RX Family
LCD Module Using Firmware Integration Technology

Introduction

This document describes the LCD module using the firmware integration technology (FIT). This module uses the LCD controller/driver (LCDC) to display data on an LCD. Hereinafter this module is referred to as LCDC FIT module.

Target Device

This API supports the following devices:

- RX113 Group
- RX23E-B Group

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 "5.1 Confirmed Operation Environment".

Related Documents

- RX Family Board Support Package Module Using Firmware Integration Technology (R01AN1685)

Contents

1. Overview

1.1 LCDC FIT Module

The LCDC FIT module can be used by being implemented in a project as an API. See section 2.9 Adding the FIT Module to Your Project for details on methods to implement this FIT module into a project.

1.2 Overview of the LCDC FIT Module

This module supports the following features:

- The internal voltage boosting method, the capacitor split method, or the external resistance division method can be selected for the LCD driver voltage generator.

- The display bias method can be selected from the 1/2 bias method, 1/3 bias method, or 1/4 bias method.

- The time slice of display can be selected from static, 2-time slice, 3-time slice, 4-time slice, or 8-time slice.

- The display waveform can be selected from waveform A or waveform B.

- The display data area can be selected from A-pattern, B-pattern, or blinking. The display data area can be switched.

- The blinking display with A-pattern and B-pattern can be achieved using the RTC periodic interrupt (PRD).

- The reference voltage, which is generated when operating the voltage boost circuit, can be selected from 16 steps (contrast adjustment).

1.3 Overview of API

Table 1.1 lists the API Functions included in this module.

Table 1.1   API Functions

Function	Description
R_LCDC_Open()	Initializes the LCDC to display data on the LCD.
R_LCDC_Write()	Sets display data.
R_LCDC_Modify()	Modifies display data.
R_LCDC_DispOn()	Displays the specified data on the LCD.
R_LCDC_DispOff()	Disables display on the LCD. Stops the voltage boost circuit or the capacitor split circuit.
R_LCDC_Control()	Changes the drive voltage method and the display data area.
R_LCDC_AdjustContrast()	Adjusts the contrast.
R_LCDC_OperateCircuit()	Enables or disables the voltage boost circuit or the capacitor split circuit.
R_LCDC_Close()	Stops the LCDC.
R_LCDC_GetVersion()	Returns the version of this module.

1.4 State Transition

Figure 1.1 shows the State Transition. When entering the other state from one state, call the function shown above the state in the figure.
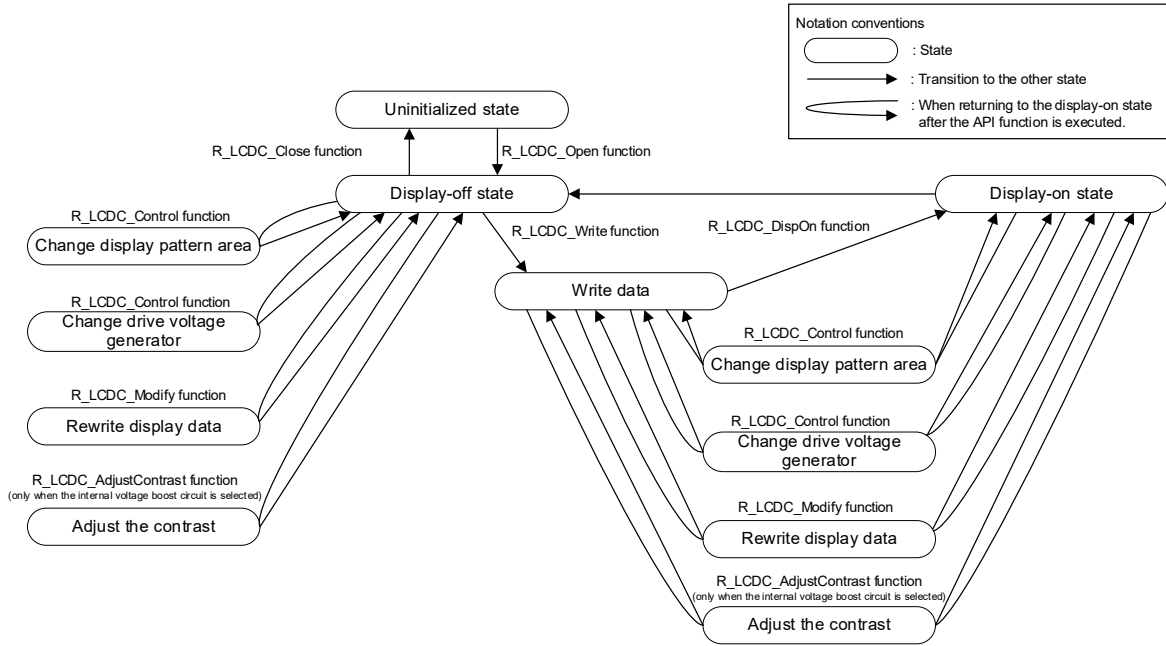

Figure 1.1 State Transition

2. API Information

This FIT module has been confirmed to operate under the following conditions.

2.1 Hardware Requirements

The MCU used must support the following functions:

- LCDC

2.2 Software Requirements

This driver is dependent upon the following FIT module:

- Renesas Board Support Package (r_bsp) v5.00 or higher

2.3 Supported Toolchain

This driver has been confirmed to work with the toolchain listed in 5.1, Confirmed Operation Environment.

2.4 Header Files

All API calls and their supporting interface definitions are located in r_lcdc_rx_if.h.

2.5 Integer Types

This project uses ANSI C99. These types are defined in stdint.h.

2.6   Configuration Overview

The configuration options in this module are specified in r_lcdc_rx_config.h. The option names and setting values are listed in the table below.

Configuration options in *r_lcdc_rx_config.h* (1/3)	

`#define`
`LCDC_CFG_PARAM_CHECKING_ENABLE`
`- Default value =`
`   BSP_CFG_PARAM_CHECKING_ENABLE`
	Select whether to include parameter checking into codes.
	- Set to 0 to omit parameter checking from codes.
	- Set to 1 to include parameter checking into codes.

`#define LCDC_CFG_DRV_GENERATOR` `- Default value = 1`	Configure the LCD drive voltage generator. - Set to 0 for the external resistance division method. - Set to 1 for the internal voltage boosting method. - Set to 2 for the capacitor split method.
`#define LCDC_CFG_VOLTAGE_WAIT` `- Default value = 1`	Select whether to processing for the voltage boosting wait time is performed in this module or in the user program. Refer to the User's Manual: Hardware for details on the voltage boosting wait time. - Set to 0 not to perform the wait processing in this module. Perform it in the user program. - Set to 1 to perform the wait processing in this module.
`#define LCDC_CFG_BIAS` `- Default value = 3`	Specify the LCD display bias method. - Set to 2 for 1/2 bias method. - Set to 3 for 1/3 bias method. - Set to 4 for 1/4 bias method.
`#define LCDC_CFG_TIME_SLICES` `- Default value = 4`	Specify the time slice of LCD display. - Set to 1 for static. - Set to 2 for 2-time slice. - Set to 3 for 3-time slice. - Set to 4 for 4-time slice. - Set to 8 for 8-time slice.
`#define LCDC_CFG_DRV_WAVEFORM` `- Default value = 0`	Specify the LCD display waveform. - Set to 0 for waveform A. - Set to 1 for waveform B.

Configuration options in *r_lcdc_rx_config.h* (2/3)	
`#define LCDC_CFG_CLOCK_DIV` `- Default value = 23`	Specifies the division of the LCD clock frequency. The division can be calculated as follows: Division = clock frequency used ÷ number of time slices ÷ frame frequency Select the closest value from the following: When the main clock, HOCO, or LOCO is selected as the LCD clock source: - Set to 17 for × 1/256 - Set to 18 for × 1/512 - Set to 19 for × 1/1024 - Set to 20 for × 1/2048 - Set to 21 for × 1/4096 - Set to 22 for × 1/8192 - Set to 23 for × 1/16384 - Set to 24 for × 1/32768 - Set to 25 for × 1/65536 - Set to 26 for × 1/131072 - Set to 27 for × 1/262144 - Set to 43 for × 1/524288 When the sub-clock is selected as the LCD clock source: - Set to 1 for × 1/4 - Set to 2 for × 1/8 - Set to 3 for × 1/16 - Set to 4 for × 1/32 - Set to 5 for × 1/64 - Set to 6 for × 1/128 - Set to 7 for × 1/256 - Set to 8 for × 1/512 - Set to 9 for × 1/1024 When the IWDT-dedicated on-chip oscillator is selected as the LCD clock source: - Set to 1 for × 1/4 - Set to 2 for × 1/8 - Set to 3 for × 1/16 - Set to 4 for × 1/32 - Set to 5 for × 1/64 - Set to 6 for × 1/128 - Set to 7 for × 1/256

Configuration options in *r_lcdc_rx_config.h* (3/3)	
`#define LCDC_CFG_REF_VCC` `- Default value = 12`	Specify the reference voltage. This constant is available only when the internal voltage boosting method is selected for the drive voltage generator. The setting value differs depending on the selected bias method (the reference voltage, 1/3 bias method, or 1/4 bias method). When the bias method is VL1 voltage (reference voltage): - Set to 4 for 1.00 V - Set to 5 for 1.05 V - Set to 6 for 1.10 V - Set to 7 for 1.15 V - Set to 8 for 1.20 V - Set to 9 for 1.25 V - Set to 10 for 1.30 V - Set to 11 for 1.35 V - Set to 12 for 1.40 V - Set to 13 for 1.45 V - Set to 14 for 1.50 V - Set to 15 for 1.55 V - Set to 16 for 1.60 V - Set to 17 for 1.65 V - Set to 18 for 1.70 V - Set to 19 for 1.75 V When the bias method is VL4 voltage (1/3 bias method): - Set to 4 for 3.00 V - Set to 5 for 3.15 V - Set to 6 for 3.30 V - Set to 7 for 3.45 V - Set to 8 for 3.60 V - Set to 9 for 3.75 V - Set to 10 for 3.90 V - Set to 11 for 4.05 V - Set to 12 for 4.20 V - Set to 13 for 4.35 V - Set to 14 for 4.50 V - Set to 15 for 4.65 V - Set to 16 for 4.80 V - Set to 17 for 4.95 V - Set to 18 for 5.10 V - Set to 19 for 5.25 V When the bias method is VL4 voltage (1/4 bias method): - Set to 4 for 4.00 V - Set to 5 for 4.20 V - Set to 6 for 4.40 V - Set to 7 for 4.60 V - Set to 8 for 4.80 V - Set to 9 for 5.00 V - Set to 10 for 5.20 V

2.7   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 2.6 Configuration Overview

The values in the table below are dconfirmed under the following conditions.

   Module Revision: r_lcdc_rx rev2.00

   Compiler Version: Renesas Electronics C/C++ Compiler Package for RX Family V3.04.00

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

   GCC for Renesas RX 8.03.00.202202

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

   IAR C/C++ Compiler for Renesas RX version 4.20.3

      (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
RX113 RX23E-B	ROM	1,035 bytes	959 bytes	1,385 bytes	1,301 bytes	1,351 bytes	1,268 bytes
	RAM	2 bytes		2 bytes		2 bytes	
	STACK	120 bytes		-		56 bytes	

2.8 Return Values

This section describes return values of API functions. This enumeration is located in r_lcdc_rx_if.h as are the prototype declarations of API functions.

```
typedef enum
{
    LCDC_ERR_ADJUST_PARAM = -2, /* R_LCDC_AdjustContrast function parameter error */
    LCDC_ERR_OUTSIDE_VOLTAGE = -1, /* Invalid voltage (outside the range) */
    LCDC_SUCCESS, /* Processing completed successfully */
    LCDC_ERR_NOT_CLOSED, /* Already locked */
    LCDC_ERR_NOT_OPEN, /* R_LCDC_Open function not executed */
    LCDC_ERR_INVALID_ARG, /* Invalid parameter */
    LCDC_ERR_UNSUPPORTED, /* Combination of display mode not supported */
    LCDC_ERR_NOT_OPERATE, /* Specified clock source not operated*/
    LCDC_ERR_RTC_NOT_OPERATE /* RTC not operated with blinking display selected */
} lcdc_err_t;
```

2.9 Adding the FIT Module to Your Project

This module must be added to each project in which it is used. Renesas recommends the method using the Smart Configurator described in (1) or (3) or (5) below. However, the Smart Configurator only supports some RX devices. Please use the methods of (2) or (4) for RX devices that are not supported by the Smart Configurator.

(1) Adding the FIT module to your project using the 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 "RX Smart Configurator User's Guide: e^2 studio (R20AN0451)" for details.

(2) Adding the FIT module to your project using the 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 "RX Family Adding Firmware Integration Technology Modules to Projects (R01AN1723)" for details.

(3) Adding the FIT module to your project using the Smart Configurator in CS+
By using the Smart Configurator Standalone version in CS+, the FIT module is automatically added to your project. Refer to "RX Smart Configurator User's Guide: CS+ (R20AN0470)" 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 "RX Family Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)" for details.

(5) Adding the FIT module to your project using the Smart Configurator in IAREW
By using the Smart Configurator Standalone version, the FIT module is automatically added to your project. Refer to "RX Smart Configurator User's Guide: IAREW (R20AN0535)" for details.

2.10   "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".

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_LCDC_Open()

This function initializes the LCDC to get it ready for displaying data on the LCD.

Format
lcdc_err_t R_LCDC_Open(void)

Parameters
None

Return Values
LCDC_SUCCESS, / Processing completed successfully */*
LCDC_ERR_NOT_CLOSED, / Already locked */*
LCDC_ERR_UNSUPPORTED, / Combination of display mode not supported */*
LCDC_ERR_NOT_OPERATE, / Specified clock source not operated */*

Properties
Prototyped in r_lcdc_rx_if.h.

Description
With the LCDC not locked, the initial settings listed below are performed for using the LCDC.

- Locks the LCDC.
- Cancels the module-stop state for the LCDC.
- Specifies the voltage boosting pin initial value switching control bit (only when the internal voltage boosting method is selected for the drive voltage generator).
- Specifies the bias method, number of time slices, and waveform.
- Configures the drive voltage generator.
- Specifies the display data area (default is A-pattern).
- Specifies the LCD clock source.
  Use the BSP_CFG_LCD_CLOCK_SOURCE constant in the BSP module to select the clock source.
- Specifies the voltage level (only when the internal voltage boosting method is selected for the drive voltage generator).
- Enables the voltage boost circuit or capacitor split circuit (only when the internal voltage boosting method or capacitor split method is selected for the drive voltage generator).

Perform the following before executing this function.
- Enable the LCD clock source oscillation.
- Specify the segment/common pin, and pins VL1, VL2, VL3, VL4, CAPH, and CAPL by the PmnPFS.PSEL[4:0] bits. (Set registers PMR, PDR, and PCR to 0 before setting these pins, then set the PMR register to 1 after setting these pins.)

Example
```
lcdc_err_t lcdc_err;
lcdc_err = R_LCDC_Open();
```

Special Notes

Configure the drive voltage generator, the display waveform, the number of time slices, and the bias method in r_lcdc_rx_config.h. Use the combination of settings listed below when configuring.

Display Mode			Drive Voltage Generator		
Display Waveform	Number of Time Slices	Bias Method	External Resistance Division	Internal Voltage Boosting	Capacitor Split
A-pattern	8	1/4	Available	Available	N/A
A-pattern	4	1/3	Available	Available	Available
A-pattern	3	1/3	Available	Available	Available
A-pattern	3	1/2	Available	N/A	N/A
A-pattern	2	1/2	Available	N/A	N/A
A-pattern	Static		Available	N/A	N/A
B-pattern	8	1/4	Available	Available	N/A
B-pattern	4	1/3	Available	Available	Available

R_LCDC_Write()

This function specifies display data.

Format

lcdc_err_t R_LCDC_Write(

uint8_t seg, /* Specifies the segment to be written. */

uint8_t data /* Data to be written. */

)

Parameters

seg

Specify the segment to be written.

data

Specify display data to be written to the specified segment.

Return Values

LCDC_SUCCESS, / Processing completed successfully */*
LCDC_ERR_INVALID_ARG, / Invalid parameter */*

Properties

Prototyped in r_lcdc_rx_if.h.

Description

Specifies the initial display data. Except for 8-time slice, store values in the lower 4 bits when writing to the A-pattern area, and in the upper 4 bits when writing to the B-pattern area.

- The display data is stored in the display data register.

Example

When displaying '2' with 4-time slice.

```
volatile lcdc_err_t lcdc_err;

lcdc_err = R_LCDC_Write(10, 0x06);
lcdc_err = R_LCDC_Write(11, 0x0b);
```

Special Notes

None

R_LCDC_Modify()

This function is used when rewriting the display data.

Format
lcdc_err_t R_LCDC_Modify(
        uint8_t seg, /* Display data */
        uint8_t data_mask, /* Specifies where to rewrite. */
        uint8_t data /* Data to rewrite */
)

Parameters
seg

    Specify the segment to be written.

data_mask

    Set 0 to the bit to be rewritten and set 1 to the other bits. To rewrite only 1 bit, refer to the table below.
    Multiple bits can be rewritten.

COM	COM3 (bit 3)	COM2 (bit 2)	COM1 (bit 1)	COM0 (bit 0)
Setting value of data_mask	0xf7	0xfb	0xfd	0xfe

data

    Specify display data to rewrite to the specified segment.

    Set "0" to bits that are not to be rewritten (bits set to "1" in data_mask).

Return Values
LCDC_SUCCESS, / Processing completed successfully */*
LCDC_ERR_INVALID_ARG, / Invalid parameter */*

Properties
Prototyped in r_lcdc_rx_if.h.

Description
Rewrites the LCD display data in 1-bit units. If a bit is not specified for rewriting, the value stored in the bit is held as it is. Before executing this function, execute the R_LCDC_Open function.

- Masks the data being displayed.
- Specifies the data to rewrite.

Example
When modifying COM0 (0 bit) and COM1 (1 bit) of SEG0.

```
lcdc_err_t lcdc_err;
lcdc_err = R_LCDC_Modify(0,0xfc, 0x03);
```

Special Notes
None

R_LCDC_DispOn()

This function specifies display data and displays the data on the LCD.

Format
lcdc_err_t R_LCDC_DispOn(void)

Parameters
None

Return Values
LCDC_SUCCESS, / Processing completed successfully */*
LCDC_ERR_NOT_OPEN, / R_LCDC_Open function not executed */*

Properties
Prototyped in r_lcdc_rx_if.h.

Description
Displays the specified data on the LCD. Before executing this function, execute the R_LCDC_Open function.

- Enables display on the LCD.

Example
```
volatile lcdc_err_t lcdc_err;

/* Configures the LCD controller */
lcdc_err = R_LCDC_Write(10, 0x06);
lcdc_err = R_LCDC_Write(11, 0x0b);

/* Enables display */
lcdc_err = R_LCDC_DispOn();
```

Special Notes
None

R_LCDC_DispOff()

This function disables display on the LCD, and stops the internal voltage boost circuit or capacitor split circuit.

Format

lcdc_err_t R_LCDC_DispOff(

uint8_t stop_select /* Selects the operation whether to disable display only or also stop the circuit. */

)

Parameters

stop_select

Select the operation, according to the specifications of the user system, whether to disable display on the LCD only or also stop the voltage boost/capacitor split circuit.

- LCDC_ONLY_DISPLAY_OFF: Only display on the LCD is disabled and the circuit is not stopped.

- LCDC_CIRCUIT_STOP: Display on the LCD is disabled and also the circuit is stopped.

This parameter can be selected only when the internal voltage boost method or capacitor split method is selected for the drive voltage generator.

The table below lists current consumption and time required for redisplay for each drive voltage method.

Argument	Drive Voltage Method	Consumption Current	Time for redisplay
LCDC_ONLY_DISPLAY_ OFF: Display is disabled.	Any methods	Reduced less consumption current than LCDC_CIRCUIT_STOP	None
LCDC_CIRCUIT_STOP: Display is disabled and the circuit is stopped.	External resistance division [1]	The result obtained is the same as above. [1]	The result obtained is the same as above. [1]
	Internal voltage boosting	Reduced more consumption current than LCDC_ONLY_DISPLAY_OFF	Reference voltage setup time [2] + voltage boosting wait time
	Capacitor split method	Reduced more consumption current than LCDC_ONLY_DISPLAY_OFF	Voltage boosting wait time [2]

Notes:

1. LCDC_CIRCUIT_STOP is not available for the external resistance division method.
2. Refer to the User's Manual: Hardware for details on the reference voltage setup time and the voltage boosting wait time.

Return Values

LCDC_SUCCESS, / Processing completed successfully */*
LCDC_ERR_INVALID_ARG, / Invalid parameter */*

Properties

Prototyped in r_lcdc_rx_if.h.

Description
Performs processing to disable display on the LCD only, or processing to disable display on the LCD and stop the internal voltage boost/capacitor split circuit.
When the circuit is stopped, the drive voltage method is switched to the external resistance division method.
When operating the circuit again, execute the R_LCDC_OperateCircuit function that switches the drive voltage method back to the method previously used, then execute the R_LCDC_DispOn function.

Example
```
lcdc_err_t lcdc_err;
lcdc_err = R_LCDC_DispOff(LCD_ONLY_DISPLAY_OFF);
```

Special Notes
None

R_LCDC_OperateCircuit()

This function enables or disables the internal voltage boost/capacitor split circuit.

Format

lcdc_err_t R_LCDC_OperateCircuit(
        uint8_t select_operation

Parameters

select_operation

Select the operation whether to enable or disable the internal voltage boost/capacitor split circuit.

- LCDC_CIRCUIT_ENABLE: The internal voltage boost circuit or capacitor split circuit is enabled.

- LCDC_CIRCUIT_DISABLE: The internal voltage boost circuit or capacitor split circuit is disabled.

Return Values

LCDC_SUCCESS, / Processing completed successfully */*
LCDC_ERR_INVALID_ARG, / Invalid parameter */*

Properties

Prototyped in r_lcdc_rx_if.h.

Description

When the internal voltage boost method or capacitor split method is selected for the drive voltage generator, this function enables or disables the internal voltage boost/capacitor split circuit.

After the circuit is enabled, the voltage boosting wait time must be processed. The user can select whether the voltage boosting wait time is processed in this function or in the user program by specifying the LCDC_CFG_VOLTAGE_WAIT constant.

When the circuit is disabled, the drive voltage method is switched to the external resistance division method after the current circuit is stopped. When operating the circuit again, the drive voltage method is switched back to the method previously used.

Example

Here shows an example to disable display on the LCD and stop the circuit with the R_LCDC_DispOff function, and then operate the circuit again:

```
volatile lcdc_err_t lcdc_err;

lcdc_err = R_LCDC_DispOff(LCDC_CIRCUIT_STOP); /* Display disabled and
                                              the circuit stopped */
lcdc_err = R_LCDC_OperateCircuit(LCDC_CIRCUIT_ENABLE); /* Circuit enabled */
```

Special Notes

None

R_LCDC_Control()

This function changes the drive voltage method and the display data area.

Format

lcdc_data_t R_LCDC_Control(

      uint8_t config_pattern, /* Selects what to be changed */

      uint8_t select_drv_gen, /* When changing the drive voltage method, selects the way to change */

      uint8_t select_display_area /* When changing the display data area, selects the way to change */

)

Parameters

config_pattern

Specify whether to change either the drive voltage method or display data area, or both. When changing both, use OR (|) to specify multiple items.

- LCDC_FIX_DRV_GENERETOR: The drive voltage method is changed.

- LCDC_FIX_DISPLAY_AREA: The display data area is changed.

select_drv_gen

When "LCDC_FIX_DRV_GENERETOR" is selected for the first parameter, select one of the arguments below. When "LCDC_FIX_DRV_GENERETOR" is not selected, set "FIT_NO_PTR" for this parameter.

- LCDC_DRV_INTERNAL_TO_CAP: The method is changed from the internal voltage boosting method to the capacitor split method.

- LCDC_DRV_CAP_TO_INTERNAL: The method is changed from the capacitor split method to the internal voltage boosting method.

select_display_area

When "LCDC_FIX_DISPLAY_AREA" is selected for the first parameter, specify the new display data area by selecting one of the arguments below. When "LCDC_FIX_DISPLAY_AREA" is not selected for the first parameter, set "FIT_NO_PTR" for this parameter.

- LCDC_PATTERN_A: The display data area is changed from B-pattern to A-pattern.

- LCDC_PATTERN_B: The display data area is changed from A-pattern to B-pattern.

- LCDC_BLINKING: The display data area is changed to blinking with A-pattern and B-pattern.

Return Values

LCDC_SUCCESS, / Processing completed successfully */*
LCDC_ERR_INVALID_ARG, / Invalid parameter */*
LCDC_ERR_RTC_NOT_OPERATE / RTC not operated with blinking display selected */*

Properties

Prototyped in r_lcdc_rx_if.h.

Description

The drive voltage method or/and the display data area can be changed. The first parameter specifies which to be changed. Also both the method and the area can be selected.

When changing the drive voltage method:

The drive voltage method can be changed from the internal voltage boosting method to the capacitor split method, or vice versa. The following is the procedure to change the drive voltage method.

- Disable display on the LCD (R_LCDC_DispOff function).

- Change the drive voltage method according to the second parameter.

- Enable display on the LCD (R_LCDC_DispOn function).

When changing the display data area:

The display data area can be changed from the current setting to A-pattern, B-pattern, or blinking.

When using blinking, the RTC is required to operate before this function is executed. To configure the RTC, follow the steps below.

1) Configure the clock oscillation used for the RTC. Refer to the User's Manual: Hardware for the detailed procedure.

2) Configure the RTC settings as listed below. Refer to the User's Manual: Hardware for the detailed procedure.

   - Periodic interrupt request: Enabled

   - Periodic interrupt cycle: 1/2 second

Example

```
lcdc_err_t lcdc_err;
uint8_t cfg_pattern;
uint8_t sel_display_area;

/* Specifies what to be changed */
cfg_pattern = LCDC_FIX_DISPLAY_AREA; /* Display data area is changed. */
sel_display_area = LCDC_PATTERN_B; /* The display data area is changed to B-pattern. */

lcdc_err = R_LCDC_Control(cfg_pattern, 0, cfg_display_area);
```

Special Notes

None

R_LCDC_AdjustContrast()

This function adjusts the contrast.

This function can be selected when the internal voltage boosting method is used for the drive voltage generator.

Format

int8_t R_LCDC_AdjustContrast (

uint8_t select_read_adjust, /* Selects whether to read or change the voltage level. */

int8_t updown_level /* Voltage step to change */

Parameters

select_read_adjust

Specify one of the following to this parameter.

- LCDC_VOL_READ: The current voltage level is read.

- LCDC_VOL_ADJUST: The voltage is changed to adjust the contrast.

updown_level

When "LCDC_VOL_READ" is selected for the first parameter, set this argument to 0.

When "LCD_VOL_ADJUST" is selected, set the voltage step to change from -15 to 15. For example, set '3 ' to specify the voltage level that is 3 steps higher than the current level. Set '-1 ' to specify the voltage level that is 1 step lower than the current level.

Return Values

LCDC_VOL_LEVEL, / Current voltage level returned when processing completed successfully.*
*   If the voltage level is changed, then the changed level is returned. */*
LCDC_ERR_OUTSIDE_VOLTAGE, / Invalid voltage (outside the range) */*
LCDC_ERR_ADJUST_PARAM   / Function parameter error */*

Properties

Prototyped in r_lcdc_rx_if.h.

Description

The voltage level is read or the voltage level is adjusted (contrast adjustment) according to the first parameter.

When LCDC_VOL_READ is selected for the first parameter:

The current voltage level is returned as the return value.

When LCDC_VOL_ADJUST is selected for the first parameter:

The current voltage level is changed according to the value passed by the first parameter. The following is the procedure to change the voltage level.

- Verify the specified level is within the selectable range.

- Stop the internal voltage boost/capacitor split circuit.

- Change the voltage level according to the specified value in the parameter.

- Enable the internal voltage boost/capacitor split circuit.

- Wait for the reference voltage setup wait time.

The table below lists the available range for the voltage level.

VL1 Voltage (Reference Voltage)	VL4 Voltage (1/3 Bias Method)	VL4 Voltage (1/4 Bias Method)
1.00 V	3.00 V	4.00 V
1.05 V	3.15 V	4.20 V
1.10 V	3.30 V	4.40 V
1.15 V	3.45 V	4.60 V
1.20 V	3.60 V	4.80 V
1.25 V	3.75 V	5.00 V
1.30 V	3.90 V	5.20 V
1.35 V	4.05 V	—
1.40 V	4.20 V	—
1.45 V	4.35 V	—
1.50 V	4.50 V	—
1.55 V	4.65 V	—
1.60 V	4.80 V	—
1.65 V	4.95 V	—
1.70 V	5.10 V	—
1.75 V	5.25 V	—

Example

```
volatile lcdc_err_t lcdc_err;

/* When changing the voltage from 4.20 to 3.75 V (3 steps) with the 1/3 bias method. */
lcdc_err = R_LCDC_AdjustContrast(LCDC_VOL_ADJUST, -3);
```

Special Notes

None

R_LCDC_Close()

This function stops the LCDC.

Format
lcdc_err_t R_LCDC_Close(void)

Parameters
None

Return Values
LCDC_SUCCESS, / Processing completed successfully */*

Properties
Prototyped in r_lcdc_rx_if.h.

Description
The LCDC is stopped. Make sure to stop the voltage boost/capacitor split circuit before executing this function.

The following is the procedure to stop the LCDC.

- Initializes the values in the LCD registers and flags.

- Enters the state to stop the LCD clock source.

- Enters the LCDC module-stop state.

- Unlocks the LCDC.

Example
```
volatile lcdc_err_t lcdc_err;

lcdc_err = R_LCDC_Close();
```

Special Notes
None

R_LCDC_GetVersion()

This function returns the API version.

Format

uint32_t R_LCDC_GetVersion(void)

Parameters

None

Return Values

Version number

Properties

Prototyped in r_lcdc_rx_if.h.

Description

Returns the API version number.

Example

```
uint32_t version;

version = R_LCDC_GetVersion ();
```

Special Notes

None

4. Pin Setting

To use the LCDC 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 before calling the R_LCDC_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. Then 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 to be Output	Remarks
RX113 RX23E-B	R_LCDC_PinSet()	

5. Appendices

5.1 Confirmed Operation Environment

This section describes confirmed operation environment for the LCD FIT module.

Table 5.1 Confirmed Operation Environment (Rev. 1.01)

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. 1.01
Board used	Renesas Starter Kit for RX113 (product No.: R0K505113S000BE)

Table 5.2 Confirmed Operation Environment (Rev. 1.02)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio Version 7.7.0 IAR Embedded Workbench for Renesas RX 4.13.1

C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.02.00
	   Compiler option: The following option is added to the default settings of the integrated development environment.
	   -lang = c99

	GCC for Renesas RX 8.03.00.201904
	   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 4.13.1 Compiler option: The default settings of the integrated development environment.
Endian	Big endian/little endian
Revision of the module	Rev. 1.02
Board used	Renesas Starter Kit for RX113 (product No.: R0K505113xxxxxx)

Table 5.3   Confirmed Operation Environment (Rev. 2.00)

Item	Contents
Integrated development environment	Renesas Electronics e^2 studio 2022-10 (22.10.0) IAR Embedded Workbench for Renesas RX 4.20.3

C compiler	Renesas Electronics C/C++ Compiler Package for RX Family V3.04.00
	   Compiler option: The following option is added to the default settings of the
	   integrated development environment.
	   -lang = c99

	GCC for Renesas RX 8.03.00.202202
	   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 4.20.3
	   Compiler option: The default settings of the integrated development
	   environment.

Endian	Big endian/little endian
Revision of the module	Rev. 2.00
Board used	Renesas Solution Starter Kit for RX23E-B (product No.: RTK0ES1001C00001BJ)

RENESAS

5.2   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 this FIT module, the board support package FIT module (BSP module) must also be added to the project. 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_lcdc_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 the error: Parameter error in configures file.

   A: The setting in the file "r_lcdc_rx_config.h" may be wrong. Check the file "r_lcdc_rx_config.h". If there is a wrong setting, set the correct value for that. Refer to 2.6 Configuration Overview Overview for details.

(4) Q: Text is not displayed on the LCD panel.

   A: 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.

6. Reference Documents

User's Manual: Hardware
The latest versions can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
The latest information can be downloaded from the Renesas Electronics website.

User's Manual: Development Tools
RX Family 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 History

Rev.	Date	Description		
		Page	**Summary**	
1.00	Dec.1.14	—	First edition issued	
1.01	May.20.19	—	Supported the following compilers: - GCC for Renesas RX - IAR C/C++ Compiler for Renesas RX	
		1	Added the section of Target Compilers.	
		9	Added the section of 2.7 Code Size.	
		11	Added the section of 2.10 "for", "while" and "do while" statements.	
		25	Updated the section of R_LCDC_GetVersion().	
		26	Added the section of 4. Pin Setting.	
		26	Added the section of 5.1 Confirmed Operation Environment.	
		program	Function related Supported the configuration option setting function with GUI in Smart Configurator. ■ Details In order to support the configuration option setting function with GUI, a configuration file has been added.	
		program	Deleted the inline expansion of the R_LCDC_GetVersion function.	
1.02	Jun.10.20	—	Changed API function comments to Doxygen style.	
		1	Deleted Related Documents R01AN1833.	
		9	Changed Section 2.7 Code Size.	
		10	Changed Section 2.9 Adding the FIT Module to Your Project.	
		12 to 25	Delete "Reentrant" item on the API description page.	
		27	Added the section of 5.2 Confirmed Operation Environment.	
		program	Fixed the following. ■ Target device All devices. ■ Details Changed processing so that there is a register that may be accessed from multiple peripheral functions at the same time, and the atomicity of writing to that register can be ensured.	
2.00	May.31.23	all	Added supported devices for RX23E-B.	
		6	Change initial value of LCDC_CFG_PARAM_CHECKING_ENABLE	
		9	Changed Section 2.7 Code Size.	
		10	Change definition of return value	
		16	Updated content of data argument in chapter "R_LCDC_Modify()"	
		22	Revised return value in chapter "R_LCDC_AdjustContrast()"	
		program	■Overview Changed the folder structure in the module and fixed some issues. ■ Details Removed targets folder and share source files.	
			Changed return value of R_LCDC_AdjustContrast function.	

RENESAS

Rev.	Date	Description	
		Page	Summary
2.00	May.31.23	program	Changed parameter check method and register setting method for R_LCDC_Control function.
			Changed the processing of the lcdc_delay function.
			Corrected the processing that may be set while the SCOC bit is "1" for the LCDM1 register.
			Fixed the problem that the LCD count source was not stopped by the lcdc_close function.
			Change initial value of LCDC_CFG_PARAM_CHECKING_ENABLE
			Removed LCDC_SEL_CIRCUIT_ENABLE and LCDC_SEL_CIRCUIT_DISABLE definitions

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/.