TRSTimes

Volume 5. No. 1. - Jan/Feb 1992 - $4.00

HAPPY 1992

LITTLE ORPHAN EIGHTY

In late 1987, when 80-Micro announced that they would no longer support the TRS-80, the seeds of TRSTimes was planted. Would anyone be interested in a new TRS-80 magazine? Frankly, I had no idea; but, why not. It was certainly worth a try. So, with the help of some good friends, publicity was generated, and issue 1.1 saw the light of day.

I was genuinely surprised by the response. There were a great many more orphaned TRS-80 owners, hungry for information, than I had anticipated. Subscriptions came by the sackful, and TRSTimes was in business. What was conceived to be a fun project to be done on weekends turned into another fulltime job (still fun, though).

My goal was to publish six issues, and I never really thought about continuing past that. However, as you can see, this is issue 5.1 - the first issue of our fifth year - unbelievable!!

The reason 80-Micro left the TRS-80 market in 1988 is easy to understand. Their subscriber base was dwindling and, more importantly, the TRS-80 vendors were changing to PC products and no longer advertising in TRS-80 magazines. No advertising revenue - no fancy magazine, and therefore - bye, bye 80-Micro.

TRSTimes, of course, did not alter history when it appeared in 1988; people did not sell their PC's to buy a TRS-80, and vendors continued to hawk their wares in PC related magazines. Yet, even with the normal dwindling subscriptions and very little advertising, I envision a good year for TRSTimes, and for the TRS-80 community in general. I may be accused of being the eternal optimist, but I can still feel the enthusiasm of the readers. You, like I, have found a machine that does the job day in and day out. It is very reliable, hardly ever in need of repair. Tandy made great computers. I just wish that they had realized it!

In todays market, it is unheard of that a series of computers should still be supported after almost 15 years, but such is the case of our TRS-80. There are still dedicated people producing new goodies even though the monetary rewards are slim. There are still people willing to repair and upgrade our 'antiques' and, as you can see from this magazine, people are still tinkering, making the machines do things they were not intended to do. So, at least for the short term, our old Tandy machines will be going strong.

Happy New Year.

This issue brings a blend of new ideas, programs, recollections and tutorials. I hope you, the reader, will find things of interest to you.

The MEMDISK/GRAFDISK information seems to have evolved into a series. There has been much favorable response to this subject, so I will continue to try to make the MEMDISK/GRAFDISK utility execute faster. This month's article is an extension of the ideas presented in the Nov/Dec 1991 issue. Future issues will introduce some new patches, as well as a way to bypass the /JCL method of installing MEMDISK and loading the system.

Roy Beck dug out a great utility from the Misosys Quarterly, called BOOT5/CMD. It was written exclusively for LDOS 5.3.0. Unfortunately, it wouldn't run on LDOS 5.3.1. Not to worry... with help from several people, the program is now running flawlessly on this latest Model III DOS. If you have a hard disk, check this out.

For the newcomers, Paul Abernathy explains how to connect a modem to a Model III or 4.

A computer history buff, Roy Beck presents part 1 of his recollections of the Model I. If you are like me, you will find this miniseries fascinating. Though I was involved with the Model I from the very beginning, there are so many things I missed. Gee, to relive those days!!

The NEWDOS/80 fans have not been forgotten. The HINTS & TIPS section feature several items for that DOS. Also, for the Model I/III programmer, there is a collection of useful PEEKs and POKEs.

Who's in First is my contribution to the world of Little-League soccer. It features a type-in Basic program for the Model 4 that will help you become a famous and in-demand Little-League Secretary.

Allen Jacobs writes about Graphics-90. In his usual in-depth manner, he manages to talk about everything from PONG to SUPER-VGA, relating it all to this newly re-released program.

Finally, Mathieu Simons from Belgium got Graphic-90. Now, he wasn't satisfied to just use it. He tore into it, disassembled the code, and came up with the patches to make it run on Model I NEWDOS/80. Yes, now GRAPHICS-90 will run on both Model I and III. A good job, indeed.

To everyone contributing to this issue, I give my sincere thanks. It is enjoyable to play editor when there are so many good people willing to share their knowledge with the rest of us. Keep the information coming; we need it!

And now...Welcome to TRSTimes 5.1

TRSTimes magazine

Volume 5. No. 1. - Jan/Feb 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.

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

HARD DISK ON MODEL I LDOS 5.3.1

Thanks for issue 4.6.! I particularly appreciated Roy Beck's article; however, I don't think the problem is with RSHARD! Instead, I think Roy Soltoff's patch needs to be applied to LDOS 5.3.1.

Art McAninch
Borger, TX

Roy Beck informs me that, because of pressing obligations, he has not had time to get back to the Model I project. He assures me, however, that he will get back to it and make Model I LDOS 5.3.1 work with a hard drive larger than 5meg... stay tuned!

Ed.

PROFILE 3 (OR 4)

PROFILE is a very good data base program, but the print program stinks. How about a program, BASIC I guess, that could be altered to read and print PROFILE fields in a custom designed print-out?

Jim Savage
Clinton, MS

That sounds like a fun programming project. How about it - anyone out there willing to tackle this one? If not, as soon as I get caught up, I will take a look at the record format of PROFILE and see what I can do about it.

Ed.

MODEL III UPGRADE

I have a Model III with 48K and I would like to know if there is an upgrade for this Model. Is there a way of getting more K's?

Mary T. Kogut
New Port Richey, FL

Putting more RAM into a Model III, even if you could, would do you no good. Neither hardware nor software would support the additional memory. If you feel you absolutely must have more memory, your best bet is to get a used Model 4 and cram it with all the RAM you want.

Ed.

ONE CLOUDY DAY IN MOSCOW....

In years gone by, on one bleak and cloudy December day, in Moscow, a Communist Party official and his wife were looking out the window. In a half conscious comment, the official said to his wife, "Natasha, my dear. It looks as though we will soon have rain, this day."

His devoted wife countered, "I beg to differ with you but I believe you are wrong. Rudolf, my loving husband, it is sure to snow."

Only slightly bewildered by this statement of disagreement from his normally passive wife he stated, "It is not cold enough for that. Thus, I restate my prediction that it will rain today, my sweet wife."

His wife, always believing the news and hearing that the temperature will soon drop, repeated her previous disagreement. She supported it with the news she heard from the radio, and restated her disagreement, saying, "...Thus, my wonderful but uninformed husband, I tell you again, it will snow."

The official, feeling the need to reassert his authority in his own home, in light that it may be questioned in even some small way, by his most staunch supporter proclaimed, "Rudolf, the Red, Knows Rain, Dear....."

Allen W. Jacobs
North Hollywood, CA

Allen suggested that we print this now, while all of us here in Southern California can still remember the concepts of communism --- and rain.

Ed.

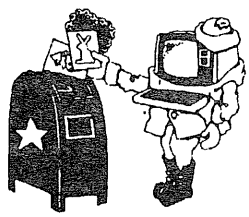
MODEL II SUPPORT

Some issues ago, you mentioned that TRSTimes might include some articles and programs for the Model II. I am waiting.... where are they?

Stan Jeffers
Ormond Beach, FL

I am afraid that they are not coming. At the time I made the tentative promise, I had just acquired a Model 16B. It turned out to have a major problem with the thinline drives. I blew out two, had them replaced, and thought I was on my way. In the process of writing some material for TRS-Times, my new drive :1 crashed and destroyed the disk. This was the last time I turned the machine on. I unplugged it, and it is now gathering dust in my garage. I may try one more time to get it running, but it will not be any time soon. Sorry!!

Ed.

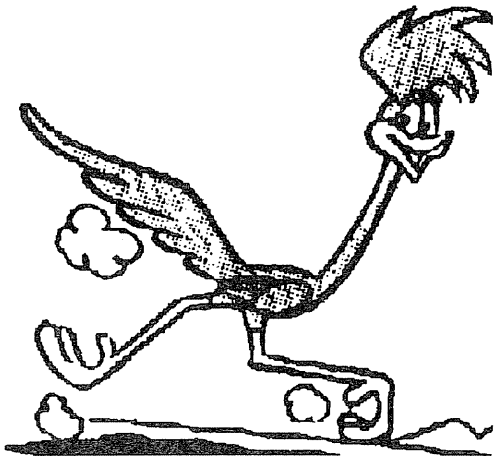


MEMDISK/GRAFDISK UPDATE

LS-DOS 6.3.1 AND DISKDISK

Model 4

By Lance Wolstrup



After reading an advance copy of the article 'A FASTER SYSTEM DRIVE' from the Nov/Dec 1991 issue of TRSTimes, Jim King turned to me and said: "This is a great idea - BUT - what a nuisance to have to insert a floppy to boot the memdisk!"

Jim was right. Though the idea of booting and loading MEMDISK or GRAFDISK from a special 'boot' floppy in drive :1 works beautifully, it is a nuisance to keep this disk handy.

"There's got to be a better way", Jim continued, "how about using DISKDISK?"

"Hmmm". Not a bad idea. It was certainly worth a try. And with that we got down to serious business. Sure enough, it worked like a charm.

For the benefit of readers who are not familiar with DISKDISK, let me explain what it is. In essence, DISKDISK is a utility, written by Roy Soltoff of Misosys, that allows you to break a disk (primarily a hard disk) into smaller segments - pretty much like 'sub-directories' in MS-DOS. The utility lets you create a disk on a disk (hence the name) with a directory of its own, completely separate from the disk it resides on. It is a very handy tool, and if you run your Model 4 (or Model III LDOS) with a hard-drive, you need DISKDISK. Contact Misosys for the latest version and price.

Using the original idea of a track-by-track BACKUP, rather than the lengthy file-by-file BACKUP to load the programs to MEMDISK/GRAFDISK, we worked out several ways to avoid the hassle of a disk in drive :1. While these methods are geared to cover as many machine configurations as possible, it is absolutely essential that the machine has 128K of memory and, in the case of GRAFDISK, a graphics board is installed. Also note that all GRAFDISK examples are for the RS graphics board. Owners of the Microlabs board must realize that they have approximately 12K less memory available.

OK, here goes.

CONFIGURATION #1 - GRAFDISK ONLY

(not enough memory for MEMDISK)

BOOT AND LOAD GRAFDISK DIRECTLY FROM SYSTEM DISK IN DRIVE :0 (WITHOUT DISKDISK)

Minimum configuration: two single-sided drives
Non-LS-DOS files needed: GRAFDISK/DCT

1. Format a 40 track, single-sided disk in drive :1. Make sure that the name of the disk is GRAFDISK and the directory is placed on track 1.
FORMAT :1 (name="GRAFDISK"),mpw=,dden,cyl=40, dir=1,q=n,abs)

2. Make the disk in drive :1 a bootable system disk by copying the /SYS files to it in the CORRECT order.
a. BACKUP SYS0/SYS:0 :1 (S)
b. BACKUP /SYS:0 :1 (S)

3. Get rid of the unneeded /SYS files.
REMOVE SYS13/SYS.SYSTEM6:1

If DEBUG is not needed:
REMOVE SYS5/SYS.SYSTEM6:1
REMOVE SYS9/SYS.SYSTEM6:1

4. Copy essential files.
COPY GRAFDISK/DCT.DRIVER:0 :1
COPY BACKUP/CMD.UTILITY:0 :1

5. Copy the other files you would like in the GRAFDISK.
COPY FORMAT/CMD.UTILITY:0 :1
COPY TED/CMD.UTILITY:0 :1
etc.

Make sure that you do not use diskspace past the first granule of track 20. To check this, type FREE :1. There MUST be at least two granules free on track 20 and tracks 21 through 39 MUST also be empty.

6. Use TED to write the following /JCL file to automate the installation and loading of the GRAFDISK, then save it as GRAFBOOT/JCL:1

```
SYSTEM (DRIVE=2,DRIVER="GRAFDISK")
B
Y
BACKUP :0 :2
Y
SYSTEM (SYSTEM=2)
SYSTEM (DRIVE=1,SWAP=2)
```

7. Insert the newly created disk in drive :0 and reboot the machine. This disk is now your 'system' disk. It will boot LS-DOS and, after the date and time prompts, you can now type: DO GRAFBOOT to set up and load the GRAFDISK very quickly.
You can make the whole process automatic by, from LS-DOS Ready, typing: AUTO DO GRAFBOOT
Now, on every boot with this disk, after entering the date and time, the GRAFDISK will be set up and loaded automatically.

CONFIGURATION #2 - MEMDISK

BOOT AND LOAD MEMDISK DIRECTLY FROM SYSTEM DISK IN DRIVE :0 (WITH DISKDISK)

Minimum configuration:	two single-sided drives
Non-LS-DOS files needed:	DD/CMD
	DDFORM/CMD

1. Format a normal 40 track, single-sided, double-density disk in drive :1.
FORMAT :1 (name="MEMBOOT",mpw=,dden,cyl=40,q=n,abs)

2. Make the disk in drive :1 a bootable system disk by copying the /SYS files to it in the CORRECT order.
a. BACKUP SYS0/SYS:0 :1 (S)
b. BACKUP /SYS:0 :1 (S)

3. Copy essential files.
COPY MEMDISK/DCT.DRIVER:0 :1
COPY BACKUP/CMD.UTILITY:0 :1
COPY DD/CMD:0 :1
(it is assumed that the two DISKDISK program files, DD/CMD and DDFORM/CMD, reside on the disk in drive :0.)

4. Create a DISKDISK called MEMBOOT on the disk in drive :1.
a. DDFORM MEMBOOT:1
b. select default **5**
c. select default **1**
d. select default **d**
e. select 14 cylinders **14**

5. You now have 46.5K left on your disk in drive :1. Copy the utilities/programs you need from drive :0 to drive :1. Yes, I know you really had 49.5K left, but you MUST leave 3K free for the /JCL file which we will write in a minute, as well as for the SYSTEM/JCL generated by the system.
For example, you can:
BACKUP BASIC/:0:1 (I)
etc.

6. Open the DISKDISK you just created and configure it as any drive you are not using. Drive :7 is a good bet.
DD :7 MEMBOOT

7. Copy the /SYS files to the DISKDISK in drive :7.
BACKUP /SYS:0 :7 (S)

8. Remove the unneeded /SYS files from the DISKDISK in drive :7.
REMOVE SYS0/SYS.SYSTEM6:7
REMOVE SYS13/SYS.SYSTEM6:7

If DEBUG is not needed
REMOVE SYS5/SYS.SYSTEM6:7
REMOVE SYS9/SYS.SYSTEM6:7

9. Copy other utilities you would like to the DISKDISK in drive :7. For example:
COPY FORMAT/CMD.UTILITY:0 :7
COPY TED/CMD.UTILITY:0 :7

10. Change the diskname of the DISKDISK in drive :7 to MEMDISK (the diskname - not the filename).
ATTRIB :7 (name="MEMDISK")

11. Use TED to write the following /JCL file to automate the installation and loading of the MEMDISK, then save it as MEMBOOT/JCL:1

```
SYSTEM (DRIVE=2,DRIVER="MEMDISK")
D
D
Y
DD :7 MEMBOOT
BACKUP :7 :2
SYSTEM (SYSTEM=2)
SYSTEM (DRIVE=1,SWAP=2)
```

12. Insert the newly created disk in drive :0 and reboot. This disk is now your 'system' disk. it will boot LS-DOS and, after the date and time prompts, you can now type: DO MEMBOOT to set up and load the MEMDISK very quickly.
You can make the whole process automatic by, from LS-DOS Ready, typing: AUTO DO MEMBOOT

CONFIGURATION #3 - GRAFDISK

BOOT AND LOAD GRAFDISK DIRECTLY FROM SYSTEM DISK IN DRIVE :0 (WITH DISKDISK)

Minimum configuration:	two single-sided drives
Non-LS-DOS files needed:	DD/CMD
	DDFORM/CMD
	GRAFDISK/DCT

1. Format a normal 40 track, single-sided, double-density disk in drive :1.
FORMAT :1 (name="GRAFBOOT",mpw=,dden,cyl=40,q=n,abs)

2. Make the disk in drive :1 a bootable system disk by copying the /SYS files to it in the CORRECT order.
a. BACKUP SYS0/SYS:0 :1 (S)
b. BACKUP /SYS:0 :1 (S)

3. Copy essential files.
COPY GRAFDISK/DCT:0 :1
COPY BACKUP/CMD.UTILITY:0 :1
COPY DD/CMD:0 :1
(It is assumed that the two DISKDISK program files, DD/CMD and DDFORM/CMD, reside on the disk in drive :0

4. Create a DISKDISK called GRAFBOOT on the disk in drive :1.
a. DDFORM GRAFBOOT:1
b. select default **5**
c. select default **1**
d. select default **d**
e. select 21 cylinders **21**

5. You now have 7.5K left on your disk in drive :1. Copy the utilities/programs you need from drive :0 to drive :1. Yes, I know you really had 10.5K left, but you MUST leave 3K free for the /JCL which we will write in a minute, as well as for the SYSTEM/JCL generated by the system.
For example, you can:
COPY DISKCOPY/CMD.UTILITY:0 :1
COPY TAPE100/CMD.UTILITY:0 :1

6. Open the DISKDISK you just created and configure it as any drive you are not using. Drive :7 is a good bet.
DD :7 GRAFBOOT

7. Copy the /SYS files to the DISKDISK in drive :7.
BACKUP /SYS:0 :7 (S)

8. Remove the unneeded /SYS files from the DISKDISK in drive :7.
REMOVE SYS0/SYS.SYSTEM6:7
REMOVE SYS13/SYS.SYSTEM6:7
If DEBUG is not needed
REMOVE SYS5/SYS.SYSTEM6:7
REMOVE SYS9/SYS.SYSTEM6:7

9. Copy other utilities you would like to the DISKDISK in drive :7. For example:
COPY FORMAT/CMD.UTILITY:0 :7
COPY TED/CMD.UTILITY:0 :7
etc.

10. Change the diskname of the DISKDISK in drive :7 GRAFDISK (the diskname - not the filename).
ATTRIB :7 (name="GRAFDISK")

11. Use TED to write the following /JCL file to automate the installation and loading of the GRAFDISK, then save it as GRAFBOOT/JCL:1.

```
SYSTEM (DRIVE=2,DRIVER="GRAFDISK")
B
Y
DD :7 GRAFBOOT
BACKUP :7 :2
SYSTEM (SYSTEM=2)
SYSTEM (DRIVE=1,SWAP=2)
```

12. Insert the newly created disk in drive :0 and reboot. This disk is now your 'system' disk. It will boot LS-DOS and, after the date and time prompts, you can now type: DO GRAFBOOT to set up and load the GRAFDISK very quickly.
You can make the whole process automatic by, from LS-DOS Ready, typing: AUTO DO GRAFBOOT

CONFIGURATION #4 - MEMDISK

BOOT AND LOAD MEMDISK DIRECTLY FROM HARD DRIVE (WITH DISKDISK)

Minimum configuration:	one single-sided drive
	hard drive
Non-LS-DOS files needed:	DD/CMD
	DDFORM/CMD

1. Create a DISKDISK called MEMBOOT on your hard drive (my first partition is drive :3, so I will use it in this example).
a. DDFORM MEMBOOT:3
b. select default **5**
c. select default **1**
d. select default **d**
e. select 14 cylinders **14**

2. Open the DISKDISK you just created and configure it as any drive you are not using. Drive :7 is a good bet.
DD :7 MEMBOOT

3. Copy the /SYS files to the DISKDISK in drive :7.
BACKUP /SYS:0 :7 (S)

4. Remove the unneeded /SYS files from the DISKDISK in drive :7.
REMOVE SYS0/SYS.SYSTEM6:7
REMOVE SYS13/SYS.SYSTEM6:7

If DEBUG is not needed
REMOVE SYS5/SYS.SYSTEM6:7
REMOVE SYS9/SYS.SYSTEM6:7

5. Copy other utilities you would like to the DISKDISK in drive :7. For example:
COPY FORMAT/CMD.UTILITY:0 :7
COPY TED/CMD.UTILITY:0 :7

6. Change the diskname of the DISKDISK in drive :7 to MEMDISK (the diskname - not the filename).
ATTRIB :7 (name="MEMDISK")

7. Use TED to write the following /JCL file to automate the installation and loading of the MEMDISK, then save it as MEMBOOT/JCL:3

```
SYSTEM (DRIVE=2,DRIVER="MEMDISK")
D
D
Y
DD :7 MEMBOOT
BACKUP :7 :2
SYSTEM (SYSTEM=2)
SYSTEM (DRIVE=1,SWAP=2)
```

Now, whenever you wish to boot and load the MEMDISK, from LS-DOS Ready type: DO MEMBOOT:3
The MEMDISK can be activated and loaded automatically each time LS-DOS is booted by typing:
AUTO DO MEMBOOT:3

CONFIGURATION #5 - GRAFDISK

BOOT AND LOAD GRAFDISK DIRECTLY FROM HARD DRIVE (WITH DISKDISK)

Minimum configuration:	one single-sided drive hard drive
Non-LS-DOS files needed:	DD/CMD DDFORM/CMD GRAFDISK/DCT

1. Create a DISKDISK called GRAFBOOT on your hard drive (my first partition is drive :3, so I will use it in this example).
a. DDFORM GRAFBOOT:3
b. select default **5**
c. select default **1**
d. select default **d**
e. select 21 cylinders **21**

2. Open the DISKDISK you just created and configure it as any drive you are not using. Drive :7 is a good bet.
DD :7 GRAFBOOT

3. Copy the /SYS files to the DISKDISK in drive :7.
BACKUP /SYS:0 :7 (S)

4. Remove the unneeded /SYS files from the GRAFDISK in drive :7.
REMOVE SYS0/SYS.SYSTEM6:7
REMOVE SYS13/SYS.SYSTEM6:7

if DEBUG is not needed
REMOVE SYS5/SYS.SYSTEM6:7
REMOVE SYS9/SYS.SYSTEM6:7

5. Copy other utilities you would like to the DISKDISK in drive :7. For example:
COPY FORMAT/CMD.UTILITY:0 :7
COPY TED/CMD.UTILITY:0 :7
etc.

6. Change the diskname of the DISKDISK in drive :7 to GRAFDISK (the diskname - not the filename).
ATTRIB :7 (name="GRAFDISK")

7. Use TED to write the following /JCL file to automate the installation and loading of the GRAFDISK, then save it as GRAFBOOT/JCL:3

```
SYSTEM (DRIVE=2,DRIVER="GRAFDISK")
B
Y
DD :7 GRAFBOOT
BACKUP :7 :2
SYSTEM (SYSTEM=2)
SYSTEM (DRIVE=1,SWAP=2)
```

Now, whenever you wish to boot and load the GRAFDISK, from LS-DOS Ready type: DO GRAFBOOT:3
The GRAFDISK can be activated and loaded automatically each time LS-DOS is booted by typing:
AUTO DO GRAFBOOT:3

You might wonder why I bother going through all this trouble when, as previous articles in this series have described, the GRAFDISK package includes two utilities, GDSAVE and GDLOAD, that should eliminate most of the steps covered in this writing.

GDSAVE/CMD is used to copy the contents of an already established GRAFDISK to a disk file. GDLOAD/CMD is used to initialize a GRAFDISK and then copy the needed/wanted files to it in one step from the disk file created by GDSAVE. Indeed, this is much easier than what we have discussed here and in the previous issues - BUT - from personal experience I know that GDLOAD does not always work correctly.

My Model 4P has a 15 meg VR DATA hard drive. GDSAVE seemed to work just fine with this machine. However, from day one, GDLOAD has refused to cooperate with the VRHARD driver, shoving nothing but garbage up to the GRAFDISK.

On the other hand, my desktop Model 4 with a 15 meg Radio Shack hard drive worked fine with GDLOAD. Establishing a GRAFDISK was fast and enjoyable. Then I added a 20 meg slave drive and, at this point, GDLOAD balked. I now have the same results as with my Model 4P - nothing but garbage is being sent up to the GRAFDISK.

So, rather than spending the time disassembling GDSAVE/CMD and GDLOAD/CMD and plowing through the code looking for possible fixes, I now use the ideas from this article to set up my GRAFDISK/MEMDISK. My desktop Model 4 (with the 15 meg RS hard drive and the 20 meg slave drive) uses configuration #4. My Model 4P (with the 15 meg VR hard drive and a Radio Shack graphics board) uses configuration #5.

To speed things up just a little more, MEMDISK/DCT v6.3.1 can be patched to skip the format verification. This chops a few seconds off the setup time - and it all helps!

PATCH MEMDISK/DCT.UTILITY (D06,76=00 00 00: F06,76=CD 59 36)

PATCH MEMDISK/DCT.UTILITY (D06,7B=00 00 00: F06,7B=CD 59 36)

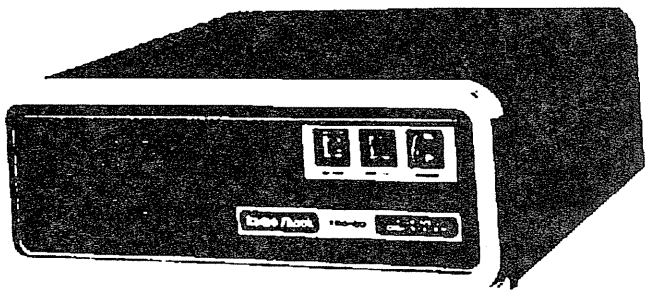
PATCH MEMDISK/DCT.UTILITY (D06,80=00 00 00: F06,80=CD 59 36)

PATCH MEMDISK/DCT.UTILITY (D06,85=00 00 00: F06,85=CD 59 36)

As always, never,never patch your master diskette. As a matter of fact, don't even make changes on your working DOS diskette; rather, make a backup and apply the patches to it. Then, if you like the changes, either copy the altered file to your working DOS diskette or apply the patches to it from scratch.

Boot your Hard Drive in LS-DOS and LDOS without a floppy!

By Roy T. Beck



Why should we have to use a floppy disk to boot a TRS hard disk machine? IBM users don't, and our machines are as capable as theirs. The answer is in the boot ROM's. The only correct one is the non-gate array Model 4P. It has the correct ROM code and floppy controller chip. All it needs is some revised code in the BOOT and SYS files of the HD, and it can take off and run from power-on, with no key-presses by the operator. M.A.D. Software has the missing piece, the software patches, for sale. With their patches on your LS-DOS or TRSDOS 6.3 BOOT and SYS0 files, you can autoboot, and faster than an IBM at that! And the price is only $15. Such a deal!

What about our other machines? The gate array 4P is almost right, but not quite. Our designing friends at Ft. Worth saved a nickel, again, and substituted a different floppy disk controller (FDC) when they designed the gate array version, but used the same ROM as in the non-gate array machine. This let a small multilegged creature (a bug) slip into the design, and the gate array 4P is not fully reliable when autobooting. Most times, the machine autoboots just fine. The malfunction, (which is random), sometimes allows the FDC to get into an unknown state, and in the recovery process it needs to read the index hole of the floppy disk in drive :0. Any disk is acceptable to it, even a blank; all it wants to do is to sense the index hole going by. After that, it continues the autoboot cycle. But regardless of what the FDC is trying to do, this defeats the autoboot! Frequently, another push of the RESET key will succeed, but as I said, it's random. Since I have only gate array 4P's, I quickly got bit by this bug! M.A.D. offers a replacement ROM for the gate array 4P's to cure this behavior. I have this in one of my machines, and it does solve the problem. There is a software workaround for the problem, but the new ROM is the best answer. Autobooting is a pleasure. This version, with the software and the new ROM, goes for $35.

The modified 4P ROM also knows whether or not your HD boot track has valid autoboot code. It will not attempt to load a bum image. This obviates the need for hitting the F2 key. Just power on and let the 4P go. The new ROM includes some other niceties, such as loading the ROM image for Mod III operation from floppy in less than six seconds. Now that's moving right along!

M.A.D. also offers the option (for an extra $10) of having your name and phone number or whatever (up to 92 characters) in the Mod 4P ROM. This could be helpful in proving ownership!

The Mod 4 and 4D family is another kettle of fish. Their ROM's know from nothing when it comes to autobooting, and to accomplish this, the owner must replace a ROM in all of them, both the non-gate array and the gate array types. M.A.D. also offers appropriate ROMs for all these models.

The upgrade package for the Mod 4 non-gate array is $30, including the ROM and the software. The gate array 4 and 4D package costs $35, the difference being for the larger ROM required. These ROMs also have some operating improvements, including a delay while waiting for a HD to come up to speed in unattended service, such as a BBS.

M.A.D. offers versions of its ROMs for four different HD controller setups, your choice. These include the stock RS drive and HDC, the MISOSYS SCSI setup, the Powersoft Driver with RS drive, and the Powersoft Series I driver with WD 100x HDC and host adapter addressed at 78-7FH. Some other features include better key debouncing, adjustable keyboard repeat rate, and correct time of day clock even when the system is running at 4 Mhz.

A utility, FINDZERO is included to identify which of your HD partitions is using the first head and track of the primary HD. This knowledge is essential to the successful installation of the AUTOBOOT, and this BASIC routine provides the answer.

For those of you using the XLR8ER, M.A.D. also includes some nice extra features. Included is a program named XLR8SET which installs necessary Houde' type patches to allow the XLR8ER board to operate with autoboot.

Another utility is a program named FORCEHI which allows the operator to force certain driver programs to install in high memory, and then allows other programs to

install in low memory. Very convenient and useful to those who need the capability.

M.A.D. also includes a command named OOPS for use when you have typed a long command line full of parameters, and get a syntax error. OOPS allows you to edit the command line and then re-execute it. OOPS cleverly uses the built-in BASIC editor for its operations, so very little code is added to the DOS. OOPS can be operated as a stand alone program, called by the OOPS command, or can be integrated into SYS13/SYS, from which it can be called by the simple command *.

Altogether, M.A.D.'s autoboot and related functions is one of those things which, after you buy it, you wonder why you didn't do it sooner! Don't miss out on this. Since I read M.A.D. Software is planning to close its doors next February, now is the time to upgrade your machine! M.A.D.'s address is Box 331323 Ft. Worth, TX 78163-1323.

And now here is a goody for those of you with both LS-DOS and LDOS on the same hard drive. Because both DOS's use the same directory structure, it is easy as pi (?) to have both DOS's on the same hard drive, operating out of different partitions, and even sharing partitions in common for easy transfer of files from LDOS to LS-DOS and vice versa. Up to now, however, it was necessary to have a boot floppy for LDOS to get it running on your HD system.

The ability to boot LDOS 5.3.0 from within LS-DOS 6.x without a boot disk is now available. This software is identified by the name BOOT5, and is offered by Adam Rubin as public domain software. A finished version is available on MISOSYS's DiskNotes 5.4, along with the ASM listing, obtainable for $10 from MISOSYS. This version will boot up LDOS 5.3.0 by simply executing an LS-DOS file named BOOT5/CMD. This file will call in SYS0/SYS of LDOS, which will load SYS1/SYS and CONFIG/SYS of LDOS. As soon as these files are loaded, the Model 4 or 4D is running under LDOS with normal access to the hard disk. BOOT5 also understands the Model 4P, and will load the MODELA/III ROM image as well. This is a very foxy piece of software, and offers you Model 4/4P/4D owners a great boon. For reference and more information, read Adam Rubin's article starting on page 29 of The MISOSYS Quarterly Volume 5, Number 4, available from MISOSYS. For their address and phone number, see their ad elsewhere in this issue.

You may have noted that the above works with LDOS 5.3.0. But what about 5.3.1, the latest version? Several people have looked at this, and the solution is a simple series of eight 1-byte patches to BOOT5/CMD, required because MISOSYS's update and reassembly of LDOS as 5.3.1 relocated some addresses internally. The patches are a joint effort by Art McAninch of Texas and Lance Wolstrup of TRSTimes, aided and abetted slightly by yours truly. Although it is not his code, Roy Soltoff has also contributed some advice.

I suggest the two versions of BOOT5 should be distinguished by renaming them BOOT530/CMD and BOOT531/CMD, respectively, but that's up to the user.

The patches for 5.3.1 are as follows:

PATCH BOOT5/CMD (D01,92=6A:F01,92=62)

PATCH BOOT5/CMD (D03,5F=31:F03,5F=30)

PATCH BOOT5/CMD (D04,2B=31:F04,2B=30)

PATCH BOOT5/CMD (D04,D3=D9:F04,D3=F6)

PATCH BOOT5/CMD (D04,D9=50:F04,D9=6D)

PATCH BOOT5/CMD (D04,DC=51:F04,DC=6E)

PATCH BOOT5/CMD (D04,E1=D3:F04,E1=F0)

PATCH BOOT5/CMD (D04,E4=12:F04,E4=2F)

Gee, it's nice to be able to boot both LS-DOS and LDOS without a floppy! Why have we all waited so long? Many thanks to you, M.A.D. Software, and also to you, Adam Rubin, for writing and making available BOOT5/CMD!

HOW TO CONNECT A MODEM TO THE MODEL 3/4/4D

By Paul G. Abernathy II

REQUIRED MATERIALS

- 1 Computer (of course)
- 1 Rs232-C Cable (R/S 26-1408)
- 1 Modem
- 1 Terminal Program (your choice)

PREPARING FOR THIS JOB

The first thing to do is assemble all of the required materials. TURN OFF your computer, and CAREFULLY turn it over onto its side (Model 4D Computers, turn the computer around so that you are looking at the back side of the computer). NEVER connect ANY cables to your computer while it is still on, you can cause SERIOUS DAMAGE to it by doing so!

CONNECTING THE RS232-C CABLE TO A MODEL 3/4

First of all, let us take a look at the Rs232-C cable. Simply grab an end and take a look at it. You will notice that the connector has a defined shape with 25 pins in it. You are holding what is termed a MALE 25 pin connector. Your cable has two of these MALE connectors on it, one on each end. The connector is designed so that when connecting it to your computer, you can not get it on backwards (but, be careful, one of my girls did the 'Impossible'!).

The Male connector must plug into a FEMALE connector. The FEMALE connector has exactly the same defined shape as the MALE, with the exception that instead of having the 25 pins, it has 25 holes (stop thinking that way!).

Now, with keeping the shape of the MALE connector in mind, take a look at the bottom of your computer, and find the FEMALE connector (The FEMALE connector is BLACK on ALL Standard Equipment.)

- Model 3 Users - It is located at the back -center of the computer.
- Model 4 Computers - It is located at the back-center of the computer, just in front of the Expansion Interface Connector.
- Model 4D Computers - Looking at the back side (not the bottom) of the computer, It is located slightly towards the left hand side, at the bottom.

Find it? If not, Radio Shack placed on the bottom of every computer they made a sticker describing the various connectors. Simply take a look at the diagram on that sticker and find the plug labeled Rs-232. Now visually locate the actual FEMALE connector, and take a look at it.

Are you still having trouble finding it? If so, get a 1988 Radio Shack Computer Catalog and turn to page 37. Towards the bottom of page 37 to the left of the Heading 'Computer and Peripheral Connectors' locate item #2 in the picture. I know it is kinda small, but that is what the FEMALE connector looks like on your computer (Only it will be black instead of white.) Now again, take a look at the bottom of your computer, and find that connector.

Ok, we are now ready to do the actual connecting of the cable to the computer. Take the either one of MALE connectors on the Rs232-C cable in hand. Next, take the MALE connector, carefully position it so that it will match up to the FEMALE connector on the computer, and plug the MALE connector into the FEMALE connector. A tiny bit of force may be required to make a snug fit. However, if it does no seem like it is going on correctly DO NOT FORCE IT! Simply take a look at the MALE connector shape, Match it to the FEMALE connectors shape, and try again. If you are still having trouble, make sure all of the pins are straight inside of the MALE connector, and proceed with the connection again.

That wasn't hard was it! We are done with the computer for a moment, so you can turn the computer right-side up once again (or around if it is a 4D). Just make sure that the Rs232-C cable does not come out or get twisted into any wierd shapes (if it does, it could break a wire in the cable, and you will be out $17.95). However, do **NOT** plug in the computer yet.

CONNECTING THE RS232-C CABLE TO THE MODEM

Congratulations, you have completed phase #1, now on to phase #2. We are now going to connect the Rs232-C cable to your favorite modem. Since there are MANY different types of modems, this section will be somewhat generic, but is sure to still pull you through!

First, lets pick-up that modem and find the FEMALE connector. Take a look at the back of it. Do you see it? If not, take a look at the sides, a couple of modem manufacturers place them there.

Ok, we are now ready to do the actual connecting of the cable to the modem. Take the remaining MALE con-

ector on the Rs232-C cable in hand. Next, take the MALE connector, carefully position it so that it will match up to the FEMALE connector on the modem, and plug the MALE connector into the FEMALE connector. A tiny bit of force may be required to make a snug fit. However, if it does no seem like it is going on correctly DO NOT FORCE IT! Simply take a look at the MALE connector shape, Match it to the FEMALE connectors shape, and try again. If you are still having trouble, make sure all of the pins are straight inside of the MALE connector, and proceed with the connection again.

CONNECTING THE MODEM TO THE TELEPHONE LINE

Don't get too impatient, we are almost finished. The Next Job at hand is to connect the modem to your telephone line. First, one thing to note: this tutorial assumes that 1) your telephone line is connected to a standard modular wall jack (its kinda square with a little projection that holds it in place) , and 2) that the jack is within a reasonable distance (2-3 ft) from where your modem and computer will sit. If you failed #1, you can go to the local discount store and buy an adapter if your wall jack plug has 4 prongs (about $2.99 at our local Walmart), or you may simply change the entire plug over by buying a modular wall jack installation kit (around $3.99). IF YOU ARE NOT A HANDYMAN AT ALL (i.e. can't use a screwdriver and follow about 5 steps), then call the telephone company and have them to come out and do the installation for you. If you failed #2, you have 2 options; A) again, run down to the nearest discount store and by a telephone line modular extention cable (not a handset extention cable!). They run about $4.99 for a 25 ft piece. Option B) is to run a complete line of new cable, and put in the wall jack yourself. (If you choose option B, You will probably want the telephone company to do it because it can be a real headache!).

Now, assuming that all of the above is fine and in working order, lets do it. There are two different ways in which modems come; 1) with the telphone line already connected (built in) to the modem - in which case, simply skip down to 'Connecting The Telephone To The Modem', or 2) those which must have the telephone line manually plugged in (thats about 97% of us). First of all, did your modem come with a 6' telephone wire with a modular plug on each end? If not, once again, go and pick-up one at the local discount store (about $1.90). Now, some modems come with a feature which allows you to connect an optional telephone to it. Take a look at the back and sides of the modem. Does the modem have one or two modular jacks? If your modem has only 1 modular jack, simply go down to the next paragraph. If your particular modem has 2 modular jacks built in, you will need to grab the modem owner's manual and find out which of the modular connector jacks is for the optional telephone and which connector jack is for the 6' telephone wire going to the wall jack. Take a look under 'Installation'. Find it? If so, down to the next paragraph, otherwise, most manufacturers usually label the two plugs as PHONE, and LINE.

Ok, now take in hand one of the ends of the 6' telephone wire. This particular plug is also made to plug in only one way. Simply position the end of the plug so that it matches the shape of the connector jack on the modem, and plug it in. Next, take the other plug end of the 6' telephone wire, and position it so that it matches the shape of the connector in the wall jack. Got it? Ok, now plug the end into the wall jack.

CONNECTING THE TELEPHONE TO THE MODEM

Please Note: This Section is only for those of us whom have **2** modular jacks built into our modems. If your modem has only one modular jack built in, please go to the next section, 'Putting It All Together'.

Second Note: This section is optional, if you do not have a spare telephone now, simply keep a copy of these instructions, and come back to them when you do- simply go to the next section, 'Putting It All Together'. (I don't reccommend going out and buying one especially for your modem use - that could get kinda expensive, depending upon your tastes.)

Ok, for those of you who are still reading, lets get to it! Take in hand one of the plug ends of the telephone wire which came with the telephone. This particular plug is also made to plug in only one way. Simply position the end of the plug so that it matches the shape of the connector jack on the telephone, and plug it in. Next, take the other plug end of the telephone wire, and position it so that it matches the shape of the remaining modular connector jack on the modem. Got it? Ok, now plug the end into the wall jack.

Now, for the big test...To see if the telephone works! Simply pick-up the hand receiver and see if there is a dial tone. If there is, then on to the next section, otherwise, you need to go back to the top of this section, and start over. Check to make sure every connection is correct. (9 times out of 10 you can bet that you have the 2 wires switched backwards on the modem. Try switching the 2 wires that are plugged into the modem).

PUTTING IT ALL TOGETHER

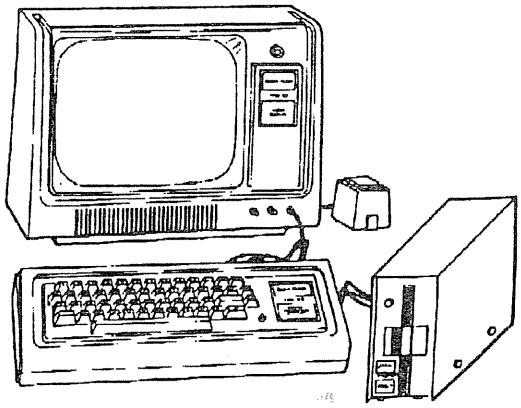
Ah, finally the last step! We are done messing with all those cables, and are now ready to plug in everything and give it a try! Follow The Following Steps IN ORDER:

* Plug in your computer
* Plug in your modem
* Turn on the power to your modem
* Turn on your computer, and boot in your favorite DOS

THAT'S IT, WE ARE DONE! Run your favorite terminal program and you are ready to enter the world of telecommunications. Good Luck, and have fun.

My Recollections of the Model I

By Roy T. Beck



The title of this article represents a small amount of literary license in that I have borrowed on the recollections and knowledge of other people in its preparation, with special credit due to Art McAninch, Jr. (806)273-6378 122 Pecan Borger Texas 79007. Art is a great repository of Model I hardware, software and lore, and is really the man to talk to about the Model I.

The Model I was Radio Shack's first venture into computer manufacturing. It was not their first offering in the market place. Indeed, I have an old catalog of theirs which predates the RSC-x series of catalogs, and it lists some S-100 stuff! But the Model I (it did not originally have a model number, it was just the TRS-80 Computer) was my first desk top machine, and like other first loves, it sticks in my memory.

I bought mine with the LARGE memory, 16K, if you please, as opposed to the standard 4K which normally came with the Level I! I believe that cost me an extra $100 when I bought it. It was the Level 1 version because that's what the store had on hand and I was anxious to get it home and play with it. I upgraded to Level 2 at a later date (for another $100).

What is the difference between the Level 1 and Level 2 ROM's? The Level 1 ROM was a 4K ROM which contained the cassette operating system (OS) and a version of BASIC known in that era as Palo Alto Tiny BASIC, written by a gentleman named Wang, I understand. The Palo Alto BASIC was very limited in size, and could only accommodate TWO strings, A$ and B$. It also was very limited in variables, 26 being allowed, and those were the letters A-Z, of course. The OS was cassette only, and was not a DOS. That came later. Tiny BASIC was delightful in one respect; most commands could be abbreviated to a single letter. You sure could write tight code with it!

Level I was a marvel of simplicity and terseness. It only had two diagnostics, which were WHAT? and SORRY. WHAT? meant you had a syntax error, and SORRY was sort of a catchall, which could mean almost anything.

The Level 2 ROM was a 12K set of 1, 2, or 3 chips (there were several versions of the Level 2 chip set) which was still a cassette OS, (by Microsoft) but which had built in "hooks" for a future DOS to connect to. Level 2 allowed far more variable names for both numeric and string variables, and you could do a lot more with it. Unfortunately, you now had to spell out all the commands; ? for PRINT was the only remaining single letter command abbreviation. Also, L? would not work in place of LPRINT; (I tried). But you could still pack BASIC statements with no spaces between symbols. That was the era when we learned how to pack BASIC so tightly it became unreadable! A write-only language, you might say! I remember several people wrote some clever utilities which would unpack and print out any BASIC listing in elegant and readable form, even a program which could not otherwise be easily analyzed.

For a while, a trick was popular which involved having both Level 1 and 2 available in the same machine, selected by a toggle switch. The idea was clever, but it was difficult to implement because of the varied physical configuration of the level 2 ROMs and because of the additional power required. The switch selected an enable line of one or the other ROM(s), giving either Level 1 or 2 operation. Shifting on the fly was not permissible; the machine would bomb. A reboot was necessary in order to achieve operation in the other BASIC.

The power supply for a Model I was a small black cube, about 3" on an edge which plugged into the power strip and had a trailing cord which plugged into the keyboard or EI. The cube was factory sealed with solvent, and could only be opened with a sharp knife or a hacksaw. When a power supply died, the knowledgeable owners knew to open it up by whatever means to gain access to the small fuse soldered inside! Never before have I seen a fuse hidden inside a piece of equipment such that it was not reasonably replaceable. Two of my power supplies are presently taped together because of this "undocumented feature".

As originally designed, the Model I keyboard had +5 Volts available on one pin of its 40 line bus. I never knew why it was there, but in theory you could use this 5 volts to power some outboard accessory. However, RS soon decided this feature was a liability, as you could easily overload the power supply serving the keyboard. Thereafter, whenever your Model I keyboard went into a RS

Computer Center for modification, repair, upgrade or whatever, the technician always cut the 5 volt source to this pin, and grounded the pin instead.

The Model I also had only upper case symbols in its screen display, as RS did not anticipate anyone doing word processing on it. In order to save about 10 cents per machine, they omitted the static memory chip for bit 6 of the screen memory, and did a little peculiar logical arithmetic which finally sent an address into the character generator ROM and caused it to output an upper case character whether you typed upper or lower case into it. This bit of logic was the cause of some strange errors in BASIC. For example, if you accidentally input a lower case version of some letters, the resulting internal representation was not per the rules of BASIC and would cause SYNTAX ERROR to appear; yet the line looked OK! As I remember, the "@" sign was one of the trouble makers.

Several solutions were developed for the problem of the missing lower case. Remember, the character generator contained the lower case character set, it just couldn't be accessed by the original firmware. The most popular fix was to add a 1 bit X 1024 static memory chip (a 2102) piggy backed on top of one of the seven original static memory chips. One trace cut and two jumpers were required to complete this fix. This fix became known generically as the "Electric Pencil" fix, because Electric Pencil users needed the lower case capability, and they showed in their manual how the customer could do it.

Unfortunately this fix gave rise to the notorious "flying a". For some reason, the character generator ROMs used by Radio Shack had the lower case "a" stored in such a way that the "a" was one or two scan lines too high on the screen. This did not affect the printed output, of course, but it sure looked funny on the screen until you got used to it! One of the apochryphal stories about this "feature" was that supposedly some other customer ordered a large batch of character generator ROM's and the ROM manufacturer located the "a" incorrectly. Radio Shack then was alleged to have bought the batch of defective ROM's "real cheap" and worked out the upper case only routine so the ROM's were usable. Who knows the truth?

However, simply adding a static memory chip revealed what else Radio shack had been doing to the firmware. With the additional memory chip installed, the firmware now called for control characters from the character generator and the machine now produced gibberish at boot time, and required a software routine to produce valid letters.

This lower case fix of adding a static RAM chip became the de facto standard, and the designers of after-market operating systems all included some logic to detect whether the machine had the extra RAM chip or not, and would handle screen displays correctly. Except Radio Shack, who had to do the fix differently! They did something else to the circuitry, I don't remember what, which made their software a little different than the aftermarket software. The aftermarket people then had to incorporate another wrinkle to detect the Radio Shack fix and apply the necessary logic to it.

Yet another lower case fix was offered by The Electronic Closet, 8187 N.E. Blakely Court W., Bainbridge Island, Washington, 98110; they are still in business and offer a substitute ROM to go with the added static memory chip. Their character generator has an extra set of lower case letters in it, one where ASCII code expects it, and another which replaces the control characters! With this setup, you can have lower case in cassette operation and all disk software works properly also. The only problem is you can't have control characters on the screen even if you try. There aren't any! They also offer other features, including custom ROM's and a selectable alternate character set, such as Greek or French or whatever.

Another problem with the Model I was the lack of a CONTROL key. Electric Pencil by Michael Schrayer (which was ported over from the CP/M world) needed a CONTROL key. The same page in the manual which showed how to add the memory chip for lower case also showed how to add the missing control key. The keyboard matrix allowed for the key, it simply wasn't there. The fix was to install any available momentary contact push switch somewhere on the keyboard and attach it electrically to the empty position in the matrix. You should have seen the modifications which sprouted on the Model I's owned by the more adventurous! This particular fix was obviated when Michael Schrayer rewrote Electric pencil to use the CLEAR key as a CONTROL key. This also became a de facto standard. Any programmer who needed a CONTROL key and who did not need the CLEAR key simply programmed the CLEAR key to be interpreted as the missing CONTROL key.

The Model I was notorious for bouncing key contacts. The actual keys on the keyboard actuate tiny switch contacts underneath the keycaps. If the tiny contacts bounce excessively, the keyboard scanner software may interpret this as multiple presses of the key and put multiple images on the screen. Some keyboards were far worse than other. Three different fixes were available. One was to attempt cleaning and adjustment of the contacts. In the original keyboard, the keycap could be pulled straight up and off, exposing the contacts to the tender ministrations of the owner or his friend. Sometimes cleaning and/or bending of the contacts improved matters, sometimes things got worse. Very iffy. The second fix was to modify the scanner software so that it would wait longer to decide if the key was closed or not. RS offered a software fix for the Level I Model I in the form of the KBFIX tape cassette program. This loaded in and replaced the scanner in ROM and did help the situation. I don't remember what they did

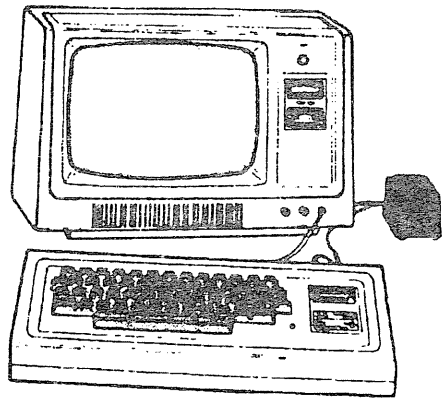
for Level II. Disk systems included the improved scanner as part of the DOS. The real fix came later as a new keyboard, made by ALPS and known to us by that name. The keys were shaped differently, which allowed easy external identification, and the keycaps were no longer removable. The ALPS keyswitch was quite different internally and was a vast improvement over the original. The Model III and later machines got the ALPS keyboards from the beginning.

Unreliable cassette operation was a problem with the original version of the Model I. The machine appeared to be critically sensitive to the playback level of the cassette when loading files.

One of the aftermarket devices offered was the Data Dubber, about a $50 item which was supposed to clean up the waveforms and allow easier loading of cassette files. I know they sold a lot of these, but I don't know how successful they were in improving loading. I am sure they were on the wrong design track in tackling the problem, as the real problem was a timing problem in the ROM.

After a year or two of complaints, RS announced the solution to the problem and a fix. The problem was a bad assumption made when the ROM was designed. The fix compensated for (but did not cure) the bad design and was designated the XRX board. It was a small board, about 2" x 2" with a couple of chips on it, a few wires to be soldered to the keyboard, and a couple of trace cuts. I had it added to my machine, and the improvement was magical. From a delicate adjustment of the volume control, operation went to almost no concern at all about the volume setting. I just left it at midscale thereafter and never had to touch it again. Radio shack would install this fix for free if you complained loudly enough. This was fixed in software in some of the later Model I ROM's, those that showed "MEM SIZE" on the screen at boot up, as opposed to "MEMORY SIZE".

Originally RS supplied a CTR-41 cassette recorder with the Model I. Later they changed to a CTR-80 machine. And then the troubles began! Owners besieged RS with complaints that the recorder was damaging prerecorded tapes. RS refused to believe this was possible. Finally the truth came out. If the recorder's STOP button was actuated in the middle of a playback sequence, the signals on the tape would be garbled. But how was this possible? When the STOP button was actuated, the motor current was interrupted, and an inductive kick was generated which somehow fed back to the record/playback head which put a glitch on the tape. But why didn't the CTR-41 do the same thing? And what could be done to cure the problem? I don't know all the answers here, but evidently the internal circuitry of the CTR-41 was sufficiently different that the inductive kick of the motor did not reach the record head. As for the cure, I don't know what that was, either, but anyone who had the problem could take



his CTR-80 to RS, who would wave their magic wand over the internals and make the CTR-80 stop messing up the tapes.

The pink pearl treatment was another trick discovered to keep the Model I working adequately well. The original problem was penny pinching by Radio Shack. The edge card connectors of the keyboard and the expansion interface (EI) were originally tinned with lead/tin alloy solder instead of the more professional gold plating. This worked fine initially, but eventually and gradually the lead would oxidize, which caused the edge connector to become plated with lead oxide instead of solder. The lead oxide was a semiconductor, and would interact with the 5 volt signals to cause strange values to appear on the buses, with disastrous results. Usually, the machine would simply cease operating (freezeup or lockup), necessitating a RESET to regain control. Spontaneous reboots were also commonplace. Someone discovered the Pink Pearl pencil eraser would clean the coating of lead oxide off the edge connectors, and everything would work OK again for a month or two. This became a ritual with Model I owners. Then along came EAP Co of Keller Texas. This outfit discovered a source of male edge connectors which could be soldered over the deficient solder tinned edge connectors, and which in turn had gold plated fingers. With these connectors installed, the spontaneous reboots and freezeups went away. EAP is still in business today, if you need the connectors.

Art McAninch reminded me of some other ingenious uses of the pink pearl erasers. Since the density doubler board plugs into the bottom of the expansion interface, there is a tendency for the doubler to work loose and fall out of its socket. A suitably carved pink pearl makes a useful spacer to hold the doubler in place. The RS-232 board had a different problem, and some people found a pink pearl was helpful there, also.

Those who bought a Model I without an expansion interface (EI) had several limitations to contend with. One was limited memory. The keyboard had either 4K or 16K (Max) of RAM. The other 32K was only available if you added the EI. Secondly, the floppy controller and the RS-232-C interfaces were in the EI, so you were limited to cassette operation only, and no Serial port if you lacked

the EI. And finally, NO PRINTER. The printer interface was also in the EI. Of course when the cost of printers was $1000 and up, (the keyboard-only machine went for about that amount, also), most of us couldn't afford new printers. Therefore the lack of a printer port was just another minor complaint. But to their credit, RS did come out with a printer interface that plugged into the keyboard, so as printers came down in cost, the port (for about $100) could be made available. Of course the port consisted of only about 6 chips....

The original RS expansion interface was a problem. As long as you operated it only as a cassette system, it usually worked adequately. But when floppy disks came along, it shortcomings became evident. Apparently the root cause of trouble was bad signal timing on the bus brought about by poor design of the bus. RS originally blamed the trouble on "non-Radio Shack" memory chips installed by owners unwilling to pay their exorbitant price for chip sets.

When this explanation failed to explain all the observed troubles, they dug deeper. The next fix was a small additional cable between the keyboard and the expansion interface spliced by means of a small DIN connector. This too was only a patch, and the troubles continued.

The next fix was the addition of a special 40 line cable between the keyboard and the expansion interface, known as the "pregnant cable" because of the small black plastic enclosure in the middle of the cable which contained a couple of buffer chips. This fix had an unintended side effect. Since the pregnant cable required +5 volts, RS elected to change the function of a ground line between the pregnancy and the expansion interface. One of the ground wires was reassigned for use as a +5 volt wire. If, later on, an unmodified cable without the pregnancy was connected to such a modified expansion interface, it promptly blew the inaccessible fuse in the power supply serving the expansion interface!

The final fix to the bad expansion interface problem was a new expansion interface board design which got rid of the bad design features and which could be relied upon to work properly. Radio Shack did this, and even kept it physically compatible with the old board so it would fit in place of the previous bad board in the same case, thus avoiding the need for a new case.

Radio Shack strongly believed the customer should stay out of the internals of his computer, and so the Model I keyboard and the EI each had one of the enclosure screws covered up with a bit of sealing wax, or something similar. If you wanted warranty service and the wax was gone, you were out of luck. When they worked on it for you, they put a special sealing label over the screw head. I can remember waiting for my warranty to expire before I dared to cautiously open up my keyboard. That was in the beginning. Later, I seldom bothered to put the screws back in place; I just slapped the boards into the case and dropped the top cover on it, and that was that.

An interesting sidelight to this problem of the deficient EI was the creation of a new company known as LNW Inc. which was in the Orange County area of southern California. This company took note of the RS design problem, and resolved to design a new EI board of their own which would function properly. They succeeded and were a factor in the Model I arena for a couple of years. They built up several different Model I-like machines, the last of which was called the "Team" machine. I never knew where the name Team came from. Their machine had provision for color, alternate CPU's, and some other features. Unfortunately, they lacked sufficient capitalization, and they were effectively following RS, not leading. They finally folded.

Another quirk of the Model I was the floppy drives. The original drives were 35 track, single sided Shugart SA-400's at a shocking price. The floppy drive industry was just taking off at that time, and apparently originally thought 3 drives was aplenty for a machine. Radio Shack, to their credit, figured they should design for 4 drives.

Needing a 4th drive select line, they opted for unused line 32 in the ribbon cable. To simplify inventory stocking and to keep customer's icky fingers out of the drives, RS developed the "pulled teeth" concept for the ribbon cable which connected as a daisy chain to all 4 drives. The effect of this was that all four drives were addressed with all drive selects enabled. The 4 connectors on the cable were unique, each one having only one select finger intact, the other three being missing. Thus the position on the cable determined what the drive number would be, and the internal drive select jumpers were never altered.

Note that the concept of a universal drive which did not have to be internally addressed by the installer was such a good idea that IBM adopted it, with a variation, for their PC machines. In their scheme, all floppy drives are also addressed the same, and the curiously twisted floppy cable takes care of the actual addressing.

However, when Shugart (or someone else in the industry) decided a 4th drive select line was necessary, they decided the 4th select line should be line 6. Worse, double sided operation was being developed, and a side select line was needed. Another someone in the industry decided the logical one to use was line 32. But Radio Shack had already assigned line 32 as the 4th drive select line! Alas and Alack! What to do about the problem of four double sided drives?

Radio Shack came up with a solution. Ignore the problem and hope it will go away! It didn't. The outside DOS vendors decided the only way was to allow four SS drives, or three DS drives, and this required a new ribbon cable

with all teeth intact, no pulled pins. So junk your old ribbon cable and go buy a new one with 5 connectors on it with all teeth intact. $Ouch!$.

And then along came an unsung genius/hero who noted two things. First, the ribbon connectors were reversible; that is, they would plug onto the edge card connectors either way if the little key was removed (it was often missing, anyway). Second, all the odd numbered connectors were ground, and only the even numbered connectors had functions assigned. The genius part was to recognize that if ALL five connectors on the drive cable were turned 180 degrees, then the previous even lines (with some teeth missing) all became odd numbered ground connectors, and the previous odd lines (with all teeth present) became even numbered signal lines. Suddenly the missing teeth problem was moot, as no more than three were missing on any one connector and the remaining lines were ample for grounding purposes.

The remaining part of the solution was to revise the addressing on each drive so the drive would respond to only one address line, instead of all four, and the side select function would respond to address line 32. Now DS operation with 3 drives was feasible for the Model I.

The floppy disk controller (FDC) originally chosen by RS was not able to separate data pulses which were very close together. With 35 tracks, the system worked adequately; with 40 track drives, the high numbered tracks were sometimes unusable. PERCOM solved this by offering a data separator designed to work with the FDC and reliably sort the data out of the noise.

Double Density was a whole new ball of snakes. The original floppy disk controller was the WD-1771, designed only for single density (SD) operation. Since the Level II ROMs in the machine knew almost nothing about reading disks, and what they did know was limited to single density, the boot track (zero) had to be in single density, no options. Several designers of double density adapters considered the problem long and carefully. Their solution was to have TWO floppy disk controllers (FDC) in the machine, the original WD-1771 working in single density to boot the machine, and a second FDC able to operate in double density. A data separator was included to make the operation reliable. The DOSes then had the smarts to know when to enable one or the other of the two FDC's. But wouldn't you know, after PERCOM had established a workable technique which was acceptable to the aftermarket DOS authors, then RS had to do it differently (probably in a futile attempt to control this niche of the market) which muddied the water for everyone. The result was the aftermarket DOSes had to be revised to tell the two schemes apart and work properly with either one. Radio Shack wrote some new software to work only with their own scheme, but the whole issue soon became moot, due to the arrival of the Model III on the scene.

Roy Beck continues his recollections in the next issue. (Mar/Apr 1992)

HINTS & TIPS

WORKING BETWEEN THE DOS's

By Karl Mohr

I have found a unique way of transferring programs and files between the various Disk Operating Systems, that is without the use of one of the many Utility programs that perform this task. The first step is to dust off your NEWDOS system, set the PDRIVE for drive #1 to example 1 in the NEWDOS manual,

TI=A,TD=A,TC=35,SPT=10,TRS=3,GPL=2,
DDSL=17,DDGA=2

and format the disk in drive 1.

This is a single density, 35 track format. I have found that some Model 4 disk controllers can't do this format, and will need to have someone format the disk for them. To continue, the following DOS's can read or write to this format directly without changing any settings: Model 4 TRSDOS, Model 4 LS-DOS, DOSPLUS. LDOS, MULTI-DOS, Model 1 NEWDOS, the latter by setting the PDRIVE to this format.

There is one other command to be mentioned and I will explain with this example. Copy a file [/bas, or /txt] from a NEWDOS system in drive 0 to the formatted disk in drive 1, leave the disk in drive 1 and boot a TRSDOS 6 or LS-DOS 6 system. Do a 'CAT' on drive 1, then copy the file from drive 1 to the DOS system in drive 0. When doing a 'CAT' on drive 0, you will see that the new file has a question mark behind it., this is a 'file open' indication and can be closed by giving the command from DOS Ready: RESET FILESPEC [Filespec being the name of the file transferred]. The file will be closed and can now be used in the usual manner. When transferring files from Model 4 down to Model 3, the RESET function will not need to be used.

For those that use Dosplus 3.4, and wish to transfer files to TRSDOS 6, or LS-DOS 6, format a disk in drive 1 using the Dosplus 3.4 system, and transfer the files to the formatted disk. Note: This MUST be a DATA DISK only, with no operating system on it. TRSDOS 6, or LS-DOS 6 can read the formatted DOSPLUS 3.4 data disk and a RESET FILESPEC is not required to use the program on the Model 4 Dos.

Lastly, I have not mentioned TRSDOS 1.3, and for good reason. This Dos is not compatible with any other Dos; as a matter of fact, I have not been able to this date get the NEWDOS system to read a TRSDOS 1.3 disk. According to the NEWDOS manual, this is possible, but if anybody has had any luck, I would like to hear from them, I haven't.

Editor's note: According to the manual, NEWDOS/80 has a PDRIVE setting that will allow copying files to and from TRSDOS 1.3. It is not possible to view a TRSDOS directory, nor is it possible to execute a program from a TRSDOS 1.3 disk while in NEWDOS/80.

NEWDOS EDTASM REFERENCE GUIDE

By Lance Wolstrup

Since we recently featured an article by M. C. Matthews, explaining how to patch the Model III Newdos EDTASM to work with TRSDOS/LS-DOS 6.x in Model 4 mode (TRSTimes 4.5 - Sep/Oct 1991), it seems only reasonable that we should publish a 'quick reference guide' so the readers can get to work using the program.

Function	Format	
ASSEMBLE	**A(/sw(/sw...))**	
Switches		
	LP	-Line Printer
	NL	-No Listing
	NO	-No Output
	NS	-No Symbol table
	WE	-Wait in Errors
BYE	**B**	Exit to DOS
DELETE	**D(line ref or range)***	
EDIT	**E(line ref)**	
Subcommands		
	L	-List working copy of line
	(n)<SPACE>	-advance (n) space(s)
	(n)Sc	-advance to (nth) "c"
	(n)D	-Delete next (n) characyer(s)
	(n)Kc	-Kill up to (nth) "c"
	(n)Cc(cc...)	-Change next (n) chr(s)
	A	-Abort changes and restart
	I	-Insert chrs: exit by ENTER or S-UP
	X	-eXtend line (enters Insert mode)
	H	-Hack of remainder (enters I mode)
	E or <ENTER>	-exit; save changes
	Q or <BREAK>	-exit and abort changes
FIND	**F(string)**	
HARD COPY	**H(line ref or range)***	

INSERT	**I(LINE REF,(INCREMENT))**
LOAD FILE	**L D=filespec**
NUMBER	**N(line ref,(increment))**
PRINT	**P(line ref or range)***
TEXT OUTPUT	**T**
WRITE FILE	**W D=filespec**

* Line references:

nnnnn -- actual line number
-- First line of text
* -- Last line of text
. -- Current line
: -- From:To separator for range ref

NEWDOS/80 TIPS

By Bruce McDowell

The 'HIMEM' command of NEWDOS/80 v2 can be a little confusing - at least it was for me. The documentation led me to believe that HIGH MEMORY' could be set using a 'CMD"HIMEM", (decimal address). I later found out that the string storage area was still going to the top of 48K, even though I had set HIMEM at 34000.

The only way to set HIGH MEMORY from BASIC is to place a POKE at the beginning of the program - before any string or variable definitions are made. Of course, the usual reason for setting the memory would be to protect a machine language routine in upper memory.

To calculate the POKE values needed, find the beginning address of the routine. Divide this number by 256. The Most Significant Byte (MSB) of the address will be the integer value of your answer. Now, multiply this number by 256 and subtract the answer from the routine address; this will be the Least Significant Byte (LSB).

On a Mod I or III:

```
POKE 16561,LSB:POKE 16562,MSB:CLEAR 50
```

This will reset all pointers, and set memory to address calculated. The CLEAR 50 can be whatever value you need for your program. It is not necessary - and it is wasteful to clear more memory than needed. But, there is always an exception to every rule. String garbage collection will be done at fewer intervals when the maximum amount of memory has been cleared.

The <BREAK> key may be disabled without the customary 'CMD"BREAK,N" used in NEWDOS/80. If this method is used, a call to DOS is made - which runs to drive :0 and takes entirely too long. An easy way to get around that is to:

```
POKE 17170,195 = BREAK OFF - MOD I
POKE 17170,201 = BREAK ON - MOD I

POKE 17528,195 = BREAK OFF - MOD III
POKE 17528,201 = BREAK ON - MOD III
```

USEFUL PEEKS AND POKES FOR MODEL I & III

from the TRSTimes Vault

In order to poke a number into an address above 32767, you must subtract 65536 from the address desired. This will, of course, give you a negative number. Keep this in mind when attempting any of the poke combinations listed below.

Model I & III Memory Locations

The ASCII of INKEY$ is stored at:	16537
The lower limit for string space is at:	16544 (LSB) 16545 (MSB)
The upper limit for string space is at:	16561 (LSB) 16562 (MSB)
Start of free memory is at:	16637 (LSB) 16638 (MSB)
The data counter is found at:	16639 (LSB) 16640 (MSB)
The beginning of program pointer is at:	16548 (LSB) 16549 (MSB)
The end of program pointer is at:	16333 (LSB) 16334 (MSB)
The program line counter is at:	16546 (LSB) 16547 (MSB)
The present byte counter is at:	16600 (LSB) 16601 (MSB)

Peek and Pokes:

POKE 16420,1 (III)
Changes to the alternate character set, containing the hearts, spades, clubs, and diamonds.

POKE 16418,N (III)
This changes the cursor to the same thing as CHR$(N).

POKE 16409,1 (I/III)
On Model III and Model I's with lowercase, this causes all keyboard input to be in uppercase.

POKE 16409,0 (I/III)
On Model III and Model I's with lowercase, this causes all keyboard input to be in lower case.

POKE 16396,195 (I/III)
POKE 16397,154
POKE 16498,10
The three pokes together will disable the <BREAK> key on all versions of Mod I/III (level II and disk basic included!)

POKE 16396,201 (I/III)
Re-enable the <BREAK> key (see above).

PEEK (14337-14464) (I/III)
These are the keyboard addresses! PEEKing on of these locations will determine if the following keys are being pressed.

MEM	1	2	4	8	16	32	64	128
14337=	'	A	B	C	D	E	F	G
14338=	H	I	J	K	L	M	N	O
14340=	P	Q	R	S	T	U	V	W
14344=	X	Y	Z					
14352=	0	1	2	3	4	5	6	7
14368=	8	9	:	;	,	-	.	/
14400=	ENT	CLR	BRK	UP	DN	LT	RT	SPC
14464=	SHF	SHF	CTL	CAP	F1	F2	F3	

(UP = up arrow, DN = down arrow, LT = left arrow, and RT = right arrow)
Model I is mapped for decimal 1 in position 14464 for either shift key. In the Model III and 4, the location 14464 would be decimal 1 for the left shift key and decimal 2 for the right shift key.

For example, if PEEK(14340) = 16, then the "T" key is being pressed. If PEEK(14400) = 1, then the <ENTER> key is being pressed. This does not use the INKEY$ routine, and therefore you can hold a key down instead of hitting it repeatedly (useful for action games). Also if two keys that have the same keyboard addresses are pressed at the same time, the value will be the sum of the two keys.
EX: if <ENT> and <CLR> are pressed at the same time, location 14400 will equal 3.

N1 = PEEK(16406) (I)
N2 = PEEK(16407)
POKE 16406,154
POKE 16407,10
These four steps disable the keyboard.

POKE 16406,N1
POKE 16407,N2
Re-enable the keyboard (see above).

POKE 164124,141 (I/III)
POKE 16415,5
Send all video "PRINTs" to line printer.

POKE 16414,88
POKE 16415,4
Restores video "PRINTs" (see above).

POKE 16422,88 (I/III)
POKE 16423,4
Sends all "LPRINTs" to video.

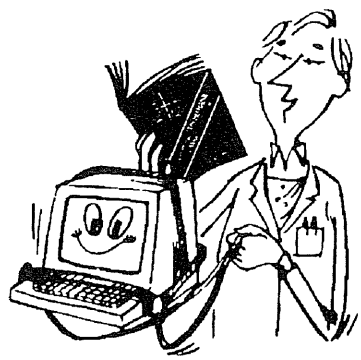
POKE 16422,141
POKE 16423,5
Restores "LPRINTs" (see above).

POKE 16804,103 (I/III)
POKE 16805,32
Changes command "LINE" to function like "LPRINT".

PEEK(16424) (I/III)
Number of lines per page +1 on line printer.

PEEK(16425) (I/III)
Number of lines printed.

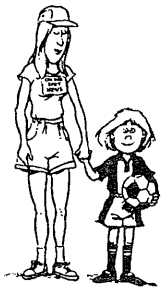
OUT 255,1 (I/III)
OUT 255,2
OUT 255,3
OUT 255,4
Sound for the computer. Hook up a speaker/amplifier to the aux plug.



WHO'S IN FIRST

Model 4 - Basic

By Lance Wolstrup



In a previous article I described how I got suckered into becoming the secretary for my son's little-league. The article presented one of the programs I wrote to schedule and maintain the different divisions in the league. I guess I don't learn from history, because it happened again.

This past fall, my youngest son, Steven, decided he wanted to play soccer. I dutifully registered him in the local little-league, paid the money, and was just about to leave when a lady came up to me and said hello. I turned out that she was one of my ex-students and she was now serving as the soccer league-secretary. "Better you than me", I said under my breath, thinking back a few years. "I'm so glad you're here", she said, "you're just the one I've been looking for!" I was just about to blurt out that my wife would probably have some objections, when she continued, "I am trying to write a program that will print out the standings of the divisions."

I told her that I would help, and a couple of evenings later we met at the league office. She had told me that "our computer is not like the ones we had at school", so I was prepared for an old Apple II, or possibly a CP/M machine. When she opened the door, I was surprised. It was a dandy-Tandy Model 4P. Hey, this was going to be fun!

When she said she needed help, she wasn't kidding. She had written about fifty lines of code that wasn't going to go anywhere. To make another long story short, I told her that I would take care of it - and have the program ready for her by Saturday morning before the first game of the season (this was Thursday night).

I went home and coded until the job got done (about 3 AM). The fruits of my nocturnal labor is shown below in listings 1 and 2.

Listing 1 is called LEAGMAKE/BAS. It is a short program that will create the datafile for a given division. As the league operated with 17 divisions in several age groups, I felt it was best to write a straight-forward program that could be easily modified to handle every division. Line 10 and line 90 should be changed to reflect the individual divisions.

Line 90 is the DATA with the names of the teams in a given division. Change these to the actual names of the teams. Then, in line 10, change variable TN to the actual number of teams in the division (must match the number of data items in line 90); change FI$ to the name of the data file and, finally, change LN$ to the name of the division. Note: the name of the data file (FI$) should be a short version of the division name. Make sure that it has the /DAT extension. Run LEAGMAKE/BAS once (after the changes have been made) for each division.

Listing 1.

```
1 'LEAGMAKE/BAS
2 'creates data files for LEAGMGR/BAS
3 '(c) copyright 1991 by Lance Wolstrup
4 ' all rights reserved
5 '
10 TN=10:FI$="DIV1/DAT:1":LN$="Division 1"
20 OPEN"O",1,FI$
30 PRINT#1,TN:PRINT#1,LN$
40 FOR X=1 TO TN
50 READ T$:PRINT#1,T$
60 FOR Y=1 TO 7:PRINT#1,0:NEXT
70 NEXT
80 CLOSE
90 DATA Kickers,Sockers,Rowdies,Cosmos,Tigers,
Bruins,Stompers,Redshirts,Kings,Dynamites
```

Now we come to the main program. Listing 2 is for LEAGMGR/BAS, which is also fairly straight-forward code. The program is error-trapped to be user-friendly, and it has a couple of neat features.

LEAGMGR/BAS prompts to make sure that you have inserted the disk with the league data files in drive :1. When you press <ENTER>, the data files are displayed on the screen. You are then prompted to type the name of the league data file you wish to work with. After a few seconds the current standings of the division are displayed on the screen. The first time you do this, of course, the standings are meaningless, as no games have been entered yet.

Each team is numbered according to its position in the standings, and you are prompted to type this number to select the 'home' team. You may type 'Q' if you wish to quit, otherwise type the number of the team. Keep in mind that if you quit the program here, any scores you may have entered during this session will NOT be saved to the data file. You are next prompted to type the number of the 'away' team.

When the two teams have been selected, you are prompted to type the number of goals that the home team scored, followed by a prompt to type in the number of goals scored by the away team.

You are then asked if this is correct. Answering 'N' will send you back to the home team selection prompt. Answering 'Y' processes the game, and the current standings are updated and displayed on the screen. Please note that the team 'standings' numbers have now most likely changed. You may also type 'Q' to this prompt. This will quit the program immediately and NOT save any work you may already have done.

You are now prompted in you would like to enter additional scores of games. You have three choices, 'Y', 'N', or 'P'.

Typing 'Y' brings you back to the 'home team prompt', starting the process over for a new game.

Typing 'N' saves the work you have done during this session to the data file and ends the program. This is the ONLY way to exit the program with the new data intact.

Pressing 'P' will print the standings on your printer, and then return to the 'additional scores' prompt. Do make sure that your printer is ready before choosing this option.

If you decide that you are not happy with the work you have done, you may type 'Q' to the 'additional scores' prompt. This will exit the program immediately and NOT save your work to the data file.

This is all there is to is to it. Simply, select the teams and enter the scores. The standings are updated after each game. Continue until all scores have been entered and, at this point, type 'P' to get hard copy of the standings. End the session by answering 'N' to the 'additional scores' prompt to save your work and exit the program.

Lines 11 through 19 contain the routine that displays the directory of the individual /DAT files on the screen. You are prompted to type the name of the data file you wish to work with. Should an error occur, such as drive :1 not being ready, or a non-existent file selected, the routine returns to the prompt in line 11. If, on the other hand, all is well, the program moves on to line 100.

Lines 100 through 165 opens the selected data file and loads the various variables. TN stores the number of teams in the file; LN$ holds the name of the division. Line 120 makes sure that we will not get an array owerflow. Most leagues will play either a round robin (each team playing the other teams once), or a double round robin (each team playing the other teams twice). Variable MX figures the maximum array size for a double round robin. Line 130 then DIMensions the needed array variables to this maximum. Lines 140 through 160 loops to load the array variables. T$(X) is the name of the team. GP(X) is the number of games the team has played. GW(X) is the number of games the team has won. GT(X) is the number of ties for the team. GL(X) is the number of games the team has lost. GF(X) is the number of goals the teams has scored. GA(X) is the number of goals that has been scored against the team. P(X) is the number of points the team has accumulated. This league uses the standard way of awarding point; that is, 2 point for a win; 1 point for a tie, and 0 point for a loss. When all data has been transferred to the array variable, the file is closed in line 160.

Lines 170 through 210 contain the routine that displays the current standings of the chosen division. Line 210 jumps to the subroutine in line 20, where the screen is formatted and the standings displayed.

Returning from the subroutine, lines 220 through 290 handle the prompts and error trapping for the 'home' and 'away' team inputs.

Lines 300 through 410 handle the prompts and error trapping for the 'goal scored' routines.

Lines 420 through 450 contain the prompt and handling of the 'Is this correct' routine.

Lines 460 through 500 determine which team won, or if the game ended in a tie. The appropriate array variables are updated to reflect the outcome of the game.

Lines 520 through 600 update the standings by using two sort routines. The first routine (lines 520 through 550) sorts the teams according to the number of points they have accumulated. The next sort, following standard practice, sorts the teams with an equal amount of points by goal difference. Thus, you will get a true picture of the standings - not like what you see in the newspapers, where the local big-league team is always listed first if tied on points.

Lines 610 through 730 handle the prompt and program flow for the 'more scores' input. The data is saved in lines 680 through 730.

As a final note, let me say that the lady league-secretary was happy. She used the program for the entire fall season, making her work a lot easier.

Listing 2.

```
1 'DIV1/BAS
2 'a league standings program for Model 4
3 '(c) copyright 1991 by Lance Wolstrup
4 ' all rights reserved
5 '
10 PRINT CHR$(15):CLS:
PRINT@(0,33),"LEAGUE MANAGER":
PRINT@(1,28),"(c) 1991 Lance Wolstrup":
PRINT@(2,0),STRING$(80,140)
11 PRINT@(4,0),CHR$(31);
"Please insert division data-disk in drive :1 and press
<ENTER> ";CHR$(14);
12 I$=INKEY$:IF I$<>CHR$(13) THEN 12 ELSE
PRINT CHR$(15)
13 ON ERROR GOTO 18: PRINT@(4,0),CHR$(31);:
SYSTEM"cat /dat:1"
14 PRINT:PRINT:PRINT"Please select the division data
file: ";CHR$(14);:INPUT" ",FI$:PRINT CHR$(15)
16 IF FI$="" THEN 11
17 GOTO 100
18 RESUME 19
19 ON ERROR GOTO 0:GOTO 11
20 V=4:FOR X=1 TO TN
```

```
21 PRINT@(V,0),"";:PRINT USING"##";X;:
PRINT" - ";T$(X)
22 PRINT@(V,19),"";:PRINT USING"##";GP(X)
23 PRINT@(V,23),"";:PRINT USING"##";GW(X)
24 PRINT@(V,27),"";:PRINT USING"##";GT(X)
25 PRINT@(V,31),"";:PRINT USING"##";GL(X)
26 PRINT@(V,35),"";:PRINT USING"###";GF(X)
27 PRINT@(V,40),"";:PRINT USING"###";GA(X)
28 PRINT@(V,47),"";:PRINT USING"##";P(X)
29 V=V+1:NEXT:RETURN
30 LPRINT:LPRINT TAB(1);LN$:
LPRINT STRING$(70,45):LPRINT:LPRINT:
FOR X=1 TO TN
31 LPRINT USING"##";X;:LPRINT" - ";T$(X);
32 LPRINT TAB(19) USING"##";GP(X);
33 LPRINT TAB(23) USING"##";GW(X);
34 LPRINT TAB(27) USING"##";GT(X);
35 LPRINT TAB(31) USING"##";GL(X);
36 LPRINT TAB(35) USING"###";GF(X);
37 LPRINT TAB(40) USING"###";GA(X);
38 LPRINT TAB(47) USING"##";P(X)
39 NEXT:RETURN
100 OPEN"i",1,FI$
105 ON ERROR GOTO 0
110 INPUT#1,TN:INPUT#1,LN$
120 MX=(TN-1)*2
130 DIM T$(MX),GP(MX),GW(MX),GT(MX),GL(MX),
GF(MX),GA(MX),P(MX)
140 FOR X=1 TO TN
150 INPUT#1,T$(X):INPUT#1,GP(X),GW(X),GT(X),
GL(X),GF(X),GA(X),P(X)
160 NEXT:CLOSE
170 PRINT CHR$(15):CLS
180 H=INT((80-LEN(LN$))/2)
190 PRINT@(0,H),LN$:PRINT@(1,0),STRING$(80,140)
200 PRINT@(3,19),"GP GW GT GL GF GA P"
210 GOSUB 20
220 PRINT@(16,0),CHR$(31);"Type # or hometeam: ";
CHR$(14);:INPUT" ",HT$:PRINT CHR$(15)
230 IF HT$="Q" OR HT$="q" THEN 730
240 HT=VAL(HT$)
250 IF HT<>INT(HT) OR HT<1 OR HT>TN THEN 220
260 PRINT@(16,0),CHR$(31);T$(HT)
270 PRINT@(18,0),CHR$(31);"Type # of away team: ";
CHR$(14);:INPUT" ",AT$:PRINT CHR$(15)
280 AT=VAL(AT$):IF AT=HT THEN 270
285 IF AT<>INT(AT) OR AT<1 OR AT>TN THEN 270
290 PRINT@(18,0),CHR$(31):PRINT@(16,11),"- ";T$(AT)
300 PRINT@(18,0),CHR$(31);"Goals for ";
T$(HT);CHR$(14);" ";:INPUT" ",HS$:PRINT CHR$(15)
310 HS=VAL(HS$)
320 IF HS<>INT(HS) THEN 300
330 IF HS$="0" THEN 350
340 IF HS<1 OR HS>99 THEN 300
350 PRINT@(16,23),"";:PRINT USING"##";HS:
PRINT@(16,26),"-"
360 PRINT@(18,0),CHR$(31);"Goals for ";T$(AT);" ";
CHR$(14);:INPUT" ",AW$:PRINT CHR$(15)
370 AW=VAL(AW$)
380 IF AW<>INT(AW) THEN 360
390 IF AW$="0" THEN 410
400 IF AW<1 OR AW>99 THEN 360
410 PRINT@(16,28),"";:PRINT USING"##";AW
420 PRINT@(18,0),CHR$(31);"Is this correct (Y/N) ";
CHR$(14);:INPUT" ",I$:PRINT CHR$(15)
430 IF I$="Y" OR I$="y" THEN 460
440 IF I$="N" OR I$="n" THEN 220
450 IF I$="Q" OR I$="q" THEN 730
460 IF HS>AW THEN P(HT)=P(HT)+2:
GW(HT)=GW(HT)+1:GL(AT)=GL(AT)+1:GOTO 490
470 IF HS<AW THEN P(AT)=P(AT)+2:
GW(AT)=GW(AT)+1:GL(HT)=GL(HT)+1:GOTO 490
480 P(HT)=P(HT)+1:GT(HT)=GT(HT)+1:
P(AT)=P(AT)+1:GT(AT)=GT(AT)+1
490 GP(HT)=GP(HT)+1:GP(AT)=GP(AT)+1
500 GF(HT)=GF(HT)+HS:
GA(HT)=GA(HT)+AW:GF(AT)=GF(AT)+AW:
GA(AT)=GA(AT)+HS
510 'sort on points
520 FL=0:FOR X=1 TO TN-1
530 IF P(X)<P(X+1) THEN SWAP T$(X),T$(X+1):
SWAP GP(X),GP(X+1):SWAP GW(X),GW(X+1):
SWAP GT(X),GT(X+1):SWAP GL(X),GL(X+1):
SWAP GF(X),GF(X+1):SWAP GA(X),GA(X+1):
SWAP P(X),P(X+1):FL=1
540 NEXT
550 IF FL=1 THEN 520
560 'sort on goal difference
570 FL=0:FOR X=1 TO TN-1
580 IF P(X)=P(X+1) THEN H(X)=GF(X)-GA(X):
H(X+1)=GF(X+1)-GA(X+1):
IF H(X+1)>H(X) THEN SWAP T$(X),T$(X+1):
SWAP GP(X),GP(X+1):SWAP GW(X),GW(X+1):
SWAP GT(X),GT(X+1):SWAP GL(X),GL(X+1):
SWAP GF(X),GF(X+1):SWAP GA(X),GA(X+1):
SWAP P(X),P(X+1):FL=1
590 NEXT
600 IF FL=1 THEN 570
610 PRINT@(4,0),CHR$(31)
620 GOSUB 20
630 PRINT@(16,0),CHR$(31);
"Enter more scores (Y/N/P) ";CHR$(14);:INPUT" ",I$
640 IF I$="Y" OR I$="y" THEN 220
650 IF I$="Q" OR I$="q" THEN 730
660 IF I$="N" OR I$="n" THEN 680
665 IF I$="P" OR I$="p" THEN GOSUB 30
670 GOTO 630
680 OPEN"O",1,FI$
690 PRINT#1,TN:PRINT#1,LN$
700 FOR X=1 TO TN
710 PRINT#1,T$(X):PRINT#1,GP(X),GW(X),GT(X),
GL(X),GF(X),GA(X),P(X)
720 NEXT
730 CLOSE:CLS
```

GRAPHICS 90, FOR THE '90's

by Allen W. Jacobs

A Review of the Model III graphics editor by:

Adaptations of the original TRSDOS 1.3 package to work with NEWDOS, DOSPLUS, and MULTIDOS, Allwrite printer drivers for the HP Laserjet and HP Deskjet printers, and Supplementary Documentation by:
Gary W. Shanafelt, Dept. of History
Mc Murry University, Abilene, TX 79697
October, 1991

You may wonder why you are first reading about a brilliant graphics editor that was first written, created, and copyrighted back in 1983. We can only thank Gary W. Shanafelt for helping to revive and enhance it as he has. The specific reasons for the package to appear at this time, for only the cost of the media and shipping, are well explained in his additional documentation included with the package. Just consider it another innocent victim of the march of progress, because that's what it was.

The instant a graphics system is even conceived of, it is already subject to a number of limitations. Thus, there are limits to every system that exists in the "real world". Among these limitations are resolution, drawing speed, color and shading dynamics (better known as the "palette"), graphics memory, image storage capacity and retrieval speed, image storage format, etc. We all know this.

It is also certainly impossible to discount the amazing advances in computer graphics that have occurred in recent years. We must realize, however, that in the future, today's graphics will look as quaint to us then, as TRS-80 low resolution graphics look to "work station" owners, today.

Do you remember your first look at a Pong game in an arcade, wishing you could have one in your home? I do. However, when I finally bought a TRS-80 graphics system, that surpassed Pong games, I was still somewhat frustrated. That's because there was no practical way to program low-rez TRS-80 graphics to run at assembly language speeds without creating my own entire graphics development system.

By the time there was any good drawing program on the market, there were so many games available that I didn't really have the intense need to purchase ANY lo-rez graphics development system. This is especially true for an animation system, when by then, I had an unused high-rez board. I imagine that my desires in this area roughly paralleled those of the general computer buying public of that day. That is probably why Graphics 90 was never marketed.

Since every graphics standard is the product of its own moment in time, we must judge any software written for the standard on the timeless qualities of elegance, originality, and utility, within that given standard. Graphics 90 certainly possesses these qualities. It is one of the few graphics programs that takes full advantage of the tradeoff possible when one is given a system that has low graphic resolution and is therefore inherently capable of high, full screen graphics speed.

Namely, any graphics system is limited, within its design, by the amount of graphic data it can move within a given unit of time. Usually, this is termed the rate at which the system draws an image. It is often expressed as the number of pixels per second the system is capable of drawing. There is also the inherent limitation of the amount of graphic data required to form a given image on the system.

What all this means is: If you have a lot of pixels to draw on a high resolution screen, it is going to take a long time to draw them. If you have very few pixels, the image can be drawn relatively rapidly.

If there are no color or intensity attributes, very little data is required to describe each pixel. For comparison, one byte on a Model III lo-rez system can describe 6 pixels. With data compression, it can represent even more.

Some high end graphics systems today require between 3 to 4 bytes (24-32 bits) just to describe ONE pixel. So, if you want to animate a large, high resolution graphics image, you need to have special hardware, usually separate from the CPU, capable of drawing a lot of pixels very rapidly. However, low resolution graphics, such as the Model III's, can be made to rapidly move while requiring a reasonably modest amount of computing power.

Except in games animation, motion is used far too infrequently because it is difficult to program the kinds of animated sequences found in such games. This is regrettable because, in some contexts and applications, animation can add interest and information delivering power to a typical lo-rez image. There is no better way to express movement or change in an image than to move it or change it.

Graphics 90 is the best animation system for the Model III low-rez screen I have ever seen. There are two reasons for this. The program takes advantage of the ability of the TRS-80 to completely redraw the low-rez screen in LESS time than it takes to display the resulting image (ie. it can do it between scans). Thus, animation on the screen

progresses seamlessly. The TRS-80 can only perform this feat, however, if it is executing graphics instructions at "assembly language speed". Basic is too slow for this purpose. Yet, assembly language is normally too arduous for most of us to use to achieve this standard.

Thus, the Graphics 90 basic language enhancements do the pixel moving tasks at top speed from within ABASIC (which stands for Animation BASIC). Basic is simply loaded and patched after the ABASIC enhancement routines are loaded. They are accessed through additional basic commands, which are explained in the manual that accompanies the disks.

The second of the dual wonders in this package is the editor. It is used to both create and display moving, low-rez graphics images. It combines ease of use with automated drawing and animation capabilities. It is not exactly a typical "draw" program, nor is it a "cad" editor. What it does is handle drawn shapes in a cad-like manner.

That means that graphics can be created one frame at a time. But, there is an additional array of tools designed to work, with motion in mind, on any graphic object of your choice. Anything from a single pixel to a complex image encompassing the entire screen can be drawn and moved. There are routines to draw lines, circles, boxes, and ellipses of your specification. There are routines to connect lines between points. You can erase, move, and copy objects. You can invert, rotate, or do both to any part of the image or the entire screen. You can complement objects or areas by reversing the "set" of all the pixels in a designated block.

"That is not so sophisticated that I am willing to send away for the media, duplication, and mailing costs for a low-rez graphics editor", you say. If that was all there is to it, then you might be right. But, if you think about it, that gives you the capability of moving up, down, right, and left, from frame to frame, producing the illusion of motion if the frames are viewed successively. "Some or all of those things can be done with most draw programs", you say. True.

However, you can box an item in a drawing and semi-automatically move it IN and OUT by re-sizing every feature in the item, thus providing the illusion of movement in ALL THREE dimensions. Since frames can be viewed in ANY order, ANY number of times, selected by you in Basic, it can give you the illusion of the object getting closer and farther. It does this by making the object correspondingly larger and smaller. You can even control scaling of the horizontal and vertical axis, individually. In Graphics-90, the technique is called the "rubber band box". It is more familiar to MS-DOS desktop publishers, however, in the context of sizing "scaleable type". Moreover, the rubber band box will apply that technique to ANY item within it, not just to type! The Graphics-90 rubber band box is a forerunner of this now widely touted scaling system. While the title is less pretentious, the technique is the same. "Now that is getting to be sophisticated", you say. Also, True.

To be sure, the higher the resolution of the image, the more inherently effective scalability becomes. But, we know that increased detail requires a higher resolution graphic standard. Proportionately, you can afford to lose more detail during re-scaling if you HAVE more detail to start with. That is the reason that type scaling on a nine pin dot matrix printer is not nearly as effective as on a 24 pin or a laser printer, unless you are working with large type. That is also why this software package never sold, which is regrettable.

The demonstration sequence, included in the package, provides a glimpse of the level of power the animation control in Graphics 90 gives you, even with low-rez graphics. It does what you can't do with type on a page. You can easily pan over portions of a high resolution scene, in low-rez, and see all the detail. It not only catches your eye, but it is also satisfying to know that with some practice you can produce the same quality of illustration you are looking at. A custom animated sequence you develop yourself will even impress MS-DOS users. This is because they can usually only show you shareware demonstrations of work that OTHERS have done and most of which doesn't move unless it's a game. Also, even with a '386 powerhouse, listen to how much their hard disk works to maintain the motion.

Color is very beneficial to any graphic image. Therefore, high-rez images obviously look best on the screen when they are in color. However, have you tried to print out one of those images with anything less than a Tektronics hot wax printer or, at least, a slower/lower resolution color bubblejet printer? Most people don't even own a color printer unless they either have a lot of money or a specific need for such a printer. Thus, most color images end up as monochrome images when they get printed.

If those images were created in color, that loss of color can sometimes be disastrous. If you don't believe this, look at a bunch of colored party balloons on a Super VGA screen and then print them out, even to the best laser printer. The detail of the balloons remains good. However, in monochrome, it no longer looks as though the party is exciting.

It is thus often easier to create graphics in the mode by which they will be viewed. Over the last few years, the buzzword to describe this has been WYSIWYG. That stands for "What You See Is What You Get". Thanks to Gary Shanafelt, Graphics 90 will now print on both HP Laserjet and HP Deskjet printers because he has provided the Allwrite printer drivers for them, in the package!

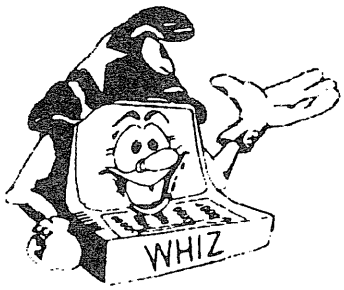
I hate to sound so salesman-like but the program, the well written and printed manual, the Model III DOS portability, the printer drivers that Gary Shanafelt has included, and the price will make you say, "This is an excellent deal!".

True.

GRAPHICS-90 FOR MODEL I

Newdos/80 v.2

By Mathieu Simons



In the process of examining the code to GRAPHIC-90, I discovered that the program can easily be patched to work on a Model I, using Newdos/80 v.2; all that needs to be done is to change the address of one ROM call, and the address of the HIMEM pointer.

The particular ROM call is $KBLINE (wait for a keyboard line); it is found at X'40' in the Model III, and it resides at X'05D9' on the Model I. The Model III HIMEM-pointer is located at X'4411', while the Model I positions it at X'4049'.

I used LSFED-II to trace these references in ABASIC and its related utilities without difficulty.

After establishing the addresses to be modified, I wrote a routine to make the individual program modules determine if they are running on a Model I or a Model III. Luckily, each of the program files had enough patch-space for this routine, which checks the contents of memory location X'0125'. The Model III stores 49H there (this is the third 'I' of Model III). If 49H is detected at X'0125', the program executes as normal. However, if 49H is not found in this location, the routine assumes that it is running on a Model I. The routine then proceeds to change the address of the HIMEM-pointer to X'4049', as well as all references to $KBLINE to X'05D9'. At this point the program will execute, running flawlessly on the Model I.

The traced addresses are:

ABASIC/CMD	X'5202', X'52D6', X'52EF', X'55E3' (HIMEM X'4411')
ARUN/CMD	X'5222' (HIMEM)
EDIT/CMD	X'54FC' (HIMEM) X'59F2', X'5B46', X'6DFB' (ROM call X'40')
SCREEN/CMD	X'5202', X'52B7', X'52C0', X'537A', X'5385' (HIMEM)
GENERATE/CMD	X'5238', X'5275', X'546A', X'54A5' (ROM calls only)
COMPILE/CMD	NONE

The patches, listed below, are in a format that can be used with Super Utility Plus or with SUPERZAP/CMD from Newdos/80.

The problem, of course, is transferring the files from the distribution disks over to Model I Newdos/80 compatible disks. If you have no other machine than a standard-issue Model I (single-sided, single-density, 35 track capability), you will have to rely on friends or aquaintances to transfer the files for you. As of this writing, the GRAPHICS-90 disks are distributed in double-density, 40 track format on Model III TRSDOS 1.3 or LDOS readable diskettes.

If, on the other hand, your Model I has been modified to read and write double-density, 40 track disks - and you have LDOS - you can use it to transfer the files to a single-density, 35 track data disk which NEWDOS/80 can then read. A Model III or 4 can also be used to transfer the files, as long as you have a DOS capable of writing single-density, 35 track disks.

NAME ABASICN/CMD
;make universal for model I/III

PATCH	ABASICN/CMD,1,49
CHANGE	2C 20 31 39 38 34,20
TO	2D 38 34 20 62 79 20
PATCH	ABASICN/CMD,1,50
CHANGE	62 79 20 4C 61 72 72 79 20 50 61 79 6E 65 0A 4D
TO	4C 61 72 72 79 20 50 61 79 6E 65 0A 4D 6F 64 73
PATCH	ABASICN/CMD,1,60
CHANGE	6F 64 73 20 66 6F 72 20 4E 45 57 44 4F 53 38 30
TO	20 4E 45 57 44 4F 53 38 30 20 32 2E 30 20 47 2E
PATCH	ABASICN/CMD,1,70
CHANGE	20 56 65 72 2E 20 32 20 62 79 20 47 61 72 79 20
TO	57 2E 53 68 61 6E 61 66 65 6C 74 0A 4D 6F 64 65
PATCH	ABASICN/CMD,1,80
CHANGE	57 2E 20
TO	6C 31 20

```
PATCH       ABASICN/CMD,1,83
CHANGE      53 68 61 6E 61 66 65 6C 74 0A 56 65 72
            73 69 6F
TO          4D 2E 53 69 6D 6F 6E 73 0A 56 2E 30 31
            2E 30 30

PATCH       ABASICN/CMD,1,93
CHANGE      6E 20 30
TO          2E 30 30

PATCH       ABASICN/CMD,1,96
CHANGE      31 2E 30 30 2E 30 30 0A 0D
TO          6D 6F 64 49 2F 49 49 49 0D

PATCH       ABASICN/CMD,1,FD
CHANGE      28 63 29
TO          E5 F5 3A

PATCH       ABASICN/CMD,4,00
CHANGE      20 31 39 38 34 20 4C 61 72 72 79 20 50
            61 79 6E
TO          25 01 FE 49 28 0F 21 49 40 22 02 52 22
            D6 52 22

PATCH       ABASICN/CMD,4,14
CHANGE      65 28 63 29 20 31 39 38 34 20
TO          EF 52 22 E3 55 F1 E1 C3 00 52

PATCH       ABASICN/CMD,4,72
CHANGE      00 52 20
TO          ED 55 20
```

NAME **ARUN/CMD**

; make universal for model I/III

```
PATCH       ARUN/CMD,2,2C
CHANGE      28 63 29 20 31 39 38 34 20 4C 61 72 72
            79 20 50
TO          E5 F5 3A 25 01 FE 49 28 06 21 49 40 22
            22 52 F1

PATCH       ARUN/CMD,2,3C
CHANGE      61 79 6E 65
TO          E1 C3 00 52

PATCH       ARUN/CMD,3,DB
CHANGE      00 52 00
TO          20 54 00
```

NAME **GENERATE/CMD**

```
PATCH       GENERATE/CMD,5,8C
CHANGE      28 63 29 20 31 39 38 34 20 4C 61 72 72
            79 20 50
TO          E5 F5 3A 25 01 FE 49 28 0F 21 D9 05 22
            38 52 22

PATCH       GENERATE/CMD,5,9C
CHANGE      61 79 6E 65 28 63 29 20 31 39 38 34 20
TO          75 52 22 6A 54 22 A5 54 F1 E1 C3 01 52

PATCH       GENERATE/CMD,6,4C
CHANGE      01
TO          74
```

NAME **EDIT/CMD**

; make universal for model I/III

```
PATCH       EDIT/CMD,33,D9
CHANGE      28 63 29 20 31 39 38 34 20 4C 61 72 72
            79 20 50
TO          E5 F5 3A 25 01 FE 49 28 12 21 49 40 22
            FC 54 21

PATCH       EDIT/CMD,33,E9
CHANGE      61 79 6E 65 28 63 29 20 31 39 38 34 20
            4C 61 72
TO          D9 05 22 F2 59 22 46 5B 22 FB 6D F1 E1
            C3 FA 54

PATCH       EDIT/CMD,35,53
CHANGE      FA 54
TO          51 73
```

NAME **SCREEN/CMD**

; make universal for model I/III

```
PATCH       SCREEN/CMD,1,B0
CHANGE      28 63 29 20 31 39 38 34 20 4C 61 72 72
            79 20 50
TO          E5 F5 3A 25 01 FE 49 28 12 21 49 40 22
            02 52 22

PATCH       SCREEN/CMD,1,C0
CHANGE      61 79 6E 65 28 63 29 20 31 39 38 34 20
            4C 61 72
TO          B7 52 22 C0 52 22 7A 53 22 85 53 F1 E1
            C3 00 52

PATCH       SCREEN/CMD,2,98
CHANGE      00 52
TO          A8 53
```

END

Mathieu Simon can be reached at:
Weg Naar Zwartberg 141
B3530 Houthalen
Belgium

ATTENTION TRSDOS 1.3. USERS!

ANNOUNCING "SYSTEM 1.5.", THE MOST COMPREHENSIVE 1.3. UPGRADE EVER OFFERED!

MORE SPEED!! MORE POWER!! MORE PUNCH!!

While maintaining 100% compatibility to TRSDOS 1.3., this DOS upgrade advances TRSDOS 1.3. into the 90's!
SYSTEM 1.5. supports 16k-32k bank data storage and 4MGHZ clock speed (4/4P/4D).
DOUBLE SIDED DRIVES ARE NOW 100% UTILIZED! (all models).

CONFIG=Y/N	CREATES CONFIG BOOT UP FILE	DATE=Y/N	DATE BOOT UP PROMPT ON or OFF
TIME=Y/N	TIME BOOT UP PROMPT ON or OFF	CURSOR='XX'	DEFINE BOOT UP CURSOR CHAR
BLINK=Y/N	SET CURSOR BOOT UP DEFAULT	CAPS=Y/N	SET KEY CAPS BOOT UP DEFAULT
LINE='XX'	SET *PR LINES BOOT UP DEFAULT	WP=d.Y/N (WP)	WRITE PROTECT ANY or ALL DRIVES
ALIVE=Y/N	GRAPHIC MONITOR ON or OFF	TRACE=Y/N	TURN SP MONITOR ON or OFF
TRON=Y/N	ADD an IMPROVED TRON	MEMORY=Y/N	BASIC FREE MEMORY DISPLAY MONITOR
TYPE=B/H/Y/N	HIGH/BANK TYPE AHEAD ON or OFF	FAST	4 MGHZ SPEED (MODEL 4'S)
SLOW	2 MGHZ SPEED (MODEL III'S)	BASIC2	ENTER ROM BASIC (NON-DISK)
CPY (parm,parm)	COPY/LIST/CAT LDOS TYPE DISKS	SYSRES=H/B/'XX'	MOVE/SYS OVERLAY(s) TO HI/BANK MEM
SYSRES=Y/N	DISABLE/ENABLE SYSRES OPTION	MACRO	DEFINE ANY KEY TO MACRO
SPOOL=H/B.SIZE	SPOOL is HIGH or BANK MEMORY	SPOOL=D.SIZE='XX'	LINK MEM SPOOLING TO DISK FILE
SPOOL=N	TEMPORARILY DISABLE SPOOLER	SPOOL=Y	REACTIVATE DISABLED SPOOLER
SPOOL=RESET	RESET (NIL) SPOOL BUFFER	SPOOL=OPEN	OPENS, REACTIVATES DISK SPOOLING
SPOOL=CLOSE	CLOSES SPOOL DISK FILE	FILTER *PR.ADLF=Y/N	ADD LINE FEEDS BEFORE PRINTING 0DH
FILTER *PR.IGLF	IGNORES 'EXTRA' LINE FEEDS	FILTER *PR.HARD=Y/N	SEND 0CH to PRINTER (FASTEST TOF)
FILTER *PR.FILTER	ADDS 256 BYTE PRINTER FILTER	FILTER *PR.ORIG	TRANSLATE PRINTER BYTE TO CHNG
FILTER *PR.FIND	TRANSLATE PRINTER BYTE TO CHNG	FILTER *PR.RESET	RESET PRINTER FILTER TABLE
FILTER *PR.LINES	DEFINE NUMBER LINES PER PAGE	FILTER *PR.WIDTH	DEFINE PRINTER LINE WIDTH
FILTER *PR.TMARG	ADDS TOP MARGIN to PRINTOUTS	FILTER *PR.BMARG	ADDS BOTTOM MARGIN to PRINTOUT
FILTER *PR.PAGE	NUMBER PAGES, SET PAGE NUMBER	FILTER *PR.ROUTE	SETS PRINTER ROUTING ON or OFF
FILTER *PR.TOF	MOVES PAPER TO TOP OF FORM	FILTER *PR.NEWPG	SET DCB LINE COUNT TO 1
FILTER *KI.ECHO	ECHO KEYS to the PRINTER	FILTER *KI.MACRO	TURN MACRO KEYS ON or OFF
ATTRIB:d.PASSWORD	CHANGE MASTER PASSWORD	DEVICE	DISPLAYS CURRENT CONFIG INFO

All parms above are installed using the new LIBRARY command SYSTEM (parm,parm). Other new LIB options include DBSIDE (enables double sided drive by treating the "other side" as a new independent drive, drives 0-7 supported) and SWAP (swap drive code table #s). Dump (CONFIG) all current high and/or bank memory data/routines and other current config to a disk data file. If your type ahead is active, you can (optional) store text in the type buffer, which is saved. During a boot, the config file is loaded back into high/bank memory and interrupts are recognized. After executing any active auto command, any stored type ahead data will be output. FANTASTIC! Convert your QWERTY keyboard to a DVORAK! Route printer output to the screen or your RS-232. Macro any key,even F1, F2 or F3. Load *01-*15 overlay(s) into high/bank memory for a memory only DOS! Enter data faster with the 256 byte type ahead option. Run 4MGHZ error free as clock, disk I/O routines are properly corrected! Spool printing to high/bank memory. Link spooling to disk (spooling updates DCB upon entering storage). Install up to 4 different debugging monitors. Print MS-DOS text files, ignoring those unwanted line feeds. Copy, Lprint, List or CATalog DOSPLUS, LS-DOS, LDOS or TRSDOS 6.x.x. files and disks. Add top/bottom margins and/or page numbers to your hard copy. Rename/Redate disks. Use special printer codes eg: LPRINT CHR$(3); toggles printer output to the ROUTE device. Special keyboard codes add even more versatility. This upgrade improves date file stamping MM/DD/YY instead of just MM/YY. Adds optional verify on/off formatting, enables users to examine *01-*15, DIR, and BOOT sectors using DEBUG, and corrects all known TRSDOS 1.3. DOS errors. Upgrade includes LIBDVR, a /CMD driver that enables LIBRARY commands, such as DIR, COPY, DEBUG, FREE, PURGE, or even small /CMD programs to be used within a running Basic program, without variable or data loss.

By special arrangement with GRL Software,
SYSTEM 1.5. is now distributed exclusively by TRSTimes magazine.

ORDER YOUR COPY TODAY!

Send $39.95 (U.S. funds) to:

TRSTimes - SYSTEM 1.5.

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

Model I & III
Public Domain Disks

PD#1: binclock/cmd, binclock/doc, checker/bas, checker/doc, chomper/bas, cls/cmd, dduty3/cmd, driver/cmd, driver/doc, drivtime/cmd, mazeswp/bas, minibase/bas, minitest/dat, mx/cmd, piazza/bas, spdup/cmd, spdwn/cmd, vici/bas, vid80/cmd, words/dic.

PD#2: creator/bas, editor/cmd, maze3d/cmd, miner/cmd, note/cmd, poker/bas, psycho/cmd, supdraw/cmd, vader/cmd

PD#3: d/cmd, trsvoice/cmd, xmodem/cmd, xt3/cmd, xt3/txt, xthelp/dat

PD#4: cobra/cmd, disklog/cmd, flight/bas, flight/doc, narzabur/bas, narzabur/dat, narzabur/his, narzabur/txt, othello/bas, vid80x24/cmd, vid80x24/txt

PD#5: eliza/cmd, lu31/cmd, sq31/cmd, usq31/cmd

PD#6: clawdos/cmd, clawdos/doc, cocoxf40/cmd, dskrnam/bas, menu/cmd, ripper3/bas, sky2/bas, sky2/his, space/cmd, stocks/bas, trs13pat/bas, vidsheet/bas

PD#7: cards/bas, cities/bas, coder/bas, eye/bas, heataudt/bas, hicalc/bas, life/bas, moustrap/bas, ohare/bas, slots/bas, stars/cmd, tapedit/bas

PD#8: craps/bas, fighter/bas, float/bas, hangman/bas, jewels/cmd, lifespan/bas, varidump/bas, xindex/bas, xor/bas

PD#9: bublsort/bas, chess/bas, finratio/bas, homebudg/bas, inflat/bas, mathdril/bas, midway/bas, nitefly/bas, pokrpete/bas, teaser/bas

PD#10: ltc21/bas, ltc21/ins, lynched/bas, match/bas, math/bas, message/bas, message/ins, portfol/bas, portfol/ins, spellegg/bas, storybld/bas

PD#11: alpha/bas, caterpil/cmd, cointoss/bas, crolon/bas, cube/cmd, dragon/cmd, fastgraf/bas, fastgraf/ins, lunarexp/bas, music/bas, music/ins, planets/bas, volcano/cmd

PD#12: baccarat/bas, backpack/bas, backpack/ins, doodle/bas, dragons/bas, dragons/ins, king/bas, sinewave/bas, snoopy/bas, wallst/bas, wallst/ins

PD#13: atomtabl/bas, boa/bas, chekbook/bas, conquer/cmd, dominos/bas, morse/bas, mountain/bas, quiz/bas, signbord/bas, sketcher/bas

PD#14: autoscan/bas, checkers/bas, craps/bas, ducks/bas, isleadv/bas, nim/bas, rtriangl/bas, sammy/cmd, typing/bas, wordpuzl/bas

PD#15: budget/bas, corp/bas, corp/ins, fourcolr/bas, fullback/bas, grapher/bas, illusion/bas, jukebox/bas, ledger/bas, maze/cmd, reactest/bas, shpspree/bas, states/bas, tapecntr/bas, tiar/bas, tiar/ins

PD#16: amchase/bas, constell/bas, filemastr/bas, foneword/bas, geometry/bas, heartalk/bas, hidnumbr/bas, lgame/bas, marvello/bas, powers/bas, scramble/bas, speed/bas, subs/bas

PD#17: conundrm/bas, eclipse/bas, esp/bas, esp/ins, hustle/bas, jacklant/bas, mindblow/bas, othello/bas, pleng/bas, rubik/bas, trend/bas, ufo/bas, veggies/bas

PD#18: backgam/bas, chess/cmd, cosmip/cmd, distance/bas, hexpawn/bas, music/cmd, stokpage/bas, texted/bas, texted/ins, trex/bas, twodates/bas, wanderer/bas

PD#19: banner/bas, cresta/cmd, lander/bas, medical/bas, moons/bas, par/bas, parchut/bas, pillbox/bas, readtrn/bas, replace/bas, ship/cmd, solomadv/bas, space/cmd, survival/bas

PD#20: bomber/bas, bumbee/cmd, ciaadv/bas, dice31/bas, dice31/ins, diskcat1/bas, firesafe/bas, flashcrd/bas, hitnmiss/bas, mazegen/bas, mazescap/cmd, roulette/bas, seasonal/bas

PD#21: aprfool/bas, catmouse/bas, d/cmd, escape/bas, header/bas, kalah/bas, mathwrld/bas, nameit/bas, note/cmd, photo/bas, read/cmd, syzygy/bas, timeshar/cmd, timeshar/doc, trace80/cmd, trsdir/cmd, worm/bas, yatz80/bas

PD#22: arcade/bas, cube/cmd, eclipse/bas, lcd/bas, leastsqr/bas, medical/bas, million/bas, pwrplant/bas, round/bas, subway/bas, tapeid/bas

PD#23: artil/bas, artil/ins, baseconv/bas, crushman/bas, dissert/bas, huntpeck/bas, jungle/bas, jungle/ins, messages/bas, monitor/bas, monster/bas, moons/bas, ohmlaw/bas, stockpage/bas, tictacto/bas

PD#24: baslist/asm, baslist/cmd, baslist/doc, cleaner3/cmd, cleaner3/doc, difkit1/bas, difkit1/doc, dirpatch/asm, dirpatch/cmd, e/cmd, ei/doc, i/cmd, newmap/bas, newmap/doc, varlst/asm, varlst/cmd, varlst/doc

PD#25: copy/bas, copy/doc, dirpw/asm, dirpw/cmd, dirpw/doc, dskfmt/bas, dskfmt/doc, himap/asm, himap/cmd, huricane/bas, hv/bas, hv/doc, keydemo/bas, keyin/bas, keyin/doc, lazyptch/asm, lazyptch/doc, salvage/bas, salvage/doc,wpflt/asm, wpflt/flt

PD#26: constell/bas, divisor/bas, frame/bas, heatfus/bas, heatfus/doc, hicalc/bas, mathlprt/bas, mathquiz/bas, molecule/bas, morscode/bas, phyalpha/bas, phyalpha/doc, remaindr/bas, usa/bas, wiring/bas

PD#27: engine/bas, fraction/bas, geosat/bas, grades/bas, julian/bas, lunarcal/bas, mailist/bas, metaboli/bas,musictrn/bas, perindex/bas, potrack/bas

PD#28: chainfil/bas, citoset/bas, convnum/bas, cursors/bas, cursors/doc, datamkr/bas, deprec/bas, gmenuii/bas, ledger12/bas, menui/bas, menuii/bas, minives/bas, ninteres/bas, refinanc/bas, regdepo/bas, rembal/bas, rndbordr/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

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