お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

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

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

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。

SH7710 グループ
イーサネット送信設定例

要旨

この資料は、SH7710/7712/7713 のイーサネット送信設定例を示します。

動作確認デバイス

SH7712

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 10 フレーム送信します。1 フレームの送信が完了してから次の送信を開始します。
・ フレーム送信完了はフレーム送信完了割り込みにより判断します。

1.2 使用機能

・ 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)
・ 割り込みコントローラ(INTC)

1.3 適用条件

・ マイコン: SH7712 (HD6417712)
・ 動作周波数: 内部クロック 198.00MHz
バスクロック 66.00MHz
周辺クロック 33.00MHz
・ 統合開発環境: ルネサステクノロジ製 High-performance Embedded Workshop Ver.4.03.00.001
・ C コンパイラ: ルネサステクノロジ製
SuperH RISC engine ファミリ C/C++コンパイラパッケージ V.9.01 release01
・ コンパイルオプション: High-performance Embedded Workshop でのデフォルト設定
(-cpu=sh3dsp -object="$(CONFIGDIR)¥$(FILELEAF).obj" -debug -gbr=auto
-chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0
-del_vacant_loop=0 -struct_alloc=1 -nologo)

1.4 関連アプリケーションノート

本資料の参考プログラムは、SH7710/7712/7713 初期設定例アプリケーションノートの設定条件で動作確認をしています。そちらも合わせてご参照ください。
また、以下のアプリケーションノートもご参照ください。
「SH7710 グループ アプリケーションノート イーサネット PHY-LSI 自動交渉設定例」
「SH7710 グループ アプリケーションノート イーサネット受信設定例」

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)の 0 系、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)の 0 系を使用します。

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 は送受信制御および MAC 間転送を行います。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 転送を専用に行います。なお、SH7713 の EtherC は MAC-0 のみで転送機能はありません。

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。EtherCはMAC層インタフェースを 2 系統(0 系、1 系)内蔵しており、それぞれ独立に送受信させることができます。また、EtherCは転送処理を制御するTSU(Transfer Switching Unit)を内蔵し、MACコントローラ間で相互にデータ転送を行うことが可能です。このTSUは、両MACコントローラに入力されたフレームの受信や転送を判定するために、32 エントリの CAM(Content Addressable Memory)エントリテーブルおよび2 本の外部CAMインタフェース入力端子を有しています。さらに、転送するフレームを保持するトータル 6kバイトの転送FIFOを内蔵しており、0 系→1 系および 1 系→0 系の各転送条件に対し転送FIFO容量の割り当てを自由に設定することができます。EtherCは 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。

なお、本初級編ではMAC間転送機能を使わないため、TSUの詳細説明は省略します。

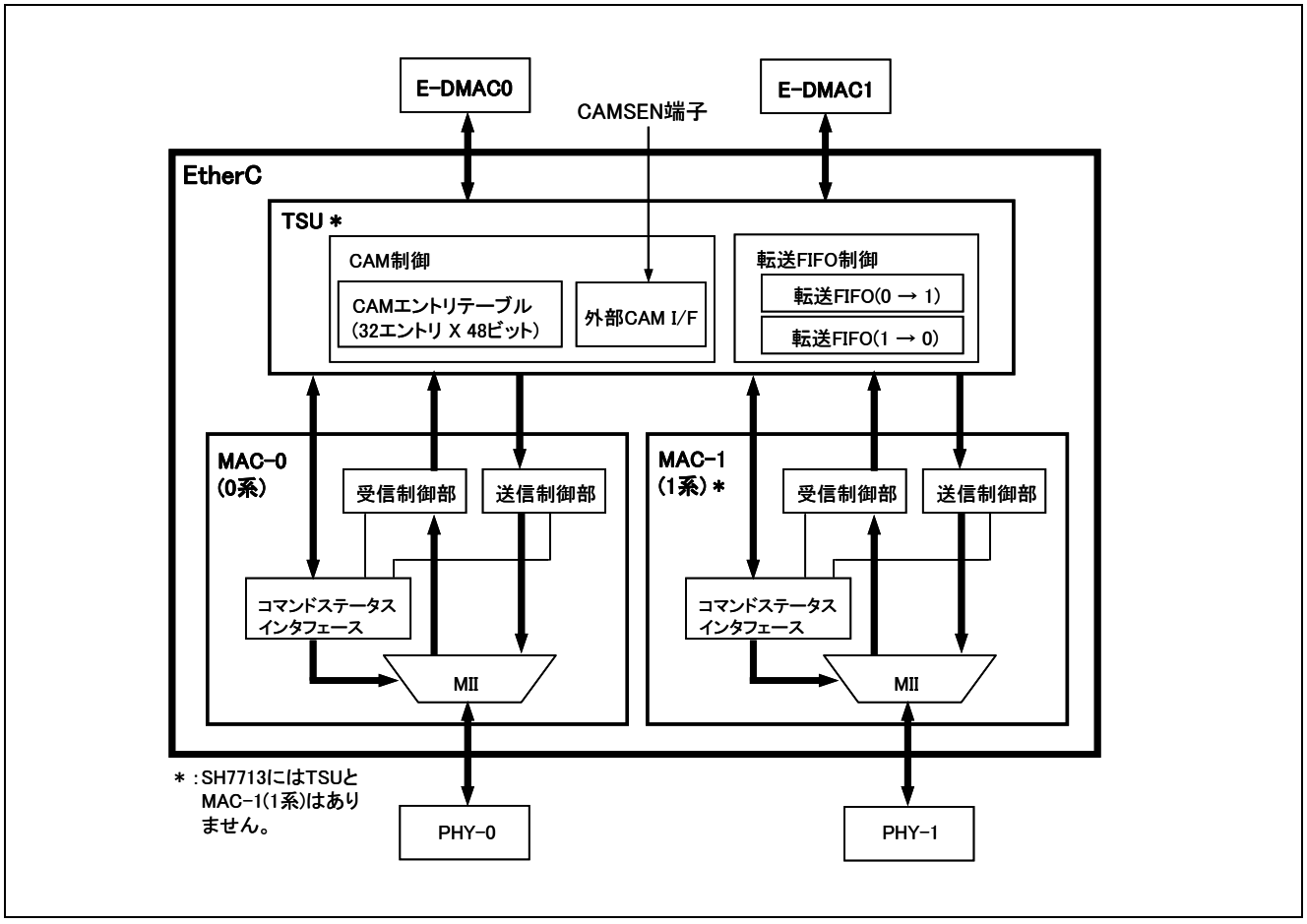


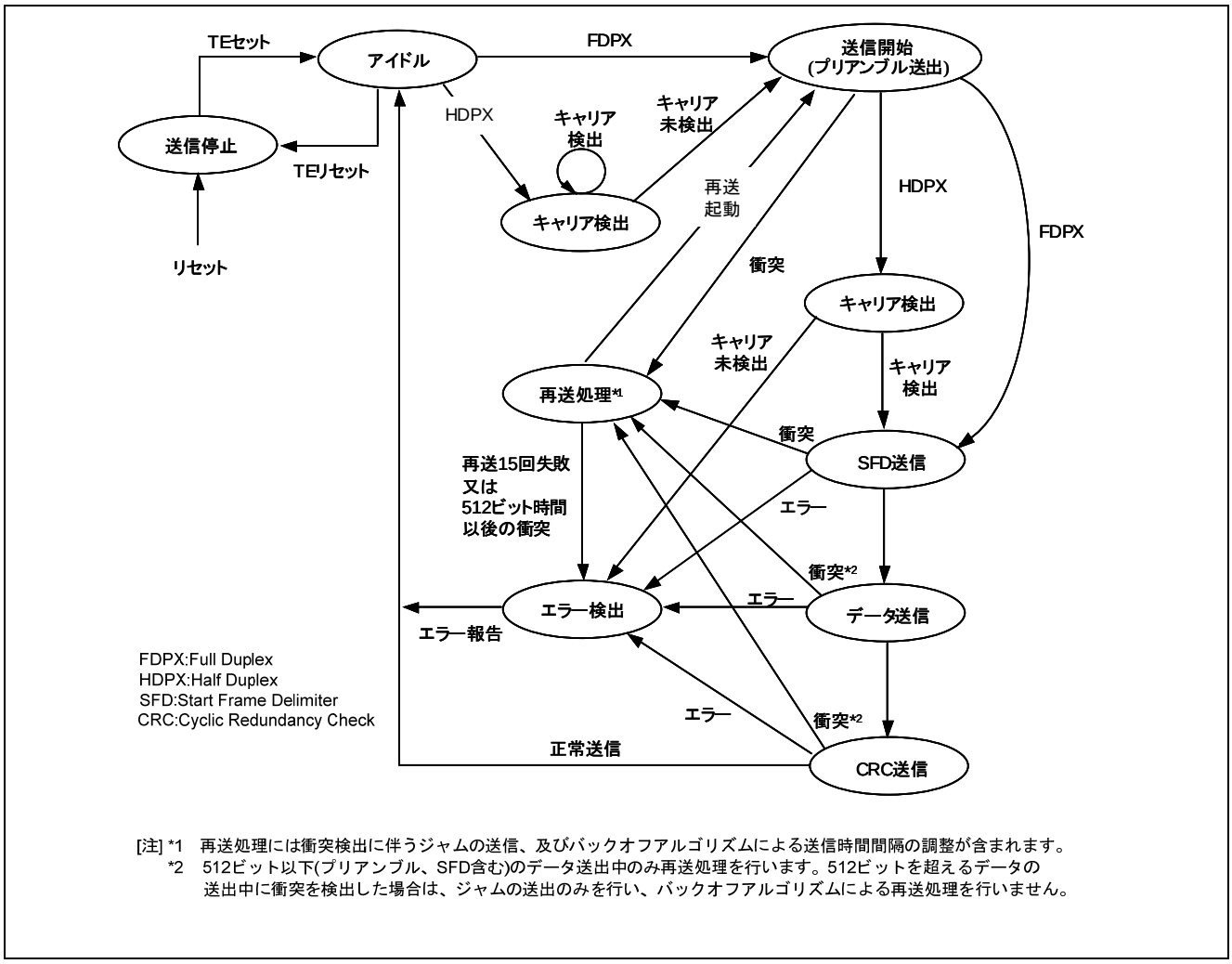
図1 EtherC の構成

2.1.2 EtherC 送信部の概要

EtherC送信部は、E-DMAC送信部から送信要求があると送信データをフレームに組み立てて MII(Media Independent Interface)に出力します。MIIを経由した送信データは、PHY-LSIによって回線上に送出されます。図2にEtherC送信部の状態遷移図を示します。この動作は0系および1系で共通です。送信動作のフローは以下のようになります。

1. EtherC は、送信許可ビット(EtherC モードレジスタ(ECMR)の TE ビット)がセットされると送信アイドル状態に遷移します。
2. (A)半二重転送方式(HDPX)時
   E-DMAC 送信部から送信要求があると EtherC はキャリア検出を行い、未検出であればフレーム間隔時間の送信延期を経てプリアンブルを MII に送出します。キャリアを検出した場合は、キャリアがなくなってからフレーム間隔時間の送信延期を経てプリアンブルを MII に送出します。

   (B)全二重転送方式(FDPX)時
   キャリア検出を必要とせず、E-DMAC 送信部から送信要求があると即座にプリアンブルを送出します。ただし連続送信時は、直前に送信したフレームから必ずフレーム間隔時間の送信延期を経てプリアンブルを送出します。
3. SFD(Start Frame Delimiter)、データ、CRC(Cyclic Redundancy Check)を順次送信します。送信を終了するとフレーム送信完了割り込み(TC)が発生します。データ送信中に衝突あるいはキャリア未検出状態になるとそれぞれの割り込みが発生します。
4. アイドル状態に遷移し、以後送信データがあれば送信を継続します。



[注] *1 再送処理には衝突検出に伴うジャムの送信、及びバックオフアルゴリズムによる送信時間間隔の調整が含まれます。
*2 512ビット以下(プリアンブル、SFD含む)のデータ送出中のみ再送処理を行います。512ビットを超えるデータの送出中に衝突を検出した場合は、ジャムの送出のみを行い、バックオフアルゴリズムによる再送処理を行いません。

図2 EtherC 送信部状態遷移図

2.1.3 E-DMAC の概要

本LSIは、EtherCに直結した 2 系統のダイレクトメモリアクセスコントローラ(E-DMAC0/1)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(次章で詳述)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込み、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込み、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)化することで、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。E-DMAC0 はEtherCのMAC-0 に対して、E-DMAC1 はEtherCのMAC-1 に対してデータ送受信を制御します。

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の特長は以下のようになります。

特長

・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 参照)

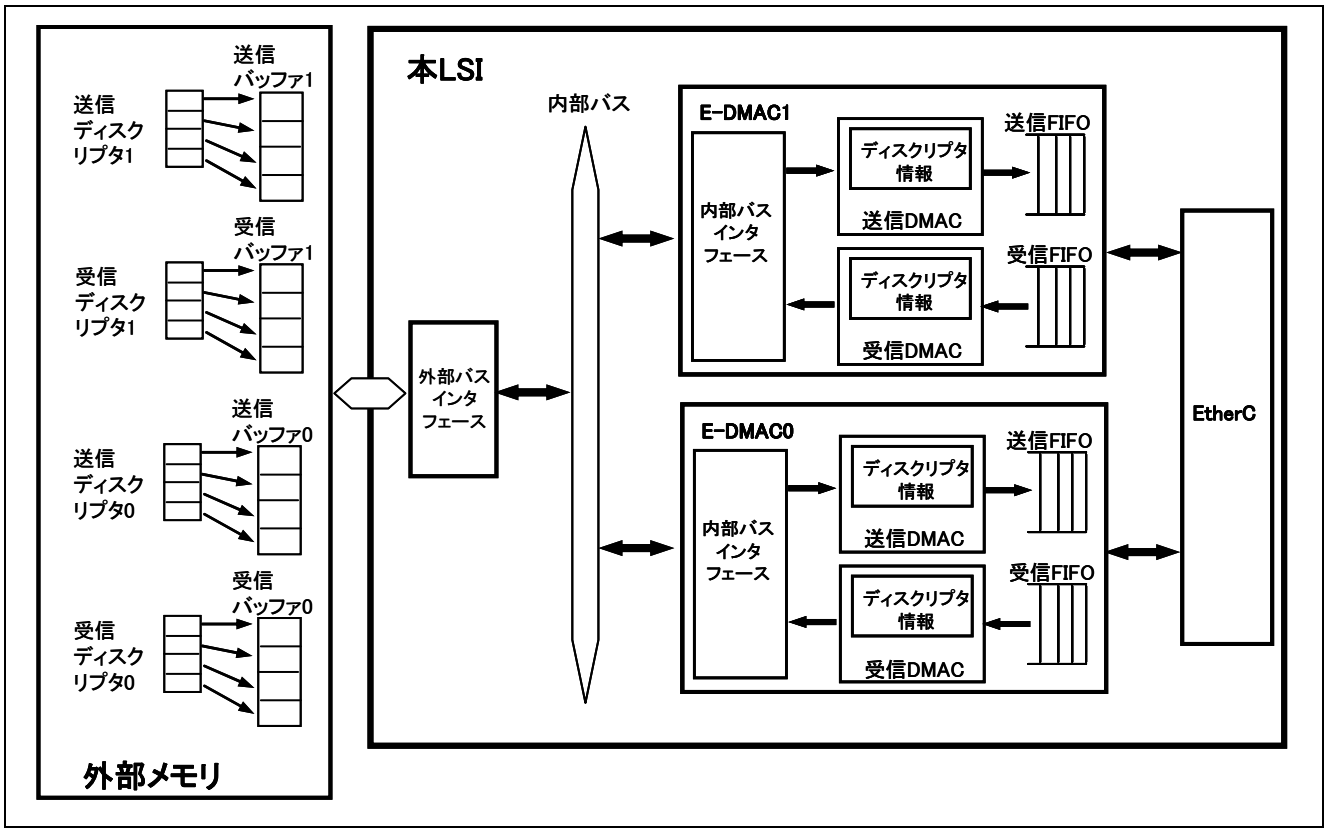


図3　E-DMAC とディスクリプタおよびバッファの構成

2.1.4 ディスクリプタの概要

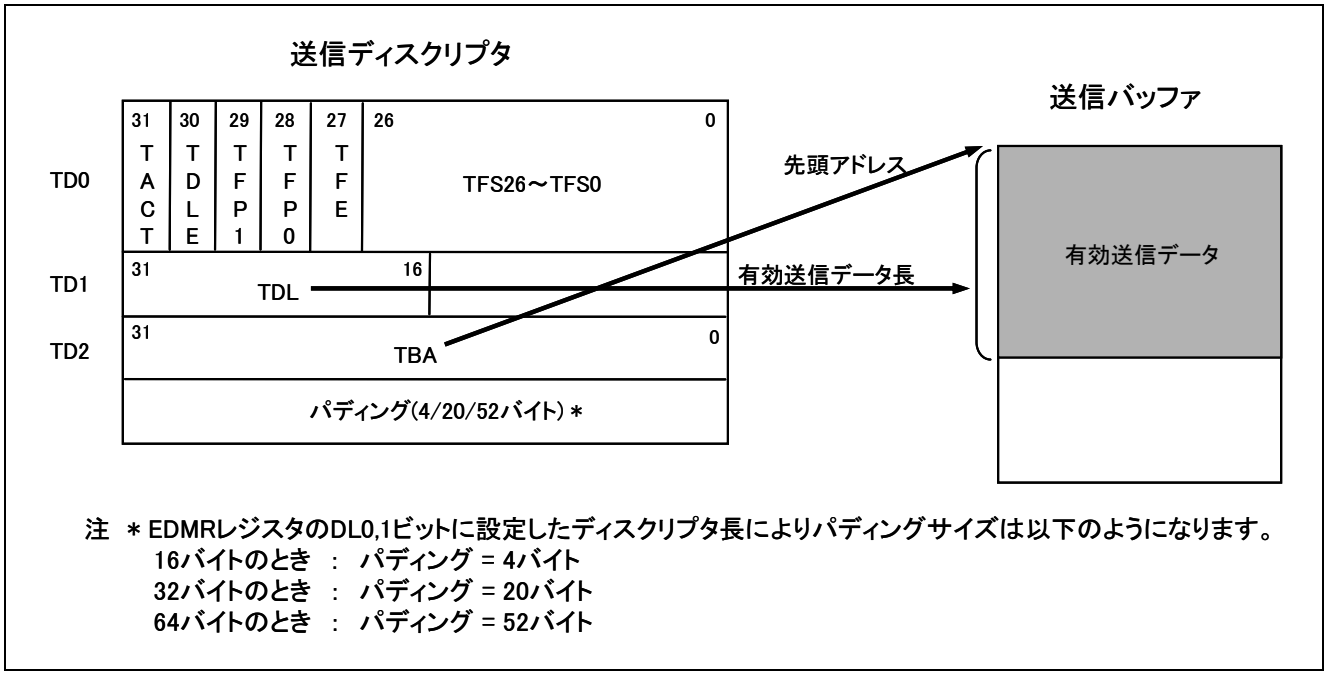
E-DMAC が DMA 転送を行うためには、ディスクリプタと呼ばれる送受信データの格納アドレス等が書かれた情報(データ)が必要になります。ディスクリプタには送信ディスクリプタと受信ディスクリプタの 2 種類があります。E-DMAC は、E-DMAC 送信要求レジスタ(EDTRR)の TR ビットが 1 になると自動的に送信ディスクリプタの読み込みを、E-DMAC 受信要求レジスタ(EDRRR)の RR ビットが 1 になると自動的に受信ディスクリプタの読み込みを開始します。ユーザは送信/受信ディスクリプタにあらかじめ送信/受信データの DMA 転送に関する情報を記述しておく必要があります。イ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 TFS26～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 が最初に読み込むディスクリプタ)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 送信ディスクリプタの概要

図 4に送信ディスクリプタと送信バッファの関係を示します。
送信ディスクリプタは、データの先頭から 32 ビット単位にTD0,TD1,TD2 およびパディングで構成されます。TD0 は、送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。TD1 はそのディスクリプタで指示する転送すべき送信バッファのデータ長を示します。TD2 は転送する送信バッファの先頭アドレスを示します。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。

送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 フレームの送信データを指定すること(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 フレームの送信データを指定すること(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/マルチバッファとしては、たとえばイーサネットフレーム中毎回の送信で固定的に使われるデータ部分を複数のディスクリプタに設定するという方法があります。具体的には、イーサネットフレーム中のあて先アドレス、送信元アドレスのデータを複数のディスクリプタで共有して、残りのデータを各々のバッファに格納するという方法が考えられます。



注 ＊EDMRレジスタのDL0,1ビットに設定したディスクリプタ長によりパディングサイズは以下のようになります。
16バイトのとき ： パディング = 4バイト
32バイトのとき ： パディング = 20バイト
64バイトのとき ： パディング = 52バイト

図4 送信ディスクリプタと送信バッファの関係

2.1.6 送信ディスクリプタの設定例

図 5に送信ディスクリプタおよび送信バッファを 3 面使用した場合の例(1 フレーム/1 ディスクリプタ)を示します。ここでは 1 回の送信要求で 1 フレームだけ送信するものとします。図では各送信ディスクリプタをTD0部分のみに簡略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。

設定は以下のようになります。

1. フレーム/1 ディスクリプタ方式のため、全ディスクリプタ面の TFP1,TFP0 ビットに B’11 を設定します。
2. 全ディスクリプタ面の TACT ビット、TFE ビット、TFS26～TFS0 ビットには初期値としてすべて 0 を設定します。
3. 第 1 面と第 2 面のディスクリプタの TDLE ビットに 0 を設定します。第 3 面のディスクリプタの TDLE ビットに 1 を設定することにより、第 3 面の処理を終了すると第 1 面のディスクリプタを読み込みます。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。
4. 図 5では省略していますが、当該ディスクリプタが参照している送信バッファのデータ長をTDLに、送信バッファの先頭アドレスをTBAに設定します。
5. この例では 1 回の送信要求で 1 フレームだけ送信するため、最初の送信では第 1 面のディスクリプタの TACT ビットにだけ 1 を設定します。次の送信では第 2 面のディスクリプタの TACT ビットにだけ 1 を設定します。送信手順の詳細は次章で説明します。

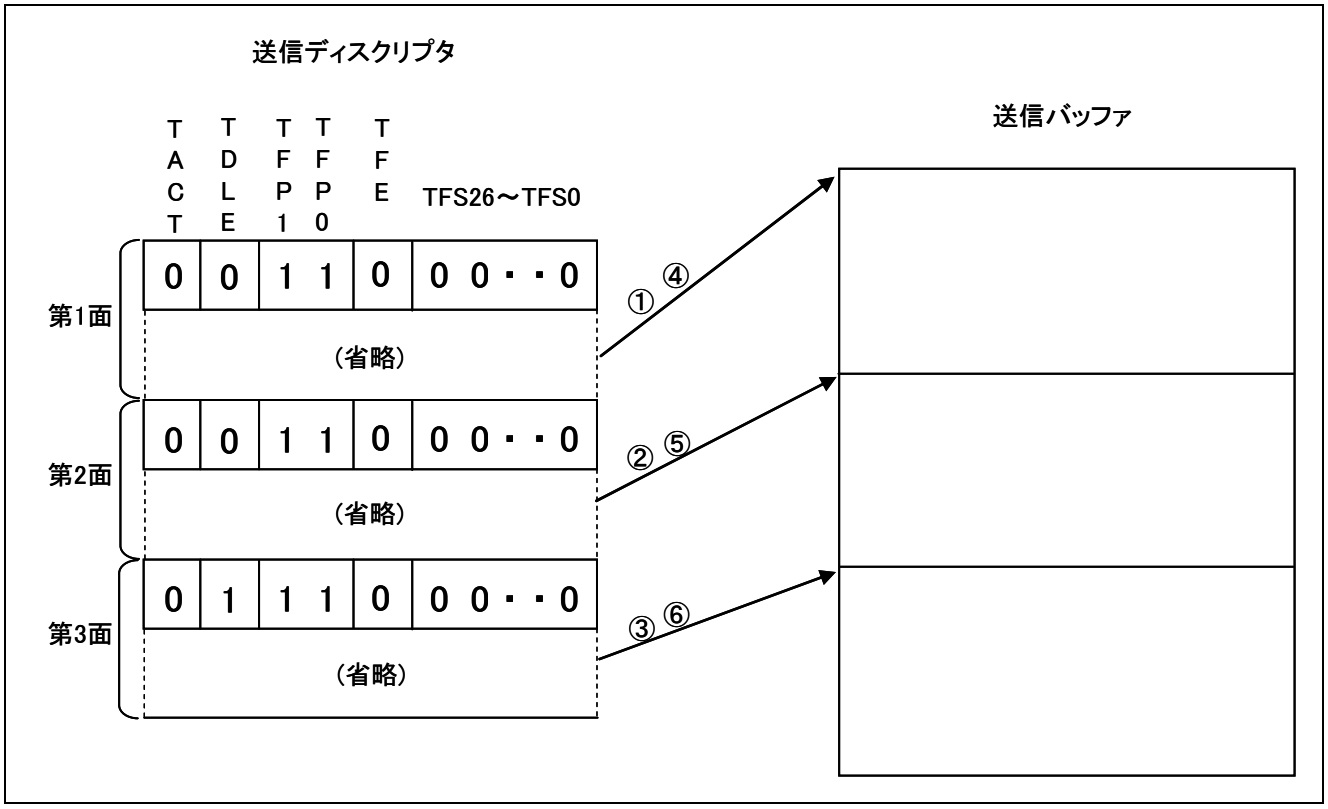


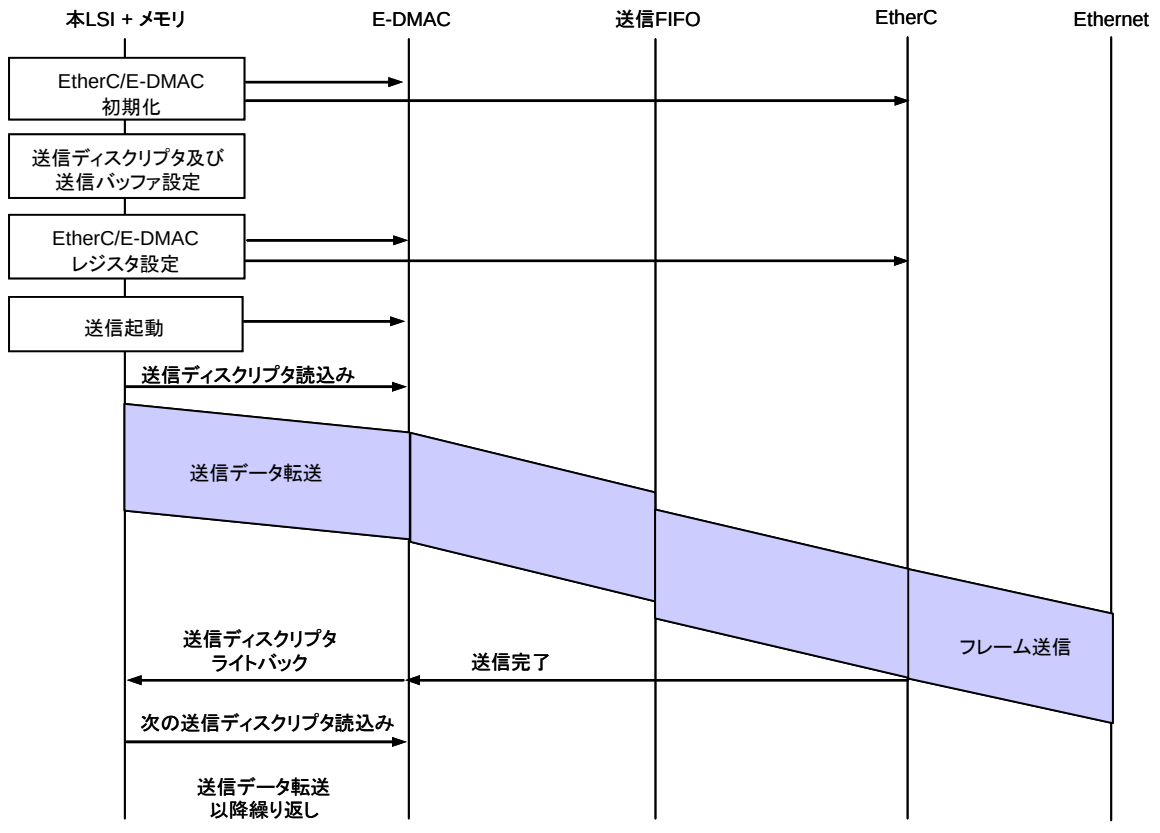
図5　送信ディスクリプタと送信バッファの関係

2.1.7 使用機能の動作手順(送信時)

EtherC モードレジスタ(ECMR)の TE ビットが 1 の状態で E-DMAC 送信要求レジスタ(EDTRR)の送信要求ビット(TR)に 1 を書き込むと E-DMAC 送信部が起動します。E-DMAC は EtherC/E-DMAC のソフトウエアリセット後、送信ディスクリプタ先頭アドレスレジスタ(TDLAR)で示すディスクリプタを読み込みます。読み込んだディスクリプタの TACT ビットが 1(有効)の場合は、E-DMAC は送信ディスクリプタの TD2 で指定される送信バッファ先頭アドレスから順次送信フレームデータを読み出して EtherC に転送します。EtherC は送信フレームを作成し MII に向けて送信を開始します。ディスクリプタ内で指示されるバッファ長分の DMA 転送後、送信ディスクリプタの TFP の値によって以下の処理を行います。

- TFP=B'00 or B'10(フレーム継続)
  DMA 転送後、ディスクリプタのライトバック(TACT ビットの 0 書き込み)を行います。その後、次のディスクリプタの TACT ビットを読み込みます。
- TFP=B'01 or B'11(フレーム終了)
  フレームの送信完了後、ディスクリプタのライトバック(TACT ビットの 0 およびステータスの書き込み)を行います。その後、次のディスクリプタの TACT ビットを読み込みます。

読み込んだディスクリプタのTACTビットが 1 のときは、フレームの送信を継続し次のディスクリプタを読み込みます。TACTビットが 0(無効)のディスクリプタを読み込むと、E-DMACはEDTRRのTRビットを 0 にして送信処理を完了します。TRビットが 0 になった後TRビットに 1 を書き込むと再度E-DMAC送信部が起動しますが、この場合は最後に送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込みます。
図 6に送信フローの例(1 フレーム/1 ディスクリプタ、複数ディスクリプタ面の場合)を示します。



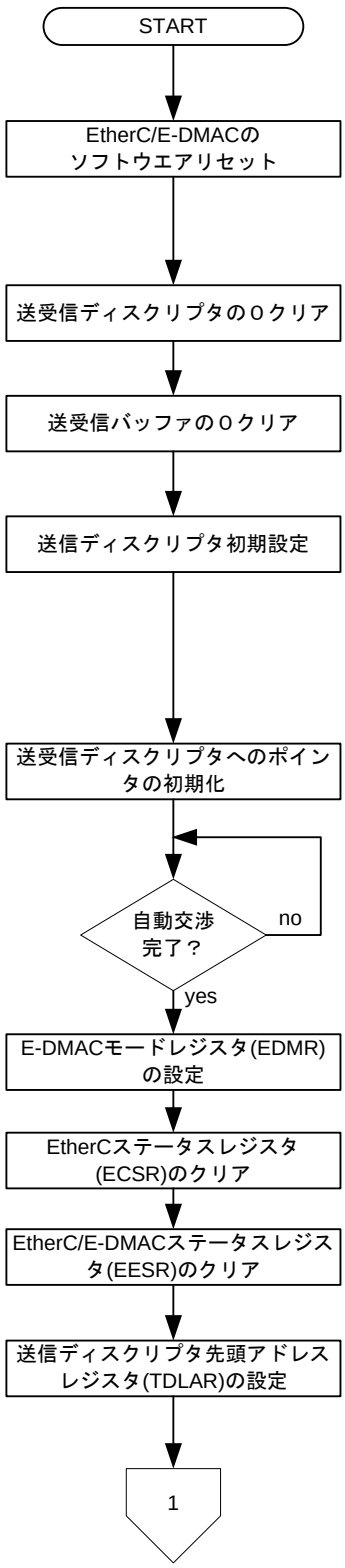
[補足説明]

EtherC/E-DMAC 初期化	: EDMR の SWR ビット 1 によるソフトウエアリセットを実行します。
EtherC/E-DMAC レジスタ設定	: 最後に ECMR の TE ビット 1、EDTRR の TR ビット 1 の書き込みを行います。
送信起動	: ECMR の TE ビット、EDTRR の TR ビット 1 の書き込みにより発生します。
送信ディスクリプタ読み込み	: E-DMAC が自動的に読み込みます。
送信データ転送	: E-DMAC の DMA 転送により送信データを送信 FIFO に書き込みます。
送信ディスクリプタライトバック	: E-DMAC が TACT ビットに 0 及び送信ステータスを送信ディスクリプタに書き込みます。

図6　送信フローの例(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(送信時)

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



・EtherC/EDMACのソフトウエアリセットを実施
SWRビットに1をライトします。
チャネル(0または1)ごとに設定が必要です。
ソフトウエアリセット発行期間中(内部バスクロックBφで
64サイクル間)は、リセットしたチャネルでのイーサネット関係の
すべてのモジュールに対するレジスタアクセスを禁止してください。

・メモリ上の送受信用ディスクリプタ領域をクリアします。

・メモリ上の送受信用バッファ領域をクリアします。

・送信ディスクリプタリスト全体の初期化
TD0: TACT: この時点ではなく送信起動の直前で設定します。
TFP1,TFP0: 1フレーム/1ディスクリプタでは11を設定します。
TDLE:最終面に1を設定します(その他の面では0を設定します)。
TD2: TBA: 各ディスクリプタに対する送信バッファの先頭アドレスを設定します。
送信バッファは16バイト境界に配置します。
パディング領域: E-DMACはこの領域を使用しません。ユーザが自由に設定できます。

・送信ディスクリプタリストの先頭アドレスを初期値として設定します。
・受信ディスクリプタリストの先頭アドレスを初期値として設定します。

・PHY-LSIが自動交渉した結果から全二重または半二重の転送方式を
取得します。
・アプリケーションノート「イーサネットPHY-LSI自動交渉設定例」をご参照
ください。

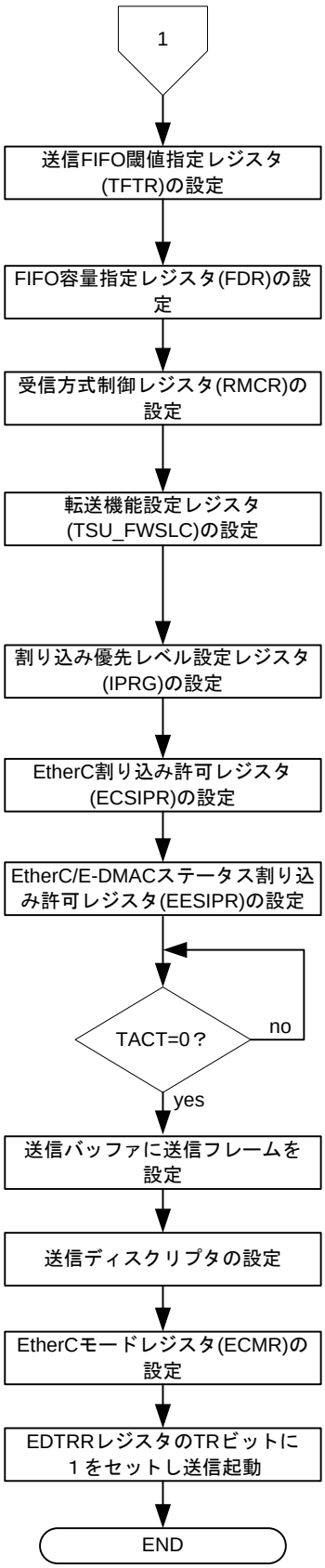
・送受信ディスクリプタ長を設定します。

・各ビットを1ライトによりクリアします。

・各ビットを1ライトによりクリアします。

・送信ディスクリプタリストの先頭アドレスを設定
指定したディスクリプタ長により下位ビットを以下のように設定します。
16バイトバウンダリ: TDLA[3:0] = 0000
32バイトバウンダリ: TDLA[4:0] = 00000
64バイトバウンダリ: TDLA[5:0] = 000000
・実際のメモリ領域もこの境界に配置してください。

図7　イーサネット送信設定フロー例(1)



・最初の送信を開始するまでの送信FIFOのしきい値を設定
　ストア&フォワードモードでは0x00000000を設定します。
・閾値を小さくするとスループットは向上しますが、アンダフローの発生に注意が必要です。

・送信FIFOおよび受信FIFOの容量を設定
　最大の2KB選択時は0x00000707を設定します。

・フレーム受信を継続するかしないかを設定
　1つのフレーム受信後、連続して受信したい場合は1を設定します。
　1つのフレーム受信後、受信を継続しない場合は0を設定します。

・E-DMAC0に対してはビット15～12の値を設定
・E-DMAC1に対してはビット11～8の値を設定
・設定0x0は優先順位レベル0(要求マスク)を、0xFは優先順位レベル15(最高レベル)を示します。

・リンク信号変化、Magic Packet検出、不正キャリア検出の割り込み許可を設定

・EtherC/E-DMACステータスレジスタ(EESR)の各ビットに対応する割り込み許可レジスタ
・ビットごとに1を設定するとその割り込みが許可されます。

・当該送信ディスクリプタ、送信バッファを次の送信に使えるかをTACTビットを確認することで判断します。

・TDLに送信データ長を設定
・TACTビットに1を設定

・全二重/半二重モードの選択：PHY-LSIの自動交渉で得た結果を反映します。
・送信許可:TEビットに1を設定

・送信起動
・注意事項：マニュアル19.4　使用上の注意事項　ご参照
　TRビットをリードし0の場合に1をライトしてください。

図8　イーサネット送信設定フロー例(2)

2.2 参考プログラムの動作

参考プログラムでは、EtherC の 0 系(MAC-0)及び E-DMAC の 0 系(E-DMAC0)を使用し、対向ホストに向けて 10 フレーム送信します。送信ディスクリプタと 1520 バイトの送信バッファを 4 面用意(1 フレーム/1 ディスクリプタ)しています。送信ディスクリプタをリング状にして使用しています。フレーム送信完了割り込み(TC)により 1 フレームの送信が完了したと判断し、次の送信を開始します。ただし、下記のルネサステクニカルアップデート TN-SH7-A575B/J 及び TN-SH7-A583A/J に従い、送信ディスクリプタの TACT ビットのポーリング及びタイムアウト処理を付加しています。

送信データについては、イーサネットフレームのうちプリアンブル、スタートフレームデリミタ(SFD)、および CRC 部を除いた部分を用意する必要があります。ヘッダ部の宛先 MAC アドレス及び送信元 MAC アドレスは、ご使用になる製品の MAC アドレスに変更していただく必要があります。なお、EtherC は送信元 MAC アドレスのチェックは行いません。

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。

テクニカルアップデート

発行番号	題名
TN-SH7-A575B/J	SH-Ether EtherC/E-DMAC ステータスレジスタ(EESR)に関する使用上の注意について(2)
TN-SH7-A583A/J	SH-Ether 送信アンダフロー発生時の使用上の注意事項について

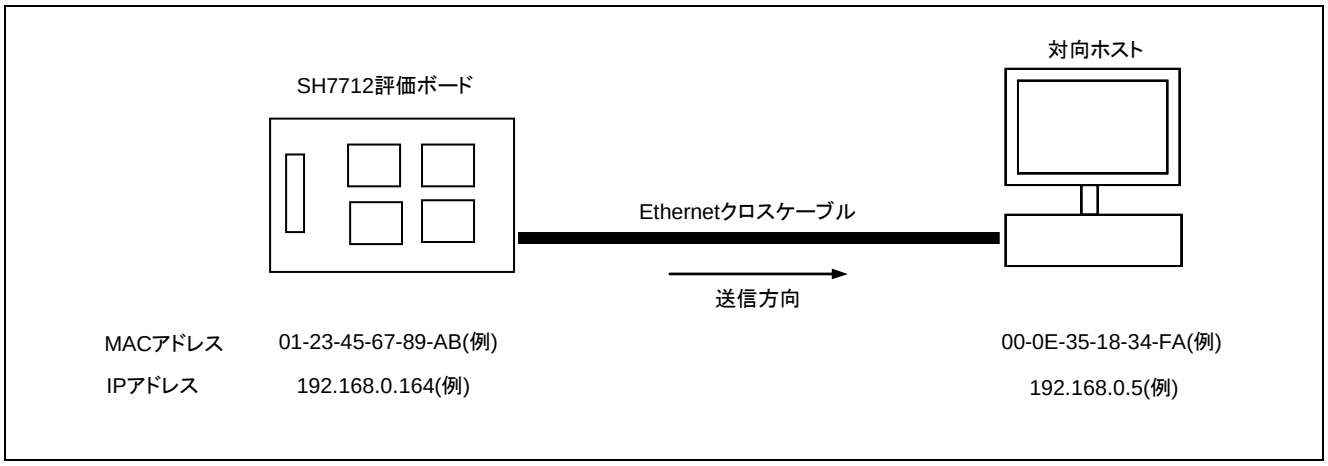


図9　参考プログラムの動作環境

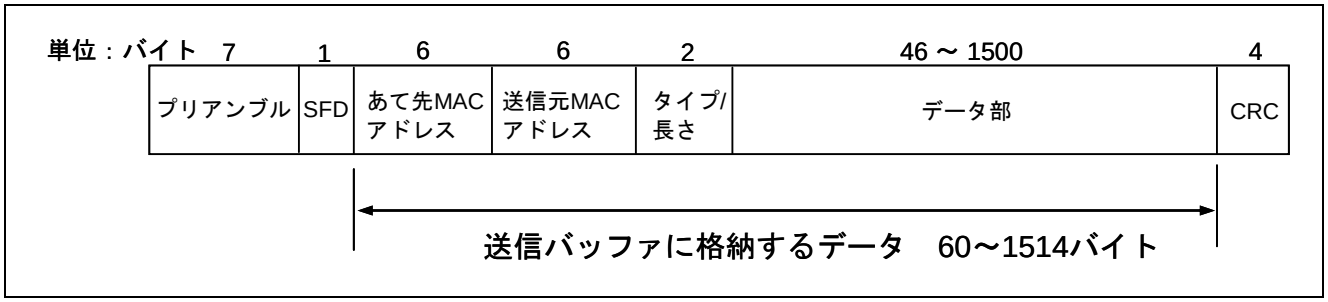


図10　イーサネットフレームフォーマット

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、ソフトウエ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送信ディスクリプタ構造体の定義と送信ディスクリプタ列の使用例を示します。

送信ディスクリプタの構造体定義

```
typedef struct tag_edmac_send_desc
{
    TD0    td0;
    TD1    td1;
    TD2    td2;
    struct tag_edmac_send_desc   *pNext;
}EDMAC_SEND_DESC;
```

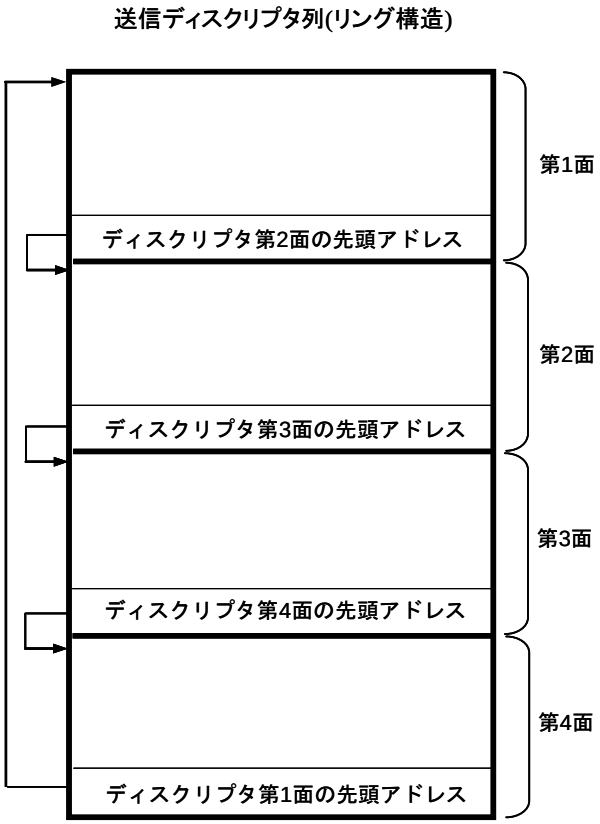


図11　送信ディスクリプタの構造体定義と送信ディスクリプタ列使用例

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。なお、EtherC/E-DMACの各種レジスタおよびディスクリプタの初期設定では受信の設定も行っていますが、受信処理は行っていません。

PHY自動交渉関数phy_autonegoの詳細は「SH7710 グループ　アプリケーションノート　イーサネットPHY-LSI自動交渉設定例」をご参照ください。

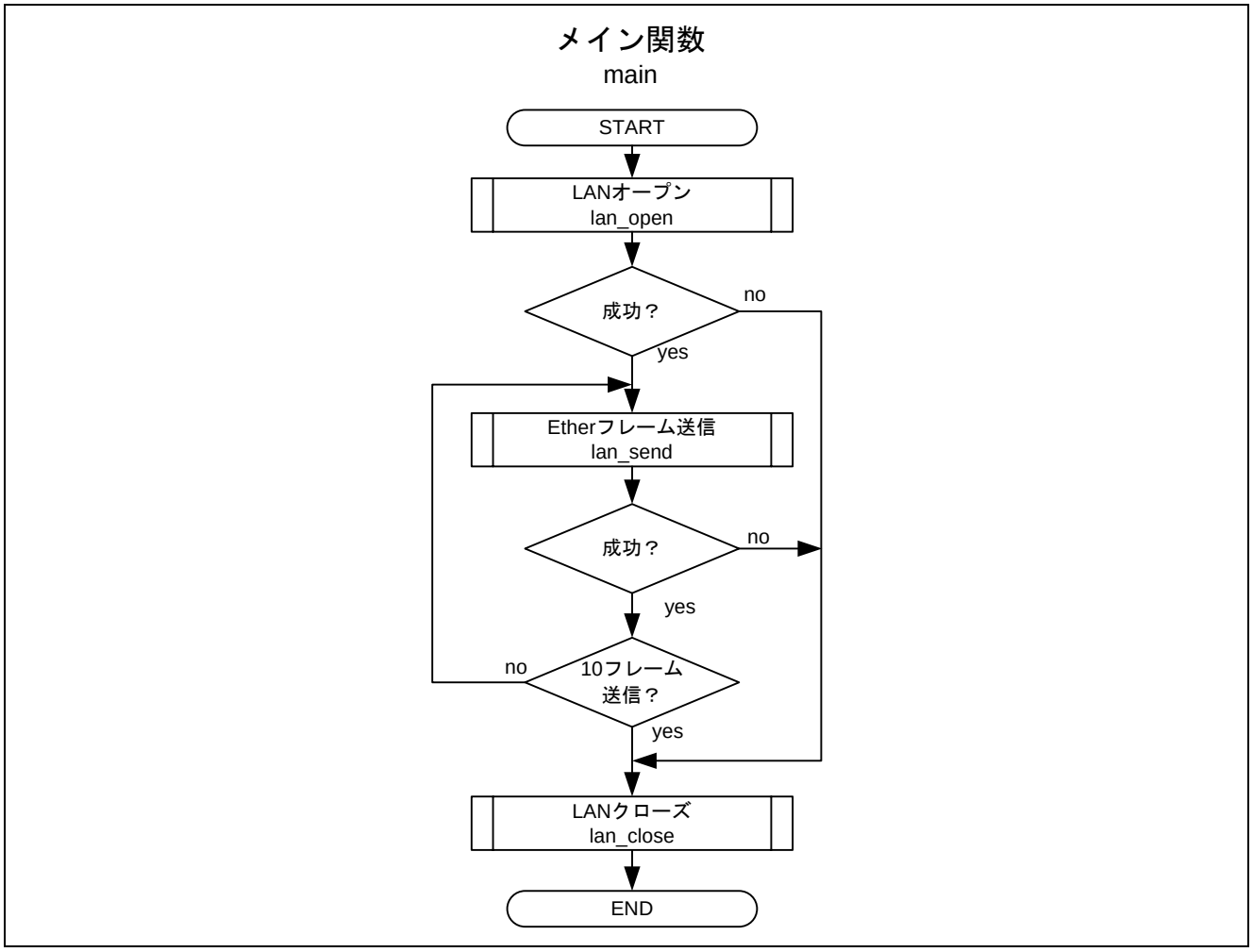


図12　メインプログラムの処理フロー例(1)

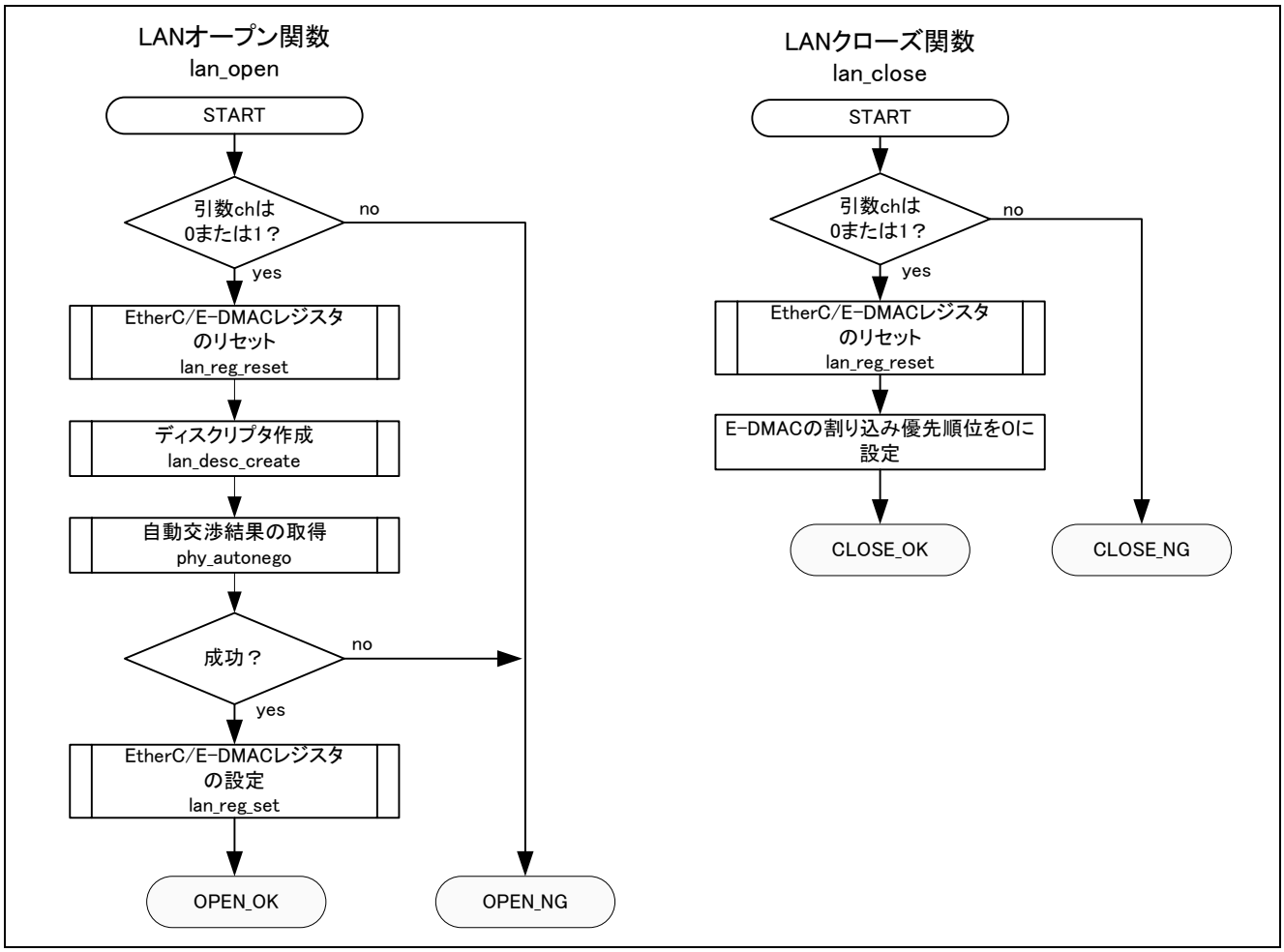


図13　参考プログラムの処理フロー例(2)

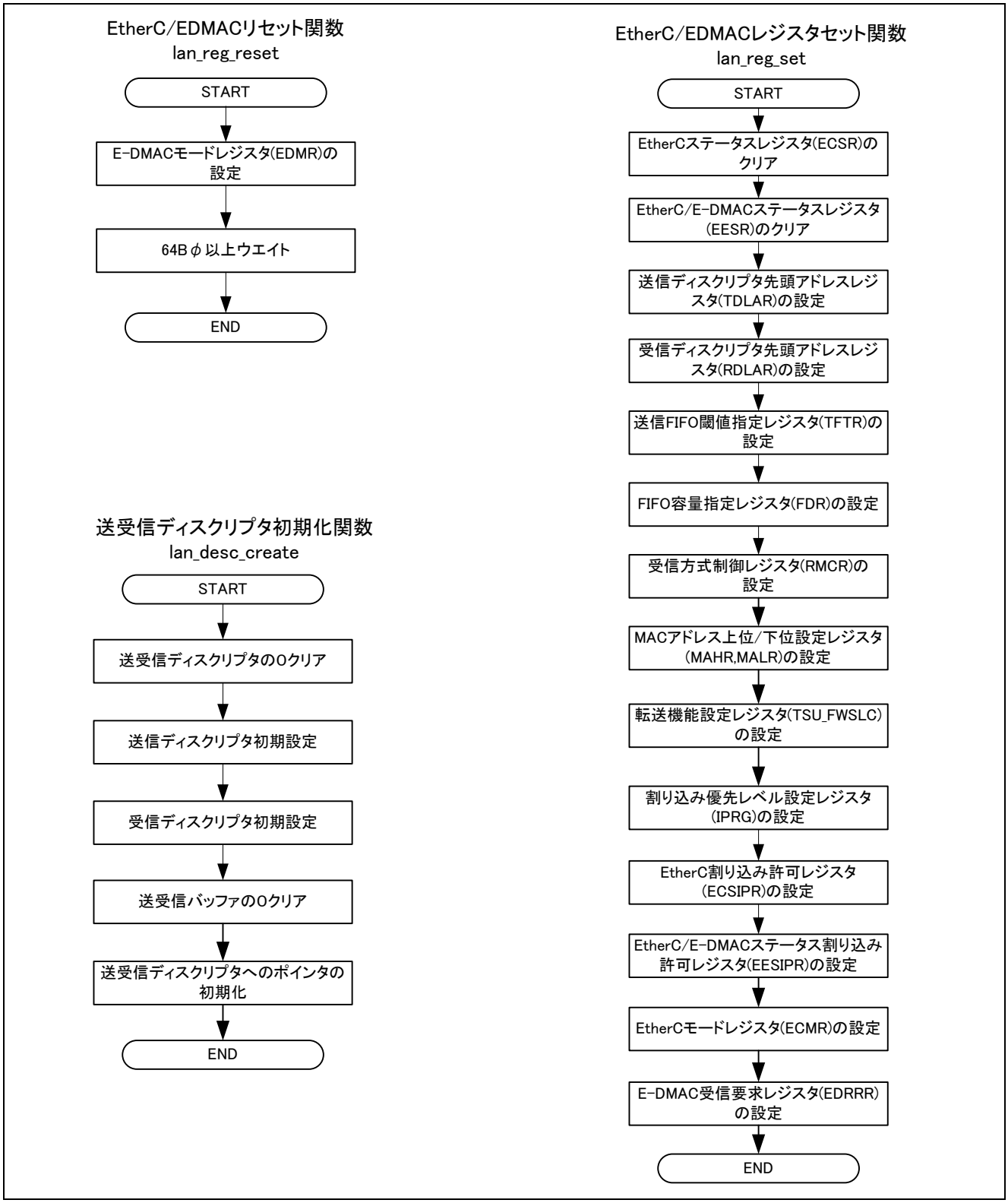
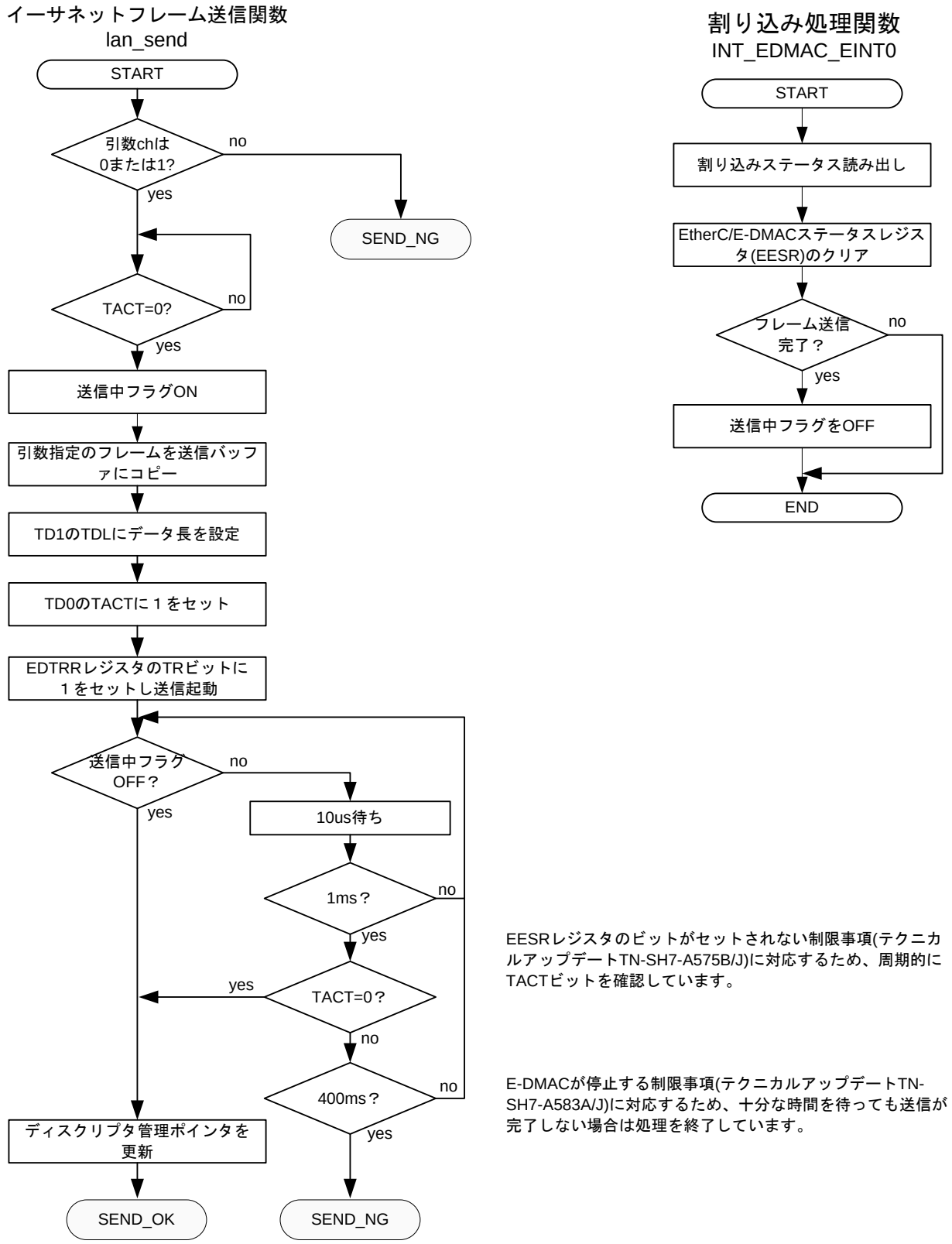


図14　参考プログラムの処理フロー例(3)



EESRレジスタのビットがセットされない制限事項(テクニカルアップデートTN-SH7-A575B/J)に対応するため、周期的にTACTビットを確認しています。

E-DMACが停止する制限事項(テクニカルアップデートTN-SH7-A583A/J)に対応するため、十分な時間を待っても送信が完了しない場合は処理を終了しています。

注)ch0とch1ではEtherC/E-DMACレジスタのチャネル番号、及びディスクリプタのチャネル番号が異なるだけなので、図では共通化しています。

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)

```
1   /*""FILE COMMENT""*********************************************************************
2   *
3   *   System Name  : SH7712 Sample Program
4   *   FILE Name    : main.c
5   *   Version      : 1.00.00
6   *   Contents     : Ethernet 送信処理
7   *   Model        : MS7712SE01
8   *   CPU          : SH7712
9   *   Compiler     : SHC9.1.1.0
10  *   OS           : None
11  *
12  *   note         : Ethernet 送信の参考プログラムです。
13  *                  <注意事項>
14  *                  本サンプルプログラムはすべて参考資料であり
15  *                  その動作を保証するものではありません。
16  *                  本サンプルプログラムはお客様のソフトウエア開発時の
17  *                  技術参考資料としてご利用ください。
18  *
19  *   Copyright  (C)  2007 Renesas Technology Corp. All Rights Reserved
20  *   AND Renesas Solutions Corp. All Rights Reserved
21  *
22  *   history      : 2007.11.07 ver 1.00.00
23  *""FILE COMMENT END""*****************************************************************/
24  #include <machine.h>
25  #include "iodefine.h"
26  #include "ether.h"
27
28  /* ==== プロトタイプ宣言 ==== */
29  void main(void);
30
31  /* ==== グローバル変数宣言 ==== */
32  /* 送信用データの準備 */
33  /* 送信元 MAC アドレス：01-23-45-67-89-AB,宛先 MAC アドレス：00-0E-35-18-34-FA */
34  /* 送信元 IP アドレス：192.168.0.164,宛先 IP アドレス：192.168.0.5 */
35  static unsigned char frame[] =
36  {0x00,0x0E,0x35,0x18,0x34,0xFA,0x01,0x23,0x45,0x67,0x89,0xAB,0x08,0x00,/* MAC ヘッダ */
37  0x45,0x00,0x00,0x5C,0x1D,0xF1,0x00,0x00,0xFF,0x11,0x14,0x16,0xC0,0xA8,0x00,0xA4,
38  0xC0,0xA8,0x00,0x05, /* IP ヘッダ */
39  0x02,0x00,0x00,0x80,0x00,0x48,0x1A,0xB0, /* UDP ヘッダ */
40  0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0A,0x0B,0x0C,0x0D,0x0E,0x0F,
41  0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1A,0x1B,0x1C,0x1D,0x1E,0x1F,
42  0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2A,0x2B,0x2C,0x2D,0x2E,0x2F,
43  0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39,0x3A,0x3B,0x3C,0x3D,0x3E,0x3F,
44  0x40,0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,0x4A,0x4B,0x4C,0x4D,0x4E,0x4F,
45  0x50,0x51,0x52,0x53,0x54,0x55,0x56,0x57,0x58,0x59,0x5A,0x5B,0x5C,0x5D,0x5E,0x5F,
46  0x60,0x61,0x62,0x63,0x64,0x65,0x66,0x67,0x68,0x69,0x6A,0x6B,0x6C,0x6D,0x6E,0x6F,
47  0x70,0x71,0x72,0x73,0x74,0x75,0x76,0x77,0x78,0x79,0x7A,0x7B,0x7C,0x7D,0x7E,0x7F,
48  0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,0x88,0x89,0x8A,0x8B,0x8C,0x8D,0x8E,0x8F,
49  0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,0x98,0x99,0x9A,0x9B,0x9C,0x9D,0x9E,0x9F,
50  0xA0,0xA1,0xA2,0xA3,0xA4,0xA5,0xA6,0xA7,0xA8,0xA9,0xAA,0xAB,0xAC,0xAD,0xAE,0xAF,
51  0xB0,0xB1,0xB2,0xB3,0xB4,0xB5,0xB6,0xB7,0xB8,0xB9,0xBA,0xBB,0xBC,0xBD,0xBE,0xBF,
```

3.2 サンプルプログラムリスト”main.c” (2)

```
52  0xC0,0xC1,0xC2,0xC3,0xC4,0xC5,0xC6,0xC7,0xC8,0xC9,0xCA,0xCB,0xCC,0xCD,0xCE,0xCF,
53  0xD0,0xD1,0xD2,0xD3,0xD4,0xD5,0xD6,0xD7,0xD8,0xD9,0xDA,0xDB,0xDC,0xDD,0xDE,0xDF,
54  0xE0,0xE1,0xE2,0xE3,0xE4,0xE5,0xE6,0xE7,0xE8,0xE9,0xEA,0xEB,0xEC,0xED,0xEE,0xEF,
55  0xF0,0xF1,0xF2,0xF3,0xF4,0xF5,0xF6,0xF7,0xF8,0xF9,0xFA,0xFB,0xFC,0xFD,0xFE,0xFF};
56
57  /*""FUNC COMMENT""*****************************************************************
58  * ID           :
59  * モジュール概要  : 送信設定例メイン
60  *-------------------------------------------------------------------------------
61  * Include      : #include  "iodefine.h"
62  *-------------------------------------------------------------------------------
63  * 宣言          : void main(void)
64  *-------------------------------------------------------------------------------
65  * 機能          : Ethernet から 10 フレームの送信を行います。
66  *              :
67  *-------------------------------------------------------------------------------
68  * 引数           : なし
69  *-------------------------------------------------------------------------------
70  * 戻り値         : なし
71  *-------------------------------------------------------------------------------
72  * 注意事項       : 本サンプルでは、送信元の MAC アドレスを 01-23-45-67-89-AB、IP アドレスを
73  *              : 192.168.0.164、送信先の MAC アドレスを 00-0E-35-18-34-FA、IP アドレスを
74  *              : 192.168.0.5 とし、298 バイト(CRC 除)のフレームを 10 フレーム送信しています。
75  *""FUNC COMMENT END""*************************************************************/
76  void main(void)
77  {
78      OPEN_STATUS opensts;
79      SEND_STATUS sendsts;
80      CLOSE_STATUS    closests;
81      int i;
82
83      /* ==== EtherC/E-DMAC、PHY、バッファメモリの初期化  ==== */
84      opensts = lan_open(0);   /* ch0 を選択 */
85
86      /* ==== オープン時に送信  ==== */
87      if(opensts == OPEN_OK){
88
89        /* ==== パケット送信  ==== */
90        for(i=0;i<10;i++){
91            sendsts = lan_send(0,frame,sizeof(frame));
92            if(sendsts == SEND_NG){    /* 送信失敗 */
93                break;
94            }
95        }
96      }
97
98      /* ==== LAN クローズ ==== */
99      closests = lan_close(0);
100     while(closests == CLOSE_NG){
101       ; /* waiting */
102     }
103 }
104 /* End of File */
```

3.3 サンプルプログラムリスト” ether.c”(1)

```
1   /*""FILE COMMENT""*****************************************************************
2   *
3   *   System Name  : SH7712 Sample Program
4   *   FILE Name    : ether.c
5   *   Version      : 1.00.00
6   *   Contents     : Ethernet 送信処理
7   *   Model        : MS7712SE01
8   *   CPU          : SH7712
9   *   Compiler     : SHC9.1.1.0
10  *   OS           :  None
11  *
12  *   note         : Ethernet 送信の参考プログラムです。
13  *                    <注意事項>
14  *                    本サンプルプログラムはすべて参考資料であり
15  *                    その動作を保証するものではありません。
16  *                    本サンプルプログラムはお客様のソフトウエア開発時の
17  *、                   技術参考資料としてご利用ください。
18  *
19  *   Copyright  (C)  2007 Renesas Technology Corp. All Rights Reserved
20  *   AND Renesas Solutions Corp. All Rights Reserved
21  *
22  *   history      : 2007.11.07 ver 1.00.00
23  *""FILE COMMENT END""*************************************************************/
24  #include <machine.h>
25  #include <string.h>
26  #include "iodefine.h"
27  #include "ether.h"
28  #include "phy.h"
29
30  /* ==== プロトタイプ宣言 ==== */
31  static void lan_reg_reset(int ch);
32  static void lan_reg_set(int ch, int link);
33  static void lan_desc_create(int ch);
34
35  /* ==== 変数宣言 ==== */
36  volatile static int   tx_flag0,tx_flag1;
37  static    EDMAC_SEND_DESC  *psenddesc0;
38  static    EDMAC_SEND_DESC  *psenddesc1;
39  /* ディスクリプタを 16 バイト境界に配置する必要があるため、専用のセクションとします。 */
40  #pragma section TXRXBUFFDESC
41  static    TXRX_BUFFER_SET       buffer0,buffer1;
42  static    TXRX_DESCRIPTOR_SET    descriptor0,descriptor1;
43  #pragma section
```

3.4 サンプルプログラムリスト” ether.c”(2)

```
44  /*""FUNC COMMENT""*****************************************************************
45  * ID            :
46  * モジュール概要  : LAN オープン関数
47  *-------------------------------------------------------------------------------
48  * Include        : #include  "iodefine.h"
49  *                : #include  "ether.h"
50  *                : #include  "phy.h"
51  *-------------------------------------------------------------------------------
52  * 宣言           : int lan_open(int ch)
53  *-------------------------------------------------------------------------------
54  * 機能           : E-DMAC,EtherC,送受信バッファおよび送受信ディスクリプタの初期化を行います。
55  *                : 引数 ch をチェックし、0ch または 1ch 以外のときにエラーを返します。
56  *                : PHY-LSI の自動交渉結果を取得し、自動交渉が失敗の場合はエラーを返します。
57  *-------------------------------------------------------------------------------
58  * 引数           : int ch: I : チャネル番号(0 または 1)
59  *-------------------------------------------------------------------------------
60  * 戻り値         :  0(OPEN_OK) : オープン成功
61  *                : -1(OPEN_NG) : オープン失敗
62  *-------------------------------------------------------------------------------
63  * 注意事項       :
64  *       :
65  *""FUNC COMMENT END""*************************************************************/
66  int lan_open(int ch)
67  {
68      unsigned int physts;
69
70      /* ==== 引数 ch の確認 ==== */
71      if(ch == 0 || ch == 1){
72
73          /* ==== EtherC/E-DMAC のリセット ==== */
74          lan_reg_reset(ch);
75
76          /* ==== 送受信ディスクリプタの初期化 ==== */
77          lan_desc_create(ch);
78
79          /* ==== PHY の自動交渉結果の取得 ==== */
80          physts = phy_autonego(ch);
81
82          /* ==== 自動交渉成功時 ==== */
83          if(physts != NEGO_FAIL){
84
85              /* ==== EtherC/E-DMAC レジスタ設定 ==== */
86              lan_reg_set(ch,physts);
87
88              return OPEN_OK;
89          }
90      }
91      return OPEN_NG;
92  }
```

3.5 サンプルプログラムリスト” ether.c”(3)

```
93  /*""FUNC COMMENT""*************************************************************
94  * ID            :
95  * モジュール概要  : LAN クローズ関数
96  *-----------------------------------------------------------------------------
97  * Include       : #include  "iodefine.h"
98  *               : #include  "ether.h"
99  *-----------------------------------------------------------------------------
100 * 宣言          : int lan_close(int ch)
101 *-----------------------------------------------------------------------------
102 * 機能          : Ether 機能を停止して送受信を禁止します。
103 *               : 引数 ch をチェックし、0ch または 1ch 以外のときにエラーを返します。
104 *-----------------------------------------------------------------------------
105 * 引数          : int ch: I : チャネル番号(0 または 1)
106 *-----------------------------------------------------------------------------
107 * 戻り値        : 0(CLOSE_OK) : クローズ成功
108 *               : -1(CLOSE_NG) : クローズ失敗
109 *-----------------------------------------------------------------------------
110 * 注意事項      :
111 *               :
112 *               :
113 *               :
114 *""FUNC COMMENT END""*************************************************************/
115 int lan_close(int ch)
116 {
117     /* ==== 引数 ch の確認 ==== */
118     if(ch == 0 || ch == 1){
119
120         /* ==== EtherC,E-DMAC 関連レジスタのリセット ==== */
121         lan_reg_reset(ch);
122
123         /* ==== 割り込み優先レベル設定レジスタ(IPRG)の設定 ==== */
124         if(ch == 0){
125             INTX.IPRG.BIT._EDMAC1 = 0x0;   /* E-DMAC0 の優先レベルを 0 に設定 */
126         }
127         else{
128             INTX.IPRG.BIT._EDMAC2 = 0x0;   /* E-DMAC1 の優先レベルを 0 に設定 */
129         }
130         return CLOSE_OK;
131     }
132     return CLOSE_NG;
133 }
```

3.6 サンプルプログラムリスト” ether.c”(4)

```
/*""FUNC COMMENT""*****************************************************************
* ID            :
* モジュール概要  : EtherC/E-DMAC リセット関数
*-------------------------------------------------------------------------------
* Include       :#include  "iodefine.h"
*-------------------------------------------------------------------------------
* 宣言          : static void lan_reg_reset(int ch)
*-------------------------------------------------------------------------------
* 機能          : E-DMAC,EtherC のソフトウエアリセットを行います。
*               :
*-------------------------------------------------------------------------------
* 引数          : int ch: I : チャネル番号(0 または 1)
*-------------------------------------------------------------------------------
* 戻り値        : なし
*               :
*-------------------------------------------------------------------------------
* 注意事項      :
*               :
*               :
*               :
*""FUNC COMMENT END""*************************************************************/
static void lan_reg_reset(int ch)
{
    volatile int t = 200;   /* 約 1us ウエイトカウント @198MHz */

    /* ==== E-DMAC モードレジスタ(EDMR)の設定 ==== */
    if(ch ==0){
        EDMAC0.EDMR.BIT.SWR =1;
    }
    else if(ch ==1){
        EDMAC1.EDMR.BIT.SWR =1;
    }
    else{
        /* DO NOTHING */
    }

    /* ==== Bφ で 64 サイクル(約 970ns@Bφ=66MHz)待つ必要あり。マニュアル 19-4 参照 ==== */
    while(--t){
        /* wait */
    }
}

```

3.7 サンプルプログラムリスト” ether.c”(5)

```
/*""FUNC COMMENT""*************************************************************
* ID            :
* モジュール概要  : EtherC/E-DMAC レジスタの初期設定
*-----------------------------------------------------------------------------
* Include        : #include  "iodefine.h"
*                : #include  "ether.h"
*                : #include  "phy.h"
*-----------------------------------------------------------------------------
* 宣言           : static void lan_reg_set(int ch, int link)
*-----------------------------------------------------------------------------
* 機能           : E-DMAC,EtherC レジスタの初期設定を行い送信可能状態にします。
*                : 受信の設定はしていますが、受信の起動は行っていません。
*                : 受信時に CAMSEN0,1 端子からの信号を参照しないように初期値を変更しています。
*-----------------------------------------------------------------------------
* 引数           : int ch:  I : チャネル番号(0 または 1)
*                : int link: I : PHY 自動交渉結果
*                :             HALF_10M(1), FULL_10M(2), HALF_TX(3), FULL_TX(4)
*                :
*-----------------------------------------------------------------------------
* 戻り値         :  なし
*                :
*-----------------------------------------------------------------------------
* 注意事項       : 外部 CAM を使用しないことを前提としています。
*                :
*""FUNC COMMENT END""*********************************************************/
static void lan_reg_set(int ch,int link)
{
    if(ch ==0){
        /* ==== EtherC ステータスレジスタ(ECSR)のクリア ==== */
        MAC0.ECSR.LONG = 0x00000007;      /* 1 ライトクリア */

        /* ==== EtherC/E-DMAC ステータスレジスタ(EESR)のクリア ==== */
        EDMAC0.EESR.LONG = 0x47FF0F9F;    /* 1 ライトクリア */

        /* ==== 送信ディスクリプタ先頭アドレスレジスタ(TDLAR)の設定 ==== */
        EDMAC0.TDLAR = descriptor0.send_desc;

        /* ==== 受ディスクリプタ先頭アドレスレジスタ(RDLAR)の設定 ==== */
        EDMAC0.RDLAR = descriptor0.recv_desc;

        /* ==== 送受信ステータスコピー指示レジスタ(TRSCER)の設定 ==== */
        EDMAC0.TRSCER.LONG = 0x00000000;

        /* ==== 送信 FIFO しきい値指定レジスタ(TFTR)の設定 ==== */
        EDMAC0.TFTR = 0x00000000;         /* ストア＆フォワードモード */

        /* ==== FIFO 容量指定レジスタ(FDR)の設定 ==== */
        EDMAC0.FDR.LONG = 0x00000707;     /* 送受信 FIFO 容量を 2KB に設定 */

        /* ==== 受信方式制御レジスタ(RMCR)の設定 ==== */
        EDMAC0.RCR.BIT.RNC = 0x1;         /* 連続受信 */

```

3.8 サンプルプログラムリスト” ether.c”(6)

```
228         /* ==== MAC アドレス上位/下位設定レジスタ(MAHR,MALR)の設定 ==== */
229         MAC0.MAHR = MAC_ADDRESS_HIGH0;
230         MAC0.MALR = MAC_ADDRESS_LOW0;
231
232         /* ==== 転送機能設定レジスタ(TSU_FWSLC)の設定 ==== */
233         TSU.TSU_FWSLC.LONG = 0x00000000;  /* CAMSEN0,1 端子を使わないときは 0 に設定してください */
234
235         /* ==== 割り込み優先レベル設定レジスタ(IPRG)の設定 ==== */
236         INTX.IPRG.BIT._EDMAC1 = EDMAC0_PRIORITY;    /* E-DMAC0 の優先レベルを設定 */
237
238         /* ==== EtherC 割り込み許可レジスタ(ECSIPR)の設定 ==== */
239         MAC0.ECSIPR.LONG = 0x0000;  /* 不許可に設定 */
240
241         /* ==== EtherC/E-DMAC ステータス割り込み許可レジスタ(EESIPR)の設定 ==== */
242         EDMAC0.EESIPR.LONG = 0x04380300;    /* 送信関連要因のみ許可 */
243
244         /* ==== EtherC モードレジスタ(ECMR)の設定 ==== */
245         if(link == FULL_TX || link == FULL_10M){
246             MAC0.ECMR.BIT.DM =1;    /* 全二重方式 */
247         }
248         else {
249             MAC0.ECMR.BIT.DM =0;    /* 半二重方式 */
250         }
251         MAC0.ECMR.BIT.RE = 1;   /* 受信許可 */
252         MAC0.ECMR.BIT.TE = 1;   /* 送信許可 */
253
254         /* ==== E-DMAC 受信要求レジスタ(EDRRR)の設定 ==== */
255         EDMAC0.EDRRR.LONG = 0x00000000; /* 受信無効状態 */
256     }
257     else if(ch ==1){
258         /* ==== EtherC ステータスレジスタ(ECSR)のクリア ==== */
259         MAC1.ECSR.LONG = 0x00000007; /* 1 ライトクリア */
260
261         /* ==== EtherC/E-DMAC ステータスレジスタ(EESR)のクリア ==== */
262         EDMAC1.EESR.LONG = 0x47FF0F9F;   /* 1 ライトクリア */
263
264         /* ==== 送信ディスクリプタ先頭アドレスレジスタ(TDLAR)の設定 ==== */
265         EDMAC1.TDLAR = descriptor1.send_desc;
266
267         /* ==== 受ディスクリプタ先頭アドレスレジスタ(RDLAR)の設定 ==== */
268         EDMAC1.RDLAR = descriptor1.recv_desc;
269
270         /* ==== 送受信ステータスコピー指示レジスタ(TRSCER)の設定 ==== */
271         EDMAC1.TRSCER.LONG = 0x00000000;
272
273         /* ==== 送信 FIFO しきい値指定レジスタ(TFTR)の設定 ==== */
274         EDMAC1.TFTR = 0x00000000;   /* ストア＆フォワードモード */
275
276         /* ==== FIFO 容量指定レジスタ(FDR)の設定 ==== */
277         EDMAC1.FDR.LONG = 0x00000707;   /* 送受信 FIFO 容量を 2KB に設定 */
278
279         /* ==== 受信方式制御レジスタ(RMCR)の設定 ==== */
280         EDMAC1.RCR.BIT.RNC = 0x1;   /* 連続受信 */
```

3.9 サンプルプログラムリスト” ether.c”(7)

```
281          /* ==== MAC アドレス上位/下位設定レジスタ(MAHR,MALR)の設定 ==== */
282          MAC1.MAHR = MAC_ADDRESS_HIGH1;
283          MAC1.MALR = MAC_ADDRESS_LOW1;
284
285          /* ==== 転送機能設定レジスタ(TSU_FWSLC)の設定 ==== */
286          TSU.TSU_FWSLC.LONG = 0x00000000;    /* CAMSEN0,1 端子を使わないときは 0 に設定してください */
287
288          /* ==== 割り込み優先レベル設定レジスタ(IPRG)の設定 ==== */
289          INTX.IPRG.BIT._EDMAC2 = EDMAC1_PRIORITY;    /* E-DMAC1 の優先レベルを設定 */
290
291          /* ==== EtherC 割り込み許可レジスタ(ECSIPR)の設定 ==== */
292          MAC1.ECSIPR.LONG = 0x0000;  /* 不許可に設定 */
293
294          /* ==== EtherC/E-DMAC ステータス割り込み許可レジスタ(EESIPR)の設定 ==== */
295          EDMAC1.EESIPR.LONG = 0x4380300;    /* 送信関連要因のみ許可 */
296
297          /* ==== EtherC モードレジスタ(ECMR)の設定 ==== */
298          if(link == FULL_TX || link == FULL_10M){
299              MAC1.ECMR.BIT.DM =1;    /* 全二重方式 */
300          }
301          else {
302              MAC1.ECMR.BIT.DM =0;    /* 半二重方式 */
303          }
304          MAC1.ECMR.BIT.RE =1;    /* 受信許可 */
305          MAC1.ECMR.BIT.TE =1;    /* 送信許可 */
306
307          /* ==== E-DMAC 受信要求レジスタ(EDRRR)の設定 ==== */
308          EDMAC1.EDRRR.LONG = 0x00000000; /* 受信無効状態 */
309      }
310      else{
311          /* DO NOTHING */
312      }
313  }
```

3.10 サンプルプログラムリスト” ether.c”(8)

```
314    /*""FUNC COMMENT""*************************************************************
315    * ID            :
316    * モジュール概要  : 送受信ディスクリプタの初期化
317    *-------------------------------------------------------------------------------
318    * Include        : #include  "ether.h"
319    *-------------------------------------------------------------------------------
320    * 宣言           : static void lan_desc_create(int ch)
321    *-------------------------------------------------------------------------------
322    * 機能           :送受信ディスクリプタを初期化します。
323    *                :
324    *-------------------------------------------------------------------------------
325    * 引数           : int ch: I :チャネル番号(0 または 1)
326    *                :
327    *-------------------------------------------------------------------------------
328    * 戻り値         :  なし
329    *                :
330    *-------------------------------------------------------------------------------
331    * 注意事項       : TACT と TDL は 0 で初期化しています。これらは lan_send 関数で設定しています。
332    *                :
333    *                :
334    *                :
335    *""FUNC COMMENT END""*********************************************************/
336    static void lan_desc_create(int ch)
337    {
338        int i;
339        EDMAC_SEND_DESC *psnd;
340        EDMAC_RECV_DESC *prcv;
341
342        if(ch ==0){
343            /* ==== 送受信ディスクリプタの0クリア ==== */
344            memset(&descriptor0,0x0,sizeof(descriptor0));
345
346            /* ==== 送信ディスクリプタ初期設定 ==== */
347            psnd = descriptor0.send_desc;
348            for(i = 0; i<NUM_OF_TX_DESCRIPTOR ;i++){
349                psnd->td2.TBA = &buffer0.send_buf[i][0];
350                psnd->td0.BIT.TFP =0x3; /* 1 フレーム/1 ディスクリプタ方式 */
351                psnd->pNext = psnd +1;
352                psnd++;
353            }
354            psnd--;
355            psnd->td0.BIT.TDLE = 1;
356            psnd->pNext = descriptor0.send_desc;
357
358            /* ==== 受信ディスクリプタ初期設定 ==== */
359                /* 省略 */
360
361            /* ==== 送受信バッファの 0 クリア ==== */
362            memset(&buffer0,0x0,sizeof(buffer0));
363
364            /* ==== 送信ディスクリプタへのポインタ初期設定 ==== */
365            psenddesc0 = descriptor0.send_desc;
366        }
```

3.11 サンプルプログラムリスト” ether.c”(9)

```
367         else if(ch ==1){
368             /* ==== 送受信ディスクリプタの 0 クリア ==== */
369             memset(&descriptor1,0x0,sizeof(descriptor1));
370
371             /* ==== 送信ディスクリプタ初期設定 ==== */
372             psnd = descriptor1.send_desc;
373             for(i = 0; i<NUM_OF_TX_DESCRIPTOR ;i++){
374                 psnd->td2.TBA = &buffer1.send_buf[i][0];
375                 psnd->td0.BIT.TFP =0x3; /* 1 フレーム/1 ディスクリプタ方式 */
376                 psnd->pNext = psnd +1;
377                 psnd++;
378             }
379             psnd--;
380             psnd->td0.BIT.TDLE = 1;
381             psnd->pNext = descriptor1.send_desc;
382
383             /* ==== 受信ディスクリプタ初期設定 ==== */
384                 /* 省略 */
385
386             /* ==== 送受信バッファの 0 クリア ==== */
387             memset(&buffer1,0x0,sizeof(buffer1));
388
389             /* ==== 送信ディスクリプタへのポインタ初期設定 ==== */
390             psenddesc1 = descriptor1.send_desc;
391         }
392         else{
393             /* DO NOTHING */
394         }
395     }
```

3.12 サンプルプログラムリスト” ether.c”(10)

```
396  /*""FUNC COMMENT""*****************************************************************
397  * ID            :
398  * モジュール概要  : イーサネットフレーム送信関数
399  *-----------------------------------------------------------------------------------
400  * Include       : #include  "iodefine.h"
401  *               : #include  "ether.h"
402  *-----------------------------------------------------------------------------------
403  * 宣言          : int lan_send(int ch, unsigned char *addr, int flen)
404  *-----------------------------------------------------------------------------------
405  * 機能          : 引数で指定されたフレームを送信バッファに設定します。
406  *               : 送信ディスクリプタを設定し、送信を起動します。送信中フラグ”を確認し、OFF であれば送信
407  *               : 完了と判断します。送信が完了するまで本関数で待ちます。
408  *               : テクニカルアップデート「TN-SH7-A575B/J」及び「TN-SH7-A583A/J」にしたがい、
409  *               : 不具合対策をしています。
410  *-----------------------------------------------------------------------------------
411  * 引数          : int ch: I : チャネル番号(0 または 1)
412  *               : unsigned char *addr : I : 送信する Ether フレームの先頭アドレス
413  *               : int flen : I   : フレームサイズ(バイト数)
414  *-----------------------------------------------------------------------------------
415  * 戻り値        : 0(SEND_OK) : 送信成功
416  *               : -1(SEND_NG): 送信失敗
417  *-----------------------------------------------------------------------------------
418  * 注意事項      : 1 フレームずつ送信し、送信完了後に次の送信のための設定をする仕様にしています。
419  *               :「TN-SH7-A575B/J」: 割り込みが発生しても EESR レジスタに反映されない場合がある不具合。
420  *               : 対策：TACT ビットを確認し、ライトバック(TACT=0)されていれば送信完了と判断しています。
421  *               :「TN-SH7-A583A/J」：送信アンダフローにより E-DMAC が動作停止する可能性がある不具合。
422  *               : 対策：FDR レジスタで FIFO サイズ 2KB を選択し、TFTR レジスタでストア&フォワードモードに
423  *               : することでアンダフロー発生を回避できます。これをしない場合、アンダフローにより
424  *               : E-DMAC が動作停止する可能性があります。本プログラムでは再送 15 回失敗による最大所要
425  *               : 時間(約 400ms)だけ待ち、それまでに TACT が 0 にならない場合はエラーとしています。
426  *""FUNC COMMENT END""*************************************************************/
427  int lan_send(int ch, unsigned char *addr, int flen)
428  {
429      volatile int w;
430      volatile int   t1ms = 100; /* 1ms カウンタ */
431      volatile int   t400ms = 400;   /* 400ms カウンタ */
432      int value;
433
434      if(ch ==0){
435          /* ==== 送信ディスクリプタの TACT ビットが 0 になるまで待つ ==== */
436          while(psenddesc0->td0.BIT.TACT == 1){
437              /* wait */
438          }
439
440          /* ==== 送信中フラグを ON ==== */
441          tx_flag0 = TX_FLAG_ON;
442
443          /* ==== 送信バッファに引数指定の送信用データをコピー ==== */
444          memcpy(psenddesc0->td2.TBA,addr,flen);
445
446          /* ==== 送信ディスクリプタの設定==== */
447          psenddesc0->td1.TDL = flen;
448          psenddesc0->td0.BIT.TACT = 1;
```

3.13 サンプルプログラムリスト” ether.c”(11)

```
449          /* ==== 送信起動 ==== */
450          /* マニュアル 19-36  19.4.1 E-DMAC 送信要求レジスタ(EDTRR)使用上の注意事項 */
451          if(EDMAC0.EDTRR.BIT.TR == 0){
452              EDMAC0.EDTRR.BIT.TR = 1;
453          }
454
455          /* ==== 送信完了の確認  ==== */
456          while(tx_flag0 == TX_FLAG_ON){  /* 送信中フラグがオン */
457
458              /* ==== 10us wait ==== */
459              for(w=0;w<2000;w++){
460                  ;
461              }
462
463              /* ==== テクニカルアップデート「TN-SH7-A575B/J」の対策 ==== */
464              if((--t1ms) <= 0){  /* 1ms 経過時 */
465                  t1ms = 100; /* 1ms カウンタの初期化 */
466                  if(psenddesc0->td0.BIT.TACT ==0){
467                      tx_flag0 = TX_FLAG_OFF;
468                      break;
469                  }
470                  /* ==== テクニカルアップデート「TN-SH7-A583A/J」の対策 ==== */
471                  else if((--t400ms) <= 0){   /* 400ms 経過時 */
472                      return SEND_NG;
473                  }
474                  else{
475                      /* DO NOTHING */
476                  }
477              }
478          }
479
480          /* ==== 送信ディスクリプタの設定==== */
481          psenddesc0 = psenddesc0->pNext; /* ディスクリプタ管理用ポインタの更新 */
482
483          return SEND_OK;
484      }
485      else if(ch ==1){
486          /* ==== 送信ディスクリプタの TACT ビットが 0 になるまで待つ=== */
487          while(psenddesc1->td0.BIT.TACT == 1){
488              /* wait */
489          }
490
491          /* ==== 送信中フラグを ON ==== */
492          tx_flag1 = TX_FLAG_ON;
493
```

3.14 サンプルプログラムリスト” ether.c”(12)

```
494             /* ==== 送信バッファに送信用データをコピー ==== */
495             memcpy(psenddesc1->td2.TBA,addr,flen);
496
497             /* ==== 送信ディスクリプタの設定==== */
498             psenddesc1->td1.TDL = flen;
499             psenddesc1->td0.BIT.TACT = 1;
500
501             /* ==== 送信起動 ==== */
502             /* マニュアル 19-36  19.4.1 E-DMAC 送信要求レジスタ(EDTRR)使用上の注意事項 */
503             if(EDMAC1.EDTRR.BIT.TR == 0){
504                 EDMAC1.EDTRR.BIT.TR = 1;
505             }
506
507             /* ==== 送信完了の確認  ==== */
508             while(tx_flag1 == TX_FLAG_ON){  /* 送信中フラグがオン */
509
510                 /* ==== 10us wait ==== */
511                 for(w=0;w<2000;w++){
512                     ;
513                 }
514                 /* ==== テクニカルアップデート「TN-SH7-A575B/J」の対策 ==== */
515                 if((--t1ms) <= 0){  /* 1ms 経過時 */
516                     t1ms = 100; /* 1ms カウンタの初期化 */
517                     if(psenddesc1->td0.BIT.TACT ==0){
518                         tx_flag1 = TX_FLAG_OFF;
519                         break;
520                     }
521                     /* ==== テクニカルアップデート「TN-SH7-A583A/J」の対策 ==== */
522                     else if((--t400ms) <= 0){   /* 400ms 経過時 */
523                         return SEND_NG;
524                     }
525                     else{
526                         /* DO NOTHING */
527                     }
528                 }
529             }
530
531             /* ==== 送信ディスクリプタの設定==== */
532             psenddesc1 = psenddesc1->pNext;      /* ディスクリプタ管理用ポインタの更新 */
533
534             return SEND_OK;
535         }
536
537         return SEND_NG;
538     }
```

3.15 サンプルプログラムリスト” ether.c”(13)

```
539  /*""FUNC COMMENT""*************************************************************
540  * ID            :
541  * モジュール概要  : E-DMAC 送信完了割り込み処理
542  *-----------------------------------------------------------------------------
543  * Include       : #include  "iodefine.h"
544  *               : #include  "ether.h"
545  *-----------------------------------------------------------------------------
546  * 宣言          : void INT_EDMAC_EINT0(void)
547  *-----------------------------------------------------------------------------
548  * 機能          : ch0 のフレーム送信完了割り込み処理を行います。
549  *               : フレーム送信完了時、”送信中フラグ”を OFF にします。
550  *-----------------------------------------------------------------------------
551  * 引数          : なし
552  *               :
553  *-----------------------------------------------------------------------------
554  * 戻り値        : なし
555  *               :
556  *-----------------------------------------------------------------------------
557  * 注意事項      :
558  *               :
559  *               :
560  *""FUNC COMMENT END""*********************************************************/
561  void INT_EDMAC_EINT0(void)
562  {
563      unsigned int status;
564
565      /* ==== 割り込みステータス読み出し ==== */
566      status = EDMAC0.EESR.LONG & EDMAC0.EESIPR.LONG;
567
568      /* ==== 割り込み要因クリア ==== */
569      EDMAC0.EESR.LONG = status;  /* 1 ライトクリア */
570
571      /* ==== フレーム送信完了時 ==== */
572      if(status & FRAME_TRANSMIT_COMPLETE){
573          tx_flag0 = TX_FLAG_OFF;
574      }
575  }
576  /*INT_EDMAC_EINT1(void)は省略 */
577  /* End of File */
```

3.16 サンプルプログラムリスト” ether.h”(1)

```
1   /*""FILE COMMENT""*************************************************************
2   *
3   *   System Name  : SH7712 Sample Program
4   *   FILE Name    : ether.h
5   *   Version      : 1.00.00
6   *   Contents     : ヘッダファイル
7   *   Model        : MS7712SE01
8   *   CPU          : SH7712
9   *   Compiler     : SHC9.1.1.0
10  *   OS           : None
11  *
12  *   note         : 送受信ディスクリプタ、送受信バッファ設定の参考プログラムです。
13  *                  <注意事項>
14  *                  本サンプルプログラムはすべて参考資料であり
15  *                  その動作を保証するものではありません。
16  *                  本サンプルプログラムはお客様のソフトウエア開発時の
17  *                  技術参考資料としてご利用ください。
18  *
19  *
20  *   Copyright  (C)  2007 Renesas Technology Corp. All Rights Reserved
21  *      AND Renesas Solutions Corp. All Rights Reserved
22  *
23  *   history      : 2007.11.07 ver 1.00.00
24  *""FILE COMMENT END""*********************************************************/
25  #ifndef _ETHER_H
26  #define _ETHER_H
27
28  /* ==== 外部参照プロトタイプ宣言 ==== */
29  int lan_open(int ch);
30  int lan_recv(int ch, unsigned char *addr);
31  int lan_send(int ch, unsigned char *addr, int flen);
32  int lan_close(int ch);
33
34  /* ==== マクロ定義 ==== */
35  #define NUM_OF_TX_DESCRIPTOR        4   /* 送信用ディスクリプタの個数 */
36  #define NUM_OF_RX_DESCRIPTOR        4   /* 受信用ディスクリプタの個数 */
37  #define SIZE_OF_TX_BUFFER        1520   /* 送信バッファサイズ 16 バイトの整数倍にすること */
38  #define SIZE_OF_RX_BUFFER        1520   /* 受信バッファサイズ 16 バイトの整数倍にすること */
39  #define MAC_ADDRESS_HIGH0   0x01234567   /* MAC アドレスが 01-23-45-67-89-AB(16 進数)の場合 */
40  #define MAC_ADDRESS_LOW0    0x000089AB
41  #define MAC_ADDRESS_HIGH1   0x01234567   /* MAC アドレスが 01-23-45-67-89-AA(16 進数)の場合 */
42  #define MAC_ADDRESS_LOW1    0x000089AA
43  #define EDMAC0_PRIORITY           0xf   /* E-DMAC0 の優先レベル */
44  #define EDMAC1_PRIORITY           0xf   /* E-DMAC1 の優先レベル */
45  #define TX_FLAG_ON                  1
46  #define TX_FLAG_OFF                 0
47  #define FRAME_TRANSMIT_COMPLETE  0x00200000
48
49  /* ==== lan_open()用戻り値の列挙定数定義 ==== */
50  typedef enum{OPEN_OK= 0,OPEN_NG= -1}OPEN_STATUS;
51
52  /* ==== lan_send()用戻り値の列挙定数定義 ==== */
53  typedef enum{SEND_OK = 0,SEND_NG =-1}SEND_STATUS;
```

3.17 サンプルプログラムリスト” ether.h”(2)

```
54    /* ==== lan_close()用戻り値の列挙定数定義 ==== */
55    typedef enum{CLOSE_OK= 0,CLOSE_NG= -1}CLOSE_STATUS;
56
57    /* ==== 送信ディスクリプタ構造体定義 ==== */
58    typedef union
59    {
60        unsigned long LONG;
61        struct{
62            unsigned int    TACT:1;         /* 送信ディスクリプタ有効ビット */
63            unsigned int    TDLE:1;         /* 送信最終ディスクリプタリスト最終 */
64            unsigned int    TFP:2;          /* 送信フレーム内位置 */
65            unsigned int    TFE:1;          /* 送信フレームエラー発生(エラーの要因は TFSx を参照) */
66            unsigned int    reserved1:11;   /* 未使用 */
67            unsigned int    reserved2:7;    /* 未使用 */
68            unsigned int    TFS8:1;         /* 送信アボート */
69            unsigned int    reserved3:4;    /* 未使用 */
70            unsigned int    TFS3:1;         /* 送信開始時キャリア未検出 */
71            unsigned int    TFS2:1;         /* 送信中キャリア消失 */
72            unsigned int    TFS1:1;         /* 遅延衝突発生 */
73            unsigned int    TFS0:1;         /* 送信リトライオーバ */
74        }BIT;
75    }TD0;
76    typedef struct
77    {
78        unsigned short  TDL;                /* 送信バッファのサイズ(バイト数) */
79        unsigned short  reserved;
80    }TD1;
81    typedef struct
82    {
83        unsigned char   *TBA;               /* 送信バッファのアドレス */
84    }TD2;
85
86    typedef struct tag_edmac_send_desc
87    {
88        TD0    td0;
89        TD1    td1;
90        TD2    td2;
91        struct  tag_edmac_send_desc    *pNext;
92    }EDMAC_SEND_DESC;
```

3.18 サンプルプログラムリスト” ether.h”(3)

```
93    /* ==== 受信ディスクリプタ構造体定義 ==== */
94    typedef union
95    {
96        unsigned long LONG;
97        struct{
98            unsigned int    RACT:1;         /* 受信ディスクリプタ有効 */
99            unsigned int    RDLE:1;         /* 最終ディスクリプタリスト最終 */
100           unsigned int    RFP:2;          /* 受診フレーム内位置 */
101           unsigned int    RFE:1;          /* 受信フレームエラー発生(エラーの要因は TFSx を参照) */
102           unsigned int    reserved1:3;    /* 未使用 */
103           unsigned int    reserved2:8;    /* 未使用 */
104           unsigned int    reserved3:6;    /* 未使用 */
105           unsigned int    RFS9:1;         /* 受信 FIFO オーバフロー */
106           unsigned int    RFS8:1;         /* 受信中アボート検出 */
107           unsigned int    RFS7:1;         /* マルチキャストアドレスフレーム受診 */
108           unsigned int    reserved4:2;    /* 未使用 */
109           unsigned int    RFS4:1;         /* 端数ビットフレームエラー */
110           unsigned int    RFS3:1;         /* ロングフレーム受信エラー */
111           unsigned int    RFS2:1;         /* ショートフレーム受信エラー */
112           unsigned int    RFS1:1;         /* PHY-LSI 受信エラー */
113           unsigned int    RFS0:1;         /* 受信フレーム CRC エラー */
114       }BIT;
115   }RD0;
116   typedef struct
117   {
118       unsigned short  RBL;                /* 受信バッファデータ長(単位：バイト、16 バイト境界で指定) */
119       unsigned short  RDL;                /* 受信データ長(フレームの最後を受信したときに設定) */
120   }RD1;
121   typedef struct
122   {
123       unsigned char   *RBA;               /* 受信バッファの開始アドレス、SDRAM 時 16 バイト境界 */
124   }RD2;
125
126   typedef struct tag_edmac_recv_desc
127   {
128       RD0    rd0;
129       RD1    rd1;
130       RD2    rd2;
131       struct  tag_edmac_recv_desc  *pNext;
132   }EDMAC_RECV_DESC;
```

3.19 サンプルプログラムリスト” ether.h”(4)

```
133   /* ==== 送受信バッファ構造体定義 ==== */
134   typedef struct
135   {
136       /* 送信バッファ領域(16 バイト境界でアラインされていること) */
137       unsigned char   send_buf[NUM_OF_TX_DESCRIPTOR][SIZE_OF_TX_BUFFER];
138
139       /* 受信バッファ領域(16 バイト境界でアラインされていること) */
140       unsigned char   recv_buf[NUM_OF_RX_DESCRIPTOR][SIZE_OF_RX_BUFFER];
141   }TXRX_BUFFER_SET;
142
143   /* ==== 送受信ディスクリプタ構造体定義 ==== */
144   typedef struct
145   {
146       /* 送信ディスクリプタ(16 バイト境界でアラインされていること) */
147       EDMAC_SEND_DESC    send_desc[NUM_OF_TX_DESCRIPTOR];
148
149       /* 受信ディスクリプタ(16 バイト境界でアラインされていること) */
150       EDMAC_RECV_DESC    recv_desc[NUM_OF_RX_DESCRIPTOR];
151   }TXRX_DESCRIPTOR_SET;
152
153   #endif
154
155   /* End of File */
```

4. 参考ドキュメント

・ ソフトウェアマニュアル
SH3、SH3E、SH3-DSP ソフトウェアマニュアル　Rev5.00
(最新版をルネサス テクノロジ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
SH7710 グループハードウェアマニュアル　Rev.2.00
SH7712 ハードウェアマニュアル　Rev.1.00
SH7713 ハードウェアマニュアル　Rev.1.00
(最新版をルネサス テクノロジホームページから入手してください)。

5. ホームページとサポート窓口

・ ルネサステクノロジホームページ
http://japan.renesas.com/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2007.11.07	—	初版発行

本資料ご利用に際しての留意事項

1. 本資料は、お客様に用途に応じた適切な弊社製品をご購入いただくための参考資料であり、本資料中に記載の技術情報について弊社または第三者の知的財産権その他の権利の実施、使用を許諾または保証するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例など全ての情報の使用に起因する損害、第三者の知的財産権その他の権利に対する侵害に関し、弊社は責任を負いません。
3. 本資料に記載の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、あるいはその他軍事用途の目的で使用しないでください。また、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、それらの定めるところにより必要な手続を行ってください。
4. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例などの全ての情報は本資料発行時点のものであり、弊社は本資料に記載した製品または仕様等を予告なしに変更することがあります。弊社の半導体製品のご購入およびご使用に当たりましては、事前に弊社営業窓口で最新の情報をご確認頂きますとともに、弊社ホームページ（http://www.renesas.com）などを通じて公開される情報に常にご注意下さい。
5. 本資料に記載した情報は、正確を期すため慎重に制作したものですが、万一本資料の記述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、弊社はその責任を負いません。
6. 本資料に記載の製品データ、図、表などに示す技術的な内容、プログラム、アルゴリズムその他応用回路例などの情報を流用する場合は、流用する情報を単独で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断して下さい。弊社は、適用可否に対する責任は負いません。
7. 本資料に記載された製品は、各種安全装置や運輸・交通用、医療用、燃焼制御用、航空宇宙用、原子力、海底中継用の機器・システムなど、その故障や誤動作が直接人命を脅かしあるいは人体に危害を及ぼすおそれのあるような機器・システムや特に高度な品質・信頼性が要求される機器・システムでの使用を意図して設計、製造されたものではありません（弊社が自動車用と指定する製品を自動車に使用する場合を除きます）。これらの用途に利用されることをご検討の際には、必ず事前に弊社営業窓口へご照会下さい。なお、上記用途に使用されたことにより発生した損害等について弊社はその責任を負いかねますのでご了承願います。
8. 第7項にかかわらず、本資料に記載された製品は、下記の用途には使用しないで下さい。これらの用途に使用されたことにより発生した損害等につきましては、弊社は一切の責任を負いません。
   1）生命維持装置。
   2）人体に埋め込み使用するもの。
   3）治療行為（患部切り出し、薬剤投与等）を行なうもの。
   4）その他、直接人命に影響を与えるもの。
9. 本資料に記載された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、実装条件およびその他諸条件につきましては、弊社保証範囲内でご使用ください。弊社保証値を越えて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、弊社はその責任を負いません。
10. 弊社は製品の品質及および信頼性の向上に努めておりますが、特に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。弊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、誤動作防止設計などの安全設計（含むハードウエアおよびソフトウエア）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エアは、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致します。
11. 本資料に記載の製品は、これを搭載した製品から剥がれた場合、幼児が口に入れて誤飲する等の事故の危険性があります。お客様の製品への実装後に容易に本製品が剥がれることがなきよう、お客様の責任において十分な安全設計をお願いします。お客様の製品から剥がれた場合の事故につきましては、弊社はその責任を負いません。
12. 本資料の全部または一部を弊社の文書による事前の承諾なしに転載または複製することを固くお断り致します。
13. 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弊社営業窓口までご照会下さい。