UNIVERSIDAD POLITECNICA DE LAS PALMAS

INGENIERIA TECNICA DE TELECOMUNICACION

PROYECTO FIN DE CARRERA

TITULO: TECNICAS GRAFICAS CON COMPUTADOR. APLICACION A UN TRAZADOR DE GRAFICAS ACTUANDO COMO PLOTTER.

AUTOR:

TERESITA RODRIGUEZ MORALES

DIRECTOR:

SEBASTIAN SUAREZ GIL

FECHA: JUNIO - 85

_PARTE_I_

PAGINA

_PARTE_II_

PAGINA

INTRODUCCION

La base de este proyecto fin de carrera es la realización de gráficas con el plotter X-Y PM8043 existente en el laboratorio de ordenadores, que hasta dicho momento no habia tenido un uso práctico.

En el proyecto podemos distinguir dos partes perfectamente diferenciadas. Una primera parte que habla de las distintas técnicas de gráficas con computador y de los tipos de plotters. Y una segunda donde hay una amplia explicación del desarrollo del proyecto y de los programas realizados.

Para llegar a la realización de las gráficas presentadas en el último apartado, se tuvo que realizar una larga labor de pruebas, al principio con más fracasos que aciertos.

La primera conexión Sistema de Desarrollo y plotter se realizó a través del SDK-85, utilizado como interface. Para terminar con la utilización de la salida PUNCH(salida para tarjeta perforada)del MDS-221, que se adapto para que funcionara como un puerto de salida. Se sustituyó el SDK por una placa formada por un contador, inversores, buffers y dos convertidores D/A.

A lo largo del proyecto se utilizó tres lenguajes de programación.

Se comenzó con el ensamblador como primera prueba de comunicación entre el MDS-221 y el SDK. Posteriormente se utilizó el PLM80 por ser un lenguaje del alto nivel y estructurado y por último el BASIC, puesto que, los programas de las gráficas y su tratamiento se realizó en el terminal HEWLETT-PACKARD HP-3000 debido a su mayor velocidad de tratamiento de datos.

Espero por último que este proyecto sea base para la realización de trabajos posteriores, que no han tenido cabida en el mismo.

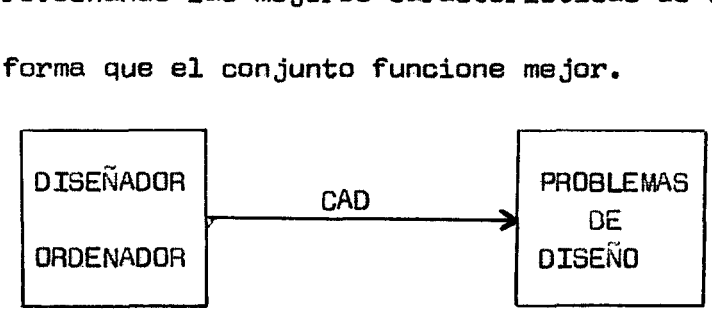
PARTE I

TECNICAS DE GRAFICAS CON COMPUTADOR

INTRODUCCION A LAS TECNICAS DE GRAFICAS CON COMPUTADOR

Las gráficas con computador es una tecnologia relativamente nueva. Existe un gran número de términos y definiciones que deben aclararse, tales como: ayuda del computador en el diseño (CAD), gráficas interactivas (IG), gráficas con computador (CG) y ayuda del computador en la fabricación (CAM).
Muchos de ellos son utilizados indistintamente o existe una gran confusión con respecto a su significado preciso.

El CAD es un conjunto de técnicas que establecen una colaboración entre el diseñador y el ordenador frente a los problemas de diseño, aprovechando las mejores caracteristicas de cada uno de ellos, de forma que el conjunto funcione mejor.



Basicamente un sistema CAD es una herramienta de diseño en el cual el computador se usa para analizar varios aspectos de un producto diseñado. El sistema CAD supone el proceso de diseño a todos los niveles: conceptual, preliminar y diseño final.

El computador puede calcular varios aspectos del producto, tales como su resistencia, inflexibilidad y peso. El diseñador puede entonces textear el producto en diferentes condiciones ambientales, tales como cambio de temperatura, o bajo distintas fuerzas mecánicas.

Aunque los sistemas CAD no emplea necesariamente gráficas con computador, la representación del objeto diseñado en una pantalla es una de las caracteristicas más valiosas de los sistemas CAD. La visión del objeto es usualmente representado en la superficie de un CRT. Las gráficas con computador permite al diseñador estudiar el objeto por rotación en la pantalla del computador, separándolo en segmentos, agrandándolo en una parte especifica para observarlo con detalle y estudiando el movimiento del mecanismo con ayuda de programas cinemáticos.

El producto final de muchos sistemas CAD se dibuja en un plotter interconectado con el computador. Uno de los problemas más dificil de los dibujos CAD es la eliminación de lineas ocultas. El computador produce el dibujo como un diagrama formado de lineas. Después de que el computador defina el objeto, se representará todas las superficies del mismo, sin tener en cuenta si

estan localizadas en la parte delantera o trasera, lo cual normalmente el ojo no puede ver.

Varios métodos se usan para generar el dibujo de una parte en la pantalla del computador. Un método es usar un modelo geométrico apropiado, en el cual las formas fundamentales y los elementos básicos se usan para construir el dibujo. Las longitudes y los arcos de los radios se pueden modificar. Por ejemplo, un cilindro es un elemento básico; la subtracción de un cilindro de radio y longitud especifico creará un orificio en la parte representada. Cualquier variación, sin embargo, mantiene la geometria completa de la parte.

Otros sistemas CAD usan tecnología de grupo en el diseño de partes. La tecnología de grupo es un método de codificación y agrupación de partes sobre la base de semejanzas en función o estructura o en las formas en que son producidas.

Recientemente los sistemas CAD estan usando el método de elementos finito (FEM) de análisis de peso. Con esta entrada el objeto a ser analizado se representa por un modelo consistente en pequeños elementos, cada uno de los cuales tiene un peso y deflexión caracteristicas. El análisis requiere la solución de

muchas ecuaciones, una tarea de la cual se encarga el computador.

Con cualquiera de estos métodos u otros que se usen, el sistema CAD produce en la etapa de diseño una simple base de datos geométrica la cual puede usarse en todas las fases del diseño y mas tarde en los procesos de fabricación, ensamblaje e inspección.

Si nos ceñimos al campo de la Ingenieria de Telecomunicación y en general de la Electrónica analógica y digital, los ejemplos de aplicación del CAD son innumerables:

1. Simulación de circuitos integrados para su diseño en continua, régimen transitorio ó en el dominio de la frecuencia.
2. Diseño automático (interactivo) de placas de circuitos impresos.
3. Ayuda al diseño de máscaras de integración: L.S.I. y V.L.S.I. sobre todo.
4. Diseño de filtros activos ó pasivos.
5. Diseño de filtros digitales.
6. Simulación de circuitos digitales: señales y diagramas de tiempo.

7. Simulación y estudio estadístico del comportamiento de circuitos en función de las condiciones ambientales: humedad, temperatura, envejecimiento y de tolerancias tecnológicas.

La ayuda del computador en la fabricación (CAM) significa el uso de un computador en la fabricación de una parte. CAM puede dividirse en dos clases principales: (1) aplicaciones ON-LINE, esto es, el uso del computador para controlar sistemas de fabricación en tiempo real. (2) aplicaciones OFF-LINE, es decir, el uso del computador en la producción planificada y asistencia en tiempo no real en la fabricación de partes. Ejemplos de CAM en OFF-LINE son la preparación de parte de programas sobre tarjetas perforadas usando el lenguaje APT(herramientas programadas automaticamente).

El APT es un lenguaje de programación de uso especifico,uno de los más comprensibles y el más conocido, el cual facilita en parte el trabajo del programador. Consta de una serie de sentencias las cuales son perforadas en tarjetas y se usan como entrada al computador. Concretamente el APT consta de 300 palabras.

Otros sistemas de programación conocidos son el ADAPT, SPLIT, EXAPT, AUTOSPOT, COMPACT II, y muchos otros que han sido desarro-

llados por varias compañias.

CAD/CAM es un sistema software, en el cual la parte de CAD esta interconectado en el interior del computador con el sistema CAM. El concepto principal de los sistemas CAD/CAM es la producción de una base de datos común la cual puede usarse para todas las actividades de diseño y fabricación. Estos incluyen especificaciones del producto, diseño conceptual, diseño final, dibujo, fabricación e inspección. En cada etapa de este proceso, los datos pueden añadirse, modificarse, usarse y distribuirse sobre las redes de terminales y computadores. La simple base de datos produce una reducción substancial en los errores humanos y una reducción significativa del tiempo requerido desde la introducción de un concepto de un producto a la fabricación del producto físico final.

El tamaño y capacidad del sistema computador que se requiere depende de la complejidad del producto. En la industria aeroespacial y aeronaútica, donde un avión completo puede diseñarse con un procesador CAD/CAM, el sistema debe acomodar y cambiar datos desde una diversidad de usuarios. En consecuencia estos sistemas deben tener unos datos fuerte con capacidad de manejo.

Por contraste, si los productos simples son diseñados por una compañia, el sistema CAD/CAM requerido necesitaria solo un terminal de computador. Hoy los mejores usuarios de sistemas CAD/CAM son las industrias aeroespaciales y automovilisticas, pero la baja de precios de estos sistemas han aumentado el número de otros usuarios.

Avanzados sistemas CAD/CAM incluyen modelos geométricos sólidos, en adicción al modelo de diagrama formado de lineas.

Durante estos años la tecnologia CAD/CAM ha perfeccionado la productividad industrial. Ello es un paso significativo hacia el diseño de la fabrica del futuro.

Las gráficas con computador es el uso de un computador para definir, almacenar, manipular, interrogar y presentar una salida gráfica. Esto es esencialmente una operación pasiva. El computador prepara y presenta la información almacenada a un observador en forma de gráfica, sin un control directo sobre la misma. Un ejemplo de aplicación de las gráficas con computador puede ser tan simple como la representación de una función usando una impresora de linea de alta velocidad o un terminal teletipo de tiempo compartido,o tan compleja como la simulación de la reen-

trada automática y el aterrizaje de una nave espacial.

Las gráficas interactivas también usa el computador para preparar y presentar el material gráfico. Sin embargo, en las gráficas interactivas el observador puede influir sobre la gráfica según se esté presentando, todo esto en tiempo real. También suelen usar sistemas CAD convirtiendose en una herramienta valiosa.

1.1 DESCRIPCION GENERAL DE LAS GRAFICAS CON COMPUTADOR

Las gráficas con computador puede ser un tema muy complejo y diverso. Abarca campos de estudio tales como electrónica, diseño mecánico de componentes utilizados en sistemas de gráficas con computador y conceptos de presentación de listados y estructuras arborescentes para la preparación y presentación de gráficas. Desde el punto de vista del usuario las gráficas con computador puede ser divididas en las siguientes áreas:

- Representación de las cuadros a presentar
- Preparación de dichos cuadros a presentar
- Presentación previamente preparada de los cuadros
- Interacción con los cuadros

Aqui la palabra "cuadro" abarca cualquier colección de lineas, puntos, textos, etc., a presentar en un dispositivo gráfico. Una gráfica puede ser tan simple como una linea o curva, o la representación compleja de un avión, barco o automóvil.

1.1.1 REPRESENTACION DE LOS CUADROS A PRESENTAR

Fundamentalmente las gráficas representadas con computador pueden considerarse como una colección de lineas, puntos y

textos. Una linea se representa por las coordenadas de sus puntos inicial y final (X_1, Y_1, Z_1) y (X_2, Y_2, Z_2), un punto por la terna de coordenadas (X_1, Y_1, Z_1), y los textos por colección de lineas y puntos.

La representación de textos es más complejo puesto que usa curvas o matrices de puntos. Sin embargo, al menos que el usuario este interesado en el reconocimiento de formas, el diseño hardware de las gráficas o el grupo de caracteres, no necesita conocer estos detalles, puesto que casi todos los dispositivos gráficos tienen ya su "hardware" interno o software generador de caracteres. La representación de curvas se realiza por sucesivas aproximaciones a pequeñas rectas. Hoy dia, esto esta solucionado usando generadores hardware de curvas.

1.1.2 PREPARACION DE LOS CUADROS A PRESENTAR

Ultimamente las gráficas se representan con puntos. Las coordenadas de estos puntos se almacenan en un fichero (matriz) antes de su uso. Este fichero se llama base de datos. Las gráficas complejas requieren base de datos complejas y a su vez un programa complejo para acceder a ellos. Estas bases de datos contienen puntos, subestructuras y otros datos no gráficos. El diseño de

las base de datos y los programas para acceder a ellas esta en continua investigación.

Los puntos son los bloques fundamentales para hacer la base de datos de una gráfica. Hay tres métodos básicos o instrucciones para tratar un punto como una entidad geométrica gráfica: mover el tablero, pluma, cursor, plotear la cabeza al punto, y dibujar una linea a ese punto, o dibujar de un punto a otro punto. En general hay dos formas de especificar la posición de un punto: en coordenadas absolutas o relativas (incremental). En coordenadas relativas o incremental la posición del punto se define dando el desplazamiento del punto respecto al punto anterior.

La especificación de la posición del punto ya sea en coordenadas relativas o absolutas requieren un número. Esto puede acarrear dificultad si el computador que se usa tiene una longitud de palabra limitada. Generalmente una palabra completa del computador se usa para especificar la posición de una coordenada. El número entero más grande que se especifica en una palabra completa del computador es 2^n-1, donde n es el número de bits en la palabra. Para un minicomputador de 16 bits, este es 32767. El cuál es aceptable para muchas aplicaciones.

Sin embargo, cuando se necesitan números enteros mayores que el

especificado se encuentran dificultades. En principio, se podria pensar superar esta dificultad usando coordenadas relativas, por ejemplo, para un número como 60.000 usando una coordenada absoluta especifica para posicionar el cursor a (30.000, 30.000) y posteriormente una coordenada relativa de (30.000, 30.000) para colocar el tablero en el punto final deseado de (60.000,60.000). Pero esto tiene un problema, ya que desde el intento de acumular la posición relativa especificadas atrás al máximo valor representado resultante seria un número de signo contrario y magnitud errónea. Para las pantallas de tubo de rayos catódicos (CRT) esto produciria el fenómeno llamado reiniciación cíclica.

La forma de salir de este dilema es usar coordenadas homogeneas. El uso de coordenadas homogeneas introduce algunas complicaciones adicionales, tales como pérdida de velocidad y resolución. No obstante, estas desventajas pesan menos que las ventajas para poder representar números enteros grandes con un computador de tamaño limitado.

En coordenadas homogeneas un espacio de n dimensiones se representa por n+1 dimensiones, por ejemplo, un dato de tres-dimensiones donde la posición de un punto se da por la tripleta

(X,Y,Z) se representa por cuatro coordenadas (hx,hy,hz,h), donde h es un número arbitrario.

Si cada una de las posiciones de las coordenadas representadas en un computador de 16 bit fuesen menor que 32767, entonces h seria igual a 1 y las coordenadas representadas directamente. Si una de las coordenadas en mayor de 32767, es decir, x=60000, entonces el poder de las coordenadas homogeneas seria aparente. En este caso podemos hacer h=1/2, y las coordenadas del punto se definen como (30000,1/2y,1/2z,1/2), todos números aceptables para un computador de 16 bit. Sin embargo, hay una pérdida de resolución puesto que x=60000 y x=59999 son representadas por la misma coordenada homogenea. De hecho la resolución es menor en todas las coordenadas si solo una de ellas excede del número máximo permitido de un computador particular.

1.1.3 PRESENTACION PREVIAMENTE PREPARADA DE LAS GRAFICAS

Las bases de datos comentadas anteriormente usadas pare preparar la gráfica para la presentación caso nunca son las mismas que los ficheros de visualización usados para presentar la gráfica. La base de datos representa la gráfica total mientras el fichero display (de visualización) representa solo

una parte, visión o escena. El fichero display se creó para transformar la base de datos. La gráfica contenida en la base de datos debe rotarse, reajustarse, transladarse o ser visualizada desde un punto particular para obtener la perspectiva necesaria antes de representarla. Muchas de estas operaciones se realizan con simples transformaciones lineales lo cuál implica multiplicaciones de matrices, en medio de las cuales se producen las rotaciones, translaciones, etc.

Una matriz 4x4 puede usarse para realizar cualquiera de estas transformaciones individuales de puntos, representados en una matriz en coordenadas homogeneas. Cuando se desea una secuencia de transformaciones, cada transformación individual se aplica secuencialmente a los puntos para archivar el resultado deseado. Sin embargo, si el número de puntos es substancial, esto es ineficaz y hay pérdida de tiempo. Una alternativa y un método más eficaz es multiplicar las matrices individuales representando cada transformación requerida a la vez, y al final se multiplica la matriz de puntos por la matriz de transformación 4x4 resultante. Esta operación con matrices se llama concatenación, e implica un ahorro de tiempo significativo.

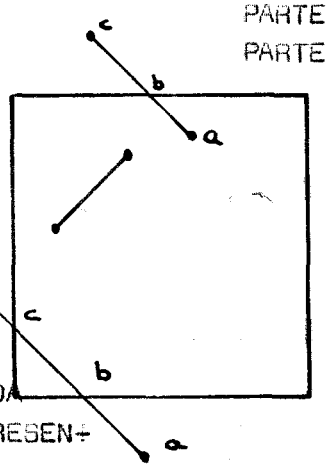
Aunque en muchas aplicaciones, la base de datos completa de

una gráfica se representa en pantalla, hay ocasiones en las que solo una parte de la misma se representa. Este proceso se llama "windowing" (ventana). La ventana no es fácil, particularmente si la base de datos de la gráfica ha sido transformada anteriormente. La operación ventana en software genera un tiempo muy necesario para las gráficas interactivas dinámicas, que en tiempo real no es posible. A veces, dispositivos gráficos sofisticados realizan esta función en hardware. En general hay dos métodos de realizar este proceso ventana: recuadro y recorte. El proceso de recuadro implica términos como lineas o porciones de lineas en la gráfica. Dichas lineas o porciones de lineas a la hora de la visualización son desechadas. En el método de recorte, el dispositivo de visualización tiene un espacio físico para el dibujo mayor que el requerido. Solo las lineas dentro de la ventana son visibles aunque las externas son dibujadas. Esto, por supuesto, requiere tiempo, al tener que dibujar la base de datos entera sin tener en cuenta si las lineas son visibles o no, mientras que en el caso del recuadro solo se dibuja parte de la base de datos.

En dos dimensiones una ventana se especifica por los valores de los lados de un rectángulo. En el recuadro es más fácil si

LINEA PARCIALMENTE DENTRO DE LA VENTANA:
PARTE DE a-b PRESENTADA
PARTE DE b-c NO PRESENTADA

LINEA ENTERAMENTE
DENTRO DE LA VENTANA
LINEA PRESENTADA



LINEA ENTERAMENTE
FUERA DE LA VENTANA:
NO PRESENTADA

LINEA PARCIALMENTE
DENTRO DE LA VENTANA:
PARTE DE b-c PRESENTADA
PARTE DE a-b,c-d NO PRESEN-
TADA

los lados del rectángulo son paralelos alos ejes de las coordenadas. Sin embargo, si este no es el caso, la rotación de la ventana puede compensarse con la retación de la base de datos en la dirección opuesta. Dicho método en dos dimensiones se representa en la figura siguiente:

Las lineas son retenidas, borradas, o parcialmente borradas, dependiendo si están completamente o parcialmente dentro o fuera de la ventana. En tres dimensiones la ventana tiene la visión del tronco de una pirámide.

Como un paso final en el proceso de presentación de la gráfica, es necesario convertir las coordenadas usadas en la base de datos llamadas coordenadas de usuario, en otras usadas por el dispositivo de visualización, llamadas coordenadas de display o

visualización. En particular, es necesario convertir las coordenadas de datos que pasan el proceso "windowing" a coordenadas de visualización, de forma que la gráfica aparezca en algun área especifica de la pantalla, llamada "área de visión". El área de visión se especifica dando sus lados, su dimensión o indicando el límite más cercano y lejano.

Un requisito adicional para la mayor parte de las gráficas es la presentación de datos alfanuméricos o letras (caracteres). Hay en general dos métodos de generación de caracteres: software y hardware. Si los caracteres son generados en software usando lineas, son tratados como cualquier elemento de la gráfica. Sin embargo, muchos dispositivos gráficos tienen alguna clase de generador de caracteres hardware. Cuando se usan los generadores de caracteres hardware, los caracteres son tratados como códigos de caracteres antes de dibujarse, y generados justo antes de comenzar la gráfica. El generador de caracteres hardware es menos flexible puesto que no permite el recorte o transformaciones infinitas, solo en tamaños y rotaciones limitadas, aunque producen rendimientos significativos.

Cuando se usan dichos generadores hardware, el programa que actua sobre el dispositivo gráfico debe especificar tamaño, orienta-

ción y la posición dende el caracter o cadena de caracteres van a empezar. Los codigos de caracteres especificando dichas caracteristicas son añadidas al fichero de visualización. Al tiempo de procesarlos el generador de caracteres los interpreta, mirando en hardware las lineas necesarias para dibujarlos y representarlos en la pantalla del dispositivo.

1.1.4 INTERACCION CON LAS GRAFICAS

La interacción con las gráficas requiere algún tipo de dispositivo de interacción para comunicarse con el programa mientras se ejecuta. Por medio de estas interrupciones del programa se puede proporcionar nuevas y diferentes informaciones. Numerosos dispositivos se utilizan para ejecutar esta tarea. El más simple es el teclado alfanumérico, como el de un teletipo. Otros dispositivos más sofisticados son los lapices luminosos, un ratón, joy sticks, track balls, interruptores de función, discos de control y tableros analógicos. Pasemos a describir por encima cada uno de estos dispositivos.

El teclado alfanumérico permite suministrar al programa información alfabética, numérica o de control. Sin embargo, no esta capacitado para alta velocidad de interacción.

Quizás el mejor dispositivo de interacción conocido es el lapiz luminoso. El lápiz luminoso consta de una célula fotoeléctrica sensible a la luz y circuiteria asociada. Cuando se posiciona sobre un segmento de linea u otra área iluminada de la CRT y se activa, la posición del lápiz es detectada y una interrupción se envia al computador.

El joy stick, ratón y track ball operan sobre el mismo principio. Moviendo el control, la información posicional en dos dimensiones se comunica al computador. Todos estos dispositivos son de naturaleza analógica. El proceso es el siguiente, un movimiento del control cambia la posición de un potenciómetro, las señales obtenidas se convierten a digital usando un convertidor analógico/digital. Estas señales digitales son entonces interpretadas por el computador como información posicional.

Los discos de control esencialmente son potenciómetros rotativos sensibles y circuiteria asociada, de tal manera que la posición del disco se capta usando técnicas de conversión analógica/digital. Son particularmente útiles para actividades de rotación, translación y sistemas software.

Los interruptores de función son de tipo palanca o pulsadores en los cuales la posición puede determinarse por el progra-

ma de las gráficas. También se suele utilezar indicadores luminosos activados por interruptore.

El tablero analógico es el dispositivo más versátil y preciso para comunicar la información posicional al computador. Propiamente usado, el tablero analógico puede realizar las funciones de un lápiz luminoso, joy stick, track ball, ratón, interruptores de función o discos de control. Asociado con el tablero hay un lápiz que se mueve sobre la superficie a la cuál es sensible. La posición del lápiz y su localización relativa en el área de visión de la gráfica estan relacionados por un cursor, de forma que el movimiento del cursor en el área de visión de la gráfica esta ligado con el del lápiz en el tablero. El tablero analógico tiene dos ventajas sobre el lápiz luminoso. Normalmente, cuando el tablero analógico se usa para ejecutar una función indicadora (comando), la indicación tiene lugar en la base de datos y no en el fichero de visualización. Asi, el programa se simplifica. También, dibujar o diseñar sobre un tablero analógico en posición horizontal es más natural que realizar las mismas funciones con lápiz luminoso en posición vertical.
Un tablero analógico puede implementarse en hardware usando una variedad de principios electromagnéticos.

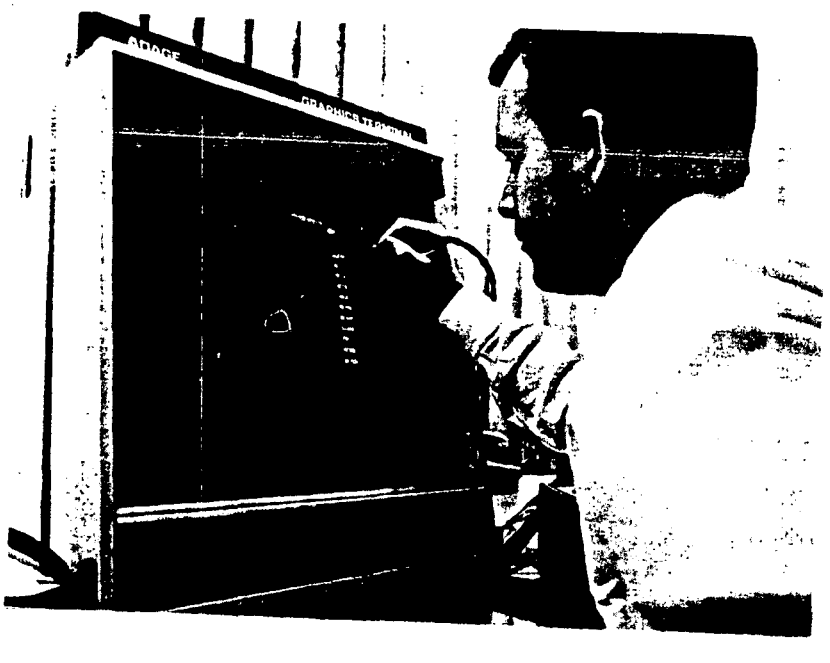
DISPOSITIVOS DE INTERACCION

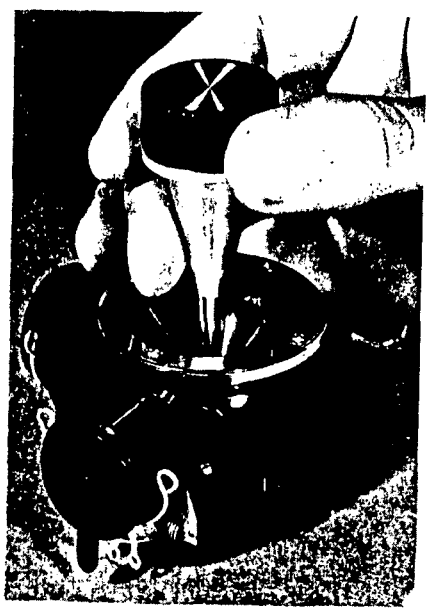
TECLADO

ALFANUMERICO

LAPIZ

LUMINOSO

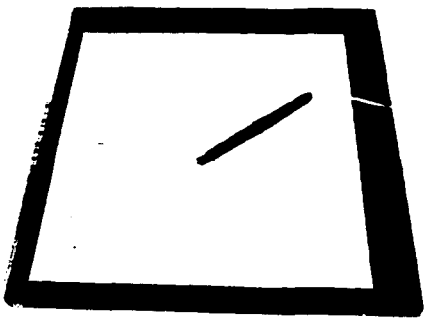


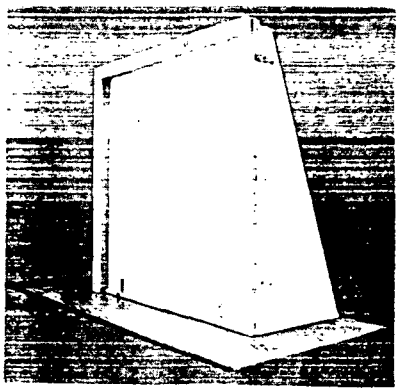


JOY

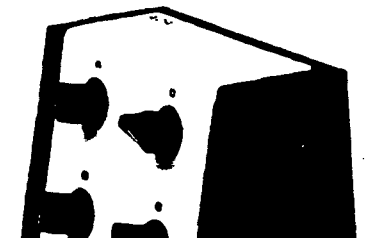
STICK

TABLERO





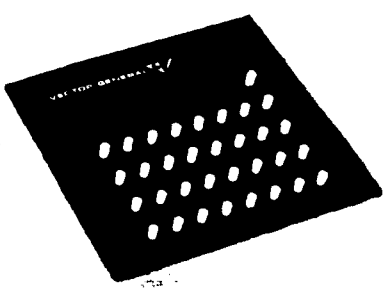
ANALOGICO



DISCOS
DE
CONTROL

TRACK BALL

INTERRUPTORES
DE
FUNCION



1.2 DESCRIPCION DE ALGUNOS TIPOS DE DISPOSITIVOS GRAFICOS

Hay un gran número de dispositivos gráficos disponibles. Pero solo voy a referirme a tres en concreto, muy caracteristicos: CRT (tubo de rayos catódicos)- con tres tipos de pantallas; memorizadora, refresco,rastreo - plotter con pluma y tinta y plotter de matriz de puntos.

La visualización en una pantalla memorizadora, también llamada pantalla memorizadora biestable puede considerarse similar a la de un osciloscopio con una fosforecencia persistente. Una linea o caracter permanecerá visible hasta que la generación de una señal electrica especifica la borre. El proceso de borrado requiere alrededor de medio segundo. Las pantallas memorizadoras tienen aspectos favorables y desfavorables. Algunas de las ventajas son: la pantalla es de parpadeo libre, resolución buena y bajo coste. Además, es relativamente fácil y rápido obtener una salida impresa automática de la gráfica y más facilidad para programar y aplicar tiempo-compartido, que en las pantallas de refresco o rastreo. La principal desventaja es que la pantalla no puede borrarse selectivamente, si se desea cambiar cualquier elemento de la gráfica hay que volver a dibujarla. En este tipo

de pantalla el movimiento dinámico no es posible. Por lo que, la interacción entre el usuario y la pantalla es más lenta que en una pantalla refrigerada.

Un CRT con pantalla refrigerada se basa como la televisión en un tubo de rayos catódicos. La televisión utiliza la técnica de rastreo para generar la imagen , mientras que el tradicional CRT con pantalla refrigerada es del tipo caligráfico o traza de linea. Dicho tubo de rayos catódicos requieren junto con el CRT: un buffer de pantalla y un controlador de pantalla.

El fósforo usado en el CRT se desvanece rapidamente, es decir, tiene corta persistencia, es necesario reponer o reconstruir la imagen muchas veces cada segundo. Esto se llama velocidad de refresco o renovación. Una velocidad de refresco demasiado baja produce un fenómeno llamado parpadeo. Similar al efecto que se produciria en un proyector al pasar una pelicula demasiado despacio. La función del buffer de pantalla es almacenar en secuencias todas las instrucciones necesarias para dibujar la imagen en el CRT. La función del controlador de pantalla es acceder a estas instrucciones a la velocidad de refresco. Inmediatamente, una limitación de la pantalla de refresco es obvia: la complejidad de la imagen esta limitada por el tamaño del buffer de la

pantalla y por la velocidad del contador. Sin embargo, la corta persistencia de la imagen se puede considerar como una ventaja para mostrar el movimiento dinámico. Además, puesto que cada elemento o instrucción necesaria para dibujar la imagen completa existe en el buffer de pantalla, cualquier elemento individual puede cambiarse, borrarse o añadirse, lo cual puede considerarse como caracteristica de borrado selectivo.

Aunque los CRT con pantalla refrigerada son generalmente más caros que los de pantalla memorizadora, las anteriores caracteristicas hacen que se prefieran cuando se requiere movimiento dinámico en tiempo real o mayor rapidez de interacción con la pantalla.

Un CRT con pantalla de rastreo usa un monitor de televisión estandar para la consola de pantalla. En las pantallas de rastreo la imagen esta compuesta de una serie de puntos. Estos puntos son seguidos según una tecnica de rastreo, como por ejemplo, una serie de lineas horizontales. La señal eléctrica básica usada para conducir la consola de pantalla es una señal analógica cuya modulación representa la intensidad de los puntos individuales los cuales forman la imagen. La utilización en la consola de la pantalla del rastreo es necesario para convertir la información de

lineas o caracteres a una forma compatible con la traza. Este proceso se llama conversión de exploración. Una vez se convierte la información debe almacenarse de forma que se pueda acceder a ella razonablemente. Con los avances en técnicas de almacenamiento de datos, esto es cada vez más factible.

Los plotters de lápiz y tinta de incrementación digital son en general de dos tipos: tablero y tambor magnético. Suelen proporcional una salida gráfica altamente cualificada. Comparándolos con los dispositivos gráficos de CRT son bastante lentos. Consecuentemente no son generalmente usados para gráficas con interacción en tiempo real. Sin embargo, donde se requieren dibujos grandes para aplicación particular, un plotter de tablero se puede utilizar como una combinación de digitalizador, plotter y un sistema de interacción gráfica-computador.

El plotter/impresora de matriz de puntos electrostático es un dispositivo de rastreo, presentando información en una linea de una vez. Al ser un dispositivo de rastreo requiere una cantidad sustancial de información almacenada en el computador para dibujar una gráfica completa. La principal desventaja es que solo se puede usar para gráficas pasivas, también es relativamente lento en resolución. Las ventajas son su alta velocidad para dibujar y una

excelente fiabilidad.

Para una mayor información sobre los plotters y sus tipos, en este proyecto hay una parte destinada a los llamados PLOTTERS GRAFICOS.

1.3 CLASIFICACION DE LOS DISPOSITIVOS GRAFICOS

Hay un gran número de métodos de clasificación de los dispositivos gráficos con computador.

En primer lugar consideremos la diferencia entre dispositivos gráficos activos y pasivos. Un dispositivo gráfico pasivo simplemente dibuja gráficas bajo el control del computador, son poder actuar sobre ellas. Algunos ejemplos son: teletipos, impresoras de linea de alta velocidad, impresoras electrostáticas de matriz de puntos, plotters de pluma y tinta, CRT memorizadores y de refresco.

Un dispositivo gráfico activo permite al usuario comunicarse con el computador graficamente. Generalmente esto implica que el usuario esta suministrando información de coordenadas de datos de alguna manera indirecta. Los dispositivos gráficos activos también tienen la capacidad de reposición de cursor y leer su nueva posición. Dichos dispositivos incluyen simples cursores de botón, digitalizadores o tableros analógicos, lapices luminosos, joy sticks, trackball o ratón. Aunque digitalizados usualmente requieren algún tipo de dispositivo gráfico pasivo para apoyarse. Este dispositivo de soporte suele ser un CRT.

Otro método de clasificación de los dispositivos gráficos es si dibujan a base de puntos o de lineas(vectores). La diferencia fundamental es si un generador de vectores hardware es aprovechable. Un generador de vectores hardware permite el dibujo de lineas con una mínima contidad de datos. Esto, por supuesto, no significa que un dispositivo que dibuje con puntos no pueda dibujar vectores usando software. Un vector puede dibujarse con una serie de puntos. Si los puntos son trazados lo suficientemente juntos, apareceran a los ojos como una linea continua. Todos los dispositivos gráficos de CRT memorizadores y la mayor parte de los refrigerados, asi como los plotters de pluma son dispositivos trazadores de lineas. Algunos CRT refrigerados, particularmente los dispositivos de frecuencia de análisis (como televisión), teletipos, impresoras de lineas de alta velocidad, e impresoras electrostáticas de matrices de puntos son clasificadas como dispositivos trazadores de puntos. La utilidad de un dispositivo puede frecuentemente en términos de su resolución.

Todavia otro método de clasificación es determinar si un dispositivo puede aceptar un dato en tres-dimensiones o si el dato debe ser primero convertido en dato de dos-dimensiones por aplicación

de alguna proyección de transformación y presentada como dato de dos-dimensiones. En esencia este método requiere determinar si un dispositivo gráfico tiene dos o tres registros para retener los datos de las coordenadas. En caso de un dispositivo de tres dimensiones el tercero o coordenada Z se usa generalmente para controlar la intensidad del haz de un CRT. Esta caracteristica se llama modulación en intensidad o graduación. Se utiliza para dar la sensación de profundidad a una gráfica.

PLOTTERS

PLOTTERS

Los plotters producen una salida impresa continua del computador consistente en lineas y curvas. Estas lineas y curvas se combinan para formar dibujos, gráficos, diagramas y toda forma de representación gráfica.

Los datos para el trazado de la grádica se almacenan o en la memoria del computador o en un disco o cinta magnética. Estos datos son llamados mediante un programa; los comandos de trazado pasan desde el programa a un controlador de plotter que manda las diferentes señales que mueven los motores de conducción del lápiz (para un plotter XY).

Existen diferentes tipos de plotter y pueden clasificarse en cuatro formas:

1. Método de trabajo y exploración
2. Si el papel es estático o móvil
3. Tamaño del plotter
4. Si el sistema gráfico esta dedicado a un uso particular o no.

2.1 METODOS DE TRABAJO Y EXPLORACION

El método de trabajo más usado es el lápiz, lápiz principalmente de tinta, lápiz con punta de fieltro o con punta de naylor. Los plotters que usan este método son llamados PLOTTERS DE LAPIZ. Los plotters de lápiz incorporan uno o mas motores que llevan los lapices en pequeños pasos en linea recta bajo el control del programa del plotter. La combinación de pequeños pasos de X e Y permite dibujar curvas con fina resolución asi como lineas en cualquier dirección. Los plotters de lápiz movidos en pequeños pasos son también llamados PLOTTERS DE INCREMENTACION. Porque pueden trazar lineas o curvas continuamente en dos direcciones, los plotters de lápiz de este tipo son también conocidos como PLOTTERS XY y PLOTTERS VECTOR.

El plotter vector de un único lápiz trabaja con un solo color de una vez. Si se desea colores adicionales, la secuencia del programa de la gráfica se divide en secciones de color. Es parado en cada sección para realizar la operación de cambiar el lápiz de otro color.

Los PLOTTERS DE MULTIPLES LAPICES trazan con varios colores sin interrumpir el programa o lineas gruesas sin tener que volver varias veces sobre ella.

Aunque los plotters vector dibujan lineas en cualquier dirección con una cabeza de escritura (que no tiene que ser el lápiz), los PLOTTERS DE RASTREO primero procesan la imagen, dividiendola en filas. De esta forma, una cabeza impresora electrostática traza cada fila de puntos, una fila cada vez. El método de impresión de los plotters electrostático se basa en el depósito de partículas de un tono más oscuro, cargando áreas de un papel especial. En detalle, un papel especialmente protegido reteniendo una carga electrostática se pasa sobre una cabeza de escritura que contiene una fila de pequeñas puntas de escritura o estiletes. Los estiletes depositan cargas electrostáticas sobre el papel especial. Desde las cargas electrostáticas no visibles, el papel cargado se pasa sobre un toner el cual es un líquido conteniendo particulas de un tono más oscuro. Las particulas son atraidas al área cargada electrostáticamente haciendose visibles. El papel es entonces secado y presentado al usuario. Estos plotters son capaces de marcar puntos individuales de una densidad de 100 a 200 puntos por pulgada, por lo que trazan textos y gráficos con igual facilidad. Gracias a esta capacidad, los plotters electrostáticos son frecuentamente considerados como PLOTTERS IMPRESORA.

2.2 MOVIMIENTO DEL PAPEL

Un criterio frecuentemente usado para clasificar los plotters es si el papel es móvil o no. Los plotters con papel estático son plotters de tablero y los plotters con papel móvil son plotters de tambor magnético o plotters de rastreo.

2.2.1 PLOTTERS DE TABLERO

Estos plotters son como tableros de dibujo. Una simple lámina de papel u otro material de dibujo se coloca sobre una superficie plana y la cabeza de escritura se mueve sobre ella en dos dimensiones. Ambas dimensiones se fijan por los posibles rangos de movimientos del lápiz y por el tamaño del tablero. Como el papel es plano y el lápiz es capaz de moverse sobre toda la superficie de dibujo, cualquier sección puede dibujarse en cualquier momento durante la operación de trazado de la gráfica. No hay necesidad de empezar y terminal al final de la traza para luego seguir con la otra, como ocurre con plotters con papel móvil.
Los lápices usados para dibujar gráficas en plotters de tablero tipicamente se mueve sobre la superficie de dibujo en un carril corredizo en pistas paralelas externas sobre la parte mas estrecha de las dos dimensiones, generalmente la horizontal.

La combinación de ambos movimientos da lugar en el papel al movimiento según el eje X e Y.

Ambos movimientos el del carril y el portalápiz se hacen en pequeños pasos generados por un motor diseñado para originar los movimientos del lápiz según ambos ejes. Por último los comandos para trazar gráficas dibujadas por el computador desde datos almacenados en memoria son transladados al lápiz.

2.2.2 PLOTTERS CON PAPEL MOVIL

El papel para los plotters de tambor magnético se obtiene desde una pila de papel o desde un rollo de papel. El cual se desenrrolla continuamente y una sección de la traza se dibuja.

Con los plotters de tambor magnético de lápiz, el lápiz se mueve en una sola dirección (eje X) a través de la dirección de movi - miento del papel (la dirección Y). La combinación de los movimientos del lápiz y del papel producen las lineas y curvas deseadas, formando la imagen completa. Tipicamente, el papel se mueve hacia adelante y hacia atrás como requisito para dibujar una curva o linea continuamente. Para impedir excesiva pérdida de tiempo en el movimiento del papel, es conveniente completar una sección de una traza larga antes de proceder a la siguiente.

Los plotters electrostáticos de rastreo son también plotters de papel móvil. Se diferencia de los plotters de tambor magnético en que ese movimiento del papel es continuo en una dirección. Para todos los plotters con papel móvil, el ancho de la traza esta limitada por el ancho del tambor magnético o por la cabeza de escritura. La longitud de la superficie de dibujo es virtualmente inlimitado.

2.3 PLOTTERS SOBRE SUPERFICIE Y ALZADOS

El área de dibujo de un plotter de tablero se da especificamente por la altura y ancho, y para los plotters de tambor magnético por el ancho. Como complemento a estas medidas especificadas, los plotters de ambos tipos son catalogados como plotters sobre superficie, usados para documentos y pequeños dibujos, y modelos alzados o de pie, usados para ingenieria y dibujos cientificos, mapas y otras ilustraciones.

2.4 RANGO DE LAS CAPACIDADES FISICAS

Aparte de las caracteristicas anteriormente dichas, también existen otras que determinan las capacidades fisicas de los plotters:

1. Plotters de lápiz con incrementación

   a. Tamaño de los pasos

   b. Longitud y ancho de la traza

   c. Velocidad de operación de los pasos por segundo

2. Plotters electrostático de rastreo

   a. Densidad del punto

   b. Velocidad de impresión en el papel en pulgadas por minuto.

Para los plotters de lápiz con incrementación el rango de los tamaños de los pasos van de 0.1 a 0.0001 para los plotters de más alta precisión. La resolución de la linea frente al tamaño del paso es mas importante cuando la linea esta un poco inclinada en horizontal o vertical. A menor tamaño del paso mejor resolución de la linea, pero adicionalmente se requiere mas tiempo para completar una linea debido a la aceleración y deceleración de la cabeza de escritura. Esto se compensa con una mayor velocidad en los pasos.

2.5 PLOTTERS DE USO ESPECIFICO: SISTEMAS DE DISEÑO AUTOMATICO

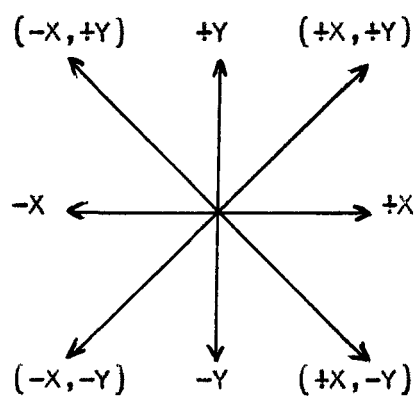
Estos sistemas hacen dibujos de ingenieria, trazado de

terrenos, diagramas de circuitos y placas de circuitos impresos, entre otros. Son frecuentemente usados en unión de digitalizadores de imagen para este propósito. Los plotters con un uso específico debe ser diferenciados de los que son usados como periféricos. Como periférico se considera como el dispositivo principal de un sistema de computador de propósito especial, los plotters como sistemas de diseño automático no son periféricos a una CPU pero son el punto central de un sistema de propósito especial.

2.6 LOS PROGRAMAS DE LOS PLOTTERS

Los plotters con programas de computador trazan curvas de negocio, visiones de objetos en tres dimensiones, representaciones de simbolos de fenómenos físicos y textos.

En la forma más simple, las instrucciones para un plotter de tablero de lápiz produce un movimiento incremental de lápiz en cualquiera de las ocho direcciones.



Los comandos típicos para el movimiento del lápiz son codificados en ASCII para su posterior representación en el plotter. Con dichos comandos cualquier gráfica puede dibujarse.

Para microcomputadores programados en BASIC, FORTRAN, u otros lenguajes comunes, con simples subrutinas el computador genera los comandos necesarios para trazar la mejor recta entre cualquiera dos puntos.

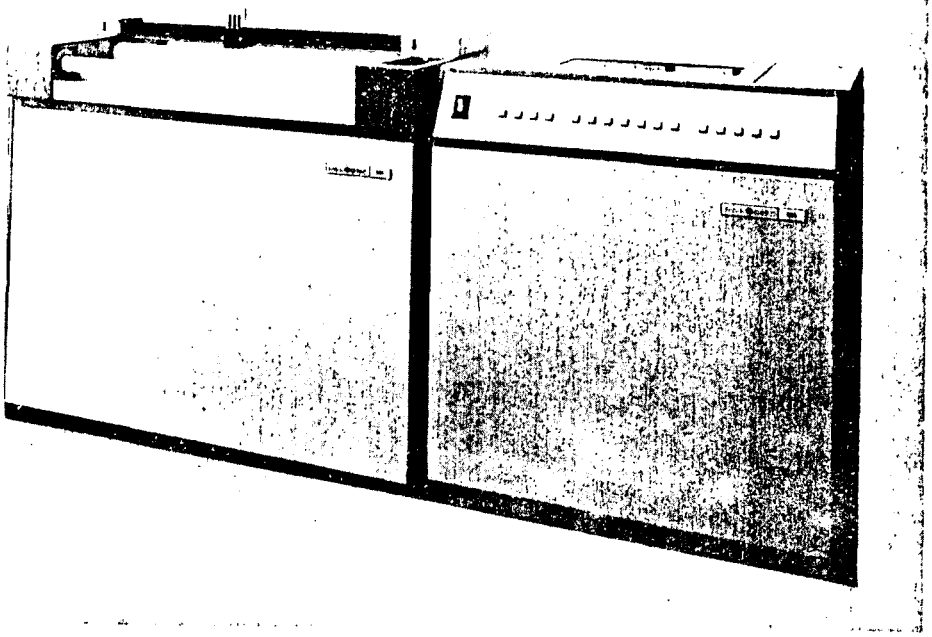
Usar cualquier tipo de instrucciones detalladas para extenderla al trazado comercial y cientifico es por supuesto un tiempo consumido y similar a escribir un programa de computador en máquina o lenguaje ensamblador. Las subrutinas escritas en lenguajes de alto nivel habilita funciones controladoras de plotters; usan simples generadores de lineas y subrutinas que ejecutan repetidas funciones para eliminar comandos innecesarios entre pasos.

Ciertamente las funciones de plotter son independiente de las aplicaciones comerciales, diseño, o cientifica - funciones como dibujar circulos, rectangulos y lineas. Además las subrutinas para dibujar lineas, curvas, y formas, generadores de caracteres en mayúsculas, minúsculas proporcionan textos para etiquetas de diagramas. Algunos plotters proporcionan varios grupos de caracteres de tamaños variables, bifurcación y dirección.

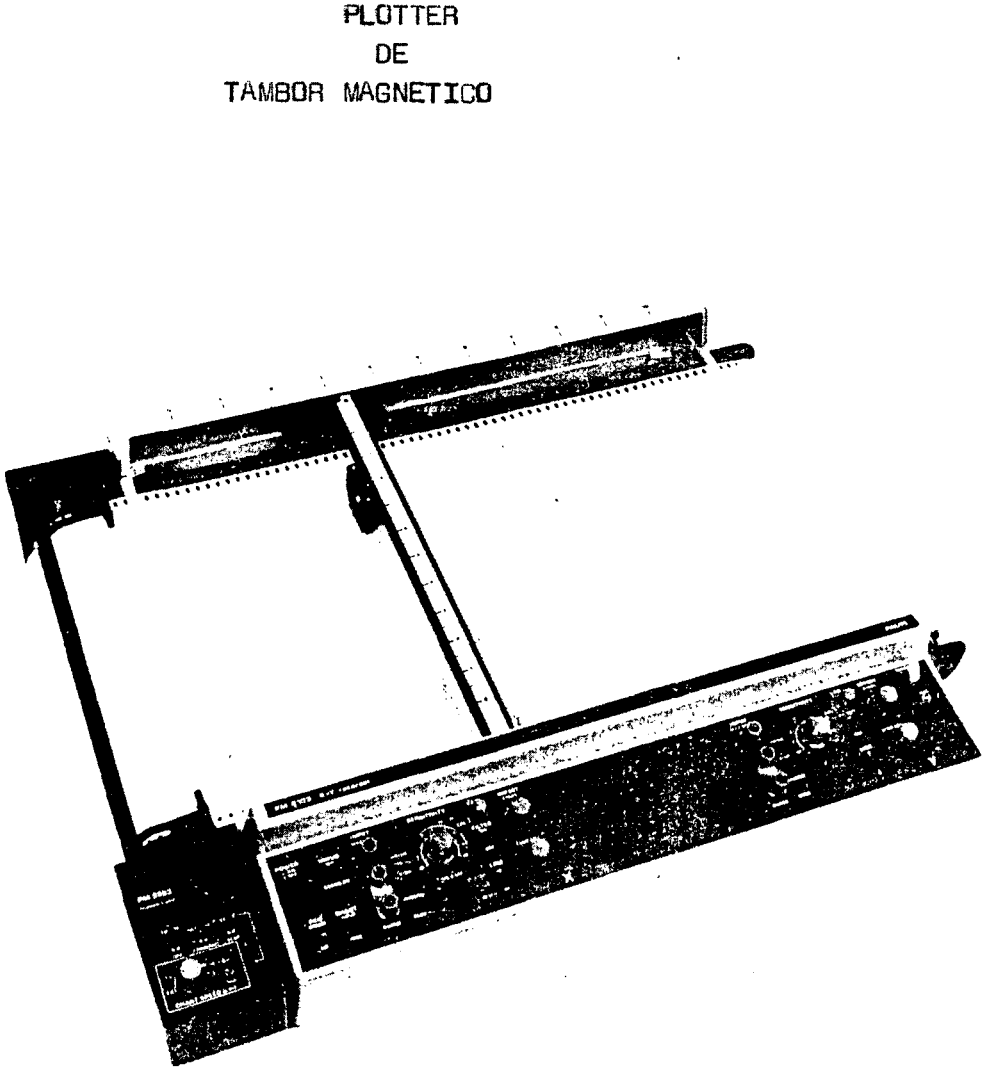
El uso de la tecnologia de los plotters y estas subrutinas tipicas hacen posible obtener una salida gráfica impresa desde un mini- o microcomputador para propósitos cientificos, ingenieria y varidad de negocios, usando las llamadas subrutinas del plotter.

Un mini- o microcomputador se necesita para transladar estos programas al plotter en comandos de lenguaje máquina. Si el plotter es de rastreo, las imagenes constan de lineas y curvas y deben ser también convertidas por el computador en un formato propio para imprimir. Aún con estos requisitos, el computador usualmente es de 100 a 1000 veces mas rápido en el procesamiento de las instrucciones que el plotter en trazar.

Cuando un minicomputador de alta-capacidad se usa para otros propósitos que hacer programas de gráficas, para que no suponga una pérdida de tiempo, una solución es escribir los comandos de trazado de la gráfica en una cinta magnética. La cinta es entonces transferida a un controlador-procesador de gráfica donde las instrucciones procesadas grabadas en la cinta magnética van al plotter a una velocidad compatible para la traza sin ligadura con el computador central.



PLOTTER
DE
TAMBOR MAGNETICO

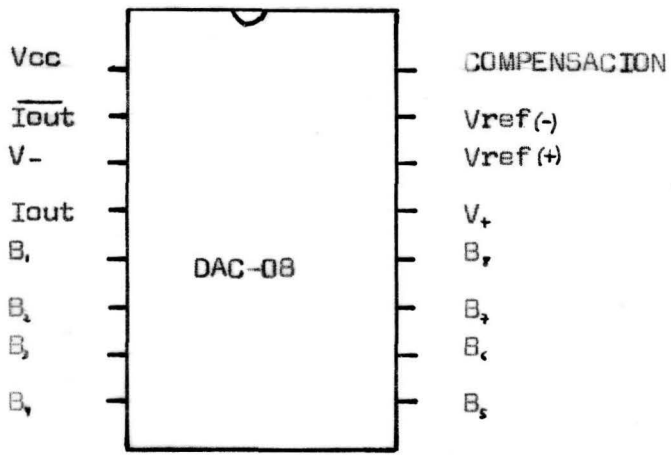


PLOTTER X-Y
PM8043

PARTE II

COMUNICACION

MDS-221 Y PLOTTER

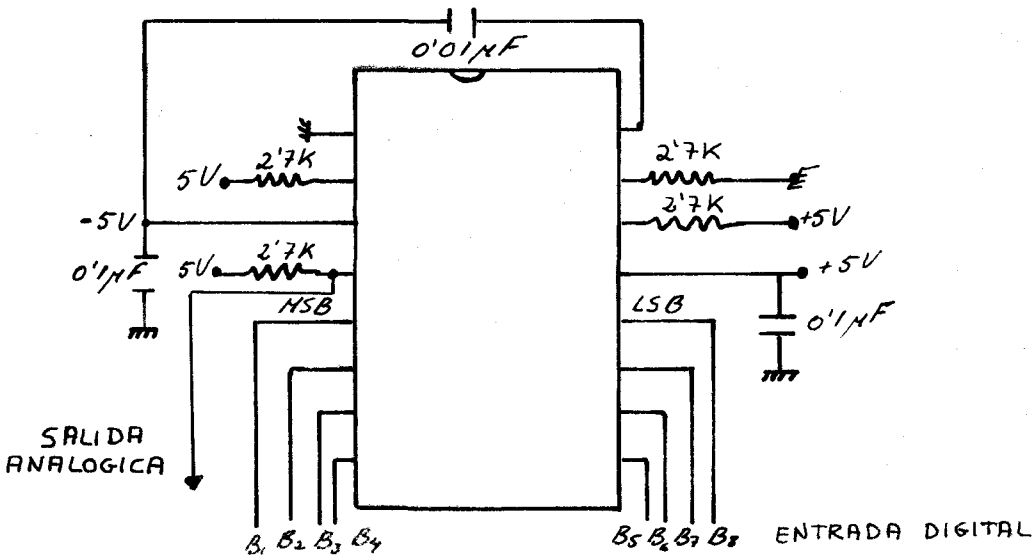


COMUNICACION MDS-221 Y PLOTTER

La idea general del proyecto es trazar gráficas con el plotter X-Y PM 8043 existente en la escuela, conectándolo con el Sistema de Desarrollo por medio de un interface:

En principio como interface se pensó en el microcomputador de enseñanza SDK-85, de forma que el Sistema de Desarrollo actue sobre el y este saque la información por los puertos al plotter. Pero al ser la salida del SDK digital y el plotter analógico era necesario un convertidor D/A, concretamente dos, uno para el puerto A y otro para el puerto B.

Se utilizó el DAC-08 por su alta velocidad, tiempo de conversión de 85nseg, y además ya habia sido utilizado en otras prácticas funcionando de acuerdo con lo que se requiere aquí.

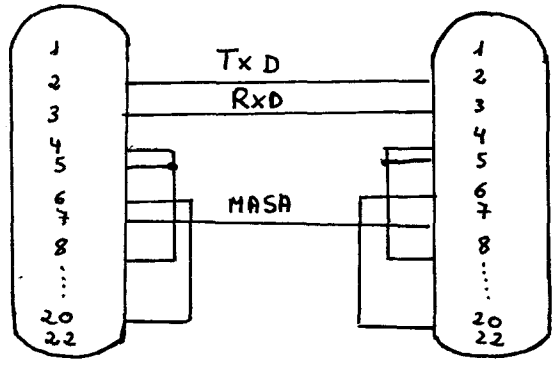


La configuración es la siguiente:

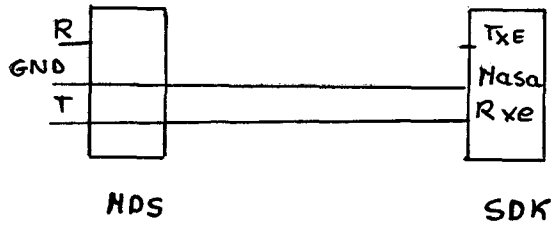
La entrada del convertidor puede ir de 00 a FF. Graduando convenientemente el plotter se consiguió que FF coincidiera con el máximo valor en la escala del plotter, tanto en el eje X como en el Y, y 00 con el menor valor en la escala.

Una vez conseguida la graduación del plotter con el SDK y la realización de programas sencillos de ondas triangular y cuadrada por medio del teclado del SDK, pase a la segunda parte del esquema de partida: comunicación MDS-221 y SDK.

Dicha comunicación se basa en la comunicación entre las USART's de ambas. Tanto para la transmisión como para la recepción sólo se necesita las lineas Txd, Rxd y masa, según la configuración:



De forma que para que el Sistema de Desarrollo y el SDK reciba según nos interesa. Ambas masas deben estar conectadas entre si, el cable de transmisión del MDS debe conectarse con el de recepción del SDK:



NOTA: Si se desea ampliar conocimientos sobre dicha comunicación referirse al proyecto de Domingo Marrero Marrero.

Un ejemplo de prueba del funcionamiento correcto de la comunicación entre el Sistema de Desarrollo y el SDK-85, es mediante dos sencillos programas. El TXE que se ejecuta en el MDS-221 y que permite sacar una onda triangular o cuadrada según se desee, a una frecuencia puesta por el usuario. Teniendo en cuenta que la frecuencia metida por el teclado del Sistema de Desarrollo está en ASCII y necesitamos su equivalente en Hexadecimal, por lo que es

conveniente conocer el código ASCII para tener opción a todas las posibilidades que este nos brinda.

Al mismo tiempo tiene que estarse ejecutando otro programa en el SDK que lo que hace es, primero programar el timer de la RAM de expansión a una frecuencia de 19 2KHz, mediante:

```
MVI  A,40H
OUT  2DH
MVI  A,0A0H
OUT  2CH
MVI  A,0C0H
OUT  28H
```

Despues se programa la USART (teniendo en cuenta que la programación de las USART s conectadas entre si han de ser iguales), esto se realiza mediante:

```
MVI  A,4EH
OUT  81H
MVI  A,27H
OUT  81H
```

Por último se hace un bucle de espera de datos para mandar los datos al acumulador para su posterior salida por cualquiera de las posibilidades que nos ofrece el SDK según el caso: en el nuestro los datos van el monitor del SDK-85:

```
ESPERA: IN  81H
        ANI  02H
        JZ   ESPERA
        IN   80 ; dato al Ac.
        CALL UPDDT
```

Se vió posteriormente la necesidad de utilizar un lenguaje de mayor potencia por lo que se pensó en el PLM-80 entre otras causas por ser un lenguaje de alto nivel, estructurado, que nos permite acceder a todas las ventajas del Sistema de Desarrollo con más facilidad.

Como aplicación a este lenguaje se llegó al programa PLOTEA.SRC que es un programa más compacto en donde la transmisión es una simple llamada a subrutina, concretamente a la subrutina TRANS. También se tuvo en cuente la conversión de ASCII a Hexadecimal.

Al tener que jugar con el valor del lápiz, eje X e Y se tuvo en consecuencia que modificar el programa del SDK y aparte de la programación del timer y de la USART habia que programar los puertos, de forma que el primer dato correspondiente al lápiz saliera por el puerto C, el segundo dato el eje X por el puerto A y el tercero el eje Y por el B.

Quedando el programa como sigue:

```
        LXI  SP,28FFH
        MVI  A,0FH ; programación de los puertos
        OUT  20
        MVI  A,FFH
        OUT  23
PROGRAMACION DEL TIMER
PROGRAMACION DE LA USART
  AQUI: CALL READY
        IN   80
        OUT 23
        CALL READY
        IN   80
        STA  4F20
        CALL READY
        IN   80
        OUT  22
        LDA  4F20
        OUT  21
        JMP  AQUI
 READY: IN   81
        ANI  02
        JZ   READY
        RET
```

Pero habia un inconveniente y era que cada vez que se deseaba utilizar el plotter habia que, aparte de ejecutar el programa PLOTEA, meter el programa anterior en el SDK y esto sin ser una

tarea complicada, era muy tediosa y suponia una perdida de tiempo. Como primera salida para evitar esto y la mas elemental era grabar el programa en la ROM del SDK a partir de una posición de memoria que con solo acceder a ella se ejecutará el programa.

Pero por otro lado surgió la pregunta: ¿ Tendrá el Sistema de Desarrollo otras posibilidades de conexión con los periféricos, concretamente con el plotter de las anteriormente estudiadas??. De esta forma nos quitariamos el trabajo de conectar el SDK que hasta este momento habia sido indispensable para conocer las posibilidades del plotter.

Por lo que mirando las caracteristicas del Sistema de Desarrollo y las posibles conexiones con perifericos se puede ver que tiene 6 conectores traseros:

1. Para una salida TTY
2. Para un terminal
3. Salida para tarjeta perforada. Punch
4. Lectora de tarjetas
5. Programador universal de PROMS
6. Salida para impresora de linea

De dichas salidas dos estan siendo utilizadas, concretamente la de

impresora y terminal. La lectora de tarjetas es una entrada y la del programador es especifica. La salida TTY se podria haber utilizado pero era poco práctico por su complicación hardware. Por lo que utilize la salida para tarjetas perforadas (PUNCH), que es una salida en paralelo de 8 bits.

Esto supuso desde el punto de vista software tener que modificar el programa PLOTEA. Ya no hacia falta la subrutina TRANS, sino que con los procedimientos de la libreria del PLM-80, tales como PO e IOSET se habilitaba la transmisión. El programa pasó a llamarse APLOT.SRC que es un programa mas compacto y la última versión de conexión MDS-221 y plotter.

Ambos el programa y las subrutinas anteriores serán explicadas en un apartado dedicado en exclusivo a ello.

Desde el punto de vista hardware el SDK se sustituyo por 7 integrados, concretamente por:

- Dos inversores 7404
- Un contador por 4, el 7493
- Tres registros de 8 bit. 74LS273
- Un decodificador/multiplexor. 74LS138

A la hora de trabajar con el Punch hay tres patillas a tener en cuenta a parte del bus de datos. Dichas patillas son la Punch command, Punch ready, Sistem ready.

La SYSTEM READY indica que la alimentación esta aplicada al Punch.

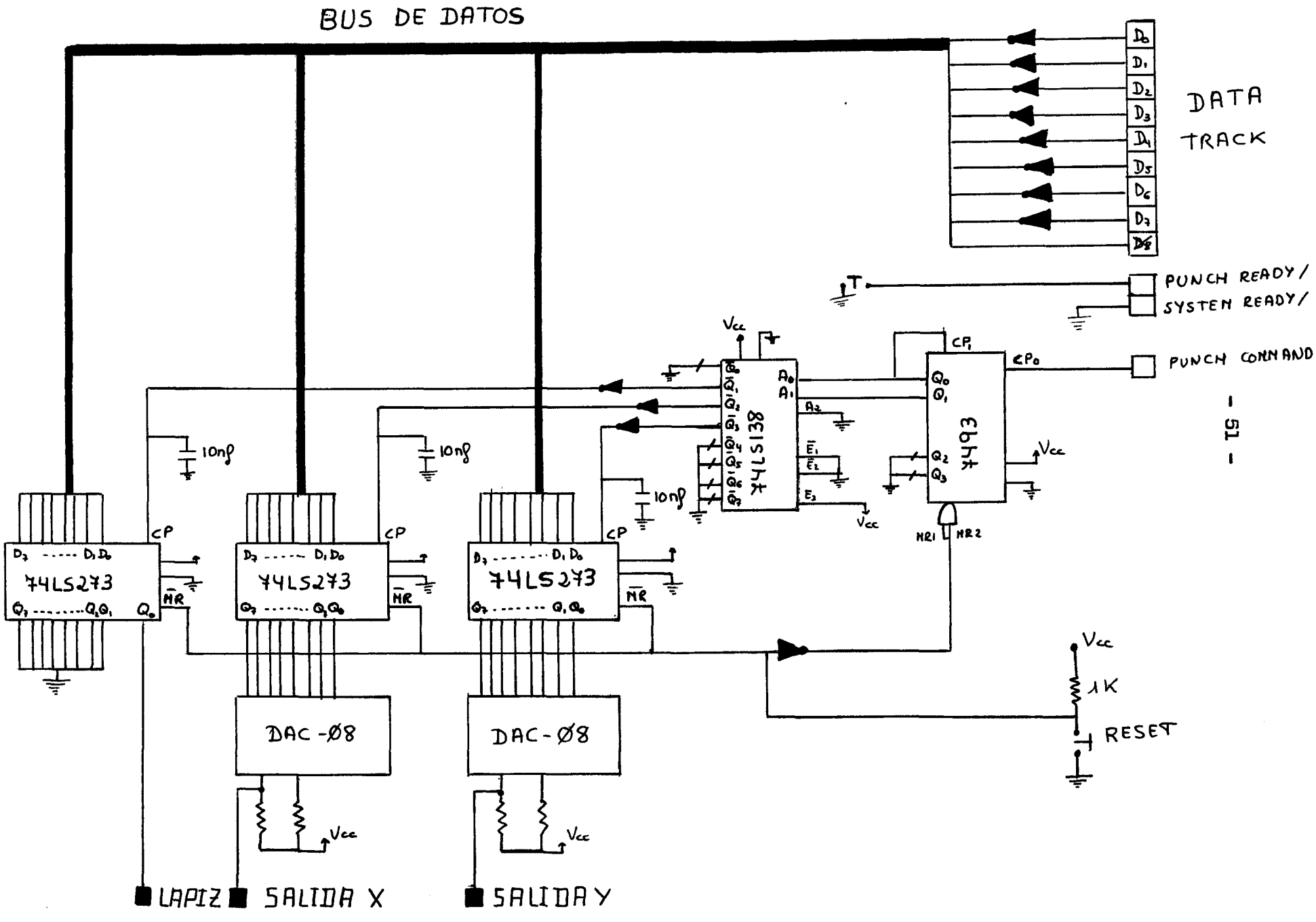
La PUNCH READY cuando el Punch esta preparado para ejecutar una operación.

El PUNCH COMMAND indica cuando hay un dato válido en el bus de datos, en cuyo caso actua sobre la entrada de reloj del contador.

El 7493 se puso como contador por 4 y activa al decodificador que a su vez va a seleccionar a cada uno de los tres registros, de forma que el primero es el lápiz, el segundo el eje X que a su vez va al DAC-08 y el último el eje Y igual que le anterior.

La cuarta cuenta del contador se soluciona por solfware poniendo un PO(Ø) para que no ocasione algun transitorio, ya que solo nos interesa las tres primeras cuentas.

Este montaje acarreó en principio un problema debido a la existencia de unos transitorios que originaba en el plotter unos picos muy pronunciados. Esto se solucionó colocando un condensador de 10nf del CP de cada registro a tierra.



PROGRAMA DE FUNCIONAMIENTO DEL PLOTTER

PROGRAMA DE FUNCIONAMIENTO DEL PLOTTER

El programa APLOT en su conjunto lo que hace, es tomar la base de datos procedente del terminal en forma de fichero por medio del QUILE6, y siempre que cumpla el formato especificado pasarlo al plotter.

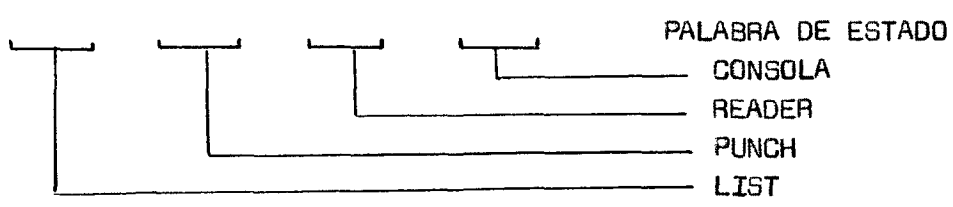
El programa hace uso de una serie de rutinas propias de la libreria del PLM80 tales como: EXIT, PO, IOSET, OPEN, READ, WRITE, CO y CI que permiten un mejor aprovechamiento de este lenguaje y un tratamiento muy potente de los periféricos del sistema. También de subrutinas internas como: HEXASC, PUNTOS, COGEB, ACCFIL, DATA$IN.

Comenzaré matizando que el programa lo primero que hace es sacar en pantalla "NOMBRE DEL FICHERO? " y haciendo una llamada a la subrutina DATA$IN; la cual permite poner el nombre del fichero a dibujar y utilizar por programa la tecla Rubout para poder borrar.

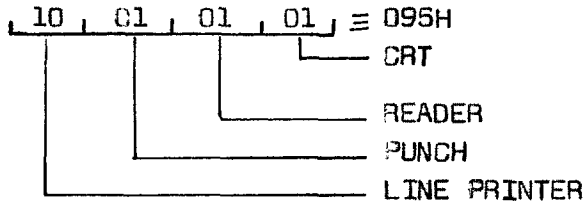
Posteriormente se modifica la configuración de I/O, para que sea uno de los conectores traseros del Sistema de Desarrollo, concretamente el PUNCH(salida para tarjeta perforada) la salida. Todo esto por medio del procedimiento IOSET. Concretamente

CALL IOSET(095H), donde el 095H se debe a:

VALOR	CONSOLA	READER	PUNCH	LIST
00	TTY	TTY	TTY	TTY
01	CRT	HS READER	HS PUNCH	CRT
10	BATCH	U.D	U.D	LINE PRINTER
11	U.D	U.D	U.D	U.D



Según esto se obtuvo:



Luego se abre el fichero y se lee los 128 byte primeros por medio de los procedimientos OPEN y READ. Para empezar se pone el lápiz arriba (0FFH) para evitar errores.

Seguidamente se llama a la subrutina PUNTOS que hace un tratamiento de los datos, obtiendo los valores de X e Y,y para ello

llama a la subrutina COGEB que toma dato por dato y a su vez llama a dos subrutinas; primero a la ACCFIL que mira si el dato es final de fichero o no y en caso afirmativo manda a leer los 128 byte siguientes, y luego llama a la subrutina HEXASC que hace la conversión de ASCII-HEXADECIMAL de la siguiente forma.

Pone el flag a cero y mira si el dato esta comprendido entre 0 y 9 o entre A y F, si esta comprendido entre 0 y 9 le hace un ANI(0FH) para tener solo la parte menos significativa y retorna el valor, si esta entre A y F aparte del ANI le suma al dato 09H y retorna el valor al punto de llamada.

Si no esta comprendido entre 0 y 9 o A y F mira si es X, L, U o CR.

Si es X pone el flag a 2 y retorna el valor que en la subrutina PUNTOS es asignado como fin de fichero.

Si es L pone el flag a 3 y retorna egualmente a la subrutina PUNTOS donde implica lápiz abajo.

Si es U pone el flag a 4 y significa lápiz arriba.

Si es CR pone el flag a 5 y retorna para que no se tenga en cuenta y no perder el sincronismo.

Una vez hecha la conversión pone el flag a uno.

Obtenidos los valores del lápiz de X e Y lo primero que se

pregunta el programa es si el valor del lápiz es igual o distinto al que tenia anteriormente. En ambos casos transmite el nuevo valor del lápiz, pero la diferencia esta en que si el nuevo valor es distinto del anterior se introduce un retrazo (CALL TIME) mayor para evitar movimientos bruscos del lápiz.
Posteriormente se transmite el valor de X y el de Y por medio del procedimiento PO(Punch output routine) que toma un byte y lo transmite al dispositivo Punch.
Por último se transmite un PO(Ø) por razones hardware explicadas en otro apartado.

Otra de las ventajas que tiene el programa y que solo mensione anteriormente por arriba es la capacidad de sincronismo, de forma que cualquier error que se meta en el fichero durante la transmisión es captado y pasado por alto siguiendo el plotter dibujando la gráfica. Dicho sincronismo se consigue con el valor del lápiz, es decir, cada vez que se transmita el valor de X e Y hay que especificar la posición del lápiz (arriba "U", abajo "L"). De esta manera se detecta cualquier error en la linea.

Hay otros procedimientos que no he mensionado y que aparecen en la subrutina DATA$IN como:

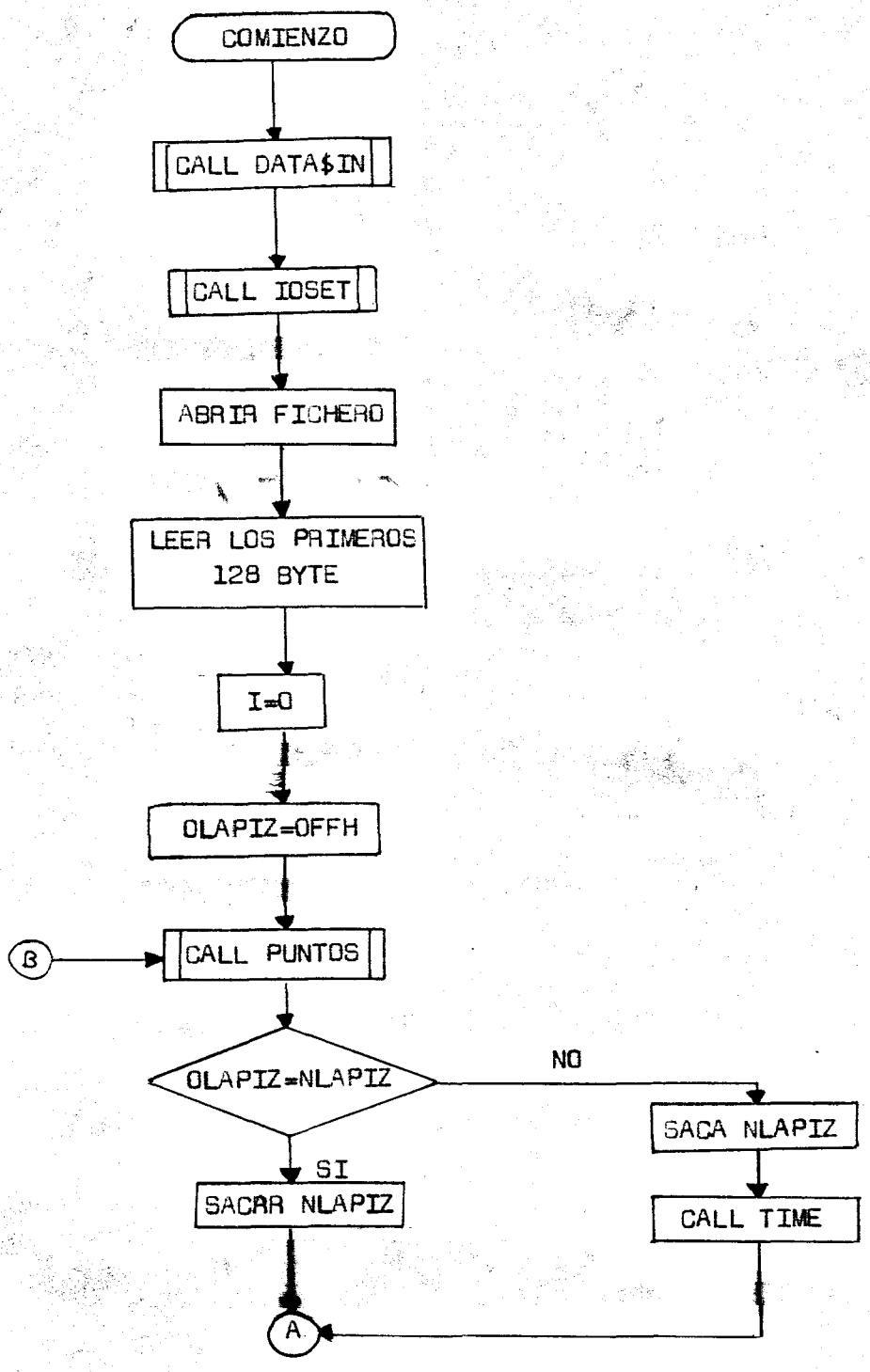
CO: Toma un caracter y lo transmite al dispositivo de salida, en

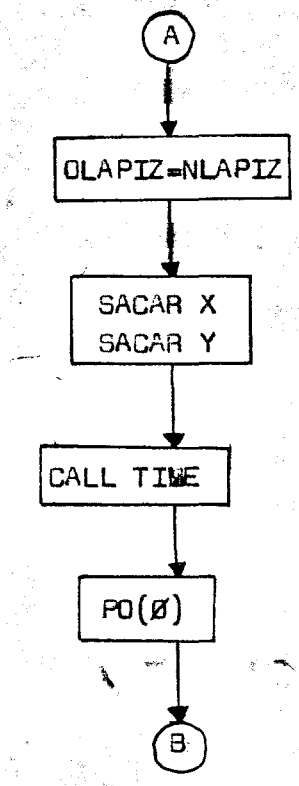
este caso la pantalla.

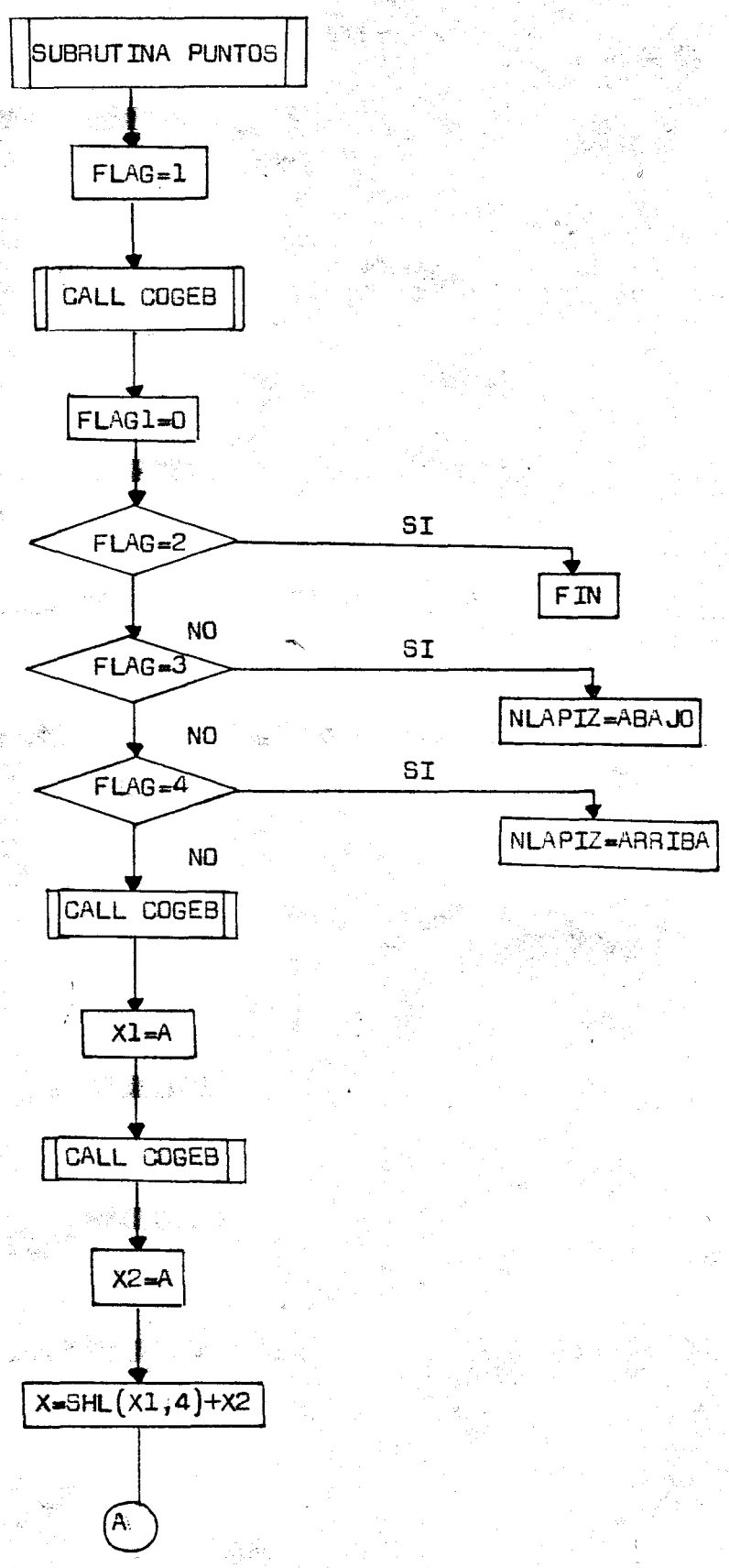
CI: Permite leer desde el teclado del INTELLEC un caracter.

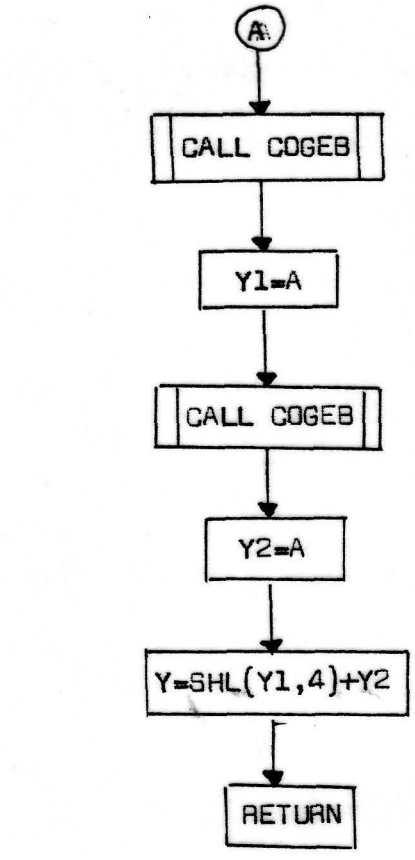
El programa termina con la rutina EXIT que retorna una vez ejecutado el fichero, el control al ISIS-II.

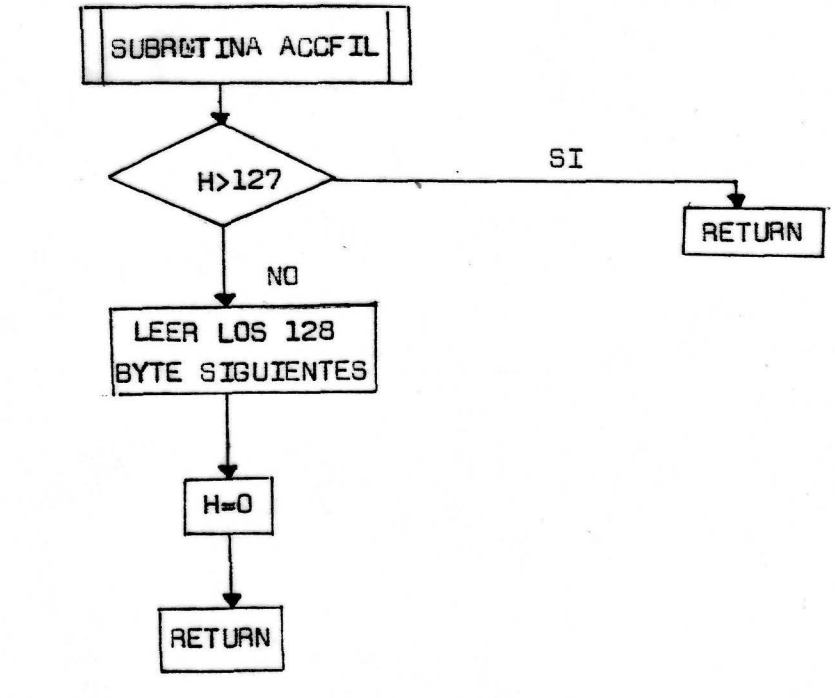
PROGRAMA PRINCIPAL

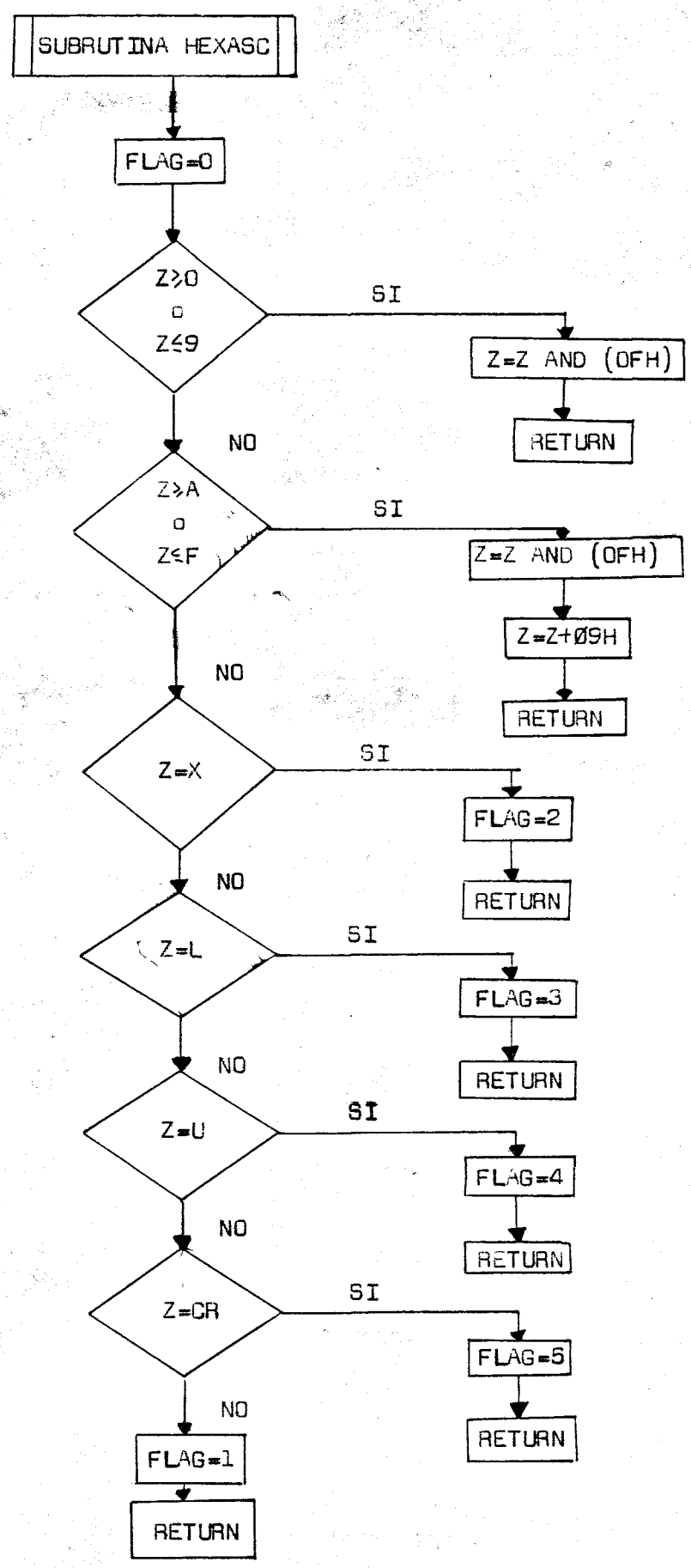


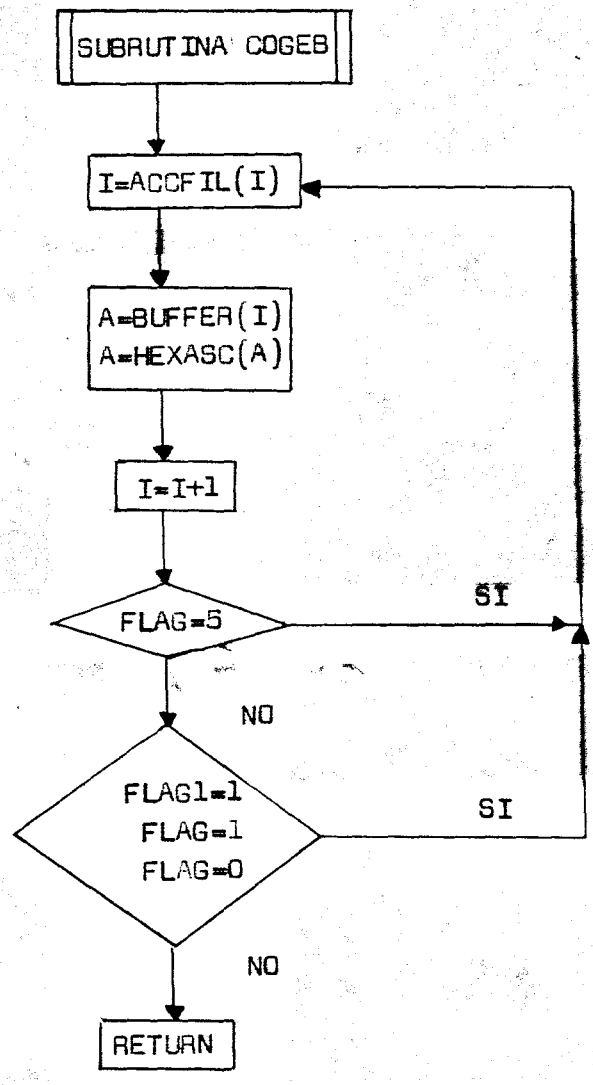












GRAPHIC5

GRAPHICS

GRAPHICS es un programa realizado en BASIC que permite dibujar cualquier función en coordenadas cartesianas en el plotter.
Para ello el programa consta de una serie de subrutinas tales como: conversión decimal-hexadecimal, formateado de variables y escritura en fichero.

En general el programa define una variable string 09$ dimensionada en 80 caracteres, de forma de que cada vez que se llene se vuelca sobre un fichero que es la base de datos.

Una de las peculiaridades del programa es la utilización de una función recursiva.

Una función recursiva es una función que se llama asi misma; por ejemplo, el factorial de un número se define como dicho número por el factorial del número anterior:

$$5!=5.4!=5.4.3!$$
$$4!=4.3!$$

Su propia definición se incluye asi misma.

En nuestro caso, cuando se tenia que trazar una linea entre dos puntos distantes en el plotter, la linea perdia linealidad. Por lo que para resolverlo se podria haber utilizado la ecuación de la

recta pero suponia mayor complicación de cálculo y del programa. Mientras que con el recurso de la función recursiva el programa se hizo mas compacto.

Profundizando en el programa tenemos que la función recursiva se definió con 5 variables:

X8,Y8 Punto inicial

X9,Y9 Punto final

L9 Es el comando para las distintas posibilidades que se pueden dar:

L9=0 Traza una recta de (X8,Y8) a (X9,Y9) con el lápiz abajo

L9=1 Va de (X8,Y8) a (X9,Y9) con el lápiz arriba

L9=2 Inicializa las variables

L9=3 Vuelca las variables en un buffer hasta un end of file

En el caso de L9=0, es decir a la hora de trazar una recta es cuando se puede observar el uso de la función recursiva.

Antes de nada hay que tener en cuenta que el plotter se considera a la hora de dibujar como un cuadrado de 256x256 puntos, y que cuando la recta es mayor de 20 pierde linealidad.

Una vez aclarado esto se podrá entender porque cuando la recta es mayor de 20 se le aplica la función recursiva.

Con la función recursiva se divide la recta por la mitad y se vuelve a comparar, así sucesivamente hasta conseguir una recta de distancia adecuada. Trazandose posteriormente la recta de división en división hasta llegar a la recta final deseada.

Como he dicho anteriormente también consta de una subrutina de conversión decimal-hexadecimal que funciona de la siguiente forma; Se toma la parte entera del múmero en decimal y se divide entre 16 si el resto es mayor de 9 se hace la conversión según la tabla:

10=A
11=B
12=C
13=D
14=E
15=F

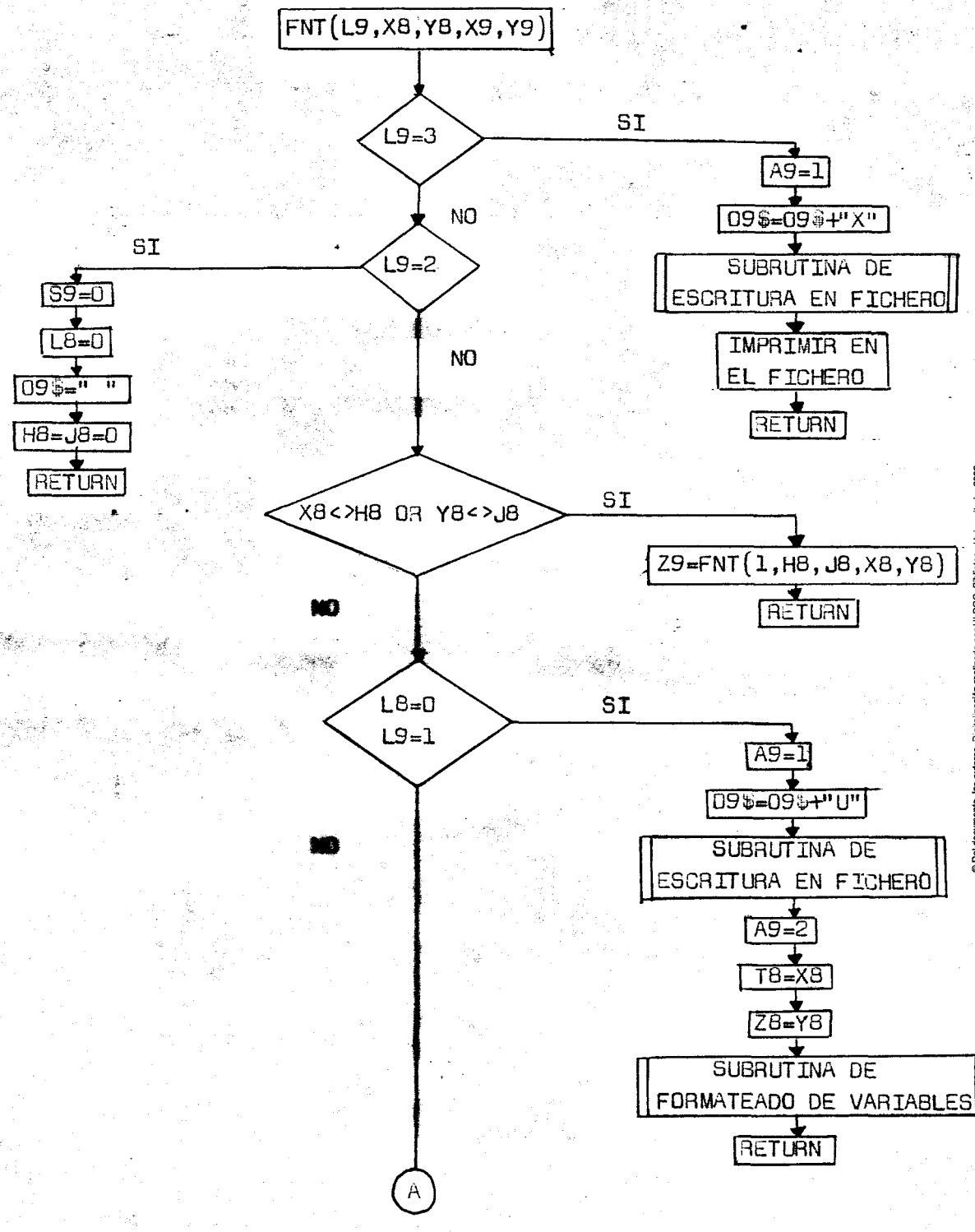
Si el resto es menor de 9 se pasa a string y se suma con A9$, que es donde va a estar el número en hexadecimal, la suma siempre en el orden indicado puesto que a la hora de escribir el número en hexadecimal se va de derecha a izquierda.

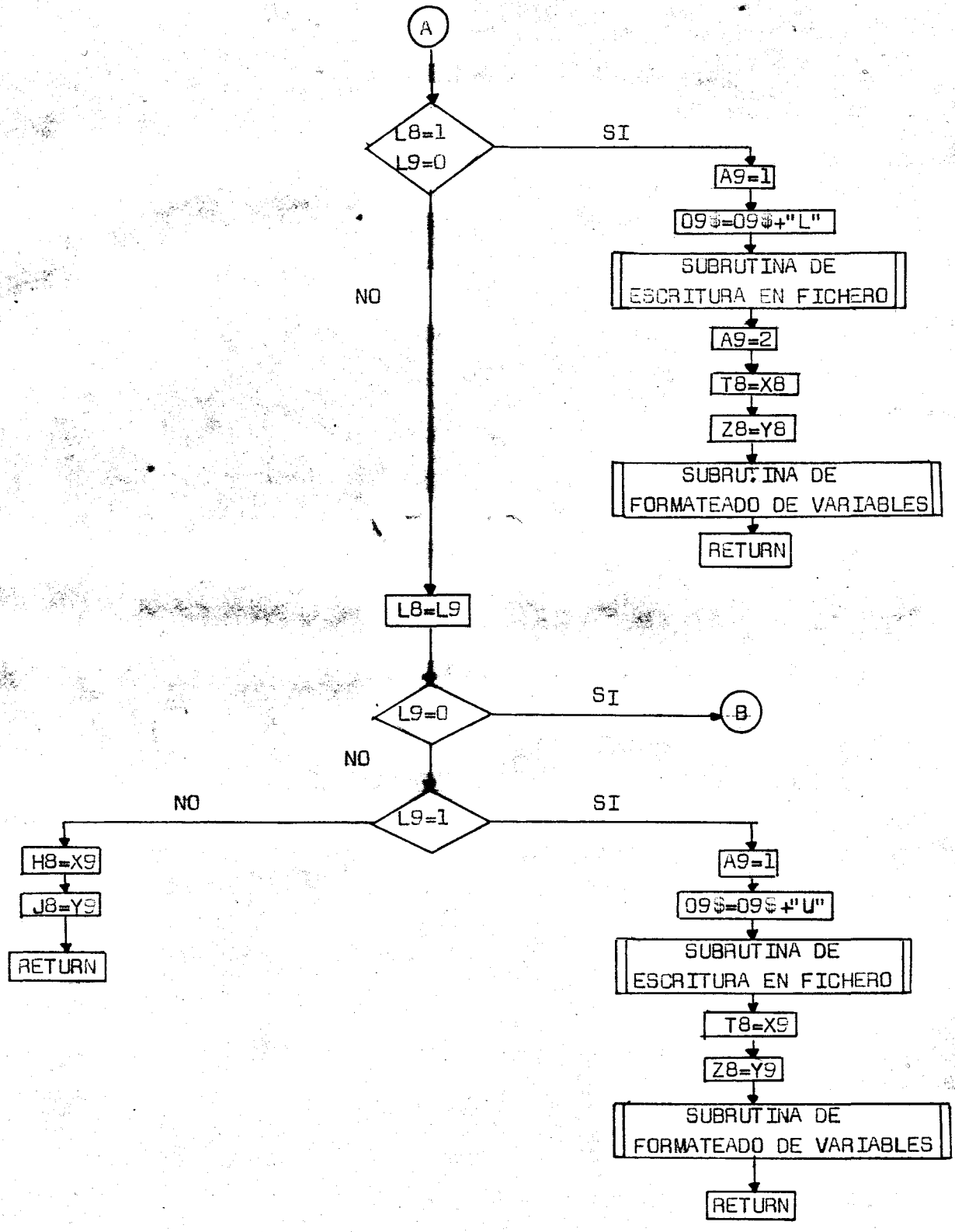
Se mira si el cociente es mayor de 16, en cuyo caso se hace un cambio de variable y se repite el proceso anterior. Si es menor de 16 se mira si la conversión esta hecha en cuyo caso se retorna el valor.

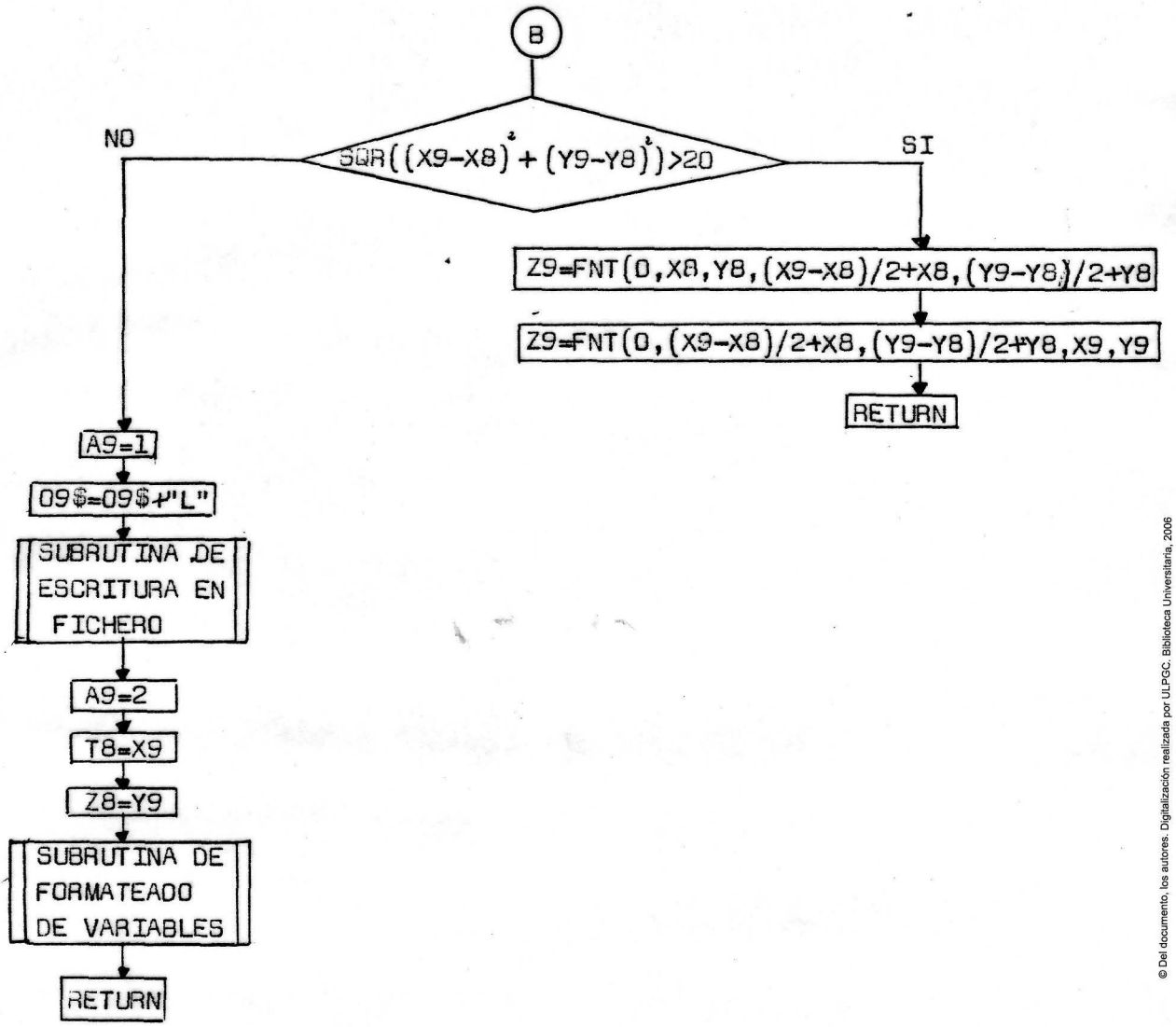
Por último otra de las caracteristicas del programa es que esta optimizado con las sentencias:

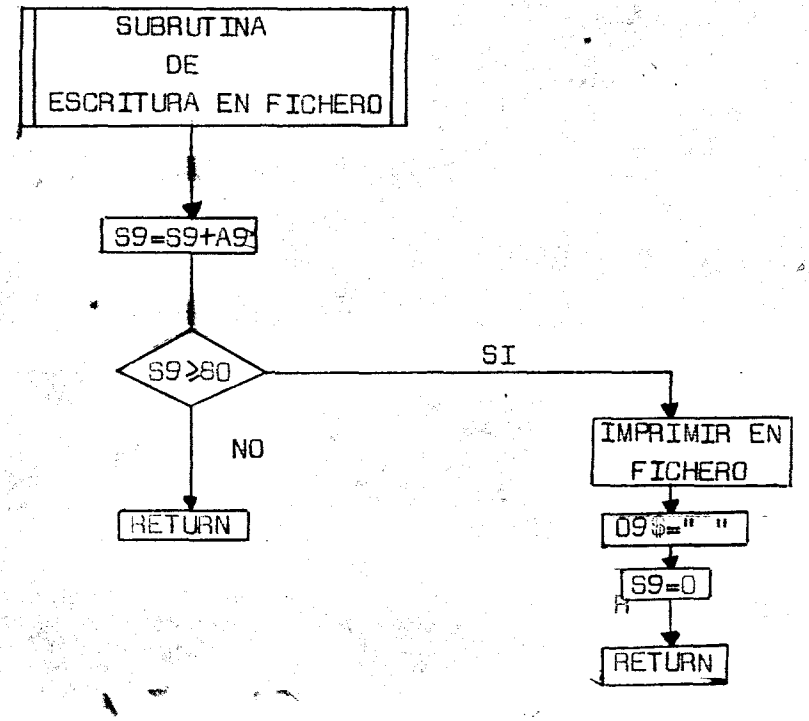
```
IF X8<>H8  OR  Y8<>J8 THEN DO
   Z9=FNT(1,H8,J8,X8,Y8)
```

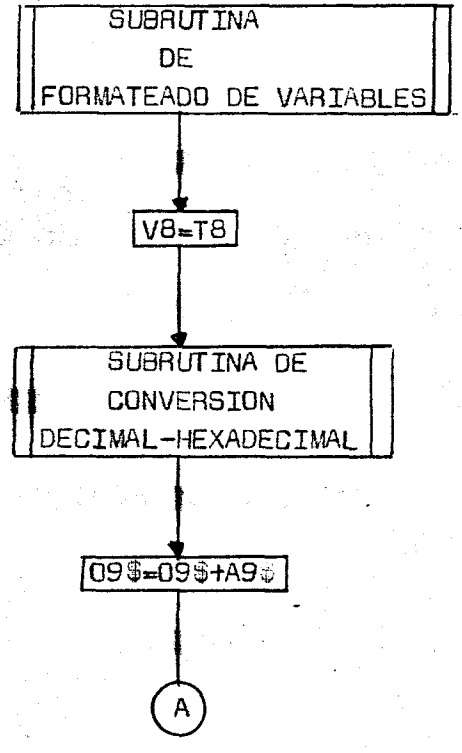
Lo cual permite ahorrar posiciones en la base de datos para poder poner mas caracteres.

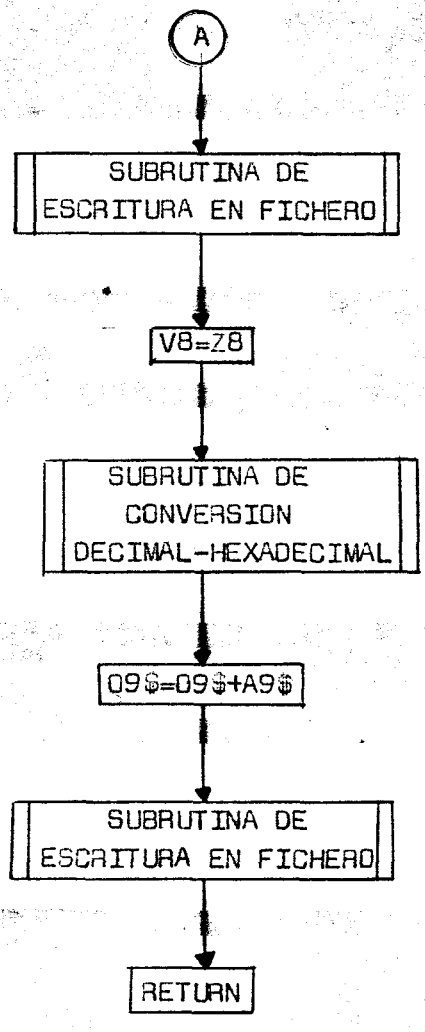


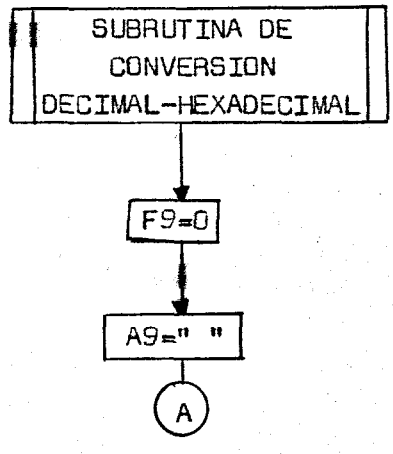


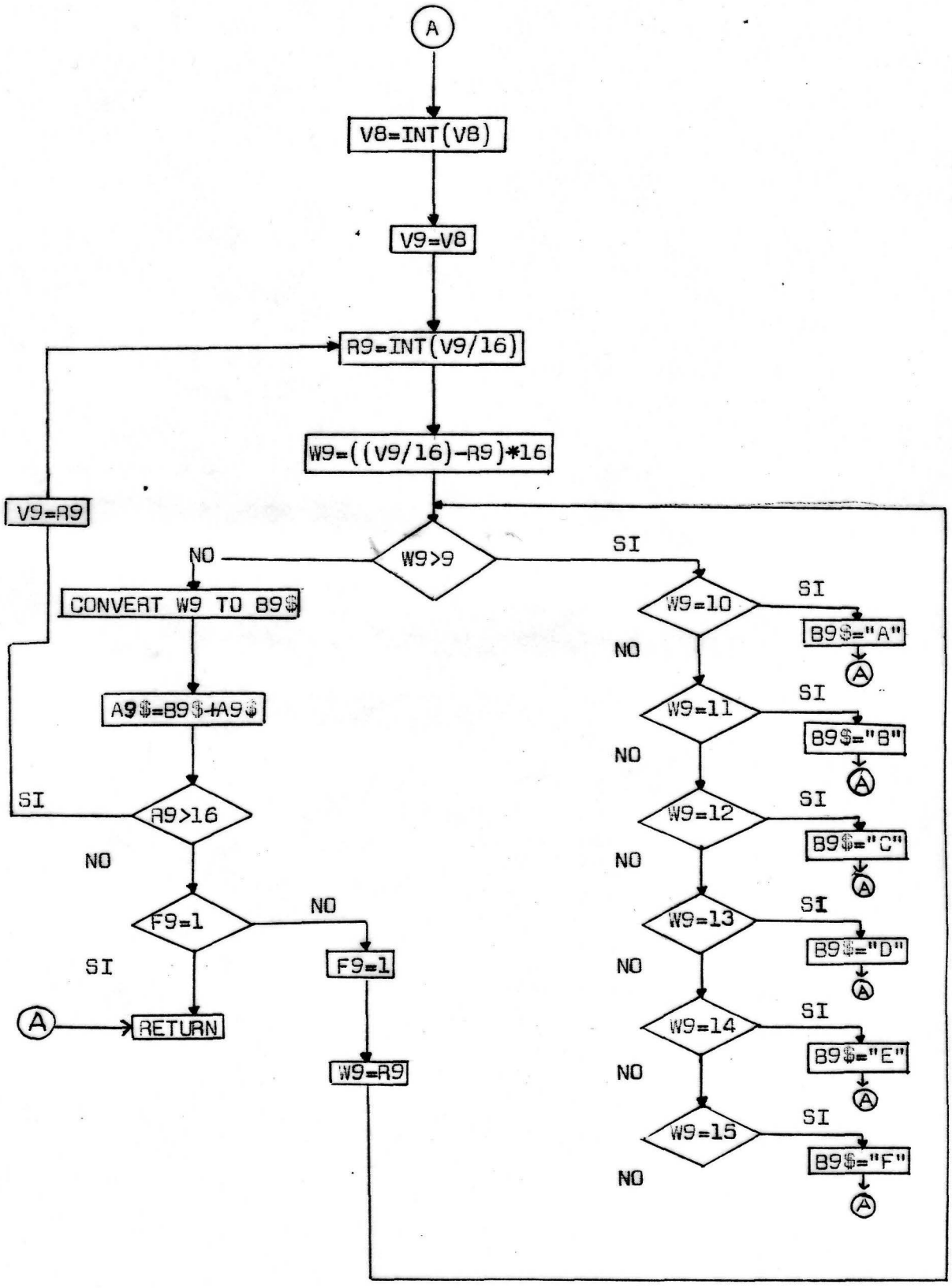












ANEXO

METODO DE TRABAJO

Para dibujar una gráfica en el plotter lo primero que hay que hacer es crear un programa en el terminal con la función a dibujar. Dicho programa tiene que cumplir un formato previo:

1. Debe empezar en la posición 1120, debido a que el GRAPHICS termina en la posición anterior.
2. Dimensionar las variables utilizadas en el GRAPHICS.

   DIM A9$(2), B9$(2), O9$(80)

3. Definir cual será la base de datos con un:

   FILES "NOMBRE"

4. Fijar los ejes si se desea, teniendo en cuenta las variables de la función recursiva.
5. Definir la función a dibujar
6. Ecuaciones de transformación puesto que hay que pasar de 3 dimensiones a dos (X e Y).

   X1=128-X*(0.707)+Y

   Y1=128-X*(0.707)+Z

7. Vigilar que la gráfica no salga de las dimensiones del plotter

Previamente, antes de escribir el programa anterior se tiene que poner:

```
: BUILD "NOMBRE"; REC=-80; F, ASCII : DISC=200
```

Para crear la base de datos de 80 caracteres y 200 lineas en ASCII.

Seguidamente el programa de la función a dibujar se linka con el GRAPHIC5:

```
SAVE "FUNCION"

GET GRAPHIC5

APPEND "FUNCION"

SAVE "PROGRAMA A CORRER"
```

De esta forma se obtiene la base de datos, la cuál se pasa al Sistema de Desarrollo con el QUILE6. Aplicando en este el programa APLOT se transmite al plotter.

GRAFICAS

Estas tres gráficas corresponden a las funciones:

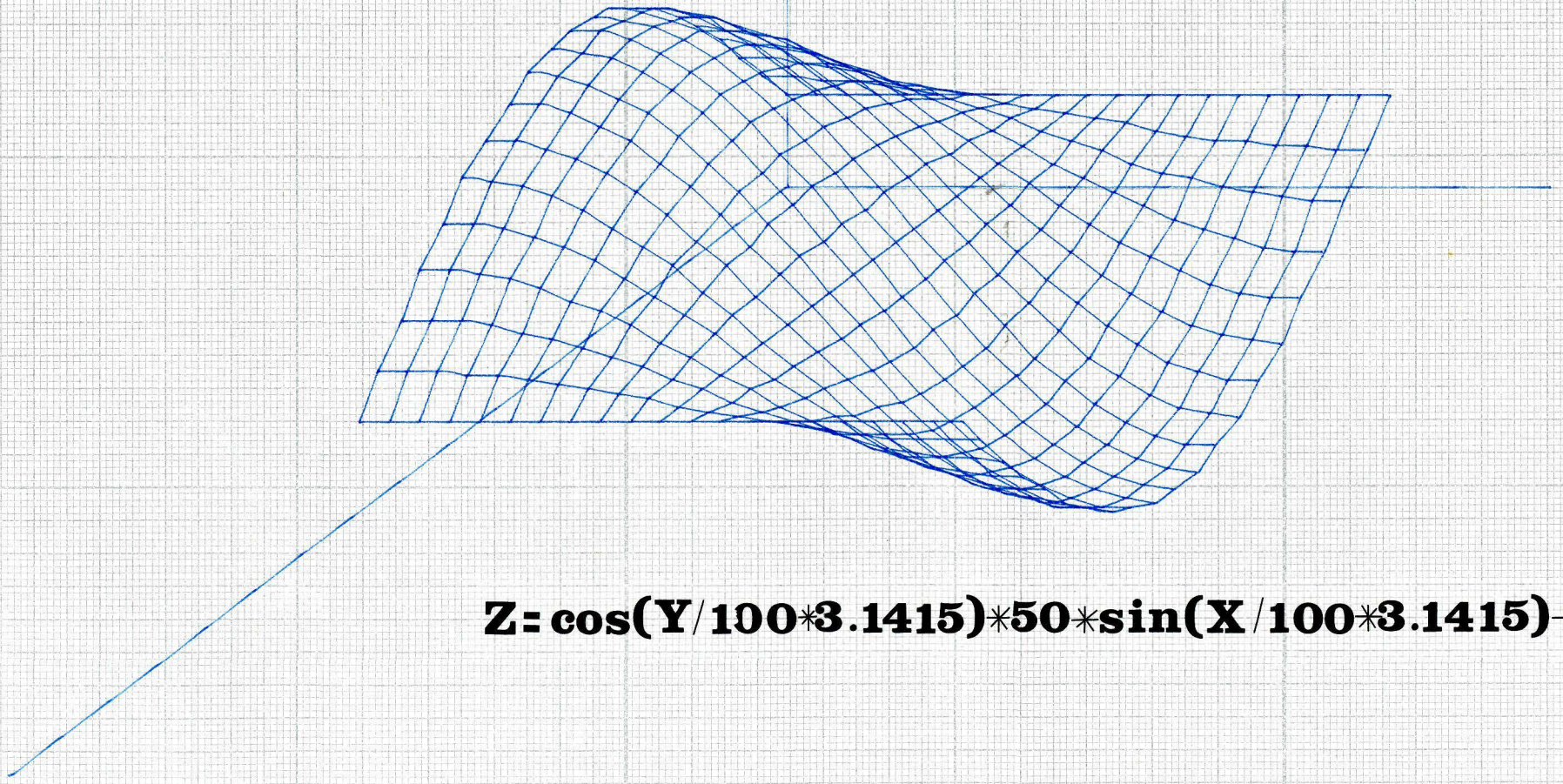
Z=cos(Y/100*3.1415)*50*sin(X/100*3.1415)+20

Z=50*cos(Y/25*3.1415)*exp(-(X/10))

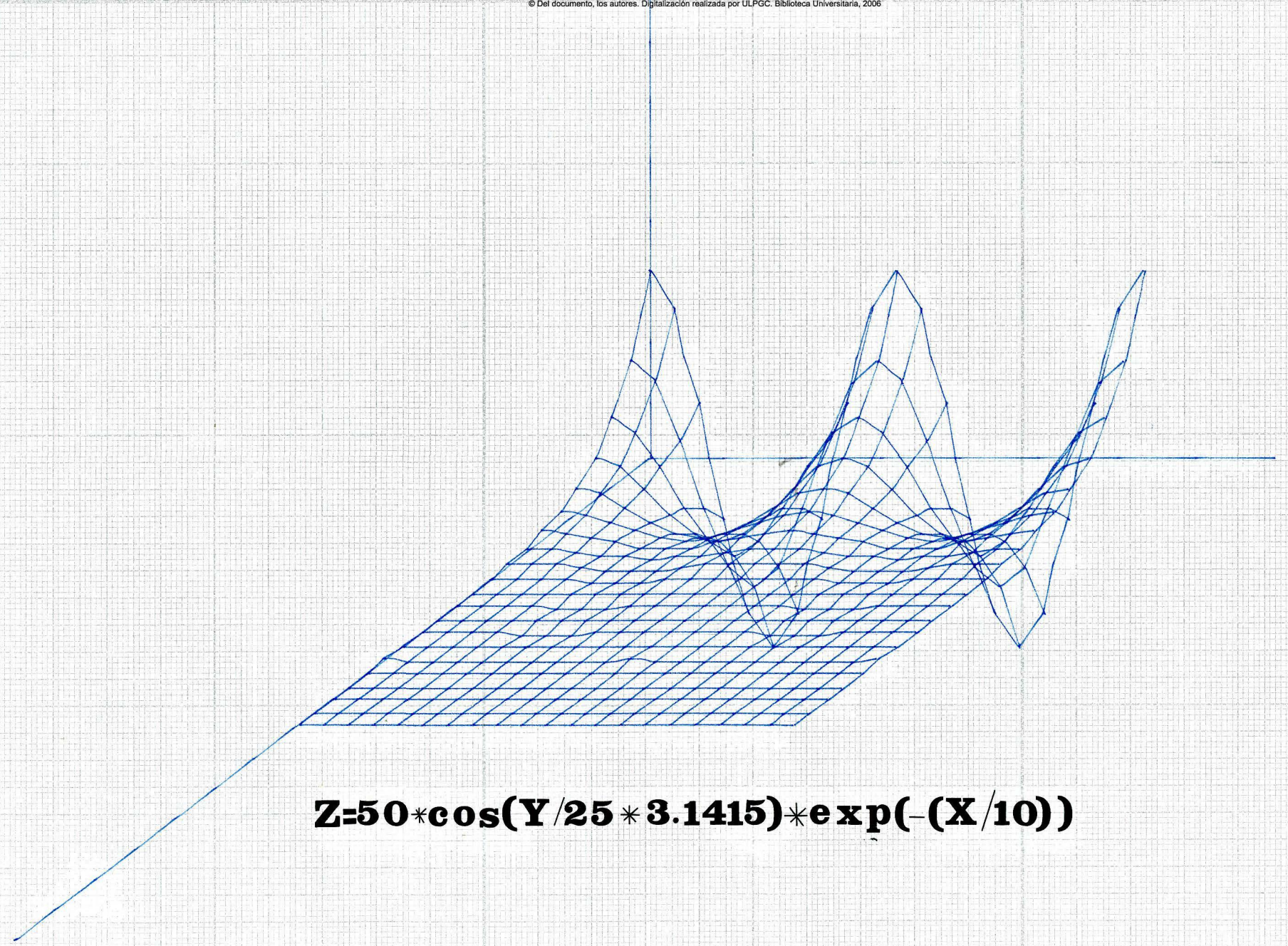
Z=cos(Y/25*3.1415)*100*exp(-SQR(X**2+Y**2))/40)

NOTA: Los programas de las tres gráficas son iguales, solo hay que cambiar la función en la sentencia 1380.

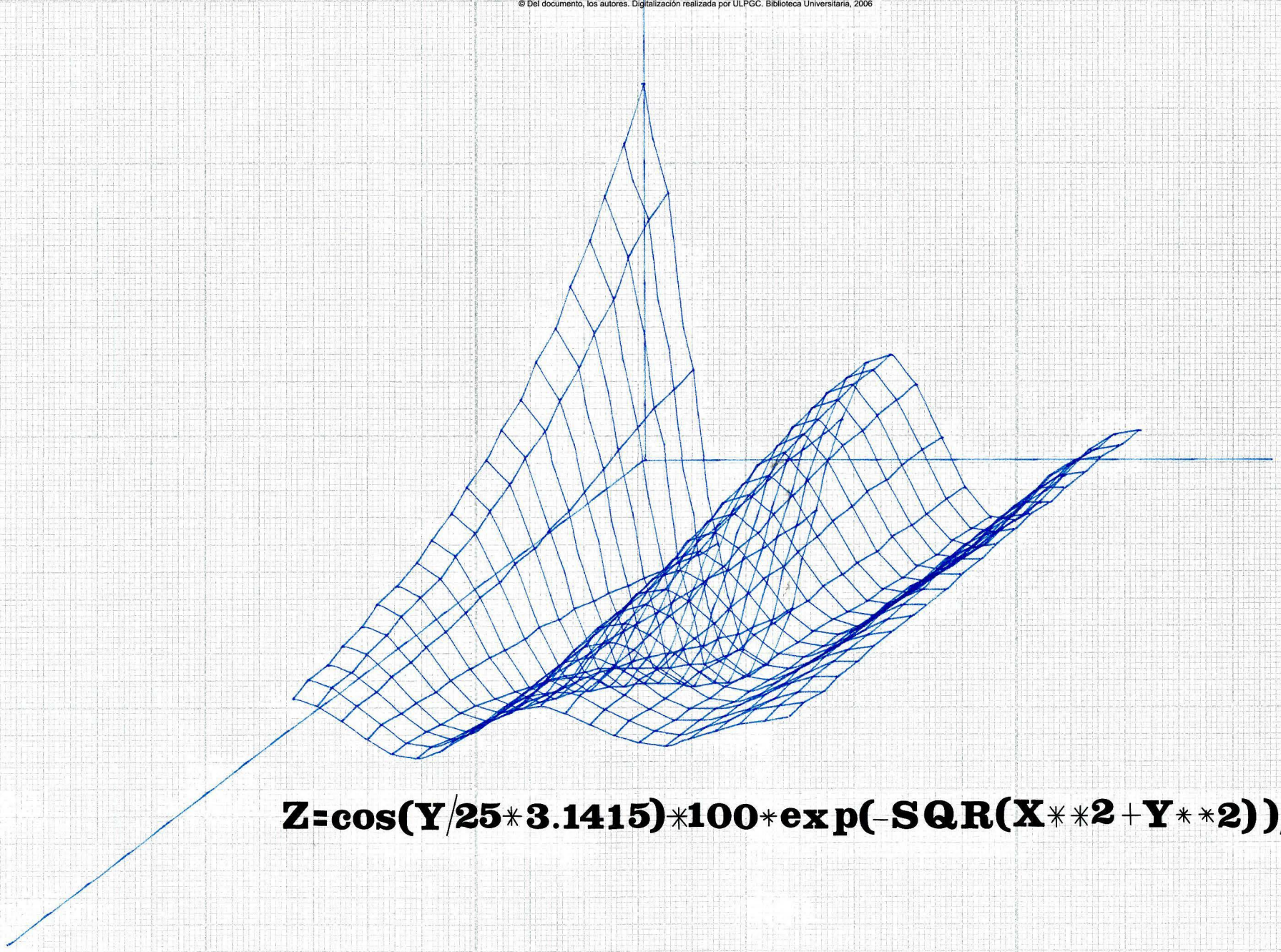
La explicación de dicho programa esta en el apartado anterior(METODO DE TRABAJO).



$$Z = \cos(Y/100*3.1415)*50*\sin(X/100*3.1415)+20$$



$$Z=50*\cos(Y/25*3.1415)*\exp(-(X/10))$$



$$Z=\cos(Y/25*3.1415)*100*\exp(-SQR(X**2+Y**2))/40$$

APENDICE A

CARACTERISTICAS DEL PLOTTER X-Y PM8043

CARACTERISTICAS DE FUNCIONAMIENTO

- Precisión	±0.25%f.s.d. de la temperatura de referencia
- Linealidad	±0.1%f.s.d.
- Amortiguación	sobrepasado el 1% máximo
- Entrada	guardando también entradas traseras impedancia 1MΩ±10% impedancia fuente ≤ 10KΩ voltage máximo 30V ac o 50dc
- Sensibilidad de entrada	9 etapas calibradas de 2mV/cm a 1V/cm en 1-2-5 secuencia
- Supresión del cero	de -5% a +105%
- Base de tiempo	proporciona sobre el eje X solo 0.5sec/cm a 10sec/cm en 1-2-5 secuencia
- Velocidad de escritura	eje X≥80cm/sec eje Y≥120cm/sec
- Aceleración	eje X 2000cm/sec^2 eje Y 5000cm/sec^2
- Sistema de escritura	fundas de lápiz de naylo
- Area	250 X 180mm

INSTRUCCIONES DE OPERACION

- INTERRUPTOR "ON"
  - Una vez conectado el plotter a la red principal colocar el interruptor "SERVO" a "STANDBY", el interruptor "PEN" a "UP".
  - Presionando el interruptor "POWER" a la posición "ON" se enciende una lámpara. Volviendo a presionar el "POWER" a la posición "OFF" se apaga.
- SISTEMA DE ESCRITURA

  Se toma el lápiz de naylo de la caja de servicio. Quitar la tapa protectora de dicho lápiz y se inserta la punta en su contenedor.

  RECORDAR: Cuando el plotter no se usa es recomendable colocar la tapa protectora otra vez:
  - Para evitar que se seque la tinta.
  - Para evitar una linea gruesa cuando un nuevo gráfico se empieza.
- MEDICIONES EN EL MODO X-Y
  - Base de tiempo
    - Colocar el interruptor "ON-OFF" en la posición "OFF"

- Ajuste de la posición del cero
  - Colocar el interruptor "SERVO ON-STANDBY" a "ON"
  - Colocar el interruptor "ZERO-RECORD" a la posición "ZERO"
  - Ajustar el lápiz a la posición requerida en la escala con el potenciómetro "ZERO-ADJ".
- Sensibilidad
  - Colocar el selector "SENSITIVITY" a la posición requerida
  - Con el potenciómetro-interruptor "CAL-VAR" en la posición "CAL" el tramo corresponde al rango seleccionado.
  - Con el potenciómetro-interruptor en posición "VAR" el "SENSITIVITY" se incrementa aproximadamente 2.5 veces con el potenciómetro "VAR" totalmente girado en sentido contrario a las agujas del reloj.
  - Colocar el interruptor "ZERO-RECORD" a posición "RECORD" para poder comenzar.
- Lápiz arriba-abajo
  - El lápiz puede subirse o bajarse con el interruptor "PEN UP-DOWN".

- MEDICIONES EN EL MODO Y
  - Colocar el interruptor "TIME BASE" en posición "ON",una lámpara

se enciende y la señal de entrada X se desconecta.

- Poner la base de tiempo en la posición requerida (sec/cm) por medio del selector "SENSITIVITY" (posiciones 0.5-1.0-2.5-5-10-sec/cm)
- El barrido comienza cuando el interruptor "START-HOLD-RESET" se coloca en "START".

  El punto de comienzo sobre el eje X es determinado por la posición del potenciómetro "ZERO ADJ" (el interruptor "ZERO-RECORD" en "ZERO".

  Al final del barrido el lápiz se levanta y permanece en dicha posición hasta que el interruptor "START-HOLD-RESET" se coloca en posición "RESET".
- Con el interruptor "PEN" en posición "DOWN" el lápiz automaticamente baja o sube al comienzo y final de cada barrido.
- En las posiciones "HOLD" y "RESET" el barrido para y retorna al comienzo respectivamente.

-OTROS CONTROLES

- El interruptor "LINE"

  Por medio de los interruptores de palanca "LINE" las coordenadas pueden dibujarse en la dirección de X y la de Y.

- Interruptor "SERVO"

  En posición "STANDBY" las señales de entrada no son registradas, asi ambos servomotores X e Y son desconectados. Los carros de X e Y pueden entonces ser desplazados manualmente lo cual facilita el cambio de papel y la colocación del lápiz.

  Cuando colocamos el interruptor "SERVO" en "STANDBY" el lápiz automaticamente se levanta.

- Interruptor"FILTER"

  En posición "ON" (posición normal) los voltages de interferencias se suprimen.

-CONTROL REMOTO ("EXT.CONTROL")

El control remoto tiene lugar cuando se aplica una señal lógica (TTL)

Nivel alto: 2.7V a 5V

Nivel bajo: 0 a 0.4V

- Control del lápiz

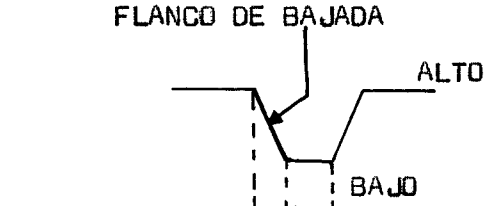
El interruptor "PEN" en el panel frontal en posición "UP"

- Base de tiempo

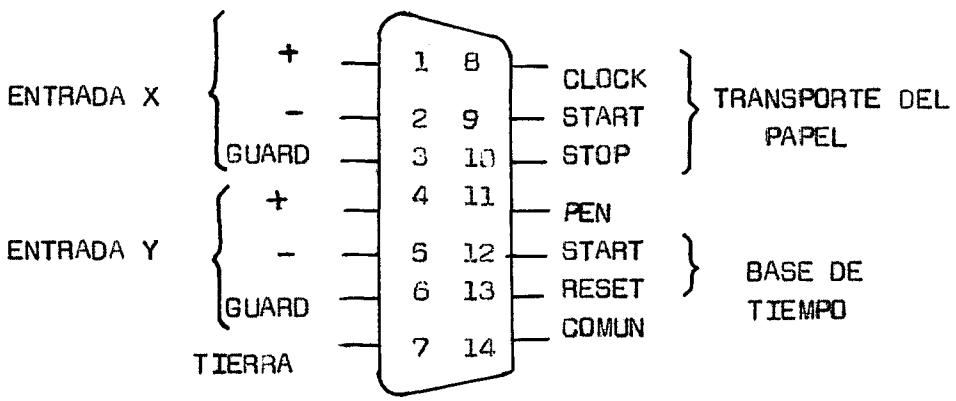
pin 12- START

pin 13- RESET

pin 14- COMUN



El interruptor "START-HOLD-RESET" en posición "RESET"



- PRINCIPIO DE OPERACION

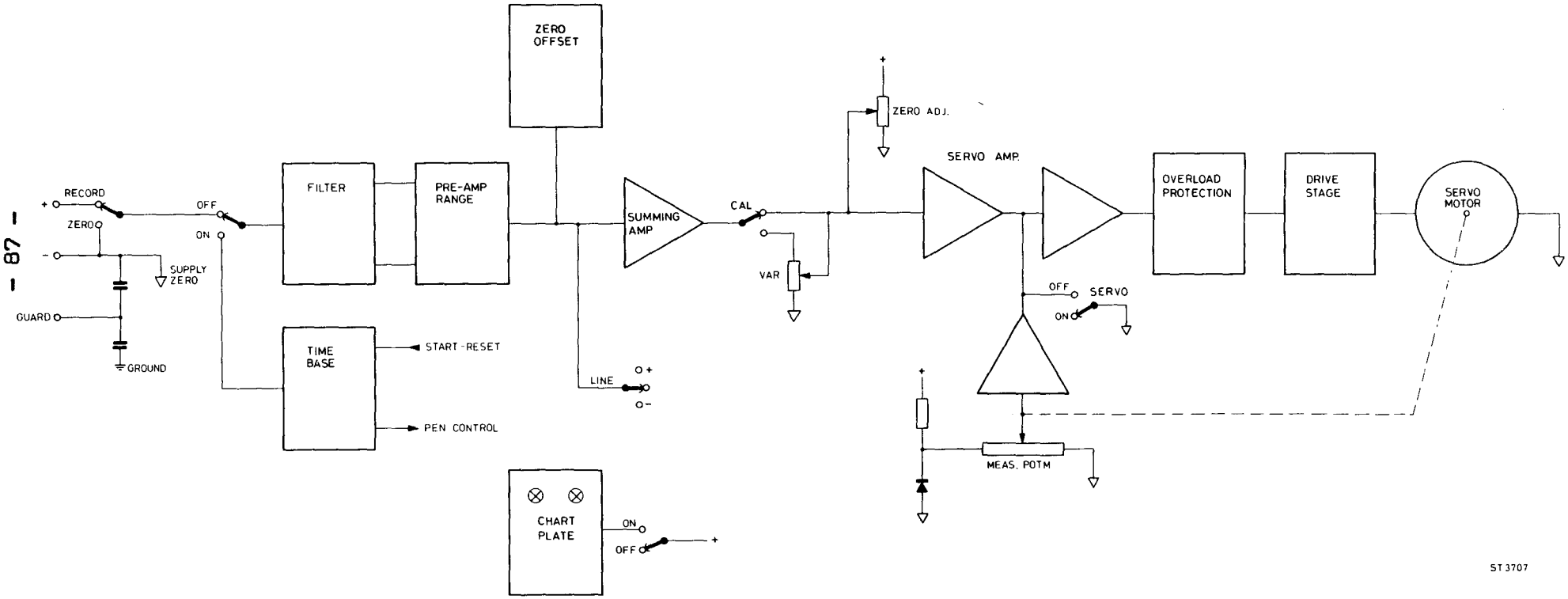
Los plotter se basan en el principio de compensación automática. Los sistemas de medida para X e Y son similares. La señal de entrada se aplica al pre-amplificador de paso al filtro. Por medio del interruptor "TIME BASE" la señal de base de tiempo se aplica al pre-amplificador en lugar de la señal de entrada X. El factor de amplificación del pre-amplificador depende del selector de rango (sensitivity). Al amplificador totalizante debe aplicarse tres señales:

- señal de salida del preamplificador
- señal de supresión del cero
- señal del modo de linea, si se utiliza

En posición "CAL" del interruptor "CAL-VAR" la señal de salida del amplificador totalizante junto con la señal del "ZERO ADJ" se aplica a la primera etapa del servo-amplificador. La salida de esta etapa junto con la señal del potenciómetro de medida se aplica a la segunda etapa.

Cuando la salida de la primera etapa es igual a la señal del potenciómetro de medida , el sistema esta en equilibrio y la salida de la segunda etapa es cero.

El servomotor esta protegido de la sobrecarga. Cuando el interruptor "SERVO" esta en posición "OFF" la entrada de la segunda etapa esta conectada a cero, asi que el servomotor no puede activarse.



APENDICE B

LISTADOS DE PROGRAMAS

- TXE (Programa de comunicación Sistema de Desarrollo y SDK)
- PLOTEA (Programa que comunica el MDS-221 con el plotter a través del SDK)
- APLOT (Programa que comunica el Sistema de Desarrollo y el plotter a través del PUNCH)
- GRAPHIC5 (Programa realizado en BASIC en el terminal, para tratar la función a dibujar obteniendo la base de datos)
- GRAPHIC3 (Programa de la función: Z=cos(Y/25*3.1415)*100*exp(-SQR(X**2+Y**2))/40)

```
                $PAGELENGTH(41)
PL/M-80  COMPILER


                $EJECT
                /*##############################################################################*/
                          /*PROGRAMA DE FUNCIONAMIENTO DEL PLOTTER A TRAVES DEL SDK*/
                /*##############################################################################*/
  1             PLOTEA:  DO;
  2     1       EXIT:      PROCEDURE EXTERNAL;
  3     2                  END;
  4     1       CO:        PROCEDURE (CHAR) EXTERNAL;
  5     2                    DECLARE CHAR BYTE;
  6     2                  END CO;
  7     1       ERROR:     PROCEDURE(ERRNUM)EXTERNAL;
  8     2                    DECLARE ERRNUM ADDRESS;
  9     2                  END;
 10     1       CI:        PROCEDURE BYTE EXTERNAL;
 11     2                  END CI;
 12     1       WRITE:     PROCEDURE (AFTN,BUFFER,COUNT,STATUS) EXTERNAL;
 13     2                    DECLARE (AFTN,BUFFER,COUNT,STATUS) ADDRESS;
 14     2                  END WRITE;
 15     1       OPEN:      PROCEDURE (AFTN,FILE,ACCESS,MODE,STATUS)EXTERNAL;
 16     2                    DECLARE (AFTN,FILE,ACCESS,MODE,STATUS)ADDRESS;
 17     2                  END OPEN;
 18     1       READ:      PROCEDURE (AFTN,BUFFER,COUNT,ACTUAL,STATUS)EXTERNAL;
 19     2                    DECLARE (AFTN,BUFFER,COUNT,ACTUAL,STATUS)ADDRESS;
 20     2                  END READ;
 21     1       TRANS:     PROCEDURE (LINEA);
 22     2                    DECLARE LINEA BYTE;
 23     2                    DECLARE COM BYTE;
 24     2       L2:        COM=INPUT(0F7H) AND 01H;
 25     2                  IF COM=00H THEN GOTO L2;
 27     2                  OUTPUT(0F6H)=LINEA;
 28     2                  END TRANS;
 29     1       HEXASC:    PROCEDURE (Z) BYTE;
 30     2                    DECLARE Z BYTE;
 31     2                  FLAG=0;
 32     2                  IF (Z>='0')AND (Z<='9')THEN GOTO L3;
 34     2                  IF (Z>='A')AND (Z<='F')THEN GOTO L4;
 36     2                  IF FLAG1=1 THEN GOTO ACA;
 38     2                  IF Z='x' THEN DO;
PL/M-80  COMPILER
```

```
 40    3                    FLAG=2;
 41    3                    GOTO L5;
 42    3                    END;
 43    2                    IF Z='L' THEN DO;
 45    3                    FLAG=3;
 46    3                    GOTO L5;
 47    3                    END;
 48    2                    IF Z='U' THEN DO;
 50    3                    FLAG=4;
 51    3                    GOTO L5;
 52    3                    END;
 53    2       ACA:         FLAG=1;
 54    2                    GOTO L5;
 55    2       L3:          Z=Z AND (0FH);
 56    2                    GOTO L5;
 57    2       L4:          Z=Z AND (0FH);
 58    2                    Z=Z+09H;
 59    2       L5:          RETURN Z;
 60    2                    END HEXASC;
 61    1       ACCFIL:      PROCEDURE (H) BYTE;
 62    2                      DECLARE H BYTE;
 63    2                      IF H>127 THEN DO;
 65    3                      CALL READ(AFT$IN,.BUFFER,128,.ACTUAL,.STATUS);
 66    3                      IF STATUS<>0 THEN GOTO P2;
 68    3                      H=0;
 69    3                      END;
 70    2                      RETURN H;
 71    2                    END ACCFIL;
 72    1       COGEB:       PROCEDURE;
 73    2       AHI:           I=ACCFIL(I);
 74    2                      A=BUFFER(I);
 75    2                      A=HEXASC(A);
 76    2                      I=I+1;
 77    2                      IF FLAG=1 THEN GOTO AHI;
 79    2                    END COGEB;
 80    1       DATA$IN:     PROCEDURE ;
 81    2                    NUM=0;
 82    2                    DO WHILE TECLA<>0DH;
PL/M-80 COMPILER
```

```
 83    3        D1:              FILE(NUM)=CI AND 07FH;
 84    3                           IF FILE(NUM)=07FH THEN DO;
 86    4                                CALL CO(08);
 87    4                                CALL CO (20H);
 88    4                                CALL CO(08);
 89    4                                 NUM=NUM-1;
 90    4                                 GOTO D1;
 91    4                           END;
 92    3                           CALL CO(FILE(NUM));
 93    3                           TECLA=FILE(NUM);
 94    3                           NUM=NUM+1;
 95    3                      END;
 96    2                      NUM=NUM-1;
 97    2                      DO WHILE NUM<=19;
 98    3                        FILE(NUM)=20H;
 99    3                          NUM=NUM+1;
100    3                      END;
101    2                      CALL CO(0DH);
102    2                      CALL CO(0AH);
103    2                      RETURN;
104    2                      END DATA$IN;
105    1        PUNTOS:       PROCEDURE;
106    2                        FLAG1=0;
107    2                        CALL COGEB;
108    2                        FLAG1=1;
109    2                        IF FLAG=2 THEN GOTO L13;
111    2                        IF FLAG=3 THEN NLAPIZ=00;
113    2                        IF FLAG=4 THEN NLAPIZ=0FFH;
115    2                        IF FLAG=0 THEN X1=A;
117    2                        ELSE DO;
118    3                        CALL COGEB;
119    3                        X1=A;
120    3                        END;
121    2                        CALL COGEB;
122    2                        X2=A;
123    2                        X=SHL(X1,4)+X2;
124    2                        CALL COGEB;
125    2                        Y1=A;
PL/M-80  COMPILER
```

```
126   2                    CALL COGEB;
127   2                    Y2=A;
128   2                    Y=SHL(Y1,4)+Y2;
129   2                  END PUNTOS;
130   1                  DECLARE (AFT$IN,STATUS,ACTUAL,AFT$CO)ADDRESS;
131   1                  DECLARE (A,B,J,I,FLAG,OLAPIZ,NLAPIZ,X,Y,X1,X2,Y1,Y2,FLAG1,NUM) BYTE;
132   1                  DECLARE (TECLA) BYTE;
133   1                  DECLARE FILE(10) BYTE ;
134   1                  DECLARE BUFFER (128) BYTE ;
135   1                  OUTPUT(0F3H)=76H;/*PROGRAMACION DE LA USART*/
136   1                  OUTPUT(0F1H)=40H;
137   1                  OUTPUT(0F1H)=00H;
138   1                  OUTPUT(0F7H)=40H;
139   1                  OUTPUT(0F7H)=4EH;
140   1                  OUTPUT(0F7H)=27H;
141   1                  NUM=0;
142   1                  CALL WRITE(0,.('NOMBRE DEL FICHERO?',0DH,0AH),21,.STATUS);
143   1                  IF STATUS<>0 THEN GOTO P2 ;
145   1                  CALL DATA$IN;
146   1        LB:       CALL OPEN(.AFT$IN,.FILE,1,0,.STATUS);/*COMIENZO DEL PROGRAMA*/
147   1                  IF STATUS<>0 THEN GOTO P2;
149   1                  CALL READ(AFT$IN,.BUFFER,128,.ACTUAL,.STATUS);
150   1                  IF STATUS<>0 THEN GOTO P2;
152   1                  I=0;OLAPIZ=0FFH;
154   1        ALLI:     CALL PUNTOS;
155   1                  IF OLAPIZ<>NLAPIZ THEN DO;
157   2                  OLAPIZ=NLAPIZ;
158   2                  CALL TRANS(NLAPIZ);
159   2                  DO J=1 TO 40;
160   3                     CALL TIME(250);
161   3                  END;
162   2                  END;
163   1                  ELSE DO;
164   2                  OLAPIZ=NLAPIZ;
165   2                  CALL TRANS(NLAPIZ);
166   2                  END;
167   1                  CALL TRANS(X);
168   1                  CALL TRANS(Y);
PL/M-80  COMPILER
```

```
 169    1                    DO J=1 TO 27;
 170    2                       CALL TIME(250);
 171    2                    END;
 172    1                    GOTO ALLI;
 173    1         P2:        CALL ERROR(STATUS);
 174    1         L13:       CALL EXIT;
 175    1                  END PLOTEA;


MODULE INFORMATION:

     CODE AREA SIZE     = 035AH    858D
     VARIABLE AREA SIZE = 00A6H    166D
     MAXIMUM STACK SIZE = 00EEH     14D
     160 LINES READ
     0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
```

```
ASM80 TXE PAGELENGTH(41)


ISIS-II 8080/8085 MACRO ASSEMBLER, V4.0          MODULE    PAGE     1


  LOC  OBJ             LINE         SOURCE STATEMENT

                          1 ;*************************************************************************
                          2 ;       PROGRAMA DE TRANSMICION DEL SISTEMA DE DESARROLLO
                          3 ;*************************************************************************
  FC9F                    4  CO         EQU       0FC9FH
  FBBE                    5  CI         EQU       0FBBEH
  FD44                    6  CSTS       EQU       0FD44H
  4000                    7             ORG       4000H
  4000 310050             8             LXI       SP,5000H
  4003 214045             9             LXI       H,4500H
  4006 3600              10             MVI       M,00H
  4008 23                11             INX       H
  4009 36FF              12             MVI       M,0FFH;   POSICION 4501
                         13 ;*************************************************************************
                         14 ;       PROGRAMACION DEL TIMER DE LA USART
                         15 ;*************************************************************************
  400B 3E76              16             MVI       A,76H
  400D D3F3              17             OUT       0F3H
  400F 3E40              18             MVI       A,40H
  4011 D3F1              19             OUT       0F1H
  4013 3E00              20             MVI       A,00H
  4015 D3F1              21             OUT       0F1H
                         22 ;*************************************************************************
                         23 ;       PROGRAMACION DE LA USART
                         24 ;*************************************************************************
  4017 3F40              25             MVI       A,40H
  4019 D3F7              26             OUT       0F7H
  401B 3E4E              27             MVI       A,4EH
  401D D3F7              28             OUT       0F7H
  401F 3E27              29             MVI       A,27H
```

```
LOC  OBJ             LINE        SOURCE STATEMENT

4021 D3F7              30            OUT       0F7H
                       31 ;*****************************************************************************
4023 210441            32 HERE:      LXI       H,OPER; OPERACION A REALIZAR
4026 CDDE40            33            CALL      PRESEN
4029 CD44FD            34 UNO:       CALL      CSTS
402C 0F                35            RRC
402D D22940            36            JNC       UNO
4030 CDBEFB            37            CALL      CI
4033 FE43              38            CPI       'C'; CUADRADA
4035 CA4240            39            JZ        DOS
4038 FE54              40            CPI       'T'; TRIANGULAR
403A CA7540            41            JZ        TRES
403D FE46              42            CPI       'F'; FIN
403F CAD640            43            JZ        FIN
4042 21EA40            44 DOS:       LXI       H,MOR
4045 CDDE40            45            CALL      PRESEN
4048 CDBEFB            46            CALL      CI
404B 57                47            MOV       D,A
404C CDBEFB            48            CALL      CI
404F 5F                49            MOV       E,A
4050 210045            50            LXI       H,4500H; COMIENZO DEL PROGRAMA
4053 4E                51            MOV       C,M
4054 CDC440            52            CALL      TRANS; RUTINA DE TRANSMICION
4057 D5                53 CIN:       PUSH      D
4058 CDCF40            54            CALL      DELAY
405B D1                55            POP       D
405C 23                56            INX       H
405D 7D                57            MOV       A,L
405E FE02              58            CPI       02H
4060 CA8F40            59            JZ        CUAT
```

```
LOC  OBJ             LINE          SOURCE STATEMENT

4063 CD44FD            60          CALL    CSTS
4066 0F                61          RRC
4067 D25740            62          JNC     CIN
406A CDBEFB            63          CALL    CI
406D FE1B              64          CPI     1BH
406F CA2340            65          JZ      HERE
4072 C35740            66          JMP     CIN
4075 21EA40            67 TRES:    LXI     H,MOR
4078 CDDE40            68          CALL    PRESEN
407B CDBEFB            69          CALL    CI
407E 57                70          MOV     D,A
407F CDBEFB            71          CALL    CI
4082 5F                72          MOV     E,A
4083 210145            73          LXI     H,4501H
4086 CDBEFB            74 OCHO:    CALL    CI
4089 FE1B              75          CPI     1BH; ESC
408B CA1340            76          JZ      HERE
408E 4E                77 SEIS:    MOV     C,M
408F CDC440            78          CALL    TRANS
4092 0D                79          DCR     C
4093 79                80          MOV     A,C
4094 FE00              81          CPI     00H
4096 CAA140            82          JZ      SIETE
4099 D5                83          PUSH    D
409A CDCF40            84          CALL    DELAY
409D D1                85          POP     D
409E C38E40            86          JMP     SEIS
40A1 CDBEFB            87 SIETE:   CALL    CI
40A4 FE1B              89          CPI     1BH
40A6 CA2340            89          JZ      HERE
```

```
LOC  OBJ            LINE        SOURCE STATEMENT

40A9 210045           90        LXI     H,4500H
40AC 4E               91 NUEVE: MOV     C,M
40AD CDC440           92        CALL    TRANS
40B0 0C               93        INR     C
40B1 79               94        MOV     A,C
40B2 FEFF             95        CPI     0FFH
40B4 CA8640           96        JZ      OCHO
40B7 D5               97        PUSH    D
40B8 CDCF40           98        CALL    DELAY
40BB D1               99        POP     D
40BC C3AC40          100        JMP     NUEVE
40BF 2B              101 CUAT:  DCX     H
40C0 2B              102        DCX     H
40C1 C35740          103        JMP     CIN
40C4 DBF7            104 TRANS: IN      0F7H; RUTINA DE TRANSMICION
40C6 E601            105        ANI     01H
40C8 CAC440          106        JZ      TRANS
40CB 79              107        MOV     A,C
40CC D3F6            108        OUT     0F6H
40CE C9              109        RET
                     110 ;
40CF 1B              111 DELAY: DCX     D
40D0 7B              112        MOV     A,E
40D1 B2              113        ORA     D
40D2 C2CF40          114        JNZ     DELAY
40D5 C9              115        RET
                     116 ;
40D6 0E46            117 FIN:   MVI     C,'F'
40D8 CDF5FC          118        CALL    CO
40DB C30000          119        JMP     00
```

```
LOC  OBJ            LINE        SOURCE STATEMENT

                     120 ;
40DE 7E              121 PRESEN: MOV     A,M
40DF FE24            122         CPI     '$'
40E1 C8              123         RZ
40E2 4F              124         MOV     C,A
40E3 CD9FFC          125         CALL    CO
40E6 23              126         INX     H
40E7 C3DE40          127         JMP     PRESEN
                     128 ;
40EA 46524543        129 MOR:    DB      'FRECUENCIA A TRANSMITIR'
40FE 55454E43
4102 49412041
4106 20545241
410A 4E534944
410E 5449532
4101 0D              130         DB      0DH,0AH
4102 0A
4103 24              131         DB      '$'
4104 47524146        132 OPER:   DB      'GRAFICAS'
4108 4743341S3
410C 0D              133         DB      0DH,0AH
410D 0A
410E 433D3D20        134         DB      'C = CUADRADA'
4112 43554144
4116 52414441
411A 0D              135         DB      0DH,0AH
411B 0A
411C 543D3D20        136         DB      'T = TRIANGULAR'
4120 54524941
4124 4E475540
```

```
LOC  OBJ            LINE          SOURCE STATEMENT

4128 4152
412A 0D             137           DB        0DH,0AH
412B 0A
412C 46203D20       138           DB        'F = FIN'
4130 46494E
4133 0D             139           DB        0DH,0AH
4134 0A
4135 24             140           DB        '$'
                    141           END


PUBLIC SYMBOLS


EXTERNAL SYMBOLS


USER SYMBOLS
        A FB8E      CIN     A 4057    CO      A FC9F    CSTS    A FD44    CUAT    A 405F    DELAY   A 406F    DOS     A 404
N       A 40D6      HERE    A 4023    MOR     A 40EA    NUEVE   A 40AC    OCHO    A 4086    OPER    A 4104    PRESENT A 40D
S       A 405E      SIETE   A 40A1    TRANS   A 40C4    TRES    A 4075    UNO     A 4029

ASSEMBLY COMPLETE,  NO ERRORS
```

```
                $ PAGELENGTH(41)
PL/M-80  COMPILER


                $ EJECT
                /*#############################################################################*/
                          /*PROGRAMA DE FUNCIONAMIENTO DEL PLOTTER*/
                /*#############################################################################*/
   1            APLOT:  DO;
   2    1       IOSET:     PROCEDURE (CONFIG) EXTERNAL;
   3    2                    DECLARE CONFIG BYTE;
   4    2                  END IOSET;
   5    1       PO:        PROCEDURE (CRTR) EXTERNAL;
   6    2                    DECLARE CRTR BYTE;
   7    2                  END PO;
   8    1       EXIT:      PROCEDURE EXTERNAL;
   9    2                  END EXIT;
  10    1       CO:        PROCEDURE (CHAR) EXTERNAL;
  11    2                    DECLARE CHAR BYTE;
  12    2                  END CO;
  13    1       ERROR:     PROCEDURE(ERRNUM)EXTERNAL;
  14    2                    DECLARE ERRNUM ADDRESS;
  15    2                  END;
  16    1       CI:        PROCEDURE BYTE EXTERNAL;
  17    2                  END CI;
  18    1       WRITE:     PROCEDURE (AFTN,BUFFER,COUNT,STATUS) EXTERNAL;
  19    2                    DECLARE (AFTN,BUFFER,COUNT,STATUS) ADDRESS;
  20    2                  END WRITE;
  21    1       OPEN:      PROCEDURE (AFTN,FILE,ACCESS,MODE,STATUS)EXTERNAL;
  22    2                    DECLARE (AFTN,FILE,ACCESS,MODE,STATUS)ADDRESS;
  23    2                  END OPEN;
  24    1       READ:      PROCEDURE (AFTN,BUFFER,COUNT,ACTUAL,STATUS)EXTERNAL;
  25    2                    DECLARE (AFTN,BUFFER,COUNT,ACTUAL,STATUS)ADDRESS;
  26    2                  END READ;
  27    1       HEXASC:    PROCEDURE (Z) BYTE;   /*CONVERSON ASCII-HEXADECIMAL*/
  28    2                    DECLARE Z BYTE;
  29    2                  FLAG=0;
  30    2                  IF (Z>='0')AND (Z<='9')THEN GOTO L3;
  32    2                  IF (Z>='A')AND (Z<='F')THEN GOTO L4;
  34    2                  IF Z=0DH THEN DO;
  36    3                  FLAG=5;
  37    3                  GOTO L5;
PL/M-80  COMPILER
```

```
 38    3                   END;
 39    2                   IF Z='X' THEN DO;
 41    3                   FLAG=2;
 42    3                   GOTO L5;
 43    3                   END;
 44    2                   IF Z='L' THEN DO;
 46    3                   FLAG=3;
 47    3                   GOTO L5;
 48    3                   END;
 49    2                   IF Z='U' THEN DO;
 51    3                   FLAG=4;
 52    3                   GOTO L5;
 53    3                   END;
 54    2                   FLAG=1;
 55    2                   GOTO L5;
 56    2         L3:       Z=Z AND (0FH);
 57    2                   GOTO L5;
 58    2         L4:       Z=Z AND (0FH);
 59    2                   Z=Z+09H;
 60    2         L5:       RETURN Z;
 61    2                   END HEXASC;
 62    1         ACCFIL:   PROCEDURE (H) BYTE;  /*COMPROBACION DE FINAL DE FICHERO*/
 63    2                     DECLARE H BYTE;
 64    2                   IF H>127 THEN DO;
 66    3                   CALL READ(AFT$IN,.BUFFER,128,.ACTUAL,.STATUS);
 67    3                   IF STATUS<>0 THEN GOTO P2;
 69    3                   H=0;
 70    3                   END;
 71    2                   RETURN H;
 72    2                   END ACCFIL;
 73    1         COGEB:    PROCEDURE;  /*TOMA DE DATOS*/
 74    2         AHI:      I=ACCFIL(I);
 75    2                   A=BUFFER(I);
 76    2                   A=HEXASC(A);
 77    2                   I=I+1;
 78    2                   IF FLAG=5 THEN GOTO AHI;
 80    2                   IF (FLAG1=1) AND((FLAG=1) OR (FLAG=0)) THEN GOTO AHI;
 82    2                   END COGEB;
PL/I-80 COMPILER
```

```
 83    1        DATA$IN:   PROCEDURE ;   /*ELECCION DEL FICHERO*/
 84    2                   NUM=0;
 85    2                   DO WHILE TECLA<>0DH;
 86    3        D1:            FILE(NUM)=CI AND 07FH;
 87    3                        IF FILE(NUM)=07FH THEN DO;
 89    4                              CALL CO(08);
 90    4                              CALL CO (20H);
 91    4                              CALL CO(08);
 92    4                              NUM=NUM-1;
 93    4                              GOTO D1;
 94    4                         END;
 95    3                       CALL CO(FILE(NUM));
 96    3                       TECLA=FILE(NUM);
 97    3                       NUM=NUM+1;
 98    3                       END;
 99    2                    NUM=NUM-1;
100    2                    DO WHILE NUM<=19;
101    3                     FILE(NUM)=20H;
102    3                     NUM=NUM+1;
103    3                    END;
104    2                    CALL CO(0DH);
105    2                    CALL CO(0AH);
106    2                    RETURN;
107    2                  END DATA$IN;
108    1        PUNTOS:   PROCEDURE;
109    2                  FLAG1=1;
110    2                  CALL COGEB;
111    2                  FLAG1=0;
112    2                  IF FLAG=2 THEN GOTO L13;
114    2                  IF FLAG=3 THEN NLAPIZ=30;
116    2                  IF FLAG=4 THEN NLAPIZ=031H;
118    2                  CALL COCEB;
119    2                  X1=A;
120    2                  CALL COGEB;
121    2                  X2=A;
122    2                  X=SHL(X1,4)+X2;
123    2                  CALL COGEB;
124    2                  Y1=A;
PL/M-80  COMPILER
```

```
125   2                 CALL COGEB;
126   2                 Y2=A;
127   2                 Y=SHL(Y1,4)+Y2;
128   2                 END PUNTOS;
129   1                 DECLARE (AFT$IN,STATUS,ACTUAL,AFT$CO)ADDRESS;
130   1                 DECLARE (A,B,J,I,FLAG,OLAPIZ,NLAPIZ,X,Y,X1,X2,Y1,Y2,FLAG1,NUM) BYTE;
131   1                 DECLARE (TECLA) BYTE;
132   1                 DECLARE FILE(10) BYTE ;
133   1                 DECLARE BUFFER (128) BYTE ;
                        /*COMIENZO DEL PROGRAMA*/
134   1                 CALL WRITE(0,.('NOMBRE DEL FICHERO?',0DH,0AH),21,.STATUS);
135   1                 IF STATUS<>0 THEN GOTO P2 ;
137   1                 CALL DATA$IN;
138   1                 CALL IOSET(0C5H);
139   1       L3:       CALL OPEN(.AFT$IN,.FILE,1,0,.STATUS);
140   1                 IF STATUS<>0 THEN GOTO P2;
142   1                 CALL READ(AFT$IN,.BUFFER,128,.ACTUAL,.STATUS);
143   1                 IF STATUS<>0 THEN GOTO P2;
145   1                 I=0;OLAPIZ=0FFH;
147   1       ALLI:     CALL PUNTOS;
148   1                 IF OLAPIZ<>NLAPIZ THEN DO;
150   2                 CALL PO(NLAPIZ);
151   2                 DO J=1 TO 40;
152   3                    CALL TIME(250);
153   3                    END;
154   2                 END;
155   1                 ELSE DO;
156   2                 CALL PO(NLAPIZ);
157   2                 END;
158   1                 OLAPIZ=NLAPIZ;
159   1                 CALL PO(X);
160   1                 CALL PO(Y);
161   1                 DO J=1 TO 15;
162   2                    CALL TIME(250);
163   2                 END;
164   1                 CALL PO(0);
165   1                 GOTO ALLI;
166   1       P2:       CALL ERROR(STATUS);
PL/M-80 COMPILER
```

```
 167    1       L13:        CALL EXIT;
 168    1                 END APLOT;


MODULE INFORMATION:

     CODE AREA SIZE      = 0343H     835D
     VARIABLE AREA SIZE  = 00A4H     164D
     MAXIMUM STACK SIZE  = 000EH      14D
     155 LINES READ
     0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
```

```
GRAPHICS
   10 DEF FNT(L9,X8,Y8,X9,Y9)
   15   REM SUBRUTINA RECURSIVA
   20   IF L9=3 THEN DO
   25   REM VUELCA LAS VARIABLES Y HACE UN END OF FILE
   30                A9=1
   40                O9$=O9$+"X"
   50                GOSUB 1020
   60                PRINT #1;O9$
   70                GOTO 1100
   80                 DOEND
   90   IF L9=2 THEN DO
   95   REM INICIACION DE LAS VARIABLES
  100                S9=0
  110                L8=0
  120                O9$=""
  130                H8=J8=0
  140                GOTO 1100
  150                 DOEND
  160   IF X8<>H8 OR Y8<>J8 THEN DO
  165   REM OPTIMIZACION DEL PROGRAMA
  170      Z9=FNT(1,H8,J8,X8,Y8)
  180                                DOEND
  190   IF L8=0 AND L9=1 THEN DO
  195   REM CONTROL DEL LAPIZ
  200                           A9=1
  210                           O9$=O9$+"U"
  220                           GOSUB 1020
  230                           A9=2
  240                           T8=X8
  250                           Z8=Y8
  260                           GOSUB 920
  270                            DOEND
  280   IF L8=1 AND L9=0 THEN DO
  285   REM CONTROL DEL LAPIZ
  290                           A9=1
  300                           O9$=O9$+"L"
  310                           GOSUB 1020
```

```
329                             A9=2
330                             T8=X8
340                             Z8=Y8
350                             GOSUB 920
360                              DOEND
370     L8=L9
380     IF L9=0 THEN DO
385     REM TRAZADO DE RECTAS
390        IF SQR((X9-X8)**2+(Y9-Y8)**2)>20 THEN DO
395        REM OPTIMIZACION DE LA DISTANCIA
400                                     Z9=FNT(0,X8,Y8,(X9-X8)/2+X8,(Y9-Y8)/2+Y8)
410                                     Z9=FNT(0,(X9-X8)/2+X8,(Y9-Y8)/2+Y8,X9,Y9)
420                                          DOEND
430                                          ELSE DO
440                                               A9=1
450                                               O9$=O9$+"L"
460                                               GOSUB 1020
470                                               A9=2
480                                               T8=X9
490                                               Z8=Y9
500                                               GOSUB 920
510                                                DOEND
520                      DOEND
530     IF L9=1 THEN DO
535     REM VA DE (X8,Y8) A (X9,Y9) CON LAPIZ ARRIBA
540                   A9=1
550                   O9$=O9$+"()"
560                   GOSUB 1020
570                   A9=2
580                   T8=X9
590                   Z8=Y9
600                   GOSUB 920
610                    DOEND
620     H8=X9
630     J8=Y9
640     GOTO 1100
```

```
650   REM SUBRUTINA CONVERSION DECIMAL HEXADECIMAL
660   F9=0
670   A9$=""
680   V8=INT(V8)
690   V9=V8
700   R9=INT(V9/16)
710   W9=((V9/16)-R9)*16
720   REM GENERACION DE RESTO
730   IF W9>9 THEN DO
740                  IF W9=10 THEN B9$="A"
750                  IF W9=11 THEN B9$="B"
760                  IF W9=12 THEN B9$="C"
770                  IF W9=13 THEN B9$="D"
780                  IF W9=14 THEN B9$="E"
790                  IF W9=15 THEN B9$="F"
800                   DOEND
810                   ELSE CONVERT W9 TO B9$
820   A9$=B9$+A9$
830   IF R9>16 THEN DO
840                   V9=R9
850                   GOTO 700
860                    DOEND
870   IF F9=1 THEN 910
880   F9=1
890   W9=R9
900   GOTO 720
910   RETURN
920   REM SUBRUTINA DE FORMATEADO DE VARIABLES
930   V8=T2
940   GOSUB 650
950   O9$=O9$+A9$
960   GOSUB 1020
970   V8=Z2
980   GOSUB 650
990   O9$=O9$+A9$
1000  GOSUB 1020
1010  RETURN
1020  REM SUBRUTINA DE ESCRITURA EN FICHERO
1030  S9=S9+A9
1040  IF S9>=60 THEN DO
```

```
1050                    PRINT #1;O9$
1060                    O9$=""
1070                    S9=0
1080                     DOEND
1090    RETURN
1100    RETURN 1
1110 FNEND
```

```
**********************************************************************
                    PROGRAMA DE LA FUNCION
              Z= COS(Y/25*3.1415)*100*EXP(-(SQR(X**2+Y**2))/40
**********************************************************************
GRAPHIC3
 1120 DIM A9$[2],B9$[2],C9$[80]
 1125 REM NOMBRE DE LA BASE DE DATOS
 1130 FILES GRAPH1
 1135 REM TRAZADO DE LOS EJES
 1140 K=FNT(2,0,0,0,0)
 1150 K=FNT(1,0,0,0,0)
 1160 K=FNT(0,0,0,128,128)
 1170 K=FNT(0,120,128,255,128)
 1180 K=FNT(0,128,128,128,255)
 1190 X2=Y2=0
 1200 L=1
 1210 FOR X=0 TO 100 STEP 5
 1220   FOR Y=0 TO 100 STEP 5
 1230     GOSUB 1380
 1240     L=0
 1250   NEXT Y
 1260   L=1
 1270 NEXT X
 1280 L=1
 1290 FOR Y=0 TO 100 STEP 5
 1300   FOR X=0 TO 100 STEP 5
 1310     GOSUB 1380
 1320     L=0
 1330   NEXT X
 1340   L=1
 1350 NEXT Y
 1360 K=FNT(3,0,0,0,0)
 1370 STOP
 1375 REM FUNCION A DIBUJAR
 1380 Z=COS(Y/25*3.1415)*100*EXP(-(SQR(X**2+Y**2))/40)
 1385 REM TRANSFORMACION DE DIMENCIONES
 1390 X1=128-X*(.707)+Y
 1400 Y1=128-X*(.707)+Z
 1410 IF X1>256 OR X1<0 OR Y1>256 OR Y1<0 THEN 1430
```

```
1420 K=FNT(L,X2,Y2,X1,Y1)
1430 X2=X1
1440 Y2=Y1
1450 RETURN
```

_ BIBLIOGRAFIA _

- DAVID F. ROGERS, J. ALAN ADAMS.

  "MATHEMATICAL ELEMENTS FOR COMPUTER GRAPHICS "

- ESCUELA T. S. INGENIEROS DE TELECOMUNICACION-MADRID.

  " FUNDAMENTOS DEL ANALISIS Y DISEÑO DE CIRCUITOS AYUDADOS POR ORDENADOR "

- JOSE LUIS PEREZ BAEZA(Articulo del mundo electrónico).

  " REPRESENTACION PERSPECTIVA DE FUNCIONES DE DOS VARIABLES "

- LOUIS HOHENSTEIN.

  " COMPUTER PERIPHERALS FOR MINICOMPUTERS, MICROPROCESSORS, AND PERSONAL COMPUTER "

- MANUAL DE OPERACION DEL PLOTTER X-Y PM8043(PHILIPS).

- YORAM KOREN.

  " COMPUTER CONTROL OF MANUFACTURING SYSTEMS "