MPC-CUnet のメールデータの扱いについて

平成 20 年 10 月 24 日

CUnet には、全体の同期サイクルに影響を与えない、メール転送機能があります。
転送レートは、(転送ブロック数　+ 3) ×　サイクルタイムです。
256byte 転送する場合は、64 プロック(8byte 1 ブロック)となり、67 ×　約 2.5msec(64 ステーション時)=約 0.17 秒です。RS-232 比較で 15kpps 程度と高速ではありませんが、データをすべてバイナリで転送することができデータの整合性の確認も必要でないため､効率のよいデータ交換が可能です。
このメール転送を MPC 上では以下の二つのコマンドと六個のグローバル変数でサポートします。これらを使用することにより、CUnet に接続された MPC-2000 間で、点データ、MBK データの情報交換を行うことができます。(USB-CUnet 含む)
また文字列の情報交換については、点データ上に文字列を織り込む機能がありますので、点データとして文字列の交換を行います。

SETP n A$ ← P(n)に文字列を収納
pr P$(n) ← P(n)を文字列として読み取る

【コマンド】

CU_POST

CU_POST はメールサーバーを起動します。CU_POST が実行されると MPC は、メールの着信を監視し、データを受け取り次第指定箇所に収納します。また、相手からのデータ転送請求に対して、自動的にデータを転送するようになります。
起動には、以下の三種類あります。(機能には相違はありません)

1) 引数なし　CU_POST が実行された時点で 31 から降順にタスクを検索し空いているタスクでメールサーバーを起動。
2) 引数 1～31 指定タスクでメールサーバーを起動。
3) 引数に VOID を与えるか、VOID|n を与えると、サーバの動作状態が表示されます。

POST　dest　MBK()|P()

POST はメール転送コマンドです。POST は指定アドレスに対して、データを転送します。

例
POST 2 P(100)
SA=2 のステーションに対して、P(100)～P(114)のデータが転送されます。
(15 個の点データは、15*16 = 240 bytes 残り 16byte はシステムで使用。)
POST 3 MBK(20)
SA=3 のステーションに対して、MBK(20)～P(139)のデータが転送されます。
(120 個の MBK データは、120*2 = 240 bytes 残り 16byte はシステムで使用。)

dest に負の値をあたえると、相手への送信請求になります。この場合は、自己側も CU_POST を実行してメールサーバーを起動しておきます。なお､dest=0 に対する要求は 64 を設定します。

【グローバル変数】

CUM_TASK　CU_POST によって使用されたタスク番号
CUM_SRC　受信メールの相手アドレス
CUM_PNT　受信メールの種類　1:　P(n) 2: MBK(n)
CUM_NUM　受信メールの P(n)もしくは MBK(n)の n の値
CUM_CNT　受信メールごとにインクリメント
CUM_ERR　エラー各 bit は以下のとおり。

BIT7	MAIL SEND RROR
BIT5	通信停止
BIT4	送信タイムアウト不正(通常 0)
BIT3	送信ブロック不正(通常 0)
BIT2	送信タイムアウト発生
BIT1	送信相手不在
BIT0	送信相手が受信待機となっていない

以下は、MKY40 を直接アクセスする場合のメモリイメージです。

MSB/MRB0 のイメージ

転送パケット(256byte)　点データの場合　NUM = n , ARY = 1 CMD = 2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	**NUM**		**ARY**	**CMD**	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv
10	X(n)				Y(n)				Z(n)				U(n)			
20	X(n+1)				Y(n+1)				Z(n+1)				U(n+1)			
30																
↓																
E0																
F0	X(n+14)				Y(n+14)				Z(n+14)				U(n+14)			

転送パケット(256byte)　MBK データの場合　NUM = n , ARY = 2 CMD = 2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	**NUM**		**ARY**	**CMD**	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv
10	mbk(n)		mbk(n+1)		mbk(n+2)		mbk(n+3)		mbk(n+4)		mbk(n+5)		mbk(n+6)		mbk(n+7)	
20																
30																
↓																
E0																
F0	mbk (n+112)		mbk (n+113)		mbk (n+114)		mbk (n+115)		mbk (n+116)		mbk (n+117)		mbk (n+118)		mbk (n+119)	

MSB のイメージ

転送パケット(8byte)　MBK データの場合　NUM = n , ARY =X CMD = 1(送信請求)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	**NUM**		**ARY**	**CMD**	-	-	-	-	-	-	-	-	-	-	-	-

USB-Cunet でのメール転送あるいは受信の方法

■　転送方法

mrb0,msb,mes,msl,msch,mscl は、MKY のマニュアルに掲載されているレジスタとバッファです。

データの準備

【*msb,msb+1 ワード型で配列の番号をいれる。;

*(msb + 2)　　配列の種類を決定　1:点データ　2:MBK 配列

*(msb + 3) = 2 ; // ライトコマンド

(msb+16)～(msb+255)にデータをいれる。】(バイナリ)]転送パケットの表参照

レジスタ操作

*mes = 0 ;

*msl = 0xFFF ;

*mscl = 0x20 ; // 32 block

*msch = 0x40 |dest ; //　dest は送り先のステーション番号。

完了待ち

while((*msch & 0x40) != 0) ; 転送終了のポーリング

■　送信請求

【*msb,msb+1 ワード型で配列の番号をいれる。

*(msb + 2)　配列の種類を決定　1:点データ　2:MBK 配列

*(msb + 3) = 1 ; // リクエスト】送信請求パケットの表参照

*mes = 0 ;

```
*msl = 0xfff ;
*mscl = 0x01 ; // 1 block
*msch = 0x40 |dest ;   // dest は送り先のステーション番号。(請求相手)
while( (*msch & 0x40) != 0) ; 転送終了のポーリング;
```

■　**受信の方法**

```
*mrc0 = 0x40 ;  // 受信イネーブル
while( (*mrc0 & 0x80) == 0 ){ // rcv フラグの監視
if((*mrc0 & 0xc0) == 0){ *mrc0 = 0x40 ; delay(10) ; } ; // エラー状態か
らのリカバリなど
if((*mes & 0x3f) != 0) { *mes = 0 ; delay(10) ; } ;
swap() ; } ;// wait rcv data]
```

以後は、バッファの内容を見る

【　　*(mrb0 + 2) 配列の種類　1:点データ　2:MBK 配列

　*(mrb0 + 3)　 2 WRITE コマンドの確認

　*msb,msb+1 ワード型の配列の番号

　　(msb+16)～(msb+255)がデータ(バイナリ) 】