

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のですが、誤りが無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、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。

SH7619 グループ

イーサネット送信設定例

要旨

この資料は、SH7619 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

SH7619

目次

1. はじめに	2
2. 応用例の説明	3
3. 参考プログラムリスト	18
4. 参考ドキュメント	34

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ライトバック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)
- ・ イーサネットフィジカルレイヤトランシーバ(PHY)
- ・ PHY インタフェース(PHYIF)
- ・ ピンファンクションコントローラ(PFC)

1.3 適用条件

- ・ マイコン: SH7619 (R4S76190)
- ・ 動作周波数: 内部クロック 125.00MHz
バスクロック 62.5MHz
周辺クロック 31.25MHz
- ・ 統合開発環境: ルネサステクノロジ製 High-performance Embedded Workshop Ver.4.03.00
- ・ 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=sh2 -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、SH7619 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。

「SH7619 グループ アプリケーションノート 内蔵 PHY モジュールの初期設定例」

「SH7619 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)、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 は送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 転送を専用に行います。

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。

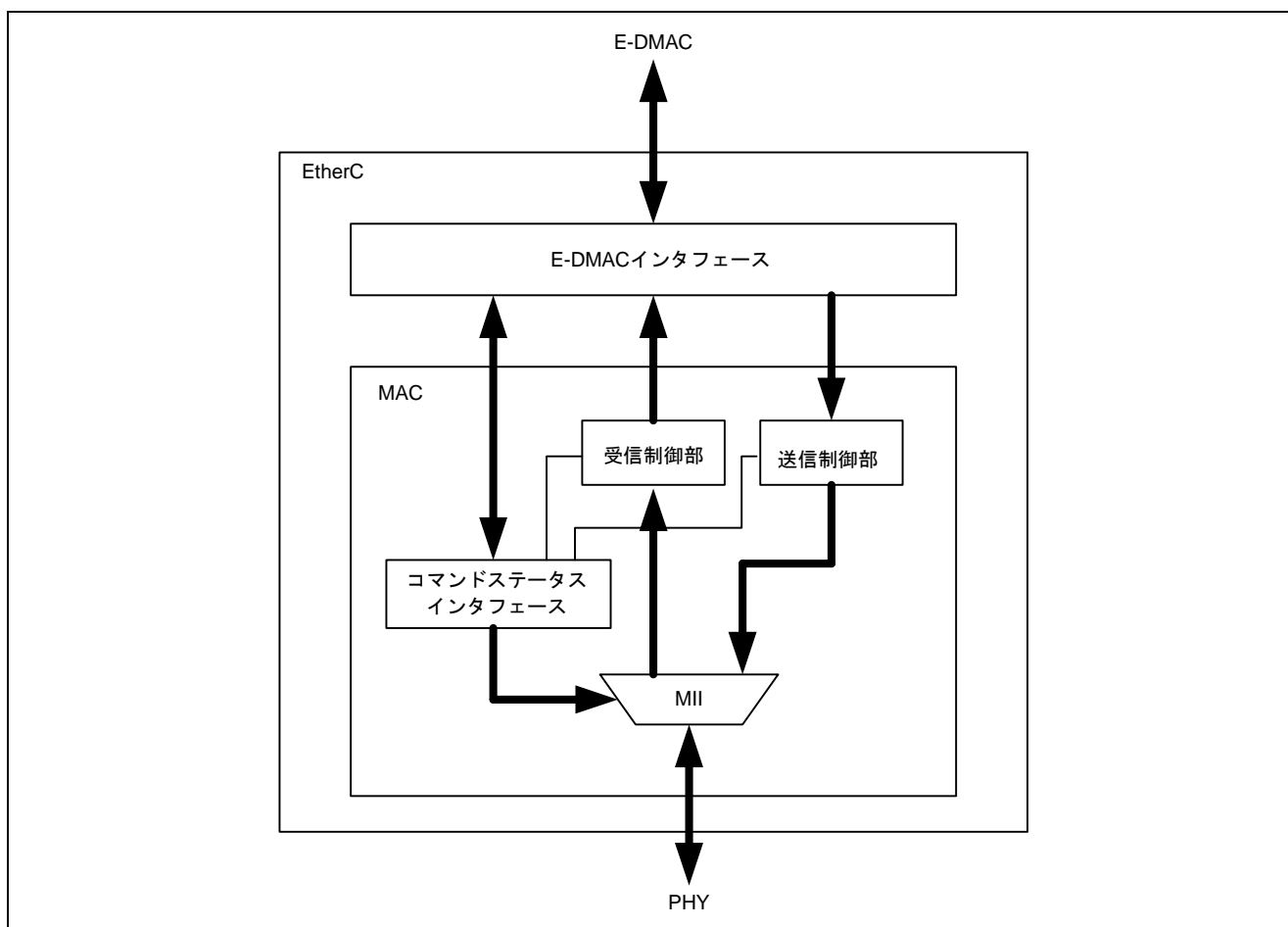


図1 EtherC の構成

2.1.2 EtherC 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。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プルを 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プルを 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プルを送出します。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プルを送出します。
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。

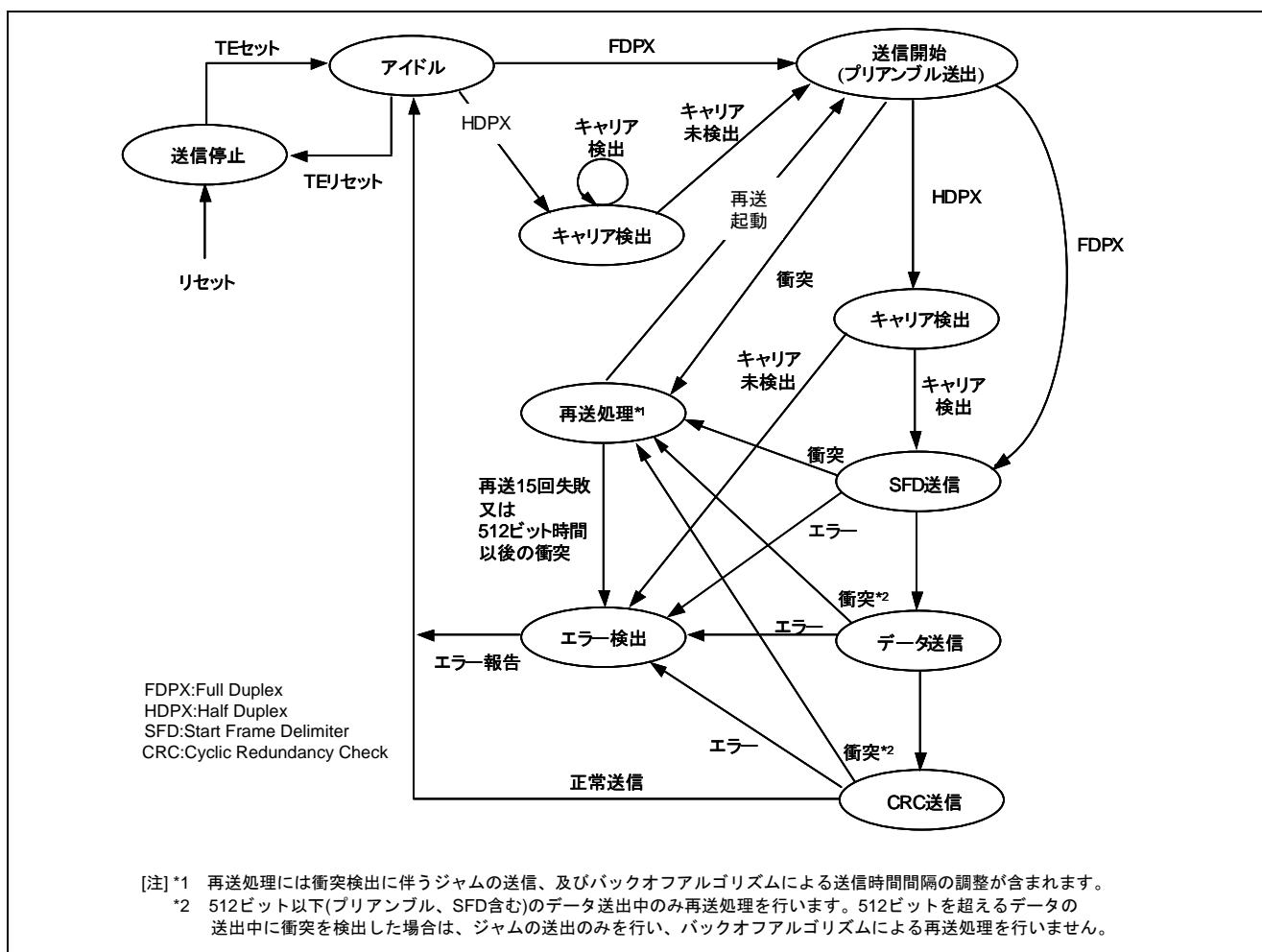


図2 EtherC 送信部状態遷移図

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(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。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)

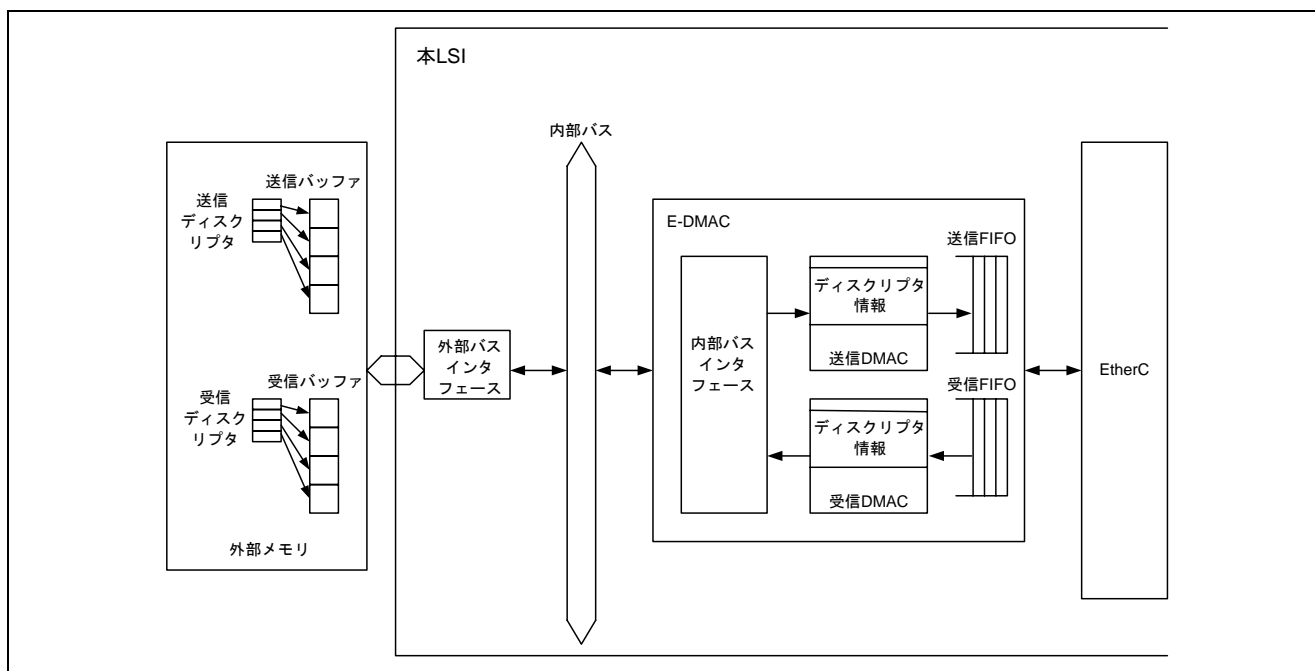


図3 E-DMAC とディスクリプタおよびバッファの構成

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。

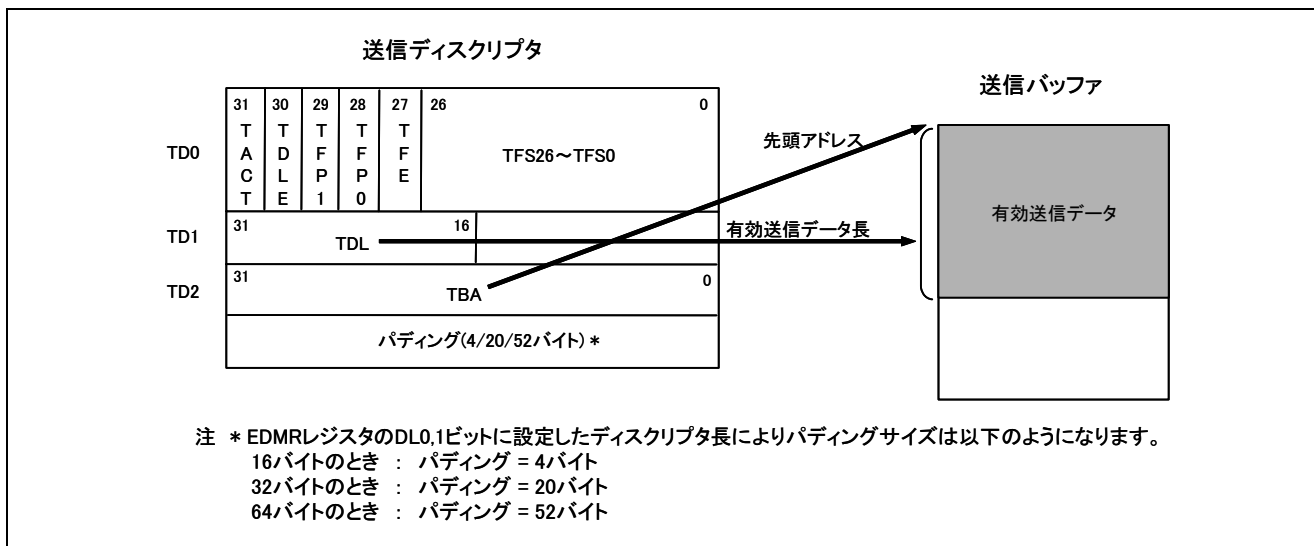


図4 送信ディスクリプタと送信バッファの関係

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。

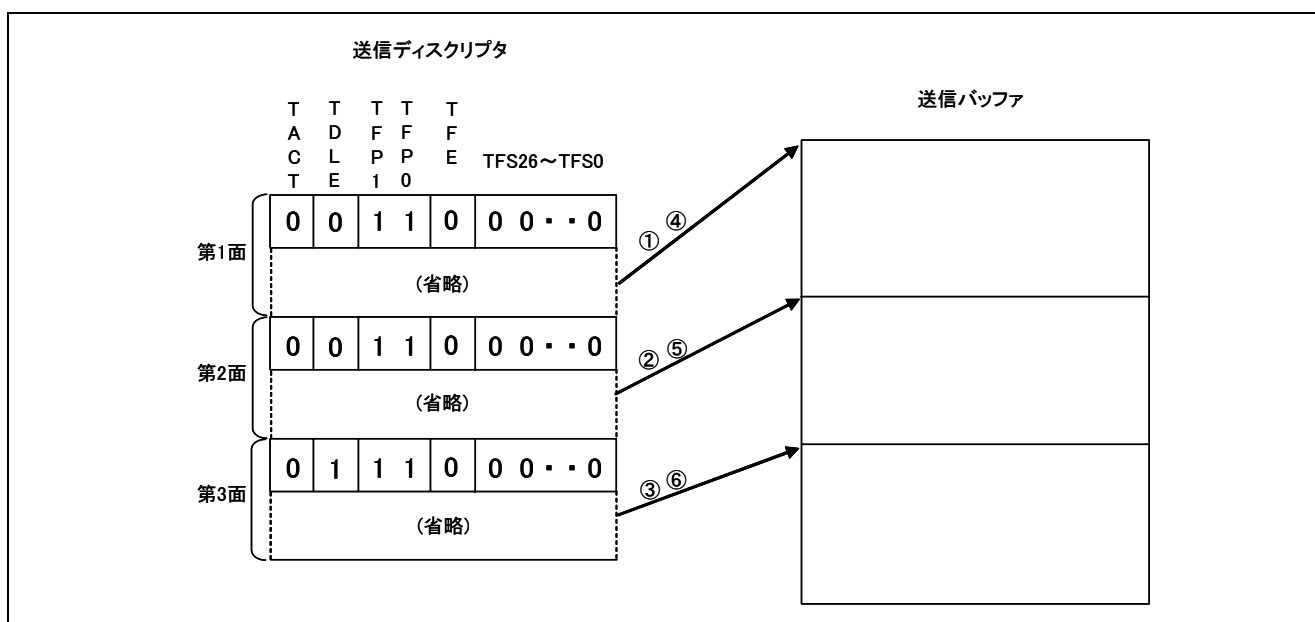


図5 送信ディスクリプタと送信バッファの関係

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 のソフトウェアリセット後、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。

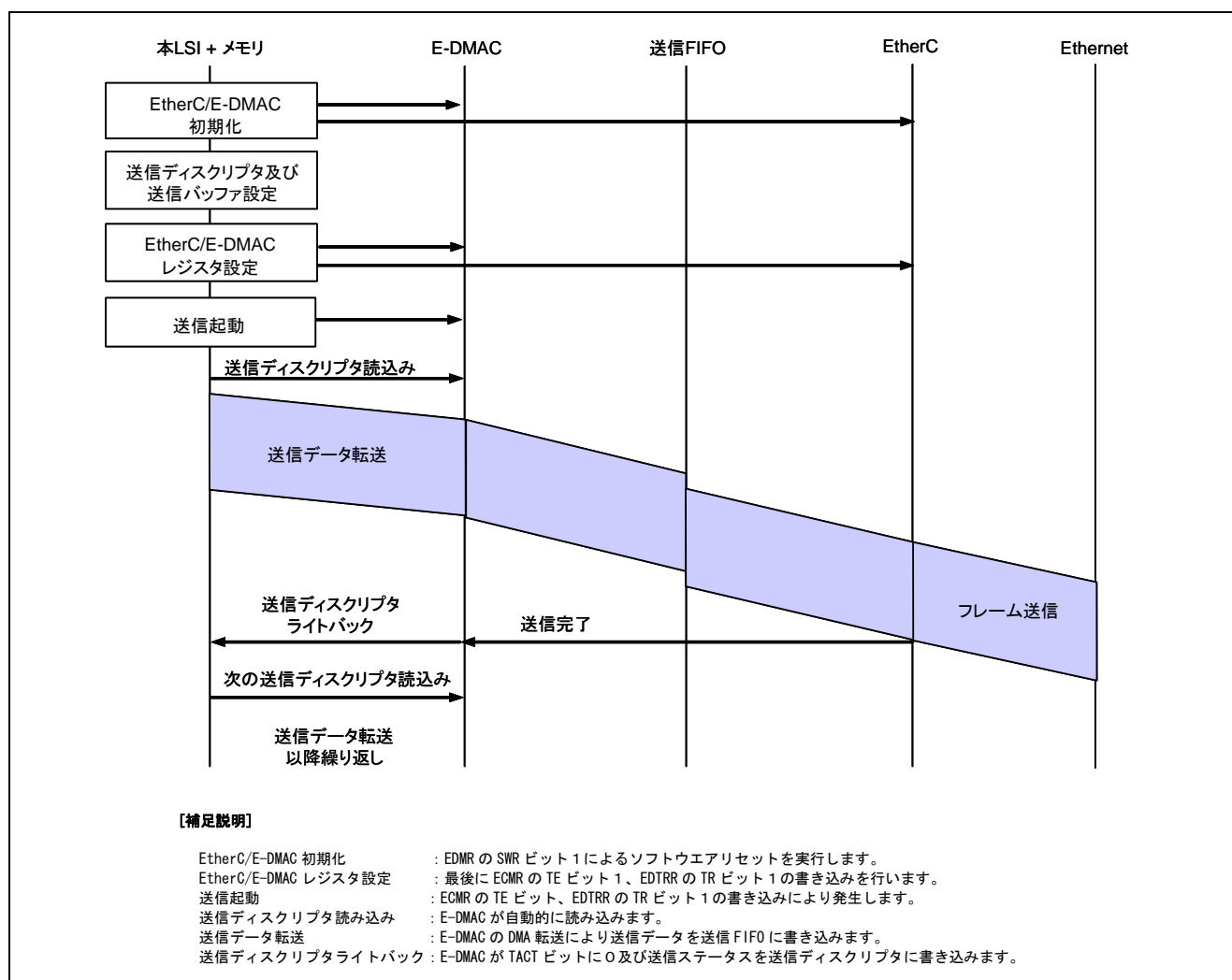


図6 送信フローの例(1 フレーム/1 ディスクリプタ)

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。

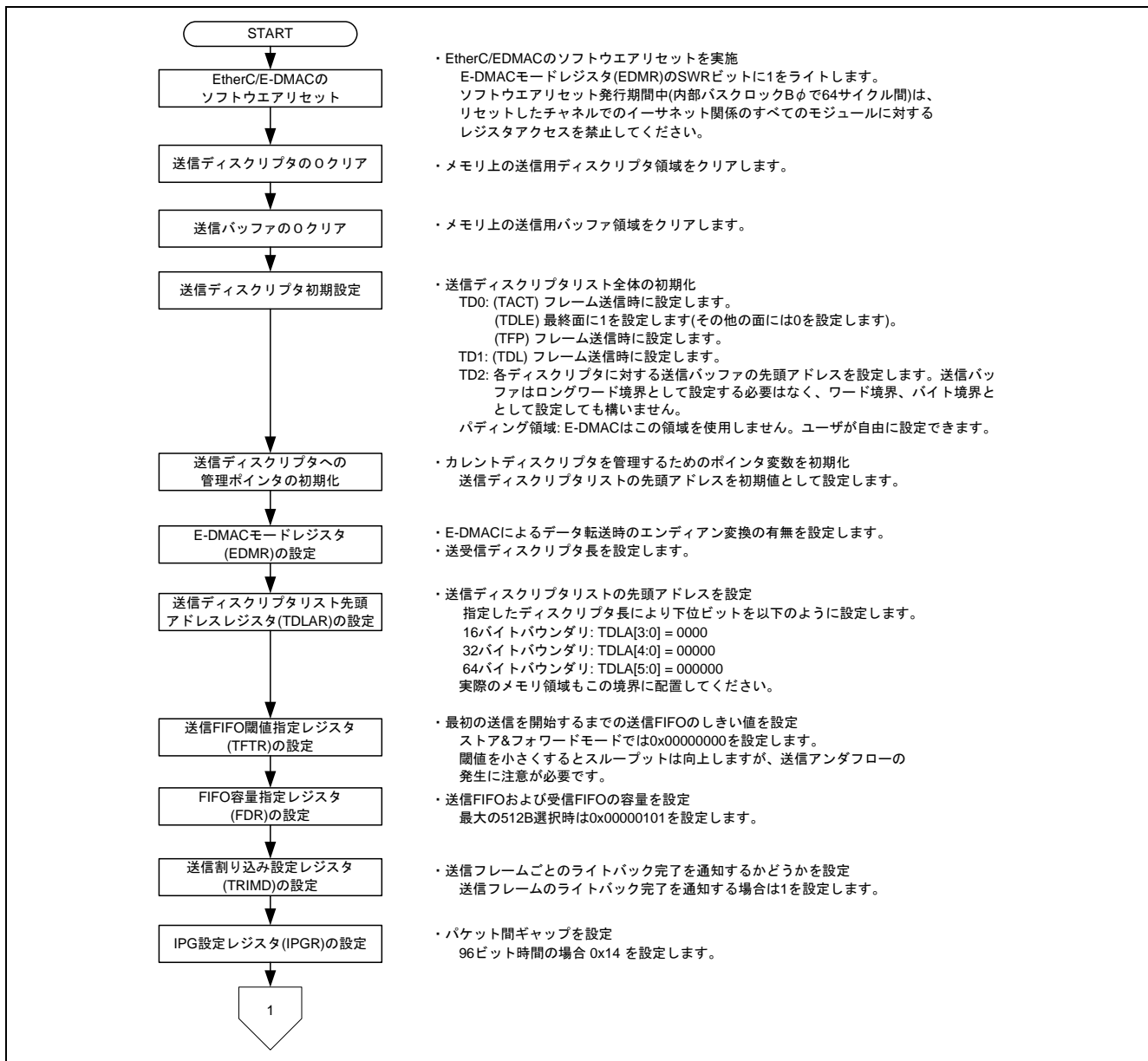


図7 イーサネット送信設定フロー例(1)

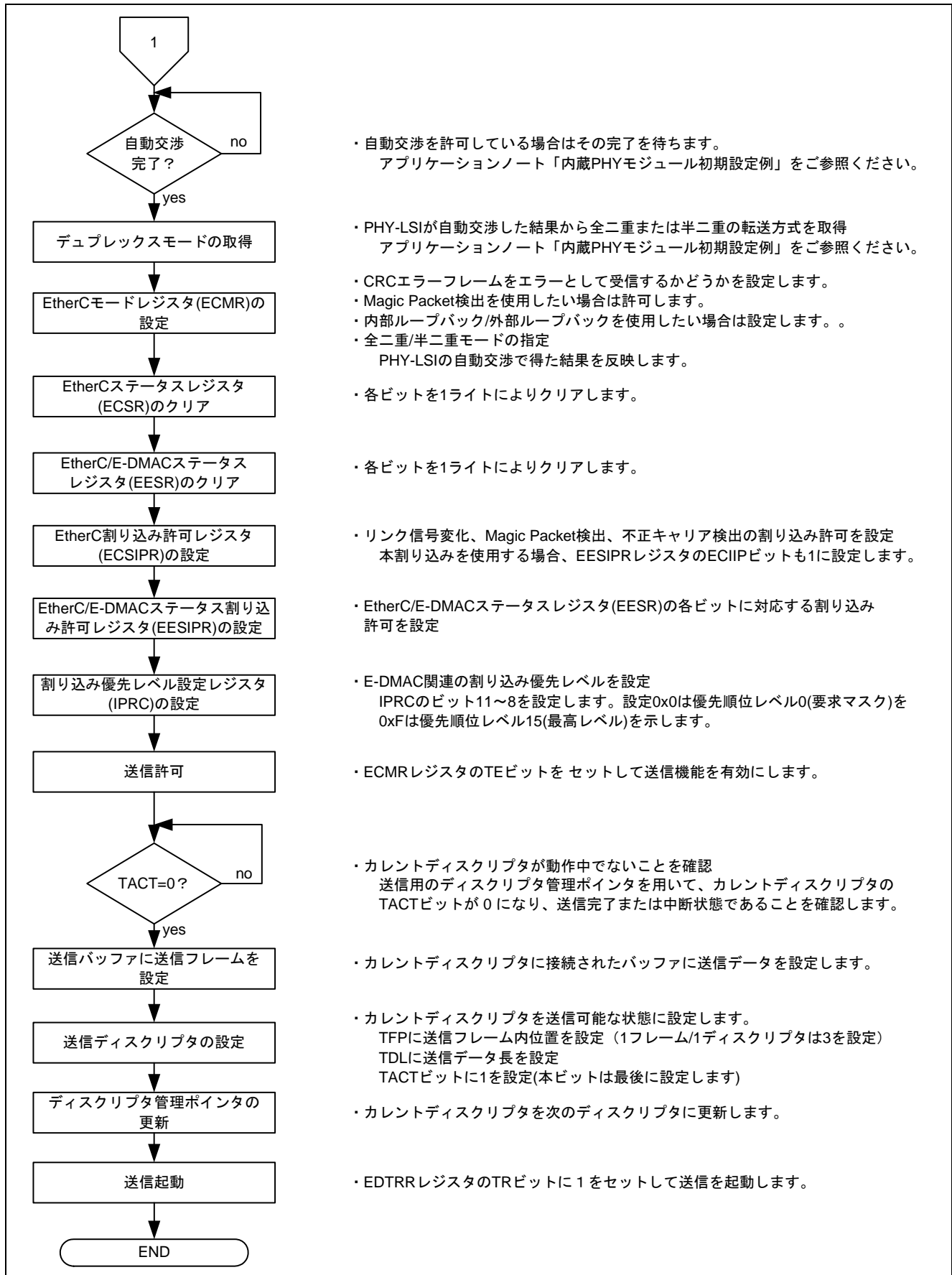


図8 イーサネット送信設定フロー例(2)

2.2 参考プログラムの動作

参考プログラムでは、EtherC のおよび E-DMAC 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。ライトバック完了割り込み(TWB)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プル、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

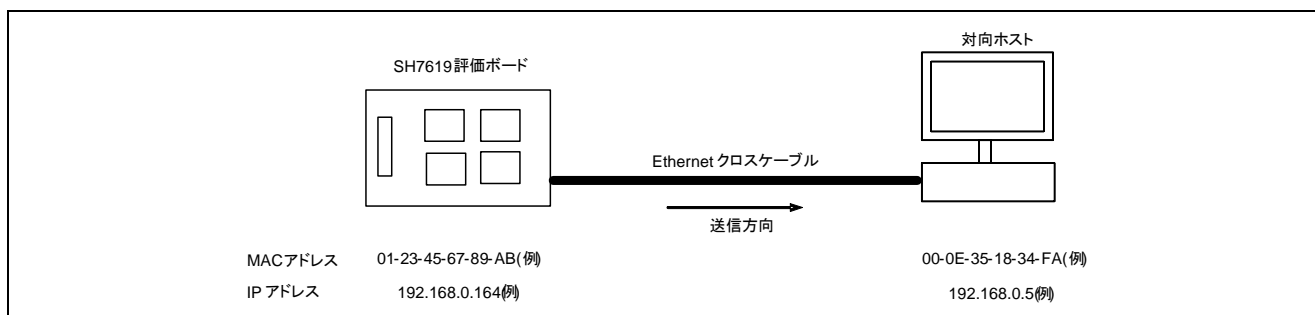


図9 参考プログラムの動作環境

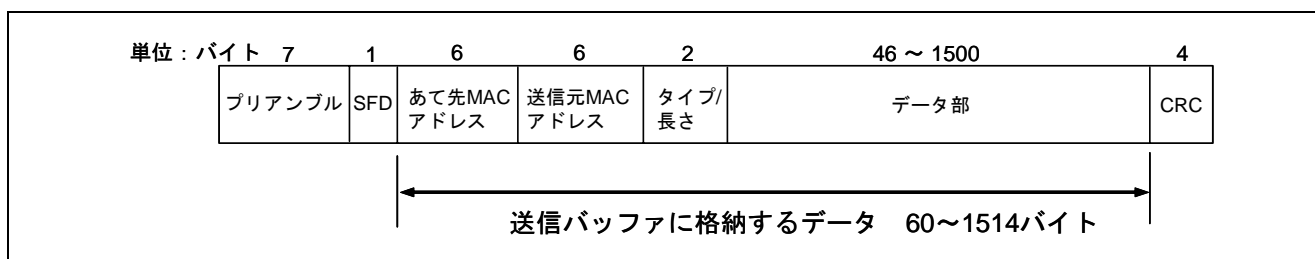


図10 イーサネットフレームフォーマット

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。

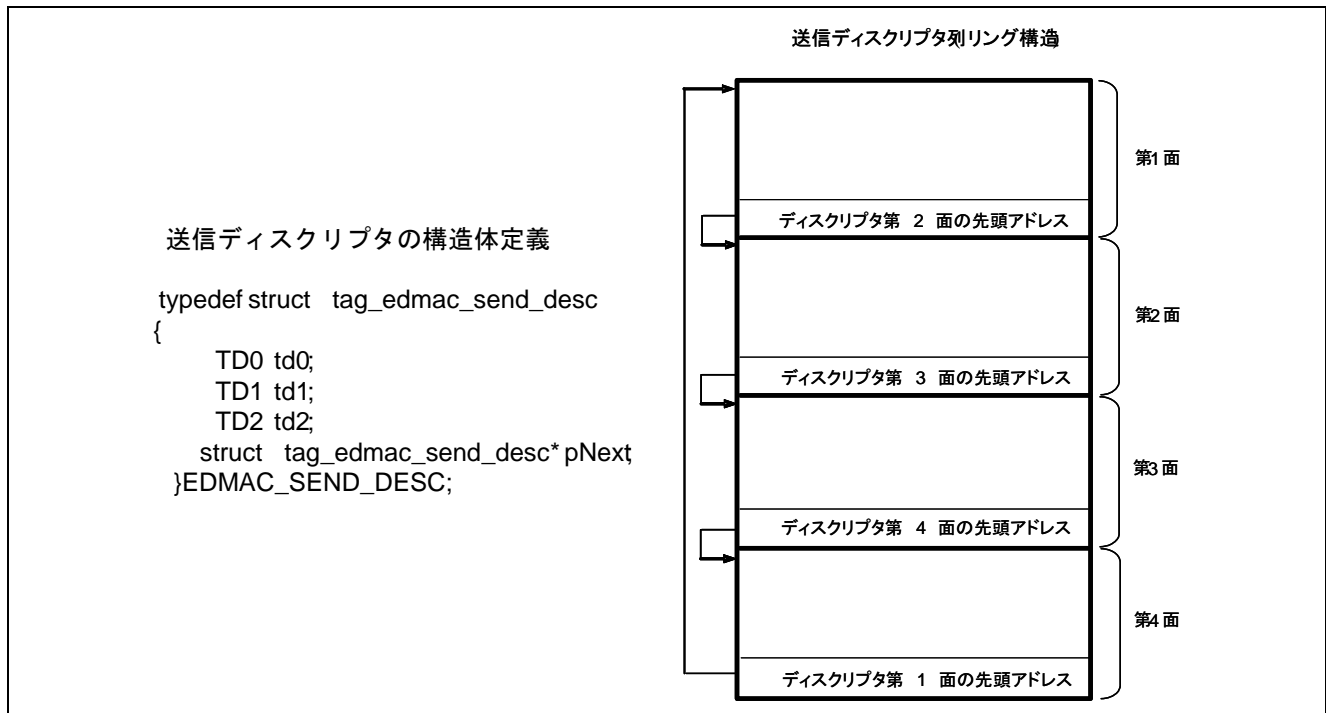


図11 送信ディスクリプタの構造体定義と送信ディスクリプタ列使用例

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、受信処理自体は行っていません。

PHY自動交渉関数phy_autonegoの詳細は「SH7619 グループ アプリケーションノート 内蔵PHYモジュール初期設定例」をご参照ください。

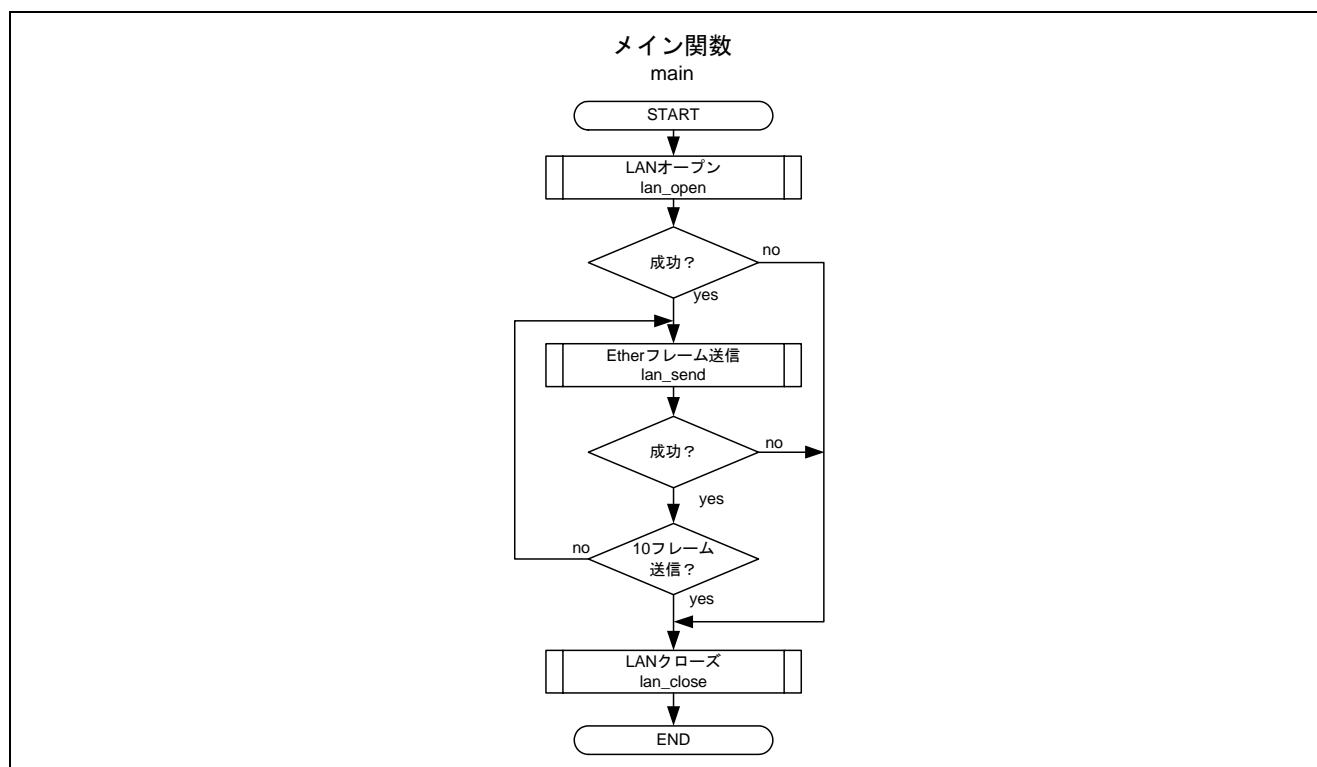


図12 メインプログラムの処理フロー例(1)

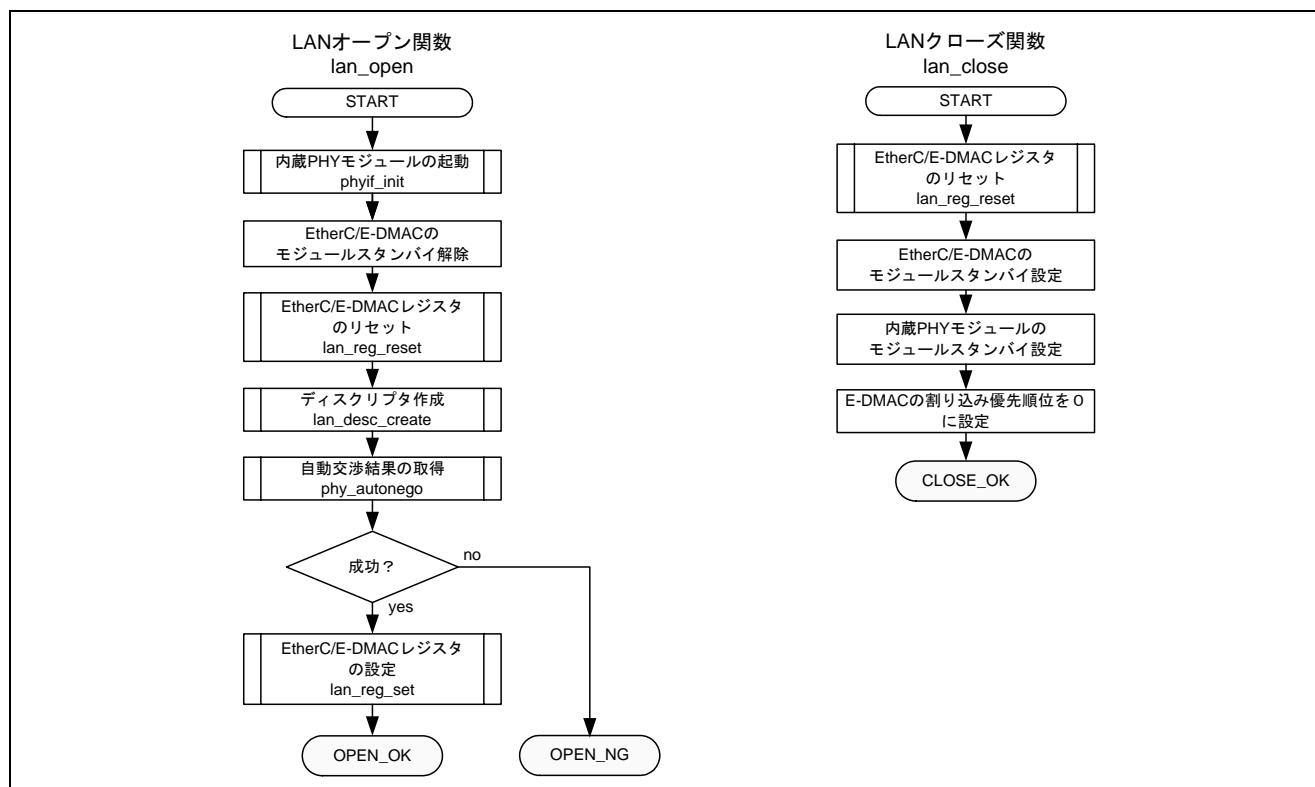


図13 参考プログラムの処理フロー例(2)

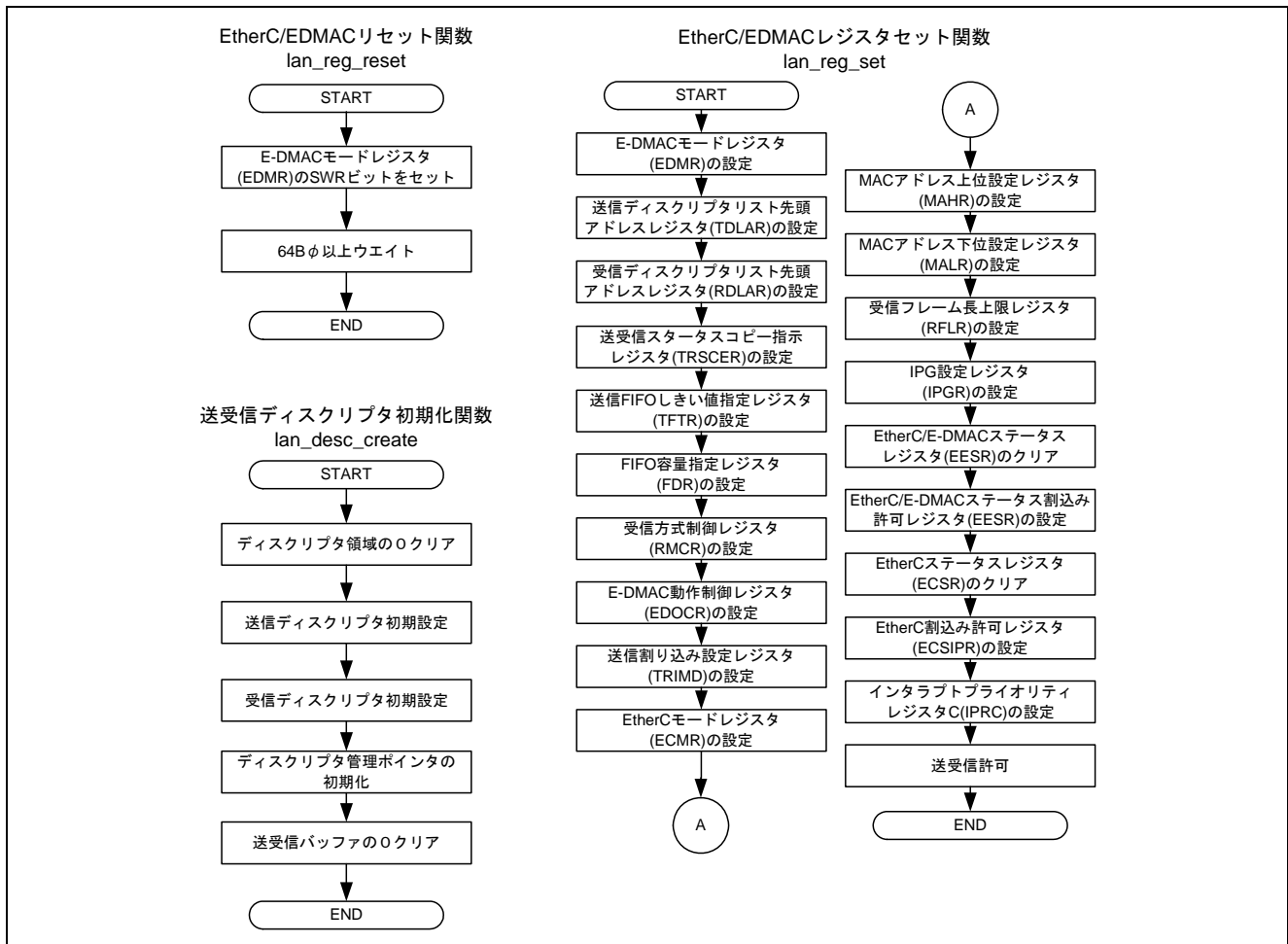


図14 参考プログラムの処理フロー例(3)

イーサネットフレーム送信関数

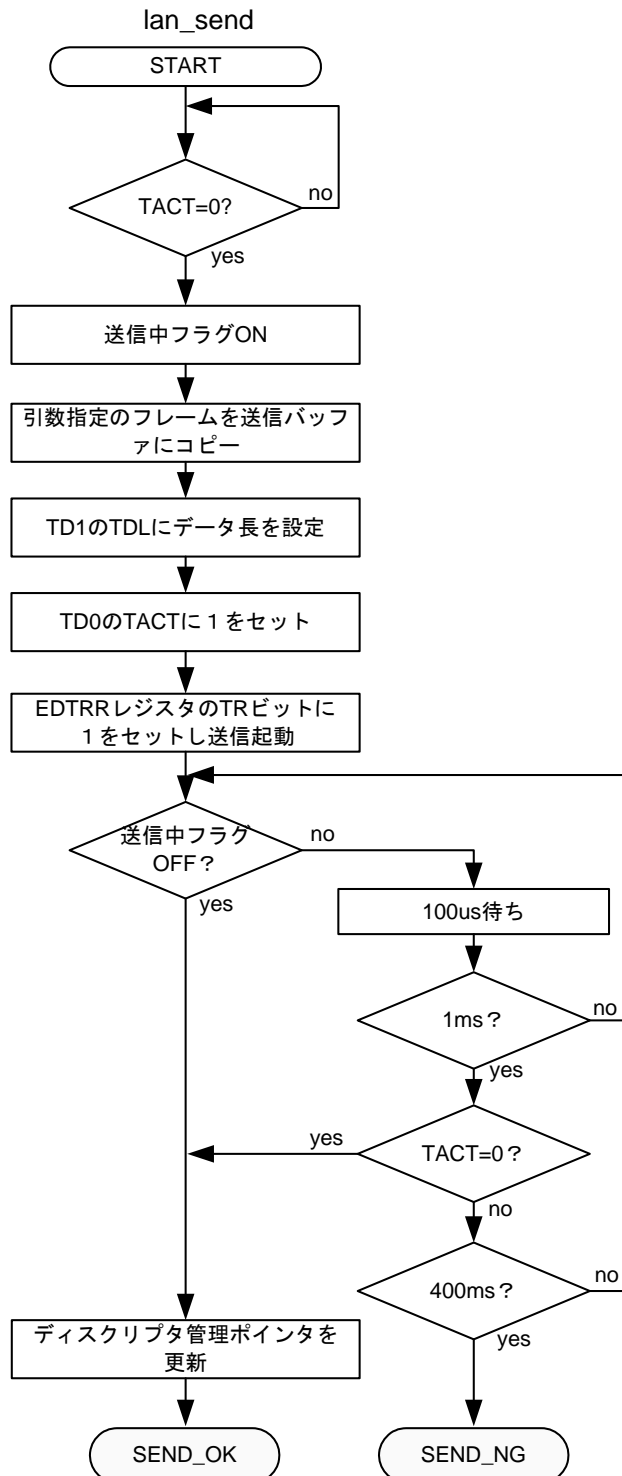


図15 参考プログラムの処理フロー例(4)

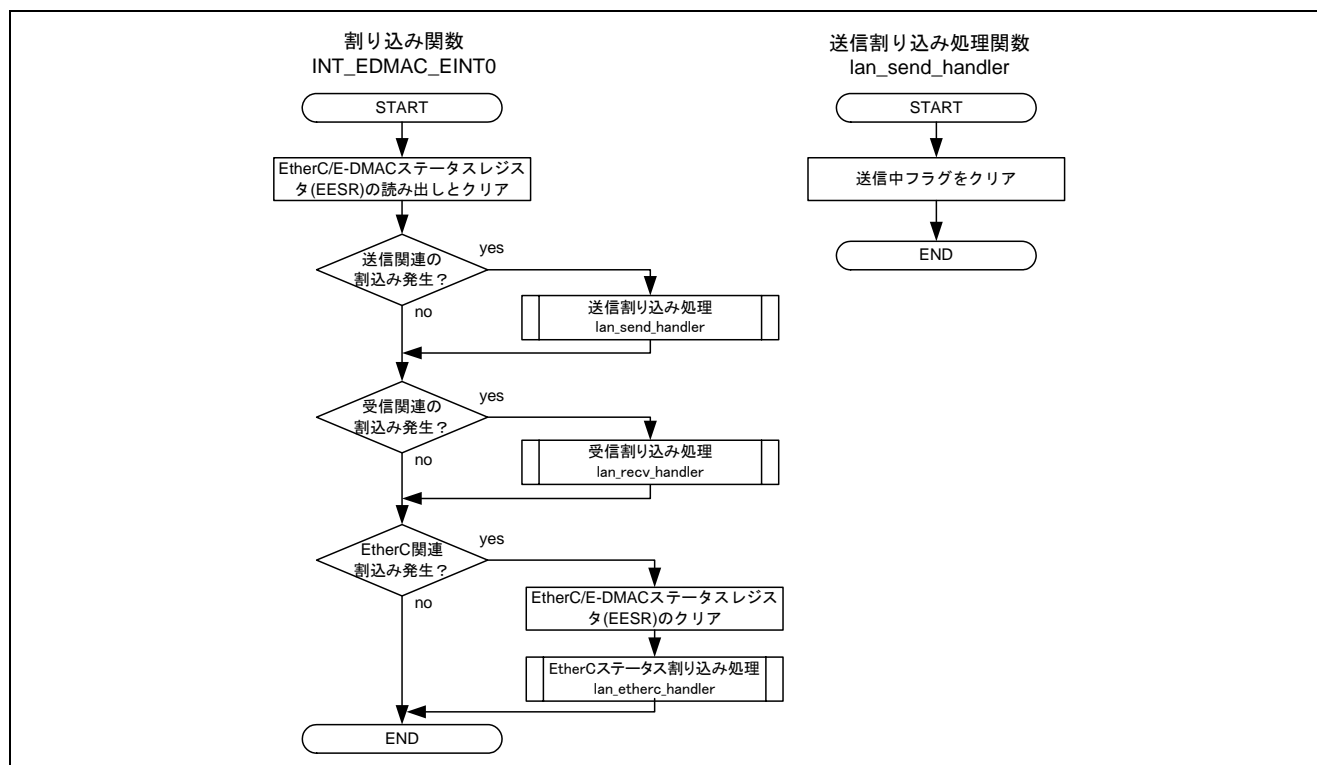


図16 参考プログラムの処理フロー例(5)

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   : SH7619 Sample Program
4  *   File Name    : main.c
5  *   Contents     : イーサネット送信設定例
6  *   Version      : 1.00.00
7  *   Model        : M3A-HS19
8  *   CPU          : SH7619
9  *   Compiler     : SHC9.1.1.0
10 *   note         :
11 *               <注意事項>
12 *               本サンプルプログラムはすべて参考資料であり、
13 *               その動作を保証するものではありません。
14 *               本サンプルプログラムはお客様のソフトウェア開発時の
15 *               技術参考資料としてご利用ください。
16 *
17 *   Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
18 *   and Renesas Solutions Corp. All Rights Reserved
19 *
20 *   history       :2007.07.03 ver.1.00.00
21 *"FILE COMMENT END"*****/
22 #include "iodefine.h"
23 #include "defs.h"
24 #include "ether.h"
25
26 /* **** プロトタイプ宣言 **** */
27 void main(void);
28
29 /* **** 変数宣言 **** */
30 static unsigned char s_frame[] = {
31     0xff,0xff,0xff,0xff,0xff,0xff, /* あて先 MAC アドレス */
32     0x00,0x01,0x02,0x03,0x04,0x05, /* 送信元 MAC アドレス (00:01:02:03:04:05) */
33     0x08,0x06, /* タイプ(ARP) */
34     0x00,0x01, /* ---H/W タイプ= Ethernet */
35     0x08,0x00, /* ---プロトコルタイプ= IP */
36     0x06,0x04, /* ---HW/プロトコルアドレス長 */
37     0x00,0x01, /* ---OPCODE= request */
38     0x00,0x01,0x02,0x03,0x04,0x05, /* ---送信元 MAC アドレス(00:01:02:03:04:05) */
39     0xc0,0xa8,0x00,0x03, /* ---送信元 IP アドレス(192.168.0.3) */
40     0x00,0x00,0x00,0x00,0x00,0x00, /* ---問合せ先 MAC アドレス */
41     0xc0,0xa8,0x00,0x05, /* ---問合せ先 IP アドレス(192.168.0.5) */
42 };
43

```

3.2 サンプルプログラムリスト”main.c” (2)

```

44  /* "FUNC COMMENT" *****
45  * ID      :
46  * Outline   : イーサネット送信サンプルプログラム メイン関数
47  * -----
48  * Include    : #include "iodefine.h"
49  * -----
50  * Declaration : void main(void)
51  * -----
52  * Function    : 内蔵イーサネットコントローラ (EtherC) とイーサネットコント
53  *              : 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
54  *              : 使用して、イーサネットフレームを送信します。
55  *              : また PHY モジュールには、SH7619 内蔵 PHY モジュールを使用します。
56  *              : 送信ディスクリプタを複数面用意して連続送信を行っています。
57  * -----
58  * Argument    : void
59  * -----
60  * ReturnValue : void
61  * -----
62  * Notice      :
63  * "FUNC COMMENT END" *****/
64  void main(void)
65  {
66      int i;
67      int ret;
68
69      /* ==== イーサネット初期設定 ==== */
70      ret = lan_open();
71      if( ret == OPEN_OK ){
72          /* ==== 10 フレーム送信開始 ==== */
73          for(i=0; i<10; i++){
74              /* ---- 送信 ---- */
75              ret = lan_send( s_frame, sizeof(s_frame) );
76              if( ret != SEND_OK ){
77                  break;
78              }
79          }
80      }
81      /* ==== イーサネット送受信停止 ==== */
82      lan_close();
83  }
84  /* 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   : SH7619 Sample Program
4  *   File Name    : ether.c
5  *   Contents     : イーサネット送信設定例
6  *   Version      : 1.01.021
7  *   Model        : M3A-HS19
8  *   CPU          : SH7619
9  *   Compiler     : SHC9.1.1.0
10 *   note         : イーサネット送信に必要な初期設定および送信処理を記述します。
11 *
12 *               <注意事項>
13 *               本サンプルプログラムはすべて参考資料であり、
14 *               その動作を保証するものではありません。
15 *               本サンプルプログラムはお客様のソフトウェア開発時の
16 *               技術参考資料としてご利用ください。
17 *
18 *   Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
19 *   and Renesas Solutions Corp. All Rights Reserved
20 *
21 *   history       :2007.07.03 ver.1.00.00
22 *                 :2007.09.28 ver.1.01.00 lan_reg_set() レジスタ設定順序変更
23 *                 lan_close() 割込みマスク追加
24 *                 lan_desc_create() メモリクリア追加
25 *                 :2007.10.22 ver.1.01.01 189 行目 f_send クリアを追加
26 *                 :2007.11.01 ver.1.01.02 lan_send() ウェイト用 for 文の条件式を修正
27 *"FILE COMMENT END"*****/
28 #include "machine.h"
29 #include "string.h"
30 #include "iodefine.h"
31 #include "defs.h"
32 #include "phy.h"
33 #include "ether.h"
34
35 /* **** プロトタイプ宣言 **** */
36 void lan_send_handler( unsigned long status );
37 static void lan_desc_create( void );
38 static void lan_reg_reset( void );
39 static void lan_reg_set( int link );
40
41 /* **** 変数宣言 **** */
42 /* ---- ディスクリプタ ---- */
43 #pragma section ETH_DESC                /* 16 バイト境界へ配置 */
44 static volatile TXRX_DESCRIPTOR_SET desc; /* ディスクリプタ領域 */
45 #pragma section
46 /* ---- バッファ ---- */
47 #pragma section ETH_BUFFER                /* 16 バイト境界へ配置 */
48 static volatile TXRX_BUFFER_SET buf;     /* 送受信バッファ領域 */
49 #pragma section
50 /* ---- MAC アドレス ---- */
51 static const unsigned long my_macaddr_h = 0x00010203;
52 static const unsigned long my_macaddr_l = 0x0405; /* 00:01:02:03:04:05 */
53 /* ---- その他 ---- */
54 static volatile int f_send = 0;          /* 送信中フラグ */
55

```

3.4 サンプルプログラムリスト” ether.c”(2)

```

56  /*"FUNC COMMENT"*****
57  * ID      :
58  * Outline  : イーサネットオープン関数
59  *-----
60  * Include   : #include "iodef.h"
61  *           : #include "phy.h"
62  *           : #include "ether.h"
63  *-----
64  * Declaration : int lan_open(void)
65  *-----
66  * Function    : E-DMAC, EtherC, PHY, バッファメモリの初期化を行う。
67  *              : 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する。
68  *              : 送受信可能な状態に設定することができないときはエラーを返す。
69  *-----
70  * Argument    : void
71  *-----
72  * ReturnValue : OPEN_OK(0) :オープン成功
73  *              : OPEN_NG(-1):オープン失敗
74  *-----
75  * Notice      :
76  /*"FUNC COMMENT END"*****/
77 int lan_open(void)
78 {
79     int link;
80
81     /* ==== 内蔵 PHY モジュール起動 ==== */
82     phyif_init();
83     /* ==== EtherC/EDMAC のモジュールスタンバイ解除 ==== */
84     SBY.CR4.BIT._EtherC = 0;
85     /* ==== EtherC,E-DMAC の停止 ==== */
86     lan_reg_reset();
87     /* ==== バッファメモリの初期化 ==== */
88     lan_desc_create();
89     /* ==== EtherC,E-DMAC の設定 ==== */
90     link = phy_autonego();          /* デュプレックスモード確認 */
91     if( link == NEGO_FAIL ){
92         return OPEN_NG;             /* OPEN 失敗 */
93     }
94     else{
95         lan_reg_set(link);
96     }
97     return OPEN_OK;
98 }

```

3.5 サンプルプログラムリスト” ether.c”(3)

```

99  /*"FUNC COMMENT"*****
100  * ID      :
101  * Outline   : イーサネットクローズ関数
102  *-----
103  * Include    : #include "iodefine.h"
104  *            : #include "ether.h"
105  *-----
106  * Declaration : int lan_close(void)
107  *-----
108  * Function    : EDMAC/EtherC, PHY の動作を停止する。
109  *              : EDMAC/EtherC, PHY はクロックの供給も停止する。
110  *-----
111  * Argument    : void
112  *-----
113  * ReturnValue : int  CLOSE_OK( 0):クローズ成功
114  *              :      CLOSE_NG(-1):クローズ失敗
115  *-----
116  * Notice      :
117  * "FUNC COMMENT END"*****/
118  int lan_close( void )
119  {
120      int i;
121
122      /* ==== EtherC,E-DMAC のリセット ==== */
123      lan_reg_reset();
124      /* ==== EtherC,E-DMAC の停止 ==== */
125      SBY.CR4.BIT._EtherC = 1;
126      /* ==== 内蔵 PHY モジュールの停止 ==== */
127      SBY.CR4.BIT._PHYIF = 1;
128      /* ==== E-DMAC 関連の割り込みを禁止 ==== */
129      INTC.IPRC.BIT._EDMAC = 0;
130
131      return CLOSE_OK;
132  }
133  /*"FUNC COMMENT"*****
134  * ID      :
135  * Outline   : フレーム送信関数
136  *-----
137  * Include    : #include "ether.h"
138  *            : #include "iodefine.h"
139  *-----
140  * Declaration : int lan_send( unsigned char *addr, int flen )
141  *-----
142  * Function    : 指定されたフレームを送信ディスクリプタに登録されたバッファに
143  *              : コピーして送信します。送信が完了するまで本関数内で待ちます。
144  *              : EESR レジスタの割り込み要求ビットがセットされない現象と、送信 FIFO
145  *              : アンダフロー時の EDMAC 動作停止に対する対応を実施しています。
146  *-----
147  * Argument    : なし
148  *-----
149  * ReturnValue : SEND_OK(0) : 登録成功
150  *              : SEND_NG(-1): 登録失敗
151  *-----
152  * Notice      :
153  * "FUNC COMMENT END"*****/

```


3.6 サンプルプログラムリスト” ether.c”(4)

```

154 int lan_send( unsigned char *addr, int flen )
155 {
156     int i;
157     int tlms = 0;
158     int t400ms = 0;
159
160     /* ==== 送信中ではないことを確認 ==== */
161     while( desc.pSend_top->td0.BIT.TACT == 1 ){
162         /* wait */
163     }
164     /* ==== 送信中フラグをセット ==== */
165     f_send = 1;                                /* 送信完了割込みでクリア */
166
167     /* ==== 送信ディスクリプタの更新 ==== */
168     memcpy( desc.pSend_top->td2.TBA, addr, flen ); /* 送信データ */
169     if( flen < 60 ){                             /* 最小フレーム 60 バイト */
170         memcpy( (desc.pSend_top->td2.TBA)+flen, 0, 60-flen ); /*パディング*/
171         flen = 60;
172     }
173     desc.pSend_top->td1.TDL = flen;                /* データ長 */
174     desc.pSend_top->td0.BIT.TACT = 1;              /* 送信許可 */
175
176     /* ==== 送信停止中であれば起動 ==== */
177     if( EDMAC.EDTRR.BIT.TR == 0 ){                /* 必ずリードして確認する */
178         EDMAC.EDTRR.BIT.TR = 1;
179     }
180     /* ==== 送信完了の確認 ==== */
181     while( f_send ){
182         for( i=LOOP_100us; i>0; i-- ){
183             /* 100us wait */
184         }
185         /* ---- 1ms 経過時はディスクリプタも確認 ---- */
186         /* フレーム送信完了ビットが消失した場合への対応 */
187         if( ++tlms > 10 ){
188             tlms = 0;
189             if( desc.pSend_top->td0.BIT.TACT == 0 ){
190                 f_send = 0;
191                 break;
192             }
193         }
194         /* ---- 400ms 経過したら EDMAC 動作停止と判断 ---- */
195         /* 送信 FIFO アンダフロー発生時の EDMAC 動作停止への対応 */
196         if( ++t400ms > 4000 ){
197             t400ms = 0;
198             return SEND_NG;
199         }
200     }
201     /* ==== カレントポインタの更新 ==== */
202     desc.pSend_top = desc.pSend_top->pNext;
203
204     return SEND_OK;
205 }
206

```

3.7 サンプルプログラムリスト” ether.c”(5)

```

207  /* "FUNC COMMENT" *****
208  * ID      :
209  * Outline  : ディスクリプタ構築関数
210  *-----
211  * Include  : #include "ether.h"
212  *-----
213  * Declaration : static void lan_desc_create( void )
214  *-----
215  * Function   : イーサネットで必要な送受信バッファとディスクリプタの初期化を
216  *             : 行います。1 フレーム/1 バッファを前提としています。
217  *-----
218  * Argument   : void
219  *-----
220  * ReturnValue : void
221  *-----
222  * Notice     :
223  * "FUNC COMMENT END" *****/
224  static void lan_desc_create( void )
225  {
226      int i;
227      /* ==== ディスクリプタ領域の構築 ==== */
228      /* ---- メモリクリア ---- */
229      memset(&desc, 0, sizeof(desc) );
230      /* ---- 送信ディスクリプタ ---- */
231      for(i=0; i<NUM_OF_TX_DESCRIPTOR; i++){
232          desc.send[i].td2.TBA = buf.send[i];          /* TD2 */
233          desc.send[i].td1.TDL = 0;                    /* TD1 */
234          desc.send[i].td0.LONG= 0x30000000;          /* TD0:1frame/1buf、送信無効*/
235          if( i != (NUM_OF_TX_DESCRIPTOR-1) ){ /* pNext */
236              desc.send[i].pNext = &desc.send[i+1];
237          }
238      }
239      desc.send[i-1].td0.BIT.TDLE = 1;
240      desc.send[i-1].pNext = &desc.send[0];
241      /* ==== 受信ディスクリプタ ==== */
242      for(i=0; i<NUM_OF_RX_DESCRIPTOR; i++){
243          desc.recv[i].rd2.RBA = buf.recv[i];          /* RD2 */
244          desc.recv[i].rd1.RBL = SIZE_OF_BUFFER;       /* RD1 */
245          desc.recv[i].rd0.LONG= 0xb0000000;          /* RD0:1frame/1buf、受信有効*/
246          if( i != (NUM_OF_RX_DESCRIPTOR-1) ){ /* pNext */
247              desc.recv[i].pNext = &desc.recv[i+1];
248          }
249      }
250      desc.recv[i-1].rd0.BIT.RDLE = 1;                  /* 最終ディスクリプタの設定 */
251      desc.recv[i-1].pNext = &desc.recv[0];
252
253      /* ---- ディスクリプタ管理情報の初期化 ---- */
254      desc.pSend_top = &desc.send[0];
255      desc.pRecv_end = &desc.recv[0];
256
257      /* ==== バッファ領域の構築 ==== */
258      /* ---- 領域クリア ---- */
259      memset(&buf, 0, sizeof(buf) );
260  }

```

3.8 サンプルプログラムリスト” ether.c”(6)

```

261  /* "FUNC COMMENT" *****
262  * ID      :
263  * Outline  : EtherC,E-DMAC レジスタ初期化関数
264  *-----
265  * Include  : #include "iodefine.h"
266  *-----
267  * Declaration : static void lan_reg_reset( void )
268  *-----
269  * Function   : EtherC と E-DMAC のレジスタをリセットします。
270  *             : 関数内でリセット期間を Bφ 64 サイクル以上確保しています。
271  *-----
272  * Argument   : void
273  *-----
274  * ReturnValue : void
275  *-----
276  * Notice     :
277  * "FUNC COMMENT END" *****/
278  static void lan_reg_reset( void )
279  {
280     volatile int j = 100;          /* Bφ 64 サイクルウェイト */
281
282     /* ---- ソフトウェアリセット ---- */
283     EDMAC.EDMR.BIT.SWR = 1;
284
285     /* ---- リセット期間の確保 ---- */
286     while(j--){
287         /* Bφ 64 サイクル ウェイト */
288     }
289 }
290 /* "FUNC COMMENT" *****
291 * ID      :
292 * Outline  : EtherC,E-DMAC レジスタ設定
293 *-----
294 * Include  : #include "iodefine.h"
295 *           : #include "phy.h"
296 *           : #include "ether.h"
297 *-----
298 * Declaration : void lan_reg_set(int link)
299 *-----
300 * Function    : E-DMAC, EtherC の初期化を行います。
301 *             : 送受信ともに有効に設定しています。
302 *-----
303 * Argument    : int link : I :EtherCに設定するデュプレックスモード。
304 *             :           :phy_autonego 関数の戻り値を利用。
305 *-----
306 * ReturnValue : void
307 *-----
308 * Notice      : 本関数は EDMAC ソフトウェアリセット後の送受信停止状態で実行ください。
309 * "FUNC COMMENT END" *****/

```

3.9 サンプルプログラムリスト” ether.c”(7)

```

310 static void lan_reg_set( int link )
311 {
312     /* ==== EDMAC ==== */
313     EDMAC.EDMR.LONG = 0x00000000; /* エンディアン変換なし(ビッグエンディアン) */
314                                     /* ディスクリプタ長 16 バイト */
315     EDMAC.TDLAR      = &desc.send[0]; /* 送信ディスクリプタ先頭 */
316     EDMAC.RDLAR      = &desc.recv[0]; /* 受信ディスクリプタ先頭 */
317     EDMAC.TRSCER.LONG = 0x00000000; /* 全ステータスをディスクリプタにコピー */
318     EDMAC.TFTR       = 0x00;        /* 送信 FIFO しきい値(ストア&フォワード) */
319     EDMAC.FDR.BIT.TFD = 1;          /* 送信 FIFO 容量(512 バイト) */
320     EDMAC.FDR.BIT.RFD = 1;          /* 受信 FIFO 容量(512 バイト) */
321     EDMAC.RCR.BIT.RNC = 1;          /* 連続受信可能 */
322     EDMAC.EDOCR.LONG = 0x00000000; /* FIFO エラー時も動作継続 */
323     EDMAC.FCFTR.LONG = 0x00070000; /* フロー制御しきい値設定 ※EtherC で禁止 */
324     EDMAC.TRIMD.BIT.TIS = 0;        /* ライトバック完了通知をしない */
325     /* ==== EtherC ==== */
326     EtherC.ECMR.LONG = 0x00000000; /* フロー制御無効 */
327                                     /* CRC フレームを「エラーあり」と認識 */
328                                     /* Magic Packet 検出を許可しない */
329                                     /* 受信禁止 */
330                                     /* 送信禁止 */
331                                     /* 内部ループバックを行わない */
332                                     /* 外部ループバックを行わない */
333                                     /* デュプレックスモード(半二重) */
334                                     /* プロミスキヤスモードを行わない */
335     if( link == FULL_TX || link == FULL_10M ){
336         EtherC.ECMR.BIT.DM = 1;      /* 全二重モードに設定 */
337     }
338     EtherC.MAHR      = my_macaddr_h; /* MAC アドレスの設定 */
339     EtherC.MALR      = my_macaddr_l;
340     EtherC.RFLR      = 0x000;        /* 最大受信フレーム長(1518 バイト) */
341     EtherC.IPGR       = 0x14;        /* パケット間ギャップ(96 ビット時間) */
342     /* ==== 割り込み関連 ==== */
343     EDMAC.EESR.LONG = 0x47FF0F9F; /* 全ステータスクリア(1 ライトでクリア) */
344     EDMAC.EESIPR.LONG = EDMAC_EESIPR_INI_SEND | EDMAC_EESIPR_INI_RECV |
EDMAC_EESIPR_INI_EtherC;
345                                     /* 送受信+EtherC の割り込みを許可 */
346     EtherC.ECSR.LONG = 0x00000017; /* 全ステータスクリア(1 ライトでクリア) */
347     EtherC.ECSIPR.LONG = EtherC_ECSIPR_INI; /* 割り込み許可 */
348     INTC.IPRC.BIT._EDMAC= 5;        /* E-DMAC(EINT0)割り込み優先順位 5 */
349     /* ==== 送受信の許可設定 ==== */
350     /* ---- EtherC ---- */
351     EtherC.ECMR.BIT.RE = 1;          /* 受信機能有効 */
352     EtherC.ECMR.BIT.TE = 1;          /* 送信機能有効 */
353     /* ---- E-DMAC ---- */
354     if( EDMAC.EDRRR.BIT.RR == 0 ){
355         EDMAC.EDRRR.BIT.RR = 0;      /* 受信無効 */
356     }
357 }

```

3.10 サンプルプログラムリスト” ether.c”(8)

```

358  /* "FUNC COMMENT" *****
359  * ID      :
360  * Outline  : 送信割込み関数
361  *-----
362  * Include  : #include "iodefine.h"
363  *          : #include "ether.h"
364  *-----
365  * Declaration : void lan_send_handler( unsigned long status )
366  *-----
367  * Function    : EDMAC(EESR)に関する送信関連の割込みハンドラ。
368  *          :
369  *-----
370  * Argument    : unsigned long status : I : EESR の状態 (割込み許可したビットのみ)
371  *-----
372  * ReturnValue  : なし
373  *-----
374  * Notice      :
375  * "FUNC COMMENT END" ***** /
376 void lan_send_handler( unsigned long status )
377 {
378     /* ==== 送信中フラグをクリア ==== */
379     f_send = 0;
380 }
381 /* "FUNC COMMENT" *****
382 * ID      :
383 * Outline  : 受信割込み関数
384 *-----
385 * Include  : #include "iodefine.h"
386 *          : #include "ether.h"
387 *-----
388 * Declaration : void lan_rcv_handler( unsigned long status )
389 *-----
390 * Function    : EDMAC(EESR)に関する受信関連の割込みハンドラ。
391 *          :
392 *-----
393 * Argument    : unsigned long status : I : EESR の状態 (割込み許可したビットのみ)
394 *-----
395 * ReturnValue  : なし
396 *-----
397 * Notice      : 本サンプルでは何も行いません。
398 * "FUNC COMMENT END" ***** /
399 void lan_rcv_handler( unsigned long status )
400 {
401 }

```

3.11 サンプルプログラムリスト” ether.c”(9)

```

402  /* "FUNC COMMENT" *****
403  * ID      :
404  * Outline : EtherC 割込み関数
405  *-----
406  * Include : #include "iodefine.h"
407  *         : #include "ether.h"
408  *-----
409  * Declaration : void lan_etherc_handler( unsigned long status )
410  *-----
411  * Function    : EtherC(ECSR)に関する割込みハンドラ。
412  *             :
413  *-----
414  * Argument    : unsigned long status : I : ECSR の状態 (割込み許可したビットのみ)
415  *-----
416  * ReturnValue : なし
417  *-----
418  * Notice     : 本サンプルでは何も行いません。
419  * "FUNC COMMENT END" ***** /
420  void lan_etherc_handler( unsigned long status )
421  {
422  }
423
424  /* End of file */

```

3.12 サンプルプログラムリスト"ether.h"(1)

```

1  /*"FILE COMMENT"*****
2  *
3  *   System Name   : SH7619 Sample Program
4  *   File Name    : ether.h
5  *   Contents     : イーサネット送信設定例
6  *   Version      : 1.00.00
7  *   Model        : M3A-HS19
8  *   CPU          : SH7619
9  *   Compiler     : SHC9.1.1.0
10 *   note         :
11 *               <注意事項>
12 *               本サンプルプログラムはすべて参考資料であり、
13 *               その動作を保証するものではありません。
14 *               本サンプルプログラムはお客様のソフトウェア開発時の
15 *               技術参考資料としてご利用ください。
16 *
17 *   Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
18 *   and Renesas Solutions Corp. All Rights Reserved
19 *
20 *   history       :2007.07.03 ver.1.00.00
21 *"FILE COMMENT END"*****
22 #ifndef _ETHER_H
23 #define _ETHER_H
24
25 /* **** マクロ定義 **** */
26 #define NUM_OF_TX_DESCRIPTOR 4
27 #define NUM_OF_RX_DESCRIPTOR 4
28 #define NUM_OF_TX_BUFFER    4
29 #define NUM_OF_RX_BUFFER    4
30 #define SIZE_OF_BUFFER      1520 /* 16 バイトの整数倍にすること */
31
32 #define OPEN_OK              0
33 #define OPEN_NG              -1
34 #define SEND_OK              0
35 #define SEND_NG              -1
36 #define CLOSE_OK            0
37 #define CLOSE_NG            -1
38 #define MIN_FRAME_SIZE      60
39 #define MAX_FRAME_SIZE      1514
40
41 #define EDMAC_EESIPR_INI_SEND 0x04280F00 /* 0x04000000 : 送信中断検出      *
42                                         * 0x00200000 : フレーム送信完了  *
43                                         * 0x00080000 : 送信 FIFO アンダフロー*
44                                         * 0x00000800 : キャリア未検出    *
45                                         * 0x00000400 : キャリア消失検出  *
46                                         * 0x00000200 : 遅延衝突検出      *
47                                         * 0x00000100 : 送信リトライオーバ */
48 #define EDMAC_EESIPR_INI_RECV 0x0205001F /* 0x02000000 : 受信中断検出      *
49                                         * 0x00040000 : フレーム受信      *
50                                         * 0x00010000 : 受信 FIFO オーバフロー *
51                                         * 0x00000010 : 端数ビットフレーム受信 *
52                                         * 0x00000008 : ロングフレーム受信  *
53                                         * 0x00000004 : ショートフレーム受信 *
54                                         * 0x00000002 : PHY-LSI 受信エラー  *
55                                         * 0x00000001 : 受信フレーム CRC エラー */

```

3.13 サンプルプログラムリスト” ether.h”(2)

```

56  #define EDMAC_EESIPR_INI_EtherC 0x00400000      /* 0x00400000 : EtherC ステータスレジスタ*/
57  #define EtherC_ECSIPR_INI      0x00000004      /* 0x00000004 : リンク信号変化 */
58
59  /* **** 型定義 **** */
60
61  /* ==== 送信ディスクリプタ ==== */
62  typedef union{
63      unsigned long LONG;
64      struct{
65          unsigned int TACT:1;          /* 送信ディスクリプタ有効          */
66          unsigned int TDLE:1;          /* 送信ディスクリプタ最終          */
67          unsigned int TFP :2;          /* 送信フレーム内位置 1,0          */
68          unsigned int TFE :1;          /* 送信フレームエラー              */
69          unsigned int reserved :23;    /* TFS26~4 : 予約                  */
70          unsigned int TFS3:1;          /* ノーキャリア検出 (EESR-CND ビット) */
71          unsigned int TFS2:1;          /* キャリア消失検出 (EESR-DLC ビット) */
72          unsigned int TFS1:1;          /* 送信中の遅延衝突検出 (EESR-CD ビット) */
73          unsigned int TFS0:1;          /* 送信リトライオーバー (EESR-TRO ビット) */
74      }BIT;
75  }TD0;
76  typedef struct{
77      unsigned short TDL;              /* 送信バッファデータ長          */
78      unsigned short reserved;
79  }TD1;
80  typedef struct{
81      unsigned char *TBA;              /* 送信バッファのアドレス        */
82  }TD2;
83  typedef struct tag_edmac_send_desc{
84      TD0 td0;
85      TD1 td1;
86      TD2 td2;
87      struct tag_edmac_send_desc *pNext;
88  }EDMAC_SEND_DESC;
89
90  /* ==== 受信ディスクリプタ ==== */
91  typedef union{
92      unsigned long LONG;
93      struct{
94          unsigned int RACT:1;          /* 受信ディスクリプタ有効          */
95          unsigned int RDLE:1;          /* 受信ディスクリプタ最終          */
96          unsigned int RFP :2;          /* 受信フレーム内位置 1,0          */
97          unsigned int RFE :1;          /* 受信フレームエラー              */
98          unsigned int reserved1:17;    /* TFS26~10 : 予約                  */
99          unsigned int RFS9:1;          /* 受信 FIFO オーバフロー (EESR-RFOF ビット) */
100         unsigned int reserved2:1;      /*          : 予約                  */
101         unsigned int RFS7:1;          /* マルチキャストフレームを受信 (EESR-RMAF ビット) */
102         unsigned int reserved3:1;      /*          : 予約                  */
103         unsigned int reserved4:1;      /*          : 予約                  */
104         unsigned int RFS4:1;          /* 端数ビットフレーム受信エラー (EESR-RRF ビット) */
105         unsigned int RFS3:1;          /* ロングフレーム受信エラー (EESR-RTLE ビット) */
106         unsigned int RFS2:1;          /* ショートフレーム受信エラー (EESR-RTSF ビット) */
107         unsigned int RFS1:1;          /* PHY-LSI 受信エラー (EESR-PRE ビット) */
108         unsigned int RFS0:1;          /* 受信フレーム CRC エラー検出 (EESR-CERF ビット) */
109     }BIT;
110  }RD0;

```


3.14 サンプルプログラムリスト” ether.h”(3)

```

111  typedef struct{
112      unsigned short RBL;                /* 受信バッファ長                */
113      unsigned short RDL;                /* 受信データ長                */
114  }RD1;
115  typedef struct{
116      unsigned char *RBA;                /* 受信バッファのアドレス        */
117  }RD2;
118  typedef struct tag_edmac_recv_desc{
119      RD0 rd0;
120      RD1 rd1;
121      RD2 rd2;
122      struct tag_edmac_recv_desc *pNext;
123  }EDMAC_RECV_DESC;
124
125  /* ==== 送受信ディスクリプタ全体(16 バイト境界に配置すること) ==== */
126  typedef struct{
127      EDMAC_SEND_DESC send[NUM_OF_TX_DESCRIPTOR];
128      EDMAC_RECV_DESC recv[NUM_OF_RX_DESCRIPTOR];
129      EDMAC_SEND_DESC *pSend_top;        /* 送信ディスクリプタの登録位置 */
130      EDMAC_RECV_DESC *pRecv_end;        /* 受信ディスクリプタの受信完了/登録位置 */
131  }TXRX_DESCRIPTOR_SET;
132
133  /* ==== 送受信バッファ(16 バイト境界に配置すること) ==== */
134  /* ---- 全送受信バッファ領域の定義 ---- */
135  typedef struct{
136      unsigned char send[NUM_OF_TX_BUFFER][SIZE_OF_BUFFER];
137      unsigned char recv[NUM_OF_RX_BUFFER][SIZE_OF_BUFFER];
138  }TXRX_BUFFER_SET;
139
140  /* **** プロトタイプ宣言 **** */
141  int lan_open( void );
142  int lan_close( void );
143  int lan_send( unsigned char *addr, int flen );
144
145
146  #endif
147
148  /* End of File */

```

3.15 サンプルプログラムリスト”intprg_eth.c”(1)

```

1  /*"FILE COMMENT"*****
2  *
3  *      System Name : SH7619 Sample Program
4  *      File Name   : intprg_eth.c
5  *      Version    : 1.01.00
6  *      Contents   : EDMAC/EtherC 送受信設定例 割り込みプログラム
7  *      Model      : M3A-HS19
8  *      CPU        : SH7619
9  *      Compiler   : SHC9.1.1.0
10 *      OS         : none
11 *
12 *      note       : 本ファイルはRenesas Project Generator (Ver.4.06) により
13 *                  生成されたファイルをアプリケーションノート用に変更しています。
14 *
15 *                  <注意事項>
16 *                  本サンプルプログラムはすべて参考資料であり、
17 *                  その動作を保証するものではありません。
18 *                  本サンプルプログラムはお客様のソフトウェア開発時の
19 *                  技術参考資料としてご利用ください。
20 *
21 *      Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
22 *      AND Renesas Solutions Corp. All Rights Reserved
23 *
24 *      history    : 2007.07.03 ver.1.00.00
25 *                  : 2007.09.28 ver.1.01.00 ベクタ番号 4-31 を RAM 転送対象外とする
26 * "FILE COMMENT END"***** /
27 #include <machine.h>
28 #include "vect.h"
29 #include <stdio.h>
30 #include "iodefine.h"
31 #include "defs.h"
32 #include "ether.h"

```

(途中省略)

3.16 サンプルプログラムリスト”intprg_eth.c”(2)

```

200 // 85 EDMAC EINT0
201 extern void lan_send_handler( unsigned long status);
202 extern void lan_recv_handler( unsigned long status);
203 extern void lan_etherc_handler( unsigned long status);
204
205 void INT_EDMAC_EINT0(void){
206     unsigned long stat_edmac;
207     unsigned long stat_etherc;
208
209     /* ---- 割込み要求フラグのクリア ---- */
210     stat_edmac = EDMAC.EESR.LONG & EDMAC.EESIPR.LONG; /* 許可した割込みのみ対象 */
211     EDMAC.EESR.LONG = stat_edmac;
212     /* ==== 送信関連 ==== */
213     if( stat_edmac & EDMAC.EESIPR.INI_SEND ){
214         lan_send_handler( stat_edmac & EDMAC.EESIPR.INI_SEND );
215     }
216     /* ==== 受信関連 ==== */
217     if( stat_edmac & EDMAC.EESIPR.INI_RECV ){
218         lan_recv_handler( stat_edmac & EDMAC.EESIPR.INI_RECV );
219     }
220     /* ==== EtherC 関連 ==== */
221     if( stat_edmac & EDMAC.EESIPR.INI_EtherC ){
222         /* ---- 割込み要求フラグのクリア ---- */
223         stat_etherc = EtherC.ECSR.LONG & EtherC.ECSIPR.LONG; /* 許可した割込みのみ対象 */
224         EtherC.ECSR.LONG = stat_etherc;
225         lan_etherc_handler( stat_etherc );
226     }
227 }

```

(以下省略)

4. 参考ドキュメント

- ソフトウェアマニュアル
SH-1/SH-2/ SH-DSP ソフトウェアマニュアル Rev7.00
(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
SH7619 グループハードウェアマニュアル Rev.5.00
(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)。

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/>
- お問合せ先
<http://japan.renesas.com/inquiry>
csc@renesas.com

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2007.12.28	—	新規作成

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を負いません。
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（含むハードウェアおよびソフトウェア）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。