Cで書いたPC9801用 TSSターミナルプログラム

大阪府立大学総合科学部　柳　瀬　　章

最近大阪大学の大型計算機センターと府立大学計算センターがネットワークでつながれて、スーパーコンピューターの利用が便利になり、筆者の計算量も増加している。この利用のさい、端末としてNECのPC9801F2に畠中プログラム[1]をMS－DOS版にして使用してきた。このプログラムは大変よくできたプログラムで、1200ボーで問題なく動作し、機能的にも普通の利用には十分である。

しかし少し長く端末で会話するときに、数分・数十分前の会話の記録を見たくなる。会話をプリンターに印字しておいて、それを引き戻して読めばよいのだが100行をこえると、捜すのにけっこう時間をとられてしまう。さらにこのようにプリンターを使用すると、紙とリボンがどんどん消耗する。これをなんとか効率的にできないかというのが、ここで紹介するプログラムを作った動機である。この頃価格がさがったメモリーを、筆者のPCもつけているので640KBの全部を有効に使うことができれば、このようなことは可能になるはずであると考え、習い始めたプログラム言語Cで実現してみた。使用したCのコンパイラーはLattice CのVer.J3.1[2]で、他にC-TOOL98[3]の関数も使用している。MSDOSはVer.3.1である。実際に使用テストをしたのは、PC9801F2の他VM2とVM4である。ご希望の方にはソースでも、コンパイル・リンクしたものでも提供できる。

§1　機能

機能の主なものを箇条書きにして示すと、

1. 8000行か250KBのどちらか少ないほうを限度とするスクロールアップ・ダウン
2. 上記の範囲での文字列の探索機能
3. 交信記録のディスクセーブ
4. ACOSとPC間のファイル送受信
5. 交信中にMSDOSコマンドを呼びだす等のユティリティー

である。スクロールを高速にするため、漢字と片仮名をあつかう機能は持たせていない。

§2　プログラムの構成

プログラムは次の6個のソースファイルにわかれている。

1. TSSX.C　main( )を含むもので交信記録を記憶するメモリーの確保、交信制御、交信記録

の集積を行う機能を含んでいる。

2. TSSE.C edit( )を窓口にして機能1.と 2.を受け持つ。
3. TSSF.C 機能3.と 4.の PC9801のディスクファイルへの入出力を受け持つ。
4. TSSS.C 機能5.を受け持つと共に、特殊キーの設定を行う。
5. TSSM.C RS 232C、プリンターの入出力のBIOSコールおよびキーボードからの入力、CRTへの出力を行うMSDOSコールの関数を含んでいる。
6. TSSL.H #include文と#define文によるマクロ定義を含むヘッダーファイル。

§3 使用法

このプログラムは通常の交信モード(A)、編集探索モード(B)、ファイルモード(C)の、三つの主な動作モードの他、MSDOSのコマンドモード(D)を持っている。それぞれの使用法を以下に説明する。

3—A 通常の交信モード

このプログラムが最初動作状態に入った時にはこのモードになっている。大文字・小文字・数字・記号字の送受信をキーボード入力、CRT出力で行うことができる。交信記録はキャリッジリターン(CR)の受信かそのキーボード入力を区切りにして、可変長のレコードにしてメモリーに記憶される。このとき空行は行としない操作をいれている。このため画面から空行が除かれるので通常の端末と少し違う印象を利用者にあたえる。レコードの行数はCRTの右下に数字で表示される。

このモードでの特殊キーの機能を次に示す。

f・1 $$$LOGON,TSS,,ASCを送信してホストとの交信を開始する。
f・2 プリンターのON/OFF。画面右上に"プリンター出力中"の文字が点滅する。
f・3 ACOSからPCのファイル転送の開始
f・4 PCからACOSのファイル転送の開始
f・5 無効。(B)で使われる。
f・6 MSDOSコマンドモードへ
f・7 交信記録のPCディスクへのSAVE。PCのファイル名をたずねてくるので、答えるとSAVEを開始する。このとき右下の数字が変化してSAVEの進行を知らせてくれる。
f・8 f・7の逆でPCのファイルの内容を交信記録のメモリーにLOADする。この機能で中断した交信の続きをすることができる。
f・9 無効。このキーはぜんぜん使われない。

f・1も以下に述べるESCキーの機能でまかなえるので将来の機能拡張用に二つ空いていることになる。

f・10　プログラムの終了

↑　編集探索モードへ

BSと←　直前に入力した1字を削除

STOP　BREAK信号の送信

CTRLB　無効

CTRLC　STOPキーと同じ

CTRLD　ホストとの結合の強制切断。回線の故障等のときの非常脱出用。

CTRLE　プログラムの終了。f・10と全く同じ。

他のCTRLコードはそのままACOSに送信される。

ESC　図1の例のようにCの関数escq( )を作ることで、escシークエンスを定義することができる。図の******でつぶしてあるところに課題番号とPASS WORDをいれコンパイルしたプログラムではESC 1 (31)$_H$と連続した入力をすると、ACOSにはそれが送られるが端末の画面には"府立大学センター"と表示される。

図1の後半のようにsclput( )を使うと、指定した文字列が、画面とACOSに送られると共にメモリーにも記録される。

他のキー　無効。

```
void escq()     /* ESC sequence を扱う */
{
int e;
for(;;)
        if(chit()){
           e=cget();
           switch(e){
                case 0x31: rslput("********$******¥r");
                           printf("府立大学センター¥n"); break;
                case 0x43:
                case 0x63: sclput("CUTV;* B DV;* PASTE BV¥r"); break;
                case 0x4c:
                case 0x6c: sclput("LIST ALL¥r"); break;
                case 0x4f:
                case 0x6f: sclput("OSAKA¥r"); break;
                case 0x52:
                case 0x72: sclput("REMO CLEARFILES¥r"); break;
                default: break;
                    }
                return;
                  }
}
```

図1　ESCキーの機能のためのescq( )

3－B　編集探索モード

交信モードから↑キーでこのモードに入る。このモードでは、矢印キーでカーソルを移動させることができる。↑でカーソルを上に移動させるとき画面が変わることがある。これはSTOPキーでBREAKしたときに ^Cが画面に残っている時などにおきる。これは↑や↓でカーソルが上下に移動するときには、メモリーの交信記録を取り出して、画面に写す操作をしているためである。この操作で最初に述べたような広い範囲のスクロールを可能にしている。カーソルが画面の上下の端にいるときには↑・↓で画面のスクロールがおきる。

ｆ・5をこのモードで押すと”捜す文字列？”の文字が画面の上端にあらわれる。そこで79文字以下の文字列を入力してCRキーを押すと、カーソルがあった行から一つ上の行からはじめて、上方に向かって探索が開始される。もしその文字列が見付かると、それを含む行を画面の最上部にした画面が現れる。ないときには”文字列＊＊＊＊はありません”と画面の最上部に書いてカーソルをもとの位置にもどす。

以上の探索機能の他に、このモードは編集機能を持っている。カーソルのある位置にキーボードから文字を入れることができる。INSキーは、このとき置換モードと挿入モードを入れ替えるトグルスィッチになっている。現在のモードは、画面右上に緑の文字で示される。またこのモードでは、DELキーはカーソルがある位置の文字を消す機能を与えられている。BSキーは←キーと同じで、カーソルを左に移動させるだけである。画面上の任意の行で編集を行ったのちCRキーを押すと、変更された行の内容がACOS側に送信されると共に、画面はもとの交信モードの画面にもどっている。ACOSへの送信が不必要なときにはSHIFTキーを押しながらCRキーを押せばもとの状態に復帰する。カーソルがそれほど上に移動していなければ、↓でカーソルを最後の行に移しても送受信モードになる。

画面でおこなった変更はカーソルがその行にあるときだけ有効である。カーソルを上下に移動させた後、変更をおこなった行にカーソルを戻すと変更前の内容が復元する。

3－C　ファイル転送

PCのファイルにACOSのファイルから転送するためには、送受信モードの会話で目的のファイルをカレントファイルにする。次にｆ・3キーを押すと”PC receive FILE”ときいてくるので受けいれるファイルを指定する。この段階でもとに戻りたければ、”,,”のような、MSDOSのファイル名に許されないものを入力する。まともなファイル名が入力されると右上の緑の表示が、”ファイル転入準備”から”ファイル転入中”に変わり受入が始まる。受入中の中断にSTOPキーを用いることができる。ACOSのTSSは送信の終わりを知らせてくれないので、ちよっとした工夫が必要である。また、STOPキーを生かすのにもMSDOSの動作に合わせた工夫がいる。これらについては次の節でプログラムのこの部分を示して説明する。

PCからACOSへのファイル転送には二通りの方法が用意されている。第1はFORTRAN SYSTEMのREADコマンドを使用する。送受信モードでFRT77のSYSTEMに入っておいて、f・4キーを押すと"PC send FILE"ときいてくるので、PC側のファイル名を入力する。つきに"ACOS File-name"ときいてきたらACOS側のファイル名を入力する。ここで右上の表示が"ファイル転出(a)準備"から"ファイル転出(a)中"に変わり転送が始まる。この場合もSTOPキーは転送の中断に使用することができる。転送が終わればPCのファイルをCLOSEしたことを知らせてくれたのち、端末は送受信モードになる。送信の最後には$$$EOFを送っているのでACOSのTSSはREADコマンドの終了処理に入り終わるとFRT77のプロンプト*を返してくる。この間1分程度待つ必要がある。ところが時によるとまてどくらせど返事がないという状態がおきることがある。そのときには$$$EOFをあらためて入力しなければならない。そして送られたファイルの最後を覗いて見ると"$$EOF"となっている。つまりどこかで$が1字消えたのが原因である。これは10%程の確率でおきる現象である。罪はACOSの方にあると思っているが本当の理由は判らない。

転送の第2の方法は次のように使用する。送受信モードでACOS TSSのEDIT SYSTEMに入り"NEW"コマンドを送ってTEXTの受入待ちの状態にする。SHIFTキーを押しながらf・4キーを押す。"PC send FILE"ときいてくるのでPC側のファイル名を入力すると転送がはじまる。このときも緑の字で右上に表示がなされるが、この場合は、第1の(a)が(b)になる。この方法はACOSからの*のプロンプトを検出して対応する形式をとっているので 転送の速度が遅くなる。NTSSを使って阪大のセンターと交信しているときには第1の方法が使えないので、この方法がつけ加えてある。

ファイル転送の状況はCRTに順次テキストの内容を示して知らせてくれるが、そのテキストはメモリーには記録されない。

3—D MSDOSコマンドモード

送受信モードでf・6を押すとこのモードに入り">>d"とプロンプトがでる。dはカレントドライブを示す。この機能を受け持つ関数escq( )を、図2に示す。この関数はtsss.cに含まれている。図で真ん中より下にでているSYSTEM( )がMSDOSコマンドを呼ぶCの標準関数である。LATTICE-Cの場合この関数を呼ぶと、カレントドライブにある実行可能なコマンド(MSDOSの外部コマンド)はどれでもロードして実行にうつる。これは大変危険でそのコマンドを終了したときに、PCがもとの状態に戻らないことがおきる。このプログラムではその点を用心して、図2で示すような関所が作ってある。図にあるコマンドはこのプログラムとは干渉しないようである。この他のコマンドが必要ならelse ifを加えて図のプログラムに続ければよい。図の前半にあるtimep resetの二つはこのプログラムで作った疑似MSDOSコマンドである。

```
void syst() /*ＭＳＤＯＳ内部コマンドの実行とtimep,reset*/
{
int x, i;
long t;
char vr[160];
char sys[50];
for(;;)
        {
        x=getdsk();
        printf("%c>>",x+0x41);
        while(chit()) x=cget();
        gets(sys);
        if(sys[0] == 0) return;
        else if(!strcmp(sys,"timep") || !strcmp(sys,"TIMEP"))
                {
                time(&t);
                clputs("Current time is ");
                clputs(ctime(&t)); clputs("¥r");
                }
        else if(!strcmp(sys,"reset") || !strcmp(sys,"RESET"))
                {
                for(i=0; i < 160 ; i++) vr[i]=0;
                poke(0xa000,2*24*80,vr,160);
                for(i=0; i < 80 ; i++)
                        poke(0xa200,160*24+2*i,"¥x81",1);
                }
        else if(!strncmp(sys,"dir",3) || !strncmp(sys,"DIR",3))
                x=system(sys);
        else if(!strncmp(sys,"type",4) || !strncmp(sys,"TYPE",4))
                x=system(sys);
        else if(!strncmp(sys,"cd",2) || !strncmp(sys,"CD",2))
                x=system(sys);
        else if(!strncmp(sys,"md",2) || !strncmp(sys,"MD",2))
                x=system(sys);
        else if(!strncmp(sys,"del",3) || !strncmp(sys,"DEL",3))
                x=system(sys);
        else if(!strncmp(sys,"copy",4) || !strncmp(sys,"COPY",4))
                x=system(sys);
        }
}
```

図2　MSDOSコマンドコール

timepは年月日時刻を呼び出しメモリーに書き込むコマンドで、交信記録の必要な位置にその情報をいれるために使用する。

resetは画面最下行の文字表示をこのプログラムの動作時のそれに戻すコマンドである。市販のソフトにMSDOSの上で他のプログラムと同時に走らせて、マルチタスク的に使用できる"ですく・きっと"[4]というものがある。この端末プログラムとも一応平行して動作し、たとえば電卓のプログラムを呼び出してその結果を直接ACOSに送ることも可能である。しかし戻るときに最下行の内容をMSDOSの標準の表示にもどしてくれるため、端末プログラムとしては表示がみだれることになる。この疑似コマンドはこの対策として加えられている。

§4　プログラムの中身

全部のプログラムがCで書かれている。Cのプログラムは長くなってソースを全部ここにのせることができないので、作成にあたって工夫した点をひろって以下に順に示す。

4—A　テキスト用大容量メモリーの形式と扱い方

大きな記憶領域を確保するために、Cコンパイラーはd－モデルで使用する。LATTICE－Cの関数getml( )でtextを頭のポインターにする250000+256バイトの領域を確保する。他にCの標準関数calloc( )でレコードの長さを記憶する場所として、texlを頭のポインターとする2バイトずつの領域を8000+50行分確保する。

textの中に、次の四つのポインターを置いて管理をする。

free-ptr　次にレコードを格納する部分の頭を指す。

now-ptr　編集探索モードでカーソルのあるレコードを指す。

end-ptr　textに確保した領域の最後を指す。

endt-ptr　このプログラムではtextで確保した領域がなくなると、また頭から前の記録に上書きして使うようにしてある。このとき、戻る直前に記録したレコードの頭を指す。

探索編集モードではtexlに書かれた長さの情報にしたがってnow-ptrを増減してtextの中を移動する。上に述べた上書き機能のためにプログラムが少し複雑になっている。なおCではレコードの終わりを$(00)_H$で判定するようになっているので、TEXT上での一つのレコードの使用領域はTEXLに記録したレコード長にこの00の1バイト加えたものになっている。

4—B　画面スクロール

編集モードで、できるだけ速くtextの内容を画面に写しとり、スクロールを高速にすることが、プログラムの使いがってをよくするために必要である。

LATTICE-Cの場合poke( )関数が文字列をまとめてメモリー間で転送するようにしてあるのでそれを使用した。textの上のポインターで指定した文字列の後に、必要な数だけ00コードをおぎなってテキストVラムにこのpoke関数で転送する。結果は丁度使いやすい速さが実現できたと考えている。図3はこの機能を受け持つ関数putvram( )である。nqは画面上でカーソルのある行を示す広域変数、またncはそのコラム位置を示す。

4—C　関数キーによる疑似割り込み

BASICにはON KEY GOSUBの機能があるが、Cにはそれにあたるものが見当たらない。このプログラムでは次のようにしてこの機能を実現した。

f・nキーの文字列には、すべて$(02)_H$コードを1字いれておく。キーボード入力の監視ルー

```
void putvram(pt,len)
char *pt;
unsigned int  len;
{
char vr[320];
int j, leng;
        if(len > 80) leng=320;
        else         leng=160;
        for(j=0; j < 320; j++) vr[j]=0;
        j = 0;
        while( (vr[j++]=*(pt++)) != 0)
                vr[j++]=0;
        if(leng==320 && nq==23)
                {
                SCRU;
                nq--;
                poke(0xa000,2*nq*80+nc*2,vr,leng-nc*2);
                }
        else
                poke(0xa000,2*nq*80+nc*2,vr,leng-nc*2);
}
```

図3　画面書き込みプログラム

プでこのコードを見付けると、図4に示した関数 chkfky( )が呼ばれる。図の peek 関数で覗いている $(0050)_H$ セギュメントの $(002A)_H$ にはPC9801 のキーが押されているかどうかを監視して対応するビットの0,1を切り換えるメモリーがある[5]。この関数ではその情報を取り出してそれぞれ所定の関数を呼び出している。SHIFTキーの同時押し下げも、この方法で判定することができる。カナキーも、CTRL キーも同様に判定できるので一つの f・nキーを、数通りに利用することも可能である。

4—D　ファイル受け入れプログラム

図5はACOSからファイルを受け取るプログラムである。受入ファイルを開いたのちＬＩＳＴコマンドをACOS に送ってカレントファイルをPCのファイルに写しとる。ここでファイルの終わりを見付けるために、変数 indを用いて行送りのコード(0a)Hに直ぐ引き続いて CRコード(0d)Hがくるのを監視している。FACOMやHITACの計算機のTSSではプロンプトと共にベルコードを送ってくるのでこのような苦労をしなくてすむ。

1280行の chit( ), cget( )は、 tssm.c に含まれているそれぞれキーボードの入力検査、1字受け取りの関数である。論理的には不必要なこの行があるのはSTOP キー の押し下げの監視を有効にするためである。このプログラムが乗っているMSDOSでは、このような細工を必要とする[6]。

4—E　BIOSコール

RS232Cの初期設定と入出力はBIOS コールを用いている。 4800ボーまでは対応できると

```
int chkfky() /*f・n key による utility(BASIC の ON KEY GOSUB)*/
{
unsigned char ct[16];
        peek(0x0050,0x002a,&ct,16);
        if(ct[12] == 0x04)
                {printf("$$$CON,TSS,,ASC¥n");
                 rslput("$$$CON,TSS,,ASC¥r");}
        else if(ct[12] == 0x08)
                {
                if(iprint)
                        {
                        iprint=0;
                        g_cls();
                        }
                else    {
                        iprint=1;
                        PRIN;
                        lsprintf("¥x1bN¥r¥a");
                        }
                }
        else if(ct[12] == 0x10)
                {
                peek(0xa000,0,vres,160*23);
                rcopen();
                reset();
                }
        else if(ct[12] == 0x20)
                {
                peek(0xa000,0,vres,160*23);
                if((ct[14] & 0x1d) == 0) sdopen();
                else if((ct[14] & 0x1d) == 1) ssdopn();
                reset();
                }
        else if(ct[12] == 0x80)
                {
                peek(0xa000,0,vres,160*23);
                syst();
                reset();
                }
        else if(ct[13] == 0x01)
                {
                peek(0xa000,0,vres,160*23);
                savetx();
                reset();
                }
        else if(ct[13] == 0x02)
                {
                peek(0xa000,0,vres,160*23);
                loadtx();
                reset();
                }
        else if(ct[13] == 0x08)
                { rsterm();
                  rcvfkey();
                  return(1);
                }
        return(0);
}
```

図4　f・nキー割り込みプログラム

```
1010:int rcopen()
1020:{
1030:FILE    *fp;
1040:char    file[64];
1050:int     i, n, ind;
1060:char    e;
1070:        g_print(450,10,"ファイル転入準備");
1080:        printf(" PC recieve File-name : ");
1090:        scanf("%s", file);
1100:        printf("¥n");
1110:
1120:        printf("¥n PC recieve FILE : %s¥n", file);
1130:
1140:        if(NULL == (fp=fopen(file,"w")))
1150:                {
1160:                printf("¥007¥n Cannot open FILE : %s¥n¥n", file);
1170:                g_cls();
1180:                if(iprint) PRIN;
1190:                return(1);
1200:                }
1210:        rslput("LIST¥r");
1220:        g_cls();
1230:        if(iprint) PRIN;
1240:        g_print(450,10,"ファイル転入中");
1250:        ind = 0;
1260:        indr= 1;
1270:        while( indr ){
1280:           if( chit()) cget();
1290:           if(n=rshit())
1300:                {
1310:                for(i=0; i<n; i++)
1320:                    {
1330:                    e=rsget();
1340:                    if (e) {
1350:                        if (e == 0xa)  ind=1;
1360:                        else if (e == 0xd)
1370:                            {
1380:                            cput(e);
1390:                            cput(0xa);
1400:                            if(ind == 1)
1410:                                 { fclose(fp);
1420:                                   printf("close file¥n");
1430:                                   g_cls(); if(iprint) PRIN;
1440:                                   return(0);
1450:                                 }
1460:                            else  fputc(0xa,fp);
1470:                            }
1480:                        else {cput(e); fputc(e,fp); ind=0;}
1490:                      }
1500:                    }
1510:                }
1520:
1530:g_cls();
1540:if(iprint) PRIN;
1550:return(2);
1560:}
```

図5　ファイル受入プログラム

文献(7)にでていたのでこの方式を採用した。このためカナの交信ができなくなっている。

プリンターへの出力もBIOSコールを用いている。この形式は文献(8)にでている。

Cのコンパイラーのせいか、MSDOSのほうが悪いのか、あるいはプログラムの書き方に問題があるのか、原因不明のおかしな現象が2点このプログラムに残っている。

一つは3－Bの編集探索モードでf・5を押して"捜す文字列?"を見た直後に入力していないのに"文字列が……がありません"とでて、もとに戻ってしまう。もう1度f・5を押せばこんどは必ず正常に動作する。最初にこの機能を使用するときによくおきるが、再現性はない。

もう一つは交信モードでf・6を押してMSDOSコマンドモードに入り、>>dのプロンプトを見てコマンドを入力すると、何故かその内容がACOSに送られてエラーメッセージがACOSから返ってくる。この場合も、もう一度f・6を押せば今度は正常に動作する。この現象は数分端末を放置したときによくおきる。このどちらも使用上はあまり問題がないので、これ以上の原因追求をしていない。

マニュアルや参考書を参照してとにかく使用できるプログラムを作ってみた。可能性が広いC言語をマスターするのにはベテランのプログラマーに手ほどきを受けるのが効果的だそうである。ここまでの筆者の経験から、このことを実感として感じた。ここに紹介したプログラムについて御批判や手直しをしていただけることを願っている。

参考文献

1) 畠中　宏　PC 9801用ターミナルプログラム　情報　15号

2) Lattice C Compiler Ver J3.1 リファレンスマニュアル　ライフボード　1986

3) C－TOOL／98　ユーザースマニュアル　ライフボート　1986

4) ですく・きっと　操作説明書　まつもと

5) PC 9801 システム解析　浅野　他　アスキー　1983

6) MS－DOS3.1　ユーザースマニュアル　NEC　1985

7) コウケツ一起　bit　臨時増刊　7　1986　コンピュータ・ネットワーク　p64

8) 横井与次郎　Cランニング・ブック　ラジオ技術社　昭和61年