AN IMAGE PROCESSING SYSTEM FOR ANALYZING FLUID SHEAR STRESSED ENDOTHELIUM

by

William Ivan Ogilvie

SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS OF THE

DEGREE OF

BACHELOR OF SCIENCE

AT THE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

MAY 1981

Signature of Author.....
Department of Electrical Engineering

Certified.....
C. Forbes Dewey, Jr.
Thesis Supervisor

Accepted by.....
Professor David Adler, Chairman,
Departmental Committee on Theses

AN IMAGE PROCESSING SYSTEM FOR ANALYZING FLUID SHEAR STRESSED ENDOTHELIUM

by

WILLIAM IVAN OGILVIE

Submitted to the Department of Electrical Engineering and Computer Science on May 22, 1981 in partial fulfillment of the requirements for the degree of Bachelor of Science.

ABSTRACT

Vascular endothelium, when it is subjected to fluid shear stress, adopts an elongated shape in the direction of fluid flow. The morphological response of vascular endothelium, to shear stress, is being studied to better understand the role that shear stress has in arterial diseases.

This thesis describes a computer system for analysing these effects. A high resolution camera is used to store an image of endothelium into computer memory, where it can be analyzed directly.

Thesis Supervisor: C. Forbes Dewey, Jr.

Title: Professor

ACKNOWLEDGEMENTS

To all the people who have helped me with this project, I express my gratitude. To my advisor, Professor C. Forbes Dewey, I am most indebted. His enthusiasm and confidence in my capabilities could not have been more motivating. My appreciation also goes to Steve Bussolari, Ken Kalamuck, and Judy Steciak, for their friendly encouragement.

To Professor Berthold K.P. Horne, and other faculty members of the Department of Electrical Engineering and Computer Science, I offer my humble thanks for the knowledge they have imparted to me. And to my parents and family, whose patience, support, and guidance have helped me grow with each step, I am most grateful.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

Vascular endothelial cells line the interior of blood vessels, where they are subjected to shear stress from the flow of blood. The magnitude of this shear stress is known to vary widely from one region of an artery to another. Since arterial diseases appear to be most prevalent in regions of high shear stress, some researchers have developed hypotheses that propose a causal relationship between fluid shear stress and the development of arterial diseases.

A consistently observed effect of fluid shear stress on vascular endothelium is the alignment of cells with the direction of blood flow. When the flow changes, the cells re-orient themselves to the new direction. A photomicrograph of cells that have not been subjected to fluid shear stress is shown in Fig. 1.

This characteristic of vascular endothelium, when it is subjected to fluid shear stress, is the subject of a

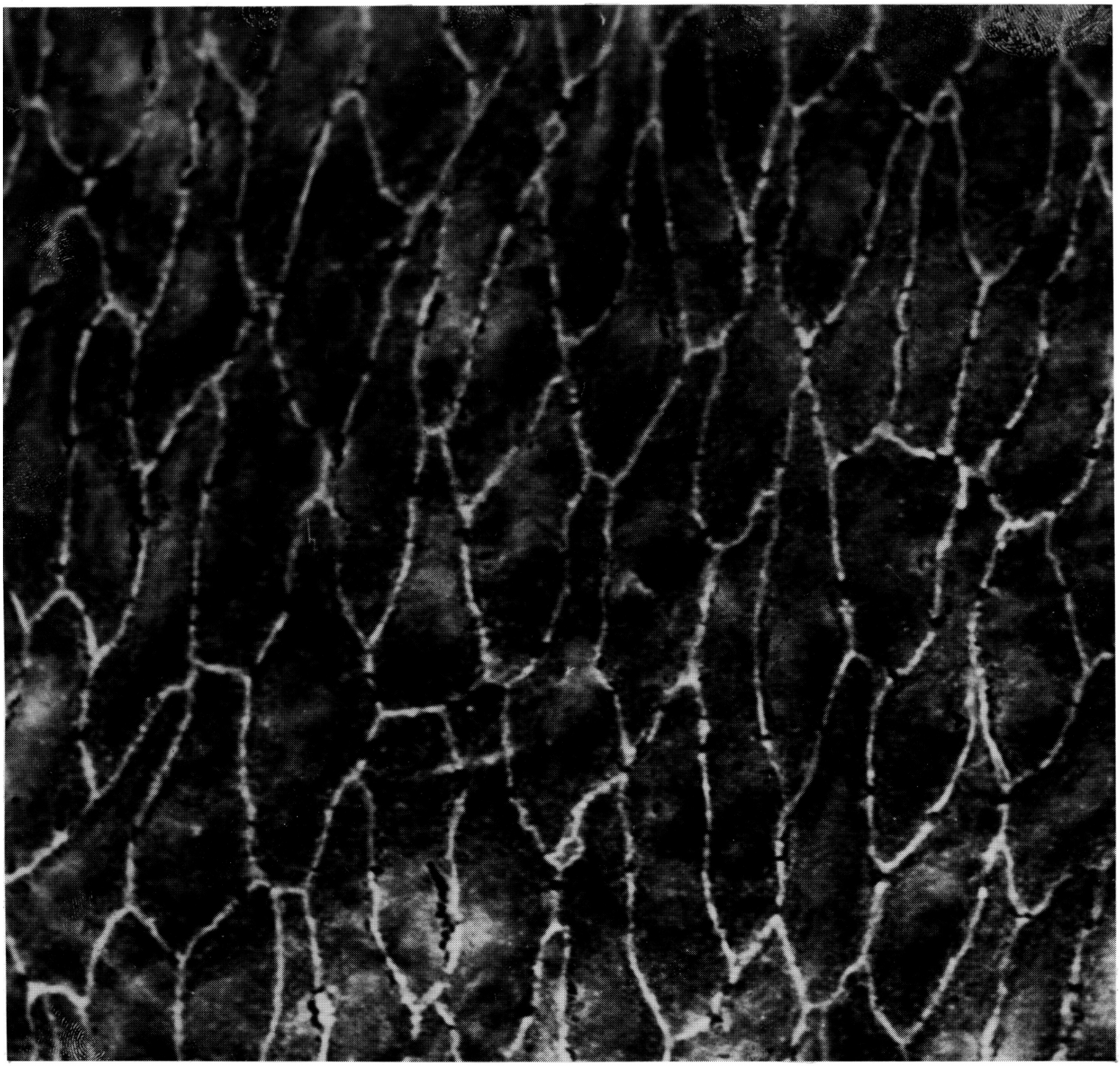


Figure 1: Vascular Endothelium

collaborative study by the Fluid Mechanics Laboratory of the Massachusetts Institute of Technology, and the Vascular Pathophysiology Laboratory of the Peter Bent Brigham Hospital. The purpose of this research is to quantify the response of vascular endothelium to shear stress. This is done by analyzing the shapes of cells that have been subjected to a known, controlled shear stress.

An apparatus for producing controlled shear stress on *in vitro* tissue was developed by Bussolari (1). As is indicated in Fig. 2, tissue cultures are placed in the lower plate, and the two plates are brought together, until the fluid in the lower plate completely wets the upper plate. A variable speed motor is magnetically coupled to the upper plate; which in turn drives the fluid. Shear stresses over the full range that are observed *in vivo* can be produced with this apparatus on *in vitro* specimens. Figure 3 shows the effect of 48 Hrs. of shear stress on a sample.

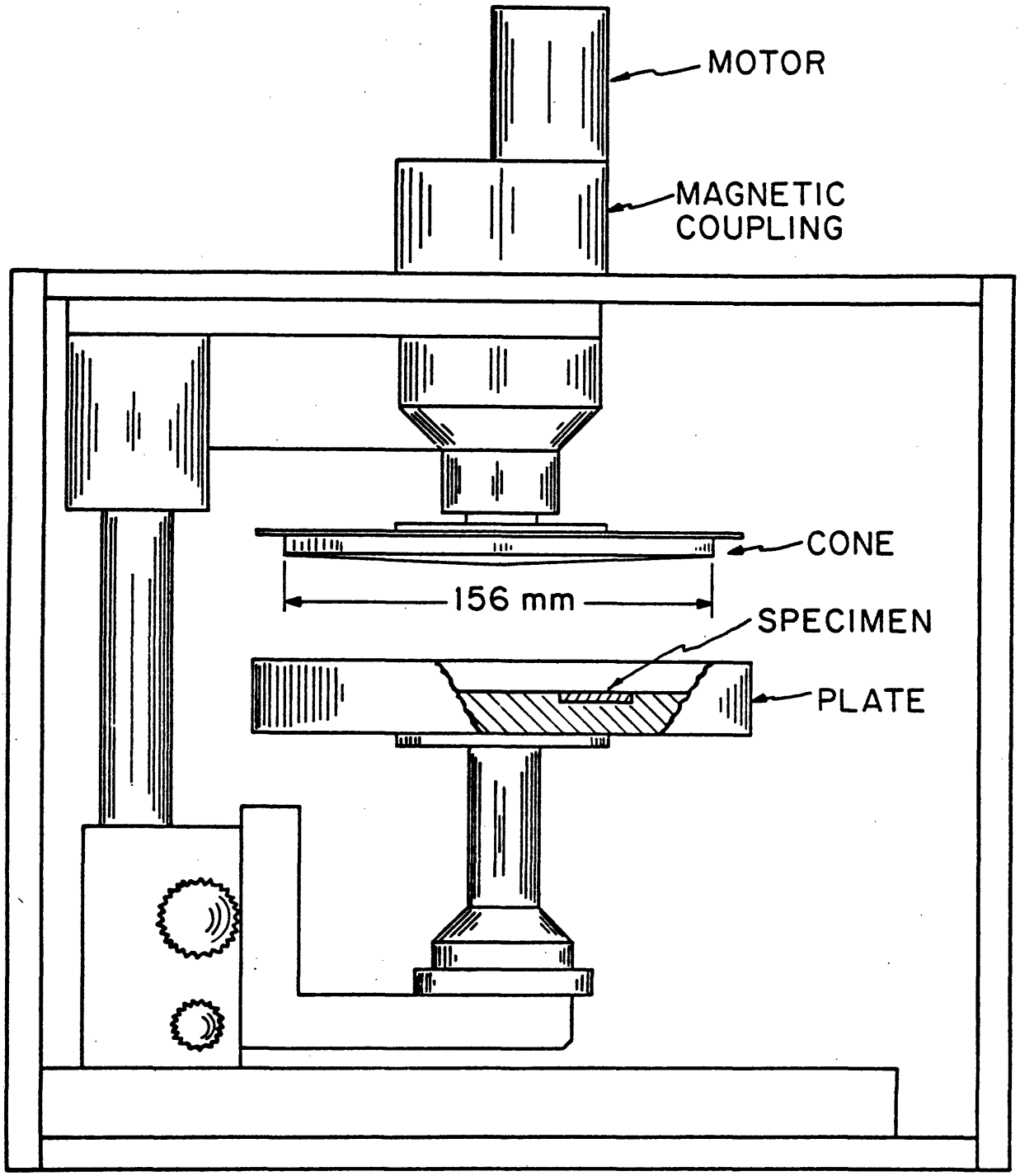


Figure 2: Line drawing of shear stress apparatus. For testing, the plate is raised until contact is made with the apex of the rotating cone.

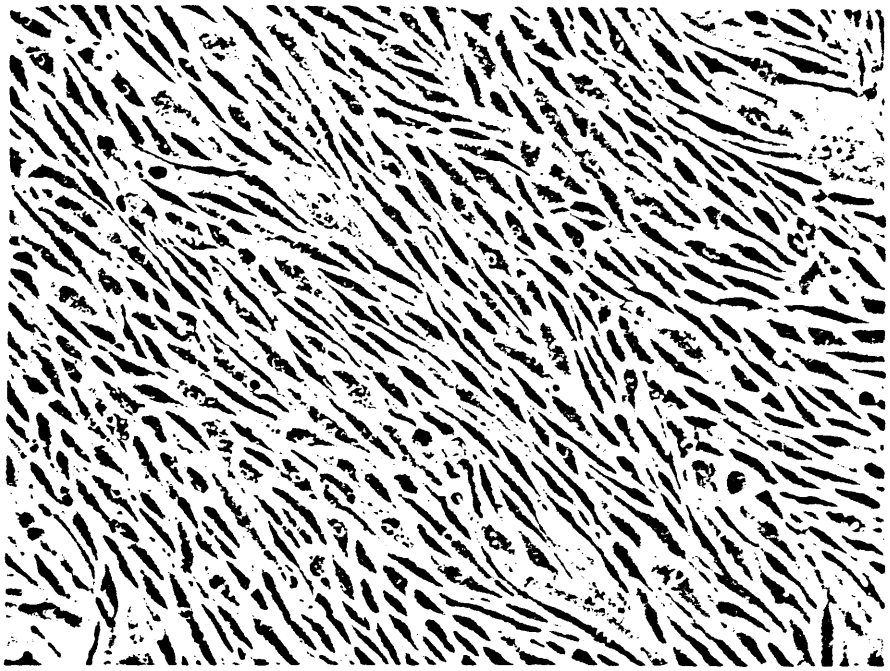


Fig. 1a: Photomicrograph of incubator control endothelial specimen at 48h (mag. 100X).

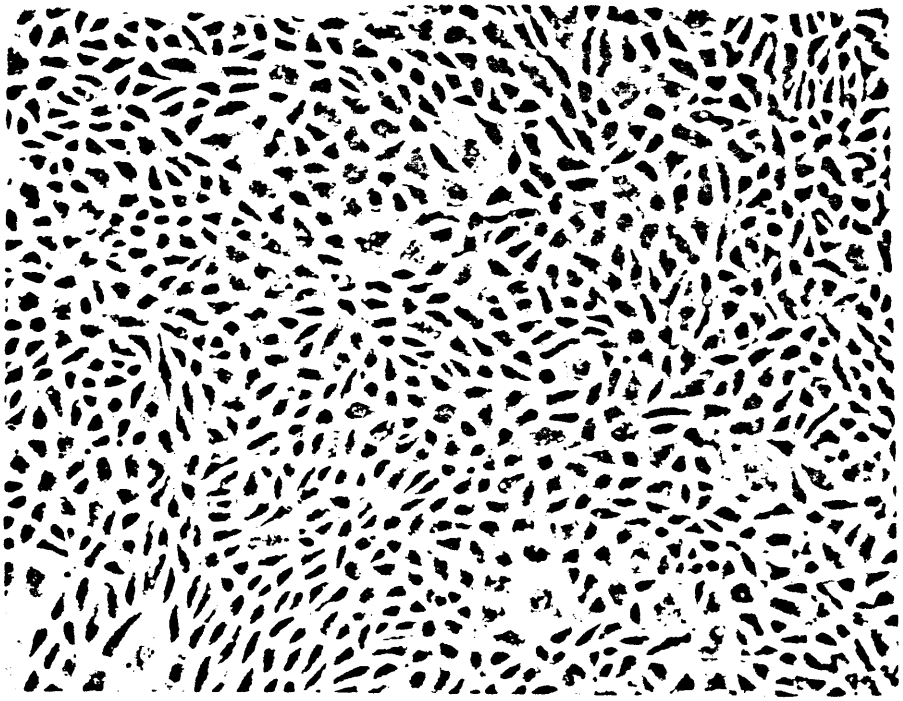


Fig 1b: Photomicrograph of endothelial specimen exposed to 8 dynes/cm^2 for 48h (mag. 100X)

The method used to analyze the effects of fluid shear stress was to fit the call shapes to an ellipse. York (2) wrote a computer program to do this calculation on a set of data points that describe a cell's shape. The program was written in Basic, and ran on a Tektronix 4051 computer. A plotter pen was used to trace out the outline of a cell, with a photomicrograph as a guide. At regular intervals, the pen's position could be transmitted to the computer.

Qualitative measurements can be done on vascular endothelium to investigate the effects of fluid shear stress, by applying York's method to cells that have been subjected to controlled shear stress in the apparatus designed by Bussolari. Small numbers of cells can be analyzed from photomicrographs, but the procedure is not suitable for collecting a large data base. A faster, and more direct way of acquiring position values to describe the cell's shape is needed.

This thesis describes an alternative method of acquiring the data points. A video camera is used to load an image of the cells into a computer's memory. Image processing programs analyze each cell, and return sets of data points to the ellipse-fit program. This method is fast, and it frees the operator from having to enter each data point.

DATA ACQUISITION FROM IMAGES

The cell shape analysis program described in York's thesis (2) was adapted to run on the Apple II computer that was dedicated to this project. The Apple is a versatile unit that, in size and appearance, resembles a portable typewriter. Like a typewriter, it has a built-in keyboard that is arranged in the usual fashion. However, it doesn't print, and requires a video monitor for listing programs or displaying graphics. Programs can be written by entering statements with the keyboard. Basic programs, such as the cell shape analysis program, are interpreted by a Basic interpreter program that is stored in read only memory (ROM). Two floppy disk drives, for permanent storage of programs and data, a printer, and a graphics tablet are also connected to the Apple.

Cell shapes can be input to the program using a graphics tablet. By tracing out a cell outline on

this tablet with a stylus, the operator can enter a set of coordinates about the perimeter of the cell. The program uses these coordinates to do an ellipse fit to the cell. For a large number of cells, this process is time-consuming. A more automated and controllable method of data input to the best-fit ellipse program is needed.

Each photomicrograph of endothelium can contain several cells (e.g. Fig. 1). With the use of a video camera, the complete image can be stored directly in memory. The shape of each cell can be determined once an image of sufficient resolution is available in memory. If this method is largely automatic, the data acquisition process will be faster and more controlled.

Once an image is in memory, it can be modified and analyzed by programs the operator selects. Extraneous artifacts, which would be noise to a shape detection program, can be removed and desireable features enhanced.

In this regard, there are two ways of performing operations to a stored image. The operator can, interactively, do a series of small changes to the image, while observing the effect on a graphics display, or he can run a program that requires no operator input. These two classes of programs are, respectively, interactive and task programs.

Many of the operations that have to be done on an image are quite involved, and would be very time consuming to do in Basic. Thus it is necessary to write these routines in the base level language of the computer. The speed advantage that comes from writing these programs in assembly language comes at a cost in complexity and difficulty of programming. Despite these apparent difficulties, an assembly language program once written, can be interfaced to a Basic program quite easily. This is important, because Basic supports disk operations, is a versatile supervisor program language, and is the environment of the cell analysis program.

An assembly language program can be linked to Basic with a CALL statement. This statement results in a jump to the memory location to which the operand of the instruction points. All information necessary for the continuation of the Basic program is stored, and the computer begins executing code at the call routine entry point. From this assembly code program, a subroutine return instruction (RTS) will transfer control back to the Basic program. Because the assembly language program can modify any area of memory, it can return data by over-writing a Basic program data area.

This method has the advantage that two operating environments are available to the user. In the Basic program, statistical, and algebraic operations can be done on data that is extracted from images in the graphics environment. The assembly language image processing routines are selectable from Basic, and always return the user to it.

HARDWARE COMPONENTS OF THE SYSTEM

The Apple II is a micro-processor based system with several built in features. It has circuitry for displaying text or graphics on a video monitor, and comes with an integral typewriter-like keyboard. A joystick, which can be used to move a cursor around on the screen, plugs into the mother board inside the case. Each of these devices requires a minimal amount of support software. For most applications, a monitor software package, written in ROM, provides the necessary support. This is particularly true for the keyboard input and text displays on the video monitor. However, high-level graphics functions are not supported.

Four memory areas are referred to as screen pages. At any time, one of these areas is used as a storage area for the video display. The location, and display characteristics of these screen pages are described in table 1.

Figure 4: Apple II graphics screen tables.

Table 1: Video Display Memory Ranges

Screen	Page	Begins at:		Ends at:	
		Hex	Decimal		
Text/Lo-Res	Primary	$400	1024	$7FF	2047
	Secondary	$800	2048	$BFF	3071
Hi-Res	Primary	$2000	8192	$3FFF	16383
	Secondary	$4000	16384	$5FFF	24575

Table 2: Screen Soft Switches

Location:			
Hex	Decimal		Description:
$C050	49232	-16304	Display a GRAPHICS mode
$C051	49233	-16303	Display a TEXT Mode
$C052	49234	-16302	Display all TEXT or GRAPHICS
$C053	49235	-16301	Mix TEXT and a GRAPHICS mode
$C054	49236	-16300	Display the Primary page (Page 1)
$C056	49238	-16298	Display LO-RES GRAPHICS mode
$C057	49239	-16297	Display HI-RES GRAPHICS mode

A program can switch between display modes by addressing certain memory locations. These addresses are referred to as soft switches. the memory locations, and the graphics modes that they enable are shown in table 2.

Other devices can be incorporated into the system by plugging an interface printed circuit board into the I/O slots on the motherboard of the Apple. A single interface board powers two cable-connected floppy disk drives, and another board drives a printer.

The interface boards each have a ROM that contains the coding for the device drivers. This memory area is enabled, and is directly executed by the Apple's microprocessor when the peripherial device is operated.

The Apple II uses a 6502 microprocessor chip, which is manufactured by Advanced Micro Devices. The device has an 8 bit wide data bus, but can address 64 Kbytes of memory.

There is one accumulator and two index registers. The first 256 locations are referred to as the base-page. these locations are used as indirect pointers to other locations. The X register is used to index through a base-page pointer table, while the Y register can be used to index through the effective address area. Each addressing mode has its uses, and since there are only a few base-page locations left for user programs, careful choices have to be made.

THE CAMERA AND ITS INTERFACE

The Hamamatsu camera is designed to be operated under the control of a computer. The unit consists of a compact camera, connected by a thick cable to a controller chassis. It is interfaced to the computer with an interface board that plugs into a slot in the controller. This board is connected with a cable to an interface board in the Apple II.

Its best performance is in subdued light, and it can be damaged if it is aimed at bright sunlight. An aperture adjustment on the lens, and a red warning LED are the only protection against this. Photo flashes can also damage the vidicon, and should not be used anywhere near it. When the camera is not in use, the lens assembly should be unscrewed, and the flat aluminium cap screwed in its place; to protect the vidicon from high light levels and flying debri.

The camera's image space consists of a 1024 X 256 frame of 8 bit pixels. With interlacing, the vertical

resolution can be increased to 512 or 1024 lines. When this feature is selected, subsequent frames are shifted vertically, and 1 or 3 extra horizontal lines appear between the original, non-interlaced ones. Although this is beyond the resolution capabilities of most video monitors, an image processing computer program can make full use of it.

The camera scans a complete frame in 1/60 Sec., but the analog to digital convertor (A/D) is not fast enough to digitize each sequential pixel in the video signal. The video signal has a bandwidth of 15 MHz., while tne A/D has a maximum conversion rate of 40 KHz. A norizontal scan line intersects with a vertical sample line at each point that a conversion is done. The controller can be programmed to start a conversion at any X address.

Successive vertical conversions in either direction can occur automatically, or by explicit computer commands to the controller. It responds to set-up commands that can be used to put it in a auto-decrement or auto-increment mode for X addressing

As each pixel in a vertical line is converted, the byte is stored in a buffer, or I/O memory area. This array of intensity bytes can be transmitted to the host computer after the entire line has been converted. When the I/O transfer takes place, the X address is automatically incremented or decremented, if eitner auto-increment or auto-decrement were selected by set-up commands. Otherwise, the next sequence of conversions are done on the same vertical line.

When interlacing is selected, two or four I/O transfers take place before the line X address is changed. Thus, every second or fourth pixel intensity value is transmitted in each array, and two or four arrays have to be merged to obtain a correctly ordered line.

The Hamamatsu camera interfaces to the Apple II with an IEEE-488 interface board. This interfacing standard was established as a common interface protocol for computer equipment. (3) Several devices can be plugged in to the common GPIB bus and controlled individually by the computer. In one sense, it is like an extension of

the Apple I/O bus. Each device has a primary address, and may have several secondary addresses that are used for selecting a device, and a function of that device.

When the computer issues a system reset command, a common reset line goes active, and each device is put in an initial, known state. This Listener state, as it is referred to in the literature, (3,4) enables the devices to receive valid set-up commands when their primary addresses are sent over the bus. Other addresses, and invalid set-up commands are ignored. Each set-up command is preceded by a secondary address, and consists of a sequence of ASCII characters.

A set-up command can be sent to a device to make it a Talker. The device then transmits the contents of its buffer to the computer, and reverts back to the Listener state.

The advantage of this protocol is, that as much as is possible, all devices are handled in a consistent way. This frees the programmer from having to concern himself with hardware aspects of pripherial devices when writing a program.

The data transferred after each Talk command consists of sequential intensity values from the 256 point vertical line. The camera controller reloads its buffer during each frame, and can transmit it during the frame blanking interval. (.75 mS.) This assumes that the call routine, that is executed by the the code stored in ROM on the interface board, can retreive the data and store it in memory in that amount of time. Otherwise, the maximum throughput, as set by the vertical line conversion, would not be realizeable. However, without even considering the timing requirements of all the I/O code, it is clear that this throughput cannot be realized; because four cycles of the 1 MHz. processor clock are required to load each of the 256 bytes in memory. This is true for both interlaced, as well as non-interlaced modes; only the scaling factors change.

Each line transfer takes two frame periods. To transfer a complete, non-interlaced frame to the Apple takes 31 seconds. Not all of this information can be

used by the Apple II. The minimum image set of the camera is 256 Kbytes; but the Apple only has 16 Kbytes of graphics memory. Clearly some intelligent pruning of information has to take place. This would decrease the time required to load an image into memory.

SOFTWARE DESIGN ISSUES

There are several issues involved in the design of an image processing system. Trade-offs have to be made between image resolution and program space, because there is never enough memory to fulfill all the requirements of both. Image processing operations often consist of a few primative functions that are done to every pixel. To speed up iterative programs, programmers will resort to in-line coding techniques that expand loops into straight line sections of code; with no backwards jumps. This approach can greatly increase the memory requirements of a program, and should be used sparingly when memory is at a premium. A better method is to use fast algorithms when at all possible

IMAGE STORAGE CONSIDERATIONS

An issue of primary importance, at the outset of this project, was how much memory to allocate for memory storage. Any useful adaption of the camera to established Apple conventions requires that part of the camera's image space be mapped to the Apple's graphics area. This is essentially the only free area of memory that is large and contiguous, and the image should be at least partly visible to tne operator. Two bits of pixel gray scale is adequate for the microscope images, and there are, conveniently, two identical high resolution graphics screen pages.

Set-up instructions to the camera allow the user to start a series of vertical scans at any X-coordinate position. The software routine which concatenates each intensity value, and stores a two bit value in the two screen pages, will use whatever block of 192 bytes is selected. Effectively, this reduces the

I/O transfer by 1/4. This means that it takes 8 Sec. to load an image in the Apple's memory. By convention, the primary page pixels have a weight of 2, and the secondary page pixels a weight of 1.

This method guarantees that any part of the camera's image space can be used to fill the 280 X 192 pixel Apple high resolution screen pages. The screens can only be displayed individually. However, it is not important for the operator to be able to see the gray scale image, because a common purpose of most image processing operations is to simplify a binary image. Much information is lost in the process, but the object is to isolate one feature of the image.

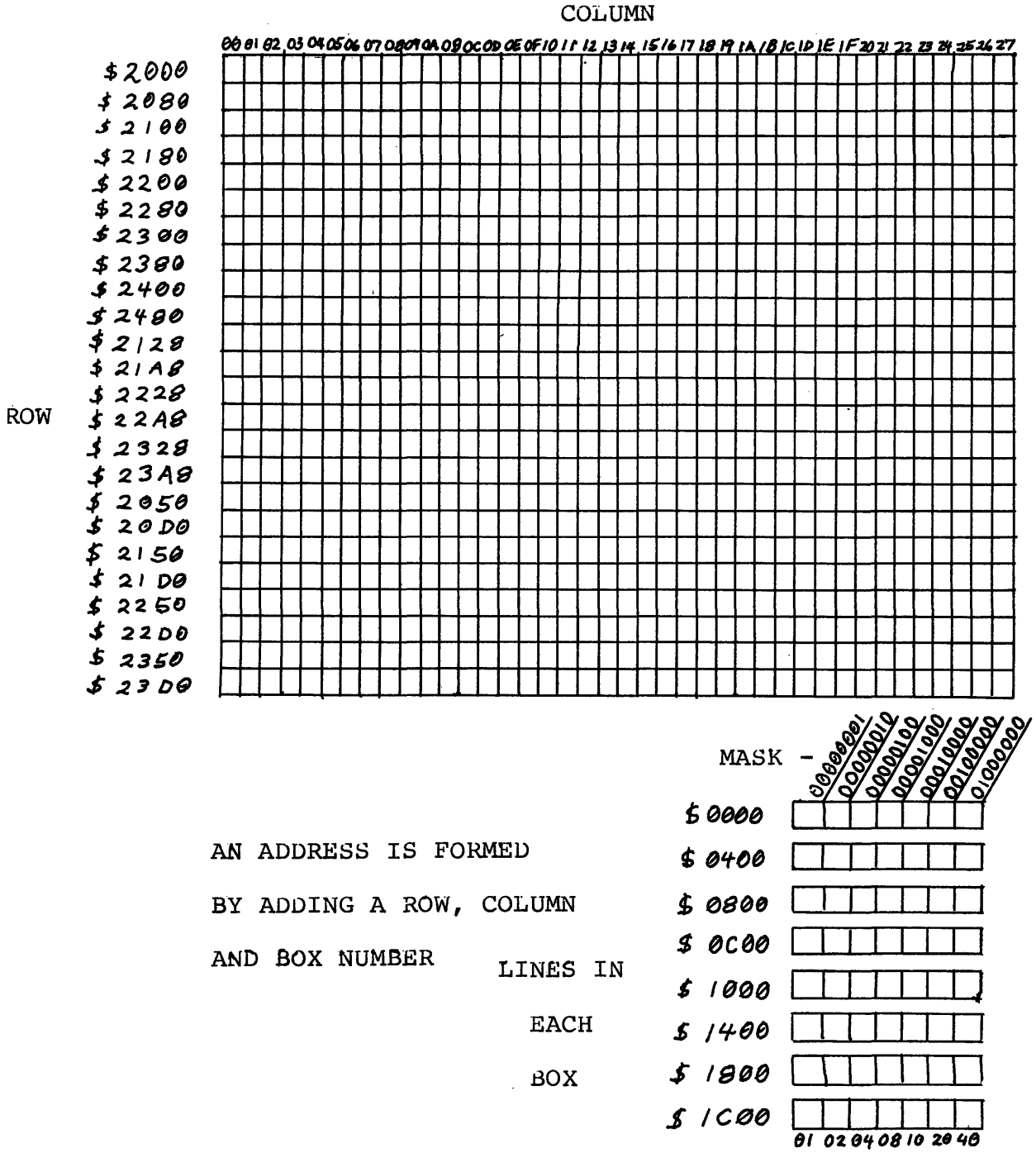
PIXEL ADDRESSING

As shown in Fig. 5, seven pixels are packed in each byte. Along a horizontal line, there is a quasi-ordering of pixels in memory. It takes three bytes to isolate any one pixel; two address bytes, and a mask byte.

Along a vertical line, there is no discernable order to the addresses. Because of this, address calculation involves the addition of several quantities, depending on what vertical section, box, or line the pixel is in. Even with table look-ups, as is used in the program, this is a slow calculation.

It is often possible to program entirely with fast address squencing. In this way, a pixel address is continually modified as the cursor single steps in the vertical or horizontal direction. This is a useful technique for single moves, but is unworkable for general step sizes. However, when single step address recalculation is used most of the time, screen operations run quickly.

Figure 5: Pixel addressing in the High resolution graphics screen of the Apple II.



Two routines, NEXTX and NEXTY modify the cursor address and mask bytes. If the accumulator is positive, when one of these routines is entered, then the cursor is moved one position to the right or one position down. A negative accumulator has the opposite result. The Y register returns 80 (hex) if the operation places the cursor at a screen edge, or tries to place the cursor past an edge. This convention is maintained throughout the cursor move routines.

There are two locations for cursor addresses in the base page, and two mask byte locations in the program data area. These bytes are usually addressed with the X register index instructions. The X register is 0 or 2, depending on which set of bytes is being accessed. To prevent indexing errors, the X register is not used for anything else. This technique allows a single routine , such as NEXTX or NEXTY. to operate on two cursor addresses independently.

The address and mask bytes are not sufficient description of a pixel location for all operations. Since this representation is not monotonic with the position of pixels, displacements can not be easily calculated, and the joystick output does not map to it well. For this, and other reasons, a dual system of position representation is used.

Besides the address and mask bytes, two X-coordinate bytes, and one Y-coordinate byte are used to form a position representation that is monotonic.

Two bytes are required for the X-position because there are 280 locations in the X direction, and a single byte can address no more than 256. To facilitate fast table look-ups of actual pixel addresses, HIGHBYTE(X) = LOWBYTE(X) mod 70 . Thus, the high X byte has a range of 0 to 3, and the low X byte has a range of 0 to 45 (hex). In the listing, they are called TXEXT and TXPOS, respectively. They are both zero at the left edge of the screen.

Similarly, the Y-coordinate byte has a range of 0 to BF (hex), and is referred to as TYPOS in the listing. It is zero at the top of the screen. Although this places the origin at the top left corner of the screen, the data that is sent to Basic is normalized to a more acceptable coordinate system that places the origin at the lower left corner of the screen.

The routines FINDX and FINDY are used to convert these coordinate bytes to a set of address and mask bytes. Absolute moves, such as are required to support the joystick, are done by calculating a new cursor address from the coordinate representation. These three coordinate bytes (TXEXT, TYPOS, TXPOS) are an output from the joystick service routine, which is called PUTPOS in the listing.

Relative moves, as required by the key-stroke cursor commands, use two other routines: XMOVE, and YMOVE. The accumulator contents are added to the respective coordinate values, when one of these routines is called. The two routines already mentioned, FINDX, and FINDY, are used to calculate the actual address.

Address calculation from the monotonic coordinate bytes *a priori* updates the address and mask bytes as well. A necessary side-effect of address recalculation is therefore the update of coordinate bytes. The overall effect is an addressing system that is optimal for speed, and conceptually clean. It is a good basis to build higher level image processing routines on.

EDGE DETECTION

Microscope images of endothelium are noisy and show many artifacts from slide preparation. For example, in Fig. 1, there are missing edges, changes in lightness along edges, and adjacent areas of similar brightness that are not part of an edge. The Hamamatsu camera can be used to bit threshold an image. This produces a binary image that has only two values of illumination. Each point that was brighter than some reference level becomes white, and all others black.

Bit thresholding an image, such as the one in Fig. 1, will not enhance the signal to noise level. This can only be done by some operation that recognizes edges. Since it is a prerequisite to edge detection, this enhancement, if needed, has to be done by the operator. There are available, in the present image processing software package, ways of doing this. Two "brushes" can be enabled, to modify either large, or small areas of the image; under cursor control.

The little brush writes a value in the pixel location under the cursor's centre. For example, if the operator enabled this feature with a "P2" command, the secondary page pixel will be zero'd, and the primary page pixel will be set to one. The big brush is similar, except that all pixels under the cursor are modified. These work for both cursor placement modes.

The cursor could be used to trace out an outline of a cell, in order to acquire points about its perimeter. However, there is no improvement in this method over other, earlier ones. A much more interesting approach is to have the program automatically search for the edge of a cell. These edges are distinct enough, even with two bits of gray scale, and "brush" work can be used to eliminate unwanted artifacts, and to make edges continuous.

One approach to edge detection would be to place the cursor on an edge, and have it "climb along the ridge", until it returned to the starting point. A sequence of evenly-spaced position values, derived from this operation, is sufficient edge definition. However, there has to be a more systematic criteria for determining where the edge is, and how the program should follow it.

Among the problems with this first approach to edge detection are:

1) The edges have to be continuous
2) Intersections cause problems
3) Inward pointing wall fragments conflict with any method for 2.
4) Thick edges cause the program to hang-up by:
   i circling within a thick region
   ii unevenly spaced positions
   iii not recognizing its return

These problems all result from the inability of the program to recognize that it is dealing with a closed curve. Since cells always have edges that are closed curves, it seems reasonable to include this factor as an integral part of the algorithm.

The method which was used successfully to recognize cell shapes anticipates a cell that is oughly circular. With the cursor in a cell's centre, a series of radial rays are generated that cross the cell wall at some region. Each individual line is produced by a sequence of horizontal and vertical steps, as Fig. 6a illustrates. These lines are not drawn on the screen, but if they were, the pattern would look like Fig. 6b.

As each ray is extended, the distance from the centre is continually, and very accurately updated.

Pixel values are integrated, and a double byte accumulation of the product of each pixel value, and its distance from the centre, is done.

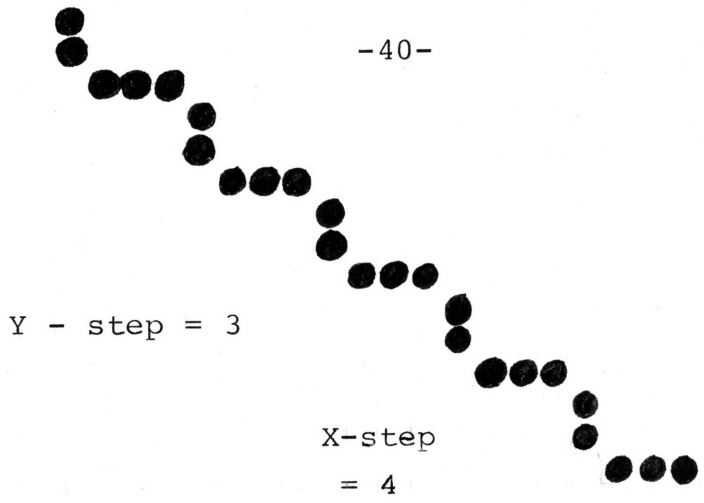


Figure 6a: A straight line as it is formed on the graphics screen

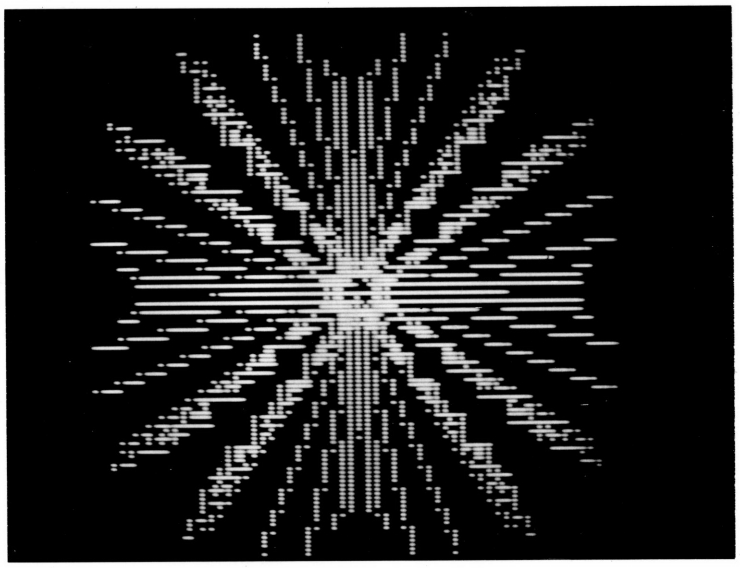


Figure 6b: Ray pattern of edge search program

When the line ends, the position of greatest luminance is calculated from these two sums. The calculated position is very close to the perceived location of the cell wall.

Although the accumulated product term is two bytes, an overflow can occur on a long ray. This happens when the accumulated sum exceeds 65,536. A typical scenerio for this is a line over 200 units long, that passes through a bright region. To eliminate this problem, and to speed up the position acquisition, a narrow tracking window is used. In Fig. 7, the window length is between -W and +W. These distances, from C, are calculated from the last ray's intersection with the cell edge. Since there will usually be no large discontinuities along the edge, this method permits the use of a small window, and tracks the edge quite well.

When no luminance is found, within the bounds of the window, one more dot is fudged, using the centre of window position as a good approximation. The window length is doubled, in anticipation of another miss, and the next ray is not fudged.

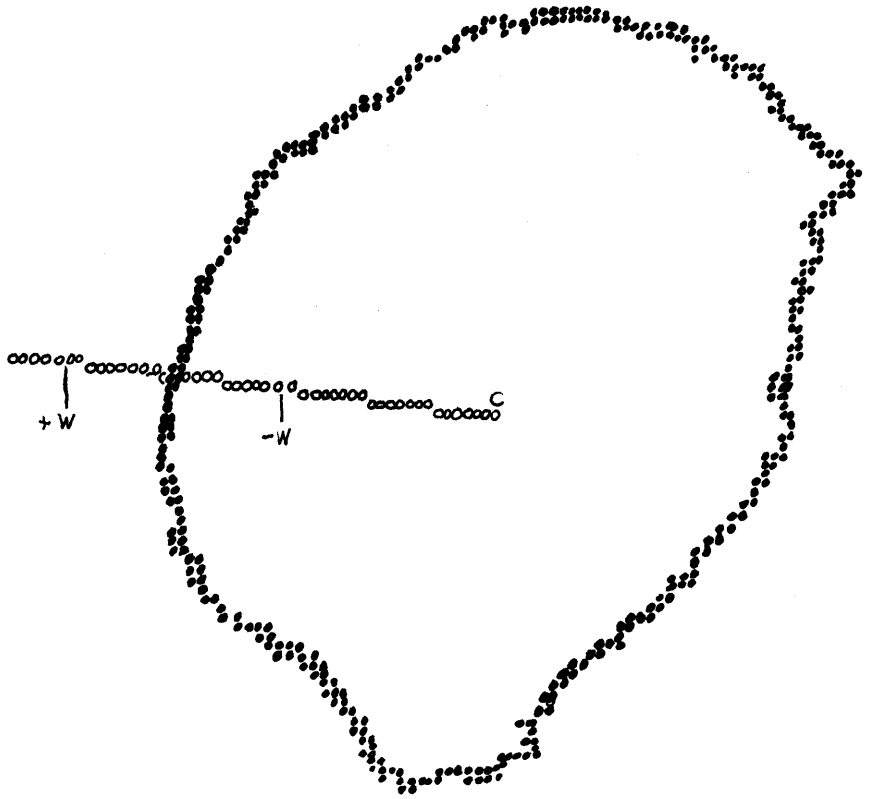


Figure 7: Dynamic window positioning

Figure 8 illustrates this use of the previous edge location to fill a gap. Subsequent rays cannot continue this pattern, since it would lead to inaccuracies. It is, nevertheless, a good feature because it offers a good solution to the miss problem with diffuse edges. Since the window is dynamically centered about the edge, noise averages out.

To improve the chances of locking back in on an edge, once it has been lost, the window length is doubled after each miss. However, this does not always guarantee that the edge will be found immediately. In Fig. 9, the sharp transition at the top results in a series of misses. The first ray, after the apex, is a miss; but it is filled in. The next ray is also a miss, despite the larger window. This can be corrected by placing the cursor closer to the apex. Figure 10 illustrates how this solves the problem, by placing the sample points closer together. The full 360 degree sweep can't be used, because the lower sharp edge will cause misses. Instead, angular segments, as shown in Fig. 11, are used.

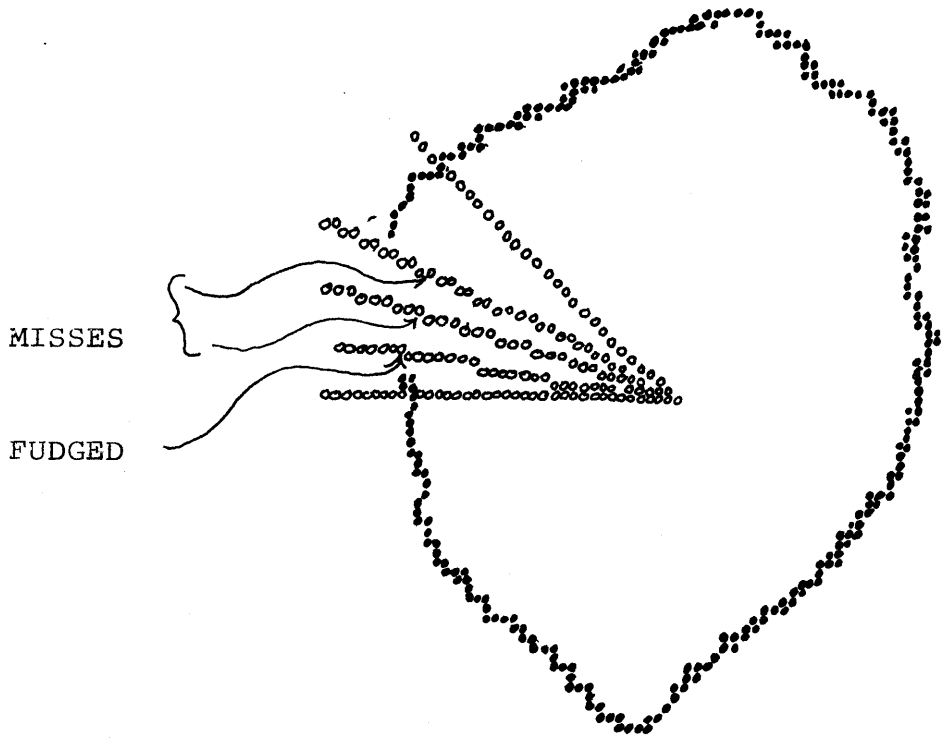


Figure 8: Filling in a gap

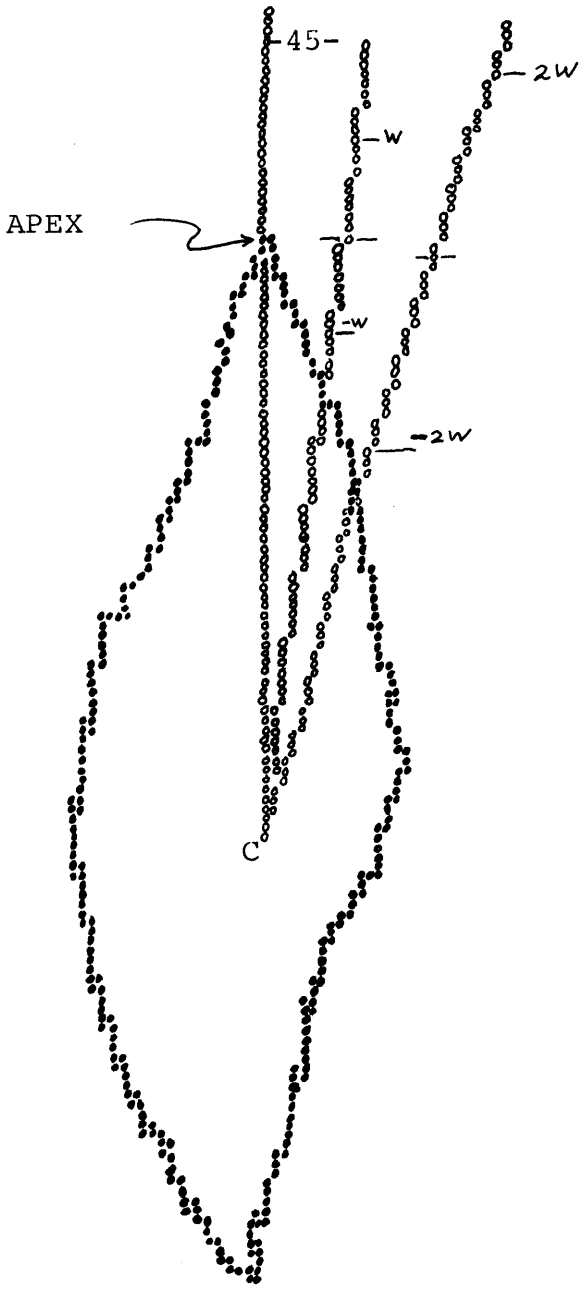


Figure 9: Problems with sharp edges

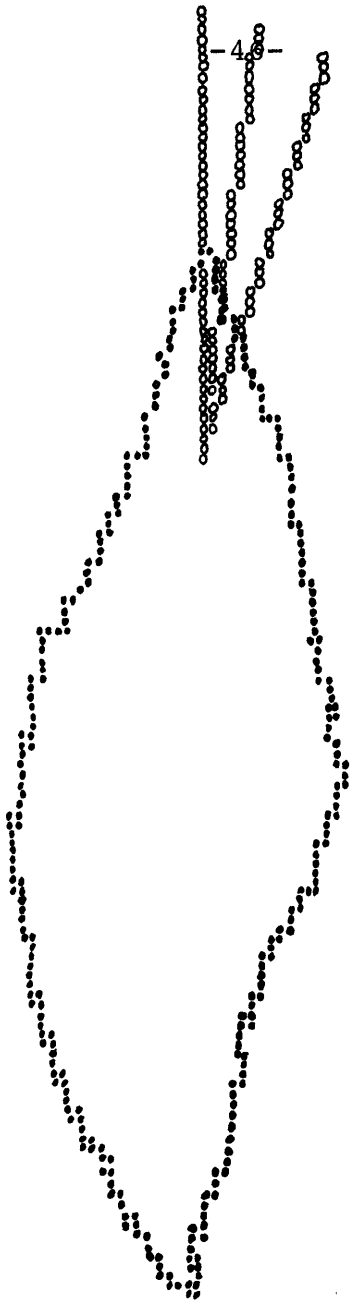


Figure 10: Resolving sharp edges

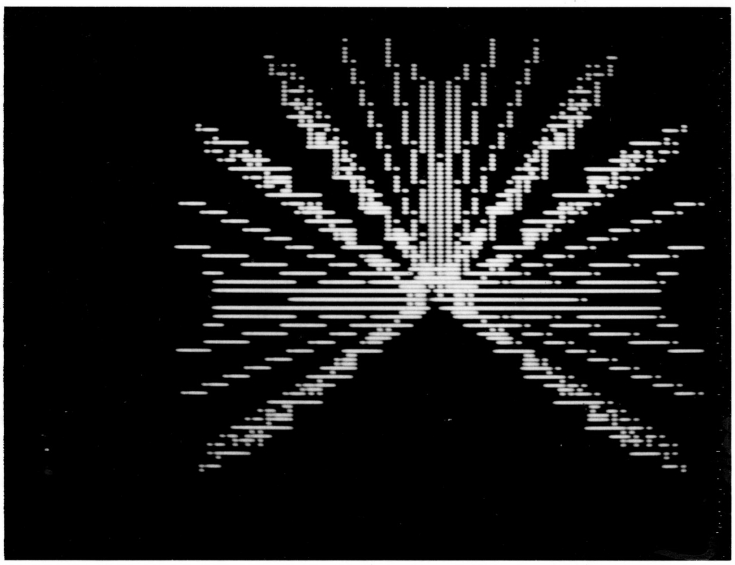


Fig. 11a: Top edge-detection angular segment, used with the "TG" command. Bottom segment the reverse

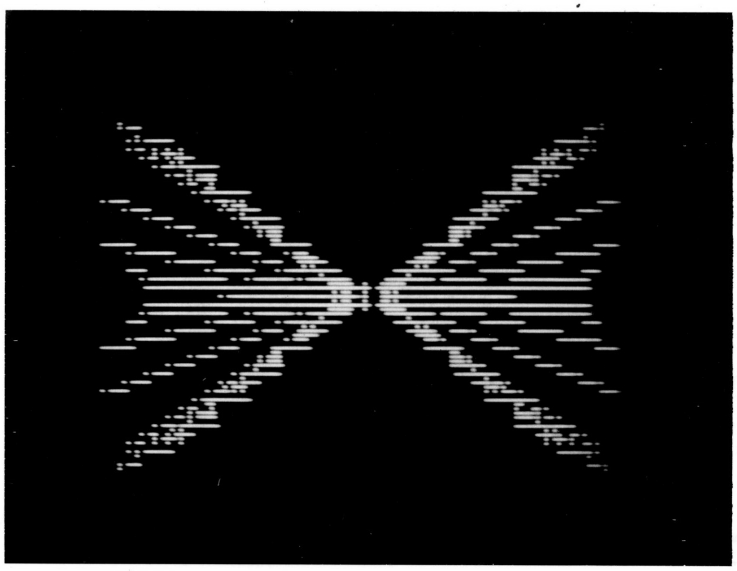


Fig. 11b: Side angular segments, used with "SG".

Messages are displayed, according to the status of an operation. After each search, a message is printed stating the number of misses. If most attempts fail, another message appears telling the operator to re-position the cursor. When the buffer is filled up, a message appears, offering the operator two choices. The buffer can be re-initialized, or the set of position values can be written in a Basic data area. Each set of three coordinate bytes is reduced to a normalized X,Y pair of bytes.

With this system, cell shapes can be acquired and transferred to Basic very quickly. It takes about 5 Sec. to acquire a well-spaced set of 64 data points. It is very accurate, with most of the points coinciding with the perceived location of the wall. This is illustrated by the actual photographs in Fig. 12 and Fig. 13. A maximum of 255 data points can be transferred to Basic this way. This is more than adequate for most purposes.

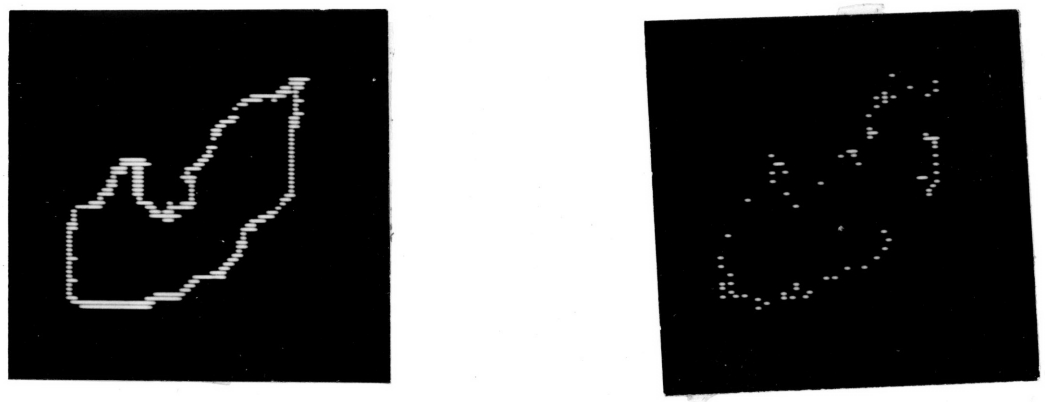


Fig. 12a: Edge detected with a TG and a MG command.

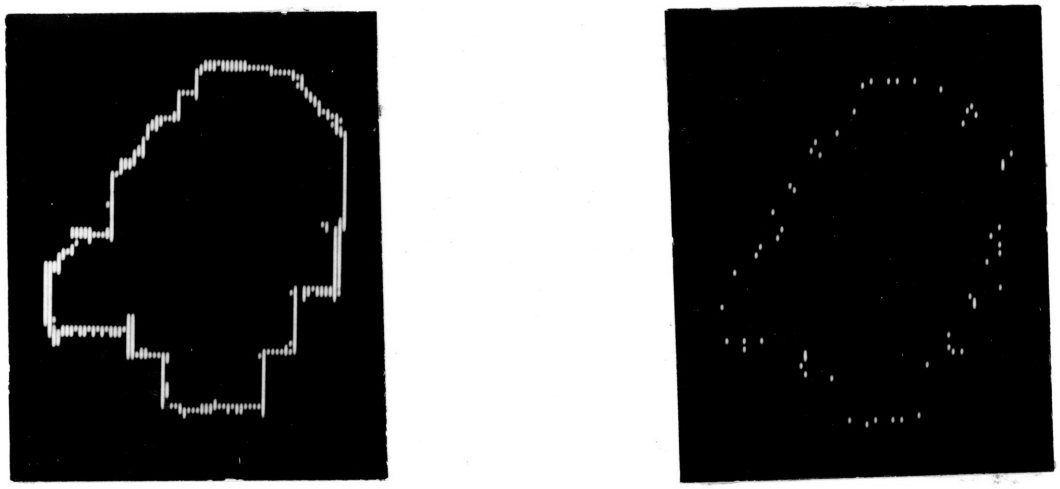


Fig. 12b: Edge detected with MG command.

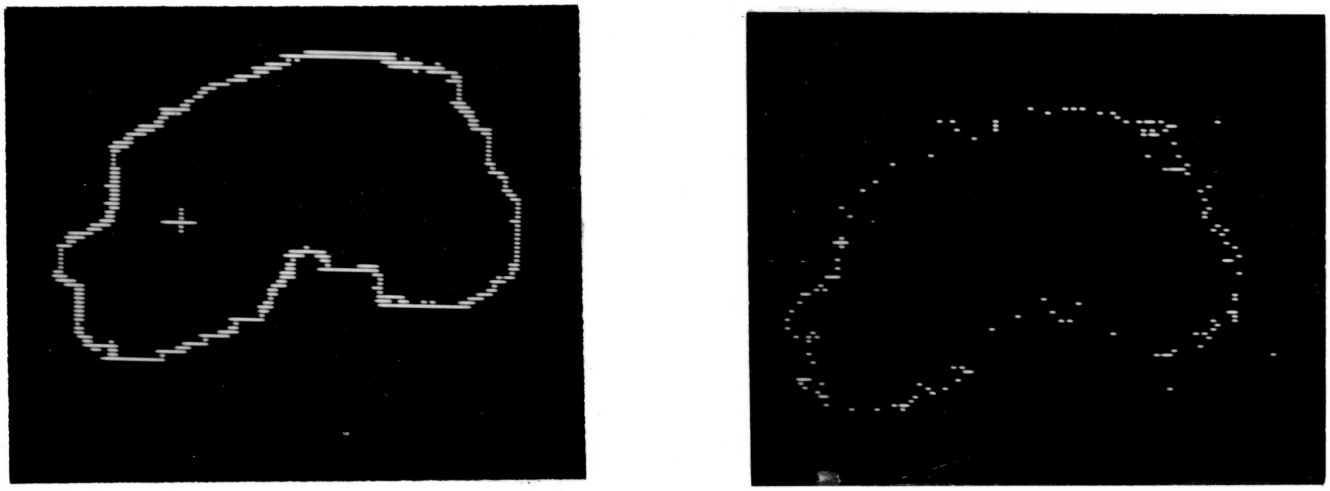


Fig. 13a: Edge detected with TG, BG, SG, and MG.

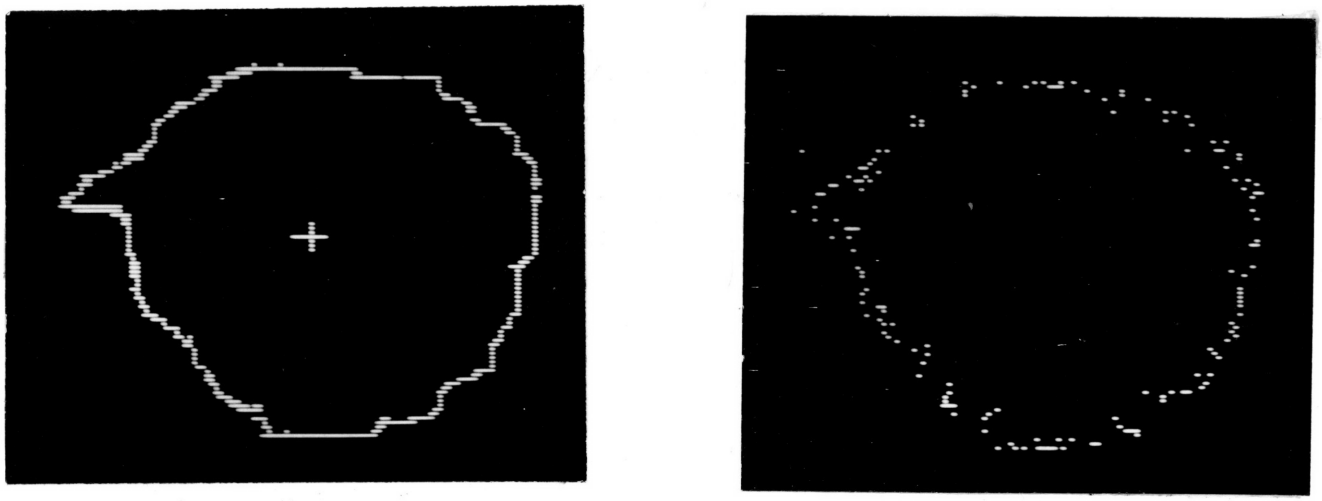


Fig. 13b: Edge detected with several of each command.

MISCELLANEOUS ROUTINES

Two task routines can be used to erase the graphics pages. In the listing, they are referred to as PBLANX and SBLANX. They run very quickly, because there is no need to address individual pixels, and the order that pixels are turned off doesn't matter.

Four programs are available for doing two-dimensional smoothing. Three, or five adjacent pixels are averaged on a line, and the centre pixel, at each step along the line is assigned that average value. Each routine has slightly different characteristics. They are called SM1, SM2, SM3, and SM4.

Two programs perform a two dimensional dervative across the screen. Negative derivatives are set to zero. This program could be more useful if more gray scale were available.

EDGEDO transforms a solid, light against dark background image to a line image of the same scene. It is useful for converting images to a format that the edge detection program can use.

Each of these routines is a TASK program, that is CALL'd from the Basic environment. Except for the screen blanking programs, they all use a program for doing an ordered scan of the screen. This program, SCAN, will do any of four screen scan patterns, and continually updates a set of pixel values that are local to the current position of the cursor.

A sequence of operations can be done to an image with these programs, to transform it to a different type of image that might have some information more accessable. Such a sequence is shown in Fig. 14.

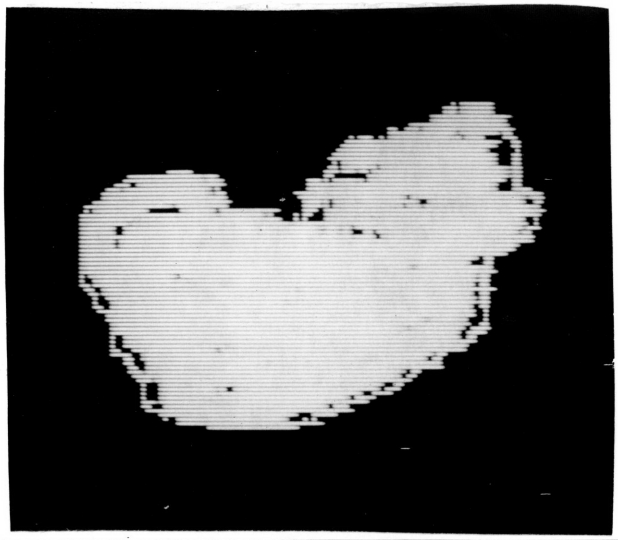


Fig. 14a: Original noisy binary image.

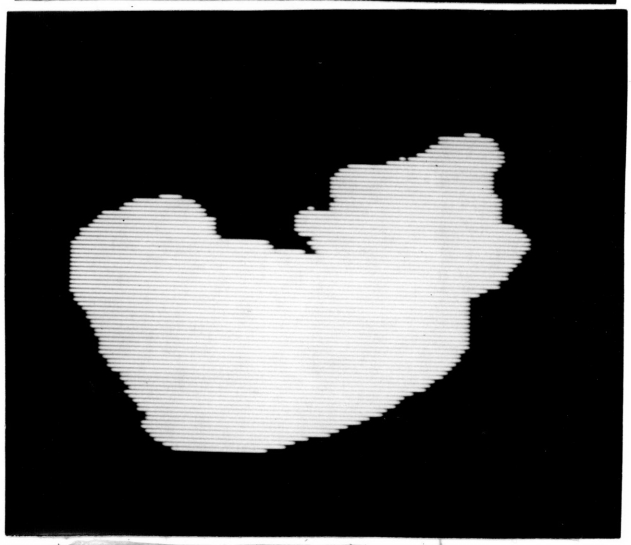


Fig. 14b: Smoothed with SM2.

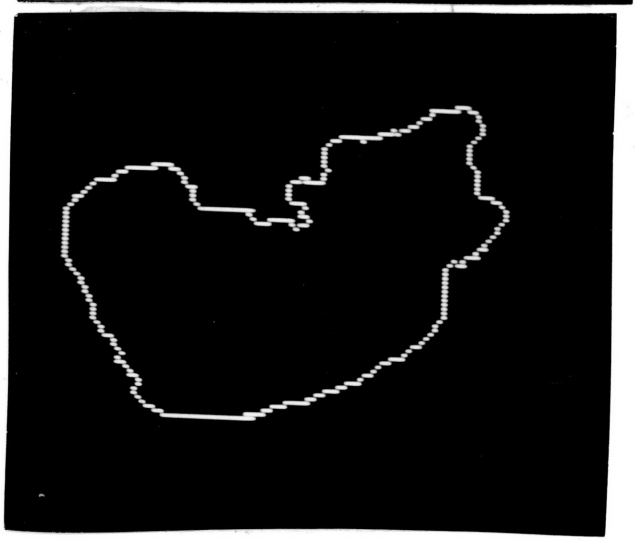


Fig. 14c: Interiors removed by edgedo.

CONCLUSION

The image processing software resides in a free area of memory immediately following the screen pages. Basic calls to image processing routines have the following syntax: CALL (exp) ; where exp is a decimal number, not greater than 65,536, and corresponding to an entry point.

There are nine entry points to different task routines, and one entry point to an interactive routine. The reason why they were not all combined in one large interactive program, is to minimize the danger of accidental image erasure. These programs have already been discussed, but they are summarized again in table 3.

The interactive program has several keystroke commands. They are all described in table 4.

The image processing system is designed to make the best use of system resources and operator interaction. A relatively simple algorithm is used to detect edges, but it is quite reliable and fast.

This system was designed with a modular structure that simplifies later expansion. It is only necessary for a programmer to satisfy the input requirements of each module, and the precedence of module operations to add other high-level operations. The software listing that follows (Appendix A) is thoroughly commented, and provides a concise guide to the program's structure. It is my hope that others will find it a useful source of knowledge of the program and of assembly methods.

Figure 15: Table of CALLable routines

NAME	CALL ADDRESS	PURPOSE
START	24576	Main interactive program, detects edges, draws on screens, two cursor modes
EDGEDO	24579	Task program, removes interior of binary image on primary page, leaving only edge.
SM1-SM4	24582,24585,24587,24590	Task routines, do two-dimensional smoothing
LAPLA1, LAPLA2	24593,24596	Task routines, do derivative in both directions, zero neg. derivative
PBLANX,SBLANX	24599,24602	Task routines, clear the primary, or secondary screens

Figure 16: Table of Key-Stroke Commands

KEY_CODE OPERATION PERFORMED

J.....enable joystick to control cursor

H.....enable key-stroke commands to move cursor

U .....move up decimal places, if in key mode

D .....move down decimal places, if in key mode

R .....move right decimal places, if in key mode

L .....move left decimal places, if in key mode

esc.....return to Basic

sp sp.....return to Basic

sp P .....write , should be less than 4, to screen at cursor

sp @ .....write *cursor to screen, Big Brush

sp E.....turn off both brushes

MG.....360 degree sweep for cell edge

TG.....270 degree sweep, centred at top

BG.....270 degree sweep, centred at bottom

SG.....two 90 degree sweeps, on either side

$.....flush buffer

X.....transfer buffer to Basic, and exit

ZØ.....display secondary screen

z1.....display primary screen

BIBLIOGRAPHY

1) Bussolari, Steven R. *An Apparatus For Subjecting Vascular Endothelium to Controlled Fluid Shear Stress IN VITRO*, S.M. Thesis, M.I.T., M.E., May , 1980.

2) York, David P. *A Method For Quantifying Cellular Alignment of Vascular Endothelium Subjected to Fluid Shear Stress*, S.B. Thesis, M.I.T.,M.E., June, 1980.

3) ANSI/IEEE, *IEEE Standard Digital Interface for Programmable Instrumentation*, std 488-1978 IEEE Inc., 345 East 47th N.Y.

4)Hamamatsu, *M999-04 General Purpose Interface Bus (GPIB)* Manual.

APPENDIX A

THE PROGRAM LISTING

```
0800             1          OCM "PR#1"      ;
0800             2  ADDRL1 EPZ $6           ;FIRST CURSOR ADDRESS
0800             3  ADDRH1 EPZ $7           ;
0800             4  ADDRL2 EPZ $8           ;SECOND CURSOR ADDRESS
0800             5  ADDRH2 EPZ $9           ;
0800             6  STAX   EPZ $1S          ;PIXEL STAX
0800             7  BUFF   EPZ $1B          ;POSITION BUFFER ADDRESS
0800             8  JOYSTK EPZ $1D          ;JOYSTIC BASE PAGE ADDRESS
0800             9  STRING EPZ $6F          ;BASIC STRING POINTER
0800            10  MTAB   EPZ $D6          ;START ADDRESS FOR MESSAGES
0800            11  MBUF   EPZ $CE          ;CHARACTER POINTER FOR MESSAGES
6000            12          ORG $6000       ;
6000            13          OBJ $800        ;
6000            14  ;                       ;
6000            15  ;                       ;
6000 4CD460     16          JMP START       ;ENTRY POINT TO INTERACTIVE PROG.
6003 4C066E     17          JMP EDGEDO      ;BINARY IMAGE EDGE ENHANCEMENT
6006 4C2C6E     18          JMP SM1         ;SMOOTH ROUTINES
6009 4C486E     19          JMP SM2         ;   "      "
600C 4C626E     20          JMP SM3         ;   "      "
600F 4C866E     21          JMP SM4         ;   "      "
6012 4C9B6E     22          JMP LAPLA1      ;X-DERIVATIVE
6015 4CB36E     23          JMP LAPLA2      ;Y-DERIVATIVE
6018 4C2269     24          JMP PBLANX      ;PRIMARY SCREEN BLANX
601B 4C2769     25          JMP SBLANX      ;SECONDARY SCREEN BLANX
601E 60         26          RTS             ;DON'T CALL HERE
601F 60         27          RTS             ;
6020 60         28          RTS             ;
6021 60         29          RTS             ;
6022            30  ;                       ;
6022            31  ;                       ;
6022            32  ;                       ;
6022 206960     33  BASIC  JSR DAR          ;RETURN SCREEN TO TEXT MODE
6025 60         34          RTS             ;RTS TO BASIC
6026            35  ;                       ;THIS ROUTINE POLLS THE KEYBOARD
6026            36  ;                       ;FOR A CHARACTER, WITHOUT WAITS
6026 AD00C0     37  QUES   LDA $C000        ;GET THE CHARACTER
6029 8D10C0     38          STA $C010       ;RESET KEYBOARD FLAG
602C 18         39          CLC             ;
602D 6980       40          ADC #$80        ;CLEAR SIGN BIT
602F C9A0       41          CMP #$A0        ;IS IT A SPACE?
6031 D025       42          BNE FOO         ;NO, GOTO OTHER TESTS
6033 209960     43          JSR TYPIN       ;YES, GET ANOTHER CHARACTER
6036 C9A0       44          CMP #$A0        ;NOT ANOTHER SPACE!
6038 D003       45          BNE NOTOUT      ;GOOD, SOMETHING ELSE
603A 203382     46          JSR OUTSET      ;GOOD-BYE
603D C900       47  NOTOUT CMP #$00        ;"P" COMMAND FOR SMALL BRUSH?
603F D006       48          BNE NPAINT      ;
6041 203863     49          JSR PAINT       ;SETUP FOR LITTLE BRUSH
6044 4C5860     50          JMP FOO         ;
6047 C9C5       51  NPAINT CMP #$C5        ;"E" COMMAND FOR BRUSHES OFF?
6049 D006       52          BNE NERASE      ;
604B 204E63     53          JSR ERASE       ;TURN OFF BRUSHES
604E 4C5860     54          JMP FOO         ;
6051 C9C0       55  NERASE CMP #$C0        ;"@" COMMAND FOR BIG BRUSH?
6053 D003       56          BNE FOO         ;
6055 206263     57          JSR BLOTS
6058 60         58  FOO    RTS
6059            59  ;
```

```
6059            60  ;
6059 200961     61  LITE   JSR INIT      ;LITE INITIATES BUFFERS, AND
605C 207760     62         JSR HAFOFF    ;ENABLES THE HIGH-RES SCREENS
605F AD50C0     63         LDA $C050     ;FULL GRAPHICS MODE, NO TEXT
6062 AD57C0     64         LDA $C057     ;GRAPHICS SOFT SWITCH
6065 207B60     65         JSR PRION     ;HIGH-RES SOFT SWITCH
6068 60         66         RTS           ;SET PRIMARY PAGE ON
6069            67  ;                    ;
6069            68  ;                    ;
6069 AD51C0     69  DAR    LDA $C051     ;DAR SETS THE TEXT MODE
606C 207B60     70         JSR PRION     ;TEXT SOFT SWITCH
606F 207760     71         JSR HAFOFF    ;PRIMARY PAGE (TEXT)
6072 60         72         RTS           ;NO JUNK PLEASE
6073            73  ;                    ;
6073 AD53C0     74  HAFON  LDA $C053     ;ENABLES THE TEXT WINDOW
6076 60         75         RTS           ;SPLIT SCREEN SOFT SWITCH
6077            76  ;                    ;
6077            77  ;                    ;
6077 AD52C0     78  HAFOFF LDA $C052     ;GETS RID OF SPLIT SCREEN
607A 60         79         RTS           ;
607B            80  ;                    ;ENABLES PRIMARY PAGE
607B            81  ;                    ;
607B AD54C0     82  PRION  LDA $C054     ;PRIMARY PAGE SOFT SWITCH
607E A980       83         LDA #$80      ;
6080 8D3F72     84         STA P6FLAG    ;SET PAGE FLAG FOR PRIMARY PAGE
6083 60         85         RTS           ;
6084            86  ;                    ;
6084            87  ;                    ;ENABLES SECONDARY PAGE
6084 AD55C0     88  SECON  LDA $C055     ;SECONDARY PAGE SOFT SWITCH
6087 A900       89         LDA #$0       ;
6089 8D3F72     90         STA P6FLAG    ;SET PAGE FLAG FOR SECONDARY
608C 60         91         RTS           ;
608D            92  ;                    ;
608D            93  ;                    ;
608D 20EDFD     94  CHAR   JSR $FDED     ;PRINTS AN "ASCII" CHARACTER
6090 60         95         RTS           ;
6091            96  ;                    ;
6091            97  ;                    ;
6091 20DAFD     98  NUMBR  JSR $FDDA     ;PRINTS ACCUMULATOR CONTENTS
6094 60         99         RTS           ;
6095           100  ;                    ;
6095           101  ;                    ;
6095 208EFD    102  LF     JSR $FD8E     ;PRINTS A LINE FEED
6098 60        103         RTS           ;
6099           104  ;                    ;
6099           105  ;                    ;
6099 200CFD    106  TYPIN  JSR $FD0C     ;GETS A CHARACTER, WITH WAIT
609C 60        107         RTS           ;
609D           108  ;                    ;
609D           109  ;                    ;
6090 209960    110  TWODIG JSR TYPIN     ;GET ANY CHARACTER
60A0 290F      111         AND #$F       ;WANT ONE HEX DIGIT
60A2 A00A      112         LDY #$A       ;
60A4 20AD6C    113         JSR MULT      ;MULTIPLY BY 10 (DECIMAL)
60A7 8D3672    114         STA TEMP1     ;SAVE
60A9 209960    115         JSR TYPIN     ;GET ANOTHER DIGIT
60AD 290F      116         AND #$F       ;
60AF 18        117         CLC           ;
60B0 6D3672    118         ADC TEMP1     ;ADD TWO DIGITS
60B3 60        119         RTS           ;
60B4           120  ;                    ;
```

```
60B4            121  ;                          ;
60B4            122  ;                          ;
60B4 AD70C0     123  GETPOS LDA $C070           ;GET COUNTER GOING
60B7 B11D       124  WHERE  LDA (JOYSTK),Y     ;READ NUMBER
60B9 F00B       125         BEQ FINALE          ;ALREADY ZERO ??!
60BB EE3772     126         INC TEMP2           ;NO, BUMP COUNTERS
60BE D0F7       127         BNE WHERE           ;OVERFLOW?
60C0 EE3672     128         INC TEMP1           ;BUMP NEXT BYTE
60C3 4CB760     129         JMP WHERE           ;
60C6 AD3672     130  FINALE LDA TEMP1          ;
60C9 1008       131         BPL GETRID          ;NO NEGATIVE NUMBERS PLEASE
60CB A900       132         LDA #$0             ;ZERO THE WORKS
60CD 8D3672     133         STA TEMP1           ;
60D0 8D3772     134         STA TEMP2           ;
60D3 60         135  GETRID RTS                ;
60D4            136  ;                          ;
60D4            137  ;                          ;
60D4            138  ;                          ;
60D4            139  ;                          ;
60D4            140  ;                          ;
60D4 205960     141  START  JSR LITE            ;TURN ON THE LITES
60D7 200961     142         JSR INIT            ;READY,
60DA A980       143         LDA #$80            ;ACTION,
60DC 8D4672     144         STA PSOURC          ;ROLL-EM.,AND SET FLAGS
60DF 8D4272     145         STA OFLAG           ;EXIT FLAG NEGATED
60E2 8D3D72     146         STA BLTFLG          ;NO BRUSHES
60E5 AD4272     147  LOPS   LDA OFLAG          ;HAD ENOUGH?
60E8 3003       148         BMI STAYIN          ;NO, DO MORE
60EA 4C2260     149         JMP BASIC           ;EXIT TO BASIC
60ED A200       150  STAYIN LDX #$0            ;USE FIRST PIXEL ADDRESS
60EF 207761     151         JSR PLACES          ;GET THE CURSOR POSITION
60F2 20386B     152         JSR XORCUR          ;DO A CURSOR
60F5 202660     153         JSR QUES            ;POLL FOR A COMMAND
60F8 8D4572     154         STA CMMND           ;SAVE IT FOR A RAINY DAY
60FB 209B61     155         JSR INPLAC          ;COMMAND INTERPRETER
60FE AD4072     156  CURGO  LDA PFLAG          ;ANY BRUSHES ON?
6101 10E2       157         BPL LOPS            ;NO
6103 206B63     158         JSR DRAW            ;YES, DO SOME DAMAGE
6106 4CE560     159         JMP LOPS            ;
6109            160  ;                          ;
6109            161  ;                          ;
6109            162  ;                          ;
6109 202E61     163  INIT   JSR BUFSET          ;SETUP POSITION BUFFER
610C AD0272     164         LDA ST1             ;CURSOR ADDRESS STAX
610F 8519       165         STA STAX            ;
6111 AD0372     166         LDA ST1+1           ;
6114 851A       167         STA STAX+1          ;
6116 AD3670     168         LDA MESTAB          ;MESSAGE POINTERS
6119 85D6       169         STA MTAB            ;
611B AD3770     170         LDA MESTAB+1        ;
611E 85D7       171         STA MTAB+1          ;
6120 A964       172         LDA #$64            ;SETUP JOYSTICK BASE PAGE LOC.
6122 851D       173         STA JOYSTK          ;
6124 A9C0       174         LDA #$C0            ;
6126 851E       175         STA JOYSTK+1        ;
6128 A900       176         LDA #$0             ;NO BRUSHES
612A 8D4072     177         STA PFLAG           ;
612D 60         178         RTS                 ;
612E            179  ;                          ;
612E            180  ;                          ;
612E            181  ;                          ;
```

```
612E AD0472  182  BUFSET LDA BUF1      ;INITIALIZE POSITION BUFFER
6131 8518    183       STA BUFF      ;
6133 AD0572  184       LDA BUF1+1    ;
6136 851C    185       STA BUFF+1    ;
6138 A97F    186       LDA #$7F      ;INITIALIZE DUMMY POSITION
613A 8D5E72  187       STA XEXT      ;BYTES TO LARGE NUMBERS,
613D 8D6072  188       STA XPOS      ;SO THAT POSITIONS CAN BE
6140 A900    189       LDA #$00      ;NORMALIZED LATER
6142 8D5F72  190       STA YPOS      ;
6145 A900    191       LDA #$0       ;NO POSITIONS STORED
6147 8D6272  192       STA ITCNT     ;
614A 8D3E72  193       STA CFLAG     ;LOTS OF ROOM
614D 60      194       RTS           ;
614E         195  ;                  ;
614E         196  ;                  ;
614E         197  ;                  ;
614E C9CA    198  COMAND CMP #$CA     ;NEW CURSOR MODE
6150 F010    199       BEQ JSTIX     ;"J" SELECTS JOYSTICK
6152 C9C8    200       CMP #$C8      ;
6154 F001    201       BEQ KEY       ;"H" SELECTS KEY-STROKE CURSOR
6156 60      202       RTS           ;
6157         203  ;                  ;
6157         204  ;                  ;
6157 A900    205  KEY    LDA #$0      ;SET FLAG FOR KEY-STROKES
6159 8D4672  206       STA PSOURC    ;
615C A003    207       LDY #$3       ;GET MESSAGE TABLE OFFSET
615E 206D61  208       JSR PEEP      ;PRINT THE MESSAGE
6161 60      209       RTS           ;
6162         210  ;                  ;
6162         211  ;                  ;
6162 A980    212  JSTIX  LDA #$80     ;SET FLAG FOR JOYSTICK
6164 8D4672  213       STA PSOURC    ;
6167 A000    214       LDY #$0       ;GET MESSAGE TABLE OFFSET
6169 206D61  215       JSR PEEP      ;
616C 60      216       RTS           ;
616D         217  ;                  ;
616D         218  ;                  ;
616D 20FC63  219  PEEP   JSR MSG      ;PRINT MESSAGE, USING Y-OFFSET
6170 209960  220       JSR TYPIN     ;WAIT FOR A KEYPRESS
6173 207760  221       JSR HAFOFF    ;TURN OFF MESSAGE SCREEN
6176 60      222       RTS           ;
6177         223  ;                  ;
6177         224  ;                  ;
6177         225  ;                  ;
6177         226  ;                  ;
6177         227  ;                  ;
6177 AD4572  228  PLACES LDA CHRCD    ;GET THE COMMAND CHARACTER
617A 204E61  229       JSR COMAND    ;CHECK FOR CURSOR MODES
617D AD4672  230       LDA PSOURC    ;GET CURSOR MODE FLAG
6180 1003    231       BPL NUMOV     ;POSITIVE, THEN KEY-STROKES
6182 4C3C64  232       JMP PUTPOS    ;NEGATIVE, JOYSTICK
6185 AD4172  233  NUMOV  LDA MFLAG     ;CURSOR MOVE COMMAND?
6188 D001    234       BNE MOVERS    ;YES, INTERPRET IT
618A 60      235       RTS           ;
618B AD4372  236  MOVERS LDA MOVE     ;GET MOVE VALUE
618E AC4472  237       LDY AXIS      ;WHICH AXIS TO MOVE ON?
6191 3004    238       BMI HORMOV    ;NEG., HORIZONTAL
6193 204360  239       JSR XMOVE     ;POS., VERTICAL
6196 60      240       RTS           ;
6197 20B76D  241  HORMOV JSR YMOVE     ;DO ADDRESS-CALCULATION YMOVE
619A 60      242       RTS           ;
```

```
619B           243  ;
619B           244  ;
619B           245  ;
619B 8D4172    246  INPLAC STA MFLAG
619E C9D5      247          CMP #$D5
61A0 D009      248          BNE NOTUP
61A2 209D60    249          JSR TWODIG
61A5 202962    250          JSR DONEG
61A8 4CB261    251          JMP HOROUT
61AB C9C4      252  NOTUP   CMP #$C4
61AD D00E      253          BNE NOTDWN
61AF 209D60    254          JSR TWODIG
61B2 8D4372    255  HOROUT  STA MOVE
61B5 A980      256          LDA #$80
61B7 8D4472    257          STA AXIS
61BA 4C2262    258          JMP FULSCR
61BD C9CC      259  NOTDWN  CMP #$CC
61BF D009      260          BNE NOTLFT
61C1 209D60    261          JSR TWODIG
61C4 202962    262          JSR DONEG
61C7 4CD161    263          JMP SIDOUT
61CA C9D2      264  NOTLFT  CMP #$D2
61CC D00E      265          BNE NOTRHT
61CE 209D60    266          JSR TWODIG
61D1 8D4372    267  SIDOUT  STA MOVE
61D4 A900      268          LDA #$0
61D6 8D4472    269          STA AXIS
61D9 4C2262    270          JMP FULSCR
61DC A000      271  NOTRHT  LDY #$0
61DE 8C4172    272          STY MFLAG
61E1 C9D4      273          CMP #$D4
61E3 D003      274          BNE NOTOP
61E5 4C3762    275          JMP TOPRAY
61E8 C9C2      276  NOTOP   CMP #$C2
61EA D003      277          BNE NOTBOT
61EC 4C9962    278          JMP BOTRAY
61EF C9CD      279  NOTBOT  CMP #$CD
61F1 D003      280          BNE NOTMID
61F3 4CBB62    281          JMP MIDRAY
61F6 C9D3      282  NOTMID  CMP #$D3
61F8 D003      283          BNE NOTSID
61FA 4CDD62    284          JMP SIDRAY
61FD C99B      285  NOTSID  CMP #$9B
61FF D009      286          BNE NOTBAS
6201 203362    287          JSR OUTSET
6204 207B60    288          JSR PRION
6207 4C2262    289          JMP FULSCR
620A C9A4      290  NOTBAS  CMP #$A4
620C D006      291          BNE NOTBUF
620E 202E61    292          JSR BUFSET
6211 4C2262    293          JMP FULSCR
6214 C9D8      294  NOTBUF  CMP #$D8
6216 D003      295          BNE NOTXFR
6218 4C7263    296          JMP XFER
621B C9DA      297  NOTXFR  CMP #$DA
621D D003      298          BNE FULSCR
621F 202C63    299          JSR SCREEN
6222 207760    300  FULSCR  JSR HAFOFF
6225 20386B    301  NOFOO   JSR XORCUR
6228 60        302          RTS
6229           303  ;
```

```
;
;
;
;SET FLAG FOR CURSOR MOVE
;"U" FOR UP
;
;GET DISTANCE
;TAKE NEGATIVE ( TO GO UP )
;LEAVE FLAGS SET FOR MOVE
;"D" FOR DOWN
;
;
;SAVE DISTANCE OF MOVE
;
;NEG. FOR VERT. MOVE
;
;"L" FOR LEFT
;
;
;NEGATIVE DISTANCE TO GO LEFT
;
;"R" FOR RIGHT
;
;
;
;
;
;CLEAR MOVE FLAG, OR
;CURSOR WON'T STOP
;"T" IS FOR TOP EDGE SCAN
;
;
;"B" IS FOR BOTTOM EDGE SCAN
;
;
;"M" IS FOR MIDDLE SCAN
;
;
;"S" IS FOR SIDE SCANS
;
;
;"ESC" RETURNS TO BASIC
;
;
;
;
;"*" CLEARS POSITION BUFFER
;
;
;
;"X" TRANSMITS BUFFER TO BASIC
;
;
;"Z" CHANGES SCREENS
;
;
;TURN OFF SPLIT SCREEN RIGHT AWAY
;DO THE CURSOR NOW
;
;
```

```
6229            304  ;                              ;
6229            305  ;                              ;
6229            306  ;                              ;
6229 8D3672     307  DONEG   STA TEMP1              ;SAVE NUMBER
622C A900       308          LDA #$0                ;
622E 38         309          SEC                    ;
622F ED3672     310          SBC TEMP1              ;GET NEGATIVE OF NUMBER
6232 60         311          RTS                    ;
6233            312  ;                              ;
6233 8E4272     313  OUTSET  STX OFLAG              ;POSITIVE OUT FLAG SENDS TO BASIC
6236 60         314          RTS                    ;
6237            315  ;                              ;
6237            316  ;                              ;
6237            317  ;                              ;
6237            318  ;                              ;
6237 201C63     319  TOPRAY  JSR INCHEX             ;CHECK THAT THERE IS NO REPEATED
623A C080       320          CPY #$80               ;CHARACTER, Y=$80 IF SO
623C F0E7       321          BEQ NOFDO              ;
623E C9C7       322          CMP #$C7               ;ONLY A "G" WORKS HERE, ALLOWS
6240 D0E3       323          BNE NOFDO              ;ABORT, IF CURSOR IS TOO FRISKY
6242 203B6B     324          JSR XORCUR             ;GET RID OF CURSOR EVIDENCE
6245 A907       325          LDA #$7                ;
6247 8D1772     326          STA QUID               ;START AT 225 DEGREES
624A A904       327          LDA #$4                ;
624C 8D1872     328          STA LQUID              ;END AT 315 DEGREES
624F 201565     329          JSR DOLOOP             ;DO CLOCK-WISE SCAN
6252 205B62     330          JSR MISSES             ;PRINT MISSES
6255 60         331          RTS                    ;
6256            332  ;                              ;
6256            333  ;                              ;
6256 203B6B     334  MISSES  JSR XORCUR             ;REPLACE THE CURSOR
6259 206062     335          JSR MISMES             ;PRINT MISSES
625C 203B6B     336          JSR XORCUR             ;REMOVE CURSOR
625F 60         337          RTS                    ;NOTE- CURSOR DISCIPLIN MAINTAINED
6260            338  ;                              ;
6260            339  ;                              ;
6260 AD3F72     340  MISMES  LDA PGFLAG             ;ON SCREEN PAGE #1 ?
6263 3001       341          BMI DO.MES             ;YES, DO MESSAGE
6265 60         342          RTS                    ;NO, GO AWAY
6266 209560     343  DO.MES  JSR LF                 ;
6269 209560     344          JSR LF                 ;GET RID OF EARLIER STUFF
626C AD3E72     345          LDA CFLAG              ;
626F 1006       346          BPL NOSHOW             ;BUFFER FULL?
6271 A009       347          LDY #$9                ;
6273 206D61     348          JSR PEEP               ;PRINT BUFFER FULL MESSAGE
6276 60         349          RTS                    ;
6277 AD0F72     350  NOSHOW  LDA FAULTS             ;GET MISSES
627A 3018       351          BMI MOSTN              ;A FEW, OR A LOT ?
627C 209160     352          JSR NUMBR              ;PRINT NUMBER OF MISSES
627F A006       353          LDY #$6                ;
6281 200B64     354  GOMESG  JSR DOMSG              ;DO "MISSES"
6284 209560     355          JSR LF                 ;
6287 209960     356          JSR TYPIN              ;WAIT FOR OPERATOR TO TYPE
628A 209560     357          JSR LF                 ;
628D 209560     358          JSR LF                 ;
6290 209560     359          JSR LF                 ;
6293 60         360          RTS                    ;
6294 A00C       361  MOSTN   LDY #$C                ;"REPOSITION CURSOR" MESSAGE
6296 4C8162     362          JMP GOMESG             ;
6299            363  ;                              ;
6299            364  ;                              ;
```

```
6299 201C63   365  BOTRAY JSR INCHEX     ;NO REPEATS PLEASE
629C C080     366        CPY #$80       ;
629E D007     367        BNE BOTGO      ;
62A0 C9C7     368        CMP #$C7       ;ONLY A "G" WILL DO
62A2 F003     369        BEQ BOTGO      ;
62A4 4C2562   370        JMP NOFOO      ;
62A7 20386B   371  BOTGO  JSR XORCUR     ;GET RID OF CURSOR
62AA A903     372        LDA #$3        ;DO FROM 45 DEGREES
62AC 8D1772   373        STA QUID       ;
62AF A900     374        LDA #$0        ;TO 135 DEGREES, CLOCKWISE
62B1 8D1872   375        STA LQUID      ;DO IT
62B4 201565   376        JSR DOLOOP     ;
62B7 205662   377        JSR MISSES     ;
62BA 60       378        RTS            ;
62BB          379  ;                    ;
62BB          380  ;                    ;
62BB          381  ;                    ;
62BB 201C63   382  MIDRAY JSR INCHEX     ;SAME AS BEFORE
62BE C080     383        CPY #$80       ;
62C0 D007     384        BNE MIDGO      ;
62C2 C9C7     385        CMP #$C7       ;
62C4 F003     386        BEQ MIDGO      ;
62C6 4C2562   387        JMP NOFOO      ;
62C9 20386B   388  MIDGO  JSR XORCUR     ;GET RID OF CURSOR
62CC A900     389        LDA #$0        ;START AT THE BEGINNING
62CE 8D1772   390        STA QUID       ;AND CONTINUE
62D1 A907     391        LDA #$7        ;UNTIL YOU REACH THE END
62D3 8D1872   392        STA LQUID      ;THEN STOP
62D6 201565   393        JSR DOLOOP     ;
62D9 205662   394        JSR MISSES     ;
62DC 60       395        RTS            ;
62DD          396  ;                    ;
62DD          397  ;                    ;
62DD          398  ;                    ;
62DD 201C63   399  SIDRAY JSR INCHEX     ;
62E0 C9C7     400        CMP #$C7       ;
62E2 F007     401        BEQ SIDGO      ;
62E4 C080     402        CPY #$80       ;
62E6 D003     403        BNE SIDGO      ;
62E8 4C2562   404        JMP NOFOO      ;
62EB 20386B   405  SIDGO  JSR XORCUR     ;BLANK OUT CURSOR
62EE A907     406        LDA #$7        ;FIRST 225 DEGREES
62F0 8D1772   407        STA QUID       ;TO
62F3 A900     408        LDA #$0        ;135 DEGREES
62F5 8D1872   409        STA LQUID      ;
62F8 201565   410        JSR DOLOOP     ;
62FB AD0F72   411        LDA FAULTS     ;
62FE 8D3A72   412        STA TEMP5      ;SAVE THOSE FIRST MISSES
6301 A903     413        LDA #$3        ;THEN DO 45 DEGREES
6303 8D1772   414        STA QUID       ;TO
6306 A904     415        LDA #$4        ;315 DEGREES
6308 8D1872   416        STA LQUID      ;
630B 201565   417        JSR DOLOOP     ;
630E AD3A72   418        LDA TEMP5      ;
6311 18       419        CLC            ;
6312 6D0F72   420        ADC FAULTS     ;
6315 8D0F72   421        STA FAULTS     ;
6318 205662   422        JSR MISSES     ;PRINT COMBINED MISSES
6318 60       423        RTS            ;
631C          424  ;                    ;
631C          425  ;                    ;
```

```
631C           426  ;                              ;
631C           427  ;                              ;
631C           428  ;                              ;
631C           429  ;                              ;
631C A000      430  INCHEX LDY #$0                  ;READS A CHARACTER AND CHEX
631E 8D3672    431         STA TEMP1                ;THAT IT IS NOT THE SAME AS THE
6321 209960    432         JSR TYPIN                ;LAST COMMAND; RETURNS $80 IN Y
6324 CD3672    433         CMP TEMP1                ;REG IF REPEATED CHARACTER
6327 D002      434         BNE GODIN                ;
6329 A080      435         LDY #$80                 ;
632B 60        436  GODIN  RTS                      ;
632C           437  ;                              ;
632C           438  ;                              ;
632C           439  ;                              ;
632C           440  ;                              ;
632C 209960    441  SCREEN JSR TYPIN                ;GETS A NUMBER, AND TURNS
632F 2901      442         AND #$1                  ;ON THE PRIMARY SCREEN IF IT
6331 D004      443         BNE SCR.1                ;IS ODD; OTHERWISE, TURNS ON
6333 208460    444         JSR SECON                ;SECONDARY SCREEN
6336 60        445         RTS                      ;
6337 207B60    446  SCR.1  JSR PRION                ;
633A 60        447         RTS                      ;
633B           448  ;                              ;
633B           449  ;                              ;
633B           450  ;                              ;
633B           451  ;                              ;
633B 201C63    452  PAINT  JSR INCHEX               ;NO REPEATED CHARACTERS
633E C080      453         CPY #$80                 ;
6340 D001      454         BNE PG0                  ;
6342 60        455         RTS                      ;
6343 2903      456  PG0    AND #$3                  ;GET BRUSH VALUE
6345 8D3B72    457         STA BRUSH                ;WILL PAINT OR ERASE ANY COMB.
6348 A980      458         LDA #$80                 ;OF SCREEN PAGES
634A 8D4072    459         STA PFLAG                ;SET UP FLAG FOR PAINTING
634D 60        460         RTS                      ;
634E           461  ;                              ;
634E           462  ;                              ;
634E A900      463  ERASE  LDA #$0                  ;TURN OFF BRUSH FLAG
6350 8D4072    464         STA PFLAG                ;SETUP
6353 AD3D72    465         LDA BLTFLG               ;BIG BRUSH?
6356 F001      466         BEQ XORBLT               ;YES, FIX LAST CURSOR
6358 60        467         RTS                      ;
6359 A980      468  XORBLT LDA #$80                 ;RESET BIG BRUSH FLAG,
635B 8D3D72    469         STA BLTFLG               ;SO LITTLE BRUSH WON'T
635E 203B6B    470         JSR XORCUR               ;GET CONFUSED, AND REMOVE
6361 60        471         RTS                      ;LAST CURSOR
6362           472  ;                              ;
6362           473  ;                              ;
6362 A900      474  BLOTS  LDA #$0                  ;
6364 8D3D72    475         STA BLTFLG               ;SET FLAG FOR BIG BRUSH
6367 203B63    476         JSR PAINT                ;
636A 60        477         RTS                      ;GET BRUSH VALUE
636B           478  ;                              ;
636B           479  ;                              ;
636B           480  ;                              ;
636B AD3B72    481  DRAW   LDA BRUSH                ;
636E 20EC68    482         JSR STRUPL               ;LITTLE BRUSH DRAW ROUTINE
6371 60        483         RTS                      ;
6372           484  ;                              ;
6372           485  ;                              ;
6372           486  ;                              ;
```

```
6372            487  ;                              ;
6372            488  ;                              ;
6372            489  ;                              ;
6372 20386B     490  XFER    JSR XORCUR             ;REMOVE CURSOR
6375 A56F       491          LDA STRING             ;GET BASIC STRING STORAGE LOC.
6377 18         492          CLC                    ;
6378 69FE       493          ADC #$FE               ;START AT BOTTOM
637A 8506       494          STA ADDRL1             ;
637C A570       495          LDA STRING+1           ;$1FF LOCATIONS TO BE FILLED
637E 6901       496          ADC #$1                ;
6380 8507       497          STA ADDRH1             ;
6382 A9FF       498          LDA #$FF               ;$FF PAIRS OF POSITION VALUES
6384 8D6172     499          STA COUNT              ;
6387 201D68     500  MOVER   JSR BOP                ;GET XEXT
638A 8D3772     501          STA TEMP2              ;SAVE IT
638D 201D68     502          JSR BOP                ;GET YPOS
6390 8D3672     503          STA TEMP1              ;SAVE IT
6393 201D68     504          JSR BOP                ;GET XPOS
6396 AC3772     505          LDY TEMP2              ;GET XEXT
6399 20C263     506          JSR XCONV              ;NORMALIZE, AND REDUCE TO ONE
639C 20EE63     507          JSR STPUSH             ;BYTE, THEN STORE AS A STRING
639F 20E163     508          JSR YCONV              ;DO SAME TO Y POSITION
63A2 20EE63     509          JSR STPUSH             ;STORE AS NEXT INDEX OF STRING
63A5 EE6172     510          INC COUNT              ;BUMP COUNTER
63A8 CE6272     511          DEC ITCNT              ;DO THEM ALL
63AB D0DA       512          BNE MOVER              ;
63AD A9FF       513  FILL    LDA #$FF               ;FILL REMAINS OF BUFFER WITH $FF
63AF A200       514          LDX #$0                ;
63B1 20EE63     515          JSR STPUSH             ;
63B4 A9FF       516          LDA #$FF               ;
63B6 20EE63     517          JSR STPUSH             ;
63B9 EE6172     518          INC COUNT              ;
63BC D0EF       519          BNE FILL               ;
63BE 203362     520          JSR OUTSET             ;SET UP FOR RETURN TO BASIC
63C1 60         521          RTS                    ;
63C2            522  ;                              ;
63C2            523  ;                              ;
63C2            524  ;                              ;
63C2            525  ;                              ;
63C2            526  ;                              ;
63C2            527  ;                              ;
63C2 8D3772     528  XCONV   STA TEMP2              ;SET TXPOS ASIDE
63C5 98         529          TYA                    ;GET TXEXT
63C6 38         530          SEC                    ;
63C7 ED5E72     531          SBC XEXT               ;NORMALIZE TO SMALLEST
63CA A8         532          TAY                    ;VALUE OF X
63CB C000       533          CPY #$0                ;SAME XEXT ?
63CD F008       534          BEQ NORM               ;
63CF 18         535  CONLOP  CLC                    ;OH NO!!
63D0 6946       536          ADC #$46               ;SLOW DIVIDE
63D2 88         537          DEY                    ;
63D3 D0FA       538          BNE CONLOP             ;
63D5 F003       539          BEQ JMPIN              ;
63D7 AD3772     540  NORM    LDA TEMP2              ;
63DA 38         541  JMPIN   SEC                    ;SUBTRACT TXPOS, TO GET
63DB ED6072     542          SBC XPOS               ;ONE BYTE FOR X POSITION
63DE A200       543          LDX #$0                ;BASE PAGE OFFSET
63E0 60         544          RTS                    ;
63E1            545  ;                              ;
63E1            546  ;                              ;
63E1            547  ;                              ;
```

```
63E1            548  ;                          ;
63E1            549  ;                          ;
63E1            550  ;                          ;
63E1 A9C0       551  YCONV  LDA #$C0             ;INVERT Y COORDINATE SO THAT
63E3 38         552         SEC                 ;IT LOOX NORMAL IE: ORIGIN
63E4 ED3672     553         SBC TEMP1           ;AT LLHS., LARGE Y AT TOP
63E7 18         554         CLC                 ;
63E8 6D5F72     555         ADC YPOS            ;NO NORMALIZE WITH SMALLEST
63EB A200       556         LDX #$0             ;VALUE FOR TYPOS
63ED 60         557         RTS                 ;
63EE            558  ;                          ;
63EE            559  ;                          ;
63EE            560  ;                          ;;
63EE            561  ;                          ;
63EE            562  ;                          ;
63EE            563  ;                          ;
63EE 8106       564  STPUSH STA (ADDRL1,X ;STORE IT ON THE STAC*
63F0 B506       565         LDA ADDRL1,X        ;GET BASE-PAGE INDEXED
63F2 38         566         SEC                 ;LOW PART OF ADDRESS
63F3 E901       567         SBC #$1             ;DECREMENT IT BY ONE
63F5 9506       568         STA ADDRL1,X        ;
63F7 B002       569         BCS STOUT           ;CARRY CLEAR? BORROW
63F9 D607       570         DEC ADDRH1,X        ;FROM HIGH ADDRESS BYTE
63FB 60         571  STOUT  RTS                 ;
63FC            572  ;                          ;
63FC            573  ;                          ;
63FC            574  ;                          ;
63FC 209560     575  MSG    JSR LF              ;GET RID OF OLD STUFF
63FF 209560     576         JSR LF              ;
6402 209560     577         JSR LF              ;
6405 AD3F72     578         LDA PGFLAG          ;NO MESSAGES ON SECONDARY
6408 3001       579         BMI DOMSG           ;PAGE, BECAUSE THEY LOO*
640A 60         580         RTS                 ;LI*E GARBAGE ( * = CAY )
640B            581  ;                          ;
640B            582  ;                          ;
640B B1D6       583  DOMSG  LDA (MTAB),Y  ;GET MESSAGE TABLE POINTER
640D 85CE       584         STA MBUF            ;SET UP MESSAGE ADDRESSES
640F C8         585         INY                 ;
6410 B1D6       586         LDA (MTAB),Y  ;HIGH ORDER ADDRESS
6412 85CF       587         STA MBUF+1          ;
6414 C8         588         INY                 ;
6415 B1D6       589         LDA (MTAB),Y  ;GET LINE COUNT
6417 8D3672     590         STA TEMP1           ;
641A A000       591         LDY #$0             ;ZERO INDEX
641C B1CE       592  P.LINE LDA (MBUF),Y  ;GET  CHARACTER COUNT
641E 8D6172     593         STA COUNT           ;
6421 C8         594         INY                 ;
6422 B1CE       595  PRINT  LDA (MBUF),Y  ;GET A CHARACTER
6424 208D60     596         JSR CHAR            ;PRINT IT ON HALF SCREEN
6427 C8         597         INY                 ;BUMP INDEX
6428 CE6172     598         DEC COUNT           ;DECREMENT CHARACTER COUNT
642B D0F5       599         BNE PRINT           ;LINE DONE ?
642D CE3672     600         DEC TEMP1           ;DECREMENT LINE COUNT
6430 F006       601         BEQ MSOUT           ;YEP, LETS GET OUTTA HERE
6432 209560     602         JSR LF              ;NEW LINE
6435 4C1C64     603         JMP P.LINE          ;DO ANOTHER LINE
6438 207360     604  MSOUT  JSR HAFON           ;TURN ON TEXT WINDOW
643B 60         605         RTS                 ;
643C            606  ;                          ;
643C            607  ;                          ;
643C            608  ;                          ;
```

```
643C AD4572   609  PUTPOS LDA CMMND      ;GET COMMAND THAT WAS TYPED IN
643F 204E61   610        JSR COMAND     ;WHAT CURSOR MODE?
6442 8A       611        TXA            ;
6443 4A       612        LSR            ;SAME AS XDIV
6444 AA       613        TAX            ;
6445 207964   614        JSR GETX       ;GET X POSITION OF CURSOR
6448 AD3772   615        LDA TEMP2      ;
644B 9D5272   616        STA TXPOS,X    ;LOAD POSITION VALUES
644E AD3672   617        LDA TEMP1      ;
6451 9D4A72   618        STA TXEXT,X    ;
6454 C902     619        CMP #$2        ;WAIT FOR HARDWARE TO SETTLE
6456 100D     620        BPL LATE       ;DOWN
6458 C901     621        CMP #$1        ;
645A F006     622        BEQ LATER      ;
645C 207564   623        JSR TIMER      ;
645F 207564   624        JSR TIMER      ;
6462 207564   625  LATER  JSR TIMER      ;
6465 207564   626  LATE   JSR TIMER      ;
6468 20BA64   627        JSR GETY       ;GET Y POSITION - ONE BYTE
646B AD3772   628        LDA TEMP2      ;
646E 9D4E72   629        STA TYPOS,X    ;SAVE Y POSITION VALUE
6471 8A       630        TXA            ;
6472 0A       631        ASL            ;SAME AS XMUL
6473 AA       632        TAX            ;
6474 60       633        RTS            ;
6475 C8       634  TIMER  INY            ;TIMEOUT ROUTINE
6476 D0FD     635        BNE TIMER      ;
6478 60       636        RTS            ;
6479          637  ;                    ;
6479          638  ;                    ;
6479 A000     639  GETX   LDY #$0        ;CURSOR ADDRESS OFFSET ON BASE PG
647B 98       640        TYA            ;SET UP COUNT LOCATIONS
647C 38       641        SEC            ;SO THAT EDGE OF SCREEN CAN
647D E908     642        SBC #$8        ;BE ACCESSED
647F 8D3772   643        STA TEMP2      ;
6482 A9FF     644        LDA #$FF       ;
6484 8D3672   645        STA TEMP1      ;
6487 20B460   646        JSR GETPOS     ;PUT A NUMBER IN TEMP1 AND TEMP2
648A A900     647        LDA #$0        ;ZERO TEMP1
648C AC3672   648        LDY TEMP1      ;SAVE OVERFLOW OF COUNT
648F 8D3672   649        STA TEMP1      ;
6492 AD3772   650        LDA TEMP2      ;PUT LOW BYTE IN ACCUMULATOR
6495 20DF64   651        JSR DVDE       ;COMPUTE MODULO $45
6498 C000     652        CPY #$0        ;
649A F00C     653        BEQ HURRY      ;
649C 18       654        CLC            ;
649D 6980     655        ADC #$80       ;TEMP1, FROM GETPOS WAS >0
649F 20DF64   656        JSR DVDE       ;HAVE TO DIVIDE INTO $80
64A2 18       657        CLC            ;TWICE TO GET FULL X POSITION
64A3 6980     658        ADC #$80       ;
64A5 20DF64   659        JSR DVDE       ;
64A8 AC3672   660  HURRY  LDY TEMP1      ;CHECK FOR OUT OF BOUNDS
64AB C004     661        CPY #$4        ;
64AD 3007     662        BMI FINE       ;
64AF A945     663        LDA #$45       ;SET TXEXT,TXPOS TO MAX
64B1 A003     664        LDY #$3        ;VALUES
64B3 8C3672   665        STY TEMP1      ;
64B6 8D3772   666  FINE   STA TEMP2      ;CONVERTED X VALUES IN TEMP1
64B9 60       667        RTS            ;AND TEMP2
64BA          668  ;                    ;
64BA          669  ;                    ;
```

```
648A            670  ;                              ;
648A A001       671  GETY   LDY #$1                 ;Y JOYSTIX OFFSET TO BASE PG
648C A900       672         LDA #$0                 ;
648E 38         673         SEC                     ;SET COUNT LOCATIONS
648F E905       674         SBC #$5                 ;FOR BEST CURSOR SWEEP
64C1 8D3772     675         STA TEMP2               ;
64C4 A9FF       676         LDA #$FF                ;
64C6 8D3672     677         STA TEMP1               ;
64C9 208460     678         JSR GETPOS              ;GET A Y-JOYSTIX NUMBER
64CC AD3672     679         LDA TEMP1               ;
64CF D008       680         BNE ENDSTP              ;IF HIGH BYTE >0 THEN CURSOR
64D1 AD3772     681         LDA TEMP2               ;IS AT THE EDGE
64D4 4A         682         LSR                     ;
64D5 C960       683         CMP #$60                ;TEST FOR LOW BYTE TO LARGE
64D7 3005       684         BMI OK                  ;
64D9 A9C0       685  ENDSTP LDA #$C0                ;LARGEST Y POSITION VALUE
64DB 8D3772     686         STA TEMP2               ;
64DE 60         687  OK     RTS                     ;RETURN WITH TYPOS IN TEMP2
64DF            688  ;                              ;
64DF            689  ;                              ;
64DF            690  ;                              ;
64DF            691  ;                              ;
64DF C900       692  DVDE   CMP #$0                 ;DOES MODULO $46 CONVERSION
64E1 3004       693         BMI SUBTR               ;TO SET X POSITION VALUES
64E3 C946       694  PSTV   CMP #$46                ;TXEXT, TXPOS IN THE RANGE
64E5 3008       695         BMI SML                 ;0,0 TO 3,45  (HEX)
64E7 38         696  SUBTR  SEC                     ;
64E8 E946       697         SBC #$46                ;
64EA EE3672     698         INC TEMP1               ;
64ED D0F4       699         BNE PSTV                ;
64EF 60         700  SML    RTS                     ;
64F0            701  ;                              ;
64F0            702  ;                              ;
64F0            703  ;                              ;
64F0            704  ;                              ;
64F0            705  ;                              ;
64F0 A200       706  NEWPOS LDX #$0                 ;UNUSED ROUTINE, EXCEPT FOR
64F2 A902       707         LDA #$2                 ;TEST--PLACES CURSOR NEAR THE
64F4 9D4A72     708         STA TXEXT,X             ;CENTRE OF SCREEN
64F7 A967       709         LDA #$67                ;
64F9 9D4E72     710         STA TYPOS,X             ;
64FC A908       711         LDA #$8                 ;
64FE 9D5272     712         STA TXPOS,X             ;
6501 60         713         RTS                     ;
6502            714  ;                              ;
6502            715  ;                              ;;
6502            716  ;                              ;
6502            717  ;                              ;
6502            718  ;                              ;
6502 A002       719  TRANS  LDY #$2                 ;TRANSFERS 0-INDEX ADDRESSES TO
6504 A506       720         LDA ADDRL1              ;2-INDEX LOCATIONS, SAVES
6506 990600     721         STA ADDRL1,Y            ;TIME FROM CALCULATION
6509 A507       722         LDA ADDRH1              ;
650B 990700     723         STA ADDRH1,Y            ;
650E AD5672     724         LDA MASK                ;
6511 995672     725         STA MASK,Y              ;
6514 60         726         RTS                     ;
6515            727  ;                              ;
6515            728  ;                              ;
6515            729  ;                              ;
6515            730  ;                              ;
```

```
6515            731  ;                              ;EDGE DETECTION ROUTINE
6515 202D68     732  DOLOOP JSR PSAVE               ;SAVE CURRENT PIXEL ADDRESS
6518 AD3E72     733         LDA CFLAG               ;IS BUFFER FULL
651B 1003       734         BPL GGO                 ;
651D 4CA865     735         JMP FINISH              ;NO, DO IT
6520 A900       736  GGO    LDA #$0                 ;
6522 8D3172     737         STA DVD                 ;SET UP DATA BASE
6525 8D0F72     738         STA FAULTS              ;NO MISSES YET
6528 A9F0       739         LDA #$F0                ;INITIAL WINDOW IS LARGE
652A 8D0B72     740         STA WIND01              ;
652D A97F       741         LDA #$7F                ;
652F 8D0C72     742         STA WIND02              ;
6532 20F965     743  NEWPIE JSR GETDIR             ;GET DIRECTION, USING HALF
6535 20AC65     744         JSR GETRAY              ;QUADRANT WORD (QUID)
6538 8C1C72     745         STY RAYOFF,             ;GET OFFSET TO LINE TABLE
653B 205668     746  PIESEC JSR PEEK                ;RESTORE ADDRESS
653E AD1772     747         LDA QUID                ;HALF QUADRANT
6541 AC1C72     748         LDY RAYOFF              ;LINE NUMBER IN SEGMENT
6544 300A       749         BMI NEXSEC              ;<0 ? THEN DO NEXT SEGMENT
6546 C008       750         CPY #$8                 ;>7 ? THEN DO NEXT SEGMENT
6548 D014       751         BNE DORAY               ;ALL SET, DO-RAY DO-RAY
654A CE1C72     752         DEC RAYOFF              ;
654D CE1C72     753         DEC RAYOFF              ;
6550 18         754  NEXSEC CLC                    ;GET DATA BASE FOR NEXT SEG.
6551 6901       755         ADC #$1                 ;
6553 2907       756         AND #$7                 ;
6555 8D1772     757         STA QUID                ;
6558 EE1C72     758         INC RAYOFF              ;
655B 4C3265     759         JMP NEWPIE              ;ONCE MORE, WITH PASSION
655E 18         760  DORAY  CLC                    ;
655F 6901       761         ADC #$1                 ;TEST FOR EVEN OR ODD SEG.
6561 4A         762         LSR                     ;
6562 2901       763         AND #$1                 ;
6564 F00E       764         BEQ YXORD               ;ODD SEGMENT, LOAD STEPS
6566 B90870     765         LDA XSTAIR,Y            ;
6569 8D1D72     766         STA XSTEP               ;
656C B91070     767         LDA YSTAIR,Y            ;
656F 8D1F72     768         STA YSTEP               ;
6572 D00C       769         BNE QBUMP               ;SAME AS A JUMP
6574 B91070     770  YXORD  LDA YSTAIR,Y           ;EVEN SEGMENT, LOAD STEPS
6577 8D1D72     771         STA XSTEP               ;
657A B90870     772         LDA XSTAIR,Y            ;
657D 8D1F72     773         STA YSTEP               ;
6580 20D965     774  QBUMP  JSR ZSMALL             ;SET UP FOR START OF LINE
6583 B92070     775         LDA FSTEP,Y             ;GET HYPOTENUSE BYTES
6586 8D1172     776         STA LOSTEP              ;TO MEASURE DISTANCE WITH
6589 B91870     777         LDA ISTEP,Y             ;
658C 8D1272     778         STA HISTEP              ;
658F AD1772     779         LDA QUID                ;
6592 2901       780         AND #$1                 ;WHICH WAY IS RAYOFF GOING?
6594 F004       781         BEQ RAYDEC              ;DECREMENTING FOR EVEN QUID
6596 C8         782         INY                     ;INCREMENTING FOR ODD QUID
6597 4C9B65     783         JMP RAYGO               ;
659A 88         784  RAYDEC DEY                    ;
659B 8C1C72     785  RAYGO  STY RAYOFF             ;
659E 200B66     786         JSR CM                  ;DO LINE, AND CENTRE OF MASS
65A1 200B65     787         JSR QTEST               ;TEST IF ALL SEGMENTS ARE DONE
65A4 C080       788         CPY #$80                ;
65A6 D093       789         BNE PIESEC              ;NO, DO NEXT SEGMENT
65A8 206568     790  FINISH JSR PRESTO              ;POP ADDRESSES
65AB 60         791         RTS                     ;
```

```
65AC            792  ;                              ;
65AC            793  ;                              ;
65AC A007       794  GETRAY LDY #$7                  ;GET FIRST POINTER TO LINE
65AE AD1772     795        LDA QUID                 ;TABLE, 7 FOR EVEN QUID,
65B1 2901       796        AND #$1                  ;0 FOR ODD QUID
65B3 F002       797        BEQ GETOUT               ;
65B5 A000       798        LDY #$0                  ;
65B7 60         799  GETOUT RTS                     ;
65B8            800  ;                              ;
65B8            801  ;                              ;
65B8            802  ;                              ;
65B8            803  ;                              ;
65B8            804  ;                              ;
65B8 AD1772     805  QTEST  LDA QUID                 ;CHEX FOR END OF SEGMENT
65BB 6A         806        ROR                      ;IF QUID EVEN, AND RAYOFF <0
65BC B007       807        BCS ODDSEG               ;
65BE AC1C72     808        LDY RAYOFF               ;
65C1 3009       809        BMI YLOAD                ;
65C3 1009       810        BPL QCON                 ;
65C5 AC1C72     811  ODDSEG LDY RAYOFF              ;OR IF QUID ODD, AND RAYOFF >7
65C8 C008       812        CPY #$8                  ;
65CA D002       813        BNE QCON                 ;THEN END OF SEGMENT
65CC A080       814  YLOAD  LDY #$80                ;EXIT WITH Y=80, SEGMENT NOT ENDED
65CE AD1772     815  QCON   LDA QUID                 ;TEST IF ALL SEGMENTS DONE
65D1 CD1872     816        CMP LQUID                ;
65D4 F002       817        BEQ BLY                  ;
65D6 A000       818        LDY #$0                  ;Y=0 IF ALL SEGMENTS DONE
65D8 60         819  BLY    RTS                      ;
65D9            820  ;                              ;
65D9            821  ;                              ;
65D9            822  ;                              ;
65D9            823  ;                              ;
65D9 AD1D72     824  ZSMALL LDA XSTEP                ;CHECK STEP SIZES, SETS SMALL
65DC CD1F72     825        CMP YSTEP                ;STEP TO ZERO FOR START, SO
65DF 300C       826        BMI XZERO                ;THAT ALL LINES START THE
65E1 A900       827  YZERO  LDA #$0                  ;SAME, SETS UP TEMPORARY STEP
65E3 8D2072     828        STA YFRACT               ;VALUES, WHICH UPDATE STEP COUNT
65E6 AD1D72     829        LDA XSTEP                ;
65E9 8D1E72     830        STA XFRACT               ;
65EC 60         831        RTS                      ;
65ED A900       832  XZERO  LDA #$0                  ;
65EF 8D1E72     833        STA XFRACT               ;
65F2 AD1F72     834        LDA YSTEP                ;
65F5 8D2072     835        STA YFRACT               ;
65F8 60         836        RTS                      ;
65F9            837  ;                              ;
65F9            838  ;                              ;
65F9            839  ;                              ;
65F9            840  ;                              ;
65F9 AD1772     841  GETDIR LDA QUID                 ;USES QUID TO GET AN OFFSET
65FC 4A         842        LSR                      ;ON THE DIRECTION TABLE
65FD A8         843        TAY                      ;THEN GETS X AND Y DIRECTIONS
65FE B90070     844        LDA XVECT,Y              ;FOR LINE
6601 8D1972     845        STA XDIR                 ;
6604 B90470     846        LDA YVECT,Y              ;
6607 8D1A72     847        STA YDIR                 ;
660A 60         848        RTS                      ;
660B            849  ;                              ;
660B            850  ;                              ;
660B            851  ;                              ;
660B            852  ;                              ;
```

```
6600           853  ;                          ;DOES LINE AND CALCULATES C.M.
6600 A900      854  CM      LDA #$0            ;BY DRAWING A LINE, INTEGRATING
6602 8D0972    855          STA NUMLO          ;THE LUMINANCE ALONG LINE
6605 8D0872    856          STA NUMHI          ;ALSO ACCUMALATES A SUM OF
6608 8D0772    857          STA MASS           ;LUMINANCE*DISTANCE
660B 8D1372    858          STA OFRACT         ;MAINTAINS MEASUREMENT OVER WINDOW
660E 8D0E72    859          STA REFUND         ;BOUNDS, AND CALCULATES EXPECTED
6611 AD0B72    860          LDA WIND01         ;LOCATION OF MAX. LUMINOUS DENSITY
6614 8D1472    861          STA OFFSET         ;BY DIVIDING SUM OF LUMINANCE
6617 AD0C72    862          LDA WIND02         ;INTO THE SUM OF PRODUCTS TERM
661A 8D1572    863          STA ENDPT          ;*** ZERO EVERYTHING FIRST ***
661D 204367    864  POINTS  JSR RAYPNT         ;DO A STEP
6620 C080      865          CPY #$80           ;HIT THE EDGE OF THE SCREEN ?
6622 F02D      866          BEQ CMFIND         ;DON'T BEAT YOUR HEAD AGAINST WALL
6624 203867    867          JSR GETDIS         ;CALCULATE DISTANCE - ACCUARTE TO
6627 30F4      868          BMI POINTS         ;LAST STEP, NEGATIVE OUT OF WINDOW
6629 20BE68    869          JSR GETVAL         ;GET LUMINANCE VALUE, 0 - 3
662C 8D3672    870          STA TEMP1          ;SAVE
662F A8        871          TAY                ;MULTIPLICAND IN Y REGISTER
6630 18        872          CLC                ;
6631 6D0772    873          ADC MASS           ;SUM LUMINANCE
6634 8D0772    874          STA MASS           ;INTEGRATION OVER WINDOW OF PIXELS
6637 203867    875          JSR GETDIS         ;GET POSITIVE DISTANCE FROM START
663A 20AD6C    876          JSR MULT           ;OF WINDOW AREA; MUL WITH PIXEL
663D 18        877          CLC                ;TWO BYTE SUM OF PRODUCTS
663E 6D0972    878          ADC NUMLO          ;ADD TO LOW BYTE
6641 8D0972    879          STA NUMLO          ;
6644 9003      880          BCC NIXCAR         ;CARRY SET -- INCREMENT HIGH BYTE
6646 EE0872    881          INC NUMHI          ;
6649 AD1072    882  NIXCAR  LDA DISTAN         ;
664C CD1572    883          CMP ENDPT          ;END OF WINDOW ?
664F 30CC      884          BMI POINTS         ;NO, ADVANCE ALONG RAY
6651 AD0772    885  CMFIND  LDA MASS           ;IS SUM OF LUMINANCE (MASS)
6654 D03C      886          BNE NONZER         ;ZERO?, BECAUSE IF IT IS, THERE
6656 AD3172    887          LDA DVD            ;IS A HOLE IN CELL
6659 D023      888          BNE TSET           ;USE LAST POSITION ONCE ONLY
665B EE0F72    889          INC FAULTS         ;BUMP THE MISS COUNTER
665E D005      890          BNE DOWAX          ;HAS IT GOTTEN REALLY BIG?
6660 A9FF      891          LDA #$FF           ;DON'T LET IT ROLL OVER
6662 8D0F72    892          STA FAULTS         ;
6665 AD0A72    893  DOWAX   LDA WIDTH         ;RE-CALCULATE WINDOW PLACEMENT
6668 18        894          CLC                ;SO THAT IT IS TWICE AS BIG
6669 6D0B72    895          ADC WIND01         ;
666C 3002      896          BMI NEGWIN         ;
666E A900      897          LDA #$0            ;START OF WINDOW CAN'T BE VIRTUAL
6670 8D0B72    898  NEGWIN  STA WIND01         ;
6673 AD0A72    899          LDA WIDTH          ;PUSH OTHER END OF WINDOW OUT
6676 18        900          CLC                ;
6677 6D0C72    901          ADC WIND02         ;
667A 8D0C72    902          STA WIND02         ;
667D 60        903          RTS                ;THAT'S IT FOR THIS LINE
667E 8D0D72    904  TSET    STA TRAVEL         ;PATCH THIS LINE, SINCE ITS
6681 18        905          CLC                ;THE FIRST MISS IN AWHILE
6682 6D0B72    906          ADC WIND01         ;
6685 38        907          SEC                ;FIX WINDOW SO THAT IT DOESN'T
6686 E908      908          SBC #$8            ;GET OBLITERATED LATER
6688 8D0B72    909          STA WIND01         ;
668B A900      910          LDA #$0            ;ZERO DVD, NEXT MISS IS A MISS
668D 8D3172    911          STA DVD            ;
6690 F00C      912          BEQ SETUP          ;SCIP C.M CALCULATION,ITS ALREADY
6692           913  ;                          ;BEEN DONE
```

```
669D AC0972  914  NONZER LDY NUMLO      ;
66A0 20E366  915         JSR BIGDIV     ;DO DIVIDE TO GET EXPECTED LOCATN
66A3 AD3172  916         LDA DVD        ;OF MAXIMUM LUMINOUS DENSITY
66A6 8D0D72  917         STA TRAVEL     ;TRAVEL HOLDS DISTANCE FROM START
66A9 AD0B72  918  SETUP  LDA WIND01     ;OF WINDOW WHERE LUM. DENSITY IS
66AC 8D1472  919         STA OFFSET     ;GET WINDOW DISTANCE FRO CURSOR
66AF A900    920         LDA #$0        ;
66B1 8D0E72  921         STA REFUND     ;REFUND IS APPROX. HYPOTENUSE
66B4 205668  922         JSR PEEK       ;RESTORE CURSOR ADDRESS
66B7 20D965  923         JSR ZSMALL     ;GET LINE PARAMETERS
66BA 204367  924  GETLOC JSR RAYPNT     ;
66BD 203867  925         JSR GETDIS     ;MOVE ALONG LINE UNTIL WINDOW
66C0 30F8    926         BMI GETLOC     ;IS HIT, THEN COUNT TRAVEL
66C2 CD0D72  927         CMP TRAVEL     ;IS DISTANCE ALONG LINE = TRAVEL
66C5 30F3    928         BMI GETLOC ,   ;NO, DO SOME MORE
66C7 206867  929  DOTTER JSR SAVDOT     ;ADDRESS IS WALL ADDRESS
66CA AD0D72  930         LDA TRAVEL     ;
66CD 38      931         SEC            ;RECALCULATE WINDOW
66CE ED0B72  932         SBC WIND01     ;SO IT IS CENTRED ABOUT WALL
66D1 8D3672  933         STA TEMP1      ;
66D4 A920    934         LDA #$20       ;
66D6 8D0C72  935         STA WIND02     ;
66D9 A908    936         LDA #$8        ;
66DB 38      937         SEC            ;
66DC ED3672  938         SBC TEMP1      ;
66DF 8D0B72  939         STA WIND01     ;
66E2 60      940         RTS            ;
66E3 8D2E72  941  BIGDIV STA DVSR       ;DOES A TWO BYTE DIVIDE
66E6 20CF6C  942         JSR DIVTWO     ;A=DIVISOR, Y=DIVIDEND
66E9 8D3072  943         STA REM        ;SAVE REMAINDER
66EC AC0872  944         LDY NUMHI      ;
66EF F029    945         BEQ DIVEND     ;IS HIGH BYTE = 0
66F1 AD3172  946         LDA DVD        ;SAVE DIVISOR FROM LOW
66F4 8D2F72  947         STA TEMDVD     ;DIVIDE
66F7 201E67  948  DIVLOP JSR PARDIV     ;DIVIDE INTO HIGH BYTE=1
66FA CE0872  949         DEC NUMHI      ;
66FD D0F8    950         BNE DIVLOP     ;
66FF AD2E72  951         LDA DVSR       ;
6702 AC3072  952         LDY REM        ;
6705 CC2E72  953         CPY DVSR       ;REMAINDER LARGER THAN DIVISOR?
6708 3010    954         BMI DIVEND     ;
670A 20CF6C  955         JSR DIVTWO     ;YES, DIVIDE THEM TOO
670D 8D3072  956         STA REM        ;REMAINDER OF THIS DIVIDE A FRACT.
6710 AD3172  957         LDA DVD        ;SUM DIVIDES
6713 18      958         CLC            ;
6714 6D2F72  959         ADC TEMDVD     ;
6717 8D3172  960         STA DVD        ;SAVE IN DVD
671A AD3072  961  DIVEND LDA REM        ;MAINTAIN ESTABLISHED CONVENTION
671D 60      962         RTS            ;
671E         963  ;                     ;
671E A0FF    964  PARDIV LDY #$FF        ;DIVIDES DIVISOR  INTO CARRY
6720 AD2E72  965         LDA DVSR       ;
6723 20CF6C  966         JSR DIVTWO     ;NOT FAST, BUT IT WORX
6726 38      967         SEC            ;
6727 6D3072  968         ADC REM        ;
672A 8D3072  969         STA REM        ;
672D 18      970         CLC            ;
672E 6D2F72  971         ADC TEMDVD     ;
6731 8D2F72  972         STA TEMDVD     ;
6734 8D3172  973         STA DVD        ;
6737 60      974         RTS            ;
```

```
6738          975  ;                         ;
6738          976  ;                         ;
6738          977  ;                         ;
6738          978  ;                         ;
6738          979  ;                         ;
6738 AD1472   980  GETDIS LDA OFFSET          ;CALCULATE APPROXIMATE  ISTANCE
673B 18       981        CLC                 ;
673C 6D0E72   982        ADC REFUND          ;REFUND IS APPROXIMATION TO
673F 8D1072   983        STA DISTAN          ;DISTANCE ALONG HYPOTENUSE
6742 60       984        RTS                 ;
6743          985  ;                         ;
6743          986  ;                         ;
6743          987  ;                         ;
6743 AD1E72   988  RAYPNT LDA XFRACT         ;MOVES ALONG THE LINE
6746 F02E     989        BEQ YFIRST          ;WHEN YSTEP IS DONE, XFRACT
6748 AD1972   990        LDA XDIR            ;IS ZERO
674B 20D069   991        JSR NEXTX           ;SINGLE STEP IN X DIRECTION
674E C080     992        CPY #$80            ;HIT EDGE OF SCREEN?
6750 F01D     993        BEQ RAYFIN          ;
6752 CE1E72   994        DEC XFRACT          ;DECREMENT STEP COUNTER
6755 D018     995        BNE RAYFIN          ;STILL SOME TO GO
6757 AD1F72   996        LDA YSTEP           ;CHANGE DATA BASE, FOR Y STEPS
675A 8D2072   997        STA YFRACT          ;
675D AD1972   998        LDA XDIR            ;
6760 20D069   999        JSR NEXTX           ;NO CORNERS TO THESE STEP LINES
6763 C080    1000        CPY #$80            ;HIT EDGE OF SCREEN?
6765 F008    1001        BEQ RAYFIN          ;
6767 AD1F72  1002        LDA YSTEP           ;NEW HYPOTENUSE VALUE?
676A CD1D72  1003        CMP XSTEP           ;REAL DISTANCE UPDATED EVERY TWO
676D 102E    1004        BPL OFFINC          ;CORNERS
676F EE0E72  1005  RAYFIN INC REFUND         ;APPROXIMATE DISTANCE ALONG HYPOT.
6772 AD1472  1006        LDA OFFSET          ;DISTANCE FROM CURSOR TO LAST
6775 60      1007        RTS                 ;HYPOTENUSE CALCULATION
6776 AD1A72  1008  YFIRST LDA YDIR           ;
6779 20546A  1009        JSR NEXTY           ;DO Y STEP
677C C080    1010        CPY #$80            ;
677E F0EF    1011        BEQ RAYFIN          ;
6780 CE2072  1012        DEC YFRACT          ;DECREMENT Y STEP COUNTER
6783 D0EA    1013        BNE RAYFIN          ;
6785 AD1D72  1014        LDA XSTEP           ;
6788 8D1E72  1015        STA XFRACT          ;
678B AD1A72  1016        LDA YDIR            ;
678E 20546A  1017        JSR NEXTY           ;
6791 C080    1018        CPY #$80            ;
6793 F00A    1019        BEQ RAYFIN          ;
6795 AD1D72  1020        LDA XSTEP           ;
6798 CD1F72  1021        CMP YSTEP           ;
679B 30D2    1022        BMI RAYFIN          ;
679D AD1172  1023  OFFINC LDA LOSTEP         ;ADD UP DISTANCE
67A0 18      1024        CLC                 ;USING TWO BYTE SUM, LOW
67A1 6D1372  1025        ADC OFRACT          ;BYTE IS A FRACTION,
67A4 8D1372  1026        STA OFRACT          ;HYPOTENUSE IS GIVEN AS A
67A7 AD1272  1027        LDA HISTEP          ;TWO BYTE, FRACTION AND
67AA 6D1472  1028        ADC OFFSET          ;INTEGER QUANTITY
67AD 8D1472  1029        STA OFFSET          ;ONLY INTEGER PART OF
67B0 A9FF    1030        LDA #$FF            ;DISTANCE IS USED IN C.M.
67B2 8D0E72  1031        STA REFUND          ;CALCULATION
67B5 4C6F67  1032        JMP RAYFIN          ;
67B8         1033  ;                         ;
67B8         1034  ;                         ;;
67B8         1035  ;                         ;
```

```
67B8 20376C 1036  SAVDOT JSR XDIV       ;INDEX'S FOR POSITION VALUES
67BB BD5272 1037         LDA TXPOS,X    ;PUSH POSITION VALUES
67BE 201268 1038         JSR BUSH       ;ON THE CELL STAX
67C1 BD4E72 1039         LDA TYPOS,X    ;
67C4 201268 1040         JSR BUSH       ;
67C7 BD4A72 1041         LDA TXEXT,X    ;
67CA 201268 1042         JSR BUSH       ;
67CD EE6272 1043         INC ITCNT      ;BUMP DOT COUNT
67D0 D005   1044         BNE SVOUT      ;
67D2 A080   1045         LDY #$80       ;BUFFER FULL? HOLDS 256 SETS
67D4 8C3E72 1046         STY CFLAG      ;
67D7 BD4A72 1047  SVOUT  LDA TXEXT,X    ;FIND THE SMALLEST X AND Y
67DA CD5E72 1048         CMP XEXT       ;POSITIONS FOR NORMALIZING
67DD F00E   1049         BEQ XBYTE      ;
67DF 1017   1050         BPL YBYTE      ;
67E1 8D5E72 1051         STA XEXT       ;
67E4 BD5272 1052         LDA TXPOS,X    ;
67E7 8D6072 1053         STA XPOS       ;
67EA 4CF867 1054         JMP YBYTE      ;
67ED BD5272 1055  XBYTE  LDA TXPOS,X    ;
67F0 CD6072 1056         CMP XPOS       ;STILL LOO*ING
67F3 1003   1057         BPL YBYTE      ;
67F5 8D6072 1058         STA XPOS       ;
67F8 AD5F72 1059  YBYTE  LDA YPOS       ;
67FB 4A     1060         LSR            ;
67FC 8D3972 1061         STA TEMP4      ;
67FF BD4E72 1062         LDA TYPOS,X    ;
6802 4A     1063         LSR            ;
6803 CD3972 1064         CMP TEMP4      ;
6806 1006   1065         BPL COMP       ;
6808 BD4E72 1066         LDA TYPOS,X    ;
680B 8D5F72 1067         STA YPOS       ;
680E 203B6C 1068  COMP   JSR XMUL       ;RESTORE NORMAL INDEX
6811 60     1069         RTS            ;
6812        1070  ;                      ;
6812        1071  ;                      ;
6812        1072  ;                      ;
6812        1073  ;                      ;
6812 A000   1074  BUSH   LDY #$0        ;PUSH ADDRESS ON POSITION BUFFER
6814 911B   1075         STA (BUFF),Y   ;
6816 E61B   1076         INC BUFF       ;
6818 D002   1077         BNE BSOUT      ;
681A E61C   1078         INC BUFF+1     ;
681C 60     1079  BSOUT  RTS            ;
681D        1080  ;                      ;
681D        1081  ;                      ;
681D        1082  ;                      ;
681D        1083  ;                      ;
681D A000   1084  BOP    LDY #$00       ;
681F A51B   1085         LDA BUFF       ;POP ADDRESS OFF POSITION BUFFER
6821 38     1086         SEC            ;
6822 E901   1087         SBC #$1        ;COMPLEMENT OF PUSH ROUTINE
6824 851B   1088         STA BUFF       ;
6826 B002   1089         BCS BGET       ;
6828 C61C   1090         DEC BUFF+1     ;
682A B11B   1091  BGET   LDA (BUFF),Y   ;
682C 60     1092         RTS            ;
682D        1093  ;                      ;
682D        1094  ;                      ;
682D        1095  ;                      ;
682D        1096  ;                      ;
```

```
682D 20376C 1097  PSAVE   JSR XDIV       ;SAVE ROUTINE FOR CURSOR ADDRESS,
6830 BD4A72 1098          LDA TXEXT,X    ;PUSHES EVERYTHING ON CURSOR STAX
6833 208E68 1099          JSR PUSH       ;
6836 BD4E72 1100          LDA TYPOS,X    ;
6839 208E68 1101          JSR PUSH       ;
683C BD5272 1102          LDA TXPOS,X    ;
683F 208E68 1103          JSR PUSH       ;
6842 203B6C 1104          JSR XMUL       ;
6845 B507   1105          LDA ADDRH1,X   ;
6847 208E68 1106          JSR PUSH       ;
684A B506   1107          LDA ADDRL1,X   ;
684C 208E68 1108          JSR PUSH       ;
684F BD5672 1109          LDA MASK,X     ;
6852 208E68 1110          JSR PUSH       ;
6855 60     1111          RTS            ;
6856        1112  ;                      ;
6856        1113  ;                      ;
6856 206568 1114  PEEK    JSR PRESTO     ;RESTORES ADDRESSES, BUT DOESN'T
6859 A006   1115          LDY #$6        ;AFFECT STA* POINTER
685B E619   1116  INCLOP  INC STAX       ;
685D D002   1117          BNE DECY       ;CAN BE CALLED ANY NUMBER OF TIMES
685F E61A   1118          INC STAX+1     ;
6861 88     1119  DECY    DEY            ;
6862 D0F7   1120          BNE INCLOP     ;
6864 60     1121          RTS            ;
6865        1122  ;                      ;
6865        1123  ;                      ;
6865 209968 1124  PRESTO  JSR POP        ;POPS THE ADDRESSES, AND POSITION
6868 9D5672 1125          STA MASK,X     ;WORDS OFF OF THE STA*
686B 209968 1126          JSR POP        ;
686E 9506   1127          STA ADDRL1,X   ;
6870 209968 1128          JSR POP        ;CHANGES STA* POINTER
6873 9507   1129          STA ADDRH1,X   ;
6875 20376C 1130          JSR XDIV       ;
6878 209968 1131          JSR POP        ;
687B 9D5272 1132          STA TXPOS,X    ;
687E 209968 1133          JSR POP        ;
6881 9D4E72 1134          STA TYPOS,X    ;
6884 209968 1135          JSR POP        ;
6887 9D4A72 1136          STA TXEXT,X    ;
688A 203B6C 1137          JSR XMUL       ;
688D 60     1138          RTS            ;;
688E        1139  ;                      ;
688E        1140  ;                      ;
688E A000   1141  PUSH    LDY #$0        ;STORE, AND MODIFY POINTER
6890 9119   1142          STA (STAX),Y   ;
6892 E619   1143          INC STAX       ;
6894 D002   1144          BNE PSOUT      ;
6896 E61A   1145          INC STAX+1     ;
6898 60     1146  PSOUT   RTS            ;
6899        1147  ;                      ;
6899        1148  ;                      ;
6899 A000   1149  POP     LDY #$0        ;;
689B A519   1150          LDA STAX       ;RETRIEVE, AND RESET POINTER
689D 38     1151          SEC            ;
689E E901   1152          SBC #$1        ;
68A0 8519   1153          STA STAX       ;;
68A2 B002   1154          BCS PGET       ;
68A4 E61A   1155          INC STAX+1     ;
68A6 B119   1156  PGET    LDA (STAX),Y   ;
68A8 60     1157          RTS            ;
```

```
68A9          1158  ;                          ;
68A9          1159  ;                          ;
68A9          1160  ;                          ;
68A9          1161  ;                          ;
68A9          1162  ;                          ;
68A9          1163  ;                          ;
68A9          1164  ;                          ;
68A9          1165  ;                          ;
68A9 A001     1166  PTRANS LDY #$1             ;TRANSFERS POSITION WORDS
68AB AD4A72   1167         LDA TXEXT           ;FROM 0 INDEX LOCATIONS
68AE 994A72   1168         STA TXEXT,Y         ;TO 1 INDEX LOCATIONS,
68B1 AD4E72   1169         LDA TYPOS           ;SAVES CALCULATIONS TIME
68B4 994E72   1170         STA TYPOS,Y         ;
68B7 AD5272   1171         LDA TXPOS           ;
68BA 995272   1172         STA TXPOS,Y         ;
68BD 60       1173         RTS                 ;
68BE          1174  ;                          ;
68BE          1175  ;                          ;GETVAL  READS THE TWO GRAPHICS
68BE          1176  ;                          ;SCREENS AND FINDS THE PIXEL
68BE          1177  ;                          ;INTENSITY VALUE
68BE A900     1178  GETVAL LDA #$0             ;ZERO INTENSITY
68C0 8D3672   1179         STA TEMP1           ;
68C3 B507     1180         LDA ADDRH1,X        ;GET HIGH PART OF PIXEL ADDRESS
68C5 8D3772   1181         STA TEMP2           ;SAVE IT
68C8 20E168   1182         JSR READ            ;READ PRIMARY PAGE
68CB 0E3672   1183         ASL TEMP1           ;SHIFT PIXEL VALUE, WEIGHT OF 2
68CE AD3772   1184         LDA TEMP2           ;
68D1 4960     1185         XOR #$60            ;SECONDARY PAGE ADDRESS
68D3 9507     1186         STA ADDRH1,X        ;
68D5 20E168   1187         JSR READ            ;READ SECONDARY PAGE
68D8 AD3772   1188         LDA TEMP2           ;
68DB 9507     1189         STA ADDRH1,X        ;
68DD AD3672   1190         LDA TEMP1           ;PIXEL INTENSITY VALUE
68E0 60       1191         RTS                 ;
68E1          1192  ;                          ;
68E1          1193  ;                          ;
68E1          1194  ;                          ;
68E1          1195  ;                          ;
68E1          1196  ;                          ;
68E1 A106     1197  READ   LDA (ADDRL1,X)      ;GET PIXEL AND ITS NEIGHBORS
68E3 3D5672   1198         AND MASK,X          ;ISOLATE PIXEL
68E6 F003     1199         BEQ BLCX            ;IS IT ZERO?
68E8 EE3672   1200         INC TEMP1           ;NO, INCREMENT VALUE COUNTER
68EB 60       1201  BLCX   RTS                 ;
68EC          1202  ;                          ;
68EC          1203  ;                          ;
68EC          1204  ;                          ;
68EC          1205  ;                          ;
68EC          1206  ;                          ;
68EC          1207  ;                          ;
68EC 8D3672   1208  STRVAL STA TEMP1           ;SAVE INTENSITY VALUE
68EF B507     1209         LDA ADDRH1,X        ;
68F1 8D3772   1210         STA TEMP2           ;CALCULATE SECONDARY PAGE ADDRESS
68F4 4960     1211         XOR #$60            ;
68F6 9507     1212         STA ADDRH1,X        ;
68F8 200769   1213         JSR WRITE           ;WRITE WEIGHT 1 BIT
68FB AD3772   1214         LDA TEMP2           ;
68FE 200769   1215         JSR WRITE           ;WRITE WEIGHT 2 BIT
6901 AD3772   1216         LDA TEMP2           ;
6904 9507     1217         STA ADDRH1,X        ;RESTORE ADDRESS
6906 60       1218         RTS                 ;
```

```
6907 9507   1224  WRITE  STA ADDRH1,X    ;FIX ADDRESS FOR RIGHT PAGE
6909 A106   1225         LDA (ADDRL1,X) ;GET PIXEL AND ITS NEIGHBORS
690B 5D5672 1226         XOR MASK,X     ;ISOLATE PIXEL
690E 2106   1227         AND (ADDRL1,X) ;
6910 4E3672 1228         LSR TEMP1      ;TEST FOR BIT SET
6913 9003   1229         BCC HIT        ;ZERO PIXEL BIT
6915 5D5672 1230         XOR MASK,X     ;SET PIXEL BIT
6918 8106   1231  HIT    STA (ADDRL1,X) ;MODIFY SCREEN
691A 60     1232         RTS            ;
691B        1233  ;                     ;
691B        1234  ;                     ;
691B        1235  ;                     ;
691B        1236  ;                     ;
691B        1237  ;                     ;
691B        1238  ;                     ;NOT USED RIGHT NOW, BLANX
691B 202269 1239  BLANK  JSR PBLANX    ;BOTH SCREENS
691E 202769 1240         JSR SBLANX     ;
6921 60     1241         RTS            ;
6922        1242  ;                     ;
6922        1243  ;                     ;
6922        1244  ;                     ;
6922        1245  ;                     ;
6922        1246  ;                     ;
6922 A920   1247  PBLANX LDA #$20       ;HIGH PART OF ADDRESS
6924 4C2C69 1248         JMP BLNX       ;ZEROS NEXT 8192 BYTES
6927        1249  ;                     ;
6927        1250  ;                     ;
6927        1251  ;                     ;
6927        1252  ;                     ;
6927        1253  ;                     ;
6927 A940   1254  SBLANX LDA #$40       ;DO SECONDARY PAGE
6929 4C2C69 1255         JMP BLNX       ;
692C        1256  ;                     ;
692C        1257  ;                     ;
692C        1258  ;                     ;
692C        1259  ;                     ;
692C        1260  ;                     ;
692C 9507   1261  BLNX   STA ADDRH1,X   ;SET HIGH BYTE OF ADDRESS
692E A900   1262         LDA #$0        ;
6930 9506   1263         STA ADDRL1,X   ;SET LOW PART OF ADDRESS
6932 A900   1264  DRLOP  LDA #$0        ;
6934 8106   1265         STA (ADDRL1,X) ;ZERO A BYTE
6936 F606   1266         INC ADDRL1,X   ;
6938 D002   1267         BNE NOCAR      ;
693A F607   1268         INC ADDRH1,X   ;
693C B507   1269  NOCAR  LDA ADDRH1,X   ;TEST FOR END OF GRAPHICS
693E 291F   1270         AND #$1F       ;AREA
6940 C91F   1271         CMP #$1F       ;
6942 D0EE   1272         BNE DRLOP      ;
```

```
694F AC2972 1280 ADSTRT LDY MOVMOD     ;WHICH WAY IS THE SCREEN
6952 1021   1281        BPL NEGNOT     ;SCANNED? POS. TOP LFT CORNER
6954 A93F   1282        LDA #$3F       ;
6956 9507   1283        STA ADDRH1,X   ;BOTTOM RIGHT CORNER
6958 A9F7   1284        LDA #$F7       ;
695A 9506   1285        STA ADDRL1,X   ;
695C A940   1286        LDA #$40       ;
695E 9D5672 1287        STA MASK,X     ;
6961 20376C 1288        JSR XDIV       ;HARDWIRE ALL ADDRESS AND
6964 A903   1289        LDA #$3        ;
6966 9D4A72 1290        STA TXEXT,X    ;POSITION BYTES
6969 A945   1291        LDA #$45       ;
696B 9D5272 1292        STA TXPOS,X    ;
696E A9BF   1293        LDA #$BF       ;
6970 9D4E72 1294        STA TYPOS,X    ;
6973 D01B   1295        BNE DNALL      ;THAT WAS FAST
6975 A920   1296 NEGNOT LDA #$20       ;
6977 9507   1297        STA ADDRH1,X   ;
6979 A900   1298        LDA #$0        ;TOP LEFT CORNER ADDRESSES
697B 9506   1299        STA ADDRL1,X   ;
697D A901   1300        LDA #$1        ;
697F 9D5672 1301        STA MASK,X     ;
6982 20376C 1302        JSR XDIV       ;
6985 A900   1303        LDA #$0        ;NOW COMES THE EASY PART!
6987 9D4A72 1304        STA TXEXT,X    ;
698A 9D4E72 1305        STA TYPOS,X    ;
698D 9D5272 1306        STA TXPOS,X    ;
6990 203B6C 1307 DNALL  JSR XMUL       ;RESTORE INDEX
6993 60     1308        RTS            ;
6994        1309 ;                     ;
6994        1310 ;                     ;
6994 A901   1311 MOVE1  LDA #$1        ;SWEEPS OUT LINES, ACCORDING TO
6996 AC2972 1312        LDY MOVMOD     ;MOVMOD VALUE, RETURNS Y=80
6999 1006   1313        BPL PLUS       ;AT END OF LINE
699B A9FF   1314        LDA #$FF       ;
699D C080   1315        CPY #$80       ;MOVMOD=0, LEFT TO RIGHT, AND DWN.
699F F002   1316        BEQ SIDEWY     ;MOVMOD=1, DOWN, NEXT LINE TO RHT.
69A1 D004   1317 PLUS   BNE UPDOWN     ;MOVMOD=80,RHT TO LEFT, AND UP
69A3 20D069 1318 SIDEWY JSR NEXTX      ;MOVMOD=FF,UP, NEXT LINE LEFT
69A6 60     1319        RTS            ;
69A7 20546A 1320 UPDOWN JSR NEXTY      ;
69AA 60     1321        RTS            ;
69AB        1322 ;                     ;
69AB        1323 ;                     ;
69AB 206568 1324 MOVE2  JSR PRESTO     ;DOES NEXT LINE FOR MOVE1
69AE 20B869 1325        JSR MOVE3      ;USUALLY CALLED AFTER E.O.L.
69B1 8C3672 1326        STY TEMP1      ;
69B4 202D68 1327        JSR PSAVE      ;
69B7 AC3672 1328        LDY TEMP1      ;
69BA 60     1329        RTS            ;
69BB        1330 ;                     ;
69BB A901   1331 MOVE3  LDA #$1        ;GETS NEXT LINE START ADDRESS
69BD AC2972 1332        LDY MOVMOD     ;
69C0 1004   1333        BPL NOTNEG     ;WHICH IS PUSHED ON STO*
69C2 A9FF   1334        LDA #$FF       ;
69C4 C080   1335        CPY #$80       ;BY MOVE2
69C6 F004   1336 NOTNEG BEQ STOS      ;
69C8 20D069 1337        JSR NEXTX      ;CALLED ONLY BY MOVE2
69CB 60     1338        RTS            ;
69CC 20546A 1339 STOS   JSR NEXTY      ;
69CF 60     1340        RTS            ;
```

```
69D0          1341  ;                         ;
69D0          1342  ;                         ;
69D0          1343  ;                         ;
69D0 8D3672   1344  NEXTX  STA TEMP1           ;SAVE STEP DIRECTION
69D3 AD3672   1345         LDA TEMP1          ;SET SIGN FLAG
69D6 303F     1346         BMI BACJ           ;NEGATIVE STEP
69D8 1E5672   1347         ASL MASK,X         ;
69DB 301B     1348         BMI OVER           ;SHIFTED IN TO SIGN, NEXT ADDR
69DD 20376C   1349  UNDER  JSR XDIV           ;POSITION INDEX
69E0 FE5272   1350         INC TXPOS,X        ;BUMP X POSITION BYTE
69E3 BD5272   1351         LDA TXPOS,X        ;
69E6 C946     1352         CMP #$46           ;TEST FOR OUT OF RANGE
69E8 3008     1353         BMI XOUT           ;
69EA A900     1354         LDA #$0            ;ZERO IT
69EC 9D5272   1355         STA TXPOS,X        ;AND INCREMENT
69EF FE4A72   1356         INC TXEXT,X        ;EXTENSION OF X
69F2 203B6C   1357  XOUT   JSR XMUL           ;RESTORE INDEX
69F5 A000     1358         LDY #$0            ;EDGE OF SCREEN NOT FOUND
69F7 60       1359         RTS                ;
69F8 B506     1360  OVER   LDA ADDRL1,X       ;GET LOW PART OF ADDRESS
69FA 297F     1361         AND #$7F           ;BECAUSE THIS IS THE PART
69FC C927     1362         CMP #$27           ;THAT CHANGES WITH X
69FE F011     1363         BEQ FTPD           ;
6A00 C94F     1364         CMP #$4F           ;TEST TO FIND WHAT VETICAL
6A02 F00D     1365         BEQ FTPD           ;BLOC* IS BEING ADDRESSED
6A04 C977     1366         CMP #$77           ;
6A06 F009     1367         BEQ FTPD           ;
6A08 A901     1368         LDA #$1            ;
6A0A 9D5672   1369         STA MASK,X         ;NOT AT THE END OF A LINE
6A0D F606     1370         INC ADDRL1,X       ;SO BUMP THE LOW PART OF
6A0F D0CC     1371         BNE UNDER          ;ADDRESS
6A11 5E5672   1372  FTPD   LSR MASK,X         ;END OF LINE
6A14 A080     1373  LRGS   LDY #$80           ;SET FLAG
6A16 60       1374         RTS                ;
6A17 5E5672   1375  BACJ   LSR MASK,X         ;SHIFT TO RIGHT TO MOVE TO LEFT
6A1A 9020     1376         BCC MUNDER         ;CARRY SET MEANS OUT OF ADDRESS
6A1C B506     1377         LDA ADDRL1,X       ;
6A1E 297F     1378         AND #$7F           ;
6A20 C900     1379         CMP #$0            ;TEST FOR BEGINNING OF LINE
6A22 F011     1380         BEQ STPD           ;
6A24 C928     1381         CMP #$28           ;AT EACH BLOC*
6A26 F00D     1382         BEQ STPD           ;
6A28 C950     1383         CMP #$50           ;
6A2A F009     1384         BEQ STPD           ;
6A2C D606     1385         DEC ADDRL1,X       ;NOT AT END OF LINE, DECREMENT
6A2E A940     1386         LDA #$40           ;LOW PART OF ADDRESS
6A30 9D5672   1387         STA MASK,X         ;NEW MAS* WORD
6A33 D007     1388         BNE MUNDER         ;
6A35 38       1389  STPD   SEC                ;MAS* ??
6A36 3E5672   1390         ROL MASK,X         ;
6A39 4C146A   1391         JMP LRGS           ;RESTORE MAS* AND SET ERROR FLAG
6A3C 20376C   1392  MUNDER JSR XDIV           ;
6A3F DE5272   1393         DEC TXPOS,X        ;FIX UP POSITION BYTES
6A42 10AE     1394         BPL XOUT           ;
6A44 A945     1395         LDA =$45           ;
6A46 9D5272   1396         STA TXPOS,X        ;
6A49 DE4A72   1397         DEC TXEXT,X        ;
6A4C 10A4     1398         BPL XOUT           ;
6A4E 203B6C   1399         JSR XMUL           ;THESE INSTRUCTIONS ARE
6A51 A080     1400         LDY #$80           ;
6A53 60       1401         RTS                ;EXTRANEOUS & NEEDLESSLY VERBOUS
```

```
6A54            1402  ;                              ;
6A54            1403  ;                              ;
6A54            1404  ;                              ;
6A54            1405  ;                              ;
6A54            1406  ;                              ;
6A54            1407  ;                              ;
6A54            1408  ;                              ;
6A54            1409  ;                              ;
6A54            1410  ;                              ;
6A54 8D3672     1411  NEXTY  STA TEMP1                ;SET SIGN FLAG
6A57 AD3672     1412         LDA TEMP1                ;TO GET DIRECTION OF MOVE
6A5A 306A       1413         BMI BACK                 ;
6A5C B507       1414         LDA ADDRH1,X             ;SAVE A VERSION OF
6A5E 8D3772     1415         STA TEMP2                ;EACH BECAUSE THEY WILL BE
6A61 B506       1416         LDA ADDRL1,X             ;MODIFIED FOR
6A63 8D3672     1417         STA TEMP1                ;TEST PURPOSES
6A66 4A         1418         LSR                      ;ZERO THE SIGN BIT
6A67 C968       1419         CMP #$68                 ;AND TEST FOR LAST LINE OF
6A69 300A       1420         BMI LINER                ;A SECTION
6A6B AD3772     1421         LDA TEMP2                ;IS IT THE LAST OF THE LAST?
6A6E C93F       1422         CMP #$3F                 ;
6A70 D003       1423         BNE LINER                ;NO, MORE VERTICAL MOVES LEFT
6A72 A080       1424         LDY #$80                 ;EDGE OF SCREEN
6A74 60         1425         RTS                      ;
6A75 AD3772     1426  LINER  LDA TEMP2                ;GET LOW ADDRESS
6A78 18         1427         CLC                      ;
6A79 6904       1428         ADC #$4                  ;TEST FOR END OF SECTION
6A7B A8         1429         TAY                      ;
6A7C 2940       1430         AND #$40                 ;
6A7E D004       1431         BNE BLOC                 ;GO TO NEXT BLOC
6A80 98         1432         TYA                      ;
6A81 4CB36A     1433         JMP LOADS                ;DONE IT, NOW EXIT
6A84 AD3772     1434  BLOC   LDA TEMP2                ;GET NEXT BLOC
6A87 38         1435         SEC                      ;ADDRESS
6A88 E91C       1436         SBC #$1C                 ;
6A8A 8D3772     1437         STA TEMP2                ;
6A8D AD3672     1438         LDA TEMP1                ;
6A90 18         1439         CLC                      ;FIX LOW ADDRESS BYTE FOR THIS
6A91 6980       1440         ADC #$80                 ;BLOC
6A93 8D3672     1441         STA TEMP1                ;
6A96 A900       1442         LDA #$0                  ;CARRY ANY OVERFLOW
6A98 6D3772     1443         ADC TEMP2                ;ADDRESSES ARE 16 BITS BUT DATA
6A9B 8D3772     1444         STA TEMP2                ;ISN'T
6A9E 2904       1445         AND #$4                  ;
6AA0 D003       1446         BNE SECTN                ;OOPS! GONE OUT OF BLOC TO SECTION
6AA2 4CAE6A     1447         JMP FLOAD                ;DONE NOW
6AA5 AD3672     1448  SECTN  LDA TEMP1                ;
6AA8 18         1449         CLC                      ;NEXT SECTION
6AA9 6928       1450         ADC #$28                 ;
6AAB 8D3672     1451         STA TEMP1                ;
6AAE AD3772     1452  FLOAD  LDA TEMP2                ;
6AB1 29FB       1453         AND #$FB                 ;
6AB3 9507       1454  LOADS  STA ADDRH1,X             ;SAVE NEW ADDRESS
6AB5 AD3672     1455         LDA TEMP1                ;
6AB8 9506       1456         STA ADDRL1,X             ;
6ABA 20376C     1457         JSR XDIV                 ;
6ABD FE4E72     1458         INC TYPOS,X              ;GET POSITION INDEXES
6AC0 203B6C     1459         JSR XMUL                 ;FIX Y POSITION BYTE
6AC3 A000       1460         LDY #$0                  ;
6AC5 60         1461         RTS                      ;
6AC6            1462  ;                              ;
```

```
6AC6            1463  ;                       ;;
6AC6            1464  ;                       ;
6AC6            1465  ;                       ;
6AC6            1466  ;                       ;BACK DOES THE NEGATIVE Y STEP
6AC6            1467  ;                       ;IT IS THE COMPLEMENT OF LINER
6AC6            1468  ;                       ;AND MOVES CURSOR UP
6AC6            1469  ;                       ;
6AC6            1470  ;                       ;
6AC6 B507       1471  BACK    LDA ADDRH1,X    ;SAVE ADDRESSES, BECAUSE THEY
6AC8 8D3772     1472          STA TEMP2       ;WILL BE MODIFIED A LOT
6ACB A8         1473          TAY             ;
6ACC B506       1474          LDA ADDRL1,X    ;
6ACE 8D3672     1475          STA TEMP1       ;
6AD1 29C0       1476          AND #$C0        ;FIRST TEST FOR LAST LINE
6AD3 D010       1477          BNE LINEX       ;
6AD5 C020       1478          CPY #$20        ;TOP LINE ADDRESS
6AD7 D00C       1479          BNE LINEX       ;
6AD9 AD3672     1480          LDA TEMP1       ;GET LOW  ADDRESS
6ADC 293F       1481          AND #$3F        ;
6ADE C928       1482          CMP #$28        ;
6AE0 1003       1483          BPL LINEX       ;
6AE2 A080       1484          LDY #$80        ;TOP LINE, CAN'T GO ANY FURTHER
6AE4 60         1485          RTS             ;
6AE5 AD3772     1486  LINEX   LDA TEMP2       ;MOVE UP A LINE IN A BOX
6AE8 38         1487          SEC             ;
6AE9 E904       1488          SBC #$4         ;BY SUBTRACTING $4
6AEB A8         1489          TAY             ;
6AEC 2920       1490          AND #$20        ;AS LONG AS ITS NOT LESS THAN 20
6AEE F004       1491          BEQ BLOCK       ;TEST FOR SIZE
6AF0 98         1492          TYA             ;
6AF1 4C1F6B     1493          JMP BLOADS      ;MOVE IS DONE, STORE NEW ADDRESS
6AF4 AD3772     1494  BLOCK   LDA TEMP2       ;MOVE UP TO NEXT BOX
6AF7 18         1495          CLC             ;
6AF8 691C       1496          ADC #$1C        ;DERIVE NEXT BOX ADDRESS
6AFA 8D3772     1497          STA TEMP2       ;SAVE IT
6AFD AD3672     1498          LDA TEMP1       ;
6B00 38         1499          SEC             ;FIX LOW PART OF ADDRESS
6B01 E980       1500          SBC #$80        ;IT CHANGES BY $80 OVER
6B03 8D3672     1501          STA TEMP1       ;EACH BOX
6B06 AD3772     1502          LDA TEMP2       ;
6B09 E900       1503          SBC #$0         ;BORROW FROM LOW BYTE
6B0B 8D3772     1504          STA TEMP2       ;
6B0E C93B       1505          CMP #$3B        ;USED UP THIS BOX?
6B10 D010       1506          BNE BLOAD       ;NO, GOOD ADDRESSES
6B12 AD3672     1507          LDA TEMP1       ;
6B15 100B       1508          BPL BLOAD       ;
6B17 38         1509          SEC             ;
6B18 E928       1510          SBC #$28        ;GO TO NEXT SECTION
6B1A 8D3672     1511          STA TEMP1       ;
6B1D A93F       1512          LDA #$3F        ;NEW HIGH ADDRESS
6B1F 8D3772     1513  BLOADS  STA TEMP2       ;
6B22 AD3772     1514  BLOAD   LDA TEMP2       ;
6B25 9507       1515          STA ADDRH1,X    ;SAVE ADDRESSES
6B27 AD3672     1516          LDA TEMP1       ;
6B2A 9506       1517          STA ADDRL1,X    ;
6B2C 20376C     1518          JSR XDIV        ;GET OTHER INDEX
6B2F DE4E72     1519          DEC TYPOS,X     ;UPDATE POSITION WORD
6B32 20386C     1520          JSR XMUL        ;
6B35 A000       1521          LDY #$0         ;JUST IN CASE
6B37 60         1522          RTS             ;
6B38            1523  ;                       ;
```

```
6B38          1524  ;                       ;** XORCUR ** DOES THE CURSOR
6B38          1525  ;                       ;BY XOR'ING A CROSS AT CURSOR LOC.
6B38 203F6C   1526  XORCUR JSR LOCATE        ;GET ADDRESSES FROM POSITION BYTES
6B3B 8D3072   1527         STA REM           ;SAVE REMAINDER FOR MAS* DERIVATN
6B3E 202D68   1528         JSR PSAVE         ;SAVE ADDRESSES AND POSITION BYTES
6B41 AD3072   1529         LDA REM           ;
6B44 C904     1530         CMP #$4           ;TWO TABLE LOOX UPS
6B46 1011     1531         BPL BIGGER        ;
6B48 A9F0     1532         LDA #$F0          ;GET XOR'ING WORDS FOR HORIZONTAL
6B4A 8D2B72   1533         STA FIRST         ;
6B4D A90F     1534         LDA #$F           ;PART OF CURSOR
6B4F 8D2C72   1535         STA SECOND        ;
6B52 A900     1536         LDA #$0           ;
6B54 8D2D72   1537         STA THIRD         ;
6B57 F011     1538         BEQ GOAHD         ;GO AND SHIFT THEM, ACCORDNG T REM
6B59 A900     1539  BIGGER LDA #$0           ;OTHER SET OF XOR'ING WORDS
6B5B 8D2B72   1540         STA FIRST         ;
6B5E A9FE     1541         LDA #$FE          ;
6B60 8D2C72   1542         STA SECOND        ;
6B63 A901     1543         LDA #$1           ;
6B65 8D2D72   1544         STA THIRD         ;
6B68 A904     1545         LDA #$4           ;
6B6A 18       1546  GOAHD  CLC               ;CLEAR CARRY BIT SO IT DOESN'T GET
6B6B CD3072   1547         CMP REM           ;SHIFTED IN
6B6E F00E     1548         BEQ FIN           ;COUNTED UP TO REM // SHIFT COUNT/
6B70 2E2B72   1549         ROL FIRST         ;SHIFT THEM ALL TOGETHER
6B73 2E2C72   1550         ROL SECOND        ;EVEN THOUGH ONE IS ALWAYS ZERO
6B76 2E2D72   1551         ROL THIRD         ;AND TWO SOMETIMES ARE
6B79 18       1552         CLC               ;
6B7A 6901     1553         ADC #$1           ;SHIFT COUNTER INCREMENT TO REM
6B7C D0EC     1554         BNE GOAHD         ;JUMP ALWAYS
6B7E C903     1555  FIN    CMP #$3           ;WHERE IS THE CURSOR IN RELATION
6B80 F029     1556         BEQ NOMOV         ;TO A PIXEL IN A BYTE?
6B82 1015     1557         BPL NXWRD         ;
6B84 A9F9     1558         LDA #$F9          ;MOVE SEVEN POSITIONS TO THE LEFT
6B86 20436D   1559         JSR XMOVE         ;
6B89 C080     1560         CPY #$80          ;HIT EDGE OF SCREEN?
6B8B F01E     1561         BEQ NOMOV         ;
6B8D 20826C   1562         JSR XFIND         ;GET ADDRESSES, POSITION BYTES
6B90 AD2B72   1563         LDA FIRST         ;WERE DIDDLED
6B93 20E66B   1564         JSR XORWRD        ;DO LEFT HORIZONTAL PART OF CURSOR
6B96 4CAB6B   1565  DONT   JMP NOMOV        ;NO NEED TO RECALCULATE ADDRESSES
6B99 A907     1566  NXWRD  LDA #$7           ;MOVE TO RIGHT SEVEN POSITIONS
6B9B 20436D   1567         JSR XMOVE         ;
6B9E C080     1568         CPY #$80          ;
6BA0 F009     1569         BEQ NOMOV         ;
6BA2 20826C   1570         JSR XFIND         ;
6BA5 AD2D72   1571         LDA THIRD         ;DO RIGHT PART OF CURSOR FIRST
6BA8 20E66B   1572         JSR XORWRD        ;
6BAB 205668   1573  NOMOV  JSR PEEK          ;RESTORE CENTRE ADDRESS
6BAE AD2C72   1574         LDA SECOND        ;DO THE IMMEDIATE PIXELS AROUND
6BB1 20E66B   1575         JSR XORWRD        ;THE CURSOR CENTRE
6BB4 A901     1576  YSTUFF LDA #$1           ;NOW FOR THE VERTICAL PART
6BB6 8D2A72   1577         STA VECTOR        ;POSITIVE VECTOR FOR MOVE DOWN
6BB9 20CB6B   1578         JSR VERT          ;DOES THREE PIXELS IN SEQUENCE
6BBC 205668   1579         JSR PEEK          ;RESTORE ADDRESS
6BBF A9FF     1580         LDA #$FF          ;GOING UP NOW
6BC1 8D2A72   1581         STA VECTOR        ;
6BC4 20CB6B   1582         JSR VERT          ;
6BC7 206568   1583         JSR PRESTO        ;POP ADDRESSES, MODIFY STAX POINT.
6BCA 60       1584         RTS               ;
```

```
6BCB          1585  ;                              ;
6BCB          1586  ;                              ;
6BCB          1587  ;                              ;
6BCB          1588  ;                              ;
6BCB          1589  ;                              ;
6BCB A9FD     1590  VERT   LDA #$FD                ;NEGATIVE THREE
6BCD 8D6172   1591         STA COUNT               ;
6BD0 AD2A72   1592  YCURS  LDA VECTOR              ;VECTOR IS EITHER POS. OR NEG.
6BD3 20546A   1593         JSR NEXTY               ;DEPENDING ON DIRECTION
6BD6 C080     1594         CPY #$80                ;OUT OF BOUNDS?
6BD8 F00B     1595         BEQ FINCUR              ;
6BDA BD5672   1596         LDA MASK,X              ;DO ONE PIXEL
6BDD 20E66B   1597         JSR XORWRD              ;
6BE0 EE6172   1598         INC COUNT               ;BUMP COUNTER
6BE3 D0EB     1599         BNE YCURS               ;
6BE5 60       1600  FINCUR RTS                     ;
6BE6          1601  ;                              ;
6BE6          1602  ;                              ;
6BE6 297F     1603  XORWRD AND #$7F                ;NO SIGN BITS PLEASE, WE HAVE NO
6BE8 8D3672   1604         STA TEMP1               ;COLOUR TV
6BEB AD3D72   1605         LDA BLTFLG              ;ARE WE DOING BLOTTO?
6BEE 3003     1606         BMI CONT.               ;NO, JUST A BORING CURSOR
6BF0 4C046C   1607         JMP BLTWRD              ;YEAH, LETS DRAW SOME PORNO
6BF3 B507     1608  CONT.  LDA ADDRH1,X            ;DO BOTH SCREENS
6BF5 8D3772   1609         STA TEMP2               ;
6BF8 4960     1610         XOR #$60                ;GET SECOND PAGE ADDRESS
6BFA 201B6C   1611         JSR XORIT               ;DO CURSOR PART
6BFD AD3772   1612         LDA TEMP2               ;GET PRIMARY PAGE ADDRESS
6C00 201B6C   1613         JSR XORIT               ;DO THAT CURSOR
6C03 60       1614         RTS                     ;
6C04          1615  ;                              ;
6C04          1616  ;                              ;
6C04 AD3B72   1617  BLTWRD LDA BRUSH               ;WHAT COMBINATION OF SCREENS?
6C07 8D3872   1618         STA TEMP3               ;
6C0A B507     1619         LDA ADDRH1,X            ;SAME SORT OF ADDRESS MANIPULATION
6C0C 8D3772   1620         STA TEMP2               ;AS ABOVE
6C0F 4960     1621         XOR #$60                ;
6C11 20256C   1622         JSR BLITZ               ;DRAW OR ERASE, AS OPPOSED TO XOR
6C14 AD3772   1623         LDA TEMP2               ;RESTORE PRIMARY ADDRESS
6C17 20256C   1624         JSR BLITZ               ;
6C1A 60       1625         RTS                     ;
6C1B          1626  ;                              ;
6C1B          1627  ;                              ;
6C1B 9507     1628  XORIT  STA ADDRH1,X            ;STORE HIGH PART OF ADDRESS
6C1D A106     1629         LDA (ADDRL1,X)          ;GET PIXEL AND ITS NEIGHBORS
6C1F 4D3672   1630         XOR TEMP1               ;XOR WITH CURSOR PART
6C22 8106     1631         STA (ADDRL1,X)          ;REWRITE TO SCREEN
6C24 60       1632         RTS                     ;
6C25          1633  ;                              ;
6C25          1634  ;                              ;
6C25 9507     1635  BLITZ  STA ADDRH1,X            ;STORE HIGH PART OF ADDRESS
6C27 A106     1636         LDA (ADDRL1,X)          ;
6C29 0D3672   1637         ORA TEMP1               ;OR IN CURSOR PART, WRITES IT
6C2C 6E3872   1638         ROR TEMP3               ;REMEMBER TEMP3?, ITS THE BRUSH
6C2F B003     1639         BCS STOREX              ;CARRY SET, DON'T ERASE CURSOR
6C31 4D3672   1640         XOR TEMP1               ;ERASE WHAT WAS WRITTEN
6C34 8106     1641  STOREX STA (ADDRL1,X)          ;WRITE IT ALL TO SCREEN
6C36 60       1642         RTS                     ;
6C37          1643  ;                              ;
6C37          1644  ;                              ;
6C37          1645  ;                              ;
```

```
6C37          1646  ;                         ;
6C37          1647  ;                         ;
6C37          1648  ;                         ;
6C37 8A       1649  XDIV   TXA                 ;SETS X-REGISTER INDEX TO POSITION
6C38 4A       1650        LSR                 ;BYTES, HAS VALUE 0,1
6C39 AA       1651        TAX                 ;
6C3A 60       1652        RTS                 ;
6C3B          1653  ;                         ;
6C3B 8A       1654  XMUL   TXA                 ;SETS X-REGISTER INDEX TO ADDRESS
6C3C 0A       1655        ASL                 ;BYTES, HAS VALUE 0,2
6C3D AA       1656        TAX                 ;
6C3E 60       1657        RTS                 ;
6C3F          1658  ;                         ;
6C3F          1659  ;                         ;
6C3F          1660  ;                         ;
6C3F          1661  ;                         ;
6C3F 20466C   1662  LOCATE JSR YFIND          ;GET ADDRESSES FROM BOTH CALCULATN
6C42 20826C   1663        JSR XFIND           ;ROUTINES,HAVE TO BE CALLED IN
6C45 60       1664        RTS                 ;THIS ORDER, OR THERES TROUBLE
6C46          1665  ;                         ;BECAUSE XTEMP,X ISN'T PASSED
6C46          1666  ;                         ;
6C46          1667  ;                         ;
6C46          1668  ;                         ;
6C46          1669  ;                         ;
6C46          1670  ;                         ;** YFIND DOES A PARTIAL TABLE **
6C46          1671  ;                         ;** LOO*-UP OF ADDRESSES, USING
6C46 8A       1672  YFIND  TXA                 ;YPOS
6C47 4A       1673        LSR                 ;THIS SHOULD BE A CALL TO XDIV
6C48 A8       1674        TAY                 ;
6C49 A920     1675        LDA #$20            ;PRIMARY PAGE IS BASE CURSOR ADDR.
6C4B 9507     1676        STA ADDRH1,X        ;
6C4D B94E72   1677        LDA TYPOS,Y         ;ROTATE TO GET SECTION #, 0,1,3
6C50 18       1678        CLC                 ;
6C51 2A       1679        ROL                 ;
6C52 2A       1680        ROL                 ;
6C53 8D3672   1681        STA TEMP1           ;SAVE THESE ROL'S,
6C56 2A       1682        ROL                 ;THIS IS SECTION #
6C57 2903     1683        AND #$3             ;HERE IT IS
6C59 A8       1684        TAY                 ;PUT SECTION # IN Y REGISTER
6C5A AD3672   1685        LDA TEMP1           ;
6C5D 2A       1686        ROL                 ;GET BOX AND LINE IN BOX
6C5E 2E3772   1687        ROL TEMP2           ;
6C61 2A       1688        ROL                 ;
6C62 2E3772   1689        ROL TEMP2           ;
6C65 2980     1690        AND #$80            ;THIS IS BASE PART OF LOW ADDRESS
6C67 192870   1691        ORA YSTRL,Y         ;LOW ADDRESS BYTE, ADD X POSITION
6C6A 9D5772   1692        STA XTEMP,X         ;DIVIDED BY 7 TO GET LOW ADDRESS
6C6D AD3772   1693        LDA TEMP2           ;
6C70 2903     1694        AND #$3             ;LINE NUMBER COMING UP
6C72 18       1695        CLC                 ;
6C73 7507     1696        ADC ADDRH1,X        ;
6C75 9507     1697        STA ADDRH1,X        ;SSAVE ADDRESS
6C77 AD3672   1698        LDA TEMP1           ;BOX NUMBER NOW
6C7A 291C     1699        AND #$1C            ;
6C7C 18       1700        CLC                 ;ADD BOX NUMBER ADDRESS PART
6C7D 7507     1701        ADC ADDRH1,X        ;
6C7F 9507     1702        STA ADDRH1,X        ;GOT IT!
6C81 60       1703        RTS                 ;
6C82          1704  ;                         ;
6C82          1705  ;                         ;
6C82          1706  ;                         ;
```

```
6C82          1707  ;                         ;
6C82          1708  ;                         ;
6C82          1709  ;                         ;
6C82          1710  ;                         ;
6C82          1711  ;                         ;
6C82          1712  ;                         ;
6C82          1713  ;                         ;
6C82          1714  ;                         ;** XFIND CALCULATES THE LOW
6C82          1715  ;                         ;** PART OF THE ADDRESS, USING
6C82 8A       1716  XFIND   TXA               ;** TXEXT, AND TXPOS
6C83 4A       1717          LSR               ;
6C84 A8       1718          TAY               ;THIS SHOULD BE XDIV
6C85 B95272   1719          LDA TXPOS,Y       ;GET X POSITION BYTE AND DIVIDE
6C88 20DC6C   1720          JSR DIV7          ;IT BY SEVEN TO GET ADDRESS DISP.
6C8B 8D3672   1721          STA TEMP1         ;REMAINDER IS USED TO DO MASX
6C8E B94A72   1722          LDA TXEXT,Y       ;USED FOR A TABLE GRAB
6C91 2903     1723          AND #$3           ;NOT NECESSARY NOW, BUT THERE ARE
6C93 A8       1724          TAY               ; 5 EXTRA BITS HERE TO USE LATER
6C94 AD3172   1725          LDA DVD           ;GET ADDRESS DISPLACEMENT
6C97 18       1726          CLC               ;
6C98 792B70   1727          ADC XSTR,Y        ;ADD BASE ADDRESS
6C9B 7D5772   1728          ADC XTEMP,X       ;ADD BASE ADDRESS FROM Y LOCATE
6C9E 9506     1729          STA ADDRL1,X      ;THATS IT FOR THE ADDRESS FOLX
6CA0 AC3672   1730          LDY TEMP1         ;NOW FOR THE MASX
6CA3 B92F70   1731          LDA MSK,Y         ;JUST GET IT, THATS ALL
6CA6 9D5672   1732  OUT     STA MASK,X        ;SAVE IT
6CA9 AD3672   1733          LDA TEMP1         ;**HAVE TO RETURN REMAINDER FOR
6CAC 60       1734          RTS               ;CURSOR PROGRAM
6CAD          1735  ;                         ;
6CAD          1736  ;                         ;
6CAD          1737  ;                         ;
6CAD          1738  ;                         ;;
6CAD          1739  ;                         ;
6CAD          1740  ;                         ;
6CAD          1741  ;                         ;
6CAD          1742  ;                         ;
6CAD          1743  ;                         ;;
6CAD          1744  ;                         ;
6CAD          1745  ;                         ;
6CAD          1746  ;                         ;
6CAD          1747  ;                         ;
6CAD          1748  ;                         ;**MULT DOES A 8 X 8 MULTIPLY
6CAD 8D3372   1749  MULT    STA MOD           ;ABOUT AS FAST AS YOU CAN DO IT
6CB0 AD3372   1750          LDA MOD           ;
6CB3 F019     1751          BEQ MDONE         ;
6CB5 A900     1752          LDA #$0           ;SET UP PARAMETERS FIRST, CAUSE
6CB7 8C3072   1753          STY REM           ;WE'LL REALLY BE GOING LATER!!
6CBA AC3072   1754          LDY REM           ;
6CBD F00F     1755          BEQ MDONE         ;CHEX FOR ANY ZEROS, EXITS IMMED.
6CBF A008     1756          LDY #$8           ;ONLY 8 SHIFTS, THATS WHY ITS FAST
6CC1 0A       1757  MLOOP   ASL               ;PRODUCT IN A REGISTER
6CC2 2E3372   1758          ROL MOD           ;ROLL OFF A BIT
6CC5 9004     1759          BCC LOEC          ;A CHEX FOR MOD=0 COULD SPEED IT
6CC7 18       1760          CLC               ;
6CC8 6D3072   1761          ADC REM           ;ADD MULTIPLIER TO PRODUCT
6CCB 88       1762  LOEC    DEY               ;DECREMENT COUNT
6CCC D0F3     1763          BNE MLOOP         ;
6CCE 60       1764  MDONE   RTS               ;THATS IT
6CCF          1765  ;                         ;
6CCF          1766  ;                         ;
6CCF          1767  ;                         ;
```

```
6CCF          1768  ;                         ;
6CCF          1769  ;                         ;
6CCF          1770  ;                         ;*** FULL POSITIVE EIGHT BIT ***
6CCF          1771  ;                         ;**DIVIDE ROUTINES SOME HARDWIRED
6CCF          1772  ;                         ;FOR CERTAIN DIVISORS
6CCF          1773  ;                         ;USES SUCCESSIVE APPROXIMATION
6CCF 8C3572   1774  DIVTWO STY NUMBER         ;SAVE NUMBER TO BE DIVIDED
6CD2 4CE16C   1775        JMP DIVIDE          ;A-REGISTER HAS DIVISOR
6CD5 8D3572   1776  DIV10  STA NUMBER        ;DOES 10 DECIMAL DIVIDE
6CD8 A90A     1777        LDA #$A             ;$A = 10 DECIMAL, IF YOU FORGOT
6CDA D005     1778        BNE DIVIDE          ;JUMP TO DIVIDE
6CDC 8D3572   1779  DIV7   STA NUMBER        ;DOE 7 DECIMAL DIVIDE
6CDF A907     1780        LDA #$7             ;
6CE1 8D3272   1781  DIVIDE STA DSR           ;SAVE DIVISOR
6CE4 8D3472   1782        STA RESLT           ;AND NUMBER TO BE DIVIDED
6CE7 0E3472   1783        ASL RESLT           ;RESLT IS USED TWICE HERE
6CEA A900     1784        LDA #$0             ;SET UP PARAMETERS
6CEC 8D3172   1785        STA DVD             ;DVD IS THE QUOTIENT
6CEF A901     1786        LDA #$1             ;BIT POSITION BYTE
6CF1 8D3372   1787        STA MOD             ;
6CF4 AD3572   1788        LDA NUMBER          ;IS IT NEGATIVE OR ZERO?
6CF7 100A     1789  NEGLOP BPL POS            ;
6CF9 EE3172   1790        INC DVD             ;ITS SLOW, BUT IT WORX
6CFC 38       1791        SEC                 ;PERHAPS SOMEONE CAN
6CFD ED3272   1792        SBC DSR             ;IMPROVE THIS SECTION
6D00 4CF76C   1793        JMP NEGLOP          ;OF CODE TO MA*E IT FAST
6D03 F03D     1794  POS    BEQ ZEROP          ;ZERO DIVIDEND, GET OUTTA HERE
6D05 CD3272   1795        CMP DSR             ;COMPARE DIVIDEND TO DIVISOR
6D08 1006     1796        BPL SHFT            ;IF DIVISOR IS LARGER, QUOTIENT
6D0A A000     1797        LDY #$0             ;IS ZERO, REMAINDER = DIVIDEND
6D0C 8C3172   1798        STY DVD             ;FAST EXIT
6D0F 60       1799        RTS                 ;
6D10 CD3472   1800  SHFT   CMP RESLT          ;
6D13 300C     1801        BMI DIGIT           ;START AT LAST BIT SET
6D15 0E3472   1802        ASL RESLT           ;AND SHIFT RIGHT NEXT STAGE
6D18 0E3272   1803        ASL DSR             ;SMALL NUMBERS DIVIDE FAST
6D1B 0E3372   1804        ASL MOD             ;GET SET-UP FOR FAST
6D1E 4C106D   1805        JMP SHFT            ;SUCCESSIVE APPROXIMATION DVD
6D21 38       1806  DIGIT  SEC                 ;DSR IS SHIFTED AFTER EACH SUBTRCT
6D22 ED3272   1807        SBC DSR             ;
6D25 8D3472   1808        STA RESLT           ;A REGISTER HAS REMAINDER OF EACH
6D28 AD3172   1809        LDA DVD             ;SUBTRACT
6D2B 18       1810        CLC                 ;
6D2C 6D3372   1811        ADC MOD             ;ADD BECAUSE NEGATIVE NUMBERS
6D2F 8D3172   1812        STA DVD             ;MODIFY DVD, AND BIT CARRYS
6D32 AD3472   1813        LDA RESLT           ;HAVE TO BE DONE
6D35 CD3272   1814  BIT    CMP DSR             ;IS IT A TRUE REMAINDER NOW?
6D38 10E7     1815        BPL DIGIT           ;
6D3A 6E3272   1816        ROR DSR             ;SHIFT EVERYTHING RIGHT
6D3D 4E3372   1817        LSR MOD             ;THATS RIGHT, RIGHT!
6D40 90F3     1818        BCC BIT             ;
6D42 60       1819  ZEROP  RTS                 ;
6D43          1820  ;                         ;
6D43          1821  ;                         ;
6D43          1822  ;                         ;
6D43          1823  ;                         ;
6D43          1824  ;                         ;
6D43          1825  ;                         ;
6D43          1826  ;                         ;
6D43          1827  ;                         ;
6D43          1828  ;                         ;
```

```
6D43          1829  ;
6D43          1830  ;
6D43 8D3672   1831  XMOVE   STA TEMP1        ;** XMOVE HAS DISTANCE VALUE IN
6D46 8A       1832          TXA               ;ACCUMULATOR, AND DIDDLES TXEXT,X
6D47 4A       1833          LSR               ;AND TXPOS,X.. CALL XFIND NEXT
6D48 AA       1834          TAX               ;
6D49 BD4A72   1835          LDA TXEXT,X       ;SAME AS XDIV
6D4C A8       1836          TAY               ;
6D4D 2907     1837          AND #$7           ;AND OUT EXTRA BITS, IN CASE THEY
6D4F 9D4A72   1838          STA TXEXT,X       ;ARE BEING USED..THEY AREN'T NOW
6D52 98       1839          TYA               ;GET X EXTENSION
6D53 29F8     1840          AND #$F8          ;
6D55 8D4972   1841          STA MODE          ;MODE IS ACTUALLY THE EXTRA BITS
6D58 BD5272   1842          LDA TXPOS,X       ;
6D5B 18       1843          CLC               ;ADD UP XPOS AND DISTANCE
6D5C 6D3672   1844          ADC TEMP1         ;
6D5F 2E3672   1845          ROL TEMP1         ;IS SIGN BIT SET?
6D62 B032     1846          BCS NEGAD         ;MIGHT BE A SIGN OF WALL BANGING
6D64 C946     1847          CMP #$46          ;SEE IF TXPOS OVERFLOWED
6D66 301E     1848          BMI GONE          ;NO, EVERYTHINGS DONE
6D68 38       1849          SEC               ;
6D69 E946     1850          SBC #$46          ;DIVIDE OUT $46
6D6B FE4A72   1851          INC TXEXT,X       ;BUMP X EXTENSION
6D6E C946     1852          CMP #$46          ;IS IT STILL >$46 ?
6D70 3006     1853          BMI REST          ;
6D72 38       1854          SEC               ;
6D73 E946     1855          SBC #$46          ;
6D75 FE4A72   1856          INC TXEXT,X       ;CAN'T BE TOO BIG
6D78 BC4A72   1857  REST    LDY TXEXT,X       ;LET'S SEE WHAT DAMAGE HAS BEEN DN
6D7B C004     1858          CPY #$4           ; 0=< TXEXT =< 3
6D7D 3007     1859          BMI GONE          ;ITS GOOD
6D7F DE4A72   1860          DEC TXEXT,X       ;HIT THE RIGHT SIDE OF SCREEN
6D82 A080     1861          LDY #$80          ;STAY THERE, NO WRAP AROUND
6D84 A945     1862          LDA #$45          ;WE HOPE!!
6D86 9D5272   1863  GONE    STA TXPOS,X       ;
6D89 BD4A72   1864          LDA TXEXT,X       ;STORE THE POSITION BYTES
6D8C 0D4972   1865          ORA MODE          ;USELESS STUFF, ALWAYS ZERO
6D8F 9D4A72   1866          STA TXEXT,X       ;
6D92 8A       1867          TXA               ;
6D93 0A       1868          ASL               ;SAME AS XMUL
6D94 AA       1869          TAX               ;
6D95 60       1870          RTS               ;
6D96 C900     1871  NEGAD   CMP #$0           ;GOING BAC*WARDS
6D98 3003     1872          BMI NRSLT         ;
6D9A 4C866D   1873          JMP GONE          ;
6D9D 18       1874  NRSLT   CLC               ;
6D9E 6946     1875          ADC #$46          ;HARDSHIP CASES, LAST
6DA0 1006     1876          BPL FIX           ;CHANCE TO PROVE THEMSELVES
6DA2 DE4A72   1877          DEC TXEXT,X       ;
6DA5 18       1878          CLC               ;
6DA6 6946     1879          ADC #$46          ;
6DA8 DE4A72   1880  FIX     DEC TXEXT,X       ;HITTING LEFT SIDE?
6DAB 10D9     1881          BPL GONE          ;
6DAD A900     1882          LDA #$0           ;
6DAF 9D4A72   1883          STA TXEXT,X       ;
6DB2 A080     1884          LDY #$80          ;
6DB4 4C866D   1885          JMP GONE          ;
6DB7          1886  ;                         ;
6DB7          1887  ;                         ;
6DB7          1888  ;                         ;
6DB7          1889  ;                         ;** YMOVE DIDDLES TYPOS,X
```

```
6DB7          1889  ;                          ;** YMOVE DIDDLES TYPOS,X
6DB7 8D3672   1890  YMOVE   STA TEMP1           ;SAVE DISTENCE BYTE
6DBA 8A       1891          TXA                ;
6DBB 4A       1892          LSR                ;SAME AS XDIV
6DBC AA       1893          TAX                ;
6DBD BD4E72   1894          LDA TYPOS,X        ;
6DC0 8D3772   1895          STA TEMP2          ;SAVE IT BECAUSE TEST MODIFIES
6DC3 A000     1896          LDY #$0            ;
6DC5 18       1897          CLC                ;
6DC6 6D3672   1898          ADC TEMP1          ;THIS DOES A TEST FOR
6DC9 2E3772   1899          ROL TEMP2          ;OUT OF RANGE COMBOS
6DCC 900C     1900          BCC MEDM           ;BY TESTING SIGN BITS IN A
6DCE 2E3772   1901          ROL TEMP2          ;FAST SEQUENCE OF SHIFTS AND
6DD1 9021     1902          BCC SAFE2          ;BCC, BCS INSTRUCTIONS
6DD3 2E3672   1903          ROL TEMP1          ;
6DD6 B015     1904          BCS OUT2           ;
6DD8 9026     1905          BCC OUT3           ;
6DDA 2E3672   1906  MEDM    ROL TEMP1          ;
6DDD 900E     1907          BCC OUT2           ;
6DDF C900     1908  SAFE1   CMP #$0            ;NEGATIVE TYPOS MEANS OUT OF RNG
6DE1 100A     1909          BPL OUT2           ;ITS FINE
6DE3 A900     1910  OUT0    LDA #$0            ;HIT THE SIDE
6DE5 A080     1911          LDY #$80           ;SET FLAG
6DE7 9D4E72   1912          STA TYPOS,X        ;
6DEA BD4E72   1913  OUT1    LDA TYPOS,X        ;DUMMY LOAD, FOR NEXT INST.
6DED 9D4E72   1914  OUT2    STA TYPOS,X        ;STORE YPOSITION
6DF0 8A       1915          TXA                ;
6DF1 0A       1916          ASL                ;SAME AS XMUL
6DF2 AA       1917          TAX                ;
6DF3 60       1918          RTS                ;
6DF4 9D4E72   1919  SAFE2   STA TYPOS,X        ;
6DF7 2A       1920          ROL                ;CHEX FOR BOTTOM OF SCREEN
6DF8 90F0     1921          BCC OUT1           ;
6DFA 2A       1922          ROL                ;CAN BE FOOLED WITH
6DFB 90ED     1923          BCC OUT1           ;BIG DISTANCES
6DFD 2A       1924          ROL                ;
6DFE F0EA     1925          BEQ OUT1           ;
6E00 A9C0     1926  OUT3    LDA #$C0           ;BOTTOM OF SCREEN
6E02 A080     1927          LDY #$80           ;
6E04 D0E7     1928          BNE OUT2           ;
6E06          1929          ICL "REST"         ;LOAD NEXT PROGRAM SEGMENT
                                               ;
BLOAD REST                                     ;
                                               ;CONVERTS A BINARY IMAGE TO
                                               ;A LINE IMAGE BY HITTING EDGES
6E06 205960      1  EDGEDO JSR LITE           ;TURN ON SCREEN
6E09 A200        2          LDX #$0            ;SET INDEX
6E0B 202769      3          JSR SBLANX         ;CLEAR SECONDARY SCREEN
6E0E A903        4          LDA #$3            ;FUNCTION WORD FOR EDGE
6E10 8D2372      5          STA PROCED         ;STORED IN PROCED
6E13 A960        6          LDA #$60           ;PAGE SET FOR DOING SECONDARY PAGE
6E15 8D2172      7          STA PAGE           ;ONE SET OF EDGES GOES THERE
6E18 A901        8          LDA #$1            ;DO LEFT TO RIGHT AND DOWN SCAN
6E1A 20C96E      9          JSR SCAN           ;DOING EDGES AS YOU GO
6E1D A900       10          LDA #$0            ;WRITE TO PRIMARY PAGE THIS TIME
6E1F 8D2172     11          STA PAGE           ;
6E22 20C96E     12          JSR SCAN           ;DO DOWN AND TO RIGHT
6E25 20D26F     13          JSR MERGE          ;MERGE SECONDARY & PRIMARY ON PRIM
6E28 206960     14          JSR DAR            ;GOTO TEXT & BASIC
6E2B 60         15          RTS                ;VAMOOSE
6E2C            16  ;
```

```
6E2C 205960   17  SM1     JSR LITE        ;TURN ON SCREEN
6E2F A200     18          LDX #$0         ;AVERAGE THREE ADJACENT PIXELS
6E31 8A       19          TXA             ;DO LEFT TO RIGHT AND DOWN SCAN
6E32 8D2372   20          STA PROCED      ;
6E35 20C96E   21          JSR SCAN        ;
6E38 A901     22          LDA #$1         ;
6E3A 20C96E   23          JSR SCAN        ;DO DOWN AND RIGHT SCAN
6E3D A980     24          LDA #$80        ;
6E3F 20C96E   25          JSR SCAN        ;DO RIGHT TO LEFT AND UP SCAN
6E42 A9FF     26          LDA #$FF        ;
6E44 20C96E   27          JSR SCAN        ;DO UP AND TO LEFT SCAN
6E47 206960   28          JSR DAR         ;GOTO TEXT
6E4A 60       29          RTS             ;
6E4B          30  ;                       ;
6E4B 205960   31  SM2     JSR LITE        ;TURN ON GRAPHICS
6E4E A200     32          LDX #$0         ;
6E50 A901     33          LDA #$1         ;DO AVERAGE OF FIVE ADJACENT PIX
6E52 8D2372   34          STA PROCED      ;
6E55 8A       35          TXA             ;LEFT TO RIGHT AND DOWN SCAN
6E56 20C96E   36          JSR SCAN        ;
6E59 A901     37          LDA #$1         ;
6E5B 20C96E   38          JSR SCAN        ;DOWN AND RIGHT SCAN
6E5E 206960   39          JSR DAR         ;
6E61 60       40          RTS             ;
6E62          41  ;                       ;
6E62 205960   42  SM3     JSR LITE        ;
6E65 A200     43          LDX #$0         ;AVERAGE THREE ADJACENT PIXELS
6E67 8A       44          TXA             ;
6E68 8D2372   45          STA PROCED      ;
6E6B 20C96E   46          JSR SCAN        ;
6E6E A901     47          LDA #$1         ;OVER ALL SCAN MODES
6E70 20C96E   48          JSR SCAN        ;
6E73 A901     49          LDA #$1         ;
6E75 8D2372   50          STA PROCED      ;
6E78 A980     51          LDA #$80        ;
6E7A 20C96E   52          JSR SCAN        ;
6E7D A9FF     53          LDA #$FF        ;
6E7F 20C96E   54          JSR SCAN        ;
6E82 206960   55          JSR DAR         ;
6E85 60       56          RTS             ;
6E86          57  ;                       ;
6E86 205960   58  SM4     JSR LITE        ;
6E89 A900     59          LDA #$0         ;
6E8B AA       60          TAX             ;AVERAGE THREE PIXELS
6E8C 8D2372   61          STA PROCED      ;TWO SCAN MODES
6E8F 20C96E   62          JSR SCAN        ;
6E92 A901     63          LDA #$1         ;
6E94 20C96E   64          JSR SCAN        ;
6E97 206960   65          JSR DAR         ;
6E9A 60       66          RTS             ;
6E9B          67  ;                       ;
6E9B 205960   68  LAPLA1  JSR LITE        ;DO DERIVATIVE OVER TWO SCANS
6E9E A200     69          LDX #$0         ;
6EA0 A902     70          LDA #$2         ;
6EA2 8D2372   71          STA PROCED      ;
6EA5 A900     72          LDA #$0         ;
6EA7 20C96E   73          JSR SCAN        ;
6EAA A901     74          LDA #$1         ;
6EAC 20C96E   75          JSR SCAN        ;
6EAF 206960   76          JSR DAR         ;
6EB2 60       77          RTS             ;
```

```
6EB3 205960   78  LAPLA2 JSR LITE       ;DO OPPOSITE ORDER OF DERIVATIVE
6EB6 A902     79         LDA #$2        ;ON SAME TWO SCANS
6EB8 8D2372   80         STA PROCED     ;
6EBB A901     81         LDA #$1        ;
6EBD 20C96E   82         JSR SCAN       ;
6EC0 A900     83         LDA #$0        ;
6EC2 20C96E   84         JSR SCAN       ;
6EC5 206960   85         JSR DAR        ;
6EC8 60       86         RTS            ;
6EC9          87  ;                     ;
6EC9          88  ;                     ;
6EC9          89  ;                     ;
6EC9          90  ;                     ;
6EC9          91  ;                     ;
6EC9          92  ;                     ;
6EC9 8D2972   93  SCAN   STA MOVMOD     ;SAVE SCAN MODE
6ECC A200     94         LDX #$0        ;INDEX
6ECE 204F69   95         JSR ADSTRT     ;GET STARTING ADDRESSES (CORNER)
6ED1 202D68   96         JSR PSAVE      ;SAVE IT ON STAX
6ED4 A202     97         LDX #$2        ;
6ED6 200265   98         JSR TRANS      ;TRANSFER IT TO OTHER LOCATION
6ED9 A202     99  LINES  LDX #$2        ;LEAD CURSOR INDEX = 2
6EDB 20BE68  100         JSR GETVAL     ;GET PIXEL VALUE
6EDE 8D2472  101         STA PIXEL0     ;LOAD UP WITH FIRST PIXEL IN LINE
6EE1 8D2572  102         STA PIXEL1     ;
6EE4 8D2672  103         STA PIXEL2     ;
6EE7 2902    104         AND #$2        ;EXCLUSIVELY FOR EDGE ROUTINE,
6EE9 F002    105         BEQ MOVIT      ;IT HAS TO NOW START COLOR
6EEB A980    106         LDA #$80       ;
6EED 8D2272  107  MOVIT  STA EULER      ;
6EF0 209469  108         JSR MOVE1      ;MOVE LEAD CURSOR ALONG LINE
6EF3 20BE68  109         JSR GETVAL     ;GET PIXEL VALUE AFTER REAL CURSOR
6EF6 8D2772  110         STA PIXEL3     ;
6EF9 A202    111  PIXEL  LDX #$2        ;INDEX FOR LEAD CURSOR
6EFB 209469  112         JSR MOVE1      ;
6EFE A202    113         LDX #$2        ;GET LAST PIXEL VALUE IN GROUP OF5
6F00 20BE68  114         JSR GETVAL     ;
6F03 8D2872  115         STA PIXEL4     ;
6F06 A200    116         LDX #$0        ;INDEX FOR CURSOR THAT MODIFYS
6F08 203F6F  117         JSR FUNCT      ;SCREEN PAGES WITH FUNCT(PROCED)
6F0B A200    118         LDX #$0        ;MOVE REAL CURSOR
6F0D 209469  119         JSR MOVE1      ;
6F10 C080    120         CPY #$80       ;END OF LINE?
6F12 F009    121         BEQ LINEND     ;
6F14 A200    122         LDX #$0        ;NEXT PIXEL IN LINE
6F16 20336F  123         JSR RELOAD     ;SHIFT PIXEL VALUES AROUND
6F19 A200    124         LDX #$0        ;
6F1B 4CF96E  125         JMP PIXEL      ;
6F1E A200    126  LINEND LDX #$0         ;
6F20 20AB69  127         JSR MOVE2      ;GET ADDRESS FOR NEXT LINE
6F23 C080    128         CPY #$80       ;END OF FRAME?
6F25 F008    129         BEQ DONE       ;
6F27 A202    130         LDX #$2        ;NO, TRANSFER ADDRESSES
6F29 200265  131         JSR TRANS      ;
6F2C 4CD96E  132         JMP LINES      ;DO NEXT LINE
6F2F 206568  133  DONE   JSR PRESTO     ;RESTORE ADDRESSES ETC - NOT
6F32 60      134         RTS            ;REALLY NEEDED
6F33         135  ;                     ;;
6F33         136  ;                     ;
6F33         137  ;                     ;
6F33         138  ;                     ;
```

```
6F33            139  ;                              ;
6F33 BD2572     140  RELOAD LDA PIXEL1,X             ;MOVE EVERY PIXEL VALUE UP
6F36 9D2472     141       STA PIXEL0,X              ;ONE POSITION, GETTING RID
6F39 E8         142       INX                       ;OF ONE
6F3A E004       143       CPX #$4                   ;
6F3C D0F5       144       BNE RELOAD                ;
6F3E 60         145       RTS                       ;
6F3F            146  ;                              ;
6F3F            147  ;                              ;
6F3F            148  ;                              ;
6F3F            149  ;                              ;
6F3F AC2372     150  FUNCT  LDY PROCED               ;TEST PROCED FOR FUNCTION
6F42 F00A       151       BEQ AVG3                  ;
6F44 88         152       DEY                       ;ANY NUMBER OF FUNCTIONS CAN
6F45 F017       153       BEQ AVG5                  ;
6F47 88         154       DEY                       ;USE SCAN TO DO OPERATIONS ON
6F48 F035       155       BEQ DERIV                 ;
6F4A 88         156       DEY                       ;SCREENS WITH SOME SCANNING
6F4B F043       157       BEQ EDGE                  ;
6F4D 60         158       RTS                       ;DIRECTION
6F4E            159  ;                              ;
6F4E            160  ;                              ;
6F4E            161  ;                              ;
6F4E AD2672     162  AVG3   LDA PIXEL2               ;DOES WEIGHTED AVERAGE OF THREE
6F51 0A         163       ASL                       ;
6F52 6D2572     164       ADC PIXEL1                ;PIXEL VALUES
6F55 6D2772     165       ADC PIXEL3                ;
6F58 4A         166       LSR                       ;
6F59 4A         167       LSR                       ;
6F5A 20EC68     168       JSR STRVAL                ;THEN STORES VALUE IN CENTRE PIXEL
6F5D 60         169       RTS                       ;
6F5E            170  ;                              ;
6F5E            171  ;                              ;
6F5E            172  ;                              ;
6F5E AD2672     173  AVG5   LDA PIXEL2               ;DOES STRAIGHT AVERAGE OF FIVE
6F61 6D2772     174       ADC PIXEL3                ;
6F64 6D2572     175       ADC PIXEL1                ;PIXELS, THEN STORES IN CENTRE
6F67 6D2872     176       ADC PIXEL4                ;
6F6A 6D2472     177       ADC PIXEL0                ;
6F6D 0A         178       ASL                       ;
6F6E 20D56C     179       JSR DIV10                 ;
6F71 C905       180       CMP #$5                   ;ROUND OFF TO NEAREST NUMBER
6F73 3003       181       BMI SMLREM                ;
6F75 EE3172     182       INC DVD                   ;
6F78 AD3172     183  SMLREM LDA DVD                  ;
6F7B 20EC68     184       JSR STRVAL                ;STORE AVERAGE IN PIXEL
6F7E 60         185       RTS                       ;
6F7F            186  ;                              ;
6F7F            187  ;                              ;
6F7F            188  ;                              ;
6F7F AD2672     189  DERIV  LDA PIXEL2               ;DOES POSITIVE DERIVATIVE
6F82 F008       190       BEQ DOUT                  ;
6F84 38         191       SEC                       ;ACROSS LINE
6F85 ED2572     192       SBC PIXEL1                ;
6F88 1002       193       BPL DOUT                  ;WRITES ZERO FOR NEGATIVE DERIV
6F8A A900       194       LDA #$0                   ;
6F8C 20EC68     195  DOUT   JSR STRVAL               ;NEEDS MORE BITS TO PLAY WITH
6F8F 60         196       RTS                       ;
6F90            197  ;                              ;
6F90            198  ;                              ;
6F90            199  ;                              ;
```

```
6F90 AD2272  200  EDGE   LDA EULER      ;DETECT EDGE, BUT WHAT TYPE?
6F93 1018    201         BPL OFF        ;EULER=0 FOR DARK BAC*GROUND
6F95 AD2772  202  ON     LDA PIXEL3     ;
6F98 2902    203         AND #$2        ;TEST FOR PRIMARY PAGE PIXEL ON
6F9A D00B    204         BNE BLANZ      ;IGNORES SECONDARY PAGE (WEIGHT 1)
6F9C A902    205         LDA #$2        ;THERE'S AN EDGE, WRITE TO SCREEN
6F9E 20BD6F  206         JSR SECWR      ;
6FA1 A900    207         LDA #$0        ;RESET EULER
6FA3 8D2272  208         STA EULER      ;
6FA6 60      209  SAMEAS RTS            ;
6FA7 A900    210  BLANZ  LDA #$0        ;
6FA9 20BD6F  211         JSR SECWR      ;ERASE SCREEN
6FAC 60      212         RTS            ;
6FAD AD2672  213  OFF    LDA PIXEL2     ;ON PIXELS=EDGE
6FB0 2902    214         AND #$2        ;
6FB2 F0F2    215         BEQ SAMEAS     ;
6FB4 20BD6F  216         JSR SECWR      ;WRITE EDGE
6FB7 A980    217         LDA #$80       ;
6FB9 8D2272  218         STA EULER      ;SET EULER FOR BRIGHT REGION
6FBC 60      219         RTS            ;
6FBD         220  ;                     ;
6FBD         221  ;                     ;
6FBD         222  ;                     ;
6FBD 4A      223  SECWR  LSR            ;WRITES TO PRIMARY OR SECONDARY
6FBE 8D3672  224         STA TEMP1      ;PAGE, DEPENDING ON VALUE OF
6FC1 B507    225         LDA ADDRH1,X   ;PARAMETER PAGE
6FC3 8D3772  226         STA TEMP2      ;
6FC6 4D2172  227         XOR PAGE       ;
6FC9 200769  228         JSR WRITE      ;
6FCC AD3772  229         LDA TEMP2      ;
6FCF 9507    230         STA ADDRH1,X   ;
6FD1 60      231         RTS            ;
6FD2         232  ;                     ;
6FD2         233  ;                     ;
6FD2         234  ;                     ;
6FD2 A900    235  MERGE  LDA #$0        ;MERGE TWO PAGES INTO PRIMARY
6FD4 8D2972  236         STA MOUMOD     ;
6FD7 A200    237         LDX #$0        ;
6FD9 204F69  238         JSR ADSTRT     ;USES SIMPLE ADDRESSING
6FDC B507    239         LDA ADDRH1,X   ;
6FDE 4960    240         XOR #$60       ;
6FE0 9507    241         STA ADDRH1,X   ;
6FE2 A202    242         LDX #$2        ;
6FE4 204F69  243         JSR ADSTRT     ;
6FE7 A000    244         LDY #$0        ;
6FE9 A106    245  MLOPS  LDA (ADDRL1,X);
6FEB 1106    246         ORA (ADDRL1),;
6FED 8106    247         STA (ADDRL1,X);
6FEF E606    248         INC ADDRL1     ;
6FF1 F606    249         INC ADDRL1,X   ;
6FF3 D0F4    250         BNE MLOPS      ;
6FF5 E607    251         INC ADDRH1     ;
6FF7 F607    252         INC ADDRH1,X   ;
6FF9 B507    253         LDA ADDRH1,X   ;
6FFB C940    254         CMP #$40       ;
6FFD D0EA    255         BNE MLOPS      ;
6FFF 60      256         RTS            ;
7000         257  ;                     ;
7000         258  ;                     ;
7000         259  ;                     ;
7000         260  ;                     ;;
```

```
7000 FF0101  263  XVECT  HEX FF0101FF
7003 FF
7004 FFFF01  264  YVECT  HEX FFFF0101
7007 01
7008 050302  265  XSTAIR HEX 05030201010
10101
700B 010101
700E 0101
7010 060403  266  YSTAIR HEX 06040302040
81020
7013 020408
7016 1020
7018 070503  267  ISTEP  HEX 07050302040
70F1F
701B 020407
701E 0F1F
7020 A00070  268  FSTEP  HEX A00070AC0F2
02010
7023 AC0F20
7026 2010
7028 002850  269  YSTRL  HEX 002850
702B 000A14  270  XSTR   HEX 000A141E
702E 1E
702F 010204  271  MSK    HEX 01020408102
040
7032 081020
7035 40
7036 3870    272  MESTAB ADR MESSGE
7038 4770    273  MESSGE ADR MESS1
703A 05      274         HEX 05
703B E970    275         ADR MESS2
703D 06      276         HEX 06
703E 9F71    277         ADR MESS3
7040 01      278         HEX 01
7041 A871    279         ADR MESS4
7043 02      280         HEX 02
7044 DF71    281         ADR MESS5
7046 02      282         HEX 02
7047 1EA0A0  283  MESS1  STR "   COMMAND
S BEGIN WITH A SPACE"
704A A0C3CF
704D CDCDC1
7050 CEC4D3
7053 A0C2C5
7056 C7C9CE
7059 A0D7C9
705C D4C8A0
705F C1A0D3
7062 D0C1C3
7065 C5
7066 14C8A0  284         STR "H  KEY-STR
OKE CURSOR"
7069 A0CBC5
706C D9ADD3
706F D4D2CF
7072 CBC5A0
7075 C3D5D2
7078 D3CFD2
707B 25CDC7  285         STR "MG FIND FU
LL EDGE/ SG FIND SIDE EDGES"
707E A0C6C9
7081 CEC4A0
7084 C6D5CC
7087 CCA0C5
708A C4C7C5
708D 0F0AD3
```

```
7093 C9CEC4
7096 A0D3C9
7099 C4C5A0
709C C5C4C7
709F C5D3
70A1 26D4C7  286          STR "TG FIND TO
P EDGE/  BG FIND BOTTOM EDGE"
70A4 A0C6C9
70A7 CEC4A0
70AA D4CFD0
70AD A0C5C4
70B0 C7C5AF
70B3 A0A0C2
70B6 C7A0C6
70B9 C9CEC4
70BC A0C2CF
70BF D4D4CF
70C2 CDA0C5
70C5 C4C7C5
70C8 20A4A0  287          STR "$  CLEAR B
UFFER/  X  SEND BUFFER"
70CB A0C3CC
70CE C5C1D2
70D1 A0C2D5
70D4 C6C6C5
70D7 D2AFA0
70DA A0D8A0
70DD A0D3C5
70E0 CEC4A0
70E3 C2D5C6
70E6 C6C5D2
70E9 1DA0CB  288  MESS2  STR " KEY-STROK
E MOVE COMMANDS ARE"
70EC C5D9AD
70EF D3D4D2
70F2 CFCBC5
70F5 A0CDCF
70F8 D6C5A0
70FB C3CFCD
70FE CDC1CE
7101 C4D3A0
7104 C1D2C5
7107 18A0A0  289          STR "      J
 JOYSTICK MODE"
710A A0A0A0
710D A0CAA0
7110 A0A0A0
7113 CACFD9
7116 D3D4C9
7119 C3CBA0
711C CDCFC4
711F C5
7120 1DA0A0  290          STR "      U (T
WO DIGITS)  MOVE UP"
7123 A0A0A0
7126 A0D5A0
7129 A8D4D7
712C CFA0C4
712F C9C7C9
7132 D4D3A9
7135 A0A0CD
7138 CFD6C5
713B A0D5D0
713E 1FA0A0  291          STR "      D (T
WO DIGITS)  MOVE DOWN"
7141 A0A0A0
```

```
7147 A8D4D7
714A CFA0C4
714D C9C7C9
7150 D4D3A9
7153 A0A0CD
7156 CFD6C5
7159 A0C4CF
715C D7CE
715E 20A0A0  292          STR "      R (T
WO DIGITS)  MOVE RIGHT"
7161 A0A0A0
7164 A0D2A0
7167 A8D4D7
716A CFA0C4
716D C9C7C9
7170 D4D3A9
7173 A0A0CD
7176 CFD6C5
7179 A0D2C9
717C C7C8D4
717F 1FA0A0  293          STR "      L (T
WO DIGITS)  MOVE LEFT"
7182 A0A0A0
7185 A0CCA0
7188 A8D4D7
718B CFA0C4
718E C9C7C9
7191 D4D3A9
7194 A0A0CD
7197 CFD6C5
719A A0CCC5
719D C6D4
719F 08A0A0  294  MESS3   STR "  MISSES"
71A2 CDC9D3
71A5 D3C5D3
71A8 15AAAA  295  MESS4   STR "***  BUFFE
R FULL  ***"
71AB AAA0A0
71AE C2D5C6
71B1 C6C5D2
71B4 A0C6D5
71B7 CCCCA0
71BA A0AAAA
71BD AA
71BE 20A7A4  296          STR "'$' CLEARS
    'X' SENDS TO BASIC"
71C1 A7A0C3
71C4 CCC5C1
71C7 D2D3A0
71CA A0A0A0
71CD A7D8A7
71D0 A0D3C5
71D3 CEC4D3
71D6 A0D4CF
71D9 A0C2C1
71DC D3C9C3
71DF 10CECF  297  MESS5   STR "NO EDGE DE
TECTED"
71E2 A0C5C4
71E5 C7C5A0
71E8 C4C5D4
71EB C5C3D4
71EE C5C4
71F0 11D2C5  298          STR "REPOSITION
```

```
71F9 CFCEA0
71FC C3D5D2
71FF D3CFD2
7202 6373     299  ST1     ADR XATS
7204 6383     300  BUF1    ADR BIGBUF
7206 00       301           HEX 00
7207 00       302  MASS    HEX 00
7208 00       303  NUMHI   HEX 00
7209 00       304  NUMLO   HEX 00
720A 0A       305  WIDTH   HEX 0A
720B 00       306  WINDO1  HEX 00
720C 00       307  WINDO2  HEX 00
720D 00       308  TRAVEL  HEX 00
720E 00       309  REFUND  HEX 00
720F 00       310  FAULTS  HEX 00
7210 00       311  DISTAN  HEX 00
7211 00       312  LOSTEP  HEX 00
7212 00       313  HISTEP  HEX 00
7213 00       314  OFRACT  HEX 00
7214 00       315  OFFSET  HEX 00
7215 00       316  ENDPT   HEX 00
7216 00       317  QUAD    HEX 00
7217 00       318  QUID    HEX 00
7218 00       319  LQUID   HEX 00
7219 00       320  XDIR    HEX 00
721A 00       321  YDIR    HEX 00
721B 00       322  STEPS   HEX 00
721C 00       323  RAYOFF  HEX 00
721D 00       324  XSTEP   HEX 00
721E 00       325  XFRACT  HEX 00
721F 00       326  YSTEP   HEX 00
7220 00       327  YFRACT  HEX 00
7221 00       328  PAGE    HEX 00
7222 00       329  EULER   HEX 00
7223 00       330  PROCED  HEX 00
7224 00       331  PIXEL0  HEX 00
7225 00       332  PIXEL1  HEX 00
7226 00       333  PIXEL2  HEX 00
7227 00       334  PIXEL3  HEX 00
7228 00       335  PIXEL4  HEX 00
7229 00       336  MOVMOD  HEX 00
722A 00       337  VECTOR  HEX 00
722B 00       338  FIRST   HEX 00
722C 00       339  SECOND  HEX 00
722D 00       340  THIRD   HEX 00
722E 00       341  DVSR    HEX 00
722F 00       342  TEMDVD  HEX 00
7230 00       343  REM     HEX 00
7231 00       344  DVD     HEX 00
7232 00       345  DSR     HEX 00
7233 00       346  MOD     HEX 00
7234 00       347  RESLT   HEX 00
7235 00       348  NUMBER  HEX 00
7236 00       349  TEMP1   HEX 00
7237 00       350  TEMP2   HEX 00
7238 00       351  TEMP3   HEX 00
7239 00       352  TEMP4   HEX 00
723A 00       353  TEMP5   HEX 00
723B 00       354  BRUSH   HEX 00
723C 00       355  FLAG    HEX 00
723D 00       356  BLTFLG  HEX 00
```

```
723E 00       357  CFLAG  HEX 00
723F 00       358  PGFLAG HEX 00
7240 00       359  PFLAG  HEX 00
7241 00       360  MFLAG  HEX 00
7242 00       361  OFLAG  HEX 00
7243 00       362  MOVE   HEX 00
7244 00       363  AXIS   HEX 00
7245 00       364  CMMND  HEX 00
7246 00       365  PSOURC HEX 00
7247 00       366  MOVX   HEX 00
7248 00       367  MOVY   HEX 00
7249 00       368  MODE   HEX 00
1A4E          369  TXEXT  DFS 4
1A52          370  TYPOS  DFS 4
1A56          371  TXPOS  DFS 4
7256 00       372  MASK   HEX 00
1A5E          373  XTEMP  DFS 7
725E 00       374  XEXT   HEX 00
725F 00       375  YPOS   HEX 00
7260 00       376  XPOS   HEX 00
7261 00       377  COUNT  HEX 00
7262 00       378  ITCNT  HEX 00
1B63          379  SPACE  DFS $100
2B63          380  XATS   DFS $1000
2C63          381  BIGBUF DFS $100
```