RX Family

R20AN0075EJ0105
Rev.1.05
Oct 01, 2016

Web server using the embedded TCP/IP M3S-T4-Tiny Module Firmware Integration Technology

Introduction

This application note explains Web server using the embedded TCP/IP M3S-T4-Tiny Module (hereafter Web server).

Web server is provided as Firmware Integration Technology (FIT) Module. Please refer to the URL to know FIT outline.

https://www.renesas.com/en-us/solutions/rx-applications/fit.html

Web server is used by combining the following middleware products.

Table 1 Middleware products

Function	Middleware Product	Web Page*1
TCP/IP	M3S-T4-Tiny (hereafter T4) (R20AN0051)	http://www.renesas.com/mw/t4
FTP server and Web server Interface	File driver for FTP server and Web server Module (R20AN0333)	http://www.renesas.com/mw/t4
File system	M3S-TFAT-Tiny(R20AN0038)	http://www.renesas.com/mw/tfat
File system Interface	M3S-TFAT-Tiny Memory Driver Interface (R20AN0335)	http://www.renesas.com/mw/tfat
MMC driver	SPI mode MultiMediaCard Driver*2	http://www.renesas.com/mw/tfat http://www.renesas.com/mw/tfs
MMC extensions (board)	Middleware Evaluation board*3	http://www.renesas.com/mw/tfat http://www.renesas.com/mw/tfs http://www.renesas.com/mw/s2 http://www.renesas.com/mw/dtmf
USB driver	USB driver	http://www.renesas.com/driver/usb

Notes: 1. The items with multiple page references can be downloaded from the related middleware sites. There are no differences between the downloadable application notes themselves.
2. The SD(less 2GB size) card that has compatible command for MMC is available on this software.
3. The middleware evaluation board must be produced by the user based on these application notes.

Since each of these middleware packages are independent, they can be combined freely if the user implements interface programs. For example, the file system can be replaced by another file system, or the MMC driver can be replaced with a USB driver.

Furthermore, since the web server program itself contains no program code that depends on the microcontroller, it can be easily ported to another microcontroller simply by replacing the TCP/IP software stack with one for the other microcontroller.

We prepared sample programs for each CPU board included in the Renesas Starter Kit. For more information, see Renesas Starter Kit for sample application notes.

Table 2 Sample application notes

sample application notes	document number	website
Application example using T4 (DHCP/DNS/FTP/HTTP) Firmware Integration Technology	R20AN0314	https://www.renesas.com/mw/t4

Target Device

RX Family

Contents

1. Outline 5
1.1 System Structure 5
1.2 Software Structure 6
1.2.1 In case MMC driver or in case USB driver 6
1.2.2 In case internal ROM 7
1.3 CGI function 7

2. API Information 8
2.1 Hardware Requirements 8
2.2 Software Requirements 8
2.3 Supported Toolchains 8
2.4 Limitations 8
2.5 Header Files 8
2.6 Configuration Overview 9
2.7 Adding Library to Your Project 10

3. API Functions 11
3.1 R_httpd 11
3.2 R_httpd_pending_release_request 12
3.3 R_T4_HTTP_SERVER_GetVersion 13

4. File driver for FTP server and Web server Module 14
4.1 Data structure 15
4.2 change_dir 16
4.3 file_close 16
4.4 file_delete 17
4.5 file_open 17
4.6 file_read 18
4.7 file_rename 18
4.8 file_exist 19
4.9 file_write 19
4.10 get_file_info 20
4.11 get_file_list_info 21
4.12 get_file_size 22
4.13 make_dir 22
4.14 remove_dir 23

5. User-Defined Function Reference 24
5.1 System timer 24
5.1.1 Data structure 24

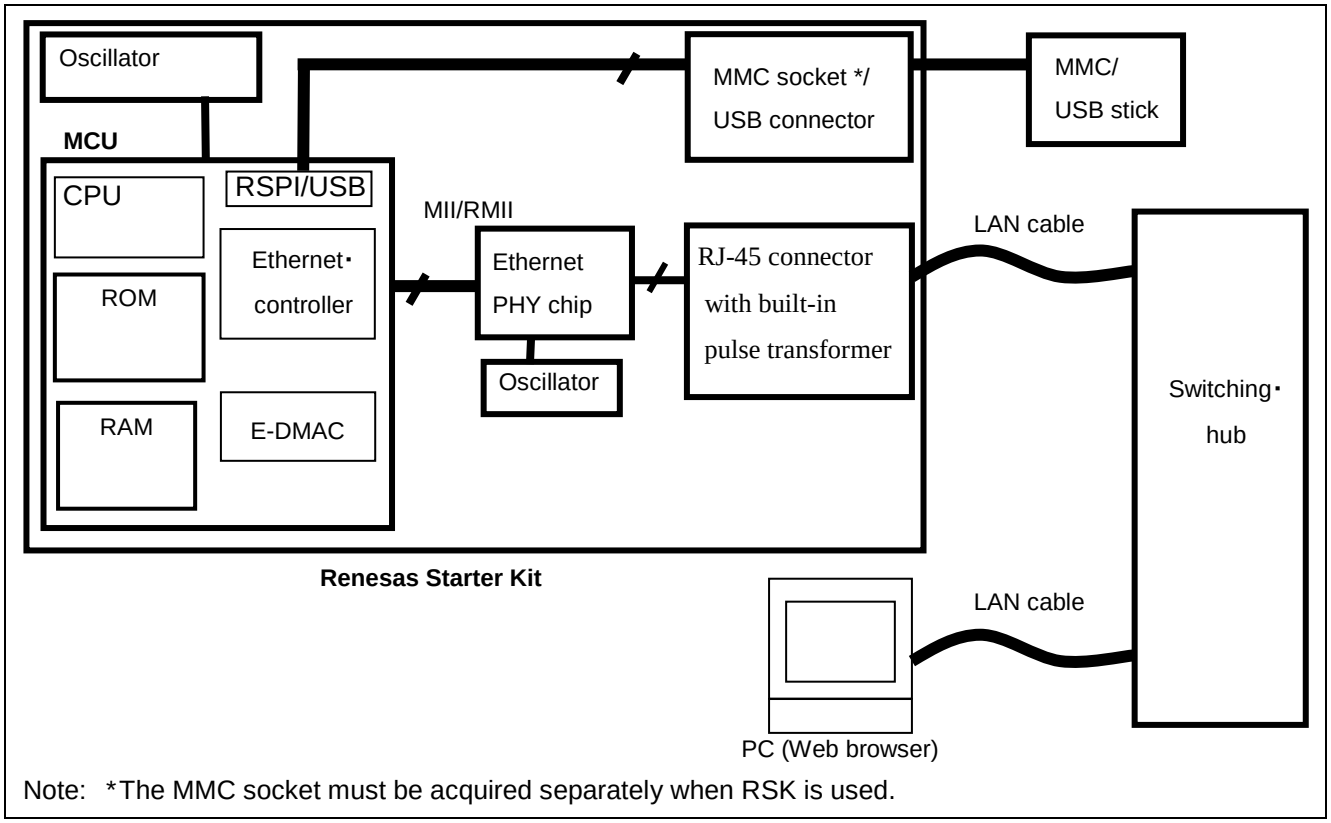
5.1.2 get_sys_time 24
6. Sample CGI Function 25
6.1 cgi_sample_function 25

1. Outline

This web server is an application that operates over TCP/IP, can be accessed from an ordinary web browser, and provides functions for transferring content stored on the web server to web browsers using TCP/IP.

1.1 System Structure

Show System Structure Example.



Note: *The MMC socket must be acquired separately when RSK is used.

Figure 1 System Structure Example

1.2 Software Structure

1.2.1 In case MMC driver or in case USB driver

Show Software Structure Example in case MMC driver or in case USB driver.

Web Server function scope

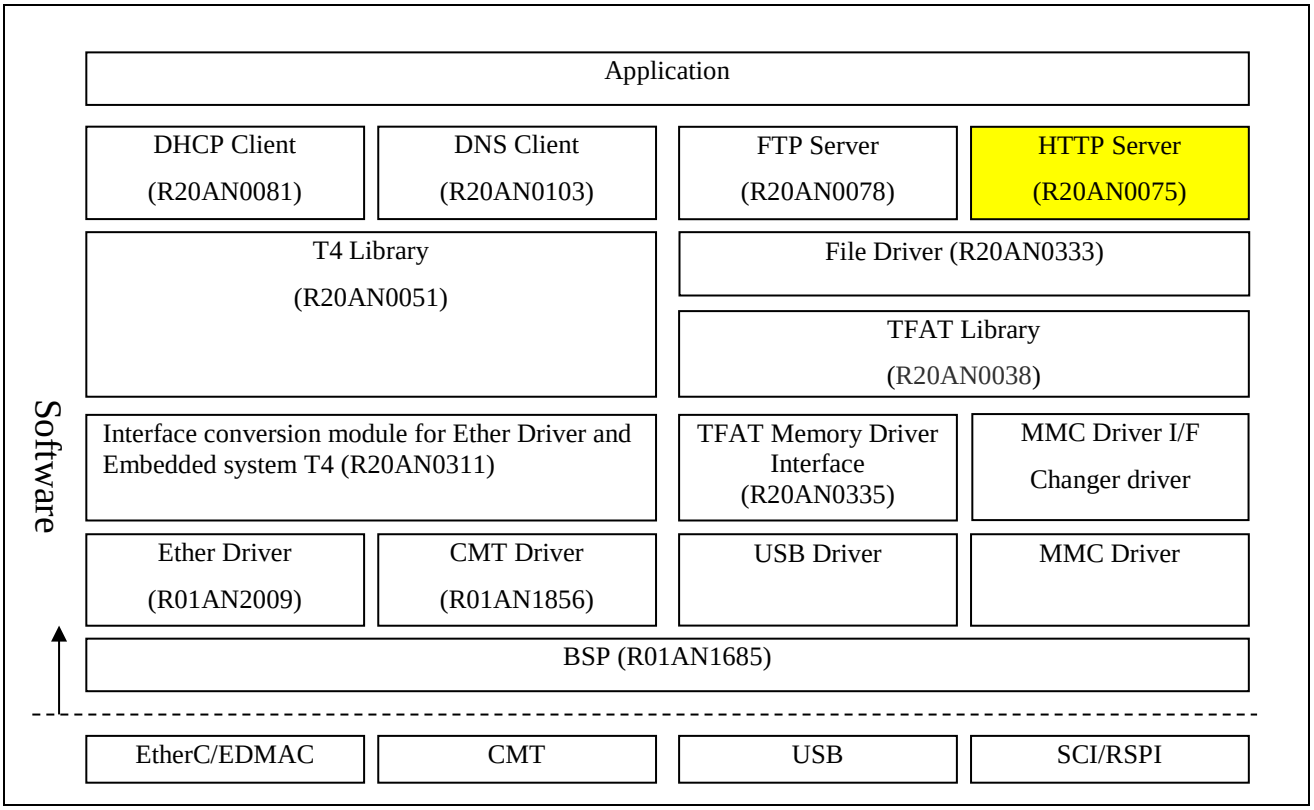


Figure 2 Software Structure Example（in case, store the web contents to MMC/USB memory）

1.2.2 In case internal ROM

Show Software Structure Example in case internal ROM.

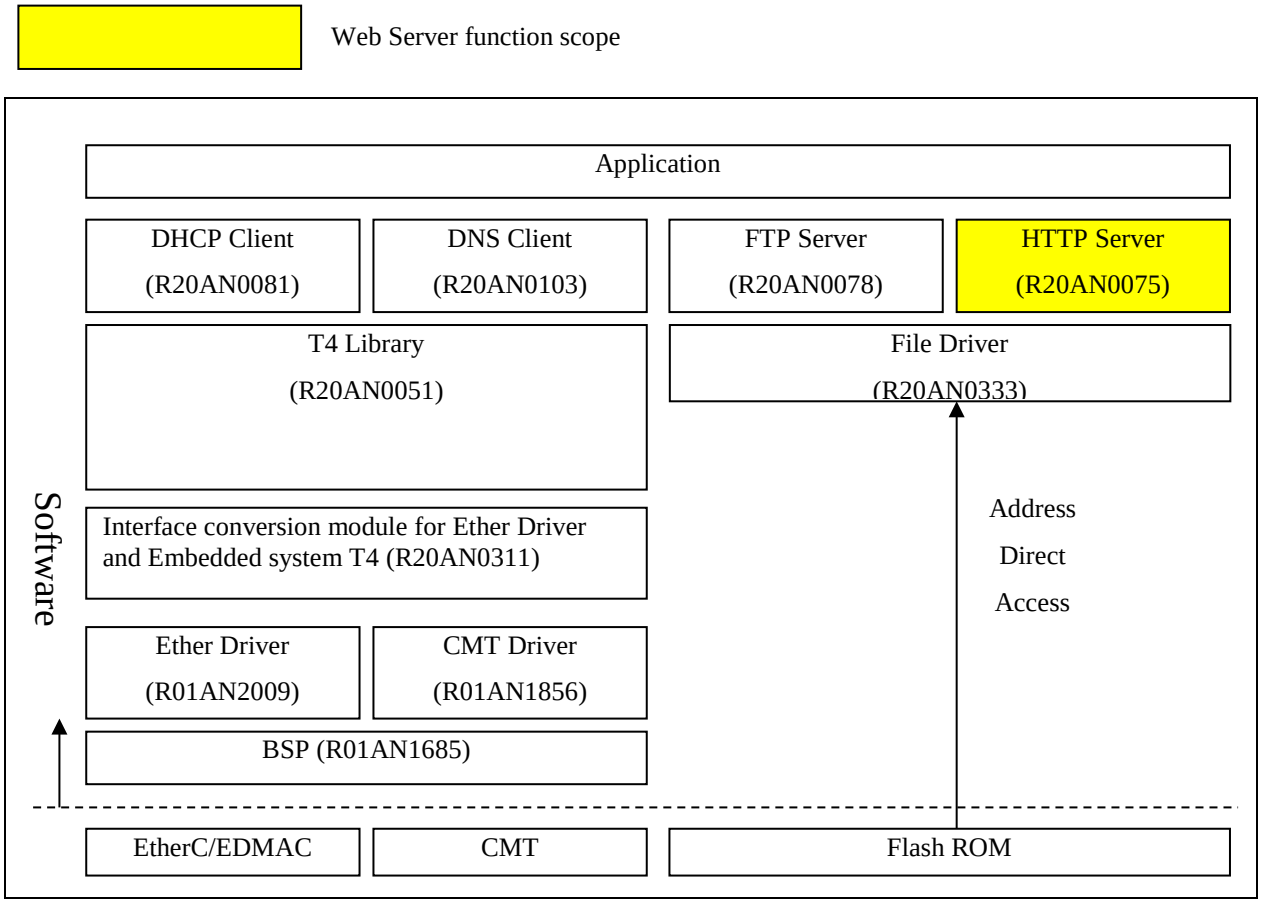


Figure 3 Software Structure Example（in case internal ROM）

1.3 CGI function

This Web server has easily implementation CGI (Common Gateway Interface) function. CGI is way to execute user program on Web server under the Web browser request. This Web server can call the CGI functions that corresponds to the URL. The CGI functions can be set in config file.

2. API Information

2.1 Hardware Requirements

None

2.2 Software Requirements

This module is dependent upon the following packages:

r_t4_rx

r_t4_file_driver_rx

2.3 Supported Toolchains

This library is tested and working with following toolchains:
Renesas RX Toolchain v.2.04.01

2.4 Limitations

This program uses stdio.h, stdlib.h, string.h, and ctype.h. Specify stdio, stdlib, string, and ctype as compiler options when compiling user programs.

2.5 Header Files

All API calls are accessed by including a single file "r_t4_http_server_rx_if.h" which is supplied with this software's project code.

2.6 Configuration Overview

All configurable options that can be set at build time are located in the file “r_t4_http_server_rx_config.h”. A summary of these settings are provided in the following table:

Table 3 Configuration options

Configuration options in ***r_t4_http_server_rx_config.h***	
#define HTTPD_VERSION_CODE ※Default value is ”“Renesas Embedded Web Server/1.03””.	Server header field. The data stored in the server header field transmitted to the web browser during communication with the web browser can be specified.
#define ROOT_DIR ※Default value is ” “”(NULL literal）”.	Root directory. Which directory in external memory is seen as the root directory can be specified. Example： #define ROOT_DIR “” #define ROOT_DIR “user” #define ROOT_DIR “user/root_dir”
#define INDEXES ※Default value is ”1”.	Displaying or not displaying an index page. The behavior when a directory is specified from the web browser can be specified. When set to 1, the directory contents are returned as the response. When set to 0, the file specified as the DEFAULT_FILE_NAME is returned as the response.
#define DEFAULT_FILE_NAME ※Default value is ”“index.htm””.	File returned as response when do not display index page is specified This file is the response when INDEXES is set to 0. A "404 Not Found" response is returned if the specified file cannot be found.
#define MAX_EXTENSION ※Default value is ”15”.	The number of corresponding content-type. This is the max number of a list of the file extensions stored in external memory.
#define EXTENSION_TYPE_TABLE_LIST ※Default value is ”{"def", "application/octet-stream"},\ {"htm", "text/html"},\ {"txt", "text/plain"},\ {"exe", "application/octet-stream"},\ {"com", "application/octet-stream"},\ {"jpg", "image/jpeg"},\ {"gif", "image/gif"},\ {"png", "image/png"},\ {"zip", "application/octet-stream"},\ {"mp3", "audio/mpeg"},\ {"wav", "Nvaudio/x-wav"},\ {"dat", "application/octet-stream"},\ {"xls", "application/vnd.ms-excel"},\ {"doc", "application/msword"},\ {"cgi", "text/html"},\”.	Corresponding content-type This is a list of the file extensions stored in external memory. If a file with an extension not defined here is transferred, the system will respond to that file with the settings for the extension defined at the head of the list.
#define MAX_CGI_FILE ※Default value is ”1”.	The number of registered CGI file

#define CGI_FILE_NAME_TABLE_LIST Default value is ” {"cgi_smpl.cgi", NULL},”.	The table includes CGI file name and coressponding internal functionCGI
#define LF_CODE ※Default value is ” "\r\n"”.	New line code used for index page generation.
#define HTTP_TCP_CEP_NUM ※Default value is ”4”.	Maximum number of clients that can be accepted at the same time This value must be set to match the number of sockets defined in config_tcpudp.c
#define HTTP_START_TCP_CEP ※Default value is “0”.	The offset value of starting position of the communication endpoint in “config_tcpudp.c”.
#define HTTP_MAX_FILE_LIST ※Default value is ”16”.	Maximum number of files that can be displayed on the index page Set this value so that BODY_BUF_SIZE is not exceeded.
#define RCV_BUF_SIZE ※Default value is ”1024”.	Reception buffer size
#define HDR_BUF_SIZE ※Default value is ”1460”.	Header file transmission buffer size
#define BODY_BUF_SIZE ※Default value is ”2048”.	Body field transmission buffer size

2.7 Adding Library to Your Project

Please refer to the Adding Firmware Integration Technology Modules to Projects (r01an1723eu0111_rx.pdf, for e^2 studio) or the Adding Firmware Integration Technology Modules to CS+ Projects (r01an1826ej0102_rx.pdf).

3. API Functions

3.1 R_httpd

This function manages the sockets required for HTTP communication.

Format

```
void R_httpd(void)
```

Parameters

None

Return Value

None

Properties

Prototyped in file “r_t4_http_server_rx_if.h”.

Description

The application calls this function periodically. This function manages the sockets required for HTTP communication. This function only performs socket management; communication itself is performed automatically by T4 as driven by interrupts.

Reentrant

No

Special Notes

None

3.2 R_httpd_pending_release_request

The function that is called when application would like to release the CGI pending

Format

```
void R_httpd_pending_release_request(ID cepid)
```

Parameters

`cepid`	`input`	`communication endpoint ID`

Return Value

None

Properties

Prototyped in file “r_t4_http_server_rx_if.h”.

Description

Application calls this function when release the CGI pending

Please refer to the section 6.1.cgi_sample_function.

Reentrant

No

Special Notes

None

3.3 R_T4_HTTP_SERVER_GetVersion

This function returns the version number of Web server.

Format

```
uint32_t      R_T4_HTTP_SERVER_GetVersion(void)
```

Parameters

None

Return Value

Version number of Web server

Properties

Prototyped in file “r_t4_http_server_rx_if.h”.

Description

Returns the version of this module. The version number is encoded such that the top two bytes are the major version number and the bottom two bytes are the minor version number.

For example, version ‘4.25’, the return value is ‘0x00040019’.

Reentrant

Yes

Special Notes

This function is inlined using the “#pragma inline” directive in “r_http_server.c”.

4. File driver for FTP server and Web server Module

The web server calls these functions. The user must code the processing performed by these functions appropriately for the file system used. Also, the web server can use this data structure to acquire information from external memory.

Table 4 API

Name	Function
change_dir()	Change current directory
file_close()	Close file
file_delete()	Delete file
file_open()	Open file
file_read()	Read file
file_rename()	Rename file
file_exist()	Confirm exit file
file_write()	Write file
get_file_info()	Get file information
get_file_list_info()	Get file list information
get_file_size()	Get file size
make_dir()	Make directory
remove_dir()	Remove directory

The gray out function is not used this sample program.

4.1 Data structure

【Date Information Structure】

```
typedef struct date_info_
{
    uint16_t year;               // 2011, 2012, ...
    uint8_t  month[4];           // Jan, Feb, Mar, ...
    uint8_t  day;                // 1-31
    uint8_t  day_of_the_week[4]; // Sun, Mon, Tus, ...
    uint16_t hour;               // 0-23
    uint16_t min;                // 0-59
    uint16_t sec;                // 0-59
}DATE_INFO;
```

【File List Structure】

```
typedef struct file_list_
{
    uint8_t file_name[13];
    uint32_t file_size;
    uint32_t file_attr;
    DATE_INFO date_info;
}FILE_LIST;
```

【Macro Definition】

```
#define FILE_WRITE  (0x10)
#define FILE_READ  (0x01)

/* File attribute bits for FILE_LIST->file_attr */
#define FILE_ATTR_RDO 0x01 /* Read only */
#define FILE_ATTR_HID 0x02 /* Hidden */
#define FILE_ATTR_SYS 0x04 /* System */
#define FILE_ATTR_VOL 0x08 /* Volume label */
#define FILE_ATTR_DIR 0x10 /* Directory */
#define FILE_ATTR_ARC 0x20 /* Archive */
```

4.2 change_dir

Description

This function sets current directory using specified argument. The argument specifies directory path in full path. Information of current directory is managed in each communication endpoint.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t change_dir(uint8_t *dir_path);
```

Parameters

dir_path	input	Pointer to directory path

Return Value

-1	No directory to change
0	Normal completion

Remark

There are two cases. The argument “dir_path” has ‘/’ termination and does not have. Please adjust for user file system.

4.3 file_close

Description

This function closes the file corresponding to the ID specified by the argument and discards the file management information.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t file_close(int32_t file_id);
```

Parameters

file_id	input	ID value of the file to close

Return Value

-1	Error
0	Normal completion

Remark

None

4.4 file_delete

Description

This function deletes the file corresponding to the ID specified by the argument. The specification of file is full path from root directory.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t file_delete(uint8_t *file_path);
```

Parameters

file_path	input	pointer to file path to delete

Return Value

-1	Error
0	Normal completion

Remark

None

4.5 file_open

Description

This function opens the file specified in its argument in exclusive read mode and saves file management information independently. It also specifies an ID value for this file management information as the return value so that the web server can reference the saved file management information by ID. The saved file management information must be saved until this ID value is passed to the file close function.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t file_open(uint8_t *file_path, uint8_t mode_flag);
```

Parameters

file_path	input	pointer to file path to open
mode_flag	input	Mode value of file open (FILE_WRITE or FILE_READ)

Return Value

-1	Error
0 and positive integer	The ID value for the opened file

Remark

The file opened state must be maintained until the corresponding ID value is passed to the file close function.

4.6 file_read

Description

This function reads the file corresponding to the ID value passed as an argument and advances the file pointer by the amount read. The file pointer is recorded in the file management information for each ID value and is maintained until the file close function is called.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t file_read(int32_t file_id, uint8_t *buf, int32_t read_size);
```

Parameters

file_id	input	ID value of the file to read
buf	output	Storage area for the file data read
read_size	input	Size of file to read

Return Value

-1	Error
0 and positive integer	Data size of receiving

Remark

None

4.7 file_rename

Description

This function renames the file specified first argument to second argument. These arguments are specified in full path from root directory.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t file_rename(uint8_t *old_name, uint8_t *new_name);
```

Parameters

old_name	input	pointer to target file name
new_name	input	pointer to after file name

Return Value

-1	Error
0	Normal completion

Remark

None

4.8 file_exist

Description

This function verifies the file or directory existing. The argument is specified in full path from root directory.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t file_exist(uint8_t *file_path);
```

Parameters

file_path	input	Pointer to file or directory path

Return Value

-1	Not exist
0	Exist

Remark

None

4.9 file_write

Description

This function writes the file corresponding to the ID value passed as an argument and advances the file pointer by the amount write. The file pointer is recorded in the file management information for each ID value and is maintained until the file close function is called.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t file_write(int32_t file_id, uint8_t *buf, int32_t write_size);
```

Parameters

file_id	input	ID value of the file to write
buf	input	Storage area for the file data write
write_size	input	Size of the file to write

Return Value

-1	Error
0	Normal completion

Remark

None

4.10 get_file_info

Description

This function reads the file management information for the file corresponding to the ID value specified as an argument and writes the file date information to a date information structure.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t get_file_info(int32_t file_id, DATE_INFO *date_info);
```

Parameters

file_id	input	ID value of the file to read
date_info	output	pointer to information of date structure to store

Return Value

-1	Error
0	Normal completion

Remark

None

4.11 get_file_list_info

Description

This function writes the file list stored at the directory path specified as an argument to a file list structure.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t get_file_list_info(uint8_t *dir_path, FILE_LIST *file_list, uint32_t num_file_list, int32_t read_index);
```

Parameters

dir_path	input	pointer to directory path to read
file_list	output	pointer to file list to store.
		This function stores ‘\0’ to end of structure
num_file_list	input	Max number of file list to read at one time
read_index	input	Index of read starting

Return Value

-1	Error
0 and positive integer	Number of file

Remark

In case return value is smaller than num_file_list, it’s the end of file list. In case return value is same value num_file_list, there is the data continuing. When this function needs continuing data, this function is called with 0 and positive integer with in read_index.

There are two cases. The argument “dir_path” has ‘/’ termination and does not have. Please adjust for user file system.

4.12 get_file_size

Description

This function reads the file management information for the file corresponding to the ID value specified as an argument and returns the file size.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t get_file_size(int32_t file_id);
```

Parameters

file_id	input	ID value of the file to read

Return Value

-1	Error
0 and positive integer	File size

Remark

None

4.13 make_dir

Description

This function makes the directory. The argument is specified in full path from root directory.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t make_dir(uint8_t *dir_path);
```

Parameters

dir_path	input	pointer to file path to make

Return Value

-1	Error
0	Normal completion

Remark

There are two cases. The argument “dir_path” has ‘/’ termination and does not have. Please adjust for user file system.

4.14 remove_dir

Description

This function removes the directory. The argument is specified in full path from root directory.

Usage

```
#include <stdint.h>
#include “r_file_driver_rx_if.h”
int32_t remove_dir(uint8_t *dir_path);
```

Parameters

dir_path	input	pointer to file path to remove

Return Value

-1	Error
0	Normal completion

Remark

There are two cases. The argument “dir_path” has ‘/’ termination and does not have. Please adjust for user file system.

5. User-Defined Function Reference

5.1 System timer

Web server calls these functions. User defines system timer.

Table 5 User-Defined function for system timer

Name	Function
get_sys_time()	Get pointer to system time

5.1.1 Data structure

【system time structure】

```
typedef struct sys_time_
{
    uint32_t sec;
    uint32_t min;
    uint32_t hour;
    uint32_t day;
    uint32_t month;
    uint32_t year;
}SYS_TIME;
```

5.1.2 get_sys_time

Description

This function gets pointer to system time.

Usage

#include <stdint.h>

#include “r_t4_http_server_rx_config.h”

SYS_TIME *get_sys_time(void);

Parameters

None

Return Value

Pointer to system time

Remark

Please specify the variable for system timer.

6. Sample CGI Function

6.1 cgi_sample_function

Description

CGI function that is defined as CGI_FILE_NAME_TABLE_LIST in "r_t4_http_server_config.h"

The second element (cgi function pointer) of CGI_FILE_NAME_TABLE_LIST will be called when web browser requests the defined cgi file URL. And next, HTTPd will call cgi function.

HTTPd behavior will be changed by the return value.

case: Normal termination

CGI process finishes in this function.

case: Internal error

CGI process errors occur in this function.

case: CGI pending

CGI process does not finish in this function. The third element (cgi function pointer) of CGI_FILE_NAME_TABLE_LIST will be called when user will call R_httpd_pending_release_request() in finishing CGI process.

Usage

```
#include "r_t4_itcpip.h"
#include "r_http_server_config.h"
#include "r_t4_http_server_rx_if.h"
ER cgi_sample_function(ID cepid, void *res_info);
```

Parameters

cepid	input	The Communication Endpoint ID that is requested CGI function execution.
res_info	input	(HTTPD_RESOURCE_INFO*)res_info->param The parameter from Web browser request.
	output	(HTTPD_RESOURCE_INFO*)res_info->res.body HTML strings as response
	output	(HTTPD_RESOURCE_INFO*)res_info->res.body_size HTML strings length as response

Return Value

-1	Internal Error
-2	CGI pending
0	Normal completion

Remark

None

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

All trademarks and registered trademarks are the property of their respective owners.

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.05	Oct.01.16	—	Updated the xml file for FIT.
1.04	Jan.05.15	1	Fixed FIT Module URL Added Support MCUs.
		4	Fixed Figure 2
		5	Fixed Figure 3
1.03	May.09.14	—	Corresponded to FIT Modules.
1.02	Apr.28.14	—	Corresponded to RX63N MCU. Corresponded to simple CGI function. Add web server runs by only internal ROM.
1.01	Sep.27.12	3	Add information about USB stick
1.00	Apr.12.11	—	First edition issued

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. 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor 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 damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

© 2016 Renesas Electronics Corporation. All rights reserved.

Colophon 5.0