NAME

rgbds — object file format documentation

DESCRIPTION

This is the description of the object files used by rgbasm(1) and rgblink(1). Please, note that the specifications may change. This toolchain is in development and new features may require adding more information to the current format, or modifying some fields, which would break compatibility with older versions.

FILE STRUCTURE

The following types are used:

LONG is a 32âbit integer stored in littleâendian format (Intel). *BYTE* is an 8âbit inte ger. *STRING* is a 0âterminated string of *BYTE*.

```
; Header

BYTE    ID[4]            ; "RGB6"
LONG    NumberOfSymbols  ; The number of symbols used in this file
LONG    NumberOfSections ; The number of sections used in this file

; Symbols

REPT    NumberOfSymbols   ; Number of symbols defined in this object file.

    STRING  Name          ; The name of this symbol. Local symbols are stored
                          ; as "Scope.Symbol".

    BYTE    Type          ; 0 = LOCAL symbol only used in this file.
                          ; 1 = IMPORT this symbol from elsewhere (unused).
                          ; 2 = EXPORT this symbol to other objects.

    IF      Type != 1     ; If symbol is defined in this object file.

        STRING  FileName  ; File where the symbol is defined.

        LONG    LineNum   ; Line number in the file where the symbol is defined.

        LONG    SectionID ; The section number (of this object file) in which
                          ; this symbol is defined.

        LONG    Value     ; The symbols value. It's the offset into that
                          ; symbol's section.

    ENDC

ENDR

; Sections

REPT NumberOfSections
    STRING  Name  ; Name of the section

    LONG    Size  ; Size in bytes of this section
```

```
    BYTE    Type  ; 0 = WRAM0
                  ; 1 = VRAM
                  ; 2 = ROMX
                  ; 3 = ROM0
                  ; 4 = HRAM
                  ; 5 = WRAMX
                  ; 6 = SRAM
                  ; 7 = OAM

    LONG    Org   ; Address to fix this section at. -1 if the linker should
                  ; decide (floating address).

    LONG    Bank  ; Bank to load this section into. -1 if the linker should
                  ; decide (floating bank). This field is only valid for ROMX,
                  ; VRAM, WRAMX and SRAM sections.

    LONG    Align ; Alignment of this section (expressed as number of low bits
                  ; to leave as 0). -1 if not defined.

    IF      (Type == ROMX) || (Type == ROM0) ; Sections that can contain data.

        BYTE    Data[Size]      ; Raw data of the section.

        LONG    NumberOfPatches ; Number of patches to apply.

        ; These types of sections may have patches

        REPT    NumberOfPatches

            STRING  SourceFile   ; Name of the source file (for printing error
                                 ; messages).

            LONG    Line         ; The line of the source file.

            LONG    Offset       ; Offset into the section where patch should
                                 ; be applied (in bytes).

            BYTE    Type         ; 0 = BYTE patch.
                                 ; 1 = little endian WORD patch.
                                 ; 2 = little endian LONG patch.
                                 ; 3 = JR offset value BYTE patch.

            LONG    RPNSize      ; Size of the buffer with the RPN.
                                 ; expression.

            BYTE    RPN[RPNSize] ; RPN expression. Definition below.

        ENDR

    ENDC

ENDR
```

RPN DATA

Expressions in the object file are stored as RPN. This is an expression of the form “2 5 +”. This will first push the value “2” to the stack. Then “5”. The “+” operator pops two arguments from the stack, adds them, and then pushes the result on the stack, effectively replacing the two top arguments with their sum. In the RGB format, RPN expressions are stored as BYTEs with some bytes being special prefixes for integers and symbols.

Value	Meaning
$00	+ operator
$01	- operator
$02	* operator
$03	/ operator
$04	% operator
$05	unary -
$10	| operator
$11	& operator
$12	^ operator
$13	unary ~
$21	&& comparison
$22	|| comparison
$23	unary!
$30	== comparison
$31	!= comparison
$32	> comparison
$33	< comparison
$34	>= comparison
$35	<= comparison
$40	<< comparison
$41	>> comparison
$50	BANK(symbol), a *LONG* Symbol ID follows.
$51	BANK(section_name), a null-terminated string follows.
$52	Current BANK().
$60	HRAMCheck. Check if the value is in HRAM, AND it with 0xFF.
$80	*LONG* integer follows.
$81	*LONG* Symbol ID follows.

SEE ALSO

rgbasm(1), rgblink(1), rgbds(7), gbz80(7)

HISTORY

rgbds 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/rednex/rgbds**