HI.CommunicationEngine
PPPクライアント
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クライアント」について説明します。

HI.CommunicationEngine PPPクライアントはユーザが準備する物理ドライバを経由してPPPサーバへの接続と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クライアント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クライアントは、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サーバと通信します。

PPPクライアントの基本機能は次のとおりです。

（1） 認証手順を経てPPPサーバに接続します。

（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。

本PPPクライアントでは、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 クライアントには、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クライアント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。

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サーバに接続後、認証が完了した後のPPPクライアントは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PPPクライアント間のインタフェースについては、「4. 物理ドライバインタフェース」の項を参照してください。

1.3　構成

PPP クライアント使用時の構成例を図 1-1 に示します。

PPP クライアントを使用する為には、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 クライアントを 1 つのドライバモジュールとして管理します。

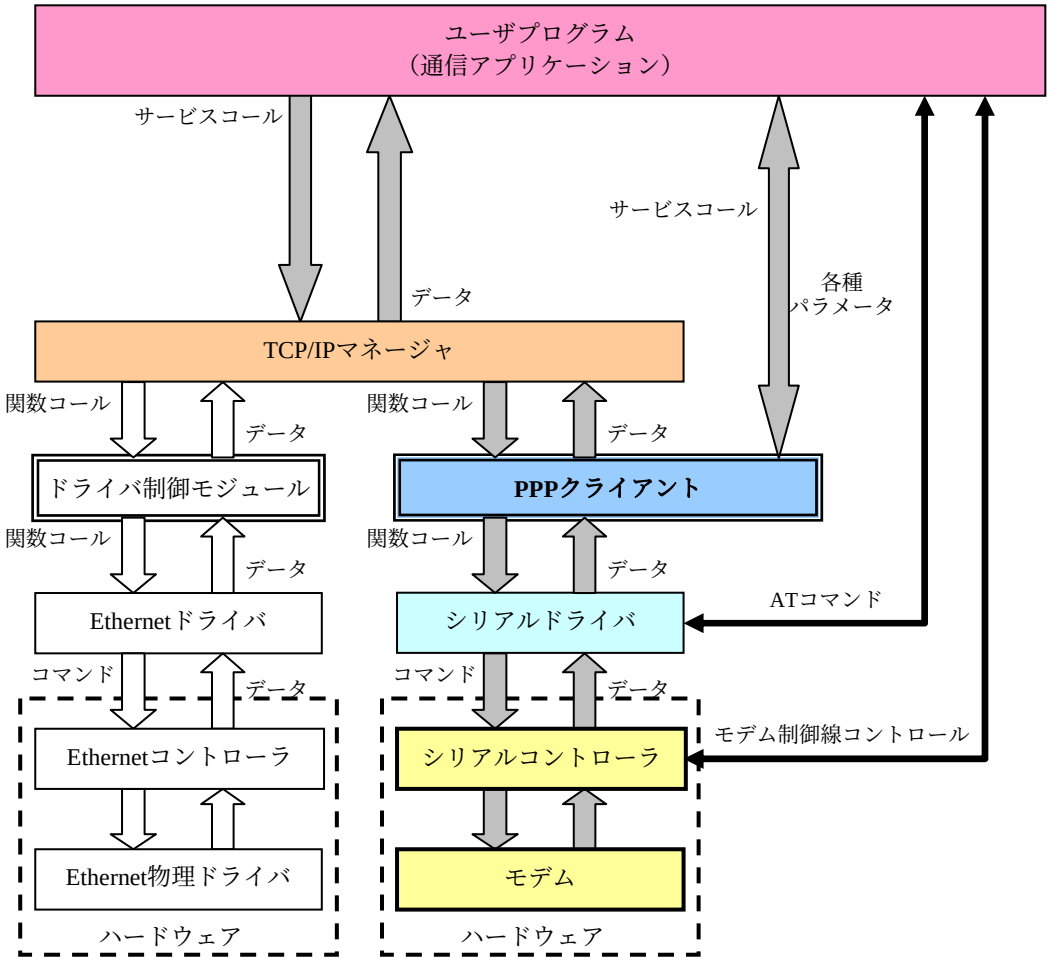


図 1-1　PPPクライアント使用時の構成例

2. PPP クライアント使用方法

2.1 構築時の設定と登録について

PPP クライアントを使うためには、ユーザのプログラム構築時に PPP クライアントの「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。

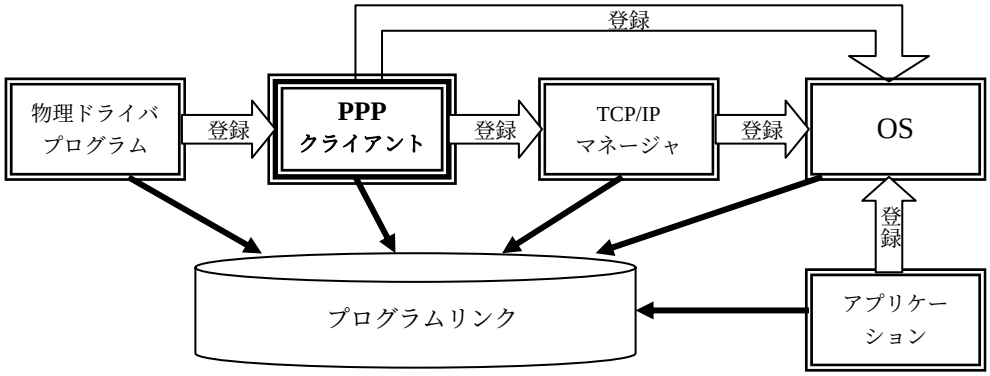


図 2-1 PPPクライアントの構築

2.1.1 タスク情報の設定

PPP クライアントはタスクとして実行されます。また、PPP クライアント内部で 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（pppConfTbl）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 クライアント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 クライアント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 クライアント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 クライアント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 クライアント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クライアントには、（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 サーバ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クライアント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クライアントに「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用物理ドライバエントリテーブル（pppDrvTbl）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クライアントの構築マニュアルを参照してください。

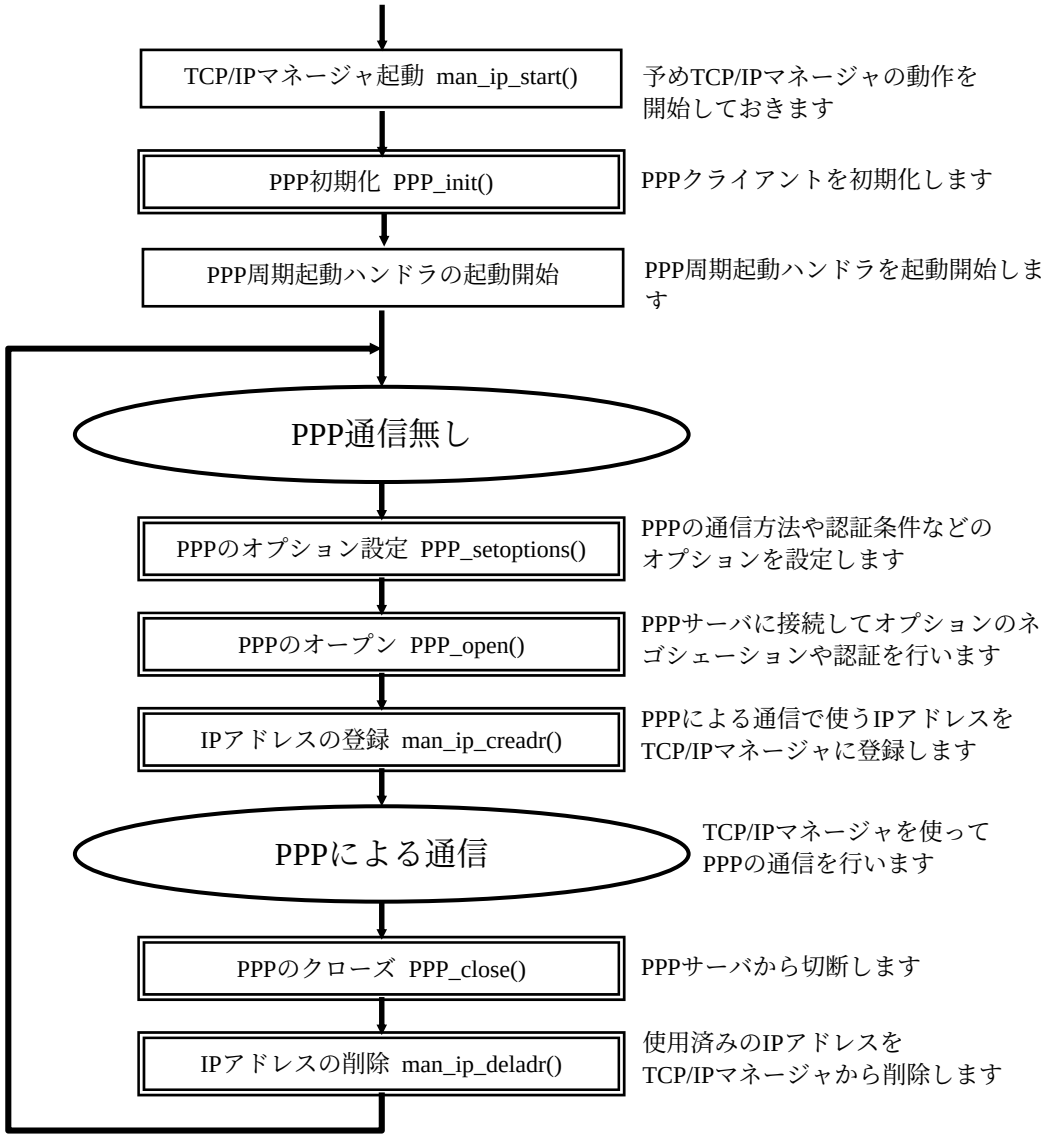
2.1.4 TCP/IP マネージャへの登録

PPPクライアントは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クライアントのエントリアドレステーブル（drv_PPP）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クライアントの構築マニュアルを参照してください。

2.2 運用時の設定と登録について

PPPクライアントを使って通信するためには、予め「PPP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オプション設定」、「PPP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 2-2 基本的なPPPの運用手順

2.2.1 PPP の初期化

PPPクライアントの運用を開始する前に、1回だけPPPの初期化関数（PPP_init）をコールしてください。
初期化関数によってPPPクライアントが初期化され、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クライアント周期起動ハンドラ（PPP_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。PPP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クライアントの初期化終了後、PPPの通信をオープンする前にPPPクライアントのオプションを設定します。PPPクライアントは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サーバ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 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クライアントとPPPサーバ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のオープン（PPP_open）をコールします。

PPPのオープンでは、タスク情報テーブル（pppConfTbl）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クライアント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クライアント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サーバに接続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のオープン完了時には、自分（クライアント）と相手（サーバ）のIPアドレスを取得することができます。

PPP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クライアントとPPPサーバ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クライアント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クライアント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クライアント）とPPPクライアントのIPアドレスの関連付けを行う必要があります。**注：PPPクライアントの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クライアントの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 のクローズ

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

PPPクライアント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クライアントは、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クライアントをPPPサーバ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	PPP_init	PPP クライアントを初期化する
オプション操作	PPP_getoptions	PPP オプションを取得する
	PPP_setoptions	PPP オプションを設定する
接続・切断	PPP_open	PPP をオープン（PPP サーバと接続）する
	PPP_close	PPP をクローズ（PPP サーバから切断）する
状態参照	PPP_status	PPP の現在の状態を取得する

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
切断通知	ppp_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クライアントがPPPサーバ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クライアント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ppp_PutPhysical（仮称）と、PPPフレームの受信用に使用するppp_GetBufPhysical（仮称）、ppp_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クライアント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クライアントに通知するにはPPP_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クライアントに通知するにはPPP_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	ppp_PutPhysical（仮称）	PPP フレームを送信する
	ppp_GetBufPhysical（仮称）	PPP フレーム受信バッファ情報を取得する
	ppp_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	PPP_RcvCallBack	PPP フレームを受信した事を通知する
送信	PPP_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 PPP_init PPP 初期化

【T／L／I】

C言語インタフェース

void PPP_init (void);

パラメータ

無し

リターンパラメータ

無し

解　説

PPPを初期化します。

PPPクライアント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。
PPP_init は、PPPクライアント使用前に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 PPP_setoptions PPP オプションの設定

【T】

C言語インタフェース

```
ER ercd = PPP_setoptions(W pd, PPP_OPTIONS *p_opts, OPTIONS_FLAG flag)
```

パラメータ

W	pd	ディスクリプタ（0固定）
PPP_OPTIONS	*p_opts	PPPオプション情報の先頭アドレス
OPTIONS_FLAG	flag	PPP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 ppp_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      mru;                        MRUの値
        UW      asyncmap;                   ASYNCマップの値
        UW      ouraddr;                    自分（PPPクライアント側）のIPアドレス
        UW      hisaddr;                    相手（PPPサーバ側）のIPアドレス
        UW      old_ipcp : 1,               旧型IPアドレスオプション
                opt_dns_p : 1,              プライマリDNSサーバアドレスオプション
                opt_nbns_p : 1,             プライマリNBNSサーバアドレスオプション
                opt_dns_s : 1,              セカンダリDNSサーバアドレスオプション
                opt_nbns_s : 1;             セカンダリNBNSサーバアドレスオプション
} PPP_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≠0、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）
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クライアントのオプションを設定します。
本サービスコールは、PPP_initを実行した後はいつでも利用することができます。

PPPクライアントは、PPP_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サーバとのネゴシェーションを行います。また、一度設定したPPPオプションはPPP_closeを呼び出すことでデフォルトに設定されます。

PPPサーバに対して要求するオプション（flag＝PPP_WANT）の設定と、PPPサーバ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。

WANT（Flag=PPP_WANT）で1を指定した項目は、PPPサーバとの接続時にPPPクライアントから要求を行う項目です。0を指定した項目は、PPPクライアントからの要求を行いません。
ALLOWED（Flag=PPP_ALLOWED）で1を指定した項目は、PPPサーバとの接続時にPPPサーバ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サーバからの要求を拒否します。

本サービスコールでPPP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	1	1
2	neg_asyncmap	ASYNC マップ	1	1
3	neg_upap	PAP 認証	0 固定※1	1
4	neg_chap	CHAP 認証	0 固定※1	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

※1 PPPクライアント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パラメータ設定値

No.	シンボル名	パラメータの意味	デフォルトの設定値	
			WANT	ALLOWED
1	mru	MRU の値	1500	未使用
2	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 クライアント側）の IP アドレス	0	未使用
2	hisaddr	相手（PPP サーバ側）の IP アドレス	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。

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
2	opt_dns_p	プライマリ DNS サーバアドレスオプション	1	未使用
3	opt_nbns_p	プライマリ NBNS サーバアドレスオプション	1	未使用
4	opt_dns_s	セカンダリ DNS サーバアドレスオプション	1	未使用
5	opt_nbns_s	セカンダリ NBNS サーバアドレスオプション	1	未使用
6	neg_vjcompression	IP ヘッダ圧縮オプション	1	1

asyncmapには、PPPクライアント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クライアント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。

ouraddrやhisaddrに0を指定した場合は、オープン時にPPPサーバからIPアドレスを取得します。
取得したIPアドレスは、PPP_openサービスコール終了後にlocaladdr（PPPクライアントの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、PPP_getoptionsサービスコールをflag＝PPP_GOTで発行することでouraddrからPPPクライアントのIPアドレスを、PPP_getoptionsサービスコールをflag＝PPP_HISで発行することでhisaddrからPPPサーバのIPアドレスを参照することができます。

old_ipcpに1を設定した場合は、旧型のIPアドレスオプション形式にてネゴシエーションを行います。通常は0を指定してください。
ネゴシエーションにより取得したDNSサーバアドレス、およびNBNS（NetBIOS ネームサーバ）アドレスは、PPP_openサービスコール終了後、PPP_statusサービスコールにて取得することができます。

3.2.2 PPP_getoptions PPP オプションの取得

【T】

C言語インタフェース

```
ER ercd = PPP_getoptions(W pd, PPP_OPTIONS *p_opts, OPTIONS_FLAG flag)
```

パラメータ

W	pd	ディスクリプタ（0固定）
PPP_OPTIONS	*p_opts	PPPオプション情報の先頭アドレス
OPTIONS_FLAG	flag	PPP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 ppp_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      mru;                        MRUの値
        UW      asyncmap;                   ASYNCマップの値
        UW      ouraddr;                    自分（PPPクライアント側）のIPアドレス
        UW      hisaddr;                    相手（PPPサーバ側）のIPアドレス
        UW      old_ipcp : 1,               旧型IPアドレスオプション
                opt_dns_p : 1,              プライマリDNSサーバアドレスオプション
                opt_nbns_p : 1,             プライマリNBNSサーバアドレスオプション
                opt_dns_s : 1,              セカンダリDNSサーバアドレスオプション
                opt_nbns_s : 1;             セカンダリNBNSサーバアドレスオプション
} PPP_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≠0、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クライアントのオプションの状態を参照します。
本サービスコールは、PPP_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サーバ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。

オープン後にPPPサーバ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。
オープン後にPPPサーバ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。

PPP_openサービスコールによって、PPPサーバから取得したIPアドレスは、PPP_open のリターンパラメータlocaladdr（PPPクライアントの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、PPP_getoptionsサービスコールをflag＝PPP_GOTで発行することでouraddrからPPPクライアントのIPアドレスを、PPP_getoptionsサービスコールをflag＝PPP_HISで発行することでhisaddrからPPPサーバの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 PPP_open PPP のオープン

【T】

C言語インタフェース

ER ercd = PPP_open (W pd, PPP_PARAM *param, TMO tmout);

パラメータ

W	pd	ディスクリプタ（0固定）
PPP_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 ppp_param{
        UB      *user;          PAP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
        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クライアント）のIPアドレス
        UW      remoteaddr;     決定した相手（PPPサーバ）のIPアドレス
        UH      mru;            受信した相手（PPPサーバ）の最大受信データ長（MRU）
        H       perrno;         詳細エラーコード
} PPP_PARAM;
```

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≠0、paramが0または4の倍数以外，user, passwd, chap_secretが0または文字列が長すぎる、tmout＜-1、tmout＝0、callbackが奇数）
E_OBJ	オブジェクト状態不正（すでにオープン済み、物理ドライバ送信中）
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

PPP_ENO_DESCRIPTOR	1	PPPのディスクリプタが不正
PPP_ENO_USERNAME	2	PAP認証ユーザネームが不正
PPP_ENO_PASSWD	3	PAP認証パスワードが不正
PPP_ENO_SECRET	4	CHAP認証パスワードが不正
PPP_ENO_TMOUT	5	タイムアウト値が不正
PPP_ENO_CBKADR	6	コールバックルーチンのアドレスが不正

E_SYS発生時の詳細エラーコード

PPP_ENO_DRVTBL	11	ドライバテーブル不正
PPP_ENO_CREFLG	12	イベントフラグ生成失敗
PPP_ENO_CRETSK	13	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

PPP_ENO_PAPFAIL	21	PAP認証失敗
PPP_ENO_CHAPFAIL	22	CHAP認証失敗
PPP_ENO_NEGLCP	23	LCPネゴシェーション失敗
PPP_ENO_NEGIPCP	24	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デバイスを通してPPPクライアント－PPPサーバ間における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サーバと接続されている必要があります。

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、localaddrには自分（PPPクライアント側）のIPアドレス、remoteaddrには相手（PPPサーバ側）のIPアドレスを格納し、mruには相手（PPPサーバ側）の最大受信データ長を格納します。

決定したPPPクライアントの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。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。

pdに0以外の値を指定した場合は、perrnoにPPP_ENO_DESCRIPTORを設定し、エラーコードとしてE_PARを返します。

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となります。その他の値を指定した場合は、perrnoにPPP_ENO_TMOUT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、userに0を指定した場合や、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PPP_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、passwdに0を指定した場合や、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PPP_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、chap_secretに0を指定した場合や、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PPP_ENO_SECRETを設定し、エラーコードとしてE_PARを返します。

callbackには、PPPが相手から切断された場合に呼び出すコールバックルーチンのアドレスを指定します。**コールバックルーチンは自分から切断した場合やタイムアウトで切断した場合は呼び出されません**。

callbackが奇数の場合は、perrnoにPPP_ENO_CBKADRを設定し、エラーコードとしてE_PARを返します。また、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。

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PPP_ENO_DRVTBLを設定し、エラーコードとしてE_SYSを返します。

PPPクライアント用のイベントフラグ生成に失敗した場合は、perrnoにPPP_ENO_CREFLGを設定し、エラーコードとしてE_SYSを返します。

PPPクライアントタスクの生成に失敗した場合は、perrnoにPPP_ENO_CRETSKを設定し、エラーコードとしてE_SYSを返します。

すでにPPPコネクションがオープンしている場合、または物理ドライバが送信中（前回、物理ドライバに送信したパケットに対する送信完了コールバックがない）の場合は、エラーコードとしてE_OBJを返します。

PAPによる認証に失敗した場合は、perrnoにPPP_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PPP_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PPP_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PPP_ENO_NEGIPCPを設定し、エラーコードとしてEV_PROTを返します。

3.3.2 PPP_close PPP のクローズ

【T】

C言語インタフェース

```
ER ercd = PPP_close (W pd);
```

パラメータ

W	pd	ディスクリプタ（0固定）

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_PAR	パラメータエラー（pd≠0）

解　説

PPPのコネクションをクローズします。

PPPクライアントー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0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。

3.3.3 PPP_status PPP の状態を取得する

【T】

C言語インタフェース

ER ercd = PPP_status (W pd, PPP_STATUS *p_st);

パラメータ		
W	pd	ディスクリプタ（0固定）
PPP_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 ppp_status{
        STAT    pppstat;        PPPのステータス
        UW      dnsaddr[2];     取得したDNSサーバアドレス
        UW      nbnsaddr[2];    取得したNBNSサーバアドレス
 PPP_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≠0、p_stが0または4の倍数以外）

解　説

PPPの状態、およびオプションパラメータを取得します。

pdに0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。
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	PPP_ST_OPENNING	1	オープン処理中（PPP_open()実行中）
2	PPP_ST_CONNECT	2	PPP接続状態
3	PPP_ST_CLOSING	3	クローズ処理中（PPP_close()実行中）
4	PPP_ST_CLOSE	4	PPP切断状態

dnsaddr[0]にはPPPサーバから取得できたプライマリDNSサーバアドレスを返します。
dnsaddr[1]にはPPPサーバから取得できたセカンダリDNSサーバアドレスを返します。
nbnsaddr[0]にはPPPサーバから取得できたプライマリNBNSサーバアドレスを返します。
nbnsaddr[1]にはPPPサーバから取得できたセカンダリNBNSサーバアドレスを返します。
これらの内容は、PPP_open()処理完了前、または取得できなかった場合は0を返します。

3.4 コールバックルーチン

3.4.1 ppp_callback（仮称） PPP の切断通知

【T】

C言語インタフェース

```
void ppp_callback (W pd, FN fncd);
```

パラメータ

W	pd	ディスクリプタ（0固定）
FN	fncd	機能コード（PPP_CBK_DISCONNECT(0x201)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に呼び出されます。
PPPコネクションに失敗した場合や自分から切断した場合には呼び出されません。

本コールバックルーチンはPPPクライアント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クライアントがPPPサーバ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クライアント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クライアントに渡されるため、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ppp_PutPhysical、ppp_GetBufPhysical、ppp_RelBufPhysicalを用いています。

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。

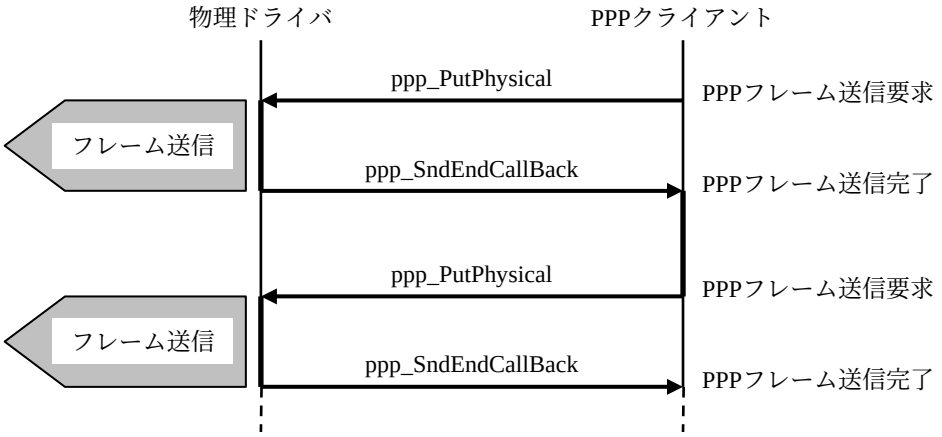


図 4-1 PPPフレーム送信シーケンス

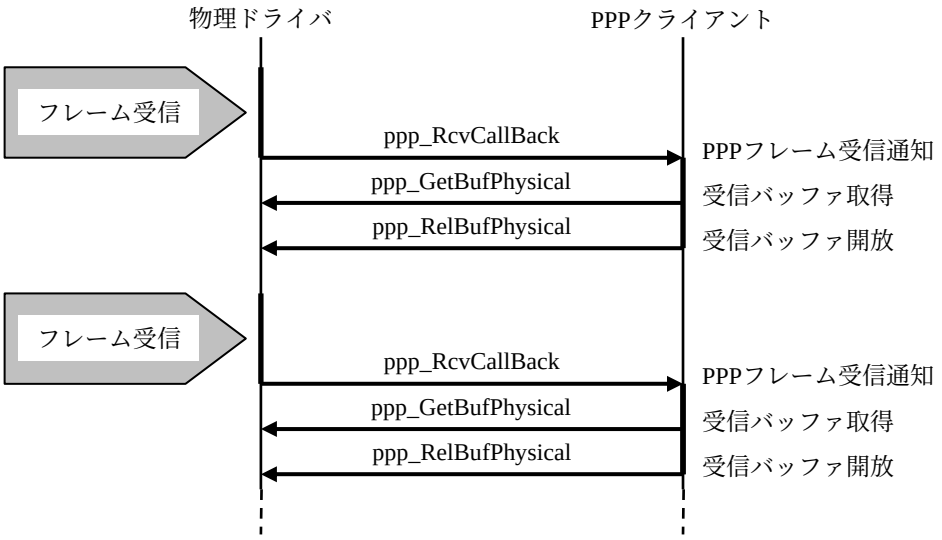


図 4-2 PPPフレーム受信シーケンス

4.1 物理ドライバ送受信関数

4.1.1 ppp_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 = ppp_PutPhysical (UB *buf, INT len);

パラメータ		
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。

PPPクライアントは、フレーム単位でこの関数を呼び出します。なお、PPPクライアントは、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、PPP_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クライアントは、PPP_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 ppp_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 = ppp_GetBufPhysical (UB **rcvbuf, UB **headp, UB **tailp);

パラメータ		
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。

物理ドライバがPPP_RcvCallBackを呼び出すと、PPPクライアントは本関数（ppp_GetBufPhysical）を使って受信データの情報を取得します。

また、PPPクライアントは本関数（ppp_GetBufPhysical）で取得した受信データの処理が終了したらppp_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（ppp_GetBufPhysical）呼び出しで取得した受信データ領域は、1回のppp_RelBufPhysical関数で開放するとは限りません。PPPクライアント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クライアントは、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クライアントは、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クライアントは、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クライアントに渡すようにしてください。

物理ドライバからPPPクライアント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PPP_RcvCallBack呼び出しを行ってもかまいません。PPPクライアントは受信データを処理できる状態になったときに本関数（ppp_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クライアント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PPP_RcvCallBackを呼び出した後の次のPPP_RcvCallBack呼び出しは、PPPクライアントが本関数（ppp_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 ppp_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 = ppp_RelBufPhysical (INT buflen);

パラメータ		
INT	buflen	開放した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

ppp_GetBufPhysicalで取得した受信データ領域を開放します。

ppp_GetBufPhysicalで取得した受信データ領域が本関数（ppp_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クライアント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ppp_GetBufPhysical関数で返した受信データ領域が本関数（ppp_RelBufPhysical）によって全て解放される前PPP_RcvCallBack呼び出しを行ってもかまいません。PPPクライアントは前回ppp_GetBufPhysical関数で取得した受信データ領域を全て開放してから、ppp_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クライアント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PPP_RcvCallBackを呼び出した後の次のPPP_RcvCallBack呼び出しは、PPPクライアントがppp_GetBufPhysical関数本関数（ppp_GetBufPhysical）で取得した受信データ領域を本関数（ppp_GetBufPhysical）で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　PPP_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 PPP_SndEndCallBack (W pd);

パラメータ

W	pd	ディスクリプタ（0固定）

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。

ppp_PutPhysicalで要求されたフレームの送信が完了したことをPPPクライアントに通知します。

4.2.2 PPP_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 PPP_RcvCallBack (W pd);

パラメータ

W	pd	ディスクリプタ（0固定）

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クライアントに通知します。

PPPクライアントは、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	Primary NBNS Server Address (RFC1877)	130
6	Secondary DNS Server Address (RFC1877)	131
7	Secondary NBNS Server Address (RFC1877)	132

HI.CommunicationEngine
PPPクライアント リファレンスマニュアル
CM7000PPP02J-6

発行年月　　2011年 4月　　第6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