RX ファミリ

イーサネットコントローラ：簡易スイッチ動作

R01AN3036JJ0111
Rev.1.11
2016.11.11

要旨

本アプリケーションノートでは、RX64M/71Mに搭載したイーサネット周辺モジュールに実装した簡易スイッチ機能をEPTPC Light FIT (Firmware Integration Technology) モジュール[1]を使用し、動作させる例を説明します。簡易スイッチ機能はイーサネットMACモジュールのポート間の転送を制御し、転送方式はストア&フォワード方式、またはカットスルー方式から選択し動作させることができます。

動作確認デバイス

この API は下記のデバイスをサポートしています。

- RX64M グループ
- RX71M グループ

本アプリケーションノートを他のルネサスマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

目次

1. 概要

本アプリケーションノートでは、RX64M/71Mに搭載したイーサネット周辺モジュールに実装した簡易スイッチ機能をEPTPC Light FITモジュール(以後、PTP簡易ドライバ)を使用し、動作させる例を説明します。簡易スイッチ機能はイーサネットMACモジュールのポート間の転送を制御し、転送方式はストア&フォワード方式、またはカットスルー方式から選択し動作させることができます。

プロジェクトはスイッチプロジェクトとテストプロジェクトの 2 種で構成します。スイッチプロジェクトは 1 つのポートから受信したテストフレームを簡易スイッチ機能により処理します。テストプロジェクトは USB メモリから読み出したデータの内容を基にテストフレームを生成後、送信します。スイッチプロジェクトが実装されている RX64M/71M により中継されるフレームが存在する場合、テストプロジェクトは中継されたフレームを受信し、送信から受信までの伝播と処理に要した時間を測定します。

1.1 FIT モジュールを使用した簡易スイッチ動作

このサンプルプログラムはプロジェクト形式で提供しており、PTP 簡易ドライバを使用したイーサネットモジュールの簡易スイッチ動作例として使用できます。

1.2 関連ドキュメント

[1] RX ファミリ EPTPC Light モジュール Firmware Integration Technology, Rev.1.11, Document No. R01AN3035JJ0111, Nov 11, 2016
[2] RX ファミリ イーサネットモジュール Firmware Integration Technology, Rev.1.12, Document No. R01AN2009JJ0112, Nov 11, 2016
[3] RX Family Flash Module Using Firmware Integration Technology, Rev.1.20, Document No. R01AN2184EU0120, Dec 22, 2014
[4] Renesas USB MCU USB Basic Host and Peripheral firmware using Firmware Integration Technology, Rev.1.10, Document No. R01AN2025EJ0110, Dec 26, 2015
[5] Renesas USB MCU USB Host Mass Storage Class Driver (HMSC) using Firmware Integration Technology, Rev.1.10, Document No. R01AN2026EJ0110, Dec 26, 2015
[6] RX ファミリ オープンソース FAT ファイルシステム M3S-TFAT-Tiny モジュール Firmware Integration Technology, Rev.3.00, Document No. R20AN0038JJ0300, Apr.01, 2014
[7] RX ファミリ EPTPC モジュール Firmware Integration Technology, Rev.1.12, Document No. R01AN1943JJ0112, Nov 11, 2016
[8] Renesas Starter Kit+ for RX64M, ユーザーズマニュアル、Rev.1.20, Document No. R20UT2590JG0102, Jun 25, 2015
[9] Renesas Starter Kit+ for RX71M, ユーザーズマニュアル、Rev.1.00, Document No. R20UT3217JG0100, Jan 23, 2015

1.3 ハードウェアの構成

このサンプルプログラムは RX64M/71M の周辺ハードウェアモジュールを使用します。イーサネット周辺モジュールは EPTPC、PTP 用イーサネットコントローラ用 DMA コントローラ（PTPEDMAC）、2 チャネルのイーサネットコントローラ（ETHERC（CH0）、ETHERC（CH1））、および 2 チャネルのイーサネットコントローラ用 DMA コントローラ（EDMAC（CH0）、EDMAC（CH1））で構成しています。

さらに、テストプロジェクトでは、USB モジュール（USB）、マルチファンクションタイマパルスユニット（MTU3）モジュール、データフラッシュを使用します。

詳細に関しては「RX64M/71M グループユーザーズマニュアル ハードウェア編」を参照ください。

1.4　ソフトウェアの構成

このサンプルプログラムは複数の FIT モジュールを使用したアプリケーション層での実行例です。アプリケーションはスイッチプロジェクトとテストプロジェクトで異なります。スイッチプロジェクトは動作シーケンスの制御と管理、及び簡易スイッチの設定をします。テストプロジェクトは全体の動作シーケンス制御と管理、MAC アドレスのデータフラッシュからの読み出しとデータフラッシュへの書き込み、標準イーサネットフレーム（以後、フレーム）の生成と比較、送信データの読み出し、転送された受信データと転送結果の書き込みと保存をします。イーサネットドライバ[2]は個々のチャネル毎にフレームの送受信、MTU3 ドライバはポート間で転送されたフレームの伝播時間と処理時間の測定、データフラッシュドライバ（フラッシュ API）[3]は通信パラメータのデータフラッシュ上の通信パラメータ[1]の消去と書き込みを、それぞれ行います。USB ホストドライバ[4], [5]は USB メモリに論理ブロック単位でアクセスし、FAT ファイルシステム（M3S-TFAT-Tiny）[6]は USB ホストドライバを使用し USB メモリ内のデータをファイルとして管理します。図 1.1にソフトウェアの典型的な構成と機能概要を示します。

[1] この例では MAC アドレスです。

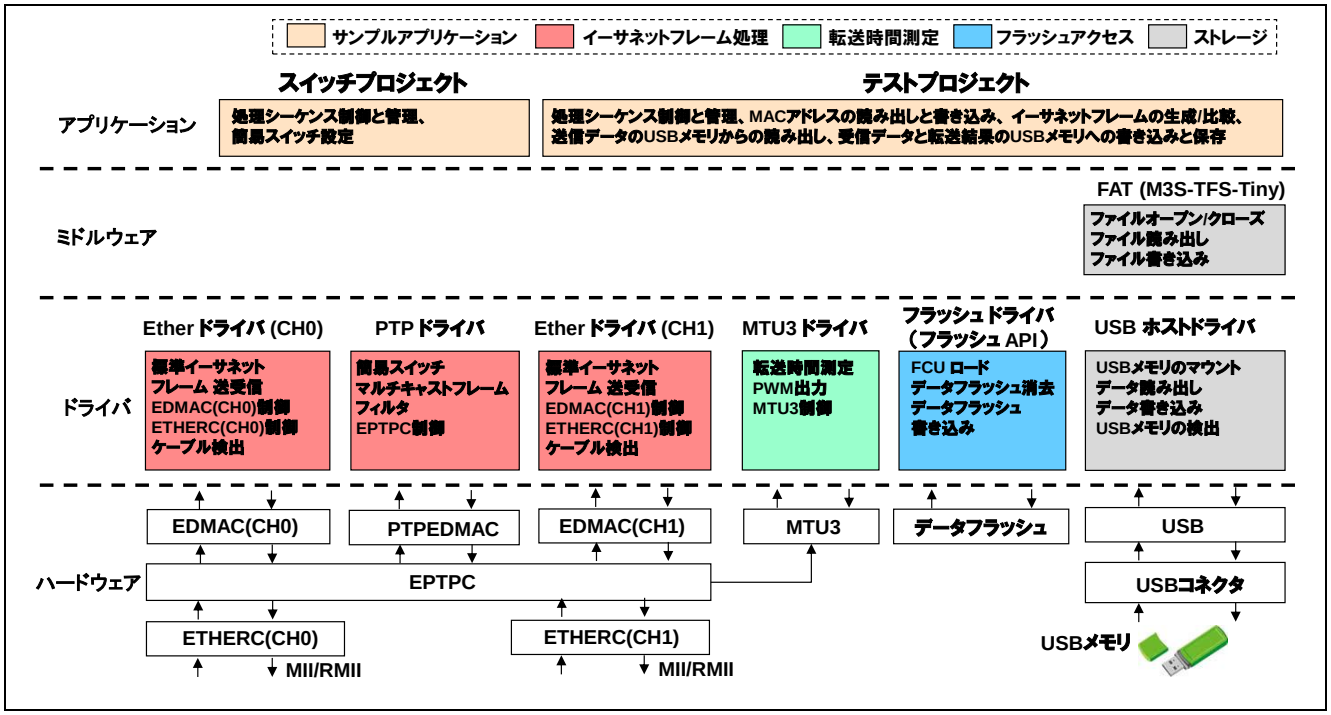

図1.1　ソフトウェア構成

1.5　ファイル構成

このサンプルプログラムのコードは demo_src とその下位階層のフォルダに格納しています。スイッチプロジェクトとテストプロジェクトのソースファイルとヘッダファイルの構成を、それぞれ図 1.2と図 1.3に示します。また、テストプロジェクトには USB メモリに格納するサンプルデータも含みます。FIT モジュール（BSP、イーサネットドライバ、フラッシュドライバ、PTP 簡易ドライバ、FAT ファイルシステム、USB ドライバ）の詳細は、個々の FIT モジュールのドキュメントを参照してください。

```
demo_src: メイン動作
|  sample_main.c
|  sample_main.h
|
+ --- usr: LED制御
|      led.c
|      led.h
|
```

```
r_bsp: BSP (Board Support Package) FITモジュール

r_config: FITモジュールの構成設定
|      r_bsp_config.h
|      r_bsp_interrupt_config.h
|      r_ether_rx_config.h
|      r_ptp_light_rx_config.h

r_ether_rx: イーサネットドライバ FITモジュール

r_ptp_light_rx: PTP簡易ドライバ FITモジュール
```

図1.2　ファイル構成（スイッチプロジェクト）

```
demo_src: メイン動作
|  sample_main.c
|  sample_main.h
|
+ --- ether_if: フレーム処理
|      ether_if.c
|      ether_if.h
|
+ --- flash_if: データフラッシュアクセス処理
|      flash_if.c
|      flash_if.h
|
+ --- tfat_if: ファイルシステムとUSBドライバ間のインタフェース
|      file_if.c
|      file_if.h
|      r_data_file.c
|      r_data_file.h
|      r_tfat_drv_if.c ;USBドライバインタフェース
|
+ --- tmr_if: タイマ（MTU3）制御
|      tmr_if.c
|      tmr_if.h
|
+ --- usb_if: USBホスト メモリアクセス制御
|      r_usb_hmsc_defep.c
|      usb_memory_access.c
|
+ --- usr: LED制御
|      led.c
|      led.h
|
+ --- usb_memory_sample: USBメモリのデータサンプル（送信フレームのデータ部分）
|      + --- TEST: SRC_X.txt; 格納データファイル (X = 0,1,2,, 200)
```

```
r_bsp: BSP (Board Support Package) FITモジュール

r_config: FITモジュールの構成設定
|      r_bsp_config.h
|      r_bsp_interrupt_config.h
|      r_ether_rx_config.h
|      r_flash_rx_config.h
|      r_ptp_light_rx_config.h
|      r_usb_basic_config.h
|      r_usb_hmsc_config.h

r_ether_rx: イーサネットドライバ FITモジュール

r_flash_rx: フラッシュドライバ（フラッシュAPI） FITモジュール

r_ptp_light_rx: PTP簡易ドライバ FITモジュール

r_tfat_rx: FATファイルシステム (M3S-TFS-Tiny) FITモジュール
|      r_tfat_lib.h ;FATライブラリ　ヘッダファイル
|      + --- lib:  FATライブラリ　格納フォルダ
|              r_mw_version.h ; ミドルウェア　バージョン情報
|              r_stdint.h ; 整数型　定義
|              tfat_rx600_big.lib ; ライブラリ（ビッグエンディアン用）
|              tfat_rx600_little.lib ; ライブラリ（リトルエンディアン用）
|

r_usb_basic: USBドライバ（USB基本処理） FITモジュール

r_usb_hmsc: USBドライバ（ホストマスストレージクラス） FITモジュール
```

図1.3　ファイル構成（テストプロジェクト）

2. 機能情報

本サンプルプログラムは以下の要件で開発しています。

2.1 ハードウェア要件

サンプルプログラムは、使用する MCU が以下の機能をサポートしている必要があります。

- EPTPC
- ETHERC
- EDMAC
- MTU3[1]
- データフラッシュ [1]
- USB[1]

[1]テストプロジェクトのみで使用しています。

2.2 ハードウェアリソース要件

サンプルプログラムで使用するドライバが必要とする周辺回路ハードウェアについて説明します。特に明記されていない限り、周辺回路はドライバで制御し、ユーザアプリケーションから直接制御し、使用することはできません。

2.2.1 EPTPC チャネル

この例では EPTPC を使用します。この周辺モジュールは簡易スイッチ機能に必要です。

2.2.2 ETHERC チャネル

この例では ETHEC（CH0）と ETHEC（CH1）を使用します。これらの周辺モジュールはイーサネット MAC 動作に必要です。

2.2.3 EDMAC チャネル

この例では、EDMAC（CH0）と EDMAC（CH1）を使用します。これらの周辺モジュールはフレーム送受信での CPU ホストインタフェースとして必要です。

2.2.4 MTU3 チャネル

この例では、MTU3（CH1）と MTU3（CH2）をポート間のフレーム転送時間の測定に使用します。なお、MTU3（CH1）と MTU3（CH2）はカスケード接続で使用しています。

2.2.5 データフラッシュ

この例では、MAC アドレスの格納にデータフラッシュを使用します。

2.2.6 USB チャネル

この例では、USB 2.0 FS ホスト/ファンクションを使用します。これらの周辺モジュールは USB メモリからの送信データの読み出し、USB メモリへの転送した受信データ、及び転送結果の書き込みに必要です。

2.3 ソフトウェア要件

サンプルプログラムは以下の FIT モジュールを使用しています。

- r_bsp
- r_ether_rx
- r_ptp_light_rx

- r_flash_rx[1]
- r_tfat_rx[1]
- r_usb_basic[1]
- r_usb_hmsc[1]

[1]テストプロジェクトのみで使用しています。

2.4　制限事項

サンプルプログラムは以下の制限事項があります。

- PTP での時刻同期は対応していません。
- PTP 簡易ドライバの代替として PTP ドライバ（完全版）[7]を使用することはできません。
- スイッチプロジェクトでは PTP フレームの受信と対応処理はできません [1]。

[1] 転送制御は可能です。

2.5　サポートされているツールチェイン

サンプルプログラムは次のツールチェインでテストと動作確認を行っています。

- Renesas RX Toolchain v2.05.00

2.6　ヘッダファイル

2.6.1　スイッチプロジェクト

すべての関数呼び出しは、プロジェクトとともに提供されているヘッダファイル r_ether_rx_if.h または r_ptp_light_rx_if.h ファイルをインクルードすることで行われます。

2.6.2　テストプロジェクト

すべての関数呼び出しは、プロジェクトとともに提供されているヘッダファイル r_ether_rx_if.h, r_ptp_light_rx_if.h, r_flash_rx_if.h, r_tfat_lib.h, r_usb_basic_if.h, r_usb_hmsc_if.h のうち 1 個のファイルをインクルードすることで行われます。

2.7　整数型

このプロジェクトでは ANSI C99 を使用し、整数型は stdint.h で定義しています。

2.8　コンパイル時の設定

サンプルプログラムのコンフィグレーションオプションについて説明します。

2.8.1　スイッチプロジェクト

このプロジェクトのコンフィグレーションオプションは sample_main.h で設定します。オプション名と設定値を以下の表に記載します。

構成設定	
#define LINK_CH - Default value = 1	最初にリンクするイーサネットコントローラのチャネルを設定します。 • 0 を設定すると、CH0 を選択します。 • 1 を設定すると、CH1 を選択します。
#define NUM_CH - Default value = 2	イーサネットコントローラのチャネル数を設定します。 • 2 を設定してください。

2.8.2　テストプロジェクト

このプロジェクトのコンフィグレーションオプションは sample_main.h と r_data_file.h で設定します。オプション名と設定値を以下の表に記載します。

構成設定	
#define NUM_TEST - Default value = 201	試行回数（フレーム転送回数）を設定します。 • 1～201 を設定してください。
#define LINK_CH - Default value = 1	最初にリンクするイーサネットコントローラのチャネルを設定します。 • 0 を設定すると、CH0 を選択します。 • 1 を設定すると、CH1 を選択します。
#define NUM_CH - Default value = 2	イーサネットコントローラのチャネル数を設定します。 • 2 を設定してください。
#define SRC_IDX - Default value = 0	フレーム送信元のチャネルを設定します。 • 0 または 1 を設定してください。 “DST_IDX”で設定した値と異なる値を設定してください。
#define DST_IDX - Default value = 1	フレーム送信先（宛先）のチャネルを設定します。 • 0 または 1 を設定してください。 “SRC_IDX”で設定した値と異なる値を設定してください。
#define MAC_ADDR_1H/2H - Default value = 0x00007490	port0/port1 の MAC アドレスの上位 16 ビットを設定します。 デフォルト値の下位 16 ビットはルネサスベンダ ID（74-90-50）の上位 16 ビットになります。 デフォルト値の上位 16 ビットは予約フィールドであり、0（00-00）を設定してください。 データフラッシュへの最初のアクセス時、または、強制消去オプション（FORCE_ERASE_PRM）を定義している場合、データフラッシュに書かれた MAC アドレスを消去します。 お客様のシステムに適用する場合、必ずこの値を変更してください。
#define MAC_ADDR_1L/2L Case of SRC_IDX = 0, - Default value = 0x5000792C (port0) - Default value = 0x5000792D (port1) Case of SRC_IDX = 1, - Default value = 0x5000792E (port0) - Default value = 0x5000792F (port1)	port0/port1 の MAC アドレスの下位 32 ビットを設定します。 デフォルト値の上位 8 ビットはルネサスベンダ ID の下位 8 ビット（74-90-50）になります。 デフォルト値の下位 24 ビットはこのサンプルプログラム固有の値です。 データフラッシュへの最初のアクセス時、または、強制消去オプション（FORCE_ERASE_PRM）を定義している場合、データフラッシュに書かれた MAC アドレスを消去します。 お客様のシステムに適用する場合、必ずこの値を変更してください。
#define FORCE_ERASE_PRM - undefined	動作完了後、データフラッシュに書き込まれた MAC アドレスを消去するか、残すかを選択します。 • 定義した場合、データフラッシュから MAC アドレスを消去します。
#define WAIT_MAX_CNT - Default value = 1000000	フレーム受信時のソフトウェアループ処理における回数の上限値を設定します。 この値がフレーム受信時のタイムアウトの指定になります。
#define RETRY_MAX_CNT - Default value = 1000	読み出しリトライ回数[1]を設定します。
#define MAX_DAT_SIZE - Default value = (1514 – 12)	送信データファイルのサイズ[2]の上限値をバイト単位で設定します。 • 1502 バイト以下の値を設定してください。

構成設定	
#define READ_DIR - Default value "TEST"	送信データファイルを格納するディレクトリ名を指定します。
#define READ_FILE - Default value "SRC"	送信データファイルの名前を指定します。 この文字列に、フレーム送信順に割り当てられた番号と拡張子である”.txt”を接続し、ファイル名となります。 例：“SRC_0.txt”, “SRC_1.txt”, ,　“SRC_200.txt”
#define WRITE_DIR - Default value "RCV"	受信データファイルを格納するディレクトリ名を指定します。
#define WRITE_FILE - Default value “DST"	受信データファイルの名前を指定します。 この文字列に、フレーム受信順に割り当てられた番号と拡張子である”.txt”を接続し、ファイル名となります。 例：“DST_0.txt”, “DST_1.txt”, ,　“DST_200.txt”
#define RESULT_FILE - Default value “RESULT"	フレーム転送でのサイズと時間を保存した転送結果ファイルの名前を指定します。 この文字列と拡張子である”.txt”を接続し、ファイル名となります。 例：“RESULT.txt”
#define FILESIZE - Default value = 2048	FAT ファイルシステムのバッファサイズを設定します。 • 2048 を設定してください。

[1] 読み出し時のリトライ処理はフレーム受信割り込みを検出後、EDMAC による受信データの受信バッファへの転送が完了しない場合に必要となります。
[2] 送信フレームのデータフィールドのサイズに相当します。

2.9　データ構造

サンプルプログラムの関数で使用するデータ構造について説明します。

2.9.1　スイッチプロジェクト

このプロジェクトのコンフィグレーションオプションは sample_main.h で設定します。

```
/* TRNMR setting values table */
typedef struct
{
    RelEnabDir rel;
    TranMode trn;
} TrnTbl;
```

2.9.2　テストプロジェクト

このプロジェクトのコンフィグレーションオプションは sample_main.h と r_data_file.h で設定します。

```
/* Ether & USB access state */
typedef enum
{
    APL_START = 0, /* Operation start state */
    APL_READ,      /* USB memory read state */
    APL_COM,       /* Ether communication state */
    APL_WRITE,     /* USB memory write state */
    APL_STOP,      /* Operation stop state */
} APLState;
```

```
/* Data access information structure */
typedef struct
{
    uint16_t size[NUM_TEST]; /* Data size */
    int8_t *src;             /* Address of source data */
    int8_t *dst;             /* Address of destination data */
    uint32_t time[NUM_TEST]; /* Operation time */
} ACCInfo;
```

```
/* Ether communication state information */
typedef enum
{
    COM_ERR = -1, /* General error */
    COM_OK = 0,    /* No error */
    COM_TOUT,      /* Timeout occured */
    COM_OTH_SRC,  /* Received from other source */
    COM_OTH_CH,   /* Received from other channel */
    COM_ERR_FRM,  /* Error frame received */
} COMInfo;
```

2.10　戻り値

サンプルプログラムの関数の戻り値を示します。スイッチプロジェクトには戻り値はありません。テストプロジェクトの戻り値は ether_if.h、flash_if.h、file_if.h にプロトタイプ宣言と共に定義しています。

```
/* Ether access return value */
typedef enum
{
    ETHIF_ERR = -1, /* General error */
    ETHIF_OK = 0,
    ETHIF_TOUT,     /* Timeout occured */
} ethif_t;
```

```
/* Data flash acccess return value */
typedef enum
{
    FLSIF_ERR = -1, /* General error */
    FLSIF_OK = 0,
    FLSIF_ERASE_ERR,  /* Erase error */
    FLSIF_WRITE_ERR,  /* Write error */
    FLSIF_VERIFY_ERR, /* Verify error */
} flshif_t;
```

```
/* File access return value */
typedef enum
{
    FLIF_ERR = -1, /* General error */
    FLIF_OK = 0,
} flif_t;
```

3. サンプルプログラムの仕様

3.1 関数の概要

スイッチプロジェクトとテストプロジェクトの関数を、表 3.1と表 3.2にそれぞれ示します。

表3.1　スイッチプロジェクトの関数

関数	内容
main()	このプロジェクトのメイン処理
wait_seq()	シーケンスの状態遷移待ち処理
EINT_Trig_isr()	フレーム受信割り込みハンドラ
led_init()	ユーザ LED の初期化
led_ctrl()	ユーザ LED 表示の更新

表3.2　テストプロジェクトの関数

関数	内容
main()	このプロジェクトのメイン処理
led_init()	ユーザ LED の初期化
led_ctrl()	ユーザ LED 表示の更新
prm_init()	処理パラメータの初期化
usb_memory_start()	USB メモリタスクの開始
Sample_Task()	サンプルアプリケーションのタスク処理
EtherStart()	イーサネットの開始処理
EtherCom()	イーサネットの通信処理
EINT_Trig_isr()	フレーム受信割り込みハンドラ
EtherErr()	イーサネットのエラーから復帰
FlashInit()	フラッシュドライバ（フラッシュ API）の初期化
load_prm()	データフラッシュからの通信パラメータ（MAC アドレス）のロード
upd_param()	データフラッシュの通信パラメータ（MAC アドレス）の更新
ers_param()	データフラッシュの通信パラメータ（MAC アドレス）の消去
file_crt_dir()	受信データディレクトリの生成
file_read()	ファイル読み出し処理
file_write()	ファイル書き込み処理
file_stop()	ファイル操作の停止と転送結果の書き込み
file_err()	ファイル操作でのエラーから復帰
mtu3_dev_start()	MTU3 の開始（モジュールストップ状態を解除）
mtu3_port_init()	MTU3 関連ポートの初期設定
mtu3_dev_stop()	MTU3 の停止（モジュールストップ状態へ遷移）
Init_timer()	通信性能測定に関連する MTU3 レジスタの初期化
start_eval()	タイマ（通信性能測定）の開始
stop_eval()	タイマ（通信性能測定）の停止
end_timer()	通信性能測定に関連する MTU3 レジスタの終了処理
get_eval_cycle()	通信性能測定でのサイクル数の取得

3.2　環境とプログラムの実行

サンプルプログラムは、2台のRX64M/71M RSKボード[1]、イーサネットケーブル、USBメモリを使用します。

実行手順の概要を以下に説明します。また、図3.1に動作環境の一例を示します。

- RX64M /71M RSK ボード（以後、RSK ボード 1）の 1 台にスイッチプロジェクトの実行コードを書き込んでください。もう 1 台の RX64M /71M RSK ボード（以後、RSK ボード 2）にテストプロジェクトの実行コードを書き込んでください。
- RSK ボード 1 と RSK ボード 2 をイーサネットケーブルで接続してください。イーサネットケーブルは互いのボードの同じチャネルに接続してください（CH0 と CH0、または CH1 と CH1）。フレームがイーサネットケーブルを介してチャネル間で転送されます。RSK ボード 2 に内蔵する MTU3 は、ボード間のフレーム転送時間を測定します。RSK ボード 2 のデータフラッシュは CH0 から CH1（CH1 から CH0）への転送時に使用する MAC アドレスをブロック 0（ブロック 1）に格納します。
- USB メモリを RSK ボード 2 の USB ポートに接続してください。USB メモリはテストフレームとして転送された送信データと受信データ、及びフレーム転送の性能測定結果を格納します。
- テストデータのサンプルをソースプロジェクトの demo_src/usb_memory_sample フォルダに準備していますので、それを USB メモリのルートにコピーすれば使用頂けます。
- RSK ボード 1 と RSK ボード 2 に電源を投入してください。
- RSK ボード 1 はイーサネットドライバの初期化と開始処理の後、ユーザ LED が全て点灯します（LED0: オン、LED1: オン、LED2: オン、LED3: オン）。MAC アドレス[2]は CH0 に”74-90-5-00-79-34”、CH1 に”74-90-5-00-79-35”があらかじめソースファイルで設定しています。
- RSK ボード 2 はイーサネットドライバ、MTU3 ドライバ、フラッシュドライバ、USB ホストドライバの初期化と開始処理の後、ユーザ LED が”0x1”パターンに点灯します（LED0: オン、LED1: オフ、LED2: オフ、LED3: オフ）。初期化処理において、あらかじめデータフラッシュに適切な値[3]の MAC アドレスが設定されてない場合、MAC アドレスの値を更新します。
- RSK ボード 1 のユーザスイッチである”SW1”を押すと、RSK ボード 1 は3.3節で説明するスイッチプロジェクトの動作シーケンスを開始します。
- ユーザ LED はユーザスイッチの”SW1”が押された回数を示します。その回数は簡易スイッチ機能の設定パターンを示します。簡易スイッチ機能の設定パターンについては3.4節を参照ください。
- 処理中にエラーが発生した場合、ユーザ LED が奇数パターンに点灯します（LED0: オフ、LED1: オン、LED2: オフ、LED3: オン）。
- RSK ボード 2 のユーザスイッチである”SW1”を押すと、RSK ボード 2 は3.3節で説明するスイッチプロジェクトの動作シーケンスを開始します。
- 動作シーケンスが正常に完了した場合、ユーザ LED が全て点灯します（LED0: オン、LED1: オン、LED2: オン、LED3: オン）。
- イーサネット通信でエラーが発生した場合、ユーザ LED が”0x6”パターンに点灯します（LED0: オフ、LED1: オン、LED2: オン、LED3: オフ）。
- USB 通信またはファイル操作でエラーが発生した場合、ユーザ LED が奇数パターンに点灯します（LED0: オフ、LED1: オン、LED2: オフ、LED3: オン）。
- 強制消去オプション（FORCE_ERASE_PARAM）が定義されており、MAC アドレスの消去操作でエラーが発生した場合、ユーザ LED が”0xE”パターンに点灯します（LED0: オフ、LED1: オン、LED2: オン、LED3: オン）。

[1] 製品名は Renesas Starter Kit+ for RX64M [8] または Renesas Starter Kit+ for RX71M [9]です。

[2] お客様のシステムに適用する場合、必ずこの値を変更してください。

[3] ルネサスのベンダ ID（74-90-50）で判定します。

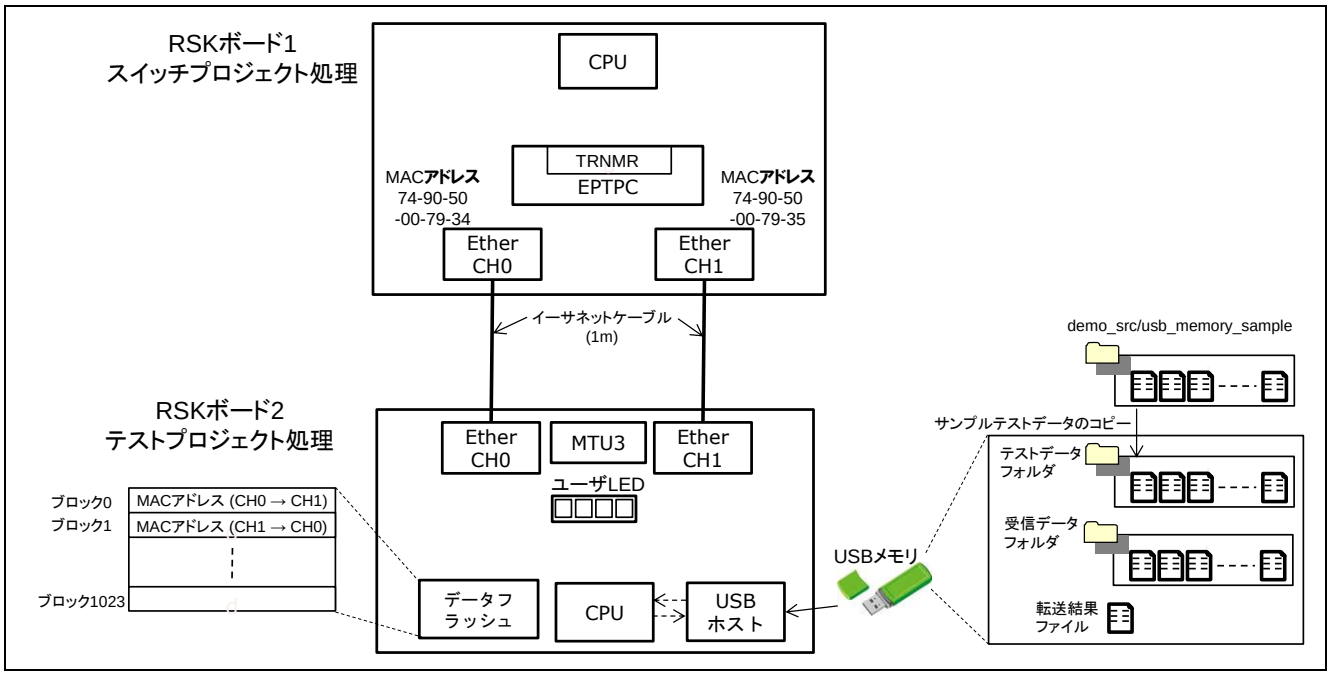

図3.1　動作環境

3.3　動作シーケンス

この例の動作シーケンスをフレームの転送方向を CH0 から CH1、試行回数[1] を 201 回として説明します。

USB メモリに格納されるデータの内容を図 3.2に示します。USB メモリには TEST フォルダ、RCV フォルダ、RESULT.txt ファイルがあります。TEST フォルダは SRC_0.txt から SRC_200.txt で構成するテストデータを格納します。RCV フォルダは DST_0.txt から DST_200.txt で構成する受信データを格納します。テストデータと受信データは、それぞれ送信フレームと受信フレームの一部を構成します。フレーム転送が正常に終了した場合、テストデータと受信データの内容は互いに一致します。図 3.2に SRC_2.txt、DST_2.txt、RESULT.txt の一例を示しています。RESULT.txt には試行回数、各フレーム転送での処理時間、フレームサイズを転送性能結果として格納します。

[1] ”NUM_TEST”で指定する。

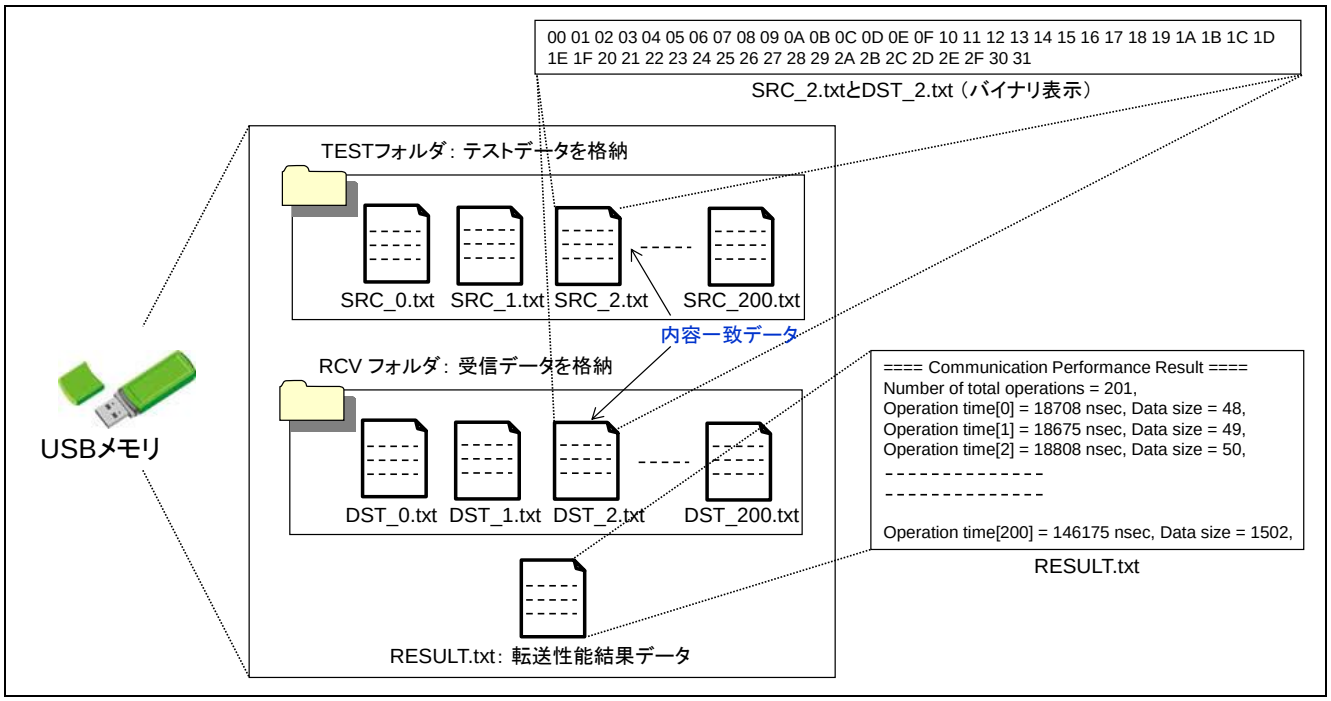

図3.2　USB メモリの内容

転送するフレームフォーマットを MAC アドレスと関連付けて図 3.3で説明します。ここで、テストデータは SRC_2.txt を使用し、フレームの転送方向を CH0 から CH1 とします。RSK ボード 2 のポート 0 の MAC アドレスは、データフラッシュのブロック 0 の最初の 6 バイト（アドレス：0x0010_0000 – 0x0010_0005）、RSK ボード 2 のポート 1 の MAC アドレスは、データフラッシュのブロック 0 の次の 6 バイト（アドレス：0x0010_0006 – 0x0010_000B）にそれぞれ書き込まれています。

宛先 MAC アドレスはポート 1 の MAC アドレスをデータフラッシュからロードし設定します。送信元 MAC アドレスはポート 0 の MAC アドレスをデータフラッシュからロードし設定します。送信元 MAC アドレスに続くタイプフィールドに相当するフィールドから、SRC_2.txt の送信データを接続し、フレームの全フレーム長は 62 バイト（12 バイト+50 バイト）になります。

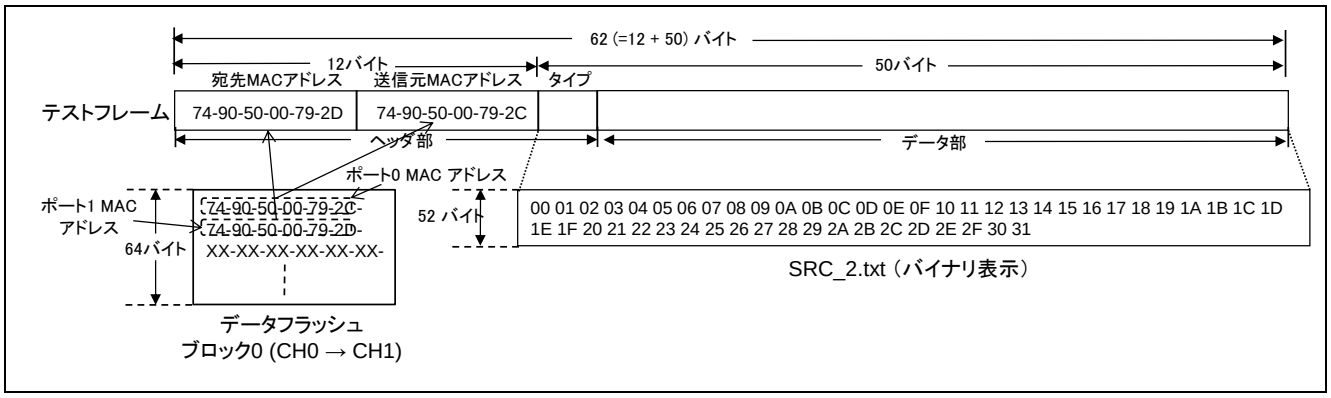

図3.3　フレームフォーマット

動作シーケンスの概要を図 3.4に示します。

- RSK ボード 1

(1) 簡易スイッチ機能を設定します。

- RSK ボード 2

(2) フレームとして転送するテストデータを USB ホスト経由で USB メモリから読み出します。

(3) データフラッシュから MAC アドレスをロードします。

(4) 図 3.3に示したフレームを生成します。

(5) ポート 0 からフレームを送信します。

(6) 中継されたフレーム [1] を受信します。

(7) MTU3 によりフレームの転送時間を測定します。

(8) フレームとして転送された受信データを USB メモリに書き込みます。

(9) USB メモリからの読み出し、転送時間測定を含むフレーム転送、USB メモリへの書き込みを設定された試行回数だけ繰り返します。その後、USB メモリへフレーム転送時間の測定結果を書き込みます。

(10) “FORCE_ERASE_PRM”オプションが定義されている場合、データフラッシュの MAC アドレスを消去します。

[1] RSK ボード 2 の簡易スイッチにより中継されたフレームのみ。

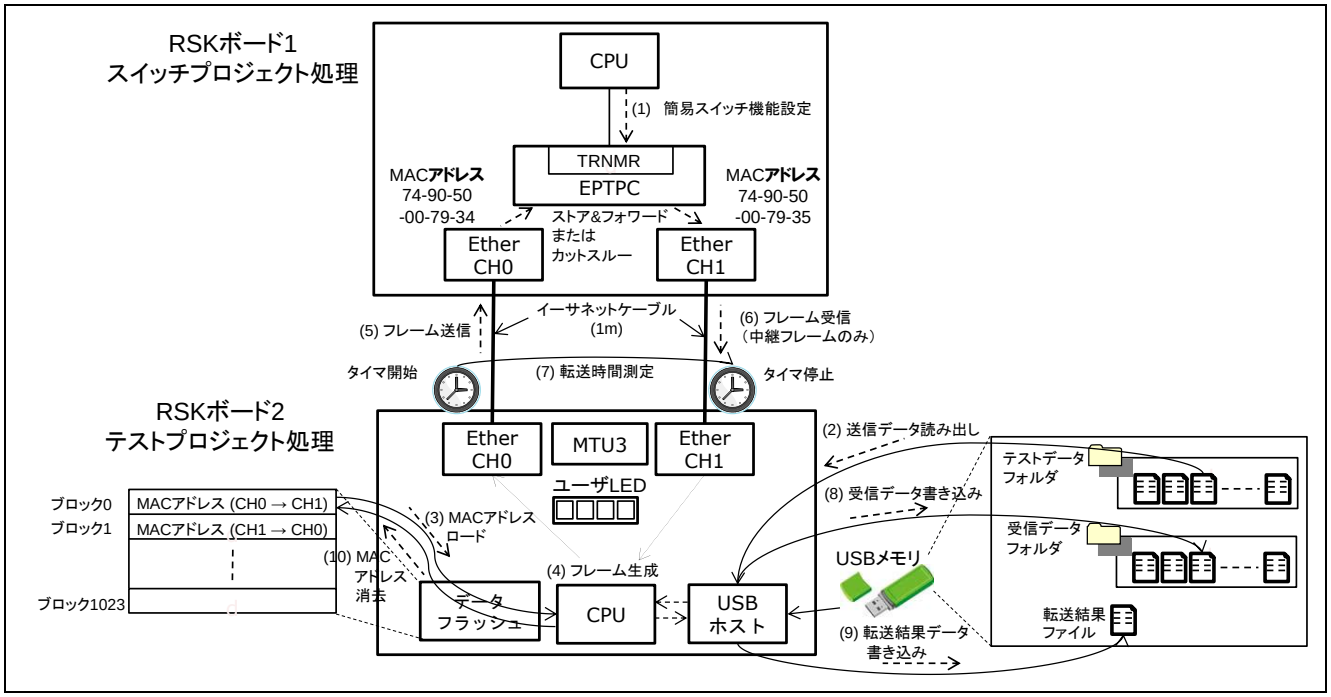

図3.4　シーケンス概要

3.4　簡易スイッチ設定

図3.5に簡易スイッチ機能に関連するモジュール構成とレジスタ仕様を示します。簡易スイッチ機能はイーサネット周辺モジュールのパケット中継部（PRC-TC）に実装しており、ストア&フォワード方式またはカットスルー方式を選択し使用できます。特に産業用ネットワークで一般的なディジーチェーン接続にカットスルー方式を適用した場合、ポート間の転送における遅延を短縮でき、有効な機能となります。また、ネットワークをチャネル毎に分離することで、独立した2個のネットワークとして使用することもできます。

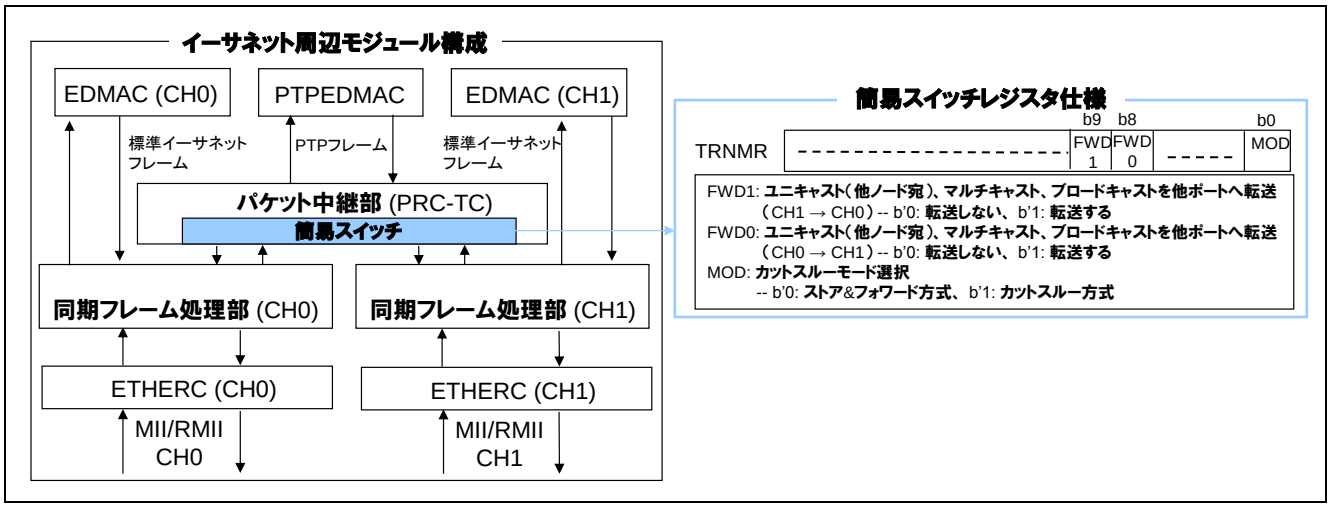

図3.5　モジュール構成とレジスタ仕様

簡易スイッチのレジスタは、RSK ボード 1 のユーザスイッチである”SW1”を押す回数に従い設定されます。簡易スイッチ設定とフレーム転送結果を表 3.3に示します。2 列目は LED の点灯パターンで、イーサネットの通信設定が完了した後、ユーザスイッチである”SW1”が押された回数に従います。3 列目は簡易スイッチの動作を決定する”TRNMR”の値を示します。8 列目は中継されるフレーム数の期待値で、マクロの”NUM_TEST”で 201 を指定した場合を示しています。一般に、フレーム転送性能はカットスルー方式を適用したほうがストア&フォワード方式を適用した場合に比較し高くなります。

表3.3　簡易スイッチ設定とフレーム転送結果

No.	LED パターン	TRNMR	FWD1 (b9)	FWD0 (b8)	MOD (b0)	テストフレーム転送方向	フレーム数	備考
1	オフ-オフ-オフ-オフ	00-00-03-01	1:転送する CH1→CH0	1:転送する CH0→CH1	1: カットスルー	CH0→CH1	201	No.3 より速い
2						CH1→CH0	201	No.4 より速い
3	オフ-オフ-オフ-オン	00-00-03-00			0: ストア& フォワード	CH0→CH1	201	
4						CH1→CH0	201	
5	オフ-オフ-オン-オフ	00-00-02-01		0:転送しない CH0→CH1	1: カットスルー	CH0→CH1	0	中継しない
6						CH1→CH0	201	No.8 より速い
7	オフ-オフ-オン-オン	00-00-02-00			0: ストア& フォワード	CH0→CH1	0	中継しない
8						CH1→CH0	201	
9	オフ-オン-オフ-オフ	00-00-01-01	0:転送しない CH1→CH0	1: 転送する CH0→CH1	1: カットスルー	CH0→CH1	201	No.11 より速い
10						CH1→CH0	0	中継しない
11	オフ-オン-オフ-オン	00-00-01-00			0: ストア& フォワード	CH0→CH1	201	
12						CH1→CH0	0	中継しない
13	オフ-オン-オン-オフ	00-00-00-01		0:転送しない CH0→CH1	1: カットスルー	CH0→CH1	0	中継しない
14						CH1→CH0	0	中継しない
15	オフ-オン-オン-オン	00-00-00-00			0: ストア& フォワード	CH0→CH1	0	中継しない
16						CH1→CH0	0	中継しない

3.5　ソフトウェア動作フロー

この節では、サンプルプログラムのソフトウェア動作フローを説明します。

3.5.1 スイッチプロジェクト

図 3.6にスイッチプロジェクトの動作フローを示します。スイッチプロジェクトはイーサネット周辺モジュールを設定しイーサネット通信を可能にします。その後、簡易スイッチ機能を設定し、フレーム受信割り込みで受信フレーム数をカウントします。

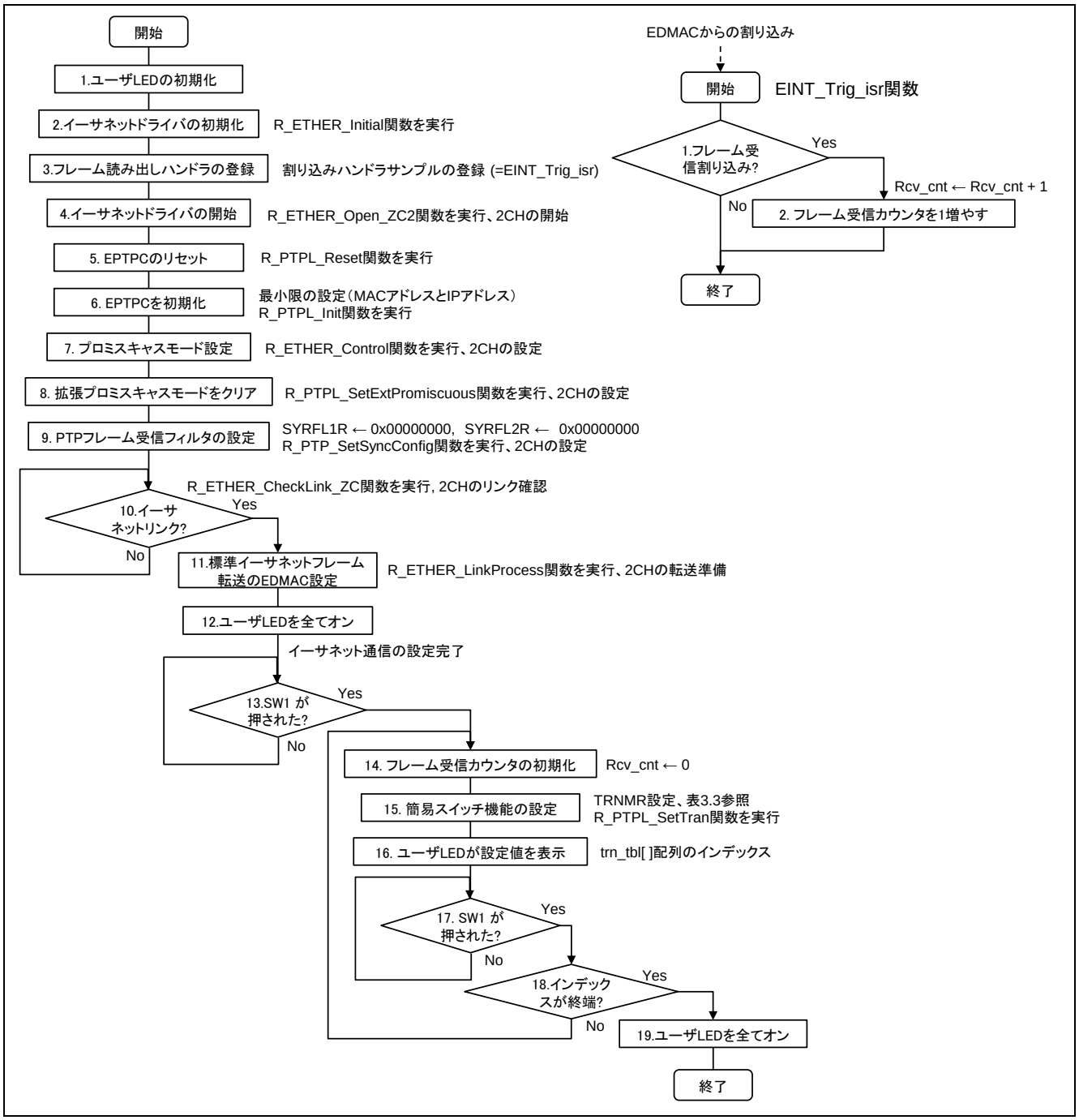

図3.6　スイッチプロジェクト動作フロー

3.5.2 テストプロジェクト

図 3.7は初期化から無限ループまでの処理を示します。サンプルアプリケーションのタスク（Sample_Task 関数）は無限ループ内の図 3.8に示す USB アプリケーションタスクを経由し呼ばれます。図 3.9はアプリケーションの状態により管理される個々のタスク構造を示します。

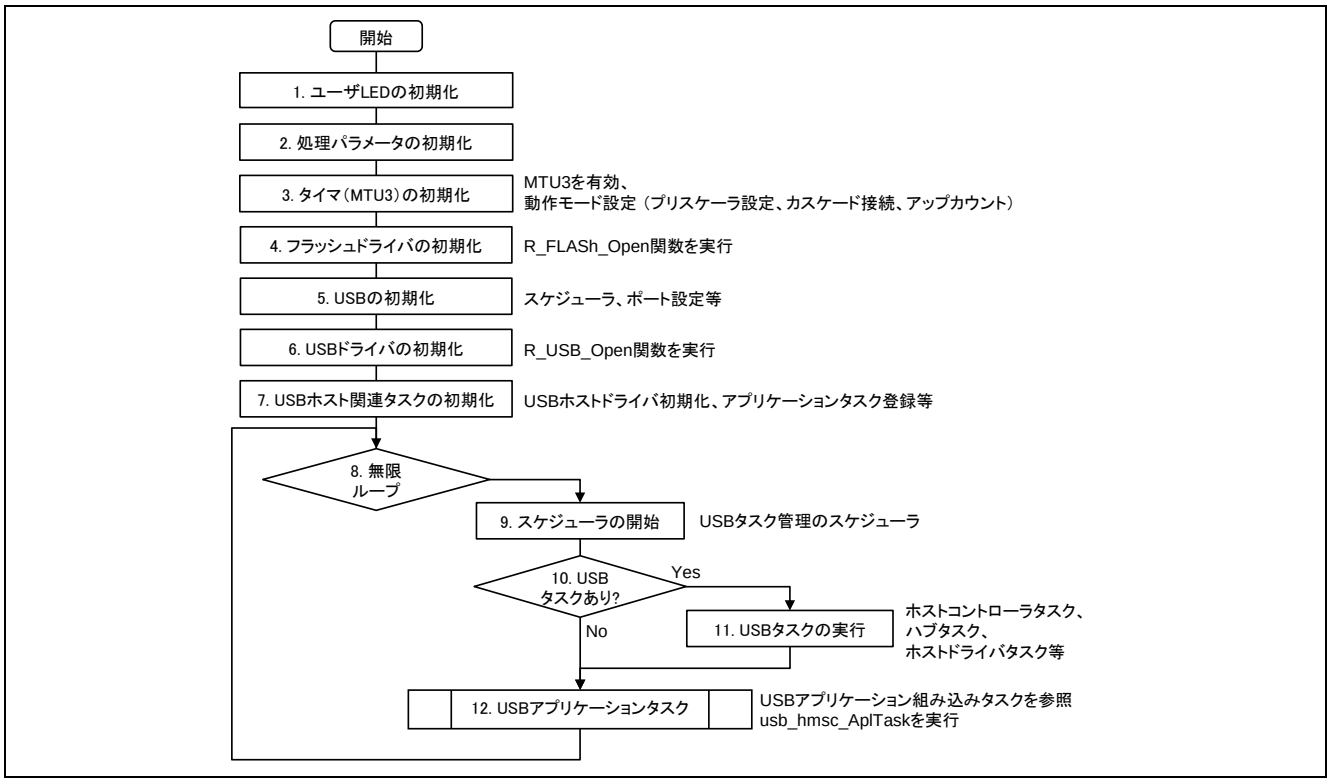

図3.7　初期化処理

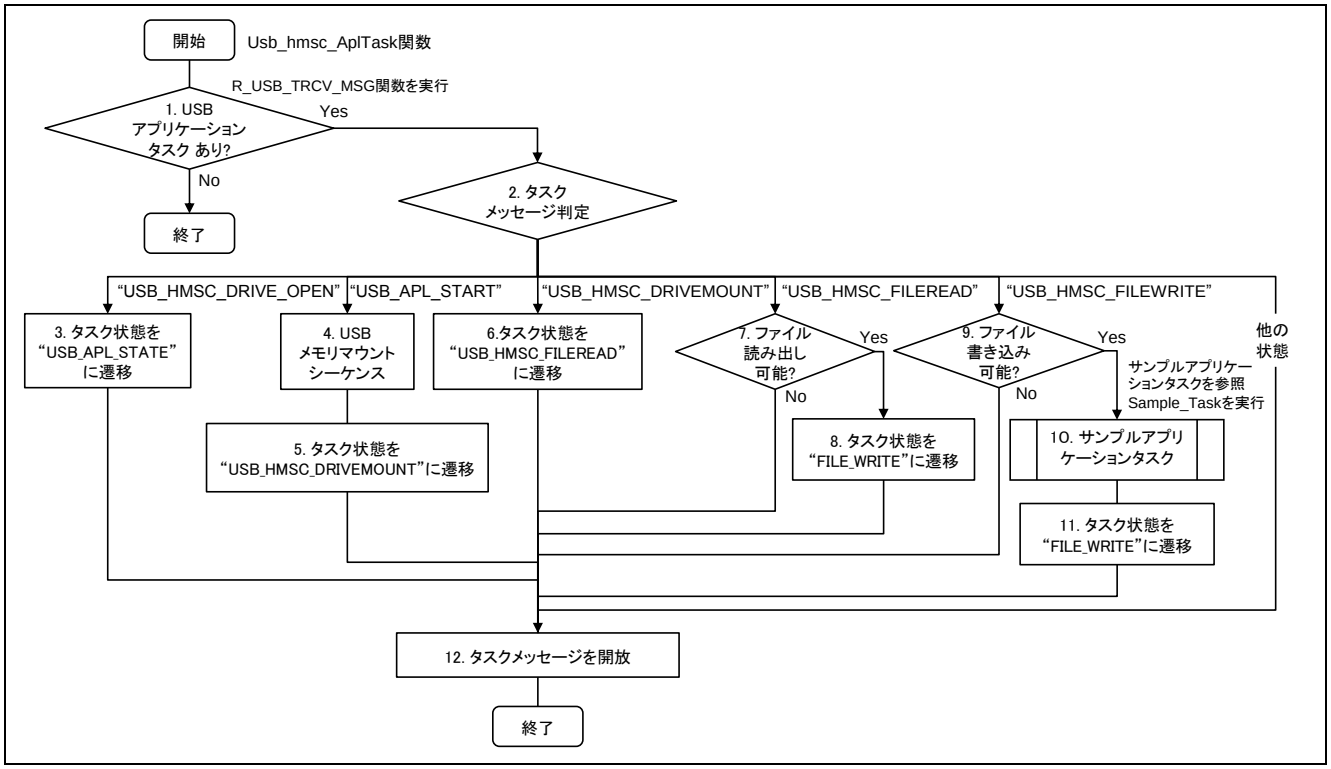

図3.8　USB アプリケーション組み込みタスク

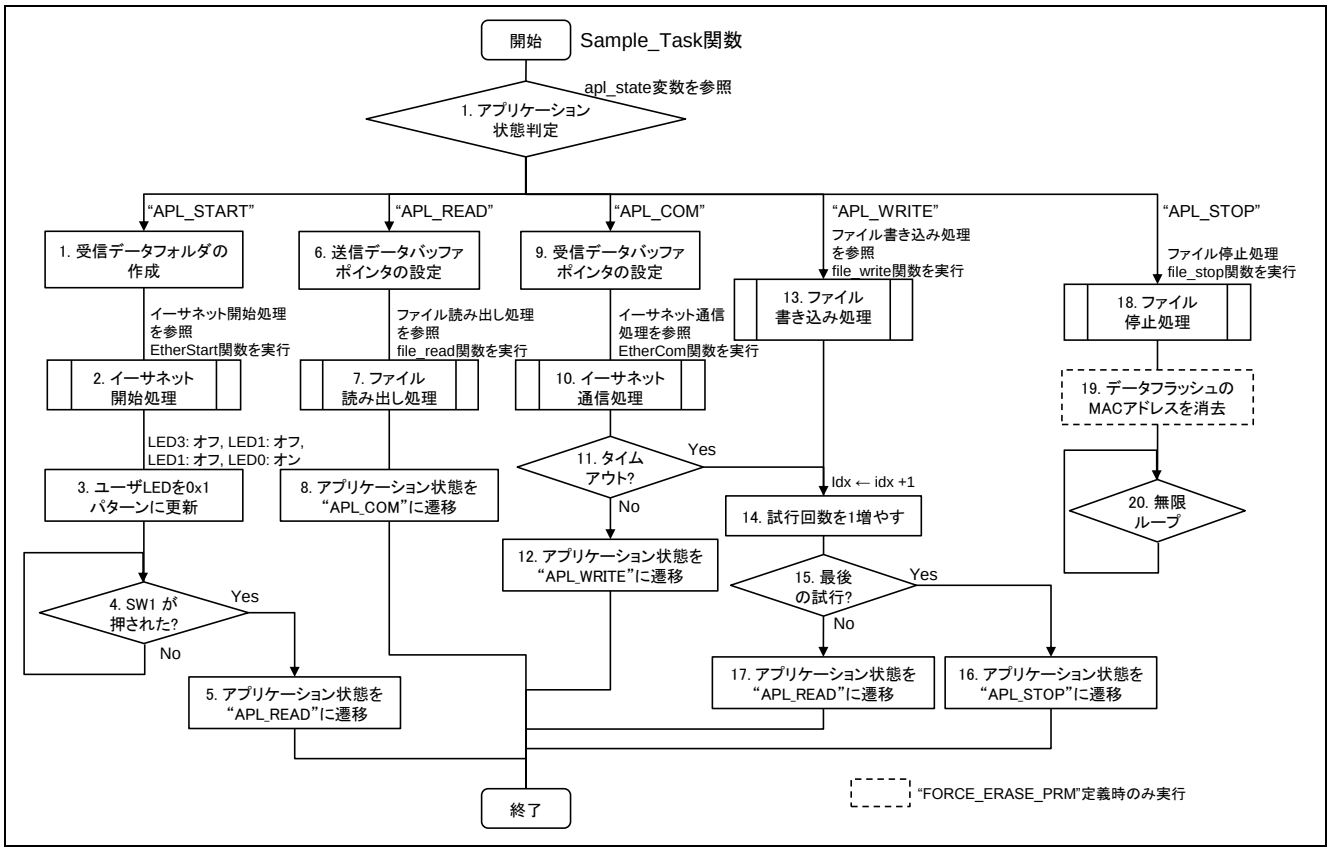

図3.9　サンプルアプリケーションタスク

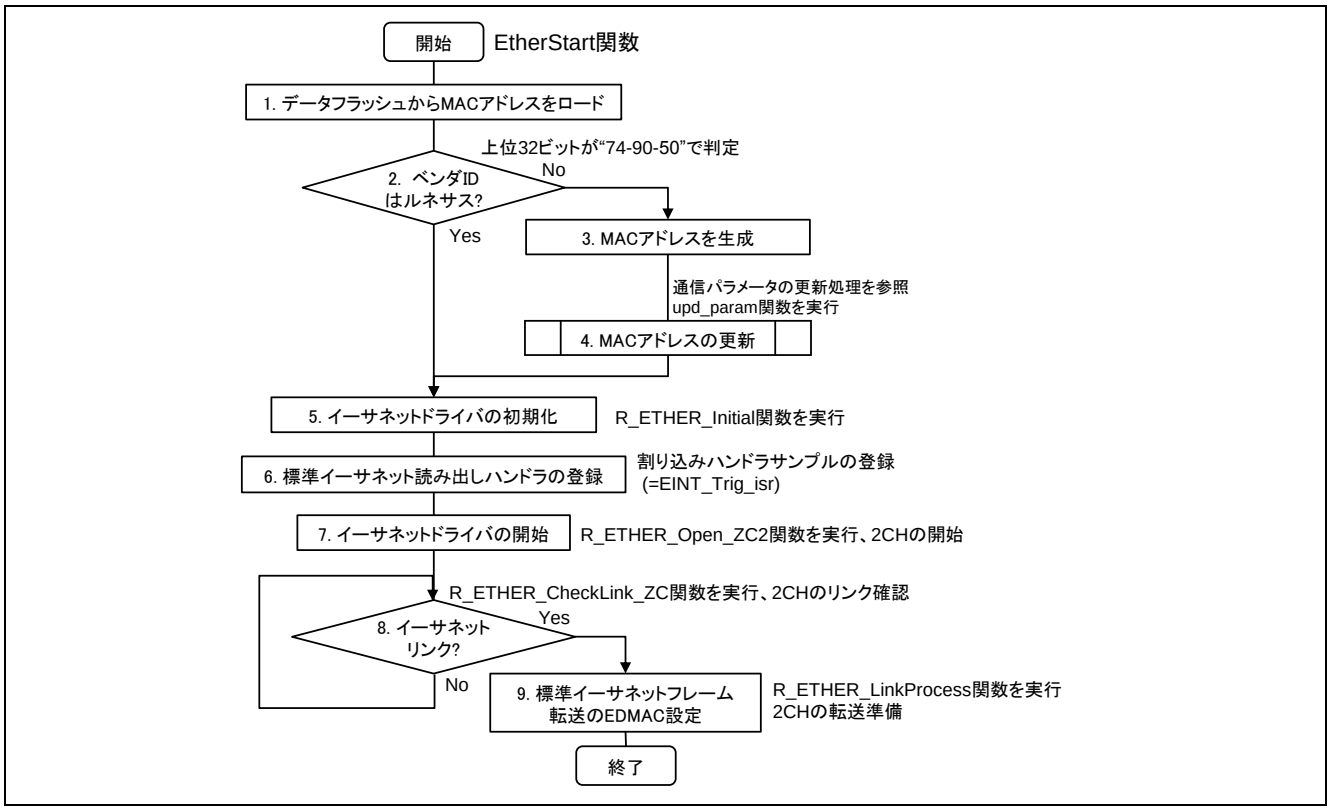

図3.10　イーサネット開始処理

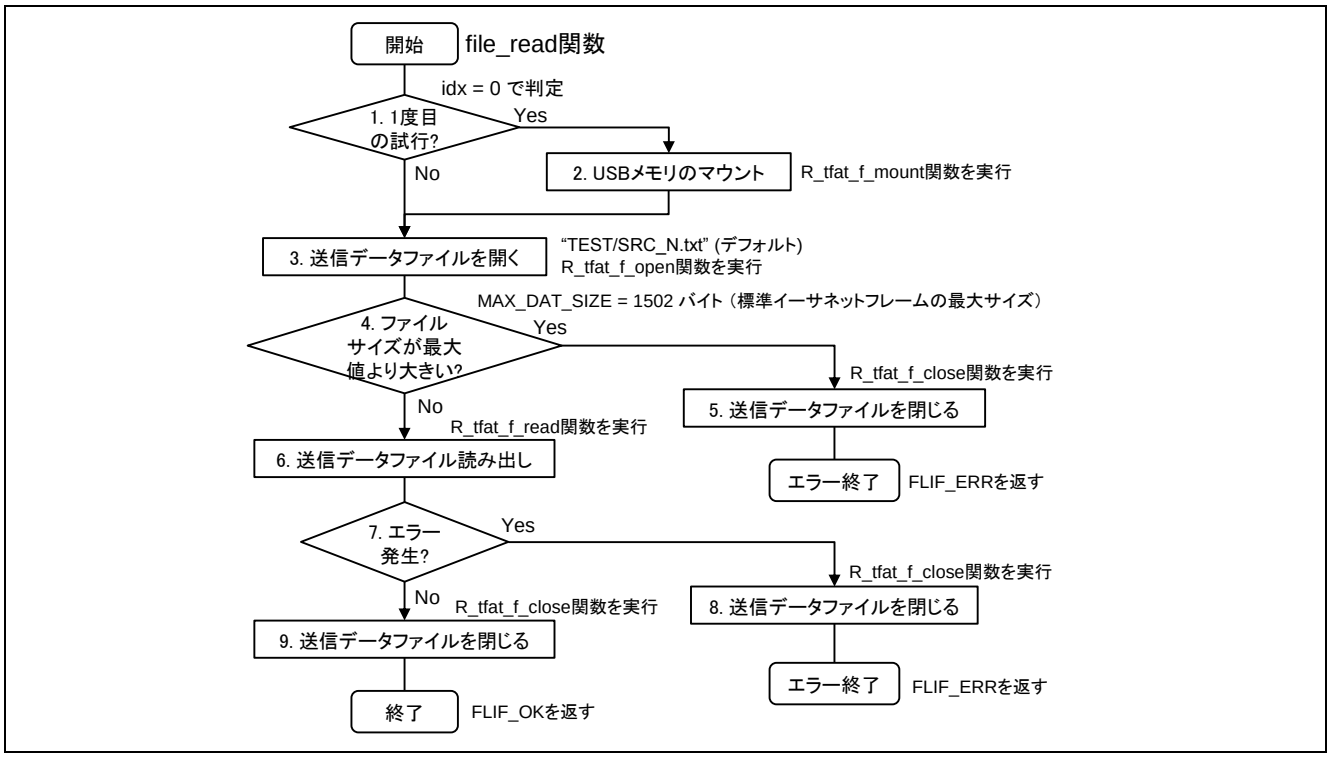

図3.11　ファイル読み出し処理

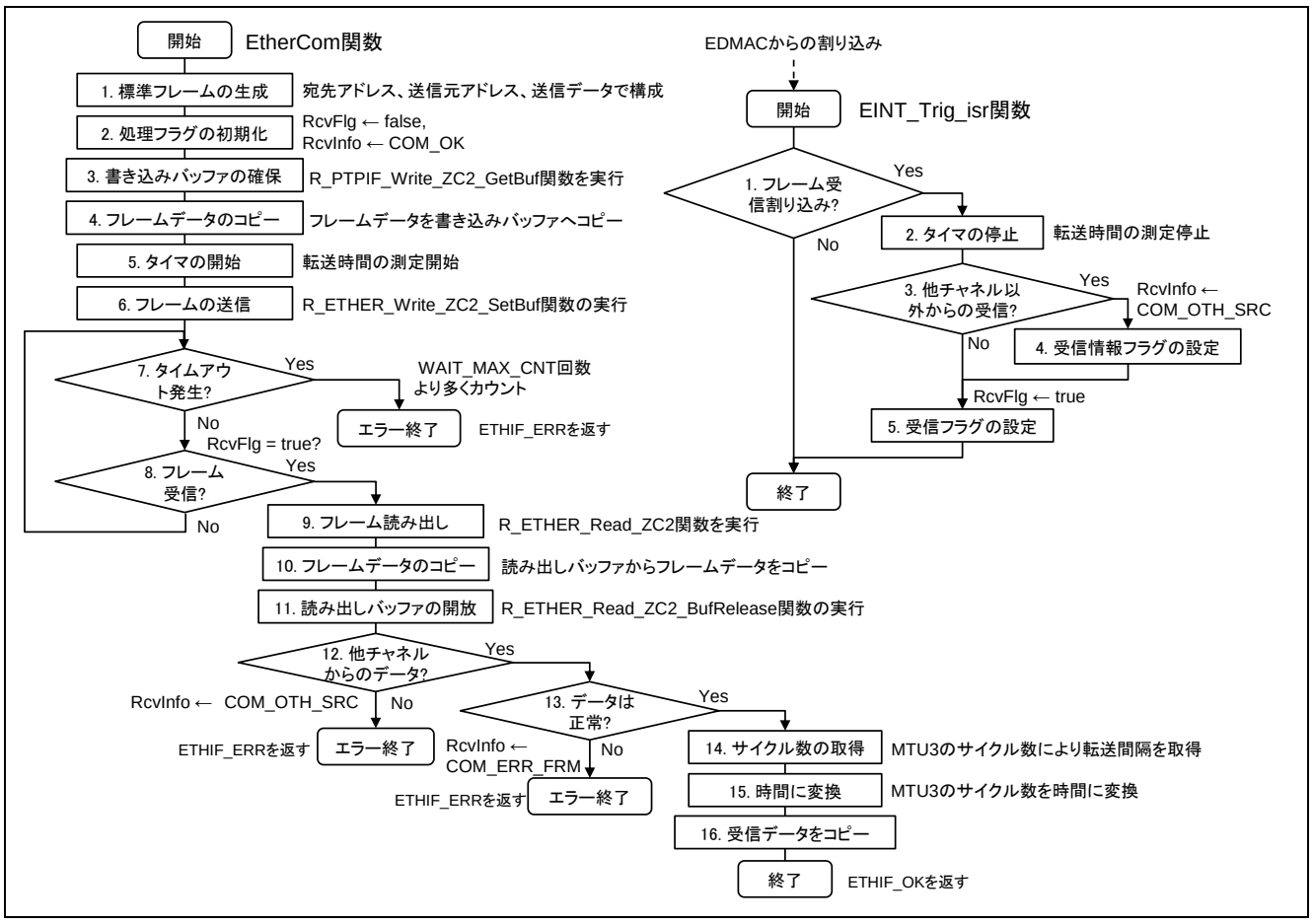

図3.12　イーサネット通信処理

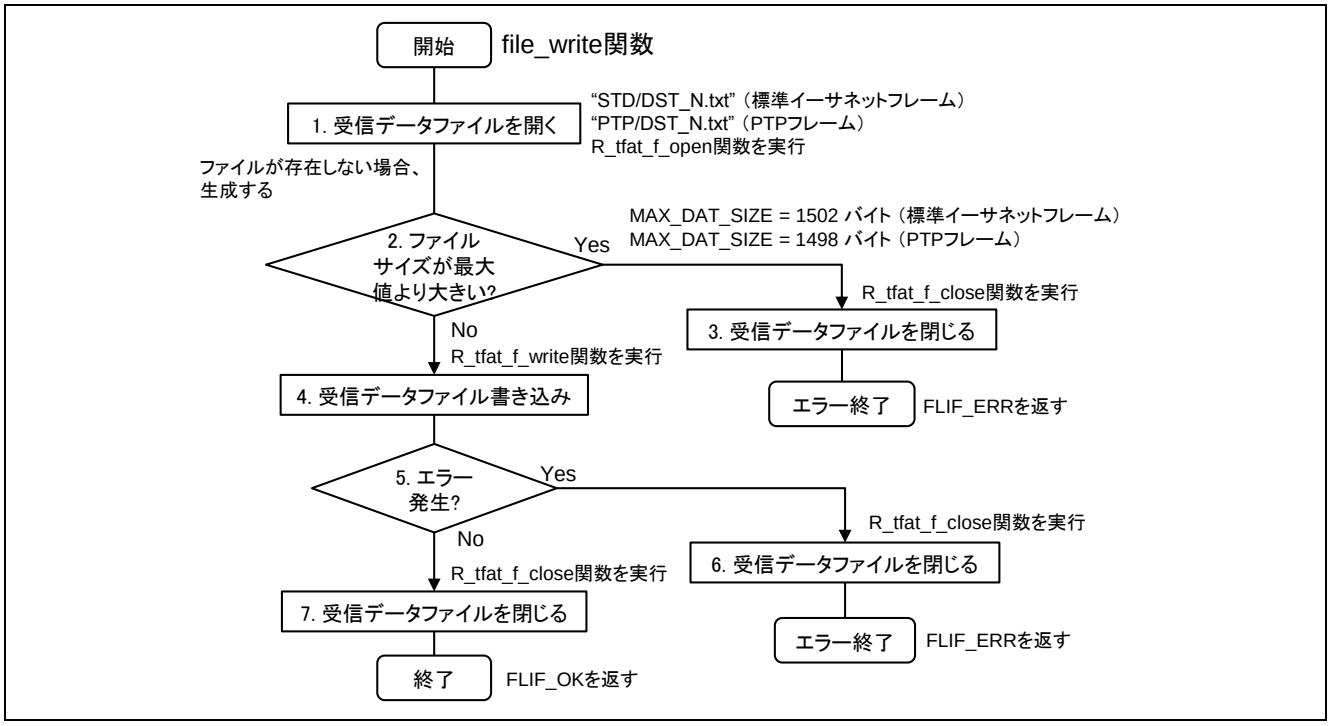

図3.13　ファイル書き込み処理

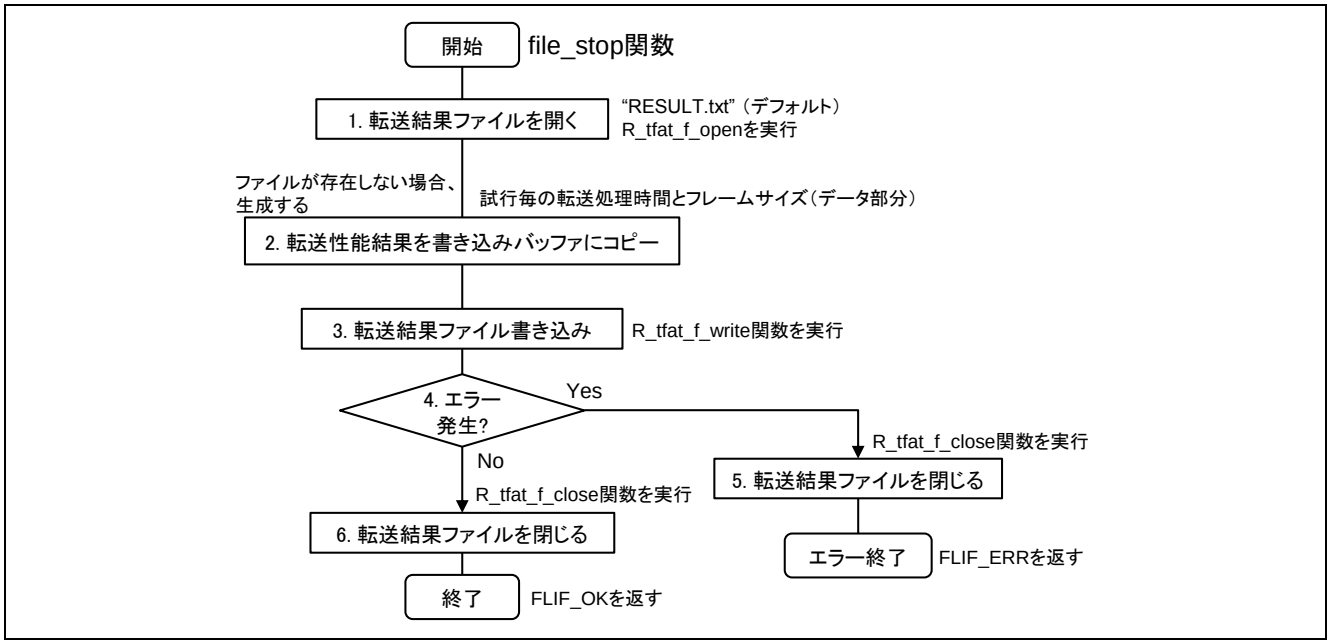

図3.14　ファイル停止処理

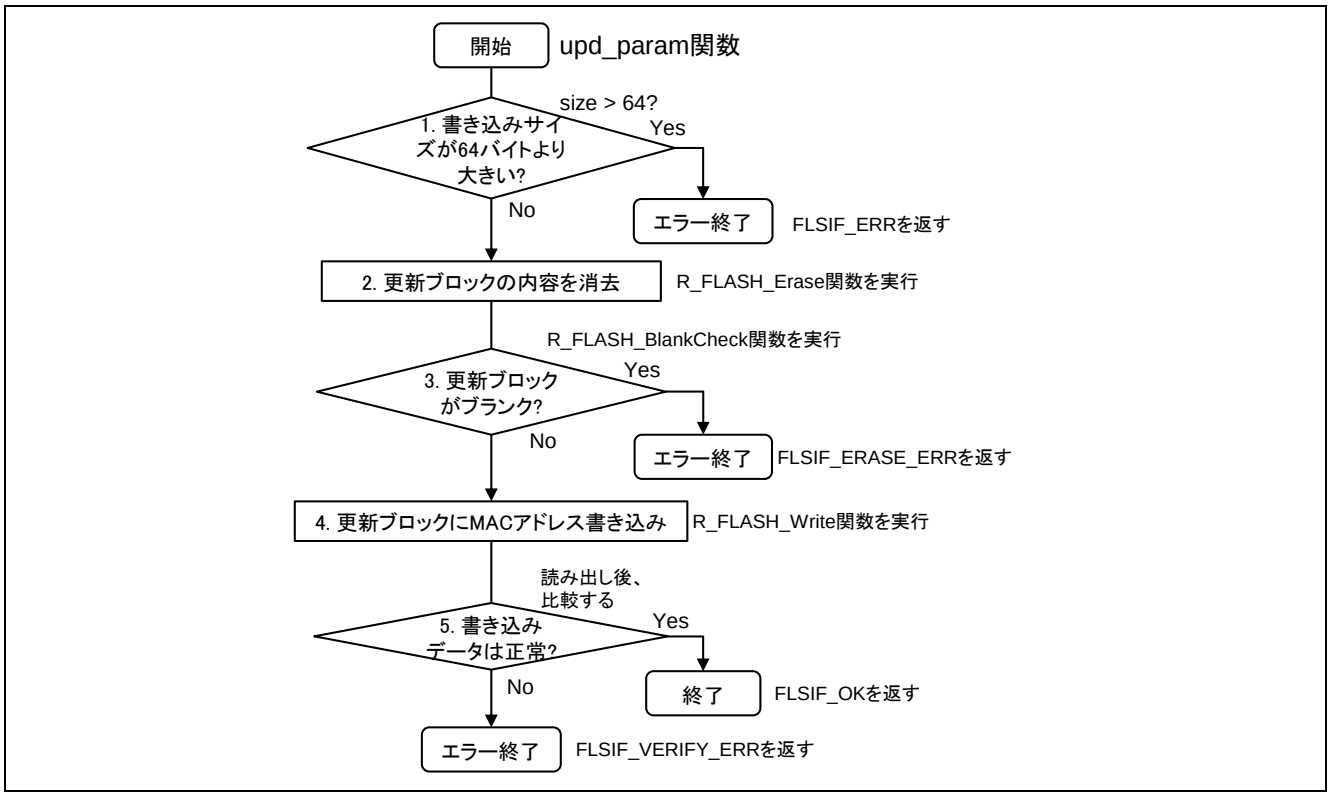

図3.15　通信パラメータ更新

3.6　ボードの設定

サンプルプログラムの動作には RX64M/71M RSK ボードのジャンパをデフォルト設定から変更する必要があります。イーサネット PHY にアクセスするチャネルはソフトウェア設定に合わせ、無矛盾に設定してください。USB[1] にアクセスするチャネルはボードのデフォルト設定から変更する必要があります。RX64M/71M RSK ボードの型名が R0K50564MC001BR または R0K5RX71MC010BR の場合、ジャンパ設定を図 3.16に示します。また、RX71M RSK ボード型名が R0K50571MC000BR の場合、ジャンパ設定を図 3.17に示します。

[1]USB の設定はテストプロジェクトにより動作させる RSK ボード 2 のみ必要です。

- イーサネットPHY アクセスの設定

ジャンパ	LINK_CH = 1（デフォルト設定）	LINK_CH = 0	使用機能
J3	2-3	1-2	ETHERC ET0MDIO または ET1MDIO
J4	2-3	1-2	ETHERC ET0MDC または ET1MDC

- USB アクセスの設定

ジャンパ	ボードのデフォルト設定	サンプル動作の設定	使用機能
J2	2-3	1-2	USBでホストモードを有効化
J6	1-2	2-3	USB USB0VBUSEN

図3.16　ジャンパ設定

- イーサネットPHY アクセスの設定

ジャンパ	LINK_CH = 1（デフォルト設定）	LINK_CH = 0	使用機能
J13	2-3	1-2	ETHERC ET0MDIO または ET1MDIO
J9	2-3	1-2	ETHERC ET0MDC または ET1MDC

- USB アクセスの設定

ジャンパ	ボードのデフォルト設定	サンプル動作の設定	使用機能
J1	2-3	1-2	USBでホストモードを有効化
J3	1-2	2-3	USB USB0VBUSEN

図3.17　ジャンパ設定

4. 参考資料

ユーザーズマニュアル: ハードウェア

RX64M グループユーザーズマニュアル ハードウェア編 Rev.1.00（R01UH0377JJ）
RX71M グループユーザーズマニュアル ハードウェア編 Rev.1.00（R01UH0493JJ）
最新版はルネサス エレクトロニクスのウェブサイトからダウンロードできます。

ユーザーズマニュアル: ソフトウェア

RX ファミリ RXv2 命令セットアーキテクチャ ユーザーズマニュアル ソフトウェア編（R01US0071JJ）
最新版はルネサス エレクトロニクスのウェブサイトからダウンロードできます。

技術情報/ニュース

最新版はルネサス エレクトロニクスのウェブサイトからダウンロードできます。

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://japan.renesas.com/

お問合せ先
http://japan.renesas.com/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2015.10.30	—	初版発行
1.10	2016.03.31	—	PTP 簡易ドライバ Rev.1.10、イーサネットドライバ Rev.1.10 を適用
1.11	2016.11.11	—	PTP 簡易ドライバ Rev.1.11、イーサネットドライバ Rev.1.12 を適用

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/