SH7216グループ
イーサネット受信設定例

R01AN0052JJ0200
Rev.2.00
2010.07.30

要旨

本アプリケーションノートは、SH7216 のイーサネット受信設定例について説明しています。

動作確認デバイス

SH7216

目次

1. はじめに

1.1 仕様

- 本応用例ではイーサネットフレームを連続で 10 フレーム受信します。

1.2 使用機能

- ピンファンクションコントローラ（PFC）
- イーサネットコントローラ（EtherC）
- イーサネットコントローラ用ダイレクトメモリアクセスコントローラ（E-DMAC）

1.3 適用条件

マイコン	SH7216
動作周波数	内部クロック： 200 MHz バスクロック： 50 MHz 周辺クロック： 50 MHz
統合開発環境	ルネサス エレクトロニクス製 High-performance Embedded Workshop Ver.4.07.00
Cコンパイラ	ルネサス エレクトロニクス製SuperH RISC engineファミリ C/C++コンパイラパッケージ Ver.9.03 Release 00
コンパイルオプション	High-performance Embedded Workshopでのデフォルト設定 （-cpu=sh2afpu -fpu=single -debug -gbr=auto -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1）

1.4 関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。合わせて参照してください。

- SH7216 グループ　初期設定例
- SH7216 グループ　イーサネット PHY-LSI 自動交渉設定例
- SH7216 グループ　イーサネット送信設定例

2.　応用例の説明

本応用例では、イーサネットコントローラ（EtherC）、およびイーサネットコントローラ用ダイレクトメモリアクセスコントローラ（E-DMAC）を使用します。

2.1　　使用機能の動作概要

本 LSI では、イーサネット通信を行う場合必ず EtherC と E-DMAC を使用します。EtherC は受信制御を行います。E-DMAC はその送信／受信 FIFO とユーザが指定するデータ格納先（バッファ）間の DMA 転送を行います。

2.1.1　　EtherC の概要

本 LSI は、イーサネットあるいは IEEE802.3 の MAC（Media Access Control）層規格に準拠したイーサネットコントローラ（EtherC）を内蔵しています。EtherC は、同規格に準拠した物理層 LSI（PHY-LSI）と接続することにより、イーサネット／IEEE802.3 フレームの送受信を行うことができます。本 LSI 内蔵の EtherC は MAC 層インタフェースを 1 系統内蔵しています。また EtherC は、本 LSI 内部で E-DMAC に接続されており、メモリとの高速アクセスが可能です。

図 1にEtherCの構成を示します。

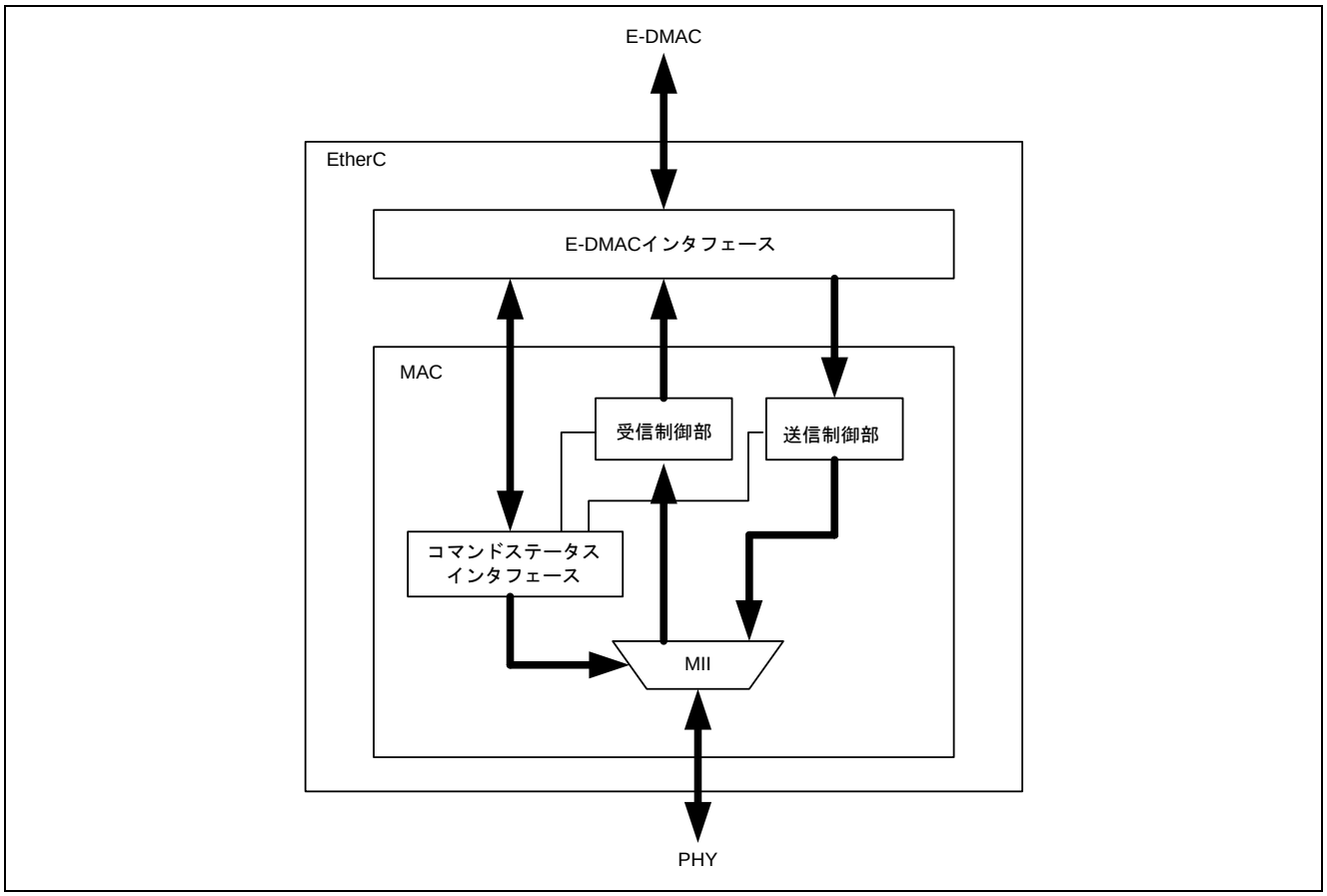


図1　EtherC の構成

2.1.2　EtherC 受信部の概要

EtherC受信部は、MII（Media Independent Interface）から入力されたフレームをプリアンブル、SFD（Start Frame Delimiter）、データおよびCRC（Cyclic Redundancy Check）データに分解します。そしてプリアンブル、SFD, CRCデータを除いた部分をE-DMAC受信部に出力します。図 2にEtherC受信部の状態遷移図を示します。受信動作のフローは以下のようになります。

1. EtherC は EtherC モードレジスタ（ECMR）の受信許可（RE）ビットがセットされると、受信アイドル状態に遷移します。
2. 受信フレームのプリアンブルに続く SFD を検出すると受信処理を開始します。不当パターンの場合はフレームを破棄します。
3. 通常モードでは、(i) 宛先 MAC アドレスが本 LSI 宛の場合、(ii) ブロードキャストフレームの場合、または (iii) マルチキャストフレームの場合にデータ受信を開始します。プロミスキャスモードでは、フレームの種類にかかわらず受信を開始します。
4. MII からのフレームを受信後、フレームデータ部の CRC チェックを行います。結果はメモリ上にフレームデータをライトした後、ディスクリプタ内にステータスとして反映されます。異常時は、エラーステータスを EtherC/E-DMAC ステータスレジスタ（EESR）に設定します。
5. 1 フレームを受信後、アイドル状態に遷移し次のフレーム受信に備えます。

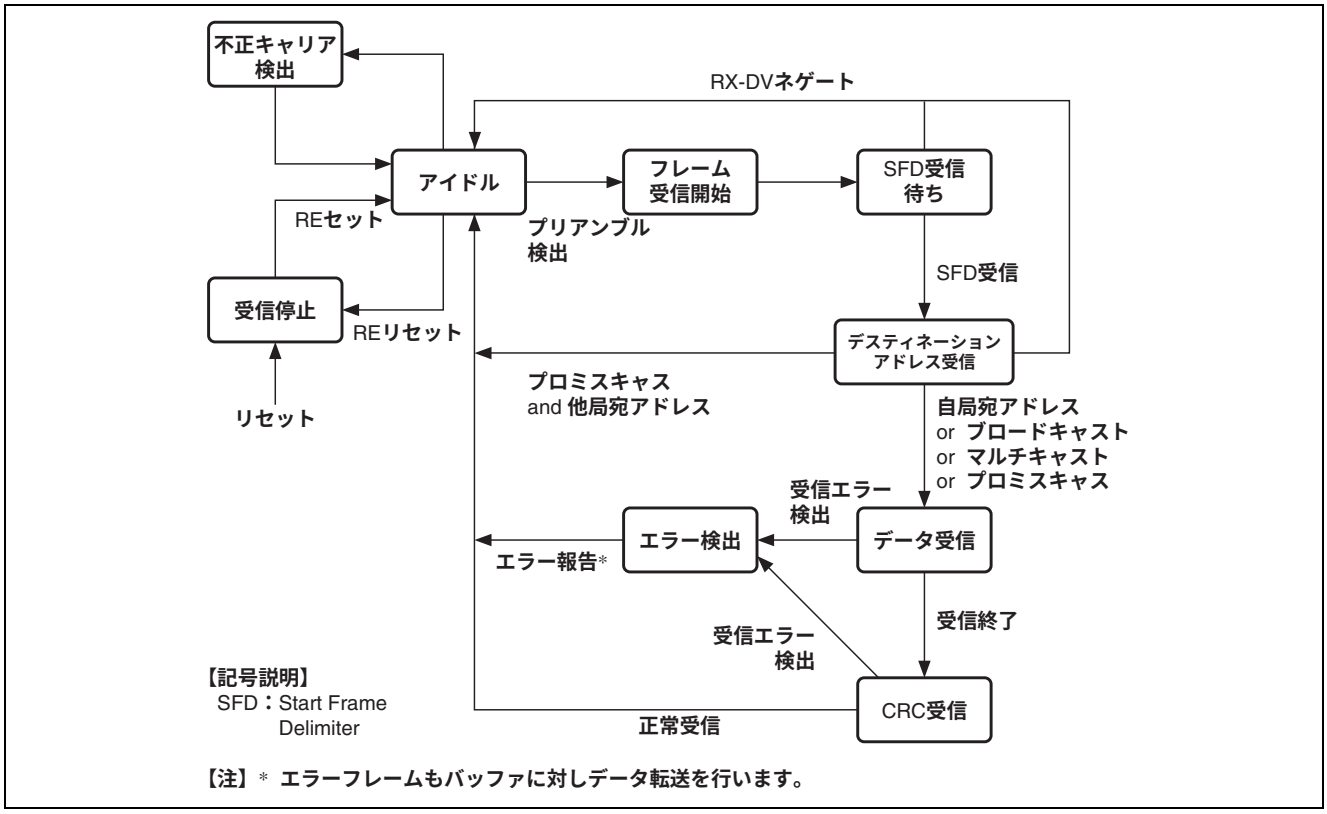


図2　EtherC 受信部状態遷移図

2.1.3　E-DMAC の概要

本 LSI は、EtherC に直結したダイレクトメモリアクセスコントローラ（E-DMAC）を内蔵しています。E-DMAC は、E-DMAC 内蔵の DMAC を使用し、E-DMAC 内の送信／受信 FIFO とユーザが指定するデータ格納先（送信／受信バッファ）との間で送受信データの DMA 転送を行います。CPU により直接送信／受信 FIFO のデータを読み書きすることはできません。この DMA 転送時に、E-DMAC が参照する情報を送信／受信ディスクリプタと呼び、ユーザがメモリ上に配置します。E-DMAC は、イーサネットフレーム送受信に先立ちディスクリプタの情報を読み出し、その内容にしたがって送信データを送信バッファから読み出し、または受信データを受信バッファへ書き込みます。このディスクリプタを複数個並べ、ディスクリプタ列（リスト）とすることで、複数のイーサネットフレームの送受信を連続的に行うことができます。

このE-DMACの機能によってCPUの負荷を軽減し、効率の良いデータ送受信制御を行うことができます。図3にE-DMACとディスクリプタおよびバッファの構成を示します。

E-DMAC の特長を以下に示します。

- 送信／受信 2 系統の独立した DMAC 内蔵
- ディスクリプタ管理方式による CPU 負荷の軽減
- 送受信フレームステータスのディスクリプタへの反映
- DMA ブロック転送（16 バイト単位）によるシステムバスの効率使用
- 1 フレーム／1 ディスクリプタ、1 フレーム／複数フレーム（マルチバッファ）方式対応可能（2.1.5節を参照）

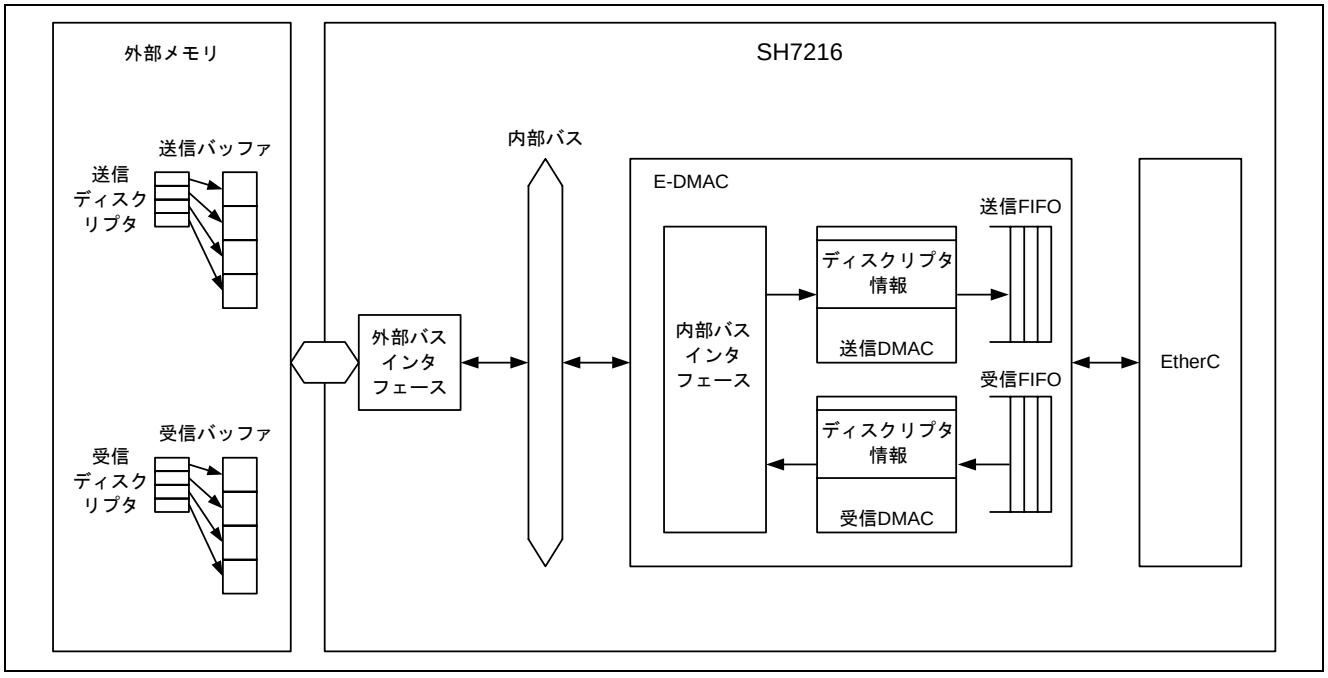


図3　E-DMAC とディスクリプタおよびバッファの構成

2.1.4　ディスクリプタの概要

E-DMAC が DMA 転送を行うためには、ディスクリプタと呼ばれる送受信データの格納アドレス等が書かれた情報（データ）が必要になります。ディスクリプタには送信ディスクリプタと受信ディスクリプタの 2 種類があります。E-DMAC は、E-DMAC 送信要求レジスタ（EDTRR）の TR ビットが 1 になると自動的に送信ディスクリプタの読み出しを、E-DMAC 受信要求レジスタ（EDRRR）の RR ビットが 1 になると自動的に受信ディスクリプタの読み出しを開始します。ユーザは送信／受信ディスクリプタにあらかじめ送信／受信データの DMA 転送に関する情報を記述しておく必要があります。イーサネットフレームの送信／受信が完了した後は、E-DMAC がディスクリプタの有効／無効ビット（送信時は TACT ビット、受信時は RACT ビット）を無効にし、送信／受信結果をステータスビット（送信時は TFS25～TFS0、受信時は RFS26～RFS0）に反映します。

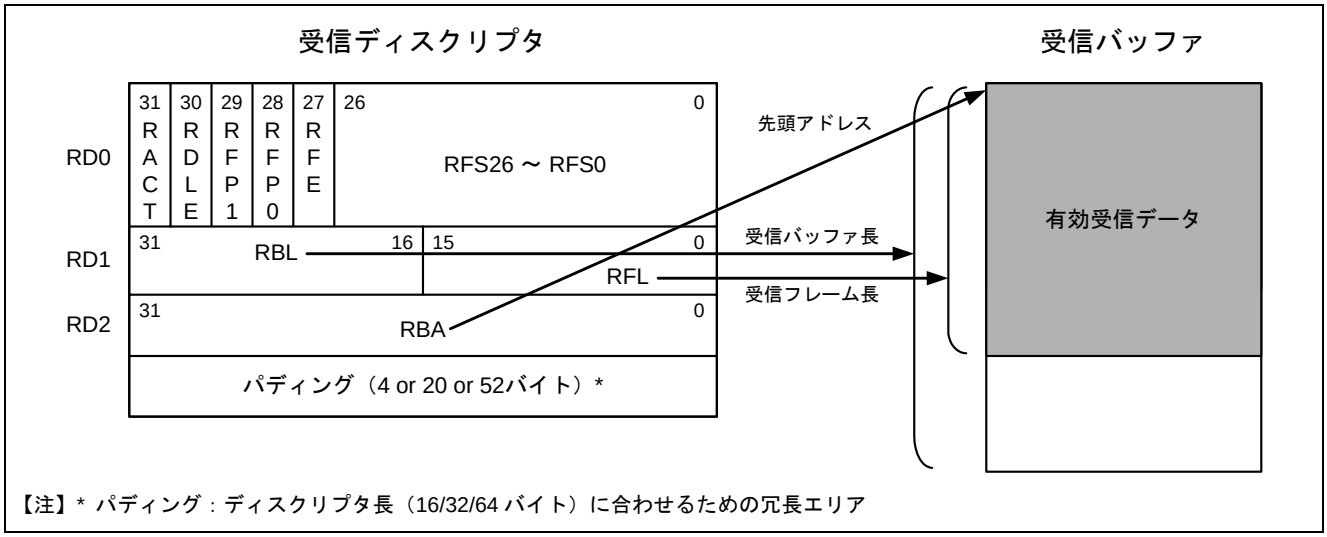
ディスクリプタは、読み書き可能なメモリ空間に配置し、先頭ディスクリプタ（E-DMAC が最初に読み出すディスクリプタ）のアドレスを送信ディスクリプタリスト先頭アドレスレジスタ（TDLAR）／受信ディスクリプタリスト先頭アドレスレジスタ（RDLAR）に設定します。複数のディスクリプタをディスクリプタ列（ディスクリプタリスト）として用意する場合には、E-DMAC モードレジスタ（EDMR）の DL0,1 ビットに設定したディスクリプタ長にしたがって連続したアドレスに配置します。

2.1.5　受信ディスクリプタの概要

図 4に受信ディスクリプタと受信バッファの関係を示します。

受信ディスクリプタは、データの先頭から 32 ビット単位に RD0, RD1, RD2 およびパディングで構成されます。RD0 は、受信ディスクリプタの有効／無効、ディスクリプタの構成情報およびステータス情報を示します。RD1 はそのディスクリプタが参照する受信バッファのサイズ（RBL）と受信したフレームのデータ長（RFL）を示します。RD2 は受信バッファの先頭アドレスを示します。最後のパディングは EDMR レジスタの DL0,1 ビットで指定するディスクリプタ長に従い長さが決まります。

受信ディスクリプタの設定内容により、ディスクリプタ 1 個で 1 フレームの受信データ全部を受信バッファに格納すること（1 フレーム／1 ディスクリプタ）も、ディスクリプタ複数個で 1 フレームの受信データを受信バッファに格納すること（1 フレーム／マルチディスクリプタ）も可能です。1 フレーム／マルチディスクリプタでは、あらかじめ複数のディスクリプタ（ディスクリプタリスト）を用意しておきます。E-DMAC は、受信したフレームがディスクリプタの RBL を超える長さのフレームを受信した場合には、連続する次のディスクリプタを使用していくことによって受信バッファに転送していきます。たとえば各ディスクリプタの RBL を 500 バイトとしたときに 1514 バイトのイーサネットフレームを受信したとします。受信したイーサネットフレームは最初のディスクリプタから順に 500 バイトずつバッファに転送され、最後の 14 バイトだけが 4 つ目のバッファに転送されます。



【注】* パディング：ディスクリプタ長（16/32/64 バイト）に合わせるための冗長エリア

図4　受信ディスクリプタと受信バッファの関係

2.1.6　受信ディスクリプタの設定例

図 5に受信ディスクリプタおよび受信バッファを各 3 面使用した場合の関係を示します。ここでは各受信バッファのサイズを 1536 バイト確保し、1 フレーム／1 ディスクリプタになるようにします。図では各受信ディスクリプタをRD0 部分のみに簡略化して記載しています。図中の番号①、②等は実行順を示します。

設定の順番は以下のようになります。

1. 全ディスクリプタ面の RFP1, RFP0 ビット、RFE ビット、RFS26～RFS0 ビットに 0 を設定します。
2. 第 1 面と第 2 面のディスクリプタの RDLE ビットに 0 を設定します。第 3 面のディスクリプタの RDLE ビットに 1 を設定することにより、第 3 面のディスクリプタの処理を終了すると第 1 面のディスクリプタを読み出します。このような設定によりディスクリプタをリング構造にすることができます。
3. 図 5では省略していますが、受信開始前に全ディスクリプタ面のRD1 のRBLに受信バッファサイズ 1536 バイトを、RD2 のRBAに対応する受信バッファの先頭アドレスを設定します。
4. 連続受信をさせるため、全ディスクリプタ面の RACT ビットに 1 を設定します。受信手順の詳細は次章で説明します。

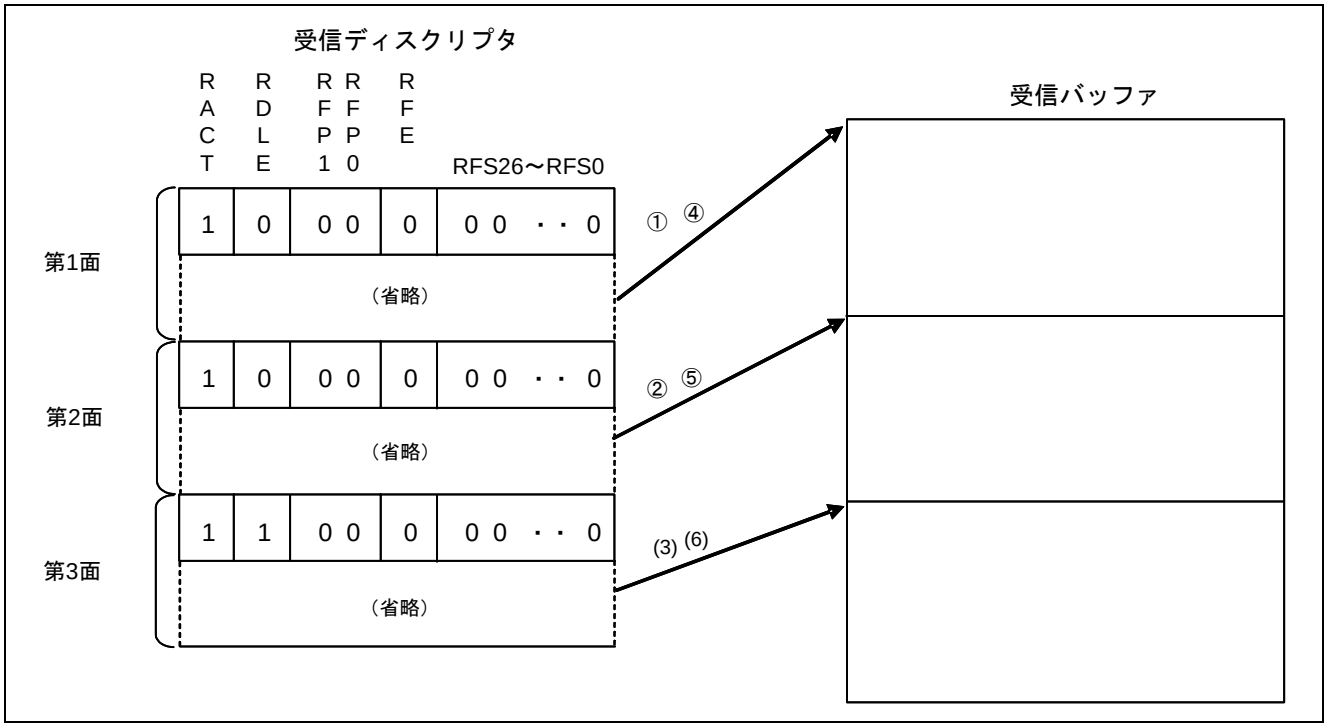


図5　受信ディスクリプタおよび受信バッファを各 3 面使用した場合の関係

2.1.7　使用機能の動作手順（受信時）

EtherC モードレジスタ（ECMR）の RE ビットが 1 の状態で E-DMAC 受信要求レジスタ（EDRRR）の受信要求ビット（RR）に 1 を書き込むと、E-DMAC 受信部が起動します。E-DMAC は、EtherC/E-DMAC のソフトウェアリセット後は受信ディスクリプタリスト先頭アドレスレジスタ（RDLAR）で示すディスクリプタを読み出し、RACT ビットが 1（有効）のときに受信待機状態になります。EtherC は自局あて（自局が受信を許可したアドレス）のフレームを受信すると、受信データを受信 FIFO に格納します。受信ディスクリプタの RACT ビットが 1 のときは、RD2 で指定される受信バッファに転送します（RACT ビットが 0（無効）の場合は、RR ビットをクリアして E-DMAC の受信動作を停止します）。受信したフレームのデータ長が RD1 で与えられるバッファ長よりも大きい場合は、E-DMAC はバッファが満了となった時点でディスクリプタにライトバック（RFP=B'10 or B'00）を行い次のディスクリプタを読み出します。フレームの受信が完了した場合、または何らかのエラーでフレーム受信を中断した場合は、当該ディスクリプタにライトバック（RFP=B'11 or B'01）を行います。その後、連続受信方式を選択している場合（受信方式制御レジスタ（RMCR）内の受信起動ビットリセットビット（RNR）が 1 の場合）、E-DMAC は次のディスクリプタを読み出し RACT ビットが 1 のときに受信待機状態になります。RACT ビットが 0 の場合（受信ディスクリプタ枯渇の場合）または、連続受信方式を選択していない場合（RMCR レジスタ内の RNR ビットが 0 の場合）は、EDRRR レジスタの RR ビットを 0 にし E-DMAC は受信処理を終了します。再度 RR ビットを 1 に設定すると、E-DMAC は最後に受信を行ったディスクリプタの次のディスクリプタを読み出します。なお RMCR レジスタの受信起動ビット non リセットモード指定ビット（RNC）を 1 に設定した場合、受信ディスクリプタ枯渇が発生しても EDRRR レジスタの RR ビットは 0 にならず、受信処理は継続されます。

図 6に受信フローの例（1 フレーム／1 ディスクリプタ、連続受信方式設定時の場合）を示します。

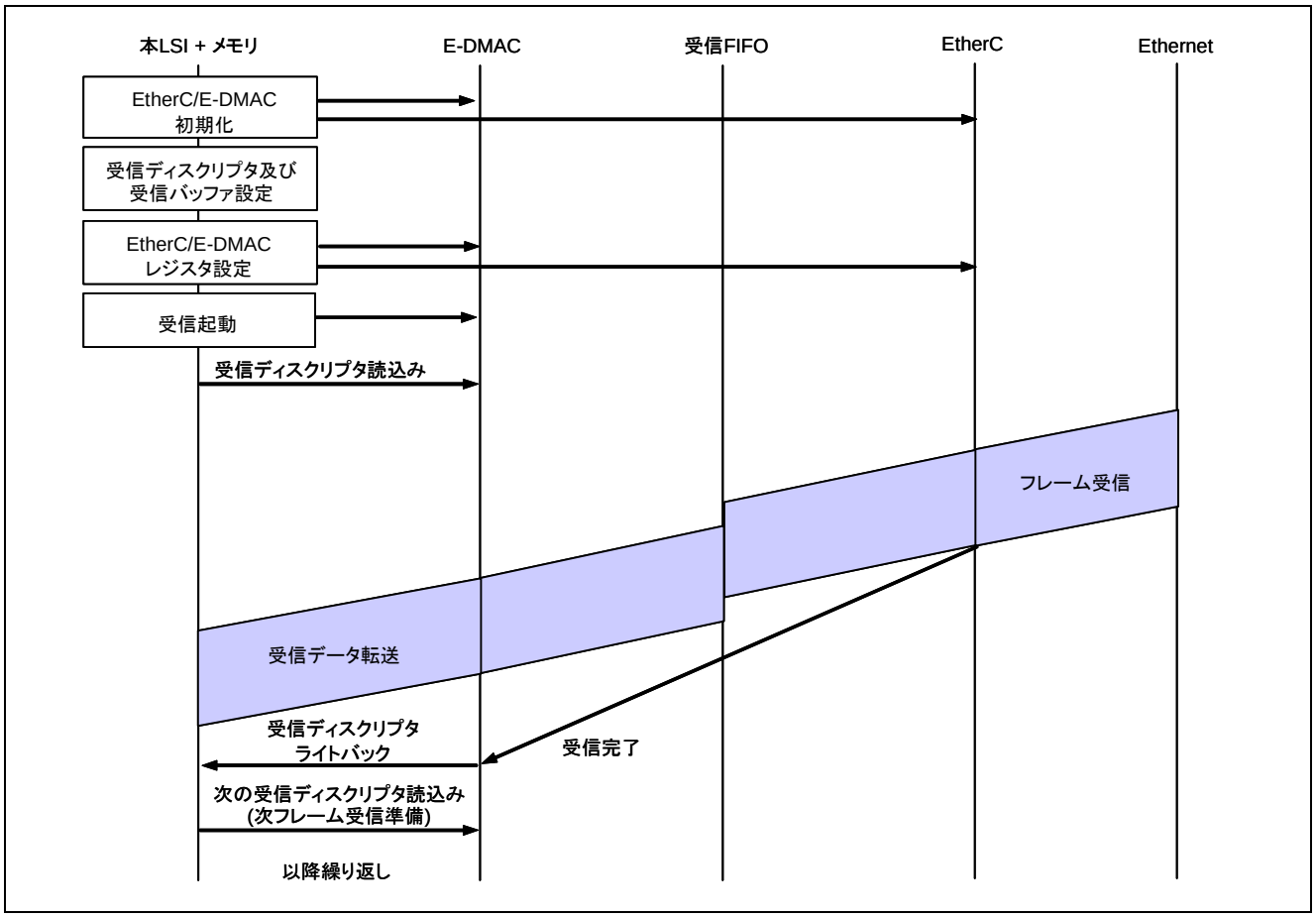


図6　受信フローの例（1 フレーム／1 ディスクリプタ）

2.1.8　使用機能の設定手順（受信時）

ここでは、イーサネット受信するための基本的な設定例について説明します。図 7および 図 8にイーサネット受信設定フロー例を示します。

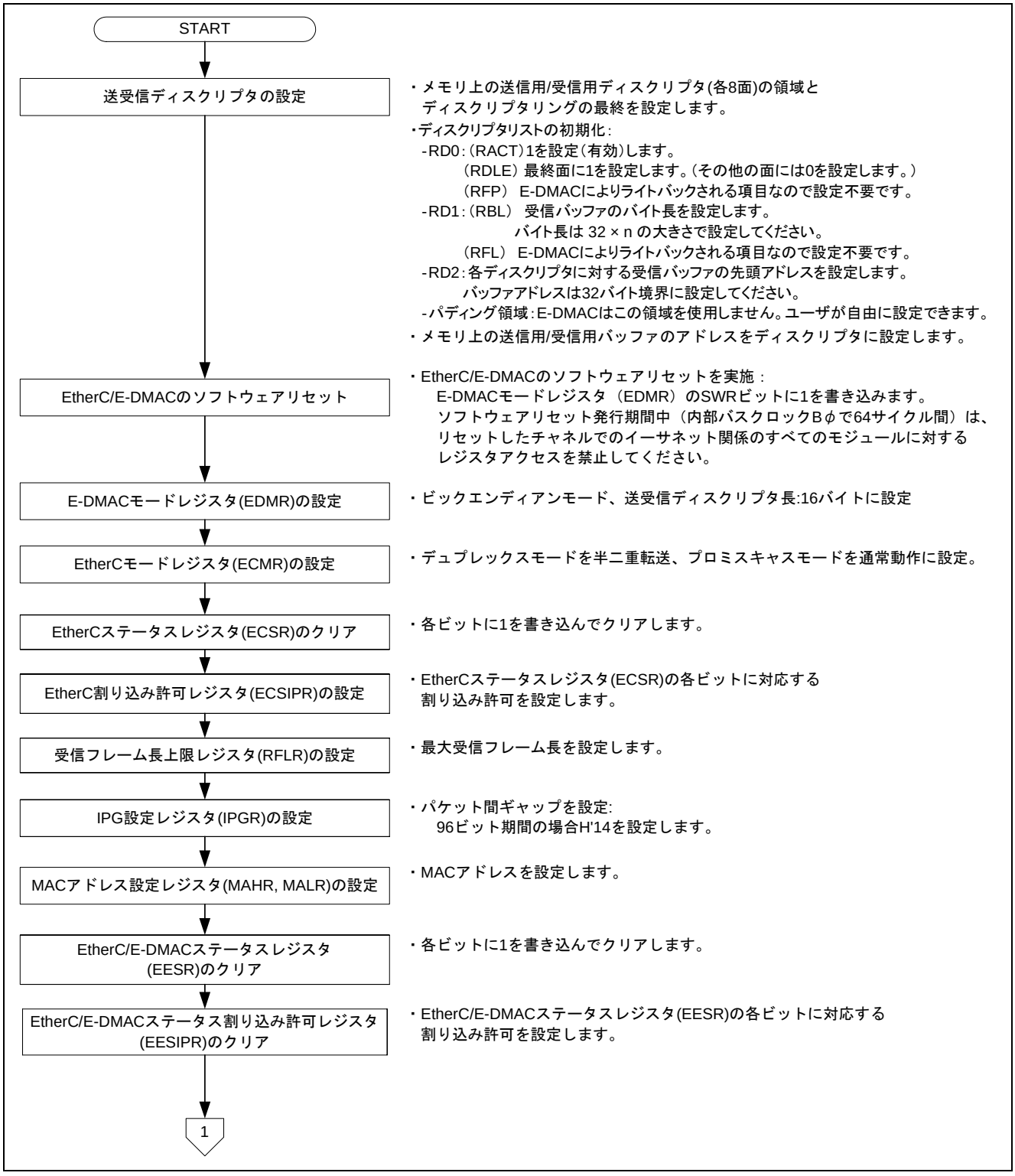


図7　イーサネット受信設定フロー例 (1)

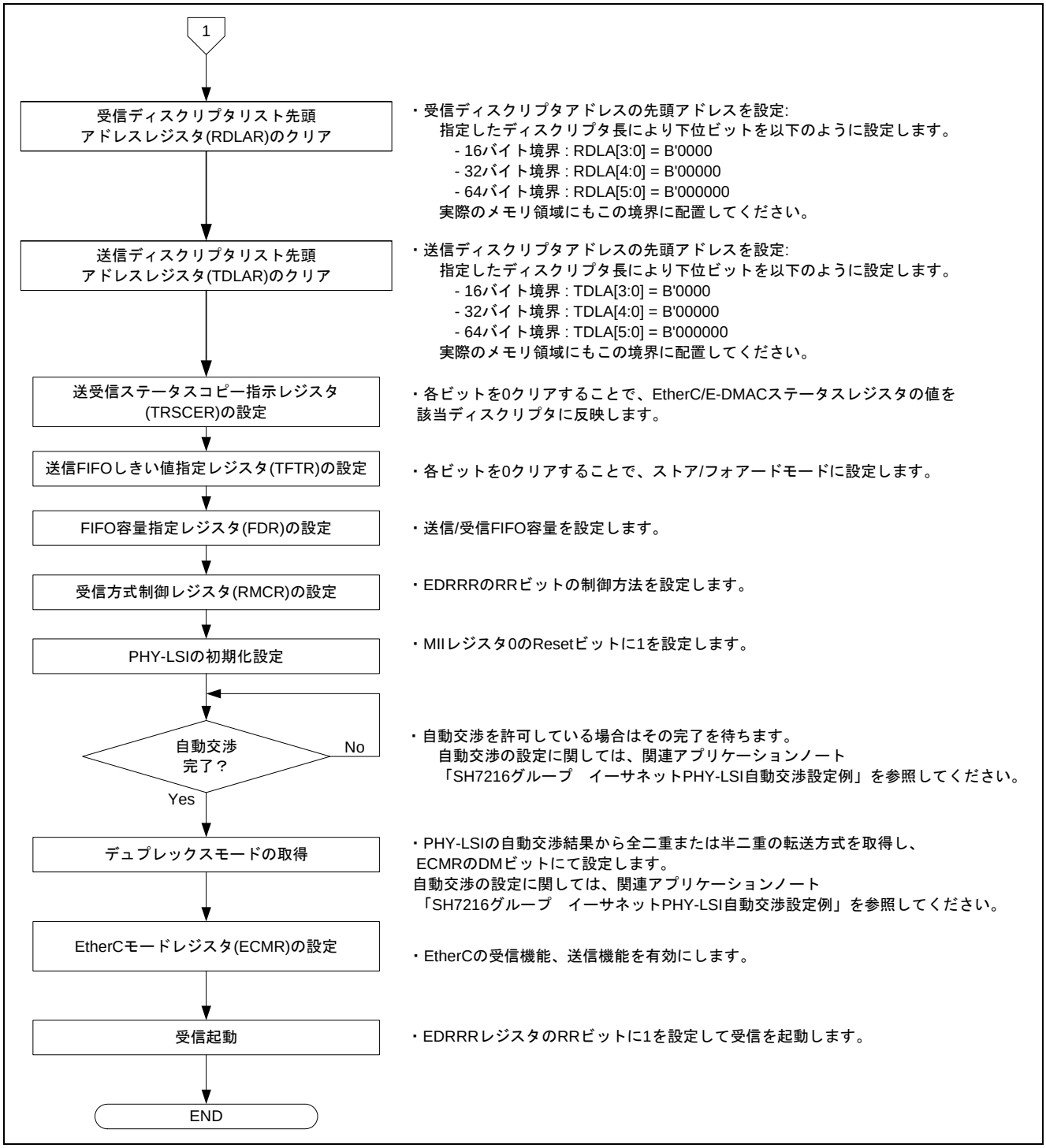


図8　イーサネット受信設定フロー例 (2)

2.2　参考プログラムの動作

参考プログラムでは、EtherC および E-DMAC を使用し、対向ホストからイーサネットフレームを 10 フレーム受信します。受信ディスクリプタと 256 バイトの受信バッファを 8 面用意しています。RMCR レジスタ内の RNR ビットに 1 を設定し、連続受信方式にしています。

なお、受信バッファにはイーサネットフレームのうちプリアンブル、SFD、および CRC を除いた部分が転送されます。

図 9に参考プログラムの動作環境を、図 10にイーサネットフレームフォーマットを示します。

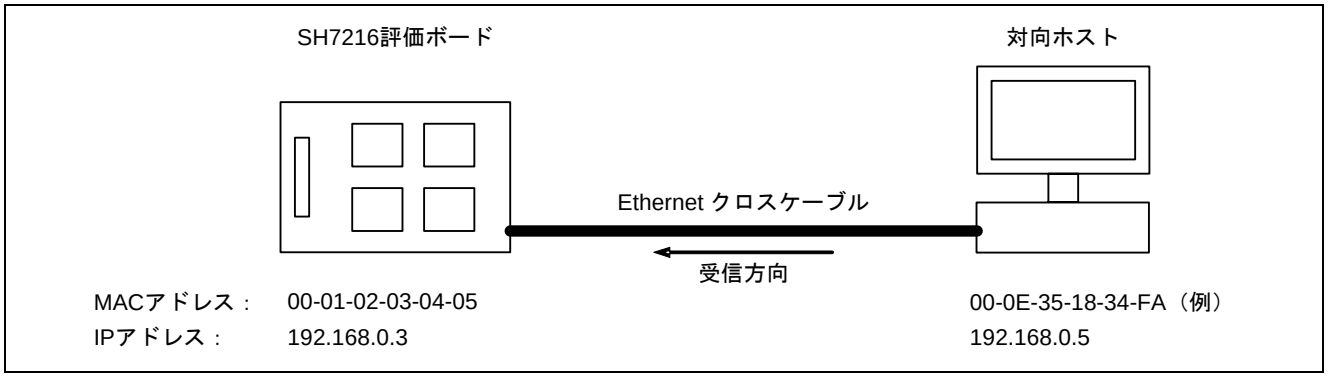


図9　参考プログラムの動作環境

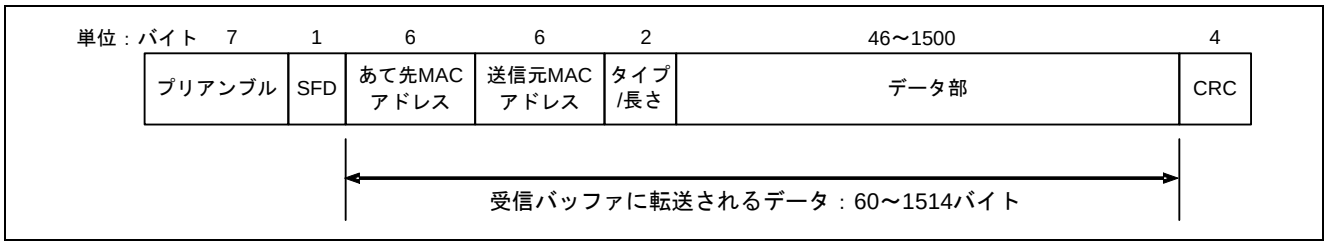


図10　イーサネットフレームフォーマット

2.3　参考プログラムのディスクリプタ定義

E-DMACではディスクリプタのパディング領域を使用しません。ユーザが自由に使用できます。本プログラムではこの領域に次のディスクリプタの先頭アドレスを設定し、ソフトウェアにてもリング構造を実現しています。図 11に参考プログラムでの受信ディスクリプタ構造体の定義と受信ディスクリプタ列の使用例を示します。

受信ディスクリプタの構造体定義

```
typedef struct Descriptor
{
        uint32_t            status;
        uint16_t            bufsize;
        uint16_t            size;
        int8_t              *buf_p;
        struct Descriptor   *next;
} ethfifo;
```

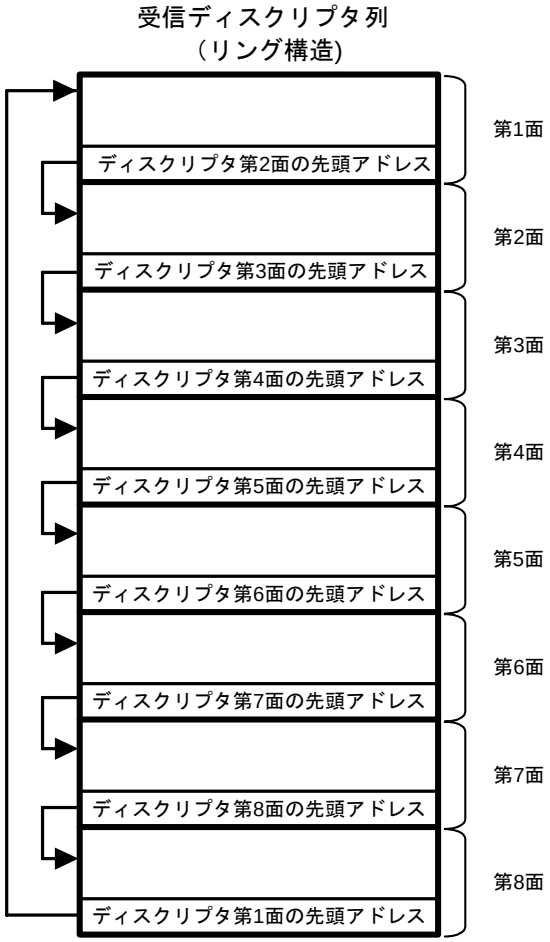


図11　受信ディスクリプタの構造体定義と受信ディスクリプタ列使用例

2.4　参考プログラムの処理手順

図 12～図 15に参考プログラムの処理フローを示します。

PHY 自動交渉関数 phy_set_autonegotiate の詳細は、関連アプリケーションノート「SH7216 グループ　イーサネット PHY-LSI 自動交渉設定例」を参照してください。

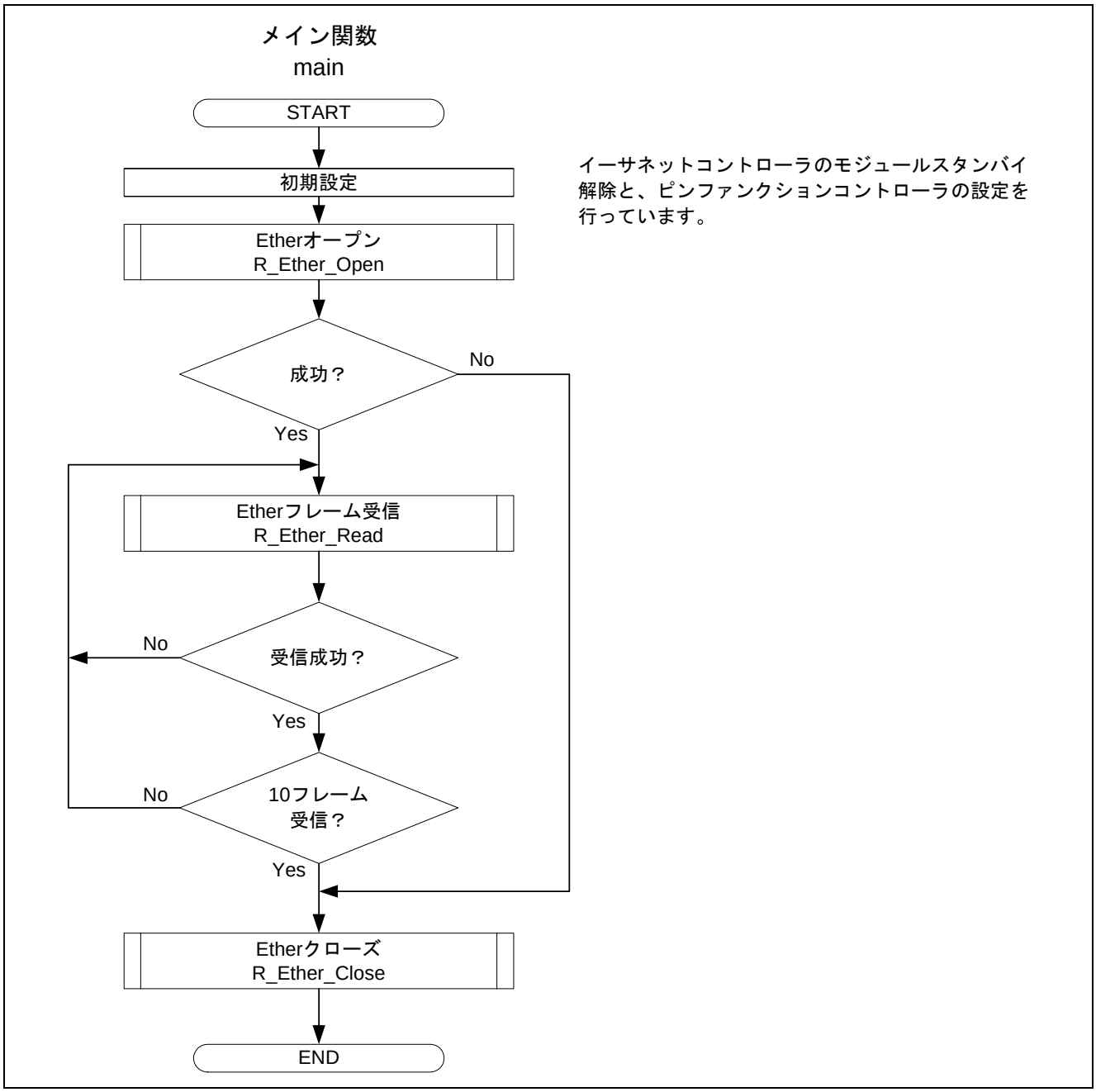


図12　参考プログラムの処理フロー例 (1)

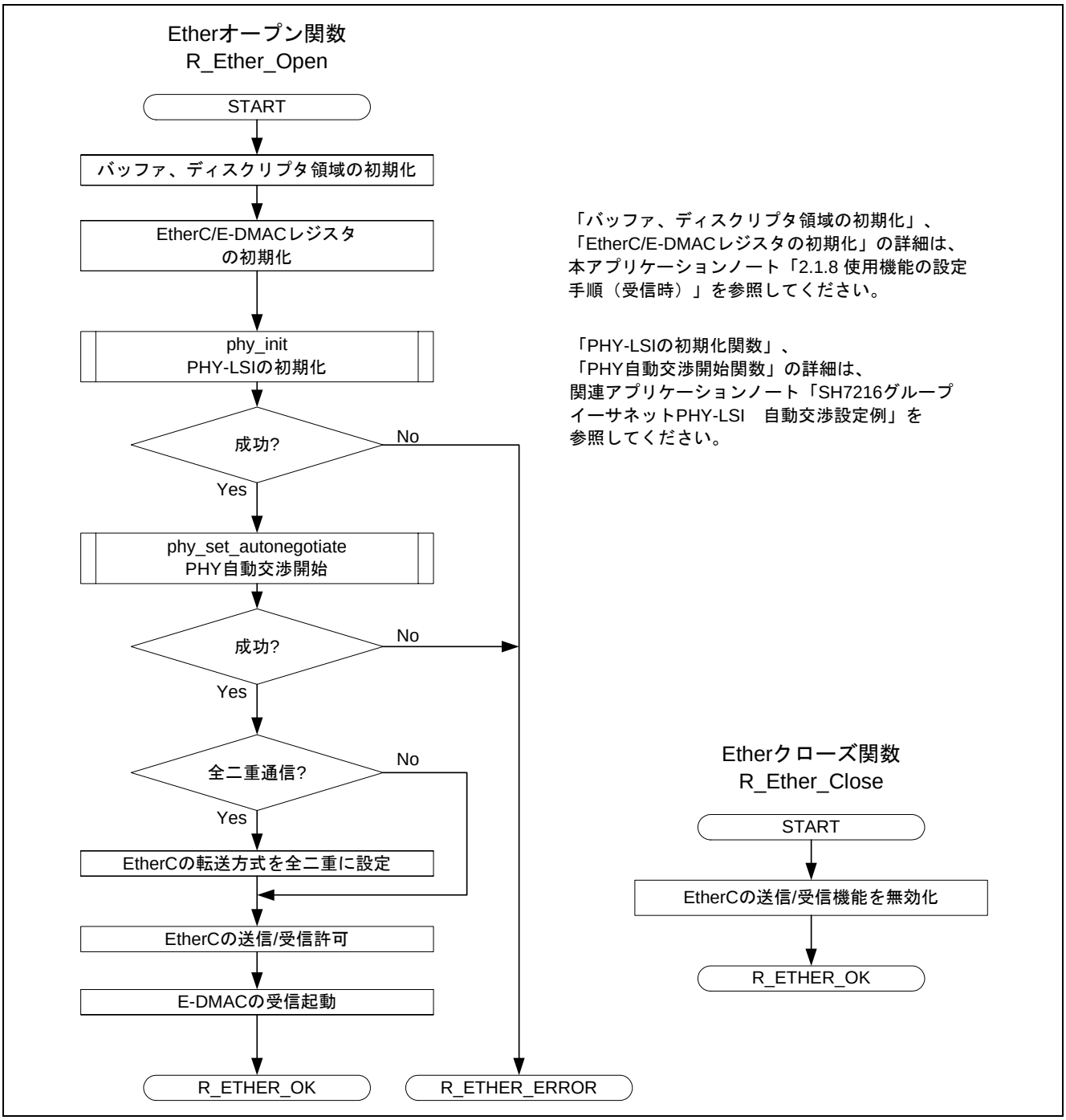


図13　参考プログラムの処理フロー例 (2)

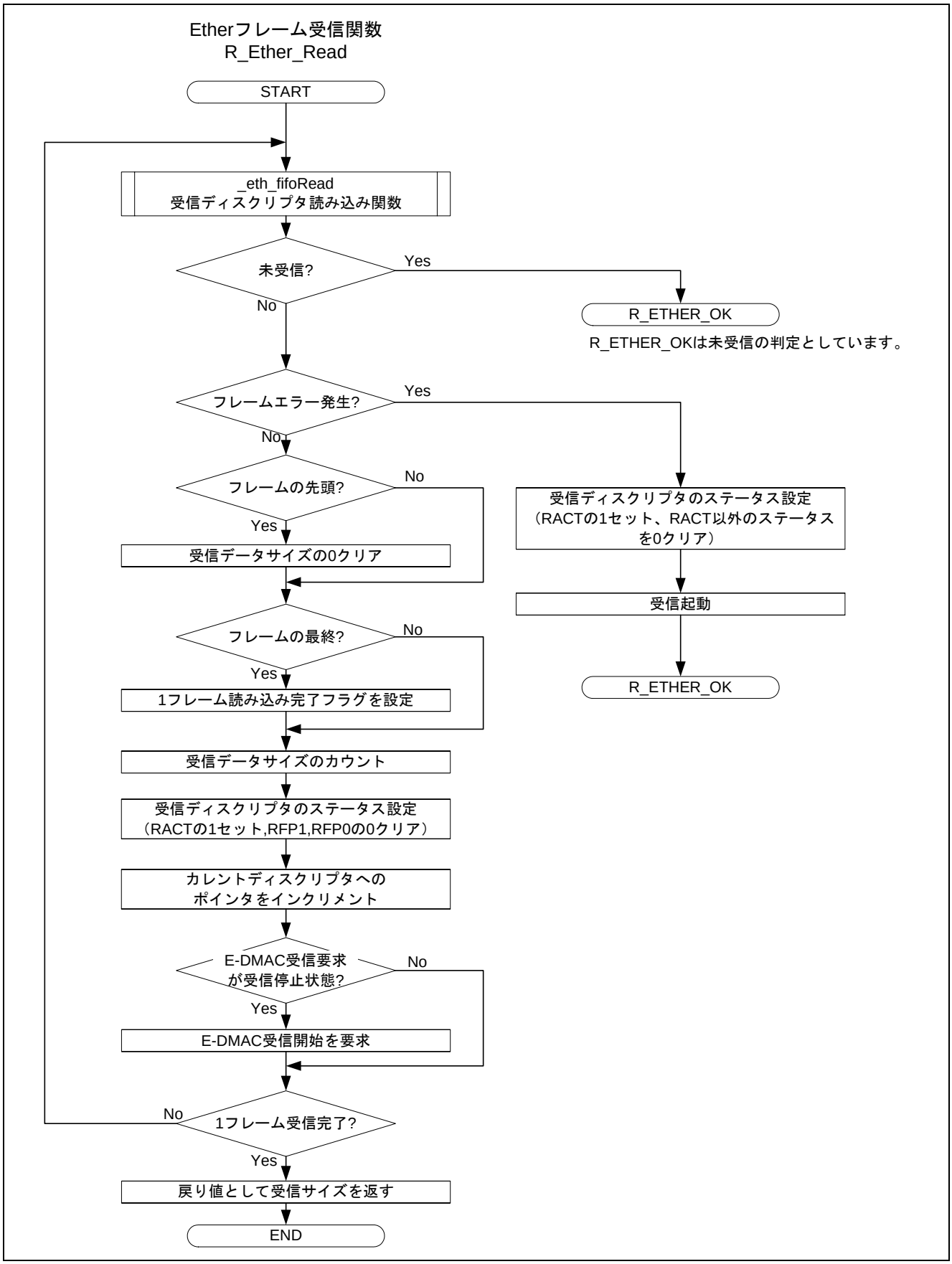


図14　参考プログラムの処理フロー例 (3)

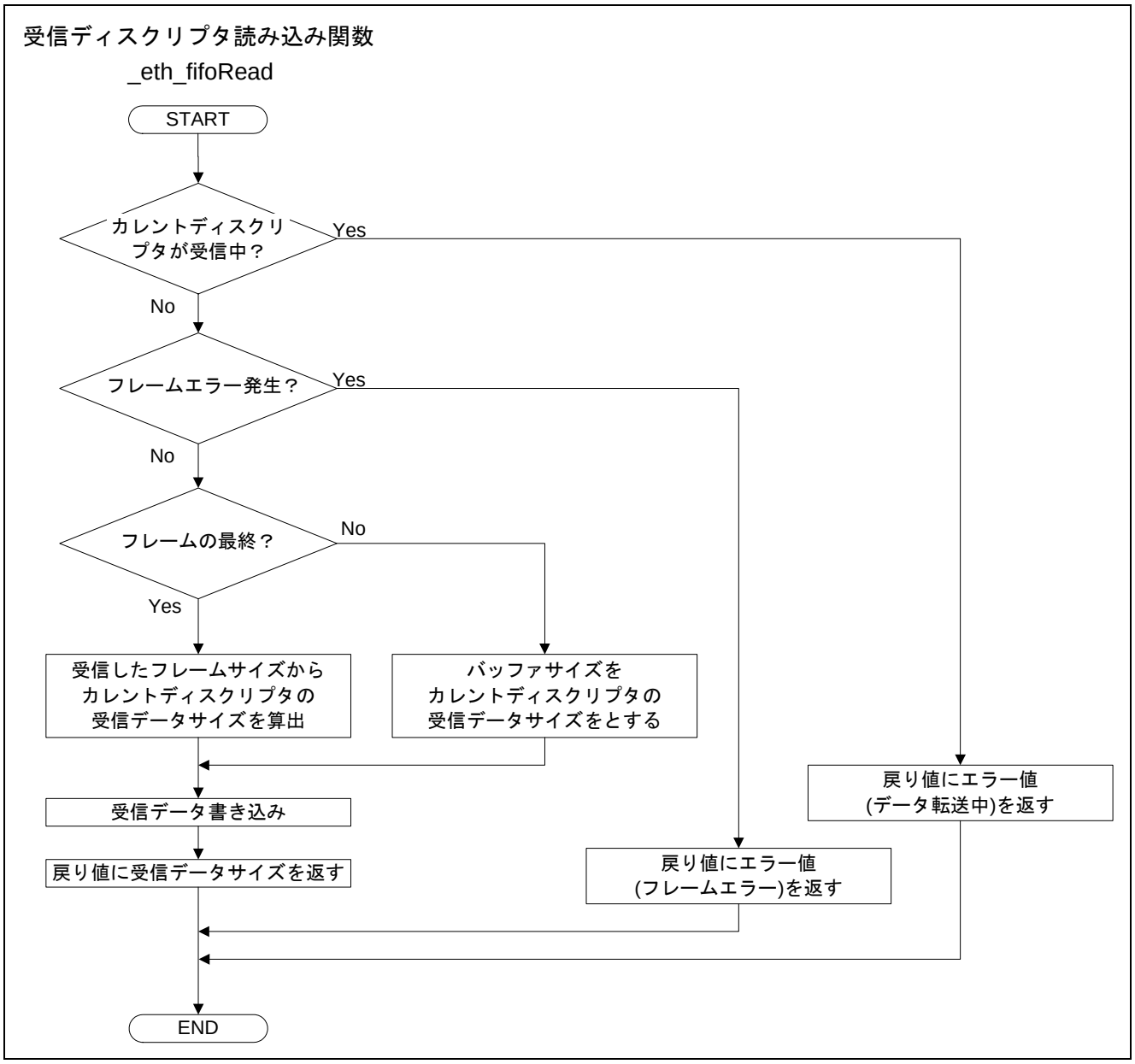


図15　参考プログラムの処理フロー例 (4)

3. 参考プログラムリスト

3.1 サンプルプログラムリスト "main.c" (1)

```
/***********************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corp. and is only
*   intended for use with Renesas products.  No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corp. and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
***********************************************************************
*   Copyright (C) 2010 Renesas Electronics Corporation. All Rights Reserved.
*""FILE COMMENT""************ Technical reference data *************************
*   System Name : SH7216 Sample Program
*   File Name   : main.c
*   Abstract    : イーサネット受信設定例
*   Version     : 2.00.00
*   Device      : SH7216
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                           (Ver.9.03 Release00).
*   OS          : None
*   H/W Platform: R0K572167 (CPU board)
*   Description : イーサネット受信に必要な初期設定および受信処理を行います。
***********************************************************************
*   History     : Nov.18,2009 Ver.1.00.00
*               : Jul.23,2010 Ver.2.00.00 ルネサス標準 API に対応
*""FILE COMMENT END""*****************************************************/
```

3.2　サンプルプログラムリスト "main.c" (2)

```
45    #include "iodefine.h"
46    #include "stdint.h"
47    #include "r_ether.h"
48    #include "phy.h"
49
50    /* ==== Prototype Declaration ==== */
51    void main(void);
52
53    /* ==== Variable Declaration ==== */
54    static uint8_t macaddr[] = {
55      0x00,0x01,0x02,0x03,0x04,0x05,      /* 送信元 MAC アドレス(00:01:02:03:04:05) */
56    };
57
58    static uint8_t r_frame[10][1536];     /* 受信フレームを格納するバッファ */
59
60    /*""FUNC COMMENT""*****************************************************************
61     * ID          :
62     * Outline     : サンプルプログラムメイン
63     *-----------------------------------------------------------------------------
64     * Include     : "iodefine.h", "stdint.h", "r_ether.h", and "phy.h"
65     *-----------------------------------------------------------------------------
66     * Declaration  : void main(void);
67     *-----------------------------------------------------------------------------
68     * Description  : 内蔵イーサネットコントローラ(EtherC)とイーサネット
69     *              : コントローラ用ダイナミックメモリアクセスコントローラ(E-DMAC)を
70     *              : 使用して、イーサネットフレームを受信します。
71     *              : また PHY モジュールには、REALTEK 社製 RTL8201CP を使用します。
72     *              : 受信ディスクリプタを複数面用意して連続受信します。
73     *-----------------------------------------------------------------------------
74     * Argument    : void
75     *-----------------------------------------------------------------------------
76     * Return Value : void
77     *-----------------------------------------------------------------------------
78     * Note        : None
79     *""FUNC COMMENT END""*************************************************************/
```

3.3　サンプルプログラムリスト "main.c" (3)

```
80    void main(void)
81    {
82     int32_t         i;
83     int32_t         ret;
84     uint32_t        ch = 0;
85
86     /* ==== EtherC/E-DMAC のモジュールスタンバイ解除 ==== */
87     STB.CR4.BIT._ETHER = 0;
88     /* ==== PFC 設定(EtherC 用) ==== */
89     PFC.PACRL4.BIT.PA12MD = 7; /* TX_CLK (input) */
90     PFC.PACRL3.WORD = 0x7777;  /* TX_EN,MII_TXD0,MII_TXD1,MII_TXD2 (output) */
91     PFC.PACRL2.BIT.PA7MD = 7;  /* MII_TXD3 (output) */
92     PFC.PACRL2.BIT.PA6MD = 7;  /* TX_ER (output) */
93     PFC.PDCRH4.WORD = 0x7777;  /* RX_DV,RX_ER,MII_RXD3,MII_RXD2 (input) */
94     PFC.PDCRH3.WORD = 0x7777;  /* MII_RXD1,MII_RXD0,RX_CLK,CRS (input) */
95     PFC.PDCRH2.WORD = 0x7777;  /* COL (input),WOL,EXOUT,MDC (input) */
96     PFC.PDCRH1.BIT.PD19MD = 7; /* LINKSTA (input) */
97     PFC.PDCRH1.BIT.PD18MD = 7; /* MDIO (input/output) */
98
99     /* ==== イーサネット初期設定 ==== */
100    ret = R_Ether_Open(ch, &macaddr[0]);
101
102    if(R_ETHER_OK == ret){
103     /* ==== 10 フレーム受信開始 ==== */
104     for(i = 0; i < 10; i++){
105       /* ---- 受信 ---- */
106       ret = R_Ether_Read(ch, &r_frame[i][0]);
107       if(ret == R_ETHER_OK){
108           i--;
109       }
110     }
111    }
112
113    /* ==== イーサネット送受信停止 ==== */
114    R_Ether_Close(ch);
115
116    while(1){
117     /* sleep */
118    }
119   }
120
121   /* End of file */
```

3.4　サンプルプログラムリスト "r_ether.c" (1)

```
1   /***********************************************************************
2   *   DISCLAIMER
3   *
4   *   This software is supplied by Renesas Electronics Corp. and is only
5   *   intended for use with Renesas products.  No other uses are authorized.
6   *
7   *   This software is owned by Renesas Electronics Corp. and is protected under
8   *   all applicable laws, including copyright laws.
9   *
10  *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11  *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12  *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13  *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14  *   DISCLAIMED.
15  *
16  *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17  *   ELECTRONICS CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18  *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19  *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20  *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21  *
22  *   Renesas reserves the right, without notice, to make changes to this
23  *   software and to discontinue the availability of this software.
24  *   By using this software, you agree to the additional terms and
25  *   conditions found by accessing the following link:
26  *   http://www.renesas.com/disclaimer
27  ***********************************************************************
28  *   Copyright (C) 2010 Renesas Electronics Corporation. All Rights Reserved.
29  *""FILE COMMENT""*********** Technical reference data *************************
30  *   System Name : SH7216 Sample Program
31  *   File Name   : r_ether.c
32  *   Version     : 2.00.00
33  *   Device      : SH7216
34  *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
35  *               : C/C++ compiler package for the SuperH RISC engine family
36  *               :                          (Ver.9.03 Release00).
37  *   OS          : None
38  *   H/W Platform: R0K572167 (CPU board)
39  *   Description : Ethernet module device driver
40  ***********************************************************************
41  *   History     : Jun.10.2009 Ver.1.00.00
42  *               : Jul.23,2010 Ver.2.00.00 ルネサス標準 API に対応
43  *""FILE COMMENT END""*************************************************************/
44  #include <machine.h>
45  #include <string.h>
46  #include "iodefine.h"
47  #include "stdint.h"
48  #include "r_ether.h"
49  #include "phy.h"
50
```

3.5　サンプルプログラムリスト "r_ether.c" (2)

```
51    /* ==== Prototype Declaration ==== */
52    void   _eth_fifoInit(ethfifo p[], uint32_t status);
53    int32_t _eth_fifoWrite(ethfifo *p, int8_t buf[], int32_t size);
54    int32_t _eth_fifoRead(ethfifo *p, int8_t buf[]);
55
56    #pragma section  _RX_DESC
57    volatile ethfifo rxDesc[ENTRY];       /* 受信ディスクリプタ */
58    #pragma section  _TX_DESC
59    volatile ethfifo txDesc[ENTRY];       /* 送信ディスクリプタ */
60    #pragma section
61
62    #pragma section  _RX_BUFF
63    int8_t rxbuf[ENTRY][BUFSIZE];         /* 受信データバッファ */
64    #pragma section  _TX_BUFF
65    int8_t txbuf[ENTRY][BUFSIZE];         /* 送信データバッファ */
66    #pragma section
67
68    /* ==== イーサネットデバイスドライバのコントロール構造体を初期化 ==== */
69    struct ei_device  le0 =
70    {
71      "eth0",      /* device name */
72      0,           /* open */
73      0,           /* Tx_act */
74      0,           /* Rx_act */
75      0,           /* txing */
76      0,           /* irq lock */
77      0,           /* dmaing */
78      0,           /* current receive descriptor */
79      0,           /* current transmit descriptor */
80      0,           /* save irq */
81      {
82        0,      /* rx packets */
83        0,      /* tx packets */
84        0,      /* rx errors */
85        0,      /* tx errors */
86        0,      /* rx dropped */
87        0,      /* tx dropped */
88        0,      /* multicast */
89        0,      /* collisions */
90
91        0,      /* rx length errors */
92        0,      /* rx over errors */
93        0,      /* rx CRC errors */
94        0,      /* rx frame errors */
95        0,      /* rx fifo errors */
96        0,      /* rx missed errors */
97
98        0,      /* tx aborted errors */
99        0,      /* tx carrier errors */
100       0,      /* tx fifo errors */
101       0,      /* tx heartbeat errors */
102       0       /* tx window errors */
```

3.6　サンプルプログラムリスト "r_ether.c" (3)

```
103     },
104     0,          /* MAC 0 */
105     0,          /* MAC 1 */
106     0,          /* MAC 2 */
107     0,          /* MAC 3 */
108     0,          /* MAC 4 */
109     0           /* MAC 5 */
110   };
111
112   /*""FUNC COMMENT""*****************************************************************
113    * ID          :
114    * Outline     : イーサネットオープン
115    *-------------------------------------------------------------------------------
116    * Include     : "iodefine.h" , "phy.h", "r_ether.h" and "stdint.h"
117    *-------------------------------------------------------------------------------
118    * Declaration  : int32_t R_Ether_Open(uint32_t ch, uint8_t mac_addr[]);
119    *-------------------------------------------------------------------------------
120    * Description  : EtherC, E-DMAC, PHY, バッファメモリの初期化を行います。
121    *              : 本関数内でイーサネットに必要な初期化を行い、送受信可能な状態に
122    *              : 設定します。
123    *              : 送受信可能な状態に設定することができなかった場合はエラーを返し
124    *              : ます。
125    *-------------------------------------------------------------------------------
126    * Argument    : uint32_t ch;       I : イーサネットのチャネル番号
127    *              : uint8_t mac_addr[]; I : 該当チャネルの MAC アドレス
128    *-------------------------------------------------------------------------------
129    * Return Value : R_ETHER_OK;    初期化成功
130    *              : R_ETHER_ERROR; 初期化失敗
131    *-------------------------------------------------------------------------------
132    * Note        : None
133    *""FUNC COMMENT END""*************************************************************/
134   int32_t R_Ether_Open(uint32_t ch, uint8_t mac_addr[])
135   {
136     int32_t  i;
137     uint32_t mac;
138     uint16_t phydata;
139
140     ch = ch;                          /* ワーニング抑止対策 */
141
142     /* ==== イーサネットデバイスドライバの設定 ==== */
143     le0.open = 1;
144     /* ==== ディスクリプタの設定 ==== */
145     _eth_fifoInit(rxDesc, (uint32_t)ACT);
146     _eth_fifoInit(txDesc, (uint32_t)0);
147     le0.rxcurrent = &rxDesc[0];
148     le0.txcurrent = &txDesc[0];
149     /* ==== MAC アドレス設定 ==== */
150     le0.mac_addr[0] = mac_addr[0];
151     le0.mac_addr[1] = mac_addr[1];
152     le0.mac_addr[2] = mac_addr[2];
153     le0.mac_addr[3] = mac_addr[3];
154     le0.mac_addr[4] = mac_addr[4];
155     le0.mac_addr[5] = mac_addr[5];
156
```

3.7　サンプルプログラムリスト "r_ether.c" (4)

```
157     /* ==== E-DMAC/EtherC の初期化 ==== */
158     EDMAC.EDMR.BIT.SWR = 1;                 /* ソフトウェアリセット有効 */
159     for( i = 0 ; i < 0x00000100 ; i++ );    /* E-DMAC/EtherC の初期化完了待ち(Bφ:64 サイクル) */
160     EDMAC.EDMR.LONG    = 0x00000000;     /* E-DMAC モードレジスタの設定 */
161                                           /* (ビックエンディアンモード) */
162                                           /* (送受信ディスクリプタ長 16 バイト) */
163     /* ==== EtherC の初期化 ==== */
164     EtherC.ECMR.LONG   = 0x00000000;     /* EtherC モードレジスタの設定 */
165                                           /* (デュプレックスモードを半二重に設定) */
166                                           /* (プロミスキャスモードを通常動作に設定) */
167
168     EtherC.ECSR.LONG   = 0x00000037;     /* EtherC ステータスのオールクリア */
169                                           /* (BFR, PSRTO, LCHNG, MPD, ICD) */
170     EtherC.ECSIPR.LONG = 0x00000020;     /* EtherC 割り込みの禁止設定 */
171          /* bit31~6 : Reserve : 0 ----- リザーブビット                    */
172          /* bit5    : BFSIPR  : 1 ----- Broadcast フレーム連続受信割り込み禁止 */
173          /* bit4    : PSRTOIP : 0 ----- Pause フレーム再送リトライオーバ割り込み禁止 */
174          /* bit3    : Reserve : 0 ----- リザーブビット                   */
175          /* bit2    : LCHNGIP : 0 ----- リンク信号変化割り込み禁止       */
176          /* bit1    : MPDIP   : 0 ----- Magic Packet 検出割り込み禁止     */
177          /* bit0    : ICDIP   : 0 ----- 不正キャリア検出割り込み禁止      */
178
179     EtherC.RFLR.LONG   = 1518;           /* 最大受信フレーム長の設定*/
180     EtherC.IPGR.LONG   = 0x00000014;     /* パケット間ギャップの設定 (96 ビット期間) */
181
182     /* ==== MAC アドレスの設定 ==== */
183     mac = ((uint32_t)mac_addr[0] << 24) |
184     ((uint32_t)mac_addr[1] << 16) |
185     ((uint32_t)mac_addr[2] << 8 ) |
186     (uint32_t)mac_addr[3];
187     EtherC.MAHR = mac;
188
189     mac = ((uint32_t)mac_addr[4] << 8 ) |
190     (uint32_t)mac_addr[5];
191     EtherC.MALR.LONG = mac;
192
193     /* ==== E-DMAC の初期化 ==== */
194     EDMAC.EESR.LONG   = 0x47FF0F9F;         /* EtherC/E-DMAC ステータスレジスタの初期化 */
195     EDMAC.EESIPR.LONG = 0x00000000;         /*EtherC/E-DMAC ステータス割り込み許可レジスタの初期化 */
196     EDMAC.RDLAR      = le0.rxcurrent;       /* 送信ディスクリプタ先頭アドレスの設定 */
197     EDMAC.TDLAR      = le0.txcurrent;       /* 受信ディスクリプタ先頭アドレスの設定 */
198     EDMAC.TRSCER.LONG = 0x00000000;         /* EtherC/E-DMAC ステータスレジスタの値を */
199                                             /*             該当ディスクリプタに反映 */
200     EDMAC.TFTR.LONG   = 0x00000000;         /* ストア&フォワードモードの設定 */
201     EDMAC.FDR.LONG    = 0x00000000;         /* 送信/受信 FIFO 容量の設定 (256 バイト) */
202     EDMAC.RMCR.LONG   = 0x00000001;         /* 受信ディスクリプタの枯渇以外では連続受信に設定 */
203
```

3.8　サンプルプログラムリスト "r_ether.c" (5)

```
204     /* ==== PHY の初期化 ==== */
205     phydata = phy_init();
206
207     if(phydata == R_PHY_ERROR){
208       return R_ETHER_ERROR;
209     }
210
211     /* ==== PHY の自動交渉開始 ==== */
212     phydata = phy_set_autonegotiate();
213
214     /* ---- 自動交渉の可否判定 ---- */
215     if(phydata == R_PHY_ERROR){             /* 自動交渉失敗 */
216       return R_ETHER_ERROR;
217     }
218
219     /* ---- 接続先性能の判定 ---- */
220     if(phydata & 0x0100){                   /* PHY-LSI のレジスタ 0 の判定 */
221                                             /* bit8 : DuplexMode : 1 ---- 全二重モードをサポート */
222       EtherC.ECMR.BIT.DM = 1;               /* 全二重通信 */
223     }
224
225     /* ==== EtherC 送受信許可 ==== */
226     EtherC.ECMR.BIT.RE = 1;
227     EtherC.ECMR.BIT.TE = 1;
228
229     /* ==== E-DMAC 受信許可 ==== */
230     EDMAC.EDRRR.LONG = 0x00000001;
231
232     return R_ETHER_OK;
233   }
234
235   /*""FUNC COMMENT""*****************************************************************
236    * ID          :
237    * Outline     : イーサネットクローズ
238    *-------------------------------------------------------------------------------
239    * Include     : "iodefine.h" , "r_ether.h" and "stdint.h"
240    *-------------------------------------------------------------------------------
241    * Declaration  : int32_t R_Ether_Close(uint32_t ch);
242    *-------------------------------------------------------------------------------
243    * Description  : EtherC/E-DMAC を停止します。
244    *-------------------------------------------------------------------------------
245    * Argument     : uint32_t ch; I : イーサネットのチャネル番号
246    *-------------------------------------------------------------------------------
247    * Return Value : R_ETHER_OK; EtherC 送受信禁止
248    *-------------------------------------------------------------------------------
249    * Note         : None
250    *""FUNC COMMENT END""*************************************************************/
```

3.9　サンプルプログラムリスト "r_ether.c" (6)

```
251    int32_t R_Ether_Close(uint32_t ch)
252    {
253      ch = ch;                                  /* ワーニング抑止対策 */
254
255      le0.open = 0;
256      EtherC.ECMR.LONG = 0x00000000;            /* EtherC 送受信禁止 */
257      le0.irqlock = 1;
258
259      return R_ETHER_OK;
260    }
261
       (省略)
317    /*""FUNC COMMENT""*****************************************************************
318     * ID          :
319     * Outline     : フレーム受信処理
320     *-------------------------------------------------------------------------------
321     * Include     : "iodefine.h" , "r_ether.h" and "stdint.h"
322     *-------------------------------------------------------------------------------
323     * Declaration  : int32_t R_Ether_Read(uint32_t ch, void *buf);
324     *-------------------------------------------------------------------------------
325     * Description  : 受信ディスクリプタに登録されたバッファから、指定されたフレーム
326     *              : をコピーして受信します。
327     *              : またデータ未受信、エラー発生の判定を行います。
328     *-------------------------------------------------------------------------------
329     * Argument    : uint32_t ch; I : イーサネットのチャネル番号
330     *             : void *buf;   I : 受信データを格納する領域のポインタ
331     *-------------------------------------------------------------------------------
332     * Return Value : R_ETHER_OK; データ受信なし
333     *              : 1以上;      受信データサイズ
334     *-------------------------------------------------------------------------------
335     * Note        : None
336     *""FUNC COMMENT END""*************************************************************/
337    int32_t R_Ether_Read(uint32_t ch, void *buf)
338    {
339      int32_t  receivesize = 0;        /* 受信データサイズ */
340      int32_t  recvd;                  /* 受信したデータのサイズ、または受信ディスクリプタの状態 */
341      int32_t  flag = 1;               /* 1フレーム受信未完了フラグ(1: 1フレーム受信未完了) */
342      uint8_t  readcount = 0;          /* 1フレーム受信完了までの受信ディスクリプタの読出し回数 */
343      int8_t   *data = (int8_t *)buf;  /* 受信データを格納する領域のポインタ */
344
345      ch = ch;                        /* ワーニング抑止対策 */
346
347      /* ==== 1フレーム受信処理 ==== */
348      while (flag){
349        recvd = _eth_fifoRead(le0.rxcurrent, data);
350        readcount++;
351        /* ---- データ未受信 ---- */
352        if (readcount >= 2 && receivesize == 0){
353            return R_ETHER_OK;            /* データ受信なし */
354        }
355
```

3.10　サンプルプログラムリスト "r_ether.c" (7)

```
356         /* ---- 受信済みデータなし ---- */
357         if (recvd == -1){
358         }
359         /* ---- 受信フレームエラー ---- */
360         else if (recvd == -2){
361             le0.stat.rx_errors++;
362 
363             /* ---- 受信ディスクリプタを再受信可能な状態に戻して終了 ---- */
364             receivesize = 0;              /* 戻り値を"R_ETHER_OK"にするための設定 */
365             le0.rxcurrent->status &= ~(FP1 | FP0 | FE);
366             le0.rxcurrent->status &= ~(RMAF | RRF | RTLF | RTSF | PRE | CERF);
367             le0.rxcurrent->status |= ACT;
368             le0.rxcurrent = le0.rxcurrent->next;
369 
370             /* ---- フレーム受信開始 ---- */
371             if (EDMAC.EDRRR.LONG == 0x00000000L){
372                 EDMAC.EDRRR.LONG = 0x00000001L;
373             }
374         }
375         /* ---- 受信データあり ---- */
376         else{
377             /* ---- フレームの先頭を受信 ---- */
378             if ((le0.rxcurrent->status & FP1) == FP1){
379                 receivesize = 0;
380             }
381             /* ---- フレームの最終を受信(フレーム受信完了) ---- */
382             if ((le0.rxcurrent->status & FP0) == FP0){
383                 le0.stat.rx_packets++;   /* 総受信フレーム数をカウント */
384                 flag = 0;                /* 1フレーム受信未完了フラグを設定(0: 1フレーム受信完了) */
385             }
386 
387             /* ---- 受信データをフレームサイズにカウント ---- */
388             receivesize += recvd;
389             /* ---- 受信ディスクリプタを受信継続可能な状態に戻す ---- */
390             le0.rxcurrent->status &= ~(FP1 | FP0);
391             le0.rxcurrent->status |= ACT;
392             le0.rxcurrent = le0.rxcurrent->next;
393             /* ---- 受信データ格納領域を更新 ---- */
394             data += recvd;
395 
396             /* ==== E-DMAC受信要求の判定 ==== */
397             if (EDMAC.EDRRR.LONG == 0x00000000L){
398                 EDMAC.EDRRR.LONG = 0x00000001L;
399             }
400         }
401       }
402 
403       return (int32_t)receivesize;
404     }
405 
```

3.11　サンプルプログラムリスト "r_ether.c" (8)

```
406    /*""FUNC COMMENT""*****************************************************************
407     * ID          :
408     * Outline     : FIFOの初期化
409     *-------------------------------------------------------------------------------
410     * Include     :
411     *-------------------------------------------------------------------------------
412     * Declaration  : void _eth_fifoInit( ethfifo p[], uint32_t status );
413     *-------------------------------------------------------------------------------
414     * Description  : E-DMACのディスクリプタを初期化します。
415     *-------------------------------------------------------------------------------
416     * Argument    : ethfifo p[];    O : ディスクリプタへのポインタ
417     *             : uint32_t status; I : ディスクリプタの初期ステータス
418     *-------------------------------------------------------------------------------
419     * Return Value : void
420     *-------------------------------------------------------------------------------
421     * Note         : None
422     *""FUNC COMMENT END""*************************************************************/
423    void _eth_fifoInit( ethfifo p[], uint32_t status )
424    {
425      ethfifo  *current = 0;
426      int32_t i, j;
427
428      for( i = 0 ; i < ENTRY ; i++ ){
429        current = &p[i];
430        /* ==== ディスクリプタステータスの判定 ==== */
431        if( status == 0 ){
432            current->buf_p = &txbuf[i][0];/* ACTビットが0のとき送信と判断 */
433        }
434        else{
435            current->buf_p = &rxbuf[i][0];/* ACTビットが1のとき受信と判断 */
436        }
437
438        /* ==== バッファのクリア ==== */
439        for( j = 0 ; j < BUFSIZE ; j++ ){
440            current->buf_p[j] = 0;
441        }
442
443        current->bufsize = BUFSIZE;
444        current->size = 0;
445        current->status = status;
446        current->next = &p[i+1];
447      }
448      /* ==== 最終FIFOのエントリー完了待ち ==== */
449      current->status |= DL;   /* カレントディスクリプタをディスクリプタリングの最終に設定 */
450      current->next = &p[0];
451    }
452
       (省略)
497    /*""FUNC COMMENT""*****************************************************************
498     * ID          :
499     * Outline     : 受信ディスクリプタの読み出し
500     *-------------------------------------------------------------------------------
501     * Include     :
502     *-------------------------------------------------------------------------------
503     * Declaration  : int32_t _eth_fifoRead( ethfifo *p, int8_t buf[]);
```

3.12　サンプルプログラムリスト "r_ether.c" (9)

```
504    *-------------------------------------------------------------------------------
505     * Description  : 受信ディスクリプタのデータを引数で指定した領域に読み出します。
506     *-------------------------------------------------------------------------------
507     * Argument     : ethfifo *p;   O : カレントディスクリプタへのポインタ
508     *              : int8_t buf[]; O : 受信データを格納する領域へのポインタ
509     *-------------------------------------------------------------------------------
510     * Return Value : 0以上;     読み出したデータのサイズ
511     *              : -1;        カレントディスクリプタが受信動作もしくは受信準備中
512     *              : -2;        受信フレームエラー発生
513     *-------------------------------------------------------------------------------
514     * Note         : None
515     *""FUNC COMMENT END""***********************************************************/
516    int32_t _eth_fifoRead( ethfifo *p, int8_t buf[])
517    {
518      int32_t i, temp_size;                  /* バッファサイズカウンタ(byte) */
519      ethfifo  *current = p;
520
521      /* ==== カレントディスクリプタが受信動作もしくは受信準備中 ==== */
522      if( (current->status & ACT) != 0){
523        return (-1);                       /* これはエラーではありません */
524      }
525
526      /* ==== 受信フレームエラー ==== */
527      else if( (current->status & FE) != 0){
528        return (-2);                       /* ディスクリプタの更新が必要 */
529      }
530
531      /* ==== 正常受信 ==== */
532      else{
533        /* ---- フレームの最終(1フレーム受信完了) ---- */
534        if ((current->status & FP0) == FP0){
535            /* ---- カレントディスクリプタの受信データサイズを算出 ---- */
536            temp_size = current->size;     /* フレームの総バイト数 */
537
538            while (temp_size > BUFSIZE){
539                temp_size -= BUFSIZE;      /* フレームの最終で受信したデータサイズを算出 */
540            }
541        }
542        /* ---- フレームの最終ではない場合 ---- */
543        else{
544            temp_size = BUFSIZE;
545        }
546
547        /* ---- 受信ディスクリプタから受信バッファにデータをコピー ---- */
548        for (i = 0; i < temp_size; i++){
549            buf[i] = current->buf_p[i];
550        }
551
552        return (temp_size);                /* 受信データサイズ */
553      }
554    }
555
556    /* End of File */
```

3.13　サンプルプログラムリスト "r_ether.h" (1)

```
1   /***********************************************************************
2   *   DISCLAIMER
3   *
4   *   This software is supplied by Renesas Electronics Corp. and is only
5   *   intended for use with Renesas products.  No other uses are authorized.
6   *
7   *   This software is owned by Renesas Electronics Corp. and is protected under
8   *   all applicable laws, including copyright laws.
9   *
10  *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11  *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12  *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13  *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14  *   DISCLAIMED.
15  *
16  *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17  *   ELECTRONICS CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18  *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19  *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20  *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21  *
22  *   Renesas reserves the right, without notice, to make changes to this
23  *   software and to discontinue the availability of this software.
24  *   By using this software, you agree to the additional terms and
25  *   conditions found by accessing the following link:
26  *   http://www.renesas.com/disclaimer
27  ***********************************************************************
28  *   Copyright (C) 2010 Renesas Electronics Corporation. All Rights Reserved.
29  *""FILE COMMENT""************ Technical reference data *************************
30  *   System Name : SH7216 Sample Program
31  *   File Name    : r_ether.h
32  *   Version      : 2.00.00
33  *   Device       : SH7216
34  *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
35  *               : C/C++ compiler package for the SuperH RISC engine family
36  *               :                          (Ver.9.03 Release00).
37  *   OS           : None
38  *   H/W Platform: R0K572167 (CPU board)
39  *   Description : Ethernet module device driver
40  ***********************************************************************
41  *   History      : Jun.10.2009 Ver.1.00.00
42  *               : Jul.23,2010 Ver.2.00.00 ルネサス標準 API に対応
43  *""FILE COMMENT END""*************************************************************/
44  #ifndef ETH_H
45  #define ETH_H
46
```

3.14　サンプルプログラムリスト "r_ether.h" (2)

```
47    /* ==== Type definition ==== */
48    typedef struct Descript
49    {
50      uint32_t     status;
51      uint16_t     bufsize;
52      uint16_t     size;
53      int8_t          *buf_p;
54      struct Descript *next;
55    } ethfifo;
56
57    /* ==== Macro definition ==== */
58    #define BUFSIZE  256
59    #define ENTRY    8
60
61    #define  ACT     0x80000000
62    #define  DL      0x40000000
63    #define  FP1     0x20000000
64    #define  FP0     0x10000000
65    #define  FE      0x08000000
66
67    #define  RFOVER   0x00000200
68    #define  RMAF    0x00000080
69    #define  RRF     0x00000010
70    #define  RTLF    0x00000008
71    #define  RTSF    0x00000004
72    #define  PRE     0x00000002
73    #define  CERF    0x00000001
74
75    #define  ITF     0x00000010
76    #define  CND     0x00000008
77    #define  DLC     0x00000004
78    #define  CD      0x00000002
79    #define  TRO     0x00000001
80
81    /* ==== Renesas Ethernet API return defines ==== */
82    #define R_ETHER_OK          0
83    #define R_ETHER_ERROR       -1
84    #define R_ETHER_HARD_ERROR  -3
85    #define R_ETHER_RECOVERABLE -4
86    #define R_ETHER_NO_DATA     -5
87
88    /* ==== Prototype Declaration ==== */
89    /* ==== Renesas Ethernet API prototypes ==== */
90    int32_t R_Ether_Open(uint32_t ch, uint8_t mac_addr[]);
91    int32_t R_Ether_Close(uint32_t ch);
92    int32_t R_Ether_Write(uint32_t ch, void *buf, uint32_t len);
93    int32_t R_Ether_Read(uint32_t ch, void *buf);
94
```

3.15　サンプルプログラムリスト "r_ether.h" (3)

```
95    /* ==== イーサネット収集データ ==== */
96    struct enet_stats
97    {
98      uint32_t rx_packets;
99      uint32_t tx_packets;
100     uint32_t rx_errors;
101     uint32_t tx_errors;
102     uint32_t rx_dropped;
103     uint32_t tx_dropped;
104     uint32_t multicast;
105     uint32_t collisions;
106
107     /* ---- 受信エラー ---- */
108     uint32_t rx_length_errors;
109     uint32_t rx_over_errors;
110     uint32_t rx_crc_errors;
111     uint32_t rx_frame_errors;
112     uint32_t rx_fifo_errors;
113     uint32_t rx_missed_errors;
114
115     /* ---- 送信エラー ---- */
116     uint32_t tx_aborted_errors;
117     uint32_t tx_carrier_errors;
118     uint32_t tx_fifo_errors;
119     uint32_t tx_heartbeat_errors;
120     uint32_t tx_window_errors;
121   };
122
123   struct ei_device
124   {
125     const int8_t *name;      /* デバイス名 */
126     uint8_t      open;
127     uint8_t      Tx_act;
128     uint8_t      Rx_act;
129     uint8_t      txing;
130     uint8_t      irqlock;
131     uint8_t      dmaing;
132     ethfifo      *rxcurrent;     /* 受信カレントディスクリプタ */
133     ethfifo      *txcurrent;     /* 送信カレントディスクリプタ */
134     uint8_t      save_irq;
135     struct enet_stats stat;      /* イーサネット収集データ */
136     uint8_t      mac_addr[6];    /* MACアドレスの格納領域 */
137   };
138
139   #endif /* ETH_H   */
```

4.　参考ドキュメント

- ソフトウェアマニュアル
  SH-2A, SH2A-FPU　ソフトウェアマニュアル Rev.3.00
  （最新版をルネサス エレクトロニクスのホームページから入手してください。）
- ハードウェアマニュアル
  SH7214 グループ、SH7216 グループ　ユーザーズマニュアル　ハードウェア編 Rev.2.00
  （最新版をルネサス エレクトロニクスのホームページから入手してください。）

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/inquiry

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2010.04.21	—	初版発行
2.00	2010.07.30	全頁	サンプルプログラムの API 変更に伴い、内容を修正。

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、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

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. リザーブアドレスのアクセス禁止

   【注意】リザーブアドレス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、事前に問題ないことをご確認下さい。

   同じグループのマイコンでも型名が違うと、内部メモリ、レイアウトパターンの相違などにより、特性が異なる場合があります。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。

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 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。
5. 輸出に際しては、「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。本資料に記載されている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の目的で使用しないでください。また、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に使用することができません。
6. 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。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されてい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生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   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

   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、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社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。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、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、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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。
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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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・電話番号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 エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

■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/inquiry

Colophon 1.0