# BGB version 1.5.7

**homepage:** http://bgb.bircd.org/
**email:** bgb at bircd dot org

this is a gameboy/color emulator/debugger for win32 which will probably never be finished

## disclaimer:

no warranty of any kind, use at your own risk. This program is guaranteed to do nothing but taking disk space. Don't blame me if it damages your computer, or erases your hard drive, etc. Using copyrighted roms with an emulator is illegal if you don't own the real cartridge or have explicit permission from the copyright holder. If you choose to do so, that is your responsibility.

## table of contents



## right click menu:

Click the right mouse button in the emulator window to get the popup menu, from which you can access the options and functions.

## default keys:

| | |
|---|---|
| numpad + | fast forward |
| numpad * | reset gameboy |
| numpad - | cheat codes on/off |
| A | B button |
| S | A button |
| Shift | Select button |
| Enter | Start button |
| F2 | save state |
| F3 | select state |
| F4 | load state |
| Esc | debugger |
| Alt+F4 | exit |

## features in this version:

- emulation of the GameBoy, GameBoy Color, and Super Gameboy
- accurate emulation of the hardware, based on research with lots of test roms, useful for debugging/rom development. some highlights:
  - clock exact timing of LCD behavior/state changes
  - realistic initial ram values - random but with specific bit patterns, and simulated values left by bootroms (for example "nibbler (pd)" depends on this)
  - accurate emulation of LCD register writes during scanline (prehistorik man, demotronic demo)
  - emulation of inaccessible VRAM and OAM as on real hardware
  - 10 sprites per line limit
  - clock exact emulation of sprites causing mode 3 to take longer
  - correct memory access timing (access happening at the last/second to last clock of an opcode)
  - accurate emulation of the differences between DMG and GBC, including timing differences, differences in hardware behavior, initial state, etc.
  - can run a GBC rom as on a DMG, and a DMG rom as on a GBC
- powerful debugger:
  - disassembler
  - assembler (change code and ability to save modified rom)
  - symbols (SYM file) support
  - "inline" editing in code, data, and stack viewer
  - breakpoints
  - break on access
  - conditional breakpoints

    - "on jump" access breakpoints for breaking before a jump into a range is taken.
    - source code breakpoints (ld b,b)
    - singlestepping/tracing/animating/step out/step over
    - vram viewer: BG map, tiles, OAM, palette.
    - IO registers viewer
    - live display of data during emulation and freezing of ram values
    - break on exceptions (accessing inaccessible VRAM, read unitialized RAM, echo ram access, access locked external ram, disable lcd outside vblank)
    - ability to modify all registers and state at any time
    - joypad window allows simulating button presses at any time while debugging
- SGB multiplayer with up to 4 gamepads
- graphics output: GDI, DirectDraw, Direct3D, OpenGL, null output
- graphics doubler: HQ2X, Scale2x, scanlines filter, blocky
- sound output: waveout, directsound, null, and disk writer supported. support for writing the 4 channels to separate wav files
- AVI recording, to installed system codec of choice. No sound included but is in sync and conveniently named with recorded wav file.
- runs almost all roms perfectly, compatibility comparable with the best GB/C emulators
- Accurate/high quality sound emulation, bandlimited synthesis
- accurate video emulation including "high color" graphics, correct sprite/background priorities, 10 sprites/line limit, and mid-scanline register changes.
- This emulator is fast.
- Joystick/Gamepad support, everything mappable to every button
- some user interface keys are configurable and can be mapped to joystick/gamepad buttons
- MBC3 Real Time Clock emulation. RTC is saved/loaded in the .sav file, compatible with VBA
- Auto delay/frameskip, emulation runs at 100% real speed and full 60 fps
- GameGenie and GameShark cheat, load/save cheats (auto and manual), "cheat searcher", easy creation of new cheat codes
- save/load state with quick (zsnes style, and mappable) keys, and preview screenshots for slots in select window.
- Load from ZIP and GZIP files
- good OS/platform compatibility/low requirements (works well on wine, windows 98, and on slow/old PCs)
- support for optional border bitmap and pseudo and real fullscreen modes.
- TCP/IP game link support

## supported platforms/system requirements

OS: supported/known to work on windows 98 up to windows 10, DirextX 7 or later. Wine on Linux and OS X.

soundcard and joystick/gamepad are optional. A videocard with 3D hardware acceleration is recommended. 200 MHz pentium or faster CPU. BGB runs perfectly on an atom netbook.

## accurate emulation and broken PD roms:

by default, BGB emulates the real hardware very accurately, including behavior that some broken PD/unofficial roms may have problems with. These roms would also fail on real hardware, or on a real cartridge: they rely on an inaccurate emulator, or a flashcart. in the "except" tab, one can choose *"Troubleshoot broken PD roms"*, and then set aspects of deliberately inaccurate emulation, to see which one a broken PD rom depends on. Choosing *"Emulate as in reality"* disables and locks out all inaccuracies and should be done if not trying to fix a problem with a bad rom.
one can also set "break on ..." settings to break (go to the debugger) if a rom misbehaves.
note that if you enable any *"break on"* settings, or set breakpoints of any kind, bgb will run in "debug mode", and will be about twice as slow because of extra checks. this is done to allow bgb to run as fast as possible if debug features are disabled.

## known problems:

- bgb 0.x will hang/crash when attempting to open bgb 1.0 save states because the old save state format had no version information.

## options window:

Popup menu --> *Options.*
here you can change a lot of settings such as the joypad keys, configure a gamepad/jostick, change the colors of the gameboy screen, and graphics and sound options. Besides obvious "preference" settings, most settings are optimal as default and should not be changed unless trying to fix a problem.

- ***Sound:***
    - ***Soundcard*** (auto) enable sound output to soundcard API. if this and WAV writer are disabled, sound rendering is disabled completely.
    - ***disable events:*** (off) do not time sound output on callback events. can be used with null output for performance testing.
    - ***8 bits output:*** (off) enable this if 16 bits audio is not supported or

gives problems. Gives lower sound quality, not recommended.
- ***mono output:*** (off) enable this to output mono and mix the channels into it. useful if you listen with one headphone, or soundcard does not support stereo.
- ***High quality sound rendering:*** (on) bandlimited synthesis. disable this if you're on a very slow PC and you're trying to speed up emulation.
- ***Latency:*** (57) setting this higher means lower latency. setting it too high causes sound to skip.
- ***WAV file writer:*** (off) writes wav file. each time it is re-inited, a new WAV with a different timestamp is created
- ***Save individual channels:*** (off) write 4 wav files for the individual hardware channels. this works in addition to/independently of "WAV file writer"
- ***animation recording:*** (off) saves a BMP screenshot for every 1 or 2 frames (as desired), with timestamp prefix matching the WAV file and a frame number.
- ***record AVI:*** (off) record animation to AVI. Give fourcc in box to choose codec. "Config" will open the codec's settings dialog.
- ***record half framerate:*** (on) record ~30 fps instead of ~60 fps, saving only every other frame.

- ***Joypad:***

  click on *"configure keyboard"* or *"configure game controller"*, to show the wizard. press the desired keys/buttons for the hilighted buttons. to change keys for user interface functions (load/save state, fast forward, reset, etc), check *"configure extra buttons"*, then click on *"configure ..."*

  To change only one one or some buttons, click *"skip/keep"* for every other button. For each button, it says which key it is currently mapped to.

  *Mappable button records* button can be mapped through *"configure extra buttons"*.

- ***Misc:***

  - ***show framerate:*** (off) shows framerate in the window title bar, 2 numbers: drawed frames/sec, executed frames/sec.
  - ***freeze recent roms menu:*** (off) Use this to make "recent roms" not change. Can be used for "favorite roms".
  - ***Show errors on rom load:*** (on) When loading a rom, show if something is wrong with the rom header so it would not work on real hardware.

- ***Show warnings on rom load:*** (on) When loading a rom, show any problems/inconsistencies in a rom header. When developing a rom, you should make sure to have no warnings.
- ***reduce CPU usage:*** (on) causes bgb to use the minimum possible cpu by issuing sleep calls when waiting
- ***reduce input latency:*** (auto) use an algorithm that, if vsync is used, delays input polling and emulation to reduce input-to-screen latency.
- ***pause if losing focus:*** (off) pauses/stops bgb if switching to another app.
- ***use fast open files dialog:*** (off) uses bgb's builtin open/save files dialog instead of the default windows one. The fast open dialog also allows selecting Unicode filenames.
- ***framerate:*** (0) can be used to make bgb run slower or faster than real speed, at a desired framerate. 0 runs at the real ~59.72 fps
- ***adjust sound speed:*** (off) change the sound's pitch to match the alternative emulation speed (like a real GB does if you change the clock signal).
- ***undelayed speed:*** (10) how fast to run if pressing the "fast forward" button, 1 is equal to real speed. can also be used to turn the key into a "slow down" key if desired (with a value less than 1)

- ***Graphics:***

  - ***Mix current and last frame:*** (off) can be used to make games show correctly which use "flickering" to mix colors (for example "ballistic")
  - ***doubler:*** (off) applies a screen filter effect such as scanlines or HQ2X, also doubles the size in pixels (useful for fullscreen pageflipped mode if you can't use 320x240)

  the "auto" settings are defaults and are recommended settings. other settings may sometimes have specific benefits or uses.
  - ***bpp:*** (auto) override bits per pixel selection
  - ***output:*** (auto) override choice of output method. use GDI if other methods have problems. null can be used for speed testing (does render, but not output. to disable graphics rendering *and* outputting, change the window's height to 0 instead).
  - ***vsync:*** (auto) override choice of vertical synchronization method. vsync eliminates tearing in the animation. one can disable vsync, or set 1 frame/vblank (instead of synchronizing to real speed)
  - ***stretch:*** (blocky) instruct the graphics output to use "blocky" (point sampling) or "blurry" (bilinear filter) stretching
  - ***fullscreen pageflipped:*** (off) run in a real double/triple buffered fullscreen mode. may have benefits for performance or smooth animation
  - ***GBC LCD colors:*** (on) make GBC colors look as on real GBC. disable

to use RGB values directly without conversion.
  - ***border BMP file:*** (*only SGB border*) you can use a BMP file as border for BGB (it will show if enabled, and there is no SGB border). use a 160x144 pixels plain, one color rectangle to define the location of the GB screen. You can also drag and drop a bitmap file to use it as background.

- ***System:***

  - ***Gameboy:*** runs any rom as on a DMG (original gameboy)
  - ***Gameboy Color:*** runs any rom as on a GBC (gameboy color). this also emulates a DMG rom as on a real GBC in DMG mode, to allow testing/debugging this (for example, legend of zerd breaks as in reality)
  - ***Super Gameboy:*** runs any rom as on a SGB. non-SGB supporting roms are run with SGB initial state, or with SGB bootrom.
  - ***automatic, prefer GBC:*** (default) runs DMG, SGB, and GBC roms on the proper system. runs GBC+SGB roms as on a GBC.
  - ***automatic, prefer SGB:*** runs DMG, SGB, and GBC roms on the proper system. runs GBC+SGB roms as on a SGB.
  - ***SGB + GBC:*** emulates both SGB and GBC features at the same time. because this system does not exist, roms may have problems/unpredictable behavior.
  - ***GBC + initial SGB border:*** starts SGB+GBC roms in SGB mode, until they transferred a border, then resets in GBC mode. this is required for some GBC+SGB roms to show a border.
  - ***Gameboy or GBC:*** runs GBC supporting roms as on GBC, otherwise on DMG (does not enable SGB)
  - ***detect GB pocket / SGB2:*** (off) when in DMG mode, make roms detect a gameboy pocket, possibly unlocking features.
  - ***detect GBA:*** (off) when in GBC mode, make roms detect a gameboy advance, possibly unlocking features (example: "legend of zelda oracle of ages/seasons")
  - ***fast SGB transfers:*** (off) run at unlimited speed during SGB VRAM transfers (inaccurate and can cause problems).
  - ***Waitloop detection:*** (on) detect and shortcut waitloops (which most roms use), speeding up emulation. this is done without compromise on accuracy and has no known problems so it should be left enabled.
  - ***boot roms:*** (off) run the DMG, SGB and GBC bootrom (rom images must be obtained elsewhere). if used with the "gameboy color" system setting, this lets the GBC bootrom colorize DMG roms. The bootroms are not necessary for normal emulation.

If a bootrom is used, it will automatically be patched in ram based on

*"detect GBP"* and *"detect GBA"* settings, to give accurate emulation of a GBP or SGB2 or GBA. If this behavior is undesired, find and set *"BootromPatch=0"* in bgb.ini.

- ***Exceptions:***

  Controls emulation relevant for debugging roms.

  For **playing games**, the defaults are recommended: *"Emulate as in reality"*, and disabling all *"break on"* settings. Enabling any *"break on"* setting will enable "debug mode" and cause emulation to use at least twice as much CPU.

  If **developing a rom**, keep *"Emulate as in reality"* and enable the *"break on"* settings to help catch common problems. If your rom works in bgb with *"Emulate as in reality"* then it is likely to work on real hardware, but you should still test that too.

  To **make a 3rd party unofficial rom work which doesn't work**, you can enable deliberate inaccuracies with with *"Troubleshoot broken PD roms"* and then enable individual inaccuracies, effectively emulating a poor emulator or a generic flash cart. This can help to pinpoint the cause of problems. Such roms will not work on a real gameboy, or on the real cart they report in the header. Do not rely on this when developing a rom.

## GameGenie/GameShark cheat:

how to use a gamegenie/shark code: press F10, or use the rightclick menu and choose *"cheat"*, to show the cheat window. click *"Add"*. click in the *"Code"* field, and paste or type the code. optionally, click in the *"Comment"* field, and enter a description. click "OK". click on the newly added code in the list, and click *"Enable"*.

the cheat window shows a list. you can add/edit/delete cheat entries, with a code and a description. you can load/save the cheats to a file.
meaning of the letters:

- G = game genie
- S = game shark
- P = patched. means a gamegenie code changed values in the rom. if it does not show, the gamegenie code does not work for the used rom.
- E = enabled (toggled with enable/disable buttons)

in the cheat edit window, you can either enter a code, and internal values will be displayed, or you can enter internal values, and the corresponding code will be

displayed.

for a gameshark cheat, you can either use *"enable"* to make it work constantly, or "poke" to make it change the ram value once.

rightclick on a cheat to set an "access breakpoint" (for gameshark codes only), or go to this cheat's rom or ram address in the debugger.

## cheat searcher:

with the cheat searcher and the debugger, you can make new gameshark and gamegenie cheats

Play the game to the point where you want to influence behavior (for example, the game action itself) press esc to enter debugger. menu: *Window -> cheat searcher*. select value type (normally 8 bits) and press start.

you may be looking for a value such as "3 lives", in that case, the value in RAM may be 3 as well, but it's not guaranteed.

continue the game, walking around, killing enemies, etc, but making the value you're looking for *not* change (not dying, not getting hurt, etc), use the criteria *"equal to the previous value"* and search. the number of addresses displayed goes down. you can repeat this a few times. now run the game, but make the value you are looking for change, for example by dying or getting hurt in the game. now search for values which are, for example, *"lower than the previous value"*. the number of matching addresses in the list become smaller each time you run the game and do a search, and you should end up with only one address.

Now, rightclick on the address, and choose *"go here in debugger"*. this will select the address in the ram viewer (on the bottom of the window). then rightclick on that address, and choose *"freeze ram value"*, and enter the desired value. This will create a gameshark code. You can open the cheat window to see the code listed.

if you want to create a gamegenie code, rightclick on the gameshark code, and choose *"set access breakpoint"*. Run the game until the value is changed (for example, die in the game), and it should stop and show the code line on which the value is changed. interpret how the code changes the value, and use game genie codes to change bytes in the code - such as changing an instruction to a nop, or change a "number to decrease" (value 01 to 00), in the cheat window, add new code, enter the address, existing and desired values, and a gamegenie code is produced automatically.

## expressions, breakpoint conditions, and debug messages

In the "condition" field of a breakpoint, one can enter an expression. for example ($ff44)=$90, to trip on the start of each vblank. the condition will only trip once every time when it becomes true. besides constant values and addresses, one can use the following:

CPU registers: AF, BC, DE, HL, SP, PC, B, C, D, E, H, L, A, ZERO, ZF, Z, CARRY, CY, IME, ALLREGS
other state values: ROMBANK, XRAMBANK, SRAMBANK, WRAMBANK, VRAMBANK, TOTALCLKS, LASTCLKS, CLKS2VBLANK

These values can also be evaluated by using the menu, *"Debug"*, *"Evaluate expression"*.

The unit of all "clocks" values is 1 nop in doublespeed mode, or about 0.5 microseconds. Clocks values are, like all other numerical values, in hexadecimal. The expression "ZEROCLKS" will reset the counter for LASTCLKS. TOTALCLKS is the same value as the clocks counter ("internal divider") in the IO map.

To add "ld d,d" debug messages into the code, assemble an opcode in the form: .msg 'message' where message is the message content. debug messages can contain expressions between %%. When enabled in the settings, whenever a debug message is encountered in the code, it will be logged to the debug messages window or log file.

Debug messages also support ternary operators in the form: "%boolean expression%text if true;text if false;".

The internal code that makes up a debug message is as follows: ld d,d; jr @end; dw $6464; dw $0000; db "message"; @end

One can also create debug messages that point to a null terminated text string elsewhere: ld d,d; jr @end; dw $6464; dw $0001; dw address; dw bank; @end

## Commandline parameters

All -params also work as --param.

- **[*-rom*] <filename of rom or state>** load a rom or save state
- ***-setting* name=value** load bgb with a custom setting. this is useful for running multiple (linked) instances on a different position (with winx setting).
- ***-set* name=value** shorthand for -setting
- ***-listen* [[addr:]port]** start bgb listening for game link. if addr/port is omitted, take from settings
- ***-connect* [addr:port]** start bgb connecting for game link. if addr/port is

omitted, take from settings
- ***-demorec* filename** record demo
- ***-demoplay* filename** play back demo
- ***-demoofs* offset** apply offset in bytes to demo when playing
- ***-demo152conv* filename** convert demo between old (1.5.2) and new format (swap nibble order). Works in both directions.
- ***-demo152format*** play or record demos in old 1.5.2 format
- ***-headless*** headless mode: run without window, without sound/graphics /joypad IO. exit when it would break to debugger or when demo finishes. useful for automation. use with start /wait to wait for it to finish.
- ***-autoexit*** terminate when breaking to debugger or finishing demo playback. implied by headless.
- ***-nowarn*** suppress most warning dialogs. useful with automated or unattended use. implied by headless.
- ***-runfast*** run undelayed. useful with headless mode.
- ***-hf*** shorthand for -headless -runfast
- ***-stateonexit* [filename]** save a state when exiting. useful with headless mode.
- ***-screenonexit* [filename]** save a screenshot when exiting. useful with headless mode.
- ***-nobatt*** disable loading/saving of .sav (battery) files.
- ***-nobattsave*** load .sav file read only if it exists, and don't save it.
- ***-loadbatt* filename** load .sav file from filename to go with rom, regardless of -nobatt or settings. useful for automation.
- ***-br* [address][/[condition][/d]][,...]** comma separated list of breakpoints. only in combination with -rom
- ***-ab* [address][/[value][/drwxj]][,...]** comma separated list of access breakpoints. only in combination with -rom
- ***-watch*** automatically reload rom when it changes on disk

BGB will exit with an exit code of 1 in case of an error, and 0 for normal termination. Sadly, console output is not currently possible (console can't be programmatically enabled or disabled).

## Demo recording/playback

There is minimal support for recording and playing back a demo. The support must be considered "experimental", and it may change in the future. It is only intended to be used from the commandline, where the rom to run is also given on the commandline (and not through the menus). One should start from a save state to ensure an exact identical starting state for recording and playback.

The following commandline loads a rom, saves a state, and exits:

```
bgb -hf -br any -rom game.gb -stateonexit game.sna
```

To record a demo:

```
bgb -rom game.sna -demorec game.dem
```

To play back a demo:

```
bgb -rom game.sna -demoplay game.dem
```

One can still load a rom directly (without using a save state) but with a risk that the playback differs from the recording.

Demo recording of linked bgb's works, but only on the same machine, not over lan. This is done because it's the only way that both instances will run at the exact same timing, to allow the demo to be played back. To do this, also link listen or connect using commandline parameters (not the menu). Use commandlines such as the following to start the two instances:

```
bgb -rom red.sna -demorec red.dem -setting winx=300 -listen 127.0.0.1:8765
bgb -rom blue.sna -demorec blue.dem -setting winx=800 -connect 127.0.0.1:8765
```

The demo file format is very simple, with 1 byte per frame for joypad buttons pressed, at the start of each vblank, and before emulation, so it could be handled easily with external tools. If LCD is disabled, no bytes are added to the demo, this was done to keep the demo format deterministic (consistent).

Note: i changed the nibble order around from what it was in 1.5.2, to make it conform to the standard used everywhere: the d-pad should be in the high bits, the other buttons in the low bits. To change existing demo files between the new and old format, use the following command:

```
bgb -demo152conv filename.dem
```

If you need it, there is support for the old format. use the parameter -demo152format (in addition to -demoplay or -demorec), but the old format must be considered deprecated.

## How to make bgb run faster if you have a slow computer

This should not be necessary on any PC later than ~1998. if you have choppy graphics on a modern PC, the problem is something else.

- disable high quality sound rendering, set the sound samplerate to 22KHz, or turn sound off.
- make sure *"debug mode"* is not active (menu, *"other"*, *"debug mode"*)
- disable *"mix current and last frame"* and set *"doubler"* to *"off"*
- Set the desktop resolution to 16 bpp, set emulation to 16 or 8 bpp
- try "fullscreen pageflipped" mode, especially combined with 8 bpp.

## full screen mode:

select window size -> full screen, to go to a "pseudo full screen" mode (borderless). the GB screen will be at the center of the screen and has the same size which the window had before going to fullscreen mode. "full screen stretched" is a pseudo fullscreen mode where the GB screen covers the whole screen.

real (pageflipped) fullscreen modes are also available in the graphics tab of the settings.

## game link:

for bgb game link to work at all, you need a modern PC (around 1-2 GHz) and <1ms lag (fast LAN, or localhost). it does not work with all games. on one bgb, you select "listen", on the other you select "connect". it should then show "linked" in the titlebar. there is also an option "remotejoy" so joypad 2 of one bgb controls the other bgb (which does not have keyboard focus, for example).

As of 1.5.2, link in most games should work, and pokemon should work without any glitches (before, pokemon had occasional glitches). one can link between different games, for example pokemon red and blue. Also, a new feature was added: automatic connecting/reconnecting and re-listening: if the link is broken, it will connect or listen again. Also, it will keep trying to connect until it succeeded. This makes the link feature easier to use, and potentially enables unattended use. If the behavior causes problems, it can be disabled in the ini with *"LinkAutoListen"* and *"LinkAutoConnect"*.

## AVI recorder

As of 1.5, one can record to AVI, in addition to recording to BMP screenshots. this allows for greatly improved efficiency while recording, and allow for longer recordings. Sound is still not included in the AVI; enable WAV file writing too, to record sound. The AVI and WAV will have the same filename (and then .avi and .wav), will be perfectly in sync, and can easily be combined in video editing/transcoding software. "media player classic" will automatically use the WAV accompanying the AVI file, allowing for quick preview.

I recommend getting and installing the "camstudio lossless codec". I tested version 1.5. then in bgb, set the codec field to "cscd" to choose the codec. click "Config" to open the configure dialog, and choose "gzip compression" (this will produce smaller AVI files still). Turn off "record half framerate" to record full ~60 fps, which will still not produce too big AVI files, if desired.

The codec can be uploaded as is to youtube, which supports it, or be edited and transcoded to another codec afterwards.

To stitch together multiple recordings (for example if interrupted by the debugger), it is recommended to first combine the video pieces, and the audio pieces, and then mux the audio and video together.

## Note for speedrun or "race" use

Speedruns, also known as "races", refer to playing games fast, competitively. Speedrun communities may or may not allow the use of emulators, but if they allow them, requirements may be imposed on them, only some emulators may be allowed, etc.

bgb will, by default, run at the real speed (~59.727 fps). When using bgb for speedruns, and one is not sure whether the settings are default, make sure of the following settings, to ensure running at the speed:

- on the *"Graphics"* tab, *"vsync"* is set to *"auto"* (the default), or to *"Vsync disabled"*
- on the *"Misc"* tab, *"framerate"* is set to "0" (the default)

BGB as of 1.5.7 has a "speedrun mode" which changes a lot of behavior. Restart bgb after changing the speedrun mode setting:

- speed is locked to real speed ~59.727 fps
- caption shows *"bgb"* + version + *"SR"*
- caption shows platform being emulated: *"D"* for DMG, *"M"* for GB pocket, *"S"* for SGB, *"S2"* for SGB2, *"C"* for GBC, *"A"* for GBA.
- caption shows *"B"* if bootrom was enabled on last reset.
- bootrom, if used, must be correct
- rom md5 hash is shown briefly when rom is loaded
- fast forward button is disabled
- rapid A/B buttons are disabled
- message is shown in caption for reset
- caption shows *"demo"* if demo is being played
- on loading a state, *"L"* + a counter is shown in caption
- on using gamegenie or gameshark cheats, *"cht"* is shown in caption
- debugger is locked out
- deliberately inaccurate emulation features are locked out
- invalid opcode hangs cpu as in reality
- in GBA mode, reset does a fade and delay like a GB player, where controls briefly still work.
- pressing L+R and U+D simultaneously setting is disabled
- recovery save state is disabled

BGB now (as of 1.5.7) has correct TID (trainer ID) in pokemon games, on GBC and GBA, with and without bootroms. confirmed on GBC and GBA for: red, blue, yellow, gold, silver, crystal (english), and crystal (japanese). confirmed on DMG for red.