NAME

rgblink — Game Boy linker

SYNOPSIS

rgblink [-dtVvwx] [-l *linker_script*] [-m *map_file*] [-n *sym_file*]
[-O *overlay_file*] [-o *out_file*] [-p *pad_value*] [-s *symbol*] *file ...*

DESCRIPTION

The **rgblink** program links RGB object files, typically created by *rgbasm*(1), into a single Game Boy ROM file. The format is documented in *rgbds*(5).

ROM0 sections are placed in the first 16 KiB of the output ROM, and ROMX sections are placed in any 16 KiB "bank" except the first. If your ROM will only be 32 KiB, you can use the `-t` option to change this.

Similarly, WRAM0 sections are placed in the first 4 KiB of WRAM ("bank 0"), and WRAMX sections are placed in any bank of the last 4 KiB. If your ROM doesn't use banked WRAM, you can use the `-w` option to change this.

Also, if your ROM is designed for a monochrome Game Boy, you can make sure that you don't use any incompatible section by using the `-d` option, which implies `-w` but also prohibits the use of banked VRAM.

Note that options can be abbreviated as long as the abbreviation is unambiguous: `--verb` is `--verbose`, but `--ver` is invalid because it could also be `--version`. The arguments are as follows:

`-d`, `--dmg`
: Enable DMG mode. Prohibit the use of sections that doesn't exist on a DMG, such as VRAM bank 1. This option automatically enables `-w`.

`-l` *linker_script*, `--linkerscript` *linker_script*
: Specify a linker script file that tells the linker how sections must be placed in the ROM. The attributes assigned in the linker script must be consistent with any assigned in the code. See *rgblink*(5) for more information about the linker script format.

`-m` *map_file*, `--map` *map_file*
: Write a map file to the given filename, listing how sections and symbols were assigned.

`-n` *sym_file*, `--sym` *sym_file*
: Write a symbol file to the given filename, listing the address of all exported symbols. Several external programs can use this information, for example to help debugging ROMs.

`-O` *overlay_file*, `--overlay` *overlay_file*
: If specified, sections will be overlaid "on top" of the provided ROM image. In that case, all sections must be fixed. This may be used to patch an existing binary.

`-o` *out_file*, `--output` *out_file*
: Write the ROM image to the given file.

`-p` *pad_value*, `--pad` *pad_value*
: When inserting padding between sections, pad with this value. Has no effect if `-O` is specified. The default is 0.

`-s` *symbol*, `--smart` *symbol*
: This option is ignored. It was supposed to perform smart linking but fell into disrepair, and so has been removed. It will be reimplemented at some point.

`-t`, `--tiny`
: Expand the ROM0 section size from 16 KiB to the full 32 KiB assigned to ROM. ROMX sections that are fixed to a bank other than 1 become errors, other ROMX sections are treated as ROM0. Useful for ROMs that fit in 32 KiB.

`-V`, `--version`
: Print the version of the program and exit.

`-v`, `--verbose`
: Verbose: enable printing more information to standard error.

`-w`, `--wramx`
: Expand the WRAM0 section size from 4 KiB to the full 8 KiB assigned to WRAM. WRAMX sections that are fixed to a bank other than 1 become errors, other WRAMX sections are treated as WRAM0.

`-x`, `--nopad`
: Disables padding the end of the final file. This option automatically enables `-t`. You can use this when not not making a ROM. When making a ROM, be careful that not using this is not a replacement for *rgbfix*(1)'s `-p` option!

EXAMPLES

All you need for a basic ROM is an object file, which can be made into a ROM image like so:

```
$ rgblink -o bar.gb foo.o
```

The resulting *`bar.gb`* will not have correct checksums (unless you put them in the assembly source). You should use *rgbfix*(1) to fix these so that the program will actually run in a Game Boy:

```
$ rgbfix -v bar.gb
```

Here is a more complete example:

```
$ rgblink -o bin/game.gb -n bin/game.sym -p 0xFF obj/title.o
obj/engine.o
```

BUGS

Please report bugs on *GitHub*: https://github.com/gbdev/rgbds/issues.

SEE ALSO

rgbasm(1), *rgblink*(5), *rgbfix*(1), *rgbds*(5), *rgbds*(7)

HISTORY

`rgblink` was originally written by Carsten Sørensen as part of the ASMotor package, and was later packaged in RGBDS by Justin Lloyd. It is now maintained by a number of contributors at https://github.com/gbdev/rgbds.