UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DISEÑO Y CONSTRUCCION DEL SISTEMA DE INTERFAZ PARA LA CAMARA DE EXTRAPOLACION DEL PATRON SECUNDARIO BETA

T E S I S

QUE PARA OBTENER EL TITULO DE :

INGENIERO MECANICO ELECTRICISTA

(AREA ELECTRICA - ELECTRONICA)

P R E S E N T A :

LUIS FERNANDO JIMENEZ CRUZ

Director : Ing. Fco. Javier Ramírez Jiménez
Codirector : Ing. Roberto Macías Pérez

México, D. F. Enero 1995.

DEDICATORIA

A mi madre, María Luisa Cruz, como homenaje póstumo a la persona que me enseño las primeras letras y puso los simientos para que yo terminará una carrera...

A mi padre, Fernando P. Jiménez, que con su ejemplo siempre me ha enseñado el valor del trabajo honesto, la constancia, la responsabilidad y el servico a los demás como base de una vida feliz ....

"El que no vive para servir no sirve para vivir . . ."

AGRADECIMIENTOS

A mi padre que siempre me ha apoyado ...

A mis tías Silvia y Lupita que siempre me han brindado su ayuda y han creido en mí a pesar de todo..

Al Instituto Nacional de Investigaciones Nucleares por brindarme la oportunidad de desarrollar el presente trabajo en sus instalaciones ...

Al Ing. Francisco Javier Ramírez, Jefe del Departamento de Diseños Especiales por su apoyo y asesoría . . .
A mis compañeros en el departameato de Diseños Especiales por brindarme su amistad y consejo..

Al personal del Taller de Electrónica de la Gerencia de Ingeniería quíen se encargo de la realizació física del proyecto . . .

A mis maestros, que desde la primaria hasta la Facultad colaboraron en mi formación . . .

A la Facultad de Ingeniería formadora de profesionales de la Ingeniería concientes de la realidad social del país y con la preparación adecuada para enfreatar el reto de sacar a México adelante,Institución de la cual soy orgulloso egresado. . .

"Lo más importante no es donde estas, si no hacia a donde te diriges ..."

SISTEMA DE INTERFAZ PARA LA CÁMARA DE EXTRAPOLACIÓN DEL PATRÓN SECUNDARIO BETA

ÍNDICE

1.- INTRODUCCIÓN

1.1 ANTECEDENTES	1
1.2 LA CÁMARA DE EXTRAPOLACIÓN	2
1.3 DESCRIPCIÓN DEL PROCESO DE MEDICIÓN	2
1.4 OBJETIVO DEL SISTEMA A DESARROLLAR	4

2. DISEÑO

2.1 ESPECIFICACIONES DEL SISTEMA	5
2.2 DIAGRAMA A BLOQUES	5
2.3 DISEÑO DEL CIRCUITO PARA MANEJAR LA FUENTE	7
2.3.1 PRINCIPIO DE OPERACIÓN DE LA FUENTE DE VOLTAJE	8
2.3.2 SOLUCIONES POSIBLES PARA AJUSTAR EL VOLTAJE	9
2.3.3 DISEÑO DEL CIRCUITO COMBINACIONAL PARA LA REFERENCIA PROGRAMABLE	11
2.4 DISEÑO DEL CIRCUITO PARA MOVER EL MOTOR DE PASOS	17
2.4.1 EL MOTOR DE PASOS	17
2.4.2 DISEÑO DEL CIRCUITO	17
2.5 IMPLEMETACIÓN DE LOS CIRCUITOS LÓGICOS	
2.5.1 OPCIONES PARA EL DISEÑO DE CIRCUITOS DIGITALES	23
2.5.2 SELECCIÓN DE LA ALTERNATIVA PARA EL DISEÑO	25
2.6 PPI Y DIRECCIONAMIENTO	28
2.7 FUENTES DE POLARIZACIÓN	
2.7.1 REQUERIMIENTOS DE POTENCIA	29
2.7.2 CÁLCULO DE COMPONENTES	30
2.8 ADQUISICIÓN DE LA INFORMACIÓN	
2.8.1 EL GPIB IEEE488	35
2.8.2 CONVERTIDOR ANALÓGICO DIGITAL	39
2.8.3 ELECTRÓMETRO	41
2.9 PROGRAMACIÓN	
2.9.1 DIVISION GENERAL DEL PROGRAMA	43
2.9.2 ADQUISICIÓN DE INFORMACIÓN	43
2.9.2.1 RUTINA CURVAS DE EXTRAPOLACIÓN	43
2.9.9.2 RUTINA OTRAS APLICACIONES	45
2.9.3 RUTINA AUTOTEST	45
2.9.4 RUTINA IMPRIMIR INFORMACIÓN	52

3. CONSTRUCCION	60
3.1 CIRCUITOS IMPRESOS	
3.1.1 TARJETA BETA	60
3.1.2 TARJETA MOTDRV	62
3.1.3 TARJETA FUENDRV	62
3.1.4 TARJETA CONTHV	63
3.2 GABINETE Y ARREGLO MECÁNICO	
3.2.1 GABINETE	64
3.2.2 ARREGLO MECÁNICO	65

4. MONTAJE Y PRUEBAS
4.1 MONTAJE MECÁNICO 68
4.2 CONEXIÓN DEL SISTEMA 69
4.3 PRUEBAS
4.3.1 PRUEBA DE VOLTAJE 73
4.3.2 PRUEBAS DE DESPLAZAMIENTO 78
4.3.2.1 PRUEBA POR COMPARACIÓN DIRECTA 78
4.3.2.2 MEDICIÓN DE DESPLAZAMIENTO POR MÉTODO CAPACITIVO 83

5. RESULTADOS Y CONCLUSIONES
5.1 RESULTADOS 85
5.2 CONCLUSIONES 86

6. REFERENCIAS 88

7. ANEXOS
A) TARJETAS DEL SISTEMA
A1.- DIBUJOS DE LOS CIRCUITOS IMPRESOS
A2.- LISTA DE COMPONENTES
A3.- DIAGRAMAS ELÉCTRICOS
B) SISTEMA MECÁNICO
C) LISTADO DEL PROGRAMA
D) HOJAS DE ESPECIFICACIÓN DE CIRCUITOS UTILIZADOS
E) MANUAL DE OPERACIÓN

RESUMEN

El Sistema de Interfaz para la Cámara de Extrapolación (SICE) es un sistema integrado por varios dispositivos interconectados a una computadora personal (PC), que en conjunto, permite la adquisición de información para la determinación de la dosis absorbida por tejido al estar expuesto a un campo de radiación beta.

Las funciones que realiza son:

a) Mide la corriente de ionización o la carga almacenada entre las placas de la cámara de extrapolación.
b) Ajusta automáticamente la distancia entre las placas de la cámara de extrapolación.
c) Ajusta automáticamente el voltaje para polarizar la cámara de extrapolación.
d) Toma información de la temperatura, presión y humedad relativa del ambiente, así como del voltaje aplicado entras las placas de la cámara de extrapolación.
e) Determina el área efectiva de las placas de la CE y la distancia real cuando el dial del tornillo micrométrico marca cero.
f) Guarda la información obtenida en medios de almacenamiento magnético, sean estos fijos o removibles.

Haciendo la comparación entre la distancia propuesta y la distancia que marca el dial de la cámara de extrapolación, se determinó que por cada paso del motor las placas de la cámara se desplazan 0.6 micrómetros, considerando una tolerancia de 9% se considera que el intervalo mínimo del desplazamiento de las placas es de 20 micras, mientras que en el voltaje se puede manejar un intervalo de -399.9 V a +399.9V con un error del 3% teniendo la capacidad de incrementar el voltaje en pasos de 0.1V como mínimo. Estos valores de precisión y exactitud del sistema son aceptables para su utilización en la determinación de la dosis absorbida absoluta impartida por fuentes de radiación beta.

1 INTRODUCCIÓN

El Sistema de Interfaz para la Cámara de Extrapolación (SICE) es un sistema integrado por varios dispositivos interconectados a una computadora persona (PC), que en conjunto, permite la adquisición de información para la determinación de la dosis absorbida por tejido al estar expuesto a un campo de radiación beta.
El sistema lo componen:

- Cámara de extrapolación.
- Convertidor Analógico Digital **ADC/16 Iotech** con puerto de comunicación IEEE488.
- Módulo **Digital-80** con puerto de comunicación IEEE488.
- Interfaz para la cámara de extrapolación.
- Tarjeta beta.
- Módulo de medición de factores ambientales.
- Electrómetro Keithley Mod. 617.
- **Motor de pasos con acoplamiento mecánico.**
- **Obturador temporizado Buchler.**
- **Fuente de voltaje HP mod. 6160A**
- **Computadora personal.**
- **Programa de operación BETAI488**

Las funciones que realiza son:

a) Mide la corriente de ionización o la carga almacenada entre las placas de la cámara de extrapolación.
b) Ajusta automáticamente la distancia entre las placas de la cámara de extrapolación.
c) Ajusta automáticamente el voltaje para polarizar la cámara de extrapolación.
d) Toma información de la temperatura, presión y humedad relativa del ambiente, así como del voltaje aplicado entre las placas de la cámara de extrapolación.
e) Determina el área efectiva de las placas de la CE y la distancia real cuando el dial del tornillo micrométrico marca cero.
f) Guarda la información obtenida en medios de almacenamiento magnético, sean estos fijos o removibles.

1.1 ANTECEDENTES

Las radiaciones ionizantes tienen una característica particular, no impresionan a nuestros sentidos, esto representa un problema a considerar, pues si el organismo humano es expuesto a niveles de radiación por encima de los límites establecidos, puede ocasionar daños . La medición de dosis de radiación tiene la función de determinar la cantidad de energía que absorbe un tejido, para determinar el daño que puede ocasionar a las células humanas la exposición a las radiaciones de una fuente radiactiva.
Considerando que en muchos lugares se utilizan fuentes radiactivas tanto para fines terapéuticos como industriales, es necesario cuantificar la magnitud de la dosis que imparten dichas fuentes.
En el Instituto Nacional de Investigaciones Nucleares (ININ) se cuenta con el Centro de Metrología de Radiaciones Ionizantes (CMRI), el cual tiene salas dedicadas a la calibración de equipos de medición en algún tipo de radiación en especial, una de ellas es la sala de betas, en ella se encuentra una cámara de extrapolación (CE). La cámara de extrapolación es un detector de radiación que trabaja bajo el principio de la cámara de ionización de gases, la CE puede detectar radiaciones del tipo beta y rayos X de baja energía.
La CE como instrumento de medición (y la instrumentación asociada) tiene varias aplicaciones, entre ellas podemos mencionar:

a) Medición de campos de radiación beta externos con fines de protección radiológica; principalmente en actividades de mantenimiento de plantas nucleares.
b) Medición de la dosis absorbida en tejido debido a fuentes de radiación beta.
c) Como patrón primario para la reproducción de la unidad de dosis.

1.2 LA CÁMARA DE EXTRAPOLACIÓN

La CE es un instrumento para la detección de la radiación que funciona bajo el principio de las cámaras de ionización de gases (descrito en párrafos anteriores); dicho instrumento permite medir incrementos de corriente de ionización DI, en función del incremento de masa de aire de un volumen de colección bajo condiciones Bragg-Gray [1]; la menor desviación de las condiciones Bragg-Gray se logra al extrapolar el cociente $\Delta I/\Delta m$ a profundidad cero de la cámara.

de tal modo que :

$$\dot{D}_t = s_{t,a}\frac{W}{e}\left(\frac{\Delta I}{\Delta m}\right)_{BGC} \qquad (1)$$

donde:

$s_{t,a}$: es la relación de la masa promedio de colisión de frenado para el tejido y aire sobre la densidad de flujo de partículas beta;
W: es la energía promedio necesaria para producir un par de iones en el aire.
e: es el valor de la carga elemental.
$(\Delta I / \Delta m_a)_{BGC}$: es el valor límite del cociente del incremento de la corriente de ionización , ΔI, entre el incremento de la masa en la cámara, Δm , obtenido bajo condiciones Bragg-Gray (BGC).

1.2.1 DESCRIPCIÓN DE LA CÁMARA DE EXTRAPOLACIÓN

En la fig. 1 se muestra un corte transversal de la cámara de extrapolación, el volumen de colección está limitado por el electrodo de colección (6) y la ventana de entrada (7) que es una delgada película de "Hostaphan" con un revestimiento grafitado. La película se mantiene estirada gracias a un arillo colocado sobre el encapsulado. La ventana se conecta al potencial de polarización vía un conector BNC (2); y la corriente de ionización se mide conectando un electrómetro al conector (8).La profundad del volumen de colección se varía cambiando la distancia entre la ventana y el electrodo de colección girando el tornillo micrométrico que a su vez desplaza el electrodo de colección, el intervalo de desplazamiento es de 0 a 20 mm con una resolución de 5 μm.

1.3 DESCRIPCIÓN DEL PROCESO DE MEDICIÓN

De la ecuación 1, para obtener la dosis absorbida necesitamos determinar el incremento en la corriente de ionización cuando se tiene la fuente radiactiva presente y el incremento de la masa en el volumen de colección, por lo que las variables a medir son la corriente de ionización de la CE y la masa del volumen de colección, si las

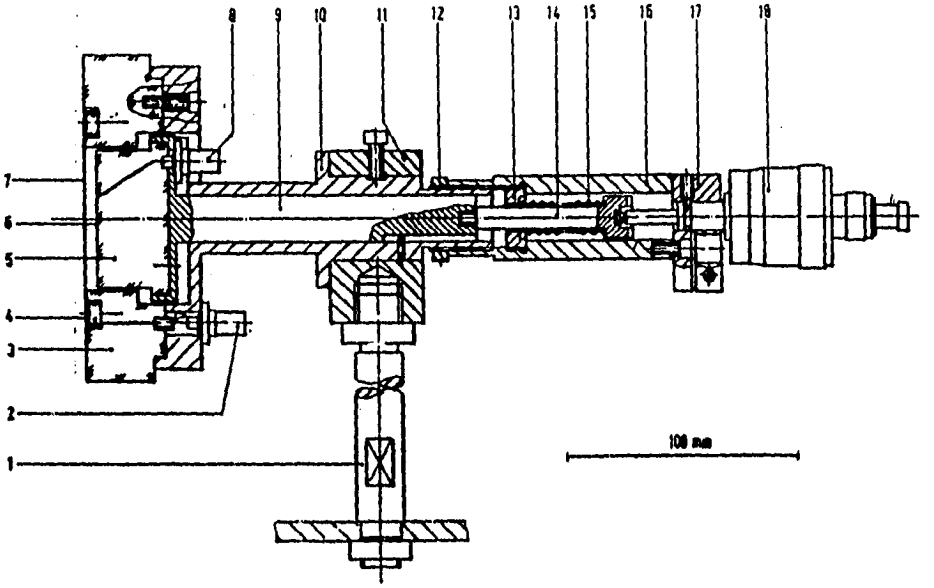


Fig. 1 Corte transversal de la cámara deextrapolación 1. Soporte; 2. enchufe para el voltaje de polarización; 3.caja de perspex; 4. Arillo de tensión; 5. Bloque de perspex; 6. Superficie grafitada en el electrodo de colección y el arrillo de guarda; 7. Ventana de entrada; 8. Enchufe para el electrodo de colección; 9. Varilla deslizante de ajuste; 10; Guia cilindrica para la varilla 9; 11. Sujetador; 12. Tuerca ajustable; 13. anillo roscado; 14. flecha; 15. Resorte; 16. Tubo; 17. pieza de sujeción; Tornillo micrométrico.

dimensiones de la cámara son fijas, a excepción de la profundidad, podemos determinar el valor de la masa en el volumen de colección; la masa de la que estamos hablando en este caso es aire, por lo que tenemos que considerar los factores ambientales para determinar con precisión la masa de aire dentro de la cámara de extrapolación, en resumen las variables que necesitamos conocer son:

De la cámara: la distancia entre placas y la corriente de ionización.
Del ambiente: temperatura, presión atmosférica y humedad relativa.

En al fig. 2 se muestra el diagrama a bloques del arreglo para medir la dosis absorbida; los pasos que se siguen para obtener los valores de las variables de interes, se describen a continuación.

1) Revisar las conexiones entre los elementos del sistema y encender la fuente de polarización y los instrumentos de medición para darles tiempo a que se estabilicen.
2)medir la temperatura, presión y humedad ambientales y registrarlos en la bitácora.
3)fijar la distancia entre placas de la cámara de extrapolación moviendo el tornillo micrométrico.
4)ajustar el voltaje de polarización de la cámara.
5)fijar la función del electrómetro e intervalo de medición.
6) Medir la corriente de fondo (sin fuente radiactiva).
7)Abrir la fuente y activar el electrómetro para que tome varias lecturas.
8)Una vez transcurrido el tiempo de exposición leer manualmente la memoria del electrómetro y registrar las lecturas.
9) Se invierte la polaridad de la fuente y se repiten los pasos del 6 al 8.
10) Se cambia la distancia entre placas del electrómetro y se repiten los pasos del 3 al 9.

11) Con la información recabada se determinan los factores de corrección y se procesan estadísticamente los valores de corriente obtenidos.
12) Se elabora un informe con los resultados obtenidos.

Como se puede apreciar muchas de las tareas descritas son repetitivas y requieren de toda la atención de la persona encargada de hacer la labor; si el proceso se automatiza podemos obtener resultados más rápido y podemos ayudar a que muchas tareas que se hacen manualmente se hagan automáticamente.

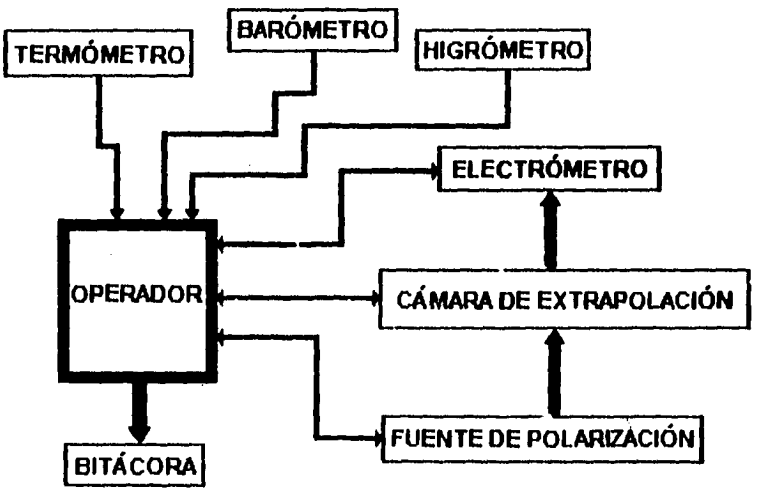


Fig. 2 Diagrama a bloques del sistema sin automatizar.

1.4 OBJETIVO DEL SISTEMA A DESARROLLAR

Diseñar e implementar un sistema para automatizar el proceso de medición de dosis utilizando como detector una cámara de extrapolación, para el CMRI el cual comprenda:

a) Medir la corriente de ionización de la cámara de extrapolación.
b) Ajustar automáticamente la distancia entre placas de la cámara.
c) Ajustar automáticamente la fuente de voltaje para polarizar la cámara de extrapolación.
c) Tomar la información de la temperatura, presión atmosférica y humedad relativa del ambiente.
d) Guardar la información obtenida para su uso posterior.

2 DISEÑO DEL SISTEMA

2.1 ESPECIFICACIONES DEL SISTEMA

Las características que debe cumplir el sistema fueron propuestas por el usuario en base a sus necesidades, estableciendo las siguientes características:

a) Utilizar los recursos ya disponibles y que a continuación se describen:

1) Cámara de extrapolación .
   -Máxima profundidad 25 000 micras.
   -Resolución del tornillo micrométrico 5 micras.
2) Electrómetro Programable Keithley Mod. 617.
   -funciones de medición: voltaje, resistencia, corriente y carga eléctrica.
   -Puerto de comunicación bajo la norma IEEE488.
   -El instrumento puede ser programado a través de este puerto.
   -Intervalo: voltaje, de 10 microV a 200V; corriente de 0.1 fA a 20 mA; resistencia de 0.1ohm a 200 G ohm; carga de 2 fC a 20 nC.
3) Fuente de alto voltaje Hewlett Packard mod. 6061A.
   -Intervalo de 0 a 3000 V y 0 a 6 mA.
   -Resolución 0.1 V.
   -Estabilidad: menos del 0.05 % +/- 5 mV de corrimiento total en 8 horas despúes de 30 min. de calentamiento y con 3° C de variación en la temperatura ambiente.
4) Termómetro digital.
5) Barómetro digital.
6) Transductor de humedad relativa.
7) Convertidor analógico digital con interfaz IEEE488.
8) Módulo digital con interfaz IEEE488.
8) computadora personal.

b) El sistema debe ser capaz de programar la fuente de alto voltaje de -400 a 400 V con una resolución de 0.1 V, generando una rampa para el incremento o decremento del voltaje.
c) La distancia entre placas se debe poder ajustar entre 0 y 25000 micras con una resolución de 5 micras al menos.
d) Tomar información ambiental de la temperatura, presión atmosférica y humedad relativa.
e) Tener una forma de calibrar la distancia entre placas.
f) Asegurar que se esté aplicando el voltaje adecuado.
g) Tomar información del electrómetro.
h) Grabar la información adquirida en un medio de almacenamiento magnético para su posterior utilización.
i) Visualizar la información adquirida en pantalla y si así se desea imprimirla o enviarla a un archivo de tipo texto.

2.2 DIAGRAMA A BLOQUES

Una vez asimilado como se realiza el proceso para medir la dosis absorbida y que es lo que se requiere para cumplir con el objetivo, se propone un diagrama a bloques del sistema automático de medición, en la fig. 3 se muestran los bloques y la forma en que interactúan, la idea general es tener un sistema sencillo constituído por componentes que se consiguen comercialmente y que ya se tienen; y diseñar los circuitos necesarios para adaptar

aquellos dispositivos que por sus características no puedan ser "conectados directamente" al sistema. En los párrafos siguientes se describe cada bloque.

2.2.1 CONTROL CENTRAL

El control central se encarga de generar las palabras de control para operar la interfaz IEEE488 recibir y enviar información para manejar el electrómetro y el ADC, también envía las palabras de control para manejar la fuente de alto voltaje y el motor de pasos, el control central lo constituye la computadora personal (PC) y es el medio de comunicación entre el sistema y el usuario permitiendole al mismo dar las instrucciones pertinentes al sistema para que opere de acuerdo a las instrucciones del programa desarrollado.

2.2.2 PPIO

Este bloque es un circuito que funciona como interfaz entre el manejador del motor y de la fuente de alto voltaje y la computadora reteniendo las palabras de control para la adecuada operación de los manejadores.

2.2.2 INTERFAZ IEEE488

La interfaz IEEE488 es un puerto paralelo de aplicación general para comunicación de instrumentos con computadoras personales que trabaja bajo la norma IEEE488, esta interfaz permite tener conectados hasta 15 intrumentos en el mismo "bus" y se pueden tener conectados instrumentos hasta una distancia de 10 m. esta interfaz nos permite comunicar a la PC con el electrómetro y con el convertidor analógico digital.

2.2.3 CONVERTIDOR ANALOGICO DIGITAL

El Convertidor Analógico Digital (ADC) es un componente comercial que va a tomar las señales analógicas entregadas por el módulo de medición de factores ambientales y las va a codificar para que puedan ser accesibles a la computadora, cuenta con un puerto de comunicación bajo la norma IEEE488, lo que facilita la comunicación entre el ADC y la computadora.

2.2.4 MODULO DE MEDICIÓN DE FACTORES AMBIENTALES

Este módulo cuenta con transductores de temperatura, presión atmosférica y humedad relativa, la señal generada por estos es acondicionada y se presenta cada factor como una señal de voltaje donde la magnitud del voltaje es proporcional al valor de la variable en cuestión, cada una de estas salidas se conecta a un canal del ADC permitiendo al control central conocer el valor de los factores ambientales, cuando el sistema lo requiera.

2.2.5 ELECTRÓMETRO

El electrómetro es el intrumento de medición que mide la corriente de ionización de la cámara de extrapolación proporcionandonos el valor de una de las variables de interés, la corriente de ionización o en el caso que se desee calibrar la cámara, la carga almacenada; este instrumento se comunica con el control central por medio de un puerto paralelo de comunicación con norma IEEE488 esto lo hace un instrumento muy versátil sobre todo para aplicaciones donde el punto de monitoreo se encuentra en un lugar diferente a donde necesitamos tener el despliegue de la información, y deseamos utilizar una computadora para el manejo de la información.

2.2.6 MANEJADOR DE LA FUENTE DE ALTO VOLTAJE Y FUENTE DE ALTO VOLTAJE

La fuente de alto voltaje proporciona a la cámara de extrapolación el campo eléctrico necesario para capturar los iones producidos por la interacción de la radiación con el aire dentro de la cámara generándose una corriente eléctrica que es medida por el electrómetro; el manejador para la fuente recibe las palabras de control del PPIO que permite el ajuste de la fuente de voltaje enviándole la instrucción respectiva al manejador para que realice las acciones necesarias para ajustar el voltaje.

2.2.7 MANEJADOR DEL MOTOR DE PASOS

El ajuste de la distancia entre placas se realiza girando un tornillo micrométrico, por lo que se pensó en utilizar un motor de pasos junto con un arreglo mecánico para mover las placas; para tener control sobre el motor se utiliza este bloque que es el puente de comunicación entre el control central y el motor. Para darle la energía necesaria al motor este bloque también debe contar con una etapa de potencia para alimentarlo.

2.2.8 CÁMARA DE EXTRAPOLACIÓN

La cámara de extrapolación es la parte central del sistema, este instrumento , sencillo en su diseño se utiliza para detectar radiación, y de ella ya se habló en el capítulo anterior.

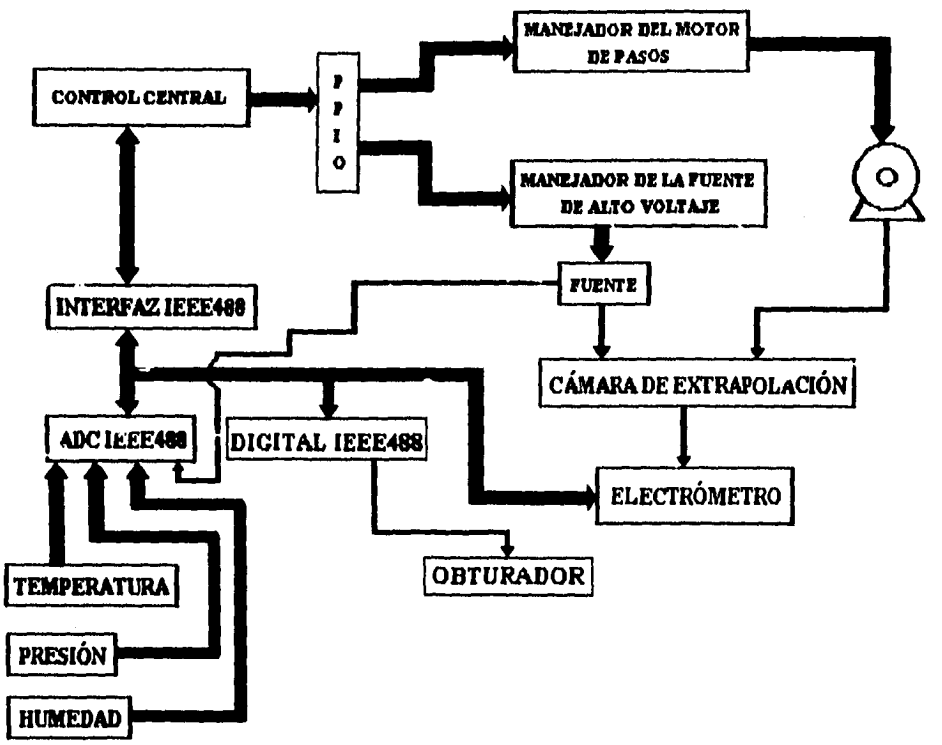


Fig. 3 Diagrama a bloques del sistema.

2.3 DISEÑO DEL CIRCUITO PARA MANEJAR LA FUENTE

Para polarizar la cámara de extrapolación, se utiliza una fuente de alto voltaje Hewlett Packard Mod. 6110 A; se utiliza esta fuente a pesar de no ser programable para aprovechar sus caracteristicas que se enlistan en seguida:

- intervalo de operación 0-3000 V y 0-6 mA
- regulación de carga menos de 10 ppm más 100 μV para carga completa y sin cambio en la corriente de salida.

-regulación de línea menos de 10 ppm en el voltaje de salida para un 10% de cambio en la línea de alimentación.
-ruido y rizo menos de 400 μV rms.
-coeficiente de temperatura menos de 10 ppm mas 50 μV en el voltaje de salida por °C despues de 30 minutos de calentamiento.
- estabilidad menos del 0.01 % mas 500 μV de corrimiento total despues de 30 minutos de calentamiento y con 3 °C de variación ambiente.

Las características de la fuente son muy buenas, la desventaja que tiene es que no tiene un puerto de comunicación para poder conectarlo a una computadora personal (**PC**), por lo que es necesario idear un circuito para hacer esto posible.

2.3.1 PRINCIPIOS DE OPERACIÓN DE LA FUENTE DE ALTO VOLTAJE

El modelo HP 6110A es una fuente de alimentación de voltaje constante limitada en corriente que utiliza un regulador con diseño "piggy-back" (como se puede ver en la fig. 4). La técnica básica consiste en colocar una fuente bien regulada de bajo voltaje "piggy-back" en serie con un doblador de voltaje no tan bien regulado. Observe, sin embargo, que la señal de error amplificada por el circuito de voltaje de entrada depende de un voltaje por arriba del voltaje total de salida y no solo del voltaje de salida de la fuente "Piggy-back". De este modo , la fuente "piggy-back" compensa continuamente cualquier deficiencia del rizo, la regulación de línea, o de la regulación de la carga, que pudiera presentarse en el doblador de voltaje, y ajustando el voltaje a través de un regulador serie, de modo que, el voltaje total de salida permanece constante a pesar de los disturbios en el doblador de alto voltaje. La fuente "piggy-back" entrega 200 V, y el doblador de alto voltaje es capaz de entregar hasta 3200 V. Con 30 V de caída a través del regulador serie, el máximo voltaje de salida es de 3370 V. De este modo, el regulador serie de la fuente "piggy-back" tiene un margen de voltaje suficiente para lograr los cambios dinámicos necesarios para compensar las variaciones de la fuente de potencia. La protección contra corto circuito para el regulador serie en la fuente "piggy-back" la realiza un diodo de protección, el cual ofrece una ruta de descarga del lado positivo de la fuente de poder al lado positivo del doblador de alto voltaje desviando la corriente de corto circuito alrededor del regulador serie. Cada vez que la resistencia de carga decrece a un valor tal que +S llegue a ser mayor que -200 volts, el diodo de protección conduce . Esto previene de cualquier voltaje inverso en las terminales de salida de la fuente "piggy-back" , y el regulador serie nunca esta expuesto a un voltaje de agotamineto mayor de 200 volts de su propio rectificador.

El voltaje de ac de la línea es elevado a un nivel adecuado y acoplado a la fuente "piggy-back", esta fuente convierte la entrada de ac en dc el cual alimenta la terminal positiva via el regulador serie y la corriente se muestrea por medio de una red de resistencias. El regulador, parte del lazo de retroalimentación, el cual esta diseñado para alterar su conducción y mantener constante el voltaje de salida o limitar la corriente de salida. El voltaje desarrollado a través del resistor muestreador de corriente es la entrada al circuito limitador de corriente. Si la corriente que pasa a través de la red de muestreo excede un determinado nivel, el circuito limitador de corriente aplica una señal de retroalimentación al regulador serie alterando la conducción del mismo evitando que la corriente no exceda un límite predeterminado.

El circuito de voltaje de entrada muestrea el voltaje de salida y lo compara contra el voltaje de control (referencia), Cualquier cambio en el voltaje de salida es detectado en el circuito de voltaje constante de entrada , amplificado por el manejador y amplificador de error, y aplicado al regulador serie en la fase y amplitud correcta para contrarrestar los cambios en el voltaje de salida . El regulador de las referencias de voltaje ofrece voltajes de referencia estables los cuales son utilizados en el circuito de voltaje constante de entrada y en el circuito limitador de corriente para propósitos de comparación.

El circuito de control de alto voltaje monitorea el voltaje a través de la fuente "piggy-back", y altera la conducción del tranformador T1, de modo que la salida del doblador de alto voltaje puede ser variada entre 0 y 3.2 kilovolts. Por ejemplo, si el control de alto voltaje es ajustado a un voltaje de salida mayor al que puede ofrecer la fuente "piggy-back", la entrada al circuito de control de alto voltaje se vuelve más negativa . El circuito de control de alto voltaje abre el control del embobinado y toda la energía de la entrada de ac es transferida al secundario del transformador T1 el cual esta conectado al doblador de alto voltaje . El resultado es que el voltaje del doblador se incrementa . Si el control de voltaje es ajustado para un voltaje menor al de la fuente "piggy-back", la entrada del

control de alto voltaje se vuelve menos negativa. El control del embobinado del transformador T1 se vuelve un corto impidiendo la transferencia de energía de la entrada de ac al secundario de T1 conectado al circuito doblador de alto voltaje. Así el voltaje de salida del doblador de alto voltaje decrece.

2.3.2 SOLUCIONES POSIBLES

La fuente de voltaje en cuestión emplea control de lazo cerrado con retroalimentación negativa para mantener estable el voltaje de salida de la fuente, el voltaje de salida es proporcional al voltaje de referencia, en

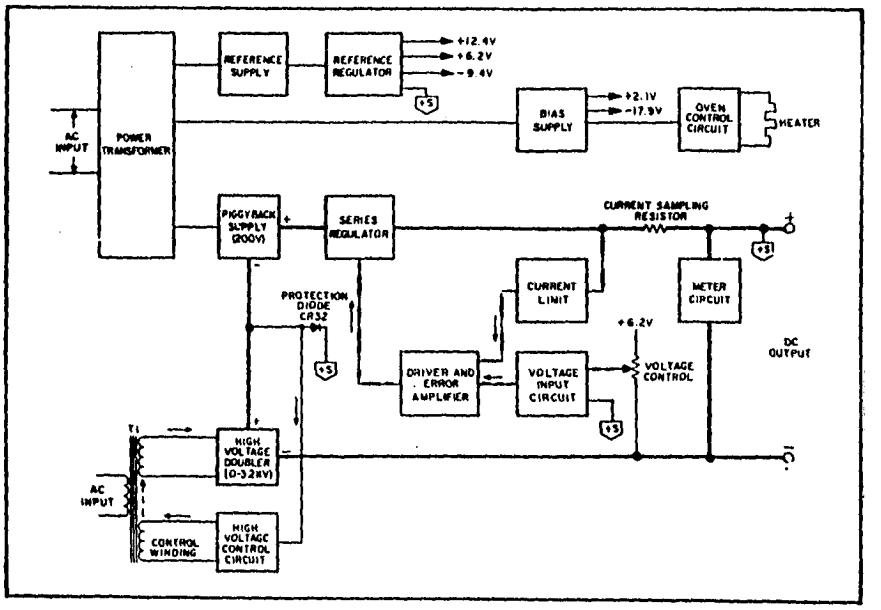


Fig. 4 Diagrama a bloques de la fuente HP 6110A

este caso el voltaje de salida es igual al voltaje de referencia. Para hacer la fuente programable; fijando el valor de la referencia por un conjunto de "bits" tenemos dos alternativas: La primera es utilizar un convertidor digital analógico y una etapa de amplificación para cubrir el intervalo de voltaje requerido, en la fig. 5 se muestra el diagrama a bloques.

La segunda opción es utilizar el mismo sistema para fijar el voltaje de referencia utilizado en el diseño, sustituyendo el conjunto de llaves por relevadores para hacer la conexión en serie de las resistencias adecuadas para dar el valor de voltaje requerido a la salida, estos relevadores se activarían por medio de una palabra de control como entrada a un circuito lógico combinacional que a su salida generaría una señal para activar cada uno de los relevadores seleccionados, en la fig. 6 se muestra el diagrama a bloques.

Fig. 5 Referencia programable utilizando un DAC

Tomando como parámetros de comparición, la dependencia de la estabilidad , el tamaño del circuito, el consumo de energía y el voltaje de polarización; se elabora la siguiente tabla comparativa entre las dos soluciones posibles.

	DAC
ESTABILIDAD	depende de la estabilidad del DAC , de la etapa de amplifiación y de la fuente de polarización del del circuito.
CONSUMO DE ENERGÍA	pequeño
TAMAÑO DEL CIRCUITO	pequeño
VOLTAJE DE POLARIZACIÓN	5 V. para los circuitos digitales; para la etapa de amplificación cuando menos el máximo voltaje que se desee en la fuente.
COMENTARIOS	Para polarizar la etapa de amplificación se requiere de una fuente con características, al menos semejantes a las de la fuente que se desea controlar.

	RELEVADORES Y RESISTENCIAS
ESTABILIDAD	Depende de la estabilidad de las resistencias
CONSUMO DE ENERGÍA	grande
TAMAÑO DEL CIRCUITO	grande
VOLTAJE POLARIZACIÓN	5 V para los circuitos digitales y 5 V o 12 V para energizar los relevadores.
COMENTARIOS	Se emplea el mismo principio que utiliza la fuente y solo se agrega un arreglo de resistencias y relevadores externos.

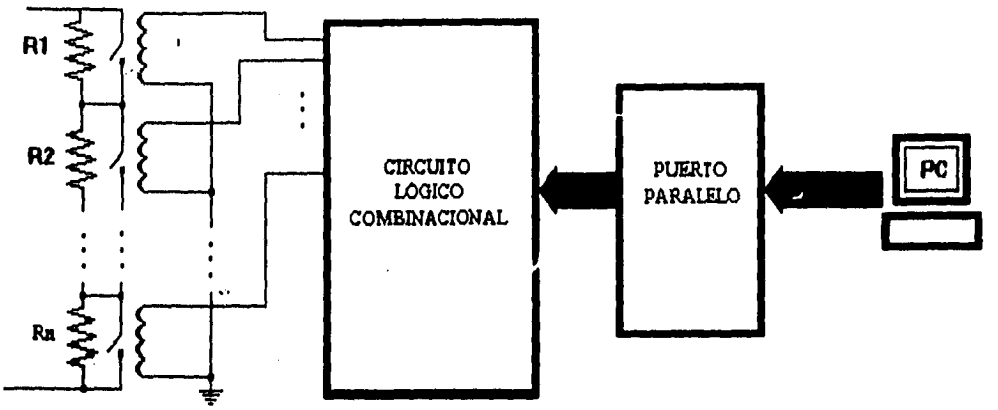


Fig. 6 Referencia programable utilizando relevadores y resistencias.

Analizando la información plasmada en la tabla anterior, la mejor solución es implementar un arreglo de resistencias y relevadores, ya que a pesar de que el circuito resultante va a ser mucho más grande, es más sencillo de diseñar y la estabilidad de la referencia depende de un solo factor que se puede garantizar con la adecuada selección de las resistencias a utilizar.

2.3.3 DISEÑO DEL CIRCUITO PARA LA REFERENCIA PROGRAMABLE

El circuito de referencia de voltaje de la fuente es un circuito sencillo e ingenioso que utiliza una fuente de voltaje referenciada en la terminal positiva de salida de la fuente y un divisor de voltaje donde una resistencia esta fija, y la otra es variable como se muestra en la fig. 7

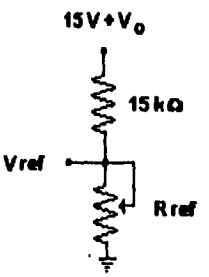


Fig. 7 referencia de voltaje

Al dar a R_{ref} un valor determinado, por ejemplo 10 KΩ, si V_0 de 0V, tenemos que:

$$V_{ref} = 15 \quad (10 \quad / (15 + 10)) = 6\ V$$

Como :

$$V_{ref} = V_0$$

ahora tenemos que

$$V_{ref} = (15 + 6)(10 / (15 + 10)) = 8.4\ V$$

siguiendo la misma secuencia

$$V_{ref} = (15 + 8.4)(10 / (15 + 10)) = 9.36\ V$$
$$V_{ref} = (15 + 9.36)(10 / (15 + 10)) = 9.74\ V$$
$$V_{ref} = (15 + 9.74)(10 / (15 + 10)) = 9.8976\ V$$
$$V_{ref} = (15 + 9.74)(10 / (15 + 10)) = 9.95904\ V$$

así podemos continuar iterando hasta que lleguemos a:

$$V_{ref} = 10\ V$$

Como se puede apreciar el valor de la resistencia es proporcional al valor del voltaje de referencia, si:

$$10\ k\Omega \Rightarrow 10\ V$$

entonces:

$$1\ k\Omega \Rightarrow 1\ V$$

Para cubrir un intervalo de 0 a 400 V con pasos de 0.1 V necesitamos 4 conjuntos de resistencias de 9 resistencias conectadas en serie de los siguientes valores :

	valor (k Ω)	voltaje (V)
BANCO 1	**0.1**	**0 - 0.9**
BANCO 2	**1.0**	**0 - 9.0**
BANCO 3	**10**	**0 - 90.0**
BANCO 4 (3 res.)	**100**	**0 - 300.0**

Si conectamos los 4 bancos y colocamos entre las resistencias interruptores activados por relevador (ver fig. 6) tenemos la capacidad de generar voltajes en un intervalo de 0 a 399.9 V con una resolución de 0.1 V. Por seguridad hay que garantizar que la fuente al energizarse esté en 0V, esto se logra conectando los relevadores en el polo normalmente cerrado, de tal manera que la resistencia total de R_{ref} sea de 0 ohms, al ir abriendo los interruptores, se van sumando resistencias a R_{ref} permitiendo de este modo ajustar el voltaje deseado. Cada bloque de resistencias tendrá entonces la estructura mostrada en la fig. 8.

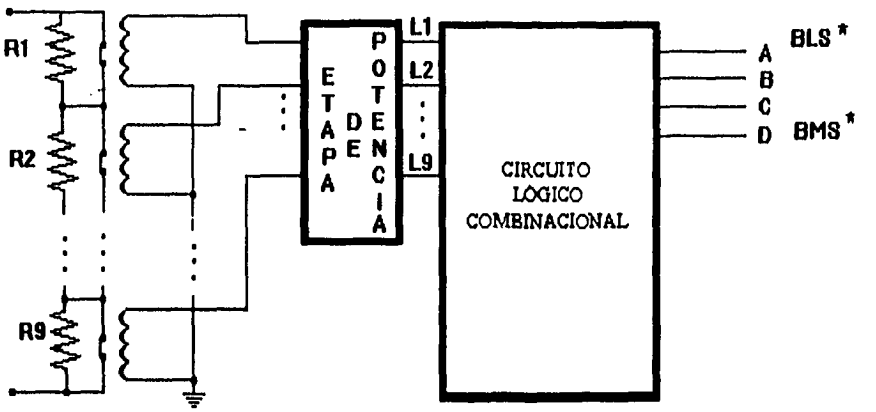


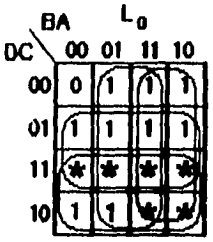
Fig. 8 diagrama a bloques de un banco de resistencias

Así, el circuito combinacional tendrá 4 entradas y 9 salidas, una por cada relevador, de tal manera que el número en BCD aplicado a la entrada del circuito sea el número de relevadores a energizar, esto se plasma en la siguiente tabla de verdad, donde un "0" indica un relevador sin energía y un "1" un relevador energizados.

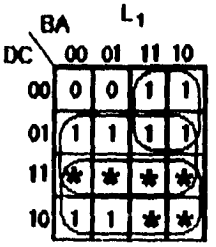
ENTRADAS	SALIDAS								
D C B A	L_0	L_1	L_2	L_3	L_4	L_5	L_6	L_7	L_8
0 0 0 0	0	0	0	0	0	0	0	0	0
0 0 0 1	1	0	0	0	0	0	0	0	0
0 0 1 0	1	1	0	0	0	0	0	0	0
0 0 1 1	1	1	1	0	0	0	0	0	0
0 1 0 0	1	1	1	1	0	0	0	0	0
0 1 0 1	1	1	1	1	1	0	0	0	0
0 1 1 0	1	1	1	1	1	1	0	0	0
0 1 1 1	1	1	1	1	1	1	1	0	0
1 0 0 0	1	1	1	1	1	1	1	1	0
1 0 0 1	1	1	1	1	1	1	1	1	1

Tabla 1 tabla de verdad para el circuito combinacional de la referencia programable

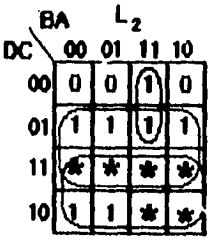
Para implementar el circuito se genera una ecuación lógica por cada salida donde intervienen las entradas , se obtienen utilizando el método de reducción de mapas de Karnaugh que se aplica a continuación para cada salida;



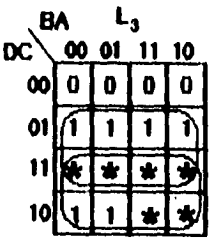
$L_0 = A + B + C + D$



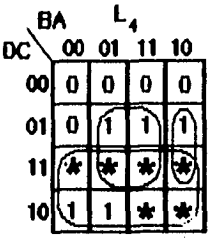
$L_1 = C + D + \bar{D}B$



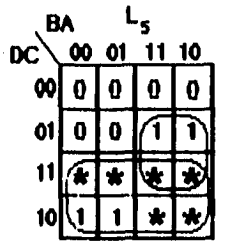
$L_2 = D + C + \bar{D}BA$



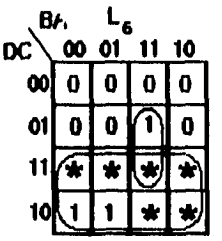
$L_3 = D + C$



$L_4 = D + CA + CB\overline{A}$



$L_5 = D + CB$



$L_6 = D + CBA$

Para L_7, no es necesario elaborar el mapa de Karnaugh respectivo; si observamos la tabla de verdad, podemos ver que cuando la entrada D esta en "1", la salida L_7 también tiene un "1", por lo que la ecuación lógica respectiva es:

$$L_7 = D$$

Para la salida L_8 tampoco es necesario elaborar el mapa respectivo, debido a que solo existe un minitérmino para esta salida de tal modo que:

$$L_8 = D\overline{C}\overline{B}A$$

Para la realización del circuito tenemos dos opciones, la primera es emplear compuertas TTL y la segunda es utilizar dispositivos lógicos programables

ETAPA DE POTENCIA

Cada salida del circuito combinacional activa un relevador que permite la conexion en serie (nivel lógico "1") o cortocircuito (nivel lógico "0") de la resistencia respectiva; para manejar la corriente del embobinado del relevador se propone el siguiente circuito.

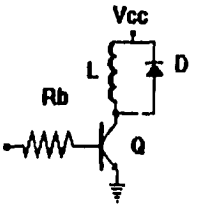


Fig. 9 Circuito para manejar la corriente de la bobina del relevador

El transistor Q funciona como un interruptor, cuando se aplica un nivel de voltaje de "1" lógico a la resistencia Rb, Q se polariza en la región de saturación permitiendo el paso de la corriente a través del colector y el emisor a tierra y con un voltaje de colector a emisor de 0.2 V aprox. , cuando se aplica un nivel de voltaje "0" lógico, el transistor se va a la región de corte, ofreciendo muy alta impedancia impidiendo el paso de la corriente. El relevador seleccionado opera a 12 V y tiene una impedancia resistiva de 320 Ω de tal modo que la corriente que demanda la bobina del relevador es de

$$I_L = 12\ V / 320\ \Omega = 37.5\ mA \qquad (11)$$

por lo tanto requerimos de un transistor del tipo NPN que soporte una corriente máxima de colector de 100 mA y un voltaje de ruptura entre colector y emisor de 30 V mín. , la frecuencia de operación no es alta, por lo que no es necesario utilizar un transistor de conmutación, un transistor de propósito general cubre la aplicación, se selecciona el transistor BC 547 que tiene las siguientes características [5]

$$BV_{CEO} = 40\ V$$
$$I_C = 0.8\ A$$
$$h_{FE}\ min = 125$$
$$h_{FE}\ máx = 450$$

Para garantizar que el transistor se sature, en el cálculo de la resistencia de base se considera la ganancia de corriente mínima; para calcular R_b se considera que I_C=37.5 mA, de las ecuaciones básicas del transistor :

$$I_C = \beta I_B \qquad (12)$$

$$V_B = I_B R_B + V_{BE} \qquad (13)$$

despejando I_B en (12) y sustituyendo valores:

$$I_B = 37.5\ mA / 125 = 300\ \mu A$$

Para calcular R_B se considera un voltaje de base de 3.5 V que es el mínimo voltaje que se considera como nivel alto en la lógica TTL ; despejando R_B de (13) y sustituyendo valores :

$$R_B = (3.4\ V - 0.7\ V) / 300 \times 10^{-6}\ A$$
$$R_B = 9\ k\Omega$$

Comercialmente no existen resistencias de 9 kΩ, por lo que se selecciona el valor más próximo comercial de 10 kΩ.

2.4 DISEÑO DEL CIRCUITO PARA MOVER EL MOTOR DE PASOS

2.4.1 MOTOR DE PASOS

Un motor de pasos opera al energizar en una secuencia determinada las fases de su embobinado, el motor utilizado avanza 200 pasos por revolución (1.8 ° por paso) cuando se utiliza una secuencia de 4 pasos (modo de paso completo)[6] y de 400 pasos por revolución si se utiliza la secuencia de ocho pasos (modo de medio paso), para energizar las bobinas del motor se utilizan transistores de potencia conectados a un circuito lógico secuencial que genere la secuencia de energizado de las bobinas; en la fig. 10 se muestra el diagrama esquemático del circuito del motor y la tabla de la secuencia de entrada para el modo de medio paso.

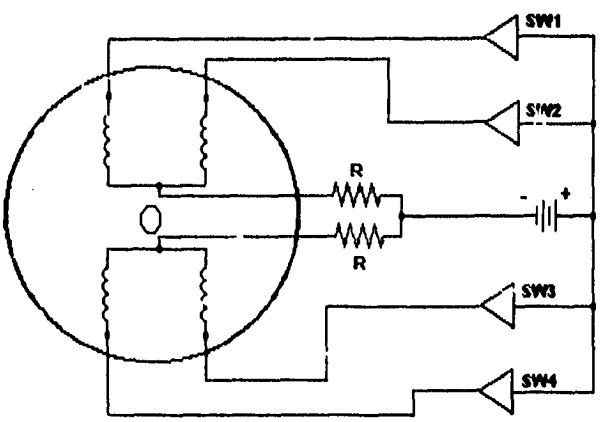


Fig. 10 Circuito del motor de pasos.

2.4.2 DISEÑO DEL CIRCUITO SECUENCIAL

El circuito que necesitamos es un contador que tenga 2 entradas, una de ellas es un reloj que por cada pulso nos dé un cambio de estado y el motor dé un paso, y la otra entrada es un bit que nos represente el sentido del giro para indicarle al circuito en que sentido deseamos que el motor gire; el circuito debe tener también 4 salidas, cada una de ellas activará el respectivo transistor para energizar la bobina. En la fig . 10 se muestra un diagrama a bloques del circuito propuesto.
En la fig. 11 , se muestra el diagrama a bloques del circuito para mover el motor de pasos, el cual lo componen el circuito lógico combinacional, el circuito de potencia y el motor; en la tabla II se muestra la secuencia de señales para energizar el motor.

PASO	SW1	SW2	SW3	SW4
1	on	off	on	off
2	on	off	off	off
3	on	off	off	on
4	off	off	off	on
5	off	on	off	on
6	off	on	off	off
7	off	on	on	off
8	off	off	on	off
1	on	off	on	off

Tabla II Secuencia de encendido para las bobinas del motor de pasos

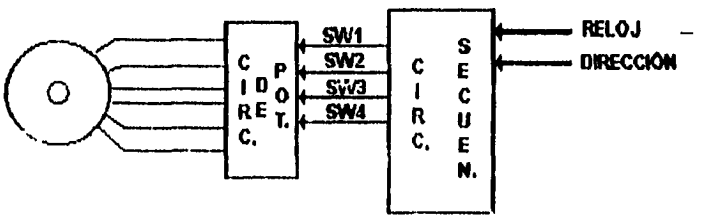


Fig.11 Diagrama a bloques del circuito para mover el motor

El ciclo completo es de 8 pasos, u ocho estados, por lo que necesitamos diseñar es un contador con tres FF (flip-flop), para tal efecto utilizamos FF tipo D, en dichos dispositivos, el estado siguiente es igual al estado presente en la entrada D cuando aparece un pulso de reloj, por comodidad denominaremos a "T" como la señal que nos de la dirección en la que debe girar el motor, si es "1" el motor gira hacia "adelante" y si es "0" gira hacia "atras" las salidas "SW" las identificaremos de la siguiente manera:

SW1 = A
SW2 = B
SW3 = C
SW4 = D

Por convención tomaremos un "1" como "ON" y un "0" como "OFF"; definidas las variables se procede a crear la tabla de estados para el circuito secuencial que se muestra a continuación:

Estado presente T=1	Estado siguiente T=0	Estado siguiente T=1	Salida
X Y Z	X Y Z	X Y Z	A B C D
0 0 0	0 0 0	1 1 1	1 0 1 0
0 0 1	0 1 0	0 0 0	1 0 0 0
0 1 0	0 1 1	0 0 1	1 0 0 1
0 1 1	1 0 0	0 1 0	0 0 0 1
1 0 0	1 0 1	0 1 1	0 1 0 1
1 0 1	1 1 0	1 0 0	0 1 0 0
1 1 0	1 1 1	1 0 1	0 1 1 0
1 1 1	0 0 0	1 1 0	0 0 1 0

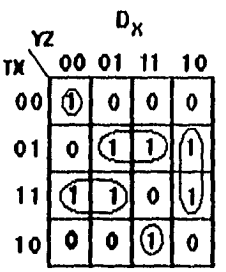
Tabla III Tabla de estados para el circuito del motor de pasos

Un circuito secuencial lo componen la "memoria" del circuito, que recuerda cual fué el estado anterior, constituida por los FF, y la otra es un circuito combinacional que genera las señales necesarias de entrada a los FF para que se genere el cambio de estado deseado, toda esta información se concentra en una tabla de excitación considerando los estados presente y siguiente, así como las entradas externas; para el circuito que se diseña se muestra la siguiente tabla de excitación considerando que se utilizan para el diseño FF tipo D.

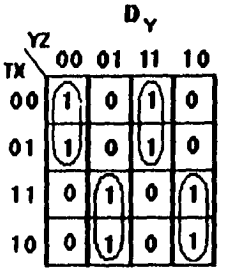
Entradas del circuito combinacional		Salidas del circuito combinacional	
Estado presente	Entrada	Estado siguiente	Entradas a los FF
X Y Z	T	X Y Z	D_X D_Y D_Z
0 0 0	1	0 0 1	0 0 1
0 0 0	0	1 1 1	1 1 1
0 0 1	1	0 1 0	0 1 0
0 0 1	0	0 0 0	0 0 0
0 1 0	1	0 1 1	0 1 1
0 1 0	0	0 0 1	0 0 1
0 1 1	1	1 0 0	1 0 0
0 1 1	0	0 1 0	0 1 0
1 0 0	1	1 0 1	1 0 1
1 0 0	0	0 1 1	0 1 1
1 0 1	1	1 1 0	1 1 0
1 0 1	0	1 0 0	1 0 0
1 1 0	1	1 1 1	1 1 1
1 1 0	0	1 0 1	1 0 1
1 1 1	1	0 0 0	0 0 0
1 1 1	0	1 1 0	1 1 0

Tabla IV Tabla de excitación del circuito para mover el motor.

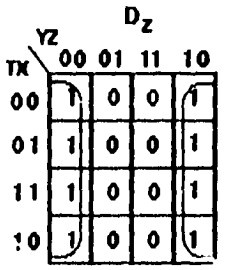
La información vertida en la tabla de excitación se transfiere a mapas de Karnaugh para simplificar el conjunto de minitérminos y reducir las ecuaciones que definen al circuito, a continuación se muestran los mapas con las respectivas reducciones.



$$D_X = \bar{T}X\bar{Y}\bar{Z} + TX\bar{Y} + \bar{T}XZ + XY\bar{Z} + T\bar{X}YZ$$

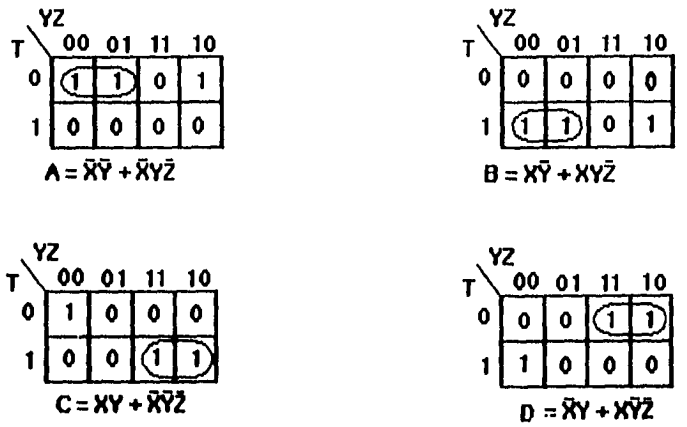


$$D_Y = \bar{T}\bar{Y}Z + T\bar{Y}Z + \bar{T}YZ + TY\bar{Z}$$



$$D_Z = \bar{Z}$$

A partir de los ocho estados del circuito se generan las cuatro salidas externas utilizando los valores de los ocho estados del circuito secuencial , la relación entre los estados del circuito secuencial y la entrada T se puede ver en la tabla de estados, donde para cada estado presente se muestra la salidarespectiva, aplicando el método de mapas de Karnaugh para esta información, se obtienen los mapas y ecuaciones resultantes para cada salida.



2.4.2.1 CIRCUITO DE POTENCIA

Para acoplar el circuito secuencial con el circuito de potencia, se utilizan optoacopladores para aislar ópticamente las dos etapas, El circuito utilizado, se muestra en la fig. 12 para esta aplicación se seleccionó un optoacoplador 4N28 con salida a transistor[7], este tipo de optoacoplador nos permite utilizarlo como un interruptor que es fácil de conmutar.

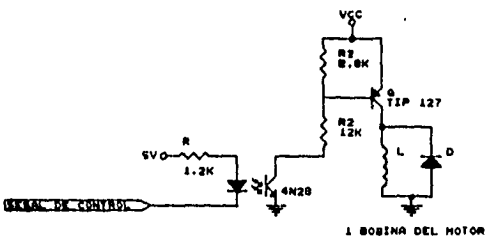


Fig. 12 Diagrama eléctrico del acoplamiento óptico

El optoacoplador seleccionado soporta una corriente a la salida de 2 ma máximo, y la corriente que necesita para operar el motor es de 1.6 A por bobina, por lo que la ganancia mínima de corriente que se necesita es:

$$\beta = 1.6\ A / 2\ mA = 800$$

para tener un márgen de seguridad, se determina que la corriente que pase por el transistor de salida del optoacoplador, sea de 1.6 mA lo que nos da un márgen de seguridad del 20 %, de tal modo que la ganancia de corriente que se requiere es de

$$\beta_{minima} = 1000$$

El transistor que requerimos debe tener una ganacia en corriente de 1000 min. y soportar una corrientede 2 A (ya considerando el 20 % de tolerancia); por lo que, se selecciona el transistor TIP 127 [8]. Las resistencias de polarización, se determinan de la siguiente manera, primero simplificamos el circuito representando R_1 y R_2 como una resistencia equivalente R_{Th} y una fuente de voltaje equivalente V_{Th} com se muestra en la fig. 13

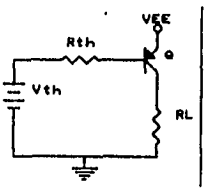


Fig. 13 Circuito equivalente

donde,

$$R_{Th} = R_1 R_2 / (R_1 + R_2) \qquad (27)$$

$$V_{Th} = V_{CC} - V_{CE\,Q1} (R_2 / (R_2 + R_1)) \qquad (28)$$

los valores que conocemos son :

$V_{CE\,Q1} = 1$ V
$V_{BE\,Q2} = 1.4$ V
$I_{C\,Q2} = 1.6$ A
$I_{B\,Q2} = 1.6$ mA
$V_{CC} = 5$ V

y deseamos conocer R_1 y R_2 por conveniencia se fija $V_{Th} = 2.5$ V .
Resolviendo la malla I, tenemos

$$V_{CC} = V_{BE\,Q1} + I_{B\,Q2} R_{Th} + V_{Th} \qquad (29)$$

despejando R_{Th} de (29) y sustituyendo valores

$$R_{Th} = (5 - 1.4 - 2.5) / 1.6 \times 10^{-3} = 687.5\ \Omega$$

despejando de (27) $R_1 + R_2$

$$R_1 + R_2 = R_1 R_2 / R_{Th} \qquad (30)$$

sustituyendo (30) en (28) tenemos

$$V_{Th} = (V_{CC} - V_{CE\,Q1}) R_{Th} / R_1 \qquad (31)$$

despejando R_1 y sustituyendo valores

$$R_1 = ((5 - 1) / 2.5)\, 687.5 = 1100\ \Omega$$

$$R_1 = ((5 - 1) / 2.5)\, 687.5 = 1100\ \Omega$$

El valor comercial más cercano es de 1.2 kΩ de tal modo que

$$R_1 = 1.2\ k\Omega$$

de la ecuación (27) despejamos R_2 y sustituimos valores

$$R_2 = (1200)(687.5) / (1200 - 678.5) = 1609.75\ \Omega$$

considerando que las resistencias tienden a medir menos del valor marcado, se selecciona el valor de 1.8 kΩ entonces

$$R_2 = 1.8\ k\Omega$$

recalculando I_{BQ2} tenemos que

$$R_{Th} = (1200)(1800) / (1200 + 1880) = 720\ \Omega$$
$$I_{BQ2} = (5 - 1.4 - 2.5) / 720 = 1.527\ mA$$

o $\quad I_{CQ2} = \beta I_{BQ2} = (1000)(1.527 \times 10^{-3}) = 1.527\ A$

que es un valor aceptable para la operación del motor, cabe aclarar que el valor de la ganancia de corriente del transistor es mínima, por lo general el valor típico es mayor, lo que garantiza que se tenga la corriente calculada.

2.5 IMPLEMENTACIÓN DE LOS CIRCUITOS LÓGICOS

2.5.1 OPCIONES PARA EL DISEÑO DE CIRCUITOS DIGITALES

Actualmente el diseñador de lógica digital tiene la oportunidad de diseñar con cualquiera de las seis categorías primarias de dispositivos digitales:

Dispositivos tipo SSI/MSI.
Dispositivos tipo LSI/VLSI.
Arreglos de compuertas.
Dispositivos "Standard-cell".
Dispositivos "Full-custom".
Dispositivos de lógica programable (PLD).

cada categoría cubre una serie de aplicaciones para las cuales puede ser empleada una categoría en especial, o una combinación de ellas.

Dispositivos tipo SSI/MSI

La categoría de dispositivos SSI/MSI (pequeña escala de integración, mediana escala de integración) comprende dispositivos como la serie 4000 CMOS y la serie 74XX00 TTL/CMOS, relativamente baratos, actualmente se utilizan en sistemas digitales siempre que se necesiten "buffers", registros de corrimiento, contadores, flip-flops, compuertas y otras funciones lógicas.
Los dispositivos SSI/MSI son baratos desde el punto de vista del costo por"chip", pero son caros e ineficientes desde el punto de vista del costo por compuerta. El bajo costo por unidad es resultado de su alto volumen de producción, ayudado por el pequeño tamaño de la oblea y mejoras en las tecnologías de semiconductor.

Dispositivos tipo LSI/VLSI

Los dispositivos LSI/VLSI (alta escala de integración, muy alta escala de integración) incluyen a los microprocesadores y dispositivos periféricos (UART's, controladores de interrupción, controladores de DMA, etc.). Estos dispositivos ofrecen un alto nivel de funcionalidad y relativamente bajo costo (medido en costo por compuerta). Los dispositivos de esta categoría, y en particular los microprocesadores, han revolucionado la industria del diseño digital. Actualmente es raro encontrar un diseño que no utilice en alguna forma un microprocesador.
Estos dispositivos ofrecen el mejor costo por compuerta que cualquier otra categoría, porque son dispositivos "full-custom", usan el area de silicio muy eficientemente, solo las compuertas necesarias se incluyen en el "chip" . La alta escala de integración reduce también los vacíos en el empaquetado y el silicio, asi como su fabricación en altos volúmenes ayuda a establecer su bajo costo por compuerta.

Dispositivos de arreglos de compuertas

Los arreglos de compuertas son dispositivos que contienen un "mar de compuertas", estos dispositivos generalmente son prefabricados, con la oblea de silicio completa a excepción de las capas finales de metalización . Las capas finales, en un arreglo de compuertas ofrecen la interconexión entre las compuertas, con las que se definen las funciones lógicas a ser implementadas por el arreglo. Los arreglos de compuertas están disponibles desde menos de 1,000 compuertas hasta arreglos con más de 50,000, aunque los más populares tienen menos de 10,000 compuertas.
La arquitectura de estos dispositivos no permite el uso muy eficiente de las compuertas del "chip", pues por ejemplo para una aplicación que requiera de 2,000 compuertas, requerirá de un arreglo de compuertas de aprox. 5,000. Típicamente la eficiencia de uso es del 40% aunque nuevos diseños afirman tener hasta un 75% de eficiencia. Trasladando esto a pesos y centavos, el ineficiente uso de las compuertas significa una gran cantidad de "vacios" de silicio dentro del arreglo de compuertas, este costo debe ser reflejado por el precio de venta del dispositivo.
Sin embargo, los arreglos de compuertas son una alternativa atractiva cuando miles de dispositivos SSI/MSI , y las funciones de muchos de estos dispositivos pueden ser convenientemente amalgamadas dentro de un solo arreglo de compuertas. Debido a que la interconexión debe ser definida, el diseño de un arreglo de compuertas comprende los cargos por ingeniería no recurrente (INR) los cargos por elaboración de la máscara y por el uso del sistema de diseño asistido por computadora (CAD). Debido al costo de la ingeniería no recurrente, los arreglos de compuertas se utilizan en aplicaciones donde el volumen de producción es medio o alto (generalmente más de 5,000 unidades) permitiendo amortizar los costos por la INR sobre un gran número de dispositivos. En algunas aplicaciones de bajo volumen de producción, donde el espacio es crítico también se usan arreglos de compuertas para ahorrar espacio, despreciando el costo de la INR.
Una vez que se ha completado el diseño, se tienen que esperar varias semanas antes que el arreglo de compuertas con el diseño requerido este disponible, dependiendo de la complejidad , la tecnología empleada, la envoltura los requerimientos de prueba y el volumen de venta, los dispositivos de arreglos de compuertas pueden costar desde unos pocos dolares hasta cientos de dolares cada uno.

Dispositivos de celdas estándar

Los dispositivos de celdas estándar son semejantes a los arreglos de compuertas, estos dispositivos "*semicustom* " implican cargos por INR en la generación del diseño, se distinguen de los arreglos de compuertas en que estos dispositivos no son prefabricados y no tienen un "mar de compuertas", en su lugar, el sistema de CAD empleado para él cuenta con una *biblioteca de celdas* donde muchos bloques funcionales del tipo SSI, MSI y LSI están definidos. Una vez que se han definido los bloques funcionales a utilizar, el sistema de CAD se encarga de definir la interconexión entre las diferentes celdas estándar semejante a unir las piezas de un rompecabezas- con estos dispositivos se pueden diseñar "tablillas completas en un "chip"", por ejemplo se puede definir un dispositivo de celdas estándar que contenga un microporcesador Z80, un controlador de DMA, un decodificador de direcciones

y un bus con "buffer". El resultado es que un "chip" puede reemplazar un circuito impreso completo con componentes. El "chip" no solo ahorra una notable cantidad de espacio en la tablilla, disminuye el costo de los componentes individuales que reemplaza, a condición de que el volumen de producción sea suficientemente grande.

Como es de esperarse, los cargos por la INR son mayores que los generados por el diseño con arreglos de compuertas, esto se debe a que se utiliza un programa más avanzado para manejar adecuadamente la definición de la interconexión de las celdas de la biblioteca y desarrollar la simulación lógica del diseño,. También hay que generar un juego completo de máscaras para elaborar la oblea de silicio.

Otra de las ventajas que ofrecen estos dispositivos es la eficiencia en el uso de las compuertas , ya que la biblioteca de celdas define todos los elementos del bloque funcional permitiendo utilizar solo las compuertas que se requieren esto hace que el area de silico se emplee eficientemente. El costo del dispositivo es proporcional al tamaño de la oblea de silicio (para un volumen suficientemente grande), el costo de los dispositivos Standard-cell es generalmente menor al de los dispositivos de arreglos de compuertas. El tiempo de entrega de estos dispositivos puede ser de varios meses.

Dispositivos "Full-Custom"

Full custom, como su nombre lo indica, abarca el diseño de dispositivos de "marca". Esta propuesta comprende grandes costos de INR y el mayor tiempo de desarrollo que otras alternativas, y es adecuado solo para aplicaciones de muy altos volúmenes, Típicamente estas aplicaciones incuyen circuitos para uso automotriz, calculadoras y relojes digitales, Su costo para altos volúmenes de producción puede ser muy económico , debido al extremadamente eficiente uso del espacio en la oblea de silicio.

Dispositivos Lógicos Programables

Los dispositvos lógicos programables combinan características que se encuentran en los dispositivos de celdas estándar algunas de las características de los arreglos de compuertas, dando como resultado una gran versatilidad y utilidad. Parecido a los dispositivos de celdas estándar, los PLD's se fabrican en grandes volúmenes con arquitecturas de norma, resultado: se reducen costos de este tipo de dispositivos. Semejantes a los arreglos de compuertas, los PLD's son *Circuitos Integrados de Aplicación Específica* (ASIC's). pueden reeplazar muchos SSI/MSI y tienen la ventaja de ser programados por el usuario contando con la adecuada circuitería y paquetes de programación.[9]

2.5.2 Selección de la alternativa para el diseño

En párrafos anteriores se describieron las alternativas con las que se cuenta para la implementación de circuitos digitales, sus ventajas, desventajas y características principales.Para determinar que alternativa es la mejor , hay que considerar la arquitectura resultante de los dispositivos, el espacio disponible en la tablilla de circuito impreso, tiempo de desarrollo, volumen esperado de producción y costo. Otras consideraciones que pueden ser aplicadas son el tiempo de propagación y el consumo de potencia. En nuestro caso las características que se requieren son:

- Corto tiempo de desarrollo.
- Volumen de producción 1 prototipo.
- Espacio reducido.
- Circuito impreso sencillo.
- facilidad para probar el circuito y de dar mantenimiento.
- bajo costo.

en la siguiente tabla comparativa se muestran las diferentes alternativas

Alternativa Lógica	Costo INR	Tiempo de Desarrollo	Tiempo de aplicación	Costo Relativo del Disp. Costo/compuerta	Volumen de Produce. (típico)
full-custom	muy alto	) 1 año	INR + meses	muy bajo	más de 100 K
standard-cell	mas de $50K	meses	INR+tiempo Des.	bajo	más de 20 K
arreglo de compuerta	$10K a 50K	semanas-meses	INR+tiempo Des.	med.-bajo	mas de 5 K
PLD	$ 0	horas-semana	horas	medio	menos de 10K
LSI/VLSI	$0	ninguno	-	muy bajo	cualquiera
SSI/MSI	$0	ninguno	-	alto	cualquiera

Tabla V Tabla comparativa de alternativas lógicas

Considerando la información de la tabla V y lo que anteriormente se ha planteado, la alternativa más adecuada para implementar los circuitos digitales es la utilización de dispositivos lógicos programables.
Para la selección del PLD el principal requisito a cumplir es el número de entradas y de salidas necesarias; para el circuito de la referencia variable de la fuente se necesitan 4 entradas y 8 salidas (considerandoque R_T "1" cuando D es "1") para este caso solo se necesitan compuertas combinacionales, seselecciona el PAL 16L8 que cuenta con 8 salidas, 7 términos producto por cada salida y 10 entradas [10].
Para la implementación del circuito secuencial necesitamos 1 entrada 4 flip-flops y 4 salidas,se selecciona el PAL 16R4; cabe hacer notar que el GAL 16V8 puede sustituir a los PAL 16L8 y 16R4 con la ventaja de ser borrables eléctricamente, lo que permite cambiar el diseño del dispositivo sin tener que desechar la unidad [11].
Para generar el archivo para la quema de fusibles del PAL se utiliza el programa CUPL que se alimenta con las ecuaciones lógicas obtenidas en párrafos anteriores; y los archivos se muestran a continuación.

Este archivo es el archivo conthvpa.jed que contiene la información para la quema de fusibles en el PAL que genera las señales para ajustar la fuente de voltaje.

```
CUPL      2.11c Serial# 1-00005-002
Device    p16l8 Library DLIB-f-23-8
Created   Fri Jan 21 11:10:30 1994
Name      CONTHVPA
Partno    XXXXX
Revision  00
Date      19/01/93
Designer  Fernando J.
Company   ININ
Assembly  XXXXX
Location  XXXXX
*QP20
```

```
*QP20
*QF2048
*G0
*F0
*L0000 11111111111111111111111111111111
*L0032 10101011101111111111111111111111
*L0256 11111111111111111111111111111111
*L0288 10111011101111111111111111111111
*L0512 11111111111111111111111111111111
*L0544 11101011101111111111111111111111
*L0576 10111011101111111111111111111111
*L0768 11111111111111111111111111111111
*L0800 11111011101111111111111111111111
*L1024 11111111111111111111111111111111
*L1056 10101111101111111111111111111111
*L1088 11111011101111111111111111111111
*L1280 11111111111111111111111111111111
*L1312 11111011101111111111111111111111
*L1344 10111111101111111111111111111111
*L1536 11111111111111111111111111111111
*L1568 11111011101111111111111111111111
*L1600 10111111101111111111111111111111
*L1632 11101111101111111111111111111111
*L1792 11111111111111111111111111111111
*L1824 11111111101111111111111111111111
*L1856 11101111111111111111111111111111
*C565A
* F8C4
```

El listado que sigue es el listado del archivo motdrvne.jed, este archivocontiene la información para la quema de fusibles de un pal 16r4 con el que se implementa el circuito lógico combinacional para generar las señales que permiten el movimiento del motor de pasos.

```
CUPL       2.11c Serial# 1-00005-002
Device     p16r4 Library DLIB-f-23-11
Created    Sun Nov 28 12:58:58 1993
Name       motdrvne
Partno     XXXXX
Revision   00
Date       26/01/93
Designer   Fernando
Company    ININ
Assembly   XXXXX
Location   XXXXX
*QP20
*QF2048
*G0
*F0
```

```
*L0000 1111111111111111111111111111111111
*L0032 1111111111101110111111111111111111
*L0064 1111111111011110110111111111111111
*L0256 1111111111111111111111111111111111
*L0288 1111111111011011111111111111111111
*L0320 1111111111011101101111111111111111
*L0512 1011111111011101101111111111111111
*L0544 0111111111101101111111111111111111
*L0576 1011111111101111101111111111111111
*L0608 1111111111011101101111111111111111
*L0640 0111111111011101101111111111111111
*L0768 1011111111111011101111111111111111
*L0800 0111111111111011101111111111111111
*L0832 1011111111111111011101111111111111
*L0864 0111111111111111011011111111111111
*L1024 1111111111111111110111111111111111
*L1536 1111111111111111111111111111111111
*L1568 1111111111101110111011111111111111
*L1600 1111111111110111011111111111111111
*L1792 1111111111111111111111111111111111
*L1824 1111111111101101110111111111111111
*L1856 1111111111011110111111111111111111
*C5108
```

2.6 PPIO Y DIRECCIONAMIENTO

Para poder comunicar los dispositivos con la computadora necesitamos un conducto por el cual dar a los circuitos diseñados las palabras de control para su operación, esta función la hacemos con un puerto paralelo de entrada salida, también necesitamos de alguna manera indicarle al puerto paralelo cuando se le esta requiriendo, esto se hace por medio de la decodificación de la dirección y de las señales de control que para tal fin se utilizan en la computadora, en la fig. 14 se muestra el diagrama eléctrico de esta parte.

Para esta parte se utiliza un puerto paralelo de entrada/salida del tipo 8255 que tiene 3 puertos de ocho bits, uno de los cuales se puede dividir en dos puertos de 4 bits que pueden operar independientemente,se puede operar en 3 modos distintos[12], para esta aplicación en particular se utiliza el modo cero y se programan los tres puertos para operar como salidas, al puerto **A** se le asignan las salidas de los bits para definir décimas de unidad y unidades en el circuito para la fuente programable, al puerto **B** se le asignan los bits para definir decenas y unidades, mientras que el puerto **C** se utilizan los dos primeros bits del puerto se utilizan uno para generar el reloj para el circuito manejador del motor, el segundo se utiliza para definir el sentido de giro del motor, y un tercer bit se utiliza para dar la señal para la inversión de voltaje en la fuente. Para determinar el puerto al que se destina la salida utiliza las primeras dos direcciones (A0 y A1) del bus de direcciones; para el direccionamiento se utiliza un comparador de magnitud 74LS688 junto con un conjunto de 8 resistencias y un "dip-switch" para fijar la dirección en la que va a operar la tarjeta, por un lado, por el otro lado se conectan al comparador las siguientes ocho direcciones (A2 aA9), cuando el "byte" del bus de datos es igual al "byte" fijado por el "dip-switch" la señal P=Q del comparador seva a un nivel bajo y activa el 8255, para habilitar al comparador, se utiliza un decodificar 74LS138 que utiliza las siguientes seis líneas de dirección (A10 a A15) utilizando el minitérmino 0 para generar el pulso de habilitación para el comparador; de acuerdo al mapa de memoria para dispositivos de entrada salida, la dirección asignada para tarjetas prototipo es de la 0300H a la 0FFFH [13] por lo que se selecciona para la tarjeta la dirección base 0300H.

La dirección que corresponde a cada puerto es:

Puerto A 0300H
Puerto B 0301H
Puerto C 0302H

La palabra de control para configurar y programar el dispositivo se escribe en la dirección 0303H.

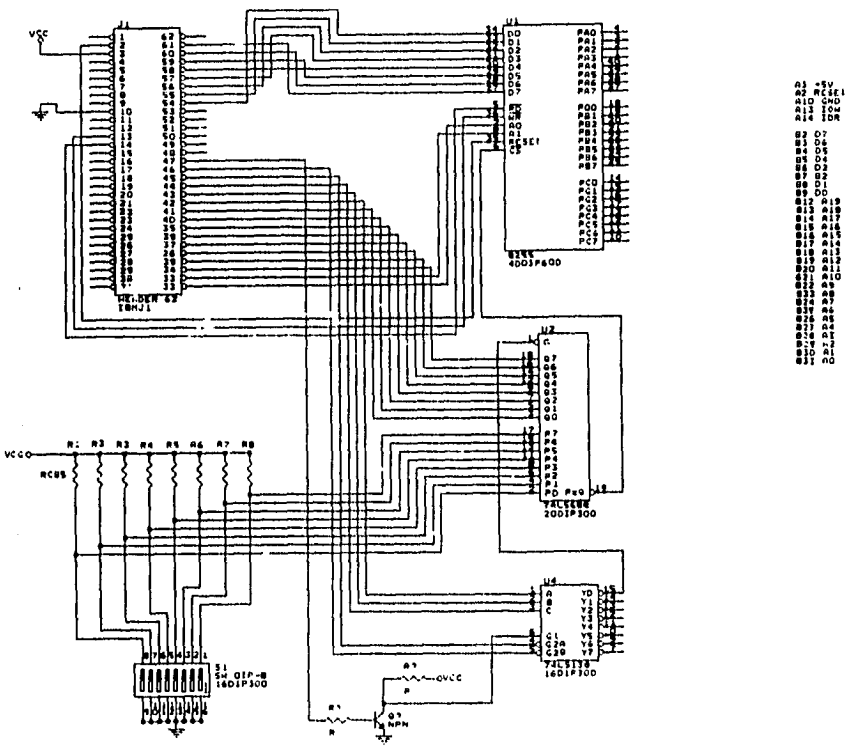


Fig.14 Diagrama eléctrico del PPIO y el direccionamiento

2.7 FUENTES DE POLARIZACIÓN

2.7.1 REQUERIMIENTOS DE POTENCIA

El conjunto de circuitos desarrollados se distribuye en tres tarjetas, en una se encuentra el circuito lógico combinacional para generar la secuencia de pulsos para energizar el motor ,el puerto paralelo y el direccionamiento, esta tarjeta se va a colocar en uno de las ranuras de expansión de la PC y se utilizará la fuente de

+5V de la PC para polarizar estos circuitos; en una segunda tarjeta se encuentran los circuitos combinacionales para manejar la referencia programable y el circuito de potencia para alimentar los relevadores, y por último en una tercera tarjeta se encuentra el circuito de potencia para energizar el motor; en la siguiente tabla se muestra los requerimientos de potencia.

TARJETA	VOLTAJES	CORRIENTE MAX.	DISPOSITIVOS
FUENDRV	+5V	1 A	4 PALS
	+12V	1.5 A	35 relevadores
DRVMOTOR	+5V	3.8 A	motor de pasos

Tabla VI Necesidades de potencia para las tres tarjetas.

Se necesitan tres fuentes, el circuito es el mismo para cada fuente, y se muestra a continuación:

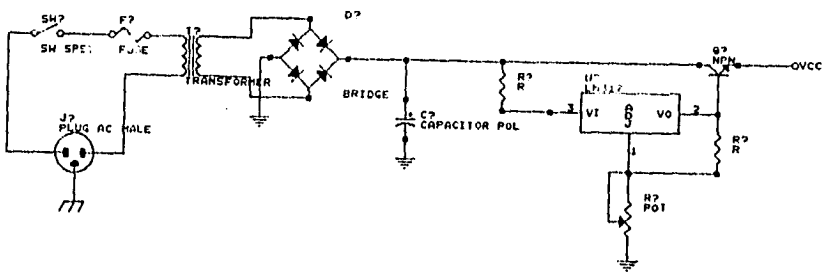


fig. 15 Diagrama eléctrico para las fuentes de alimentación.

2.7.2 CÁLCULO DE COMPONENTES

Para el cálculo de los componentes de la fuente se emplean las siguientes relaciones:

$$V_r(rms) = 2.4\, I_{CC} / C = 2.4\, V_{CC} / (R_L C) \quad (32)$$

$$V_r(rms) = V_r(p\text{-}p) / 2 * 3^{1/2} \quad (33)$$

$$V_r(rms) / V_m = r / (1 + 3^{1/2} r) \quad (34)$$

$$V(rms) = V_m / 2^{1/2} \quad (35)$$

$$r = V_r(rms) / V_{CC} \quad (36)$$

[14]

FUENTE DE 12V , 1.5 A

Los requerimientos para esta fuente son:

$V_o = 12V$
$I_o = 1.5\ A$
$r = 5\%$

El regulador seleccionado para esta aplicación es el LM 317 que es un regulador de voltaje integrado variable, la caída de voltaje en el regulador debe ser de al menos 2.5 V [15]considerando un 20 % como factor de seguridad, se considera $V_I - V_O = 3\ V$, por lo que el voltaje de entrada al regulador debe ser de :

$$V_I = 12 + 3 = 15\ V$$

entonces el voltaje que debe entregar el rectificadfor y filtro debe ser :

$$V_{CC} = 15 + V_r\,(p\text{-}p) / 2 \qquad (37)$$

considerando $V_{CC} = 17\ V$ y despejando $V_r(p\text{-}p)$ de la ecuación (37) por lo que el

$$V_r(p\text{-}p) \leq 4V$$

sustituyendo valores en las relaciones (33) y (36) determinamos que

$$V_r(rms) \leq 1.1154\ V$$
$$r \leq 6.8\ \%$$

por lo que el $r = 5\%$ es un valor adecuado.

Para determinar el valor del capacitor se emplea la relación (32), sustituyendo valores, tenemos que:

$$C = 2.4(1500) / 0.85 = 4235.2\ \mu F$$

el valor más próximo comercial es de 5000 μF.

Para determinar el voltaje efectivo del transformador empleamos las relaciones (36) y (34) , despejando V_m de (34) tenemos que:

$$V_m = (V_r(rms)/r)(1 + 3^{1/2}\,r) \qquad (38)$$

empleando la ecuación (36) en (38) :

$$V_m = (1 + 3^{1/2}\,r)\,V_{CC}$$

sustituyendo valores, tenemos que:

$$V_m = 18.5\ V$$

considerando que la caida de voltaje en cada diodo del puente es de 1 V, el voltaje rms del secundario del transformador, es :

$$V_S = (18.5 + 2) / 2^{1/2} = 14.49\ V$$

$$V_S = 15\ V$$

La corriente que debe soportar el secundario:

$$I_{RMS} = (2^{1/2})\,I_O = 2^{1/2}\,(1.5) = 2.15\ A$$

Para el puente rectificador, la corriente promedio por diodo es de:

$$I_F = I_O / 2 = 1.5 / 2 = 0.75 \text{ A}$$

El voltaje inverso que debe soportar cada diodo

$$V_{inv} \geq 18.5 \text{ V}$$

debido a que la máxima corriente que soporta el regulador es de 1.0 A se necesita un transistorque maneje la corriente, las especificaciones del transistor son:

$$I_{CMAX} \geq 1.5 \text{ A}$$
$$V_{CEO} \geq 18.5 \text{ A}$$

considerando que la mínima corriente para que pueda operar el regulador es de 10 mA, y para que la corriente de base del transisitor sea despreciable con respecto a la corriente mínima de entrada al regulador se considera que

$$I_{BIAS} = 10\, I_{BQ} \qquad (39)$$

de esta relación obtenemos que I_{BQ} = 1mA por lo que :

$$hFE_{min} = 1.5 / 10^{-3} = 1500$$

El transistor seleccionado, es el TIP 127.

FUENTE DE 5 V A 1 A

Siguiendo el mismo procedimiento desarrollado en las lineas anteriores, se determina que para esta fuente , los valores de los componentes son :

Capacitor

C = 5000 µF

Transformador

Vs = 9 V
Irms = 1.4 A

Diodos del puente

Vinv. = 13 V
I prom. = 0.5 A

FUENTE DE 5 V A 3.2 A

Siguiendo el mismo procedimiento desarrollado en las líneas anteriores, se determina que para esta fuente , los valores de los componentes son :

Capacitor

C = 15000 µF

Transformador

Vs = 9 V
Irms = 4.5 A

Diodos del puente

Vinv. = 18.50 V
I prom. = 0.5 A

CALCULO DEL DISIPADOR

Un requerimiento importante para la operación de cualquier dispositivo semiconductor, el cual puede ser un circuito integrado o un transistor, es mantener la temperatura de la unión por debajo del valor máximo especificado dado en las hojas de información, la temperatura de operación esta dada por:

$$T_j = T_A + P_D \theta_{JA} \qquad (40)$$

donde

T_j = temperatura de la unión (° C)
T_A = temperatura del ambiente (° C)
P_D = potencia disipada por el dispositivo (watts)
θ_{JA} = resistencia térmica de la union al ambiente (°C / W)

La resistencia térmica union-ambiente, θ_{JA} , en la ecuación (40) puede ser expresada como la suma de las resistencias térmicas como se muestra a continuación:

$$\theta_{JA} = \theta_{JC} + \theta_{CS} + \theta_{SA} \qquad (41)$$

donde

θ_{JC} = resistencia térmica entre la unión y la envoltura
θ_{CS} = resistencia térmica entre la envoltura y el disipador
θ_{SA} = resistencia térmica entre el disipador y el ambiente

La ecuación (41) se aplica solo cuando se utiliza un disipador de calor externo, si no se emplea un disipador , θ_{JA} es igual a la θ_{JA} de la envoltura del dispositivo dado en las hojas de especificaciones del mismo.[16]

En nuestro caso, el dispositivo que necesita disipador es el transistor que maneja la corriente de salida de cada una de las fuentes, el transistor seleccionado es es el TIP 107 , en las hojas de especificaciones se obtienen :
θ_{JC} = 3 °C / W, para el encapsulado TO 220 utilizando un aislador θ_{CS} = 1.6 °C / W.
Con esta información procedemos a determinar el valor de θ_{SA} para cada transistor ; despejando θ_{SA} de la ecuación (40)

$$\theta_{SA} = ((T_j - T_A) / P_D) - \theta_{JC} - \theta_{CS} \qquad (42)$$

aplicando esta ecuación para la fuente de 12 @ 1.5 A;

la potencia disipada por el transisitor es :

$$P_D = (V_{CC} - V_O) I_O \qquad (43)$$

sustituyendo valores en (43)

$$P_D = (17 - 12)\, 1.5 = 7.5 \text{ W}$$

sustituyendo valores en (42)

$$\theta_{SA} = ((125 - 21) / 7.5) - 1.6 - 3$$

$$\theta_{SA} \leq 12.033 \text{ °C / W}$$

para la fuente de 5V @ 1 A

$$P_D = (10 - 5)\ 1 = 5\ W$$

$$\theta_{SA} = ((125 - 21) / 5) - 1.6 - 3$$

$$\theta_{SA} \leq 16.2\ ^\circ C / W$$

para la fuente de 5V @ 3.6 A

$$P_D = (10 - 5)\ 3.6 = 18\ W$$

$$\theta_{SA} = ((125 - 21) / 18) - 1.6 - 3$$

$$\theta_{SA} \leq 1.177\ ^\circ C / W$$

2.8 ADQUISICIÓN DE LA INFORMACIÓN

Una parte importante en un sistema automático de adquisición de información, son los elementos que se utilizan para llevar la información del mundo real , a una computadora; exiten varios caminos para hacer llegar la información a una computadora, en particular, en este caso se aplica la norma IEEE488 de comunicación por las ventajas que ofrece para la programación y la gran variedad de instrumentos en el mercado que cuentan con un puerto de comunicación bajo esta norma.

2.8.1 EL GPIB IEEE488

El canal de comunicación IEEE488 es un canal de comunicación para instrumentación con normas de programación y circuitería originalmente adoptados por el IEEE (Institute of Electrical and Electronic Engineers) en 1975 dándole la designación IEEE-488 y actualizado en 1978.
El puerto IEEE488 el cual frenecuentemente se le llama GPIB (General Purpose Interfase Bus), fué diseñado como un medio de tranferencia en paralelo para optimizar la transferencia de información sin usar un excesivo número de líneas en el canal de comunicación. Para este fin, el canal cuenta con ocho líneas para la transferencia de información y también para algunos comandos, 5 lineas de administración y tres lineas de protocolo que complementan la señalización del canal.
Una configuración típica se muestra en la fig. 16 . Un sistema típico tiene al menos un controlador y uno o más dispositivos a los que se les envían comandos y, en la mayoría de los casos, de los cuales se recibe información, existen tres categorías que describen la operación de un dispositivo: controlador, transmisor, y receptor. El controlador es un dispositivo que "controla" otros dispositivos conectados al canal de comunicación, un transmisor envía información (usualmente al controlador) mientras que un receptor recibe información. Dependiendo del instrumento, un dispositivo en particular puede ser solo transmisor, solo receptor, o ambos transmisor y receptor.

ESTRUCTURA GENERAL

El principal propósito del GPIB es el de transferir bytes de información entre dos o más dispositivos, la información en el canal es de dos tipos, bytes de comandos y bytes de información. Los bytes de comando son enviados por el controlador activo (uno de los dispositivos del canal, usualmente una computadora) a los dispositivos conectados al canal para definir su configuración, los bytes de datos comprenden la información que es transmitida entre los dispositivos conectados al canal.
La interpretación de los bytes de comando esta estrictamente especificada por la norma IEEE488 y todos los dispositivos compatibles deben responder a estos como lo especifica la norma.
Una de las señales del canal, Atención (ATN), se utiliza para distinguir entre un comando y un dato, cuando ATN, es activado por el controlador activo, toda la actividad del canal se detiene para permitir al controlador activo enviar comandos. Cuando ATN es desactivado, los dispositivos conectados al canal son capaces de enviar y recibir datos.
Antes de que un byte de datos sea transferido entre dispositivos, es necesario especificar que disipositivo envía el dato y cual lo recibe. Esto es especificado por un comando enviado por el controlador activo.
El controlador especifica cual dispositivo va a recibir la información, enviando primero un comando que dice a todos los dispositivos que "no escuchen", posteriormente envía un comando que especifica que dispositivos van a recibir el dato. Estos dispositivos son direccionados como "Listen" , que son los receptores activos, los demás dispositivos ignoran el dato tranferido.
En resumen, para la transferencia de datos dentro del canal IEEE488, el controlador activo:

a) Activa la señal "atención" (ATN) para detener la actividad en el canal.
b) Desactiva el modo "escucha" de todos los dispositivos protegiendo contra escuchas furtivos

c) Designa que dispositivos van a enviar la información y los direcciona como "Talk".
d) Designa que dispositivos van a recibir los datos y los direcciona como " Listen".
e) Permite la transferencia de información desactivando ATN.

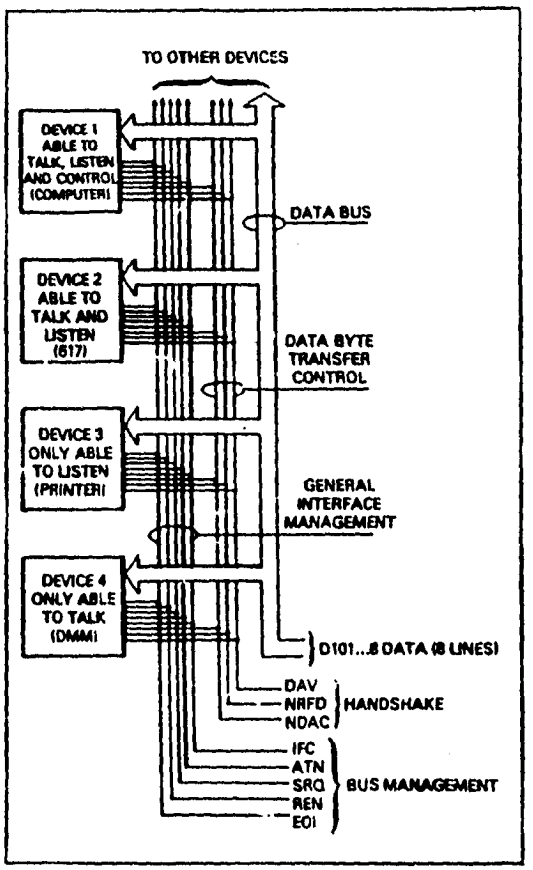


Fig. 16 Configuración típica de un sistema comunicado con norma IEEE488

Una vez que la tranferencia de datos ha sido iniciada, el "transmisor " activo espera hasta que todos los "escuchas" activos esten listos para recibir un byte de información y poner el primer byte en el canal, es entonces cuando la señal de dato válido se presenta y espera hasta que el byte haya sido aceptado por todos los "escuchas", una vez que el byte ha sido aceptado, el "transmisor" pone el siguiente byte en el canal y espera hasta que los "escuchas " den la señal de estar listos para recibir otro byte. Cuando esto ocurre, el "escucha" indica que el dato es válido y la secuencia se repite hasta que la transferencia se completa.

LÍNEAS DEL CANAL

El canal IEEE488 consiste de 16 señales y ocho líneas de tierra, ocho de las líneas de señal son líneas de datos, DIO1 a DIO8, las cuales llevan comandos y datos , un byte a la vez, DIO1 es el bit menos significativo, mientras que DIO8 es el más significativo. Tres de las restantes líneas son líneas de "protocolo" estas señales de control son las de "ready-for-data", "data-valid" y "data accepted" que se utilizan en la secuencia de transferencia de datos. Las cinco líneas restantes se denominan líneas de administración y desarrollan una variedad de funciones importantes.
Cuando la información es puesta en las líneas de datos (DIO1-DIO8), esta puede representar un byte de comando o de dato. Si la línea de administración "atención" (ATN) es activada mientras el dato es transferido, entonces las líneas de datos llevan un comando multilínea que es recibido por todos los dispositivos conectados al canal. Si ATN no es activado, el dato solo es tranferido al dispositivo designado como receptor activo.
El canal IEEE488 también tiene varios comandos monolínea los cuales, como su nombre lo indica, son instrucciones que utilizan una sola línea de administración del canal.

DIRECCIONAMIENTO

La norma IEEE488 normalmente permite hasta 15 dispositivos, incluyendo el controlador, en un mismo sistema. cada uno de estos dispositivos tiene una dirección en el canal, un número entre 0 y 30, que debe ser único para evitar conflictos y confusión. El método por el cual cada dispositivo determina su especificación esta diseñado por el fabricante, algunos emplean un juego de interruptores tipo DIP en la circuitería, otros lo hacen por programación o por un control en el panel frontal .

EL SISTEMA DE CONTROL

El sistema de control es un dispositivo que siempre retiene el control máximo del canal, cuando el sistema es activado por primera vez, el controlador del sistema es el controlador activo y controla todas las transacciones del canal. Es posible para el controlador del sistema pasar el control a otro dispositivo, haciendo de este un nuevo controlador activo, el cual puede, a su vez, controlar a otro dispositivo. Aun cuando el controlador del sistema no sea el controlador activo, el controlador del sistema mantiene el control de la señal de despejar interefaz (IFC, Interface Clear) y activar modo remoto(REN, Remote Enable) y puede tomar el control del canal siempre que lo desee.

LÍNEAS DE ADMINISTRACIÓN

El canal IEEE utiliza 5 líneas de administración, a las señales de estas líneas frecuentemente se les denomina como comandos monolínea, estas señales son activas bajas , es decir un voltaje bajo representa un "1" lógico, y un voltaje alto representa un "0" lógico. Estas líneas pueden ser activadas por cualquier dispositivo conectado al bus, y la señal será activada si algún dispositivo es manejado en la línea. A la inversa, la señal será desactivada si el dispositivo no es manejado en la línea.

Atención (ATN attention)

ATN es una de las líneas más importantes para la administración del canal, cuando ATN se activa, la información contenida en las líneas de datos se interpreta como un comando multilínea, cuando no está activa, la información es interpretada como dato por los dispositivos que esten operando como "escuchas". ATN es manejado por el controlador activo.

Limpiar interfaz (ICF Interface Claer)

El controlador del sistema usa la línea IFC para poner las interfaces de los dispositivos conectados al canal , en un estado conocido. IFC pone los dispositivos en estado "receptor " y "transmisor" ocioso (ni transmisor activo ni receptor activo) y hace que el controlador del sistema sea el controlador activo.

Activar modo remoto (Remote Enable)

El controlador del sistema activa REN para permitir que los dispositivos conectados al canal respondan a los comandos enviados por el control del sistema, si esta desactivado entonces los dispositivos deben ignorar lo que este en el bus.

Identificación o fin (EOI End or Identify)

EOI se utiliza para marcar el último bye o multibyte de información transferido. El dispositivo que envía la información activa EOI durante el último byte de información. EOI no siempre es necesario, debido a que el último byte puede ser un caracter especial tal como un retorno de carro o línea-nueva(line-feed).

Solicitud de servicio (SRQ Service Request)

Esta señal es similar a una interrupción, si se tiene un nivel bajo indica que algún dispositivo requiere servicio del controlador para transmitir datos sincrónicamente. El controlador puede determinar que dispositivo requiere de su atención preguntando a cada uno de ellos o a todos a la vez.

LÍNEAS DE "HANDSHAKE"

El bus utiliza tres líneas para el protocolo, éste asegura la confiabilidad de la transferencia de información a la velocidad del más lento de los receptores. Una línea (DAV) es controlada por el emisor, mientras que las otras dos (NRFD y NDAC) son líneas "wired-or" compartidas por todos los receptores activos, las líneas de "handshake" al igual que las otras líneas del canal IEEE son activas bajas.

Dato válido (DAV)

DAV es controlado por el emisor activo. Antes de enviar cualquier dato, el emisor verifica que NDAC esté activo el cual indica que todos los receptores han aceptado el byte de información previo. El emisor entonces coloca un byte en las líneas de datos y espera hasta que NRFD se desactive lo cual indica que todos los receptores direccionados están listos para aceptar la información. Cuando NRFD y NDAC están en su propio estado, el emisor activa DAV para indicar que el dato en el canal es válido.

No está listo para recibir dato (NRFD Not Ready for Data)

NRFD es usado por los receptores para informar al emisor cuando están listos para recibir un nuevo dato. El emisor debe esperar que cada receptor desactive esta línea (nivel alto) lo cual lo hace cada dispositivo cuando está listo para recibir más información, esto asegura que todos los dispositivos hayan recibido la información y estén listos a recibir más.

No acepta datos (NDAC Not Data Accepted)

NDAC es también controlado por los receptores e indica al emisor que cada uno de los dispositivos direccionados como receptores han aceptado la información, cada dispositivo libera NDAC (alto) a su propia velocidad, pero NDAC no se va a alto hasta que el más lento de los receptores haya aceptado el byte de información.

COMANDOS MULTILÍNEA

Los comandos multilínea son bytes enviados por el controlador activo sobre el bus de datos con ATN activo. Estos comandos se dividen en cinco grupos: grupo de direccionamiento de receptores (Listen Address Group LAG), grupo de direccionamiento a transmisores (Talk Adress Group TAG), grupo de comandos secundarios (Secundary Address Group SCG), grupo de comandos direccionados (Addressed Command Group ACG) y grupo de comandos universales (Universal Command Group UCG).

Grupo de direccionamiento de receptores (LAG)

Estos comandos direccionan como receptores a un dispositivo específico en el bus (20-3E para dispositivos direccionados de 0 a 30) o "Unlisten" todos los dispositivos del bus (3F).

Grupo de Direccionamiento de Emisores (TAG)

Estos comandos direccionan como emisor a un dispositivo en el bus (40-5E para dispositivos direccionados de 0 a 30) o Untalk (UNT) todos los dispositivos del bus (5F).

Grupo de comandos secundarios (SCG)

Estos comandos son usados para especificar una sub-dirección o subfunción dentro de los dispositivos del bus. Estos también son usados en la secuencia de una configuración de escrutinio en paralelo.

Grupo de comandos universales (ACG y UGS)

Estos comandos ejecutan varias funciones del bus. Los comandos direccionables (ACG) afectan solamente al receptor activo, mientras que los comandos universales (UCG) afectan a todos los dispositivos del bus.[17]

2.8.2 CONVERTIDOR ANALÓGICO DIGITAL

Un convertidor analógico a digital es un dispositivo cuya función es acondicionar y transformar una señal analógica a digital; el convertidor utilizado es un convertidor analógico-digital con puerto de comunicación IEEE488 , esto le permite comunicarse a una computadora a través de una interfaz con norma de comunicación IEEE488 lo que permite configurar y manipular el dispositivo a distancia, el ADC utilizado es el ADC488/16 fabricado por IOtech que puede explorar y muestrear señales analógicas a una velocidad de hasta 100 kHz (10 microseg. por "scan" para un canal) con 16 bits de resolución . El ADC488/16 se puede configurar para trabajar con 16 canales de entrada analógica en modo simple u ocho en modo diferencial que se multiplexan con un intervalo programable. pueden guardar hasta 2048 muestras. La memoria puede ser expandida con una tarjeta de memoria RAM de expansión MEMX1 (131,072 muestras) o MEMX2 (262,144 muestras). La capacidad de canales puede ser incrementada si se enlazan y sincronizan dispositivos ADC488 en un arreglo maestro/esclavo. Cuando se conecta a una computadora puede mantener una velocidad de transferencia de información por arriba de 200 kB/s; su precisión es suficiente como para funcionar como un voltímetro digital, y tan flexible en la recolección y registro de

información para funcionar como un osciloscopio de muestreo a baja velocidad. El disparo para la adquisición se puede hacer por medio de un cambio de nivel TTL externo, por un nivel analógico predefinido o por un comando IEEE488 GET o TALK. El "scan buffer" puede ser programado para almacenar muestras antes del disparo, después del disparo, y retardar un número específico de muestras antes del disparo.
El panel frontal muestra con LED's los estados de Trigger, Talk, Listen, SRQ, Error, Test y Potencia. Los controles del panel anterior incluyen el interruptor de energía y un interruptor tipo DIP para fijar la dirección del dispositivo en el bus IEEE488. Además se tienen los conectores para energía, disparo externo (BNC), bus IEEE488, I/O digital, y entradas analógicas.

Características analógicas:

- 16 entradas en modo sencillo o en modo diferencial.
- convertidor A/D de 16 bits y velocidad de muestreo de 100 kHz.
- Voltaje de aislamiento entre canal y canal en modo común (CMV) de 10 V.
- 250 V de CMV de aislamiento de canal a tierra (común digital).
- Cuatro intervalos bipolares programables (±1, ±2, ±5 y ±10 v).
- Precisión basica (23 ± 5°C):

Intervalo		Resolución	Precisión (% a escala completa)
±1 volt	16bit	33 μV/bit	±(0.02%)
±2 volt	16bit	66 μV/bit	±(0.02%)
±5 volt	16bit	166 μV/bit	±(0.02%)
±10 volt	16bit	333 μV/bit	±(0.02%)

- razón de rechazo de modo común (mayor de 70 dB de DC a 100Hz).
-Calibración Digital.

Caracteristicas de colección de información y almacenamiento:

- Salida binaria en el bus IEEE 488 mayor que 200kB/s
-buffer de 2Kbytes
-expansión de RAM opcional (hasta 256 K lecturas)
-formatos de salida programables:

-ASCII punto fijo
-ASCII hexadecimal
-Decimal
-Binario

-Multiples fuentes de disparo:

-Por nivel de entrada analógica
-Señal externa (nivel TTL)
-GET
-TALK
-Modos de disparo programable "one-shot" y continuo.[18]

SELECCION DE LA DIRECCIÓN DEL DISPOSITIVO EN EL BUS IEEE 488

La dirección se fija por medio del interuptor S104-1 al S104-5 (localizado en la parte posterior de la unidad). La dirección puede fijarse entre 0 y 30. La dirección se selecciona por el peso binario de cada interruptor, donde 1 es el bit menos significativo y el interruptor 5 es el bit más significativo, la dirección por omisión es la 14; si la dirección seleccionada es la 31, toma el valor por omisión de 30 porque la norma IEEE 488 se reserva la dirección 31; en la figura (16) se muestra el interruptor S104 con el valor de la dirección selecionada.

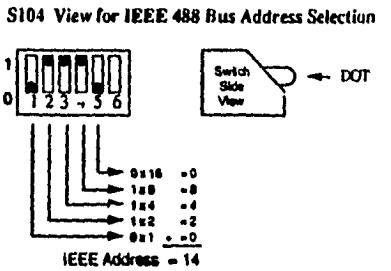


Fig.16 Vista del interruptor para la selección de la dirección.

2.8.3 ELECTRÓMETRO

El electrómetro programable Keithley modelo 617 es un instrumento altamente sensitivo diseñado para medir voltaje, corriente, carga, y resistencia. Para medir resistencia cuenta con dos métodos para medir: uno, el método de corriente constante , y el otro, de voltaje constante que utiliza una fuente de voltaje interconstruida para mayor sensibilidad. Los intervalos de medición estan:

-entre 10 μV y 200V para voltaje,
-0.1 fA y 20 mA en el modo de corriente,
-0.1 W y 200 GW (hasta 10^{16} W usando la fuente de voltaje interconstruida),
-10 fC y 10 nC en el modo carga.

La muy alta impedancia de entrada y la extremadamente baja corriente de entrada de "offset" permite mediciones exactas en situaciones donde muchos otros instrumentos pueden tener efectos nocivos en el circuito de medición. Un indicador de 4 1/2 digitos y una interfaz IEEE 488 dan al usuario fácil acceso a la información del instrumento.
Sus principales características incluyen:

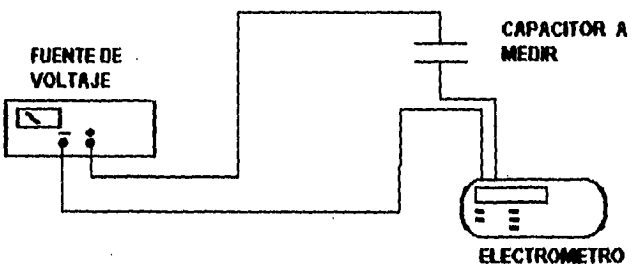
Indicador de 4 1/2 digitos, un panel de "leds" en la parte frontal con una mantisa de 4 1/2 dígitos, más dos dígitos para determinar el exponente alfanumérico.
Autointervalo- Incluido en todas las funciones e intervalos.
Calibración digital- El instrumento puede ser calibrado digitalmente a través del panel frontal o utilizando la interfaz de comunicación IEEE-488.
Corrección del cero- Un control en el panel frontal permite al usuario cancelar cualquier "offset".

Supresión de línea base- Una función para suprimir la lectura de línea de base está disponible en el panel frontal o a través de la interfaz IEEE-488.
Disparo "One-shot"- en el panel frontal se tiene un botón para tomar una lectura cada vez que éste es presionado.
Fuente de voltaje aislada de 100V- El instrumento cuenta con una fuente de voltaje interconstruida aislada de la sección del electrómetro que puede programarse en pasos de 50 mV.
Protección seleccionable- un "driver" de cable de guarda se incluye para optimizar velocidad y blindaje.
Interfaz IEEE-488 -La interfaz permite completa operación, programable a través del bus IEEE-488.
Salidas analógicas - preamp y 2V de intervalo completo se incluyen en el panel posterior.
Memoria para almacenar 100 datos- Un "buffer" interno permite almacenar hasta 100 lecturas y se puede acceder desde el panel frontal o desde la interfaz IEEE488.[19]

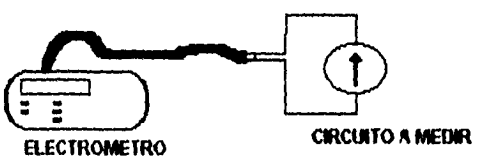
Para la aplicación que se requiere, este instrumento va a servir para :

medir dosis abosorbida midiendo ya se sea carga o corriente.
medir capacitancia para estimar la distancia entre placas midiendo la carga almacenada.

Básicamente requerimos de las funciones de medir corriente y medir carga , mismas que se contemplan en el programa de operación del sistema, el circuito de medición para medir carga y corriente se muestra en la fig. (17)



b)



a)

Fig. 17 Circuitos de medición para medir a) carga , b)corriente.

La parte de programación proporciona y ordena las rutinas para generar las secuencias de operación necesarias para cumplir con el objetivo del sistema que es el de adquirir información, generar las instrucciones necesarias para mover el motor y fijar el valor del voltaje de la fuente ; el lenguaje de programación utilizado es el lenguaje C.
C es un lenguaje de programación de empleo general, caracterizado por su concisión y por poseer un moderno control de flujo y estructuras de datos, así como un rico conjunto de operadores. No es un lenguaje de muy alto nivel ni grande, no está especializado en un área de aplicación en particular. Su carencia de restricciones y su generalidad lo hacen más eficaz y conveniente para muchas tareas a diferencia de otros lenguajes, supuestamente más potentes.[20]

2.9.1 DIVISION GENERAL DEL PROGRAMA

El programa se divide en tres funciones principales , una es la parte de adquisición de datos, en este bloque de funciones se encuentran las rutinas para la adquisición de información para la construcción de curvas de extrapolación y la rutina de adquisición de información para otras aplicaciones, en general,estas rutinas fijan las características de operación del electrómetro, del ADC y del dispositivo digital, toman información del electrómetro y del ADC a través del bus IEEE-488 para ser almacenada en un archivo en un dispositivo de almacenamiento magnético, y la generación de las señales para el ajuste de la distancia y del voltaje de la fuente, La función de "autotest" tiene una rutina para medir la distancia entre placas y verificar el error en la distancia, y una rutina para generar diferentes voltajes y medir cada uno de ellos utilizando como instrumento de medición el ADC; por último, la función de imprimir información permite desplegar la información en pantalla obtenida anteriormente y si se desea enviar esta información a una impresora o a un archivo de tipo texto.
El diagrama de flujo general se muestra en la fig. (18) , el listado global del programa se muestra en el anexo C.

2.9.2 ADQUISICIÓN DE INFORMACIÓN

En la función de adquirir información se tienen 2 rutinas. la primera es una rutina para adquirir información para la construcción de curvas de extrapolación para la determinación de la dosis absorbida por tejido, la segunda rutina es un algoritmo para adquirir información destinada a otras aplicaciones distintas a las de construcción de curvas de extrapolación, El diagrama de fllujo de esta opción del menu se muestra en la fig. 19.

2.9.2.1 CURVAS DE EXTRAPOLACIÓN

La rutina para curvas de extrapolación obtiene información de corriente de ionización o carga entre las placas de la cámara de extrapolación, toma lecturas de factores ambientales (temperatura, presión y humedad) y mide el voltaje aplicado entre placas, esta información es guardada en un archivo con extensión .DAT , la adquisición de la información se hace durante una secuencia de acciones en las que se hace mover el motor para que las placas de la cámara de extrapolación se desplacen y se ajuste la fuente de voltaje al valor deseado. La información que el algoritmo necesita es :

nombre del archivo de trabajo: Es el nombre del archivo donde se guarda la información una vez terminado el proceso.
nombre del usuario: Es el nombre de la institución, empresa o persona que solicita el servicio.

tipo de fuente: Es el nombre del elemento radiactivo que se utiliza como fuente de emisión de radiaciones.

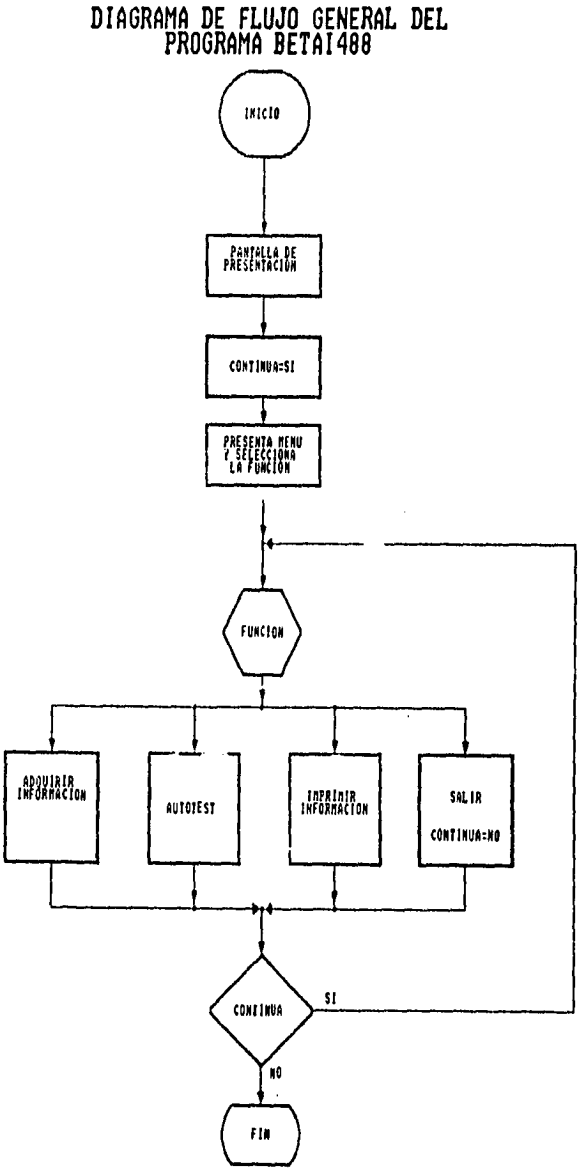


Fig. 18 Diagrama de flujo general del programa BETA488

distancia inicial: Es la distancia en la que se inicia el barrido de distancia, la distancia entre placas máxima permitida es de 20 000 micras y la mínima, para evitar que las placas se lleguen a tocar mientras se encuentran polarizadas, o se dañe la película de la ventana es de 100 micras.
distancia final: Es la distancia hasta donde van a llegar las placas al final del proceso.
paso: Es el intervalo en micras en que se va a cubrir la distancia comprendida entre la distancia inicial y la distancia final.
gradiente: Es la relación entre distancia y voltaje que determina el voltaje a aplicar para polarizar las placas de la cámara de extrapolación, el valor máximo permisible es de 50 V/cm.
número de lecturas: Es el número de veces que el electrómetro va a medir en cada intervalo comenzando en la distancia inicial y terminado en la distancia final.
tiempo entre lectura y lectura: Es el lapso entre lectura y lectura en el electrómetro dado en segundos, el máximo lapso permisible es de 65 535 segundos (aprox. 1 lectura cada 9 horas máx.).
función e intervalo del electrómetro: las funciones que se utilizan en esta aplicación son las de medir carga y las de medir medir corriente, y el intervalo determina la corriente máxima que se puede medir.

Una vez que el usuario proporciona esta información al sistema, lo primero que se hace es ajustar la distancia entre placas a la distancia inicial, se ajusta el voltaje y se mide el fondo que se comsidera como ruido en el sistema y que permite detectar alguna falla en las conexiones.Si el nivel de fondo es aceptado, entonces se procede a iniciar el proceso de medición , una vez terminado; la información adquirida se guarda en el archivo bajo el nombre que se proporciona al inicio de la rutina, el algoritmo de esta función se muestra en la fig. (20)

2.9.2.2 OTRAS APLICACIONES

Esta rutina inicia pidiendo al usuario el nombre del archivo donde se va a guardar la información recabada durante el proceso, posteriormente se pide al usuario cual es la distancia a la que se desea se ajusten las placas y el gradiente a utilizar para ajustar el voltaje de polarizaciónde las placas de la cámara de extrapolación, en esta primera distancia se mide el fondo y si el valor promedio es aceptado por el operador, el programa inicia las operaciones para tomar información, el número de lecturas y el tiempo entre cada una de las lecturas se fija y estos valores se conservan subsecuentemente, se pide al usuario fije la distancia y el gradiente a utilizar, la rutina continua hasta que el usuario no desea seguir continuando, en esta rutina además de guardarse las lecturas del electrómetro , los valores de los factores ambientales y el voltaje aplicado, también se guardan cada una de las distancias utlizadas y el gradiente de voltaje utilizado en cada distancia. Para finalizar la rutina, la información es guardada en el archivo abierto al inicio de la rutina; por último las placas de la cámara de extrapolación se regresan a la distancia inicial de 20 000 micras y la fuente de voltaje se ajusta a cero volts.
El diagrama de flujo de esta rutina se muestra en la fig. (21)

2.9.3 AUTOTEST

En esta rutina se determina el área efectiva del electrodo colector y la separación efectiva de los electrodos.
Empleando las expresiones que definen la capacidad eléctrica y el cálculo de la misma a partir de las dimensiones geométricas para un capacitor de placas paralelas planas:[21]

$$C=\Delta C/\Delta V \qquad (44)$$

$$C=\varepsilon_0(\Lambda/d) \qquad (45)$$

FUNCION ADQUIRIR_INFORMACION

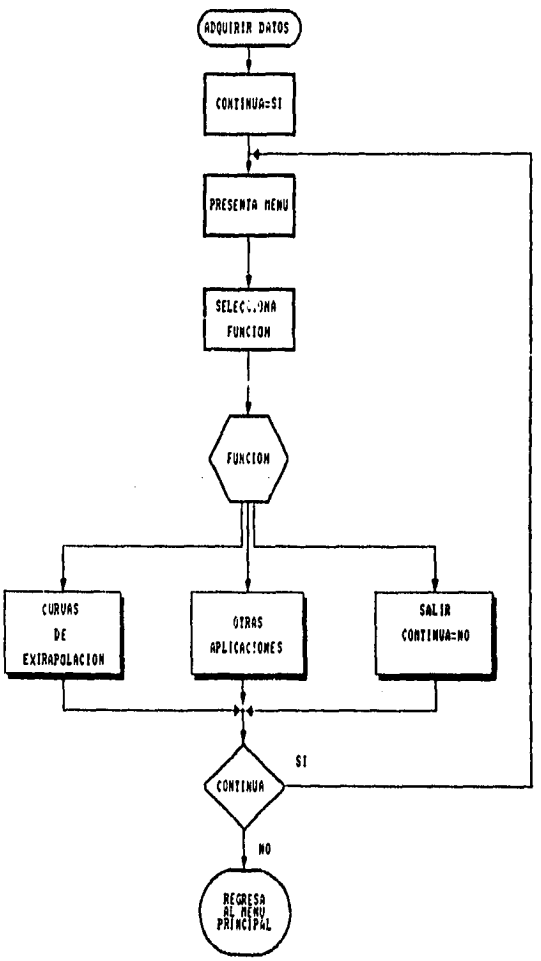


Fig. 19 Diagrama de flujo de la función ADQUIRIR INFORMACIÓN.

FUNCION CURVAS_DE EXTRAPOLACION

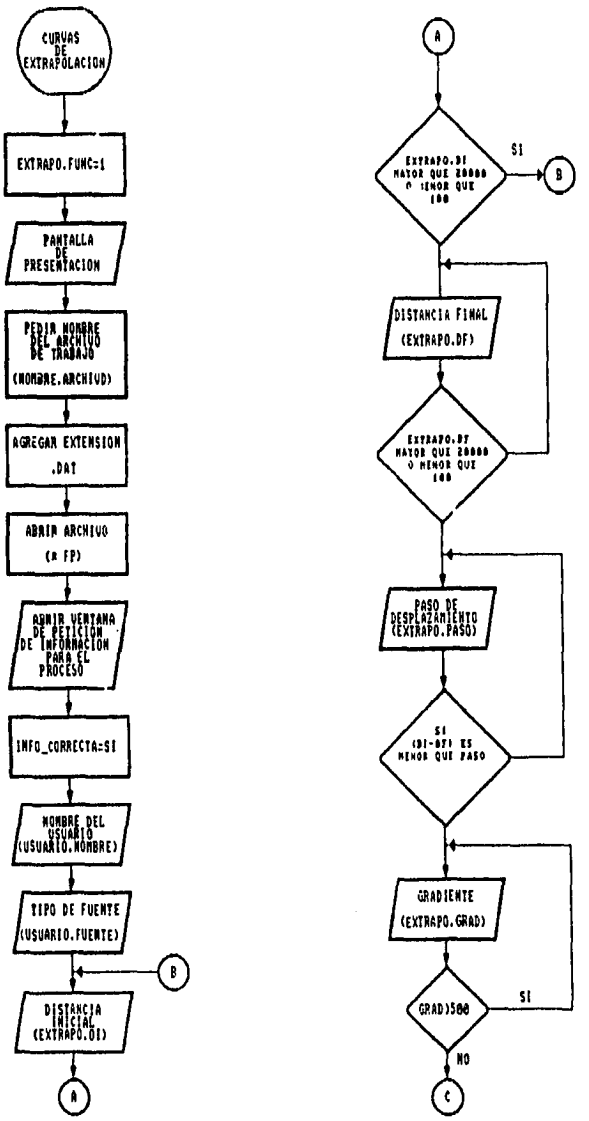


Fig. 20 Diagrama de flujo de la rutina CURVAS DE EXTRAPOLACIÓN

DIAGRAMA DE FLUJO FUNCION CURVAS DE EXTRAPOLACION (CONTINUACION)

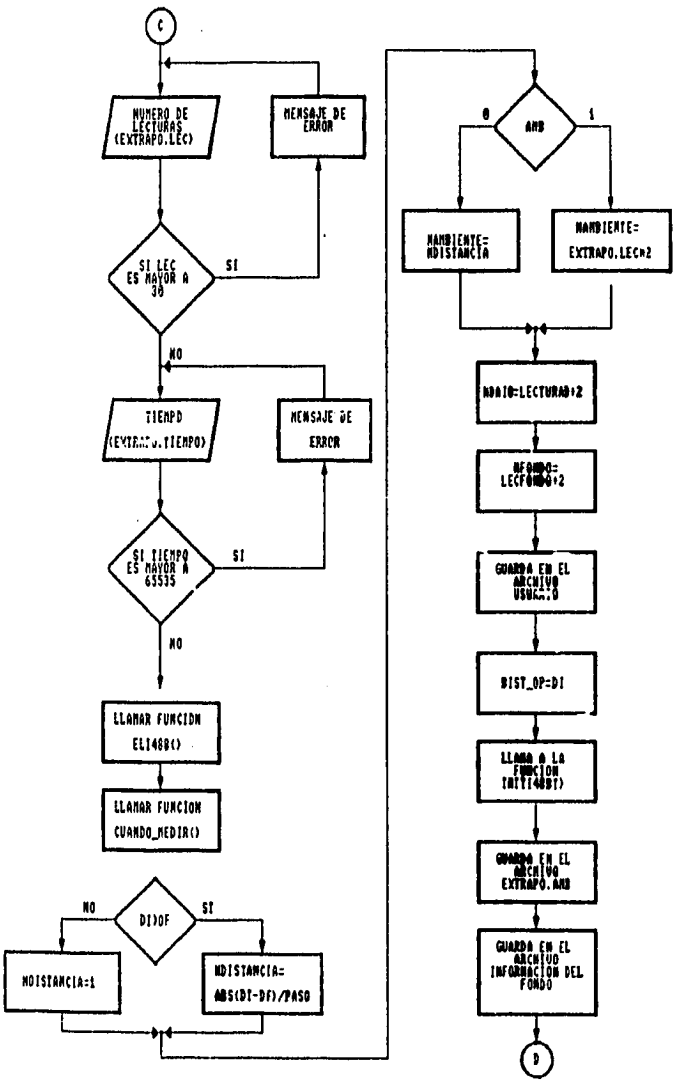


Fig. 20A Diagrama de flujo de la rutina CURVAS DE EXTRAPOLACIÓN

DIAGRAMA DE FLUJO DE LA FUNCION CURVAS DE EXTRAPOLACION
CONTINUACION

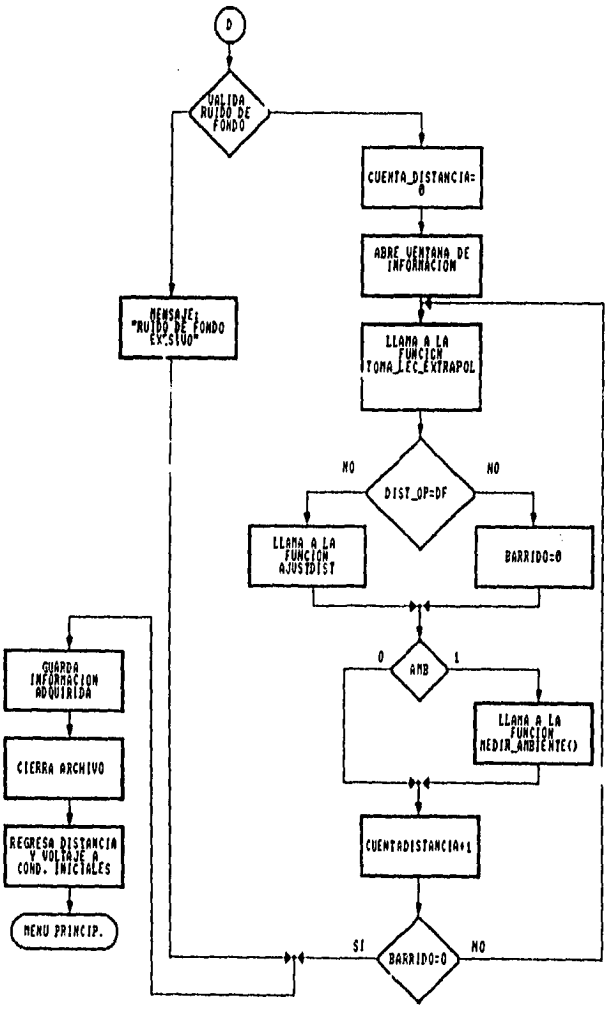


Fig. 20B Diagrama de flujo de la rutina CURVAS DE EXTRAPOLACIÓN

DIAGRAMA DE FLUJO FUNCION: OTRAS_APLICACIONES

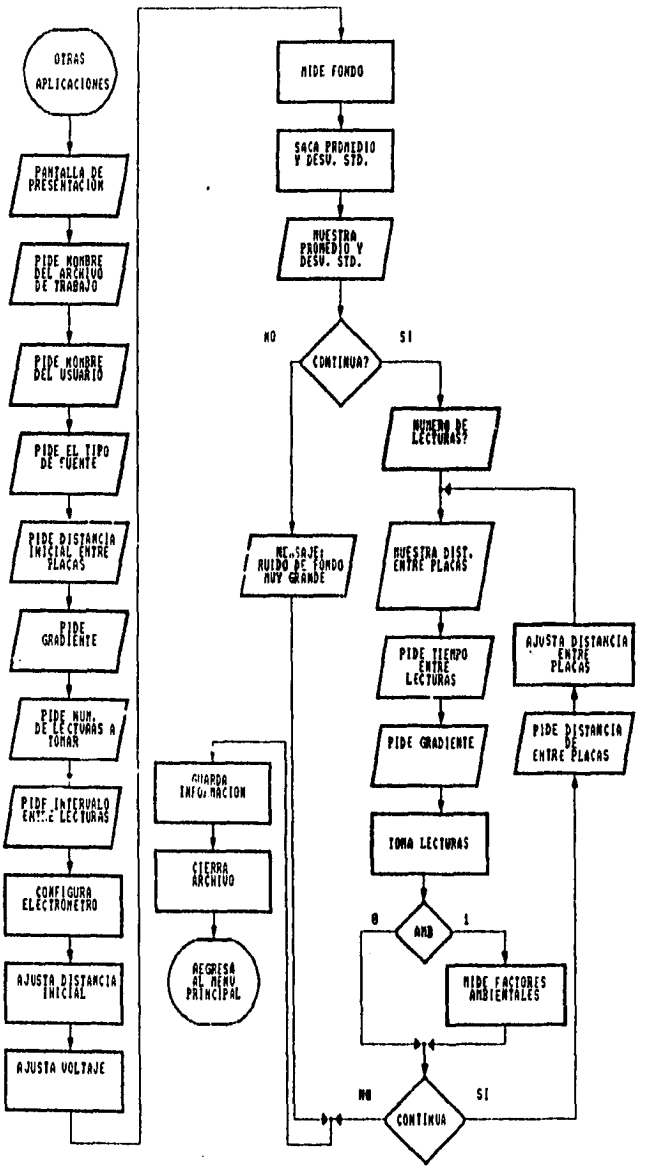


Fig. 21 Diagrama de flujo de la rutina OTRAS APLICACIONES

donde :

A : área de una placa del condensador.
d :separación de las placas
ΔC : incremento de carga almacenda.
ΔV : incremento en el voltaje aplicado.

despejando d de (45) se tiene:

$$d = A \cdot \varepsilon_0 \cdot C^{-1} \qquad (46)$$

Esta ecuación se puede ajustar con un modelo de regresión lineal simple mediante la siguiente expresión:

$$d = d_0 + A_0 \cdot \varepsilon_0 \cdot C^{-1} \qquad (47)$$

Donde :

d_0 : es la separación efectiva de las placas cuando el tornillo micrométrico de cámara de extrapolación marca la posición cero
A_0 : es el área efectiva del electródo colector de la cámara de extrapolación.

Para determinar d_0 y A_0 , la ecuación (47) tiene la forma de la expresión general para una recta :

$$Y = mX + b \qquad (48)$$

Donde :

m : pendiente de la recta.
b : ordenada al orígen.

de tal modo que un conjunto de datos se pueden ajustar a una recta utilizando las ecuaciones:[22]

$$m = \sum^{i}_{n=0} (X_i - X) (Y_i - Y) / \sum^{i}_{n=0} (X_i - X)^2 \qquad (49)$$

$$b = Y - mX \qquad (50)$$

como la forma de la ecuación (47) es semejante a la ecuación (48), podemos hacer el ajuste de un conjunto de datos de capacitancia y distancia con las siguientes ecuaciones:

$$A_0 \cdot \varepsilon_0 = \sum^{i}_{n=0} (C^{-1} - C^{-1}) / \sum^{i}_{n=0} (d_i - d)^2 \qquad (51)$$

$$d_0 = d - A_0 \cdot \varepsilon_0 \cdot C^{-1} \qquad (52)$$

Para calcular C^{-1} se utiliza la expresión (44), es decir, para una separación entre placas d, se polarizan los electrodos con un V_0 (0 V), se mide inmediatamente la carga Q_0 , a continuación se polariza la cámara de extrapolación con un voltaje V_1 , y se mide la carga Q_1 , para cada aplicación de voltaje a la misma distancia se toman 5 lecturas de carga y se mide el voltaje respectivo , estos valores se guardan en un arreglo para su uso posterior .

Este procedimiento se repite para una serie de distancias d y se obtiene el respectivo valor de C^{-1}; las distancias entre electrodos son a 10.0 mm, 5.0 mm, 2.0 mm, 1.0 mm, y 0.5 mm.
El diagrama de flujo para esta rutina se muestra en la fig. (22)

CALIBRACIÓN DE VOLTAJE

Para la calibración del voltaje se utiliza el ADC como voltímetro y se van fijando valores de voltajes en decadas, es decir 0.1 , 0.2, hasta 1V, en 1 V se incrementa de uno en uno hasta 10 V , en 10V se incrementa de 10 en 10V hasta 100, y en 100V hasta 300, cada vez que se fija el voltaje se mide, esta información nos permite obtener una gráfica de voltaje "propuesto " contra voltaje medido.
El diagrama de flujo para esta rutina se encuenrtra en la fig. (23)

2.9.4 IMPRIMIR INFORMACION

Esta rutina toma la información de un archivo generado por alguna de las rutinas de adquisición de información y la presenta en la pantalla del monitor, si el usuario lo desea puede enviar la información a la impresora o enviarla a un archivo con extención .ASC de tipo texto para que pueda ser utilizado por una hoja de cálculo, el archivo con extensión .DAT se encuentra enpaquetado en binario por lo que esta rutina debe transformar la información a caracteres para que se envie a la impresora o al archivo con extensión .ASC, la estructura del archivo en binario se muestra en la fig. (24).

FUNCIONES AUXILIARES

Dentro del programa se encuentran rutinas que por ser utilizadas en varios bloques del programa se definieron como funciones, estas funciones son:

AJUSTAR DISTANCIA

Esta función genera el tren de pulsos que activa al circuito secuencial que genera la palabra de control para energizar las bobinas del motor, esta rutina calcula el número de pulsos necesarios para mover las placas de la cámara de extrapolación a la distancia deseada, a partir de la distancia de operación (distancia a la que se encuentran las placas en el momento de hacer el ajuste de la distancia), debido a los esfuerzo mecánicos del acoplamiento entre el motor de pasos y el tornillo micrométrico, en esta función se inserta una rutina para hacer la correción del error, el diagrama de flujo de esta función se muestra en la fig. (25).

AJUSTAR VOLTAJE DE LA FUENTE DE ALIMENTACIÓN

Esta rutina genera las palabras de control para los circuitos combinacionales, generando el incremento de voltaje en escalera, y dependiendo de la polaridad de voltaje deseada permite tambien la inversión del mismo.
El voltaje a aplicar se calcula utilizando la relación del gradiente :

$$V = G / D \qquad (53)$$

donde:

V: es el voltaje a aplicar.
G: es la relación entre el voltaje y la distancia y se encuentra dado en [v/cm].
D: distancia entre placas dada en [cm].
el diagrama de flujo de esta rutina se muestra en la fig. (26)

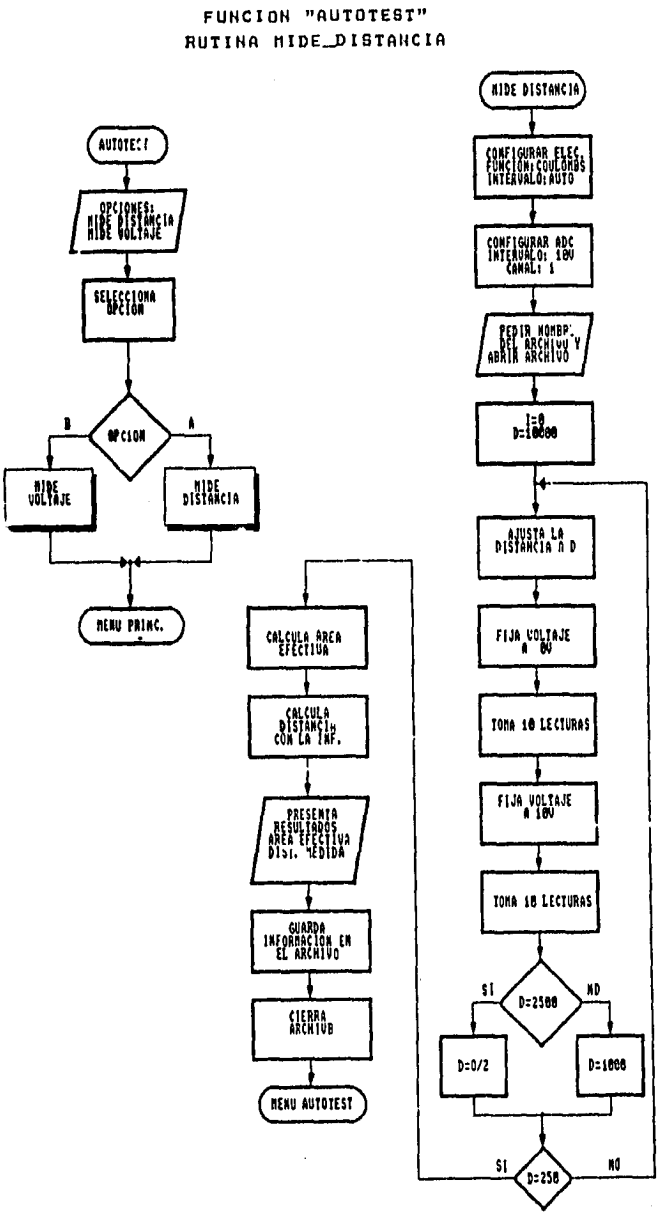


Fig. 22 Diagrama de flujo de la rutina MIDE DISTANCIA

FUNCION "AUTOTEST"
RUTINA MIDE_VOLTAJE

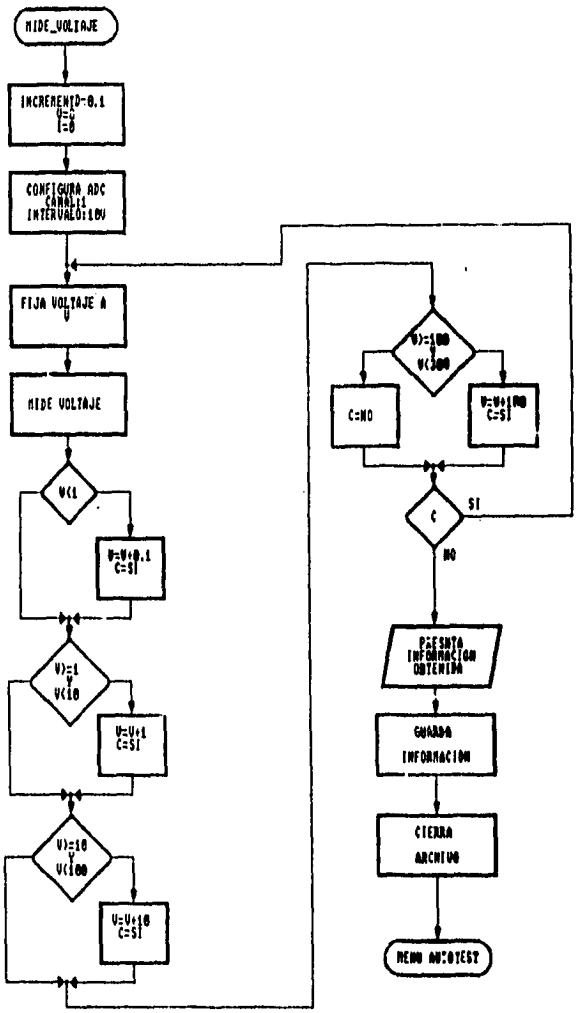


Fig. 23 Diagrama de flujo de la función MIDE VOLTAJE

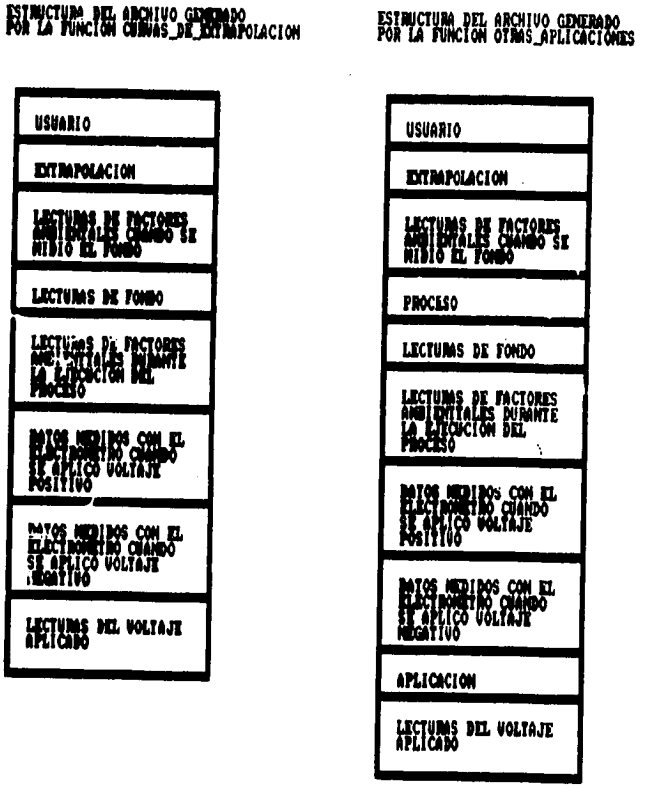


Fig. 24 Estructura del archivo

FUNCION AJUSDIST

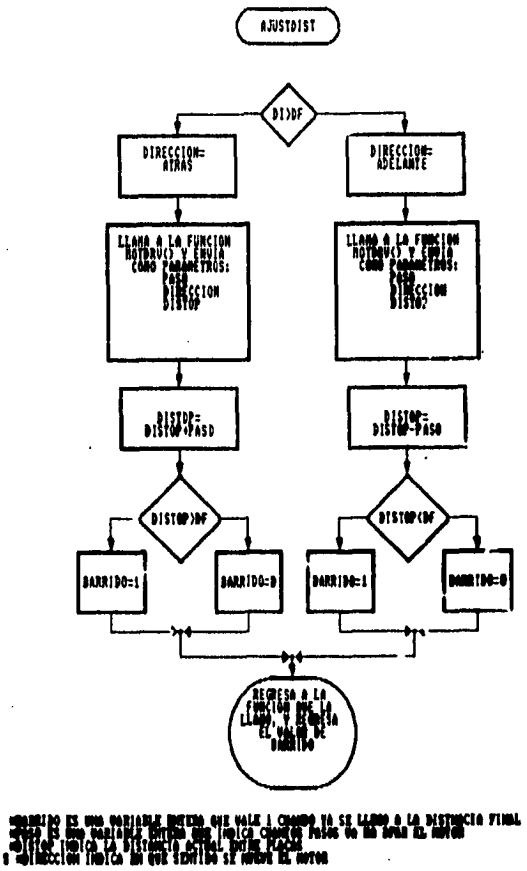


Fig. 25 Diagrama de flujo de la función AJUSDIST

FUNCION DRVMOTOR

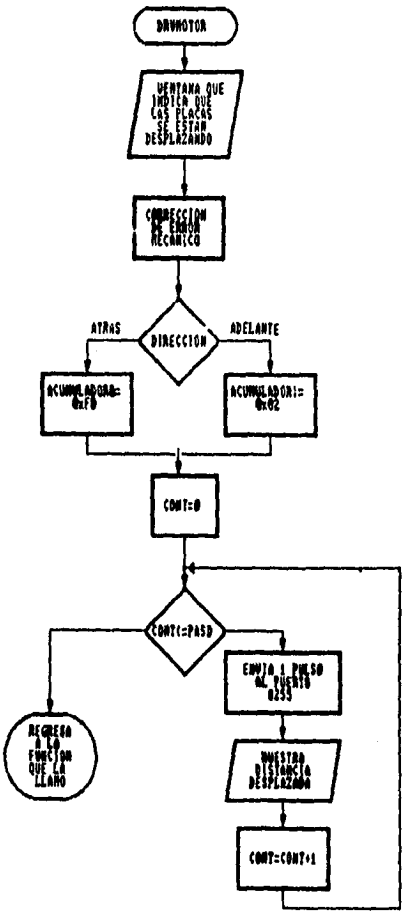


Fig. 25A Diagrama de flujo de la función DRVMOTOR

FUNCION CONTHV

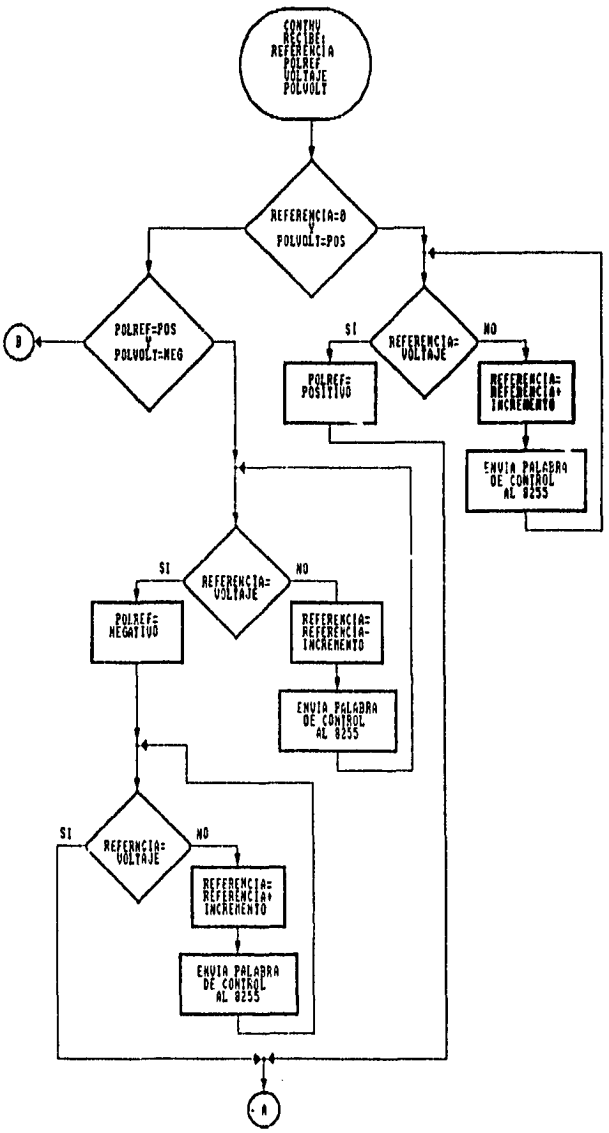
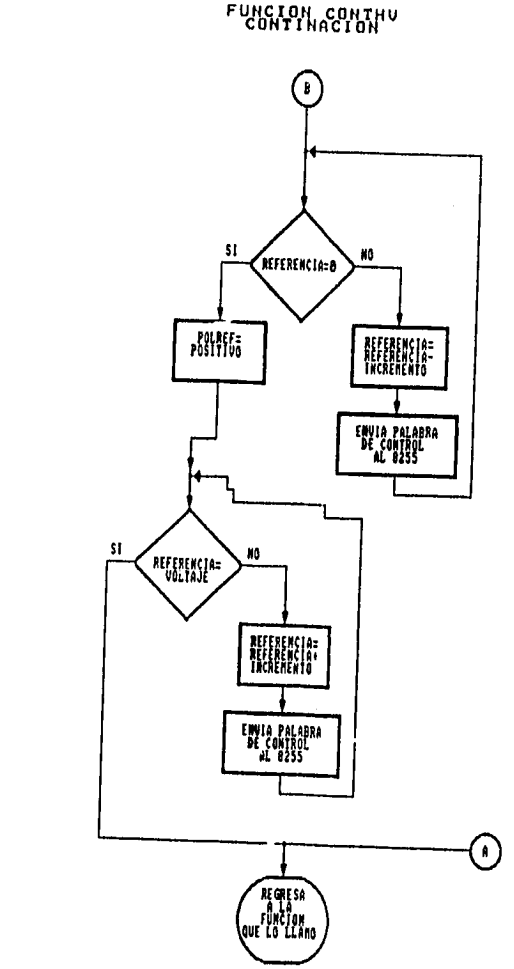


Fig. 26 Diagrama de flujo de la rutina CONTHV



*REFERENCIA GUARDA EL VALOR DEL VOLTAJE ACTUAL
*POLREF GUARDA LA POLARIDAD DEL VOLTAJE ACTUAL
*VOLTAJE ES EL VALOR DEL VOLTAJE QUE SE DESEA APLICAR
*POLVOLT ES LA POLARIDAD DEL VOLTAJE QUE SE DESEA APLICAR

Fig. 26A Diagrama de flujo de la función CONTHV

3 CONSTRUCCIÓN

El sistema SICE (Sistema de Interfaz para la Cámara de Extrapolación)está integrado por una parte electrónica y una parte mecánica; la parte electrónica se distribuye en 4 tablillas de circuito inpreso:

- Tarjeta BETA.
- Tarjeta CONTHV.
- Tarjeta MOTDRV.
- Tarjeta FUENDRV.

La parte mecánica, la componen el soporte para el motor y el acoplamiento entre la flecha del motor de pasos y el tornillo micrométrico de la CE (Cámara de Extrapolación).

3.1 CIRCUITOS IMPRESOS

3.1.1 TARJETA BETA

Esta tarjeta se diseña para contener los circuitos correspondientes al PPIO, el direccionamiento del mismo, el circuito lógico secuencial para mover el motor de pasos y los optoacopladores para aislar la etapa de potencia de la etapa digital, esta tarjeta se inserta en cualquiera de las ranuras de expansión en la PC, utilizando la norma ISA que define el tamaño del peine de conexión así como la identificación de cada terminal en el peine, en la fig .27 se muestra la identificación de las terminales.

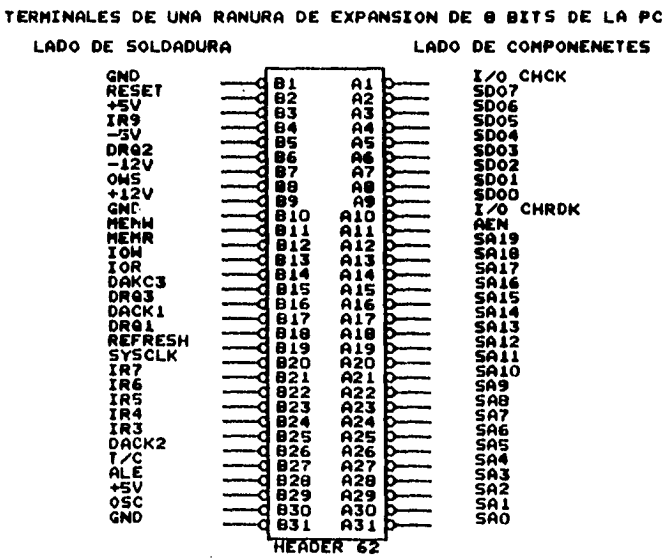


Fig. 27 Identificación de terminales del bus ISA

Las dimensiones de la tarjeta y la distribución de loscomponentes de muestran en la fig. 28, el diseño del circuito impreso se muestra en el anexo A.

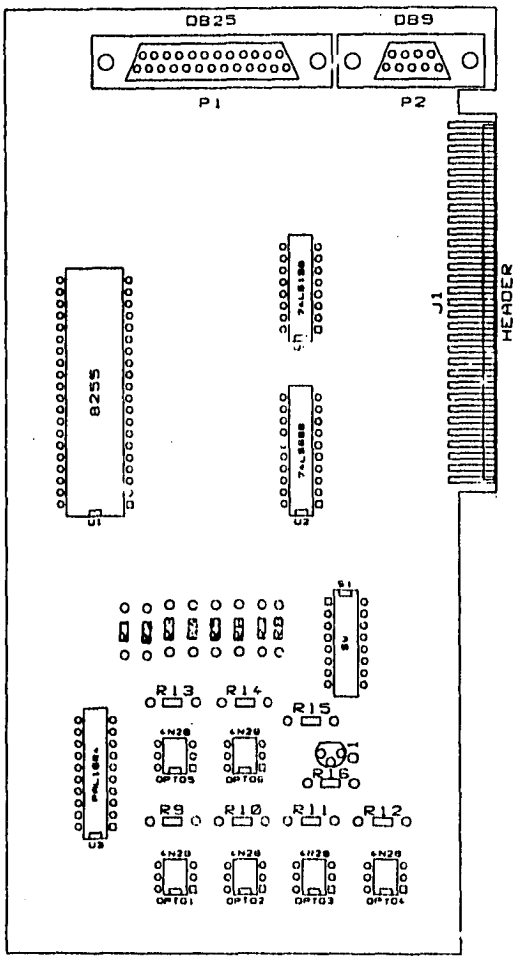


Fig. 28 Distribución de componentes y dimensiones de la tarjeta BETA

3.1.2 TARJETA MOTDRV

La tarjeta motdrv alberga la etapa de potencia para alimentar las bobinas del motor de pasos. Enla fig. 29 se muestra dicha tarjeta.

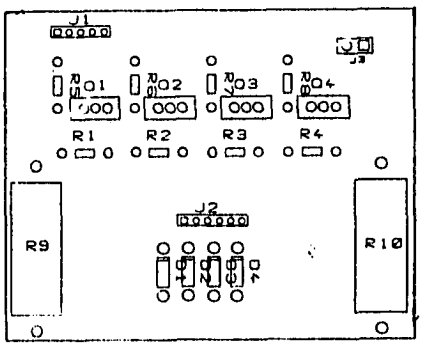


Fig. 29 tarjeta MOTDRV

3.1.3 TARJETA FUENDRV

En la tarjeta FUENDRV se ubican las tres fuentes de alimentación que proporcionan la polarización al sistema, 2 alimentan a la tarjeta CONTHV y la restante proporciona energía al motor de pasos.

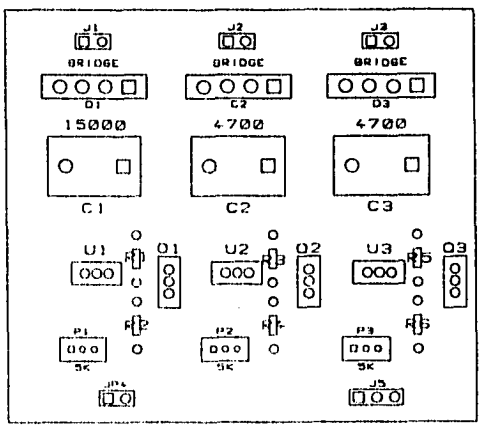


Fig.30 tarjeta FUENDRV

3.1.4 TARJETA CONTHV

En esta tarjeta se ubican, los circuito lógicos combinacionales para la conmutación de los relevadores que seleccionan las resistencias de retroalimentación,y los bancos de resistencia, así como un relevador que se utiliza para invertir la polaridad del potencial de polarización, las señales llegan a esta tarjeta a través de un conector db25, las dimensiones de esta tarjetay la distribución de componentes se muestra en al fig. 31.

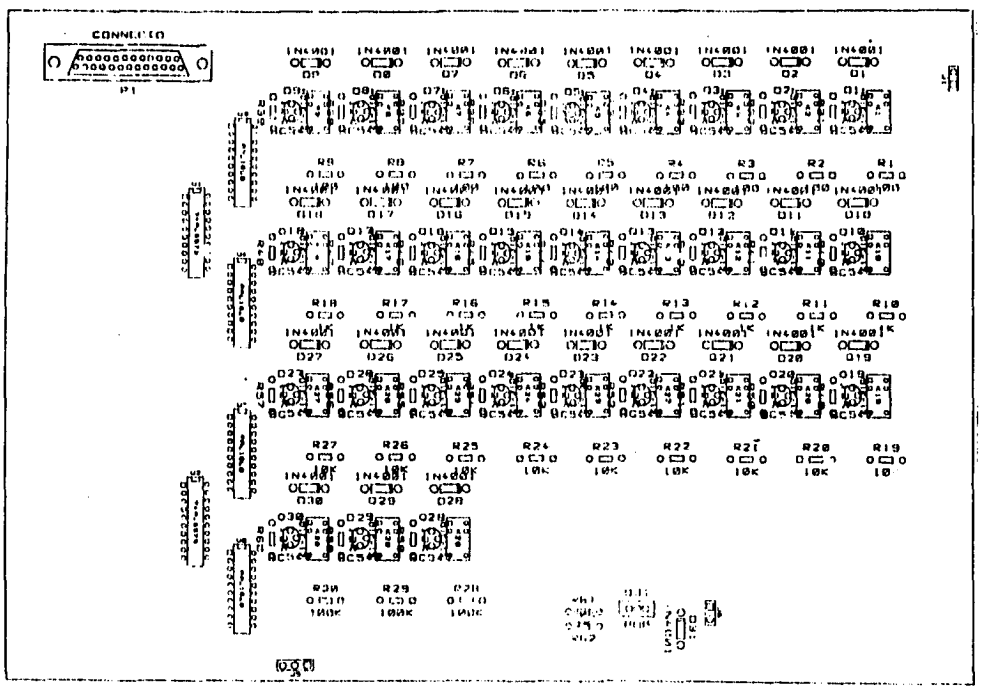


Fig. 31 a) Dimensiones físicas y distribución de componentes tarjeta CONTHV

3.2 GABINETE Y ARREGLO MECÁNICO

3.2.1 GABINETE

Dentro del gabinete se acomodan tres tarjetas, la tarjeta CONTHV, MOTDRV ,FUENDRV , además, el transformador que proporciona la energía de la línea a la tarjeta FUENDRV y un filtro de línea para interferencia electromagnética, en la parte posterior del gabinete se encuentran conectores tanto de entrada como de salida, en las figuras 32 y 33 se ilustra el gabinete.

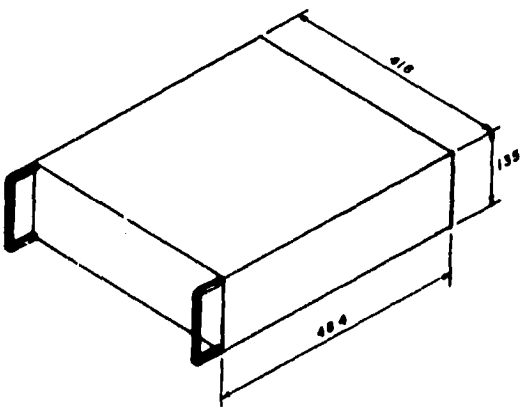


Fig. 32 Vista en isométrico del gabinete (cotas en mm)

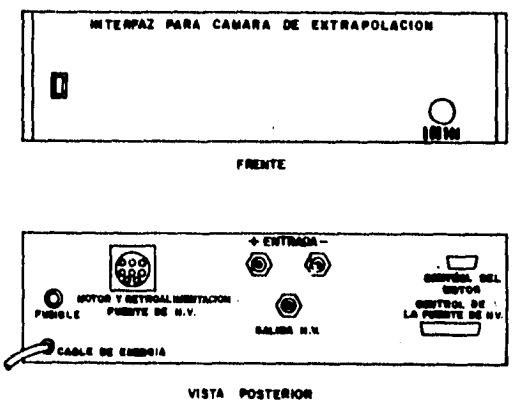


Fig. 33 Vista anterior y posterior del gabinete

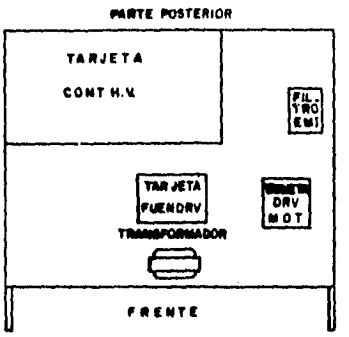


Fig. 34 Ubicación de las tarjetas dentro del gabinete

3.2.2 ARREGLO MECÁNICO

El arreglo mecánico lo componen, el soporte para el motor de pasos y el acoplamiento mecánico, en la fig. 35 se muestra el conjunto, montado en la plataforma de la cámara de extrapolación, en la fig. 36 se muestra el acoplamiento mecánico entre el motor de pasos y el tornillo micrométrico de la CE, el arreglo mecánico se elaboró en el taller de electrónica de la Gerencia de Ingeniería; las dimensiones de las piezas se muestran en el anexo B.

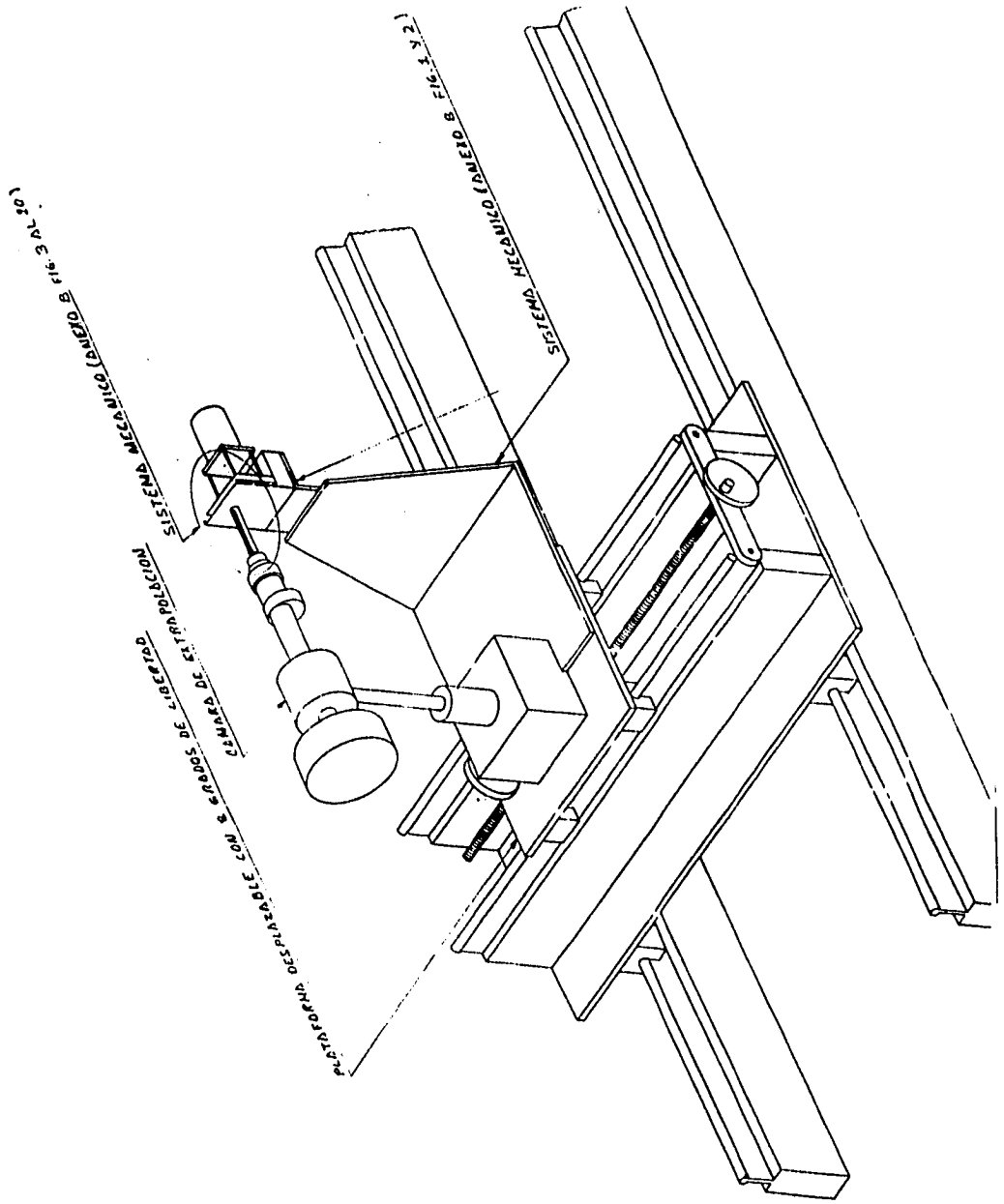


Fig. 35 Vista del conjunto montado en la plataforma de la CE

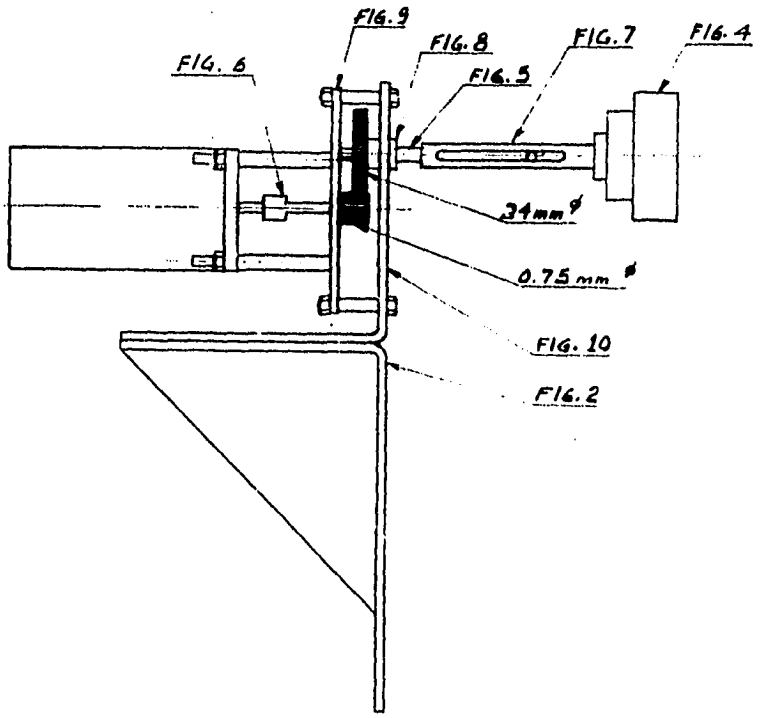


Fig 36 Acoplamiento mecánico

4 MONTAJE Y PRUEBAS

4.1 MONTAJE MECÁNICO

El arreglo mecánico lo componen: el soporte para el motor y el acoplamiento mecánico entre el tornillo micrométrico y el motor de pasos.
Para montar el soporte del motor, se retiran los ocho tornillos, que sujetan la plataforma superior, se coloca el soporte sobre la plataforma y se fija con tornillos utilizando los mismos barrenos de donde se retiraron los tornillos, la fig. (37) muestra el montaje de la plataforma.

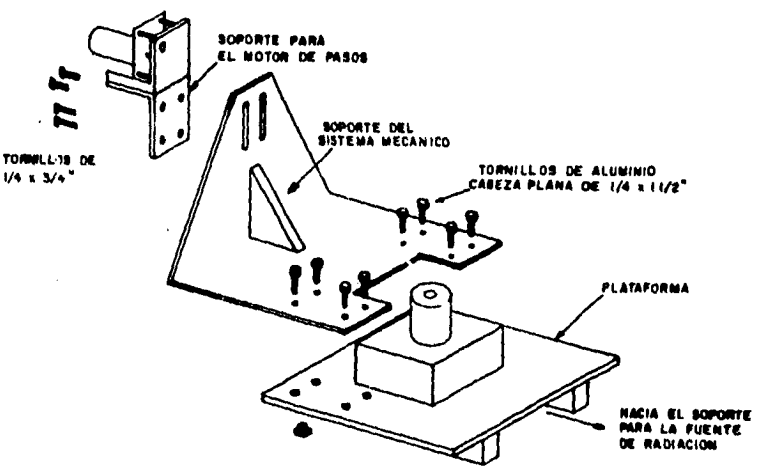


Fig.37 Montaje del soporte en la plataforma .

Una vez fijado el soporte, se procede a fijar el acoplamiento mecánico al tornillo micrométrico de la CE, para esto primero se desplazan las placas de la cámara hasta que se encuentren separadas 1 mm, posteriormente se ajusta la altura de la CE a la altura adecuada, por último se alinea el eje del acoplamiento mecánico con el eje del tornillo micrométrico, para esto; el soporte cuenta con ranuras que permiten mover el motor tanto horizontal como verticalmente, una vez ajustada, se aprietan los tornillos respectivos para fijar el motor, y los opresores para fijar la campana al tornillo micrométrico.

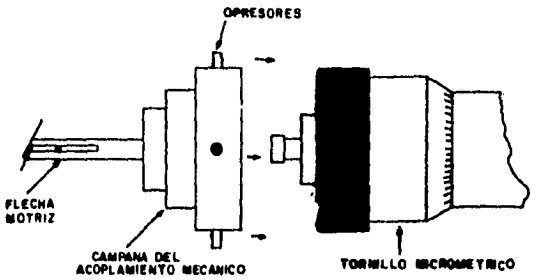


Fig. 38 Fijando el tornillo micrométrico al acoplamiento mecánico.

4.2 CONEXIÓN DEL SISTEMA

Para hacer la conexión de los instrumentos del sistema, lo primero es asegurar que todos los instrumentos y la interfaz para la CE se encuentren apagados, el sistema en su parte electrónica se compone de:

- Tarjeta BETA
- Interfaz para la CE
- Tarjeta de Interfaz Power IEEE488.
- Convertidor Analógico Digital 16
- Módulo digital 80-I488.
- Electrómetro Keithley 617.
- Fuente de HP 6110A.

1.-La tarjeta beta, se inserta en alguna de las ranuras de expansión de la PC, esta tarjeta se conecta a la interfaz por medio de dos cables, el primero tiene un conectores DB9 en los extremos, este se conecta a la interfaz de la cámara de extrapolación en el conector DB9 macho etiquetado con el nombre de **control del motor de pasos**, el otro cable tiene en sus extremos conectores hembra DB25 etiquetados cada uno como **conthv** y **PC** respectivamente, el conector marcado con **conthv** se conecta a la interfaz de cámara de extrapolación y al conector macho DB25 marcado con la leyenda **control de la fuente de HV**.

2.- La fuente de voltaje se fija a ceros, y se retira el conector de seguridad que se encuentra en el costado derecho de la fuente. Fig. (39).

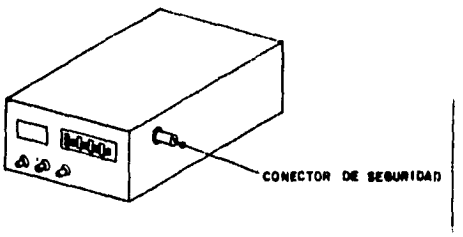


Fig.39 Ubicación del conector de seguridad de la fuente .

Se conecta el cable de la ruta de retroalimetación que tiene en un extremo un conector circular de 2 terminales y en el otro un Conector Circular de Plástico (CPC) de 9 terminales, el conector de 9 terminales se conecta a la interfaz de la cámara de extrapolación a un conector circular de 9 terminales macho, etiquetado con la leyenda **motor de pasos.**

3.-El mismo conector CPC de 9 terminales, sirve también para proporcionar energía al motor de pasos a través de un cable que en su extremo tiene un conector DIN macho de nueve terminales .

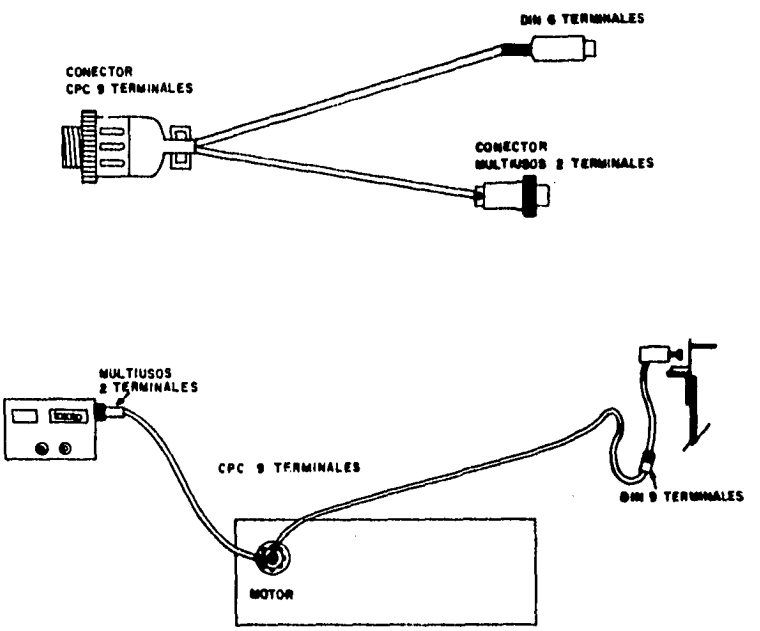


Fig.40 Conexiones del conector CPC-9 .

4.- La salida de la fuente de alto voltaje se conecta a la interfaz de la CE como se muestra el la figura.

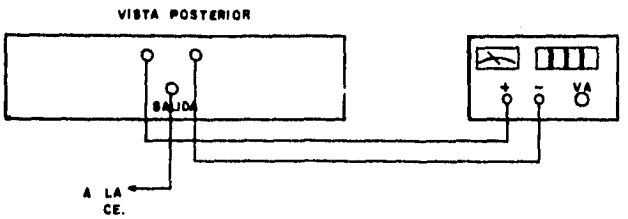


Fig. 41 Conexión de la fuente de alto voltaje.

5.- Los dispositivos con puerto IEEE488 (electrómetro, ADC, Módulo digital) se conectan a la tarjeta Power IEEE488 que se encuentra en la PC (si esta tarjeta no está instalada consultar el manual del usuario de la tarjeta Power IEEE488 para su instalación).

6.- Del módulo de medición de condiciones ambientales se toman las señales de presión, temperatura y humedad relativa y se conectan al ADC como muestra la Fig.42 .

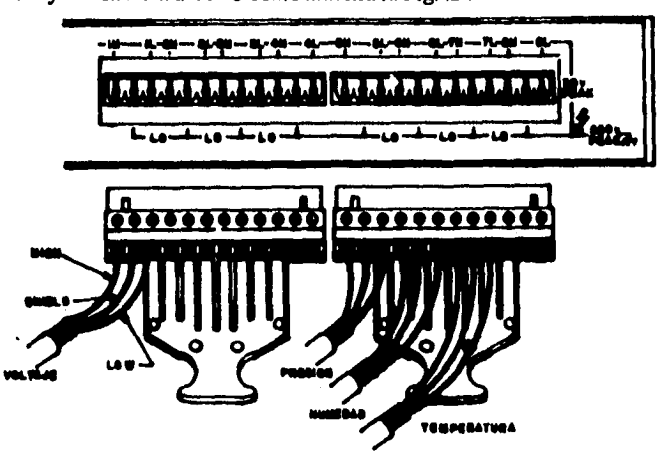


Fig.42 Conexión de las señales de factores ambientales al ADC .

7.- La entrada del electrómetro se conecta a la placa de colección de la CE como se muestra en la Fig.43 .

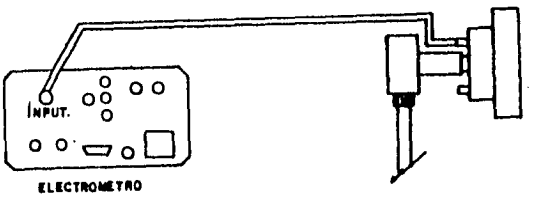


Fig.43 Conexión del electrómetro al electródo de colección de la CE .

8.- Verifique que todas las conexiones se hayan hecho adecuadamanete; la conexión de todo el sistema se muestra en la fig. 44.

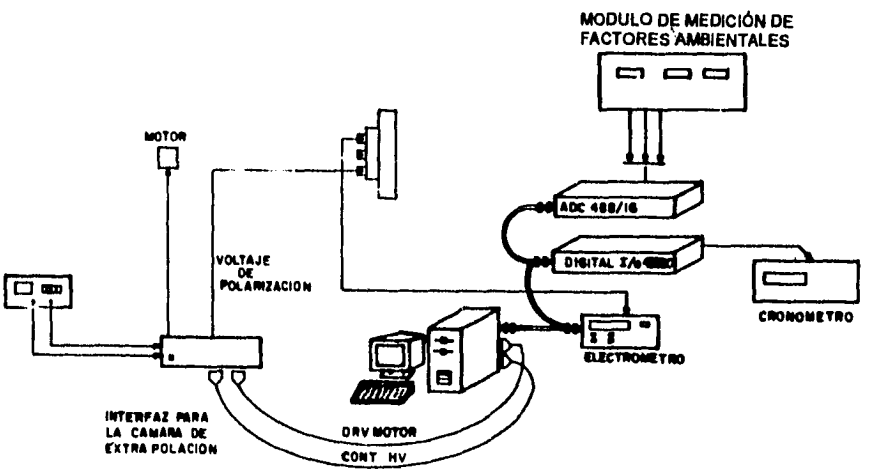


Fig. 44 Configuración del sistema de interfaz para la cámara de extrapolación.

4.3 PRUEBAS

Las pruebas realizadas, se encaminan a determinar el error en el voltaje y el desplazamiento de las placas de la CE, bajo este razonamiento, se realiza una prueba de voltaje, y otra de desplazamiento de las placas.

4.3.1 PRUEBA DE VOLTAJE

Para la prueba de voltaje, se escribe un pequeño programa que genera una escalera de voltaje con los siguientes pasos:

0 a 1 V escalones de 0.1 V
1 a 10 V escalones de 1 V
10 a 100 V escalones de 10 V
100 a 300 V escalones de 100V

En cada escalón se mide el voltaje y se registra; con la información obtenida, se elabora una gráfica donde el voltaje propuesto esta en el eje de las abcisas y el voltaje medido en el eje de las ordenadas, los datos se ajustan a una recta empleando el método de mínimos cuadrados para obtener el modelo matemático.
En otra prueba de voltaje, se utiliza el programa BETAI488 en la opción de curvas de extrapolación, en esta función se fija la distancia inicial y final a 20000 micras, y el gradiente de voltaje a 50 V/cm, se inicia la ejecución del programa; durante el ajuste del voltaje de la fuente, se va interrumpiendo aleatoriamente la ejecución para medir el voltaje aplicado, la información se registra, y con esta información se construye una curva, los resultados de las pruebas de voltaje se muestran en la tabla VI y la tabla VII.

voltaje propuesto	voltaje medido	voltaje propuesto	voltaje medido
0.1	0.1028	7	7.127
0.2	0.2050	8	8.146
0.3	0.3064	9	9.163
0.4	0.4088	10	10.039
0.5	0.5108	20	20.13
0.6	0.6130	30	30.23
0.7	0.7143	40	40.37
0.8	0.8157	50	50.50
0.9	0.9175	60	60.61
1	1.0184	70	70.71
2	2.038	80	80.81
3	3.046	90	90.95
4	4.075	100	101.99
5	5.092	200	203.3
6	6.109	300	305.1

Tabla VI voltaje medido en pasos uniformes.

voltaje propuesto	voltaje medido	voltaje propuesto	voltaje medido
0.0	0.0	57.9	58.2
0.2	0.2	60.3	60.5
0.4	0.41	66.2	66.5
0.5	0.5	70.1	70.4
0.6	0.6	77.7	78.00
0.7	0.7	79.4	9.8
1	1	80.2	80.5
1.5	1.5	82.9	83.3
2.1	2.13	87.1	87.5
3	3.04	90.3	90.7
4	4.05	94.6	95.1
5	5.06	97.4	97.9
6.1	6.17	100.5	101.9
7.2	7.29	101.8	103.2
7.5	7.59	110	111.7
8.3	8.4	120.7	122.1
9.9	10	130.4	131.9
10	10.09	140	141.6
10.8	10.8	150	151.7
15.1	15.15	160	161.7
20	20.03	170	170.7
22.8	22.87	180.1	181.9
25.2	25.4	190	191.9
30	30.07	200	202.2
35.7	35.83	257.2	259.7
41.1	41.2	275.9	278.5
45	45.2	300	303.3
50	50.2	350	353.5
55.3	55.5	372.7	376.3
		399.8	404

Tabla VII voltaje medido aleatoriamente de 0 a 399.8 .

voltaje propuesto	voltaje medido	voltaje propuesto	voltaje medido
0.3	0.3	11	11.01
0.5	0.511	12	12.02
0.7	0.715	13.1	13.14
0.9	0.918	14	14.05
1.4	1.427	14.6	14.66
2.0	2.037	15.2	15.27
3.0	3.054	16.5	16.59
3.6	3.666	17	17.09
4.1	4.16	18	18.11
5.1	5.7	19	19.12

5.7	5.78	19.7	19.83
6.1	6.18	21	21.06
7.4	7.5	22.0	22.08
8.0	8.1	23	23.09
8.9	9.2	24.0	24.1
9.1	9.22	25.0	25.1

Tabla VIII voltaje medido de 0 a 25 V .

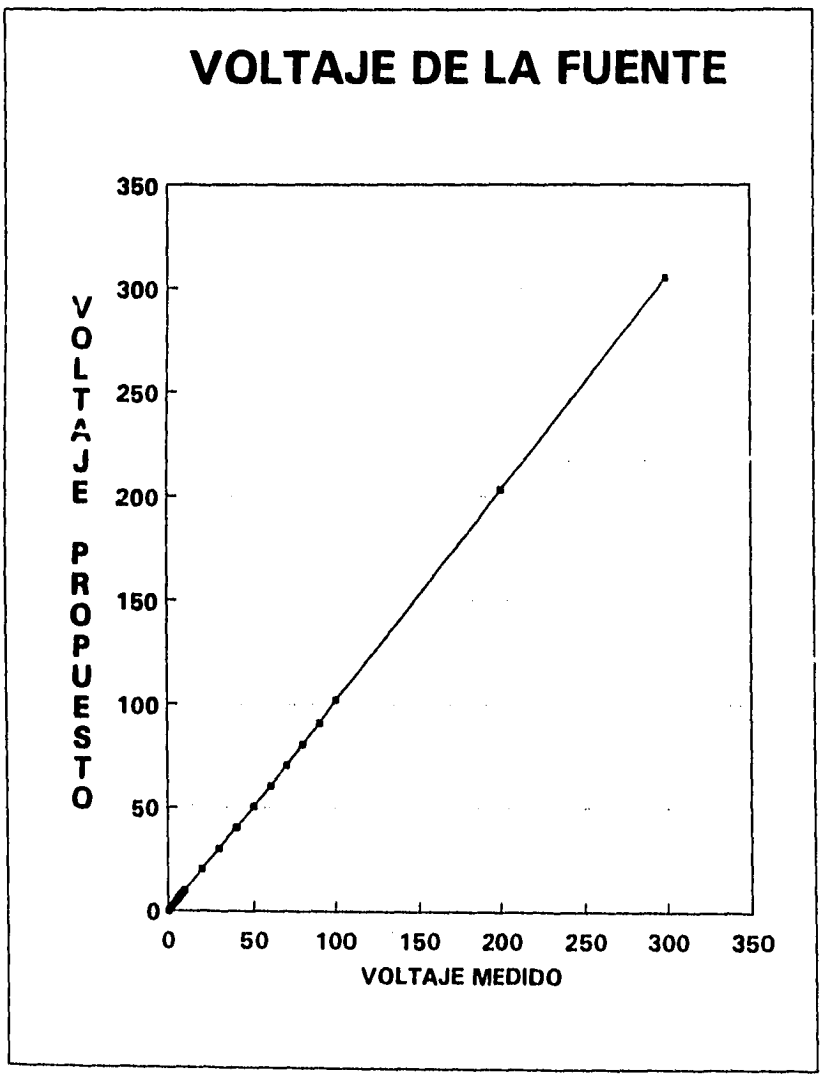


Fig. 45 Gráfica de voltaje medido contra voltaje propuesto de la tabla VI .

VOLTAJE DE LA FUENTE DE 0 A 399.8 V

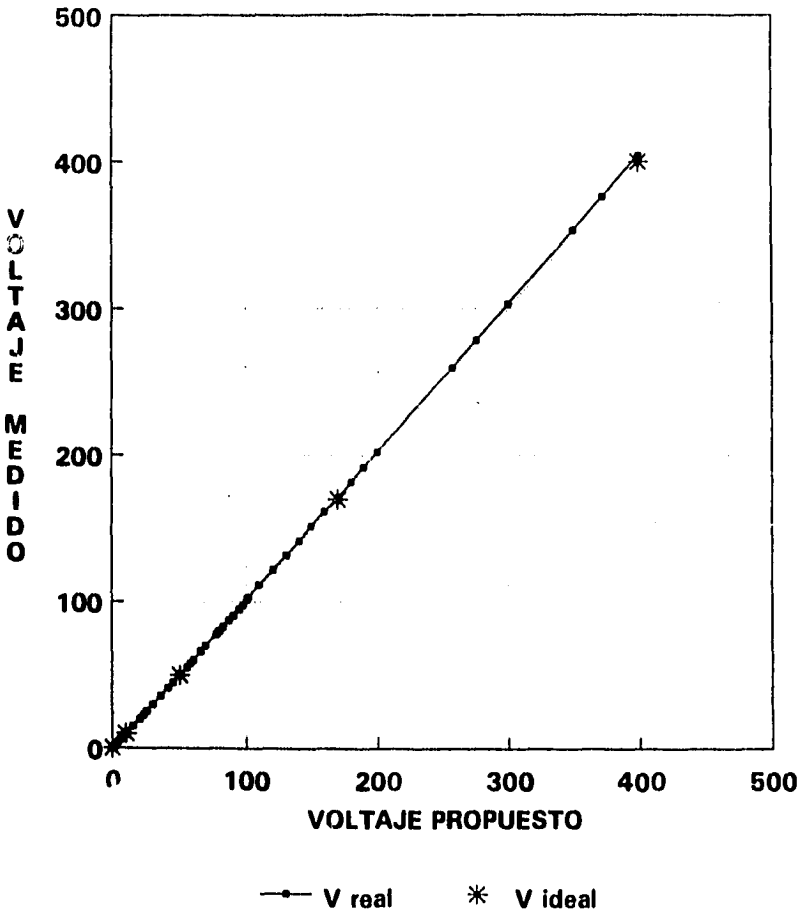


Fig. 46 Gráfica de la tabla VII .

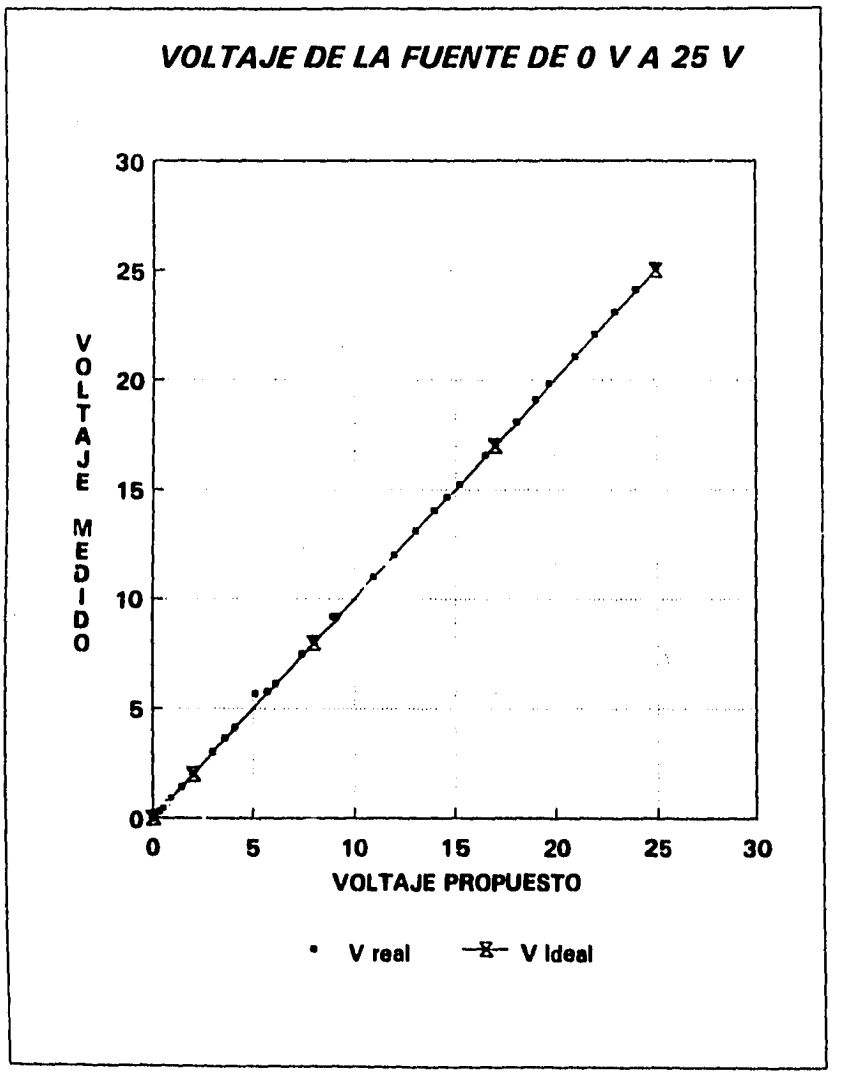


Fig. 47 Gráfica de la tabla VIII .

4.3.2 PRUEBA DE DESPLAZAMIENTO

4.3.2.1 PRUEBA POR COMPARACIÓN DIRECTA.

Con esta prueba se determina la relación de desplazamiento de las placas, por cada paso del motor, el desplazamiento se lleva de 0 a 19000 pasos en intervalos de 1000 pasos, en cada paso, se mide la distancia desplazada con la escala del micrómetro, con esta información se construye una tabla en la que se tabula el número de pasos contra el desplazamiento medido; con esta información se construye una gráfica, en la que la pendiente de la recta es la relación existente entre el número de pasos y la distancia desplazada, la información obtenida se muestra a continuación.

Num. de pasos	desplazamiento en micras
500	300
1000	595
2000	1190
3000	1785
4000	2385
5000	2980
6000	3580
7000	4175
8000	4770
9000	5370
10000	5970
11000	6565
12000	7160
13000	7755
14000	8355
15000	8950
16000	9545
17000	10145
18000	10740
19000	11340
20000	11935

Tabla IX Número de pasos contra distancia desplazada .

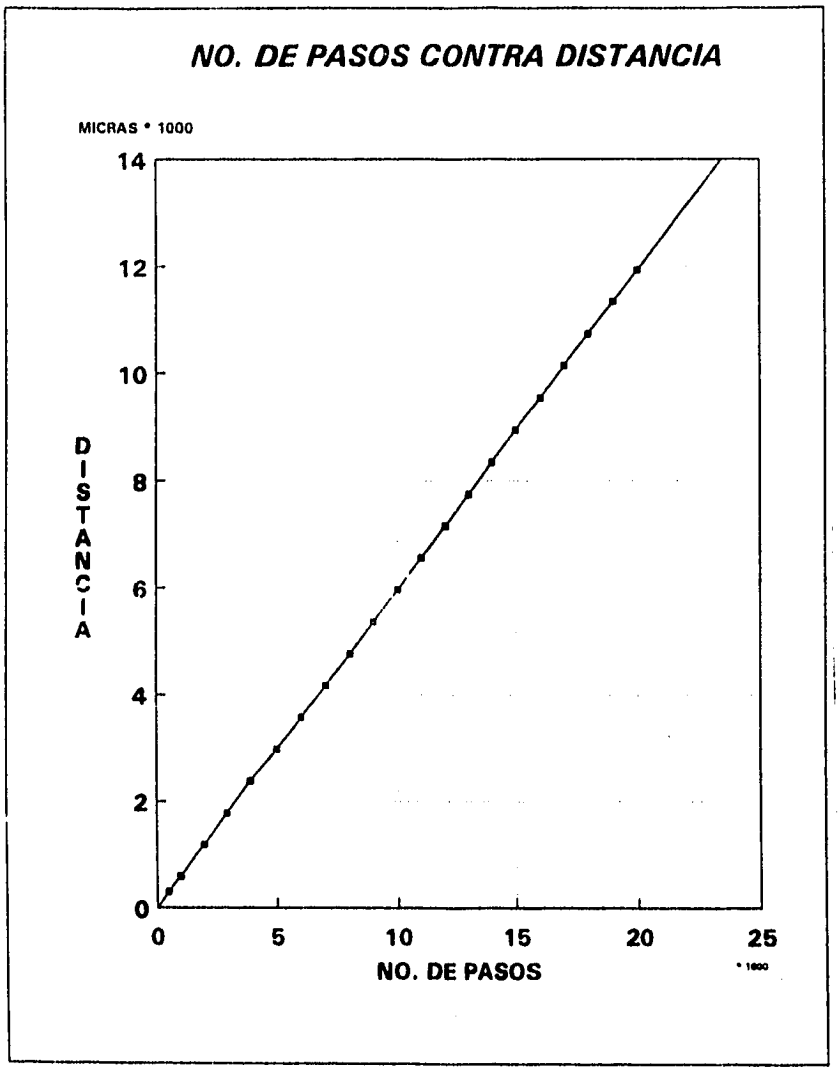


Fig.48 Gráfica de la tabla IX.

La pendiente obtenida es de m = 0.6 [micras/paso] , de esta relación obtenemos que:

1 paso = 1.6667 micras.

Con este valor se corrige, la rutina de ajuste de distancia pues se asumió que un paso del motor correspondía a 1 micra de desplazamiento.
Con este valor corregido, se realizó la misma prueba, solo que ahora en vez de dar un número de pasos determinados se propusieron distancias determinadas, se construyeron dos gráficas, en la primera, las

placas se desplazaron de 0 a 1000 micras en pasos de 100 micras , y en la segunda, las placas se desplazaron de 0 a 10000 micras en pasos de 1000 micras, los resultados se muestran a continuación:

distancia propuesta	distancia medida
100	100
200	200
300	300
400	398
500	500
600	698
700	698
800	800
900	902
1000	10000

Tabla X Desplazamiento de 0 a 1000 micras .

distancia propuesta	distancia medida
1000	990
2000	1985
3000	2982.5
4000	3975
5000	4970
6000	5967.5
7000	6962.5
8000	7967.5
9000	8952.5
10000	9947.5

Tabla XI Desplazamiento entre placas de 0 a 10000 micras..

Como se puede observar, cuando el desplazamineto es en pasos de 1000 micras, existe un error máximo de 52.5 micras, esto se compensa en la rutina que genera los pulsos para mover el motor.

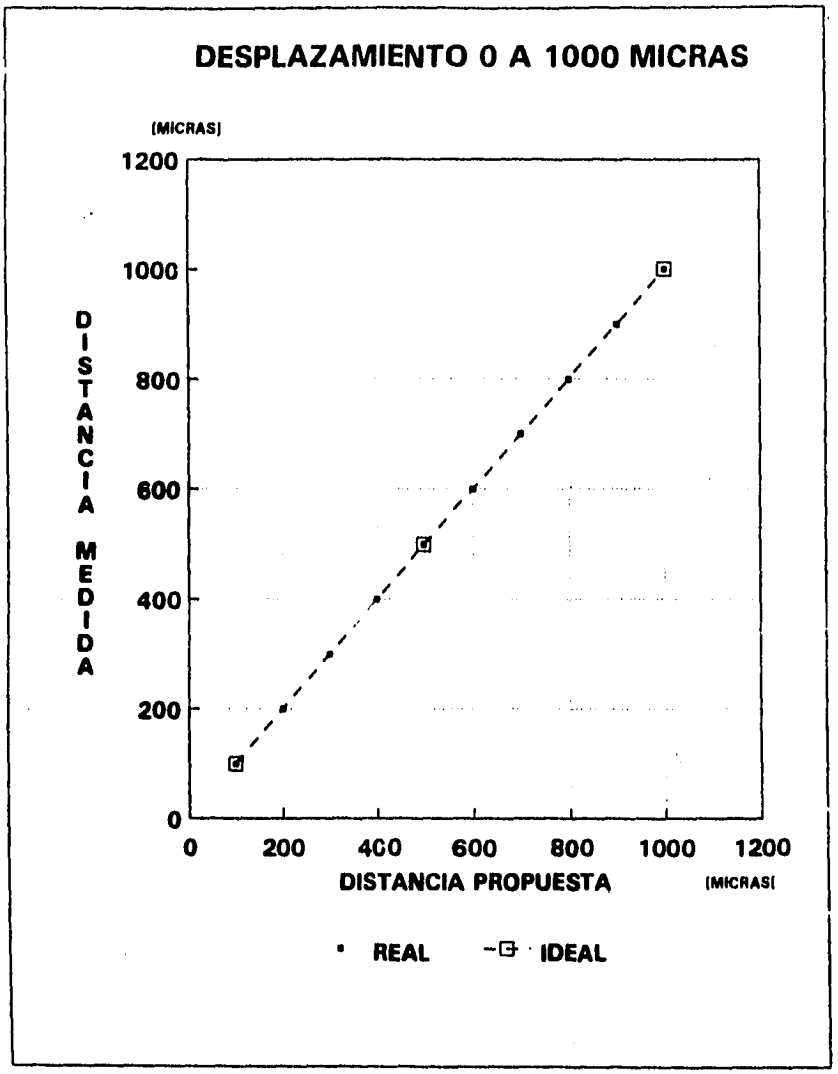


Fig. 49 Gráfica de la tabla X .

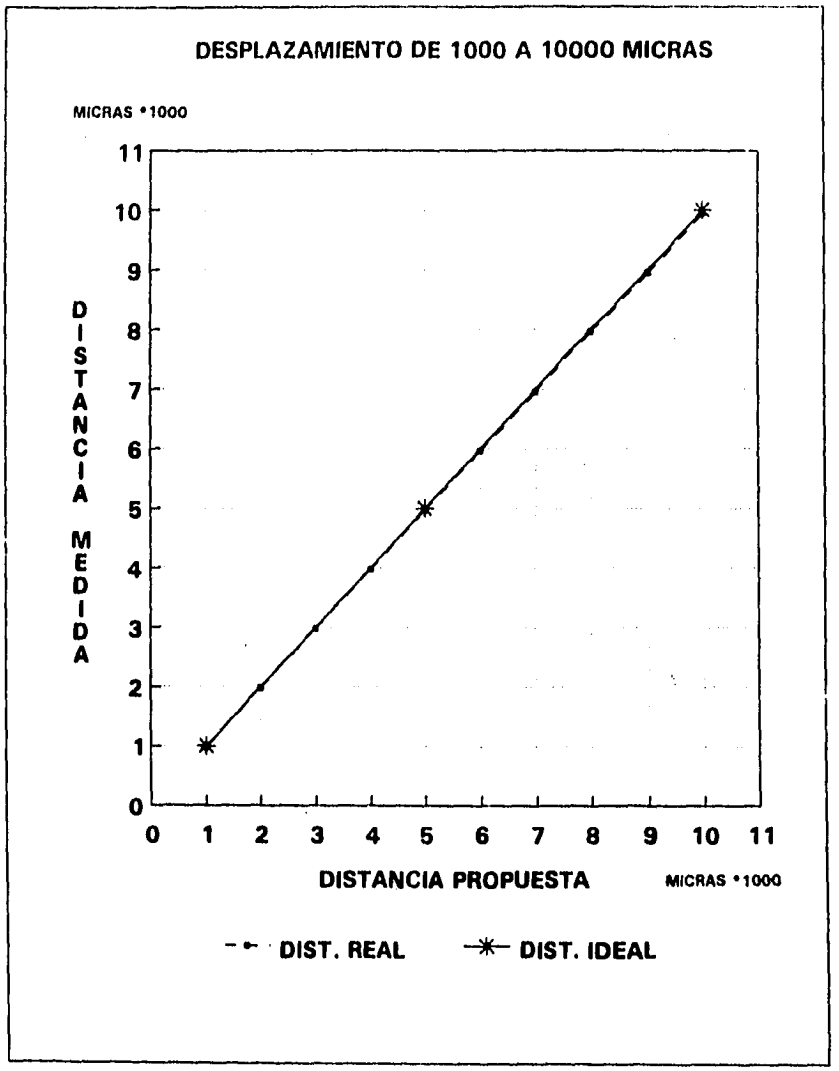


Fig. 50 Gráfica de la tabla XI .

4.3.2.2 MEDICIÓN DE DESPLAZAMIENTO POR MÉTODO CAPACITIVO

En esta prueba se emplea la rutina "AUTOTEST" descrita anteriormente, con el fin de determinar la distancia real entre placas y en la que se toman 5 puntos 10000 , 5000, 2500, 1000 y 500 micras en la siguiente tabla se muestran los resultados obtenidos.

INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES
CENTRO DE METROLOGIA DE RADIACIONES IONIZANTES
CALIBRACION DE LA CAMARA DE EXTRAPOLACIION
recta normalizada: L= -8.5581851e-05 + 6.5680506e-15 Ce-1 [m]
area efectiva = 7.4215262e-04 [m^2]
L real cuando dial marca 0 = -8.5581851e-05 [m]

distancia propuesta [micrometros]	distancia medida [m]	capacitancia [F]
10000	1.00419708e-02	6.54059923e-13
5000	5.16502326e-03	1.27164013e-12
2500	2.61391024e-03	2.51273004e-12
1000	1.06336444e-03	3.17666942e-12
500	5.43640810e-04	1.20815996e-11

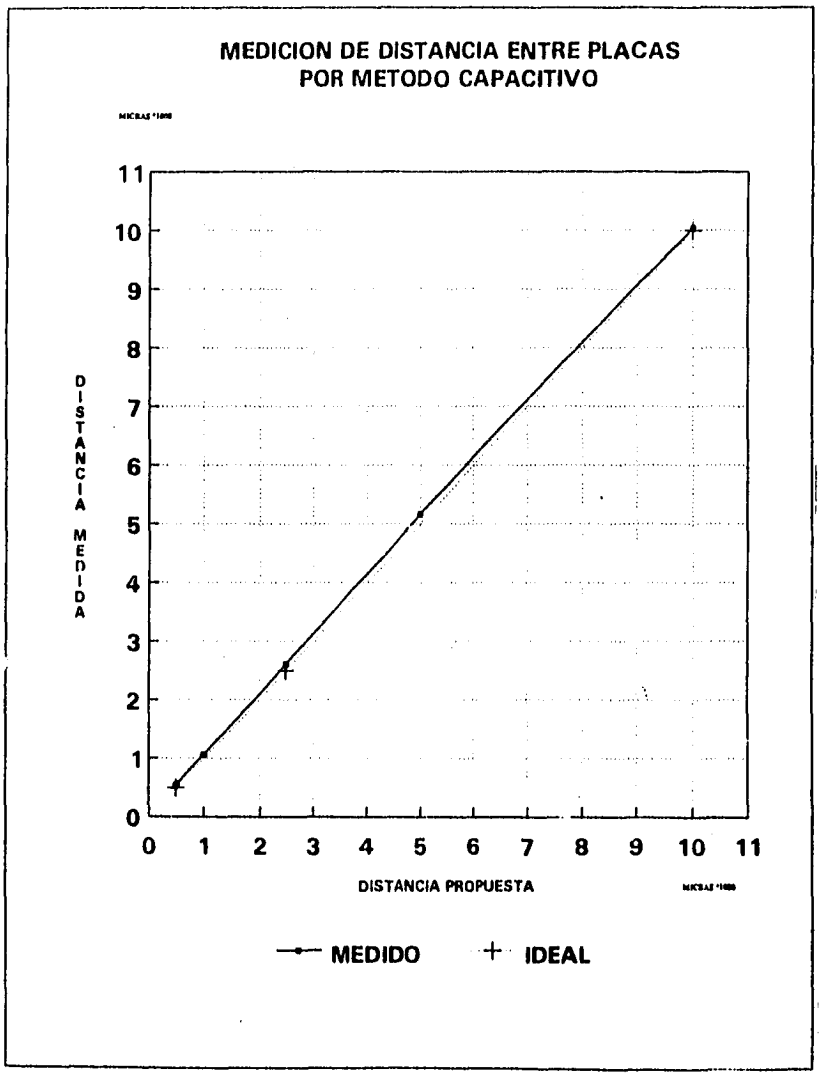


Fig. 51 gráfica de la prueba de medición de distancia por método capacitivo.

5 RESULTADOS Y CONCLUSIONES

5.1 RESULTADOS

El sistema diseñado es un sistema de adquisición de datos para la construcción de curvas de extrapolación utilizadas en la determinación de la dosis absorbida absoluta debido a fuentes de radiación beta externas, el conjunto de rutinas para llevar a cabo el proceso de medición se encuentra en el programa **BETAI488**. Para la ejecución del proceso de medición, el sistema es capaz de ajustar la distancia entre placas de la cámara de extrapolación en un intervalo de 20 000 a 100 micras. En la prueba de desplazamiento con intervalos de 100 en 100 micras, en un desplazamineto de 100 a 1 000 micras el error es mínimo, no siendo este el caso en la prueba realizada en un recorrido de 10 000 a 1 000 micras con intervalos de 1 000 micras, en este caso, se tiene un error máximo de 1% considerando el desplazamiento total, un error máximo de 5.25% en un intervalo de 1 000 micras; durante la ejecución de la rutina de adqusición, se observó un corrimeinto de aproximadamente 15 micras al final del recorrido, en un recorrido de 10 000 a 100 micras; en la prueba de medición de distancia por método capacitivo se observa, como se puede apreciar, que en los extremos de la curva (puntos cercanos a 500 micras y a 10 000 micras, ver fig. 51 el error es mínimo, teniendo dentro del intervalo de 500 a 10000 micras un error máximo del 9% . Este error es atribuible al sistema mecánico, debido a la fricción en el acoplamiento mecánico y a que los engranes no son de la medida que se especificó; y a que la cámara esté descalibrada.

El ajuste de la fuente de polarización, se puede hacer en un intervalo de -399.9V a +399.9 V con una resolución de 0.1 V con un error de un 2% lo que comprueba que el error en la fuente está en función del error en la referencia que en este caso son las resistencias en el banco de resistencias de la referencia programable estas resistencias tienen una tolerancia de un 2%.

El sistema, a través del ADC y del electrómetro adquiere información de la temperatura, presión y humedad ambientales, y de la corriente o carga almacenada entre las placas de la CE, el sistema presenta como resultado un listado con los valores de las variables de interés, un ejemplo de este archivo se muestra a continuación:

INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES
CENTRO DE METROLOGIA DE RADIACIONES IONIZANTES
USUARIO: fernando
TIPO DE FUENTE: sr-90
MEDICION DEL RUIDO DE FONDO
distancia entre placas: 10000 voltaje aplicado : 5.000 V

n	lectura [A]
0	2.9270e-14
1	3.3300e-14
2	3.9040e-14
3	3.7170e-14
4	4.0620e-14

promedio: 3.5880e-14 A desv. std.: 4.1101e-15 A

MEDICIONES CON IRRADIACION
distancia entre placas: 10000 micras
presion: 710.000 mbar humedad: 45.000 % temperatura 20.000 C

presion: 710.000 mbar humedad: 45.000 % temperatura 20.000 C

n	VOLTAJE APLICADO 5.000V [A]	-5.000V [A]
0	-3.5442e-13	-9.7333e-12
1	-4.9360e-13	-1.1452e-11
2	-5.6548e-13	-1.1497e-11
3	-6.0158e-13	-1.1473e-11
4	-6.1769e-13	-1.1488e-11
promedio:	-5.2655e-13	-1.1129e-11
desv.srtd :	9.6090e-14	6.9786e-13

MEDICIONES CON IRRADIACION
distancia entre placas: 9500 micras
presion: 710.000 mbar humedad: 45.000 % temperatura 20.000 C

n	VOLTAJE APLICADO 5.000V [A]	-5.000V [A]
0	-1.4782e-13	-1.0456e-11
1	-6.0488e-13	-1.0979e-11
2	-6.1252e-13	-1.0934e-11
3	-6.1353e-13	-1.0966e-11
4	-6.1653e-13	-1.0986e-11
promedio:	-5.1906e-13	-1.0864e-11
desv.srtd :	1.8566e-13	2.0478e-13

MEDICIONES CON IRRADIACION
distancia entre placas: 9000 micras
presion: 710.000 mbar humedad: 45.000 % temperatura 20.000 C

n	VOLTAJE APLICADO 5.000V [A]	-5.000V [A]
0	-3.0325e-13	-9.5133e-12
1	-6.2026e-13	-1.0406e-11
2	-6.2711e-13	-1.0392e-11
3	-6.3020e-13	-1.0413e-11
4	-6.2967e-13	-1.0414e-11
promedio:	-5.6210e-13	-1.0228e-11
desv.srtd :	1.2947e-13	3.5729e-13

Este archivo con pequeñas modificaciones puede ser utilizado con algún paquete de manejo estadístico de información, por ejemplo SAS; con esta información se puede determinar la dosis impartida por una fuente de radiación beta.

extrapolación, la recta normalizada y la distancia real entre placas cuando el dial del tornillo micrométrico marca 0, el resultado obtenido en la prueba de medición por método capacitivo no concuerda en algunas mediciones con lo que se mide en el dial del tornillo micrométrico, al hacer la medición directa en el tornillo micrométrico se mide una diferencia máxima de 15 micras en comparación a la diferencia que marca el método capacitivo que indica una diferencia máxima de 160 micras, esto se puede deber a efectos físicos no contemplados en el método de medición o de alguna irregularidad mecánica en la CE.
La interfaz con el usuario se hace a través de menus que llevan al usuario de una forma sencilla a través de las opciones del sistema.

5.2 CONCLUSIONES

La aplicación de sistemas de automatización tanto en industrias como en laboratorios no es ajena al ININ, pues este proyecto surgío como una necesidad planteada por el Centro de Metrología de Radiaciones Ionizantes; del desarrollo de este proyecto se pueden obtener la siguientes conclusiones:

Parte importante de la formación de recursos humanos, y sobre todo en áreas como la ingeniería, es poner en contacto al estudiante con problemas reales que permitan la aplicación de sus conocimientos y le estimulen a buscar las soluciones adecuadas al problema que se le presenta, utilizanado los conocimientos con los que cuenta como una base sobre la que apoyarse, para que en base a la investigación se pueda dar una solución adecuada al problema.

Para la realización del proyecto se emplearon los recursos con los que se cuenta, adaptando al sistema elementos que no eran propiamente programables (como la fuente de voltaje HP 6010A) por medio de una interfaz con la computadora; integrando un sistema con elementos que se pueden obtener comercialmente, y, diseñando y construyendo circuitos, como la referencia programable para la fuente de polarización , y el circuito para manejar el motor de pasos, que permitieron integrar estos dispositivos al sistema.

La utilización de PLD's en el diseño de circutos digitales reduce el tamaño de los circuitos, y mejora sobre manera su confiablidad y facilidad de mantenimiento, los PLD's permitieron reducir de alrededor de 10 "chips" LSI a un solo "chip", haciendo que el diseño del circuito impreso sea mucho más reducido y fácil de diseñar; los dispositivos lógicos programables ofrecen una muy buena opción para el diseño de circuitos digitales en los casos en los que el tamaño del circuito sea un factor importante, así como la confiabilidad del circuito, aunque una desventaja de los PLD's es su consumo de potencia, que comparado con dispositivos LSI de tecnología MOS en los que el consumo es muy bajo.

El sistema diseñado puede crecer si así lo decide el usario, en esta etapa del proyecto solo se tiene control automático sobre el desplazamiento entre las placas de la cámara, pero se puede hacer el posicionamineto tridimensional de la cámara en forma automática.

6 REFERENCIAS

1. Álvarez Romero José T. 1993:" Medición de rapidez de dosis absorbida en profundidad impartida por fuentes de patrones secundarios de radiación beta externos. Parte I .- Medición de rapidez de dosis absorbida en la superficie de tejido blando para isotopos de $^{90}Sr/^{90}Y$, ^{147}Pm y ^{204}Tl.", pags. (9-10),Informe Técnico,ININ, Gerencia de Seguridad Radiológica, enero 1993.

2. Bhöm J. 1986 :" The national primary standard of the PTB for realizing the unit of the absorbed dose rate to tissue for Beta radiation",pags. (11-13), PTB-DOS 13, ISSN 0172-7095, Abril 1986, RFA.

3. Bhöm J. 1986 :" The national primary standard of the PTB for realizing the unit of the absorbed dose rate to tissue for Beta radiation", (12-13), PTB-DOS 13, ISSN 0172-7095, Abril 1986, RFA.

4. Hewlett Packard : "Operating and service manual: dc power supply stb series, model 6110A", Hewlett Packard/ Harrison Division, may 1967.

5. National Semiconductor : "señal pequeña:

6. Slo Syn 1979: "dc steeping motors", Slo-Syn, USA 1979

7. Motorola :"Master selection guide"

8. Motorola 1987: "Bipolar Power Transistor Data", pags. 3-1100 a 3-1103,Motorola Inc., USA 1987.

9. Alford Roger 1989: " Programmable logic designer's guide", pags. 2 a 8, Howard W. sams and Company 1989.

10. National semiconductor 1989 :"Programmable logic devices databook and design guide",pags. (2-23 a 2-45) National semiconductor, USA 1989.

11. National semiconductor 1989:"Programmable logic devices databook and design guide", pags. (2-129 a 2-144) National semiconductor, USA 1989.

12. Intel 1986: " Microsystem Components Handbook Peripheals Vol. II", pags. 6-307 a 6-327, Intel Inc.1986

13 Brenner C. Robert 1991 :"IBM PC personal computer troubleshoothing and repair", pags. 97 a 139, Howard W. Sams and Company, USA 1991.

14. Motorola 1989:" Linear/switchmode voltage regulator handbook",pags. 85 a 93, Motorola Inc., U.S.A. 1989.

15. Motorola 1989:" Linear/switchmode voltage regulator handbook", pag. 274,Motorola Inc., U.S.A. 1989.

16. Motorola 1989:" Linear/switchmode voltage regulator handbook", pags. 73 a 76,Motorola Inc.,U.S.A. 1989.

17. Iotech 1990: "Power 488 PC/IEEE controler User's manual V.1", Iotech Inc., U.S.A 1990

18. Iotech 1990: "ADC 488/16 and ADC 488/8 analog to digital converter", Iotech Inc., U.S.A 1990

19. Keithley 1984:"Model 617 Programmable Electrometer Instruction Manual", secciones 1 y 2, Keithley , November 1984.

20. Fernández A. Jorge, Castillo F.J.C 1991:"Curso básico de lenguaje C",pags. 1 a 3,Gerencia de Recursos Humanos ININ ,Noviembre 1991.

ANEXO
A

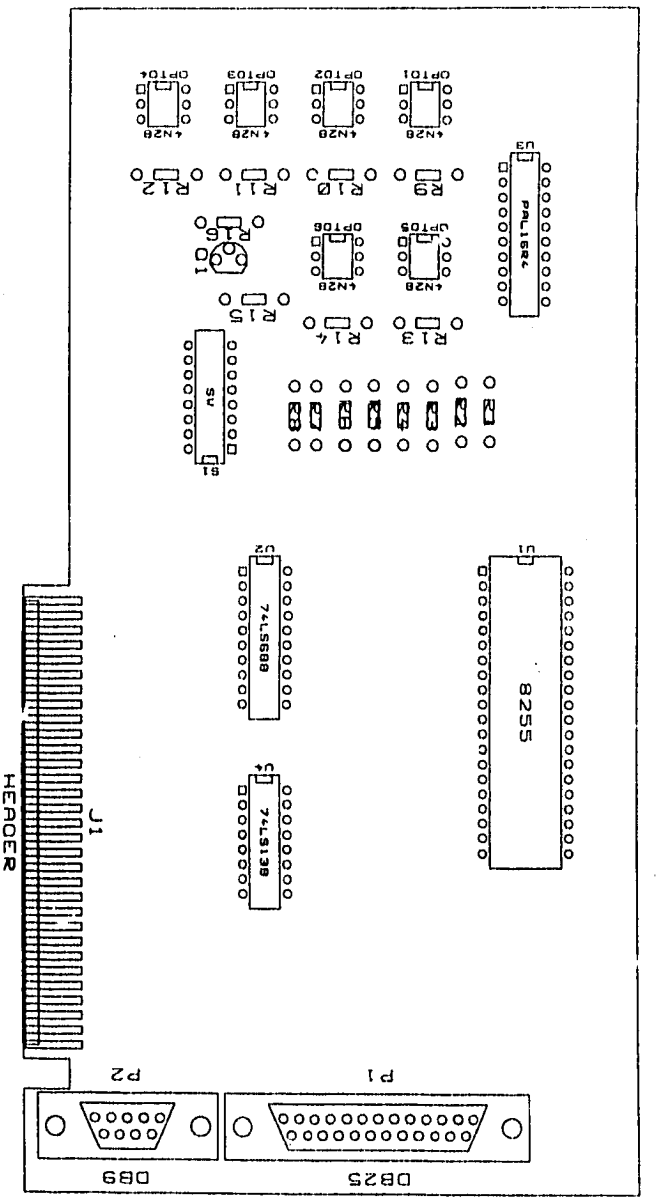
TARJETAS DEL SISTEMA

A1

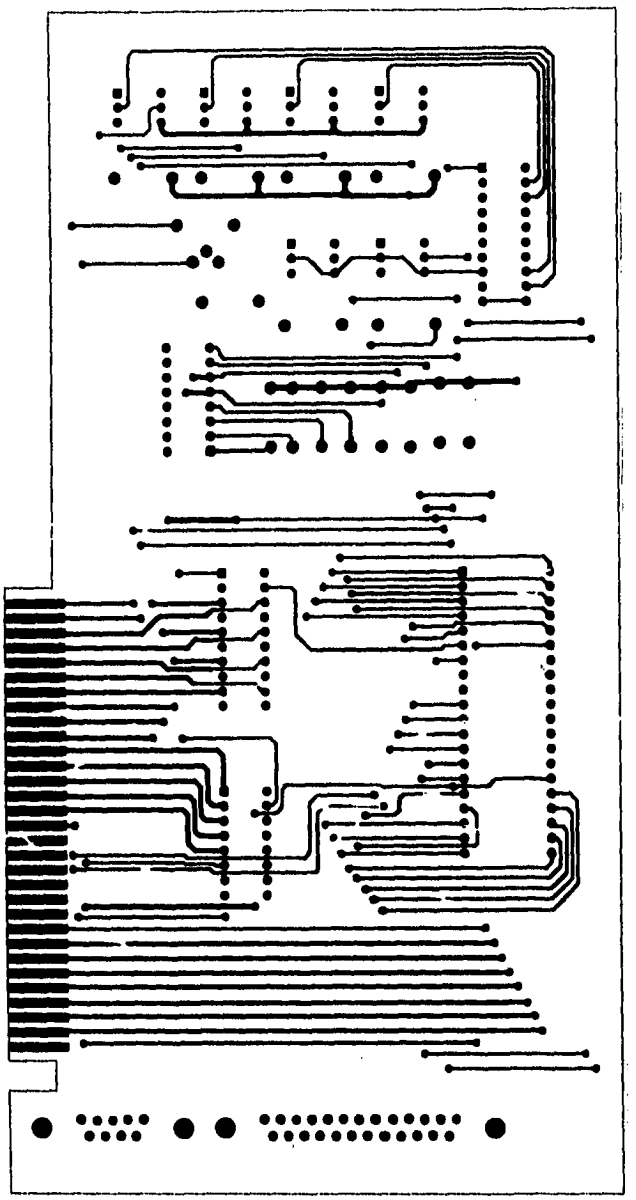
DIBUJOS DE LOS CIRCUITOS IMPRESOS

TESIS SIN PAGINACION

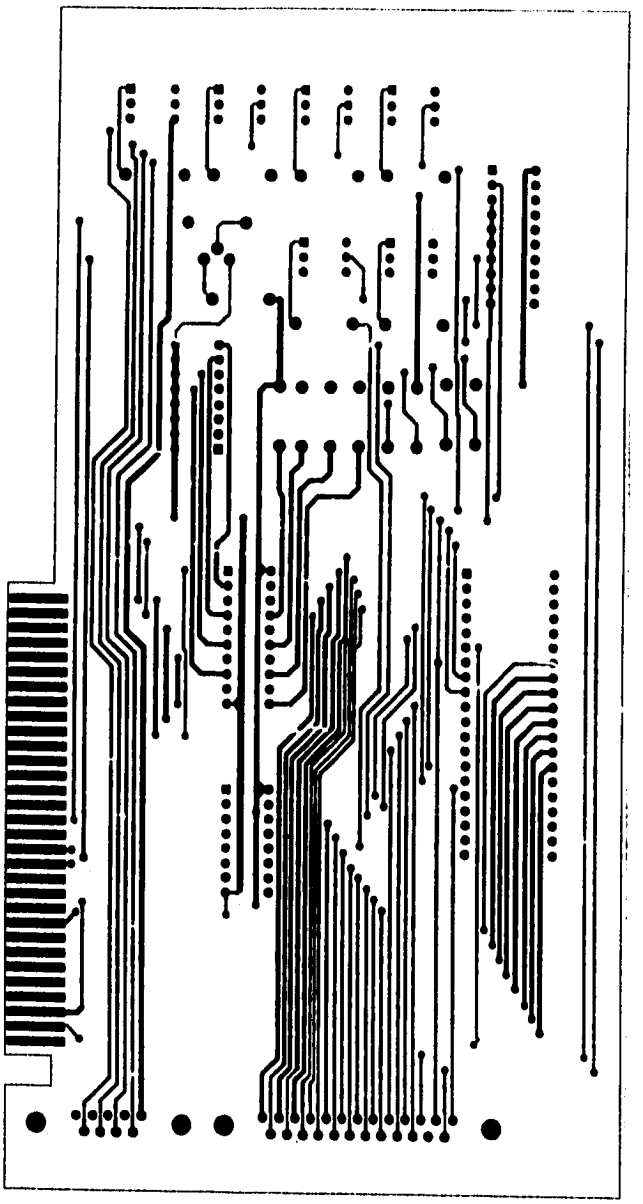
COMPLETA LA INFORMACION



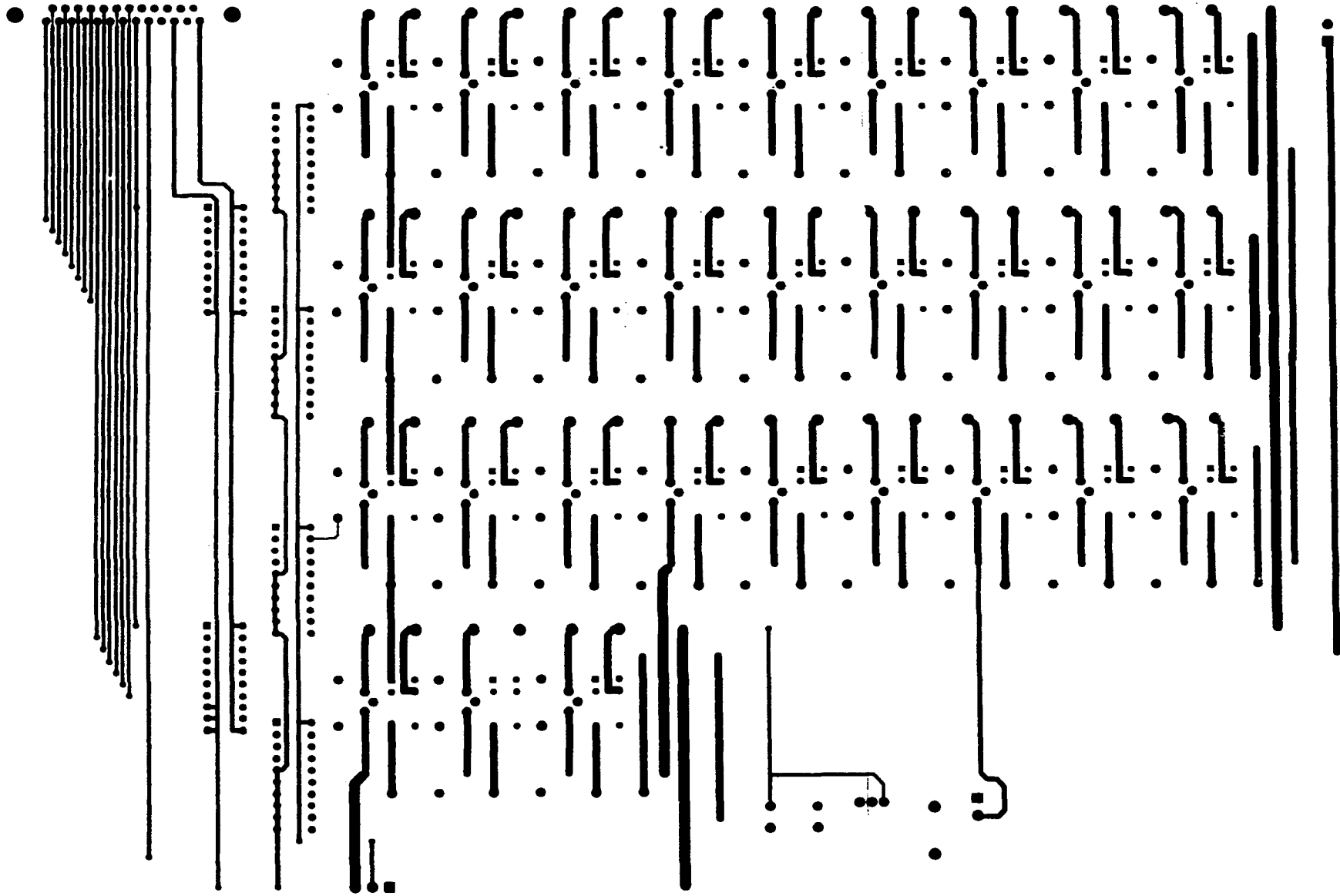
Plantilla de ubicación de componentes de la tarjeta BETA



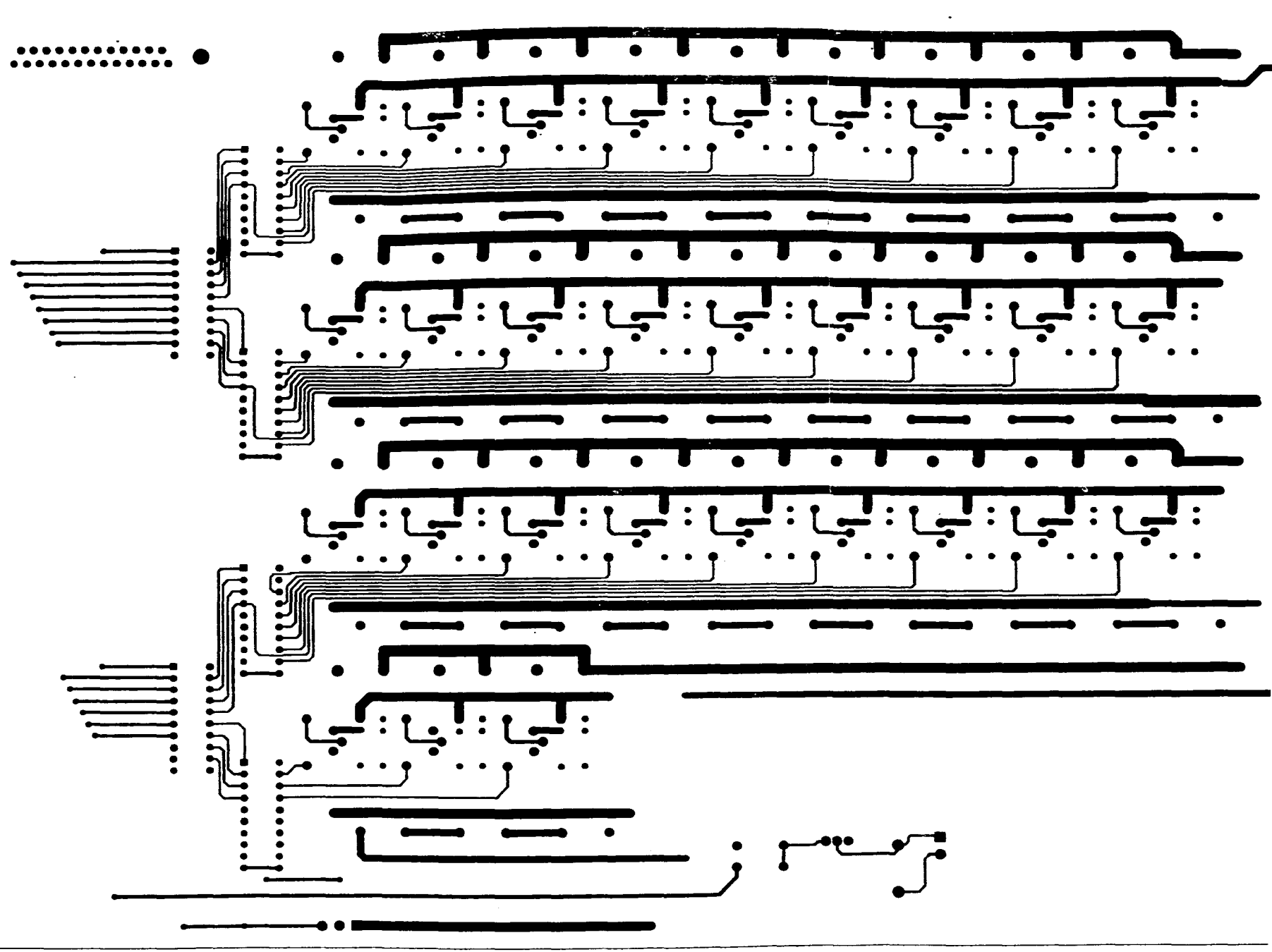
Diseño del circuito impreso, lado de componentes, tarjeta BETA

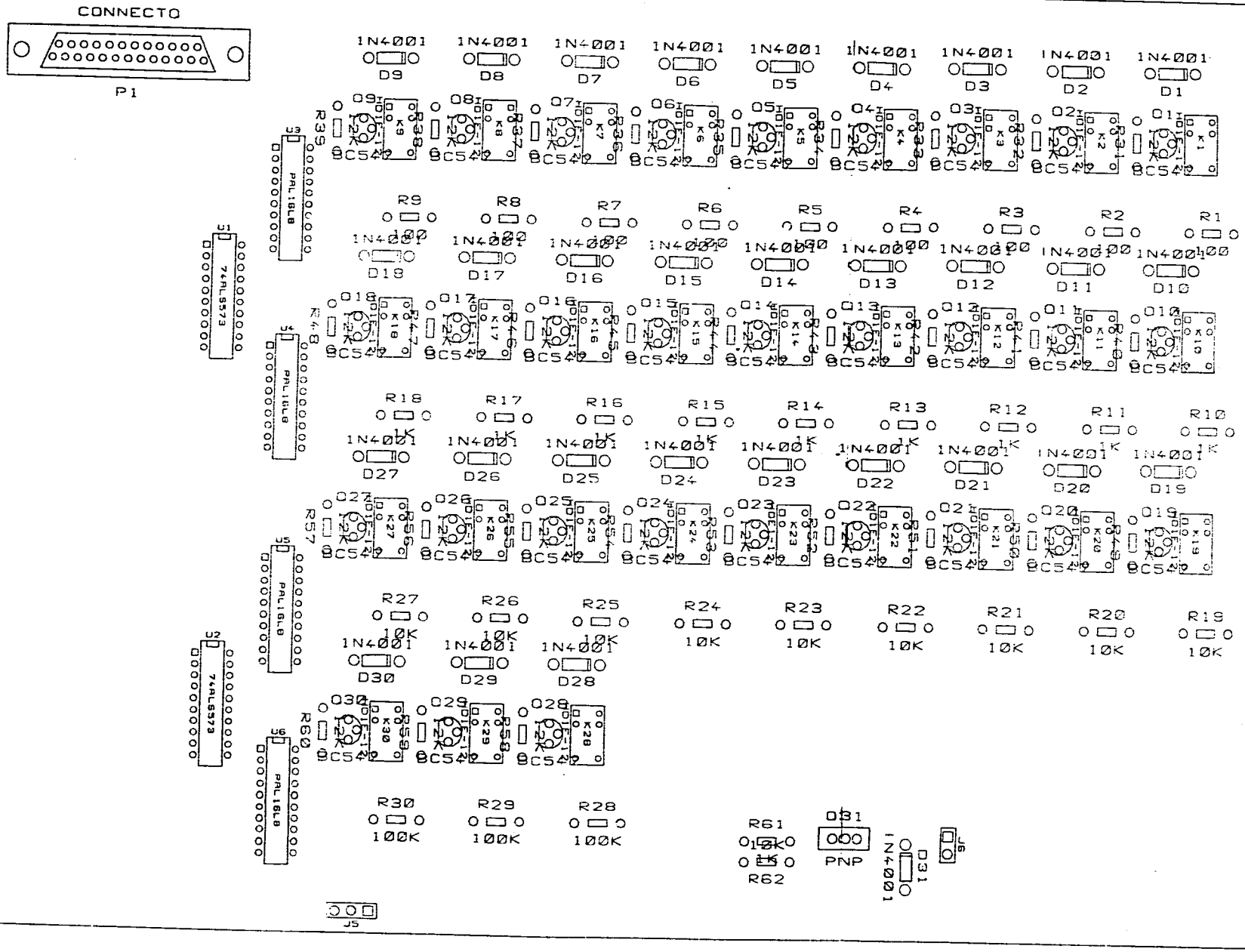


Diseño del circuito impreso,lado de soldadura, tarjeta BETA

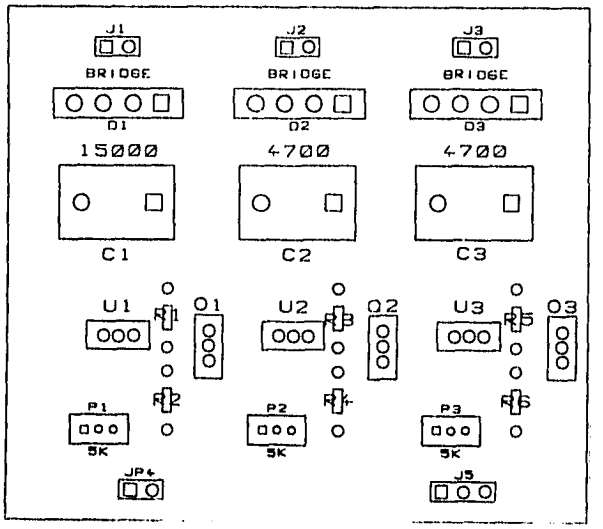


Diseño del circuito impreso, lado de componentes, tarjeta CONTHV

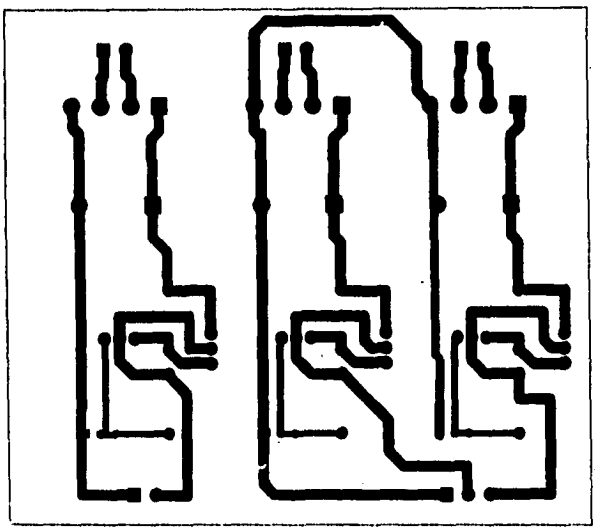




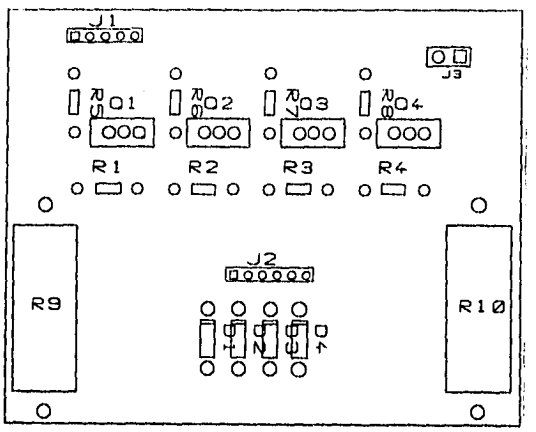
Plantilla de ubicación de componentes de la tarjeta CONTHV



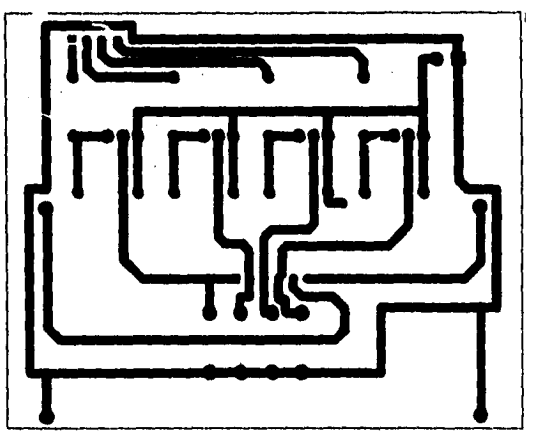
Plantilla de ubicación de componentes de la tarjeta FUENDRV



Diseño del circuito impreso, lado de soldadura, tarjeta FUENDRV



Plantilla de ubicación de componentes de la tarjeta MOTDRV



Diseño del circuito impreso.lado de soldadura, tarjeta MOTDRV

A2

LISTA DE COMPONENTES

SISTEMA SICE
PPIO Y MANEJADOR DEL MOTOR
TARJETA BETA

Elem.	Cantidad	Referencia	Parte
1	1	J1	*******
2	1	U1	8255
3	1	S1	SW DIP-8
4	1	U2	74LS688
5	1	U4	74LS138
6	6	OPTO1,OPTO2,OPTO3,OPTO4, OPTO5,OPTO6	4N28
7	1	Q1	BC547
8	8	R1,R2,R3,R4,R5,R6,R7,R8	10K
9	1	U3	PAL16R4 o GAL16V8*
10	8	R12,R9,R10,R11,R13,R14, R15,R16	1.2 K
11	1	P2	conector DB9 macho, para soldar en circuito impreso
12	1	P1	conector DB25 macho, para soldar en circuito impreso

* se puede utilizar cualquiera de los dos tipos de dispositivos, lo que cambia es el archivo JEDEC para grabar el dispositivo a utilizar, para grabar el PAL se emplea el archivo MOTDRVNE.JED y para el gal se emplea el archivo MTDRVGN.JED que se encuentran en el disco del sistema.

SISTEMA SICE
TARJETA CONTHV

Elem.	Cantidad	Referencia	Parte
1	4	U3,U4,U5,U6	PAL16L8 o GAL16V8*
2	3	J2,J3,J4	Conector BNC para montar en gabinete.
3	2	K31A,K31B	Relevador
4	27	D1,D3,D4,D5,D6,D7,D8,D9, D10,D12,D13,D14,D15,D16, D17,D18,D19,D21,D22,D23, D24,D25,D26,D27,D28,D29, D30	1N4001
5	9	R1,R2,R3,R4,R5,R6,R7,R8, R9	100 Ω ,resistencia de pel_cula metßlica de 1/2 W y 1% de tolerancia
6	30	R31,R32,R33,R34,R35,R36, R37,R38,R39,R40,R41,R42, R43,R44,R45,R46,R47,R48, R49.R50,R51,R52,R53,R54, R55,R56,R57,R58,R59,R60	12KΩ , resistencia de carb_n de 1/2 W y 10% de tolerancia.
7	30	Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8, Q9,Q10,Q11,Q12,Q13,Q14, Q15,Q16,Q17,Q18,Q19,Q20, Q21,Q22,Q23,Q24,Q25,Q26, Q27,Q28,Q29,Q30	BC547 , transistor.
8	2	U1,U2	74ALS573
9	10	R10,R11,R12,R13,R14,R15, R16,R17,R18,R61	1KΩ, resistencia de pel_cula metßlica de 1/2 W y 1% de tolerancia.
10	4	D11,D2,D20,D31	IN4001
11	10	R19,R20,R21,R22,R23,R24, R25,R26,R27,R62	10KΩ, resistencia de pel_cula metßlica de 1/2 W y 1% de tolerancia.
12	3	R28,R29,R30	100KΩ, resistencia de pel_cula metßlica de 1/2 W y 1% de tolerancia
13	1	J5	conector E.I.S. de 3 postes con paso de 0.150"

14	1	Q31	TIP 125. arreglo darlington.
15	1	J1	conector E.I.S. de 2 postes con paso de 0.150" marca AMP
16	29	K1,K2,K3,K4,K5,K6,K7,K8, K9,K10,K11,K12,K13,K14, K15,K16,K17,K18,K19,K20, K21,K22,23,K24,K25,K26,K27, K28,K29,K30	HD1E-12, relevador de 1 polo dos tiros marca AROMAT
18	1	J6	conector E.I.S. de 2 postes con paso de 0.150" marca AMP
19	1	P1	conector DB25 macho, para soldar en circuito impreso.

* se puede utilizar cualquiera de los dos tipos de dispositivos, lo que cambia es el archivo JEDEC para grabar el dispositivo a utilizar, para grabar el PAL se emplea el archivo CONTHVPA.JED y para el gal se emplea el archivo CONTHVGA.JED que se encuentran en el disco del sistema.

SISTEMA SICE
TARJETA MOTDRV

Elem.	Cantidad	Referencia	Parte
1	8	R1,R2,R3,R4	1.2 KΩ, resistencia de carb_n de 1/2 W al 10% de tolerancia
		R5,R6,R7,R8	1.8 KΩ, resistencia de carb_n de 1/2 W al 10% de tolerancia
2	1	J1	conector E.I.S de 5 postes con paso de 0.1" marca AMP.
3	1	J2	conector E.I.S de 6 postes con paso de 0.1" marca AMP
4	1	J3	conector E.I.S de 2 postes con paso de 0.150" marca AMP
5	4	D2,D1,D3,D4	1N4001
6	2	R9,R10	1 Ω, resistencia de alambre de 5W
7	4	Q4,Q1,Q2,Q3	TIP 125

SISTEMA SIDE
TARJETA FUENDRV

Elem.	Cantidad	Referencia	Parte
1	4	J2,J1,J3,JP4	Conector E.I.S. de 2 postes, con paso de 0.150" Marca AMP.
2	3	U1,U2,U3	Regulador de voltaje LM317
3	3	Q3,Q1,Q2	Transistor TIP 107
4	3	R1,R3,R5	Resistencia de carbón de 68Ω a 1/2 W
5	2	R2,R4,R6	Resistencia de carbón de 220 Ω a 1/2 W
6	4	P1,P2,P3	Trimpot de 5K Ω
7	3	D2,D1,D3	Puente de diodos
8	1	J5	Conector E.I.S. de 3 postes con paso de 0.150" marca AMP.
9	1	C1	capacitor electrolítico de 15000 µF
10	2	C2,C3	capacitor electrolítico de 4700 µF

A3

DIAGRAMAS ELÉCTRICOS

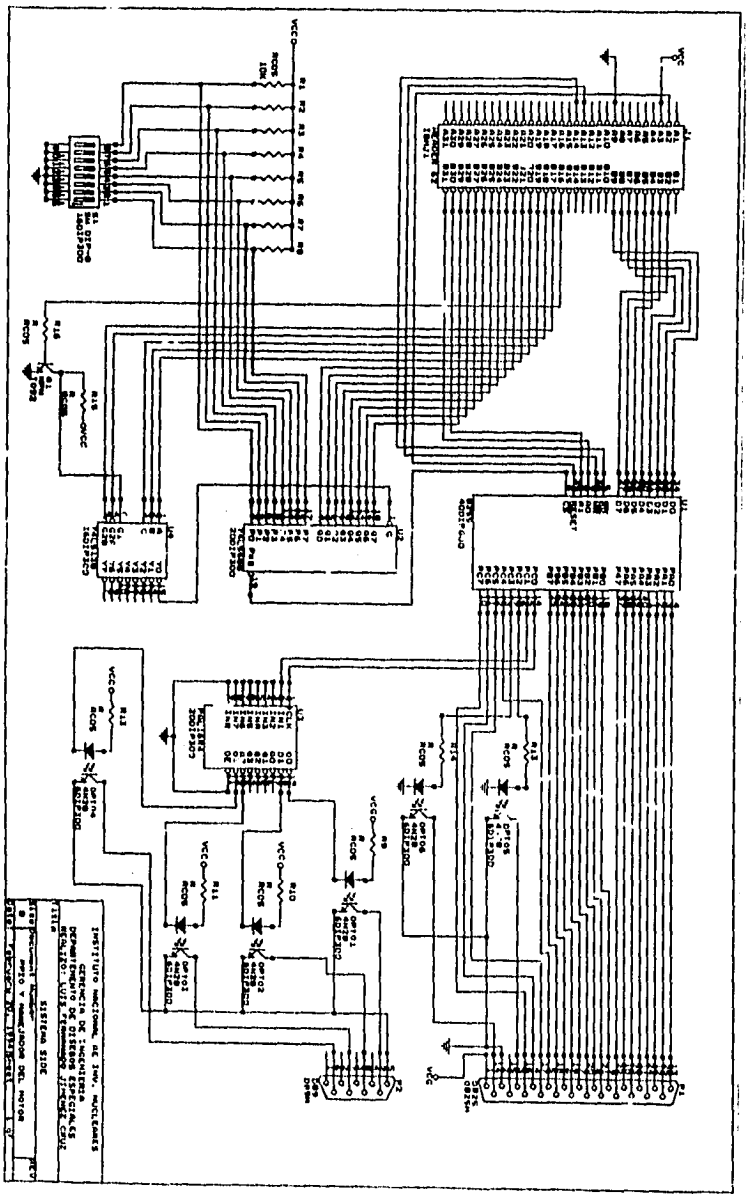


Diagrama eléctrico de la tarjeta BETA

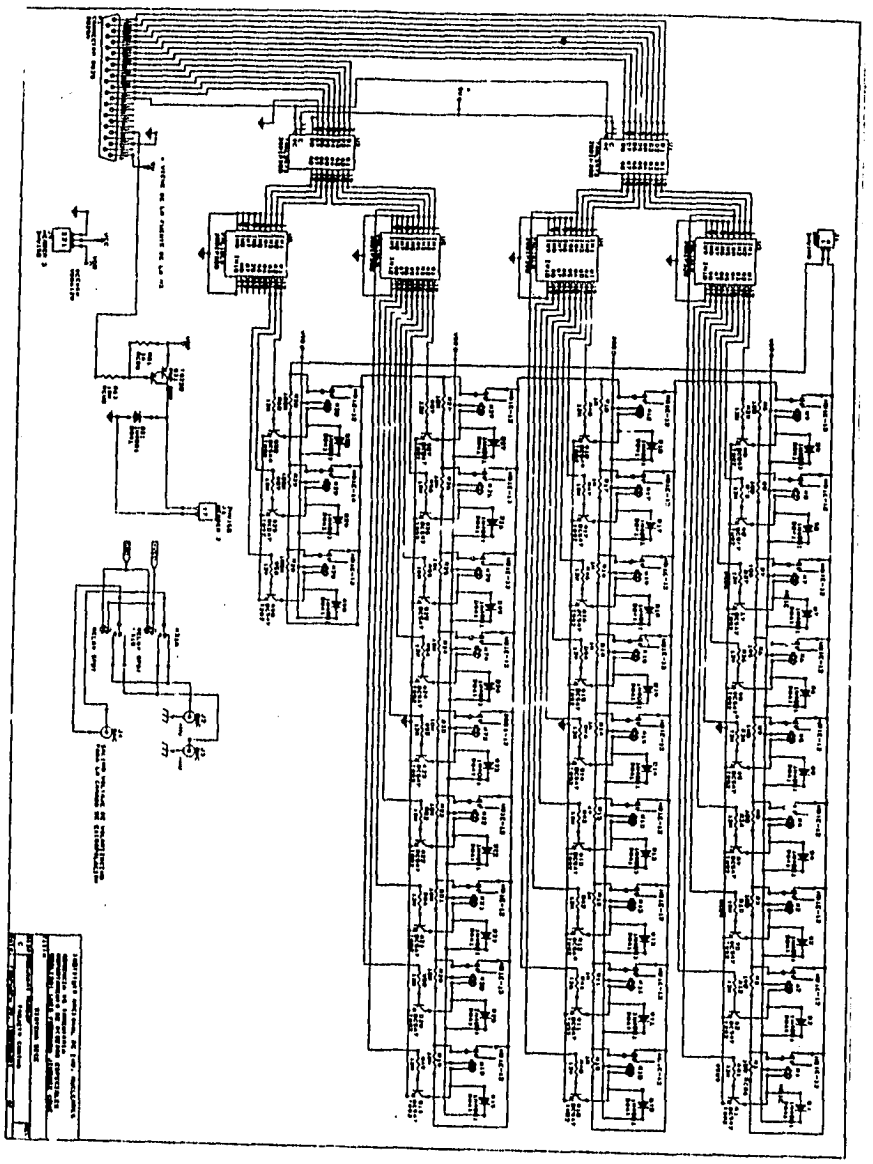


Diagrama eléctrico de la tarjeta CONTHV

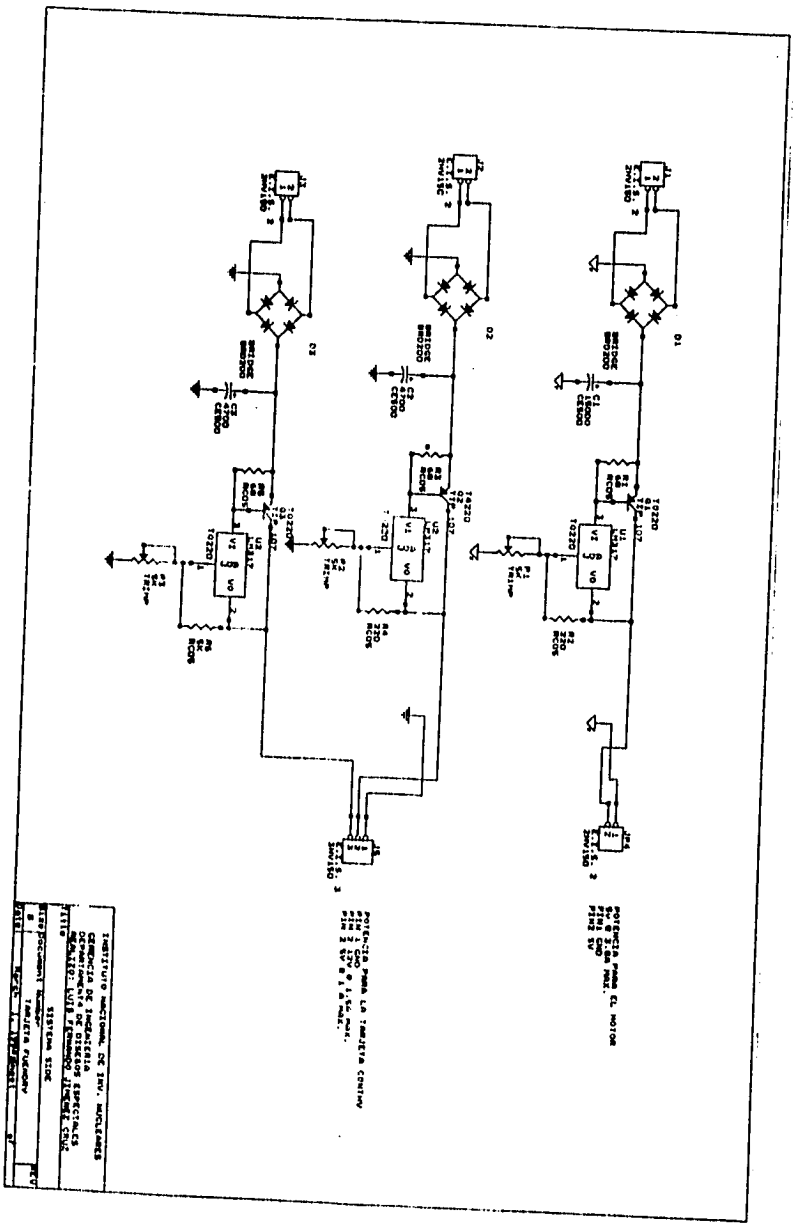


Diagrama eléctrico de la tarjeta FUENDRV

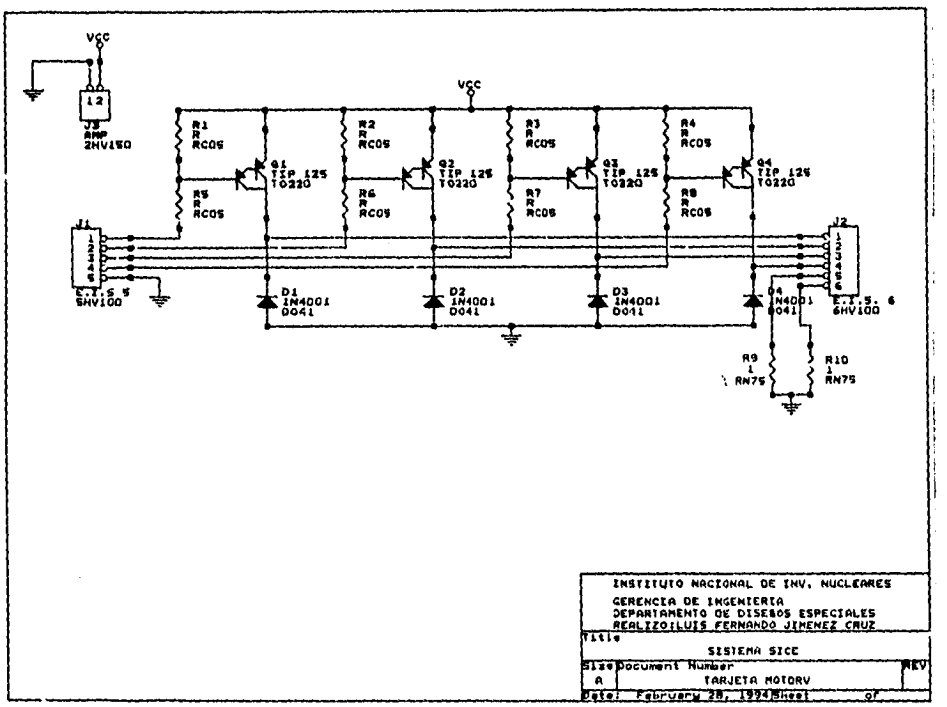
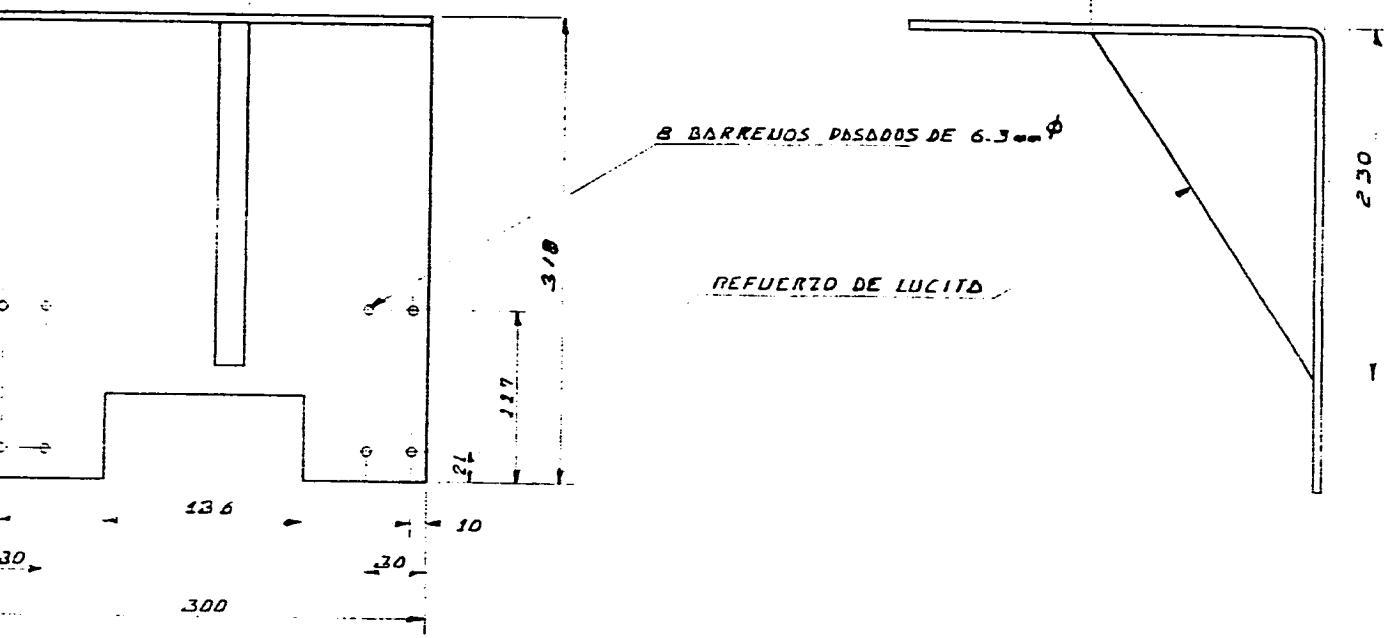
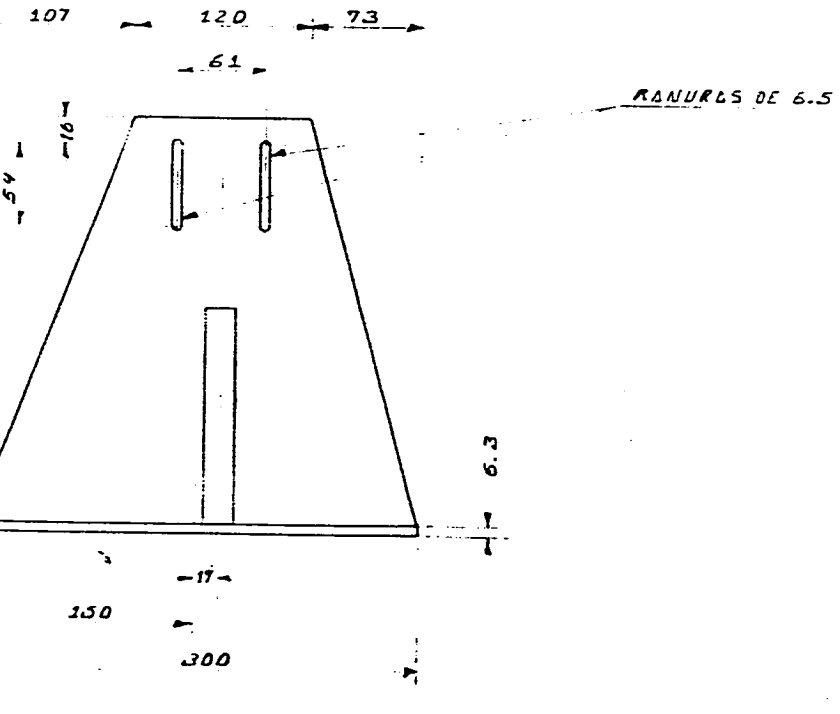


Diagrama eléctrico de la tarjeta MOTDRV

ANEXO
B

SISTEMA MECÁNICO





PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIO
	1	SOPORTE	ALUMINIO	

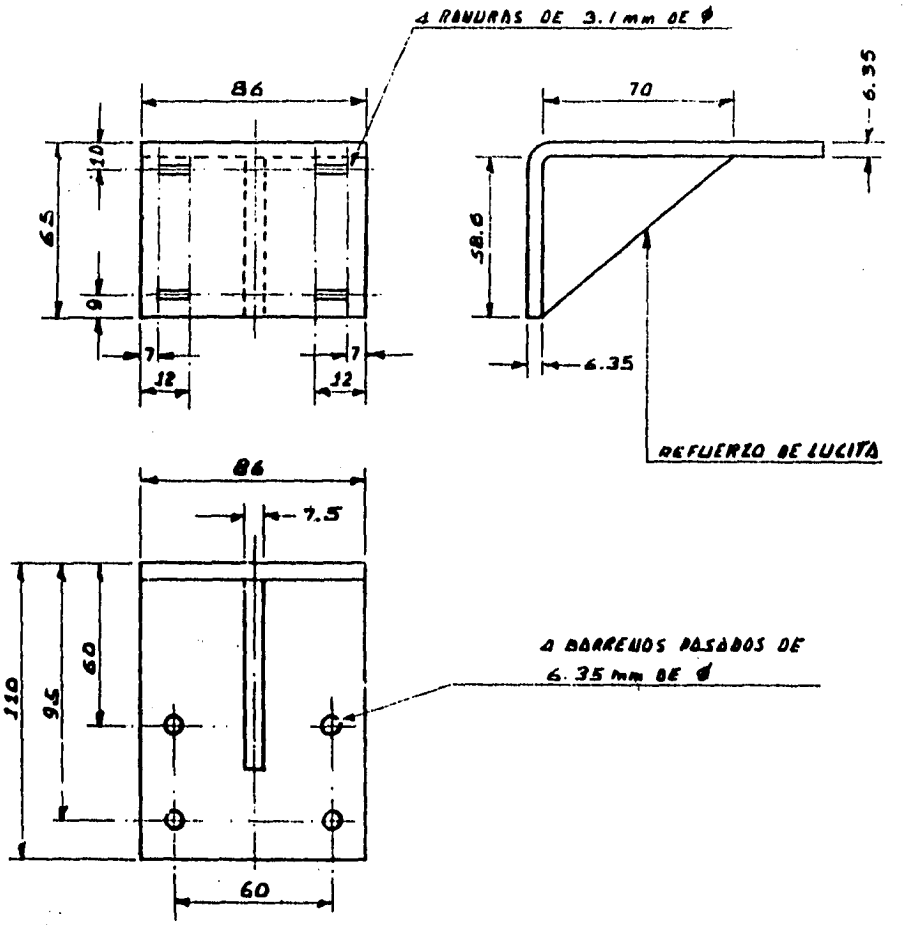
	NOMBRE	FECHA
DISEÑO	ING.	
DIBUJO	M.J.J.N.	
REVISO		
APROBO		

ININ

Instituto nacional de investigaciones
Centro Nuclear Salazar, Edo.

GCIA. DE INGEIERIA

TITULO: SOPORTE



PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA
	1	SOPORTE	ALUMINIO		

	NOMBRE	FECHA
DISEÑO		
DIBUJO	H.J.N.N	
REVISO		
APROBO		

ESCALA 1:2 ACOT. : mm

Instituto nacional de investigaciones nucleares

Centro Nuclear — Salazar, Edo. de México

GUIA DE INGENIERIA

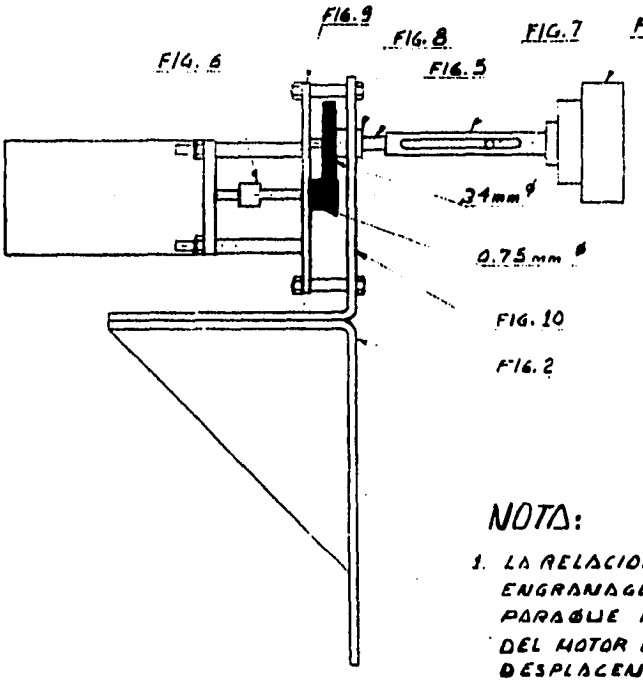
TITULO: SOPORTE

PROYECTO:

PLANO No. F.16.2

ARREGLO MECANICO

ENSAMBLADO



NOTA:

1. LA RELACION IDEAL DE ENGRANAGE ES DE 1:2.5 PARA QUE POR CADA PASO DEL MOTOR LAS PLACAS SE DESPLACEN 1 MICRA LOS DIAMETROS SENALADOS SON DE LOS ENGRANES QUE SE ENCONTRARON.

PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA

	NOMBRE	FECHA
DISEÑO		
DIBUJO	M.J.d.M.	
REVISO		
APROBO		
ESCALA S/E	ACOT. S/ACOT.	

ININ

Instituto nacional de investigaciones nucleares

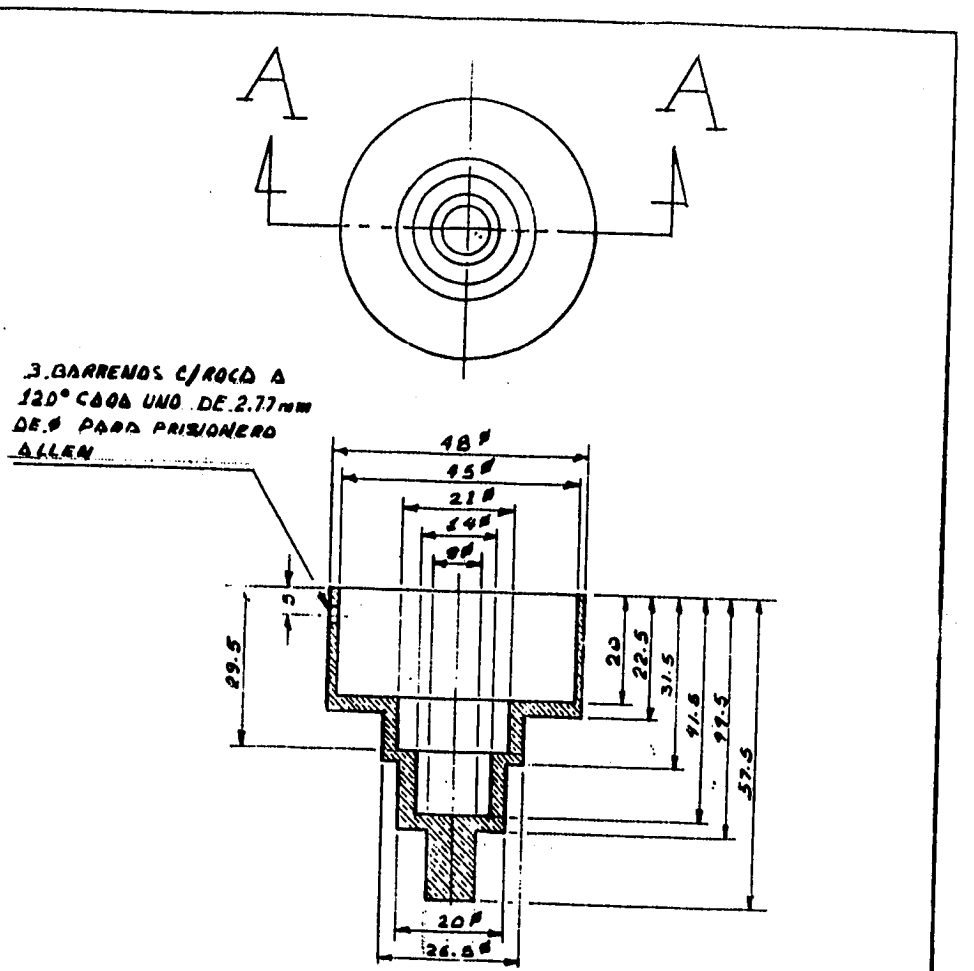
Centro Nuclear Salazar, Edo. de México

GERENCIA DE INGENIERIA

TITULO: ARREGLO MECANICO

PROYECTO:

PLANO No. FIG. 3



CORTE A - A'

PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA
	1	ACOPLADOR	LATON		

	NOMBRE	FECHA
DISEÑO		
DIBUJO	M.d.J.M.	
REVISO		
APROBO		

ESCALA 1:1 ACOT. : mm

.ININ.

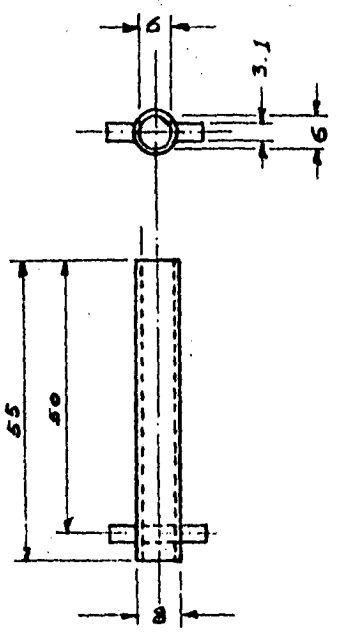
Instituto nacional de investigaciones nucleares
Centro Nuclear Salazar, Edo. de México

GERENCIA DE INGENIERIA

TITULO: ACOPLADOR

VECTOR

FIG.



PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA
			6a. de Aragón		

	NOMBRE	FECHA
DISEÑO		
DIBUJO		
REVISO	M.J. J.U	
APROBO		
ESCALA 1:1	ACOT. m/m	

ININ

Instituto nacional de investigaciones nucleares

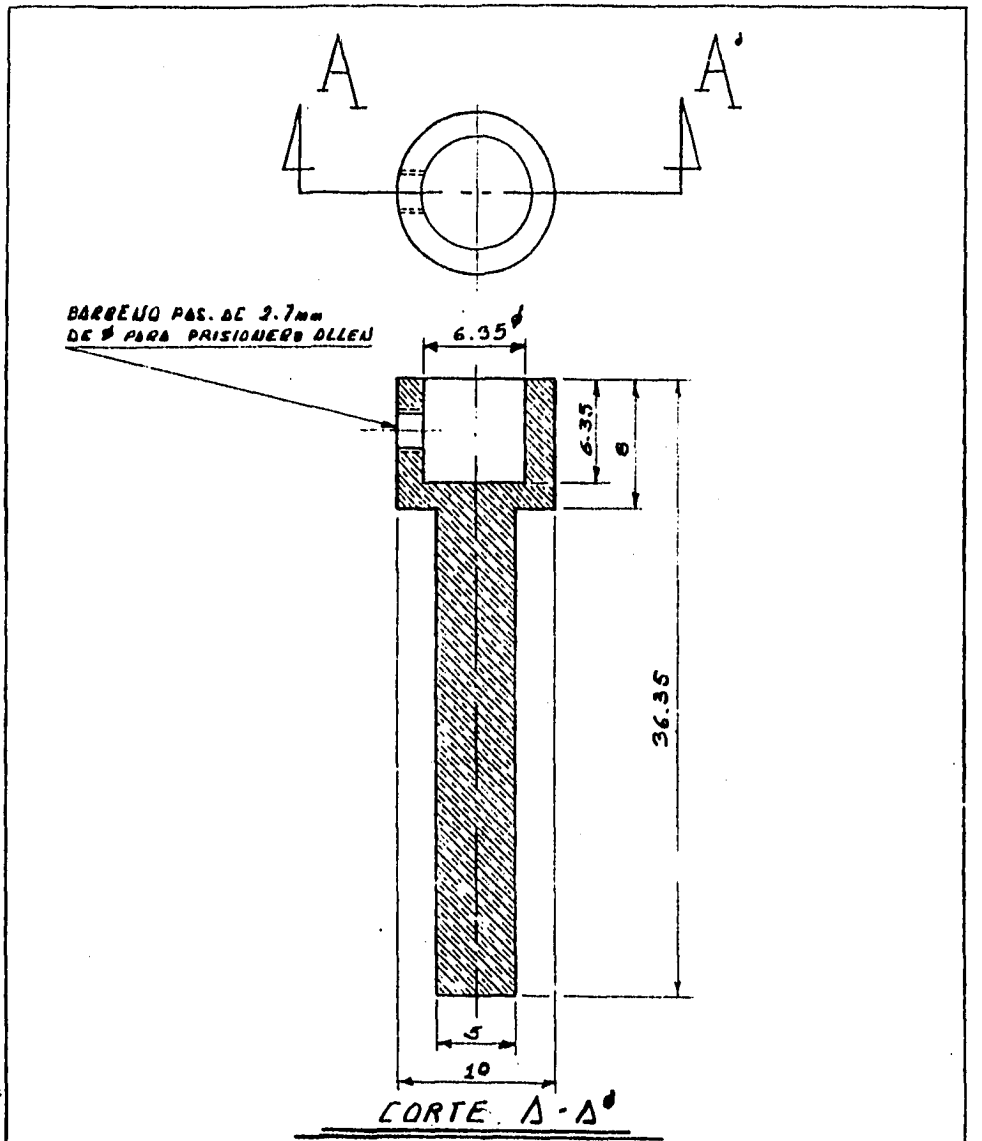
Centro Nuclear — Salazar, Edo. de México

GERENCIA DE INGENIERIA

TITULO:

PROYECTO:

PLANO No. Fig. 5



PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA
			LATON		

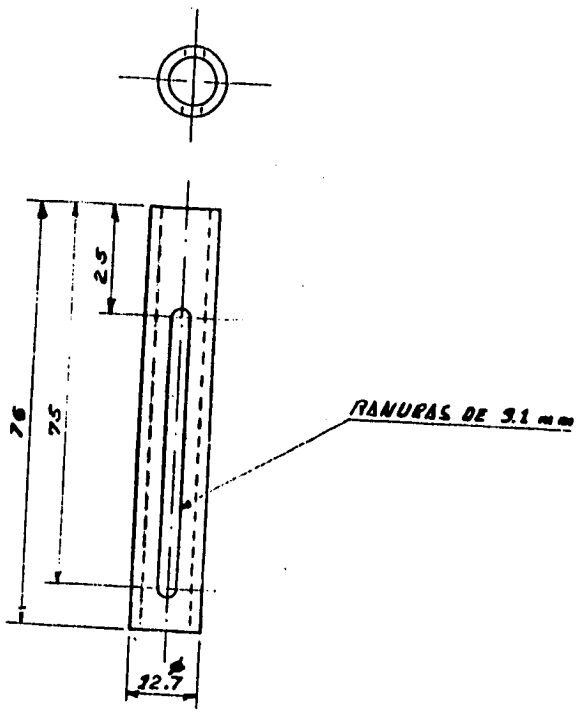
	NOMBRE	FECHA
DISEÑO		
DIBUJO		
REVISO		
APROBO		

ESCALA 1:1 ACOT.

Instituto nacional de investigaciones nucleares
Centro Nuclear Salazar, Edo. de México
GERENCIA DE INGENIERIA

TITULO

PLANO No.



PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA

	NOMBRE	FECHA
DISEÑO		
DIBUJO		
REVISO	M. J. J. U	
APROBO		

ESCALA 1:1 ACOT. mm

ININ

Instituto nacional de investigaciones nucleares

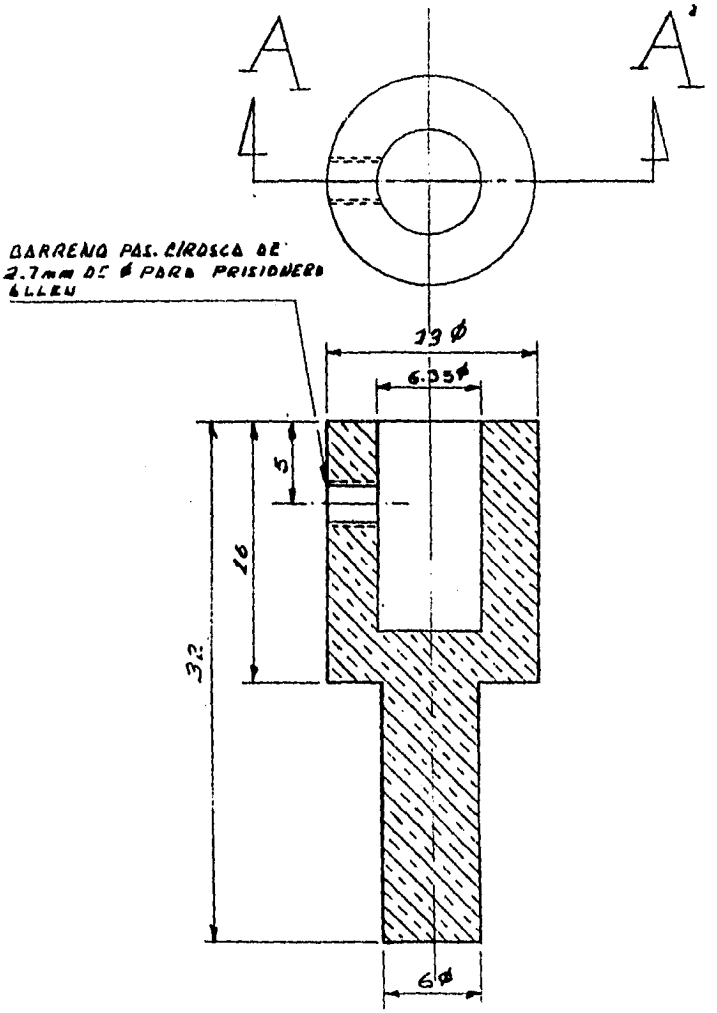
Centro Nuclear Salazar, Edo. de México

GERENCIA DE INGENIERIA

TITULO:

PROYECTO:

PLANO No. F1



CORTE A-A'

PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA
1			LATON		

	NOMBRE	FECHA
DISEÑO		
DIBUJO	M. J. J. N.	
REVISO		
APROBO		

ESCALA 1:1 ACOT. mm

ININ

Instituto nacional de investigaciones nucleares

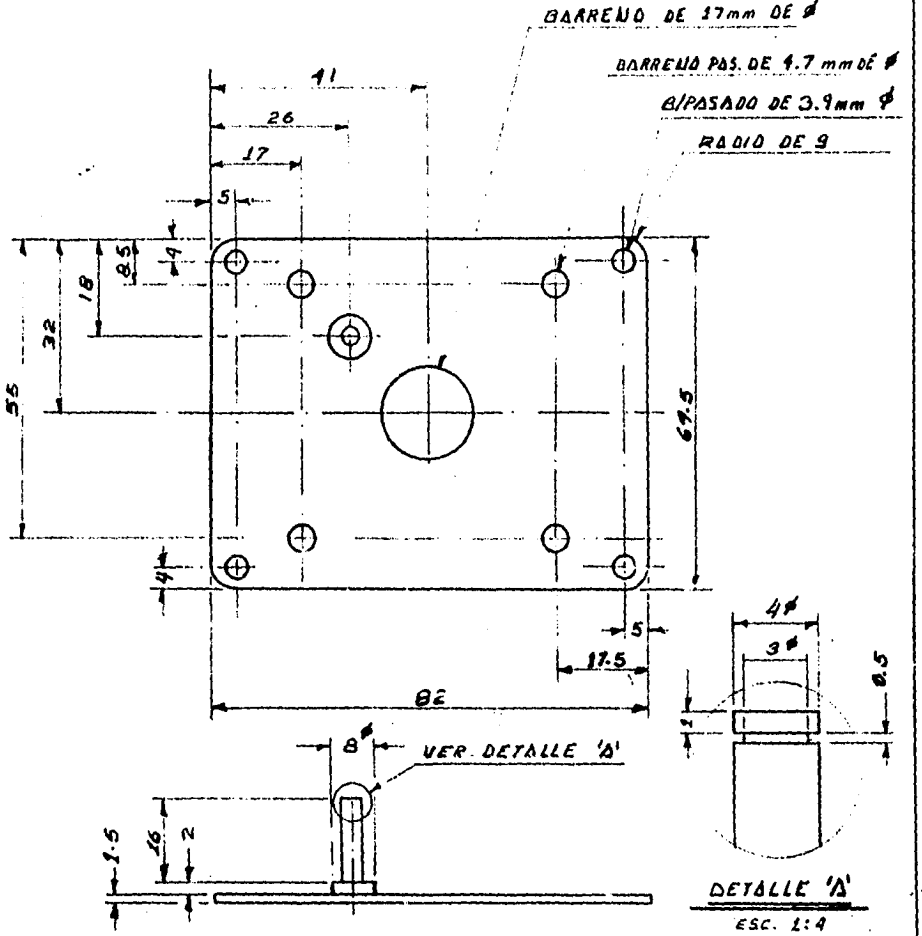
Centro Nuclear Salazar, Edo. de México

GERENCIA DE INGENIERIA

TITULO:

PROYECTO:

PLANO No. FIG. 8



PZA. No.	CANT.	NOMBRE	MATERIAL	OBSERVACIONES	REFERENCIA

	NOMBRE	FECHA
DISEÑO		
DIBUJO		
REVISO	M.J.J.B.	
APROBO		

ESCALA 1:1 ACOT. mm

ININ

Instituto nacional de investigaciones nucleares

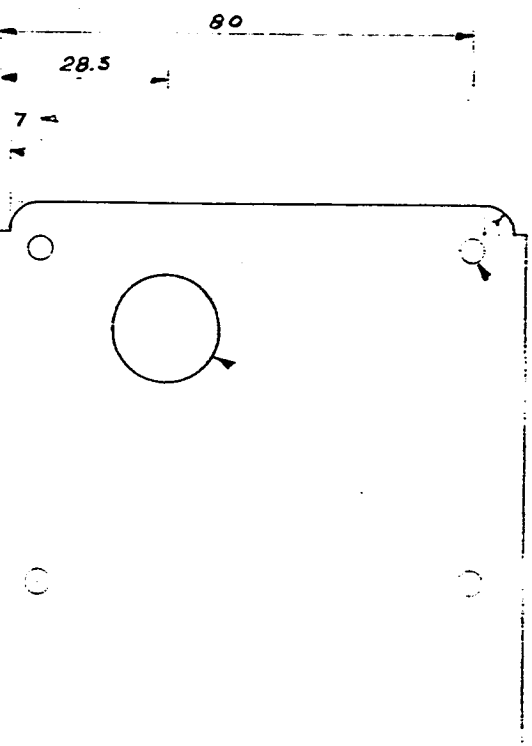
Centro Nuclear Salazar, Edo. de México

GERENCIA DE INGENIERIA

TITULO:

PROYECTO:

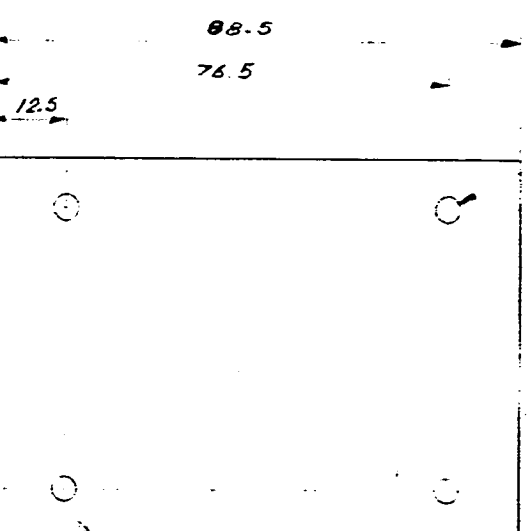
PLANO No. Fig. 9



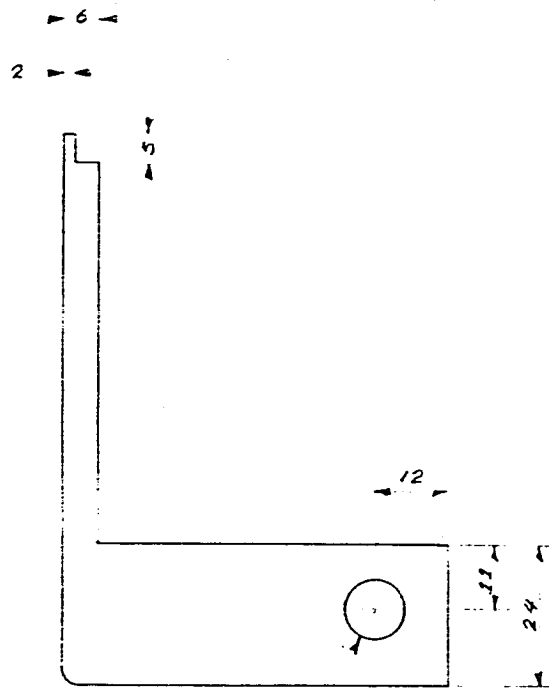
R=5

4 BARRENOS PDS. DE
3.9 mm DE Ø

BARRENO PDS. DE
12.7mm DE Ø



4 BARRENOS PDS. DE
3.9 mm DE Ø



BARRENO PDS. DE
10 mm DE Ø

PZA No.	CANT	NOMBRE	MATERIAL	OBSERVACI

	NOMBRE	FECHA
DISEÑO		
DIBUJO		
REVISO		

Instituto nacional de investigacione
Centro Nuclear Salazar, Ed

ANEXO
C

LISTADO DEL PROGRAMA

```
/*****************************************************************************
************                        GLOBAL.H                        ************
************          DEFINICION DE VARIABLES GLOBALES              ************
*****************************************************************************/

int inicio=20000;              /* distancia inicial entre placas */
char *rango_electrometro,      /* rango del electrometro         */
   *funcion_electrometro;      /* funcion del electrometro       */
RADIACION extrapolacion;
AMBIENTE ambiente;
USUARIO  usuario;
PROCESO  proceso;
OTRAPL   aplicacion;
float  afondo[50];     /* arreglo donde se almacena la inf del fond*/
FILE   *fp;            /* apuntador del archivo de trabajo         */
float  aambiente[3][600]; /* arreglo donde se almacena cond. amb.  */
unsigned int  nambiente,
              ndato,
              nfondo,
              ndistancia;
int       cuentadistancia;
float     referencia=0.0; /*voltaje inicial de la fuente           */
int       polref=POSITIVO; /*polaridad inicial de la fuente        */
unsigned int  acumulador=0x00;/* varible para el manejo de bits de salida */
int       bandera;
float     voltaje_pos[30];
float     voltaje_neg[30];


/*****************************************************************************
**********                        CONSTANT.H                        **********
**********  DEFINICION DE CONSTANTES UTILIZADAS EN EL PROGRAMA  **********
*****************************************************************************/

/********** CONSTANTES PARA EL MANEJO DEL MOTOR **********/
#ifndef ADELANTE
#define ADELANTE 1
#define ATRAS !ADELANTE
#endif
#define VELOCIDAD 500

/********** CONSTANTES PARA EL MANEJO DE LA FUENTE ******/
#ifndef POSITIVO
#define NEGATIVO 0
#define POSITIVO !NEGATIVO
#endif
/********** CONSTANTES GENERALES *************************/
#ifndef SI
#define NO 0
#define SI !NO
#endif
#ifndef TRUE
#define FALSE 0
#define TRUE  1
```

```
#endif
#define TEMPERATURA 0
#define PRESION    1
#define HUMEDAD    2


/*******   VALORES DE LAS TECLAS PARA SELECCION POR BARRA ILUMINADA *******/
#define CR      13      /**** ENTER       *******/
#define UP      72      /**** 0 -> arriba  *******/
#define LEFT    75      /**** 0 -> izquerda *******/
#define DOWN    80      /**** 0 -> abajo   *******/
#define RITE    77      /**** 0 -> derecha *******/

/************** CONSTANTES PARA AUTOTEST ***********************/
#define EPSILON0 8.85e-12
#define AREA    7.1617E-4


/*****************************************************************************/
/*******                          VENTBETA.H                           *******/
/*******          archivo cabecera que define las variables y funciones *******/
/*******          de la biblioteca de funciones VENTBETA.C              *******/
/*****************************************************************************/
#ifndef SI
#define SI 1
#define NO !SI
#endif

typedef struct {
        char nombre[40],     /* nombre del solicitante delcservicio */
           fuente[15];     /* tipo de fuente a medir          */
        int    funcion;     /* funcion del electrometro        */
                            /* si funcion=1 mide "coulombs"    */
                            /* si funcion=0 mide "amperes"     */
        } USUARIO;
typedef struct{
        float temperatura,  /* temperatura ambiente            */
            presion,     /* presion atmosferica             */
            humedad;     /* humedad relativa                */
        }AMBIENTE;
typedef struct{ float  voltaje,
                       gradiente;  /* gradiente de voltaje a utilizar*/
        unsigned int  distancia;  /*distancia entre placas        */
                 int  lecturas;   /* num. de lecturas a adquirir   */
        unsigned int  tiempo;     /* retardo entre lectura y lectura*/

        }PROCESO;
 typedef struct{float gradiente;   /* gradiente de voltaje a utilizr */
    unsigned int  di,          /* distancia inicial de las placas*/
                  df;          /* distancia final de las placas  */

    unsigned int  tiempo;        /* tiempo entre mediciones       */
          int  paso,         /* intervalo entre distancias    */
```

```
                    lecfondo,
                    lecturas;     /* num. de lecturas por cada dist.*/
   unsigned int  amb:1,              /* bandera para lec. de inf. amb. */
                    func:1;

        }RADIACION;
typedef struct{ unsigned int   distancia[20];
                float gradiente[20];} OTRAPL;
void informacion_fecha(void);
void informacion_proceso(PROCESO *proceso, int imprime);
void informacion_ambiente(AMBIENTE *ambiente,int imprime);
void informacion_usuario(USUARIO *usuario,int imprime);
void cmri(void);

/* biosarea.h: ROM BIOS data area at 0x0040:0 in memory */

#ifndef byte
#define byte unsigned char /* define byte como un tipo */
#endif
/* CAMPOS DE BIT USADOS EN EL AREA DE DATOS ROM BIOS */
typedef struct {
unsigned hasFloppies : 1, /* 1= system has floppy drives */
         nu1 : 1,                 /* not used */
         mbRAM : 2, /* mothereboard RAM size (obsoiete) */
         initVideo : 2,       /* initial videob mode */
         nDsiks : 2,          /* nbr of floppy drives */
         nu8 :1,                  /* not used */
         nSerialPorts : 3, /* nbr de serial ports attached*/
         gamePort : 1,        /* 1= game port attached */
         nu13 : 1,                /* not used */
         nLPT : 2;            /* number of printers */
} EQFLAGS;     /* estructura de las banderas del equipo */

typedef struct {
unsigned riteShiftDown : 1,   /* 1=rigth shift key down */
         leftShiftDown : 1,   /* 1=left shift key down  */
         ctrlShiftDown : 1,   /* 1=ctrl-shift combo down */
         altShiftDown : 1,    /* 1=alt-shift combo down */
         scrollLockOn : 1,    /* 1=scroll lock mode on */
         numLockOn : 1,       /* 1=num lock mode on */
         capsLockOn : 1,      /* 1=caps lock mode on */
         insOn : 1,           /* 1= ins mode on */
         unused : 3,              /* spare bit */
         ctrlNumLockOn : 1,   /* 1=ctrl-NumLock mode on */
         scrollLockDown : 1,  /* 1=scroll lock key down */
         numLockDown : 1,     /* 1=num lock key down */
         capsLockDown : 1,    /* 1=caps lock key Down */
         insDown : 1;         /* 1=ins key down */
}KBDFLAGS;   /* estrtuctura de las banderas del teclado */

typedef struct {

 unsigned serialPortAdrr[4];
 unsigned parallelPortAdrr[4];
```

```
 EQFLAGS  eqpFlags;
 byte     mfgrTestFlags;
 unsigned mainMem;
 unsigned expRAM;
 KBDFLAGS kbdStat;
 byte     keypad;
 unsigned kbdBufferHead;
 unsigned kbdBuffTail;
 char     kbdBuff[32];
 byte     seekStat;
 byte     motorStat;
 byte     motorcnt;
 byte     DiskErr;
 byte     NECStatus[7];
 byte     videoMode;
 unsigned scrnWidth;
 unsigned vidBuffSz;
 unsigned vidBuffOfs;
 byte     cursPos[8][2];
 byte     cursBottom;
 byte     cursTop;
 byte     activeDispPage;
 unsigned activeDispPort;
 byte     CRTModeReg;
 byte     palette;
 unsigned dataEdgeTimeCount;
 unsigned CRCReg;
 char     LastInputValue;
 unsigned tick;
 int      hour;
 byte     timeOverflow;
 byte     brkStat;
 unsigned resetFlag;
 long     hardDiskStat;
 byte     parallelTimeout[4];
 byte     serialTimeout[4];
 unsigned kbdBuffOfs;
 unsigned kbdBuffEnd;
} BIOSDATA;

/******** CURSOR.H funciones prototipo para la apariencia del cursor ******/
void cursoff(void);  /***** desaparece el cursor *******/
void curson(void);   /***** aparece el cursor **********/
void cursShape(int top, int bottom); /**** cambia la forma ******/

/*****************************************************************
*********                      fuendrv.h                    *********
*****************************************************************/
void fuendrv(float *voltaje, int *polvolt,float *referencia,
             int *polref,float *incremento,int *retardo),
   incrementa(float *voltaje,float *referencia,float *incremento,
                  int *retardo,int *polaridad, int *polvolt),
   decrementa(float *referencia,float *incremento,
                  int *retardo,int *polaridad, int *polvolt),
```

```
   enviadato(float *referencia);

/*************************************************************************
********                POPUP.H : prototipos y definicion detipos para la        ********
********                biblioteca  POPUP.C                                      ********
*************************************************************************/
extern struct text_info;        /* de conio.h*/
typedef struct {
 int left, top, right, bottom,   /* localizacion de limites */
    style,                  /* estilo del limite */
    normal,hilite,          /* atributos del texto */
    normback,hiback,border;     /* atributos del fondo */
 char *text;                /* texto a desplegar */
 void *save;                /* apuntador al buffer */
 struct text_info prev;         /* estado anterior del video */
}POPUP;
typedef struct{
 int row,                   /* linea donde aparece la barra */
    interval,               /* columna entre el primer caracter */
    fore, back;             /* color del fondo y del primer plano */
 char *choice;              /* apuntador al contenido del texto */
}MENUBAR;
void popShow(POPUP *pop);       /* funcion para aparecer una ventana */
void popErase(POPUP *pop);      /* funcion para borrar una pantala  */
void popCenter(POPUP *win,char *string);
void popRewrite(POPUP *win, int row, char attrib);
void popHilite(POPUP *win, int row);
void popNormal(POPUP *win, int row);
```

```
/*****************************************************************************
******        INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES          ******
******                          BETA1488.C                             ******
****** Programa para la operación del sistema de interfaz para la cámara de extrapola- ******
****** ción del patron secundario beta del CMRI del ININ.              ******
****** Diseñado en el departamento de Diseños Especiales de la Gerencia de Ing. ******
****** por : LUIS FERNANDO JIMÉNEZ CRUZ                               ******
*****************************************************************************/
#include <conio.h>
#include <dos.h>
#include <stdio.h>
#include <ctype.h>
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\textbox.h"
#include "a:\ventana\popup.h"
#include "a:\ventana\biosarea.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\constant.h"
#include "a:\ventana\global.h"

/***************** DECLARACION DE FUNCIONES PROTOTIPO ****************/
void salir(void),
    errado(POPUP *),
    operacion(int),
    presentacion(void);

void main(void)
{
/**** DEFINIENDO VENTANAS ******/

POPUP menuprincipal={29,11,51,16,2,WHITE,BLACK ,GREEN,LIGHTGRAY,BLACK};
POPUP letrero={11, 1, 70, 4, 1, WHITE, BLUE, 1, 3,WHITE };
POPUP error={11,20,40,22,2,WHITE,WHITE,RED,CYAN,YELLOW,
        "error vuelve a seleccionar!"};
char opcion ,        /* seleccion para menu principal*/
                archivotrabajo[40], /* nombre del archivo a trabajar */
                *leyenda1=" SISTEMA  AUTOMATICO  DE  MEDICION\r\n",
                *menutext=
                " A  ADQUIRIR DATOS\r\n B  AUTOTEST\r\n C  IMPRIMIR DATOS\r\n
Q  SALIR";
int continua=TRUE,      /* indica si continua en el programa */
        looping,
        indic;          /* indicador de opcion     */

/*****************************************************************************
******                PROGRAMANDO LOS PUERTOS DEL 8255              ******
******                     DIRECCION BASE H300                      ******
*****************************************************************************/
outportb(0x0303,0x80);   /** puerto A (H0300) ->              ***/
                         /** puerto B (H0301) ->              ***/
                         /** puerto C (H0302) parte baja ->   ***/
                         /** puerto C (H0302) parte baja <-   ***/
outportb(0x301,0x40);    /** inicializando los puertos con ceros ******/
outportb(0x300,0x00);
```

```
 outportb(0x302,0x03);
 outportb(0x301,0x00);
 /*****************************************************************************/
/*****************          PANTALLA DE PRESENTACIÓN          ***************/
/*****************************************************************************/
 clrscr();
/* curson();*/
 presentacion();
 gotoxy (6,15);
/* cprintf("nombre del archivo de trabajo?");
 scanf("%s",archivotrabajo);*/
 clrscr();
 continua=TRUE;
 while(continua)
 {
  /* cursoff();*/
  /********** definiendo pantalla principal *************/
  window(1,1,80,25);
  textcolor(14);
  textbackground(15);
  clrscr();
  window(13,2,71,5);
  textbackground(BLACK);
  clrscr();
  popShow(&letrero);
  popCenter(&letrero,leyenda1);
  /***********************************************/
  /************  DESPLIEGA EL MENU  *************/
  /***********************************************/
  /******** dibuja la sombre **********/
  window(30,12,52,17);
  textbackground(0);
  clrscr();

  /******** muestra menu y se selecciona opcion ***********/
  menuprincipal.text=menutext;
  popShow(&menuprincipal);

  /****** seleccion de opcion por barra iluminada o por presion ****/
  /******                de la letra de identificacion                ****/
  indic=1;
  looping=TRUE;
  while(looping){
   popHilite(&menuprincipal,indic);
   opcion=toupper(getch());
   popNormal(&menuprincipal,indic);
   switch(opcion){
    case 'A':
                  /* curson(); */
                  operacion(1);
                  looping=FALSE;
                  break;
    case 'B': operacion(2);
                  looping=FALSE;
```

```
                    break;
    case 'C': operacion(3);
                    looping=FALSE;
                    break;
    case 'Q': continua=FALSE;
                    looping=FALSE;
                    break;
    /*** si se presiona enter en la opcion resaltada ****/
    case CR : if(indic != 4)
                    {
                     operacion(indic);
                     looping=FALSE;
                    }
                    else
                    {
                     looping=FALSE;
                     continua=FALSE;
                    }
                    break;
    /***** si el usuario presiona las flechas *****/
    case 0  : opcion=getch();
                    switch(opcion){
                     case UP:
                     case LEFT: if(--indic==0)
                                    indic=4;
                                    popHilite(&menuprincipal,indic);
                                    break;
                     case DOWN:
                     case RITE: popNormal(&menuprincipal,indic);
                                if(++indic==5)
                                    indic=1;
                                popHilite(&menuprincipal,indic);
                                break;
                    }
                    break;
    default : errado(&error);
                    window(30,12,51,18);
                    break;
   }
  }
 }
/************ limpia la pantalla y sale  ***********/
 window(1,1,80,25);
 textcolor(7);
 textbackground(BLACK);
 clrscr();
 /*curson();*/
}
```

```
/*************************************************************************/
/******                        ADQDI488.C                           ******/
/****** RUTINA PARA ADQUIRIR INFORMACIÓN DEL ELECTRÓMETRO  ******/
/*************************************************************************/
#include <stdlib.h>
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#include <alloc.h>
#include "a:\ventana\ieeeio.h"
#include "a:\ventana\textbox.h"
#include "a:\ventana\popup.h"
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\constant.h"
#include "a:\ventana\iottc20.h"

/*****************************************/
/*****          variables globales          *****/
/*****************************************/
extern int inicio;
extern char *rango_electrometro,
            *funcion_electrometro;
extern RADIACION extrapolacion;
extern AMBIENTE  ambiente;
extern USUARIO   usuario;
extern PROCESO   proceso;
extern OTRAPL    aplicacion;
extern float     afondo[];
extern FILE *fp; /*** apuntador del archivo *****/
extern float  aambiente[3][600];
extern float  referencia;
extern  int polref;
extern unsigned int  nambiente,
                     ndato,
                     nfondo,
                     ndistancia;
extern int        cuentadistancia;
extern float      voltaje_pos[],
                     voltaje_neg[];

/*****************  declaracion de funciones utilizadas ***************/
 void        otras_aplicaciones(void);
 void        curvas_de_extrapolacion(void);
 void        elIEEE488(void),
                drvmotor(unsigned int paso,unsigned int direccion,unsigned int dist_inicial),
                fuendrv(float *,int *,float *,int *,
                        float *,int *),
                medir_ambiente(void),
                tomar_lecturas_fondo(unsigned int *distancia_operacion,float *voltaje,int polaridad,
                                     unsigned long int fondo_tiempo),
                procesa_fondo(float *desv_std,float *promedio),
                guarda_datos_fondo(void),
```

```
                cuando_medir(void),
                toma_lecturas_extrapolI488(unsigned int *,float [][30],float [][30]),
                medir_ambientel488(void),
        near    guarda_datosotapl(float [][30],float [][30]);
int       ajustar_distancia(int *distancia_operacion);
void      adquisicion(int);
int       ieeeinit(void);
void      termina(unsigned int *);
void  near  guarda_voltajeotapl(void);
/*************************************************************/


void adquieredatos(void)
{

/****** definiendo ventanas *********/
POPUP    menu={11,15,60,19,1,WHITE,YELLOW,BLUE,LIGHTGRAY,WHITE};

int indic,
        looping;
char opcion,
                continua = 'S',
                *opciones =
                "A CONSTRUCCION DE CURVAS DE EXTRAPOLACICN\r\nB OTRAS
APLICACIONES\r\n\
S SALIR";

/****** LIMPIANDO PANTALLA ******/
window(1,1,80,25);
textcolor(WHITE);
textbackground(CYAN);
clrscr();
cmri();
/*****************************************************************************
*****           SE INICIALIZA LA INTERFAZ IEEE488 SE INICIALIZA EL      *****
*****             ELECTROMETRO Y SE DEFINEN:                            *****
*****           -CANALES A OPERAR EN EL ADC                             *****
*****           -FORMA DE ENVIAR LA INFORMACION DEL ADC                 *****
*****           -INTERVALO Y TIEMPO DE MUESTREO DEL ADC                 *****
*****************************************************************************/
if(ieeeinit() != -1)
{
/*****************************************************************************/
/*****           SE ESTABLECE LA DIRECCION DEL ADC (14)                  *****/
/*****************************************************************************/
ieeewt("abort\n");
ieeewt("remote 14\n");
ieeewt("clear 14\n");
/*****************************************************************************/
/**** SE ESTABLECE LA DIRECCION DEL ELECTROMETRO (27), SE PONE EN:    *****/
/****                  MODO REMOTO       REMOTE 27                   *****/
/****                  LECTURA CON PREFIJO  G0                       *****/
/****                  LECTURA POR LLAMADA  T1                       *****/
/*****************************************************************************/
```

```
ieeewt("remote 27\n");
ieeewt("output 27; T1G0X\n");

/*************************************************************************/
/*****      SE ESTABLECE LA DIRECCION DEL MODULO DIGITAL (9)          *****/
/*****     SE PONE ENN MODO REMOTO  REMOTE 9                          *****/
/*****     SE SELECCIONA EL PUERTO 5  P5                              *****/
/*****     SE ESPECIFICAN COMO SALIDAS C5                             *****/
/*****    SE UTILZA EL CANAL 1 Y SE CONFIGURA PARA SEA SALIDAS DE     *****/
/*****    ALTA CORRIENTE POR HARDWARE                                 *****/
/*************************************************************************/
ieeewt("remote 9\n");
ieeewt("output 9; C5P5\n");
ieeewt("output 9; A39A40\n");

/*************************************************************************
**********    se muestran las opciones a utilizar y se hace la     *****
**********    seleccion por presion de la letra de identificacion  *****
**********    o por barra iluminada                                *****
*************************************************************************/
menu.text=opciones;
while(continua=='S')
{
        popShow(&menu);
        indic= 1;
        looping=TRUE;
        while(looping){
         popHilite(&menu,indic);
         opcion=toupper(getch());
         popNormal(&menu,indic);
         switch(opcion)
         {
                case 'A': adquisicion(1);
                looping=FALSE;
                break;
                case 'B': adquisicion(2);
                looping=FALSE;
                break;
                case 'S': continua=FALSE;
                looping=FALSE;
                break;
/****** si se presiona enter  en la opcion resaltada :  ************/
                case 13 : if(indic!=3)
                {
                 adquisicion(indic);
                 looping=FALSE;
                }
                else
                {
                 looping=FALSE;
                 continua=FALSE;
                }
                break;
/******* si el usuario presiona las flechas:          ************/
```

```
                case 0 : opcion=getch();
                switch(opcion){
                 case UP:
                 case LEFT: if(--indic==0)
                                indic=3;
                                popHilite(&menu,indic);
                                break;
                 case DOWN:
                 case RITE: popNormal(&menu,indic);
                                if(++indic==4) indic=1;
                                popHilite(&menu,indic);
                                break;
                          }

                          break;
                default: break;
        }
        }
 printf("desea continuar?");
 continua=toupper(getch());
}
}
else
{
 window(22,11,58,15);
 textcolor(YELLOW);
 textbackground(RED);
 clrscr();
 gotoxy(2,2);
 printf("NO SE PUEDE INICIALIZAR EL SISTEMA");
 gotoxy(2,3);
 printf("PRESIONE ENTER PARA CONTINAUR");
 getche();
}
}

void adquisicion(int seleccion)
{

 POPUP menu={11,15,60,19,1,WHITE,YELLOW,BLUE,WHITE,WHITE};
 switch(seleccion)
 {

  case 1:  curvas_de_extrapolacion();
                break;
  case 2:  otras_aplicaciones();
                popErase(&menu);
                break;
 }
}

/*************************************************************************/
/*****                    FUNCIÓN OTRAS_APLICACIONES( )                  *****/
/*************************************************************************/
```

```
void otras_aplicaciones(void)
{
 POPUP     electrometro={11,8,70,23,2,BLACK,WHITE,WHITE,BLACK,BLUE},
           archivo={6,8,57,10,1,WHITE,BLACK,MAGENTA,WHITE,BLACK},
           ventana5={2,11,45,21,2,MAGENTA,MAGENTA,WHITE,WHITE,WHITE},
           ventana6={3,23,78,25,2,0,0,15,15,0},
           aviso_distancia_inicial={20,8,61,10,2,WHITE,YELLOW,BLUE,RED,WHITE};


float  datopos[20][30],
       datoneg[20][30],
       voltaje=0.0,
       desv_std=0.0,
       incremento=0.1,
       promedio_ruido_de_fondo,
       auxf;

 char nombre_archivo[40]; /**** nombre del archivo donde se guarda la inf. */
 int  polvolt= POSITIVO,
              paso,
               i,
              j,
              retardo=100;
 unsigned int  fondo_tiempo;
 unsigned int  difondo;
 unsigned int dist_operacion;
 /*****************     pantalla de presentacion     ******************/
 window(1,1,80,25);
 textbackground(CYAN);
 textcolor(WHITE);
 clrscr();
 cmri();
 informacion_usuario(&usuario,NO);
 informacion_fecha();
 informacion_ambiente(&ambiente,NO);
 informacion_proceso(&proceso,NO);
 /********** presentacion de la ventana de adquisicion de datos *******/
 popShow(&archivo);
 window(archivo.left+11,archivo.top,archivo.right-12,archivo.top);
 textcolor(archivo.normal);
 textbackground(archivo.normback);
 clrscr();
 gotoxy(1,1);

 /*******     peticion del nombre del archivo de trabajo y apertura     *******/
 /*******     del mismo                                                 *******/

 printf("NOMBRE DEL ARCHIVO DE TRABAJO");
 window(archivo.left+1,archivo.top+1,archivo.right-1,archivo.bottom-1);
       /*************************************************************/
       popShow(&ventana6);

       extrapolacion.func=0;
       window(4,24,77,24);
```

```
        textcolor(ventana6.normal);
        textbackground(ventana6.normback);
        clrscr();
        gotoxy(1,1);
        printf("nombre del usuario?  ");
gets(usuario.nombre);
gotoxy(1,1);
clreol();
cprintf("tipo de fuente? ");
scanf("%s",usuario.fuente);
clrscr();
informacion_usuario(&usuario,SI);
/*************************************************************/
/*******              pedir informacion para leer fondo                ******/
/*************************************************************/
window(ventana5.left+1,ventana5.top+1,ventana5.right-1,ventana5.top+1);
textcolor(ventana5.normal);
textbackground(ventana5.normback);
clrscr();
gotoxy(13,1);
clreol();
cprintf("LECTURAS DE FONDO");
window(4,24,77,24);
textcolor(ventana6.normal);
textbackground(ventana6.normback);
clrscr();
/**********  LEYENDO DISTANCIA ENTRE PLACAS  **********/
do
{
 gotoxy(1,1);
 clreol(),
 cprintf("distancia entre placas (en micras)?  ");
 scanf("%u",&extrapolacion.di);
 if(extrapolacion.di>inicio || extrapolacion.di<100)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  if(extrapolacion.di>inicio)
       cprintf("LA DISTANCIA MAXIMA ES DE %D MICRAS!",inicio);
  if(extrapolacion.di < 100)
       cprintf("LA DISTANCIA MINIMA ES DE 100 MICRAS!");
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
  printf("ERROR");
  delay(1000);
  popErase(&aviso_distancia_inicial);
```

```
 }
}while(extrapolacion.di>inicio||extrapolacion.di<100);
/****** LEYENDO GRADIENTE  **********/
do
{

 gotoxy(1,1);
 clreol();
 cprintf("gradiente? ");
 scanf("%f",&extrapolacion.gradiente);
 if(extrapolacion.gradiente>50.0)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  if(extrapolacion.gradiente>50.0)
       cprintf("EL GRADIENTE MAXIMO ES DE 50 V/cm");
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
  printf("ERROR");
  delay(1000);
  popErase(&aviso_distancia_inicial);
 }
}while(extrapolacion.gradiente>50.0);
/********* LEYENDO NUMERO DE LECTURAS *******/
do
{
 gotoxy(1,1);
 clreol();
 cprintf("numero de lecturas? ");
 scanf("%d",&extrapolacion.lecfondo);
 if(extrapolacion.lecfondo>30)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  if(extrapolacion.gradiente>50.0)
       cprintf("EL NO. DE LECTURAS MAX. ES DE 30");
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
```

```
   printf("ERROR");
   delay(1000);
   popErase(&aviso_distancia_inicial);
  }
 }while(extrapolacion.lecfondo>30);
 /********* LEYENDO TIEMPO ENTRE LECTURAS *******/
 do
 {
  gotoxy(1,1);
  clreol();
  cprintf("intervalo entre lecturas (en segundos)?  ");
  scanf("%u",&fondo_tiempo);
  if(fondo_tiempo>65535l)
  {
   popShow(&aviso_distancia_inicial);
   window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
   aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
   textcolor(aviso_distancia_inicial.normal);
   textbackground(aviso_distancia_inicial.normback);
   clrscr();
   gotoxy(1,2);
   if(fondo_tiempo>65535l)
        cprintf("EL INTERVALO MAX. ES DE 65535l");
   window(37,8,41,8);
   textcolor(aviso_distancia_inicial.border);
   textbackground(aviso_distancia_inicial.normback);
   clrscr();
   gotoxy(1,1);
   printf("ERROR");
   delay(1000);
   popErase(&aviso_distancia_inicial);
  }
 }while(fondo_tiempo>65535l);
 gotoxy(1,1);
 clreol();
 proceso.gradiente=extrapolacion.gradiente;
 proceso.distancia=extrapolacion.di;
 proceso.lecturas=extrapolacion.lecfondo;
 proceso.tiempo=fondo_tiempo;
 voltaje=(double)(extrapolacion.gradiente * extrapolacion.di /10000.0);
 proceso.voltaje=voltaje;
 informacion_proceso(&proceso,SI);
 window(4,24,77,24);
 textcolor(ventana6.normal);
 textbackground(ventana6.normback);
 gotoxy(1,1);
 clreol();
 popShow(&electrometro);
 window(electrometro.left+1,electrometro.top+1,electrometro.right-1,
        electrometro.bottom-1);
 textcolor(electrometro.normal);
 textbackground(electrometro.normback);
 clrscr();
 gotoxy(2,5);
```

```
elIEEE488();
gotoxy(32,5);
getch();
popErase(&electrometro);

/*************************************************************/
/**********                  medir fondo                **********/
/*************************************************************/

if(inicio != extrapolacion.di)
{
 cprintf("ajustando distancia inicial a %d micras",extrapolacion.di);
 paso=inicio-extrapolacion.di;
 drvmotor(paso,ADELANTE,inicio);
 gotoxy(1,1);
 clreol();
 cprintf("distancia entre placas ajustada a %d micras",extrapolacion.di);
 delay(2000);
 gotoxy(1,1);
 clreol();
}
else
{
 cprintf("distancia inicial %d micras",inicio);
 delay(1000);
}
/**********  ajustando voltaje de polarizacion *****************/
gotoxy(1,1);
clreol();
cprintf("AJUSTANDO EL VOLTAJE DE A: %lg V",voltaje);
fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
gotoxy(1,1);
clreol();
cprintf("VOLTAJE AJUSTADO");
medir_ambientel488();
informacion_ambiente(&ambiente,SI);
popShow(&ventana6);
window(4,24,77,24);
textcolor(ventana6.normal);
textbackground(ventana6.normback);
clrscr();
difondo=extrapolacion.di;
tomar_lecturas_fondo(&difondo,&voltaje,POSITIVO,fondo_tiempo);
procesa_fondo(&desv_std,&promedio_ruido_de_fondo);
gotoxy(1,1);
clreol();
printf("RUIDO DE FONDO PROMEDIO: %lg ",promedio_ruido_de_fondo);
gotoxy(1,1);
getch();
clreol();
printf("INCERTIDUMBRE: %lg",desv_std);
getch();
gotoxy(1,1);
clreol();
```

```
printf("DESEA CONTINUAR TOMANDO MEDICIONES?(s/n)");

if(toupper(getch()) == 'S')
{
 gotoxy(1,1);
 clreol();
 cprintf("EL NUMERO DE LECTURAS QUE FIJE SE USARA SUBSECUENTEMENTE");
 delay(2000);
 /******* LEYENDO NUMERO DE LECTURAS CON FUENTE EXPUESTA ******/
 do
 {
  gotoxy(1,1);
  clreol();
  cprintf("NUMERO DE LECTURAS:  ");
  scanf("%d",&extrapolacion.lecturas);
  if(extrapolacion.lecturas>30)
  {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  if(extrapolacion.lecturas>30)
       cprintf("EL No. DE LECTURAS MAX. ES DE 30");
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
  printf("ERROR");
  delay(1000);
  popErase(&aviso_distancia_inicial);
 }
}while(extrapolacion.lecturas>30);
proceso.lecturas=extrapolacion.lecturas;
/******* LEYENDO TIEMPO ENTRE LECTURAS  *******/
do
{
 gotoxy(1,1);
 clreol();
 printf("tiempo entre lectura y lectura (segundos): ");
 scanf("%u",&extrapolacion.tiempo);
 if(extrapolacion.tiempo>65535l)
 {
      popShow(&aviso_distancia_inicial);
      window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
      aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
      textcolor(aviso_distancia_inicial.normal);
      textbackground(aviso_distancia_inicial.normback);
      clrscr();
      gotoxy(1,2);
      if(extrapolacion.tiempo>65535l)
```

```
        cprintf("EL INTERVALO MAXIMO ES DE 65535 S.");
        window(37,8,41,8);
        textcolor(aviso_distancia_inicial.border);
        textbackground(aviso_distancia_inicial.normback);
        clrscr();
        gotoxy(1,1);
        printf("ERROR");
        delay(1000);
        popErase(&aviso_distancia_inicial);
  }
 }while(extrapolacion.tiempo>65535L);
 gotoxy(1,1);
 clreol();
 dist_operacion=extrapolacion.di;
 fwrite(&usuario,sizeof(USUARIO),1,fp);
 fwrite(&extrapolacion,sizeof(RADIACION),1,fp);
 fwrite(&ambiente,sizeof(AMBIENTE),1,fp);
 guarda_datos_fondo();
 cuentadistancia=0;
 do
 {
 /*************************************************************/
 /*********        pedir informacion para tomar lecturas        **********/
 /*************************************************************/
 if (cuentadistancia != 0)
 {
  do
  {
        gotoxy(1,1);
        clreol();
        cprintf("distancia entre placas (en micras)?  ");
        scanf("%u",&extrapolacion.di);
        if(extrapolacion.di>inicio || extrapolacion.di<100)
        {
          popShow(&aviso_distancia_inicial);
          window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
          aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
          textcolor(aviso_distancia_inicial.normal);
          textbackground(aviso_distancia_inicial.normback);
          clrscr();
          gotoxy(1,2);
          if(extrapolacion.di>inicio)
            cprintf("LA DISTANCIA MAXIMA ES DE %D MICRAS!",inicio);
          if(extrapolacion.di<100)
            cprintf("LA DISTANCIA MINIMA ES DE 100 MICRAS!");
          window(37,8,41,8);
          textcolor(aviso_distancia_inicial.border);
          textbackground(aviso_distancia_inicial.normback);
          clrscr();
          gotoxy(1,1);
          printf("ERROR");
          delay(1000);
          popErase(&aviso_distancia_inicial);
        }
```

```
}while(extrapolacion.di>inicio||extrapolacion.di<100);
do
{

     gotoxy(1,1);
     clreol();
     cprintf("gradiente? ");
     scanf("%f",&extrapolacion.gradiente);
     if(extrapolacion.gradiente>50.0)
     {
      popShow(&aviso_distancia_inicial);
      window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
      aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
      textcolor(aviso_distancia_inicial.normal);
      textbackground(aviso_distancia_inicial.normback);
      clrscr();
      gotoxy(1,2);
      if(extrapolacion.gradiente>50.0)
      cprintf("EL GRADIENTE MAXIMO ES DE 50 V/cm");
      window(37,8,41,8);
      textcolor(aviso_distancia_inicial.border);
      textbackground(aviso_distancia_inicial.normback);
      clrscr();
      gotoxy(1,1);
      printf("ERROR");
      delay(1000);
      popErase(&aviso_distancia_inicial);
     }
}while(extrapolacion.gradiente>50.0);

}
else
{
 gotoxy(1,1);
 clreol();
 printf("distancia entre placas (micras): %d  micras",extrapolacion.di);
 getch();
 gotoxy(1,1);
 clreol();
}

informacion_proceso(&proceso,SI);
window(ventana5.left+1,ventana5.top+1,ventana5.right-1,ventana5.top+1);
textcolor(ventana5.normal);
textbackground(ventana5.normback);
clrscr();
gotoxy(10,1);
clreol();
cprintf("LECTURAS DE RADIACION");
window(4,24,77,24);
textcolor(ventana6.normal);
textbackground(ventana6.normback);
clrscr();
gotoxy(1,1);
```

```
cuando_medir();
aplicacion.gradiente[cuentadistancia]=extrapolacion.gradiente;
aplicacion.distancia[cuentadistancia]=extrapolacion.di;

/*************************************************************/
/******                    MEDIR                          ******/
/*************************************************************/
paso=dist_operacion-extrapolacion.di;
drvmotor(paso,ADELANTE,dist_operacion);
toma_lecturas_extrapolI488(&extrapolacion.di,datopos,datoneg);
if(extrapolacion.amb == 0)
{
 medir_ambienteI488 ();
 aambiente[TEMPERATURA][cuentadistancia]=ambiente.temperatura;
 aambiente[PRESION][cuentadistancia]=ambiente.presion;
 aambiente[HUMEDAD][cuentadistancia]=ambiente.humedad;
}

/*************************************************************/
/************          DESEA CONTINUAR          **************/
/*************************************************************/

cuentadistancia+=1;
proceso.lecturas=cuentadistancia;
dist_operacion=extrapolacion.di;
gotoxy(1,1);
clreol();
printf("desea continuar tomando lecturas? (s/n)");

}while(toupper(getch())=='S');
/**********   regreseando a condicionesiniciales  ********/
window(1,1,80,25);
clrscr();
termina(&dist_operacion);
}
else
{
gotoxy(1,1);
clreol();
printf("RUIDO DE FONDO MUY GRANDE");
fwrite(&usuario,sizeof(USUARIO),1,fp);
extrapolacion.lecturas=0;
fwrite(&extrapolacion,sizeof(RADIACION),1,fp);
fwrite(&ambiente,sizeof(AMBIENTE),1,fp);
guarda_datos_fondo();
}

/*************************************************************/
/************          guarda datos          ************/
/*************************************************************/
gotoxy(1,1);
clreol();

printf("GUARDANDO INFORMACION EN EL ARCHIVO");
```

```
  fwrite(&proceso,sizeof(PROCESO),1,fp);
  if(extrapolacion.amb==0 && cuentadistancia>1)
  {
   for(i=0;i<3;i++)
   {
    for(j=0;j<cuentadistancia-1;j++)
    {
     auxf=aambiente[i][j];
     fwrite(&auxf,sizeof(float),1,fp);
    }
   }
  }
  guarda_datosotapl(datopos,datoneg);
  fwrite(&aplicacion,sizeof(OTRAPL),1,fp);
  guarda_voltajeotapl();
  fclose(fp);
  gotoxy(1,1);
  clreol();

  /*************************************************************/
  /*******                    desea realizar mas mediciones              *******/
  /*************************************************************/
  /*printf("DESEA REALIZAR OTRO CONJUNTO DE MEDICIONES? (s/n)");*/
}
void near  guarda_datosotapl(float adatovp[][30] ,float adatovn[][30])
{
 int i,
    j;
 float auxd;
 /********** escribiendo en el archivo datos con vp ************/
 for(i=0;i<cuentadistancia;i++)
 {
  for(j=0;j<extrapolacion.lecturas+2;j++)
  {
   auxd=adatovp[i][j];
   fwrite(&auxd,sizeof(float),1,fp);
  }
 }
 /********escribiendo en el archivo datos con vn  *************/
 for(i=0;i<cuentadistancia;i++)
 [
  for(j=0;j<extrapolacion.lecturas+2;j++)
  {
   auxd=adatovn[i][j];
   fwrite(&auxd,sizeof(float),1,fp);
  }
 }
}
void near guarda_voltajeotapl()
{
 int  i;
 float auxf;

 for(i=0;i<cuentadistancia;++i)
```

```
{
 auxf=voltaje_pos[i];
 fwrite(&auxf,sizeof(float),1,fp);
}
for(i=0;i<cuentadistancia;++i)
{
 auxf=voltaje_neg[i];
 fwrite(&auxf,sizeof(float),1,fp);
}
}
```

```
/*****************************************************************************
**************                    ELI488.C                        **************
**************      funcion para seleccionar la funcion del electrometro      **************
**************      y el rango de operacion.                    **************
*****************************************************************************/
#include "a:\ventana\ieeeio.h"
#include <ctype.h>
#include <conio.h>
#include <stdio.h>
#include <dos.h>
#include "a:\ventana\biosarea.h"
#include "a:\ventana\popup.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\iottc20.h"
extern USUARIO usuario;
extern char *funcion_electrometro;
extern char *rango_electrometro;

/**********      DECLARACION DE FUNCIONES PROTOTIPO *********/
void error(void),       /* detecta error de seleccion de rango */
    seleccion(void),    /* letrero */
    r0(void),          /*************************************************/
    r1(void),          /*                                               */
    r2(void),          /*        envian el codigo del                   */
    r3(void),          /*                                               */
    r4(void),          /*                                               */
    r5(void),          /*        rango respectivo                       */
    r6(void),          /*                                               */
    r7(void),          /*                                               */
    r8(void),          /*        al electrometro                        */
    r9(void),          /*                                               */
    r10(void),         /*                                               */
    r11(void),         /*************************************************/
    ventana_electrometro(void);

void elIEEE488(USUARIO *usuario)
{
 char funcion,    /***  clave de seleccion de la funcion  del elec. ***/
     rango;      /***  clave de seleccion del rango del electrometro */
 int  bandera_funcion=1, /**** bandera del loop de parametros del 617  **/
     bandera_rango;  /**** bandera del loop de rango del electrometro */
  while(bandera_funcion==1)
  {
   ventana_electrometro();
   gotoxy(20,2);
   printf("FUNCIONES  DISPONBLES:");
   gotoxy(18,6);
   printf(" A          AMPERES");
   gotoxy(18,7);
   printf(" C          COULOMBS");
   gotoxy(14,9);
   printf("PRESIONE LA LETRA DE LA OPCION DESEADA");
   funcion=toupper(getch());
```

```
switch(funcion)
{
 case 'A': gotoxy(1,14);
                funcion_electrometro="AMPERES";
                usuario->funcion=0;
                ieeewt("output 27;F1X\n");
                printf("output 27;F1X");
                bandera_rango=1;
                while(bandera_rango)
                {
                 ventana_electrometro();
                 gotoxy(12,1);
                 printf("LOS RANGOS DISPONIBLES SON:");
                 gotoxy(12,2);
                 printf("    a     auto");
                 gotoxy(12,3);
                 printf("    b     2   pA");
                 gotoxy(12,4);
                 printf("    c     20  pA");
                 gotoxy(12,5);
                 printf("    d     200 pA");
                 gotoxy(12,6);
                 printf("    e     2   nA");
                 gotoxy(12,7);
                 printf("    f     20  nA");
                 gotoxy(12,8);
                 printf("    g     200 nA");
                 gotoxy(12,9);
                 printf("    h     2   microA");
                 gotoxy(12,10);
                 printf("    i     20  microA");
                 gotoxy(12,11);
                 printf("    j     200 microA");
                 gotoxy(12,12);
                 printf("    k     2   mA");
                 gotoxy(12,13);
                 printf("    l     20  mA");
                 gotoxy(1,14);
                 seleccion();
                 rango=getch();
                 window(12,22,69,22);
                 textcolor(WHITE);
                 textbackground(BLUE);
                 clrscr();
                 gotoxy(1,1);
                 switch(rango){
                  case 'a':
                  case 'A': r0();
                            rango_electrometro="AUTORANGO";
                            bandera_rango=0;
                            break;
                  case 'b':
                  case 'B': r1();
                            rango_electrometro="2 pA";
```

```
            bandera_rango=0;
            break;
case 'c':
case 'C': r2();
            rango_electrometro="20 pA";
            bandera_rango=0;
            break;
case 'd':
case 'D': r3();
            rango_electrometro="200 pA";
            bandera_rango=0;
            break;
case 'e':
case 'E': r4();
            rango_electrometro="2 nA";
            bandera_rango=0;
            break;
case 'f':
case 'F': r5();
            rango_electrometro="20 nA";
            bandera_rango=0;
            break;
case 'g':
case 'G': r6();
            rango_electrometro="200 nA";
            bandera_rango=0;
            break;
case 'h':
case 'H': r7();
            rango_electrometro="2 microA";
            bandera_rango=0;
            break;
case 'i':
case 'I': r8();
            rango_electrometro="20 microA";
            bandera_rango=0;
            break;
case 'j':
case 'J': r9();
            rango_electrometro="200 microA";
            bandera_rango=0;
            break;
case 'k':
case 'K': r10();
            rango_electrometro="2 mA";
            bandera_rango=0;
            break;
case 'l':
case 'L': r11();
            rango_electrometro="20 mA";
            bandera_rango=0;
            break;
default: error();
}
```

```
                    }
                    bandera_funcion=0;
                    break;
case 'C':  ventana_electrometro();
                    gotoxy(1,14);
                    ieeewt("output 27;F3X\n");
                    printf("output 27;F3X\n");
                    funcion_electrometro="COULOMBS";
                    usuario->funcion=1;
                    bandera_rango=1;
                    while(bandera_rango)
                    {
                     gotoxy(16,5);
                     cputs("LOS RANGOS DISPONIBLES SON:");
                     gotoxy(12,6);
                     cputs("        a       auto");
                     gotoxy(12,7);
                     cputs("        b       200 pC");
                     gotoxy(12,8);
                     cputs("        c       2   nC");
                     gotoxy(12,9);
                     cputs("        d       20  nC");
                     gotoxy(1,14);
                     seleccion();
                     rango=getch();
                     window(12,22,69,22);
                     textcolor(WHITE);
                     textbackground(BLUE);
                     clrscr();
                     switch(rango)
                     {
                      case 'a':
                      case 'A': r0();
                                rango_electrometro="AUTORANGO";
                                bandera_rango=0;
                                break;
                      case 'b':
                      case 'B': r1();
                                rango_electrometro="200 pC";
                                bandera_rango=0;
                                break;
                      case 'c':
                      case 'C': r2();
                                rango_electrometro="2 nC";
                                bandera_rango=0;
                                break;
                      case 'd':
                      case 'D': r3();
                                rango_electrometro="20 nC";
                                bandera_rango=0;
                                break;
                      default: error();

                     }
```

```
                    }
                    bandera_funcion=0;
                    break;
    default: error();
             break;

   }
   cprintf("  %d",usuario->funcion);
  }
}

/***************** definiendo seleccion() ************************/
void seleccion()
{
 printf("PRESIONE LA LETRA DE LA OPCION DESEADA");
}
/******************** definiendo error() ************************/
void error()
{
 POPUP atencion={25,12,65,14,1,YELLOW,WHITE,RED,BLUE,BLUE};
 char  *texto="LA OPCION SELECCIONADA NO EXISTE !!!";

 atencion.text=texto;
 popShow(&atencion);
 getch();
 popErase(&atencion);
}
/***************** definiendo r0()-r11() ************************/
void r0()
{
 ieeewt("output 27;ROX\n");
 printf("output 27;R0X");
}
void r1()
{
 ieeewt("output 27;R1X\n");
 printf("output 27;R1X");
}
void r2()
{
 ieeewt("output 27;R2X\n");
 printf("output 27;R2X");
}
void r3()
{
 ieeewt("output 27;R3X\n");
 printf("output 27;R3X");
}
void r4()
{
 ieeewt("output 27;R4X\n");
 printf("output 27;R4X");
}
void r5()
```

```
{
 ieeewt("output 27;R5X\n");
 printf("output 27;R5X");
}
void r6()
{
 ieeewt("output 27;R6X\n");
 printf("output 27;R6X");
}
void r7()
{
 ieeewt("output 27;R7X\n");
 printf("output 27;R7X");
}
void r8()
{
 ieeewt("output 27;R8X\n");
 printf("output 27;R8X");
}
void r9()
{
 ieeewt("output 27;R9X\n");
 printf("output 27;R9X");
}
void r10()
{
 ieeewt("output 27;R10X\n");
 printf("output 27;R10X");
}
void r11()
{
 ieeewt("output 27;R11X\n");
 printf("output 27;R11X");
}

void ventana_electrometro()
{
 window(12,9,69,22);
    textcolor(WHITE);
    textbackground(BLUE);
    clrscr();
}
```

```
/*****************************************************************************
******                          AJUSDIST.C                                 ******
******          funcion que ajusta la distancia entre placas de la camara  ******
******          de extrapolacion y ajusta la distancia inicial entre       ******
******          las placas, recibe como parametros los parametros:         ******
******                  &distancia_operacion                               ******
******                  &inicializa                                        ******
******          regresa una bandera si ya se hizo el barrido completo de   ******
******          entre la distancia inicial y la distancia final            ******
*****************************************************************************/
#include <stdio.h>
#include <conio.h>
#include "a:\ventana\popup.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\constant.h"
extern RADIACION  extrapolacion;
extern int inicio;

/****************** DECLARACION DE FUNCIONES PROTOTIPO *********************/
void drvmotor(unsigned int,unsigned int, unsigned int);
/***************************************************************************/
int  ajustar_distancia(int *distancia_operacion)
{
 int   barrido;

  if(extrapolacion.di > extrapolacion.df)
   {
   drvmotor(extrapolacion.paso,ADELANTE,*distancia_operacion);
   *distancia_operacion -= extrapolacion.paso;
   if( *distancia_operacion < extrapolacion.df)
         barrido=0;
   else barrido=1;
   }
  else
   {
   drvmotor(extrapolacion.paso,ATRAS,*distancia_operacion);
   *distancia_operacion += extrapolacion.paso;
   if( *distancia_operacion > extrapolacion.df)
         barrido=0;
   else barrido=1;
   }
 return(barrido);
}
```

```
/*****************************************************************************
******************                    ANEXO.C                  ******************
******************                contiene las subrutinas:     ******************
******************                operacion(int tarea)         ******************
******************                presentacion (void)          ******************
******************                procesainformacion(void)     ******************
******************                salir(void)                  ******************
******************                errado(POPUP *)              ******************
*****************************************************************************/
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include "a:\ventana\textbox.h"
#include "a:\ventana\popup.h"
#include "a:\ventana\cursor.h"

/******************** FUNCIONES PROTOTIPO  ********************/
void adquieredatos(void),
   autotest(void),
   procesainformacion(void),
   presentainformacion(void);

/******** selecion de la operacion a realizar  **********/
void operacion (int tarea)
{
  switch(tarea)
  {

   case 1: window(1,1,80,25);
           /* curson();*/
            textcolor(WHITE);
            textbackground(BLACK);
            clrscr();
            adquieredatos();
            break;
   case 2:  /*curson();*/
              autotest();
              break;
   case 3: presentainformacion();
              break;
  }
}
/****** portada de presentacion  *********/
void presentacion(void)
{
 window(1,1,80,25);
 textbackground(1);
 clrscr();
 window(6,6,79,23);
 textbackground(0);
 clrscr();
 window(3,3,77,22);
 textbackground(7);
 textcolor(5);
```

```
clrscr();
textbox(2,1,74,20,1);
window(4,4,76,21);
gotoxy(5,4);
cprintf("I.N.I.N");
gotoxy(61,4);
cprintf("C.M.R.I.");
gotoxy(28,9);
cprintf("AUTOMATIZACION SALA BETA\n");
delay(2000);
}
/*************************************************************************
***********              se procesa la informacion adquirida en el paso          ***********
***********              de adquisicion de informacion                            ***********
*************************************************************************/
void procesainformacion(void)
{
 clrscr();
 gotoxy(10,12);
 printf("procesar informacion\n");
 getch();
}
/*************************************************************************
***********              se cierran archivos y se confirma si el usuario         ***********
***********              desea abandonar el programa                              ***********
*************************************************************************/
void salir(void)
{
  clrscr();
  gotoxy(1,12);
  printf("esta parte del programa cierra archivos y da un reporte\n");
  printf ("de el archivo utilizado, da un mensaje de despedida");
  getch();
}

/*************************************************************************
***********                    presenta un mensaje de error                    ***********
*************************************************************************/

void errado(POPUP *error)
{
  window(12,21,41,23);
  textbackground(BLACK);
  clrscr();
  popShow(error);
  getch();
  popErase(error);
  window(12,21,41,23);
  textbackground(WHITE);
  clrscr();
}
```

```
/*****************************************************************************
*******                          AUTOTEST.C                             *******
******* FUUNCION PARA LA CALIBRACION DE LA CAMARA DE EXTRAPOLACION ******
*****************************************************************************/
#include <conio.h>
#include <stdio.h>
#include <dos.h>
#include <stdlib.h>
#include "a:\ventana\textbox.h"
#include "a:\ventana\popup.h"
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\constant.h"
#include "a:\ventana\ieeeio.h"
extern int inicio;

/********* DECLARACION DE FUNCIONES PROTOTIPO ***************/
int   ieeeinit(void);
void  drvmotor(unsigned int,unsigned int,unsigned int),
        fuendrv(float *,int *,float *,int *,float *,int *);
int   ieeescnf(char *format, ...);
float mide_voltaje(void);
void  cero_correct(void);
void autotest()
{
 float qa[5][10],
        qb[5][10],
        auxiliar=0.0,
        permitividad_A,
        dist_medida,
        dato,
        b=0.0,
        divisor=0.0,
        numerador=0.0,
        mediacap=0.0,
        pendiente=0.0,
        capacitancia[5][12],
        referencia=0,
        voltaje=0,
        dist[5]={10000e-6,5000e-6,2500e-6,1000e-6,500e-6},
        incremento=1.0,
        area_efec;

 float v[5], auxf;
 int   polvolt=POSITIVO,
        polref=POSITIVO,
        retardo=300,
        i=0,
        j=0;
        int  distancia, paso, dist_inicial=inicio;
 POPUP ventana={4,7,76,23,1,BLACK,LIGHTBLUE,LIGHTGRAY,BLACK,BLACK};
 POPUP pedir_archivo={11,3,70,5,1,LIGHTBLUE,CYAN,BLACK,LIGHTGRAY,LIGHTCYAN};
 char  archivo[40],
        archo[40];
```

```
FILE  *apuntador_archivo;

/*****************************************************************************
*******                    habriendo ventana de presentacion                 *******
*****************************************************************************/
window(1,1,80,25);
textbackground(BLUE);
clrscr();
cmri();
popShow(&ventana);
window(5,8,75,22);
textbackground(ventana.normback);
textcolor(ventana.normal);
clrscr();
gotoxy(15,1);
puts("CALIBRACION DE LA CAMARA DE EXTRAPOLACION");

/*****************************************************************************/
/******                    configuracion del electrometro                    ******/
/*****************************************************************************/
/*****************************************************************************
**          SE INICIALIZA LA INTERFAZ IEEE488 SE INICIALIZA EL              **
**          ELECTROMETRO Y SE DEFINEN:                                       **
**               CANALES A OPERAR EN EL ADC                                  **
**              -FORMA DE ENVIAR LA INFORMACION DEL ADC                      **
**              -INTERVALO Y TIEMPO DE MUESTREO DEL ADC                      **
*****************************************************************************/
popShow(&pedir_archivo);
window(27,3,54,3);
textcolor(pedir_archivo.border);
textbackground(pedir_archivo.normback);
clrscr();
printf("NOMBRE DEL ARCHIVO DE TRABAJO");
window(pedir_archivo.left+1,pedir_archivo.top+1,
pedir_archivo.right-1,pedir_archivo.bottom-1);
textcolor(pedir_archivo.border);
textbackground(pedir_archivo.normback);
clrscr();
gets(archivo);
gotoxy(40,1);
archo[0]='\0';
strcat(archo,archivo);
strcat(archo,".TST");
if((apuntador_archivo=fopen(archo,"w")) !=NULL)
{
 popErase(&pedir_archivo);
 if(ieeeinit() != -1)
 {

 /*****************************************************************************/
 /******                SE ESTABLECE LA DIRECCION DEL ADC (14)              ****** /
 /*****************************************************************************/
/* ieeewt("abort\n");
 ieeewt("remote 14\n");
```

```
ieeewt("clear 14\n");*/

/*************************************************************************/
/**** SE ESTABLECE LA DIRECCION DEL ELECTROMETRO (27), SE PONE EN: ****/
/****                          MODO REMOTO        REMOTE 27            ****/
/****                          LECTURA CON PREFIJO  G0                 ****/
/****                          LECTURA POR LLAMADA  T1                 ****/
/*************************************************************************/
ieeewt("clear 27\n");
ieeewt("remote 27\n");
ieeewt("output 27; T1G0X\n");
ieeewt("output 27; F3R0X\n");
ieeewt("output 27; C0X\n");
/*************************************************************************/
dist_inicial=inicio;
distancia=10000;
referencia=0;
i=0;
do
{
 paso=(dist_inicial - distancia);
 drvmotor(paso,ADELANTE,dist_inicial);
 cero_correct();
 /*************************************************************************/
 /******* se ajusta voltaje a cero y se mide carga      *********/

 voltaje=0.0;
 polvolt=POSITIVO;
 fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
/* v[i]=mide_voltaje();*/
 gotoxy(2,3);
 clreol();
 /*printf("voltaje medido= %f",v[i]);*/
 printf("voltaje medido= %f",0.0);
 for(j=0;j<10;++j)
 {
  gotoxy(28,8);
  clreol();
  printf("LECTURA NO.: %d",j);
  /********** se corrige el cero del electrometro    *****/
 /* cero_correct();*/
  /**********   tomando informacion del electrometro  ************/
  ieeewt("enter 27\n");
  ieeescnf("%*4s%e",&auxf);
  dato=(float)(auxf);
  qa[i][j]=dato;
  gotoxy(28,9);
  clreol();
  cprintf("carga= %g",qa[i][j]);
  delay(350);
 }
 gotoxy(28,8);
 clreol();
 /*************************************************************************/
```

```
/*********************************************************************/

/*********************************************************************/
/********** se ajusta voltaje a 10 V y se mide carga *********/

referencia=0.0;
voltaje=10.0;
fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
referencia=10.0;
voltaje=0.0;
/********** midiendo voltaje **********/
v[i]=10.0;
gotoxy(2,3);
clreol();
printf("voltaje medido= %f",v[i]);
/*********************************************************************/
for(j=0;j<10;++j)
{
 gotoxy(28,8);
 clreol();
 printf("LECTURA NO.: %d",j);
 /********** se ajusta el cero del electrometro **********/
 /*cero_correct();*/

 /********** tomando informacion del electrometro **********/
 iecswt("enter 27\n");
 iecescnf("%*4s%e",&auxf);
 qb[i][j]=(float)(auxf);
 gotoxy(28,9);
 clreol();
 cprintf("carga= %g",qb[i][j]);
 delay(350);
}
gotoxy(28,8);
clreol();
/*********************************************************************/
/*********************************************************************/

/*********************************************************************/
/********* se cambia distancia **********/
dist_inicial=distancia;
if(distancia==2500)
  distancia=1000;
else
  distancia/=2;
 ++i;

}while(distancia > 250);

/******* ajustando voltaje y distancia a condiciones iniciales *******/
/**** ajustando voltaje ***/
referencia=10.0;
voltaje=0.0;
fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
```

```
/**** ajustando distancia ****/
paso=19500;
dist_inicial=500;
drvmotor(paso,ATRAS,dist_inicial);

/*********************************************************************/
/*********                    procesamiento de la informacion          *********/
/*********************************************************************/
/* se aplica el metodo de minimos cuadrados para obtener la recta */
/* que relaciona la capacitancia con la distancia entre placas    */

/************ determinando diferencia de carga para cada ************/
/************ distancia                                  ************/
for(i=0;i < 5;++i)
{
 for(j=0;j < 10;++j)
 {
  capacitancia[i][j]=qb[i][j] - qa[i][j];
 }
}
/*********************************************************************/

/********** se calcula el valor medio de capacitancia por ********/
/********** cada distancia                        ********/
for(i=0;i < 5;++i)
{
 capacitancia[i][10]=0.0;
 for(j=0;j<10;++j)
 {
  gotoxy(28,10);
  cprintf("voltaje = %f",v[i]);
  capacitancia[i][j]=(float)(capacitancia[i][j]/v[i]);
  capacitancia[i][10]+=capacitancia[i][j];
 }
}
for(i=0;i< 5;++i)
 capacitancia[i][10] /= 10.0;

/******** media del inverso de las capacitancias *****************/
/******** media de la distancia 0.0038 [m]       *****************/
for(i=0;i<5;++i)
 mediacap+=(1/capacitancia[i][10]);
mediacap/=5;
clrscr();
gotoxy(5,8);
cprintf("media de Ce-1 = %g",mediacap);
getche();
clrscr();
/*********************************************************************/
numerador=0.0;
for(i=0;i<5;++i)
{
  numerador += ( (1/capacitancia[i][10]) - mediacap)*(dist[i]-0.0038);
}
```

```
gotoxy(5,8);
clreol();
/*getche();
cprintf("numerador = %g",numerador);*/

divisor=0.0;
for(i=0;i<5;++i)
{
  divisor += (1/capacitancia[i][10] - mediacap)*(1/capacitancia[i][10] - mediacap);
}
gotoxy(5,8);
clreol();
/*cprintf("divisor = %g",divisor);*/
pendiente= numerador/divisor;

/************** determinando el intercepto          *********/
b=0.0038 - (pendiente * mediacap);
area_efec=pendiente/EPSILON0;

fputs("INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES\n",apuntador_archivo);
fputs("CENTRO DE METROLOGIA DE RADIACIONES IONIZANTES\n",apuntador_archivo);
fputs("CALIBRACION DE LA CAMARA DE EXTRAPOLACIION\n",apuntador_archivo);
gotoxy(5,5);
printf("recta normalizada: L= %12.7e + %12.7e Ce-1 [m]",b,pendiente);
fprintf(apuntador_archivo,"recta normalizada: L= %12.7e + %12.7e Ce-1 [m]\n",b,pendiente);
gotoxy(5,6);
printf("area efectiva = %12.7e[m^2]",area_efec);
fprintf(apuntador_archivo,"area efectiva = %12.7e [m^2]\n",area_efec);
gotoxy(5,7);
printf("L real cuando dial marca 0 = %12.7e [m^2]",b);
fprintf(apuntador_archivo,"L real cuando dial marca 0 = %12.7e [m]\n\n\n",b);
getch();

/*****************************************************************/
/******      IMPRESION DE INFORMACION CAPACITANCIA, DISTANCIA     ******/
/******      PROPUESTA Y DISTANCIA MEDIDA                         ******/
/*****************************************************************/
permitividad_A=EPSILON0*AREA;
distancia=10000;
clrscr();
gotoxy(16,2);
cprintf("distancia propuesta  distancia medida  capacitancia");
fputs("distancia propuesta  distancia medida  capacitancia\n",apuntador_archivo);
gotoxy(16,3);
cprintf(" [micrometros]        [m]          [F]");
fputs(" [micrometros]        [m]          [F]\n",apuntador_archivo);
gotoxy(16,4);
for(i=0;i<5;++i)
{
  gotoxy(16,4+i);
  dist_medida=pendiente / capacitancia[i][10];
  cprintf("    %5d       %13.8e   %13.8e",distancia,dist_medida,
          capacitancia[i][10]);
  fprintf(apuntador_archivo,"    %5d       %13.8e   %13.8e\n",distancia,dist_medida,
```

```
            capacitancia[i][10]);
   if(distancia==2500)
    distancia=1000;
   else
        distancia/=2;
 }
 getche();
 fclose(apuntador_archivo);
 if(ferror(apuntador_archivo))
  cprintf("ERROR AL CERRAR ARCHIVO");
 }
 else
 {
  window(22,11,58,15);
  textcolor(YELLOW);
  textbackground(RED);
  clrscr();
  gotoxy(2,2);
  printf("NO SE PUEDE INICIALIZAR EL SISTEMA");
  gotoxy(2,3);
  printf("PRESIONE ENTER PARA CONTINAUR");
  getche();
 }
}
else
{
 clrscr();
 cprintf("no se puede abrir el archivo, prsione ENTER");
 getch();
}
}

void cero_correct()
{
 ieeewt("output 27; C1Z0X\n");
 delay(1000);
 ieeewt("output 27; Z1X\n");
 delay(1000);
 ieeewt("output 27; C0Z0X\n");
}
```

```
/*****************************************************************/
/****                          CURSOR.C                         ****/
/****          RUTINAS PARA LA APARIENCIA DEL CURSOR            ****/
/*****************************************************************/
#include <dos.h>
#include "a:\ventana\biosarea.h"
BIOSDATA  *bios=(BIOSDATA *) MK_FP (0x0040,0);

/*** desaparece el cursor ****/
void cursoff(void)
{
union REGS reg;
 reg.h.ah=1;
 reg.h.ch=bios->cursTop| 0x20;   /* prende el bit 5*/
 reg.h.cl=bios->cursBottom;
 int86(0x10,&reg,&reg);
}

 /***** aparece el cursor ********/
 void curson(void)
 {
 union REGS reg;
  reg.h.ah=1;
  reg.h.ch=bios->cursTop & 0xDF;
  reg.h.cl=bios->cursBottom;
  int86(0x10,&reg,&reg);
}

/**** cambia la forma del cursor ****/
void cursshape(int top, int bottom)
{
union REGS reg;
 reg.h.ch=top;
 reg.h.cl=bottom;
 reg.h.ah=1;
 int86(0x10,&reg,&reg);
}
```

```
/*********************************************************************/
/*****                        EXPI488.C                          *****/
/*****    Función que obtiene información para la construcción de curvas  *****/
/*****    de extrapolación.                                       *****/
/*********************************************************************/
#include <math.h>
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#include <alloc.h>
#include <dos.h>
#include "a:\ventana\textbox.h"
#include "a:\ventana\popup.h"
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\constant.h"
/*************************************************/
/*****              variables globales           *****/
/*************************************************/
extern int inicio;
extern char *rango_electrometro,
            *funcion_electrometro;
extern RADIACION extrapolacion;
extern AMBIENTE  ambiente;
extern USUARIO   usuario;
extern PROCESO   proceso;
extern OTRAPL    aplicacion;
extern float     afondo[];
extern FILE *fp;  /*** apuntador del archivo *****/
extern float  aambiente[3][600];
extern int referencia;
extern int polref;
extern unsigned int nambiente,
                    ndato,
                    nfondo,
                    ndistancia;
extern int  cuentadistancia;
extern float voltaje_pos[];
extern float voltaje_neg[];
/*********************************************************************/
/*****          declaracion de funciones utlizadas en esta funcion     *****/
/*********************************************************************/
 int   ajustar_distancia(int *distancia_operacion),
         inicializacion(void);
 void  toma_lecturas_extrapol(int *distancia_operacion,float adatovp[][30],float adatovn[][30]),
         termina(int *distancia_operacion),
         ellEEE488(USUARIO *usuario),
         cuando_medir(void),
         guarda_datos_fondo(void),
         toma_lecturas_extrapolI488(int *distancia_operacion,float adatovp[][30],
                                    float adatovn[][30]),
         medir_ambientel488(void),
         guarda_datos(float adatovp[][30],float adatovn[][30]);
 void  guarda_voltaje(void);
```

```
/*************************************************************************/
/*****                    funcion para obtener curvas de extrapolacion                    *****/
/*************************************************************************/

void  curvas_de_extrapolacion(void)

{

POPUP  archivo={6,8,57,10,1,WHITE,BLACK,MAGENTA,WHITE,BLACK},
       parametros={1,7,80,21,2,MAGENTA,BLACK,WHITE,YELLOW,BLACK},
       electrometro={11,8,70,23,2,BLACK,WHITE,WHITE,BLACK,BLUE};
POPUP  comunicacion={5,7,76,21,2,YELLOW,CYAN,MAGENTA,RED,BLACK};
POPUP  aviso_distancia_inicial={20,8,61,10,2,WHITE,YELLOW,BLUE,RED,WHITE};

USUARIO usuario;


int   barrido,
      valida_ruido_de_fondo,
      i,j,
      distancia_operacion;
char  *dato_correcto="los datos anteriores son correctos? (S o N)",
      correccion,
      nombre_archivo[40],
      archo[45];
float  auxf;
float  adatovp[20][30],
      adatovn[20][30];
/*********************************************************/
extrapolacion.func=1;
window(1,1,80,25);

/************ PRESENTACIÓN DE PANTALLA INICIAL *************/
textbackground(CYAN);
textcolor(WHITE);
clrscr();
cmri();
/******** PRESENTACIÓN DELA PANTALLA DE ADQUISICIÓN DE DATOS *******/
popShow(&archivo);
window(archivo.left+11,archivo.top,archivo.right-12,archivo.top);
textcolor(archivo.normal);
textbackground(archivo.normback);
clrscr();
gotoxy(1,1);

/***** PETICIÓN DEL NOMBRE DEL ARCHIVO DE TRABAJO Y APERTURA    *******/
/***** DEL MISMO                                                *******/

printf("NOMBRE DEL ARCHIVO DE TRABAJO");
window(archivo.left+1,archivo.top+1,archivo.right-1,archivo.bottom-1);
textbackground(archivo.normback);
textcolor(archivo.normal);
clrscr();
```

```
gotoxy(3,2);
/*while(getchar() != '\n');*/
nombre_archivo[0]='\0';
gets(nombre_archivo);
strcat(nombre_archivo,".DAT");
fp=fopen (nombre_archivo,"wb");
popErase(&archivo);

/****** VENTANA DE INFORMACIÓN DE LOS DATOS DEL PROCESO Y PETICIÓN******/
/****** DE LA INFORMACIÓN NECESARIA PARA EL PROCESO                  ******/
extrapolacion.func=1;
popShow(&parametros);
window(parametros.left+1,parametros.top+1,parametros.right-1,
        parametros.bottom-1);
textcolor(parametros.normal);
textbackground(parametros.normback);
clrscr();
gotoxy(28,1);
puts("PARAMETROS DEL PROCESO");
gotoxy(4,3);
puts("NOMBRE DEL USUARIO:");
gotoxy(4,4);
puts("TIPO DE FUENTE:");
gotoxy(4,5);
puts("DISTANCIA INICIAL:");
gotoxy(69,5);
puts("micras");
gotoxy(4,6);
puts("DISTANCIA FINAL:");
gotoxy(69,6);
puts("micras");
gotoxy(4,7);
puts("PASO DEL DESPLAZAMIENTO");
gotoxy(69.7);
puts("micras");
gotoxy(4,8);
puts("GRADIENTE:");
gotoxy(69,8);
puts("V/cm");
gotoxy(4,9);
puts("NUMERO DE LECTURAS POR PASO:");
gotoxy(4,10);
puts("TIEMPO ENTRE LECTURAS:");
gotoxy(69,10);
puts("segundos");
gotoxy(4,11);
puts("FUNCION DEL ELECTROMETRO:");
gotoxy(4,12);
puts("RANGO");

 /***** SE PIDEN LOS DATOS NECESARIOS ******/
do{
 window(38,10,79,11);
 textcolor(parametros.normal);
```

```
textbackground(parametros.normback);
clrscr();
gotoxy(1,1);
gets(usuario.nombre);
gotoxy(1,2);
gets(usuario.fuente);
window(38,12,66,20);
textcolor(parametros.normal);
textbackground(parametros.normback);
clrscr();
/***** LEYENDO LA DISTANCIA INICIAL ********/
do
{
 gotoxy(1,1);
 clreol();
 scanf("%d",&extrapolacion.di);
 if(extrapolacion.di>inicio||extrapolacion.di<100)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  if(extrapolacion.di>inicio)
       cprintf("LA DISTANCIA MAXIMA ES DE %d MICRAS!",inicio);
  else
       cprintf("LA DISTANCIA MINIMA ES DE 100 MICRAS!");
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
  printf("ERROR");
  delay(1000);
  popErase(&aviso_distancia_inicial);
 }
}while(extrapolacion.di>inicio||extrapolacion.di<100);
/******* LEYENDO DISTANCIA FINAL **********/
do
{
 gotoxy(1,2);
 clreol();
 scanf("%d",&extrapolacion.df);
 if(extrapolacion.df>inicio||extrapolacion.df<100)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
```

```
   if(extrapolacion.df>inicio)
        cprintf("LA DISTANCIA MAXIMA ES DE %d MICRAS!",inicio);
   else
        cprintf("LA DISTANCIA MINIMA ES DE 100 MICRAS!");
   window(37,8,41,8);
   textcolor(aviso_distancia_inicial.border);
   textbackground(aviso_distancia_inicial.normback);
   clrscr();
   gotoxy(1,1);
   printf("ERROR");
   delay(1000);
   popErase(&aviso_distancia_inicial);
  }
}while(extrapolacion.df>inicio||extrapolacion.df<100);
/**********  LEYENDO EL PASO  *************/
do
{
 gotoxy(1,3);
 clreol();
 scanf("%d",&extrapolacion.paso);
 if(extrapolacion.paso>abs(extrapolacion.di-extrapolacion.df))
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  cprintf("EL PASO MAXIMO ES DE %d",abs(extrapolacion.di-extrapolacion.df));
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
  printf("ERROR");
  delay(1000);
  popErase(&aviso_distancia_inicial);
 }
}while(extrapolacion.paso>abs(extrapolacion.di-extrapolacion.df));
/*********  LEYENDO EL GRADIENTE  ********/
do
{
 gotoxy(1,4);
 clreol();
 scanf("%f",&extrapolacion.gradiente);
 if(extrapolacion.gradiente>500)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
```

```
  gotoxy(1,2);
  cprintf("EL GRADIENTE MAXIMO ES DE 500 V/cm");
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
  printf("ERROR");
  delay(1000);
  popErase(&aviso_distancia_inicial);
 }
}while(extrapolacion.gradiente>500.0);
/********** LEYENDO NUMERO DE LECTURAS *******/
do
{
 gotoxy(1,5);
 clreol();
 scanf("%d",&extrapolacion.lecturas);
 if(extrapolacion.lecturas>30)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  cprintf("EL NO. DE LECTURAS MAX. ES 30");
  window(37,8,41,8);
  textcolor(aviso_distancia_inicial.border);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,1);
  printf("ERROR");
  delay(1000);
  popErase(&aviso_distancia_inicial);
 }
}while(extrapolacion.lecturas>30);
/********** LEYENDO TIEMPO *********/
do
{
 gotoxy(1,6);
 clreol();
 scanf("%d",&extrapolacion.tiempo);
 if(extrapolacion.tiempo>65535L)
 {
  popShow(&aviso_distancia_inicial);
  window(aviso_distancia_inicial.left+1,aviso_distancia_inicial.top+1,
  aviso_distancia_inicial.right-1,aviso_distancia_inicial.bottom-1);
  textcolor(aviso_distancia_inicial.normal);
  textbackground(aviso_distancia_inicial.normback);
  clrscr();
  gotoxy(1,2);
  cprintf("EL TIEMPO MAX. ES DE 65535 S.");
```

```
    window(37,8,41,8);
    textcolor(aviso_distancia_inicial.border);
    textbackground(aviso_distancia_inicial.normback);
    clrscr();
    gotoxy(1,1);
    printf("ERROR");
    delay(1000);
    popErase(&aviso_distancia_inicial);
   }
 }while(extrapolacion.tiempo>65535L);
 /******* FIJANDO RANGO Y FUNCION DEL ELECTROMETRO *****/
 popShow(&electrometro);
 window(electrometro.left+1,electrometro.top+1,electrometro.right-1,
        electrometro.bottom-1);
 textcolor(electrometro.normal);
 textbackground(electrometro.normback);
 clrscr();
 gotoxy(2,5);
 elIEEE488(&usuario);
 /*****************************************/
 cprintf(" %d",usuario.funcion);
 getch();
 /*****************************************/
 gotoxy(32,5);
 popErase(&electrometro);
 gotoxy(1,7);
 puts(funcion_electrometro);
 gotoxy(1,8);
 puts(rango_electrometro);

 window(2,20,79,20);
 textcolor(parametros.normal);
 textbackground(parametros.normback);
 clrscr();
 gotoxy(4,1);
 puts(dato_correcto);
 correccion=toupper(getch());
 while(getchar() != '\n');
 window(2,20,79,20);
 textcolor(parametros.normal);
 textbackground(parametros.normback);
 clrscr();
}while(correccion=='N');
cuando_medir();
if(extrapolacion.di>extrapolacion.df)
 ndistancia=((extrapolacion.di-extrapolacion.df)/extrapolacion.paso)+1;
else
 {if (extrapolacion.di==extrapolacion.df)
  ndistancia=1;

 else

  ndistancia=((extrapolacion.df-extrapolacion.di)/extrapolacion.paso)+1;
 }
```

```
if(extrapolacion.amb==1)

 nambiente=extrapolacion.lecturas*2;
else
 nambiente=ndistancia;
ndato=extrapolacion.lecturas+2;
nfondo=extrapolacion.lecfondo+2;
/*************************************************************************/
fwrite(&usuario,sizeof(USUARIO),1,fp);

/****** rutina para la inicializacion del sistema ********************/
distancia_operacion=extrapolacion.di;
valida_ruido_de_fondo = inicializacion();
fwrite(&extrapolacion,sizeof(RADIACION),1,fp);
fwrite(&ambiente,sizeof(AMBIENTE),1,fp);
guarda_datos_fondo();
/*************************************************************************/
/********** SE ESCRIBE LA INFORMACIÓN DE LOS PARÁMETROS  **********/
/********** INICIALES EN EL ARHIVO                       **********/
/*************************************************************************/

/******** apertura de ventana para adquisicion de informacion ********/
popShow(&comunicacion);
 window(comunicacion.left+1,comunicacion.top+1,comunicacion.right-1,
 comunicacion.bottom-1);
 textcolor(comunicacion.normal);
 textbackground(comunicacion.normback);
 clrscr();

if(valida_ruido_de_fondo == 1)
{
 /********** RUTINA PARA LA ADQUSICIÓN DE INFORMACIÓN **********/
 cuentadistancia=0;
 do
 {
  popShow(&comunicacion);
  window(comunicacion.left+1,comunicacion.top+1,comunicacion.right-1,
  comunicacion.bottom-1);
  textcolor(comunicacion.normal);
  textbackground(comunicacion.normback);
  clrscr();
  gotoxy(13,1);
  cprintf("ADQUIRIENDO INFORMACION PARA LA CONSTRUCCION");
  gotoxy(22,2);
  cprintf("DE CURVAS DE EXTRAPOLACION");
  toma_lecturas_extrapol1488(&distancia_operacion,adatovp,adatovn);
  if(distancia_operacion == extrapolacion.df)
   barrido=0;
  else
  {
   barrido=ajustar_distancia(&distancia_operacion);
   if(extrapolacion.amb == 0)
   {
    medir_ambiente1488 ();
```

```
    aambiente[TEMPERATURA][cuentadistancia]=ambiente.temperatura;
    aambiente[PRESION][cuentadistancia]=ambiente.presion;
    aambiente[HUMEDAD][cuentadistancia]=ambiente.humedad;
   }
  }
  ++cuentadistancia;
 }while(barrido);
 gotoxy(24,8);
 cprintf("ESPERE POR FAVOR ...");
}
else
{
 gotoxy(20,7);
 cprintf("EL RUIDO DE FONDO ES EXCESIVO!");
 gotoxy(16,8);
 cprintf("EL PROCESO DE MEDICION SE HA ABORTADO");

}

/**************** GUARDANDO INFORMACIÓN EN EL ARCHIVO  **************/

gotoxy(11,12);
cprintf("SE ESTA GUARDANDO LA INFORMACION EN EL ARCHIVO:");
gotoxy(14,13);
cprintf("%s",nombre_archivo);

/*************************************************************************/
if(extrapolacion.amb==0 && ndistancia>1)
{
 for (l=0;i<3;i++)
  {
   for(j=0;j<ndistancia-1;j++)
   {
    auxf=aambiente[i][j];
    fwrite(&auxf,sizeof(float),1,fp);
   }
  }
}

guarda_datos(adatovp,adatovn);

guarda_voltaje();

/*************************************************************************/

fclose(fp);
popErase(&comunicacion);
/************ ajustando la fuente y la distancia entre placas a cond ******/
/************ iniciales voltaje= 0 V  y distancia= 25000 micras     ******/
popShow(&comunicacion);
window(comunicacion.left+1,comunicacion.top+1,comunicacion.right-1,
    comunicacion.bottom-1);
textcolor(comunicacion.normal);
textbackground(comunicacion.normback);
```

```
clrscr();
gotoxy(19,1);
cprintf("AJUSTANDO A CONDICIONES INICIALES");
termina(&distancia_operacion);
gotoxy(1,1);
clreol();
gotoxy(24,7);
cprintf("RUTINA DE FINALIZACION CONCLUIDA !");
delay(2000);
popErase(&comunicacion);
}
/************** FUNCIÓN PARA GURDAR DATOS EN EL ARCHIVO ***************/
void guarda_datos(float adatovp[][30] ,float adatovn[][30])

{
int i,
   j;
float auxd;


/********** ESCRIBIENDO EN EL ARCHIVO DATOS CON VP ************/
for(i=0;i<ndistancia;i++)
{
 for(j=0;j<extrapolacion.lecturas+2;j++)
 {
  auxd=adatovp[i][j];
  fwrite(&auxd,sizeof(float),1,fp);
 }
}
/******** ESCRIBIENDO EN EL ARCHIVO DATOS CON VN **************/
for(i=0;i<ndistancia;i++)
{
 for(j=0;j<extrapolacion.lecturas+2;j++)
 {
  auxd=adatovn[i][j];
  fwrite(&auxd,sizeof(float),1,fp);
 }
}
}

/*************************************************************************/
/*************************************************************************/
void guarda_datos_fondo()

{
int i;
float aux;
for(i=0;i<extrapolacion.lecfondo+2;i++)
 {
  aux=afondo[i];
  fwrite(&aux,sizeof(float),1,fp);
 }
}
```

```
/********* FUNCIÓN CUANDO MEDIR ********/

void cuando_medir()
{
POPUP  factores_amb={18,11,63,16,1,LIGHTRED,BLACK,LIGHTGRAY,CYAN,BLACK};
/******* el usuario decide cada cuando va a medir factores ambientales ***/
   popShow(&factores_amb);
   window(factores_amb.left+1,factores_amb.top+1,
        factores_amb.right-1,factores_amb.bottom-1);
   textcolor(factores_amb.normal);
   textbackground(factores_amb.normback);

   do{
        clrscr();
        gotoxy(2,1);
        cprintf("LA INFORMACION DE LOS FACTORES AMBIENTALES");
        gotoxy(2,2);
        cprintf("DESEA QUE SE MIDA:");
        gotoxy(2,3);
        cprintf("A) CADA VEZ QUE SE TOME UNA LECTURA");
        gotoxy(2,4);
        cprintf("B) CADA VEZ QUE SE AJUSTE LA DISTANCIA");

        if(toupper(getch()) =='A')
        {
          extrapolacion.amb=1;
          clrscr();
          gotoxy(2,1);
          cprintf("los factores ambientales se van a medir");
          gotoxy(2,2);
          cprintf("cada vez que se tome una lectura");
          gotoxy(2,3);
          cprintf("  confirme la opcion (s/n)");
        }
        else
        {
          extrapolacion.amb=0;
          clrscr();
          gotoxy(2,1);
          cprintf("los factores ambientales se van a medir");
          gotoxy(2,2);
          cprintf("cada vez que se ajuste la distancia");
          gotoxy(2,3);
          cprintf("  confirme la opcion (s/n)");
        }
   }while(toupper(getch()) =='N');
   popErase(&factores_amb);
}
void guarda_voltaje()
{
int  i;
float auxf;
for(i=0;i<ndistancia;++i)
{
```

```
 auxf=voltaje_pos[i];
 fwrite(&auxf,sizeof(float),1,fp);
 }
 for(i=0;i<ndistancia;++i)
 {
 auxf=voltaje_neg[i];
 fwrite(&auxf,sizeof(float),1,fp);
 }
}
```

```
/*****************************************************************************
****                          EXTI488.C                                    ****
****                funcion para ajustar el voltaje de la fuente           ****
****                utilizando la funcion FUENDRV.C                        ****
*****************************************************************************/
#include "a:\ventana\ieeeio.h"
#include <stdio.h>
#include <time.h>
#include <conio.h>
#include <dos.h>
#include <math.h>
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\popup.h"
#include "a:\ventana\fuendrv.h"
#include "a:\ventana\constant.h"
extern RADIACION extrapolacion;
extern AMBIENTE ambiente;
extern FILE *fp;
extern int cuentadistancia;
extern float ruido_fondo[];
extern float referencia;
extern int polref;
extern float aambiente[3][600];
extern int bandera;
extern float voltaje_pos[];
extern float voltaje_neg[];
extern int ndistancia;

/********* declaracion de funciones utilizadas en esta funcion ********/
void start(void),
     stop(void);
void tomar_lecturas(int *,float *,int ,float [][30]),
                    estadistica(float adato[][30]);
int ieeescnf(char *format, ...);
void medir_ambienteI488(void);
float mide_voltaje(void);

/*****************************************************************************/
void toma_lecturas_extrapolI488(distancia_operacion,adatovp,adatovn)
int *distancia_operacion;
float adatovp[][30];
float adatovn[][30];
{

 float voltaje,
       incremento;

 int polvolt,
     retardo;
 polvolt=POSITIVO;
 incremento=0.1;
 retardo=300;
 /************* ABRIENDO VENTANA DE ADQUISICION DE INFORMACION ***********/
```

```
/************  PARA CONSTRUIR CURVAS DE EXTRAPOLACION  ***********/

/*****************************************************************************/
voltaje=(double)(extrapolacion.gradiente * *distancia_operacion / 10000.0);
gotoxy(14,3);
clreol();
cprintf("AJUSTANDO EL VOLTAJE A: %lg V",voltaje);
if(bandera == 0)
 fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
bandera=0;
gotoxy(14,3);
clreol();
gotoxy(16,3);
voltaje_pos[cuentadistancia]=mide_voltaje();
cprintf("VOLTAJE AJUSTADO A: %f V",voltaje_pos[cuentadistancia]);
gotoxy(16,3);
clreol();
/***** tomar lecturas del electrometro *******/

tomar_lecturas(distancia_operacion,&voltaje,POSITIVO,adatovp);
/**** obteniendo la media y la desviacion standard para voltaje positivo****/
estadistica(adatovp);

/***** invertir la polaridad del voltaje *******/
polvolt=NEGATIVO;
gotoxy(14,3);
clreol();
cprintf("AJUSTANDO EL VOLTAJE A: -%lg V",voltaje);
fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
gotoxy(14,3);
clreol();
gotoxy(16,3);
voltaje_neg[cuentadistancia] = mide_voltaje();
cprintf("VOLTAJE AJUSTADO A: -%lg V",voltaje_neg[cuentadistancia]);
gotoxy(16,3);
clreol();
/******  tomar lecturas del electrometro con voltaje negativo ******/

tomar_lecturas(distancia_operacion,&voltaje,NEGATIVO,adatovn);

/**** obteniendo la media y la desviacion standard par voltaje neg. ***/
estadistica(adatovn);
/****** indicaciones al usuario de que el proceso de medicion ******/
/****** se a terminado                                  ******/
gotoxy(10,5);
cprintf("PARA LA DISTANCIA DE %d micras",*distancia_operacion);
gotoxy(12,6);
cprintf("SE TOMARON %d LECTURAS CON %lg V",extrapolacion.lecturas,voltaje);
gotoxy(12,7);
cprintf("SE TOMARON %d LECTURAS CON -%lg V",extrapolacion.lecturas,voltaje);
gotoxy(12,8);
gotoxy(1,5);
clreol();
gotoxy(1,6);
```

```
clreol();
gotoxy(1,7);
clreol();
gotoxy(1,8);
clreol();
}

/*****************************************************************************
*****       funcion para la adquisicion de informacion     ************
*****       del electrometro para enviada a un archivo     ************
*****************************************************************************/
void tomar_lecturas(distancia_operacion,voltaje,polaridad,adato)
int   *distancia_operacion;
float *voltaje;
int   polaridad;
float adato[][30];
{
 POPUP aviso_lecturas={16,12,65,20,2,BLACK,WHITE,WHITE,BLUE,MAGENTA};
 int contador=0;
 unsigned int tiempo,tiempo_ejec,i;
 float dato;
 float auxf;
 clock_t inicio,fin;
 popShow(&aviso_lecturas);
 window(aviso_lecturas.left+1,aviso_lecturas.top+1,
        aviso_lecturas.right-1,aviso_lecturas.bottom-1);
 gotoxy(8,1);
 cprintf(" T O M A N D O   I N F O R M A C I O N");
 gotoxy(2,2);
 cprintf("DISTANCIA ENTRE PLACAS: %d micras", *distancia_operacion);
 gotoxy(2,3);
 if(polaridad==POSITIVO)
        cprintf("VOLTAJE APLICADO :    %lg V", *voltaje);
 else
        cprintf("VOLTAJE APLICADO :   -%lg V", *voltaje);
 gotoxy(2,4);
 cprintf("NUMERO DE LECTURAS = %d",extrapolacion.lecturas);
 gotoxy(2,5);
 cprintf("SE HAN TOMADO 0");
 /*tiempo=(1000*extrapolacion.tiempo);*/

 /************** se habre obturador ****************************/
 tiempo_ejec=0;
 start();
 /************ se toman lecturas    ****************************/

 /*delay(1000-55);*/
 /*if(extrapolacion.tiempo>1)
 {
  for(i=0;i<extrapolacion.tiempo-1;++i)
    delay(1000);
 }*/
 for(contador=0;contador<extrapolacion.lecturas;++contador)
 {
```

```
  delay(1000-tiempo_ejec);
  if(extrapolacion.tiempo>1)
  {
   for(i=0;i<extrapolacion.tiempo-1;++i)
        delay(1000);
  }
  inicio=clock();
  ieeewt("enter 27\n");
  ieecscnf("%*4s%e",&auxf);
  dato=(float)(auxf);
  adato[cuentadistancia][contador]=dato;
  gotoxy(15,5);
  cprintf(" %d",contador+1);
  /**** tomando informacion del electrometro ******/
  if(extrapolacion.amb == 1)
  {
        medir_ambiente1488();
        /*** guardando informacion en el archivo *******/
        fwrite(&ambiente,sizeof(AMBIENTE),1,fp);
  }
  fin=clock();
  tiempo_ejec=(unsigned int) (1000*(fin-inicio)/CLK_TCK);
  /*gotoxy(4,6);
  cprintf("tiempo de ejecusion %d",tiempo_ejec);
  getche();*/
 }

 /*************** se cierra obturador ***********************/
 stop();
 /*********************************************************************/
 popErase(&aviso_lecturas);
}
/*************************************************************************/
/******* funcion para obtener la media y la desviacion standard ******/
/*************************************************************************/
void estadistica(float adato[][30])
{
 float media,
        s=0.0,
        desv_standard=0.0;
 int  contador_estadistico;
 media=0;
 for(contador_estadistico=0;contador_estadistico<extrapolacion.lecturas;
   contador_estadistico++)
 media+=adato[cuentadistancia][contador_estadistico];
 media/=extrapolacion.lecturas;
 adato[cuentadistancia][extrapolacion.lecturas]=media;
 /*** desviacion standard ****/
 for(contador_estadistico=0;contador_estadistico<extrapolacion.lecturas;
   contador_estadistico++)
 {
  s= (adato[cuentadistancia][contador_estadistico] - media)*
   (adato[cuentadistancia][contador_estadistico] - media);
 desv_standard += s;
```

```
}

desv_standard /= extrapolacion.lecturas;
desv_standard = sqrt(desv_standard);
adato[cuentadistancia][extrapolacion.lecturas+1]=desv_standard;
}
float mide_voltaje(void)
{
float auxf;
int  voltaje_medido;
/*ieeewt("output 14;A0\n");
ieeewt("output 14;C0\n");
ieeewt("output 14;R#0,3\n");
ieeewt("output 14;G4\n");
ieeewt("output 14;T6\n");
ieeewt("output 14;X\n");
ieeewt("enter 14\n");
ieeescnf("%d",&voltaje_medido);
auxf=(float)(voltaje_medido/30.0);*/
auxf=5.0;
return auxf;
}
void start(void)
{
ieeewt("output 9; B39X\n");
delay(50);
ieeewt("output 9; A39X\n");
}
void stop(void)
{
ieeewt("output 9; B40X\n");
delay(50);
ieeewt("output 9; A40X\n" );
}
```

```
/*****************************************************************************
**********                        FUENDRV.C                          **********
**********          funcion que controla  la fuente de alto voltaje  **********
**********          ,el programa principal envia la siguiente informa **********
**********          cion:                                            **********
**********          *voltaje   voltaje que se desea aplicar          **********
**********          *referencia voltaje actual de la fuente          **********
**********          *polvolt   polaridad del voltaje deseado         **********
**********          *polref    polaridad del voltaje actual          **********
**********          *retardo   retardo en el cambio de voltaje       **********
**********          *incremento paso del incremento de voltaje       **********
*****************************************************************************/

#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include "a:\ventana\popup.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\fuendrv.h"
#include "a:\ventana\constant.h"
/********** VARIABLES GLOBALES  ************/
extern inicio;
extern unsigned int acumulador;

/*************** FUNCIONES PROTOTIPO ************ *********/

void  enviadato(float *);

/*************************************************************/
void  fuendrv(float *voltaje,int *polvolt,float *referencia,int *polref.
              float *incremento,int *retardo)
{
 POPUP fuente={10,12,70,20,2,LIGHTBLUE,BLACK,LIGHTCYAN,MAGENTA,LIGHTMAGENTA};
 /* cursoff();*/
 popShow(&fuente);
 window(fuente.left+1,fuente.top+1,fuente.right-1,fuente.bottom-1);
 textbackground(fuente.normback);
 textcolor(fuente.normal);
 clrscr();
 gotoxy(5,4);
 cprintf("VOLTAJE DE LA FUENTE:");
 if(*polref==POSITIVO)
 {
  gotoxy(27,4);
  clreol();
  cprintf("%lg V",*referencia);
 }
 else
 {
  gotoxy(27,4);
  clreol();
  cprintf("-%lg V",*referencia);
 }
 if(*referencia==0.0 && *polref==POSITIVO)
```

```
 }
 else
  {
  if(*polref==POSITIVO && *polvolt==NEGATIVO)
    {
    decrementa(referencia,incremento,retardo,polref,polvolt);
    *polref=NEGATIVO;
    incrementa(referencia,voltaje,incremento,retardo,polref,polvolt);
    }
  else
    {
    decrementa(referencia,incremento,retardo,polref,polvolt);
    *polref=POSITIVO;
    incrementa(referencia,voltaje,incremento,retardo,polref,polvolt);
    }
  }
 popErase(&fuente);
/* curson(); */
}

void incrementa(float *referencia,float *voltaje,float *incremento,
    int *retardo,int *polaridad, int *polvolt)
{
 enviadato(referencia);
 while(*referencia < *voltaje)
 {
  *incremento*=10;
  *referencia*=10;
  *referencia+=*incremento;
  enviadato(referencia);
  if(*referencia == 0.0 && *polvolt==POSITIVO)
  {
   acumulador &= 0xFB;
   outportb(0x302,acumulador);
  }
  else
  {
  if(*referencia == 0.0 && *polvolt==NEGATIVO)
  {
   acumulador |= 0x05;
   outportb(0x302,acumulador);
  }
  }
  *referencia/=10;
  *incremento/=10;
  gotoxy(27,4);
  clreol();
  if(*polaridad==POSITIVO)
  cprintf("%lg V",*referencia);
  else
  cprintf("-%lg V",*referencia);
  delay(*retardo);
 }
}
```

```
void decrementa(float *referencia,float *incremento,int *retardo,int *polaridad,int *polvolt)
{
 enviadato(referencia);
 while(*referencia> 0.0)
 {
  *incremento*=10;
  *referencia*=10;
  *referencia-=*incremento;
  enviadato(referencia);
  if(*referencia == 0.0 && *polvolt==NEGATIVO )
  {
   acumulador |= 0x05;
   outportb(0x0302,acumulador);
  }
  else
  {
   if(*referencia==0.0 && *polvolt == POSITIVO)
   {
    acumulador &= 0xFB;
    outportb(0x0302,acumulador);
   }
  }
  *referencia/=10;
  *incremento/=10;
  gotoxy(27,4);
  clreol();
  if(*polaridad==POSITIVO)
  cprintf("%lg V",*referencia);
  else
  cprintf("-%lg V",*referencia);
  delay(*retardo);
 }
}

void enviadato(float *voltaje)
{
 float
              vbase;
 unsigned int  centenas,
              decenas,
              unidades,
              decimas,
              puertoa,
              puertob;
 vbase= *voltaje;
 centenas=(unsigned int)(vbase/1000.0);
 vbase=vbase-(centenas*1000);
 decenas=(unsigned int)(vbase/100.0);
 vbase=vbase - (decenas*100);
 unidades=(unsigned int)(vbase/10.0);
 vbase=vbase-(unidades*10);
 decimas=(unsigned int)vbase;
 puertoa=decimas;
```

```
unidades<<=4;
puertoa|=unidades;
puertob=decenas;
centenas<<=4;
puertob|=centenas;
outportb(0x0300,puertoa);
outportb(0x0301,puertob);
puertob|=0x40;
outportb(0x0301,puertob);
delay(5);
puertob&=0x3F;
outportb(0x301,puertob);
}
```

```
/*****************************************************************************/
/*****                          IMPRIME1.C                                *****/
/*****          funcion para desplegar la informacion adquirida           *****/
/*****************************************************************************/
#include <stdio.h>
#include <conio.h>
#include <alloc.h>
#include <stdlib.h>
#include <ctype.h>
#include <alloc.h>
#include <dos.h>
#include <string.h>
#include "a:\ventana\popup.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\constant.h"

/*********** VARIABLES GLOBALES UTILIZADAS EN LA FUNCION **********/
extern PROCESO     proceso;
extern USUARIO     usuario;
extern RADIACION   extrapolacion;
extern AMBIENTE    ambiente;
extern float       afondo[];
extern OTRAPL      aplicacion;
extern int         nambiente;
extern int         ndato;
extern int         nfondo;
extern int         ndistancia;
extern float       voltaje_pos[];
extern float       voltaje_neg[];
FILE *archivo_txt;
/************ funciones prototipo ******************/
 void presentacion_en_pantalla (float *vp,float *vn,FILE *apuntador_archivo);
 void tabular(int n);
 void muestra_info(FILE *apuntador_archivo, char []);
void presentainformacion(void)
{
 int    cd;
 float  auxd;

 float     *adatovp;
 float  *adatovn;
 FILE   *apuntador_archivo;
 POPUP  pedir_archivo={11,3,70,5,1,LIGHTBLUE,CYAN,BLACK,LIGHTGRAY,LIGHTCYAN};
 char  archivo[40],
     archo[40];
 /*****************************************************************************/
 /********* IMPRESION DEL INFORME DE DATOS ADQUIRIDOS EN PANTALLA *********/
/* curson();*/
 window(1,1,80,25);
 textcolor(BLACK);
 textbackground(LIGHTGRAY);
 clrscr();
 /**** ventana para pedir nombre del archivo de trabajo **************/
```

```
popShow(&pedir_archivo);
window(27,3,54,3);
textcolor(pedir_archivo.border);
textbackground(pedir_archivo.normback);
clrscr();
printf("NOMBRE DEL ARCHIVO DE TRABAJO");

window(pedir_archivo.left+1,pedir_archivo.top+1,
pedir_archivo.right-1,pedir_archivo.bottom-1);
textcolor(pedir_archivo.border);
textbackground(pedir_archivo.normback);
clrscr();
/****** se lee el nombre del archivo y se abre **********/
/*while(getchar() !='\n');*/
gets(archivo);
gotoxy(40,1);
archo[0]='\0';
strcat(archo,archivo);
strcat(archo,".DAT");
if((apuntador_archivo=fopen(archo,"rb")) != NULL)
{

 popErase(&pedir_archivo);

 /*******se lee la informacion de cabecera del archivo ************/
 fread(&usuario,sizeof(USUARIO),1,apuntador_archivo);
 fread(&extrapolacion,sizeof(RADIACION),1,apuntador_archivo);
 fread(&ambiente,sizeof(AMBIENTE),1,apuntador_archivo);
 cd=0;
 for(cd=0;cd<extrapolacion.lecfondo+2;++cd)
 {
  fread(&auxd,sizeof(float),1,apuntador_archivo);
  afondo[cd]=auxd;
 }
 if(extrapolacion.func==0)
  fread(&proceso,sizeof(PROCESO),1,apuntador_archivo);
 /******** se determina el tama&o de los vectores ******************/
 if(extrapolacion.func==1)
 {
  if(extrapolacion.di>extrapolacion.df)
  ndistancia=((extrapolacion.di-extrapolacion.df)/extrapolacion.paso)+1;
  else
  {
        if(extrapolacion.di==extrapolacion.df)
         ndistancia=1;
        else
         ndistancia=((extrapolacion.df-extrapolacion.di)/extrapolacion.paso)+1;
  }
 if(extrapolacion.amb==1)
        nambiente=extrapolacion.lecturas*2*ndistancia;
 else
 {
  if(nambiente!=1)
 {
```

```
   nambiente=ndistancia;
  }
 }
}
else
{
 ndistancia=proceso.lecturas;
 if(extrapolacion.amb==1)
    nambiente=ndistancia*2;
 else
 {
  if(nambiente!=1)
    nambiente=ndistancia;
 }
}
/* adatovp=(float *)malloc((extrapolacion.lecturas+2)*ndistancia*sizeof(float));
adatovn=(float *)malloc((extrapolacion.lecturas+2)*ndistancia*sizeof(float));
if(adatovp == NULL)
{
 printf("no hay suficiente espacio en memoria");
 delay(1000);
 exit(1);
} */
muestra_info(apuntador_archivo,archivo);
}
else
{
 clrscr();
 cprintf("no se puede abrir el archivo; presione ENTER");
 getch();
}
}

/**********************lectura de la informacion del archivo ***********/
void muestra_info( FILE *apuntador_archivo, char archivo[])
{
float adatovp[12][20],
      adatovn[12][20];
int   cd,
      cont,
      fordistancia,
      contador,
      localidad,
      loop,
      i;
unsigned int distancia_op=0,
                        incdist;
char  seleccion;
char  arch_disco[40];
char  confirma;
float auxf;
float factamb[3][600];
float auxd,
      voltaje_aplicado;
```

```
/*************************************************************************/
if(extrapolacion.amb==0)
 contador=ndistancia-1;
else
 contador=ndistancia*extrapolacion.lecturas*2;

 /******* leyendo el bloque de lecturas ambiente ***************/
if(ndistancia>1 && extrapolacion.amb==0)
{
 for(cont=0;cont<3;cont++)
 {
  for(cd=0;cd<contador;cd++)
  {
   fread(&auxf,sizeof(float),1,apuntador_archivo);
   factamb[cont][cd]=auxf;
  }
 }
}
else
{
 if(extrapolacion.amb==1)
 {
  for(cd=0;cd<contador;cd++)
  {
   for(cont=0;cont<3;cont++)
   {
    fread(&auxf,sizeof(float),1,apuntador_archivo);
    factamb[cont][cd]=auxf;
   }
  }
 }
}
/******* leyendo el bloque de datos con voltaje pos ***********/
for(cd=0;cd<ndistancia;cd++)
{
 for(cont=0;cont<extrapolacion.lecturas+2;cont++)
 {
  fread(&auxd,sizeof(float),1,apuntador_archivo);
  adatovp[cd][cont]=auxd;
 }
}
/***** leyendo el bloque de datos con voltaje negativo ***********/
for(cd=0;cd<ndistancia;cd++)
{
 for(cont=0;cont<extrapolacion.lecturas+2;cont++)
 {
  fread(&auxd,sizeof(float),1,apuntador_archivo);
  adatovn[cd][cont]=auxd;
 }
}
/*************************************************************************/
 if(extrapolacion.func==0)
  fread(&aplicacion,sizeof(OTRAPL),1,apuntador_archivo);
```

```
/*********************************************************************/
/*********************************************************************/
/************** se lee informacio del voltaje aplicado ************/
for(cd=0;cd<ndistancia;cd++)
  fread(&voltaje_pos[cd],sizeof(float),1,apuntador_archivo);
for(cd=0;cd<ndistancia;cd++)
  fread(&voltaje_neg[cd],sizeof(float),1,apuntador_archivo);
/*********************************************************************/
gotoxy(2,2);
cprintf("NOMBRE DEL USUARIO: %s",usuario.nombre);
gotoxy(2,3);
cprintf("TIPO DE FUENTE:    %s",usuario.fuente);
window(1,5,80,25);
textcolor(BLACK);
textbackground(LIGHTGRAY);
clrscr();
gotoxy(2,1);

/******** impresion de la informacion de las mediciones del fondo *******/
cprintf("MEDICION DEL FONDO");
gotoxy(2,2);
cprintf("DISTANCIA ENTRE PLACAS: %05d micras",extrapolacion.di);
gotoxy(40,2);
voltaje_aplicado=(float)(extrapolacion.gradiente*extrapolacion.di/10000.0);
cprintf("VOLTAJE APLICADO: %07.3f V",voltaje_pos[0]);
gotoxy(2,3);
cprintf("presion %7.3f mbar  temperatura  %7.3f C  humedad %07.3f %",
        ambiente.presion,ambiente.temperatura,ambiente.humedad);
gotoxy(2,4);
if(usuario.funcion == 0)
 cprintf("promedio= %12.5le A      desv. std.= %12.5le A",afondo[extrapolacion.lecfondo],
  afondo[extrapolacion.lecfondo+1]);
else
 cprintf("promedio= %12.5le C      desv. std.= %12.5le C",afondo[extrapolacion.lecfondo],
  afondo[extrapolacion.lecfondo+1]);
gotoxy(18,5);
if(usuario.funcion == 0)
 cprintf("[A]");
else
 cprintf("[C]");
for(cont=0;cont<extrapolacion.lecfondo;cont++)
 {
  gotoxy(10,6+cont);
  cprintf(" %d      %12.5le",cont,afondo[cont]);
 }
 getche();
clrscr();
/********** imprime  los valores medidos con irradiacion **********/
if(extrapolacion.di>extrapolacion.df || extrapolacion.di ==extrapolacion.df)
  incdist=extrapolacion.paso;
else
        incdist= -1 * extrapolacion.paso;
if(extrapolacion.amb==1)
```

```
/**** si los factores ambientales se miden cada vez que se tome una lectura **/
{

 if(extrapolacion.func==1)

 /**** si se adquirieron datos en modo curvas de extrapolacion *******/
 {
        fordistancia=abs(extrapolacion.di-extrapolacion.df);
        if(incdist!=0)
        fordistancia/=incdist;
 distancia_op=extrapolacion.di;
 }
 else
 /*** si se adquirieron datos en modo otras aplicaciones *******/
 {
 fordistancia=proceso.lecturas;
 }

 /**** envio de informacion al monitor ******/
 for(cd=0;cd<=fordistancia;cd++)
 {
        gotoxy(1,1);
        cprintf("LECTURAS CON RADIACION");
 gotoxy(1,2);
 cprintf("distancia entre placas: %5d micras",distancia_op);
 gotoxy(41,2);
 /******** calculo del voltaje aplicado **********/
 if(extrapolacion.func==1)
  voltaje_aplicado=
    (float)(extrapolacion.gradiente * distancia_op/10000.0);
 else
  voltaje_aplicado=
    (float)(aplicacion.gradiente[cd] * aplicacion.distancia[cd]/10000.0);
 cprintf("voltaje: %07.3f V",voltaje_pos[cd]);
/*************************************************************************/
        gotoxy(1,3);
 cprintf("n  lectura            TEMPERATURA    PRESION   HUMEDAD");
 gotoxy(1,4);
 if(usuario.funcion==0)
  cprintf("    [A]               [C]        [mbar]    [%]");
 else
  cprintf("    [C]               [C]        [mbar]    [%]");

  /***  muestra informacion adquirida con voltaje positivo ****/

  for(cont=0;cont<extrapolacion.lecturas;cont++)
  {
         gotoxy(1,5+cont);
         localidad=(extrapolacion.lecturas*2*cd)+cont;
   cprintf("%2d   %13.5le       %7.3f        %7.3f    %7.3f ",
    cont,adatovp[cd][cont],factamb[TEMPERATURA][localidad],factamb[PRESION][localidad],
   factamb[HUMEDAD][localidad]);
  }
  gotoxy(1,6+cont);
```

```
 if(usuario.funcion==0)
  cprintf("valor promedio %13.5le A   desviacion standard  %13.5le A",
   adatovp[cd][extrapolacion.lecturas],adatovp[cd][extrapolacion.lecturas+1]);
 else
  cprintf("valor promedio %13.5le C   desviacion standard  %13.5le C",
   adatovp[cd][extrapolacion.lecturas],adatovp[cd][extrapolacion.lecturas+1]);
         getch();
         gotoxy(41,2);
 clreol();

 /**** muestra informacion adquirida con voltaje negativo ***********/
 cprintf("voltaje: %7.3f V",voltaje_neg[cd]);
 for(cont=0;cont<extrapolacion.lecturas;cont++)
 {
  gotoxy(1,5+cont);
  clreol();
  localidad=(extrapolacion.lecturas*cd*2)+extrapolacion.lecturas+cont;
  cprintf("%d   %13.5le     %7.3f     %7.3f   %7.3f ",
    cont,adatovn[cd][cont],factamb[TEMPERATURA][localidad],factamb[PRESION][localidad],
                 factamb[HUMEDAD][localidad]);
         }
 gotoxy(1,6+cont);
 if(usuario.funcion==0)
  cprintf("valor promedio %13.5le A   desviacion standard  %13.5le A",
          adatovn[cd][extrapolacion.lecturas],adatovn[cd][extrapolacion.lecturas+1]);
 else
  cprintf("valor promedio %13.5le C   desviacion standard  %13.5le C",
          adatovn[cd][extrapolacion.lecturas],adatovn[cd][extrapolacion.lecturas+1]);

 if(extrapolacion.func==1)
 {
  distancia_op-=incdist;
         getch();
         }
 clrscr();
 }


}

else
/*********** si los factores ambientales se miden cada vez que se cambie la distancia***/
{
 if(extrapolacion.func==1)
 {
  if(extrapolacion.di>=extrapolacion.df)
  {
   fordistancia=extrapolacion.di-extrapolacion.df;
   if(incdist!=0)
    fordistancia/=extrapolacion.paso;
  }
  else
  {
   fordistancia=extrapolacion.df-extrapolacion.di;
```

```
}
else
{
 fordistancia=proceso.lecturas-1;
}
 for(cd=0;cd<=fordistancia;cd++)
 {
 gotoxy(1,1);
 cprintf("LECTURAS CON RADIACION");
 gotoxy(1,2);
 if(extrapolacion.func==1)
 {
  cprintf("distancia entre placas: %05d micras",distancia_op);

          voltaje_aplicado=
  (float)(extrapolacion.gradiente*distancia_op/10000.0);
 }
 else
 {
  cprintf("distancia entre placas: %05d micras",aplicacion.distancia[cd]);
  voltaje_aplicado=
  (float)(aplicacion.gradiente[cd] * aplicacion.distancia[cd]/10000.0);
 }
 gotoxy(1,3);
 if(ndistancia==1)
 cprintf("presion: %7.3f mbar  temperatura %7.3f C  humedad %7.3f %",
        ambiente.presion,ambiente.temperatura,ambiente.humedad);
         else
 {
  if(cd==0)
   cprintf("presion: %7.3f mbar  temperatura %7.3f C  humedad %7.3f %",
   ambiente.presion,ambiente.temperatura,ambiente.humedad);
  else
   cprintf("presion: %7.3f mbar  temperatura %7.3f C  humedad %7.3f %",
   factamb[PRESION][cd-1],factamb[TEMPERATURA][cd-1],factamb[HUMEDAD][cd-1]);
 }
 gotoxy(1,4);
 cprintf("                        VOLTAJE APLICADO");
 gotoxy(1,5);
         cprintf("n                %7.3f V             %7.3f V",voltaje_pos[cd],voltaje_neg[cd]);
         gotoxy(1,6);
 if(usuario.funcion == 0)
  cprintf("                  [A]                 [A]");
 else
  cprintf("                  [C]                 [C]");
 for(cont=0;cont<extrapolacion.lecturas;cont++)
 {
  gotoxy(1,7+cont);
  cprintf("%d             %13.5le          %13.5le",cont,adatovp[cd][cont],adatovn[cd][cont]);
 }
 gotoxy(1,8+cont);
 cprintf("corriente promedio %13.5le          %13.5le ",
         adatovp[cd][extrapolacion.lecturas],adatovn[cd][extrapolacion.lecturas]);
```

```
        gotoxy(1,9+cont);
  cprintf("desv. std      %13.5le      %13.5le ",
  adatovp[cd][extrapolacion.lecturas+1],adatovn[cd][extrapolacion.lecturas+1]);
  getch();
  clrscr();
  if(extrapolacion.func==1)
   distancia_op-=incdist;
  }

}
gotoxy(2,20);
printf("desea imprimir los datos?");
/*************************************************************************/
/******          rutina para enviar la informacion a una impresora          ******/
/*************************************************************************/
confirma=toupper(getch());
if(confirma=='S')
{
 window(1,1,80,25);
 textcolor(MAGENTA);
 textbackground(CYAN);
 clrscr();
 loop=1;
 do
 {
  gotoxy(27,12);
  cprintf("SALIDA A:");
  gotoxy(27,13);
  cprintf("A IMPRESORA");
  gotoxy(27,14);
  cprintf("B ARCHIVO CON EXTENSION .ASC");
  seleccion=toupper(getch());
  if(seleccion=='A'|| seleccion=='B')
    loop=0;
 }while(loop);
 clrscr();
 if(seleccion=='A')
 {
  /*popShow(&aviso_impresion);*/
  window(20,12,61,14);
  textcolor(LIGHTGREEN);
  textbackground(BLACK);
  clrscr();
  gotoxy(2,2);
  cprintf("IMPRIMIENDO DATOS !!!");
  archivo_txt=stdprn;
 }
 else
 {
  arch_disco[0]='\0';
  strcat(arch_disco,archivo);
  strcat(arch_disco,".ASC");
  if((archivo_txt=fopen(arch_disco,"w"))==NULL)
   {
```

```
    puts("no se puede abrir el archivo\n");
    exit(1);
   }
   window(20,12,61,14);
   textcolor(YELLOW);
   textbackground(BLACK);
   clrscr();
   gotoxy(11,1);
   cprintf("ESCRIBIENDO ARCHIVO");
   gotoxy(2,3);
   cprintf("%s",arch_disco);
  }
  /***********determinando el tama&o de los vectores ****************/
 /*************************************************************************/
  if(extrapolacion.amb==1)
  {
   fputs("INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES\n",archivo_txt );
   fputs("CENTRO DE METROLOGIA DE RADIACIONES IONIZANTES\n",archivo_txt);
   fprintf(archivo_txt,"USUARIO: %s\n",usuario.nombre);
   fprintf(archivo_txt,"TIPO DE FUENTE: %s\n",usuario.fuente);
   fputs("MEDICION DEL RUIDO DE FONDO\n",archivo_txt);
          fprintf(archivo_txt,"%s %4.3f %s    %s %4.3f %s    %s %4.3f %s\r",
          "presion:",ambiente.presion,"mbar","humedad:",ambiente.humedad,
          "%","temperatura",ambiente.temperatura,"C");
   voltaje_aplicado= (float)(extrapolacion.gradiente*extrapolacion.di/10000.0);
   fprintf(archivo_txt,"%s %5d","distancia entre placas:",extrapolacion.di);
   tabular(5);
   fprintf(archivo_txt,"%s %7.3lf %c\n","voltaje aplicado :",voltaje_pos[0],'V');
   tabular(33);
   fprintf(archivo_txt,"%c      %s\n",'n',"lectura");
   tabular(42);
   if(usuario.funcion == 0)
    fputs("[A]\n",archivo_txt);
   else
          fputs("[C]\n",archivo_txt);
          for(i=0;i<extrapolacion.lecfondo;i++)
   {
    tabular(31);
    fprintf(archivo_txt,"%2d     %13.5le\n",i,afondo[i]);
   }
   tabular(10);
   if(usuario.funcion==0)
    fprintf(archivo_txt,"%s %13.5le %c  %s %13.5le %c\n","promedio:",
     afondo[extrapolacion.lecfondo],'A',"desv. std.:",afondo[extrapolacion.lecfondo+1],'A');
   fprintf(archivo_txt,"\n");
   distancia_op=extrapolacion.di;
   for(cd=0;cd<=fordistancia;cd++)
   {
    /**********  lecturas con voltaje positivo ************************/
    fputs("MEDICIONES CON IRRADIACION\n",archivo_txt);
    voltaje_aplicado= (float)(extrapolacion.gradiente*distancia_op/10000.0);
    fprintf(archivo_txt,"%s %5d","distancia entre placas:",distancia_op);
    tabular(5);
    fprintf(archivo_txt,"%s %7.3lf %c\n","voltaje aplicado :",voltaje_pos[cd],'V');
```

```
    fputs("n    LECTURA    PRESION    HUMEDAD  TEMPERATURA   HORA\n",archivo_txt);
    if(usuario.funcion==0)
     fputs("       [A]      [mbar]     [%]      [C]\n",archivo_txt);
    else
     fputs("       [A]      [mbar]     [%]      [C]\n",archivo_txt);
    for(i=0;i<extrapolacion.lecturas;i++)
    {
     localidad=(extrapolacion.lecturas*2*cd)+i;
     fprintf(archivo_txt,"%2d %13.5le   %7.3f   %7.3f   %7.3f\n",
i,adatovp[cd][i],factamb[PRESION][localidad],factamb[HUMEDAD][localidad],factamb[TEMPERATUR
A][localidad]);
    }
    if(usuario.funcion==0)
     fprintf(archivo_txt,"%s  %13.5le %c   %s  %13.5le %c\n",
      "promedio:",adatovp[cd][extrapolacion.lecturas],'A',"desv. std.:",
      adatovp[cd][extrapolacion.lecturas+1],'A');
    else
     fprintf(archivo_txt,"%s  %13.5le %c   %s  %13.5le %c\n",
                    "promedio:",adatovp[cd][extrapolacion.lecturas],'A',"desv. std.:",
                    adatovp[cd][extrapolacion.lecturas+1],'A');
    fprintf(archivo_txt,"\n");
    /*********  lecturas con voltaje negativo **********************/
    fputs("MEDICIONES CON IRRADIACION\n",archivo_txt);
    voltaje_aplicado= (float)(extrapolacion.gradiente*distancia_op/10000.0);
    fprintf(archivo_txt,"%s %5d","distancia entre placas:",distancia_op);
    tabular(5);
    fprintf(archivo_txt,"%s %7.3lf %c\n","voltaje aplicado :", voltaje_neg[cd],'V');
    fputs("n    LECTURA    PRESION    HUMEDAL  TEMPERATURA   HORA\n",archivo_txt);
    if(usuario.funcion==0)
     fputs("       [A]      [mbar]     [%]      [C]\n",archivo_txt);
    else
                    fputs("       [C]      [mbar]     [%]      [C]\n",archivo_txt);
    for(i=0;i<extrapolacion.lecturas;i++)
    {
     localidad=(extrapolacion.lecturas*2*cd)+extrapolacion.lecturas+i;
     fprintf(archivo_txt,"%2d %13.5le   %6.3f   %6.3f   %6.3f\n",

i,adatovn[cd][i],factamb[PRESION][localidad],factamb[HUMEDAD][localidad],factamb[TEMPERATUR
A][localidad]);

    }
    if(usuario.funcion==0)
    fprintf(archivo_txt,"%s  %13.5le %c  %s  %13.5le %c\n",
      "promedio:",adatovn[cd][extrapolacion.lecturas],'A',"desv. std.:",
      adatovn[cd][extrapolacion.lecturas+1],'A');
    else
            fprintf(archivo_txt,"%s  %13.5le %c  %s  %13.5le %c\n",
                    "promedio:",adatovn[cd][extrapolacion.lecturas],'C',"desv. std.:",
     adatovn[cd][extrapolacion.lecturas+1],'C');
   fprintf(archivo_txt,"\n");
   distancia_op-=incdist;
   }
   fprintf(archivo_txt,"\n");
   fprintf(archivo_txt,"\n");
```

```
}
else
{
/****   impresion de datos cuando se miden factores ambientales **********/
/****   cada vez que se cambie la distancia                  **********/
       fputs("INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES\n",archivo_txt );
       fputs("CENTRO DE METROLOGIA DE RADIACIONES IONIZANTES\n",archivo_txt);
fprintf(archivo_txt,"USUARIO: %s\n",usuario.nombre);
fprintf(archivo_txt,"TIPO DE FUENTE: %s\n",usuario.fuente);
fputs("MEDICION DEL RUIDO DE FONDO\n",archivo_txt);
voltaje_aplicado= (float)(extrapolacion.gradiente*extrapolacion.di/10000.0);
fprintf(archivo_txt,"%s %5d","distancia entre placas:",extrapolacion.di);
fprintf(archivo_txt,"voltaje aplicado : %4.3f %c\n",voltaje_pos[0],'V');

fputs("          n      lectura\n",archivo_txt);
if(usuario.funcion==0)
 fprintf(archivo_txt,"                              %s\n","[A]");
else
       fprintf(archivo_txt,"                              %s\n","[C]");
       for(i=0;i<extrapolacion.lecfondo;i++)
{
 fprintf(archivo_txt,"                    %2d     %13.5le\n",i,afondo[i]);
}
if(usuario.funcion==0)
 fprintf(archivo_txt,"       %s %13.5le %c  %s %13.5le %c\n","promedio:",
 afondo[extrapolacion.lecfondo],'A',"desv. std.:",afondo[extrapolacion.lecfondo+1],'A');
else
 fprintf(archivo_txt,"       %s %13.5le %c  %s %13.5le %c\n","promedio:",
 afondo[extrapolacion.lecfondo],'C',"desv. std.:",ator.do[extrapolacion.lecfondo+1],'C');
fprintf(archivo_txt,"\n");
fprintf(archivo_txt,"\n");
distancia_op=extrapolacion.di;
for(cd=0;cd<=fordistancia;cd++)
{
  fprintf(archivo_txt,"%s\n","MEDICIONES CON IRRADIACION");
  if(extrapolacion.func==1)
  fprintf(archivo_txt,"%s  %5d %s\n","distancia entre placas:",distancia_op,"micras");
  else
  fprintf(archivo_txt,"%s  %5d %s\n",
  "distancia entre placas:",aplicacion.distancia[cd],"micras");
  if(cd==0)
  fprintf(archivo_txt,"%s %7.3f %s   %s %7.3f %s    %s %7.3f %s\n",
  "presion:",ambiente.presion,"mbar","humedad:",ambiente.humedad,
  "%","temperatura",ambiente.temperatura,"C");
  else
       fprintf(archivo_txt,"%s %7.3f %s   %s %7.3f %s    %s %7.3f %s\n",
       "presion:",factamb[PRESION][cd-1],"mbar","humedad:",factamb[HUMEDAD][cd-1],
       "%","temperatura",factamb[TEMPERATURA][cd-1],"C");
  if(extrapolacion.func==1)
       voltaje_aplicado= (float)(extrapolacion.gradiente*extrapolacion.di/10000.0);
  else
       voltaje_aplicado=
       (float)(aplicacion.gradiente[cd] * aplicacion.distancia[cd] /10000.0);
  fprintf(archivo_txt,"                %s\n","n              VOLTAJE APLICADO");
```

```
    fprintf(archivo_txt,"                    %7.3lf%c          %7.3lf%c\n",
          voltaje_pos[cd],'V',(-1)*voltaje_neg[cd],'V');
          if(usuario.funcion == 0)
           fprintf(archivo_txt,"                    %s          %s\n","[A]","[A]");
          else
           fprintf(archivo_txt,"                    %s          %s\n","[C]","[C]");
          for(i=0;i<extrapolacion.lecturas;i++)
          {
           fprintf(archivo_txt,"          %2d      %13.5le
%13.5le\n",i,adatovp[cd][i],adatovn[cd][i]);
          }
          fprintf(archivo_txt,"\n");
          fprintf(archivo_txt,"          %s      %13.5le      %13.5le\n","promedio:",
          adatovp[cd][extrapolacion.lecturas],adatovn[cd][extrapolacion.lecturas]);
          fprintf(archivo_txt,"          %s      %13.5le      %13.5le\n","desv.srtd :",
          adatovp[cd][extrapolacion.lecturas+1],adatovn[cd][extrapolacion.lecturas+1]);
          distancia_op-=incdist;
          fprintf(archivo_txt,"\n");
          fprintf(archivo_txt,"\n");

         }
         fprintf(archivo_txt,"\n");
         fprintf(archivo_txt,"\n");
 }

 /* popErase(&aviso_impresion); */
 }
 fclose(apuntador_archivo);
 fclose(archivo_txt);
 if(ferror(apuntador_archivo))
 cprintf("ERROR AL CERRAR EL ARCHIVO");
}
/*************** funcion tabular     ***********************/
void tabular(int n)
{
 int i;
 for(i=0;i< n;i++)
  fprintf(archivo_txt," ");
}
```

```
/*****************************************************************************
******                          INITI488.C                                  ******
******        FUNCION DE INICIALIZACION DE LA DISTANCIA ENTRE               ******
******        PLACAS                                                        ******
*****************************************************************************/
#include "a:\ventana\ieeeio.h"
#include <ctype.h>
#include <conio.h>
#include <stdio.h>
#include <dos.h>
#include <math.h>
#include "a:\ventana\popup.h"
#include "a:\ventana\cursor.h"
#include "a:\ventana\biosarea.h"
#include "a:\ventana\ventbeta.h"
#include "a:\ventana\fuendrv.h"
#include "a:\ventana\constant.h"
extern int inicio;
extern USUARIO  usuario;
extern RADIACION  extrapolacion;
extern AMBIENTE  ambiente;
extern FILE *fp;
extern float referencia;
extern int polref;
extern float afondo[];
extern float  *aambiente;
extern float *adatovp;
extern float *adatovn;
extern int bandera;

/********** DECLARACION DE FUNCIONES PROTOTIPO  ***************/

int   ieeescnf(char *format, ...);
void  medir_ambiente(void);
void  medir_ambienteI488(void), /** obtiene los valores de temp,pres,y humedad ***/
     tomar_lecturas_fondo(unsigned int *distancia_operacion,float *voltaje,int polaridad,
                          unsigned int fondo_tiempo),
     procesa_fondo(float *desv_std,float *promedio),
     drvmotor(int paso,int direccion,int distancia_inicial);
int inicializacion()
{
 POPUP aviso={10,6,70,21,2,YELLOW,WHITE,CYAN,BLACK,MAGENTA};
 int paso;

 unsigned int  difondo;
 unsigned int   fondo_tiempo;
 float voltaje=0,
        desv_std=0.0,
        incremento=0.1,
        promedio_ruido_de_fondo;

 int polvolt=POSITIVO,
    validacion,
    retardo=300;
```

```
/*cursoff();*/
 popShow(&aviso);
 window(aviso.left+1,aviso.top+1,aviso.right-1,aviso.bottom-1);
 textcolor(aviso.normal);
 textbackground(aviso.normback);
 clrscr();
 gotoxy(18,1);
 cprintf("RUTINA DE INICIAIZACION");
if(inicio != extrapolacion.di)
{
 gotoxy(16,2);
 cprintf("AJUSTANDO DISTANCIA INICIAL");
 gotoxy(20,3);
 cprintf("A %d MICRAS",extrapolacion.di);
 delay(2000);
 /*****************borrando linea 3 a 5 *************************/
 gotoxy(1,5);
 clreol();
 /********** operacion de ajuste de la distancia ******************/
 paso=inicio - extrapolacion.di;
 drvmotor(paso,ADELANTE,inicio);
 gotoxy(1,3);
 clreol();
 gotoxy(1,4);
 clreol();
 gotoxy(7,2);
 cprintf("DISTANCIA ENTRE PLACAS AJUSTADA A: %d micras",extrapolacion.di);
 delay(1500);
 gotoxy(1,2);
 clreol();
}
else
{
 gotoxy(15,2);
 cprintf("DISTANCIA INICIAL %d MICRAS",inicio);
 delay(1500);
 gotoxy(15,2);
 clreol();
 gotoxy(18,6);
 clreol();

}
/******** AJUSTANDO EL VOLTAJE DE POLARIZACION DE LAS PLACAS **********/
bandera=1;
voltaje=(float)(extrapolacion.gradiente * extrapolacion.di / 10000.0);
gotoxy(10,2);
cprintf("AJUSTANDO EL VOLTAJE A: %lg V",voltaje);
fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
gotoxy(1,2);
clreol();

/****** ADQUSICION DE INFORMACION  ************/
/*cursoff();*/
gotoxy(16,2);
```

```
cprintf("TOMANDO LECTURAS DE FONDO");
gotoxy(5,4);
cprintf("cuantas lecturas de medicion del fondo va ha tomar: ");
scanf("%d",&extrapolacion.lecfondo);
gotoxy(5,5);
cprintf("tiempo entre medicion y medicion: ");
scanf("%d",&fondo_tiempo);
/********* determinar el tama&o para acomodar el arreglo afondo *********/
/*****   midiendo factores ambientales  **********/
medir_ambienteI488();

/**** regresando a la ventana original **********/
popShow(&aviso);
window(aviso.left+1,aviso.top+1,aviso.right-1,aviso.bottom-1);
textcolor(aviso.normal);
textbackground(aviso.normback);
clrscr();

/******  tomando lecturas de fondo **********/

difondo=extrapolacion.di;
tomar_lecturas_fondo(&difondo,&voltaje,POSITIVO,fondo_tiempo);
procesa_fondo(&desv_std,&promedio_ruido_de_fondo);
if(usuario.funcion == 1)
{
 gotoxy(5,8);
 printf("RUIDO DE FONDO PROMEDIO: %lg [C]",promedio_ruido_de_fondo);
 gotoxy (5,9);
 cprintf("DESVIACION ESTANDAR: %lg [C]",desv_std);
}
else
{
 gotoxy(5,8);
 printf("RUIDO DE FONDO PROMEDIO: %lg [A]",promedio_ruido_de_fondo);
 gotoxy (5,9);
 cprintf("DESVIACION ESTANDAR: %lg [A]",desv_std);
}

/*****************************************************************/
/*********                se escribe en el archivo el promedio                *********/
/*****************************************************************/
/*********  se escribe la informacion de los parametros   *********/
/*********  iniciales en el archivo                       *********/
/*****************************************************************/
gotoxy(11,10);
printf("CONTINUO CON EL PROCESO DE MEDICION? (S/N)");

if(toupper(getch()) == 'S')
  validacion=1;
else
  validacion=0;
gotoxy(17,11);
clreol();
popErase(&aviso);
```

```
 /*curson();*/
 return(validacion);
}
/*****************************************************/
/******        adqusicicion de variables ambientales        ******/
/*****************************************************/
void medir_ambienteI488(void)
{
 int  temperatura,
    presion,
    humedad;
 /**********      medicion de variables ambientales    ***************/
  /***** midiendo presion *******/
 /* ieeewt("output 14;A0\n");
  ieeewt("output 14;C5\n");
  ieeewt("output 14;R#5,0\n");
  ieeewt("output 14;G4\n");
  ieeewt("output 14;T6\n");
  ieeewt("output 14;X\n");
  ieeewt("enter 14\n");
  ieeescnf("%d",&presion);  */
  /***** midiendo humedad ***
/*  ieeewt("output 14;A0\n");
  ieeewt("output 14;C6\n");
  ieeewt("output 14;R#6,0\n");
  ieeewt("output 14;G4\n");
  ieeewt("output 14;T6\n");
  ieeewt("output 14;X\n");
  ieeewt("enter 14\n");
  ieeescnf("%d",&humedad);*/

  /***** midiendo temperatura **********/
/*  ieeewt("output 14;A0\n");
  ieeewt("output 14;C7\n");
  ieeewt("output 14;R#7,1\n");
  ieeewt("output 14;G4\n");
  ieeewt("output 14;T6\n");
  ieeewt("output 14;X\n");
  ieeewt("enter 14\n");
  ieeescnf("%d",&temperatura);*/
  /****** ajustando valores a float *********/
 /* ambiente.presion=(float)(presion/30.0);
  ambiente.temperatura=(float)(temperatura/750.0);
  ambiente.humedad=(float)(humedad/30.0);*/
  /*** LOS TRES RENGLONES ANTERIORRES DEBEN ACTIVARSE CUANDO SE USE ***/
  /*** EL ADC Y DEBEN BORRARSE LOS TRES SIGUIENTES RENGLONES      ***/
  ambiente.presion=710.0;
  ambiente.temperatura=20.0;
  ambiente.humedad=45.0;
  /*****************************************************************/
  if (extrapolacion.func==0)
  {
   informacion_ambiente(&ambiente,SI);
   window(4,24,77,24);
```

```
  }
}

/******* se  procesa informacion de fondo *********/
void procesa_fondo(desv_std,promedio)

float *promedio;
float *desv_std;
{
POPUP aviso_fondo={20,12,60,18,2,WHITE,BLUE,BLUE,WHITE,MAGENTA};
int i;
popShow(&aviso_fondo);
window(aviso_fondo.left+1,aviso_fondo.top+1,aviso_fondo.right-1,
        aviso_fondo.bottom-1);
textcolor(aviso_fondo.normal);
textbackground(aviso_fondo.normback);
clrscr();
gotoxy(1,1);
cprintf("se procesa informacion del fondo ...!");
*promedio=0.0;
for(i=0; i<extrapolacion.lecfondo; i++)
*promedio +=afondo[i];
*promedio=(float)(*promedio/ extrapolacion.lecfondo);
*desv_std=0.0;
for(i=0;i<extrapolacion.lecfondo;i++)
*desv_std +=(afondo[i] - *promedio) * (afondo[i] - *promedio);
*desv_std /=extrapolacion.lecfondo;
*desv_std = sqrt(*desv_std);
afondo[extrapolacion.lecfondo]=*promedio;
afondo[extrapolacion.lecfondo+1]=*desv_std;
popErase(&aviso_fondo);
}

/************ funcion para regresar el sistema a condiciones iniciales ******/
void termina(int *distancia_operacion)
{
int    paso,
          polvolt=POSITIVO,
          retardo=300;
float  voltaje=0,
          incremento=0.1;
/****** declarando funciones utilizadas **********************/
void drvmotor(int paso,int direccion,int distancia_inicial);

/*****************************************************************/

gotoxy(22,2);
cprintf("AJUSTANDO EL VOLTAJE A: 0 V");
fuendrv(&voltaje,&polvolt,&referencia,&polref,&incremento,&retardo);
gotoxy(1,2);
clreol();
gotoxy(10,2);
cprintf("AJUSTANDO LA DISTANCIA ENTRE PLACAS A: %d micras",inicio);
paso=inicio - *distancia_operacion + 2 ;
```

```
drvmotor(paso,ATRAS,*distancia_operacion);
gotoxy(1,2);
clreol();
gotoxy(25,7);
cprintf("OPERACION CONCLUIDA");
}

/****** rutina para tomar lecturas de fondo *************************/

void tomar_lecturas_fondo(distancia_operacion,voltaje,polaridad,fondo_tiempo)
unsigned int   *distancia_operacion;
float *voltaje;
int   polaridad;

unsigned int   fondo_tiempo;
{
 POPUP aviso_lecturas={16,12,65,20,2,BLACK,WHITE,WHITE,BLUE,MAGENTA};
 int contador=0;
 float auxf;
 float dato;

 popShow(&aviso_lecturas);
 window(aviso_lecturas.left+1,aviso_lecturas.top+1,
        aviso_lecturas.right-1,aviso_lecturas.bottom-1);
 gotoxy(8,1);
 cprintf(" T O M A N D O   I N F O R M A C I O N");
 gotoxy(2,2);
 cprintf("DISTANCIA ENTRE PLACAS: %d micras", *distancia_operacion);
 gotoxy(2,3);
 if(polaridad==POSITIVO)
  cprintf("VOLTAJE APLICADO :    %lg V", *voltaje);
 else
  cprintf("VOLTAJE APLICADO :   -%lg V", *voltaje);
 for(contador=0;contador<extrapolacion.lecfondo; contador++)
 {
 gotoxy(2,4);
 cprintf("SE HAN TOMADO %d DE %d LECTURAS",contador,extrapolacion.lecfondo);
 /****  tomando informacion del electrometro ******/
 ieeewt("output 27;C0\n");
 ieeewt("enter 27\n");
 ieeescnf("%*4s%e",&auxf);
 dato=(float)(auxf);
 afondo[contador]=dato;
 gotoxy(20,6);
 clreol();
 gotoxy(20,5);
 clreol();
 fondo_tiempo=(unsigned int)(1000*extrapolacion.tiempo);
 delay(fondo_tiempo);
 }
 popErase(&aviso_lecturas);
}
```

```
/*****************************************************/
/*** POPUP.C:  biblioteca de funciones manejo de ventanas  ***/
/*****************************************************/
#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include <alloc.h>
#include <string.h>
#include "a:\ventana\textbox.h"
#include "a:\ventana\biosarea.h"
#include "a:\ventana\popup.h"

/********  funcion para crear ventanas   *********/
void popShow (POPUP *pop)
{
 int bufsize, x;
 gettextinfo(&(pop->prev)); /** guarda la pantalla inicial ***/
 window(1, 1, 80, 25);    /** activa la pantalla completa ***/
 /*** se asegura que los limites sean los correctos  **/
 if(pop->left > pop->right){
  x=pop->left;
  pop->left=pop->right;
  pop->right=x;
 }
 if(pop->top > pop->bottom){
  x=pop->top;
  pop->top=pop->bottom;
  pop->bottom=x;
 }

 /**** salva el area bajo la nueva ventana *****/
 bufsize=(pop->right - pop->left + 2)*   /** tama&o del bufer **/
          (pop->bottom - pop->top + 2)*2;
 pop->save=malloc(bufsize);              /*aparta espacio en memoria*/
 /*** salva la imagen inicial ****/
 gettext(pop->left,pop->top,pop->right,pop->bottom,pop->save);

 /*** dibuja los limites de la ventana  ***/
 textcolor(pop->border);         /*** color de  primer plano ***/
 textbackground(pop->normback);   /*** color de fondo ***/
 textbox(pop->left,pop->top,pop->right,pop->bottom,pop->style);

 /**** abre la ventana  ***/
 textcolor(pop->normal);   /**** color del texto ***/
             /** abre ventana **/
 window(pop->left+1,pop->top+1,pop->right-1,pop->bottom-1);
 clrscr();
 if(pop->text != NULL)         /**** escribe el texto  *****/
  cputs(pop->text);
}

/*** borra la ventana creada restableciendo la imagen cubierta ***/
void popErase (POPUP *pop)
{
```

```
/** se asegura que la ventana existe, si no, se sale ****/
if(pop->save == NULL)
 return;

/** restablece las caracteristicas de la ventana previa ***/
window(pop->prev.winleft,pop->prev.wintop,pop->prev.winright,
        pop->prev.winbottom);
gotoxy(pop->prev.curx,pop->prev.cury); /* define posicion del cursor */
textcolor(pop->prev.attribute&0x0F); /** color anterior */
textbackground(pop->prev.attribute>>4);

/** restablece area cubierta **/
puttext(pop->left,pop->top,pop->right,pop->bottom,pop->save);
free(pop->save);
pop->save=NULL;
}

void popCenter(POPUP *win, char *string)
{
int i, tab;
tab=(win->right - win->left - strlen(string))/2;
for(i=0; i<tab; i++)
 cputs(" ");
cputs(string);
}

/** reescribe una linea con nuevos atributos de caracter **/

void popRewrite(POPUP *win, int row, char attrib)
{
int  p, nchars;
union REGS reg;
struct text_info text;
gettextinfo(&text);          /* adquiere los para.netros actuales */
nchars=win->right - win->left;  /*  ancho de la ventana */
for(p=1; p<nchars; p++){
 gotoxy(p,row);      /* adquiere un cartacter */
 reg.h.ah=8;          /* en la pagina 0      */
 reg.h.bh=0;          /* via ROM BIOS */
 int86(0x10,&reg, &reg);   /* escribe el fondo     */
 reg.h.ah=9;          /* con atributo hilite  */
 reg.h.bl=attrib;     /* en la pagina 0       */
 reg.h.bh=0;          /* un caracter          */
 reg.x.cx=1;
 int86(0x10,&reg,&reg);
}
gotoxy (text.curx, text.cury);  /* restablece el cursor */
}

/*** resalta el texto en la ventana  ***/
void popHilite( POPUP *win, int row)
{
char attrib;
attrib= win->hilite + (win->hiback << 4); /* resalta atributos */
```

```
popRewrite(win,row,attrib);
}

/** pone el texto dentro de la ventana con atributos normales **/

void popNormal(POPUP *win, int row)
{
 char attrib;                    /** atributos normales **/
 attrib=win->normal + (win->normback<<4);
 popRewrite(win, row, attrib);
}

/** escribe una barra de menu descrita por  spec */
void menubar(MENUBAR *spec)
{
 struct  text_info text;
 int     p, s, c=0;
 gettextinfo(&text);       /* adquiere el estado actual del video */
 textcolor(spec->fore);    /* define colores  */
 textbackground(spec->back);
 gotoxy(1,spec->row);
 for(s=1; s < text.winright - text.winleft + 2; s++)
  cprintf("%c",' ');      /* define el fondo */
 gotoxy(1,spec->row);      /* inicio de la barra de menu  */
 for(p=0; spec->choice [p]; p++)    /* copia texto a la barra */
  if(spec->choice [p] != '\n')
    cprintf("%c",spec->choice [p]); /* escribe un caracter */
  else
    gotoxy(spec->interval * ++c, spec->row);  /* siguiente */
 /* regresa al estado inicial */
 textcolor(text.attribute & 0x8F);
 textbackground(text.attribute >> 4);
 gotoxy (text.curx, text.cury);
}
```

```
/*************************************************************************/
/*                  TEXTBOX.C: rutina  para dibujar cajas en modo texto   */
/*                   El argumento style determina el tipo de caja a dibujar: */
/*                   0 = no se dibuja caja                                 */
/*                   1 = linea simple                                      */
/*                   2 = doble linea                                       */
/*************************************************************************/

# include <conio.h>
void textbox (int left, int top, int rigth, int bottom , int style)
{
register r, c;
static bord [] [6]={
     {196, 179, 218, 191, 217, 192},
     {205, 186, 201, 187, 188, 200}
};
 if (style==0) return;
 --style;

 /*** dibuja lineas horizontales ***/
 for (c=left+1; c<rigth; c++){
  gotoxy(c,top);
  cprintf("%c", bord [style] [0]);
  gotoxy(c,bottom);
  cprintf("%c", bord [style] [0]);
 }

 /*** dibuja lineas verticales ***/
 for (r=top+1; r<bottom; r++){
  gotoxy (left,r);
  cprintf ("%c", bord [style] [1]);
  gotoxy (rigth,r);
  cprintf ("%c", bord [style] [1]);
 }

 /*** pon esquinas ***/
 gotoxy (left, top);  cprintf( "%c", bord [style] [2]);
 gotoxy (rigth, top); cprintf( "%c", bord [style] [3]);
 gotoxy (rigth, bottom); cprintf( "%c", bord [style] [4]);
 gotoxy (left, bottom); cprintf( "%c", bord [style] [5]);
}
```

ANEXO D

HOJAS DE ESPECIFICACIONES DE LOS CIRCUITOS UTILIZADOS

National Semiconductor

Programmable Array Logic (PAL®) 20-Pin Medium PAL Family

General Description

The 20-pin Medium PAL family contains four of the most popular PAL architectures used in industry. National Semiconductor's advanced Schottky TTL process with titanium tungsten fusible links is used in manufacturing the standard, Series-A, Series-A2, Series-B and Series-B2 devices. Series-D devices are manufactured using National Semiconductor's isoplanar "FAST-Z" TTL process with highly reliable "vertical-fuse" programmable cells. Vertical fuses are implemented using avalanche-induced migration ("AIM") technology offering very high programming yields and is an extension of National's FAST logic family. The 20-pin Medium PAL Family provides high-speed user-programmable replacements for conventional SSI/MSI logic with significant chip-count reduction.

Programmable logic devices provide convenient solutions for a wide variety of application-specific functions, including random logic, custom decoders, state machines, etc. By programming the programmable cells to configure AND/OR gate connections, the system designer can implement custom logic as convenient sum-of-products Boolean functions. System prototyping and design iterations can be performed quickly using these off-the-shelf products. A large variety of programming units and software makes design development and functional testing of PAL devices quick and easy.

The PAL logic array has a total of 16 complementary input pairs and 8 outputs generated by a single programmable AND-gate array with fixed OR-gate connections. Device outputs are either taken directly from the AND-OR functions (combinatorial) or passed through D-type flip-flops (registered). Registers allow the PAL device to implement sequential logic circuits. TRI-STATE® outputs facilitate busing and provide bidirectional I/O capability. The medium PAL family offers a variety of combinatorial and registered output mixtures, as shown in the Device Types table below.

On power-up, Series-D devices reset all registers to simplify sequential circuit design and testing. For these devices, direct register preload is also provided to facilitate device testing. Security fuses can be programmed to prevent direct copying of proprietary logic patterns in all the family devices.

Features

- As fast as 10 ns maximum propagation delay (combinatorial)
- User-programmable replacement for TTL logic
- High programming yield and reliability of vertical-fuse technology for Series-D products. (Programming equipment with certified vertical-fuse algorithm required.)
- Extension of FAST product line (Series-D).
- Large variety of JEDEC-compatible programming equipment and design development software available
- Fully supported by National PLAN™ development software
- Power-up reset for registered outputs (Series-D)
- Register preload facilitates device testing (Series-B,D)
- Security fuse prevents direct copying of logic patterns

Device Types

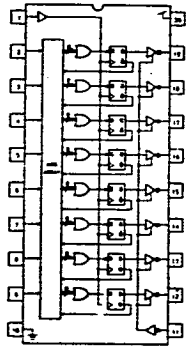
Device Type	Dedicated Inputs	Registered Outputs (with Feedback)	Combinatorial	
			I/Os	Outputs
PAL16L8	10	—	6	2
PAL16R4	8	4	4	—
PAL16R6	8	6	2	—
PAL16R8	8	8	—	—

Speed/Power Versions

Series	Example	Commercial		Military	
		t_{PD}	I_{CC}	t_{PD}	I_{CC}
Standard	PAL16L8	35 ns	180 mA	45 ns	180 mA
A	PAL16L8A	25 ns	180 mA	30 ns	180 mA
A2	PAL16L8A2	35 ns	90 mA	50 ns	90 mA
B	PAL16L8B	15 ns	180 mA	20 ns	180 mA
B2	PAL16L8B2	25 ns	90 mA*	30 ns	90 mA*
D	PAL16L8D	10 ns	180 mA		

*For the registered devices I_{CC} = 100 mA

Block Diagram—PAL16R8



TL/L/9291-1

Standard Series (PAL16L8, PAL16R4, PAL16R6, PAL16R8)

Absolute Maximum Ratings (Note 1)

If military/aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC}) (Note 2)	−0.5V to +7.0V
Input Voltage (Notes 2 and 3)	−1.5V to +5.5V
Off-State Output Voltage (Note 2)	−1.5V to +5.5V
Input Current (Note 2)	−30.0 mA to +5.0 mA
Output Current (I_{OL})	100 mA
Storage Temperature	−65°C to +150°C
Ambient Temperature with Power Applied	−65°C to +125°C
Junction Temperature	−65°C to +150°C

Recommended Operating Conditions

Symbol	Parameter		Military			Commercial			Units
			Min	Nom	Max	Min	Nom	Max	
V_{CC}	Supply Voltage		4.5	5	5.5	4.75	5	5.25	V
T_A	Operating Free-Air Temperature		−55			0		75	°C
T_C	Operating Case Temperature				125				°C
t_W	Clock Pulse Width	Low	25			25			ns
		High	25			25			ns
t_{SU}	Setup Time from Input or Feedback to Clock		45			35			ns
t_H	Hold Time of Input after Clock		0	−15		0	−15		ns
f_{CLK}	Clock Frequency (Note 4)	With Feedback		25	12.2		25	16.6	MHz
		Without Feedback		35	20		35	20	MHz

Electrical Characteristics Over Recommended Operating Conditions (Note 5)

Symbol	Parameter	Test Conditions			Min	Typ	Max	Units
V_{IL}	Low Level Input Voltage (Note 6)						0.8	V
V_{IH}	High Level Input Voltage (Note 6)				2			V
V_{IC}	Input Clamp Voltage	V_{CC} = Min, I = −18 mA				−0.8	−1.5	V
I_{IL}	Low Level Input Current (Note 7)	V_{CC} = Max, V_I = 0.4V				−0.02	−0.25	mA
I_{IH}	High Level Input Current (Note 7)	V_{CC} = Max, V_I = 2.4V					25	µA
I_I	Maximum Input Current	V_{CC} = Max, V_I = 5.5V					1.0	mA
V_{OL}	Low Level Output Voltage	V_{CC} = Min	I_{OL} = 12 mA	MIL		0.3	0.5	V
			I_{OL} = 24 mA	COM				
V_{OH}	High Level Output Voltage	V_{CC} = Min	I_{OH} = −2 mA	MIL	2.4	2.9		V
			I_{OH} = −3.2 mA	COM				
I_{OZL}	Low Level Off-State Output Current (Note 7)	V_{CC} = Max	V_O = 0.4V				−100	µA
I_{OZH}	High Level Off-State Output Current (Note 7)	V_{CC} = Max	V_O = 2.4V				100	µA
I_{OS}	Output Short-Circuit Current (Note 8)	V_{CC} = 5V, V_O = 0V			−30	−70	−130	mA
I_{CC}	Supply Current	V_{CC} = Max, Outputs Open				120	180	mA

Standard Series (PAL16L8, PAL16R4, PAL16R6, PAL16R8) (Continued)

Note 1: Absolute maximum ratings are those values beyond which the device may be permanently damaged. Proper operation is not guaranteed outside the specified recommended operating conditions.

Note 2: Some device pins may be raised above these limits during programming and preload operations according to the programmable specification.

Note 3: It is recommended that precautions be taken to minimize electrostatic discharge when handling and testing this product. Pins 1 and 11 are connected directly to the security fuses, and the security fuses may be damaged, preventing subsequent programming and verification of patterns.

Note 4: f_{CLK} with feedback is derived as $(t_{CLK} + t_{SU})^{-1}$.
f_{CLK} without feedback is derived as $(2t_{wh})^{-1}$.

Note 5: All typical values are for $V_{CC} = 5.0V$ and $T_A = 25°C$.

Note 6: These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.

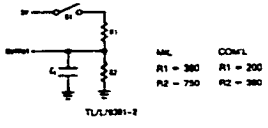
Note 7: Leakage current for bidirectional I/O pins is the worst case between I_{IL} and I_{OZL} or between I_{IH} and I_{OZH}.

Note 8: To avoid invalid readings in other parameter tests it is preferable to conduct the I_{OS} test last. To minimize internal heating, only one output should be shorted at a time with a maximum duration of 1.0 sec. each. Prolonged shorting of a High output may raise the chip temperature above normal and permanent damage may result.

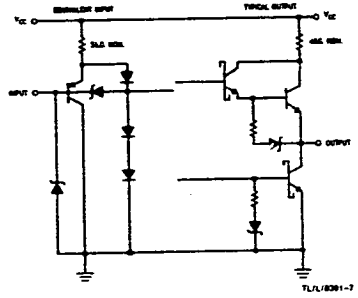
Switching Characteristics Over Recommended Operating Conditions

Symbol	Parameter	Test Conditions	Military			Commercial			Units
			Min	Typ	Max	Min	Typ	Max	
t_{PD}	Input or Feedback to Combinatorial Output	C_L = 50 pF, S1 Closed		25	45		25	35	ns
t_{CLK}	Clock to Registered Output or Feedback	C_L = 50 pF, S1 Closed		15	25		15	25	ns
t_{PZXG}	$\overline{G}$ Pin to Registered Output Enabled	C_L = 50 pF, Active High: S1 Open, Active Low: S1 Closed		15	25		15	25	ns
t_{PXZG}	$\overline{G}$ Pin to Registered Output Disabled	C_L = 5 pF, From V_{OH}: S1 Open, From V_{OL}: S1 Closed		15	25		15	25	ns
t_{PZX}	Input to Combinatorial Output Enabled via Product Term	C_L = 50 pF, Active High: S1 Open, Active Low: S1 Closed		25	45		25	35	ns
t_{PXZ}	Input to Combinatorial Output Disabled via Product Term	C_L = 5 pF, From V_{OH}: S1 Open, From V_{OL}: S1 Closed		25	45		25	35	ns

Test Load

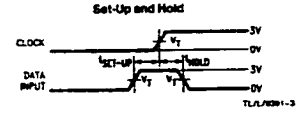


Schematic of Inputs and Outputs

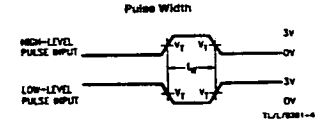


Test Waveforms

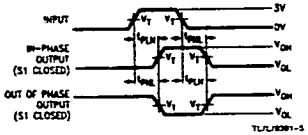
Set-Up and Hold



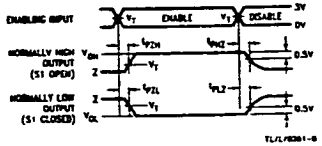
Pulse Width



Propagation Delay

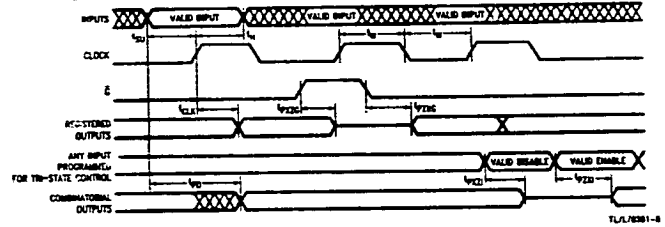


Enable and Disable

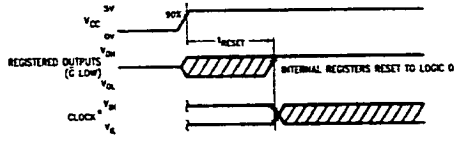


Notes:
V_T = 1.5V
C_L includes probe and jig capacitance.
In the examples above, the phase relationships between inputs and outputs have been chosen arbitrarily.

Switching Waveforms



Power-Up Reset Waveform



*The clock input should not be switched from low to high until after time t_{RESET}.

Functional Description

All of the 20-pin Medium PAL logic arrays consist of 16 complementary input lines and 64 product-term lines with a programmable cell at each intersection (2048 cells). The product terms are organized into eight groups of eight each. Seven or eight of the product terms in each group connect into an OR-gate to produce the sum-of-products logic function, depending on whether the output is combinatorial or registered.

For the fuse-link PAL devices (all PAL devices prior to National Series-D), an unprogrammed (intact) fuse establishes a connection between an input line (true or complement phase of an array input signal) and a product term; programming the fuse removes the connection. In the National Series-D vertical fuse PAL devices, a programmed vertical fuse cell establishes a connection between an input line and a product term. A product term is satisfied (logically true) while all of the input lines connected to it (via unprogrammed fuses for the fuse-link devices, or by programming the corresponding cells for the vertical fuse devices) are in the high logic state. Therefore, if both the true and complement of at least one array input is connected to a product line, that product term is always held in the low logic state (which is the state of all product terms in an unprogrammed fuse-link device). Conversely, if all input lines are disconnected from a product line, the product term and the resulting logic function would be held in the high state (which is the state of all product terms in an unprogrammed National Series-D PAL device).

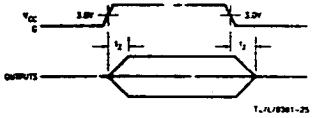
The medium PAL family consists of four device types with differing mixtures of combinatorial and registered outputs. The 16L8, 16R4, 16R6 and 16R8 architectures have 0, 4, 6 and 8 registered outputs, respectively, with the balance of the 8 outputs combinatorial. All outputs are active-low and have TRI-STATE capability.

Each combinatorial output has a seven product-term logic function, with the eighth product term being used for TRI-STATE control. A combinatorial output is enabled while the TRI-STATE product term is satisfied (true). Combinatorial outputs also have feedback paths from the device pins into the logic array (except for two outputs on the 16L8). This allows a pin to perform bidirectional I/O or, if the associated TRI-STATE control product term were programmed to remain unsatisfied (always false), the output driver would remain disabled and the pin could be used as an additional dedicated input.

Registered outputs each have an eight product-term logic function feeding into a D-type flip-flop. All registers are triggered by the high-going edge of the clock input pin. All registered outputs are controlled by a common output enable ($\overline{G}$) pin (enabled while low). The output of each register is also fed back into the logic array via an internal path. This provides for sequential logic circuits (state machines, counters, etc.) which can be sequenced even while the outputs are disabled.

Series-D Medium PAL devices reset all registers to a low state upon power-up (active-low outputs assume high logic levels if so called). This may simplify sequential circuit design and test. To ensure successful power-up reset, V_{CC} must rise monotonically until the specified operating voltage is attained. During power-up, the clock input should assume a valid, stable logic state as early as possible to avoid interfering with the reset operation. The clock input should also remain stable until after the power-up reset operation is completed to allow the registers to capture the proper next state on the first high-going clock transition.

For the National Series-D PAL device, during power-up, all outputs are held in the high-impedance state until DC power supply conditions are met (V_{CC} approximately 3.0V), after which they may be enabled by the TRI-STATE control product terms (combinatorial outputs) or the $\overline{G}$ pin (registered outputs). Whenever V_{CC} goes below 3V (at 25°C), the outputs are disabled as shown in *Figure 1* below.



T-7/8391-25

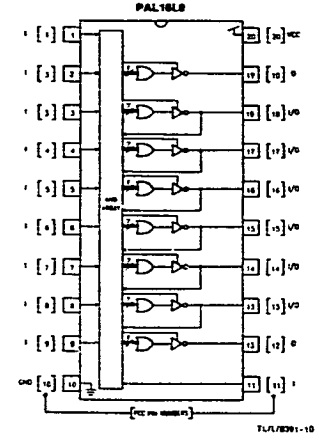
Note: t_2 is less than 100 ns.

FIGURE 1. Series-D Power-Up TRI-STATE Waveform

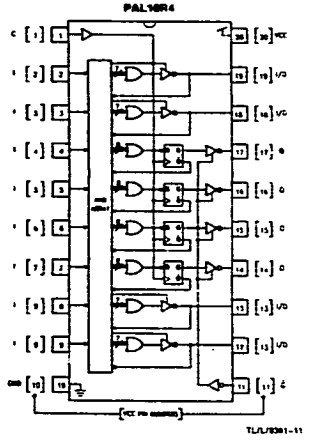
In an unprogrammed National Series-D PAL device, no array inputs are connected to any product-term lines. Therefore, all combinatorial outputs would be enabled and driving low logic levels (after power-up is completed). All registers would still initialize to the low state, but would become permanently set (low-level outputs, if enabled) following the first clock transition.

As with any TTL logic circuits, unused inputs to a PAL device should be connected to ground, V_{OL}, V_{OH}, or resistively to V_{CC}. However, switching any input not connected to a product term or logic function has no effect on its output logic state.

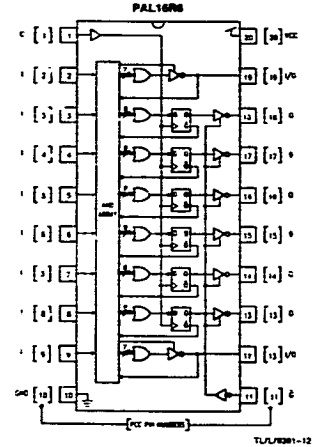
20-Pin Medium PAL Family Block Diagrams—DIP Connections



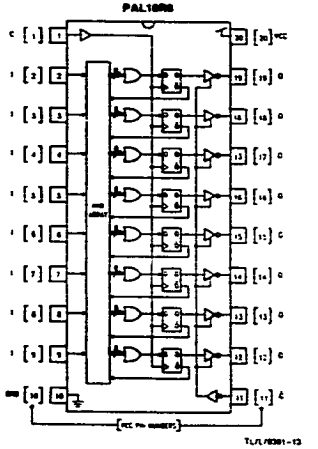
TL/L/8391-10



TL/L/8391-11

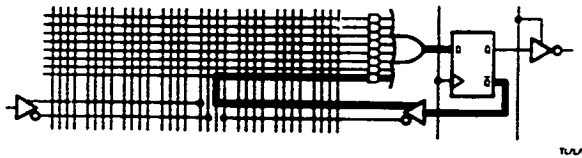


TL/L/8391-12



TL/L/8391-13

Typical Registered Logic Function With Feedback



TL/L/9391-16

Functional Description (Continued)

CLOCK FREQUENCY SPECIFICATION

The clock frequency (f_{CLK}) parameter specifies the maximum speed at which a registered PAL device is guaranteed to operate. Clock frequency is defined differently for the two cases in which register feedback is used versus when it is not. In a data-path type application, where the logic functions fed into the registers are not dependent on register feedback from the previous cycle (i.e.—based only on external inputs), the minimum required cycle period (t_{CLK}^{-1} without feedback) is defined as the greater of the minimum clock period (t_W high + t_W low) and the minimum "data window" period (t_{SU} + t_H). This assumes optimal alignment between data inputs and the clock input. In sequential logic applications such as state machines, the minimum required cycle period (t_{CLK}^{-1} with feedback) is defined as t_{CLK} + t_{SU}. This provides sufficient time for outputs from the registers to feed back through the logic array and set-up on the inputs to the registers before the end of each cycle.

Output Register Preload

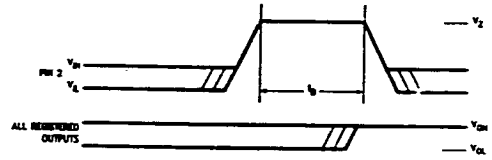
In the 20-pin Medium PAL family, register preload is available on the Series-D PAL devices only.

The output register preload feature simplifies device testing since any state may be loaded into the registers at any time during the functional test sequence. This allows complete verification of sequential logic circuits, including states that are normally impossible or difficult to reach. Register preload is not an operational mode and is not intended for board level testing because elevated voltage levels are required. The programming system normally provides the preload capability as part of its functional test facility.

The preload function allows the register to be loaded directly and asynchronously with any desired pattern. These Vertical-fuse devices provide two register preload operations:

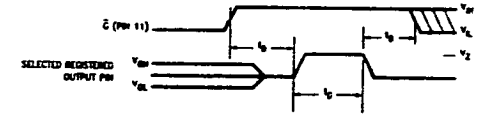
1. All registers can be reset to the low state (high-level outputs) by applying the elevated control voltage (V_Z) to input Pin 2* for time t_D (*Figure 2a*).
2. Selected registers can be set to the high state (low-level outputs) as follows (*Figure 2b*):
   a. All registered outputs are disabled by raising the $\overline{G}$ input Pin 1* to V_{IH}.
   b. After time t_D, the selected registered output pins are raised to the elevated control voltage (V_Z) for time t_D to set the corresponding registers.

*Applies to both DIP and PCC packages for 20-pin PAL devices



TL/L/9391-25

a) To Reset All Registers



TL/L/9391-27

b) To Set Selected Registers

Note: V_Z = 9.5V to 10.0V, t_D min = 500 ns.

FIGURE 2. Series-D Register Preload Waveforms

Security Fuse

Security fuses are provided on all National PAL devices which, when programmed, inhibit any further programming or verifying operations. This feature prevents direct copying of proprietary logic patterns. The security fuses should be programmed only after programming and verifying all other device fuses. Register preload is not affected by the security fuses.

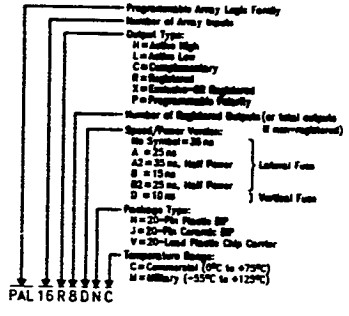
Design Development Support

A variety of software tools and programming hardware is available to support the development of designs using PAL products. Typical software packages accept Boolean logic equations to define desired functions. Most are available to run on personal computers and generate JEDEC-compatible "fuse maps". The industry-standard JEDEC format ensures that the resulting fuse-map files can be down-loaded into a variety of programming equipment. Many software packages and programming units support a wide variety of programmable logic products as well. The PLAN™ software package from National Semiconductor supports all programmable logic products available from National and is fully JEDEC-compatible. PLAN software also provides automatic device selection based on the designer's Boolean logic equations.

In National Series-D PAL devices, logical and physical connections between array input lines and product-term lines are established when vertical fuse cells are programmed. This is opposite to other PAL products based on fusible links in which connections are established when fuses are left unprogrammed (intact). This difference is compensated by the vertical-fuse PAL programming algorithm so that the *user's design development process looks the same.* (The only functional difference due to vertical-fuse technology is the behavior of "unprogrammed" devices.) The JEDEC programming maps produced by PAL development software for all Medium PAL devices denote a "connection" with a "0", and a "non-connection" with a "1". The programming algorithms for most fuse-link PLDs program fuses where ones are located in the map to remove corresponding connections, whereas the algorithm for National Series-D PAL products automatically compensates by programming vertical-fuse cells where zeroes are located in the map to establish connections. Therefore, the *same JEDEC map* representing the user's desired logic equations produces the *same functional results* when using either PAL technology. The user need only provide the appropriate device code and/or adapter for the programming equipment to invoke the proper programming algorithm. Only programmers with the certified National Series-D vertical-fuse PAL programming algorithm can be used to program these vertical-fuse devices.

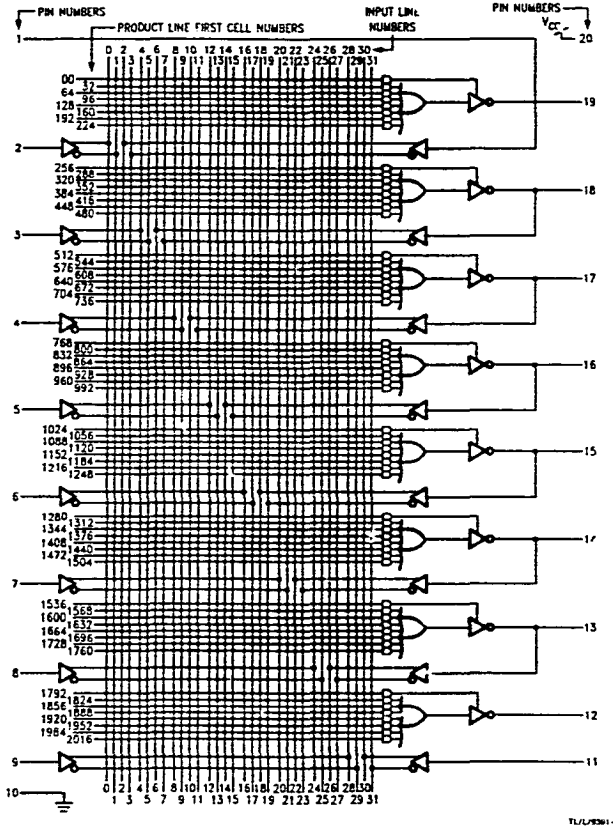
Detailed logic diagrams showing all JEDEC fuse-map addresses for the 20-pin Medium PAL family are provided for direct map editing and diagnostic purposes. For a list of current software and programming support tools available for these devices, please contact your local National Semiconductor sales representative or distributor. If detailed specifications of the PAL programming algorithm are needed, please contact the National Semiconductor Programmable Device Support Department.

Ordering Information



TL/L/9391-18

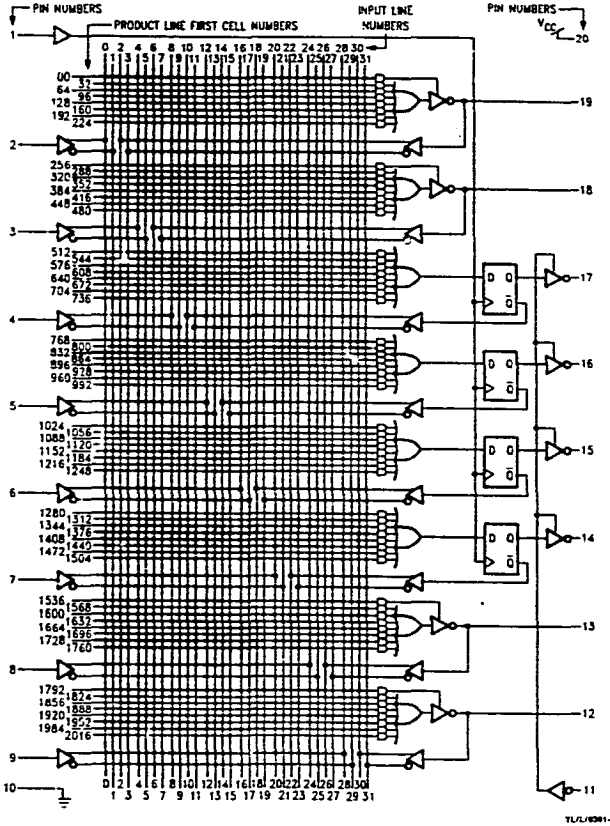
ogic Diagram—PAL16L8



TL/L/9391-19

DEC Logic Array Cell Number = Product Line First Cell Number + Input Line Number

Logic Diagram—PAL16R4



TL/L/9391-20

JEDEC Logic Array Cell Number = Product Line First Cell Number + Input Line Number

National Semiconductor

GAL16V8 Generic Array Logic

General Description

The NSC E²CMOS™ GAL® device combines a high performance CMOS process with electrically erasable floating gate technology. This programmable memory technology applied to array logic provides designers with reconfigurable logic and bipolar performance at significantly reduced power levels.

The 20-pin GAL16V8 features 8 programmable Output Logic Macrocells (OLMCs) allowing each TRI-STATE® output to be configured by the user. Additionally, the GAL16V8 is capable of emulating, in a functional/fuse map/parametric compatible device, all common 20-pin PAL® device architectures.

Programming is accomplished using readily available hardware and software tools. NSC guarantees a minimum 100 erase/write cycles.

Unique test circuitry and reprogrammable cells allow complete AC, DC, cell and functionality testing during manufacture. Therefore, NSC guarantees 100% field programmability and functionality of the GAL devices. In addition, electronic signature is available to provide positive device ID. A security circuit is built-in, providing proprietary designs with copy protection.

Features

- Electrically erasable cell technology
  - Reconfigurable logic
  - Reprogrammable cells
  - Guaranteed 100% yields
- High performance E²CMOS technology
  - Low power: 45 mA/90 mA max active
  - High Speed: 20 ns–35 ns max access
- Eight output logic macrocells
  - Maximum flexibility for complex logic designs
  - Also emulates 20-pin PAL devices with full function/fuse map/parametric compatibility
- Preload and power-on reset of all registers
  - 100% functional testability
- Fully supported by National PLAN™ development software
- High speed programming algorithm
- Security cell prevents copying logic

PAL Replacement by Device Type

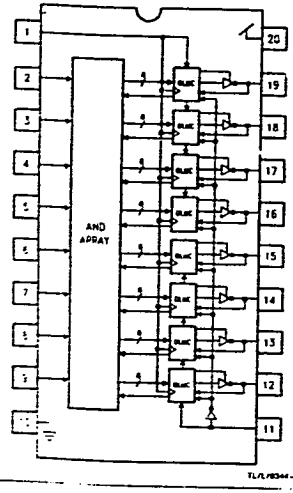
"Small-PAL" Mode				"Registered-PAL" Mode			"Medium-PAL" Mode
10L8	12L6	14L4	16L2	16R8	16R8	16R4	16L8
10H8	12H6	14H4	16H2	16RP8	16RP8	16RP4	16H8
10P8	12P6	14P4	16P2				16P8

PAL Replacement by Speed/Power

PAL			GAL
Rev	Speed	Power*	Rev
B-2	25 ns	90 mA	25L or 20L or 25Q (45 mA)
A	25 ns	180 mA	25L or 20L (90 mA)
B-2 MIL	30 ns	90 mA	35L or 25L (110 mA) or 30Q (55 mA)
A MIL	30 ns	180 mA	35L or 25L (110 mA)
B-4	35 ns	45 mA	30Q
A-2	35 ns	90 mA	30Q (45 mA)
STD	35 ns	180 mA	25L (90 mA)

*Shown for Medium PAL products

Block Diagram—GAL16V8



TL/L/10344–1

Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{CC}) (Note 2)	−0.5V to +7.0V
Input Voltage (Note 2)	−2.5V to V_{CC} +1.0V
Off-State Output Voltage (Note 2)	−2.5V to V_{CC} +1.0V
Output Current	+100 mA
Storage Temperature	−65°C to +150°C
Ambient Temperature with Power Applied	−65°C to +125°C
Junction Temperature	−65°C to +150°C
Lead Temperature (Soldering, 10 seconds)	260°C
ESD Tolerance	500V

C_{ZAP} = 100 pF
R_{ZAP} = 150Ω
Test Method: Human Body Model
Test Specification: NSC SOP-5-028

Recommended Operating Conditions

SUPPLY VOLTAGE AND TEMPERATURE

Symbol	Parameter	Commercial			Industrial			Military			Units
		Min	Nom	Max	Min	Nom	Max	Min	Nom	Max	
V_{CC}	Supply Voltage	4.75	5	5.25	4.5	5	5.5	4.5	5	5.5	V
T_A	Operating Free-Air Temperature	0	25	75	−40	25	85	−55	25		°C
T_C	Operating Case Temperature									125	°C

AC TIMING REQUIREMENTS

Symbol	Parameter		GAL16V8-20L		GAL16V8-25Q GAL16V8-25L		GAL16V8-30Q GAL16V8-30L		Units
			COM/IND		COM/IND MIL*		COM/IND* MIL		
			Min	Max	Min	Max	Min	Max	
t_{SU}	Set-Up Time (Input or Feedback before Clock)		15		20		25		ns
t_H	Hold Time (Input after Clock)		0		0		0		ns
t_W	Clock Pulse Width (High/Low)		12		15		15		ns
t_{CYCLE}	Clock Cycle Period (with Feedback) (Note 3)		30		35		45		ns
f_{CLK}	Clock Frequency (Note 4)	With Feedback		33.3		28.5		22.2	MHz
		Without Feedback		41.6		33.3		33.3	
f_I	Input Frequency (Note 5)			50.0		40.0		33.3	
t_{PR}	Clock Valid after Power-Up			100		100		100	ns

* -25Q available only in COM/IND operating ranges;
-30L available only in MIL operating range (for COM/IND, use -25L or -30Q).

Note 1: Absolute maximum ratings are those values beyond which the device may be permanently damaged. Proper operation is not guaranteed outside the specified recommended operating conditions.

Note 2: Some device pins may be raised above these limits during programming and preload operations according to the applicable specification.

Note 3: $t_{CYCLE} = t_{SU} + t_{CLK}$

Note 4: f_{CLK} (with feedback) = $(t_{CYCLE})^{-1}$
f_{CLK} (without feedback) = $(2\ t_W)^{-1}$

Note 5: $f_I = (t_{PD})^{-1}$

Electrical Characteristics Over Recommended Operating Conditions

Symbol	Parameter		Conditions		Temperature Range	Min	Typ	Max	Units
V_{IH}	High Level Input Voltage					2.0		$V_{CC}+1$	V
V_{IL}	Low Level Input Voltage					−1.0		0.8	V
V_{OH}	High Level Output Voltage		V_{CC} = Min	I_{OH} = −3.2 mA	COM/IND	2.4			V
				I_{OH} = −2.0 mA	MIL	2.4			V
V_{OL}	Low Level Output Voltage		V_{CC} = Min	I_{OL} = 24 mA	COM/IND			0.5	V
				I_{OL} = 12 mA	MIL			0.5	V
I_{OZH}	High Level Off State Output Current		V_{CC} = Max, V_O = V_{CC} (Max)					10	μA
I_{OZL}	Low Level Off State Output Current		V_{CC} = Max, V_O = GND					−10	μA
I_I	Maximum Input Current		V_{CC} = Max, V_I = V_{CC} (Max)					10	μA
I_{IH}	High Level Input Current		V_{CC} = Max, V_I = V_{CC} (Max)					10	μA
I_{IL}	Low Level Input Current		V_{CC} = Max, V_I = GND					−10	μA
I_{OS}*	Output Short Circuit Current		V_{CC} = 5.0V, V_O = GND			−30		−150	mA
I_{CC}	Supply Current	Quarter Power (GAL16V8-25Q, -30Q)	f = 15 MHz, V_{CC} = Max		COM			45	mA
		Half Power (GAL16V8-20L, -25L, -30L)	f = 15 MHz, V_{CC} = Max		MIL/IND			55	mA
					COM			90	mA
					MIL/IND			110	mA
C_I	Input Capacitance		V_{CC} = 5.0V, V_I = 2.0V					12	pF
$C_{I/O}$	I/O Capacitance		V_{CC} = 5.0V, $V_{I/O}$ = 2.0V					15	pF

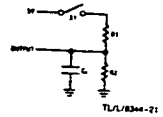
*One output at a time for a maximum duration of one second @ 25°C.

Switching Characteristics Over Recommended Operating Conditions

Symbol	Parameter	Conditions	GAL16V8-20L COM/IND		GAL16V8-25*, GAL16V8-25L COM/IND MIL*		GAL16V8-30Q GAL16V8-30L COM/IND* MIL		Units
			Min	Max	Min	Max	Min	Max	
t_{PD}	Input or Feedback to Combinatorial Output	S1 Closed, C_L = 50 pF		20		25		30	ns
t_{CLK}	Clock to Registered Output or Feedback	S1 Closed, C_L = 50 pF		15		15		20	ns
t_{PZXG}	$\overline{G}$ ↓ to Registered Output Enabled	Active High: S1 Open, C_L = 50 pF Active Low: S1 Closed, C_L = 50 pF		18		20		25	ns
t_{PXZG}	$\overline{G}$ ↑ to Registered Output Disabled	From V_{OH}: S1 Open, C_L = 5 pF From V_{OL}: S1 Closed, C_L = 5 pF		18		20		25	ns
t_{PZXI}	Input to Combinatorial Output Enabled via Product Term	Active High: S1 Open, C_L = 50 pF Active Low: S1 Closed, C_L = 50 pF		20		2-		30	ns
t_{PXZI}	Input to Combinatorial Output Disabled via Product Term	From V_{OH}: S1 Open, C_L = 5 pF From V_{OL}: S1 Closed, C_L = 5 pF		20		25		30	ns
t_{RESET}	Power-Up to Registered Output High	S1 Closed, C_L = 50 pF		45		45		45	μs

*-25Q available only in COM/IND operating ranges.
-30L available only in MIL operating range (for COM/IND use -25L or -30Q)

AC Test Load

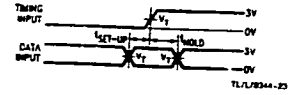


	MIL	COM/IND
R1 =	390 Ω	200 Ω
R2 =	750 Ω	390 Ω

TL/L/8344-21

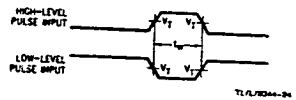
Test Waveforms

Setup and Hold



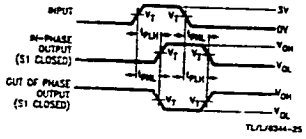
TL/L/8344-23

Pulse Width



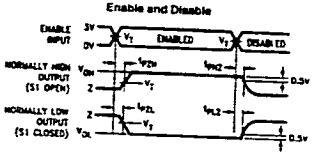
TL/L/8344-24

Propagation Delay



TL/L/8344-25

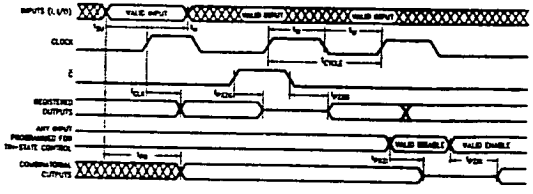
Enable and Disable



TL/L/8344-26

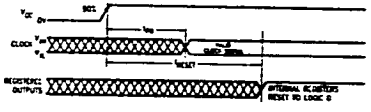
Notes:
C_L includes probe and jig capacitance.
V_T = 1.5V.
Test inputs have ...se and fall times of 5 ns between 0.3V and 2.7V.
In the examples above, the phase relationships between inputs and outputs have been chosen arbitrarily.

Switching Waveforms



TL/L/8344-7

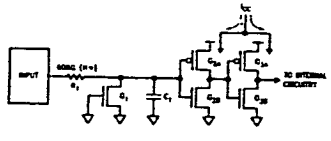
Power-Up Reset Waveforms



TL/L/8344-10

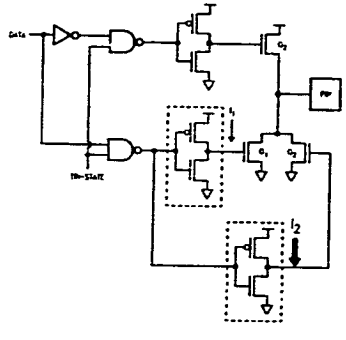
Input/Output Schematics

Input Translator/Buffer



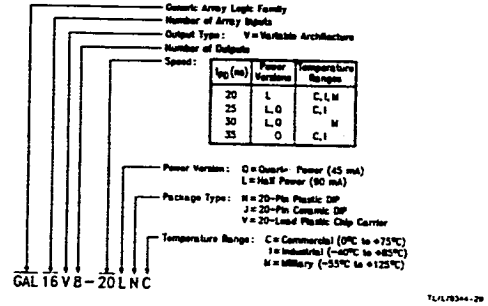
TL/L/8344-27

Phased Output Turn-On Circuit



TL/L/8344-28

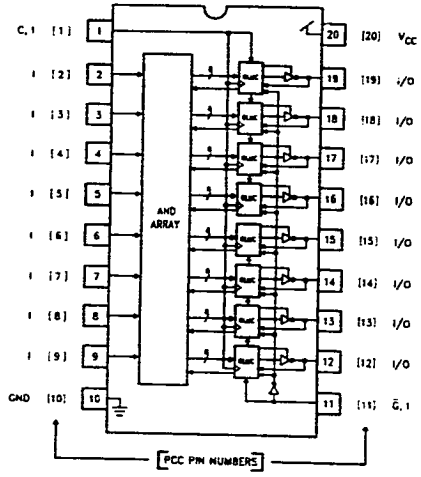
Ordering Information



t_{PD} (ns)	Power Versions	Temperature Ranges
20	L	C, I, M
25	L, Q	C, I
30	L, Q	M
35	Q	C, I

TL/L/8344-29

GAL16V8 Block Diagram—DIP Connections



TL/L/8344-30

FIGURE 1

Functional Description

The GAL logic array consists of a programmable AND array with fixed OR-gate connections, similar to the bipolar PAL architecture. The logic array is organized as 16 complementary input lines crossing 64 "product term" lines with a programmable E²PROM cell at each intersection (2048 cells). Each programmable cell may establish a connection between an input line (true or complement phase of an array input signal) and a product term. A product term is satisfied (logically true) while all of the input lines "connected" to it are in the high logic state.

The 64 product terms are organized into eight output groups with eight terms each. Seven or eight of the product terms in each output group feed into an OR-gate to produce each output logic function; one of the product terms may instead be used to control the associated TRI-STATE device output. The fundamental transfer function of each GAL output is the familiar Boolean sum-of-products. Design development software is available which accepts Boolean equations and converts them automatically into GAL programming patterns.

As shown in the GAL16V8 Block Diagram *(Figure 1)*, a total of eight output logic functions are available. Each of the AND/OR logic functions feeds into an "output logic macrocell" (OLMC). The eight OLMCs control the flow of input and output signals between the logic array and the device's I/O pins.

Under control of an OLMC, each output may be designated either registered or combinatorial (non-registered). In the registered output configuration, the logic function output passes through a D-type flip-flop triggered by the rising edge of the clock input. Additionally, the logic function's output polarity may be designated active-low or active-high (adjusted before the register, if present). OLMC options such as these are selected using a set of programmable architecture control cells. These architecture cells are normally configured automatically by the development software or programming hardware.

All of the possible I/O configurations of the GAL16V8 are classified into three basic modes: "Small-PAL" mode, "Registered-PAL" mode and "Medium-PAL" mode. These modes correspond to the architectures of the PAL families which the GAL16V8 can emulate. The modes determine the mixture of OLMC configurations which can be selected for the device. The OLMC Selection table (Table I) lists which functions can be selected on the device pin" 1 and pins" 11 through 19 for each of the three modes. The logic diagrams in *Figure 3* illustrate these OLMC functions.

"OUTPUT" represents the always-active combinatorial output configuration available in the "Small-PAL" mode. "REGISTER" is the registered output with register feedback available in the "Registered-PAL" mode. "I/O" is the combinatorial bidirectional I/O available in "Registered-PAL" and "Medium-PAL" modes. "TRI-STATE" is the TRI-STATE combinatorial output function appearing on pins 12 and 19 in the "Medium-PAL" mode. "INPUT" in Table I denotes an OLMC used as a dedicated input only.

*Applies to both 20-pin DIP and 20-lead PCC packages for GAL16V8.

20-Lead PCC Connection Diagram

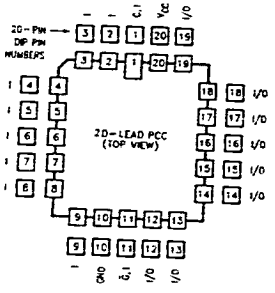
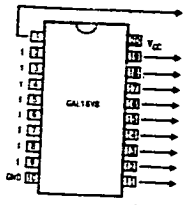


FIGURE 2

TL/L/8344-31

OLMC Selection Table



TL/L/8344-41

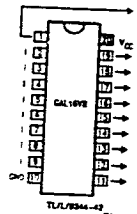
TABLE I

"Small-PAL" Mode	"Registered-PAL" Mode	"Medium-PAL" Mode
INPUT	CLOCK	INPUT
INPUT or OUTPUT*	REGISTER or I/O	TRI-STATE**
INPUT or OUTPUT*	REGISTER or I/O	I/O
INPUT or OUTPUT*	REGISTER or I/O	I/O
OUTPUT*	REGISTER or I/O	I/O
OUTPUT*	REGISTER or I/O	I/O
INPUT or OUTPUT*	REGISTER or I/O	I/O
INPUT or OUTPUT*	REGISTER or I/O	I/O
INPUT or OUTPUT*	REGISTER or I/O	I/O
INPUT	OUPUT ENABLE ($\overline{G}$)	TRI-STATE**
		INPUT

*Active combinatorial output

**TRI-STATE combinatorial output

PAL Replacement Configurations



TL/L/8344-42

TABLE II

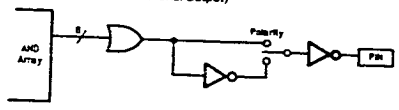
"Small PAL" Mode				"Registered-PAL"			"Medium-PAL"
INPUT	INPUT	INPUT	INPUT	CLOCK	CLOCK	CLOCK	INPUT
OUTPUT*	INPUT	INPUT	INPUT	REGISTER	I/O	I/O	TRI-STATE**
OUTPUT*	OUTPUT*	INPUT	INPUT	REGISTER	REGISTER	I/O	I/O
OUTPUT*	OUTPUT*	OUTPUT*	INPUT	REGISTER	REGISTER	REGISTER	I/O
OUTPUT*	OUTPUT*	OUTPUT*	OUTPUT*	REGISTER	REGISTER	REGISTER	I/O
OUTPUT*	OUTPUT*	OUTPUT*	OUTPUT*	REGISTER	REGISTER	REGISTER	I/O
OUTPUT*	OUTPUT*	OUTPUT*	INPUT	REGISTER	REGISTER	REGISTER	I/O
OUTPUT*	OUTPUT*	INPUT	INPUT	REGISTER	REGISTER	I/O	I/O
OUTPUT*	INPUT	INPUT	INPUT	REGISTER	I/O	I/O	TRI-STATE**
INPUT	INPUT	INPUT	INPUT	G	G	G	INPUT
10L8	12L6	14L4	16L2	16R8	16R6	16R4	16L8
10H8	12H6	14H4	16H2	16RP8	16RP6	16RP4	16H8
10P8	12P6	14P4	16P2				16P8
EMULATED PAL PRODUCTS							

*Active combinatorial output.

**TRI-STATE combinatorial output.

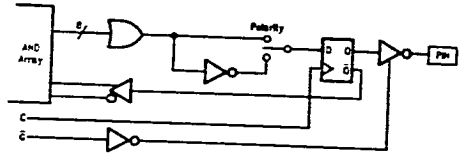
OLMC Configurations

OUTPUT (Active Combinatorial Output)



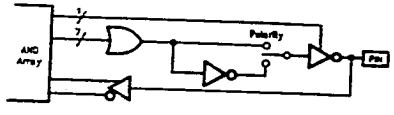
TL/L/8344–32

REGISTER (Registered Output)



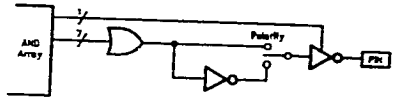
TL/L/8344–33

I/O (Combinatorial Input/Output)



TL/L/8344–34

TRI-STATE (TRI-STATE Combinatorial Output)



TL/L/8344–#3

FIGURE 3

Functional Description (Continued)

In the "Small-PAL" and "Medium-PAL" modes (Table I), pins* 1 and 11 are always dedicated inputs. In the "Registered-PAL" mode, however, pin* 1 becomes the clock input controlling all OLMC registers, and pin* 11 becomes the output enable ($\overline{G}$) input controlling the TRI-STATE outputs of all registered OLMCs. Within the "Small-PAL" and "Registered-PAL" modes in Table I, the functions of pins* 12 through 19 can be selected individually from either of the two functions listed. For example, in "Registered-PAL" mode, pins* 12 through 19 can each be designated as either a registered output or a combinatorial I/O. The "Medium-PAL" mode represents a single fixed configuration used to emulate combinatorial medium PAL devices (16L8, 16H8, 16P8).

Table II lists the bipolar PAL products which the GAL16V8 can emulate, and the specific input/output configurations used. This is just a subset, however, of all the configurations provided in Table I.

All registers in a GAL device are reset to the low state upon power-up. The active-low outputs, in turn, assume high logic levels (if enabled) regardless of the selected output polarity. This may simplify sequential circuit design and test. To ensure successful power-up reset, V_{CC} must rise monotonically until the specified operating voltage is attained. During power-up, the clock input should assume a valid, stable logic state as early as possible (within the specified time, t_{PR}) to avoid interfering with the reset operation. The clock input should also remain stable until after the power-up reset operation is completed to allow the registers to capture the proper next state on the first high-going clock transition.

It should be noted that the switching of any input not logically connected to a product term or logic function has no effect on the associated output logic state. To minimize power consumption, however, unused inputs should be connected to a stable logic level such as ground or V_{CC} (CMOS GAL inputs may be tied directly to the supply voltage without causing excessive loading conditions).

*Applies to both 20-pin DIP and 20-lead PCC packages for GAL16V8.

Clock/Input Frequency Specifications

The clock frequency (f_{CLK}) parameter listed in the Recommended Operating Conditions table specifies the maximum speed at which the GAL registers are guaranteed to operate. Clock frequency is defined differently for the two cases in which register feedback is used versus when it is not. In a data-path type application, when the logic functions fed into the registers are not dependent on register feedback from the previous cycle (i.e. based only on external inputs), the minimum required cycle period (t_{CLK}^{-1} without feedback) is defined as the greater of the minimum clock period (t_w high + t_w low) and the minimum "data window" period (t_{su} + t_h). This assumes optimal alignment between data inputs and the clock input. In sequential logic applications such as state machines, the minimum required cycle period (t_{CYCLE} = t_{CLK}^{-1} with feedback) is defined as $t_{CLK} = t_{su}$. This provides sufficient time for outputs from the registers to feed back through the logic array and set up on the inputs to the registers before the end of each cycle.

The input frequency (f_I) parameter specifies the maximum rate at which each GAL input can be toggled and still produce valid logic transitions on each combinatorial output. The f_I specification is derived as the inverse of the combinatorial propagation delay (t_{pd}).

Design Development Support

A variety of software tools and programming equipment is available to support the development of designs using GAL products. Typical software packages accept Boolean logic equations to define desired functions. Most are available to run on personal computers and generate a JEDEC-compatible "cell-map" (analogous to a PAL "fuse-map"). The industry-standard JEDEC format ensures that the resulting cell-map file can be down-loaded into a variety of programming equipment. Many software packages and programming units support a large variety of programmable logic products as well. The PLANT™ software package from National Semiconductor supports all programmable logic products available from National and is fully JEDEC-compatible. PLAN software also provides automatic device selection based on the designer's Boolean logic equations.

National strongly recommends using only approved programming hardware and software for developing GAL designs. Programming using unapproved equipment generally voids all guarantees. Approved programmers incorporate specialized programming algorithms that program the array and automatically configure the architecture cells. To ensure data retention and reliability, the programming algorithm also tracks the number of programming cycles to which each GAL device has been subjected since shipment, and stores this information automatically in the device.

The special GAL programming algorithm can also program a GAL device using a standard fuse-map developed for any of the emulated PAL products. PAL fuse-maps can be created by any JEDEC-compatible PAL development software or by loading the fuse pattern from an existing programmed PAL device into the programming unit (provided the PAL device has not been secured). However, to utilize the full flexibility of the GAL architecture, true GAL development software (such as PLAN software) is recommended.

Detailed logic diagrams showing all JEDEC cell-map addresses in the GAL logic array and OLMC are provided for direct map editing and diagnostic purposes (see "Programming Details"). For a list of current software and programming support tools available for these devices, please contact your local National sales representative or distributor. If detailed specifications of the GAL programming algorithm are needed, please contact the National Semiconductor Programmable Device Support department.

Security Cell

A security cell is provided on all GAL16V8 devices as a deterrent to unauthorized copying of the array configuration patterns. Once programmed, the circuitry enabling array access is disabled, preventing further programming or verification of the array. The security cell can be erased only in conjunction with the array during a bulk erase cycle, so the original configuration can never be examined once this cell is programmed.

Electronic Signature

Each GAL device contains an electronic signature word consisting of 64 bits of reprogrammable memory. The electronic signature word can be programmed to contain any identification information desired by the user. Some uses include pattern identification labels, revision numbers, dates, inventory control information, etc. The data stored in the electronic signature word has no effect on the functionality of the device. The information is read out of the device using the normal program verification procedure provided by the programming equipment. The information may be accessed at any time independent of the state of the security cell. National's PLAN development software allows electronic signature data to be entered by the user and downloaded to the programming equipment.

Bulk Erase

The programming equipment automatically performs a bulk erase operation prior to each programming operation. No special erase operation need be performed by the user. Bulk erase clears the logic array, architecture cells, security cell, and electronic signature information. The GAL device is thereby reverted back to its virgin state.

Latch-Up Protection

GAL devices are designed with an on-chip charge pump to negatively bias the substrate. The negative bias is of sufficient magnitude to prevent input undershoots from causing the circuitry to latch. Additionally, outputs are designed with n-channel pullups instead of the traditional p-channel pullups to eliminate any possibility of SCR induced latching.

Manufacturer Testing

Because of E²CMOS technology, GAL devices can be reprogrammed in milliseconds. This allows each device to be completely tested by the manufacturer using numerous logic array and architecture patterns prior to shipping. Every programmable cell and every logic path through every device is fully tested for programmability, functionality and performance to all AC and DC parameters. The customer can therefore expect 100% programming and functional yield and 100% compliance of all GAL products to datasheet specifications.

The testing procedure performed on all GAL devices by the manufacturer tests all aspects of device operation. Extensive testing of all programmable cells in the device include margin testing, internal verify, and program retention during high-temperature bake. All DC and AC parameters are tested at hot and cold temperatures using a variety of worst case logic and signal patterns. Functional tests include reprogramming each OLMC to all valid architectural configurations.

Register Preload

The register preload feature allows OLMC registers to be directly loaded with any desired data pattern. It also allows the present state of OLMC registers to be examined regardless of TRI-STATE control conditions. This simplifies testing of devices after programming. A device may be put into any desired register state at any point during the functional test sequence. The test sequence may then be resumed to verify proper next-state transitions. This allows complete verification of sequential logic circuits, including states that are normally impossible or difficult to reach. It may also shorten the overall test time significantly.

Register preload is not an operational mode and is not intended for board-level testing because elevated voltage levels must be applied to the device. The programming equipment normally provides the register preload capability as part of its functional test facility. Note that the testing of GAL devices after programming by the user may be considered unnecessary because all E²CMOS GAL products are completely tested by the manufacturer, guaranteeing 100% post-programming functional yield.

The register preload algorithm is described for those users who wish to test programmed GAL devices using test equipment other than approved GAL programming equipment. As shown in the Register Preload Waveform in *Figure 5*, the preload sequence must not begin until the normal power-up reset operation has completed (after time t_{RESET}). The device is placed into preload mode by raising the "PRLD" input (pin" 11) to voltage V_{IES}, as specified in the Register Preload Specifications (Table III).

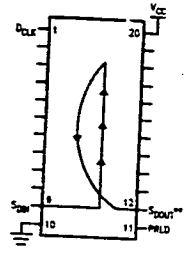
To preload the OLMC registers, a series of data bits are shifted into the device on the "S_{DIN}" input (pin" 9), one bit for each OLMC in which registered output has been selected. (Non-registered OLMCs are bypassed.) The shift sequence is clocked by the rising edge of the "D_{CLK}" input (pin" 1). The data stream is shifted in through the registered OLMC with the lowest corresponding pin number, and then "upward" through all remaining registered OLMCs in pin-number ascending order. Therefore, the first data bit in the series is ultimately loaded into the registered OLMC with the highest corresponding pin number, as shown in *Figure 4*.

As the data series is shifted into the S_{DIN} input, the contents of all registers (in registered OLMCs) are shifted "upward" and out onto the "S_{DOUT}" output (pin" 12). Complete present-state information can be examined in this manner. Test fixtures can be devised to test several GAL devices in which the S_{DOUT} pin of each chip is connected to the S_{DIN} pin of the next, and all preload and present-state data can be shifted around a single serial loop.

Note that when shifting register data into S_{DIN} or out of S_{DOUT}, V_{IL}/V_{OL} = register reset (0), and V_{IH}/V_{CH} = register set (1). These 0 and 1 register states are always inverted (active-low) on the normal output pins regardless of the selected output polarity (polarity affects logic function values before register inputs).

* Applies to both 20-pin DIP and 20-lead PCC Packages for GAL16V8.

Register Preload (Continued)



TL/L/9344–17

**The S_{DOUT} output buffer is an open drain output during preload. This pin should be terminated to V_{CC} with a 10 kΩ resistor.

FIGURE 4. Output Register Preload Pinout

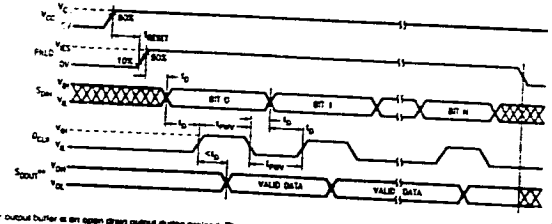
Register Preload Specifications

TABLE III

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{IH}	Input Voltage (High)		2.40		V_{CC}	V
V_{IL}	Input Voltage (Low)		0.00		0.50	V
V_{IES}	Registered Preload Input Voltage		14.5	15	15.5	V
V_{OH}	Output Voltage (High) (Note 1)				V_{CC}	V
V_{OL}	Output Voltage (Low) (Note 1)	$I_{OL} \leq 12$ mA	0.00		0.50	V
I_{IH} / I_{IL}	Input Current (Programming)			±1	±10	µA
I_{OH}	High Level Output Current (Note 1)	$V_{OH} \leq V_{CC}$			10	µA
t_{PWV}	Verify Pulse Width		1	5	10	µs
t_D	Preload Sequence Delay		1	5	10	µs
t_{RESET}	Register Reset Time from Valid V_{CC}				45	µs

Note 1: The S_{DOUT} output buffer is an open drain output. This pin should be terminated to V_{CC} with a 10k resistor.

Register Preload Waveforms

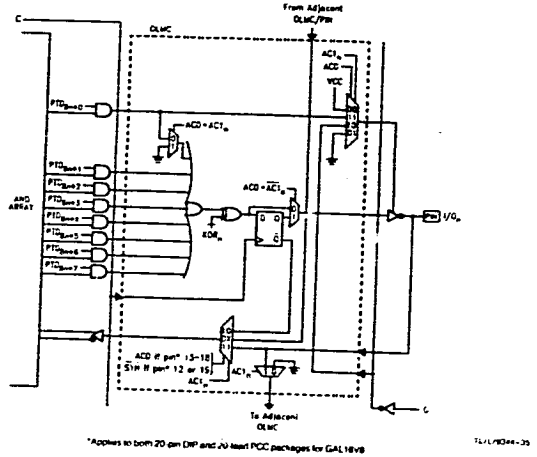


**The S_{DOUT} output buffer is an open drain output during preload. This pin should be terminated to V_{CC} with a 10 kΩ resistor.

TL/L/9344–18

FIGURE 5

OLMC Logic Diagram



*Applies to both 20-pin DIP and 20-lead PCC packages for GAL16V8

TL/L/10444-35

FIGURE 6

OLMC Architecture Programming

TABLE IV

	"Small-PAL" Mode			"Registered-PAL" Mode			"Medium-PAL" Mode	
	Function		JEDEC Input Line #s (Note 1)	Function		JEDEC Input Line #s (Note 1)	Function	JEDEC Input Lines #s (Note 1)
Pin 1	INPUT	INPUT	2,3	CLOCK	CLOCK		INPUT	2,3
*** Pin 19	OUTPUT*	INPUT	6,7	REGISTER	I/O	2,3	TRI-STATE**	
*** Pin 18	OUTPUT*	INPUT	10,11	REGISTER	I/O	6,7	I/O	6,7
*** Pin 17	OUTPUT*	INPUT	14,15	REGISTER	I/O	10,11	I/O	10,11
*** Pin 16	OUTPUT*	NC		REGISTER	I/O	14,15	I/O	14,15
*** Pin 15	OUTPUT*	NC		REGISTER	I/O	18,19	I/O	18,19
*** Pin 14	OUTPUT*	INPUT	18,19	REGISTER	I/O	22,23	I/O	22,23
*** Pin 13	OUTPUT*	INPUT	22,23	REGISTER	I/O	26,27	I/O	26,27
*** Pin 12	OUTPUT*	INPUT	26,27	REGISTER	I/O	30,13	TRI-STATE**	
Pin 11	INPUT	INPUT	30,31	$\overline{G}$	$\overline{G}$		INPUT	30,31
	$AC1_n = 0$	$AC1_n = 1$		$AC1_n = 0$	$AC1_n = 1$		$AC1_n = 1$	
	SYN = 1, AC0 = 0			SYN = 0, AC0 = 1			SYN = 1, AC0 = 1	
	All outputs are combinatorial and always active.			At least one output is registered.			All I/O pins are combinatorial.	

Note: Pin numbers above apply to both 20-pin DIP and 20-lead PCC packages for GAL16V8

Note 1: All even and odd numbered JEDEC input line numbers correspond to true and complement array inputs, respectively.

*Active combinatorial output

**TRI-STATE combinatorial output

*** $AC1_n$ applies to these I/O pins only

Programming Details

Understanding the information in this section is not essential when using approved programming equipment and software for developing GAL designs. This is a more thorough disclosure of the GAL architecture provided for direct JEDEC cell-map editing and diagnostic purposes. This section alone, however, does not contain sufficient information to implement the GAL programming algorithm. If detailed specifications of the GAL programming algorithm are needed, please contact the National Semiconductor Programmable Device Support department.

As mentioned in the Functional Description, the OLMC is responsible for selecting input and/or output paths, registered vs. combinatorial outputs, active-high or low polarity, and common vs. locally-controlled TRI-STATE control. Additionally, the OLMCs select between alternate logic array input paths to maintain JEDEC cell-map compatibility with either "small-PAL" or "medium-PAL" logic arrays.

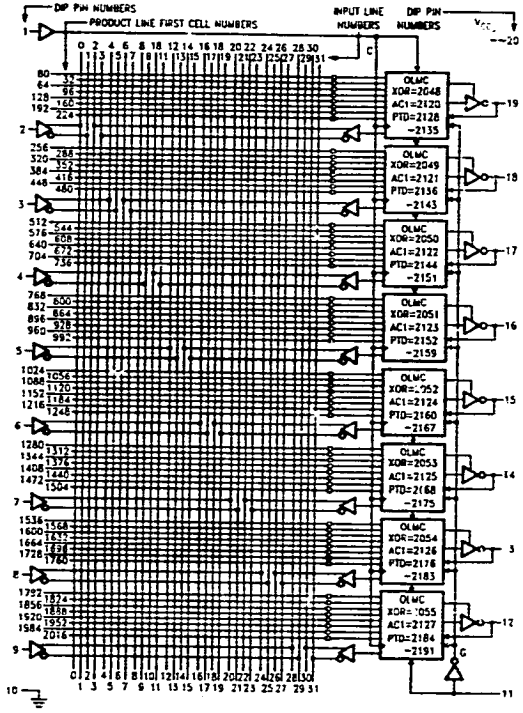
The various configurations of the OLMCs are controlled by a set of programmable "architecture" cells, separate from the logic-defining array cells. Each GAL device contains two "global" architecture cells, "SYN" and "AC0", which affect all OLMCs. Each of the device's eight OLMCs also contains two "local" cells, "AC1" and "XOR". The OLMC Logic Diagram in *Figure 6* shows how the architecture cells select the different paths through the OLMC.

The SYN bit controls whether a device will have any registered outputs (SYN = 0) or will be purely combinatorial (SYN = 1). The SYN bit determines whether device pins* 1 and 11 are used as the clock and global TRI-STATE control inputs (SYN = 0) or whether they are ordinary inputs (SYN = 1). The AC0 bit selects between the "Small-PAL" mode and the "Medium/Registered-PAL" modes. The function of the AC1 bits depend on the state of the AC0 bit. In "Small-PAL" mode (AC0 = 0), the AC1 bit in each OLMC determines whether the associated device pin is an output (AC1 = 0) or an input (AC1 = 1). In "Registered-PAL" mode (AC0 = 1), the AC1 bit determines whether each OLMC is registered (AC1 = 0) or combinatorial (AC1 = 1). In "Medium-PAL" mode (AC0 = 1), the AC1 bits in all OLMCs must be set to 1 (combinatorial). All of the valid architecture bit configurations are shown in the OLMC Architecture table (Table IV), which has the same familiar format used in the OLMC Selection table (Table I).

Independent of SYN, AC0 and the AC1 bits, the XOR bit in each OLMC selects between active-low (XOR = 0) or active-high (XOR = 1) output polarity.

*Applies to both 20-pin DIP and 20-lead PCC packages for GAL16V8

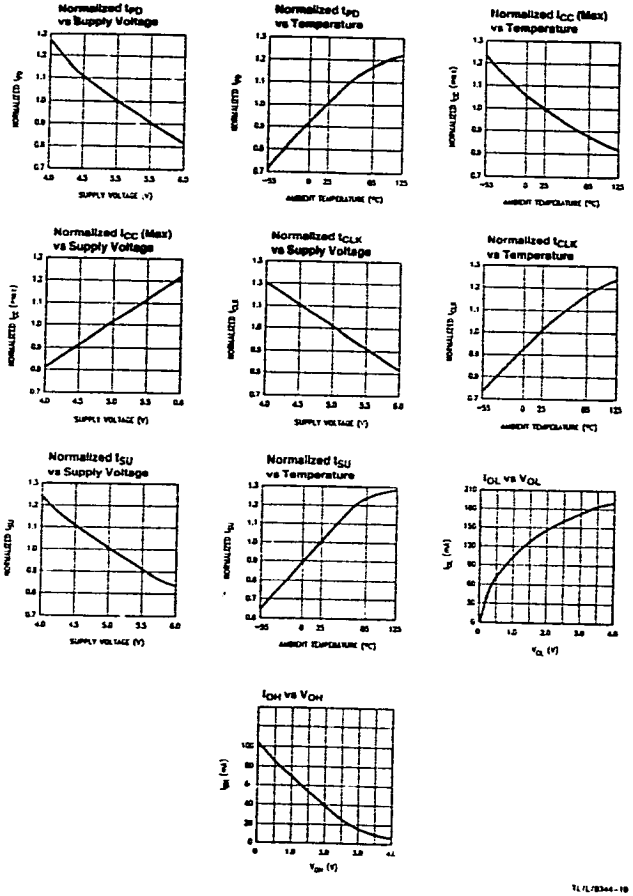
GAL16V8 Logic Diagram



JEDEC Logic Array Cell Number = Product Line First Cell Number + Input Line Number

FIGURE 7

Typical Performance Characteristics



MOTOROLA

SN54/74LS138

1-OF-8-DECODER/ DEMULTIPLEXER

LOW POWER SCHOTTKY

DESCRIPTION — The LSTTL/MSI SN54LS/74LS138 is a high speed 1-of-8 Decoder/Demultiplexer. This device is ideally suited for high speed bipolar memory chip select address decoding. The multiple input enables allow parallel expansion to a 1-of-24 decoder using just three LS138 devices or to a 1-of-32 decoder using four LS138s and one inverter. The LS138 is fabricated with the Schottky barrier diode process for high speed and is completely compatible with all Motorola TTL families.

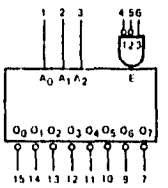
- DEMULTIPLEXING CAPABILITY
- MULTIPLE INPUT ENABLE FOR EASY EXPANSION
- TYPICAL POWER DISSIPATION OF 32 mW
- ACTIVE LOW MUTUALLY EXCLUSIVE OUTPUTS
- INPUT CLAMP DIODES LIMIT HIGH SPEED TERMINATION EFFECTS

PIN NAMES

		LOADING (Note a) HIGH	LOW
$A_0 - A_2$	Address Inputs	0.5 U.L.	0.25 U.L.
$\overline{E}_1, \overline{E}_2$	Enable (Active LOW) Inputs	0.5 U.L.	0.25 U.L.
E_3	Enable (Active HIGH) Input	0.5 U.L.	0.25 U.L.
$\overline{O}_0 - \overline{O}_7$	Active LOW Outputs (Note b)	10 U.L.	5(2.5) U.L.

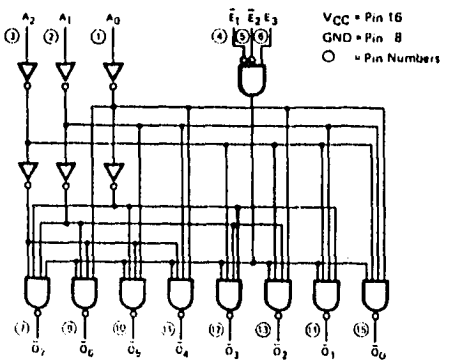
NOTES:
a. 1 TTL Unit Load (U.L.) = 40 μA HIGH/1.6 mA LOW.
b. The Output LOW drive factor is 2.5 U.L. for Military (54) and 5 U.L. for Commercial (74) Temperature Ranges.

LOGIC SYMBOL

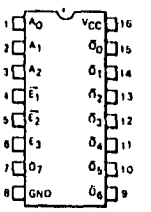


V_{CC} = Pin 16
GND = Pin 8

LOGIC DIAGRAM



CONNECTION DIAGRAM DIP (TOP VIEW)



J Suffix — Case 620-09 (Ceramic)
N Suffix — Case 648-08 (Plastic)

NOTE:
The Flatpak version has the same pinouts (Connection Diagram) as the Dual In-Line Package.

MOTOROLA

SN54/74LS682
SN54/74LS684
SN54/74LS688

8-BIT MAGNITUDE COMPARATORS

LOW POWER SCHOTTKY

DESCRIPTION — The SN54LS/74LS682, 684, 688 are 8-bit magnitude comparators. These device types are designed to perform comparisons between two eight-bit binary or BCD words. All device types provide $\overline{P=Q}$ outputs and the LS682 and LS684 have $\overline{P>Q}$ outputs also.

The LS682, LS684 and LS688 are totem pole devices. The LS682 has a 20 kΩ pullup resistor on the Q inputs for analog or switch data.

TYPE	$\overline{P=Q}$	$\overline{P>Q}$	OUTPUT ENABLE	OUTPUT CONFIGURATION	PULLUP
LS682	yes	yes	no	totem-pole	yes
LS683	yes	yes	no	open-collector	yes
LS684	yes	yes	no	totem-pole	no
LS685	yes	yes	no	open-collector	no
LS686	yes	yes	yes	totem-pole	no
LS687	yes	yes	yes	open-collector	no
LS688	yes	no	yes	totem-pole	no
LS689	yes	no	yes	open-collector	no

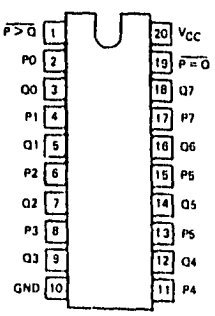
FUNCTION TABLE

INPUTS			OUTPUTS	
DATA	ENABLES			
P, Q	$\overline{G}$, $\overline{GT}$	$\overline{G2}$	$\overline{P=Q}$	$\overline{P>Q}$
P = Q	L	L	L	H
P > Q	L	L	H	L
P < Q	L	L	H	H
X	H	H	H	H

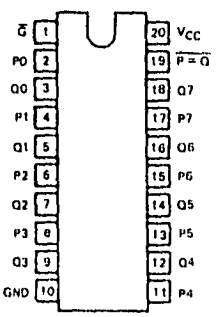
H = high level, L = low level, X = irrelevant

CONNECTION DIAGRAMS
(TOP VIEW)

SN54LS/74LS682/684



SN54LS/74LS688



J Suffix — Case 732-03 (Ceramic)
N Suffix — Case 738-03 (Plastic)

National Semiconductor Corporation

DM54ALS573B/DM74ALS573B Octal D-Type Transparent Latches with TRI-STATE® Outputs

General Description

These 8-bit registers feature totem-pole TRI-STATE outputs designed specifically for driving highly-capacitive or relatively low-impedance loads. The high-impedance state and increased high-logic-level drive provide these registers with the capability of being connected directly to and driving the bus lines in a bus-organized system without need for interface or pull-up components. They are particularly attractive for implementing buffer registers, I/O ports, bidirectional bus drivers, and working registers.

The eight latches of the ALS573B are transparent D-type latches. While the enable (G) is high the Q outputs will follow the data (D) inputs. When the enable is taken low the output will be latched at the level of the data that was set up.

A buffered output control input can be used to place the eight outputs in either a normal logic state (high or low logic levels) or a high-impedance state. In the high-impedance state the outputs neither load nor drive the bus lines significantly.

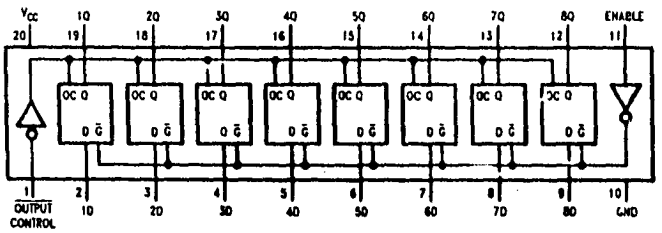
The output control does not affect the internal operation of the latches. That is, the old data can be retained or new data can be entered even while the outputs are off.

Features

- Switching specifications at 50 pF
- Switching specifications guaranteed over full temperature and V_{CC} range
- Advanced oxide-isolated, ion-implanted Schottky TTL process
- Functionally equivalent with LS373
- Improved AC performance over LS373 at approximately half the power
- TRI-STATE buffer-type outputs drive bus lines directly

Connection Diagram

Dual-In-Line Package



TL/F/6226-1

Order Number DM54ALS573BJ, DM74ALS573BWM or DM74ALS573BN
See NS Package Number J20A, M20B or N20A

Function Table

$\overline{\text{Output Control}}$	Enable G	D	Output Q
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

L = Low State, H = High State, X = Don't Care

Z = High Impedance State

Q_0 = Previous Condition of Q

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

- MCS-85™ Compatible 8255A-5
- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with Intel® Microprocessor Families
- Improved Timing Characteristics
- Direct Bit Set/Reset Capability Easing Control Application Interface
- Reduces System Package Count
- Improved DC Driving Capability
- Available in EXPRESS
  - —Standard Temperature Range
  - —Extended Temperature Range

The Intel® 8255A is a general purpose programmable I/O device designed for use with Intel® microprocessors. It has 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. In the first mode (MODE 0), each group of 12 I/O pins may be programmed in sets of 4 to be input or output. In MODE 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining 4 pins, 3 are used for handshaking and interrupt control signals. The third mode of operation (MODE 2) is a bidirectional bus mode which uses 8 lines for a bidirectional bus, and 5 lines, borrowing one from the other group, for handshaking.

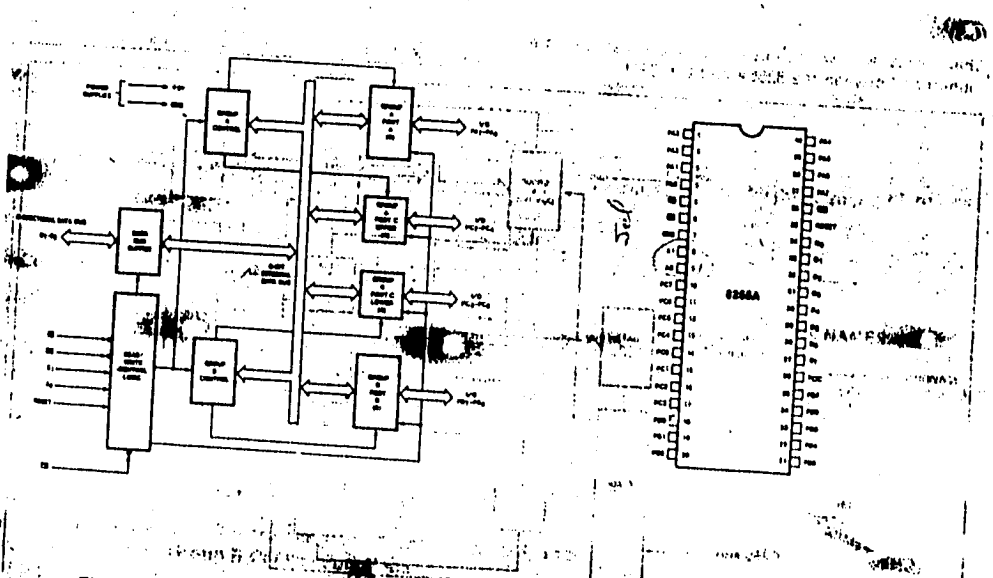


Figure 1. 8255A Block Diagram

Figure 2. Pin Configuration

RESET)

A "high" on this input clears the control register and all ports (A, B, C) are set to the input mode.

Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 8255A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 8255A.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A – Port A and Port C upper (C7-C4)
Control Group B – Port B and Port C lower (C3-C0)

The Control Word Register can Only be written into. No read operation of the Control Word Register is allowed.

Ports A, B, and C

The 8255A contains three 8-bit ports (A, B, and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255A.

Port A. One 8-bit data output latch/buffer and one 8-bit data input latch.

Port B. One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

Port C. One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with ports A and B.

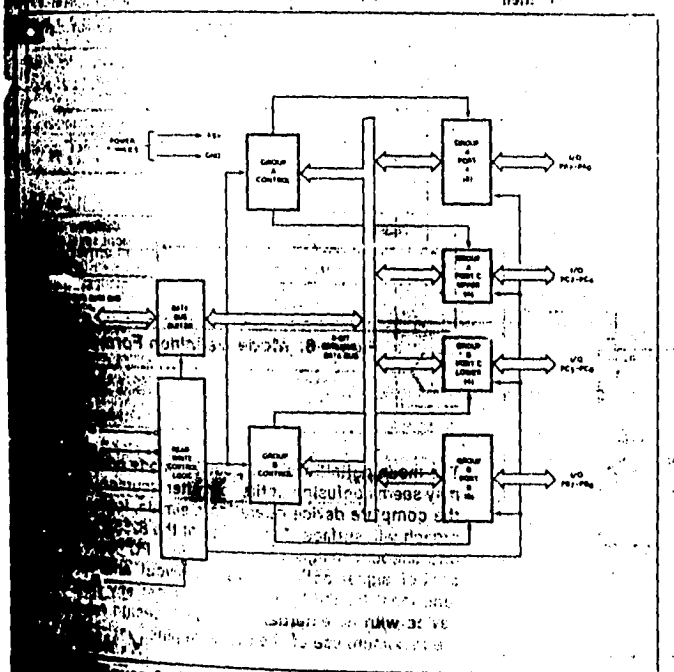
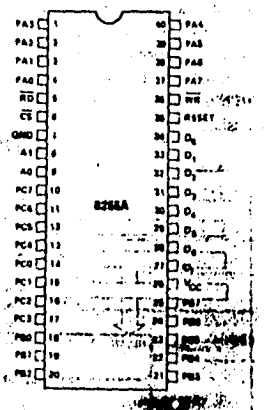


Figure 4. 8225A Block Diagram Showing Group A and Group B Control Functions

PIN CONFIGURATION



PIN NAMES

D_7-D_0	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

8255A FUNCTIONAL DESCRIPTION

General

The 8255A is a programmable peripheral interface (PPI) device designed for use in Intel® microcomputer systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the microcomputer system bus. The functional configuration of the 8255A is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

Data Bus Buffer

This 3-state bidirectional 8-bit buffer is used to interface the 8255A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

($\overline{CS}$)

Chip Select. A "low" on this input pin enables the communication between the 8255A and the CPU.

($\overline{RD}$)

Read. A "low" on this input pin enables the 8255A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255A.

($\overline{WR}$)

Write. A "low" on this input pin enables the CPU to write data or control words into the 8255A.

(A_0 and A_1)

Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word registers. They are normally connected to the least significant bits of the address bus (A_0 and A_1).

8255A BASIC OPERATION

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	PORT A → DATA BUS
0	1	0	1	0	PORT B → DATA BUS
1	0	0	1	0	PORT C → DATA BUS
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	DATA BUS → PORT A
0	1	1	0	0	DATA BUS → PORT B
1	0	1	0	0	DATA BUS → PORT C
1	1	1	0	0	DATA BUS → CONTROL
					DISABLE FUNCTION
X	X	X	X	1	DATA BUS → 3-STATE
1	1	0	1	0	ILLEGAL CONDITION
X	X	1	1	0	DATA BUS → 3-STATE

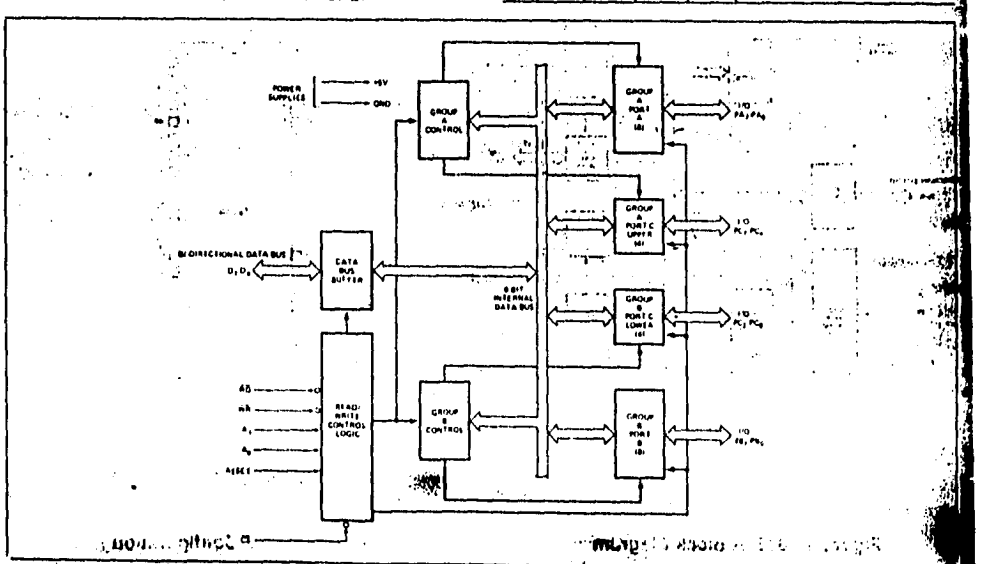


Figure 3. 8255A Block Diagram Showing Data Bus Buffer and Read/Write Control Logic Functions

ANEXO
E

MANUAL DEL USUARIO

SISTEMA DE INTERFÁZ PARA LA CÁMARA DE EXTRAPOLACIÓN DEL PATRÓN SECUNDARIO BETA

MANUAL DEL USUARIO

1 DESCRIPCIÓN DEL SISTEMA

El Sistema de Interfaz para la Cámara de Extrapolación (SICE) es un sistema integrado por varios dispositivos interconectados a una computadora personal (PC), que en conjunto, permite la adquisición de información para la determinación de la dosis absorbida por tejido al estar expuesto a un campo de radiación beta.
El sistema lo componen:

- Cámara de extrapolación.
- Convertidor Analógico Digital **ADC/16 Iotech** con puerto de comunicación IEEE488.
- Módulo **Digital-80** con puerto de comuniación IEEE488.
- Interfaz para la cámara de extrapolación.
- Tarjeta beta.
- Módulo de medición de factores ambientales.
- Electrómetro Keithley Mod. 617.
- Motor de pasos con acoplamiento mecánico.
- Cronómetro Buchler.
- Fuente de voltaje HP mod. 6160A

El ADC convierte la señales analógicas de voltaje entregadas por los transductores de presión, humedad y temperatura ambientales provenientes del módulo de medición, y un voltaje proporcional al voltaje de polarización de la CE; en un código digital que puede ser manejado por la computadora.

El módulo digital proporciona una señal para abrir o cerrar el obturador de la fuente radiactiva.

La interfaz para la cámara de extrapolación contiene la circuitería para el ajuste de la fuente de polarización de la CE y el circuito de potencia para alimentar el motor de pasos.

El electrómetro mide la corriente de ionización de la CE o la carga almacenada en las placas de la misma, según sea el caso.

La fuente de voltaje proporciona la diferencia de potencial para polarizar las placas de la CE.

El cronómetro, genera la señal para abrir o cerrar el obturador de la fuente de radiación y junto con el sistema de seguridad se encarga, en caso de ser necesario de cerrar el obturador para interrumpir la radiación por alguna contingencia.

El programa proporciona el algorítmo de operación del sistema.
La cámara de extrapolación es el dispositivo sensible a la radiación que opera bajo el principio de las cámaras de ionización de gases.
La tarjeta BETA es una tarjeta que se coloca en alguna de las ranuras de expansión que se encuentre disponible y comprende la circutería que envía las señales a la interfaz de la cámara de extrapolación para el ajuste automático de la referencia de la fuente de alimentación y el circuito de potencia que alimenta al motor de pasos, que permite el ajuste automático de la distancia entre placas de la CE.

2 INSTALACIÓN DEL SISTEMA

La instalación del sistema es un procedimiento sencillo que solo requiere de tomar algunas precauciones, lo primero es verificar que contamos con todos los elementos del sistema, la lista de componentes se muestra en el inciso anterior; para la instalación del sistema se necesita la siguiente herramienta:

- desarmador de cruz
- llave Allen de 1/8"
- desarmador plano
- llave española de 1/2"

El arreglo mecánico lo componen: el soporte para el motor y el acoplamiento mecánico entre el tornillo micrométrico y el motor de pasos.
Para montar el soporte del motor, se retiran los ocho tornillos, que sujetan la plataforma superior, se coloca el soporte sobre la plataforma y se fija con tornillos utilizando las mismas oradaciones de donde se retiraron los tornillos, la fig (1) muestra el montaje de la plataforma.

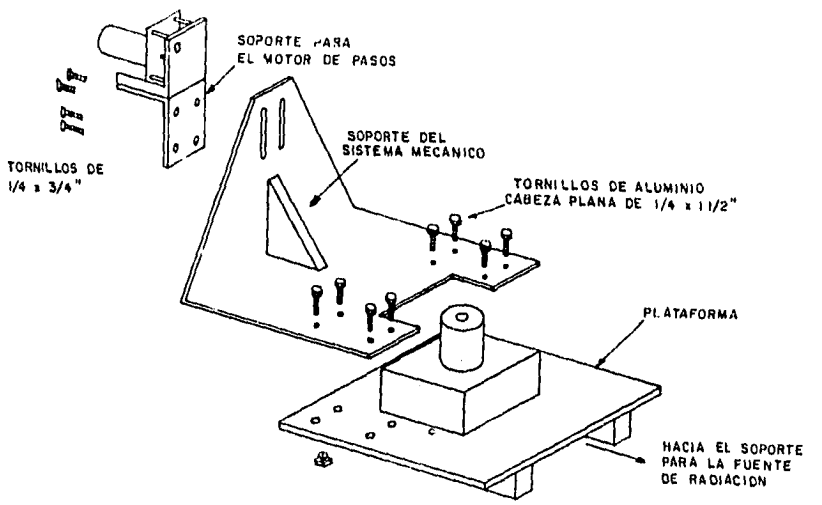


Fig.1 Montaje del soporte en la plataforma .

Una vez fijado el soporte, se procede a fijar el acoplamiento mecánico al tornillo micrométrico de la CE, para esto primero se desplazan las placas de la cámara hasta que se encuentren separadas 1 mm, posteriormente se ajusta la altura de la CE a la altura adecuada, por último se alinea el eje del acoplamiento mecánico con el eje del tornillo micrométrico, para esto, el soporte cuenta con ranura que

permiten mover el motor tanto horizontal como verticalmente, una vez ajustada,se aprietan los tornillos respectivos para fijar el motor, y los opresores para fijar la campana al tornillo micrométrico.

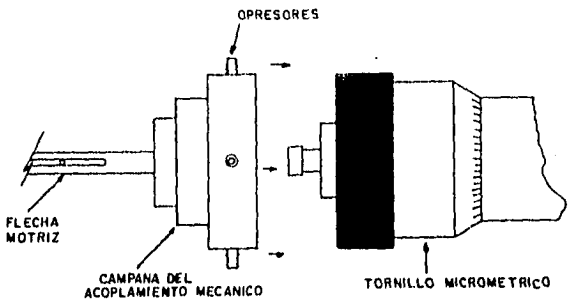


Fig. 2 Fijando el tornillo micrómetrico al acoplamiento mecánico.

2.1 CONEXIÓN DEL SISTEMA

Para hacer la conexión de los instrumentos del sistema, lo primero es asegurarse que todos los instrumentos y la interfaz para la CE se encuentren apagados, el sistema en su parte electrónica se compone de:

- Tarjeta beta
- Interfaz para la CE
- Tarjeta de Interfaz Power IEEE488.
- Convertidor Analógico Digital 16
- Módulo digital 80-I488.
-Electrómetro Keithley 617.
- Fuente de Voltaje HP 6110A.
- Sistema de medición de condiciones ambientales.

1.-La tarjeta beta, se inserta en alguna de las ranuras de expansión de la PC, esta tarjeta se conecta a la interfaz por medio de dos cables, el primero tiene un conectores DB9 en los extremos este se conecta a la interfaz de la cámara de extrapolación en el conector DB9 macho etiquetado con el nómbre de **control del motor de pasos**, el otro cable tiene en sus extremos conectores hembra DB25 etiquetados cada uno como **conthv** y **PC** respectivamente, el conector marcado con **conthv** se conecta a la interfaz de cámara de extrapolación y al conector macho DB25 marcado con la leyenda **control de la fuente de HV**.

2.-La fuente de voltaje se fija a ceros, y se retira el conector de seguridad que se encuentra en el costado derecho de la fuente. Fig. (3).

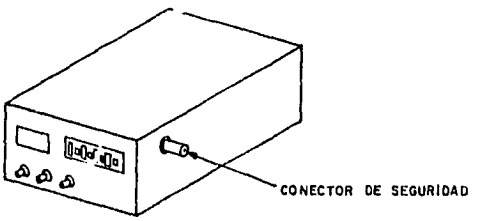


Fig.3 Ubicación del conector de seguridad de la fuente .

y se conecta el cable de la ruta de retroalimentación que tiene en un extremo un conector circular de 2 terminales y en el otro un Conector Circular de Plástico (CPC) de 9 terminales, el conector de 9 terminales se conecta a la interfaz de la cámara de extrapolación a un conector circularde 9 terminales macho, etiquetado con la leyenda **motor de pasos.**

3 El mismo conector CPC de 9 terminales, sirve también para proporcionar energía al motor de pasos, a través de un cable que en su extremo tiene un conector DIN macho de nueve terminales .

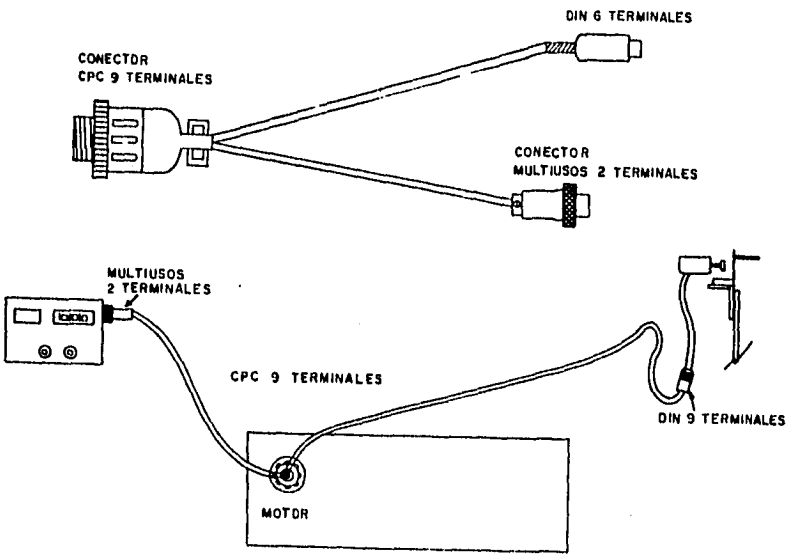


Fig.4 Conexiones del conector CPC-9 .

4. La salida de la fuente de alto voltaje se conecta a la interfaz de la CE como se muestra el la figura.

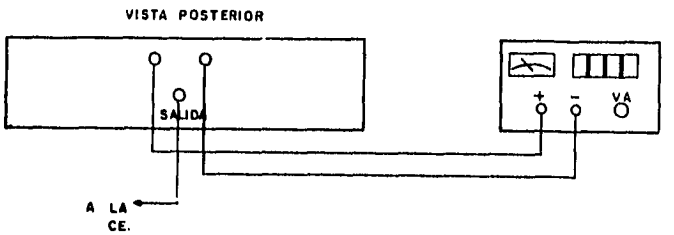


Fig. 5 Conexión de la fuente de alto voltaje .

5. Los dispositivos con puerto IEEE488 (electrómetro, ADC, Módulo digital) se conectan a la tarjeta Power IEEE488 que se encuentra en la PC (si esta tarjeta no está instalada consultar el manual del usuario de la tarjeta Power IEEE488 para su instalación).
6. Del módulo de medición de condiciones ambientales se toman las señales de presión, temperatura y humedad relativa y se conectan al ADC como muestra la Fig.(6)

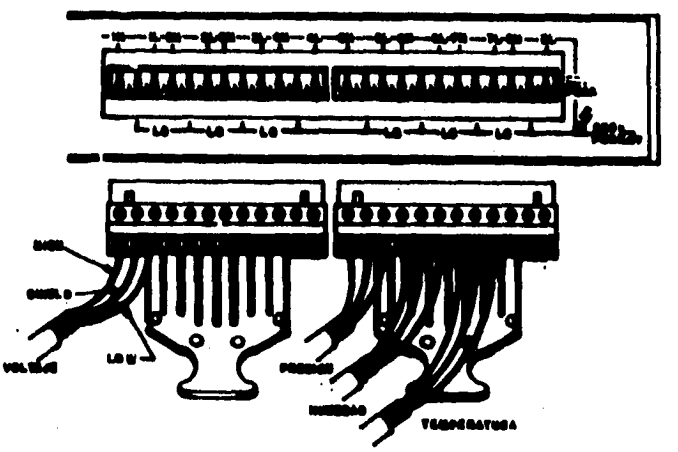


Fig.6 Conexión de las señales de factores ambientales al ADC .

7. La entrada del electrómetro se conecta a la placa de colección de la CE como se muestra en la Fig.7

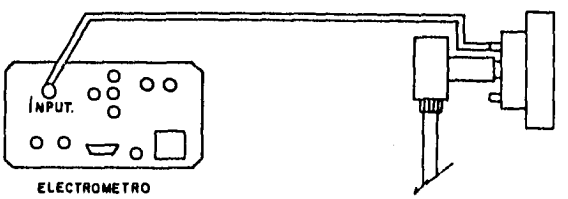


Fig.7 Conexión del electrómetro al electródo de colección de la CE .

8. verifique que todas las conexiones se hayan hecho adecuadamanete, la conexión de todo el sistema se muestra en la fig. 8 .

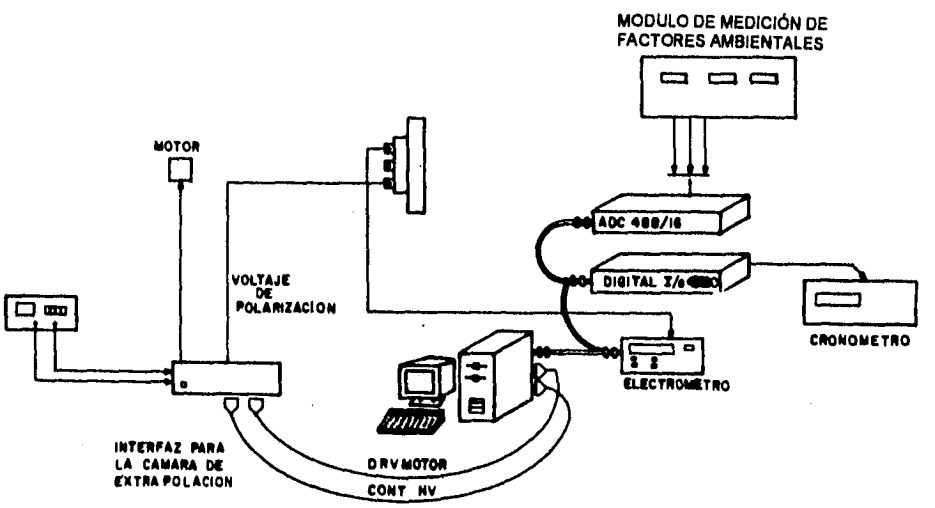


Fig. 8 Configuración del sistema de interfaz para la cámara de extrapolación .

2.2 INSTALACIÓN DE LOS PROGRAMAS

1.- Instale el "driver" para la tarjeta IEEE488. para su instalación consulte el manual del usuario (volumen 1 de la tarjeta power IEEE488).
2.- Copie en el disco duro el programa BETAI488.EXE que viene en el disco del sistema etiquetado como **SISTEMA BETA**.

3 OPERACIÓN DEL SISTEMA

3.1 PRECAUCIONES

Antes de iniciar el proceso de medición y para evitar algun daño en el equipo es necesario tomar algunas precausiones.

1.- Verifique que el cable para la ruta de retroalimentación de la fuente de retroalimentación este colocado en la fuente. **(si la fuente de voltaje es encendida y el lazo de retroalimentación no está cerrado por medio del conector de seguridad o el cable de retroalimentación, el voltaje de la fuente se dispara hasta 3200 V lo que puede dañar la fuente de polarización, componentes de la interfaz para la cámara de extrapolación e incluso dañar la ventana de la cámara de extrapolación).**

2.- Libere el seguro de las placas de la cámara de extrapolación (CE) girando el tornillo del seguro en sentido de las manecillas del reloj.

3.- Revise que todas las conexiones esten en su lugar.

3.2 ENCENDIDO DEL SISTEMA

Para energizar los equipos del sistema siga el siguiente orden de encendido:

1.- Computadora personal.
2.- Fuente de polarización.
3.- Electrómetro, ADC y Módulo digital.
4.- Cronómetro Bouchler.
5.- Interfaz para la cámara de extrapolación.

3.3 OPERACIÓN DEL PROGRAMA

Una vez energizados los equipos del sistema, y colocada la CE y la fuente radiactiva en su lugar, puede iniciar la operación del proceso de medición.
Para llamar a ejecución el programa del sistema, teclee **BETA** y después presione la tecla de **ENTER,** en el monitor aparecerá la siguente pantalla de presentación:

INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES

CENTRO DE METROLOGÍA DE RADIACIONES IONIZANTES

SISTEMA AUTOMÁTICO DE MEDICIÓN

Fig.9 Pantalla de presentación del programa BETA .

Una vez que apareció esta ventana presione la tecla **ENTER** , a continuación aparece una ventana de menú en la que aparecen las siguientes opciones:

A) ADQUIRIR INFORMACIÓN
B) IMPRIMIR INFORMACIÓN
C) AUTOTEST
Q) SALIR

para seleccionar cualquiera de las funciones del menú, tiene 2 formas; una es presionando la tecla de la letra con la que se identifica en el menú, y la segunda, es moviendo la barra iluminada con las flechas del teclado, colocándola en la función que desee seleccionar y presionando la tecla de **ENTER**.
La función **ADQUIRIR INFORMACIÓN** es una rutina para capturar información y guardarla en medios de almacenamiento magnético, la función **IMPRIMIR INFORMACIÓN** toma la información generada por la función **ADQUIRIR INFORMACIÓN** y la convierte a un archivo de tipo texto que puede ser escrito en un archivo en algún medio de almacenamiento magnético, o enviar el archivo a una impresora, la función **AUTOTEST**, tiene dos rutinas, una para calibrar el desplazamiento de las placas de la CE y obtener el área efectiva de las placas, y otra para calibrar la fuente de polarización; si se selecciona salir, el programa cierra todos los archivos que se hallan abierto y sale al sisterma operativo.

3.3.1 FUNCIÓN ADQUIRIR INFORMACIÓN

Al seleccionar la función **ADQUIRIR INFORMACIÓN** aparecen un menú con 3 opciones:

A) CONSTRUCCIÓN DE CURVAS DE EXTRAPOLACIÓN
B) OTRAS APLICACIONES
Q) SALIR

La selección de la función deseada, se hace de la misma forma que en la selección de opciones en el menú principal, la opción **CONSTRUCCIÓN DE CURVAS DE EXTRAPOLACIÓN** es una rutina para la adquisición de información para la determinación de la dosis impartida por una fuente de radiación beta, mientras que la función **OTRAS APLICACIONES**, es una rutina más general en la que los parámetros de operación del sistema pueden ser cambiados durante la ejecución del proceso de medición.

3.3.1.1 CONSTRUCCIÓN DE CURVAS DE EXTRAPOLACIÓN.

Al seleccionar esta opción, aparecera una pantalla en la que se pide al usuario la siguiente información:
NOMBRE DEL ARCHIVO DE TRABAJO: teclee el nombre del archivo en el que se va a guardar la información sin extensión, ya que el sistema pone por omisión la extensión .DAT.
NOMBRE DEL USUARIO: es el nombre de la institución o persona a la que se le está prestando el servicio.
TIPO DE FUENTE: identificación del tipo de fuente que se esta utilizando.
DISTANCIA INICIAL: es la distancia entre placas de la cámara de extrapolación a partir de la cual se va a iniciar el barrido en distancia.
DISTANCIA FINAL: es la distancia entre placas de la cámara de extrapolación en la cual va a terminar el barrido de distancia.
GRADIENTE: es la relación entre el voltaje y la distancia entre placas que se utiliza como parámetro para aplicar el voltaje de polarización a las placas de la CE, dado en V/cm, el valor máximo es de 50 V/cm
NÚMERO DE LECTURAS: es el número de lecturas que toma en cada intervalo de tiempo.
PASO: es el intervalo de distancia que avanza cada vez, hasta llegar a la distancia final.
TIEMPO: es el lapso entre lectura y lectura del electrómetro.
FUNCIÓN E INTERVALO: son parámetros de operación del electrómetro.

Una vez que el usuario da al sistema esta información, aparece un letrero para que se confirme si la información proporcionada es correcta, si existe algún error, presione en este punto la tecla N y proceda a volver a teclear toda la información.
Confirmada la información para el proceso de ejecución, aparece una pantalla para seleccionar cuando se miden factores ambientales, puede escoger entre dos opciones, una, medir factores ambientales cada vez que se tome una lectura, o ,dos, medir factores ambientales cada vez que se cambia la distancia entre placas.
Después de esto se inica el proceso de medición, durante la ejecución del proceso, se muestra una ventana de status que muestra que es lo que está realizando el sistema en el momento, al finalizar el proceso crea un archivo con el nombre que se pidió al principio y con la extensión .DAT terminado el proceso, el programa regresa al menú principal.

3.3.1.2 OTRAS APLICACIONES.

Al seleccionar esta función, el sistema requiere de la misma información que la función **CURVAS DE EXTRAPOLACIÓN**, con la diferencia que en esta función, el usuario puede cambiar el gradiente y la distancia entre placas durante el proceso de medición, cosa que no sucede en la función anterior, ya que una vez fijados estos parámetros no se pueden alterar, esta función, al igual que la función **CURVAS DE EXTRAPOLACIÓN**, genera un archivo con extensión .DAT .

3.3.2 IMPRIMIR INFORMACIÓN

Al seleccionar la función **IMPRIMIR INFORMACIÓN ,** el usuario puede ver la información adquirida por el sistema, esta información se encuentra en archivos con extensión .DAT y para verla solo tiene que teclear el nombre del archivo que desea ver sin poner la extensión, en la pantalla aparecerá la información de esta manera:

INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES
CENTRO DE METROLOGIA DE RADIACIONES IONIZANTES
USUARIO: fernando
TIPO DE FUENTE: sr-90
MEDICION DEL RUIDO DE FONDO
distancia entre placas: 10000voltaje aplicado : 5.000 V

n	lectura [A]
0	2.9270e-14
1	3.3300e-14
2	3.9040e-14
3	3.7170e-14
4	4.0620e-14

promedio: 3.5880e-14 A desv. std.: 4.1101e-15 A

MEDICIONES CON IRRADIACION
distancia entre placas: 10000 micras
presion: 710.000 mbar humedad: 45.000 % temperatura 20.000 C

n	VOLTAJE APLICADO	
	5.000V	-5.000V
	[A]	[A]
0	-3.5442e-13	-9.7333e-12
1	-4.9360e-13	-1.1452e-11
2	-5.6548e-13	-1.1497e-11

3	-6.0158e-13	-1.1473e-11
4	-6.1769e-13	-1.1488e-11
promedio:	-5.2655e-13	-1.1129e-11
desv.srtd :	9.6090e-14	6.9786e-13

La información aparace en la pantalla del monitor, y al final el programa pregunta si desea imprimir la información, si la respuesta es afirmativa, aparece un menú con dos opciones:

a) guardar la información en un archivo en modo texto
b) enviar la información a una impresora

La opción "a" envía la información a un archivo en disco con el nombre del archivo de donde procede la información , y con extensión .ASC, este archivo en modo texto puede ser leído por cualquier paquete estadístico, por ejemplo SAS.
La "b" envía la información a la impresora para tener una copia de la información en papel.

3.3.3 AUTOTEST

La función **AUTOTEST** tiene dos opciones:

A) CALIBRACIÓN DE LA CÁMARA DE EXTRAPOLACIÓN
B) CALIBRACIÓN DE LA FUENTE DE VOLTAJE

La función autotest tiene como propósito verificar la operación del sistema y el correcto funcionamiento del sistema, principalmente del ajuste de la fuente de polarización y del ajuste de la distancia entre placas de la cámara de extrapolación.

3.3.3.1 CALIBRACIÓN DE LA CÁMARA DE EXTRAPOLACIÓN.

Esta es una rutina que calcula el área efectiva de las placas de la cámara de extrapolación, la distancia real entre placas cuando el dial del tornillo micrométrico marca cero y una tabla comparativa de distrancia propuesta contra distancia medida por medio de un método capacitivo, una vez terminado el proceso de medición, el sistema presenta un informe como el que se muestra a continuación:

INSTITUTO NACIONAL DE INVESTIGACIONES NUCLEARES
CENTRO DE METROLOGIA DE RADIACIONES IONIZANTES
CALIBRACION DE LA CAMARA DE EXTRAPOLACIION
recta normalizada: L= -8.5581851e-05 + 6.5680506e-15 Cc-1 [m]
area efectiva = 7.4215262e-04 [m^2]
L real cuando dial marca 0 = -8.5581851e-05 [m]

distancia propuesta [micrometros]	distancia medida [m]	capacitancia [F]
10000	1.00419708e-02	6.54059923e-13
5000	5.16502326e-03	1.27164013e-12
2500	2.61391024e-03	2.51273004e-12
1000	1.06336444e-03	6.17666942e-12
500	5.43640810e-04	1.20815996e-11

al finalizar el programa crea un archivo con extensión .TST de tipo texto donde se muestra la información obtenida.

3.3.2 CALIBRACIÓN DE LA FUENTE DE VOLTAJE

Esta función genera una escalera voltaje, y en cada paso mide el voltaje , generando una tabla de voltaje medido contra voltaje propuesto con esta información se puede determinar si el ajuste de voltaje está funcionando adecuadamente.