commododore

commododore

FLOPPY

FLOPPY

MODEL3040 USER MANUAL

commodore japan limited.

FLOPPY

第1章　概要 ―― 5
1. 梱包の解き方 ―― 6
2. 設置に関して ―― 6
3. フロント・パネル ―― 7
4. バック・パネル ―― 7
5. 内部構造 ―― 7
6. ディスケット ―― 8
7. 仕様 ―― 10
8. ディスケットの取り扱い ―― 9

第2章　セットアップ ―― 12
1. モデル3040をコンピューターに接続 ―― 12
2. パワー・オン・テストの実行 ―― 13
3. パーフォーマンス・テスト ―― 14

第3章　モデル3040の使用法 ―― 18
〈モデル3040に関する各種コマンド〉
1. OPEN ―― 19
2. CLOSE ―― 19
3. SAVE ―― 19
4. VERIFY ―― 20
5. LOAD ―― 20
6. PRINT＃ ―― 20

第4章　ディスケット使用に関して ―――― 23
1.ブランクディスケットの場合 ―――― 23
2.一旦使用されたディスケットの場合 ―――― 24
3.現在使用中のディスケットの場合 ―――― 24
4.プログラムをディスケットに書き込む ―――― 25
5.ディスケットからプログラムを読む ―――― 25
6.ディスケットのディレクトリィーを表示する ―― 26
7.ディスケット・ファイルの用い方 ―――― 27
[1]VALIDATE ―――― 27
[2]DUPLICATE ―――― 27
[3]COPY ―――― 28
[4]RENAME ―――― 28
[5]SCRATCH ―――― 28
8.ファイル形態の決定 ―――― 29
9.データ・チャネルの使用 ―――― 30
[1]OPEN ―――― 30
[2]データ・チャネルを閉じる ―――― 30
[3]コマンド・チャネルを閉じる ―――― 31
[4]ディスケットへのデータの連送 ―――― 31
[5]ディスケットからのデータの連送 ―――― 31
10.エラー・メッセージ〈DOSエラー〉 ―――― 32
11.コマンドの簡略化 ―――― 35
[1]DOS SUPPORTプログラムのローディング ―― 35
[2]DOS SUPPORTシンボルの用い方：>及び@ ―― 35
[3]/によるプログラムのローディング ―――― 36
[4]↑によるプログラムのLOAD及び実行 ―――― 36
[5]DOS SUPPORTに関して ―――― 36

第5章　上級ディスク・プログラミング ——— 38
1. DOS(ディスク・オペレーティング・システム) — 38
　ダイレクトアクセスのためのOPENとCLOSE —
2. ディスク・ユーティリティ・コマンド ——— 39
3. BLOCK—READ ——— 40
4. BLOCK—WRITE ——— 41
5. BLOCK—EXECUTE ——— 41
6. BUFFER—POINTER ——— 41
7. BLOCK—ALLOCATE ——— 41
8. MEMORY—WRITE ——— 42
9. MEMORY—READ ——— 42
10. MEMORY—EXECUTE ——— 42
11. USER ——— 43
12. ランダム・アクセスの例 ——— 43
13. シーケンシャル・ファイルの例 ——— 46

第1章　概要

この度、コモドール3040デュアル・フロッピー・ディスク・ドライブをご購入いただき有難うございました。モデル3040のご使用により、コモドール2001及び3000シリーズ　パーソナル・コンピューターの能力を大巾に高めていただけるものと思います。本マニュアルは3040の接続方法及び使用方法について述べてあります。より深く知る必要を感じた場合、2001或いは3000シリーズ　パーソナル・コンピューターのユーザー・マニュアルをご参照下さい。

本マニュアルでは、コモドール3040デュアル・ドライブ・フロッピー・ディスクを指す場合、単にモデル3040と呼ぶ事にします。

モデル3040は、デュアル・ドライブ・インテリジェント・ミニフロッピー・ディスクであり、読み取り／書き込みコントロール・ドライブ・モーター・エレクトロニクス、２つのドライブ・メカニズム、２つのリード／ライト・ヘッドそしてトラック位置決めメカニズムから構成されています。他のコモドール周辺機器と同様に、IEEE-488インターフェイスを用います。しかし、インテリジェント・フロッピーですから、コンピューター側のユーザーメモリーは一切使用しません。従ってモデル3040を使用してもコンピューター側の有効バイト数は減りません。

(注)2040というモデル名のデュアル・フロッピー・ディスクをお持ちの方も、本マニュアルを御利用願います。

1. 梱包の解き方

梱包を完全に解く前にカートンボックスに重大な破損が無い事を御確認下さい。もし破損がある場合は、特に念入りに内部をチェックして下さい。梱包材料を取り除き以下のチェックが終るまで梱包材料は捨てないで下さい。そして右記の物が含まれていないか、或いはモデル3040が正常に動作しない場合、お買い上げになった弊社販売店まで御連絡下さい。

1. モデル3040　デュアル・ドライヴ・フロッピー・ディスク
2. ユーザー・マニュアル
3. テスト・ディスク
4. 保証書

モデル3040をコンピューターに接続する場合、別売の接続ケーブルが必要となります。ケーブルの種類に関しては、販売店にお問い合わせ下さい。

2. 設置に関して

モデル3040の設置場所として、平らな振動の無い場所をお選び下さい。又、ホコリの多い場所を避けて下さい。

故障の際は、販売店にお問い合わせ下さい。

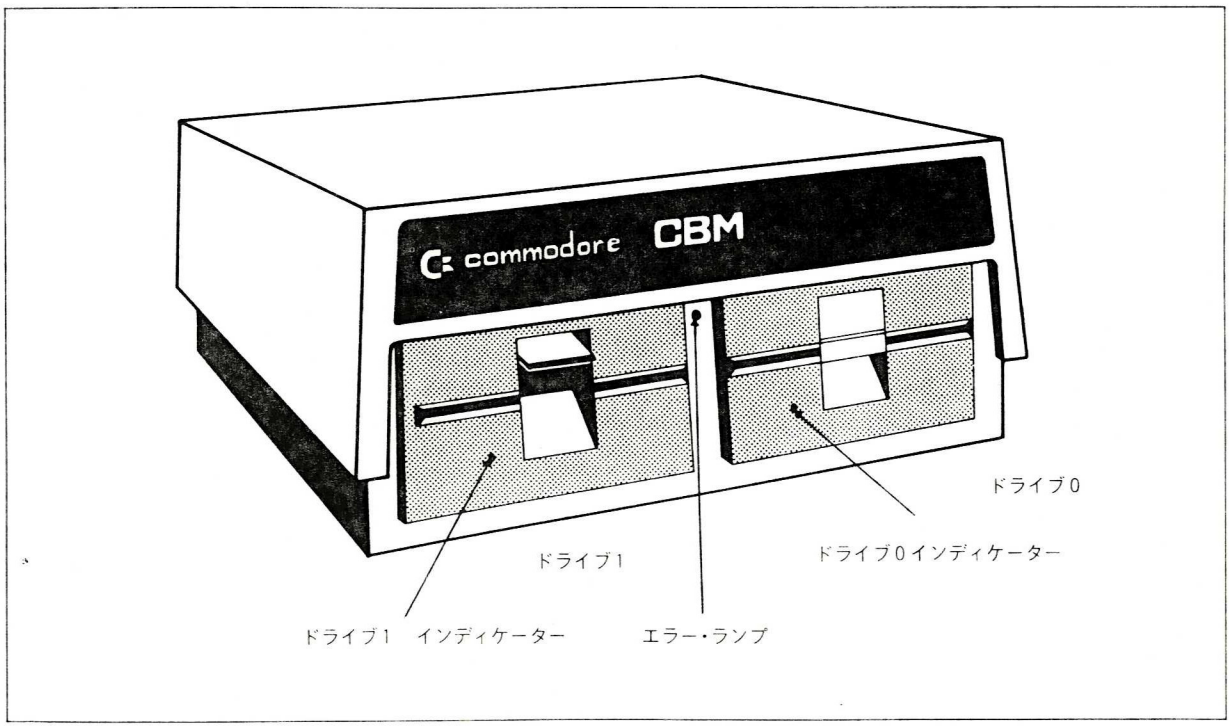

図1. モデル3040：フロント・パネル

3.フロント　パネル

モデル3040のフロント　パネル(図１)には、ディスケットのそう入スロット、ディスケットそう入後閉じるドア、そして３つのLEDライトがあります。ドアを閉じると同時にディスケットは、スピンドルハブの上に自動的にセットされます。３つあるLEDライトの内、右側のライトは、ドライブ０がアクセスされている時に点灯し、又、左側のライトは、ドライブ１がアクセスされている時に点灯します。中央のライトは、ディスク・エラーが生じた時に点灯します。

4.バック　パネル

図２に示されるバック　パネルには、IEEE-488インターフェイス　コネクターの他に、電源スイッチ、ヒューズ、電源コードがあります。

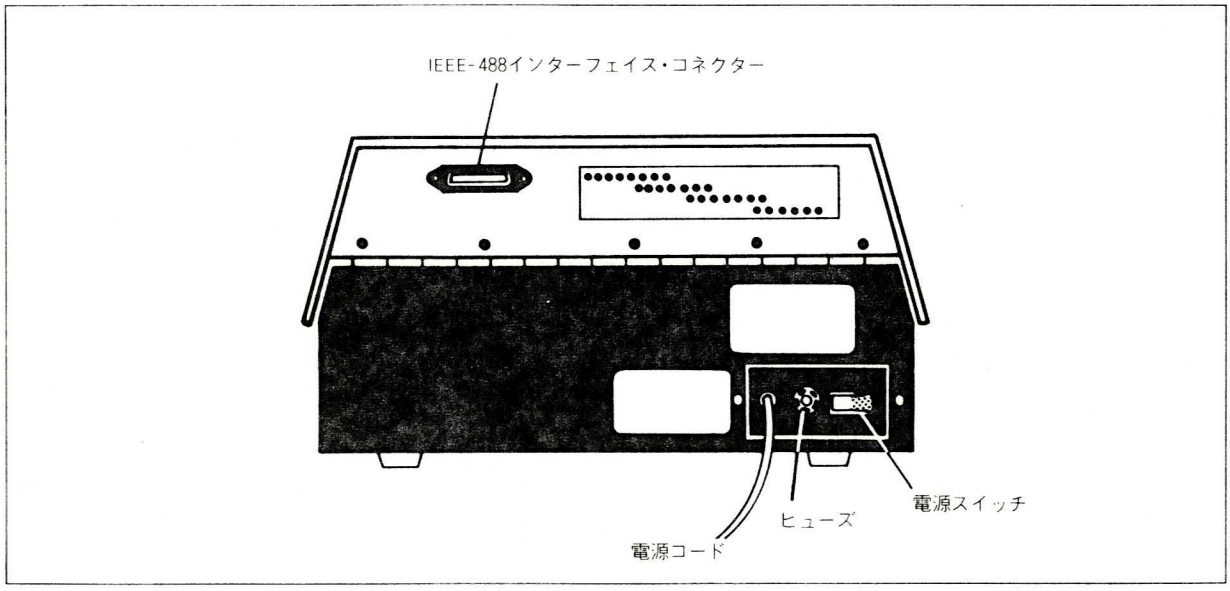

図2.モデル3040：バック・パネル

5.内部構造

ドライブには、シュガート390が２台使用されています。論理回路は、上ケースの内側にあります。概略は図３に示されています。

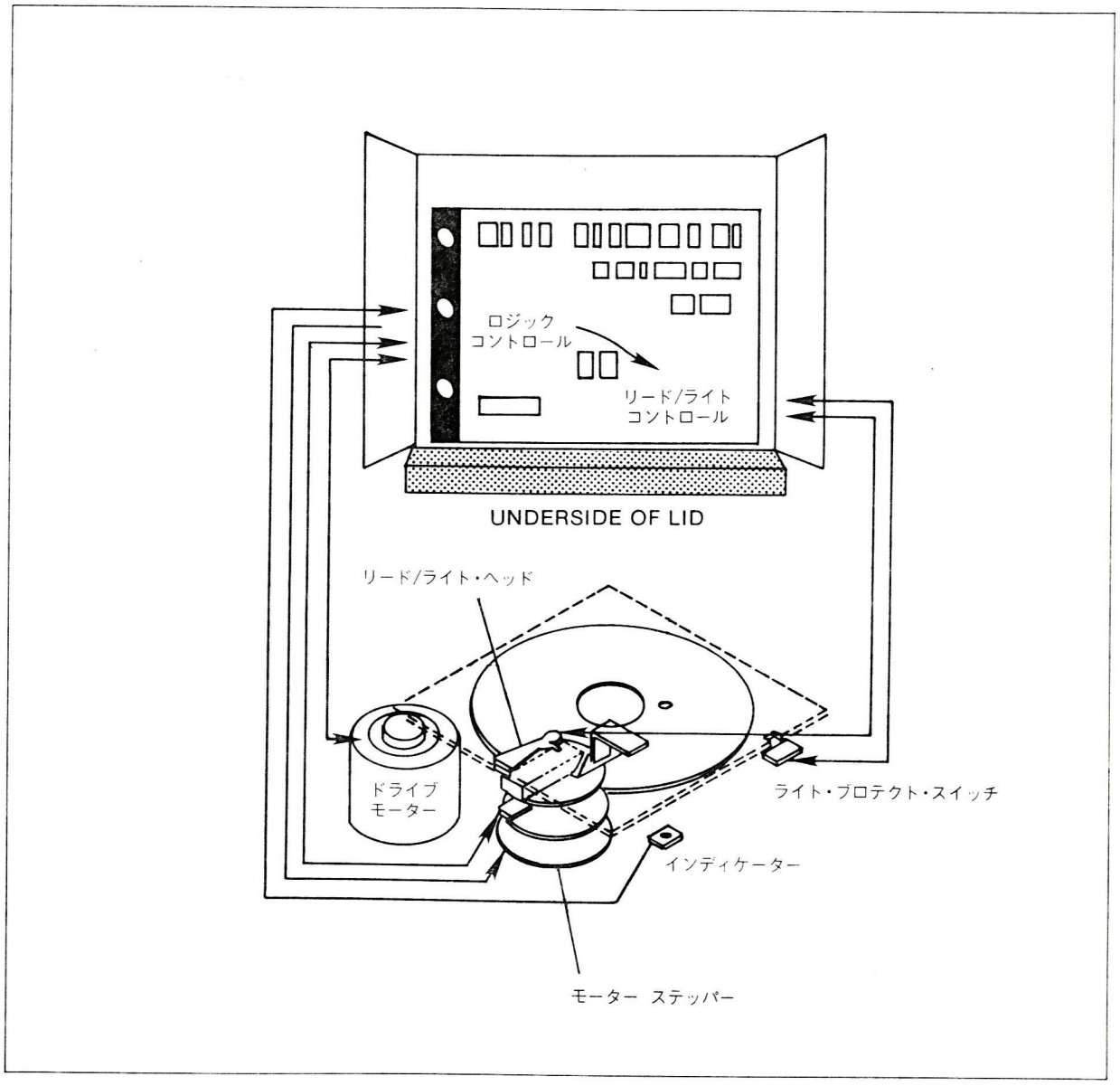

図3.モデル3040：内部構造

6.ディスケット

ディスケット(ミニ・フロッピー、フロッピー、ディスケット或いはミニ・ディスケットとも呼ばれています)は標準サイズの5¼インチのソフト・セクターをご使用下さい。(弊社販売店でお求めになれます。)図4に、その概図が示されています。

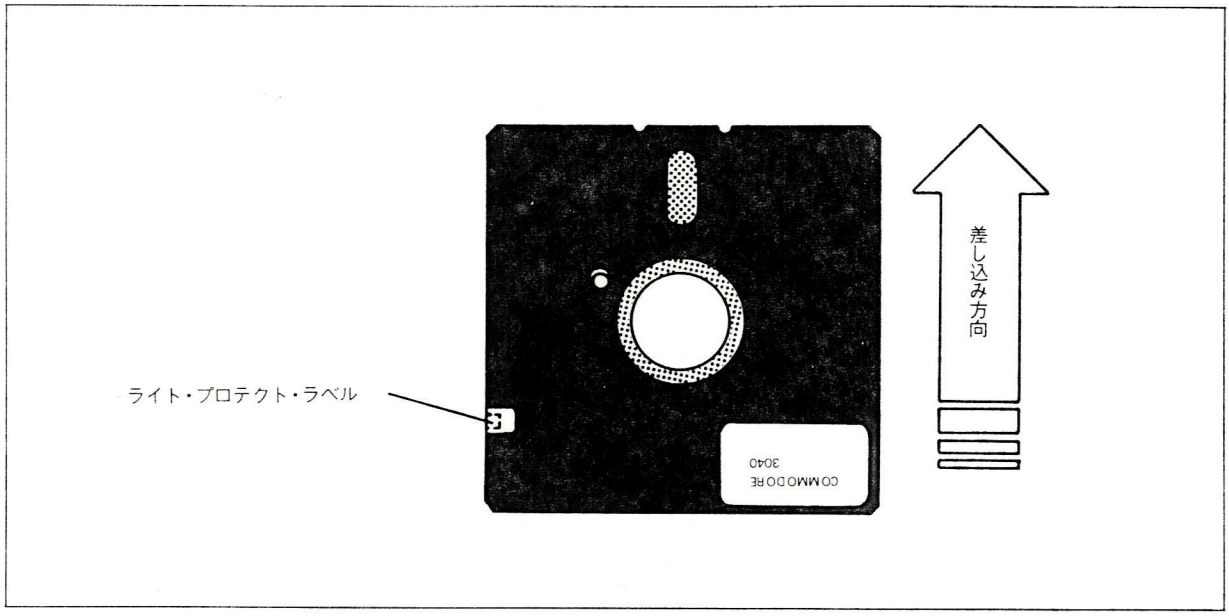

図4.ディスケットの差し込み方

7. 仕　様

モデル3040の仕様については、表１をご参照下さい。

寸法	
高　さ	16.50cm
巾	38.10cm
奥　行	36.45cm

電源	
電　圧	120VAC
周 波 数	50/60Hz
消費電力	50Watts

使用半導体

コントローラー	
6504	マイクロ・プロセッサー
6530	I/O, RAM, ROM

6522	インターバル・タイマー

インターフェイス	
6502	マイクロ・プロセッサー
6532(2)	インターバル・タイマー
6333(2)	ROM

共通部分	
6114(8)	4×1KRAM

使用ドライブ

シュガート Diskettes	SA390(×2) 標準ミニ・フロッピー(ソフト・セクター)

容量

トータル・キャパシティー	176640 bytes/diskette
シクエンシャル	170180 bytes/diskette
ランダム	170850 bytes/diskette
セクター/トラック	17 to 21
バイト/セクター	256
トラック	35
ブロック	690

表1. モデル3040：仕様

8. ディスケットの取り扱い

ディスケットは充分に気をつけて取り扱って下さい。下記の事項に留意し、大切なプログラム或いはデーターを失わないようにして下さい。

a.ドライブからディスケットを取り出した後は、常にケースにしまって下さい。
b.ディスケットを磁石等に近づけないで下さい。
c.ディスケットそのものをプラスティック・カバーから外さないで下さい。
d.プラスティック・ジャケットの上に鉛筆、或いはボールペンで直接字を書かないで下さい。
e.タバコの灰を落とすと、その熱でフロッピーに書き込まれてあったデーターがこわされる事があります。
f.直射日光に当らないように気をつけて下さい。
g.ディスケットの表面を布などでこすらないで下さい。
h.ディスケットを入れたまま、電源ON/OFFは絶対に避けて下さい。

commodore japan limited.

FLOPPY

第2章　セットアップ

モデル3040を使用される前に、モデル3040が適正に作動できるように設置されているかどうかご確認下さい。コンピューターに直接或いは他の周辺機器を通じて正しく接続されている事を確認の上、電源を入れ、同梱されていますテスト、ディスケットを用いて動作チェックをして頂きます。

1. モデル3040をコンピューターに接続

下記の２本の接続ケーブル内の１本が必要となります。(共に別売)

a.PET-IEEEケーブル
　このケーブルは、モデル3040が１台目の周辺装置である場合、必要になります。
b.IEEE-IEEEケーブル
　このケーブルは、モデル3040が１台目の周辺装置でない場合に必要になり、既に接続されている周辺装置の後部のIEEEコネクター経由でコンピューターと接続されます。

(注)コモドール　パーソナル　コンピューターモデル2001をお持ちの方は、販売店にてROMを交換して頂く必要がある場合があります。詳細は販売店にお問い合わせ下さい。(ROMは、モデル3040お買い上げの方には無償交換致します。)

次に以下の手順でモデル3040を接続して下さい。
a.コンピューターの電源を切って下さい。
b.モデル3040をコンピューターにできるだけ、近づけて設置して下さい。この時、電源コードを差し込まないで下さい。
c.PET-IEEEケーブルを用い、コンピューターとモデル3040を接続して下さい。この場合、もし既に他の周辺装置をご使用の場合、IEEE-IEEEケーブルが必要となります。(図５参照)
d.モデル3040の電源コードをコンセントに差し込んで下さい。但し、電源スイッチはOFFのままにしておいて下さい。

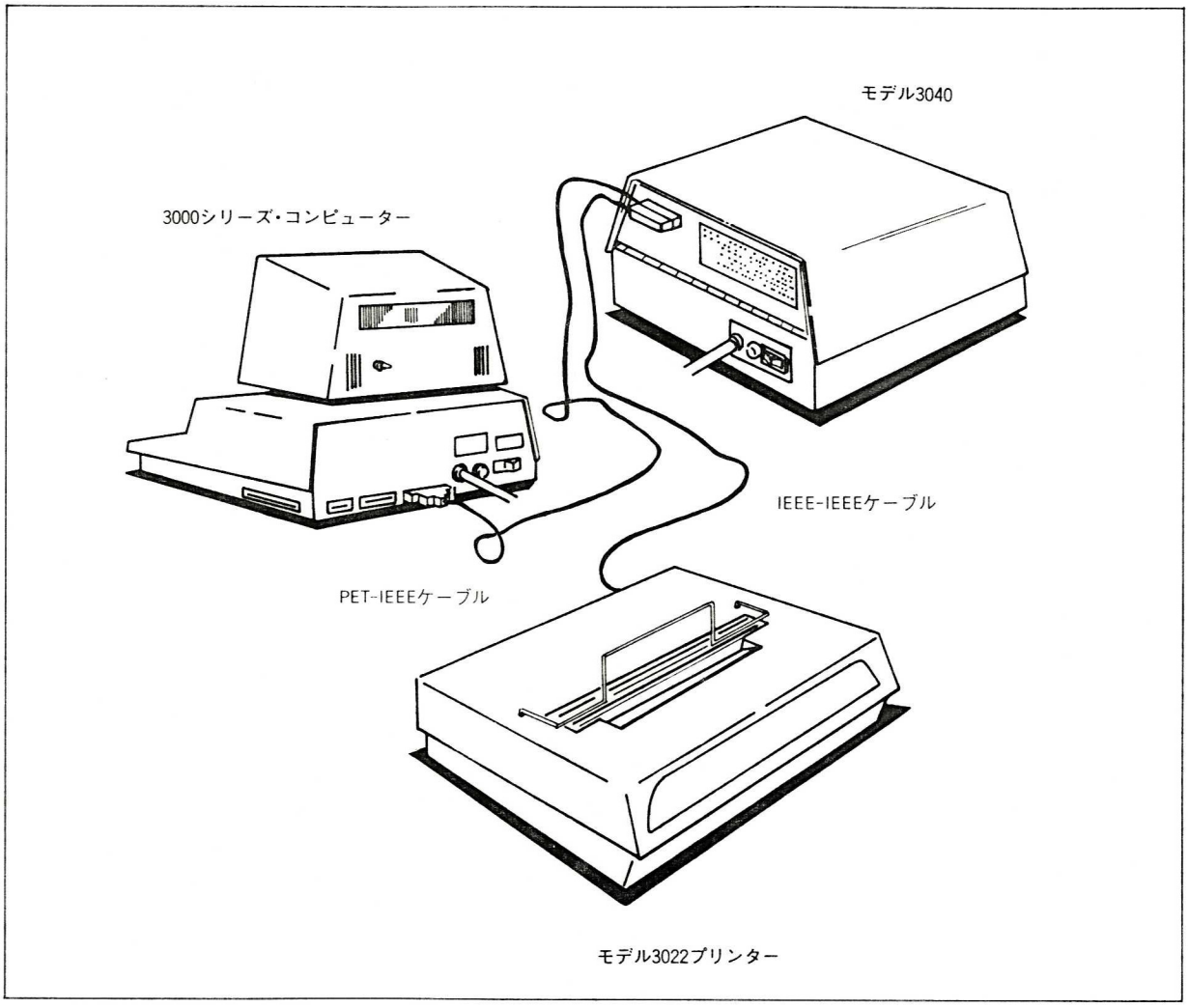

図5.接続方法

2.パワー・オン・テストの実行

a.コンピューターの電源スイッチを入れて下さい。

b.モデル3040のドアを両方共、開いて下さい。そして、どちらのドライブにもディスケットが入っていない事を確認して下さい。入っている場合には、抜いて下さい。

c.モデル3040の電源スイッチをONにして下さい。フロント　パネルにある3つのLEDライトが一瞬点灯した後に消える筈です。もし、3つの内のいづれかのLEDライトが3秒以上点灯し続けたなら、モデル3040の電源スイッチをOFFにして下さい。約1分後に同じ事を繰り返し、同じ現象となった場合、弊社販売店へご連絡下さい。

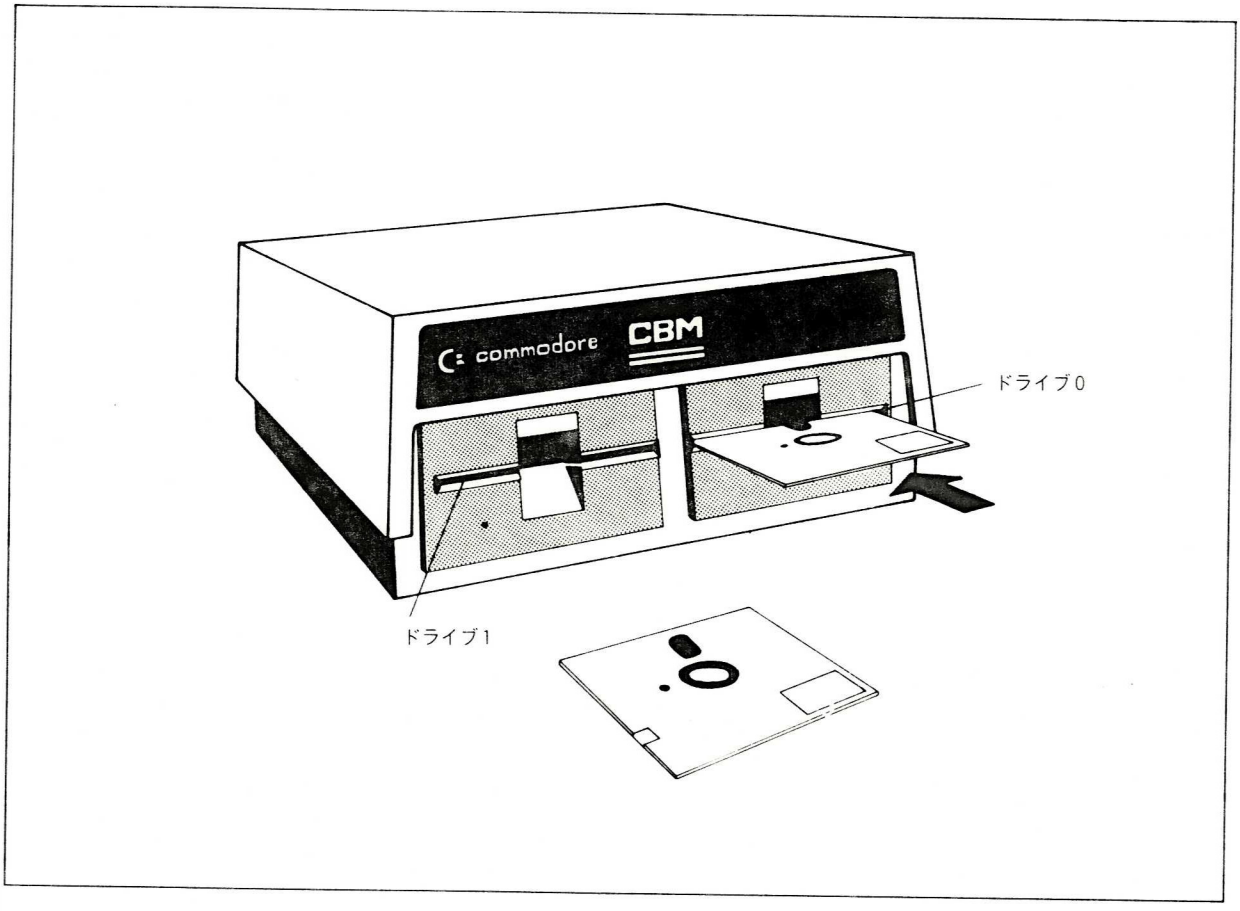

図6.ディスケットの差し込み方

3.パーフォーマンス・テスト

前項のパワー・オン・テスト終了後、このパーフォーマンス・テストに入ります。以下でテストの実行方法を説明しますが、その内容を理解する必要はありません。(詳しいコマンドの説明は第3章以降になされています。)
ここでは単に以下の手順で実行して下さい。又、操作の途中で異常が発生した場合、再度繰り返して下さい。応々にして操作ミスが原因となり得るからです。

コマンドをタイプし終った後、必ずRETURNキーを押して下さい。

【注意事項】

各種コマンドをキー・インする場合、必らず示されるインストラクションに従って下さい。余分なスペースが入った場合エラーとなります。もしエラー・ランプが点灯しても、続けて操作可能です。最後のコマンドを再度キー・インして、エラー・ランプが消えれば、そのまま続けて下さい。

a.ドライブ0に、本体に同梱されていたテスト・ディスケットを入れて下さい。又、ドライブ1には、ブランクのディスケットを入れて下さい。(図6参照)

b.キーボードから、以下を入力して下さい。

OPEN 1, 8, 15

これにより、デバイス番号8(モデル3040)にロジカルファイル1をオープンします。セカンダリー・アドレスの15は、3040へのコマンド・チャンネルをオープンします。スクリーンには入力した通りが表示され、READYと出ます。

【注意事項】

ロジカル・ファイル番号は1から255までのいかなる番号でも可能ですが、セカンダリー・アドレスは常に15である必要があります。又、デバイス番号は工場出荷時にモデル3040の場合は8にセットされていますが、8～15の間の数字に変更可能です。従って、デバイス番号を9に変更した場合、勿論上記の8は9に変える必要があります。

c.次に、キーボードから以下のように入力して下さい。

PRINT#1, "NEW1 : TEST, 99"

PRINT#1により、コマンドがモデル3040に送られます。ドライブ1に入っているディスケットがフォーマットされ、そのディスケットにTESTという名称を与え、又認識番号99を与えます。ディスプレイ上に上記の入力及びREADYが表示されます。ドライブ1のLEDライトが点灯し、ドライブ、モーターが回転開始し、約1分間この状態が続きます。モーターの回転が止まると同時にLEDライトも消えます。

d.キーボードから以下を入力して下さい。

PRINT#1, "I0"

このコマンドにより、ドライブ0がイニシァライズされます。(ドライブ1はフォーマットされた時点でイニシァライズされています。)このイニシァライズにより、磁気ヘッドが正しい位置にセットされます。ディスプレイには、入力メッセージ及びREADYを表示します。ドライブ0のLEDライトが点灯し、モーターが回転します。モーターの回転が止まると、LEDライトも消えます。

e.キーボードから以下を入力して下さい。

LOAD "$0", 8

このコマンドにより、0のドライブに入っているディスケットのディレクトリィーをコンピューターのメモリーに移します。ディスプレイは、以下を表わします。

LOAD "$0", 8
SEARCHING FOR $0
LOADING
READY.

この時、ドライブ0のLEDランプが点灯し、モーターが回転し、自動的に停止します。

f.キーボードから以下を入力して下さい。

LIST

このコマンドにより、コンピューターのメモリーに入ったドライブ0に入っているディスケットのディレクトリィーがディスプレイ上に表示されます。ディスプレイ上には、LISTというコマンド及びドライブ0内のディスケットのディレクトリィー全体が表示されます。このLISTというコマンドは、モデル3040に何ら動作を起させなかった事に気付かれたと思います。

g.次にキーボードから以下を入力して下さい。

LOAD "0 : DIAGNOSTIC BOOT", 8

このコマンドは、ドライブ0に入っているディスケットから自己診断プログラム(DIAGNOSTIC BOOT)をコンピューターのメモリーに入れます。
ディスプレイは入力されたコマンドと以下を表示しま

す。

SEARCHING FOR 0：DIAGNOSTIC BOOT
LOADING
READY.

ディスプレイ上の表示と同時に、ドライブ0のLEDライトが点灯し、ドライブ0のモーターが回転開始します。次項に移る前にディスプレイの表示の指示に従い、ディスクを抜く事を忘れないで下さい。(hでもう一度警告します。)

h.次にキーボードから以下を入力して下さい。

RUN

RETURNキーを押すと同時にプログラムが実行されます。このプログラムが実行されます。このプログラムに依り、I/O, IC, RAM, ROMがチェックされます。ディスプレイ上の指示に従って下さい。3つのLEDライトが連続して点滅すれば、テスト結果はOKという事になります。(gで警告しましたが、表示内容を良く読んで、ディスケットは必ず抜き取って下さい。)
もし3つのLEDのどれかが点灯し続ける場合、その点灯しているLEDライトにより、ある程度の原因は把握できますが、念の為弊社販売店まで御連絡下さい。30秒間プログラムをRUNさせ、モデル3040の電源スイッチをOFFにしてリセットして下さい。以下のステップへ移る前に勿論電源を入れて下さい。

i.次に RUN STOP キーを押し、それから SHIFT キーを押しながら CLR HOME キーを押し、ディスプレイをクリヤーし、又、ディスケットをドライブに戻して下さい。

j.次にキーボードから、以下を入力して下さい。

PRINT#15, "I0"
LOAD "0：PET DISK", 8

このコマンドによりPET DISKというプログラムがドライブ0からコンピューターのメモリーに移されます。コンピューターのディスプレイは、入力内容とともに以下を表示します。

SEARCHING FOR 0：PET DISK
LOADING
READY.

ドライブ0のモーターが回転開始します。

h.次にキーボードから以下を入力して下さい。

RUN

これによりPET DISKプログラムが実行開始され、(RUN STOP キーにより中断されるまで)プログラムが終了すると同時に次のプログラムを自動的にモデル3040より呼び出すので、モーターは何度か回転します。

l RUN STOP キーを押し、プログラムを中断させ、iと同じように SHIFT キーを押し、がら CLR HOME キーを押して下さい。これにより、モデル3040のパーフォーマンス・テストは終了です。

(注)テスト・ディスケットには、ライト・プロテクト・ラベルがついていますので、プログラムの途中でエラーになります。このディスケットは非常に有意義なので、一旦ラベルを外しても、必ず貼っておいて下さい。

commodore japan limited.

FLOPPY

第3章　モデル3042の使用法

〈モデル3040に関する各種コマンド〉

モデル3040に関するコマンドの説明に入る前に、この周辺装置を実際に使われる方は既にBASICをある程度知っておられる事を前提とする事を確認しておきます。下記を理解する為には是非ともコンピューター(2001或いは3000シリーズ)のユーザー・マニュアルをお手元に置き理解しにくい部分に関してはユーザー・マニュアルを参考にして下さい。

説明を簡単にする為に各種省略形を用いますので、その説明から始めます。

省略形	意味
COMMAND STRING	コマンド・ストリング
DATASTRING	ディスケットに入れられるデータ
DDR	データを受ける側のドライブ(デスティネーション・ドライブ)
DN	デバイス・ナンバー(改造がされていない限り、モデル3040のディバイス番号は8です。)
DR	ドライブ番号
DSKNAME	ディスク・ネーム(16文字まで)
FN	ファイル・ネーム(16文字まで)
FT	ファイル・タイプ(PRG, SEQ或いはUSRのみ)
ID	ディスクの認識番号(2文字)
LFN	ロジカル・ファイル番号
MODE	転送モード：書き込み或いは読み取り
SA	セカンダリー・アドレス
SDR	データーを送り出す側のドライブ(ソース・ドライブ)
VNAME	変数名

又、説明文の中で〔　〕が使用されますが、これは〔　〕内は必ずしも必要がない旨の表示です。つまり、N〔EW〕とある場合、NEWと入力しても良いし、只Nだけでも良いという意味です。

【注意事項】
ここで説明しますコマンドは、モデル3040に関しての説明であり、一部のコマンドは、コンピューター或いは他の周辺機器には違った働きをするケースもあります。

この章で説明しますコマンドには以下があり、これはモデル3040からデータを読み込む或いはモデル3040にデータを書き込む事を可能にします。

OPEN, CLOSE, SAVE, VERIFY, LOAD, PRINT#

まずこれらのコマンドが実際どのようにBASICプログラミングの中で用いられるのかを説明し、その働きに関しては後の章で説明します。

1.OPEN

このコマンドにより、モデル3040とのデーターのやりとりが可能になり、又コンピューターとモデル3040の間にチャンネルをオープンします。使用法は次のようになります。

OPEN LFN, DN, SA

LFN(ロジカル・ファイル・ナンバー)は1から255までのいかなる番号でも可となります。DN(デバイス・ナンバー)は通常8になります。SA(セカンダリー・アドレス)はこの場合15となります。15がコマンド及びエラー・メッセージ・チャンネルとなっており、他の番号の使用方法は後に説明します。

2.CLOSE

OPENコマンドでオープンされたファイルをこのコマンドにより閉じます。使用法は次のようになります。

CLOSE LFN

この場合のLFNは、OPENで開けられたものと同一番号を用います。ファイルは必要でない場合すぐに閉じるようにして下さい。コンピューター側では10個、モデル3040側では5個のファイルを同時にOPENできますが有効に活用する為には不必要なファイルはすぐ閉じて下さい。

3.SAVE

このコマンドは、コンピューターのメモリー中にキープされているプログラムを指定されたドライブ番号に入っているディスケットに書き込みます。このコマンドを用いた場合、ファイル・タイプは常にPRG(プログラム)となります。使用方法は次のようになります。

SAVE "DR:FN", DN

DR(ドライブ番号)は1或いは0となり、FN(ファイルネーム)は16文字以下である必要があります。
又、もう1つの使用法は、次のようになります。

SAVE "@DR:FN", DN

この場合、@をつけた事により、DRに入っているFNと同じファイル・ネームを持ったプログラムが新たに作成したプログラムにより置き換えられます。

【注意事項】

@を使用してSAVEする場合、連続して用いることは、さけて下さい。もし連続して使用する場合は、VALIDATEコマンド(後述)を実行してからSAVEして下さい。

4. VERIFY

SAVEコマンドにより、ディスケットにプログラムを書き込んだ後に、このコマンドによりディスケットに書き込まれたプログラムとコンピューターのメモリー内にあるプログラムをバイト毎に比較します。使用法は、

VERIFY "DR:FN", DN

言うまでもなく、DR, FN及びDNは上記3のSAVE使用時と同じものになります。

もう1つの使用法は、

VERIFY "*", DN

この場合、最後にSAVEされたプログラムの照合が行われます。
プログラムをSAVEした後、常にこのコマンドを用いる習慣をつける事により、作られたプログラムが正しく、SAVEされた事を確認できます。

5. LOAD

このコマンドにより、プログラムを所定のディスケットからコンピューターのメモリーに移す事が可能になります。使用法は以下の如くになります。

LOAD "DR:FN", DN

これに依り、DNのDRからFNがコンピューターのメモリーに読み込まれます。

6. PRINT

このコマンドにより、ディスク・コマンドをモデル3040に送る事ができます。使用法は、

PRINT#LFN, "COMMANDSTRINS"

この場合、COMMANDSTRINGには以下のものがあります。

N—(NEW)ディスケットをフォーマットし、イニシアライズします。

I —(INITIALIZE) ドライブヘッドを所定の位置におき、ディレクトリィー・トラックを読み、ディスク・データをDOS(DISK OPERATING SYSTEM)に送り込みます。

V—(VALIDATE) 使用可能BLOCK MAP(BLOCK AVAILABILITY MAP——BAM)を作成し、イニシアライズします。

D—(DUPLICATE) 同一のディスケットを作ります。

C—(COPY) 1つのディスケットの中から特定のプログラムのコピー或いはマージを行ないます。

R—(RENAME) ファイルの名称だけを変えます。

S—(SCRATCH) ファイルを削除します。

ここまで、6つのBASICコマンドそして7つのディスク・コマンドの概略を説明しました。これ以降は、これらのコマンドの実際の使用法を説明します。

commodore japan limited.

FLOPPY

第4章 ディスケット使用に関して

ディスケットをモデル3040にそう入後、そのディスケットを使用する前に常にディスクをイニシャライズする必要があります。INITIALIZE, DUPLICATE或いはNEWというコマンドを用いた場合イニシアライズの働きをも実行します。

【注意事項】

ディスク・コマンドは常にPRINT＃コマンドと共に用いられる必要があります。又、PRINT＃コマンドを用いる場合は、必らずその前に使用されるLFN（ロジカル・ファイル・ナンバー）がOPENされている事が必要です。

1. ブランクディスケットの場合

ブランクディスケットはフォーマット指定がしてありませんので、使用前にフォーマッティングをする必要があります。（弊社で販売しておりますコモドール・ブランドのディスケットは、フォーマット済です。）フォーマット手順は次の通りです。

a.ディスケットを入れる前にモデル3040の電源を入れて下さい。（ディスケットを入れた後に電源のON/OFFをすると、ディスケットの内容を失う場合があります。）

b.次のようにキーボードから入力して下さい。

```
OPEN 1, 8, 15
```

c.次にNEWというディスク・コマンドを使用し、ディスケットをフォーマットし、又イニシャライズします。

"N〔EW〕DR:DSKNAME, ID"

この場合、DRは、実際にディスケットを入れたドライブ番号になり、DSKNAMEは16文字までの長さの適当な名称であり、IDは２文字の認識番号です。

このNEWというコマンドは、35のトラックすべてにヘッダーを書き込み、ディスクのディレクトリィーをクリャーし、BAMをイニシャライズします。

（例）
```
OPEN 1, 8, 15
PRINT＃1, "N0：TEST, XX"
```

2. 一旦使用されたディスケットの場合

このディスケットも、上記1の手順により、ブランク・ディスケットと全く同様に用いる事ができます。元のディスク名、IDが消され、新しくつけた名称及びIDに置き換えられます。

【注意事項】
一旦NEWをかけると、元のデータを一切失う事になりますので気をつけて下さい。

もし必要ならば、ディスクの内容だけを除去し、ディスク名はそのままにしておくか或いはディスク名を変える事も出来ます。この場合2文字のIDは変わりません。この方法でNEWを行う事のメリットは2つ考えられます。1つには、前者の方法ですと、80秒かかるのに対して、後者の方法では20秒しか、かかりません。2つ目は、ID番号を変えない事によりディスケットの管理が明確になります。この場合NEWの方法は、以下の如くになります。

"N〔EW〕DR:DSKNAME"

ID番号を入れずにNEWをかける訳です。

(例) OPEN1，8，15
PRINT#1，"NEW1：NEW DATA"

しかし、もしフォーマットし直す必要がある場合は、この方法を用いる事はできません。

トラック18

3. 現在使用中のディスケットの場合

一旦フォーマッティングされたディスケットは、何のデータが入っていなくとも、2度目以降はフォーマッティングをする必要はありません。この場合、実際に使用する前にイニシャライズする必要はあります。
このINITIALIZEというコマンドは、トラック1にヘッドの位置を合わせます。そしてその後、トラック18に移動し、ディスク名とIDを読み取り、それをDOSメモモリーに入れます。INITIALIZEコマンドの使用法は、以下の通りです。

"I〔NITIALIZE〕〔DR〕"

この場合DRを0ないし1と指定しない限り、両方のドライブがイニシャライズされます。

(例1) OPEN1，8，15
PRINT#1，"I0"
この場合ドライブ0がイニシャライズされます。

(例2) OPEN1，8，15
PRINT#1，"INITIALIZE"
この場合、両方のドライブがイニシャライズされます。

(例3) OPEN1，8，15
PRINT#1，"NEW1：ACCTS, CC"
PRINT#1，"I0"
この場合、両方のドライブがイニシャライズされています。ドライブ1をイニシャライズしていないのは、NEWというコマンドの中には、イニシャライズの機能が含まれているからです。

4. ディスケットにプログラムを書き込む

プログラムをコンピューターのメモリー内で作成し終った後、このプログラムをディスケットに書き込み保存しておく事ができます。この手順はSAVEコマンドの項で説明しました。SAVEを用いてディスケットに書き込まれたデータはすべてプログラム(PRG)と分類されて記憶されます。下記の例でプログラムがディスケットに書き込めるかおわかり頂けると思います。

```
(例)10      ?"THIS IS A TEST"
            SEVE"0 : TESTPROG", 8
            VERIFY"0 : TEST PROG" 8
```

5. ディスケットからプログラムを読む

ディスケットからプログラムを読み取る事は、ディスケットに書き込むのと同様に非常に簡単です。これはLOADコマンドを用いる事で可能になります。ドライブ番号、プログラム名そしてデバイス番号を指定する必要があります。

下記の例は上記4でSAVEしたプログラムをどのようにディスケットから読み取り、コンピューターのメモリーに移しそのプログラムを実行するかを示しています。(この例を実際に演習される場合は、NEWと入力後にして下さい。何故なら、4のプログラムは、コンピューターのメモリー内に残っているからです。)

```
(例1)    LOAD"0 : TESTPROG", 8
         READY
         RUN
         THIS IS A TEST
```

下の例2は、カセット、モデル3040、モデル3022(プリンター)そしてコンピューターを用いた例です。この例では、テープに入っているプログラムをディスケットに移し、その後にディスケットから同じプログラムをコンピューターに読み込ませ、又そのプログラムをプリンターを用いて印字させる方法を示しています。この例では、カセットにDEMOという名称のプログラムが入っていると仮定しました。(太字部分がキーボードからの入力されるべき部分です。

(例2)

```
LOAD"DEMO"

PRESS PLAY ON TAPE#1
OK

SEARCHING FOR DEMO
FOUND DEMO
LOADING
READY.
```

カセットからコンピューターにファイルをLOADします。

```
SAVE"1:DEMO", 8
VERIFY"1:DEMO", 8
READY
NEW
LOAD"1:DEMO", 8
SEARCHING FOR 1:DEMO
LOADING
READY.
RUN
THIS IS A DEMONSTRATION TO
SHOW HOW WE MOVE A PROGRAM
FROM ONE MEDIUM TO ANOTHER.

READY.
OPEN 1, 4

READY.

CMD 1
LIST
PRINT# 1
CLOSE 1
```

コンピューターからディスケットにファイルを入れます。

メモリーからすべてを削除します。

ディスケットからメモリーへプログラムを入れ直します。

実際にプログラムがLOADされた事を確認するためにRUNさせて下さい。

プリンターにプログラムのリストをプリントさせます。
(以下をプリントします。)

```
 10 PRINT"THIS IS A DEMONSTRATION TO"
 20 PRINT"SHOW HOW WE MOVE A PROGRAM"
 30 PRINT"FROM ONE MEDIUM TO ANOTHER"
READY.
```

6, ディスケットのディレクトリィーをディスプレイに表示する

ディレクトリィーをディスプレイに表示するには、以下の方法を用います。

LOAD"$DR", DN

これによりDR(0又は1)のディレクトリィーを表示できます。ディスクのディレクトリィーは、ディスク名及びIDを示すヘッダーを含んでいます。更にファイル名、それぞれのファイルの使用ブロック数、ファイルの種類そして使われていないブロック数も示されます。(1ブロックは256バイトになります。)

本体に同梱されていたTESTディスケットをドライブ0に入れて、そのディレクトリィーを取るならば以下の例のようになります。

(例) LOAD"$0", 8
SEARCHING FOR $0
LOADING
READY.

そしてその内容を表示させるには、LISTと入力して下さい。

7. ディスケット・ファイルの用い方

実際にモデル3040を使い始めると、プログラムをロードしたりセーブしたり以上の事をする筈です。以下ではファイルの扱いに関するディスク・コマンドの説明をします。これらのコマンドは；

	コマンド	
ディスク・レベル	VALIDATE	BAMの作成
	DUPLICATE	ディスクのコピー
ファイル・レベル	COPY	ファイルのコピー
	RENAME	ファイルの名称を変える
	SCRATCH	ファイルを削除する

1 VALIDATE

ディスクに入っているデータに基づき、 BAM(BLOCK AVAILABILITY MAP) を作成します。使い方は；

PRINT＃LFN, “V〔ALIDATE〕DR”

もしDR(ドライブ番号)を指定しなければ最後に使われたドライブに入っているディスケットがVALIDATEされます。

又、このコマンドはBAMを作成するだけでなく正しく閉じられなかったファイルを削除します。VALIDATE実行中にリード・エラーが生じた場合は、ディスケットは元の状態のままとなります。そしてイニシャライズし直してから再度繰り返して下さい。

(例) OPEN 1 , 8 , 15
PRINT＃1, “V1”

2 DUPLICATE

このコマンドはディスク・ネーム、IDそしてすべてのファイルを含めてまったく同一のディスケットを作り出します。この際重要なのは、オリジナル及びコピーに使用されるディスケットとともに良い状態である事を確めて下さい。このコマンドの使い方は；

PRINT＃LFN“D〔UPLICATE〕DDR＝SDR”

この場合DDRにSDRの内容をコピーする事になります。

【注意】
DDRとSDRを逆にすると、必要なデータをすべて失う事になります。充分気をつけて下さい。

(例) OPEN 1 , 8 , 15
PRINT＃1, “D0＝1”

この場合1の内容がドライブ0に入っているディスケットにコピーされます。

3 COPY

このコマンドにより、指定したファイルを同一ディスケット上、或いは、別のディスケットにコピーできます。又、このコマンドはデータファイルをつなぐ場合にも使用できます。このコマンドの使用法は；

PRINT#LFN, "C〔OPY〕DDR:DFN=S
DR:SFN"

この場合DFNそしてSFNともに＊或いは？といった省略形を用いる事はできません。(SCRATCHの項を参照して下さい。)
又、もしSFNがデータファイル(シーケンシャルファイル)の場合、COPYは以下のように使えます。

PRINT#LFN, "C〔ORY〕DDR:DFN=S
DR:SFN, SDR:SFN, SDR=SFN…"

つまり、ファイルをまとめて1つにし、コピーする事も可能です。この場合、最大限4つのファイルまでつなぐ事ができます。
右の例1では、ドライブ0のファイルをドライブ1にファイルする場合を示してあり、例2では、両方のファイルに入っているデータファイルを1つにまとめドライブ1に書き込ませる方法を示したものです。

(例1) PRINT#1, "C1:ACCT1=0:ACCT"

(例2) PRINT#1, "C1:JDATA=1:ACCT1, 0:ADATA, 0:
BDATA"

4 RENAME

既にディスケットに収められているファイルの名称のみを変えるためのコマンドです。この場合、そのディスケットに既に使用されているものと同一の名称を用いるとエラーとなります。使用方法は、

PRINT#1, "R〔ENAME〕DR:NFN=OFN"

この場合、NFNは新しいファイル名であり、OFNは元のファイル名となります。

5 SCRATCH

このコマンドは、特定のディスケットから指定したファイルを削除する場合に用います。この場合、複数のファイルを同時に削除する事もできます。使用法は、

PRINT#LFN, "S〔CRATCH〕DR:FN,
〔DR:〕FN, 〔DR:〕FN……"

2番目以降のFNはDRが指定されない限り、その前で指定されたDRとみなされますが、指定されたDRにFNのない場合は、別のDRもチェックし、あればSCRATCHします。

同じようなファイル名を持ったファイルを同時に削除する場合は＊を用いる事ができます。例えば、ディスクにBASIC, BASE, BASKETというファイルがあり、これらすべてを削除する場合、以下の方法があります。
PRINT#1, "S1:BAS＊"
或いは
PRINT#1, "S1:BA＊"
或いは
PRINT#1, "S1:B＊"

但し、もし同じディスケットにBALLというファイルがあ

り、２番目或いは３番目の方法でSCRATCHすると、削除したくないBALLまでSCRATCHされてしまいます。

一方、文字列の長さが同じファイル名をSCRATCHする場合、その文字列の長さに等しくなる数の？を入力し、SCRATCHする事も可能です。例えば、ABCD, PQRS, VWXYと言った４文字からなるファイルをドライブ０から全部削除する場合、

PRINT＃1, “S0:？？？？”

以上をまとめると以下の如くになります。

削除したいファイル名	入力の仕方	注意事項
FILE A FILE B FILE C	FILE＊	最初の４文字がFILEであるものは、すべて削除されます。
PET FILE PRG FILES POKEFILING	P？？？FI＊	PとFの間にスペースを含め３文字そしてIの後が何であってもこの条件を満たすファイルはすべて削除されます。
AB D FGHIJK XY Z	？？？＊	スペースを含め３文字以上のプログラムはすべて削除されます。

又、PRINT＃1, “S0:＊”

とする事により、ドライブ０に入っているディスケットのファイルをすべてSCRATCHできますが、この場合はむしろNEWコマンドの方が早く削除できます。

8. ファイル形態の決定

ここまでの説明では、プログラムファイルの話しを進めて来ました。このファイルはSAVE或いはLOADコマンドに依りディスケットへの書き込み或いはディスケットからの読み取りができます。この他にファイルの種類にはシーケンシャル(SEQ)とユーザー(USR)があります。これら２つのファイルには、SAVEもLOADも用いる事はできません。

9. データ・チャネルの使用

これまでのところで、プログラムファイルの書き込み・読み取りをSAVE・LOADで実行する方法を説明しましたが、この際、コマンド・チャネルとしてセカンダリー・アドレスの15を利用しました。そして、ロジカル・ファイルは一時に複数オープンできる事も説明しました。コモドールのコンピューター及びモデル3040はその他にも色々な事ができます。色々な種類のデータを記憶させ、またその記憶されたデータを取り出せます。PRINT＃コマンドを用いてモデル3040にデータ及びコマンドを送れますし、又モデル3040からコンピューターにINPUT＃或いはGET＃というコマンドでデータの転送ができます。これらのコマンドの詳しい使用法に関しては、コンピューターのユーザー・マニュアルを参照下さい。

1 OPEN

前にも述べましたように、まずファイルとチャネルをオープンする必要があります。この場合のOPENの使い方は；

OPEN LFN, DN, SA, "DR:FN, FT, MODE"

そして、LFNは1～255までの数、DNは8、SAは2～14までの数、FNはファイル・ネーム、FTはファイル・タイプつまりPRGかUSRかSEQとなり、MODEは、SAの使用されるモード、つまり書き込み(WRITE)か読み取り(READ)の区別です。

(例) OPEN2, 8, 2, "0:FILE1, SEQ, WRITE"
OPEN3, 8, 9, "1:TEST DATA, PRG, WRITE"
OPEN8, 8, 8, "0:NUM, USR, READ"

ドライブ番号の前に@をつける事により、古い同名のファイルを削除し新しいものに自動的に置き換えます。

(例) OPEN3, 8, 5, "@0:JDATA, USR, WRITE"

この場合、JDATAというファイルが置き換えられる訳ですが、もしJDATAが無ければ、@が無い時と同じように機能します。
又、OPENコマンドの中に入るものに変数名を与える事もできます。

(例) AA$＝"0:FILE A, SEQ, READ"
OPEN1, 8, 14, AA$
或いは、
BB$＝"0:FILE A"
OPEN1, 8, 14, BB$＋", SEQ, WRITE"

この方法は、同一のファイルにいくつかのチャネルをオープンする時に便利です。

2 データチャネルのCLOSE

このコマンドは、ファイル及びデータそしてそれに関連するチャネルを閉じます。書き込み用チャネルで開けられたファイルは、閉じられると同時にデータをディスケットに書き込み、ディレクトリィーにファイル名を書き加えます。読み取り用チャネルによって開けられたファイルは、ただ閉じられるだけです。

【注意事項】
INITIALIZE, NEW, DUPLICATE又はVALIDATEにより、ドライブがイニシャライズされる時、そのドライブにつながっているすべてのチャネルは切られます。従って、ファイルが開けられている状態でこれらのコマンドが使われると、ファイルがこわされるケースがあります。

3 コマンドチャネルのCLOSE

コマンドチャネルを閉じる事により、モデル3040に関するすべてのチャネルが閉じられる事になります。
以下の例は、１つのCLOSEコマンドで、いくつかのチャネルを一度に閉じる例です。

(例)

```
OPEN1, 8, 15          コマンドチャネルが開けられました。
OPEN3, 8, 2, "0:
FILE2, SEQ, WRITE"    データチャネルが書き込み用に開けられました。
OPEN4, 8, 5, "0:
FILE2, SEQ, WRITE"

PRINT#3, "IMPORTANT
DATA"

PRINT#4, "MORE DATA"

OPEN3, 4              誤ったチャネルをプリンター用に開けてしまいました。
```

この場合、ディスプレイ上に、？FILE OPEN ERRORというエラー・メッセージが出ます。エラーが発生したので、すべてのコンピューター側のロジカル・ファイルは閉じられております。但し、モデル3040では、それらのチャネルはオープンになっています。そのチャネルを閉じるには、次のように入力して下さい。

```
OPEN1, 8, 15
CLOSE 1
```

これにより、すべてのチャネルが閉じられました。

4 ディスケットへのデータの転送

前に開けられたディスケット・ファイルにPRINT#コマンドを用いてデータを送り込む事ができます。セミコロン(;)をPRINT#コマンドの終わりの意味で入力しない限り、BASICではキャリッジ・リターン(CR)とライン・フィード(LF)がディスケットにデータの１部として書き込まれてしまいます。これをおぼえておかないと、データをディスケットから読み込む時に、CR及びLFが読み込まれてしまいます。

(例) PRINT#2, "JONESABC";CHR$(13);

CHR$(13)は、INPUT#ステートメントを終わらせる為のCRです。

5 ディスケットからのデータの転送

読み取り用にチャネルをオープンした後、INPUT#或いはGET#コマンドを用いて、ディスクからデータを読み取る事ができます。(これらのコマンドの用い方は、コンピューターのユーザー・マニュアルを参考にして下さい。)
INPUT#の使用法は次の通りです。

INPUT#LFN, VNAME, VNAME……

この場合VNAMEはディスクから読む数字か文字列で、数字である場合は1又は2文字の形を取り、文字列の場合は1又は2文字に$サインを追加した形をとります。例えば、

INPUT#1, XY, AB$, PQ

又、GET#の使い方は次のようになります。

GET#LFN, VNAME

このコマンドは、データがBASICにそぐわない形で記憶されている時に便利です。これは又、プログラマーがあるキャラクターにアクセスする場合に便利です。

10. エラー・メッセージ—DOSエラー

モデル3040のエラーランプが点灯した時には、エラーが発生しています。どうしてエラーが発生したのか知る為には、下に示します短かいプログラムを実行する必要があります。実行と同時にエラーの原因をディスプレイ上に表示し、エラー状態を解除します。

まずエラー・メッセージ・チャネルを開き、コンピューターに転送された4つまでの変数を呼び出し、これら変数を表示させる必要があります。下記のプログラムはその1例です。

```
(例) 10  OPEN1, 8, 15
     20  INPUT#1, A$, B$, C$, D$
     30  PRINT  A$, B$, C$, D$
```

エラー・メッセージのフォーマットは以下の如くです。

A, B, C, D,

A：エラー・番号(表2参照)
B：エラー・メッセージ　(表2参照)
C：00或いはトラック番号
D：00或いはセクター番号

エラー番号	エラー・メッセージ	トラック番号	セクター番号
	ステータス・メッセージ		
00	OK	00	00
01.	FILES SCRATCHED	# FILES	00
	リード・エラー		
20	READ ERROR (block header not found)	T	S
21	READ ERROR (no synch character)	T	S
22	READ ERROR (data block not present)	T	S
23	READ ERROR (checksum error in data block)	T	S
24	READ ERROR (byte decoding error)	T	S
27	READ ERROR (checksum error in header)	T	S
	ライト・エラー		
25	WRITE ERROR (write-verify error)	T	S
26	WRITE PROTECT ON	T	S
28	WRITE ERROR (long data block)	T	S
29	DISK ID MISMATCH	T	S

シンタックス・エラー

30	SYNTAX ERROR (general syntax)	00	00
31	SYNTAX ERROR (invalid command)	00	00
32	SYNTAX ERROR (long line)	00	00
33	SYNTAX ERROR (invalid file name)	00	00
34	SYNTAX ERROR (no file given)	00	00
60	WRITE FILE OPEN	00	00
61	FILE NOT OPEN	00	00
62	FILE NOT FOUND	00	00
63	FILE EXISTS	00	00
64	FILE TYPE MISMATCH	00	00
65	NO BLOCK	T	S
70	NO CHANNELS	00	00
71	DIR ERROR	00	00
72	DISK FULL	00	00

表2　エラー・メッセージ

応々にしてここで説明されるエラーは、ハード的なエラーであり、これらのエラーは簡単には直せません。一方他のエラーはコマンドを書き直す事で充分です。

〈書き込みエラー：20, 21, 22, 23, 24, 27〉

20：リードエラー(ブロック・ヘッダー見つからず)

求められるデータ・ブロックのヘッダーをディスク・コントローラーが見つけられない状態です。セクター番号が間違っていたか或いは何らかの理由でヘッダーがこわされた時に生じます。

21：リードエラー(同期キャラクター無し)

ディスク・コントローラーが指定されたトラック上に同期マークをみつけられなかった場合に生じるエラーです。原因として考えられるのは、ヘッドが正しくセットされていないか、ディスケットが入っていないか或いはハードウェアの故障があります。

22：リード・エラー(データ・ブロック無し)

正しく書かれなかったデータ・ブロックをディスコ・コントローラーが読み込み又は、VERIFYの指示を受けた時に生ずるエラーです。このエラー・メッセージはどのブロック或いはどのセクターでエラーが生じたかも表示します。

23：リード・エラー(データ・ブロック内のチェックサム・エラー)

データ・バイトにエラーがある時に、このメッセージが表示されます。データはDOSに読み込まれ、その後チェックサム・エラーになる場合です。このメッセージはグラウンドの問題である可能性もあります。

24：リード・エラー(バイト・デコーディング・エラー)

指定されたデータ・ブロックのヘッダーにエラーがあった場合のメッセージです。この場合そのブロックは読み込まれません。グラウンドの問題である場合もあります。

27：リード・エラー(ヘッダーのチェックサム・エラー)

指定されたデータ・ブロックのヘッダーにエラーがあった場合のメッセージです。この場合そのブロックは読み込まれません。グラウンドの問題である場合もあります。

〈書き込みエラー：25, 26, 28, 29〉

25：ライト・エラー(WRITE-VERIFYエラー)

このメッセージは書き込まれたデータとメモリー内のデータが一致しない場合に表示されます。

26：ライト・プロテクト

ディスケットに書き込みができないようにライト・プロテクト・ラベルが貼ってあるにもかかわらず、そのディスケットに書き込みしようとした場合に発生します。

28：ライト・エラー(データ・ブロックが長すぎる)

データ・ブロックを書き込んだ後、次のヘッダーの同期マークを捜しますが、見つからない場合このエラーが発生します。これは、ディスケットのフォーマットがうまく行かなかった場合(データが次のブロックまで割り込んでしまう)或いはハードウェアに欠陥がある場合に生じます。

29：ディスクIDミスマッチ

イニシャライズされていないディスケットに書き込みを指示した場合、又はヘッダーが良くない場合に発生します。

〈SYNTAXエラー：30, 31, 32, 33, 34〉

30：SYNTAX ERROR(一般的な文法上のエラー)

DOSがコマンド・チャネルに送られたコマンドを判断できない場合に生じます。典型的な例としては、ファイルネームに許されていない番号が用いられたり、パターンが許されない方法で用いられるケースです。例えば、COPYコマンドの後にDFNが2つある場合です。

31：SYNTAX ERROR(未登録コマンド)

DOSが判断できないコマンドを用いた場合表示されます。

32：SYNTAX ERROR(長すぎるコマンド)

40キャラクター以上のコマンドが送られた時に表示されます。

33：SYNTAX ERROR(間違ったファイルネーム)

OPEN或いはSAVEコマンドの中でパターン・マッチングが誤って用いられた場合表示します。

34：SYNTAX ERROR(ファイル名なし)

コマンドの中にファイルを入れ忘れたか、或いはDOSが入れたファイル名を見つけられなかった時、表示します。通常、コマンドの中に“或いは：を用い忘れた場合にこのようなエラーが発生します。

〈ファイル・エラー：60, 61, 62, 63, 64, 65〉

60：WRITE FILE OPEN

書き込む為に開かれたファイルが閉じられる事なく読み取りの為に開かれた場合に生じるエラーです。
開かれていないファイルにアクセスしようとした場合に表示されます。時によって、このメッセージは、出ない場合もあります。

62：FILE NOT FOUND

指定されたドライブ内のディスケットに指示されたファイルの無い場合に表示されます。

63：FILE EXISTS

ディスケット中にあるファイル名と同じファイル名でファイルをディスケットに作成しようと試みる場合に表示します。

64：FILE TYPE MISMATCH

PRGファイルをSEQファイルとして用いようとしたり又はその逆と試みる時に発生します。

65：NO BLOCK

このメッセージはB—Aコマンド(上級ディスク・プログラミングの章を参考にして下さい。)を用いて、アロケートしようとするブロックが既にアロケートされている場合にこのメッセージが表示されます。表示されるパラメーターは、次の空いているトラック及びセクター番号を表示します。もし、パラメーターが0の場合は、指定した番号より大きい番号に空きがないことを示しています。

〈SYSTEM ERRORS：70, 71, 72〉

70：NO CHANNELS

指示したチャネルが無いか或いはすべてのチャネルが使用されている時に表示されます。

71：DIR(ECTORY) ERROR

BAMが、内部でのカウントと一致しない場合に表示します。BAMアロケーションに問題があるか或いはBAMがDOSのなかで重複しています。解決法としては、BAMをメモリーに入れ直すためにイニシャライズし直す必要がありますが、この場合開いているファイルはその時点で終わりとみなされる事もあります。

72：DISK FULL

ディスク上のブロックが完全に使い尽されているか、或いはディレクトリィー(MAX152件)が使い尽されている場合に表示されます。

11. コマンドの簡略化

1 DOS SUPPORTプログラムのローディング

モデル3040に同梱されているディスケットの最初に入っているプログラムが、DOS SUPPORTと呼ばれています。このプログラムを実行しますと、これから述べます事が可能になります。コンピューターのメモリーにこのプログラムに転送する為に特殊LOADコマンドが使用できます。つまり、電源ON後に、特殊LOADコマンドが使用できます。つまり、電源ON後に、

LOAD"✳", 8

と入力する事に依り、DOS SUPPORTをLOADするだけでなく、その前にチャネルをオープンし、自動的にドライブ0をイニシャライズも実行します。(御参考までに、電源をONにした後、上記の入力をすれば、どのディスケットであっても、イニシャライズ後一番最初に入っているプログラムをLOADします。)

一旦LOADし終ったら、RUNと入力し、実行して下さい。他のプログラムと違って、メモリーの最高部に置き変えられます。こうする事により、BASICで書かれているプログラムにもかかわらず、NEWによってもクリヤーされませんし、他のプログラムをLOADしてもこのプログラムは影響されません。

2 DOS SUPPORTシンボルの用い方：〉及び@

一旦DOS SUPPORTが実行されると、ディスクコマンドに、、PRINT#LFNをつける必要がなくなります。ディスク・コマンドに〉或いは@をつけるだけで済みます。又、OPENコマンドも必要なくなります。ここでは、

(例)	入力	機能
	〉I0	PRINT#1, "I0"
	〉S0：FILE1	PRINT#4, "S 0：FILE1"

これらのシンボルは、ディレクトリィーをLOADする場合にも用いる事ができます。もしLOAD"$DR"、8としてディレクトリィーをLOADした場合には、メモリーに入っているプログラムは破壊されてしまいますが、〉SDRとした場合は、プログラムに何ら影響を与えません。

(例) 〉$0　　ドライブ0のディレクトリィーすべてディスプレイ上に表示する。

〉$1：Q＊　　ドライブ1のディレクトリィーの内Qで始まるファイルをディスプレイ上に表示します。

【注意事項】
ディレクトリィーの表示している途中でスクロールを止めるには、スペースキーを押して下さい。押されたと同時に、ストップし、再度続ける場合は、キーボードから何か入力して下さい。

〉(或いは@)の3つ目の使い方は、エラー・メッセージを見る為のものです。

(例)〉は以下のプログラムと同じ意味を持ちます。

```
10  OPEN2, 8, 15
20  INPUT#2, A$, B$, C$, D$
30  PRINTA$, B$, C$, D$
```

3 / によるプログラムのLOAD

/を用いる事により、ディスケットからプログラムを：LOADする事ができます。もし、ドライブ番号を指定しない場合は、両方のドライブをチェックします。

(例) /ACCT
ACCTというプログラムをLOADします。

4 ↑によるプログラムのLOADと実行

↑はプログラムをLOADし、自動的に実行します。

(例) ↑JDATA
JDATAというプログラムをLOAD し実行します。

5 DOS SUPPORTに関して

DOS SUPPORTには、制限があります。これらは、

a.コンピューターの電源が切られた場合には、このプログラムを再びLOADし実行していただく必要があります。

b.DOS SUPPORTに含まれる特殊コマンドは、モデル3040をダイレクト・モードにより操作している時に使えます。言いかえれば、これら特殊コマンドは、プログラム中では使えません。

c.ディスクのディレクトリー(目次)は、以下のコマンドを入力する事により、プリンターに打ち出す事ができます。

```
OPEN4, 4：CMD4
〉$0
PRINT#4：CLOSE4
```

commodore japan limited.

FLOPPY

第5章 上級ディスク・プログラミング

このセクションでは、DOSの構造及びディスク・ユーティリティ・コマンドの説明をします。これらのコマンドは、特殊なディスク・ハンドリング・ルーティン及びランダム・アクセスと言った使用法に使われます。

1. DOS(ディスク・オペレーティング・システム) ダイレクト・アクセスのためのOPENとCLOSE

ディスク・コントローラーとIEEE-488バス間の情報のすべてを管理するのがDOSファイル・コントローラーです。ほとんどのディスクI/Oは、チャネル経由で行われます。つまり、BASICのOPENコマンドを用いてチャネルに対してスペースを割り当て、又1つ或いは2つのI/Oバッファー・エリアを割り当てます。その場合、もしスペース或いはバッファーが無い場合NO CHANNELエラーとなります。DOSは、ディレクトリーを捜したり、ファイルを削除したり、そしてファイルをコピーする場合にもチャネルを通じて行います。
ディスク・コントローラーとファイル・コントローラー間の共通メモリーは、256バイトのバッファー・エリアとしても使われます。16のバッファーの内3つは、BLOCK AVAILABICITY MAP(BAM)、変数スペース、コマンド・チャネルI/Oそしてディスク・コントローラーのジョブ待ちとして、DOSが使用します。

このジョブ待ちは、2つコントローラーをリンクするために重要なものです。ジョブはファイル側から、ディスク・コントローラーにセクター・ヘッダーとコマンドの種類を送り出します。そして、ディスク・コントローラーは、どのジョブを実行するのかを判断して実行します。その時エラー情報がかえってくるので、ジョブが正しく実行されなかった場合、ファイル側では、エラー・メッセージを出す前にコマンドの種類により決められた回数だけ、ジョブの再入力を行います。

OPENコマンド内で用いられるセカンダリー・アドレスはDOSにより、チャネル番号として用いられます。指定する番号は、単にリファレンスとして用いられるだけであり、チャネルの順番を決めるという意味はまったくありません。LOADコマンドはセカンダリー・アドレス0を、そしてSAVEコマンドは1を自動的に指定します。DOSは、これらのセカンダリー・アドレスをLOAD或いはSAVEであると判断します。従って、ファイルを開ける場合、この2つ以外に用いるならば、0及び1の使用は避けて下さい。残りの2～14までは、データのために5本までを開けるためのセカンダリー・アドレスとして使

用可能です。そして、ダイレクト・アクセス用にOPEN及びCLOSEは以下のように使用します。

OPEN2, 8, 4, "#"

又は、

OPEN2, 8, 4, "#12"

これらは、ブロック・コマンドとともに使用され、バッファー1つに対し、1つのチャネルがOPENされます。一番目の例では、チャネル番号4に、空いている最初のバッファーが割り当てられ、2番目の例では、チャネル4に12番目のバッファーが割り当てられます。

もし、バッファーが無い場合、NO CHANNELエラーが生じます。バッファー番号を指定できることにより、実行コマンド内などで特定のバッファーをとっておく時に便利です。
既に割り当てられたバッファーを調べるには、GET#ステートメントを実行します。転送されたバイト数が、バッファー番号です。但しそのバッファーに対して、読み取り或いは書き込みコマンドが用いられた後には、バッファー番号は調べられません。

一方、CLOSEコマンドは、OPENされたチャネルをクリャーし、BAMをディスケットに書き込みます。混乱を防ぐために、ダイレクト・アクセス・チャネルを用いる場合、一方のドライブのみを使用することをすすめます。

2 .ディスク・ユーティリティ・コマンド

モデル3040には以下のコマンドがあります。

コマンド	省略形	使用方
BLOCK-READ	**B-R**	**"B-R:*ch,dr,t,s*"**
BLOCK-WRITE	**B-W**	**"B-W:*ch,dr,t,s*"**
BLOCK-EXECUTIVE	**B-E**	**"B-E:*ch,dr,t,s*"**
BUFFER-POINTER	**B-P**	**"B-P:*ch,p*"**
BLOCK-ALLOCATE	**B-A**	**"B-A:*dr,t,s*"**
BLOCK-FREE	**B-F**	**"B-F:*dr,t,s*"**
memory-write	**M-W**	**"M-W"*adl/adh/nc/data***
memory-read	**M-R**	**"M-R"*adl/adh***
memory-execute	**M-E**	**"M-E"*adl/adh***
USER	**U**	**"Ui*[:parms]*"**

この場合、

ch　チャネル番号。この番号はOPENステートメントに用いられたセカンダリー・アドレスと同一番号である事が必要です。

dr　ドライブ番号(0又は1)

t　トラック番号(1から35)

s　セクター番号(0から20、トラック番号とセクター数の関係は、以下のようになっています。)

トラック番号	セクター	合計
1 ～ 17	0 ～ 20	21
18 ～ 24	0 ～ 19	20
25 ～ 30	0 ～ 17	18
31 ～ 35	0 ～ 16	17

p　バッファー内のポインターの位置

adk　アドレスのローバイト数*

adk　アドレスのハイバイト数*

nc　キャラクター数：1～34*

data　データのバイト数：max34

i　ユーザー・テーブルのインデックス

parms　Uコマンドに関するパラメーター(使用しなくても良い。)

*これらの値は、モデル3040内にシングル・バイトとして入っていますから、CHR$(n)で値を呼び出す必要があります。この場合、nはバイトの10進値です。

以下の例で省略形の用い方を示します。

(例)

"BLOCK-READ：2, 1, 4, 0"

"B-R2, 1, 4, 0"

"B-R"2;1;4;0

"B-READ："2;1;4;0

"　"内のキーワードに続くパラメーターは、以下のキャラクターによりセパレートする事ができます。

キャラクター名	キーボード
スキップ	⇦ CRSR ⇨
スペース	スペース・キー
コンマ	,

これらのキャラクターにより、ASCIIストリング及び整数を送れるようになります。"　" 外のパラメーターは、;でセパレートされる必要があります。以下のセクションでは、PRINT#がすべての例で省略されて説明されています。

3. BLOCK-READ

このコマンドにより、ディスケット上のいかなるブロックへもダイレクト・アクセスが可能になります。他のブロック・コマンドと一緒に使用されると、BASICによりランダム・アクセス・ファイルが作れます。

このコマンドは、ブロックの0ポジションにキャラクター・ポインターを置きます。そして、GET#又はINPUT#により、キャラクターにアクセスすると、EOI(END-OR-IDENTIFY)が送られます。これにより、INPUT#が終了し、ステータス・ワードが64にセットされます。以下の例では、ドライブ1のトラック18、セクター0からブロックをチャネル5のバッファーエリアに読みます。

(例) "B—R";5;1;18;0

4. BLOCK-WRITE

このコマンドが使われると、現在用いられているバッファー・ポインターが、キャラクター・ポインターとして用いられ、ブロックの0ポジションに置かれます。ブロックがディスケットに書き込まれると、バッファー・ポインターの値は1になります。以下の例では、チャネル7のバッファーが、ドライブ0、トラック35、セクター10のブロックに書き込まれます。

(例) "B—W";7;0;35;10

5. BLOCK-EXECUTE

このコマンドにより、ディスケット上にかかれた1ブロック長のプログラムを実行できます。1ブロックがバッファーに読まれた後、ファイル・コントローラーは、内容の実行にかかります。実行の終了には、サブルーチンから戻す命令(RTS)が必要となります。将来のシステム拡張が、このテクニックを使う事により可能になります。又、ユーザーにより命令語をつくることがこのコマンドにより可能となります。以下の例により、ドライブ1の、トラック1、セクター10のブロックをチャネル6のバッファーに読み込み、その内容をバッファー内の0ポジションから実行開始します。

(例) "B—E";6;1;1;;10

6. BUFFER-POINTER

このコマンドにより、指定されチャネルに関するポインターを新しい値に置き換えます。このコマンドは、ブロック内のレコードのある特定のフィールドにアクセスする場合、或いはブロックがレコード別に分けられているならば、個々のレコードを、データをプリントする或いは受ける為にセットする時に便利です。以下の例では、チャネル2のポインターを、ダイレクト・アクセス・バッファー内のデータ・エリアの先頭にセットします。

(例) "B—P";2;1

7. BLOCK-ALLOCATE

DOS内で、指定されたブロックが割り当て済であるべく、BAMが書き換えられます。このコマンドを用いてブロックをアロケートすれば、後のシーケンシャル・オペレーション中にこのブロックは使用されません。書き込み用

ファイルを閉じた時、或いはダイレクト・アクセス・チャネルを閉じた時に新しいBAMがディスケット上に書き込まれます。
もし、指示されたブロックが既に割り当て済みの場合、NO BLOCKを表示し、エラー・チャネルは次の空きブロック番号を表示します。もし、指示されたブロック数より大きな番地内に空きブロックが無い場合、トラック及びセクター・パラメーターとしては0が表示されます。

8. MEMORY-WRITE

このコマンドを含め、メモリー・コマンドはバイトに基づいています。これらは、マシン語でのプログラミング用のものです。メモリー・コマンドを用いてある情報にアクセスする場合は、CHR$を用いれば、BASICステートメントで可能です。メモリー・コマンドは省略形のみ使用可能です。例えば、MEMORY—WRITEは受けつけられませんので、M—Wとする必要があります。又、：の使用も認められていません。

MEMORY—WRITEコマンドは、DOSメモリーへのダイレクト・アクセスを可能にします。このコマンドにより、特殊なルーチンをモデル3040に入れる場合に使えます。このコマンド1つにつき、34バイトまで可能です。アドレスのロー・バイトがハイ・バイトより優先します。下記の例では、バッファー2に4バイト書き込まれます。
($1200又は4008)　4608

(例) "M—W" CHR$(0 0)
CHR$(18)
CHR$(4)
CHR$(32)
CHR$(0)
CHR$(17)
CHR$(96)

9. MEMORY-READ

このコマシドは、コマンド・ストリング中のアドレスに従いバイト数をセットする。このコマンドにより、DOSから変数を或いはバッファーからその内容を読み取ります。このコマンドは又、エラー・チャネルの中味を変えます。何故なら、このチャネルは情報をコンピューター側に送る為に使用されるからです。エラーチャネル(15)に次にくるGET#が、バイトを送ります。INPUT#コマンドは、MEMORYコマンド以外のDOSコマンドが実行されるまで用いられてはなりません。

(例) "M—R" CHR$(128);CHR$(0)

10. MEMORY-EXECUTE

このコマンドにより、サブ・ルーチンを実行する事ができます。DOSに戻るには、RTS、$60でサブ・ルーチンを実行する事ができます。DOSに戻るには、RTS、$60でサブ・ルーチンを終らせて下さい。下記の例は、$ 3180にあるコードを実行させるものです。

(例) "M—E"CHR$(128);CHR$(49)

11. USER

このコマンドは、ポインターにより示されたジャンプ・テーブルに従い、マシン語とのリンクを可能にします。コマンドの中の2番目のキャラクターが、テーブルのインデックスとして使用されます。ASCIIキャラクターの0から9又はAから0が使用可能です。0は、特殊ルーチンへのリンクを含む標準ジャンプ・テーブルにユーザー・ポインターをセットします。

U1は、ブロックをバッファーに読み取り、キャラクター・カウントを225にセットします。そして、ポジション1から255までのすべてのバイトに対しアクセスを可能とします。

U2の使い方は、

U2：ch, dr, t, s

特殊USERコマンドの内、U1(又はUA)とU2(UB)は、B—R或いはB—Wと置き換えられます。
U1の使用法は、

U1：ch, dr, t, s　　C

U2は、バッファーの内容をブロックに書き込みます。この際B—Wと異り、ポジション0の内容を変えません。このコマンドは、B—Rによりブロックを読み取り、B—Pによりポインターの値を変え、U2によりディスケットに書き再度書き込むという具合に使えます。

下記の表3は、電源ON時の、ROM内の標準ジャンプ・テーブルです。

標準ジャンプ・テーブル

ユーザー指定(A)	ユーザー指定(B)	ファンクション
U1	UA	BLOCK-READ replacement
U2	UB	BLOCK-WRITE replacement
U3	UC	jump to $1300
U4	UD	jump to $1303
U5	UE	jump to $1306
U6	UF	jump to $D008
U7	UG	jump to $D00B
U8	UH	jump to $D00E
U9	UI	jump to $D0D5
U:	UJ	power up $E18E

表3

12. ランダム・アクセスの例

B—Aコマンドは、エラー・チャネルを通して次に空いているブロック番号を戻してくるので、レコードの割り当てにも使えます。この仕様故に、実際のディスケットの構成を考えずにランダム・ファイルを作成する事が出来ます。しかしながら、BASICプログラム中でアロケートされたブロック番号を参照するためには、それを記録しておく必要があります。

以下の例で、ブロック・アクセス・コマンドの使い方を説明します。U1及びU2コマンドが用いられている事に注意して下さい。これらのコマンドが用いられたのは、複数の情報が記憶されており、BASICでもって、情報の終わり(END—OF—RECORD)を管理する必要があるからです。尚、より短いアプリケーションの場合、B—R及びB—Wコマンドが利用できます。

この例は、リラティブ・レコード方式を基礎として作られており、BASICプログラミングにより1つの情報へのアクセスを可能にしています。ライン番号2000以下のプログラミングは、リラティブ・レコード・アクセスです。

フィールド・アクセス・ルーチンは、バイナリー及びアルファ・フィールドは左づめし、ニューメリック・フィールドは右づめします。

現実には、プログラムはエラーメッセージを表示するか或いはフィールドに合うように数字をまるめるなどの訂正をする筈です。プログラムに、ソート・サーチそしてキー・フィールドを加える事も可能です。フィールド・マーカーを含めたレコードの長さは、254キャラクター以下である必要があります。BASICのINPUT#の制限故に、フィールドの長さは80バイト以下でなければなりません。

この例では、ランダム・アクセスをサポートするために、2つのシーケンシャル・ファイルが用いられています。それぞれのファイルは、ファイル・コード(ライン番号1100から1180)の際に与えられるファイル名プラス6字の拡張性を持っています。つまり、基本的ファイル名は、10字あるいはそれ以下の長さなので、ファイル名は、スペースにより長くされています。これら2つのファイルは、FILENAME .DESCRそしてFILENAME .KEY 01と名づけられています。

.DESCRには、レコード構成と位置に関する情報が入っており、.KEY01には、各レコードの第1フィールド及びリラティブ・レコード番号が入っています。ここの例では、ランダム・レコードがプログラムの入ったディスケットとは別のディスケットに収められるようになっています。これにより、ランダムデータにより大きなスペースを与えられます。OPENコード(ライン番号1200から1275)によりランダム・ファイルのディスクのIDを比較します。

(例)

〈ファイルの作成〉

1. RAND1 . 0 をLOADし実行する。

a,モデル3040に同梱されていたディスケットをドライブ0に入れる。

b.コマンド・チャネルをOPENし、イニシャライズする。
OPEN15, 8, 15, "I0" RETURN

c.RAND1. 0 をLOADする。
LOAD"0：RAND＊", 8 RETURN

d.ブランク・ディスケットをドライブ1に入れ、以下を入力する。
PRINT#15, "N1：MAILING LIST" RETURN

e.RUN RETURN

2.コンピューターは、DO YOU WISH TO CREATE A FILE？と聞いてきますので、以下を入力して下さい。
Y RETURN

3.コンピューターが、RANDOM FILE NAME？と聞いてきますので、以下を入力して下さい。
PHONE LIST RETURN

4.コンピューターが、KEY FILE DRIVE NUMBER？と表示しますので、以下を入力して下さい。
1 RETURN

5. RANDOM FILE NUMBER?が表示されたら、以下の入力を行って下さい。
   1 RETURN
6. ENTER ID OF RANDOM DISK?が表示されたら、以下を入力して下さい。
   CS RETURN
7. NUMBER OF RECORD?が表示されます。(これは、ファイルの持つべきレコード最大数です。)例えば10件ということにし、以下を入力して下さい。
   10 RETURN
8. NUMBER OF FIELDS PER RECORD?が表示されます。(1レコードあたりの項目の数です。)例えば、4項目とし、以下を入力して下さい。
   4 RETURN
9. INPUT FIELD NAME, FIELD SIZE, FIELD TYPE"
   TYPES: 0=BINARY, 1=NUMBERIC, 2=ALPHA
   と表示し、次に以下を表示する都度、指示に従い入力して下さい。

〈表示〉	〈入力〉
FIELD 1 ?	NAME, 20, 2 RETURN
FIELD 2 ?	PHONE, 15, 2 RETURN
FIELD 3 ?	ADDRESS, 40, 2 RETURN
FIELD 4 ?	COMMENT, 40, 2 RETURN

〈レコードの作成〉

10. WHOSE RECORD DO YOU WISH TO SEE?と表示されます。まだレコードの入力がされていないので、以下のように入力して下さい。
    RETURN
11. ＊＊＊ADD RECORD＊＊＊と表示されます。次に以下の表示の都度、指示に従い入力をして下さい。

〈表示〉	〈入力〉
NAME?	COMMODORE RETURN
PHONE?	06-922-7781 RETURN
ADDRESS?	8-14, 1-CHOME, IKUE, ASAHI-KU, OSAKA RETURN
COMMENT?	PET COMPUTER RETURN

12. WHOSE RECORD DO YOU WISH TO SEE?と再度表示されますので、RETURN キーを押して下さい。
13. ＊＊＊ADD RECORD＊＊＊が表示されますので11と同様にレコードを入力して下さい。

〈レコードの検索〉

14. コンピューターにWHOSE RECORD DO YOU WISH TO SEE?と表示された時に、見たいレコードを入力して下さい。

〈レコードの修正〉

15. レコードを表示した後、ANY MODS?と表示されますので、YES RETURN と入力して下さい。
16. WHICH FIELD? と聞いて来ますので、修正したいFIELDの番号を入力し、修正する事ができます。
17. 修正終了後、ANY MODS?に対し、NOと入力して下さい。

〈レコードの氏名のリストを見る場合〉

18. WHOSE RECORD DO YOU WISH TO SEE?と表示された時点で、以下を入力する事により、リストが見られます。
    / DIR RETURN

〈プログラムを終了する場合〉

19. 18の表示の時点で以下を入力して下さい。
    // RETURN

SEQUENTIAL 1.00

```
 1 REM ************************
 2 REM *       EXAMPLE        *
 3 REM *  READ AND WRITE A    *
 4 REM *  SEQUENTIAL DATA     *
 5 REM *  FILE USING DRIVE 0 *
 9 REM ************************
 10 PRINT"■■INITIALIZE DISK"
 20 DIMA$(25):REM                          SET A$ ARRAY
 30 DIMB(25):REM                           SET B ARRAY
 40 OPEN15,8,15:REM                        OPEN THE COMMAND CHANNEL
 50 PRINT#15,"I0":REM                      INITIALIZE DRIVE ZERO
 60 GOSUB 1000:REM                         READ THE ERROR CHANNEL
 70 CR$=CHR$(13):REM                       SET STRING CR$ TO A CARRIAGE RETURN
 90 PRINT"■WRITE TEST FILE"
 100 REM ************************
 101 REM *                      *
 102 REM *   WRITE TEST FILE    *
 103 REM *                      *
 105 REM ************************
 110 OPEN2,8,2,"@0: TEST FILE ,S,W":REM OPEN LOGICAL FILE 2 ON DISK 8 TO
 111 REM                                   CHALLEL 2 REPLACE DATA FILE NAMED
 112 REM                                   TEST FILE WITH SEQUENTIAL WRITE
 115 GOSUB 1000:REM                        READ THE ERROR CHANNEL
 120 INPUT"A$,B";A$,B:REM                  INPUT NAME, NUMBER INTO A$ AND B
 130 IFA$="END"THEN 160:REM                STOP THE DATA INPUT
 140 PRINT#2,A$","STR$(B)CR$;:REM          PRINT TO THE DISK
 145 GOSUB 1000:REM                        READ THE ERROR CHANNEL
 150 GOTO 120
 160 CLOSE 2:REM                           CLOSE TEST FILE
 200 REM ************************
 201 REM *                      *
 202 REM *   READ TEST FILE     *
 203 REM *                      *
 205 REM ************************
 206 PRINT"■READ TEST FILE"
 210 OPEN2,8,2,"0: TEST FILE ,S,R":REM  OPEN LOGICAL FILE 2 ON DISK 8 TO
 211 REM                                   CHANNEL 2 NAMED TEST FILE WITH
 212 REM                                   SEQUENTIAL READ
 215 GOSUB 1000:REM                        READ THE ERROR CHANNEL
 220 INPUT#2,A$(I),B(I):REM                READ STRING INTO STRING ARRAY A$
 221 REM                                   AND NUMBER INTO ARRAY B
 224 RS=ST:REM                             STORE THE DISK STATUS
 225 GOSUB 1000:REM                        READ THE ERROR CHANNEL
 230 PRINTA$(I),B(I):REM                   PRINT WHAT WAS READ
 240 IFR S=64 THEN 300:REM                 CHECK FOR END OF FILE STATUS
 250 IF RS<>0 THEN 400:REM                 CHECK FOR ERROR IN FILE STATUS
 260 I=I+1:REM                             ADD 1 TO ARRAY POINTER
 270 GOTO 220
 300 CLOSE 2:REM                           CLOSE TEST FILE
 310 END:REM                               END THE PROGRAM EXECUTION
 400 PRINT"■BAD■■DISK■STATUS■IS■■"RS
 410 CLOSE 2:REM                           CLOSE TEST FILE
 420 END:REM                               END THE PROGRAM EXECUTION
 1000 REM ************************
 1001 REM *    READ THE ERROR    *
 1002 REM *       CHANNEL        *
 1005 REM ************************
 1010 INPUT#15,EN$,EM$,ET$,ES$ REM         READ ERROR
 1011 REM                                  EN$ IS THE ERROR NUMBER
 1012 REM                                  EM$ IS THE ERROR MESSAGE
 1013 REM                                  ES$ IS THE ERROR SECTOR
 1020 IF EN$="00" THEN RETURN:REM          RETURN TO MAIN LOGIC IF NO ERRORS
 1030 PRINT"■ERROR ON DISK":REM            PRINT THE ERROR
 1040 PRINTEM$,EN$,ET$ES$:REM              PRINT THE ERROR
 1050 CLOSE 2:REM                          CLOSE TEST FILE
 1060 END:REM                              END THE PROGRAM EXECUTION
READY.
```

RANDOM 1.00

```
 1 REM RANDOM 1.0
 2 REM SUBROUTINES TO MANAGE RANDOM ACCESS FILES
 3 REM  VARIABLES ARE SET FROM DATA OF DESCRIPTOR FILE & KEY LIST FILES...
 4 REM ...DEFINED BY USER PROGRAM
 5 REM VARIABLES SHOULD REFLECT DESIRED FILE STRUCTURE
 6 REM ALL FUNCTIONS ACT UPON THE VARIABLES DEFINED BELOW
10 REM
11 REM ************************
12 REM
15 M$=CHR$(13):REM FIELD MARKER
16 SP$="                                                  "+"":REM SPACE FOR PADDING
20 C0=2:        REM DIRECT CHANNEL
21 C1=3:        REM SEQUENTIAL CHANNEL
25 CC=15:       REM COMMAND CHANNEL
30 D=0:         REM CURRENT DRIVE #
31 T=0:         REM CURRENT TRACK #
32 S=0:         REM CURRENT SECTOR #
35 DD=0:        REM DESCRIPTOR DRIVE #
36 RD=0:        REM RANDOM DRIVE #
40 ID$="":      REM RANDOM DISK ID
45 NR=0:        REM # RECORDS IN R-FILE
46 CR=0:        REM CURRENT RECORD #
47 FR=0:        REM 1ST FREE RECORD UNUSED
50 NF=0:        REM # FIELDS IN RECORD
51 CF=0:        REM CURRENT FIELD #
55 RB=0:        REM # RECORDS PER BLOCK
56 RS=0:        REM RECORD SIZE IN BYTES
60 NB=0:        REM # BLOCKS IN R-FILE
65 E=0:         REM ERROR FLAG, OK =0
66 REM EN$,EM$,ET$,ES$,ET,ES  ERROR CHANNEL VARIABLES
70 EP=.5/256:   REM INTEGER CORRECTION
75 AS=0:        REM INDEX ARRAY ADDRESSING STRATEGY
76 REM AS=0: USE ARRAY INDEX; AS=1: T&S ARE SET, CR= RECORD OFFSET IN BLOCK
90 REM "A" VARIABLES ARE TEMPORARY
95 DN=8:OPENCC,DN,CC:   REM DN= DEVICE NUMBER
98 GOTO2000:    REM START OF USER PROGRAM
99 REM
100 REM ************************
101 REM RANDOM FILE DIMENSION ROUTINE
102 REM  1ST SET NR, NF & NB
103 REM
105 GOSUB150
110 IFFP%=-1THENRETURN
111 FP%=-1
115 DIM FS%(NF) :REM FIELD SIZE
120 DIM FP%(NF) :REM FIELD POSITION
125 REM          FP%(I)= SUM [FS%(I-1)]
130 DIM FT%(NF) :REM FIELD TYPE: 0:BINARY, 1:NUMERIC, 2:ALPHA
135 DIM FH$(NF) :REM FIELD HEADING
140 DIM F$(NF)  :REM FIELD ARGS-ALPHA,BINARY
145 DIM F(NF)   :REM FIELD ARGS-NUMERIC
146 RETURN
150 IFIT%=-1THENRETURN
151 IT%=-1
155 DIM IT%(NB) :REM TRACK INDEX ARRAY
160 DIM IS%(NB) :REM SECTOR INDEX ARRAY
165 DIM K1$(NR) :REM PRIMARY KEY VALUE
170 DIM RR%(NR) :REM RELATIVE RECORD LIST PER KEY
```

```
175 RETURN
200 REM ************************
201 REM UPDATE RECORD, CR
202 REM
205 GOSUB900
210 PRINT#CC,"U1:"C0;D;T;S
215 PRINT#CC,"B-P:"C0;RP
220 FORCF=1TONF
225 GOSUB500
230 NEXTCF
235 PRINT#CC,"U2:"C0;D;T;S
240 GOSUB1000:IFETHEN1900
245 RETURN
300 REM ************************
301 REM READ RECORD, CR
302 REM
305 GOSUB900
310 PRINT#CC,"U1:"C0;D;T;S
315 PRINT#CC,"B-P:"C0;RP
320 GOSUB1000:IFETHEN1900
325 FORCF=1TONF
330 GOSUB600
335 NEXTCF
340 RETURN
400 REM ************************
401 REM UPDATE FIELD(CF) OF RECORD CR, SINGLE FIELD UPDATE
402 REM
405 GOSUB900
410 PRINT#CC,"U1:"C0;D;T;S
415 GOSUB1000:IFETHEN1900
420 PRINT#CC,"B-P:"C0;FP%(CF)+RP
425 GOSUB500 :REM UPDATE FIELD
430 PRINT#CC,"U2:"C0;D;T;S
435 GOSUB1000:IFETHEN1900
440 RETURN
450 REM ************************
451 REM READ FIELD(CF) OF RECORD CR, SINGLE FIELD READ
452 REM
455 GOSUB900
460 PRINT#CC,"U1:"C0;D;T;S
465 GOSUB1000:IFETHEN1900
470 PRINT#CC,"B-P:"C0;FP%(CF)+RP
475 GOSUB600 :REM READ FIELD
480 RETURN
500 REM ************************
501 REM UPDATE FIELD(CF), B-P IS SET
502 REM
510 IFFT%(CF)<>1THEN520
515 A$=RIGHT$(SP$+STR$(F(CF)),FS%(CF)):GOTO530
520 A$=LEFT$(F$(CF)+SP$,FS%(CF))
530 PRINT#C0,A$;M$;
535 RETURN
600 REM ************************
601 REM READ FIELD(CF), B-P IS SET
602 REM
610 IF FT%(CF) THEN645
615 A1$=""
620 FORJ=1TOFS%(CF)
625 GET#C0,A$:IFA$=""THENA$=CHR$(0)
630 A1$=A1$+A$
635 NEXT:F$(CF)=A1$
```

```
640 GET#CO,A$:RETURN
645 INPUT#CO,F$(CF)
650 IFFT%(CF)<>1THEN RETURN
655 F(CF)=VAL(F$(CF)):RETURN
700 REM ************************
701 REM ALLOCATE ONE BLOCK, T & S =REQUESTED TRACK & SECTOR
702 REM  RETURNED T & S ARE ALLOCATED VALUES  (T=18 IS SKIPPED)
703 REM
710 GOSUB800:IFETHEN1900: REM CHECK T & S
715 PRINT#CC,"B-A:"D;T;S
720 INPUT#CC,EN,EM$,ET,ES
725 IFEN=0THENRETURN
730 IFEN<>65THEN1900
735 IFET=18THENT=19:S=0:GOTO715
736 T=ET:S=ES
740 GOTO715
750 REM ************************
751 REM FREE ONE BLOCK, T & S = TRACK & SECTOR
752 REM
760 GOSUB800:IFETHEN1900: REM CHECK T & S
770 PRINT#CC,"B-F:"D;T;S
780 INPUT#CC,EN,EM$,ET,ES
785 IFEN=0THENRETURN
790 GOTO1900
800 REM ************************
801 REM CHECK MAX SECTOR
802 REM
810 IFT>35THEN1900
820 E=0:IFT=0THEN=40:GOTO1900
840 A3=16:IFT>30THEN880
850 A3=17:IFT>24THEN880
860 A3=19:IFT>17THEN880
870 A3=20
880 IFS>A3THEN1900
890 RETURN
900 REM ************************
901 REM SET RECORD'S TRACK, SECTOR & RECORD POINTER FROM INDEX ARRAYS
902 REM
905 D=RD
910 E=0
915 IFAS=-1THENRP=CR*RS+1:GOTO950
920 RP=INT((CR-1)/RB+EP):IFRP>NB OR RP<0THENEN=41:GOTO1900
930 T=IT%(RP):S=IS%(RP)
940 RP=INT(((CR-1)/RB-RP+EP)*RS*RB)+1
950 IFRP>254THEN EN=41:GOTO1900
960 RETURN
1000 REM ************************
1001 REM INPUT 2040 ERROR STATUS
1002 REM
1005 INPUT#CC,EN$,EM$,ET,ES
1010 EN=VAL(EN$):E=0
1015 IF EN$="00" THEN RETURN
1017 ET$=STR$(ET):ES$=STR$(ES)
1020 IFEN$<>RIGHT$("0"+MID$(STR$(EN),2),2)THEN1070
1030 IF EN=1 THEN EM$= ET$+" "+EM$: RETURN
1035 E=E+1
1040 EM$="■"+EN$+"■ "+EM$
1050 IF EN<30 OR EN=65 THEN EM$=EM$+" ON "+ET$+", "+ES$
1060 RETURN
1070 EM$="■SYSTEM NOT RESPONDING PROPERLY"
```

```
1080 EM$=EM$+EN$+EM$+ET$+CR$
1085 E=E+1
1090 RETURN
1100 REM ************************
1101 REM CREATE DESCRIPTOR FILE
1102 REM INPUT: F$= FILENAME
1103 REM        ID$,NR,NF,FS%(),FT%(),FH$()
1104 REM        DD= DESCRIPTOR FILE DRIVE #
1105 REM        RD= RANDOM DISK DRIVE #
1106 REM DRIVES MUST BE INITIALIZED
1109 REM
1110 RS=1:D=RD
1115 FORA0=1TONF:FP%(A0)=RS:RS=FS%(A0)+RS+1:NEXT:RS=RS-1
1116 RB=INT(254/RS+EP)
1120 OPENC0,DN,C0,"#":GOSUB1000:IFETHEN1900
1121 GOSUB1280
1122 PRINT#CC,"B-P:"C0,1
1123 FORA0=1TORB:FORA1=1TONF
1124 PRINT#C0,LEFT$(SP$,FS%(A1));M$;
1126 NEXTA1,A0
1130 NB=INT(NR/RB+EP):IF(NR/RB-NB)*RB>=1THENNB=NB+1
1135 T=1:S=0:GOSUB150
1140 FORA0=0TONB-1:GOSUB710:IFETHEN1900
1145 IT%(A0)=T:IS%(A0)=S:GOSUB430:NEXT
1150 GOSUB710
1152 PRINT#CC,"B-P:"C0,1
1155 PRINT#C0,NR;M$;1;M$;NB;M$;RS;M$;RB;M$;NF;M$;
1160 PRINT#CC,"B-W:"C0;D;T;S
1165 A$=STR$(DD)+":"+LEFT$(F$+SP$,10)+".DESCR,U,W"
1166 OPENC1,DN,C1,A$
1167 GOSUB1000:IFETHEN1900
1168 PRINT#C1,ID$;M$;T;M$;S;M$;
1170 FORA0=1TONF:PRINT#C1,CHR$(FS%(A0));CHR$(FT%(A0));FH$(A0);M$;:NEXT
1175 FORA0=0TONB-1:PRINT#C1,CHR$(IT%(A0));CHR$(IS%(A0));:NEXT
1180 CLOSEC1:CLOSEC0:RETURN
1200 REM ************************
1201 REM OPEN RELATIVE FILE
1202 REM INPUT: F$= FILENAME
1203 REM        DD= DESCRIPTOR FILE DRIVE #
1204 REM        RD= RANDOM DISK DRIVE #
1205 REM DRIVES MUST BE INITIALIZED
1209 REM
1210 A$=STR$(DD)+":"+LEFT$(F$+SP$,10)+".DESCR,U,R"
1215 OPENC1,DN,C1,A$:GOSUB1000:IFETHEN1900
1220 INPUT#C1,ID$,T,S
1225 OPENC0,DN,C0,"#":GOSUB1000:IFETHEN1900
1226 GOSUB1280
1227 PRINT#CC,"B-R:";C0;RD;T;S:GOSUB1000:IFETHEN1900
1230 INPUT#C0,NR,FR,NB,RS,RB,NF
1235 GOSUB100:FT%(0)=T:FS%(0)=S
1240 FORA0=1TONF:GOSUB1298:FS%(A0)=ASC(A$)
1245 GOSUB1298:FT%(A0)=ASC(A$)
1250 INPUT#C1,FH$(A0):NEXT
1255 FORA0=0TONB-1:GOSUB1298:IT%(A0)=ASC(A$)
1260 GOSUB1298:IS%(A0)=ASC(A$):NEXT
1265 GOSUB1000:IFETHEN1900
1270 CLOSEC1
1275 RETURN
1280 PRINT#CC,"U1:";C0;RD;",18,0":GOSUB1000:IFETHEN1900
1285 PRINT#CC,"B-P:";C0;162
```

```
1286 GET#C0,A$,A1$:A$=A$+A1$:IFID$<>A$THENEN=43:EM$="WRONG RAND DISK":GOTO1900
1290 RETURN
1298 GET#C1,A$:IFA$=""THENA$=CHR$(0)
1299 RETURN
1400 REM **************************
1401 REM CLOSE RELATIVE FILE
1402 REM INPUT: VARIABLES FROM OPEN SHOULD BE VALID
1409 REM
1410 PRINT#CC,"B-P:"C0;1
1420 PRINT#C0,NR;M$;FR;M$;NB;M$;RS;M$;RB;M$;NF;M$;
1430 PRINT#CC,"B-W:"C0;D;FT%(0);FS%(0)
1440 CLOSEC0
1490 RETURN
1900 E=E+1:RETURN
2000 INPUT"■■■DO YOU WISH TO CREATE A FILE  N■■■■";A$:IFLEFT$(A$,1)<>"Y"THEN2100
2001 INPUT"■RANDOM FILE NAME";F$
2002 INPUT"KEY FILE DRIVE NUMBER";DD
2003 INPUT"RANDOM FILE DRIVE NUMBER";RD
2005 INPUT"ENTER ID OF RANDOM DISK   ___■■■■";ID$:ID$=LEFT$(ID$,2)
2006 INPUT"NUMBER OF RECORDS";NR
2007 INPUT"NUMBER OF FIELDS PER RECORD";NF
2010 GOSUB110
2015 PRINT"■ INPUT FIELD NAME,FIELD SIZE,FIELD TYPE"
2016 PRINT"     TYPES:  0=BINARY,  1=NUMERIC,  2=ALPHA■"
2019 RS=0
2020 FORI=1TONF:PRINT"FIELD";I,:INPUTFH$(I),FS%(I),FT%(I):RS=FS%(I)+RS+1:NEXT
2025 A$="I":IFDD=RDTHENA$="I"+STR$(DD)
2030 PRINT#CC,A$
2040 GOSUB1100:IFETHEN3900
2050 OPEN4,8,4,STR$(DD)+":"+LEFT$(F$+SP$,10)+",KEY01,U,W"
2055 PRINT#4,0;M$;:CLOSE4
2090 GOTO2120
2100 REM OPEN RANDOM FILE FOR ACCESS
2103 INPUT"■RANDOM FILE NAME";F$
2105 INPUT"KEY FILE DRIVE NUMBER";DD
2110 INPUT"RANDOM FILE DRIVE NUMBER";RD
2120 GOSUB1200:IFETHEN3900
2140 OPEN4,8,4,STR$(DD)+":"+LEFT$(F$+SP$,10)+",KEY01,U"
2142 INPUT#4,RR:IFRR=0THEN2147
2145 FORI=1TORR:INPUT#4,K1$(I),RR%(I):NEXT
2147 CLOSE4
2150 PRINT"■■■■■■■■■■■■■■■■■■■■SAMPLE RANDOM ACCESS■"
2155 PRINT"TYPE // TO QUIT■"
2156 PRINT"(HIT RETURN TO ADD RECORD)"
2160 PRINT"■■WHOSE RECORD DO YOU"
2161 INPUT"WISH TO SEE   ■■■";RR$
2165 IFRR$=" "THEN2310
2167 IFRR$="//"THEN2400
2168 IFRR$="/DIR"THENGOSUB4000:GOTO2160
2170 FORII=1TORR:IFK1$(II)<>RR$THENNEXT:GOTO2300
2175 CR=RR%(II):GOSUB300
2180 FORI=1TONF:PRINTI;")"FH$(I)":",F$(I):NEXT:PRINT
2185 FF=0
2190 INPUT"ANY MODS   ■■■";A$:IFLEFT$(A$,1)<>"Y"THEN2220
2195 INPUT"WHICH FIELD";A
2200 PRINT"  "F$(A):PRINT"□",:INPUTF$(A):F(A)=VAL(F$(A))
2210 FF=1:GOTO2190
2220 IFFF=0THEN2160
2222 IFA=1THENK1$(II)=F$(A)
2225 GOSUB200
2230 GOTO2160
```

```
 2300 PRINT"■RECORD NOT PRESENT"
 2305 INPUT"DO YOU WISH TO ADD";A$:IFLEFT$(A$,1)<>"Y"THEN2160
 2310 PRINT"■■■■ ADD RECORD ■■■■■■"
 2312 IFFR>NRTHEN2500
 2315 CR=FR:FR=FR+1:RR=RR+1
 2320 FORI=1TONF:PRINTFH$(I);:INPUTF$(I):F(I)=VAL(F$(I)):NEXT
 2330 GOSUB200
 2340 K1$(RR)=F$(1):RR%(RR)=CR
 2350 GOTO2160
 2400 REM CLOSE RAND FILE
 2405 GOSUB1400
 2410 OPEN4,8,4,"@"+STR$(DD)+":"+LEFT$(F$+SP$,10)+",KEY01,U,W"
 2420 GOSUB1000:IFETHEN3900
 2430 PRINT#4,RR;M$;
 2440 FORI=1TORR:PRINT#4,K1$(I);M$;RR%(I);M$;:NEXT
 2445 GOSUB1000:IFETHEN3900
 2450 CLOSE4
 2455 GOSUB1000:IFETHEN3900
 2500 PRINT"THE FILE IS FULL, NO ADDITIONAL RECORDS MAY BE ADDED"
 2510 GOTO2160
 3900 PRINTE,EM$:STOP
 4000 FORDI=0TONR:PRINTK1$(DI):NEXT:RETURN
READY.
```

commodore japan limited.

8 14 IKUE 1-CHOE, ASAHI-KU, OSAKA 535 JAPAN
PHONE:(06)922-7781
TELEX: 5298866 COMOKO J

TOKYO OFFICE
5 32 AKASAKA 8-CHOME, MINATO-KU, TOKYO 107 JAPAN
PHONE: (03)479-2131

commodore basic

The fastest full floating-point BASIC implemented on a micro-computer. Allows communication directly from BASIC to IEEE-488 standard devices, cassettes, display, and keyboard built into PET. Accurate built-in clock is settable and readable from BASIC in decimal or string value. Full command set, including:

Standard Dartmouth BASIC Statements

LET READ PRINT DATA IF
THEN FOR NEXT DIM END
GOTO

Extended BASIC Statements

RESTORE REM GET GOSUB DEF
RETURN STOP STEP INPUT FN
ON . . . GOTO ON . . . GOSUB

Scientific Functions

SGN INT ABS SQR RND SIN
COS TAN ATN LOG EXP π

Logical Operators

AND OR NOT

Operation Commands

RUN NEW CLR LIST CONT FRE

Formatting Functions

TAB POS SPC

Machine Level Statements

PEEK POKE
Allow the user to examine and store at specific memory locations.

USR SYS
Link BASIC to machine language subroutines with parameter passing or developmental subsystems.

WAIT
Monitors status of a memory location such as an I/O port until specified bits are set.

String Functions

LEFT$ RIGHT$ MID$
Returns substrings (of specified length and position) of string acted upon.

CHR$ ASC
CHR$ returns a character, given a numeric code.
ASC returns a numeric code corresponding to a character.

LEN
Returns the length of a string.

VAL STR$
Convert decimal values to numeric strings and vice-versa.

Extended I/O Statements

OPEN CLOSE
Control association of a logical file number to a physical device and, optionally, a file name on the device.

SAVE LOAD VERIFY
Store and retrieve a program, with optional file name, on a physical device. Load allows for program overlay, VERIFY compares contents of memory to stored program.

PRINT# INPUT# GET#
Allow communication with logical device numbers other than keyboard or screen. GET# inputs one character.

CMD
Permits communication with multiple devices simultaneously.

Example of I/O Operations

Tape-to-tape file copy

```
10 OPEN 5,1,0,"OLD FILE"
20 OPEN 6,2,1,"NEW FILE"
30 INPUT#5,A$
40 IF ST AND 64 GO TO 70
50 PRINT#6,A$
60 GO TO 30
70 CLOSE 5
80 CLOSE 6
```

Program locates "OLD FILE" on tape #1, writes file header for "NEW FILE" on tape #2, then copies tape #1 to #2 until it encounters an EOF on #1, and then writes an EOF on #2.

Variables

TYPES: Real Integer (%) String ($)
NAMES: Variable names are uniquely given as a letter or a letter followed by a letter or a digit.

Special Variables

TI TI$ Time of day
ST Status word for I/O operations

commodore

Presents with Pride...

the PET Personal Computer

全世界で新世代のパーソナル　コンピュータとして注目されているコモドール　PET　2001（8K）を．日本において発売することになりました。

MODEL 2001

RAM（ユーザー　メモリー　）　8K付

ROM（オペレイティング　システム）　14Kバイト

英文　マニュアル附

価格　298,000 円（先払）
2,000 円（送料）

納期　御注文後　６０日以内

別紙注文書および払込を完了された方より先着順に御注文受書に番号を入れ返送致します。

PET 2001(8K)は、当社にてご覧になれます。

担当　高木

commodore
コモドール・ジャパン株式会社
大阪市旭区生江1丁目8番14号　〒535
電話（06）922－7781番（代表）

注　文　書

昭和　年　月　日

コモドール　ジャパン株式会社　御中

PET 2001 (8K)　＿＿　台を注文します。

代金　298,000円 + 2,000 円（送料）は、富士銀行　天満橋支店

普通預金口座789634 に払込みました。

御名前　　㊞　年令　才

御自宅住所　郵便番号　　電話

御届先

下記項目にもれなく御記入ください。

お勤め先　　電話	
所属	
ご職種 1 経営者 2 研究、開発、教育 3 生産、運営 4 経理、営業 5 学生 6 その他	ご使用の目的

commodore

COMMODORE JAPAN, LIMITED
8-14 IKUE 1-CHOME,
ASAHI-KU OSAKA 535
PHONE: (06) 922-7781

RECORDED CASSETTE TAPE FOR PET-2001

1	SQUIRM	¥1,500.-
2	SQUIGGLE	¥1,500.-
3	BRAIN STRAIN	¥1,700.-
4	HYPNOTIST	¥2,000.-
5	TRIG	¥2,000.-
6	RANDOM MAIZE	¥2,000.-
7	TIC-TAC-TOE	¥2,000.-
8	GRAPHIC	¥2,000.-
9	BIORHYTHM	¥2,500.-
10	LUNAR LANDER	¥2,500.-
11	ROTATE	¥2,500.-
12	GOMOKU	¥2,500.-
13	BLACK JACK	¥2,500.-
14	OSERO	¥3,000.-
15	AMORTIZATION	¥3,000.-

コモドール・ジャパン株式会社
大阪市旭区生江1－8－14
TEL （06） 922－7781

定価表／注文書

分類	商品名	備考	定価	送料	注文数	合計金額
マイクロ・コンピューター	PET-2001	完成品	￥298,000	￥2,000		
シングルボード・コンピューター	KIM-1	完成品	69,800	1,000		
PET-2001用プログラム・カセット						
	SQUIGGLE	ランダム関数プログラム演習用	1,500	0		
	SQUIRM	〃	1,500	0		
	RANDOM MAZE	〃	2,000	0		
	TRIG	ピタゴラス定理教育用	2,000	0		
	HYPNOTIST	催眠術・タイマー応用例	2,000	0		
	GRAPHIC	画面処理演習用	2,000	0		
	BIORHYTHM	バイオリズム	2,500	0		
	BRAIN STRAIN	ゲーム	1,700	0		
	TIC-TAC-TOE	〃	2,000	0		
	LUNAR LANDER	〃	2,500	0		
	ROTATE	〃	2,500	0		
	BLACKJACK	〃	2,500	0		
	OTHELLO	〃	3,000	0		
	AMORTIZATION	経理計算演習応用例	3,000	0		
マニュアル	PET-2001・初級マニュアル(英文)		1,500	0		
合計金額						

●発売予定（時期未定）PET-2001用オプション
ドット・マトリックス・プリンター
RAM拡張用ボード
フロッピー・ディスク
プロッター
セカンド・カセット・デッキ
モデム

●下記項目にもれなく御記入ください。

お勤め先

所属

ご職種 1.経営者 2.研究，開発，教育
3.生産，運営 4.経理，営業
5.学生 6.その他

ご使用の目的

上記を注文致します。代金は，

□富士銀行天満橋支店普通預金口座 789634 へ振込みました。

□現金書留にて送りました。

御氏名 ㊞

〒
御住所 TEL

お届先 〒

御勤務先会社名