PC-9801

マウスキー・ドライバ

TMK

■高井　徳男

私もフリーソフトのディスク・ユーティリティ「FD」を重宝しています.「FD」はよく考えられて作ってあるので，素人プログラマーにとってはとても勉強になります.

それでも,「ファイルを見たりディレクトリをあちこち行ったり来たりするのにマウスが使えたらなぁ」と思ったりしたのです.

私が思うまでもなく，マウス操作を可能にする「FDM」というフリーソフトもすでにあるそうですが，残念ながら私はもっていません．そこで，自分なりに作ってみたのが『TMK』です.

私はこれまで常駐プログラムというものを作ったことがなかったので,「これはひとつ，作ってみるしかない．私に与えられた試練だ」と（勝手に）思い，初めて作った常駐プログラムです.

もちろん，FD 専用というわけではありません．いままでカーソル・キーで動かして確定したり ESC キーでキャンセルしていたようなソフトは,『TMK』をインストールすればそのままマウスで操作できるようになります.

TMKの特徴

❶マウス・ドライバ（MOUSE.SYS など）を組み込む必要がない.
❷左右のボタンの設定が変更できる．ただし，あらかじめ『TMK』の用意した中からに限られる.
❸解放しないでボタンの変更ができる.
❹全体として取り扱いが簡単である.

また，これは特徴とは言えませんが，タイマ割り込み（INT 08_H）を専有します.

常駐画面

```
D:¥EXE¥TMK-> tmk
<< TMK Ver.1.00 >> Copyritght (c)1991 Norio Takai.
Installed: Used INT 08H.
LEFT : RETURN
RIGHT: ESC
B.STS: OFF

D:¥EXE¥TMK->
```

CL　CU　CR　SI　SU　VOID　NWL　INS　REP　^Z

常駐／解放の方法

❶常駐

A＞TMK［L(0～9)］［R(0～9)］［B(0～9)］↵

L/R に続く番号は，マウス・ボタンのキー設定を指定します．番号とキーの対応は次のとおりです．スイッチの順番は問いません.

0	RETURN
1	ESC
2	TAB
3	SPACE
4	CTRL＋XFER
5	BS
6	INS
7	DEL
8	HOME CLR
9	HELP

つまり，"TMK L3 R9"としたときは，左ボタンは SPACE キー，右ボタンは HLPE キー対応になります．すでに常駐しているときはこの設定に変更しますし，常駐していないときはこの設定で常駐します.

また，マウスの移動はカーソル・キーに対応していて，これは変更できません．

スイッチを省略して〝TMK ↵〟としたときは，すでに常駐しているときはボタンの設定を表示し，常駐していないときはデフォルト（左はRETURN，右はESCに対応）で常駐します．

『TMK』はキー入力を真似てデータをリング・バッファに格納しています．すべてのキーに対応できるようにすればよいのではと思うでしょうが，すべてのキーのデータをもたせることはプログラム・サイズの増大につながって，常駐プログラムの性格上好ましくないと判断したので，この程度にしました．

実際にはデフォルトの設定で事足りるのではないかと思います．Bスイッチのこともありますし…．

●Bスイッチ

Bについては，本来のキーボードBIOSの処理動作について少し説明の必要があります．キーボードは，キーが押されたときにと離されたときにデータを発します．これによりINT 09_H割り込みがかかり，BIOSに制御が移ります．

BIOSは「入力状態テーブル」を更新して，押されたキーがシフト・キーであれば「シフト・ステータス」を変更します．押されたキーが一般のキーであれば，「シフト・ステータス」参照した上で，変換テーブルからキーデータをキーコードとともに「リング・バッファ」に格納します（このリング・バッファが一杯になるとビープ音が鳴る）．

ここでBスイッチは，この「入力状態テーブル」の更新をするかどうかを選択します．「入力状態テーブル」は，すべてのキーと1対1で用意されているビットマップ・テーブルで，押されていれば〝1〟，押されていなければ〝0〟を示します．

このビットはキーが押されたか離されたことで毎回反転されるだけなので，ユーザーが勝手に割り入ってたまたまタイミングを狂わされたら，以後ずっとキーが押されているかどうかの判別が反転してしまうという危険性をもっているわけです．

そこで，普段（デフォルト）はこのビット操作をしないことにしています．〝B1〟にすると，マウスのボタンや移動によって対応するビットを一度だけ強制的に立て，あとは強制的に0にする操作を行ないます．

4番の CTRL ＋ XFER はFEPを起動するときなどに使えますが，CTRLのための「シフト・ステータス」を変更することはしていません．

また，ボタンを押し続けたとき，『TMK』はソフト的にリピートを疑似的に再現します．

❷解放

```
A>TMK -R ↵
```

スイッチに〝-R〟を指定すると解放します．

❸デフォルト

```
A>TMK -D ↵
```

スイッチに〝-D〟を指定すると，〝L0,R1,B0〟の状態に変更したり，常駐したりします．

❹ヘルプ

不正なスイッチを指定したときは，ヘルプを表示します．

その他

気がついたことなどを述べてみます．

●FEPとの相性

これはとりあえず完成してから気づいたことなのですが，FEPを使うときは，ものによっては相性があるようです．『TMK』を常駐した状態で CTRL ＋ XFER でFEPを起動し，文字を入力したり，そのときマウスを動かしたりしてみました．

手元にあった4つのFEP（ATOK6,DFJ，NECAI，EGBridge）で試してみたら，DFJのときにのみキー入力と画面表示のタイミングがずれるという異常が見られました．

ADDDRVで組み込んだときは**DELDRV**で外すと異常はなくなったので，DFJと『TMK』のバッファ操作が内部的にかち合っているのかも知れません．が，結局解決できませんでした．

また，4を^XFERのマウスのボタンに設定して，ボタンを押してこれら4つのFEPをON/OFFしたところ，ATOK6,DFJ，NECAIの3つはきちんとON/OFFできました．しかし，EGBridgeについてはONはできたのですがOFFができませんでした．EGBridgeはCTRLキーのシフト・ステータスを参照しているのかもしれません．

●直接指定でフル・キー対応に

プログラム・サイズをなんとか小さくしようとしたのですが，これが限界です．やはりアセンブラを使うべきなのでしょうか（でもアセンブラがない）．

しかし，これからゲームやユーティリティを作るときはキーボード対応にすればいいし，マウス対応にプログラミングすることもないでしょう．

キーデータ・コードをプログラム内部に持たせない

ことで，直接指定して組み込むような仕様にすれば，すべてのキーがマウスのボタンや移動に対応させることが割と簡単にできると思います．それが使いやすいかどうかは別としてですが…．

● Turbo C の割り込み

また，私の使っている Turbo C には割り込みをチェーンする関数（MS-C の**_chain_intr()**関数に相当するもの）がないので，先に述べたように INT 08_H を専有する仕様になっています．

単に元の割り込みをコールして専有しないようにしてもいいのですが，この割り込みに多くの TSR が割り込むと，スタックがオーバーフローするおそれがあるのでこうしています．

興味のある方は，ローカル・スタックを設定したり，チェーン関数を自作したりして解決してください．

◘参考文献

1）"TSR プログラミング研究"，C マガジン，'91年7月号，ソフトバンク

2）新版 PC9800テクニカルデータブック，アスキー出版局

3）星野　操：MS-DOS レジデントプログラム入門，技術評論社

4）東工大電算機愛好会＆小高輝真：98ハードに強くなる本II，技術評論社