## Skeleton config file for RetroArch

# Save all save files (\*.srm) to this directory. This includes related files like .bsv, .rtc, .psrm, etc ...

# This will be overridden by explicit command line options.

# savefile\_directory =

# Save all save states (\*.state) to this directory.

# This will be overridden by explicit command line options.

# savestate\_directory =

# Automatically saves a savestate at the end of RetroArch's lifetime.

# The path is $SRAM\_PATH.auto.

# RetroArch will automatically load any savestate with this path on startup if savestate\_auto\_load is set.

# savestate\_auto\_save = false

# savestate\_auto\_load = true

# Load libretro from a dynamic location for dynamically built RetroArch.

# This option is mandatory.

# If a directory, RetroArch will look through the directory until it finds an implementation

# that appears to support the extension of the ROM loaded.

# This could fail if ROM extensions overlap.

# libretro\_path = "/path/to/libretro.so"

# Path to core options config file.

# This config file is used to expose core-specific options.

# It will be written to by RetroArch.

# A default path will be assigned if not set.

# core\_options\_path =

# Path to ROM load history file.

# RetroArch keeps track of all ROMs loaded in RGUI and from CLI directly for convenient quick loading.

# A default path will be assigned if not set.

# game\_history\_path =

# Number of entries that will be kept in ROM history file.

# game\_history\_size = 100

# Sets the "system" directory.

# Implementations can query for this directory to load BIOSes, system-specific configs, etc.

# system\_directory = /home/pi/RetroPie/BIOS

# Sets start directory for RGUI ROM browser.

# rgui\_browser\_directory =

# Sets start directory for RGUI config browser.

# rgui\_config\_directory =

# Show startup screen in RGUI.

# Is automatically set to false when seen for the first time.

# This is only updated in config if config\_save\_on\_exit is set to true, however.

# rgui\_show\_start\_screen = true

# Flushes config to disk on exit. Useful for RGUI as settings can be modified.

# Overwrites the config. #include's and comments are not preserved.

# config\_save\_on\_exit = false

#### Video

# Video driver to use. "gl", "xvideo", "sdl"

# video\_driver = "sdl"

# Which OpenGL context implementation to use.

# Possible ones for desktop are: glx, x-egl, kms-egl, sdl-gl, wgl.

# By default, tries to use first suitable driver.

# video\_gl\_context =

# Windowed xscale and yscale

# (Real x res: base\_size \* xscale \* aspect\_ratio, real y res: base\_size \* yscale)

# video\_xscale = 3.0

# video\_yscale = 3.0

# Fullscreen resolution. Resolution of 0 uses the resolution of the desktop.

# video\_fullscreen\_x = 0

# video\_fullscreen\_y = 0

# Start in fullscreen. Can be changed at runtime.

# video\_fullscreen = false

# If fullscreen, prefer using a windowed fullscreen mode.

# video\_windowed\_fullscreen = true

# Which monitor to prefer. 0 (default) means no particular monitor is preferred, 1 and up (1 being first monitor),

# suggests RetroArch to use that particular monitor.

# video\_monitor\_index = 0

# Forcibly disable composition. Only works in Windows Vista/7 for now.

# video\_disable\_composition = false

# Video vsync.

# video\_vsync = true

# Attempts to hard-synchronize CPU and GPU. Can reduce latency at cost of performance.

# video\_hard\_sync = false

# Sets how many frames CPU can run ahead of GPU when using video\_hard\_sync.

# Maximum is 3.

# video\_hard\_sync\_frames = 0

# Inserts a black frame inbetween frames.

# Useful for 120 Hz monitors who want to play 60 Hz material with eliminated ghosting.

# video\_refresh\_rate should still be configured as if it is a 60 Hz monitor (divide refresh rate by 2).

# video\_black\_frame\_insertion = false

# Use threaded video driver. Using this might improve performance at possible cost of latency and more video stuttering.

# video\_threaded = true

# Smoothens picture with bilinear filtering. Should be disabled if using pixel shaders.

# video\_smooth = true

# Forces rendering area to stay equal to game aspect ratio or as defined in video\_aspect\_ratio.

# video\_force\_aspect = true

# Only scales video in integer steps.

# The base size depends on system-reported geometry and aspect ratio.

# If video\_force\_aspect is not set, X/Y will be integer scaled independently.

# video\_scale\_integer = false

# A floating point value for video aspect ratio (width / height).

# If this is not set, aspect ratio is assumed to be automatic.

# Behavior then is defined by video\_aspect\_ratio\_auto.

# video\_aspect\_ratio =

# If this is true and video\_aspect\_ratio is not set,

# aspect ratio is decided by libretro implementation.

# If this is false, 1:1 PAR will always be assumed if video\_aspect\_ratio is not set.

# video\_aspect\_ratio\_auto = false

# Forces cropping of overscanned frames.

# Exact behavior of this option is implementation specific.

# video\_crop\_overscan = true

# Path to shader. Shader can be either Cg, CGP (Cg preset) or XML/GLSL format if support is enabled.

# video\_shader = "/path/to/shader.{cg,cgp,shader}"

# Load video\_shader on startup.

# Other shaders can still be loaded later in runtime.

# video\_shader\_enable = false

# Defines a directory where shaders (Cg, CGP, XML) are kept for easy access.

# video\_shader\_dir =

# CPU-based filter. Path to a bSNES CPU filter (\*.filter)

# video\_filter =

# Path to a TTF font used for rendering messages. This path must be defined to enable fonts.

# Do note that the \_full\_ path of the font is necessary!

# video\_font\_path =

# Size of the TTF font rendered.

# video\_font\_size = 48

# Attempt to scale the font to fit better for multiple window sizes.

# video\_font\_scale = true

# Enable usage of OSD messages.

# video\_font\_enable = true

# Offset for where messages will be placed on screen. Values are in range 0.0 to 1.0 for both x and y values.

# [0.0, 0.0] maps to the lower left corner of the screen.

# video\_message\_pos\_x = 0.05

# video\_message\_pos\_y = 0.05

# Color for message. The value is treated as a hexadecimal value.

# It is a regular RGB hex number, i.e. red is "ff0000".

# video\_message\_color = ffffff

# Video refresh rate of your monitor.

# Used to calculate a suitable audio input rate.

# video\_refresh\_rate = 59.95

# Allows libretro cores to set rotation modes.

# Setting this to false will honor, but ignore this request.

# This is useful for vertically oriented games where one manually rotates the monitor.

# video\_allow\_rotate = true

# Forces a certain rotation of the screen.

# The rotation is added to rotations which the libretro core sets (see video\_allow\_rotate).

# The angle is <value> \* 90 degrees counter-clockwise.

# video\_rotation = 0

#### Audio

# Enable audio.

# audio\_enable = true

# Audio output samplerate.

# audio\_out\_rate = 44100

# Audio driver backend. Depending on configuration possible candidates are: alsa, pulse, oss, jack, rsound, roar, openal, sdl, xaudio.

# audio\_driver =sdl

# Override the default audio device the audio\_driver uses. This is driver dependant. E.g. ALSA wants a PCM device, OSS wants a path (e.g. /dev/dsp), Jack wants portnames (e.g. system:playback1,system:playback\_2), and so on ...

# audio\_device =

# External DSP plugin that processes audio before it's sent to the driver.

# audio\_dsp\_plugin =

# Will sync (block) on audio. Recommended.

# audio\_sync = true

# Desired audio latency in milliseconds. Might not be honored if driver can't provide given latency.

# audio\_latency = 64

# Enable experimental audio rate control.

# audio\_rate\_control = true

# Controls audio rate control delta. Defines how much input rate can be adjusted dynamically.

# Input rate = in\_rate \* (1.0 +/- audio\_rate\_control\_delta)

# audio\_rate\_control\_delta = 0.005

# Audio volume. Volume is expressed in dB.

# 0 dB is normal volume. No gain will be applied.

# Gain can be controlled in runtime with input\_volume\_up/input\_volume\_down.

# audio\_volume = 0.0

#### Input

# Input driver. Depending on video driver, it might force a different input driver.

# input\_driver = sdl

# Joypad driver. (Valid: linuxraw, sdl, dinput)

# input\_joypad\_driver =

# Defines axis threshold. Possible values are [0.0, 1.0]

# input\_axis\_threshold = 0.5

# Path to input overlay

# input\_overlay =

# Overlay opacity

# input\_overlay\_opacity = 1.0

# Overlay scale

# input\_overlay\_scale = 1.0

# Enable input auto-detection. Will attempt to autoconfigure

# joypads, Plug-and-Play style.

# input\_autodetect\_enable = true

# Directory for joypad autoconfigs (PC).

# If a joypad is plugged in, that joypad will be autoconfigured if a config file

# corresponding to that joypad is present in joypad\_autoconfig\_dir.

# Input binds which are made explicit (input\_playerN\_\*\_btn/axis) will take priority over autoconfigs.

# Autoconfigs can be created with retroarch-joyconfig, manually, or with a frontend.

# Requires input\_autodetect\_enable to be enabled.

# joypad\_autoconfig\_dir =

# Enable debug input key reporting on-screen.

# input\_debug\_enable = false

# Sets which libretro device is used for a player.

# Devices are indentified with a number.

# This is normally saved by RGUI.

# Device IDs are found in libretro.h.

# These settings are overridden by explicit command-line arguments which refer to input devices.

# None: 0

# Joypad (RetroPad): 1

# Mouse: 2

# Keyboard: 3

# Generic Lightgun: 4

# Joypad w/ Analog (RetroPad + Analog sticks): 5

# Multitap (SNES specific): 257

# Super Scope (SNES specific): 260

# Justifier (SNES specific): 516

# Justifiers (SNES specific): 772

# input\_libretro\_device\_p1 =

# input\_libretro\_device\_p2 =

# input\_libretro\_device\_p3 =

# input\_libretro\_device\_p4 =

# input\_libretro\_device\_p5 =

# input\_libretro\_device\_p6 =

# input\_libretro\_device\_p7 =

# input\_libretro\_device\_p8 =

# Keyboard input. Will recognize normal keypresses and special keys like "left", "right", and so on.

# Keyboard input, Joypad and Joyaxis will all obey the "nul" bind, which disables the bind completely,

# rather than relying on a default.

# input\_player1\_a = x

# input\_player1\_b = z

# input\_player1\_y = a

# input\_player1\_x = s

# input\_player1\_start = enter

# input\_player1\_select = rshift

# input\_player1\_l = q

# input\_player1\_r = w

# input\_player1\_left = left

# input\_player1\_right = right

# input\_player1\_up = up

# input\_player1\_down = down

# input\_player1\_l2 =

# input\_player1\_r2 =

# input\_player1\_l3 =

# input\_player1\_r3 =

# Two analog sticks (DualShock-esque).

# Bound as usual, however, if a real analog axis is bound,

# it can be read as a true analog.

# Positive X axis is right, Positive Y axis is down.

# input\_player1\_l\_x\_plus =

# input\_player1\_l\_x\_minus =

# input\_player1\_l\_y\_plus =

# input\_player1\_l\_y\_minus =

# input\_player1\_r\_x\_plus =

# input\_player1\_r\_x\_minus =

# input\_player1\_r\_y\_plus =

# input\_player1\_r\_y\_minus =

# If desired, it is possible to override which joypads are being used for player 1 through 8.

# First joypad available is 0.

# input\_player1\_joypad\_index = 0

# input\_player2\_joypad\_index = 1

# input\_player3\_joypad\_index = 2

# input\_player4\_joypad\_index = 3

# input\_player5\_joypad\_index = 4

# input\_player6\_joypad\_index = 5

# input\_player7\_joypad\_index = 6

# input\_player8\_joypad\_index = 7

# Joypad buttons.

# Figure these out by using RetroArch-Phoenix or retroarch-joyconfig.

# You can use joypad hats with hnxx, where n is the hat, and xx is a string representing direction.

# E.g. "h0up"

# input\_player1\_a\_btn =

# input\_player1\_b\_btn =

# input\_player1\_y\_btn =

# input\_player1\_x\_btn =

# input\_player1\_start\_btn =

# input\_player1\_select\_btn =

# input\_player1\_l\_btn =

# input\_player1\_r\_btn =

# input\_player1\_left\_btn =

# input\_player1\_right\_btn =

# input\_player1\_up\_btn =

# input\_player1\_down\_btn =

# input\_player1\_l2\_btn =

# input\_player1\_r2\_btn =

# input\_player1\_l3\_btn =

# input\_player1\_r3\_btn =

# Axis for RetroArch D-Pad.

# Needs to be either '+' or '-' in the first character signaling either positive or negative direction of the axis, then the axis number.

# Do note that every other input option has the corresponding \_btn and \_axis binds as well; they are omitted here for clarity.

# input\_player1\_left\_axis =

# input\_player1\_right\_axis =

# input\_player1\_up\_axis =

# input\_player1\_down\_axis =

# Holding the turbo while pressing another button will let the button enter a turbo mode

# where the button state is modulated with a periodic signal.

# The modulation stops when the button itself (not turbo button) is released.

# input\_player1\_turbo =

# Describes the period and how long of that period a turbo-enabled button should behave.

# Numbers are described in frames.

# input\_turbo\_period = 6

# input\_turbo\_duty\_cycle = 3

# This goes all the way to player 8 (\*\_player2\_\*, \*\_player3\_\*, etc), but omitted for clarity.

# All input binds have corresponding binds for keyboard (none), joykeys (\_btn) and joyaxes (\_axis) as well.

# Toggles fullscreen.

# input\_toggle\_fullscreen = f

# Saves state.

# input\_save\_state = f2

# Loads state.

# input\_load\_state = f4

# State slots. With slot set to 0, save state name is \*.state (or whatever defined on commandline).

# When slot is != 0, path will be $path%d, where %d is slot number.

# input\_state\_slot\_increase = f7

# input\_state\_slot\_decrease = f6

# Toggles between fast-forwarding and normal speed.

# input\_toggle\_fast\_forward = space

# Hold for fast-forward. Releasing button disables fast-forward.

# input\_hold\_fast\_forward = l

# Key to exit emulator cleanly.

# Killing it in any hard way (SIGTERM, SIGKILL, etc, will terminate emulator without saving RAM, etc.)

# input\_exit\_emulator = escape

# Applies next and previous XML/Cg shader in directory.

# input\_shader\_next = m

# input\_shader\_prev = n

# Hold button down to rewind. Rewinding must be enabled.

# input\_rewind = r

# Toggle between recording and not.

# input\_movie\_record\_toggle = o

# Toggle between paused and non-paused state

# input\_pause\_toggle = p

# Frame advance when game is paused

# input\_frame\_advance = k

# Reset the game.

# input\_reset = h

# Configures DSP plugin

# input\_dsp\_config = c

# Cheats.

# input\_cheat\_index\_plus = y

# input\_cheat\_index\_minus = t

# input\_cheat\_toggle = u

# Mute/unmute audio

# input\_audio\_mute = f9

# Take screenshot

# input\_screenshot = f8

# Netplay flip players.

# input\_netplay\_flip\_players = i

# Hold for slowmotion.

# input\_slowmotion = e

# Enable other hotkeys.

# If this hotkey is bound to either keyboard, joybutton or joyaxis,

# all other hotkeys will be disabled unless this hotkey is also held at the same time.

# This is useful for RETRO\_KEYBOARD centric implementations

# which query a large area of the keyboard, where it is not desirable

# that hotkeys get in the way.

# Alternatively, all hotkeys for keyboard could be disabled by the user.

# input\_enable\_hotkey =

# Increases audio volume.

# input\_volume\_up = kp\_plus

# Decreases audio volume.

# input\_volume\_down = kp\_minus

# Toggles to next overlay. Wraps around.

# input\_overlay\_next =

# Toggles eject for disks. Used for multiple-disk games.

# input\_disk\_eject\_toggle =

# Cycles through disk images. Use after ejecting.

# Complete by toggling eject again.

# input\_disk\_next =

# Toggles RGUI menu.

# input\_menu\_toggle = f1

# Toggles mouse grab. When mouse is grabbed, RetroArch hides the mouse,

# and keeps the mouse pointer inside the window to allow relative mouse games

# to work better.

# input\_grab\_mouse\_toggle = f11

#### Misc

# Enable rewinding. This will take a performance hit when playing, so it is disabled by default.

# rewind\_enable = false

# Rewinding buffer size in megabytes. Bigger rewinding buffer means you can rewind longer.

# The buffer should be approx. 20MB per minute of buffer time.

# rewind\_buffer\_size = 20

# Rewind granularity. When rewinding defined number of frames, you can rewind several frames at a time, increasing the rewinding speed.

# rewind\_granularity = 1

# Pause gameplay when window focus is lost.

# pause\_nonactive = true

# Autosaves the non-volatile SRAM at a regular interval. This is disabled by default unless set otherwise.

# The interval is measured in seconds. A value of 0 disables autosave.

# autosave\_interval =

# When being client over netplay, use keybinds for player 1.

# netplay\_client\_swap\_input = false

# Path to XML cheat database (as used by bSNES).

# cheat\_database\_path =

# Path to XML cheat config, a file which keeps track of which

# cheat settings are used for individual games.

# If the file does not exist, it will be created.

# cheat\_settings\_path =

# Directory to dump screenshots to.

# screenshot\_directory =

# Records video after CPU video filter.

# video\_post\_filter\_record = false

# Records output of GPU shaded material if available.

# video\_gpu\_record = false

# Screenshots output of GPU shaded material if available.

# video\_gpu\_screenshot = true

# Block SRAM from being overwritten when loading save states.

# Might potentially lead to buggy games.

# block\_sram\_overwrite = false

# When saving a savestate, save state index is automatically increased before

# it is saved.

# Also, when loading a ROM, the index will be set to the highest existing index.

# There is no upper bound on the index.

# savestate\_auto\_index = false

# Slowmotion ratio. When slowmotion, game will slow down by factor.

# slowmotion\_ratio = 3.0

# The maximum rate at which games will be run when using fast forward. (E.g. 5.0 for 60 fps game => 300 fps cap).

# RetroArch will go to sleep to ensure that the maximum rate will not be exceeded.

# Do not rely on this cap to be perfectly accurate.

# A negative ratio equals no FPS cap.

# fastforward\_ratio = -1.0

# Enable stdin/network command interface.

# network\_cmd\_enable = false

# network\_cmd\_port = 55355

# stdin\_cmd\_enable = false

# input\_exit\_emulator\_btn = "10"