RX Family

R20AN0078EJ0104
Rev.1.04
Oct 01, 2016

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

Introduction

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

FTP 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

FTP 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 FTP 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

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

1.1 System Structure

Show System Structure Example.

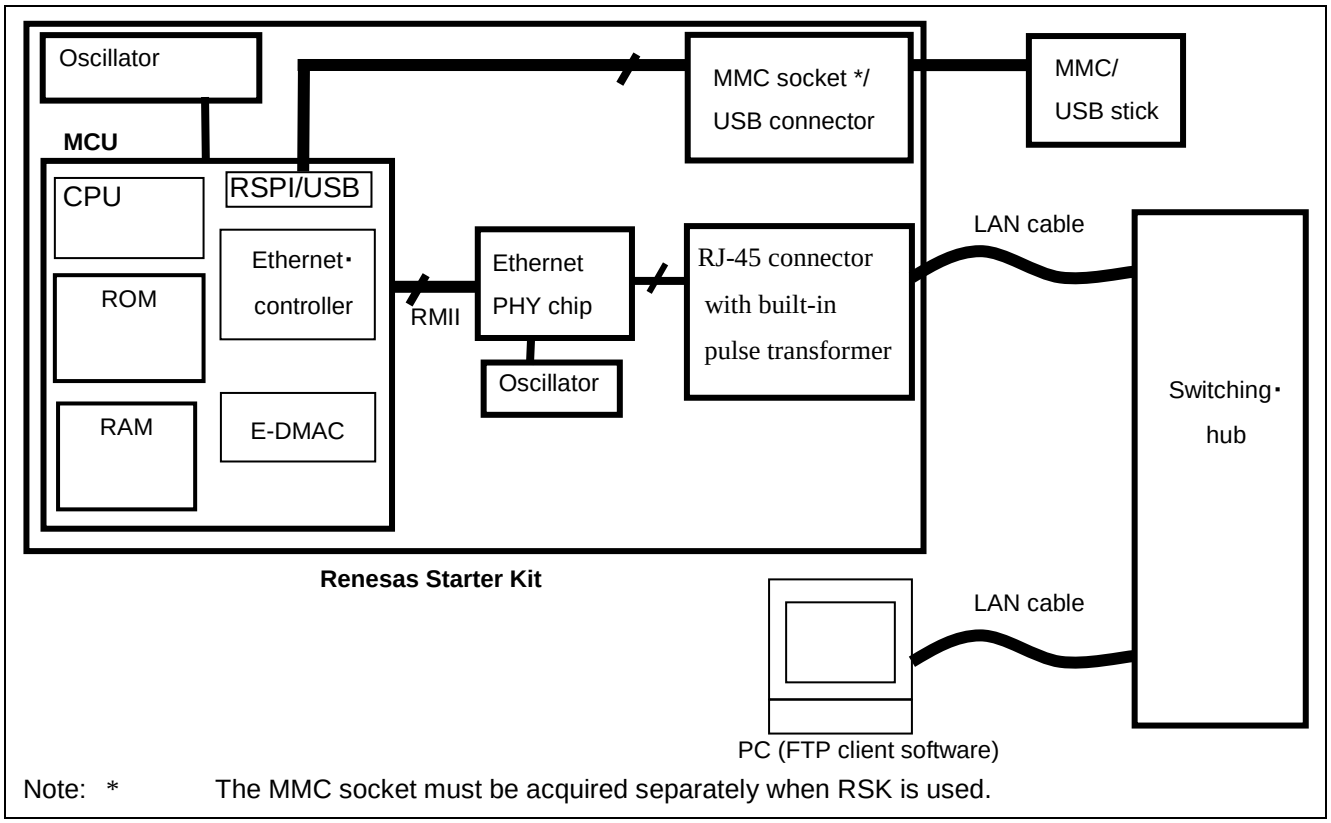

Figure 1 System Structure Example

1.2 Software Structure

Show Software Structure Example.

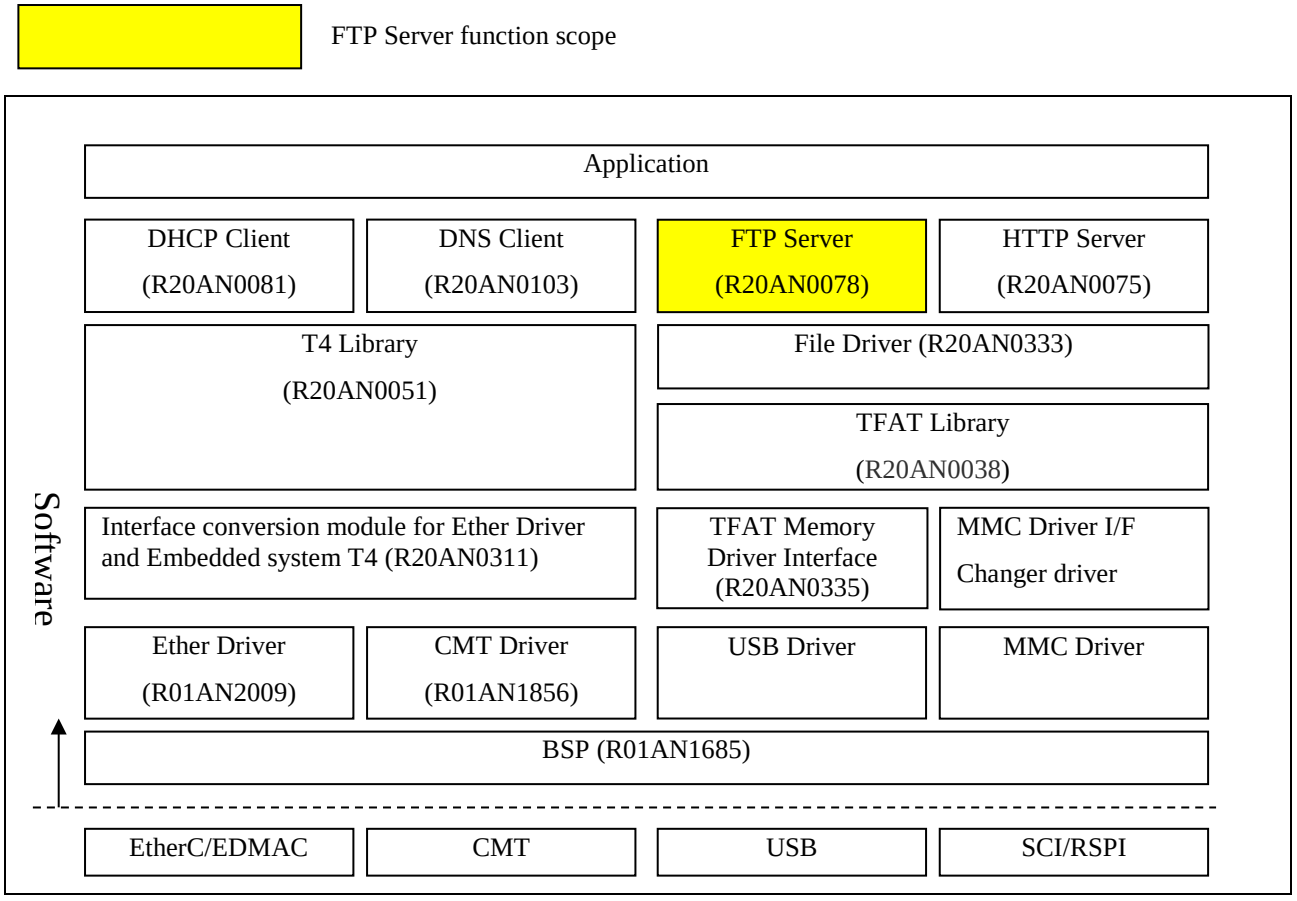

Figure 2 Software Structure Example（in case, store the web contents to MMC/USB memory）

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_ftp_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_ftp_server_rx_config.h”. A summary of these settings are provided in the following table:

Table 3 Configuration options

Configuration options in *r_t4_ftp_server_rx_config.h*	
#define FTP_TCP_CEP_NUM ※Default value is “6”.	Max communication endpoint number. The number of communication endpoint which is used by FTP server. Two communication endpoints are allocated for one user. This parameter should be set in same value number of communication endpoint specified in “config_tcpudp.c”.
#define FTP_START_TCP_CEP ※Default value is “0”.	The offset value of starting position of the communication endpoint in “config_tcpudp.c”.
#define FTP_MAX_FILE_LIST ※Default value is “10”.	Max file list number. At the NLST and LIST response data creation, it is maximum number of file (directory) information acquired from the file driver at a time. The processing speed improves because it can transmit a lot of information at a time by large the value. The acquired RAM size should be smaller than that of DATA_BUF_SIZE. The required RAM size = MAX_FILE_LIST * 65 (65 = max file information size per one file)
#define CMD_BUF_SIZE ※Default value is “272”.	Buffer size of command port. This is capacity of the buffer that the FTP server uses. The data transfer efficiency goes up by the value large.
#define DATA_BUF_SIZE ※Default value is “2560”.	Buffer size of data port. This is capacity of the buffer that the FTP server uses. The data transfer efficiency goes up by the value large.
#define MAX_USER ※Default value is “3”.	Max user number. The data specifies max user can connect to FTP server in same time.
#define LF_CODE ※Default value is “\r\n”.	Line feed code which FTP server uses.
#define PATH_NAME_SIZE ※Default value is “64”.	Limited size of path.
#define USER_LIST ※Default value is “{"user1"}, {"user2"}, {"user3"}”.	Login user name. The data specifies login user name. Login user name can be set 15 characters with ASCII code.
#define PASS_LIST ※Default value is “{"user01"}, {"user02"}, {"user03"}”.	Login user password. The data specifies login user password. Login user password can be set 15 characters with ASCII code.
#define ROOT_DIR_LIST ※Default value is “{"/"}, {"/"}, {"/"}”.	Login user root directory. The data specifies login user root directory. Login user root directory can be set 15 characters with ASCII code.

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_ftp_srv_open

This function initializes communication endpoint for FTP server.

Format

```
void R_ftp_srv_open(void)
```

Parameters

None

Return Value

None

Properties

Prototyped in file “r_t4_ftp_server_rx_if.h”.

Description

The application calls this function in initial sequence. This function initializes communication endpoint for FTP server.

Reentrant

No

Special Notes

None

3.2 R_ftpd

This function manages the sockets required for FTP communication.

Format

```
void R_ftpd(void)
```

Parameters

None

Return Value

None

Properties

Prototyped in file “r_t4_ftp_server_rx_if.h”.

Description

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

Reentrant

No

Special Notes

None

3.3 R_ftp_srv_close

This function releases communications endpoint data for FTP server.

Format

```
void R_ftp_srv_close(void)
```

Parameters

None

Return Value

None

Properties

Prototyped in file “r_t4_ftp_server_rx_if.h”.

Description

The application calls this function in the end of sequence. This function releases communication endpoint data for FTP server.

Reentrant

No.

Special Notes

None

3.4 R_T4_FTP_SERVER_GetVersion

This function returns the version number of FTP server.

Format

```
uint32_t      R_T4_FTP_SERVER_GetVersion (void)
```

Parameters

None

Return Value

Version number of FTP server

Properties

Prototyped in file “r_t4_ftp_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_ftp_server.c”.

4. File driver for FTP server and Web server Module

The FTP server calls these functions. The user must code the processing performed by these functions appropriately for the file system used. Also, the FTP 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

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. Internal function specification

FTP server calls these functions and uses data structure. These functions are corresponding FTP commands. User can add supported FTP command to add corresponded function in case user needs.

Table 5 Internal function

Name	Function
ftps_abor()	Abort communication
ftps_cdup()	Move to parent directory
ftps_cwd()	Move to current directory
ftps_dele()	Delete file
ftps_list_nlst()	Generate file list
ftps_mkd()	Make directory
ftps_noop()	No operation (only ACK)
ftps_pass()	Authentication (input password)
ftps_pasv()	Transition to passive mode
ftps_port()	Specify port number and IP address for FTP server to connect
ftps_pwd()	Request current directory
ftps_quit()	Disconnect to FTP server
ftps_retr()	Download file
ftps_rmd()	Delete directory
ftps_rnfr()	Get file name to rename
ftps_rnto()	Rename file
ftps_stor()	Upload file
ftps_type()	Set transition mode
ftps_user()	Authentication (input user name)

5.1 Data structures

【Communication endpoint structure】

```
typedef struct cep_
{
    uint8_t  status;
    uint8_t  *buff_ptr;
    int32_t  remain_data;
    int32_t  now_data;
    T_IPV4EP dstaddr;
    T_IPV4EP myaddr;
    uint8_t  api_cancel;
} FTP_CEP;
```

【FTP server structure】

```
typedef struct
{
    uint8_t   cmd_buff[CMD_BUF_SIZE];
    uint8_t   data_buff[DATA_BUF_SIZE];
    int8_t    current_path[PATH_NAME_SIZE];
    int16_t   valid_dstaddr;
    uint8_t   trans_mode;
    uint8_t   rnfr;
    int8_t    fname[PATH_NAME_SIZE];
    int32_t   file_index;
    int16_t   exec_command;
    int16_t   exec_command_subseq;
    int16_t   user_id;
    uint8_t   read_crlf_check;
    uint8_t   cep_reset_req;
    int32_t   dir_read_index;
    FILE_LIST file_list[FTP_MAX_FILE_LIST];
} _FTP_STAT;
```

5.2 ftps_abor

Description

When the "ABOR" command is received, the FTP server calls this function. The FTP server interrupts the data communication, and generates the result response.

Usage

```
static int16_t ftps_abor(int16_t argc, int8_t **argv, ID cepid);
```

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication endpoint number

Return Value

-1	Error
0 and positive integer	Normal completion

Remark

None

5.3 ftps_cdup

Description

When the "CDUP" command is received, the FTP server calls this function. The FTP server moves current directory to parent directory, and generates the result response.

Usage

```
static int16_t ftps_cdup(int16_t argc, int8_t **argv, ID cepid);
```

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication endpoint number

Return Value

-1	Error
0	Normal completion

Remark

None

5.4 ftps_cwd

Description

When the "CWD" command is received, the FTP server calls this function. The FTP server moves current directory to specified directory from client, and generates the result response.

Usage

static int16_t ftps_cwd(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.5 ftps_dele

Description

When the "DELE" command is received, the FTP server calls this function. The FTP server deletes file specified directory from client, and generates the result response.

Usage

static int16_t ftps_dele(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.6 ftps_list_nlst

Description

When the "LIST" or "NLST" commands are received, the FTP server calls this function. The FTP server generates file list information of current directory, and generates the result response.

Usage

static int16_t ftps_list_nlst(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

This function ignores file name or directory path which is in parameter.

5.7 ftps_mkd

Description

When the "MKD" command is received, the FTP server calls this function. The FTP server makes directory, and generates the result response.

Usage

static int16_t ftps_mkd(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.8 ftps_noop

Description

When the "NOOP" command is received, the FTP server calls this function. The FTP server generates the result response only. FTP client often generate “NOOP” command not to disconnect from FTP server because of timeout.

Usage

static int16_t ftps_noop(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.9 ftps_pass

Description

When the "PASS" command is received, the FTP server calls this function. The FTP server generates the result response. The FTP server authenticates the FTP client using “USER” and “PASS” command. The FTP server accepts only "USER" and "PASS" command until authentication successfully..

Usage

static int16_t ftps_pass(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.10 ftps_pasv

Description

When the "PASV" command is received, the FTP server calls this function. The FTP server generates communication endpoint information for FTP client to connect using passive mode, and the result response.

Usage

static int16_t ftps_pasv(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.11 ftps_port

Description

When the "PORT" command is received, the FTP server calls this function. The FTP server gets communication endpoint information from the FTP client to connect and generates the result response.

Usage

static int16_t ftps_port(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.12 ftps_quit

Description

When the "QUIT" command is received, the FTP server calls this function. The FTP server generates disconnect message as result response.

Usage

static int16_t ftps_quit(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.13 ftps_pwd

Description

When the "PWD" command is received, the FTP server calls this function. The FTP server generates current directory information as result response.

Usage

static int16_t ftps_pwd (int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.14 ftps_retr

Description

When the "RETR" command is received, the FTP server calls this function. The FTP server starts transmitting file specified from the FTP client (Download), and generates result response.

Usage

static int16_t ftps_retr(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.15 ftps_rmd

Description

When the "RMD" command is received, the FTP server calls this function. The FTP server deletes directory, and generates result response.

Usage

static int16_t ftps_rmd(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.16 ftps_rnfr

Description

When the "RNTO" command is received, the FTP server calls this function. The FTP server renames file to specified “RNTO” command from specified “RNFR” command, and generates result response.

Usage

static int16_t ftps_rnfr(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.17 ftps_rnto

Description

When the "RNTO" command is received, the FTP server calls this function. The FTP server renames file to specified “RNTO” command from specified “RNFR” command, and generates result response.

Usage

static int16_t ftps_rnto(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.18 ftps_stor

Description

When the "STOR" command is received, the FTP server calls this function. The FTP server starts transmitting file specified to the FTP client (Upload), and generates result response.

Usage

static int16_t ftps_stor(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.19 ftps_type

Description

When the "TYPE" command is received, the FTP server calls this function. The FTP server sets the transmission mode, and generates result response.

Usage

static int16_t ftps_type(int16_t argc, int8_t **argv, ID cepid);

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

5.20 ftps_user

Description

When the "USER" command is received, the FTP server calls this function. The FTP server generates the result response. The FTP server authenticates the FTP client using “USER” and “PASS” command. The FTP server accepts only "USER" and "PASS" command until authentication successfully.

Usage

```
static int16_t ftps_user(int16_t argc, int8_t **argv, ID cepid);
```

Parameters

argc	input	Number of reception message
argv	input	Address of reception message array
cepid	input	Reception communication end point number

Return Value

-1	Error
0	Normal completion

Remark

None

Website and Support

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

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

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.04	Oct.01.16	—	Updated the xml file for FIT.
1.03	Jan.05.15	1	Fixed FIT Modules URL Added Support MCUs.
		5	Fixed Figure 2.
1.02	May.09.14	—	Corresponded to FIT Modules.
1.01	Sep.27.12	4	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

Colophon 5.0