- FORWARD -

Welcome to the exciting world of The Grafyx Solution! (tm)

Your TRS-80 (tm) will be transformed into a powerful new system, with graphics capabilities far beyond your dreams after you've installed your Grafyx (tm) 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 addition, you get the benefit of being able to display eighty columns of characters for business forms and word processing.

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 of the Board..... 2

2. Installation and Checkout..... 3

3. Theory of Operation..... 17

4. Using The Grafyx Solution

   A. Adding Graphics to Basic Programs..... 19
   B. Using The 80-Column Program..... 28
   C. Using The Supplied Programs..... 31
   D. Advanced Programming & Mode Control... 39
   E. Assembly Language Techniques..... 46

5. Other Software Available..... 54

1/ A BRIEF DESCRIPTION

The Grafyx Solution gives any configuration of a TRS-80 Model III computer a resolution of 512 x 192 for a total of 98,304 individually accessible points. The included graphics package allows you to set and reset points, lines, circles, figures or boxes and complement or clear the screen using simple Basic commands. Alternate resolutions of 256 x 192, 128 x 192, and 128 x 96 are also possible.

The ability to add impressive and informative graphics with a resolution better than even the Apple opens up many new possiblities for business, educational, and recreational programs.

Software support and documentation for the Grafyx Solution is also impressive. An eighty character display for business forms and word processing is possible with the supplied COLUMN80 program. In addition to a number of instructive and interesting demonstration programs, 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: Epson MX70, MX80 with Graftrax, MX100, RS LPVII, LPVIII, DMP-100, 200, IDS 445G, 460G, 560G, Prism, Okidata Microline 82A, 83A with Okigraph, 84, Anadex 9500, 9501, Seikosha (Axiom) GP-100, Centronics 739, C. Itoh 8510A, and NEC PC-8023.

When enabled, the hi-res graphics screen is displayed on top of the normal character display. The Grafyx Solution contains 12,288 bytes of additional read/write memory that doesn't conflict with the TRS-80 address space. Upgrading your computer requires no soldering and is as easy as clipping on some micro-clips, cutting one trace, and removing two IC's from sockets.

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 III Computer before, doing so will void your limited 90-day warranty, and may make your Repair Center a little upset if anything goes wrong - so you should read through the installation procedure and make sure that you understand what's involved. If you don't feel up to the task of moving two socketed integrated circuits and cutting one trace, consider getting help from someone with electronics experience. Your local computer or ham club is a good place to start, and if that option is not available, a electronics repair shop will usually be happy to do the job for you.

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 III case.) Next, assemble the following tools:

- 1 Phillips head screwdriver
- 1 X-acto knife
- 1 integrated circuit puller or small screwdriver
- 1 hacksaw (optional)

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 serious 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 probably be looking at the RF shielding; a flat metal panel with slits in the top. (Computers manufactured before January 1981 do not comply with FCC emission guidelines and do not have this panel.) To remove this panel and expose the main computer board, you must 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 should now be held in place only by the thick aluminumn foil stuck to the bottom of the panel. Carefully peel this foil off without tearing it and remove the panel.

5. Locate the two integrated circuits U67 and U81. These IC's should be easy to find since, unlike most of the other IC's, these two are in sockets.

5a. Unless you have the J.E.S. Graphics 4MHz clock speedup board installed, go on to step 6. Otherwise, your actions in step 6 will be slightly modified. When you remove the 20 pin IC from U67 in step 6, unsolder the wire from pin 19 which is presently going to the J.E.S. board. Then bend pin 19 back in place before inserting the chip in the 80-GRAFIX board as explained in step 6. Pin 19 of the red 20 pin IC header located on the bottom of the 80-GRAFIX board should now be bent up and the wire from the J.E.S. board should be attached to it. An alternate way which is recmmended if you can leave the metal shield off is to insert a level of low-profile sockets between the 80-GRAFIX board and the TRS-80 board. You can then bend up and solder to pin 19 of the socket instead of the 80-GRAFIX connector. Make sure that you know what you are doing since you may not return a modified board.

6. Pick up the Grafyx board and hold it so that the writing is right side up, and you can see how it will plug into the two sockets which presently contain the IC's mentioned in step 5. You may want to refer to Figure 1 which shows the board completely installed. Now, paying careful attention to the orientation of the two IC's, remove them from the CPU board, and plug them into the corresponding sockets on the Grafyx board. This step must be done with great care in order to keep from bending or breaking any of the pins on the IC's. If you have an IC puller, place it over the IC and rock it back and forth until the IC comes out of the socket. If you don't have an IC puller, use a metal nail file or a small flat blade screwdriver to pry up first one side of the IC and then the other until the IC is clear of the socket. When plugging the IC's into the Grafyx board,

make sure that none of the pins have been bent and that they all match up with the corresponding holes in the IC sockets.

IMPORTANT - DO NOT press the IC's into the Grafyx board without firmly supporting the board so that no pressure is placed on the pins protruding from the bottom of the Grafyx board. Bending any of these pins will weaken them and may cause problems later.

After inserting the IC's, make sure that the pins protruding from the bottom of the board are all straight. If they aren't, carefully straighten them with needle-nose pliers.

7. You should now double check the orientation of the IC's which you moved to the Grafyx board. The top of an integrated circuit is the end with and indentation of some sort in it. Holding the Grafyx board so that the name is right side up, the two IC's which you inserted should have their tops nearest the top of the board.

8. There is one trace on the main TRS-80 computer board which must be cut with an X-acto knife. The trace is located above and to the right of IC U82 and is shown in Figure 2. In order to insure that the trace is completely cut, it is best to make two cuts very close to each other (about 1/16") and scrape off the portion of the trace in between the two cuts. Be careful not to damage any of the traces close to the ones being cut. There is a second trace cut which is optional. Performing this modification prevents clear streaks from interfering with the Grafyx Solution display during screen updates. The interference is not too noticable unless you have a white on black display so you might want to skip this step until you have everything else working properly. The second optional trace cut is performed in the same manner as the first one. It is the trace running down from pin 12 of IC U17 and is shown in Figure 4.

9. Now that that's done, the next step is to install the Grafyx board. First, fasten the yellow micro clip to pin 13 of IC U52 as shown in Figure 4. Try to place all of the micro-clips as close to parallel with the CPU board as possible. Be sure that the clip is firmly attached and does not touch any adjacent pins on the IC. Be careful not to pull on the micro clips since they are easily bent out of shape. You should next fasten the green micro-clip to pin 9 of IC U55.

Before proceeding, you need to determine what version of the TRS-80 board you own. Some computers manufactured after August 1982 have a minor modification which affects the way the Grafyx board is installed. The affected computers are those which have a G

following the seven digit number just above IC #U4. If this is not the case, go on to step 9a. The affected computers have three inductors; L2, L3, and L4 just to the right of IC U95 which stand up about 1/2 to 5/8 of an inch above the board and prevent the Grafyx board from being pressed all the way into the empty IC sockets.

There are several ways around the problem. One is to sandwich an 18 and 20 pin socket between the Grafyx and TRS-80 boards, thus raising the Grafyx board above the inductors. If you can bend the inductors so that they are no more than 1/2 and inch above the board, then only low profile sockets are needed; otherwise you need the non-low-profile kind. Either may be obtained from Micro-Labs at no charge. If this approach is used, be sure and place electrical tape over the top of the inductors so that they do not short out to the bottom of the Grafyx board.

The following solutions are recommended only for those with soldering expertise. The inductors were added to reduce RF interference when using the parallel printer port and are not essential. Therefore, it is possible to remove the inductors and place solid wire jumpers in their place. You might also be able to cut the inductors in half or use longer leads and lay the inductors down.

9a. Now, very carefully line up the Grafyx board with the two empty sockets. See Figure 1 for a picture showing how the board should look. Note that the board goes under the five wires from the cassette connector. You can now gently work the Grafyx pins into the sockets a little bit at a time, supporting the main TRS-80 circuit board with your hand as much as possible. If the pins will not go into the sockets without excessive pressure, you will need to "break in" the two IC sockets. This is done by inserting and removing a straight pin several times in each of the empty socket holes. After this is done, you will find that the Grafyx board will press all the way into the sockets without excessive force. Once the board has been plugged in, adjust the light so that you can look at it from the side and make sure that the board is plugged ALL THE WAY IN. If not, either apply more pressure or remove the board and "break in" the sockets before re-inserting.

10. Now all you have to do is connect the remaining micro-clips. The brown micro-clip is unique in that it clips onto a wire instead of an IC. The wire, shown in Figure 3, is in the second slot of the rightmost connector at the top of the TRS-80 main board. This wire should be pulled up OUT OF THE CONNECTOR and one-fourth an inch of

insulation removed from the end using a X-acto knife or wire strippers. The brown clip is then clipped onto the end of the wire and placed in a secure position behind the main board. Do not put the wire back in the connector. You should also place some electrical tape over the connection so that it does not accidentally make contact with the metal brackets. Be sure to keep the plastic clip away from the socketed memory in that corner since some brands of IC's get quite warm.

11. As shown in Figure 4, the red clip should be attached to pin 8 of IC U33. The orange clip goes to pin 12 of IC U18. The fourteen pin ribbon connector will now be attached to IC U20. A piece of double-stick tape is stuck to the bottom of the connector. An extra piece of tape is enclosed in case you remove the connector and want to re-attach it with a new piece of tape. Now remove the protective backing from the tape on the connector and press it firmly down onto the IC, rocking it back and forth to get it started. Make sure that all leads make contact with their respective IC leads and are not touching any other leads.

    In all of our tests, this type of connection has proven reliable. It is therefore not recommended that the conector be soldered directly to the IC since it makes the connector difficult to remove, and the IC may be damaged if overheated. course, if you have a lot of experience with this type of work, you may prefer to solder a low-profile socket to IC U20, and plug the connector into it.

12. The following connections are illustrated in Figure 4: The blue clip should be attached to pin 7 of IC U41. The purple clip goes to pin 4 of U57. The connection of the grey clip depends on whether or not you performed the optional trace cut in step 8. If you did make the modification, the grey clip should be connected to pin 12 of U17. If you did not, or have not yet made the second trace cut, DO NOT connect the grey clip. Make sure that it does not touch anything by putting electrical tape over the end of the clip or by cutting the clip off at the board. Lastly, the white clip should be connected to pin 13 of U57.

13. Now for the moment you've been waiting for!
    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 III display should come up just as it did before you installed The Grafyx Solution, with the date prompt for Disk systems or the 'Cass?' message for tape systems. If anything looks out of sorts, immediately turn off and unplug the computer! Go to the end of this section and follow the instructions for

trouble shooting your installation.

14. If everything looks normal, you can proceed to load and execute the machine language GTEST program as explained in section 4C. The GTEST program will verify every memory location in the Grafyx board and alert you to any problems. In case there is a problem, the GTEST program also displays the graphics screen so that the error can be pinpointed. If an error does occur, or the display looks incorrect, unplug the computer, and go to the trouble shooting instructions at the end of this section.

15. Assuming that everything now works, you can go ahead and finish the job by putting your Model III back together. Just to be safe, be sure to unplug your computer and wait 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 because it hits the top of the Grafyx board. You will need to try it out since clearance is not a problem on later model computers. If there is a problem, you will probably want to modify the shield so that it can be put back in place and reduce RF interference although it is not necessary for proper electrical operation. If you do not need to modify the shield, or you're computer didn't come with one, you can go on to step 16.

    The two cuts shown in Figure 5 should now be made in the shield with a hacksaw. Be sure and perform this operation AWAY from your computer and make sure that you wipe off the resulting metal particles. You may also need to use an X-acto knife to remove the burrs from the saw cut edges. The next step is to lay the shield flat on a table with the edge of the table being six inches from the bottom of the shield and push down to bend the shield. The section over the Grafyx Solution should be bent so that it is two inches higher than the rest of the shield at the bottom end. The shield can then be put back on over the CPU board if you press on the bottom of the protruding panel when installing it so that it is bowed. The section right over the Grafyx Solution will be about a fourth of an inch higher than before which should be just enough to clear the highest points on the Grafyx board. You should be able to replace all of the screws holding the shield on except for the one on the bottom side nearest the Grafyx board. Finally, stick the thick aluminumn foil to the bottom of the shield, as it was before. You may need to replace the tape holding it on for it to stay in place, although it really isn't too important.

    You should now go back and repeat step 14 to make sure that you didn't bump anything while putting the shield on.

16. The installation operation is completed by carefully placing the Model III case top over the case bottom, making 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 rear panel of the case. Holding the two halves of the case together with your hands, turn the Model III 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.

17. 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. Assuming that everything is okay, skip the following material and go on to the next section.

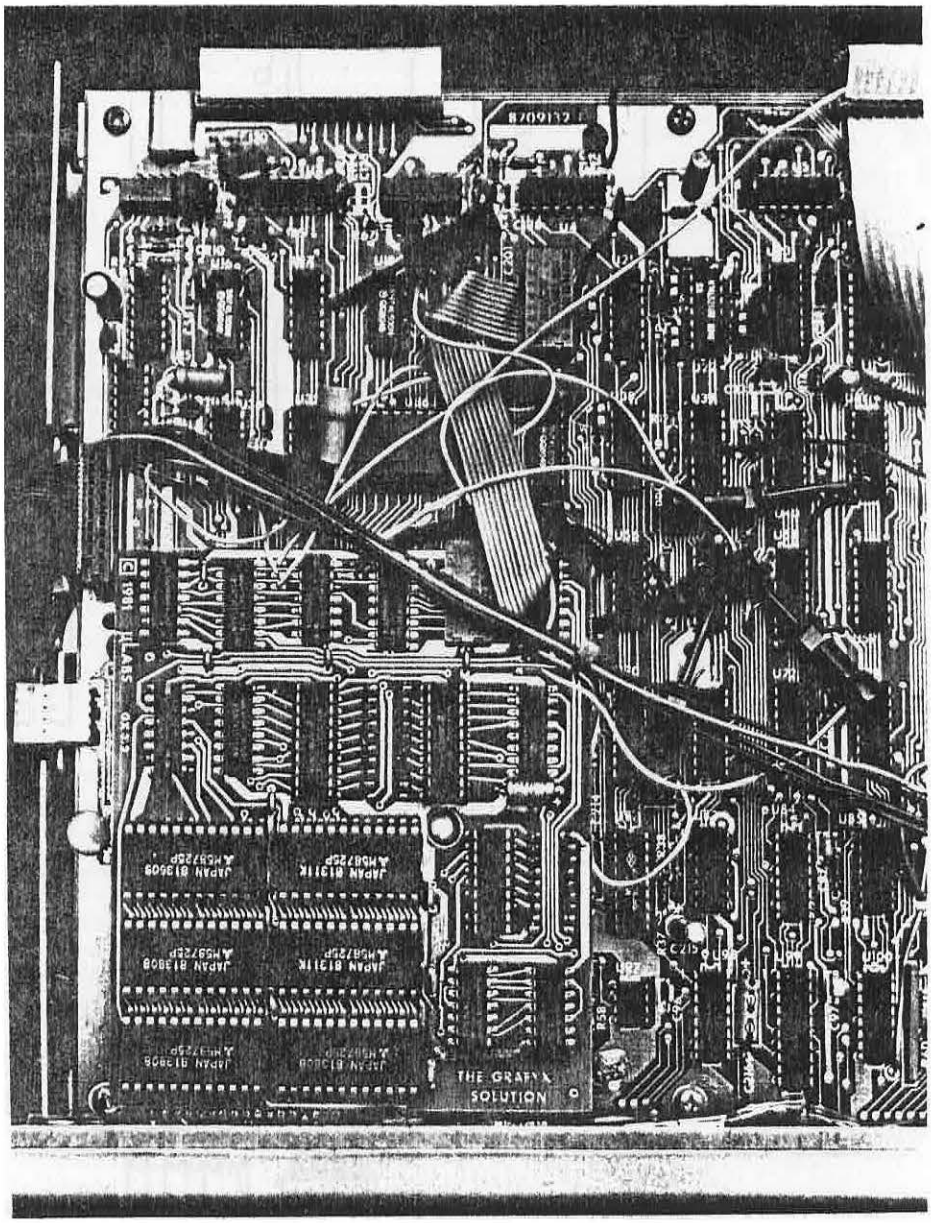


Figure 1 Installed Grafyx Board

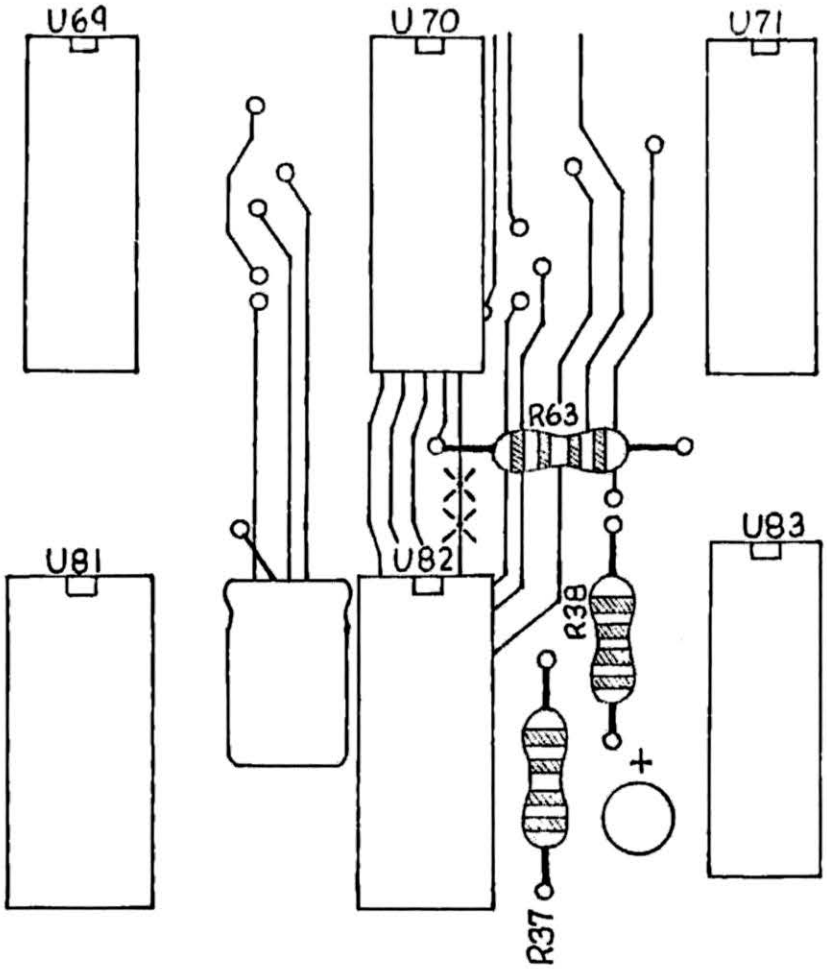


Figure 2. POSITION OF TRACE CUT

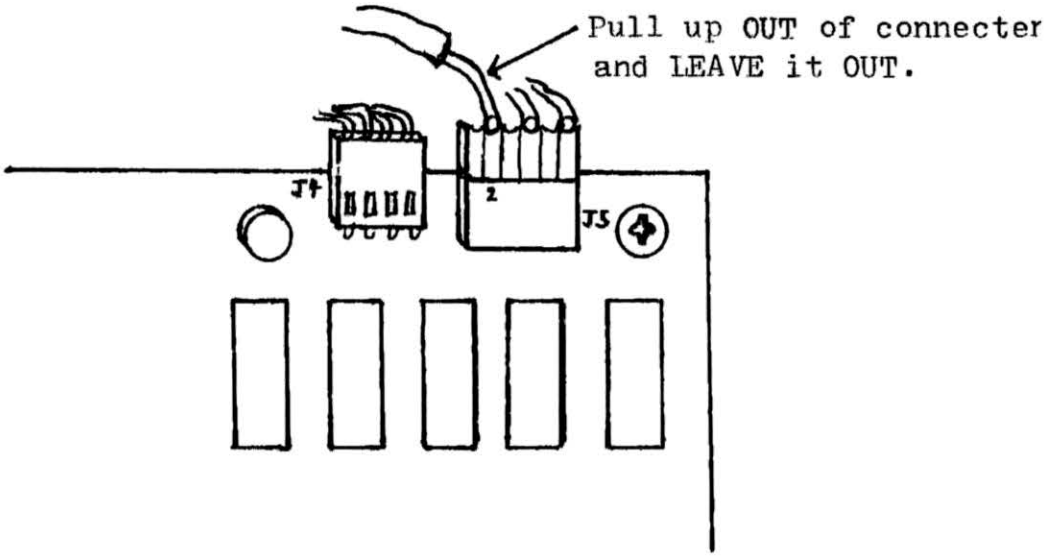


Figure 3.

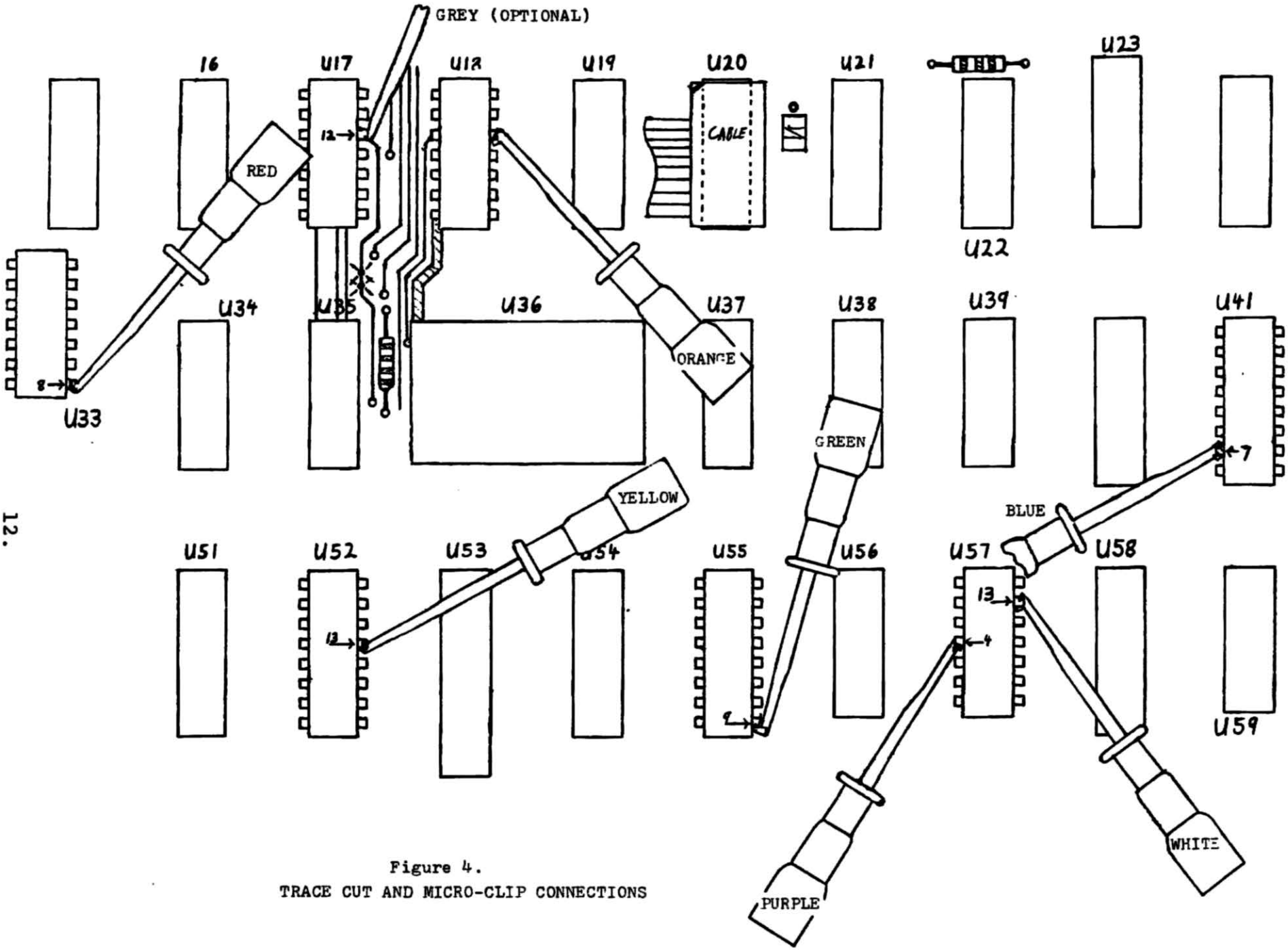


Figure 4.
TRACE CUT AND MICRO-CLIP CONNECTIONS

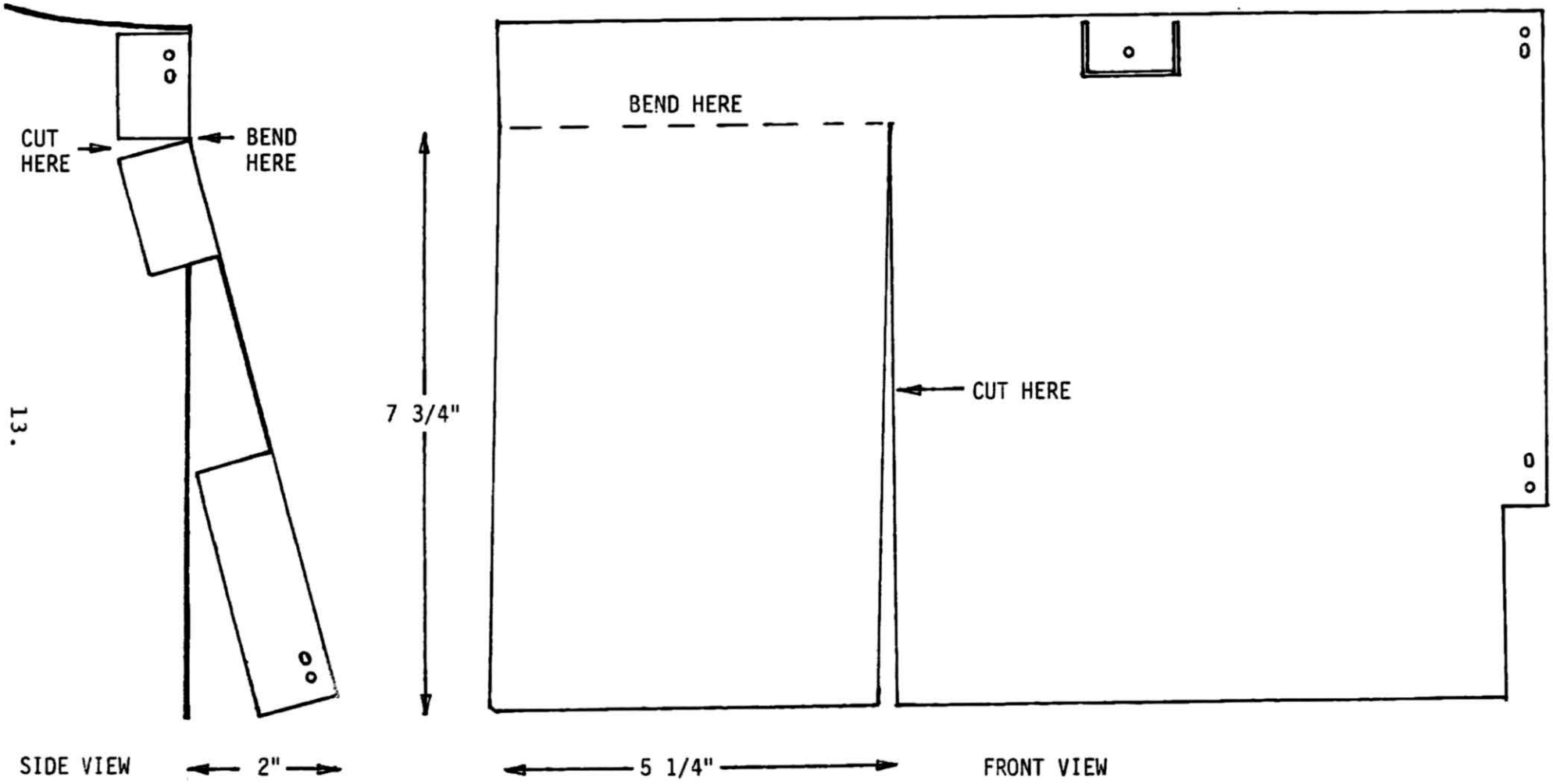


Figure 5. RF SHIELD MODIFICATIONS

TROUBLE SHOOTING YOUR INSTALLATION

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.

In general, you should carefully check every connection to make sure that it is in the right place and that the connectors are oriented properly. Go back and compare what you did to see if it matches the illustrations. Look at the connections of the micro clips with a magnifying glass and make sure that they are firmly attached and do not touch any adjacent pins. Also make sure that the connectors and IC's are well seated and making good contact. Stand back and look at the cables and connectors on the main TRS-80 board to make sure that you did not accidentally bump any of them while installing the board. If you can't find anything wrong, it is a good idea to have someone else go through the steps of installation with you as they might notice something that you missed.

Here are some of the more common problems and their probable causes:

1. If on power-up, you get a totally blank screen, then you should first suspect the connection of the brown micro-clip.

2. If the screen is covered with a fuzzy, grey picture with lines in it, you should check the yellow clip.

3. One possible problem is evidenced by a lot of strange characters being displayed on the screen. However, when something is typed on the keyboard you can see that the display is changed and that characters are being put on the screen properly, even though the wrong characters are being diplayed. This problem is most likely caused by the Grafyx board not being pressed into the sockets all the way and not making good contact. (See step 9 of the installation insructions.)

4. If all you see is a screen full of random or double-width characters, you should check to make sure that the two IC's taken from the TRS-80 were inserted in the Grafyx board in the proper orientation and that no pins were bent underneath the IC's when they were inserted. You should also check to make sure that the proper trace(s) were cut without damaging any other traces in the vicinity. Also make absolutely sure that you have dug into the circuit board far enough to insure that the

trace(s) are completely severed.

5. If the computer or graphics board quite working after the metal shield was put back on, you should remove the shield and attempt to get the computer and board to work without it. Most likely, the Grafyx board was not pressed into the sockets as far as it could have been and/or the shield did not clear the board and placed pressure on one side of the Grafyx board causing it to be tilted and come part way out of the sockets. Also check for any micro-clips which may have been knocked loose.

6. If the disk drives don't work properly after putting the computer back together, take the computer top and shield off. You will then be able to see a flat white connecting cable located at the top of the TRS-80 board above the Grafyx board. 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.

The following problems are ones which are only apparent when the GTEST program is run and the results shown on the screen. The GTEST program should alternate showing patterns in the top and bottom halves of the screen. In the blank half of the screen, the number of read errors is shown.

7. If the screen goes completely white when GTEST is executed, then check to make sure that pins 3, 4, 5, and 6 of the ribbon cable connecter are making contact with the IC leads underneath. You may have to remove the connector and bend the pins inward slightly.

8. If the hi-resolution dots appear all over the screen, and no read errors are shown, then you should suspect the purple micro-clip. However, if there are a large number of read errors, check the orange clip.

9. If no hi-resolution dots appear and a large number of read errors are reported, take a second look at the blue micro-clip.

10. If no hi-resolution dots appear and there are no read errors, check the red and green micro-clips for the proper connections.

11. If the cross-hatch pattern has several horizontal lines duplicated, but doesn't give any read errors, check to make sure that the fourteen pin header is well seated, making good contact, and is on the correct IC (U20).

12. If the hi-resolution display mode is not disabled when you press the orange reset button, then the white clip is not properly in place.

If you are still unable to find and correct the problem, completely remove the board and then try installing it again, being careful to READ EVERY STEP as you to along. While the board is out, check to make sure that you didn't bend any of the pins on the bottom of the Grafyx board when inserting it. If you did, carefully straighten the pin(s) with needle nose pliers and try installing the board a second time.

If you still can't find the problem, write to the factory and give a detailed description of all aspects of the problem. If you do find and correct error(s) in installation, and the Grafyx Solution still fails to perform properly, it is possible, although not likely, that one of your mistake(s) may have damaged the Grafyx board, the TRS-80, or both. If this is the case, be sure to describe everything you did in your letter.

REMOVING THE GRAFYX BOARD

If you decide to remove the Grafyx board now or at a later date, you will need to restore the TRS-80 to its original configuration. This is done by first removing the two socketed IC's from the Grafyx board and putting them in the corresponding sockets on the TRS-80. You may then try to solder the cut trace(s) back but this is difficult because of the skill required. We recommend using a jumper with a micro-clip at each end instead. These can be purchased from Radio Shack. You will need to jumper pin 10 of U19 to pin 10 of U81. A jumper from pin 12 of U17 to pin 8 of U39 is required if the optional trace cut shown in Figure 4 was performed. You will also need to re-insert the wire removed from the connector in the upper-right portion of the TRS-80 board. You can refer to the drawing which was used to show you how to remove it. Your computer should now work exactly as it did before it was modified.

3/ THEORY OF OPERATION

Even if you aren't interested in the electrical aspects of your computer, it couldn't possibly hurt you to read through this brief section. If you're really interested in understanding all of the details, it will help to read the video display section in the Model III Technical Manual (available from Radio Shack) before continuing.

The video display section of your TRS-80 consists of, among other things, a 1024 byte block of video RAM (two 2114's), a Character ROM which generates the alpha-numeric, Japanese, and special characters, and a multiplexer (74LS153) which allows the generation of crude graphics with a resolution of 128 horizontal by 48 vertical. The 1024 locations (bytes) of video memory contain the values of the alpha-numeric or graphic characters which are displayed on the TRS-80 screen as 16 rows of 64 characters or 16 rows of 32 characters. In the 32 character mode, only every other byte of video memory is displayed. The video memory in the TRS-80 occupies memory locations 15360 through 16383 (3C00 to 3FFF in hex), and can be directly examined or modified by the Basic PEEK and POKE statements. See the sections of the TRS-80 Level II manual concerning PRINT@, PEEK, POKE, CHR$, and the Memory Map for more information on screen addressing and updating.

Adding the Grafyx board expands the graphics capabilities of the TRS-80 to give a resolution of 512 horizontal by 192 vertical (256 horizontal by 192 vertical in the 32 character display mode). The hi-resolution display, when enabled, actually overlays the normal screen display. 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 12K bytes of memory made up of six 2K x 8 static memory chips. The memory addresses used to drive the Grafyx memory are actually the same scan and address lines used to drive the normal 1K screen memory. By plugging into socket U67 on the TRS-80, the eight data lines are made available for reading and writing to the hi-resolution memory. However, the Grafyx memory does not use up any memory address space. This is accomplished by bank selecting. When reading or writing from the hi-resolution memory, the appropriate bank is enabled by an OUT command to port 255 which sets the board to the program/read mode and contains the proper bank number. When in the program/read mode one of the twelve 1K byte banks of memory occupies the address range normally devoted to the video screen - 15360 to 16383 decimal, 3C00 to 3FFF hex. Note that when in this

mode, data written to the video screen will not be displayed, but will instead be sent to the Grafyx memory. Data is written a byte at a time so that eight dots on the screen are updated with each memory access. So if you want to set only one bit in the byte, you must first read the byte, set the appropriate bit, and then write it back.

When in the hi-resolution display mode the eight bit output of the hi-resolution memory is sent to an on-board parallel to serial shift register (74LS166) which is synchronized with the TRS-80 character shift register. The Grafyx Solution takes the standard serial output from the TRS-80 and ORs it with its own graphics signal and then feeds the combined signal to the raster scan video generation circuitry in the TRS-80.

The Grafyx Solution also contains a six bit Select Latch. This latch takes advantage of the unused seven most significant bits of the cassette port and can only be written to and not read from. Bit number seven (the MSB) determines whether or not the board is in the program/read mode. Bit number six determines whether or not the hi-resolution display is enabled. Bit number five is not stored, but is used only to latch the other six bits into the Select Latch. Bits number one through four determine which bank of memory is being accessed. There are three basic modes of operation for the Grafyx Solution:

1. Normal Display - Characters and block graphics are displayed as if the Grafyx Solution was not installed.

2. Hi-Resolution - The hi-resolution graphics screen is displayed in addition to the normal alphanumeric, graphics, and special characters.

3. Program/Read Mode - Data can be read from or written to the Grafyx memory. While in this mode, data can not be read from or written to the normal video display screen. There are twenty-four variations of this mode depending on which memory bank is being programmed and whether or not the Hi-Resolution mode is also enabled.

The Grafyx Solution is set to the Normal Display mode whenever the reset key is pressed or the computer is turned on.

4/ USING THE GRAFYX SOLUTION

The Grafyx board, while not difficult to program, can be used on two levels of complexity. By using the supplied GBASIC, COLM80, and SAVLOAD programs, in addition to other available software, the advanced graphics can be used by the majority of users whose major interest is not programming. The enthusiastic programmer, on the other hand, will probably want to learn exactly how the hi-resolution graphics are programmed and displayed in order to make the most of the board's capabilities.

4A/ 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 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 load the GBASIC program as explained in section 4C. You should now be in Basic with an underline character prompt indicating that you are in Extended Graphics Basic. 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 511. Y is the Y-coordinate and can range from 0 to 191. 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 (255,96,1) 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 hit the CLEAR button or 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 (X1,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. Since you can't see what you are typing when most of the screen is lit, carefully type Z=&REV again.

The Grafyx Solution software also gives you four different plot densities so 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 512 x 192, mode 2 gives 256 x 192, mode 3 gives 128 x 192, and mode 4 gives 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, an FC or Illegal Function Call error will result. 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.

Note that if you are in the TRS-80 32 character display mode, the resolution resulting with each mode does not change. The exception is that in mode 1, the resolution will be decreased to 256 x 192. If you are displaying a hi-resolution picture, you should not change from 32 to 64 characters or vice versa since the display will be garbled the same way that the normal display is.

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:

All commands must be preceeded 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, 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 following statements,

X,X1,X2 are X-coordinate values from 0 to 511
Y,Y1,Y2 are Y-coordinate values form 0 to 191
C represents the color of the point(s) plotted by the function; 0 clears the point(s), 1 sets the point(s), and 2 complements the point(s). A color of 3 draws a dotted line when used with the LINE and BOX commands.

&ON - Enable the hi-resolution display. This statement will usually be at the beginning of every program.

&OFF - Turn the hi-resolution display off. When first loaded, GBASIC assumes the status of the display to be off.

&CLS - Clear the hi-resolution screen without affecting the standard text characters.

&MODE (M) - Sets the Grafyx Solution to the desired plotting resolution. M=1 for 512 x 192, 2 for 256 x 192, 3 for 128 x 192, and 4 for 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 Z equal to 1 if the X,Y position is set; 0 if it is clear.

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

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

an inverse display.

&LPRINT (P) - Copy the contents of the hi-resolution and text screen to a printer. P corresponds to the type of printer you have. P=1 for Epson MX70, MX80 with Graphtrax, MX100; 2 for Enhanced MX80 with Graphtrax; 3 for RS LPVII, LPVIII (DMP-100, 200 substituted on request); 4 for IDS 445G, 460G, 560G, Prism; 5 for Seikosha (Axiom) GP-100 (Centronics 739 substituted on request); 6 for Okidata Microline 82A, 83A with Okigraph; 7 for Anadex 9500, 9501; 8 for the C. Itoh 8510A, NEC PC-8023; and 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 a center at X,Y. The radius can be a value from 1 to 511 and corresponds to the number of points in the X direction. Portions of a circle may extend beyond the edge of the screen without any adverse affects.

&FILL (X,Y,S) - Color in the shape surrounding the point X,Y. S=0 for half shading; S=1 for solid shading.

&GET (X1,Y1,X2,Y2,Z%(0)) - Copy the contents of the rectangular section of the screen with corners at the given coordinates into a single dimension integer array variable such as Z%. The array must have a dimension at least equal to INT((ABS(X2-X1)+1) * (ABS(Y2-Y1)+1) / 16) + 3.

&PUT (X,Y,Z%(0),F) - Take the contents of the array created with the &GET statement and display it beginning at the given X,Y position. If a portion of the rectangular area extends past the edge of the screen, it 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; and 3 to OR the contents of the array and screen.

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 apparant 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 400,0, while moving the other endpoint across the bottom of the screen.

```
10 CLS :'Clear normal screen
15 Z=&CLS+&ON+&MODE (1)     :'Get ready to draw lines
20 FOR X=0 TO 511 STEP 5    :'Change one end of line
25 Z=&LINE (400,0,X,191,1) :'Draw line
30 NEXT X :'Keep on till all lines done
```

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+&ON+&MODE(1) :'Set up a clear screen
25 Z=&CIRCLE (255,96,48,1) :'Draw circle
30 Z=&FILL (255,96,0) :'Fill in circle
```

```
35 Z=&GET (255+56,96+28,255-55,96-29,A%(0))
40 Z=&PUT (512*RND(0),192*RND(0),A%(0),
   1+2*INT(RND(0)+.3))
45 GOTO 40 :'Go place another circle on the screen
```

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 512*RND(0) results in a random X-coordinate between 0 and 511 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.

It is possible to draw arcs by first drawing a circle and then using the GET command to get only a portion of it. You can then put the section of a circle on the screen using the PUT command.

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

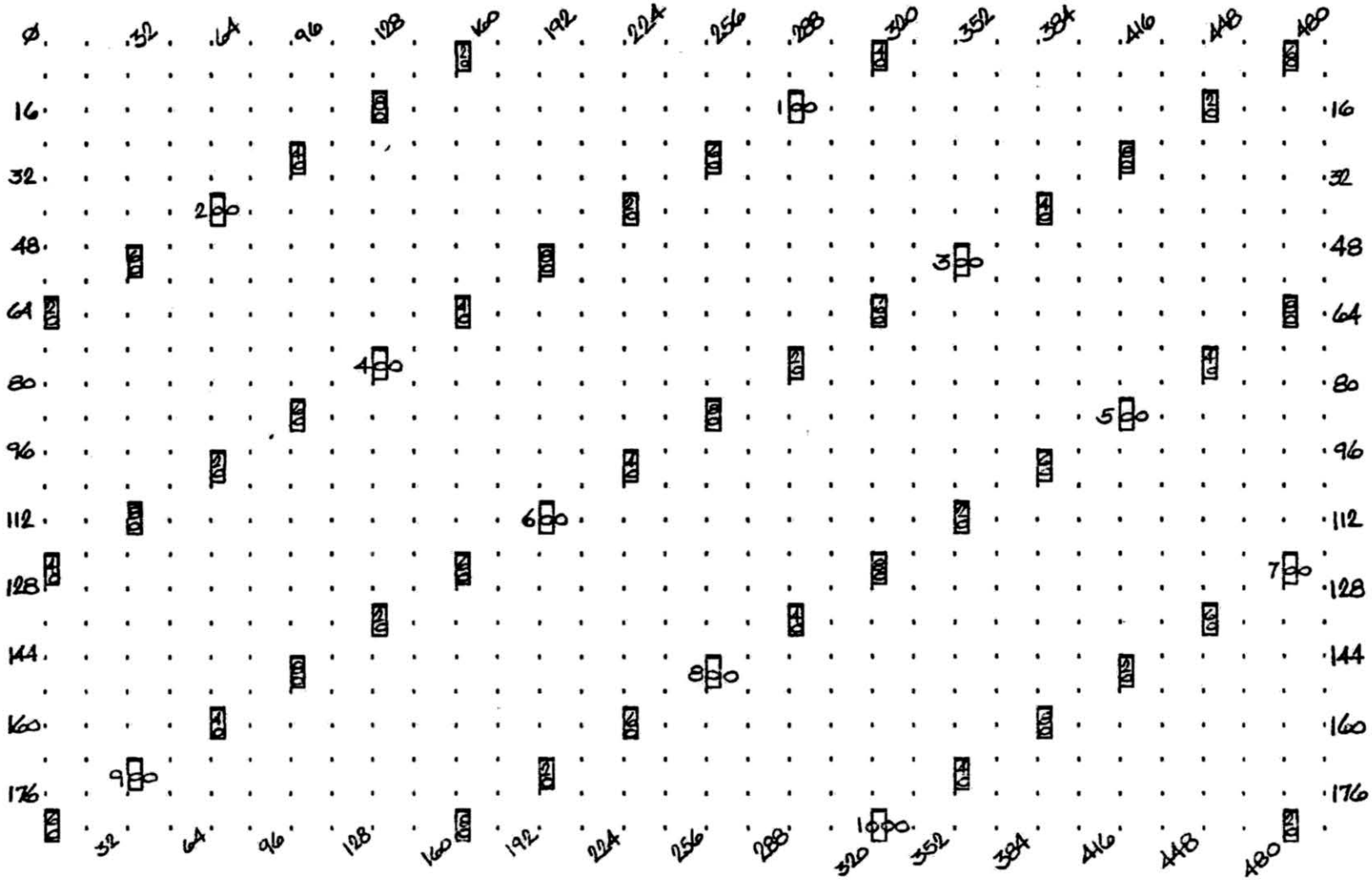
includes one extra row of dots along the bottom. This is because we want to move the shape up the screen one dot at a time and have the old shape erased as we go. Line 45 is a loop which places the missile on the screen in different vertical positions using the PUT function. The value S determines how 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 diagonaly. 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 at the end of this section which shows a matrix of 80 x 24 dots. This is equivalent to the number of dots used in two rows of ten characters since the standard alphanumeric characters are defined on an 8 x 12 matrix. This sheet was carefully constructed so that it has the same proportions as the points will when they are displayed on the screen. You will probably want to make a number of copies of this grid for future use.

On the next page is a grid of dots with the corresponding X, Y coordinates labeled. By referring to this page, you can easily and quickly determine the coordinates which correspond to the position you want on the screen. In addition, the sheet shows the PRINT @ locations so that you can easily position text in relation to points on the hi-resolution screen.

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!



X,Y Coordinates & PRINT @ Locations

DAP
9/6/82

4B/ USING THE 80/85 COLUMN PROGRAMS

One of the neat features of the Grafyx Solution is that it lends itself to use as an 80 or 85 column display board. This allows you to view 1280 (80 x 16) or 1360 (85 x 16) characters on the screen instead of just 1024 (64 x 16). This is important in many applications where you simply need to view more information than the normal display allows. In order to accomplish this, we have included the COLM programs which provides for a software driven 80 or 85 column display using the hi-resolution graphics to create characters. You should now execute one of these programs as explained in section 4C.

The COLM programs are compatible with all versions of DOS and Basic. They also work with any program which uses the internal character display routines. So, for example, most Basic programs, EDTASM, and EDAS all work properly. The only programs which are not compatible are those which place characters directly into video memory such as SCRIPSIT, VISICALC, and SUPERZAP.

Once enabled, all characters sent to the screen by TRSDOS and Basic operations such as LIST and EDIT will be shown in the new format. A neat function that has been added is the ability to have an inverse video display. The display is switched from normal to inverse, or vice versa, each time a CHR$(16) is PRINTed. All of the CHR$ cursor positioning and control codes described on page 228 of the Basic Reference Manual work correctly. The feature which allows you to protect from 0 to 7 lines at the top of the screen is also supported. The TAB function performs correctly as do the default tabs obtained by separating PRINT items with a comma.

However, it is not possible to have a blinking cursor. Another slight difference is that the PRINT @ statement (except for PRINT @ 0,) will not work with the new display format. However, the same effect is possible by directly modifying the cursor position in memory. This is done by placing the desired X position (0-79/84) in memory location 16416 and the Y position (0-15) in location 16417. For example, the following line will move the cursor to the center of the screen and print "Hello".

```
10 POKE 16416,40: POKE 16417,8: PRINT"Hello"
```

A more universal method of accomplishing the same thing which works correctly with both 64 and 80/85 character displays is to include the following line at the beginning of your Disk Basic program:

```
10 DEF FNCURSR$(COL,ROW) = CHR$(28) + STRING$(COL,25)
   + STRING$(ROW,26)
```

You may then position the cursor anywhere in your program using the CURSR$ function. So to put "Hello" in the screen center would require the statement PRINT CURSR$(40,8)"Hello"

Changing the PRINT @ statements is usually all that is necessary in order to insure that a Basic program will work with the COLM programs. The only other difference from normal operation is that the block graphics and special characters numbered 128-255 can not be displayed with the PRINT statement. Character numbers from 128-191 are not printed, and the ones from 192-255 are always interpreted as space compression codes. However, the POKE statement is not affected and places normal size text and block graphics on the screen so it can still be used to display the graphics and special characters.

The clear to the end of the screen - CHR$(31) control code and CLS command are both designed so that they affect both the normal display screen and the 80 or 85 character display. This allows you to place standard graphics or characters on the screen with the POKE statement knowing that they will be erased when the 80 or 85 character screen is cleared. Since the hi-resolution character screen is ORed with the standard character display, it is not possible to mix standard characters when the COLM programs are in the inverse video mode since the normal characters will be "covered up".

In order to return to the 64 character display, simply press the orange reset key or execute the statements POKE 16414,115: POKE 16415,4. If you use the PEEK statement to read the values in these two locations before writing over them, you can put the origional values back to re-enable the COLM program. If you wish to re-enable the program after pressing reset, it is necessary to load and execute the program again as explained in section 4C.

The COLM programs are designed so that they can be run in conjuction with the GBASIC package. This means that you can mix graphic displays with the 80 or 85 character display. In addition, you can get a printed copy of the screen using the GBASIC &LPRINT function. Before using any GBASIC commands you must enter Z=&ON because GBASIC assumes that the hi-resolution screen is turned off when it is first executed. Also, it obviously does not make sense to ever use the Z=&OFF statement since GBASIC would then blank the screen each time points were plotted and COLM would be turning it back on each time a character was displayed. Note that both the CLS and &CLS commands now erase both the hi-resolution graphics and the text characters. Also, if you have a graphics display, you should not print characters at the bottom of the screen and cause it to scroll. If this happens, portions of the graphics will not scroll and the

resulting garbled display must be cleared with either the CLS or &CLS command.

PROGRAMMING NEW 80/85 COLUMN CHARACTERS:

If you have disks, the CHAREDIT program allows you to change the characters displayed by the COLM programs. This allows you to turn your computer into one which displays Russian, German, Spanish, or some other foriegn language. Other applications include programming a set of APL characters or special symbols which are used in your particular field of interest.

In order to edit characters, you must first type LOAD COLM8xxx/CMD in response to the TRSDOS Ready prompt. Note that you should not be in the 80 or 85 column display mode. You should then go to Basic, being sure to set the memory size to 41466 for COLM8x32 or 57850 for COLM8x48. You can then run the CHAREDIT/BAS program.

CHAREDIT begins by asking you for the number of the character you wish to edit. After entering a number from 32 to 127, the old character contents will be displayed and you can then change the character. Although the characters are displayed on a 6 x 12 grid, realize that for the sake of speed, the COLM programs will only use the data contained in the upper, left 5 x 9 grid of points. Once you are in the edit mode, you have the following commands available:

Arrow keys - are used to move the blinking cursor anywhere within the 6 x 12 character grid.

Space bar - complements the point beneath the cursor.

Clear key - clears all of the points within the character being edited.

Enter key - saves the character being edited and allows you to specify another character to update.

X - terminates editing of the current character without saving it and lets you specify another character to edit.

S - saves the new 80 or 85 column program and character set under a file name which you specify and returns to TRSDOS.

When you type S to save your character set, CHAREDIT actually creates a whole new copy of the COLM program which differs only in the character data table. So all of the information such as the memory size values also apply to the program you create. The new program is executed simply by typing its name in response to the TRSDOS Ready prompt.

4C/ 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 III.

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

LOADING INSTRUCTIONS FOR DISK BASED SYSTEMS:

The supplied programs come on a TRSDOS compatible data disk. If you have two disk drives, you may place the Grafyx Solution Software disk in the second drive and access all of the programs from a TRSDOS System disk placed in drive 0. If you have only one drive, the programs must be moved to a TRSDOS System disk. The Grafyx disk comes with a special copy utility which will copy all of the files to a formatted data or TRSDOS System disk even if you have only one disk drive. It is suggested that you make a backup copy by placing the supplied disk in drive 0 and pressing the orange reset key. A heading, followed by the disk name and date, and all files will be displayed, along with the number of grans required to make a copy. At this time you will be prompted for a destination drive. If you specify drive 0, you will be prompted to change disks. After drive selection, the destination directory will be read into memory, and the amount of free space determined. If there is enough room, all of the files will be transferred. If not, you will be prompted for which files to copy. You may abort the copy at any time by pressing the BREAK key. Once all the files have been copied, you will then be able to access all of them and utilize them as explained below.

To run a machine language program stored on disk, simply type its name in response to the TRSDOS Ready prompt. (Note that GBASICxx and COLM8xxx have special procedures which are described later.) To run a Basic program, go to GBASIC and type RUN"Programname/BAS". The assembly language source code files are loaded using and editor/assembler such as EDTASM or EDAS. Screen Files are loaded using either SAVLOAD program.

Skip over the following page which is for tape based systems and go on to the descriptions of the supplied programs.

LOADING INSTRUCTIONS FOR TAPE BASED SYSTEMS:

Assuming that you've recieved a cassette tape from Micro-Labs, all of the programs for a tape based system are recorded one time on one side at the 1500 (High) baud rate. The programs are saved in the following order: GTEST/CMD, GBASIC16/CMD, GBASIC32/CMD, GBASIC48/CMD, SAVLOAD/CMD, COLM8032/CMD, COLM8048/CMD, OVALS/BAS, LGRAPH/BAS, SPIRAL /BAS, ROSE2/BAS, TRIANGLE/BAS, FRAME/BAS, TUNNEL/BAS, CLOVER /BAS, MISSILE/BAS, HAT/BAS, HAT/HR, SUNSET/HR, and INFINITY/HR.

If you have problems loading from tape, try adjusting the volume or changing tape recorders. Also be sure that you are starting the tape recorder just prior to the program you are trying to load by unplugging the ear jack and listening to the tape. The procedure used to load each type of file is now explained:

1. Machine Language Programs: Type 'SYSTEM' and press 'ENTER'. Next type the name of the program you wish to load and hit 'ENTER' (Note: do not include the /CMD extention in the name.) Start the tape recorder and the program will be loaded. When you get the *? prompt again, press '/' followed by the 'ENTER' key to run the program. Note that GBASICxx and COLM80xx require that the proper value be entered for Memory Size as explained later.

2. Basic Programs: Simply use the CLOAD command to load the program from tape and type 'RUN' to execute the program after it has been loaded.

3. Screen Files: SAVLOAD must first be loaded and run. This program then allows you to load the hi-resolution screen picture into the Grafyx Solution memory.

Descriptions of the supplied programs and important information regarding GBASIC and COLM80 follow in the next few pages. Note that a number of programs are omitted from the tape. Those omitted were either considered unimportant or were not applicable to a tape based system.

If you upgrade to disk at a later date, a copy of the software on diskette may be obtained by sending $6.00 to Micro-Labs.

SUPPLIED DEMONSTRATION PROGRAMS

OVALS/BAS

An interesting demonstration which draws a different design every time you press a key. You must be in GBASIC for this program to work. Press the 'BREAK' key to exit this program.

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.

SPIRAL/BAS, ROSE/BAS, ROSE2/BAS, DESIGNS/BAS, TRIANGLE/BAS
FRAME/BAS, TUNNEL/BAS, CLOVER/BAS, CUBIC/BAS, SPHERE/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 any key.

MISSILE/BAS

The techniques necessary to program moving graphics using GBASIC are demonstrated in this program and explained in section 4A. In this example a small arrow is moved vertically across the screen at different speeds.

HAT/BAS

This is the GBASIC program which generates the HAT/HR picture. Because of the many calculations required for each point and the large number of points, this program runs very slow and is included mainly so that you can see how the HAT picture was created.

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 4E.

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 4E.

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 4D.

LINE/BAS

This Basic program demonstrates the algorithm used to draw a line. This program is explained and listed in section 4D.

UTILITY PROGRAMS

GTEST/CMD

This program will check all of the memory in the Grafyx Solution and display a number of different shaded patterns so that you can visually verify that the board is working properly. Note that you should not be in the 80-column display mode when executing this program.

GB32/BLD, GB48/BLD

These command files allow disk users to easily enter GBASIC. If you have 32K of memory, simply type DO GB32 in response to the TRSDOS Ready prompt; if you have 48K, type DO GB48. Note that these files set the Basic Memory Size to 43616 and 60000 respectively.

GBASIC16/CMD, GBASIC32/CMD, GBASIC48/CMD

Once GBASICxx has been executed, a number of graphics commands are available to the user. A detailed explanation of these commands is contained in section 4B. Almost all of the programs you write will take advantage of this program.

There are three versions of this program which are different only in that GBASIC16 is designed to work with 16K of memory installed while GBASIC32 requires 32K, and GBASIC48 requires 48K. GBASIC uses less than 5K of program memory. Before you can use this program with Basic, it is vital that the proper value be entered in response to the "Memory Size?" prompt. This value should be 27232 for GBASIC16, 43616 for GBASIC32, and 60000 for GBASIC48. However, if you plan to use the 80-column program in conjuction with GBASIC, the proper values are 41466 for GBASIC32 and 57850 for GBASIC48. If you are already in Basic, and have not specified the appropriate value, you must hit the orange reset key before proceeding.

The procedure for installing GBASIC differs depending on whether or not you have disks. If you don't have disks, then the first step after entering the proper memory size is to load and execute the appropriate version of GBASIC in the same manner as that previously described for other machine language programs. (And you can ignore the next two paragraphs.)

If you do have disks, then several steps are required in order to install GBASIC. However, these steps have been combined into one command file using the DOS BUILD command. The supplied command files GB32/BLD and GB48/BLD contain the exact sequence of keystrokes that you would need to type if you were to do the process manually:

```
BASIC<enter>
<enter>
mmmmm<enter>
DEFUSR1=&Haaaa:CMD"L","GBASICxx/CMD":?USR1(0)<enter>
```

In the above example, xx is either 32 or 48, mmmmm is the appropriate five digit Memory Size, and aaaa is either B4A0 for 32K or F4A0 for 48K. Using command files, you can go to GBASIC simply by typing DO GB32 or DO GB48 in response to the DOS Ready prompt.

To install GBASIC with LDOS the command sequence is

```
LBASIC (MEM=mmmmm)<enter>
DEFUSR1=&Haaaa:CMD"L","GBASICxx/CMD":?USR1(0)<enter>
```

The series of commands for DOSPLUS is as follows:

```
BASIC -M:mmmmm<enter>
CMD"GBASICxx"
```

SAVLOAD/CMD

When executed, this routine lets you load or save a hi-resolution picture to disk or tape. SAVLOAD first

prompts you for the desired operation and 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 12,288 bytes of Grafyx memory are saved as 12 consecutive pages of 1024 bytes. In a disk file, each of these pages is composed of 4 consecutive records of 256 bytes each for a total of 48 records in the file. The tape file is composed of 12 blocks each containing a header and 1024 bytes of data.

SAVLOAD/BAS

This program is a Basic version of SAVLOAD/CMD but can only save or load screen files to disk. If you are already in Basic and don't want to return to DOS, it may be more convenient to use this version rather than the machine language version.

GB8032/BLD, GB8048/BLD, GB8532/BLD, GB8548/BLD

These command files allow disk users to easily enable the 80 or 85 column display and enter GBASIC at the same time. If you have 32K of memory, simply type DO GB8032 or DO GB8532 in response to the TRSDOS Ready prompt; if you have 48K, type DO GB8048 or DO GB8548. Executing this file also relieves you of having to enter Z=&ON prior to executing any other GBASIC commands.

COLM8032/CMD, COLM8048/CMD, COLM8532/CMD, COLM8548/CMD

These programs turns the screen display format into 80 x 16 or 85 x 16 columns instead of the normal 64 x 16. Much more information can then be displayed at one time. Its use is described in section 4B.

There are four versions of this program. The programs containg an 80 will give you 80 columns per line while the other two give you 85. The ones containing the number 32 are designed to work with 32K of memory installed while those with a 48 should be used if you have 48K. If you have only 16K and want to use this program, we strongly recommend

spending about $20 to add an additional 16K of memory. Before you can use this program with Basic, it is vital that the proper value be entered in response to the "Memory Size?" prompt. This value should be 41466 for COLM8x32 and 57850 for COLM8x48. If you are already in Basic, and have not specified this value, you must hit the orange reset key before proceeding. Other than remembering to enter the proper memory size if you want to use the program with Basic, this program is executed in the same manner as that previously described for machine language programs.

CHAREDIT/BAS

This program allows you to modify or create a new set of characters to be displayed when using the 80-column display. This program's use is described in detail in section 4B.

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=1 to disable the graphics display and V=2 to enable the display. 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 commands.

KEYDRAW/BAS

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

HI-RESOLUTION PICTURES

HAT/HR

A symetrical three dimensional picture of a mathmatical function which looks like a hat. It was created by the HAT/BAS program.

SUNSET/HR

A picture which was drawn using GBASIC commands.

INFINITY/HR

This picture was created by drawing a series of circles with slightly increasing radi, and then reversing the image.

ART1/HR

Almost every GBASIC command was used in order to create this bizarre picture.

BALLS/HR

A Basic program was used to draw a sphere. The sphere was then manipulated with the DRAW package to obtain the final result.

SURFACE/HR

The SURFACE PLOT package was used to create the basic image. The DRAW package was then used to quickly shade in appropriate areas.

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.

4D/ ADVANCED PROGRAMMING & MODE CONTROL

This section should be read by everyone, even though it is not necessary in order to use the Grafyx Solution. As explained in the Theory of Operation, the Grafyx board contains 98,304 bits of memory, each of which corresponds to one dot position on the 512 x 192 hi-resolution screen. In order to easily access this many dots, the hi-resolution memory (HRM) is divided into twelve 'pages', each of which contains 1024 bytes each composed of eight bits. This means that anytime you change or look at the HRM, you will be updating or reading eight consecutive points on the screen. 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.

However, before a particular point or group of eight points can be accessed, the address/location of the proper byte which contains the point(s) must be determined. The first thing that must be determined is which of the twelve pages of memory contains the correct byte. Each standard character is made up of a grid of 8 x 12 dots, each of which can be on or off. Since there are 64 characters on each line, this gives a resolution of 64 x 8 dots = 512 dots in the horizontal direction. The 16 lines of characters shown on the screen result in 16 x 12 dots = 192 dots in the vertical direction. Therefore, each dot shown on the hi-resolution screen corresponds to a possible dot contained in a character since the Grafyx Solution also has a resolution of 512 x 192 dots. The HRM is organized such that each of the 12 horizontal lines in each of the rows of characters is associated with a particular page. For example, any point which when displayed would appear in the top line of any one of the sixteen rows of characters would have its address in page 0. A point located in the second line of any one of the characters would be in page 1, and so on up to page 11. This way, since there are 1024 (16 lines x 64 characters per line) characters displayed on the screen, each HRM page also contains 1024 bytes. Don't worry if this seems confusing, a programming example which clearly demonstrates this principle will be shown as soon as we learn a few more things about addressing.

Now that we know which page the byte is in, we need to figure out which of the 1024 page positions is the correct one. The position of a byte is calculated by first determining an X and Y position where 0<=X<=63 and 0<=Y<=15. The Y value is multiplied by 64 and added to the X value to get the relative memory address very similarly to the method used to determine the value for a PRINT@ statement. The X value begins with 0 and counts to 63 for each consecutive byte up to the last byte in each of the 16 lines. So if the desired byte was in the bottom, left corner of the screen, X would be 0, Y would be 15, and the relative position would

be X+Y*64=960. In order to get the actual memory location which would be used in a Basic POKE or PEEK statement, 15360 must be added to the relative position.

Now that we have determined the address of the byte which contains our point(s), we have to figure out what data to put in that byte 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 using the PEEK statement. 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 POKE statement. A program example which should clarify any ideas which still seem vague is presented later.

Now that the algorithm used to determine the proper page, address and bit to be changed 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 255. There are three basic modes of operation which are enabled by a simple Basic OUT statement to this port: Normal, Hi-Resolution, and Program/Read. 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. It should be noted that when in the Program/Read mode, data normally written to the video screen is sent to the Grafyx board instead. Therefore, if your program leaves the board in this mode, or the program is halted while in

this mode, the display screen will remain unchanged, even though the computer may be sending information to the screen. This means that the computer will still do what you tell it to do, even though what you type does not appear on the screen. So if you are in Basic, you can usually type OUT 255,32 to set the mode back to Normal Display mode. If you are in DOS, and have the MODE program saved on disk as explained in section 4C, 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. One other minor consideration which is normally taken care of by GBASIC involves the automatic display of the real time clock in the right corner of the display. In order to keep the Grafyx memory from being inadvertently changed, you should always have the clock display turned off when the Grafyx board is in the Program/Read mode.

The function and use of all three modes is explained below:

1. Normal Display. When in this mode the computer operates as if the Grafyx Solution was not installed and standard graphics are displayed. This mode may be entered and exited without losing the contents of the hi-resolution memory. The BASIC statement OUT 255,32 sets the board to this mode.

2. Hi-Resolution. The hi-resolution screen is displayed on top of the normal TRS-80 screen. OUT 255,96 enables this mode.

3a. Program/Read with Normal Display. While in this mode, data can be written to or read from the hi-resolution memory. You should not remain in the programming modes any longer than necessary since things like error messages normally sent to the screen will be sent to the hi-resolution memory and can not be seen. The OUT statement which enables this mode is different depending on which page of hi-resolution memory you are referencing. As explained earlier, there are twelve pages numbered 0 to 11. So if variable P contains the page number, the statement OUT 255,160+P*2 will allow you to access the proper page of memory with the hi-resolution display disabled.

3b. Program/Read with Hi-Resolution Display. This mode is identical to Program/Read with Normal Display except that the hi-resolution graphics display is enabled. OUT 255,224+P*2 (where P is the page number) enables this mode.

To demonstrate how you would enable the above modes, enter and run the following program: (Note that in this and the following programs, the comment statements beginning

with ' do not have to be typed in for the program to work.)

```
10 OUT 255,96:' Set to show hi-resolution screen
20 FOR X=1 TO 1000: NEXT:' Delay momentarily
30 OUT 255,32:' 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 255,96, 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. In order to change every single point on the graphics screen you must sequence through all twelve pages of memory while writing data into all 1024 locations of each page. 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.

```
10 FOR P=0 TO 11        :' Sequence through all 12 pages
20    OUT 255,224+P*2 :' Enable program/hi-res mode
30    FOR X=0 TO 1023 :' Change every byte in each page
40       POKE 15360+X,170 :' Put data in Grafyx memory
50    NEXT X  :' Go to next byte
60 NEXT P     :' Go to next page
70 OUT 255,96 :' Set to hi-resolution mode
```

Now change the 170 in line 40 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 (page)	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 32 and a Y coordinate of 8. So the memory address of the proper bytes would be 32+8*64=544 since all twelve lines of the missile fit in one normal character. This means that the memory address can remain fixed and only the page being programmed is changed. On the other hand, if the missile was going to be moved up one dot at a time, as it moved, it would overlap into the character/address position of the line above and the address value would be 64 bytes less. 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=32 :' Set X position to the middle of screen
140 Y=8  :' Set Y position to the middle of screen
150 FOR P=0 TO 11:' Number of lines/pages in character
160 READ D :' Get data value for the line
170 OUT 255,224+P*2 :' Set to program/hi-res mode
180 POKE 15360+X+Y*64,D :' Put data into HRM
190 NEXT P :' Go to the next page
200 OUT 255,96 :' Set to the Hi-Resolution mode
210 ' Data values for the twelve lines in missile:
220 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 512 x 192 matrix is transformed into the proper bit, page, and address. If you recieved 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 1000. In general, what the subroutine does is figure out the X position (0-63) of the byte containing the point and the bit position within the byte. The Y position (0-15) and page number are then calculated. An address is then calculated using the previously determined X and Y positions. The byte is read in from HRM (hi-resolution memory), the proper bit is set, and the updated byte is then written back into the HRM. 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.

```
0 ' PLOT/BAS
1 ' This program demonstrates the point plotting algorithm
2 ' After clearing the screen, the point coordinates X and Y
3 ' are randomly generated and the point plotting subroutine
4 ' is called.  Hitting any key will exit this program.
5 '
10 CLS:' Clear video screen
12 S=15360:' Address of video screen and HRM
15 OUT 255,96:' Enable hi-resolution display
20 ' Clear hi-resolution screen
30 FOR P=0 TO 11: OUT 255,224+P*2
40 FOR X=0 TO 1023: POKE S+X,0: NEXT: NEXT
50 ' Generate random X and Y coordinates
60 X=512*RND(0): Y=192*RND(0)
70 ' Plot point, loop again if no key pressed
80 GOSUB 1000: IF INKEY$="" THEN 60
90 OUT 255,32:' Put in normal display mode
99 END

1000 ' Point plotting routine
1008 S=15360:' Address of video screen
1010 X3 = INT(INT(X)/8):' Get byte # in horizontal direction
1020 X1 = 7-INT(X)+X3*8:' Get bit within byte (= remainder)
1030 Y3=0: Y1=INT(Y)
1040 ' Calculate which of twelve lines/pages point is in
1050 IF (Y1>11) THEN Y1=Y1-12: Y3=Y3+1: GOTO 1050
1060 ' Now put board in hi-res, programming proper page
1070 OUT 255,224+Y1*2
1080 ' Figure address of byte which contains the point
1090 PS = S + X3 + Y3*64
1100 ' Place new point in byte, save byte back
1110 POKE PS,(PEEK(PS) OR 2!X1)
1120 OUT 255,96: RETURN:' Put in hi-res mode, 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+&ON+&MODE (1)
25 CLS: PRINT"Enter line endpoints X1,Y1,X2,Y2";: INPUT
X1,Y1,X2,Y2
30 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.

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.

4E/ 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 if the proper procedure is followed. Before any routines can be used, be sure to load but not necessarily execute GBASICxx/CMD where xx corresponds to the amount of memory you have.

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 8 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    A,96            ;Load Y coordinate directly
      LD    (Y1POS),A       ;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 CALLING program's responsibility to make sure that the values are within bounds. A program segment to verify that HL ranges from 0-511, the A register ranges from 0-191, and B ranges from 0-2 follows:

```
      SRL   H               ;Test for 0-511
      JP    NZ,ERROR        ;If non-zero, is out of bounds
      RL    H               ;Fix HL back
      BIT   7,A             ;Test for 0-191
      JR    Z,OK            ;If zero, then is in range
      BIT   6,A
      JP    NZ,ERROR        ;If two MSB's set, then error
OK    EX    AF,AF'          ;Save A so can test B
      LD    A,B             ;Put B into A
```

```
CP      0
JP      M,ERROR         ;If less than zero, an error
CP      3
JP      P,ERROR         ;If greater than 2, then error
EX      AF,AF'          ;Restore A
```

A list of all of the parameter addresses and labels is presented next. Note that the addresses for both the parameters and subroutines are given in hexadecimal for the 48K version of GBASIC. In order to change them to the appropriate values for 16 and 32K, simply subtract 4000H and 8000H respectivly. This is equivelent to changing the leading 0F in each address to 07 for 16K and 0B for 32K.

PARAMETER	ADDRESS	# BYTES	RANGE
MODE	0FB17	1	1-4
X1POS	0FB14	2	0-511
Y1POS	0FBA2	1	0-191
COLOR	0FBE2	1	0-2
X2POS	0FC1F	2	0-511
Y2POS	0FC37	1	0-191
DOT	0FD81	1	0-1
PRINTR	0FBE2	1	1-8
XPOS	0F634	2	0-511
YPOS	0F636	1	0-191
RADIUS	0F7BB	2	1-511
SHADE	0FC37	1	0-1
GXPOS	0F63F	2	0-511
GYPOS	0F641	1	0-511
BUFFER	0FC1F	2	0-65536
FUNCT	0F63B	1	0-3

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 destory 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 0F53C will enable the hi-resolution screen.

&OFF - Simply make a call to 0F549.

&CLS - Call 0F50E to clear the screen.

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

&PLOT (X1POS,Y1POS,COLOR) - Once the three parameters have been loaded, just make a call to 0FAD4.

&POINT (X1POS,Y1POS) - After making a call to 0F578, the status of the point 0 for clear, 1 for set will be contained in HL.

&LINE (X1POS,Y1POS,X2POS,Y2POS,COLOR) - The address of the line routine is 0F5A2. If (and only if) you want to draw a dotted line, place a 0 in DOT before making the call to 0F5A2 and then be sure and put a 1 in DOT after returning from the LINE routine.

&REV - Calling 0F559 will complement the graphics screen.

&LPRINT (PRINTR) - The address of this routine is 0F5BF.

&BOX (XPOS,YPOS,X2POS,Y2POS,COLOR) - The parameters for this routine have the additional restriction that XPOS<>X2POS and YPOS<>Y2POS. The routine address is 0F5E6.

&CIRCLE (XPOS,YPOS,RADIUS,COLOR) -The address of this routine is 0F679. Note: this routine uses a ROM call to determine the square root of different numbers. Therefore, since Basic initializes some RAM used by this routine, on disk systems, it is necessary for Basic to have been previously executed. However, you may go back to DOS to run your assembly language program.

&FILL (XPOS,YPOS,SHADE) - Subroutine address is 0F7DF.

&GET (XPOS,YPOS,GXPOS,GYPOS,BUFFER) - This routine is located at 0F905. BUFFER should be a 16 bit address 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.)
        DEFW   0000    ;Offset to next variable
        DEFB   1       ;Number of dimensions
        DEFW   1000    ;The array dimension
BUFFER  DEFS   2000    ;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(XPOS-GXPOS)+1)*(ABS(YPOS-GYPOS)+1)/16)+3 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 attempt to jump to a Basic error routine and return to Basic. If your program is being called from Basic there is no problem. However, if Basic has not been initialized then you will probably have 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 (XPOS,YPOS,BUFFER,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 0FA0E. 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 program will attempt to jump to a Basic error routine and return to Basic.

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   03/22/82
;
; DRAWS LINES GIVEN A LIST OF ENDPOINTS
;
; NOTE: GBASIC must have been loaded before executing
;
; The sample data included draws a pyramid
;
        ORG     6000H
;
; Choose the appropriate one of the following 3 lines
; and delete the other two before assembling
OFFSET EQU      8000H           ;Use this line for 16K
OFFSET EQU      4000H           ;Use this line for 32K
OFFSET EQU      0000H           ;Use this line for 48K
;
MODE   EQU      0FB17H+OFFSET   ;Mode value
CLEAR  EQU      0F50EH+OFFSET   ;Clear screen routine
HRON   EQU      0F53CH+OFFSET   ;Enable hi-res routine
X1POS  EQU      0FB14H+OFFSET   ;Begin X coordinate
```

```
Y1POS   EQU     0FBA2H+OFFSET   ;Begin Y coordinate
COLOR   EQU     0FBE2H+OFFSET   ;Line color
X2POS   EQU     0FC1FH+OFFSET   ;End X coordinate
Y2POS   EQU     0FC37H+OFFSET   ;End Y coordinate
LINE    EQU     0F5A2H+OFFSET   ;Line routine address
;
; Beginning of program
BEGIN   LD      A,1
        LD      (MODE),A        ;Set so draw single pnts
        CALL    CLEAR           ;Clear hi-res screen
        CALL    HRON            ;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      (X1POS),DE      ;Save X1 for line routine
        INC     HL              ;Pnt to next value
        LD      A,(HL)          ;Get Y1
        LD      (Y1POS),A       ;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      (Y2POS),A       ;Save Y2 for line routine
        INC     HL              ;Pnt to next value
        LD      A,(HL)          ;Get color of line
        LD      (COLOR),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 endpnts in format X1,Y1,X2,Y2,COLOR
; This list draw a pyramid.
LIST    DEFW    220             ;These five values
        DEFB    50              ;determine one line.
        DEFW    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
;
        DEFW    -1              ;This signals end of list
;
        END     BEGIN           ;Set execute add to 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 is 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 calculating dot positions. This source file is also included with the Grafyx board.

```
;
; POINT/ASM  02/22/82
;
; SUBROUTINES WHICH WILL CLEAR THE GRAFYX SCREEN
; AND CALCULATE POINT ADDRESSES
;
;
; SUBROUTINE WHICH CLEARS THE GRAFYX SCREEN
CLEAR  LD     A,32+128        ;Starting OUT value
LOOP1  OUT    (255),A         ;Enable proper page
       LD     BC,1023         ;Do all 1024 loc. in page
       LD     HL,3C00H        ;Point to memory address
       LD     DE,3C01H        ;Point to memory + 1
       LD     (HL),0          ;We want to zero memory
       NOP                    ;Delay for TRS-80
       LDIR                   ;Put zeros everywhere
       NOP                    ;Delay for TRS-80
       ADD    A,2             ;Go to next page
       CP     24+32+128       ;See if done 12 pages
       JR     NZ,LOOP1        ;Loop if not
       LD     A,32+64         ;Now put board in
       OUT    (255),A         ;hi-res mode
       RET                    ;Return
;
;
;
; SUBROUTINE TO CALCULATE DOT POSITIONS
;
; On entry:  (XPOS)=X position (16 bit word)
;            (YPOS)=Y position (8 bit word)
;
; On return:
;    HL, (MEMORY) = memory location (3C00H-3FFFH)
;    A, (BMASK) = mask for the proper bit in the byte
;    (PAGE) = page number (0-11)
;    the proper page (0-11) is set for programming
```

```
;
DOTCAL PUSH    BC              ;Save registers
       PUSH    DE
       LD      DE,64           ;# bytes per line
       LD      HL,3C00H-64     ;Start at begin of screen
       LD      A,(YPOS)        ;Get Y value
DOTCA1 ADD     HL,DE           ;Point to next line
       SUB     12              ;12 dots per line
       JR      NC,DOTCA1       ;Loop till Y value < 0
       ADD     A,12            ;Make value positive
       LD      (PAGE),A        ;Remainder is page number
       ADD     A,A             ;Multiply by two
       ADD     A,32+64+128     ;Add in proper bits so
       OUT     (255),A         ;that Grafyx can be set
       LD      DE,(XPOS)       ;Get X value
       LD      A,E             ;Move LSByte to A
       AND     7               ;Leave 3 lowest bits
       LD      C,A             ;Save in C
       SRL     D               ;Divide X by 8
       RR      E
       SRL     E
       SRL     E
       LD      D,0             ;Make sure MSByte zero
       ADD     HL,DE           ;Add X offset (0-63)
       LD      (MEMORY),HL     ;resulting in memory add.
       LD      B,0             ;Clear MSByte
       EX      DE,HL           ;Save memory address
       LD      HL,BTABLE       ;Point to bit table
       ADD     HL,BC           ;Add in table offset
       LD      A,(HL)          ;Get bit mask
       LD      (BMASK),A       ;and save it
       EX      DE,HL           ;Get memory address back
       POP     DE              ;Restore registers
       POP     BC
       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    1               ;Storage for Y coordinate
MEMORY DEFS    2               ;Add. of byte to change
BMASK  DEFS    1               ;Mask for proper bit
PAGE   DEFS    1               ;Page number (0-11)
       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     HL,10          ;Directly load X position
LD     (XPOS),HL      ;Save it for DOTCAL rtn
LD     A,20           ;Directly load Y position
LD     (YPOS),A       ;Save it for DOTCAL rtn
CALL   DOTCAL         ;Determine dot addresses
OR     (HL)           ;Set bit in the byte
LD     (HL),A         ;Write new value back
NOP                   ;Delay for the TRS-80
LD     A,32+64        ;Now set to hi-resolution
OUT    (255),A        ;display mode
```

If you were clearing a point, the statement OR (HL) would be replaced with the two statements CPL and then AND (HL). To complement a point, simple use XOR (HL) instead of OR (HL).

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.

5/ OTHER SOFTWARE 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 Grafyx Solution 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 the Grafyx Solution follows. All programs require 32K of memory and disk except for DRAW which is available on tape and BIZGRAPH which requires 48K. Orders should be sent to Micro-Labs, 902 Pinecrest, Richardson, TX 75080. Shipping is pre-paid.

BIZGRAPH by Ted Carter $98.00

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 Line Graph, Bar Chart, Pie Chart, Area Plot, Histogram Plot, and Scatter Plots. The resulting plots may be saved, changed, or printed on a graphics printer. An important feature is the ability not only to 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 is a flexible program providing automatic labeling of X and Y axis points using 85 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 use your creative abilities to easily draw a picture or design a graphics screen. You may also enter text into hi-resolution memory to fine tune a business chart, for example. By moving a cursor around using the arrow keys and a few commands, you can set, clear or complement points, lines, circles, or boxes. The size of the points that you are setting can be changed at any time. You can even complement the screen or shift the entire screen in any direction. Any section of the screen can be saved in a shape table so that it can be moved or copied elsewhere. Sections of the screen may also be filled in with one of two patterns. And of course you can save copies

of the screen or obtain a hardcopy printout.

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.

3DPLOT by Ronald Hoard $39.95

This is a vector oriented package which enables you to enter data in the form of X, Y, Z coordinates in order to create complex three-dimensional figures and shapes. These shapes may then be rotated, reversed, scaled, transformed in any dimension, placed anywhere on the screen, and viewed from any distance. Screens and shape tables can be saved on disk. Your results may also be sent to a printer with graphics capabilities.

SURFACE PLOT by Ronald Hoard $39.95

Surface Plot lets you create amazing three-dimensional views. You enter an equation 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 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. You can also specify the size of the resulting image. The program automatically removes hidden lines. The documentation contains complete instructions and sample equations so that you will have your computer hard at work without delay. The finished plot can be saved on disk or printed.

GBASIC PATCHES by John Ratzlaff $19.95

This set of programs allows you to merge the GBASIC software with the BASIC program on TRSDOS 1.3, NEWDOS80 2.0, DOSPLUS 3.4 and 3.5, and LDOS 5.1.2. Once merged into a single /CMD file, you simply type GBASIC in response to the DOS Ready prompt. You do not have to specify a memory size or worry about not using the top of memory. The same file works with both 32 and 48K of memory. You can re-enter Basic by typing GBASIC *. In addition, you have from 54 to 640 (depending on which DOS) more bytes of user memory than with the old method of installing GBASIC. Versions of the 80 and 85 column drivers are supplied that are compatible

with the new GBASIC/CMD file.

USA by Harley Dyk $24.00

This 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.

JOYSTICK-MOUSE

A/D INTERFACE

JOY-MOUSE is a unique hardware add-on for the TRS-80 Models III, 4 and 4P which provides the following features:

1) Allows up to two of any of the following Radio Shack input devices to be connected: Color Mouse 26-3025, Koala Touch Pad 26-1185, Deluxe Joystick 26-3012, Joysticks 26-3008, and Electronic Book 26-3141.

2) Hardware provides almost instantaneous high resolution X and Y position values ranging from 0 to 255.

3) Provides SOUND and MUSIC through a 2 1/4" speaker. The Cassette cable plugs into the built in audio amplifier which features a proportional volume control with on/off switch.

4) Works with DRAW and a high resolution graphics board for simple, direct, quick cursor positioning and drawing.

5) Can be used with either Basic or Assembly language.

6) Works with most game programs written for joysticks.

7) Joystick or other input devices may also be read in a non-proportional mode. In this simpler mode, one of eight device positions is obtained by the statement INP(0).

8) The fire button may be read at any time without delay.

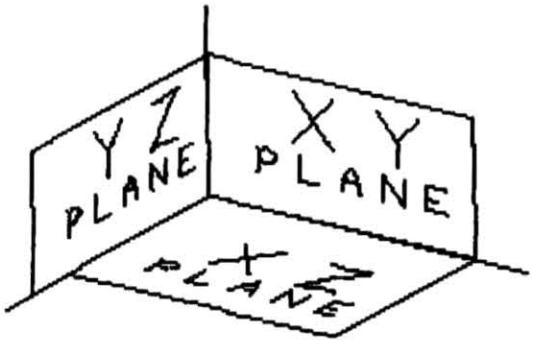
9) Compatible with any device designed to plug into the the TRS-80 Color Computer joystick port.

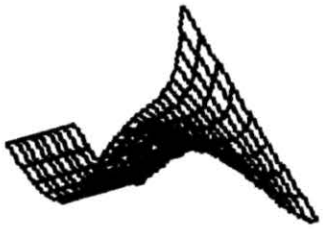
10) Provides FOUR A/D (analog to digital) ports for monitoring of any analog type signals such as temperature, wind speed, light intensity, pressure, voltage, etc.

11) Comes in a 2 x 4 x 6 inch black plastic case and runs off its own power supply.

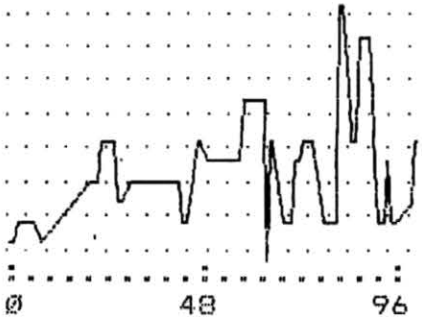
12) Connects to the Model III, 4 or 4P I/O expansion port. The I/O bus is extended so that most peripherals using this port may be connected at the same time.

JOY-MOUSE may be ordered for $99.95. Shipping is free on pre-paid orders. Otherwise, add $4.50 for shipping and handling. Foreign orders add $16.00 for airmail shipping. Micro-Labs, Inc., 902 Pinecrest, Richardson, Texas 75080.





G.I.N.A. GRAPHICS PROCESSOR



GINA is a powerful and flexible graphics design tool for the TRS-80 Models I, III, 4, or 4P with one disk drive. It provides the following features:

1) The TRS-80 standard block graphics screen is actually a scaleable window on a much larger 65536 x 65536 dot tablet. (A hi-res graphics board is not used; the drawing is shown in low-res but comes out in hi-res when sent to the printer.)

2) The arrow keys or joystick are used to easily move a cursor around the screen and set or draw the desired points to create the finished design. Text may also be added in any size in any dimension and at any angle.

3) Allows Scaling, translation, and rotation in 2 and 3 dimensions with simple single key commands.

4) The drawing table consists of up to 16 independent levels which can be scaled and moved together or separately. Each level can also be turned on or off.

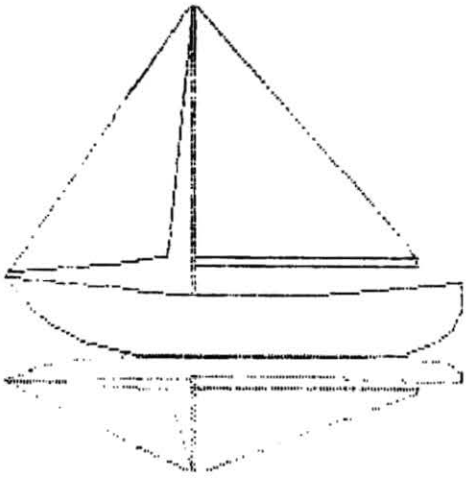
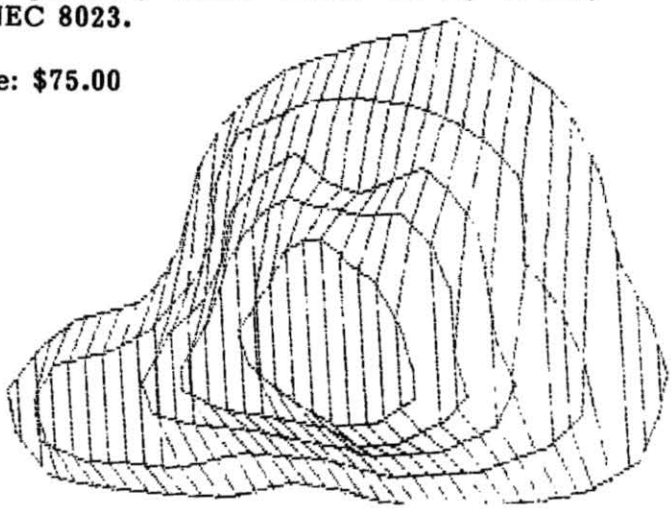
5) Each level can be displayed in three modes: Points only, points joined by lines or pairs of points joined by lines.

6) Points can be specified as occupying a position on the Z axis (perpendicular to the X-Y plane), thus allowing a drawing or data plot to have true depth.

7) The display of three-dimensional data can be in either isometric or perspective projection.

8) The display can be saved for recall in any of four auxiliary display areas. The auxiliary displays can also be superimposed on the current video display.

9) A hardcopy of what is currently in the window can be made. The finished drawing can be printed in hi-resolution on any of the following printers: all Epson, Star Micronics printers, Radio Shack LPVII, LPVIII, DMP-series, C. Itoh Prowriter 8510, NEC 8023.



Price: $75.00