Nintendo 64

The Nintendo 64 is a 64-bit fifth-generation console released by Nintendo on June 23, 1996 in Japan; September 29, 1996 in America for $199.99 USD; March 1, 1997 for the rest of the world. One of the last major home consoles to use the cartridge format, unless you count the recent Nintendo Switch as a home console. Probably one of the most complicated pieces of hardware ever devised, utilizing a combination of 64-bit (hence its name) and 32-bit components. Here's a fun article written by Rodrigo Copetti about it.

Emulation for Nintendo 64 is still experimental, and accuracy issues crop up no matter which hardware emulator you use. Be prepared to switch emulator/core on a per-game basis. If you'd like to watch a video explanation of what's explained on this page, check out Batocera Nation's excellent video on how to setup Nintendo 64 in Batocera.

This system scrapes metadata for the “n64” group(s) and loads the n64 set from the currently selected theme, if available.

Quick reference

- **Accepted ROM formats:** `.z64 .n64 .v64 .zip .7z`
- **Folders:** `/userdata/roms/n64 /userdata/roms/n64dd`

Emulators
libretro: Mupen64Plus-Next
libretro: ParaLLel_N64
Mupen64Plus: GLideN64
Mupen64Plus: GLide64mk2
Mupen64Plus: Rice

BIOS

The N64 emulators don't need a BIOS to function.

However, if intending on playing N64DD games specifically, the following BIOS file is required:

MD5 checksum	Share file path	Description
Fix Me!	`bios/Mupen64plus/IPL.n64`	N64 Dynamic Disk Initial Program Loader

ROMs

Place your Nintendo 64 and N64DD ROMs in `/userdata/roms/n64/`.

Mupen64Plus has some trouble loading compressed `.zip` files. If you cannot load your ROMs, try unzipping them.

File extensions may not necessarily reflect their true contents, but here are the generally accepted formats:

Extension	Format
`.bin`	Binary representation of data on the ROM.
`.n64`	ROM dump stored in little-endian byte-order.
`.N64`	ROM dump stored in big-endian byte-order.
`.v64`	ROM dumps produced by or compatible with the Doctor V64 CD-ROM drive, with data stored in a byte-swapped version of the N64's native byte-order.
`.z64`	ROM dumps produced by or compatible with the Z64, stored in big-endian byte-order.
`.z64.ndd`	Dynamic Disk ROM dumps produced by or compatible with the Z64, stored in big-endian byte-order.

There is no practical difference as to which format to use for loading the ROMs in Batocera.

Emulators

RetroArch

RetroArch (formerly SSNES), is a ubiquitous frontend that can run multiple “cores”, which are essentially the emulators themselves. The most common cores use the libretro API, so that's why cores run in RetroArch in Batocera are referred to as “libretro: (core name)”. RetroArch aims to unify the feature set of all libretro cores and offer a universal, familiar interface independent of platform.

RetroArch configuration

RetroArch offers a **Quick Menu** accessed by pressing `[HOTKEY]` +  which can be used to alter various things like RetroArch and core options, and controller mapping. Most RetroArch related settings can be altered from Batocera's EmulationStation.

Standardized features available to all libretro cores: `n64.videomode`, `n64.ratio`, `n64.smooth`, `n64.shaders`, `n64.pixel_perfect`, `n64.decoration`, `n64.game_translation`

ES setting name `batocera.conf_key`	**Description ⇒ ES option `key_value`**
Settings that apply to all cores of this emulator	
GRAPHICS BACKEND `n64.gfxbackend`	Choose your graphics rendering ⇒ OpenGL `opengl`, Vulkan `vulkan`.
AUDIO LATENCY `n64.audio_latency`	Audio latency in milliseconds, turn it up if you hear crackles ⇒ 256 256, 192 192, 128 128, 64 64, 32 32, 16 16, 8 8.

ES setting name `batocera.conf_key`	Description ⇒ ES option `key_value`
THREADED VIDEO `n64.video_threaded`	Improves performance at the cost of latency and more video stuttering. Use only if full speed cannot be obtained otherwise. ⇒ On `true`, Off `false`.

libretro: Mupen64Plus-Next

Mupen64Plus-Next is a N64 emulation library for the libretro API, based on Mupen64Plus. It is also the successor of the old Mupen64Plus libretro core. It's still experimental, but incorporates Retroarch's features, such as input remapping from its Quick Menu (`[HOTKEY]` + ).

This is the only emulator in Batocera capable of playing N64DD games.

libretro: Mupen64Plus-Next Configuration

ES setting name `batocera.conf_key`	Description ⇒ ES option `key_value`
Settings that apply to all systems this core supports	
VIDEO 4:3 RESOLUTION `global.mupen64plus-43screensize`	The rendering resolution when using a 4:3 aspect ratio. If set to auto, will match your current video mode. Severe performance impact. `320×240` for native, `640×480` for “Hi-Res” mode (eg. Perfect Dark). ⇒ `320×240` `320×240`, `640×480` `640×480`, `960×720` `960×720`, `1280×960` `1280×960`, `1600×1200` `1600×1200`, `1920×1440` `1920×1440`, `2240×1680` `2240×1680`, `2560×1920` `2560×1920`, `3200×2400` `3200×2400`, `3520×2640` `3520×2640`, `3840×2880` `3840×2880`.
VIDEO 16:9 RESOLUTION `global.mupen64plus-169screensize`	The rendering resolution when using a 16:9 aspect ratio. If set to auto, will match your current video mode. Severe performance impact. `320×240` for native, `640×480` for “Hi-Res” mode (eg. Perfect Dark). ⇒ `640×360` `640×360`, `960×540` `960×540`, `1280×720` `1280×720`, `1920×1080` `1920×1080`, `2560×1440` `2560×1440`, `3840×2160` `3840×2160`, `7680×4320` `7680×4320`.
WIDESCREEN HACK `global.mupen64plus-aspect`	Enhancement. Widescreen hack. Very glitchy. Only works when using a 16:9 video mode **and** with bezels (decorations) disabled. Recommended to use ROM patches instead. Some games natively support 16:9 in their in-game options. ⇒ Off 243, On `16:9 adjusted`.

ES setting name `batocera.conf_key`	Description ⇒ ES option `key_value`
BILINEAR FILTERING `global.mupen64plus-BilinearMode`	Enhancement. Chooses which bilinear filtering method is used to smooth textures on 3D objects. `3point` is accurate to the unique 3-point bilinear filtering the N64 employs as a cost-saving measure, however this introduces some distortion to the resulting textures. `standard` uses a traditional 4-point bilinear filter for more accurate textures. Negligible performance cost. ⇒ standard `standard`, 3 point `3point`.
ANTI-ALIASING (MSAA) `global.mupen64plus-MultiSampling`	Enhancement. Applies MSAA to smooth out jagged edges on 3D object polygons. Most noticeable at lower resolutions. Does not affect textures. The N64 has no MSAA natively, but this was masked by its blurry video output. Significant performance cost. 4x for powerful enough hardware. Higher values give diminishing returns at an exponential performance cost. Should only be disabled if running at native resolution for a more authentic experience. ⇒ Off `0`, 2x 2, 4x 4, 8x 8, 16x 16.
TEXTURE FILTERING `global.mupen64plus-txFilterMode`	Enables mip-mapping to smooth out textures on distant 3D objects based on distance and angle. Has a minimal performance impact. The “Smooth” settings don't have anisotropic filtering enabled, resulting in very blurry distant textures. The “Sharp” settings enable anisotropic filtering, dramatically improving the clarity of distant textures at the cost of making UI elements blurrier. Has a minimal performance cost. Very subjective, some N64 games actually did use mip-mapping and require it for certain graphical effects (eg. the Peach painting in Super Mario 64) while others completely omitted it, resulting in aliasing textures for distant objects. ⇒ Off `None`, Smooth 1 `Smooth filtering 1`, Smooth 2 `Smooth filtering 2`, Smooth 3 `Smooth filtering 3`, Smooth 4 `Smooth filtering 4`, Sharp 1 `Sharp filtering 1`, Sharp 2 `Sharp filtering 2`.
TEXTURE ENHANCEMENT (XBRZ) `global.mupen64plus-txEnhancementMode`	Enhancement. Uses upscalers/resamplers on textures to improve their clarity. Improvements are subjective. ⇒ Off `None`, As Is `As Is`, X2 X2, X2SAI X2SAI, HQ2X HQ2X, HQ2XS HQ2XS, LQ2X LQ2X, LQ2XS LQ2XS, HQ4X HQ4X, 2xBRZ 2xBRZ, 3xBRZ 3xBRZ, 4xBRZ 4xBRZ, 5xBRZ 5xBRZ, 6xBRZ 6xBRZ.
CONTROLLER PAK 1 `global.mupen64plus-pak1`	The accessory pack plugged into controller 1. ⇒ Off `none`, memory `memory`, rumble `rumble`.

ES setting name batocera.conf_key	Description ⇒ ES option key_value
CONTROLLER PAK 2 global.mupen64plus-pak2	Same as above for the second controller. ⇒ Off none, memory memory, rumble rumble.
CONTROLLER PAK 3 global.mupen64plus-pak3	Same as above for the third controller. ⇒ Off none, memory memory, rumble rumble.
CONTROLLER PAK 4 global.mupen64plus-pak4	Same as above for the fourth controller. ⇒ Off none, memory memory, rumble rumble.

Other configuration settings must be configured from RetroArch's **Quick Menu** ([HOTKEY] + ).

libretro: ParaLLel_N64

A low level emulator that incorporates the Angrylion RDP renderer and Vulkan APIs for hardware accelerated emulation. It is more accurate but lacks some enhancement options.

libretro: ParaLLel_N64 configuration

ES setting name batocera.conf_key	Description ⇒ ES option key_value
Settings that apply to all systems this core supports	
VIDEO RESOLUTION global.parallel-n64-screensize	Improve the fidelity of 3D models (does not affect 2D sprites). Severe performance impact. 320×240 for native, 640×480 for “Hi-Res” mode (eg. Perfect Dark). ⇒ 320×240 320×240, 640×480 640×480, 960×720 960×720, 1280×960 1280×960, 1440×1080 1440×1080, 1600×1200 1600×1200, 1920×1440 1920×1440, 2240×1680 2240×1680, 2880×2160 2880×2160, 5760×4320 5760×4320.
WIDESCREEN global.parallel-n64-aspectratiohint	Break aspect ratio and stretch the image. ⇒ Off normal, On widescreen.
TEXTURE FILTERING global.parallel-n64-filtering	Enhancement. Chooses which bilinear filtering method is used to smooth textures on 3D objects. N64 3-point is accurate to the unique 3-point bilinear filtering the N64 employs as a cost-saving measure, however this introduces some distortion to the resulting textures. Bilinear uses a traditional 4-point bilinear filter for more accurate textures. Negligible performance cost. ⇒ Automatic automatic, N64 3-point N64 3-point, Bilinear bilinear, Nearest nearest.

ES setting name batocera.conf_key	Description ⇒ ES option key_value
FRAMERATE global.parallel-n64-framerate	Fullspeed can break some games. ⇒ fullspeed `fullspeed`, original `original`.
CONTROLLER PAK 1 global.parallel-n64-pak1	The accessory pack plugged into controller 1. ⇒ Off `none`, memory `memory`, rumble `rumble`.
CONTROLLER PAK 2 global.parallel-n64-pak2	Same as above for the second controller. ⇒ Off `none`, memory `memory`, rumble `rumble`.
CONTROLLER PAK 3 global.parallel-n64-pak3	Same as above for the third controller. ⇒ Off `none`, memory `memory`, rumble `rumble`.
CONTROLLER PAK 4 global.parallel-n64-pak4	Same as above for the fourth controller. ⇒ Off `none`, memory `memory`, rumble `rumble`.

Other configuration settings must be configured from RetroArch's **Quick Menu** (`[HOTKEY]` + ).

Mupen64Plus

A standalone N64 emulator. Can utilize multiple plugins for various aspects of its emulation. For Batocera's purposes, each “core” can also be a different video plugin of the same emulator.

- “Mupen64Plus: GLide64mk2” for the enhanced **GLide64mk2** video plugin.
- “Mupen64Plus: GLideN64” for the **GLideN64** video plugin. Allows the use of hi-res texture packs.
- “Mupen64Plus: Rice” for the **Rice** video plugin. Allows use of hi-res texture packs.

Mupen64Plus configuration

Standardized features available to all video plugins of this emulator: `n64.videomode`, `n64.ratio`

ES setting name batocera.conf_key	Description ⇒ ES option key_value
Settings that apply to most video plugins of this emulator (exceptions noted)	
FRAME SKIP [GL64MK2 ONLY] n64.mupen64plus_frameskip	Skip frames to improve performance (smoothness). ⇒ Off `0`, Autodetect `automatic`, 1 1, 2 2, 3 3, 4 4, 5 5.

ES setting name `batocera.conf_key`	Description ⇒ ES option `key_value`
TEXTURE MIP-MAPPING (BLUR) **`n64.mupen64plus_Mipmapping`**	Enhancement. Enables mip-mapping to smooth out textures on distant 3D objects based on distance and angle. Should be used in conjunction with `n64.mupen64plus_Anisotropic`. Has a minimal performance cost. Refer to this page for details. Generally advised to leave this on (some N64 games actually used mip-mapping for graphical effects eg. the Peach painting in Super Mario 64), but some users may prefer the 'hardness' of distant objects with it off, especially at increased rendering resolutions. ⇒ Off `0`, nearest 1, bilinear 2, trilinear 3.
ANISOTROPIC FILTERING **`n64.mupen64plus_Anisotropic`**	Enables anisotropic filtering to enhance perspective textures. Dramatically improves the clarity of distant objects when mip-mapping is turned on. Has a small performance impact. Refer to this page for details. Recommended to set it as high as you can go. If not using mip-mapping, it is recommended to leave this off as it can make UI elements unnecessarily blurry. ⇒ Off `0`, 2x 2, 4x 4, 8x 8, 16x 16.
ANTI-ALIASING (MSAA) **`n64.mupen64plus_AntiAliasing`**	Enhancement. Applies MSAA to smooth out jagged edges on 3D object polygons. Most noticeable at lower resolutions. Does not affect textures. Has a significant performance impact. 4x for powerful enough hardware. Higher values give diminishing returns at an exponential performance cost. Should only be disabled if running at native resolution for a more authentic experience. ⇒ Off `0`, 2x 2, 4x 4, 8x 8, 16x 16.
TEXTURE ENHANCEMENT (XBRZ) [RICE ONLY] **`n64.mupen64plus_TextureEnhancement`**	Enhancement. Uses upscalers/resamplers on textures to improve their clarity. Improvements are subjective. ⇒ Off `0`, 2X 1, 2XSAI 2, HQ2X 3, LQ2X 4, HQ4X 5, Sharpen 6, Sharpen More 7, External 8, Mirrored 9.
HIRES TEXTURES **`n64.mupen64plus_LoadHiResTextures`**	Uses the hi-res texture pack for the game, if any. Only confirmed for the Rice and GLideN64 video plugins. No harm leaving on `True` if there aren't any files present anyway. ⇒ Off `False`, On `True`.
DISABLE 4MB EXPANSION RAM PACK **`n64.mupen64plus_DisableExtraMem`**	Disables the expansion pack required for some games. Polarity for this setting is reversed for EmulationStation's user-facing setting. Emulation of this can cause issues in games that don't require it, so it is suggested to leave disabled unless needed. ⇒ Off `False`, On `True`.

You can use switch the connected virtual controller accessory to a Mempak with L3 and a Rumblepak with R3 while in game!

Mupen64Plus standalone cannot ordinarily be configured by altering its CFG file in userdata, as that gets overwritten every launch. If you're feeling adventurous, you can edit the

`/usr/share/batocera/configgen/configgen-defaults-arch.yml` and utilize Batocera's overlay feature to keep the settings. Support cannot be provided should you do this, however.

Hi-res texture packs

Hi-res texture packs replace the textures in a game with other (typically higher resolution) textures. Support for this has been baked into Mupen64 ever since textures could be extracted from the game itself and naturally Batocera supports their use. The “MUPEN64PLUS / GLIDE64MK2”, “MUPEN64PLUS / RICE” and Fix Me! “LIBRETRO / MUPEN64PLUS-NEXT” emulators support this feature.

This feature was buggy in Batocera **v31**, but fixed in Batocera **v32**.



The author of the texture pack will usually include instructions on how to install them on standalone Mupen64Plus, but in case they don't here are the general steps:

1. Grab your texture pack and put it in `/userdata/system/.local/share/mupen64plus/hires_texture/` (for .htc that would be `…/mupen64plus/cache/` instead).

   Bug: For some reason this path is actually `/userdata/bios/mupen64plus/cache/` instead.

2. Go to **ADVANCED GAME OPTIONS** in EmulationStation and enable **HIRES TEXTURES**. This something you can enable on a game per game basis, when you have the corresponding texture pack.
3. HD bing bing wahoos!

Texture packs can be very heavy (several hundreds of MB) and have an impact on the emulation performance if your system is on the lower side of the spectrum.

Raw PNG “Loose” packs are supported by the Rice and GLide64mk2 video plugins, whereas the `.htc` file is only supported by GLide64mk2! `.dat` formats are depreciated and not compatible with any available emulator in Batocera.

Some good places to start searching for N64 texture replacements:

- Batocera Nation's download page (scroll down to "HD Texture Packs for Nintendo 64"). These packs are already in a ready-to-go format.
- The "Completed Projects" page on the Emutalk forum. Most packs include installation instructions written by the author. Do note that this comes from the perspective of using the standalone emulators, and not all texture packs may support their libretro equivalents.
- Emulation King's N64 section (scroll down to "N64 Texture Packs") is another good, organised resource, however it usually excludes the author's installation instructions on the website itself. A lot of texture packs do feature readmes on how to use them, though.
- You may find some on the romhacking.net forum, however it's not organized per platform.
- The Texture Packs subreddit is a good place to discuss and share texture packs.
- The Game Upscale subreddit is similar but focuses more on upscaling the existing content with sophisticated AI instead of redrawing it or doing it in a different style. You can browse for just the "finished" projects by using its tag.

You might also want to look into 60 FPS patches/codes for your game while you're at it. Batocera cannot provide support for such hacks if something goes wrong, though, so the first step you should do if something goes wrong is to try it without the patch/code!

N64 Disc Drive

Better known as the N64DD, this was an expansion available for the original N64 that would allow for the loading from discs instead of cartridges.

Largely a commercial failure, never left Japan. But the few titles that were developed and released for it were incredible.

To replay these gems:

1. Place your N64DD cartridge ROMs in `/userdata/roms/n64dd`. They should have the extension `.z64`.
2. Place your N64DD disk ROMs in the same `/userdata/roms/n64dd`. They should have the extension `.z64.dd`.

3. While hovering over the game in the gamelist, press [SELECT] and set the emulator to libretro: Mupen64Plus-Next.

Emulators/configuration are the same as the regular N64 emulators.

Controls

Batocera requires a controller with a (real or not) right analogue stick if the C-buttons are needed. If configured via the UI in Batocera, they are automatically bound to the appropriate positive/negative values of the right stick axis.

Batocera infers right-stick down and right-stick right as the opposite axis values to the ones you assign in the controller configuration screen. If you've assigned buttons to these controls instead, you'll only be able to use the buttons you have assigned (no C-down or C-left)!

If using a controller that doesn't have a traditional right analogue stick (such as a USB N64 controller), you can work around this limitation by editing the `/userdata/system/configs/emulationstation/es_input.cfg` manually. Add the following lines to your `es_input.cfg` file to have Batocera recognize them:

Manually configured controller configs

- RetroFighter's Brawler64

es_input.cfg

```
<inputConfig type="joystick" deviceName="Brawler64 USB Gamepad"
deviceGUID="030000000d0f0000c100000011010000">
    <input name="a" type="button" id="10" value="1" code="314" />
    <input name="b" type="button" id="2" value="1" code="306" />
    <input name="down" type="hat" id="0" value="4" />
    <input name="hotkey" type="button" id="8" value="1" code="312"
/>
    <input name="joystick1left" type="axis" id="0" value="-1"
code="0" />
    <input name="joystick1up" type="axis" id="1" value="-1"
code="1" />
    <input name="joystick2down" type="button" id="3" value="1"
code="307" />
    <input name="joystick2left" type="button" id="0" value="1"
code="304" />
    <input name="l2" type="button" id="6" value="1" code="310" />
    <input name="left" type="hat" id="0" value="8" />
    <input name="pagedown" type="button" id="5" value="1"
code="309" />
```

```
    <input name="pageup" type="button" id="4" value="1" code="308"
/>
    <input name="right" type="hat" id="0" value="2" />
    <input name="select" type="button" id="8" value="1" code="312"
/>
    <input name="start" type="button" id="9" value="1" code="313"
/>
    <input name="up" type="hat" id="0" value="1" />
    <input name="x" type="button" id="11" value="1" code="315" />
    <input name="y" type="button" id="1" value="1" code="305" />
</inputConfig>
```

- Retro-bit Tribute64 Controller - USB® Port (X-input mode)

es_input.cfg

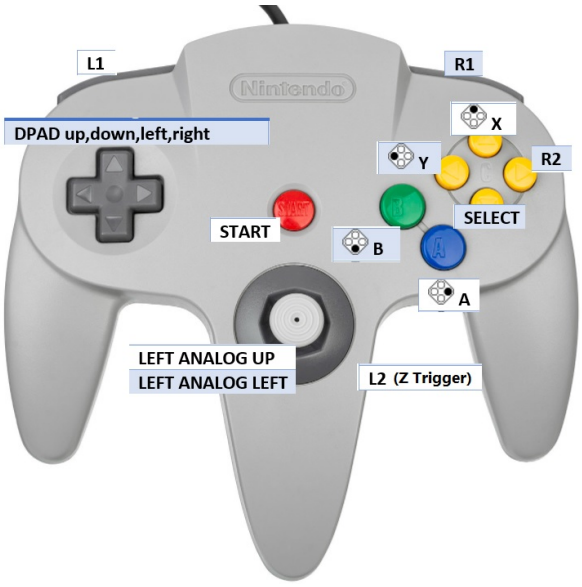
```
<inputConfig type="joystick" deviceName="Microsoft X-Box 360 pad"
deviceGUID="030000005e0400008e02000072050000">
    <input name="a" type="button" id="9" value="1" code="317" />
    <input name="b" type="button" id="0" value="1" code="304" />
    <input name="down" type="hat" id="0" value="4" />
    <input name="hotkey" type="button" id="6" value="1" code="314"
/>
    <input name="joystick1left" type="axis" id="0" value="-1"
code="0" />
    <input name="joystick1up" type="axis" id="1" value="-1"
code="1" />
    <input name="joystick2left" type="button" id="3" value="1"
code="308" />
    <input name="joystick2down" type="button" id="2" value="1"
code="307" />
    <input name="l2" type="axis" id="2" value="1" code="2" />
    <input name="left" type="hat" id="0" value="8" />
    <input name="pagedown" type="button" id="5" value="1"
code="311" />
    <input name="pageup" type="button" id="4" value="1" code="310"
/>
    <input name="r2" type="axis" id="5" value="1" code="5" />
    <input name="right" type="hat" id="0" value="2" />
    <input name="select" type="button" id="6" value="1" code="314"
/>
    <input name="start" type="button" id="7" value="1" code="315"
/>
    <input name="up" type="hat" id="0" value="1" />
    <input name="x" type="button" id="10" value="1" code="318" />
    <input name="y" type="button" id="1" value="1" code="305" />
</inputConfig>
```

- The official Nintendo Switch's N64 USB controller.

es_input.cfg

```
<inputConfig type="joystick" deviceName="N64 Controller"
deviceGUID="050000007e0500001920000001000000">
        <input name="a" type="button" id="1" value="1" code="305"
/>
    <input name="b" type="button" id="0" value="1" code="304" />
    <input name="down" type="hat" id="0" value="4" />
    <input name="hotkey" type="button" id="10" value="1"
code="314" />
    <input name="joystick1left" type="axis" id="0" value="-1"
code="0" />
    <input name="joystick1up" type="axis" id="1" value="-1"
code="1" />
    <input name="l2" type="button" id="7" value="1" code="311" />
    <input name="left" type="hat" id="0" value="8" />
    <input name="pagedown" type="button" id="5" value="1"
code="309" />
    <input name="pageup" type="button" id="4" value="1" code="308"
/>
    <input name="r2" type="button" id="8" value="1" code="312" />
    <input name="right" type="hat" id="0" value="2" />
    <input name="select" type="button" id="6" value="1" code="310"
/>
    <input name="start" type="button" id="9" value="1" code="313"
/>
    <input name="up" type="hat" id="0" value="1" />
    <input name="x" type="button" id="2" value="1" code="306" />
    <input name="y" type="button" id="3" value="1" code="307" />
</inputConfig>
```

Then simply remap the controls as necessary in RetroArch's Quick Menu:



Batocera Configure controller	Remap in Retroarch Mupen64plus_next, Parallel
DPAD up,down,left,right	AS IS
A	Remap > Button A
B	Remap > Button B
X	Remap > C Buttons Y-
Y	Remap > C Buttons X-
LEFT ANALOG UP	AS IS
LEFT ANALOG LEFT	AS IS
START	AS IS
SELECT	Remap > C Buttons Y+
L1	AS IS
R1	AS IS
L2 (Z Trigger)	AS IS
R2	Remap > C Buttons X+

Beware, if doing this manual remap Batocera will use C-right as the “Back” button.

Here are the default Nintendo 64's controls shown on a Batocera Retropad for libretro cores:



Here are the default Nintendo 64's controls shown on a Batocera Retropad for Mupen64Plus:



Remapping for standalone Mupen64Plus

If instead you'd like to change the way that Mupen64Plus handles the inputs sent from ES/Batocera, you can change *its* remapping configuration.

Unlike with the libretro core, remaps for standalone Mupen64Plus can be configured by editing `userdata/system/configs/mupen64/input.xml`. Back up this file before making edits to it. When you open it, you'll see the bindings like so:

```
<inputList>
<input name="AnalogDeadzone" value="0,0" />
<input name="AnalogPeak"     value="32768,32768" />
<input name="l3"             value="Mempak switch" />
<input name="r3"             value="Rumblepak switch" />
<input name="a"               value="C Button R" />
<input name="b"               value="A Button" />
<input name="x"               value="C Button U" />
...
```

The input names on the left are the virtual Retropad Batocera uses to convert your controller inputs. You don't typically need to edit these.

The values on the right are the controller signal sent to the game. Feel free to switch these around with each other.

Mupen64 can only accept one name per value; if multiple are detected it will only use the first occurrence. For example, if you bind the n64 Z value to both L2 and R2 Retropad names, only the L2 button will activate the Z value.

You can assign various Mupen64 commands here like `Mempack switch`!

Troubleshooting

2D graphics are appearing with gaps/lines in them

Turn off any upscaling settings such as increasing the rendering resolution. By default, Batocera will not be using any upscaling settings.

The RetroArch Quick Menu is graphically bugged

This issue affects Batocera **v35** on RK3326-based images such as the Odroid Go Advance.

If the Quick Menu has missing text and RetroArch notifications look like multi-colored boxes like this:



Then try going to **ADVANCED SYSTEM OPTIONS** → **GRAPHICS API** and set it to “OPENGL”.

Further troubleshooting

For further troubleshooting, refer to the generic support pages.

From:
https://wiki.batocera.org/ - **Batocera.linux - Wiki**

Permanent link:
https://wiki.batocera.org/systems:n64?rev=1660875158

Last update: **2022/08/19 02:12**