RX Family

Graphic LCD Controller Module
Using Firmware Integration Technology

Introduction

This application note describes the graphic LCD controller module using firmware integration technology (FIT). This module uses the graphic LCD controller (GLCDC) to display image data on the liquid crystal display (LCD) panel.

This module is hereinafter referred to as GLCDC FIT module.

Target Devices

The following is a list of devices that are currently supported by this API:

- RX65N, RX651 Groups, ROM capacity: 1.5 MB to 2 MB

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Target Compilers

- Renesas Electronics C/C++ Compiler Package for RX Family
- GCC for Renesas RX
- IAR C/C++ Compiler for Renesas RX

For details of the confirmed operation contents of each compiler, refer to 6.1 Operation Confirmation Environment.

Related Documents

- Firmware Integration Technology User’s Manual (R01AN1833)
- Board Support Package Firmware Integration Technology Module (R01AN1685)

Contents

1. Overview

1.1 GLCDC FIT Module

The GLCDC FIT module can be used by implementing it to the project. For implementing the GLCDC FIT module, refer to 2.12 Adding FIT Module to Your Project.

1.2 Overview of the GLCDC FIT Module

The GLCDC FIT module uses the GLCDC to provide the method to output image data read from memory to the LCD panel.

The GLCDC FIT module supports the following features:

- 32- or 16-bit per pixel image data and 8-, 4-, or 1-bit CLUT (color lookup table) data format
- Superimposition of three planes (and alpha blending of two planes)
- Correction of brightness, contrast, and RGB gamma for the LCD panel to output image data
- Parallel data output of RGB888, RGB666, and RGB565. Dithering for the output data format.

1.3 Summary of the API

Table 1.1 lists API functions included in the GLCDC FIT module:

Table 1.1 API Functions

Function	Description
R_GLCDC_Open	Initializes the GLCDC FIT module.
R_GLCDC_Close	Closes the GLCDC FIT module.
R_GLCDC_Control	Performs control processing for the GLCDC FIT module.
R_GLCDC_LayerChange	Changes operation of graphics 1 and graphics 2 of the GLCDC.
R_GLCDC_ColorCorrection	Changes settings for brightness, contrast, and gamma correction of the GLCDC.
R_GLCDC_ClutUpdate	Updates the CLUT memory of the GLCDC.
R_GLCDC_GetStatus	Obtains the GLCDC status.
R_GLCDC_GetVersion	Returns the version number of the GLCDC FIT module.

1.4 State Transition

Figure 1.1 shows the state transition diagram of the GLCDC FIT module.

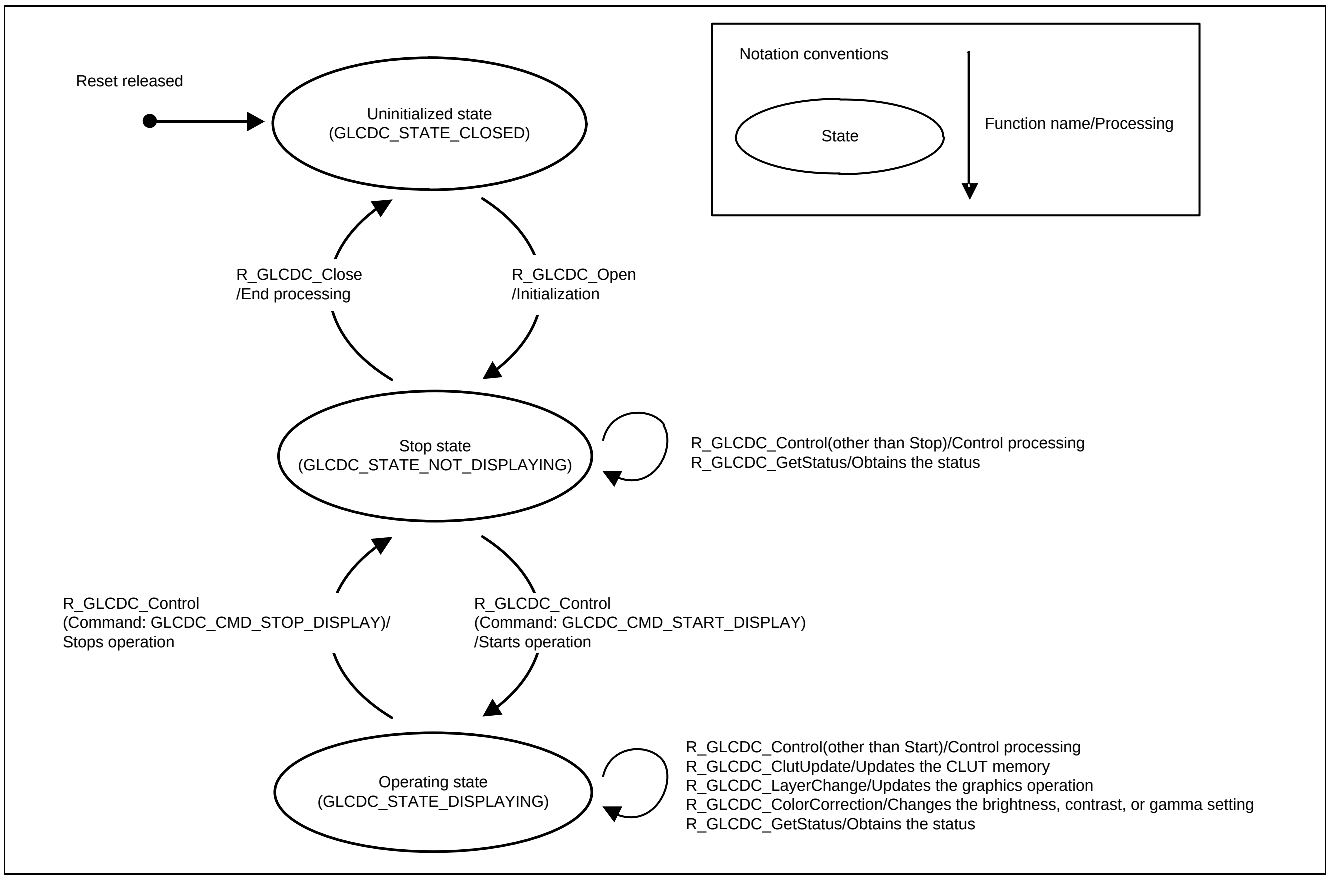
Notation conventions
State
Function name/Processing
Reset released
Uninitialized state
(GLCDC_STATE_CLOSED)
R_GLCDC_Close
/End processing
R_GLCDC_Open
/Initialization
Stop state
(GLCDC_STATE_NOT_DISPLAYING)
R_GLCDC_Control(other than Stop)/Control processing
R_GLCDC_GetStatus/Obtains the status
R_GLCDC_Control
(Command: GLCDC_CMD_STOP_DISPLAY)/
Stops operation
R_GLCDC_Control
(Command: GLCDC_CMD_START_DISPLAY)
/Starts operation
Operating state
(GLCDC_STATE_DISPLAYING)
R_GLCDC_Control(other than Start)/Control processing
R_GLCDC_ClutUpdate/Updates the CLUT memory
R_GLCDC_LayerChange/Updates the graphics operation
R_GLCDC_ColorCorrection/Changes the brightness, contrast, or gamma setting
R_GLCDC_GetStatus/Obtains the status

Figure 1.1 State Transition Diagram of the GLCDC FIT Module

1.5 Limitations

The GLCDC FIT module has the following limitations.

- Output of serial RGB data is not supported.
- Input of an external clock (LCD_EXTCLK) is not supported.

2. API Information

The sample code in this application note has been run and confirmed under the following conditions.

2.1 Hardware Requirements

This FIT module requires that your MCU support the following features:

- GLCDC

2.2 Software Requirements

This FIT module is dependent upon the following packages:

- Renesas Board Support Package (r_bsp) Rev.5.20 or higher.

2.3 Supported Toolchains

This FIT module is tested and working with toolchains listed in 6.1 Operation Confirmation Environment.

2.4 Interrupt Vector

When the R_GLCDC_Open function is executed, the VPOS, GR1UF, and GR2UF interrupts are enabled according to the parameter values.

Table 2.1 lists the interrupt vector used in the GLCDC FIT Module.

Table 2.1 Interrupt Vector Used in the GLCDC FIT Module

Device	Interrupt Vector
RX65N	GROUPAL1 interrupt (vector no. : 113) ● VPOS interrupt (group interrupt source number: 8) ● GR1UF interrupt (group interrupt source number: 9) ● GR2UF interrupt (group interrupt source number: 10)

2.5 Header Files

All API calls and their supporting interface definitions are located in file “r_glcdc_rx_if.h”.

2.6 Integer Types

The GLCDC FIT module uses ANSI C99. These types are defined in file “stdint.h”.

2.7 Configuration Overview

All configurable options that can be set at build time are located in file “r_glcdc_rx_config.h”. A summary of these settings are provided in the following table.

Configuration options in r_glcdc_rx_config.h	
`GLCDC_CFG_PARAM_CHECKING_ENABLE 1`	Defines whether to include parameter checking in the code or not. If the equate is set to 0, the parameter checking is omitted from the build and code size is reduced. If the equate is set to 1, parameter checking is included in the build.
`GLCDC_CFG_INTERRUPT_PRIORITY_LEVEL 5`	Specifies the interrupt priority level for the group AL1 interrupt. Specify the level from 0 to 15.

2.8 Code Size

The sizes of ROM, RAM and maximum stack usage associated with this module are listed below.

The ROM (code and constants) and RAM (global data) sizes are determined by the build-time configuration options described in 2.7 Configuration Overview.
The values in the table below are confirmed under the following conditions.

Module Revision: r_glcdc_rx rev.1.10

Compiler Version: Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00

(The option of "-lang = c99" is added to the default settings of the integrated development environment.)

GCC for Renesas RX 4.8.4.201801

(The option of "-std=gnu99" is added to the default settings of the integrated development environment.)

IAR C/C++ Compiler for Renesas RX version 4.10.1

(The default settings of the integrated development environment.)

Configuration Options: Default settings

ROM, RAM and Stack Code Sizes

Device	Category	Memory Used					
		Renesas Compiler		GCC		IAR Compiler	
		With Parameter Checking	Without Parameter Checking	With Parameter Checking	Without Parameter Checking	With Parameter Checking	Without Parameter Checking
RX65N	ROM	**5890 bytes**	**4714 bytes**	**13128 bytes**	**10304 bytes**	**9675 bytes**	**7748 bytes**
	RAM	**114 bytes**		**24 bytes**		**44 bytes**	
	STACK	**184 bytes**		**-**		**208 bytes**	

2.9 Parameters

This section describes the API data structures used as arguments for the API functions. These structures are located in file “r_glcdc_rx_if.h” as are the prototype declarations.

```
/* Settings for the GLCDC Main */
typedef struct st_glcdc_cfg
{
   /** Generic configuration for display devices */
   glcdc_input_cfg_t input[GLCDC_FRAME_LAYER_NUM];  // GLCDC input image setting
   glcdc_output_cfg_t output;                        // GLCDC output setting
   glcdc_blend_t blend[GLCDC_FRAME_LAYER_NUM];         // Setting for blending
   glcdc_chromakey_t chromakey[GLCDC_FRAME_LAYER_NUM]; // Setting for chroma key
   glcdc_clut_cfg_t clut[GLCDC_FRAME_LAYER_NUM];       // Setting for CLUT

   /** Interrupt setting **/
   glcdc_detect_cfg_t detection;                     // GLCDC detection setting
   glcdc_interrupt_cfg_t interrupt;                  // GLCDC interrupt setting

   /** Setting upon occurrence of GLCDC events **/
   void (*p_callback)(void *);                        // Pointer to the
                                                      // callback function

} glcdc_cfg_t;
```

```
/* GLCDC input image setting */
typedef struct st_glcdc_input_cfg
{
   uint32_t * p_base;              // Start address of the frame buffer
   uint16_t hsize;                 // Horizontal pixel size of image data
   uint16_t vsize;                 // Vertical pixel size of image data
   int32_t offset;                 // Offset value to the next line
   glcdc_in_format_t format;       // Data format setting
   bool frame_edge;                // Show/hide setting of the graphics area
                                   // frame
   glcdc_coordinate_t coordinate; // Position to start displaying image data
   glcdc_color_t bg_color;         // Background color setting for graphics

} glcdc_input_cfg_t;
```

```
/* GLCDC output setting */
typedef struct st_glcdc_output_cfg
{
   glcdc_timing_t       htiming;    // Setting for horizontal synchronous
                                    // signal (HSYNC) timing
   glcdc_timing_t       vtiming;    // Setting for vertical synchronous
                                    // signal (VSYNC) timing
   glcdc_out_format_t   format;     // Setting for output data format
   glcdc_endian_t       endian;     // Bit endian setting for output data
   glcdc_color_order_t color_order; // Pixel sequence setting
   glcdc_sync_edge_t sync_edge; // Setting for output phase of HSYNC/VSYNC/data

   glcdc_color_t        bg_color;   // Setting for background color

   glcdc_brightness_t   brightness; // Setting for brightness
   glcdc_contrast_t     contrast;   // Setting for contrast
   glcdc_gamma_correction_t gamma;  // Setting for gamma correction
   glcdc_correction_proc_order_t correction_proc_order; // Setting for sequence
                                                         // of correction processing
   glcdc_dithering_t    dithering;        // Setting for dithering
   glcdc_tcon_pin_t tcon_hsync;           // Output pin setting for horizontal
                                          // sync signal (HSYNC)
   glcdc_tcon_pin_t tcon_vsync;           // Output pin setting for vertical
                                          // sync signal (VSYNC)
   glcdc_tcon_pin_t tcon_de;              // Output pin setting for data enable
                                          // signal (DE)
   glcdc_signal_polarity_t data_enable_polarity; // Polarity setting for data
                                                 // enable signal (DE)
   glcdc_signal_polarity_t hsync_polarity; // Polarity setting for horizontal
                                           // sync signal (HSYNC)
   glcdc_signal_polarity_t vsync_polarity; // Polarity setting for vertical
                                           // sync signal (VSYNC)

   glcdc_clk_src_t clksrc;                 // Clock source setting
   glcdc_panel_clk_div_t clock_div_ratio;  // Setting for the panel clock
                                           // division ratio

} glcdc_output_cfg_t;
```

```
/* Setting for blending */
typedef struct st_glcdc_blend
{
   glcdc_blend_control_t blend_control; // Control setting for blending
   bool visible;                        // Show/hide setting of image data
   bool frame_edge;                     // Show/hide setting for the frame of
                                        // the rectangle alpha blending area
   uint8_t fixed_blend_value;           // Alpha value setting
   uint8_t fade_speed;                  // Setting for increased/decreased value
                                        // of alpha value
   glcdc_coordinate_t start_coordinate; // Start position of blending
   glcdc_coordinate_t end_coordinate;   // End position of blending

} glcdc_blend_t;
```

```
/* Setting for chroma key */
typedef struct st_glcdc_chromakey
{
   bool          enable; // Enable/disable setting of RGB chroma keying
   glcdc_color_t before; // RGB value setting used for chroma keying
   glcdc_color_t after;  // ARGB value setting after chroma key replacement

} glcdc_chromakey_t;
```

```
/* GLCDC interrupt setting */
typedef struct st_glcdc_interrupt_cfg
{
   bool vpos_enable;    // Enable/disable setting of the VPOS interrupt
   bool gr1uf_enable;   // Enable/disable setting of the GR1UF interrupt
   bool gr2uf_enable;   // Enable/disable setting of the GR2UF interrupt

} glcdc_interrupt_cfg_t;
```

```
/* GLCDC detection setting */
typedef struct st_glcdc_detect_cfg
   {
   bool vpos_detect;    // Enable/disable setting of VPOS detection
   bool gr1uf_detect;   // Enable/disable setting of GR1UF detection
   bool gr2uf_detect;   // Enable/disable setting of GR2UF detection
} glcdc_detect_cfg_t;
```

```
/* Argument for the GLCDC callback function */
typedef struct st_glcdc_callback_args
{
   glcdc_event_t event; // Event code

} glcdc_callback_args_t;
```

```
/* GLCDC status */
typedef struct st_glcdc_status
{
   glcdc_operating_status_t state;        // Status of the GLCDC FIT module
   glcdc_detected_status_t state_vpos;    // Status of notification for
                                          // graphics 2 specified line
   glcdc_detected_status_t state_gr1uf;   // Status of graphics 1 underflow
                                          // detection
   glcdc_detected_status_t state_gr2uf;   // Status of graphics 2 underflow
                                          // detection
   glcdc_fade_status_t fade_status[GLCDC_FRAME_LAYER_NUM];
                                          // Status of alpha blending

} glcdc_status_t;
```

```
/* Setting for dithering */
typedef struct st_glcdc_dithering
{
   bool dithering_on;                                // Enable/disable setting of
                                                     // dithering
   glcdc_dithering_mode_t dithering_mode;            // Dithering mode selection
   glcdc_dithering_pattern_t dithering_pattern_a; // Pattern value A of
                                                     // dithering with 2x2 pattern
   glcdc_dithering_pattern_t dithering_pattern_b; // Pattern value B of
                                                     // dithering with 2x2 pattern
   glcdc_dithering_pattern_t dithering_pattern_c; // Pattern value C of
                                                     // dithering with 2x2 pattern
   glcdc_dithering_pattern_t dithering_pattern_d; // Pattern value D of
                                                     // dithering with 2x2 pattern

} glcdc_dithering_t;
```

```
/* Setting for GLCDC CLUT memory */
typedef struct st_glcdc_clut_cfg
{
   bool       enable;   // Enable/disable setting of CLUT memory
   uint32_t * p_base;   // Pointer to the start address of the CLUT
   uint16_t   start;    // Start entry number for the CLUT memory to be updated
   uint16_t   size;     // Size of the CLUT memory to be updated

} glcdc_clut_cfg_t;
```

```
/* Setting during the GLCDC operation */
typedef struct st_glcdc_runtime_cfg
{
   glcdc_input_cfg_t input;         // GLCDC graphics setting
   glcdc_blend_t     blend;         // Setting for blending
   glcdc_chromakey_t chromakey;     // Setting for chroma key

} glcdc_runtime_cfg_t;
```

```
/* Setting for correction */
typedef struct st_glcdc_correction
{
   glcdc_brightness_t brightness;   // Setting for brightness
   glcdc_contrast_t   contrast;     // Setting for contrast
   glcdc_gamma_correction_t gamma;  // Setting for gamma correction

} glcdc_correction_t;
```

```
/* Setting for gamma correction */
typedef struct st_glcdc_gamma_correction
{
   bool enable;                 // Enable/disable setting of gamma correction
   gamma_correction_t * p_r;  // Setting of gamma correction table for R value
   gamma_correction_t * p_g;  // Setting of gamma correction table for G value
   gamma_correction_t * p_b;  // Setting of gamma correction table for B value

} glcdc_gamma_correction_t;
```

```
/* Setting for gamma correction table */
typedef struct st_gamma_correction
{
   uint16_t gain[GLCDC_GAMMA_CURVE_GAIN_ELEMENT_NUM]; // Gain setting
   uint16_t threshold[GLCDC_GAMMA_CURVE_THRESHOLD_ELEMENT_NUM];
                                                      // Threshold value setting

} gamma_correction_t;
```

```
/* Setting for contrast */
typedef struct st_glcdc_contrast
{
   bool enable;   // Enable/disable setting of contrast adjustment
   uint8_t r;     // Contrast adjustment value for R signal
   uint8_t g;     // Contrast adjustment value for G signal
   uint8_t b;     // Contrast adjustment value for B signal

} glcdc_contrast_t;
```

```
/* Setting for brightness */
typedef struct st_glcdc_brightness
{
   bool enable;   // Enable/disable setting of brightness adjustment
   uint16_t r;    // Brightness adjustment value for R signal
   uint16_t g;    // Brightness adjustment value for G signal
   uint16_t b;    // Brightness adjustment value for B signal

} glcdc_brightness_t;
```

```
/* Coordinate setting */
typedef struct st_glcdc_coordinate
{
   int16_t x; // X-coordinate
   int16_t y; // Y-coordinate

} glcdc_coordinate_t;
```

```
/* Color setting */
typedef struct st_glcdc_color
{
   union
   {
     uint32_t argb;
     struct
     {
        uint32_t a:8; // Value for A
        uint32_t r:8; // Value for R
        uint32_t g:8; // Value for G
        uint32_t b:8; // Value for B
     } byte;
   };

} glcdc_color_t;
```

```
/* Setting for signal output timing */
typedef struct st_glcdc_timing
{
   uint16_t display_cyc;      // Number of cycles for data valid period
   uint16_t front_porch;      // Number of cycles for front porch
   uint16_t back_porch;       // Number of cycles for back porch
   uint16_t sync_width;       // Assertion period

} glcdc_timing_t;
```

```
/* Commands of the R_GLCDC_ColorCorrection function */
typedef enum e_glcdc_correction_cmd
{
   GLCDC_CORRECTION_CMD_SET_ALL,    // All correction setting.
   GLCDC_CORRECTION_CMD_BRIGHTNESS, // Brightness correction setting.
   GLCDC_CORRECTION_CMD_CONTRAST,   // Contrast correction setting.
   GLCDC_CORRECTION_CMD_GAMMA,      // Gamma correction setting.

} glcdc_correction_cmd_t;
```

```
/* Commands of the R_GLCDC_Control function */
typedef enum e_glcdc_control_cmd
{
   GLCDC_CMD_START_DISPLAY,         // Starts GLCDC operation.
   GLCDC_CMD_STOP_DISPLAY,          // Stops GLCDC operation.
   GLCDC_CMD_SET_INTERRUPT,         // Interrupt setting
   GLCDC_CMD_CLR_DETECTED_STATUS,   // Clears detection status.
   GLCDC_CMD_CHANGE_BG_COLOR,       // Change background color in back ground
                                    // screen.

} glcdc_control_cmd_t;
```

```
/* Definition of graphics display */
typedef enum e_glcdc_frame_layer
{
   GLCDC_FRAME_LAYER_1 = 0, // Graphics 1
   GLCDC_FRAME_LAYER_2 = 1  // Graphics 2

} glcdc_frame_layer_t;
```

```
/* Definition of operation mode in the GLCDC FIT module */
typedef enum e_glcdc_state
{
   GLCDC_STATE_CLOSED = 0,           // Before initialization
   GLCDC_STATE_NOT_DISPLAYING = 1,   // GLCDC operation is stopped.
   GLCDC_STATE_DISPLAYING = 2        // GLCDC is operating.

} glcdc_operating_status_t;
```

```
/* Event definition */
typedef enum e_glcdc_event
{
   GLCDC_EVENT_GR1_UNDERFLOW = 1,  // Graphics 1 underflow detected
   GLCDC_EVENT_GR2_UNDERFLOW = 2,  // Graphics 2 underflow detected
   GLCDC_EVENT_LINE_DETECTION = 3, // Graphics 2 specified line notification
                                   // detected

} glcdc_event_t;
```

```
/* Definition of image data format for the frame buffer */
typedef enum e_glcdc_in_format
{
   GLCDC_IN_FORMAT_16BITS_RGB565 = 0,      // RGB(565), 16 bits
   GLCDC_IN_FORMAT_32BITS_RGB888 = 1,      // RGB(888), 32 bits
   GLCDC_IN_FORMAT_16BITS_ARGB1555 = 2,    // ARGB(1555), 16 bits
   GLCDC_IN_FORMAT_16BITS_ARGB4444 = 3,    // ARGB(4444), 16 bits
   GLCDC_IN_FORMAT_32BITS_ARGB8888 = 4,    // ARGB(8888), 32 bits
   GLCDC_IN_FORMAT_CLUT8 = 5,              // CLUT(8), 8 bits
   GLCDC_IN_FORMAT_CLUT4 = 6,              // CLUT(4), 4 bits
   GLCDC_IN_FORMAT_CLUT1 = 7,              // CLUT(1), 1 bit

} glcdc_in_format_t;
```

```
/* Definition of output data format */
typedef enum e_glcdc_out_format
{
   GLCDC_OUT_FORMAT_24BITS_RGB888 = 0,     // RGB(888), 24 bits
   GLCDC_OUT_FORMAT_18BITS_RGB666 = 1,     // RGB(666), 18 bits
   GLCDC_OUT_FORMAT_16BITS_RGB565 = 2,     // RGB(565), 16 bits

} glcdc_out_format_t;
```

```
/* Definition of endianness */
typedef enum e_glcdc_endian
{
   GLCDC_ENDIAN_LITTLE = 0,   // Endianness of output data is little endian.
   GLCDC_ENDIAN_BIG = 1,      // Endianness of output data is big endian.

} glcdc_endian_t;
```

```
/* Definition of pixel sequence */
typedef enum e_glcdc_color_order
{
   GLCDC_COLOR_ORDER_RGB = 0, // Pixel sequence is R-G-B in order.
   GLCDC_COLOR_ORDER_BGR = 1  // Pixel sequence is B-G-R in order.

} glcdc_color_order_t;
```

```
/* Definition of polarity */
typedef enum e_glcdc_signal_polarity
{
   GLCDC_SIGNAL_POLARITY_HIACTIVE = 0, // High active
   GLCDC_SIGNAL_POLARITY_LOACTIVE = 1, // Low active

} glcdc_signal_polarity_t;
```

```
/* Definition of edge for synchronization */
typedef enum e_glcdc_sync_edge
{
   GLCDC_SIGNAL_SYNC_EDGE_RISING = 0,  // Synchronized at a rising edge
   GLCDC_SIGNAL_SYNC_EDGE_FALLING = 1, // Synchronized at a falling edge

} glcdc_sync_edge_t;
```

```
/* Definition for alpha blending */
   typedef enum e_glcdc_blend_control
   {
   GLCDC_BLEND_CONTROL_NONE = 0,    // Alpha blending disabled
   GLCDC_BLEND_CONTROL_FADEIN = 1,  // Fade-in
   GLCDC_BLEND_CONTROL_FADEOUT = 2, // Fade-out
   GLCDC_BLEND_CONTROL_FIXED = 3,   // Fixed alpha value
   GLCDC_BLEND_CONTROL_PIXEL = 4    // Per-pixel alpha blending

} glcdc_blend_control_t;
```

```
/* Definition for fade-in/fade-out status */
typedef enum e_glcdc_fade_status
{
   GLCDC_FADE_STATUS_NOT_UNDERWAY,    // Fade-in/fade-out being stopped
   GLCDC_FADE_STATUS_FADING_UNDERWAY, // Fade-in/fade-out being executed
   GLCDC_FADE_STATUS_UNCERTAIN        // Register value for the graphics
                                      // being specified

} glcdc_fade_status_t;
```

```
/* Clock source definition */
typedef enum e_glcdc_clk_src
{
   GLCDC_CLK_SRC_INTERNAL = 1,        // PLL clock used

} glcdc_clk_src_t;
```

```
/* Definition of the division ratio for the panel clock */
typedef enum e_glcdc_panel_clk_div
{
   GLCDC_PANEL_CLK_DIVISOR_1 = 1,   // x1
   GLCDC_PANEL_CLK_DIVISOR_2 = 2,   // x1/2
   GLCDC_PANEL_CLK_DIVISOR_3 = 3,   // x1/3
   GLCDC_PANEL_CLK_DIVISOR_4 = 4,   // x1/4
   GLCDC_PANEL_CLK_DIVISOR_5 = 5,   // x1/5
   GLCDC_PANEL_CLK_DIVISOR_6 = 6,   // x1/6
   GLCDC_PANEL_CLK_DIVISOR_7 = 7,   // x1/7
   GLCDC_PANEL_CLK_DIVISOR_8 = 8,   // x1/8
   GLCDC_PANEL_CLK_DIVISOR_9 = 9,   // x1/9
   GLCDC_PANEL_CLK_DIVISOR_12 = 12, // x1/12
   GLCDC_PANEL_CLK_DIVISOR_16 = 16, // x1/16
   GLCDC_PANEL_CLK_DIVISOR_24 = 24, // x1/24
   GLCDC_PANEL_CLK_DIVISOR_32 = 32, // x1/32
} glcdc_panel_clk_div_t;
```

```
/* Definition of output pin */
typedef enum e_glcdc_tcon_pin
{
   GLCDC_TCON_PIN_0 = 0, // LCD_TCON0 selected
   GLCDC_TCON_PIN_1 = 1, // LCD_TCON1 selected
   GLCDC_TCON_PIN_2 = 2, // LCD_TCON2 selected
   GLCDC_TCON_PIN_3 = 3  // LCD_TCON3 selected
} glcdc_tcon_pin_t;
```

```
/* Definition for sequence of correction processing */
typedef enum e_glcdc_correction_proc_order
{
   GLCDC_BRIGHTNESS_CONTRAST_TO_GAMMA = 0, // Brightness, contrast ->
                                                      Gamma correction
   GLCDC_GAMMA_TO_BRIGHTNESS_CONTRAST = 1 // Gamma correction ->
                                                  brightness, contrast

} glcdc_correction_proc_order_t;
```

```
/* Definition of dithering mode */
typedef enum e_glcdc_dithering_mode
{
   GLCDC_DITHERING_MODE_TRUNCATE = 0,  // Dithering not processed
   GLCDC_DITHERING_MODE_ROUND_OFF = 1, // 0: Truncated, 1: Rounded
   GLCDC_DITHERING_MODE_2X2PATTERN = 2 // Dithering with 2x2 pattern
} glcdc_dithering_mode_t;
```

```
/* Definition of pattern value for dithering with 2x2 pattern */
typedef enum e_glcdc_dithering_pattern
{
   GLCDC_DITHERING_PATTERN_00 = 0, // Pattern '00'.
   GLCDC_DITHERING_PATTERN_01 = 1, // Pattern '01'.
   GLCDC_DITHERING_PATTERN_10 = 2, // Pattern '10'.
   GLCDC_DITHERING_PATTERN_11 = 3  // Pattern '11'.

} glcdc_dithering_pattern_t;
```

```
/* Definition for detection */
typedef enum e_glcdc_detected_status
{
   GLCDC_NOT_DETECTED,  // Not detected
   GLCDC_DETECTED       // Detected

} glcdc_detected_status_t;
```

2.10 Return Value

This section describes return values for the API functions. This enumeration is located in file “r_glcdc_rx_if.h” as are the prototype declarations.

```
/* GLCDC return values */
typedef enum e_glcdc_err
{
   GLCDC_SUCCESS = 0,                 // Processing has been completed
                                      // successfully.
   GLCDC_ERR_INVALID_PTR,             // NULL pointer is passed to the parameter.
   GLCDC_ERR_LOCK_FUNC,               // GLCDC resource is used by another process
   GLCDC_ERR_INVALID_ARG,             // Invalid argument value
   GLCDC_ERR_INVALID_MODE,            // Function cannot be executed in this mode.
   GLCDC_ERR_NOT_OPEN,                // R_GLCDC_Open has not been executed.
   GLCDC_ERR_INVALID_TIMING_SETTING,  // Register update timing is invalid.
   GLCDC_ERR_INVALID_LAYER_SETTING,  // Graphics screen setting is invalid.
   GLCDC_ERR_INVALID_ALIGNMENT,       // Start address of the frame buffer is
                                      // invalid.
   GLCDC_ERR_INVALID_GAMMA_SETTING,  // Gamma correction setting is invalid.
   GLCDC_ERR_INVALID_UPDATE_TIMING,  // Update timing of the register value is
                                      // invalid.
   GLCDC_ERR_INVALID_CLUT_ACCESS,    // CLUT memory setting is invalid.
   GLCDC_ERR_INVALID_BLEND_SETTING,  // Setting for blending is invalid.

} glcdc_err_t;
```

2.11 Callback Function

In the GLCDC FIT module, a callback function set up by the user is called when the VPOS interrupt, the GR1UF interrupt, or the GR2UF interrupt occurs.
The callback function is set up by storing the address of the callback function in the p_callback structure member described in 2.9 Parameters. When the callback function is called, the constant listed in Table 2.2 is passed as a parameter.
Since the argument type is passed as a pointer to void type, a variable of type pointer to void should be used as the callback function parameter. See an example below as a reference.
To use the argument in the function, its type should be cast.
Unintended specified line notification from graphic 2 (VPOS flag) and graphic 1,2 underflow (GR1UF flag, GR2UF flag) is detected only the first time after GLCDC software reset release. Therefore, do nothing with first VPOS interrupt processing after execution of R_GLCDC_Open function, execute user process from next interrupt.

Table 2.2 Parameters for the Callback Function (enum glcdc_event_t)

Constant Definition	Description
GLCDC_EVENT_LINE_DETECTION	Callback function called from the VPOS interrupt handling
GLCDC_EVENT_GR1_UNDERFLOW	Callback function called from the GR1UF interrupt handling
GLCDC_EVENT_GR2_UNDERFLOW	Callback function called from the GR2UF interrupt handling

```
/* Callback function example */
bool first_interrupt_flag = false;

void my_glcdc_callback(void * pdata)
{
   if (false == first_interrupt_flag)
   {
      first_interrupt_flag = true;
      /* do nothing */
   }
   else
   {
      glcdc_callback_args_t * pdecode;
      pdecode = (glcdc_callback_args_t *)pdata; // cast pointer to
                                                 // glcdc_callback_args_t
      ...
   }
}
```

2.12 Adding FIT Module to Your Project

This module must be added to each project in which it is used. Renesas recommends using “Smart Configurator” described in (1) or (3). However, “Smart Configurator” only supports some RX devices. Please use the methods of (2) or (4) for unsupported RX devices.

(1) Adding the FIT module to your project using “Smart Configurator” in e^2 studio
By using the “Smart Configurator” in e^2 studio, the FIT module is automatically added to your project. Refer to “Renesas e^2 studio Smart Configurator User Guide (R20AN0451)” for details.

(2) Adding the FIT module to your project using “FIT Configurator” in e^2 studio
By using the “FIT Configurator” in e^2 studio, the FIT module is automatically added to your project. Refer to “Adding Firmware Integration Technology Modules to Projects (R01AN1723)” for details.

(3) Adding the FIT module to your project using “Smart Configurator” on CS+
By using the “Smart Configurator Standalone version” in CS+, the FIT module is automatically added to your project. Refer to “Renesas e^2 studio Smart Configurator User Guide (R20AN0451)” for details.

(4) Adding the FIT module to your project in CS+
In CS+, please manually add the FIT module to your project. Refer to “Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)” for details.

2.13 “for”, “while” and “do while” statements

In this module, “for”, “while” and “do while” statements (loop processing) are used in processing to wait for register to be reflected and so on. For these loop processing, comments with “WAIT_LOOP” as a keyword are described. Therefore, if user incorporates fail-safe processing into loop processing, user can search the corresponding processing with “WAIT_LOOP”.

Target devices describing “WAIT_LOOP”

- RX651, RX65N Group

The following shows example of description.

```
while statement example :
/* WAIT_LOOP */
while(0 == SYSTEM.OSCOVFSR.BIT.PLOVF)
{
    /* The delay period needed is to make sure that the PLL has stabilized. */
}

for statement example :
/* Initialize reference counters to 0. */
/* WAIT_LOOP */
for (i = 0; i < BSP_REG_PROTECT_TOTAL_ITEMS; i++)
{
    g_protect_counters[i] = 0;
}

do while statement example :
/* Reset completion waiting */
do
{
    reg = phy_read(ether_channel, PHY_REG_CONTROL);
    count++;
} while ((reg & PHY_CONTROL_RESET) && (count < ETHER_CFG_PHY_DELAY_RESET)); /* WAIT_LOOP */
```

3. API Functions

R_GLCDC_Open ()

This function initializes the GLCDC FIT module. This function must be called before calling any other API functions.

Format

```
glcdc_err_t R_GLCDC_Open (
      glcdc_cfg_t const * const p_cfg
                              /* Pointer to the GLCDC setting data structure */
)
```

Parameters

*glcdc_cfg_t   * p_cfg*
  Pointer to the GLCDC setting data structure.

The following table lists the glcdc_cfg_t structure members and setting values to be referenced.
Only parameters listed below are referenced. Thus the other parameters do not need to be specified when this function is executed.

Table 3.1 glcdc_cfg_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
output.htiming. back_porch	Horizontal back porch	See 5.1 Screen Definition.	Specifies the assertion timing of the STHy signal and the start position of the horizontal active display.
output.htiming. sync_width	Horizontal assertion width	See 5.1 Screen Definition.	Specifies the assertion timing of the STHy signal, the STHy signal assertion width, and the start position of the horizontal active display.
output.vtiming. back_porch	Vertical back porch	See 5.1 Screen Definition.	Specifies the assertion timing of the STVy signal and the start position of the vertical active display.
output.vtiming. sync_width	Vertical assertion width	See 5.1 Screen Definition.	Specifies the assertion timing of the STVy signal, the STVy signal assertion width, and the start position of the vertical active display.
output.htiming. display_cyc	Horizontal active display width	See 5.1 Screen Definition.	Specifies the STHy signal assertion width and the horizontal active display width.
output.vtiming. display_cyc	Vertical active display width	See 5.1 Screen Definition.	Specifies the STVy signal assertion width and the vertical active display width.
output.htiming. front_porch	Horizontal front porch	See 5.1 Screen Definition.	Specifies the horizontal active display width and the start position of horizontal active display.
output.vtiming. front_porch	Vertical front porch	See 5.1 Screen Definition.	Specifies the vertical active display width and the start

			position of vertical active display.
p_callback	Pointer to the callback function	Address of the callback function	Executes the callback function at the address designated by the pointer when an interrupt source occurs.
		FIT_NO_FUNC or NULL	The callback function is not executed even if an interrupt source occurs.
output.clksrc	Clock source	GLCDC_CLK_ SRC_INTERNAL	PLL clock is used.
output.clock_div_ratio	Clock division ratio	1/1 to 1/32 (see "glcdc_panel_clk_div_t" in 0 Parameters for details.	Specifies the division ratio for LCD_CLK.
output.format	Output data format	GLCDC_OUT_FORMAT _ 24BITS_RGB888	Sets RGB888 as the output data format and the output format, and sets the pixel clock to 'no division'.
		GLCDC_OUT_FORMAT _ 18BITS_RGB666	Sets RGB666 as the output data format and the output format, and sets the pixel clock to 'no division'.
		GLCDC_OUT_FORMAT _ 16BITS_RGB565	Sets RGB565 as the output data format and the output format, and sets the pixel clock to 'no division'.
output.sync_edge	Output phase control for TCON and DATA	GLCDC_SIGNAL_ SYNC_EDGE_RISING	Outputs synchronizing with a rising edge of LCD_CLK.
		GLCDC_SIGNAL_ SYNC_EDGE_FALLING	Outputs synchronizing with a falling edge of LCD_CLK.
output.tcon_hsync	Output pin of the horizontal sync signal (HSYNC)	GLCDC_TCON_PIN_0	TCON0 is used for HSYNC output.
		GLCDC_TCON_PIN_1	TCON1 is used for HSYNC output.
		GLCDC_TCON_PIN_2	TCON2 is used for HSYNC output.
		GLCDC_TCON_PIN_3	TCON3 is used for HSYNC output.
output.hsync_polarity	Polarity of the horizontal sync signal (HSYNC)	GLCDC_SIGNAL_ POLARITY_LOACTIVE	Sets polarity to low active.
		GLCDC_SIGNAL_ POLARITY_HIACTIVE	Sets polarity to high active.
output.tcon_vsync	Output pin of the vertical sync signal (VSYNC)	GLCDC_TCON_PIN_0	TCON0 is used for VSYNC output.
		GLCDC_TCON_PIN_1	TCON1 is used for VSYNC output.

		GLCDC_TCON_PIN_2	TCON2 is used for VSYNC output.
		GLCDC_TCON_PIN_3	TCON3 is used for VSYNC output.
output.vsync_polarity	Polarity of the vertical sync signal (VSYNC)	GLCDC_SIGNAL_ POLARITY_LOACTIVE	Sets polarity to low active.
		GLCDC_SIGNAL_ POLARITY_HIACTIVE	Sets polarity to high active.
output.tcon_de	Output pin of the data enable signal (DE)	GLCDC_TCON_PIN_0	TCON0 is used for DE output.
		GLCDC_TCON_PIN_1	TCON1 is used for DE output.
		GLCDC_TCON_PIN_2	TCON2 is used for DE output.
		GLCDC_TCON_PIN_3	TCON3 is used for DE output.
output.data_enable_ polarity	Polarity of the data enable signal (DE)	GLCDC_SIGNAL_ POLARITY_LOACTIVE	Sets polarity to low active.
		GLCDC_SIGNAL_ POLARITY_HIACTIVE	Sets polarity to high active.
output.bg_color.byte.r	R value for the background color	00h to FFh	Specifies the R value for the background color.
output.bg_color.byte.g	G value for the background color	00h to FFh	Specifies the G value for the background color.
output.bg_color.byte.b	B value for the background color	00h to FFh	Specifies the B value for the background color.
input.format	Data format of the frame buffer	GLCDC_IN_FORMAT_ 32BITS_ARGB8888	ARGB8888 is used.
		GLCDC_IN_FORMAT_ 32BITS_RGB888	RGB888 is used.
		GLCDC_IN_FORMAT_ 16BITS_RGB565	RGB565 is used.
		GLCDC_IN_FORMAT_ 16BITS_ARGB1555	ARGB1555 is used.
		GLCDC_IN_FORMAT_ 16BITS_ARGB4444	ARGB4444 is used.
		GLCDC_IN_FORMAT_ CLUT8	8-bit CLUT is used.
		GLCDC_IN_FORMAT_ CLUT4	4-bit CLUT is used.
		GLCDC_IN_FORMAT_ CLUT1	1-bit CLUT is used.
input.p_base	Start address of the frame buffer	0000 0040h to FFFF FFC0h Lower 6 bits are 0.	Specifies the start address of the frame buffer.
		NULL	The target graphics becomes disabled.

			(Setting values of structure members under glcdc_cfg_t.input are ignored.)
input.bg_color.byte.r	R value for the background color of graphics 1 and 2	00h to FFh	Specifies the R value for the background color of graphics 1 and 2.
input.bg_color.byte.g	G value for the background color of graphics 1 and 2	00h to FFh	Specifies the G value for the background color of graphics 1 and 2.
input.bg_color.byte.b	B value for the background color of graphics 1 and 2	00h to FFh	Specifies the B value for the background color of graphics 1 and 2.
input.hsize	Horizontal width of image data	See 5.1 Screen Definition.	Specifies the horizontal width of image for graphics 1 and 2.
input.vsize	Vertical width of image data	See 5.1 Screen Definition.	Specifies the vertical width of image for graphics 1 and 2.
input.offset	Macro line offset	-32768 to 32704 (Multiple of 64)	Specifies the macro line offset for graphics 1 and 2.
input.frame_edge	Show/hide setting of the graphics area frame	true	Sets the graphics area frame to be displayed.
		false	Sets the graphics area frame not to be displayed.
input.coordinate.x	X-coordinate of display start position	See 5.1 Screen Definition.	Specifies the horizontal start position of the graphics area.
input.coordinate.y	Y-coordinate of display start position	See 5.1 Screen Definition.	Specifies the vertical start position of the graphics area.
blend.blend_control	Control setting for blending	GLCDC_BLEND_ CONTROL_NONE	Disables alpha blending.
		GLCDC_BLEND_ CONTROL_FADEIN	Sets to fade-in.
		GLCDC_BLEND_ CONTROL_FADEOUT	Sets to fade-out.
		GLCDC_BLEND_ CONTROL_FIXED	Sets to fixed alpha value.
		GLCDC_BLEND_ CONTROL_PIXEL	Sets to per-pixel alpha blending.
blend.visible	Show/hide setting of the image	true	Sets the image to be displayed.
		false	Sets the image not to be displayed.
blend.frame_edge	Show/hide setting of the rectangle	true	Sets the frame of the rectangle alpha blending area to be displayed.

	alpha blending area frame	false	Sets the frame of the rectangle alpha blending area not to be displayed.
blend.fixed_blend_value	Fixed alpha value	00h to FFh	Specifies the fixed alpha value (valid only when blend_control is 'GLCDC_BLEND_CONTROL_FIXED').
blend.fade_speed	Alpha value to be increased/decreased	00h to FFh	Specifies the alpha value to be increased or decreased (valid only when blend_control is 'GLCDC_BLEND_ CONTROL _FADEIN' or 'GLCDC_BLEND _CONTROL_FADEOUT').
blend.start_coordinate.x	X-coordinate of the blending start position	See 5.1 Screen Definition.	Specifies the horizontal width of the rectangle alpha blending area and the horizontal start position of the rectangle alpha blending.
blend.end_coordinate.x	X-coordinate of the blending end position	See 5.1 Screen Definition.	
blend.start_coordinate.y	Y-coordinate of the blending start position	See 5.1 Screen Definition.	Specifies the vertical width of the rectangle alpha blending area and the vertical start position of the rectangle alpha blending.
blend.end_coordinate.y	Y-coordinate of the blending end position	See 5.1 Screen Definition.	
chromakey.enbale	Enable/disable setting of chroma key	true	Enables chroma keying.
		false	Disables chroma keying. (Setting values of structure members under glcdc_cfg_t.chromakey are ignored.)
chromakey.before.byte.r	R value for chroma keying	00h to FFh	Specifies the R value for chroma keying.
chromakey.before.byte.g	G value for chroma keying	00h to FFh	Specifies the G value for chroma keying.
chromakey.before.byte.b	B value for chroma keying	00h to FFh	Specifies the B value for chroma keying.
chromakey.after.byte.a	A value after chroma key replacement	00h to FFh	Specifies the A value after replacement by chroma keying
chromakey.after.byte.r	R value after chroma key replacement	00h to FFh	Specifies the R value after replacement by chroma keying
chromakey.after.byte.g	G value after chroma key replacement	00h to FFh	Specifies the G value after replacement by chroma keying

chromakey.after.byte.b	B value after chroma key replacement	00h to FFh	Specifies the B value after replacement by chroma keying
output.endian	Bit endianness of the output data	GLCDC_ENDIAN_LITTLE	Sets to little endian.
		GLCDC_ENDIAN_BIG	Sets to big endian.
output.color_order	Pixel sequence of the output data	GLCDC_COLOR_ ORDER_RGB	Sets the pixel sequence of the output data to R-G-B in order.
		GLCDC_COLOR_ ORDER_BGR	Sets the pixel sequence of the output data to B-G-R in order.
output.correction_ proc_order	Sequence of correction processing	GLCDC_ BRIGHTNESS _CONTRAST_TO_ GAMMA	Performs brightness and contrast adjustments first, and then gamma correction.
		GLCDC_ GAMMA_TO_ BRIGHTNESS_ CONTRAST	Performs gamma correction first, and then brightness and contrast adjustments.
output.dithering. dithering_on	Dithering mode selection	true	Sets to ‘0: truncated, 1: rounded’ or dithering with 2x2 pattern.
		false	Sets to ‘truncated’. (Setting values of structure members under glcdc_cfg_t.output.dithering are ignored.)
output.dithering. dithering_mode	Dithering mode selection 2	GLCDC_DITHERING_ MODE_ROUND_OFF	Sets to ‘0: truncated, 1: rounded’.
		GLCDC_DITHERING_ MODE_2X2PATTERN	Sets to dithering with 2x2 pattern.
output.dithering. dithering_ pattern_a	Dithering pattern value A	GLCDC_DITHERING_ PATTERN_00	Specifies pattern value A of dithering with 2x2 pattern (valid only when dithering_mode is ‘GLCDC_DITHERING_MODE_ 2X2PATTERN’).
		GLCDC_DITHERING_ PATTERN_01	
		GLCDC_DITHERING_ PATTERN_10	
		GLCDC_DITHERING_ PATTERN_11	
output.dithering. dithering_ pattern_b	Dithering pattern value B	GLCDC_DITHERING_ PATTERN_00	Specifies pattern value B of dithering with 2x2 pattern (valid only when dithering_mode is ‘GLCDC_DITHERING_MODE_ 2X2PATTERN’).
		GLCDC_DITHERING_ PATTERN_01	
		GLCDC_DITHERING_ PATTERN_10	
		GLCDC_DITHERING_ PATTERN_11	
output.dithering. dithering_ pattern_c	Dithering pattern value C	GLCDC_DITHERING_ PATTERN_00	Specifies pattern value C of dithering with 2x2 pattern

		GLCDC_DITHERING_ PATTERN_01	(valid only when dithering_mode is 'GLCDC_DITHERING_MOD E_ 2X2PATTERN').
		GLCDC_DITHERING_ PATTERN_10	
		GLCDC_DITHERING_ PATTERN_11	
output.dithering. dithering_ pattern_d	Dithering pattern value D	GLCDC_DITHERING_ PATTERN_00	Specifies pattern value D of dithering with 2x2 pattern (valid only when dithering_mode is 'GLCDC_DITHERING_MOD E_ 2X2PATTERN').
		GLCDC_DITHERING_ PATTERN_01	
		GLCDC_DITHERING_ PATTERN_10	
		GLCDC_DITHERING_ PATTERN_11	
output.brightness. enable	Enable/disable setting of brightness adjustment	true	Enables brightness adjustment.
		false	Disables brightness adjustment. (Values for RGB brightness adjustment are set to 0 regardless of setting values of structure members under glcdc_cfg_t.output.brightness .)
output.brightness.r	Brightness adjust. value for R signal	0000h: -512 : 200h: 0 : 3FFh: +511	Specifies the brightness adjustment value for the R signal.
output.brightness.g	Brightness adjust. value for G signal		Specifies the brightness adjustment value for the G signal.
output.brightness.b	Brightness adjust. value for B signal		Specifies the brightness adjustment value for the B signal.
output.contrast.enable	Enable/disable setting of contrast adjustment	true	Enables contrast adjustment.
		false	Disables contrast adjustment. (Values for RGB contrast adjustment are set to 1.000 regardless of setting values of structure members under glcdc_cfg_t.output.contrast.)
output.contrast.r	Contrast adjustment value for R signal	00h: 0/128 = 0.000 : 80h: 128/128 = 1.000 : FFh: 255/128 = 1.992	Specifies the contrast adjustment value for the R signal.
output.contrast.g	Contrast adjustment value for G signal		Specifies the contrast adjustment value for the G signal.
output.contrast.b	Contrast adjustment value for B signal		Specifies the contrast adjustment value for the B signal.

output.gamma.enable	Enable/disable setting of gamma correction	true	Enables gamma correction.
		false	Disables gamma correction. (Setting values of structure members under glcdc_cfg_t.output.gamma are ignored.)
output.gamma.p_r	Gamma correction table for the R signal	See 5.2 Calculating Gamma Correction Value.	Specifies the gain value and the start threshold value for each R signal area.
output.gamma.p_g	Gamma correction table for the G signal	See 5.2 Calculating Gamma Correction Value.	Specifies the gain value and the start threshold value for each G signal area.
output.gamma.p_b	Gamma correction table for the B signal	See 5.2 Calculating Gamma Correction Value.	Specifies the gain value and the start threshold value for each B signal area.
clut.enable	Enable/disable setting of CLUT memory	true	Update CLUT memory.
		false	Not update CLUT memory. (Setting values of structure members under glcdc_cfg_t.clut are ignored.)
clut.p_base	Pointer to the start address of the CLUT memory	Other than NULL	Reads the value at the address designated by the pointer and copies it to the CLUT memory.
clut.start	Start entry number of the CLUT memory to be updated	0 to 255 (start + size < 257)	Starts updating the CLUT memory from the entry number specified.
clut.size	Entry size of the CLUT memory to be updated	1 to 256 (start + size < 257)	Updates the CLUT memory for the specified size.
detection.vpos_detect	Enable/disable setting of VPOS detection	true	Enables VPOS detection.
		false	Disables VPOS detection.
detection.gr1uf_detect	Enable/disable setting of GR1UF detection	true	Enables GR1UF detection.
		false	Disables GR1UF detection.
detection.gr2uf_detect	Enable/disable setting of GR2UF detection	true	Enables GR2UF detection.
		false	Disables GR2UF detection.
interrupt.vpos_enable	Enable/disable setting of the VPOS interrupt	true	Enables the VPOS interrupt.
		false	Disables the VPOS interrupt.
interrupt.gr1uf_enable	Enable/disable setting of the GR1UF interrupt	true	Enables the GR1UF interrupt.
		false	Disables the GR1UF interrupt.

interrupt.gr2uf_enable	Enable/disable setting of the GR2UF interrupt	true	Enables the GR2UF interrupt.
		false	Disables the GR2UF interrupt.

Return Values

GLCDC_SUCCESS	*/* Processing has been completed successfully. */*
GLCDC_ERR_INVALID_PTR	*/* The p_cfg parameter is NULL pointer. */*
GLCDC_ERR_LOCK_FUNC	*/* GLCDC resource is used by another process */*
GLCDC_ERR_INVALID_ARG	*/* The argument for the GLCDC setting data is invalid. */*
GLCDC_ERR_INVALID_MODE	*/* Function cannot be executed in this mode. */*
GLCDC_ERR_INVALID_TIMING_SETTING	*/* Timing setting of the panel output signal is invalid. */*
GLCDC_ERR_INVALID_LAYER_SETTING	*/* Graphics screen setting is invalid. */*
GLCDC_ERR_INVALID_ALIGNMENT	*/* Start address of the frame buffer is invalid. */*
GLCDC_ERR_INVALID_GAMMA_SETTING	*/* Gamma correction setting is invalid. */*
GLCDC_ERR_INVALID_CLUT_ACCESS	*/* CLUT memory setting is invalid. */*
GLCDC_ERR_INVALID_BLEND_SETTING	*/* Setting for blending is invalid. */*

Properties

Prototyped in file “r_glcdc_rx_if.h”

Description

This function releases the GLCDC module-stop state and software reset to enable the GLCDC operation. Then it specifies the panel clock, the panel output signal timing, background screen, graphics screen, CLUT memory, output data format, correction processing, and interrupts used by the GLCDC.
This function can be executed when the mode is 'GLCDC_STATE_CLOSED'. When processing in this function has been completed successfully, a transition is made to 'GLCDC_STATE_NOT_DISPLAYING'.

Reentrant

No.

Example

```
volatile glcdc_err_t ret_glcdc;
glcdc_cfg_t p_cfg;

p_cfg.htiming.back_porch = 2;
... // Set arguments parameter.
p_cfg.interrupt.gr2uf_enable = true;

ret_glcdc = R_GLCDC_Open(&p_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

Special Notes:

- If the target graphics screen is disabled by setting p_base to NULL in this function

  The graphics screen setting in the R_GLCDC_LayerChange function and CLUT memory updates in the R_GLCDC_ClutUpdate function becomes disabled. To enable the disabled graphics, execute the R_GLCDC_Open function again and set the target graphics screen to be enabled.

- Notes on macro line offset setting

  On the hardware specification, since data is read from the frame buffer for every 64 bytes, set a multiple of 64 for structure member input.offset (macro line offset). If it is not possible to observe this restriction, refer to 5.5 When Macro Line Offset Restrictions Cannot Be Followed.

R_GLCDC_Close ()

This function closes the GLCDC FIT module.

Format

```
glcdc_err_t R_GLCDC_Close (
      void
)
```

Parameters

None.

Return Values

GLCDC_SUCCESS	*/* Processing has been completed successfully. */*
GLCDC_ERR_NOT_OPEN	*/* R_GLCDC_Open has not been executed. */*
GLCDC_ERR_INVALID_MODE	*/* Function cannot be executed in this mode. */*

Properties

Prototyped in file “r_glcdc_rx_if.h”

Description

To close the GLCDC FIT module, this function disables interrupts used by the GLCDC. Then it executes the software reset and place the GLCDC in the module-stop state.
This function can be executed when the mode is 'GLCDC_STATE_NOT_DISPLAYING'. When processing in this function has been completed successfully, a transition is made to ' GLCDC_STATE_CLOSED'.

Reentrant

No.

Example

```
volatile glcdc_err_t ret_glcdc;

ret_glcdc = R_GLCDC_Close();
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

Special Notes:

When this function is executed, registers except registers associated with the CLUT memory are initialized.
To enable GLCDC operation again, specify necessary settings again when the R_GLCDC_Open function is executed.

R_GLCDC_Control ()

This function performs processing according to the control command.

Format

```
glcdc_err_t R_GLCDC_Control (
      glcdc_control_cmd_t cmd        /* Control command */
      void const * const  p_args     /* Pointer to the setting parameters */
                                     /* structure */
)
```

Parameters

glcdc_control_cmd_t *cmd*
Control command to specify.
*void const * const* *p_args*
Pointer to the setting parameters structure.

The following table lists available control commands. A void pointer set to the argument is cast to an appropriate type and processed.

Table 3.2 Control Commands of the R_GLCDC_Control Function

Command	Description	Type Set to p_args
GLCDC_CMD_START_ DISPLAY	Enables GLCDC operation and outputs image data on the LCD panel. This command can be executed when the mode is 'GLCDC_STATE_NOT_DISPLAYING'. When processing for this command has been completed successfully, a transition is made to 'GLCDC_STATE_ DISPLAYING'.	Not used. Set NULL or FIT_NO_FUNC.
GLCDC_CMD_STOP_ DISPLAY	Disables GLCDC operation. This command can be executed when the mode is 'GLCDC_STATE_DISPLAYING'. When processing for this command has been completed successfully, a transition is made to 'GLCDC_STATE_NOT_DISPLAYING'.	Not used. Set NULL or FIT_NO_FUNC.
GLCDC_CMD_SET_ INTERRUPT	Specifies interrupts used by the GLCDC. This command can be called at any time after the R_GLCDC_Open function is executed. The mode remains unchanged after processing for this command is complete.	glcdc_interrupt_cfg_t *
GLCDC_CMD_CLR_ DETECTED_STATUS	Clears the status flag for detection of graphics 2 specified line notification, detection of graphics 1 underflow, and detection of graphics 2 underflow. This command can be called at any time after the R_GLCDC_Open function is executed. The mode remains unchanged after processing for this command is complete.	glcdc_detect_cfg_t *
GLCDC_CMD_ CHANGE_BG_COLOR	Specifies background color of the background screen. The mode remains unchanged after processing for this command is complete.	glcdc_color_t *

The following lists the glcdc_interrupt_cfg_t structure members and setting values to be referenced.

When the GLCDC_CMD_SET_INTERRUPT command is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.3 glcdc_interrupt_cfg_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
vpos_enable	Enable/disable setting of the VPOS interrupt	true	Enables the VPOS interrupt.
		false	Disables the VPOS interrupt.
gr1uf_enable	Enable/disable setting of the GR1UF interrupt	true	Enables the GR1UF interrupt.
		false	Disables the GR1UF interrupt.
gr2uf_enable	Enable/disable setting of the GR2UF interrupt	true	Enables the GR2UF interrupt.
		false	Disables the GR2UF interrupt.

The following lists the glcdc_detect_cfg_t structure members and setting values to be referenced.
When the GLCDC_CMD_CLR_DETECTED_STATUS command is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.4 glcdc_detect_cfg_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
vpos_detect	Clearing the VPOS detection flag	true	Clears the VPOS detection flag.
		false	Not clear the VPOS detection flag.
gr1uf_detect	Clearing the GR1UF detection flag	true	Clears the GR1UF detection flag.
		false	Not clear the GR1UF detection flag.
gr2uf_detect	Clearing the GR2UF detection flag	true	Clears the GR2UF detection flag.
		false	Not clear the GR2UF detection flag.

The following lists the glcdc_color_t structure members and setting values to be referenced.
When the GLCDC_CMD_CHANGE_BG_COLOR command is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.5 glcdc_color_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
byte.r	R value of the background color	00h to FFh	Specifies R value of the background color
byte.g	G value of the background color	00h to FFh	Specifies G value of the background color
byte.b	B value of the background color	00h to FFh	Specifies B value of the background color

Return Values

GLCDC_SUCCESS	*/* Processing has been completed successfully. */*
GLCDC_ERR_INVALID_PTR	*/* The p_args parameter is NULL pointer. */*
GLCDC_ERR_INVALID_ARG	*/* The argument set is invalid. */*
GLCDC_ERR_INVALID_MODE	*/* Function cannot be executed in this mode. */*
GLCDC_ERR_NOT_OPEN	*/* R_GLCDC_Open has not been executed. */*
GLCDC_ERR_INVALID_UPDATE_TIMING	*/* Update timing of the register is invalid. */*

Properties
Prototyped in file "r_glcdc_rx_if.h"

Description
This function performs GLCDC control processing according to the control command.

Reentrant
No.

Example
```
/* Enables GLCDC operation */
volatile glcdc_err_t ret_glcdc;

ret_glcdc = R_GLCDC_Control(GLCDC_CMD_START_DISPLAY, NULL);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

```
/* Disables GLCDC operation */
volatile glcdc_err_t ret_glcdc;

ret_glcdc = R_GLCDC_Control(GLCDC_CMD_STOP_DISPLAY, NULL);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

```
/* Changes enable/disable setting of the GLCDC interrupt */
volatile glcdc_err_t ret_glcdc;
glcdc_interrupt_cfg_t int_cfg;

int_cfg.vpos_enable = true;
int_cfg.gr1uf_enable = true;
int_cfg.gr2uf_enable = true;

ret_glcdc = R_GLCDC_Control(GLCDC_CMD_SET_INTERRUPT, (void *)&int_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

```
/* Clears the GLCDC detection status */
volatile glcdc_err_t ret_glcdc;
glcdc_detect_cfg_t detect_cfg;

detect_cfg.vpos_detect = true;
detect_cfg.gr1uf_detect = true;
detect_cfg.gr2uf_detect = true;

ret_glcdc = R_GLCDC_Control(GLCDC_CMD_CLR_DETECTED_STATUS, (void *)&detect_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

```
/* Changes the GLCDC background color */
volatile glcdc_err_t ret_glcdc;
glcdc_color_t bg_color;

bg_color.byte.r = 0xFFh;
bg_color.byte.g = 0xFFh;
bg_color.byte.b = 0xFFh;

ret_glcdc = R_GLCDC_Control(GLCDC_CMD_CHANGE_BG_COLOR, (void *)&bg_color);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

Special Notes:

When the GLCDC_CMD_STOP_DISPLAY command is executed, the GLCDC stops its operation after the frame end of background generating block. To enable GLCDC operation again, wait for the frame end of an output signal to the LCD panel, and then enable GLCDC operation. Otherwise, the GLCDC may not operate correctly depending on the LCD panel used.

R_GLCDC_LayerChange ()

This function changes operation of graphics 1 and graphics 2.

Format

```
glcdc_err_t R_GLCDC_LayerChange (
      glcdc_frame_layer_t frame     /* Graphics screen to change operation */
      glcdc_runtime_cfg_t const * const  p_args       /* Pointer to the setting
                                                      /* parameter structure */
```

Parameters

glcdc_frame_layer_t   *frame*
  Graphics screen to change operation.
*void const * const*   *p_args*
  Pointer to the setting parameters structure.

The following lists the glcdc_runtime_cfg_t structure members and setting values to be referenced.

When this function is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.6 glcdc_runtime_cfg_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
input.format	Data format of the frame buffer	GLCDC_IN_FORMAT_32BITS_ARGB8888	ARGB8888 is used.
		GLCDC_IN_FORMAT_32BITS_RGB888	RGB888 is used.
		GLCDC_IN_FORMAT_16BITS_RGB565	RGB565 is used.
		GLCDC_IN_FORMAT_16BITS_ARGB1555	ARGB1555 is used.
		GLCDC_IN_FORMAT_16BITS_ARGB4444	ARGB4444 is used.
		GLCDC_IN_FORMAT_CLUT8	8-bit CLUT is used.
		GLCDC_IN_FORMAT_CLUT4	4-bit CLUT is used.
		GLCDC_IN_FORMAT_CLUT1	1-bit CLUT is used.
input.p_base	Start address of the frame buffer	0000 0040h to FFFF FFC0h Lower 6 bits are 0.	Specifies the start address of the frame buffer.
input.bg_color.byte.r	R value of the background color for graphics 1 and 2.	00 to FFh	Specifies the R value of the background color for graphics 1 and 2.
input.bg_color.byte.g	G value of the background color for graphics 1 and 2.	00 to FFh	Specifies the G value of the background color for graphics 1 and 2.
input.bg_color.byte.b	B value of the background color for graphics 1 and 2.	00 to FFh	Specifies the B value of the background color for graphics 1 and 2.
input.hsize	Horizontal width of image data	See 5.1 Screen Definition.	Specifies the horizontal width of image for graphics 1 and 2.

input.vsize	Vertical width of image data	See 5.1 Screen Definition.	Specifies the vertical width of image for graphics 1 and 2.
input.offset	Macro line offset	-32768 to 32704 (Multiple of 64)	Specifies the macro line offset for graphics 1 and 2.
input.frame_edge	Show/hide setting of the graphics area frame	true	Sets the graphics area frame to be displayed.
		false	Sets the graphics area frame not to be displayed.
input.coordinate.x	X-coordinate of display start position	See 5.1 Screen Definition.	Specifies the horizontal start position of the graphics area.
input.coordinate.y	Y-coordinate of display start position	See 5.1 Screen Definition.	Specifies the vertical start position of the graphics area.
blend.blend_control	Control setting for blending	GLCDC_BLEND_ CONTROL_NONE	Disables alpha blending.
		GLCDC_BLEND_ CONTROL_FADEIN	Sets to fade-in.
		GLCDC_BLEND_ CONTROL_FADEOUT	Sets to fade-out.
		GLCDC_BLEND_ CONTROL_FIXED	Sets to fixed alpha value.
		GLCDC_BLEND_ CONTROL_PIXEL	Sets to per-pixel alpha blending.
blend.visible	Show/hide setting of the image	true	Sets the image to be displayed.
		false	Sets the image not to be displayed.
blend.frame_edge	Show/hide setting of the rectangle alpha blending area frame	true	Sets the rectangle alpha blending area frame to be displayed.
		false	Sets the rectangle alpha blending area frame not to be displayed.
blend.fixed_blend_val ue	Fixed alpha value	00h to FFh	Specifies the fixed alpha value (valid only when blend_control is 'GLCDC_BLEND_ CONTROL_FIXED').

blend.fade_speed	Alpha value to be increased/decreased	00h to FFh	Specifies the alpha value to be increased or decreased (valid only when blend_control is 'GLCDC_BLEND_CONTROL_FADEIN' or 'GLCDC_BLEND_CONTROL_FADEOUT').
blend.start_coordinate.x	X-coordinate of the blending start position	See 5.1 Screen Definition.	Specifies the horizontal width of the rectangle alpha blending area and the horizontal start position of the rectangle alpha blending.
blend.end_coordinate.x	X-coordinate of the blending end position	See 5.1 Screen Definition.	
blend.start_coordinate.y	Y-coordinate of the blending start position	See 5.1 Screen Definition.	Specifies the vertical width of the rectangle alpha blending area and the vertical start position of the rectangle alpha blending.
blend.end_coordinate.y	Y-coordinate of the blending end position	See 5.1 Screen Definition.	
chromakey.enbale	Enable/disable setting of chroma keying	true	Enables chroma keying.
		false	Disables chroma keying. (Setting value of structure members under glcdc_runtime_cfg_t.chromakey are ignored.)
chromakey.before.byte.r	R value for chroma keying	00h to FFh	Specifies the R value for chroma keying.
chromakey.before.byte.g	G value for chroma keying	00h to FFh	Specifies the G value for chroma keying.
chromakey.before.byte.b	B value for chroma keying	00h to FFh	Specifies the B value for chroma keying.
chromakey.after.byte.a	A value after chroma key replacement	00h to FFh	Specifies the A value after replacement by chroma keying
chromakey.after.byte.r	R value after chroma key replacement	00h to FFh	Specifies the R value after replacement by chroma keying
chromakey.after.byte.g	G value after chroma key replacement	00h to FFh	Specifies the G value after replacement by chroma keying
chromakey.after.byte.b	B value after chroma key replacement	00h to FFh	Specifies the B value after replacement by chroma keying

Return Values

GLCDC_SUCCESS	*/* Processing has been completed successfully. */*
GLCDC_ERR_INVALID_PTR	*/* The p_args parameter is NULL pointer. */*
GLCDC_ERR_INVALID_ARG	*/* The argument set is invalid. */*
GLCDC_ERR_INVALID_MODE	*/* Function cannot be executed in this mode. */*
GLCDC_ERR_NOT_OPEN	*/* R_GLCDC_Open has not been executed. */*
GLCDC_ERR_INVALID_UPDATE_TIMING	*/* Update timing of the register is invalid. */*
GLCDC_ERR_INVALID_ LAYER_SETTING	*/* Graphics screen setting is invalid. */*
GLCDC_ERR_INVALID_ ALIGNMENT	*/* Start address of the frame buffer is invalid. */*
GLCDC_ERR_INVALID_ BLEND_SETTING	*/* Setting for blending is invalid. */*

Properties
Prototyped in file “r_glcdc_rx_if.h”

Description
This function changes operation of graphics 1 and 2.
This function can be executed when the mode is 'GLCDC_STATE_DISPLAYING'. The mode remains unchanged after processing in this function is complete.

Reentrant
No.

Example

```
/* Changes settings for graphics 1 */
volatile glcdc_err_t ret_glcdc;
glcdc_frame_layer_t frame;
glcdc_runtime_cfg_t runtime_cfg;

frame = GLCDC_FRAME_LAYER_1;

runtime_cfg.input.format = GLCDC_IN_FORMAT_CLUT8;
runtime_cfg.input.p_base = (uint32_t *)0x00800000;
runtime_cfg.input.hsize = 448;
runtime_cfg.input.vsize = 253;
runtime_cfg.input.offset = 448;
runtime_cfg.input.frame_edge = false;
runtime_cfg.input.bg_color.byte.r = 0xCC;
runtime_cfg.input.bg_color.byte.g = 0xCC;
runtime_cfg.input.bg_color.byte.b = 0xCC;
runtime_cfg.input.coordinate.x = 16;
runtime_cfg.input.coordinate.y = 9;

runtime_cfg.blend.blend_control = GLCDC_BLEND_CONTROL_NONE;
runtime_cfg.blend.visible = true;
runtime_cfg.blend.frame_edge = false;
runtime_cfg.blend.fixed_blend_value = 0x00;
runtime_cfg.blend.fade_speed = 0x00;
runtime_cfg.blend.start_coordinate.x = 0;
runtime_cfg.blend.start_coordinate.y = 0;
runtime_cfg.blend.end_coordinate.x = 0;
runtime_cfg.blend.end_coordinate.y = 0;

runtime_cfg.chromakey.enable = false;
runtime_cfg.chromakey.before.byte.g = 0x00;
runtime_cfg.chromakey.before.byte.b = 0x00;
runtime_cfg.chromakey.before.byte.r = 0x00;
runtime_cfg.chromakey.after.byte.a = 0x00;
runtime_cfg.chromakey.after.byte.g = 0x00;
runtime_cfg.chromakey.after.byte.b = 0x00;
runtime_cfg.chromakey.after.byte.r = 0x00;

ret_glcdc = R_GLCDC_LayerChange(frame, &runtime_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

Special Notes:
None.

R_GLCDC_ColorCorrection ()

This function changes settings for brightness, contrast, and gamma correction of the GLCDC.

Format

```
glcdc_err_t R_GLCDC_ColorCorrection (
      glcdc_correction_cmd_t cmd    /* Command to change the setting */
      void const * const p_args     /* Pointer to the setting parameter */
                                    /* structure */
)
```

Parameters

glcdc_correction_cmd_t *cmd*
Command to change the setting
*void const * const* *p_args*
Pointer to the setting parameter structure

The following table lists the available control commands. A void pointer set to the argument is cast to an appropriate type and processed.

Table 3.7 Control Commands of the R_GLCDC_ColorCorrection Function

Command	Description	Type Set to p_args
GLCDC_CORRECTION_ CMD_SET_ALL	Specifies settings for brightness and contrast adjustments, and gamma correction.	glcdc_correction_t *
GLCDC_CORRECTION_ CMD_BRIGHTNESS	Specifies the setting for brightness adjustment.	glcdc_brightness_t *
GLCDC_CORRECTION_ CMD_CONTRAST	Specifies the setting for contrast adjustment.	glcdc_contrast_t *
GLCDC_CORRECTION_ CMD_GAMMA	Specifies the setting for gamma correction.	glcdc_gamma_ correction_t *

The following lists the glcdc_correction_t structure members and setting values to be referenced.
When the GLCDC_CORRECTION_CMD_SET_ALL command is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.8 glcdc_correction_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
brightness.enable	Enable/disable setting of brightness adjustment	true	Enables brightness adjustment.
		false	Disables brightness adjustment. (Values for RGB brightness adjustment are set to 0 regardless of setting values of the structure members under glcdc_correction_t.brightness.)
brightness.r	Brightness adjust. value for R signal	000h: -512 : 200h: 0 : 3FFh: +511	Specifies the brightness adjustment value for the R signal.
brightness.g	Brightness adjust. value for G signal		Specifies the brightness adjustment value for the G signal.
brightness.b	Brightness adjust. value for B signal		Specifies the brightness adjustment value for the B signal.
contrast.enable	Enable/disable setting of contrast adjustment	true	Enables contrast adjustment.
		false	Disables contrast adjustment. (Values for RGB contrast adjustment are set to 1.000 regardless of setting values of structure members under glcdc_correction_t.contrast)
contrast.r	Contrast adjustment value for R signal	00h: 0/128 = 0.000 : 80h: 128/128 = 1.000 : FFh: 255/128 = 1.992	Specifies the contrast adjustment value for the R signal.
contrast.g	Contrast adjustment value for G signal		Specifies the contrast adjustment value for the G signal.
contrast.b	Contrast adjustment value for B signal		Specifies the contrast adjustment value for the B signal.
gamma.enable	Enable/disable setting of gamma correction	true	Enables gamma correction.
		false	Disables gamma correction. (Setting values of structure members under glcdc_correction_t.gamma are ignored.)
gamma.p_r	Gamma correction table for the R signal	See 5.2 Calculating Gamma Correction Value.	Specifies a gain and start threshold value for each R signal area.

gamma.p_g	Gamma correction table for the G signal	See 5.2 Calculating Gamma Correction Value.	Specifies a gain and start threshold value for each G signal area.
gamma.p_b	Gamma correction table for the B signal	See 5.2 Calculating Gamma Correction Value.	Specifies a gain and start threshold value for each B signal area.

The following lists the glcdc_brightness_t structure members and setting values to be referenced.
When the GLCDC_CORRECTION_CMD_BRIGHTNESS command is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.9 glcdc_brightness_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
enable	Enable/disable setting of brightness adjustment	true	Enables brightness adjustment.
		false	Disables brightness adjustment. (Values for RGB brightness adjustment are set to 0 regardless of setting values of structure members under glcdc_brightness_t.)
r	Brightness adjust. value for R signal	000h: -512 : 200h: 0 : 3FFh: +511	Specifies the brightness adjustment value for the R signal.
g	Brightness adjust. value for G signal		Specifies the brightness adjustment value for the G signal.
b	Brightness adjust. value for B signal		Specifies the brightness adjustment value for the B signal.

The following lists the glcdc_contrast_t structure members and setting values to be referenced.
When the GLCDC_CORRECTION_CMD_CONTRAST command is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.10 glcdc_contrast_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
enable	Enable/disable setting of contrast adjustment	true	Enables contrast adjustment.
		false	Disables contrast adjustment. (Values for RGB contrast adjustment are set to 1.000 regardless of setting values of structure members under glcdc_contrast_t)
r	Contrast adjustment value for R signal	00h: 0/128 = 0.000 :	Specifies the contrast adjustment value for the R signal.

g	Contrast adjustment value for G signal	80h: 128/128 = 1.000 : FFh: 255/128 = 1.992	Specifies the contrast adjustment value for the G signal.
b	Contrast adjustment value for B signal		Specifies the contrast adjustment value for the B signal.

The following lists the glcdc_gamma_correction_t structure members and setting values to be referenced. When the GLCDC_CORRECTION_CMD_ GAMMA command is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.11 glcdc_gamma_correction_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
enable	Enable/disable setting of gamma correction	true	Enables gamma correction.
		false	Disables gamma correction. (Setting values of structure members under glcdc_gamma_correction_t are ignored.)
p_r	Gamma correction table for the R signal	See 5.2 Calculating Gamma Correction Value.	Specifies a gain and start threshold value for each R signal area.
p_g	Gamma correction table for the G signal	See 5.2 Calculating Gamma Correction Value.	Specifies a gain and start threshold value for each G signal area.
p_b	Gamma correction table for the B signal	See 5.2 Calculating Gamma Correction Value.	Specifies a gain and start threshold value for each B signal area.

Return Values

GLCDC_SUCCESS	*/* Processing has been completed successfully. */*
GLCDC_ERR_INVALID_PTR	*/* The p_args parameter is NULL pointer. */*
GLCDC_ERR_INVALID_ARG	*/* The argument set is invalid. */*
GLCDC_ERR_INVALID_MODE	*/* Function cannot be executed in this mode. */*
GLCDC_ERR_NOT_OPEN	*/* R_GLCDC_Open has not been executed. */*
GLCDC_ERR_INVALID_UPDATE_TIMING	*/* Update timing of the register is invalid. */*
GLCDC_ERR_INVALID_GAMMA_SETTING	*/* Gamma correction setting is invalid. */*

Properties

Prototyped in file “r_glcdc_rx_if.h”

Description

This function changes settings for brightness, contrast, and gamma correction of the GLCDC. The setting to be changed is determined according to the first parameter of this function.
This function can be executed when the mode is 'GLCDC_STATE_DISPLAYING'. The mode remains unchanged after processing for this command is complete.

Reentrant

No.

Example

```
/* Changes settings for all items */
volatile glcdc_err_t ret_glcdc;
glcdc_correction_t correction_cfg;

correction_cfg.brightness.enable = true;
correction_cfg.brightness.r = 0x200;
correction_cfg.brightness.g = 0x200;
correction_cfg.brightness.b = 0x200;

correction_cfg.contrast.enable = true;
correction_cfg.contrast.r = 0x80;
correction_cfg.contrast.g = 0x80;
correction_cfg.contrast.b = 0x80;

correction_cfg.gamma.enable = true;
correction_cfg.gamma.p_r = (gamma_correction_t *)&g_gamma_table;
correction_cfg.gamma.p_g = (gamma_correction_t *)&g_gamma_table;
correction_cfg.gamma.p_b = (gamma_correction_t *)&g_gamma_table;

ret_glcdc = R_GLCDC_ColorCorrection(GLCDC_CORRECTION_CMD_SET_ALL,
                                    (void *)&correction_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

```
/* Changes the setting for brightness adjustment */
volatile glcdc_err_t ret_glcdc;
glcdc_brightness_t brightness_cfg;

brightness_cfg.enable = true;
brightness_cfg.r = 0x200;
brightness_cfg.g = 0x200;
brightness_cfg.b = 0x200;

ret_glcdc = R_GLCDC_ColorCorrection(GLCDC_CORRECTION_CMD_BRIGHTNESS,
                                    (void *)&brightness_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

```
/* Changes the setting for contrast adjustment */
volatile glcdc_err_t ret_glcdc;
glcdc_contrast_t contrast_cfg;

contrast_cfg.enable = true;
contrast_cfg.r = 0x80;
contrast_cfg.g = 0x80;
contrast_cfg.b = 0x80;

ret_glcdc = R_GLCDC_ColorCorrection(GLCDC_CORRECTION_CMD_CONTRAST,
                                    (void *)&contrast_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

```
/* Changes the setting for gamma correction */
volatile glcdc_err_t ret_glcdc;
glcdc_gamma_correction_t gamma_cfg;

gamma_cfg.enable = true;
gamma_cfg.p_r = (gamma_correction_t *)&g_gamma_table;
gamma_cfg.p_g = (gamma_correction_t *)&g_gamma_table;
gamma_cfg.p_b = (gamma_correction_t *)&g_gamma_table;

ret_glcdc = R_GLCDC_ColorCorrection(GLCDC_CORRECTION_CMD_GAMMA,
                                    (void *)&gamma_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

Special Notes:
None.

R_GLCDC_ClutUpdate ()

This function updates the CLUT memory of the GLCDC.

Format

```
glcdc_err_t R_GLCDC_ClutUpdate (
    glcdc_frame_layer_t frame          /* Graphics screen to change operation */
    glcdc_clut_cfg_t const * const p_clut_cfg  /* Pointer to the CLUT memory */
                                               /* structure */
)
```

Parameters

glcdc_frame_layer_t   frame
  Graphics screen to change operation
glcdc_clut_cfg_t   p_clut_cfg
  Pointer to the CLUT memory structure

The following lists the glcdc_clut_cfg_t structure members and setting values to be referenced.
When this function is executed, only parameters listed below are referenced and the other parameters do not need to be specified.

Table 3.12   glcdc_clut_cfg_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
enable	Enable/disable setting of CLUT memory	true	Update CLUT memory.
		false	Not update CLUT memory, if this function is executed.
p_base	Pointer to the start address of the CLUT memory	Other than NULL	Reads the value at the address designated by the pointer and copies it to the CLUT memory.
start	Start entry number of the CLUT memory to be updated	0 to 255 (start + size < 257)	Starts updating the CLUT memory from the entry number specified.
size	Entry size of the CLUT memory to be updated	1 to 256 (start + size < 257)	Updates the CLUT memory for the specified size.

Return Values

GLCDC_SUCCESS	*/* Processing has been completed successfully. */*
GLCDC_ERR_INVALID_PTR	*/* The p_clut_cfg parameter is NULL pointer. */*
GLCDC_ERR_INVALID_ARG	*/* The argument set is invalid. */*
GLCDC_ERR_INVALID_MODE	*/* Function cannot be executed in this mode. */*
GLCDC_ERR_NOT_OPEN	*/* R_GLCDC_Open has not been executed. */*
GLCDC_ERR_INVALID_UPDATE_TIMING	*/* Update timing of the register is invalid. */*
GLCDC_ERR_INVALID_CLUT_ACCESS	*/* CLUT memory setting is invalid. */*

Properties

Prototyped in file “r_glcdc_rx_if.h”

Description

This function updates the CLUT memory of the GLCDC.
This function can be executed when the mode is 'GLCDC_STATE_DISPLAYING'. The mode remains unchanged after processing in this function is complete.

Reentrant

No.

Example

```
/* Updates all the CLUT memory for graphics 1 */
volatile glcdc_err_t ret_glcdc;
glcdc_clut_cfg_t clut_cfg;

clut_cfg.enable = true;
clut_cfg.p_base = (uint32_t *)g_gr_clut_table;
clut_cfg.size = 256;
clut_cfg.start = 0;

ret_glcdc = R_GLCDC_ClutUpdate(GLCDC_FRAME_LAYER_1, &clut_cfg);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

Special Notes:

None.

R_GLCDC_GetStatus ()

This function obtains the GLCDC status.

Format

```
glcdc_err_t R_GLCDC_GetStatus (
      glcdc_status_t * const p_status /* Pointer to the structure which */
                                      /* stores the obtained status. */
)
```

Parameters

*glcdc_status_t * const   p_status*
   Pointer to the structure which stores the obtained status.

Table 3.13   glcdc_status_t Structure Members and Setting Values

Structure Member	Outline	Setting Value	Description
state	Transition status of the GLCDC FIT module	GLCDC_STATE_NOT_ DISPLAYING	GLCDC stopped.
		GLCDC_STATE_ DISPLAYING	GLCDC operating.
state_vpos	Detection status of graphics 2 specified line notification	GLCDC_NOT_ DETECTED	Not detected.
		GLCDC_DETECTED	Detected.
state_gr1uf	Detection status of graphics 1 underflow	GLCDC_NOT_ DETECTED	Not detected.
		GLCDC_DETECTED	Detected.
state_gr2uf	Detection status of graphics 2 underflow	GLCDC_NOT_ DETECTED	Not detected.
		GLCDC_DETECTED	Detected.
fade_status	Fading status of graphics 1 and 2	GLCDC_FADE_STATUS_ NOT_UNDERWAY	Fade-in/fade-out being stopped.
		GLCDC_FADE_STATUS_ FADING_UNDERWAY	Fade-in/fade-out being executed.
		GLCDC_FADE_STATUS_ UNCERTAIN	Register value being specified for the graphics.

Return Values

GLCDC_SUCCESS	*/* Processing has been completed successfully. */*
GLCDC_ERR_INVALID_PTR	*/* The p_status parameter is NULL pointer. */*
GLCDC_ERR_NOT_OPEN	*/* R_GLCDC_Open has not been executed. */*

Properties

Prototyped in file “r_glcdc_rx_if.h”

Description

This function obtains the GLCDC status. The obtained status is written to the p_status structure passed with the argument.
This function can be called at any time after the R_GLCDC_Open function is executed. The mode remains unchanged after processing in this function is complete.

Reentrant

No.

Example

```
/* Obtains the GLCDC status */
volatile glcdc_err_t ret_glcdc;
glcdc_status_t status;

ret_glcdc = R_GLCDC_GetStatus(&status);
if (GLCDC_SUCCESS != ret_glcdc)
{
      /* error processing */
}
```

Special Notes:
None.

R_GLCDC_GetVersion ()

This function returns the current version of this API.

Format

```
uint32_t R_GLCDC_GetVersion (void)
```

Parameters

None.

Return Values

Version of this API.

Properties

Prototyped in file “r_glcdc_rx_if.h”

Description

This function will return the version of the currently running API. The version number is encoded where the top 2 bytes are the major version number and the bottom 2 bytes are the minor version number. For example, Version 4.25 would be returned as 0x00040019.

Reentrant

Yes.

Example

```
/* Gets the GLCDC FIT module version. */
volatile uint32_t version;

version = R_GLCDC_GetVersion();
```

Special Notes:

None.

4. Pin Setting

To use the GLCDC FIT module, assign input/output signals of the peripheral function to pins with the multi-function pin controller (MPC). The pin assignment is referred to as the “Pin Setting” in this document. Please perform the pin setting after calling the R_GLCDC_Open function.

When performing the Pin Setting in the e^2 studio, the Pin Setting feature of the FIT configurator or the Smart Configurator can be used. When using the Pin Setting feature, a source file is generated according to the option selected in the Pin Setting window in the FIT configurator or the Smart Configurator. Pins are configured by calling the function defined in the source file. Refer to Table 4.1 for details.

Table 4.1 Function Output by the FIT Configurator

MCU used	Function generated	Remarks
RX65N	R_GLCDC_PinSet()	

5. Using the GLCDC FIT Module

5.1 Screen Definition

In the GLCDC FIT module, reference points, the active display area, and the display start position for each screen are determined based on the parameter values of functions R_GLCDC_Open and R_GLCDC_LayerChange. Specify the arguments referencing Figure 5.1 Screen Definition and Table 5.1 Arguments and Available Setting Values.

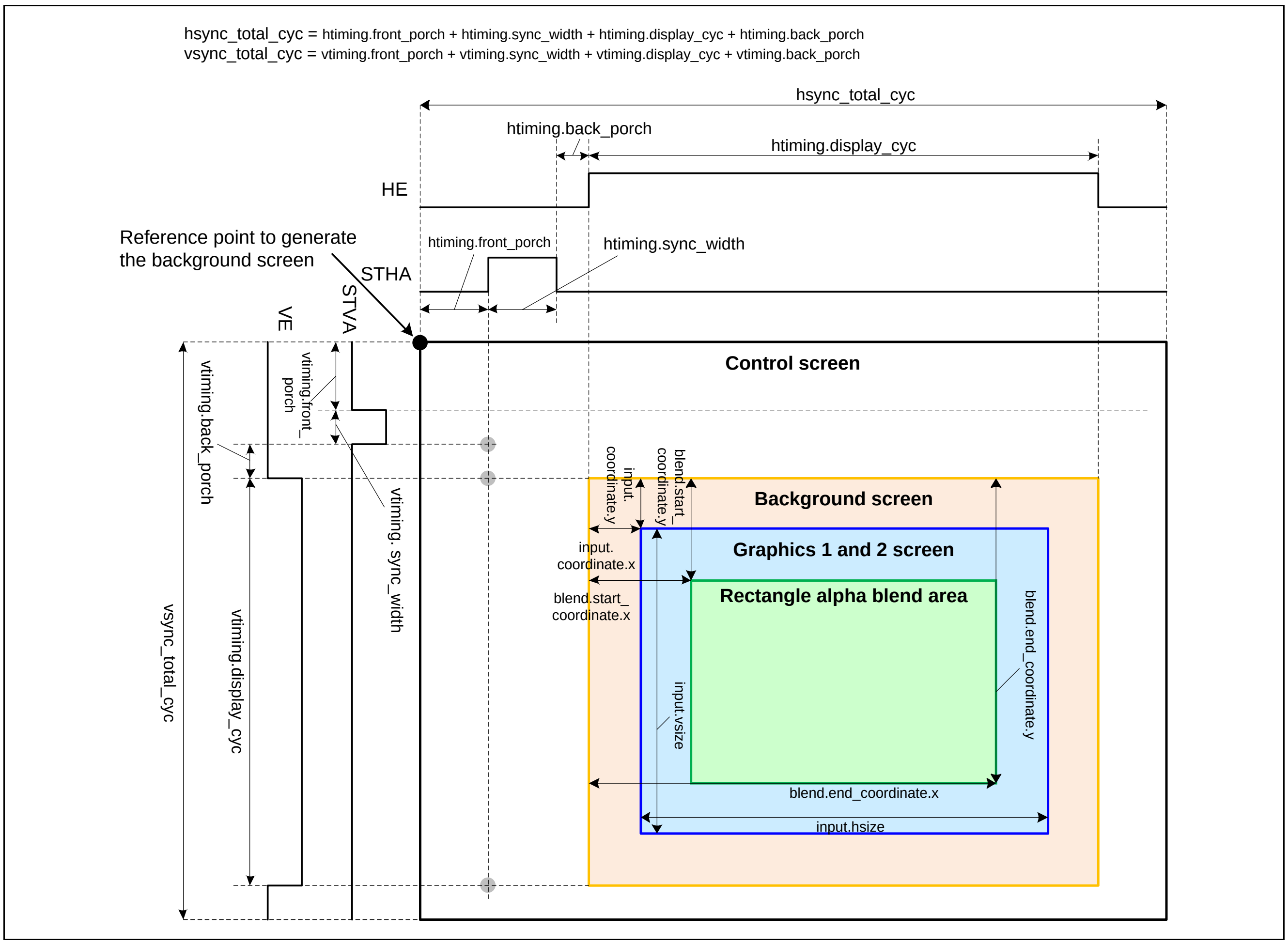

Figure 5.1 Screen Definition

Table 5.1 Arguments and Available Setting Values

Argument Name	Setting Value	Remarks
htiming.front_porch	2 < htiming.front_porch < 18	Specify a value in the range *23 < hsync_total_cyc < 1025* where *hsync_total_cyc = (htiming.front_porch + htiming.back_porch + htiming.display_cyc + htiming.sync_width)*. Specify a value in the range *5 < ((htiming.front_porch - 2) + htiming.back_porch + htiming.sync_width).* When using dithering with 2x2 pattern, specify htiming.display_cyc to a multiple of 4.
htiming.back_porch	0 < htiming.back_porch	
htiming.display_cyc	15 < htiming.display_cyc	
htiming.sync_width	0 ≤ htiming.sync_width	
vtiming.front_porch	1 < vtiming.front_porch < 17	Specify a value in the range *19 < vsync_total_cyc < 1025* where *vsync_total_cyc = (vtiming.front_porch + vtiming.back_porch + vtiming.display_cyc + vtiming_syncwidth)*. Specify a value in the range *2 < ((vtiming.front_porch - 1) + vtiming.back_porch + vtiming.sync_width).* When using dithering with 2x2 pattern, specify vtiming.display_cyc to a multiple of 2.
vtiming.back_porch	0 < vtiming.back_porch	
vtiming.display_cyc	15 < vtiming.display_cyc	
vtiming.sync_width	0 ≤ vtiming.sync_width	
input.hsize	15 < input.hsize < (htiming.display_cyc + 1)	Specify an even value.
input.coordinate.x	0 ≤ input.coordinate.x < (htiming.display_cyc - 15)	Specify a value in the range *(input.coordinate.x + input.hsize) < (htiming.display_cyc + 1)*.
input.vsize	15 < input.vsize < (vtiming.display_cyc + 1)	
input.coordinate.y	0 ≤ input.coordinate.y < (vtiming.display_cyc - 15)	Specify a value in the range *(input.coordinate.y + input.vsize) < (vtiming.display_cyc + 1)*.
blend.start_coordinate.x	0 ≤ blend.start_coordinate.x < blend.end_coordinate.x < htiming.display_cyc and 0 ≤ blend.start_coordinate.x < blend.end_coordinate.x < 1017	Specify a value in the range *(htiming.back_porch + htiming.sync_width + blend.start_coordinate.x) < 1006*. If use horizontal range between *100* and *200*, set *blend.start_coordinate.x* to *100* and *blend.end_coordinate.x* to *(200 + 1)*.
blend.end_coordinate.x		
blend.start_coordinate.y	0 ≤ blend.start_coordinate.y < blend.end_coordinate.y < vtiming.display_cyc and 0 ≤ blend.start_coordinate.y < blend.end_coordinate.y < 1021	Specify a value in the range *(vtiming.back_porch + vtiming.sync_width + blend.start_coordinate.y) < 1007*. If use vertical range between *100* and *200*, set *blend.start_coordinate.y* to 100 and *blend.end_coordinate.y* to (200 + 1).
blend.end_coordinate.y		

5.2 Calculating Gamma Correction Value

This section describes how to calculate a gamma correction value in the GLCDC FIT module.

By using the gamma correction feature in the GLCDC FIT module, brightness of the LCD panel can be adjusted based on the characteristic of the panel used. To perform gamma correction properly, specify a gain value to the GAMxLUTn register (n =1 to 8) and a threshold value of the area to the GAMxAREAn register (n = 1 to 5).

An example below describes calculation of the gain value for each area.

$$Dout = \left(\frac{Din}{pixel}\right)^{\frac{1}{\gamma}} \times pixel$$

In the above calculation formula, γ is gamma, *pixel* is the number of pixels, *Din* is a brightness value before correction, and *Dout* is a brightness value after correction. Note that the GLCDC calculates I/O signal with 10 bits. Thus *pixel* becomes 1023.

For example, if the width is set to 64 for each area and the gamma value γ is 0.7, then, when *Din* is 0 *Dout* becomes 0, and when *Din* is 64 *Dout* becomes 19.512.

$$gain = \frac{Dout_{m+1} - Dout_m}{width} \quad (m = 0\ to\ 15)$$

In the above calculation formula, *Dout(m+1)* is a brightness value after correction for area 1, *Dout(m)* is a brightness value after correction for area 0, and *width* is the width of the area 0 when calculating the gain value of the area 0.

The gain value for area 0 becomes 0.304875 with the formula above. The gain value set to the register for area 0 becomes “0.304875 × 1024 = 312 (rounding off one decimal place)”. Repeat the procedure above for 16 areas and configure the gamma correction table.

Set the threshold for setting the width of each area to be *TH(k)* < *TH(k+1)*. However, only in case of *TH(k) = 0x3FF*, it can be *TH(k) = TH(k+1)*.

An example below shows configuring the gamma correction table with each gamma correction value.

```
/* Gamma correction table when the gamma correction value is 0.5 */
const gamma_correction_t g_gamma_table =
{
  /* gain (r = 0.5) */
  { 64, 192, 320, 448, 577, 705, 833, 961, 1089, 1217, 1345, 1473, 1602,
   1730, 1858, 1954 },
  /* threshold */
  { 64, 128, 192, 256, 320, 384, 448, 512, 576, 640, 704, 768, 832, 896, 960 }
  };
/* Gamma correction table when the gamma correction value is 0.7 */
const gamma_correction_t g_gamma_table =
{
  /* gain (r = 0.7) */
  { 312, 528, 659, 762, 849, 926, 995, 1057, 1116, 1170, 1222, 1270, 1316, 1361,
   1403, 1421 },
  /* threshold */
  { 64, 128, 192, 256, 320, 384, 448, 512, 576, 640, 704, 768, 832, 896, 960 }
};
/* Gamma correction table when the gamma correction value is 0.9 */
const gamma_correction_t g_gamma_table =
{
  /* gain (r = 0.9) */
  { 753, 873, 925, 961, 988, 1010, 1029, 1046, 1061, 1074, 1086, 1097, 1107,
  1117, 1126, 1116 },
  /* threshold */
  { 64, 128, 192, 256, 320, 384, 448, 512, 576, 640, 704, 768, 832, 896, 960 }
};
/* Gamma correction table when the gamma correction value is 1.1 */
const gamma_correction_t g_gamma_table =
{
  /* gain (r = 1.1) */
  { 1317, 1157, 1103, 1069, 1045, 1026, 1010, 997, 986, 976, 967, 959, 952, 945,
   939, 919 },
  /* threshold */
  { 64, 128, 192, 256, 320, 384, 448, 512, 576, 640, 704, 768, 832, 896, 960 }
};
  /* Gamma correction table when the gamma correction value is 1.3 */
  const gamma_correction_t g_gamma_table =
{
  /* gain (r = 1.3) */
  { 1941, 1367, 1211, 1119, 1056, 1008, 970, 938, 911, 888, 868, 850, 834, 819,
  806, 781 },
  /* threshold */
  { 64, 128, 192, 256, 320, 384, 448, 512, 576, 640, 704, 768, 832, 896, 960 }
};
```

5.3 Notes on Blending Setting

In the Show/hide setting of the image, Control setting for blending, Enable/disable setting of chroma keying, there are limitation to the combination of the setting values. The combination of setting values shows Table 5.2 Combination of Setting Values. Don't use other than combination of setting values described.

Table 5.2 Combination of Setting Values

Show/hide setting of the graphics (blend.visible)	Control setting for blending (blend.blend_control)	Enable/disable setting of chroma keying (chromakey.enable)	Display contents
false	GLCDC_BLEND_ CONTROL_NONE	false	Lower-layer graphics
false	GLCDC_BLEND_ CONTROL_PIXEL	false	Lower-layer graphics
true	GLCDC_BLEND_ CONTROL_NONE	false	Current graphics
true	GLCDC_BLEND_ CONTROL_FADEIN	true	Within rectangular area, Fade-in of current graphics and lower-layer graphics. Outside rectangular area, per-pixel alpha blending of chroma keyed current graphics and lower-layer graphics
		false	Within rectangular area, Fade-in of current graphics and lower-layer graphics. Outside rectangular area, per-pixel alpha blending of current graphics and lower-layer graphics
true	GLCDC_BLEND_ CONTROL_FADEOUT	true	Within rectangular area, Fade-out of current graphics and lower-layer graphics. Outside rectangular area, per-pixel alpha blending of chroma keyed current graphics and lower-layer graphics
		false	Within rectangular area, Fade-out of current graphics and lower-layer graphics. Outside rectangular area, per-pixel alpha blending of current graphics and lower-layer graphics
true	GLCDC_BLEND_ CONTROL_FIXED *1	true	Within rectangular area, Rectangular alpha blending of current graphics and lower-layer graphics. Outside rectangular area, per-pixel alpha blending of chroma keyed current graphics and lower-layer graphics
		false	Within rectangular area, Rectangular alpha blending of current graphics and lower-layer graphics. Outside rectangular area, per-pixel alpha blending of current graphics and lower-layer graphics

true	GLCDC_BLEND_ CONTROL_PIXEL	true	Per-pixel alpha blending of chroma keyed current graphics and lower-layer graphics
		false	Per-pixel alpha blending of current graphics and lower-layer graphics

Notes: 1. If this value is set on the graphics screen, the obtained status “fade_status” when executing R_GLCDC_GetStatus function is always “GLCDC_FADE_STATUS_FADING_UNDERWAY”.

5.4 Notes on Priority Order Setting of Internal Main Bus 2

For internal main bus 2 used by GLCDC, there is a priority order setting. After a reset is released, the order is graphic 1 > graphic 2, thus the data of graphic 1 is read first. The priority order can be set using the board support package module (BSP module). Refer to the Board Support Package Module Using Firmware Integration Technology (R01AN1685), "3.2.10 Expansion Bus Master Priority Setting" for details.

5.5 When Macro Line Offset Restrictions Cannot Be Followed

If it is not possible to observe macro line offset restrictions due to the data format or the horizontal width of the frame buffer, create an image that satisfies the macro line offset restrictions by expanding the horizontal width of the image to create a margin.

For example, the following explains how to display an image of the CLUT(8) data format and a horizontal width of 480 px of the frame buffer on the LCD. Usually, macro line offset should be set to 480 (the number of bytes per pixel x horizontal width of the image = 1 x 480). However, 480 is not a multiple of 64, which is the macro line offset restriction. Therefore, expand the image to a horizontal width of 512 pixels including margin so that the condition is satisfied, and write the expanded image to the frame buffer. After that, by setting the horizontal width (input.hsize) of the image data to 480 pixels, it is possible to display the image at any horizontal width. There will be redundancy in the frame buffer by expansion, and memory usage for that will increase.

For details, refer to the chapter on the Graphic LCD Controller (GLCDC) in the User’s Manual: Hardware of each device.

Figure 5.1 below shows an image processing example. The red line in the expanded image indicates the expanded portion.

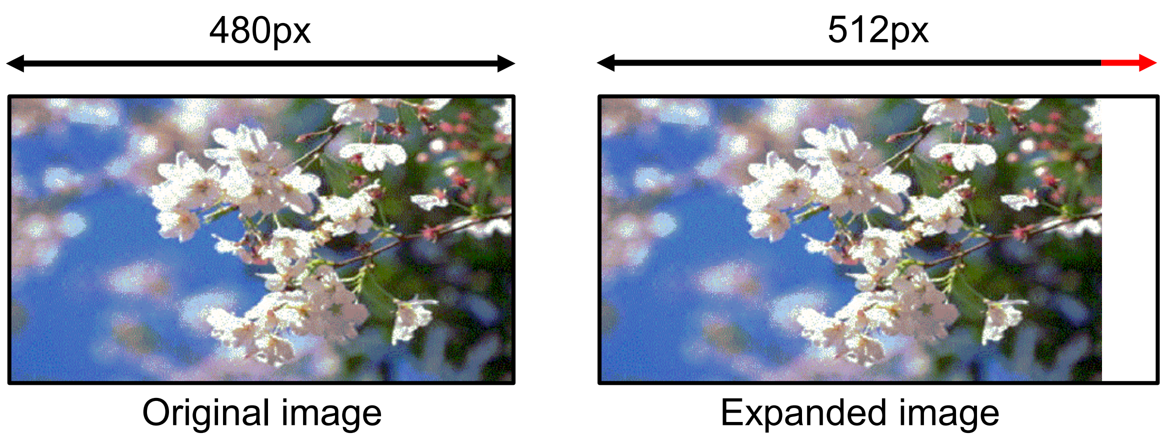

Figure 5.1 Expanded Image Sample

6. Appendices

6.1 Operation Confirmation Environment

This section describes operation confirmation environment for the GLCDC FIT module.

Table 6.1 Operation Confirmation Environment (Rev. 1.00)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 6.0.0.001
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V2.07.00
	Compiler option: The following option is added to the default settings of the integrated development environment. -lang = C99
Endian	Big endian/little endian
Revision of the module	Rev.1.00
Board used	Renesas Starter Kit+ for RX65N-2MB (product No.: RTK50565N2SxxxxxBE)

Table 6.2 Operation Confirmation Environment (Rev. 1.01)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.3.0
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00
	Compiler option: The following option is added to the default settings of the integrated development environment. -lang = C99
Endian	Big endian/little endian
Revision of the module	Rev.1.01

Table 6.3 Operation Confirmation Environment (Rev. 1.10)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.3.0 IAR Embedded Workbench for Renesas RX 4.10.1
C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00 Compiler option: The following option is added to the default settings of the integrated development environment. -lang = C99
	GCC for Renesas RX 4.8.4.201801 Compiler option: The following option is added to the default settings of the integrated development environment. -std = gnu99
	IAR C/C++ Compiler for Renesas RX Version 4.10.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev.1.10
Board used	Renesas Starter Kit+ for RX65N-2MB (product No.: RTK50565N2SxxxxxBE)

6.2 Troubleshooting

(1) Q: I have added the FIT module to the project and built it. Then I got the error: Could not open source file "platform.h".

A: The FIT module may not be added to the project properly. Check if the method for adding FIT modules is correct with the following documents:

- Using CS+:

  Application note "Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)"
- Using e^2 studio:

  Application note "Adding Firmware Integration Technology Modules to Projects (R01AN1723)"

When using a FIT module, the board support package FIT module (BSP module) must also be added to the project. For this, refer to the application note "Board Support Package Module Using Firmware Integration Technology (R01AN1685)".

(2) Q: I have added the FIT module to the project and built it. Then I got the error: This MCU is not supported by the current r_glcdc_rx module.

A: The FIT module you added may not support the target device chosen in your project. Check the supported devices of added FIT modules.

(3) Q: I have added the FIT module to the project and built it. Then I got an error for when the configuration setting is wrong.

A: The setting in the file "r_glcdc_rx_config.h" may be wrong. Check the file "r_glcdc_rx_config.h". If there is a wrong setting, set the correct value for that. Refer to 2.7 Configuration Overview.

(4) Q: I have to set three signals in this FIT module, Vsync, Hsync and DE. Does the module support Vsyntc and Hsync only, or DE only LCD?

A: By performing pin setting (MPC setting) of signals to be used, it is possible to support each LCD module.

Signals for which the pin has not been set will not be output. All signals must be allocated even though they are not used.

(5) Q: There is a line detection (VPOS interrupt) function. Let me know the line detection occurrence timing.

A: Refer to 5.1 Screen Definition.

Detection occurs when the STHA signal is asserted at the last line of the entire control screen shown in Figure 5.1 Screen Definition.

(6) Q: Images cannot be displayed as I expect.

Q-1: Image data are not displayed on LCD panel.

A-1: The pin setting may not be performed correctly. When using this FIT module, the pin setting must be performed. Refer to 4 Pin Setting for details.

Q-2: When I change the image data format (32bpp, 16bpp, 8bpp, etc.), images cannot be displayed as I expect.

A-2: Check the following parameters.

1. Data format of the frame buffer (input.format)

   Specify the data format appropriate for the image data.

2. Image horizontal width (input.hsize)

3. Macro line offset (input.offset)

Set the macro line offset (number of bytes per pixel x horizontal width) to a multiple of 64. If it is not possible to observe this restriction, refer to 5.5 When Macro Line Offset Restrictions Cannot Be Followed.

Q-3: When I set RX MCU to big endian, images cannot be displayed as I expect.
A-3: Perform endian conversion of the image data. The method of endian conversion differs depending on the data format. For details, refer to the chapter on the Graphic LCD Controller (GLCDC) in the User’s Manual: Hardware of each device.

Q-4: Image color tones are not normal.
A-4: Check that the pixel order of the frame buffer is ARGB (alpha value, red value, green value, blue value). Also, check the pixel order (output.color_order) of the output data.

7. Reference Document

User's Manual: Hardware
RX65N Group, RX651 Group User's Manual: Hardware (R01UH0590)
(The latest version can be downloaded from the Renesas Electronics website.)

Technical Update/Technical News
(The latest version can be downloaded from the Renesas Electronics website.)

User's Manual: Development Tools
[CS+][e^2 studio] RX C/C++ Compiler CC-RX User's Manual (R20UT3248)
(The latest version can be downloaded from the Renesas Electronics website.)

Related Technical Updates

This module reflects the content of the following technical updates.
None

Revision Record

Rev.	Date	Description	
		Page	**Summary**
1.00	Oct.01.2017	—	First edition issued
1.01	Feb.01.2019	52	Added Table 6.2 Operation Confirmation Environment (Rev. 1.01)
		—	Changes associated with functions: Added support setting function of configuration option Using GUI on Smart Configurator. [Description] Added a setting file to support configuration option setting function by GUI.
1.10	May.31.2019	—	Added support for GCC and IAR compilers.
		1	Added the target compiler.
		7	Changed the code size format.
		19	Added section 2.13 “for”, “while” and “do while” statements
		30	Added “Notes on macro line offset setting” to Special Notes.
		60	Added section 5.4 “Notes on Priority Order Setting of Internal Main Bus 2” and section 5.5 “When Macro Line Offset Restrictions Cannot Be Followed”
		62	Added Table 6.2 Operation Confirmation Environment (Rev. 1.10).
		63-64	Added (4) to (6) in section 6.3 Troubleshooting

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: www.renesas.com/contact/.