0x7500]
mov qword [var_50h], rax
lea r9, [var_20h]
lea r8, [0x1402d3660]
call Patch_CFX_Menu
mov al, 1
add rsp, 0x60 ; ""
pop rfb
ret
```

- Almost no differences between both functions
- Setup_CPS1_With_ROM_info() takes two arguments :
 - the object of the chosen game (this)
 - an address to a structure...

SF2CE_Config



```
.qword 0x00000001402de240 ; str.Street_Fighter_II:_Champion_Edition
.qword 0x00000001402de268 ; str.920313_USA
.qword 0x00000001402de278 ; str.StreetFighterII_CE
.qword 0x00000001402de290 ; str.Capcom_StreetFighterII_CE
.qword 0x00000001402bc3e8
.qword 0x00000001402de2b0 ; str.StreetFighterII_CE.ua.68k
.qword 0x00000001402de2d0 ; str.StreetFighterII_CE.vrom
.qword 0x0000000000000000
.qword 0x00000001402ddf30 ; str.eagle_logo.vrom
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x00000001402bd090
.qword 0x00000001402de290 ; str.Capcom_StreetFighterII_CE
.qword 0x00000001402de2e8 ; str.StreetFighterII_CE.z80
.qword 0x00000001402de300 ; str.StreetFighterII_CE.oki
```

1402bc3e8 => Dipswitches



```
0x1402bc3e8 .qword 0x0000001402bc00 : Difficulty
0x1402bc3f0 .qword 0x0000001402bc9a0 : Free Play
0x1402bc3f8 .qword 0x0000001402bc840 : Demo Sound
0x1402bc400 .qword 0x0000001402bc6e0 : Continue
0x1402bc408 .qword 0x0000001402bc580 : Game Mode
0x1402bc410 .qword 0x0000000000000000
0x1402bc418 .qword 0x0000000000000000
0x1402bc420 .qword 0x0000001402dde50 : str.Two_Players_Game
0x1402bc428 .qword 0x0000001402dde68 : str.1_Credit_No_Continue
0x1402bc430 .qword 0x0000000000000008
0x1402bc438 .qword 0x0000001402dde80 : str.2_Credits_Winner_Continue
0x1402bc440 .qword 0x0000000000000000
0x1402bc448 .qword 0x0000000000000000
0x1402bc450 .qword 0x0000000000000000
```

Test Mode



1402bd090 => ??



```
.qword 0x0000000600000002
.dword 0x00b71b00
.dword 0x0000000b
.dword 0x00000032
.dword 0x00000001
.dword 0x00000026
.dword 0x00000006
.dword 0x00000030
.dword 0x00000002
.dword 0x00000028
.dword 0x00000003
.dword 0x0000002a
.dword 0x00000004
.dword 0x0000002c
.dword 0x00000005
.dword 0x0000002e
.dword 0x00000007
.dword 0x00000000
.dword 0x00000008
.dword 0x00000002
.dword 0x00000009
.dword 0x00000004
.dword 0x0000000a
.dword 0x00000006
.dword 0x0000000c
.dword 0x00000036
.dword 0x0000000f
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0xffffffff
.dword 0x00000002
.dword 0x00000004
.dword 0x00000008
.dword 0x00000030
.dword 0x00000030
.dword 0x00000000
.qword 0x00000001402c81f0
```



CPS Board



- A set of these data are copied to the “StreetFighterII_CE” object attributes
- The first 4-bytes data value (0xb71b00) is used in a method of the class “Moo_Sys_CPS1”.
- This method is used to execute the 68000 code ROM through an emulator.



- A set of these data are copied to the “StreetFighterII_CE” object attributes
- The first 4-bytes data value (0xb71b00) is used in a method of the class “Moo_Sys_CPS1”.
- This method is used to execute the 68000 code ROM through an emulator.

Clock frequency

- $0xb71b00 == 12000000 = 12\text{Mhz}$
- The processor Motorola 68000 used for SF2CE runs at 12Mhz

CPS-B Registers



- The original arcade board of CPS1 games contains several registers :
 - priority mask : used to set the tiles priority levels
 - palette control register : indicates which palette pages to copy from gfxram to dedicated ram
 - test register : used for self test checks
 - etc.

CPS-B Registers



```
/* CPS ID multiply protection unknown ctrl priority masks palctrl layer enable masks */
#define CPS_B_01 -1, 0x0000, not applicable, 0x26, {0x28, 0x2a, 0x2c, 0x2e}, 0x30, {0x02, 0x04, 0x08, 0x30, 0x30}
#define CPS_B_02 0x20, 0x0002, not applicable, 0x2c, {0x2a, 0x28, 0x26, 0x24}, 0x22, {0x02, 0x04, 0x08, 0x00, 0x00}
#define CPS_B_03 -1, 0x0000, not applicable, 0x30, {0x2e, 0x2c, 0x2a, 0x28}, 0x26, {0x20, 0x10, 0x08, 0x00, 0x00}
#define CPS_B_04 0x20, 0x0004, not applicable, 0x2e, {0x26, 0x30, 0x28, 0x32}, 0x2a, {0x02, 0x04, 0x08, 0x00, 0x00}
#define CPS_B_05 0x20, 0x0005, not applicable, 0x28, {0x2a, 0x2c, 0x2e, 0x30}, 0x22, {0x02, 0x08, 0x20, 0x14, 0x14}
#define CPS_B_11 0x32, 0x0401, not applicable, 0x26, {0x28, 0x2a, 0x2c, 0x2e}, 0x32, {0x08, 0x16, 0x28, 0x00, 0x00}
#define CPS_B_12 0x20, 0x0402, not applicable, 0x2c, {0x28, 0x28, 0x26, 0x24}, 0x22, {0x02, 0x04, 0x08, 0x00, 0x00}
#define CPS_B_13 0x2e, 0x0403, not applicable, 0x22, {0x24, 0x26, 0x28, 0x2a}, 0x2c, {0x20, 0x02, 0x04, 0x00, 0x00}
#define CPS_B_14 0x1e, 0x0404, not applicable, 0x12, {0x14, 0x16, 0x18, 0x1a}, 0x1c, {0x08, 0x20, 0x16, 0x00, 0x00}
#define CPS_B_15 0x0e, 0x0405, not applicable, 0x02, {0x04, 0x06, 0x08, 0x0a}, 0x0c, {0x04, 0x02, 0x20, 0x00, 0x00}
#define CPS_B_16 0x00, 0x0406, not applicable, 0x0c, {0x0a, 0x08, 0x06, 0x04}, 0x02, {0x10, 0x0a, 0x0a, 0x00, 0x00}
#define CPS_B_17 0x08, 0x0407, not applicable, 0x14, {0x12, 0x10, 0x0e, 0x0c}, 0x0a, {0x08, 0x14, 0x02, 0x00, 0x00}
conversion needs 0x04 for the 2nd layer enable on one level, gtx confirmed to appear on the PCB, register at the time is 0x0e, so 0
#define CPS_B_18 0x10, 0x0408, not applicable, 0x1c, {0x1a, 0x18, 0x16, 0x14}, 0x12, {0x10, 0x08, 0x02, 0x00, 0x00}
#define CPS_B_21_DEF 0x32, -1, 0x00, 0x02, 0x04, 0x06, 0x08, -1, -1, 0x26, {0x28, 0x2a, 0x2c, 0x2e}, 0x30, {0x02, 0x04, 0x08, 0x30, 0x30}
to 0x26 on startup
#define CPS_B_21_BT1 0x32, 0x0800, 0x0e, 0x0c, 0x0a, 0x08, 0x06, 0x04, 0x02, 0x28, {0x26, 0x24, 0x22, 0x20}, 0x30, {0x20, 0x04, 0x08, 0x12, 0x12}
#define CPS_B_21_BT2 -1, -1, 0x1e, 0x1c, 0x1a, 0x18, -1, 0x0c, 0x0a, 0x20, {0x2e, 0x2c, 0x2a, 0x28}, 0x30, {0x30, 0x08, 0x30, 0x00, 0x00}
#define CPS_B_21_BT3 -1, -1, 0x06, 0x04, 0x02, 0x00, 0x0e, 0x0c, 0x0a, 0x20, {0x2e, 0x2c, 0x2a, 0x28}, 0x30, {0x20, 0x12, 0x12, 0x00, 0x00}
#define CPS_B_21_BT4 -1, -1, 0x06, 0x04, 0x02, 0x00, 0x1e, 0x1c, 0x1a, 0x28, {0x26, 0x24, 0x22, 0x20}, 0x30, {0x20, 0x10, 0x02, 0x00, 0x00}
#define CPS_B_21_BT5 0x32, -1, 0x0e, 0x0c, 0x0a, 0x08, 0x1e, 0x1c, 0x1a, 0x20, {0x2e, 0x2c, 0x2a, 0x28}, 0x30, {0x20, 0x04, 0x02, 0x00, 0x00}
#define CPS_B_21_BT6 -1, -1, -1, -1, -1, -1, -1, -1, -1, 0x20, {0x2e, 0x2c, 0x2a, 0x28}, 0x30, {0x20, 0x14, 0x14, 0x00, 0x00}
#define CPS_B_21_BT7 -1, -1, -1, -1, -1, -1, -1, -1, -1, 0x2c, { -1, -1, -1, -1 }, 0x12, {0x14, 0x02, 0x14, 0x00, 0x00}
#define CPS_B_21_QS1 -1, -1, -1, -1, -1, -1, -1, -1, -1, 0x22, {0x24, 0x26, 0x28, 0x2a}, 0x2c, {0x10, 0x08, 0x04, 0x00, 0x00}
#define CPS_B_21_QS2 -1, -1, -1, -1, -1, -1, -1, -1, -1, 0x0a, {0x0c, 0x0e, 0x00, 0x02}, 0x04, {0x16, 0x16, 0x16, 0x00, 0x00}
#define CPS_B_21_QS3 0x0e, 0x0c00, -1, -1, -1, -1, 0x2c, -1, -1, 0x12, {0x14, 0x16, 0x08, 0x0a}, 0x0c, {0x04, 0x02, 0x20, 0x00, 0x00}
#define CPS_B_21_QS4 0x2e, 0x0c01, -1, -1, -1, -1, 0x1c, 0x1e, 0x08, 0x16, {0x00, 0x02, 0x28, 0x2a}, 0x2c, {0x04, 0x08, 0x10, 0x00, 0x00}
#define CPS_B_21_QS5 0x1e, 0x0c02, -1, -1, -1, -1, 0x0c, -1, -1, 0x2a, {0x2c, 0x2e, 0x30, 0x32}, 0x1c, {0x04, 0x08, 0x16, 0x00, 0x00}
```

FIGURE 4 –

<https://github.com/mamedev/mame/blob/master/src/mame/video/cps1.cpp>

- Luckily, the values to be set in CPS-B registers for each game are listed in mame cps1 video source code

CPS-B Registers for SF2CE



```
{"sf2ceua",      CPS_B_21_DEF, mapper_S9263B, 0x36 }
```

FIGURE 5 – name, CPSB, gfx mapper, in2

CPSB-21-DEF



```
/* CPSB ID multiply protection unknown ctrl priority masks palctrl layer enable masks */
#define CPS_B 01 -1, 0x0000, not_applicable, 0x26, {0x28,0x2a,0x2c,0x2e},0x30, {0x02,0x04,0x08,0x30,0x30}
#define CPS_B 02 0x20,0x0002, not_applicable, 0x2c, {0x2a,0x28,0x26,0x24},0x22, {0x02,0x04,0x08,0x00,0x00}
#define CPS_B 03 -1, 0x0000, not_applicable, 0x30, {0x2e,0x2c,0x2a,0x28},0x26, {0x20,0x10,0x08,0x00,0x00}
#define CPS_B 04 0x20,0x0004, not_applicable, 0x2e, {0x26,0x30,0x28,0x32},0x2a, {0x02,0x04,0x08,0x00,0x00}
#define CPS_B 05 0x20,0x0005, not_applicable, 0x28, {0x2a,0x2c,0x2e,0x30},0x32, {0x02,0x08,0x20,0x14,0x14}
#define CPS_B 11 0x32,0x0401, not_applicable, 0x26, {0x28,0x2a,0x2c,0x2e},0x30, {0x08,0x10,0x20,0x00,0x00}
#define CPS_B 12 0x20,0x0402, not_applicable, 0x2c, {0x2a,0x28,0x26,0x24},0x22, {0x02,0x04,0x08,0x00,0x00}
#define CPS_B 13 0x2e,0x0403, not_applicable, 0x22, {0x24,0x26,0x28,0x2a},0x2c, {0x20,0x02,0x04,0x00,0x00}
#define CPS_B 14 0x1e,0x0404, not_applicable, 0x12, {0x14,0x16,0x18,0x1a},0x1c, {0x08,0x20,0x10,0x00,0x00}
#define CPS_B 15 0x0e,0x0405, not_applicable, 0x02, {0x04,0x06,0x08,0x0a},0x0c, {0x04,0x02,0x20,0x00,0x00}
#define CPS_B 16 0x00,0x0406, not_applicable, 0x0c, {0x0a,0x08,0x06,0x04},0x02, {0x10,0x0a,0x0a,0x00,0x00}
#define CPS_B 17 0x08,0x0407, not_applicable, 0x14, {0x12,0x10,0x0e,0x0c},0x0a, {0x08,0x14,0x02,0x00,0x00}
conversion needs 0x04 for the 2nd layer enable on one level, gfx confirmed to appear on the PCB, register at the time is 0x8e, so 0
#define CPS_B 18 0x10,0x0408, not applicable, 0x1c, {0x1a,0x18,0x16,0x14},0x12, {0x10,0x08,0x02,0x00,0x00}
#define CPS_B 21 DEF 0x32, -1, 0x00,0x02,0x04,0x06, 0x08, -1, -1, 0x26, {0x28,0x2a,0x2c,0x2e},0x30, {0x02,0x04,0x08,0x30,0x30}
```

Makes more sense !



```
.qword 0x0000000000000000
.dword 0x00b71b00 ; 12Mhz
.dword 0x0000000b
.dword 0x00000032 ; CPSB_ID
.dword 0x00000001
.dword 0x00000026 ; ctrl
.dword 0x00000006
.dword 0x00000030 ; palctrl
.dword 0x00000002
.dword 0x00000028 ; priority_mask[0]
.dword 0x00000003
.dword 0x0000002a ; priority_mask[1]
.dword 0x00000004
.dword 0x0000002c ; priority_mask[2]
.dword 0x00000005
.dword 0x0000002e ; priority_mask[3]
.dword 0x00000007
.dword 0x00000000 ; multiply_protection[0]
.dword 0x00000008
.dword 0x00000002 ; multiply_protection[1]
.dword 0x00000009
.dword 0x00000004 ; multiply_protection[2]
.dword 0x0000000a
.dword 0x00000006 ; multiply_protection[3]
.dword 0x0000000c
.dword 0x00000036 ; in2
.dword 0x0000000f ; end marker
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0xffffffff ; ID (-1)
.dword 0x00000002 ; layer_enable_mask[0]
.dword 0x00000004 ; layer_enable_mask[1]
.dword 0x00000008 ; layer_enable_mask[2]
.dword 0x00000030 ; layer_enable_mask[3]
.dword 0x00000030 ; layer_enable_mask[4]
.dword 0x00000000
.qword 0x00000001402c81f0 ; gfx_mapper
```



GFX Mapper



```
.dword 0x00000000
.dword 0x00007fff
.dword 0x00000000
.dword 0x00007fff
.dword 0x00000000
.dword 0x0000ffff
.dword 0x00000000
.dword 0x00007fff
.dword 0x00010000
.dword 0x0001ffff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffffff
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0x00004000
.dword 0x00004fff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffffff
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0x00005000
.dword 0x00007fff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffffff
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0x00002000
.dword 0x00003fff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffffff
```



GFX Mapper



```
#define mapper_S9263B { 0x8000, 0x8000, 0x8000, 0 }, mapper_S9263B_table
static const struct gfx_range mapper_S9263B_table[] =
{
    // verified from PAL dump:
    // FIXME there is some problem with this dump since pin 14 is never enabled
    // instead of being the same as pin 15 as expected
    // bank0 = pin 19 (ROMs 1,3) & pin 18 (ROMs 2,4)
    // bank1 = pin 17 (ROMs 5,7) & pin 16 (ROMs 6,8)
    // bank2 = pin 15 (ROMs 10,12) & pin 14 (ROMs 11,13)
    // pins 12 and 13 are the same as 14 and 15

    /* type          start   end     bank */
    { GFXTYPE_SPRITES, 0x00000, 0x07fff, 0 },

    { GFXTYPE_SPRITES, 0x08000, 0x0ffff, 1 },

    { GFXTYPE_SPRITES, 0x10000, 0x11fff, 2 },
    { GFXTYPE_SCROLL3, 0x02000, 0x03fff, 2 },
    { GFXTYPE_SCROLL1, 0x04000, 0x04fff, 2 },
    { GFXTYPE_SCROLL2, 0x05000, 0x07fff, 2 },
    { 0 }
};
```

FIGURE 6 –

<https://github.com/mamedev/mame/blob/master/src/mame/video/cps1.cpp>

GFX Mapper



```
.dword 0x00000000
.dword 0x00007fff
.dword 0x00000000
.dword 0x00007fff
.dword 0x00008000
.dword 0x0000ffff
.dword 0x00008000
.dword 0x00007fff
.dword 0x00010000
.dword 0x0001ffff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffff
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0x00004000
.dword 0x00004fff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffff
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0x00005000
.dword 0x00007fff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffff
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.qword 0x0000000000000000
.dword 0x00000000
.dword 0x00002000
.dword 0x00003fff
.dword 0x00010000
.dword 0x00007fff
.dword 0xffffffff
```

```
/* type      start  end    bank */
{ GFXTYPE_SPRITES, 0x00000, 0x07fff, 0 },

{ GFXTYPE_SPRITES, 0x00000, 0x0ffff, 1 },

{ GFXTYPE_SPRITES, 0x10000, 0x11fff, 2 },
{ GFXTYPE_SCROLL3, 0x02000, 0x03fff, 2 },
{ GFXTYPE_SCROLL1, 0x04000, 0x04fff, 2 },
{ GFXTYPE_SCROLL2, 0x05000, 0x07fff, 2 },
{ 0 }
```

GFX Mapper



```
.dword @x00000000 ; GFXTYPE_SPRITES start
.dword @x00007fff ; GFXTYPE_SPRITES end
.dword @x00000000 ; bank start
.dword @x00007fff ; bank end
.dword @x00008000 ; GFXTYPE_SPRITES start
.dword @x0000ffff ; GFXTYPE_SPRITES end
.dword @x00008000 ; bank start
.dword @x00007fff ; bank end
.dword @x00010000 ; GFXTYPE_SPRITES start
.dword @x00011fff ; GFXTYPE_SPRITES end
.dword @x00010000 ; bank start
.dword @x00007fff ; bank end
.dword @xffffffff ; end marker GFXTYPE_SPRITES
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x00000000
.dword @x00004000 ; GFXTYPE_SCROLL1 start
.dword @x00004fff ; GFXTYPE_SCROLL1 end
.dword @x00010000 ; bank start
.dword @x00007fff ; bank end
.dword @xffffffff ; end marker GFXTYPE_SCROLL1
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.dword @x00000000
.dword @x00005000 ; GFXTYPE_SCROLL2 start
.dword @x00007fff ; GFXTYPE_SCROLL2 end
.dword @x00010000 ; bank start
.dword @x00007fff ; bank end
.dword @xffffffff ; end marker GFXTYPE_SCROLL2
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.qword @x0000000000000000
.dword @x00000000
.dword @x00002000 ; GFXTYPE_SCROLL3 start
.dword @x00003fff ; GFXTYPE_SCROLL3 end
.dword @x00010000 ; bank start
.dword @x00007fff ; bank end
.dword @xffffffff ; end marker GFXTYPE_SCROLL3
```



How to load an additional game ?

- Convert the rom to Moo compatible one
- Hijack the roms loading with the converted ones
- Patch the CPSB data with the ones from the new game
- Patch the GFX mapper

Demo



DEMO



I wish I could play Ghouls'n'Ghost :(

- Some games can be set to freeplay through their dipswitches (no coins needed)
- What about the games that do not have freeplay available ?

How to fix in a “generic” way



Patch emulator game memory

- Get the address of the coins through the cheat engine included in Mame debugger
- Hijack the handler of an opcode that is used to read a word value from the game VRAM to set some coins
- Enjoy moar games (:

Enjoy moar games



FIGURE 7 – before VRAM patching

Enjoy moar games



FIGURE 8 – after VRAM patching



How to load an additional game ?

- Convert the rom to Moo compatible one
- Hijack the roms loading with the converted ones
- Patch the CPSB data with the ones from the new game
- Patch the GFX mapper
- **Either patch dipswitches to set freeplay game mode or patch game VRAM if freeplay not available**

Table of Contents



- 1 Introduction
- 2 From Moo to Arcade
- 3 Play additional games
- 4 Netcode



What's the problem ?

- The online version of ssf2x is not running at the correct speed
- The problem exists since launch day and hasn't been fixed until now

Workflow when running ssf2x online



- Init the following object :
 - **Game_SuperStreetFighterII_Turbo : Moo_Sys_CPS2 : MooBase**
- Parse and retrieve game assets from the filesystem
- **Load save state from assets to avoid desynch**
- Map the GFXs using bank mappers
- Render graphics, run the 68k emulator with the maincpu rom



Save state ?

- Moo supports memory save state (emulator snapshot memory)
- When playing offline mode, it is used to save game progress
- For online mode, it is used for both players to start the game at the same state
- For the four games available to play online, there are four saved state files embedded in the mbundle files

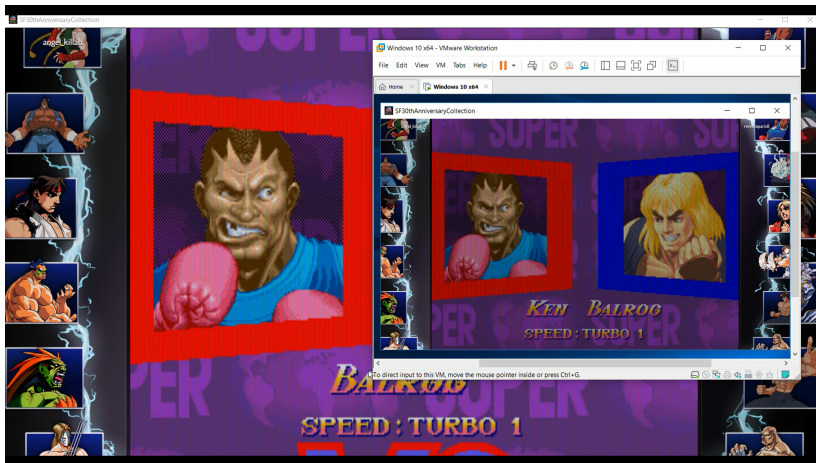
Solution : patch and hijack save state



Steps

- RE the save state format and patch the turbo value with the correct one
- Hijack the save state loading with the patched one
- Enjoy

SSF2X online speed FIX



Play a different game online



Netplay

- When reversing the netplay code to fix the sf2x speed problem, we noticed something interesting . . .
 - the roms are loaded **LOCALLY** for both players !!!!

Enable netplay for MOAR games



How to play additional games online

- Convert the rom to Moo compatible one
- Hijack the roms loading with the converted ones
- Patch the CPSB data with the ones from the new game
- Patch the GFX mapper
- **Play the additional game offline, at the menu, select two players and save a memory snapshot**
- **Take out the new save state from the memory and write it to a file**
- **Hijack the save state loading with the new one**
- Either patch dipswitches to set freeplay game mode or patch game VRAM if freeplay not available

Demo



DEMO

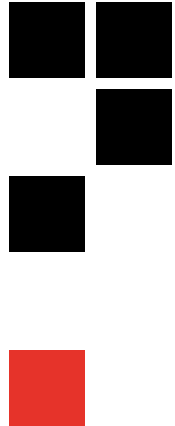
Source code



<https://github.com/angelkillah/MooHijack>



QUESTIONS?



Thank you for your attention

