UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
FACULTAD DE CIENCIAS

GRAFICACION INTERACTIVA EN UN MICROCOMPUTADOR APPLE II

T E S I S
QUE PARA OBTENER EL TITULO DE
A C T U A R I O
PRESENTA
RICARDO ENRIQUE VITE SAN PEDRO
1983

NOTA ACLARATORIA

La presente tesis fue impresa por medio de una computadora, por ello, existe omisión de ciertos signos de nuestro idioma, tal es el caso de los acentos y de la letra eñe.

RECONOCIMIENTOS

Agradezco al Matematico Guillermo Espinosa su valiosa orientacion, direccion y gran paciencia, lo que hizo posible la realizacion de este trabajo

A MIS PADRES

Por formar una gran familia,
gracias a su confianza,
cariño y comprensión.

Por darme siempre la libertad
y el apoyo necesario para
buscar mi propio camino.

A MIS HERMANOS
POSI
ERASMO
LUPE
NACHO
CRISTINA

A CHEMA
LUIS
PATY

A MIS ABUELOS
ANGELINA
NICOLAS

A MAMA RO

A DAVID
SERGIO
NACHO
LUIS
JUAN CARLOS
NELE

A ROLANDO
LEONARDO
ANGEL
FRANCISCO

A TERE

I N D I C E

INTRODUCCION

La graficacion computarizada forma un campo dentro de la computacion, el cual se puede considerar bastante reciente, pues los primeros dispositivos aparecieron a mediados de los años 50's, e inicios de los 60's, teniendo su principal desarrollo, tanto teorico como tecnico, en la decada de los 70's, es por eso que hasta estos momentos no existe una unificacion de criterios que formen la base para el desarrollo del software y hardware, aun cuando en julio de 1980 en el congreso SIGGRAPH se presentaron los estandares en forma teorica faltando ahora el que sean implementados por los fabricantes de equipos de graficacion, es por esto que no se pueden desarrollar paquetes de graficacion en forma jenerica, sino en forma exclusiva para cada marca. Esta situacion motiva que existan discrepancias de dispositivo a dispositivo, esto es, que el mecanismo que se pueda realizar en una unidad, no sea posible aplicarlo en otra

Por otro lado tenemos que hasta hace unos pocos años, los sistemas de computacion con unidades para graficacion, eran tan costosos que solamente la Industria Automotriz, Aereo Espacial, el Gobierno y algunas Universidades podian tenerlos, restringiendo por ello el desarrollo en este campo.

La evolucion de los microprocesadores ocasiono un cambio drastico en el terreno de la graficacion, pues elimino la distribucion elitista al permitir la adquisicion de dispositivos graficadores a precios mas accesibles, pero no ayudo a la unificacion de criterios para el desarrollo, pues nuevamente cada marca posee su propio formato y programacion.

En la graficacion computarizada existen dos areas claramente delimitadas y a las cuales se dedican esfuerzos importantes para su desarrollo. Estas areas son: las graficas no interactivas y las interactivas. Las primeras tienen la caracteristica de que sus dispositivos necesitan un lapso relativamente grande para la elaboracion de las graficas, tal es el caso de los platters o graficadores y las microfilmadoras; las segundas proporcionan el hardware y el software necesarios para establecer una comunicacion instantanea entre el usuario y la computadora, esto es posible gracias a la pantalla graficadora como dispositivo de salida, que permite desplegar varias imagenes por segundo, en sistemas de este tipo, se tienen como instrumentos de entrada dispositivos tales como tabletas, joysticks o lightpens.

La graficacion interactiva es utilizada en un sinnumero de areas entre las cuales podemos enumerar a la Industria Automotriz, Aeronautica, que la utilizan para realizar el diseño de los automoviles, aviones, cohetes, etc.; en la

arquitectura encuentra su aplicación en el diseño de casas y edificios, complejos industriales, también es utilizada por los científicos para representar moléculas y simular reacciones químicas; así mismo, los ingenieros la emplean para el diseño de circuitos electrónicos, con lo cual el tiempo y trabajo destinado al diseño disminuye considerablemente, pues los cambios a las estructuras son hechas directamente a la memoria del computador y en minutos se realizan correcciones complejas, posibilidad que no se podría llevar a cabo sin esta ayuda, pues al dibujar un diagrama de este tipo se llevaría días e incluso semanas y no se elimina la posibilidad de error por parte del dibujante.

La teoría existente es genericamente similar aunque la terminología es propia de cada autor y por otro lado esta orientada exclusivamente a los equipos grandes, los cuales no tienen las limitantes en cuestión de memoria, velocidad, capacidad de proceso, etc. que tienen las microcomputadoras y que hacen que la teoría disponible no sea utilizable directamente en estos equipos, o incluso que no pueda ser aplicada.

Por ello el presente trabajo, intenta encontrar los lazos de unión de la teoría existente y su posible aplicación en un microprocesador, pues a la fecha se carece de literatura enfocada hacia la graficación con microcomputadoras.

El microprocomputador en el que se va a hacer la aplicacion es un Apple II, por ello, el capitulo II ofrece una breve descripcion de las caracteristicas de este microcomputador.

Los capitulos III y IV contienen el estudio de Transformaciones geometricas. Traslacion en el capitulo III y Escala en el IV, en cada uno de ellos hay tres secciones, la primera que se llama Adaptacion a la Apple II ofrece un enfoque de como se puede realizar la transformacion por medio de la Apple basandose en la teoria existente, asi mismo denota los problemas que se presentan debido a las limitaciones y caracteristicas del sistema Apple.

En la seccion llamada Programacion se comentan aquellas rutinas que por su algoritmo o tecnica aplicada se consideran relevantes. Hay que hacer notar que aun cuando las rutinas estan hechas lo mas optimo posible en lo que respecta a recursos y tiempo de ejecucion, se pueden hacer mas optimas, no realizandose esto para no perjudicar la claridad del programa.

Por ultimo se encuentra la seccion Programa y Ejemplo, en la que como su nombre lo indica, se localiza un listado de la Transformacion, ejemplos graficos de ella y relacion de tiempos de ejecucion.

Al final del trabajo se incluyen una serie de anexos a los cuales se hace referencia en el transcurso de este trabajo.

II APPLE II

La Apple II es un microcomputador dotado de un microprocesador 6502, cuya velocidad es de 1,023,000 ciclos de maquina por segundo y puede hacer hasta quinientas mil sumas o restas en un segundo. Su longitud de palabra es de ocho bits y puede direccionar hasta 65,536 bytes (64k bytes). Proporciona 56 instrucciones en lenguaje ensamblador con 13 modos de direccionamiento (una lista de las instrucciones y modos de direccionamiento puede encontrarse en el Apendice A).

La memoria de la Apple contiene tanto ROM (Read Only Memory) como RAM (Random Access Memory), en el primero se encuentran programas tales como el System Monitor, Apple Autostart ROM, Integer Basic, Applesoft II Basic y el Programmer's Aid # 1 que tiene subrutinas de utileria.

En RAM se puede tener de 16k a 48k que es la maxima configuracion y que junto con el ROM forman los 64k bytes que puede direccionar el 6502.

El RAM es utilizado para almacenar todos los programas y

datos que se manejan en cualquier momento, esta incluye tambien los datos y variables utilizados por los sistemas residentes en RAM, los cuales ocupan 2% de RAM (Ver seccion II.1).

La informacion contenida dentro del RAM es perdida una vez que se apaga la maquina, por ello siempre es conveniente contar con alguna unidad de almacenamiento secundario, para esto se cuenta con casettes y diskette, siendo necesario al utilizar este ultimo contar con un sistema operativo residente en RAM, lo cual disminuye el espacio disponible en RAM en 10.5k.

En la Apple II, existen dos niveles de graficacion, uno llamado Low Resolution Graphics que tiene una resolucion de 40 por 48 con 16 colores y High Resolution Graphics con una resolucion de 280 por 192 con 6 colores.

No siendo posible realizar una grafica conjugando ambos metodos y considerando a Hi-Res (High Resolution Graphics) como el mas util de ambos niveles por tener una mayor densidad de puntos graficables, en lo sucesivo se hablara unicamente de Hi-Res.

Hi-Res, posee dos paginas en las cuales se puede trabajar, cada una de estas paginas necesita de 8k para almacenar su

información de los 53,760 puntos que puede desplegar en la pantalla, a esta area de memoria la llamaremos "picture buffer", para ser consistentes con los manuales de la Apple II. O sea, que se tienen dos pictures buffers, uno para la primera pagina y otro para la segunda. La primera pagina reside desde la localidad 8192 hasta la localidad 16383, (en base hexadecimal se representaria de la 2000 a la 3FFF), la segunda pagina se encuentra de la localidad 16384 a la 24575 en (hexadecimal de la 4000 a la 5FFF).

El formato que se utilita para representar cada punto de la pantalla en la memoria es el siguiente: cada punto de la pantalla esta representado por un bit dentro del picture buffer, siete de los ocho bits en cada byte corresponden a puntos e la pantalla, y el bit restante es utilizado para representar el color de los puntos en ese byte (ver abajo), con esto tenemos que cuarenta bytes son desplegados en cada linea de la pantalla llegando con esto a los 280 puntos, estas 280 posiciones se encuentran numeradas del 0 al 279, teniendo igual numero de columnas pares, que de columnas nones.

El bit menos significativo (bit 0) del primer byte corresponde al primer punto de la izquierda de la linea, esto es, se encuentra sobre la columna cero, el siguiente bit (bit 1) desplegara al punto sobre la columna uno, el punto en la columna dos es representado por el bit 2, etc., llegando al bit 7 que no

es desplegado en la pantalla, teniendo que continuar con el primer bit del siguiente byte y asi sucesivamente.

En el computo del color se considera lo siguiente: si un bit esta apagado (igual a cero), su color siempre sera negro, si el bit esta prendido (igual a uno), su color dependera de la posicion en la pantalla.esto es, si el punto en una columan par aparece en color violeta, y si se encuentra en cualquier columna non sera de color verde. Si dos puntos consecutivos sobre un mismo renglon son prendidos (uno verde y otro violeta), ambos aparecerán como blancos.

Ahora, si el octavo bit se encuentra prendido, los colores verde y violeta son sustituidos por el rojo y el azul respectivamente. Si deseamos utilizar los seis colores en Hi-Res, nos vemos sujetos a las siguientes restricciones:

1) Los puntos en columnas pares pueden ser negros, violetas o azules

2) Puntos en columnas nones desplegaran los colores negro, verde o rojo

3) Cada byte solo puede ser un byte violeta/verde o un byte azul/rojo. No es posible mezclar verde y azul, verde y rojo, violeta y azul o violeta y rojo en un mismo byte, esto es debido a que el octavo bit permite desplegar los colores negro, violeta, verde y blanco con un byte al encontrarse apagado, o los colores negro, azul, rojo y blanco si se encuentra prendido.

4) Dos puntos coloreados y consecutivos en una linea siempre apareceran como blancos, aun cuando sean de diferentes bytes

Debido a estas restricciones, podemos decir primeramente que si utilizamos colores, la resolucion en Hi-Res es de 140 x 192 no como se especifica en la documentacion de la Apple II, pues aunque tenemos 280 columnas para trabajar, solo la mitad de ellas pueden desplegar el color verde, violeta, azul o rojo. Esto problema se hara mas palpable cuando se presenten las subrutinas de traslacion y escalamiento. Por otro lado, si trabajamos unicamente con blanco y negro, entonces si tendremos la solucion de 280 x 192. De esto podemos decir que tenemos dos modos, el modo de color y el modo blanco y negro, en lo sucesivo todo lo que se exponga partira de la suposicion de que se trabaja en modo color. Esta suposicion implica mayor labor, sobre todo en el caso de la escala, pero considero de utilidad, pues con esta suposicion tambien se engloba en cierta medida al modo blanco y negro

Considerando nuevamente las restricciones y enfocando el modo color, podemos decir que dentro de este modo, existen dos gamas, la gama de colores negro, violeta, verde y blanco (bit siete apagado) y la gama negro, azul, rojo y blanco (bit siete prendido). Esto se hace notar, debido a que al aplicar las transformaciones a las figuras representan graves obstaculos, a los cuales se hara mencion en el momento adecuado.

Hi-Res puede ser utilizado desde Applesoft o desde Integer Basic, aunque en este ultimo en forma un poco mas compleja, debido a que no cuenta con comandos directos como el caso de Applesoft, sino que la graficacion se realiza con llamadas a subrutinas en ensamblador, y pasando los parametros en localidades especificas, y no es posible realizarla si no se cuenta con el ROM llamado Programmer's Aid # 1, el cual contiene las subrutinas necesarias para graficar en High Resolution Graphics.

Considerando que el presente trabajo esta enfocado hacia la graficacion interactiva, es de vital importancia tomar en cuenta los tiempos, por ello todo lo subsiguiente es utilizable unicamente desde Integer Basic y el ROM llamado Programmer's Aid # 1, debido a que Integer Basic es mas rapido que Applesoft, en el apendice C se representan dos programas, uno en Integer Basic y otro en Applesoft, que realizan exactamente lo mismo, mostrando para cada uno de ellos sus tiempos de ejecucion. Por

otro lado la documentación para el código del ensamblador que realiza HiTech se encuentra disponible únicamente para el código contenido en el Programmer's Aid # 1.

II.1 MAPA DE LA MEMORIA

System Memory Map		
Page Number: Decimal	Hex	
0	$00	RAM (48K)
1	$01	
2	$02	
.	.	
.	.	
.	.	
190	$BE	
191	$BF	
192	$C0	I/O (2K)
193	$C1	
.	.	
.	.	
.	.	
198	$C6	
199	$C7	
200	$C8	I/O ROM (2K)
201	$C9	
.	.	
.	.	
.	.	
206	$CE	
207	$CF	
208	$D0	ROM (12K)
209	$D1	
.	.	
.	.	
.	.	
254	$FE	
255	$FF	

Table 16: RAM Organization and Usage			
Page Number: Decimal	Hex	Used For:	
0	$00	System Programs	
1	$01	System Stack	
2	$02	GETLN Input Buffer	
3	$03	Monitor Vector Locations	
4	$04	Text and Lo-Res Graphics Primary Page Storage	
5	$05		
6	$06		
7	$07		
8	$08	Text and Lo-Res Graphics Secondary Page Storage	FREE RAM
9	$09		
10	$0A		
11	$0B		
12 through 31	$0C $1F		
32 through 63	$20 $3F	Hi-Res Graphics Primary Page Storage	
64 through 95	$40 $5F	Hi-Res Graphics Secondary Page Storage	
96 through 191	$60 $BF		

Table 17: ROM Organization and Usage			
Page Number: Decimal	Hex	Used By:	
208	$D0	Programmer's Aid #1	Applesoft II BASIC
212	$D4		
216	$D8		
220	$DC		
224	$E0	Integer BASIC	
228	$E4		
232	$E8		
236	$EC		
240	$F0		
244	$F4	Utility Subroutines	
248	$F8	Monitor ROM	Autostart ROM
252	$FC		

System Memory Map

Page Number: Decimal	Hex	
0	$00	RAM (48K)
1	$01	
2	$02	
.	.	
.	.	
.	.	
150	$DE	
191	$BF	
192	$C0	I/O (2K)
193	$C1	
.	.	
.	.	
.	.	
198	$C6	
199	$C7	
200	$C8	I/O ROM (2K)
201	$C9	
.	.	
.	.	
.	.	
215	$CE	
215	$CF	
216	$D0	ROM (12K)
217	$D1	
.	.	
.	.	
.	.	
254	$FE	
255	$FF	

Table 16: RAM Organization and Usage

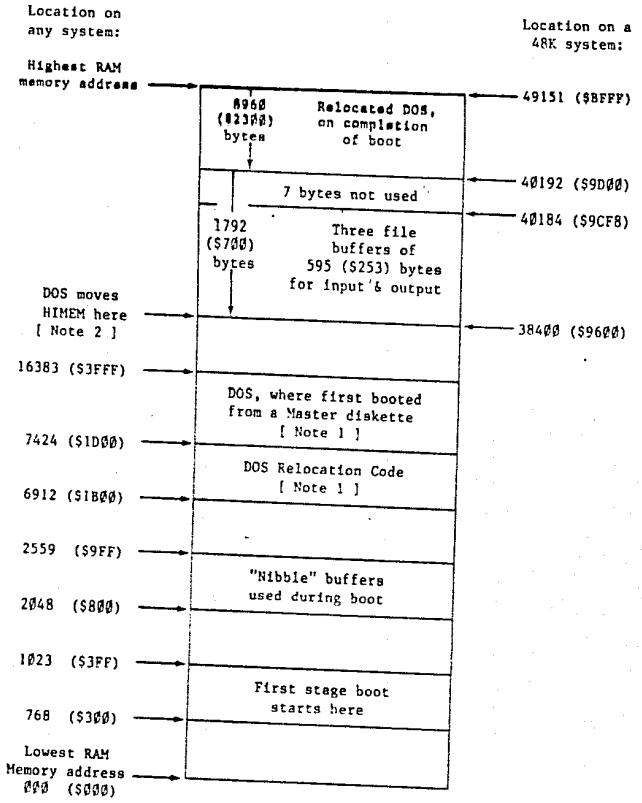
Page Number: Decimal	Hex	Used For:	
0	$00	System Programs	
1	$01	System Stack	
2	$02	GETLN Input Buffer	
3	$03	Monitor Vector Locations	
4	$04	Text and Lo-Res Graphics Primary Page Storage	
5	$05		
6	$06		
7	$07		
8	$08	Text and Lo-Res Graphics Secondary Page Storage	
9	$09		
10	$0A		
11	$0B		
12 through 31	$0C through $1F		FREE RAM
32 through 63	$20 through $3F	Hi-Res Graphics Primary Page Storage	
64 through 95	$40 through $5F	Hi-Res Graphics Secondary Page Storage	
96 through 191	$60 through $BF		

Table 17: ROM Organization and Usage

Page Number: Decimal	Hex	Used By:	
208	$D0	Programmer's Aid #1	Applesoft II BASIC
212	$D4		
216	$D8		
220	$DC		
224	$E0	Integer BASIC	
228	$E4		
232	$E8		
236	$EC		
240	$F0		
244	$F4	Utility Subroutines	
248	$F8	Monitor ROM	Autostart ROM
252	$FC		

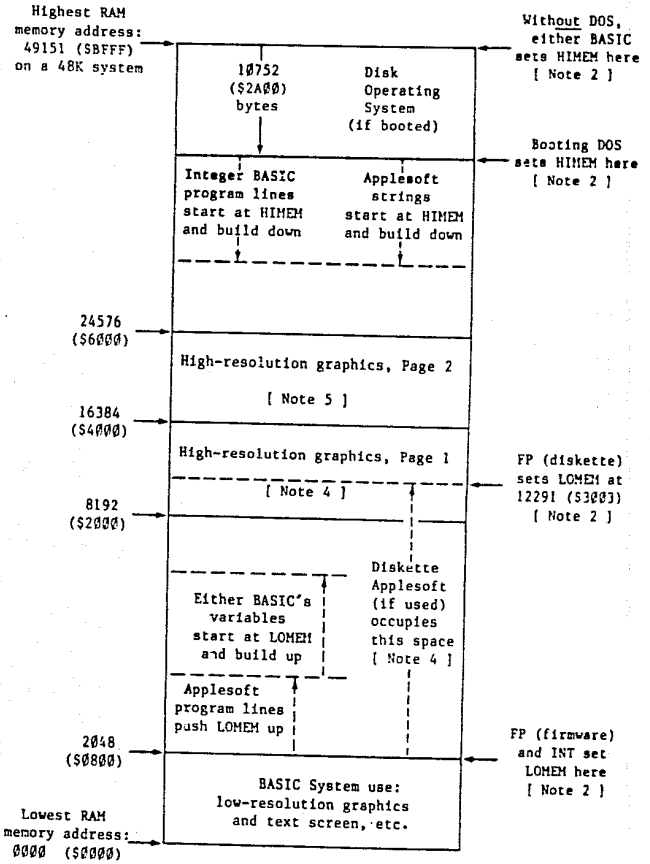
TABLE 1: APPLE II MEMORY MAPS

A. MEMORY AREAS OVER-WRITTEN WHEN BOOTING DOS



Note 1. This memory area is not affected when booting a Slave diskette: DOS is placed directly below the Highest RAM Memory address that was available on the system that INITialized the Slave diskette, whether appropriate to the present system or not.

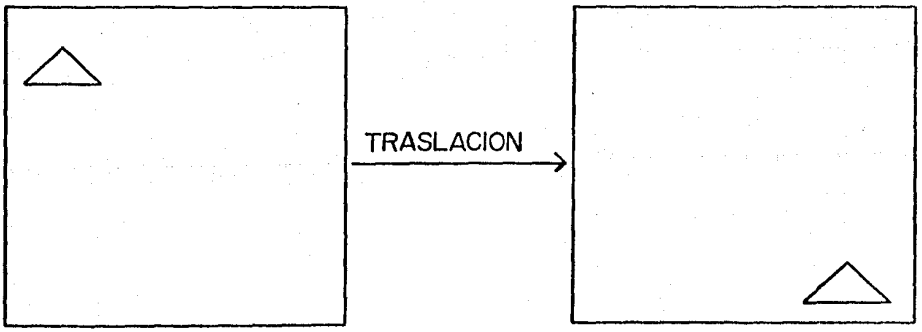
B. MEMORY AREAS USED BY DOS AND EITHER BASIC



Note 2. If your system is in Integer BASIC, the HIMEM pointer can be found (low byte first, then high byte) in locations 76-77 ($4C-$4D). If your system is in APPLESOFT BASIC, the HIMEM pointer is in locations

II. TRASLACION

• Supongamos que tenemos desplegada en nuestra pantalla una grafica, la cual deseamos cambiar de lugar, pero sin modificar su tamaño ni orientacion. Supongamos, por ejemplo que tenemos un triangulo en la parte superior izquierda de la pantalla y deseamos pasarlo al angulo inferior derecho, esto se realiza mediante una transformacion geometrica que se denomina **Traslacion.**



La traslacion es una transformacion que a cada punto de

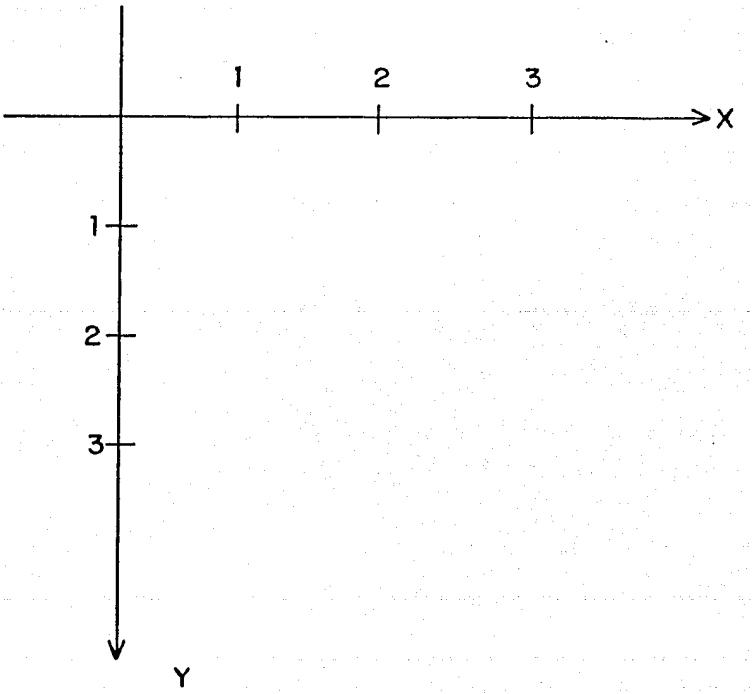
coordenadas cartesianas (X ,Y), la manda a otro punto (X',Y') donde X' es el resultado de sumar un valor constante al valor X y Y' es igual al valor Y mas una constante, de donde obtenemos la ecuacion de la traslacion que es:

$$X' = X + C1$$

$$Y' = Y + C2 \text{ Donde } C1, C2 \text{ son constantes}$$

III.1 ADAPTACION A LA APPLE II

La primera consideracion importante para cualquier aplicacion de tipo grafico a un microcomputador Apple II, es que el eje de las Y's esta invertido, esto es, la coordenada (0,0) se encuentra en el angulo superior izquierdo y el punto (0,191) se localiza en el angulo inferior izquierdo, quedando los ejes de la siguiente forma:



La importancia del hecho reside en que si los parametros de la traslacion (incrementos a cada coordenada) son proporcionados tecleando los valores y no indicando dos puntos en la pantalla que sean un punto de referencia y su punto transformado, esto podria ocasionar confusiones en el sentido de que si deseamos subir una figura, tendremos que proporcionar un valor para y negativo, en lugar de positivo.

De los dos metodos expuestos anteriormente para proporcionar los parametros de la traslacion, no podemos decir que uno sea mejor que otro, pues aun cuando el metodo de indicar los dos puntos en la pantalla sea mas ilustrativo a primera vista, nos topamos con la estructura del picture buffer y con el mecanismo de la seleccion del color, lo que hace restarle funcionalidad a este metodo y dar mayores creditos al mecanismo de teclear los valores. Esto es debido a que si se utilizan los seis colores disponibles (se habla de seis colores en lugar de ocho, debido a que existen dos negros y dos blancos), la traslacion tendra que realizarse en multiplos pares de siete, esto es, 14, 28, 42, etc. o sea multiplos de 14, porque de hacerlo de otra manera se cambiarian los colores, por ejemplo: si la traslacion la realizamos en cualquier multiplo non de 7, el color verde se cambiara a violeta, el rojo al azul y viceversa, teniendo esto como explicacion el que un punto en la columna par pase a una non y una non a la par, ocasionando con ello que el color se cambie por su complemento, puesto que el bit 7 permanecera inalterado.

En el caso de realizar la traslacion sin tomar en cuenta a los multiplos de siete, el problema seria aun mayor, por que en el caso anterior, todo lo que estaba en verde, se cambia a violeta, todo lo rojo a azul, lo violeta a verde y lo azul a rojo, pero en este caso solo una parte del verde cambiaria y la otra continuaria siendo verde, deformando con esto, las fronteras originales entre los colores.

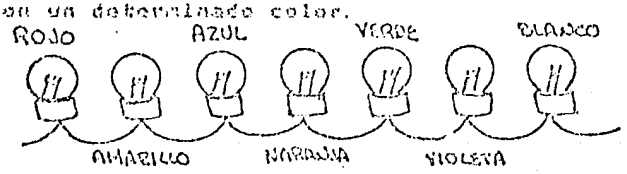
El color al que cambia, depende de dos factores, el primero es si el parametro es par o es non y el segundo depende a que byte vaya a ser trasladado cada punto en cuestion, el cual puede tener el bit 6 prendido o apagado. La siguiente tabla nos muestra el cambio de colores dependiendo de cada condicion.

	PARAMETRO PAR		PARAMETRO NON	
	BYTE CON BIT 8 = 1	BYTE CON BIT 8 = Ø	BYTE CON BIT 8 = 1	BYTE CON BIT 8 = Ø
BLANCO	BLANCO	BLANCO	BLANCO	BLANCO
NEGRO	NEGRO	NEGRO	NEGRO	NEGRO
VERDE	ROJO	VERDE	AZUL	VIOLETA
VIOLETA	AZUL	VIOLETA	ROJO	VERDE
ROJO	ROJO	VERDE	AZUL	VIOLETA
AZUL	AZUL	VIOLETA	ROJO	VERDE

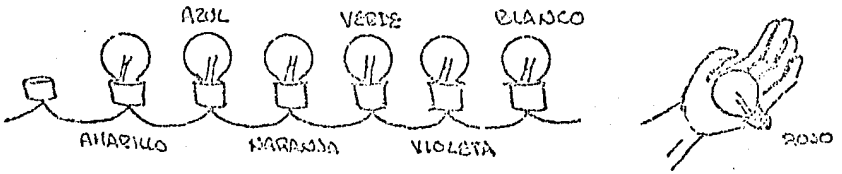
III.2 PROGRAMACION

Al disenar el programa se tomo en cuenta si la traslacion se realizaba en multiplos de siete o no, esto es, por palabras completas o por bits. Si se realizaba por multiplos de siete (palabras completas) el tiempo de ejecucion era reducido y el codigo era minimo. Pero si deseabamos trasladar bits, el tiempo de ejecucion aumentaria en un poco mas de la mitad y tambien el codigo necesario seria mayor.

Por otro lado, al tratar de utilizar el minimo de recursos, se propuso emplear la pagina uno de Hi-Res solamente, pero eso traia como consecuencia, el aumentar el codigo de la traslacion, situacion que no ocurria al trabajar con ambas paginas. Lo anterior era debido a que dependiendo de los signos de los parametros a la traslacion el recorrido en el picture buffer debia hacerse en direcciones determinadas si no se deseaba perder informacion, esto es facil ilustrarlo si suponemos que un renglon del picture buffer es una serie de foquitos navidenos, cada uno con un determinado color.

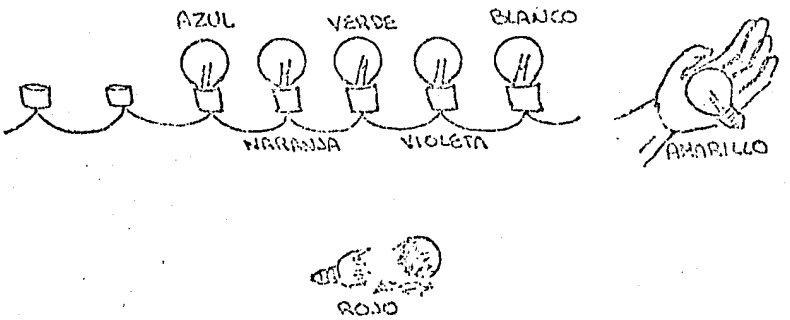


Al recorrer cada foquito una posicion a la derecha teniendo la limitante de tener a lo mas un foco en las manos y haciendo la traslacion de izquierda a derecha, obtendriamos lo siguiente:



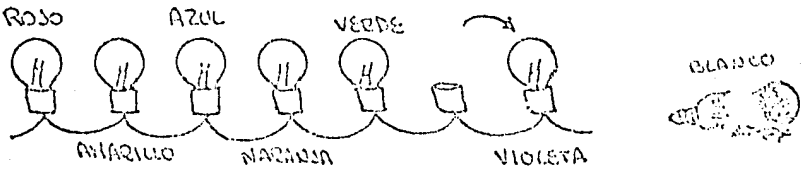
Primero quitariamos el foco rojo de su lugar, para tratar de ponerlo en la segunda posicion (lugar del foco amarillo).

Como segundo paso, quitariamos el foco amarillo para poner el rojo, pero como solo podemos sostener un foco con la mano, el foco rojo se cae y se rompe.



Tenemos el foco amarillo pero ya perdimos el foco rojo con lo cual no realizariamos nuestro trabajo satisfactoriamente.

Ahora, si nosotros en lugar de tomar el foco rojo como primer paso, desconectamos el foco blanco y lo tiramos para tener las manos vacias, podemos desconectar el foco violeta y conectarlo en el lugar del foco blanco sin tener problema.



Y continuar de la misma manera con el verde, naranja, azul, morado y rojo finalizando con ello nuestro trabajo.

Al hacer la similitud de que los focos sean los bytes, y la cara al acumulador, vemos que dependiendo de la direccion en que recorramos el picture buffer, realizaremos el trabajo correctamente o perderemos informacion.

Es ahi que dependiendo de los signos de los parametros, es como debemos hacer el recorrido, de abajo hacia arriba si tenemos un valor positivo en el incremento de la Y, de arriba hacia abajo si el incremento de Y es negativo, de derecha a izquierda cuando el desplazamiento en X sea positivo y de izquierda a derecha cuando sea negativo, mostrando esto en el siguiente cuadro.

VALOR DEL PARAMETRO	DIRECCION DEL RECORRIDO
Y >= 0	abajo hacia arriba
Y < 0	arriba hacia abajo
X >= 0	derecha a izquierda
X < 0	izquierda a derecha

Hay que recordar nuevamente que esto se puede eliminar, si se utilizan ambas paginas de Hi-Res para la traslacion, teniendo que hacer una evaluacion de que nos conviene mas, si el trabajar con una sola pagina o con dos.

En nuestro caso, consideramos mas conveniente emplear una sola pagina, pues aun cuando el codigo es mas extenso, el espacio en memoria ocupado por el es menor que el picture buffer.

Asi mismo, unicamente se desarrollo la programacion para el caso en que los dos parametros sean positivos, debido a que el presente trabajo, persigue como finalidad principal, el hacer resaltar el cambio de razonamiento necesario para trabajar con un microcomputador y no el desarrollo de los programas en si.

Por otro lado, al realizar la programacion para el caso en que el incremento de X no es multiplo de siete, se partio de la suposicion de que en la pantalla solo se tenian una gama de color, esto es, la gama de colores cuando el bit ocho se encuentra apagado (negro, verde, violeta, blanco), o cuando esta prendido (negro, roja, azul, blanco), esto es logico pensar pues si se tienen ambas gamas de colores, al realizar una traslacion de este tipo, la distincion de colores seria importante.

Debido a la estructura que posee el picture buffer, fue necesario crear dos tecnicas diferentes para realizar la traslacion.

En primer lugar se procedio a dividir el incremento sobre el eje de las x's entre siete y analizar su resultado y el residuo. Si el residuo es diferente de cero, implica la necesidad de transportar un numero menor de siete bits de una palabra a otra, este procedimiento corresponde a la primera tecnica, que consiste en lo siguiente:

Se toma la primera palabra que esta afectada por la traslacion y se le aplica una instruccion RCL (Rotate Left), misma que inserta el carry en el primer bit, recorre los demas hacia la izquierda y pone el octavo bit en el carry, este bit lo pierde el programa debido a que es el bit de control. Efectua

un segundo ROL quedando el carry con un bit de información que es transportado a la siguiente palabra mediante otro ROL, mismo que pondrá el bit de control de esta última en el carry, que nuevamente es perdido por el programa. Se vuelven a reiniciar los ROL's a partir de la primera, pasando un bit de información a la segunda palabra y esta a la tercera que dejara su bit de control en el carry, así sucesivamente hasta realizar tantos reinicios como valor tenga el residuo y en cada reinicio va incorporando una nueva palabra que pierde su bit de control, posteriormente continua con este mismo procedimiento, pero el reinicio se efectua en una palabra posterior cada vez hasta el final del renglon.

Una segunda etapa consiste en poner a cada palabra su bit de control, es en esta parte donde se supone que se utiliza una sola game en los colores, pues una vez conocido el valor del bit de control, a partir del color de fondo, este se incluye en cada palabra.

Una vez trasladados los bits de una palabra a otra en toda la pagina de HIRES, se aplica la segunda tecnica que consiste en trasladar una palabra completa de una posicion a otra, cubriendo toda la pagina.

El direccionamiento de la pantalla en HIRES, se realiza con

una rutina que recorre los renglones de abajo hacia arriba, utilizando para ello, una variable que se le asigna un valor de 192 que es el total de renglones, esta variable se va decrementando y dependiendo de los valores que tome, es el incremento que se daría al apuntador, esto es, si el residuo de la variable entre 64 es igual a cero se dara un incremento de 8074 (1F58 en hexadecimal) al apuntador, si el residuo de la variable entre 8 es igual a cero se tendra un incremento de 7040 (1B80 en hexadecimal), en caso contrario, se tendra un incremento de 1024 (400 en hexadecimal).

Hay que hacer notar que 2 ** 6 = 64 y que 2 ** 3 = 8 con lo cual, si expresamos estos numeros en binario y hexadecimal seria:

64 = 100,0000 = 40

8 = 1000 = 8

Ahora, si deseamos analizar el residuo, hacemos un AND de la variable y 3F para el caso de 64 y 07 para 8 y analizamos si el resultado es cero, eso implica que el residuo es igual a cero.

Los incrementos o decrementos en los apuntadores, son debido a la estructura del picture buffer (Apendice C), en el cual podemos ver en forma inicial que existen unos incrementos

de 1024 que van desde 0 a 7168, o sea que sirven para el calculo de ocho renglones consecutivos. En la parte principal del diagrama de picture buffer entre subseccion y subseccion, hay un incremento de 128 en ocho ocasiones, despues de las cuales existen otras dos secciones cuyo incremento entre subseccion y subseccion es de 128, o sea que tenemos 3 secciones de 64 renglones.

Con esto tenemos que:

8 renglones con incrementos de 1024
x 8 subsecciones con incrementos de 128
x 3 secciones
192 renglones totales en el eje

Ahora bien, si deseamos pasar del primer renglon de la segunda subseccion (posicion 8320), al renglon anterior que es el ultimo renglon de la primera subseccion (posicion 8192 + 7168 = 19360), tenemos que realizar un incremento de 7040 (19360 - 8320 = 7040). Por otro lado, si pasamos del primer renglon de la segunda seccion (posicion 8232) al anterior (posicion 9088 + 7168 = 16256), tenemos que realizar un incremento de 8024 (16256 - 8232 = 8024), que son precisamente los incremetos de que hablamos hablado anteriormente.

Variando adecuadamente las variables de contadores y apuntadores fue posible mediante una sola rutina llevar a cabo el calculo para cada pagina, utilizando el modo de direccionamiento absoluto indexado con el registro de trabajo X. Mediante esta forma si la X = 0 nos estamos refiriendo a la pagina uno de HIRES y si la X = 2 estamos trabajando con la pagina dos

III.3 PROGRAMA Y EJEMPLOS DE LA TRASLACION

En las siguientes paginas se puede ver el listado en ensamblador 6502 para la rutina de traslacion. El codigo de esta rutina es puesto a partir de la localidad 0C00 Hex.

Posterior al listado se podran encontrar algunas graficas, la primera de ellas es un grafico utilizado en las pruebas de los programas, hay que aclarar que dichas graficas estan rotadas 90 , esto es, el eje X se encuentra en forma vertical a la izquierda de la hoja, y el eje Y esta en forma horizontal en la parte interior.

La segunda grafica muestra el resultado de aplicar una traslacion con parametros (21,40), la apreciacion se puede hacer si nos fijamos en la parte superior derecha de la hoja.

La tercera grafica nos muestra el ejemplo de una traslacion con parametros (70,0).

Como se había mencionado anteriormente la traslación posee las técnicas, mismas que se emplean dependiendo de los parámetros, debido a esto los tiempos de ejecución varían de acuerdo a los parámetros y por consiguiente a las técnicas que se utilicen, a continuación se muestra una relación de tiempos de acuerdo a los parámetros y técnicas aplicadas.

PARÁMETROS	TÉCNICAS APLICADAS	TIEMPO DE EJECUCIÓN
(14,30)	2a.	0.6 seg.
(20,30)	1a y 2a.	3.0 seg.

III 3 PROGRAMA Y EJEMPLOS DE LA TRASLACION

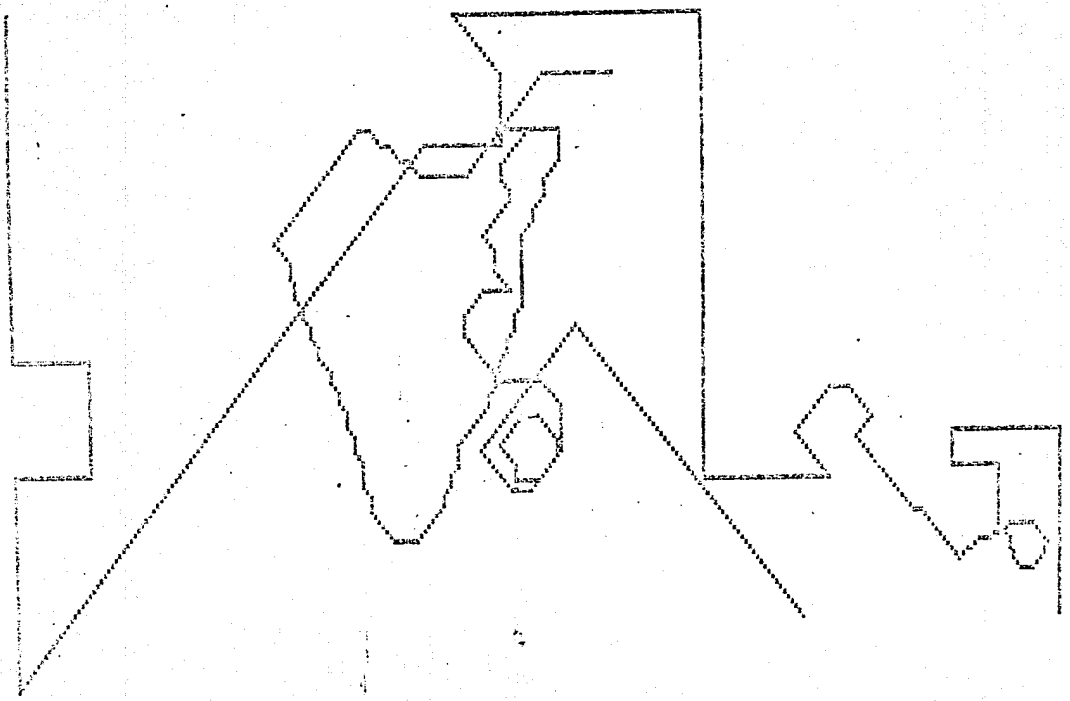
```
'REM PROGRAMA QUE LLEVA A LA RUTINA DE
   TRASLACION
'CLEAR
   10 POKE 23635,PEEK(23635)+PEEK(23636)
      *256+250
   20 POKE 23636,99: POKE 65535,16
   25 AA=USR
   30 DIM A$(5)
   35 POKE 65535,0
   37 PRINT "CARGAR TRAZO"
   40 INPUT "QUIERES CAMBIAR ALGUNA FIGU
      RA? ";A$: IF A$="NO" THEN GOTO
      100
   50 PRINT "BLOAD ";A$
   60 CALL 2437
  100 INPUT "PARAMETROS DE LA TRASLACI
      ON (DX,DY) ";XD,YD
  110 CALL 2048
  120 END
```

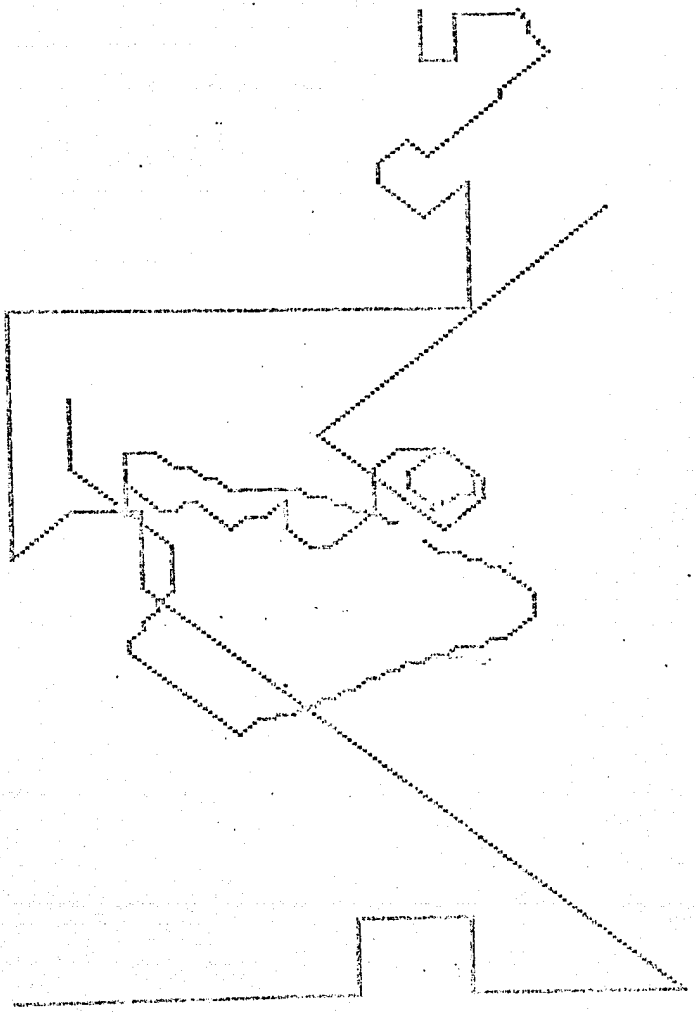
```
0010  LWAX  DL 0006     PARAMETROS DE
0020  HGAX  DL 0007     TRASLACION X
0030  AY    DL 0008     (LOW,HIGH),Y
0040  Q     DL 0009     Q = X DIV 7
0050  MODX  DL 000A     MODX = AX MOD 7
0060  FORK  DL 000B     VAR. LOOP K
0070  FON   DL 000D     MSC DEL FONDO
0080  I192  DL 0019     CONT RENGL PAG1
0090  LPAG  DL 001A     DIREC INICIAL
0100  HPAG  DL 001B     RENGL HIRES PG1
0110  FORI  DL 001C     CONT RENGL PAG2
0120  LPGD  DL 001D     DIREC RENGL PG2
0130  HPGD  DL 001E     HIRES DESTINO
0140  LIMI  DL 90EB     LIMITE LOOP I
0150  BICO  DL 00F8     BITS 7 HIRES
0160  BKGN  DL 0323     COLOR FONDO
0170  GETX  DL D290     RUTINA OBT AX
0180  GETY  DL D3AF     RUTINA OBT AY
0190  ;*********************************
0200  ;* RUTINA DE TRASLACION HI-RES  *
0210  ;*********************************
0220        JSR GETX     OBT EL INC DE
0230        STX *LWAX    LA TRASLACION
0240        STY *HGAX    A PARTIR DE VAR
0250        JSR GETY     EN INT. BASIC
0260        STA *AY
0270        LDA *HGAX
0280        JSR DIV7     DIV  AX / 7
0290        STA *MODX
0300        LDA *MODX
0310        BEQ SIG      SI AX MOD 7 >0
0320        JSR MUBI     MUEVE BITS
0330  SIG   LDA *0
0340        BNE MD>0     SI AX DIV 7 >0
0350        LDA *AY      OR AY >0
0360        BEQ RET
0370  MD>0  JSR MUPA     MUEVE PALABRAS
0380  RET   RTS
0390  ;=================================
0400  MUBI  LDA 0D0      RECORRE HI-RES
0410        STA *LPAG    DE ABAJO HACIA
0420        LDA 03F      ARRIBA. SALTAN-
0430        STA *HPAG    DO LOS RENGL
0440        LDA 0C0      QUE NO SE VAN A
0450        STA *I192    TRASLADAR
0460        LDX 000
0470        LDY *AY
0480  LOOP  BEQ TERM
0490        JSR REG
0500        DEY
0510        JMP LOOP
0520  TERM  JSR MURE     MUEVE BITS EN
0530        LDX 000
0540        JSR REG
0550        BNE TERM
0560        RTS
0570  ;=================================
0580  MURE  JSR BIT7     QUITA BITS 7
0590        LDA BKGN     DEL RENGLON
0600        STA *FON
0610        LDA 000      FOR I=0 TO MODX
0620        STA *LWAX     -1
0630        STA *FORI
0640        LDX *MODX
0650        ASL *FON      ASL FON
0660  LOP0  LDY 000      FOR Y=0 TO I
0670        ASL *FON
0680        ROL *LWAX     ROL (LPAG),Y
0690  LOP1  ROR *LWAX     TENIENDO EN EL
0700        LDA (LPAG),Y  CARRY EL BIT A
0710        ROL           TRANSFERIR.
0720        STA (LPAG),Y  QUE SE ENCUEN-
```

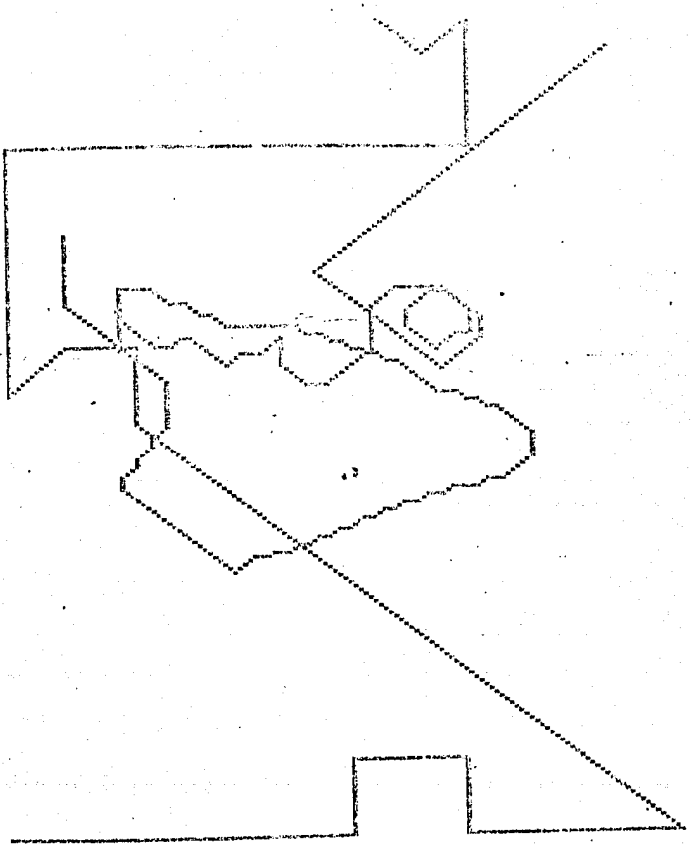
```
0730        ROL *LWAX      TRA EN EL BIT
0740        INY            0 DE LWAX
0750        CPY *FORI
0760        BMI LOP1
0770        SED LOP1
0780        INC *FORI
0790        CPX *FORI
0800        BNE LOP0       ---------------
0810        LDA 028
0820        SEC
0830        SBC #0
0840        STA *LIMI      LIMI=40-0
0850        LDA 000        FOR I=0 TO LIMI
0860        STA *FORI      -1
0870        INX
0880  LOP2  LDA 000
0890        STA *FORK
0900        STA *LWAX
0910  LOP3  CLC
0920        ADC *FORI      FOR K=0 TO MODX
0930        CMP *LIMI
0940        BED CAM2
0950        ROR *LWAX
0960        TAY            ROR(LPAG),I+K
0970        LDA (LPAG),Y
0980        ROL
0990        STA (LPAG),Y
1000        ROL *LWAX
1010        INC *FORK
1020        LDA *FORK      NEXT K
1030        CPX *FORK
1040        BNE LOP3
1050  CAM2  LDY *FORI      PONE EL BIT 7
1060        LDA (LPAG),Y EN (LPAG),Y
1070        CLC
1080        ROR
1090        ORA *BICO
1100        STA (LPAG),Y
1110        INY
1120        STY *FORI
1130        CPY *LIMI
1140        BNE LOP2
1150        RTS
1160  ;---------------------------------
1170  BIT7  LDA 080        EL BIT DE CONT
1180        AND BKGN       LO IGUALA AL
1190        STA *BICO      DEL FONDO
1200        RTS
1210  ;---------------------------------
1220  DIV7  LDY 08         ESTA RUTINA DIV
1230        SEC            ENTRE 7
1240        SBC 007        DEJANDO EL RES
1250  D1    PHP            EN 0 Y EL MOD
1260        ROL #0         EN EL ACUM
1270        ASL *LWAX
1280        ROL
1290        PLP
1300        BCC SUM
1310        SBC 007
1320        JMP D2
1330  SUM   ADC 007
1340  D2    DEY
1350        BNE D1
1360        BCS ULT
1370        ADC 007
1380        CLC
1390  ULT   ROL #0
1400        RTS
1410  ;=================================
1420  MUPA  LDA 000        MUEVE PALABRAS
1430        STA *LPAG      DE UN RENGLON
1440        STA *LPAD      A OTRO.COMEN-
```

```
1450        LDA 03F       ZANDO DE ABAJO
1460        STA *HPAG     HACIA ARRIBA Y
1470        STA *HPGD     DE DER A IZQ.
1480        LDA 0C0       INICIALIZA LOS
1490        STA *I192     RENG Y VAR PARA
1500        STA *FORI     LOS LOOPS
1510        SEC           EN LWAX PONE
1520        LDA 027       EL IND DE LA
1530        SBC *Q        IZQ EN RENGL
1540        STA *LWAX     ORG = 39 - Q;
1550        LDX 000
1560        LDY *AY       SALTA RENG QUE
1570  QUIT  BEQ TRAS      NO SE PUEDEN
1580        JSR REG       TRASLADAR;
1590        DEY           REG CAL NVO
1600        JMP QUIT      RENGL ORIGEN
1610  TRAS  JSR TREG      TRASLADA RENGL
1620        LDX 003
1630        JSR REG       CAL NVO RENG
1640        LDX 000
1650        JSR REG
1660        BNE TRAS      DESTINO Y ORIG
1670        LDA *AY       ==============
1680        BEQ RT        BORRA LA PARTE
1690  BORR  LDY 027       SUPERIOR DE LA
1700        LDA BKGN      PANTALLA. PONI
1710  BPAL  STA (LPGD),Y  ENDO EL COLOR
1720        DEY           DEL FONDO
1730        BPL BPAL
1740        LDX 003
1750        JSR REG
1760        BNE BORR
1770  RT    RTS
1780  ;--------------------------------
1790  TREG  LDX *LWAX     MARGEN DER REN
1800        LDY 027       ORG-X, REN DES
1810        STX *FORK     -Y
1820        STY *LIMI
1830  TPAL  LDY *FORK     TRASLADA LA
1840        LDA (LPAG),Y  FORK DEL RENGL
1850        LDY *LIMI     ORIG A LA PAL
1860        STA (LPGD),Y  LIMI DEL RENGL
1870        DEC *LIMI     DESTINO
1880        DEC *FORK
1890        BPL TPAL
1900        LDA *Q        ==============
1910        BEQ RT        BORRA LA PARTE
1920        LDA BKGN      IZQ DE LA PAN-
1930        LDY *LIMI     TALLA PONIENDO
1940  REY   STA (LPGD),Y  EL COLOR COMO
1950        DEY           EL FONDO
1960        BPL REY
1970        RTS
1980  ;================================
1990  REG   DEC *I192,X   CAL RENGL INME-
2000        BEQ RTRG      DIATO ANTERIOR;
2010        LDA 03F       SI X=0 PAG1
2020        AND *I192,X   SI X=3 PAG2
2030        BEQ M3
2040        LDA 007       FOR I=191 TO 1.-
1
2050        AND *I192,X   IF I MOD 64=0
2060        BEQ M4          LPAG=*+$1F58
2070        SEC           ELSE
2080        LDA *HPAG,X   IF I MOD 8 = 0
2090        SBC 004         LPAG=*+$1B80
2100        STA *HPAG,X   ELSE
2110        JMP RTRG        LPAG=*-$400
2120  M4    CLC
2130        LDA *LPAG,X
2140        ADC 080
2150        STA *LPAG,X
```

```
2160        LDA *HPAG,X
2170        ADC 01B
2180        STA *HPAG,X
2190        JMP RTRG
2200  M3    CLC
2210        LDA *LPAG,X
2220        ADC 058
2230        STA *LPAG,X
2240        LDA *HPAG,X
2250        ADC 01F
2260        STA *HPAG,X
2270  RTRG  RTS
2280  ;*********************************
2290        LDA 020       INICIALIZACION
2300        STA HPAG      HIRES, SIN BO-
2310        LDA $C057     RRAR LA FIGURA
2320        LDA $C053     CARGADA.
2330        LDA $C050
2340        LDA 000
2350        TAX
2360        TAY
2370        JSR $D02E
2380  FIN   RTS
2390        .EN
```







IV ESCALA

La escala nos sirve para amplificar o reducir alguna figura, esta amplificacion o reduccion puede ser realizada unicamente en alguno de los ejes, o puede ser que en un eje se aplique una amplificacion y en el restante una reduccion.

La escala es una transformacion geometrica que a cada punto de coordenadas cartesianas (X , Y) lo asigna a otro punto (X' , Y') donde X' es igual a X multiplicada por una constante y Y' es igual a Y multiplicada por otra constante, de aqui obtenemos las ecuaciones para la Escala.

$$X' = C1 \; X$$

Donde C1 , C2 son contantes

$$Y' = C2 \; Y$$

A C1 y C2 se les llama parametros de la escala y

dependiendo de los valores que se le asignen se realiza una amplificacion o una reduccion. Esto es, en caso de amplificar una figura el parametro tiene que ser mayor que uno, y para la reduccion el parametro esta entre cero y uno.

A continuacion se presentan algunas figuras para ejemplificar la Escala con diferentes parametros.

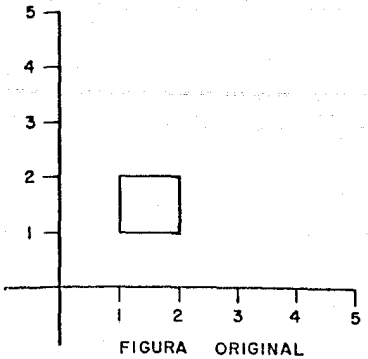
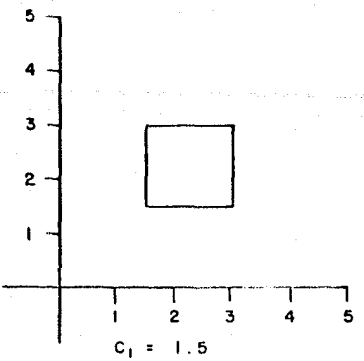
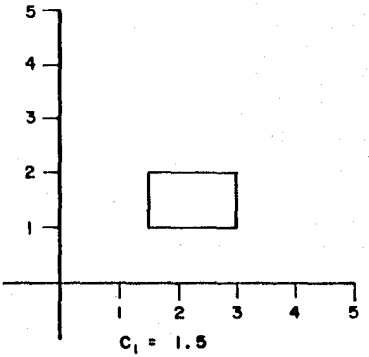


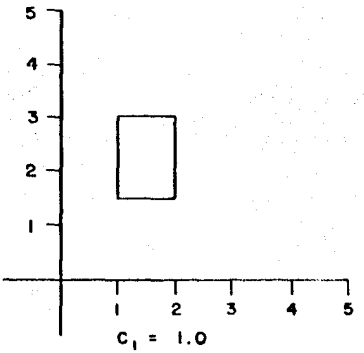
FIGURA ORIGINAL



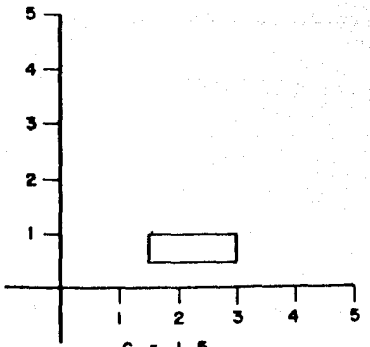
$C_1 = 1.5$
$C_2 = 1.5$



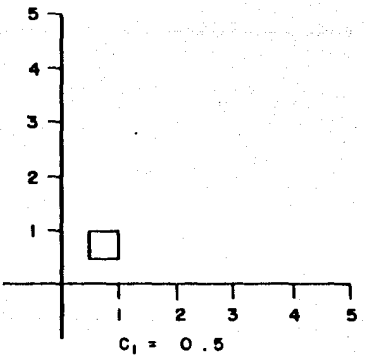
$C_1 = 1.5$
$C_2 = 1.0$



$C_1 = 1.0$
$C_2 = 1.5$



$C_1 = 1.5$
$C_2 = 0.5$



$C_1 = 0.5$
$C_2 = 0.5$

IV.1 ADAPTACION A LA APPLE II

Tomando en cuenta la resolucion que tiene la Apple que es aunque pequeña, no es posible asignar a los parametros valores reales en el caso de la amplificacion, debido a que no se puede evaluar mitades o cuartos de puntos, por este motivo los parametros para la amplificacion solamente asumen cantidades enteras.

Los valores que pueden tomar los parametros en el caso de la amplificacion son 1, 2, 3, 4, o 5 debido a que si tomamos el eje de las X's (el que posee mayor resolucion) y lo amplificamos en una proporcion de cinco, de la figura original utilizariamos unicamente 56 puntos, que considero es una cantidad minima.

En el caso de la reduccion y para facilitar la programacion, los valores que los parametros pueden asumir son -2, -3, -4, -5, significando realmente lo siguiente.

-2	0.5	reducir a la mitad
-3	0.3333	reducir a una tercera parte
-4	0.25	reducir a una cuarta parte
-5	0.20	reducir a una quinta parte

Ahora bien, para tener una continuidad en los valores, los numeros -1 0 y 1 se toman internamente como 1. Al cero se le elimina de su verdadero valor, debido a que si algun parametro es cero practicamente seria borrar la pantalla.

Comunmente uno de los procedimientos para amplificar una figura es tomar un punto y repetirlo tantas veces como lo indique el parametro. Este mecanismo es posible implementarlo en la Apple pero posee una desventaja dentro del eje de las X's, pues solamente se podria trabajar en blanco y negro, debido a que si un punto esta prendido y lo repetimos dos o tres veces el resultado, es una linea blanca.

Por este motivo se ideo un procedimiento alterno, el cual si permite trabajar con una gama de colores, este procedimiento consiste de lo siguiente:

Se toma un par de puntos para repetirlos tantas veces como lo especifique el parametro, teniendo cuidado de que un punto tomado de una posicion par, sea puesto a su vez en una posicion

par considerando la misma regla para los puntos nones, esto garantiza el que se pueda seguir utilizando colores y que estos no cambien por su complemento.

Para la amplificacion del eje Y el renglon se copia tantas veces como lo exprese el parametro.

Para hacer la reduccion en el eje de las X's primeramente se toman tantos puntos como indique el parametro y se analizan cuantos estan prendidos, no importando su posicion (color), si su numero sobrepasa a la mitad del valor absoluto del parametro, ese grupo de puntos correspondera a un punto prendido en la figura reducida, en caso de que el numero de puntos prendidos sea menor a la mitad del valor absoluto del parametro se genera un punto apagado.

Si el numero de puntos prendidos es igual, es necesario efectuar otro analisis que consiste en analizar la diferencia entre los puntos pares prendidos y los nones prendidos, si esta diferencia es mayor que cero y la posicion que ocupara el nuevo punto es par entonces se prendera, si es non se apagara. Si la diferencia es menor que cero y la posicion del nuevo punto es par se apagara, si es non se prendera y en caso de que la diferencia sea igual a cero el punto se prendera.

Para la reduccion en el eje de las Y's se aplican las siguientes formulas logicas:

PARAMETRO	FORMULA
-1	A
-2	A + B
-3	AB → AC + BC
-4	AB+ AC + AD →BC →BD →CD
-5	ABC + ABD →ABE →ACD →ACE + ADE + BCD → BCE + BDE + CDE

Donde A es el primer renglon reducido, B es el segundo, C el tercero, etc.

Debido a que la graficacion en la Apple no se basa en viewport, o sea que se tome solamente una parte del grafico para desplegarla en la reduccion, es necesario regenar la pantalla con espacios.

IV.2 PROGRAMACION

En la programacion de la escala fue necesario tener especial cuidado en el tamaño del programa, debido a que en realidad son dos programas (amplificacion y reduccion), que no se podian realizar por separado debido a que era posible llegar a combinarlos en su utilizacion.

Por este motivo, el programa fue diseñado para que ambas opciones compartieran el mayor numero de instrucciones, esto fue posible en mayor medida en las rutinas de lectura de datos y de actualizacion de apuntadores, tal es el caso de la subrutina PAI cuya funcion es obtener el punto inicial, que en el caso de la amplificacion sobre X indica el primer punto al que se puede aplicar la transformacion y en la reduccion es el primer punto que va a tener informacion. Esto es, en el primer caso tiene utilidad en la primera pagina y para la reduccion en la segunda. Esto mismo ocurre en la rutina ACTR que actualiza el apuntador del renglon de la pagina uno o dos dependiendo si es amplificacion o reduccion en Y, pero con un detalle adicional que consiste en pasar por medio del registro X un valor que indica sobre que pagina ha de realizarse la actualizacion, utilizando para ello la rutina comentada en la Traslacion.

En la rutina ACTP el numero de renglones que se recorren de abajo hacia arriba varia dependiendo del parametro segun lo indica la siguiente tabla:

PARAMETRO	RENGLONES DE MENOS
0,1	0
2	48
3	64
4	72
5	76

Analizando al numero de renglones de menos, notamos que entre 64 y 48 existe una diferencia de 16, entre 72 y 64 hay 8 y entre 76 y 72 hay 4; ahora bien, las diferencias 16, 8 y 4 tienen la relacion de ser la mitad de la diferencia anterior. Esto forma la base de la rutina para obtener el numero de renglones de menos, puesto que al acumulador se le asigna un valor inicial de 48 y a otra variable de 16. Si el parametro es dos el calculo termina y en el acumulador tiene el valor de 48, en caso contrario se suma la variable al acumulador y se hace un ROR a la variable que es como si se dividiera entre dos, posteriormente se analiza el valor del parametro para ver si ya se finaliza el calculo. En la amplificacion del eje X si el parametro es igual a uno se realiza una copia del renglon de la pagina uno a la pagina dos, en caso contrario se realiza propiamente la amplificacion que consiste en tomar un par de bits del renglon origen, esto se hace por medio de un ROR en el renglon y un ROL a una variable, repitiendo esta secuencia una

segunda vez. Ahora bien, en virtud de que el par de bits se obtienen por medio de un ROR y un ROL el valor de ese par de bits se invertiria en algunos casos segun se muestra a continuacion.

CUENTA DE BITS ORIGINAL	CUENTA DE BITS EN LA VARIABLE
00	00
01	10
10	01
11	11

Despues de haber obtenido ese par de bits y utilizando un contador de campos disponibles se procede a limpiar en la palabra destino, los campos disponibles, procedimiento que se hace de la siguiente forma, al acumulador se le prenden todos sus bits y se hace un OR exclusivo con un valor determinado, el cual depende del valor que posea el contador de campos disponibles segun lo muestra la siguiente tabla:

CAMPOS DISPONIBLES	VALOR EN HEXADECIMAL
0	00
1	C0
2	F0
3	FC
4	FF

Este procedimiento en realidad obtiene el complemento a uno de los valores expuestos anteriormente y con esto se hace un AND con la palabra a la cual se le va a depositar la información, dejando con ello limpios los campos disponibles.

Hay que hacer nota, que se esta empacando la información en las palabras, esto es, no se esta considerando el bit de control y en su lugar se encuentre otro punto graficable.

Despues de haber limpiado los campos disponibles, se procede a poner los nuevos puntos, que como se había planteado anteriormente, consiste en repetir el par de puntos tantas veces como valor tenga el parametro, para esto el acumulador es cargado con la repeticion de ese par de puntos, utilizando para ello a la variable en la cual se depositaran los dos bits.

Valor de la variable	Valor a cargar (Hex.)	Valor a cargar (Bin.)
0	00	0000.0000
1	AA	1010.1010
2	55	0101.0101
3	FF	1111.1111

Hay que recordar que cuando la variable posee un valor de 1 el par de bits originales tenían un valor de 2, es por eso que se tuvo que invertir los valores a cargar al acumulador.

Ya teniendo el simulador cargado, se procede a hacer un AND con alguno de los valores expuestos en la tabla de campos disponibles y finalmente un OR con la palabra destino, con objeto de conservar la información que previamente haya tenido almacenada.

Finalmente se ajusta la variable de campos disponibles y se continua con estas funciones hasta el final del renglon.

Para la rutina de reduccion en Y es necesario aplicar las formulas logicas expuestas anteriormente, las cuales a primera vista son diferentes entre si, por lo cual se tendria que hacer una rutina para cada parametro, solucion que es totalmente ineficiente. Haciendo uso del diseño logico se logro establecer una similitud que se podia ir desarrollando conforme se realizaban las escalas en el eje X.

PARAMETRO FORMULA

- 2 A + B
- 3 AB + AC + BC = AB + (A+B) C
- 4 AB + AC + AD + BC + BD + CD
  AB + (A+B) C + (A+B+C) D
- 5 ABC+ABD+ABE+ACD+ACE+ADE+
  BCD+BCE+BDE+CDE=
  ABC+ABD+(AB+B)+AEE+(A+B)CE+
  (A+B+C)DE =
  ABC+(AB+(A+B)C)D+(AE+(A+B)C+(A+B+C)D)E

Para el desarrollo de estas formulas se requeria una sola rutina debido a que existen factores comunes, asi mismo, unicamente se necesitaban tres registros auxiliares para su desarrollo, mismo que a continuacion se describe.

Existe una variable que lleva el numero de renglones involucrados en la reduccion sobre el eje y, cuando esta variable es igual al valor absoluto del parametro, la reduccion para un grupo de registros se ha finalizado y el resultado se encuentra en el tercer registro auxiliar.

Ahora bien, cuando la variable posee un valor de cero, el registro ya transformado sobre X, es copiado a los registros de trabajo 1, 2 y 3, despues de esto la variable es incrementada en uno y se realizan las comparaciones expuestas anteriormente. Si la variable tiene un valor de uno se hace un OR del nuevo registro con el registro 1 y el resultado se deposita en los registros 1 y 3, en esta misma etapa se hace un AND del registro y REG2 (Registro 2) dejandolo en REG2, finalmente los registros quedan como se muestra a continuacion:

REG1	REG2	REG3
A+B	AB	A+B

Hay que observar que REG3 tiene el resultado cuando el parametro es =2

Cuando la variable es igual a dos, el nuevo registro se relaciona con REG3 por medio de un AND y despues se hace un OR con REG2 almacenando el resultado en REG3, posteriormente se hace un OR del registro con REG1 dejando el resultado en el mismo REG1 y un AND del registro y REG2 poniendo la informacion en REG2, con esto los registros quedan de la forma siguiente:

REG1	REG2	REG3
A+B+C	ABC	AB+(A+B)C

En caso de que la variable llegara a tener un valor de tres se hace un AND de REG3 y el registro seguido de un OR con REG2 dejando la informacion en REG2, despues se hace un AND del registro y REG1, a ese resultado se le aplica un OR con REG3, para dejar los registros como se ilustra a continuacion:

REG1	REG2	REG3
A+B+C	ABC+[AB+(A+B)C]D	AB+(A+B)C+(A+B+C)D

Por último cuando la variable llega a adoptar el valor de cuatro se hace un AND del nuevo registro y REG3 seguido de un OR con el contenido de REG2 y eso se deposita en REG3, dejando los registros como a continuación se indica

REG1	REG2	REG3
A+B+C	ABC+[AB+(A+B)C]D	ABC+[AB+(A+B)C]D
		+[AB+(A+B)C+(A+B+C)D]E

PROGRAMA Y EJEMPLOS DE LA TRASLACION

Dentro de este anexo se encuentra el listado de la rutina de Escala en ensamblador de la 6502. El codigo de esta rutina es puesto a partir de la localidad 800 Hex.

Asi mismo, se incluyen una serie de graficas, la primera de ellas es el original, las siguientes son ejemplos de escalas.

La segunda grafica es el resultado de aplicar los parametros (2,2), ahora bien, tomando a esa figura como original y aplicandole los parametros (-2, -2) obtenemos el tercer grafico, que muestra solamente una parte del original debido a que, como se habia hecho notar, la Apple no trabaja a base de viewports.

Las dos siguientes graficas son el resultado de aplicar a la figura original los parametros (3,0) y (0,3), respectivamente.

Por ultimo se tiene una pequeña figura que se obtuvo al

aplicar a la figura original primero un parametro (5,5) y posteriormente (-5,-5); esto fue hecho con el objeto de mostrar que realmente la informacion utilizada para aplicar una escala de (5,5) o (-5,-5) es minima.

Se puede ver que existen algunas graficas que tienen doble linea, esto es debido a que para conservar el color se tiene que prender puntos en las posiciones pares o en las nones unicamente, con lo que en los monitores en blanco y negro se vera una doble linea como es el caso de las presentes figuras.

A continuacion se ofrece una relacion de tiempos de ejecucion para diferentes parametros.

PARAMETRO	TIEMPO DE EJECUCION
(0,2)	1.0 seg
(2,0)	7.6 seg
(2,2)	4.3 seg
(0,5)	1.0 seg
(5,0)	6.8 seg
(5,5)	2.2 seg
(0,-2)	1.4 seg
(-2,0)	9.5 seg
(-2,-2)	9.6 seg
(0,-5)	1.4 seg

(-5,0)	4.0 seg
(-5,-5)	4.4 seg

Como se puede ver existe una gran disparidad en los tiempos, esto es debido a que conforme es mayor el parametro se reduce la porcion de la figura original que hay que procesar, esto no pasa cuando unicamente se utiliza el parametro de Y pues el proceso de amplificacion se realiza mediante una copia de renglones que no tiene ningun proceso adicional, es por eso que los tiempos para los parametros (0,2) y (0,5) son iguales.

IV.3 PROGRAMA Y EJEMPLOS DE ESCALA

```
>REM PROGRAMA QUE LLAMA A LA RUTINA DE
     ESCALA
>LIST
   10 POKE 74,0: POKE 204,0: REM  LOWM
      EM $1000
   20 POKE 75,16: POKE 205,16
   25 X0=Y0
   30 DIM A$(8)
   35 POKE 803,0: REM  COLOR DEL FONDO

   37 PRINT "BLOAD REDC"
   40 INPUT "DESEAS CARGAR ALGUNA FIGU
      RA? ",A$: IF A$="NO" THEN GOTO
      100
   50 PRINT "BLOAD ";A$
   60 CALL 2953
  100 INPUT "PARAMETROS PARA LA ESCALA
       <X,Y> ",X0,Y0
  110 CALL 2048
  120 END
```

```
>REM PROGRAMA QUE LLAMA A LA RUTINA DE
     ESCALA
>LIST
   10 POKE 74,0: POKE 204,0: REM  LOWM
      EM $1000
   20 POKE 75,16: POKE 205,16
   25 X0=Y0
   30 DIM A$(8)
   35 POKE 803,0: REM  COLOR DEL FONDO

   37 PRINT "BLOAD REDC"
   40 INPUT "DESEAS CARGAR ALGUNA FIGU
      RA? ",A$: IF A$="NO" THEN GOTO
      100
   50 PRINT "BLOAD ";A$
   60 CALL 2953
  100 INPUT "PARAMETROS PARA LA ESCALA
       <X,Y> ",X0,Y0
  110 CALL 2048
  120 END
```

```
0010 LWAX .DL 0006     PARAM DE ESCALA
0020 RENI .DL 0007     RENGL DE MENOS
0030 CDIS  DL 0007     CAMPOS DISPONIB
0040 AROL .DL 0007     CONTADOR PAL
0050 NOR  .DL 00#1     # RENG REDUC Y
0060 AY    DL 0008     PARAM DE ESCALA
0070 PINI .DL 0009     PALAB INICIAL
0080 BINI .DL 000A     BIT INICIAL
0090 CPYL  DL 000B     APUNT RENG PAG
0100 CPYH .DL 000C     ESC EN Y;
0110 BI   .DL 000B     TRANSITORIA
0120 PI   .DL 000C     TRANSITORIA
0130 BPAR .DL 000D     NO BITS POS PAR
0140 BNON .DL 000E     NO BITS POS NON
0150 POSI .DL 000F     POS INICIAL
0160 POSF .DL 0010     POS FINAL
0170 BITS .DL 0010     2 BITS A ESCALA
0180 I192 .DL 0019     CONT RENG HIRES
0190 LPAG .DL 001A     APUNT RENG PAG1
0200 HPAG .DL 001B
0210 FORI .DL 001C     CONT RENG HIRES
0220 LPGD .DL 001D     APUNT RENG PAG2
0230 HPGD .DL 001E
0240 CRES .DL 001F     CAMP RESTANTES
0250 LMEM .DL 004A     INIC VAR BASIC
0260 BKGN .DL 0323     COLOR FONDO
0270 NPAG .DL 0326     # PAG BORRAR
0280 REG1 .DL 57D0     RENG 1 ESCALA Y
0290 REG2 .DL 5BD0     RENG 2 ESCALA Y
0300 REG3 .DL 5FD0     RENG 3 ESCALA Y
0310 BPG2 .DL D00E     BORRA PAG 2
0320 GETX .DL D39D     RUTINA OBT X
0330 GETY .DL D3AF     RUTINA OBT Y
0340 RERR .DL EE68     RUT RANGE ERR
0350 :*********************************
0360 :* RUTINA DE ESCALA EN HIRES   **
0370 :*********************************
0380       LDA 040     BORRA PAG 2
0390       STA NPAG
0400       JSR BPG2
0410       LDA 020
0420       STA NPAG
0430       JSR OBTP    OBT PARAM ESCAL
0440       JSR PALI    CAL PAL INICIAL
0450       LDA *AY     VE SI AMPLIA O
0460       BMI NEG     REDUCE PARA
0470       LDX 000
0480       JMP ACT
0490 NEG   LDX 003
0500 ACT   JSR ACTR    ACT APUNT RENG
0510 CONT  LDA *LWAX   SI LWAX>=0
0520       BMI C2
0530       JSR ESCX      AMPLIA X
0540       BNE C3
0550       BEQ C5
0560 C2    JSR REDX    SINO REDUCE X
0570       BEQ C5
0580 C3    LDA *AY     SI AY<0
0590       BPL C4
0600       JSR REDY      REDUCE Y
0610       BNE CONT
0620 C4    JSR ESCY    SINO AMPLIA Y
0630       BNE CONT
0640 C5    JSR P2-1    PASA PAG 2 A 1
0650       RTS
0660 :*********************************
0670 OBTP  LDY 005      OBT PARAM ESCA
0680       JSR LEE     LA PARA X,Y
0690       STA *LWAX   PARAM X => LWAX
0700       LDY 00C
0710       JSR LEE     PARAM Y => AY
0720       STA *AY     APUNT AL FINAL
0730       LDA 0D0     DE LOS RENGL DE
0740       STA *LPAG   LAS PAG 1 Y 2
0750       STA *LPGD   DE HIRES
0760       LDA 03F     PAG1=HPAG,LPAG
0770       STA *HPAG   PAG2=HPGD,LPGD
0780       LDA 05F
0790       STA *HPGD
0800       LDA 0C0
0810       STA *I192
0820       STA *FORI
```

```
0830        LDA 000
0840        STA *NOR
0850  RETS  RTS
0860  ;================================
0870  LEE   LDA (LMEM),Y OBT PARAM A PAR
0880        CMP 006       TIR VARIABLES
0890        BMI LE1       INT-BASIC; DA
0900        JMP RERR      ERROR SI NO
0910  LE1   CMP 0FB       ESTAN EN EL
0920        BPL LE2       RANGO 5,-5
0930        JMP RERR
0940  LE2   CMP 0FF
0950        BEQ LE3
0960        CMP 001
0970        BNE LE4
0980  LE3   LDA 000
0990  LE4   RTS
1000  ;********************************
1010  PALI  LDA *LWAX     SACA EL VALOR
1020        BPL XABS      ABSOLUTO DEL
1030        EOR 0FF       PARAM X
1040        CLC
1050        ADC 001
1060  XABS  TAX
1070        BNE X2-5      OBT PAL Y BIT
1080        LDY 000       INICIAL EN RENG
1090        LDX 007       ESC   PAL   BIT
1100        JMP XRES      0,1    0     7
1110  X2-5  CPX 003        2    10     7
1120        BMI X2         3    13     4
1130        BEQ X3         4    15     6
1140        CPX 004        5    16     7
1150        BEQ X4
1160        LDY 010
1170        LDX 007
1180        JMP XRES      PAL INI=PINI
1190  X2    LDY 00A       PAL INI=PINI
1200        LDX 007
1210        JMP XRES
1220  X3    LDY 00D
1230        LDX 004
1240        JMP XRES
1250  X4    LDY 00F
1260        LDX 006
1270  XRES  STY *PINI
1280        STX *BINI
1290        RTS
1300  ;********************************
1310  ACTR  LDA *AY       SACA EL VALOR
1320        BPL YABS      ABSOLUTO DE AY
1330        EOR 0FF
1340        CLC
1350        ADC 001       CALCULA CUANTOS
1360  YABS  TAY           RENG DE MENOS
1370        BEQ CON1      POR ESCALA EN Y
1380        LDA 010       ESC   -RENG
1390        STA *RENI     0,1    0
1400        LDA 030       2     48
1410        DEY           3     64
1420  L1    DEY           4     72
1430        BEQ CON1      5     76
1440        CLC
1450        ADC *RENI
1460        ROR *RENI
1470        JMP L1
1480  CON1  STA *RENI
1490  L2    DEC *RENI     ACT RENGL DEPEN
1500        BMI RETS      DIENDO DE X
1510        BEQ RETS
1520        JSR REG
1530        JMP L2
1540  ;********************************
1550  ESCX  JSR INI0      HACE LA ESCALA
1560        BEQ COPY      SOBRE EL EJE X
1570        LDA (LPAG),Y
1580        JSR INI1
1590        STA (LPAG),Y
1600        JSR ESCR      ESCALA RENGL
1610        JSR ROTA
1620        LDX 000
1630        JSR REG
1640        RTS           ACT LOS APUNT
```

```
1650  ;================================
1660  INI0 LDY *PINI     INIC APUNT PAL
1670       BEQ RET1      BITS INICIALES
1680       STY *PI
1690       LDX *BINI
1700       STX *BI
1710       RTS
1720  ;================================
1730  INI1 CPX 007       SI BINI<7 HACE
1740       BEQ RET1      ROR PARA ACT
1750       ROR           APUNT BITS EN
1760       INX           PAL INICIAL
1770       JMP INI1
1780  ;================================
1790  COPY LDY 027       COPIA EL RENG
1800  C1   LDA (LPAG),Y  ORIGEN AL DEST
1810       STA (LPGD),Y
1820       DEY
1830       BPL C1
1840       LDX 000
1850       JSR REG
1860  RET1 RTS
1870  ;================================
1880  ROTA LDY 000       POR TENER EMPA-
1890       STY *AROL     CADA LA INF EN
1900  R0   LDA (LPGD),Y  LOS 8 BITS, SE
1910       ROL           HACE ROL PARA
1920       PHP           OBT BIT-8 Y ROR
1930       CLC           PARA PONER CONT
1940       ROR
1950       STA (LPGD),Y
1960       INY
1970  R1   PLP
1980       LDA (LPGD),Y
1990       ROL
2000       STA (LPGD),Y
2010       PHP
2020       INY
2030       CPY 028
2040       BMI R1
2050       PLP
2060       INC *AROL
2070       LDY *AROL
2080       CPY 028
2090       BMI R0
2100       RTS
2110  ;================================
2120  PARB LDA 000       OBTIENE UN PAR
2130       STA *BITS     DE BITS DEL
2140       LDA *LWAX
2150       STA *CRES
2160       TYA           RENG ORIGEN Y
2170       PHA           LOS PONE EN
2180       CLC
2190       LDY *PI
2200       JSR UNB       BITS:PI=PAL INI
2210       JSR UNB
2220       PLA
2230       TAY
2240       RTS           BI=BITS REST
2250  ;--------------------------------
2260  UNB  LDA (LPAG),Y  AL HACER ROR Y
2270       ROR           ROL EL 2 => 1
2280       ROL *BITS     Y EL 1 => 2
2290       STA (LPAG),Y
2300       DEC *BI
2310       BNE RET2
2320       INC *PI
2330       INY
2340       LDA 007
2350       STA *BI
2360  RET2 RTS
2370  ;================================
2380  ESCR LDY 000       CDIS=4:Y=0
2390       LDA 004       CRES=LWAX
2400  L4   STA *CDIS     SI CRES=0 OBT
2410       JSR PARB      UN PAR DE BITS
2420  L5   LDX *CDIS     SI CDIS=0 INCRE-
2430       BNE L6        MENTA EN 1 EL
2440       INY           APUNT DE LA PAL
2450       CPY 023       EN RENG DESTINO
2460       BEQ RET2      (Y) Y VE SI YA
```

```
2470        LDX 004       TERMINO (Y=35)
2480        STX *CDIS     CDIS=4;
2490  L6    LDA 0FF       PONE A CEROS -
2500        EOR TDIS,X    LOS CAMPOS DISP
2510        AND (LPGD),Y
2520        STA (LPGD),Y
2530        LDX *BITS     CODIGO COLOR
2540        LDA MASC,X    AND CDIS OR PAL
2550        LDX *CDIS
2560        AND TDIS,X
2570        ORA (LPGD),Y
2580        STA (LPGD),Y
2590        LDA *CDIS      AC=CDIS-CRES
2600        SEC            SI AC=0 CDIS=0
2610        SBC *CRES         CRES=0
2620        BPL L4         SI AC>0 CRES=0
2630        EOR 0FF           CDIS=AC
2640        ADC 001        SI AC<0 CDIS=0
2650        STA *CRES         CRES= -AC
2660        LDA 000
2670        STA *CDIS
2680        JMP L5
2690  :TABLA DE BITS DISPONIBLES SEGUN
2700  :VALOR DE CDIS 4=TODOS,3=BITS 2-7
2710  :2=BITS 4-7, 1=BITS 6-7,0=NINGUNO
2720  TDIS .HS 00C0F0FCFF
2730  :*********************************
2740  :MASCARAS DE BITS SEGUN EL VALOR
2750  :DEL PAR DE BITS QUE ESTA EN BITS
2760  MASC .HS 00AA55FF
2770  :*********************************
2780  REDX JSR INI0       REDUCCION EJE
2790        LDA (LPGD),Y  X'S
2800        JSR INI1
2810        STA (LPGD),Y
2820        JSR REDR      REDUCE RENG
2830        LDX 000       ACT APUNT PAG1
2840        JSR REG
2850        RTS
2860  :================================
2870  REDR LDY 007       REDUCCION RENG
2880        STY *CDIS     X'S
2890        LDY 000
2900        STY *POSI
2910        STY *POSF
2920        LDA *LWAX
2930        EOR 0FF       X=-LWAX
2940        CLC
2950        ADC 001
2960        TAX           INIC VAR
2970  INIV  STX *CRES     PARA CONT BITS
2980        LDA 000       PRENDIDOS EN
2990        STA *BNON     POS PARES O
3000        STA *BPAR     NONES
3010  NOBT  LDA (LPAG),Y
3020        ROR
3030        STA (LPAG),Y
3040        BCC ACTA
3050        LDA 001
3060        AND *POSI
3070        BEQ PAR
3080        INC *BNON
3090        JMP ACTA
3100  PAR   INC *BPAR
3110  ACTA  INC *POSI
3120        DEC *CDIS     ACTUALIZA APUN
3130        BNE REVR       TADORES RENG
3140        INY            ORIGEN=PAG1
3150        CPY 028
3160        BNE A1
3170        JSR ANAL      PRENDE O APAGA
3180        LDY *PI       ULT GRUPO BITS
3190        LDA (LPGD),Y Y AJUSTA LA PAL
3200  A0    DEC *BI
3210        BMI A2
3220        CLC
3230        ROR
3240        JMP A0
3250  A2    ROR
3260        STA (LPGD),Y
3270        RTS
3280  A1    LDA 007
```

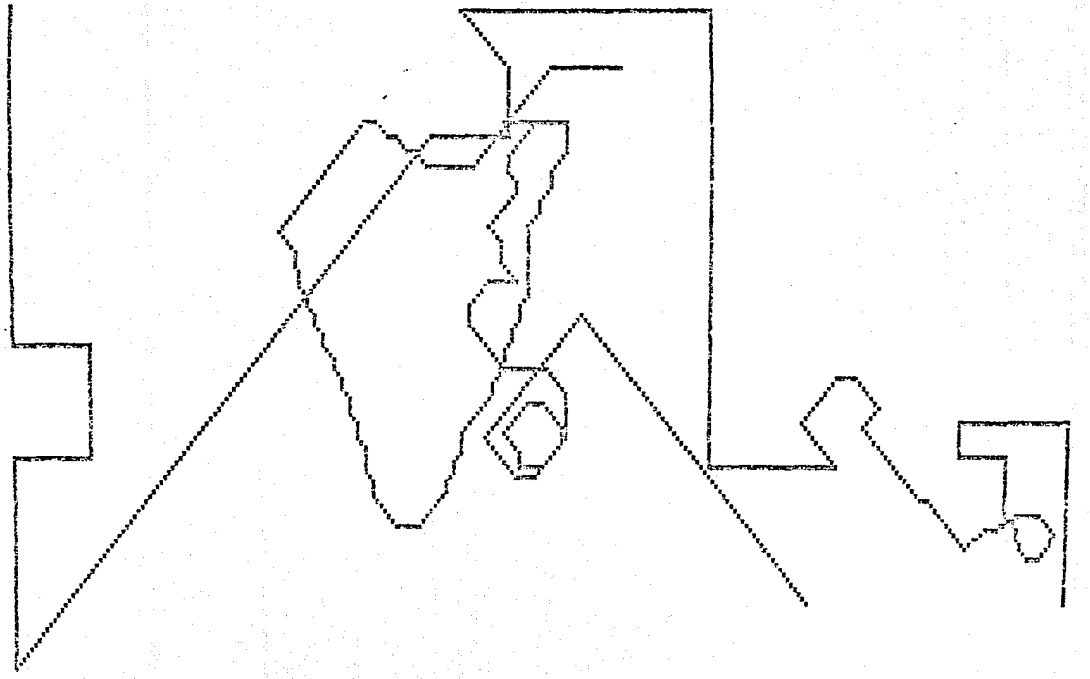
```
3290        STA *CDIS
3300  REVR  DEC *CRES      VE SI YA CONTO
3310        BNE NOBT      EL # DE BITS
3320        JSR ANAL      DE LA ESCALA
3330        JMP INIV      PARA PRENDERLO
3340  :=================================
3350  ANAL  TXA           ANALIZA SI SE
3360        STA *CRES     PRENDE O SE
3370        CLC           APAGA EL BIT
3380        ROR           EN RENG DEST
3390        SEC           PAG2
3400        SBC *BPAR     SI LWAX/2-BPAR
3410        SEC              -BNON > 0
3420        SBC *BNON        APAGA
3430        BMI PREN         < 0 PRENDE
3440        BEQ PARA
3450  APAG  CLC
3460        JMP PRED
3470  PREN  SEC
3480        JMP PRED
3490  PARA  ROR *CRES     SI LWAX=NON
3500        BCS APAG        APAGA
3510        SEC
3520        LDA *BPAR     SI BPAR-BNON=0
3530        SBC *BNON        PRENDE
3540        BEQ PREN
3550        BMI <NON
3560        LDA *POSF     SI BPAR>BNON
3570        ROR             SI POSF=PAR
3580        BCC PREN           PRENDE
3590        BCS APAG        SINO APAGA
3600  <NON  LDA *POSF     SI BPAR<BNON
3610        ROR             SI POSF=PAR
3620        BCC APAG           APAGA
3630        BCS PREN        SINO PRENDE
3640  :---------------------------------
3650  PRED  TYA           CARRY=1 PRENDE
3660        PHA           CARRY=0 APAGA
3670        LDY *PI       PALABRA RENG
3680        LDA (LPGD),Y  DESTINO(PAG2)
3690        ROR
```

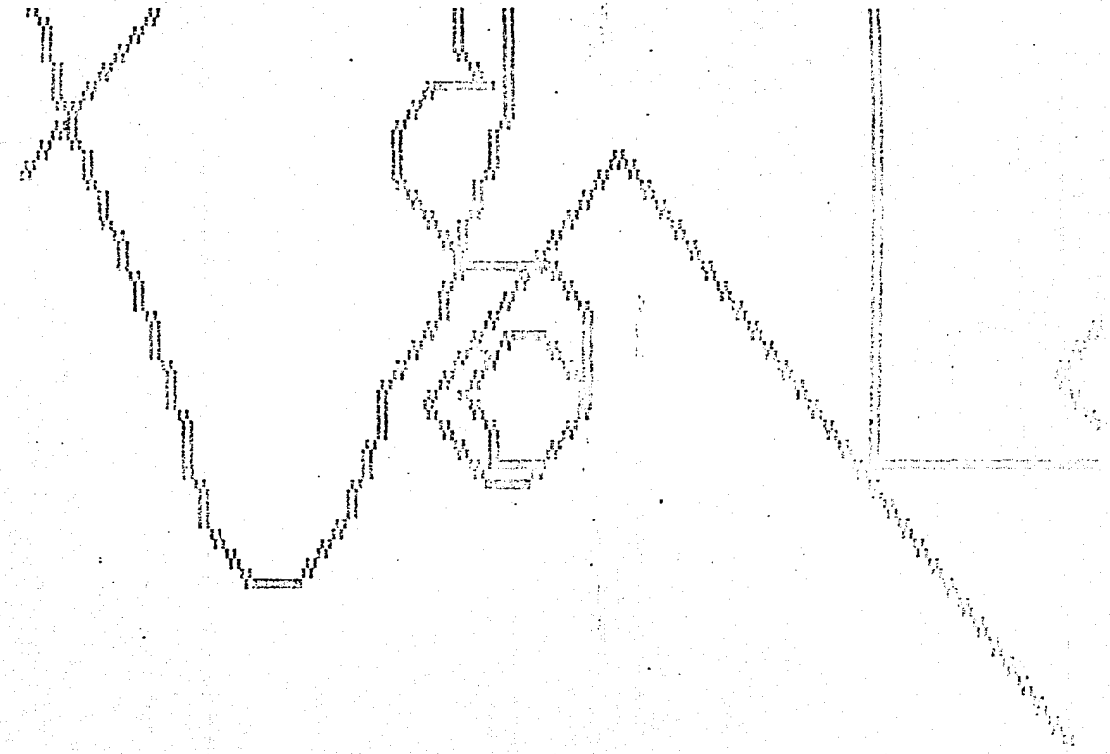
```
3700        STA (LPGD),Y
3710        INC *POSF
3720        DEC *BI
3730        BNE RTS1
3740        CLC
3750        ROR              PONE BIT-8=0
3760        STA (LPGD),Y
3770        INC *PI
3780        LDA 007
3790        STA *BI
3800  RTS1  PLA
3810        TAY
3820        RTS
3830  :*********************************
3840  ESCY  LDA *AY         HACE LA ESCALA
3850        BEQ E3          PARA Y COPIAN-
3860        STA *RENI
3870        LDA *LPGD       DO EL ULT RENG
3880        STA *CPYL       DEST
3890        LDA *HPGD
3900        STA *CPYH
3910  E1    DEC *RENI
3920        BEQ E3
3930        LDX 003
3940        JSR REG
3950        BEQ E4
3960        LDY 027
3970  E2    LDA (CPYL),Y
3980        STA (LPGD),Y
3990        DEY
4000        BPL E2
4010        BMI E1
4020  E3    LDX 003
4030        JSR REG
4040  E4    RTS
4050  :*********************************
4060  REDY  LDY 027
4070        LDA *NOR      REDUCCION EJE
4080        BEQ MRG0      Y'S
4090        CMP 002       DEPENDIENDO DEL
4100        BMI MRG1      VALOR DE NOR
```

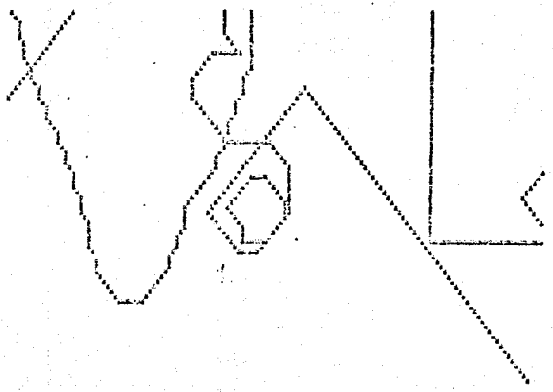
```
4110        BEQ MRG2       ES LA ETAPA DE
4120        CMP 004        LA REDUCCION
4130        BMI MRG3
4140        BEQ MRG4
4150  MRG0  LDA (LPGD),Y **************
4160        STA REG1,Y
4170        STA REG2,Y
4180        STA REG3,Y
4190        DEY
4200        BPL MRG0
4210        JMP SIG
4220  MRG1  JSR MR1        **************
4230        STA REG3,Y
4240        JSR MR2
4250        DEY
4260        BPL MRG1
4270        JMP SIG
4280  MRG2  JSR MR3        **************
4290        STA REG3,Y
4300        JSR MR1
4310        JSR MR2
4320        DEY
4330        BPL MRG2
4340        JMP SIG
4350  MRG3  JSR MR3        **************
4360        STA REG2,Y
4370        LDA (LPGD),Y
4380        AND REG1,Y
4390        STA REG1,Y
4400        ORA REG3,Y
4410        STA REG3,Y
4420        DEY
4430        BPL MRG3
4440        JMP SIG
4450  MRG4  JSR MR3        **************
4460        STA REG3,Y
4470        DEY
4480        BPL MRG4
4490  SIG   INC *NOR       **************
4500        LDA *AY        SI NOR=-AY COP
4510        CLC            REG3 QUE TIENE
4520        ADC *NOR       EL RESULT A PG2
4530        BNE RTS2
4540        LDY 027
4550  S0    LDA REG3,Y
4560        STA (LPGD),Y
4570        DEY
4580        BPL S0
4590        LDA 000
4600        STA *NOR
4610        LDX 003        ACT APUNT PAG2
4620        JSR PEG
4630  RTS2  RTS
4640  ;=================================
4650  MR1   LDA (LPGD),Y
4660        ORA REG1,Y
4670        STA REG1,Y
4680        RTS
4690  ;=================================
4700  MR2   LDA (LPGD),Y
4710        AND REG2,Y
4720        STA REG2,Y
4730        RTS
4740  ;=================================
4750  MR3   LDA (LPGD),Y
4760        AND REG3,Y
4770        ORA REG2,Y
4780        RTS
4790  ***********************************
4800  P2-1  LDA 000        PASA HIRES DE
4810        STA *LPAG      LA PAG 2 A LA 1
4820        STA *LPGD
4830        LDA 02F
4840        STA *HPAG
4850        LDA 05F
4860        STA *HPGD
4870        LDX 01F
4880        LDY 000
4890  P1    LDA (LPGD),Y
4900        STA (LPAG),Y
4910        DEY
4920        BNE P1
```

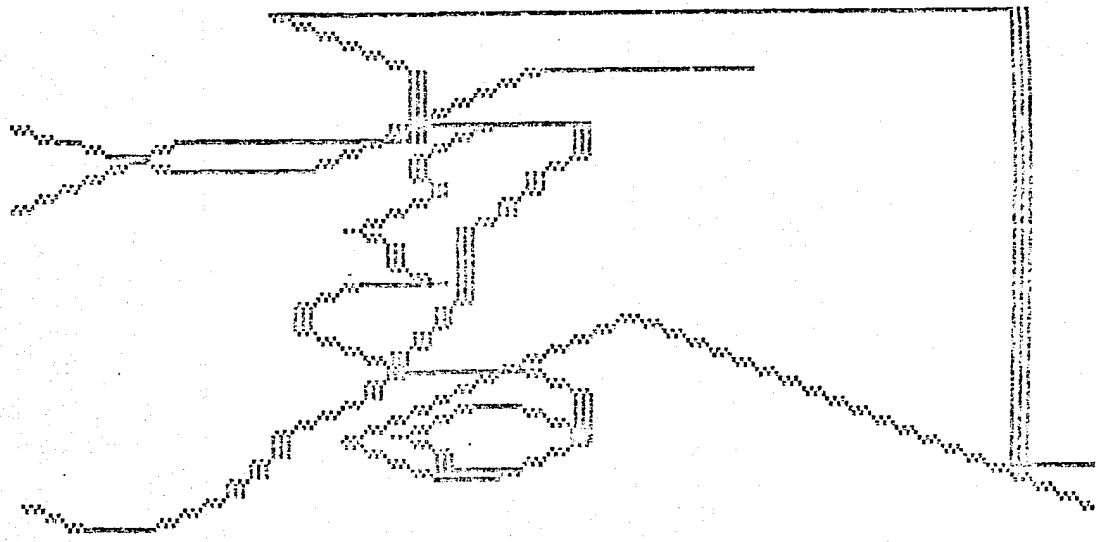
```
4110        BEQ MRG2        ES LA ETAPA DE
4120        CMP 004         LA REDUCCION
4130        BMI MRG3
4140        BEQ MRG4
4150  MRG0  LDA (LPGD),Y ***************
4160        STA REG1,Y
4170        STA REG2,Y
4180        STA REG3,Y
4190        DEY
4200        BPL MRG0
4210        JMP SIG
4220  MRG1  JSR MR1         ***************
4230        STA REG3,Y
4240        JSR MR2
4250        DEY
4260        BPL MRG1
4270        JMP SIG
4280  MRG2  JSR MR3         ***************
4290        STA REG3,Y
4300        JSR MR1
4310        JSR MR2
4320        DEY
4330        BPL MRG2
4340        JMP SIG
4350  MRG3  JSR MR3         ***************
4360        STA REG2,Y
4370        LDA (LPGD),Y
4380        AND REG1,Y
4390        STA REG1,Y
4400        ORA REG3,Y
4410        STA REG3,Y
4420        DEY
4430        BPL MRG3
4440        JMP SIG
4450  MRG4  JSR MR3         ***************
4460        STA REG3,Y
4470        DEY
4480        BPL MRG4
4490  SIG   INC *NOR        ***************
4500        LDA *AY         SI NOR=-AY COP
4510        CLC             REG3 QUE TIENE
4520        ADC *NOR        EL RESULT A PG2
4530        BNE RTS2
4540        LDY 027
4550  S0    LDA REG3,Y
4560        STA (LPGD),Y
4570        DEY
4580        BPL S0
4590        LDA 000
4600        STA *NOR
4610        LDX 003         ACT APUNT PAG2
4620        JSR REG
4630  RTS2  RTS
4640  ;=================================
4650  MR1   LDA (LPGD),Y
4660        ORA REG1,Y
4670        STA REG1,Y
4680        RTS
4690  ;=================================
4700  MR2   LDA (LPGD),Y
4710        AND REG2,Y
4720        STA REG2,Y
4730        RTS
4740  ;=================================
4750  MR3   LDA (LPGD),Y
4760        AND REG3,Y
4770        ORA REG2,Y
4780        RTS
4790  ;*********************************
4800  P2-1  LDA 000         PASA HIRES DE
4810        STA *LPAG       LA PAG 2 A LA 1
4820        STA *LPGD
4830        LDA 03F
4840        STA *HPAG
4850        LDA 05F
4860        STA *HPGD
4870        LDX 01F
4880        LDY 000
4890  P1    LDA (LPGD),Y
4900        STA (LPAG),Y
4910        DEY
4920        BNE P1
```

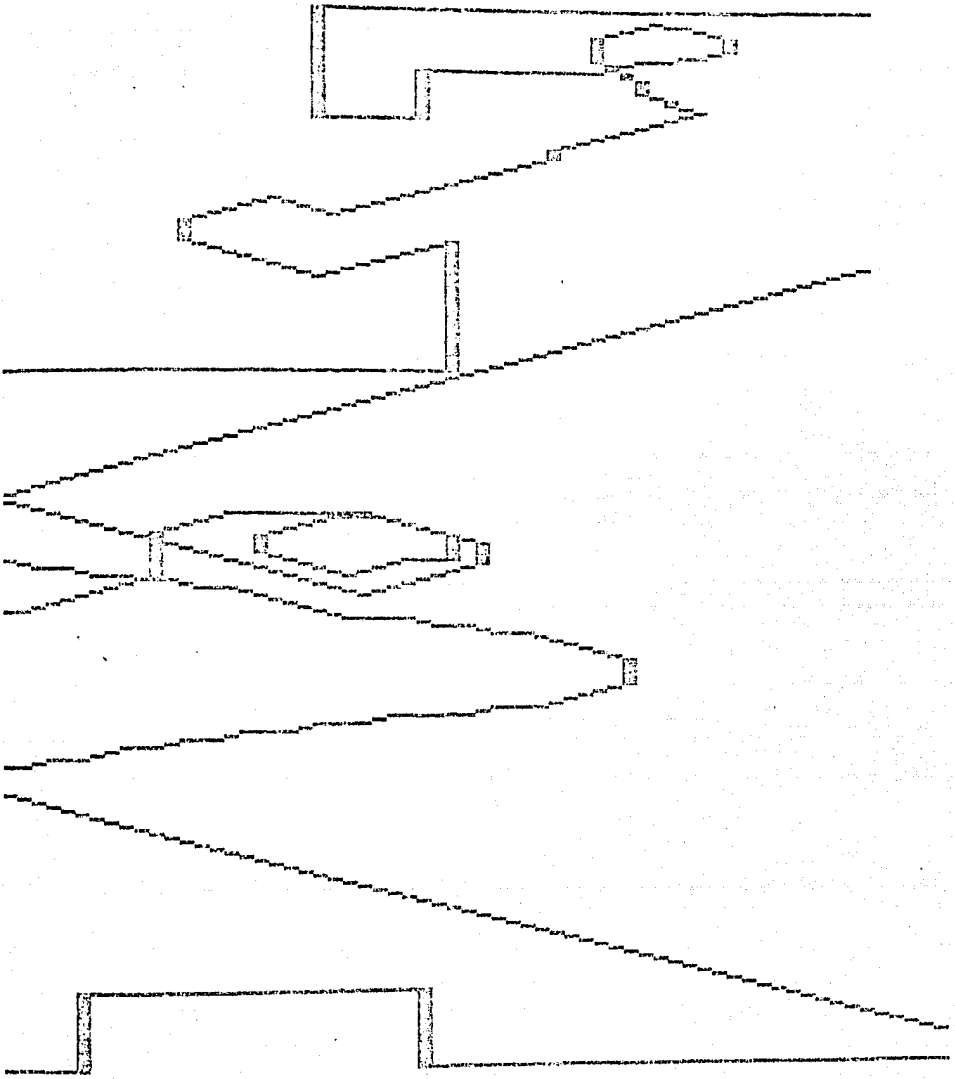
```
4930        DEC *HPAG
4940        DEC *HPGD
4950        DEX
4960        BPL P1
4970        RTS
4980  ;*********************************
4990  REG   DEC *I192,X  CAL RENGL INME-
5000        BEQ RTRG     DIATO ANTERIOR;
5010        LDA 03F      SI X=0 PAG1
5020        AND *I192,X  SI X=3 PAG2
5030        BEQ M3
5040        LDA 007      FOR I=191 T 1,-1
5050        AND *I192,X  IF I MOD 64=0
5060        BEQ M4         LPAG=*+$1F58
5070        SEC          ELSE
5080        LDA *HPAG,X  IF I MOD 8 = 0
5090        SBC 004        LPAG=*+$1B80
5100        STA *HPAG,X  ELSE
5110        JMP RTRG       LPAG=*-$400
5120  M4    CLC
5130        LDA *LPAG,X
5140        ADC 080
5150        STA *LPAG,X
5160        LDA *HPAG,X
5170        ADC 01B
5180        STA *HPAG,X
5190        JMP RTRG
5200  M3    CLC
5210        LDA *LPAG,X
5220        ADC 058
5230        STA *LPAG,X
5240        LDA *HPAG,X
5250        ADC 01F
5260        STA *HPAG,X
5270  RTRG  RTS
5280  ;*********************************
5290        LDA 020
5300        STA HPAG
5310        LDA $C057
5320        LDA $C053
5330        LDA $C050
5340        LDA 000
5350        TAX
5360        TAY
5370        JSR $D02E
5380  FIN   RTS
5390        .EN
```

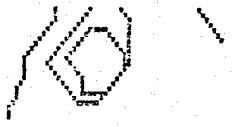












B I B L I O G R A F I A

Apple II Basic Programming Manual, 1978

Applesoft Basic Programming Reference Manual, 1978

The Applesoft Tutorial, 1979

Dos 3.2 Disk Operating System Instructional and Reference Manual , 1979

Apple II Reference Manual, January 1978

Apple II Reference Manual, 1979

Programmer's Aid #1 Installation and Operating Manual, 1978

Programming the 6502, Rodnay Zaks, Ed. Sybex, 1980.

Principles of Interactive Computer Graphics, William M. Newman.

Data Structures, Computer Graphics and Pattern Recognition, Ed. Mc Graw Hill, New York, 1973.

Tutorial: Computer Graphics, Kellogg S. Booth

Interactive Computer Graphics. Wolfgang B. Grimm, Ed. Prentice Hall, 1978

Interactive Graphics for Computers. David Price M., Ed. Addison Wesley, 1971

The Apple Orchard, Fall 1980.

Apendice "A"

LISTA DE INSTRUCCIONES DEL PROCESADOR 6502

6502—INSTRUCTION SET: HEX AND TIMING

MNEMONIC		IMPLIED			ACCUM			ABSOLUTE			ZERO PAGE			IMMEDIATE			ABS, X			ABS, Y		
		OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
ADC	(1)							6D	4	3	65	3	2	69	2	2	7D	4	3	79	4	3
AND	(1)							2D	4	3	25	3	2	29	2	2	3D	4	3	39	4	3
ASL					0A	2	1	0E	6	3	06	5	2				1E	7	3			
BCC	(2)																					
BCS	(2)																					
BEQ	(2)																					
BIT								2C	4	3	24	3	2									
BMI	(2)																					
BNE	(2)																					
BPL	(2)																					
BRK		00	7	1																		
BVC	(2)																					
BVS	(2)																					
CLC		18	2	1																		
CLD		D8	2	1																		
CLI		58	2	1																		
CLV		B8	2	1																		
CMP								CD	4	3	C5	3	2	C9	2	2	DD	4	3	D9	4	3
CPX								EC	4	3	E4	3	2	E0	2	2						
CPY								CC	4	3	C4	3	2	C0	2	2						
DEC								CE	6	3	C6	5	2				DE	7	3			
DEX		CA	2	1																		
DEY		88	2	1																		
EOR	(1)							4D	4	3	45	3	2	49	2	2	5D	4	3	59	4	3
INC								EE	6	3	E6	5	2				FE	7	3			
INX		E8	2	1																		
INY		C8	2	1																		
JMP								4C	3	3												
JSR								20	6	3												
LDA	(1)							AD	4	3	A5	3	2	A9	2	2	BD	4	3	B9	4	3
LDX	(1)							AE	4	3	A6	3	2	A2	2	2				BE	4	3
LDY	(1)							AC	4	3	A4	3	2	A0	2	2	BC	4	3			
LSR					4A	2	1	4E	6	3	46	5	2				5E	7	3			
NOP		EA	2	1																		
ORA								0D	4	3	05	3	2	09	2	2	1D	4	3	19	4	3
PHA		48	3	1																		
PHP		08	3	1																		
PLA		68	4	1																		
PLP		28	4	1																		
ROL					2A	2	1	2E	6	3	26	5	2				3E	7	3			
ROR					6A	2	1	6E	6	3	66	5	2				7E	7	3			
RTI		40	6	1																		
RTS		60	6	1																		
SBC	(1)							ED	4	3	E5	3	2	E9	2	2	FD	4	3	F9	4	3
SEC		38	2	1																		
SED		F8	2	1																		
SEI		78	2	1																		
STA								8D	4	3	85	3	2				9D	5	3	99	5	3
STX								8E	4	3	86	3	2									
STY								8C	4	3	84	3	2									
TAX		AA	2	1																		
TAY		A8	2	1																		
TSX		BA	2	1																		
TXA		8A	2	1																		
TXS		9A	2	1																		
TYA		98	2	1																		

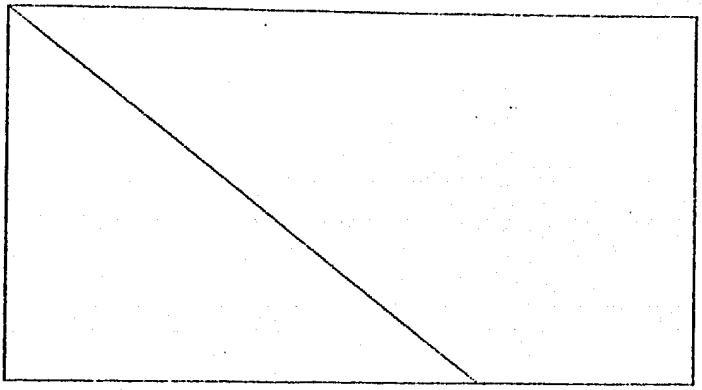
MNEMONIC	(IND, X)			(IND), Y			Z PAGE, X			RELATIVE			INDIRECT			Z PAGE, Y			PROCESSOR STATUS CODES								MNEMONIC
	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	B	D	I	Z	C	
ADC	61	6	2	71	5	2	75	4	2										•	•				•	•	ADC
AND	21	6	2	31	5	2	35	4	2										•					•		AND
ASL							16	6	2										•					•	•	ASL
BCC										90	2	2														BCC
BCS										B0	2	2														BCS
BEQ										F0	2	2														BEQ
BIT																			M7	M6				•		BIT
BMI										30	2	2														BMI
BNE										D0	2	2														BNE
BPL										10	2	2														BPL
BRK																					1		1			BRK
BVC										50	2	2														BVC
BVS										70	2	2														BVS
CLC																									0	CLC
CLD																						0				CLD
CLI																							0			CLI
CLV																				0						CLV
CMP	C1	6	2	D1	5	2	D5	4	2										•					•	•	CMP
CPX																			•					•	•	CPX
CPY																			•					•	•	CPY
DEC							D6	6	2										•					•		DEC
DEX																			•					•		DEX
DEY																			•					•		DEY
EOR	41	6	2	51	5	2	55	4	2										•					•		EOR
INC							F6	6	2										•					•		INC
INX																			•					•		INX
INY																			•					•		INY
JMP													6C	5	3											JMP
JSR																										JSR
LDA	A1	6	2	B1	5	2	B5	4	2										•					•		LDA
LDX																B6	4	2	•					•		LDX
LDY							B4	4	2										•					•		LDY
LSR							56	6	2										0					•	•	LSR
NOP																										NOP
ORA	01	6	2	11	5	2	15	4	2										•					•		ORA
PHA																										PHA
PHP																										PHP
PLA																			•					•		PLA
PLP																			•	•	•	•	•	•	•	PLP
ROL							36	6	2										•					•	•	ROL
ROR							76	6	2										•					•	•	ROR
RTI																			•	•	•	•	•	•	•	RTI
RTS																										RTS
SBC	E1	6	2	F1	5	2	F5	4	2										•	•				•	•	SBC
SEC																									1	SEC
SED																					1					SED
SEI																							1			SEI
STA	81	6	2	91	6	2	95	4	2																	STA
STX																96	4	2								STX
STY							94	4	2																	STY
TAX																			•					•		TAX
TAY																			•					•		TAY
TSX																			•					•		TSX
TXA																			•					•		TXA
TXS																										TXS
TYA																			•					•		TYA

(1) Add 1 to n if crossing page boundary

(2) Add 2 to n if branch within page
Add 3 to n if branch to another page

Apendice "B"

TIEMPOS DE EJECUCION EN APPLESOFT E INTEGER BASIC



Esta es la grafica que se realiza por medio de los tres programas que a continuacion se presentan y que sirven para mostrar un estudio comparativo de tiempos entre Applesoft, Integer Basic y Ensamblador.

Hay que hacer notar que los tiempos fueron obtenidos por medio de un cronometro accionado manualmente, por lo que los tiempos reportados pueden tener un margen de error, el cual en el programa en ensamblador consideramos es muy grande.

```
]REM PROGRAMA EN APPLESOFT
]LIST
10  HGR : HCOLOR= 3:X = 0 Y = 0
20  HPLOT X,Y X = X + 1:Y = Y + 1
      GOTO 20
40  REM  ESTE PROGRAMA TARDA 2.4
      SEG


]REM PROGRAMA EN INTBCOR BASIC
]LIST
   10 X0=Y0=COLR X0=0 Y0=0:COLR=127
      CALL -11200
   20 CALL -11505
   30 X0=X0+1 Y0=Y0+1  GOTO 20
   40 REM  ESTE PROGRAMA TARDA 1 6 SEG


]REM PROGRAMA EN ENSAMBLADOR 6502

0050  XOL   DL 0320       X-COORD 0-279
0060  XOH   DL 0321
0065  YO    DL 0322       Y-COORD 0-191
0080  HCOL  DL 0324       COLOR PLOT
0111  POSN  DL 002E
0112  ENT   DL F00C       LEE CARACTER
0120        JSR F0000     INIC HI-RES
0130        LDA 000
0140        TAX
0150        TAY
0160        JSR POSN
0180        LDA 07F
0190        STA HCOL
0240        JSR PINT
0290        RTS
0295  ;*********************************
0760  PINT JSR PREN
0770       JMP PINT
0775       RTS
0780  ;*********************************
0930  PREN LDA 002
0940       JSR NY0       NVA COOR Y0
0950       PHA
0960       JSR NX0       NVA COORD X0
0970       PLA
0980       JSR #007A
0990       RTS
1070  NY0  LDA 001
1080       CLC
1090       ADC YO
1100       CMP 0C0       ERR SI )191
1110       BCC RETY
1120       JMP #EE68     RUTINA DE ERROR
1130  EQY0 LDA YO
1140  RETY RTS
1150  NX0  TXA
1220       LDA 001       AC=3.6.9
1250       CLC
1260       ADC XOL
1270       TAX
1280       LDA XOH
1290       ADC 000
1300  FILT TAY
1310       CPX 018       ERR SI )279
1320       SBC 001
1330       BCC RETX
1340       JMP #EE68     RUTINA DE ERROR
1450  RETX RTS
1460  ;*********************************
1560       .EN

]REM ESTE PROGRAMA TARDA MENOS DE 0 6 SEG
```

Apendice "C"

PICTURE BUFFER DE HIGH RESOLUTION GRAPHICS

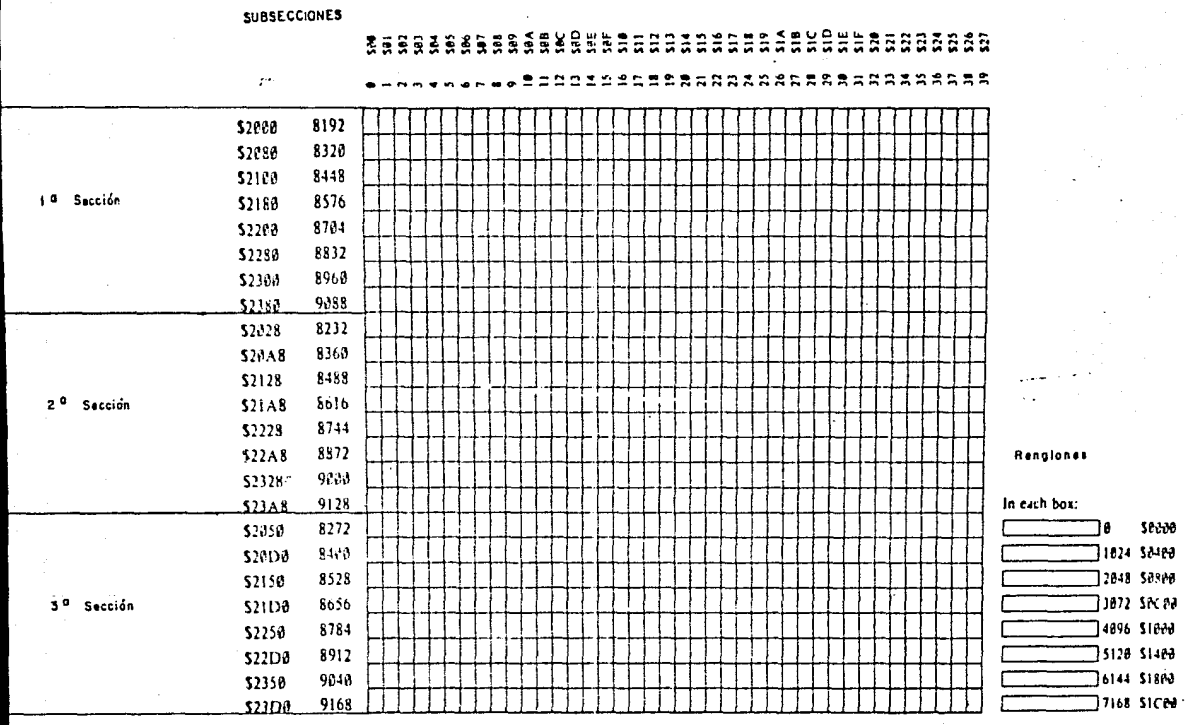


Figure 3. Map of the High-Resolution Graphics Screen

Apendice "D"

SUBRUTINAS DE HIGH RESOLUTION GRAPHICS

```
                        *****************************
                        *                           *
                        * APPLE-II HI-RESOLUTION    *
                        *  GRAPHICS SUBROUTINES     *
                        *                           *
                        *    BY WOZ   9/13/77       *
                        *                           *
                        *  ALL RIGHTS RESERVED      *
                        *                           *
                        *****************************

                        * HI-RES EQUATES
                SHAPEL  EQU   $1A POINTER TO
                SHAPEH  EQU   $1B SHAPE LIST
                HCOLOR1 EQU   $1C RUNNING COLOR MASK
                COUNTH  EQU   $1D
                HBASL   EQU   $26 BASE ADR FOR CURRENT
                HBASH   EQU   $27  HI-RES PLOT LINE.  A
                HMASK   EQU   $30
                A1L     EQU   $3C MONITOR A1.
                A1H     EQU   $3D
                A2L     EQU   $3E MONITOR A2.
                A2H     EQU   $3F
                LOMEML  EQU   $4A BASIC 'START OF VARS'.
                LOMEMH  EQU   $4B
                DXL     EQU   $50 DELTA-X FOR HLIN, SHAPE
                DXH     EQU   $51
                SHAPEX  EQU   $51 SHAPE TEMP.
                DY      EQU   $52 DELTA-Y FOR HLIN, SHAPE
                QDRNT   EQU   $53 ROT QUADRANT (SHAPE).
                EL      EQU   $54 ERROR FOR HLIN.
                EH      EQU   $55
                PPL     EQU   $CA BASIC START OF PROG PTR.
                PPH     EQU   $CB
                PVL     EQU   $CC BASIC END OF VARS PTR.
                PVH     EQU   $CD
                ACL     EQU   $CE BASIC ACC.
                ACH     EQU   $CF
                XOL     EQU   $320 PRIOR X-COORD SAVE
                XOH     EQU   $321 AFTER HLIN OR HPLOT.
                YO      EQU   $322 HLIN, HPLOT Y-COORD SAVE.
                BXSAV   EQU   $323 X-REG SAVE FOR BASIC.
                HCOLOR  EQU   $324 COLOR FOR HPLOT, HPOSN
                HNDX    EQU   $325 HORIZ OFFSET SAVE.
                HPAG    EQU   $326 HI-RES PAGE (=$20 NORMAL)
                SCALE   EQU   $327 SCALE FOR SHAPE, MOVE.
                SHAPTL  EQU   $328 START OF
                SHAPTH  EQU   $329    SHAPE TABLE.
                COLLSN  EQU   $32A COLLISION COUNT.
                HIRES   EQU   $C057 SWITCH TO HI-RES VIDEO
                MIXSET  EQU   $C053 SELECT TEXT/GRAPHICS MIX
                TXTCLR  EQU   $C050 SELECT GRAPHICS MODE.
                MEMFUL  EQU   $E36B BASIC MEM FULL ERROR.
                RNGERR  EQU   $EE68 BASIC RANGE ERROR.
                ACADR   EQU   $F11E 2-BYTE TAPE READ SETUP.
                RD2BIT  EQU   $FCFA TWO-EDGE TAPE SENSE.
                READ    EQU   $FEFD TAPE READ (A1,A2)
                READX1  EQU   $FFD2 READ WITHOUT HEADER.

                * HIGH RESOLUTION GRAPHICS INITS
                *
                * ROM VERSION $D000 TO $D3FF
                *
                        ORG   $D000
                        OBJ   $A000
D000 A9 20      SETHRL  LDA   #$20 INIT FOR $2000-3FFF
D002 8D 26 03   HRG     STA   HPAG HI-RES SCREEN MEMORY.
D005 AD 57 C0   HRG1    LDA   HIRES SET HIRES DISPLAY MODE
D008 AD 53 C0           LDA   MIXSET WITH TEXT AT BOTTOM
D00B AD 50 C0           LDA   TXTCLR SET GRAPHICS DISPLAY MODE
D00E A9 00              LDA   #$0
D010 85 1C      HCLR    STA   HCOLOR1 SET FOR BLACK BKGND.
D012 AD 26 03   BKGND0  LDA   HPAG
D015 85 1B              STA   SHAPEH INIT HI-RES SCREEN MEM
D017 A0 00              LDY   #$0 FOR CURRENT PAGE, NORMALLY
D019 84 1A              STY   SHAPE! $2000-3FFF OR $4000-5FFF
D01B A5 1C      BKGND1  LDA   HCOLOR1
D01D 91 1A              STA   (SHAPEL),Y
D01F 20 A2 D0           JSR   CSHFT2 (SHAPEL,H) WILL SPECIFY
D022 C8                 INY   32 SEPARATE PAGES
D023 D0 F6              BNE   BKGND1 THROUGHOUT THE INIT.
D025 E6 1B              INC   SHAPEH
D027 A5 1B              LDA   SHAPEH
D029 29 1F              AND   #$1F TEST FOR DONE.
D02B D0 EE              BNE   BKGND1
D02D 60                 RTS

                        * HI-RES GRAPHICS POSITION AND PLOT SUBRS
D02E 8D 22 03   HPOSN   STA   YO ENTER WITH Y IN A-REG.
D031 8E 20 03           STX   XOL XL IN X-REG.
D034 8C 21 03           STY   XOH AND XH IN Y-REG.
D037 48                 PHA
D038 29 C0              AND   #$C0
D03A 85 26              STA   HBASL FOR Y-COORD = ODABCDEF.
D03C 4A                 LSR   ;CALCULATES BASE ADDRESS
D03D 4A                 LSR   ; IN HBASL, HBASH FOR
D03E 05 26              ORA   HBASL    ACCESSING SCREEN MEM
D040 85 26              STA   HBASL    VIA (HBASL),Y ADDRESSING MODE
D042 68                 PLA
D043 85 27              STA   HBASH
D045 0A                 ASL   ;CALCULATES
D046 0A                 ASL   ;HRASH = PPPFGHCD.
D047 0A                 ASL   ;HRASL = EABAB000
D048 26 27              ROL   HBASH
D04A 0A                 ASL   ;WHERE PPP=001 FOR $2000-3FFF
D04B 26 27              ROL   HBASH SCREEN MEM RANGE AND
D04D 0A                 ASL   ; PPP=010 FOR $4000-7FFF
D04E 66 26              ROR   HBASL (GIVEN Y-COORD=ABCDEFGH)
D050 A5 27              LDA   HBASH
D052 29 1F              AND   #$1F
D054 0D 26 03           ORA   HPAG
D057 85 27              STA   HBASH
D059 8A                 TXA   DIVIDE XO BY 7 FOR
D05A C0 00              CPY   #$0 INDEX FROM BASE ADR
D05C F0 05              BEQ   HPOSN2 (QUOTIENT) AND BIT
D05E A0 23              LDY   #$23 WITHIN SCREEN MEM BYTE
D060 69 04              ADC   #$4 (MASK SPEC'D BY REMAINDER)
D062 CB         HPOSN1  INY
D063 E9 07      HPOSN2  SBC   #$7 SUBTRACT OUT SEVENS.
D065 B0 FB              BCS   HPOSN1
D067 8C 25 03           STY   HNDX WORKS FOR XO FROM
D06A AA                 TAX   0 TO 279, LOW-ORDER
D06B BD EA D0           LDA   MSKTBL-249,X BYTE IN X-REG.
D06E 85 30              STA   HMASK HIGH IN Y-REG ON ENTRY
D070 98                 TYA
D071 4A                 LSR   ; IF ON ODD BYTE (CARRY SET)
D072 AD 24 03           LDA   HCOLOR THEN ROTATE HCOLOR ONE
D075 85 1C      HPOSN3  STA   HCOLOR1 BIT FOR 180 DEGREE SHIFT
D077 B0 29              BCS   CSHFT2 PRIOR TO COPYING TO HCOLOR1.
D079 60                 RTS
D07A 20 2E D0   HPLOT   JSR   HPOSN
D07D A5 1C      HPLOT1  LDA   HCOLOR1 CALC BIT POSN IN HBASL,H
D07F 51 26              EOR   (HBASL),Y HNDX, AND HMASK FROM
D081 25 30              AND   HMASK Y-COORD IN A-REG,
D083 51 26              EOR   (HBASL),Y X-COORD IN X,Y-REGS.
D085 91 26              STA   (HBASL),Y FOR ANY '1' BITS OF HMASK
D087 60                 RTS   SUBSTITUTE CORRESPONDING
                              BIT OF HCOLOR1.
```

```
                    140 *  HI-RES GRAPHICS L,R,U,D SUBRS
D038 10 24          141 LFTRT  BPL   RIGHT USE SIGN FOR LFT/RT SELECT
D03A A5 30          142 LEFT   LDA   HMASK
D03C 4A             143        LSR   ; SHIFT LOW-ORDER
D03D B0 05          144        BCS   LEFT1 7 BITS OF HMASK
D03F 49 C0          145        EOR   #$C0 ONE BIT TO LSB.
D041 85 30          146 LR1    STA   HMASK
D043 60             147        RTS
D044 88             148 LEFT1  DEY   DECR HORIZ INDEX.
D045 10 02          149        BPL   LEFT2
D047 A0 27          150        LDY   #$27 WRAP AROUND SCREEN
D049 A9 C0          151 LEFT2  LDA   #$C0 NEW HMASK, RIGHTMOST
D04B 85 30          152 NEWNDX STA   HMASK DOT OF BYTE.
D04D 8C 25 03       153        STY   HNDX UPDATE HORIZ INDEX.
D050 A5 1C          154 CSHIFT LDA   HCOLOR1
D052 0A             155 CSHFT2 ASL   ; ROTATE LOW-ORDER
D053 C9 C0          156        CMP   #$C0 7 BITS OF HCOLOR1
D055 10 06          157        BPL   RTS1 ONE BIT POSN.
D057 A5 1C          158        LDA   HCOLOR1
D059 49 7F          159        EOR   #$7F ZXYXYXYX -> ZYXYXYXY
D05B 85 1C          160        STA   HCOLOR1
D05D 60             161 RTS1   RTS
D05E A5 30          162 RIGHT  LDA   HMASK
D060 0A             163        ASL   ; SHIFT LOW-ORDER
D061 49 80          164        EOR   #$80 7 BITS OF HMASK
D063 30 DC          165        BMI   LR1 ONE BIT TO MSB.
D065 A9 81          166        LDA   #$81
D067 C8             167        INY   NEXT BYTE.
D068 C0 28          168        CPY   #$28
D06A 90 DF          169        BCC   NEWNDX
D06C A0 00          170        LDY   #$0 WRAP AROUND SCREEN IF >279
D06E B0 DB          171        BCS   NEWNDX ALWAYS TAKEN.

                    173 *  L,R,U,D. SUBROUTINES.
D070 18             174 LRUDX1 CLC   NO 90 DEG ROT (X-OR).
D071 A5 51          175 LRUDX2 LDA   SHAPEX
D073 29 04          176        AND   #$4 IF B2=0 THEN NO PLOT.
D075 F0 27          177        BEQ   LRUD4
D077 A9 7F          178        LDA   #$7F FOR EX-OR INTO SCREEN MEM
D079 25 30          179        AND   HMASK
D07B 31 26          180        AND   (HBASL),Y SCREEN BIT SET?
D07D D0 18          181        BNE   LRUD3
D07F EE 2A 03       182        INC   COLLSN
D082 A9 7F          183        LDA   #$7F
D084 25 30          184        AND   HMASK
D086 10 12          185        BPL   LRUD3 ALWAYS TAKEN.
D088 18             186 LRUD1  CLC   NO 90 DEG ROT.
D089 A5 51          187 LRUD2  LDA   SHAPEX
D08B 29 04          188        AND   #$4 IF B2=0 THEN NO PLOT.
D08D F0 0F          189        BEQ   LRUD4
D08F 31 26          190        LDA   (HBASL),Y
D091 45 1C          191        EOR   HCOLOR1 SET HI-RES SCREEN BIT
D093 25 30          192        AND   HMASK TO CORRESPONDING HCOLOR1
D095 D0 03          193        BNE   LRUD3 IF BIT OF SCREEN CHANGES
D097 EE 2A 03       194        INC   COLLSN THEN INCR COLLSN DETECT
D09A 51 26          195 LRUD3  EOR   (HBASL),Y
D09C 91 26          196        STA   (HBASL),Y
D09E A5 51          197 LRUD4  LDA   SHAPEX ADD QDRNT TO
D0A0 65 53          198        ADC   QDRNT SPECIFIED VECTOR
D0A2 29 03          199        AND   #$3 AND MOVE LFT, RT,
                    200 EQ3    EQU   *-1 UP, OR DWN BASED
D0A4 C9 02          201        CMP   #$2 ON SIGN AND CARRY.
D0A6 6A             202        ROR
D0A7 B0 8F          203 LRUD   BCS   LFTRT
D0A9 30 30          204 UPDWN  BMI   DOWN4 SIGN FOR UP/DWN SELECT
D0AB 18             205 UP     CLC
D0AC A5 27          206        LDA   HBASH CALC BASE ADDRESS
D0AE 2C EA D1       207        BIT   EQ1C (ADR OF LEFTMOST BYTE)
D1D1 D0 22          208        BNE   UP4 FOR NEXT LINE UP
D1D3 06 26          209        ASL   HBASL IN (HBASL, HBASH)

D1D5 B0 1A          210        BCS   UP2 WITH 192-LINE WRAPAROUND
D1D7 2C F3 D0       211        BIT   EQ3
D1DA F0 05          212        BEQ   UP1
D1DC 69 1F          213        ADC   #$1F **** BIT MAP ****
D1DE 38             214        SEC
D1DF B0 12          215        BCS   UP3 FOR ROW = ABCDEFGH.
D1E1 69 23          216 UP1    ADC   #$23
D1E3 48             217        PHA
D1E4 A5 26          218        LDA   HBASL HBASL = EABAB000
D1E6 69 B0          219        ADC   #$B0 HBASH = PPPFGHCD
D1E8 B0 02          220        BCS   UP3
D1EA 69 F0          221        ADC   #$F0 WHERE PPP=001 FOR PRIMARY
D1EC 85 26          222 UP3    STA   HBASL HI-RES PAGE (#2000-#3FFF)
D1EE 68             223        PLA
D1EF B0 02          224        BCS   UP3
D1F1 69 1F          225 UP2    ADC   #$1F
D1F3 66 26          226 UP3    ROR   HBASL
D1F5 69 FC          227 UP4    ADC   #$FC
D1F7 85 27          228 UPDWN1 STA   HBASH
D1F9 60             229        RTS
D1FA 18             230 DOWN   CLC
D1FB A5 27          231 DOWN4  LDA   HBASH
D1FD 69 04          232        ADC   #$4 CALC BASE ADR FOR NEXT LINE
                    233 EQ4    EQU   *-1 DOWN TO (HBASL,HBASH)
D1FF 2C EA D1       234        BIT   EQ1C
D202 D0 F3          235        BNE   UPDWN1

D204 06 26          236        ASL   HBASL WITH 192-LINE WRAPAROUND
D206 90 19          237        BCC   DOWN1
D208 69 E0          238        ADC   #$E0
D20A 18             239        CLC
D20B 2C 2E D1       240        BIT   EQ4
D20E F0 13          241        BEQ   DOWN2
D210 A5 26          242        LDA   HBASL
D212 69 50          243        ADC   #$50
D214 49 F0          244        EOR   #$F0
D216 F0 02          245        BEQ   DOWN3
D218 49 F0          246        EOR   #$F0
D21A 85 26          247 DOWN3  STA   HBASL
D21C AD 26 03       248        LDA   HPAG
D21F 90 02          249        BCC   DOWN2
D221 69 E0          250 DOWN1  ADC   #$E0
D223 66 26          251 DOWN2  ROR   HBASL
D225 90 D0          252        BCC   UPDWN1

                    254 *  HI-RES GRAPHICS LINE DRAW SUBRS
D227 48             255 HLINRL PHA
D228 A9 00          256        LDA   #$0 SET (XOL,XOH) AND
D22A 8D 20 03       257        STA   XOL YO TO ZERO FOR
D22D 8D 21 03       258        STA   XOH REL LINE DRAW
D230 8D 22 03       259        STA   YO (DX, DY).
D233 68             260        PLA
D234 48             261 HLIN   PHA   ON ENTRY
D235 38             262        SEC   XL: A-REG
D236 ED 20 03       263        SBC   XOL XH: X-REG
D239 48             264        PHA   Y: Y-REG
D23A 8A             265        TXA
D23B ED 21 03       266        SBC   XOH
D23E 85 53          267        STA   QDRNT CALC ABS(X-XO)
D240 B0 0A          268        BCS   HLIN2 IN (DXL,DXH)
D242 68             269        PLA
D243 49 FF          270        EOR   #$FF X DIR TO SIGN BIT
D245 69 01          271        ADC   #$1 OF QDRNT.
D247 48             272        PHA   0=RIGHT (DX POS)
D248 A9 00          273        LDA   #$0 1=LEFT (DX NEG)
D24A E5 53          274        SBC   QDRNT
D24C 85 51          275 HLIN2  STA   DXH
D24E 85 55          276        STA   EH INIT (EL,EH) TO
D250 68             277        PLA   ABS(X-XO)
D251 85 50          278        STA   DXL
```

```
D163 85 54      279         STA   EL
D165 68         280         PLA
D166 8D 20 03   281         STA   XOL
D169 8E 21 03   282         STX   XOH
D16C 98         283         TYA
D16D 18         284         CLC
D16E ED 22 03   285         SBC   YO CALC -ABS(Y-O)-1
D171 90 04      286         BCC   HLIN3 IN DY.
D173 49 FF      287         EOR   #$FF
D175 69 FE      288         ADC   #$FE
D177 85 52      289 HLIN3   STA   DY ROTATE Y DIR INTO
D179 6C 22 03   290         STY   YO QDRNT SIGN BIT
D17C 66 53      291         ROR   QDRNT (O=UP, 1=DOWN)
D17E 38         292         SEC
D17F E5 50      293         SBC   DXL INIT (COUNTL, COUNTH).
D181 AA         294         TAX   TO -(DELTX+DELTY+1)
D182 A9 FF      295         LDA   #$FF
D184 E5 51      296         SBC   DXH
D186 85 1D      297         STA   COUNTH
D188 AC 25 03   298         LDY   HNDX HORIZ INDEX
D18B B0 05      299         BCS   MOVEX2 ALWAYS TAKEN.
D18D 0A         300 MOVEX   ASL   ; MOVE IN X-DIR. USE
D18E 20 BB D0   301         JSR   LFTRT QDRNT B6 FOR LFT/RT SELECT
D191 38         302         SEC
D192 A5 54      303 MOVEX2  LDA   EL ASSUME CARRY SET.
D194 65 52      304         ADC   DY (EL,EH)-DELTY TO (EL,EH)
D196 85 54      305         STA   EL NOTE: DY IS (-DELTY)-1
D198 A5 55      306         LDA   EH CARRY CLR IF (EL,EH)
D19A E9 00      307         SBC   #$0 GOES NEG.
D19C 85 55      308 HCOUNT  STA   EH
D19E B1 26      309         LDA   (HRASL),Y SCREEN BYTE.
D1A0 45 1C      310         EOR   HCOLOR1 PLOT DOT OF HCOLOR1.
D1A2 25 30      311         AND   HMASK CURRENT BIT MASK.
D1A4 51 26      312         EOR   (HRASL),Y
D1A6 91 26      313         STA   (HBASL),Y
D1A8 E8         314         INX   DONE (DELTX+DELTY)
D1A9 D0 04      315         BNE   HLIN4 DOTS?
D1AB E6 1D      316         INC   COUNTH
D1AD F0 6B      317         BEQ   RTS2 YES. RETURN.
D1AF A5 53      318 HLIN4   LDA   QDRNT FOR DIRECTION TEST
D1B1 B0 DA      319         BCS   MOVEX IF CAR SET, (EL, EH) POS
D1B3 20 F9 D0   320         JSR   UPDWN IF CLR, NEG, MOVE YDIR
D1B6 18         321         CLC
D1B7 A5 54      322         LDA   EL (EL,EH)+DELTX
D1B9 65 50      323         ADC   DXL TO (EL,EH).
D1BB 85 54      324         STA   EL
D1BD A5 55      325         LDA   EH CAR SET IF (EL,EH) GOES POS
D1BF 65 51      326         ADC   DXH
D1E1 50 D9      327         BVC   HCOUNT ALWAYS TAKEN.
D1E3 81         328 MSKTBL  HEX   81 LEFTMOST BIT OF BYTE.
D1E4 82 84 88   329         HEX   82,84,88
D1E7 90 A0      330         HEX   90,A0
D1E9 C0         331         HEX   C0 RIGHTMOST BIT OF BYTE.
D1EA 1C         332 EQ1C    HEX   1C
D1EB FF FE FA   333 COS     HEX   FF,FE,FA,F4,EC,E1,D4,C5,B4
D1F4 A1 8D 7B   334         HEX   A1,8D,7B,61,49,31,18,FF
                336 * HI-RES GRAPHICS COORDINATE RESTORE SUBR
D1FC A5 26      337 HFIND   LDA   HBASL
D1FE 0A         338         ASL   ; CONVERTS BASE ADR
D1FF A5 27      339         LDA   HBASH TO Y-COORD.
D201 29 03      340         AND   #$3
D203 2A         341         ROL   ; FOR HBASL = EABABOOO
D204 05 26      342         ORA   HBASL HBASH = PPPFGHCD
D206 0A         343         ASL
D207 0A         344         ASL   ; GENERATE
D208 0A         345         ASL   ; Y-COORD = ABCDEFGH
D209 8D 22 03   346         STA   YO
D20C A5 27      347         LDA   HBASH (PPP=SCREEN PAGE,
D20E 4A         348         LSR   ; NORMALLY 001 FOR
D20F 4A         349         LSR   ; $2000-$3FFF
D210 29 07      350         AND   #$7 HI-RES SCREEN)
D212 0D 22 03   351         ORA   YO
D215 8D 22 03   352         STA   YO CONVERTS HNDX (INDEX
D218 AD 23 03   353         LDA   HNDX FROM BASE ADR)
D21B 0A         354         ASL   ; AND HMASK (BIT
D21C 6D 25 03   355         ADC   HNDX MASK) TO X-COORD
D21F 0A         356         ASL   ; IN (XOL, XOH)
D220 AA         357         TAX   (RANGE $0-$133)
D221 CA         358         DEX
D222 A5 30      359         LDA   IMASK
D224 29 7F      360         AND   #$7F
D226 E8         361 HFIND1  INX
D227 4A         362         LSR
D228 D0 FC      363         BNE   HFIND1
D22A 8D 21 03   364         STA   XOH
D22D 8A         365         TXA
D22E 18         366         CLC   CALC HNDX*7 +
D22F 6D 25 03   367         ADC   HNDX LOG (BASE 2) HMASK.
D232 90 03      368         BCC   HFIND2
D234 EE 21 03   369         INC   XOH
D237 8D 20 03   370 HFIND2  STA   XOL
D23A 60         371 RTS2    RTS
                373 * HI-RES GRAPHICS SHAPE DRAW SUBR
                374 *
                375 * SHAPE DRAW
                376 * R = O TO 63
                377 * SCALE FACTOR USED (1=NORMAL)
                378 *
D23B 86 1A      379 DRAW    STX   SHAPE1 DRAW DEFINITION
D23D 84 1B      380         STY   SHAPEH POINTER.
D23F AA         381 DRAW1   TAX
D240 4A         382         LSR   ; ROT (#0-#3F)
D241 4A         383         LSR
D242 4A         384         LSR   ; QDRNT O=UP, 1=RT,
D243 4A         385         LSR   ; 2=DWN, 3=LFT.
D244 85 53      386         STA   QDRNT
D246 8A         387         TXA
D247 29 0F      388         AND   #$F
D249 AA         389         TAX
D24A BC EB D1   390         LDY   COS,X SAVE COS AND SIN
D24D 84 50      391         STY   DXL VALS IN DXL AND DY.
D24F 49 0F      392         EOR   #$F
D251 AA         393         TAX
D252 BC EC D1   394         LDY   COS+1,X
D255 C8         395         INY
D256 84 52      396         STY   DY
D258 AC 25 03   397 DRAW2   LDY   HNDX BYTE INDEX FROM
D25B A2 00      398         LDX   #$0 HI-RES BASE ADR.
D25D 8E 2A 03   399         STX   COLLSN CLEAR COLLISION COUNT.
D260 A1 1A      400         LDA   (SHAPE1,X) 1ST SHAPE DEF BYTE.
D262 85 51      401 DRAW3   STA   SHAPEX
D264 A2 80      402         LDX   #$80
D266 86 54      403         STX   EL EL,EH FOR FRACTIONAL
D268 86 55      404         STX   EH L,R,U,D VECTORS.
D26A AE 27 03   405         LDX   SCALF SCALE FACTOR.
D26D A5 54      406 DRAW4   LDA   EL
D26F 38         407         SEC   IF FRAC COS OVFL
D270 65 50      408         ADC   DXL THEN MOVE IN
D272 85 54      409         STA   EL SPECIFIED VECTOR
D274 90 04      410         BCC   DRAW5 DIRECTION.
D276 20 D8 D0   411         JSR   LRUD1
D279 18         412         CLC
D27A A5 55      413 DRAW5   LDA   EH IF FRAC SIN OVFL
D27C 65 52      414         ADC   DY THEN MOVE IN
D27E 85 55      415         STA   EH SPECIFIED VECTOR
D280 90 03      416         BCC   DRAW6 DIRECTION +90 DEG
D282 20 D9 D0   417         JSR   LRUD2
D285 CA         418 DRAW6   DEX   LOOP ON SCALE
D286 D0 E5      419         BNE   DRAW4 FACTOR
D288 A5 51      420         LDA   SHAPEX
D28A 4A         421         LSR   ; NEXT 3-BIT VECTOR
```

```
D2B8 4A          422         LSR   ; OF SHAPE DEF.
D2BC 4A          423         LSR
D2BD D0 D3       424         BNE   DRAW3 NOT DONE THIS BYTE.
D2BF E6 1A       425         INC   SHAPEL
D2C1 D0 02       426         BNE   DRAW7 NEXT BYTE OF
D2C3 E6 1B       427         INC   SHAPEH SHAPE DEFINITION.
D2C5 A1 1A       428 DRAW7   LDA   (SHAPEL,X)
D2C7 D0 C9       429         BNE   DRAW3 DONE IF ZERO.
D2C9 60          430         RTS

                 432 *  HI-RES GRAPHICS SHAPE EX-OR SUBR
                 433 *
                 434 * EX-OR SHAPE INTO SCREEN.
                 435 *
                 436 * ROT = 0 TO 3 (QUADRANT ONLY)
                 437 * SCALE 1B USED
                 438 *
D2CA 86 1A       439 XDRAW   STX   SHAPEL SHAPE DEFINITION
D2CC 84 1B       440         STY   SHAPEH POINTER.
D2CE AA          441 XDRAW1  TAX
D2CF 4A          442         LSR   ; ROT (#0-#3F)
D2D0 4A          443         LSR
D2D1 4A          444         LSR   ; QDRNT 0=UP, 1=RT,
D2D2 4A          445         LSR   ; 2=DWN, 3=LFT.
D2D3 85 93       446         STA   QDRNT
D2D5 8A          447         TXA
D2D6 29 0F       448         AND   #$F
D2D8 AA          449         TAX
D2D9 BC E8 D1    450         LDY   COS,X SAVE COS AND SIN
D2DC 84 50       451         STY   DXL VALS IN DXL AND DY.
D2DE 49 0F       452         EOR   #$F
D2E0 AA          453         TAX
D2E1 BC EC D1    454         LDY   COS+1,X
D2E4 C8          455         INY
D2E5 84 52       456         STY   DY
D2E7 AC 25 03    457 XDRAW2  LDY   HNDX INDEX FROM HI-RES
D2EA A2 00       458         LDX   #$0 BASE ADR.
D2EC 8E 2A 03    459         STX   COLLSN CLEAR COLLISION DETECT
D2EF A1 1A       460         LDA   (SHAPEL,X) 1ST SHAPE DEF BYTE.
D2F1 85 51       461 XDRAW3  STA   SHAPEX
D2F3 A2 80       462         LDX   #$80
D2F5 86 54       463         STX   EL  EL,EH FOR FRACTIONAL
D2F7 86 55       464         STX   EH L,R,U,D, VECTORS.
D2F9 AE 27 03    465         LDX   SCALE SCALE FACTOR.
D2FC A5 54       466 XDRAW4  LDA   EL
D2FE 38          467         SEC   IF FRAC COS OVFL
D2FF 65 50       468         ADC   DXL THEN MOVE IN
D301 85 54       469         STA   EL SPECIFIED VECTOR
D303 90 04       470         BCC   XDRAW5 DIRECTION
D305 20 C0 D0    471         JSR   LRUDX1
D308 18          472         CLC
D309 A5 55       473 XDRAW5  LDA   EH IF FRAC SIN OVFL
D30B 65 52       474         ADC   DY THEN MOVE IN
D30D 85 55       475         STA   EH SPECIFIED VECTOR
D30F 90 03       476         BCC   XDRAW6 DIRECTION +90 DEG.
D311 20 D9 D0    477         JSR   LRUD2
D314 CA          478 XDRAW6  DEX   LOOP ON SCALE
D315 D0 E5       479         BNE   XDRAW4 FACTOR.
D317 A5 51       480         LDA   SHAPEX
D319 4A          481         LSR   ; NEXT 3-BIT VECTOR
D31A 4A          482         LSR   ; OF SHAPE DEF.
D31B 4A          483         LSR
D31C D0 D3       484         BNE   XDRAW3
D31E E6 1A       485         INC   SHAPEL
D320 D0 02       486         BNE   XDRAW7 NEXT BYTE OF
D322 E6 1B       487         INC   SHAPEH SHAPE DEF.
D324 A1 1A       488 XDRAW7  LDA   (SHAPEL,X)
D326 D0 C9       489         BNE   XDRAW3 DONE IF ZERO.
D328 60          490         RTS

                 492 *  ENTRY POINTS FROM APPLE-II BASIC
D329 20 90 D3    493 $POSN   JSR   PCOLR POSN CALL, COLR FROM BASIC

D32C 8D 24 03    494         STA   HCOLOR
D32F 20 AF D3    495         JSR   GETYO YO FROM BASIC.
D332 48          496         PHA
D333 20 9A D3    497         JSR   GETXO XO FROM BASIC.
D336 68          498         PLA
D337 20 2E D0    499         JSR   HPOSN
D33A AE 23 03    500         LDX   $XSAV
D33D 60          501         RTS
D33E 20 F9 D2    502 $PLOT   JSR   $POSN PLOT CALL (BASIC).
D341 4C 7D D0    503         JMP   HPLOT1
D344 AD 25 03    504 $LIN1   LDA   HNDX
D347 4A          505         LSR   ; SET HCOLOR1 FROM
D348 20 90 D3    506         JSR   PCOLR BASIC VAR COLR.
D34B 20 75 D0    507         JSR   HPOSN3
D34E 20 9A D3    508 $LINE   JSR   GETXO LINE CALL, GET XO FROM BASIC
D351 8A          509         TXA
D352 48          510         PHA
D353 98          511         TYA
D354 AA          512         TAX
D355 20 AF D3    513         JSR   GETYO YO FROM BASIC
D358 A8          514         TAY
D359 68          515         PLA
D35A 20 64 D1    516         JSR   HLIN
D35D AE 23 03    517         LDX   $XSAV
D360 60          518         RTS
D361 20 90 D3    519 $GND    JSR   PCOLR BACKGROUND CALL
D364 4C 10 D0    520         JMP   $GNDO

                 522 *  DRAW ROUTINES
D367 20 F9 D2    523 $DRAW1  JSR   $POSN
D36A 20 51 D3    524 $DRAW   JSR   $DRAWX DRAW CALL FROM BASIC
D36D 20 38 D2    525         JSR   DRAW
D370 AE 23 03    526         LDX   $XSAV
D373 60          527         RTS
D374 20 F9 D2    528 $XDRW1  JSR   $POSN
D377 20 51 D3    529 $XDRAW  JSR   $DRAWX EX-OR DRAW
D37A 20 9A D2    530         JSR   XDRAW FROM BASIC.
D37D AE 23 03    531         LDX   $XSAV
D380 60          532         RTS
D381 8E 23 03    533 $DRAWX  STX   $XSAV SAVE FOR BASIC.
D384 A0 32       534         LDY   #$32
D386 20 92 D3    535         JSR   PBYTE SCALE FROM BASIC.
D389 8D 27 03    536         STA   SCALE
D38C A0 28       537         LDY   #$28
D38E 20 92 D3    538         JSR   PBYTE ROT FROM BASIC
D391 48          539         PHA   SAVE ON STACK.
D392 A0 28 03    540         LDA   SHAPEL
D395 85 1A       541         STA   SHAPEL START OF
D397 A0 29 03    542         LDA   SHAPEH SHAPE TABLE.
D39A 85 1B       543         STA   SHAPEH
D39C A0 20       544         LDY   #$20
D39E 20 92 D3    545         JSR   PBYTE SHAPE FROM BASIC.
D3A1 F0 39       546         BEQ   RERR1
D3A3 A2 00       547         LDX   #$0
D3A5 C1 1A       548         CMP   (SHAPEL,X) > NUM OF SHAPES?
D3A7 F0 02       549         BEQ   $DRWX1
D3A9 B0 31       550         BCS   RERR1 YES, RANGE ERR.
D3AB 0A          551 $DRWX1  ASL
D3AC 90 03       552         BCC   $DRWX2
D3AE E6 1B       553         INC   SHAPEH
D3B0 18          554         CLC
D3B1 A8          555 $DRWX2  TAY   SHAPE NO. * 2.
D3B2 B1 1A       556         LDA   (SHAPEL),Y
D3B4 65 1A       557         ADC   SHAPEL
D3B6 AA          558         TAX   ADD 2-BYTE INDEX
D3B7 C8          559         INY   TO SHAPE TABLE
D3B8 B1 1A       560         LDA   (SHAPEL),Y START ADR
D3BA 6D 29 03    561         ADC   SHAPEH (X LOW, Y HI).
D3BD A8          562         TAY
D3BE 68          563         PLA   ROT FROM STACK.
D3BF 60          564         RTS
```

```
                        366 *  BASIC PARAM FETCH SUBR'S
D390 A0 16              367 PCOLR   LDY   #$16
D392 B1 4A              368 PBYTE   LDA   (LOMEML),Y
D394 D0 16              369         BNE   RERR1 GET BASIC PARAM.
D396 88                 370         DEY   (ERR IF >255)
D397 B1 4A              371         LDA   (LOMEML),Y
D399 60                 372 RTSB    RTS
D39A 8E 23 03           373 GETXO   STX   $XSAV SAVE FOR BASIC.
D39D A0 05              374         LDY   #$5
D39F B1 4A              375         LDA   (LOMEML),Y XO LOW-ORDER BYIE.
D3A1 AA                 376         TAX
D3A2 C8                 377         INY
D3A3 B1 4A              378         LDA   (LOMEML),Y HI-ORDER BYTE.
D3A5 A8                 379         TAY
D3A6 E0 18              380         CPX   #$18
D3A8 E9 01              381         SBC   #$1 RANGE ERR IF >279
D3AA 90 ED              382         BCC   RTSB
D3AC 4C 6B EE           383 RERR1   JMP   RNGERR
D3AF A0 0D              384 GETYO   LDY   #$D OFFSET TO YO FROM LOMEM
D3B1 20 92 D3           385         JSR   PBYTE GET BASIC PARAM YO
D3B4 C9 C0              386         CMP   #$C0 (ERR IF >191)
D3B6 B0 F4              387         BCS   RERR1
D3B8 60                 388         RTS
                        390 *  SHAPE TAPE LOAD SUBROUTINE
D3B9 8E 23 03           391 SHLOAD  STX   $XSAV SAVE FOR BASIC.
D3BC 20 1E F1           392         JSR   ACADR READ 2-BYTE LENGTH INTO
D3BF 20 FD FE           393         JSR   READ BASIC ACC
D3C2 A9 00              394         LDA   #$00 ;START OF SHAPE TABLE IS $0800
D3C4 85 3C              395         STA   A1L
D3C6 8D 28 03           396         STA   SHAPXL
D3C9 18                 397         CLC
D3CA 65 CE              398         ADC   ACL
D3CC A8                 399         TAY
D3CD A9 08              600         LDA   #$08 ;HIGH BYTE OF SHAPE TABLE POINTER
D3CF 85 3D              601         STA   A1H
D3D1 8D 29 03           602         STA   SHAPXH
D3D4 65 CF              603         ADC   ACH
D3D6 B0 25              604         BCS   MFULL1 NOT ENOUGH MEMORY.
D3D8 C4 CA              605         CPY   PPL
D3DA 48                 606         PHA
D3DB E5 CB              607         SBC   PPH
D3DD 68                 608         PLA
D3DE B0 1D              609         BCS   MFULL1
D3E0 84 3E              610         STY   A2L
D3E2 85 3F              611         STA   A2H
D3E4 C8                 612         INY
D3E5 D0 02              613         BNE   SHLOD1
D3E7 69 01              614         ADC   #$1
D3E9 84 4A              615 SHLOD1  STY   LOMEML
D3EB 85 4B              616         STA   LOMEMH
D3ED 84 CC              617         STY   PVL
D3EF 85 CD              618         STA   PVH
D3F1 20 FA FC           619         JSR   RD2BIT
D3F4 A9 03              620         LDA   #$3 .5 SECOND HEADER.
D3F6 20 02 FF           621         JSR   READX1
D3F9 AE 23 03           622         LDX   $XSAV
D3FC 60                 623         RTS
D3FD 4C 6B E3           624 MFULL1  JMP   MEMFUL

--- END ASSEMBLY ---

TOTAL ERRORS: 00
```