USB-CUnet 製品別マニュアル

2011/09/12

目次

CUnet とは....2
USB-CUnet の概要....2
ブロックイメージ....3
ネットワーク構成例....4
ハードウェア....5
ソフトウェア....7
usbcunet.dll 関数....9
Visual Basic 参照宣言例....16
CUnet Monitor (CUMON.EXE)....17
関係資料....21
諸注意....22
グローバルメモリアドレス表....24

CUnet とは

CUnet は、ステップテクニカ社のオリジナル通信方式であり、シリアル通信回線で接続された複数の拠点間で、同一メモリ空間をリアルタイムで共有するリモートメモリ共有技術です。

USB-CUnet は CUnet デバイス「MKY40」を搭載、パソコンと CUnet 機器間でメモリの共有を実現します。

USB-CUnet の概要

Cunet のグローバルメモリは 512byte です。8byte を 1 ブロックとして、64 ブロックを SA0～SA63 として管理します。

各 CUnet ステーションは、すべてのエリアの値を読むことができます。対して、書き込みエリアは、開始ブロックのステーションアドレス(sa)と占有ブロック数(ow)で指定します。usbcunet.dll の初期化コマンドでは以下のようにします。

```
cunet_init 5 4 31     'USB-CUnet 初期化
      5    sa = 開始ブロック          SA5～
      4    ow = 占有ブロック数        4ブロック占有する。(SA8まで)
     31    en = 共有最終ブロック      共有エリアを31までに限定する
```

この例では、SA5～SA8 が書き込み可能となります。

このエリアに対応するアドレスは、IN/OUT(※)では 2040～2071 です。このため、OUT コマンドでは、OUT dat 2043 などと使用することができます。ON/OFF では、2320～2575 が対応することになります。

IN(),SW()はどのステーションからでも使用することができます。

共有ブロックとは、MKY40 が常時通信により、メモリを同期化させるブロック数です。ここでは、31 としているため、SA32～SA63 は、同期しません。

同期エリアを制限すると、全体が同一データになるまでの時間が短縮できます。

512byte 64 ブロックすべてを共有する場合で約 2.5msec

256byte 32 ブロックでは、1.2msec

16byte 2 ブロックでは 102μ 秒となります。 必要に応じて調整します。

※IN,OUT,SW,ON,OFF は MPC の言語表現です。usbcunet.dll では cunet_in, cunet_out, cunet_sw, cunet_on, cunet_off が対応します。

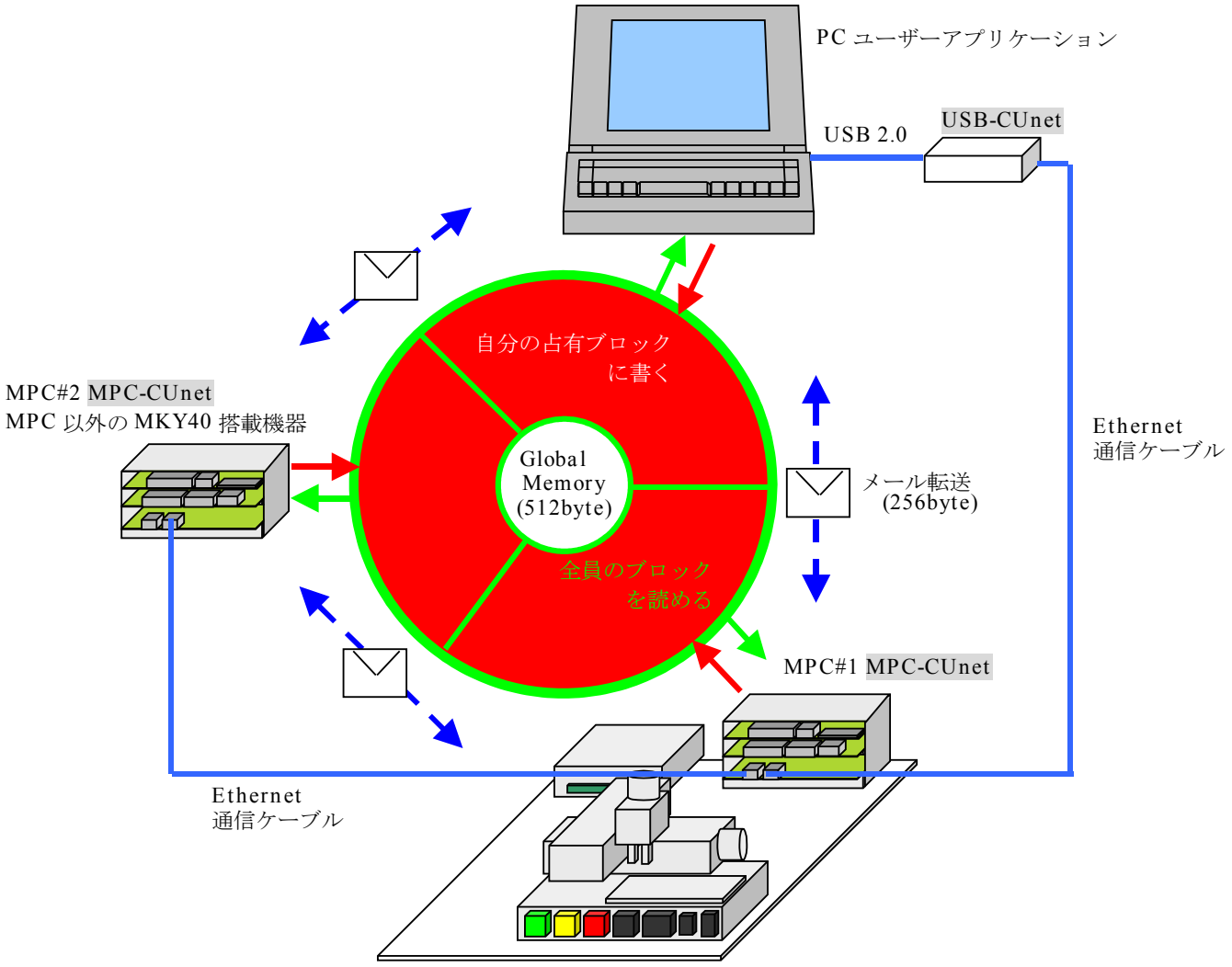
※CUnet を使って MPC のプログラムの読み込み・保存・編集はできません。

ブロックイメージ

SA	IN/OUT バンク番号	ON/OFF/SW ビット番号	例) cunet_init 5,4,31 と設定すると (MPCコマンドでは CUNET 5,4,31)
0	2000～2007	2000～2063	
1	2008～2015	2064～2127	
2	2016～2023	2128～2191	
3	2024～2031	2192～2255	
4	2032～2039	2256～2319	
5	2040～2047	2320～2383	＜占有開始　sa=5
6	2048～2055	2384～2447	占有数 ow=4 書込可能
7	2056～2063	2448～2511	
8	2064～2071	2512～2575	＜ ここまで占有
9	2072～2079	2576～2639	
10	2080～2087	2640～2703	
11	2088～2095	2704～2767	
12	2096～2103	2768～2831	
13	2104～2111	2832～2895	
14	2112～2119	2896～2959	
15	2120～2127	2960～3023	読込は全ブロック可能
16	2128～2135	3024～3087	
17	2136～2143	3088～3151	
18	2144～2151	3152～3215	
19	2152～2159	3216～3279	
20	2160～2167	3280～3343	
21	2168～2175	3344～3407	
22	2176～2183	3408～3471	
23	2184～2191	3472～3535	
24	2192～2199	3536～3599	
25	2200～2207	3600～3663	
26	2208～2215	3664～3727	
27	2216～2223	3728～3791	
28	2224～2231	3792～3855	
29	2232～2239	3856～3919	
30	2240～2247	3920～3983	
31	2248～2255	3984～4047	＜ 共有最終 en=31
32	2256～2263	4048～4111	
33	2264～2271	4112～4175	
34	2272～2279	4176～4239	
35	2280～2287	4240～4303	
36	2288～2295	4304～4367	
37	2296～2303	4368～4431	
38	2304～2311	4432～4495	
39	2312～2319	4496～4559	
40	2320～2327	4560～4623	
41	2328～2335	4624～4687	
42	2336～2343	4688～4751	
43	2344～2351	4752～4815	
44	2352～2359	4816～4879	
45	2360～2367	4880～4943	
46	2368～2375	4944～5007	
47	2376～2383	5008～5071	
48	2384～2391	5072～5135	
49	2392～2399	5136～5199	
50	2400～2407	5200～5263	
51	2408～2415	5264～5327	
52	2416～2423	5328～5391	
53	2424～2431	5392～5455	
54	2432～2439	5456～5519	
55	2440～2447	5520～5583	
56	2448～2455	5584～5647	
57	2456～2463	5648～5711	
58	2464～2471	5712～5775	
59	2472～2479	5776～5839	
60	2480～2487	5840～5903	
61	2488～2495	5904～5967	
62	2496～2503	5968～6031	
63	2504～2511	6032～6095	

※　巻末に白表があります。I/O 割付時などにご利用ください。

ネットワーク構成例



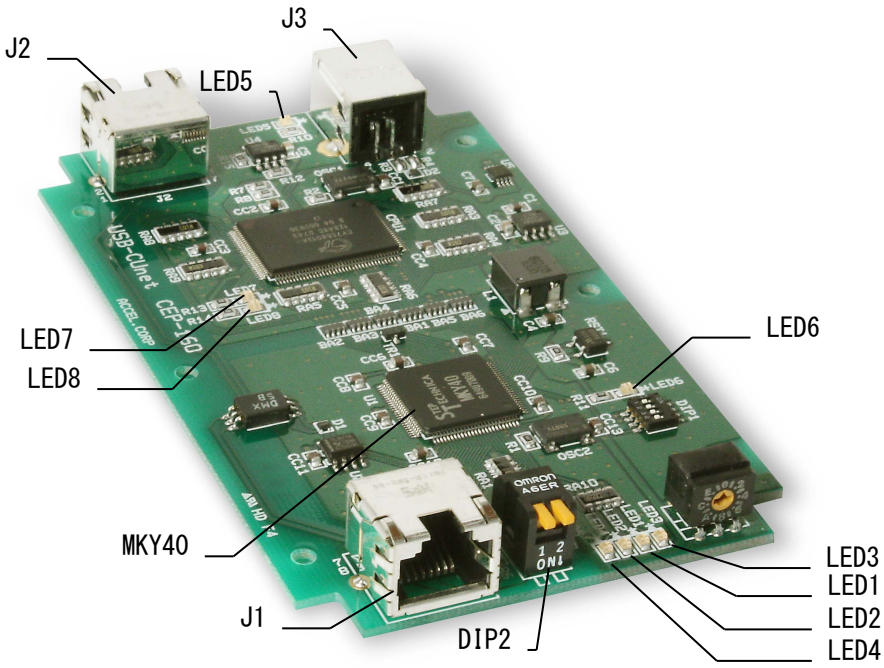
全ステーションで 512 バイトのグローバルメモリを共有します。

グローバルメモリとは別に送受信 256 バイトのメールバッファが有り、任意のステーション間でメール転送ができます。

MPC の I/O が不足した場合、簡単なプログラムで別の MPC をリモート I/O 的に使用できます。

ハードウェア

■ 外観

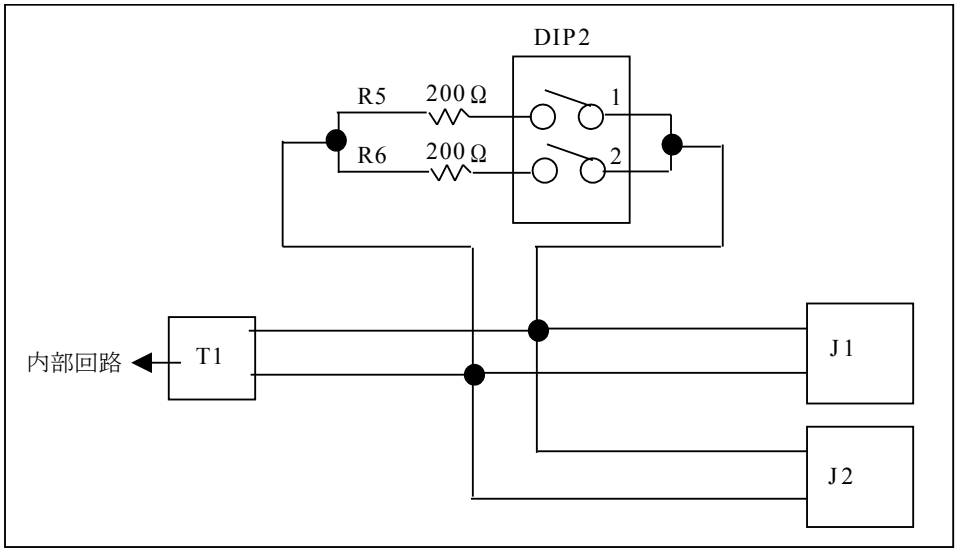


※今後の製品に DSW は実装されません

J1, J2	モジュラーコネクタ	RJ-45 8 極。他局間 CUnet 接続
J3	USB コネクタ	USB-B タイプ 4 芯メス。パソコン間 USB 接続
DIP2	ディップスイッチ	終端抵抗設定。抵抗無し/200Ω/100Ω
LED1 (R)	メンバー減少表示 (MKY40 #MCARE)	メンバステータスが減少したときに、約 50ms 点灯。1 サイクルが 50ms 以内の場合、数サイクルに渡って 1 つのパルスが出力されることになります。
		点灯の可能性： 外来ノイズによる通信障害。
LED2 (R)	リンク切れ表示 (MKY40 #LCARE)	一旦リンクが成立していた装置が、リンク不成立になったことを検出した時に、約 50ms 点灯。1 サイクルが 50ms 以内の場合、数サイクルに渡って 1 つのパルスが出力されることになります。
		点灯の可能性：(正常に共有動作を継続しているステーションで点灯) 通信ケーブルの断線、コネクタの外れ、装置の離脱、装置の電源断、ドライバ/レシーバ部品の故障。 初期化(CUNET, init_cunet)の不適合。
LED3 (G)	サイクル開始通知 (MKY40 #STB)	サイクルの先頭で毎回パルスで通知します。 (※パルスが短くて LED では認識できません)
LED4 (G)	リンク成立表示 (MKY40 #MON)	メンバステータスを持ったステーション(装置)が存在している時に点灯。(他装置とのメモリ共有がコンスタントにできていることを示す)
LED5 (G)	USB 電源	USB 通電中点灯
LED6 (G)	MKY40 電源	MKY40 通電中点灯
LED7 (G)	CPU(CY7C68013)動作	ファームウェア動作中点滅
LED8 (G)	USB アクセス	PC が USB アクセス中点灯

※LED 1～4 の意味参考：「Let's Try! CUnet フィールドエンジニア編」
・ネットワークにおいて異常が発生した場合、どうなりますか?
・MEM モードの MKY40 接続概要図
など

■ 終端抵抗



両端のステーションは終端抵抗を設定して下さい(出荷時 DIP2 は 1,2 とも OFF(抵抗無し))
稼動中 LED1,2 が点灯する場合は通信異常です。終端抵抗の設定で正常化する場合があります。

※参考：「CUnet テクニカルガイド（ネットワーク用）」
・通信ケーブルにおける終端処理の原理
・終端抵抗の具体的な接続
など

■ 通信仕様

- CUnet ステーション間
  RS-485(SN75LBC176D、パルストランスアイソレート)
  転送レート 12Mbps
  Ethernet 通信ケーブル　10BASE-T、Cat3 以上、かつ一括シールドの通信ケーブルの利用を推奨

  ※参考：「CUnet テクニカルガイド（ネットワーク用）」
  ・通信ケーブルの選択
  など

- PC/USB-Cunet 間
  USB2.0

■ **CUnet のパケット受信時の検定**

Cunet 専用 IC には、パケットを受信する際に以下の 3 つの検定を重畳して実行することが義務付けられています。これにより一般的な通信によって生じがちなデータ化けの現象は、全く発生しません。

- フォーマット（構成形式）検定：パケットを受信し終えた瞬間に、特定のスタートパターンから始まるパケットの構成形式が CUnet プロトコルの規約に適合しているかを検定します。
- CRC-16 検定：一般にイーサネット LAN や大容量通信においては、数千～数万ビットの配列に対して 1 つの CRC-12 ブロックチェックコード（BCC）が利用されています。これに対して CUnet プロトコルは、1 つのパケット（数百～数千ビットの配列）に付き CRC-12 よりも検定率の高い CRC-16 ブロックチェックコード（BCC）を採用しています。このブロックチェックコードは、パケットを受信し終えた瞬間に検定されます。
- RZ 信号形式の正当性検定（RZ 検定）：シリアル通信信号のパケットは、“4.1 時間の基準単位”に記述された RZ（Return to Zero：“マンチェスタ符号”とも呼ばれる）のパルス配列から構成されています。CUnet プロトコルにおいては、受信するパケットの信号が RZ の形式を保っているかを 1 ビット単位毎に検定します。

※参考：「CUnet 導入ガイド　（CUnet プロトコル基本解説）」
・データの品質保証
など

ソフトウェア

■ MPC-2000 情報サイト

USB-CUnet のデバイストライバ、TOOL、資料などは弊社ホームページの MPC-2000 情報 に掲載しています。

■ デバイスドライバのインストール

USB-CUnet を接続せずに MPC-2000 情報 から「USB-CUnet Device Driver , DLL」をダウンロードするか 、ACCEL Tools セットアップ「Accel_Setup_jpn.msi」を実行してください。

掲載場所：MPC-2000 サイト ＞ DOWNLOAD ボタン ＞ カテゴリ [TOOL]
ダウンロードファイル例) usbcunet_fw1015_dll1028.zip

Zip ファイルを任意のフォルダで展開すると下記のフォルダが現れます。
Accel_Setup_jpn.msi を実行した場合は C:¥Program Files (x86)¥Accel または C:¥Program Files¥Accel フォルダ内に作成されます。

usbcunet_x64	64 ビット Windows7 用 デバイスドライバ設定フォルダ
usbcunet_x86_wlh	32 ビット Windows7 用 デバイスドライバ設定フォルダ
usbcunet_x86_wxp	32 ビット WindowsXP 用 デバイスドライバ設定フォルダ

各フォルダには次のファイルが入ってします。

Cyload1.spt	ファームウェアスクリプトファイル
cyusb.inf	設定ファイル
CYUSB.SYS	デバイスドライバ
USB-CUnet_readme.txt	改版情報
usbcunet.dll	インターフェースライブラリ

パソコンの USB ポートに USB-CUnet を接続するとウィザードが起動します。パソコンの機種に応じたフォルダを指定してください。

usbcunet.dll は実行アプリケーション(EXE)と同じフォルダか上記の Windows のシステムフォルダにコピーしてください。(DLL は\system32 に置くのが一般的の様です)
ライブラリコマンドについては USB-CUnet_readme.txt、サイト掲載のサンプルプログラム、アプリケーションノートをご覧下さい。

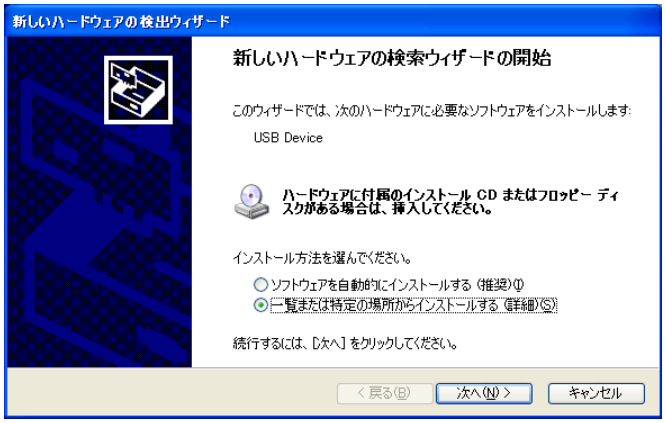
■ ファームウェア、DLL のアップデート

USB-CUnet のファームウェアのアップデートは Cyload1.spt の差し替えで行います。USB-CUnet を非接続でシステムフォルダの Cyload1.spt を差し替えてください。
同じくライブラリも USB-CUnet を非接続で usbcunet.dll を差し替えてください。
バージョンの確認は CUMON.EXE や DLL のコマンドで行えます。

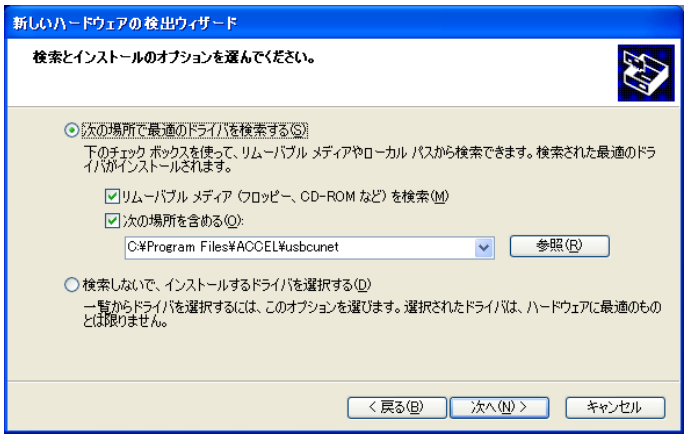
■ XP のインストール例

（Windows7 のインストールは
MPC-2000 情報 ＞ 技術情報 ＞ 「Windows7 MPC 関係ソフトウェアのインストール」 をご参照ください)

XP のインストール作業は 2 回行われます。

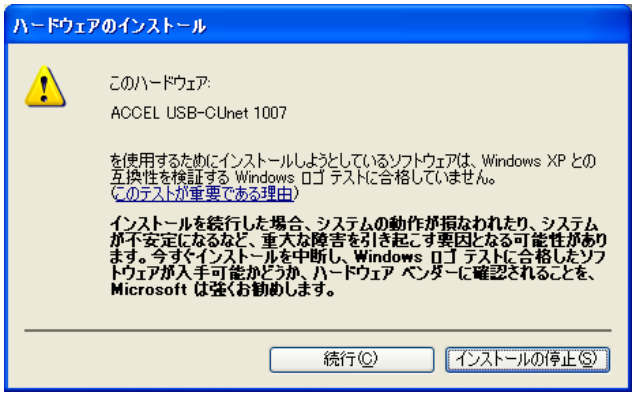


① USB-CUnet を挿入すると
「USB Device」のウィザードが起動します。
「一覧または～」を選択します。



② 「次の場所を含める」にドライバの設定フォルダを指定します。

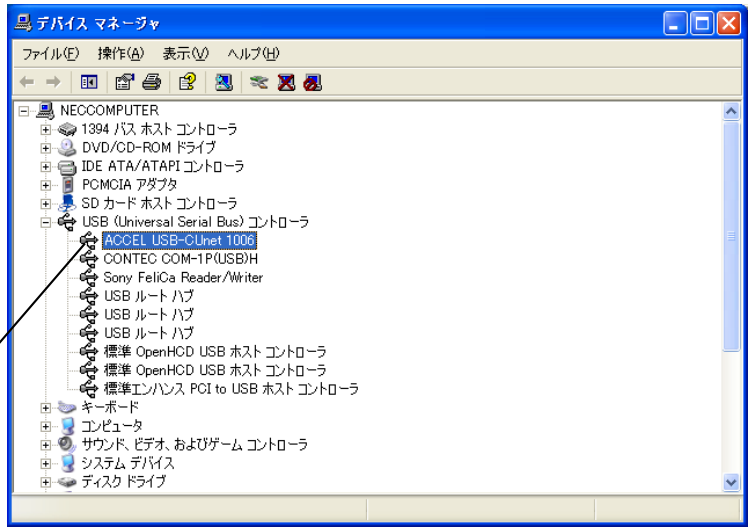
例)C:\Program Files\ACCEL\usbcunet_x86_wxp



③ 「続行」をクリックします。

④「ACCEL USB-CUnet 1007」のインストール完了後、すぐに「ACCEL USB-CUnet 1006」のウィザードが開始されますので、①から同じ手順で繰り返します。

インストール完了後のデバイスマネージャ →



usbcunet.dll 関数

※usbcunet.dll は 32 ビットアプリケーションです。

■ 初期化、各種情報

- cunet_usb_open()

機能
  USB オープン
引数
  無し
戻値
  0:失敗(デバイス認識不能)
  1:成功
  2:失敗(HighSpeed 接続不能(EHCI では無いと思われる))
使用例
```
  If cunet_usb_open() <> 1 Then
     MsgBox("open error")
     Exit Sub
  End If
```

- **cunet_init(sa, ow, en)**

機能
  初期化
引数
  sa:占有するブロックの先頭 SA(Station Address)番号。
    例) SA0～SA3 を占有するなら sa=0,SA12～SA15 なら sa=12,sa=255 とすると MKY40 Reset
  ow:占有するブロック数。
    例) SA0～SA3 を占有するなら ow=4,SA12～SA15 も ow=4
  en:共有する最終ブロック。
    例) SA0～SA15 を共有するなら en=15
戻値
  無し
使用例
```
  cunet_init(255, 0, 0)        'USB-CUnet Reset (これはグローバルメモリをクリアしません)
  Sleep(500)
  cunet_init(0, 4, 7)          'USB-CUnet Initialize (これでグローバルメモリをクリアします)
```

・USB-CUnet 通信レートの変更方法
MPC-CUnet2～USB-CUnet の通信レートは 12Mbps です。上記の設定で 12Mbps に設定されますが、MPC 以外の機器で 3Mbps または 6Mbps に変更するには次のように行います。
```
  cunet_init(&H80 + sa, ow, en)    ' 6Mbps
  cunet_init(&H40 + sa, ow, en)    ' 3Mbps
```
使用例
```
  cunet_init(255, 0, 0)
  Sleep(500)
  cunet_init(&H80 + 4, 4, 7)       ' rate=6M,SA=4,ow=4,en=7
```

- **cunet_fw_vid()**

機能
  VID 取得
引数
  無し
戻値
  VID 番号
使用例
```
  Label_fw_vid.Text = "FW VID: " + Hex$(cunet_fw_vid())
```

- **cunet_fw_pid()**

機能
  PID 取得
引数
  無し
戻値
  PID 番号
使用例
```
  Label_fw_pid.Text = "FW PID: " + Hex$(cunet_fw_pid())
```

- **cunet_fw_ver()**

機能
  ファームウェアのバージョン番号取得
引数
  無し
戻値
  バージョン番号
使用例
```
Label_fw_ver.Text = "FW VER: " + CStr(cunet_fw_ver())
```

- **cunet_dll_ver()**

機能
  DLL のバージョン番号取得
引数
  無し
戻値
  バージョン番号
使用例
```
Label_dll_ver.Text = "DLL VER: " + CStr(cunet_dll_ver())
```

■ グローバルメモリ Read/Write

- **cunet_in(adr, siz)**

機能
  パラレル読み込み(全てのブロックを参照可能)
引数
  adr:バンク番号(2000~)
  siz:読み込みサイズ(1:1byte 符号無、 2:2Byte 符号付、 4:2Byte 符号無、 8:4Byte 符号付)
戻値
  パラレル入力値
使用例
```
Label_in.Text = cunet_in(TextBox_in.Text, 1)
```

- **cunet_out(dat, adr, siz)**

機能
  パラレル書き込み(占有したブロックのみ)
引数
  dat:書き込みデータ
  adr:バンク番号(2000~)
  siz:書き込みサイズ(1:1byte 符号無、 2:2Byte 符号付、 4:2Byte 符号無、 8:4Byte 符号付)
戻値
  無し
使用例
```
cunet_out(TextBox_out_dt.Text, TextBox_out_bnk.Text, 1)
```

- **cunet_sw(adr)**

機能
  ビット読み込み(全てのブロックを参照可能)
引数
  adr:ビット番号(2000~)
戻値
  0:オフ
  1:オン
使用例
```
Label_sw.Text = cunet_sw(TextBox_sw.Text)
```

- **cunet_on(adr)**

機能
  ビットオン(占有したブロックのみ)
引数
  adr:ビット番号(2000~)
戻値
  無し
使用例
```
cunet_on(TextBox_on_off.Text)
```

- **cunet_off(adr)**

機能
　ビットオフ(占有したブロックのみ)
引数
　adr:ビット番号(2000~)
戻値
　無し
使用例
　cunet_off(TextBox_on_off.Text)

- **cunet_in_blk(stadr, length, ar)**

機能
　グローバルメモリ バルク読み込み(全てのブロックを参照可能)
引数
　stadr:読み込み開始バンク番号(2000~)
　length:読み込み長(バイト)
　ar:格納配列
戻値
　無し
使用例
　Dim ar(512) As Integer
　cunet_in_blk(2000, 512, ar(0))

- **cunet_out_blk(stadr, length, ar)**

機能
　グローバルメモリ バルク書き込み(占有したブロックのみ)
引数
　stadr:書き込み開始バンク番号(2000~)
　length:書き込み長(バイト)
　ar:書込む配列
戻値
　無し
使用例
　Dim ar(512) As Integer
　cunet_out_blk(2000, 512, ar(0))

■ MPC 間メール転送

CUnet のメール機能を使い PC～MPC 間で点・MBK・I/O エリアのデータ転送を行いなます。MPC は CU_POST コマンドを実行します。

- **cunet_post_pnt(dest_sa, ar_top, send_ar)**

機能
　MPC-CUnet ステーションへ点データを送信
引数
　dest_sa:送信先 SA
　ar_top :送信先の P()先頭番号
　send_ar:送信する配列
戻値
　-1:MSC レジスタ SEND が下がらない
　0 以上:MESR(Mail Error Status Register)の内容(0 なら正常)
使用例
　Dim ar(60) As Integer
　res = cunet_post_pnt(4, 1000, ar(0))
注意 MPC の点データには XYZU の順番で入ります
　PC 側:　　ar(0)　ar(1)　ar(3)　ar(2)
　　　　　　↓　　↓　　↓　　↓
　MPC 側:　P(1000) X=20000 Y=20001 U=20003 Z=20002

- **cunet_post_mbk(dest_sa, ar_top, send_ar)**

機能
　MPC-CUnet ステーションへ MBK データを送信
引数
　dest_sa:送信先 SA
　ar_top :送信先の MBK()先頭
　send_ar:送信する配列
戻値
　-1:MSC レジスタ SEND が下がらない
　0 以上:MESR(Mail Error Status Register)の内容(0 なら正常)
使用例
　Dim ar(120) As Integer
　res = cunet_post_mbk(4, 5000, ar(0))

- **cunet_post_mbk_lng(dest_sa, ar_top, send_ar)**

機能
  MPC-CUnet ステーションへ MBK データをロング(4byte 長)で送信
引数
  dest_sa:送信先 SA
  ar_top :送信先の MBK()先頭
  send_ar:送信する配列
戻値
  -1:MSC レジスタ SEND が下がらない
  0 以上:MESR(Mail Error Status Register)の内容(0 なら正常)
備考
  タッチパネルのデータ長をロングにした場合に使用
使用例
```
  Dim ar(60) As Integer
  res = cunet_post_mbk_lng(4, 5000, ar(0))    → 対応する MPC の関数は MBK(5000~Lng) など
```

- **cunet_post_io(dest_sa, ar_num, send_data)**

機能
  MPC-CUnet ステーションへ I/O データを送信
引数
  dest_sa :送信先 SA
  ar_num   :送信先の I/O バンク番号
  send_data:送信する変数
戻値
  -1:MSC レジスタ SEND が下がらない
  0 以上:MESR(Mail Error Status Register)の内容(0 なら正常)
備考
  ar_num の 1Byte に書き込みます。負の数値を入れるとメモリー I/O へ書き込みます。
使用例
```
  res = cunet_post_io(4, -1, dt)
```

- **cunet_post_io_blk(dest_sa, ar_top, send_ar)**

機能
  MPC-CUnet ステーションへ I/O データを送信
引数
  dest_sa:送信先 SA
  ar_top :送信先の I/O 先頭バンク番号
  send_ar:送信する配列
戻値
  -1:MSC レジスタ SEND が下がらない
  0 以上:MESR(Mail Error Status Register)の内容(0 なら正常)
備考
  ar_top から 240Byte 連続で書き込みます。負の数値を入れるとメモリー I/O に書き込みます。
使用例
```
  Dim ar(0 To 239) As Integer
  res = cunet_post_io_blk(4, -1, ar(0))
```

- **cunet_post_pnt_str(dest_sa, ar_top, send_str)**

機能
  MPC-CUnet ステーションへ PNT 文字列を送信
引数
  dest_sa :送信先 SA
  ar_top :送信先の P()先頭
  send_str:送信する文字列(max239 キャラクタ)
戻値
  -1:MSC レジスタ SEND が下がらない
  0 以上:MESR(Mail Error Status Register)の内容(0 なら正常)
使用例
```
  Dim s As String
  Dim res As Integer
  s = TimeOfDay() + vbNullChar
  res = cunet_post_pnt_str(4, 2000, s)
  If res <> 0 Then
      MsgBox("error res=" + CStr(res))
  End If
```
MPC プログラム例
```
  S$=P$(2000)              /* P(2000)～ 文字列読み込み
```

- **cunet_post_mbk_str(dest_sa, ar_top, send_str)**

機能

```
  MPC-CUnetステーションへMBK文字列を送信
引数
  dest_sa :送信先SA
  ar_top  :送信先のMBK()先頭
  send_str:送信する文字列(max239キャラクタ)
戻値
  -1:MSCレジスタSENDが下がらない
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
使用例
  Dim s As String
  Dim res As Integer
  s = TimeOfDay() + vbNullChar
  res = cunet_post_mbk_str(4, 2000, s)
  If res <> 0 Then
      MsgBox("error res=" + CStr(res))
  End If
MPCプログラム例
  S$=MBK$(2000,18)        /* MBK(2000)～ 文字列読み込み,18文字
```

- **cunet_req_pnt(req_sa, ar_top, rcv_ar)**

```
機能
  MPC-CUnetステーションへ点データを要求
引数
  req_sa:要求するSA
  ar_top:要求するP()先頭
  rcv_ar:格納する配列
戻値
  -1:MSCレジスタSENDが下がらない。
  -2:MRC0レジスタRDYが立たない。
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
使用例
  Dim ar(60) As Integer
  res = cunet_req_pnt(4, 2000, ar(0))
注意 配列には XYZU の順番で格納されます
  MPC側: P(2000) X=10000 Y=10001 U=10002 Z=10003
                  ↓       ↓       ↓       ↓
  PC側:          ar(0)   ar(1)   ar(3)   ar(2)
```

- **cunet_req_mbk(req_sa, ar_top, rcv_ar)**

```
機能
  MPC-CUnetステーションへMBKデータを要求
引数
  req_sa:要求するSA
  ar_top:要求するMBK()先頭
  rcv_ar:格納する配列
戻値
  -1:MSCレジスタSENDが下がらない。
  -2:MRC0レジスタRDYが立たない。
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
使用例
  Dim ar(120) As Integer
  res = cunet_req_mbk(4, 6000, ar(0))
```

- **cunet_req_mbk_lng(req_sa, ar_top, rcv_ar)**

```
機能
  MPC-CUnetステーションへMBKデータをロング(4byte長)で要求
引数
  req_sa:要求するSA
  ar_top:要求するMBK()先頭
  rcv_ar:格納する配列
戻値
  -1:MSCレジスタSENDが下がらない。
  -2:MRC0レジスタRDYが立たない。
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
備考
  タッチパネルのデータ長をロングにした場合に使用
使用例
  Dim ar(60) As Integer
  res = cunet_req_mbk_lng(4, 6000, ar(0))    ← 対応するMPCのコマンドは S_MBK data 6000~Lng など
```

- **cunet_req_io(req_sa, ar_num, rcv_data)**

```
機能
```

```
  MPC-CUnetステーションへI/Oデータを要求
引数
  req_sa  :要求するSA
  ar_num  :要求するI/Oバンク番号
  rcv_data:格納する変数
戻値
  -1:MSCレジスタSENDが下がらない。
  -2:MRC0レジスタRDYが立たない。
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
備考
  ar_numの1Byteを読み込みます。負の数値を入れるとメモリーI/Oを読み込みます。
使用例
  res = cunet_req_io(4, -1, dt)
```

- **cunet_req_io_blk(req_sa, ar_top, rcv_ar)**

```
機能
  MPC-CUnetステーションへI/Oデータを要求
引数
  req_sa:要求するSA
  ar_top:要求するI/O先頭バンク番号
  rcv_ar:格納する配列
戻値
  -1:MSCレジスタSENDが下がらない。
  -2:MRC0レジスタRDYが立たない。
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
備考
  ar_topから240Byte連続で読み込みます。負の数値を入れるとメモリーI/Oを読み込みます。
使用例
  Dim ar(0 To 239) As Integer
  res = cunet_req_io_blk(4, -1, ar(0))
```

- **cunet_req_pnt_str(req_sa, ar_top, rcv_str, str_len)**

```
機能
  MPC-CUnetステーションへPNT文字列を要求
引数
  req_sa :要求するSA
  ar_top :要求するP()先頭
  rcv_str:格納する文字列変数(max239キャラクタ)
  str_len:返された文字数
戻値
  -1:MSCレジスタSENDが下がらない
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
使用例
  Dim s As String = Space(240)
  Dim res, l As Integer
  res = cunet_req_pnt_str(4, 1000, s, l)
  If res <> 0 Then
      MsgBox("error res=" + CStr(res))
  Else
      s = Mid(s, 1, l)
      TextBox_res.Text = s + "<eol>" + "len=" + CStr(l)
  End If
MPCプログラム例
  TM$="now "+HEX$(TIME(0))
  SETP 1000 TM$                  /* P(1000)～  文字列書き込み
```

- **cunet_req_mbk_str(req_sa, ar_top, rcv_str, str_len)**

```
機能
  MPC-CUnetステーションへMBK文字列を要求
引数
  req_sa :要求するSA
  ar_top :要求するMBK()先頭
  rcv_str:格納する文字列変数(max239キャラクタ)
  str_len:返された文字数
戻値
  -1:MSCレジスタSENDが下がらない
  0以上:MESR(Mail Error Status Register)の内容(0なら正常)
使用例
  Dim s As String = Space(240)
  Dim res, l As Integer
  res = cunet_req_mbk_str(4, 1000, s, l)
  If res <> 0 Then
      MsgBox("error res=" + CStr(res))
```

```
  Else
      s = Mid(s, 1, l)
      TextBox_res.Text = s + "<eol>" + "len=" + CStr(l)
  End If
```

MPC プログラム例

```
  TM$="now "+HEX$(TIME(0))
  S_MBK TM$ 1000 12              /* MBK(1000)～ 文字列書き込み,12 文字
```

■ MKY40 レジスタ Read/Write

- **cunet_peek(adr)**

機能
  MKY40 レジスタ読み込み
引数
  adr:MKY40 レジスタアドレス
戻値
  adr で指定されたレジスタの値(1byte)

- **cunet_peek_Lng(adr)**

機能
  MKY40 レジスタ読み込み
引数
  adr:MKY40 アドレス
戻値
  adr で指定されたレジスタの値(4byte)

- **cunet_poke(dat, adr)**

機能
  MKY40 レジスタ書き込み
引数
  dat:レジスタへ書き込むデータ(1byte)
  adr:MKY40 レジスタアドレス
戻値
  無し

- **cunet_poke_Lng(dat, adr)**

機能
  MKY40 レジスタ書き込み
引数
  dat:レジスタへ書き込むデータ(4byte)
  adr:MKY40 アドレス
戻値
  無し

- **cunet_poke_msb(ar)**

機能
  MKY40 Mail Send Buffer に 256 バイト書込み
引数
  ar:書込む配列
戻値
  無し

- **cunet_peek_mrb0(ar)**

機能
  MKY40 Mail Receive Buffer0 から 256 バイト読み込み
引数
  ar:格納配列
戻値
  無し

- **cunet_peek_mrb1(ar)**

機能
  MKY40 Mail Receive Buffer1 から 256 バイト読み込み
引数
  ar:格納配列
戻値
  無し

- **cunet_chk_mfr(sa)**

機能
  MFR(Member Flag Register)の SA ビットチェック
引数
  sa:SA 番号 0～63
戻値
  1:SA 有り(リンク成立)
  0:SA 無し(リンク不成立)
備考
  MFR には SA0～SA63 まで個々のメンバの状態が格納されています。
使用例
```
  If cunet_chk_mfr(4) = 0 Then      'SA4 の有無を確認
      MsgBox("相手 SA がない")
  End If
```

- **cunet_chk_run()**

機能
  SCR(System Control Register)の RUN phase ビットチェック
引数
  無し
戻値
  1:RUN フェーズである
  0:RUN フェーズでない
備考
  ボード上の LED4(緑)と連動します。
使用例
```
  If cunet_chk_run = 0 Then
      MsgBox("RUN フェーズではない")
  End If
```

- **cunet_chk_mcare()**

機能
  CCTR(Care CounTer Register)の MCC0～7(Member Care Counter)をリード&セット
引数
  無し
戻値
  MCC 値(最高 0xff)
備考
  MCARE 発生時はボード上の LED1(赤)が約 50ms 点灯します。
使用例
```
  c = cunet_chk_mcare           ' 読み込み後カウンタを 0 にする
  If c <> 0 Then
      MsgBox("MCARE 発生" + CStr(c))
  End If
```

- **cunet_chk_lcare()**

機能
  CCTR(Care CounTer Register)の LCC0～7(Link Care Counter)をリード&セット
引数
  無し
戻値
  LCC 値(最高 0xff)
備考
  LCARE 発生時はボード上の LED2(赤)が約 50ms 点灯します。
使用例
```
  c = cunet_chk_lcare           ' 読み込み後カウンタを 0 にする
  If c <> 0 Then
      MsgBox("LCARE 発生" + CStr(c))
  End If
```

Visual Basic 参照宣言例

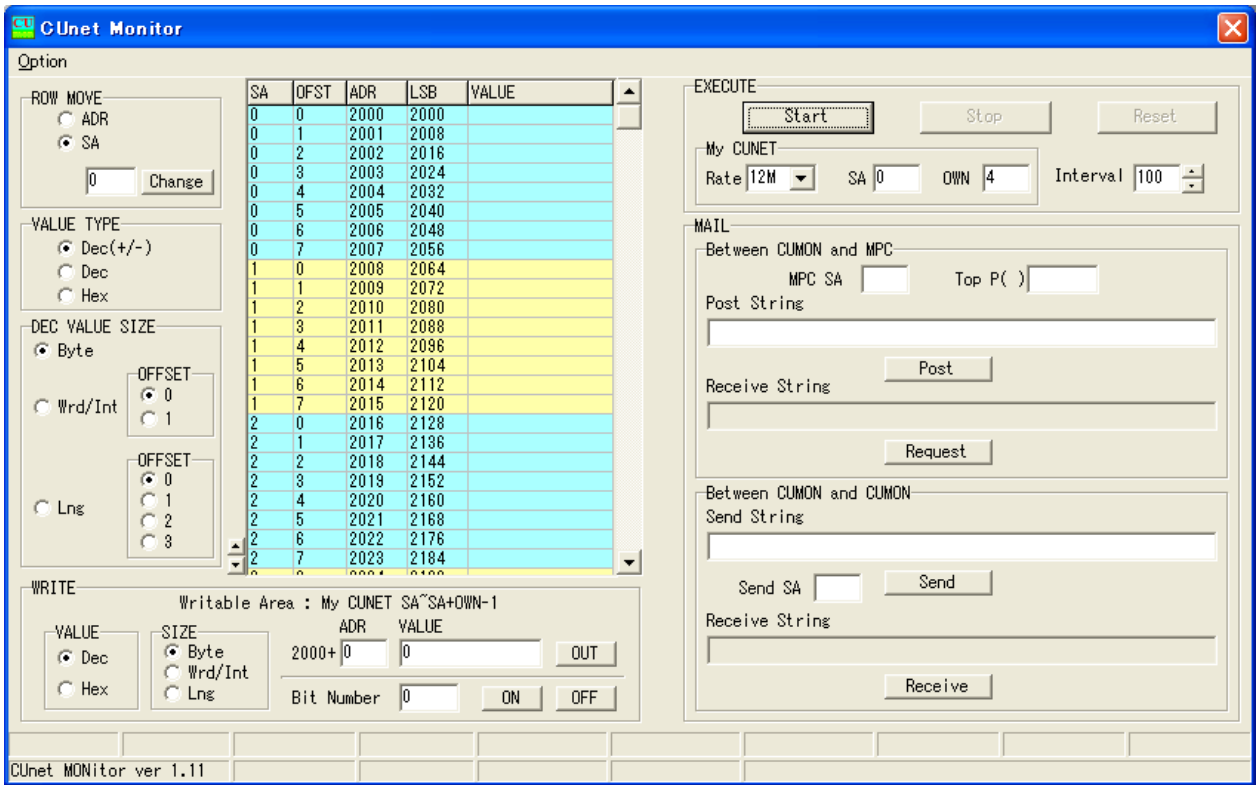
コメント付きの宣言モジュールをホームページからダウンロードできます。
セマフォ付き関数宣言とセマフォ無し関数宣言がありますが、前者の使用を推奨します。

MPC-2000 サイト ＞ ボタン[DOWNLOAD] ＞ グループ[サンプルプログラム] ＞ モジュール[Module_usbcunet]
http://departonline.jp/mpc2000/ref/dlf.php?files=download/Module_usbcunet.zip

CUnet Monitor (CUMON.EXE)

CUMON は USB-CUnet のグローバルメモリ、メール送受信、動作状況をモニタするツールで、MPC-CUnet とのデータ授受、I/O チェック、MPC-CUnet 間の状況監視等が行えます。

■ 画面



グループ	説明
EXECUTE	モニタの開始と停止
	・開始手順：「My CUNET」と「Interval」設定→「START」でモニタ実行。 ・SA：CUnet Monitor で占有する SA(ステーションアドレス)の先頭番地(0～63)。 ・OWN：占有するブロック数。 ユーザーが作成するアプリケーションの代替確認をする場合はそのアプリケーションと同じ SA、OWN を設定。グローバルメモリをモニタするだけ(例えば MPC 同士の通信監視)なら他のステーションと重複しない値を設定します。(EN は 63 固定) ・Interval：モニタサイクル(msec)。 ・Reset：USB-CUnet リセット(実行中の場合は停止)。SA、OWN を変更する場合は「Reset 」→「Start」。
ROW MOVE	モニタグリッドの行を移動
	・操作手順：「ADR」(バイト単位)、「SA」(ブロック単位)を選択→番地入力→「Change」
VALUE YPE	表示する値の型を指定
	・Dec(+/-)：符号付 10 進数 ・Dec：符号無 10 進数 ・Hex：16 進数
DEC VALUE SIZE	10 進数表示時の値のサイズを指定
	・Byte：バイト ・Wrd/Int：2 バイト ・Lng：4 バイト ・OFFSET：Wrd/Int･Lng 表示時のバイトオフセット
WRITE	占有ブロックへの書込み
	・数値書き込み 「VALUE」(10 進、16 進)選択→「SIZE」(バイト、2 バイト、4 バイト)選択→「ADR」(バイト単位)入力→「VALUE」(書き込む値)入力→「OUT」 ・ビット操作 「Bit Number」入力→「ON」または「OFF」

MAIL	MKY40 のメール機能によるデータ送受信(CUMON では文字列の送受信)
	CUMON<->MPC ・対 MPC-CUnet 間メール。MPC 側は事前に CUNET 初期化、CU_POST 実行。 ・「MPC SA」(MPC 側 SA 先頭番号)と「Top P()」(メールで使用する点データの先頭番号)入力 ・送信：「Post String」に送信する文字列(240 文字以下)入力→「Post」で上記設定 MPC の点データエリアに送信。 ・受信：「Request」で上記設定 MPC の点データエリアからリクエスト&受信
	CUMON<->CUMON ・対 CUMON 間メール(別の PC との通信)。 ・送信：「Send String」に送信する文字列(256 文字以下)入力→「Send SA」(受信側 SA 先頭番号)入力→「Send」。 ・受信：「Receive」で受信バッファからメール読み込み。
モニタグリッド	グローバルメモリのモニタ
	・SA 列：ステーションアドレスの番号(8 バイトで 1 ブロック) ・OFST 列：ブロック内のバイト位置 ・ADR 列：バイト単位のアドレスの番号(MPC の OUT、IN コマンドに対応) ・LSB 列：各バイトの最下位のビット番号(MPC の ON、OFF、SW コマンドに対応)
メインメニュー	・Option MKY40 R/W Check MKY40 のグローバルメモリをチェックします。正常終了しない場合は USB-CUnet のハード故障、USB 通信障害などが考えられます。

■ ステータスバー

下部のステータスバーに MKY40 の主要レジスタの内容、ファームウェアや DLL のバージョン等を表示します。

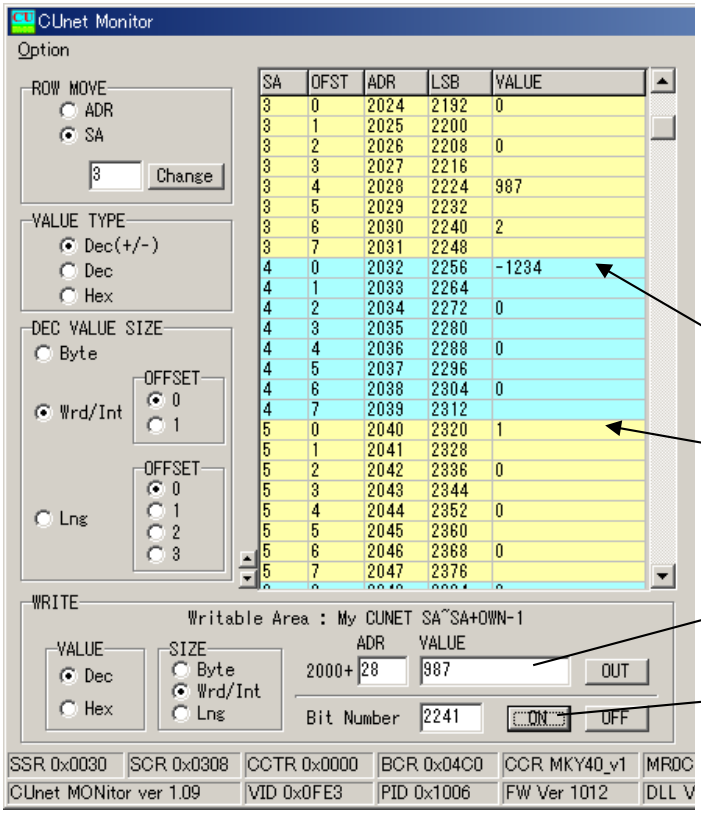
例えば、CUnet 間に通信障害が生じてメンバーが変化すると LED1(MCARE),LED2(LCARE)が点灯し、その発生回数が CCTR に加算されます(maxFF 回)。実行中に増えていく場合は各ステーションの SA、OWN、ケーブル、終端抵抗設定等を確認してください。(※CCTR は他のステーションがリセットされたり、途中参加した場合でも加算されますが、それは異常ではありません)

表示	概要
SSR (System Status Register)	ネットワークの稼働における各種ステータスが格納されるレジスタです。
SCR (System Control Register)	CUnet のネットワークをコントロールするレジスタです。
CCTR (Care CounTer Register)	LCARE 信号および MCARE 信号の発生回数が格納されるレジスタです。 上位バイト MCARE(LED1)、下位バイト LCARE(LED2)
BCR (Basic Control Register)	CUnet を構築する MKY40 の基本設定が格納されるレジスタです。
CCR (Chip Code Register)	MKY40 チップコード。USB-CUnet の場合“MKY40_v1” で正常です。それ以外の場合は USB-CUnet のハード障害が考えられます。
MR0CR (Mail Receive 0 Control Register)	MRB0(Mail Receive Buffer 0)に対応するメール受信をコントロールするレジスタです。
MR1CR (Mail Receive 1 Control Register)	MRB1(Mail Receive Buffer 1)に対応するメール受信をコントロールするレジスタです。
MSCR (Mail Send Control Register)	MSB(Mail Send Buffer)へライトしたデータセットのメール送信をコントロールするレジスタです。
MESR (Mail Error Status Register)	メール送信の開始以後、メール送信エラーが発生した場合のエラー内容を示すレジスタです。
MSRR (Mail Send Result Register)	メール送信に要した所要時間が格納されるレジスタです。
MFR (Member Flag Register)	個々のメンバ状態が格納されています。 下位ビットより SA0～63 に対応しています。

※レジスタの詳細内容は ㈱ステップテクニカ「CUnet 専用 IC　MKY40 ユーザーズマニュアル」を参照。

CUnet MONitor Ver	本アプリケーションのバージョン
VID (Vender ID)	USB ベンダー ID(FE3)です。
PID (Product ID)	USB プロダクト ID(1006)です。
FW Ver (Firmware Version)	USB-CUnet ファームウェア(Cyload1.spt)のバージョンです。
DLL Ver (DLL Version)	usbcunet.dll のバージョンです。

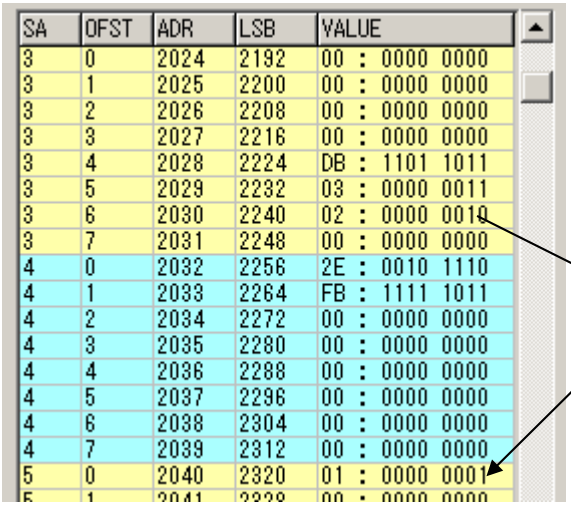
■ 実行例 1 [グローバルメモリのモニタ]



◆MPC の操作

```
#CUNET 4 4 8
#OUT -1234  2032~Wrd
#ON 2320
#
#PR IN(2028~Wrd)
987
#PR SW(2241)
1
```

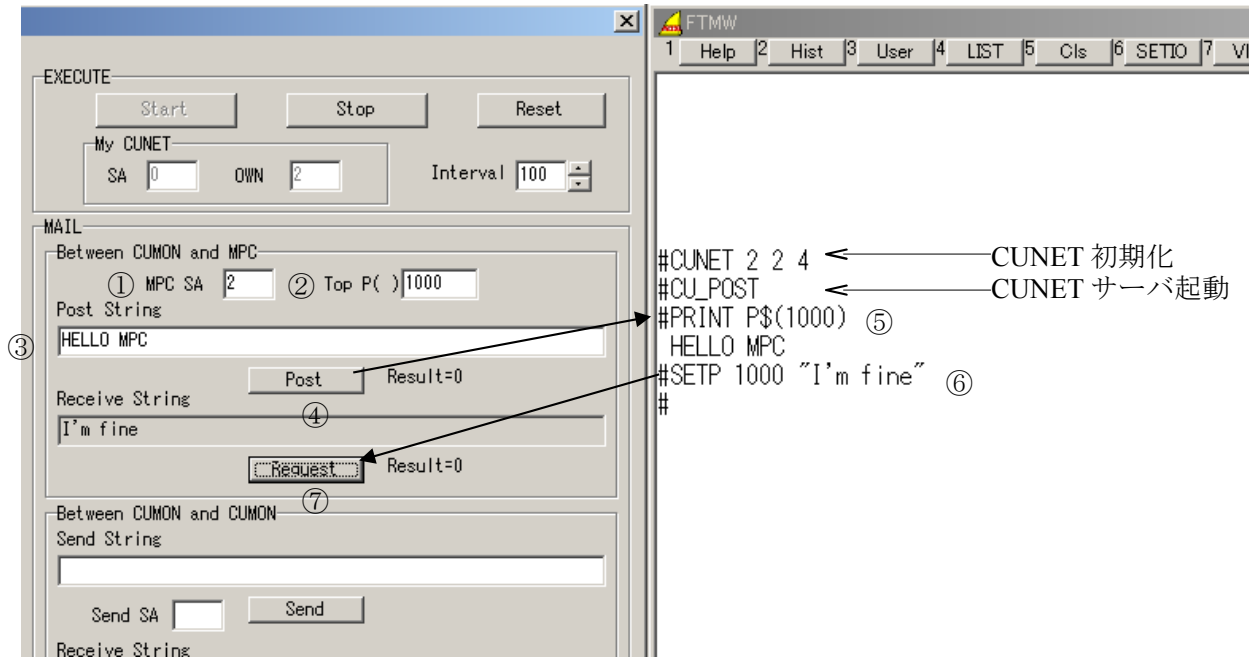
- ON/OFF 操作の時は Hex 表示にすると分かりやすくなります。



SA	OFST	ADR	LSB	VALUE
3	0	2024	2192	00 : 0000 0000
3	1	2025	2200	00 : 0000 0000
3	2	2026	2208	00 : 0000 0000
3	3	2027	2216	00 : 0000 0000
3	4	2028	2224	DB : 1101 1011
3	5	2029	2232	03 : 0000 0011
3	6	2030	2240	02 : 0000 0010
3	7	2031	2248	00 : 0000 0000
4	0	2032	2256	2E : 0010 1110
4	1	2033	2264	FB : 1111 1011
4	2	2034	2272	00 : 0000 0000
4	3	2035	2280	00 : 0000 0000
4	4	2036	2288	00 : 0000 0000
4	5	2037	2296	00 : 0000 0000
4	6	2038	2304	00 : 0000 0000
4	7	2039	2312	00 : 0000 0000
5	0	2040	2320	01 : 0000 0001

```
#CUNET 4 4 8
#OUT -1234  2032~Wrd
#ON 2320
#
#PR IN(2028~Wrd)
987
#PR SW(2241)
1
```

■ **実行例 2 [対 MPC 間メール]**



①メールを送受信する MPC の SA を入力

② MPC 点データのメールで使用するエリアの先頭番号を入力

③送信する文字列を入力

④「Post」を押す → MPC の P(1000)以降に文字列が入る

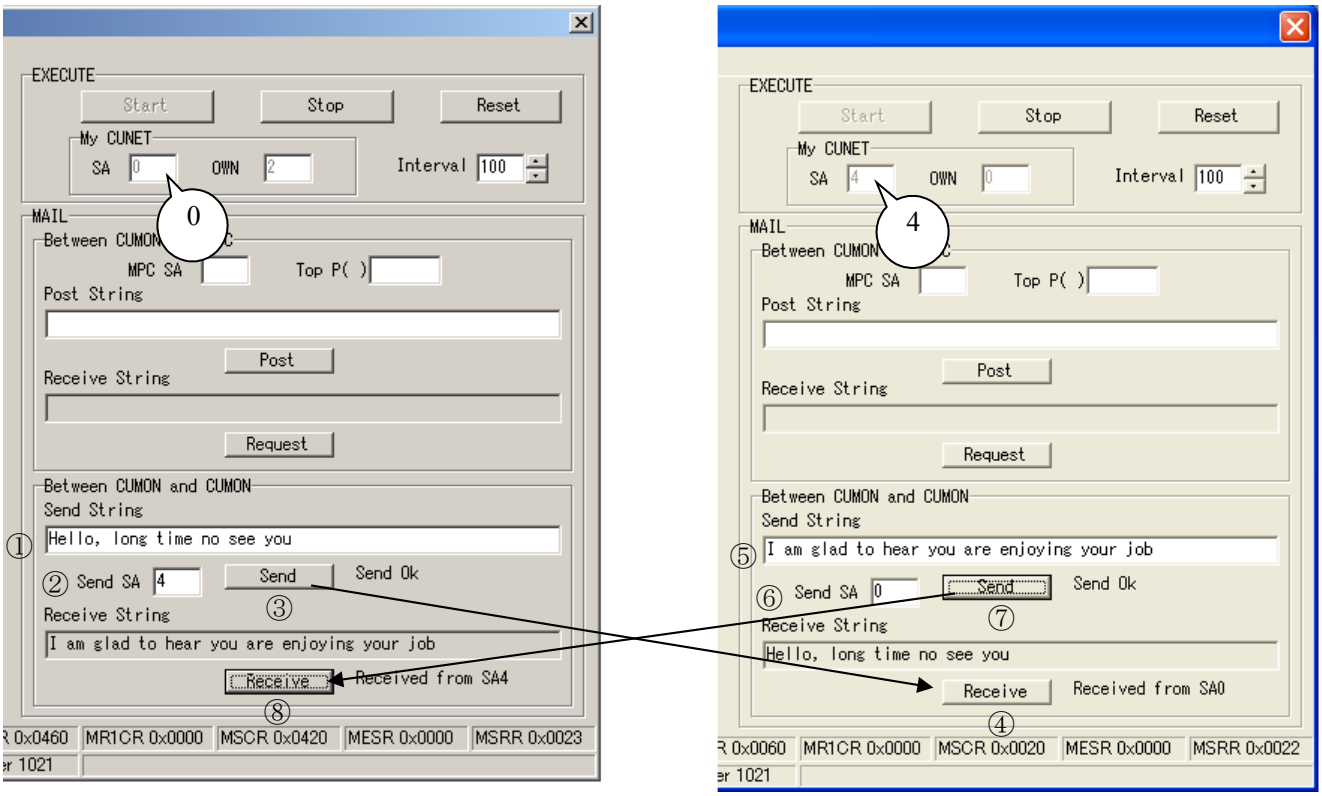
⑤ MPC 側で確認表示

⑥ MPC 側で P(1000)以降にメール文字列作成

⑦ MPC にメールをリクエスト&受信

※送受信結果(Result)は Mail Error Status Register の内容です。正常なら Result=0 になります。

■ 実行例 3 [対 CUMON 間メール]



Windows 2000　　Windows XP

①(2000)送信する文字列を入力
②(2000)送信先の SA を入力
③(2000)「Send」を押す → Send Buffer に書込み → 指定 SA へ送信
④(XP)受信側「Receive」を押す → Receive バッファから読み込む
⑤(XP)送信する文字列を入力
⑥(XP)送信先の SA を入力
⑦(XP)「Send」を押す → Send Buffer に書込み → 指定 SA へ送信
⑧(2000)受信側「Receive」を押す → Receive バッファから読み込む

関係資料

■ サンプルプログラム
MPC-2000 情報 の[アプリケーションノート] や DOWNLOAD ＞ [サンプルプログラム] に使用例がありますのでご参照ください。

■ 出典、引用
「CUnet」は、株式会社ステップテクニカの登録商標です。

株式会社ステップテクニカ　http://www.steptechnica.com/
「CUnet ユーザーズマニュアル」 第 2 版
「Let's Try! CUnet 入門者編・フィールドエンジニア編」
「CUnet 導入ガイド　（CUnet プロトコル基本解説）」
「CUnet テクニカルガイド（ネットワーク用）」

これらの資料から、引用または原文のまま掲載しています。

諸注意

■ グローバルメモリについて

アプリケーションを終了してもグローバルメモリは保持されます。グローバルメモリで他局とインターロックをしている場合、各機器の再起動時のタイミングによってはインターロックが不整合になることもあります。必要に応じて FormClosing 等で cunet_init(0, 4, 7) などと実行してグローバルメモリをクリアしてください。

■ usbcunet.dll の MPC 間メール転送について

MPC 間メール転送関数の開始時に送信先の MPC-CUnet が受信不可状態(NORDY)であると、転送作業を実行することができません。その場合、これらの関数は直ちに戻ってきます。post 関数ならデータは MPC に伝わりません。req 関数なら受信データに 0 が格納されて戻ります。

MPC のタスク数が多い、時間浪費タスク(MON コマンドで!マークが付くタスク)ある場合 NORDY 状態が発生します。このような状況下では正常なデータ授受を行えません。レスポンスを確認し、NORDY ならば MPC のプログラムを見直して下さい。

戻値は MKY40 のレジスタの内容を表します。
0:正常です。
-1:MSC レジスタ SEND が下がらない。 ＝送信が完了しない。
-2:MRC0 レジスタ RDY が立たない。 ＝メール着信の許可ができない (post 関数に-2 はありません)
1 以上:MESR(Mail Error Status Register)の内容。

MKY40 Mail Error Status Register		
bit	status	意味
5	STOP	通信フェーズが RUN フェーズ以外のため通信できなかった。
4	LMFLT	メールセンドリミットレジスタに設定された値が不正。
3	SZFLT	メールの送信サイズは不正
2	TOUT	メールセンドリミットレジスタに設定された制限時間内に送信が完了しなかった
1	NOEX	送信先のステーションアドレスが存在しない
0	NORDY	送信先のステーションが受信待機中でない

- リトライの例

```
For i = 0 To 17      ' pnt データ連続書き込み
    Do
        res = cunet_post_pnt(4, 1000 + i * 15, ar(i * 15 * 4))
        TextBox_res.SelectedText = CInt(i) & " Mail Error Status register " & Hex$(res) & vbCrLf
        If res = 0 Then Exit Do
    Loop
Next i
-----
For i = 0 To 15      ' MBK データ連続読み込み
    Do
        res = cunet_req_mbk_lng(4, i * 120 + 1000, ar(i * 60))
        If res = 0 Then Exit Do
    Loop
Next i
```

■ MPC の CU_POST の実行タスク

MPC のプログラムで CU_POST のパラメータを省略すると CUnet メールサーバは空タスクで実行されますが、次のような順番で宣言すると、LOAD 後の初回実行時にタスク 31 が重複します。

```
CU_POST          /* パラメータを省略するとタスク 31 から空タスクを探して割り当てられる
MEWNET  38400 1  /* ch1 はタスク 31 で実行。MEWNET は ch によりタスクが固定される。
```

FORK されたタスクとかぶることもあります。かぶるとメール通信に障害が発生します(NORDY)。
明確なタスク管理のために CU_POST は明示的にタスクを指定することを推奨します。

```
CU_POST  25      /* タスク 25
```

■ VB.net タイマーコンポーネント

VB6 と VB.net のタイマーイベント発生には違いがあります(VB6 では Timer コントロール、VB.net では

Timer コンポーネントと言っています)。
　VB6 は一回の処理が終らないと次のタイマーイベントは発生しませんでしたが、VB.net は処理中でも次のイベントが発生して不具合の原因となります。Timer を使って繰り返し処理を行う場合は注意してください。
　確実に Timer1.Interval ＞ Timer1_Tick 所要時間 ならば問題はありませんが、そうでない場合は次のように Timer1.Enabled = False / Timer1.Enabled = True やフラグでイベントの交通整理をして下さい。

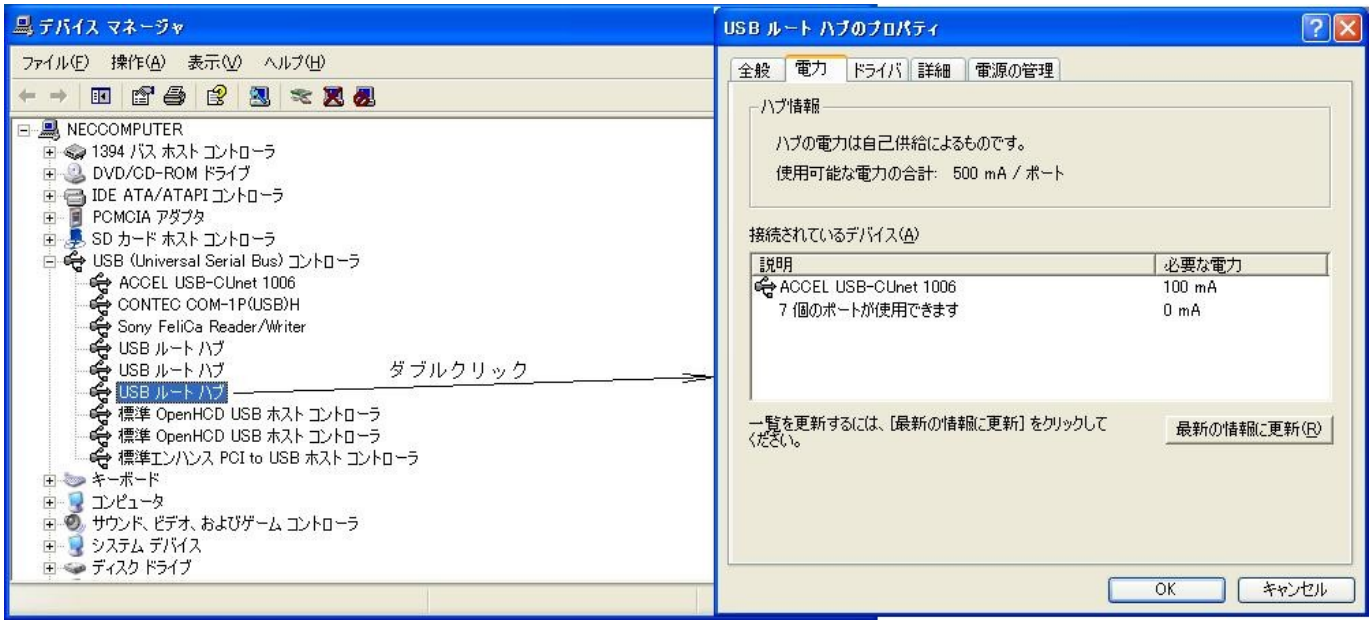
```
Private Sub Timer1_Tick(ByVal sender As System.Object, …) Handles Timer1.Tick
    Timer1.Enabled = False     ' または Timer1.Stop()
    (イベント処理)
    Timer1.Enabled = True      ' または Timer1.Start()
End Sub
```

　詳しくは 弊社 web サイト
アプリケーションノート ＞ [an2k-013] VB2008 Timer コンポーネントの実験(タイマー使用時の注意と対策)
をご覧下さい。

■ パソコン USB ハブ消費電流確認 (Windows XP)

　パソコンの同一 USB ハブに多数の USB 機器を接続し消費電流が多くなると USB-CUnet のアクセスが不安定になるようです。
　グローバルメモリのアクセスに時間がかかるといった症状がでるようです。
　1 ポート当たりの消費電流が定格未満でもハブの総消費電流に注意してください。
　マザーボード直結の USB ポートでも不具合が出る PC もあるようです。そういう場合は増設ボードをご検討ください。

↓ Windows XP



グローバルメモリアドレス表

SA	IN/OUT バンク番号	ON/OFF/SW ビット番号	
0	2000～2007	2000～2063	
1	2008～2015	2064～2127	
2	2016～2023	2128～2191	
3	2024～2031	2192～2255	
4	2032～2039	2256～2319	
5	2040～2047	2320～2383	
6	2048～2055	2384～2447	
7	2056～2063	2448～2511	
8	2064～2071	2512～2575	
9	2072～2079	2576～2639	
10	2080～2087	2640～2703	
11	2088～2095	2704～2767	
12	2096～2103	2768～2831	
13	2104～2111	2832～2895	
14	2112～2119	2896～2959	
15	2120～2127	2960～3023	
16	2128～2135	3024～3087	
17	2136～2143	3088～3151	
18	2144～2151	3152～3215	
19	2152～2159	3216～3279	
20	2160～2167	3280～3343	
21	2168～2175	3344～3407	
22	2176～2183	3408～3471	
23	2184～2191	3472～3535	
24	2192～2199	3536～3599	
25	2200～2207	3600～3663	
26	2208～2215	3664～3727	
27	2216～2223	3728～3791	
28	2224～2231	3792～3855	
29	2232～2239	3856～3919	
30	2240～2247	3920～3983	
31	2248～2255	3984～4047	
32	2256～2263	4048～4111	
33	2264～2271	4112～4175	
34	2272～2279	4176～4239	
35	2280～2287	4240～4303	
36	2288～2295	4304～4367	
37	2296～2303	4368～4431	
38	2304～2311	4432～4495	
39	2312～2319	4496～4559	
40	2320～2327	4560～4623	
41	2328～2335	4624～4687	
42	2336～2343	4688～4751	
43	2344～2351	4752～4815	
44	2352～2359	4816～4879	
45	2360～2367	4880～4943	
46	2368～2375	4944～5007	
47	2376～2383	5008～5071	
48	2384～2391	5072～5135	
49	2392～2399	5136～5199	
50	2400～2407	5200～5263	
51	2408～2415	5264～5327	
52	2416～2423	5328～5391	
53	2424～2431	5392～5455	
54	2432～2439	5456～5519	
55	2440～2447	5520～5583	
56	2448～2455	5584～5647	
57	2456～2463	5648～5711	
58	2464～2471	5712～5775	
59	2472～2479	5776～5839	
60	2480～2487	5840～5903	
61	2488～2495	5904～5967	
62	2496～2503	5968～6031	
63	2504～2511	6032～6095	

----- EOF ----