CUB-PCI Rev 1.01

ユーザーズマニュアル

2004,7,28

STEP TECHNICA CO.,LTD

はじめに

この度は、弊社 CUB-PCI をお買い上げ頂き誠にありがとうございます。本製品の御利用にあたり本ユーザーズマニュアルを良くお読みいただき、十分な御理解の上、完全に使いこなして頂ければ弊社メーカーといたしましても大変喜ばしい限りでございます。

尚、本ユーザーズマニュアルの PDF 書類及び弊社製品の最新情報がインターネットのホームページから入手可能です。是非、常に最新の情報をご確認ください。

HomePage URL http://www.steptechnica.com/

【ご注意】

○　本製品の使用、保管の際は静電気および衝撃などに十分に注意してお取り扱い願います。

○　パソコンの電源（POWER）スイッチを“ON”にしたままで本製品およびケーブルなどの脱着を行わないでください。

○　本製品本体の拡張バスコネクタの金メッキ部分には絶対に手を触れないでください。動作不良の原因になります。

○　本製品上の MKY40 に関する詳細につきましては、「CUnet ユーザーズマニュアル」（ステップテクニカ）などをご参照ください。

○　本製品の改造およびその使用にともなう弊害につきましては、当社は一切責任を負いかねますのでご了承願います。

○　本マニュアルに掲載される製品名は、一般に開発メーカの商標または登録商標です。

株式会社ステップテクニカ

1　ハードウェア

1.1　特徴

CUB-PCI は、ステップテクニカ製 MKY40 チップを 1 基搭載した PCI 拡張バス対応の CUnet 通信ボードです。主に、MKY40 の評価および学習の利用をターゲットとしており、付属の Windows2000 用のライブラリと併せて利用することで、MKY40 の機能を簡単に利用しやすいように設計されています。

コネクタには、8pin のモジュラコネクタを採用しており、10BASE-T/100BASE-TX 用の市販の UTP ストレートケーブルで動作を評価することができるようになっています（※）。

CUB-PCI での経験の多くは、MKY40 を搭載したマイコンシステムでも、そのまま活かす事ができます。

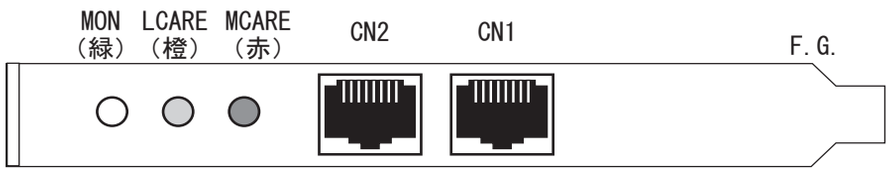
（※動作の学習・評価以外での UTP ケーブルの使用は推奨いたしません。耐ノイズ性能を高める為には別に指定する STP ケーブルをご使用ください。）

1.2　仕様

名　称	CUB-PCI
搭載チップ	MKY40 × 1基
通信方式	CUnet通信方式
転送レート	3M/6M/12M bps
対応バス	PCI Ver2.1に準拠した、32ビット・33MHz拡張バス
占有リソース	4KBの連続したメモリエリア（PnPにて自動割当）
割り込み	1ライン使用（PnPにて自動割当）
コネクタ	RJ-45 モジュラコネクタ（ﾋﾛｾ TM11R-LF-88）
電　源	DC +5V
消費電流	500mA以下
使用条件	温度0～50℃　湿度20～90%（非結露）
サイズ	122mm(W)×107mm(H)　※パネル部含まず

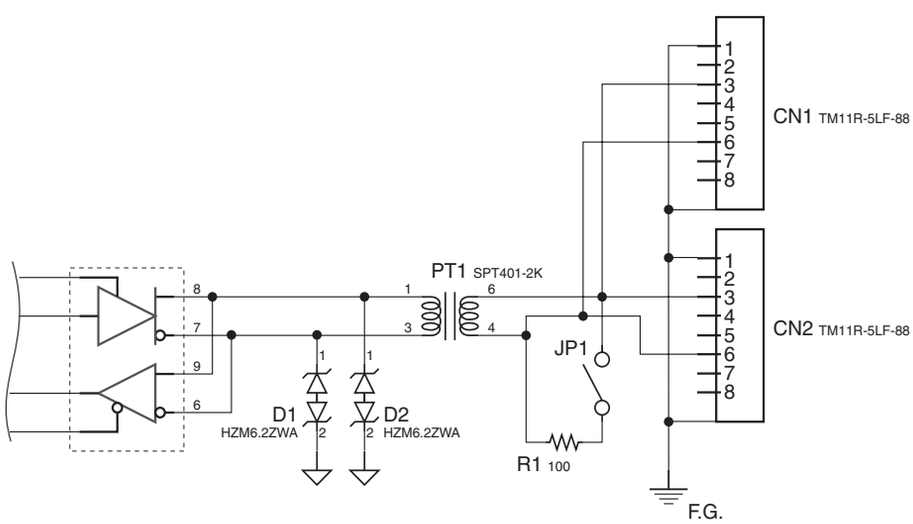
1.3　コネクタ仕様

CUB-PCI のパネル面とその詳細を以下に示します。



MON	MKY40の同名の端子に接続したLED(緑)。MKY40の通信開始とともに点灯。
LCARE	MKY40の同名の端子に接続したLED(橙)。LCARE発生時に点灯。 MKY40の設定によっては視認できないこともあります。
MCARE	MKY40の同名の端子に接続したLED(赤)。MCARE発生時に点灯。 MKY40の設定によっては視認できないこともあります。
CN1,CN2	CUnet通信ライン。市販の10BASE-T/100BASE-TX用UTPストレートケーブルで評価可能です。EartherNetボードとの誤接続に注意して下さい。
F.G.	パネルは、CN1,CN2の1番ピン及びシェルに接続され、基板本体のGNDとは絶縁されています。“F.G.”とは便宜上の名前であり、筐体を通して適切にグランド処理しなければ、浮いた状態になってしまいます。

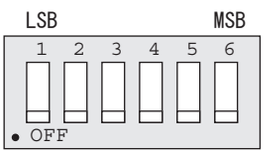
CN1,CN2 の結線および、通信ドライバ・レシーバ周りの回路図を以下に示します。



1.4　ディップスイッチ

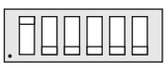
CUB-PCI には 4 個のディップスイッチがあります。これらのスイッチの設定は、JP1 によるターミネーションの設定を除いてソフトウェアによって設定することも可能です。S.A. や OWN などを予め固定的に設定して置きたい場合などに有効です。

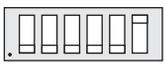
また、複数の CUB-PCI を同一の機器でコントロールする場合、各 CUB-PCI に違う S.A.(OWN) を設定して置き、その値を読み込むことで、ソフトウェアから特定の CUB-PCI デバイスを見つけることが出来ます。ただし、このような使い方をした場合、必要に応じて実際に使用する S.A.(OWN) は、ソフトウェアで設定し直すことになります。

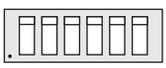


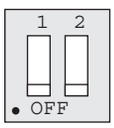
S.A./OWN

Ex.


= 1


= 32


= 63



BPS

Ex.


= 12Mbps


= 6Mbps


= 3Mbps


= 設定禁止

JP1

Ex.


= Termination DISABLE


= Termination ENABLE

JP1 によるターミネーション抵抗設定は通信ラインの物理的な始点・終点の 2 箇所のみ有効にしてください。

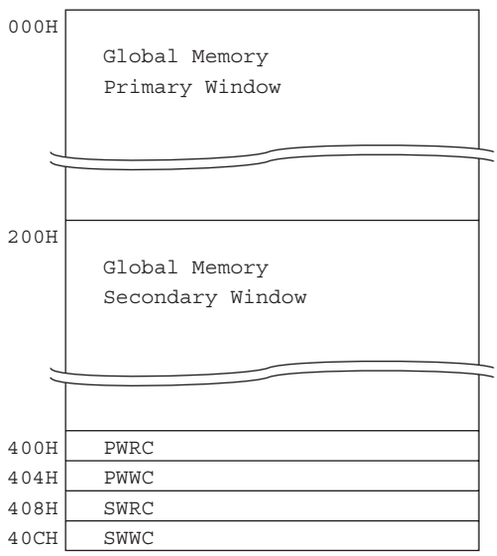
ターミネーション抵抗設定は、伝送距離に直接影響します。

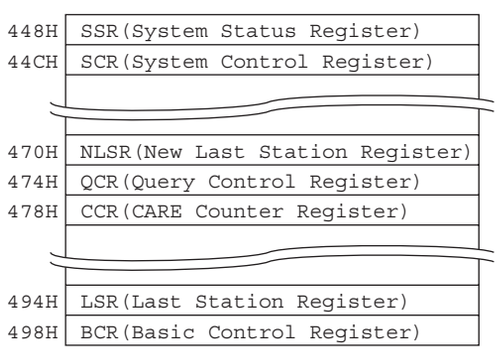
1.5. メモリマップ

以下に、CUB-PCI のメモリマップを示します。

メモリマップ中のアドレスはCUB-PCIの先頭アドレスからの相対値であり、実際のアドレスはボードの先頭アドレス値を加算したアドレスになります。

各エリアの詳細は、『CUnet ユーザーズマニュアル』をご参照ください。





410H	RFR	[31:0]
414H	(Receive Flag Register)	[63:32]
418H	LFR	[31:0]
41CH	(Link Flag Register)	[63:32]
420H	DRFR	[31:0]
424H	(Data Renewal Flag Reg.)	[63:32]
428H	CFR	[31:0]
42CH	(Connection Flag Reg.)	[63:32]
430H	DRCR	[31:0]
434H	(Data Renewal Check Flag Reg.)	[63:32]
438H	LGR	[31:0]
43CH	(Link Group Register)	[63:32]
440H	MGR	[31:0]
444H	(Member Group Reg.)	[63:32]

450H	INT0C(INT0 Control)
454H	INT1C(INT1 Control)
458H	INT2C(INT2 Control)
45CH	INT0S(INT0 Status)
460H	INT1S(INT1 Status)
464H	INT2S(INT2 Status)
468H	ITC0(Interrupt Timing Control 0)
46CH	ITC1(Interrupt Timing Control 1)

800H	CHIP RESET FLAG

47CH	MSC(Mail Send Control)
480H	MSL(Mail Send Limit)
484H	MES(Mail Error Status)
488H	MSR(Mail Send Result)
48CH	MRC0(Mail Receive Control 0)
490H	MRC1(Mail Receive Control 1)
500H	MSB(Mail Send Buffer)
600H	MRB0(Mail Receive Buffer 0)
700H	MRB1(Mail Receive Buffer 1)

CHIP RESET FLAG (800H)

	801H								800H							
Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	RST	INT2	INT1	INT0	-	-	-	-
Initial Value:	X	X	X	X	X	X	X	X	0	0	0	0	X	X	X	X
Read/Write:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W

アドレス 800H のビット 7 は“CHIP RESET FLAG”です。

このフラグに 1 を書き込むことで、MKY40 チップの $\overline{\text{RST}}$ 端子へリセット信号を印加する事が出来ます。リセットの終了に伴い、フラグも自動的に 0 に書き戻されます。

INT2 ～ INT0 は、MKY40 の同名の各端子の状態を表すリード専用ビットです。割り込みを使用する場合、割り込み処理ルーチン内で割り込みの種類を判別するために参照します。

現在､初期値不定 (X) で､書き込みも可能になっているビットは､将来の予約となっていますので､これらのビットを操作してはいけません。

具体的には、リセットの際は、一度 800H をバイトリードし、その値とリセットのための“80H”を論理 OR した値を再び 800H にバイトライトします。

802H,803H はリード専用で常に FFH が読めます。800H ～ 803H の 4 バイトは 800H 以降の全エリアにリード可能なイメージがありますが、将来のためにアクセスしないようお願いします。

2　付属ソフトウェア

2.1　概要

CUB-PCI 付属 CD-R 内の“Dll”フォルダには、Microsoft Windows Me/2000 Pro に対応したドライバ及び DLL が含まれております。

初めて CUB-PCI を搭載して PC を立ち上げる時、Windows のハードウェア追加ウィザードに従い、付属 CD-R より WDM ドライバをインストールしてください。尚、Windows 2000 では、デバイスドライバのインストールにはアドミニストレータ権限が必要になりますので、ログインの際にご注意ください。

Windows の仕様上、ユーザアプリケーションから CUB-PCI へのアクセスは、全て付属のデバイスドライバを通して行われます。しかし、デバイスドライバ呼び出しは複雑な処理が必要なので、それらをラッピングして簡略化したインターフェイスを提供するのが CUBPCI.DLL に含まれる CUB API です。

Microsoft Windows Me/2000 をホスト OS として利用される場合は、これらの付属ソフトウェアを利用することで、Microsoft Visual Basic をはじめ、多数のプログラミングツールを使って簡単に MKY40 をコントロールすることができます。

今後も、対応 OS の拡大や、アップバージョンのリリース等も予定しておりますので、随時、弊社 web ページ（http://www.steptechnica.com/）にてご確認ください。

2.2　著作権・免責

本製品付属 CD-R に収められた、全てのドキュメント・プログラム・プログラムソースの著作権は、株式会社ステップテクニカが所有しています。株式会社ステップテクニカは、以下の注意事項を了承された個人・法人、または、その他の団体が弊社製品 CUB-PCI を利用する場合に限り、これら著作物の複製・利用をする権利をライセンスするものであり、株式会社ステップテクニカに無断でこれら著作物の一部または全部を改訂・再配布したり、上記以外の目的のために複製・利用することはできません。

【注意事項】

- **本製品付属 CD-R 内のソフトウェア及び、弊社 web ページより入手した全てのソフトウェアの使用による、いかなる結果に対しても弊社は一切責任を負いません。**
- **ライブラリの説明は、よく読み、正しくお使いください。**
- **仕様・内容は、将来予告無く変更になる場合があります。弊社は、将来への互換性について、一切保証いたしません。**
- **弊社製品以外の OS や開発環境等に関するお問い合わせはサポートいたしかねます。**
- **バグ・不具合などを発見された方は、弊社技術部までご連絡ください。**

技術部 e-mail: wish@steptechnica.com

2.3　ファイルの種類

付属 CD-R の “Dll” フォルダに収められたファイルは以下のとおりです。

CubPci.dll

DLL 本体です。Windows のシステムフォルダか、本 DLL を使用するユーザプログラムと同じディレクトリにコピーしてお使いください。

CubPci.lib

Microsoft Visual C++ 用のインポートライブラリです。
同バージョン 5.0 で作成した物です。

CubPci.h

DLL のヘッダファイルです。ご使用の際は、Windows.h より後ろにインクルードしてください。

2.4 使用方の概略

CUBPCI.DLL を使いコンピュータに搭載された CUB-PCI デバイスをコントロールするためには、必要な初期化処理と終了処理があります。その際の手順を以下に示します。

```
// DLL バージョン 1.xx 用に作ったアプリケーションの場合
    int Version = CubGetVersion();
    if( Version < 0x100 || Version > 0x1FF ) {
            printf(" 互換性の無いバージョンの CUBPCI.DLL です。\n");
            exit(1);
    }

//1. コンピュータに搭載された CUB-PCI デバイスの数を取得します。
// (1 枚しか搭載されていないことが分かっている場合は、省略しても構いません。)
    const int Count = CubCountDevice();
    if( Count < 1 ) {
            printf("CUB-PCI が存在しません。\n");
            exit(1);
    }

//2.CUB-PCI デバイスへのハンドルを取得します。
// (1 枚しか搭載されていないことが分かっている場合は、引数を 0 でオープンします。)
    HANDLE  CubHandle[Count];
    for( int i = 0; i < Count; i++ ) {
            CubHandle[i] = CubOpenHandle( i );
            if( CubHandle[i] == INVALID_HANDLE_VALUE ) exit(1);
    }
//
// この位置で CUB-PCI の制御を行えます。
//
//3. コントロールの終了したハンドルをクローズします。
    for( int i = 0; i < Count; i++ ) {
            CubCloseHandle( i );
    }
```

2.5 制限事項

本ライブラリ内のAPIは複数スレッドから同時に使用することはできません。アプリケーションをマルチスレッド構成にする場合、同時呼び出しが起こらないよう配慮してください。

2.6 API仕様

CubGetVersion

書式

UINT CUBAPI CubGetVersion(void)

説明

ライブラリのバージョンを調べます。

パラメータ

なし

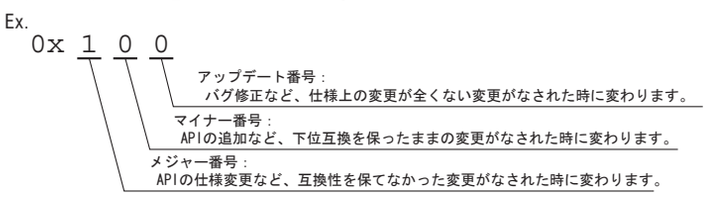
戻り値

ライブラリのバージョンを表す符号なし整数値。

備考

必須ではありませんが、CUBPCI.DLLを利用するユーザアプリケーションで、DLLに対する互換性チェックを行うことで安全性を高めることが出来ます。ここでいう安全性とは、互換性の無い関数コールをあらかじめ避けることでプログラムの強制終了などを避けることを意味します。

CubGetVersion()APIはDLLのバージョンを取得するAPIです。このAPIの返す16進数のバージョン番号には以下の意味があります。



上記のとおり、16進2桁目以下の数字は互換上無視して構いませんが、3桁目より上位の値が変わっている場合、CUBPCI.DLLのAPIをコールしないことをお奨めします。この互換性チェックは、初期化処理よりも先に行う必要があります。

CubCountDevice

書式

```
INT CUBAPI CubCountDevice( void );
```

説明

存在するCUB-PCIデバイスの数を調べます。CUB-PCIの枚数が既に分かっている場合は、呼び出す必要はありません。

この関数で1以上の戻り値が得られない場合は、CubOpenHandle() も必ず失敗します。

パラメータ

なし

戻り値

-1	10枚以上。
0	1枚も存在しない。
1 ～ 9	1 ～ 9枚。

※10枚以上を利用される場合は、弊社技術部にお問い合わせください。

CubGetLastError

書式

```
UINT CUBAPI CubGetLastError( void );
```

説明

プロセスが最後に呼び出したCUB APIの終了状態を調べます。

パラメータ

なし

戻り値

CubPci.hで以下のように定義されています。

```
文字定数                    値
CUB_SUCCESS                 0       /* 正常終了。*/
CUB_ERR_DEVICENOTEXIST      1       /* デバイスが存在しない。*/
CUB_ERR_ALREADYOPENED       2       /*すでにオープンされている。*/
CUB_ERR_CLOSED              3       /* CubOpenHandle() が一度もコールされていない。*/
CUB_ERR_INVALIDPARAM        4       /* 無効なパラメータでコールされた。*/
CUB_ERR_NORESOUCE           5       /* 実行に必要なリソースが足りない。*/
CUB_ERR_FAILED              6       /* 原因不明により処理が遂行されなかった。*/
CUB_NOTCALLYET              99      /* まだ1度もCUBAPIがコールされていない。*/
```

CubOpenHandle

書式

```
HANDLE CUBAPI CubOpenHandle( int Instance );
```

説明

指定したインスタンスのCUB-PCIボードへのハンドルを返します。同じインスタンスから複数のハンドルを得ることも可能です。現バージョンでCUB_ERR_ALREADYOPENEDによる失敗はありません。

これ以降に解説するCUB APIはすべて、CubOpenHandle()で取得したハンドルを使って呼び出します。また、全ての処理が終わったあとは、CubCloseHandle()を呼び出してハンドルを閉じてください。

パラメータ

Instance 0基点のボードの通し番号

複数ボードが存在する時に、各ボードのハンドルを個別に得るためのボード番号であり、搭載されているボードが1枚のみであれば、0を引数にしてオープンします。

戻り値

成功時にはHANDLE型のハンドルが返されます。失敗時にはINVALID_HANDLE_VALUEが返され、CubGetLastError()で原因を確認します。

INVALID_HANDLE_VALUE文字定数はwindows.hをインクルードすることで使用可能です。

CubCloseHandle

書式

```
BOOL CUBAPI CubCloseHandle( HANDLE CUBHandle );
```

説明

CubOpenHandle()で取得したハンドルを閉じます。

パラメータ

CUBHandle　　閉じるハンドル

戻り値

成功すればTRUE、失敗したらFALSEを返します。

CubReadByte,CubReadWord,CubReadDWord

書式

```
BOOL CUBAPI CubReadByte( HANDLE CUBHandle, ULONG Adr, BYTE* Dat )
BOOL CUBAPI CubReadWord( HANDLE CUBHandle, ULONG Adr, WORD* Dat )
BOOL CUBAPI CubReadDWord( HANDLE CUBHandle, ULONG Adr, DWORD* Dat )
```

説明

CUB-PCI の指定アドレスからバイト / ワード / ダブルワードデータを取得します。

パラメータ

CUBHandle	対象となる CUB-PCI へのハンドル
Adr	ボード先頭からのオフセットアドレス。ワードアクセスでは 2 の倍数、ダブルワードアクセスでは 4 の倍数でなくてはいけません。
Dat	取得した値を格納するバイト / ワード / ダブルワード領域へのポインタ

戻り値

成功すれば TRUE を返し、失敗したら FALSE を返します。

CubWriteByte,CubWriteWord,CubWriteDWord

書式

```
BOOL CUBAPI CubWriteByte( HANDLE CUBHandle, ULONG Adr, BYTE Dat )
BOOL CUBAPI CubWriteWord( HANDLE CUBHandle, ULONG Adr, WORD Dat )
BOOL CUBAPI CubWriteDWord( HANDLE CUBHandle, ULONG Adr, DWORD Dat )
```

説明

CUB-PCI の指定アドレスへバイト / ワード / ダブルワードデータを書き込みます。

パラメータ

CUBHandle	対象となる CUB-PCI へのハンドル。
Adr	ボード先頭からのオフセットアドレス。ワードアクセスでは 2 の倍数、ダブルワードアクセスでは 4 の倍数でなくてはいけません。
Dat	書き込むバイト / ワード / ダブルワードデータ。

戻り値

成功すれば TRUE を、失敗したら FALSE を返します。

CubGetBlock

書式

```
BOOL CUBAPI CubGetBlock( HANDLE CUBHandle, ULONG Bno, void* Dat )
```

説明

プライマリウインドーを利用して、グローバルメモリの指定したプロックに対しデータハンドリングされた読み込みを行います。読み込むデータは、必ず 1 ブロック（8 バイト）になります。格納先のサイズに注意してください。

データハンドリングについての詳細は、「CUnet ユーザーズマニュアル」をご覧ください。

パラメータ

CUBHandle	対象となる CUB-PCI へのハンドル。
Bno	先頭ブロックを 0 とし、何ブロック目を読み込むのかを指定します。(0 ～ 63)
Dat	読み込んだ 1 ブロック（8 バイト）のデータの格納場所へのポインタ。

戻り値

成功すれば TRUE を、失敗したら FALSE を返します。

CubSetBlock

書式

```
BOOL CUBAPI CubSetBlock( HANDLE CUBHandle, ULONG Bno, void *Dat )
```

説明

プライマリウインドーを利用して、グローバルメモリの指定したプロックに対しデータハンドリングされた書き込みを行います。書き込むデータは、必ず 1 ブロック（8 バイト）になります。書き込むデータのサイズに注意してください。

データハンドリングについての詳細は、「CUnet ユーザーズマニュアル」をご覧ください。

パラメータ

CUBHandle	対象となる CUB-PCI へのハンドル。
Bno	先頭ブロックを 0 とし、何ブロック目に書き込むのかを指定します。(0 ～ 63)
Dat	書き込む 1 ブロック（8 バイト）のデータの格納場所へのポインタ。

戻り値

成功すれば TRUE を、失敗したら FALSE を返します。

MEMO

株式会社ステップテクニカ
〒358-0011　埼玉県入間市下藤沢757-3
TEL 04-2964-8804　　FAX 04-2964-7653