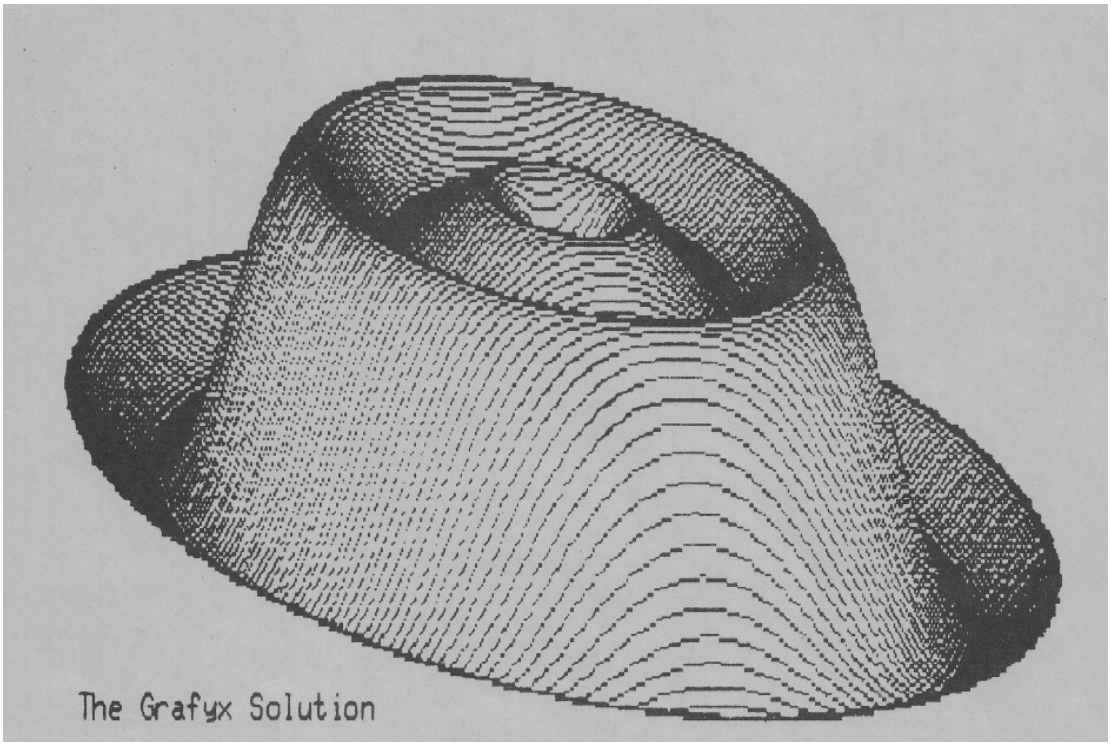
THE GRAFYX SOLUTION ™

USER'S REFERENCE MANUAL V1.3

MICRO-LABS, INC.
902 Pinecrest Drive
Richardson, Texas 75080
(214) 235-0915



μLABS

MICRO-LABS, INC.

THE GRAFYX SOLUTION

Developed by:

Ted Carter

LIMITED WARRANTY

Micro-Labs, Inc. shall have no liability or responsibility to purchaser or any other person or entity with respect to any liability, loss or damage caused or alleged to be caused directly or indirectly by the use or installation of this product, including but not limited to personal injuries, any interruption of service, loss of business and anticipatory profits or consequential damages resulting from the use or operation of this product. This product will be repaired or replaced at our option within ninety (90) days from date of purchase if found defective in manufacturing, labeling or packaging, but except for such repair or replacement the sale or subsequent use of this product is without warranty or liability.

Micro-Labs, Inc. and its authors are not responsible for damage to hardware, software, or data because of its products other than replacement of said product. Except for such replacement, the sale or any subsequent handling of these products is without warranty or liability even through defect, damage, or loss by negligence or other fault.

NOTE: The installation of The Grafyx Solution may void the original ninety day Radio Shack warranty of your TRS-80 computer system.

- FORWARD -

Welcome to the exciting world of The Grafyx Solution![tm]

Your TRS-80 will be transformed into a powerful new system, with graphics capabilities far beyond your dreams after you've installed your Grafyx board and opened up new worlds of programming adventure. The ability to add impressive and informative high resolution graphics opens up many new possibilities for business, educational, and recreational programs.

In the following pages we will discuss the operation of the board, and explore the various techniques which can be used to take advantage of the new high resolution graphics in your computer. Later, you will see some actual programming examples that you may employ in your own works, including a few approaches in Level II and Disk Basic, as well as some assembly source listings for the more adventuresome. Don't worry if you can't understand everything at first; with the supplied point and line subroutines, using the board is as simple as a subroutine call. In addition, you can take advantage of the large number of graphics programs written for other systems with advanced graphics.

With The Grafyx Solution, your TRS-80 is once again a state of the art microcomputer. We look forward to your unique ideas and applications for this revolutionary add-on device for your TRS-80.

- TABLE OF CONTENTS -

1/ A BRIEF DESCRIPTION

The Grafyx Solution gives any configuration of a TRS-80 Model 4 computer a maximum resolution of 640 x 240 for a total of 153,600 individually accessible points. The included graphics package allows you to set and reset points, lines, circles, ellipses, arcs, areas, figures or boxes and complement or clear the screen using simple Basic commands. Alternate resolutions of 320 x 240, 160 x 240, and 160 x 120 are also possible. In a different mode you may choose from resolutions of 512 x 192, 256 x 192, 128 x 192, and 128 x 96.

The ability to add impressive and informative graphics with a resolution better than even the Apple lie or IBM PC opens up many new possibilities for business, educational, scientific and recreational programs.

Software support and documentation for the Grafyx Solution is also impressive. In addition to a number of instructive and interesting demonstration programs and pictures, the board comes with software to save or load a graphics screen and to send a hi-res screen to any of the following printers: all Epson, Star Micronics Gemini printers; Radio Shack LPVII, LPVIII, DMP-series; IDS 445G, 460G, 560G, Prism; Okidata Microline 82A, 83A with Okigraph, 84, 92, 93, 192, 193; Anadex 9500, 9501; Centronics 739; C. Itoh Prowriter I (8510A); and NEC PC-8023.

When enabled, the hi-res graphics screen can be displayed on top of the normal character display or by itself. The Grafyx Solution contains 20,480 bytes of additional read/write memory that doesn't conflict with the TRS-80 address space. Note that 1280 bytes of this is free for you to use in any manner you wish. Upgrading your computer requires no soldering and is as easy as pressing the board onto the corresponding connector and clipping on one micro-clips.

These impressive features are what give The Grafyx Solution its unmatched price/performance ratio.

2/ INSTALLATION AND CHECKOUT

Before you grab your screwdriver. . . As you know, if you haven't opened your Model 4 or 4D computer before, doing so will void your limited 90-day warranty - so you should read through the installation procedure and make sure that you understand what's involved. However, the only real qualifications for the job is the ability to follow directions and use a screwdriver.

Willing to try? Here we go . . . First, clear a working space about the size of a card table. (It is wise to use a soft, non-static surface to avoid scratching the Model 4 case.) Next, assemble the following tools: 1 Phillips head screwdriver and 1 flat head screwdriver. Unplug the power cord, remove any external cables, and follow these steps:

1. Position the computer on its rear panel to provide easy access to the case bottom and remove the ten screws which hold the case together. (One will probably be covered by a black warranty sticker which you must either remove or poke the screwdriver through it). Note the different types and lengths of screws and their locations.

2. Set the computer upright and remove the black screw from the top center of the back panel of the case.

> **WARNING!!!** Use EXTREME CAUTION in the next steps when removing and replacing the cover to avoid damage to the Cathode Ray Tube!!! NEVER allow the rear of the CRT to strike any surface, as it may IMPLODE, and cause injury from flying glass!

3. Now, facing the front of the computer, carefully remove the case top by lifting straight up and setting it to the left on its side. Be careful not to pull on the cables connecting the two case halves.

4. Now position the computer so that you are looking at the back panel, with the keyboard furthest from you. You will now be looking at the RF shielding; a flat metal panel with openings in the top. On the latest

version Model 4 there is a small connector coming out of the middle of the shield at the top that must be disconnected before the shield will come off. If this is the case, mark or remember the orientation of this connector and then pull up to remove it. Now to remove the panel and expose the main computer board, locate and remove the six screws holding it on. Two screws are located to the right, two on the top, and two on the left side of the panel. The metal shield may now be held in place only by the thick aluminum foil stuck to the bottom of the panel. If it is, then carefully peel this foil off without tearing it and remove the panel.

5. At this point you must determine if you have the latest version Model 4 computer. If you bought your computer prior to September 1984 then you will probably have the original version. In any case, it is easily determined by locating the 34 prong graphics connector. On the original version model #26-1069 it is oriented horizontally in the left, middle of the main computer board and labeled J10. On the new version model #26-1069A and on the model 4D it is positioned vertically in the top, center of the computer board and labeled J12. In the following steps variations in installation required for the "new version" are in parenthesis.

6. Remove the black rectangular jumper between E15 and E14 (refer to Figure 1. located at the end of this section) You may want to save the jumper by pushing it all the way down on just post E14 or taping it inside the TRS-80 case.
   (On the new version the jumper is located between pins 16 and 18 of the graphics connector J12.)

7. Remove the screw on the main TRS-80 board that is just above and to the left of integrated circuit number U44. (Refer to Figure 1.) There should be a matching screw in the upper, left corner of the Grafyx Solution board with an aluminum spacer crimped onto it.
   (If you have the new version and there is a screw and spacer in the upper, left corner of the Grafyx Solution board then they should be removed and discarded.)

8. Remove the protective backing from the thick double stick tape located on

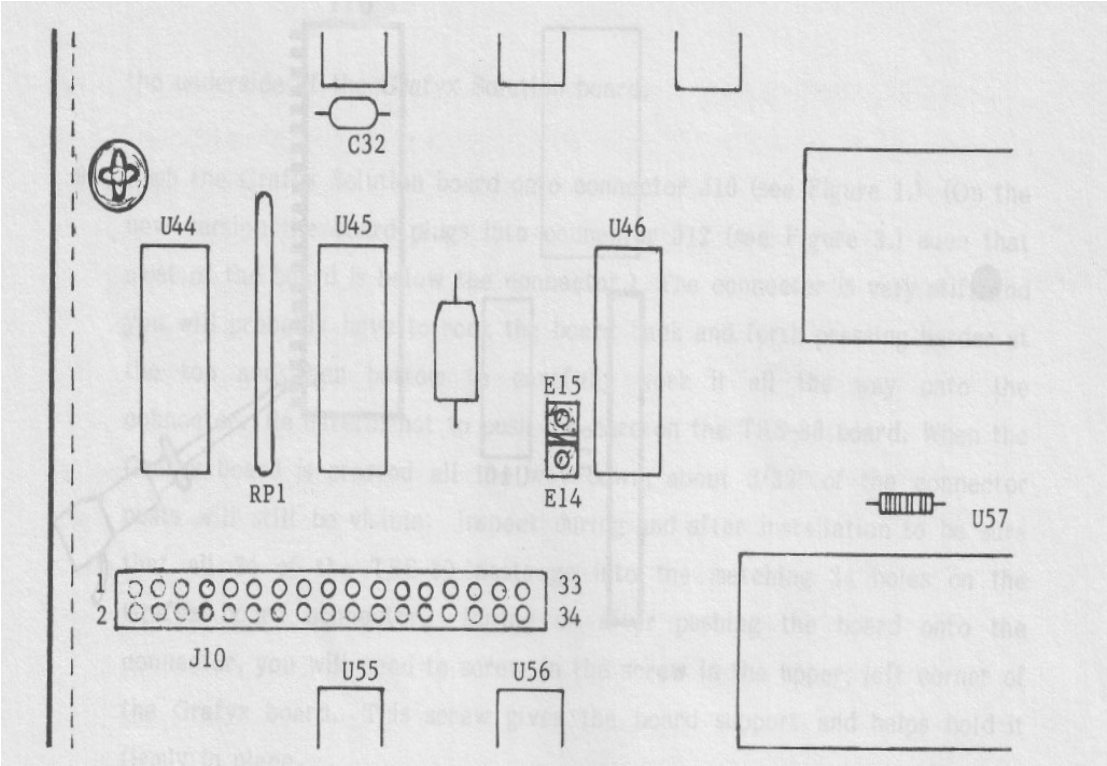


Figure 1. Board Connections (Original version)

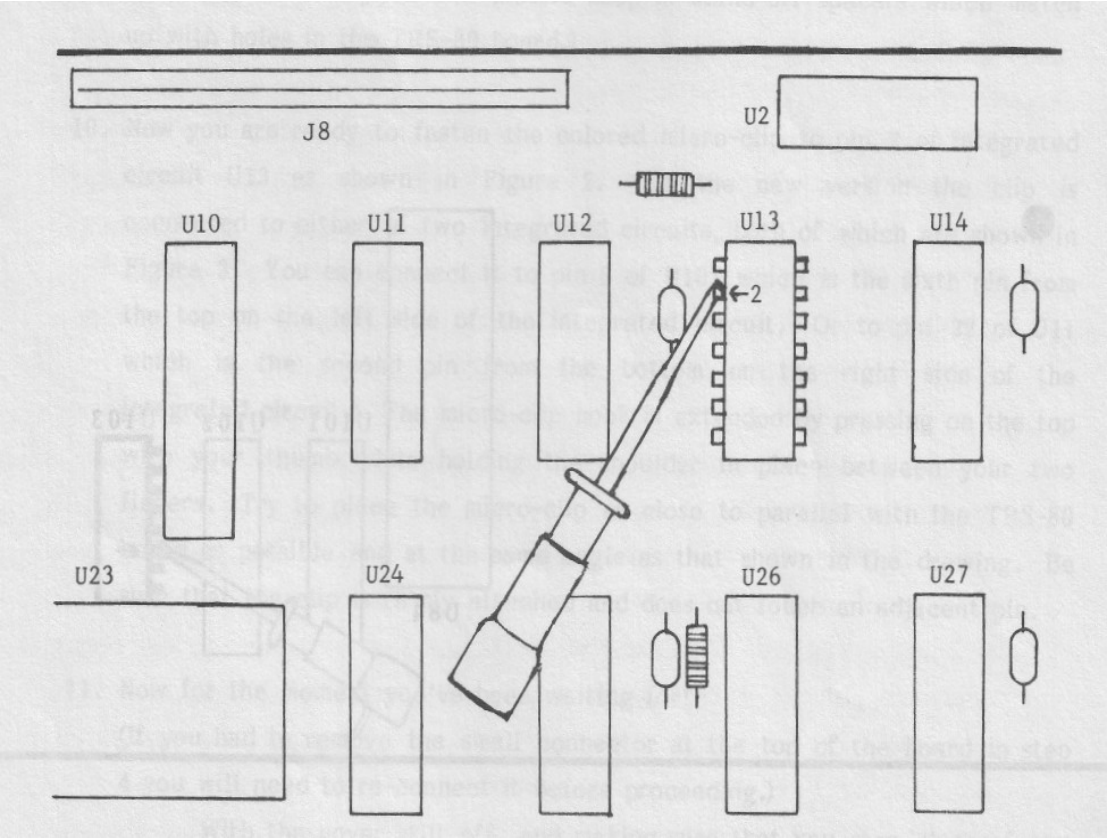


Figure 2. Micro-Clip Connection (Original version)

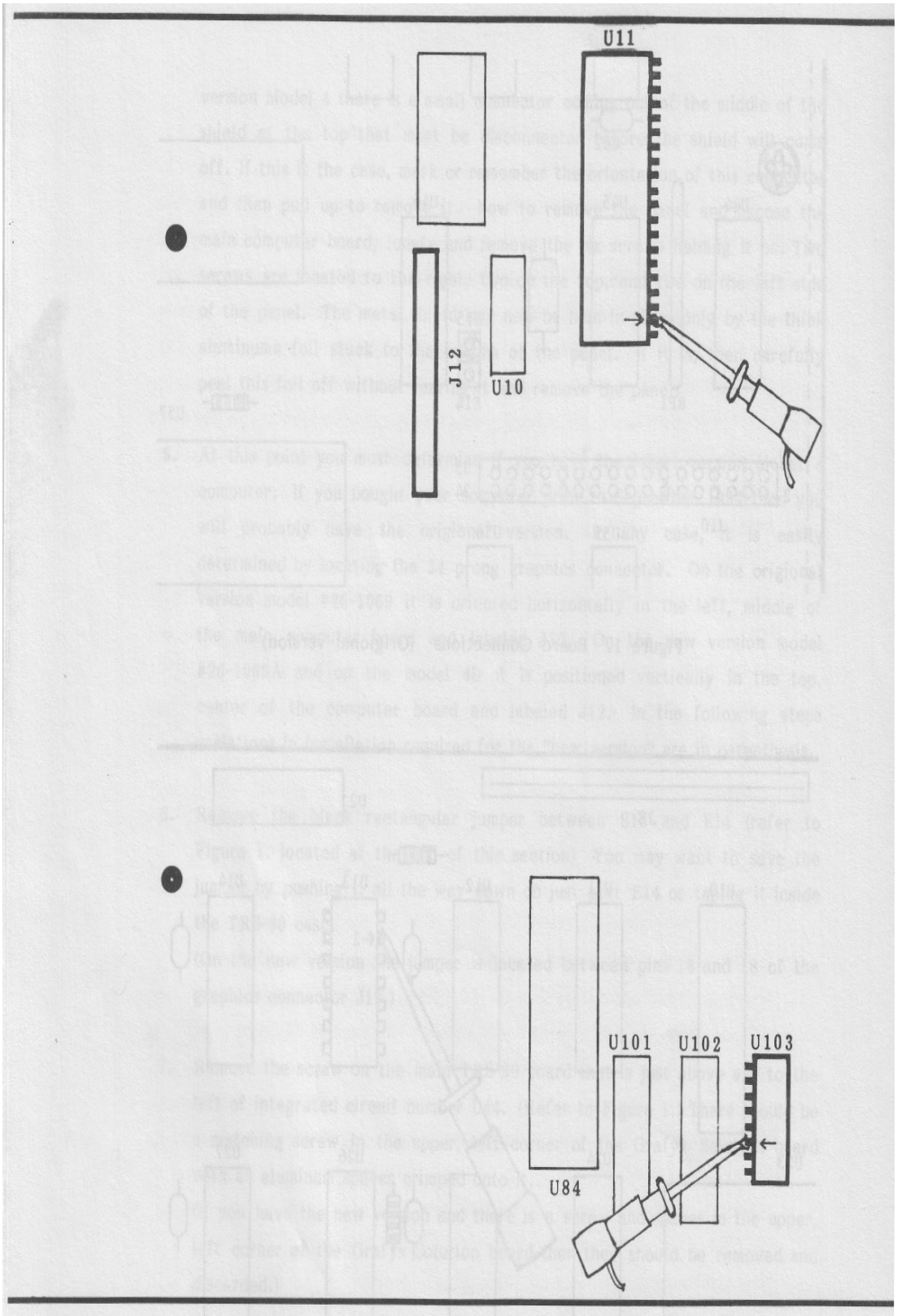


Figure 3. Board Connections (New version)

the underside of the Grafyx Solution board.

9. Push the Grafyx Solution board onto connector J10 (see Figure 1.) (On the new version the board plugs into connector J12 (see Figure 3.) such that most of the board is below the connector.) The connector is very stiff and you will probably have to rock the board back and forth pressing harder at the top and then bottom to carefully work it all the way onto the connector. Be careful not to push too hard on the TRS-80 board. When the Grafyx board is pressed all the way down, about 3/32" of the connector posts will still be visible. Inspect during and after installation to be sure that all 34 of the TRS-80 posts go into the matching 34 holes on the Grafyx board connector. During or after pushing the board onto the connector, you will need to screw in the screw in the upper, left corner of the Grafyx board. This screw gives the board support and helps hold it firmly in place.
   (On the new version there is no screw or metal spacer to deal with. The Grafyx board is secured by the double stick tape. In addition or in place of the tape, there may be two plastic snap-in stand-off spacers which match up with holes in the TRS-80 board.)

10. Now you are ready to fasten the colored micro-clip to pin 2 of integrated circuit U13 as shown in Figure 2. (On the new version the clip is connected to either of two integrated circuits, both of which are shown in Figure 3. You can connect it to pin 6 of U103 which is the sixth pin from the top on the left side of the integrated circuit. Or to pin 22 of U11 which is the second pin from the bottom on the right side of the integrated circuit.) The micro-clip hook is extended by pressing on the top with your thumb while holding the shoulder in place between your two fingers. Try to place the micro-clip as close to parallel with the TRS-80 board as possible and at the same angle as that shown in the drawing. Be sure that the clip is firmly attached and does not touch an adjacent pin.

11. Now for the moment you've been waiting for!
   (If you had to remove the small connector at the top of the board in step 4 you will need to re-connect it before proceeding.)
   With the cover still off, and making sure that you stay clear of any

of the computer electronics, plug in the power cord and turn the computer on. After the video warms up, your Model 4 display should come up just as it did before. Disk owners should have a copy of TRSDOS 1.3, 6.1.2 or 6.2.x in drive 0 and the corresponding supplied Grafyx software disk in drive 1. In response to the TRSDOS Ready prompt, type the following: GBASIC GTEST/ BAS

(Owners with 1 disk drive or a tape system refer to Section 3A for modified loading instructions.)

This command will load the Graphics Basic and execute the basic program GTEST. If everything is working properly, the screen will clear, a logo will be drawn, text will be printed in all four borders and different designs will be drawn. If anything looks wrong, immediately turn off the computer! Otherwise, skip over the next number and go to step 13.

12. Read this if you have a problem.

Every Grafyx Solution board is thoroughly tested before it leaves the factory. Therefore, if any problems in operation are encountered, you should first suspect an installation problem. Removing the board and micro-clip, then installing it again while someone else is carefully double-checking your work will solve most of the problems. Be sure to refer to the drawings and make sure they match up exactly with what you are doing on the TRS-80 computer board. Also make sure the micro-clip is making good contact and only touching the second pin from the top on the left side of part U13. (The sixth pin from the top on the left side of U103 or the second pin from the bottom on the right side of U11 on the new version.) Also be sure that you removed the jumper in step 6.

If you have the new version and the computer does not work at all, make sure that the connector removed in step 4 is plugged back in and in the proper orientation.

If the disk drives don't work properly on the original version computer after putting it back together, take the computer top and shield back off. You will then be able to see a flat white connecting cable located at the top of the TRS-80 board on the left side. This type of cable connection is very poor and it probably got bumped or bent during installation. First try wiggling this cable in order to get it to make contact. If that doesn't work, remove the cable from the TRS-80 main

board and re-insert it. When you do this, be absolutely sure that the conductors line up with the contacts. It is also possible that the cable was pulled loose from the other end, but be warned that if you disconnect the cable from that end that it is very difficult to re-insert.

If you still can't find the problem, write or call Micro-Labs and give a detailed description of the problem and what you have done. And don't worry because we stand behind our product and will get it working for you and if for some very strange reason we can't, we will refund your money.

13. Assuming that everything now works, you can go ahead and finish the job by putting your Model 4 back together. Just to be safe, be sure and **unplug your computer and wait** at least 30 minutes before proceeding. Since the Grafyx board sits above the main computer board, the metal shield that we took off earlier may not go back in place because it hits the top of the Grafyx board. If this is the case, you can solve the problem very easily. The left side of the shield, which is above the Grafyx board, needs to be bowed outward away from the Grafyx board. This can be done by placing the shield over a table edge and pushing down on the two end corners on the left side. If done properly, there will be 1/4 to 1/2" clearance above the Grafyx board and you will still be able to screw back in all six screws.

14. After screwing the shield back on, the installation is completed by carefully placing the Model 4 case top over the case bottom. Make sure that all wires are inside the case. AGAIN, BE CAREFUL NOT TO HIT THE CRT NECK SINCE IT COULD IMPLODE OR BREAK OFF. Install the black #6 x 3/8" sheet metal screw in the top, center of the rear panel of the case. Holding the two halves of the case together, carefully turn the Model 4 on it's back, as before, and replace the ten screws: five 1" sheet metal screws towards the rear, three 7/8" screws along the front, and two 1" screws in the remaining positions.

15. Turn the computer right-side-up and plug it in! You will probably want to run the test program again just to make sure that nothing got bumped when putting the case back together.

REMOVING THE GRAFYX BOARD

If for some reason you want to remove the Grafyx board, simply reverse the series of steps outlined above. The main thing to remember is to put the jumper between E15 and E14 (between pins 16 and 18 of J12 on the new version) back in place or else you will have an inverse video screen.

3/ USING THE GRAFYX SOLUTION

The Grafyx Solution board can be used on three distinct levels of difficulty. To get started, you will probably go to Enhanced Graphics Basic and experiment with all of the supplied demonstration and utility programs. This will give you and understanding of what is possible with the Grafyx board and give you ideas for other things to try. You can also purchase any of the specific applications programs listed in the back of the manual and use them without ever doing any programming.

After getting aquatinted with the Grafyx board you will probably want to learn how to write your own Basic programs. Using the new commands available with Enhanced Graphics Basic it is very easy and rewarding to create new graphics displays and applications programs.

The final level of complexity, which very few will probably do much serious work with, involves accessing the Grafyx board directly. Programs in Basic, Assembly language, or another language can be written which read and write the graphics data directly to the board or which access the Graphics Basic using direct subroutine calls.

In the following sections, you will learn how to use your new Grafyx board on any level of complexity that your needs and interests require.

3A/ USING THE SUPPLIED PROGRAMS

There are four types of files supplied with the Grafyx Solution: machine language programs identified by the /CMD name extension; Basic programs identified by /BAS; assembly language source files identified by /ASM; and Hi-res Screen Data Files identified by /HR. Each of these files is loaded and run differently depending on whether or not you are loading them from a disk or tape based Model 4.

If you received a diskette but have a tape based system, you may obtain a copy of the programs on cassette from Micro-Labs. (Please return the diskette or include $4.00 to cover shipping and handling.)

LOADING INSTRUCTIONS FOR DISK BASED SYSTEMS:

There are two versions of all of the graphics software. One side of the supplied data diskette contains all of the programs for use under TRSDOS 6.1.2, 6.2.0 or DOSPLUS 4 in the Model 4 mode of operation. The opposite side of the flippy diskette contains program versions for use under TRSDOS 1.3 and other Model 3 mode DOSes. (In some cases you may receive two separate diskettes.) If you have two disk drives, you should place the Grafyx Solution Software disk in the top drive number 1 with the label matching the TRSDOS System disk in drive 0 facing up. You will then be able to access all of the graphics programs. Later, after familiarizing yourself with the software, you will probably want to copy the most commonly used files (such as GBASIC) onto your System disk so that they are always readily accessible. If you have only one drive, then you should exchange the disk(s) for ones which have their own self-contained file transfer utility.

Note that in the Model 4 mode you must use the latest version of TRSDOS 6.1.2, 6.2.0 or DOSPLUS 4 which contains Basic 1.1.0. If this is not the version that you are currently using, it should be obtained from Radio Shack or Micro-Systems Software and used in place of the older version.

Before proceeding any further, it is recommended that you use the TRSDOS BACKUP command to make a working copy of the Grafyx Solution software disks as described in the Model 4 disk owners manual.

To run a machine language program stored on disk, simply type its name in response to the TRSDOS Ready prompt. To run a Basic program, enter **GBASIC Programname/BAS** or if you are already in GBASIC, type **RUN "Programname/BAS".** The assembly language source code files are loaded using and editor/assembler such as EDTASM or EDAS/PRO-CREATE by MISOSYS. Screen Files are loaded using the SAVLOAD/CMD program from DOS or the graphics LOAD command from GBASIC.

UTILITY PROGRAMS

GTEST/BAS

This program will run through a series of graphics displays. The successful completion of these tests will assure you that your Grafyx Solution board is working properly. You must be in Graphics Basic to run this program.

GBASIC/CMD, GBASICB/CMD, GBASICD/CMD, L4BASIC/CMD, NBASIC/CMD, D4BASIC/CMD, D5BASIC/CMD

These files are all versions of the Enhanced Graphics Basic designed to work with all of the various major disk operating systems. You enter Graphics Basic in exactly the same manner as you enter regular Basic except that you add a few letters to the word BASIC. For example, to go to Basic and reserve 1 file under TRSDOS 1.3, you would enter **GBASIC -F:1** All other variations such as GBASIC * or **GBASIC file/BAS** are supported. Graphics Basic is actually merged with the standard Basic, using up about 5K bytes additional memory. Therefore, there is no need to set any special memory size since no high memory is used by GBASIC.

A detailed explanation of the commands available in Graphics Basic is contained in Section 3B. Almost all of the programs you write will take advantage of this program.

To run Graphics Basic under disk operating systems other than TRSDOS, you must use that disk operating system's convert command to transfer the files to that DOS. The files and matching DOSes are as follows: TRSDOS 1.3, 6.1.2, 6.2.x use the GBASIC on the matching disk; the following are for use in the Model 3 mode: LDOS 5.1.4 use L4BASIC, NEWDOS80 use NBASIC, DOSPLUS 3.4 use D4BASIC, and DOSPLUS 3.5 use D5BASIC.

The model 4 mode file GBASICD works with the latest version of DOSPLUS 4 which contains Basic 1.1.0. The file GBASICB is the version to use if you have appended the BE11C1VID enhancements file to DOSPLUS 4 Basic 1.1.0. If you do not have the proper version of Basic you will need to obtain the upgrade disk from Micro-Systems Software.

GBASIC/LOD

This file contains the assembly language Graphics Basic commands. It loads into high memory so that your own assembly or other language programs

can make calls to the graphics subroutines. With TRSDOS you would type LOAD GBASIC/LOD to load it and then type the name of your program to execute. See Section 3D on Assembly language techniques. Note that when loaded the program occupies memory space from 60000 to 65439.

SAVLOAD/CMD, SAVLOAD/BAS

When executed, this routine lets you load or save a hi-resolution picture to disk or tape. SAVLOAD first prompts you to load a picture, save a picture in maximum resolution 640 x 240 mode, or save a picture using 512 x 192 dots. (The program will automatically LOAD a picture of either resolution.) The program then asks you for the name of the Screen Data File. At this point you may enter a number from 0-3 to obtain a disk directory for the matching drive number. (This feature will only work with TRSDOS.) The SAVLOAD program automatically adds the /HR extension to the file name if you don't. You may use a name without an extension by adding just the / at the end of the name.

For your convenience, in response to the DOS Ready prompt, you are allowed to type **SAVLOAD L filename** or **SAVLOAD S filename** to load or save a Screen Data File.

In case you want to write a program which directly accesses a Hi-Res Screen Data File, you should know that the 19,200 bytes of Grafyx memory are saved as 75 consecutive records of 256 bytes. SAVLOAD/BAS is a Basic version of the &SAVE and &LOAD commands available in GBASIC. It is included mainly as an example program showing how to access both the graphics memory and the /HR data files.

MODE/CMD

This program gives you a new DOS command which will turn the Grafyx Solution display on or off. In response to the TRSDOS Ready prompt, simply **type MODE V,** where V=0 to disable the graphics display, 1 to enable the 512 x 192 display mode, and 3 to enable 640 x 240 resolution display mode. Non-disk owners have no need for this program since they can accomplish the same thing with the proper Basic OUT statement or GBASIC &OFF, &ON, or &ON1 command.

KEYDRAW/BAS

This is a very simple program which lets you use the cursor keys to draw on the hi-resolution screen.

SUPPLIED DEMONSTRATION PROGRAMS

OVALS/BAS, SPIRAL/BAS, ROSE/BAS, ROSE2/BAS, TRIANGLE/BAS, LILY /BAS, RIBBON/BAS, MOUNTAIN/BAS, FLOWER/BAS, FRAME/BAS, TUNNEL / BAS, CLOVER/BAS, CUBIC/BAS, DESIGN/BAS

These programs draw various geometrical designs while you watch. You must be in GBASIC for these programs to work. All of these short programs are good examples which show how to make use of the Grafyx Solution's capabilities. To exit these programs and turn off the hi-resolution display, simply press and hold down the ENTER key. The LILY and FLOWER programs allow you to start a new picture at any time by pressing the space bar.

ANIMATE/CMD

This assembly language program is an amazing and amusing demonstration of the hi-res animation now possible. If you like this program we recommend the Dog Catcher program listed in the back of the manual. (Model 4 mode only)

LGRAPH/BAS

This GBASIC program demonstrates how effective and attractive high resolution graphics can be when presenting information in the form of a line graph.

DFILL/BAS

This program demonstrates the wide variety of FILL patterns possible. You can exit this program by pressing the Break key.

MISSILE/BAS

The techniques necessary to program moving graphics using GBASIC are demonstrated in this program and explained in Section 3C. In this example a small arrow is moved vertically across the screen at different speeds.

HAT/BAS

This GBASIC program generates a three-dimensional picture which resembles a hat. Because of the many calculations required for each point and the large number of points, it takes a couple hours to complete the picture.

PLOT/BAS

This is a Basic program which demonstrates the algorithm used to plot a

point from Basic. This program is explained and listed in Section 3C.

LINE/ BAS

This Basic program demonstrates the algorithm used to draw a line. This program is explained and listed in Section 3C.

VECTORS/ASM

This source code file contains an assembly language program which demonstrates how to interface with and use the GBASIC graphics routines. This particular program draws lines specified by a table of endpoints in order to create a pyramid. Dramatic graphics displays can be created very easily by taking advantage of the GBASIC assembly language routines as explained in Section 3D.

POINT/ASM

This file contains the source code for the clear screen and point plotting algorithms. Designed to be merged with your own assembly language programs, its use is explained in Section 3D.

HI-RESOLUTION PICTURES

SUNSET/HR

A picture which was drawn using GBASIC commands.

SURFACE/HR

The SURFACE PLOT II package was used to create this display.

BIZGRAPH/HR

This picture was created by using the BIZGRAPH package to read in Visicalc data and plot it in one of its many possible graph formats.

SHUTTLE/HR

This picture was created with the DRAW program.

3B/ ADDING GRAPHICS TO BASIC PROGRAMS

In order to make the most of your new, advanced computer, we have supplied the GBASIC program which ties into Basic and becomes transparent to the user. This program allows you to perform a large number of functions at 4 Mhz machine language speeds using simple Basic statements such as &PLOT (X, Y, C).

In order to try using all of the graphics commands as they are explained, we recommend that you now go to GBASIC by typing GBASIC in response to the TRSDOS Ready prompt. (See Section 3A if you are using a DOS other than TRSDOS) You should now be in Extended Graphics Basic.

The examples in this text use the syntax which is correct for operation in the Model 3 mode. If you are in the Model 4 mode, change **Z=&** to **@** and **+&** to **:@**

The first thing that you should do is clear the graphics screen by typing **Z=&CLS** (be sure and end all lines by hitting the **ENTER** key). This statement is necessary since the display probably has random dots in it from when the computer was first turned on. You should next type **Z=&ON** so that the hi-resolution screen will be displayed. Note that you will not see any change in the display since the hi-resolution screen was just cleared. You can now use the &PLOT (X,Y,C) statement to plot points anywhere on the screen. X is the value of the X-coordinate and can be from 0 to 639. Y is the Y-coordinate and can range from 0 to 239. C is the color of the point; 0 will clear the point, 1 will set the point, and 2 will complement the point.

Try this statement out by typing **Z=&PLOT (256,96,1)** You should see a small dot appear in the middle of the screen. Now enter and run the following Basic program:

```
10 FOR X=0 TO 511
20 Z=&PLOT (X,X*192/512,1): NEXT
```

You should see a diagonal line drawn across the screen as both the X and Y coordinates increase. To turn off the hi-resolution display, type **Z=&OFF** and the line will disappear. Even though it is not displayed, the contents of the Grafyx Solution remain intact, and the same picture will appear if you type **Z=&ON** to re-enable the graphics. In fact, you could have just as easily run the above program with the hi-resolution screen turned off, and then enabled it to

see the result. An interesting thing to do is to continue to hit **ENTER** until the screen scrolls. As a result of this experiment it is obvious that changing the normal display has no effect on the hi-resolution display.

The results of the above program could have been achieved much faster by taking advantage of the &LINE (Xl,Y1,X2,Y2,C) statement. This statement will draw a line between any two points X1,Y1 and X2,Y2 with the color determined by the value of C as in the &PLOT statement. Now enable the hi-resolution screen and clear it by entering **Z=&ON+&CLS** Next, type **Z=&LINE (0,0,511,191,1)** and you will see the same line produced earlier using the &PLOT statement.

To demonstrate another neat statement, type **Z=&REV** with the Grafyx Solution enabled. This changes every point that was on to off and vice versa. You can type **Z=&REV** again to return the screen to being white on black.

All of these experiments were done using the 512 x 192 resolution mode of the Grafyx board. This is due to the fact that in Model III mode under TRSDOS 1.3, the regular text screen overlays the 512 x 192 graphics screen so that you can see what you are doing. Now type the statement **Z=&ON1**. The text will disappears and you will see the full 640 x 240 maximum screen resolution. Even though you can not see what you type, your commands are still being interpreted by the computer. To prove this, carefully type **Z=&ON** to switch back to the text overlay mode.

The Grafyx Solution software also gives you four different plot densities within each of the two 640 x 240 or 512 x 192 resolutions. If you do not need the mode with the highest resolution, your programs don't have to update as many points and will run faster. Mode 1 (the default mode) gives you dot resolutions of 640 x 240 (512 x 192), mode 2 gives 320 x 240 (256 x 192), mode 3 gives 160 x 240 (128 x 192), and mode 4 gives 160 x 120 (128 x 96). You can change the mode by using the statement &MODE (M) where M is the number of the mode. The X,Y values in each mode have the same range as before, meaning that more than one X,Y value maps to a particular point in the lower resolutions. If you try to plot a point outside the appropriate range, the software will ignore it. You can change the mode at any time without affecting the display since the lower resolutions are simulated with software and not determined by the hardware. This allows you to mix crude graphics with

detailed displays in order to take advantage of faster updating with the low resolution graphics while maintaining the high picture quality where necessary. For example, you may find that when drawing vertical lines you need to be in mode two in order to make the lines stand out enough.

That's all there is to using the Grafyx Solution. GBASIC lets you do everything you need to do from Basic with easy to use commands. The most commonly used commands have now been explained so we will proceed to some of the more sophisticated commands. But to keep you from getting confused by all of the new functions, we'll go ahead and present a summary of every function available with GBASIC:

GBASIC COMMAND SUMMARY:

In the Model 3 mode all commands must be preceded by an ampersand and appear as part of an expression that is evaluated. In other words, a variable must be set equal to the function as in A=&ON After this statement is executed, variable A will usually be changed to some random value. Memory can be saved by combining functions that appear on consecutive lines. For example, Z=&CLS: Z=&MODE (2): Z=&ON can be changed to Z=&CLS + &MODE (2) + &ON

In the Model 4 mode the @ symbol next to the P key is used in place of the ampersand. Also, each command is a separate statement and should not be set equal to anything. For example, instead of Z=&ON+&CLS it would be @ON:@CLS

As used in the GBASIC command summary,

X,X1,X2 are X-coordinate values from 0-639.

Y,Y1,Y2 are Y-coordinate values from 0-239.

X=Y=O is the upper, left corner of the screen.

X,Y values from -4000 to 4000 are allowed but not plotted.

C is the color of the point(s) plotted by the function; 0 clears the point(s), 1 sets the point(s); 2 complements the point(s). In the LINE and BOX commands, C may equal a value up to 255 for a wide range of line patterns. For example, C=3 draws a grey line.

&ON - Enable the hi-res display with 640 x 240 resolution. (512 x 192 in the

Model 3 mode.) The text screen will also be displayed.

&ON1 - Enable the hi-res display with 640 x 240 resolution. The text screen will be blanked.

&OFF - Turn the hi-res display off. On entry, GBASIC assumes the status of the display to be off.

&CLS - Clear the hi-res screen without affecting the normal text screen.

&MODE (M) - Set the Grafyx Solution to the desired plotting resolution. M=1 for 640 x 240 (512 x 192); 2 for 320 x 240 (256 x 192); 3 for 160 x 240 (128 x 192); 4 for 160 x 120 (128 x 96). The mode can be changed at any time since it does not affect points already drawn.

&PLOT (X,Y,C) - Plot a point. The size of the point depends on the current plotting Mode.

Z=&POINT (X,Y) - Read the point status and set variable Z equal to 1 if the X,Y position is set; 0 if it is clear; 2 if it is out of bounds.

&LINE (X1,Y1,X2,Y2,C) - Draw a line between the two endpoints.

&LINE (X2,Y2) - Draw a line going from last point plotted to X2,Y2 using the previous line color.

&REV - Complement every point on the hi-resolution screen.

&LPRINT (P) - Print the contents of the current display. P=0 for Radio Shack DMP-series; 1 for Epson, Star Micronics printers; 2 for enhanced Epson, Star Micronics printers; 3 for Radio Shack LPVII, LPVIII; 4 for Okidata Microline 82A, 83A with Okigraph, 84 step 2, IDS 445G, 460G, 560G; 5 for Centronics 739; 6 for Okidata Microline 92, 93; 7 for Anadex 9500, 9501; 8 for C. Itoh Prowriter I (8510A), NEC PC-8023; 9 for Okidata Microline 84.

&BOX (X1,Y1,X2,Y2,C) - Draw a box whose diagonal corners are equal to the two given coordinates. Note that X1<>X2 and Y1<>Y2.

&CIRCLE (X,Y,R,C) - Draw a circle with a radius of R and center at X,Y. The

radius is the number of points in the X direction.

&CIRCLE (X,Y,R,C,,AR) - Draw an ellipse using and aspect ratio of .0001 to 1000. (AR=.5 draws a circle)

&CIRCLE (X,Y,R,C,ST,EN) - Draw an arc beginning at ST and ending at EN radians (0<=ST,EN<=6.28319).

&CIRCLE (X,Y,R,C,ST,EN,AR) - Draw a section of an ellipse.

&FILL (X,Y,S) - Color in the shape surrounding the point X,Y. Shading pattern 0<=S<=255. Values less than 128 skip every other line and are less dense. S=255 gives a solid fill; S=170 gives a grey color.

&SAVE, &LOAD - Save or load a 640 x 240 format hi-res picture file from or to the Grafyx display. You must use the standard Disk Basic commands to open and close the file as follows:

```
OPEN"R",1,"File/HR": Z=&LOAD: CLOSE
```

Note that the file **must** always be file number 1. Also, note that if you enter TRSDOS 1.3 GBASIC specifying a number of files other than 3 AND you append a V to the number of files to specify variable length files, then you must precede the OPEN statement by POKE 28705,123

&POS (X,Y,D) - Causes all PRINT statements to display on the graphics screen beginning at position X,Y. All normal and special characters 192-255 can be displayed. D=0 prints text from left to right; 1 prints sideways from top to bottom; 2 prints upside down from right to left; 3 prints sideways from bottom to top. &RESET or &OFF restore normal PRINT operation.

&RESET - Undoes the effect of the POS command so that text PRINTed will once again be shown on the standard text screen.

&GET (X1,Y1,X2,Y2,Z%(0)) - Copy the rectangular section of the screen with corners at the given coordinates into a one or two letter single dimension integer array variable such as Z%. The array must have a dimension at least equal to INT((ABS(X2-Xl)+1) *(ABS(Y2-Yl)+l) / 16) + 4.

&PUT (X,Y,Z%(0),F) - Display the contents of the array created with the &GET statement beginning at the given X,Y position. Portions of the rectangular area extending past the edge of the screen will wrap around

and be placed on the opposite side. F=0 to complement every point within the rectangle; 1 to copy the contents of the array to the screen; 2 to AND the contents of the array and screen; 3 to OR the contents of the array and screen; and 4 to XOR the contents of the array and screen.

&USING (X,Y,X1,Y1) - The two X,Y coordinates define the corners of the rectangular viewing area to be used. Any points from that point on that are outside this area will not be plotted. It is a good idea to reset the viewing area with the statement &USING (0,0,639,239) when the smaller viewport is no longer required.

&USING (X,Y,X1,Y1,F) - Same as above plus: F=0 draws a box of color=0 around area to be used; 1 draws a box of color=l around the area; 2 erases the contents of area; 3 paints the area solid; 4 to 129 fills in the area with various patterns; 130 to 255 draws a box with various line patterns around the area to be used.

SOME PROGRAMMING EXAMPLES:

The first example shows you how simple it is to plot a function on an X-Y axis. You should enter and run the GBASIC program to see what it does. (Note that you don't need to type in the comments for the program to work.)

```
10 CLS :' Clear normal screen
15 Z=&CLS+&ON :' Clear hi-res screen and turn it on
20 Z=&MODE (2):' Set to double width points
25 Z=&LINE (0,96,511,96,1) :'Draw X axis
30 Z=&LINE (0,0,0,191,1)    :'Draw Y axis
35 FOR X=0 TO 511 STEP .5 :'Step through X values
40 Z=&PLOT (X,96-40*SIN(X/20),1) :'Plot point
45 NEXT X :'Do all 1024 points
50 Z=&MODE (1) :'We want small grid points
55 FOR X=10 TO 510 STEP 20 :'Now draw grid by changing
60 FOR Y=6 TO 190 STEP 10 :'both X and Y in steps and
65 Z=&PLOT (X,Y,1)            :'setting points.
70 NEXT Y: NEXT X
```

In the above program, the function to be plotted is placed in the PLOT statement as the Y-coordinate value. The value that X is divided by within the SIN function determines the period of the sine wave. Since the SIN function returns a value between 1 and -1, we need to scale it in the vertical direction

and negate it so it is right side up by multiplying by -40. We then have a value between -40 and 40 so we have to add 96 to it in order for it to be centered on the screen. Any other kind of function could be used in place of the SIN function as long as the values were scaled so that they all fit in the range of 0 to 191. From this program it is apparent that the graphics function's parameters can actually be complex numerical expressions. This program could easily be expanded so that it printed a title at the top of the graph and labeled the axis values.

The second program demonstrates how easy it is to create a unique design using the LINE statement. All this simple program does is draw a series of lines, keeping one end at the point 500,0, while moving the other endpoint across the bottom of the screen.

```
10 CLS :'Clear normal screen
20 Z=&CLS+&ON1+&MODE (1) :'Get ready to draw lines
30 FOR X=0 TO 639 STEP 7 :'Change one end of line
40 Z=&LINE (500,0,X,239,1) :'Draw line
50 NEXT X :'Keep on till all lines done
60 IF INKEY$=""THEN 60 ELSE Z=&OFF:'Now wait for keypress
```

Now change the color of 1 in line 40 to be the expression X/3 and run it. As you can see there is a wide variety of line styles available. The line pattern actually corresponds to the binary bit pattern for the color specified but you don't have to know that to use it do you!

Now, add the following lines to your program and run it:

```
25 Z=&USING(160,60,480,180,1)
55 Z=&USING(0,0,639,239)
```

As you can see, the USING command in line 25 specifies that you only want to use a subset of the entire screen and the color of 1 draws a box around that area. The command in line 55 resets the graphics screen to the normal viewing area.

The next program demonstrates the use of the CIRCLE, FILL, GET, and PUT statements all in just a few lines of code. Enter the program and try and figure out what it does before you run it.

```
10 DIM A%(409) :'Make room for circle shape
15 CLS :'Clear normal screen
20 Z=&CLS+&ONl+&MODE(1) :'Set up a clear screen
25 Z=&CIRCLE (320,120,48,1) :'Draw circle
```

```
30 Z=&FILL (320,120,170) :'Fill in circle
35 Z=&GET (320+56,120+28,320-55,120-29,A%(0))
40 Z=&PUT (640*RND(0),240*RND(0),A%(0),1+2*INT(RND(0)+.3))
45 IF INKEY$="" THEN 40 ELSE Z=&OFF
```

Line 25 places a circle in the middle of the screen which has a radius of 48 dots in the horizontal direction. Line 30 colors in the circle with a shaded pattern. Line 35 places the rectangular section of the screen which includes the circle into the array A%. Line 40 is the main part of the whole program. What this line does is take the contents of the array A% (which contains the shaded circle shape) and places it onto the screen in a random location. The function 640*RND(O) results in a random X-coordinate between 0 and 639 and the same thing is done to come up with a Y-coordinate. All the function 1+2*INT(RND(0)+.3) does is randomly generate either a value of 1 or 3, but usually a 1. This means that the shaded circle will usually be copied directly to the screen, erasing any previous circles. But when the function generates a 3, the circle will be ORed onto the screen and merged with any other circles in the area. Both values are generated so that you can see how they affect the operation of the PUT statement. You should now have a better understanding of the CIRCLE, GET, and PUT statements and be able to go ahead and try writing some programs which use them. To stop this program press any key.

Now lets learn some more about the very powerful and flexible CIRCLE command. Enter and run the following program:

```
10 Z=&CLS+&ON1+&MODE(1):'Set up clear screen
20 FOR AR=.2 TO 2 STEP .2:'Go through range of aspect ratios
30 Z=&CIRCLE(320,120,100,1,,AR):NEXT: 'Draw ellipse
40 IF INKEY$="" THEN 40 ELSE Z=&OFF:'Wait for keypress
```

Now change line 30 to be `Z=&CIRCLE(320,120,100,1,0,3.141,AR):NEXT`

When you run this you will see that semicircles are drawn. This is because we specified that the circle start at the top (0 radians) and end at the bottom (pi=3.141 radians). You can also leave off the AR parameter if you want sections of circles instead of ellipses.

Now, assuming that you like the hi-resolution picture you just created, we will save it on disk and then load it back to prove that it worked. Enter the following program which is self-explanatory.

```
10 OPEN"R",1,"TEST/HR":'Use picture file named TEST
20 Z=&SAVE:'Write current hi-res screen to disk
30 Z=&CLS:'Erase picture
40 Z=&ON1+&LOAD:'Load picture back onto the screen
50 CLOSE:'Always close the file when we are done with it
60 IF INKEY$="" THEN 60 ELSE Z=&OFF:'Wait for keypress
```

Now suppose that you really like your hi-resolution picture and want to print it. Since you are now looking at the Basic Ready prompt, do you have to write a new program? No! Any of the GBASIC commands may be entered directly as a command. Refer to the command summary of &LPRINT for the value corresponding to your printer, but let's say you had an EPSON printer. If you wish to print **only** the hi-res screen you could type **CLS:Z=&LPRINT(1)** However, if you do not clear the normal text screen then GBASIC will print things exactly as you see them on the screen, even if you have text, low-resolution block graphics or Special Characters displayed.

A very useful and neat command (I think its neat since you can write letters upside down) is demonstrated in the following program:

```
10 Z=&CLS+&ON1+&MODE(1):'Clear screen
20 Z=&POS(320,120,0):PRINT "Testing.1.";
30 Z=&POS(320,120,1):PRINT "Testing.2.";
40 Z=&POS(320,120,2):PRINT "'Testing.3.';
50 Z=&POS(320,120,3):PRINT "Testing.4.';
60 Z=&POS(0,0,0):'Put cursor at top of screen and end
```

Note that when the program ends, you are still in the 640 x 240 mode and the text being printed is actually being programmed into the hi-resolution board. In fact, you can type **LIST** to see your program. However, when an attempt is made to print past the bottom of the screen in an invalid location, the &POS command will be disabled. Normally you would use the &RESET or &OFF commands to reset the video driver to print regular text. Note that not only can you display text with the &POS command, but you can use the CHR$(x) statement to print low-resolution block graphics codes and any of the 64 Special Characters!

The following program uses the &PUT command to move a missile up the screen one dot at a time. Since this program is included under the name MISSILE/BAS, the following listing contains only the main program:

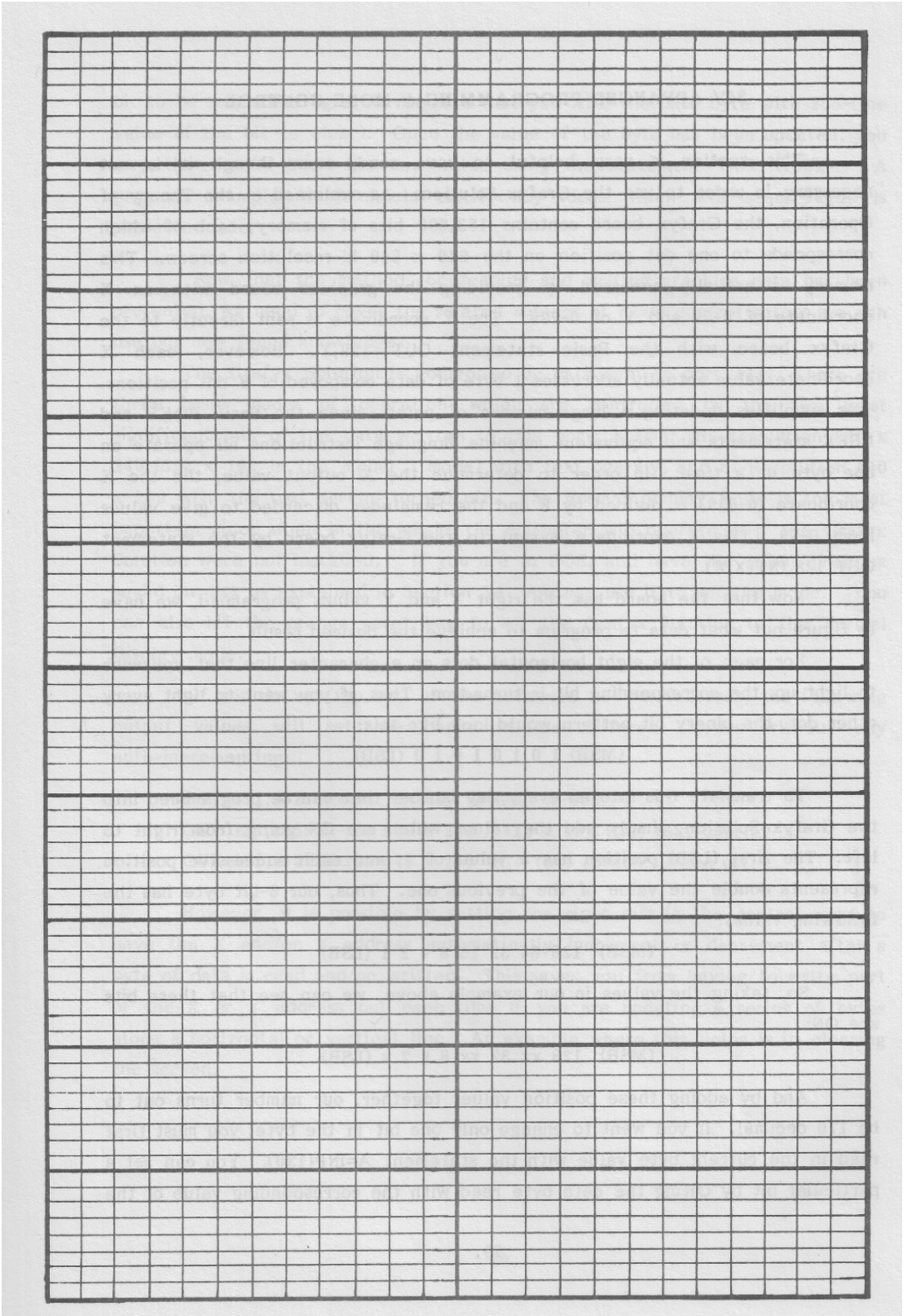
```
15 CLS: Z=&CLS+&ON+&MODE(1)
25 READ X,Y: IF X>=0 THEN Z=&PLOT (X,Y,1): GOTO 25
30 Z=&GET (1,1,8,15,A%(0)) : S=-1
35 Z=&CLS
45 FOR Y=175 TO 0 STEP S: Z=&PUT (256,Y,A%(0),1): NEXT
50 S=S-1: IF S>-20 THEN 35
60 DATA 4,1,5,1,...,-1,-1
```

Line 15 clears the normal and hi-resolution screens and makes sure that the mode is 1. Line 25 is a loop which continues to read X,Y coordinates and plot them until -1,-1 is read. After line 25 has plotted all the data, the result is a small missile figure in the upper, left corner of the screen. Line 30 takes this figure and moves it into array A%. Although not immediately obvious, the area saved includes one extra row of dots along the bottom. This is because we want to move the shape up the screen one dot at a time and have the old shape erased as we go. Line 45 is a loop which places the missile on the screen in different vertical positions using the PUT function. The value S determines how many dots are skipped between figures. Since S initially equals -1, the missile will move up one dot at a time. However, line 50 increases the magnitude of S so that the number of spaces skipped continues to increase until S reaches -20. When the program is run, you will see that after the first pass, more and more dots fail to be erased since the shape we made only included one row of trailing blank dots.

This same idea could be used to move a shape in **any** direction, even diagonally. You just need to be sure that there are enough blank dots to erase the previous figure when you place the new one in a slightly shifted position. Keep your figures small so that the time required to move the many points is minimized.

To make it easier to draw large objects, we have included a grid sheet on the next page which shows a matrix of 80 x 24 dots. You will probably want to make a number of copies of this grid for future use.

That's all there is to using the Grafyx Solution. By now I'm sure that you are overflowing with ideas for neat little graphics displays. So go ahead and experiment! No longer will low resolution graphics damper your imagination!



3C/ ADVANCED PROGRAMMING & MODE CONTROL

This section is very helpful to have read, even though it is not necessary in order to use the Grafyx Solution. As explained in the Theory of Operation, the Grafyx board contains 153,600 bits of memory, each of which corresponds to one dot position on the 640 x 240 hi-resolution screen. This memory is actually accessed by providing the graphics board with an X coordinate of 0-79 and Y of 0-239. The Y coordinate is sent directly to the Grafyx board with the Basic statement OUT 129,Y. However, each X coordinate value actually addresses 1 byte of data composed of 8 dot positions. This simplifies updating a large number of points since the Basic POKE and PEEK statements and equivalent machine language instructions all operate on one byte at a time. In order to determine the X output value, the the X coordinate (0-639) is divided by 8 and the remainder discarded to give values from 0-79. This X coordinate is sent to the Grafyx board by the statement OUT 128,INT(X/8).

Now that the board has the right X and Y values programmed, we have to figure out what data to program to achieve the desired result.

For each of the eight horizontal dots on a character line that you wish to light up, the corresponding bit is turned on. Thus, if you want to light every other dot, the binary bit pattern would look like this:

```
(MSB) 1 0 1 0 1 0 1 0 (LSB)
```

To translate this into an every day number that can be programmed into the Grafyx Solution, simply add the values which are ON going from right to left. The first (LSB) **position** has a value of 1, and each successive position represents **double** the value of the previous one. Thus, our 8-bit byte has the following values:

```
(MSB) 128 64 32 16 8 4 2 1 (LSB)
```

So, taking the values in our example above, we can see that these bits are ON:

```
(MSB) 128 xx 32 xx 8 x 2 x (LSB)
```

And by adding these position values together, our number turns out to be 170 decimal. If you want to change only one bit in the byte, you must first read in the current byte value with the statement A=INP(130). You can set a particular bit by ORing the data byte read with the corresponding value of the

bit to be set. Clearing a bit requires that you AND the data byte with 255-(the value of the bit to clear). Once the value of the byte has been updated, you may then write back the new value using the statement OUT (130),A. A program example which should clarify any ideas which still seem vague is presented later.

Now that the method for reading and writing graphics data has been presented, we need to cover the related subject of mode control before we can use our new found knowledge.

The Grafyx Solution is controlled by data sent to port 131. There are three basic modes of operation which are controlled by the two least significant bits of the data written to this port: Normal, Hi-Resolution 640 x 240 (512 x 192 in the Model 3 mode) with text overlay, and Hi-Resolution 640 x 240 with the text screen turned off. The Normal Display mode is the most important since when in this mode, the computer will operate as if the Grafyx Solution were not installed. If you are in DOS, and have the MODE program saved on disk as explained in Section 3A, you can use it to set the mode. You can also hit the orange reset key to set the Grafyx Solution to the Normal Display mode.

If you do not use any of the advanced control features, the following output values will set the board in the different modes without any auto-incrementing:

```
OUT 131,252:'Normal display without hi-res
OUT 131,253:'Hi-res 640 x 240 / 512 x 192 with text overlay
OUT 131,255:'Hi-res 640 x 240 with text screen turned off
```

However, it is possible by setting the right bits in the control word to have the X and/or Y address automatically increment or decrement after a byte of data is read and/or written. This saves you from having to write part of the X or Y address out each time if you are updating a series of bytes along a horizontal or vertical line. An example where this helps is in clearing the screen.

A summary of the Grafyx Solution Port
Mappings follows:

OUT 128,x coordinate (0-79)

OUT 129,y coordinate (0-239)

OUT 130,read/write data byte

OUT 131,control register as defined below:

Bit #	Sum value	Bit definition	
0	1	graphics / alpha*	
1	2	640x240/512x192*	
2	4	X register dec/inc*	1 selects decrement
3	8	Y register dec/inc*	" "
4	16	X clock read*	0 clocks after read
5	32	Y clock read*	" " "
6	64	X clock write*	0 clocks after write
7	128	Y clock write*	

The many combinations of auto-incrementing or auto-decrementing X or Y addresses are obtained by going down the list of control bits and deciding if the bit should be on (=1) or off (=0) according to the bit definition. You then
add all of the "sum values" for bits that you determined should be on to get an output value from 0 to 255.

For example, if we want to increment the X register after every data write and want the 640 x 240 mode we would want the following bits on: 0,1,4,5,7. Adding the sum values gives us 179.

To demonstrate how you would enable the above modes, enter and run the following program:

```
10 OUT 131,255:' Set to show hi-resolution screen
20 FOR X=1 TO 1000: NEXT:' Delay momentarily
30 OUT 131,252:' Set back to Normal Display mode
```

This program simply turns the Grafyx board on, waits a little bit, and turns it off again. The contents of the Grafyx display will depend on what was last programmed in it, or if nothing has been programmed, the screen will be full of random dots and/or lines. Note that the OUT statement can be used as a command while in Basic. This means that you could type **OUT 131,255,** for example, in response to the Basic > prompt and the graphics screen would be enabled.

Now we are finally ready to put it all together and write a program which programs the Grafyx Solution with a meaningful display. Our program will sequence through all displayed memory locations. The data value stored in

each location depends on what kind of pattern you want the screen to be filled with. In order to clear the graphics screen, the data would be 0; to paint the screen dark, it would be 255. To make things interesting, lets have our program use the value 170, which, as we have seen previously, is the value needed to have a display with alternating light and dark points. The following program should now be entered and run. This program should help you to understand how a particular graphics point is addressed since Basic is slow enough that you can see the points as they are updated.

```
20 OUT 131,1+2+16+32+64:' Enable hi-res mode, Y auto-inc
30 FOR X=0 TO 79: OUT 128,X:' Output X coordinate
40 OUT 129,0:' Set beginning Y coordinate = 0
50 FOR Y=0 TO 239: OUT 130,170: NEXT:'Write to vertical column
60 NEXT X:' Do next column
```

Now change the 170 in line 50 to 0 and run the program again so that the graphics screen is cleared and ready for the next project.

A common application of the Grafyx board might be to display a graphics figure such as a missile. The first thing that you must do is determine which points need to be set by possibly sketching the figure on a copy of the supplied grid sheet. Once this is done, you can determine the program data by looking at which bits in each line are set and adding up their values as explained previously. Assuming that you want to draw a missile, you might end up with data which looks like this:

Line	Character	Binary	Dec	Hex
0	**	0 0 0 1 1 0 0 0	24	18
1	****	0 0 1 1 1 1 0 0	60	3C
2	* ** *	0 1 0 1 1 0 1 0	90	5A
3	**	0 0 0 1 1 0 0 0	24	18
4	**	0 0 0 1 1 0 0 0	24	18
5	**	0 0 0 1 1 0 0 0	24	18
6	**	0 0 0 1 1 0 0 0	24	18
7	**	0 0 0 1 1 0 0 0	24	18
8	****	0 0 1 1 1 1 0 0	60	3C
9	******	0 1 1 1 1 1 1 0	126	7E
10	*** ***	1 1 1 0 0 1 1 1	231	E7
11	** **	1 1 0 0 0 0 1 1	195	C3

Now lets assume that the missile is to be drawn in the middle of the screen. The position of the whole character would then have an X coordinate of 40 and a Y coordinate of 120. So, moving on, you should be able to figure

out how the following program works by studying the comments and the resulting display when the program is run:

```
100 ' This program places a missile on the screen
110 ' Note: we assume that the Grafyx screen is clear
120 CLS :' Clear regular screen
130 X=40 :' Set X position to the middle of screen
140 Y=120:' Set Y position to the middle of screen
150 OUT 131,1+2+16+32+64:' Clock after write
160 OUT 128,X: OUT 129,Y:' Write location
170 FOR Z=1 to 12
180 READ D:' Get data value for the line
190 OUT 130,D:' Write data byte, auto-inc Y address
200 NEXT Z
210 IF INKEY$="" THEN 210 ELSE OUT 131,252
220 ' Data values for the twelve lines in missile:
230 DATA 24,60,90,24,24,24,24,24,60,126,231,195
```

Now that you basically understand how the Grafyx Solution is programmed, we will present a general purpose algorithm for plotting a point. This program should be studied in order to see how an X,Y coordinate on a 640 x 240 matrix is translated to a bit within the proper byte. If you received your programs on a disk, this and the following line drawing program are included so that you don't have to type them in. The program first clears the screen and then sets points at random screen locations. The point plotting routine is written as a subroutine starting at line 200. In general, what the subroutine does is figure out the X position (0-79) of the byte containing the point and the bit position within the byte. The byte is read in from the Grafyx memory, the proper bit is set, and the updated byte is then written back. As you will soon see, Basic point and line drawing routines are slow, which is why you will usually want to use the machine language routines available in GBASIC.

```
10 ' PLOT/BAS
20 ' This program demonstrates the point plotting algorithm
30 ' After clearing the screen, the point coordinates X and Y
40 ' are randomly generated and the point plotting subroutine
50 ' is called. Hitting any key will exit this program.
60 '
70 CLS:' Clear video screen
80 OUT 131,1+2+16+32+64:' Enable hi-res, auto-inc Y
90 ' Clear hi-resolution screen
100 FOR X=0 to 79:OUT 128,X:' Output X coordinate
110 OUT 129,0:' Set beginning Y coordinate = 0
120 FOR Y=0 TO 239: OUT 130,0: NEXT:' Write zero to vertical row
130 NEXT X
```

```
140 ' Generate random X and Y coordinates for our point
150 X=RND(0)*640: Y=RND(0)*240
160 ' Plot point, loop again if no key pressed
170 GOSUB 200 : IF INKEY$="" THEN 150
180 OUT 131,252:' Put in normal display mode
190 END

200 ' Point plotting routine
220 OUT 131,255:' Put in hi-res, no auto-increment
230 X3=INT(INT(X)/8):' Get byte # in horizontal direction
240 X1=7-INT(X)+X3*8:' Get bit within byte (= remainder)
250 OUT 128,X3: OUT 129,Y: D=INP(130):' Get byte of data
260 OUT 130,D OR 2'X1:' Set bit in byte, write back
270 RETURN
```

The next program demonstrates the algorithm used to draw a line. For the sake of clarity, we will take advantage of the GBASIC CLS and PLOT functions.

```
10 ' LINE/BAS
15 ' This program demonstrates the line drawing algorithm
20 Z=&CLS+&MODE (1)
25 CLS: PRINT"Enter line endpoints X1,Y1,X2,Y2";: INPUT
X1,Y1,X2,Y2
30 Z=&ON: GOSUB 500: GOTO 25

500 ' Line drawing subroutine
505 X=X2-X1: Y=Y2-Y1
510 I=1: IF X<0 THEN X=-X: I=3
515 IF Y<0 THEN Y=-Y: I=I+4
520 IF X<Y THEN T=X: X=Y: Y=T: I=I+8
525 E=-X/2: C=0
530 IF C>X-.5 THEN RETURN
535 E=E+Y: IF E>0 THEN E=E-X: Q=I+1: GOSUB 550: GOTO 545
540 Q=I GOSUB 550
545 C=C+1: GOTO 530
550 ON Q GOTO 565,560,585,590,565,570,585,580,555,560,555,590,575,
570,575,580
555 Y1=Y1+1: GOTO 595
560 X1=X1+1: Y1=Y1+1: GOTO 595
565 X1=X1+1: GOTO 595
570 X1=X1+1: Y1=Y1-1: GOTO 595
575 Y1=Y1-1: GOTO 595
580 Y1=Y1-1: X1=X1-1: GOTO 595
585 X1=X1-1: GOTO 595
590 X1=X1-1: Y1=Y1+1
595 Z=& PLOT (X1,Y1,1): RETURN
```

It is admittedly very difficult to understand how the line drawing program works since the algorithm was chosen for its speed and not its simplicity. However, it is presented for those of you who have the desire and

determination to figure it out.

You should now know how to read and write bytes of data to any of the X (0-79) and Y (0-239) addresses and how those addresses correspond to the graphics display. What you don't realize is that Y can actually go from 0-255. So you can use all of the X values where 240<=Y<=255 to store anything you want. It is a "hidden" memory area capable of holding 1280 bytes of data.

That concludes this section of the manual. Since this is the most difficult portion of the manual, it is a good idea to go through it again and experiment with the different ideas as they are presented.

3D/ ASSEMBLY LANGUAGE TECHNIQUES

This section is meant for the experienced assembly language programmer although it may be interesting reading for anyone who at least knows what assembly language is. It is also assumed that the previous manual sections have been read and understood. The first part of this section explains how you can execute any of the GBASIC graphics routines with the proper machine language call. The last part of this section contains a machine language subroutine to clear the hi-res screen and plot points.

USING GBASIC ROUTINES

All of the graphics routines available in Extended Graphics Basic can be accessed from assembly language or alternate high-level language if the proper procedure is followed. Before any routines can be used, be sure to load but not execute GBASIC/LOD. Also note that the routines occupy memory from 60000 to 65439.

Before calling the graphics routines, in most cases it is necessary to save one or more parameters in the proper memory location(s). For example, before you can call the routine to set a point, you must save the 16 bit X coordinate value in the memory location X1POS, the 16 bit Y coordinate value in Y1POS, and the 8 bit color in COLOR. A program segment which sets a point at 255,96 is shown:

```
        LD      HL,255          ;Load X coordinate directly
        LD      (X1POS),HL      ;Save X value
        LD      HL,96           ;Load Y coordinate directly
        LD      (Y1POS),HL      ;Save Y value
        LD      A,1             ;Load color of point directly
        LD      (COLOR),A       ;Save color
        CALL    PLOT            ;Plot point
```

When initializing parameters it is the responsibility of the **calling** program to make sure that the values are within bounds. A program segment "TEST" which verifies that X ranges from 0-639, Y ranges from 0-191, and Color is from 0-2 follows:

```
; Compare HL and DE. Carry set if HL<DE
RST18H  LD      A,H             ;Compare MSB
        SUB     D
        RET     NZ              ;Return if not equal
        LD      A,L             ;Compare LSB
```

```
            SUB     E
            RET                     ;Return with status

TEST        LD      HL,(X)
            LD      DE,640
            CALL    RST18H
            JP      NC,ERROR
;
            LD      HL,(Y)
            LD      DE,240
            CALL    RST18H
            JP      NC,ERROR
;
            LD      A,(C)           ;Get color in A
            CP      0
            JP      M,ERROR         ;If less than zero, an error
            CP      3
            JP      P,ERROR         ;If greater than 2, then error
```

Alternately, you could simply load the X and Y values into XCOORD and YCOORD and make a call to the POINT routine. If the HL register has a value of 2 on return then the point is outside the current viewing area.

A list of all of the hexadecimal parameter and subroutine addresses and labels follows:

PARAMETER	V1.3 ADDRESS	V6.x ADDRESS	# BYTES	RANGE
MODE	0F2D1	0F282	1	1-4
XCOORD	0F312	0F2C3	2	0-639
YCOORD	0F324	0F2D9	2	0-239
COLOR	0F35E	0F317	1	0-2
AX2	0F62B	0FF81	2	0-639
AY2	0F62D	0FF83	2	0-239
X2	0F3B2	0F36B	2	0-639
Y2	0F3C5	0F37E	2	0-239
LINEC	0F2A4	0F255	1	0-255
BX2	0F634	0FF8A	2	0-639
BY2	0F636	0FF8C	2	0-239
RR2	0F640	0FF96	4	S.P. #
SHADE	0F051	0F000	1	0-255
XPOS	0F577	0F54C	2	0-639
YPOS	0F557	0F52C	2	0-239
TDIR	0F5D9	0F5B0	1	0-3
MODDCB	N/A	0F4F8	2	0-65536
VARADD	0F63A	0FF90	2	0-65536
FUNCT	0F640	0FF96	1	0-4
XMIN	0F316	0F2C7	2	0-639
YMIN	0F327	0F2DC	2	0-239
XMAX	0F31C	0F2CF	2	0-639
YMAX	0F32D	0F2E4	2	0-239

SUBROUTINE	V1.3 ADDRESS	V6.x ADDRESS
AON	0EBD5	0EB7D
AON1	0EBD1	0EB79
AOFF	0EBE2	0EB81
ACLS	0EBC0	0EB72
NENT	0F2CD	0F27E
XLIND	0F29D	0F24E
LINE	0F3A6	0F35F
AREV	0EC05	0EBB0
ALPRNT	0ED1E	0ECE7
ABOX	0ED4E	0ED11
ACIRCL	0EDB8	0ED6F
AFILL	0EF46	0EEF6
APOS	0ECF5	0ECC4
OLDV	0EBE6	0EB85
AGET	0F0A6	0F056
APUT	0F1B1	0F169

The graphics routine addresses and required parameters are now listed. The resulting output of all functions is identical to the output obtained when they are called from Basic. You should assume that every subroutine will destroy all registers except for the contents of HL which is always preserved. You should also assume that the parameter values (except for MODE) are no longer valid after a subroutine call.

&ON - A simple subroutine call to AON will enable the 640 x 240 (512 x 192 in Model 3 mode) hi-resolution screen with text overlay.

&ON1 - A call to AON1 will enable the 640 x 240 hi-resolution screen and turn the text screen off.

&OFF - Simply make a call to AOFF.

&CLS - Call ACLS to clear the screen.

&MODE - Rather than making a subroutine call, you should simply load the mode value into the parameter named MODE.

&PLOT (XCOORD,YCOORD,COLOR) - Once the three parameters have been loaded, just make a call to **NENT.**

&POINT (XCOORD,YCOORD) - After making a call to XLIND, the status of the point 0 for clear, 1 for set, or 2 for out of bounds is in register HL.

&LINE (XCOORD,YCOORD,X2,Y2,LINEC) - Make a subroutine call to LINE.

&REV - Calling AREV will complement the graphics screen.

&LPRINT (COLOR) - The address of this routine is ALPRNT.

&BOX (AX2,AY2,BX2,BY2,LINEC) - The parameters for this routine have the additional restriction that AX2<>BX2 and AY2<>BY2. The routine address is ABOX.

&CIRCLE (AX2,AY2,*RR2,COLOR,*ST,*EN,*AR) - The address of this routine is ACIRCL. Note that the radius should be stored as a 4 byte single precision number starting at RR2. In order to use the default parameters for ST, EN, and AR the HL register must point to a data byte equal to ')°. If you wish to use your own values for ST, EN, and AR you must point HL to the equivalent ASCII segment of the Basic command line beginning after the COLOR parameter. For example, you could have HL point to the line ,0,3.1415,3*.4)

Note: this routine uses a ROM call to perform a math function. As a result, since Basic initializes some RAM used *by this* routine it is necessary for you to execute your program from Basic.

&FILL (AX2,AY2,SHADE) - Subroutine address is AFILL. You must also load a value of 1 into the COLOR parameter.

&POS (XPOS,YPOS,TDIR) - Address to call is APOS. Before calling this function under TRSDOS 6.x, it must be initialized with the following routine:

```
LD     DE,'OD'
LD     A,82
RST     28H
LD     (MODDCB),HL
```

&RESET - Address to call is OLDV.

&GET (AX2,AY2,BX2,BY2,VARADD) - This routine is located at AGET. At memory location VARADD you should put the 16 bit address of your BUFFER which points to the first free byte in a section of memory. This section of memory must be set up to look like a single dimension integer array. The format of this memory area is as follows:

```
        DEFB    2          ;Code for an integer variable
        DEFM    'B'        ;The array name (2nd char.)
        DEFM    'A'        ; "     '°    "     (1st char.)
        DEFB    0          ;Omit this line if in Model 3 mode
        DEFW    0000       ;Offset to next variable
        DEFB    1          ;Number of dimensions
        DEFW    400        ;The array dimension
BUFFER  DEFS    800        ;Data store = 2 x dimension
```

Although the array name and offset to the next variable are "don't care values", this space must be reserved. The array dimension should be equal to or greater than INT((ABS(AX2°BX2)+1)*(ABS(AY2-BY2)+1)/16)+4 and the number of data storage bytes reserved should be equal to double the array dimension. You must be absolutely sure that the buffer is set up properly because if anything is incorrect, the subroutine will give spurious results and may even crash your program forcing you to reset the computer to regain control.

After this routine has been executed, the memory area beginning at BUFFER will contain the following data: The first two bytes contain the difference in the X coordinates; the third byte contains the difference in Y coordinates; the remaining bytes contain a continuous string of bits corresponding to the points read from the graphics screen going across each line and then down a line until all points in the specified area have been saved.

&PUT (AX2,AY2,VARADD,FUNCT) - This routine moves data stored in a buffer as defined in the &GET statement onto the screen. Once the parameters have been defined, call APUT. If you are creating your own buffer or modifying a buffer created by the &GET statement, be sure that the buffer is defined properly or the subroutine will give spurious results and may even crash your program.

&USING (XMIN,YMIN,XMAX,YMAX) - You simply load the integer parameters in the specified locations; no subroutine call is necessary.

Following is an example program which uses several of the graphics routines in order to draw a series of lines to form a pyramid. This program, which is supplied with the Grafyx board, is pretty much self explanatory.

```
; VECTORS/ASM 09/30/84
;
; DRAWS LINES GIVEN A LIST OF ENDPOINTS;
;
; NOTE: GBASIC must have been loaded before executing ; The
sample data included draws a pyramid
;
ORG              6000H
;
; The following equates must be changed if using TRSDOS 6.x
```

```
;
MODE     EQU     0F2D1H          ;Mode value
ACLS     EQU     0EBC0H          ;Clear screen routine
AON      EQU     0EBD5H          ;Enable hi-res routine
XCOORD   EQU     0F312H          ;Begin X coordinate
YCOORD   EQU     0F324H          ;Begin Y coordinate
X2       EQU     0F3B2H          ;End X coordinate
Y2       EQU     0F3C5H          ;End Y coordinate
LINEC    EQU     0F2A4H          ;Line color
LINE     EQU     0F3A6H          ;Line routine address ,
; Beginning of program
BEGIN    LD      A,1
         LD      (MODE),A        ;Set so draw single pnts
         CALL    ACLS            ;Clear hi-res screen
         CALL    AON             ;Turn hi-res board on
         LD      HL,LIST         ;Pnt to table of lines
LOOP     LD      E,(HL)          ;Get X start point
         INC     HL              ;Pnt to 2nd byte of value
         LD      D,(HL)          ;DE now has X1
         BIT     7,D             ;See if at end of list
         RET     NZ              ;If so, then return
         LD      (XCOORD),DE     ;Save X1 for line routine
         INC     HL              ;Pnt to next value
         LD      A,(HL)          ;Get Y1
         LD      D,0             ;Clear half of DE
         LD      E,A             ;Move Y to DE
         LD      (YCOORD),DE     ;Save Y1 for line routine
         INC     HL              ;Pnt to next value
         LD      E,(HL)          ;Get X2 1 byte at a time
         INC     HL              ;Pnt to next value
         LD      D,(HL)          ;Get 2nd byte of X2
         LD      (X2POS),DE      ;Save X2 for line routine
         INC     HL              ;Pnt to next value
         LD      A,(HL)          ;Get Y2
         LD      D,0             ;Clear half of DE
         LD      E,A             ;Move Y2 to DE
         LD      (Y2),DE         ;Save Y2 for line routine
         INC     HL              ;Pnt to next value
         LD      A,(HL)          ;Get color of line
         LD      (LINEC),A       ;Save line color
         INC     HL              ;Pnt to next line data
         PUSH    HL              ;Save pointer value
         CALL    LINE            ;Draw line
         POP     HL              ;Get pointer back
         JR      LOOP            ;Go see if more lines
;
; Data list of endpoints in format X1,Y1,X2,Y2,COLOR ; This
list draws a pyramid.
LIST     DEF W   220             ;These five values
         DEFB    50              ;determine one line.
         DEF W   118
         DEFB    90
         DEFB    1
```

```
;the following data should be included in the above 5 value format -
220,50,150,140,1,         221,50,151,140,1,         220,50,343,125,1,
221,50,344,125,1,         119,90,151,140,1,         118,90,150,140,1,
150,140,343,125,1
          DEF W    -1              ;This signals end of list
          END      BEGIN           ;Start execution at BEGIN
```

WRITING YOUR OWN GRAPHICS ROUTINES

In order to realize the full potential of the Grafyx board it may be necessary to bypass the supplied graphics routines and access the board directly. The organization of the Grafyx memory and mode control have already been discussed in the Advanced Programming and Mode Control section. All of the programs and information in that section applies directly to assembly language and contains everything you need to know. The purpose of this section is then not to explain anything new, but basically to present a well commented assembly language version of two previously explained subprograms. The two assembly language subroutines perform the most vital functions: clearing the hi-resolution screen and reading and writing points. This source file is also included with the Grafyx board.

```
; POINT/ASM 03/29/84
,
; SUBROUTINES WHICH WILL CLEAR THE GRAFYX SCREEN
; AND CALCULATE POINT ADDRESSES


; SUBROUTINE WHICH CLEARS THE GRAFYX SCREEN
CLEAR     LD        A,255-128-8-3    ;Starting OUT value
          OUT       (131),A          ;Output control byte
          LD        C,0              ;Reset X counter
LOOP1     LD        A,C
          OUT       (128),A          ;Write X to port
          XOR       A                ;Clear A
          OUT       (129),A          ;Set Y starting count=0
          LD        B,240            ;Number bytes Y direction
LOOP2     OUT       (130),A          ;Write data, auto inc-Y
          DJNZ      LOOP2            ;Do loop 240 times
          INC       C                ;Bump X counter LD A,C
          CP        80               ;See if done all columns
          JR        NZ,LOOP1         ;Loop if not
          RET
;
;
```

```
; SUBROUTINE TO CALCULATE DOT POSITION, TAKE ACTION
;
; On entry: (XPOS)=X-position (16 bit word)
;         0<=X<=639
;         (YPOS)=Y-position (16 bit word)
;         0<=Y<=239
;         (COLOR)=0 to clear, 1 to set
;         2 to complement, 3 to read
;
; On return:
;   If COLOR=3, zero flag, acc. are clear for zero bit;
;   if point was set, then acc. and flag are non-zero
;
POINT    LD       A,(YPOS)          ;Assume valid value
         OUT      (129),A           ;Output Y coordinate
         LD       A,243             ;Control byte no auto-inc
         OUT      (131),A
         LD       HL,(XPOS)
         LD       A,L
         AND      7                 ;Get remainder
         LD       C,A               ;Save for later LDB,0
         SRL      H                 ;Divide X by 8 RR L
         SRL      H
         RR       L
         SRL      L
         LD       A,L
         OUT      (128),A           ;Output X coordinate
         LD       HL,BTABLE
         ADD      HL,BC             ;Point to bit within byte
;
         LD       A,(COLOR)
         LD       B,A               ;Get action to perform
         IN       A,(130)           ;Read byte of data
;
; Check to see what we want to do with data
         INC      B
         DJNZ     NOT0              ;Skip if not color 0
; Clear point
         CPL
         OR       (HL)              ;Set bit high CPL
WRITE    OUT      (130),A           ;Write new data byte back
         RET
NOT0     DJNZ     NOT1              ;Skip if not color 1
; Set point
         OR       (HL)              ;Set bit high
         JR       WRITE             ;Now output data
NOT1     DJNZ     NOT2              ;Skip if not color 2
; Complement point
         XOR      (HL)              ;Complement point
         JR       WRITE             ;Now output data
```

```
; Read point
NOT2       AND      (HL)         ;Set flags
           RET
;
;BTABLE    DEFB     128          ;Table of bit values
           DEFB     64
           DEFB     32
           DEFB     16
           DEFB     8
           DEFB     4
           DEFB     2
           DEFB     1
;
;
XPOS       DEFS     2            ;Storage for X coordinate
YPOS       DEFS     2            ;Storage for Y coordinate
COLOR      DEFS     1            ;Color action parameter
           END
```

Using the above routines, it is easy to write a short program which clears the screen and places a point at 10,20.

```
           CALL     CLEAR        ;Clear the screen
           LD       BC,10        ;Load X position
           LD       DE,10        ;Load Y position
           LD       A,1          ;Set point
           LD       (XPOS),BC    ;Pass parameter for POINT routine
           LD       (YPOS),DE    ;Pass parameter for POINT routine
           LD       (COLOR),A    ;Save point color=1
           CALL     POINT        ;Plot point LD     A,255
           OUT      (131),A      ;Set to display 640 x 240 hi-res
```

That concludes this section on assembly language programming. You should now be able to write programs which interface with the GBASIC routines as well as programs which access the Grafyx Solution directly.

4/ COMPARISON WITH OTHER GRAPHICS BOARDS

This section will try and compare the hardware and software of the Model 4 Grafyx Solution board which you own to our Model 3 version of the Grafyx Solution and to the Radio Shack Models 3 and 4 hi-resolution boards.
The Model 3 Grafyx Solution has as its maximum resolution 512 x 192 dots which can be overlayed with the standard text screen. The Model 4 version has both an identical 512 x 192 mode for compatibility as well as the 640 x 240 mode. The Model 3 board is memory mapped while the Model 4 is port mapped. Therefore, software written which directly accesses the hardware using OUT or POKE statements will not transport from one board to the other without some modifications. However, the Extended Graphics Basic offered on the Model 4 is a superset of the Model 3 version for upward compatibility. The Model 4 GBASIC is also available for the Model 3 for $29.95 for cross-compatibility. The Model 3 hi-resolution files may be loaded into the higher-resolution Model 4 using the Model 4 SAVLOAD command. The Model 4 SAVLOAD program can also be used to save a section of the 640 x 240 screen in the file format used by the Model 3 version. However, the Model 4 GBASIC &LOAD and &SAVE commands work only in the 640 x 240 format.

The Model 4 Grafyx Solution hardware has both the 640 x 240 and 512 x 192 modes of operation. When in the 640 x 240 mode it works exactly like the Radio Shack boards. Therefore, any software which writes directly to the graphics boards should run on either board. The hi-resolution screen dump files made by the Radio Shack and Micro-Labs software are identical and interchangeable. However, The graphics Basics do differ in capabilities and syntax. Therefore, programs written in the respective graphics Basics must be modified. ALL of the Radio Shack commands have an equivalent in our graphics Basic. However, the converse does not hold. For example, Micro-Labs' GBASIC has commands for complementing, printing, saving and loading the hi-resolution screen while Radio Shack does not. Therefore, it is recommended that you purchase our GBASIC version for the Radio Shack board for $49.95 and never have to translate between different Basics.

5/ CONVERTING BASICG PROGRAMS TO GBASIC

Although all commercial software available supports the Micro-Labs hi-res board, some public domain programs have been written to run using the command syntax of the Radio Shack version of graphics Basic: BASICG. To use these programs with the Micro-Labs Model 4, 4D, or 4P board you can obtain a copy of the Radio Shack BASICG disk and run the program without any changes on the Micro-Labs board.

If you have the Model 3 Micro-Labs board, can not obtain a copy of the BASICG disk, or want to take advantage of the additional capabilities of GBASIC, you must change the syntax of the BASICG graphics commands to their GBASIC equivalent. For those who choose the last option, each variation of the BASICG commands are listed in alphabetical order below followed by the equivalent model 4 mode GBASIC command. The model 3 GBASIC command can be obtained by changing the @ symbol to **Z=&**

```
CIRCLE (x,y),r
   @CIRCLE (x,y,r,1)

CIRCLE (x,y),r,c
   @CIRCLE (x,y,r,c)

CIRCLE (x,y),r,c,st
   @CIRCLE (x,y,r,c,1.57,7.854-ABS(st))
   IF st<O THEN @LINE(x,y,x+COS(st)*r,y+SIN(st)*r/2,1)

CIRCLE (x,y),r,c,,en
   @CIRCLE (x,y,r,c,7.854-ABS(en),7.854)
   IF en<O THEN @LINE(x,y,x+COS(en)*r,y+SIN(en)*r/2,1)

CIRCLE (x,y),r,c,st,en
   @CIRCLE (x,y,r,c,7.854-ABS(en),7.854-ABS(st))
   IF st<O THEN @LINE(x,y,x+COS(st)*r,y+SIN(st)*r/2,1)
   IF en<O THEN @LINE(x,y,x+COS(en)*r,y+SIN(en)*r/2,1)

CIRCLE (x,y),r,,,,ar
   @CIRCLE (x,y,r,l„ar)

CIRCLE (x,y),r,c,,,ar
   @CIRCLE (x,y,r,c,,ar)

CIRCLE (x,y),r,c,st,,ar
   @CIRCLE (x,y,r,c,1.57,7.854-ABS(st),ar)
   IF st<O THEN
   @LINE(x,y,x-COS(st)*r/(ar*(ar>1)+(ar<=1)),y-SIN(st)*r*(ar*(ar<=1)+
(ar>1)),1)

CIRCLE (x,y),r,c,,en,ar
   @CIRCLE (x,y,r,c,7.854-ABS(en),7.854,ar)
   IF en<O THEN
   @LINE(x,y,x-COS(en)*r/(ar*(ar>1)+(ar<=1)),y-SIN(en)*r*(ar*(ar<=1)+
(ar>1)),1)
```

```
CIRCLE (x,y),r,c,st,en,ar
   @CIRCLE (x,y,r,c,7.854-ABS(en),7.854-ABS(st),ar)
   IF st<O THEN
   @LINE(x,y,x-COS(st )* r/ (ar *(ar> 1)+(ar<=1)),y-SI N(st )* r*(ar
*(ar<=1)+(ar> 1)),1)
   IF en<O THEN
   @LINE(x,y,x-COS(en)*r/(ar*(ar> 1)+(ar<=1)),y-SIN(en)*r*(ar*(ar<=1)+
(ar>1)),1)

CLR
   @CLS

GET(xl,yl)-(x2,y2),z
   @GET(xl,yl,x2,y2,z96(0))

GLOCATE(x,y),d
   @POS(x,y,d)

LINE(xl,yl)-(x2,y2)
   @LINE(xl,yl,x2,y2,1)

LINE(xl,yl)-(x2,y2),c
   @LINE(xl,yl,x2,y2,c)

LINE-(x2,y2)
   @LINE(x2,y2)

LINE-(x2,y2),c
   @LINE(x2,y2)

LINE(xl,yl)-(x2,y2),,B
   @BOX(xl,yl,x2,y2,1)

LINE(xl,yl)-(x2,y2),c,B
   @BOX(xl,yl,x2,y2,c)

LINE(xl,yl)-(x2,y2)„BF
   @USING(xl,yl,x2,y2,3):@USING(0,0,639,239)

LINE(xl,yl)-(x2,y2),c,BF
   @USING(xl,yl,x2,y2,2+c):@USING(0,0,639,239)
```

In the following 4 equivalences there is not a direct match of the line style variable st with the line color variable c. You just need to match each unique line style with a unique line color from 3 to 255.

```
LINE(xl,yl)-(x2,y2),,,st
   @LINE(xl,yl,x2,y2,c)

LINE(xl,yl)-(x2,y2),c„st
   IF c=0 THEN @REV:@LINE(xl,yl,x2,y2,c):@REV
   IF c=1 THEN @LINE(xl,yl,x2,y2,c)
```

```
LINE(xl,yl)-(x2,y2)„b,st
   @BOX(xl,yl,x2,y2,c)

LINE(xl,yl)-(x2,y2),c,b,st
   IF c=0 THEN @REV:@BOX(xl,yl,x2,y2,c):@REV
   IF C=1 THEN @BOX(xl,yl,x2,y2,c)

PAINT(x,y)
   @FILL(x,y,255)

PAINT(x,y),c
   IF c=0 THEN @REV:@FILL(x,y,255):@REV IF
   c=1 THEN @FILL(x,y,255)
```

In all of the remaining variations of the Paint command there is not a direct match of the tiling pattern being generated with the fill color variable c. You just need to match each unique tiling pattern with a Fill color from 0 to 255.

```
PAINT(x,y),ti$,bd,ba
    @FILL(x,y,c)

Z=&POINT(x,y)
    Z=&POINT(x,y):IF Z=2 THEN Z=-1

PKESET(x,y)
    @PLOT(x,y, 0)

PRESET(x,y),c
    @PLOT(x,y,c)

PRINT #-3
    PRINT

PSET(x,y)
    @ PLOT(x,y,l)

PSET(x,y),c
    @PLOT(x,y,c)

PUT(x,y),z
    @PUT(x,y,z%(0),4)

PUT(x,y),z,PSET
    @PUT(x,y,z%(0),1)

PUT(x,y),z,PRESET
    @REV:@PUT(x,y,z%(0),1):@REV

PUT(x,y),z,XOR
    @PUT(x,y,z%(0),4)
```

```
PUT(x,y ),z,OR
    @PUT(x,y,z%(0),3)

PUT(x,y),z,AND
    @PUT(x,y,z%(0),2

SCREEN 0
    @ON1

SCREEN 1
    @OFF

SCREEN c
    IF c=0 THEN @ON1 ELSE @OFF
```

For the following 4 equivalences note that the @CLS command ignores the new boundaries and the statement @USING(xl,yl,x2,y2,2) must be used instead. Also, the VIEW command changes the origin of 0,0 from the upper, left corner to xl,yl so an offset of xl,yl must be added to all subsequent graphics commands.

```
VIEW(xl,yl)-(x2,y2)
    @USING(xl,yl,x2,y2)

VIE W(xl,yl)-(x2,y2),c
    @USING(xl,yl,x2,y2,c+2)

VIE W(xl,yl)-(x2,y2)„b
    @USING(x1,yl,x2,y2,b)

VIEW(xl,yl)-(x2,y2),c,b
    IF b+c=0 THEN @USING(xl,yl,x2,y2,2)
    IF b+c=2 THEN @USING(xl,yl,x2,y2,3)
    IF b=1 AND c=0 THEN @USING(x1,yl,x2,y2,2):@BOX(xl,yl,x2,y2,1)
    IF b=0 AND c=1 THEN @USING(x1,yl,x2,y2,3):@BOX(xl,yl,x2,y2,0)

Z=&VIEW(f)
f=0 returns the upper, left X-coordinate; f=1 the upper, left Y-coordinate
f=2 returns the lower, right X-coordinate; f=3 the lower, right Y-coordinate
    There is no direct equivalent for this command but it is not needed if
you store the boundary parameters used by the USING command in some
permanent variables which are updated each time you use the USING command.

SYSTEM"GLOAD filename"
    OPEN"R",1,"filename":@LOAD:CLOSE

SYSTEM"GPRINT"
    @LPRINT(0)

SYSTEM"GSAVE filename"
    OPEN"R",1,"filename": @SAVE:CLOSE
```

6/ THEORY OF OPERATION

If you're really interested in understanding all of the details, it will help to read the video display section in the Model 4 Technical Manual (available from Radio Shack catalog number 26-2110) before continuing.

Adding the Grafyx board expands the graphics capabilities of the TRS-80 to give a maximum resolution of 640 horizontal by 240 vertical (512 horizontal by 192 vertical in an optional display mode). The hi-resolution display, when enabled, actually overlays the normal screen display when they are both using the same resolution. In other words, when you are in Model III 64 character mode, the 64 characters x 8 dots equals 512 dots in the X direction, so characters can be overlayed with only the 512 x 192 graphics mode. In the Model 4 80 column mode, the character text will overlay with the 640 x 240 graphics mode. This allows you to easily mix text and graphics. An interesting feature of the graphics screen is that it will remain fixed, even if the normal character lines are being scrolled.

To achieve this resolution, the Grafyx Solution contains its own 20K bytes of memory made up of ten 2K x 8 static memory chips. This memory is accessed through three port addresses. Output port 128 is used for the X coordinate address (0-79). Port 129 is for the Y coordinate address (0-239). These coordinates are latched into four 74LS191 up/down counters on the Grafyx board. When you read or write data to the I/O port 130, the outputs from these counters are presented to the memory chips as the data address and the data is passed through the 74LS245 bi-directional buffer.

The Grafyx board is controlled by sending data to the write only port number 131. The control bits which specify the operating parameters of the Grafyx board are latched into part 74LS273 where they remain the same until changed. However, when the reset button is pressed, this latch is set to all zero which is why it disables the hi-resolution display.

When the graphics RAM is not being accessed, the two 74LS393 counters sequence through the whole range of memory chip data addresses. This presents the appropriate sequence of data bytes to the 74LS166 chip which converts them from 8 bits parallel to 8 serial bits. These output data bits are then XORed with the standard TRS-80 video data stream to get the resulting hi-resolution and text display.

7/ OTHER PRODUCTS AVAILABLE

Although we have attempted to include much of the software that you might need, there are countless more programs to be written. If you feel that you have written a program which takes advantage of the hi-resolution graphics and would be of use to others, please send us a copy and we will review it for possible sale on a royalty basis.

A list of the programs currently available for use with the hi-resolution board follows. Unless otherwise noted, all programs are supplied on a diskette containing both Model 3 and 4 mode versions. Orders should be sent to Micro-Labs, 902 Pinecrest, Richardson, TX 75080. Shipping is free on pre-paid orders.

BIZGRAPH by Ted Carter $75.00

This is a complete self-prompting data entry and plotting package designed to be easily used by anyone. It is perfect for managers, small businessmen, and analysts. The Bizgraph package can display a Line Graph, Bar Chart, Stacked Bar Chart, Pie Chart, Area Plot, Histogram Plot, or Scatter Plot. The resulting plots may be saved, chained, or printed. Not only can you plot data entered from the keyboard, but you can also plot data created with the Radio Shack Visicalc package or any other program which saves data in the DIF standard Data Interchange Format. Multiple data sets can be combined on one graph. Bizgraph provides automatic labeling of X and Y axis points using 80 characters per line. Forecasting future trends is possible using line fitting, quadratic, and third order linear regression analysis. Data smoothing using moving averages is also possible.

DRAW by Ted Carter $39.95

This program is essential for any application since it is a complete graphics/text editing package. This program allows you to easily draw a picture or design a graphics screen. By moving a cursor with the arrow keys and entering one letter commands, you can set, clear or complement points, lines, circles, ellipses, arcs, or boxes. Portions of the screen can be filled in with any of 256 patterns. Text labels can be placed anywhere on the screen. Cursor size and speed can be changed at any time. The screen can be shifted in any

direction, reversed, or complemented. Rectangular sections of any size can be saved to create custom symbol sets and figures for later recall using any of five logical functions. And of course you can save copies of the screen or obtain a hardcopy printout.

SURFACE PLOT II by Rapidynamic Software $39.95

Surface Plot lets you create amazing three-dimensional views by entering equations of the form Z=F(X,Y) where Z is the height above the surface for a given X, Y coordinate. For example, entering the equation Z=10-X*X draws a rectangular contour surface resembling a hill. The final picture can be viewed from any position in space so you can see an image from underneath, above, or even inside a hill or valley on the plot surface. The program automatically removes hidden lines. Complete instructions and sample equations are supplied so that you will have your computer hard at work without delay. The finished plot can be saved on disk or printed. This program has been re-written in machine language for an improvement in plotting speed of 10 to 60 times!

3D-PLOT by Ronald Hoard $39.95

This is a vector oriented package which accepts data in the form of X, Y, Z coordinates. This allows you to create complex three-dimensional figures and shapes which are composed of connected lines. These shapes may then be rotated, reversed, scaled, transformed in any dimension, placed anywhere on the screen, and viewed from any distance. A limited hidden line removal option is also supported. Screens and shape tables can be saved on disk. Your results may also be sent to a printer with graphics capabilities.

MATHPLOT by Ronald Hoard $39.95

This program allows you to plot your science, math, and engineering equations. Up to ten functions of the form Y=F(X) may be graphed. As an added feature, you may have the program calculate either the integral or derivative of your equation(s). These results are then manually or automatically scaled so that the maximum number of points fit on the screen. You may then print or save the finished graph. Hand entered data may also be shown on a line or scatter graph. As a bonus, you also get a program which allows you to make plots using polar notation.

xT.CAD by Microdex $345.00

xT.CAD is intended for professional technical drafting. Single keystroke commands create drawing components (lines, circles, arcs) directly on the screen. Any component can be erased and/or replaced by another. Drawings can be made to any arbitrary scale as well as a number of standard scales. Small, intricate details can be created by "zooming-in" on any section of the picture. The drawing can be split into overlays and text may be placed anywhere on the drawing. Although a dot matrix screen dump is supported, the program is oriented towards producing a high-quality, proportional, and scaled output on any of the Houston Instruments plotters. More detailed information on this program and plotters is available on request. A manual for review is $15.

Tournament CHESS by Rapidynamic Software, Inc. $49.95

Play against the computer or a friend in this classic game. The chess board and pieces are shown in amazing detail using the hi-res display. This program provides complete game control with over 40 execution options and features. It responds so quickly that it can play Rapid Transit speed Chess. Its play is so powerful that it can beat any other TRS-80 computer Chess game on the market. Tournament Chess has an opening book containing over 2000 moves and its knowledge base is so extensive that it can play all opening styles, middlegame attacks, and endgame strategy equally well. You can select any of 10 skill levels as well as how passive or aggressive the computer plays.

Tournament REVERSI by Rapidynamic Software, Inc. $29.95

This program plays the game of Othello while using the hi-res display to show a clear, detailed view of the playing board and pieces. Complete game control is provided with over 20 execution options and features. All of the following playing modes are supported: adaptable tournament level, 10 practice levels, win finder, best win finder, computer vs player, player vs player, and game review. The ability of the program to swap sides, suggest moves, test moves, view legal moves, and take back moves help you to quickly improve your game play.

Move-Perfect 3-D TIC-TAC-TOE by Rapidynamic Software, Inc. $19.95

This is one of the first programs to play a perfect game of Tic-Tac-Toe using a 4 x 4 x 4 playing board. Moving first at its highest skill level, it

always wins. It features powerful play and superfast speed - it responds in under 10 seconds for 90% of its moves. It also provides ten skill levels so that anyone from beginner to expert can have a good game. The playing board matrix is shown in simulated 3D using the hi-res display. Over 20 execution options and features give you complete game control. Example features include the ability to turn the cube in any of three dimensions, swap sides, suggest moves, display the game history, or back up any number of moves.

DOG CATCHER by Rapidynamic Software, Inc. $19.95
You have to see the cartoon-like animation in this real-time arcade game to believe that it is possible. The arrow keys are used to move your dog across a proportionally moving background. The object is to find the bone and return to the dog house before one of the dog catchers gets you. A unique feature of the game is an instant replay of the last round. You can also save the round to disk for later viewing so that you can analyze your own play and use it to improve or show off your skill to a friend. (For Model 4/4P only.)

GPRINT by Rapidynamic Software, Inc. $24.00

This program allows you to create your own customized hi-res screen dump utility. So if your printer supports hi-res dot graphics, but is not one of those supported by Gbasic, then this program solves your problem. However, the flexibility of this program makes it useful to everyone. You can specify that the printout be horizontal or vertical. You can specify the number of times each horizontal and vertical dot is sent to the printer to create printouts from very small to very large. You can specify that only a section of the hi-res screen be printed. This allows you to enlarge a section of the screen and even print pictures larger than a single sheet of paper. Sample screen dump routines are supplied for some of the more common printers. Note that the screen dump function operates as a DOS command and does not tie into Basic.

ZBASIC 3.0 by Zedcor, Inc. $89.95 for each computer version
This is a compiled graphics Basic which supports hi-resolution graphics and uses the same programming commands regardless of the computer and runs faster than any other Basic. Versions are available for the TRS-80 Models 3, 4, CPM 80 versions 2.2 and 3.0, IBM-PC, PC/XT, PC/AT and compatibles, Apple IIc and IIe, and Macintosh. In addition to machine interchangeability, Zbasic

includes device independent graphics (graphic commands are exactly the same on all computers), up to 54 digit numeric accuracy (selectable by the programmer), a built-in interactive editor and compiler, a choice of alphanumeric labels or line numbers, structured programming constructs, the same file commands on all computers and much more. In addition there are no licensing or royalty fees for programs written with Zbasic.

SLIDESHOW by Rapidynamic Software, Inc. $19.95
This is a disk of utilities which take advantage of the supplied Rapidos operating system to perform extremely fast hi-res screen operations. Programs supplied allow you to clear, view, load, save, or merge pictures, reverse the screen or add text to the screen. There are also automatic and manual slide show programs which will step through and display a list of hi-res files.

BASICE by Micro-Labs, Inc. $39.95 or $19.95 to GBASIC 3.0 owners
This utility adds the Micro-Labs graphics Basic hi-res screen print and screen complement commands to the Radio Shack version of graphics Basic: BASICG. Both model 3 and 4 versions are supplied.

BASIC-PLUS by Micro-Labs, Inc. $8.00
This is a version of the standard Graphics Basic which works with TRSDOS 6.2 which has the 6.2 PLUS file BE1/CMD appended to Basic.

PCHAR by Ted Carter $14.95
This package of programs turns the Grafyx Solution into a programmable character board. Up to 160 characters can be edited using the standard 8 x 8 dot matrix character format. Once a character set is created, the GBASIC POS command is used to place them anywhere on the screen. New character sets can be saved on disk.

LET'S WRITE MUSIC by Earl Grimmett $49.95
This program allows both the amateur or professional musician to put his composition on a computer screen. Here it can be altered, printed on most dot matrix printers, or stored and retrieved from disk. A full line of music is displayed at one time, showing both the treble and bass staffs. The program supports five different time values and rests. Much effort has been made to provide most of the signs and symbols used in writing and composing music.

This program allows you to make neat, readable, professional looking written music and also has many educational uses.

GW-CONVERT by Dennis Allen $99.95

This program accepts any GW-BASIC program written for MS-DOS and IBM PC-DOS machines and compatibles like the Tandy 1000, 1200, or 2000. It then translates all of the commands to the equivalent TRS-80 Model 4 Basic command. Graphics commands are translated to the appropriate Micro-Labs Graphics Basic command. Note that this program accepts as input a TRSDOS 6.x ASCII text file and you would need another program such as Powersoft's SuperCROSS utility to copy the file off of an IBM formatted disk.

BIORYTHM and **USA** by Harley Dyk $19.95

The BIORYTHM program plots your emotional, physical and intellectual states given your birthdate and current date. The result is shown in hi-resolution and you can print the results for a permanent record.

The USA program is designed to help one to learn to identify and correctly spell the 50 states and capitals of the United States with the help of a high resolution map. One may study states or capitals, in either the multiple choice or spelling mode. All questions are randomly selected for a different sequence each time.

MICRO-LABS, INC.
902 Pinecrest Drive
Richardson, TX 75080
(214) 235-0915 Phone orders welcome.

☐ Mod I ☐ Mod III ☐ Mod 4
☐ Mod 4P Other _____

PAYMENT:
☐ Check or Money Order
☐ C.O.D.
☐ VISA ☐ Mastercard Exp. date_____
Card # _____
☐ P.O. # _____
(schools and approved companies only)
TERMS: Net 30 FOB: Richardson, TX

***SHIPPING:** We pay shipping on pre-paid orders. VISA, Mastercard, COD, dealer, Net 30, and quantity discount orders add $3.00 for orders under $100, $4.50 otherwise. UPS 2nd Day Air is $4.00 additional.

Outside U.S.A.: Remit U.S. funds only. Add additional shipping.

ORDERED BY:
Name _____
Firm _____
Address _____
City _____
State _____ Zip _____

SHIP TO (if different):
Name _____
Firm _____
Address _____
City _____
State _____ Zip _____

Qty.	Order No.	Product Description	(T)ape/ (D)isk	Unit Price	Total Price

Subtotal	
Tex. residents add 5⅛% tax	
Shipping and Handling*	
TOTAL	

Phone # _____

Signature _____

Registration # 5772

PRODUCT REGISTRATION FORM

Please fill out and return the following information to Micro-Labs, Inc., 902 Pinecrest, Richardson, Texas 75080. This will help insure that you receive product updates and information on new products.

Name _____

Street _____

City, State, Zip _____

No. Disk Drives ____ **Memory** ____**K** **DOSes Used** _____

Speed-up Kit? _____ **Hard Drive?** _____ **Printer Model** _____

Other Enhancements _____

Desired Future Products _____

_____

Product Purchased From _____