MOTOROLA
Semiconductor Products Inc.

AN-825
Application Note

AN INTERACTIVE GRAPHIC SYSTEM USING THE MC6809

Prepared by
Hunter Scales
Microprocessor Applications Engineer
Austin, Texas

INTRODUCTION

The increased use of computers in all types of data processing has created as well as solved user problems. The decreased cost of owning and operating small computer systems has created a large demand because many small businesses, professionals and others now see the computer as a way of cutting costs or the time involved in solving or handling the problems they have.

One of the problems created by the expanded use of computers is display of data. Many systems have software that generate mountains of output that baffles even the author of the programs. Some systems have expensive options which generate relatively simple bar graphs, but most rely on a printer which is not well suited to displaying information in graphic form. And yet, it is well known that most people can grasp information better, more quickly, and with improved retention if the information is displayed pictorially.

Large computer systems often have impressive graphics capabilities because the cost of the graphics peripherals are a small percentage of the total system cost. If a system, which would provide some of the graphics abilities of the larger systems, could be designed for use on the popular microprocessor-based small business system and still be available for a small portion of the system cost, the acceptability of microprocessor-based systems would increase.

To do this an inexpensive video graphics generator, a low-cost display system, and some sort of intelligent controller are needed. The Motorola MC6847 Video Display Generator (VDG) is a cost-effective means of putting graphic information on a video screen. When used in conjunction with the MC1372 Color TV Video Modulator, the VDG can be used with standard television receivers to display graphics in a variety of modes.

To be effective as a display system, some intelligence is needed to format and maintain the data for the VDG to display. The MC6809 microprocessor is capable of providing the variety of logical bit manipulations and addressing modes which are necessary for high-density graphics display programs. The MC6809 can format and display raw data passed to it by a Basic report generating program fast enough to permit semi-real time animation, if desired.

HARDWARE COMPONENTS

As shown in Figure 1, the hardware components consist of the MC6809 with its stack and user RAM, an MCM2716 EPROM which contains the display routines, 6K of display RAM, the MC6847 video display generator, and the MC1372 color television video modulator.

A complete schematic of the system is given in Figure 2.

This system could be used as a peripheral processor to a microcomputer running the main system software or the MC6809 could be the main processor and use the VDG as its display. In this version, the system runs Basic and the USR function is used to call the video display subroutines.

MC6847 VIDEO DISPLAY GENERATOR (VDG)

The display modes of the VDG are controlled by mode control pins $\overline{A}/G$, $\overline{A}/S$, $\overline{INT}/EXT$, GM0, GM1, GM2, CSS, and INV. The high resolution graphics six mode is used in this system and the VDG is programmed with resistors tied to the proper voltage levels. Table 1 shows the pin configuration.

Table 1. Mode Control Pin Programming for High Resolution Graphics Six Mode

Pin Name	$\overline{A}/G$	$\overline{A}/S$	$\overline{INT}/EXT$	GM2	GM1	GM0	CSS	INV
Logic Level	1	X	X	1	1	1	0	X

X denotes don't care

In the high resolution graphics six mode, the active screen is divided into 6144 bytes: 32 bytes (256 bits) horizontal by 192 bits vertical. Therefore, 12 MCM2114, 1K × 4-bit static RAMs are used as the display memory. The 8T28 and 8T97 three-state buffers are used to isolate the MPU from the VDG during normal VDG operations. The address lines of the VDG sequentially read the display RAM and display the information found there.

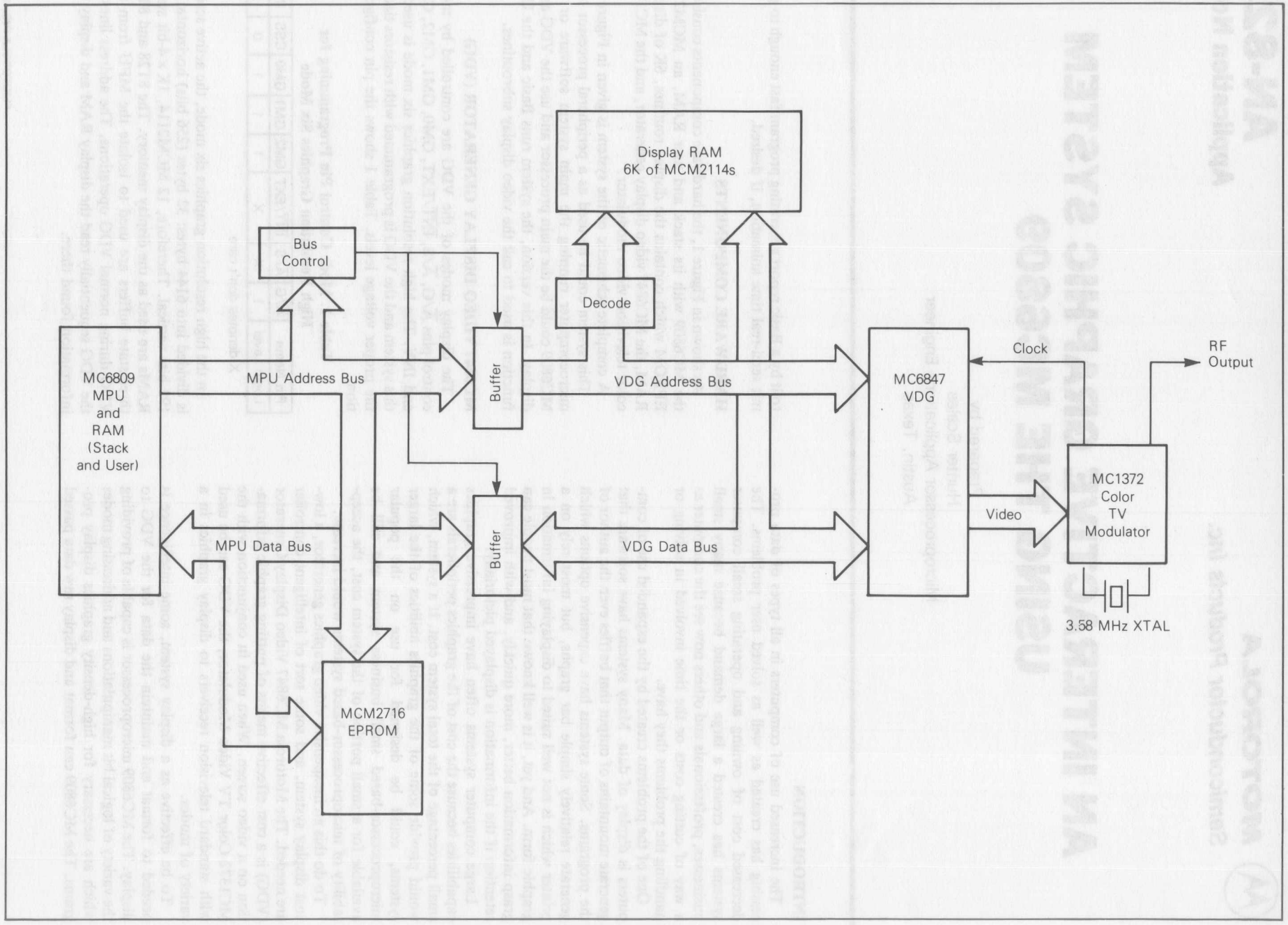


Figure 1. Interactive Graphics System — Block Diagram

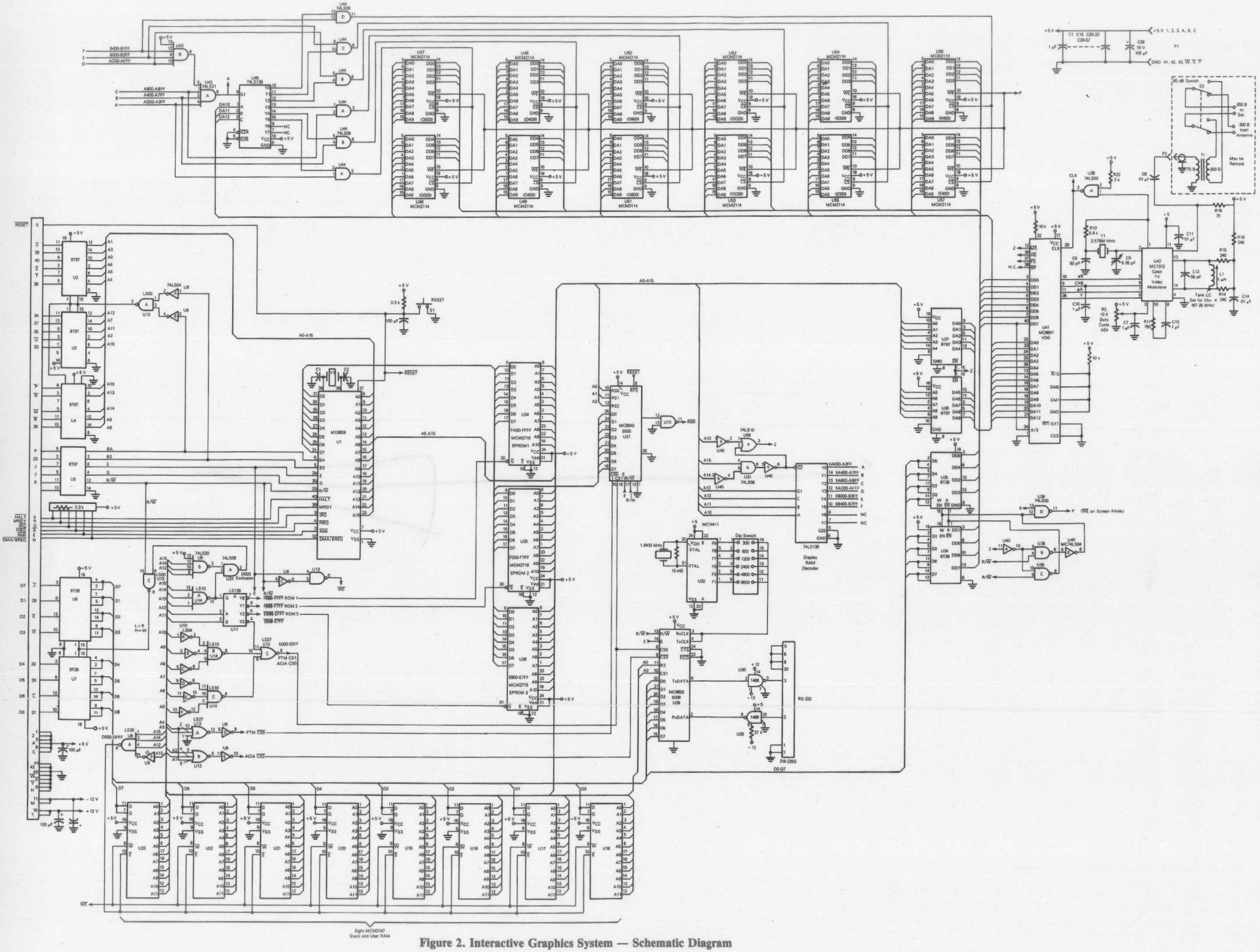
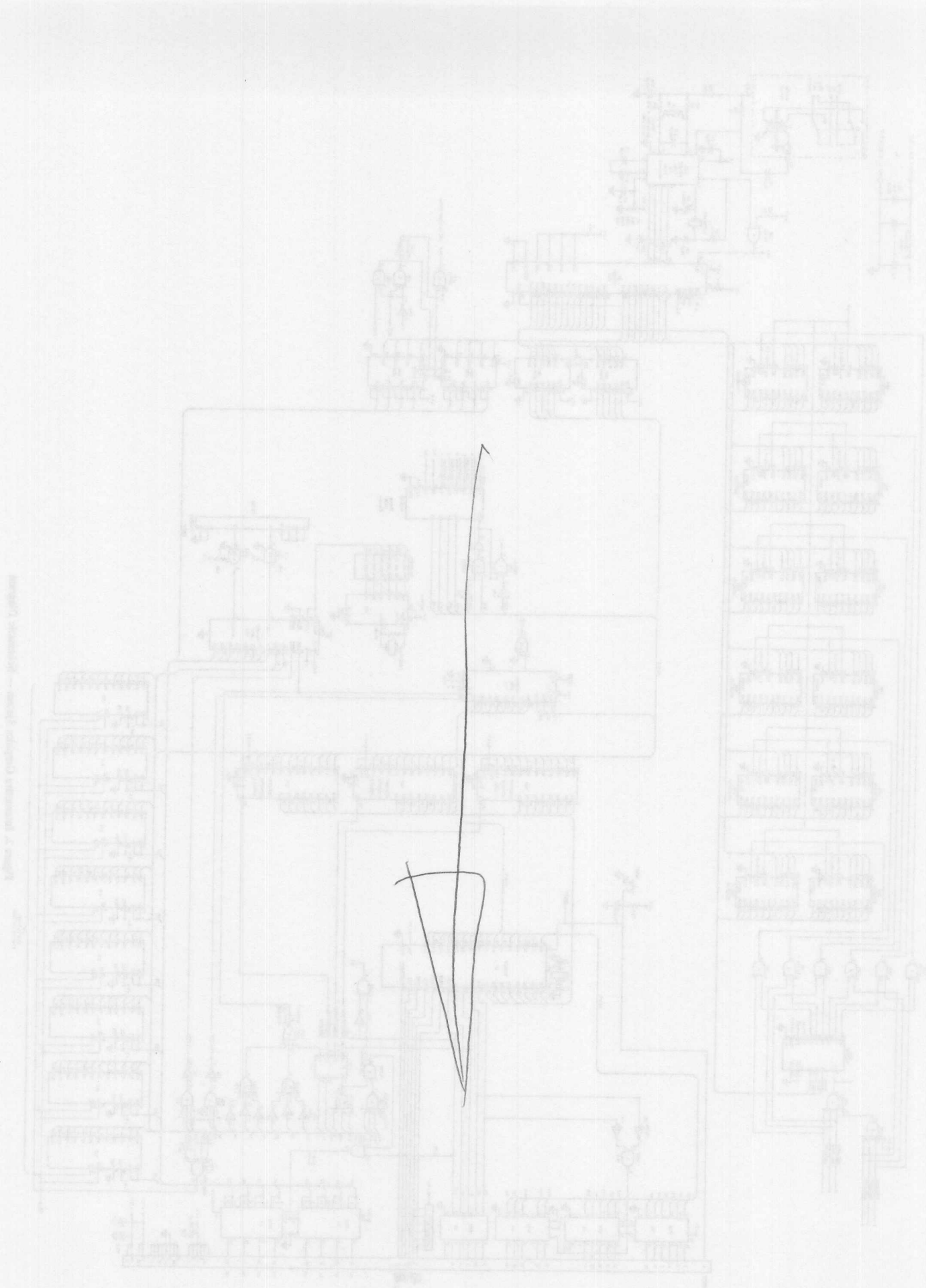


Figure 2. Interactive Graphics System — Schematic Diagram



When the MPU wishes to access the display RAM, the buffers are turned on and the VDG address lines are put in a high-impedance state by the bus control logic bringing the memory select ($\overline{MS}$) input low. This is done to prevent bus contention between the MPU and VDG.

SYSTEM SOFTWARE

To assist in understanding how the software operates on the display, refer to Figure 3 to see how the display area is mapped into the display RAM. This illustration shows how the display screen is interpreted by the VDG.

Each of the 32 bytes on a line has eight bits, with each bit corresponding to a pixel, for a total of 256 separate pixels. Each pixel can be buff or black. If a bit is set, the corresponding pixel is buff and if it is reset the pixel is black. Therefore, when the memory is written all ones ($FF in each byte), the screen is buff. To put a black dot on the screen, the appropriate bit in the appropriate byte must be written with zero. The main task of the software is to determine which byte and which bit(s) in that byte must be set or reset.

To simplify interfacing between the main or driving program (for example, a financial report generator) and the graphics routines, the display screen is considered as an X-Y plane with the origin (X = 0, Y = 0) in the center of the screen as shown in Figure 4. Points can be plotted on the screen by giving the X-Y Cartesian coordinates in twos complement hexadecimal form to the line generation routine (LINGEN). The limits, as shown, are (in decimal) $-128 < X < 127$ and $-96 < Y < 95$.

The main graphics-generating routines are line generation (LINGEN) and point address (PNTADD). Two X-Y points are passed to LINGEN and it calculates which pixels best approximate a staight line drawn between them and sets those bits in the display memory to "draw" the line. LINGEN then calls PNTADD which, given an X-Y coordinate, calculates the address of the byte in which the pixel should be and returns the address and a mask of one byte in which one bit is set. This is the location of the pixel in the addressed byte.

LINE GENERATION ROUTINE (LINGEN) — LINGEN is the main line-drawing routine. It uses a modified Bresenham's algorithm to calculate the pixels which approximate the line. In this method, an error term which shows the deviation of the plotted pixel from the true line is calculated. The X direction is always incremented and, if the error term is positive, the Y direction is also incremented and the point is plotted. The error term is then updated by adding the incremental slope of the line.

For the following discussion, reference the LINGEN flowchart in Figure 5. Upon entry, LINGEN transfers the points to variable locations and then calls PNTADD to get the ending address (ENDADD) and a mask with the final pixel (EMASK). PNTADD is called a second time to get the beginning address in index register X and the first pixel in accumulator A (MASK).

Next, the delta terms representing the change in X (DELTAX) and the change in Y (DELTAY) are calculated. The error term (E) is set to zero and the X increment (XINC) and Y increment (YINC) terms are set to one. Notice that the unity increment for the Y direction is -32 (decimal). This is because the screen format is such that locations addressed 32 apart are directly above or below one another.

Now the delta terms are checked to see if they are negative. If so, they are negated to get the absolute values. The increment terms are also negated. If DELTAX is negative, the error term is set to -1. DELTAX is then subtracted from the error. The first point is "plotted" by inverting MASK and ANDing with the pixels already on screen.

The sign bit of error term E is now checked. If it is negative, the true line will pass above on the next iteration so the Y direction is changed by YINC (which could be negative) and E is updated by adding DELTAX. If it is positive, the line will pass to the right so the X direction is incremented or decremented and E is updated by subtracting DELTAY from E. The address of the current pixel is compared to ENDADD and if they are equal then the current mask in the A register is compared to the EMASK. If they are equal, the line is finished, otherwise a branch is made to plot more points.

Once the end and beginning addresses are found PNTADD is not called again. This is done to save the time it would take to calculate the address of each pixel. Instead, the address is kept in the X register and continually updated. When the X direction is to be incremented or decremented, a rotate of the MASK is done and the shifting of the pixel into the carry bit indicates the crossing of a byte boundary. The address is then incremented and decremented accordingly.

POINT ADDRESS ROUTINE (PNTADD) — PNTADD finds the address of a specified X-Y coordinate by the formula:

$$\frac{(X + XOFF)}{8} - (Y + YOFF) \times 32 + SCRNOFF + SCRNSTART$$

where:

XOFF and YOFF = the offsets necessary to place the origin at the center of the display screen.

SCRNOFF = the length of the screen minus 32. This is the address of the first byte in the last line.

The original X coordinate is then used to find the pixel location within the byte. The last three bits of the coordinate are used as an index into a lookup table (BITAB) which consists of 8 bytes in which there is one set bit in descending order. This is returned in the A register as the mask.

LINE LISTS

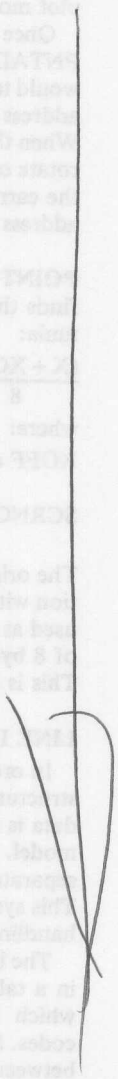
In order to further ease user interface to the system, a data structure which allows the simplest manipulation of the raw data is needed. For this system, the X-Y plotter is used as a model. In machines of this type, the pen is controlled by separate X-Y axis arms and is raised or lowered to draw lines. This system emulates this behavior by providing routines for handling line lists.

The line list consists of a head (the pointer to which is kept in a table) and elements. An element consists of two bytes which are either Cartesian coordinates (X,Y) or special codes. Since the Y coordinate can legally take on values only between -96 and $+95$ ($A0 and $5F) the remaining values can serve as function codes.

The list is drawn by the subroutine DRWLIS in the following manner. A point is checked for a legal Y value and, if legal, a line is drawn from the previous point in the list to the current one. If the Y value is $6F, the imaginary pen is moved to the next point and processing continues. In this way non-continuous objects can be drawn. If the Y value is $70, the next two bytes in the list are taken as the address of the continuation ofthe list (indirection). Finally, if the Y value is $71, the list is considered ended and control is returned to the caller. A flowchart of DRWLIS is given in Figure 6.

CURSOR CONTROL MODE

In order to allow the user to easily generate the desired format for his particular system, a cursor-oriented input system is provided. This set of programs takes care of the "bookkeeping" of the line lists and pointer table. A cursor is pro-



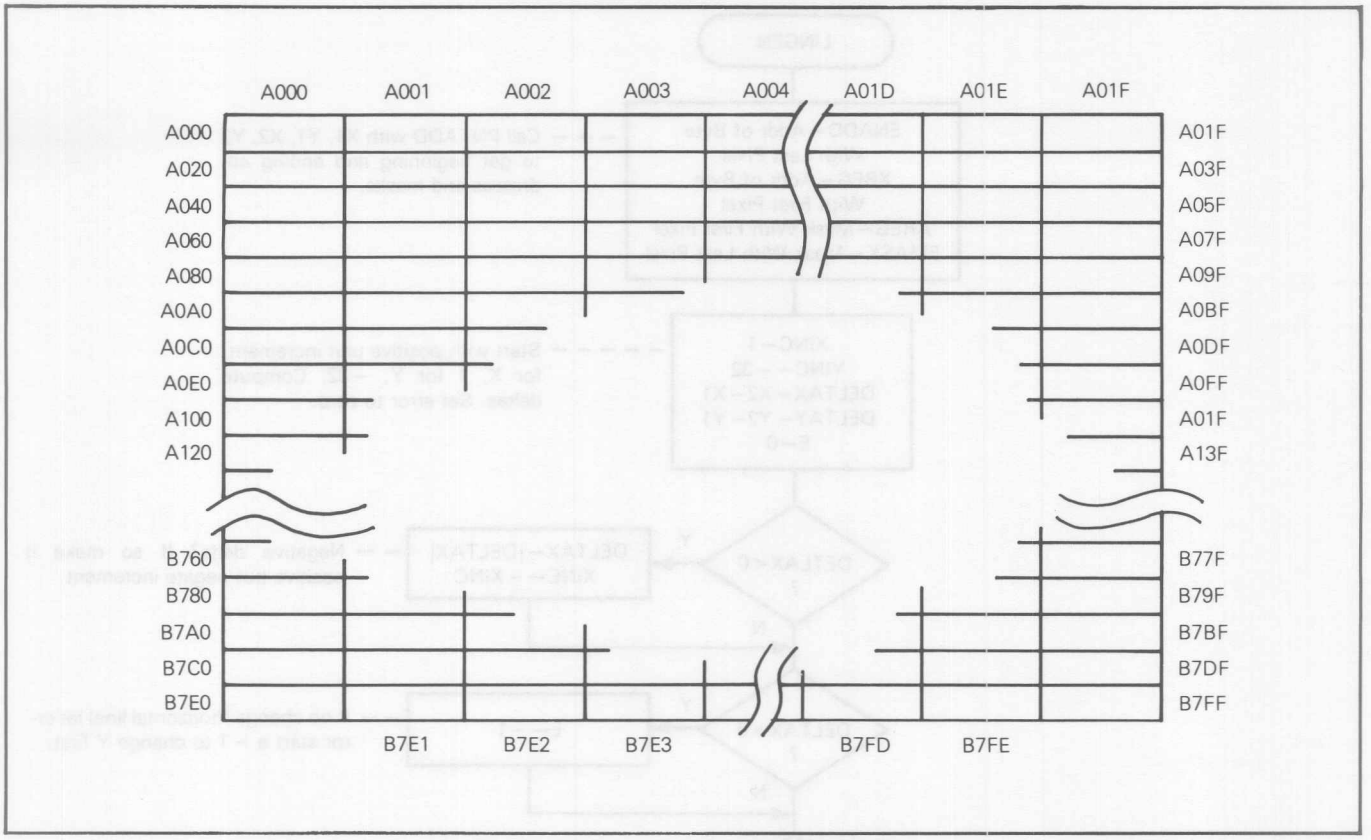


Figure 3. Memory Map of Display RAM

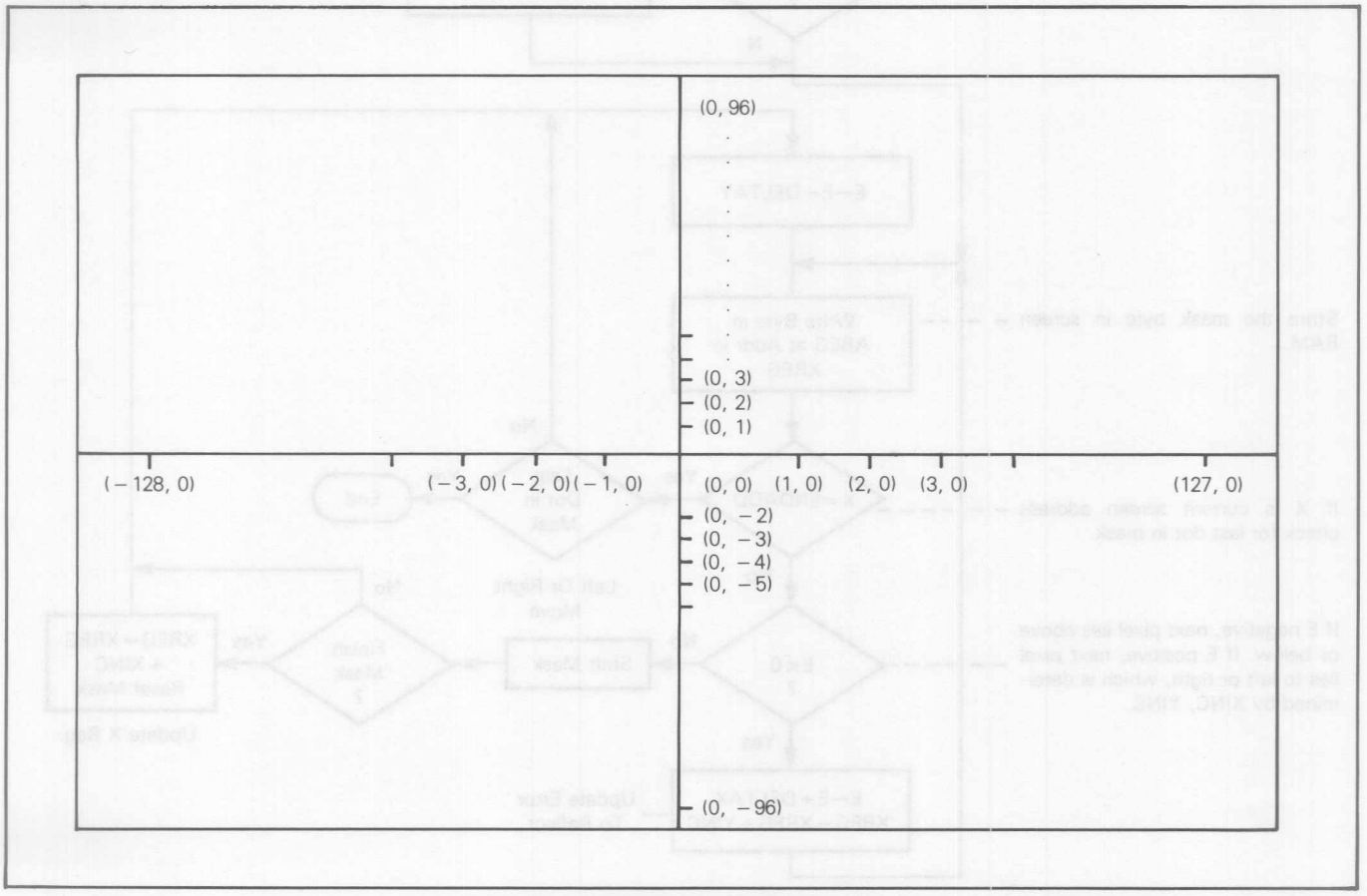


Figure 4. Software Map of Display Screen

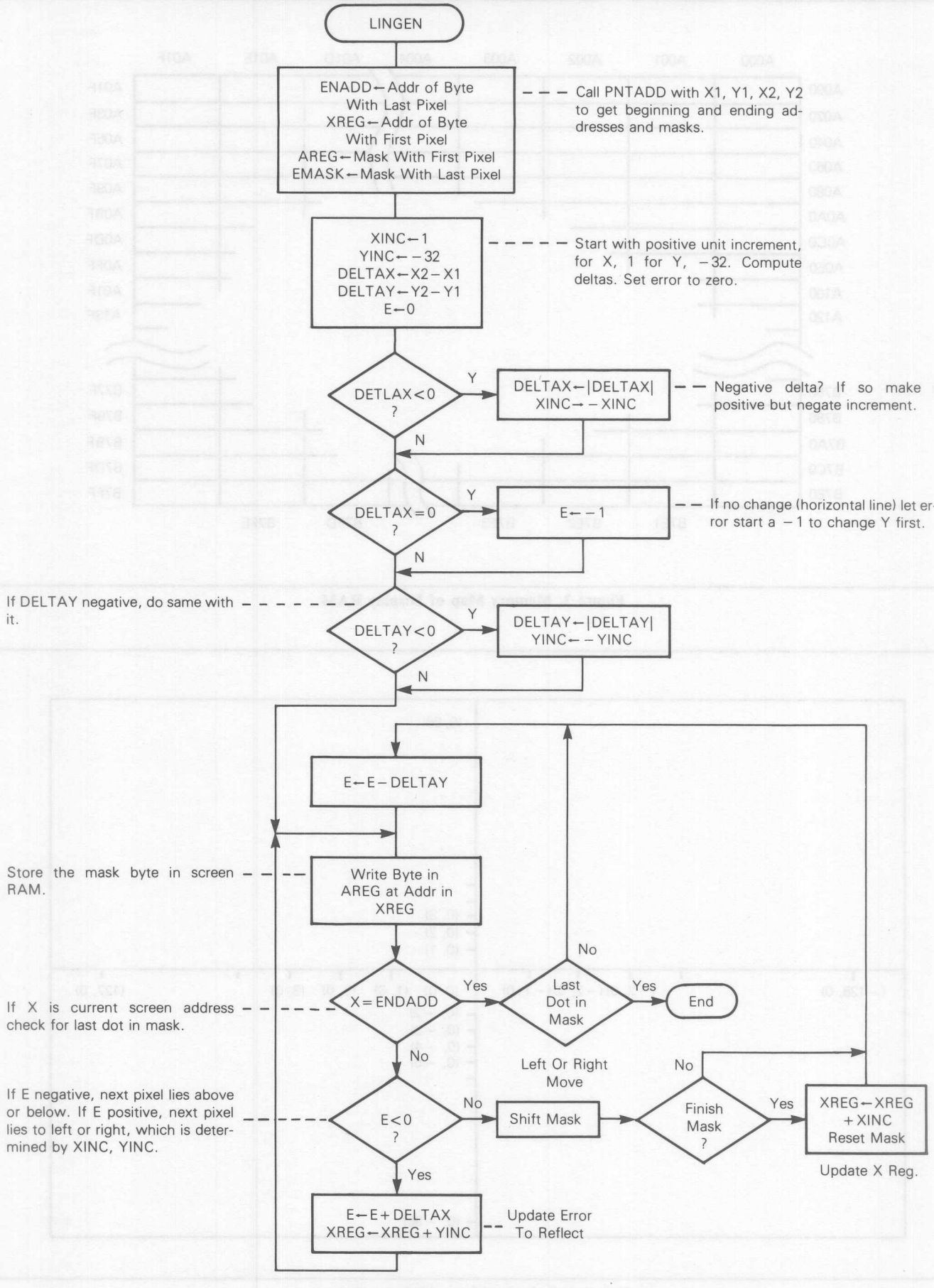


Figure 5. Flowchart of Subroutine LINGEN

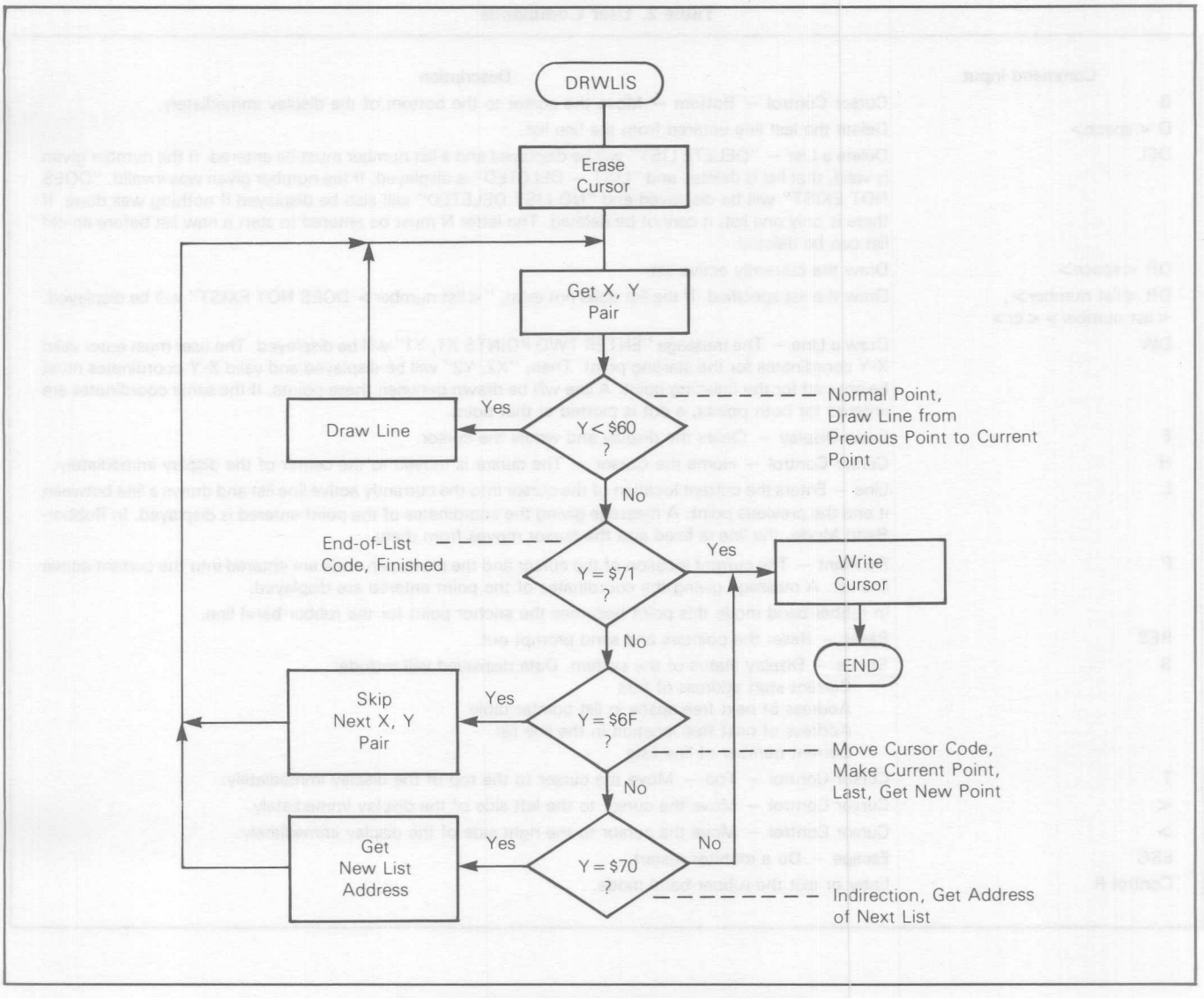


Figure 6. Flowchart of Subroutine DRWLIS

vided to give visual feedback to the user and several commands allow the entry, modification, and manipulation of the line lists.

Entry into the main program at GRAPH causes the initialization of the necessary pointers and flags and prints a prompt asking for an address in memory in which the lists can be stored. The command handler is then entered, the screen is cleared, and the cursor is written to the screen. The system is then ready to execute commands.

The cursor consists of a line width that is one bit wide horizontally and four vertical lines high. The cursor position is defined as the point at which the top of this character is located.

It can be manipulated by using standard ASCII control characters. Back space ($08) moves the cursor to the left one position (one bit width), Control-L ($0C) moves the cursor right one position, Control-K (Vertical Tab $0B) moves the cursor up one position (one line width), and Control-J (Line Feed $0A) moves the cursor down one line.

Line lists are compiled by the use of the Point and Line functions. Entering a P on the console terminal causes the current location of the cursor, along with the move pen control code, to be entered into the currently active line list. Thus, when the list is drawn the "pen" is moved to this location before proceding with the next point. The first point in a list should always be this code followed by the starting coordinate of the first line.

Typing an L on the console terminal causes a line to be drawn on the screen from the point entered to the current cursor location and the coordinates of the cursor to be entered as a point in the line list. Appropriate messages are printed on the console confirming these actions.

Typing an N on the console terminal causes an end code to be entered in the current list and a new list to be started.The screen can be erased by entering an E command and lists can be drawn using the DR command. A complete list of the user commands and their syntax is given in Table 2.

RUBBER-BAND MODE

The cursor control mode is not the only mode of operation of the system. Sometimes it can be helpful to have a continuous line drawn to the cursor at every instant. This is known as a rubber-band line and is a very common method of interactive graphics input. This mode is accomplished by drawing and erasing a line from the last point entered to the current cursor location on a move-by-move basis. The P (fix

Table 2. User Commands

Command Input	Description
B	Cursor Control — Bottom — Move the cursor to the bottom of the display immediately.
D <space>	Delete the last line entered from the line list.
DEL	Delete a List — "DELETE LIST" will be displayed and a list number must be entered. If the number given is valid, that list is deleted and "LIST — DELETED" is displayed. If the number given was invalid, "DOES NOT EXIST" will be displayed and "NO LIST DELETED" will also be displayed if nothing was done. If there is only one list, it cannot be deleted. The letter N must be entered to start a new list before an old list can be deleted.
DR <space>	Draw the currently active list.
DR <list number>, <list number><cr>	Draw the list specified. If the list does not exist, "<list number> DOES NOT EXIST" will be displayed.
DW	Draw a Line — The message "ENTER TWO POINTS X1, Y1" will be displayed. The user must enter valid X-Y coordinates for the starting point. Then, "X2, Y2" will be displayed and valid X-Y coordinates must be entered for the finishing point. A line will be drawn between these points. If the same coordinates are entered for both points, a dot is plotted at that point.
E	Erase Display — Clears the display and writes the cursor.
H	Cursor Control — Home the Cursor — The cursor is moved to the center of the display immediately.
L	Line — Enters the current location of the cursor into the currently active line list and draws a line between it and the previous point. A message giving the coordinates of the point entered is displayed. In Rubber-Band Mode, the line is fixed and the cursor moves from there.
P	Fix Point — The current location of the cursor and the move pen code are entered into the current active line list. A message giving the coordinates of the point entered are displayed. In rubber-band mode this point becomes the anchor point for the rubber-band line.
RES	Reset — Reset the pointers and send prompt out.
S	Status — Display status of the system. Data displayed will include: Current start address of lists Address of next free space in list pointer table Address of next free location in the line list Current number of line lists
T	Cursor Control — Top — Move the cursor to the top of the display immediately.
<	Cursor Control — Move the cursor to the left side of the display immediately.
>	Cursor Control — Move the cursor to the right side of the display immediately.
ESC	Escape — Do a monitor restart.
Control R	Enter or exit the rubber-band mode.

point) command causes the line to be entered and the end of the line to become the new "anchor" of the rubber-band line. Of course, cursor movement is slower due to the extra processing of the line generation routine, but this is made up by the utility of the mode. To enter the rubber-band mode, type Control-R on the console terminal. To exit the rubber-band mode and return to the cursor control mode, simply enter Control-R again.

SYSTEM USE

The routines have been programmed into an MCM2716 EPROM and are used with the ASSIST09 monitor (see MC6809-MC6809E Microprocessor Programming Manual for a listing of ASSIST09). The only system dependent routines are the I/O routines and an expression handler. These are accessed using a software interrupt (SWI) instruction followed by a function code, for example: 02 is the code for an output character routine. These routines are simple to write but are hardware dependent and must be provided by the user for his system configuration.

In general, these routines are called as subroutines by a larger program which usually does the data processing and then passes the data to a formatter. This would be a routine to scale the data and put it in the line list format to be displayed by DRWLIS. In this way the routines are system-independent and form a powerful way to display data that formerly was difficult to do on low-end microcomputer-based systems.

PROGRAM LISTING

Figure 7 is a complete listing of the program contained in the MCM2716 EPROM.

```
PAGE  001  GRAFPAKA.SA:1  GRAFPA

00001                          NAM    GRAFPAK
00002                          OPT    ABS,LLE=81
00003                        *  GRAFPAK TO BE LOCATED AT ADDRESS "ROM2"
00004                        *  GLOBALS LOCATED AT ROM2-RAMOFS
00005                        *  TO ACCOMODATE THE HARDWARE CONFIGURATION
00006                        *  OF THE 6809 EVALUATION BOARD.
00007                        *
00008            F800    A ROM1   EQU    $F800     ADDRESS OF MONITOR
00009            F000    A ROM2   EQU    ROM1-$800 ADDRESS OF SECOND ROM
00010            1300    A RAMOFS EQU    $1300     ON BOARD RAM IS HERE
00011            9000    A SCRN   EQU    $9000     VDG SCREEN RAM (6K)
00012            B441    A MODCON EQU    $B441     ADDRESS OF VDG CONTROL PORT
00013            1800    A RAMLEN EQU    $1800     LENGTH OF VDG RAM
00014            0080    A XOFF   EQU    128       X OFFSET
00015            0060    A YOFF   EQU    96        Y OFFSET
00016            FCDF    A GET    EQU    ROM1+$4DF ADDR OF EXPRES HANDLER
00017            DFF4    A ECHO   EQU    ROM1-$180C ECHO FLAG FOR ASSIS09
00018            00FF    A CLERCH EQU    $FF       TO FILL SCREEN
00019                        *
00020A DD00                    ORG    ROM2-RAMOFS
00021                        *
00022                        *  GLOBAL VARIABLES FOLLOW
00023                        *
00024A DD00      0001    A FLAG   RMB    1         0=WRITE  $FF=UNWRITE
00025A DD01      0002    A TEMP   RMB    2         FOR ARITH RESULTS
00026A DD03      0001    A X1     RMB    1         X LOC OF 1ST POINT
00027A DD04      0001    A Y1     RMB    1         Y LOC OF 1ST POINT
00028A DD05      0001    A X2     RMB    1         X LOC OF 2ND POINT
00029A DD06      0001    A Y2     RMB    1         Y LOC OF 2ND POINT
00030A DD07      0001    A XINC   RMB    1         FOR LINGEN
00031A DD08      0001    A YINC   RMB    1
00032A DD09      0001    A MASK   RMB    1         DOT MASK OF LINE
00033A DD0A      0001    A EMASK  RMB    1         MASK AT END LINE
00034A DD0B      0002    A DELTAX RMB    2         CHANGE IN X OF LINE
00035A DD0D      0002    A DELTAY RMB    2         CHANGE IN Y OF LINE
00036A DD0F      0002    A E      RMB    2         ERROR IN LINE
00037A DD11      0002    A ENDADD RMB    2         LAST ADDRESS IN LINE
00038A DD13      0002    A STACK  RMB    2         SAVE USER STACK HERE
00039A DD15      0001    A MODE   RMB    1         CURSOR OR RUBBER-BAND MODE
00040A DD16      0002    A NEXTAD RMB    2         NEXT FREE SPACE IN LIST TABLE
00041A DD18      0002    A LISTP  RMB    2         START OF LIST TABLE
00042A DD1A      0001    A LISCNT RMB    1         NUMBER OF LISTS
00043A DD1B      0002    A CURSAV RMB    2         SAVE UNDER CURSOR
00044A DD1D      0002    A CURPOS RMB    2         X-Y POS OF CUSRSOR
00045A DD1F      0002    A NEXTPT RMB    2         NEXT FREE ADDR IN LIST
00046A DD21      0002    A CURAD  RMB    2         ADDRESS OF CURSOR
00047A DD23      0001    A CURCH  RMB    1         CURSOR MASK
00048A DD24      0001    A DELIM  RMB    1         DELIMITER FROM GETEXP
00049A DD25      0010    A BUFFER RMB    16        GETLIS SAVES LIST NUM
00050A DD35      0005    A SAVC   RMB    5         SAVE CURSOR HERE
00051                        *
00052A F000                    ORG    ROM2
00053                        *
00054                        *  THE FOLLOWING ARE FUNCTION CODES FOR USE WITH
00055                        *  THE ASSIST09 MONITOR SWI FUNCTIONS.
00056                        *
00057            0000    A INCHC  EQU    0         INPUT CHAR NO PARITY
00058            0001    A OUTCHC EQU    1         OUTPUT CHAR TO CONSOLE
```

Figure 7. Interactive Graphics System Program Listing (Sheet 1 of 20)

```
PAGE  002  GRAFPAKA.SA:1  GRAFPA

00059              0002    A PDATAC EQU   2        PRINT STRING EOT AS TERMINATOR
00060              0003    A PDAT1C EQU   3        AS ABOVE WITH CR LF
00061              0006    A PCRLFC EQU   6        PRINT CR LF STRING
00062              0007    A SPACEC EQU   7        PRINT SPACE TO CONSOLE
00063              0008    A MONITC EQU   8        RESTART THE MONITOR
00064                      *
00065                      *  FOLOWING ARE THE MONITOR DEPENDENT ROUTINES
00066                      *
00067                      *  OUTS-PRINT ONE SPACE TO CONSOLE AND RETURN
00068                      *  VOLATILE: A
00069                      *
00070A F000 3F             OUTS   SWI             SWI FUNCTION
00071A F001      07      A        FCB    SPACEC   FUNCTION CODE
00072A F002 39                    RTS
00073                      *
00074                      *  CRLF-SET A CARRIAGE RETURN-LINE FEED STRING TO
00075                      *  CONSOLE
00076                      *  VOLATILE: X,A
00077                      *
00078A F003 3F             CRLF   SWI             MONITOR FUNCTION
00079A F004      06      A        FCB    PCRLFC   CODE
00080A F005 39                    RTS
00081                      *
00082                      *  OUTHL-PRINT THE LEFT NIBBLE IN A REGISTER TO
00083                      *  CONSOLE
00084                      *  VOLATILE: A
00085                      *
00086A F006 44             OUTHL  LSRA            SHIFT LEFT NIBBLE
00087A F007 44                    LSRA            INTO RIHT NIBBLE AND
00088A F008 44                    LSRA            THEN FALL INTO OUTHR
00089A F009 44                    LSRA
00090                      *
00091                      *  OUTHR-PRINT THE RIGHT NIBBLE OF A REGISTER TO
00092                      *  CONSOLE.
00093                      *  VOLATILE: A
00094                      *
00095A F00A 84   0F      A OUTHR  ANDA   #$F      MASK TOP NIBBLE
00096A F00C 8B   30      A        ADDA   #$30     CONVERT TO ASCII
00097A F00E 81   39      A        CMPA   #$39
00098A F010 23   02   F014        BLS    OUTC
00099A F012 8B   07      A        ADDA   #7
00100A F014 BD   F048    A OUTC   JSR    OUTCH    PRINT HEX DIGIT
00101A F017 39                    RTS
00102                      *
00103                      *  OUTHEX-PRINT A REGISTER AS TWO HEX DIGITS
00104                      *
00105A F018 34   02      A OUTHEX PSHS   A        SAVE REGISTER
00106A F01A BD   F006    A        JSR    OUTHL    PRINT LEFT NIBBLE
00107A F01D 35   02      A        PULS   A        GET CHAR BACK
00108A F01F BD   F00A    A        JSR    OUTHR    PRINT RIGHT NIBBLE
00109A F022 39                    RTS
00110                      *
00111                      *  OUT4HS-OUTPUT X REGISTER AS FOUR HEX DIGITS
00112                      *  AND A SPACE.
00113                      *
00114A F023 34   10      A OUT4HS PSHS   X        SAVE CHAR
00115A F025 35   02      A        PULS   A        GET LSBYTE
00116A F027 BD   F018    A        JSR    OUTHEX   PRINT IT
```

Figure 7. Interactive Graphics System Program Listing (Sheet 2 of 20)

```
PAGE  ØØ3  GRAFPAKA.SA:1  GRAFPA

ØØ117A F02A 35   Ø2      A        PULS   A        GET MSBYTE
ØØ118A F02C BD   F018    A        JSR    OUTHEX
ØØ119A F02F BD   F000    A        JSR    OUTS
ØØ120A F032 39                     RTS
ØØ121                         *
ØØ122                         *  GETEXP-WILL RETURN A HEX NUMBER FROM THE
ØØ123                         *  CONSOLE IN THE D REGISTER
ØØ124                         *  ON RETURN Z=1 IF VALID HEX NUMBER, Z=Ø IF
ØØ125                         *  NOT A VALID HEX NUMBER
ØØ126                         *  THE TERMINTING DELIMITER IS SAVED IN DELIM
ØØ127                         *  VOLATILE:  D
ØØ128                         *
ØØ129A F033 34   Ø8      A GETEXP PSHS   DP       SAVE DIRECT PAGE
ØØ130A F035 86   5F      A        LDA    #$5F
ØØ131A F037 1F   8B      A        TFR    A,DP     SET DP FOR MONITOR
ØØ132A F039 BD   FCDF    A        JSR    GET      EXPRESSION HANDLER IN ASSISTØ9
ØØ133A F03C 34   Ø1      A        PSHS   CC       SAVE COND CODE
ØØ134A F03E B7   DD24    A        STA    DELIM
ØØ135A F041 DC   9B      A        LDD    $9B      GET NUMBER
ØØ136A F043 35   89      A        PULS   PC,DP,CC
ØØ137                         *
ØØ138                         *  INCH-INPUT ONE CHAR FROM THE CONSOLE IN THE
ØØ139                         *  A REGISTER
ØØ140                         *  VOLATILE: A
ØØ141                         *
ØØ142A F045 3F                 INCH   SWI
ØØ143A F046      ØØ      A             FCB    INCHC
ØØ144A F047 39                        RTS
ØØ145                         *
ØØ146                         *  OUTCH-OUTPUT ONE ASCII CHARACTER IN THE A
ØØ147                         *  REGISTER TO THE CONSOLE.
ØØ148                         *  VOLATILE: NONE
ØØ149                         *
ØØ150A F048 3F                 OUTCH  SWI
ØØ151A F049      Ø1      A             FCB    OUTCHC
ØØ152A F04A 39                        RTS
ØØ153                         *
ØØ154                         *  PDATA1-SEND A STRING POINTED TO BY THE X REGISTER
ØØ155                         *  AND TERMINATED BY AN ASCII EOT (Ø4) TO THE
ØØ156                         *  CONSOLE PRECEDED BY A CRLF.
ØØ157                         *  VOLATILE: X,A
ØØ158                         *
ØØ159A F04B 3F                 PDATA1 SWI              SEND CRLF THEN PRINT STRING
ØØ160A F04C      Ø3      A             FCB    PDAT1C   FUNCTION COIDE
ØØ161A F04D 39                        RTS
ØØ162                         *
ØØ163                         *  PDATA-SAME AS ABOVE WITHOUT CRLF STRING.
ØØ164                         *
ØØ165A F04E 3F                 PDATA  SWI
ØØ166A F04F      Ø2      A             FCB    PDATAC
ØØ167A F050 39                        RTS
ØØ168                         *
ØØ169                         *  MONIT-DOES A SWI FUNCTION OF THE MONITOR WHICH
ØØ170                         *  CAUSES A RESTART OF THE MONITOR
ØØ171                         *
ØØ172A F051 3F                 MONIT  SWI
ØØ173A F052      Ø8      A             FCB    MONITC   FUNCTION CODE FOR RESTART
ØØ174                         *
```

Figure 7. Interactive Graphics System Program Listing (Sheet 3 of 20)

```
PAGE  004  GRAFPAKA.SA:1  GRAFPA

00175                          *
00176                          *  GRAPH-ENTRY ROUTINE INTO GRAPHICS MODE.
00177                          *  POINTERS AND VARIABLES ARE INITIALIZED AND THE
00178                          *  USER IS PROMPTED TO SUPPLY A STARTING ADDRESS
00179                          *  FOR THE LINE LIST TO BE STORED AT.  THIS ROUTINE
00180                          *  IS SELF-CONTATAINED EXCEPT FOR THE
00181                          *  MONITOR FUNCTIONS  ALREADY MENTIONED AND
00182                          *  AND HAS ITS OWN COMMAND HANDLER
00183                          *  AND COMMAND LISTS.
00184                          *
00185A F053 10FF DD13     A GRAPH   STS    STACK    SAVE STACK
00186A F057 8E   D000     A         LDX    #ROM2-$2000 RAM FOR LIST TABLE
00187A F05A BF   DD18     A         STX    LISTP    SETUP LIST TABLE POINTER
00188A F05D BF   DD16     A         STX    NEXTAD   ALSO START OF TABLE
00189A F060 7F   DD00     A         CLR    FLAG     DEFAULT FOR LINGEN
00190A F063 7F   DD1A     A         CLR    LISCNT
00191A F066 86   7F       A         LDA    #$7F     START CURSOR CHAR
00192A F068 B7   DD23     A         STA    CURCH
00193A F06B 7F   DD15     A         CLR    MODE     DEFAULT TO CURSOR
00194A F06E 8E   9BF0     A         LDX    #SCRN+$BF0 CENTER OF SCREEN
00195A F071 BF   DD21     A         STX    CURAD    SAVE IN CURSOR ADDR
00196A F074 8E   DD35     A         LDX    #SAVC    AREA
00197A F077 BF   DD1B     A         STX    CURSAV   ADDR TO SAVE UNDER CURSOR
00198A F07A 7F   DD1D     A         CLR    CURPOS   CURSOR IN CENTER OF SCREEN
00199A F07D 7F   DD1E     A         CLR    CURPOS+1
00200A F080 BD   F140     A         JSR    CLEARS   CLEAR SCREEN
00201A F083 8E   F0CC     A GRAPH1  LDX    #MSG10   SIGN-ON MESSAGE
00202A F086 BD   F04B     A         JSR    PDATA1
00203A F089 8E   F0EE     A GRAPH2  LDX    #MSG15   "ENTER START LIST ADDRESS"
00204A F08C BD   F04B     A         JSR    PDATA1
00205A F08F BD   F033     A         JSR    GETEXP   GET LIST ADDR FROM USER
00206A F092 26   F5     F089        BNE    GRAPH2
00207A F094 1F   02       A         TFR    D,Y      GET ADDR IN Y FOR NEXTL1
00208A F096 BD   F5B8     A         JSR    NEXTL1   SETUP NEXTPT AND NEXTAD
00209A F099 BD   F2DA     A GRAPH3  JSR    PROMPT   PUT OUT PROMPT
00210A F09C 10FE DD13     A GRAPH4  LDS    STACK    RESTORE STACK ON ABORT
00211A F0A0 86   FF       A         LDA    #$FF
00212A F0A2 B7   DFF4     A         STA    ECHO     INHIBIT ECHO
00213A F0A5 BD   F045     A         JSR    INCH     GET COMMAND
00214A F0A8 81   10       A         CMPA   #$10
00215A F0AA 25   1B     F0C7        BLO    GRAPH6   CONTROL CHAR
00216A F0AC BD   F048     A         JSR    OUTCH    ECHO CHAR IF NOT CONTRL
00217A F0AF 7F   DFF4     A         CLR    ECHO     ECHO NEXT COMMANDS
00218A F0B2 8E   F108     A         LDX    #COMLIS  GET BASE ADDR OF COMMAND LIST
00219A F0B5 A1   84       A GRAPH5  CMPA   0,X      CHECK FOR MATCH
00220A F0B7 27   09     F0C2        BEQ    FOUND    GOT IT
00221A F0B9 30   03       A         LEAX   3,X      BUMP POINTER
00222A F0BB 8C   F132     A         CMPX   #COMEND
00223A F0BE 26   F5     F0B5        BNE    GRAPH5   BACK FOR MORE
00224A F0C0 20   D7     F099        BRA    GRAPH3   NOT VALID COMMAND
00225A F0C2 AD   98 01    A FOUND   JSR    [1,X]
00226A F0C5 20   D2     F099        BRA    GRAPH3   BACK FOR NEXT COMMAND
00227A F0C7 17   02B2   F37C GRAPH6  LBSR   MOVEC    CHECK FOR CURSOR CONTRL
00228A F0CA 20   D0     F09C        BRA    GRAPH4   BACK FOR NEXT COMMAND
00229                          *
00230                          *
00231                          *
00232                          *
```

Figure 7. Interactive Graphics System Program Listing (Sheet 4 of 20)

```
PAGE  005  GRAFPAKA.SA:1  GRAFPA

00233A F0CC       36      A MSG10  FCC    /6809-6847 GRAPHICS LINE PROCESSOR/
00234A F0ED       04      A        FCB    4
00235                       *
00236A F0EE       45      A MSG15  FCC    /ENTER START LIST ADDRESS /
00237A F107       04      A        FCB    4
00238                       *
00239                       *
00240A F108       50      A COMLIS FCC    /P/
00241A F109       F2F9    A        FDB    FIXP      FIX POINT AND RETURN
00242A F10B       4C      A        FCC    /L/
00243A F10C       F317    A        FDB    LINER     FIX LINE RETURN
00244A F10E       44      A        FCC    /D/
00245A F10F       F23D    A        FDB    DHAND     HANDLE D COMMNANDS
00246A F111       4E      A        FCC    /N/
00247A F112       F59F    A        FDB    NEXTL     START NEW LIST
00248A F114       45      A        FCC    /E/
00249A F115       F140    A        FDB    CLEARS
00250A F117       53      A        FCC    /S/       STATUS REPORT
00251A F118       F147    A        FDB    STATUS
00252A F11A       1B      A        FCB    $1B       'ESC'
00253A F11B       F1D8    A        FDB    EXIT      EXIT TO CALLING ROUTINE
00254A F11D       12      A        FCB    $12       CNTRL-R RUBBER-BAND MODE
00255A F11E       F139    A        FDB    RBAND
00256A F120       3E      A        FCC    />/
00257A F121       F1DB    A        FDB    RIGHT     MOVE CURSOR QUICK RIGHT
00258A F123       3C      A        FCC    /</
00259A F124       F1E8    A        FDB    LEFT      MOVE CURSOR QUICK LEFT
00260A F126       54      A        FCC    /T/
00261A F127       F1F5    A        FDB    UP        MOVE CURSOR QUICK UP
00262A F129       42      A        FCC    /B/
00263A F12A       F202    A        FDB    DWN       MOVE CURSOR QUICK DOWN
00264A F12C       48      A        FCC    /H/
00265A F12D       F20F    A        FDB    HOME      HOME CURSOR
00266A F12F       52      A        FCC    /R/       RESET AND RESTART
00267A F130       F2E3    A        FDB    RESET
00268             F132    A COMEND EQU    *
00269                       *
00270                       *
00271A F132 81    18      A ABORT  CMPA   #$18      CAN?
00272A F134 26    02    F138       BNE    ABORT2
00273A F136 35    10      A        PULS   X         CLEAN RETURN ADDR
00274A F138 39              ABORT2 RTS
00275                       *
00276                       *  RBAND-WILL TOGGLE THE MODE BYTE TO CHANGE FROM
00277                       *  RUBBER BAND MODE TO CURSOR OR VICE-VERSA.
00278                       *
00279A F139 73    DD15    A RBAND  COM    MODE
00280A F13C BD    F37C    A        JSR    MOVEC
00281A F13F 39                     RTS
00282                       *
00283A F140 BD    F5FD    A CLEARS JSR    CLEAR
00284A F143 BD    F414    A        JSR    WRITEC
00285A F146 39                     RTS
00286                       *
00287                       *
00288                       *  STATUS-STATUS REPORT OF SYSTEM
00289                       *  ADDRESS OF STARTING LIST
00290                       *  ADDRESS OF NEXT ENTRY INTO LIST TABLE
```

Figure 7. Interactive Graphics System Program Listing (Sheet 5 of 20)

```
PAGE  006  GRAFPAKA.SA:1  GRAFPA

00291                    *  ADDRES OF NEXT FREE LIST POINT
00292                    *
00293A F147 8E   F17A    A STATUS LDX    #MSG18   "START LIST ADDR= $"
00294A F14A BD   F04B    A        JSR    PDATA1
00295A F14D AE   9F DD18 A        LDX    [LISTP]
00296A F151 BD   F023    A        JSR    OUT4HS
00297A F154 8E   F18D    A        LDX    #MSG19   "NEXT LIST TABLE ENTRY AT $"
00298A F157 BD   F04B    A        JSR    PDATA1
00299A F15A BE   DD16    A        LDX    NEXTAD
00300A F15D BD   F023    A        JSR    OUT4HS
00301A F160 8E   F1A8    A        LDX    #MSG20   "NEXT LIST POINT AT $"
00302A F163 BD   F04B    A        JSR    PDATA1
00303A F166 BE   DD1F    A        LDX    NEXTPT
00304A F169 BD   F023    A        JSR    OUT4HS
00305A F16C 8E   F1BD    A        LDX    #MSG25   "CURRENT NUMBER OF LISTS= "
00306A F16F BD   F04B    A        JSR    PDATA1
00307A F172 B6   DD1A    A        LDA    LISCNT
00308A F175 4C           A        INCA            ACTUALLY ONE MORE
00309A F176 BD   F018    A        JSR    OUTHEX
00310A F179 39           A        RTS
00311                    *
00312A F17A      53      A MSG18  FCC    /START LIST ADDR= $/
00313A F18C      04      A        FCB    4
00314A F18D      4E      A MSG19  FCC    /NEXT LIST TABLE ENTRY AT $/
00315A F1A7      04      A        FCB    4
00316A F1A8      4E      A MSG20  FCC    /NEXT LIST POINT AT $/
00317A F1BC      04      A        FCB    4
00318A F1BD      43      A MSG25  FCC    /CURRENT NUMBER OF LISTS= $/
00319A F1D7      04      A        FCB    4
00320                    *
00321                    *
00322A F1D8 7E   F051    A EXIT   JMP    MONIT    BACK TO MONITOR
00323                    *
00324                    *  FASMOV PROVIDES AUXILIARY CURSOR FUNCTIONS.
00325                    *  THE SPECIFIED FUNCTION TO BE PERFORMED AND
00326                    *  THE INPUT ROUTINE IS CALLED FOR THE NEXT
00327                    *  COMMAND UNTIL A NON-FAST MOVE
00328                    *  CHARACTER IS ISSUED.
00329                    *  T-MOVE THE CURSOR TO THE TOP OF THE SCREEN
00330                    *  B-MOVE THE CURSOR TO THE BOTTOM OF THE SCREEN
00331                    *  L-MOVE THE CURSOR TO THE LEFT SIDE OF THE SCREEN
00332                    *  R-MOVE THE CURSOR TO THE RIGHT SIDE OF THE SCREEN
00333                    *  H-HOME MOVE THE CURSOR TO 0,0 (CENTER OF SCREEN)
00334                    *
00335A F1DB BD   F234    A RIGHT  JSR    CHKMOD   CHECK FOR R-BAND MODE
00336A F1DE BD   F400    A        JSR    ERASEC   GET RID OF CURSOR
00337A F1E1 86   7E      A        LDA    #$7E     FAR RIGHT MAX POS
00338A F1E3 B7   DD1D    A        STA    CURPOS   CURSOR X POSITION
00339A F1E6 20   33   F21B        BRA    FASMV1   CHECK FOR NEXT COMMAND
00340A F1E8 BD   F234    A LEFT   JSR    CHKMOD   SEE IF RUBBER-BAND
00341A F1EB BD   F400    A        JSR    ERASEC   GET RID OF CURSOR
00342A F1EE 86   81      A        LDA    #$81     FAR LEFT MAX POS
00343A F1F0 B7   DD1D    A        STA    CURPOS   CURSOR X POS
00344A F1F3 20   26   F21B        BRA    FASMV1   NEXT
00345A F1F5 BD   F234    A UP     JSR    CHKMOD   CHECK MODE
00346A F1F8 BD   F400    A        JSR    ERASEC   REMOVE CURSOR
00347A F1FB 86   5F      A        LDA    #$5F     TOP POSITION
00348A F1FD B7   DD1E    A        STA    CURPOS+1 CURSOR Y POS
```

Figure 7. Interactive Graphics System Program Listing (Sheet 6 of 20)

```
PAGE  007  GRAFPAKA.SA:1  GRAFPA

00349A F200 20    19    F21B        BRA     FASMV1    NEXT
00350A F202 BD    F234     A DWN    JSR     CHKMOD    CHECK MODE
00351A F205 BD    F400     A        JSR     ERASEC    TAKE CURSOR OFF
00352A F208 86    A4       A        LDA     #$A4      BOTTOM OF SCREEN
00353A F20A B7    DD1E     A        STA     CURPOS+1  CURSOR Y POS
00354A F20D 20    0C    F21B        BRA     FASMV1    NEXT
00355A F20F BD    F234     A HOME   JSR     CHKMOD
00356A F212 BD    F400     A        JSR     ERASEC
00357A F215 CC    0000     A        LDD     #00       CENTER CO-ORDINATES
00358A F218 FD    DD1D     A        STD     CURPOS    PUT IN CURSOR
00359A F21B FC    DD1D     A FASMV1 LDD     CURPOS    GET FINAL POSITION
00360A F21E BD    F61D     A        JSR     PNTADD    GET ADDRESS
00361A F221 BF    DD21     A        STX     CURAD     SAVE FOR USE
00362A F224 43                      COMA              INVERT PIXEL
00363A F225 B7    DD23     A        STA     CURCH     NEW CURSOR CHAR
00364A F228 BD    F414     A        JSR     WRITEC    PUT CURSOR BACK
00365A F22B 7D    DD15     A        TST     MODE      CHECK FOR R-BAND
00366A F22E 27    03    F233        BEQ     FASRET    NO
00367A F230 BD    F2AC     A        JSR     DRAWL     DRAW THE LINE
00368A F233 39                   FASRET RTS           BACK TO COMHAND
00369                            *
00370                            *
00371                            *  CHKMOD-CHECKS TO SEE IF IN RUBBER
00372                            *  BAND MODE AND, IF SO,
00373                            *  TO ERASE THE CURRENT LINE.
00374                            *
00375A F234 7D    DD15     A CHKMOD TST     MODE      RUBBER BAND ?
00376A F237 27    03    F23C        BEQ     CHRET     NOPE
00377A F239 BD    F2B1     A        JSR     ERASEL    YES, ERASE LINE
00378A F23C 39                   CHRET  RTS
00379                            *
00380                            *
00381                            *    DHAND  A "D" COMMAND CAUSES CONTROL TO BE
00382                            *    PASSED HERE FOR FURTHER PROCESSING.
00383                            *    D <SP>  DELETE LAST LINE, PUT CURSOR AT
00384                            *             LAST POINT
00385                            *    DEL     DELETE SPECIFIED LIST NUMBER
00386                            *    DR <SP> DRAW CURRENT LIST
00387                            *    DR <LIST#,LIST#,..> DRAW SPECIFIED LISTS
00388                            *    DA      DRAW ALL LISTS
00389                            *    DW      DRAW A LINE TO BE SPECIFIED
00390                            *
00391A F23D BD    F045     A DHAND  JSR     INCH      GET NEXT CHAR
00392A F240 BD    F132     A        JSR     ABORT     CHECK FOR "CAN" ABORT CHAR
00393A F243 81    20       A        CMPA    #$20      SPACE?
00394A F245 26    1F    F266        BNE     DH1       NOPE
00395A F247 BD    F400     A        JSR     ERASEC    REMOVE CURSOR
00396A F24A 86    FF       A        LDA     #$FF      FLAG=UNWRITE
00397A F24C FE    DD1F     A        LDU     NEXTPT    GET ADDR OF NEXTPT IN LIST
00398A F24F 33    5E       A        LEAU    -2,U      DELETE POINT
00399A F251 FF    DD1F     A        STU     NEXTPT    UPDATE POINTER
00400A F254 10AE  5E       A        LDY     -2,U      GET X1Y1
00401A F257 AE    C4       A        LDX     0,U       GET X2,Y2
00402A F259 BD    F645     A        JSR     LINGEN    UNDRAW LINE
00403A F25C BD    F414     A        JSR     WRITEC    PUT CURSOR BACK
00404A F25F 8E    F28D     A        LDX     #MSG30    "LINE DELETED"
00405A F262 BD    F04E     A        JSR     PDATA
00406A F265 39                      RTS               MORE COMMANDS
```

Figure 7. Interactive Graphics System Program Listing (Sheet 7 of 20)

```
PAGE  008  GRAFPAKA.SA:1  GRAFPA

00407A F266 81   45      A DH1    CMPA   #'E      AN "E"?
00408A F268 26   0A   F274        BNE    DH2
00409A F26A BD   F045    A        JSR    INCH     GET NEXT CHAR
00410A F26D 81   4C      A        CMPA   #'L
00411A F26F 26   1B   F28C        BNE    DHRET
00412A F271 BD   F43E    A        JSR    DELIST   DELETE A LIST
00413A F274 81   52      A DH2    CMPA   #'R
00414A F276 26   04   F27C        BNE    DH3
00415A F278 BD   F4F3    A        JSR    GETLIS   GET NUMBER AND DRAW
00416A F27B 39                    RTS
00417A F27C 81   41      A DH3    CMPA   #'A      DRAW ALL LISTS
00418A F27E 26   04   F284        BNE    DH4
00419A F280 BD   F29D    A        JSR    DRALL
00420A F283 39                    RTS
00421A F284 81   57      A DH4    CMPA   #'W      DRAW A LINE
00422A F286 26   04   F28C        BNE    DHRET
00423A F288 BD   F733    A        JSR    DRAW     LINE DRAWING ROUTINE
00424A F28B 39                    RTS
00425A F28C 39                DHRET RTS
00426                         *
00427A F28D      20      A MSG30  FCC    /   LINE DELETED/
00428A F29C      04      A        FCB    4
00429                         *
00430                         *  DRALL-DRAW ALL LISTS.  ALL THE LISTS ARE DRAWN
00431                         *  ON THE SCREEN.
00432                         *
00433A F29D F6   DD1A    A DRALL  LDB    LISCNT   GET LIST COUNT
00434A F2A0 5C                    INCB             REAL LIST NUM
00435A F2A1 34   04      A DRA1   PSHS   B        SAVE COUNTER
00436A F2A3 BD   F519    A        JSR    GETL3    DRAW LIST IN B
00437A F2A6 35   04      A        PULS   B        GET COUNTER
00438A F2A8 5A                    DECB             DECREMENT COUNTER
00439A F2A9 26   F6   F2A1        BNE    DRA1     FINISHED
00440A F2AB 39                    RTS
00441                         *
00442                         *  DRAWL  DRAW A LINE BETWEEN THE CURSOR AND THE LAST
00443                         *  POINT IN THE CURRENT LIST.
00444                         *
00445                         *
00446A F2AC 4F                DRAWL  CLRA             FLAG=WRITE
00447A F2AD 34   02      A        PSHS   A        SAVE FLAG
00448A F2AF 20   04   F2B5        BRA    DRAWL2
00449A F2B1 86   FF      A ERASEL LDA    #$FF     FLAG=UNWRITE
00450A F2B3 34   02      A        PSHS   A        SAVE FLAG
00451A F2B5 BD   F400    A DRAWL2 JSR    ERASEC   GET RID OF CURSOR
00452A F2B8 FC   DD1D    A        LDD    CURPOS   GET CURSOR POSITION
00453A F2BB BE   DD1F    A        LDX    NEXTPT
00454A F2BE 10AE 1E      A        LDY    -2,X     GET LAST POINT
00455A F2C1 108C 0071    A        CMPY   #$71
00456A F2C5 27   0F   F2D6        BEQ    DRET
00457A F2C7 108C 006F    A        CMPY   #$6F     CHECK FOR MOVE PEN
00458A F2CB 26   02   F2CF        BNE    DRAW3    NOPE
00459A F2CD 1F   02      A        TFR    D,Y      YEP, DRAW DOT
00460A F2CF 1F   01      A DRAW3  TFR    D,X      2ND POINT
00461A F2D1 35   02      A        PULS   A        GET FLAG BACK
00462A F2D3 BD   F645    A        JSR    LINGEN   DRAW LINE
00463A F2D6 BD   F414    A DRET   JSR    WRITEC
00464A F2D9 39                    RTS              BACK
```

Figure 7. Interactive Graphics System Program Listing (Sheet 8 of 20)

```
PAGE  009  GRAFPAKA.SA:1  GRAFPA

00465                          *
00466                          *
00467                          *    PROMPT WRITES THE PROMPT CHAR (%) AND CRLF
00468                          *    AND RETURNS.
00469                          *
00470A F2DA BD    F003     A PROMPT JSR    CRLF
00471A F2DD 86    25       A        LDA    #'%
00472A F2DF BD    F048     A        JSR    OUTCH
00473A F2E2 39                    RTS
00474                          *
00475                          *  RESET WILL REST ALL THE POINTERS AND START FRESH
00476                          *
00477A F2E3 BD    F045     A RESET  JSR    INCH     GET NEXT CHAR IN COMM
00478A F2E6 81    45       A        CMPA   #'E
00479A F2E8 26    0E    F2F8       BNE    RRET
00480A F2EA BD    F045     A        JSR    INCH
00481A F2ED 81    53       A        CMPA   #'S
00482A F2EF 26    07    F2F8       BNE    RRET
00483A F2F1 10FE DD13      A        LDS    STACK    GET USER STACK
00484A F2F5 7E    F053     A        JMP    GRAPH    BACK TO RESTART
00485A F2F8 39                 RRET  RTS
00486                          *
00487                          *
00488                          *
00489                          *  LINER  PUT CURSOR LOCATION AS NEXT
00490                          *  POINT IN CURRENT LIST AND DRAW
00491                          *  A LINE TO IT FROM THE PREVIOUS POINT.
00492                          *
00493                          *
00494                          *  FIXP  PUT MOVE PEN CODE IN NEXT POINT AND THEN PUT
00495                          *        CURRENT CURSOR POSITION IN POINT LIST AND
00496                          *        PLOT A POINT AT CURPOS;  WRITE THE CURSOR
00497                          *        AND RETURN TO COMMAND HANDLER.
00498                          *
00499A F2F9 7D    DD15     A FIXP   TST    MODE     CHECK FOR R-BAND
00500A F2FC 27    03    F301       BEQ    FIX
00501A F2FE BD    F2B1     A        JSR    ERASEL
00502A F301 8E    F32F     A FIX    LDX    #MSG26   "POINT FIXED AT $"
00503A F304 BD    F04E     A        JSR    PDATA
00504A F307 BD    F357     A        JSR    PCURSE   PRINT POINT POS
00505A F30A FE    DD1F     A        LDU    NEXTPT
00506A F30D 8E    006F     A FIX1   LDX    #$6F     MOVE PEN COMMAND
00507A F310 AF    C1       A        STX    0,U++    PUT IN LIST,INC POINTER
00508A F312 FF    DD1F     A        STU    NEXTPT   UPDATE POINTER
00509A F315 20    09    F320       BRA    LINER2
00510A F317 8E    F343     A LINER  LDX    #MSG27   "LINE ENTERED AT"
00511A F31A BD    F04E     A        JSR    PDATA
00512A F31D BD    F357     A        JSR    PCURSE
00513A F320 BD    F2AC     A LINER2 JSR    DRAWL    DRAW LINE OR PUT DOT
00514A F323 FE    DD1F     A        LDU    NEXTPT   GET POINTER
00515A F326 BE    DD1D     A        LDX    CURPOS
00516A F329 AF    C1       A        STX    0,U++    PUT IN POINT, INC POINTER
00517A F32B FF    DD1F     A        STU    NEXTPT   UPDATE POINTER
00518A F32E 39                 FIXRET RTS
00519                          *
00520A F32F      20       A MSG26  FCC    /  POINT FIXED AT  /
00521A F342      04       A        FCB    4
00522A F343      20       A MSG27  FCC    / LINE ENTERED AT /
```

Figure 7. Interactive Graphics System Program Listing (Sheet 9 of 20)

```
PAGE  010  GRAFPAKA.SA:1  GRAFPA

00523A F356      04      A        FCB    4
00524                     *
00525                     *  PCURSE-PRINTS THE CURRENT X,Y POSITION OF THE
00526                     *  CURSOR
00527                     *
00528A F357 8E   F370    A PCURSE LDX    #MSG31   " X=$"
00529A F35A BD   F04E    A        JSR    PDATA
00530A F35D B6   DD1D    A        LDA    CURPOS   GET X POSITION
00531A F360 BD   F018    A        JSR    OUTHEX   PRINT IT
00532A F363 8E   F376    A        LDX    #MSG32   " ,Y=$"
00533A F366 BD   F04E    A        JSR    PDATA
00534A F369 B6   DD1E    A        LDA    CURPOS+1 Y POSITION
00535A F36C BD   F018    A        JSR    OUTHEX
00536A F36F 39                    RTS
00537                     *
00538A F370      20      A MSG31  FCC    /  X=$/
00539A F375      04      A        FCB    4
00540A F376      20      A MSG32  FCC    / ,Y=$/
00541A F37B      04      A        FCB    4
00542                     *
00543                     *  MOVEC  UPDATES THE POSITION OF THE
00544                     *  CURSOR ACCORDING TO THE KEYBOARD
00545                     *  INPUT FROM THE USER.
00546                     *
00547                     *  ON ENTRY  A=CONTROL CHARACTER
00548                     *
00549                     *  CNTRL-H  MOVE CURSOR LEFT 1 PIXEL SPACE
00550                     *  CNTRL-L  MOVE CURSOR RIGHT 1 PIXEL SPACE
00551                     *  CNTRL-K  MOVE CURSOR UP ONE SCAN LINE
00552                     *  CNTRL-J  MOVE CURSOR DOWN ONE SCAN LINE
00553                     *
00554                     *  CHECK IS MADE TO MAKE SURE THAT THE
00555                     *  CUROSR STAYS ON THE SCREEN.
00556                     *
00557A F37C 34   02      A MOVEC  PSHS   A        SAVE CON CAHR
00558A F37E BD   F234    A        JSR    CHKMOD   CHECK MODE
00559A F381 BD   F400    A        JSR    ERASEC   REMOVE CURSOR
00560A F384 BE   DD21    A MOV2   LDX    CURAD    GET CURSOR ADDR
00561A F387 35   02      A        PULS   A        GET CHAR BACK
00562A F389 F6   DD1D    A        LDB    CURPOS   GET X POS IN B
00563A F38C 81   08      A        CMPA   #8       (CNTRL-H) MOVE LEFT?
00564A F38E 26   1D   F3AD        BNE    MOVE1
00565A F390 5A                    DECB            YES,DEC X POS ONE
00566A F391 C1   80      A        CMPB   #$80     LEFT BORDER?
00567A F393 27   5C   F3F1        BEQ    CURET2   YES,TOO MUCH LEAVE IT
00568A F395 B6   DD23    A        LDA    CURCH    GET CURSOR CHAR
00569A F398 1A   01      A        ORCC   #1       SET CARRY
00570A F39A 49                    ROLA            CHECK TO SEE IF
00571A F39B 25   01   F39E        BCS    MOK1     CURSOR CROSSED A BYTE
00572A F39D 49                    ROLA            BOUNDARY
00573A F39E B7   DD23    A MOK1   STA    CURCH
00574A F3A1 1F   98      A        TFR    B,A      GET XPOS IN A
00575A F3A3 84   07      A        ANDA   #7       MASK TOP 5 BITS
00576A F3A5 88   07      A        EORA   #7       SEE IF 3 LSBITS ZERO
00577A F3A7 26   45   F3EE        BNE    CURET1   IF SO CROSSED BYTE BOUNDARY
00578A F3A9 30   1F      A        LEAX   -1,X     DEC CURSOR ADDR
00579A F3AB 20   41   F3EE        BRA    CURET1   OK, UPDATE POSITION
00580A F3AD 81   0C      A MOVE1  CMPA   #$0C     (CNTRL-L) MOVE RIGHT?
```

Figure 7. Interactive Graphics System Program Listing (Sheet 10 of 20)

```
PAGE  011  GRAFPAKA.SA:1  GRAFPA

00581A F3AF 26   1B   F3CC        BNE   MOVE2
00582A F3B1 5C                   INCB                INC X POS ONE
00583A F3B2 C1   7F      A       CMPB  #$7F          RIGHT BORDER?
00584A F3B4 27   3B   F3F1       BEQ   CURET2        TOO MUCH
00585A F3B6 B6   DD23    A       LDA   CURCH
00586A F3B9 1A   01      A       ORCC  #1
00587A F3BB 46                   RORA
00588A F3BC 25   01   F3BF       BCS   MOK2
00589A F3BE 46                   RORA
00590A F3BF B7   DD23    A MOK2  STA   CURCH
00591A F3C2 1F   98      A       TFR   B,A
00592A F3C4 84   07      A       ANDA  #7
00593A F3C6 26   26   F3EE       BNE   CURET1
00594A F3C8 30   01      A       LEAX  1,X           INC CURAD
00595A F3CA 20   22   F3EE       BRA   CURET1        UPDATE
00596A F3CC F6   DD1E    A MOVE2 LDB   CURPOS+1      GET Y POS IN B
00597A F3CF 81   0B      A       CMPA  #$0B          (CNTRL-K) MOVE UP?
00598A F3D1 26   0A   F3DD       BNE   MOVE3
00599A F3D3 5C                   INCB                IN Y POS ONE
00600A F3D4 C1   5F      A       CMPB  #$5F          TOP BORDER?
00601A F3D6 2E   19   F3F1       BGT   CURET2        YEP
00602A F3D8 30   88 E0   A       LEAX  -$20,X
00603A F3DB 20   0C   F3E9       BRA   CURET         OK, UPDATE
00604A F3DD 81   0A      A MOVE3 CMPA  #$0A          (CNTRL-J) MOVE DOWN?
00605A F3DF 26   10   F3F1       BNE   CURET2        ERROR, RETURN UNCHANGED
00606A F3E1 5A                   DECB                DEC Y POS ONE
00607A F3E2 C1   A4      A       CMPB  #$A4          BOTTOM BORDER?
00608A F3E4 2D   0B   F3F1       BLT   CURET2        YES
00609A F3E6 30   88 20   A       LEAX  $20,X
00610A F3E9 F7   DD1E    A CURET STB   CURPOS+1
00611A F3EC 20   03   F3F1       BRA   CURET2
00612A F3EE F7   DD1D    A CURET1 STB  CURPOS       UPDATE X POS
00613A F3F1 BF   DD21    A CURET2 STX  CURAD        UPDATE CURSOR ADDR
00614A F3F4 BD   F414    A       JSR   WRITEC        PUT CURSOR BACK
00615A F3F7 7D   DD15    A       TST   MODE
00616A F3FA 27   03   F3FF       BEQ   CURET3
00617A F3FC BD   F2AC    A       JSR   DRAWL         YES,DRAW CURRENT LINE
00618A F3FF 39              CURET3 RTS               RETURN   FOR NEXT COMMAND
00619                       *
00620                       *
00621                       *  ERASEC  ERASE CURSOR FROM SCREEN BY REPLACING
00622                       *          IT WITH PREVIOUSLY SAVED PIXELS.
00623                       *
00624A F400 BE   DD21    A ERASEC LDX   CURAD        GET CURSOR ADDR
00625A F403 10BE DD1B    A       LDY   CURSAV       WHERE TO SAVE PIXELS
00626A F407 C6   04      A       LDB   #4           COUNTER
00627A F409 A6   A0      A ERAC1 LDA   ,Y+          GET BYTE
00628A F40B A7   84      A       STA   ,X           SAVE IT
00629A F40D 30   88 20   A       LEAX  $20,X        NEXT LINE
00630A F410 5A                   DECB                FINSHED ALL LINES?
00631A F411 26   F6   F409       BNE   ERAC1
00632A F413 39                   RTS
00633                       *
00634                       *
00635                       *  WRITEC   PUT CURSOR ON SCREEN AT CURRENT X-Y
00636                       *           LOCATION IN CURPOS, SAVE INFORMATION
00637                       *           UNDERNEATH.[H.
00638                       *
```

Figure 7. Interactive Graphics System Program Listing (Sheet 11 of 20)

```
PAGE  012  GRAFPAKA.SA:1  GRAFPA

00639A F414 BD   F42A    A WRITEC JSR    SAVEC    SAVE PIXELS UNDER CURSOR
00640A F417 BE   DD21    A        LDX    CURAD    CURSOR ADDR
00641A F41A C6   04      A        LDB    #4       LINE COUNT
00642A F41C B6   DD23    A WRIT1  LDA    CURCH    GET CURSOR CHAR
00643A F41F A4   84      A        ANDA   ,X       MERGE WITH PIXELS ON SRN
00644A F421 A7   84      A        STA    ,X
00645A F423 30   88 20   A        LEAX   $20,X    NEXT LINE
00646A F426 5A                    DECB
00647A F427 26   F3   F41C        BNE    WRIT1
00648A F429 39                    RTS
00649                        *
00650                        *  SAVEC WILL SAVE THE PIXELS UNDER
00651                        *  THE LOCATION OF THE CURSOR AT
00652                        *  CURAD, AND PUT THEM IN LOCATION CURSAV.
00653                        *
00654A F42A BE   DD21    A SAVEC  LDX    CURAD    GET CURSOR ADDR
00655A F42D 10BE DD1B    A        LDY    CURSAV   WHERE TO SAVE THEM
00656A F431 C6   04      A        LDB    #4       LINE CNT
00657A F433 A6   84      A SAV1   LDA    ,X
00658A F435 A7   A0      A        STA    ,Y+
00659A F437 30   88 20   A        LEAX   $20,X
00660A F43A 5A                    DECB
00661A F43B 26   F6   F433        BNE    SAV1
00662A F43D 39                    RTS
00663                        *
00664                        *  DL   DELETE A LIST FROM LIST TABLE.
00665                        *  "DELETE LIST #" WILL BE PRINTED. ANY NON-HEX
00666                        *  CHARACTER WILL CAUSE AN ABORT,  A VALID LIST
00667                        *  WILL CAUSE THAT LIST TO BE DELETED IF IT EXISTS
00668                        *  AND THE MESSAGE "LIST #   DELETED" TO BE PRINTED.
00669                        *  IF IT DOES NOT EXIST, A MESSAGE SO STATING IS
00670                        *  PRINTED.  IN ANY CASE IF NOT LIST IS ACTUALLY
00671                        *  DELETED, THE MESSAGE "NO LIST DELETED" IS PRINTED.
00672                        *
00673A F43E 8E   F4C7    A DELIST LDX    #MSG11
00674A F441 BD   F04B    A        JSR    PDATA1   DELETE LIST #"
00675A F444 BD   F033    A        JSR    GETEXP
00676A F447 27   02   F44B        BEQ    DELIS1
00677A F449 20   24   F46F        BRA    DLRET    NOT VALID HEX
00678A F44B 5D               DELIS1 TSTB
00679A F44C 27   06   F454        BEQ    NOLIST   CAN NEVER HAVE ZERO LISTS
00680A F44E 5A                    DECB            ONE LESS THAN ACTUAL
00681A F44F F1   DD1A    A        CMPB   LISCNT   IS VALID LIST #?
00682A F452 23   08   F45C        BLS    DELIS2   YES
00683A F454 8E   F54A    A NOLIST LDX    #MSG13
00684A F457 BD   F04B    A        JSR    PDATA1   "NO SUCH LIST #"
00685A F45A 20   13   F46F        BRA    DLRET    NO.RETURN
00686A F45C 25   18   F476 DELIS2 BLO    DELIS3   OTHER THAN LATEST LIST
00687A F45E 5D                    TSTB            CAN'T DELETE LAST LIST
00688A F45F 27   0E   F46F        BEQ    DLRET
00689A F461 BE   DD16    A        LDX    NEXTAD
00690A F464 30   1E      A        LEAX   -2,X     SET NEXTAD BACK
00691A F466 AE   84      A        LDX    0,X      GET ADDRESS OF CURRENT LIST
00692A F468 BF   DD1F    A        STX    NEXTPT   RESET NEXT POINTER THERE
00693A F46B 34   04      A        PSHS   B        SAVE NUMBER TO PRINT
00694A F46D 20   42   F4B1        BRA    DELIS5   FINISH UP
00695A F46F 8E   F4DF    A DLRET  LDX    #MSG21   "LIST NOT DELETED"
00696A F472 BD   F04B    A        JSR    PDATA1
```

Figure 7. Interactive Graphics System Program Listing (Sheet 12 of 20)

```
PAGE  013  GRAFPAKA.SA:1  GRAFPA

00697A F475 39          DLRET2 RTS             BACK TO COMMAND HANDLER
00698A F476 BE   DD18    A DELIS3 LDX   LISTP   GET BASE OF LIST TABLE
00699A F479 34   04      A        PSHS  B       SAVE LIST NUMBER
00700A F47B 58                    LSLB          X2
00701A F47C 30   85      A        LEAX  B,X     POINT TO LIST TO BE DELETED
00702A F47E 10AE 84      A        LDY   0,X     HEAD OF LIST TO BE DELETED
00703A F481 EE   02      A        LDU   2,X     HEAD OF FOLLOWING LIST
00704A F483 A6   C0      A DELIS4 LDA   0,U+    GET POINT
00705A F485 A7   A0      A        STA   0,Y+    COPY POINT UP
00706A F487 11B3 DD1F    A        CMPU  NEXTPT  END OF LISTS?
00707A F48B 26   F6   F483        BNE   DELIS4
00708A F48D 10BF DD1F    A        STY   NEXTPT  NEW END OF LISTS
00709A F491 EC   84      A        LDD   ,X      GET POINTER TO LIST
00710A F493 34   06      A        PSHS  D       SAVE
00711A F495 EC   02      A        LDD   2,X     GET PTR TO NEXT LIST
00712A F497 A3   E4      A        SUBD  0,S     GET DIFFERENCE
00713A F499 ED   E4      A        STD   0,S     SAVE DIFF
00714A F49B 30   02      A        LEAX  2,X     POINT TO NEXT LIST
00715A F49D EC   84      A LOOP   LDD   ,X      GET POINTER
00716A F49F A3   E4      A        SUBD  ,S      SUB DIFF
00717A F4A1 ED   1E      A        STD   -2,X    PUT IN TABLE
00718A F4A3 30   02      A        LEAX  2,X
00719A F4A5 BC   DD16    A        CMPX  NEXTAD
00720A F4A8 26   F3   F49D        BNE   LOOP
00721A F4AA 30   1E      A        LEAX  -2,X
00722A F4AC BF   DD16    A        STX   NEXTAD
00723A F4AF 35   10      A        PULS  X       CLEAN STACK
00724A F4B1 7A   DD1A    A DELIS5 DEC   LISCNT  UPDATE LIST COUNTER
00725A F4B4 8E   F4CE    A        LDX   #MSG11+7
00726A F4B7 BD   F04B    A        JSR   PDATA1  "LIST # "
00727A F4BA 35   02      A        PULS  A       GET LIST NUMBER DELETED
00728A F4BC 4C                    INCA          LIST NUMBER
00729A F4BD BD   F018    A        JSR   OUTHEX  PRINT IT
00730A F4C0 8E   F4D6    A        LDX   #MSG12
00731A F4C3 BD   F04E    A        JSR   PDATA   "DELETED"
00732A F4C6 39                    RTS
00733A F4C7      44      A MSG11  FCC   /DELETE LIST # /
00734A F4D5      04      A        FCB   4
00735A F4D6      20      A MSG12  FCC   / DELETED/
00736A F4DE      04      A        FCB   4
00737                     *
00738A F4DF      20      A MSG21  FCC   / --LIST NOT DELETED/
00739A F4F2      04      A        FCB   4
00740                     *
00741                     *
00742                     *  GETLIS-WHEN DR IS TYPED IN THE MESSAGE
00743                     *  "DRAW LIST #" IS PRINTED.  IF A VALID
00744                     *  HEX NUMBER IS INPUT, THAT LIST IS DRAWN
00745                     *  IF IT EXISTS, OTHERWISE A MESSAGE IS
00746                     *  PRINTED TO THAT EFFECT.
00747                     *
00748A F4F3 8E   F592    A GETLIS LDX   #MSG14
00749A F4F6 BD   F04B    A        JSR   PDATA1  "DRAW LIST # "
00750A F4F9 CE   DD35    A        LDU   #BUFFER+16
00751A F4FC BD   F033    A GETL1  JSR   GETEXP  GET LIST NUMBER
00752A F4FF 26   30   F531        BNE   CURLIS  DEFAULT TO CURRENT LIS
00753A F501 36   04      A        PSHU  B
00754A F503 B6   DD24    A        LDA   DELIM
```

Figure 7. Interactive Graphics System Program Listing (Sheet 13 of 20)

```
PAGE  014  GRAFPAKA.SA:1  GRAFPA

00755A F506 81   2C      A        CMPA   #',      MORE LISTS
00756A F508 27   F2   F4FC        BEQ    GETL1
00757A F50A 37   04       A GETL2  PULU   B
00758A F50C 34   40       A        PSHS   U
00759A F50E 8D   09   F519        BSR    GETL3
00760A F510 35   40       A        PULS   U
00761A F512 1183 DD35     A        CMPU   #BUFFER+16
00762A F516 26   F2   F50A        BNE    GETL2
00763A F518 39                    RTS
00764A F519 5D              GETL3 TSTB
00765A F51A 27   14   F530        BEQ    GETR      NO LIST ZERO
00766A F51C 5A                    DECB               ACTUAL LIST NUM
00767A F51D F1   DD1A     A       CMPB   LISCNT
00768A F520 27   0F   F531        BEQ    CURLIS    CURRENT LIST
00769A F522 25   1A   F53E        BLO    GETL4     NOT CURRENT
00770A F524 1F   98       A NODR  TFR    B,A
00771A F526 4C                    INCA
00772A F527 BD   F00A     A       JSR    OUTHR     OUTPUT LIST NUMBER
00773A F52A 8E   F54A     A       LDX    #MSG13    NOT VALID LIST #
00774A F52D BD   F04E     A       JSR    PDATA     "  DOES NOT EXIST"
00775A F530 39              GETR  RTS
00776A F531 108E 0071     A CURLIS LDY   #$71      END LIST CODE
00777A F535 BE   DD1F     A       LDX    NEXTPT    POINTER TO LIST
00778A F538 10AF 84       A       STY    0,X       STORE TEMP END CODE
00779A F53B F6   DD1A     A       LDB    LISCNT
00780A F53E BE   DD18     A GETL4 LDX    LISTP     BASE ADD OF LIST TABLE
00781A F541 58                    LSLB               X2
00782A F542 30   85       A       LEAX   B,X       ADD OFFSET INTO TABEL
00783A F544 EE   84       A       LDU    0,X       GET ADDR OF LIST
00784A F546 BD   F560     A       JSR    DRWLIS    DRAW OBJECT
00785A F549 39                    RTS               BACK FOR NEXT COMM
00786                        *
00787A F54A       20      A MSG13 FCC    / DOES NOT EXIST/
00788A F559       04      A       FCB    4
00789                        *
00790                        *   DRWLIS
00791                        *  SUBROUTINE DRWLIS WILL PROCESS A LIST OF POINTS
00792                        *  ACCORDING TO THE FOLLOWING RULES:
00793                        *        FORMAT
00794                        *  THE LIST MUST BE A SET OF ALTERNATING X-Y POINTS
00795                        *  WITHIN THE BOUNDS DESCRIBED IN DRAW
00796                        *  (-128<X<127;-96<Y<95)
00797                        *  AND BE IN CONTIGUOUS MEMORY LOCATIONS.
00798                        *  FOR Y VALUES LESS THAN $60,
00799                        *  A LINE WILL BE DRAWN BERWEEN TWO CONSECU-
00800                        *  TIVE X-Y PAIRS AND THE CURSOR
00801                        *  MOVED TO THE LAST USED PAIR
00802                        *  FOR Y VALUES GREATER THAN $60
00803                        *  THE FOLLOWING CODES APPLY:
00804                        *  Y=$6F:  MOVE CURSOR TO NEXT POINT
00805                        *          WITHOUT DRAWING A LINE
00806                        *  Y=$70:  NEXT TWO BYTES POINT TO

00807A F55A 00   0000     A          A NEW LIST (INDIRECTION)
00808                        *  Y=$71:  END OF LIST, TERMINATES

00809A F55D 00   0000     A          LIST AND RETURNS
00810                        *
```

Figure 7. Interactive Graphics System Program Listing (Sheet 14 of 20)

```
PAGE  015  GRAFPAKA.SA:1  GRAFPA

00811                          *  ON ENTRY  U=ADDR OF LIST TO BE PROCESSED
00812                          *  ALL REGISTERS ZAPPED
00813                          *
00814A F560 BD   F400     A DRWLIS JSR    ERASEC    REMOVE CURSOR
00815A F563 33   5E       A        LEAU   -2,U      START BACK TO GET FIRST POINT
00816A F565 C6   60       A LIST0  LDB    #$60      GET MAX Y TO COMPARE
00817A F567 10AE C1       A LIST1  LDY    ,U++      GET X-Y PAIR,INC POINTER
00818A F56A E1   41       A        CMPB   1,U       CHECK Y
00819A F56C 2C   1A    F588        BGE    LIST4     OK TO DRAW
00820A F56E E6   41       A        LDB    1,U       GET SPECIAL CODE
00821A F570 C1   71       A        CMPB   #$71      END OF LIST CODE
00822A F572 26   04    F578        BNE    LIST2     CHECK MORE
00823A F574 BD   F414     A LISRET JSR    WRITEC    PUT CURSOR BACK
00824A F577 39                    RTS               RETURN
00825A F578 C1   6F       A LIST2  CMPB   #$6F      MOVE CURSOR?
00826A F57A 26   04    F580        BNE    LIST3
00827A F57C 33   42       A        LEAU   2,U       YES, BUP POINTER
00828A F57E 20   E5    F565        BRA    LIST0     BACK FOR MORE POINTS
00829A F580 C1   70       A LIST3  CMPB   #$70      INDIRECTION?
00830A F582 26   F0    F574        BNE    LISRET    ERROR,RETURN
00831A F584 EE   42       A        LDU    2,U       YES,GET ADDR NEXT LIST
00832A F586 20   DD    F565        BRA    LIST0     GO PROCESS IT
00833A F588 AE   C4       A LIST4  LDX    0,U       GET NEXT X-Y PAIR
00834A F58A B6   DD00     A        LDA    FLAG
00835A F58D BD   F645     A        JSR    LINGEN    DRAW LINE
00836A F590 20   D3    F565        BRA    LIST0     CONTINUE UNTIL DONE
00837                          *
00838A F592      44       A MSG14  FCC    /DRAW LIST # /
00839A F59E      04       A        FCB    4
00840                          *
00841                          *  NEXTL
00842                          *
00843                          *  "N" COMMAND NEXT LIST
00844                          *  THIS ROUTINE ENDS THE CURRENT LIST BY AFFIXING A
00845                          *  $71 IN THE Y POINT LOCATION
00846                          *  AND BEGINS A NEW LIST IMMEDIATELY FOLLOWING IT.
00847                          *  THE LIST COUNT AND LINE LIST TABLES ARE UPDATED
00848                          *
00849A F59F 8E   F5C6     A NEXTL  LDX    #MSG29    "END LIST #"
00850A F5A2 BD   F04E     A        JSR    PDATA
00851A F5A5 B6   DD1A     A        LDA    LISCNT
00852A F5A8 4C                    INCA
00853A F5A9 BD   F018     A        JSR    OUTHEX
00854A F5AC CC   0071     A        LDD    #$71      END LIST CODE
00855A F5AF 10BE DD1F     A        LDY    NEXTPT    GET POINTER
00856A F5B3 ED   A1       A        STD    0,Y++     END PRESENT LIST
00857A F5B5 7C   DD1A     A        INC    LISCNT    UP COUNT
00858A F5B8 10BF DD1F     A NEXTL1 STY    NEXTPT
00859A F5BC BE   DD16     A        LDX    NEXTAD    GET NEXT FREE SPACE IN TABLE
00860A F5BF 10AF 81       A        STY    0,X++     PUT ADDR IN TABLE
00861A F5C2 BF   DD16     A        STX    NEXTAD    UPDATE POINTER
00862A F5C5 39                    RTS
00863                          *
00864                          *
00865A F5C6      20       A MSG29  FCC    /  END OF LIST #/
00866A F5D5      04       A        FCB    4
00867                          *
00868                          *  MESSAGES FOLLOW
```

Figure 7. Interactive Graphics System Program Listing (Sheet 15 of 20)

```
PAGE  016  GRAFPAKA.SA:1  GRAFPA

00869                          *
00870A F5D6       49       A MSG5   FCC    /INPUT TWO POINTS/
00871A F5E6       0D       A        FCB    $0D,$0A
00872A F5E8       58       A MSG6   FCC    /X1,Y1 /
00873A F5EE       04       A        FCB    4
00874A F5EF       58       A MSG7   FCC    /X2,Y2 /
00875A F5F5       04       A        FCB    4
00876A F5F6       4D       A MSG9   FCC    /MODE= /
00877A F5FC       04       A        FCB    4
00878                          *
00879                          *  CLEAR-FILL SCREEN WITH CLEAR CHARACTER CLERCH
00880                          *  FOR BLACK LINES ON BUFF BACKGROUND USE INVERTED
00881                          *  LOGIC. A BIT SET IS BUFF A BIT CLEARED IS BLACK
00882                          *  SO CLERCH IS $FF.
00883                          *
00884A F5FD D6    FF       A CLEAR  LDB    CLERCH   GET CLEAR CHARACTER
00885A F5FF 8E    9000     A        LDX    #SCRN    BEGINNING OF SCREEN RAM
00886A F602 108E A800      A        LDY    #SCRN+RAMLEN END OF RAM
00887A F606 8D    01   F609          BSR    FILL1    FILL SCREEN
00888A F608 39                       RTS             BACK TO COM HANDLER
00889                          *
00890                          *  FILL1-COPY CHARACTER IN B FROM LOCATION IN X
00891                          *  TO LOCATION IN Y
00892                          *  INIT:  X=FROM
00893                          *         Y=UP TO
00894                          *         B=CHAR
00895                          *  VOLATILE: X
00896A F609 34    20       A FILL1  PSHS   Y        SAVE TO LOCATION
00897A F60B AC    E4       A FILL2  CMPX   ,S       COMPARE TO END
00898A F60D 24    04   F613          BHS    FILL3    NOT FINISHED
00899A F60F E7    80       A        STB    ,X+      STORE CHAR
00900A F611 20    F8   F60B          BRA    FILL2    BACK FOR MORE
00901A F613 35    A0       A FILL3  PULS   PC,Y     RETURN
00902                          *
00903                          *
00904                          *  SUBROUTINE PNTADD
00905                          *  GET ADDRESS IN SCREEN MEMORY
00906                          *  OF ANY POINT IN THE X-Y
00907                          *  SCREEN SPACE
00908                          *  INIT:  A=XPT
00909                          *         B=YPT
00910                          *  ON EXIT:  IX=ADDRESS  A=MASK
00911                          *  VOLATILITY:  A,B,X,Y
00912                          *
00913A F615       80       A BITAB  FCB    %10000000
00914A F616       40       A        FCB    %01000000
00915A F617       20       A        FCB    %00100000
00916A F618       10       A        FCB    %00010000
00917A F619       08       A        FCB    %00001000
00918A F61A       04       A        FCB    %00000100
00919A F61B       02       A        FCB    %00000010
00920A F61C       01       A        FCB    %00000001
00921                          *
00922A F61D 9B    80       A PNTADD ADDA   XOFF     OFFSET
00923A F61F 34    02       A        PSHS   A        SAVE A
00924A F621 44                       LSRA            XPT:=XPT/8
00925A F622 44                       LSRA
00926A F623 44                       LSRA
```

Figure 7. Interactive Graphics System Program Listing (Sheet 16 of 20)

```
PAGE  Ø17  GRAFPAKA.SA:1  GRAFPA

ØØ927A F624 34    Ø2      A        PSHS   A        SAVE
ØØ928A F626 DB    6Ø      A        ADDB   YOFF     OFFSET B
ØØ929A F628 86    2Ø      A        LDA    #32
ØØ930A F62A 3D                    MUL             YPT:=32*YPT
ØØ931A F62B 4Ø                    NEGA
ØØ932A F62C 5Ø                    NEGB            YPT:=-32*YPT
ØØ933A F62D 82    ØØ      A        SBCA   #Ø       PROP BORROW
ØØ934A F62F EB    E4      A        ADDB   ,S       =XPT/8-32*YPT
ØØ935A F631 89    ØØ      A        ADCA   #Ø       PROP CARRY
ØØ936A F633 C3    A7EØ    A        ADDD   #$17EØ+SCRN =XPT/8-32*YPT+$17EØ
ØØ937A F636 1F    Ø1      A        TFR    D,X      ADDRESS IS NOW IN IX
ØØ938                        *  NOW GET BIT MASK AND LEAVE IN ACCA
ØØ939A F638 A6    61      A        LDA    1,S      GET XPT
ØØ940A F63A 84    Ø7      A        ANDA   #%ØØØØØ111 SAVE 3 LSBITS
ØØ941A F63C 1F    89      A        TFR    A,B      SAVE SHIFT CNT
ØØ942A F63E 31    8C D4            LEAY   BITAB,PCR GET TABLE ADDR
ØØ943A F641 A6    A6      A        LDA    A,Y      GET PIXEL MASK
ØØ944A F643 35    AØ      A        PULS   PC,Y     RTS, CLEAN STACK
ØØ945                        *
ØØ946                        *  SUBROUTINE LINGEN.
ØØ947                        *  GIVEN TWO POINTS (X1,Y1) AND (X2,Y2) THIS ROUTINE
ØØ948                        *  CALCULATES AND PLOTS A SERIES OF PIXELS
ØØ949                        *  REPRESENTING A LINE BETWEEN THE TWO POINTS.
ØØ95Ø                        *  THE FORMAT IS FOR A DISPLAY 32 BYTES BY 8 BITS
ØØ951                        *  HORIZONTAL AND 192 LINES VERTICAL
ØØ952                        *  A MODIFIED BRESENHAM'S ALGORITHM
ØØ953                        *  IS USED IN WHICH AN ERROR TERM
ØØ954                        *  IS CALCULATED FOR THE EXACT LOCATION OF THE
ØØ955                        *  LINE AND THE ACTUAL POINT PLOTTED AND EITHER
ØØ956                        *  X OR Y ARE INCREMENTED (OR DECREMENTED) DEPENDING
ØØ957                        *  ON THE SIGN OF THIS TERM.
ØØ958                        *
ØØ959                        *  X1 AND X2 MUST BE -128<X<127
ØØ96Ø                        *  Y1 AND Y2 MUST BE -96<Y<95
ØØ961                        *
ØØ962                        *  IF FLAG=Ø, LINE WILL BE DRAWN
ØØ963                        *  IF FLAG=$FF, LINE WILL BE ERASED
ØØ964                        *
ØØ965                        *  REGISTERS MODIFIED:  A,B,X,Y
ØØ966                        *
ØØ967                        *  FIRST THE ADDRESS AND POINT LOCATION
ØØ968                        *  WITHIN THE BYTE OF THE FIRST X,Y
ØØ969                        *  PAIR IS CALCULATED.  THE ADDRESS
ØØ97Ø                        *  OF THE CURRENT BYTE TO BE PLOTTED
ØØ971                        *  IS ALWAYS IN THE X REGISTER
ØØ972                        *  THE CURRENT PIXEL LOCATION IS IN
ØØ973                        *  "MASK".  TO PLOT A POINT,
ØØ974                        *  THIS WORD IS "ORED" OR "ANDED" WITH
ØØ975                        *  THE INFORMATION ALREADY ON THE SCREEN
ØØ976                        *  DEPENDING ON THE WRITE-UNWRITE FLAG.
ØØ977                        *
ØØ978                        *  CALCULATE ADDRESS AND MASK
ØØ979                        *  OF BEGIN AND END POINTS
ØØ98Ø                        *  FOR LATER COMPARISON
ØØ981                        *
ØØ982                        *  INIT:  X=X2:Y2   Y=X1:Y1  A=FLAG
ØØ983                        *
ØØ984A F645 BF    DDØ5    A LINGEN STX    X2
```

Figure 7. Interactive Graphics System Program Listing (Sheet 17 of 20)

```
PAGE  018  GRAFPAKA.SA:1  GRAFPA

00985A F648 10BF DD03     A          STY    X1        SAVE X1,Y1
00986A F64C B7   DD00     A          STA    FLAG      SAVE WRITE-UNWRITE FLAG
00987A F64F FC   DD05     A          LDD    X2        SETUP FOR PNTADD
00988A F652 BD   F61D     A          JSR    PNTADD    GET END ADDR
00989A F655 BF   DD11     A          STX    ENDADD    SAVE
00990A F658 B7   DD0A     A          STA    EMASK
00991A F65B FC   DD03     A          LDD    X1
00992A F65E BD   F61D     A          JSR    PNTADD    GET ADDR AND MASK
00993A F661 B7   DD09     A          STA    MASK      SAVE MASK
00994                               *
00995                               *    THE X REGISTER NOW HAS THE ADDRESS IN DISPLAY
00996                               *    MEMORY OF THE BYTE IN WHICH THE FIRST X,Y
00997                               *    POINT LIES.  "MASK" NOW HAS AN 8-BIT BYTE
00998                               *    WITH ONLY ONE BIT SET IN THE LOCATION OF THAT
00999                               *    POINT.
01000                               *
01001                               *    NOW SET UP TO GENERATE POINT SEQUENCE. START
01002                               *    WITH POSITIVE INCREMENTS.
01003                               *
01004A F664 86   E0       A          LDA    #-32      LINE INCREMENT FOR VDG RAM
01005A F666 B7   DD08     A          STA    YINC      IS 32 IN Y DIRECTION
01006A F669 86   01       A          LDA    #1        BIT INCREMENT IN X DIRECTION
01007A F66B B7   DD07     A          STA    XINC      IS 1
01008A F66E F6   DD03     A          LDB    X1        GET X1
01009A F671 1D                      SEX               SIGN EXTEND TO DO DOUBLE
01010A F672 FD   DD01     A          STD    TEMP      BYTE ARITHMETIC,SAVE
01011A F675 F6   DD05     A          LDB    X2        GET X2 AND SIGN EXTEND
01012A F678 1D                      SEX               TO CALCULATE DOUBLE
01013A F679 B3   DD01     A          SUBD   TEMP      SUB FOR DELTAX
01014A F67C FD   DD0B     A          STD    DELTAX    SAVE DELTAX
01015A F67F F6   DD04     A          LDB    Y1        GET Y1 AND DO
01016A F682 1D                      SEX               THE SAME THING
01017A F683 FD   DD01     A          STD    TEMP
01018A F686 F6   DD06     A          LDB    Y2
01019A F689 1D                      SEX               CALCULATE DOUBLE PRECISION
01020A F68A B3   DD01     A          SUBD   TEMP      DELTAY
01021A F68D FD   DD0D     A          STD    DELTAY    AND SAVE
01022A F690 7F   DD0F     A          CLR    E         SET ERROR TERM TO ZERO
01023A F693 7F   DD10     A          CLR    E+1
01024A F696 7D   DD0B     A          TST    DELTAX    CHECK FOR NEGATIVE DELTA
01025A F699 2A   0D   F6A8          BPL    NOTNEG    IF NOT NEGATIVE
01026A F69B FC   DD0B     A          LDD    DELTAX    GET ABSOLUTE VALUE OF
01027A F69E 40                      NEGA              DELTAX
01028A F69F 50                      NEGB
01029A F6A0 82   00       A          SBCA   #0
01030A F6A2 FD   DD0B     A          STD    DELTAX    SAVE
01031A F6A5 70   DD07     A          NEG    XINC      BUT NEGATE INCREMENT
01032A F6A8 7D   DD0C     A NOTNEG   TST    DELTAX+1  CHECK FOR ZERO DELTA
01033A F6AB 26   06   F6B3          BNE    NOTZER
01034A F6AD 73   DD0F     A          COM    E         IF SO SET E=-1
01035A F6B0 73   DD10     A          COM    E+1
01036A F6B3 7D   DD0D     A NOTZER   TST    DELTAY
01037A F6B6 2A   0D   F6C5          BPL    POSY      DELTAY<0?
01038A F6B8 FC   DD0D     A          LDD    DELTAY    DELTAY:=ABS(DELTAY)
01039A F6BB 40                      NEGA
01040A F6BC 50                      NEGB
01041A F6BD 82   00       A          SBCA   #0
01042A F6BF FD   DD0D     A          STD    DELTAY
```

Figure 7. Interactive Graphics System Program Listing (Sheet 18 of 20)

```
PAGE  019  GRAFPAKA.SA:1  GRAFPA

01043A F6C2 70   DD08     A        NEG   YINC     NEGATE Y INCREMENT
01044A F6C5 B6   DD09     A POSY   LDA   MASK     GET BYTE WITH PIXEL
01045A F6C8 20   09   F6D3        BRA   PLOTER
01046A F6CA FC   DD0F     A XE     LDD   E        E:=E+DELTAY
01047A F6CD B3   DD0D     A        SUBD  DELTAY
01048A F6D0 FD   DD0F     A        STD   E
01049A F6D3 B6   DD09     A PLOTER LDA   MASK     GET BIT MASK
01050A F6D6 7D   DD00     A        TST   FLAG     WRITE OR UNWRITE?
01051A F6D9 2A   04   F6DF        BPL   WRITE
01052A F6DB AA   84       A        ORA   ,X       GET RID OF DOT
01053A F6DD 20   03   F6E2        BRA   PDOT     REPLACE ON SCREEN
01054A F6DF 43                WRITE  COMA     INVERT PIXEL
01055A F6E0 A4   84       A        ANDA  ,X       COMBINE WITH PREVIOUS PIXELS
01056A F6E2 A7   84       A PDOT   STA   ,X       REPLACE ON SCREEN
01057A F6E4 BC   DD11     A        CMPX  ENDADD   FINISHED?
01058A F6E7 27   41   F72A        BEQ   FIN1     YES
01059A F6E9 7D   DD0F     A NOTFIN TST   E        IS ERROR NEGATIVE?
01060A F6EC 2A   10   F6FE        BPL   UPX      IF NOT INCREMENT X DIR
01061A F6EE FC   DD0F     A        LDD   E        IF SO E:=E+DELATAX
01062A F6F1 F3   DD0B     A        ADDD  DELTAX
01063A F6F4 FD   DD0F     A        STD   E
01064A F6F7 B6   DD08     A        LDA   YINC     GET Y INCREMENT
01065A F6FA 30   86       A        LEAX  A,X      ADD TO POINTER REGISTER
01066A F6FC 20   D5   F6D3        BRA   PLOTER   MORE DOTS TO GO
01067A F6FE B6   DD09     A UPX    LDA   MASK     GET BIT MASK
01068A F701 7D   DD07     A        TST   XINC     LEFT OR RIGHT
01069A F704 2A   12   F718        BPL   SHIFTR   POSITVE,SHIFT RIGHT
01070A F706 1C   00       A SHIFTL ANDCC #0       C=0
01071A F708 49                    ROLA           XINC IS NEG SO SHIFT LEFT
01072A F709 27   05   F710        BEQ   DECAD    FINISHED THIS BYTE?
01073A F70B B7   DD09     A        STA   MASK     NO,JUMP BACK TO LOOP
01074A F70E 20   BA   F6CA        BRA   XE
01075A F710 49                DECAD  ROLA           GET BIT BACK IN MASK
01076A F711 B7   DD09     A        STA   MASK     SET UP WITH 1 IN LSBIT AGAIN
01077A F714 30   1F       A        LEAX  -1,X     DECREMENT X
01078A F716 20   B2   F6CA        BRA   XE       BACK IN LOOP
01079A F718 1C   00       A SHIFTR ANDCC #0
01080A F71A 46                    RORA
01081A F71B 27   05   F722        BEQ   INCAD    FINISHED MASK?
01082A F71D B7   DD09     A        STA   MASK     NO
01083A F720 20   A8   F6CA        BRA   XE
01084A F722 46                INCAD  RORA           YES, INC ADDR POINTER
01085A F723 B7   DD09     A        STA   MASK     SET UP WITH 1 IN MSBIT
01086A F726 30   01       A        LEAX  1,X      UP ADDR PTR
01087A F728 20   A0   F6CA        BRA   XE       LOOP AGAIN
01088A F72A B6   DD09     A FIN1   LDA   MASK
01089A F72D B1   DD0A     A        CMPA  EMASK    HIT LAST DOT?
01090A F730 26   B7   F6E9        BNE   NOTFIN   HIT END YET?
01091A F732 39                    RTS            YES, EXIT
01092                          *
01093                          *  DRAW-PRINTS A PROMPTING MESSAGE ASKING FOR X-Y
01094                          *  COORDINATES BETWEEN WHICH TO DRAW A LINE.
01095                          *  COORDINATES MUST BE IN 2'S COMPLEMENT HEX
01096                          *
01097                          *
01098A F733 BD   F003     A DRAW   JSR   CRLF     SEND CRLF
01099A F736 8E   F5D6     A        LDX   #MSG5    "ENTER TWO POINTS"
01100A F739 BD   F04E     A        JSR   PDATA    "X1,Y1"
```

Figure 7. Interactive Graphics System Program Listing (Sheet 19 of 20)

```
PAGE  020  GRAFPAKA.SA:1  GRAFPA

01101A F73C BD   F033    A          JSR    GETEXP   GET X1
01102A F73F 26   37    F778         BNE    GTRET    IF NOT HEX REUTRN
01103A F741 F7   DD03    A          STB    X1       SAVE X1
01104A F744 BD   F000    A          JSR    OUTS     PRINT SPACE
01105A F747 BD   F033    A          JSR    GETEXP   GET Y1
01106A F74A 26   2C    F778         BNE    GTRET    NOT HEX
01107A F74C F7   DD04    A          STB    Y1       SAVE Y1
01108A F74F 8E   F5EF    A          LDX    #MSG7    "X2,Y2"
01109A F752 BD   F04B    A          JSR    PDATA1   CRLF THEN STRING
01110A F755 BD   F033    A          JSR    GETEXP   X2
01111A F758 26   1E    F778         BNE    GTRET    NOT HEX
01112A F75A F7   DD05    A          STB    X2       SAVE X2
01113A F75D BD   F000    A          JSR    OUTS
01114A F760 BD   F033    A          JSR    GETEXP   Y2
01115A F763 26   13    F778         BNE    GTRET    NOT HEX
01116A F765 F7   DD06    A          STB    Y2       SAVE Y2
01117A F768 BD   F003    A          JSR    CRLF     NEXT LINE
01118A F76B 10BE DD03    A          LDY    X1       Y=X1Y1 FOR LINGEN
01119A F76F BE   DD05    A          LDX    X2       X=X2Y2 FOR LINGEN
01120A F772 B6   DD00    A          LDA    FLAG     A=0 FOR WRITE,$FF FOR UNWRITE
01121A F775 BD   F645    A          JSR    LINGEN   DRAW LINE
01122A F778 39              GTRET   RTS              RTS TO COMMAND HANDLER
01123                       END
```

Figure 7. Interactive Graphics System Program Listing (Sheet 20 of 20)

MOTOROLA *Semiconductor Products Inc.*

3501 ED BLUESTEIN BLVD., AUSTIN, TEXAS 78721 • A SUBSIDIARY OF MOTOROLA INC.

A14556 PRINTED IN USA 7-81 IMPERIAL LITHO 10,000

AN-825