HI.CommunicationEngine
PPPサーバ
リファレンスマニュアル

株式会社 ルネサス北日本セミコンダクタ

ご注意

1. 本製品(ソフトウエア製品及びその関連ソフトウエア製品を含む。以下、同じ。)の使用に際しては、「外国為替及び外国貿易法」等、技術輸出に関する日本及び関連諸国の関係法規の遵守が必要となります。
2. 弊社は、本製品の使用に際しては、弊社もしくは第三者の特許権、著作権、商標権、その他の知的所有権等の権利に関し、別途、個別の契約書等(マニュアルの記載を含む。以下、同じ。)にて弊社による明示的な許諾がある場合を除き、その保証または実施権の許諾を行うものではありません。また本製品を使用したことにより第三者の知的所有権等の権利に関わる問題が生じた場合、弊社はその責を負いませんので予めご了承ください。
3. 本製品およびその仕様、またはマニュアルに記載されている事柄については、将来、事前の予告なしに変更することがありますので、最終的な設計、ご購入、ご使用に際しましては、事前に最新の製品規格または仕様書(マニュアルを含む)をご確認ください。
4. 本製品の使用(マニュアル記載事項に基づくものも含む)により直接または間接に生ずるいかなる損害についても、弊社は一切の責任を負いません。また、本製品の配布に使用される搭載機器や媒体が原因の損害に対しましても、弊社は一切の責任を負いません。
5. 本製品を、宇宙、航空、原子力、燃焼制御、運輸、交通、各種安全装置、ライフサポート関連の医療機器等のように、特別な品質・信頼性が要求され、その故障や誤動作が直接人命を脅かしたり、人体に危害を及ぼす恐れのある用途向けには使用できません。お客様の用途がこれに該当するかどうか疑問のある場合には、事前に弊社営業担当迄ご相談をお願い致します。
6. 本製品を使用してお客様のシステム製品を設計される際には、通常予測される故障発生率、故障モードをご考慮の上、本製品の動作が原因での事故、その他の拡大損害を生じないようにフェールセーフ等の充分なシステム上の対策を講じて頂きますようお願い致します。
7. 本製品およびマニュアルの著作権は弊社が所有しております。お客様は、弊社から提供された本製品を、別途、個別の契約書等にて定める場合を除き、いかなる場合においても全体的または部分的に複写・解析・改変することはできないものとします。
8. お客様は、別途、個別の契約書等にて定める場合を除き、本製品のマニュアルの一部または全部を無断で使用、複製することはできません。
9. 弊社は、本製品を1台のコンピュータで使用する権利をお客様に対してのみ許諾します。よって、本製品を第三者へ譲渡、貸与、賃借することは許諾しないものとします。但し、別途、個別の契約書等にて定められる場合はその条件に従います。
10. 本製品をはじめ弊社製品およびその関連製品についてのお問い合わせ、ご相談は弊社営業担当迄お願い致します。

はじめに

　このマニュアルは、HI.CommunicationEngine TCP/IPマネージャ上で動作するポイントーポイント間プロトコルのドライバ「PPPサーバ」について説明します。
　HI.CommunicationEngine PPPサーバはユーザが準備する物理ドライバを経由し、接続されたPPPクライアントの認証と、1500バイトまでのパケットの送受信を行う機能を提供します。
　このリファレンスマニュアルではPPPサーバのサービスコールとその使い方および関連事項を説明します。TCP/IPマネージャについては関連マニュアルを参照してください。

【関連マニュアル】
・HI.CommunicationEngine TCP/IPマネージャ リファレンスマニュアル
・使用するμITRON のユーザーズマニュアル

目次

図表目次

1. 概要

1.1 機能

PPPサーバは、ユーザが別途準備した物理ドライバプログラムを制御し、ハードウェア上の物理デバイス等を介して、PPPクライアントと通信します。

PPPサーバの基本機能は次のとおりです。

（1） 認証手順を経てPPPクライアントを接続します。

（2） TCP/IPマネージャから渡されたIPパケットをPPPに変換して物理ドライバに渡します。また、物理ドライバから渡されたPPPのパケットをIPパケットに変換してTCP/IPマネージャに渡します。

（3） 次の機能を実装しています。

- LCP（Link Control Protocol）を実装しています[*1]。
- 認証プロトコルとして、PAP および CHAP を実装しています。
- IPCPを実装しています[*1]。
- アドレスおよび制御フィールド圧縮を実装しています。
- プロトコルフィールド圧縮を実装しています。
- マルチプロトコルデータグラム（IP のみ）を実装しています。
- MRU 最大 bytes を設定できます（最大 1500 まで）。
- VJ 圧縮（IP ヘッダ圧縮）を実装しています。
- マルチセッションに対応しています。（最大 10 セッション）
- 経路設定によりセッション毎に送信パケットの振り分けができます。

本PPPサーバでは、次の機能は実装していません。

- **CCP 等の圧縮プロトコル**は実装していません。
- **IPXCP** は実装していません。
- **LQR 品質制御**は実装していません。
- **CBCP**（コールバックコントロールプロトコル）は実装していません。
- **MP**（マルチリンクプロトコル）は実装していません。

その他、1.1(3)に記載された機能以外は未サポートです。

PPP サーバには、シリアルコントローラ及びモデムの制御は含んでいません。モデムと接続して使用するためには、AT コマンドを使用したモデムの制御プログラムやモデム制御用のシリアルドライバプログラム等を別途作成する必要があります。

*1 サポートしているオプション項目については「付録A. サポートオプション」を参照してください。

1.2 関連するドライバ

PPPサーバは物理ドライバプログラムを介してデータ通信を行います。物理ドライバプログラムは使用する物理デバイスにあわせて別途作成する必要があります。また、PPPサーバがマルチセッションに対応するには、マルチセッションに対応した物理ドライバプログラムが必要となります。

物理ドライバプログラムの例としてSolutionEngine搭載のSuperI/O内蔵シリアル用ドライバプログラムをサンプルとして提供しております（サンプルのシリアルドライバプログラムにはモデム制御機能はありません）。

PPPクライアントが接続後、認証が完了した後のPPPサーバはTCP/IPマネージャのドライバモジュールとして動作します。作成する物理ドライバプログラムとTCP/IPマネージャ間のインタフェースについては、サンプルのシリアルドライバプログラムを参考にしてください。

1.3　構成

PPP サーバ使用時の構成例を図 1-1 に示します。

PPP サーバを使用する為には、シリアルドライバ等の物理ドライバプログラムの準備が必要ですが、モデムを使用するためには、モデムの制御（AT コマンド、制御線コントロール）についても別途準備する必要があります。

TCP/IP マネージャでは PPP サーバを 1 つのドライバモジュールとして管理します。

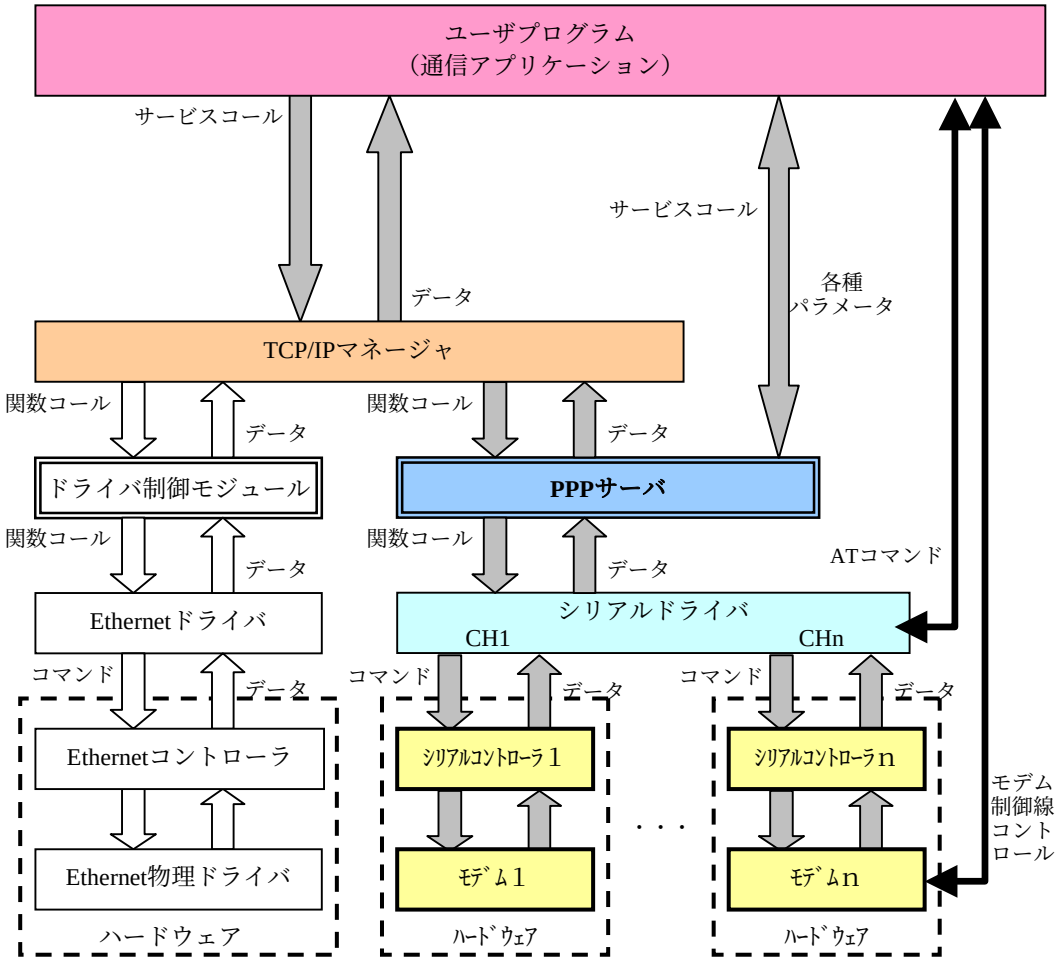


図 1-1　PPPサーバ使用時の構成例

2. PPP サーバ使用方法

2.1 構築時の設定と登録について

PPP サーバを使うためには、ユーザのプログラム構築時に PPP サーバの「タスク情報の設定」、「物理インタフェースドライバの登録」、「TCP/IP マネージャへの登録」を行う必要があります。

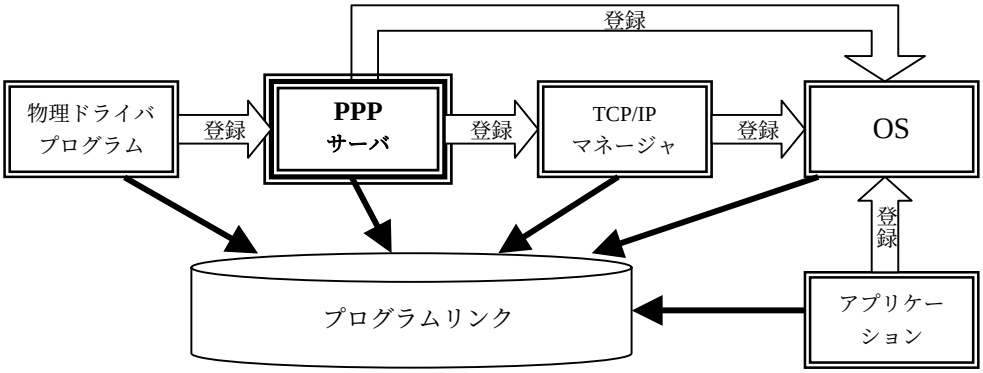


図 2-1 PPPサーバの構築

2.1.1 タスク情報の設定

PPP サーバはタスクとして実行されます。また、PPP サーバ内部で OS のイベントフラグを使用しています。そのためタスク情報テーブル（ppdConfTbl）に「タスクの優先度」、「タスク ID」、「イベントフラグ ID」、「タスクのスタックサイズ」といったタスク情報を登録しなければなりません。タスク情報の登録についての詳細は、PPP サーバの構築マニュアルを参照してください。

2.1.2 周期起動ハンドラ

PPP サーバは周期起動ハンドラにより内部の時間管理を行っています。そのため PPP サーバの周期起動ハンドラを一定時間の周期（100ms）にてコールする必要があります。PPP サーバ周期起動ハンドラは OS の周期起動ハンドラとして登録することも可能です。周期起動ハンドラについての詳細は、PPP サーバの構築マニュアルを参照してください。

2.1.3 物理インタフェースドライバの登録

本PPPサーバには、（シリアルコントローラ等の）物理インタフェース用のドライバは含まれていません。そのため、ユーザが準備した「物理インタフェース用ドライバプログラム」を使って、PPPクライアントとの通信を行います。

ユーザが準備する「物理インタフェース用ドライバプログラム」のPPPサーバから見たインタフェース関数の仕様については、「4　物理ドライバインタフェース」を参照してください。

PPPサーバに「物理インタフェース用ドライバプログラム」を登録するには、PPPサーバ用物理ドライバエントリテーブル（ppdDrvTbl）に「物理インタフェース用ドライバプログラム」の送信用関数（関数名は任意）と受信用関数（関数名は任意）を登録します。登録の詳細については、PPPサーバの構築マニュアルを参照してください。

2.1.4 TCP/IP マネージャへの登録

PPPサーバはTCP/IPマネージャにとって1つのドライバモジュールです。

PPPサーバのエントリアドレステーブル（drv_PPD）をTCP/IPマネージャのドライバ制御モジュール登録テーブル（drvConfTbl）に登録することによって、TCP/IPマネージャから認識されるようになります。ドライバモジュールとしての登録の詳細については、PPPサーバの構築マニュアルを参照してください。

2.1.5 PPP サーバのディスクリプタ番号について

本PPPサーバはマルチセッションに対応しています。そのため、各セッションに対応する物理デバイスを区別するために各サービスコールにてディスクリプタ番号を指定する必要があります。各サービスコールで指定されたディスクリプタ番号は、シリアルドライバ等の物理ドライバプログラムに渡されますので、物理ドライバにて対応する物理デバイスに対して送受信を行ってください。

ディスクリプタ番号には1～PPP_DESCRIPTORの範囲の値を指定することができます。PPP_DESCRIPTORは構築時にユーザにて設定することができます。設定方法に関してはPPPサーバの構築マニュアルを参照してください。

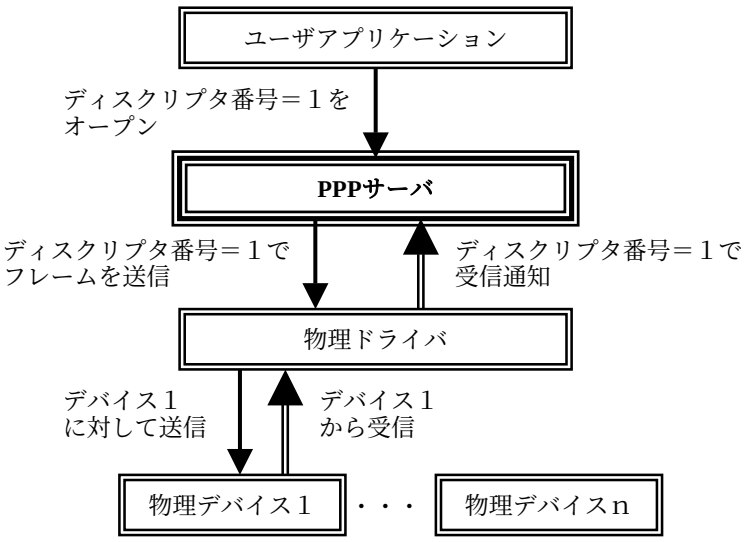
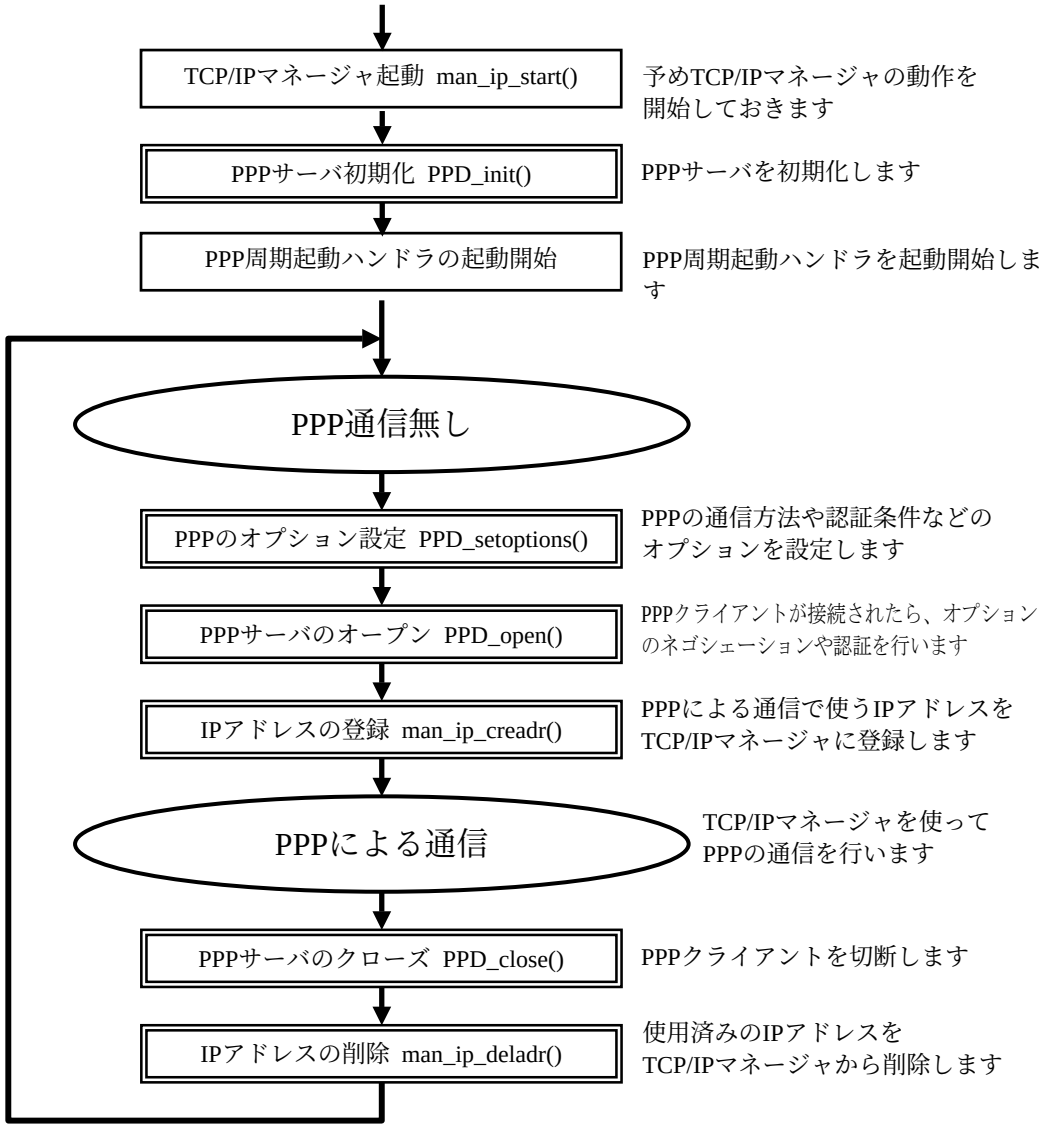


図 2-2　ディスクリプタ番号の概要

2.2 運用時の設定と登録について

PPPサーバを使って通信するためには、予め「PPPサーバの初期化」を行っておく必要があります。運用開始時には「PPPサーバオプション設定」、「PPPサーバのオープン」、「IPアドレスの登録」を行います。また、運用終了時には「PPPサーバのクローズ」、「IPアドレス削除」を行います。



注）man_ip_start()、man_ip_creadr()、man_ip_deladr()はTCP/IPマネージャのサービスコールです

図 2-3 基本的なPPPの運用手順

2.2.1 PPP サーバの初期化

PPPサーバの運用を開始する前に、1回だけPPPの初期化関数（PPD_init）をコールしてください。
初期化関数によってPPPサーバが初期化され、使用可能になります。
尚、PPPの初期化はOSのシステム初期化時に実施しても、ユーザアプリケーションの中で実施してもどちらでもかまいません。

2.2.2 PPP 周期ハンドラの起動

PPPサーバ周期起動ハンドラ（PPD_timerHandler）の周期起動を開始します。周期起動ハンドラは、PPPの初期化後、PPPのオープンを実施する前に起動してください。全てのセッションのクローズ後は、周期起動を停止してもかまいませんが、再びPPPのオープンを行う場合は、周期起動を開始してください。

2.2.3 PPP のオプション設定

PPPサーバの初期化終了後、PPPの通信をオープンする前にPPPサーバのオプションを設定します。PPPサーバはPPPのオープン時に、ここで設定したオプションに従ってPPPクライアントとのネゴシェーションと認証を実施します。

PPPのオプション設定に関する詳細は、「3.2　オプション操作サービスコール」を参照してください。

2.2.4 PPP サーバのオープン

シリアルドライバなどの初期化を済ませて、PPPサーバとPPPクライアント間の物理インタフェースが確立したら（モデムを経由したシリアル通信なども確立しておいてください）、PPPサーバのオープン（PPD_open）をコールします。

PPPサーバのオープンでは、タスク情報テーブル（ppdConfTbl）に設定されている「タスクID」、「タスクの優先度」、「タスクのスタックサイズ」に従って、PPPサーバタスクを生成し起動します。また、タスク情報テーブルに設定されている「イベントフラグID」に従って、PPPサーバ用のイベントフラグを生成します。

タスクの生成と起動、イベントフラグの生成が成功すると、PPPクライアントに対してオプションのネゴシェーションと認証を行って、PPPによる通信を可能にします。

PPPサーバのオープン完了時には、ネゴシエーションにて決定した自分（サーバ）と相手（クライアント）のIPアドレス等を取得することができます。

PPPサーバのオープンに関する詳細は、「3.3　接続・切断サービスコール」を参照してください。

2.2.5 IP アドレスの登録

PPPのオープンによって、オプションのネゴシェーションと認証が完了すると、PPPサーバとPPPクライアント間の通信ができる状態になりますが、この時点ではTCP/IPマネージャはPPPサーバを有効なドライバモジュールとして認識していません。

TCP/IPマネージャにPPPサーバを有効なドライバモジュールとして認識させるためには、TCP/IPマネージャのIPアドレス登録サービスコール（man_ip_creadr）を使って、ドライバモジュール（ここではPPPサーバ）とPPPサーバのIPアドレスの関連付けを行う必要があります。**注：PPPサーバのIPアドレスが予め決まっており、TCP/IPマネージャ起動サービスコール（man_ip_start）にてすでにPPPサーバのIPアドレスを登録してある場合はこの必要はありません。**

TCP/IPマネージャの起動サービスコール（man_ip_start）およびIPアドレス登録サービスコール（man_ip_creadr）の詳細については、「TCP/IPマネージャ リファレンスマニュアル」を参照してください。

2.2.6 PPP サーバのクローズ

PPPによる通信が終了したら、PPPサーバのクローズ（PPD_close）をコールします。PPPサーバのクローズでは、PPPクライアントとの通信を切断した後、PPPサーバタスクを終了して、PPPサーバタスクとPPPサーバ用のイベントフラグを削除します。

PPPサーバのクローズに関する詳細は、「3.3　接続・切断サービスコール」を参照してください。

2.2.7 IP アドレスの削除

PPPサーバのクローズが完了したら、TCP/IPマネージャのIPアドレス削除サービスコール（man_ip_deladr）を使って、PPPで使用済のIPアドレスをTCP/IPマネージャから削除します。IPアドレスを削除する前には、そのIPアドレスを使用している通信端点や受付口（BSDソケットAPIの場合はソケット）を削除しておいてください。

TCP/IPマネージャのIPアドレス削除サービスコール（man_ip_deladr）の詳細については、「TCP/IPマネージャ リファレンスマニュアル」を参照してください。

2.3 インタフェース

PPPサーバは、ユーザがPPPの接続と切断を制御するための「サービスコール（インタフェース）」と、ユーザが作成した物理ドライバとの間で送受信を行う「物理ドライバインタフェース」を持っています。

2.3.1 サービスコール

サービスコールはユーザがPPPサーバをPPPクライアントと接続する機能を提供します。

表 2-1 サービスコール

区分	サービスコール名称	サービスコールの機能
初期化	PPD_init	PPP サーバを初期化する
オプション操作	PPD_getoptions	PPP オプションを取得する
	PPD_setoptions	PPP オプションを設定する
接続・切断	PPD_open	PPP をオープン（PPP クライアントと接続）する
	PPD_close	PPP をクローズ（PPP クライアントから切断）する
状態参照	PPD_status	PPP サーバの現在の状態を取得する
経路操作	PPD_addrpute	経路情報を追加する
	PPD_delroute	経路情報を削除する
	PPD_refroute	経路情報を参照する

表 2-2 コールバック関数

区分	コールバックルーチン名称	コールバックルーチンの機能
接続完了通知	ppd_Callback（仮称）	PPP の接続完了またはエラーを通知する
切断完了通知	ppd_Callback（仮称）	PPP の切断完了を通知する
切断通知	ppd_Callback（仮称）	PPP の切断を通知する

2.3.2 物理ドライバインタフェース

ユーザは、PPPサーバがPPPクライアントと通信するための物理ドライバを準備しなければなりません。

PPPサーバが使用する物理ドライバ上のインタフェース関数は、PPPフレームの送信用に使用するppd_PutPhysical（仮称）と、PPPフレームの受信用に使用するppd_GetBufPhysical（仮称）、ppd_RelBufPhysical（仮称）があります。

また、ユーザが準備する物理ドライバでは、フレームの送信完了とフレームの受信をPPPサーバに通知するためにコールバックルーチンを呼び出す必要があります。PPPフレームの送信が完了した事をPPPサーバに通知するにはPPD_SndEndCallBackコールバックルーチンを、PPPフレームを受信した事をPPPサーバに通知するにはPPD_RcvCallBackコールバックルーチンを呼び出してください。

表 2-3 物理ドライバ送受信関数

区分	関数の名称	関数の機能
送信・受信	ppd_PutPhysical（仮称）	PPP フレームを送信する
	ppd_GetBufPhysical（仮称）	PPP フレーム受信バッファ情報を取得する
	ppd_RelBufPhysical（仮称）	PPP フレーム受信バッファを解放する

表 2-4 物理ドライバコールバック関数

区分	コールバックルーチン名称	コールバックルーチンの機能
受信	PPD_RcvCallBack	PPP フレームを受信した事を通知する
送信	PPD_SndEndCallBack	PPP フレームの送信が完了した事を通知する

詳細は、「4 物理ドライバインタフェース」を参照してください。

3. サービスコール

サービスコールは、ユーザアプリケーションからPPPの接続と切断、およびオプションを利用する場合のインタフェースを提供します。

本節では、サービスコールについての詳細な説明を以下の形式で行っています。

No. サービスコール名　　機能

【発行可能なシステム状態[*1]】

C言語インタフェース

マネージャコール呼出し形式[*2]

パラメータ

型	パラメータ	パラメータの意味
・	・	・
・	・	・
・	・	・

リターンパラメータ

型	パラメータ	パラメータの意味
・	・	・
・	・	・

パケットの構造

リターン値／エラーコード

リターン値またはニモニック	リターン値またはエラーコードの意味
・	・
・	・
・	・

解　説

・・・・・・・・・・・・

*1発行可能なシステム状態を以下のアルファベットで示します

Ｔ：タスク実行状態

Ｄ：ディスパッチ禁止状態

Ｌ：ＣＰＵロック状態

Ｉ：非タスク部実行状態

なお、各状態の詳細は各ITRONのユーザーズマニュアルを参照してください。

発行可能なシステム状態以外の状態でサービスコールを発行した場合、システムの正常な動作は保証されません。

3.1 初期化サービスコール

3.1.1 PPD_init PPP サーバの初期化

【T／L／I】

C言語インタフェース

```
void PPD_init ( void );
```

パラメータ

無し

リターンパラメータ

無し

解　説

PPPサーバを初期化します。

PPPサーバの内部変数を初期化し、PPPをオープン可能な状態にします。
PPD_init は、PPPサーバ使用前に１回だけ実行してください。

3.2 オプション操作サービスコール

3.2.1 PPD_setoptions PPP サーバオプションの設定

【T】

C言語インタフェース

```
ER ercd = PPD_setoptions(W pd, PPD_OPTIONS *p_opts, OPTIONS_FLAG flag)
```

パラメータ

W	pd	ディスクリプタ
PPD_OPTIONS	*p_opts	PPPサーバオプション情報の先頭アドレス
OPTIONS_FLAG	flag	PPPサーバオプションの設定フラグ

リターンパラメータ

ER	ercd	リターン値またはエラーコード

パケットの構造

```
typedef struct ppd_options {
        UW      neg_mru : 1,                MRUのネゴシェーション
                neg_asyncmap : 1,           ASYNCマップのネゴシェーション
                neg_upap : 1,               PAP認証のネゴシエーション
                neg_chap : 1,               CHAP認証のネゴシエーション
                neg_magicnumber : 1,        マジックナンバのネゴシエーション
                neg_pcompression : 1,       HDLCプロトコルフィールド圧縮のネゴシエーション
                neg_accompression : 1,      HDLCアドレス･制御フィールド圧縮のネゴシエーション
                neg_lqr : 1,                品質プロトコルのネゴシェーション（未サポート）
                neg_cbcp : 1;               コールバックのネゴシェーション（未サポート）
                neg_vjcompression : 1;      IPヘッダ圧縮のネゴシエーション
        UH      silent;                     サイレントモード
        UH      mru;                        MRUの値
        UW      asyncmap;                   ASYNCマップの値
        UW      ouraddr;                    自分（PPPサーバ側）のIPアドレス
        UW      hisaddr;                    相手（PPPクライアント側）のIPアドレス
        UW      old_ipcp : 1,               旧型IPアドレスオプション
} PPD_OPTIONS;

typedef enum {
        PPP_WANT = 0,       ネゴシェーション時の要求項目を指定
        PPP_ALLOWED,        ネゴシェーション時の受付項目を指定
        PPP_GOT,            ネゴシェーション結果の要求項目を指定
        PPP_HIS             ネゴシェーション結果の受付項目を指定
} OPTIONS_FLAG;
```

リターン値／エラーコード

E_OK	正常終了
E_PAR	パラメータエラー（pd＞PPP_DESCRIPTOR、p_optsが0または4の倍数以外、flagがPPP_WANTでもPPP_ALLOWEDでもない、mru＞1500、mru＜128、IPアドレスがブロードキャスト、またはマルチキャストアドレス、または重複している）
E_OBJ	オブジェクト状態不正（PPPコネクションがクローズされていない）

解　説

PPPサーバのオプションを設定します。
本サービスコールは、PPD_initを実行した後はいつでも利用することができます。
PPPサーバは、PPD_openサービスコール実行時に、ここで設定されたオプションに従ってPPPクライアントとのネゴシェーションを行います。また、一度設定したPPPサーバオプションはPPD_closeを呼び出すことでデフォルト値に設定されます。

PPPクライアントに対して要求するオプション（flag＝PPP_WANT）の設定と、PPPクライアントからの要求を許可するオプション（flag＝PPP_ALLOWED）の設定を2回に分けて別々に行う必要があります。設定を行わない側のオプションについてはデフォルト設定となります。

pdには設定するディスクリプタ番号を指定します。ディスクリプタ番号はマルチセッションにてPPPを利用する際に各セッションを区別する為の番号です。0を指定した場合は、全てのディスクリプタに対して設定を行います。
WANT（Flag=PPP_WANT）で1を指定した項目は、PPPクライアントの接続時にPPPサーバから要求を行う項目です。0を指定した項目は、PPPサーバからの要求を行いません。
ALLOWED（Flag=PPP_ALLOWED）で1を指定した項目は、PPPクライアントとの接続時にPPPクライアントからの要求を受け付ける項目です。0を指定した項目は、PPPクライアントからの要求を拒否します。

認証の設定（neg_upap、neg_chap）と実際に行う認証は次の表を参照してください。PAP、CHAP両方を有効にした場合は、CHAP認証が優先されます。

		neg_chap（WANT設定）	
		0	1
neg_upap（WANT設定）	0	認証無し	CHAP認証
	1	PAP認証	CHAP認証

本サービスコールでPPPサーバオプションを設定しない場合は、次表のデフォルト設定でネゴシェーションを行います。

デフォルトのLCPネゴシェーション設定

No.	シンボル名	ネゴシェーション項目	有効(1)／無効(0)	
			WANT	ALLOWED
1	neg_mru	MRU	0	1
2	neg_asyncmap	ASYNC マップ	1	1
3	neg_upap	PAP 認証	0	0 固定※1
4	neg_chap	CHAP 認証	1	0 固定※1
5	neg_magicnumber	マジックナンバ	1	1
6	neg_pcompression	HDLC プロトコルフィールド圧縮	1	1
7	neg_accompression	HDLC アドレス・制御フィールド圧縮	1	1
8	neg_lqr	品質プロトコル（未サポート）	0	0
9	neg_cbcp	コールバック（未サポート）	0	0
10	neg_vjcompression	IP ヘッダ圧縮	1	1

※1 相手側が認証方法を指定することはできません。

デフォルトのネゴシェーションバラメータ設定値

No.	シンボル名	バラメータの意味	デフォルトの設定値	
			WANT	ALLOWED
1	silent	サイレントモード	1	未使用
2	mru	MRU の値	1500	未使用
3	asyncmap	ASYNC マップの値	0x000A0000	未使用

本サービスコールでは、flag＝PPP_WANTの場合のみ有効です。

デフォルトのIPアドレス設定値

No.	シンボル名	IP アドレスの用途	デフォルトの設定値	
			WANT	ALLOWED
1	ouraddr	自分（PPP サーバ側）の IP アドレス	0	未使用
2	hisaddr	相手（PPPクライアント側）のIPアドレス※1	0	未使用

本サービスコールでは、flag＝PPP_WANTの場合のみ有効です。
※1 相手IPアドレスの指定は、PPD_openサービスコールのユーザ認証情報の指定が優先されます。

デフォルトのIPCPネゴシェーション設定

No.	シンボル名	ネゴシェーション項目	有効(1)／無効(0)	
			WANT	ALLOWED
1	old_ipcp	旧型 IP アドレスオプション	0	未使用

本サービスコールでは、old_ipcpの指定値はflag＝PPP_WANTの場合のみ有効です。

silentに1を設定した場合は、PPPクライアントがネゴシエーションを開始するまでPPPサーバからは何も送信しません。silentに0を指定した場合は、PPPサーバからネゴシエーションを開始します。

asyncmapには、PPPサーバが受信の際にエスケープ変換が必要なキャラクタを指定します。32ビットの各ビットが、コード0x00～0x1fのキャラクタに対応します（例えば、0x00000001は0x00を示し, 0x800000000は0x1fを示します）。物理ドライバにてソフトウェアフロー制御（XON/XOFF）を使用する場合は、0x000A0000（0x11(XON)、0x13(XOFF)）を設定してください。PPPサーバから送信されるデータも、この設定に従ってエスケープ変換を行います。

hisaddrには、PPPクライアントに割り当てるIPアドレスを指定します。hisaddrに0を指定した場合、ネゴシェーション時にPPPクライアントからIPアドレスを取得します。ただし、相手IPアドレスの指定は、PPD_openサービスコールのユーザ認証情報に指定するIPアドレスの値が優先されます。
ouraddrはTCP/IPマネージャに登録されている（またはこれから登録する）PPPサーバ側のIPアドレスを指定してください。ouraddrに0を指定した場合、ネゴシェーション時にPPPクライアントからIPアドレスを取得します。

ouraddrやhisaddrにブロードキャスト、またはマルチキャストアドレスを指定した場合、および、hisaddrに他のディスクリプタのhisaddrと重複するIPアドレスを指定した場合は、エラーコードとしてE_PARを返します。

決定したIPアドレスは、PPD_openサービスコール終了後にlocaladdr（PPPサーバのIPアドレス）とremoteaddr（PPPサーバのIPアドレス）で参照できます。また、PPD_getoptionsサービスコールをflag＝PPP_GOTで発行することでouraddrからPPPサーバのIPアドレスを、PPD_getoptionsサービスコールをflag＝PPP_HISで発行することでhisaddrからPPPクライアントのIPアドレスを参照することができます。

3.2.2 PPD_getoptions PPP サーバオプションの取得

【T】

C言語インタフェース

```
ER ercd = PPD_getoptions(W pd, PPD_OPTIONS *p_opts, OPTIONS_FLAG flag)
```

パラメータ

W	pd	ディスクリプタ
PPD_OPTIONS	*p_opts	PPPサーバオプション情報の先頭アドレス
OPTIONS_FLAG	flag	PPPサーバオプションの設定フラグ

リターンパラメータ

ER	ercd	リターン値またはエラーコード

パケットの構造

```
typedef struct ppd_options {
        UW      neg_mru : 1,                MRUのネゴシェーション
                neg_asyncmap : 1,           ASYNCマップのネゴシェーション
                neg_upap : 1,               PAP認証
                neg_chap : 1,               CHAP認証
                neg_magicnumber : 1,        マジックナンバ
                neg_pcompression : 1,       HDLCプロトコルフィールド圧縮
                neg_accompression : 1,      HDLCアドレス・制御フィールド圧縮
                neg_lqr : 1,                品質プロトコルネゴシェーション（未サポート）
                neg_cbcp : 1;               コールバックネゴシェーション（未サポート）
                neg_vjcompression : 1;      IPヘッダ圧縮のネゴシエーション
        UH      silent;                     サイレントモード
        UH      mru;                        MRUの値
        UW      asyncmap;                   ASYNCマップの値
        UW      ouraddr;                    自分（PPPサーバ側）のIPアドレス
        UW      hisaddr;                    相手（PPPクライアント側）のIPアドレス
        UW      old_ipcp : 1;               旧型IPアドレスオプション
} PPD_OPTIONS;

typedef enum {
        PPP_WANT = 0,       ネゴシェーション時の要求項目を指定
        PPP_ALLOWED,        ネゴシェーション時の受付項目を指定
        PPP_GOT,            ネゴシェーション結果の要求項目を指定
        PPP_HIS             ネゴシェーション結果の受付項目を指定
} OPTIONS_FLAG;
```

リターン値／エラーコード

E_OK	正常終了
E_PAR	パラメータエラー（pdが0またはpd＞PPP_DESCRIPTOR、p_optsが0または4の倍数以外、flag＞3）
E_OBJ	オブジェクト状態不正（オープン処理中 または、クローズ処理中）

解　説

PPPサーバのオプションの状態を参照します。
本サービスコールは、PPD_initを実行した後はいつでも利用することができます。

pdには状態を参照したいディスクリプタを指定します。ディスクリプタ番号はマルチセッションにてPPPを利用する際に各セッションを区別する為の番号です。pdに0を指定した場合はエラーコードとしてE_PARを返します。
PPPクライアントに対して要求するオプションの状態を参照したい場合は、flagにPPP_WANTを指定します。
PPPクライアントからの要求を受付けるオプションの状態を参照したい場合は、flagにPPP_ALLOWEDを指定します。
前回接続時にPPPクライアントに対して要求したオプションの結果を参照したい場合は、flagにPPP_GOTを指定します。
前回接続時にPPPクライアントからの要求を受付けたオプションの結果を参照したい場合は、flagにPPP_HISを指定します。

オープン処理またはクローズ処理を行っている最中にflagにPPP_GOTまたはPPP_HISを指定して本サービスコールを発行した場合は、エラーコードとしてE_OBJを返します。

PPD_openサービスコールによって、決定したIPアドレスは、PPD_openサービスコール終了後にlocaladdr（PPPサーバのIPアドレス）とremoteaddr（PPPクライアントのIPアドレス）で参照できます。また、PPD_getoptionsサービスコールをflag＝PPP_GOTで発行することでouraddrからPPPサーバのIPアドレスを、PPD_getoptionsサービスコールをflag＝PPP_HISで発行することでhisaddrからPPPクライアントのIPアドレスを参照することができます。

3.3 接続・切断サービスコール

3.3.1 PPD_open PPP サーバのオープン

【T】

C言語インタフェース

```
ER ercd = PPD_open (W pd, PPD_PARAM *param, TMO tmout);
```

パラメータ

W	pd	ディスクリプタ
PPD_PARAM	*param	PPP接続情報のアドレス
TMO	tmout	タイムアウト値

リターンパラメータ

ER	ercd	リターン値またはエラーコード
UW	param->localaddr	自分のIPアドレス
UW	param->remoteaddr	相手のIPアドレス
UH	param->mru	相手の最大受信データ長
H	param->perrno	E_PAR, E_SYS, EV_PROT発生時の詳細エラー情報

パケットの構造

```
typedef struct ppd_param{
        PPD_SECRETS   *secrets;     ユーザ認証情報配列の先頭アドレス
        UH      secrets_cnt;        ユーザ認証情報配列の数
        UB      *localname;         認証にて使用するローカルシステム名
        UW      dnsaddr[2];         プライマリ、およびセカンダリDNSサーバのIPアドレス
        UW      winsaddr[2];        プライマリ、およびセカンダリWINSサーバのIPアドレス
        UH      echointerval;       LCPエコー要求を送信する間隔（秒）
        UH      echofailure;        LCPエコー要求に応答がない場合に切断するまでの
                                    LCPエコー要求送信カウント数
        FP      callback;           コールバックルーチンのアドレス
        UW      localaddr;          決定した自分（PPPサーバ側）のIPアドレス
        UW      remoteaddr;         決定した相手（PPPクライアント側）のIPアドレス
        UH      mru;                受信した相手（PPPクライアント側）の最大受信データ長（MRU）
        H       perrno;             詳細エラーコード
} PPD_PARAM;

typedef struct ppd_secrets{
        UB      *user;              認証におけるユーザネーム文字列のアドレス
        UB      *passwd;            PAP認証におけるパスワード文字列のアドレス
        UB      *chap_secret;       CHAP認証におけるパスワード文字列のアドレス
        UW      ipaddr;             PPPクライアントに割り当てるIPアドレス
} PPD_SECRETS;
```

リターン値／エラーコード

E_OK	正常終了
E_WBLK	ノンブロッキングコール受け付け
E_PAR	パラメータエラー（pd＜0またはpd＞PPP_DESCRIPTOR、paramが0または4の倍数以外、callbackが奇数、secrets_cnt＞256、user, passwd, chap_secret, localnameが0または文字列が長すぎる、ipaddrがブロードキャストまたはマルチキャストアドレスまたは重複、tmout＜-2またはtmout＝0）
E_OBJ	オブジェクト状態不正（すでにオープンしている）
E_CLS	接続切断（相手応答なし、切断要求受信）
E_SYS	システムエラー（ドライバテーブル不正、イベントフラグ生成失敗、タスク生成失敗）
E_TMOUT	タイムアウトエラー
EV_PROT	プロトコルエラー（認証失敗、ネゴシェーション失敗）

詳細エラーコード

E_PAR発生時の詳細エラーコード

PPD_ENO_DESCRIPTOR	1	PPPのディスクリプタが不正
PPD_ENO_USERNAME	2	PAP認証ユーザネームが不正
PPD_ENO_PASSWD	3	PAP認証パスワードが不正
PPD_ENO_SECRET	4	CHAP認証パスワードが不正
PPD_ENO_TMOUT	5	タイムアウト値が不正
PPD_ENO_CBKADR	6	コールバックルーチンのアドレスが不正
PPD_ENO_SECRETINF	7	ユーザ認証情報が不正
PPD_ENO_LOCALNAME	8	ローカルシステム名が不正
PPD_ENO_ REMOTEADDR	9	相手IPアドレスがブロードキャスト、またはマルチキャストアドレスまたは相手IPアドレスが重複
PPD_ENO_DNSADDR	10	DNSサーバアドレスがブロードキャスト、またはマルチキャストアドレス
PPD_ENO_WINSADDR	11	WINSサーバアドレスがブロードキャスト、またはマルチキャストアドレス

E_SYS発生時の詳細エラーコード

PPD_ENO_DRVTBL	21	ドライバテーブル不正
PPD_ENO_CREFLG	22	イベントフラグ生成失敗
PPD_ENO_CRETSK	23	PPPクライアントタスク生成失敗

EV_PROT発生時の詳細エラーコード

PPD_ENO_PAPFAIL	31	PAP認証拒否
PPD_ENO_CHAPFAIL	32	CHAP認証拒否
PPD_ENO_NEGLCP	33	LCPネゴシェーション失敗
PPD_ENO_NEGIPCP	34	IPCPネゴシェーション失敗

解　説

PPPのコネクションをオープンします。
与えられたI/OデバイスをPPPサーバーPPPクライアント間におけるPPPのリンクを確立させます。
本サービスコールを呼出す前にモデム等を通じてPPPクライアントと接続されている必要があります。

pdにはディスクリプタ番号を指定します。ディスクリプタ番号はマルチセッションにてPPPを利用する際に各セッションを区別する為の番号です。ディスクリプタ番号は物理ドライバの呼出しの際にパラメータとして物理ドライバに渡されます。
pdに0を指定した場合は、オープンしていない全てのディスクリプタをオープンします。【注１】

PPPコネクション確立の際に、PPPクライアントに対して認証（デフォルトではCHAP認証をクライアントに要求します。認証についての設定はPPD_setoptionsにて変更することができます）を要求することができます。正しい認証ができないとPPPコネクションは確立されません。
PPPクライアントの認証情報は、接続を許可するユーザ毎にsecretsに指定し、認証情報の数をsecrets_cntに指定します。secretsが示すユーザ認証情報の内容は、オープンの処理が終了するまで書き換えないでください。書き換えた場合の動作は保証しません。
secrets->ipaddrに0を指定した場合は、PPPクライアントからIPアドレスを取得します（IPCPネゴシエーション）。

localnameには認証の際にPPPクライアントに渡されるローカルシステム名を指定します。

dnsaddr、winsaddrにはPPPクライアントに渡すDNSサーバ（プライマリ、セカンダリ）、およびWINSサーバ（プライマリ、セカンダリ）のIPアドレスを指定します。これらのIPアドレスを渡す必要がない場合は、0を指定してください。

echointerval、echofailureには、LCPエコー要求を送信する間隔（秒）、LCPエコー要求に対する応答がない場合に切断するまでのLCPエコー要求送信回数を指定します。これにより、モデム接続の切断など、通信ができなくなった場合に自動切断することができます。これらの値に0を指定した場合はLCPエコー要求は送信しません（この場合は自動切断しません）。

callbackには、PPPサーバが呼び出すユーザのコールバックルーチンのアドレスを指定します。callbackにNULL(0)を指定した場合は、コールバックルーチンの呼び出しは行いません。ただし、secretsおよびcallbackの両方にNULL(0)を指定した場合は、perrnoにPPD_ENO_ CBKADRを設定し、エラーコードとしてE_PARを返します。コールバックの種類に関しては「3.5 コールバックルーチン」を参照してください。

PPPコネクションのオープンに成功した場合は、perrnoに0を設定し、リターンコードとしてE_OKを返します。また、ネゴシエーションにて決定した自分（PPPサーバ側）のIPアドレス、相手（PPPクライアント側）のIPアドレス、相手（PPPクライアント側）の最大受信データ長を、それぞれ localaddr、remoteaddr、mruに格納します。

PPPクライアントに割り当てられるIPアドレスを次の表に示します。

		認証情報(secrets)
認証設定	有り	受信したユーザ名と一致するuserを持つsecretsのipaddr
	無し	PPD_setoptions()にて設定したhisaddr

決定したPPPサーバのIPアドレスにてTCP/IP通信を行う場合は、IPアドレスをTCP/IPマネージャに登録する必要があります（「2.2.5　IPアドレスの登録」の項を参照）。相手の最大受信データ長(mru)は、TCP/IPマネージャのMTU（最大送信データ長）として自動的に登録され、TCP/IPマネージャからPPPサーバ経由で送信されるパケットはMTUを超えないパケット長にて送信されます。ただし、PPPサーバをマルチセッションで使用する場合は、IPアドレスを登録した時点のmruがMTUとして登録される為、セッション毎にMTUを変更することはできません。mruのデフォルト値は1500となっています。

paramが0、または4の倍数以外の場合は、エラーコードとしてE_PARを返します。詳細エラーコードperrnoの設定は行いません。

tmoutには、オープンを待つ時間を指定します。待ち時間（正の値）が経過するまでオープンが完了しなかった場合、処理を中断し、エラーコードとしてE_TMOUTを返します。
tmoutにTMO_FEVR(-1)を指定すると無限待ち、TMO_WBLK(-2)を指定するとノンブロッキングコールとなります。
ノンブロッキングコールを指定した場合は、待ち状態にならずにエラーコードとしてE_WBLKを返して直ちにリターンしますが処理は継続されます。その後コネクションが確立、または失敗した時点でPPPの接続完了／エラー通知のコールバックルーチンにより通知します。
PPPの接続完了／エラー通知コールバックの詳細に関しては「3.5.1　ppd_callback（仮称）　　PPPの接続完了／エラー通知」を参照してください。
尚、pdに0を指定した場合は、tmoutに指定した値は無効となります（ノンブロッキングコールとなります）。

secretsが4の倍数以外、secrets_cntが256を超えている場合は、perrnoにPPD_ENO_SECRETINFを設定し、エラーコードとしてE_PARを返します。
PAP認証またはCHAP認証が選択されている状態で、secrets->userに0を指定した場合や、secrets->userの指し示す文字列が**ヌル文字を含めて256文字**を超えている場合は、perrnoにPPD_ENO_USERNAMEを設定し、エラーコードとしてE_PARを返します。
PAP認証が選択されている状態で、secrets->passwdに0を指定した場合や、secrets->passwdの指し示す文字列が**ヌル文字を含めて256文字**を超えている場合は、perrnoにPPD_ENO_PASSWDを設定し、エラーコードとしてE_PARを返します。
CHAP認証が選択されている状態で、secrets->chap_secretに0を指定した場合や、secrets->chap_secretの指し示す文字列が**ヌル文字を含めて256文字**を超えている場合は、perrnoにPPD_ENO_SECRETを設定し、エラーコードとしてE_PARを返します。
CHAP認証が選択されている状態で、localnameに0を指定した場合や、localnameの指し示す文字列が**ヌル文字を含めて256文字**を超えている場合は、perrnoにPPD_ENO_LOCALNAMEを設定し、エラーコードとしてE_PARを返します。
secrets->ipaddrがマルチキャストアドレス、またはブロードキャストアドレス、または重複したIPアドレスが設定された場合は、perrnoにPPD_ENO_ REMOTEADDRを設定し、エラーコードとしてE_PARを返します。

callbackが奇数の場合、secretsおよびcallbackの両方にNULL(0)を指定した場合、ノンブロッキング指定でcallbackにNULL(0)を指定した場合は、perrnoにPPD_ENO_CBKADRを設定し、エラーコードとしてE_PARを返します。

物理ドライバエントリテーブルの設定が不正（アドレスが0または奇数）の場合は、perrnoにPPD_ENO_DRVTBLを設定し、エラーコードとしてE_SYSを返します。

dnsaddrがマルチキャストアドレスまたはブロードキャストアドレスの場合は、perrnoにPPD_ENO_DNSADDRを設定し、エラーコードとしてE_PARを返します。
winsaddrがマルチキャストアドレスまたはブロードキャストアドレスの場合は、perrnoにPPD_ENO_WINSADDRを設定し、エラーコードとしてE_PARを返します。

PPPサーバ用のイベントフラグ生成に失敗した場合は、perrnoにPPD_ENO_CREFLGを設定し、エラーコードとしてE_SYSを返します。
PPPサーバタスクの生成に失敗した場合は、perrnoにPPD_ENO_CRETSKを設定し、エラーコードとしてE_SYSを返します。
PAPによる認証に失敗した場合は、perrnoにPPD_ENO_PAPFAILを設定し、エラーコードとしてEV_PROTを返します。
CHAPによる認証に失敗した場合は、perrnoにPPD_ENO_CHAPFAILを設定し、エラーコードとしてEV_PROTを返します。
LCPのネゴシェーションに失敗した場合は、perrnoにPPD_ENO_NEGLCPを設定し、エラーコードとしてEV_PROTを返します。
IPCPのネゴシェーションに失敗した場合は、perrnoにPPD_ENO_NEGIPCPを設定し、エラーコードとしてEV_PROTを返します。

【注１】pdに0を指定する場合は、予めppd_setoptionsにて全てのディスクリプタのサイレントモードを有効（silent=1）にしておいてください。

3.3.2 PPD_close PPP サーバのクローズ

【T】

C言語インタフェース

ER ercd = PPD_close (W pd, TMO tmout);

パラメータ

W	pd	ディスクリプタ
TMO	tmout	ブロッキング／ノンブロッキング指定

リターンパラメータ

ER	ercd	リターン値またはエラーコード

リターン値／エラーコード

E_OK	正常終了
E_WBLK	ノンブロッキングコール受け付け
E_PAR	パラメータエラー（pd＜0またはpd＞PPP_DESCRIPTOR、tmoutが-1,-2以外）
E_OBJ	オブジェクト状態不正（クローズ処理がペンディング中）

解　説

PPPのコネクションをクローズします。

クローズによってPPPクライアント－PPPサーバ間におけるPPPのリンクを解放します。また、PPPオプションをデフォルト値に設定します。

pdにはディスクリプタ番号を指定します。ディスクリプタ番号はマルチセッションにてPPPを利用する際に各セッションを区別する為の番号です。0を指定した場合は、クローズしていない全てのディスクリプタをクローズします。

pdに接続待ちとなっているディスクリプタ番号を指定した場合は、接続の強制キャンセルを行います。その場合、PPD_openはエラーコードとしてE_CLSを返し終了します。PPD_openがノンブロッキング処理中の場合は、PPPの接続完了／エラー通知コールバックルーチンがコールされます。

tmoutにTMO_FEVR(-1)を指定するとブロッキングコール、TMO_WBLK(-2)を指定するとノンブロッキングコールとなります。ノンブロッキングコールを指定した場合は、待ち状態にならずにエラーコードとしてE_WBLKを返して直ちにリターンします。

その後、コネクションが切断した時点でPPPの切断完了通知のコールバックルーチンにより通知します。

コールバックルーチンのアドレスはPPD_openのパラメータcallbackにて指定します。

PPPの切断完了通知コールバックの詳細に関しては「3.5.3　ppd_callback（仮称）　PPPの切断通知」を参照してください。

pdに0を指定した場合は、tmoutに指定した値は無効となります（ノンブロッキングコールとなります）。

PPD_closeの処理が既にペンディング中の場合は、エラーコードとしてE_OBJを返します。

3.3.3 PPD_status PPP サーバの状態を取得する

【T】

C言語インタフェース

```
ER ercd = PPD_status (W pd, PPD_STATUS *p_st);
```

パラメータ

W	pd	ディスクリプタ
PPD_STATUS	*p_st	PPPのステータス情報を返す領域のアドレス

リターンパラメータ

ER	ercd	リターン値またはエラーコード

パケットの構造

```
typedef struct ppd_status{
        STAT    pppstat;        PPPのステータス
 PPD_STATUS;
```

リターン値／エラーコード

E_OK	正常終了
E_PAR	パラメータエラー（pdが0またはpd＞PPP_DESCRIPTOR、p_stが0または4の倍数以外）

解　説

PPPの状態を取得します。

pdに0を指定した場合は、エラーコードとしてE_PARを返します。
p_stに0、または4の倍数以外の値を指定した場合は、エラーコードとしてE_PARを返します。

pppstatの内容を以下に示します。

No.	シンボル名	値	意味
1	PPD_ST_OPENNING	1	オープン処理中（PPD_open()ペンディング中）
2	PPD_ST_CONNECT	2	PPP接続状態
3	PPD_ST_CLOSING	3	クローズ処理中（PPD_close()ペンディング中）
4	PPD_ST_CLOSE	4	PPP切断状態

3.4 経路操作サービスコール

3.4.1 PPD_addroute 経路情報を追加する

【T】

C言語インタフェース

ER ercd = PPD_addroute (PPD_ROUTE *p_route);

パラメータ

PPD_ROUTE	*p_route	PPP経路情報を格納した領域のアドレス

リターンパラメータ

ER	ercd	リターン値またはエラーコード

パケットの構造

```
typedef struct ppd_route{
        UW      dstaddr;        経路の宛先IPアドレス
        UW      netmask;        経路のサブネットマスク
        W       pd;             送信先ディスクリプタ番号
PPD_ROUTE;
```

リターン値／エラーコード

E_OK	正常終了
E_PAR	パラメータエラー（p_routeが0または4の倍数以外、pdが0または pd＞PPP_DESCRIPTOR、経路情報が重複）
E_OBJ	オブジェクト状態不正（未接続、またはPPP_MAXROUTEを超えて追加しようとした）

解　説

PPPの経路情報に新しい経路を追加します。最大で、PPP_MAXROUTE分の経路を登録できます。
PPP_MAXROUTEは構築時にユーザにて設定することができます。設定方法に関してはPPPサーバの構築マニュアルを参照してください。

経路情報は、送信パケットを宛先IPアドレスによって複数のディスクリプタに対して振り分ける際に使用します。また、送信パケットのフィルタリングを行い、適切な経路がない場合は送信しません。

dstaddrには、送信対象となるネットワークまたはホストのIPアドレスを指定します。
dstaddrにネットワークのIPアドレスを指定した場合は、netmaskにネットマスクを指定します。dstaddrにホストのIPアドレスを指定した場合は、netmaskに0xFFFFFFFF（255.255.255.255）を指定します。
dstaddr、またはnetmaskに0x00000000（0.0.0.0）を指定した場合は、デフォルト経路（全ての経路情報に当てはまらない場合に使用される経路）として使用されます。
pdには、経路の送信先となるディスクリプタ番号を指定します。

p_routeに0、または4の倍数以外の値を指定した場合は、エラーコードとしてE_PARを返します。
pdに0、またはPPP_DESCRIPTORより大きい値が指定された場合は、エラーコードとしてE_PARを返します。
pdに指定されたディスクリプタ番号が未接続の場合は、エラーコードとしてE_OBJを返します。
同じ経路情報が既に存在する場合は、エラーコードとしてE_PARを返します。

PPP接続完了時、接続されたPPPクライアントのIPアドレスの経路を自動的に追加します。自動的に追加される経路情報はPPP_DESCRIPTORの値によって異なります。自動的に追加される経路情報を以下に示します。

PPP_DESCRIPTOR＝1の場合	PPP_DESCRIPTOR＝2以上の場合
デフォルト経路を追加します （dstaddr=0、netmask=0、pd=1）	接続されたPPPクライアントに対する経路を追加します （dstaddr=PPPクライアントのIPアドレス、netmask=0xFFFFFFFF、pd=接続したディスクリプタ番号）

PPP接続切断時、そのディスクリプタに対する経路情報は自動的に削除されます。
適切な経路がなく、かつデフォルト経路が登録されていない場合、送信パケットは送信されません。
経路情報を変更する場合は、PPD_delrouteにて削除した後、PPD_addrouteにて変更後の経路を追加してください。

以下に設定例を示します。

デフォルト経路としてディスクリプタ1を設定する場合：
dstaddr=0　netmask=0　pd=1

210.2.3.0/24の経路としてディスクリプタ2を設定する場合：
dstaddr=0xD2020300　netmask=0xFFFFFF00　pd=2

ブロードキャストの経路としてディスクリプタ5を設定する場合：
dstaddr=0xFFFFFFFF　netmask=0xFFFFFFFF　pd=5

3.4.2 PPD_delroute 経路情報を削除する

【T】

C言語インタフェース

```
ER ercd = PPD_delroute (PPD_ROUTE *p_route);
```

パラメータ

PPD_ROUTE	*p_route	PPP経路情報を格納した領域のアドレス

リターンパラメータ

ER	ercd	リターン値またはエラーコード

パケットの構造

```
typedef struct ppd_route{
        UW      dstaddr;        経路の宛先IPアドレス
        UW      netmask;        経路のサブネットマスク
        W       pd;             （未使用）
PPD_ROUTE;
```

リターン値／エラーコード

E_OK	正常終了
E_PAR	パラメータエラー（p_routeが0または4の倍数以外）
E_OBJ	オブジェクト状態不正（一致する経路がない）

解　説

PPPの経路情報から、指定した経路を削除します。

dstaddrには、削除対象となる経路のネットワークまたはホストのIPアドレスを指定します。
netmaskには、削除対象となる経路のネットマスクを指定します。
pdは使用しません。

p_routeに0、または4の倍数以外の値を指定した場合は、エラーコードとしてE_PARを返します。
指定された経路が経路情報に存在しない場合は、エラーコードとしてE_OBJを返します。

3.4.3 PPD_refroute 現在の経路情報を参照する

【T】

C言語インタフェース

```
ER ercd = PPD_refroute (PPD_ROUTE *p_route, W size);
```

パラメータ

PPD_ROUTE	*p_route	PPP経路情報を格納する領域の先頭アドレス
W	size	経路情報を格納できる領域の数

リターンパラメータ

ER	ercd	リターン値またはエラーコード

パケットの構造

```
typedef struct ppd_route{
        UW      dstaddr;        経路の宛先IPアドレス
        UW      netmask;        経路のサブネットマスク
        W       pd;             送信先ディスクリプタ番号
PPD_ROUTE;
```

リターン値／エラーコード

0以上	格納した経路情報の数
E_PAR	パラメータエラー（p_routeが0または4の倍数以外、sizeが不正）
E_NOMEM	メモリ不足（sizeが経路情報の数より小さい）

解　説

PPPの現在の経路情報を参照します。

p_routeには、経路情報を格納する領域の先頭アドレスを指定します。１経路につき、12バイトの領域が必要となります。
sizeには、p_routeに格納可能な経路情報の数を設定します。

dstaddrには、経路のネットワークまたはホストのIPアドレスを返します。
netmaskには、経路のネットマスクを返します。dstaddr、およびnetmaskが0の場合、デフォルト経路を表します。
pdには、経路の送信先となるディスクリプタ番号を返します。

p_routeに0、または4の倍数以外の値を指定した場合は、エラーコードとしてE_PARを返します。
経路情報の数が、sizeで指定された分より多い場合、size分の経路情報を格納し、エラーコードとしてE_NOMEMを返します。
経路情報が１つも設定されていない場合は、0を返します。

3.5 コールバックルーチン

3.5.1 ppd_callback（仮称） PPP の接続完了／エラー通知

【T】

C言語インタフェース

```
void ppd_callback (W pd, FN fncd, VP *p_parblk);
```

パラメータ

W	pd	ディスクリプタ
FN	fncd	機能コード（PPD_CBK_OPENED (0x223)固定）
VP	*p_parblk	パラメータブロックを格納した領域の先頭アドレス

リターンパラメータ

無し

パケットの構造

```
typedef struct ppd_secrets{
        ER      rtncd;          リターン値またはエラーコード
        UW      localaddr;      決定した自分（PPPサーバ側）のIPアドレス
        UW      remoteaddr;     決定した相手（PPPクライアント側）のIPアドレス
        UH      mru;            決定した相手（PPPクライアント側）の最大受信データ長（MRU）
        H       perrno;         EV_PROT発生時の詳細エラー情報
} PPD_OPENED;
```

解　説

クライアントとのPPPリンクが確立、または失敗したことを通知します。

本コールバックはPPD_openにてcallbackルーチンのアドレスを指定し、かつノンブロッキング指定の場合（tmout=TMO_WBLKまたはpd=0指定）にのみ呼び出されます。

pdにはリンクが確立、または失敗したディスクリプタ番号が渡されます。

リンクが確立した場合は、rtncdにE_OKが渡されます。また、localaddr、remoteaddr、mruには、ネゴシエーションにて決定した自分（PPPサーバ側）のIPアドレス、相手（PPPクライアント側）のIPアドレス、相手（PPPクライアント側）の最大受信データ長が渡されます。

リンクに失敗した場合は、rtncd、perrnoにエラーコードが渡されます。詳細エラーコードの種類に関してはPPD_openを参照してください。

本コールバックルーチンはPPPサーバタスクからサブルーチンとして呼び出されます。そのため、コールバックルーチン内で待ちになるサービスコールの発行や時間のかかる処理の実行は行わないでください。また、コールバックルーチンから復帰する場合は、タスクの状態を呼び出されたときの状態に戻してから復帰してください。

3.5.2 ppd_callback（仮称） PPP の切断完了通知

【T】

C言語インタフェース

```
void ppd_callback (W pd, FN fncd, VP *p_parblk);
```

パラメータ

W	pd	ディスクリプタ
FN	fncd	機能コード（PPD_CBK_CLOSED(0x224)固定）
VP	*p_parblk	未使用(0固定)

リターンパラメータ

無し

解　説

PPD_closeの処理が完了したことを通知します。

本コールバックはPPD_closeにてノンブロッキングコールが指定された場合（tmout=TMO_WBLK、またはpd=0指定）にのみ呼び出されます。

pdにはクローズが完了したディスクリプタ番号が渡されます。

本コールバックルーチンはPPPサーバタスクからサブルーチンとして呼び出されます。そのため、コールバックルーチン内で待ちになるサービスコールの発行や時間のかかる処理の実行は行わないでください。また、コールバックルーチンから復帰する場合は、タスクの状態を呼び出されたときの状態に戻してから復帰してください。

3.5.3 ppd_callback（仮称） PPP の切断通知

【T】

C言語インタフェース

void ppd_callback (W pd, FN fncd, VP *p_parblk);

パラメータ

W	pd	ディスクリプタ
FN	fncd	機能コード（PPD_CBK_DISCONNECT(0x221)固定）
VP	*p_parblk	未使用(0固定)

リターンパラメータ

無し

解　説

PPPコネクションが切断されたことを通知します。

本コールバックは接続状態のPPPコネクションが相手から切断された場合、または相手からのLCPエコー応答がなく、PPPサーバが自動切断した場合に呼び出されます。

PPD_closeにより切断した場合には呼び出されません。

pdには切断されたディスクリプタ番号が渡されます。

本コールバックルーチンはPPPサーバタスクからサブルーチンとして呼び出されます。そのため、コールバックルーチン内で待ちになるサービスコールの発行や時間のかかる処理の実行は行わないでください。また、コールバックルーチンから復帰する場合は、タスクの状態を呼び出されたときの状態に戻してから復帰してください。

4. 物理ドライバインタフェース

物理ドライバインタフェースは、PPPサーバがPPPクライアントとの通信で使用するシリアルドライバ等の物理ドライバを実装するために準備されています。実際に実装する物理トライバはユーザが準備しなければなりません。

この章ではPPPサーバの物理ドライバインタフェース仕様について説明します。尚、物理ドライバはユーザが作成し、そのアドレスがPPPサーバに渡されるため、モジュール名はユーザ任意となります。このリファレンスマニュアルでは仮称としてppd_PutPhysical、ppd_GetBufPhysical、ppd_RelBufPhysicalを用いています。

マルチセッションでの通信をサポートするため、PPPサーバでは各セッションをディスクリプタ番号（1～PPP_DESCRIPTOR）によって区別します。PPPサーバが物理ドライバをコールする際、引数としてディスクリプタ番号が渡されます。物理ドライバでは、ディスクリプタ番号に対応する物理デバイスに対して送受信を行ってください。

図 4-1にPPPフレーム送信シーケンスを図 4-2にPPPフレーム受信シーケンスを示します。尚、PPPフレーム送信シーケンスとPPPフレーム受信シーケンスは各々が同時に（全２重で）動作する必要があります。

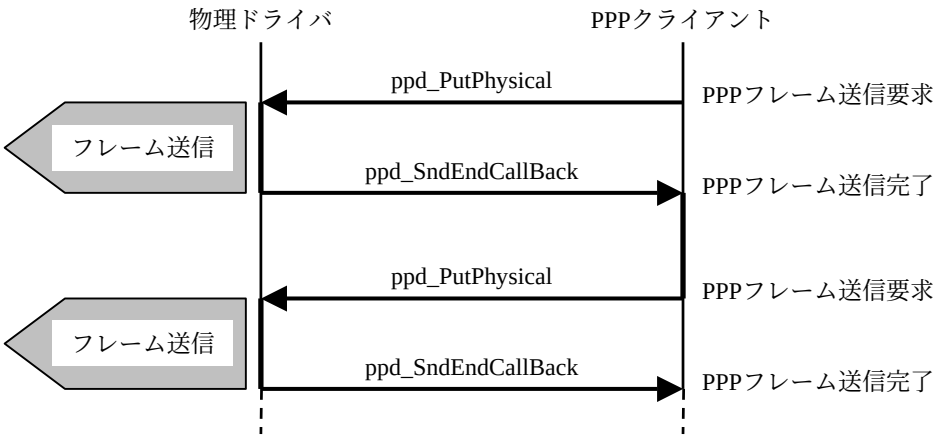


図 4-1　PPPフレーム送信シーケンス

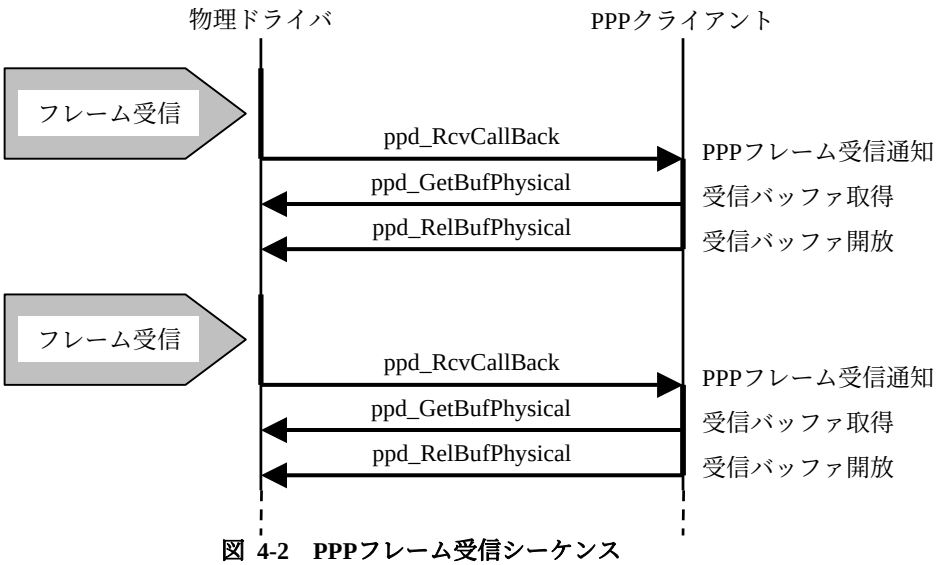


図 4-2　PPPフレーム受信シーケンス

4.1 物理ドライバ送受信関数

4.1.1 ppd_PutPhysical（仮称） PPP フレームを送信する

【T】

C言語インタフェース

ER ercd = ppd_PutPhysical (W pd, UB *buf, INT len);

パラメータ		
W	pd	ディスクリプタ
UB	*buf	送信するフレームの先頭アドレス
INT	len	送信するフレームの長さ

リターンパラメータ		
ER	ercd	リターン値またはエラーコード

リターン値／エラーコード	
E_OK	正常終了

解　説

PPPフレームを送信します。

pdにはディスクリプタ番号が渡されます。ディスクリプタ番号はマルチセッションにてPPPを利用する際に各セッションを区別する為の番号です。マルチセッションでPPPを使用する場合は、ディスクリプタ番号に対応した物理デバイスからPPPフレームを送信してください。

PPPサーバは、フレーム単位でこの関数を呼び出します。なお、PPPサーバは、フレームの前後に H’7E を付けてPPPフレームとして送信を行います。フレーム前後の H’7E が不要な物理インタフェースの場合は、物理ドライバで H’7E を削除してください。

要求されたフレームの送信が完了したら、PPD_SndEndCallBackを呼び出してください。

PPPフレームの送信を受付けたら、送信の完了を待たずにただちにリターンしてもかまいません。PPPサーバは、PPD_SndEndCallBackが呼び出されるまで送信バッファの再利用は行いません。

4.1.2 ppd_GetBufPhysical（仮称） 受信バッファ情報を取得する

【T】

C言語インタフェース

ER_UINT ercd = ppd_GetBufPhysical (W pd, UB **rcvbuf, UB **headp, UB **tailp);

パラメータ		
W	pd	ディスクリプタ
UB	**rcvbuf	受信したデータの先頭アドレスを返す領域のアドレス
UB	**headp	受信バッファの先頭アドレスを返す領域のアドレス
UB	**tailp	受信バッファの終了アドレスを返す領域のアドレス

リターンパラメータ		
ER_UINT	ercd	リターン値またはエラーコード

リターン値／エラーコード	
正の値	受信したデータの長さ

解 説

PPPフレームの受信データ情報を取得します。
pdにはディスクリプタ番号が渡されます。マルチセッションでPPPを利用する場合は、ディスクリプタ番号に対応した物理デバイスから受信したデータを返してください。

物理ドライバがPPD_RcvCallBackを呼び出すと、PPPサーバは本関数（ppd_GetBufPhysical）を使って受信データの情報を取得します。
また、PPPサーバは本関数（ppd_GetBufPhysical）で取得した受信データの処理が終了したらppd_RelBufPhysical関数を呼び出して、受信バッファの開放を行います。
1回の本関数（ppd_GetBufPhysical）呼び出しで取得した受信データ領域は、1回のppd_RelBufPhysical関数で開放するとは限りません。PPPサーバは受信データ領域を複数回に分けて開放する場合もあるので、注意してください。

rcvbufの示す領域には、受信データの先頭アドレスを返してください。
headpの示す領域には、受信バッファの先頭アドレスを返してください。
tailpの示す領域には、受信バッファの終了アドレス（バッファの最終アドレス＋1）を返してください。
リターン値には、受信データの長さを返してください。

PPPサーバは、rcvbufの示すアドレスから受信データの長さ分データを取得しますが、途中でtailpの示すアドレスに達した場合、続きのデータをheadpの示すアドレスから取得するリングバッファ対応の処理を行います。受信バッファがリングバッファでない場合は、受信バッファ情報として「headp=rcvbuf」、「tailp=rcvbuf+受信データ長」を設定して返してください。

PPPサーバは、受信データがフレーム単位でなくても、受信データ中の H’7E から H’7E までを1つのPPPフレームとして認識することができますが、1つのフレームの受信を分割して行うとオーバヘッドが大きくなるため、できるだけフレーム単位で受信データを通知するようにしてください。

PPPサーバは、データの H’7E から H’7E までを1つのPPPフレームとして認識します。フレームの前後に H’7E が付かない物理インタフェースからの受信データには、フレームの前後に H’7E を付けてからPPPサーバに渡すようにしてください。

物理ドライバからPPPサーバに通知する受信バッファがリングバッファの場合は、受信が発生する毎にPPD_RcvCallBack呼び出しを行ってもかまいません。PPPサーバは受信データを処理できる状態になったときに本関数（ppd_GetBufPhysical）を使って受信バッファ情報を取得します。この場合の受信バッファ情報としては、その時点での最新の受信バッファ情報を返してください。

物理ドライバからPPPサーバに通知する受信バッファが受信ごとに独立した複数のバッファになっている場合は、一度PPD_RcvCallBackを呼び出した後の次のPPD_RcvCallBack呼び出しは、PPPサーバが本関数（ppd_GetBufPhysical）で取得した受信データ領域を全て開放するまで行わないでください。

4.1.3 ppd_RelBufPhysical（仮称） 受信バッファを解放する

【T】

C言語インタフェース

ER ercd = ppd_RelBufPhysical (W pd, INT buflen);

パラメータ

W	pd	ディスクリプタ
INT	buflen	開放する受信バッファの長さ

リターンパラメータ

ER	ercd	リターン値またはエラーコード

リターン値／エラーコード

E_OK	正常終了

解　説

ppd_GetBufPhysicalで取得した受信データ領域を開放します。

pdにはディスクリプタ番号が渡されます。マルチセッションでPPPを利用する場合は、ディスクリプタ番号に対応した物理デバイスの受信データ領域を解放してください。

buflenには解放する受信バッファ内のデータ長が渡されますので、受信データの先頭からbuflen分のデータを解放してください。buflenに0が渡された場合は受信バッファ内の全てのデータを解放してください。

ppd_GetBufPhysicalで取得した受信データ領域が本関数（ppd_RelBufPhysical）の1回の呼び出しで解放されるとは限りません。

解放前のデータ領域が書き換えられた場合の動作は保証できません。

物理ドライバからPPPサーバに通知する受信バッファがリングバッファの場合は、前回ppd_GetBufPhysical関数で返した受信データ領域が本関数（ppd_RelBufPhysical）によって全て解放される前にPPD_RcvCallBack呼び出しを行ってもかまいません。PPPサーバは前回ppd_GetBufPhysical関数で取得した受信データ領域を全て開放してから、ppd_GetBufPhysical関数によって次の受信データ領域を取得します。この場合の受信バッファ情報としては、その時点での最新の受信バッファ情報を返してください。

物理ドライバからPPPサーバに通知する受信バッファが受信ごとに独立した複数のバッファになっている場合は、一度PPD_RcvCallBackを呼び出した後の次のPPD_RcvCallBack呼び出しは、PPPサーバがppd_GetBufPhysical関数で取得した受信データ領域を全て開放するまで行わないでください。

4.2 物理ドライバコールバック関数

4.2.1 PPD_SndEndCallBack フレームの送信完了通知

【T／D／L／I】

C言語インタフェース

```
void PPD_SndEndCallBack (W pd);
```

パラメータ

W	pd	ディスクリプタ

リターンパラメータ

無し

解　説

PPPフレームの送信完了を通知します。

マルチセッションでPPPを利用する場合は、物理デバイスに対応したディスクリプタ番号をpdに指定してください。

ppd_PutPhysicalで要求されたフレームの送信が完了したことをPPPサーバに通知します。

4.2.2 PPD_RcvCallBack PPP データの受信通知

【T／D／L／I】

C言語インタフェース

void PPD_RcvCallBack (W pd);

パラメータ

W	pd	ディスクリプタ

リターンパラメータ

無し

解　説

PPPデータを受信した事をPPPサーバに通知します。

マルチセッションでPPPを利用する場合は、物理デバイスに対応したディスクリプタ番号をpdに指定してください。

PPPサーバは、受信データがフレーム単位でなくても、受信データ中の H’7E から H’7E までを1つのPPPフレームとして認識することができますが、1つのフレームの受信を分割して通知するとオーバヘッドが大きくなるため、できるだけフレーム受信単位で通知するようにしてください。

付録A　　サポートオプション

A.1　　LCPオプション

表A-1　LCPオプション

項番	オプション	タイプ
1	Maximum Receive Unit	1
2	Async Control Character Map	2
3	Authentication Protocol	3
4	Magic Number	5
5	Protocol Field Compression	7
6	Adress and Control Field Compression	8

A.2　　CHAP認証アルゴリズムオプション

項番	オプション	タイプ
1	CHAP with MD5(RFC1994)	5

A.3　　IPCPオプション

表A-2　IPCPオプション

項番	オプション	タイプ
1	IP Address deprecated (RFC1332)	1
2	IP Compression Control (RFC1332)	2
3	IP Address (RFC1332)	3
4	Primary DNS Server Address (RFC1877)	129
5	Secondary NBNS Server Address (RFC1877)	130
6	Secondary DNS Server Address (RFC1877)	131
7	Secondary NBNS Server Address (RFC1877)	132

HI.CommunicationEngine
PPPサーバ リファレンスマニュアル
CM7000PPD02J-4

発行年月　　2006年　9月　　第4版
発　　行　　株式会社 ルネサス北日本セミコンダクタ
編　　集　　株式会社 ルネサス北日本セミコンダクタ