5-27-1986

Micro computer animation

Peter Sterling Franks

ROCHESTER INSTITUTE OF TECHNOLOGY

A THESIS SUBMITTED TO THE FACULTY OF THE COLLEGE OF FINE AND APPLIED ARTS IN CANDIDACY FOR THE DEGREE OF MASTER OF FINE ARTS

MICRO COMPUTER ANIMATION

BY PETER STERLING FRANKS

MAY 27, 1986

Approvals

Chief Advisor: Prof. James Ver Hague

James Ver Hague Date: 10/30/86

Associate Advisor: Prof. Robert Keough

Robert Keough Date: 11-11-86

Associate Advisor: Prof. Guy Johnson

Guy Johnson Date: 11-21-86

Special Assistant to the
Dean for Graduate Affairs: Prof. Philip Bornarth

Philip Bornarth Date: 12/10/86

Dean, College of
Fine and Applied Arts: Dr. Robert Johnston

Dr. Robert H. Johnston Ph.D. Date: 12/15/86

I, Peter Franks, insist to be contacted each time a request for publication is made.

I can be reached at the following address:

Peter Franks

_____ Date: _____

Thesis Committee

Chief Advisor:	James Ver Hague
Associate Advisor:	Robert Keough
Associate Advisor:	Guy Johnson

Acknowledgments

Special thanks should go to Danny LePage for his time and effort in providing critical information for the completion of my thesis. I would like to thank Jim Ver Hague, Robert Keough and Guy Johnson for their patience and support throughout this entire project.

Table of Contents

List of Illustrations

List of Appendices

Introduction

The purpose of my thesis is to investigate the possibilities of creating a two-dimensional and a three-dimensional computer animation program for the Artronics Computer Graphic System. This thesis will develop the ground work for future development in bit map computer graphics at RIT, enabling the designer to have much more flexibility in creating imagery through programming. The thesis may be described as a technical thesis, but the graphic designer is the most important benefactor.

The two basic types of computer animation are raster or bit map graphics and vector graphics. Vector graphics is the most widely used type of computer animation used in the commercial industry. The problem with vector graphics is hardware. Vector computer graphics hardware is very expensive, but requires no more extra database than raster graphics. The effects of vector graphics can be spectacular. Raster graphics was introduced in the mid-seventies. The advantage of raster graphics is the refresh buffers which enable an image to be handled with relatively inexpensive hardware, and the use of standard television technology. The Artronics Computer Graphics System is a raster computer graphics system and is a fairly inexpensive system.

The Artronics Computer Graphic system has been developed around the IBM personal computer. This particular system is based on the Leading Edge personal computer, which is comparable to the IBM PC. My thesis develops around the hypothesis that I should be able to combine the programming of the Leading Edge and the PCPAINT program of Artronics. This hypothesis turned into a very complicated situation which took over

five months of technical research. There were two critical aspects to the entire project. The first was the ability to access 256 colors in the PCPAINT program of the Artronics through programming. Most personal computers only have a maximum of 32 colors. The ability to provide 256 colors to a designer enables the designer to have a much wider variety of possibilities. The other crucial aspect is the possibility of realtime animation or frame-by-frame animation. Frame-by-frame animation is not particularly difficult to create until the exclusive (XOR) function is implemented. XOR is the fundamental concept in the development of bit map animation. The XOR or complement mode is useful when wishing to draw an image temporarily, and removing the image leaving the original image. This can be accomplished by writing the image twice in complement mode, since the complement of the complement of a color is the original color. The XOR function is extremely important for realtime animation and will be the determining factor for animation on the Artronics. The idea is to create an image with the Artronics PCPAINT program and then animate the object over a background created in the PCPAINT program.

There is a great deal of technical research and development which must be investigated. I will provide a fundamental foundation for the further development of bit map graphics. My thesis is meant to encourage designers to realize the possibilities of bit graphics, and to demonstrate that programming can be and should be a major consideration in the education of the computer graphic designer.

Technical Research

A. Programming Structures and Sprites

It is extremely difficult to develop an animation software package from scratch. Most companies or educational institutions have teams of programmers that spend long periods of time on the development of just one software package. The time constraints for the completion of this thesis made it extremely difficult for one person to create a software package. The logical alternative was to begin my investigation with existing software.

It was necessary to choose a starting point which would provide a solid foundation for the research. The Apple IIe supports an animation software package called TAKE 1. This software enables a designer to create two dimensional animation. The program is fairly well organized and easy to learn. The program seemed ideal for the Artronics.

The one main feature of the TAKE 1 animation package is that it uses a computer graphic function called exclusive OR. This function enables the designer to define an area around an object. Once the object has been defined, it turns the area around the object into a transparent overlay. The animation effect is then achieved by placing and moving the object over a background. It is a marvelous function which allows a designer to create many types of two-dimensional animation.

The TAKE 1 program has another strength besides exclusive OR. The program menus are straightforward and readable. This makes it easy for the designer to flip through the menus and run the program. The program enables the designer to create actors, backgrounds, screens and movies fairly quickly. I believe the TAKE 1 program

would be a very impressive software package if it could run on the Artronics.

There are a couple of drawbacks with the TAKE 1 program, which could be resolved on the Artronics. The first drawback is the number of colors. The Apple TAKE 1 program provides the designer with only sixteen colors. This is very limited compared to the 256 colors of the Artronics. The other problem with the Apple is low resolution. The Apple IIe has a resolution of 280 x 192 whereas the Artronics has a resolution of 512 x 480. The combination of greater color capabilities and resolution on the Artronics would provide the designer with a tool to create complicated images and animation.

Most computer animation software packages are written in either "C" or assembly language. C programming language is relatively new to computer graphics, but it is becoming the language of computer graphics. Assembly language is often used in writing computer programs. This enables the programmer to calculate large and complex mathematical routines in the matter of a split second. Assembly language enables a program to be quick and efficient. The problem with assembly language is the code can become extremely long. This code makes it difficult for an inexperienced computer programmer to debug. Time contraints dictated that the time be spent in a more productive manner. I decided to try and locate some essential parts in the program. There are subroutines in TAKE 1 which define an area for animation and the exclusive OR of that area. TAKE 1 refers to these functions as Sprites. The Sprite subroutine could be interpreted into an understandable syntax and rewritten for the Artronics. I was unable to locate any of the subroutines for Sprites. A programmer and software company try to make it as difficult as possible for someone else to pirate their software.

Sprites are technically the area which is being XORed (exclusive OR). The process of performing the sprite function becomes much more complicated. The process involved in running the actual sprite is technically difficult on the Artronics. The subroutine code in assembly language on the Apple activates the display buffer and color register in memory. The color register and display buffer are stored in two different locations in memory. Once the information has been stored in these two locations, a "black box" is needed to execute both memory locations and plot the points on the screen. The entire process will not work without this "black box". The "black box" acts as a device that links in memory the color register and display buffer. Therefore, not only is the sprite subroutine needed to run the program, but also the black box to display the visual information.

Sprites are very common with the Commodore Computer System. I researched the possibilities of converting the Commodore sprites into a code which can be used on the Artronics. Commodore publishes many books on how to program sprites and create animation. The problem with these sprites is that the Commodore computer is specially designed to handle the commands for sprites. Commodore does not release the information on how to structure the hardware and software to accept the commands for programming sprites. Therefore, it is not feasible to use Commodore sprite functions in creating sprites.

The original concept of developing an Artronics animation software package based on the Apple Computer and TAKE 1 proved unworkable. I concluded that it would take a computer science software programmer to develop animation using this process of converting TAKE 1 to the Artronics. The next step was to investigate the possibilities

of programming directly on the Artronics and to find which language would be most suitable for animation on the Artronics.

B. Programming Languages

The programming language is critical to the development of an animation program. The speed of drawing and redrawing an image with XOR is the significant factor for realtime animation. Realtime animation becomes increasingly difficult when combining hardware and software capabilities. The next step was to find software that is compatible with the Leading Edge computer. The Artronics has a Number Nine Computer board built into the Leading Edge. A programming language that is most compatible with the Number Nine board had to be chosen. The options are assembly language, BASIC or C. They all have strengths and weaknesses when conforming to the Leading Edge.

The same difficulties exist with assembly language as encountered with debugging the TAKE 1 animation program. It would take some time to learn and write assembly programs. Assembly would also be difficult to compile and run with the Artronics. The most exciting possibility of assembly is the speed at which the program could run. This is possible due to the ability of assembly commands to pick the functions right off of the Number Nine board and would probably give us realtime animation.

C is the most popular computer graphics language being used today. C is a fairly easy programming language to learn and has possibilities of producing realtime animation. The problem with C from a programming aspect is that it allows for sloppy programming and difficult debugging. C is the language which must be used in future pro-

gramming on the Artronics. Danney LePage has developed a C editor and compiler for the Artronics. It is fairly easy to write, compile and run programs. One drawback with the package is that the programmer must write out complete routines to perform functions such as drawing a line. These extra calculations waste valuable time for the purpose of animation especially when the functions become complex.

The final possibility is BASIC. The Leading Edge computer provides a basic package to program the Artronics. It is a standard BASIC editor and compiler. The major problem with this BASIC package is the same problem with C. The programmer must write out the entire routine to perform a function. Since we are dealing with BASIC, extremely slow graphics are produced. BASIC is a three step compiling process before graphics can be drawn on the screen. The BASIC program must first be compiled by the BASIC compiler, then transferred into assembly code, and finally the routines have to be accessed off of the Number Nine board. In addition, the program requires functions to be written out as a procedure. BASIC is not particularly well suited for animation, but BASIC is the ideal computer language for beginning programmers to learn the fundamentals of computer graphics programming.

C. Complications of Artronics

The final decision on the programming language depended on the internal structure of the Artronics. A former student was able to program on the Artronics. He was able to draw a line and change the color intensity of the palette. The only way this student was able to control the color palette and draw a line was with the help of Artronics,

Inc. Artronics gave this student the memory locations of the screen coordinates and the color palette in assembly code. This meant that his BASIC program was poking the locations on the board in creating imagery. This information implied that all the graphic routines must be on the board and all I needed to do was access these commands.

The Artronics runs off of an eight bit, gen lock, banked Number Nine board. This board has a capability of 256 displayable colors and runs on a 18.432 megahertz signal. This information became extremely important as the investigation developed. The next step was to contact the makers of the graphics board, the Number Nine Computer Company. I was hoping that Number Nine would provide valuable information as far as accessing routines off of the board. Number Nine would not release any information, but they suggested I speak with Media Cybernetics Corporation. Media Cybernetics creates software which is designed to interact with the Number Nine board.

Media Cybernetics makes a software package called Halo. Halo allows a programmer to program directly to a Number Nine board. The software package enables a programmer to use C or BASIC. The cost of the software is $300. The software package is supplied with one programming language and each additional programming language cost $150. Jim Ver Hague and I decided to purchase the BASIC package. For the time being, BASIC would have much more value to the Computer Graphics Design Programmer. The amount of speed lost from not using C would be offset in the educational value of BASIC to beginning programmers. This package is much quicker than the Leading Edge BASIC package and should provide exciting computer animation. The next step to investigate was the capabilities of the Number Nine board of the Artronics with Halo.

7

Adapting Halo for Artronics

The Halo software is rich with various types of features. For example, Halo allows a programmer to interface with a wide variety of input and output devices. Halo can provide both bitmapped and stroke lines, circles and text. This software allows the programmer to construct menus through window management. The commands are very simple to use in creating graphics. Halo provides a Learnhalo program which instructs the programmer on how to use the features of Halo. Unfortunately, the Learnhalo program is not presently compatible with the Artronics.

The Learnhalo program was loaded. The computer shut down when it attempted to run. This failure was definitely a serious problem. I decided to call Media Cybernetics, Artronics, and Number Nine for information. Each insisted the problem was with the other company. I finally deduced that the software was not compatible with the Number Nine board. This problem meant that the software had to be modified to conform with the board. This change led to the first of many major breakthroughs.

Meanwhile, another student happened to be working on direct-to-video capabilities for the Artronics. He was adapting a Lenco Sync Generator and a Color Encoder to the Artronics. This hardware was not compatible with the normal Number Nine board. The normal board does not have the video functions to enable the syncing of all the hardware. Artronics provided RIT with a new Number Nine board with the necessary video commands. The new board was installed and conformed to the Artronics hardware. While he was working on the direct-to-video, I was unsuccessfully trying to run the Learnhalo program on other hardware. As a last resort, I decided to try and run Learnhalo on

the new video Number Nine board. The program worked to a degree. There was an image on the screen, but it was completely distorted. The difference between the new and old board was the megahertz signal. The regular board runs on a 20 megahertz signal while the new board runs on a 18.432 megahertz signal.

Next, I concentrated on why the screen was being completely distorted. The image drawn by Learnhalo was flickering and being stretched across the screen on a diagonal. This seemed to be another serious problem. I consulted Artronics, Media Cybernetics, and the Number Nine Company. They were of no help and passed the blame on the other company. I then decided to get some outside consultation from a software programming specialist. Mr. Danny LePage is a Masters student at RIT in Computer Science. Danny has the knowledge of computer hardware and software to figure out almost any problem. Danny realized the problem immediately. The scan rate of the monitor was not in sync with the device driver and the Number Nine board. This problem could be solved if Artronics and Media Cybernetics cooperate with one another. First, I had to call Artronics and find out the sync parameters for the Artronics. After a struggle, Artronics finally released the parameters (See Appendix A). Next, Media Cybernetics had to reveal the location of the parameters in the device driver. The company refused to release this information. Obviously, Media Cybernetics did not want Danny or I to reprogram their device driver files. This meant that Danny had to take an extremely long shot to try to locate the parameters on the device driver. The Debug program of Leading Edge was used to break into the device driver program. Again, we seemed to get a lucky break. Danny was able to find the location of the parameters within an hour. Danny replaced the old

sync parameters with the new set of sync parameters given to us from Artronics. We were anxiously anticipating the outcome of the new sync parameters in the reconstructed device driver. We loaded and ran the Learnhalo program. Learnhalo ran without a hitch, drawing lines, circles and boxes in full color! This breakthrough was a major accomplishment and gave renewed hopes for the success of this project.

The next step was to create a BASIC program and then run the program. The Halo software comes with five disks. Each disk is critical in developing a working software package. The five disks are BASIC Programming disk, Learnhalo disk, a Locator disk, a Device Driver disk, and a Printer disk. Files from each of these disks must be put on to one disk. These files will be the basis for the software package. The programmer must figure out the device which is going to be programmed. The Artronics needs the device driver to access the Number Nine board. The file name is Halonine.dev and requires a drawing mode of thirteen. The locator must then be determined. The locators will activate all the different types of hardware being used by the system such as the tablet, pen and keyboard. Next, any printer output devices should be loaded onto the disk of the software package. There are many types of output devices provided which are optional to the programmer. The final files which must be loaded are the BASIC libraries and execution files. This BASIC software is an interpreted BASIC package. All files relating to the interpreted BASIC package must be loaded onto the software. This transformation input completes the transfering of all files for the software package. There is only one set of files left which must be transfered to the software package. The Halo software does not provide a BASIC editor. The Leading Edge BASIC Editor files must

be copied to the software package. It would be very convenient for the Artronics PCPAINT program to be on the software, but the disk does not have enough room. Now the Halo software and the Leading Edge BASIC Editor have been converted to one disk, which will be referred to as BasArt.

BasArt concluded five months of research and development in structuring a workable BASIC programing package for the Artronics. It would be simple to convert this BASIC software to a useable C software package. I find it amazing to realize the complexity involved in developing a software package for not only the Artronics, but any type of computer. The Artronics was finally ready to receive and run BASIC programs and eventually the Super Animation Program (SAP).

Creating an Animation Program for the Artronics

The Leading Edge BASIC editor has the same capabilities as any of the standard BASIC editors. There are no unusual differences between writing a Halo BASIC program and a standard BASIC program. It is important to plan an efficient BASIC program which will enable the program to run much smoother and create a quicker animation. The first test for animation was to see how fast a line could be drawn on the screen. The first program I wrote was a short linetest program (See Appendix B). This program will draw a line from one point to another. The speed of drawing was excitingly quick which led to another short line animation program. This program was developed to test the speed of drawing a large amount of lines and the color capabilities (See Appendix C). The speed again was very quick and gives the perception of realtime animation (See Figures 1, 2, 3). The color capability and control is wonderful. The programmer can access any of the 256 colors and even create their own color palette.

The next test program written was an animation of polygons transforming from one into another. This test program was to see the quickness of animating polygons and the ability to change colors. The actual animation decreased in speed, but was nothing to be concerned about for the time being. The test programs gave enough useful information to begin work on a major animation program in BASIC. The next program was Super Animation Program (SAP).

The program which I wrote combined a wide variety of operations. The fundamental program is a Three-Dimensional Wireframe Animation program with hidden line elimination. The program is frame-by-frame animation.

Realtime animation is impossible with BASIC. There is a strong possibility of realtime animation with C programming language. The three-demensional animation program provides myself and the first year computer graphics students with the capablities of animation on the Artronics. The first year majors are programming on the DEC PRO 350 with no color and a limited memory for creating objects. The Super Animation Program (SAP) will provide the students with 256 colors. Also, the students will be able to create more complicated shapes and a greater variety of objects. The program will enable me to study and create thumbnail drawings for my glass sculptures. The program will be able to XOR wireframe objects and areas on the screen. SAP will be an innovative program which will be a stepping stone to the development of bit map computer graphics on a personal computer.

The goal of the Super Animation Program is to make the program as user friendly as possible with enough flexibility for the designer to be creative. The program is developed around four major menus. A Shape Services, a Viewer Services, and Camera Services menus are included in SAP, while a fourth menu called Animation Services was created exclusively as a test menu for different types of animation (See Appendix E). Shape Services menu provides the user with such options as create shape, load shape, store shape and transform one shape to another. These options are the most difficult options to construct, but provide the most information as to the capabilites of the Artronics. The Viewer Services menu enables the user to change the viewing distance, move shape, scale shape, rotate shape, change any point to a new location, change line style, and change line width. These options all have frame-by-frame animation with straightfoward and readable instructions. The Camera Services menu provides all the movements of a camera mov-

ing around an object on the screen. The camera movements have the ability to spin, pan, tilt, and zoom. All the menus have the options of changing the background and object color, clearing screen, drawing shape and the flexibility to interchange between menus.

The program provides a list of shapes already developed by the program such as a cube and cross. The program provides the user with a list of all objects in present memory. The screen can display up to five shapes at a time. Presently, the program only allows a maximum of 55 points to be displayed at one time on the screen. Every option has complete instructions and restrictions of operation. Presently, the first year graduate students are using SAP to create a three dimensional wireframe animation. The students are learning how to run SAP and are providing valuable feedback. The students are finding the program fairly easy to run.

The animation menu activates some very special options. The menu allows the user to save and restore images only created with SAP. These routines can not save and restore images created in PCPAINT. The user at this time can not store and restore color palettes. It seems Halo and PC-PAINT color palettes are stored in two different locations. Another option in the Animation Menu is the exclusive OR of wireframe objects. The user is prompted to turn on and off the exclusive OR mode. There is only one problem with this feature. The object is drawn with planes and hidden line in SAP. A test is built into the program to determine whether or not a line should be drawn. The calculations of planes will cause one line to be drawn on top of another. This overlaping causes XOR to repeat the XOR mode one more time than it should, and therefore disrupts the XOR mode in animating lines. The other type of XOR is of an area. The user can define an area and move the area across the screen frame-by-frame. The XOR of an area works

in SAP over a black background. This malfunction is due to complex but solvable programming problems. One problem is the area being defined. The area must be properly defined after every frame of motion. Also, the object must be defined and redefined through the SET and RESET commands of Halo. These commands undoubtedly will slow the basic programs. C programming language would be crucial and may deliver realtime animation. Today, there are newer and faster computer chips in the marketplace such as the 8085, 8088/6, and 80186. These chips are easy to install on a board and rather inexpensive. These chips may produce realtime animation if programmed in C.

SAP allows the user to combine PCPAINT images and SAP images. It is possible to create an image in PCPAINT and then animate an object over top of the PCPAINT image. It is highly recommended to create palettes in PCPAINT before entering SAP. The user is able to specify a color by number and then draw with that color. There is a problem storing and recalling color palettes in SAP. This difficulty is a solvable problem which requires fundamental understanding of hardware and software programming.

SAP can be used for other creative arts besides animation. SAP could be used as a tool to design scuplture. I used SAP to design and study patterns which were used in designing scuplture. I was able to visually conceive line interactions within the piece, and then change a line or color in a matter of seconds. I believe this program could be a beginning of eventually combining computer graphics and other visual arts. The computer would act as a tool to the designer for visualization of a design before the actual design is created. SAP is designed to compliment the designer. Animation has a definite future with the Artronics. The beauty of the package is the ease in which a program can be written.

Documentation

The following documentation will explain how to write and run a BASIC program on the Artronics. This documentation will not explain all the functions which are available in the Halo reference manual. The Halo reference manual provides information on more sophisticated programming techniques such as a complete explanation of all the functions description, usage, and synopsis; and information on adapting more input and output devices. I will document the booting of software and the software package SAP, running the basic editor, constructing a BASIC program for visual output on Artronics and the possibilities of animation.

A. Booting System and Software

The programmer must have in his possession three disks: PCPAINT Video version, BasArt or Halo software, and SAP programming disk or blank formatted disk.

There are two ways of booting the Artronics for programming. The booting of the Artronics depends on whether the programmer wants the Halo color palette or a PCPAINT color palette.

1. The first method involves the use of the Halo color palette. The computer must be activated and the BasArt or Halo software should be placed in disk drive A. The computer will respond with A] in which the user should type:

A] Halorbi (return)

This command will access the Halo library which enables the programmer to use any of the Halo functions. The com-

puter will respond with a message informing the user the library is available. The A] will return and the programmer must enter the following to activate the basic editor.

A] Basic (return)

A message will be displayed welcoming the programmer to the BASIC editor. The computer is now ready to be programmed.

2. The other way of booting the software is with an Artronics color palette. The programmer must first boot the PCPAINT program. Now, the programmer should Clearscreen and activate the proper color palette. The screen should be cleared of the menu, brush, and if desired the color palette. Next, the programmer should press the following keys at the same time:

(CTRL) (ALT) (DEL)

The computer will reboot and the A] will return. The same procedure takes place as in booting the Halo software. The programmer must replace the PCPAINT software disk with the BasArt disk. The SAP program disk or a blank disk should be placed in disk drive B. Enter the following:

A] Halorbi (return)

A] Basic (return)

This command will boot the basic editor and prepare the programmer to write his own program or run SAP.

B. BASIC Editor

The BASIC editor is very easy to use. The editor has all the standard procedures of a BASIC editor and some

extra helpful commands. The keyboard has a set of function keys on the left side numbered F1 to F10. These keys are very useful and help decrease editing time. For example, F2 is a run command which runs a program when activated. F3 is a load command and F4 is a save command. The functions are listed on the bottom of the screen with the corresponding key number.

The only problem with the editor is that a programmer can only edit one line at a time. This task becomes difficult when a small correction has to be made in a list of twenty lines and each line has to be edited individually. Another problem area may arise when storing and recalling shapes. The programmer must be sure to describe the proper disk drive. A (B:) must be used to store or recall a program in disk drive B. There is no actual complier. The program will automatically compile and run with the RUN command. There should be no problems with the editor. There is a BASIC reference manual for the Leading Edge which will answer any questions.

C. Development of a Program Structure

Halo supplies a MEMORY.BAS program which identifies the values for the Clear and Def Seg statements to the interpreted BASIC package. These values will change depending on the hardware. The Def Seg and Clear values are needed to interface with the library. The Def Seg value is &H3800 and there is no value for the Clear statement.

The first and most important statement of the program is to define the offsets to the Halo routines. The Halo functions are all registered to specific locations in memory. These functions must be initialized before the program

can be activated. This initialization is done by placing all of the Halo functions into a subroutine in the program. (See Appendix E). The next statement must define Def Seg as &H3800. Then, the program must load the binary files for Halo and the device driver. All the statements must be activated before any type of program can be written. These statements are as follows:

```
10 Gosub 10000
20 Def Seg = &H3800
30 Bload "haloi.bin",0
40 Device$ = "halonine.dev"
50 Call Setdev(Device$)
```

This example program has now activated the functions of Halo, the binary files, and the device driver. Now, the graphics mode has to be initialized. There are two types of graphics modes. INITGRAPHICS command initializes the built-in Halo package, which clears screen to black and restores the Halo color palette. The STARTGRAPHICS initializes the graphics mode, but does not clear the screen black and leaves the present color palette. The STARTGRAPHICS command must be used when interfacing with the Artronics PCPAINT program.

```
60 Mode% = 0
70 Call STARTGRAPHICS(Mode%)
```

There is one more set of statements which must be incorporated to specify the parameters of the screen. These statements will set the viewport and window of the screen. The viewport parameters have been calculated to define the entire screen. The Border and Back variables are set to the current colors of the background and border.

The window or world coordinates are adjustable to the programmer's desire. The program will specify the center of the screen at 0,0. The maximum x and y coordinates are 201 while the minimum x and y values are -201. The reason for -201 is that the screen will not plot the last point number specified by window. The program actually plots an area of -200 to 200. Also, the YW = -151 is specified because the screen is a rectangle shape and there needs to be an adjustment to create a square image area. Otherwise, a cube would turn out looking like a rectangle.

```
 80 xv=.02:yv=.04:x2v=-.95:y2v=.95
 90 Boarder%=-1:Back%=-1
100 Call Setviewport(xv,yv,x2v,y2v)
110 xw=-210:yw=-151:x2w=201:y2w=151
120 Call Setworld(xy,yw,x2w,y2w)
```

The viewport, window and world features can be manipulated to the configuration of the programmer's needs. The following commands are the basic parameters which must be set for creating graphics. There is an automatic clipping of the screen. The programmer can also control the clip by using the Halo Clipping function.

The rest of the programming structure is fundamental programming logic for BASIC. It is good practice to always dimension arrays at the beginning of a program. The better organized program will run more efficiently. This organization is crucial for animation. Remark statements are also good practice and should be incorporated into a program. REM statements are not only helpful to the programmer, but to an individual debugging the program. There is sufficient memory available for storing programs and dimensioning arrays.

There is only one more area of programming structure which must be discussed. The calling of functions should not be a difficult task. The syntax of the functions are well documented in the Halo reference manual. The following code is an example of drawing a line from one point to another point.

```
130 rem set color to first position
135 rem
140       cl%=1
150       call Setcolor(cl%)
155 rem
156 rem calling the start point of line
157 rem
160       x1=100:y1=100
170       Call Movabs(x1,y1)
180 rem
190       x2=150:y2=150
200       call Lnabs(x2,y2)
```

The program draws a line from 100,100 to 150,150. The move and line functions are absolute numbers. There are a special set of routines for relative numbers. Color is called by location on the color palette up to 256. The color variables must have a % sign for the software to interpret. A list of all the functions available in Halo are listed in Appendix F. The first program has been written except for the closing statements. The program must close the graphics mode before exiting. The following statement will conclude this program.

```
  210       Call Closegraphics
  220       End
  300 rem
10000 rem
10010 rem subroutine for halo offsets
10020 rem
10030       (see Appendix F)
```

D. Animation

There are two types of animation experimented with in SAP. The Halo software package has a Setxor command that will convert any program into an exclusive OR mode. This conversion worked fairly well as discussed in Creation of Animation Program. The other type of animation is defining an area on the screen and moving the area to another location. The process has some complications. Presently, the area being defined is stored in an array with Movefrom. The area is then saved and moved to its new position with Moveto. This process is repeated to remove the area and restore the background. Finally, the next position must be calculated and then the entire process is repeated. This repetition continues until the area has reached its final position. The area animation in SAP only works on a black background. There are some minor and major corrections needed to animate an area over a colored background. These changes have been discussed in Creation of Animation Program. This type of animation is possible and would be a wonderful future thesis.

The animation concludes a general documentation of writing a basic program for the Artronics. The only way to really understand the software package and its capabilities is to write programs and play with the BasArt software package.

Conclusion

The sole purpose of this thesis is to encourage future development not only of animation, but bit map graphics. There is no question that animation is possible on the Artronics. Frame-by-frame animation is demonstrated through the Super Animation Program (SAP). Realtime animation is possible if the correct programming language is used and the exclusive OR function is developed. The designer has the ability to create animation through programming on the Artronics. This project's objective was not only to create an animation program, but to investigate the capabilites of bit map computer graphics. The personal computer market is an enormous growth area in the computer business. The declining cost of personal computers is providing the ability for the average household to purchase personal computers. The personal computer has more capabilities then ever before in the industry. According to my knowledge, not too many personal computers have the capability of 256 colors for two-dimensional animation. I feel that there could be a major market for individuals wanting to create this type of animation if the price is affordable. The Apple TAKE 1 and other animation packages have proven to be successful in the marketplace. Vector graphics has made tremendous strides in the past ten years, but the hardware is still expensive and bulky. I think that vector graphics will be possible on a personal computer in the future. For the time being, I believe the personal computer can use bit map graphics to produce high quality and affordable computer animation.

This thesis poses many new questions and provides the basis for future research. Animation for the Artronics poses the most interesting questions. Most questions revolve around the possibilities of realtime animation. The

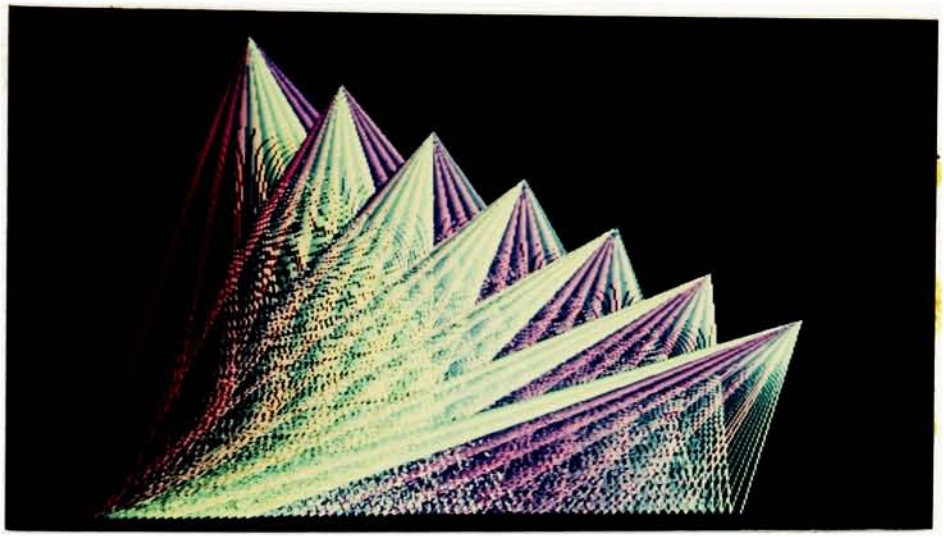
most obvious question is whether or not C programming will produce realtime animation. I firmly believe that a well organized program written in C will produce realtime animation. All the problems with exclusive OR have been determined and are rectifiable. There is a limit to the amount of memory available, but the use of pointers in C would solve most of these problems. There is a problem with the location of color palettes in memory. The location of the PCPAINT color palette and Halo color palettes are in two different locations on the board. These locations become a problem when trying to load and save color palettes. There must be a way of storing color palettes through Halo by finding out the location of storing and loading routines from the Artronics Corporation. I would also suggest working with a computer scientist who specializes in software.

The technical aspect of this project should not overshadow the aesthetic values. The reason for programming the Artronics was to create more visually interesting forms which could be used by a designer. I think there are enormous possibilities with 256 colors to create beautiful controlled patterns (See figures 5,6). A designer could create an image in PCPAINT and combine that image with a personal program using BasArt. This possibility could be another thesis topic studying the combination of technical computer programming art and paint program art.

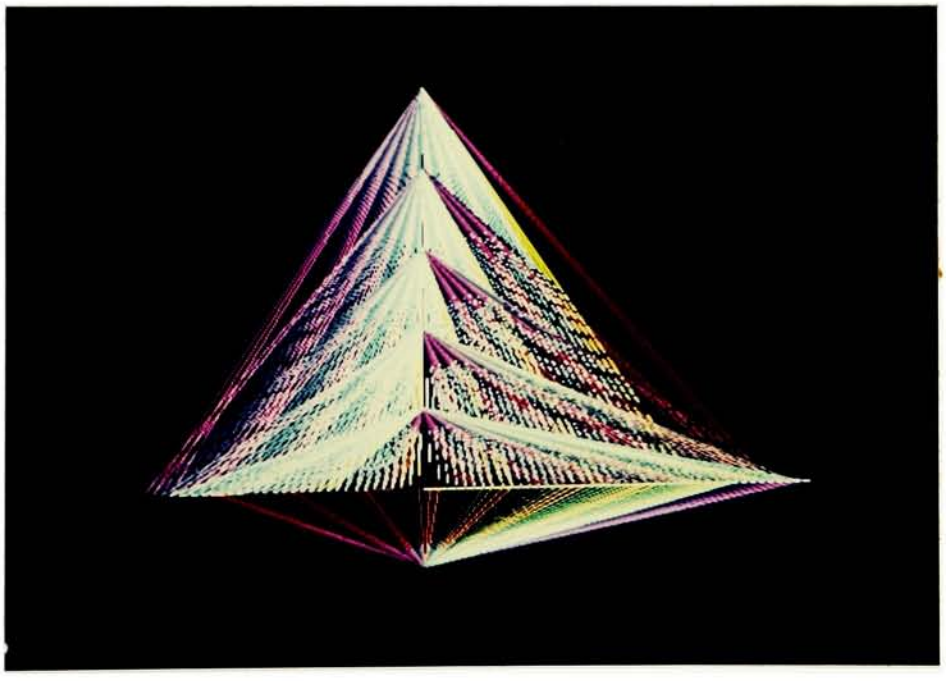
I would like to create a relationship between the crafts and computer graphics design. I believe that the computer graphics designer does not give enough thought to the applications of computer graphics in other areas of art such as the crafts. I incorporated scupltural glass and computer graphics. The general feedback was that either a person enjoyed the glass or was impressed with my computer

graphic images. A mere handful of people asked about the combination of glass and computer graphics. I therefore hope that the future will eventually draw more excitement to the combining of computer graphics and other media of art. This issue revolves around the fact that computer graphic imagery has not been accepted by the art community. Computer art is but still youthful in its stages. However, the major weakness in the industry is hardcopy output. Slide production quality is increasing with higher resolution and better film recorders. Hardcopy output is still expensive. It is difficult to match the quality of the image on a monitor to a print. Fortunately, the computer graphics industry is constantly improving hardware capabilities such as output.

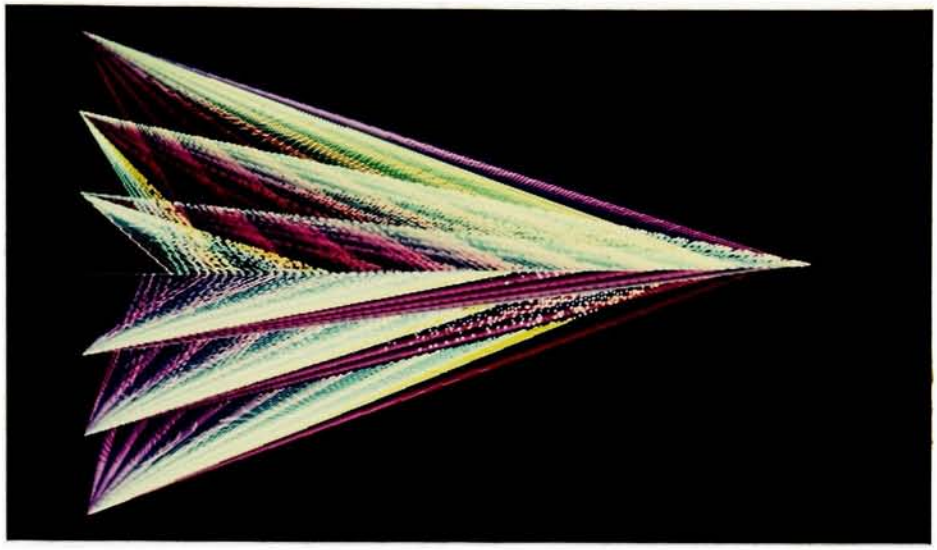
I believe my thesis and other theses in the department are beginning to address these problems. I hope in the future the R.I.T.'s MFA program in Computer Graphics Design will tackle and solve some of these problems. I feel the success of this program depends on the students' willingness to solve the tough issues facing computer graphics in the future. This technical thesis is a start towards tackling the issues of bit map computer animation for the future. I truly hope that someone will follow up and expand upon my reseach of computer animation.



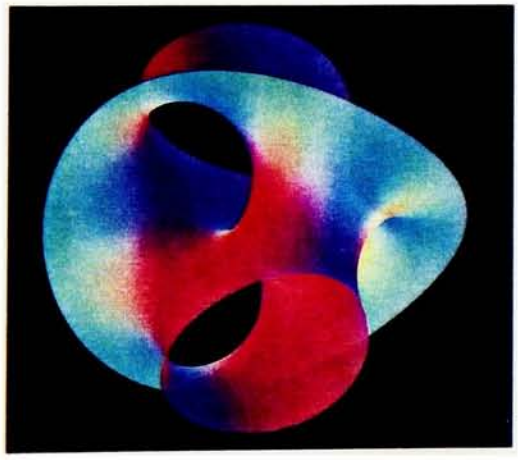
1. Superline1



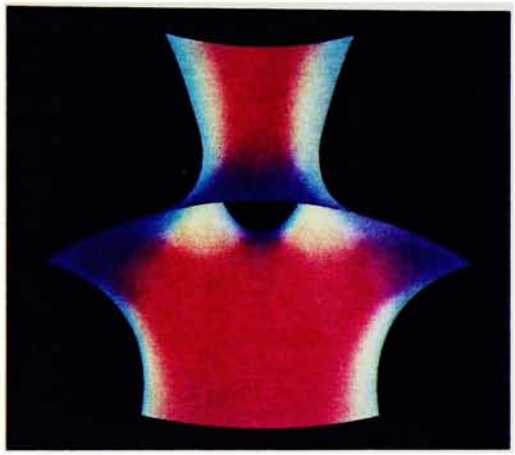
2. Superline2

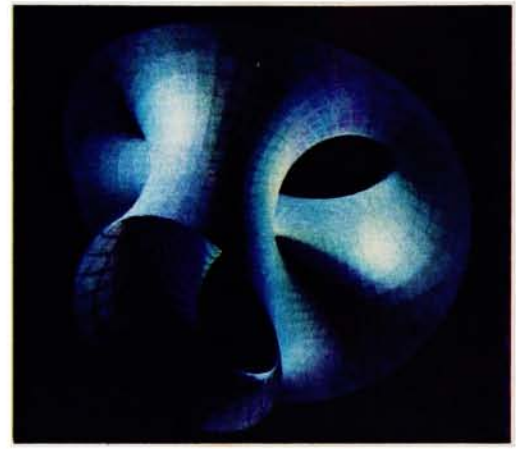


3. Superline3



4. Omniline





5. Omniline1

Bibliography

Andrew, Mark, *Apple Roots*, Berkeley:Osborne McGraw-Hill, 1986.

Bloom, Lloyd, Number Nine Computer Company, Cambridge, Massachusetts. Consultation, January-March 1986.

Consumers Union, "IBM-Compatible Computers", *Consumer Reports*, March 1986, pp. 166-169.

Foley, J.D., and Van Dam, A. *Fundamentals of Interactive Computer Graphics,* Reading:Addison-Wesley Publishing Company, 1982.

Fox, David, and Waite, Mitchell, *Computer Animation Primer,* New York: McGraw-Hill Book Company, 1984.

Hager, Paul, Artronics Computer Company, South Plainfield, New Jersey. Consultation, February-March 1986.

Henderson, Brian, Media Cybernetics, Inc., Silver Springs, Maryland. Consultation, Feb.-March 1986.

Keeler, Graham, *Machine Level Programming on the Apple* II/IIe, Englewood Cliffs: Prentice-Hall International, 1984.

Kelly, Bill, Artronics Computer Company, South Plainfield, New Jersey. Consultation, February-March 1986.

Kernighan, Brian W., and Ritchie, Dennis M., *The C Programming Language,* Englewood Cliffs: Prentice Hall, 1986.

LePage, Danny, Candidate for Masters of Computer Science, Rochester Institute of Technology, Rochester, New York. Consultation, Jan.-May 1986.

Magnenat-Thalmann, Nadia, and Thalmann, Daniel, *Computer Animation,* Tokyo:Springer-Verlag, 1985.

Malkin, Leonard L., *Hi-Res Graphics and Animation Using Assembly Language,* Hasbrouck Heights:Hayden Book Company, Inc., 1986.

Moore, Herb, *Sound and Graphics for the Commodore 64,* New York:John Wiley & Sons, Inc., 1985.

Myers, Roy E., *Microcomputer Graphics for the IBM PC,* Reading:Addison-Wesley Publishing Company, 1984.

Norwark, Frank, Media Cybernetics, Inc., Silver Springs, Maryland. Consultation, Feb.-March 1986.

West, Raeto Collin, *Programming the Commodore 64,* Greensboro: COMPUTE! Publications, Inc., 1985.

APPENDIX

APPENDIX A

Artronics Sync Parameters

Sink_Commands_for_the_Artronics

SINK	PARA	PITCH
1fh	02h	40h
3ah	00h	
64h	00h	
0ch	7fh	
03h		
02h		
f0h		
44h		

APPENDIX B

Linetest Program

```
10 REM
20 REM
30 REM     linetest
40 REM
50 REM     setup parameters
60 REM
70      GOSUB 1000
80      DEF SEG = &H3800
90      BLOAD "B:HALOI.BIN",0
100     DEVICE$ = "B:HALONINE.DEV"
110     CALL SETDEV(DEVICE$)
120     MODE% = 0: CALL INITGRAPHICS(MODE%)
130     X = 0: YMAX = 200: XMAX = 200: Y = 0: CALL SETWORLD(X , YMAX, XMAX , Y)
180 REM
190 REM
200     INPUT "Enter the number of color - ",CL%
210     PRINT "*"
220     INPUT "Enter the first point of line - ",X,Y
230     PRINT "*"
240     INPUT "Enter the end point of line - ",XX,YY
250 REM
255     CALL SETCOLOR(CL%)
260     CALL MOVABS(X,Y)
270     CALL LNABS(XX,YY)
280 REM
290     INPUT "Do you want to draw another line - ",T$
300     IF T$ = "y" OR "Y" THEN GOTO 200
310     CALL CLOSEGRAPHICS
320     END
900 REM
1000    CLOSEGRAPHICS=&H2C3
1010    INITGRAPHICS=&H302
1020    LNABS=&H15
1030    MOVABS=&H23
1040    SETCOLOR=&H0
1050    SETWORLD=&H142
1060    SETDEV=&H261
1090    RETURN
```

APPENDIX C

Line Animation Program

```
10 REM
20 REM
30 REM     linetest
40 REM
50 REM     setup parameters
60 REM
70         GOSUB 1000
80         DEF SEG = &H3800
90         BLOAD "halci.bin",0
100        DEVICE$ = "halonine.dev"
110        CALL SETDEV(DEVICE$)
120        MODE% = 0: CALL INITGRAPHICS(MODE%)
130        X = 0: YMAX = 400: XMAX = 400:Y = 0
135        CALL SETWORLD(X,YMAX,XMAX,Y)
140 REM
145 REM
150        CLS:COUNT=1
155        INPUT "Enter the number of color - ",CL%
160        PRINT "*"
165        INPUT "Enter the first point of line - ",X,Y
170        PRINT "*"
175        INPUT "Enter the end point of line - ",XX,YY
180        PRINT "*"
185        INPUT "Enter the x and y movement of first point - ",RX,RY
200 REM
205        RR=20
210        FOR I= 1 TO 140
220            CALL SETCOLOR(CL%)
230            CALL MOVABS(X,Y)
240            CALL LNABS(XX,YY)
245        CL%=CL%+1
250             IF CL%=254 THEN CL%=125
253                 IF I<70 THEN XX=XX+4
257                 IF I>70 THEN XX=XX-4
260          IF I=140 THEN X=X+(RX*(-1))
263          IF I=140 THEN Y=Y+(RY*(-1))
290        NEXT I
294        COUNT=COUNT+1
295        IF COUNT<8  THEN GOTO 210
300 REM
310       INPUT "Do you want to draw another line -",T$
312       CL%=1
313       CALL SETCOLOR(CL%)
315       CALL CLR
320       IF T$ = "y" THEN GOTO 150
330       CALL CLOSEGRAPHICS
340       END
400 REM
1000 REM
1010      CLOSEGRAPHICS    =  &H2C3
1020      CLR              =  &H2CA
1030      INITGRAPHICS     =  &H302
1040      LNABS            =  &H15
1050      MOVABS           =  &H23
1060      SETCOLOR         =  &H0
1070      SETWORLD         =  &H142
1080      SETDEV           =  &H261
1090      RETURN
```

APPENDIX D

Test Animation Program

```
100 '
110 '
120 '
130 '------------------------------
140 '   This is a test program for the purpose of
150 '        finding the parameters of an animation
160 '            program, and testing a small animation.
170 '                ---------------------------------------
180 '
190 CLEAR
200 GOSUB 880
210 DEF SEG =&H3800
220 BLOAD   "haloi.bin",0
230 DEVICE$ =   "halonine.dev"
240 CALL SETDEV(DEVICE$)
250 MODE%=0: CALL INITGRAPHICS(MODE%)
260 X=25:YMAX=425:XMAX=425:Y=50
270 CALL SETWORLD(X,YMAX,XMAX,Y)
280 '
290 '
300 CLS: KEY OFF: LOCATE 10,15
310 PRINT "   w e l c o m e  to A R T R O N I C S    "
320 LOCATE 20,15
330 PRINT "       Written by Peter Franks       "
340 FOR I=1 TO 2000
350 NEXT I
360 '
370 '
380 CLS: LOCATE 10,10
390 VV=0
400 COUNT=0
410 REM
420 INPUT "Input the number of points in shapes ",NP
430 DIM X1(NP),Y1(NP)
440 DIM X2(NP),Y2(NP)
450 DIM DX(NP),DY(NP)
460 DIM SX(NP),SY(NP)
470 REM
480 INPUT "Input the number of steps in transformation ",NS
490 REM
500 INPUT "Input the color of object by number ",CL%
510 REM
520 PRINT "Input the coordinates of the first shape"
530 FOR I=1 TO NP
540     INPUT "Input point ",X1(I),Y1(I)
550 NEXT I
560 REM
570 CLS
580 PRINT "Input the coordinates of the second shape"
590 FOR I=1 TO NP
600     INPUT "Input point ",X2(I),Y2(I)
610 NEXT I
620 REM
630 REM
640 FOR J=1 TO NS
650    FOR K=1 TO NP
660        DX(K)=(X2(K)-X1(K))/NS
670        DY(K)=(Y2(K)-Y1(K))/NS
680     SX(K)=(X1(K)+J*DX(K))
690     SY(K)=(Y1(K)+J*DY(K))
700       CALL SETCOLOR(CL%)
```

```
        IF K=1 THEN GOTO 750
           CALL MOVABS(SX(K-1),SY(K-1))
           CALL LNABS(SX(K),SY(K))
      CL%=CL%+1
  NEXT K
   WX=X1(1)+J*DX(1)
   WY=Y1(1)+J*DY(1)
  CALL MOVABS(SX(NP),SY(NP))
  CALL LNABS(WX,WY)
NEXT J
    GOTO 860
COUNT=COUNT+1
IF COUNT=5 THEN GOTO 860
GOTO 630
CALL CLOSEGRAPHICS
END
'---------------------------------------
CLOSEGRAPHICS=&H2C3
CLR=&H2CA
INITGRAPHICS=&H302
LNABS=&H15
MOVABS=&H23
SETCOLOR=&H0
SETWORLD=&H142
SETDEV=&H261
RETURN
```

APPENDIX E

Super Animation Program (SAP)

```
100   REM
105   REM                                          peter franks
110   REM                                          april 10,1986
115   REM                                          three-d animation
120   REM                                          thesis
130   REM
135   REM ************************************************************
140   REM
145   REM       This is it!!  Super Animation Program
150   REM
155   REM ************************************************************
160   REM
200   REM
205   REM  entering any code for chaining programs
210   REM
300   REM
305   REM  setting up parameters for program
310   REM
315          CLS:GOSUB 10000
320          DEF SEG = &H3800
325          BLOAD "haloi.bin",0
330          DEVICE$ = "halonine.dev"
335          CALL SETDEV(DEVICE$)
340          MODE%=0: CALL STARTGRAPHICS(MODE%)
345          XV=.02:YV=.04:X2V=.95:Y2V=.95
350          BORDER%=-1:BACK%=-1
355          CALL SETVIEWPORT(XV,YV,X2V,Y2V,BORDER%,BACK%)
360          XW=-201:YW=-151:X2W=201:Y2W=151
365          CALL SETWORLD(XW,YW,X2W,Y2W)
370   REM
375          DIM AF(45),BF(45),CF(45)
380          DIM DF(45),FLAG(45),XSP(45),YSP(45)
382          DIM X1(45),Y1(45),Z1(45),X2(45),Y2(45),Z2(45)
384          DIM OBLIST(45),PL(45,45)
386          DIM X(45),Y(45),Z(45)
388          DIM XS(45),YS(45),SP(45),EP(45)
390          DIM DX(45),DY(45),DZ(45)
392          DIM XX1(45),YY1(45),ZZ1(45)
394          DIM XX2(45),YY2(45),ZZ2(45)
396          DIM CX(45),CY(45),CZ(45),BX(45),BY(45),BZ(45)
397          DIM ARRAY%(1253),OFFSET%(1253)
398          DISK=0
```

```
400   REM
405   REM  opening message to programmer before starting
410   REM
415       CLS:LOCATE  7,20
420       PRINT "Three Dimensional Wireframe Animation"
430       LOCATE 10,20
435       PRINT "       created by peter franks          "
440       LOCATE 13,20
445       PRINT "          Thesis - April 1986           "
450       LOCATE 16,20
455       PRINT "                  S A P                 "
460       FOR I=1 TO 2000: NEXT I
500   REM
510   REM first main menu
520   REM
525       CLS:PRINT "*"
530       PRINT "-----------------------------------------"
535       PRINT "                  Menu                   "
540       PRINT "                  ------                 "
545       PRINT "               Main  Menu                "
550       PRINT "-----------------------------------------"
555       PRINT "  1) Shape Services Menu                 "
560       PRINT "  2) Viewer Services Menu                "
565       PRINT "  3) Camera Services Menu                "
570       PRINT "  4) Animation Test Menu                 "
575       PRINT "  5) Shapes in Storage Table            "
577       PRINT "  6) Quit                                "
580       PRINT "-----------------------------------------"
585       LOCATE 15,1
590       PRINT "-----------------------------------------"
595       LOCATE 14,1
600       INPUT "Enter option by number: ",E
605       IF E=1 THEN GOTO 665
610          IF E=2 THEN GOTO 1065
615             IF E=3 THEN GOTO 865
620                IF E=4 THEN GOTO 3515
625                   IF E=5 THEN GOTO 4015
630                IF E=6 THEN GOSUB 6930
635                   GOTO 525
650   REM
655   REM this the shape services menu
660   REM
665      CLS:GOSUB 7565
666      LOCATE 1,1:PRINT "*"
670       PRINT "-----------------------------------------"
675       PRINT "                  Menu                   "
680       PRINT "                  ------                 "
685       PRINT "           Shape Services Menu           "
690       PRINT "-----------------------------------------"
695       PRINT " 1) Create Shape      7) Clear Screen    "
700       PRINT " 2) Load Shape        8) Viewer Services Menu "
705       PRINT " 3) Store Shape       9) Camera Services Menu "
710       PRINT " 4) Transformation   10) Print Image(Ink Jet) "
715       PRINT " 5) Color change     11) Text
720       PRINT " 6) Draw             12) Main Menu       "
725       PRINT "-----------------------------------------"
730       LOCATE 15,1
735       PRINT "-----------------------------------------"
740       LOCATE 14,1
745       INPUT "Enter the option by number: ",F
750       IF F=1 THEN GOSUB 7765
755          IF F=2 THEN GOTO 1265
760             IF F=3 THEN GOTO 1465
765                IF F=4 THEN GOTO 3015
770                   IF F=5 THEN GOSUB 8015
```

```
775                    IF F=6 THEN GOSUB 7220
780                 IF F=7 THEN GOSUB 7515
785                    IF F=8 THEN GOTO 1065
790                 IF F=9 THEN GOSUB 865
795                    IF F=10 THEN GOSUB 8515
800                       IF F=11 THEN GOSUB 3315
805                    IF F=12 THEN GOTO 525 ELSE GOTO 665
850   REM
855   REM this is the camera services menu
860   REM
865       CLS:GOSUB 7565
866       LOCATE 1,1:PRINT "*"
870        PRINT "-----------------------------------------"
875        PRINT "                  Menu                   "
880        PRINT "                  ------                 "
885        PRINT "          Camera Services Menu           "
890        PRINT "-----------------------------------------"
895        PRINT " 1) Zoom            6) Draw              "
900        PRINT " 2) Tilt            7) Shape Services Menu "
905        PRINT " 3) Pan             8) Viewer Services Menu "
910        PRINT " 4) Spin            9) Main Menu         "
915        PRINT " 5) Color change   10) Quit              "
920        PRINT "-----------------------------------------"
925        LOCATE 14,1
930        PRINT "-----------------------------------------"
935        LOCATE 13,1
940        INPUT "Enter the option by number: ",G
945        IF G=1 THEN GOTO 2120
950           IF G=2 THEN GOTO 2415
955              IF G=3 THEN GOTO 2515
960                 IF G=4 THEN GOTO 2315
965                    IF G=5 THEN GOSUB 8015
970                       IF G=6 THEN GOSUB 7220
975                    IF G=7 THEN GOTO 665
980                       IF G=8 THEN GOTO 1065
985                          IF G=9 THEN GOTO 525
990                       IF G=10 THEN GOSUB 6930 ELSE 865
1050  REM
1055  REM this is the viewer services menu
1060  REM
1065      CLS:GOSUB 7565
1066      LOCATE 1,1:PRINT "*"
1070      PRINT "-----------------------------------------"
1075      PRINT "                  Menu                   "
1080      PRINT "                  ------                 "
1085      PRINT "          Viewer Services Menu           "
1090      PRINT "-----------------------------------------"
1095      PRINT " 1) Distance change   8) Print Image     "
1100      PRINT " 2) Move shape        9) Line width      "
1105      PRINT " 3) Scale shape      10) Line style      "
1110      PRINT " 4) Rotate shape     11) Camera Services Menu "
1115      PRINT " 5) Change any pt.   12) Shape Services Menu  "
1120      PRINT " 6) Draw shape       13) Main Menu       "
1125      PRINT " 7) Clear screen     14) Change color    "
1130      PRINT "-----------------------------------------"
1135      LOCATE 16,1
1140      PRINT "-----------------------------------------"
1145      LOCATE 15,1
1150      INPUT "Enter the option by number: ",H
1155      IF H=1 THEN GOTO 1615
1160         IF H=2 THEN GOTO 1715
1165            IF H=3 THEN GOTO 1815
1170               IF H=4 THEN GOTO 2215
1175                  IF H=5 THEN GOTO 1965
1180                     IF H=6 THEN GOSUB 7220
1185                  IF H=7 THEN GOSUB 7515
1190                     IF H=8 THEN GOSUB 8415
```

```
1190                    IF H=8 THEN GOSUB 8410
1195                 IF H=9 THEN GOSUB 8120
1200                    IF H=10 THEN GOSUB 8215
1205                 IF H=11 THEN GOTO 865
1210                    IF H=12 THEN GOTO 665
1215                 IF H=13 THEN GOTO 525
1220                    IF H=14 THEN GOSUB 8015
1230                    GOTO 1065
1250  REM
1255  REM restoring shape
1260  REM
1265       CLS:PRINT "*"
1268       GOSUB 9015
1269       LOCATE 2,1
1270       INPUT "Enter the viewing distance of shape: ",D
1271       PRINT "-----"
1275       INPUT "Enter the number of files to be restored: ",NS
1278       PRINT "-----"
1279       PRINT "Enter filename in following form:  b:filename.dat "
1280       SPT=0:EPT=0:EPL=0:SPL=0
1285       FOR S=1 TO NS
1290            INPUT "Enter shape filename: ",FILE$
1295            OPEN "I",#1,FILE$
1300            INPUT#1,NP:N=5*S
1305            FOR I=1 TO NP
1310                K=I+EPT:INPUT#1,A:INPUT#1,B:INPUT#1,C
1315                X(K)=A:Y(K)=B:Z(K)=C:NEXT I
1320         INPUT#1,NPL
1325         FOR I=1 TO NPL
1330              K=I+EPL:INPUT#1,PL(K,0)
1335              FOR J=1 TO PL(K,0)
1337                   IF EOF(1) THEN GOTO 1347
1340                   INPUT#1,E:PL(K,J)=E+EPT:NEXT J
1345              NEXT I
1347              CLOSE#1
1350         SPT=EPT+1:EPT=EPT+NP
1355         SPL=EPL+1:EPL=EPL+NPL
1360         OBLIST(N+1)=SPT:OBLIST(N+2)=EPT
1365         OBLIST(N+3)=SPL:OBLIST(N+4)=EPL
1370         IF S=1 THEN PP$=FILE$
1375         IF S=2 THEN Q$=FILE$
1380         IF S=3 THEN R$=FILE$
1385         IF S=4 THEN S$=FILE$
1387         IF S=5 THEN T$=FILE$
1390         NEXT S
1395         OBLIST(1)=1:OBLIST(2)=EPT
1400         OBLIST(3)=1:OBLIST(4)=EPL
1402         IF DICK=1 THEN GOTO 3050
1403         IF DICK=2 THEN GOTO 3085
1404        GOSUB 8017
1405         GOTO 665
1450  REM
1455  REM  saving shape
1460  REM
1465       GOSUB 7565:LOCATE 18,1
1467       PRINT "Enter filename in the following form: B:FILENAME.DAT"
1470       INPUT "Enter the filename of shape: ",FILE$
1475       OPEN "O",#1,FILE$
1478     SPT=1:EPT=NP
1480       PRINT#1,NP
1485         FOR I=SPT TO EPT
1490              A=X(I):B=Y(I):C=Z(I)
1495              PRINT#1,A
1500              PRINT#1,B
1505              PRINT#1,C: NEXT I
1510         PRINT#1,NPL
1515         FOR I=1 TO NPL
```

```
1515        FOR I=1 TO NPL
1520            PRINT#1,PL(I,0)
1525            FOR J=1 TO PL(I,0)
1530                E=PL(I,J):PRINT#1,E:NEXT J
1540        NEXT I
1545      CLOSE#1: GOTO 665
1600  REM
1605  REM  change distance
1610  REM
1615      CLS
1620      PRINT "*"
1625      PRINT "Changing Viewing Distance"
1627      PRINT "------"
1628      GOSUB 7120
1630      PRINT "------"
1635      INPUT "Enter the new viewing distance: ",VD
1640      INPUT "Enter the number of steps: ",F
1645      PRINT "------"
1650      GOSUB 9115
1655      VD=VD-D:VD=VD/F
1660         FOR FR=1 TO F
1665             D=D+VD:GOSUB 8320:GOSUB 7220
1670         NEXT FR
1675      STC=0:GOTO 1065
1700  REM
1705  REM move shape
1710  REM
1715      CLS
1720      PRINT "*"
1725      PRINT "Moving object anywhere on screen"
1726      PRINT "------"
1727      GOSUB 7120
1730      PRINT "------"
1735      INPUT "Enter the amount of movement on the x-axis: ",MX
1740      INPUT "Enter the amount of movement on the y-axis: ",MY
1745      INPUT "Enter the amount of movement on the z-axis: ",MZ
1750      INPUT "Enter the number of steps: ",F
1755      PRINT "------"
1757      GOSUB 9115
1760      FOR FR=1 TO F
1765          FOR I=W1 TO W2
1770              X(I)=X(I)+MX/F:Y(I)=Y(I)+MY/F:Z(I)=Z(I)+MZ/F
1775          NEXT I:GOSUB 8320:GOSUB 7220
1780      NEXT FR
1785      IF TIT=1 THEN GOTO 2460
1787      IF PIN=1 THEN GOTO 2560
1788      STC=0
1790      GOTO 1065
1800  REM
1805  REM change scale
1810  REM
1815      CLS
1820      PRINT "*"
1825      PRINT "Scaling object up or down"
1830      PRINT "------"
1835      GOSUB 7120
1840      PRINT "------"
1845      PRINT "The scaling factor is entered as 1 = current size,"
1850      PRINT "2 = double current size, and .5 = half current size."
1855      INPUT "Enter the scale factor for the x axis: ",SX
1860      INPUT "Enter the scale factor for the y axis: ",SY
1865      INPUT "Enter the scale factor for the z axis: ",SZ
1870      INPUT "Enter number of steps: ",F
1872      PRINT "------"
1874      GOSUB 9115
1875      FOR I=W1 TO W2
```

```
1880          EX=SX*(X(I)-XC)/FXC:EY=SY*(Y(I)-YC)/FYC
1885          EZ=SZ*(Z(I)-ZC)+ZC:DX(I)=(EX-X(I))/F
1890          DY(I)=(EY-Y(I))/F:DZ(I)=(EZ-Z(I))/F
1895       NEXT I
1900       FOR FR=1 TO F
1905          FOR I=W1 TO W2
1910             X(I)=X(I)+DX(I):Y(I)=Y(I)+DY(I):Z(I)=Z(I)+DZ(I)
1915          NEXT I:GOSUB 8320:GOSUB 7220
1920       NEXT FR
1930       STC=0:GOTO 1065
1950    REM
1955    REM change x,y,z
1960    REM
1965       CLS
1970       PRINT "*"
1975       PRINT "Change any point on screen"
1980       PRINT "------"
1985       GOSUB 7120
1990       PRINT "------"
1995         GOSUB 7520
2000         HHT%=2:WTH%=1:PH%=0:MD%=1
2005         CALL SETTEXT(HHT%,WTH%,PH%,MD%)
2010         FORE=2:BACK=0
2013         CALL SETTEXTCLR(FORE,BACK)
2016         HGT=4:WIDT=2:BAC=0
2019         CALL INITTCUR(HGT,WIDT,BAC)
2022         FOR I=1 TO NP
2023            XS(I)=(D*X(I))/(D+Z(I))
2024            YS(I)=(D*Y(I))/(D+Z(I))
2025            CALL PTABS(XS(I),YS(I))
2026            CALL MOVTCURABS(XS(I),YS(I))
2027            TXT= I : CALL TEXT(TXT)
2029         NEXT I
2035       INPUT "Enter the point number to change: ",PT
2040       INPUT "Enter the amount of change in x,y, and z: ",XX,YY,ZZ
2045       INPUT "Enter the number of steps: ",F
2047       PRINT "------"
2048       GOSUB 9115
2050       FOR FR=1 TO F
2055          X(PT)=X(PT)+XX/F:Y(PT)=Y(PT)+YY/F:Z(PT)=Z(PT)+ZZ/F
2060          GOSUB 8320:GOSUB 7220
2065       NEXT FR
2070       STC=0:GOTO 1065
2100    REM
2105    REM zoom
2110    REM
2120      CLS
2125      PRINT "*"
2130      PRINT "Zoom screen in and out"
2135      PRINT "------"
2140      PRINT "You will be asked to enter a zoom factor."
2142      PRINT "The image will become smaller if the zoom  "
2143      PRINT "factor is less than 1, and the image will  "
2144      PRINT "become larger if the zoom factor is greater than 1."
2145      PRINT "------"
2147      INPUT "Enter the zoom factor: ",ZM
2150      INPUT "Enter the number of steps: ",F
2152       PRINT "------"
2154       GOSUB 9115
2155         WXM=XW/ZM:WXX=X2W/ZM
2160         WYM=YW/ZM:WYX=Y2W/ZM
2162      FACTOR%=0
2165          GA=(WXM-XW)/F:GB=(WXX-X2W)/F
2170          GC=(WYM-YW)/F:GD=(WYX-Y2W)/F
2175        FOR FR=1 TO F
2180            XW=XW+GA:X2W=X2W+GB
2185            YW=YW+GC:Y2W=Y2W+GD
```

```
2183            XW=XW+GC:X2W=X2W+GD
2187            CALL SETWORLD(XW,YW,X2W,Y2W)
2189            GOSUB 8320:GOSUB 7220
2191          NEXT FR
2193          STC=0:GOTO 865
2200  REM
2205  REM rotate shape
2210  REM
2215        CLS:PRINT "*"
2220        PRINT "Rotation of object"
2225        PRINT "------"
2230        PRINT "This will rotate an object around its center"
2232        PRINT "------"
2235        GOSUB 7120
2240        PRINT "------"
2245        INPUT "The object will rotate around which axis (x,y or z): ",AX$
2250        INPUT "Enter the degrees of rotation: ",ANGLE
2255        INPUT "Enter the number of frames in which rotation will occur: ",F
2257        PRINT "------"
2258        GOSUB 9115
2260            IF AX$ = "x" THEN GOTO 8625
2265            IF AX$ = "y" THEN GOTO 8690
2270            IF AX$ = "z" THEN GOTO 8755
2275          GOTO 1065
2300  REM
2305  REM spin shape
2310  REM
2315        CLS
2320        PRINT "*"
2325        PRINT "Spin object around center of screen"
2330        PRINT "------"
2335        GOSUB 7120
2340        PRINT "------"
2345        INPUT "Enter the degrees of spin: ",SP
2347        INPUT "Enter number of frames: ",F
2348        PRINT "------"
2349        GOSUB 9115
2350        AZ=-1*SP
2355        XSC=(XMIN+XMAX)/2:YSC=(YMIN+YMAX)/2
2360        RS=(AZ*3.14)/180
2365        FOR FR=1 TO F
2367            FOR I=W1 TO W2
2370               X1(I)=(X(I)-XSC)*COS(RS/F)-(Y(I)-YSC)*SIN(RS/F)+XSC
2375               Y1(I)=(X(I)-XSC)*SIN(RS/F)+(Y(I)-YSC)*COS(RS/F)+YSC
2380               X(I)=X1(I):Y(I)=Y1(I)
2383            NEXT I
2386            GOSUB 8320:GOSUB 7220
2389        NEXT FR
2392        STC=0:GOTO 865
2400  REM
2405  REM tilt shape
2410  REM
2415        CLS
2417        TIT=1
2420        PRINT "*"
2425        PRINT "Tilt camera around object"
2430        PRINT "------"
2435        GOSUB 7120
2440        PRINT "------"
2445        INPUT "Enter the amount of tilt: ",TT
2450        MZ=0:MY=-1*TT:MX=0
2455        GOSUB 1750
2460        TIT=0
2465        STC=0:GOTO 865
2500  REM
2505  REM pan shape
2510  REM
```

```
2510  REM
2515      CLS
2517      PIN=1
2520      PRINT "*"
2525      PRINT "Pan camera around object"
2530      PRINT "------"
2535      GOSUB 7120
2540      PRINT "------"
2545      INPUT "Enter the amount of pan: ",PN
2550      MZ=0:MY=0:MX=-1*PN
2555      GOSUB 1750
2560      PIN=0
2565      STC=0:GOTO 865
3000  REM
3005  REM transformation
3010  REM
3015        GOSUB 8015
3017        CLS:PRINT "*"
3020        PRINT "Transformation of two objects"
3025        PRINT "------"
3027        PRINT "The shapes must have to same number of points"
3030        INPUT "Enter the viewing distance of objects: ",D
3033        INPUT "Enter the number of shapes being transformed: ",NS
3034        PRINT "------"
3035        PRINT "The first shape can be entered one of two ways"
3036        PRINT "  1) From disk   or  2) last shape drawn on screen"
3037        DICK=1
3038        INPUT "Enter 1 or 2: ",JJ
3039        IF JJ = 2 THEN GOTO 3050
3040        PRINT "        Enter the first shape "
3045        GOSUB 1279
3050             FOR I=1 TO EPT
3055                 XX1(I)=X(I):YY1(I)=Y(I):ZZ1(I)=Z(I)
3060             NEXT I
3065  REM
3068        DICK=2
3070        PRINT "------"
3075        PRINT "        Enter the second shape "
3077        SPT=0:EPT=0:SPL=0:EPL=0
3080        GOSUB 1279
3085             FOR I=1 TO EPT
3090                 XX2(I)=X(I):YY2(I)=Y(I):ZZ2(I)=Z(I)
3095             NEXT I
3100  REM
3105        INPUT "Enter the number of frames: ",F
3107        PRINT "------"
3108        GOSUB 9115
3110        FOR I=1 TO EPT
3115            CX(I)=(XX2(I)-XX1(I))/F
3120            CY(I)=(YY2(I)-YY1(I))/F
3125            CZ(I)=(ZZ2(I)-ZZ1(I))/F
3130        NEXT I
3135  REM
3140        FOR FR=1 TO F
3145            FOR S=1 TO NS
3150                N=5*S
3155                SPT=OBLIST(N+1):EPT=OBLIST(N+2)
3160                SPL=OBLIST(N+3):EPL=OBLIST(N+4)
3165                FOR I=SPT TO EPT
3170                    BX(I)=XX1(I)+FR*CX(I)
3175                    BY(I)=YY1(I)+FR*CY(I)
3180                    BZ(I)=ZZ1(I)+FR*CZ(I)
3183                X(I)=BX(I):Y(I)=BY(I):Z(I)=BZ(I)
3185                NEXT I
3190                GOSUB 7220
3195            NEXT S
```

```
3175            IF FR=1 THEN GOTO 3200
3200            GOSUB 8320
3205          NEXT FR
3207         STC=0
3210         DICK=0:GOTO 665
3300  REM
3305  REM text on screen
3310  REM
3315         CLS:PRINT "*"
3320         PRINT "Placement of text on screen"
3325         PRINT "------"
3330         PRINT "Text involves many variables which must be entered"
3332         INPUT "Enter height and width of a letter in pixels: ",AH,AW
3333         PRINT "------"
3335         INPUT "Do you want a border(1) around type or not(0):1 or 0: ",AB
3336         PRINT "------"
3337         PRINT "How do you want text to read?   1) normal left to right"
3339         PRINT "2) 90 degrees  3) 180 degrees  4) 270 degrees"
3342         INPUT "Enter number: ",AD
3345         PRINT "------"
3350         INPUT "Enter the foreground and background color of text: ",AF,AG
3353         PRINT "------"
3356         INPUT "Enter the height and width of text cursor attributes: ",AC,AV
3357         INPUT "Enter the color of text cursor: ",AN
3358         PRINT "------"
3360         INPUT "Enter the x and y position to start text: ",AXS,AYS
3363         PRINT "------"
3366         INPUT "Enter the text: ",ATT$
3380         HGHT%=AH:WDTH%=AW:PTH%=AD:MD%=AB
3385            CALL SETTEXT(HGHT%,WDTH%,PTH%,MD%)
3390         FORE%=AF:BACK%=AG
3395            CALL SETTEXTCLR(FORE%,BACK%)
3400         HHT%=AC:WTH%=AV:CLR%=AN
3405            CALL INITTCUR(HHT%,WTH%,CLR%)
3410         X%=AXS:Y%=AYS
3415            CALL MOVTCURABS(X%,Y%)
3420         TEXT$ = ATT$
3425            CALL TEXT(TEXT$)
3480         INPUT "Do you want more text on screen? y or n: ",XT$
3485            IF XT$ = "y" THEN GOTO 3315 ELSE  GOTO 665
3500  REM
3505  REM  animation test menu
3510  REM
3515         CLS
3520         LOCATE 1,1:PRINT "*"
3525         PRINT "----------------------------------------"
3530         PRINT "                  Menu                  "
3535         PRINT "                  ------                "
3540         PRINT "          Animation Test Menu           "
3545         PRINT "----------------------------------------"
3550         PRINT "    1) Save Palette                     "
3555         PRINT "    2) Restore Palette                  "
3560         PRINT "    3) Save Image                       "
3565         PRINT "    4) Restore Image                    "
3570         PRINT "    5) Convert Main Program to XOR      "
3575         PRINT "    6) Animate Area on Screen in XOR    "
3580         PRINT "    7) Main Menu                        "
3585         PRINT "----------------------------------------"
3590         LOCATE 16,1
3595         PRINT "----------------------------------------"
3600         LOCATE 15,1
3605         INPUT "Enter option by number: ",K
3610           IF K = 1 THEN GOTO 4315
3620              IF K = 2 THEN GOTO 4215
3625                 IF K = 3 THEN GOTO 4412
3630                    IF K = 4 THEN GOTO 4512
3635                    IF K = 5 THEN GOTO 3715
```

```
3635              IF K = 5 THEN GOTO 3715
3640                 IF K = 6 THEN GOSUB 3810
3645              IF K = 7 THEN GOTO 525 ELSE 3515
3700 REM
3705 REM  setting xor
3710 REM
3715      GOSUB 8915
3717      CLS:PRINT "*"
3720      PRINT "Exclusive OR"
3725      PRINT "-------"
3728      PRINT "*"
3730      INPUT "Do you want turn on(n) or turn off(f) XOR: ",YY$
3735       IF YY$ = "n" THEN GOTO 3755
3740         SWITCH% = 0
3745       PRINT "*"
3747       PRINT "XOR has been turned off..."
3748       FOR I=1 TO 1500: NEXT I
3750       GOTO 3780
3755          SWITCH% = 1
3760       PRINT "*"
3765       PRINT "XOR is now being activated..."
3770       FOR I=1 TO 1500: NEXT I
3780       CALL SETXOR(SWITCH%)
3785         GOTO 3515
3800 REM
3805 REM  area movement in xor
3810 REM
3811      GOSUB 8915
3812      GOSUB 3950
3815      CLS:PRINT "*"
3820      PRINT "Definiting Area for Amination"
3825      PRINT "-------"
3830      PRINT "First, the area to be moved must be definited"
3835      INPUT "Enter the upper left x and y coordinates: ",UX,UY
3840      INPUT "Enter the lower right x and y coordinates: ",LX,LY
3845      PRINT "---"
3850      INPUT "Enter the amount of movement on x axis: ",MX
3855      INPUT "Enter the amount of movement on y axis: ",MY
3860      INPUT "Enter the number of frames: ",FRM
3865      PRINT "---"
3870      INPUT "Enter the mode 1,2,3(xor),4,5,6,7,8: ",MODE%
3872           JAY=0
3875           VX=MX/FRM:VY=MY/FRM
3880        CALL MOVEFROM(UX,UY,LX,LY,ARRAY%(0))
3881           CL%=0:CALL SETCOLOR(CL%)
3882           PX=UX:PY=UY
3883           QX=LX:QY=LY
3884           CALL BAR(PX,PY,QX,QY)
3885           FOR FR=1 TO FRM
3890                CALL MOVETO(UX,UY,ARRAY%(0),MODE%)
3891                   IF JAY=1 THEN GOTO 3894
3892                   FOR I=1 TO 500:NEXT I
3893                   JAY=JAY+1:GOTO 3890
3894                   JAY=0
3895                UX=UX+VX
3900                UY=UY+VY
3902  FOR H=1 TO 500:NEXT H
3905           NEXT FR
3910      PRINT "-------"
3915      INPUT "Do you want to run XOR(x) and quit(q): ",F$
3917          IF F$ = "x" THEN GOTO 3810
3919             IF F$ = "r" THEN GOTO 3845
3925        GOTO 3515
3950 REM
3955 REM small sub for area
3960 REM
```

```
3965      CLS:PRINT "*"
3967       PRINT "A test for locating an area on screen"
3970      INPUT "Do you want to plot any point for test(y or n): ",D$
3975        IF D$ = "n" THEN GOTO 3995
3980    CLS:INPUT "Enter the x and y coordinate: ",X,Y
3985              CALL PTABS(X,Y)
3990        INPUT "Do you want to plot any more(y or n): ",S$
3992        IF S$ = "y" THEN GOTO 3980
3995            RETURN
4000  REM
4005  REM  table for storing shapes
4010  REM
4015       CLS:PRINT "*"
4020       PRINT "Table of available shapes on disk"
4025       PRINT "*"
4030       PRINT "- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -"
4035       PRINT "      filename                  description      "
4040       PRINT " - - - - - - - - - - - - - - - - - - - - - - - - - - - - -"
4045       PRINT " 1)  cube1.dat                -cube in top right hand corner "
4050       PRINT "                               of screen.                    "
4055       PRINT " 2)  cube2.dat                -cube in lower left hand corner"
4060       PRINT "                               of screen.                    "
4065       PRINT " 3)  cool.dat                 -an neat looking six sided "
4070       PRINT "                               figure.    "
4075       PRINT " 4)  cross.dat                -a Swiss Red Cross Symbol "
4080       PRINT "                               in the middle of screen. "
4175          PRINT "------"
4180          INPUT "Hit (return) to return to Main Menu: ",B$
4185              GOTO 525
4200  REM
4205  REM  restoring color palette
4210  REM
4215       GOSUB 8915
4217       CLS:PRINT "*"
4220       PRINT "Restoring Color Palette "
4225       PRINT "------"
4230       PRINT "      This operation will only work if the user has"
4235       PRINT " previously stored a color pallete with this program."
4237       PRINT "----"
4240       INPUT " To proceed hit <return> else type (o): ",HJ$
4245            IF HJ$ = "o" THEN GOTO 4295
4250       PRINT "------"
4255       INPUT "Enter the name of color palette as b:filename.plt: ",PFILE$
4260       PDES$ = "linear palette"
4263       PRINT "*"
4264       PRINT "Restoring color palette..."
4270       CALL PREAD (PFILE$,PDES$)
4295       GOTO 3515
4300  REM
4305  REM saving color palette
4310  REM
4315       GOSUB 8915
4320       CLS:PRINT "*"
4325       PRINT "Saving a color palette "
4330       PRINT "------"
4335       INPUT "Enter the name of color palette as b:filename.plt: ",PFILE$
4340       PDES$ = "linear palette"
4345       MODE% = 0
4347       PRINT "*"
4348       PRINT "Saving color palette..."
4350       CALL PWRITE (PFILE$,PDES$,MODE%)
4390       GOTO 3515
4400  REM
4405  REM saving image
4410  REM
4412      GOSUB 8915
```

```
4415        CLS:PRINT "*"
4420        PRINT "Saving an image "
4425        PRINT "------"
4430        INPUT "Enter the name of image as follows- b:filename.img: ",GFILE$
4435        PRINT "*"
4440        PRINT "Image is being saved...."
4445        CALL GWRITE(GFILE$)
4490        GOTO 3515
4500  REM
4505  REM restoring image
4510  REM
4512        GOSUB 8915
4515        CLS:PRINT "*"
4520        PRINT "Restore an image"
4525        PRINT "------"
4526        PRINT "    This operation will only work if the user has "
4527        PRINT " previously stored an image with this program.   "
4528        PRINT "----"
4529        INPUT "To proceed hit <return> else type (o): ",HJ$
4530            IF HJ$ = "o" THEN GOTO 4590
4531        PRINT "----"
4533        INPUT "Enter the name of image as follows- b:filename.img: ",GFILE$
4535        PRINT "*"
4540        PRINT "Image is being restored...."
4545        CALL GREAD(GFILE$)
4590        GOTO 3515
6900  REM
6910  REM  ending program
6920  REM
6930        CLS:LOCATE 15,15
6935        PRINT "This is the end of the program "
6940        LOCATE 17,15
6942        PRINT " Thank you for playing with me "
6944        LOCATE 19,15
6945        PRINT "     Go have a beer on me       "
6947        CALL CLOSEGRAPHICS
6949        END
6950  REM
6960  REM  ------------------------------------------------
6970  REM                Subroutines
6980  REM  ------------------------------------------------
6990  REM
7000  REM
7010  REM
7020        LOCATE 18,10
7030        PRINT "This OPERATION is not in service "
7040        PRINT "------Sorry Charlie !!!  - Going crazy"
7050        FOR I=1 TO 1500:NEXT I
7060        RETURN
7100  REM
7105  REM this will pick shape
7110  REM
7115        CLS:LOCATE 20,1
7120        PRINT "Enter number of shape for operation"
7125        INPUT "(see shapes):Enter 0 for all shapes: ",S
7130        MDR=1
7135        IF S=0 THEN MDR=NS
7140           FOR M=1 TO MDR:N=5*S
7145               IF S=0 THEN N=5*M
7150           NEXT M
7155        SPT=OBLIST(N+1):EPT=OBLIST(N+2)
7160        SPL=OBLIST(N+3):EPL=OBLIST(N+4)
7165        IF S=0 THEN SPT=1
7166        IF S=0 THEN SPL=1
7170        W1=SPT:W2=EPT:RETURN
7200  REM
```

```
7205  REM draw shape
7210  REM
7215        GOSUB 7115
7220        CALL SETCOLOR(CL%)
7221        PRINT "Drawing...."
7222        FOR S=1 TO NS
7225            N=5*S:SPT=OBLIST(N+1):EPT=OBLIST(N+2)
7230            SPL=OBLIST(N+3):EPL=OBLIST(N+4)
7235        XMAX=-1000:XMIN=1000 :YMAX=-1000
7240        YMIN=1000:ZMAX=-1000:ZMIN=1000
7245        FOR I=SPT TO EPT
7250            IF X(I)>XMAX THEN XMAX=X(I)
7255            IF X(I)<XMIN THEN XMIN=X(I)
7260            IF Y(I)<YMIN THEN YMIN=Y(I)
7265            IF Y(I)>YMAX THEN YMAX=Y(I)
7270            IF Z(I)<ZMIN THEN ZMIN=Z(I)
7275            IF Z(I)>ZMAX THEN ZMAX=Z(I)
7280         NEXT I
7285          XC=(XMAX+XMIN)/2:YC=(YMAX+YMIN)/2:ZC=(ZMAX+ZMIN)/2
7290         FOR I=SPL TO EPL
7295            XF=X(PL(I,1)):YF=Y(PL(I,1)):ZF=Z(PL(I,1))
7300            A1=X(PL(I,3))-X(PL(I,2))
7305            A2=Y(PL(I,3))-Y(PL(I,2))
7310            A3=Z(PL(I,3))-Z(PL(I,2))
7315            B1=X(PL(I,1))-X(PL(I,2))
7320            B2=Y(PL(I,1))-Y(PL(I,2))
7325            B3=Z(PL(I,1))-Z(PL(I,2))
7330               AF(I)=(A2*B3)-(A3*B2)
7335               BF(I)=(A3*B1)-(A1*B3)
7340               CF(I)=(A1*B2)-(A2*B1)
7345             DF(I)=(-AF(I)*XF)-(BF(I)*YF)-(CF(I)*ZF):NEXT I
7350         FOR N=SPL TO EPL
7355            TEST=(AF(N)*XC)+(BF(N)*YC)+(CF(N)*ZC)+DF(N)
7360            IF TEST>=0 THEN GOTO 7375
7365               AF(N)=-AF(N):BF(N)=-BF(N)
7370               CF(N)=-CF(N):DF(N)=-DF(N)
7375         NEXT N
7378         IF CENT=1 THEN XC=0 AND YC=0
7380         FOR I=SPL TO EPL
7385            FLAG(I)=0
7390            FOR J=1 TO PL(I,0)
7395                TESTA=(AF(I)*X(PL(I,J)))+(BF(I)*Y(PL(I,J)))
7400                TESTA=TESTA+(CF(I)*(Z(PL(I,J))+D))
7405                IF TESTA>=0 THEN GOTO 7415
7410                   FLAG(I)=1:GOTO 7420
7415            NEXT J
7420         NEXT I
7425         FOR I=SPT TO EPT
7430            XSP(I)=(D*X(I))/(D+Z(I))
7435            YSP(I)=(D*Y(I))/(D+Z(I)):NEXT I
7440         FOR I=SPL TO EPL
7445            IF FLAG(I)=1 THEN GOTO 7480
7450            FOR J=1 TO PL(I,0)
7455                IF J=1 THEN GOTO 7470
7460                CALL MOVABS(XSP(PL(I,(J-1))),YSP(PL(I,(J-1))))
7465                CALL LNABS(XSP(PL(I,J)),YSP(PL(I,J)))
7470            NEXT J
7475              CALL LNABS(XSP(PL(I,1)),YSP(PL(I,1)))
7480         NEXT I
7485         NEXT S:RETURN
7500  REM
7505  REM this will clear the screen
7510  REM
7515       LOCATE 21,1
7520       INPUT "Do you want to clear screen:Y or n ",P$
7525       IF P$ = "n" THEN GOTO 7540
7530       INPUT "Enter color #: " COL%
```

```
7530      INPUT "Enter color #:  ",COL%
7532      CALL SETCOLOR(COL%)
7535      CALL CLR
7540      RETURN
7550  REM
7555  REM restoring shape I.D.
7560  REM
7565      LOCATE 5,50
7570      PRINT "Shapes in memory"
7575      LOCATE 6,50: PRINT "----------------"
7580      LOCATE 7,50: PRINT "1) ";PP$
7585      LOCATE 8,50: PRINT "2) ";Q$
7590      LOCATE 9,50: PRINT "3) ";R$
7595      LOCATE 10,50: PRINT "4) ";S$
7597      LOCATE 11,50: PRINT "5) ";T$
7598      RETURN
7600  REM
7605  REM this will do the calculation for rotation
7610  REM
7615      LOCATE 22,1
7620      INPUT "Enter the number of steps: ",F
7625      AXA=((3.14*AX)/180)/F
7630      AYA=((3.14*AY)/180)/F
7635      AZA=((3.14*AZ)/180)/F
7640      FOR FR=1 TO F
7645          FOR I=W1 TO W2
7650              X1(I)=X(I)
7655              Y1(I)=((Y(I)-YC)*COS(AXA))-((Z(I)-ZC)*SIN(AXA))+YC
7660              Z1(I)=((Y(I)-YC)*SIN(AXA))+((Z(I)-ZC)*COS(AXA))+ZC
7665              X2(I)=((X1(I)-XC)*COS(AYA))+((Z1(I)-ZC)*SIN(AYA))+XC
7670              Y2(I)=Y1(I)
7675              Z2(I)=(-(X1(I)-XC)*SIN(AYA))+((Z1(I)-ZC)*COS(AYA))+ZC
7680              X(I)=((X2(I)-XC)*COS(AZA))-((Y2(I)-YC)*SIN(AZA))+XC
7685              Y(I)=((X2(I)-XC)*SIN(AZA))+((Y2(I)-YC)*COS(AZA))+YC
7690              Z(I)=Z2(I):NEXT I
7695               GOSUB 8320:GOSUB 7220
7700              NEXT FR: RETURN
7750  REM
7755  REM  this will create shape
7760  REM
7765       CLS:PRINT "*"
7770       PRINT "This is create shape.  In this section"
7775       PRINT "you will create your own three dimensional"
7780       PRINT "shape.  Once you create shape, make sure  "
7785       PRINT "to save in Shape Services Menu."
7790       PRINT "----"
7795       INPUT "Enter the number of points in shape: ",NP
7797       PRINT "----"
7800       FOR K=1 TO NP
7805           INPUT "Enter the x,y,z coordinates for point: ",X(K),Y(K),Z(K)
7810       NEXT K
7815       CLS: INPUT "Enter the viewing distance for shape: ",D
7820       INPUT "Enter the number of connecting lines for shape: ",NL
7825       FOR I=1 TO NP
7830           XS(I)=(D*X(I))/(D+Z(I))
7835           YS(I)=(D*Y(I))/(D+Z(I))
7840           CALL PTABS(XS(I),YS(I))
7845       NEXT I
7847       PRINT "----"
7850       FOR K=1 TO NL
7855           INPUT "Enter two numbers to connect lines: ",SP(K),EP(K)
7860           CALL MOVABS(XS(SP(K)),YS(SP(K)))
7865           CALL LNABS(XS(EP(K)),YS(EP(K)))
7870       NEXT K
7875       CLS:PRINT "Now you must enter the planes in shape"
7880       INPUT "Enter the number of planes: ",NPL
```

```
7885        FOR I=1 TO NPL
7890            INPUT "Enter the number of corners pts. for plane: ",PL(I,0)
7895            K=PL(I,0)
7900            FOR J=1 TO K
7905                INPUT "For this plane, enter corner pts.: ",PL(I,J)
7910            NEXT J
7915        NEXT I
7917        NS=1
7918        SPT=1:EPT=NP:SPL=1:EPL=NPL
7920        CLS:PRINT "You have created a shape in three dimensional form."
7925        PRINT "If you would like to see shape in true three de. form,"
7930        PRINT "then proceed to Viewer and Camera Services Menu.  If    "
7935        PRINT "you are not happy with shape then type (r) and start    "
7940        PRINT "over. If you are delighted with shape then proceed to   "
7945        PRINT "Shape Services Menu and SAVE shape by just hiting return."
7950        PRINT "----"
7955        INPUT "Enter (r) to create new shape or hit return: ",Y$
7960        IF Y$ = "r" THEN GOTO 7765
7965        RETURN
8000    REM
8005    REM this is to change color
8010    REM
8015        CLS
8017        PRINT "------"
8020        PRINT "There are up to 256 colors to choose from"
8025        PRINT "The colors will depend on the current palette"
8030        PRINT "----"
8035        PRINT "Enter the color by location on palette"
8040        INPUT "Enter the color for object: ",CL%
8041        PRINT "----"
8042        PRINT "To change background color;go to Clear Screen option "
8043        INPUT "Enter the color for background: ",COL%
8045        CALL SETCOLOR(CL%)
8050        RETURN
8100    REM
8110    REM   setting the line width
8115    REM
8120        CLS:PRINT "*"
8122        PRINT "Line Width"
8124        PRINT "------"
8125        PRINT "The width of a line can between 1 and 1023, where "
8130        PRINT "1 is the thinest line. The number that is being "
8135        PRINT "entered must be odd."
8140        PRINT "----"
8145        INPUT "Enter line width: ",WH%
8150        CALL SETLNWIDTH(WH%)
8155        RETURN
8200    REM
8205    REM   setting the line style
8210    REM
8215        CLS: PRINT "*"
8217        PRINT "Line Style"
8219        PRINT "------"
8220        PRINT "Once a certain line style has been choose, "
8221        PRINT "the color of the object can not be changed "
8222        PRINT "unless the line style is reset to Solid."
8223        PRINT "There are three different types of lines styles"
8225        PRINT " 1) Solid    ------- "
8230        PRINT " 2) Broken   - - - - "
8235        PRINT " 3) Dots     . . . . "
8240        PRINT "----"
8245        INPUT "Enter number of line style: ",STYLE%
8250        CALL SETLNSTYLE(STYLE%)
8255        RETURN
8300    REM
```

```
8305  REM clear screen between animation
8310  REM
8320       CLS: LOCATE 21,1
8325      IF STC=1 THEN GOTO 8385
8330       INPUT "Do you want to clear screen(y or n): ",M$
8340      IF M$ = "n" THEN GOTO 8360
8350       CALL SETCOLOR(COL%)
8355       CALL CLR
8360       INPUT "Do you want to change the color: y or n: ",CC$
8365          IF CC$ = "n" THEN GOTO 8380
8370       INPUT "Enter the new color for object: ",CL%
8375       CALL SETCOLOR(CL%)
8380       RETURN
8385      INPUT "Do you want to clear sreen:y or n: ",WW$
8387          IF WW$ = "n" THEN GOTO 8391
8388       CALL SETCOLOR(COL%)
8389          CALL CLR
8391       CL% = CL% + 1
8392       CALL SETCOLOR(CL%)
8394       RETURN
8400  REM
8405  REM  printer
8410  REM
8415       CLS
8420       PRINT "*"
8425       PRINT "Apple Imagewriter"
8430       PRINT "------"
8435       PRINT "This option will allow the image on screen"
8440       PRINT "to be printed in color.  This option is "
8445       PRINT "not available at this time do to not having"
8450       PRINT "the printer."
8455       FOR I=1 TO 4000:NEXT I
8460       RETURN
8500  REM
8505  REM  printer(ink jet)
8510  REM
8515       CLS
8520       PRINT "*"
8525       PRINT "Tektronic Ink Jet Printer"
8530       PRINT "------"
8535       PRINT "Instructions (please read the entire process before starting)"
8540       PRINT "     1) Insert MSDOS disk with Tektronic program"
8545       PRINT "        in drive A.  "
8550       PRINT "     2) Hit <CTRL>,<ALT>,<DEL> at the same time. "
8555       PRINT "     3) Wait for A> and type  TEK4596 .  "
8560       PRINT "     4) The printer should be hooked up to back of PC. "
8565       PRINT "     5) Follow instructions which will be given to you."
8570       PRINT "  It is very easy to use and can provide some visually "
8575       PRINT "  interesting prints.   "
8580       INPUT "Hit (Return) to return to program: ",KK$
8585       RETURN
8600  REM
8605  REM new calculations for rotation
8610  REM
8615  REM   x
8620  REM
8625        FOR DEGREES=(ANGLE/F) TO ANGLE STEP (ANGLE/F)
8630            RADIANS=(ANGLE*3.14159)/(180*F)
8635            FOR I=W1 TO W2
8640                Y1=(Y(I)-YC)*COS(RADIANS)-(Z(I)-ZC)*SIN(RADIANS)+YC
8645                Z1=(Y(I)-YC)*SIN(RADIANS)+(Z(I)-ZC)*COS(RADIANS)+ZC
8650                Y(I)=Y1:Z(I)=Z1
8655            NEXT I
8660            GOSUB 8320:GOSUB 7220
8665        NEXT DEGREES
8667         STC=0
8670        GOTO 1065
```

```
8870        GOTO 1065
8675  REM
8680  REM   y
8685  REM
8690        FOR DEGREES=(ANGLE/F) TO ANGLE STEP (ANGLE/F)
8695            RADIANS=(ANGLE*3.14159)/(180*F)
8700            FOR I=W1 TO W2
8705                X1=(X(I)-XC)*COS(RADIANS)+(Z(I)-ZC)*SIN(RADIANS)+XC
8710                Z1=(X(I)-XC)*(-SIN(RADIANS))+(Z(I)-ZC)*COS(RADIANS)+ZC
8715                X(I)=X1:Z(I)=Z1
8720            NEXT I
8725            GOSUB 8320:GOSUB 7220
8730        NEXT DEGREES
8733         STC=0
8735        GOTO 1065
8740  REM
8745  REM   z
8750  REM
8755        FOR DEGREES=(ANGLE/F) TO ANGLE STEP (ANGLE/F)
8760            RADIANS=(ANGLE*3.14159)/(180*F)
8765            FOR I=W1 TO W2
8770                X1=(X(I)-XC)*COS(RADIANS)-(Y(I)-YC)*SIN(RADIANS)+XC
8775                Y1=(X(I)-XC)*SIN(RADIANS)+(Y(I)-YC)*COS(RADIANS)+YC
8780                X(I)=X1:Y(I)=Y1
8785            NEXT I
8790            GOSUB 8320:GOSUB 7220
8795        NEXT DEGREES
8797        STC=0
8800        GOTO 1065
8900  REM
8905  REM warning subroutine
8910  REM
8915        CLS:PRINT "*"
8920        PRINT "Warning Message "
8925        PRINT "-------"
8930        PRINT "    The following operation is a TEST. When using "
8935        PRINT " this operation beware that the computer may    "
8940        PRINT " do some strange things and may even completely "
8945        PRINT " shut down.  "
8950        PRINT " -----Sorry, I am not always perfect..... "
8955        PRINT "*"
8960        INPUT "If you want to continue hit return else type (o): ",SD$
8965            IF SD$ = "o" THEN GOTO 3515
8970        RETURN
9000  REM
9005  REM menu for stored shapes
9010  REM
9015        LOCATE 10,55: PRINT "  Shapes Available  "
9020        LOCATE 11,55: PRINT "--------------------"
9025        LOCATE 12,55: PRINT "    1) b:cube1.dat   "
9030        LOCATE 13,55: PRINT "    2) b:cube2.dat   "
9035        LOCATE 14,55: PRINT "    3) b:cool.dat    "
9037        LOCATE 15,55: PRINT "    4) b.cross.dat
9038  GOTO 9090
9040        LOCATE 16,55: PRINT "    5)a. b:hip1.dat "
9045        LOCATE 17,55: PRINT "       b. b:hip2.dat "
9050        LOCATE 18,55: PRINT "       c. b:hip3.dat "
9055        LOCATE 19,55: PRINT "       d. b:hip4.dat "
9090              RETURN
9100  REM
9105  REM   cycle color through palette
9110  REM
9115        STC=0
9120        PRINT "Do you want to run through a specific part "
9125        INPUT "of the color palette:y or n: ",RR$
9130            IF RR$ = "n" THEN GOTO 9170
```

```
9135        INPUT "Enter the starting color of objects: ";CL%
9137        PRINT "-------"
9140        CL% = CL% - 1
9145        STC=1
9170        RETURN
9200  REM
9205  REM
10000 REM
10100 REM these are the assembly commands
10110 REM
10120       CLOSEGRAPHICS    = &H2C3
10130       CLR              = &H2CA
10140       INITGRAPHICS     = &H302
10150       STARTGRAPHICS    = &H452
10160       SETCOLOR         = &H0
10170       SETWORLD         = &H142
10180       SETVIEWPORT      = &H134
10190       SETDEV           = &H261
10200       SETLNSTYLE       = &H51D
10210       SETLNWIDTH       = &H524
10220       POLYLNABS        = &H508
10230       POLYFREL         = &H237
10240       MOVEFROM         = &H372
10250       MOVETO           = &H379
10260       POLYFABS         = &H230
10270       PTABS            = &H7
10280       MOVABS           = &H23
10290       LNABS            = &H15
10300       ZOOM             = &H460
10310       SETTEXT          = &HCB
10320       SETTEXTCLR       = &HD2
10330       INITTCUR         = &H2A7
10340       MOVTCURABS       = &HE7
10350       TEXT             = &HD9
10360       FCLR             = &H2ED
10370       LNREL            = &H1C
10380       MOVREL           = &H2A
10390       RLNABS           = &H206
10400       RLNREL           = &H20D
10410       DELLN            = &H229
10420       SETXOR           = &H1F8
10430       GREAD            = &H1D5
10440       GWRITE           = &H1DC
10450       MOVEFX           = &H380
10460       PWRITE           = &H18F
10470       PREAD            = &H196
10480       MOVETX           = &H387
10490       BAR              = &H28B
10800     RETURN
```

APPENDIX F

Offset for Routines

```
10  '-------------------------------------------------------------------------
20  '  Example.int     Sample program using Halo from Basic interpreter
30  '-------------------------------------------------------------------------
40  ' Running the example program:
50  '    1: MODIFY  the example program, as necessary, according to the comments
60  '               in the program.
70  '    2: EXECUTE the resident driver HALORBI.EXE
80  '    3: EXECUTE the program MEMORY.BAS to determine the correct values
90  '               for the CLEAR and the DEF SEG statements below.
100 '    4: EXECUTE the program EXAMPLE.INT.
110 '
120 '
130 'NOTE: The following HALO files should all be in the current directory -
140 '          MEMORY.BAS  HALORBI.EXE  HALOI.BIN  EXAMPLE.INT
150 '      and a device driver.   i.e. "HALOIBM.DEV"
165 '
170 '-------------------------------------------------------------------------
180                                       ' The CLEAR and DEF SEG statements may
190                                       ' have to be modified according to
200 GOSUB 8000                            ' your hardware.  Run the program
210 DEF SEG = &H3800                    ' MEMORY.BAS to determine the proper
220 BLOAD "HALOI.BIN",0                   ' values.
230 '
240 '
250 '-------------------------------------------------------------------------
260 '  The statements beginning at line 8000 are necessary.
270 '  They define the offsets to the Halo routines.
280 '
290 '-------------------------------------------------------------------------
300 ' The call to SETSEG below must be included if you are using the
310 ' SCION board or the Virtual Rasterization Interface with Halo.
320 ' See the Halo manual for information on setting ADDRESS% .
330 '
340 ' address% = 7                    The SCION board is at segment 7
350 ' call setseg(address%)
360 '-------------------------------------------------------------------------
370 '
380 '                            MAIN PROGRAM
390 '-------------------------------------------------------------------------
400 '  Install the device driver for MULTIHALO
410 '-------------------------------------------------------------------------
420 DEVICE$ = "HALONINE.DEV"              :REM Use the IBM color graphics card
430 CALL SETDEV(DEVICE$)
440 '-------------------------------------------------------------------------
450 '  Init graphics and write message
460 '-------------------------------------------------------------------------
```

```
470 MODE% = 0: CALL INITGRAPHICS(MODE%)
480 X = 0: YMAX = 199: XMAX = 319: Y = 0: CALL SETWORLD(X , YMAX , XMAX , Y)
490 TEXT$ = "This was made with Halo and Basic"
500 TX = (XMAX + 1 - (LEN(TEXT$) * 8)) / 2          :REM Center text on screen
510 TY = (YMAX + 1) / 20
520 CALL MOVTCURABS(TX , TY): CALL TEXT( TEXT$ ): CALL DELTCUR
530 '-------------------------------------------------------------------------
540 '  Move to center and draw pie
550 '-------------------------------------------------------------------------
560 XC = (XMAX + 1) / 2: YC = (YMAX + 1) / 2
570 CALL MOVABS( XC , YC )
580 RADIUS = (XMAX + 1) / 4: ARC1 = .5: ARC2 = 2: CLR% = 1
590 CALL PIE(RADIUS , ARC1 , ARC2 , CLR%)
600 STYLE% = 2: CALL SETHATCHSTYLE(STYLE%)
610 ARC1 = ARC2: ARC2 = 3.5
620 CALL PIE(RADIUS , ARC1 , ARC2 , CLR%)
630 STYLE% = 3: CALL SETHATCHSTYLE(STYLE%)
640 ARC1 = ARC2: ARC2 = 5.78
650 CALL PIE(RADIUS , ARC1 , ARC2 , CLR%)
660 XREL = 20: YREL = 0
670 CALL MOVREL(XREL , YREL)
680 STYLE% = 4: CALL SETHATCHSTYLE(STYLE%)
690 ARC1 = ARC2: ARC2 = .5
700 CALL PIE(RADIUS , ARC1 , ARC2 , CLR%)
710 '-------------------------------------------------------------------------
720 '  Wait for keypress before exiting
730 '-------------------------------------------------------------------------
740 K$ = INKEY$: IF K$ = "" THEN 740
750 CALL CLOSEGRAPHICS: END
8000 ARC                                    = &H85
8001 BAR                                    = &H28B
8002 BOX                                    = &HA1
8003 CFREEZE                                = &H2B5
8004 CGRAB                                  = &H2BC
8005 CIR                                    = &H77
8006 CLOSEGRAPHICS                          = &H2C3
8007 CLR                                    = &H2CA
8008 COMLINE                                = &H467
8009 CSETALUT                               = &H2D8
8010 CSETLUT                                = &H2D1
8011 CSNAP                                  = &H2DF
8012 DEFHATCHSTYLE                          = &HBD
8013 DEFLNSTYLE                             = &H516
8014 DELBOX                                 = &H21B
8015 DELCIR                                 = &H222
8016 DELHCUR                                = &H292
8017 DELLN                                  = &H229
8018 DELTCUR                                = &H299
8019 DISPLAY                                = &H2E6
8020 ELLIPSE                                = &H70
8021 FCIR                                   = &H93
8022 FCLR                                   = &H2ED
8023 FILL                                   = &H9A
8024 FLOOD                                  = &HA8
8025 FLOOD2                                 = &HAF
8026 FTCOLOR                                = &H103
8027 FTEXT                                  = &H10A
8028 FTINIT                                 = &H111
8029 FTLOCATE                               = &H118
8030 FTSIZE                                 = &H11F
8031 GPRINT                                 = &H1B9
8032 GREAD                                  = &H1D5
8033 GSCAN                                  = &H26F
8034 GWRITE                                 = &H1DC
8035 HALLOC                                 = &H46E
8036 HFREE                                  = &H475
```

```
8037 IMREST                = &H2FB
8038 IMSAVE                = &H2F4
8039 INITALPHA             = &H309
8040 INITGRAPHICS          = &H302
8041 INITHCUR              = &H2A0
8042 INITLP                = &H310
8043 INITMARKER            = &H4E5
8044 INITTCUR              = &H2A7
8045 INQAPAL               = &H317
8046 INQARC                = &H8C
8047 INQAREA               = &HB6
8048 INQASP                = &H69
8049 INQBKND               = &H4D
8050 INQCLR                = &H62
8051 INQCRANGE             = &H1AB
8052 INQDEV                = &H19D
8053 INQDISPLAY            = &H325
8054 INQDRANGE             = &H1B2
8055 INQERR                = &H54
8056 INQFT                 = &H126
8057 INQFTCOLOR            = &H12D
8058 INQFUN                = &H31E
8059 INQGCUR               = &H5B
8060 INQHCUR               = &H3F
8061 INQINTERLACE          = &H32C
8062 INQLOCATOR            = &H25A
8063 INQLPA                = &H333
8064 INQLPG                = &H33A
8065 INQMARKER             = &H4EC
8066 INQMODE               = &H341
8067 INQPAL                = &H348
8068 INQPRN                = &H1C0
8069 INQRGB                = &H34F
8070 INQSCAN               = &H276
8071 INQSCREEN             = &H356
8072 INQSTANG              = &H4D7
8073 INQSTAT               = &H35D
8074 INQSTEXT              = &H4C2
8075 INQSTSIZE             = &H4D0
8076 INQTCUR               = &HF5
8077 INQTEXT               = &HE0
8078 INQTSIZE              = &HFC
8079 INQVBW                = &H364
8080 INQVERSION            = &H1A4
8081 INQVIEWPORT           = &H188
8082 INQWORLD              = &H181
8083 LAPPEND               = &H491
8084 LCLOSE                = &H48A
8085 LNABS                 = &H15
8086 LNJOINT               = &H52B
8087 LNREL                 = &H1C
8088 LOPEN                 = &H483
8089 LRECORD               = &H498
8090 LREST                 = &H540
8091 LSAVE                 = &H54E
8092 LSETUP                = &H547
8093 LSWITCH               = &H49F
8094 MAPDTON               = &H17A
8095 MAPDTOW               = &H15E
8096 MAPNTOD               = &H173
8097 MAPNTOW               = &H16C
8098 MAPWTOD               = &H157
8099 MAPWTON               = &H165
8100 MARKERABS             = &H4F3
8101 MARKERREL             = &H4FA
8102 MEMCOM                = &H1E3
```

```
8103
8104 MEMMOV                    = &H1EA
8105 MONO                      = &H1F1
8106 MOVABS                    = &H36B
8107 MOVEFROM                  = &H23
8108 MOVEFX                    = &H372
8109 MOVETO                    = &H380
8110 MOVETX                    = &H379
8111 MOVHCURABS                = &H387
8112 MOVHCURREL                = &H31
8113 MOVREL                    = &H38
8114 MOVTCURABS                = &H2A
8115 MOVTCURREL                = &HE7
8116 ORGLOCATOR                = &HEE
8117 PAN                       = &H253
8118 PEXPAND                   = &H38E
8119 PFNORM                    = &H39C
8120 PIE                       = &H3AA
8121 POLYCABS                  = &H7E
8122 POLYCREL                  = &H532
8123 POLYFABS                  = &H539
8124 POLYFREL                  = &H230
8125 POLYLNABS                 = &H237
8126 POLYLNREL                 = &H508
8127 PREAD                     = &H50F
8128 PROTATE                   = &H196
8129 PTABS                     = &H395
8130 PTNORM                    = &H7
8131 PTREL                     = &H3A3
8132 PWRITE                    = &HE
8133 RBOX                      = &H18F
8134 RCIR                      = &H1FF
8135 READLOCATOR               = &H214
8136 RECOLOR                   = &H24C
8137 REPPAL                    = &H3B1
8138 RLNABS                    = &H3B8
8139 RLNREL                    = &H206
8140 ROAM                      = &H20D
8141 SAVOVL                    = &H3BF
8142 SCROLL                    = &H268
8143 SETAPAL                   = &H3C6
8144 SETASP                    = &H3CD
8145 SETBATTR                  = &H2AE
8146 SETBORDER                 = &H3D4
8147 SETBW                     = &H3DB
8148 SETCAMERA                 = &H3E2
8149 SETCLIP                   = &H3E9
8150 SETCOLOR                  = &H13B
8151 SETCPAL                   = &H0
8152 SETCRANGE                 = &H41A
8153 SETDEGREE                 = &H3F0
8154 SETDEV                    = &H501
8155 SETDRANGE                 = &H261
8156 SETFONT                   = &H3F7
8157 SETGLOCK                  = &H4A6
8158 SETHATCHSTYLE             = &H3FE
8159 SETIEEE                   = &HC4
8160 SETINTERLACE              = &H47C
8161 SETIPAL                   = &H405
8162 SETLATTR                  = &H421
8163 SETLNSTYLE                = &H245
8164 SETLNWIDTH                = &H51D
8165 SETLOCATOR                = &H524
8166 SETMASK                   = &H23E
8167 SETMATTR                  = &H40C
8168 SETPATTR                  = &H413
                               = &H1CE
```

```
8169 SETPRN                    = &H1C7
8170 SETPWD                    = &H4DE
8171 SETRGB                    = &H428
8172 SETSCAN                   = &H27D
8173 SETSCATTR                 = &H284
8174 SETSCREEN                 = &H43D
8175 SETSEG                    = &H46
8176 SETSEG2                   = &H436
8177 SETSTANG                  = &H4C9
8178 SETSTCLR                  = &H4B4
8179 SETSTEXT                  = &H4AD
8180 SETTEXT                   = &HCB
8181 SETTEXTCLR                = &HD2
8182 SETVBW                    = &H444
8183 SETVIEWPORT               = &H134
8184 SETWINDOW                 = &H149
8185 SETWORLD                  = &H142
8186 SETXOR                    = &H1F8
8187 SETXPAL                   = &H42F
8188 SHIFT                     = &H44B
8189 STARTGRAPHICS             = &H452
8190 STEXT                     = &H4BB
8191 TEXT                      = &HD9
8192 VPAN                      = &H459
8193 WORLDOFF                  = &H150
8194 ZOOM                      = &H460
9000 RETURN
```