

# CP/M-86およびコンカレントCP/M-86への exec 機能の組み込み

井田 昌之

CP/M-86 とコンカレント CP/M-86 に exec 機能を組み込む方法を紹介している。exec 機能とはプログラム中から、他のプログラムを実行する機能である。たとえば Lisp インタプリタを実行中にエディタを呼び出したりできる。この機能は MS-DOS にはあるが、CP/M-86 にはない。(本誌)

## プログラム内での コマンド実行: exec 機能

プログラムの内部でコマンド・インタプリタを呼び出し、(キーボードからではなく) 内部からコマンドとして文字列を与え任意に実行できる機能を、C 言語のシステム・コール命名法にちなんで exec 機能と呼ぶことにする。exec 機能があれば組み込みコマンドや、第三者の作成したソフトウェアをそのまま自然に自分のプログラムの一部として実行できる。伝統的な意味で、直接に最も利用価値があるものの一つとしてプリプロセサの形で言語処理系を作ることがある。たとえば C のコードを中間ファイルに出力しておいて、続けて内部的に C コンパイラを呼び出すといった細工が可能になる。

16 ビット・パソコンでよく用いられている OS として MS-DOS と CP/M-86 があげられるが、この両者を exec 機能の点で見ると完全に優劣がついている。すなわち、MS-DOS には exec 機

能があるが CP/M-86 にはない。これはささいなことだが、CP/M-86 の使い勝手を悪くしている。ここで述べる exec 機能はそうした点を自前で補うことを目的としている。

図 1 に筆者のシステム上での会話例を示す。Lisp を実行途中で CP/M のコマンドを実行したくなったときには、

即座に実行させ、また Lisp の処理を継続できる。この機能を組み込んだので、Lisp の中に動的にたくわえていったリスト構造をメモリにおいたままちょっと寄り道をしたり、あるいは画面エディタを呼び出したりできるようになった。この機能を作り、使い始めてから 1 年ほど経過し、その間順調に使ってきた。またコンカレント CP/M-86 へテスト移植し、そこでも正常に動いているので、それらについて紹介する。

ここに述べる方法は Lisp 処理系にだけ有効というわけではない。広く一

```
B>LISP
```

```
      LISP05   ver.2.8
(C) 1984 Masayuki IDA, Aoyama Gakuin Univ.
CS1CS2DSES = 50004C002C003C00
                CS2(USRCSEG) = 16k for V2
```

```
eval> (cpm "dir *.mod")
```

```
B: LSUBR1  MOD : LET      MOD : FUNCALL  MOD : LISP    MOD
B: DO      MOD : PRINT   MOD : LSUBR2  MOD : EVAL    MOD
B: OBLIST  MOD : MICS    MOD : LLXIO   MOD : GETCELL  MOD
B: PROG    MOD : READ    MOD : APPLY   MOD
```

```
B>^C
```

```
Value is...NIL
```

```
eval> (cpm "stat lisp.mod")
```

```
Drive B:                               User : 0
Recs  Bytes  FCBs  Attributes  Name
  39    6k    1 Dir RW      B:LISP .MOD
-----
Total:    6k    1
B: RW, Free Space:          36k
Value is...NIL
```

図1 CP/M-86上で動作しているプログラムから他のプログラムを呼び出した例

筆者は青山学院大学勤務。理工学部経営工学科情報科学研究室に所属している。

般のプログラムに組み込んで用いることができる。

また CP/M-86 への組み込みは、CP/M-86 の開発者である Digital Research およびデジタルリサーチジャパンとは無関係であり、筆者が試作研究の一部として行ったものでだれが利用してもよいが、筆者も含めてだれもその結果について保証するものではないことをお断りしておく。

まず基本的 exec 機能を持っているコンカレント CP/M-86 を説明し、次に CP/M-86 へ移ることにする。

### コンカレント CP/M-86 の exec 機能

コンカレント CP/M-86 は、コマンド・インタプリタ呼び出しのためのエントリを用意している。したがってそれを呼び出せばよい。そのための基本命令列は

```
mov cl, 96h
mov dx, offset command
int 224
```

である。

cl レジスタにはコマンド・インタプリタ呼び出しであることを示す 96<sub>h</sub> を置き、dx レジスタには実行したいコマンド文字列領域の先頭番地を与える。その領域は、先頭 1 バイトを 0<sub>h</sub> とし、直後にコマンド文字列を続け、最後にまた 0<sub>h</sub> を置く。先頭の 1 バイトにゼロを置く点が常識的な文字列表現と異なると思えばよい。したがって次のようになる。

```
command db 0, 'dir *. *', 0
```

この exec 機能はサブタスクの attach であり、自分自身と並行して先へ進むことができる。要求した実行の結果を待つて先へ進みたいとしたら、同期を要求して待たなければならない。そのための一つの方法として、仮想コンソール要求がある。コマンド・インタプリタを起動するとそれがコンソール使用権を握るので、呼び出したプログラムではまったくコンソール I/O ができない状態となる。そこでコンソール要求を出し、これによってタスク終了待ちにできる。このための手順は次のようになる。

```
mov cl, 92h
int 224
```

### タスク優先度の処理:コンカレント CP/M-86 で必要な工夫

これにより、領域 command にセットしておいたコマンドが実行されるが、コンカレント CP/M-86 がマルチタスク/マルチウインドウ OS であることから生ずる約束があり、それを守らなければ正しく機能しない。それはタスクの優先度である。

コンカレント CP/M-86 下で動作するプログラムは OS がタスクとして管理する。もちろん Tmp と呼ばれるコマンド・インタプリタも同様である。

優先度は 1 バイトで表され、通常ユーザ・プログラムの優先度は 0C8<sub>h</sub> である。値が小さい方が優先度が高い。Tmp の優先度は 0C6<sub>h</sub> である。したがって通常は、

優先度 (Tmp) > 優先度 (ユーザ・

プログラム)

となっている。

そこで上述したように、コマンド文字列を与えてコマンドを実行させると、Tmp の優先度の方が高いので、呼び出されたプログラム (コマンド) の実行後、呼び出した側のプログラムに制御が戻らない。コマンド・インタプリタが制御を握ったままになってしまう。

コマンド実行を起動する前に自身の優先度を Tmp より高く変更し、その後コマンド実行を行う。制御が帰ってきたら、元の優先度に戻せばよい。

自分自身の優先度を変更するには、cl レジスタに 91<sub>h</sub>、dl レジスタに優先度を入れてシステムを呼び出す。たとえば優先度を 0C4<sub>h</sub> にしたければ次のようにする。

```
mov cl, 91h
mov dl, 0C4h
int 224
```

これで、優先度を Tmp よりも高くできる。

これらをまとめた手順を図 2 に示す。

### CP/M-86 への exec 機能組み込み

#### BDOS #47 の利用

本来 CP/M-86 には exec 機能がな。そこで、この機能を OS の標準機能に部分的にパッチすることにより代用させる。このために、BDOS #47 (プログラム・チェーン) に白羽の矢をたてた。このプログラム・チェーン機能は、コマンド行となる文字列を与え、そこへ制御を移す機能である。

BDOS #47はプログラムをチェーンするためのエントリであり、いくつかのフェイズに分けて仕事をする場合を想定して用意されている。その基本的な手順は次の通りである。

(1) DMA セグメントの設定および DMA アドレスの設定により、実行すべきコマンドを表す文字列の先頭番地を指定する。

(2) BDOS #47 を実行する。

この手順により、BDOS #47を発行したプログラムは消滅し、指定したコマンドを探し、存在すれば実行する。

制御が移行してしまうので、いったん BDOS #47 を実行すると呼び出したプログラムがそのときまでにメモリ上に構築した環境・状態は一瞬のうちに消滅してしまう。

### BDOS #47 の機能の修正

そこで BDOS #47 のエントリの中をデバガ (DDT 86) を使って、

- (1) 制御を取り戻す方法はないか。
- (2) メモリ・イメージをそのまま保つにはどうしたらよいか。

の2点を念頭におきながら調べていった。その結果、

- ① メモリを解放するか否かのスイッチとパラメータがあり、それを調整すれば、呼び出し側プログラムをメモリにおいたままチェーンできる。
- ② チェインをしたあとは、コマンド・インタプリタへまっすぐ入り、処理が進む。warm bootの所 ('A>' などのコマンド・プロンプトを出力する所) が事実上の終了点である。そこでコマンド・プロンプトを出力

する代わりに、制御を保存しておいた呼び出し側のプログラムに戻すという2点を推理し、パッチにとりかかった。

作成してみると一応正しく動作した。しかし、②の点については、組み込みコマンドを実行した場合、筆者の方法では、パッチ個所を通らず、'A>'の

プロンプトを出してしまう。そこでは更に通常のコマンド実行ができる。呼び出したプログラムは生きているので、コントロールCをタイプすれば呼び出し側プログラムに制御を返すことができる。

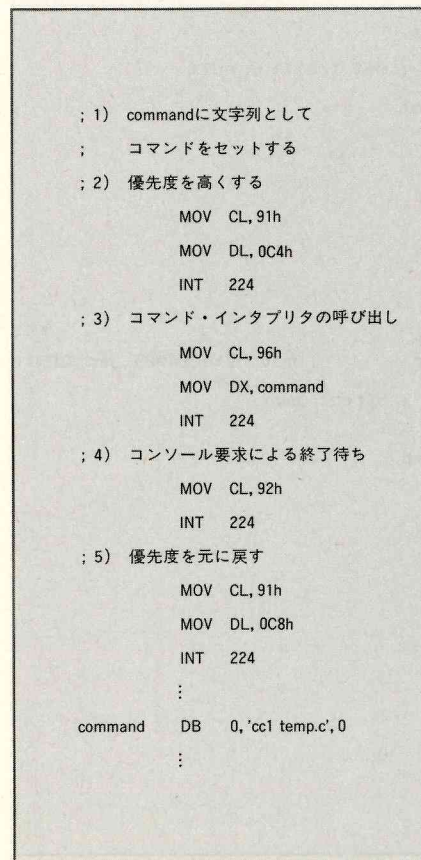


図2 コンカレントCP/M-86におけるexec機能

すべての場合に、指定したコマンドを一つだけ実行し、終了後即座に呼び出し側プログラムに制御が戻るようにするには、ここで述べる方法をさらに拡張しなければならないが、実際に1年間、この形で使っていて都合がいい場合もあるのでこのままにしている。

### 実際のプログラム

以上で述べた手順を図3に示す。また実際のプログラムを図4に示す。

図4の中で、ESレジスタに0040<sub>H</sub>をセットしているが、これは日本電気N 5200-05用CP/M-86のCPM.SYSのセグメント・ベースである。PC 9800

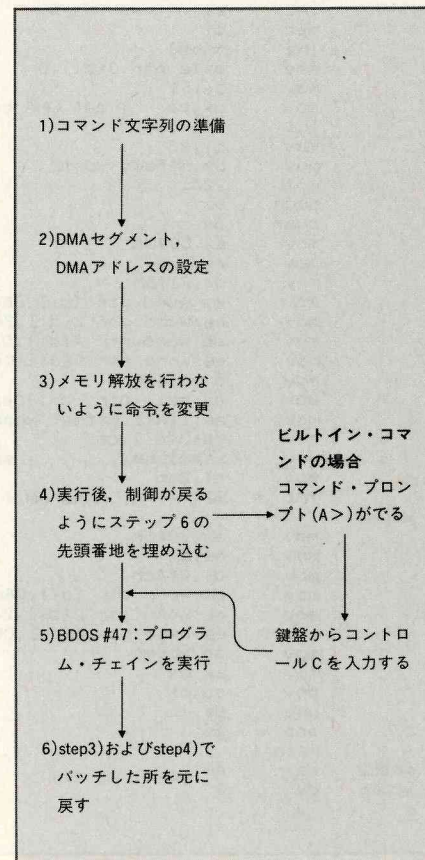


図3 CP/M-86のexec機能の動作概要

用 CP/M の場合には、0040<sub>H</sub> ではなく 0060<sub>H</sub> としなければならない。

自分が用いている CP/M-86 の CPM.SYS のセグメント・ベースがわからない場合、これを確認するにはい

くつかの方法がある。簡単な方法としては、DDT86 デバガを呼び出し、D 0:0 コマンドで表示される 1 番先頭の部分を見ればよい。そこには割り込みのための分岐表がある。2 バイト

ごとに入っている値が、およそ CPM.SYS のベースであると思ってよい。

図 4 の説明を少ししておこう。これは、筆者の Lisp05 におけるプログラムそのままである。この手順は CP/M-86 V1.1 でのみ動作する。しかし、これ以外の版は現在存在していないので、だいたいの CP/M-86 で動くはずである。

Lisp05 では引数は Lisp のポインタ形式で bx レジスタに渡される。説明は省略するが「shl bx, 1」を 2 回行うことにより普通の意味での番地になる。

また、そうして得られた番地+1 の所に文字列長、番地+4 の所に文字列へのポインタがあるという形式になっていて、CP/M 形式(最後に null(0<sub>H</sub>: ゼロ値)が来る所までが一つの文字列)とは少々異なっている。またコマンド・インタプリタに与える文字列は大文字でなければならないので、その変換をあわせて行いながらコマンド・バッファへ文字列をセットする。これがラベル xcpm3 近辺のループである。

その後、DMA セグメントの設定と DMA アドレスの設定を行い、CP/M-86 常駐部の BDOS の一部とコマンド解釈ループの一部とに変更を加え、BDOS #47 を実行する(図 5 参照)。実行を終えて戻ってきたら変更した所を元に戻してやる。

## exec 機能の利用方法

普通のプログラムでこのコードを利用するために、xcpmx というエントリを設けてある。もしこのコードを利用

```

;
; ===== CPM ===== subr 1
;
; CP/M-86 command execution ONLY for CP/M-86 V1.1a
public xcpm
xcpm:
shl bx,1 ! jnc xcpm1 ! retnil
xcpm1: shl bx,1 ; get H-address
mov ax,cs
mov ds,ax
mov si,4[bx] ; get string addr
mov cl,1[bx] ; get string length
xcpmx: ; ----- another entry -----
mov di,offset xcpm2
xcpm3:
mov al,0[si]
cmp al,61h
jc xcpm5
xor al,20h
xcpm5:
mov 0[di],al
inc si ! inc di
dec cl
jnz xcpm3
mov byte ptr 0[di],0 ; set trailing null
mov cl,51
mov dx,cs ; set DMA segment
int 224
mov cl,26
mov dx,offset xcpm2
int 224
push es
push ds
mov ax,0040h
mov es,ax
mov di,21f5h
mov es:word ptr [di],06c6h
mov es:word ptr 2[di],2498h
mov es:word ptr 4[di],0e900h
mov es:word ptr 6[di],0ed54h ; mov byte[2498],jmp 0f51
mov di,0f4ch
mov es:byte ptr [di],0eah ; direct jmpf
mov es:1[di],offset xcpm4
mov es:3[di],cs
mov xcpm6,sp ; save sp
mov cl,2fh
int 224 ; program link
xcpm4:
mov ax,0040h
mov es,ax
mov di,0f4ch
mov es:word ptr [di],06c6h
mov es:word ptr 2[di],2498h
mov es:byte ptr 4[di],00
mov sp,xcpm6
mov ax,cs
mov ss,ax
pop ds
pop es
retnil
xcpm2 rs 40
xcpm6 dw 0

```

図 4 「retnil」はコード・マクロになっており「mov ax, nil ! ret」である。普通に利用するには単に「ret」のみでよい

新登場!!

フロッピー

ダビングマシン

低価格の高速ダビングマシンが完成しました。プログラムの量産、データのバックアップ等にお手軽に御利用頂けます。

<特徴>

- ワンタッチ式です  
電源ONですぐ使えます。プログラムの読み込みなど複雑な操作は一切不要。フロッピーをセットしてスタートボタンを押すだけです。
- あらゆるフォーマットに対応  
単密、倍密の別、セクタ数、セクタ長を問いません。トラックからトラックヘダイレクトにコピーします。
- コピープロテクトもOK  
セクタ破壊、中間トラックの特殊フォーマットもそのまま再現します。
- 超高速です  
5インチ両面1枚を35秒でコピーします。(2D仕様)
- 高信頼性  
子から孫へと5世代目のコピーでも正常動作を確認しています。
- 小型軽量です  
本機単体で動作します。パソコン等への接続は一切不要です。

- 2Dタイプ.....¥288,000
- 2DD,2HDタイプ.....各¥328,000
- 8インチ用.....¥388,000

フロッピードライブ 取扱中

- 価格の一例(送料¥1,000)
- 松下 JA551(5"2D) ¥28,000
- 〃 JU595H(5"2DD/2HD切換) ¥48,000
- 〃 JU363(3.5"1MBタイプ) ¥35,000
- 日電 1165AV(8"PC用純製) ¥65,000
- \*日電、松下、YE他各社ドライブ、ハードディスク等も取扱っています。
- \*大量購入の場合はご相談に応じます。

創業30年の実績を誇る

本州商会

〒556 大阪市浪速区日本橋5-7-10(山田ビル401)  
☎06(644)1881(代表)  
(日曜・祭日も平常通り営業致しております)

<資料請求番号255>

するなら、ここから下を利用するとよい。

この場合、ラベル xcpmx への呼び出し規則は、

si レジスタに文字列の先頭番地  
cl レジスタに文字列の長さ

を与えて、

call xcpmx

と呼ぶ。レジスタ(ax~dx, si, di)はセグメント・レジスタを除いてまったく保存していない。これは筆者の経験的な規則であるので、必要なレジスタがあれば各自保存してほしい。

利用にあたっての問題点・考慮点

最後に、このCP/M-86へのexec機能組み込み上の問題点を述べる。

- ①組み込みコマンド(dir, type等)を実行するとコマンド・プロンプトに戻ってしまう。鍵盤でコントロールCを入力すれば制御を回復できるが、それが困難な場合には問題となる。
- ②BDOS #47のプログラム・チェーンを利用しているので、このexec機能により実行させたプログラムが

SUBMIT コマンドである場合、あるいは、プログラム・チェーンを行う場合、そして、

- ③自分自身と同じメモリ・イメージを呼び出し、そこでさらにexec呼び出しを行う場合、などに不具合が生じる。

②に対しては、SUBMITであれば、内容に対応する解析を行い、順にプログラムを起動してやればできないことはない。チェーンを使っているものを起動する場合もその流れが事前にわかっているのならば対応する順序でxcpmxを呼び出せばできないことはない。たとえばマルチパス方式のCコンパイラなどがこれに該当する。

なお、余談になるが、図4に示したプログラムを含めて筆者のプログラムはDigital ResearchのRASM86というリローケータブル・アセンブラとLIB86, LINK86, (そして一部DRC), を用いて作成している。RASM86にはいわゆるマクロ機能はないが、分割アSEMBルできるので少なくともASM86より便利である。また、MS-DOSのMASMを含めたインテル系アセンブラに比べて「鉛筆が減らない」ので、結構重宝している。 □

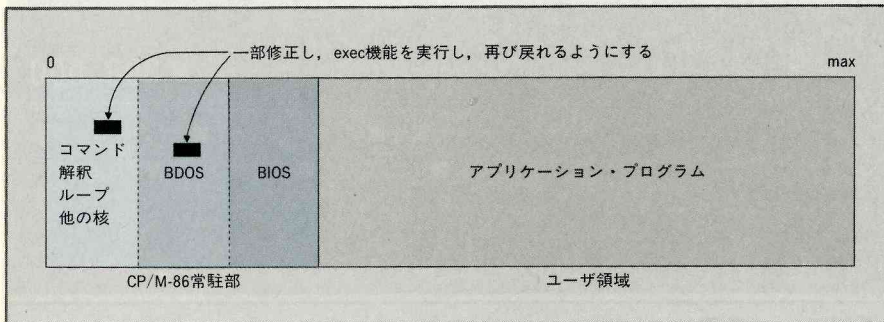


図5 CP/M-86下でのexec機能の実現に必要な常駐部の一時的修正

IN-CIRCUIT EMULATOR

# NICE Z80

## パソコンに直結しCP/Mで、走る 豊富なソフトを有効に活用できる

## EPROMプログラマ内蔵の 開発支援装置!

③従来高価格であったICEと同等の機能(リアルタイムトレース、ブレイクポイントの領域指定、実行時間の計測など)を低価格で実現します。

### 特長

- ①お手持ちのパーソナルコンピュータとRS-232Cで手軽に接続でき、CP/M上で走る数多くの開発用ソフト(EOC-C等)がそのまま利用できます。
- ②ターゲットシステムのメモリ、I/O空間、割込(NMI・INT)、DMA、WAIT等を全て開放しています。
- ④汎用EP-ROMプログラマを内蔵しており、開発済みのソフトの書き込み、ターゲットシステムでの実行が一貫して行えます。

②ターゲットシステムのメモリ、I/O空間、割込(NMI・INT)、DMA、WAIT等を全て開放しています。

ホストとして使用できるパソコン

NEC PC-8001/PC-8801/PC-9801(PLUS80、CP/M86必要) シャープ  
MZ80/MZ2000/MZ2200 富士通 FM-7/8(Z-80カード必要)  
東芝 バンビア7 沖 11-800 富士通 SMC-70 サンヨー MPC-200  
Apple Apple IIe(Z-80カード必要) エプソン QC-10 ソード M-23  
その他 各種CP/Mマシン

### NICE-Z80仕様

- 対象CPU ●Z80 ●Z80A ●Z80B
- メモリ、I/O、INT、NMI、DMA、WAITはターゲットに完全に開放
- 64Kバイトのエミュレーションメモリを内蔵。

- 64K BitダイナミックRAM
- マッピング—256バイト単位
- ライトプロテクト—256バイト単位(ライトプロテクトはターゲット側に対しても有効)

### ■ブレイクポイント

- ブレイク条件: アドレス、データ、ステータス(MI, MR, MW, IR, IW, IA), バスカウンタ、外部トリガ、ライトプロテクトされた領域への書き込み
- ブレイクポイントの指定は無制限。領域指定可能。
- 16Bitバスカウンタ(ループカウンタ)
- ブレイクポイントは上記のものA、Bの条件成立後、Bの条件の成立でブレイクさせるといった2段階検出方式も可能で、複雑なループ内でのブレイクに威力を発揮します。

### ■リアルタイムトレース

- トレース容量: 32Bit×2048マシンサイクル
- トレース内容: アドレス、データ、ステータス(MREQ、IORQ、RD、WR、MI)INT、外部入力2点。
- ポイントによるスタート、ストップ、トレース量の指定が可能。

### ■実行時間の計測

- CPUのクロックを33Bitでカウント(4MHz時、1μs—35分まで)

### ■EPROMライター

- 書き込み可能ROM: 2758、2508、2716、2816A、HN48016、2732、2732A、2532、2764、2564、MCM68764、HN58064、27128、27256、MBM27C256
- 富士通、インテルの高速書き込み可能。
- 2816、48016(EEP-ROM)を消去可能。
- 逆差し検出機能付。

### ■コントロールソフト付属

- ZSIDコンパイル。
- シンボルによるデバッグも可能。
- 5、8パソコンの機種別に用意されています。

### ■パソコンとのインターフェース

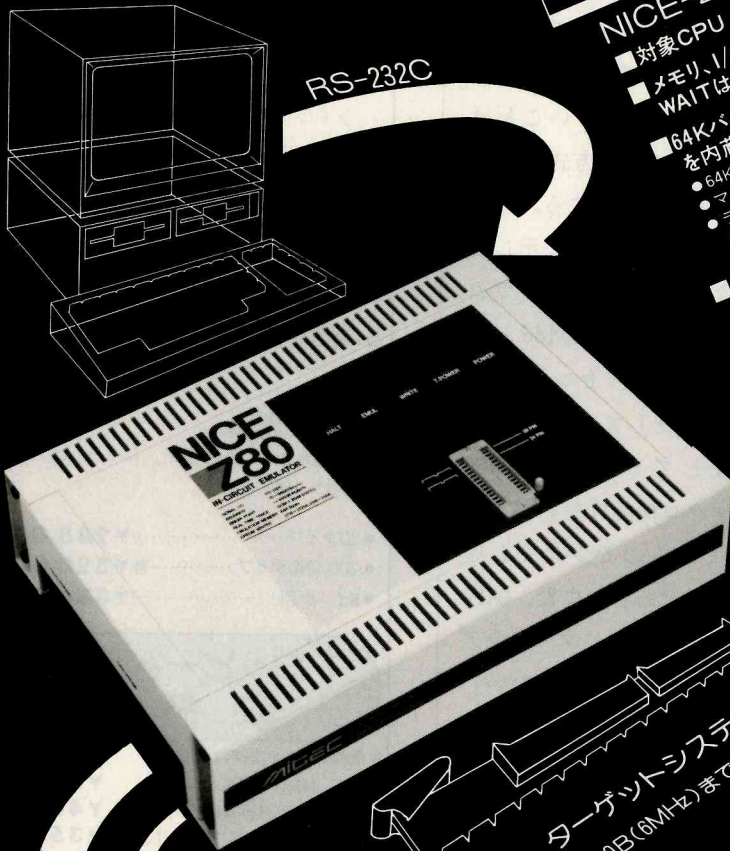
- RS232C(150—19200ボー)

### ■電源内蔵

- AC100V ±10%(50/60Hz) 2A

### ■プロフィール

- 79(H)×330(W)×210(D) 2.9kg



RS-232C

トリガ・プローブ  
CPUプローブ

ターゲットシステム  
Z80B(8MHz)まで対応

¥398,000

- 附属品 ■コントロールソフト ■RS-232Cケーブル  
■CPUプローブ ■トリガ・プローブ ■マニュアル

☆詳しい資料は、お気軽に電話かファクシミリで下記宛お問合せ下さい。

株式会社 **マイテック** 〒103 東京都中央区日本橋茅場町2-13-13(市川ビル)  
TEL: 03-661-3366(代) FAX: 03-639-3044