RX ファミリ
組み込み用 TCP/IP M3S-T4-Tiny を用いた FTP サーバ モジュール
Firmware Integration Technology

R20AN0078JJ0104
Rev.1.04
2016.10.01

要旨

本アプリケーションノートでは、組み込み用 TCP/IP M3S-T4-Tiny を用いた FTP サーバ(以降、FTP サーバ)を導入するための情報を提供します。

FTP サーバは、Firmware Integration Technology(FIT)として提供されます。FIT の概念については、以下の URL を参照してください。

https://www.renesas.com/ja-jp/solutions/rx-applications/fit.html

FTP サーバは、以下のミドルウェア製品と組み合わせて使用します。

表 1　ミドルウェア一覧

機能	ミドルウェア製品	ウェブページ(※1)
TCP/IP	M3S-T4-Tiny(以降、T4) (R20AN0051)	http://www.renesas.com/mw/t4
FTP/Web サーバ インタフェース	FTP/Web サーバ用ファイルドライバ (R20AN0333)	http://www.renesas.com/mw/t4
ファイルシステム	M3S-TFAT-Tiny) (R20AN0038)	http://www.renesas.com/mw/tfat
ファイルシステムインタフェース	M3S-TFAT-Tiny メモリドライバインタフェース (R20AN0335)	http://www.renesas.com/mw/tfat
MMC ドライバ	SPI モードマルチメディアカードドライバ(※2)	http://www.renesas.com/driver/mmc_sd http://www.renesas.com/mw/tfat http://www.renesas.com/mw/tfs
MMC 拡張 (ボード)	ミドルウェア評価ボード (※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 ドライバ	USB ドライバ	http://www.renesas.com/driver/usb

【注】※1 複数紹介のあるものは、関連のある各ミドルウェアのサイトからダウンロード出来るもので、ダウンロードできるアプリケーションノート自体には差はありません。
※2 MMC とのコマンド互換を持つ一部の SD カード(2GB 以下)は読み書き可能です。
※3 ミドルウェア評価ボードはアプリケーションノートを参考にしてユーザが作成する必要があります。

各ミドルウェアは独立しているので、それぞれのインタフェースプログラムをユーザが作成すれば任意のソフトウェアと組み合わせることが可能です。例えばファイルシステムだけを別のものに置き換えたり、MMC ドライバの代わりに USB ドライバに置き換えたりすることが出来ます。

また、FTP サーバプログラム自体もマイコンに依存したプログラムコードを含んでいないので、TCP/IP 以下のソフトウェアスタックを別マイコン用のものに置き換えることで容易に別マイコンに移植することが可能です。

各種 Renesas Starter Kit 上で動作するサンプルプログラムを用意しています。詳細は、Renesas Starter Kit 用サンプルアプリケーションノートを参照してください。

表 2　サンプルアプリケーションノート

サンプルアプリケーションノート	ドキュメント番号	ウェブページ
T4 を使った応用例 (DHCP/DNS/FTP/HTTP) Firmware Integration Technology	R20AN0314	https://www.renesas.com/mw/t4

動作確認デバイス

RX ファミリ

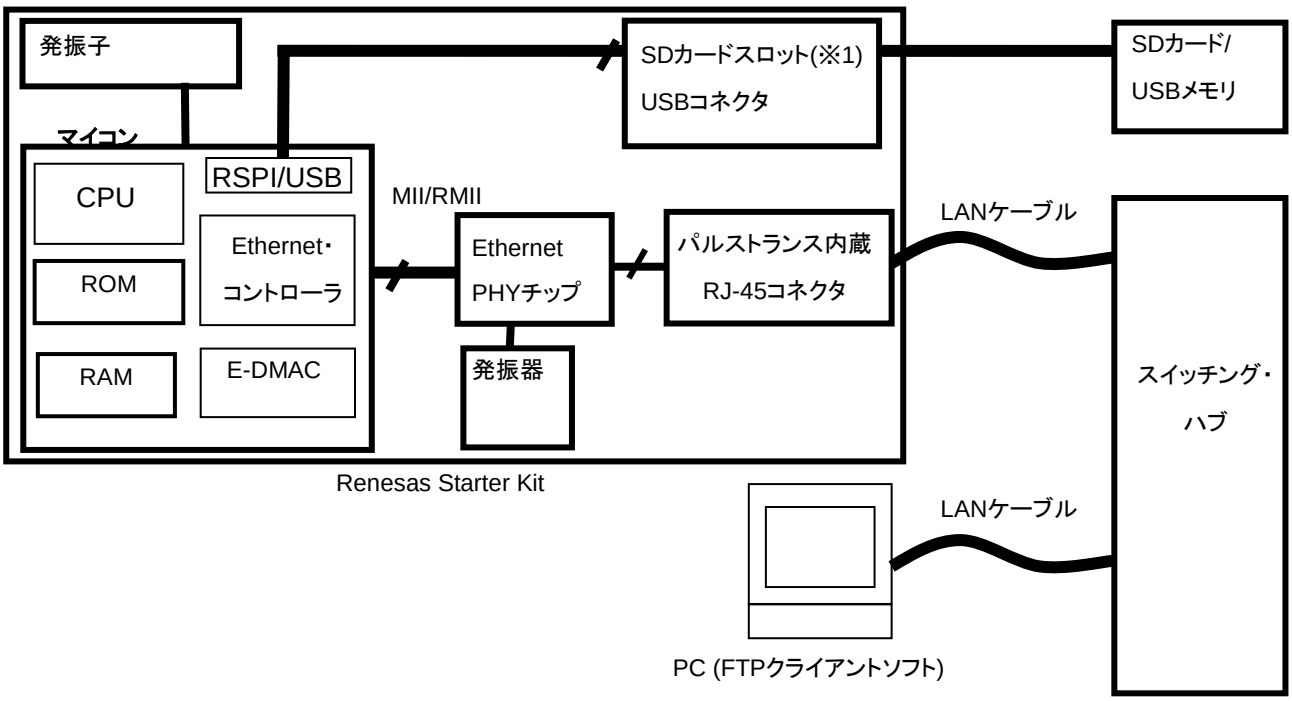
目次

1. 概要

FTP サーバは TCP/IP 上で動作するアプリケーションプログラムであり、FTP サーバ上に保存されているファイルを TCP/IP を用いて FTP クライアント、または FTP クライアントが指定するアドレスに転送する機能を提供します。

1.1 システム構成

システム構成例を示します。



【注】 ※1　RX62N CPU ボードまたは RX63N CPU ボードを利用する場合、SD カード拡張基板(CK-35)が別途必要です

図 1　システム構成例

1.2　ソフトウェア構成

ソフトウェア構成例を示します。

FTP サーバ機能適用範囲

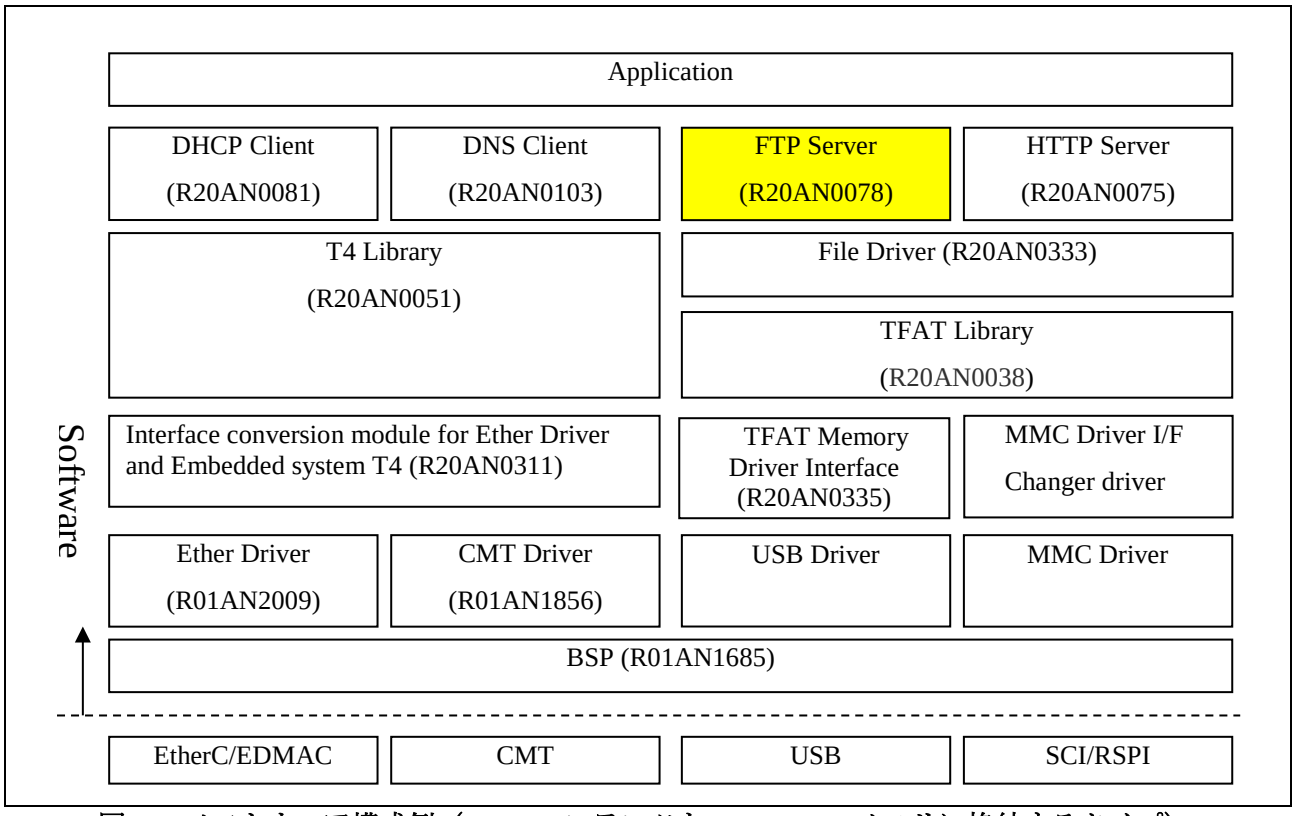


図 2　ソフトウェア構成例（**WEB** コンテンツを **USB/MMC** メモリに格納するタイプ）

2. API 情報

2.1 ハードウェアの要求

なし

2.2 ソフトウェアの要求

本 FIT モジュールは、以下のパッケージに依存しています。

r_t4_rx

r_t4_file_driver_rx

2.3 サポートされているツールチェイン

本 FIT モジュールは、以下のツールチェインで動作を確認しています。

Renesas RXC Toolchain v.2.04.01

2.4 制限事項

本プログラムは、stdio.h、stdlib.h、string.h、ctype.h を使用しています。ユーザプログラムでコンパイルオプションに”stdio”、 ”stdlib”、”string”、”ctype”を指定してください。

2.5 ヘッダファイル

すべての API 呼び出しとそれをサポートするインタフェース定義は r_t4_ftp_server_rx_if.h に記載しています。

2.6　コンパイル時の設定

本モジュールのコンフィギュレーションオプションの設定は、r_t4_ftp_server_rx_config.h で行います。

本オプション名および設定値に関する説明を、下表に示します。

表 3　オプション一覧

Configuration options in *r_t4_ftp_server_rx_config.h*	
#define FTP_TCP_CEP_NUM ※デフォルトは“6”	通信端点数 FTP サーバが使用する通信端点数です。１ユーザは２個通信端点を使用します。config_tcpudp.c で定義されている FTP サーバの通信端点の個数と合わせてください。
#define FTP_START_TCP_CEP ※デフォルトは“0”	config_tcpudp.c で定義されている FTP サーバの通信端点の開始位置のオフセット
#define FTP_MAX_FILE_LIST ※デフォルトは“10”	最大ファイルリスト数 LIST、NLST 応答データ作成時、ファイルドライバから一度に取得するファイル（ディレクトリ）情報の最大数です。値が大きいほど一度に多くの情報を転送出来るため処理速度が向上します。必要 RAM サイズは DATA_BUF_SIZE より小さい必要があります。 必要 RAM サイズ＝MAX_FILE_LIST×65 （65=1 ファイルあたりの最大ファイル情報サイズ）
#define CMD_BUF_SIZE ※デフォルトは“272”	コマンドポート用バッファサイズ FTP サーバが使用するバッファ容量です。値が大きいほどデータ転送効率が上がります。
#define DATA_BUF_SIZE ※デフォルトは“2560”	データポート用バッファサイズ FTP サーバが使用するバッファ容量です。値が大きいほどデータ転送効率が上がります。
#define MAX_USER ※デフォルトは“3”	最大ユーザ数 FTP サーバに同時に接続することができるユーザ数を指定することが出来ます。
#define LF_CODE ※デフォルトは“¥r¥n”	FTP サーバが使用する改行コード
#define PATH_NAME_SIZE ※デフォルトは“64”	パス制限サイズ
#define USER_LIST ※デフォルトは“{"user1"},　{"user2"}, {"user3"}”	ログインユーザ名 ユーザ名を指定します。ユーザ名は ASCII コードで 15 文字まで設定可能です。
#define PASS_LIST ※デフォルトは“{"user01"}, {"user02"}, {"user03"}”	ログインユーザパスワード ユーザパスワードを指定します。ユーザパスワードは ASCII コードで 15 文字まで設定可能です。
#define ROOT_DIR_LIST ※デフォルトは“{"/"}, {"/"}, {"/"}”	ログインユーザルートディレクトリ ユーザのルートディレクトリを指定します。ユーザのルートディレクトリは ASCII コードで 15 文字まで設定可能です。

2.7　モジュールの追加方法

e² studio/CS+に組み込む方法は、”r01an1723ju0111_rx.pdf”(e² studio)か”r01an1826jj0102_rx.pdf”(cs+)をご参照ください。

3. API 関数

3.1 R_ftp_srv_open

FTP の通信に必要な通信端点の初期化を実行するための関数です。

Format

```
void R_ftp_srv_open(void)
```

Parameters

なし

Return Value

なし

Properties

r_t4_ftp_server_rx_if.h にプロトタイプ宣言がされています。

Description

アプリケーションは本関数をシステム起動時に呼び出します。本関数は、FTP の通信に必要な通信端点の初期化を実行します。

Reentrant

非対応

Special Notes

なし

3.2　R_ftpd

FTP の通信に必要な通信端点を管理するための関数です。

Format

```
void R_ftpd(void)
```

Parameters

なし

Return Value

なし

Properties

r_t4_ftp_server_rx_if.h にプロトタイプ宣言がされています。

Description

アプリケーションは本関数を定期的に呼び出します。本関数は、FTP の通信に必要な通信端点を管理します。本関数は通信端点の管理のみを行い、通信自体は T4 が割り込み駆動により自動的に行います。

Reentrant

非対応

Special Notes

なし

3.3　R_ftp_srv_close

FTP の通信に使用した通信端点のデータを開放するための関数です。

Format

```
void R_ftp_srv_close(void)
```

Parameters

なし

Return Value

なし

Properties

r_t4_ftp_server_rx_if.h にプロトタイプ宣言がされています。

Description

アプリケーションは本関数をシステム終了時に呼び出します。本関数は FTP の通信に使用した通信端点のデータを開放します

Reentrant

非対応

Special Notes

なし

3.4　R_T4_FTP_SERVER_GetVersion

FTP サーバのバージョンを返します。

Format

```
uint32_t     R_T4_FTP_SERVER_GetVersion (void)
```

Parameters

なし

Return Value

FTP サーバのバージョン

Properties

r_t4_ftp_server_rx_if.h にプロトタイプ宣言がされています。

Description

本関数は、現在インストールされているモジュールのバージョンを返します。バージョン番号はコード化されています。最初の 2 バイトがメジャーバージョン番号で、後の 2 バイトがマイナーバージョン番号です。例えば、バージョンが 4.25 の場合、戻り値は ‘0x00040019’ となります。

Reentrant

対応

Special Notes

本関数は、r_ftp_server.c にインライン関数として定義されています。

4. FTP/Web サーバ用ファイルドライバモジュール

FTP サーバは本関数群を呼び出します。ユーザはファイルシステムに応じて適切に本関数の処理内容を定義します。また、FTP サーバは本データ構造体を使用し、外部メモリの情報を取得することが出来ます。

表 4　関数一覧

関数名	機能概要
change_dir()	作業ディレクトリの変更
file_close()	ファイルのクローズ
file_delete()	ファイルの削除
file_open()	ファイルのオープン
file_read()	ファイルの読み込み
file_rename()	ファイル名の変更
file_exist()	ファイルの有無を確認
file_write()	ファイルの書き込み
get_file_info()	ファイル情報の取得
get_file_list_info()	ファイルリストの取得
get_file_size()	ファイルサイズの取得
make_dir()	ディレクトリの作成
remove_dir()	ディレクトリの削除

4.1　データ構造体

【日付情報構造体】

```
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;
```

【ファイルリスト構造体】

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

【マクロ定義】

```
#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

本関数は引数で指定されたディレクトリパスを作業ディレクトリに設定します。ディレクトリパスはフルパスで指定します。作業ディレクトリの情報は、通信端点毎に管理されます。

Usage

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

Parameters

dir_path	入力	指定されたディレクトリパスの格納先

Return Value

-1	ディレクトリが存在しない
0	ディレクトリが存在する

Remark

dir_path の終端には’/’が付く場合と付かない場合があります。使用するファイルシステムにあわせて’/’の有無を調整してください。

4.3 file_close

Description

本関数は引数で指定された ID 値に対応するファイルをクローズし、管理情報を破棄します。

Usage

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

Parameters

file_id	入力	クローズするファイルの ID 値

Return Value

-1	エラー
0	正常終了

Remark

無し

4.4 file_delete

Description

本関数は引数で指定されたファイルを削除します。ファイルの指定はルートディレクトリからのフルパスで指定します。

Usage

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

Parameters

file_path	入力	ファイルのフルパスの格納先

Return Value

-1	エラー
0	正常終了

Remark

無し

4.5 file_open

Description

本関数は第 1 引数で指定されたファイルを第 2 引数で指定されたモードでオープンし、管理情報を独自で保存します。また、保存した管理情報を FTP サーバが ID 参照できるように、戻り値として管理情報の ID 値を指定します。保存した管理情報はファイルのクローズ関数で ID 値が指定されるまで保持しなければなりません。

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	入力	ファイルのフルパスの格納先
mode_flag	入力	ファイルオープンのモード (FILE_WRITE または FILE_READ)

Return Value

-1	エラー
0 以上	オープンしたファイルの ID 値

Remark

ファイルオープンの状態はファイルのクローズ関数で対応する ID 値が指定されるまで保持しなければなりません。

4.6 file_read

Description

本関数は第 1 引数で指定された ID 値に対応するファイルデータを、第 2 引数が示すアドレスに、最大第 3 引数の値で示すサイズ分読み込まれます。第 1 引数の ID 値に対応する管理情報内のファイルポインタは読み込んだ分だけ更新され、ファイルのクローズ関数が呼び出されるまで保持します。

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	入力	読み込むファイルの ID 値
buf	出力	読み込んだファイルデータの格納先
read_size	入力	読み込むファイルサイズ

Return Value

-1	エラー
0 以上	読み込んだデータサイズ

Remark

無し

4.7 file_rename

Description

本関数は第 1 引数で指定されたファイルまたはディレクトリを第 2 引数で指定された名前に変更します。第 1 引数、第 2 引数ともにルートディレクトリからのフルパスで指定します。

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	入力	変更対象のファイルまたはディレクトリ
new_name	入力	変更後の名前

Return Value

-1	エラー
0	正常終了

Remark

無し

4.8 file_exist

Description

本関数は引数で指定されたファイルまたはディレクトリの有無を確認します。引数はルートディレクトリからのフルパスで指定します。

Usage

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

Parameters

file_path	入力	有無を確認するファイルまたはディレクトリ

Return Value

-1	存在しない
0	存在する

Remark

無し

4.9 file_write

Description

本関数は第 1 引数で指定された ID 値に対応するファイルに対し、第 2 引数で指定されたアドレスから第 3 引数で指定されたサイズ分のデータを書き込みます。第 1 引数の ID 値に対応する管理情報内のファイルポインタは書き込んだ分だけ更新され、ファイルのクローズ関数が呼び出されるまで保持します。

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	入力	書き込むファイルの ID 値
buf	入力	書き込むデータの先頭アドレス
write_size	入力	書き込むサイズ

Return Value

-1	エラー
0	正常終了

Remark

無し

4.10 get_file_info

Description

本関数は第 1 引数で指定された ID 値に対応するファイルの管理情報を読み込み、ファイルの日付情報を第 2 引数で示す日付情報構造体に書き出します。

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	入力	読み込むファイルの ID 値
date_info	出力	日付情報の格納先

Return Value

-1	エラー
0	正常終了

Remark

無し

4.11 get_file_list_info

Description

本関数は第 1 引数で指定されたディレクトリパスに格納されているファイルまたはディレクトリの情報を第 2 引数で指定されたファイルリスト構造体に書き出します。一度に書き出す最大情報個数は第 3 引数で指定し、第 4 引数でファイルリストの読み出し開始位置を指定します。

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	入力	読み出すディレクトリパスの格納先
file_list	出力	読み出したファイルリストの格納先 リストの最後にはファイル名格納領域の先頭に’¥0’を格納します。
num_file_list	入力	一度に読み出すファイルリスト情報の最大個数
read_index	入力	ファイルリストの読み出し開始位置

Return Value

-1	エラー
0 以上	読み出したファイルの個数

Remark

戻り値が num_file_list より小さい値を返した場合は、ファイルリスト情報の読み出しが終了したことを示し、num_file_list と同じ値を返した場合はファイルリスト情報に続きがあることを示します。本関数はファイルリストの続きを読み出す際に、read_index にファイルリストの読み出し開始位置を指定して呼び出します。

dir_path の終端には’/’が付く場合と付かない場合があります。使用するファイルシステムにあわせて’/’の有無を調整してください。

4.12 get_file_size

Description

本関数は引数で指定された ID 値に対応するファイルの管理情報を読み込み、ファイルサイズを返します。

Usage

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

Parameters

file_id	入力	読み込むファイルの ID 値

Return Value

-1	エラー
0 以上	ファイルサイズ

Remark

無し

4.13 make_dir

Description

本関数は引数で指定されたディレクトリを作成します。ディレクトリパスはフルパスで指定します。

Usage

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

Parameters

dir_path	入力	作成するディレクトリ名

Return Value

-1	エラー
0	正常終了

Remark

dir_path の終端には’/’が付く場合と付かない場合があります。使用するファイルシステムにあわせて’/’の有無を調整してください。

4.14 remove_dir

Description

本関数は引数で指定されたディレクトリを削除します。ディレクトリパスはフルパスで指定します。

Usage

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

Parameters

dir_path	入力	削除するディレクトリ名

Return Value

-1	エラー
0	正常終了

Remark

dir_path の終端には’/’が付く場合と付かない場合があります。使用するファイルシステムにあわせて’/’の有無を調整してください。

5. 内部関数リファレンス

FTP サーバは本関数群及びデータ構造体を内部的に呼び出します。本関数群は、FTP コマンドに対応した関数になっています。FTP コマンドに対応した関数を追加することで、必要に応じて本プログラムが対応する FTP コマンドを追加することが可能です。

表 5　内部関数一覧

関数名	機能概要
ftps_abor()	転送中断
ftps_cdup()	親ディレクトリへの移動
ftps_cwd()	作業ディレクトリの移動
ftps_dele()	ファイル削除
ftps_list_nlst()	ファイルリスト作成
ftps_mkd()	ディレクトリ作成
ftps_noop()	何もしない(確認応答するのみ)
ftps_pass()	パスワード指定
ftps_pasv()	パッシブモードに移行
ftps_port()	サーバが接続する IP アドレスとポート設定
ftps_pwd()	作業ディレクトリ名要求
ftps_quit()	接続終了
ftps_retr()	ファイルダウンロード
ftps_rmd()	ディレクトリ削除
ftps_rnfr()	変更対象ファイル名取得
ftps_rnto()	ファイル名変更
ftps_stor()	ファイルアップロード
ftps_type()	転送モード設定
ftps_user()	ユーザ名指定

5.1　データ構造体

【通信端点管理構造体】

```
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 サーバ管理構造体】

```
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

FTP サーバは「ABOR」コマンドを受信するとこの関数を呼び出します。データ通信を中断し、結果応答を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0 以上	正常終了

Remark

無し

5.3 ftps_cdup

Description

FTP サーバは「CDUP」コマンドを受信するとこの関数を呼び出します。作業ディレクトリを親ディレクトリに移動し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.4 ftps_cwd

Description

FTP サーバは「CWD」コマンドを受信するとこの関数を呼び出します。作業ディレクトリを指定されたディレクトリに変更し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.5 ftps_dele

Description

FTP サーバは「DELE」コマンドを受信するとこの関数を呼び出します。指定されたファイルを削除し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.6 ftps_list_nlst

Description

FTP サーバは「LIST」または「NLST」コマンドを受信するとこの関数を呼び出します。作業ディレクトリのファイルリストデータを作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

引数にファイル名やディレクトリパスが設定されていても無視します。

5.7 ftps_mkd

Description

FTP サーバは「MKD」コマンドを受信するとこの関数を呼び出します。指定された、ディレクトリを作成し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.8 ftps_noop

Description

FTP サーバは「NOOP」コマンドを受信するとこの関数を呼び出します。FTP クライアントはタイムアウトによるコネクションの切断防止のため、定期的に「NOOP」コマンドを発行することがあります。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.9 ftps_pass

Description

FTP サーバは「PASS」コマンドを受信するとこの関数を呼び出します。「USER」で指定されたユーザを「PASS」で指定されたパスワードで認証し、結果情報を作成します。認証が正常終了するまでは、FTP サーバは「USER」と「PASS」コマンドしか受け付けません。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.10 ftps_pasv

Description

FTP サーバは「PASV」コマンドを受信するとこの関数を呼び出します。パッシブモードでのデータ通信を行うために、クライアントへのデータ通信端点情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.11 ftps_port

Description

FTP サーバは「PORT」コマンドを受信するとこの関数を呼び出します。指定されたクライアント側データ通信端点情報を取り込み、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.12 ftps_quit

Description

FTP サーバは「QUIT」コマンドを受信するとこの関数を呼び出します。コネクション終了メッセージを作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.13 ftps_pwd

Description

FTP サーバは「PWD」コマンドを受信するとこの関数を呼び出します。現在の作業ディレクトリ名を応答データに設定します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.14 ftps_retr

Description

FTP サーバは「RETR」コマンドを受信するとこの関数を呼び出します。指定されたファイルのクライアントへの送信(ダウンロード)を行います。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.15 ftps_rmd

Description

FTP サーバは「RMD」コマンドを受信するとこの関数を呼び出します。指定されたディレクトリを削除し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.16 ftps_rnfr

Description

FTP サーバは「RNFR」コマンドを受信するとこの関数を呼び出します。「RNFR」コマンドで指定されたファイルを「RNTO」コマンドで指定された名称に変更し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.17 ftps_rnto

Description

FTP サーバは「RNTO」コマンドを受信するとこの関数を呼び出します。「RNFR」コマンドで指定されたファイルを「RNTO」コマンドで指定された名称に変更し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.18 ftps_stor

Description

FTP サーバは「STOR」コマンドを受信するとこの関数を呼び出します。指定されたファイル名でファイルを作成し、データ受信(アップロード)を行い、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.19 ftps_type

Description

FTP サーバは「TYPE」コマンドを受信するとこの関数を呼び出します。データ送受信の転送モードを設定し、結果情報を作成します。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

5.20 ftps_user

Description

FTP サーバは「USER」コマンドを受信するとこの関数を呼び出します。「USER」で指定されたユーザを「PASS」で指定されたパスワードで認証し、結果情報を作成します。認証が正常終了するまでは、FTP サーバは「USER」と「PASS」コマンドしか受け付けません。

Usage

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

Parameters

argc	入力	受信したメッセージ数
argv	入力	受信したメッセージの個々のアドレス
cepid	入力	メッセージを受信した通信端点番号

Return Value

-1	エラー
0	正常終了

Remark

無し

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://www.renesas.com/

お問合せ先
http://www.renesas.com/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.04	2016.10.01	—	FIT 用 xml ファイルを更新しました。
1.03	2014.07.01	1	FIT モジュールの URL を修正しました。 サポートマイコンを追加しました。
		5	図 2 を修正しました
1.02	2014.05.09	—	FIT モジュール化
1.01	2012.09.27	4	USB メモリについて情報を追加
1.00	2011.04.12	—	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレス（予約領域）のアクセス禁止
   【注意】リザーブアドレス（予約領域）のアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子
   （または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、
   各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、
   家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通用信号機器、
   防災・防犯装置、各種安全装置等
   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 本資料に記載されている当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。
注2. 本資料において使用されている「当社製品」とは、注１において定義された当社の開発、製造製品をいいます。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社　〒135-0061　東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/