RetroArch Android (v0.9.9)

Hans Kristian Arntzen, Daniel De Matteis, Michael Lelli

May 27, 2013

Contents

1 Introduction

RetroArch Android is an app that has been designed to run and play:

- Games
- Emulators
- Tech demos (OpenGL and non-OpenGL)

Emulators and games that can be run on RetroArch come in the form of pluggable 'engines' which are called 'libretro cores'. The version that you just installed already has most of the full library of 'cores' preinstalled.

2 Disclaimer

RetroArch Android is released for free and will always be free. There are no ads (push or otherwise), there is no 'spying' going on in the form of analytics or collecting stats, there is no 'paid DLC', and on and on - all the unsavory and bad aspects of this ’new generation of computing’ are not to be found here. It will never be sold with a pricetag - not even disguised as a ’donationware version’. If you happen to have ’paid’ for RetroArch Android or a derivative of it, you have been scammed and you should probably demand your money back from the scam artist in question (and scam artists they are).

Just because the GPL allows people to make derivative copies of RetroArch for commercial purposes does not mean that we support it or even approve of it. If you sell RetroArch or a derivative copy of it for any commercial purpose, you are part of the problem and you need to be learnt a quick lesson in etiquette. Note to any ’entrepreneurs’ out there that might be tempted by this ’easy route to makin’ some money’ - I honestly wouldn’t bother - we will undercut you by offering this all for free and doing a better job at it to boot. That and I severely doubt you can come up with many trinkets that will persuade people to throw away their money on a derivative version when they can have it all for free to begin with - just saying - save yourself the time and the effort, because it isn’t going to work out.

3 How to run

At startup, RetroArch Android shows you a list view of ’cores’. A ’libretro core’ supports games with certain extensions. To consult which core does what, you should read the ’RetroArch Cores Manual’.

3.1 Select a core

Select one of these cores in the menu.

3.2 Select a game

Cores need data files in order to work. Most emulator-based cores will typically need a ’ROM’ image, game-based cores will typically need some kind of ’executable’.

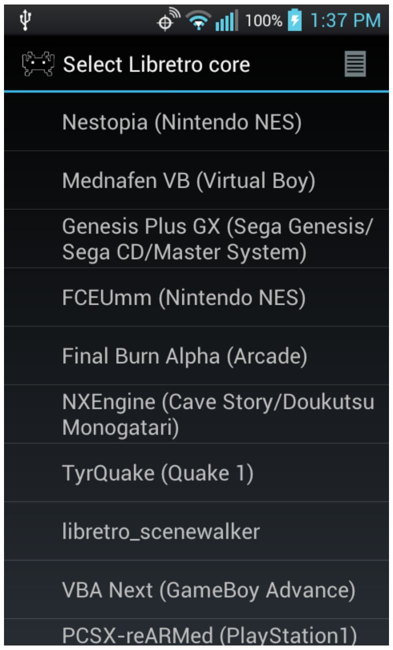
After you have selected the core, you will now need to select a compatible data file from the filebrowser. It will then attempt to load the core with that specific data file (in most cases - a game image).

4 Controls

4.1 Touchscreen overlay

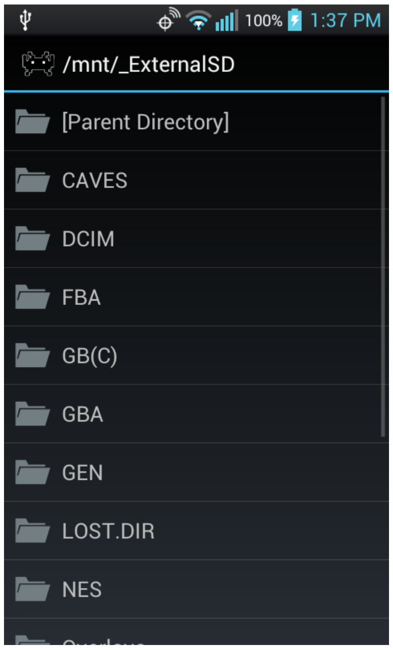
RetroArch Android uses an overlay as a ’mock’ gamepad to play with. The ’overlay’ controls will always be bound to Player 1.

The gamepad abstraction that you see here is what we call the ’RetroPad’. When you make a libretro core and when your core supports a gamepad, it will always use as its gamepad model this pad.



Nestopia (Nintendo NES)
Mednafen VB (Virtual Boy)
Genesis Plus GX (Sega Genesis/ Sega CD/Master System)
FCEUmm (Nintendo NES)
Final Burn Alpha (Arcade)
NXEngine (Cave Story/Doukutsu Monogatari)
TyrQuake (Quake 1)
libretro_scenewalker
VBA Next (GameBoy Advance)
PCSX-reARMed (PlayStation1)

Figure 1: Select a core from this menu.



[Parent Directory]
CAVES
DCIM
FBA
GB(C)
GBA
GEN
LOST.DIR
NES

Figure 2: After selecting the core, you will need to load a game.

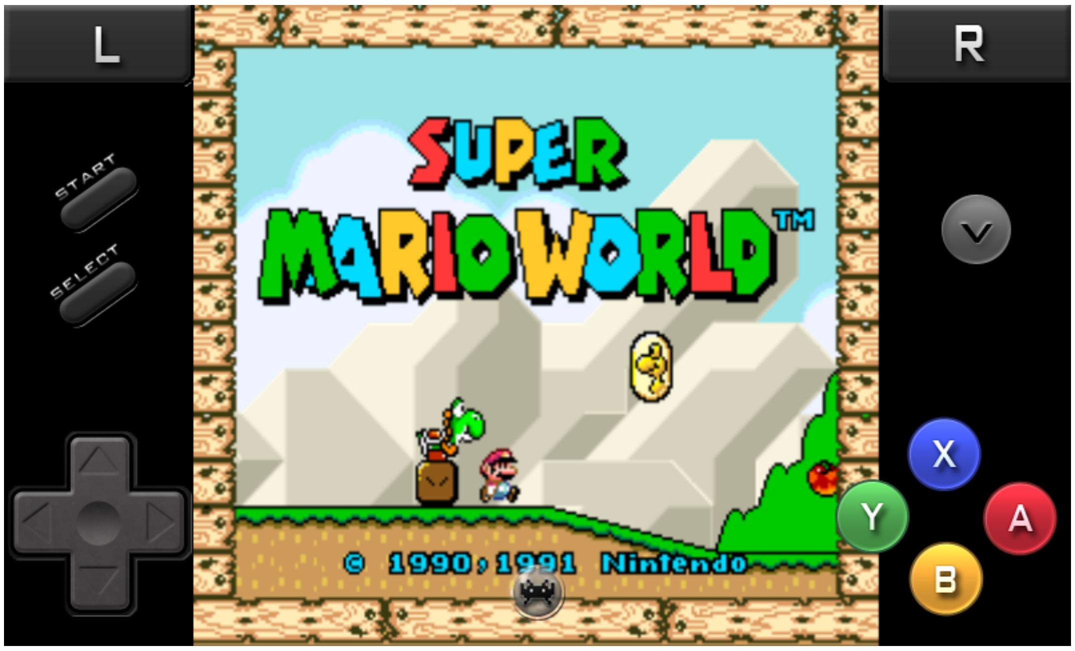


Figure 3: 'RetroPad overlay' screen.

The RetroPad has the same face and shoulder buttons as a Super Nintendo gamepad. In addition to this, it also has the L2/R2/L3/R3 additional buttons and the twin analog sticks from a Sony DualShock gamepad.

4.2 Touchscreen menu navigation

Each touchscreen overlay has a couple of screens that can be navigated to. To go to the next screen of the overlay, you press the 'circle' icon at the bottom.

Most of the overlays that come bundled with RetroArch Android have the same screen order.

4.2.1 Gamepad screen

You can control the game with this screen. It can also be used inside RGUI for navigating the menu.

4.2.2 Quick Menu screen

The actions on this screen have various effects on the game currently running.

- LOAD STATE - Load a save state from the currently selected save state slot.
- SAVE STATE - Save state to the currently selected save state slot.

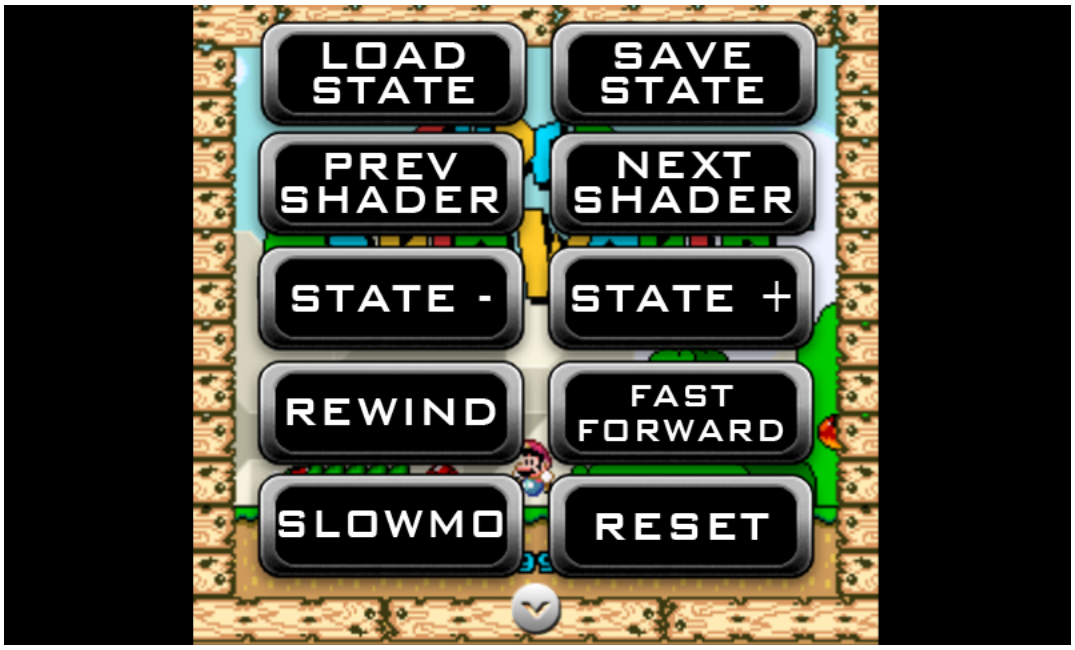


Figure 4: 'Quick Menu' screen.

- STATE MINUS - Go back one save state slot.
- STATE PLUS - Go forward one state slot.
- REWIND - Rewind the game in real-time. Note - the 'Rewind' option needs to be enabled at the Settings menu or else this option won't work.
- SLOWMOTION - Press and hold this button to let the game run in slow-motion.
- RESET - Resets the game/system.
- FAST FORWARD - Fast forward the game in real-time.
- NEXT SHADER - Load the next shader in the folder (NOTE: only if shaders are enabled)
- PREVIOUS SHADER - Load the previous shader in the folder (NOTE: only if shaders are enabled)

4.2.3 Gameplay screen

This screen is useful for when you are playing with an USB or Bluetooth gamepad but you would still like to have access to the Quick Menu or Gamepad

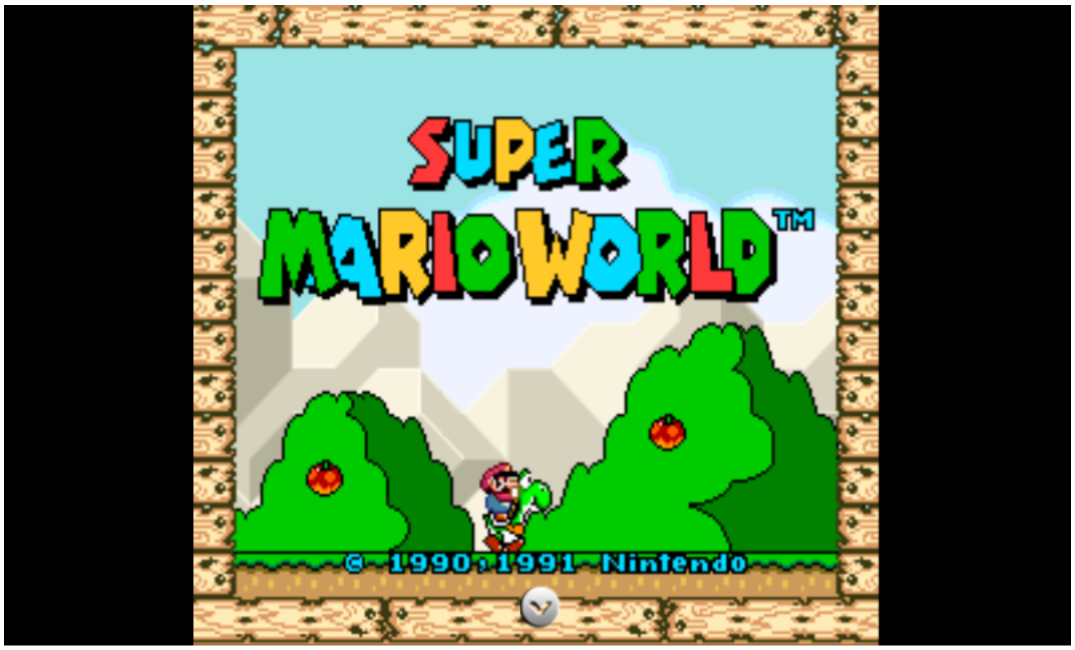


Figure 5: 'Gameplay' screen.

screen without outright disabling overlays. If you press the 'icon' at the bottom of this screen, you will go back to the 'Gamepad' screen'.

4.3 Variations

RetroArch Android comes packaged with a number of different-looking overlays. Below is an image showing the different overlays:

You can select between a number of different overlays from either the Settings menu and/or RGUI.

4.3.1 Making your own custom overlays

You can make your own custom overlays for use with RetroArch Android. If you want to learn how to do this, you should read the 'Overlay Guide'.

4.4 USB gamepads

Next to the standard touchscreen input, RetroArch Android autodetects and autoconfigures various input devices automatically. Most of these are USB joysticks/gamepads.

A list of the gamepads that are supported by autodetection can be found inside the app. (go to Help).

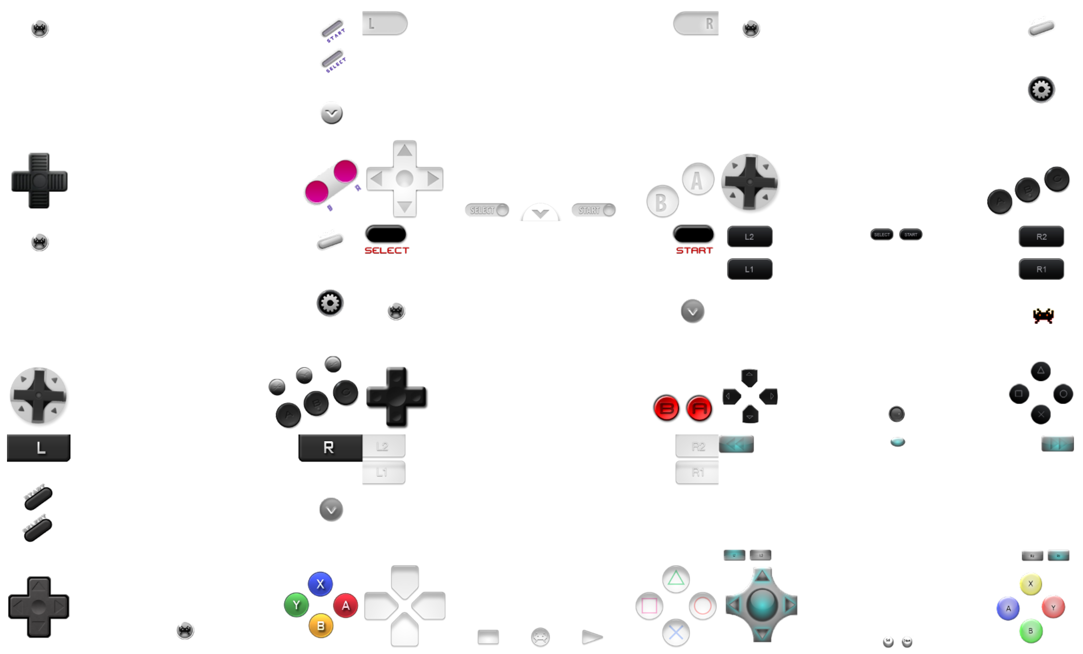


Figure 6: All the default high-resolution overlays packaged with RetroArch Android.

You connect the device to your tablet/phone. You press a button while ingame. If your pad is supported, it should bring up a message saying: “RetroPad #? detected: “ and then the name of the device it found. Buttons and control layout will then be autoconfigured and mapped to the RetroPad layout.

4.4.1 Unsupported gamepads

If your pad is unsupported, it will likely show “Unknown HID” instead. If you want this pad supported, contact us.

4.4.2 Notes

If a USB gamepad that is listed above does not work immediately, your controller may require a powered USB hub or perhaps a HID driver may be missing of sorts.

4.5 Bluetooth

RetroArch supports Bluetooth right now only through the use of IME apps. A couple of IME apps are supported by RetroArch Android - if you use the default key layouts with the IME apps listed below, your pads will be automatically configured:

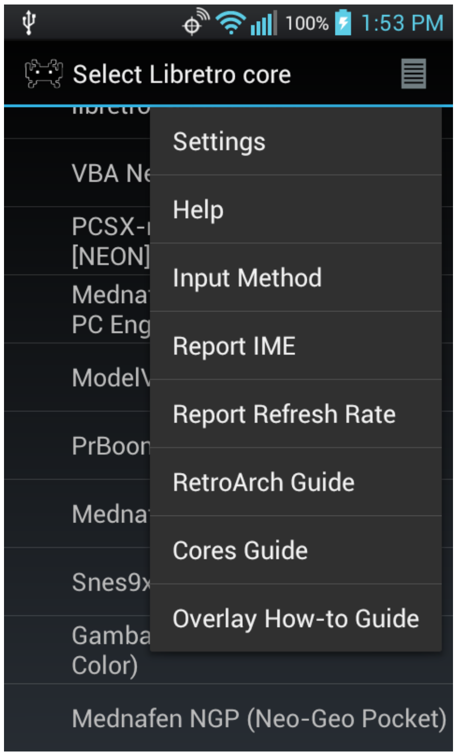


Figure 7: Setting an IME app from the RetroArch menu by clicking on the ’Settings’ icon.

- https://play.google.com/store/apps/details?id=com.dancingpixelstudios.sixaxiscontroller Dancing Pixel Studios SixAxis Controller
- https://play.google.com/store/apps/details?id=com.ccpcreations.android.WiiUseAndroid ccpcreations.Wiiuse.Android

Remember that you will have to change your Input Method to the needed IME first before starting RetroArch. This can also be done from the menu by clicking on the top righthand side 'Settings' icon and then selecting 'Input Method' (see image).

4.6 Notes

When using PS3 controller via Bluetooth, use SixAxis adapter app and after you've got the controller setup, make sure to go to menu then preferences and then Game pad settings, and enable Gamepad. This turns it into a native android controller and no IME switch is needed. Same for the MOGA controller via bluetooth, make sure to use the MOGA Universal Driver and not the one that MOGA recommends. In the app, make sure 'Enable left analog input' is checked, and that it's in System Mode to make it a native gamepad for android and no need to switch IMEs.

5 Settings

You can configure many aspects of RetroArch. To go to the Settings menu, click on the 'Settings' icon at the top righthand side of the screen and then select 'Settings'.

5.1 Path Settings

- ROM Directory
  Set the directory that will be used as a default starting point for the filebrowser.
- Save Files - Enable custom directory
  Enables use of custom save file folder. (.srm) save files will be saved and loaded to the configured directory. if not enabled, save files will reside in ROM folder.
- Save Files - Savefile directory
  Sets directory where to save and load game save files.
- Save States - Enable custom directory
  Enables use of custom save statefolder. (.state) save states will be saved and loaded to configured directory. If not enabled, save states will reside in ROM folder.

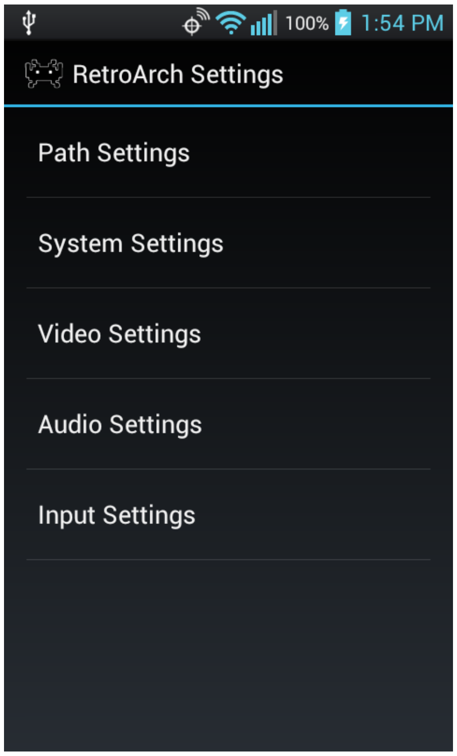
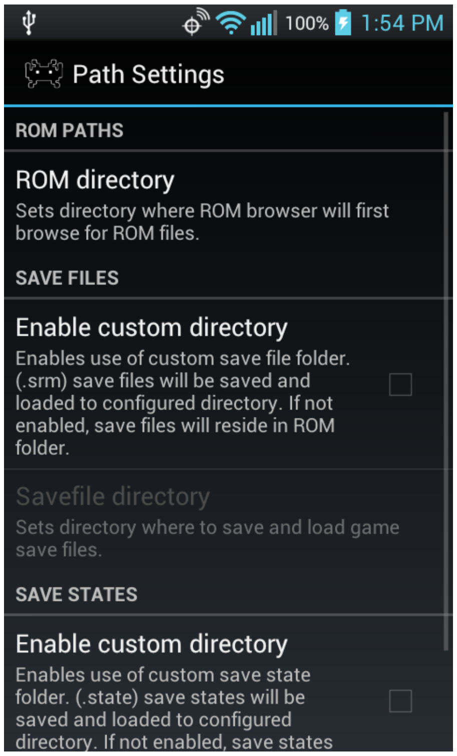


Figure 8: ’Settings’ menu.



1:54 PM
100%

Path Settings

ROM PATHS

ROM directory
Sets directory where ROM browser will first browse for ROM files.

SAVE FILES

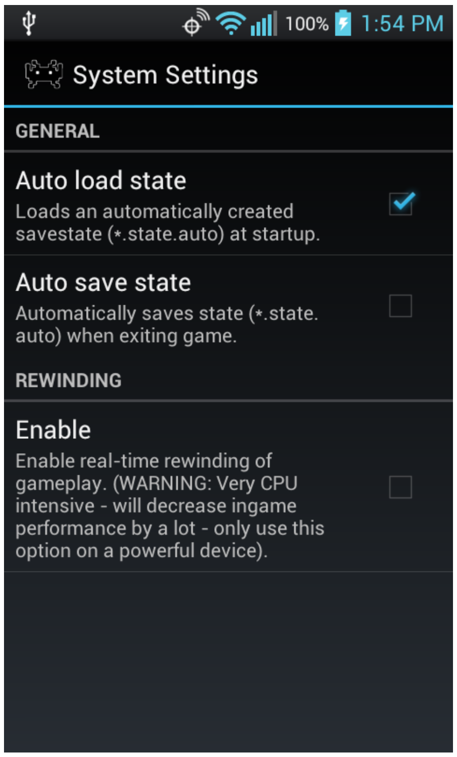
Enable custom directory
Enables use of custom save file folder. (.srm) save files will be saved and loaded to configured directory. If not enabled, save files will reside in ROM folder.

Savefile directory
Sets directory where to save and load game save files.

SAVE STATES

Enable custom directory
Enables use of custom save state folder. (.state) save states will be saved and loaded to configured directory. If not enabled, save states

Figure 9: 'Path Settings' screen.



1:54 PM

System Settings

GENERAL

Auto load state [✓]
Loads an automatically created savestate (*.state.auto) at startup.

Auto save state [ ]
Automatically saves state (*.state. auto) when exiting game.

REWINDING

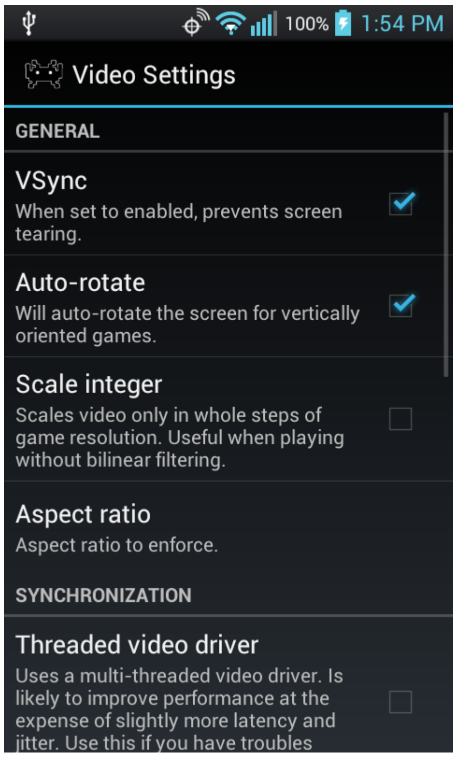
Enable [ ]
Enable real-time rewinding of gameplay. (WARNING: Very CPU intensive - will decrease ingame performance by a lot - only use this option on a powerful device).

Figure 10: 'System Settings' screen.

- Save state directory
  Sets directory where to save and load game save states.
- Enable custom directory
  Enables use of custom system folder. Cores will look for system specific files, like BIOSes, in this folder. If not enabled, it will look in the ROM folder instead.
- System directory
  Sets directory where system files are loaded from.

5.2 System Settings

- Auto load state
  Loads an automatically created savestate at startup.
- Auto save state
  This will make a save state when you exit the game. This auto savestate will be automatically loaded the next time you start up the game. Useful for on-the-go gaming.


100% 1:54 PM

Video Settings

GENERAL

VSync
When set to enabled, prevents screen tearing. ✓

Auto-rotate
Will auto-rotate the screen for vertically oriented games. ✓

Scale integer
Scales video only in whole steps of game resolution. Useful when playing without bilinear filtering. ☐

Aspect ratio
Aspect ratio to enforce.

SYNCHRONIZATION

Threaded video driver
Uses a multi-threaded video driver. Is likely to improve performance at the expense of slightly more latency and jitter. Use this if you have troubles ☐

Figure 11: ’Video Settings’ screen.

- Rewinding Enable
  This allows you to rewind the game in real-time to undo ’mistakes’ you made while playing the game. (NOTE - this is very CPU intensive - you should only enable this if the core is running at least 2x realtime on your system).

5.3 Video Settings

- Vsync
  Unchecking this will cause screen tearing but faster performance.

- Auto-rotate
  Will auto-rotate the screen for vertically oriented games.

- Scale integer
  Scales video only in whole steps of game resolution. useful when playing without bilinear filtering.

- Aspect ratio
  Select the aspect ratio to enforce.

- Threaded video driver
  Uses a multi-threaded video driver. It is likely to improve performance at the expense of slightly more latency and jitter. Use this if you have troubles getting good video and/or audio.

- Forced refresh rate (Hz)
  Force a specific refresh rate to be detected. Only use this if auto-detection of refresh rate reports wrong refresh rate.

- Calibrate refresh rate
  Attempts to find the true refresh rate of a monitor. Updates value in 'Force refresh rate (Hz)' option. To help ensure accuracy, make sure no intense background services are running, and avoid triggering screensaver.

- Set OS-reported refresh rate
  Sets refresh rate equal to OS-reported value. This might not be accurate for your device.

- Shaders (1st pass) Bilinear filter
  Applies bilinear filtering, smooths out edges (setting still apply even if no shader is selected).

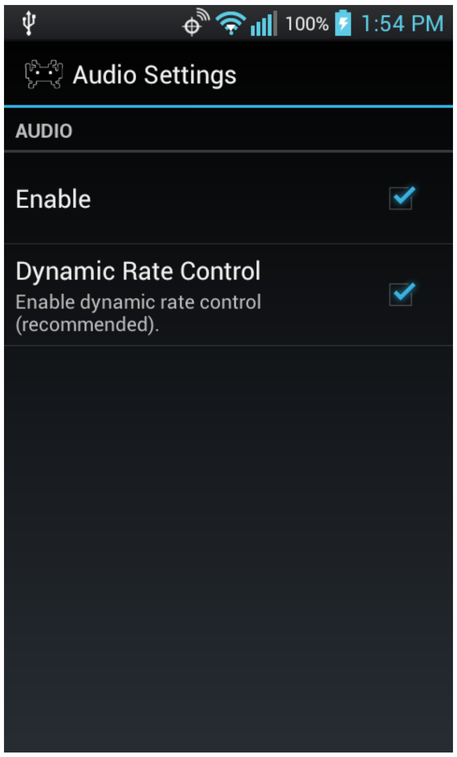
- Shaders (1st pass) Enable
  Enable the currently selected shader.

- Shaders (1st pass) XML Shader
  Select this option to select a shader from the filesystem. RetroArch comes prepackaged with a collection of shaders.

- Enable on-screen fonts
  Enable rendering of on-screen fonts for system messages.

5.3.1 Notes on shaders

- The shaders that come prepackaged with RetroArch Android come from the PS3 and Xbox 360 ports of RetroArch. Unfortunately, most Android GPUs are very weak compared to the ones inside the PS3 and 360 - so most of these shaders will run extemely slow on nearly all Android devices right now. To make these shaders usable we will have to wait until GPUs on Android-powered devices catch up with PS3 and 360. They will make for good GPU benchmarks in the meantime. These shaders are far more intensive on the GPU than the trivial 'shaders' used in commercial games - which are mostly used for menial tasks instead of applying an expensive image-enhancing algorithm to the entire screen like the 'shader filters' seen here.



Audio Settings

AUDIO

Enable ✓

Dynamic Rate Control
Enable dynamic rate control (recommended). ✓

Figure 12: 'Audio Settings' screen.

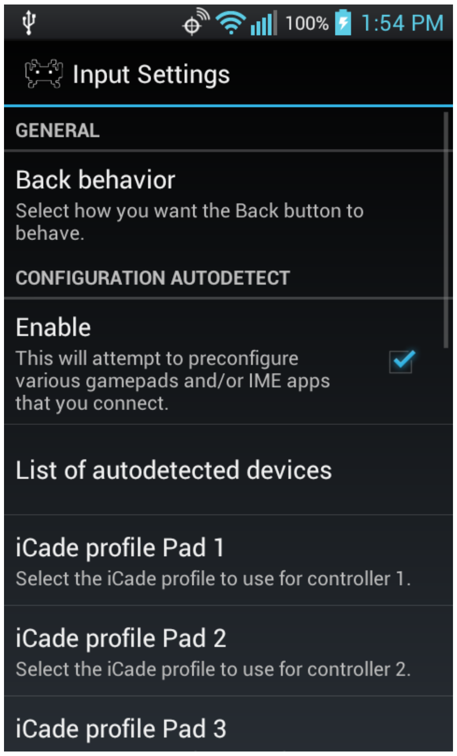
5.3.2 Notes on refresh rate

- Some devices (like the Samsung Galaxy S3) erroneously report that the screen refresh rate is 60Hz. For these devices, it is recommended that you set 'forced refresh rate' manually to a lower rate until you find the right value that gives you good audio/video with no audio pops.

5.4 Audio Settings

- Audio Enable
  Uncheck this to disable sound.

- Dynamic Rate Control
  Dynamic rate control tries to prevent sound pops by dynamically adjusting samplerate. It is recommended that you leave this on for RetroArch Android.



1:54 PM 100%

Input Settings

GENERAL

Back behavior
Select how you want the Back button to behave.

CONFIGURATION AUTODETECT

Enable
This will attempt to preconfigure various gamepads and/or IME apps that you connect. ✓

List of autodetected devices

iCade profile Pad 1
Select the iCade profile to use for controller 1.

iCade profile Pad 2
Select the iCade profile to use for controller 2.

iCade profile Pad 3

Figure 13: 'Input Settings' screen.

5.5 Input Settings

- Back behavior
  Select how you want the Back button to behave. You can either have the Back button behave like a 'Quick Exit' button from RetroArch, or to make it a 'menu toggle' where it will toggle the builtin menu (RGUI) on or off.

- Configuration Autodetect Enable
  This will attempt to preconfigure various gamepads and/or IME apps that you connect.

- List of autodetected devices
  Shows a list of all devices that should be autodetected and autoconfigured.

- iCade profile Pad 1
  Select the iCade profile to use for Player 1. You can choose between several pad configurations.

- iCade profile Pad 2
  Select the iCade profile to use for Player 2. You can choose between several pad configurations.

- iCade profile Pad 3
  Select the iCade profile to use for Player 3. You can choose between several pad configurations.

- iCade profile Pad 4
  Select the iCade profile to use for Player 4. You can choose between several pad configurations.

- Debug Input Reporting Enable
  This will report keycodes onscreen generated by your input device(s). You should use this option when you want us to support a gamepad that you use. You should use this option then to see which keycodes are generated by all the buttons on your gamepad/input device and then report this back to us.

- Touchscreen Overlay Enable
  You can disable the overlay system entirely by disabling this.

- Input overlay
  You can select a different overlay by choosing this option.

- Overlay opacity
  Set the opacity of the touch overlay.

- Custom Binds - Player 1 Custom Binds
  You can set custom keybinds for your gamepad (Player 1) in case it is not automatically detected and configured.

- Custom Binds - Player 2 Custom Binds
  You can set custom keybinds for your gamepad (Player 2) in case it is not automatically detected and configured.
- Custom Binds - Player 3 Custom Binds
  You can set custom keybinds for your gamepad (Player 3) in case it is not automatically detected and configured.
- Custom Binds - Player 4 Custom Binds
  You can set custom keybinds for your gamepad (Player 4) in case it is not automatically detected and configured.

6 RetroArch on other platforms

RetroArch isn't only available for Android. It is available on other platforms as well, including:

- PlayStation3
- Xbox 1
- Xbox 360
- Wii/Gamecube
- Raspberry Pi
- PC (Mac/Linux/Windows)
- iOS
- Blackberry (10/Playbook)

And it will be ported to even more platforms in the future. You might even see the libretro cores running in the official mainline version of XBMC shortly.

7 About Us

Homepage: `http://www.libretro.org`
IRC: #retroarch at freenode
Github (libretro organization): `https://github.com/libretro`
RetroArch @ Github: `https://github.com/Themaister/RetroArch`
Libretro @ Twitter: `https://twitter.com/libretro`
Libretro @ Facebook: `https://www.facebook.com/libretro.retroarch`

8 Troubleshooting

8.1 Audio choppiness

8.1.1 Likely culprit #1: Mismatch in reported refresh rate vs. screen's refresh rate

If you are experiencing audio choppiness in games, the most likely cause is the refresh rate of your Android device's screen. RetroArch uses static synchronization which requires that the RetroArch side is correctly informed about the exact refresh rate of your screen.

There are two methods to 'query' the refresh rate of a screen on Android:

- Through the use of an API function call provided by the Android SDK
- Through 'inferring' the refresh rate by way of some video display test

The problem with the first method is that certain manufacturers 'lie' about the exact refresh rate of the screen. Samsung for instance reports incorrect refresh rates for its Galaxy S3 and Galaxy Note II models. Therefore, if the API function reports '60Hz' but your screen's refresh rate is in actuality 58Hz, then this will lead to a mismatch of refresh rates between the frontend app (RetroArch) and your device's screen. This will lead to stuttering video and/or audio.

For these problematic devices, we created a second method to 'detect' the refresh rate - this is our 'Video synchronization' test. You will be prompted to run this the first time RetroArch is installed, but you can also run the same test later on by going to the 'Video Settings' menu. The refresh rate you get back from this video synchronization test must be regarded as an approximation- it might be off by a little bit. If you find that you are still experiencing audio crackles after running this test, it can help by manually adjusting the new 'refresh rate' by a small amount (either higher or lower) and seeing if that helps.

8.1.2 Likely culprit #2: Dalvik garbage collector cleanup stalls

There is a second and even more serious problem that might lead to stuttering/choppy audio and unfortunately this seems to be an architectural problem. Due to Android being Java-based (and due to the native activity being a basic 'C/C++ glue layer for Java', there is a big chance that the Dalvik garbage collector cleanup routines will run from time to time during the runtime state of RetroArch Android. When this happens, it will create stalls amounting from as low to 2ms to 10ms or more.

Speculation that a second core could 'pick up the slack' has turned out to be unfounded when even quad-core Android phones produce the same audio crackles whenever the garbage collector cleanup is running throughout your game.

This is obviously hazardous for any app that relies on static syncing. What is even worse is that there seems to be no way to 'instruct' Dalvik not to do this and that it can be totally unpredictable if/when this happens. The garbage collector cleanup stalls might be caused by some innocuous weather app service running in the background that does some server queries or it could be caused by a mail check interval by some e-mail app service. Google Play Store services have also been confirmed to have periodic scan intervals which kick the garbage collector cleanup routines into gear, and they can be bad for performance.

The only way I have been able to get around these issues has been to forcibly remove/disable these services that are kicking the garbage collector into gear. Unfortunately, by doing this you would also sacrifice Google Play Store and lots of apps that do this for which there seems to be no way to disable the periodic intervals at which they 'scan'. For most this won't be an acceptable compromise and therefore we are stuck between a rock and a hard place.

One conceivable route would be to rely on frameskipping - which we find to be terrible. For version 0.9.9 we have included a 'frameskip' option in PCSX ReARMed as an experiment. If this produces good results for people we might be tempted to include a similar frameskip option in all cores. We definitely advise that you do NOT set frameskip higher than 0 except if you are experiencing severe performance issues (such as audio crackles) - it will definitely run less smooth and you can easily tell the difference between non-frameskipped gameplay and frameskipped gameplay.

Another 'solution' would be to use threaded video - unfortunately, this 'solution' suffers from the same drawbacks as 'frameskip' in that it is impossible to get 'smooth' vsynced graphics by doing video rendering in this manner.

Issues like these are the big Achilles Heel when it comes to Android - until Google makes a serious attempt at resolving issues like these and gives us more power over the kind of periodic scan intervals that apps are allowed to do (plus more control over Dalvik's GC behavior - plus fix the horrendous audio latency of AudioFlinger) - we have to regretfully conclude that iOS and Blackberry QNX are far superior platforms for RetroArch. Which is also unfortunate since said platforms/ecosystem do not allow us to 'exist' officially whereas Google's ecosystem does.

8.2 Why don't the input overlays show up when I try to select one from the Settings menu?

We have seen this problem manifest itself on a Samsung Galaxy S3 - it appears to be some rights issue to do with accessing the application-specific cache directory. If this happens, please report the bug to us on your specific device and we will put even more effort into getting this issue resolved.

8.3 Shaders make the gameplay unbearably slow

This is to be expected as most shader developers target PS3/360/PC-spec GPUs, and mobile devices are lightyears removed from reaching that stage yet.

The situation might start to change over time when SoCs come out with more powerful GPUs like Tegra 4 and the next-gen ARM Malis. For now, if you find any (if not all) shaders are simply too slow on your device, just don't use them.

8.4 Why can't I find PCSX ReARMed in the cores list?

You possibly have an Android device with a MIPS or x86 processor. PCSX ReARMed is only meant for ARM-based devices.

8.5 I have some other issue that is not addressed here

Please consult RetroArch Android's built-in Help Menu for more information. It attempts to address a lot more possible issues than this PDF manual can hope to do.

9 Philosophy

We do RetroArch Android's distribution this way because:

1. We don't like this 'monetization' drive of these newfangled app stores.
2. We don't like the 'hucksterism' - we don't like the 'MBA meets brogrammer' movement, we don't like the slime, we don't like the cynical 'pretending we are a business providing a service' angle.
3. We don't like the 'rebranding' of apps to make it look like these are 'separate' programs that are somehow 'different' from the apps they are based on. Most of the time they are just direct ports with zero substantial alterations - I won't name names although it would be easy to do so.
4. We don't like lots of 'e-mails' flooding our mailboxes with things like 'Maximize your revenues from your app!' and/or 'do a better job at monetizing your app by downloading our monetization SDK!'. I have a nice backcatalogue of that shit already flooding my mailbox everyday - unwanted. And least of all, we don't want to be a 'part' of this racket. Pardon my French, but screw this entire cottage industry of shit - and everybody in it.
5. We think it's intellectually dishonest to make a big stink about Nintendo/Sony/Microsoft wanting to 'pull' emulators from an app store when you are simultaneously charging users for the privilege of running 'copyrighted' ROMs. Therefore, we don't engage in that intellectually dishonest doublethink altogether. As soon as you sell your wares - you are fair game - everbody and his lawyer knows this, and frankly, everybody who does this will get what is coming to him/her - there is no free lunch.

6. We believe that a future where you have to 'pay' for every little 'app' that is out there and where Google/Apple/Blackberry/Microsoft decide what you can run and where every 'dev' can just push 'ads' to you and 'steal' your personal information is anathema to what computing was supposed to be. Once this was an industry where information was supposed to be 'open' and companies like id Software and others went out of their way to 'share' code (and even cutting-edge information on engines as soon as the game was complete) in the spirit of training future generations and standing on the shoulder of giants. That is what 'science' is supposed to be about - it's not about 'conformity' or 'knowledge hoarding' or 'differential 'advantages'. Now stuff just gets locked up in a vault because of 'differential advantages'. Once patents were made so that after the patent had ran its course, it would be released into the public domain. Fat chance of that happening now when companies can just arbitrarily extend the copyright by an arbitrary amount of years.
   Rather than being part of the problem and going with the catch-all rationalization 'people are paying for convenience' like so many cowards that went before us, we have decided instead to 'not' be part of the problem and to do everything that is the total diametric opposite of what this new age of 'app stores' and 'convenience devices' are doing to software development, homebrew and free and open source software.
   Let it be said - everybody that rationalizes this stuff to themselves by going 'oh well, GPL was never about 'free' as in libre AND 'free' as in beer" is either lying to themselves rationalizing the evil this is bringing forth or is just a plain coward and/or a sell-out opportunist - if the shoe fits, wear it. I don't care how you people rationalize this to yourself - if you don't see what the 'monetization' of apps and this whole 'convenience' era is bringing forth then frankly you never had any wits about you and you are just being used as a useful tool by pragmatic opportunists and companies that are employing their own brand of 'Embrace, Extend and Extinguish' on your FLOSS movement and the products it brings forth. No, bringing up 'RedHat' as a justification doesn't fly either - we have heard all this crap before, and it still doesn't fly. Not to mention that honest devs who release their own code under the GPL license will feel less and less motivation to do so in the future because they don't like to see everbody and his dog running away with ill-gained profits. Rather than help 'build up the GPL', this is actually destroying the GPL's desirability.
   For pragmatic reasons, RetroArch is and will likely always be GPL. But that doesn't mean we don't see what is clearly happening before our eyes and how it has become the 'plaything' of cynical wannabe MBAers pretending to be 'entrepreneurs', and we don't like it one bit. While the FLOSS zealots are going into technicalities about their 'license' and how it allows for all this and how we should all approve of it, cynical moneymen are laughing their arses off at this childish naivety and running up those cash (and donation) registers. GPL software is now regarded as 'easy to steal - easy to make money off - while we contribute nothing of

any significant value to those things and hoard everything we do of any value ourselves'. That is the stark reality of what most 'mobile startups' think to themselves but would never admit in public, whether you like it or not - and it's time to come to grips with that reality and make your own move as to how to proceed - because this clearly isn't working out to your movement's advantage and reeks of a co-opting.

7. We provide all the means to any and all devs to make their own builds/packages from source - unlike others everything we do is immediately committed to Github so whatever you can fetch from source on our Github repositories is exactly the current state of development.
   We know this leaves us wide open to the same kind of 'abuse' as other projects have suffered - RetroArch being GPL alone sees to that. But we hold the overriding belief that by being both 'free' as in libre and 'free' as in beer, we can undercut any and all cynical moneymaking 'copycats' that pop up that have their entire code based on RetroArch. That and 'naming and shaming' should do the trick. It is hard to beat this non-compromising strategy, and they know it - being 'free' of ads and being 'free' itself is a hard winning combination to beat for an opportunist who only cares about raking in money.

10 Credits

RetroArch Android
Hans-Kristian Arntzen (Themaister)
Daniel De Matteis (Squarepusher2/Twinaphex)
Michael Lelli (ToadKing)

RetroArch Android contributions
Meancoot
Opium2k (overlay images)

Thanks to
Notaz (PCSX ReARMed libretro port - RetroArch ARM Linux patches)
FBA Team (for adopting libretro upstream - FBA)
Ekeeke (for adopting libretro upstream - Genesis Plus GX)
CaH4e3 (for adopting libretro upstream - FCEUmm)
Rdanbrook (for adopting libretro upstream - NEStopia Undead)
XBMC devs (for adopting libretro vis a vis RetroPlayer)
Zeromus