HARDWARE REVERSE ENGINEERING INSIGHTS FROM THE MAME PROJECT: A PATH TOWARDS FREE FIRMWARE

FELIPE SANCHES
JUCA@MEMBERS.FSF.ORG

PREVIOUS TALKS

FiSL 2014/2015
PORTO ALEGRE - BRAZIL

BACKGROUND

2007: INKSCAPE DEVELOPMENT

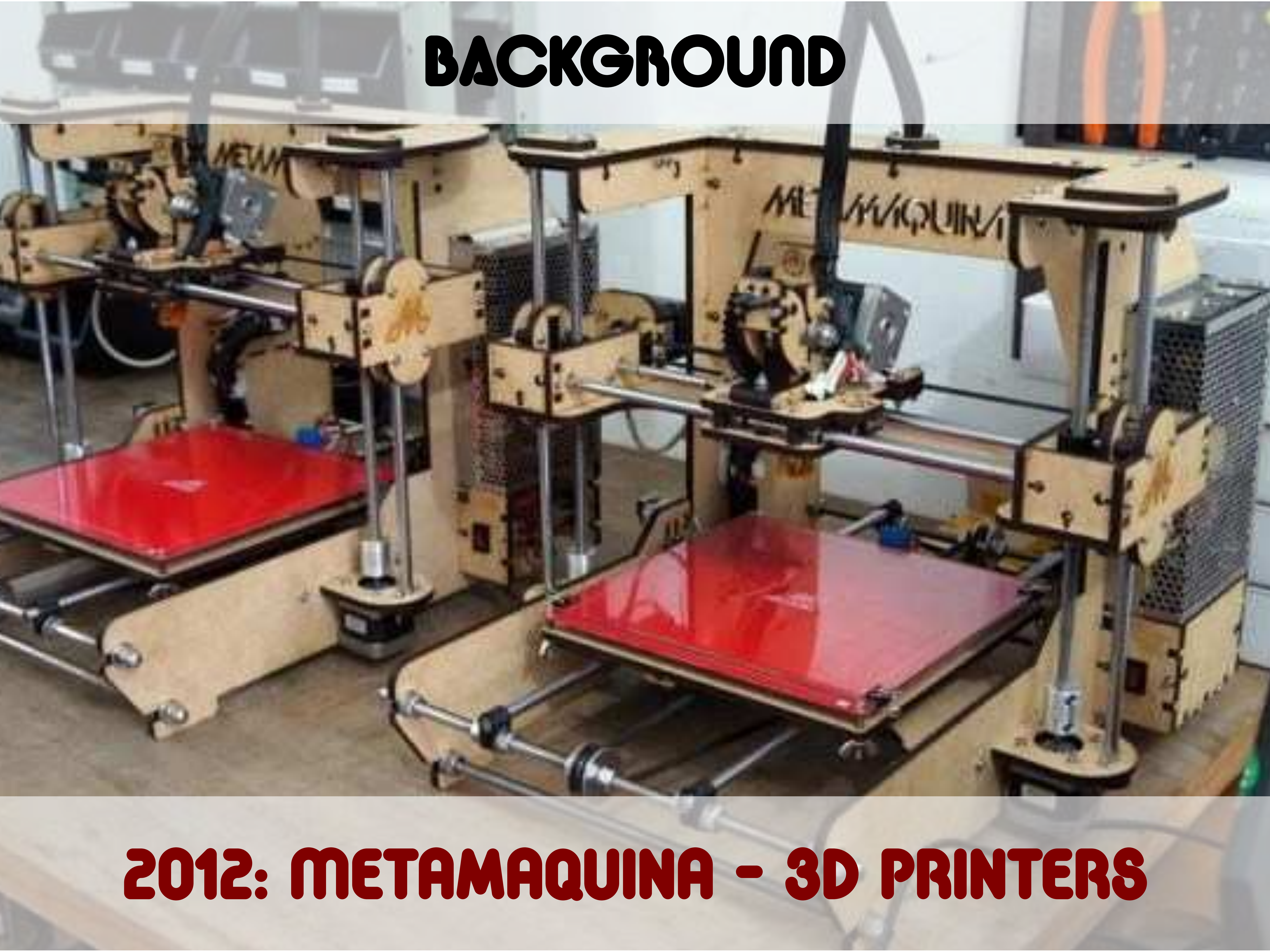
BACKGROUND

2010: GNU LIBREDWG

BACKGROUND

2011: GAROA HACKER CLUBE

BACKGROUND



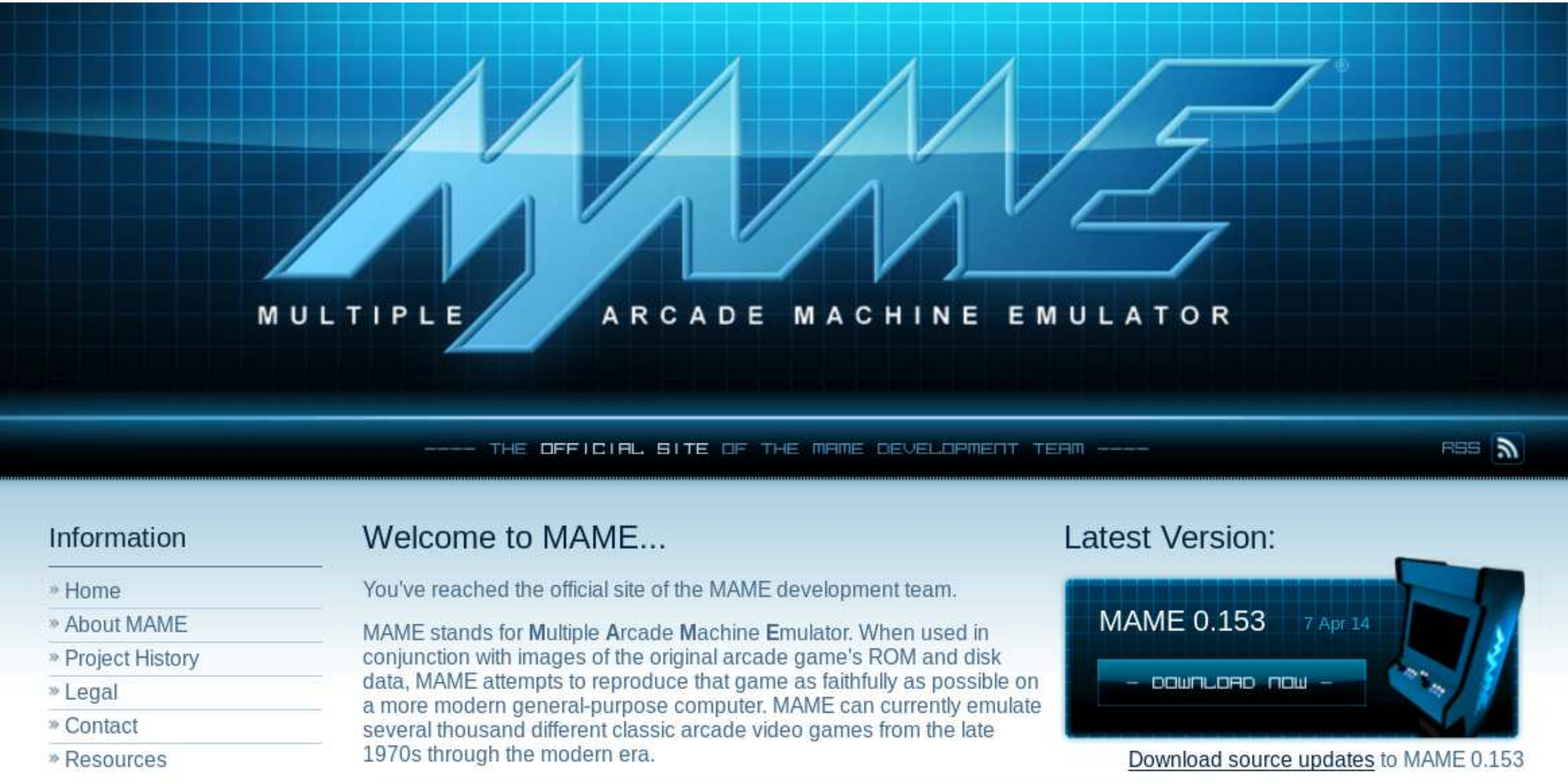
2012: METAMAQUINA - 3D PRINTERS

BACKGROUND

2015: HW DESIGN - MOD-DEVICES

BACKGROUND

2015: TYPOGRAPHY TOOLS



Welcome to MAME...

You've reached the official site of the MAME development team.

MAME stands for **M**ultiple **A**rcade **M**achine **E**mulator. When used in conjunction with images of the original arcade game's ROM and disk data, MAME attempts to reproduce that game as faithfully as possible on a more modern general-purpose computer. MAME can currently emulate several thousand different classic arcade video games from the late 1970s through the modern era.

Download source updates to MAME 0.153

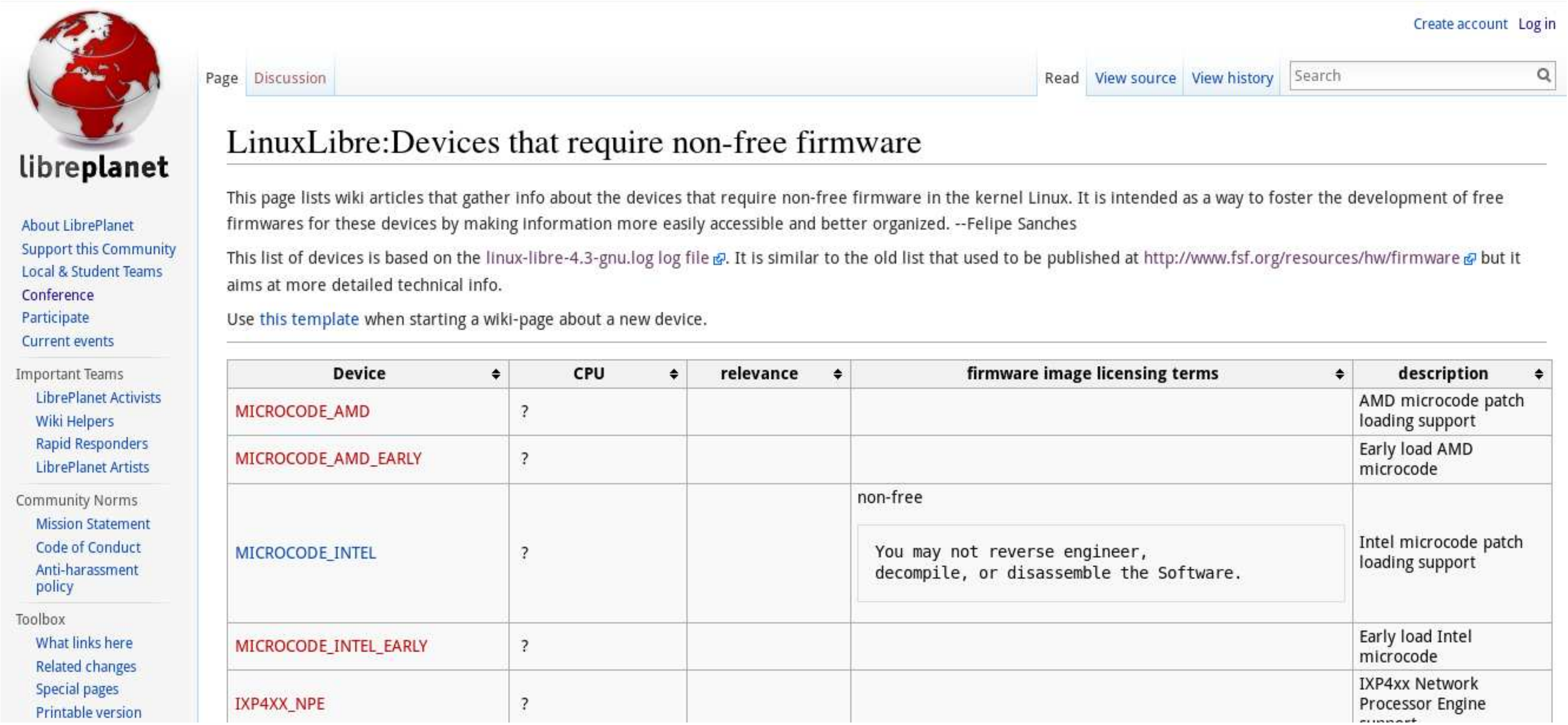
MULTI ARCADE MACHINE EMULATOR

MULTI EMULATOR SUPER SYSTEM

LINUX-LIBRE

FULLY FREE KERNEL

LINUX-LIBRE



LinuxLibre:Devices that require non-free firmware

This page lists wiki articles that gather info about the devices that require non-free firmware in the kernel Linux. It is intended as a way to foster the development of free firmwares for these devices by making information more easily accessible and better organized. --Felipe Sanches

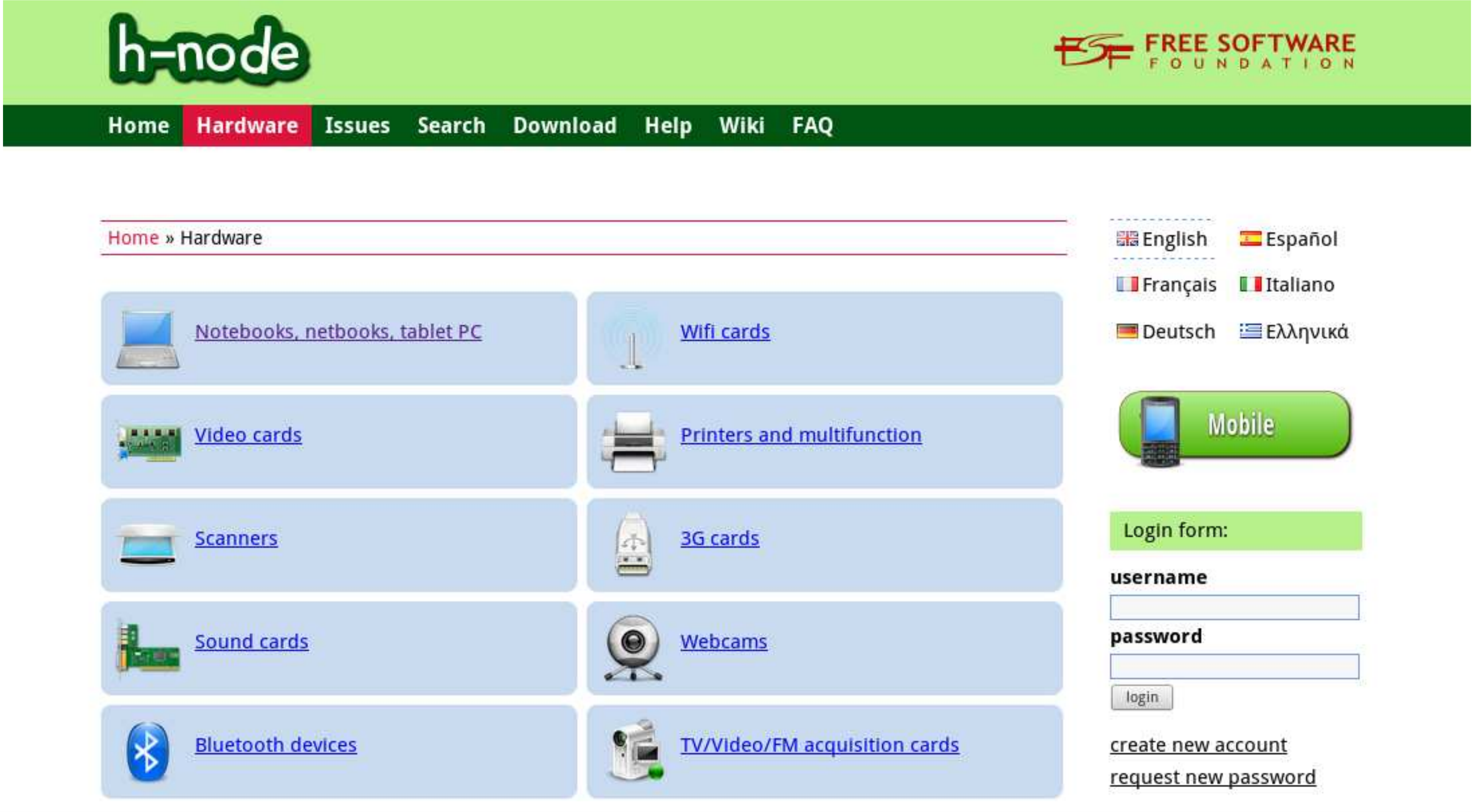
This list of devices is based on the linux-libre-4.3-gnu.log log file. It is similar to the old list that used to be published at http://www.fsf.org/resources/hw/firmware but it aims at more detailed technical info.

Use this template when starting a wiki-page about a new device.

Device	CPU	relevance	firmware image licensing terms	description
MICROCODE_AMD	?			AMD microcode patch loading support
MICROCODE_AMD_EARLY	?			Early load AMD microcode
MICROCODE_INTEL	?		non-free `You may not reverse engineer, decompile, or disassemble the Software.`	Intel microcode patch loading support
MICROCODE_INTEL_EARLY	?			Early load Intel microcode
IXP4XX_NPE	?			IXP4xx Network Processor Engine

NON-FREE FIXES

STRATEGIES #1



Home » Hardware

- Notebooks, netbooks, tablet PC
- Wifi cards
- Video cards
- Printers and multifunction
- Scanners
- 3G cards
- Sound cards
- Webcams
- Bluetooth devices
- TV/Video/FM acquisition cards

Login form:

username

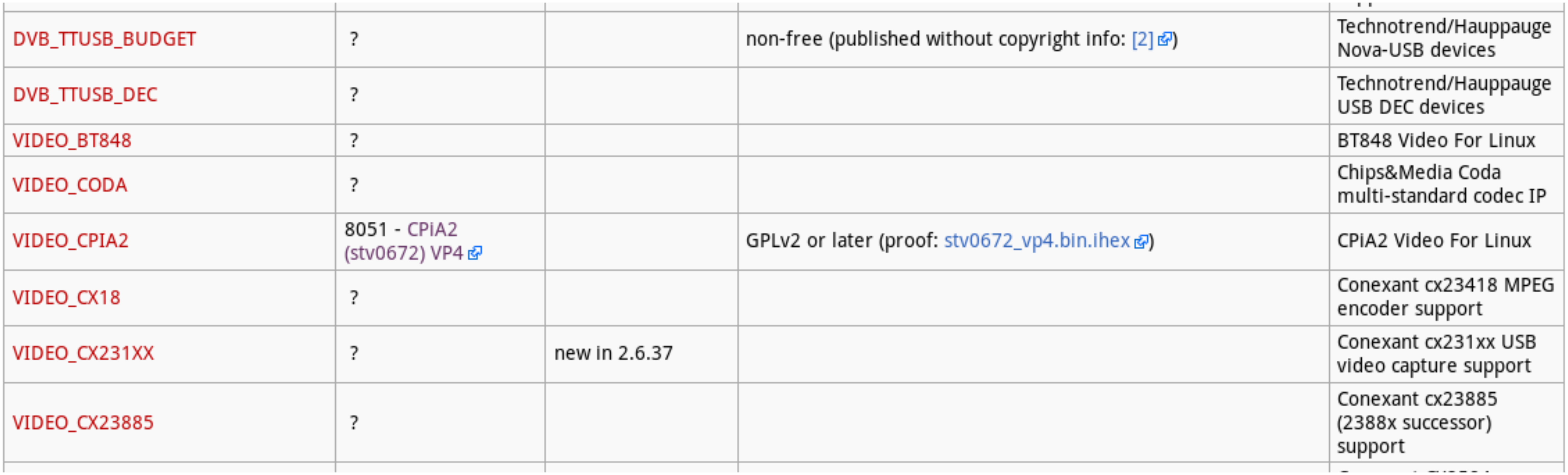
password

create new account

request new password

BUY NEW HARDWARE

STRATEGIES #2



DVB_TTUSB_BUDGET	?		non-free (published without copyright info: [2])	Technotrend/Hauppauge Nova-USB devices
DVB_TTUSB_DEC	?			Technotrend/Hauppauge USB DEC devices
VIDEO_BT848	?			BT848 Video For Linux
VIDEO_CODA	?			Chips&Media Coda multi-standard codec IP
VIDEO_CPIA2	8051 - CPiA2 (stv0672) VP4		GPLv2 or later (proof: stv0672_vp4.bin.ihex)	CPiA2 Video For Linux
VIDEO_CX18	?			Conexant cx23418 MPEG encoder support
VIDEO_CX231XX	?	new in 2.6.37		Conexant cx231xx USB video capture support
VIDEO_CX23885	?			Conexant cx23885 (2388x successor) support

EXTRACT SOURCE FROM GPL'D BINARIES

HW DEVICE: KEYSPAN USB-SERIAL

=> contained a bug in binary

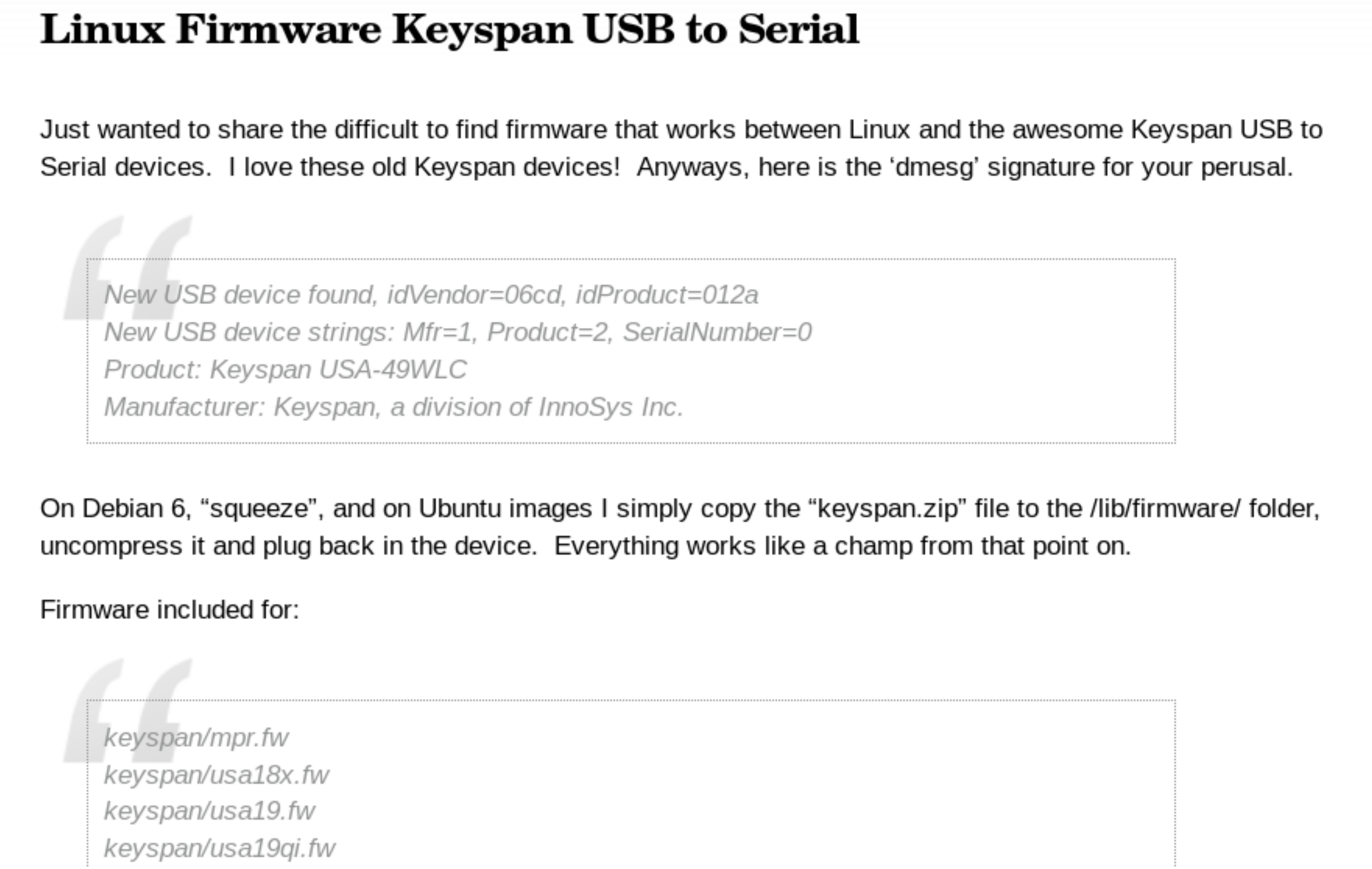
=> did not perfectly match
its libre assembly source code

=> we need build-rules for fw binaries
directly from the kernel source-tree

=> source and binaries are
likely to de-sync otherwise

EXTRACT SOURCE FROM GPL'D BINARIES

HW DEVICE: KEYSPAN USB-SERIAL



Linux Firmware Keyspan USB to Serial

Just wanted to share the difficult to find firmware that works between Linux and the awesome Keyspan USB to Serial devices. I love these old Keyspan devices! Anyways, here is the 'dmesg' signature for your perusal.

```
New USB device found, idVendor=06cd, idProduct=012a
New USB device strings: Mfr=1, Product=2, SerialNumber=0
Product: Keyspan USA-49WLC
Manufacturer: Keyspan, a division of InnoSys Inc.
```

On Debian 6, "squeeze", and on Ubuntu images I simply copy the "keyspan.zip" file to the /lib/firmware/ folder, uncompress it and plug back in the device. Everything works like a champ from that point on.

Firmware included for:

```
keyspan/mpr.fw
keyspan/usa18x.fw
keyspan/usa19.fw
keyspan/usa19qi.fw
```

EXTRACT SOURCE FROM GPL'D BINARIES

STRATEGIES #3

For development of free replacements:

=> time-consuming

=> not guaranteed to yield results
in a reasonable ammount of time

=> scarcity of man-power

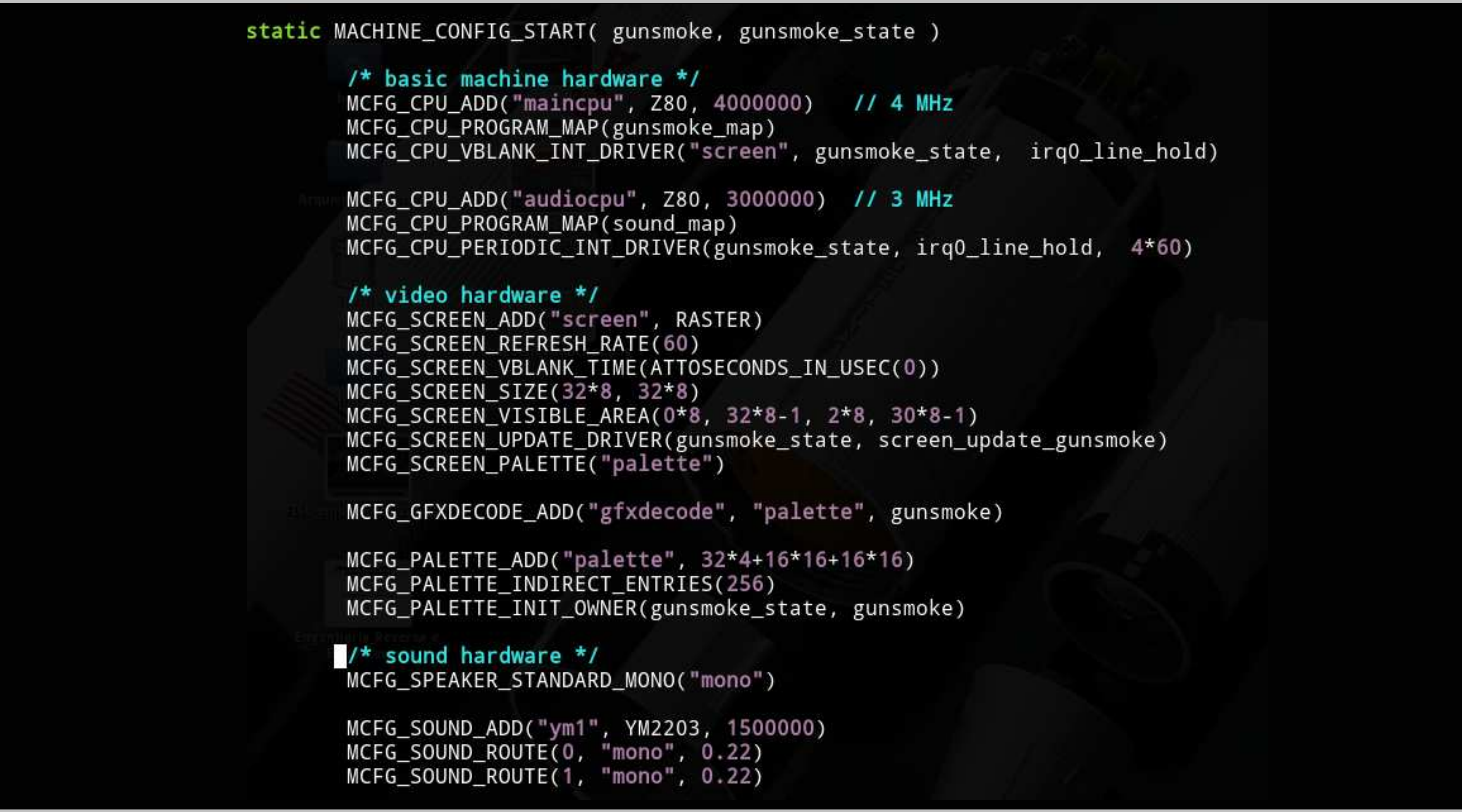
=> potential legal issues

HOW MAME WORKS

```
=> CPU core emulators
=> auxilary chips
=> drivers
  ==> schematics of a PCB
  ==> instantiation of cores
  ==> relationship between chips
  ==> memory map layouts
```

VERY BRIEF OVERVIEW

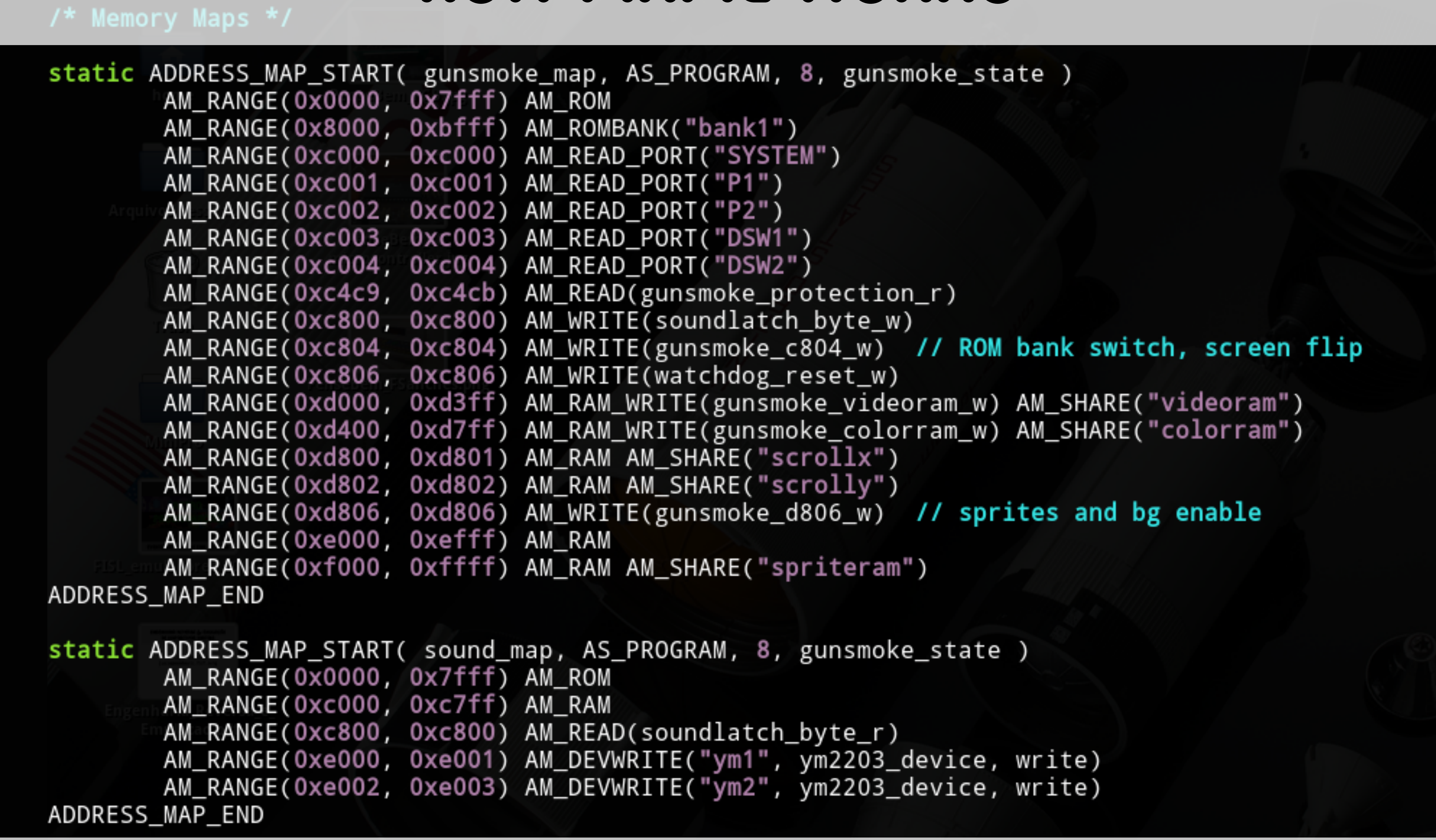
HOW MAME WORKS



```
static MACHINE_CONFIG_START( gunsmoke, gunsmoke_state )

        /* basic machine hardware */
        MCFG_CPU_ADD("maincpu", Z80, 4000000)   // 4 MHz
        MCFG_CPU_PROGRAM_MAP(gunsmoke_map)
        MCFG_CPU_VBLANK_INT_DRIVER("screen", gunsmoke_state,  irq0_line_hold)

        MCFG_CPU_ADD("audiocpu", Z80, 3000000)  // 3 MHz
        MCFG_CPU_PROGRAM_MAP(sound_map)
        MCFG_CPU_PERIODIC_INT_DRIVER(gunsmoke_state, irq0_line_hold,  4*60)

        /* video hardware */
        MCFG_SCREEN_ADD("screen", RASTER)
        MCFG_SCREEN_REFRESH_RATE(60)
        MCFG_SCREEN_VBLANK_TIME(ATTOSECONDS_IN_USEC(0))
        MCFG_SCREEN_SIZE(32*8, 32*8)
        MCFG_SCREEN_VISIBLE_AREA(0*8, 32*8-1, 2*8, 30*8-1)
        MCFG_SCREEN_UPDATE_DRIVER(gunsmoke_state, screen_update_gunsmoke)
        MCFG_SCREEN_PALETTE("palette")

        MCFG_GFXDECODE_ADD("gfxdecode", "palette", gunsmoke)

        MCFG_PALETTE_ADD("palette", 32*4+16*16+16*16)
        MCFG_PALETTE_INDIRECT_ENTRIES(256)
        MCFG_PALETTE_INIT_OWNER(gunsmoke_state, gunsmoke)

        /* sound hardware */
        MCFG_SPEAKER_STANDARD_MONO("mono")

        MCFG_SOUND_ADD("ym1", YM2203, 1500000)
        MCFG_SOUND_ROUTE(0, "mono", 0.22)
        MCFG_SOUND_ROUTE(1, "mono", 0.22)
```

INSTANTIATION OF EMULATION CORES

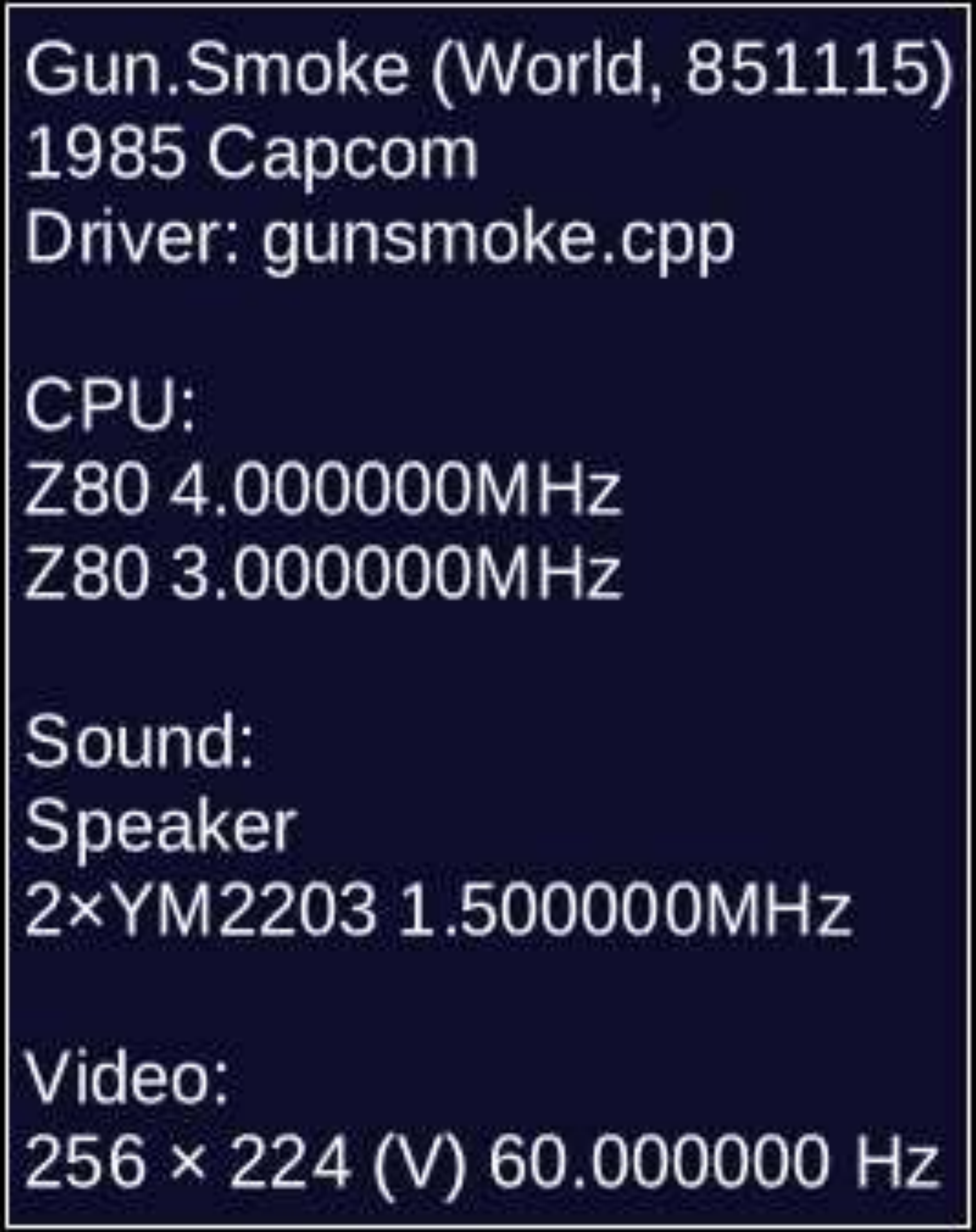
HOW MAME WORKS



```
/* Memory Maps */

static ADDRESS_MAP_START( gunsmoke_map, AS_PROGRAM, 8, gunsmoke_state )
	AM_RANGE(0x0000, 0x7fff) AM_ROM
	AM_RANGE(0x8000, 0xbfff) AM_ROMBANK("bank1")
	AM_RANGE(0xc000, 0xc000) AM_READ_PORT("SYSTEM")
	AM_RANGE(0xc001, 0xc001) AM_READ_PORT("P1")
	AM_RANGE(0xc002, 0xc002) AM_READ_PORT("P2")
	AM_RANGE(0xc003, 0xc003) AM_READ_PORT("DSW1")
	AM_RANGE(0xc004, 0xc004) AM_READ_PORT("DSW2")
	AM_RANGE(0xc4c9, 0xc4cb) AM_READ(gunsmoke_protection_r)
	AM_RANGE(0xc800, 0xc800) AM_WRITE(soundlatch_byte_w)
	AM_RANGE(0xc804, 0xc804) AM_WRITE(gunsmoke_c804_w)  // ROM bank switch, screen flip
	AM_RANGE(0xc806, 0xc806) AM_WRITE(watchdog_reset_w)
	AM_RANGE(0xd000, 0xd3ff) AM_RAM_WRITE(gunsmoke_videoram_w) AM_SHARE("videoram")
	AM_RANGE(0xd400, 0xd7ff) AM_RAM_WRITE(gunsmoke_colorram_w) AM_SHARE("colorram")
	AM_RANGE(0xd800, 0xd801) AM_RAM AM_SHARE("scrollx")
	AM_RANGE(0xd802, 0xd802) AM_RAM AM_SHARE("scrolly")
	AM_RANGE(0xd806, 0xd806) AM_WRITE(gunsmoke_d806_w)  // sprites and bg enable
	AM_RANGE(0xe000, 0xefff) AM_RAM
	AM_RANGE(0xf000, 0xffff) AM_RAM AM_SHARE("spriteram")
ADDRESS_MAP_END

static ADDRESS_MAP_START( sound_map, AS_PROGRAM, 8, gunsmoke_state )
	AM_RANGE(0x0000, 0x7fff) AM_ROM
	AM_RANGE(0xc000, 0xc7ff) AM_RAM
	AM_RANGE(0xc800, 0xc800) AM_READ(soundlatch_byte_r)
	AM_RANGE(0xe000, 0xe001) AM_DEVWRITE("ym1", ym2203_device, write)
	AM_RANGE(0xe002, 0xe003) AM_DEVWRITE("ym2", ym2203_device, write)
ADDRESS_MAP_END
```

MEMORY MAPS

HARDWARE METADATA



Gun.Smoke (World, 851115)
1985 Capcom
Driver: gunsmoke.cpp

CPU:
Z80 4.000000MHz
Z80 3.000000MHz

Sound:
Speaker
2×YM2203 1.500000MHz

Video:
256 × 224 (V) 60.000000 Hz

FIRMWARE CHECKSUMS AS WELL!

NON-FREE ROMS !!!

=> #1: Do not burn books!
 ==> Archival of culturaly relevant assets
 ==> History of computing

=> #2: Baremetal SW
 ==> evidence of HW caracteristics
 ==> can be leveraged (clean room)

SW FREEDOM CONSIDERATIONS

PROCEDURAL CONSIDERATIONS

=> Every firmware rev
is tagged in MAME

=> Similar procedures
for LinuxLibre?

COLLECT METADATA OF EVERYTHING!

REVERSE ENGINEERING EDUCATION

=> young people! They're curious!

=> Maker movement

=> hackerspaces as strategic labs

=> capacity-building on hw hacking

=> how to share discoveries ?

iN A LEGAL MANNER.

HW DEVICE: DREAMCAST SOUND

=> actually free firmware

=> LinuxLibre non-free
black-list false-positive

=> emulation in MAME helped
figure out technical details

=> source is available
under a free license

VERY OLD FW SOURCE CODE

HW DEVICE: DREAMCAST SOUND

```
Build results in diverging binary files:

 => old toolchain

 => validate via deployment on hardware ?

 => validate via emulation ?
```

WE NEED FW REPRODUCIBLE BUILDS

HW DEVICE: GALAXY S3 CAMERAS

=> front camera / back camera

=> issues on Replicant OS

=> could emulation help?

=> how to detect CPU architecture?

EXAMPLE: ARM OPCODE VALUE PATTERNS

QUESTIONS ?

Felipe Sanches
juca@members.fsf.org

HAPPY HACKING!