RX Family

emWin v6.14g Module Firmware Integration Technology

Introduction

This application note describes the emWin v.6.14g module which uses Firmware Integration Technology (FIT). This module is hereinafter referred to as “the emWin FIT module”.

The emWin FIT module is the modularized emWin （https://www.segger.com/products/user-interface/emwin/add-ons/emwin-support-renesas-rx-mcu/） by SEGGER by using FIT

For the details of ”emWin” and GUI design tool, “AppWizard”, contact SEGGER （https://www.segger.com/）.

Target Devices

- RX65N group、RX651group　　ROM capacity：1.5MB to 2MB
- RX72N group　　ROM capacity：1.5MB to 4MB

When this application note is applied to 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 “2.9 Adding the FIT Module to Your Project”.

Related Documents

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

Contents

1. Overview

1.1 emWin FIT Module

The emWin FIT module is used as an API, by implementing in a project. Refer to “2.9 Adding the FIT Module to Your Project” for how to implement the emWin FIT module into a project.

1.2 Overview of emWin FIT module

The emWin FIT module enables emWin to be easily implemented in a user’s program with Smart Configurator by making emWIN (V.6.14g) by SEGGER correspond to FIT. We will continue to support the upgraded version of the emWin V6.14g in the future.

For the details of emWin, refer to the document below.

・emWin Graphic Library with Graphical User Interface User Guide & Reference Manual

（https://www.segger.com/downloads/emwin/UM03001）

The emWin FIT module has the limitations mentioned below.

・Recommend using the DRW2D FIT module

・In color depth, 32bps has not been supported yet.

・OS: supports only FreeRTOS and OS-less

・Does not support emFILE or embOS by SEGGER

When using the emWin FIT module, take note of the following issues.

・When using the RX65N, section setting is required.

In the emWin FIT module, two 256Kbyte-buffers need to be secured. When using in RX65N, the two buffers need to be placed separately because of the relation of address placement. Therefore, when 256Kbyte is secured from 0x00000100 and 0x00800000 each, set SU section and after that which have been originally set to 0x0084000 and after that.

・Heap memory size needs to be changed from the default value.

Change “Heap size” of “r_bsp” to 0x4000 using Smart Configurator.

・Compiler option setting is required.

Requires to set the compiler option of “locate const modifier variable to the section of which alignment number is 4 (-nostuff=C)”

When using e^2 studio, open property screen from [Project]→[C/C++ Project Settings], open [Tool setting] tab from [C/C++build]→[Setting], and check the box of “locate const modifier variable to the section of which alignment number is 4 (-nostuff=C)” from [Compiler]→[Object].

・Image format

When using images and so on with the emWin FIT module, use the data in bit map format (.bmp).

1.3 API Overview

The tables below list the API functions included in the emWin FIT module. Table 1.1 Functions which emWin calls from the Inside lists the functions which emWin calls from the inside. Table 1.2 lists the API functions which are called from the application.

For the details, refer to “3.Functions Called from emWin” and “4.API Functions Called from the Application.”

Table 1.1 Functions which emWin calls from the Inside

Function	Description
GUI_X_Config	Registers memory block which is used in emWIn memory management system
LCD_X_Config	Initializes LCD and device driver
LCD_X_DisplayDriver	Calls back function of display driver
GUI_X_Init	Initializes necessary hardware
GUI_X_Delay	Waits the specified time
GUI_X_ExecIdle	Called from Window Manager when GUI is not up to date and there is no content to be processed
GUI_X_GetTime	Current system time is obtained in integer in milliseconds.
GUI_X_ErrorOut	When a fatal error occurs, called from emWin with an error string as an input
GUI_X_Warn	When a warning occurs, called from emWIn with a warning string as an input.
GUI_X_Log	When a message occurs, called from emWin with a message string as an input.
GUI_X_InitOS	When using under multitask environment, generates semaphore or mutex
GUI_X_Unlock	When using under multitask environment, unlocks GUI
GUI_X_Lock	When using under multitask environment, locks GUI
GUI_X_GetTaskId	When using under multitask environment, obtains task ID
GUI_X_WaitEvent	When using under multitask environment, executes the waiting for an event
GUI_X_SignalEvent	When using under multitask environment, executes event notification
GUI_X_WaitEventTimed	When using under multitask environment, executes the waiting for an event during the specified period
PID_X_SetLayerIndex	Sets layer number
PID_X_Init	Initializes Pointer Input Device
GUI_TOUCH_X_ActiveX	Enables voltage measurement of x axis of Touch IC
GUI_TOUCH_X_ActiveY	Enables voltage measurement of y axis of Touch IC
GUI_TOUCH_X_MeasureX	Returns the voltage measurement result of x axis obtained from Touch IC
GUI_TOUCH_Y_MeasureY	Returns the voltage measurement result of y axis obtained from Touch IC
APPW_X_FS_Init	Initializes the file system access of AppWizard

Table 1.2 AP Functions Called from Application

Function	Function Description
R_EMWIN_GetBufferAddr	Obtains the address of frame buffer
R_EMWIN_GetD2	Obtains the handle of Dave2D function
R_EMWIN_EnableDave2D	Turns the Dave2D function into the enable state
R_EMWIN_DisableDave2D	Turns the Dave2D function into the operation inhibition state
R_EMWIN_GetDaveActive	Obtains the operating state of Dave2D function
R_EMWIN_GetVersion	Obtains the version of emWin
_VSYNC_ISR()	Performs Vsync interrupt processing (Assumes callback function of GLCDC)

1.4 Software Configuration

The application which uses the emWin FIT module has a software configuration shown in Figure 1.1.

Application uses the emWin FIT module. The emWin FIT module uses the DRW2D FIT module to create a figure, and the GLCDC FIT module to display to the LCD. Touch panel information is controlled by using the SCI-I2C FIT module.

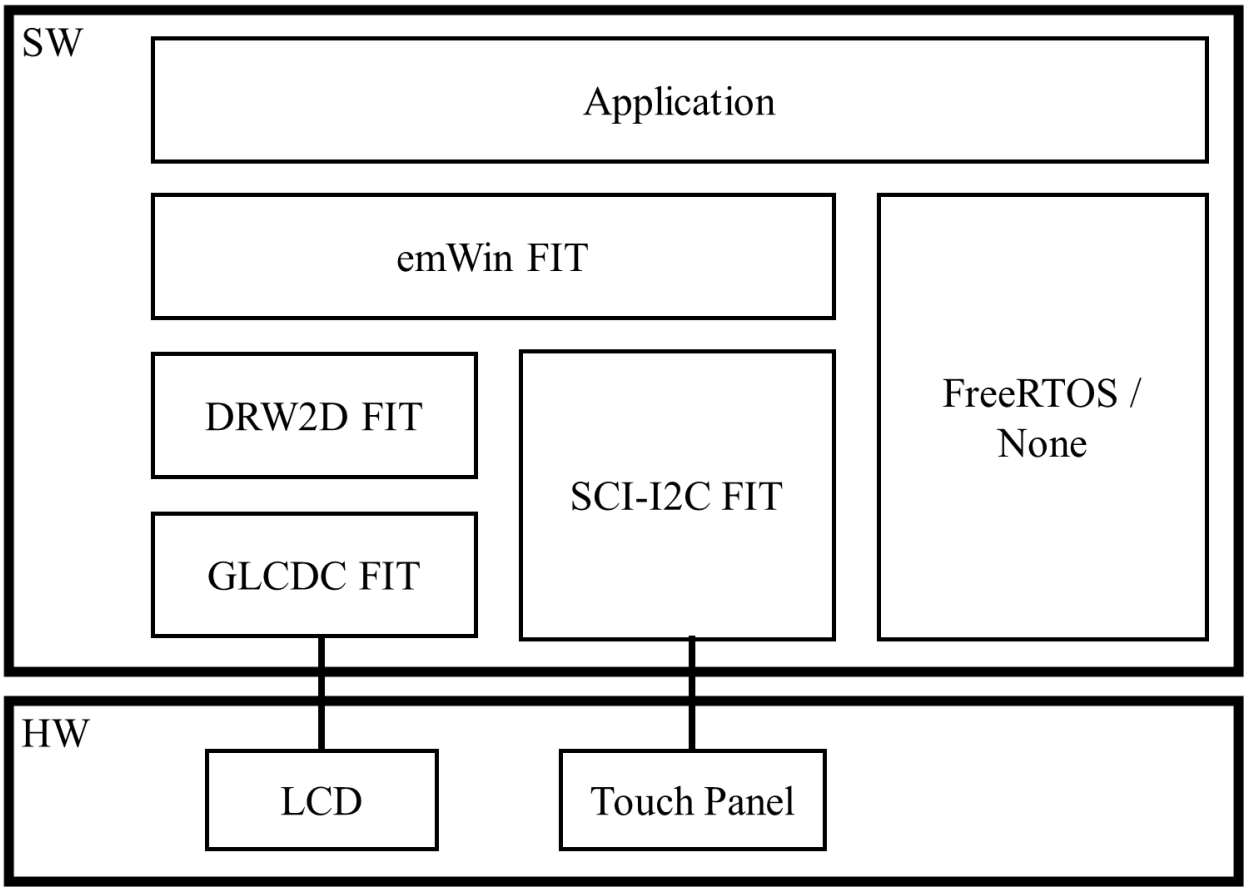

Figure 1.1 Software Configuration

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:

- GPIO
- DMAC
- CMT
- SCI
- GLCDC

2.2 Software Requirements

This driver is dependent upon the following FIT module:

- Board Supprt Package Module (r_bsp) Rev.5.52 or higher
- GPIO Module (r_gpio_rx) Rev.3.50 or higher
- DMAC Module (r_dmaca_rx) Rev.2.40 or higher
- CMT Module (r_cmt_rx) Rev.4.40 or higher
- Simple I^2C Module (r_sci_iic_rx) Rev.2.46 or higher
- Graphic LCD Controller Moduke (r_glcdc_rx) Rev.1.40 or higher
- DRW2D Drover Module (r_drw2d_rx) Rev.1.10 or higher

2.3 Supported Toolchain

This FIT module has been confirmed to work with the toolchain listed in 8.1 Confirmed Operation Environment.

2.4 Header Files

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

2.5 Integer Types

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

2.6 Configuration while Compiling

The configuration option settings of the emWin FIT module are performed in r_emwin_rx_config.h. The option names and setting values are listed in the table below:

Configuration options in r_emwin_rx_config.h

EMWIN_GUI_NUM_BYTES	Specifies the maximum memory size used in GUI.
EMWIN_XSIZE_PHYS	Specifies horizontal LCD size.
EMWIN_YSIZE_PHYS	Specifies vertical LCD size.
EMWIN_USE_DRW2D	Selectable whether to use DRW2D module. - When this is set to 0, DRW2D module is not used. - When this is set to 1, DRW2D module is used.
EMWIN_USE_MULTITOUCH	Selectable whether to use multi-touch function. - When this is set to 0, multi touch function is not used. - When this is set to 1, multi touch function is used.
EMWIN_SLAVE_ADDRESS	Specifies slave address of touch panel.
EMWIN_MAX_NUM_TOUCHPOINTS	Specifies the maximum number of touch panel points. The value is used when the multi-touch function is used.
EMWIN_GUI_FRAME_BUFFER1	Specifies start address of the frame buffer 1 to display image.
EMWIN_GUI_FRAME_BUFFER2	Specifies start address of the frame buffer 2 to display image.
EMWIN_USE_DISP_SIGNAL_PIN	Selectable whether to use LCD reset pin. - When this is set to 0, LCD reset pin is not used. - When this is set to 1, LCD reset pin is used.
EMWIN_DISP_SIGNAL_PIN	Specifies GPIO pin to use as LCD reset pin. To set this configuration, use gpio_port_pin_t member in GPIO FIT module.
EMWIN_USE_BACKLIGHT_PIN	Selectable whether to use LCD backlight pin. - When this is set to 0, LCD backlight pin is not used. - When this is set to 1, LCD backlight pin is used.
EMWIN_BACKLIGHT_PIN	Specifies GPIO pin to use as LCD backlight pin. To set this configuration, use gpio_port_pin_t member in GPIO FIT module.
EMWIN_USE_TOUCH_IC_RESET_PIN	Selectable whether to use LCD touch IC reset pin. - When this is set to 0, LCD touch IC reset pin is not used. - When this is set to 1, LCD touch IC reset pin is used.
EMWIN_TOUCH_IC_RESET_PIN	Specifies GPIO pin to use as LCD touch IC reset pin. To set this configuration, use gpio_port_pin_t member in GPIO FIT module.
EMWIN_SCI_IIC_NUMBER	Specifies SCI channel number to use in I^2C communication to touch panel.
EMWIN_BITS_PER_PIXEL	Selects color depth value. - When this is set to 1, color depth is set as 1 bpp. Then, DRW2D FIT module can not be used. - When this is set to 4, color depth is set as 4 bpp. Then, DRW2D FIT module can not be used. - When this is set to 8, color depth is set as 8 bpp. Then, DRW2D FIT module can not be used. - When this is set to 16, color depth is set as 16 bpp. This value is recommended in this emWin FIT module. - When this is set to 32, color depth is set as 32 bpp. This value is not supported in this emWin FIT module.
EMWIN_DISPLAY_ORIENTATION	Selects LCD orientation. To set this configuration, use the definition defined in r_emwin_rx_config.h. - When this is set to ORIENTATION_0, the displayed image is not rotated. - When this is set to ORIENTATION_CW, the displayed image is rotated clockwise by 90 degrees. - When this is set to ORIENTATION_180, the displayed image is rotated by 180 degrees. - When this is set to ORIENTATION_CCW, the displayed image is rotated counter-clockwise by 90 degrees.

2.7 Code Size

The sizes of ROM, RAM and maximum stack usage of the emWin FIT 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 while Compiling”

The values in the table below are confirmed under the following conditions

Module Revision: emWin Rev6.14g FIT Rev1.30

Compiler Version: Renesas Electronics C/C++ Compiler Package for RX Family V3.02.00

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

GCC for Renesas RX 8.03.00.202002

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

IAR C/C++ Compiler for Renesas RX version 4.14.1

(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
RX65N	ROM	**103K bytes**	**127K bytes**	**59K bytes**
	RAM	**84K bytes**	**84K bytes**	**83K bytes**
	STACK	**408 bytes**	**440 bytes**	**576 bytes**
RX72N	ROM	**118K bytes**	**128K bytes**	**59K bytes**
	RAM	**84K bytes**	**84K bytes**	**83K bytes**
	STACK	**620 bytes**	**440 bytes**	**576 bytes**

2.8 Parameter

This section describes the parameter structure used by the API functions in this module. The structure is located in r_emwin_rx_if.h as are the prototype declarations of API functions

2.9 Adding the FIT Module to Your Project

The emWin FIT module must be added to each project in which it is used. Renesas recommends the method using the Smart Configurator described in (1) below. However, the Smart Configurator only supports some RX devices. Please use the methods of (2) 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 the application note, “RX Smart Configurator User’s Guide: e^2 studio (R20AN0451)” for details
Note : When there are emWin FIT modules with other versions in the directory to store downloaded FIT modules, Smart Configurator may not add the emWin FIT module precisely. Please store the latest emWin FIT module and do not leave other emWin FIT modues in the directory.

(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 the application note, “RX Family Adding Firmware Integration Technology Modules to Projects (R01AN1723)” for details.

2.10 for, while, and do while Expressions

This module uses *for*, *while*, and *do while* expressions (loop processing) for standby states such as waiting for register values to be updated. These instances of loop processing are indicated by the keyword WAIT_LOOP in the comments. Therefore, if you wish to incorporate failsafe processing into the instances of loop processing, you can locate them in the code by searching for the keyword WAIT_LOOP.

Target devices for which WAIT_LOOP is indicated in the comments

- RX651 Group and RX65N Group
- RX66N Group
- RX72M Group
- RX72N Group

An example code listing is shown below.

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

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

Example of a do while expression:
/* 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. Functions Called from emWin

3.1 GUI_X_Config()

This function is a function to register memory block used in the memory management system of emWin

Format
void GUI_X_Config(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
Used to register memory block which is used in the memory management system of emWin

In the emWin FIT module, assigns memory by using GUI block function.

Reentrant
- No

3.2 LCD_X_Config ()

This function is a function to initialize LCD and device drivers.

Format

void LCD_X_Config(void)

Parameters

None

Return Values

None

Properties

Prototyped in LCD.h

Description

Used to initialize LCD and device drivers

The emWin FIT module initializes LCD using GUI block function, and also initializes the DRW2D FIT module.

Reentrant

- No

3.3 LCD_X_DisplayDriver ()

This function is the callback function of display driver.

Format

```
int LCD_X_DisplayDriver(
        unsigned layer_index,
        unsigned cmd,
        void * p_data
        )
```

Parameters

layer_index	Input	Layer number
cmd	Input	Executed command
p_data	Input	Pointer to data structure

Return Values

0:	Command has been executed normally
-1:	Command has not been executed
-2:	Error occurs

Properties

Prototyped in LCD.h

Description

Used as a callback function of the display driver. Called from display driver and executes callback routine.

In the emWin FIT module, initializes the GLCDC FIT module according to a command, registers figure generation function using the DRW2D FIT module, sets Lookup Table entry, turns on/off display and switches buffer.

Command	Value	Meaning	Supporting status ○ : Supported × : Not supported
LCD_X_INITCONTROLLER	0x01	Initializes display controller	○
LCD_X_SETVRAMADDR	0x02	VSets Video RAM address	×
LCD_X_SETORG	0x03	Sets standard within layer	×
LCD_X_SETLUTENTRY	0x04	Sets Lookup Table entry	○
LCD_X_ON	0x05	Switches on display	○
LCD_X_OFF	0x06	Switches off display	○
LCD_X_SETSIZE	0x07	Sets layer size	×
LCD_X_SETPOS	0x08	Sets layer position	×
LCD_X_SETVIS	0x09	Sets layer visualization	×
LCD_X_SETALPHA	0x0A	Sets layer alpha value	×
LCD_X_SETALPHAMODE	0x0B	Sets alpha blending mode	×
LCD_X_SETCHROMAMODE	0x0C	Sets chroma blending mode	×
LCD_X_SETCHROMA	0x0D	Sets chroma value	×
LCD_X_SHOWBUFFER	0x0E	Switches buffer	○

Reentrant

- No

3.4 GUI_X_Init ()

This function is a function to initialize hardware necessary to GUI.

Format

void GUI_X_Init(void)

Parameters

None

Return Values

None

Properties

Prototyped in GUI.h

Description

A function to initialize necessary hardware

In the emWin FIT module, used to initialize compare match timer which is used for latency measurement

Reentrant

- No

3.5 GUI_X_Delay ()

This function is a function to wait for a specified time.

Format

```
void GUI_X_Delay(
        int ms
        )
```

Parameters

ms	Input	Latency [a millisecond]

Return Values

None

Properties

Prototyped in GUI.h

Description

Waits for a specified time

In the emWin FIT module, waits for a specified time by utilizing time information obtained from compare match timer.

Reentrant

- No

3.6 GUI_X_ExecIdle ()

This function is a function called from Window Manager when there is no content to be processed because GUI is up to date.

Format
void GUI_X_ExecIdle(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
Called from Window Manager when GUI is up to date and there is no content to be processed.

In the emWin FIT module, performs no processing

Reentrant
- No

3.7 GUI_X_GetTime ()

This function is a function in which the current system time is obtained with integer type of millisecond unit.

Format

```
GUI_TIMER_TIME GUI_X_GetTime(
                    int ms
                    )
```

Parameters

None

Return Values

System time [millisecond]

Properties

Prototyped in GUI.h

Description

The current system time is obtained with integer type of millisecond unit.

In the emWin FIT module, returns a value obtained from compare match timer

Reentrant

- No

3.8 GUI_X_ErrorOut ()

This function is a function called from emWin with an error string as an input when a fatal error occurs.

Format

```
void GUI_X_ErrorOut(
        const char *s
        )
```

Parameters

s	Input	Error string

Return Values

None

Properties

Prototyped in GUI.h

Description

When a fatal error occurs, called from emWin with an error string as an input

Enabled when GUI_DEBUG_LEVEL ≧ 3

In emWin FIT module, performs no processing.

Reentrant

- No

3.9 GUI_X_Warn ()

This function is a function called from emWin with a warning string as an input when a warning occurs.

Format

```
void GUI_X_Warn(
        const char *s
        )
```

Parameters

s	Input	Warning string

Return Values

None

Properties

Prototyped in GUI.h

Description

When a warning occurs, called from emWin with a warning string as an input

Enabled when GUI_DEBUG_LEVEL ≧ 4

In the emWin FIT module, performs no processing.

Reentrant

- No

3.10 GUI_X_Log ()

This function is a function called from emWin with a message string as an input when a message occurs.

Format

```
void GUI_X_Log(
        const char *s
        )
```

Parameters

s	Input	Message string

Return Values

None

Properties

Prototyped in GUI.h

Description

When a message occurs, called from emWin with a message string as an input

Enabled when GUI_DEBUG_LEVEL ≧ 5

In the emWIn FIT module, performs no processing.

Reentrant

- No

3.11 GUI_X_InitOS ()

This is a function to generate a semaphore or a mutex when used under multitask environment。

Format
void GUI_X_InitOS(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
A function to generate a semaphore or a mutex when used under multitask environment.

In the emWin FIT module, generates a semaphore and an event using FreeRTOS function when using FreeRTOS. When not using Free RTOS, performs no processing.

Reentrant
- No

3.12 GUI_X_Unlock ()

This function is a function to unlock GUI when used under multitask environment.

Format
void GUI_X_Unlock(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
A function to unlock GUI when used under multitask environment

In the emWin FIT module, releases a semaphore using FreeRTOS function when using FreeRTOS. When not using FreeRTOS, performs no processing.

Reentrant
- No

3.13 GUI_X_Lock ()

This function is a function to lock GUI when used under multitask environment.

Format
void GUI_X_Unlock(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
A function to lock GUI when used under multitask environment

In the emWIN FIT module, obtains a semaphore using FreeRTOS function when using FreeRTOS. When not using FreeRTOS, performs no processing.

Reentrant
- No

3.14 GUI_X_GetTaskId ()

A function to obtain a task ID when used under multitask environment

Format
U32 GUI_X_GetTaskId(void)

Parameters
None

Return Values
Task ID

Properties
Prototyped in GUI.h

Description
A function to obtain a task ID when used under multitask environment

In the emWin FIT module, obtains a task ID using FreeRTOS function when using FreeRTOS. When not using FreeRTOS, constantly returns 1.

Reentrant
- No

3.15 GUI_X_WaitEvent ()

A function to execute the waiting for an event when used under multitask environment

Format

void GUI_X_WaitEvent(void)

Parameters

None

Return Values

None

Properties

Prototyped in GUI.h

Description

A function to execute the waiting for an event when used under multitask environment

In the emWin FIT module, executes the waiting for an event using FreeRTOS function when using FreeRTOS. On this occasion, maximum waiting time is 60000 milliseconds. When not using FreeRTOS, performs no processing.

Reentrant

- No

3.16 GUI_X_SignalEvent ()

A function to execute event notification when used under multitask environment

Format
void GUI_X_SignalEvent(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
A function to execute event notification when used under multitask environment

In the emWin FIT module, executes event notification using FreeRTOS function when using FreeRTOS. When not using FreeROTS, performs no processing.

Reentrant
- No

3.17 GUI_X_WaitEventTimed ()

This function is a function to execute the waiting for an event for a specified period when used under multitask environment.

Format

```
void GUI_X_WaitEventTimed(
        int period
        )
```

Parameters

Period	Input	Specified period

Return Values

None

Properties

Prototyped in GUI.h

Description

A function to execute the waiting for an event for a specified period when used under multitask environment.

In the emWin FIT module, executes the waiting for an event for a specified period using FreeRTOS function when using FreeRTOS. In this occasion, maximum waiting time is 60000 milliseconds. When not using FreeRTOS, performs no processing.

Reentrant

- No

3.18 PID_X_SetLayerIndex ()

This function is a function to set a layer number.

Format

```
void PID_X_SetLayerIndex(
        int layer_index
        )
```

Parameters

LayerIndex	Input	Layer number

Return Values

None

Properties

Prototyped in PIDConf.h

Description

Sets a layer number

In the emWin FIT module, sets a layer number to internal variable.

Reentrant

- No

3.19 PID_X_Init ()

This function is a function to initialize Pointer Input Device.

Format
void PID_X_Init(void)

Parameters

Return Values
None

Properties
Prototyped in PIDConf.h.

Description
Initializes Pointer Input Device

In the emWin FIT module, resets Touch IC, initializes SCI-I2C, boots compare match timer and registers callback function to obtain touch information, and enables multi-touch function.

Reentrant
- No

3.20 GUI_TOUCH_X_ActiveX ()

This function is a function to enable the voltage measurement of the X axis of Touch IC.

Format
void GUI_TOUCH_X_ActivateX(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
A function to enable voltage measurement of the X axis of Touch IC

In the emWin FIT module, performs no processing.

Reentrant
- No

3.21 GUI_TOUCH_X_ActiveY ()

This function is a function to enable the voltage measurement of the Y axis of Touch IC.

Format
void GUI_TOUCH_X_ActivateY(void)

Parameters
None

Return Values
None

Properties
Prototyped in GUI.h

Description
A function to enable the voltage measurement of the Y axis of Touch IC

In the emWIn FIT module, performs no processing.

Reentrant
- No

3.22 GUI_TOUCH_X_MeasureX ()

This function is a function to return the X axis voltage measurement result obtained from Touch IC.

Format
int GUI_TOUCH_X_MeasureX(void)

Parameters
None

Return Values
0

Properties
Prototyped in GUI.h

Description
A function to return the X axis voltage measurement result obtained from Touch IC

In the emWin FIT module, constantly returns 0.

Reentrant
- No

3.23 GUI_TOUCH_Y_MeasureY ()

This function is a function to return the Y axis voltage measurement result obtained from Touch IC.

Format
int GUI_TOUCH_X_MeasureY(void)

Parameters
None

Return Values
0

Properties
Prototyped in GUI.h

Description
A function to return the Y axis voltage measurement result obtained from Touch IC

In the emWin FIT module, constantly returns 0.

Reentrant
- No

3.24 APPW_X_FS_Init ()

This function is a function to initialize the file system access of AppWizard.

Format
void APPW_X_FS_Init (void)

Parameters
None

Return Values
None

Properties
Prototyped in AppWizard.h

Description
A function to initialize the file system access of AppWizard

In the emWin Fit module, performs no processing.

Reentrant
- No

4. API Functions Called from the Application

4.1 R_EMWIN_GetBufferAddr()

This function is a function to obtain the address of the frame buffer which is used in the emWin FIT module.

Format
void * R_EMWIN_GetBufferAddr (void)

Parameters
None

Return Values
Frame buffer address

Properties
Prototyped in r_emwin_rx_if.h

Description
Obtains the address of the frame buffer used in the emWin FIT module.

Reentrant
- No

4.2 R_EMWIN_GetD2 ()

This function is a function to obtain the handle of the Dave2D function of the emWin FIT module.

Format
d2_device * R_EMWIN_GetD2 (void)

Parameters
None

Return Values
Handle of Dave2D

Properties
Prototyped in r_emwin_rx_if.h

Description
Obtains the handle of the Dave2D function of the emWin FIT module

This function is enabled only when the DRW2D FIT module is used.

Reentrant
- No

4.3 R_EMWIN_EnableDave2D ()

This function is a function to turn the Dave2D function of the emWin FIT module into the enable state.

Format
void R_EMWIN_EnableDave2D (void)

Parameters
None

Return Values
None

Properties
Prototyped in r_emwin_rx_if.h

Description
Turns the Dave2D function of the emWin FIT module into the enabled state.

This function is enabled only when the DRW2D FIT module is used.

Reentrant
- No

4.4 R_EMWIN_DisableDave2D ()

This function is a function to turn the Dave2D function of the emWin FIT module into the operation inhibition state.

Format
void R_EMWIN_DisableDave2D (void)

Parameters
None

Return Values
None

Properties
Prototyped in r_emwin_rx_if.h

Description
Turns the Dave2D function of the emWin FIT module into the operation inhibition state.

This function is enabled only when the DRW2D FIT module is used.

Reentrant
- No

4.5 R_EMWIN_GetDaveActive ()

This function is a function to obtain the operation state of the Dave2D function of the emWin FIT module.

Format
uint32_t R_EMWIN_GetDaveActive (void)

Parameters
None

Return Values
Dave2D operation state （0:state of forbidding operation, 1:State of enabling operation）

Properties
Prototyped in r_emwin_rx_if.h

Description
Obtains the operation state of the Dave2D function of the emWin FIT module

This function is enabled only when the DRW2D FIT module is used.

Reentrant
- No

4.6 R_EMWIN_GetVersion ()

This function is a function to obtain the version number of the emWin FIT module.

Format

void R_EMWIN_GetVersion(st_emwin_version_t * version)

Parameters

* version	Output	Pointer of the storage destination of a version number

Return Values

None

Properties

Prototyped in r_emwin_rx_if.h

Description

Obtains the version number of the emWin FIT module

Reentrant

- No

4.7 _VSYNC_ISR ()

This function is a function to perform V-sync interrupt processing

Format

void _VSYNC_ISR(void * p)

Parameters

* p	Output	Callback argument from GLCDC

Return Values

None

Properties

Prototyped in r_emwin_rx_if.h

Description

Performs V-sync interrupt processing

Assuming the callback function of the GLCDC FIT module

Reentrant

- No

5. Pin Setting

The pin setting to use the emWin FIT module can be performed with QE for Display [RX].

The pins which require the setting are the reset pin of the LCD panel, the backlight pin of the LCD panel, the reset pin of the touch IC mounted in the LCD panel.

In case of e^2 studio, by using the pin setting function of the emWin setting dialog of the QE for Display [RX], pin setting can be performed. When using the QE for Display [RX], pin setting regarding r_emwin_rx with Smart Configurator is not required.

Information of the selected pin is applied to qe_emwin_config.h. Macro definition value shown in 2.6 Configuration while Compiling. When QE for Display [RX] is used, macro definitions in r_emwin_rx_config.h are disabled.

When performing the pin setting without using the QE for Display [RX], compile r_emwin_rx_config_reference.h included in the emWin FIT module and generate r_emwin_rx_config.h.

6. Notation to implement the emWin FIT module

When the emWin FIT module is implemented, please note the following matters.

6.1 Selecting the library file

The emWin FIT module includes following library files. Please select the library correspond to the MCU and compiler.

Library files	
emWinLib_CCRX.lib	A library file to be used with Renesas Electronics C/C++ Compiler for RX Family.
libemWinLib_GCC.a	A library file to be used with GCC for Renesas RX.
emWinLib_RXv2_IAR.a	A library file to be used with MCUs which loads RXv2 core such as RX65N and IAR C/C++ Compiler for Renesas RX.
emWinLib_RXv3_IAR.a	A library file to be used with MCUs which loads RXv3 core such as RX72N and IAR C/C++ Compiler for Renesas RX.

6.2 Implementing to the environment exclusive of operation confirmed

When the emWin FIT module is implemented to the environment exclusive of operation confirmed, please note following matters.

・Initialization of GLCDC FIT module

In the emWin FIT module, configuration option of Smart Configurator or QE for Display is used to initialize GLCDC FIT module. When the GLCDC FIT module needs to be initialized with setting data structure and without the configuration option, please insert the data to R_GLCDC_Open function in init_controller function which is located in LCDConf.c file.

・Using touch panel

In the emWin FIT module, touch panel operation is implemented to use the touch panel which is attached to the evaluation board with I^2C interface. When the touch panel process needs to handle other touch panels or use other interfaces, please edit the corresponding processes which are located in PIDConf.c file.

7. Sample Application

Sample application is stored in doc/Training folder. For the detail, refer to the document below.

・emWin Training

8. Appendix

8.1 Confirmed Operation Environment

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

Table 8.1 Confirmed Operation Environment (Envision Kit)

Item	Contents
Integrated Development Environment	Renesas Electronics e^2 studio 2021-01
C compiler	Renesas Electronics C/C++ Compiler for RX Family(CC-RX) V3.02.00 Compile option： Add the option below to the default setting of the Integrated Development Environment. -lang = c99
	GCC for Renesas RX 8.03.00.202002 Compile option： Add the option below to the default setting of the Integrated Development Environment -std=gnu99
	IAR C/C++ Compiler for Renesas RX version 4.14.1 Compile option：the default setting of the Integrated Development Environment
Endian	Little endian
Version of the Module	Ver.1.30
Board used	Renesas Envision KIT RPBRX65N（Product No.：RTK5RX65N2C00000BR） Renesas Envision Kit RPBRX72N（Product No.：RTK5RX72N0C00000BJ）

Table 8.2 Confirmed Operation Environment (Renesas Starter Kit)

Item	Contents
Integrated Development Environment	Renesas Electronics e^2 studio 2021-01
C compiler	Renesas Electronics C/C++ Compiler for RX Family(CC-RX) V3.02.00 Compile option： Add the option below to the default setting of the Integrated Development Environment. -lang = c99
Endian	Little endian
Version of the Module	Ver.1.30
Board used	Renesas Starter Kit+ for RX65N-2MB（Product No.：RTK50565N2S10010BE） Renesas Starter Kit+ for RX72N（Product No.：RTK5572NNHS10000BE）

8.2 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:

- When 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)”

9. Reference Documents

User's manual: Software

・emWin Graphic Library with Graphical User Interface User Guide & Reference Manual

（https://www.segger.com/downloads/emwin/UM03001）

User's manual: Hardware

・RX Family RX65N Group, RX651 Group User's Manual Hardware（R01UH0590）

（The latest version can be downloaded from the Renesas Electronics website.）

・RX Family RX72N Group User's Manual Hardware（R01UH0824）

（The latest version 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 Environment

RX Family C/C++ Compiler CC-RX User's Manual（R20UT3248）

（The latest version can be downloaded from the Renesas Electronics website.）

Related Technical Update

This module has no technical update.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	2020.8.5	—	First edition issued
1.10	2020.9.4	1	Contacts of SEGGER and EmbITek：added
		3	Future version up policy: added
		3	Limitations: added
1.20	2020.12.25	—	Improve the application note according to the template
1.30	2021.3.31	—	Revised emWin version in document title to v6.14g
		12	Added section “for, while, and do while Expressions”
		46	Updated explanation of config header when QE is used
		47	Added section “Sample Application”

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.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

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/.