Disc image compression

Systems from the 5th and 6th console generation often use optical discs like CD or DVD to store game data. Disc image sizes range from 700 MB (CD), 1.4 GB (GC Mini-DVD), 4.7 GB (single-layered DVD) and even 25 GB (Blu-Ray) for newer systems. Often times only a fraction of the actual disc size is used for game data and the rest is filled with dummy data to be optimized for read access by the optical drive laser.

The Super Mario 25th Anniversary Wii disc itself is a 4.7GB, when really the actual game data is only a single SNES ROM (12 MB of useful data). Disc compression helps to save more than 99% of hard disk space and can speed up loading time on emulators.

On the other side systems like the Sony PlayStation often use multi-track images (data and audio tracks) containing several `*.bin` and `*.cue` files which can become cumbersome in daily usage with Batocera or other emulation front-ends.

Choosing the right disc image compression can help in both cases and should be the preferred format for most of the disc based systems. This page will help you to find the suitable compression method and tool set.

If not familiar with using command line tools there are also GUI-based helper tools available:

- **Windows:** namDHC
- **Linux:** game-converter

CHD

CHD is short for *(C)ompressed (H)unks of (D)ata*. It is a lossless compression format originally developed for MAME, for the hard-drive contents of certain arcade machines. It has since been used in several other emulators as a means of storing disc based game data. It compresses the contents of a disc image to a handy single `.chd` file. Here's a comparison between 7z file compression and CHD (originally an 700 MB image):

Supported systems

CHD is supported by many systems and the recommended format for the following systems:

- 3do
- dreamcast
- segacd
- neogeocd
- pcenginecd
- psx
- ps2 (available with Batocera v31)
- saturn

If you have one of the PAL PlayStation games that features LibCrypt copy protection, you have a `.sbi` file in addition to the `.bin`/`.cue` file. The CHD creation process doesn't process the `.sbi` file. Therefore you will need the `.sbi` file in the same directory as the `.chd` file for the game to run.

Installing chdman

CHD files can be created using the `chdman` program, developed by the MAME project. Its installation process varies based on which operating system is being used to create the files.

Batocera

`chdman` is already included with Batocera. It can only be summoned by its command line interface though. Open up a terminal and run `/usr/bin/mame/chdman` to use it. For instructions further on in this document, just replace `chdman` with `/usr/bin/mame/chdman`.

Windows

The official MAME package contains a copy of the executable file. You can download the EXE file and open it up like an archive with 7-zip, then extract `chdman.exe`.

Linux

On most Linux based distributions, this is included with the `mame-tools` package.

MacOS

On macOS use brew to install the `rom-tools` package.

Creating CHDs

chdman is a commmand line application. It can create a .chd file by targeting *.cue, *.iso, .nrg or *.gdi files.

If there is a *.cue file available, use this for creating the .chd. Otherwise choose the *.iso (Sony PlayStation 2 DVDs) or *.gdi (Dreamcast GD-ROM) file. Note that GameCube/Wii have their own compressed format.

Open the command line in either Windows with cmd.exe or in Linux with your favorite terminal emulator. Put both chdman and the game file in the same folder, navigate to that folder in your terminal and run:

```
chdman createcd -i <game.cue> -o <game.chd>
```

For example, if the file structure was like so:

```
/totally-rad-folder-name/
  ├─ chdman
  ├─ Spiderman.cue
  └─ Spiderman.bin
```

Switch to the folder first with cd /totally-rad-folder-name then run:

```
chdman createcd -i Spiderman.cue -o Spiderman.chd
```

Then make yourself a cup of tea. This can take a while. After that you'll have <game.chd> and you can delete the *.cue/*.bin, *.iso, .nrg or *.gdi/*.bin file(s).

The compression is fully lossless so it is possible to redo the compression and get the original files back:

```
chdman extractcd -i <game.chd> -o <game.cue>
```

CHD batch converting

You can batch convert all files in a directory like following (replace .cue with the correct file type you want to convert):

Linux/macOS:

Navigate to the folder that has all your game ROMs and run the following for loop to convert all the files in one go:

cue-to-chd.sh

```
#!/bin/bash
for i in *.cue;
do chdman createcd -i "${i}" -o "${i%.*}.chd";
done
```

replacing `*.cue` as needed. If you're converting a lot of files, practice some yoga poses. This may take a while.

Here's a command to convert all the games in the current folder and recursive folders to CHD:

For `*.cue`:

```
find . -name "*cue" -exec chdman createcd -i {} -o {}.chd \;
```

For `*.iso`:

```
find . -name "*iso" -exec chdman createcd -i {} -o {}.chd \;
```

For `*.gdi`:

```
find . -name "*gdi" -exec chdman createcd -i {} -o {}.chd \;
```

For `*.toc`:

```
find . -name "*toc" -exec chdman createcd -i {} -o {}.chd \;
```

Note: find command is recursive, so it will convert files from subfolders.

Windows:

Just place the `chdman.exe` into the directory where you have your disc images.

Here's some handy BAT scripts you can just double-click to instantly convert all the games in the current folder to CHD:

For `*.cue`:

cue-to-chd.bat

```
for /r %%i in (*.cue) do chdman createcd -i "%%i" -o "%%~ni.chd"
```

For `*.iso`:

iso-to-chd.bat

```
for /r %%i in (*.iso) do chdman createcd -i "%%i" -o "%%~ni.chd"
```

For `*.gdi`:

gdi-to-chd.bat

```
for /r %%i in (*.gdi) do chdman createcd -i "%%i" -o "%%~ni.chd"
```

For `*.toc`:

gdi-to-chd.bat

```
for /r %%i in (*.toc) do chdman createcd -i "%%i" -o "%%~ni.chd"
```

Or combine both: for /r i in (*.cue, *.gdi, *.iso, *.toc) do chdman createcd -i "i“ -o ”%%~ni.chd“

If you're converting a lot of files, play some tunes on your guitar. This may take a while.

CDI / NRG / IMG TO CHD

chdman 0.138b and below support converting `.cdi` files to CHD as well. It requires the additional cdifile executable to be in the same directory as it.

Newer versions of chdman (0139+) are not compatible with cdifile.

Put the files `chdman.exe` and `cdifile.exe`in the same directory

And run the code line command:

For Windows:

```
cdifile.exe game.cdi -createcd
cdifile.exe game.nrg -createcd
cdifile.exe game.cue -createcd
cdifile.exe game.img -createcd
```

These commands can also be run from WINE in any Linux distro.

It's a good idea to test the game is still working after the conversion before deleting anything.

CSO

CSO is a compression method for the ISO image format and is the short form for *Compressed ISO*. It also called CISO.

It was originally used to compress PlayStation Portable UMD disc images.

Supported systems

CSO/CISO is supported by multiple systems and the recommended format for the following systems:

- ps2 (prior to Batocera v31)
- psp

The Wii/Gamecube also support a CSO/CISO format, that is unrelated to the compression method explained here. More information can be found at the wit: Wiimms ISO Tool.

Creating CSOs

With MaxCSO

If not familiar with using command line tools, a GUI-based helper tool is available:

- **Windows:** https://github.com/sethfoxen/maxcsoGUI/releases (requires the maxcso executable to be copied to the same folder to function)

There are different tools to create CSO files. maxcso is a free command line application that is available to Linux, macOS and Windows.

To compress an ISO file just use:

```
maxcso <game.iso>
```

This will create a `<game.cso>` file in the same folder.

The compression is fully lossless so it is possible to redo the compression and get the original files back:

```
maxcso --decompress <game.cso>
```

MaxCSO batch converting

Batch converting can be done with a simple `for` loop.

Linux/macOS:

iso-to-cso.sh

```
#!/bin/bash
for i in *.iso;
```

```
do maxcso "${i}";
done
```

Windows:

iso-to-cso.bat

```
for %i in (*.iso) do maxcso.exe "%i"
```

With ciso

If not familiar with using command line tools, a GUI-based helper tool is available:

- **Windows:** https://sites.google.com/site/theleecherman/cisomulticompressor
- **Linux:** Actually there is none, but you can run cisomulticompressor in Wine with the necessary dependencies in a Debian-based distro:

```
sudo apt update
sudo apt install mono-complete libmono-microsoft*
sudo apt install wine1.4
```

ciso is a command line utility for converting PSP iso files to cso. It is notable for being more compatible with a real PSP than MaxCSO.

Compress an ISO to CSO:

```
ciso <level> game.iso game.cso
```

Where `<level>` is the compression level, from 1 (fast, poor compression) to 9 (slow, high compression).

To decompress a CSO to ISO:

```
ciso 0 game.cso game.iso
```

Here, the “level” is 0, that is “no compression”.

ciso batch converting

Batch converting can be done with a simple `for` loop.

Linux/MacOS:

iso-to-ciso.sh

```
#!/bin/bash
for i in *.iso;
do ciso 9 "${i}" "${i}".ciso;
done
```

Windows:

iso-to-ciso.sh

```
for %i in (*.iso) do ciso.exe 9 "%i" "%i".ciso
```

PBP

PBP supports multiple discs and is compatible with .BIN+.CUE, .IMG image formats.

Good to compress PSX multi-discs into one unique PBP file.

The compression is NOT fully lossless, it is possible to redo the decompression of a .PBP file and get the .CUE+.BIN files, but they are not the same originals used before but a complete new file!

Supported systems

PBP is supported by multiple systems:

- psp
- psx

Creating PBPs

With PSX2PSP

- **Windows:** https://psp.brewology.com/downloads/download.php?id=9697
- **Linux:** Actually there is none, but you can run PSX2PSP in Wine or Lutris

PSX2PSP is a program for Windows that enables you to convert your PlayStation backup disc images (.ISO, .IMG and .CUE+.BIN), also multi-discs, to a single PBP file format.

Open the PSX2PSP.exe application:

Windows:

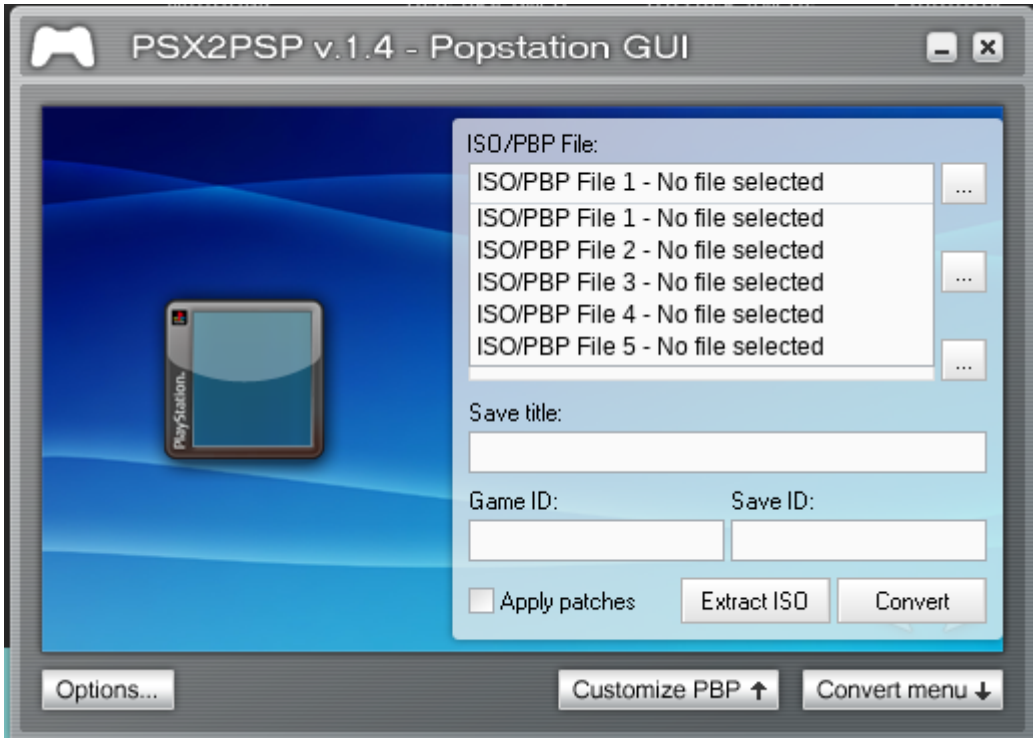
```
PSX2PSP.exe /theme Opens program in theme mode.
PSX2PSP.exe /classic Opens program in classic mode.
PSX2PSP.exe /batch Opens program in batch mode.
PSX2PSP.exe "c:\file.iso" Opens c:\file.iso.
```

Linux:

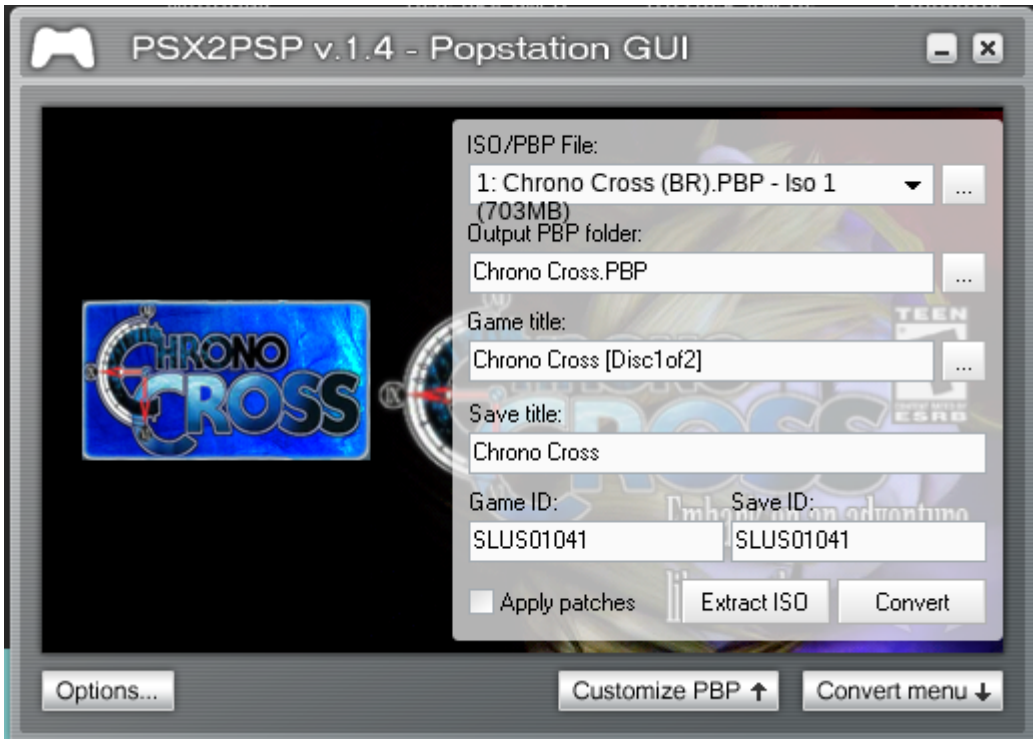
```
wine PSX2PSP.exe
```

or open it with Lutris

Select one or multiple ISO files to convert



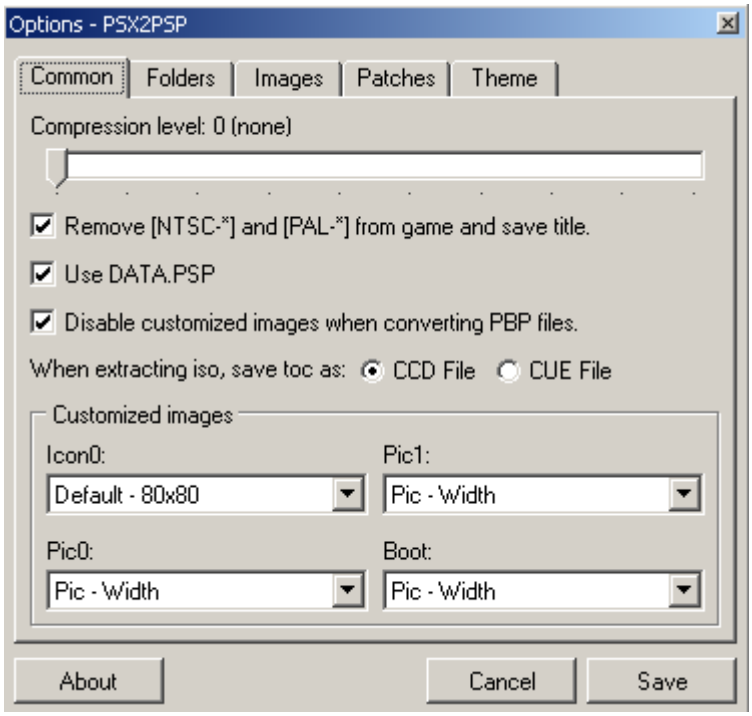
Select the Output destination of the EBOOT.PBP file



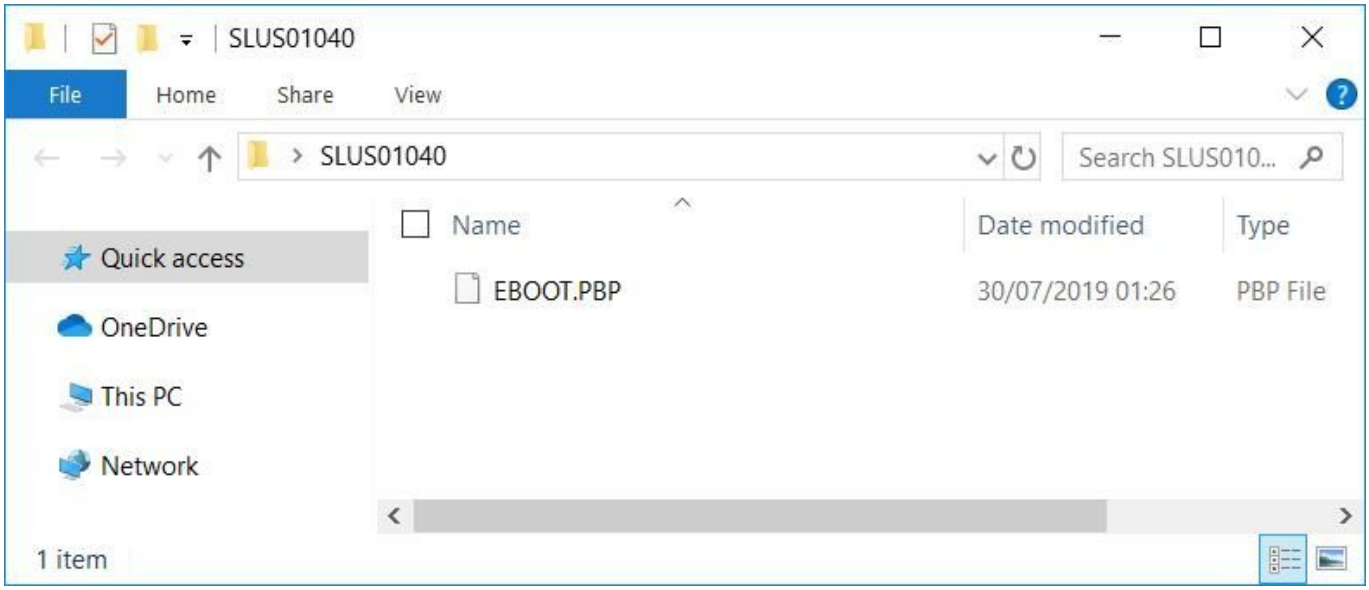
Choose an ICON0.png and a BACKGROUND.png images (at least)

Select Convert

By Default, the PSX2PSP is set to 0 compress, but you can change it to max compression 9 at Options.



A folder with the GAMEID and the EBOOT.PBP file will be generated.



Rename the EBOOT.PBP to the game's name for better scrapper.

To convert the .PBP back to .CUE+.BIN do the same process, but instead select .PBP as the input

With PSXPackager

- **Windows/Mac/Linux:** https://github.com/RupertAvery/PSXPackager/releases/

PSXPackager is a command-line executable (Windows/Linux/OSX) and a Windows-only GUI available.

It enables you to convert your PlayStation backup disc images (.ISO, .IMG and .CUE+.BIN), also multi-discs (with .M3U), to a single PBP file format.

The GUI allows you to select and process several images in a queue (Batch mode).

It needs .NET 7:

```
sudo apt install dotnet7
```

Run the command line: psxpackager -l <compress level from 0 to 9> -i <GAME.cue> -o <GAME.pbp>

```
psxpackager -l 9 -i GAME.cue -o GAME.pbp
```

Convert back to .BIN+.CUE:

```
psxpackager -i GAME.pbp -o /output_path/
```

For multi-discs, create a .m3u with the content refering to the multi-discs:

GAME.m3u

```
GAME - Disc 1.cue
GAME - Disc 2.cue
GAME - Disc 3.cue
```

Run PSXPackager calling the .m3u file

```
psxpackager -l 9 -i GAME.m3u -o GAME.pbp
```

A folder with the GAME and the GAME.PBP file will be generated.

PSXPackager batch converting

Batch converting can be done with a simple `for` loop.

Linux/MacOS:

iso-to-pbp

```
#!/bin/bash
for i in *.iso;
do psxpackager -l 9 -i "${i}" -o "${i}".pbp;
done
```

Here's a command to convert all the games in the current folder and recursive folders to PBP:

all-to-pbp.sh

```
find . -name "*iso" -exec psxpackager -l 9 -i {} -o {}.pbp \;
find . -name "*cue" -exec psxpackager -l 9 -i {} -o {}.pbp \;
find . -name "*m3u" -exec psxpackager -l 9 -i {} -o {}.pbp \;
```

Windows:

all-to-pbp.bat

```
for %i in (*.iso) do psxpackager.exe -l 9 -i "%i" -o "%i".pbp
for %i in (*.cue) do psxpackager.exe -l 9 -i "%i" -o "%i".pbp
for %i in (*.m3u) do psxpackager.exe -l 9 -i "%i" -o "%i".pbp
```

OR

Convert all supported files from SupportedFiles to .PBP

```
psxpackager -i "C:\SupportedFiles\*.*"
psxpackager -i "/SupportedFiles/*.*"
```

Convert all PBP files to .BIN+.CUE

```
psxpackager -i "C:\PBPFiles\*.PBP"
```

```
psxpackager -i "/PBPFiles/*.PBP"
```

Convert all BIN matching 'GAME - Disc *.BIN' files to .PBP

```
psxpackager -i "C:\BinFiles\GAME - Disc ?.bin"
psxpackager -i "/BinFiles/GAME - Disc ?.bin"
```

RVZ

The Dolphin team developed a new compression format based on WIA called RVZ. Unlike all the previous formats, RVZ is lossless and can preserve the padding data on Wii discs as well as the necessary files needed by the Wii's IOS. Usage is very similar to GCZ in Dolphin itself, but it only works on newer Dolphin versions.

Supported systems

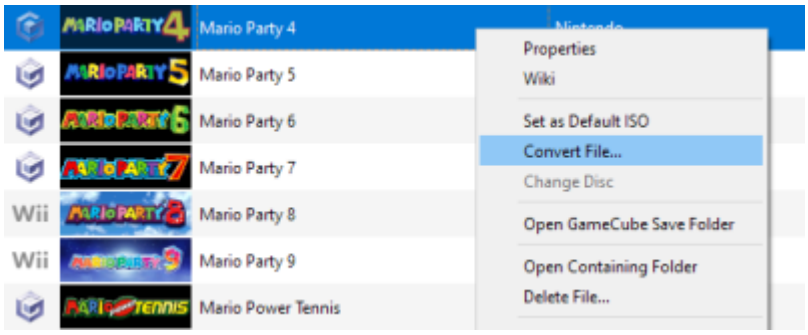
RVZ is only supported by the Dolphin emulator and therefore the recommended format for the following systems:

- gamecube
- wii

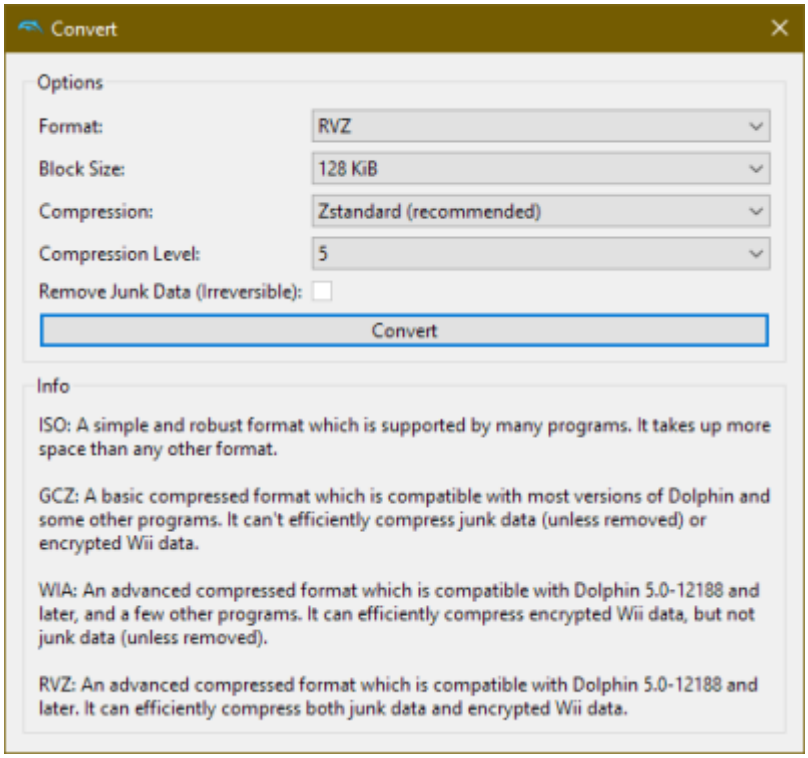
Creating RVZs

You need the Dolphin emulator (minimum version 5.0-12188+) to compress ISO files. At the moment the compression has to be done via `dolphin-config` in the **Applications** folder of the `[F1]` file manager.

Find the game in Dolphin's game list, right click and select **Convert File...**



The default settings should be fine, however for a better compression use those settings: RVZ, 2MiB, LZMA, level 9



This process is completely reversible.

RVZ batch converting

Just multi-select with pressing the *CTRL* key to choose multiple games, right-click and select **Convert File...**

SquashFS

SquashFS allows you to losslessly compress entire folders and all their contained items. This image can be extracted to be identical to the files that were compressed. Unlike ZIP/7Z/GZ files, SquashFS allows Batocera to present the contents of the compressed image to the system *as if though it weren't compressed at all* by automatically mounting them upon launch. This has been a latent feature but full integration with the tools necessary to handle SquashFS images was introduced in Batocera **v33**.

There are some limitations to keep in mind:

- SquashFS is **read-only**. Obviously, this causes issues with games that attempt to write to the directory it has been installed to, which is typical for many PC games to do. Some are exceptions, such as basic DOS titles. For Windows, this can be worked around. (Fix Me! is this only for Windows?)
- Only systems that specify in their `_info.txt` as supporting `squashfs` will be able to launch SquashFS images.

SquashFS is particularly good for PS3 and Wii U games, which are stored as folders which are not written to by the system.

Technical details

First, to be listed in EmulationStation, SquashFS images must be added technically by a developer to the list of eligible system. (aka, dos and ps3 are added for the moment, but it is very easy to add any system while it supports folder as rom). Squashfs images are mounted under /var/run/squashfs/<rom> and this is this directory that is given to the emulator.

Supported systems

System short name	SquashFS Status	v34 stable	v35 beta	Issue	More info
ports	Working	✗	✔	-	Need to create run.sh
scummvm	Working	✔	✔	-	Some games will only works if they are pre-configured on ScummVM
wiiu	Working	✔	✔	https://github.com/batocera-linux/batocera.linux/issues/6114	File <game>/code/<game>.rpx can't be located at /var/run/squashfs/<game> - FIXED on V34.
windows	Working	✔	✗	https://github.com/batocera-linux/batocera.linux/issues/6120	Only works for .wine games: .wtgz performance is very slow compared to .wsquashfs
ps3	Working	✔	✔	-	-
dos	Working	✔	✔	-	-
daphne	Working	✔	✔	-	-
xbox	Working	✔	✔	https://github.com/batocera-linux/batocera.linux/issues/5802	File can't load due to /var/run/squashfs/<game> not be a regular file
easyrpg	Working	✔	✔	-	-
mugen	Not tested yet	-	-	https://github.com/batocera-linux/batocera.linux/issues/6120	Wine issues
ecwolf	Working	✔	✔	-	-

Creating a SquashFS file

This is currently only possible via SSH. Batocera utilizes `mksquashfs` to create SquashFS images. Syntax: `mksquashfs <game> <game>.squashfs`.

For example:

```
cd /userdata/roms/dos
mksquashfs skweek.pc skweek.pc.squashfs
```

Or to compress a single file :

```
cd /userdata/roms/xbox
mksquashfs crazytaxi3.iso crazytaxi3.iso.squashfs
```

Or to compress a single file with the xz compression option:

```
mksquashfs crazytaxi3.iso crazytaxi3.iso.squashfs -comp xz
```

Or to compress multiple files :

```
for i in *; do if [ -d "$i" ]; then mksquashfs "$i" "$i.squashfs"; fi; done
or
find . -maxdepth 1 -name "[^.]*" -type d -exec mksquashfs {} {}.squashfs \;
```

Or to compress multiple files with the xz compression option:

```
for i in *; do if [ -d "$i" ]; then mksquashfs "$i" "$i.squashfs" -comp xz;
fi; done
or
find . -maxdepth 1 -name "[^.]*" -type d -exec mksquashfs {} {}.squashfs -
comp xz \;
```

Or compress multiple .iso files to .squashfs:

```
for i in *.iso; do mksquashfs "${i}" "${i}".squashfs; done
for i in *.iso; do mksquashfs "${i}" "${i}".squashfs -comp xz; done
```

After confirming that the game still launches and runs correctly, the old folder can be safely removed.

If you use the btrfs filesystem, you can enable compression, and you'll get the same compression result without any action.

The run.sh file

When using squashfs for ports games, you must include a `run.sh` file at the root of the compressed filesystem which will run the game.

This file will be executed with the working directory set to `/userdata`, so the first operation you need to do is `cd` into the script's parent directory.

If your game is a 64-bit GOG.com game which requires no special tweaks or additional libraries, you should just be able to use this example script as-is:

run.sh

```
#!/bin/sh
cd "$(dirname "$(readlink -f "$0")")"

unclutter-remote -h
exec ./start.sh
```

If your game is a 32-bit GOG.com game which requires no special tweaks or additional libraries, this example script should do:

run.sh

```
#!/bin/sh
cd "$(dirname "$(readlink -f "$0")")"

unclutter-remote -h
export LD_LIBRARY_PATH=/lib32
export LIBGL_DRIVERS_PATH=/lib32/dri
export SPA_PLUGIN_DIR="/lib32/spa-0.2:/usr/lib/spa-0.2"
export PIPEWIRE_MODULE_DIR="/lib32/pipewire-0.3:/usr/lib/pipewire-0.3"
exec ./start.sh
```

To explain the `cd` line:

- In any shell script, `$0` contains the command used to run the script
- `readlink -f` resolves any symlinks and makes it absolute, ensuring the path will always contain a parent directory
- `dirname` strips off the last component to get the parent directory
- ...and then we `cd` into that.

Uncompress a SquashFS file

Connect via samba and decompress files using a compatilble decompress tool (use the smart uncompress to create a new folder): 7-zip, Picodrive

It can also be uncompressed using the `unsquashfs` command. Syntax: `unsquashfs -f -d <destination> <folder/game>.squashfs`

For example:

```
unsquashfs -f -d /userdata/roms/<system>/<gamedirectory>
/userdata/roms/<system>/<game>.squashfs
```

A successful decompression would look something like this:

```
batocera$ cd /userdata/roms/wiiu/
batocera$ unsquashfs -f -d ExtractionFolder SuperMario.squashfs
Parallel unsquashfs: Using 4 processors
23 inodes (2649 blocks) to write

[============================================================\] 2672/2672
100%

created 23 files
created 4 directories
created 0 symlinks
created 0 devices
created 0 fifos
created 0 sockets
created 0 hardlinks
```

For multiple `unsquashfs` uncompress command:

```
for i in *.squashfs; do unsquashfs -f -d ./"${i%.*}" ./"$i"; done
```

Manually mount a squashfs

It is possible to manually mount a squashFS. This could be used to simply access the data and copy it to somewhere else.

```
mount -t squashfs path/to/file.squashfs /mnt
```

WUA

WUA is a format used by the Wii U emulator Cemu. It is a single compressed file that contains the base game, its updates and installed DLC content.

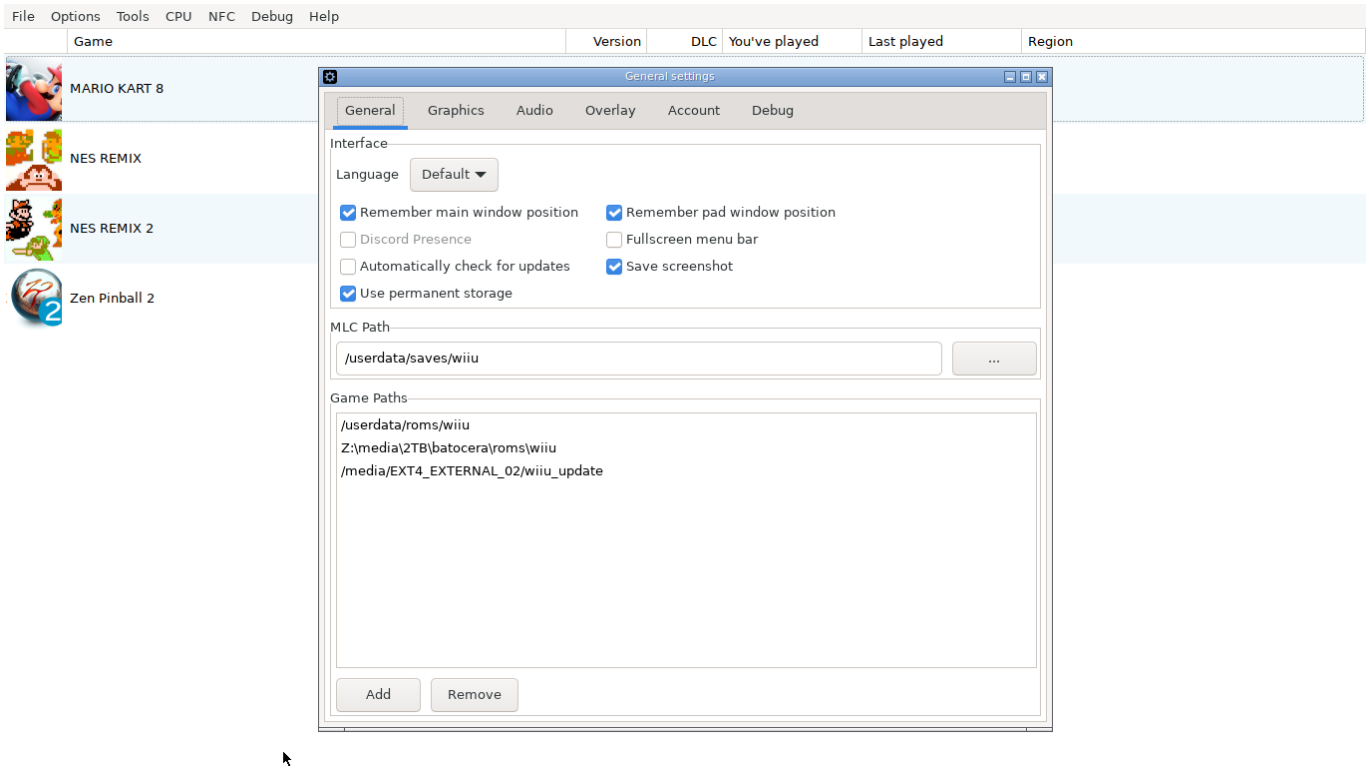
Converting to WUA must be done via Cemu's interface. In Batocera, this can be found in the `[F1]` file manager in **Applications**.

Setup

This is automatically done if Cemu has been launched at least once from ES. However if this is the first time Cemu has been opened, ensure that the paths are pointing to the correct locations:

- **MLC Path:** `/userdata/saves/wiiu`
- **Game Paths:** `/userdata/roms/wiiu`

This screencap does not reflect that. Are multiple paths actually required?



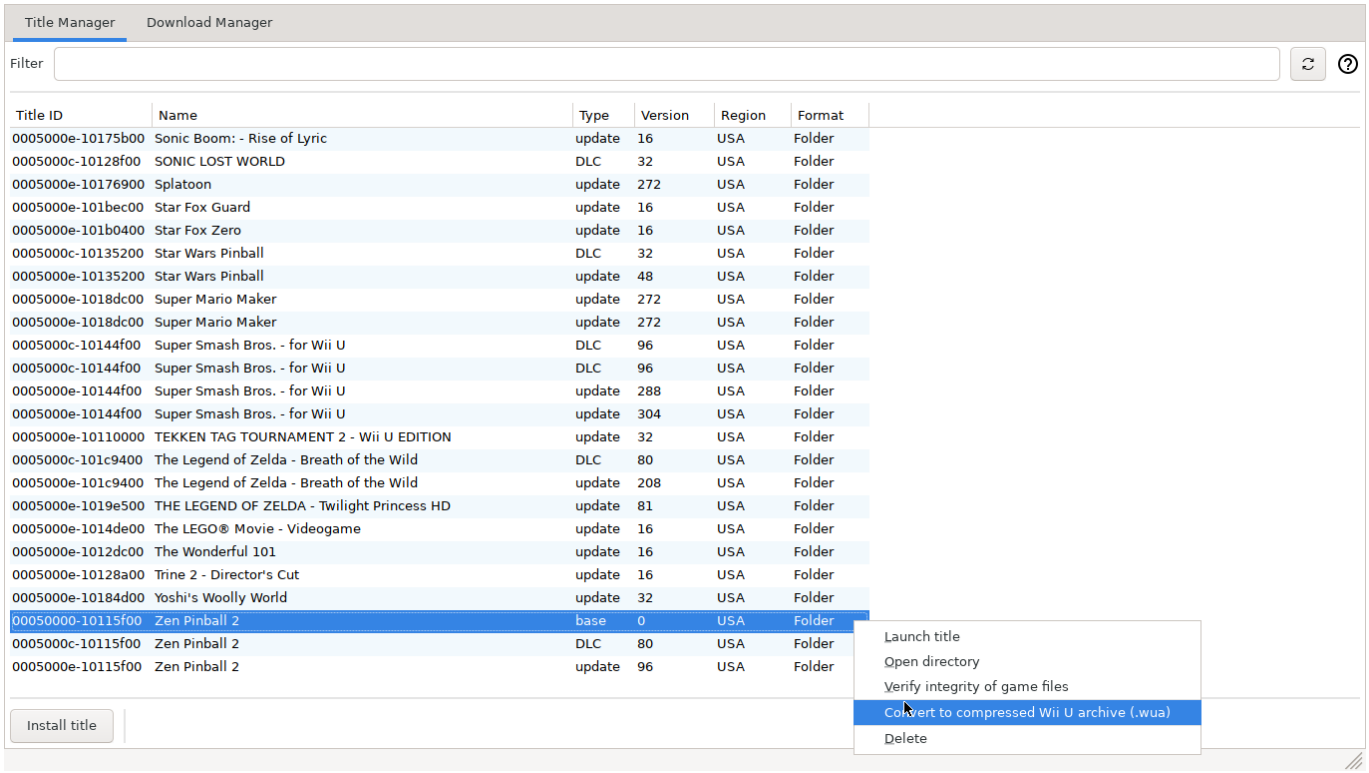
Cemu cannot detect squashFS files. Either mount them first and add them, or decompress them.

Cemu will automatically include any updates and DLCs that are installed when creating the WUA file. Ensure they are installed first before doing this.

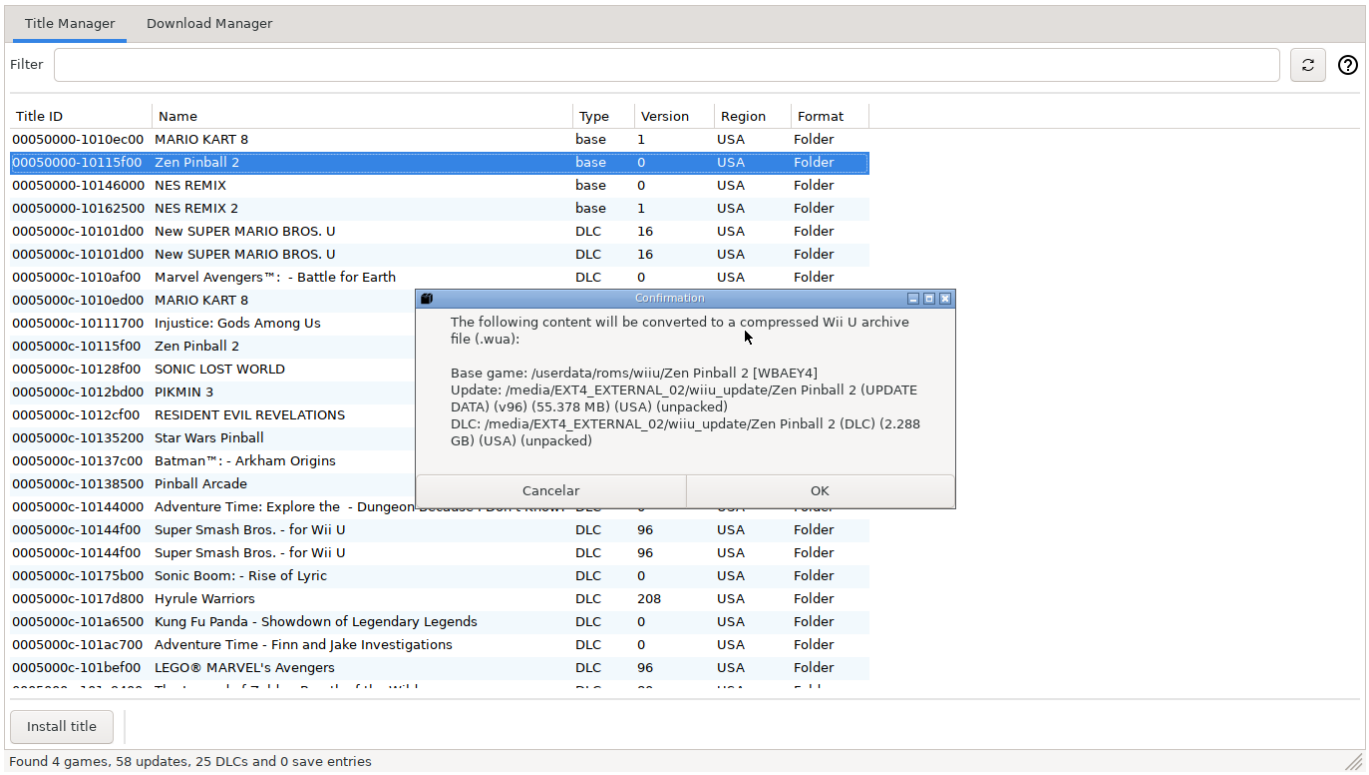
Compress to WUA

Now, go to title manager, to convert them to .wua:

Navigate to **Tools → Title Manager →** (Fix Me! wtf is this?) Left button in the BASE game item with folder format → **Convert to Compressed Wii U archive (.wua)**, then set the destination folder as `/userdata/roms/wiiu`.



Here is the last chance to ensure it's detecting the update and DLC files. The base game, update and DLC must be the same region to be detected.



Hit **OK**.