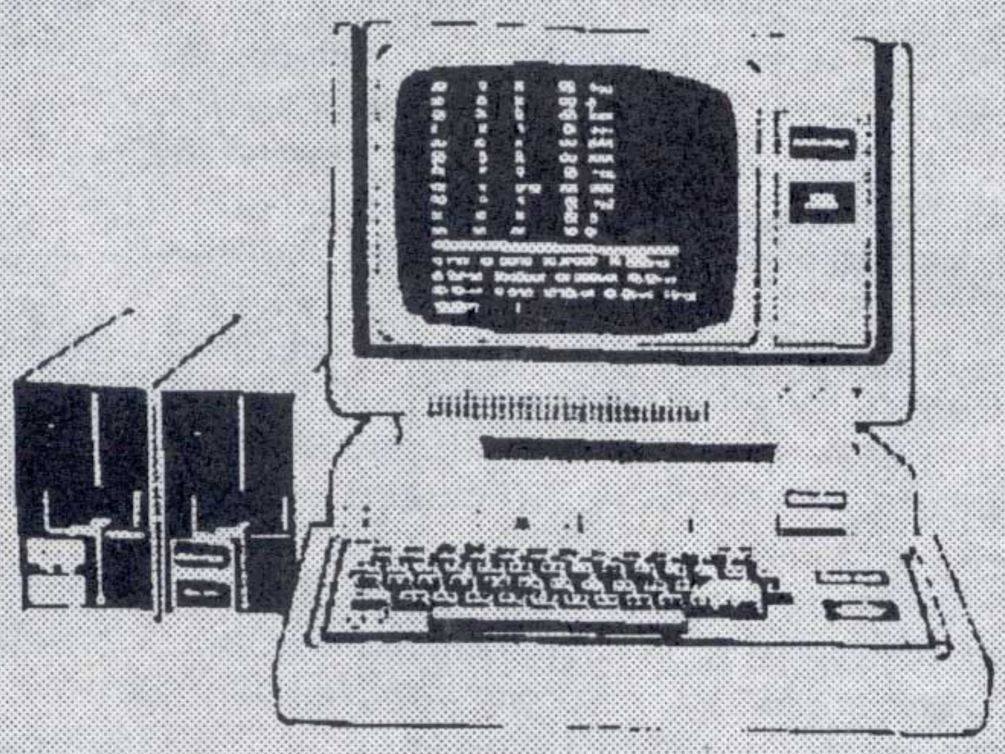
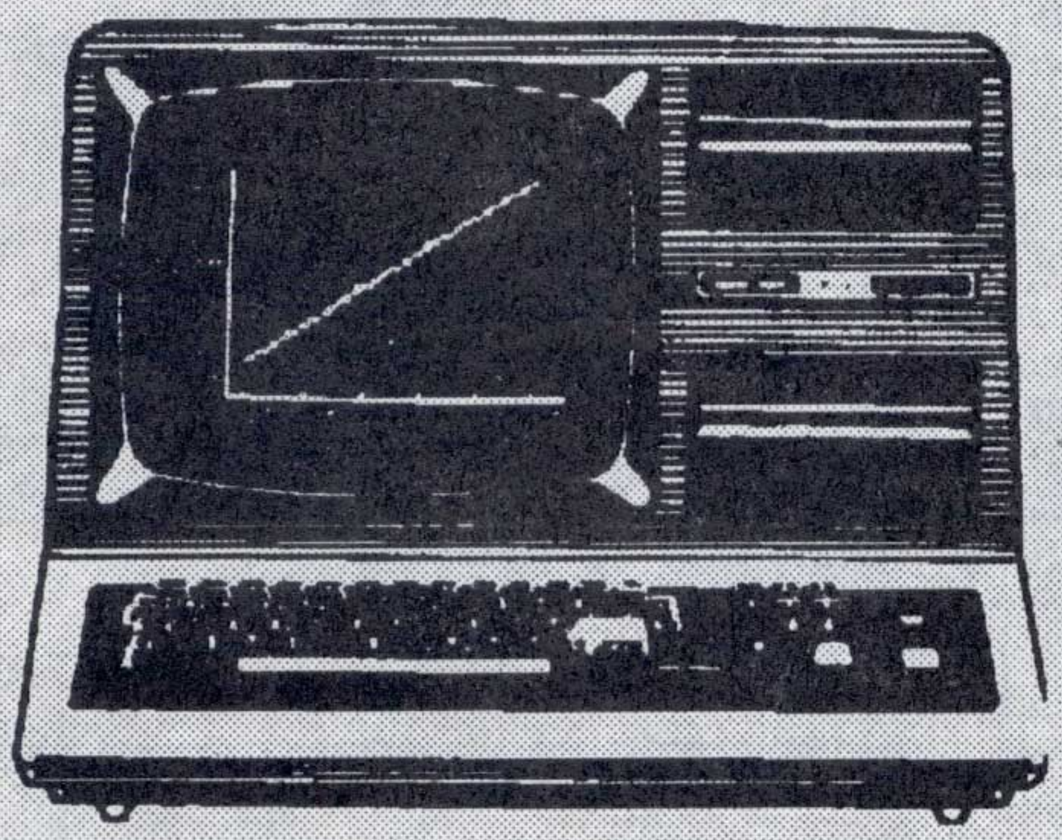
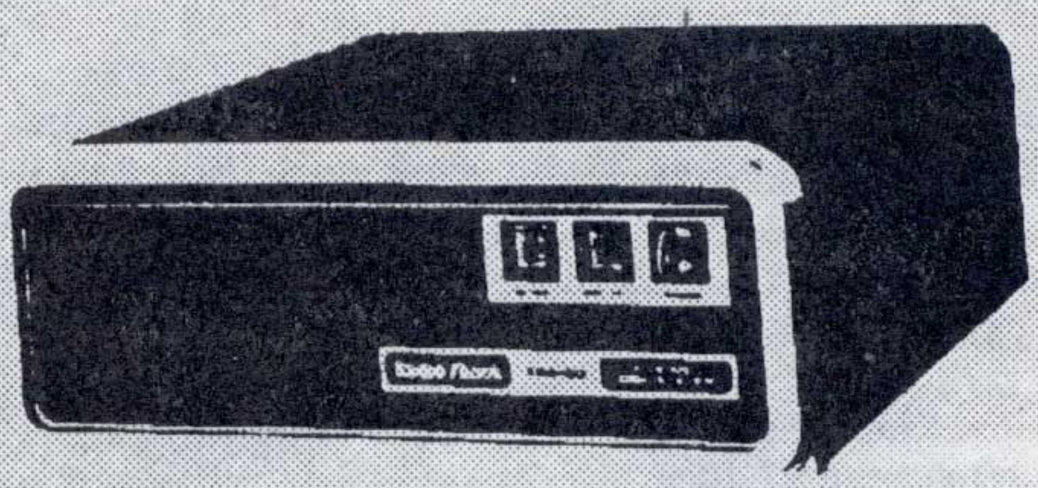
TRSTimes

Volume 5. No. 5. - Sep/Oct 1992 - $4.00







LITTLE ORPHAN EIGHTY

Just when you think that you have solved a problem once and for all, something happens that puts you back where you started. This was the case with the Jul/Aug 1992 issue of TRSTimes, where "the unconquerable forces of nature turned against me"; in other words -- I blew it.

What happened really irritates me, because it shouldn't have occured at all. But let me start at the beginning....

As I have chronicled in earlier issues of TRSTimes, the magazine is developed and put together using IBM-PC compatible equipment. I am using a 386/33 clone with an HP series II laser printer. This is good equipment that allows me to get the immediate job at hand done quickly. However, the equipment also have some serious drawbacks for my chosen task.

Obviously, TRSTimes is about TRS-80 computers and the articles, whether written by me or by someone else, are usually written on TRS-80 equipment, using TRS-80 software, and stored on TRS-80 formatted disks. These, of course, are not readable on PC equipment, so I copy the contents of the submission disks to a PC readable disks, using SuperCross4. Once the material is in PC format, I load the article and/or the program listing into WordPerfect 5.1, where the document is spellchecked, and minor corrections are made. I then save the document in WordPerfect format, using a new filename. The final step is to load the new file into Ventura Publishing where the text is formatted, headlines are blown up, and graphics are inserted, etc.

Now, it would seem reasonable that this method of file-transfer should be prone to error, but I have had no problems, so far -- SuperCross4 (and TRSCross) have worked flawlessly. Loading the just-transferred files into WordPerfect 5.1 has not been a problem either, though LeScript and (Super)Scripsit files have displayed some odd formatting behaviour.

The only problem I have encountered is the transfer of files from WordPerfect to Ventura Publishing. Though Ventura has a utility to read WordPerfect files, there's a bug that bites me from time to time. Ventura uses the "greater than" (>) and "less than" (<) signs as text formatting commands. Should the text just imported to Ventura contain certain control characters in combination with the "less than" or "greater than" signs, the file is displayed strangely. This is a minor problem as it is visually obvious what has just happened and, thus, rather easy to correct.

Other control characters combining with the "less than" or "greater than" signs produce a somewhat more subtle error in the file: while the file appears to be normal -- *the "less than" and "greater than" signs have all disappeared!*

This is exactly what happened to Chris Fara's "PROGRAMMING TIDBITS" column on page 27. The article makes no sense at all when the "<" and ">" signs are left out. Now, I caught these errors the night before the issue was going to the printers, so I dutifully made the corrections and printed out a new master sheet. As it turns out, I then proceeded to put the original sheet back with the masters, while I threw the new, corrected sheet in the waste basket. Yes, funny things happen at 2 A.M.

Anyway, I apologize to the readers and to Chris Fara for butchering a fine article. To make matters right, the corrected version is featured elsewhere in this issue.

Also, apologies go out to Doug Kilarski of Computer Monthly. It was not the intension of Michael Ecker, nor that of TRSTimes, to associate Mr. Kilarski to the "computer writer turned Editor" talked about on page 19 of issue 5.4.

Many people have asked why TRSTimes does not have a BBS. Well, if you remember a few years ago, we did try to arrange for one to support the TRS-80. Unfortunately, the Sysop was not all that interested in TRS-80's, so the effort failed. Somehow, Roy Beck and I ended up discussing this failure a couple of weeks ago, and we agreed that it might be worth an effort to set one up ourselves. Now, this is not a firm commitment to open a TRS-80 BBS in Southern California, but we are talking about it, and we just might talk ourselves into it. We'll keep you up to date on this project.

And now for the sad part. If you use or have used AllWrite, chances are that you have called ProSoft in North Hollywood for clarification and help with their fine wordprocessor. The expert that guided you and eventually solved your problems was Ron Malo.

On Friday, Aug 14, 1992, Ron suffered a heart attack and passed away during the night.

Ron was a longtime member of the Valley TRS-80 Hacker's Group and, along with his wife, Donna, promoted and coordinated the activities of VTUG.

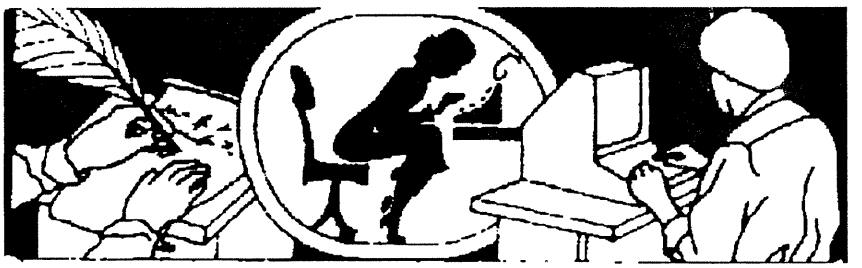
The TRS-80 world has lost a good friend. Ron, we miss you.

TRSTimes magazine

Volume 5. No. 5. - Sep/Oct 1992

PUBLISHER-EDITOR
Lance Wolstrup
CONTRIBUTING EDITORS
Roy Beck
Dr. Allen Jacobs
Dr. Michael W. Ecker
TECHNICAL ASSISTANCE
San Gabriel Tandy Users Group
Valley TRS-80 Users Group
Valley Hackers' TRS-80 Users Group

TRSTimes magazine is published bi-monthly by TRSTimes Publications. 5721 Topanga Canyon Blvd, Suite 4. Woodland Hills, CA. 91364. (818) 716-7154.
Publication months are January, March, May, July, September and November.
Entire contents [c] copyright 1992 by TRSTimes publications.
No part of this publication may be reprinted or reproduced by any means without the prior written permission from the publishers.
All programs are published for personal use only. All rights reserved.
1992 subscription rates (6 issues):
UNITED STATES & CANADA: $20.00 (U.S. currency)
EUROPE, CENTRAL & SOUTH AMERICA: $24.00 for surface mail or $31.00 for air mail. (U.S. currency only)
ASIA, AUSTRALIA & NEW ZEALAND: $26.00 for surface maile or $34.00 for air mail. (U.S. currency only)
Article submissions from our readers are welcomed and encouraged. Anything pertaining to the TRS-80 will be evaluated for possible publication. Please send hardcopy and, if at all possible, a disk with the material saved in ASCII format. Any disk format is acceptable, but please note on label which format is used.



THE MAIL ROOM

MODEL 2

I continue to look forward to each issue of TRSTimes. Did I read a couple of issues back that you had acquired a Model 2, and might write an article on it? I came by a dead one via a yard sale about a month ago. Got it going after some tinkering, so I would be interested in the article, if you choose to do it.

In experimenting with it and reading the manuals that came with it, it would seem the Model 4 was somewhat of a remake of the Model 2. Basic is very similar, though not quite identical. Disks maintain 2 directories??

Please keep up the good work with TRSTimes.

Richard M. Gilfillan
Wellington, KS

For a short time I had a Model 16B. Unfortunately, the disk drives kept giving me trouble, finally crashing my Editor/Assembler disk which had several programs almost ready for TRSTimes. As it was just a lark, I gave up on it right then and there. You are right, the Model 2 is a lot like the Model 4. As a matter of fact, you can buy a version of LS-DOS 6.3.1 that will run on the Mod 2. You can get it from Misosys.

Ed.

NETWORK 4

Through your magazine, I purchased the M.A.D. Software Rom chips for three of my Radio Shack Model 4's. There were several references to a "NETWORK 4 network" and the hardware. Having never heard of this before, I am naturally interested. Can you give me any information on "NETWORK 4" or point me in a direction?

Martin J. Rapoport, CPA
Trexlertown, PA

Not being familiar with NETWORK 4 myself, I asked Roy Beck about it. Not only had he heard about it, he had the hardware. So, per your request, he will come up with an article for a future issue. We aim to please!

Ed.

BOOT5

Thanks for the answer to my request for info on possible PATCHes on BOOT5/CMD to be able to boot LDOS 5.3.1 instead of LDOS 5.3.0 from LS-DOS on my Misosys 40 meg. hard disk. I typed the patches into a /JCL file, PATCHed BOOT5/CMD, and it worked fine -- naturally. I really love my TRSTimes and appreciate your hard work!

Anthony B. Mizzell
Chesapeake, VA

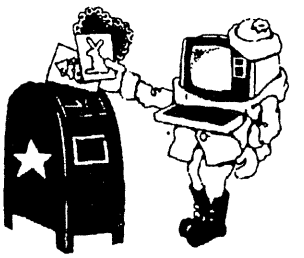
MODEL I, III & 4 UPGRADE KITS

I'm the manager of the Tandy Service Center in Syracuse, NY. Below is a list that may be of interest to the readers. The list includes several Model 1, 3 and 4 upgrade kits that have we have available at reduced prices:

26-1102	Model 1 & 3 16K RAM Kit	$ 9.95
26-1122	Model 4 64K RAM Kit	$19.95
26-1123	Model 3 to 4 Upgrade Kit	$49.95
26-1125	Model 3 Graphics Board Kit	$19.95
26-1126	Model 4 Graphics Board Kit	$19.95
26-1127	Model 4 Drive 0 Kit	$49.95
26-1132	Model 1 Hard Drive Interface Kit	$14.95
26-1145	Model 1 RS-232 Upgrade Kit	$ 9.95
26-1148	Model 3 & 4 RS-232 Upgrade Kit	$49.95
26-1162	Model 3 Drive 0 Kit	$49.95
26-1163	Model 3 & 4 Drive 1 Kit	$49.95
26-1164	Model 3 & 4 External Disk Drive	$39.95

All kits are brand new and have a 30 day warranty. Supplies are limited and are subject to being sold out. Kits may be purchased in person by cash, check or credit card. Mail orders may be purchased by check, money order, phone or FAX using a credit card. Add $3.00 shipping to all mail orders regardless of the number of kits ordered. Check and money orders are to be made payable to: TANDY SERVICE. All shipments will be made by UPS, so be sure to include a street address we can ship to; no P.O. Boxes. New York State residents must include 7% Sales Tax on all orders, including the shipping charge. Allow 3 weeks for delivery.

Walt Danylak, Manager
Tandy Service #40-7112
Geddes Plaza
527 Charles Avenue
Syracuse , NY 13209
Phone: 315-468-3366
FAX: 315-488-2891



GRAPHICS-90:
THE SAGA CONTINUES

By Gary W. Shanafelt

By now, some of you are probably starting to wonder if this magazine should be retitled something like the "GRAPHICS-90 Review". Every other issue seems to have something about Software Affair's GRAPHICS-90 program. And here is another one! Will they never cease? If you've run the program, you'll know what all the hoopla is about. But those with Model 1's up until now weren't able to, since it only ran on the Model 3 (or 4 in 3 mode), unless they took advantage of Mathieu Simons' patches for Model 1 NEWDOS/80. That is no longer the case; a revised version of the GRAPHICS-90 package is now available, one that supports most Model 1 DOSes as well as those of the Model 3 -- plus containing some other new features.

What happened was this: Pieter Plomp, in the Netherlands, inspired Mathieu Simons in Belgium to investigate Model 1 compatibility for the programs, particularly the core program, ABASIC. Yours truly, in Abilene, Texas, figured that a whole reworking of the program modules would be more convenient for people than pages of patches. Most of the addresses between the Model 1 and 3 are the same, though some critical ones used by GRAPHICS-90 are different -- such as the high memory pointer -- so that if you tried to run any of the original modules outside the Model 3 environment, they would crash. What we did was to add a routine to each of the modules to check on entry if it was running on a Model 3 or 1, and to change the relevent addresses accordingly. In the case of ABASIC, the Model 1 BASIC link addresses also had to be recalculated. The result is that the various versions of ABASIC now adjust themselves automatically to work on either machine. If you've got a Model 3 or 4, they work just like the originals; but if you have a Model 1, the new routines will kick in and they will now run for the first time.

This really was an international enterprise (Europe-Texas!). I would send files to Pieter and Mathieu, and they sent files to me, both of us making corrections in the addresses and routines of the other.

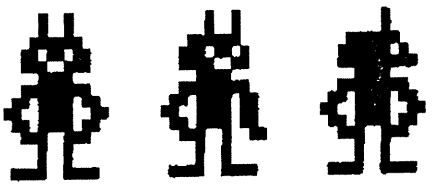
The story doesn't quite end there. We ended up with eight versions of ABASIC. The TRSDOS version now automatically selects for Model 3 TRDSOS 1.3 or Model 1 TRSDOS 2.3; we added versions for LDOS 5.1.x, DOSPLUS 3.4, and MULTIDOS 1.6, in addition to the earlier versions for LDOS 5.3, DOSPLUS 3.5, MULTIDOS 1.7, and NEWDOS/80 v. 2..

The thought naturally occurred, why not create one version that would fit ALL the major DOSes? There was no way that I was going to attempt that; I was happy just to get the separate versions. Mathieu, however, decided to go for it. He created a universal ABASIC which automatically determines whether it is being run on a Model 1 or 3, then determines what DOS and version it is running under, then makes the relevant adjustments in memory to the code, and finally runs the program. All these versions are included in the new package. Why so many? If you're just running one DOS, a specific version is the best, for it loads quickly and doesn't take up much disk space. On the other hand, if you run a lot of DOSes and want to avoid confusion, the universal ABASIC is for you. Finally, if you're thinking of writing some ABASIC programs and then distributing them to others, the universal ABASIC is ideal: you can include it with your program, so that whatever DOS version the recipient has, he'll be able to run it. Kudos to Mathieu Simons!

Next, the nitty-gritty. The GRAPHICS-90 package is still distributed at cost, namely $9.75 for the Model 3 version (this includes postage and handling); the Model 1 version is slightly higher at $11.50 (also includes shipping and handling) There are now three formats. As before, the programs are available on two Model 3 LDOS 5.3 data disks or two Model 3 TRSDOS 1.3 data disks. For Model I users without double density capacity, they are also available on four single density 35-track TRSDOS 2.3 disks. Order whichever one best suits your needs. One caveat for Model 1 users: the demonstration program, CARTOON/BAS, is so long that it takes up two single density disks. To run it on a single density system, you'll need either three drives or else you'll have to do some disk swapping while it runs. Even with this inconvenience, you'll love what you see.

One final note. A reader a few issues ago asked about how to end that demonstration program. It is an infinite loop, designed to run all day in Radio Shack stores to impress the customers. You stop it by hitting the <BREAK> key. This leaves no files open, for all its disk accesses are through machine language routines built into ABASIC.

There are no future plans for GRAPHICS-90, so if you get nervous every time you open a new copy of TRSTimes, you're probably safe for at least the next few issues...



REAL PROGRAMMERS DON'T EAT QUICHE

Humor from the TRSTimes vault

- Real programmers don't eat quiche. They like Twinkies, Coke and palate-scorching Szechen food.

- Real programmers don't write application programs. They program right down to the base metal. Application programming is for dullards who can't do system programming.

- Real programmers don't write specs. Users should be grateful for whatever they get; they are lucky to get any programs at all.

- Real programmers don't comment their code. If it was hard to write, it should be even harder to understand and modify.

- Real programmers don't document. Documentation is for simpletons who can't read listings or the object code from the dump.

- Real programmers don't draw flowcharts, Flowcharts are, after all, the illiterate's form of documentation. Cavemen drew flowcharts; look how much good it did them.

- Real programmers don't read manuals. Reliance on a reference is the hallmark of a novice and a coward.

- Real programmers don't write in RPG. RPG is for gum-chewing dimwits who maintain ancient payroll programs.

- Real prorammers don't write in COBOL COBOL is for COmmon Business Oriented Laymen who can run neither a business nor a real program.

- Real programmers don't write in FORTRAN. FORTRAN is for wimp engineers who wear white socks. They get excited over finite state analysis and nuclear reactor simulation.

- Real programmers don't write in PASCAL, ADA, BLISS, or any of those other sissy computer languages. Strong typing is a crutch for people with weak memories.

- Real programmers don't write in PL/1 PL/1 is for insecure anal retentives who can't choose between COBOL and FORTRAN.

- Real programmers' programs never work right the first time. But if you throw them on the machine they can be patched into working order in "only a few" 30 hour debugging sessions.

- Real programmers never work 9 to 5. If any real programmers are around at 9 a.m., it is because they were up all night long.

- Real programers don't play tennis, or any other sport which requires a change of clothes. Mountain climbing is OK, and real programmers wear climbing boots to work in case a mountain should suddenly spring up in the middle of the machine room.

- Real programmers distain structured programming. Structured programming is for compulsive neurotics who were prematurely toilet-trained. They wear neckties and carefully line up sharp pencils on an other wise clear desk.

- Real programmers don't like the Team programming concept. Unless, of course, they are the Chief Programmer.

- Real programmers never "write" memos on paper. They "send" memos via EMAIL.

- Real programmers have no use for managers, Managers are a necessary evil. They exist only to deal with personnel bozos, bean counters, senior planners, and other mental midgets.

- Real programmers scorn floating point arithmetic. The decimal point point was invented for pansy bedwetters who were unable to think big.

- Real programmers don't believe in schedules. Planners make up schedules. Managers "firm up" schedules. Frightened coders strive to meet schedules. Real programmers ignore schedules.

- Real programmers don't bring brown-bag lunches. If the vending machine sells it, they eat it. If the vending machine doesn't sell it, they don't eat it. Vending machines don't sell quiche.

VIDEO GAME PROTOTYPE

Model 4 - Hi res

by J.F.R. Slinkman

Are Nintendo- and Sega-type arcade games possible on the TRS-80? Well, if you're not too picky -- especially about color -- then the answer is "yes."

Actually, the TRS-80 640 x 240 high resolution graphics board affords us more detailed graphics than either the old 8-bit Nintendo or Sega games, which were limited to 320 x 240 graphics or less. Unfortunately, our 640 x 240 x 2 resolution doesn't look too good compared to the 640 x 480 x 32K capabilities of the new 16-bit video games.

The enclosed listings explain how to go about creating the animation for a video game, along with some suggestions for a standardized animation data format for such games.

The basic procedure for this kind of animation is:

- 1. Keep a copy of the game background graphics data in computer RAM.
- 2. Keep all animation graphics in computer RAM.
- 3. Punch a "hole" in the background where the animation image will appear.
- 4. Fill the "hole" with the animation image.
- 5. Replace the affected area of the display on the screen with the new data created in steps 3 and 4 above.

To see how this is done, look at the data arrangement in BILL/DAT. This file contains images of the character, "Bill," in various different positions, namely front, back, right, right-step-going-right, left-step-going-left, etc.

Admittedly, this data is rather crude, and does not produce particularly smooth motion. For example, when "Bill" moves right, there are four images each offset two pixels from the previous image. First, he's simply facing right. Second, he's taking a step to the right with his right foot. Third, facing right. Fourth, taking a step to the right with his left foot.

The motion would be a lot smoother if there were eight frames per cycle, each offset one pixel from the previous, with a display rate of 20 frame/sec.

Also, I learned that even black portions of the image ("Bill's" shoes, in this case) should be outlined in white, as they "disappear" when "Bill" is walking through a black area of background. For this reason, it's a good idea to put a mostly non-black image on the screen before running ANIMATE/CMD.

The first two bytes for each image are 6 and 48. This means each image is 6 bytes (48 pixels) wide, and 48 lines high. The data is arranged column by column, with each column requiring 48 bytes of both "mask" and "image" data for each column. Due to way graphics board RAM is addressed, this is much more efficient than arranging the data row by row.

What the program must do with this data is keep track of the location where "Bill" must be displayed, and react to keyboard or joystick input by displaying different images of "Bill" in different positions on the screen.

It does this by coordinating the screen location with the background image data in RAM, reading background data from RAM, punching a "hole" in it by ANDing background and mask data and writing the result to a RAM buffer, then inserting the animation image by ORing that result with the image data, and writing the result to the display.

Now we're ready to take a look at the listing, ANIMATE/ASM.

First, we allocate 19,200 bytes of RAM for the background image, and 240 bytes (the size of one full graphics board column) for the ANDing and ORing work buffer. We also initialize some one-byte values for things like image position, and keeping track of which was the last animation image shown in an animation cycle to keep the movement as smooth as possible.

Next is a medium priority background task labeled "TASK," which applies a brake to keep animation from proceeding too quickly. In this program, animation can be done at the rates of 30, 15, 10, 7.5 and 6 frames per second, depending on the value loaded into COUNTER and TIMER.

(For smoother animation, you would want to double the number of images, as described above, and install this task in task slot 11, which would allow the animation to proceed twice as fast {i.e., 60, 30, 20, 15 and 12 frames/sec}).

Obviously, changing the animation rate would be one way of increasing the difficulty of a game as the player proceeds through various game stages.

At BEGIN, whatever image is on the screen is copied to the IMAGE buffer, and becomes the "background" through which "Bill" will be moved about.

In an actual game program, the background(s) would probably be loaded from disk, possibly in a compressed format, into RAM, and then copied to the graphics board RAM.

Additional animation data could be stored on disk, and loaded as needed. Obviously, program speed considerations require all animation data used in each game stage to be in RAM.

Also, our hardware pretty much restricts us to a maximum of one 640 x 240 background per game stage, as moving the background would be far too slow. However, if you keep the background simple, and only move small portions of it (clouds, for example, or other small objects), it might be possible to make the background APPEAR to move.

Next the animation braking task is installed in task slot 8, and the "front" image of "Bill" is displayed on the screen.

Now the keyboard and joystick port are polled for user input. For the keyboard, this is done via the more-or-less undocumented "code 255" function of the @CTL supervisor call (SVC). It loads an 8-byte bit image of the keyboard to the RAM area pointed to by the IY register. Upon return, the IY register points to the first byte past the end of the buffer (i.e., KBDMAP + 8).

First, the byte mapped to keys "0" through "7" is checked. If any of the keys "1" through "7" are pressed, the corresponding bit in (IY-4 = KBDMAP + 4) will be set. ANDing this byte with 3EH isolates bits 5 through 1, which correspond to keys "5" through "1." If any of these bits are set, the value is decoded, and the value in TIMER altered accordingly.

Now the joystick port is read (after loading B with a non-zero value to force a read from external port 00H, instead of internal port 00H, on XLR8er-equipped Model 4s). The port image is then complemented and stored in the E register.

Next the keyboard image at KBDMAP + 6 (IY-2) is read, and shifted right three times so the bits for the up, down, left and right arrow keys and space bar will line up with the bits for up, down, left, right and fire button from the joystick port image.

The keyboard and joystick data are then merged by ORing them together. If there was no valid input, a branch is made to GETKEY to try again.

Once valid joystick or keyboard input is detected, the FLAG byte in the animation task is polled until a non-zero value is read, at which time the -1 flag value is changed to zero and the down-counter is reset before the program proceeds.

At this point at least one of the bits 4 through 0 must be set, and they tell the DISPLAY subroutine what to do. The branching is done via calculated jump address, as this makes it easier to invoke subroutines to perform complicated action, such as having the character go through a 20-frame jump sequence if a certain key combination is detected, for example.

However, in this simple demo, if the only set bit is bit 4, the space bar/fire button bit, the program terminates.

The code at each of the eight directions performs tests to make sure the image does not move off the screen, and updates XPOS and YPOS to tell the DISPLAY routine where to place the next image.

Finally, at DISPLAY, the reading and processing the width, height, mask and image data is performed.

The width and height values become counters in the processing loop, which reads the background bytes from IMAGE, ANDs them with the mask data while writing the resulting column of data to WRKBUF. Then the animation data is merged in via OR instructions, and the result is sent to the graphics board for display.

The concept is actually pretty simple, but then I'm much better at programming than at art.

Hopefully, someone more artistically inclined than I can come up with some ideas for good video games for the Model 4, and use the enclosed code to produce a finished game product.

Alternatively, if a TRSTimes Reader wants to do the art, and leave the programming to me, I'd be happy to cooperate in such a project. I can be reached at: 1511 Old Compton Road, Richmond, Va. 23233, or CompuServe 72411,650

ANIMATE/ASM

```
;
@GTDCB  EQU  82
@ADTSK  EQU  29
@CTL    EQU  5
@MUL16  EQU  91
@RMTSK  EQU  30
@CKBRKC EQU 106
@EXIT   EQU  22
COMTROL EQU 83H
XREG    EQU  80H
```

```
YREG    EQU  81H
GFXDAT  EQU  82H
;
        ORG  3000H
;
IMAGE   DS   19200
WRKBUF  DS   240
KBDMAP  DS   8
XPOS    DB   37
YPOS    DB   96
XTEMP   DB   0
YTEMP   DB   0
RYTCYC  DB   -1
LEFTCYCDB    -1
UPCYCLE      DB      -1
DOWNCYC      DB      -1
;
*GET    BILL/DAT
;
TASK    DW   WAIT          ;called every 1/30th
                           ;of a second
COUNTER DB   2             ;IX+2
FLAG    DB   0             ;IX+3
WAIT    DEC  (IX+2)
        RET  NZ            ;if COUNTER > 0
        LD   (IX+2),2      ;else reset COUNTER
TIMER   EQU  $-1
        LD   (IX+3),-1     ;  and set flag every
                           ;TIMER/30 seconds
        RET
;
;
BEGIN   LD   DE,'IK'       ;get address of *ki drvr
        LD   A,@GTDCB
        RST  40
        LD   (KIDCB),HL    ;store for keyboard scan
;
; copy gfx board RAM to computer RAM @ IMAGE
;
        DI
        LD   A,11010111B   ;gfx on, txt off,
                           ;auto inc Y on read
        OUT  (CONTROL),A   ;CONTROL = 83H
        LD   HL,IMAGE
        LD   BC,XREG       ;XREG = 80H
        LD   E,0           ;x position
        LD   A,80
BGN010  OUT  (C),E         ;send E out port 80H
        INC  C
        OUT  (C),B         ;send 0 out port 81H
        INC  C             ;C = 82H
        LD   B,240
        INIR               ;read column to image
                           ;buffer
        DEC  C
        DEC  C             ;C = 80H
        INC  E
        DEC  A
        JR   NZ,BGN010
;
        LD   DE,TASK
        LD   C,8
        LD   A,@ADTSK      ;install timing brake task
        RST  40
        EI
;
        LD   A,01110111B   ;gfx on, txt off, inc Y on write
        OUT  (CONTROL),A
        LD   HL,FRONT
        JP   DISPLAY       ;put Bill in middle of screen
;
GETKEY  LD   DE,$-$        ;copy kbd map to KBDMAP
KIDCB   EQU  $-2
        LD   C,255
        LD   IY,KBDMAP
        LD   A,@CTL
        RST  40
        LD   A,(IY-4)      ;p/u byte for keys
                           ;'0' - '7'
        AND  3EH           ;mask out bits for '7',
                           ;'6' & '0'
        JR   Z,GTK030      ;go if '1' thru '5' not
                           ;pressed
        LD   C,0
GTK010  RRA
        JR   C,GTK020
        INC  C
        JR   GTK010
GTK020  LD   A,C           ;A = one less than # of
                           ;shifts to CY
        DI                 ; (e.g., 02H becomes 1;
                           ;20H becomes 5)
        LD   (TIMER),A     ;change brake task
                           ;delay time
        EI
;
GTK030  LD   BC,0FF00H     ;B < > 0 forces extern,
                           ;C = joystick port #
        IN   A,(C)         ;read joystick port
        CPL                ;reverse bits
        LD   E,A           ;and store result in E
        LD   A,(IY-2)      ;get keyboard image of
                           ;arrows + space, etc.
        SRL  A             ;lose ENTER bit
        SRL  A             ;lose CLEAR bit
        SRL  A             ;shift BREAK bit to
                           ;C flag
        JP   C,EXIT        ;go if break was
                           ;pressed
        OR   E             ;merge keyboard and
                           ;joystick input
        JR   Z,GETKEY      ;go if no input
        PUSH AF            ;  else save input data
;
GTK050  LD   A,(FLAG)
        OR   A
```

```
        JR      Z,GTK050        ;wait for task to OK next
                                ;graphics write
        DI
        INC     A
        LD      (FLAG),A        ;reset FLAG
        LD      A,(TIMER)
        LD      (COUNTER),A     ;reset COUNTER
        EI
;
        POP     AF              ;restore input data
        LD      HL,DISPLAY      ;RET address for all
                                ;branches
        PUSH    HL
;
; The bits in A have the following meanings:
;       bit 0 = north           (up)
;          1 = south            (down)
;          2 = west             (left)
;          3 = east             (right)
;          4 = fire             (fire button/space bar)
;        5-7 - not used
;
        SLA     A               ;each address is 2 bytes
        LD      L,A
        LD      H,0
        LD      DE,GTK060       ;address table base
        ADD     HL,DE
        LD      A,(HL)          ;p/u lsb
        INC     HL
        LD      H,(HL)          ;p/u msb
        LD      L,A
        JP      (HL)
;
GTK060  DW      NOTHING         ;00000 = no keyboard/
                                ;joystick input
        DW      NORTH           ;00001 = N
        DW      SOUTH           ;00010 = S
        DW      NOTHING         ;00011 = S-N
                                ;(not used in this
                                ;program)
        DW      WEST            ;00100 = W
        DW      NORTHWEST       ;00101 = W-N
        DW      SOUTHWEST       ;00110 = W-S
        DW      WEST            ;00111 = W-S-N
        DW      EAST            ;01000 = E
        DW      NORTHEAST       ;01001 = E-N
        DW      SOUTHEAST       ;01010 = E-S
        DW      EAST            ;01011 = E-S-N
        DW      NOTHING         ;01100 = E-W
                                ;(not used in this
                                ;program)
        DW      NORTH           ;01101 = E-W-N
        DW      SOUTH           ;01110 = E-W-S
        DW      NOTHING         ;01111 = E-W-S-N
                                ;(not used in this
                                ;program)
        DW      FIRE            ;10000 = fire only
        DW      NORTH           ;10001 = F-N
        DW      SOUTH           ;10010 = F-S
        DW      FIRE            ;10011 = F-S-N
        DW      WEST            ;10100 = F-W
        DW      NORTHWEST       ;10101 = F-W-N
        DW      SOUTHWEST       ;10110 = F-W-S
        DW      WEST            ;10111 = F-W-S-N
        DW      EAST            ;11000 = F-E
        DW      NORTHEAST       ;11001 = F-E-N
        DW      SOUTHEAST       ;11010 = F-E-S
        DW      EAST            ;11011 = F-E-S-N
        DW      FIRE            ;11100 = F-E-W
        DW      NORTH           ;11101 = F-E-W-N
        DW      SOUTH           ;11110 = F-E-W-S
        DW      FIRE            ;11111 = F-E-W-S-N
;
NOTHING POP     HL              ;clear RET addr off stack
        JP      GETKEY
;
FIRE    POP     HL              ;clear stack
        JP      EXIT
;
NORTH   LD      A,-1
        LD      (RYTCYC),A
        LD      (LEFTCYC),A
        LD      (DOWNCYC),A
        LD      A,(UPCYCLE)
        INC     A
        AND     3
        LD      (UPCYCLE),A
        BIT     0,A
        JR      NZ,NTH010
        LD      HL,BACK
        JR      NTH030
NTH010  BIT     1,A
        JR      NZ,NTH020
        LD      HL,RSTEPB
        JR      NTH030
NTH020  LD      HL,LSTEPB
NTH030  LD      A,(YPOS)
        OR      A
        RET     Z               ;no change if already
                                ;at top
        SUB     3
        LD      (YPOS),A        ;else move three lines up
        OUT     (YREG),A
        RET
;
SOUTH   LD      A,-1
        LD      (RYTCYC),A
        LD      (LEFTCYC),A
        LD      (UPCYCLE),A
        LD      A,(DOWNCYC)
        INC     A
        AND     3
        LD      (DOWNCYC),A
        BIT     0,A
        JR      NZ,STH010
        LD      HL,FRONT
```

```
        JR      STH030
STH010  BIT     1,A
        JR      NZ,STH020
        LD      HL,RSTEPF
        JR      STH030
STH020  LD      HL,LSTEPF
STH030  LD      A,(YPOS)
        CP      195
        RET     NC              ;no change if already
                                ;at bottom
        ADD     A,3
        LD      (YPOS),A        ;else move three lines up
        OUT     (YREG),A
        RET
;
NORTHEAST
        CALL    NTH030
        JR      EAST
;
SOUTHEAST
        CALL    STH030
;
EAST    LD      A,(XPOS)
        CP      74
        JR      Z,EST010        ;no change if already
                                ;max right
        INC     A
        LD      (XPOS),A        ;else move one byte right
EST010  LD      A,-1
        LD      (LEFTCYC),A
        LD      (UPCYCLE),A
        LD      (DOWNCYC),A
        LD      A,(RYTCYC)
        INC     A
        AND     3
        LD      (RYTCYC),A
        BIT     0,A
        JR      NZ,EST020
        LD      HL,RIGHT
        RET
EST020  BIT     1,A
        JR      NZ,EST030
        LD      HL,RSTEPR
        RET
EST030  LD      HL,LSTEPR
        RET
;
SOUTHWEST
        CALL    STH030
        JR      WEST
;
NORTHWEST
        CALL    NTH030
;
WEST    LD      A,(XPOS)
        OR      A
        JR      Z,WST010        ;no change if already
                                ;max left
        DEC     A
        LD      (XPOS),A        ;else move one byte left
WST010  LD      A,-1
        LD      (RYTCYC),A
        LD      (UPCYCLE),A
        LD      (DOWNCYC),A
        LD      A,(LEFTCYC)
        INC     A
        AND     3
        LD      (LEFTCYC),A
        BIT     0,A
        JR      NZ,WST020
        LD      HL,LEFT
        RET
WST020  BIT     1,A
        JR      NZ,WST030
        LD      HL,RSTEPL
        RET
WST030  LD      HL,LSTEPL
        RET
;
DISPLAY LD      B,(HL)          ;p/u # of columns
        INC     HL
        LD      C,(HL)          ;p/u # of bytes/column
        INC     HL              ;HL -> data
        PUSH    HL
        PUSH    BC
        LD      HL,(XPOS)       ;L = X, H = Y
        LD      (XTEMP),HL
        LD      B,H             ;store Y value
        LD      H,0             ;HL = X value
        LD      C,240           ;multiply * # bytes
                                ;/column
        LD      A,@MUL16
        RST     40
        LD      H,L             ;MSB to H
        ADD     A,B             ;add Y value to LSB
        LD      L,A             ;put sum in LSB of HL
        LD      A,H             ;p/u MSB
        ADC     A,30H           ;add base address of
                                ;table + CY from LSB
                                ;add
        LD      H,A             ;and stuff in MSB
        EX      DE,HL           ;DE -> up-left byte of
                                ;background in RAM
        POP     BC              ;counters
        POP     HL              ;HL -> animation data
;
DSP010  PUSH    BC              ;save column counter
                                ;in B
        LD      IX,WRKBUF
        PUSH    BC              ;save # bytes in C
        PUSH    BC              ;  ...twice
DSP020  LD      A,(DE)          ;p/u background byte
        INC     DE
        AND     (HL)            ;apply mask to punch
                                ;"hole" in background
        INC     HL
```

```
        LD      (IX),A
        INC     IX
        DEC     C
        JR      NZ,DSP020       ;repeat masking until
                                ;for C bytes
;
        POP     BC              ;# of bytes in C
        PUSH    DE
        LD      DE,WRKBUF
DSP030  LD      A,(DE)          ;p/u masked back-
                                ;ground byte
        OR      (HL)            ;add animation data
        LD      (DE),A          ;and return to buffer
        INC     HL
        INC     DE
        DEC     C
        JR      NZ,DSP030
        POP     DE
;
        LD      A,(XTEMP)
        OUT     (XREG),A
        INC     A
        LD      (XTEMP),A
        LD      A,(YTEMP)
        OUT     (YREG),A
        POP     BC              ;# of bytes in C
        LD      B,C
        LD      C,GFXDAT
        PUSH    HL
        LD      HL,WRKBUF
        OTIR                    ;write buffer data to
                                ;gfx board
        POP     HL
;
        POP     BC              ;B = # columns,
                                ;C = # of bytes
        LD      A,240
        SUB     C               ;A = # of bytes offset
                                ;to top of next backgnd
        ADD     A,E             ;column in RAM
        LD      E,A
        LD      A,D
        ADC     A,0
        LD      D,A             ;DE -> next background
                                ;column in RAM
;
        DJNZ    DSP010
        JP      GETKEY
;
EXIT    LD      A,11111100B     ;gfx off, text on
        OUT     (CONTROL),A
        LD      C,8             ;remove timer task
        LD      A,@RMTSK
        RST     40
        LD      A,@CKBRKC
        RST     40
        LD      HL,0            ;return to DOS ready
        LD      A,@EXIT
        RST     40
;
        END     BEGIN
```

BILL/DAT

```
*LIST OFF
;
FRONT   DB      6,48
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-2,-8,240,-8,-2,-1,-1,-1
        DB      -1,-1,-1,240,224,192,80H,80H,80H,80H,80H,
                80H,80H,80H,80H,80H
        DB      -1,-1,-2,-2,-2,-2,-4,-4,-4,-4,-4,-4,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0
        DB      0,0,0,0,7,15,30,30,30,30,30,30,30,30,21,21
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,192,80H,0,0,0,0,0,0,0,80H,224
        DB      240,240,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,1,1,1,1,3,3,3,3,3,3,-1,-1,-1,-1
        DB      0,0,0,0,0,31,63,63,99,7EH,7DH,7FH,73H,
                56,31,7
        DB      3,3,7,-1,-1,-1,7FH,7FH,7FH,7FH,63,63,63,
                63,63,63
        DB      63,62,78H,78H,78H,78H,240,240,240,90H,
                0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,3,1,0,0,0,0,0,0,0,0,1,7
        DB      15,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,80H,80H,80H,80H,192,192,192,192,
                192,192,-1,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-4,198,7EH,190,-2,206,28,
                -8,224
        DB      192,192,224,-1,-1,-1,-2,-2,-2,-2,-4,-4,-4,-4,
                -4,-4
        DB      -4,7CH,30,30,30,30,15,15,15,9,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,7FH,31,15,31,7FH,
                -1,-1,-1
        DB      -1,-1,-1,15,7,3,1,1,1,1,1,1,1,1,1,1
        DB      -1,-1,7FH,7FH,7FH,7FH,63,63,63,63,63,63,
                -1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,64,0,0,0,0,0
        DB      0,0,0,0,224,240,78H,78H,78H,78H,78H,78H,
                78H,78H,168,168
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
```

```
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
BACK    DB      6,48
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-2,-8,240,
                -8,-2,-1,-1,-1
        DB      -1,-1,-1,240,224,192,80H,80H,80H,
                80H,80H,80H,80H,80H,80H,80H,80H
        DB      -1,-1,-2,-2,-2,-2,-4,-4,-4,-4,-4,-4,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0
        DB      0,0,0,0,7,15,30,30,30,30,30,30,30,
                30,21,21
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,192,80H,0,0,0,0,0,0,0,
                80H,224
        DB      240,240,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,1,1,1,1,3,3,3,3,3,3,-1,-1,-1,-1
        DB      0,0,0,0,0,31,63,63,7FH,-1,-1,-1,7FH,
                63,31,7
        DB      3,3,7,-1,-1,-1,7FH,7FH,7FH,7FH,63,
                63,63,63,63,63
        DB      63,62,78H,78H,78H,78H,240,240,240,
                240,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,3,1,0,0,0,0,0,0,0,0,1,7
        DB      15,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,80H,80H,80H,80H,192,192,192,
                192,192,192,-1,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-4,-2,-1,-1,-1,-2,-4,-8,224
        DB      192,192,224,-1,-1,-1,-2,-2,-2,-2,-4,-4,-4,
                -4,-4,-4
        DB      -4,7CH,30,30,30,30,15,15,15,15,0,0,0,
                0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,7FH,31,15,31,
                7FH,-1,-1,-1
        DB      -1,-1,-1,15,7,3,1,1,1,1,1,1,1,1,1,1
        DB      -1,-1,7FH,7FH,7FH,7FH,63,63,63,63,
                63,63,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,192,0,0,0,0,0
        DB      0,0,0,0,224,240,78H,78H,78H,78H,78H,
                78H,78H,78H,78H,168,168
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
RIGHT   DB      6,48
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-2,-2,-2,-2,-2,-2,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-2,-2,-2,-2,-2,-2,-2,-2,-2,-2
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,224,80H,0,0,0,0,0,0,0,80H,192
        DB      224,224,192,80H,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,80H,80H,80H,80H,80H,192,192,192,
                -1,-1,-1,-1
        DB      0,0,0,0,0,31,63,7FH,7FH,78H,73H,78H,7FH,
                63,31,15
        DB      7,7,15,31,63,7FH,73H,73H,73H,73H,73H,73H,
                79H,79H,7AH,7AH
        DB      63,63,63,63,31,31,31,31,31,7,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,1,0,0,0,0,0,0,0,0,0,0,0
        DB      0,15,1,0,0,0,0,0,0,0,0,0,0,0,1,1
        DB      1,1,3,3,3,3,7,7,7,3,0,0,-1,-1,-1,-1
        DB      0,0,0,0,0,-2,-1,-1,226,-1,-1,-1,231,240,-1,-2
        DB      224,192,224,-4,-2,-1,231,231,231,231,
                206,206,204,204,168,168
        DB      -8,-8,240,240,240,240,224,224,224,0,0,0,0,
                0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,7FH,63,63,7FH,63,15,15,63,63,
                63,7FH
        DB      -1,-1,-1,-1,7FH,63,63,63,63,63,7FH,7FH,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,31,15,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,192,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

```
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
RSTEPR  DB      6,48            ;right step going right
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-2,-2,-2,-2,-2,-2,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-2,-2,-2,-2,-2,-2,-2,-2,
                -2,-2
        DB      -2,-2,-2,-4,-8,240,224,192,80H,80H,
                -8,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,1,3,7,7,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,224,80H,0,0,0,0,0,0,0,0,
                80H,192
        DB      224,224,192,80H,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,8,12,2,1,240,-1,-1,-1,-1
        DB      0,0,0,0,0,31,63,7FH,7FH,78H,73H,
                78H,7FH,63,31,15
        DB      7,7,15,31,63,7FH,73H,73H,73H,73H,
                73H,73H,79H,79H,7AH,7AH
        DB      7FH,7FH,63,9FH,207,231,227,81H,0,0,
                0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,1,0,0,0,0,0,0,0,0,0,0,0
        DB      0,15,1,0,0,0,0,0,0,0,0,0,0,0,1,1
        DB      1,1,1,1,1,1,0,0,0,0,80H,71,-1,-1,-1,-1
        DB      0,0,0,0,0,-2,-1,-1,226,-1,-1,-1,231,240,
                -1,-2
        DB      224,192,224,-4,-2,-1,231,231,231,231,
                206,206,204,204,168,168
        DB      -8,-8,-8,-8,-8,-8,-4,-2,-8,48,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,7FH,63,63,7FH,63,15,15,
                63,63,63,7FH
        DB      -1,-1,-1,-1,7FH,63,63,63,63,63,7FH,
                7FH,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,0,0,3,7FH,-1,-1,-1,
                -1,-1
        DB      0,0,0,0,0,0,0,0,0,0,192,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
```

```
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
LSTEPRDB6,48;left step going right
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-2,-2,-2,-2,-2,-2,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-2,-2,-2,-2,-2,-2,-2,-2,-2,-2
        DB      -2,-2,-2,-4,-8,240,224,192,80H,80H,-8,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,1,3,7,7,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,224,80H,0,0,0,0,0,0,0,0,80H,192
        DB      224,224,192,80H,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,8,12,2,1,240,-1,-1,-1,-1
        DB      0,0,0,0,0,31,63,7FH,7FH,78H,73H,78H,7FH,
                63,31,15
        DB      7,7,15,31,63,7FH,73H,73H,73H,73H,73H,73H,
                79H,79H,7AH,7AH
        DB      7FH,7FH,7FH,-1,-1,-4,243,193,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,1,0,0,0,0,0,0,0,0,0,0,0
        DB      0,15,1,0,0,0,0,0,0,0,0,0,0,0,1,1
        DB      1,1,1,1,1,1,0,0,0,0,80H,71,-1,-1,-1,-1
        DB      0,0,0,0,0,-2,-1,-1,226,-1,-1,-1,231,240,-1,-2
        DB      224,192,224,-4,-2,-1,231,231,231,231,
                206,206,204,204,168,168
        DB      240,224,200,98H,56,-8,-4,-2,-8,48,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,7FH,63,63,7FH,63,15,15,63,
                63,63,7FH
        DB      -1,-1,-1,-1,7FH,63,63,63,63,63,7FH,7FH,
                -1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,0,0,3,7FH,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,192,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
```

```
LEFT    DB      6,48
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-2,-4,-4,-2,-4,240,240,
                -4,-4,-4,-2
        DB      -1,-1,-1,-1,-2,-4,-4,-4,-4,-4,-2,-2,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-8,240,
                -1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,80H,0,0,0,0,0,0,0,0,0,0,0
        DB      0,240,80H,0,0,0,0,0,0,0,0,0,0,80H,80H
        DB      80H,80H,192,192,192,192,224,224,224,
                192,0,0,-1,-1,-1,-1
        DB      0,0,0,0,0,7FH,-1,-1,71,-1,-1,-1,231,
                15,-1,7FH
        DB      7,3,7,63,7FH,-1,231,231,231,231,73H,
                73H,51,51,21,21
        DB      31,31,15,15,15,15,7,7,7,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,7,1,0,0,0,0,0,0,0,0,1,3
        DB      7,7,3,1,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,1,1,1,1,1,3,3,3,-1,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-2,-2,30,206,30,-2,-4,-8,
                240
        DB      224,224,240,-8,-4,-2,206,206,206,206,
                206,206,9EH,9EH,94,94
        DB      -4,-4,-4,-4,-8,-8,-8,-8,-8,224,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,7FH,7FH,7FH,7FH,
                7FH,7FH,-1,-1,-1
        DB      -1,-1,-1,-1,-1,7FH,7FH,7FH,7FH,7FH,
                7FH,7FH,7FH,7FH,7FH,7FH
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
RSTEPL  DB      6,48            ;right step going left
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-2,-4,-4,-2,-4,240,240,-4,-4,-4,-2
        DB      -1,-1,-1,-1,-2,-4,-4,-4,-4,-4,-2,-2,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,0,0,192,-2,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,3,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,80H,0,0,0,0,0,0,0,0,0,0,0
        DB      0,240,80H,0,0,0,0,0,0,0,0,0,0,80H,80H
        DB      80H,80H,80H,80H,80H,80H,80H,0,0,0,0,1,226,
                -1,-1,-1,-1
        DB      0,0,0,0,0,7FH,-1,-1,71,-1,-1,-1,231,15,-1,7FH
        DB      7,3,7,63,7FH,-1,231,231,231,231,
                73H,73H,51,51,21,21
        DB      15,7,19,25,28,31,63,7FH,31,12,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,7,1,0,0,0,0,0,0,0,0,1,3
        DB      7,7,3,1,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,16,48,64,80H,15,-1,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-2,-2,30,206,30,-2,-4,-8,240
        DB      224,224,240,-8,-4,-2,206,206,206,206,
                206,206,9EH,9EH,94,94
        DB      -2,-2,-2,-1,-1,63,207,83H,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,7FH,7FH,7FH,7FH,7FH,
                7FH,-1,-1,-1
        DB      -1,-1,-1,-1,-1,7FH,7FH,7FH,7FH,7FH,7FH,
                7FH,7FH,7FH,7FH,7FH
        DB      7FH,7FH,7FH,63,31,15,7,3,1,1,31,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,80H,192,224,224,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
LSTEPL  DB      6,48            ;left step going left
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
```

```
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-2,-4,-4,-2,-4,240,240,
                -4,-4,-4,-2
        DB      -1,-1,-1,-1,-2,-4,-4,-4,-4,-4,-2,-2,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,0,0,192,-2,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,80H,0,0,0,0,0,0,0,0,0,0,0
        DB      0,240,80H,0,0,0,0,0,0,0,0,0,0,0,80H,80H
        DB      80H,80H,80H,80H,80H,80H,80H,0,0,0,0,1,
                226,-1,-1,-1,-1
        DB      0,0,0,0,0,7FH,-1,-1,71,-1,-1,-1,231,15,
                -1,7FH
        DB      7,3,7,63,7FH,-1,231,231,231,231,73H,
                73H,51,51,21,21
        DB      31,31,31,31,31,31,63,7FH,31,12,0,0,
                0,0,0,0
;
        DB      -1,-1,-1,-1,7,1,0,0,0,0,0,0,0,1,3
        DB      7,7,3,1,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,16,48,64,80H,15,-1,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-2,-2,30,206,30,-2,-4,
                -8,240
        DB      224,224,240,-8,-4,-2,206,206,206,206,
                206,206,9EH,9EH,94,94
        DB      -2,-2,-4,-7,243,231,199,81H,0,0,0,0,0,
                0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,7FH,7FH,7FH,7FH,
                7FH,7FH,-1,-1,-1
        DB      -1,-1,-1,-1,-1,7FH,7FH,7FH,7FH,7FH,
                7FH,7FH,7FH,7FH,7FH,7FH
        DB      7FH,7FH,7FH,63,31,15,7,3,1,1,31,-1,
                -1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,80H,192,224,224,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

RSTEPF  DB      6,48            ;right step going forward
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-2,-8,240,-8,-2,-1,-1,-1
        DB      -1,-1,-1,240,224,192,80H,80H,80H,80H,
                80H,80H,80H,80H,80H,80H
        DB      -1,-1,-2,-2,-2,-2,-2,-4,-4,-4,-4,-4,-4,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0
        DB      0,0,0,0,7,15,30,30,30,30,30,30,30,30,21,21
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,192,80H,0,0,0,0,0,0,0,80H,224
        DB      240,240,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,1,1,1,1,1,3,3,3,3,3,3,-1,-1,-1
        DB      0,0,0,0,0,31,63,63,99,7EH,7DH,7FH,73H,
                56,31,7
        DB      3,3,7,-1,-1,-1,7FH,7FH,7FH,7FH,63,63,63,
                63,63,63
        DB      63,62,78H,78H,78H,78H,78H,240,240,240,
                90H,0,0,0,0,0
;
        DB      -1,-1,-1,-1,3,1,0,0,0,0,0,0,0,1,7
        DB      15,15,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,80H,80H,80H,192,192,192,192,192,192,
                -1,-1,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-4,198,7EH,190,-2,206,28,
                -8,224
        DB      192,192,224,-1,-1,-1,-2,-2,-2,-2,-4,-4,-4,-4,-4,-4
        DB      -4,7CH,30,30,30,15,15,15,9,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,7FH,31,15,31,7FH,
                -1,-1,-1
        DB      -1,-1,-1,15,7,3,1,1,1,1,1,1,1,1,1,1
        DB      -1,-1,7FH,7FH,7FH,63,63,63,63,63,63,
                -1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,64,0,0,0,0,0
        DB      0,0,0,0,224,240,78H,78H,78H,78H,78H,
                78H,78H,78H,168,168
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
RSTEPB  DB      6,48            ;right step going back
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
```

```
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-2,-8,240,-8,-2,
                -1,-1,-1
        DB      -1,-1,-1,240,224,192,80H,80H,80H,80H,80H,
                80H,80H,80H,80H,80H,80H
        DB      -1,-1,-2,-2,-2,-2,-2,-4,-4,-4,-4,-4,-4,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0
        DB      0,0,0,0,7,15,30,30,30,30,30,30,30,30,
                21,21
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,192,80H,0,0,0,0,0,0,0,0,
                80H,224
        DB      240,240,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,1,1,1,1,1,1,3,3,3,3,3,3,-1,-1,-1
        DB      0,0,0,0,0,31,63,63,7FH,-1,-1,-1,7FH,
                63,31,7
        DB      3,3,7,-1,-1,-1,7FH,7FH,7FH,7FH,63,
                63,63,63,63,63
        DB      63,62,78H,78H,78H,78H,78H,240,240,
                240,240,0,0,0,0,0
;
        DB      -1,-1,-1,-1,3,1,0,0,0,0,0,0,0,0,1,7
        DB      15,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,80H,80H,80H,192,192,192,192,
                192,192,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-4,-2,-1,-1,-1,-2,-4,-8,224
        DB      192,192,224,-1,-1,-1,-2,-2,-2,-2,-4,-4,-4,
                -4,-4,-4
        DB      -4,7CH,30,30,30,15,15,15,15,0,0,0,0,0,
                0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,7FH,31,15,31,
                7FH,-1,-1,-1
        DB      -1,-1,-1,15,7,3,1,1,1,1,1,1,1,1,1,1
        DB      -1,-1,7FH,7FH,7FH,63,63,63,63,63,63,
                -1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,192,0,0,0,0,0
        DB      0,0,0,0,224,240,78H,78H,78H,78H,
                78H,78H,78H,78H,168,168
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;

LSTEPF  DB      6,48            ;left step going forward
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-2,-8,240,-8,-2,-1,-1,-1
        DB      -1,-1,-1,240,224,192,80H,80H,80H,80H,80H,
                80H,80H,80H,80H,80H,80H
        DB      -1,-1,-2,-2,-2,-4,-4,-4,-4,-4,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0
        DB      0,0,0,0,7,15,30,30,30,30,30,30,30,30,21,21
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,192,80H,0,0,0,0,0,0,0,80H,224
        DB      240,240,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,1,1,1,3,3,3,3,3,3,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,31,63,63,99,7EH,7DH,7FH,73H,
                56,31,7
        DB      3,3,7,-1,-1,-1,7FH,7FH,7FH,7FH,63,63,63,
                63,63,63
        DB      63,62,78H,78H,78H,240,240,240,90H,0,0,
                0,0,0,0,0
;
        DB      -1,-1,-1,-1,3,1,0,0,0,0,0,0,0,1,7
        DB      15,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,80H,80H,80H,80H,80H,192,192,192,
                192,192,192,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-4,198,7EH,190,-2,206,28,
                -8,224
        DB      192,192,224,-1,-1,-1,-2,-2,-2,-2,-4,-4,-4,-4,
                -4,-4
        DB      -4,7CH,30,30,30,30,30,15,15,15,9,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,7FH,31,15,31,7FH,-1,
                -1,-1
        DB      -1,-1,-1,15,7,3,1,1,1,1,1,1,1,1,1,1
        DB      -1,-1,7FH,7FH,7FH,7FH,7FH,63,63,63,63,
                63,63,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,64,0,0,0,0,0
        DB      0,0,0,0,224,240,78H,78H,78H,78H,78H,
                78H,78H,78H,168,168
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
;
LSTEPB  DB      6,48            ;left step going back
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
```

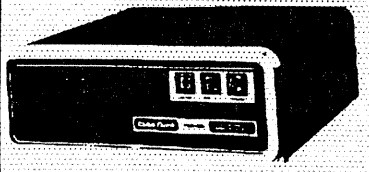
```
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-2,-8,240,-8,-2,
                -1,-1,-1
        DB      -1,-1,-1,240,224,192,80H,80H,80H,80H,80H,
                80H,80H,80H,80H,80H,80H
        DB      -1,-1,-2,-2,-2,-4,-4,-4,-4,-4,-4,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0
        DB      0,0,0,0,7,15,30,30,30,30,30,30,30,
                30,21,21
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,192,80H,0,0,0,0,0,0,0,0,
                80H,224
        DB      240,240,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,1,1,1,3,3,3,3,3,3,-1,-1,-1,-1,-1
        DB      0,0,0,0,0,31,63,63,7FH,-1,-1,-1,7FH,
                63,31,7
        DB      3,3,7,-1,-1,-1,7FH,7FH,7FH,7FH,63,63,
                63,63,63,63
        DB      63,62,78H,78H,78H,240,240,240,240,
                0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,3,1,0,0,0,0,0,0,0,0,1,7
        DB      15,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,80H,80H,80H,80H,80H,192,192,
                192,192,192,192,-1,-1,-1
        DB      0,0,0,0,0,-8,-4,-4,-2,-1,-1,-1,-2,-4,-8,224
        DB      192,192,224,-1,-1,-1,-2,-2,-2,-2,-4,-4,-4,
                -4,-4,-4
        DB      -4,7CH,30,30,30,30,30,15,15,15,15,
                0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,7FH,31,15,31,
                7FH,-1,-1,-1
        DB      -1,-1,-1,15,7,3,1,1,1,1,1,1,1,1,1,1
        DB      -1,-1,7FH,7FH,7FH,7FH,7FH,63,63,63,
                63,63,63,-1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,192,0,0,0,0,0
        DB      0,0,0,0,224,240,78H,78H,78H,78H,78H,
                78H,78H,78H,168,168
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
                -1,-1,-1
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
;
*LIST ON
        END
```


HARD DRIVES FOR SALE

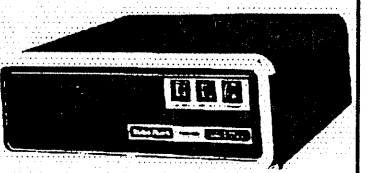
Genuine Radio Shack Drive Drive Boxes with Controller, Power Supply, and Cables.
Formatted for TRS 6.3, installation JCL included.
Hardware write protect operational.
Documentation and new copy of MISOSYS RSHARD5/6 included.
90 day warranty.

5 Meg $175 **10 Meg $225** **15 Meg $275** **35 Meg $445**

Shipping cost add to all prices



Roy T. Beck
2153 Cedarhurst Dr.
Los Angeles, CA 90027



(213) 664-5059

PROGRAMMING TIDBITS

Copyright 1992 by Chris Fara, Microdex

Basic Days of the Week

Browsing through Dr.Ecker's "TRS-80 Column" in "Computer Monthly", in the December 1990 issue I noticed a routine to calculate the day of the week for any date. A handy thing for many applications. So I was about to jot it down in my notebook, when Dr.Ecker's comment about the routine's math caught my attention: "a bit beyond me", he wrote. Well, as a math professor, at that time hiding under a pen-name "David Wade", he was perhaps exaggerating a bit for effect. But I had to admit that the math was at least beyond me. Unnecessarily convoluted, it did a great job of obscuring the simple logic of the calendar. A habitual itch to clarify and simplify kept me awake until I cooked up a more straightforward scheme.

```
10 INPUT "Year, Month, Day "; y,m,d
20 x = y + (m<3)
30 y = y-1 + x\4 - x\100 + x\400
40 m = (m-1+(m>2))*2 + (m-(m>7))\2
50 d = (y+m+d) MOD 7
```

The first idea here is that starting from any arbitrary date, each next year shifts the weekdays by one day (there are 52 weeks plus one day in a normal year). This is expressed in line 30 by Y-1. Each leap year shifts the weekdays by one more day, except in January and February. The intermediate variable X in line 20 accounts for this "except": a BASIC comparison such as (M<3) yields -1 if true, 0 if false. Thus in line 30 the three "integer divisions" say: add one day for each leap year (X\4) except in century years (X\100), but do count century years divisible by 400. The result Y is the sum of shifts caused by elapsed years.

If all months had 28 days then weekdays would be the same in each month. This not being the case, line 40 calculates shifts caused by months. The first expression adds 2 shifts for each month except February. The second part adds one extra day every other month, corrected for the back-to-back 31-day months of July and August.

In line 50 add up those shifts plus the day of the month, and calculate the remainder of division by 7 (modulo 7). The result is a number 0-6. To synchronize it with the "real" calendar look up any known date and add a constant to Y+M+D. As it turns out, the constant is zero for our current "Gregorian" calendar. Hence conveniently D=0 for Sunday, D=1 Monday, and so on.

In those BASICs that don't sport the backslash "integer division", use the FIX function. For example instead of X\4 write FIX(X/4). Similarly replace the MODulo operator with expressions such as...

```
50 d = y+m+d: d = d - FIX(d/7)*7
```

To spell out weekday names add something like this...

```
60 w$ = "SunMonTueWedThuFriSat"
70 PRINT MID$(w$, d*3+1, 3)
```

Prior to 1583 Europe and its colonies used the "Julian" calendar established in 46 BC by Julius Caesar for the Roman empire. All century years, not just those divisible by 400, used to be leap years. As a result the calendar was running too slow. To catch up, the Gregorian system deleted 10 days from people's life (October 5-14, 1582), which caused some wild street riots! But no riot is needed to modify our routine for the old Julian dates. Simply shorten line 30...

```
30 y = y-1 + x\4
```

and in line 50 add a constant 5...

```
50 d = (y+m+d+5) MOD 7
```

In the English-speaking world, including US territories, the mess was even worse. The Gregorian system was not adopted until 1752 when 11 days were "lost" (September 3-13). Not only that: before the switch New Year's day used to be March 25! Care to tweak our algorithm to calculate that "Old Style" birthday of your grand-grand-grand...?

Speaking about "julian" dates. Some BASICs, for example Mod-III, have a command to convert a date to a consecutive number of the day in a year. They call it "julian" date, but it has nothing to do with the old Roman calendar, nor is it quite like the Julian dating system known in astronomy. Astronomers count consecutive days starting from January 1, 4713 BC, a theoretical date when the solar and lunar cycles would have been in synch with the ancient Roman tax collection cycle (...and you thought IRS was our modern American invention). By now those day numbers are truly astronomical, close to 2,450,000. The term "julian" was coined in 1582 by the author of this dating system, the astronomer Scaligeri, in memory of his father Julius.

HINTS & TIPS

GETTING STARTED ON A BULLETIN BOARD SYSTEM

By Jim Marshall

Here are a few points to consider before you make your first telephone call to a BBS.

1. There are two kinds of services you can access with your modem. The first is a Bulletin Board System (which is usually free of charge) and the others are what I will call "User Pay Data-Bases" such as GEnie, Compuserve, etc. You are charged an hourly rate to use these systems. Long distance charges will apply if the BBS or data-base is outside your local calling area.

2. You must apply for access on all of these services before you can use them. You will be asked for your name, address, phone number, etc. for the sysop (System Operator) to verify that you are who you say you are. You will also be asked for a password. Some thought should be given to selecting a password, as it is a way to safeguard against someone signing on as you and possibly getting you into trouble with the Sysop. Also, once chosen, you should keep your password secret. This is the reason that your password will not be displayed on the screen as you type it during log-on. (In most cases).

There is, however, a lot which you can learn on this first visit. Most systems will let you read messages, but will not let you leave messages or up-load and down-load files. The exception, of course, will be your access application.

It would be to your advantage to print out the command menu(s) on the printer (or make notes if a printer is not available). Take the first visit slowly and READ the messages left for the benefit of novices (which we ALL are at first). Please follow the log-off procedures when you have finished. While it is unlikely you will cause the BBS to crash, it is inconsiderate and may get you dumped off the board before you get on it.

3. The idea behind the BBS is the exchange of information in the form of chit chat in the message section and programs or utilities in the file section. Some boards concentrate on one more than another, while most reach a balance somewhere in between.

There are a lot of people out there who just take all the time and rarely contribute. This tends to defeat the purpose for which the board was put up in the first place. As a novice user there is nothing wrong with down-loading files without up-loading something in return. But after a while you should have gained some experience and it will be time to start giving something back. Your contribution need not be a re-invention of the wheel, so to speak, but it should not be an outright copy of something you took from another board either!

4. The BBS is usually set up by a private individual on his own equipment at his own cost. This makes him the authority on how his board will be used. If he doesn't like what you put on his board in the way of offensive or vulgar messages or copyright software, he will not only delete it from the board but might cancel your access privilages. This brings me to an important point. Access to a BBS is a privilege, not a right as some people seem to think.

Remember that there is a human being operating the computer at the other end of the line. If you do make an error of some kind, admitting the goof goes a long way to keeping in the Sysop's good graces. Even if you don't goof, Sysops like to be recognized as much as anyone else. A note to let him/her know you appreciate having this resource is a step in the right direction.

To conclude, courtesy and consideration to the Sysop and other users of the BBS will add to everyone's enjoyment of the System.

INTEGRATED CIRCUIT TECHNOLOGY

by Karl Mohr

The worldwide sales of the electronics industry will be approximately a trillion dollars by the end of the century. Estimates place its current sales above $200 billion, which is roughly equal to the gross national product of India and is larger than the G.N.P. of every country in the world except the top dozen or so.

Perhaps the most familiar device fabricated from semiconducting electronic materials is the transistor. Ever since its invention in 1948 workers have sought to miniaturize this most fundamental of electronic components. The motivation is twofold. As transistors shrink in size, electronic systems can be made smaller and the sophistication of the equipment can be increased.

The rate at which transistors have been miniaturized is remarkable. The line width, or smallest feature size of a circuit, has decreased dramatically over the past quarter century. A typical line width in 1960 was 30 micrometers, or 30 millionths of a meter. Today line widths are commonly on the order on one micrometer, or about one-75th the width of a human hair. Most recently half-micrometer line widths have been achieved. They represent an important step in the development of semiconducting chips with more than 100,000 transistors used for carrying out high-speed logic operations and chips capable of storing up to 16 million bits of information.

Extrapolations based on the historical trend suggest that line widths of .1 micrometer will be reached, at least at the research and development level, in the mid-1990's. Many theoreticians believe such a line width represents a serious limit in the design of electronic circuits; devices smaller than .1 micrometer may require modes of operation different from the present ones. As the end of the century nears, therefore, the development of transistors is broadening into areas other than miniaturization. Workers are exploiting exotic effects to increase the speed of the present generation of transistors. They are experimenting with novel combinations of materials and new transistor designs. They are even beginning to dream of "biochips."

Two kinds of transistors now common in the electronics industry are the bipolar transistor and the field-effect transistor, or FET. Both are central to the operation of computer. Bipolar devices are the basic building blocks of a computer's central processing unit, the part of the machine that does operations on data. FET's, on the other hand, are generally the basic building blocks of a computer's memory. FET's are also increasingly used to carry out logic operations in small- to medium size computers. The division is a consequence of the fact that in switching operations bipolar transistors are faster than FET's.

The speed of the present generation of integrated circuits can, as I have mentioned, be increased by miniaturizing their components. To etch progressively smaller structures on semiconducting chips, special approaches will have to be followed. In particular, the relatively long wavelength of light is poorly suited to cutting extremely sharp features into the photoresist that cover the wafers. One way this limitation can be skirted is by substituting beams of high-energy electrons. The wavelength associated with the electrons is a fraction of the diameter of an atom. It is this approach that made possible the fabrication of the half-micrometer line-width structures I mentioned above.

In another approach under consideration, very short wavelength X rays would be used to cut circuit elements into chips. The X rays would be produced by synchrotrons, machines that accelerate charged particles to high velocities by means of electric and magnetic fields. As the particles are accelerated they throw off large amounts of electromagnetic radiation. The wave-length of the radiation can be "tuned" by changing the acceleration of the particles. For X ray "lithography" to be effective, mask-alignment techniques would need to be developed. In addition, present day synchrotrons are quite large; tabletop models would be disirable.

Plasma etching is another evolving technology expected to play an increasingly important role in miniaturization. In this technique a plasma beam, a "soup" of charged particles containing approximately equal numbers of positive ions and electrons, is aimed at the surface of a chip. By varying the direction of the beam it is possible to remove atoms at particular locations on the chip. In addition to have a high degree of directionality, plasma etching has the advantage of not involving liquids, which can cause corrosion or can become entrapped in cavities. It has a disadvantage too: the energetic beam can damage the surface of the chip. Either plasmas with lower energies or techniques for eliminating surface damage will have to be developed.

Integrated circuits might also be made faster by operating them at low temperatures. As temperatures decreases, charge carriers are less likely to scatter from phonons (vibrations of atoms in a crystal), and their mobility is thereby increased. Moreover, harmful, thermally activated processes such as electromigration slow down. In addition the level of back ground noise, the unwanted electrical signals associated with resistors, decreases. Reduction in noise means that transistors can be run with less power and can be built closer together.

Biochips represent another futuristic technology that has gained much popular coverage. There is no precise definition of the term biochip. It can refer to the notion of fabricating a chip from organic (carbon-containing) molecules or from biological or biological-like molecules. It can also refer to the idea that a chip functions somewhat like a natural organism. To date none of the definitions has been physically realized. One can speculate, however, that someday it may be possible to arrange the base pairs in DNA, the genetic material, to store information as complex as a Mahler symphony. Perhaps the scanning tunneling microscope, which has a tip with a single atom on its end, could be employed to store and access such information. Perhaps gene-splicing techniques will be exploited to manufacture the pre-coded DNA!

MODEL I/III PEEKS & POKES

by Bruce McDowell

16420	character set. 0 = regular. 1 = alternate
16521	peeks number of lines printed +1
16526	model 3 only - number of characters printed per line plus 1
16527	max characters per lprint line
16913	model 3 only - cassette baud rate 0 = 500, non-zero = 1500
16916	model 3 only - video display scroll protect 0 = no scroll protect 1-7 = scroll protect from top

---

Model 4
Public Domain Disks

M4GOODIES#1: day/cmd, day/txt, gomuku/cmd, llife/cmd, llife/doc, writer/cmd, writer/doc, writer/hlp, yahtzee/bas

M4GOODIES#2: arc4/cmd, arc4/doc, cia/bas, etimer/cmd, index/cmd, index/dat, mail/bas, mail/txt, trscat/cmd, trscat/txt, util4/cmd, xt4/cmd, xt4/dat, xt4hlp/dat

M4GOODIES#3: convbase/bas, dates/bas, dctdsp/cmd, dmu/cmd, dmu/doc, dskcat5/cmd, dskcat5/doc, editor/cmd, editor/doc, fedit/cmd, fkey/asm, fkey/cmd, fkey/doc, hangman/cmd, m/cmd, m/src, membrane/bas, miniop2/cmd, miniop2/src, move/cmd, move/doc, othello4/bas, scroll4/cmd, scroll4/src, setdate6/cmd, setdate6/doc, setdate6/fix, spaceadv/bas, taxman/bas, utilbill/bas, utilbill/doc

M4GOODIES#4: WORD WIZARD disk #1 of 3
anima/bas, archi/bas, autos/bas, battuere/bas, captus/bas, contvert/bas, curro/bas, dico/bas, ducere/bas, eulogos/bas, facere/bas, fluere/bas, gradi/bas, jacere/bas, kata/bas, male/bas, metron/bas, naus/bas, startup/bas, startup/jcl, stig/bas, tangere/bas, wordmenu/bas

M4GOODIES#5: WORD WIZARD disk #2 of 3
cognos/bas, frangere/bas, juris/bas, medius/bas, mittere/bas, monos/bas, numbers/bas, orare/bas, pandemos/bas, para/bas, pathos/bas, pendere/bas, philanth/bas, phongrap/bas, polynom/bas, prefix1/bas, prefix2/bas, premere/bas, sal/bas, startup/bas, startup/jcl, statuere/bas, wordmenu/bas

M4GOODIES#6: WORD WIZARD disk 3 of 3
bible/bas, french1/bas, french2/bas, french3/bas, italian/bas, latphras/bas, lit1/bas, lit2/bas, myths/bas, places/bas, plicare/bas, spanish/bas, stagnare/bas, stare/bas, startup/bas, startup/jcl, synpath/bas, televid/bas, tenere/bas, vaco/bas, valere/bas, vox/bas, wordmenu/bas

M4GOODIES#7: calendar/cmd, castladv/bas, civilwar/bas, crimeadv/bas, dctdsp/cmd, ed6/cmd, ed6/doc, edittext/bas, fedit/cmd, mail/bas, mail/txt, scramble/bas, states/bas, textpro/cmd, time4/bas, wizard/bas, wizard/doc, worldcap/bas

M4GOODIES#8: books/bas, books/doc, dmu/cmd, dmu/doc, hamcalc/bas, hamhelp/bas, network/bas, network/doc, pirate/bas, pirate/doc, vmap/bas, vmap/doc, vmap2/bas, vmap2/doc, zork1/doc, zork2/doc, zork3/doc

M4GOODIES#9: ft/cmd, ft/doc, pterm/cmd, pterm/doc, r/cmd, r/doc, scrconv/bas, scrconv/doc, video4/asm, video4/cmd

M4GOODIES#10: checker/cmd, crossref/cmd, crossref/doc, ddir/cmd, diskcat/cmd, diskcat/doc, division/bas, division/doc, getput/bas, getput/doc, host/cmd, hv/bas, maszap4/cmd, maszap4/doc, park/cmd, profile4/doc, protect/bas, protect/doc, rename/bas, replace/bas, restore/bas, rm/bas, scrndump/bas, scrndump/doc, super/hlp, vers/cmd

M4GOODIES#11: benchmrk/bas, bigcal/bas, bigcal/doc, birthday/bas, dearc4/cmd, dezip2/cmd, dname/cmd, docufile/bas, docufile/doc, docufile/mrg, escape/bas, mem4/cmd, million/bas, nomad/bas, password/bas, password/dat, password/doc, password/jcl, roman/bas, sixtymin/bas, startrek/bas, trekinst/bas

M4GOODIES#12: awari/bas, buyimg/bas, crasher/bas, curvfit2/bas, gradebk/bas, mortcost/bas, mortcost/doc, print/bas, print/doc, reiman/bas, square/bas, starlane/bas, staybus/bas, sunrise/bas, synonym/bas, timezon1/bas, timezon2/bas, travel/bas, vmap2/bas, vmap2/doc, weekday/bas

M4GOODIES#13: calndr1/bas, calndr2/bas, calndr3/bas, formltrs/bas, invloan/bas, limerick/bas, martian/bas, mission/bas, moneymkt/bas, munchmth/bas, numbrfun/bas, smith/bas, smith/doc, star2000/bas, starfind/bas, starfind/dat, starfind/doc, starfind/jcl, states/bas, wallst/bas

M4GOODIES#14: alphahex/bas, bowlchng/bas, bowlcrea/bas, bowldetl/bas, bowlfinl/bas, bowling/doc, bowlmenu/bas, bowlprnt/bas, bowlrcap/bas, bowlrecd/bas, bowlrecp/bas, bowlschd/bas, bowlscor/bas, bowlsort/bas, buscheck/bas, calculat/bas, chekform/bas, deprec/bas, futrdate/bas, membrain/bas, minimath/bas, normalz/bas, numconv/bas, pcbdest/bas, pcbdest/doc, pcform/bas, pcpm/bas, pcpm/doc, pcpm/jcl, utscan/bas, yagibeam/bas, zeller/bas

M4GOODIES#15: laughs/bas, laughs/dat, laughs/doc, laughs1/dat, laughs2/dat, laughs3/dat, laughs4/dat, laughs5/dat, laughs6/dat, laughs7/dat, laughs8/dat, laughs9/dat, laughs10/dat, laughs11/dat, laughs12/dat, laughs13/dat, laughs14/dat, laughs15/dat

M4GOODIES#16: trivia/bas, trivia/doc, trivia1/dat, trivia2/dat, trivia3/dat, trivia4/dat

M4GOODIES#17: acrs/bas, amorloan/bas, clokmod/bas, compound/bas, dcform/bas, decide/bas, easyword/bas, editno/bas, epslabel/bas, esckey/bas, expect/bas, funct1/bas, funct2/bas, gasform/bas, hexprint/bas, hexsay/bas, lostgold/bas, mathfunc/bas, mpgcalc/bas, neclabel/bas, nicelist/bas, nonlin/bas, nonlin/rem, payback/bas, peekprnt/bas, percent/bas, prntcall/bas, proverbs/bas, randseed/bas, savings/bas, speech/bas, tasklist/bas, tempconv/bas, weightfm/bas

Each disk is $5.00 (U.S.)
or get any 3 disks for $12.00 (U.S.)
please specify the exact disks wanted.

TRSTimes PD-DISKS
5721 Topanga Canyon Blvd., Suite 4
Woodland Hills, CA. 91367

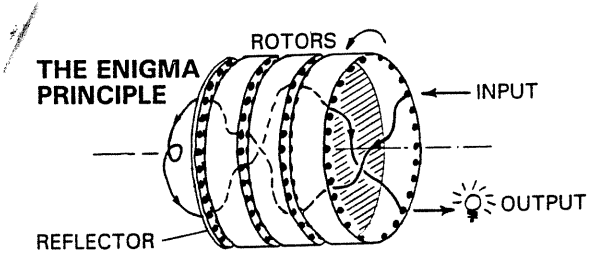
PROGRAMMING TIDBITS

Copyright 1992 by Chris Fara, Microdex

Basic Enigma Machine

One of the most intriguing figures in the history of computers was the British mathematician Alan Turing (1912-1954). At the age of 24 he published a paper on some famous issue of algebra. But that issue seems less relevant today than the way Turing went about solving it. To prove his point, he described what essentially was a digital computer. The features of his "universal computing machine" (since then dubbed the "Turing machine", often mis-spelled "Touring") are taken for granted today, but were quite novel at that time: binary computing using only zeros and ones, conditional branching of program flow, sharing of computer memory by programs and data. He imagined "memory" as an impossibly long paper "ticker" tape. The implementation of his machine had to wait until electronics made it practical. All our computers today are "Turing machines".

Turing was (and still is) relatively little known, except in the rarefied circles of higher science. Even less known is his crucial role during World War II, in breaking the system of the "Enigma" machine used by Germans to encode military messages. That machine was a sort of 26-letter electric typewriter. When a key was pressed, current went to one of 26 input pins located on one rim of a drum-like wheel (rotor). Inside the rotor, the input pins were wired at random to 26 output pins on the opposite rim. These output pins passed the current to the input pins of the next wheel, and so forth. The original Enigma had 3 rotors, later expanded to 4 and even 5. The output from the last (inner) rotor went to a pin on a stationary plate called "reflector" where it was wired to another pin on that plate and thus was "reflected" as input to the backside of the inner rotor. From there it went back to the first (outer) rotor, and finally to one of 26 light bulbs that indicated the encoded letter.



After one letter was encoded, the outer rotor would make 1/26-th of the full turn, automatically changing the relation of pins. After the outer rotor made one full turn, the next rotor would advance by one pin, and so on, like an odometer. Thus each letter would be encoded by a different wiring path, and this made the Enigma so hard to crack. All breaking of encrypted messages is based on analyzing typical patterns of letters. It didn't work with the Enigma. Even with just 3 rotors, identical pattern could be only guaranteed after exactly 26*26*26 = 17,576 characters! The encoding also depended on the initial setting of the rotors in respect to each other at the beginning of a message. There were some other minor complexities. It all added up to millions of combinations.

Enigma was considered unbreakable, and Germans trusted it 100%. Yet the code was broken already before the war by a couple of Polish mathematicians who later passed the secret to the Allies. But when Germans added the 4th rotor, the original manual decrypting procedure proved too time consuming. An automated solution was the only hope. This fell right in line with Turing's vision of computing machines, and he eventually led a team that developed the "Colossus", arguably the first-ever modern computer, even though its mission was limited to the deciphering of the Enigma code.

So much for history. The Enigma machine was a bulky clanker stuffed with a jungle of wiring. The operator had to watch the light bulbs and jot down the letters one by one. Today, the Enigma can be easily implemented on any desktop computer (our "universal Turing machine") with a simple BASIC program. To see how it works, let's assume a simplified 6-letter alphabet symbolized by numbers 0-5. Let's also assume for now that our Enigma machine has only 2 rotors. The wirings can be written in a table:

	0	1	2	2'
0	4	1	5	0
1	2	3	0	1
2	1	4	1	4
3	5	5	4	2
4	0	2	2	3
5	3	0	3	5

Rows represent the 6 pins. Column 0 represents the wiring of the "reflector". For example row 3 has value 5 which means that the wire from pin 3 goes to pin 5. The wiring is "self-inversing": row 5 has value 3. Column 1 represents the random wiring of the "inner" rotor, column

2 is the "outer" wheel. Column 2-prime will be discussed in a moment. Now suppose we want to encode the digit 1. Starting from the outer rotor (column 2) in row 1 we find 0. Therefore output from the outer rotor goes to pin 0 (row 0) on the inner rotor, where we find 1. In the "reflector" row 1 routes it to pin 2. Now going back, we look for the "output pin" 2 in column 1. This is found in row 4. In the outer column we look for 4 and find it in row 3. Hence 1 has been encoded into 3.

After the first character has been transmitted, the outer rotor turns by one pin. Its new, shifted "wiring" position is shown in column 2-prime. If we now try to encode 1, then the result will be 4, not 3 (the sequence is 1-1-3-5-3-4). In the same manner we can trace the reverse encoding of 4 into 1 (4-3-5-3-1-1). This symmetry is the result of the self-inversing "reflector". Thus decoding of a secret message is the reverse of encoding, with the rotors set in the same starting position in both cases.

This entire process can be written in just a few lines of BASIC. Let's call our "wiring" table the array R(2,5). In addition, we'll need a small array S(2) to keep track of the setting of each wheel. The number to be encoded is passed to the routine in the variable N.

```
10 for x=2 to 1 step -1
11 N = R(x, ( N + S(x) ) mod 6)
12 next

13 N = R(0, N)

14 for x=1 to 2: for y=0 to 5
15 if N=R(x,(y+S(x)) mod 6) then N=y:y=5
16 next y, x

17 for x=2 to 1 step -1
18 S(x) = (S(x)+1) mod 6: if S(x) then x=1
19 next
```

The first 3 lines encode the original value of N, starting from the outer rotor. Notice how the encoding depends on the setting S(x) for each rotor. Line 13 is the "reflector". Lines 14-16 pass the code back to the outer rotor, and yield the final output N. In the last 3 lines the "odometer" advances the rotor settings. That's it.

Actually, the "physical" Enigma behaved slightly differently from our computerized model. A rotation of any wheel not only shifted the pins in respect to the input, but obviously also shifted the output pins in respect to the next rotor. But that was an accidental property of the mechanical implementation of the system, irrelevant to the coding process, because the wiring of each rotor is random anyway. So we can skip that detail.

To make a complete working Enigma, DIMension the arrays for the desired number of rotors (columns) and characters (rows), fill in all cells with proper "wiring" numbers, adjust loop counters and "modulo" values, and provide for input and output. Or if you don't want to bother with programming, send me 5 bucks for a "Basic Enigma on the disk" (c/o Microdex, 1212 N.Sawtelle, Tucson AZ 85716; specify Mod-III TRSDOS or LDOS, Mod-4, or MSDOS). It's a neat game for kids and for war history buffs. Great for secret love letters, too.

After the war, Turing went on to build a general-purpose computer, and then turned to the mathematical principles of genetics and evolution. But the memory of Alan Turing often comes to mind today for reasons other than computing. He was "gay" and, unlike his contemporaries, insisted on being open about it when homosexuality was still illegal in England. It got him into trouble with the courts, which in turn may have been a cause of his suicide when he was just reaching the peak of his career. How much more might his genius have achieved, had he lived in our more tolerant age? How much more can ANY person achieve in an environment that accepts harmless deviations from arbitrary rules?

Author's note: This essay was written on June 23, 1992, on the 80-th anniversary of Turing's birthday, as a small "thank you" for our wonderful "Turing machines".

TIRED OF SLOPPY DISK LABELS?
TIRED OF SEARCHING DISKS?

YOU NEED 'DL'

'DL' will automatically read your TRSDOS6/LDOS compatible disk and then print a neat label, listing the visible files (maximum 16).
You may use the 'change' feature to select (or reject) the filenames to print.
You may even change the diskname and diskdate.
'DL' is written in 100% Z-80 machine code for efficiency and speed.

Don't be without it - order your copy today.

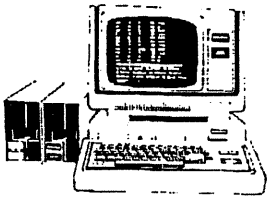
'DL' is available for TRS-80 Model 4/4P/4D using TRSDOS 6.2/LS-DOS 6.3.x. with either an Epson compatible or DMP series printer

'DL' for Model 4 only $9.95

TRSTimes magazine - Dept. 'DL'
5721 Topanga Canyon Blvd. #4
Woodland Hills, CA 91367

DOCUMENTING THE DOCUMENTS

by Roy T. Beck



Recently I noted there were several different versions of the TRS-DOS/LS-DOS manuals. The manuals I am referring to are from Radio Shack and are mostly the familiar brown binders, 3 rings, and with various titles imprinted on the spine. Looking around my book shelves, I saw "Model 4/4P TRSDOS Version 6 Disk Operating System and BASIC Interpreter" on one such manual, and "Model 4 Disk System Owner's Manual" on another. Are these the same manual with different binders or are they different? If the latter, how do they differ?

In addition to the DOS manuals, there are two other types of manuals for most Radio Shack computer equipment. The first is the Technical Reference Manual, useful to advanced programmers ("power users"?). The second is the Service Manual, useful to hackers and repairmen.

I have accumulated many of the available manuals and DOSes for the models I, III and 4, and so much, but not all of the following is from such documentation. Other information is from other knowledgeable persons, including Art McAninch of Borger, Texas, Lance Wolstrup, publisher of TRSTimes, and Roy Soltoff of MISOSYS.

Before I get into the details of the different versions of the manuals, it would be well to summarize the DOSes which were available for each of the machines. In order to make the presentation as logical as possible, I will attempt to present the same information for the Models I, III and 4/4P in that order.

In the case of the Model I, I am deliberately ignoring most of the numerous aftermarket DOSes, partly because I intend to restrict this to the "official" DOSes, and partly because of my own ignorance. I admit to a certain degree of inconsistency, as of course Radio Shack never recognized the 1.4 and 1.5 versions of the Model III DOS. I included the LDOS family for both Model I and III because these were supplied by Radio Shack when you purchased a hard drive from Radio Shack for either of these two machines.

The reader might also wish to read a previous article "My Recollections of the Model I" which was a history of the Model I. That article included more extensive information on the aftermarket DOSes available for the Model I. See the Jan/Feb and Mar/Apr 1992 issues of TRSTimes.

TRS-80 MODEL I DOSes

Table 1

	DOS Version	Catalog Number
Radio Shack DOSes	1.0	26-
	2.0	26-
	2.1	26-0313
	2.2	26-0313
	2.3	26-0316
	2.3B	26-0316
	2.7DD	26-1143
	2.8DD	26-1143
Logical Systems, Inc (LSI) DOSes	5.0.0	
	5.1.0	
	5.1.1	
	5.1.2	
	5.1.3	
	5.1.4	
MISOSYS Inc. DOSes	5.3.0	
	5.3.1	

Notes on the various DOS versions

V 1.0

This version of the DOS existed, and was terrible. I recently read some old comments by Irv Schmidt, former publisher of 80-US and BASIC Computing. He was speaking of this earliest DOS, and offered the following list of things which didn't work! COPY, ROUTE, CHAIN, APPEND. There were probably more, but this gives you an idea of how bad that first DOS was.

V 2.0

This was a major rework, was still troubled, but could be made to work, marginally.

V 2.1

This version was widely distributed and used, but still contained significant errors; disk file handling was flawed. Files sometimes disappeared into a black hole. This version was used for a couple of years, although it was known to be less than perfect. Being the only DOS available, it provided the impetus for Apparat in Denver, Colorado to come out with a patched version of the DOS. Their version of Radio Shack's DOS was known as NEWDOS 2.1.

V 2.2

An attempt by Radio Shack to clean up all the known errors in V 2.1. Unfortunately, it contained a new and fatal error; it would kill all open files when told to close a file.

V 2.3

This version followed V 2.2 within less than two weeks. It was apparently bug-free, and thereafter remained the standard Radio Shack DOS for the Model I.

V 2.3.B

This version of TRSDOS changed the way the End of File and Number of Records was handled in the directory. The original Radio Shack (Randy Cook) method worked, but was confusing. After long usage (up through Version 2.3), Radio Shack decided to revise their method to a logically correct method. V 2.3.B was the result, and OLD disks had to be corrected to NEW conditions with the use of a utility named UPGRADE which came on the 2.3.B DOS. After upgrading, the NEW program disks would usually show one less # REC than before UPGRADEing, but only if the EOF was not 00. Of greater concern, OLD disks (V 2.1, 2.2, and 2.3) would not run properly under 2.3.B, and NEW disks (created or UPGRADEd under 2.3.B) would not run properly under 2.1, 2.2, or 2.3.

Apparently this change came about when TRSDOS 1.3 was issued for the Model III, and which contained the same changes.

To insure you did not UPGRADE a 2.3 system disk and then attempt to run the NEW programs under the DOS 2.3 on the disk, the UPGRADE utility killed off any files from the following list that it found on the 2.3 disk:

SYS0/SYS	SYS1/SYS	SYS2/SYS
SYS3/SYS	SYS4/SYS	SYS5/SYS
SYS6/SYS	FORMAT/CMD	BACKUP/CMD
BASICR/CMD	BASIC/CMD	

Not only did UPGRADE kill all the system files, it also killed off BASIC. Note, it did not replace the 2.3 system files with the corresponding 2.3.B files, it simply turned the OLD system disk into a NEW data disk. Even more strange, V 2.3.B did not include an upgraded BASIC. Thus, if you UPGRADEd a disk of BASIC programs to V 2.3.B, you could no longer run them, as 2.3.B didn't have a BASIC! Strange are the ways of Ft. Worth, at times.

V 2.7DD

This DOS required a double density adapter, and most likely would only function with Radio Shack's proprietary unit, and not with the various aftermarket doublers. Because of the model I ROM, track 0 still had to be single density, but the remainder of the disk was double density. It was a cassette DOS only, and was restricted to 500 baud.

Apparently this DOS was an attempt to appease Model I owners by offering them many of the features of the new Model III. Radio Shack said this version was defined to be similar to V 1.3 on the Model III, but in their introduction they went on to say no software was available to run under this DOS. Since V 2.3 programs would not run under it, the DOS was really an orphan, with value only to dedicated Model I users who refused to update to the III. (I remember this era well; I was one of the holdouts). Apparently you were very much on your own with this DOS.

Many DOS and ROM routines were identified in the accompanying manual for use by programmers. Radio Shack noted some additional commands were added beyond those of V 2.3. Included in its BASIC were:

- CMD "C" -- compress BASIC files, and delete comments and spaces.
- CMD "L" -- load a DOS routine.
- CMD "O" -- sort a string into ASCII sequence.
- NAME -- renumber a BASIC program.
- MASTER -- allows the user to require DOS to begin searching for files on a specific drive instead of the default :0.
- USER -- allows the user to define his own DOS command.
- WP -- a software write protect.

Some other added (DOS) commands included TRACE, UNKILL, VERIFY, SETCOM, SPOOL, and ERASE. (An ERASED file could not be unkilled). It also included most of the LIBRARY and BASIC commands of V 1.3.

Model I SD programs could be COPYed to V 2.7, and vice versa, but this DOS could not read a Model III disk. It was not interchangeable with the V 2.3 family. A granule was 3 sectors.

The rear cover of the user's manual was marked 8749342. A quick reference card with it carried a rear cover number of 8759156, Copyright 1982. Apparently there was no separate Technical Reference Manual.

V 2.8DD

This later version existed, was an update of 2.7DD, and probably was issued to fix errors in the 2.7DD version.

LDOS 5.0.0

This DOS came about partly as the result of the efforts of a company in Southern California, LOBO International. Their product was the LOBO expansion interface, intended to replace the notoriously unreliable Expansion Interface initially produced by Radio Shack. Because LOBO included some new features which TRSDOS V 2.3 could not handle, LOBO looked around for someone to create a new DOS. Logical Systems, Inc. (LSI) of Milwaukee was the choice.

The members of LSI were Bill Schroeder, Roy Soltoff, Chuck Jensen, Tim Mann, and Dick Konop. LDOS for the LOBO and the Model I (a separate version for each) was the resulting new DOS.

LDOS V 5.1.0
An update of the previous version.

LDOS V 5.1.1
An update of the previous version.

LDOS V 5.1.2
An update of the previous version.

LDOS V 5.1.3
This version was sold over a long time span. When Radio Shack announced the 5 Meg hard drive for the Model III, they also decided to offer a retrofit package to allow operation of the 5 Meg hard drive on a Model I. LDOS V 5.1.3 plus a Radio Shack driver thereby became the official hard drive DOS for the Model I.

LDOS V 5.1.4
This was the last update by the LSI team. Roy Soltoff (MISOSYS) now owns all rights to LDOS and continues to sell it to both Model I and III users.

LDOS V 5.3.0
This revision was brought out by Soltoff (MISOSYS) to update LDOS to agree, to the maximum extent possible, with LS-DOS 6.3.0. This version extended the dating range to 12-31-99. Note the deliberate gap in the Version numbering system. This was done to align the LDOS version with the LS-DOS (TRSDOS) Model 4 version.

LDOS V 5.3.1
This is the current version offered by MISOSYS to support the Model I. This version extends dates to 12-31-2011.

TRS-80 MODEL I OFFICIAL RADIO SHACK MANUALS

Level I BASIC Manual
Included with the Model I was David Lien's famous manual on BASIC. This manual was well received and applies to Microsoft Basic on many machines, not just Radio Shack.

This manual applied to the Level I version of the cassette machine with Palo Alto Tiny BASIC in a 4 K ROM.

Level II BASIC Manual
When you updated to Level II BASIC in ROM, you received Microsoft's BASIC in a 12 K ROM, with much more capability. This manual provided the knowledge to use it.

DOS Manuals
Radio Shack has issued at least two soft cover versions of the DOS manual for the Model I, and apparently also two three-ring binder versions of the manuals. I have what appears to be a V 2.1 manual, updated to V 2.3.

Other Manuals
For the Model I, Radio Shack issued a manual entitled "TRS-80 Micro Computer Technical Reference Handbook" It is Cat No 26-2103 with no numbers on the rear cover. In my copy, the copyright page shows FIRST EDITION FIRST PRINTING - 1978 Copyright 1978.

This manual contained Theory of Operations, Schematics, Adjustments and Troubleshooting and Parts Listing sections. While it was not terribly profound, it was a good beginning, and on a par with the machine itself. One of its weaknesses, for example, is that the schematic section did not include the external power supply which was very much a part of the machine. It also did not include the lower case keyboard conversion nor the double density floppy disk controller, both Radio Shack (afterthought) products.

"Radio Shack USER'S MANUAL FOR LEVEL 1 TRS-80 MICRO COMPUTER SYSTEM", Cat No 26-2101 was issued for use with the LEVEL I Basic Cassette machine. Level I was Palo Alto Tiny Basic with most commands abbreviated to one letter, this was the best feature of Level I. This manual was written by David Lien of San Diego, and is highly regarded as a good teaching tool for BASIC, although limited to LEVEL I.

There is a similar manual for the Level II cassette version of the Model I, which contained Microsoft BASIC in ROM. This was entitled "Getting Started With TRS-80 BASIC for use with Models I & III", Cat No 26-2107. Rear cover marked 8749165-10-82-SL. This manual is similar in style to David Lien's Level I manual, but was written by George Stewart.

"Getting Started With TRS-80 BASIC for use with Models I, III & IV", Cat No 26-21??. Rear cover is marked 8749414 283 S-L. This is the same manual as Cat No 26-2107 for the Models I & III, but with the title changed to include the Model 4.

Rear cover marked 8749165-10-82-SL "Getting Started With TRS-80 BASIC for use with Models I, III & IV", Cat No 26-21??. This is the same manual as Cat No 26-2107, but with the title changed to include the Model 4.

"Radio Shack TRS-80 Model Micro Computer Double Density Disk System Owner's Manual". This manual is Cat No 26-1143, rear cover marked 8749342. It is in the brown three ring binder, and contained a pocket card identified "TRS-80 MODEL I DOUBLE-DENSITY MICROCOMPUTER SYSTEM", copyright 1982, last page marked 8759156.

"TRS-80 micro computer technical reference handbook", Cat No 26-2103, copyright 1978 is a black, soft

cover manual containing most of the model I schematics (The external power supplies are not included). The Lower Case and Double-Density adapters were still in the future at the time of printing of this manual, so of course they also were not covered.

"Radio Shack Service Manual EXPANSION INTERFACE Catalog Number 26-1140" copyright 1979, is the manual for the early, unreliable designed Expansion Interface. Since the unit could be purchased with 0, 16 or 32 K of RAM, it carried the corresponding Cat Nos of 26-1140/1141/1142. The serial numbers of the 26-1140 supposedly ran from 1 to 34,999. The serial numbers of the 26-1141 and 26-1142 units supposedly ran from 1 to 7,999. Do not depend upon these serial numbers to determine if you have an early or late model Expansion Interface. The reason is that Radio Shack upgraded many of these early boards by simply substituting a later board in the early case; the serial numbers are attached to the case, not the board, and were not changed in such board swaps.

"Expansion Interface Catalog Number 26-1140/1141/1142 HARDWARE", copy right 1979, is a black, soft cover manual which pertains to the late, reliable Expansion interface. Again, the catalog number shifts with the amount of RAM installed at the factory. 26-1140 had 0 K, 26-1141 had 16K, and 26-1142 had 32 K. Again the Cat No on the bottom is not conclusive, as it was a simple matter to add more memory, and most units were expanded to 32 K, making them 26-1140's, regardless of the Cat. No. on the bottom. The serial numbers of these later units began with 035000 for the 26-1140, and 008000 for both the 26-1141 and 26-1142.

This manual appears to be a combination user's manual, technical reference manual, and service manual. There are at least two different versions of this manual. The earliest version has a serious error on page 37. The photo on that page is supposed to be of the late model, and is intended to locate various adjustments. Unfortunately, Radio Shack used a photo of the OLD board here! Just to further confuse things, they corrected this photo in a later version, but provided no easy way of determining which manual is which. My early one has no marks on the rear cover. The later one, (of which I have some xerox pages), has the correct photo on page 37, and has 778-2980094 marked on the rear cover.

In both units, the RAM chips are arranged in two parallel rows. In the early unit, the AXES of the ROWS runs across the short dimension of the board, from front to rear. In the later unit, the AXES of the ROWS runs parallel to the long axis of the unit, from right to left. This information is definitive, and you can identify the unit by peering into the ventilation cracks on the bottom. Be sure you are looking for the AXES of the ROWS, not the axes of the individual chips, which of course run the other way.

To be continued in the next issue..

YES, OF COURSE !

WE VERY MUCH DO TRS-80 !

MICRODEX CORPORATION

SOFTWARE

CLAN-4 Mod-4 Genealogy archive & charting ***$69.95***
Quick and easy editing of family data. Print elegant graphic ancestor and descendant charts on dot-matrix and laser printers. ***True Mod-4 mode***, fast 100% machine language. Includes 36-page manual. ***NEW!***

XCLAN3 converts Mod-3 Clan files for Clan-4 ***$29.95***

DIRECT from CHRIS Mod-4 menu system ***$29.95***
Replaces DOS-Ready prompt. Design your own menus with an easy full-screen editor. Assign any command to any single keystroke. Up to 36 menus can instantly call each other. Auto-boot, screen blanking, more.

xT.CAD Mod-4 Computer Drafting ***$95.00***
The famous general purpose precision scaled drafting program! Surprisingly simple, yet it features CAD functions expected from expensive packages. Supports Radio Shack or MicroLabs hi-res board. Output to pen plotters. ***Includes a new driver for laser printers!***

xT.CAD BILL of Materials for xT.CAD ***$45.00***
Prints alphabetized listing of parts from xT.CAD drawings. Optional quantity, cost and total calculations.

CASH Bookkeeping system for Mod-4 ***$45.00***
Easy to use, ideal for small business, professional or personal use. Journal entries are automatically distributed to user's accounts in a self-balancing ledger.

FREE User Support Included With All Programs !

MICRODEX BOOKSHELF

MOD-4 by CHRIS for TRS/LS-DOS 6.3 ***$24.95***
MOD-III by CHRIS for LDOS 5.3 ***$24.95***
MOD-III by CHRIS for TRSDOS 1.3 ***$24.95***
Beautifully designed owner's manuals completely replace obsolete Tandy and LDOS documentation. Better organized, with more examples, written in plain English, these books are a ***must for every TRS-80 user.***

JCL by CHRIS Job Control Language ***$7.95***
Surprise, surprise! We've got rid of the jargon and JCL turns out to be simple, easy, useful and fun. Complete tutorial with examples and command reference section.

Z80 Tutor I Fresh look at assembly language ***$9.95***
Z80 Tutor II Programming tools, methods ***$9.95***
Z80 Tutor III File handling, BCD math, etc. ***$9.95***
Z80 Tutor X All Z80 instructions, flags ***$12.95***
Common-sense assembly tutorial & reference for novice and expert alike. Over 80 routines. No kidding!

Add S & H. Call or write MICRODEX for details
1212 N. Sawtelle Tucson AZ 85716 602/326-3502

SO, WHAT'S NEW?

The LDOS & LS-DOS BASIC Reference Manual

a review by Lance Wolstrup

It is rather appropriate to follow Roy Beck's walk-through of the various TRS-80 system manuals with a writeup of the latest entry to this series. Roy Soltoff, of Misosys, has recently published a companion volume to the 'LDOS & LS-DOS Reference Manual' which was discussed in our last issue. This companion volume is called the 'LDOS & LS-DOS BASIC Reference Manual' and is, as the title suggests, a complete reference to Model I, III and 4 LDOS/LS-DOS Basic.

If you do any programming in Basic, this book is invaluable. It covers every aspect of the language, every command, every function, every statement and, while it is not touted as a tutorial, it contains so many examples that anyone with a passing knowledge of any Basic should be writing programs in no time at all.

Along with Model I, III & 4 LDOS/LS-DOS Basic, the manual also covers EhnComp, the compiler Basic available from Misosys. This is a sly move by Mr. Soltoff - when you read what EhnComp can do - you will be sorely tempted to order it immediately.

But back to normal interpreter Basic; one would assume that an old hand like me would have no need for a new Basic manual. -- Wrong!!! Oh sure, I have all the LDOS and LS-DOS manuals and I have a reasonably good grasp on the Basic programming language; but I don't know it all. A good example is that while thumbing through the manual, I began reading about the OPEN statement. Now, I have written hundreds of programs using this statement to access sequential and random files, but I never knew that LDOS Basic had the capability of returning an error when opening a file for output (write) that does not exist.

Apparently there are several ways to open a file for output.

OPEN"O",1,"FILENAME"	if the file exists it will be opened. If not, the file will be ceated.
OPEN"OO",1,"FILENAME"	the file must exist.
OPEN"ON",1,"FILENAME"	the file must not exist.

There are additional options, but the point is that if I had known about these extra capabilities, many of my programs would have been pages shorter. Well, that's the beauty of computers - there's always just one more thing to learn.

The LDOS & LS-DOS BASIC Reference Manual is sized at 5 1/2 x 8 1/2 and has 336 pages. It begins with an 'Introduction to Interpreter Basic', followed by a chapter on 'Editing and Program Maintenance', and 'Compiler Basic: Editing and Compiling'. These are important chapters, as they explain how Basic and Compiler Basic (EhnComp) work.

The meat and potatoes of the book, however, is the 'Basic Statements and Functions' section which is 227 pages long. Here, each statement and function is explained in detail, complete with a compatibility box and programming examples.

Finally, the chapter on 'Technical Information' includes a quick reference listing of statements, functions, algebraic, Boolean and logical operators. There is also discussions on 'variable storage format' and 'file buffer allocation', as well as 'Compiler BASIC Support Subroutine Descriptions', 'Compiler BASIC Z80 Assembler Introduction', and both compiler and interpreter Basic error code listings..

The LDOS & LS-DOS BASIC Reference Manual is a worthwhile addition to any TRS-80 owner's library, and it should be especially appealing to anyone using LDOS/LS-DOS Basic.

On a personal note, let me just say that Roy Soltoff has over the years provided much support for our Model I, III and 4's. He is continuously bringing out new quality products for machines that the industry declared dead 6-7 years ago. He is supporting us, so let's make sure that we in turn support his company, Misosys. Believe me, we need the expertise.

The LDOS & LS-DOS BASIC Reference Manual is $25.00, and it is available from:

MISOSYS, Inc.
P.O. Box 239
Sterling, VA 22170-0239
(703) 450-4181

Get your copy today.

MISOSYS, Inc.

Aerocomp Hardware is now available from MISOSYS

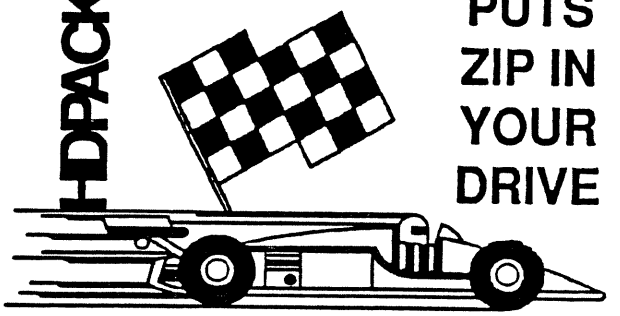
Model I DDen Controller (DDC)	$45 + $6S&H
Model III/4 FDC board	$45 + $6S&H
Model III/4 RS232 board	$45 + $6S&H
Model III/4 RS232 Kit	$50 + $6S&H
WD1002S-SHD HDC (new)	$75 + $5S&H
Aerocomp 5 Meg HD	$250 + S&H
Aerocomp 20 Meg HD	$400 + S&H
Aerocomp 40 Meg HD	$500 + S&H
MM CP/M 2.2 HD drivers	$29.95 + $3S&H

***The MISOSYS Quarterly* subscriptions**

Keep up to date with the latest information on MISOSYS products, programs, patches, and articles in a professional magazine format. A subscription to *TMQ* will provide you with information, news, and announcements concerning our entire product line and related machine environments. **As a special for new subscribers, we'll provide you with five issues and start you off with issue VLiv, and send you three past issues at no charge.** That's eight issues for the price of four! Subscriptions are: $25 US; $30 Canada; $35 Europe; $40 Australia

MISOSYS, Inc.
P. O. Box 239
Sterling, VA 20167-0239
703-450-4181
orders: 800-MISOSYS

HDPACK



PUTS ZIP IN YOUR DRIVE

When your hard drive files become fragmented with excessive directory extents, access speed degrades. Your program will finish in less than the optimum time. Now with our HDPACK utility, you can restore that ZIP to your computer. HDPACK will automatically, and intelligently, re-pack the fragmented files on your drive which will improve the performance of file access time.

HDPACK provides a visual display of its de-fragging operation, which in minutes can restore a ten-megabyte directory of files to a minimum number of extents. HDPACK can even work on floppy diskettes, too.

HDPACK, cat. no. M-33-400, is available for Model 4 DOS 6 only, and is priced at $39.95 + $3S&H.

ITEMS OF INTEREST

WANTED: Need docs, disks, advice, etc., to get a Bi-Tech Multiplexer operating.

Roy T. Beck
2153 Cedarhurst Dr.
Los Angeles, CA 90027
(213) 664-5059

FOR SALE: Desktop Model 4 - 128K - 2 single-sided drives - Micro-labs hi res board. This computer has been well cared for. Make a reasonable offer.

Lance Wolstrup
TRSTimes magazine
5721 Topanga Canyon Blvd., Suite 4
Woodland Hills, CA 91367
(818) 716-7154

FOR SALE: Complete set of 10 volumes of the ENCYCLOPEDIA FOR THE TRS-80. $60,00 plus shipping.

Lance Wolstrup
TRSTimes magazine
5721 Topanga Canyon Blvd., Suite 4
Woodland Hills, CA 91367

(818) 716-7154

PUBLIC DOMAIN GOOD GAMES FOR MODEL I/III.

GAMEDISK#1: amazin/bas, blazer/cmd, breakout/cmd, centipede/cmd, elect/bas, madhouse/bas), othello/cmd, poker/bas, solitr/bas, towers/cmd
GAMEDISK#2: cram/cmd, falien/cmd, frankadv/bas, iceworld/bas, minigolf/bas, pingpong/cmd, reactor/bas, solitr2/bas, stars/cmd, trak/cmd
GAMEDISK#3: ashka/cmd, asteroid/cmd, crazy8/bas, french/cmd, hexapawn, hobbit/bas, memalpha, pyramid/bas, rescue/bas, swarm/cmd
GAMEDISK#4: andromed/bas, blockade/bas, capture/cmd, defend/bas, empire/bas, empire/ins, jerusadv/bas, nerves/bas, poker/cmd, roadrace/bas, speedway/bas

Price per disk: $5.00 (U.S.)
or get all 4 disks for $16.00 (U.S.)

TRSTimes - PD GAMES
5721 Topanga Canyon Blvd. #4
Woodland Hills, CA. 91364

*** DR. PATCH ***

BRAND-NEW UTILITY FOR TRS-80 MODEL 4 AND LS-DOS 6.3.1

A 'MUST HAVE' FOR ALL LS-DOS 6.3.1 OWNERS.

DR. PATCH MODIFIES LS-DOS 6.3.1 TO DO THINGS THAT WERE NEVER BEFORE POSSIBLE.
COMPLETELY SELF-CONTAINED - MENU-DRIVEN FOR MAXIMUM USER CONVENIENCE.
FAST & SAFE - EACH MODIFICATION IS EASILY REVERSED TO NORMAL DOS OPERATION.

DISABLE PASSWORD CHECK IN FORMAT/CMD
FORMAT DOUBLE-SIDED AS DEFAULT
FORMAT 80 TRACKS AS DEFAULT
DISABLE VERIFY AFTER FORMAT
CHANGE 'DIR' TO 'D'
CHANGE 'CAT' TO 'C'
VIEW DIR/CAT WITH (I) PARAMETER AS DEFAULT
VIEW DIR/CAT WITH (S,I) PARAMETERS AS DEFAULT
CHANGE 'REMOVE' TO 'DEL'
CHANGE 'RENAME' TO 'REN'
CHANGE 'MEMORY' TO 'MEM'
CHANGE 'DEVICE' TO 'DEV'
DISABLE THE BOOT 'DATE' PROMPT
DISABLE THE BOOT 'TIME' PROMPT
DISABLE FILE PASSWORD PROTECTION
ENABLE EXTENDED ERROR MESSAGES

DISABLE PASSWORD CHECK IN BACKUP/CMD
BACKUP WITH (I) PARAMETER AS DEFAULT
BACKUP WITH VERIFY DISABLED
DISABLE BACKUP 'LIMIT' PROTECTION
DISABLE PASSWORD CHECK IN PURGE
PURGE WITH (I) PARAMETER AS DEFAULT
PURGE WITH (S,I) PARAMETERS AS DEFAULT
PURGE WITH (Q=N) PARAMETER AS DEFAULT
IMPLEMENT THE DOS 'KILL' COMMAND
CHANGE DOS PROMPT TO CUSTOM PROMPT
TURN 'AUTO BREAK DISABLE' OFF
TURN 'SYSGEN' MESSAGE OFF
BOOT WITH NON-BLINKING CURSOR
BOOT WITH CUSTOM CURSOR
BOOT WITH CLOCK ON
BOOT WITH FAST KEY-REPEAT

DR. PATCH IS THE ONLY PROGRAM OF ITS TYPE EVER WRITTEN FOR THE TRS-80 MODEL 4 AND LS-DOS 6.3.1. IT IS DISTRIBUTED EXCLUSIVELY BY TRSTIMES MAGAZINE ON A STANDARD LS-DOS 6.3.1 DATA DISKETTE, ALONG WITH WRITTEN DOCUMENTATION.

*** DR. PATCH *** $14.95

SHIPPING & HANDLING: U.S & CANADA - NONE
ELSEWHERE - ADD $4.00
(U.S CURRENCY ONLY, PLEASE)

TRSTimes magazine - dept. DP
5721 Topanga Canyon Blvd. #4
Woodland Hills, CA 91367

DON'T LET YOUR LS-DOS 6.3.1 BE WITHOUT IT!

GRAPHICS-90

NOW for Model I or III

The definitive graphics program that Radio Shack SHOULD have released is now, through the courtesy of author Larry Payne and the hard work of Gary Shanafelt, available from TRSTimes. Our agreement is to sell this wonderful package at COST. Neither Larry Payne, Gary Shanafelt nor TRSTimes will make a dime on this venture. The price charged will cover the cost of reproducing the manual, disks, and mailing only. The only one profiting is YOU!!

GRAPHICS-90

Model III $ 9.75
Model I $ 11.50

Shipping: Europe: add $4.50 air mail or $2.75 surface mail
Asia, Austrailia & New Zealand: add $5.25 air mail or $3.50 surface mail
Please specify Model I or III and DOS preference

TRSTimes magazine - dept. GR90

5721 Topanga Canyon Blvd. #4
Woodland Hills, CA 91367

RECREATIONAL & EDUCATIONAL COMPUTING

REC is the only publication devoted to the playful interaction of computers and 'mathemagic' - from digital delights to strange attractors, from special number classes to computer graphics and fractals. Edited and published by computer columnist and math professor Dr. Michael W. Ecker, REC features programs, challenges, puzzles, program teasers, art, editorial, humor, and much more, all laser printed. REC supports many computer brands as it has done since inception Jan. 1986. Back issues are available.

To subscribe for one year of 8 issues, send $27 US or $36 outside North America to REC, Att: Dr. M. Ecker, 909 Violet Terrace, Clarks Summit, PA 18411, USA or send $10 ($13 non-US) for 3 sample issues, creditable.

TRSTimes on DISK #9

Issue #9 of TRSTimes on DISK is now available, featuring the programs from the Jan/Feb, Mar/Apr and May/Jun 1992 issues:

TRSTimes on DISK is reasonably priced:

U.S. & Canada: $5.00 (U.S.)
Other countries: $7.00 (U.S.)

Send check or money order to:

TRSTimes on DISK
5721 Topanga Canyon Blvd. #4
Woodland Hills, CA 91367

TRSTimes on DISK #1, 2, 3, 4, 5, 6, 7 & 8 are still available at the above prices