統計数理 第33巻 第2号（1985）

パーソナル・コンピュータ（NEC/PC-9801シリーズ）によるTSSフロントプロセッサTERM98TN

統計数理研究所　中　村　　隆

（1985年6月　受付）

1. はじめに

コンピュータは，“計算”機としてよりは，研究の過程で発生する様々な情報の処理機器として使われることが多くなってきている．とくに，パーソナル・コンピュータは，簡単な作表計算，日本文や英文のワードプロセッシング，小規模なデータベース管理などのために使われることの方が多いといってよいであろう．このような目的のためには，大型計算機よりもパーソナル・コンピュータの方が，小廻りがきき，使いやすいといえる．

共通OSのもとでハードディスクを装備すれば，比較的安価な個人用ワークステーションとして現段階でも満足のいく環境を整えることができる．もちろん，研究者にとって大量・高速の計算も必要であり，大型計算機とパーソナル・コンピュータを使いわける，というよりは，両者をうまく統合して使うことが研究環境のためにますます重要になってくる．そこで鍵になるのが，パーソナル・コンピュータを大型計算機の端末とする使い方である．大学の計算センタの広報などにパーソナル・コンピュータを端末機にする記事が多く見受けられるのも，研究者の間でこのような使い方への関心が強いことを示している（市川（1983）；柴山（1984）；赤繁（1985））．

パーソナル・コンピュータを大型計算機の端末とすることは比較的容易に行なえる．RS-232Cポートがあれば，BASICのTERM文を使用するか，あるいはBASICで簡単なプログラムを作成すればよい．ただし，上述のような統合環境を作るには，最低限ファイル転送の機能が欲しい．ファイル転送ができれば，大型計算機で実行した計算結果をパーソナル・コンピュータ側に取り込んでワードプロセッサで処理することや，パーソナル・コンピュータ側で作成したプログラムやデータを大型計算機に転送して計算を実行したりすることが可能となるからである．また，パーソナル・コンピュータの文字処理機能を生かして，画面編集などの使いやすい機能も実現したいところである．

これから紹介するTERM98TNは，HITAC-MシリーズVOS3のTSS端末としてパーソナル・コンピュータNEC/PC-9801シリーズ（以下，PC-98）を利用するために開発した画面編集・ファイル転送機能をもつフロントプロセッサである．ここで，フロントプロセッサと呼ぶのは，ホストへの送信は行単位のままであるがパーソナル・コンピュータ側の文字処理機能を使うことによって擬似的な画面編集端末として使えるようにしているからである．基本的機能は，仁木氏の富士通FM-7/8版の端末プログラム（仁木（1986））を踏襲するものであるが，PC-98上での実現方法はまったく別のものであり，また，いくつかの独自の機能をもっている．

TERM98TNを起動すれば，それだけで，パーソナル・コンピュータをごく普通の行編集端末として用いることができる．さらに，画面編集機能によって画面上にある文字列を適当に編集してから送信することができるので，画面編集端末として用いることが可能である．その他に，ファイル転送を含むローカルOS機能も有しており，ホストとの交信途中あるいは終了後も効率的にデータを処理することができる．

どのような機能をどのような方法で実現するかを考えるとき，操作性と汎用性とはしばしば対立する．操作性を優先して使いやすさを追求すれば使用する機種の能力を極力使うことになり，その機種に依存する部分が多くなる．一方，汎用性を優先すれば操作性が限定されてしまう．TERM98TNの開発では，個人が日常的に使う点を考慮し，どちらかといえば汎用性よりも操作性を重視することにした．また，開発言語にBASICを用いたが，これも開発の簡便性と公開情報の豊富さによる．したがって，パーソナル・コンピュータの他機種への移植や他のホストとの相性についてはあまり考えていない．HITACのVOS3/TSSとNECのPC-98およびBASICとの組合せを前提にしたプログラムであることを断っておく必要があろう．ただし，(ひとりの研究者が)どのような機能を欲しているか，また実現できるかの参考にはなると思う．

TERM98TNは，当初スタンドアロンのN88-BASIC(86)上で動いていたが，MS-DOS版のN88-BASIC(86)インタプリタ Ver 1.0(以下，MS-DOS/BASIC)上に移植・改良を行なった．この方が，MS-DOS上の豊富なアプリケーションを利用するにも有利だからである．

本稿では，MS-DOS/BASICによる新版TERM98TNを題材に，第2節では単純版，第3節ではプロトタイプ版，第4節では簡略版，と段階を追いながら端末プログラムの1つの作成方法について解説する．最後に，第5節では現段階での完成版プログラムのマニュアルを掲載する．

＊ MS-DOSはマイクロソフト社の登録商標である．

2. 簡単な端末プログラム ——TERM98TN0——

パーソナル・コンピュータを大型計算機の端末とすることは比較的容易である．簡単な端末プログラムの例を図1に示す．ここでは，基本的に，受信バッファをソフト的にスキャンする方式を採用している．機能的には全然足りないが，これだけでもホストとの交信はできる．TERM98TNの背骨ともいえるプログラムである．

プログラムは大きく3つの部分からなる．第1は，通信回線をオープンする部分([120])．第2は，受信バッファをスキャンする部分([140]～[170])．第3は，キーボードをスキャンする部分([190]～[220])である．要するに，第2，第3の部分を往き来するだけの仕組である．

```
100 '***** TERM98TN0: A Simple Version *****
110 '
120  OPEN "COM:E71XS" AS #1
130 '
140 *FR.HOST
150  IF LOC(1)=0 THEN*KEYBOARD
160  H$=INPUT$(LOC(1),#1)
170  PRINT H$;
180 '
190 *KEYBOARD
200  K$=INKEY$
210  IF K$<>"" THEN PRINT K$; : PRINT#1,K$;
220  GOTO*FR.HOST
```

図1. 端末プログラム（単純版）—TERM98TN0—
Figure 1. A Simple Terminal Program, TERM98TN0

(1) 通信回線のオープン

RS-232Cによるホストとの通信条件は，PC-98では基本的にはメモリスイッチによって管理している（開発機での設定は，SW1＝78h, SW2＝B5h, SW3＝CChである）．このうち，通信速度は1,200ボーである．MS-DOSでの設定にはSPEEDコマンドを用いればよい．その他の条件はBASICのOPEN文で変更することが可能であり，[120]行の“E71XS”の意味は順に，

偶数パリティ，データ長7ビット，ストップビット1，
データフロー制御有効，SI/SO有効

であることを示す．以上は，統計数理研究所での現行の通信条件である．

(2) 受信バッファのスキャン

受信バッファをスキャンする部分では，まず，バッファが空であるかを調べ，空であればキーボードのスキャンに移る（[150]）．空でなければバッファから文字列を取り出し（[160]），それを画面に出力して（[170]），キーボードのスキャンに移る．

(3) キーボードのスキャン

キーボードをスキャンする部分では，キー入力があるかを調べ（[200]），キー入力があればそれを画面に表示するとともにホストへ送信し（[210]），通信バッファのスキャンに移る（[220]）．

このプログラムで曲りなりにもホストとの通信はできるが，機能的には不十分である．第1に，ブレーク信号が送れない．ブレーク信号が送れないと，長いリストの表示などを途中でキャンセルできないので往生することになる．第2に，行内での編集機能さえもないといってよい（ただし，〈BS〉は機能する）．また，実行してみればわかるが，カーソルが表示されない（PC-98では，INKEY$関数を使用するとカーソルが消えてしまう）．

この程度なら，BASICのTERMコマンドを用いてPC-98を端末化した方がずっと使いやすい．しかし，ファイル転送機能や画面編集機能のことを考えると自前のプログラムを作る必要がある．また，MS-DOS/BASICがTERMコマンドをサポートしていないという事情もある．次節では，端末プログラムのプロトタイプを示す．

＊　リスト中の行番号を［nnn］で示す．
＊　制御文字およびキー種別を〈LF〉，〈CR〉，〈BS〉などのように〈　〉で括って示す．

3. 端末プログラムのプロトタイプ　——TERM98TN1——

前節で不足していた機能を追加するとともに，ファイル転送を含むローカルOS機能や画面編集機能を実現できるような構造にした端末プログラムのプロトタイプ版を図2に示す．TERM98TNの骨格ともいうべきものである．

このプログラムには大幅な変更があるが，基本的考え方は図1のプログラムと同じである．大きくは，通信回線のオープンを含む初期設定の部分（[510]および[770]～[910]），通信バッファとキーボードのスキャンを往復するメインループの部分（[515]～[620]），送信の下請や画面編集を行なうサブルーチンの部分（[625]～[730]），ブレーク信号の送信等を行なう割り込み処理ルーチンの部分（[735]～[765]）に分れる．

```
500 '***** TERM98TN1: A Prototype ************
505 '
510 CLEAR &H100 : GOSUB*INITIALIZE
515 '########################## main loop #####
520 *FR.HOST : COLOR 5
525  IF LOC(1)=0 THEN*TO.HOST
530 '
535  CALL CUROFF
540  WHILE LOC(1)>0 : DUM#=FRE(0)
545    H$=INPUT$(LOC(1),#1)
550    FOR I=1 TO LEN(H$)
555      A$=MID$(H$,I,1)
560      IF A$=LF$ THEN        : GOTO 575
565      IF A$=CR$ THEN PRINT : GOTO 575
570      PRINT A$;
575    NEXT I
580  WEND
585 '
590 *TO.HOST
595  IF KBUF>0 THEN GOSUB*SEND.LINES
600 '
605 *KEYBOARD : COLOR 7
610  K$=INKEY$ : CALL CURON
615  IF K$<>"" THEN GOSUB*SCRN.EDIT
620  GOTO*FR.HOST
625 '######################## subroutines #####
630 *SEND.LINES
635  B$=LBUF$(KBUF) : KBUF=0
640  IF HOST=1 THEN PRINT#1,B$;CR$;
645  IF HOST=2 THEN GOSUB*LOCAL
650  RETURN
655 '=========================================
660 *SCRN.EDIT
665  IF K$<" " THEN*ED.IT ELSE*E.ANK
670 '
675 *ED.IT
680  IF K$=CR$ THEN*E.CR
685  IF K$=BS$ THEN*E.ANK
690  RETURN
695 '
700 *E.ANK : PRINT K$; : L$=L$+K$ : RETURN
705 '
710 *E.CR : PRINT
715  KBUF=1 : LBUF$(KBUF)=L$ : L$=""
720  RETURN
725 '=========================================
730 *LOCAL : COLOR 6 : PRINT B$ : RETURN
735 '##################### Fkey interrupt #####
740 *INT.LOCAL : HOST=3-HOST : RETURN
745 '-----------------------------------------
750 *INT.BREAK
755  OUT &H32,&H3F : FOR IBRK=1 TO 1000 : NEXT
760  OUT &H32,&H37
765  RETURN
770 '##################### initialization #####
775 *INITIALIZE : CONSOLE ,,,1 : SCREEN 3
780  DEF SEG=SEGPTR(2) : GOSUB*INSTL.MCODE
785  BS$=CHR$(8): LF$=CHR$(10): CR$=CHR$(13)
790  OPEN "COM:E71XS" AS #1 : PRINT#1,CR$;
795  HOST=1
800  L$="" : KBUF=0 : DIM LBUF$(1)
805  GOSUB*FKEY.ALLOC : COLOR 4 : CLS
810  RETURN
815 '
820 *INSTL.MCODE
825  RESTORE*MCODE.DATA : I=0 : READ A$
830  WHILE NOT A$="#"
835    POKE I,VAL("&H"+A$) : I=I+1 : READ A$
840  WEND
845  CURON=0 : CUROFF=30
850  RETURN
855 *MCODE.DATA
860  DATA 36,8B,16,38,04,36,8B,0E,36,04
865  DATA 50,B8,A0,00,F6,E1,D1,E2,03,D0
870  DATA 58,B4,13,CD,18,B4,11,CD,18,CF
875  DATA B4,12,CD,18,CF,"#"
880 '
885 *FKEY.ALLOC
890  KEY 6,"LOCAL" : KEY 10,"break!"
895  ON KEY GOSUB ,,,,,*INT.LOCAL
900  ON KEY GOSUB ,,,,,,,,,*INT.BREAK
905  KEY (6) ON : KEY (10) ON
910  RETURN
```

図2. 端末プログラム（プロトタイプ版）—TERM98TN1—
Figure 2. Prototype of a Terminal Program, TERM98TN1

(1) メインループ

メインループの部分は，受信バッファのスキャン（＊ FR.HOST），ホストへの送信（＊ TO.HOST），キーボードのスキャン（＊ KEYBOARD）から成っている．

(2) 受信バッファのスキャン

受信バッファのスキャンでは，バッファが空であるかを調べ，空であれば送信部へ移る（[525]）．バッファが空でなければ，カーソル表示を消し（[535]），受信バッファが空になるまで文字列を取り出し画面に出力する（[540]～[580]）．受信内容は水色で表示している（COLOR 5, [520]）．ここでの工夫は，BASICが自動的に行なう文字列領域のガーベジコレクション（による見掛け上の中断）への対処としてこまめに強制的なガーベジコレクションを実施している点（DUM #＝FRE(0), [540]），および，HITAC側からの改行記号が〈LF〉＋〈CR〉のため〈LF〉を無視し（[560]）〈CR〉のときに改行している点（[565]）である．改行の明示的出力は画面編集機能を実現する時に行の区切を識別するために必要である．

(3) ホストへの送信

ホストへの送信は，送信すべき行があるとき（KBUF＞0）に下請ルーチン（＊ SEND.LINES）によって行なう（[595]）．送信すべき行は，キーボードからの〈CR〉の入力により“有”となる（[715]）．

送信の下請ルーチンでは，まず，送信行バッファ（LBUF$（　））から1行分を（B$へ）取り出す（[635]）．ここでは，送信行バッファとして1行分しか利用していないが，ここを工夫すれば画面編集機能で複数の行を送信行バッファに蓄え一度の〈CR〉の入力で一括して送信することが可能となる．つぎに，送信先がホスト（M280H）のとき（HOST＝1）には1行（B$）を通信回線に出力する（[640]）．行の終端記号として〈CR〉＋〈LF〉を送ると，M-280H側では〈CR〉を行の終端記号として受付けるため，〈LF〉が残留しブレーク信号によるキャンセル時に不都合が起こることがある．そのため，行の終端記号としては〈CR〉のみを送るようにしている．一方，ローカルOS機能を使うとき（HOST＝2）にはその1行を携えてローカルOSルーチン（＊ LOCAL）に跳ぶ（[645]）．こうしておけば，ホストとローカルのどちらの場合でも同じ画面編集機能が使える．ホストかローカルかの切り換えは，ファンクションキー6（〈f・6〉）による割込み処理で行なっている（＊ INT.LOCAL, [740]）．ローカルOSルーチンでは，B$を解読してパソコン側のファイルの表示・削除・複写・転送などのローカルOS機能を実現することが可能である．ただし，このプログラムでは黄色でエコーバックするだけである（[730]）．

(4) キーボードのスキャン

キーボードのスキャンでは，カーソルを表示し（[610]），キー入力があれば画面編集ルーチン（＊ SCRN.EDIT）に跳び（[615]），なければ受信バッファのスキャンに戻る（[620]）．キー入力は白で表示している（COLOR 7, [605]）．カーソル表示の制御には機械語ルーチンを用いている（CALL CURON および CALL CUROFF）．

画面編集ルーチンでは，まず，制御文字か通常文字かを識別し（[665]），前者ならば画面編集処理を行ない（＊ ED.IT），後者ならばその文字を画面に表示するとともに入力行バッファ（L$）に蓄える（＊ E.ANK. [700]）．ここでは〈CR〉と〈BS〉の処理以外に画面編集処理は何もやっていないが，制御文字に応じた処理を工夫することによって，いろいろな画面編集機能を実現することが可能である．制御文字が〈CR〉のときは，改行を画面に出力し入力行バッ

ファを送信行バッファにプッシュするとともに先にも述べたように送信行“有”にする（[710]～[720]）.〈BS〉のときはホストがサポートしてくれるのでこのプログラムでは通常文字と同様に扱っている([685]). しかし，ローカルOSルーチンではそうはいかないので実際にはきちんとした処理が必要である.

(5) ブレーク信号の送信

ブレーク信号の送信は，ファンクションキー10(〈f•10〉)による割込み処理で行なう(＊INT. BREAK, [750]～[765]). PC-98では，I/Oポートの32h番に3Fhと37hを適当な時間をおいて出力すればよい.

(6) 初期設定ルーチン

説明するのが最後になったが，初期設定ルーチンでは，画面の設定（[775]），機械語ルーチンのロード（[780] および [820]～[875]），文字定数の設定（[785]），通信回線のオープン([790]), 入力行および送信行バッファの初期化([800]), ファンクションキー割込の設定([805]および [885]～[910]）などを行なっている.

ここで，参考のために主な文字型変数の使用目的をまとめておく.

H$····ホストからの受信用
K$····キーボードからの1文字入力
L$····入力行バッファ
LBUF$····送信行バッファ
B$····ホスト（またはローカル）への送信用

これらの使い方は，次節の簡略版のプログラムでも同じである.

4. 簡略版端末プログラム ——TERM98TN2——

前節のプロトタイプ版でTERM98TNの基本的構造を示した. プロトタイプ版でもホストとの通信はできるが, 機能的には中途半端である. 画面編集機能とローカルOS機能の仕組はあるものの実際には何の機能も果していない. 紙面の制約上，完成版のTERM98TNプログラム全部を掲載するわけにいかないので，ここでは完成版のほぼ半分にあたる簡略版を図3に示す.

簡略版の基本構造はプロトタイプ版・完成版と同じである. 主に，画面編集機能とローカルOS機能を簡略化して実現した形になっている. その他に，セッションのプリンタ出力機能とLOGON/LOGOFF機能も使える. 画面編集機能としては，カーソルの上下左右への移動,〈BS〉と〈DEL〉による1文字削除,〈INS〉による挿入モード切換機能が使える. また,〈CR〉によってそれまでに編集した複数の行を一括して送信することができる. ローカルOS機能では，PC-98側のディレクトリ表示，ファイルの表示•削除•複写など，およびホストとPC-98間でのファイル転送を行なうことができる. ファイル転送以外はMS-DOS/BASICのチャイルドプロセス制御機能を用いて実現している.

以下で，簡略版に追加した部分について解説する.

(1) メインループ

メインループの部分は，プロトタイプ版とほとんど同じである. 違うのは，セッションのプ

リンタ出力機能のために変更したところと（[1180]，[1200]，[1440] および [2640]），ホストへの送信の仕組を変更したところである（[1240] および [1280]～[1300]）．セッションのディスク出力機能も，プリンタ出力機能と同様にして取り入れることができる．ただし，出力ファイルのオープン/クローズ処理が必要である．

ホストへの送信の仕組の変更は，複数の行を一括して送信するための変更である．ローカルOSの方へ複数の行をまわす場合とホスト（M-280H）へ1行目を送信する場合（[1280]～[1290]）はホストからの応答を待たずに行ない，ホストへの2行目以降の送信はホストからの応答を待ってから行なっている（[1240]）．後者については厳密な処理ではないが，全二重のため画面表示が汚れる程度で実害はない．

キーボードのスキャン部は実質的には前とまったく同じである．ただし，完成版で特殊なキーセンスを行なうのでサブルーチン化した（＊ KEY.SENSE, [1720]）．

送信の下請ルーチン（＊ SEND.LINES, [1370]～[1420]）には追加・変更がある．まず，画面編集のためホストからのプロンプト（ここでは，>>と E> を想定）が送信行の先頭に付く場合があるのでそれを削除している（[1380]，[1390]）．つぎに，送信行が2行以上ある場合にそれらを送信するたびに画面へ緑色でエコーバック表示するようにしている（GOSUB ＊ SEND.HOST.ECHO, [1400]）．最後は，送信行がなくなったかどうかを調べ（KBUF＞＝LBUF），なくなっていれば送信行バッファをリセット（KBUF＝0：LBUF＝0, [1410]），まだあるならばつぎの送信行を準備する（KBUF＝KBUF＋1, [1410]）．

(2) 入力行バッファ

画面編集機能を実現するために入力行バッファ（L$）を設定している．これを制御するパラメータが，P, Q, R, S, MAXL である．P は入力行バッファ上でのカーソル位置を表わし，0≦P≦S である．Q は有効行頭，R は有効行末，S は論理行長，MAXL は入力行バッファ長を表わし，0≦Q≦R≦S≦MAXL(＝254)という関係を維持している．ここで，有効行というのは入力行バッファのうち実際に送信する部分のことである（＊ PUSH.LBUF, [1690]を参照）．カーソル移動の際に P が [Q, R] の範囲外へ出ることもある．論理行とは画面上では切れていても論理的には繋がっている行のことであり，S＝80, 160, 240, MAXL という値をとる．入力行バッファ長 MAXL は 254 に固定されている．

入力行バッファL$のリセットはサブルーチン＊ CLR.LINE（[1490]）で行なう．MAXL 個の空白文字列として L$を確保した後，パラメータを P＝Q＝R＝0, S＝80 とする．

入力行バッファの初期セットはサブルーチン＊ GET.LINE（[1500]～[1520]）で行なう．入力行バッファをリセットした後，機械語ルーチン LINEINFO によって画面上のカーソル位置を含む論理行の連続状態を調べ，それに従い機械語ルーチン GETVRAM によって VRAM から文字列を転送してくる．

入力行バッファの更新は画面編集ルーチンによって行なう．

(3) 画面編集ルーチン

キー入力があると画面編集ルーチン（＊ SCRN.EDIT, [1750]）に跳んでくる．入力文字が制御文字か通常文字かを調べ，制御文字ならばまず＊ ED.IT（[1780]～[1830]）に進みそこから各機能に分岐する．通常文字ならばその文字を入力行バッファに加える（＊ E.ANK, [1850]～[1910]）．このとき，挿入モードか否かで操作が異なる．

この簡略版では，制御文字による画面編集機能として，〈↑〉〈↓〉〈←〉〈→〉によるカーソルの上下左右移動（＊ E.AL, ＊ E.AR, ＊ E.AU, ＊ E.AD, [2110]～[2190]），〈BS〉と〈DEL〉

による1文字削除（＊ E.BS, ＊ E.DL, [1990]～[2070]），〈CLR〉による画面クリア（＊ E.CL, [2210]），〈INS〉による挿入モードの設定／解除（＊ E.IN, [2090]），そして〈CR〉による送信開始（＊ E.CR, [1930]～[1970]）をサポートしている．詳細については省略するが，画面編集すなわち入力行バッファの更新では，一般につぎのような手順を踏む．(i) 入力行バッファ・パラメータの状態からその機能が使えるかを調べる．(ii) 機能に応じ入力行バッファを更新する．(iii) バッファ・パラメータの状態を更新する．(iv) 画面表示およびカーソル位置を更新する．(v) 入力行バッファの内容が変化していれば変更“有”とする（CHANGE＝YES）．機能によっては以上の手順の中で省略できる操作もある．

カーソル上下の機能では，編集している行から抜け出すことがある．このとき，変更“有”ならば入力行バッファの内容を送信行バッファに移し（＊ PUSH.LBUF, [1690]），新しい行を入力行バッファに取込む．この操作はサブルーチン＊ XCHG.LINE（[1610]～[1660]）で行なっている．

〈INS〉キーはトグルになっており，挿入モードの設定／解除を行なう（＊ INS.XCHG, [2350]）．挿入モードにあるときはカーソルの形状が下線状になる．カーソルの形状の設定は機械語ルーチン（CALL INSON および CALL INSOFF, [2360], [2370]）による．〈CR〉キー入力により挿入モードは解除される（[1970]）．

(4) ローカル OS 機能

端末プログラムを終了することなく PC-98 側のファイル操作やホストとの間でのファイル転送が行なえるようにローカル OS 機能を設けている．ファイル転送と FILES コマンド以外は MS-DOS/BASIC のチャイルドプロセス制御機能を利用しているので，MS-DOS のコマンドがそのまま使える．チャイルドプロセス制御機能を使うためには，そのための領域を CLEAR 文により確保する必要がある．（CLEAR &H100....&H1000 [1060]）

このローカル OS 機能に入るには〈f・6〉を押す．画面右上にローカル OS モードの表示が出，MS-DOS と同じようにカレント・ドライブを表わすプロンプトが現われる．このキー割込により，HOST＝2 となり（＊ LOCAL.START, [2700]～[2710]），以後はホストへ送信する代りにローカル OS ルーチンが働くことになる．したがって，画面編集機能は共通に使える．

ローカル OS 機能を抜けるには〈f・6〉を押すか，QUIT コマンドを入力する．そうすると，HOST＝1 になり（＊ LOCAL.QUIT, [2720]～[2730]），以後はホストへ送信される．

コマンドは，A：などのカレント・ドライブの変更，COPY, DEL, DIR, PRINT, TYPE など，MS-DOS のものがそのまま使える．FILES コマンドは DIR と同じ機能だが，チャイルドプロセスによらないのでその分時間がかからない（＊ LOCAL.FILES, [2990]）．ファイルの編集には，MS-DOS の EDLIN などを使えばよい．ただし，MS-DOS の外部コマンドはそのプログラムをユーザ側で準備しておかなければならない．これらのコマンドの実行後，すぐに TERM98TN に復帰する（＊ LOCAL.CHILD, [2950]）．しばらく MS-DOS の機能を使う場合は，DOS コマンドによって MS-DOS に移っておくのがよい（＊ LOCAL.DOS, [2970]）．TERM98TN への復帰には，MS-DOS の EXIT コマンドを用いる．

ホストとのファイル転送には，ローカル OS 機能の ULOAD, DLOAD コマンドを用いる．

ホストへファイルを転送するには，

ULOAD filename

とする（＊ LOCAL.ULOAD, [3010]～[3210]）．filename は転送元である PC-98 側のファイルの名前である．転送には VOS3/EDIT の DSI コマンドを利用している．そのため，ホストに転

送先のデータセットをEDITモードで準備しておく必要がある．

ホストからファイルを転送してくるには，

DLOAD filename

とする（＊ LOCAL.DLOAD, [3230]～[3420]）．filenameは転送先であるPC-98側のファイルの名前である．転送にはVOS3/EDITのDSOコマンドを利用している．そのため，転送したいホスト側のデータセットをEDITモードにしておく必要がある．

(5) 機械語ルーチン

TERM98TNでは，処理の高速化とテキスト領域の節約のために，いくつかの機械語ルーチンを使っている．TERM98TN2では，機械語コードをDATA文にもっている（[4290]～[4730]）が，完成版では機械語ファイルからBLOADステートメントによりインストールするようにしている．

MS-DOS/BASICでは

```
CLEAR &H100 : MACHINE#=SEGPTR(2) : DEF SEG=MACHINE#
```

のようにして，機械語領域の確保およびセグメントの設定を行なう．機械語ルーチンの呼び出しはすべてCALLステートメントによっている．この方が，ルーチンの意味をわかりやすくできるからである．

次に，機械語ルーチンの機能と使い方を整理しておく．

CALL CURON‥‥カーソルを表示する．

CALL CUROFF‥‥カーソル表示を消す．

CALL INSON‥‥挿入モード用にカーソル形状を下線状にする．

CALL INSOFF‥‥カーソル形状を通常の状態に戻す．

CALL LINEINFO (YTOP, YBOT, Z, S, P)‥‥カーソルがある論理行についての情報を返す．

(IN) [YTOP, YBOT]：画面の範囲（YTOP＝0, YBOT＝23）

(OUT) Z：論理行頭のオフセット値
S：論理行長
P：論理行上のカーソル位置

CALL GETVRAM (Z, N, L$, R)‥‥VRAM1から文字列を転送する．

(IN) Z：論理行頭のオフセット値
N：転送文字数

(OUT) L$：転送文字列
R：有効行末

CALL PUTATTR (X, N, A)‥‥文字アトリビュートを変更する．

(IN) X：先頭のオフセット値
N：変更文字数
A：アトリビュート

CALL TOUPPER (X$)‥‥大文字へ変換

(IN/OUT) X$：文字型変数

以下に，簡略版にはないが完成版にある機械語ルーチンの使い方も並べておく．

CALL CRXLF (H$, XLF, XCR, XBL)‥‥受信コードを変換する.
(IN/OUT) H$: 文字型変数
(IN) XLF: 〈LF〉の代替コード
XCR: 〈CR〉の代替コード
XBL: 〈BELL〉の代替コード
CALL KEYSENSE (KY, SH)‥‥キーボードをセンスする.
(OUT) KY: 押下キー番号
SH: シフトキー状態
CALL SHKEY (SH)‥‥シフトキーをセンスする.
(OUT) SH: シフトキー番号
CALL PUTKBUF (X$)‥‥キーボード・バッファに文字を戻す.
(IN) X$: 文字型変数
CALL VRAM12 (Z, M, N, K)‥‥VRAM1 と VRAM2 の間で矩形領域の文字列を転送する.
(IN) Z: VRAM1 の先頭オフセット値
M: 転送行数
N: 転送列数
K: 転送方法 (0: VRAM1 をクリア;
1: VRAM1 ← VRAM2;
2: VRAM2 ← VRAM1)

CALL TEKTRO (FLAG, H$)‥‥TEKTRONIX 描画
(IN/OUT) FLAG: 状態フラグ
H$: 受信コード (文字型変数)

カーソルの X, Y 座標などの情報がある BASIC のワークエリアは, スタンドアロン BASIC ではセグメント 0060h にあり多くの文献でその内容を知ることができる (たとえば, 藤田・幸田(1984)など). オフセット値はほとんど同じなので, MS-DOS/BASIC でもこれが利用できる. セグメントはスタックセグメントを参照すればよく, push ss/pop ds などとしてセットすればよい.

機械語ルーチンの開発は, MS-DOS の MASM, LINK, EXE2BIN を利用すれば容易に行なえる. EXE2BIN でできあがった .BIN ファイルを直接 BLOAD ステートメントで該当領域にロードすればよい.

5. TERM98TN ユーザーズ・マニュアル (Ver 3.2)

[1] TERM98TN の概要

TERM98TN は, HITAC-M シリーズの TSS 端末としてパーソナル・コンピュータ NEC/PC-9801 シリーズを利用するために開発された画面編集機能をもつフロントプロセッサである. TERM98TN は MS-DOS 版 N88-BASIC(86)インタプリタ上で作動する. 作動環境は, 以下のとおりである.

- PC-9800 シリーズ本体 (PC-9801E/F/M/V)
- メイン・メモリ (RAM) 384KB 以上

・専用高解像度ディスプレイ
・ディスク・ユニット（8インチ，5インチ，ハード・ディスク）
・プリンタ（オプション）
・MS-DOS Ver 2.11 システム・ディスク
・MS-DOS 版 N88-BASIC(86) インタプリタ Ver 1.0

TERM98TN を起動すれば，そのままでごく普通の行編集端末として用いることができる．画面編集機能を使わなければ操作に関して何ら特別の知識を必要としない．一方，これから解説する操作を用いて画面上にある文字列を適当に編集して送信すれば，便利な画面編集端末として使うことができる．ファイル転送機能を含むローカルOS機能も有しており，ホストとの交信途中あるいは終了後も効率的にデータを処理することができる．ファイル転送以外のローカルOS機能はMS-DOS版N88-BASIC(86)のチャイルドプロセス制御機能を利用しており，TERM98TN を終了せずにMS-DOSのコマンドが実行できる．

送信したい文字はキーボードからそのまま入力すればよい．画面編集機能は特殊キーや制御キーを用いて実行できる．また，いくつかの機能は表に示すファンクションキー割込みによって利用できる．

開始キー	機　　能	取消キー
〈f・1〉	LOGON／LOGOFF	〈ESC〉
〈f・2〉	シークレット	〈CR〉
〈f・3〉	TEKTRONIX 4010	〈ESC〉
〈f・4〉	改行記号の変更	
〈f・5〉	入力履歴（HISTORY）	
〈f・6〉	ローカルOS	〈f・6〉, QUIT
〈f・7〉	セッションのプリンタ出力	〈f・7〉
〈f・8〉	セッションのディスク出力	〈f・8〉
〈f・9〉	オプション／イディオム	〈ESC〉
〈f・10〉	ブレーク信号の送信	

画面上では，一般に，編集中の行は白，送信する行は緑，受信した行は水色で表示される．表示色によらず画面上に存在する文字は利用することができる．編集中のモードの変更は，カーソルの色・形状の変化，または画面右上へのモード表示によって示される．

[2] TERM98TN の起動と終了

(1) TERM98TN の起動

〈1〉 MS-DOS を起動する．

起動パラメータとして，CONFIG.SYS ファイル内に，

FILES＝9

を設定しておく必要がある（9以上であればよい）．

〈2〉 カレントディレクトリを変更する．

TERM98TN で使用するファイルはディレクトリ¥TERM98 下にまとめられているので，ここにカレントディレクトリを移しておく方がよい．すなわち，

A>cd ¥term98

とする（A> は MS-DOS のプロンプトである．下線のところが入力部分である．入力の最後に〈CR〉が必要であるが，ここでは省略している）．

〈3〉 N88-BASIC(86) を起動する．

A>n88basic /f:4

同時ファイル OPEN 数は4以上にしておく．N88BASIC がカレントディレクトリ¥TERM98 にない場合は，MS-DOS の PATH コマンドにより N88BASIC の存在するディレクトリを指定しておく必要がある（たとえば，PATH＝¥BASIC のように）．

〈4〉 TERM98TN を起動する．

通常の方法でプログラムをロードし実行すればよい．すなわち，

load "term98"
run

または，

run "term98"

とする．

あるいは，〈2〉，〈3〉のところで，

A>cd ¥term98
A>n88basic /f:4/t:run"term98"

とすれば，カレントディレクトリの変更，BASIC と TERM98TN の起動が一度にできる．これらを適当なバッチファイル（たとえば，TERM98.BAT）として作成しておけばバッチファイルの名前を入力するだけでよいので便利である．

(2) TERM98TN の終了

〈STOP〉または〈CTRL・C〉を押下する．

Quit〈Q/〉, Save & Quit〈S〉, or Continue〈C〉?

の確認が来る．'Q' または〈CR〉を応答すれば TERM98TN が終了し，BASIC に戻る（TSS のセッションを LOGOFF し忘れないように注意）．再び TERM98TN を起動したければ

run （または，〈f・5〉）

とすればよい．MS-DOS に戻るには，

system （または，〈f・1〉+〈CR〉）

とする．一方，'S' を応答すればメモ内容をメモ・ファイルにセーブして TERM98TN を終了する．

誤って上記の終了キーを押してしまったような場合は，'C' を応答すれば，TERM98TN に復帰することができる．うまく復帰できなければ，次の異常終了時の処置をとる．

(3) 異常終了の場合

〈a〉 再び，run を行なう．通常，これで復帰する．

〈b〉 プログラムを再ロードし，実行する．
〈c〉 システムをリセットし，TERM98TNを再起動する．
〈d〉 原因不明の場合は，開発者に連絡をとる．

(4) 必要とするファイル

〈a〉 TERM98.BAS‥‥TERM98TN本体のBASICプログラム
〈b〉 TERM98.BIN‥‥機械語ファイル
〈c〉 TERM98.ADR‥‥機械語ルーチン・アドレス・ファイル
〈d〉 TERM98.HLP‥‥ヘルプ・メッセージ・ファイル
〈e〉 TERM98.UID‥‥ユーザIDファイル（オプション）
〈f〉 TERM98.MMO‥‥メモ・ファイル（オプション）
〈g〉 TERM98.IDM‥‥イディオム・ファイル（オプション）
〈h〉 TERM98.SAV‥‥セッションのディスク出力用ファイル（オプション）
〈i〉 TERM98.BAT‥‥TERM98TN起動用バッチ・ファイル（オプション）

これらは，カレントディレクトリである¥TERM98下にまとめておく．

[3] 画面編集機能

(1) 行の送信/画面操作

機能キー	機能	同一機能キー
〈CR〉	変更行および編集行を送信する．	
〈SHFT・CR〉	変更行を送信する．（編集行は送信しない）	〈GRPH・CR〉
〈CTRL・U〉	編集行の編集を取り消す．（カーソルは画面の最下端に移る）	
〈CTRL・V〉	複数の変更行を送信する際に，再表示するかどうかのトグルスイッチ．（起動時はon）	
〈CLR〉	全変更行をクリアし，送信を取り消す．（テキスト画面もクリアする）	
〈SHFT・CLR〉（=〈HOME〉）	全変更行をクリアし，送信を取り消す．（テキスト画面はクリアしない．カーソルは画面の最下端に移る）	
〈GRPH・CLR〉	グラフィック画面をクリアする．（変更行はクリアしない）	〈GRPH・DEL〉
〈CTRL・W〉	画面のスクロール範囲を設定/解除する．（カーソル以下をスクロール範囲とする．白い水平線が引かれ，その範囲を示す．再び，〈CTRL・W〉を押下すれば解除される）	
〈CTRL・Z〉	【画面のゾーン操作モード】に入る．（カーソルを操作したい画面範囲の左上または右下に移動し，この〈CTRL・Z〉を押下する）	

【画面のゾーン操作モード】

〈ESC〉	何もせず，本モードを終了する．	
〈→〉	1桁右へ移動．	〈CTRL•L〉
〈←〉	1桁左へ移動．	〈CTRL•H〉
〈↑〉	上の物理行の同じ桁へ移動．	〈CTRL•K〉
〈↓〉	下の物理行の同じ桁へ移動．	〈CTRL•J〉
L	〔ロード〕直前に削除，セーブ，複写，移動されたゾーンを指定位置にロードし，本モードを終了する．	〈f•1〉
S	〔セーブ〕指定ゾーンをセーブし，本モードを終了する．	〈f•2〉
D	〔削　除〕指定ゾーンを削除し，本モードを終了する．	〈f•3〉，〈DEL〉，E
C	〔複　写〕指定ゾーンを指定位置に複写し，本モードを終了する．これを押下後，指定位置までカーソルを移動し，〈CR〉を押下する．	〈f•6〉
M	〔移　動〕指定ゾーンを指定位置に移動し，本モードを終了する．これを押下後，指定位置までカーソルを移動し，〈CR〉を押下する．	〈f•7〉

ここで，「変更行」とは画面上の物理的な行のこと，「論理行」とは編集・送信の単位となる行，「編集行」とは現在編集している論理行（白で表示されている），「送信行」とはすでに編集された論理行であるがまだ送信されていない行（一般に緑で表示されている）のことである．

(2) カーソルの移動

機能キー	機　　能	同一機能キー
〈→〉	1桁右へ移動．	〈CTRL•L〉
〈SHFT•→〉	論理行の末尾（の次）へ移動．	〈CTRL•X〉
〈CTRL•→〉	後方サーチの【カーソル移動モード】へ．	〈CTRL•F〉
〈GRPH•→〉	8桁右へ移動．	
〈←〉	1桁左へ移動．	〈CTRL•H〉
〈SHFT•←〉	論理行の先頭へ移動．	
〈CTRL•←〉	前方サーチの【カーソル移動モード】へ．	〈CTRL•B〉
〈GRPH•←〉	8桁左へ移動．	
〈↑〉	上の物理行の同じ桁へ移動．	〈CTRL•K〉
〈SHFT•↑〉	5つ上の物理行の同じ桁へ移動．	
〈CTRL•↑〉	論理行頭に移動し，後方サーチの【カーソル移動モード】へ．	〈SHFT•←〉＋〈CTRL•→〉
〈GRPH•↑〉	直前に〈CR〉を押した位置に移動．	
〈↓〉	下の物理行の同じ桁へ移動．	〈CTRL•J〉
〈SHFT•↓〉	5つ下の物理行の同じ桁へ移動．	
〈CTRL•↓〉	論理行末に移動し，前方サーチの【カーソル移動モード】へ．	〈SHFT•→〉＋〈CTRL•←〉
〈GRPH•↓〉	画面の最末尾へ移動．	

〈TAB〉	次のタブ位置へ移動.	
〈SHFT・TAB〉	直前のタブ位置へ移動.	
〈GRPH・TAB〉	タブ位置表示（垂直線）のトグルスイッチ.（起動時はOFF）	〈CTRL・TAB〉

【カーソル移動モード】（カーソルが緑色になりブリンクが止まる）

〈ESC〉	本モードを終了する.	〈CR〉 〈↑〉, 〈CTRL・K〉 〈↓〉, 〈CTRL・J〉
〈→〉	カーソルを右に移動し，本モードを終了する.	〈CTRL・L〉
〈←〉	カーソルを左に移動し，本モードを終了する.	〈CTRL・H〉
〈CTRL・B〉	前の語の先頭へ移動，または，サーチ方向を反転.	〈CTRL・←〉
〈CTRL・F〉	次の語の先頭へ移動，または，サーチ方向を反転.	〈CTRL・→〉
〈大文字〉	サーチ方向のその文字位置までカーソルを移動.（見つからない場合は移動しない）	
〈小文字〉	サーチ方向のその文字位置とその文字の大文字位置の近い方までカーソルを移動.（見つからない場合は移動しない）	
〈記号〉	サーチ方向のその記号位置までカーソルを移動.（見つからない場合は移動しない）	

カーソル移動によって別の論理行に移る場合，編集行に何らかの変更がなされていれば，その行は「変更行」となり緑に変化する．変更がなされていなければ，変更行とはならないが水色に変化する．

(3)　文字の削除

機能キー	機　　能	同一機能キー
〈DEL〉	カーソル位置にある文字を削除し，その論理行を詰める.（カーソルが論理行末の次にあるときは〈BS〉と同じ）	
〈BS〉	カーソル位置を1桁左に戻し，〈DEL〉の操作を行なう.（論理行頭では何もしない）	
〈CTRL・D〉	カーソルの位置が語の中であれば，その語のカーソル位置以降を削除し，その論理行を詰める．語の中でなければ，〈DEL〉の操作を行なう.	
〈CTRL・E〉	その論理行のカーソル位置以降を削除する.	

(4)　文字の挿入

機能キー	機　　能	同一機能キー
〈INS〉	上書き/挿入モードの切換えトグル.（起動時は上書きモード．挿入モードの時はカーソルの形状が下線状’’になる）	

〈SHFT・INS〉	挿入モードの継続/解除の切換えトグル． （〈CR〉のとき挿入モードを解除するかを設定する．起動時は解除）	〈CTRL・INS〉 〈GRPH・INS〉

(5) タブ位置の設定

機能キー	機能	同一機能キー
〈CTRL・T〉	【タブ設定モード】に入る． （画面右上にモード表示がなされ，既存のタブ位置に垂直線が引かれる）	

【タブ設定モード】

〈ESC〉	本モードを終了する．	〈CTRL・T〉
〈→〉	1桁右へ移動．	〈SP〉，〈CTRL・L〉
〈SHFT・→〉	8桁右へ移動．	〈CTRL・→〉 〈GRPH・→〉
〈←〉	1桁左へ移動．	〈BS〉，〈CTRL・H〉
〈SHFT・←〉	8桁左へ移動．	〈CTRL・←〉 〈GRPH・←〉
〈↑〉	カーソルを最左端（第1桁）へ移動．	〈CTRL・K〉
〈↓〉	カーソルを最右端（第80桁）へ移動．	〈CTRL・J〉
〈TAB〉	次のタブ位置へ移動．	
〈SHFT・TAB〉	直前のタブ位置へ移動．	
〈XFER〉	タブの設定/解除． （カーソル位置にタブを設定/解除する）	
C	タブ設定のクリア．	〈f・1〉，〈CLR〉
D	標準（8桁ごと）のタブ設定．	〈f・2〉
F	FORTRAN プログラム用のタブ設定．	〈f・3〉
V	タブ位置表示（垂直線）のトグルスイッチ （ON であれば黄色，OFF であれば白で表示）．	〈f・4〉

(6) 文字列の抽出/文字列の複写・移動・削除

機能キー	機能	同一機能キー
〈XFER〉	【文字列抽出モード】に入る． （画面右上にモード表示がなされ，既存のタブ位置に垂直線が引かれる）	
〈CTRL・Q〉	抽出した文字列を出力する． （文字列抽出モードでの削除・複写・移動の操作と組み合せることにより，結果的に文字列の削除・複写・移動が行なえる．新たに文字列を抽出しないかぎり何度でも同じ文字列を出力できる）	

【文字列抽出モード】（カーソルが黄色になりブリンクが止まる）

〈ESC〉	何もせず，本モードを終了する．	〈SHFT・CR〉
〈CR〉	文字列を抽出するとともに，タグなしでメモにプッシュして，本モードを終了する．	
〈f・3〉	〔削　除〕文字列を抽出し，本モードを終了する．文字列は削除する．	〈DEL〉
〈f・6〉	〔複　写〕文字列を抽出し，本モードを終了する．終了後，挿入モードになる．	
〈f・7〉	〔移　動〕文字列を抽出し，本モードを終了する．文字列を削除し，終了後，挿入モードになる．	
〈英数字〉/〈記号〉	文字列を抽出するとともに，入力文字をタグとしてメモにプッシュし，本モードを終了する．	
〈→〉	カーソルを1桁右へ移動し，抽出文字列の範囲を変更する．	〈SP〉，〈CTRL・L〉
〈←〉	カーソルを1桁左へ移動し，抽出文字列の範囲を変更する．	〈BS〉，〈CTRL・H〉
〈↑〉	カーソルを論理行の先頭へ移動し，抽出文字列の範囲を変更する．	〈CTRL・K〉
〈↓〉	カーソルを論理行の末尾へ移動し，抽出文字列の範囲を変更する．	〈CTRL・J〉

(7)　メ　　モ

メモ・ファイルTERM98.MMOがあれば，起動時にメモ内容が自動的にロードされている．メモ内容の追加は前項の文字列の抽出による．また，メモ・ファイルを直接編集してもよい．メモ・ファイルの構造はシーケンシャルなアスキー・ファイルであり，1行が1つのメモ内容，各行の先頭1文字がタグである．コントロール・キャラクタも"^B"などとして利用できるのでかなり複雑な編集操作も可能である．

コントロール・キャラクターは一般に"^B", ..., "^Z"という文字列で代用する．ただし，〈→〉は"^L"，〈←〉は"^H"，〈↑〉は"^K"，〈↓〉は"^J"，〈BS〉は"^A"，〈CLR〉は"^C"，〈DEL〉は"^O"，〈ESC〉は"^S"で代用する．また，"^0", ..., "^9"によって直後の文字をその数字回分繰り返すことを指定できる．

機能キー	機　　能	同一機能キー
〈ROLL・UP〉	【メモ選択出力モード】に入る．	〈CTRL・M〉 〈CTRL・CR〉
〈ROLL・DOWN〉 ＋〈タグ文字〉	タグの一致するメモを出力する．	〈CTRL・Y〉 ＋〈タグ文字〉

【メモ選択出力モード】（メモ内容が黄色で表示される）

〈ESC〉	何もせず，本モードを終了する．	〈ROLL・DOWN〉
〈CR〉	表示されているメモ内容を出力して，本モードを終了する．	

〈ROLL•UP〉	次のメモ内容を表示する.	〈SP〉
〈BS〉	直前のメモ内容を表示する.	
〈DEL〉	表示されているメモ内容を削除する.	
〈INS〉	上書き/挿入モードの切換えトグル.	
〈タグ文字〉	タグの一致するメモを出力する.	
〈f•1〉	全メモ内容をクリアする.	〈CLR〉
〈f•3〉	メモ内容をプリンタに出力する.	
〈f•4〉	メモ内容をメモ・ファイルに書き込む.	
〈f•5〉	メモ内容をメモ・ファイルから読み込む.	

[4] LOGON/LOGOFF 機能

ユーザ ID ファイル TERM98.UID を用意しておけば，起動時に自動的にインストールされ，LOGON コマンドの投入が簡単になる．このファイルはシーケンシャルなアスキー・ファイルであり，適当なエディタで作成・更新すればよい．内容は，1行ごとに

userid/passwd

とする．合計8組まで登録しておくことができる．LOGON コマンドを送信するときに，"/passwd" 以下は画面に表示されない．

機能キー	機　　能	同一機能キー
〈f•1〉	《LOGON/LOGOFF 機能》の開始.	

《LOGON/LOGOFF 機能》

〈ESC〉	何もせず，本機能を終了する.	
〈CR〉	LOGOFF を送信し，本機能を終了する.	〈0〉，〈f•10〉
〈1〉～〈9〉	対応する番号の userid/passwd を LOGON 付で送信し，本機能を終了する.	〈f•1〉～〈f•9〉
〈SHFT•1〉～〈SHFT•9〉	対応する番号の userid/passwd 送信し，本機能を終了する．LOGON は付加しない.	〈SHFT•f1〉～〈SHFT•f9〉
L	ユーザ ID ファイル TERM98.UID を再ロードする．ユーザ ID ファイルを変更した後などに用いる.	

[5] シークレット機能

キーボードからの入力を〈CR〉が入力されるまで直接ホストに送信する．カーソルは移動するが，画面への表示は行なわない．パスワードを単独で入力するときなどに利用する．

機能キー	機　　能	同一機能キー
〈f•2〉	《シークレット機能》の開始.	

《シークレット機能》

〈CR〉	シークレット機能の終了.

[6] TEKTRO機能

XYプロッタ出力を画面に表示する機能である．

〈1〉 必要な資源を割り当てるために，セッションで一度"galloc"を送信しておく．

〈2〉 XYプロッタ出力を実行するFORTRANプログラムをVOS3のEDITモードにする．

〈3〉 TEKTRO機能を実行する．すなわち，〈f･3〉を押下する．これにより"grun98"が送信される．TERM98TNは受信コードを解読し，グラフィック画面上にXYプロッタ出力を描画する．

機能キー	機　　能	同一機能キー
〈f･3〉	《TEKTRO機能》の開始．	

《TEKTRO機能》

〈ESC〉	本機能を終了する． （これを押下しなければ本機能は終了しない．描画が完了したことを確認すること）
〈DEL〉	画面をクリアし，本機能を終了する．
〈f･10〉	ブレーク信号を送信し，本機能を終了する． （描画を途中で中止するときに用いる）
〈CLR〉	テキスト画面をクリアする．
〈COPY〉	ハードコピー出力．

[7] 改行記号の変更

受信行の改行記号として〈CR〉をとるか，〈LF〉をとるかを指定する．〈CR〉をとった場合，〈CR〉の受信時に画面と（プリンタ）に改行出力を行ない，〈LF〉の時には無視する．〈LF〉をとった場合はその逆である．それぞれの場合，ファンクションキー4の表示エリアに，"¥n＝CR"あるいは"¥n＝LF"を表示する．FORTRANプログラムの実行時には，改行記号として〈CR〉＋〈CR〉＋〈LF〉が送られてくるので，"¥n＝LF"を指定するのがよい．

機能キー	機　　能	同一機能キー
〈f･4〉	改行記号の指定変更． （トグル．起動時は"¥n＝CR"）	

[8] 入力履歴（HISTORY）機能

TERM98TNは，これまで送信した2文字以上からなる行を，最高30行まで自動的に記憶している．このキーを押すごとに過去の行にさかのぼり順次表示する．適当な行をそのまま．あるいは適当に編集してから送信すればよい．

機能キー	機　　能	同一機能キー
〈f･5〉	過去の入力行の想起．	

[9] ローカルOS機能とファイル転送

(1) ローカルOS機能

パソコン側のファイル操作，パソコン・ホスト間のファイル転送などが，TERM98TNを終了せずに実行できる．パソコン・ホスト間のファイル転送機能とFILESコマンド以外はBASICのチャイルドプロセス制御機能を利用しており，MS-DOSのコマンドをそのまま用いてファイルの削除・複写・表示・名前変更などが行える．ファイルの編集にはEDLINを用いればよい．ただし，チャイルドプロセス領域として65KBしか確保していないので，実行可能なプログラムは47KBまでの大きさのものである．

機能キー	機能	同一機能キー
〈f•6〉	《ローカルOS機能》の開始．	

《ローカルOS機能》

QUIT	本機能を終了する．	〈f•6〉
FILES	FILES [pathname] ファイル名を表示する． (" "は必要ない．同じ機能のDIRはチャイルドプロセスによる)	
DOS	DOS MS-DOSに制御が移る． (TERM98TNに戻るにはEXITコマンドを実行する．カレント・ディレクトリを変更した場合は，復帰前にカレント・ディレクトリを¥TERM98に戻しておく必要がある)	
MS-DOSコマンド	MS-DOSコマンド チャイルドプロセスによりMS-DOSコマンドを実行する．実行後，TERM98TNに復帰する．	
¥TSSコマンド	¥TSSコマンド 先頭に¥記号があれば，ホスト側へコマンドを送信する．	
ULOAD	ULOAD [pathname] 〔アップロード〕ホストへファイルを転送する．(転送中止は〈ESC〉)	
DLOAD	DLOAD [pathname] 〔ダウンロード〕ホストからファイルを転送する．(転送中止は〈ESC〉)	
SEND	SEND [pathname] 〔センド〕ホストへファイル内容を送信する．(転送中止は〈ESC〉)	

(2) ファイル転送

パソコン・ホスト間のファイル転送には，ローカルOS機能のULOAD, DLOADコマンドを用いる．これらはVOS3/EDITのそれぞれDSI, DSOコマンドを発行してファイル転送を行なっている．したがって，ホスト側のファイルをEDITモードにしておく必要がある．転送で

きるのは，シーケンシャルなアスキー・ファイルである（双方のファイルのレコード長に注意）．

ファイル転送の別法は，アップロードとしてローカルOS機能のSENDコマンドを，ダウンロードとしてセッションのディスク出力を用いることである．ただし，ULOAD, DLOADコマンドのようにパッケージ化されていない．

以下に，ULOAD, DLOADコマンドによる転送手順例を示す．ここでは，パソコン側のファイルをPC98.DAT，ホスト側のファイルをM280H.DATAと仮定する．

〈1〉　パソコンからホストへ

転送先のファイルを（新規に作成し，）EDITモードにする．

```
>>edit m280h.data,new,nonum〈CR〉
>>00100 : 〈CR〉
```

ローカルOS機能を開始し，

```
E>〈f・6〉
```

アップロード・コマンドを投入する．

```
A>uload pc98.dat〈CR〉
```

次のようなメッセージが出力されるので

```
Up-load : HOST 〈＝＝PC98[  pc98.dat ]
  Dsi add nonum 〈1/〉   Dsi add num 〈2〉   Dsi mod 〈3〉   ?2
```

DSIコマンドのオプションを選択する．選択直後から転送が開始される．転送中断には〈ESC〉を押下する．すべての行を転送すると，ビープ音を発し，自動的にホストへブレーク信号を送信して転送を終了する．

ローカルOS機能を終了する．

```
E>〈f・6〉
```

ファイルをセーブし，EDITを終了する．

```
E>end save〈CR〉
```

〈2〉　ホストからパソコンへ

転送元のファイルをEDITモードにする．

```
>>edit m280h.data〈CR〉
```

ローカルOS機能を開始し，

```
E>〈f・6〉
```

ダウンロード・コマンドを投入する．

```
A>dload pc98.dat〈CR〉
```

確認メッセージが次のように出力される．

```
Down-load : HOST ⟹ PC98[ pc98.dat ]   ?〈CR〉
```

〈CR〉を応答すると，直後に転送が開始される（転送中断は〈ESC〉）．"END OF DATA"をセンスすると転送が終了する（転送元のファイルにこの文字列があるとそこで転送が終了してしまうので注意）．

ローカルOS機能を終了し，

A〉〈f•6〉

EDITを終了する．

E〉end〈CR〉

[10] セッションのプリンタ/ディスク出力

機能キー	機能	同一機能キー
〈f•7〉	《セッションのプリンタ出力》の開始/終了．	
〈f•8〉	《セッションのディスク出力》の開始/終了．（TERM98.SAVというファイルに出力する．適当な時に名前を変更，あるいは，別のファイルに複写するのがよい．最初はOUTPUTモードで，2度目からAPPENDモードでオープンする）	
〈SHFT•f8〉	《セッションのディスク出力》の開始/終了．（OUTPUTモードでオープン）	

[11] オプション/イディオム機能

機能キー	機能	同一機能キー
〈f•9〉	《オプション/イディオム機能》の開始/終了．	

《オプション/イディオム機能》

〈ESC〉	本機能を終了する．	〈CR〉
B	ホストからプロンプトとして送られてくるベル音を無視するかどうかのトグルスイッチ．（起動時は無視）	
I	イディオムを出力する．（イディオム・ファイルTERM98.IDMの内容を順次表示して，本機能を終了する．時たま使うやや複雑な定型句をイディオム・ファイルに集積しておくとよい．出力中断は〈ESC〉）	
X	ホストへ〈XON〉を送信する．	

[12] その他の機能

機能キー	機能	同一機能キー
〈f•10〉	ブレーク信号をホストへ送信する．	
〈HELP〉	プリンタを改ページする．	
〈SHFT・HELP〉	ヘルプ・メッセージを表示する．	

6. おわりに

パーソナル・コンピュータを画面編集機能やファイル転送機能をもつTSS端末とするプログラム，称して，TSSフロントプロセッサTERM98TNについて，その基本的構成と諸機能について紹介した．ホスト側からみればパーソナル・コンピュータは単純なダム端末のままであるが，ユーザはパーソナル・コンピュータを画面編集端末として使うことができる．ダム端末の簡便性と自前の（かなり高度な）画面編集機能が利用できることになる．

ホスト側のエディタとしては，送信行の先頭で行番号を指定して編集できる場合に相性がいい．画面に表示したリストを適当に変更して送信すればよいからである．VOS3ではEDITがそれである．一方，HQEDは，使えないことはないが，画面編集機能が十分生かせない．

画面編集機能を実現するには，パーソナル・コンピュータの内部情報が不可欠である．たとえば，VRAMのことがわかれば画面上の文字列を簡単に利用できるし，キーバッファの構成がわかれば文字をそこに戻すことができ編集機能は著しく向上する．この点で，使用した機種は公開情報が多く有利であった．

また，実行速度がそれほど問題にならなければ，BASICインタプリタの使用もこのような自前のプログラムの開発では，その簡便性ゆえに推薦される．プログラムが見にくくなることは否めないが，必要な機能をすぐ追加して実行できるし，問題が起ったらその場で修正して再実行することができるからである．

このようなプログラムでの次の目標は，同様の画面編集機能が使えるAPL端末プログラムと漢字端末プログラムである．APL端末プログラムでは特殊なAPL文字をどう実現するかが，漢字端末プログラムでは2バイトの漢字コードの扱いとホスト側とは異なる漢字処理システム間での変換などが問題となる．

計算機の利用形態は転機を迎えつつある．汎用の大型機に多数の端末を接続し，プログラム開発も，大量の計算の実行も，ワードプロセッシングも何もかもやらせるという方向からの転換である．端末としてパーソナル・コンピュータを利用することが多くなっているのも，パーソナル・コンピュータ側の様々な機能やソフトウェアと統合して，安価なパーソナル・ワークステーション(PWS)としたい要求があるからであろう．このようなPWSの充実と，一方，大量・高速な計算は大型機に任せ，そのためのプログラム開発やグラフィック処理は近年台頭しているエンジニアリング・ワークステーション（EWS）で行なうようになると思われる．いわば，大型機とEWSとPWS（パーソナル・コンピュータ）の3極分化である．そして，これらがネットワークで結合される姿が想像できる．このような時代に，パーソナル・コンピュータは研究者個人の一番身近な道具として重要な機能を果たすであろうし，便利な通信プログラムも鍵となる位置を占めることと思われる．

研究者にとって研究環境を整えることは研究の重要な1ステップであるが，そのための投資が本来の研究を圧迫するようでは本末転倒である．しかし一方，その投資を怠れば長い間には結局本来の研究時間を失うことになる．このささやかなプログラムが，何人かの研究者にとっていくらかでも時間の節約になることを願うばかりである．

仁木直人助教授には，このプログラムの開発動機と，開発途上での貴重な助言をいただいた．種村正美助教授，馬場康維助教授，中央大学の鎌倉稔成講師には，未完成のプログラムを試用して下さり，その機能について有益な意見をいただいた．ここに記して深く感謝いたします．

なお，本プログラムの開発にあたっては，文部省科学研究費特定研究2(59209011)および一般研究C（59530017）の援助を受けました．

```
1000 '*************************************************************************
1010 '*          ===== TERM98TN2: Hitac M-280H via PC-9801E&F =====           *
1020 '*          =====     Concise Version 2.1 (85/05/05)     =====           *
1030 '*          =====         on MS-DOS/N88-BASIC(86)        =====           *
1040 '*          =====    Programmed by T. Nakamura  (ISM)    =====           *
1050 '*************************************************************************
1060  CLEAR  &H100,,,&H1000 : GOSUB*INITIALIZE
1070 '########################################################## main loop ######
1080 *FR.HOST : COLOR C5
1090  IF LOC(1)=0 THEN*TO.HOST
1100 '
1110  CALL CUROFF
1120  WHILE LOC(1)>0 : DUM#=FRE(0)
1130    H$=INPUT$(LOC(1),#1)
1140    FOR I=1 TO LEN(H$)
1150      A$=MID$(H$,I,1)
1160      IF A$=LF$ THEN 1210
1170      IF A$=CR$ THEN  ELSE 1200
1180        PRINT : DUM#=FRE(0) : IF PR THEN LPRINT
1190        GOTO 1210
1200      PRINT A$; : IF PR THEN LPRINT A$;
1210    NEXT I
1220  WEND
1230  GOSUB*GET.LINE : Q=P : R=P : XMAX=POS(0) : YMAX=CSRLIN
1240  IF KBUF>1 THEN GOSUB*SEND.LINES
1250  GOTO*KEYBOARD
1260 '
1270 *TO.HOST
1280  IF KBUF=1 THEN GOSUB*SEND.LINES
1290  IF KBUF>1 AND HOST<>1 THEN GOSUB*SEND.LINES
1300  IF LOC(1)>0 THEN*FR.HOST
1310 '
1320 *KEYBOARD : COLOR C7
1330  GOSUB*KEY.SENSE
1340  IF K$<>"" THEN GOSUB*SCRN.EDIT
1350  GOTO*FR.HOST
1360 '###################################################### subroutines ######
1370 *SEND.LINES : B$=LBUF$(KBUF)
1380  IF MID$(B$,2,1)=">" THEN B$=MID$(B$,3)
1390  IF MID$(B$,1,1)=">" THEN B$=MID$(B$,2)
1400  IF LBUF=1 THEN GOSUB*SEND.HOST ELSE GOSUB*SEND.HOST.ECHO
1410  IF KBUF>=LBUF THEN KBUF=0:LBUF=0 ELSE KBUF=KBUF+1
1420  GOSUB*WAIT10 : RETURN
1430 *SEND.HOST.ECHO : COLOR C4 : PRINT B$
1440 *SEND.HOST      : IF PR THEN LPRINT B$
1450  IF HOST=1 THEN PRINT#1,B$;CR$;
1460  IF HOST=2 THEN GOSUB*LOCAL
1470  RETURN
1480 '=========================================================================
1490 *CLR.LINE : L$=STRING$(MAXL," ") : P=0:Q=0:R=0:S=80 : CHANGE=NO : RETURN
1500 *GET.LINE : GOSUB*CLR.LINE
1510  CALL LINEINFO(YTOP,YBOT,Z,S,P) : CALL GETVRAM(Z,S,L$,R)
1520  RETURN
1530 *PUT.LINE : LOCATE (320+X-P+Q)MOD 80,Y-P¥80
1540  PRINT MID$(L$,Q+1,R-Q); : GOSUB*CHK.YMAX
1550  RETURN
1560 *POS.PUT.LINE     : GOSUB*POS.XY : GOSUB*PUT.LINE               : RETURN
1570 *POS.PUT.LINE.LOC : GOSUB*POS.XY : GOSUB*PUT.LINE : LOCATE X,Y : RETURN
1580 *POS.PUT.STR.LOC  : GOSUB*POS.XY : PRINT KK$;
1590  YY=(X+LEN(KK$))¥80:IF Y+YY>YBOT THEN Y=YBOT-YY
1600  LOCATE X,Y : RETURN
1610 *XCHG.LINE : IF CHANGE THEN*XCHG.1 ELSE*XCHG.2
1620  *XCHG.1 : GOSUB*PUSH.LBUF
1630   COLOR C4 : GOSUB*POS.PUT.LINE.LOC : GOTO*XCHG.E
1640  *XCHG.2 : A=A5 : GOSUB*PUT.ATTR    : GOTO*XCHG.E
1650  *XCHG.E : PRINT K$; : GOSUB*GET.LINE : A=A7 : GOSUB*PUT.ATTR
1660  RETURN
1670 *PUT.ATTR : GOSUB*POS.Z : I1=Z-(P-Q)*2 : N=R-Q : CALL PUTATTR(I1,N,A)
1680  RETURN
1690 *PUSH.LBUF : IF LBUF<MBUF THEN LBUF=LBUF+1:LBUF$(LBUF)=MID$(L$,Q+1,R-Q)
1700  RETURN
1710 '=========================================================================
1720 *KEY.SENSE : K$=INKEY$ : CALL CURON : RETURN
1730 *KEY.WAIT  : K$=INKEY$ : CALL CURON : IF K$="" THEN*KEY.WAIT ELSE RETURN
1740 '=========================================================================
1750 *SCRN.EDIT
1760  IF K$<" " THEN*ED.IT  ELSE IF K$=DL$ THEN*E.DL  ELSE*E.ANK
1770 '-------------------------------------------------------------------------
1780 *ED.IT
1790  IF K$=CR$ THEN*E.CR  ELSE IF K$=BS$ THEN*E.BS  ELSE IF K$=IN$ THEN*E.IN
1800  IF K$=AL$ THEN*E.AL  ELSE IF K$=AR$ THEN*E.AR
1810  IF K$=AU$ THEN*E.AU  ELSE IF K$=AD$ THEN*E.AD
1820  IF K$=CL$ THEN*E.CL
1830  BEEP:RETURN
1840 '--------------------
1850 *E.ANK : IF INS THEN*E.ANK.INS ELSE*E.ANK.NRM
1860 *E.ANK.NRM : IF P>=MAXL THEN BEEP:RETURN
1870  KK$=K$ : MID$(L$,P+1,1)=KK$ : GOTO*E.ANK.E
```

```
1880 *E.ANK.INS : IF R>=MAXL THEN BEEP:RETURN ELSE IF R<P THEN*E.ANK.NRM
1890  KK$=K$+MID$(L$,P+1,R-P) : MID$(L$,P+1,R-P+1)=KK$ : GOSUB*INC.R
1900 *E.ANK.E : GOSUB*INC.P : GOSUB*POS.PUT.STR.LOC : PRINT AR$;
1910  CHANGE=YES : RETURN
1920 '--------------------
1930 *E.CR  : COLOR C4 : GOSUB*POS.PUT.LINE : YCR=CSRLIN
1940  IF YBOT>YMAX THEN LOCATE 0,YMAX+1 : GOTO*E.CR.E
1950  LOCATE 0,YBOT : IF YMAX>=YCR THEN PRINT
1960  *E.CR.E : GOSUB*PUSH.LBUF : KBUF=1 : GOSUB*GET.LINE
1970   GOSUB*INS.OFF : RETURN
1980 '--------------------
1990 *E.BS  : IF P<=0 THEN RETURN
2000  GOSUB*CHK.R : KK$=MID$(L$,P+1,R-P)+" " : MID$(L$,P,R-P+1)=KK$
2010  GOSUB*DEC.P : PRINT AL$; : GOSUB*POS.PUT.STR.LOC
2020  R=R-1 : CHANGE=YES : RETURN
2030 '--------------------
2040 *E.DL  : IF P>=R THEN RETURN
2050  GOSUB*CHK.Q : KK$=MID$(L$,P+2,R-P-1)+" " : MID$(L$,P+1,R-P)=KK$
2060  GOSUB*POS.PUT.STR.LOC
2070  R=R-1 : CHANGE=YES : RETURN
2080 '--------------------
2090 *E.IN  : GOSUB*INS.XCHG : RETURN
2100 '--------------------
2110 *E.AL  : IF P>0 THEN P=P-1 : PRINT AL$; : RETURN  ELSE RETURN
2120 '--------------------
2130 *E.AR  : IF P<S THEN P=P+1 : PRINT AR$; : RETURN  ELSE RETURN
2140 '--------------------
2150 *E.AU  : IF P>=80 THEN P=P-80 : PRINT AU$; : RETURN
2160  IF CSRLIN>YTOP THEN GOSUB*XCHG.LINE : RETURN  ELSE RETURN
2170 '--------------------
2180 *E.AD  : IF P+80<=S THEN P=P+80 : PRINT AD$; : RETURN
2190  IF CSRLIN<YBOT THEN GOSUB*XCHG.LINE : RETURN  ELSE RETURN
2200 '--------------------
2210 *E.CL  : COLOR C4 : CLS 1 : LBUF=0 : GOSUB*CLR.LINE : YMAX=0 : RETURN
2220 '=============================================================================
2230 *INC.P : GOSUB*CHK.Q : P=P+1 : GOSUB*CHK.R : RETURN
2240 *DEC.P : GOSUB*CHK.R : P=P-1 : GOSUB*CHK.Q : RETURN
2250 *CHK.R : IF P>R THEN R=P:GOSUB*CHK.R:RETURN
2260 *CHK.Q : IF P<Q THEN Q=P:RETURN ELSE RETURN
2270 *INC.R : R=R+1 : GOSUB*CHK.S : RETURN
2280 *CHK.S : IF R>S THEN S=S+80 : IF S>MAXL THEN S=MAXL:RETURN ELSE RETURN
2290 '--------------------
2300 *CHK.YMAX : IF YMAX<CSRLIN THEN YMAX=CSRLIN:RETURN ELSE RETURN
2310 *POS.XY  : X=POS(0) : Y=CSRLIN : RETURN
2320 *POS.XYZ : X=POS(0) : Y=CSRLIN : Z=2*X+160*Y : RETURN
2330 *POS.Z   : Z=2*POS(0)+160*CSRLIN : RETURN
2340 '--------------------
2350 *INS.XCHG : IF INS THEN*INS.OFF ELSE*INS.ON
2360 *INS.ON   : INS=YES : CALL INSON  : RETURN
2370 *INS.OFF  : INS=NO  : CALL INSOFF : RETURN
2380 '--------------------
2390 *MODE.DISP.ON.KEY  : GOSUB*MODE.DISP.ON : KEY OFF : RETURN
2400 *MODE.DISP.ON  : X$=STRING$(LEN(MODE1$)," ") : GOSUB*MODE.DISP
2410  MODE2$=MODE1$ : MODE1$=MODE$ : X$=MODE$ : GOSUB*MODE.DISP : RETURN
2420 *MODE.DISP.OFF.KEY : GOSUB*MODE.DISP.OFF : GOSUB*FKEY.ALLOC : RETURN
2430 *MODE.DISP.OFF : X$=STRING$(LEN(MODE$)," ") : GOSUB*MODE.DISP
2440  MODE$=MODE1$ : MODE1$=MODE2$ : X$=MODE1$ : GOSUB*MODE.DISP : RETURN
2450 *MODE.DISP
2460  C=VAL(LEFT$(X$,1))
2470  FOR I=2 TO LEN(X$)
2480    PUT (8*(78-LEN(X$)+I),0),KANJI(ASC(MID$(X$,I,1))),PSET,0,C
2490  NEXT I : RETURN
2500 '--------------------
2510 *WAIT0 : FOR WT=1 TO NW : FOR W=1 TO 100 : NEXT : NEXT : RETURN
2520 *WAIT1 : NW= 1 : GOTO*WAIT0
2530 *WAIT2 : NW= 2 : GOTO*WAIT0
2540 *WAIT5 : NW= 5 : GOTO*WAIT0
2550 *WAIT10: NW=10 : GOTO*WAIT0
2560 *WAIT30: NW=30 : GOTO*WAIT0
2570 '################################################ Fkey interrupt traps ######
2580 *INT.LOGON  : GOSUB*SUB.LOGON  : RETURN
2590 *INT.SECRET : RETURN
2600 *INT.TEKTRO : RETURN
2610 *INT.CRXLF  : RETURN
2620 *INT.HISTRY : RETURN
2630 *INT.LOCAL  : GOSUB*SUB.LOCAL  : RETURN
2640 *INT.PRINT  : PR=NOT PR : GOSUB*FKEY.PRNT : RETURN
2650 *INT.SAVE   : RETURN
2660 *INT.OPTIDM : RETURN
2670 *INT.BREAK  : GOSUB*SEND.BREAK : RETURN
2680 '############################################################ local mode ######
2690 *SUB.LOCAL : IF HOST<>2 THEN*LOCAL.START  ELSE*LOCAL.QUIT
2700 *LOCAL.START : HOST=2 : MODE$="6 LOCAL " : GOSUB*MODE.DISP.ON.KEY
2710  ON KEY GOSUB ,,,,,,,,*LOCAL.BREAK : KEY ON : GOSUB*LOCAL.EXIT : RETURN
2720 *LOCAL.QUIT  : HOST=1 : GOSUB*MODE.DISP.OFF.KEY
2730  PRINT:PRINT : GOSUB*CHK.YMAX : LBUF=0 : GOSUB*CLR.LINE : RETURN
2740 *LOCAL.BREAK : GOSUB*SEND.BREAK : GOTO*LOCAL.EXIT
2750 *LOCAL.ERR   : COLOR C2 : PRINT USING"error ## in ####:":ERR,ERL:
```

```
 2760  IF ERR=53 THEN PRINT"File not found"
 2770  RESUME*LOCAL.EXIT
 2780 *LOCAL.EXIT  : COLOR C6 : PRINT:PRINT DR$+">"; : XMAX=0:YMAX=CSRLIN
 2790  ON ERROR GOTO 0 : GOSUB*GET.LINE : Q=P : R=P : CALL CURON : RETURN
 2800 *LOCAL.INIT  : DR$="A" : RETURN
 2810 '=============================================================================
 2820 *LOCAL : ON ERROR GOTO*LOCAL.ERR
 2830  BB$=B$ : IF B$="" THEN*LOCAL.EXIT
 2840  GOSUB*GETTOK : D$=LEFT$(TOK$,3) : CALL TOUPPER(D$)
 2850  IF D$="A:" THEN DR$=LEFT$(D$,1) : GOTO*LOCAL.CHILD
 2860  IF D$="B:" THEN DR$=LEFT$(D$,1) : GOTO*LOCAL.CHILD
 2870  IF D$="C:" THEN DR$=LEFT$(D$,1) : GOTO*LOCAL.CHILD
 2880  IF D$="D:" THEN DR$=LEFT$(D$,1) : GOTO*LOCAL.CHILD
 2890  IF D$="E:" THEN DR$=LEFT$(D$,1) : GOTO*LOCAL.CHILD
 2900  IF D$="FIL" THEN*LOCAL.FILES
 2910  IF D$="DOS" THEN*LOCAL.DOS
 2920  IF D$="DLO" THEN*LOCAL.DLOAD
 2930  IF D$="ULO" THEN*LOCAL.ULOAD
 2940  IF D$="QUI" THEN*LOCAL.QUIT
 2950 *LOCAL.CHILD : CHILD BB$,1 : GOSUB*INT.RESET : GOTO*LOCAL.EXIT
 2960 '-----------------------------------------------------------------------------
 2970 *LOCAL.DOS   : CHILD    ,1 : GOSUB*INT.RESET : GOTO*LOCAL.EXIT
 2980 '-----------------------------------------------------------------------------
 2990 *LOCAL.FILES : GOSUB*SKPSP : COLOR C7 : FILES B$ : GOTO*LOCAL.EXIT
 3000 '-----------------------------------------------------------------------------
 3010 *LOCAL.ULOAD : COLOR C7
 3020  GOSUB*GET.F1 : IF F1$="" THEN INPUT"Up-Loading file name";F1$
 3030  OPEN F1$ FOR INPUT AS #2
 3040  COLOR C6 : PRINT : PRINT"Up-load:   HOST <== PC-98[ ":F1$;" ]" : COLOR C7
 3050  PRINT"   Dsi add nonum <1/>     Dsi add num <2>     Dsi mod <3>    ?";
 3060 *LOCAL.ULOAD.WAIT : GOSUB*KEY.WAIT
 3070  IF K$="1" THEN B$="dsi add nonum" : GOTO*LOCAL.ULOAD.GO
 3080  IF K$=CR$ THEN B$="dsi add nonum" : GOTO*LOCAL.ULOAD.GO
 3090  IF K$="2" THEN B$="dsi add num"   : GOTO*LOCAL.ULOAD.GO
 3100  IF K$="3" THEN B$="dsi mod"       : GOTO*LOCAL.ULOAD.GO
 3110  IF K$=EC$ THEN*LOCAL.ULOAD.EXIT ELSE*LOCAL.ULOAD.WAIT
 3120 *LOCAL.ULOAD.GO : COLOR C7 : CALL CUROFF
 3130  PRINT : GOSUB*LOCAL.SEND.HOST : GOSUB*WAIT10
 3140  WHILE NOT EOF(2)
 3150    IF LOC(1)=1 THEN H$=INPUT$(LOC(1),#1)
 3160    IF LOC(1)>1 THEN H$=INPUT$(LOC(1),#1):COLOR C5:PRINT H$;:GOTO 3160
 3170    LINE INPUT#2,L$ : COLOR C7 : PRINT L$ : PRINT#1,L$:CR$:
 3180    IF INKEY$=EC$ THEN*LOCAL.ULOAD.BREAK ELSE DUM#=FRE(0):GOSUB*WAIT2
 3190  WEND : BEEP : PRINT"END OF DATA"
 3200 *LOCAL.ULOAD.BREAK : GOSUB*WAIT30 : GOSUB*SEND.BREAK
 3210 *LOCAL.ULOAD.EXIT  : CLOSE#2 : GOTO*LOCAL.EXIT
 3220 '-----------------------------------------------------------------------------
 3230 *LOCAL.DLOAD : COLOR C7 : CALL CUROFF
 3240  GOSUB*GET.F1 : IF F1$="" THEN INPUT"Down-loading file name":F1$
 3250  OPEN F1$ FOR OUTPUT AS #2
 3260  COLOR C6 : PRINT : PRINT"Down-load:   HOST ==> PC-98[ ":F1$:" ]":
 3270  COLOR C7 : PRINT"    ?";
 3280 *LOCAL.DLOAD.WAIT : GOSUB*KEY.WAIT
 3290  IF K$=CR$ THEN*LOCAL.DLOAD.GO
 3300  IF K$=EC$ THEN*LOCAL.DLOAD.EXIT ELSE*LOCAL.DLOAD.WAIT
 3310 *LOCAL.DLOAD.GO : COLOR C7 : CALL CUROFF : B$="dso"
 3320  PRINT : GOSUB*LOCAL.SEND.HOST : GOSUB*WAIT10
 3330 *LOCAL.DLOAD.LOOP : DUM#=FRE(0)
 3340    LINE INPUT#1,H$
 3350    IF  LEFT$(H$,1)=LF$ THEN H$=MID$(H$,2)
 3360    IF RIGHT$(H$,1)=LF$ THEN H$=LEFT$(H$,LEN(H$)-1)
 3370    IF INSTR(H$,"END OF DATA")>0 THEN BEEP:GOTO*LOCAL.DLOAD.BREAK
 3380    COLOR C7 : PRINT H$ : PRINT#2,H$
 3390    IF INKEY$=EC$ THEN GOSUB*WAIT5:GOSUB*SEND.BREAK:GOTO*LOCAL.DLOAD.BREAK
 3400  GOTO*LOCAL.DLOAD.LOOP
 3410 *LOCAL.DLOAD.BREAK : PRINT H$ : GOSUB*WAIT2
 3420 *LOCAL.DLOAD.EXIT  : CLOSE#2 : CLOSE#1:GOSUB*OPEN.M280H : GOTO*LOCAL.EXIT
 3430 '=============================================================================
 3440 *LOCAL.SEND.HOST : HOST=1 : GOSUB*SEND.HOST.ECHO : HOST=2 : RETURN
 3450 '=============================================================================
 3460 *INT.RESET : ON ERROR GOTO*LOCAL.ERR
 3470  ON STOP GOSUB*QUIT : STOP ON
 3480  GOSUB*OPEN.M280H : GOSUB*FKEY.ALLOC : RETURN
 3490 '=============================================================================
 3500 *GET.F1 : GOSUB*SKPSP : F1$=B$ : RETURN
 3510 '=============================================================================
 3520 *GETTOK : GOSUB*SKPSP : IF B$="" THEN TOK$=B$ : RETURN
 3530  I=INSTR(B$," ") : IF I<=0 THEN TOK$=B$ : B$="" : RETURN
 3540  TOK$=LEFT$(B$,I-1) : B$=MID$(B$,I+1) : RETURN
 3550 '
 3560 *SKPSP : WHILE LEFT$(B$,1)=" " : B$=MID$(B$,2) : WEND : RETURN
 3570 '################################################## logon/logoff #####
 3580 *SUB.LOGON : MODE$="6 LOGON/OFF " : GOSUB*MODE.DISP.ON.KEY
 3590  KEY 1,"1" : KEY 2,"2" : KEY 3,"3" : KEY 4,"4" : KEY 5,"5"
 3600  KEY 6,"6" : KEY 7,"7" : KEY 8,"8" : KEY 9,"9" : KEY 10,"0"
 3610
 3620  COLOR C7 : GOSUB*CLR.LINE
 3630  PRINT : FOR I=0 TO NU
```

```
3640    PRINT USING"   ##) ";I; : PRINT UID$(I);
3650    IF (I MOD 5)=0 THEN PRINT:PRINT
3660  NEXT I
3670  LOCATE 0,CSRLIN : PRINT"?"; : GOSUB*KEY.WAIT
3680  IF K$=EC$ THEN PRINT                    : GOTO*SUB.LOGON.EXIT
3690  IF K$=CR$ OR K$="0" THEN B$="logoff"    : GOTO*SUB.LOGON.SEND
3700  B$="logon " : K=VAL(K$) : IF K>NU THEN   GOTO*SUB.LOGON.SEND
3710  IF K<=0 THEN PRINT                      : GOTO*SUB.LOGON.EXIT
3720   B$=B$+UID$(K) : COLOR C4 : PRINT:PRINT : PRINT B$
3730   IF PSWD$(K)<>"" THEN B$=B$+PSWD$(K)
3740   PRINT#1,B$;CR$;                        : GOTO*SUB.LOGON.EXIT
3750  '
3760 *SUB.LOGON.SEND : PRINT:PRINT : GOSUB*SEND.HOST.ECHO
3770 *SUB.LOGON.EXIT : GOSUB*MODE.DISP.OFF.KEY : RETURN
3780 '-------------------------------------------------------------------------
3790 *INSTL.UID : ERASE UID$,PSWD$ : DIM UID$(10),PSWD$(10)
3800  ON ERROR GOTO*INSTL.UID.ERR
3810  NU=0 : UID$(0)="logoff" : OPEN "logon.uid" FOR INPUT AS #2
3820  WHILE NOT EOF(2) AND NU<10 : NU=NU+1 : LINE INPUT#2,UID$(NU)
3830    I=INSTR(UID$(NU),"/")
3840    IF I>0 THEN PSWD$(NU)=MID$(UID$(NU),I) : UID$(NU)=LEFT$(UID$(NU),I-1)
3850  WEND
3860  CLOSE#2 : NU=NU+1 : UID$(NU)="others" : GOTO*INSTL.UID.EXIT
3870 *INSTL.UID.ERR  : RESUME*INSTL.UID.EXIT
3880 *INSTL.UID.EXIT : CLOSE#2 : ON ERROR GOTO 0 : RETURN
3890 '##################################################### break routine #####
3900 *SEND.BREAK : COLOR C6 : PRINT" break !!" : COLOR C5
3910  OUT &H32,&H3F : GOSUB*WAIT10 : OUT &H32,&H37:RETURN
3920 '################################################### initialization #####
3930 *INITIALIZE : DEFINT A-Z : CONSOLE 0,24,1,1 : WIDTH 80,25 : SCREEN 3
3940       ON STOP GOSUB*QUIT : STOP ON
3950  A0=&H1  :A1=&H21 :A2=&H41 :A3=&H61 :A4=&H81 :A5=&HA1 :A6=&HC1 :A7=&HE1
3960  C0=0    :C1=1    :C2=2    :C3=3    :C4=4    :C5=5    :C6=6    :C7=7
3970  BS$=CHR$(&H8)  : LF$=CHR$(&HA)  : CR$=CHR$(&HD)  : DL$=CHR$(&H7F)
3980  IN$=CHR$(&H12) : CL$=CHR$(&HC)  : EC$=CHR$(&H1B)
3990  AR$=CHR$(&H1C) : AL$=CHR$(&H1D) : AU$=CHR$(&H1E) : AD$=CHR$(&H1F)
4000  NO=(1<>1) : YES=NOT NO
4010  '
4020  INS=NO : CHANGE=NO
4030  MBUF=20 : DIM LBUF$(MBUF) : KBUF=0 : LBUF=0
4040  MAXL=254 : GOSUB*CLR.LINE
4050  '
4060  YTOP=0 : YBOT=23 : CONSOLE YTOP,YBOT-YTOP+1,1,1
4070  '
4080  MACHINE#=SEGPTR(2) : DEF SEG=MACHINE# : GOSUB*INSTL.MCODE
4090  PR=NO : SV=NO : GOSUB*FKEY.ALLOC
4100  '
4110  DIM UID$(1),PSWD$(1) : GOSUB*INSTL.UID
4120  '
4130  GOSUB*LOCAL.INIT
4140  GOSUB*OPEN.M280H : PRINT#1,CR$ : HOST=1
4150  '
4160  COLOR C4 : CLS : COLOR C6 : PRINT"M-280H TSS starts !!"
4170  '
4180  RETURN
4190 '=========================================================================
4200 *OPEN.M280H : M280H$="COM:E71XS" : OPEN M280H$ AS #1 : RETURN
4210 '=========================================================================
4220 *INSTL.MCODE : DEF SEG=MACHINE#
4230  RESTORE*MCODE.DATA : A$="" : I=0
4240  WHILE NOT A$="#" : READ A$ : POKE I,VAL("&H"+A$) : I=I+1 : WEND
4250  CURON=&H0 : CUROFF=&H20 : INSON=&H30 : INSOFF=&H50
4260  LINEINFO=&H70 : GETVRAM=&H120 : PUTATTR=&H1A0 : TOUPPER=&H1D0
4270  RETURN
4280 '---------------------
4290 *MCODE.DATA
4300 '.....curon.........
4310  DATA 36,8B,16,38,04,36,8B,0E,36,04,50,B8,A0,00,F6,E1
4320  DATA D1,E2,03,D0,58,B4,13,CD,18,B4,11,CD,18,CF,90,90
4330 '.....curoff........
4340  DATA B4,12,CD,18,CF,90,90,90,90,90,90,90,90,90,90,90
4350 '.....inson.........
4360  DATA B0,4B,E6,62,B0,8F,E6,60,B0,8E,E6,60,B0,7A,E6,60
4370  DATA CF,90,90,90,90,90,90,90,90,90,90,90,90,90,90,90
4380 '.....insoff........
4390  DATA B0,4B,E6,62,B0,8F,E6,60,B0,80,E6,60,B0,7A,E6,60
4400  DATA CF,90,90,90,90,90,90,90,90,90,90,90,90,90,90,90
4410 '.....lineinfo......
4420  DATA 51,52,06,57,56,1E,53,C4,7F,0C,26,8B,0D,D1,E1,D1
4430  DATA E1,81,C1,16,03,C4,7F,10,26,8B,05,D1,E0,D1,E0,05
4440  DATA 16,03,36,8B,16,36,04,D1,E2,D1,E2,81,C2,16,03,50
4450  DATA 8B,D8,3B,D9,73,22,8B,07,80,FC,80,72,11,83,C3,04
4460  DATA 3B,D9,73,14,8B,07,80,FC,80,72,03,83,C3,04,3B,D3
4470  DATA 76,06,58,05,04,00,EB,D7,59,81,E9,16,03,D1,E9,D1
4480  DATA E9,81,EA,16,03,D1,EA,D1,EA,2A,D1,B8,50,00,F6,E2
4490  DATA 8B,16,38,04,03,C2,5E,1F,C4,3C,26,89,05,81,EB,16
4500  DATA 03,D1,EB,D1,EB,2A,D9,FE,C3,B8,50,00,F6,E3,C4,7C
4510  DATA 04,26,89,05,B8,A0,00,F6,E1,C4,7C,08,26,89,05,5E
```

```
4520  DATA 5F,07,5A,59,CF,90,90,90,90,90,90,90,90,90,90,90
4530 '.....getvram........
4540  DATA 56,57,1E,06,53,51,52,C4,3F,06,57,C4,7F,08,26,8B
4550  DATA 15,83,FA,00,7E,4F,52,C4,7F,04,52,26,8B,0D,26,8B
4560  DATA 7D,02,80,FD,00,74,14,32,ED,51,36,8B,16,A4,06,B1
4570  DATA 04,D3,EA,8C,D1,03,D1,59,EB,06,90,36,8B,16,10,14
4580  DATA 8E,C2,5A,C5,77,0C,8B,34,B8,00,A0,8E,D8,58,3B,C1
4590  DATA 7D,02,8B,C8,FC,33,D2,FE,C6,AD,3C,20,74,02,8A,D6
4600  DATA AA,E2,F4,32,F6,5F,07,26,89,15,5A,59,5B,07,1F,5F
4610  DATA 5E,CF,90,90,90,90,90,90,90,90,90,90,90,90,90,90
4620 '.....putattr........
4630  DATA 56,57,1E,06,51,52,C4,7F,04,26,8B,0D,83,F9,00,7E
4640  DATA 15,C4,3F,26,8B,05,50,C4,7F,08,26,8B,3D,B8,00,A2
4650  DATA 8E,C0,58,FC,F3,AB,5A,59,07,1F,5F,5E,CF,90,90,90
4660 '.....toupper........
4670  DATA C4,3F,52,26,8B,0D,26,8B,75,02,80,FD,00,74,14,32
4680  DATA ED,51,36,8B,16,A4,06,B1,04,D3,EA,8C,D1,03,D1,59
4690  DATA EB,06,90,36,8B,16,10,14,8E,DA,5A,80,F9,00,7E,10
4700  DATA 8A,04,3C,61,72,07,3C,7B,73,03,80,24,DF,46,E2,F0
4710  DATA CF,90,90,90,90,90,90,90,90,90,90,90,90,90,90,90
4720 '
4730  DATA "#"
4740 '=============================================================================
4750 *FKEY.CLR
4760  KEY 1,"" : KEY 2,"" : KEY 3,"" : KEY 4,"" : KEY 5,""
4770  KEY 6,"" : KEY 7,"" : KEY 8,"" : KEY 9,"" : KEY 10,""
4780  RETURN
4790 '-----------------------------------------------------------------------------
4800 *FKEY.ALLOC
4810  KEY 1,"logon"  : KEY 2,"secret" : KEY 3,"TEKTRO"
4820  KEY 4,"¥n="+CR$: KEY 5,"Histry" : KEY 6,"LOCAL"
4830  GOSUB*FKEY.PRNT : GOSUB*FKEY.SAVE
4840  KEY 9,"op/idm" : KEY 10,"break!"
4850  ON KEY GOSUB *INT.LOGON,*INT.SECRET,*INT.TEKTRO,*INT.CRXLF,*INT.HISTRY
4860  ON KEY GOSUB ,,,,,*INT.LOCAL,*INT.PRINT,*INT.SAVE,*INT.OPTIDM
4870  ON KEY GOSUB ,,,,,,,,,*INT.BREAK
4880  KEY ON : RETURN
4890 '--------------------
4900 *FKEY.PRNT : IF PR THEN KEY 7,"pr ON":RETURN  ELSE KEY 7,"pr off":RETURN
4910 *FKEY.SAVE : IF PR THEN KEY 8,"sv ON":RETURN  ELSE KEY 8,"sv off":RETURN
4920 '################################################### quit routine ######
4930 *QUIT : COLOR C4 : CLOSE : KEY OFF
4940  KEY  1,"system"
4950  KEY  2,"edit*fr.host"+CHR$(13)
4960  KEY  3,"llist "
4970  KEY  4,"list "
4980  KEY  5,"run"+CHR$(13)
4990  KEY  8,"renum 1000"
5000  KEY  9,"edit ."+CHR$(13)
5010  KEY 10,"save"+CHR$(&H22)+"term98V2
5020  STOP
```

図3. 端末プログラム（簡略版）—TERM98TN2—
Figure 3. Concise Version of TSS Front Processor, TERM98TN2

参 考 文 献

赤繁直樹（1985）. C言語によるターミナル・プログラム, 日経バイト, 第**7**号, 217-229.

浅野・壁谷・金磯・桑野（1983）. PC-9800システム解析（上）, アスキー出版局.

藤田英時・幸田敏記（1984）. PC-Techknow 9800, システムソフト.

市川伸一（1983）. TSS端末としてのパーソナル・コンピュータ, 行動計量学, **11**巻, 1号, 23-32.

石田晴久（1984）. マイクロコンピュータのプログラミング, 岩波書店.

NEC（1983a）. PC-9801Fユーザーズ・マニュアル, 日本電気.

NEC（1983b）. PC-9801シリーズMS-DOS2.0ユーザーズマニュアル, 日本電気.

NEC（1984a）. N88-日本語BASIC（86）（MS-DOS版）リファレンス・マニュアル, 日本電気.

NEC（1984b）. N88-日本語BASIC（86）（MS-DOS版）ユーザーズ・マニュアル, 日本電気.

仁木直人（1986）. パーソナル・コンピュータ（FM-7/8）を用いた画面編集ターミナル, 統計数理, **33**巻, 2号, 199-220.

柴山 守(1984). TTYフルスクリーン端末とファイル転送—PC9801F, IBM5550を使用して—, 京都大学大型計算機センター広報, Vol. **17**, No. 4, 240-249.

Screen-Edit TSS Front Processor Using Personal Computer (NEC/PC-9801 Series)

Takashi Nakamura

(The Institute of Statistical Mathematics)

A personal computer with common operating system and a hard-disk unit provides a useful research environment for scientists and statisticians. Communication with a host computer holds the key to this environment, where a terminal control program such as the screen-edit TSS front processor introduced here can play an important role.

This processor runs under the MS-DOS version of the N88-BASIC (86) interpreter for the NEC/PC-9801 personal computer series. Users can operate a personal computer as a quasi-full-screen-edit terminal with a variety of editing functions and TEKTRONIX 4010 graphics. Handling the files of the personal computer can be executed in the local OS mode of this program by issuing MS-DOS commands which are supported by the child process control of the BASIC. Functions of up-loading and down-loading files to/from the host computer are also available in the local OS mode.

The paper describes how to develop a terminal control program following three versions of the program : a simple one, a prototype, and a concise version. It also contains the user's manual of the final version of the TSS front processor.