NAME
>   **rgbasm** — Game Boy assembler

SYNOPSIS
>   **rgbasm** [**-EVvw**] [**-b** *chars*] [**-D** *name*[=*value*]] [**-g** *chars*] [**-I** *path*]
>   [**-M** *depend_file*] [**-MG**] [**-MP**] [**-MT** *target_file*] [**-MQ** *target_file*]
>   [**-o** *out_file*] [**-P** *include_file*] [**-p** *pad_value*] [**-Q** *fix_precision*]
>   [**-r** *recursion_depth*] [**-s** *features:state_file*] [**-W** *warning*]
>   [**-X** *max_errors*] *asmfile*

DESCRIPTION
>   The **rgbasm** program creates an RGB object file from an assembly source file. The object file format is
>   documented in rgbds(5).
>
>   The input *asmfile* can be a path to a file, or **-** to read from standard input.
>
>   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:
>
>   **-b** *chars*, **--binary-digits** *chars*
>   >   Change the two characters used for binary constants. The defaults are 01.
>
>   **-D** *name*[=*value*], **--define** *name*[=*value*]
>   >   Add a string symbol to the compiled source code. This is equivalent to *name* **EQUS** "*value*" in
>   >   code, or *name* **EQUS** "1" if *value* is not specified.
>
>   **-E**, **--export-all**
>   >   Export all labels, including unreferenced and local labels.
>
>   **-g** *chars*, **--gfx-chars** *chars*
>   >   Change the four characters used for gfx constants. The defaults are 0123.
>
>   **-I** *path*, **--include** *path*
>   >   Add a new "include path"; *path* must point to a directory. When a **INCLUDE** (including the
>   >   implicit one from **-P**) or **INCBIN** is attempted, **rgbasm** first looks up the provided path from its
>   >   working directory; if this fails, it tries again from each of the "include path" directories, in the order
>   >   they were provided.
>
>   **-M** *depend_file*, **--dependfile** *depend_file*
>   >   Write make(1) dependencies to *depend_file*.
>
>   **-MG**   To be used in conjunction with **-M**. This makes **rgbasm** assume that missing files are auto-gener-
>   >   ated: when **INCLUDE** (including the implicit one from **-P**) or **INCBIN** is attempted on a non-ex-
>   >   istent file, it is added as a dependency, then **rgbasm** exits normally instead of erroring out. This
>   >   feature is used in automatic updating of makefiles.
>
>   **-MP**   When enabled, this causes a phony target to be added for each dependency other than the main file.
>   >   This prevents make(1) from erroring out when dependency files are deleted.
>
>   **-MT** *target_file*
>   >   Add a target to the rules emitted by **-M**. The exact string provided will be written, including spaces
>   >   and special characters.
>   >   **-MT** fileA **-MT** fileB
>   >   is equivalent to
>   >   **-MT** 'fileA fileB'.
>   >   If neither this nor **-MQ** is specified, the output file name is used.

−MQ *target_file*
      Same as **−MT**, but additionally escapes any special make(1) characters, essentially '$'.

−o *out_file*, **−−output** *out_file*
      Write an object file to the given filename.

−P *include_file*, **−−preinclude** *include_file*
      Pre-include a file. This acts as if a **INCLUDE** "*include_file*" was read before the input *asmfile*.

−p *pad_value*, **−−pad-value** *pad_value*
      Use this as the value for **DS** directives in ROM sections, unless overridden. The default is 0x00.

−Q *fix_precision*, **−−q-precision** *fix_precision*
      Use this as the precision of fixed-point numbers after the decimal point, unless they specify their own precision. The default is 16, so fixed-point numbers are Q16.16 (since they are 32-bit integers). The argument may start with a '.' to match the Q notation, for example, **−Q** *.16*.

−r *recursion_depth*, **−−recursion-depth** *recursion_depth*
      Specifies the recursion depth past which **rgbasm** will assume being in an infinite loop. The default is 64.

−s *features*:*state_file*, **−−state** *features*:*state_file*
      Write the specified *features* to *state_file*, based on the final state of **rgbasm** at the end of its input. The expected *features* are a comma-separated subset of the following:

      equ   Write all numeric constants as **def** *name* **equ** *value*.

      var   Write all variables as **def** *name* **=** *value*.

      equs   Write all string constants as **def** *name* **equs** "*value*".

      char   Write all characters as **charmap** *name*, *value*.

      macro Write all macros as **macro** *name* ... **endm**.

      all   Acts like **equ,var,equs,char,macro**.

      This flag may be specified multiple times with different feature subsets to write them to different files (see **EXAMPLES** below).

−V, **−−version**
      Print the version of the program and exit.

−v, **−−verbose**
      Be verbose.

−W *warning*, **−−warning** *warning*
      Set warning flag *warning*. A warning message will be printed if *warning* is an unknown warning flag. See the **DIAGNOSTICS** section for a list of warnings.

−w   Disable all warning output, even when turned into errors.

−X *max_errors*, **−−max-errors** *max_errors*
      If more than this number of errors (not warnings) occur, then abort the assembly process; **−X −0** disables this behavior. The default is 100 if **rgbasm** is printing errors to a terminal, and 0 otherwise.

DIAGNOSTICS

Warnings are diagnostic messages that indicate possibly erroneous behavior that does not necessarily compromise the assembling process. The following options alter the way warnings are processed.

–Werror

Make all warnings into errors.

–Werror=

Make the specified warning into an error. A warning's name is appended (example: **–Werror=obsolete**), and this warning is implicitly enabled and turned into an error. This is an error if used with a meta warning, such as **–Werror=all**.

The following warnings are "meta" warnings, that enable a collection of other warnings. If a specific warning is toggled via a meta flag and a specific one, the more specific one takes priority. The position on the command-line acts as a tie breaker, the last one taking effect.

–Wall

This enables warnings that are likely to indicate an error or undesired behavior, and that can easily be fixed.

–Wextra

This enables extra warnings that are less likely to pose a problem, but that may still be wanted.

–Weverything

Enables literally every warning.

The following warnings are actual warning flags; with each description, the corresponding warning flag is included. Note that each of these flag also has a negation (for example, **–Wcharmap-redef** enables the warning that **–Wno-charmap-redef** disables). Only the non-default flag is listed here. Ignoring the "no-" prefix, entries are listed alphabetically.

–Wno-assert

Warn when **WARN**-type assertions fail. (See "Aborting the assembly process" in rgbasm(5) for **ASSERT**).

–Wbackwards-for

Warn when **FOR** loops have their start and stop values switched according to the step value. This warning is enabled by **–Wall**.

–Wbuiltin-args

Warn about incorrect arguments to built-in functions, such as **STRSUB**() with indexes outside of the string's bounds. This warning is enabled by **–Wall**.

–Wcharmap-redef

Warn when re-defining a charmap mapping. This warning is enabled by **–Wall**.

–Wdiv

Warn when dividing the smallest negative integer (-2**31) by -1, which yields itself due to integer overflow.

–Wempty-macro-arg

Warn when a macro argument is empty. This warning is enabled by **–Wextra**.

–Wempty-strrpl

Warn when **STRRPL**() is called with an empty string as its second argument (the substring to replace). This warning is enabled by **–Wall**.

–Wlarge-constant

    Warn when a constant too large to fit in a signed 32-bit integer is encountered. This warning is enabled by **–Wall**.

–Wmacro-shift

    Warn when shifting macro arguments past their limits. This warning is enabled by **–Wextra**.

–Wno-obsolete

    Warn when obsolete constructs such as the **_PI** constant or **PRINTT** directive are encountered.

–Wnumeric-string=

    Warn when a multi-character string is treated as a number. **–Wnumeric-string=0** or **–Wno-numeric-string** disables this warning. **–Wnumeric-string=1** or just **–Wnumeric-string** warns about strings longer than four characters, since four or fewer characters fit within a 32-bit integer. **–Wnumeric-string=2** warns about any multi-character string.

–Wpurge=

    Warn when purging symbols which are likely to have been necessary. **–Wpurge=0** or **–Wno-purge** disables this warning. **–Wpurge=1** or just **–Wpurge** warns when purging any exported symbol (regardless of type). **–Wpurge=2** also warns when purging any label (even if not exported).

–Wshift

    Warn when shifting right a negative value. Use a division by 2∗∗N instead.

–Wshift-amount

    Warn when a shift's operand is negative or greater than 32.

–Wtruncation=

    Warn when an implicit truncation (for example, **db** to an 8-bit value) loses some bits. **–Wtruncation=0** or **–Wno-truncation** disables this warning. **–Wtruncation=1** warns when an N-bit value is 2∗∗N or greater, or less than -2∗∗N. **–Wtruncation=2** or just **–Wtruncation** also warns when an N-bit value is less than -2∗∗(N-1), which will not fit in two's complement encoding.

–Wunmapped-char=

    Warn when a character goes through charmap conversion but has no defined mapping. **–Wunmapped-char=0** or **–Wunmapped-char** disables this warning. **–Wunmapped-char=1** or just **–Wunmapped-char** only warns if the active charmap is not empty. **–Wunmapped-char=2** warns if the active charmap is empty, and/or is not the default charmap 'main'.

–Wno-user

    Warn when the **WARN** built-in is executed. (See "Aborting the assembly process" in rgbasm(5) for **WARN**).

EXAMPLES

    You can assemble a source file in two ways.

    Straightforward way:

```
$ rgbasm -o bar.o foo.asm
```

    Pipes way:

```
$ cat foo.asm | rgbasm -o bar.o -
$ rgbasm -o bar.o - < foo.asm
```

The resulting object file is not yet a usable ROM image—it must first be run through `rgblink`(1) and then `rgbfix`(1).

Writing the final assembler state to a file:

```
$ rgbasm -s all:state.dump.asm foo.asm
```

Or to multiple files:

```
$ rgbasm -s equ,var:numbers.dump.asm -s equs:strings.dump.asm
foo.asm
```

BUGS

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

SEE ALSO

`rgbasm`(5), `rgblink`(1), `rgbfix`(1), `rgbgfx`(1), `gbz80`(7), `rgbds`(5), `rgbds`(7)

HISTORY

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