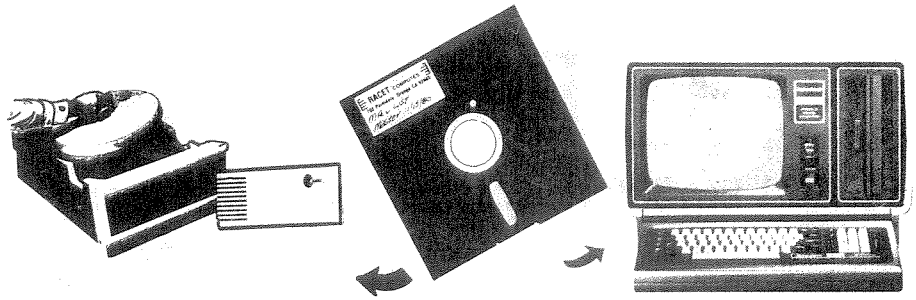
RACET COMPUTES

1330 N. Glassel, Suite 'M', Orange, CA 92665
(714) 637-5016

SPRING CATALOG 1981

HARD DISK MULTIPLEXOR
FOR TRS-80* Mod II



NOW YOU CAN HAVE THAT LARGE COMMON DATA BASE!!

- Allows up to 4 Mod II's to connect to a single controller — up to 4 hard disk drives per controller. Users may access the same file simultaneously (first-come first-served).
- Uses Cameo controller and standard 10-megabyte cartridge (hard) disk drives along with RACET Hard/Soft Disk System (HSD) software. Removable Disk Pack Backup!
- Access times 3 to 8 times faster than floppy. Mixed floppy/hard disk operation supported.
- Compatible with your existing TRSDOS programs — you need only change filenames! All BASIC statements are identical.
- A single file may be as large as one disk. Directory expandable to handle thousands of files.
- Includes special utilities — XCOPY for backup and copies, XPURGE for multiple deletions, DCS directory catalog system, and Hard Disk Superzap. FORMAT utility includes options for specifying sectors/gran, platters/drive, logical disk size, etc.

HARD DISK DRIVE AND CONTROLLER $5995 **RACET HSD Software $400** Call for multiuser pricing. Dealers call for OEM pricing.

★★**NEW**★★ **DISCAT** (32K 1-drive Min) **Mod I, III $50.00**
This comprehensive Diskette Cataloguing/Indexing utility allows the user to keep track of thousands of programs in a categorized library. Machine language program works with all TRSDOS and NEWDOS versions. Files include program names and extensions, program length, diskette numbers, front and back, and diskette free space.

★★**NEW**★★ **KFS-80** (1-drive 32K Min — Mod II 64K) **Mod I, III $100.00; Mod II $175.00**
The keyed file system provides keyed and sequential access to multiple files. Provides the programmer with a powerful disk handling facility for development of data base applications. Binary tree index system provides rapid access to file records.

★★**NEW**★★ **MAILLIST** (1-drive 32K Min — Mod II 64K) **Mod I, III $75.00; Mod II $150.00**
This ISAM-based maillist minimizes disk access times. Four keys — no separate sorting. Supports 9-digit zip code and 3-digit state code. Up to 30 attributes. Mask and query selection. Record access times under 4 seconds!!

★★**NEW**★★ **LPSPOOL** (32K 1-drive Min) **Mod I $75.00**
LPSPOOL — Add multi-tasking to permit concurrent printing while running your application program. The spooler and despooler obtain print jobs from queues maintained by the system as print files are generated. LPSPOOL supports both parallel and serial printers.

BASIC LINK FACILITY 'BLINK' (Mod I Min 32K 1-disk) **Mod I $25.00; Mod II $50.00; Mod III $30.00**
Link from one BASIC program to another saving all variables! The new program can be smaller or larger than the original program in memory. The chained program may either replace the original program, or can be **merged** by statement number. The statement number where the chained program execution is to begin may be specified!

INFINITE BASIC (Mod I & Mod III Tape or Disk) **Mod I $50.00; Mod III $60.00**
Extends Level II BASIC with complete MATRIX functions and 50 more string functions. Includes RACET machine language sorts! Sort 1000 elements in 9 seconds!! Select only functions you want to optimize memory usage.

INFINITE BUSINESS (Requires Infinite BASIC) **Mod I & III $30.00**
Complete printer pagination controls — auto headers, footers, page numbers. Packed decimal arithmetic — 127 digit accuracy +, −, *, /. Binary search of sorted and unsorted arrays. Hash codes.

COMPROC (Mod I & Mod III — Disk only) **Mod I $20.00; Mod III $30.00**
Command Processor. Auto your disk to perform any sequence of instructions that you can give from the keyboard. DIR, FREE, pause, wait for user input, BASIC, No. of FILES and MEM SIZE, RUN program, respond to input statements, BREAK, return to DOS, etc. Includes lowercase driver software, debounce and screenprint!

GSF (Mod I & III Tape or Disk — Specify Memory Size) **Mod I $25.00; Mod II $50.00; Mod III $30.00**
Generalized Subroutine Facilities. The STANDARD against which all other sorts are compared! Machine language — fast and powerful! Multi-key multi-variable and multi-key character string. Zero and move arrays. Mod II includes USR PEEKS and POKES. Includes sample programs.

DSM (Mod I Min 32K 2-drive system. Mod II 64K 1-drive. Mod III Min 32K 1-drive) **Mod I $75.00; Mod II $150.00; Mod III $90.00**
Disk Sort/Merge for RANDOM files. All machine language stand-alone package for sorting speed. Establish sort specification in simple BASIC command File. Execute from DOS. Only operator action to sort is to change diskettes when requested! Handles multiple diskette files! Super fast sort times — improved disk I/O times make this the fastest Disk Sort/Merge available on the TRS.

UTILITY PACKAGE (Mod II 64K) **$150.00**
Important enhancements to the Mod II. The file recovery capabilities alone will pay for the package in even one application! Fully documented in 124 page manual! XHIT, XGAT, XCOPY and SUPERZAP are used to reconstruct or recover data from bad diskettes! XCOPY provides multi-file copies, 'wild-card' mask select, absolute sector mode and other features. SUPERZAP allows examine/change any sector on diskette including track-0, and absolute disk backup/copy with I/O recovery. DCS builds consolidated directories from multiple diskettes into a single display or listing sorted by disk name or file name plus more. Change Disk ID with DISKID. XCREATE preallocates files and sets 'LOF' to end to speed disk accesses. DEBUGII adds single step, trace, subroutine calling, program looping, dynamic disassembly and more!!

BASIC CROSS REFERENCE UTILITY (Mod II 64K) **$50.00**
SEEK and FIND functions for Variables, Line Numbers, Strings, Keywords. 'All' options available for line numbers and variables. Load from BASIC — Call with 'CTRL'R. Output to screen or printer!

DEVELOPMENT PACKAGE (Mod II 64K) **$125.00**
Includes RACET machine language SUPERZAP, Apparat Disassembler, and Model II interface to the Microsoft 'Editor Assembler Plus' software package including uploading services and patches for Disk I/O. Purchase price includes complete copy of Editor Assembler + and documentation for Mod I. Assemble directly into memory, MACRO facility, save all or portions of source to disk, dynamic debug facility (ZBUG), extended editor commands.

CIRCLE READER REQUEST FOR FREE 24-PAGE CATALOG
*TRS-80 IS A TRADEMARK OF TANDY CORPORTION

CHECK, VISA, M/C, C.O.D., PURCHASE ORDER
TELEPHONE ORDERS ACCEPTED (714) 637-5016

RACET COMPUTES
1330 N. GLASSEL, SUITE 'M', ORANGE, CA 92665

★★★★★ COMMAND PROCESSOR "COMPROC" ★★★★★

TRS-80 PROFESSIONAL PROGRAMMING TOOLS GIVE THE TRS-80 LARGE SYSTEM CAPABILITIES

COMPROC extends the DOS-AUTO command to perform MULTIPLE steps either at power-up time or as a single user system command. Running of an application program is now made simple and automatic — even for the inexperienced user.

COMPROC executes a script consisting of a sequence of commands or data (such as VERIFY, DIR, BASIC, LOAD, RUN, data, etc.) This script is easily created, changed and saved using the AUTO/EDIT/SAVE facilities of BASIC.

COMPROC includes many additional features — including:

- Optional self-relocating key-DEBOUNCE routine with screen print capability.
- Software lower-case driver for those with lower case modifications.
- User specified fields may be inserted automatically in the users script when COMPROC is executed.
- Pauses may be inserted to allow the user to enter data at specified locations during execution.
- COMPROC is automatically relocated so as not to conflict with existing BASIC or machine language programs — runs in ANY SIZE TRS-80 system.

COMPROC is used by following the general steps given below:

- The user creates a command file (CMF) containing the list of commands or data to be processed. A very simple CMF is shown in the example below. Note that this is a standard BASIC file containing REM statements (the ' is an abbreviation for REM).
- The user initiates COMPROC by entering a standard DOS command as shown in the example.
- COMPROC scans the CMF, extracts information, and inserts optional fields to form a compressed control program.
- The compressed control program is relocated to a temporarily reserved area of memory and passes the desired information to the system as if the user had entered it directly from the keyboard.
- After all statements have been processed COMPROC will release any reserved area and return control to the keyboard.

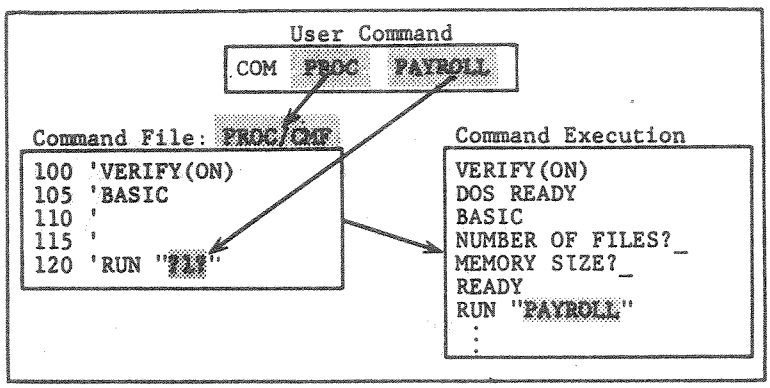


Figure (1). Simple COMPROC Example.

Note that in the above example the user command could be set with the AUTO command as follows:
AUTO COM PROC
This would execute the above CMF script at power-up time without any other user action required.

ORDER COMPROC MOD I $20.00; MOD III $30.00

PROFESSIONAL SOFTWARE FROM RACET COMPUTES

*** REMODEL * PROLOAD * COPSYS * TIMSER * Y-YBAR * MDA ***

MODEL I TRS-80 PROFESSIONAL PROGRAMMING TOOLS YOU HAVE BEEN WAITING FOR

REMODEL + PROLOAD is the "Cadillac" of Renumbering Programs

—ALSO WORKS WITH DOS—

- RENUMBER any portion or all of a basic program with line number references adjusted.
- MOVE any portion of a basic program from one location to another.
- DELETE lines or ranges of lines while using the utility.
- MERGE all or any portion of a basic program from tape (load line 300-500 from your tape into existing program starting at line 1000 in steps of 5) with renumbering on the way in.
- SAVE combined/merged programs or any portion of a program to tape.
- VERIFY bit-for-bit the contents of saved programs.

REMODEL eliminates manual restructuring and renumbering of program lines. This allows the user to rapidly create working and well organized programs. Blocks of lines may be moved from one location to another thus enhancing program readibility and ease of documentation.

PROLOAD allows creation of a library of BASIC programs which can then be loaded individually or in combination with other programs. This greatly enhances the users programming productivity by avoiding re-creation of commonly used routines. Also, a library of DATA statements can be selectively saved and loaded, which is a more effective method of storing and retrieving fixed data than using data tape facilities. Complete with Users Manuals - specify 16, 32, or 48K versions.

ORDER 16, 32 OR 48K VERSION OF REMODEL + PROLOAD AT $35.00

COPSYS is designed to copy system tapes (machine language object code) generated from the Radio Shack Editor/Assembler program. COPSYS can be used to duplicate those valuable system tapes before they become damaged.

ORDER COPSYS MOD I $20.00; MOD III $30.00

WITH COPSYS YOU CAN:

- COPY SYSTEM (machine language object) tapes rapidly and accurately.
- MERGE independently assembled object tapes into one load module.
- VERIFY bit-for-bit system tapes duplicated by COPSYS.

TIMSER is a Time Series Analysis program that fits data to nine different 1st, 2nd, and 3rd order curves. Goodness of fit data, tables, including projected data and confidence limits, and graphical displays of curve fit and variance are included. The program provides for correction factors and functions to test seasonal variations, inflation corrections, or cyclical variations. A detailed Users Manual is included complete with illustrated examples.

ORDER **TIMSER** MOD-I AT $15.00
MOD-III AT $30.00

Y-YBAR is an optical system design program that implements the latest Y-YBAR diagram techniques allowing manipulation of ray heights at lens surfaces. User documentation is provided.

ORDER **Y-YBAR** MOD-I AT $15.00
MOD-III AT $30.00

MINI-DISK DRIVE ALIGNMENT PROGRAM (MDA) allows technicians who are not familiar with mini disk drive alignment procedures to align disk drives using simple step-by-step procedures. Alignment includes: 1) disk head radial alignment, 2) index (sector 0) sensor adjustment, and 3) motor speed adjustment. The program utilizes a TRS-80 Model I computer with expansion interface to simulate a mini disk tester. An alignment diskette is included in the purchase. The only other equipment required is a dual trace oscilloscope and common hand tools.

MODEL I MINIMUM 16K 1-DISK

ORDER MDA AT $109.00

REMODEL EXAMPLES

The examples below illustrate some of the capabilities of REMODEL. A trivial sample program is used to show the status before and after the specified action is completed. User input is underlined and modified areas are shaded.

1. REQUIREMENT: Renumber the entire program using a starting number of 10 and incrementing by 10.

```
User: START?      END?      NEW?10    BY?       MODE?
Used: START?5     END?125   NEW?10    BY?10     MODE?RES
```

PROGRAM BEFORE

```
5   PRINT 1: GOTO 24
24  PRINT 2: GOTO 125
25  PRINT 3: GOTO 63
50  PRINT 4: END
63  PRINT 5: GOTO 50
125 PRINT 6: GOTO 25
```

PROGRAM AFTER

```
10  PRINT 1: GOTO 20
20  PRINT 2: GOTO 60
30  PRINT 3: GOTO 50
40  PRINT 4: END
50  PRINT 5: GOTO 40
60  PRINT 6: GOTO 30
```

COMMENTS: Note that default values were supplied automatically for the fields left blank.

2. REQUIREMENT: Move lines #50-63 between lines #5 and #24, renumbering in increments of 10 starting at 10.

```
User: START?50    END?100   NEW?10    BY?       MODE?
Used: START?50    END?63    NEW?10    BY?10     MODE?MOV
```

PROGRAM BEFORE

```
5   PRINT 1: GOTO 24
24  PRINT 2: GOTO 125
25  PRINT 3: GOTO 63
50  PRINT 4: END
63  PRINT 5: GOTO 50
125 PRINT 6: GOTO 25
```

PROGRAM AFTER

```
5   PRINT 1: GOTO 24
10  PRINT 4: END
20  PRINT 5: GOTO 10
24  PRINT 2: GOTO 125
25  PRINT 3: GOTO 20
125 PRINT 6: GOTO 25
```

Comments: Blocks may be moved to any new location as long as there is sufficient room between sequence numbers, as above.

3. REQUIREMENT: Move lines #50-63 between lines #24 and #25.

```
User: START?25    END?25    NEW?40    BY?       MODE?
Used: START?25    END?25    NEW?40    BY?10     MODE?RES

User: START?50    END?63    NEW?30    BY?5      MODE?
Used: START?50    END?63    NEW?30    BY?5      MODE?MOV
```

PROGRAM BEFORE

```
5   PRINT 1: GOTO 24
24  PRINT 2: GOTO 125
25  PRINT 3: GOTO 63
50  PRINT 4: END
63  PRINT 5: GOTO 50
125 PRINT 6: GOTO 25
```

PROGRAM AFTER

```
5   PRINT 1: GOTO 24
24  PRINT 2: GOTO 125
30  PRINT 4: END
35  PRINT 5: GOTO 30
40  PRINT 3: GOTO 35
125 PRINT 6: GOTO 40
```

COMMENTS: Two steps were required in the above example. The first renumbered line #25 to #40, and the second moved lines #50-63 between #24 and #40 producing the desired result.

REMODEL CAN BE USED IN DOS OR NON-DOS ENVIRONMENT

PROLOAD EXAMPLES

PROLOAD expands the capabilities of REMODEL as illustrated in the examples shown below. For these examples it is assumed that the following two program are on tape and positioned correctly when needed:

PROGRAM "A"

```
10  PRINT 1: GOTO 30
15  PRINT 2: END
30  PRINT 3: GOTO 15
```

PROGRAM "B"

```
25  PRINT 4: GOTO 35
30  PRINT 5: STOP
35  PRINT 6: GOTO 30
```

User input is underlined and areas loaded or saved to tape are shaded.

1. REQUIREMENT: Load program "A" into memory.

```
User: START?      END?      NEW?      BY?       MODE?"A"
Used: START?0     END?65535 NEW?      BY?       MODE?"A"
```

MEMORY BEFORE

MEMORY AFTER

```
10  PRINT 1: GOTO 30
15  PRINT 2: END
30  PRINT 3: GOTO 15
```

COMMENTS: Default values were supplied for the START and END fields. The NEW and BY fields were not used, as a result the statements were not renumbered.

2. REQUIREMENT: Load statements #30 and #35 of program "B" between lines #15 and #30 of program "A", renumbering to #20 and #25.

```
User: START?30    END?35    NEW?20    BY?5      MODE?"B"
Used: START?30    END?35    NEW?20    BY?5      MODE?"B"
```

MEMORY BEFORE

```
10  PRINT 1: GOTO 30
15  PRINT 2: END
30  PRINT 3: GOTO 15
```

MEMORY AFTER

```
10  PRINT 1: GOTO 30
15  PRINT 2: END
20  PRINT 5: STOP
25  PRINT 6: GOTO 20
30  PRINT 3: GOTO 15
```

COMMENTS: This operation was successful because there was room between statements # 15 and #30 for the two lines desired.

3. REQUIREMENT: Save the contents of memory which contains program "A" from lines #15 to #30 to tape, renaming to program "C"

```
User: START?15    END?30    NEW?      BY?       MODE?(C)
Used: START?15    END?30    NEW?      BY?       MODE?(C)
```

MEMORY BEFORE

```
10  PRINT 1: GOTO 30
15  PRINT 2: END
30  PRINT 3: GOTO 15
```

PROGRAM "C" TAPE

```
15  PRINT 2: END
30  PRINT 3: GOTO 15
```

COMMENTS: Note that the NEW and BY fields were not used. Any range of lines can be saved to tape. Tapes written can be verified by using the =C= mode as for the example above.

PROLOAD CAN BE USED IN A DOS OR NON-DOS ENVIRONMENT

∞ INFINITE BASIC ∞™
FOR THE TRS-80 LEVEL-II TAPE AND DISK SYSTEMS

∞ INFINITE BASIC EXPANDS THE POWER OF YOUR TRS-80

Over 80 features have been added to the existing Radio Shack BASIC system. Now, a single BASIC statement can perform such complex operations as:

SORTING - STRING CENTERING / ROTATION / TRUNCATION / JUSTIFICATION - DATA COMPRESSION - STRING TRANSLATION/ COPYING - DISPLAY SCREEN MANIPULATION

MATRIX INVERSION / SIMULTANEOUS EQUATIONS - DYNAMIC ARRAY RESHAPING / DELETION - BASIC SUBROUTINE CALL FACILITY WITH ARGUMENTS - MATRIX INPUT / OUTPUT.

INFINITE BASIC will operate with either **TAPE** or **DISK** oriented systems

∞ INFINITE BASIC INCREASES THE SPEED OF YOUR APPLICATIONS

The use of INFINITE BASIC will often dramatically increase the speed of your applications. For example, the SORT function is **30-50** times **FASTER** than possible when written in BASIC. In addition, having a large selection of complex functions readily available will increase your programming productivity.

∞ INFINITE BASIC OPTIMIZES THE USE OF MEMORY

The user selects only those functions required for loading into memory. A loader program is included with INFINITE BASIC which automatically builds a load module of functions selected by the user.

The load module can be **RELOCATED** anywhere within protected memory. This provides for compatibility with existing machine language user programs.

∞ INFINITE BASIC PROVIDES A MODULAR APPROACH TO YOUR COMPUTING REQUIREMENTS

INFINITE BASIC is a MODULAR system providing a wide range of capabilities. **TWO** application modules are contained in the core package distributed - the **STRING** and **MATRIX** modules. The contents of these modules are summarized in the attached list of INFINITE BASIC FUNCTIONS.

INFINITE BASIC provides a path for upward mobility. Additional modules will be available to further expand the power and ease of use of the TRS-80.

Currently available is the BUSINESS module. This module provides the following facilities which are especially useful in business applications:

Automatic printer pagination header/footer system. This allows the user to specify heading and footing lines for output page printers. The user just prints the data desired, INFINITE BASIC will automatically print the specified headers, footers, along with page numbers if requested.

Powerful search and insert commands. This allows the user to rapidly locate an element in either a sorted or unsorted array, or to insert an element into a sorted array.

Packed decimal floating point functions. This package of functions provides increased precision as well as reduces round-off errors required for business applications. Arithmetic operations may be performed on numbers up to 500 significant digits of precision.

Hash number encoding function. This function provides a convenient method for generating disk index numbers for a hash encoded information retrieval system.

Additional modules will be available soon. All add-on modules require prior purchase of INFINITE BASIC which contains the loader.

∞ EXAMPLE USE OF INFINITE BASIC

The short example below illustrates one use of INFINITE BASIC. In this example four arrays are generated (NM$, SX$, IG, and WT), sorted, and printed. The use of the INFINITE BASIC functions SRV$, SRTV, and STC$ are explained below:

Line #	Description
30	SRV$ generates a string of random characters. In this case the length of the string is controlled by RND(10), i.e. a random length string of 1 to 10 characters will be generated. The characters generated will be randomly selected from the range of 65-91 (ASCII for letters A-Z).
40	SRV$ again is used to generate a fixed length string containing only one character (F or M).
80	SRTV$ is used to sort elements 2-4 of the array. The first argument specifies which arrays are to be connected in the sort, and the sort order. In this case SX$ is the primary sort variable with the "+" sign indicating ascending sort order. The secondary sort key will be IG, with the "–" sign indicating descending sort order. Similarly WT will be the third key in ascending order. NM$, containing no "+" or "–" sign will be carried along in the sort. The last two arguments specify that only elements 2-4 are to be included in the sort.
100	STC$ is used to center the character string in a field 20 characters wide for printing.

The sample BASIC program is shown below. Note that each INFINITE BASIC FUNCTION is preceeded by an "&".

```
 1  CLEAR 1000: DEFUSR = &HEFAC: PRINT USR(1)
10  DIM NM$(7),SX$(7),IG(7),WT(7)
20  FOR I = 0 TO 5
30  NM$(I) = &SRV$(RND(10),65,91)
40  SX$(I) = &SRV$(1,"FM")
50  IG(I) = RND(100)
60  WT(I) = RND(0)*300.0
70  NEXT
80  IC = &SRTV(" + SX$, – IG, + WT,NM$",2,4)
90  FOR I = 2 TO 4
100 PRINT &STC$(NM$(I),20);SX$(I);IG(I);WT(I)
110 NEXT
```

To use INFINITE BASIC, as in the above example, the required functions must first be loaded into memory. Given below is the load procedure required to create the three functions used above. Note that each function selected below is preceeded by two "@@" characters. User input is shown underlined.

IBLOAD

```
IBLOAD - DISK VERSION 2.0 - COPYRIGHT 1979, RACET COMPUTES
ENTER SUBROUTINE NAMES REQUIRED? @@SRTV
? @@SRV$
? @@STC$
?
HIGH/LOW MEMORY ALLOCATION(H/L)? H
ENTER STARTING ADDRESS? FA00H
DISK/TAPE INPUT (D/T)? D
ENTER INPUT DISK FILESPEC? SREL
FOLLOWING SUBROUTINE(S) NOT YET FOUND
@AESYA @ARGN @ARGV @CPTH @TA$ @ACMOD @AT @BALS
@RETS @SKIP @T$
ENTER INPUT DISK FILESPEC? XREL
MEMORY START = X'EF9A',END = X'FA00',TRA = X'402D'
  DEFUSR = &HEFAC
DUMP MEMORY TO TAPE (Y/N)? N
XE  PROCESSING COMPLETED
DOS READY
```

STRING FUNCTIONS

- **String Sort Functions.**

SRTC Character String Multi-key Sort.
SRTV Multi-variable Sort.

- **String Manipulation Functions.**

SBJ$ Justify Both Sides of String.
SDS$ Delete substring.
SIV$ Invert (reverse) string.
SLJ$ Left justify string.
SLR$ Left rotate string.
SLS$ Left shift string.
SLT$ Left truncate string.
SRR$ Right rotate string.
SRT$ Right truncate string.
SRV$ Generate random string.
SSI$ Insert substring.
STC$ Center string text.
STJ$ Justify string text.
STP$ Pack string text.

- **String Data Compression/Decompression.**

SC4$ Compress bytes to 4-bit format.
SC5$ Compress bytes to 5-bit format.
SC6$ Compress bytes to 6-bit format.
SC7$ Compress bytes to 7-bit format.
SCL$ Convert to lower case.
SCU$ Convert to upper case.
SD4$ Decompress 4-bit format to bytes.
SD6$ Decompress 5-bit format to bytes.
SD6$ Decompress 6-bit format to bytes.
SD7$ Decompress 7-bit format to bytes.
SCP$ Compress string by removing multiple characters.
SDP$ Decompress string packed by SCP$.
SCPM Compress memory bytes by removing multiple characters.
SDPM Decompress memory packed by SCPM.

- **Display Screen Functions.**

SDHL Draw Horizontal line.
SDVL Draw Vertical line.
SEHL Erase horizontal line.
SEVL Erase vertical line.
SSCL Scroll screen left.
SSCR Scroll screen right.
SSDN Scroll screen down.
SSUP Scroll screen up.

- **String Translation/search/copy.**

STL$ String translate function.
SVER String verify function.
SMSK String search under mask.
SCPY String matrix copy (in order by index).
SEQU String matrix copy (in sequential order with source repeated).
SABP Create absolute string pointer.
SAP$ Propogate byte consecutively in memory.
SVP$ Create String Value.

MATRIX FUNCTIONS

- **Matrix Mathematical Functions.**

MEQN Solution of simultaneous linear equations.
MINV Matrix inversion.
MIDT Matrix identity.
MMPY Matrix multiply.
MTRN Matrix transpose.

- **Matrix scalar arithmetic functions.**

MSAD Add scalar to array.
MSMP Multiply array by scalar.
MSDV Divide array elements by scalar.
MSSD Subtract scalar from array elements.

MATRIX FUNCTIONS (*cont.*)

- **Matrix array arithmetic functions.**
(Operations performed in order by corresponding index values)

MAAD Add array elements.
MADV Divide array elements.
MAMP Multiply array elements.
MASB Subtract array elements.
MCPY Copy array elements.

- **Matrix array arithmetic functions.**
(Operations performed sequentially, with repeating source array elements)

MELA Add array elements.
MELC Copy array elements.
MELD Divide array elements.
MELM Multiply array elements.
MELS Subtract array elements.

- **Matrix input/output functions.**

MELR Matrix element read (corresponds to BASIC READ statement)
MRDT Matrix read tape.
MRST Restore READ DATA statement number location.
MWRT Matrix write tape.

- **Subroutine Call functions.**

MCAL Subroutine call - passes arguments to BASIC subroutine.
MRTN Subroutine return - returns arguments from BASIC subroutine.

- **General purpose functions.**

MSHP Dynamically reshapes or deletes arrays.
MEQU Copies arrays (sequentially without repeating).
PLUK Pluck (Peek) a word from memory.
PLUG Plug (Poke) a word into memory.
NOIB Deactivate INFINITE BASIC.

BUSINESS FUNCTIONS

- **Search, Insert, and Hash Functions.**

BHSH Hash number generation.
BINS Insert string into sorted array.
BNUL Set null character compare byte.
BSSA Search sorted string array for matching element.
BSUA Search unsorted string array for matching element.

- **Printer Functions.**

BCLN Find current printer line number.
BCPN Find current page number.
BHDR Define/activate page headings and footings.
BHDX Deactivate pagination system.
BPGN Initialize pagination.
BRLN Reset current line number.
BVTB Tab to a given line number.

- **Packed Decimal Functions.**

BAP$ Add packed decimal.
BCP$ Convert to packed decimal.
BDP$ Divide packed decimal.
BMP$ Multiply packed decimal.
BPC$ Convert from packed decimal to character.
BPRC Set packed decimal floating point precision.
BSP$ Subtract packed decimal.

GENERALIZED SUBROUTINE FACILITY "GSF" FOR MODEL I

- A SYSTEM for incorporating multiple machine language programs in a unified structure that provides easy access by TRS-80 BASIC users. Easily expanded by the user to include additional functions. Instructions furnished describing conventions used.
- GSF INCLUDES eighteen machine language subroutines providing the TRS-80 with much greater flexibility and extended capabilities.

SORT 1000-ELEMENT ARRAYS IN 9 SECONDS!!! Two in-core sort subroutines are provided for sorting data in memory. The first subroutine sorts records consisting of corresponding elements of up to 15 arrays (mixed string, floating point, and integer) using multiple ascending or descending sort keys. The second subroutine sorts records consisting of elements of a character string array using multiple substrings as ascending or descending sort keys. These sort systems are very fast, versatile, and easy to use. Sort times average 30 TIMES faster than the fastest BASIC sort routines.

READ AND WRITE ARRAYS TO TAPE. Two subroutines are provided to read and write data to cassette tape. This provides the user with the capability of reading or writing an entire array or screen image with one command. No leaders are written between data items. This significantly reduces read/write times making data tapes much more practical. Data validity checking is performed to ensure correct data is read, thus eliminating many data input/output errors.

COMPRESS AND UNCOMPRESS DATA. Two subroutines are provided which compress data in memory by removing repeated characters with the ability to uncompress the data in the minimum space possible with subsequent regeneration. This, coupled with the Read and Write Tape Data Subroutines, provides an efficient method of data storage and retrieval.

MOVE ARRAYS IN MEMORY. This subroutine moves data from one location in memory to another. This can be used to rapidly set one array of data equal to another or to move data into protected memory. The latter option provides a "common" area that can be passed from one BASIC program to another.

DUPLICATE MEMORY. Two subroutines are provided to duplicate a byte in memory. This is useful for setting arrays to zero or rapidly placing rows or columns of repeated characters on the screen.

FAST HORIZONTAL AND VERTICAL LINES. This two subroutines draw graphic lines of any length and location on the screen. These routines dramatically decrease times necessary to generate graphical displays containing lines.

DISPLAY SCREEN CONTROL. Five subroutines are provided for scrolling the screen up, down, left, right, and for generating inverse graphic displays. This can add impact to screen displays.

ORDER 16, 32, OR 48K VERSION OF GSF MOD I $25.00; MOD III $30.00

FOR THE SERIOUS DISK SYSTEM USER - DISK OPERATING SYSTEM SORT MERGE "DOSORT"

SORT/MERGE multi-diskette sequential files. DOSORT utilizes a control program written in BASIC with very efficient machine language routines for the time-critical tasks of sorting and data comparisons. An extended GSF is included which is used as an interface between the BASIC and machine language programs. The resulting system is both a versatile and effective sort merge package for the TRS-80 system.

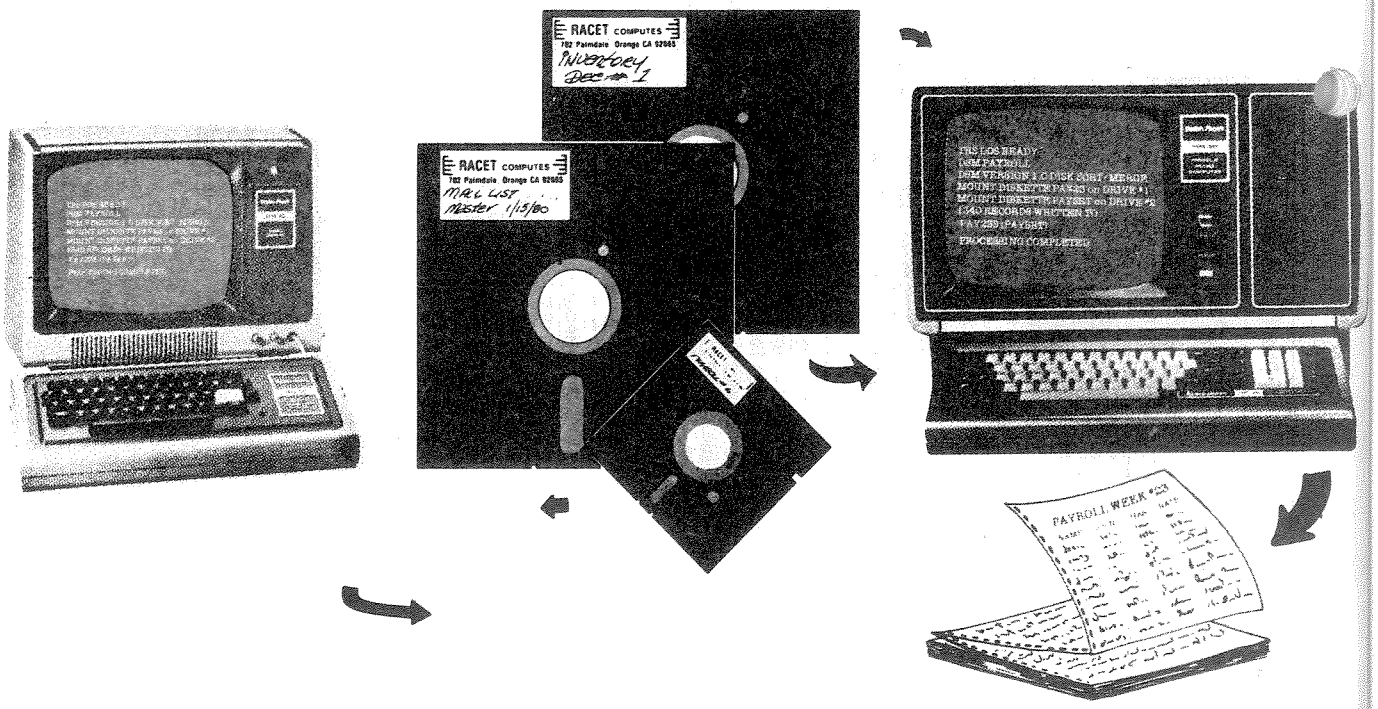
- DOSORT can process data files that are read or written using standard TRS-80 input/output commands.
- DOSORT sorts input files accouding to user specified sequences utilizing multiple ascending or descending sort keys. User exits are provided during initial input and final output processing allowing specialized processing.
- TWO MODES of sort variable specification may be used - multi-mixed-array mode and character string mode.
- MULTI-VOLUME (diskette) files may be sorted or merged on a minimum two-disk 32K system. System performance is improved with a 48K or three- or four-disk system.
- PRESORTED files may be included in a given sort/merge application. Data files may be spread over several diskettes.
- DOSORT is self-instructing - supported by comprehensive user documentation. Specification of sort parameters has been designed to be easy - even for the inexperienced user.

ORDER 32 OR 48K VERSION OF DOSORT MOD I $35.00; MOD III $45.00

GSF SUBROUTINE SUMMARY

GSF#	Page#	Arg#	Mode	Description
1	5			Invert Graphic Video
		1	Address	Graphic data to be inverted
		2	Integer	Number of bytes to be inverted.
			Return Value	0
2	15			Read Tape Data Block
		1	Address	Location where data to be placed.
		2	Integer	Maximum number of bytes to be read.
		3	Integer	Tape Block ID number.
			Return Value	-1 Tape block ID does not match. -2 Data read error. >0 Number of bytes read.
3	15			Write Tape Data Block
		1	Address	Location of data to be written.
		2	Integer	Number of bytes to be written.
		3	Integer	Tape Block ID number.
			Return Value	Address last byte written +1.
4	9			Duplicate Memory Serially
		1	Address	Location of start of data.
		2	Integer	Number of bytes to be duplicated.
			Return Value	Address last byte duplicated.
5	5			Scroll Screen Up
		None		
			Return Value	0
6	5			Scroll Screen Down
		None		
			Return Value	0
7	5			Scroll Screen Left
		None		
			Return Value	0
8	5			Scroll Screen Right
		None		
			Return Value	0
9	9			Duplicate Memory Incrementally by 64
		1	Address	Location of start of data.
		2	Integer	Number of bytes to be duplicated.
			Return Value	Address last byte duplicated.
10	13			Compress Data
		1	Address	Data to be compressed.
		2	Address	Where compressed data is to be placed.
		3	Integer	Number of bytes to be compressed
			Return Value	Number of bytes in compressed area.
11	13			Uncompress Data
		1	Address	Data to be uncompressed.
		2	Address	Where uncompressed data to be placed.
			Return Value	Number of bytes in uncompressed area.
12	7			Draw Vertical Line
		1	Integer	Row number for vertical line.
		2	Integer	Column number for vertical line.
		3	Integer	Length of vertical line.
			Return Value	0
13	7			Draw Horizontal Line
		1	Integer	Row number for horizontal line.
		2	Integer	Column number for vertical line.
		3	Integer	Length of vertical line.
			Return Value	0
14	11			Move Data
		1	Address	Location of data to be moved.
		2	Address	Location where data is to be moved.
		3	Integer	Number of bytes of data to be moved.
			Return Value	Address last byte +1 (Arg#3+Arg#1).
15				Fetch GSF Argument
		1	Integer	GSF argument # to be fetched.
			Return Value	Integer argument saved by GSF.
16				Fetch Memory Word
		1	Address	Address of memory location fetched.
			Return Value	Integer value at memory location.
17	17			In-Core Sort - Multiple Variable Mode
		1	Address	Pointer to sort key string.
		2	Integer	Start index for sort.
		3	Integer	End index for sort.
			Return Value	0 Sort completed successfully. 1 Null Argument #1. 2 Missing variable. 3 Array specified not found. 4 Array found not single dimension. 5 Array too small.
18	17			In-Core Sort - Character String Mode
		1	Address	Pointer to array to be sorted.
		2	Integer	Start index for sort.
		3	Integer	End index for sort.
		4	Address	Sort key parameter list.
			Return Value	0 Sort completed successfully. 1 Argument #4 array not integer. 2 Argument #4 array multi-dimension. 3 No substrings specified. 4 Substring location 0 specified.

RANDOM FILE DISK SORT/MERGE FOR THE TRS MOD-I/II



★ ★ ★ FAST ★ ★ ★ LARGE SYSTEM FEATURES ★ ★ ★ FAST ★ ★ ★

SORT AN 85K DISKETTE IN LESS THAN THREE MINUTES!

RECORDS PHYSICALLY SORTED NO KEY-FILES REQUIRED!

MACHINE LANGUAGE STAND-ALONE SYSTEM

Now the TRS-80 user has available the professional tool required for BUSINESS applications. Perfect for those LARGE mailing lists, inventory control, or report generation systems.

DSM IS POWERFUL!!!!!!!

- Sorts large multiple diskette files on a minimum one drive Mod-II or two drive Mod-I disk system.
- All records are physically rearranged — no key files are required.
- Sorts random files created by BASIC, including files containing sub-records spanning sectors.
- Sorts on one or more fields in ascending or descending order. Fields may be character, binary integer, or floating point.
- The sorted output file may optionally have fields deleted, rearranged, or padded.
- Sort commands can be saved for reuse in production applications.
- Single sort, merge, or mixed sort/merge operations may be performed in a single DSM application.
- Sorted output may be written to a new file, or replace the original input file.

DSM IS FAST!!!!!!

DSM is written entirely in machine language for fast sorting. The sorting operations use very efficient techniques. Internal operations are optimized to allow running at maximum disk I/O speeds. The sort timings shown below illustrate the speed of DSM:

Type	File Size (Byte)	Time (Sec)
SORT	16K	33
SORT	32K	49
SORT	85K	173
SORT	170K	445

Type	File Size (Bytes)	Time (Sec)
SORT	340K	1081
SORT	680K	2569
SORT + MERGE	85K 1275K	1757

Times will vary based on sort and system configurations. The sort times shown above are for a Mod-I 48K 4-drive configuration, 64 byte records, and 5 sort keys. Model-II sort times are TWICE AS FAST!

DSM IS EASY TO USE!!!!!

DSM requires only a few simple commands for a given sort application. The sort commands may be saved for production applications, requiring only a single DOS command for execution.

The sort commands are entered in any order in a free format. Default values are provided for user convenience. The example below illustrates sorting a typical mailing list in order by last name, first name, and ZIP code.

```
10' FILEIN     LRECL=64,TYPE=FB
20' SORTIN     INMAIL(DISK1)
30' FIELD      LNAME(CH,20,11),FNAME(CH,10,1),ZIP(BI,2,31)
40' UNIT       1,2,3
50' WORK       DISKA,DISKB
60' SORTOUT    OUTMAIL(DISK2)
```

The "FILEIN" statement describes the length and type of records (FB = fixed blocked). The "SORTIN" statement identifies the input filespec ("INMAIL" on diskette "DISK1"). The "FIELD" statement specifies the name, type, length, and starting column number of each field (CH = character, BI = binary integer). All other unspecified fields would be automatically carried along in the sort. The "UNIT" and "WORK" statements specify the disk drive and work diskettes available for use by DSM. The "SORTOUT" statement specifies the output filespec for the sorted records.

The application above could be changed to sort in descending order by ZIP code by adding the following command:

```
70' -ZIP,LNAME,FNAME
```

Adding the following statements would keep only the fields LNAME and ZIP, deleting FNAME and other unspecified fields from the sorted output file:

```
80' KEEP       LNAME,ZIP
90' FILEOUT    LRECL=32
```

Note in the above example the resulting sub-records would be only 32 bytes in length.

DSM IS FLEXIBLE!!!!!

The many powerful features of DSM exceed those commonly found on even the largest computer systems. Additional features include:

- Ability to specify multiple file input from the same or different diskettes. For example LINE #20 above could be written: 20' SORTIN IN1(DISK1),IN2(DISK2)
- Number of input records can be specified. For example, Line #20 above could be written: 20' SORTIN INMAIL(DISK1,1247).
- The "ORDER" option may be used to retain input sequence for identical records.
- The "PRINT" and "MSGLEVEL" options may be used to print and/or display DSM messages.
- The "MEMORY" option allows the user to protect upper memory for other purposes (such as RS232 drivers).
- The "INEXIT" and "OUTEXIT" options allow the user to use machine language subroutines to intercept and control input and final sorted output records.

ORDER DSM FOR MOD I $75.00; MOD II $150.00; MOD III $90.00

GSF MACHINE LANGUAGE SORTS FOR THE TRS MOD II

A Generalized Subroutine Facility (GSF) is available now for the TRS Mod II. Machine Language functions available through USR calls include:

- Multi-key Multivariable In-Memory Sort.
- Multi-key Character String In-Memory Sort.
- USR PEEK and POKE capability; both byte and word. Fetch argument.
- Compress and Uncompress data.
- Move blocks of data.
- Propogate across arrays.

The sorts are extremely fast and flexible. Sort 1000 elements in six seconds! 5000 elements in 26 seconds! Carry up to 15 arrays together with multiple mixed ascending/descending keys! Sort on multiple columns in character string sort mode. One line call from BASIC.

ORDER GSF FOR THE MOD-II AT $50.00

UTILITY PACKAGE

For The
TRS-80* MODEL-II SYSTEM

XCOPY SZAP XHIT XGAT DCS XCREATE DISKID DEBUG-II

EXTEND THE POWER OF YOUR TRS-80 MODEL-II SYSTEM WITH THESE COMMANDS!

This Utility Package extends the power of the Model-II DOS system by providing a number of very useful commands. These commands can be used in addition to the COPY, DIR, LIST, and other functions available in the standard DOS system.

Each of the new commands is fully documented in a 128-page manual. In addition, a large section has been included on the **Recovery of Blown Files.** This section will be found to be very valuable if a diskette has a directory or other error caused by a system, user, or hardware problem. Detailed information is provided on how files are stored on a diskette, the recovery techniques that can be tried, along with a number of examples.

A summary of the new commands available and their capabilities is given below:

XCOPY — This command, similar to the COPY command, is used for copying files. However, a number of additional capabilities have been provided, including: copying more than one file at a time, selecting groups of files to copy based on a "wild card" mask, easy to use single disk copies, input/output and directory error recovery copies, absolute sector copy, and many others.

SZAP — This utility provides the capability to read and modify any sector on a diskette. In addition, it provides a generalized facility for copying any number of sectors from one area (or disk) to another. This includes recovery from input/output errors, allowing backup of diskette data not possible with the BACKUP command.

XHIT — The purpose of XHIT is to provide a tool for the analysis and correction of the Hash Index Table, which is part of a diskette directory. This is very useful in the recovery of blown files.

XGAT — The purpose of XGAT is to provide a tool for the analysis of diskette directories. In particular, this program analyzes the Granule Allocation Table and associated directory entries for accuracy. This utility is also designed to be used in the recovery of blown files.

DCS — The Directory Catalog System is a utility for the management of users diskettes. Up to 1200 individual filenames and associated information from multiple diskettes may be combined, sorted, and selectively listed or printed. The composite directories may also be saved on disk for later processing by DCS or a users BASIC program.

XCREATE — The XCREATE command is used to preallocate and initialize a file by writing an end-of-file record to the file. A BASIC "LOF" function will then return the correct ending record number.

DISKID — The purpose of the DISKID program is to allow the user to change the disk identification name of a diskette (set formerly only during FORMAT or BACKUP). This function is useful in maintaining a library of diskettes by allowing the user at any time to assign each diskette a unique name.

DEBUG-II — DEBUG-II replaces the DEBUG system provided with the standard DOS. It incorporates a number of features not found in the DEBUG processor including: single instruction cycle, auto (Loop) breakpoints, subroutine calling, break-key detection, and many others. These functions greatly simplify the process of debugging assembly language programs.

The user will find that the capabilities of these new commands, and the information provided in the detailed manual will greatly expand the usefulness of the Model-II DOS system.

UTILITY PACKAGE EXAMPLES

Given below are representative examples of the usefulness of the new commands provided by the utility package.

XCOPY

1. XCOPY is used for copying files. This example illustrates that XCOPY can be used to copy several files in one step. Also note that the copy was forced (by the {0} option) to use only drive #0. The user was prompted to mount the source and destination diskettes at the appropriate time.

```
XCOPY {0} FA TO FA, FB TO FB, FABC TO FABC
MOUNT SOURCE DISK=          ON DRIVE=0
    - PRESS ANY KEY ?
READING SOURCE FILE -->FA:0(DISKA)
READING SOURCE FILE -->FB:0(DISKA)
READING SOURCE FILE -->FABC:0(DISKA)
MOUNT DESTINATION DISK=          ON DRIVE=0
    - PRESS ANY KEY?
**** WRITING DESTINATION FILE ===> FA:0(DISKB)
**** WRITING DESTINATION FILE ===> FB:0(DISKB)
**** WRITING DESTINATION FILE ===> FABC:0(DISKB)
```

This results in the following file allocation:

BEFORE DISKB	DISKA	AFTER DISKB
	FA	FA
	FB	FB
	FABC	FABC
	FA/EX1	
	FB/EX1	
	FABC/EX1	
	FA/EX2	
	FB/EX2	
	FABC/EX2	
GA	GA	GA
GB	GB	GB
GABC	GABC	GABC

2. This example shows how easy it is to copy a group of files. In this case all files starting with the letter "F" will be copied. The option {0,ABS} forces the copy on one drive, and copies even if the file is already on the destination diskette.

```
XCOPY {0,ABS} F?/? TO ?/?
MOUNT DESTINATION DISK=          ON DRIVE=0
    - PRESS ANY KEY ?
MOUNT SOURCE DISK=          ON DRIVE=0
    - PRESS ANY KEY ?
READING SOURCE FILE -->FA:0(DISKA)
READING SOURCE FILE -->FB:0(DISKA)
READING SOURCE FILE -->FABC:0(DISKA)
READING SPURCE FILE -->FA/EX1:0(DISKA)
READING SOURCE FILE -->FB/EX1:0(DISKA)
READING SOURCE FILE -->FABC/EX1:0(DISKA)
READING SOURCE FILE -->FA/EX2:0(DISKA)
READING SOURCE FILE -->FB/EX2:0(DISKA)
READING SOURCE FILE -->FABC/EX2:0(DISKA)
MOUNT DESTINATION DISK=DISKB     ON DRIVE=0
    - PRESS ANY KEY ?
**** WRITING DESTINATION FILE ===> FA:0(DISKB)
**** WRITING DESTINATION FILE ===> FB:0(DISKB)
**** WRITING DESTINATION FILE ===> FABC:0(DISKB)
**** WRITING DESTINATION FILE ===> FA/EX1:0(DISKB)
**** WRITING DESTINATION FILE ===> FB/EX1:0(DISKB)
**** WRITING DESTINATION FILE ===> FABC/EX1:0(DISKB)
**** WRITING DESTINATION FILE ===> FA/EX2:0(DISKB)
**** WRITING DESTINATION FILE ===> FB/EX2:0(DISKB)
**** WRITING DESTINATION FILE ===> FABC/EX2:0(DISKB)
```

This results in the following file allocation:

BEFORE DISKB	DISKA	AFTER DISKB
FA	FA	FA
FB	FB	FB
FABC	FABC	FABC
	FA/EX1	FA/EX1
	FB/EX1	FB/EX1
	FABC/EX1	FABC/EX1
	FA/EX2	FA/EX2
	FB/EX2	FB/EX2
	FABC/EX2	FABC/EX2
GA	GA	GA
GB	GB	GB
GABC	GABC	GABC

SZAP

3. SZAP is used to display/change data on any track of a diskette. This example shows the general display format used by SZAP. This shows the directory track (2C.3) with the first four file entries. The SZAP screen edit mode will allow the user to easily change the information.

V-3 INVALID SECTOR 2C-3

```
0 2C 03 00  A0 00 00 00 00 00 53 59 53  31 30 20 20 20 53 59 53 *.....SYS 10   SYS*
        10  96 42 FF FF 09 00 2E 01  01 01 FF FF FF FF FF FF *.B..... ........*
        20  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF *........ ........*
        30  FF FF FF 00 01 40 4F 0A  1E 06 4F 0C 0A FF FF 00 *.....@O. ..O.....*
        40  A0 00 00 00 00 53 59 53  31 31 20 20 20 53 59 53 *.....SYS 11   SYS*
        50  96 42 FF FF 09 00 2E 41  FF FF FF FF FF FF FF FF *.B.....A ........*
        60  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF *........ ........*
        70  FF FF FF FF FF FF 4F 0A  1E 06 4F 0A 1E FF FF 00 *......O. ..O.....*
        80  A0 00 00 00 00 53 59 53  31 32 20 20 20 53 59 53 *.....SYS 12   SYS*
        90  96 42 FF FF 09 00 5E 81  FF FF FF FF FF FF FF FF *.B....^. ........*
        A0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF *........ ........*
        B0  FF FF FF FF FF FF 4F 0A  1E 06 4F 0A 1E FF FF 00 *......O. ..O.....*
        C0  A0 00 00 00 00 53 59 53  31 33 20 20 20 53 59 53 *.....SYS 13   SYS*
        D0  96 42 FF FF 09 00 2F 21  FF FF FF FF FF FF FF FF *.B..../! ........*
        E0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF *........ ........*
        F0  FF FF FF FF FF FF 4F 0A  1E 06 4F 0A 1E FF FF 00 *......O. ..O.....*
```

XHIT

4. XHIT is used for detecting a bad directory HIT table. This example shows the dialog between XHIT and user during the analysis of a bad diskette.

```
XHIT
XHIT - HIT TABLE ANALYSIS PROGRAM
Copyright 1980 - RACET computes - All Rights Reserved

MOUNT DISK TO BE CHECKED - CONTINUE (Y/Q) ?Y
SYS11/SYS HIT TABLE ERROR - CORRECT OR QUIT (Y/N/Q) ?Y
SYSCB/SYS HIT TABLE ERROR - CORRECT OR QUIT (Y/N/Q) ?Y
SYS15B/SYS HIT TABLE ERROR - CORRECT OR QUIT (Y/N/Q) ?Y

EXTRA DIRECTORY ENTRY IN HIT TABLE - REMOVE IT (Y/N/Q)
?Y
OK TO REWRITE CORRECTERD HIT TABLE (Y/Q) ?Y
HIT TABLE NOW UPDATED
```

XGAT

5. XGAT is used for detecting some errors in the GAT and directory tracks. This example shows a typical display for a bad diskette.

ERROR XGAT DIRECTORY LISTING

```
GAT & Directory Analysis Program - Copyright 1980 - RACET computes
Dir                  Bad     Ext      Extents(Hex)          Extents(Dec)
 #     Filename      Dir#.Ext  #   Trk Sec #Sec  Tot   Trk Sec #Sec Tot
---  ------------  --------  ---  ---- --- ---- ----  ---- --- ---- ----
  1 SYS10/SYS                 0    2E  01 000A 000A    46   1   10   10
                      **      1    01  01 000A 0014     1   1   10   20
  3 SYS12/SYS         *       0    5E  15 000A 000A    94  21   10   10
 12 IODVRS/SYS        **  1.1  0    01  01 0014 0014     1   1   20   20
 28 BACKUP            $       0    06  0B 0014 0014     6  11   20   20

Following GAT Table Entries Not Allocated To Any File
  (HEX)      (DEC)
Trk Sect   Trk Sect
--- -----  --- -----
 10   1-5   16   1-5
 2E 15-19   46 21-25
 2F   1-5   47   1-5
PROCESSING TERMINATED
```

DCS

6. DCS is used to display or print a combined directory of several diskettes. This example shows part of a listing sorted by diskname.

DCS LISTING - DISK SORT ORDER

```
DIRECTORY CATALOG SYSTEM      TUE MAY 27,1980(148)  08.45.18      PAGE 1.
Diskname     File-a          File-b          File-c          File-d
--------  ---------------  ---------------  ---------------  ---------------
BACKUP3      CAT/ASM          DISKID/ASM       DSECT/ASM        DSM/ASM
             DSMA1/ASM        DSMA2/ASM        DSMB/ASM         DSMC/ASM
             DSMD/ASM         DSMIO/ASM        DSMIO2/ASM       *E
             GENDSM/ASM       *N               TLRL             TLRL/ASM
             TLRL/BAS         TWLRL            TWLRL/ASM        XCREATE/ASM
BACKUP4      *CAT             *CBOOT           *DISKID          *DSM
             *DSMA1           *DSMA2           *DSMB            DSMBOOT
             *DSMC            *DSMD            *DSMIO           *DSMOBJ
             *E               EX1/CMF          EX1B/CMF         EX2/CMF
              EX2B/CMF        EX3/CMF          EX3B/CMF         *GENDSM
             *HELLO           HELLO/ASM        IN               INB
             *MBOOT           *N               OUT              OUTB
             *SZAP            UDSECT/ASM       X                *XCREATE
```

DCS LISTING - FILE SORT ORDER

7. This is another DCS example showing a listing in order by filename.

```
DIRECTORY CATALOG SYSTEM      TUE MAY 27,1980(148)  08.48.31      PAGE 1.
Filename/Ext    Disk-a        Disk-b        Disk-c        Disk-d        Disk-e
------------  ----------  ----------  ----------  ----------  ----------
&               TS11
AA              TS11
ACS1            DOC1
ACS2            DOC1
ACS3            DOC1
ACSLET          DOC1
ACSPROP         DOC1
ADUMP/BAS       SMOKE1      SMOKE2
ASCUP           *DEMOIII    *MASTER2    *MASTER3    *SMOKE1     *SMOKE2
                *WORKB
```

8. DCS can also be used to prepare a more detailed directory listing, as shown in this example.

FULL DCS LISTING

```
DIRECTORY CATALOG SYSTEM      TUE MAY 27,1980(148)  08.59.19      PAGE 1.
Diskname     Filename        Last Use  Created   Attr Type Lrecl #Recs  Eof
--------  ---------------  --------  --------  ---- ---- ----- -----  ---
BACKUP3      TLRL/BAS        02/26/80  02/26/80  DUO  F     256     0    0
DSMA         POUT/BAS        02/08/80  02/08/80  DUO  F       1   219  218
DSMII        TLRL/BAS        02/26/80  02/26/80  DUO  F     256     0    0
DSMSYS       MAHT/BAS        05/07/80  05/07/80  DUO  F     256     0    0
             MAHT1/BAS       05/07/80  05/05/80  DUO  F     256     1    0
             MAHT2/BAS       05/05/80  05/05/80  DUO  F     256     0    0
MASTER1      PIGLATIN/BAS    01/05/80  01/05/80  DUO  F       1  2066   17
             SNOOPY/BAS      01/05/80  01/05/80  DUO  F       1  4443   90
MASTER2      BIORHYTH/BAS    01/30/80  01/30/80  DUO  F     256    14    0
             PIGLATIN/BAS    01/30/80  01/30/80  DUO  F       1  2066   17
             SNOOPY/BAS      01/30/80  01/30/80  DUO  F       1  4443   90
```

XCREATE

9. XCREATE is used to create and to initialize it to the end. This example shows how to create a file FILED containing 45-records, each 100-bytes long.

```
                    XCREATE__FILED
LR=45,NR=100
```

10. DISKID is used to change the name of a diskette. This example shows how to change a diskettes name to "ALPHA" (the diskette originally was named "TRSDOS".

```
DISKID  ALPHA
DISKID PROGRAM - Copyright 1980 - RACET computes
MOUNT DISK TO BE CHANGED IN DRIVE #0 -->
       PRESS ENTER TO CONTINUE ?_
DISK #0 OLD DISKID=TRSDOS   NEW DISKID=ALPHA
       --  OK(Y/N) ?Y
FUNCTION COMPLETED SUCCESSFULLY
```

DEBUG-II

11. DEBUG-II replaces the DEBUG system of standard DOS. This example illustrates the display screen format. In this case a "CALL" instruction is shown just about to be executed. Note the *CA disassembled instruction and trace information on the next to the last line on the screen.

```
DEBUG-II — *CA EXAMPLE

35 21 5B 36 <PC =2A0E=>  CD FF 2E     *CA    NEXT<PC =2EFF=> 46 04 05 C8 3E 08
<IR =015D >  <AF =0254=>  P  Z PO NC            <AF'=21A0=>  M NZ PE NC
       F  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E      F0123456789ABCDE
2EFF  46 04 05 C8 3E 08 CF 23 18 F6 21 04 35 06 50 36     *F..H>.O#.v!.5.P6*
2F0F  20 23 10 FB 2A A8 35 11 04 35 7E FE 0D 28 0D FE     * #.{*(5..5~~.(.~*
2F1F  20 28 05 12 23 13 18 F2 23 BE 28 FC 22 A8 35 ED     * (..#..r#>(¦*(5m*
2F2F  53 B5 35 7E FE 0D C9 F5 3E 2A CF C1 21 04 35 3E     *S55~~.Iu>*OA!.5>*
EB F4 0E 2A <SP =2A00=>  31 00 2A 1B 1B CB C3 ED 53 BA    kt.*<SP =>1.*..KCmS:
00 00 00 00 <BC =3C58=>  00 00 00 00 00 00 00 00 00 00    ....<BC =>..........
00 00 00 00 <DE =EFFD=>  00 00 00 C3 1E F0 C3 61 F1 C3    ....<DE =>...C.pCaqC
20 DC 2A 00 <HL =365B=>  44 49 52 45 43 54 4F 52 59 20     \*.<HL =>DIRECTORY
2E CD FF 2E <IX =FFFF=>  C3 3E 24 3E 24 3E 24 3E 24 C3    .M..<IX =>C>$>$>$>$C
2E CD FF 2E <IY =FFFF=>  C3 3E 24 3E 24 3E 24 3E 24 C3    .M..<IY =>C>$>$>$>$C
ED 52 38 04 <BC'=02FF=>  ED 53 52 00 01 11 11 CD 60 05    mR8.<BC'=>mSR....M`.
FF FF FF 03 <DE'=2701=>  44 43 53 0D 2E 2E 2E 2E 2E 2E    ....<DE'=>DCS.......
EB F4 0E 2A <HL'=2A00=>  31 00 2A 1B 1B CB C3 ED 53 BA    kt.*<HL'=>1.*..KCmS:


GOING FROM 2A0E TO 2EFF VIA CD FF 2E     *CA  WITH SP=29FE
In,Cal,Lp,St,Tr,Jp,Vu,Hx,Klr,f1,Fnd,Gsb,Rpd,Ua,Wt,Dsp,Psc,Ext,CQ,CX,CE,CS,CI,CR
```

ORDER UTILITY PACKAGE AT $150.00

EXTENDED DEVELOPMENT PACKAGE

For The
TRS-80* MODEL COMPUTER SYSTEM

MACASM **SZAP** **DIS2**

THIS PACKAGE IS A **MUST** FOR MODEL-II ASSEMBLY LANGUAGE PROGRAMMERS!

This development package contains the necessary tools needed by a user for assembly language programming. These have been collected in a single package for the convenience of the user. These systems are major enhancements of those previously available for the Model-I system.

A summary of the capabilities of each system is shown below:

MACASM — This is an enhanced editor/assembler, including such features as macro conditional assembly capabilities, in-memory compiles, dynamic debug facilities, and many others. Source programs can be saved on disk and subsequently loaded back into memory. A range of lines can also be loaded or saved. All the previous editor/assembler features found in the Model-I version have been retained. Uploaded source programs from the Model-I system can be assembled without change by MACASM.

SZAP — This utility provides the capability to read and modify any sector on a diskette. In addition, it provides a generalized facility for copying any number of sectors from one area (or disk) to another. This includes recovery from input/output errors, allowing backup of diskette data not possible with the BACKUP command.

DIS2 — DIS2 is a system for the disassembly of Z80 machine language code. The code to be disassembled can be from memory or from a standard DOS load module from disk. Provisions are made to allow automatic offset for each load address, restart options, and a cross reference list of referenced locations.

All the above systems have been written in assembly language for fast and convenient use. Included is a 32-page manual, and 120 page assembly language reference booklet written by William Barden, Jr. The entire package is available for $125 from RACET COMPUTES, 702 Palmdale, Orange, CA 92665, or its distributors.

ORDER EXTENDED DEVELOPMENT PACKAGE AT $125.00

MACASM

MACASM provides the Model-II interface to the Microsoft "Editor Assembler Plus" software package. This provides the Model-II user with enhanced editor/assembler capabilities including macro and dynamic debug facilities. Listed below are the important features of this package:

Contains all the convenient facilities of the earlier editor/assembler package available on the Model-I. This includes a powerful assembler as well as an in-memory edit facility. This can be used to easily build, edit, and assemble source programs.

Provides the ability to assemble directly into memory. This allows the user to repeatedly assemble and execute a machine language program without reloading the editor assembler.

Includes a powerful MACRO facility capable of automatically generating in-line code. This facility is used to define one or more groups of generalized assembler statements called "macros". The assembler will automatically insert the group of instructions when referenced by a user defined operation code. Conditional assembly of statements is also provided.

Provides the ability to save programs to a disk file and subsequently reload them into memory. Facilities are also provided to save or load portions of the source program. This is very useful for extracting or saving common program segments.

Includes a dynamic debug facility (ZBUG) which can be used to trace, inspect, and change machine language programs. ZBUG displays memory with optional user program symbolic references.

Adds many new editor commands, including substitute, move, copy, and extend commands. New line number range specification formats have also been added to the system.

Allows the user to eliminate either or both the ZBUG or assembler portions of the system if additional memory is required.

Extends the assembler expression capabilities to include multiplication, division, and many logical operators.

Included with MACASM is the "Editor Assembler Plus" instruction manual written by William Barden, Jr. Additional features or differences in operation are described in the RACET manual.

SZAP

The purpose of SZAP is to allow the user to access, modify, copy, zero, and print any sector on a Model-II diskette. The features of SZAP include:

All tracks from 0-76 and sectors 1-26 can be read or written using Extended Development Package SZAP.

Hexadecimal as well as ASCII representation of the data is shown for each sector.

A convenient screen editor allows the user to easily update any sector on the diskette. This includes complete cursor control and the ability to enter changes in hexadecimal or ASCII.

Sectors can be easily scanned in either a forward or backward direction. The desired starting location can also be specified.

SZAP can function on a one drive system or multiple drive system.

The current sector displayed on the screen can be printed.

A special print mode will automatically print each sector displayed on the screen.

A repeat mode automatically repeats the previous command, allowing viewing and/or printing of a series of sectors.

A powerful copy command is available allowing the user to copy any number of sectors from one location to another, or from one disk to another.

SZAP is a necessary utility for manipulating data directly on the disk. Most important is the ability to examine and modify the directory. For example, this allows the user to recover lost files. SZAP can also be used to backup a disk that the standard BACKUP utility cannot because of I/O errors. In this case SZAP will note the errors but continue.

DIS2

DIS2 is a system for the disassembly of Z80 machine language code. The features of DIS2 are essentially identical to the original version distributed by Apparat. These features include:

Disassembles Z80 machine language code into standard mnemonics.

Disassembles code either directly from memory, or from a standard DOS load module previously saved on disk.

Provides an option to automatically offset each load address of the machine code being disassembled. This allows a module to be disassembled which resides in one location, but actually executes from another location.

Provides a restart option which allows a large disassembly to begin at a specified location.

Produces a cross reference list of referenced locations. This is useful in determining which areas of a program are being referenced by other locations.

Allows the reference table to be written to a file. This can then be processed by other programs.

DIS2 is executed as a standard DOS command followed by additional sub-commands as required.

***** REFERENCE/MOD-II *****

BASIC Program Cross Reference Utility
for the
TRS-80* Model-II System
A UTILITY PROGRAM REQUIRED BY EVERY BASIC PROGRAMMER!

Now You Can Display or Print:

Sorted cross references to all numbers or variables within a program.

Cross references to occurrences of a specific number or variable name within a program.

References to BASIC keywords.

Line numbers of statements containing a specified ASCII character string.

REFERENCE/MOD-II Is Easy to Use:

Exists as part of the BASIC system in a separate area of memory. Does NOT interfere with the users BASIC program area.

Loaded into memory by using a simple SYSTEM "REF" command.

Activated anytime by entering a **(CTRL) R** key.

Simple search command is entered.

Every programmer will find day-to-day use for REFERENCE/MOD-II. Combining and modifying existing programs now becomes a simple task. This utility also greatly reduces the requirement for printed listings.

ORDER BASIC XREF UTILITY AT $50.00

REFERENCE/MOD-II EXAMPLES

1. Cross reference by statement number. Note that all references are in sorted order. Multiple occurrences are shown by a "/" followed by the number found.

```
0   260 400/2 480 560 790 1030 1040
1   160 170 240 250 270/2 330 340 360/2 430 470 520/3 620 700
    750 760 800 840 880 900 910 920/2 930 940 950/3 960/2 990
    1020 1080 1180 1200 1210 10000 10040 10050 10060
2   230 320 570 580 620 640 740 750 860/2 880 930 950 980/2 990
    1080 1220
3   150/2 170 570/2 580/2 630 640/4 650/2 760 800 830 860/4 880
    940 950 980 1010 1020 1080
4   140/2 440/2 520 660
5   670 770/2 810/2

A   150( 160( 190$ 230( 250( 280( 320( 340( 370( 860$ 1160( 1220

B   180$ 220 570 570$ 580$ 640 640$ 860$
C   890$ 900$ 910$ 920$ 930$ 940$ 960$ 990$ 1020$ 1070$ 1110$

D   210 220 250/4 280/2 460 510$ 560$ 630$2
E   540$ 840 850$ 860 860$2 1050$
H   210 260/4 740
I   160/2 170/2 270 280 360 370 430 440/4 760 770/4 800 810/2
```

2. Cross reference of all variables. Note that dimensioned variables are followed by a "(", and all string variables by a "$".

3. Specific string searches are shown in this example. In this case the strings "BIORHYTHM", BIRTH" AND "DATE" were specified to be located.

```
BIORHYTHM/S  30 560 630 1255 10030
BIRTH/S   200 570 630
DATE/S    300 570 580 630 680
```

4. Search for BASIC keywords can be accomplished by looking for the corresponding Hexadecimal number. For example, 8C = IF, AF = PRINT, AD = LPRINT. This is the easy way to locate all "PRINT" statements!

```
8C/H 240 250 260 330 340 480 950 980 1010 1040 1160 1190 10010

AF/H 200 300 410 470 500 530 560 570 580 590 600 610 870 880

AD/H 2 525 1050 1060 1070 1080 1100 1110 1120 1250 1255 10020
     10060
```

'DISCAT'
DISKETTE CATALOG INDEX PROGRAM
for the TRS-80 Models I, and III

DISCAT is a comprehensive Diskette Cataloguing/Indexing Utility. This utility allows the user to keep track of thousands of programs in a categorized library. A 32K system can index up to 800 program location records with up to nine indexes in each catalog. A 48K system can provide cataloguing maintenance for over 1900 program location records in each of the nine indexes. Your catalog of indexes could contain data for over **17000** programs.

DISCAT is a machine-language program that operates with TRSDOS, NEWDOS+, and NEWDOS80 operating systems. Once a specific index is loaded into memory, access to any specific program is less than three seconds. Other important features of DISCAT include:

- Automatically keeps track of free space available for each of your diskettes.
- Will allow operation with a one-drive system, or allows selection of drive for updating on multiple-drive systems.
- Provides for automatic file maintenance. Simply insert your diskette with program changes into the update active drive, and press automatic update key.
- Allows for manual program entries for 'foreign' operating system diskettes (e.g. CPM, FORTH, PASCAL).
- Sort menu allows for selection of either program name or disk number sort. 800 entries are sorted in less than 25 seconds.
- Allows deletion of programs, an entire diskette of programs, or clearing of an entire index file from memory for creating of a new index.
- Provides for single or double column printer options.
- After index file maintenance is performed, DISCAT allows the user to save the current index to disk with the same filename or to select a new filename.

DISCAT file entries include program names and extensions, diskette number where program is stored, program location on diskette (front or back side), and length (in granules) of each program. A special display provides a listing of the free granules of each of the diskettes in the index.

ORDER DISCAT MOD I $50.00
MOD III $50.00

'KFS-80'
KEYED FILE SYSTEM
for the TRS-80 Models I, II, and III

KFS-80 provides the BASIC programmer the ability to rapidly insert or access keyed records in one or more data files. Features of this system include the following:

- Easy to use Basic Interface.
- Records are maintained in sorted order by a specified key, such as name, zip-code, etc.
- Records may be inserted or retrieved by supplying the key for a specific record.
- Records may be retrieved sequentially in sorted order.
- Binary tree index system provides consistently rapid access to any record in a file, regardless of the number of records being maintained.
- Multiple index files can be easily created which allow access of a single database by multiple keys, such as by both name and zip-code.

KFS-80 provides the application programmer with a powerful disk handling facility for development of data base applications. KFS-80 is ideally suited for inventory, accounts payable, accounts receivable, and virtually any application where rapid access is required to named records.

File access functions include:

INITIALIZE — initializes KFS-80
OPEN — opens files
CLOSE — closes files, flushes buffers
INSERT — adds records to file
DELETE — delete records from file
FIND — positions record pointer to named record
READKEY — retrieves record by KEY
WRITEKEY — writes record by KEY
READ — retrieves record by record pointer
WRITE — writes record by record pointer
EXTRACTKEY — retrieves record by KEY, but record pointer not advanced
EXTRACT — retrieves record by record pointer
STATUS — returns record pointer and file status.

KFS—80 provides keyed and sequential access to multiple files, with records up to 240 bytes. Sectors are dynamically added to the file as necessary. Also, a sector buffering mechanism minimizes disk accesses by 'remembering' which sectors are in memory. Utility routines are included for file maintenance.

ORDER KFS-80 MOD I $100.00
MOD II $175.00

'MAILLIST'
A MAILING LIST DATABASE SYSTEM
for the TRS-80 Models I, II, and III

MAILLIST is ideally suited for organization mailing lists, personal addressbook functions, or mail lists based on dates such as reminders for birthdates or dues payable.

MAILLIST uses an ISAM-based file system to minimize disk access times. A database consists of a master file, and up to four index files. This allows the primary file to be maintained in sorted order by name, zipcode, and up to two dates. File access for update/delete functions is by name. Duplicate last names are automatically handled by the system. The ZIPCODE and DATE files reference the master file for data. The DATEs significance is specified by the user.

MAILLIST allows up to 30 attributes to be specified. These attributes are referenced symbolically and are defined by the user. They are used in record selection when generating reports or mailing labels. MAILLIST supports 9-digit zipcodes and 3-digit state codes.

Reports and labels may be ordered by NAME, ZIP, DATE1, or DATE2. Printing may be started or ended at any point in the list. Records are selected by attribute combinations specified when the report is requested. A mask and value selection mechanism permits database query access.

MAILLIST capacity is 600 names for Model I, 1200 names for Model III, more than 3500 names for Model II, and more than **38000** names for a Model II with the RACET Hard/Soft Disk System.

ORDER MAILLIST MOD I $ 75.00
MOD II $150.00
MOD III $ 75.00

'LPSPOOL'
LINE PRINTER SPOOLER
for the TRS-80 Model I

LPSPOOL is a facility which permits concurrent printing and processing. The significant features of LPSPOOL include the following:

- Computer 'throughput' is increased by allowing printing to be done while the computer is used for other purposes.
- Multiple copies of output may be produced automatically, without user program changes.
- Output may be held until a more convenient time for printing, or deleted if not required.
- Output may be temporarily stopped, resumed or restarted at any time if printer or paper problems are encountered.
- Supports parallel or serial printers.

Print output is written (spooled) to disk automatically by LPSPOOL, which subsequently prints (despools) the information. All these features are available without changes to a user's program.

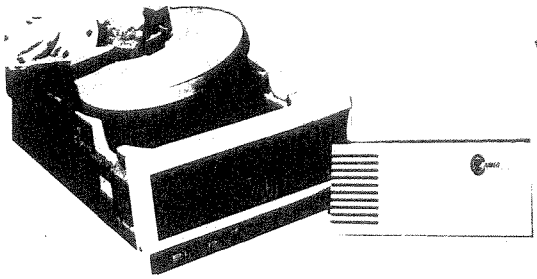
LPSPOOL includes a 'multi-tasking' monitor which automatically keeps the printer busy (foreground task), while at the same time allows the users program to run (background task). LPSPOOL, acts like an extension to the lineprinter driver. When LPSPOOL is active, print data is directed to spool disk files. As print files are created, their names are inserted in a spool queue. The foreground despooler, also working from a queue, looks for active entries ready for printing. The spool and despool queue may be the same to provide automatic printing as spool files are generated.

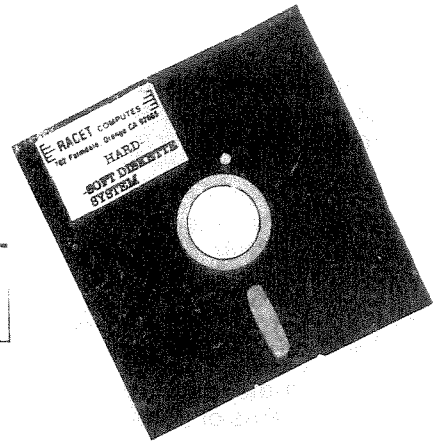
A set of LPSPOOL commands is provided to control the spool and despool activity. These commands may be entered by the user with the SPOOL library command or by BASIC with the LPRINT command. The LPRINT may be used in an application program or by the user in BASIC direct execution (keyboard) mode. Assembly language programs may also invoke these commands by calls to the line printer driver. An on-line display facility provides a dynamic status display of the LPSPOOL system.

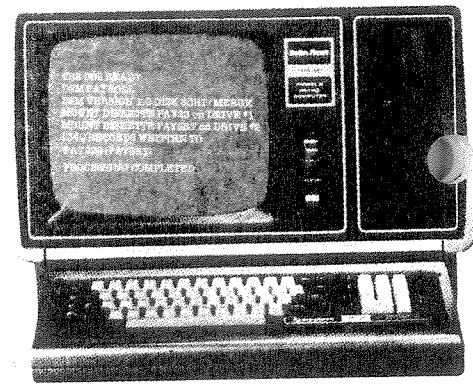
LPSPOOL may be used to significantly increase the 'throughput' of the TRS-80. It can be used to save print information in disk files for subsequent and repeated use. This allows multiple copies of output without the necessity of re-running the application programs which generated them. Further, a despool queue can be established which allows unattended operation for long print jobs.

Each queue entry allows the specification of spool filename, number of copies, form type, whether the file is to be printed immediately, and whether the queue entry and/or file is to be deleted after printing.

ORDER SPOOLER MOD I $75.00







HARD (CARTRIDGE) DISK SYSTEM

DO YOU NEED 5-10-40 MEGABYTES OF DISK STORAGE? WOULD YOU LIKE DISK ACCESSES 3 - 8 TIMES FASTER? ALL WITH THE CONVENIENCE OF TRSDOS!!

RACET COMPUTES has selected Cartridge Disk Drives with **removable media** (disk packs) as their choice for the TRS Model II. Cartridge drives allow interchangeability of the on-line data by simple pack swapping. This allows different applications to be stored on removable disk packs when not needed (e.g. massive mailing lists used Quarterly). Dual platters provide **easy backup** procedures!!!

Cartridge drives available include the 10-Megabyte (5-fixed, 5-removable) Western Dynex Series 6000 and Control Data Corporation 9427H Hawk hard disks. Cartridge technology is a mature, well-documented technology and has been used on large computers for many years. Service is available in most major cities throughout the world.

The CAMEO DC-500 Controller is used to interface the Model II to the cartridge drives. Each controller can handle up to four cartridge drives. The DC-500 is designed to make installation by non-technical personnel easy. The controller is a **stand-alone** device with two interface boards; one to plug into the Model II backplane, the other to adapt to the selected disk drive. The DC-500 has its own independent power supply, thus placing virtually no demands on the Model II power supplies. The DC-500 Controller includes all the components necessary to connect the Model II to the hard disk drive.

SEE PRICE LIST ON INSIDE BACK COVER FOR HARDWARE PRICES

MULTI-USER HARD DISK SYSTEM

WITH THE ADDITION OF THE MX200 OR MX1600 MULTIPLEXER TO THE HARD DISK SYSTEM, YOU CAN HAVE MULTIPLE MODEL II ACCESS TO THE SAME DATA BASE!!

The MX200 2-port MUX plugs onto the existing DC500 Controller. The MX1600 — 4-port MUX stand-alone box connects to the DC500 Controller. These multiplexers, along with the RACET Hard/Soft Disk System software will provide a powerful system for multiple-file, multiple-record access. *(Available January 1981.)*

SEE PRICE LIST ON INSIDE BACK COVER FOR HARDWARE PRICES

HARD/SOFT DISK SYSTEM

THE HARD DISK SOFTWARE IMPLEMENTATION YOU HAVE BEEN WAITING FOR!!!!
MODEL II **TRSDOS** COMPATIBLE

Features of the Hard/Soft Disk System (HSD) include the following:

- Compatible with your existing BASIC programs & files.
- All BASIC OPEN, CLOSE, GET, PUT, INPUT#, PRINT#L, KILL, FIELD, etc. statements are identical to the existing TRSDOS commands *(see note 1)*.

- Compatible with most machine language programs that use the standard calling sequence *(see note 2).*
- You can access both your floppy disk drives **and** HSD files interchangeably (Hard disk drives are numbered 4, 5, 6 . . .)
- Files can be dynamically assigned and extended using comprehensive techniques superior to those used on the floppy disk.
- A single file can be as large as one disk *(see note 1)* — *i.e.,* a file can be as large as 10-million, 20-million bytes or larger.
- The directory can be expanded by the user to handle thousands of files.
- HSD includes the RACET XCOPY utility program enhanced for HSD. Floppy or HSD files can be copied from one device to another. This includes multiple-file copies and powerful select options.
- HSD includes the RACET DCS utility program enhanced for HSD. This utility allows the user to display/print combined directories from one or more drives, with enhanced select options.
- HSD includes the RACET Superzap (HZAP) utility program modified for the HSD. This allows you to look at or modify the contents of an HSD sector.
- HSD includes a paramaterized FORMAT utility for hard disc drives. This includes options for specifying the number of sectors/track, platters/drive, sectors/granule, sectors/directory, etc. You can build an HSD system to fit both your hardware and software requirements.

NOTES:

1. The BASIC GET n,I and PUT n,I statements only allow 'integers' for the value of 'I.' This limits the record range to 1-32767. This corresponds to a maximum file size of 7,388,352 bytes (LRECL = 64). A separate module is included which modifies BASIC to allow a 24-million byte record number range. This module, however, will require a change (at a small handling charge) with each new operating system release by Radio Shack.

2. The initial version of HSD does not support the type 'V' record. This feature is NOT used with BASIC programs — but might be used by COBOL or machine language programs.

3. HSD has a memory overhead of approximately 2K.

ORDER **HARD/DISK SYSTEM** SOFTWARE AT $400

★ ★ ★ ★ BASIC LINK FACILITY "BLINK" ★ ★ ★ ★

LINK FROM ONE BASIC PROGRAM TO ANOTHER — SAVING ALL VARIABLES!!!
MERGE-OVERLAY CAPABILITIES — SAVING ALL VARIABLES!!!
START EXECUTION AT ANY LINE NUMBER ON CHAINED OR MERGED PROGRAM!!!

BLINK allows the user to run one BASIC program, and then transfer control to another BASIC program without losing variables in memory. The features and use of BLINK are summarized below:

- BLINK is a small machine language routine residing in protected memory.
- BLINK is given control by a simple USR statement to link to the next program (e.g. ?USR ("next filename")
- The BLINK 'Merge' Option allows merge overlays of new program material intermixed with the existing program lines when BLINK is exercised with the ',M' option. The program to be merged must be saved in standard format (Not ASCII). All variables are retained in the resultant merged program.
- Execution may start at any line number when the ',(line number)' option is selected — either on a standard chained, or on a merged program.
- This small, but powerful, utility simplifies the process of creating large programs. Programs can now be easily modularized for ease of programming, versatility, and efficient memory management.

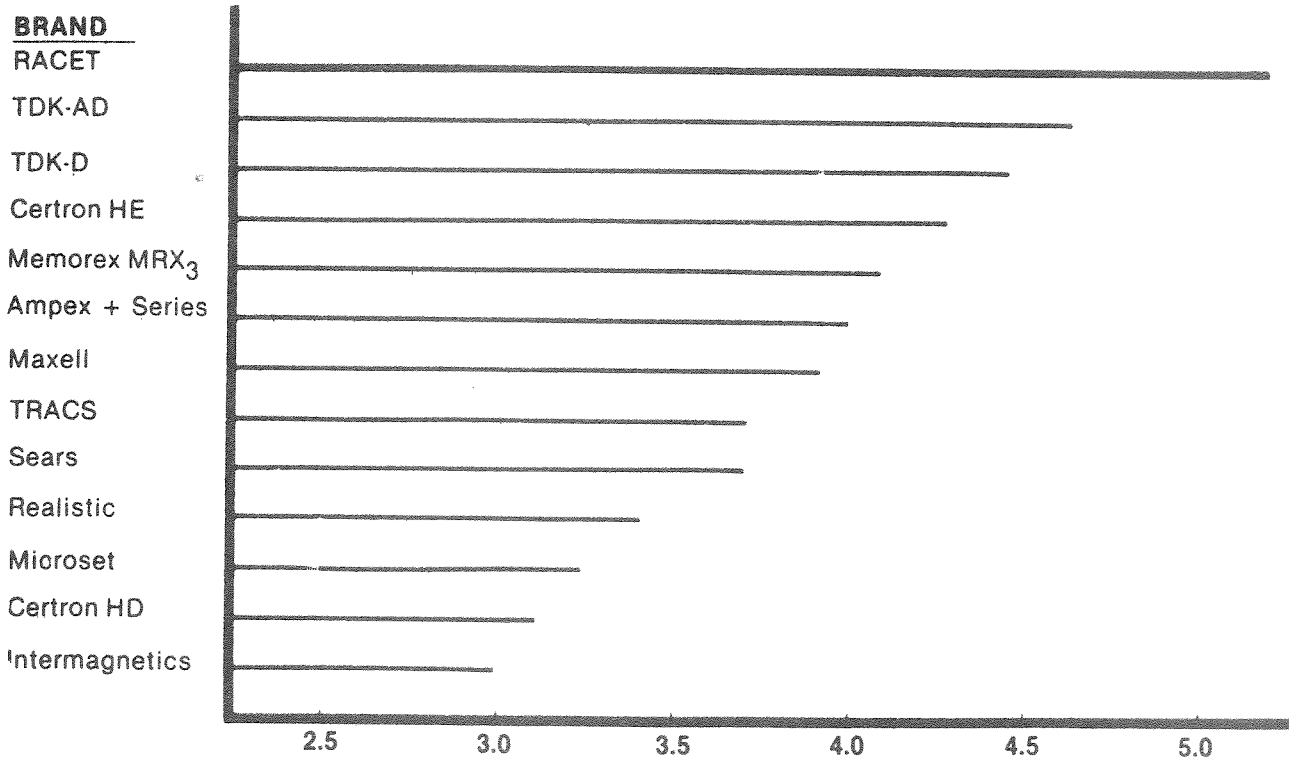
ORDER **BLINK** AT $25.00 FOR MOD-I
$40.00 FOR MOD-II
$30.00 FOR MOD-III

AT LAST — A CASSETTE TAPE THAT IS RIGHT FOR YOUR MICROCOMPUTER

RACET COMPUTES has found the answer. RACET is now announcing the availability of a special formulation cassette tape optimized for microcomputer applications. This tape, after extensive research and evaluation of existing magnetic media, has been shown to have many outstanding characteristics including:

- Extremely broad frequency response — works well with higher baud-rate systems.
- Exceptional density characteristics — results in acceptance of high signal levels before saturation and a much broader playback range.
- Consistency in recording characteristics — most important in sensitive systems.
- Top quality tape materials to provide long life, reduced breakage, and longer usage before heads and capstans need cleaning.
- Top quality 5-screw shells with lubricated liners to provide smoother running through the tape transport and to facilitate repairs.
- Furnished with hard plastic case to provide protection from dust and handling.

Our test results for several brands of audio tape, which are summarized below, indicate the overall superiority of RACET COMPUTES Microcomputer Tape in terms of playback range with no data loss. Test data was accumulated on both modified and unmodified TRS-80 Microcomputers in both machine language and BASIC.



PLAYBACK RANGE (Average) — No Lost Bits

The best test is your application; we welcome your comparison of RACET COMPUTES Microcomputer Tape with your present tape. RACET COMPUTES Microcomputer Tapes will reduce the problems you have probably experienced with your microcomputer, such as lost programs, noise and extraneous bits of information in your programs, and the sheer frustration of the sensitivity of your system to critical volume settings. — C-20 cassettes at 10 for $14.95 (postpaid) or 100 for $119.95 (freight collect).

ORDER BLANK

>>>>>>>> SOFTWARE PRICING <<<<<<<<

PAGE	PROGRAM	PRICE			CHECK MEMORY SIZE			TOTAL
		MOD I	MOD II[1]	MOD III[2]	16K	32K	48K	
2	REMODEL[3]	$25	N/A	N/A				
2	REMODEL + PROLOAD[3]	$35	N/A	$35				
6	GSF[3]	$25	$50	$30				
6	DOSORT (Includes GSF)[3]	$35	N/A	$45	N/A			
2	COPSYS (Non-DOS)[3]	$15	N/A	$20				
2	TIMSER	$15	N/A	$30				
2	Y-YBAR[3]	$15	N/A	$30				
2	MDA	$109	N/A	N/A				
1	COMPROC[3]	$20	N/A	$30				
4	INFINITE BASIC[3]	$50	N/A	$60				
4	INFINITE BUSINESS[3]	$30	N/A	$30				
8	DSM	$75	$150	$90				
15	CROSS-REFERENCE	N/A	$50	N/A				
13	DEVELOPMENT SYSTEM	N/A	$125	N/A				
10	UTILITY PACKAGE	N/A	$150	N/A				
19	BLINK (DOS only)	$25	$40	$30				
18	HARD/SOFT DISK SYSTEM	N/A	$400	N/A				
16	KFS-80 (ISAM)	$100	$175	$100				
17	MAILLIST	$75	$150	$75				
17	LPSPOOL	$75	N/A	N/A				
16	DISCAT	$50	N/A	$50				
20	CASSETTE TAPES (C-20)	10/$ 14.95		100/$119.95				

>>>>>>>> HARDWARE PRICING <<<<<<<<

PRODUCT	DESCRIPTION	PRICE
DC500 Controller	The DC500, adaptors and cables	$1500
Subsystem # 1	Western Dynex Drive and DC500	$5995
Subsystem # 2	CDC Drive and DC500	$6995
Second Drive	Western Dynex Series 6000	$4495
	CDC 9472H Hawk	$5495
Multiplexer	MX200 — 2 Port Mux (does not include extra computer adaptor)	$ 395
	MX1600 — 4 Port Mux (does not include extra computer adaptors)	$ 695
Adaptors	Extra Computer Adaptors (each)	$ 200
	Extra Drive Adaptors (each)	$ 40
Disk Packs	Control Data Disk Packs	$ 100

[1] Please specify version 1.2 or 2.0 for Mod II Programs.
[2] MOD III Programs available 1st Quarter 1981.
[3] Furnished on cassette. On formatted only (non-DOS) diskette add $6.00.

SUBTOTAL _____
California Orders Add 6% _____
Foreign Special Handling _____
TOTAL _____

- NO SHIPPING CHARGES ON U.S. OR CANADIAN ORDERS. FOREIGN ORDERS ADD $5.00 SPECIAL HANDLING.
- ALL FOREIGN ORDERS PAYABLE IN U.S. DOLLAR INSTRUMENTS.
- U.S. SHIPMENTS BY UPS UNLESS OTHERWISE SPECIFIED.

RACET COMPUTES
1330 N. Glassel, Suite 'M'
Orange, CA 92665
(714) 637-5016

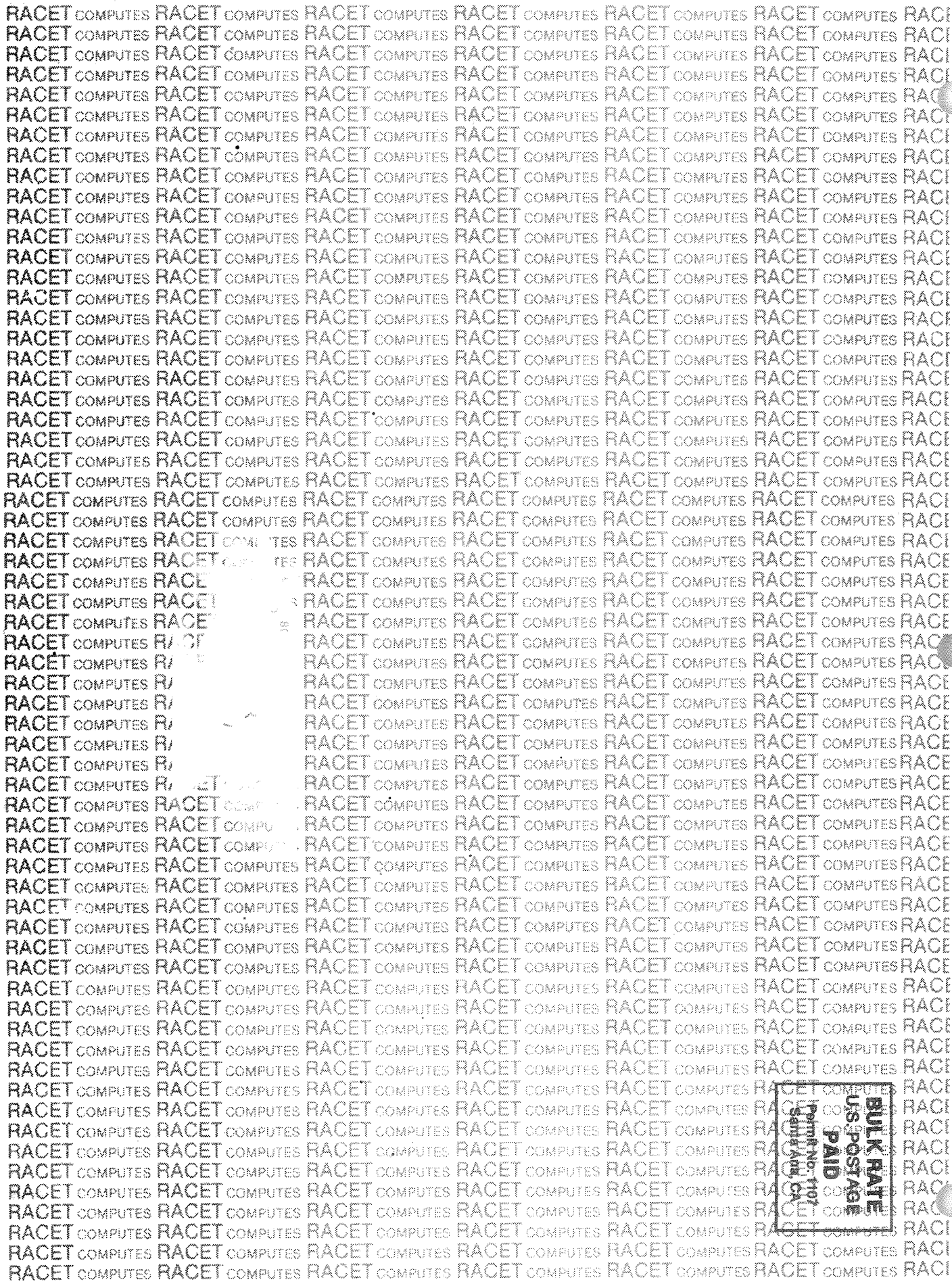
NAME _____
SHIPPING ADDRESS _____
CITY _____ STATE _____
COUNTRY _____ ZIP CODE _____

☐ Ship COD ☐ Check Enclosed ☐ Master Charge VISA

ACCOUNT NO. _____ EXPIRATION DATE _____

IF M/C. INTERBANK NO. _____ SIGNATURE _____

RACET COMPUTES products are available from dealers and distributors world-wide. If your local dealer doesn't presently carry the RACET COMPUTES line — let him know about us. It is RACET's objective to support local dealers, thereby making our products readily available to the local user. Your support will assure ready local availability of top quality programs AND the chance to review documentation BEFORE you buy. Your local dealer will benefit by carrying the top name in TRS utilities — RACET COMPUTES



BULK RATE
U.S. POSTAGE
PAID
Permit No. 1107
Santa Ana, CA