UNIVERSIDAD NACIONAL AUTONOMA
DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES
CUAUTITLAN

"DISEÑO DE EQUIPOS DE PRUEBA PARA GRABADORA DE DATOS DRE-5000"

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO MECANICO ELECTRICISTA

P R E S E N T A

PEDRO OCTAVIO DIEZ DE SOLLANO MONTES DE OCA

ASESOR

ING. ANTONIO HERRERA MEJIA

CUAUTITLAN IZCALLI, EDO. DE MEX. 1994

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN
UNIDAD DE LA ADMINISTRACION ESCOLAR
DEPARTAMENTO DE EXAMENES PROFESIONALES

U. N. A. M.
FACULTAD DE ESTUDIOS SUPERIORES-CUAUTITLAN

DEPARTAMENTO DE EXAMENES PROFESIONALES

ASUNTO: VOTOS APROBATORIOS

DR. JAIME KELLER TORRES
DIRECTOR DE LA FES-CUAUTITLAN
P R E S E N T E .

AT'N: Ing. Rafael Rodriguez Caballos
Jefe del Departamento de Exámenes
Profesionales de la F.E.S. - C.

Con base en el art. 28 del Reglamento General de Exámenes, nos permitimos comunicar a usted que revisamos la TESIS TITULADA:

"DISEÑO DE EQUIPOS DE PRUEBA PARA GRABADORA DE DATOS DRE-5000"

que presenta el pasante: PEDRO OCTAVIO DIEZ DE SOLLANO MONTES DE OCA

con número de cuenta: 7105797-2 para obtener el TITULO de:

INGENIERO MECANICO ELECTRICISTA .

Considerando que dicha tesis reúne los requisitos necesarios para ser discutida en el EXAMEN PROFESIONAL correspondiente, otorgamos nuestro VOTO APROBATORIO.

A T E N T A M E N T E .
"POR MI RAZA HABLARA EL ESPIRITU"
Cuautitlán Izcalli, Edo. de Méx., a 10 de ENERO de 1994

PRESIDENTE	ING. ANTONIO HERRERA MEJIA	14/ENE/94
VOCAL	ING. JOSE LUIS RIVERA LOPEZ	13/1/94
SECRETARIO	ING. UBALDO RAMIREZ URIZAR	
PRIMER SUPLENTE	ING. NICOLAS CALVA TAPIA	
SEGUNDO SUPLENTE	ING. JUAN GONZALEZ VEGA	13/ENERO/94

UAE/DEP/VAP/01

A mi esposa: Por su amor y comprensión

A mis hijos: Con cariño

A mi madre: Por su ejemplo

A mis hermanos: Por su apoyo

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN

INGENIERIA MECANICA ELECTRICISTA

TESIS:

"DISEÑO DE UN EQUIPO DE PRUEBA PARA GRABADORA DE DATOS DRE-5000"

DIRECTOR DE TESIS: INGENIERO ANTONIO HERRERA MEJIA

SUSTENTANTE: PEDRO OCTAVIO DIEZ DE SOLLANO MONTES DE OCA

Tabla de Contenidos

CAPITULO 1

Introducción

1.1 La Grabadora de Datos DRE-5000.

La grabadora DRE, serie 5000, es un dispositivo de almacenamiento de datos que almacena y recupera información digital utilizando un cartucho de cinta magnética de un cuarto de pulgada; del tipo 3M DC300A o equivalente. Entre las principales aplicaciones de la grabadora DRE serie 5000 se encuentran:

A).- Captura de datos en tiempo real

B).- Almacenamiento de respaldo para mini y microcomputadoras

C).- Preparación de datos de teclado a cinta

D).- Reemplazo de cintas de papel perforadas

Esta grabadora ha sido diseñada para montaje en panel, con un ángulo de inclinación de 28 grados que permite montar varias grabadoras en un panel, con muy poca distancia entre ellas, economizando espacio de este modo. Además el

acceso para cargar y descargar el cartucho ha sido diseñado ergonómicamente para facilidad de uso; para cargar el cartucho basta insertarlo y cerrar la puerta de la cubierta y automaticamente el cartucho queda alineado con la cabeza de lectura/ escritura y con el capstan. Para descargarlo, simplemente basta con abrir la puerta de la cubierta.

Los datos se graban en forma serial de fase codificada (de la cual se hablara mas ampliamente en la sección de teoría de funcionamiento) en un solo track a una velocidad ya sea de 63 o 31.5 bits por milimetro (1600 u 800 bits/pulgada), resultando en una capacidad teórica máxima del track de 5.6 millones de bits. La DRE-5000 puede obtenerse en dos opciones; dos o cuatro tracks y la selección del track queda bajo control del programa. Para proveer seguridad en los datos, se cuenta con la facilidad de lectura mientras se escribe. La lectura puede llevarse a cabo en cualquier dirección, ya sea hacia adelante (forward) o hacia atrás (reverse), mientras que la escritura solo puede efectuarse cuando hay movimiento hacia adelante (forward). Los datos son transferidos a una velocidad nominal de 48,000 o 24,000 bits por segundo, dependiendo de la densidad que se este usando (63 o 31 bits/mm). Se cuenta también con la facilidad de búsqueda rápida de block a una velocidad de 2.286 metros/segundo (90 pulgadas/segundo) en cualquier dirección, por lo que la longitud total de la cinta puede escudriñarse en menos de 50 segundos.

Los requerimientos de voltaje para alimentación de la grabadora son de +5, +18 y -18 voltios. La DRE-5000 cuenta con sensores para detectar principio de cinta, fin de cinta y advertencias (early warning), así como lógica para impedir correr la cinta mas allá de estas marcas. En la tabla 1 se muestran las especificaciones de la grabadora de datos DRE serie 5000.

TABLA 1.

cartucho	3M tipo DC300A
densidad de empaque	63 bits/mm (1600 bits/in) 31.5 bits/mm.(800 bits/in)
método de grabación	fase codificada
velocidad de la cinta lectura/escritura búsqueda.	 0.762 m/s (30 in/s) 2.286 m/s (90 in/s)
variación de velocidad largo plazo corto plazo	 3% máximo 7% máximo
transferencia de datos 63 bits/mm 31.5 bits/mm	 48 kbit/s + 0.1% 24 kbit/s + 0.1%
tiempo de inicio de escritura (tiempo en el cual la velocidad normal de lectura/escritura es alcanzada después del inicio del comando RUN)	30 milisegundos
tiempo de paro de lectura/escritura	30 milisegundos max.
distancia de comienzo de escritura	15.75 mm max.
distancia de comienzo de lectura	12.45 mm max.
distancia de paro de lect./escr.	11.3 mm max.
tiempo de paro de búsqueda	90 ms max
distancia de paro de búsqueda	127 mm max.
separación entre cabezas de lectura y escritura	3.8 mm
tiempo de selección de track	15 ms max.
dimensiones	250mm x 231mm x178mm
peso máximo	4 kg.

Especificaciones de la grabadora DRE-5000

La grabadora DRE-5000 es utilizada como equipo periférico en las centrales telefónicas digitales tipo AXE que existen en nuestro país. Las funciones que esta grabadora de datos desempeña en AXE serán mencionadas mas adelante.

1.2 Necesidad del Diseño de un Equipo de Prueba para DRE-5000.

Como se mencionó anteriormente, la grabadora DRE-5000 es utilizada en las centrales telefónicas digitales AXE. Para que la administración pueda mantener una alta calidad de servicio a un costo razonable debe contar con Centrales Telefónicas con buenas propiedades operacionales y de mantenimiento. El uso de la técnica SPC en el sistema AXE, junto con sus funciones de operación y mantenimiento permite un manejo simple y efectivo de las Centrales. Estas pueden permanecer sin personal y simplemente interactuar con centros de mantenimiento a través de data links. El trabajo de operación y mantenimiento es efectuado principalmente por comunicación hombre-máquina desde estos centros, a través de dispositivos periféricos, entre los cuales se encuentra incluida la grabadora de datos DRE-5000. Es importante contar con un buen sistema de mantenimiento para estos dispositivos, que asegure su correcto funcionamiento y por ende la comunicación hombre-máquina necesaria para la operación de la central telefónica. Actualmente, las centrales AXE ubicadas en las ciudades de Tijuana, Ensenada, Mexicali y San Luis R.C. cuentan con este tipo de dispositivos como unidades CT, así como también un buen número de centrales AXE ubicadas en las ciudades de México, Puebla, Veracruz y Villahermosa. La necesidad de diseñar un equipo de prueba para la DRE-5000 surge debido a que actualmente no se cuenta con un probador adecuado. Las señales y voltajes son conectadas manualmente, lo cual considerando el número de estas hace el trabajo bastante difícil. La mayoría

de las veces no es posible simularlas todas, especialmente cuando la reparación o calibración tiene que efectuarse fuera del laboratorio. Además, despues de reparar o calibrar una grabadora de datos, es necesario esperar a tener una posición libre en alguna central para poder comprobar el funcionamiento, ocasionando demoras y perdidas de tiempo. El probador aquí diseñado permitirá la prueba de la grabadora de datos sin necesidad de utilizar una posición en la central telefónica. Además, diagnosticará fallas basado en las pruebas y sus resultados. Este equipo será de tipo portátil, para que pueda ser fácilmente transportado a las centrales. Esto es importante, ya que en ocasiones es conveniente reparar y calibrar el equipo en la misma central cuando existe urgencia por tener los dispositivos funcionando. Tambien en el caso de centrales foráneas, usualmente es necesario hacer visitas periodicas a la zona y a cada una de las centrales, y efectuar la calibración y reparación allí mismo. Para el diseño de este equipo de prueba, se ha pensado en aprovechar el creciente uso de las computadoras personales del tipo IBM ó compatible. En laboratorio se cuenta con computadoras personales de este tipo, tanto de escritorio como portátiles. Para poder utilizar una de estas computadoras personales es necesario el contar con circuitos y programas que permitan la interface y el control de la grabadora de datos DRE-5000 desde la computadora. Se ha pensado en utilizar el puerto paralelo de la computadora personal, ya que de esta manera se contará con flexibilidad, y se evitará el depender de una tarjeta en especial y de su instalación en la computadora. Será necesario contar con circuitos de interface entre el puerto paralelo y la grabadora DRE-5000. Más adelante se verá el diseño de estos. Para el desarrollo de los programas necesarios, se decidió utilizar un lenguaje de alto nivel, y uno de bajo nivel. Esto debido a la facilidad que el lenguaje de alto nivel presenta para programar una interface hacia el usuario y

la que el lenguaje de alto nivel presenta para manejar los circuitos, especialmente en los casos en que es necesario proveer señales con requerimientos de tiempo. El lenguaje de alto nivel seleccionado ha sido Pascal, y el lenguaje de bajo nivel será el ensamblador 8088/86, ya que serán utilizadas computadoras personales IBM ó compatibles.

1.3 Las Centrales Telefónicas Digitales AXE y sus Equipos Periféricos.

Las centrales digitales AXE utilizan la técnica SPC (Stored Program Control) o control por programa almacenado y tienen la siguiente estructura funcional.

- Sistemas
- Subsistemas
- Bloques Funcionales
- Unidades Funcionales

Sistemas:

El Sistema AXE ha sido dividido en un Sistema de Conmutación(APT) y un Sistema de Procesamiento de Datos (APZ).

Subsistemas:

Tanto el Sistema de Conmutación como el Sistema de Procesamiento de Datos se subdividen en un número de subsistemas, los cuales incluyen las principales

funciones características dentro del Sistema, tales como la de abonados (SSS), señalización (TSS) o selector de grupo (GSS).

Bloque Funcional:

Cada Subsistema se subdivide en un número de Bloques Funcionales. Las diferentes partes de las cuales consiste un Bloque Funcional son: programas, datos y hardware. Aunque en algunos casos el Bloque Funcional puede consistir solamente de programas y datos y no incluir hardware.

Unidades Funcionales:

Las Unidades Funcionales son las partes constructivas mas pequeñas en la estructura funcional del Sistema AXE. Están formadas por unidades de Hardware y constituyen módulos funcionales, los cuales se combinan a nivel de bloque para construir el hardware total de los bloques individuales. De acuerdo a su función, una Unidad Funcional individual puede ser utilizada como parte de diferentes Bloques Funcionales. Las funciones de las Unidades de Hardware son implementadas con circuitos lógicos en ensambles de tarjetas de circuito impreso.

El Sistema de Procesamiento de Datos (APZ) es utilizado en AXE para controlar al Sistema de Conmutación (APT). Este Sistema (APZ) consiste de cuatro Subsistemas:

A). CPS - Subsistema de Procesador Central

B). CMAS - Subsistema de Mantenimiento

C). IOS - Subsistema de Entrada y Salida

D). RPS - Subsistema de Procesador Regional

En la figura 1 se muestra una ilustración de los sistemas APZ y APT.

Para que el personal de la central pueda interactuar con la central existe un lenguaje hombre-máquina. El Subsistema I/O (IOS) es el que provee las facilidades hombre-máquina y máquina-máquina, donde la información es transferida desde y hacia APZ. Este subsistema es el que controla a la grabadora de datos DRE-5000.

Las funciones provistas por IOS incluyen entrada y salida de datos alfanuméricos a través de terminales de video y teleimpresores, así como entrada y salida de datos orientados en archivos a través de cartuchos magnéticos y cintas magnéticas (como es el caso de la grabadora de datos DRE-5000). También pueden incluirse otros tipos de memorias en esta ultima categoría. El Subsistema IOS también incluye facilidades para que el AXE pueda comunicarse con otras computadoras en cualquier localidad remota. También pueden emplearse modems para operar dispositivos periféricos (I/O) remotos. Todas las funciones de alarma del Sistema están incluidas también en IOS. IOS consiste principalmente de dos Bloques Funcionales:

A).- Bloques de Dispositivos, conteniendo software y hardware

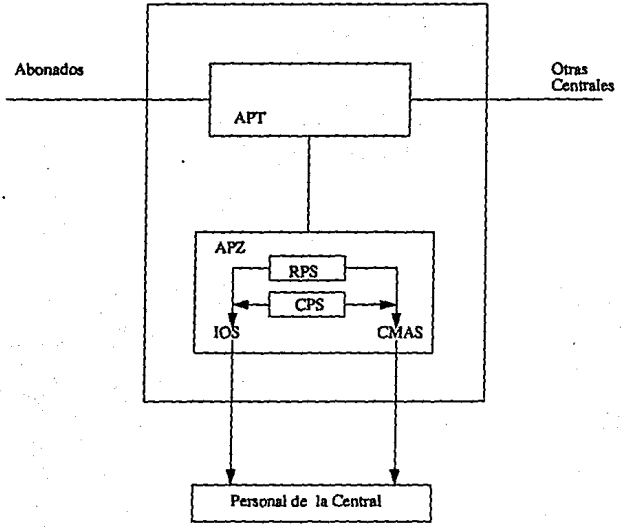
B).- Bloques Funcionales conteniendo solamente software

Los Bloques de Dispositivos disponibles en IOS son para las siguientes funciones:

A).- Teleimpresor (TW)

B).- Display (DH)

FIGURA 1



Sistemas APZ y APT

C).- Cartridge Tapes (CT) (DRE-5000)

D).- Magnetic Tapes (MT)

E).- Data channel (DC)

F).- Alarm Devices (AL,EXAL)

Cada Bloque de Dispositivos consiste de las siguientes unidades:

a).- Una o mas unidades de software central

b).- Una unidad de software regional

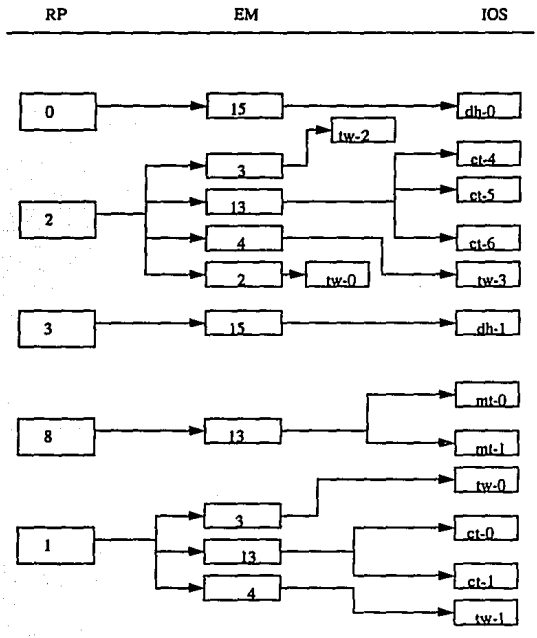
c).- Un interface de hardware (EM)

d).- Un dispositivo (hardware)

Cada dispositivo individual tiene un nombre único designado por el nombre del bloque funcional y el número individual. Por ejemplo CT-1 correspondería a la grabadora de cartucho magnético (DRE-5000) número uno (Cartridge Tape 1).

En la figura 2 se muestra una ilustración de la forma en que se encuentran conectados los dispositivos periféricos a sus respectivos módulos de extensión y procesadores regionales.

FIGURA 2



Conexión de dispositivos IOS en AXE

1.4 Funciones y Comandos Para la DRE-5000 Dentro de una Central AXE.

1.4.1.Funciones de la DRE-5000 dentro de una Central Telefónica Digital AXE.

Los dispositivos periféricos dentro del sistema AXE se clasifican de la siguiente manera:

Dispositivos Alfanuméricos.- Dispositivos que manejan datos para comunicación directa entre hombre-máquina; presentan la información hacia el personal en forma alfanumérica. Por ejemplo un teleimpresor, una terminal de video.

Dispositivos de Archivo.- Estos dispositivos manejan la información mediante almacenamiento de datos en cintas generalmente grabadoras de cartucho magnético o grabadoras de cinta magnética como es el caso de la grabadora DRE-5000. La información no es presentada directamente al personal, pero puede utilizarse un teleimpresor o una terminal de video para leer la información almacenada en los cartuchos o cintas magnéticas.

La grabadora de datos DRE-5000 es un dispositivo de archivo, y la información que almacena es del siguiente tipo:

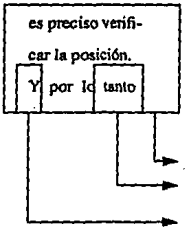
- Información de recarga del Sistema.
- Unidades de Software relocalizables.
- Datos dependientes de la Central.

- Datos de bitácora.

Además de ser un dispositivo de archivo, en ocasiones la grabadora DRE-5000 funciona también como un dispositivo alfanumérico. Esto sucede cuando la información que recibiría normalmente un teleimpresor o una terminal de video se redirige a la DRE-5000 para almacenarla.

La central AXE almacena información en la grabadora de datos DRE-5000 en una estructura que se puede comparar a la correspondiente a un texto. Cuando leemos un texto, notamos que este consiste de un número de enunciados. Cada enunciado consiste de un número de palabras y cada palabra a su vez de un número de caracteres (letras, figuras, espacios, caracteres especiales, etc.). En la figura 3 se muestra esta analogía. La información se estructura en la cinta en un número de bloques de datos, los cuales corresponderían a los enunciados en un texto. Un bloque de datos es la unidad mas pequeña que puede ser manejada por el procesador central; ya sea para lectura o para escritura. Una palabra en un texto correspondería a un "carácter" de información en cinta. Dependiendo del formato usado, dicho carácter puede consistir de ocho o cuatro bits. Ocho bits si se utiliza formato ISO y cuatro se usa formato binario. Un bloque de datos puede estar codificado en cualquiera de los dos formatos y ambos formatos pueden ocurrir en una misma cinta. La correspondencia de un carácter en un texto la encontraríamos en los "ceros" o "unos" en los bits de un bloque de datos. Los ceros y unos pueden considerarse como representados por pequeñas áreas de cinta con una determinada dirección de magnetización para "uno" y la opuesta para "cero".

FIGURA 3



TEXTO	INFORMACION EN CINTA
enunciado	Bloque de Datos
palabra	Caracter
caracter	"0" o "1"

Analogía entre la estructura en un texto y la información en una cinta magnética

El formato binario puede utilizarse en la cinta en los casos en que solo es utilizada información binaria y los caracteres utilizados pueden representarse en forma hexadecimal (0-F)

Ejemplo: "9" en forma hexadecimal queda representado

como 1001.

El formato ISO se utiliza para representación de letras o caracteres especiales. La traducción de caracteres ISO se efectúa de acuerdo al alfabeto CCITT número cinco. En este caso el número nueve quedaría representado por el número 39 hexadecimal (0011 1001 en dígitos binarios).

1.4.2.Comandos Para la DRE-5000 en una Central AXE.

Como se mencionó anteriormente, la comunicación entre el personal y una central AXE se lleva a cabo mediante un lenguaje hombre-máquina, el cual consiste en una serie de comandos y salidas. Un comando es una orden del personal hacia la central, para que realice una función específica. Esta puede ser una orden para cambiar el funcionamiento de la central o para requerir información acerca del estado de la misma. Una salida consiste en un mensaje de la central hacia el personal. El mensaje puede ser una respuesta a un comando o un aviso de un funcionamiento anormal de la central.

A continuación se da un ejemplo de esta comunicación hombre-máquina. Se trata de un caso en el que se ordena a la central ejecutar un borrado de cinta, orden que no puede ser ejecutada por la central debido a una falla en la grabadora en que

debiera llevarse a cabo el borrado, a la cual la central contesta con un mensaje de salida de error.

Comando: IOTSI:IO=CT-4;

Salida: NOT ACCEPTED

FAULT CODE 7

El comando IOTSI significa que hay que hacer un borrado de cinta en un dispositivo IO (IO Tape Scratching). IO=CT-4 designa a la grabadora de cartucho (una DRE-5000) número cuatro como el dispositivo en el cual debe llevarse a cabo el borrado de cinta. El mensaje de salida NOT ACCEPTED significa que el comando no pudo ser efectuado. FAULT CODE 7 da el código de error número siete que diferencia la falla en la grabadora de cartucho de otras posibles fallas que hubieran podido impedir la ejecución del comando. La descripción de cada código de error se encuentra dentro de la información de la biblioteca de la central, bajo el titulo de códigos de errores; en dicha información puede encontrarse que FAULT

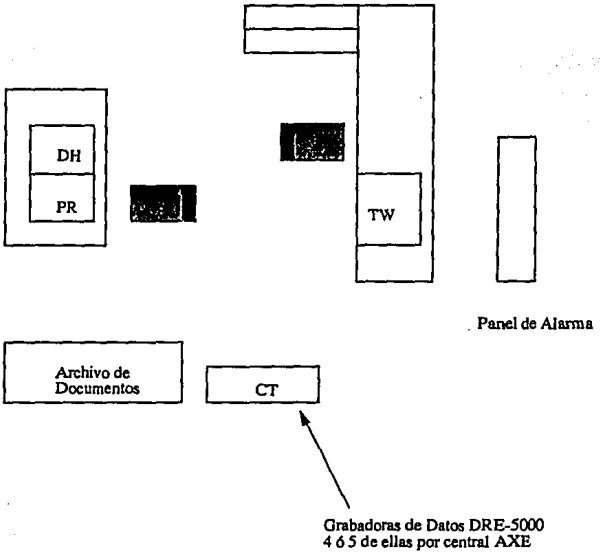
CODE 7 significa que el dispositivo, la grabadora de datos, no funciona correctamente. Este codigo de error es la única referencia a fallas en la grabadora DRE-5000, y no proporciona ningún indicio sobre donde pueda estar localizada la falla dentro de la grabadora de datos.

El envio de mensajes hombre-máquina se efectúa mediante un teleimpresor o una terminal de video. La central puede utilizar además un panel de alarmas para comunicarse con el personal. También en una forma mas indirecta puede llevarse a cabo mediante la grabación de los mensajes en una grabadora de cartucho o de cinta magnética.

En la figura2 se muestra un esquema de un cuarto de control típico de una central telefónica AXE. En dicho cuarto de control se encuentran localizados los dispositivos periféricos necesarios para la comunicación hombre-máquina, entre los que se encuentra la grabadora de datos DRE-5000. Las posiciones DH y PR son para terminal de video y teleimpresor. La tendencia actualmente es a reemplazar estas por computadoras personales del tipo IBM compatible. Mas adelante se verá que este hecho influyó en el diseño del probador para la grabadora de datos DRE-5000.

A continuación se muestran ejemplos de comandos utilizados en la central AXE para los dispositivos de archivo llamados CT y que son grabadoras de cartucho DRE-5000.

FIGURA 4



Ejemplo de un cuarto de control típico de una Central AXE

(Visto desde arriba).

IOTIP.- (Input Output Informante Print) Este comando hace posible obtener una impresión de la información contenida en la cinta de la grabadora de cartucho.

Ejemplo: IOTIP:IO1=CT-1,PRINT=ALL;

IOTSI.- Borrado de cinta. Este comando ordena el borrado de la información contenida en el dispositivo marcado por el parámetro IO.

Ejemplo: IOTSI:IO=CT-4;

Borrado de cinta en grabadora de cartucho número cuatro.

IOTXP.- Permite la impresión de texto en la cinta magnética

Ejemplo: IOTXP:THIS TEXT IS PRINTED;

IOTPT.- Copiado de cinta.

IOCMI.- Iniciación de lectura de comandos desde una cinta.

Ejemplo: IOCMI:IO=CT-4,PROC=X;

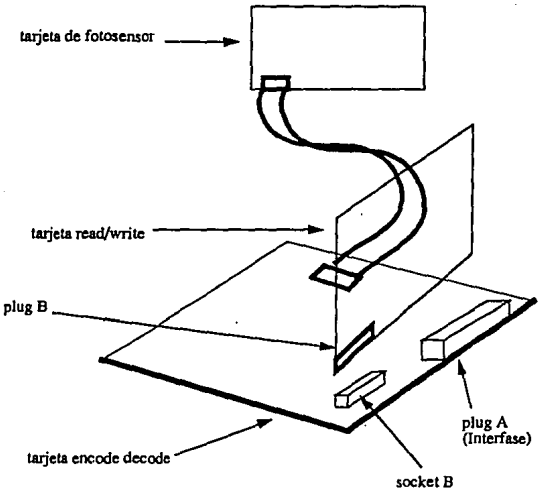
CAPITULO 2 Teoría de Funcionamiento de la DRE-5000

Los circuitos electrónicos de la grabadora de datos DRE-5000 se encuentran distribuidos en tres tarjetas de circuito impreso y un disipador de calor. Las funciones de cada una de las tarjetas de circuito impreso sera descrita a continuación. La interconexión de dichas tarjetas se muestra en la figura 5.

2.1 Tarjeta de Circuito Impreso del Fotosensor (Photosense PCB).

Dos fototransistores montados en esta tarjeta detectan perforaciones en la cinta. La señal de estos transistores es amplificada y alimentada a compuertas TTL para proveer pulsos de duración nominal entre 1.5 microsegundos y 10 microsegundos

FIGURA 5



Interconexión de tarjetas de circuito impreso

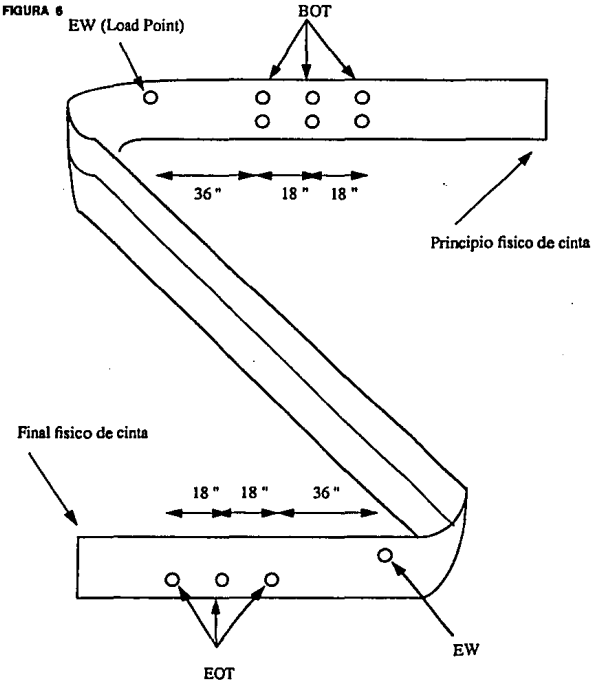
que son alimentados a la tarjeta de circuito impreso servo encode/decode para su decodificación. Las perforaciones en la cinta existen para señalar el principio de cinta BOT (beginning of tape); el fin de cinta EOT (end of tape); el punto de carga LP (load point), que es el punto donde puede comenzarse la grabación de los datos con la seguridad de que la cinta ha sido recorrida lo suficiente para garantizar que no habrá perdida de datos, y por ultimo la perforación de aviso temprano EW (early warning) que marca el lugar donde debe suspenderse la grabación de datos, por estar la cinta próxima a terminarse. En la figura 6 puede apreciarse el modo en que estas perforaciones aparecen en la cinta. Además de las funciones anteriormente descritas, la tarjeta del fotoconductor contiene los switches que dan origen a las señales de cartucho cargado (cartridge loaded) y datos seguros (data safe).

2.2 Tarjeta de Circuito Impreso Servo Encode/Decode.

Esta tarjeta contiene toda la electrónica para el control del motor, exceptuando los transistores de potencia, los cuales van montados en el disipador de calor. El control del motor incluye todas las características requeridas como son:

- manejo de la cinta hacia adelante
- manejo de la cinta en reversa
- velocidad de lectura y escritura
- velocidad de búsqueda (en ambas direcciones)
- periodos de aceleración y desaceleración

FIGURA 6



Perforaciones en la cinta para EW, EOT y BOT

El movimiento de la cinta queda bajo un completo control servo para asegurar que el manejo de la misma sea suave y mantenga la operación de los cartuchos libre de problemas. Esta tarjeta cuenta para tal efecto con un servo amplificador que acepta las señales provenientes de un decodificador de comandos y velocidad requerida para controlar el circuito de manejo del capstan del motor (capstan-motor-drive circuit).

El motor cuenta con un transductor (TACHO) que provee una señal de retroalimentación al servo-amplificador. Otra de las funciones de esta tarjeta consiste en proveer la lógica para codificación y decodificación (encode/decode logic). Para escritura de datos la circuitería recibe las señales write data y write strobe provenientes de la interface y las codifica en señal modulada en fase para suministrarla al amplificador de escritura que se encuentra localizado en la tarjeta de circuito impreso de lectura/escritura. Para lectura de datos recibe la señal amplificada del detector de cruce de la tarjeta de lectura/escritura y decodifica la señal (modulada en fase) en pulsos read data y read strobe, los cuales son presentados en forma separada en dos líneas diferentes a la salida de la interface de la grabadora de datos.

Esta tarjeta cuenta también con la lógica para la decodificación de los pulsos correspondientes a las marcas de comienzo de cinta, fin de cinta y advertencias tempranas. Las señales de las perforaciones marcadas en la cinta y detectadas por la tarjeta del fotosensor son aquí decodificadas y presentadas a la salida de la interface, disponibles para el equipo del usuario (en este caso la central telefónica AXE). Las señales principio de cinta y fin de cinta son utilizadas también por medio de compuertas lógicas para detener al motor y evitar de este modo que la

cinta sea recorrida mas allá del principio o fin de cinta físico, lo que ocasionaría que esta fuera zafada del carrete transportador.

2.3 Tarjeta de Circuito Impreso READ/WRITE.

Esta tarjeta cuenta con dos amplificadores; uno para lectura y otro para escritura. El amplificador de lectura acepta la señal proveniente de la cabeza de lectura, detecta los picos y provee una señal de salida compatible con TTL que es presentada a la tarjeta Servo Encode/Decode y a la salida de la interface de la grabadora de datos.

El amplificador puede ser conectado a cualquiera de los cuatro tracks utilizando una o dos de las líneas de selección de track disponibles en la interface. Estas líneas de selección de track también son utilizadas para seleccionar los tracks de escritura. La salida del amplificador de lectura también puede utilizarse para comprobación de los datos escritos durante una operación de escritura. El amplificador de escritura acepta la información proveniente de la tarjeta de circuito impreso Servo Encode/Decode y su salida es aplicada a la cabeza de escritura; la cual es conectada al track apropiado por medio de la selección efectuada en las líneas de tracks en la interface. En la figura 7 se muestra el diagrama a bloques del funcionamiento de la grabadora de datos DRE-5000.

FIGURA 7

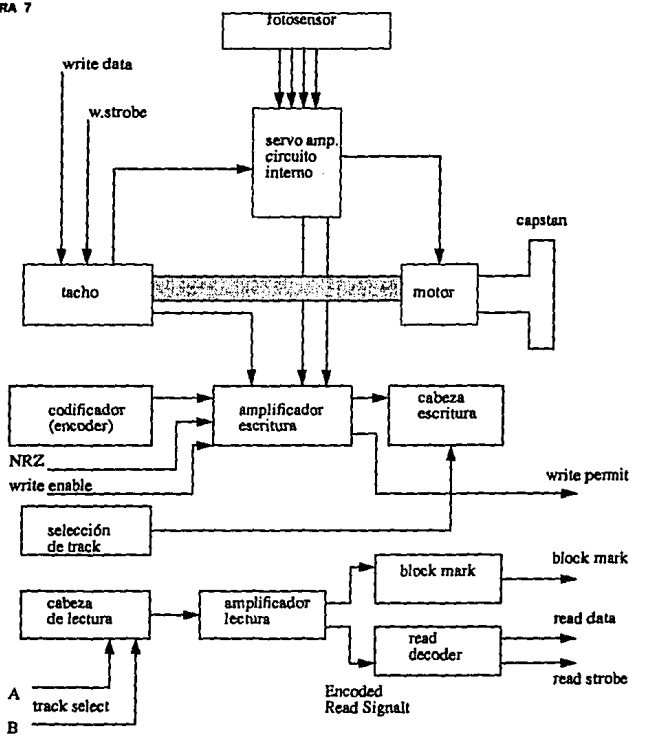


Diagrama de Bloques del funcionamiento de la DRE-5000

2.4 Interfase de la Grabadora DRE-5000.

La interface de la grabadora DRE-5000 se encuentra disponible en el plug A de la tarjeta servo-encode/decode. En la tabla 3 se da un sumario de las señales de dicha interface. La descripción de las Señales de Interface de la DRE-5000 se da a continucación.

Señales de Control Para movimiento de la Cinta.

Comando RUN.- Cuando esta línea se hace verdadera (nivel lógico bajo o LO), el movimiento de la cinta se inicia siempre y cuando a su vez la señal READY sea verdadera. La velocidad y dirección de la cinta dependen de las señales search y reverse. El movimiento de la cinta se detiene cuando cualquiera de los siguientes casos ocurre:

a) El comando run cambia de estado (verdadero a falso).

b) La señal ready se hace falsa.

c) La señal beginning of tape se hace verdadera durante el movimiento de la cinta en reversa.

d) La señal end of tape se hace verdadera mientras la cinta se mueve hacia adelante.

Comando REVERSE.- El estado de esta línea determina la dirección del movimiento de la cinta. Cuando falso, el movimiento de la cinta es hacia adelante; cuando verdadero, la cinta se mueve en reversa.

Comando Search.- Determina la velocidad de la cinta. Cuando falso, la cinta corre a velocidad de lectura/escritura. Cuando verdadero, la cinta corre a velocidad de

búsqueda. Siempre y cuando, para cualquiera de los dos casos la señal RUN sea verdadera.

Señales de Control de Datos.

Señal WRITE ENABLE.- Esta línea es efectiva cuando:

a) El comando Search es falso.

b) La señal Ready es verdadera.

c) La señal Write Permit es verdadera.

Señal WRITE DATA ONE.- El estado de esta señal determina si un "1" o un "0" sera escrito cuando ocurre la transición de la señal Write Strobe de falsa a verdadera (lo-hi). Un "1" sera escrito si esta señal se encuentra en estado alto y un "0" si se encuentra en estado bajo. Esta señal es efectiva solo si:

a) La señal Write Enable es verdadera.

b) El comando Search es falso.

c) La señal Ready es verdadera.

d) La señal Write Permit es verdadera.

Señal WRITE STROBE.- Esta señal permite la escritura del dato indicado por la señal Write Data One cuando ocurre una transición de subida en esta línea.

Señales TRACK SELECT.- Las líneas Track Select A y Track Select B determinan el track que sera utilizado, de la manera como se muestra en la tabla 2.

Señal READ DATA.- El estado de esta línea indica si el ultimo bit de datos leido es "1" o "0". El estado correspondiente a cada uno varia segun la dirección de la cinta.

Cuando se lee hacia adelante:

Read Data se encuentra en estado bajo para "0"

Read Data se encuentra en estado alto para "1"

Cuando se lee en reversa:

Read Data se encuentra en estado alto para "0"

Read Data se encuentra en estado bajo para "1"

Señal READ STROBE.- El propósito de esta línea es definir el tiempo en el cual la señal read data debe ser muestreada. Este muestreo se efectúa durante la transición de estado bajo a estado alto de esta línea.

Señal BLOCK MARK.- Esta señal se utiliza para determinar si las transiciones de flujo en la cabeza de lectura están siendo detectadas.

Señal NRZ.- Esta señal no es utilizada en fase codificada.

TABLA 2.

Track Select A	HI	LO	HI	LO
Track Select B	HI	HI	LO	LO
Track Seleccionado	1	2	3	4

Tabla de Valores Para Selección de Tracks

Señales de Status.

Señal READY.- Esta línea es verdadera solo cuando se dan las siguientes condiciones:

-Los voltajes +5V, +18V y -18V están presentes.

-El cartucho esta correctamente cargado.

-El bulbo para el sensado de principio de cinta, fin de cinta y advertencias tempranas se encuentra encendido.

-Cuando la cinta se encuentra presente en el ensamble del fotosensor.

Señal WRITE PERMIT.- Esta señal sera falsa cuando el cartucho de cinta magnética se encuentre protegido contra escritura.

Señal BOT.- Esta línea proporcionara un pulso cuando la marca de fin de cinta sea detectada. La duración del pulso se encuentra entre 0.5 y 1.5 mS.

Señal EOT.- Esta línea provee un pulso con duración mínima de 10 uS. cuando la marca de fin de cinta es detectada.

Señal EW.- Esta señal (EARLY WARNING) proporciona un pulso de duración mínima de 10 uS. cuando la marca del punto de carga o la de advertencia temprana son detectadas.

TABLA 3.

pin número	señal	nivel lógico	entrada a salida de
1	+ 18 V. (estabilizador)	-	entrada cd
2	- 18V. (estabilizador)	-	entrada cd
3			
4	comando REVERSE	lo	entrada
5	comando SEARCH	lo	entrada
6	READY	lo	salida
7	marca de EOT	pulso (hi)	salida
8	marca de EW	pulso (hi)	salida
9	marca de BOT	pulso (hi)	salida
10	+ 18 V (al motor)	-	entrada cd
11	- 18 V (al motor)	-	entrada cd
12	0 V	-	regreso cd
13	0 V	-	regreso cd
14	comando RUN	lo	entrada
15	BLOCK MARK	lo	salida
16	+ 5 V	-	entrada cd
17	WRITE DATA ONE	hi	entrada
18			
19			
20	WRITE STROBE	transición de subida	entrada
21	READ DATA	lo=0, hi=1 hi=0, lo=1 (reversa)	salida
22	READ STROBE	transición de subida	salida
23	conexión interna	-	-
24	selección de TRACK B	lo - tracks 2 6 4	entrada
25	read NRZ	-	salida
26	selección de TRACK A	lo - tracks 3 6 4	entrada
27	WRITE PERMIT	lo	salida
28	WRITE ENABLE	lo	entrada
29	write NRZ	-	entrada

Señales de la Interface de la DRE-5000

CARACTERISTICAS DE LA INTERFASE DE LA DRE-5000.

El conector para la interface de la DRE-5000 se encuentra fijado a la tarjeta SED (Servo Encode/Decode) de la grabadora y consiste en un plug con 29 líneas o pins del tipo macho 1-582390-3 de AMP. El cable de interface que se acopla a este conector es del tipo 1-582389-3 de AMP, con sockets hembras. Este conector se encuentra ubicado de tal forma que pueda accesarse fácilmente desde la parte trasera de la grabadora. En la tabla 3 se encuentra la relación de las señales correspondientes a cada línea, así como sus niveles lógicos cuando estas son verdaderas.

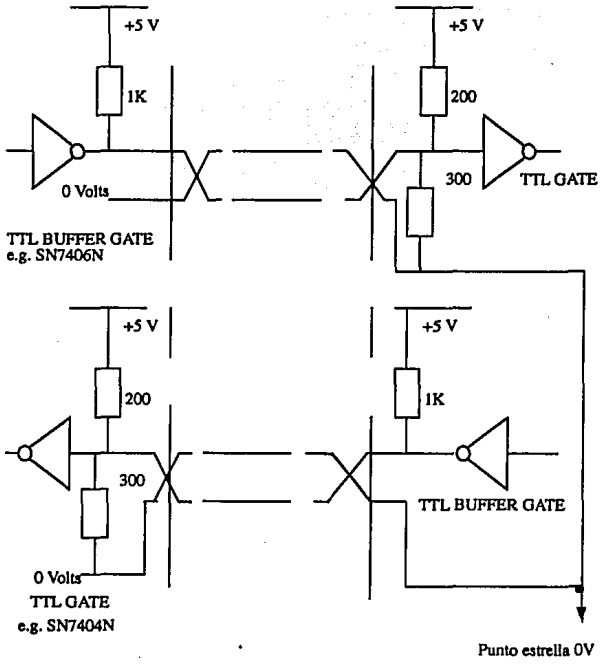
CARACTERISTICAS DE LINEA.

Cada una de las líneas de interface es manejada por un circuito TTL sn 7406 de colector abierto, con una resistencia de pull-up de 1k ohm conectada a +5V. Cada línea esta terminada con una resistencia de 200 ohms conectada a +5V y una de 300 ohms conectada a 0V y el equivalente a una unidad de carga de una compuerta TTL. Este tipo de arreglo se muestra en la figura 8. Cada señal de la interface es transmitida a través de un par torcido de alambres, con menos de 3.5 metros de longitud y con impedancia característica de 100 ohms.

NIVELES LOGICOS.

Los niveles lógicos para las señales de la interface son compatibles con los niveles de la familia lógica TTL y se han definido con los siguientes valores nominales:

FIGURA 8



Circuito de línea para interface

lo = 0.4 voltios

hi = 3.0 voltios

Las señales de la interface se consideran falsas o verdaderas de acuerdo a estos niveles lógicos, sin embargo no todas ellas tienen el mismo nivel lógico cuando son verdaderas o falsas. En la tabla 3 se encuentra el nivel lógico correspondiente a cada señal cuando esta es verdadera.

CAPITULO 3 Diseño de Los Circuitos.

En la tabla 3 se encuentra la descripción de todas las señales de entrada y salida para la interface de la DRE-5000. Dichas señales han sido tomadas como base para el diseño del probador, el cual deberá suministrar las señales de entrada necesarias y comprobar la existencia y exactitud de las señales de salida correspondientes. Como se vio en el capítulo cuatro, las señales de la interface de la DRE-5000 pueden agruparse del siguiente modo:

1.- Señales de Control para Movimiento de la Cinta.

2.- Señales de Control de Datos.

3.- Señales de Status.

Además de las señales mencionadas anteriormente existen líneas para el suministro de voltajes de alimentación. Existen también líneas que no son utilizadas para grabación en formato de fase codificada, y por lo tanto no serán tomadas en cuenta para el diseño del equipo de prueba. El equipo de prueba contara con circuitos para monitorear y checar las señales generadas por la DRE-5000, así como también simulara las señales que normalmente son generadas por un procesador para controlar la operación de la DRE-5000. Además se incluirá en el diseño un circuito de protección que evitara la presencia en la grabadora de los voltajes +18V y -18V mientras no se encuentre presente el suministro de +5V. Como podrá notarse, las tareas que el equipo de prueba deberá desempeñar son complicadas, por lo que considerando que en el laboratorio se cuenta con computadoras personales de tipo IBM compatible,tanto de mesa como portátiles, y que pronto todas las centrales AXE contaran también con computadoras semejantes, se decidió hacer el diseño del equipo de prueba de manera que pueda ser controlado por este tipo de computadoras. Por esta razón, el diseño se dividirá en dos partes; la primera que sera concerniente a el diseño de los circuitos necesarios tanto para control de la la grabadora de datos, como para su interface con la computadora personal. La segunda parte sera concerniente a el diseño de los programas necesarios para que la computadora pueda manejar a la grabadora de datos.

3.1 Circuitos de Interfase.

La interface entre la grabadora de datos y la computadora consistirá básicamente en circuitos que hagan posible la comunicación de las señales entre una y otra, para que de este modo, la computadora pueda manejar a la DRE-5000 mediante el uso

de programas. En la tabla 3 se puede observar que la interface de la grabadora de datos necesita 8 señales de entrada y 8 de salida (descontando las señales NRZ no utilizadas, así como las de c.d., proporcionadas por la fuente de voltaje). Para evitar hacer el probador dependiente de alguna computadora o de algún tipo especial de tarjeta de interface se decidió aprovechar el puerto paralelo existente en la computadora personal, ya que todas las máquinas cuentan con uno, el cual es idéntico en cualquier IBM o compatible. De este modo se evita el tener que contar con una tarjeta especial en cada computadora, o el tener que insertar dicha tarjeta cuando sea necesario. Además, la utilización de este puerto permite economizar tanto en circuitos elctrónicos como en dinero. El puerto paralelo de la computadora personal IBM cuenta con un puerto de salida de 8 bits, un puerto de entrada/salida de 4 bits, y un puerto de 4 bits de entrada. Por lo que en total puede contarse con hasta 12 bits de salida y 4 de entrada, o 8 de salida y 8 de entrada, o una combinación. En este caso se utilizaran 8 de entrada y 8 de salida (respectivamente correspondientes a salida y entrada en la DRE-5000), siendo exactamente lo necesitado. En las tablas 4, 5 y 6, se muestran los valores lógicos y asignación de pins correspondientes a el puerto paralelo de la computadora personal IBM o compatible, los cuales junto con los de la DRE-5000 serán utilizados como base para el diseño tanto de los circuitos como de los programas. Como puede notarse en las tablas, tanto los pins del puerto paralelo en la computadora personal IBM o compatible, como los pins de la interface de la grabadora DRE-5000 funcionan con niveles TTL.

TABLA 4.

BIT	dirección Hex 0378 Nivel TTL con "1" escrito al Puerto.	Pin en el conector DB-25
0	ALTO	2
1	ALTO	3
2	ALTO	4
3	ALTO	5
4	ALTO	6
5	ALTO	7
6	ALTO	8
7	ALTO	9

Pins Para el Puerto de 8 Bits de Salida

TABLA 5.

Bit	Dirección H637A Nivel TTL con "1" escrito	Dirección H637A Nivel TTL con "1" leído	Pin en el conector DB-25
0	Bajo	Bajo	1
1	Bajo	Bajo	14
2	Bajo	Alto	16
3	Bajo	Bajo	17
4	Ena IRQ7	Ena IRQ7	-
5	No utilizado	No utilizado	-
6	No utilizado	No utilizado	-
7	No utilizado	No utilizado	-

Pins Para el Puerto de 4 Bits de Entrada/Salida

TABLA 6.

Bit	Dirección H0079 Nivel TTL con "1" leído	Pin en conector DB-25
0	-	-
1	-	-
2	-	-
3	-	-
4	ALTO	13
5	ALTO	12
6	ALTO	10
7	BAJO	11

Pins Puerto de 4 bits de Entrada

Como se menciono anteriormente, se utilizarán 8 bits de salida y 8 de entrada del puerto paralelo de la computadora personal, asignados a las señales correspondientes en la DRE-5000 como se muestra en la tablas 7 y 8. Una vez hecha la asignación de pins conforme a estas tablas, puede trabajarse en el diseño del circuito que ha de funcionar como interface entre el puerto de la computadora y la grabadora DRE-5000. Anteriormente se vio (sección 2.4) el tipo de requerimientos y características de línea de interface para la grabadora de datos. Los requerimientos son los siguientes:

1).- Cada línea de interface deberá ser manejada por una compuerta TTL de colector abierto SN7406N (buffer gate) con una resistencia de pull-up de 1K conectada a +5 Voltios.

2).- Cada línea debe ser terminada en el extremo de recepción con una resistencia de 200 ohms conectada a +5 Voltios, una resistencia de 300 ohms conectada a 0 Voltios, y con una compuerta SN7404N (gate) equivalente a una unidad de carga estandar TTL.

3).- El cable debe ser tipo par trenzado, con una impedancia característica de aproximadamente 100 ohms, y no deberá de exceder una longitud de 3.5 metros.

El puerto paralelo de las computadoras personales IBM y compatibles trabaja con niveles TTL, por lo que no presenta ningún problema para trabajar con los circuitos SN7406N y SN7404N. Siendo así, para cumplir con los demás requerimientos y características de línea de la DRE-5000, el circuito de interface estará dividido en

TABLA 7.

Pin de Puerto Paralelo	Pin y Señal en DRE-5000
2	5 (SEARCH)
3	14 (RUN)
4	17 (WRITE DATA ONE)
5	20 (WRITE STROBE)
6	24 (SEL.TRACK B)
7	26 (SEL.TRACK A)
8	28 (WRITE ENABLE)
9	4 (REVERSE)

Asignación de Señales de Entrada DRE-5000 a Pins de Puerto Paralelo

TABLA 8.

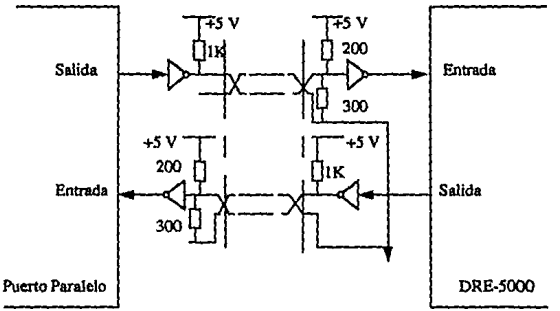
Pin de Puerto Paralelo	Pin y Señal en DRE-5000
1	27 (WRITE PERMIT)
10	6 (READY)
11	7 (EOT)
12	8 (EW)
13	9 (BOT)
14	15 (BLOCK MARK)
16	21 (READ DATA)
17	22 (READ STROBE)

Asignación de Señales de Salida de DRE-5000 a Pins de Puerto Paralelo

dos partes; una tarjeta de circuito impreso conectada a un plug hembra del tipo 1-582389-3 de AMP, con 8 compuertas TTL SN7404N para las señales de entrada, y ocho compuertas TTL SN7406N para las señales de salida. En esta tarjeta contara también con unos conectores para los voltajes de la fuente de poder, así como el circuito de protección que sera discutido mas adelante. Esta tarjeta se conectara por un lado directamente al conector macho tipo 1-582390-3 de AMP en la DRE-5000, y por el otro lado estará conectada por medio de un cable de 20 hilos (par trenzado) con impedancia de 100 ohms a otra tarjeta de circuito impreso con 8 compuertas TTL SN7406N para las señales de salida y 8 compuertas SN7404N para las señales de entrada, que conectaran a un conector DB-25 macho que acoplara con el DB-25 hembra que hay en el puerto paralelo de la computadora personal. Los +5 Voltios necesarios para la alimentación de las compuertas en ambos lados serán tomados del pin 16 del conector de la DRE-5000. En la figura 9 se muestra el diagrama de circuito para una señal de entrada y una de salida de la DRE-5000; este circuito sera repetido 8 y 8 veces para entrada y salida respectivamente. El diagrama completo de los circuitos será presentado mas adelante.

Además de las compuertas TTL mencionadas anteriormente, se usaran 2 circuitos TTL 7474. Estos circuitos son de tipo Flip-Flop, y su función sera proveer un estado alto en la salida al detectar un pulso de subida en el reloj. Este circuito se utilizara para las señales EW, BOT y EOT. La finalidad es que los Flip-Flops

FIGURA 9



Circuito de interfase entre la DRE-5000 y puerto de la computadora personal

detecten y memoricen la activación de las señales. De tal manera que no sea necesario mantener a la computadora ocupada permanentemente con un programa en lenguaje ensamblador tratando de detectar estos pulsos (duración de 10 uS), sino que sea suficiente el correr una rutina de ensamblador que cheque el estado de los Flip-Flops cada unos cuantos milisegundos. El diagrama para los Flip-Flops se muestra en la figura 5.7. Estos Flip-Flops serán reseteados cuando la señal RUN se haga falsa. Cuando RUN se haga verdadera de nuevo, el CLEAR de los Flip-Flops se hará falso, permitiéndoles a estos funcionar y reconocer cuando EW, BOT o EOT segun el caso se hagan presentes. Esto se puede hacer ya que no es necesario que los Flip-Flops se mantengan activos mientras la cinta no se encuentre en movimiento. Al aprovechar la señal de RUN, se evita el tener que utilizar otro bit de interface. Como podrá notarse, el utilizar la computadora personal para el equipo de prueba simplifica mucho el diseño de los circuitos, ya que la mayoría de las funciones se implantarán por medio de programas que estarán corriendo en la computadora. En la tabla 9 se muestran los valores que sera necesario escribir a la dirección H0378 del puerto paralelo para manejar a las señales que controlan a la DRE-5000. Los valores de esta tabla se generan de los valores mostrados en la tabla 4 anteriormente mencionada, asi como de los valores requeridos por las señales de interfase de la DRE-5000 mostrados en la tabla 3. La relación entre una y otra tabla esta dada por la tabla 7, en donde se muestra la asignación de pins DRE-5000-Puerto Paralelo. Por ejemplo, para un movimiento rapido en reversa, se necesita que las señales SEARCH, RUN y REVERSE de entrada hacia la DRE-5000 sean validas. En la tabla 3 se observa que estas señales son activas cuando su estado lógico es bajo ("0" lógico). En la tabla 7 podemos ver que estas señales están relacionadas con los pins 2, 3 y 9 repectivamente. En la tabla 4 vemos que al

TABLA 9.

7	6	5	4	3	2	1	0	tipo de movimiento
0	1	1	1	0	0	0	0	reversa
1	1	1	1	0	0	0	0	adelanto
1	1	1	1	0	0	1	0	detener cinta
1	0	1	1	0	0	0	1	escritura track 1
1	0	0	1	0	0	0	1	escritura track 2
1	0	1	0	0	0	0	1	escritura track 3
1	0	0	0	0	0	0	1	escritura track 4
1	1	1	1	0	0	0	1	lectura track 1
1	1	0	1	0	0	1	0	lectura track 2
1	1	1	0	0	0	1	0	lectura track 3
1	1	0	0	0	0	1	0	lectura track 4

Valores en puerto H0378 para movimiento de la cinta en la DRE-5000.

escribir un "1" lógico a estos pins resulta en un nivel bajo a su salida, por lo tanto, será necesario escribir "0" lógico a estos bits para que la salida de estos que será conectada como entrada a las señales SEARCH RUN y REVERSE presente un estado bajo. En la tabla 4 podemos ver que los pins asignados a estas señales (2, 3 y 9) corresponden a los bits 0, 1 y 7 del puerto en la dirección hexadecimal 0378. Una vez obtenidos los valores que deben presentar los bits 0, 1 y 7, es necesario determinar que valores deben tener el resto de los bits. En la tabla 7 tenemos que los restantes bits están asignados a las señales WRITE DATA ONE, WRITE STROBE, SEL.TRACK B, SEL.TRACK A y WRITE ENABLE. En el caso de movimiento rápido en reversa, no es posible tener escritura, por lo que las señales relacionadas deben estar inactivas. En la tabla 3 vemos que WRITE STROBE es inactiva en estado bajo, mientras que WRITE ENABLE lo es en estado alto. El estado de WRITE DATA ONE es indiferente, ya que mientras las señales anteriores no esten activas, esta línea no será leída. Siendo asi, podemos elegir arbitrariamente que tenga un valor de "0". De la tabla 7 y 4 se obtiene que WRITE DATA ONE, WRITE STROBE y WRITE ENABLE están relacionadas respectivamente a los pins 4, 5 y 8 y estos a su vez corresponden a los bits 2, 3 y 6 del puerto H0378, los cuales deberán tener los valores 0,0,1 respectivamente. Para las señales TRACK SEL.B y TRACK SEL.A, supondremos que el movimiento se efectúa mientras estamos posicionados en el track 1 (el número de track es indiferente durante el movimiento rápido en reversa, ya que únicamente está regresandose la cinta), lo cual de acuerdo a la tabla 2 da valores de "1" para ambas señales. De nuevo, de la tabla 7 obtenemos que estas señales están conectadas a los pins 6 y 7. En la tabla 4 encontramos que estos corresponden a los bits 4 y 5 respectivamente, los cuales tendrán que tener un valor de "1". Una vez encontrados

los valores para cada bit, tenemos que el valor para movimiento en reversa está representado con "01110000", que es el valor mostrado en la tabla 9 para este tipo de movimiento. Los demás valores mostrados en la tabla fueron asignados de acuerdo al procedimiento anterior, utilizando los valores mostrados en las tablas 3, 4, 5, 6 , 7 y 8, según el caso. Estos valores son niveles lógicos que necesitarán ser escritos al puerto pararalelo de la computadora personal, para que este a su vez genere en su salida los niveles de voltaje necesarios para que a traves de los circuitos y de la interface maneje a la grabadora de datos DRE-5000. Para poder llevar esto a cabo, será necesario diseñar y desarrollar programas para la computadora, lo cual se verá posteriormente.

En las figuras 10 y 11 se muestran los diagramas para los circuitos del probador para la grabadora DRE-5000, los cuales como ya se mencionó anteriormente, estarán divididos en dos tarjetas de circuito impreso, las cuales estarán unidas por cable de tipo par trenzado, para cumplir con los requerimientos de línea de la interfase de la grabadora DRE-5000.

FIGURA 10

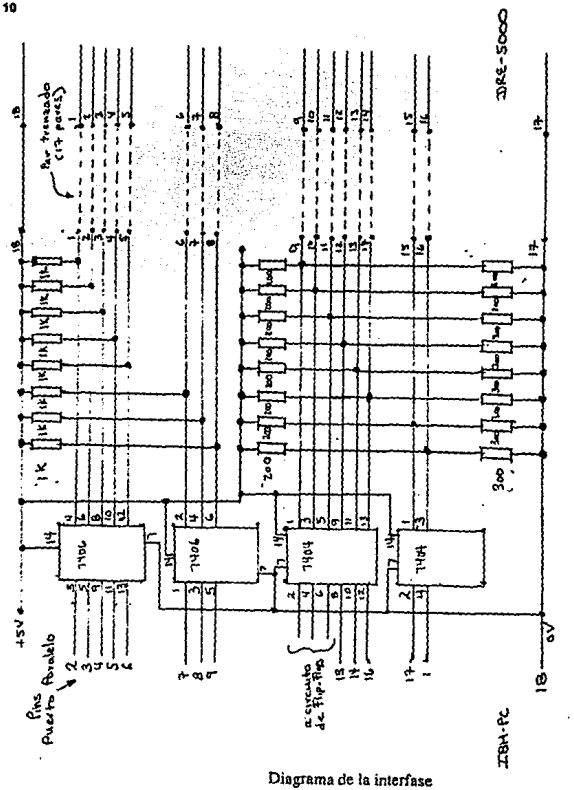


Diagrama de la interfase

FIGURA 11

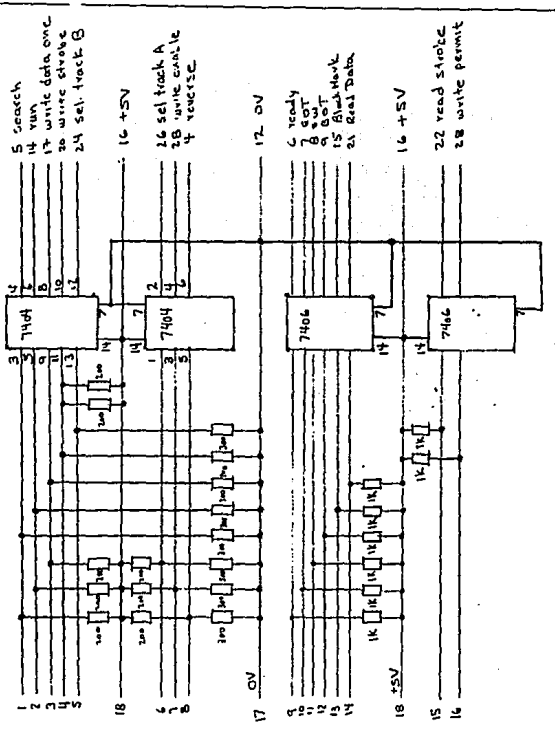


Diagrama de la interfase

3.2 Circuito de Protección en Fuentes de Alimentación.

Se requieren tres diferentes voltajes de alimentación para la grabadora DRE-5000. Estos son, +5V para la circuitería lógica y de control, +18V y -18V para el motor y los estabilizadores. Los requerimientos para dichos voltajes son los siguientes:

+5 V.c.d. +20%	1.5 Amp.Maximo
+18 V.c.d. + 5%	2 Amp. promedio
	5 Amp. picos con duración de 20mS
-18 V.c.d. + 5%	2 Amp. promedio
	5 Amp. picos con duración de 20mS

Las siguientes notas deben tomarse en cuenta:

a) Solo una de las líneas de 18 V. estará significativamente cargada en cualquier tiempo dado, mientras que la otra tendrá una carga de solo algunos cientos de miliamperes.

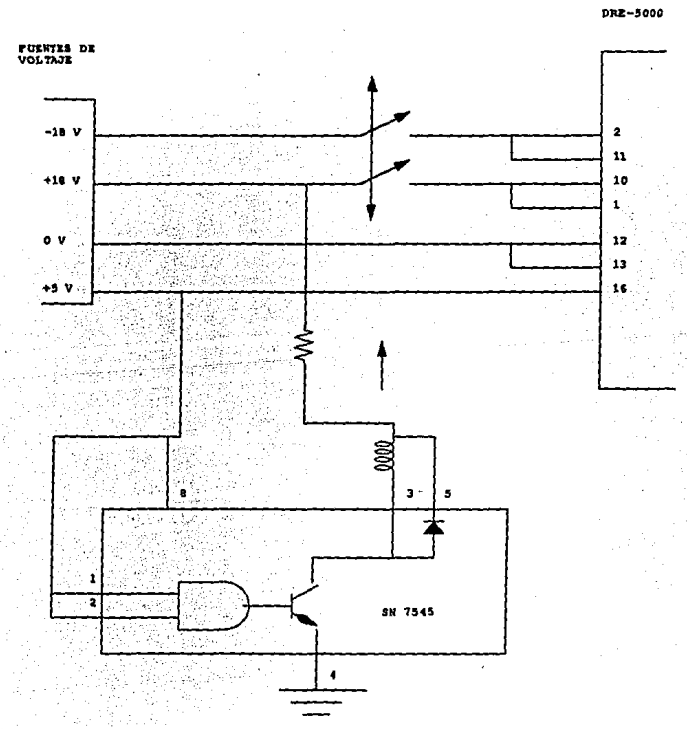
b) Los picos de corriente ocurrirán solamente durante el tiempo en que la cinta este siendo acelerada. Durante este tiempo es permisible que la línea de 18 V. que esta siendo cargada caiga a un 90% de su valor nominal.

c) Para evitar la tendencia de la cinta a moverse cuando se aplican los voltajes de alimentación a la DRE-5000, el voltaje de +5V. deberá ser aplicado el primero y desconectado el ultimo. De lo contrario se corre el riesgo de que el motor, sin control, tienda a exigir un suministro ilimitado de corriente, dañando algún componente de la circuitería.

Los tres voltajes de alimentación mencionados anteriormente se encuentran disponibles en las centrales AXE en las cuales operan las grabadoras DRE-5000. Por lo que no existe necesidad de que el equipo de prueba cuente con fuentes de alimentación propias. Lo cual resulta benéfico, ya que buscándose que dicho equipo sea portátil, se evita un peso excesivo en este, por lo cual podrá ser mas fácilmente transportado. Sin embargo, como se vio anteriormente en el inciso d de las notas, es muy importante contar con un circuito que asegure que los voltajes de +18V. no estarán presentes en la DRE-5000, si no se encuentra presente el voltaje de +5V. Un circuito de protección tal no existe actualmente en las centrales ni en las grabadoras DRE-5000, por lo que se considera necesario incluirlo en el diseño del equipo de prueba. El circuito de protección consistirá básicamente en un par de relevadores, los cuales cerraran el circuito para las alimentaciones de +18V., solo cuando +5V. se encuentre presente Se seleccionaron los relevadores tipo R10-E6-X2-V185, ya que puede manejar una corriente de hasta 7.5 amp.en los contactos. Como se mostró anteriormente en los requisitos para las fuentes de voltaje, la corriente máxima suministrada por las fuentes de +18V. es un pico de 5 amp. con duración máxima de 20 mS., por lo que este relevador puede manejar con facilidad dicha corriente. Este tipo de relevador requiere de 12V., y una corriente de bobina de 60 mA. para ser activado. Para manejar a este relevador se usara un circuito

integrado SN75475, como se muestra en la figura 12. Las entradas de la compuerta and estarán conectadas a +5 Voltios, de modo que mientras este voltaje no este presente (la compuerta ni siquiera tendrá alimentación) el relevador no actuara y la DRE-5000 no recibirá voltajes de + 18 Voltios. Este circuito es completamente independiente de los circuitos de interface entre la grabadora de datos y la computadora; es decir, no estará conectado directamente a la computadora, ni sera controlado por esta.

FIGURA 12


DRE-5000
FUENTES DE VOLTAJE
-18 V
+18 V
0 V
+5 V
2
11
10
1
12
13
16
8
3
5
1
2
SN 7545
4

CAPITULO 4 Diseño de Los Programas.

4.1 Los Lenguajes de Programación.

Después de haber definido las conexiones en la interface, así como que bits del puerto paralelo corresponden a cada señal, puede comenzarse el diseño de los programas que se encargaran de manejar dichas señales. Para estos programas se ha decidido utilizar dos lenguajes. Pascal y Ensamblador 8088/8086. La razón para utilizar dos lenguajes, es que uno es de alto nivel (Pascal) y el otro es de bajo nivel (ensamblador).

4.1.1 Pascal

Un lenguaje de alto nivel como es Pascal permite elaborar con mayor facilidad programas de interface hacia el usuario (menús, instrucciones, etc.), sin embargo, para aplicaciones que requieren un tiempo exacto, es mejor utilizar un lenguaje de bajo nivel como es el ensamblador 8088/8086. Para compilar los programas hechos en Pascal, se utilizara Turbo Pascal. Este compilador permite integrar procedimientos o subprogramas en lenguaje ensamblador mediante dos métodos. Uno llamado "in-line" que permite que el código ensamblador forme parte directamente de un programa. El otro método consiste en ensamblar separadamente la rutina y enlazarla con el resto del programa utilizando la orden external. Este ultimo método ha sido escogido, ya que permite una organización modular, y se evita tener que modificar el programa principal cuando sean necesarios cambios a una rutina de ensamblador como veremos posteriormente.

4.1.2 Ensamblador 8088/8086.

Como se menciono anteriormente se decidió utilizar el lenguaje ensamblador para rutinas que requieren velocidad y exactitud de tiempo, ya que en este lenguaje se conoce el número exacto de ciclos de máquina que toma a una instrucción ejecutarse, y por lo tanto puede calcularse el tiempo con exactitud. Para la grabadora de datos DRE-5000, las señales write strobe, read strobe, write data one, read data, tienen requerimientos de tiempo del orden de microsegundos, que

pueden proveerse con exactitud con instrucciones de lenguaje ensamblador. por ejemplo, una instrucción "out dx,al" toma 8 ciclos de máquina en ejecutar e una computadora personal IBM o compatible con un procesador 8088, cada ciclo de máquina toma 1/4.77MHz por lo tanto la instrucción anterior se ejecuta en 1.67 microsegundos. Mas adelante veremos los cálculos para cada una de las señales. Este diseño estará basado en el procesador 8088, ya que todas las computadoras personales con que se cuenta son IBM PC compatible que tienen este micropocesador y trabajan a 4.77 MHz. Si llegara a requerirse utilizar los programas en una computadora con otro microprocesador (8086 por ejemplo) o con otra velocidad de reloj, solamente seria necesario recalcular el número de instrucciones requeridas de acuerdo a la duración en tiempo de cada una de la misma forma que se vio arriba. Esta ultima consideración hace aun mas atractivo el utilizar una combinación de lenguaje de alto nivel y ensamblador, ya que si se requieren cambios para ajuste de tiempos, estos se harán en pequeña subrutinas en ensamblador, mientras que el programa principal no seria afectado. Por esta razón también, el diseño se hará en forma modular, para hacer posible que las modificaciones se hagan solo a el programa o subrutina que sea necesario.

4.2 El Programa Principal (Pascal).

El diseño del programa principal se hará en forma de menús, con el fin que sea amigable hacia el usuario. Este programa también se encargara de manejar las rutinas de ensamblador y de interfasar los requerimientos o mensajes de estas hacia

el operador. El menú principal constara de las diferentes opciones de prueba que serán ofrecidas por el equipo de prueba. Estas son:

a) Regreso de cinta

b) Adelanto de cinta

c) Pruebas de lectura y escritura

d) Lectura continua de cinta

e) Opción para salir del programa en forma adecuada

Cada opción del menú será manejada por un subprograma (tambien en lenguaje Pascal), el cual se encargará de las tareas necesarias, incluyendo el llamado a rutinas de ensamblador cuando así sea requerido. El uso de subprogramas permite modularidad y simplica posibles posteriores modificaciones si estas son necesarias. Tambien simplifica el diseño ya que permite concentrarse en una tarea a la vez. En esta sección se explicará cada una de las partes de el programa en Pascal, el cual queda subdividido en una sección de declaraciones, un procedimiento principal y subprocedimientos para cada una de las opciones del menú del procedimiento principal.

4.2.1 La sección de declaraciones.

```
program dri5000 (input, output);

label 1;

  var
```

```
    dummy, choice : char;
    dummy2 : integer;
  procedure clr; external 'clr.com';
  procedure rever; external 'rever.com';
  procedure forwar; external 'forwar.com';
  procedure leec; external 'leec.com';
  procedure deten; external 'deten.com';
  procedure ini37a; external 'ini37a.com';
  procedure wrtrk1; external 'wrtrk1.com';
  procedure wrtrk2; external 'wrtrk2.com';
  procedure wrtrk3; external 'wrtrk3.com';
  procedure wrtrk4; external 'wrtrk4.com';
  function cready(x:integer):integer; external 'cready.com';
  function cew(x:integer):integer; external 'cew.com';
  function cbot(x:integer):integer; external 'cbot.com';
  function ceot(x:integer):integer; external 'ceot.com';
  function cwperm(x:integer):integer;external 'cwperm.com';
```

```
function wrdri(x:integer):integer; external 'wrdri.com';

function lee(x:integer):integer; external 'lee.com';

function leetrk1(x:integer):integer;external 'leetrk1.com';

function leetrk2(x:integer):integer;external 'leetrk2.com';

function leetrk3(x:integer):integer;external 'leetrk3.com';

function leetrk4(x:integer):integer;external 'leetrk4.com';
```

La porción del programa mostrada comienza por las declaraciones necesarias. El nombre del programa (dre5000) que tiene entrada y salida. Despues declara las variables que serán utilizadas. A continuación declara las rutinas de lenguaje ensamblador como funciones y procedimientos externos, de manera que puedan ser llamadas por el programa posteriormente. En el lenguaje Pascal todas las declaraciones se hacen al principio del programa. Despues se incluyen los procedimientos y al final el programa principal. La parte mostrada arriba seria la inicial con las declaraciones. Para mayor claridad se alterara este orden durante la explicación, y al final se mostrará el programa completo con el orden debido.

4.2.2 El procedimiento principal.

```
begin

gotoxy(15,9);writeln('Probador para DRE-5000'); writeln(' ');

gotoxy(15,11);writeln('Diseñado por Pedro Octavio Diez de Sollano');

gotoxy(15,13);writeln('Director de Tesis: Ing. Antonio Herrera Mejia');
```

```
gotoxy(15,15);writeln('Facultad de Estudios Superiores Cuautitlan');

gotoxy(15,17);writeln('Universidad Naciónal Autonoma de Mexico');

gotoxy(20,24);writeln('Presionar ENTER para continuar.');

readln;

clr;

gotoxy(8,4);

writeln('Los siguientes puntos deberan checarse antes de comenzarlas pruebas:');

gotoxy(8,8);

writeln('a) La DRI-5000 debera estar conectada a la interfase de prueba');

gotoxy(8,10);

writeln('b) La fuente de voltaje debe estar encendida');

gotoxy(8,12);

writeln('c) Un cartucho de prueba y uno de referencia deben estar disponibles');

gotoxy(5,24);

writeln('Presionar ENTER para continuar, C para terminar el programa.');

readln;

clr;
```

```
choice:='a';
while  choice <> '9' do
        begin
        deten;  { poner cinta en reposo siempre que se llegue a este menu }
        gotoxy(30, 10);writeln('MENU');
        gotoxy(25, 12);writeln('1) Regresar cinta');
        gotoxy(25, 13);writeln('2) Adelantar cinta');
        gotoxy(25, 14);writeln('3) Pruebas de lectura y escritura');
        gotoxy(25, 15);writeln('4) Lectura continua');
        gotoxy(25, 16);writeln('9) Terminar pruebas');
        gotoxy(12, 21);writeln('Introduzca número de opción deseada:');
        read(choice);
        case choice of
                        '1' : rewinds;
                        '2' : fforwards;
                        '3' : tests;
                        '4' ; lecont;
```

```
        end;  { case }

        clr;

end;  { while }

1 : gotoxy(23,12);writeln('fin de pruebas');

deten;

end.
```

El procedimiento principal comienza por escribir en la pantalla los datos de esta tésis como son el nombre del autor, nombre del director de tésis, nombre de la escuela y universidad. Para que estos datos aparezcan ordenados y centrados en la pantalla, se utiliza el comando `gotoxy' para comenzar a escribir en la posición deseada. Por ejemplo, el comando `gotoxy(15,9)' nos posiciona en la columna 15, renglón número nueve de la pantalla. De esta manera se puede controlar la manera en que será desplegada la información. Así se puede dar tambien la impresión de los menús con opciones para el usuario. A una instrucción `gotoxy' le sigue un comando `writeln()' que es el que escribe la información en la pantalla. Al terminar de escribir los datos en la pantalla se muestra tambien el mensaje: `Presionar ENTER para continuar' para indicar que debe presionarse la tecla de enter cuando se este listo para seguir con el programa. De esta manera se permite que el usuario pueda decidir que tanto tiempo permanece observando o leyendo la información de la pantalla. Para lograr que el programa espere se utiliza la instrucción `readln' cuya función consiste en recibir la información de las teclas que presione el usuario y terminar cuando la tecla ENTER es presionada. En este caso cualquier otra tecla

que fuera presionada no será utilizada y simplemente se continuará ejecutando el programa cuando 'readln' reconozca que la tecla ENTER fue presionada. Continuando con la secuencia del programa, despues de presionar ENTER la pantalla será limpiada con 'clr', que es en realidad el llamado a una subrutina en lenguaje ensamblador. En la parte correspondiente a los programas en ensamblador se verá el contenido de esta subrutina y como esta limpia la pantalla. Utilizando nuevamente comandos 'gotoxy' combinados con 'writeln()' se imprimen instrucciones en la pantalla, en las que se recuerda al usuario los puntos que debe checar antes de continuar con el programa. Nuevamente se utiliza 'readln' para esperar a que el usuario conteste presionando ENTER. Una vez que esto ocurre el programa muestra el menu con las opciones para prueba de la grabadora DRE-5000, el cual se verá en la pantalla de la siguiente forma:

MENU:

1) Regresar cinta

2) Adelantar cinta

3) Pruebas de lectura y escritura

4) Lectura continua

9) Terminar pruebas

Para que el programa obtenga la respuesta del usuario con la opción elegida se utilizará ahora la instrucción 'read', la cual toma la información de la primera tecla

presionada por el usuario y la guarda en una variable (choice en este caso). Ya con el valor en la variable se utiliza `case' para que dependiendo de la tecla presionada por el usuario se proceda con una de 4 opciones. Si la tecla no corresponde con ninguna opción, se regresará a utilizar el comando `read' hasta que se presione una tecla válida. Se ha utilizado la etiqueta `1:' en la instrucción para imprimir "fin de pruebas" simplemente para contar con una salida del programa principal. Antes de realmente finalizar el programa se llama a la subrutina de ensamblador `deten' para que se encargue de poner los valores de voltaje necesarios en la interfase para que la cinta en la grabadora DRE-5000 quede en estado de reposo. En el listado mostrado anteriormente, correspondiente al procedimiento principal puede notarse que al seleccionar cada opción en el menú se hará un llamado a un subprocedimiento. Los subprocedimientos son: rewinds, fforwards, tests y lecont, y estos serán explicados a continuación.

4.2.3 El procedimiento rewinds.

```
procedure rewinds;

  var

   a,estat,esta2,term : integer;

  begin

  clr;

  gotoxy(6,2);writeln('Antes de comenzar, favor de verificar:');

  gotoxy(10,6);writeln('- Fuentes de voltaje conectadas y encendidas');
```

```
gotoxy(10,8);writeln('- Bulbo de fotosensor encendido');
gotoxy(10,10);writeln('- Cartucho de prueba correctamente cargado');
gotoxy(10,12);writeln('- Probador conectado correctamente');
gotoxy(10,20);writeln('Presionar enter para continuar');
readln;
clr;
{ aqui se llama a cready.com para checar si ready = lo (activo) }
estat:=cready(a);
if estat > 0
  then
    begin
      gotoxy(10,4);writeln('ready no esta activa. ');
      gotoxy(10,6);writeln('checar circuito servo y corregir falla');
      gotoxy(10,8);writeln('presionar ENTER para continuar');
      readln;
    {   choice:='9';  para abortar al retornar }
    end
```

```
  else
  begin
  gotoxy(10,2);writeln('verificar que la cinta este en movimiento mientras');
  gotoxy(10,3);writeln('este mensaje aparezca en pantalla');
  gotoxy(10,8);writeln('si esto no sucede, interrumpir el programa, y checar:');
  gotoxy(13,9);writeln('- circuito "servo" si la cinta no se movio');
  gotoxy(13,10);writeln('- circuito de "EW" en tarjeta de fotosensor si el');
  gotoxy(13,11);writeln(' mensaje "EW detectado" no aparece abajo y el');
  gotoxy(13,12);writeln(' programa sigue corriendo despues que la cinta pare');
    gotoxy(13,13);writeln('- circuito de BOT si el mensaje "EW detectado"
aparece');
   gotoxy(13,14);writeln(' y el programa sigue corriendo despues que la cinta
pare');
  gotoxy(30,16);writeln('regresando cinta');
  rever;          { aqui se llama a rever.com }
  estat:=0;
  while estat = 0 do  { esperar por EW }
```

```
 begin
   estat:=cew(a); { aqui se llama a cew.com };
   gotoxy(30,16);writeln('regresando cinta');
end;  { while }
gotoxy(10,18);writeln('EW detectado');
esta2:=0;
while esta2 = 0 do  { esperar por BOT }
 begin
   esta2:=cbot(a);  { aqui se llama a cbot.com };
   gotoxy(30,16);writeln('regresando cinta');
 end; { while }
clr;
gotoxy(10,8);writeln('BOT (principio de cinta) detectado');
gotoxy(10,10);writeln('Cinta detenida? (S/N)');
read(choice);
case choice of
   'S', 's' : begin
```

```
            gotoxy(10,12);
            writeln('correcto, presionar ENTER para continuar');
            readln(dummy2);
           end;
      'N', 'n' : begin
                 gotoxy(10,12);
                 writeln('Falla en circuito SERVO y de INTERLOCK');
                 gotoxy(10,14);
                 writeln('Checar y corregir falla antes de correr pruebas de nuevo');
                 gotoxy(10,16);
                 writeln('presionar ENTER para continuar');
                 readln;
                 choice := '9';  { para abortar programa al retornar }
                end;
    end; { case }
  end  { else }
end;  { rewinds }
```

Entre las declaraciones mostradas en la parte del programa que fue presentada anteriormente se encuentra la declaración de el procedimiento 'rewind', el cual es un subprograma en Pascal que es llamado si el usuario selecciona la opción número uno en el menú mencionado anteriormente. La función de este procedimiento consiste en regresar la cinta que se encuentre en la grabadora de datos DRE-5000. Para ello se comienza por dar instrucciones al usuario sobre las condiciones requeridas para poder hacer el regreso de cinta, de tal forma que este pueda checar y corregir si alguna de estas condiciones falta. Una vez que el usuario presiona ENTER para ordenar al programa continuar, este comienza por llamar a la subrutina de ensamblador 'cready', la cual verifica que la señal 'ready' proveniente de la interfase de la grabadora de datos. Si esto no se cumple, entonces el programa mandará un mensajes al usuario mencionando que 'ready' no se encontraba activa y que el circuito servo deberá ser checado. Nuevamente, para continuar, el usuario deberá presionar ENTER. A la variable 'choice' se le asigna el valor 9 de tal manera que al regresar del subprograma al programa principal este reconozca a 9 como indicación para detener el programa para que el usuario pueda corregir la falla. Si la subrutina 'cready' encuentra que la señal ready esta activa, entonces lo anterior no será necesario, y el subprograma 'rewind' continua. La siguiente tarea de este subprograma será poner las señales necesarias en la interfase de la DRE-5000 para que la cinta sea puesta en movimiento de reversa. Además monitoreará las señales EW y BOT (early warning y beginning of tape respectivamente) para verificar que estas se presenten y detener la cinta cuando el principio de cinta sea detectado. Como el programa no tendrá manera de saber que tiempo exactamente

tardará la cinta en regresar y existe la posibilidad de que haya un daño en los circuitos y por lo tanto las señales EW y BOT no sean detectadas por el programa, antes de comenzar el movimiento de la cinta se imprimen mensajes en la pantalla indicando al operador que hacer en caso que esto último ocurriera. Las posibilidades son que la cinta no se ponga en movimiento en cuyo caso el programa indica checar el circuito 'servo'. Que la cinta sea regresada y se detenga despues, indicando que todo funcionó bien, pero las señales EW y/o BOT no fueron detectadas, en cuyo caso el programa indica checar el circuito de la tarjeta del fotosensor. Si EW y BOT son detectadas, entonces se imprimen mensajes en la pantalla indicandolo asi y se pregunta al usuario si la cinta esta ya detenida. Si la cinta no se detuvo, se manda un mensaje al usuario indicandole checar el circuito servo y de interlock para corregir las fallas antes de intentar de nuevo. En caso de que haya habido alguna falla el valor de 'choice' se pone en 9 para que al regresar al programa principal este pare. Si el usuario llegara a preferir que el programa no termine al encontrar errores, sino que permanezca dentro del menú principal, bastará con cambiar el valor 9 por algun otro (8 por ejemplo) en la instrucción 'while' dentro del programa principal, para que de esta manera ignore el 9 regresado por los subprogramas en caso de errores, conservando la opción para salir del programa ahora al presionar la tecla 8. La segunda opción en el menú principal es el adelanto de cinta. A continuación se muestra el subprograma 'fforwards' que realiza esta tarea:

4.2.4 El procedimiento fforwards.

```
procedure fforwards;

  var
```

```
a,estat,term : integer;
begin
clr;
gotoxy(6,2);writeln('Antes de comenzar, favor de verificar:');
gotoxy(10,6);writeln('- Fuentes de voltaje conectadas y encendidas');
gotoxy(10,8);writeln('- Bulbo de fotosensor encendido');
gotoxy(10,10);writeln('- Cartucho de prueba correctamente cargado');
gotoxy(10,12);writeln('- Probador conectado correctamente');
gotoxy(10,20);writeln('Presionar enter para continuar');
readln;
clr;
  { aqui se llama a cready.com para checar si ready = lo (activo) }
estat:=cready(a);
if estat > 0
  then
    begin
      gotoxy(10,4);writeln('ready no esta activa. ');
```

```
        gotoxy(10,6);writeln('checar circuito servo y corregir falla');
        gotoxy(10,8);writeln('presionar ENTER para continuar');
        readln;
      { choice:='9'; para abortar al retornar }
      end
    else
  begin
  gotoxy(10,2);writeln('verificar que la cinta este en movimiento mientras');
  gotoxy(10,3);writeln('este mensaje aparezca en pantalla');
  gotoxy(10,8);writeln('si esto no sucede, interrumpir el programa, y checar:');
  gotoxy(13,9);writeln('- circuito "servo" si la cinta no se movio');
  gotoxy(13,10);writeln('- circuito de "EW" en tarjeta de fotosensor si el');
  gotoxy(13,11);writeln(' mensaje "EW detectado" no aparece abajo y el');
  gotoxy(13,12);writeln(' programa sigue corriendo despues que la cinta pare');
    gotoxy(13,13);writeln('- circuito de EOT si el mensaje "EW detectado"
aparece');
```

```
    gotoxy(13,14);writeln('  y el programa sigue corriendo despues que la cinta
pare');
   gotoxy(26,16);writeln('adelantando cinta');
   forwar;{ aqui se llama a forwar.com }
   estat:=0;
   while estat = 0 do  { esperar por EW }
    begin
     estat:=cew(a);{ aqui se llama a cew.com }
     gotoxy(26,16);writeln('adelantando cinta');
    end;
   gotoxy(10,18);writeln('EW detectado');
   while estat > 0 do  { esperar por EOT }
    begin
     estat:=ceot(a);{ aqui se llama a ceot.com }
     gotoxy(26,16);writeln('adelantando cinta');
    end;
   clr;
```

```
gotoxy(10,8);writeln('Fin de cinta detectado');
gotoxy(10,10);writeln('Cinta detenida? (S/N)');
read(choice);
case choice of
  'S', 's' : begin
       gotoxy(10,12);
       writeln('correcto, presionar ENTER para continuar');
       readln(dummy2);
      end;
  'N', 'n' : begin
          gotoxy(10,12);
          writeln('Falla en circuito SERVO y de INTERLOCK');
          gotoxy(10,14);
          writeln('Checar y corregir falla antes de correr pruebas de nuevo');
          gotoxy(10,16);
          writeln('presionar ENTER para continuar');
          readln;
```

```
            choice := '9';  { para abortar programa al retornar }
          end;
      end; { case }
    end { else }
  end;  { fforwards }
```

Este subprograma efectua una función muy parecida a la de el subprograma anterior, sólo que en este caso el movimiento de la cinta es para adelantarla, y en vez de checar la señal BOT (beginning of tape) se checa la señal EOT (end of tape). Al igual que 'rewinds', el subprograma comienza poniendo recordatorios al usuario sobre los puntos que deben estar listos antes de poder regresar la cinta. Una vez que el usuario contesta el programa checa que la señal 'ready' este activa haciendo un llamado a la subrutina de lenguaje ensamblador `cready'. Si esta señal no se encuentra activa se envia el mensaje al usuario para que revise el circuito servo y sea corregida la falla. Si 'ready' se encuentra activa, entonces se muestran en la pantalla los posibles casos que no podrían ser detectados por el programa (como se mencionó anteriormente, el programa no tiene forma de saber que tanto tiempo tomará adelantar la cinta, ya que esto depende de que tan lejos esté el fin de cinta de la posición en que se inicia el movimiento). A continuación se comienza el movimiento de la cinta en velocidad de búsqueda hacia adelante por medio de un llamado a la subrutina de lenguaje ensamblador 'forwar' y despues se llama a las subrutinas 'cew' y `ceot' para verificar que las señales 'EW' y 'EOT' se presenten.

Si ocurriera algún error, el programa imprime mensajes en la pantalla sobre los circuitos a revisar según el caso, los cuales pueden ser el circuito servo o el circuito de EOT en el fotosensor. Se asume en este caso que la prueba de regreso de cinta fue corrida anteriormente y funcionó bien, por lo que no se menciona revisar el circuito de EW. La tercera opción del menú es la de pruebas de lectura y escritura, las cuales son efectuadas por el subprograma `tests', el cual se lista a continuación:

4.2.5 El procedimiento tests.

```
procedure tests;

 var

  a,este,prl1,prl2,prl3,prl4,term : integer;

 begin

 clr;

 gotoxy(10,4);writeln('insertar cartucho de cinta de prueba');

 gotoxy(10,6);writeln('presionar ENTER para continuar');

 readln;

 ini37a; {inicializa puerto 037AH para poder leer correctamente}

 clr;

 este:=cwperm(a);

 if este = 0
```

```
 then
  begin
   gotoxy(10,2);writeln('falla al leer WRITE PERMIT');
  end
 else
  begin
   gotoxy(10,2);writeln('WRITE PERMIT leido correctamente');
 end;
gotoxy(10,4);writeln('regresando cinta');
rever;              { aqui se llama a rever.com }
este:=0;
while este = 0 do
 begin
   este:=cbot(a);          { aqui se llama a cbot.com }
   gotoxy(10,4);writeln('regresando cinta');
 end; {while}
deten;   { aun cuando la cinta esta detenida,necesita este para }
```

```
        { poder iniciar movimiento otra vez }
gotoxy(10,6);writeln('escribiendo a cinta para checar BLOCK MARK');
este:=wrdri(a);  { aqui se llama a wrdri.com para escribir a cinta }
if este = 0      { pin 14 = 0 Si block mark no fue detectado }
  then
    begin
      gotoxy(10,8);writeln('BLOCK MARK no fue detectado');
      gotoxy(10,10);writeln('detener el programa');
      gotoxy(10,12);writeln('y checar circuitos BLOCK MARK Generator y');
      gotoxy(10,14);writeln('READ Amplifier');
      gotoxy(10,20);writeln('presionar ENTER para continuar');
    { choice:='9'; }
      deten;  { poner cinta en estado de reposo }
      readln;
    end
  else
    begin
```

```
    gotoxy(10,14);writeln('BLOCK MARK detectado');
  end;
deten;
gotoxy(10,24);writeln('regresando cinta');
rever;              { aqui se llama a rever.com }
este:=0;
while este = 0 do
 begin
   este:=cbot(a);          { aqui se llama a cbot.com }
   gotoxy(10,24);writeln('regresando cinta');
 end; {while}
clr;
deten;
gotoxy(10,2);writeln('escribiendo a track 1');
wrtrk1;
gotoxy(10,3);writeln('escribiendo a track 2');
wrtrk2;
```

```
gotoxy(10,4);writeln('escribiendo a track 3');

wrtrk3;

gotoxy(10,5); writeln('escribiendo a track 4');

wrtrk4;

gotoxy(26,7);writeln('regresando cinta');

rever;

este:=0;

while este=0 do

 begin

  este:=cbot(a);

  gotoxy(26,7);writeln('regresando cinta');

 end; { while }

deten;

ini37a;

gotoxy(10,9);writeln('si la cinta y el programa no paran significa que');

gotoxy(10,10);writeln('read strobe no esta presente.');

gotoxy(10,11);writeln('en ese caso, checar circuito READ DECODER, ');
```

```
gotoxy(10,12);writeln('READ AMPLIFIER y cabeza de lectura');
gotoxy(23,14);writeln('leyendo de cinta en track 1');
prl1:=leetrk1(a);
gotoxy(23,15);writeln('leyendo de cinta en track 2');
prl2:=leetrk2(a);
gotoxy(23,16);writeln('leyendo de cinta en track 3');
prl3:=leetrk3(a);
gotoxy(23,17);writeln('leyendo de cinta en track 4');
prl4:=leetrk4(a);
{ este:=lee(a);          aqui se llama a lee.com }
case prl1 of
 0 : begin
      gotoxy(10,19);writeln('error de lectura en track 1');
     { choice:='9'; }
     end;
 1 : begin
      gotoxy(10,19);writeln('lectura correcta en track 1');
```

```
    end;
end;  { case }
case prl2 of
 0 : begin
      gotoxy(10,20);writeln('error de lectura en track 2');
     end;
 1 : begin
      gotoxy(10,20);writeln('lectura correcta en track 2');
     end;
end; { case }
case prl3 of
 0 : begin
      gotoxy(10,21);writeln('error de lectura en track 3');
     end;
 1 : begin
      gotoxy(10,21);writeln('lectura correcta en track 3');
     end;
```

```
    end; {case}
    case pri4 of
     0 : begin
          gotoxy(10,22);writeln('error de lectura en track 4');
         end;
     1 : begin
          gotoxy(10,22);writeln('lectura correcta en track 4');
         end;
    end; { case }
    deten;
    gotoxy(10,24);writeln('fin de pruebas de lectura y escritura');
    gotoxy(10,25);writeln('presionar ENTER para continuar');
    readln;
  end; { tests }
```

Este comienza comienza por poner un mensaje en la pantalla indicando al usuario que debe poner un cartucho de cinta de prueba en la grabadora de datos DRE-5000.

A continuación hace un llamado a la subrutina de lenguaje ensamblador 'ini37a', la cual se encarga de poner al puerto 037AH de el puerto paralelo de la computadora personal IBM o compatible en el estado adecuado para poder leer correctamente. Esto es necesario ya que este puerto utiliza circuitos de colector abierto, por lo que su estado solo podra ser cambiado si originalmente es alto. Más adelante se explicará esta subrutina 'ini37A' con detalle. Una vez que la subrutina termina, el programa limpia la pantalla y llama a la subrutina 'cwperm' para checar que la señal 'WRITE PERMIT' este activa. Si no lo esta, lo indica con un mensaje en la pantalla. Si lo esta, entonces lo indica con otro mensaje y llama a la subrutina de lenguaje ensamblador 'rever' para poner la cinta en movimiento de reversa rápida. LLama entonces a la subrutina 'cbot' para ordenar detener la cinta cuando el principio de cinta es detectado (cew no es llamada, ya que se asume que las pruebas 1 y 2 fueron corridas anteriormente y funcionaron, por lo tanto el circuito que proporciona la señal 'early warning' funciona correctamente). La subrutina de ensamblador 'deten' es llamada para poner a la grabadora de datos en estado de reposo. A continuación se llama a la subrutina 'wrdri' para escribir a cinta con el propósito de detectar si se produce la señal 'BLOCK MARK'. Esta subrutina escribe una serie de 15 ceros y 1 uno (preamble) a cinta, y despues verifica que la señal 'BLOCK MARK' este activa (debe hacerse verdadera despues que se han detectado los 4 primeros bits del preamble, indicando asi que las transiciones de flujo fueron detectadas). Si 'BLOCK MARK' no estuviera activa, entonces la subrutina regresa un valor de 0 en la variable 'este' y el programa pone mensajes en la pantalla indicando la falla y que los circuitos 'Block Mark Generator' y 'Read Amplifier' necesitan ser revisados para corregir la falla y hace un llamado a la subrutina 'deten' para poner la cinta en reposo. Si 'BLOCK MARK' se encuentra

activa, entonces se indica con un mensaje en pantalla y se hacen llamados a las subrutinas `deten', `rever' y `cbot' para detener y regresar la cinta hasta el principio de esta. Despues de esto se llama a las subrutinas de lenguaje ensamblador `wrtrk1', `wrtrk2', `wrtrk3' y `wrtrk4' para escribir en cada uno de los 4 tracks. Estas subrutinas escriben primero una serie de 15 ceros y 1 uno (preamble) y despues un texto que será diferente para cada track, lo cual permitirá que al hacer la lectura se este leyendo el texto correspondiente al track. Si se escribiera el mismo texto en todos los tracks, se correria el riesgo de no detectar si se hace cambio de track o no. Al terminar de escribir en los 4 tracks se detiene y regresa la cinta. Se hace despues un llamado a `ini37A' para asegurar que el puerto 037AH se encuentre en el estado correcto y se imprimen mensajes en la pantalla para indicar al usuario que hacer en caso de que ocurriera alguna situación que el programa no pudiera detectar, como seria el caso de que `READ STROBE' no sea generado. En este caso el programa indica verificar el circuito `Read Amplifier' y la cabeza de lectura. Se hace un llamado a la subrutinas de lenguaje ensamblador `leetrk1', `leetrk2', `leetrk3'y `leetrk4' para leer cada uno de los tracks. Se imprimirán mensajes de acuerdo al resultado de lectura en cada uno de los tracks. La opción número 4 del programa provee lectura continua de la cinta en la grabadora de datos DRE-5000. Esto es con el objeto de que el usuario pueda hacer ajustes con el osciloscopio. A continuación se lista el subprograma `lecont' que corresponde a esta opción:

4.2.6 El procedimiento lecont.

```
procedure lecont;

  var
```

```
  term, tempor : integer;
begin
clr;
gotoxy(10,2);writeln('lectura continua de datos');
gotoxy(10,6);writeln('presionar ENTER para continuar');
readln;
clr;
gotoxy(10,6);writeln('leyendo de cinta');
leec;  { aqui se llama a leec.com }
end;  { lecont }
```

Este subprograma pone mensajes en la pantalla para el usuario y una vez que este presion ENTER hace un llamado a la subrutina de lenguaje ensamblador 'leec', la cual se encarga de poner a la grabadora de datos DRE-5000 en estado de lectura durante un tiempo determinado, el cual originalmente ha sido fijado en 2 minutos, pero puede ser cambiado si el usuario requiriera un tiempo mayor o menor para efectuar los ajustes. Este cambio se efectuaría únicamente en la subrutina de lenguaje ensamblador 'leec', la cual se verá con mayor detalle posteriormente.

A continuación se muestra el listado completo del programa en Pascal:

4.2.7 Listado completo del programa.

```
program dri5000 (input, output);

label 1;

 var

  dummy, choice : char;

  dummy2 : integer;

procedure clr; external 'clr.com';

procedure rever; external 'rever.com';

procedure forwar; external 'forwar.com';

procedure leec; external 'leec.com';

procedure deten; external 'deten.com';

procedure ini37a; external 'ini37a.com';

procedure wrtrk1; external 'wrtrk1.com';

procedure wrtrk2; external 'wrtrk2.com';

procedure wrtrk3; external 'wrtrk3.com';

procedure wrtrk4; external 'wrtrk4.com';

function cready(x:integer):integer; external 'cready.com';
```

```
function cew(x:integer):integer; external 'cew.com';
function cbot(x:integer):integer; external 'cbot.com';
function ceot(x:integer):integer; external 'ceot.com';
function cwperm(x:integer):integer;external 'cwperm.com';
function wrdri(x:integer):integer; external 'wrdri.com';
function lee(x:integer):integer; external 'lee.com';
function leetrk1(x:integer):integer;external 'leetrk1.com';
function leetrk2(x:integer):integer;external 'leetrk2.com';
function leetrk3(x:integer):integer;external 'leetrk3.com';
function leetrk4(x:integer):integer;external 'leetrk4.com';
procedure rewinds;
  var
    a,estat,esta2,term : integer;
  begin
  clr;
  gotoxy(6,2);writeln('Antes de comenzar, favor de verificar:');
  gotoxy(10,6);writeln('- Fuentes de voltaje conectadas y encendidas');
```

```
gotoxy(10,8);writeln('- Bulbo de fotosensor encendido');
gotoxy(10,10);writeln('- Cartucho de prueba correctamente cargado');
gotoxy(10,12);writeln('- Probador conectado correctamente');
gotoxy(10,20);writeln('Presionar enter para continuar');
readln;
clr;
{ aqui se llama a cready.com para checar si ready = lo (activo) }
estat:=cready(a);
if estat > 0
  then
    begin
      gotoxy(10,4);writeln('ready no esta activa. ');
      gotoxy(10,6);writeln('checar circuito servo y corregir falla');
      gotoxy(10,8);writeln('presionar ENTER para continuar');
      readln;
  {   choice:='9';  para abortar al retornar }
    end
```

```
  else
  begin
  gotoxy(10,2);writeln('verificar que la cinta este en movimiento mientras');
  gotoxy(10,3);writeln('este mensaje aparezca en pantalla');
  gotoxy(10,8);writeln('si esto no sucede, interrumpir el programa, y checar:');
  gotoxy(13,9);writeln('- circuito "servo" si la cinta no se movio');
  gotoxy(13,10);writeln('- circuito de "EW" en tarjeta de fotosensor si el');
  gotoxy(13,11);writeln('  mensaje "EW detectado" no aparece abajo y el');
  gotoxy(13,12);writeln('  programa sigue corriendo despues que la cinta pare');
    gotoxy(13,13);writeln('- circuito de BOT si el mensaje "EW detectado"
aparece');
   gotoxy(13,14);writeln('  y el programa sigue corriendo despues que la cinta
pare');
  gotoxy(30,16);writeln('regresando cinta');
  rever;          { aqui se llama a rever.com }
  estat:=0;
  while estat = 0 do  { esperar por EW }
```

```
  begin
    estat:=cew(a); { aqui se llama a cew.com };
    gotoxy(30,16);writeln('regresando cinta');
  end; { while }
  gotoxy(10,18);writeln('EW detectado');
  esta2:=0;
  while esta2 = 0 do { esperar por BOT }
   begin
    esta2:=cbot(a); { aqui se llama a cbot.com };
    gotoxy(30,16);writeln('regresando cinta');
   end; { while }
  clr;
  gotoxy(10,8);writeln('BOT (principio de cinta) detectado');
  gotoxy(10,10);writeln('Cinta detenida? (S/N)');
  read(choice);
  case choice of
    'S', 's' : begin
```

```
            gotoxy(10,12);
            writeln('correcto, presionar ENTER para continuar');
            readln(dummy2);
           end;
      'N', 'n' : begin
              gotoxy(10,12);
              writeln('Falla en circuito SERVO y de INTERLOCK');
              gotoxy(10,14);
              writeln('Checar y corregir falla antes de correr pruebas de nuevo');
              gotoxy(10,16);
              writeln('presionar ENTER para continuar');
              readln;
              choice := '9'; { para abortar programa al retornar }
             end;
    end; { case }
  end  { else }
end;  { rewinds }
```

```
procedure fforwards;
  var
    a,estat,term : integer;
  begin
  clr;
  gotoxy(6,2);writeln('Antes de comenzar, favor de verificar:');
  gotoxy(10,6);writeln('- Fuentes de voltaje conectadas y encendidas');
  gotoxy(10,8);writeln('- Bulbo de fotosensor encendido');
  gotoxy(10,10);writeln('- Cartucho de prueba correctamente cargado');
  gotoxy(10,12);writeln('- Probador conectado correctamente');
  gotoxy(10,20);writeln('Presionar enter para continuar');
  readln;
  clr;
     { aqui se llama a cready.com para checar si ready = lo (activo) }
  estat:=cready(a);
  if estat > 0
    then
```

```
    begin
      gotoxy(10,4);writeln('ready no esta activa. ');
      gotoxy(10,6);writeln('checar circuito servo y corregir falla');
      gotoxy(10,8);writeln('presionar ENTER para continuar');
      readln;
    { choice:='9'; para abortar al retornar }
     end
  else
begin
gotoxy(10,2);writeln('verificar que la cinta este en movimiento mientras');
gotoxy(10,3);writeln('este mensaje aparezca en pantalla');
gotoxy(10,8);writeln('si esto no sucede, interrumpir el programa, y checar:');
gotoxy(13,9);writeln('- circuito "servo" si la cinta no se movio');
gotoxy(13,10);writeln('- circuito de "EW" en tarjeta de fotosensor si el');
gotoxy(13,11);writeln(' mensaje "EW detectado" no aparece abajo y el');
gotoxy(13,12);writeln(' programa sigue corriendo despues que la cinta pare');
```

```
    gotoxy(13,13);writeln('- circuito de EOT si el mensaje "EW detectado"
aparece');
   gotoxy(13,14);writeln('  y el programa sigue corriendo despues que la cinta
pare');
  gotoxy(26,16);writeln('adelantando cinta');
  forwar;{ aqui se llama a forwar.com }
  estat:=0;
  while estat = 0 do  { esperar por EW }
   begin
    estat:=cew(a);{ aqui se llama a cew.com }
    gotoxy(26,16);writeln('adelantando cinta');
   end;
  gotoxy(10,18);writeln('EW detectado');
  while estat > 0 do  { esperar por EOT }
   begin
    estat:=ceot(a);{ aqui se llama a ceot.com }
    gotoxy(26,16);writeln('adelantando cinta');
```

```
  end;
clr;
gotoxy(10,8);writeln('Fin de cinta detectado');
gotoxy(10,10);writeln('Cinta detenida? (S/N)');
read(choice);
case choice of
  'S', 's' : begin
        gotoxy(10,12);
        writeln('correcto, presionar ENTER para continuar');
        readln(dummy2);
       end;
  'N', 'n' : begin
          gotoxy(10,12);
          writeln('Falla en circuito SERVO y de INTERLOCK');
          gotoxy(10,14);
          writeln('Checar y corregir falla antes de correr pruebas de nuevo');
          gotoxy(10,16);
```

```
                writeln('presionar ENTER para continuar');
                readln;
                choice := '9'; { para abortar programa al retornar }
              end;
   end; { case }
   end { else }
  end; { forwards }
procedure tests;
  var
   a,este,prl1,prl2,prl3,prl4,term : integer;
  begin
  clr;
  gotoxy(10,4);writeln('insertar cartucho de cinta de prueba');
  gotoxy(10,6);writeln('presionar ENTER para continuar');
  readln;
  ini37a; {inicializa puerto 037AH para poder leer correctamente}
  clr;
```

```
este:=cwperm(a);
if este = 0
 then
  begin
   gotoxy(10,2);writeln('falla al leer WRITE PERMIT');
  end
 else
  begin
   gotoxy(10,2);writeln('WRITE PERMIT leido correctamente');
 end;
gotoxy(10,4);writeln('regresando cinta');
rever;              { aqui se llama a rever.com }
este:=0;
while este = 0 do
 begin
   este:=cbot(a);          { aqui se llama a cbot.com }
   gotoxy(10,4);writeln('regresando cinta');
```

```
 end; {while}
deten;   { aun cuando la cinta esta detenida,necesita este para }
        { poder iniciar movimiento otra vez }
gotoxy(10,6);writeln('escribiendo a cinta para checar BLOCK MARK');
este:=wrdri(a);  { aqui se llama a wrdri.com para escribir a cinta }
if este = 0      { pin 14 = 0 Si block mark no fue detectado }
  then
    begin
      gotoxy(10,8);writeln('BLOCK MARK no fue detectado');
      gotoxy(10,10);writeln('detener el programa');
      gotoxy(10,12);writeln('y checar circuitos BLOCK MARK Generator y');
      gotoxy(10,14);writeln('READ Amplifier');
      gotoxy(10,20);writeln('presionar ENTER para continuar');
    { choice:='9'; }
      deten;  { poner cinta en estado de reposo }
      readln;
    end
```

```
  else
    begin
      gotoxy(10,14);writeln('BLOCK MARK detectado');
    end;
  deten;
  gotoxy(10,24);writeln('regresando cinta');
  rever;              { aqui se llama a rever.com }
  este:=0;
  while este = 0 do
   begin
     este:=cbot(a);          { aqui se llama a cbot.com }
     gotoxy(10,24);writeln('regresando cinta');
   end;  {while}
  clr;
  deten;
  gotoxy(10,2);writeln('escribiendo a track 1');
  wrtrk1;
```

```
gotoxy(10,3);writeln('escribiendo a track 2');

wrtrk2;

gotoxy(10,4);writeln('escribiendo a track 3');

wrtrk3;

gotoxy(10,5); writeln('escribiendo a track 4');

wrtrk4;

gotoxy(26,7);writeln('regresando cinta');

rever;

este:=0;

while este=0 do

 begin

  este:=cbot(a);

  gotoxy(26,7);writeln('regresando cinta');

 end; { while }

deten;

ini37a;

gotoxy(10,9);writeln('si la cinta y el programa no paran significa que');
```

```
gotoxy(10,10);writeln('read strobe no esta presente.');
gotoxy(10,11);writeln('en ese caso, checar circuito READ DECODER, ');
gotoxy(10,12);writeln('READ AMPLIFIER y cabeza de lectura');
gotoxy(23,14);writeln('leyendo de cinta en track 1');
pr11:=leetrk1(a);
gotoxy(23,15);writeln('leyendo de cinta en track 2');
pr12:=leetrk2(a);
gotoxy(23,16);writeln('leyendo de cinta en track 3');
pr13:=leetrk3(a);
gotoxy(23,17);writeln('leyendo de cinta en track 4');
pr14:=leetrk4(a);
{ este:=lee(a);        aqui se llama a lee.com }
case pr11 of
 0 : begin
      gotoxy(10,19);writeln('error de lectura en track 1');
     { choice:='9'; }
     end;
```

```
    1 : begin
          gotoxy(10,19);writeln('lectura correcta en track 1');
        end;
    end;  { case }
    case prl2 of
     0 : begin
          gotoxy(10,20);writeln('error de lectura en track 2');
        end;
     1 : begin
          gotoxy(10,20);writeln('lectura correcta en track 2');
        end;
    end; { case }
    case prl3 of
     0 : begin
          gotoxy(10,21);writeln('error de lectura en track 3');
        end;
     1 : begin
```

```
            gotoxy(10,21);writeln('lectura correcta en track 3');
          end;
        end; {case}
        case prl4 of
         0 : begin
            gotoxy(10,22);writeln('error de lectura en track 4');
          end;
         1 : begin
            gotoxy(10,22);writeln('lectura correcta en track 4');
          end;
        end; { case }
        deten;
        gotoxy(10,24);writeln('fin de pruebas de lectura y escritura');
        gotoxy(10,25);writeln('presionar ENTER para continuar');
        readln;
      end;  { tests }
    procedure lecom;
```

```
  var
    term, tempor : integer;
  begin
  clr;
  gotoxy(10,2);writeln('lectura continua de datos');
  gotoxy(10,6);writeln('presionar ENTER para continuar');
  readln;
  clr;
  gotoxy(10,6);writeln('leyendo de cinta');
  leec;   { aqui se llama a leec.com }
  end;  { lecont }
begin
gotoxy(15,9);writeln('Probador para DRE-5000'); writeln(' ');
gotoxy(15,11);writeln('Diseñado por Pedro Octavio Diez de Sollano');
gotoxy(15,13);writeln('Director de Tesis: Ing. Antonio Herrera Mejia');
gotoxy(15,15);writeln('Facultad de Estudios Superiores Cuautitlan');
gotoxy(15,17);writeln('Universidad Nacional Autonoma de Mexico');
```

```
gotoxy(20,24);writeln('Presionar ENTER para continuar.');

deten; {para tener interfase preparada para dre-5000 en reposo}

readln;

clr;

gotoxy(8,4);

writeln('Los siguientes puntos deberan checarse antes de comenzar las pruebas:');

gotoxy(8,8);

writeln('a) La DRI-5000 debera estar conectada a la interfase de prueba');

gotoxy(8,10);writeln('b) La fuente de voltaje debe estar encendida');

gotoxy(8,12);

writeln('c) Un cartucho de prueba y uno de referencia deben estar disponibles');

gotoxy(5,24);

writeln('Presionar ENTER para continuar, ^C para terminar programa.');

readln;

clr;

choice:='a';

while choice <> '9' do
```

```
begin
 deten;  { poner cinta en reposo siempre que se llegue a este menu }
 gotoxy(30, 10);writeln('MENU');
 gotoxy(25, 12);writeln('1) Regresar cinta');
 gotoxy(25, 13);writeln('2) Adelantar cinta');
 gotoxy(25, 14);writeln('3) Pruebas de lectura y escritura');
 gotoxy(25, 15);writeln('4) Lectura continua');
 gotoxy(25, 16);writeln('9) Terminar pruebas');
 gotoxy(12, 21);writeln('Introduzca numero de opcion deseada: ');
 read(choice);
 case choice of
   '1' : rewinds;
   '2' : fforwards;
   '3' : tests;
   '4' : lecont;
 end;  { case }
clr;
```

```
 end;  { while }

1 : gotoxy(23,12);writeln('fin de pruebas');

deten;

end.
```

4.3 Las Rutinas de Lenguaje Ensamblador.

En la sección anterior se describió el programa principal elaborado en Pascal y se vió la manera en que este hace llamados a las subrutinas en lenguaje ensamblador. Estas rutinas serán ahora aquí discutidas.

4.3.1 La rutina cready.asm:

```
code  segment        'code'

      assume cs:code

creadyproc near      ;debe ser near porque turbo pascal usa

                              ;el modelo de memoria pequeno del 8088


      push bp

      mov bp,sp

      and            al,00000000b  ; limpia acumulador

      mov            dx,379H       ; checa puerto H037B
```

```
        in          al,dx
        and     ax,0040H ; enmascara para leer pin 10 (bit 6)
        pop bp
        ret                                     ; regresa valor leido
creadyendp
code ends
        end
```

Esta rutina comienza por salvar el stack, despues pone el acumulador en ceros y mueve el valor hexadecimal 379 al registro dx. El valor 379 hexadecimal corresponde a la dirección del puerto paralelo de la computadora personal IBM o compatible, en donde será leída la información de las señales de la grabadora de datos DRE-5000 a través del probador aquí diseñado. Una vez que el valor se encuentra en el registro dx se efectúa la instrucción `in' que lee la información del puerto y la almacena en la parte baja del acumulador (1 byte). Despues se utiliza una instrucción `and' para hacer ceros la parte alta del acumulador y enmascarar el bit 6 de modo que solo el valor de este sea considerado. El valor contenido en el acumulador será leído por el programa principal cuando la rutina termine y regrese a este. El valor que el programa principal esperará recibir como indicación de que la señal ready se encuentra activa es un 0. Como se recordará, en la sección de diseño de los circuitos se vió que la señal ready de la grabadora de datos DRE-5000

quedaría conectada al pin 10 del puerto paralelo de la computadora personal IBM o compatible, el cual corresponde al bit 6 del puerto que se encuentra en la dirección 379 hexadecimal. Para terminar, la rutina recupera el stack que había salvado utilizando la instrucción 'pop'.

4.3.2 La rutina rever.asm:

```
code segment        'code'
assume cs:code
rever proc near                 ; rever pone dri5000 en reversa rapida
     push bp
     mov bp,sp
     mov            ax,007CH    ; pone valor para reversa rapida en ax
     mov            dx,378H     ; direccion de puerto de salida
     out  dx,al                 ; escribe valor para reversa rapida en puerto
     pop bp
     ret                        ; regresa
rever endp
code ends
```

```
end
```

Como se vió en la sección de diseño de los circuitos, el valor necesario para poner a la grabadora de datos DRE-5000 en movimiento de reversa rápida es 7C hexadecimal. Este valor es puesto en el acumulador y el valor 378 hexadecimal es puesto en el registro dx. Este último valor corresponde a la dirección del puerto de salida dentro del puerto paralelo de la computadora personal que es utilizado para controlar las señales de entrada de la DRE-5000, entre las que están incluidas RUN y REVER. Una vez que esta rutina pone el valor para el movimiento en el puerto mediante la instrucción 'out', recupera el stack y regresa al programa principal.

4.3.3 La rutina cew.asm:

```
code  segment       'code'
      assume cs:code
cew   proc near
      push bp
      mov bp,sp
      and           al,0000H      ; limpia acumulador
      mov           dx,379H       ; lee puerto H0379
```

```
        in          al,dx
        and   ax,0020H      ; enmascara para leer pin 12 (bit 5)
        pop bp
        ret                                   ; regresa valor leido
cew     endp
code    ends
        end
```

Esta rutina pone el valor de puerto 379 hexadecimal para leer en el acumulador la información de las señales de la DRE-5000. Esta información es enmascarada para dejar únicamente la correspondiente al bit 5 (pin 12), que según se vió en la sección de diseño de los circuitos corresponde a la señal EW de la grabadora de datos. El valor leído por esta rutina es regresado en el acumulador al programa principal.

4.3.4 La rutina cbot.asm:

```
code    segment     'code'
        assume cs:code
cbot    proc near   ;debe ser near porque turbo pascal usa
                    ;el modelo de memoria pequeno del 8088
```

```
        push bp
        mov bp,sp
        and     al,0000H    ; limpia acumulador
        mov     dx,379H     ; lee puerto H0379
        in      al,dx
        and     ax,0010H    ; enmascara para leer pin 13 (bit 4) BOT
        pop bp
        ret                 ; regresa valor leido
cbot    endp
code    ends
        end
```

Esta rutina hace prácticamente el mismo trabajo que la rutina anterior, con la excepción que ahora el bit enmascarado es el bit 4 (pin 13), el cual corresponde a la señal BOT de la grabadora de datos.

4.3.5 La rutina ceot.asm:

```
code segment    'code'
```

```
        assume cs:code
ceot    proc near
        push bp
        mov bp,sp
        and     al,0000H    ; limpia acumulador
        mov     dx,379H     ; lee puerto H0379
        in      al,dx
        and     ax,0080H    ; enmascara para leer pin 11 (bit 7) EOT
        pop bp
        ret                 ; regresa valor leido
ceot    endp
code    ends
        end
```

Esta rutina es la que se encarga de verificar que la señal EOT se produzca. Como se explicó en la sección del diseño de los circuitos, esta señal queda conectada al pin 11 del puerto paralelo de la computadora personal. Este pin 11 corresponde al bit 7 del puerto 379 hexadecimal (como se explicó tambien anteriormente). La

rutina lee la información de este puerto, enmascara el bit 7 y regresa la información al programa principal.

4.3.6 La rutina cwperm.asm:

```
code  segment      'code'
      assume cs:code          ;checa valor de write permit
cwperm             proc near
      push bp
      mov bp,sp
      and          al,0000H   ; limpia acumulador
      mov          dx,37AH    ; lee puerto H037A
      in           al,dx
      and          ax,0001H   ; enmascara para leer pin 1  (bit 1)
      pop bp
      ret                     ; regresa valor leido
cwperm             endp
code  ends
```

```
        end
```

Esta rutina tambien lee la información del puerto en la dirección 37A hexadecimal, pero enmascara el bit 1, el cual corresponde al pin 1 del puerto paralelo de la computadora que a su vez está conectado a la señal `WRITE PERMIT' de la grabadora de datos DRE-5000.

4.3.7 La rutina deten.asm:

```
. code  segment      'code'
        assume cs:code
        org 100H
    BEGIN:jmp deten
    ; ---------------------------------------------
    deten proc near
        push bp
        mov bp,sp
        mov     dx,0378H    ; puerto para escritura
        mov     ax,0FFH     ; prepara para detener cinta
        out     dx,ax       ; detiene cinta
```

```
        pop bp
        ret                         ; regresa
deten endp
code  ends
        end
```

Esta rutina escribe el valor FF hexadecimal en el puerto de salida en la dirección 378 hexadecimal. El valor FF en este puerto pone a la grabadora de datos DRE-5000 en estado de reposo como se vió en el capítulo de diseño de los circuitos. La dirección 378 hexadecimal es puesto en el registro dx, y el valor FF hexadecimal tambien es puesto en el acumulador. El valor se escribe al puerto por medio de la instrucción `out'.

4.3.8 La rutina Init37a.asm:

```
code  segment       'code'
        assume cs:code
        org 100H
BEGIN:jmp init37a
; ---------------------------------------------
init37a proc near
```

```
        push bp
        mov bp,sp
        mov     dx,037AH    ; puerto para escritura
        mov     ax,04H      ; valor para inicializar puerto 037AH
        out     dx,ax       ; inicializa puerto
        pop bp
        ret                 ; regresa
init37endp
code ends
        end
```

Esta rutina pone el valor 4 hexadecimal en el puerto 37A hexadecimal. Esto tiene el objeto de inicializar los bits de este puerto, el cual como se recordará es de escritura/lectura. Las salidas del puerto son de colector abierto y esto es lo que permite que pueda ser utilizado para lectura. Sin embargo, los bits de este puerto deben ponerse de tal manera que su salida presente un nivel alto, el cual podrá ser cambiado a un nivel bajo por una señal externa, mientras que si su salida se encuentra en nivel bajo esta no podrá ser levantada por la señal externa. Este puerto cuenta con 4 bits (bit 0, 1, 2, 3), pero sólo el bit 2 lee un 1 cuando su salida es alta.

Los otros 3 bits leen un 0 cuando la salida es alta. Por esta razón se utiliza el valor 4 hexadecimal (valor 1 en el bit 2).

4.3.9 La rutina clr.asm:

```
cseg segment 'code'
        assume cs:cseg
clr proc near
        push ax                 ;salva registros
        push bx
        push cx
        push dx
        mov cx,0                ;comienzo en 0,0
        mov dh,24               ;final en la fila 24
        mov dl,79               ;columna 79
        mov ah,6                ;pone la opcion scroll
        mov al,0
        mov bh,7
```

```
        int 10h
        pop dx                  ;restaura registros y sale
        pop cx
        pop bx
        pop ax
        ret
clr endp
cseg ends
        end
```

Esta rutina borra la pantalla utilizando una interrupción de BIOS del sistema operativo MS-DOS o PC-DOS. Comienza por salvar los registros ya que serán utilizados, y sus valores serán alterados. Pone el valor 6 en la parte alta del acumulador para indicar que utilizará la opción scroll de la interrupción. Pone el valor 24 en el registro dh para indicar que finalize en la fila 24 y el valor 79 en del para indicar columna 79 (pantalla de 25x80). Una vez con los valores necesarios llama a la interrupción 10h de BIOS que efectúa un desplazamiento de líneas y columnas que tiene como resultado limpiar la pantalla. Se utiliza esta rutina ya que Pascal no cuenta con una instrucción propia para borrado de la pantalla.

4.3.10 La rutina wrdrl.asm:

```
code segment        'code'

        assume cs:code

        org 100H

BEGIN:jmp wrdri                 ; salta abajo de los datos

;-------------------------------------------------------------------

tempodw             ?

;-------------------------------------------------------------------

wrdri proc near                 ; escribe datos a la dri5000


        push bp

        push cx

        mov bp,sp

        mov             ax,00B1H    ; pone valor para movimiento de escritura

        mov             dx,378H     ; direccion de puerto de salida

        out   dx,al                 ; comienza movimiento de cinta

        push            ax          ; salva registros

        push            cx
```

```
        push    dx
        mov     ah,02H      ; retardo para asegurar mecanismo este listo
        int     1AH         ; pregunta la hora a dos
        mov     tempo,dx    ; almacena tiempo
retard: movah,02H           ; comienza loop de retardo aqui
        int     1AH         ; checa si ha transcurrido el tiempo limite
        sub     dx,tempo    ; tiempo actual - inicial
        sub     dx,07D0H    ; menor de 2 mil jump
        jb      retard
        pop     dx          ; retardo terminado, restaura registros
        pop     cx
        pop     ax
        mov     cx,14       ; prepara loop de preamble 15 ceros y un uno
        or      ax,0008H    ; cambia unicamente bit 3 write strobe a uno
ceros:out       dx,al       ; inicia write strobe de 50 ciclos 10.4uS
        push    cx          ; solo para tener 15 ciclos de retardo
        pop     cx          ; 12 ciclos mas de retardo, total = 27
```

```
and     ax,00FFH    ; 4 ciclos mas de retardo, total = 31
cmp     ax,cx       ; 3 ciclos mas de retardo, total = 34
and     ax,00F7H    ; hace bajo a write strobe,4 ciclos, tot=38
out     dx,al       ; write strobe baja. 12 ciclos mas. total=50
or      ax,0008H    ; prepara pulso alto. 4 ciclos
dec     cx          ; 2 ciclos de retardo. total = 6
cmp     ax,cx       ; 3 ciclos de retardo. total = 9
cmp     cx,ax       ; 3 ciclos de retardo. total = 12
cmp     ax,cx       ; 3 ciclos de retardo. total = 15
test    ax,00FFH    ; 5 ciclos de retardo, total = 20
jcxz    ceros       ; 18 ciclos si jump, 6 si no. normal total=38
                    ; out en ceros: dara 12 ciclos mas, total=50
out     dx,al       ; escribe ultimo cero,complem. 50 si no jump
or      ax,0004H    ; write data one =1. 4 ciclos
and     ax,00F7H    ; prepara strobe bajo, 4 ciclos, total=8
push    cx          ; 15 ciclos de retardo, total=23
pop     cx          ; 12 ciclos de retardo, total=35
```

```
        cmp     ax,cx       ; 3 ciclos de retardo, total=38
        out     dx,al       ; hace strobe bajo,wdo alto,12 ciclos,total=50
        or      ax,0008H    ; prepara strobe alto. 4 ciclos
        push    cx          ; 15 ciclos retardo. total=19
        pop     cx          ; 12 ciclos retardo. total=31
        mov     cx,0FFFFH   ; 4 ciclos.total=35.prepara loop de unos
        cmp     ax,cx       ; 3 ciclos retardo. total=38
unos:   out     dx,al       ; 12, total=50.escribe uno.fin de preamble
        push    cx          ; solo para tener 15 ciclos de retardo
        pop     cx          ; 12 ciclos mas de retardo, total = 27
        and     ax,00FFH    ; 4 ciclos mas de retardo, total = 31
        cmp     ax,cx       ; 3 ciclos mas de retardo, total = 34
        and     ax,00F7H    ; hace bajo a write strobe. 4 ciclos, tot=38
        out     dx,al       ; write strobe baja. 12 ciclos mas. total=50
        or      ax,0008H    ; prepara pulso alto. 4 ciclos
        dec     cx          ; 2 ciclos de retardo. total = 6
        cmp     ax,cx       ; 3 ciclos de retardo. total = 9
```

```
        cmp     cx,ax       ; 3 ciclos de retardo. total = 12
        cmp     ax,cx       ; 3 ciclos de retardo. total = 15
        test    ax,00FFH    ; 5 ciclos de retardo, total = 20
        jcxz    unos        ; loop para escribir unos
        mov     dx,037AH    ; para leer block mark
        in      al,dx       ; lee el puerto
        and     ax,0002H    ; enmascara para leer bit2, pin 14
        mov     dx,0378H    ; detener cinta antes de salir
        push    ax          ; salva valor de ax
        mov     ax,00F2H    ; valor para detener cinta
        out     dx,al       ; detiene cinta
        pop     ax          ; recupera valor a regresar
        pop     cx
        pop bp
        ret                 ; regresa
wrdri endp
code  ends
```

end

Esta rutina es utilizada para escribir una serie de 15 ceros y 1 uno (preamble) a la cinta de la grabadora de datos DRE-5000. Esta rutina comienza por poner a la cinta en movimiento de escritura. y despues espera un tiempo de retardo para asegurar que el mecanismo de la DRE-5000 alcance la velocidad normal de escritura. Como se recordará, en las especificaciones de la DRE-5000 se indica que este tiempo es de 30 milisegundos mínimo. Para este retardo se hace uso de un llamado a la interrupción 1AH a BIOS del sistema operativo MS-DOS o PC-DOS según el caso. Esta interrupción lee el reloj de la computadora. Y es utilizada de la siguiente manera; se lee el reloj al iniciar y se salva esta información en la variable tempo. Posteriormente se entra en un ciclo en el que se permanece leyendo continuamente el reloj de la computadora personal, hasta que han transcurrido mas de dos mil cuentas. Una vez que estas han transcurrido, se termina el ciclo, se restauran los valores de los registros (los cuales habian sido salvados con anterioridad mediante instrucciones 'push') y se comienza la escritura del 'preamble'. Como ya la DRE-5000 se encuentra en movimiento de escritura, es necesario únicamente cambiar el valor del bit 3 al valor que se escribió anteriormente al puerto de salida 378 hexadecimal, ya que el este bit corresponde a la señal WRITE STROBE (WRITE DATA permanecerá en cero hasta que se termine de imprimir los 15 ceros). El bit 3 es enmascarado por medio de una función 'and' con el valor 8 hexadecimal. La escritura se efectúa a 63 bits/mm, por lo que el ciclo de WRITE STROBE tendrá una duración de 20.83 uS, siendo el tiempo que permanece alto la mitad de este, o sea 10.42 uS. En el listado de la rutina se muestra el tiempo que toma cada instrucción en ciclos de máquina, así como el tiempo total para cada grupo de

instrucciones, que es de 50 ciclos. Ya que cada ciclo toma 1/4.77 MHz cada ciclo representa en tiempo 0.209 uS, por lo que 50 ciclos equivale a 10.4 uS que es el tiempo requerido. Para cumplir los 50 ciclos, se introdujeron algunas instrucciones que no tienen objetivo alguno otro que el de proporcionar ciclos adicionales. Por ejemplo algunas combinaciones de 'push' y 'pop', y algunas otras como 'cmp'. Por último, la rutina lee el estado de la señal 'BLOCK MARK' (bit 2 del puerto 37A) y pasa este valor al programa principal.

4.3.11 La rutina wrtrk1.asm:

```
code segment    'code'
        assume cs:code
        org     100H
BEGIN:JMP       SHORT WRTRK1
; ----------DEFINICIONES ------ ----------
TEXTODB         30H,30H,30H,30H,30H;tabla de datos a escribir
        DB      30H,30H,30H,30H,30H;comenzando
        DB      30H,30H,30H,30H,30H;por el preamble (15 ceros y un 1)
        DB      31H,2EH,2EH,2EH,2EH;1....
        DB      22H,45H,73H,74H,61H;"Esta       TABLA DE TEXTO
```

```
DB      20H,65H,73H,20H,75H; es u
DB      6EH,61H,20H,70H,72H;na pr
DB      75H,65H,62H,61H,20H;ueba
DB      70H,61H,72H,61H,20H;para
DB      6CH,61H,20H,67H,72H;la gr
DB      61H,62H,61H,64H,6FH;abado
DB      72H,61H,20H,44H,52H;ra DR
DB      45H,2DH,35H,30H,30H;E-500
DB      30H,2CH,20H,65H,6EH;0, en
DB      20H,65H,6CH,20H,74H; el t
DB      72H,61H,63H,6BH,20H;rack
DB      6EH,75H,6DH,65H,72H;numer
DB      6FH,20H,75H,6EH,6FH;o uno
DB      22H,2EH,2EH,2EH,2EH;"....
DB      2EH,2EH,2EH,2EH,2EH;.....
;-------------------------------------------------
```

```
WRTRK1 proc near ;rutina para escribir a track 1 de cinta
        push ax      ;salva registros
        push bx      ;
        push dx      ;
        push cx      ;valor 00B1H pone cinta en track 1, movimiento de
        mov  ax,00B1H;de escritura, write data=0 y write strobe=bajo
        mov  dx,378H ;direccion de puerto en dx
        out  dx,al   ;comienza movimiento de cinta
        mov  ax,30   ;genera retardo de 30 veces 1 milisegundo
DELAY:mov  cx,109H;cuenta para retardo de 1 milisegundo
DLY1: dec  cx      ;decrementa cuenta
        jnz  DLY1    ;cicla si cuenta no igual a cero
        dec  ax      ;decrementa cuenta de milisegundos
        jnz  DELAY   ;otro milisegundo de retardo si no termino
        MOV  CL,0    ;termino delay.total elementos en tabla (0-99)
        MOV  CH,8    ;cuenta para shifts
        MOV  AL,0B1H;prepara valor write data y write enable bajos
```

```
        MOV BH,0   ;poner 00CL en BX [4]
CICLO:MOV BL,CL;pone en BX el indice de la tabla [2]
        MOV BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
ROTA:SHL BL,1   ;shift en BL para checar si msb es 0 o 1 [2]
        JC ES_UNO ;si carry msb=1, brinca. [16 o 4]
        JNC SIGUE  ;si msb=0, entonces sigue sin modificar AL [16]
ES_UNO: MOV AL,0B5H;[4 ciclos] pone write data one (bit 2) en 1
SIGUE:OUT DX,AL;escribe (salida,con valor write data) a puerto [12]
        OR AL,8H   ;prepara pulso de write strobe [4]
        OUT DX,AL ;pone pulso write strobe alto [12]
        PUSH CX    ;salva valor de CX [15]
        MOV CX,3   ;cuenta para retardo de 42 ciclos [4]
DLYBAJ: DEC CX ;loop de retardo [[2]]
        JNZ DLYBAJ; [[16 o 4]]. total retardo=42 ciclos [42]
        NOP        ;solo para retardo [3]
        NOP        ;solo para retardo [3]
        POP CX     ;recupera valor de CX [12]
```

```
        MOV DX,378H;direccion de puerto [4]
        MOV AL,0B1H;valor para write strobe y data bajos[4]
        OUT DX,AL ;Hace bajo a write strobe [12]
        DEC CH      ;decrementa contador de shift [3]
        JNZ COMPLE;brinca a completar ciclos antes nuevo shift [16/4]
        MOV CH,8H ;reinicializa contador de shifts [4]
        INC CL      ;incrementa cuenta [2]
        CMP CL,100 ;checar si ya termino [4]
        JNZ CICLO  ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP FIN     ;cuenta=100, termino, brinca a detener cinta
COMPLE: JNZ MAS;brinca para seguir completando [16]
MAS:JNZ ROTA   ;completo, brinca a seguir shifts [16]
FIN: MOV AL,0F3H;termino de escribir, ahora detener cinta [4]
        MOV DX,378H; [4]
        OUT DX,AL ;detiene cinta [12]
        pop cx      ;termino. hace pop de registros y regresa
        pop dx
```

```
        pop bx
        pop ax
        ret
WRTRK1endp
code ends
        end         BEGIN
```

Esta rutina comienza por definir el texto que será escrito a cinta por medio de instrucciones DB (define byte) para el ensamblador. Posteriormente comienza el movimiento de escritura en la DRE-5000 escribiendo el valor B1 hexadecimal al puerto 378H. Una vez comenzado el movimiento genera un retardo de 30 milisegundos por medio de 30 retardos de 1 milisegundo utilizando un loop de instrucciones. En esta rutina se efectúa la escritura a 31.5 bits/mm, de manera que el ciclo de tiempo que tomará WRITE STROBE será en este caso de 41.67 uS, siendo el tiempo que permanece alto igual al tiempo que permanece bajo, y este es igual a 20.83 uS. Anteriormente se vió que cada ciclo de máquina en la computadora toma 0.209 uS, por lo que 99 ciclos nos da un total de 20.7 uS. Como puede verse, el texto que se escribirá a cinta incluye referencia a que está escribiendose en el track número 1, de manera que sea posteriormente posible distinguir este texto de el escrito a otros tracks.

4.3.12 La rutina wrtrk2.asm:

```
code  segment      'code'

      assume cs:code

      org          100H

BEGIN:JMP          SHORT WRTRK2

;----------DEFINICIONES ------ ----------

TEXTODB            30H,30H,30H,30H,30H;tabla de datos a escribir comienza

      DB           30H,30H,30H,30H,30H;por el preamble (15 ceros y un 1)

      DB           30H,30H,30H,30H,30H;

      DB           31H,2EH,2EH,2EH,2EH;1....

      DB           22H,45H,73H,74H,61H;"Esta      TABLA DE TEXTO

      DB           20H,65H,73H,20H,75H; es u

      DB           6EH,61H,20H,70H,72H;na pr

      DB           75H,65H,62H,61H,20H;ueba

      DB           70H,61H,72H,61H,20H;para

      DB           6CH,61H,20H,67H,72H;la gr

      DB           61H,62H,61H,64H,6FH;abado

      DB           72H,61H,20H,44H,52H;ra DR
```

```
        DB          45H,2DH,35H,30H,30H;E-500
        DB          30H,2CH,20H,65H,6EH;0, en
        DB          20H,65H,6CH,20H,74H; el t
        DB          72H,61H,63H,6BH,20H;rack
        DB          6EH,75H,6DH,65H,72H;numer
        DB          6FH,20H,64H,6FH,73H;o dos
        DB          22H,2EH,2EH,2EH,2EH;"....
        DB          2EH,2EH,2EH,2EH,2EH;.....
; ---------------------------------------------------
WRTRK2 proc near ;rutina para escribir a track 2 de cinta
        push ax     ;salva registros
        push bx     ;
        push dx     ;
        push cx     ;valor 0091H pone cinta en track 2, movimiento
        mov  ax,0091H;de escritura, write data=0 y write strobe=bajo
        mov  dx,378H ;direccion de puerto en dx
        out  dx,al  ;comienza movimiento de cinta
```

```
        mov  ax,30     ;genera retardo de 30 veces 1 milisegundo
DELAY:mov  cx,109H;cuenta para retardo de 1 milisegundo
DLY1: dec  cx          ;decrementa cuenta
        jnz  DLY1      ;cicla si cuenta no igual a cero
        dec  ax        ;decrementa cuenta de milisegundos
        jnz  DELAY     ;otro milisegundo de retardo si no termino
        MOV  CL,0      ;termino delay.total elementos en tabla (0-99)
        MOV  CH,8      ;cuenta para shifts
        MOV  AL,091H;prepara valor write data y write enable bajos
        MOV  BH,0      ;poner 00CL en BX [4]
CICLO:MOV  BL,CL;pone en BX el indice de la tabla [2]
        MOV  BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
ROTA:SHL  BL,1     ;shift en BL para checar si msb es 0 o 1 [2]
        JC   ES_UNO ;si carry msb=1, brinca. [16 o 4]
        JNC  SIGUE  ;si msb=0, entonces sigue sin modificar AL [16]
ES_UNO: MOV  AL,095H;[4 ciclos] pone write data one (bit 2) en 1
SIGUE:OUT  DX,AL;escribe (salida,con valor write data) a puerto [12]
```

```
        OR  AL,8H   ;prepara pulso de write strobe [4]
        OUT DX,AL   ;pone pulso write strobe alto [12]
        PUSH CX     ;salva valor de CX [15]
        MOV CX,3    ;cuenta para retardo de 42 ciclos [4]
DLYBAJ: DEC CX      ;loop de retardo [[2]]
        JNZ DLYBAJ; [[16 o 4]]. total retardo=42 ciclos [42]
        NOP         ;solo para retardo [3]
        NOP         ;solo para retardo [3]
        POP CX      ;recupera valor de CX [12]
        MOV DX,378H;direccion de puerto [4]
        MOV AL,091H;valor para write strobe y data bajos[4]
        OUT DX,AL   ;Hace bajo a write strobe [12]
        DEC CH      ;decrementa contador de shift [3]
        JNZ COMPLE;brinca a completar ciclos antes nuevo shift [16/4]
        MOV CH,8H   ;reinicializa contador de shifts [4]
        INC CL      ;incrementa cuenta [2]
        CMP CL,100  ;checar si ya termino [4]
```

```
        JNZ  CICLO   ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP  FIN     ;cuenta=100, termino, brinca a detener cinta
COMPLE: JNZ  MAS;brinca para seguir completando [16]
MAS:JNZ  ROTA   ;completo, brinca a seguir shifts [16]
FIN:  MOV  AL,0D3H;termino de escribir, ahora detener cinta [4]
        MOV  DX,378H; [4]
        OUT  DX,AL ;detiene cinta [12]
        pop cx       ;termino. hace pop de registros y regresa
        pop dx
        pop bx
        pop ax
        ret                                    ;
WRTRK2endp
code  ends
        end          BEGIN
```

Esta rutina es muy similar a la mencionada anteriormente, con la diferencia de que ahora la escritura es en el track 2. El texto que será escrito a cinta incluye mención

a que se está escribiendo en este track. El valor que se pone en el puerto 378H es ahora 91 hexadecimal para hacer el cambio en la selección del track.

4.3.13 La rutina wrtrk3.asm:

```
code segment    'code'
    assume cs:code
    org         100H
BEGIN:JMP       SHORT wrtrk3
; ----------DEFINICIONES ------ ----------
TEXTODB         30H,30H,30H,30H,30H;tabla datos a escribir comenzando
    DB          30H,30H,30H,30H,30H;por el preamble (15 ceros y un 1)
    DB          30H,30H,30H,30H,30H;
    DB          31H,2EH,2EH,2EH,2EH;1....
    DB          22H,45H,73H,74H,61H;"Esta      TABLA DE TEXTO
    DB          20H,65H,73H,20H,75H; es u
    DB          6EH,61H,20H,70H,72H;na pr
    DB          75H,65H,62H,61H,20H;ueba
```

```
        DB      70H,61H,72H,61H,20H;para
        DB      6CH,61H,20H,67H,72H;la gr
        DB      61H,62H,61H,64H,6FH;abado
        DB      72H,61H,20H,44H,52H;ra DR
        DB      45H,2DH,35H,30H,30H;E-500
        DB      30H,2CH,20H,65H,6EH;0, en
        DB      20H,65H,6CH,20H,74H; el t
        DB      72H,61H,63H,6BH,20H;rack
        DB      6EH,75H,6DH,65H,72H;numer
        DB      6FH,20H,74H,72H,65H;o tre
        DB      73H,22H,2EH,2EH,2EH;s"...
        DB      2EH,2EH,2EH,2EH,2EH;.....
; ---------------------------------------------------

wrtrk3 proc near    ;rutina para escribir a track 3 de cinta
    push ax         ;salva registros
    push bx         ;
```

```
        push dx         ;
        push cx         ;valor 00A1H pone cinta en track 3, movimiento de
        mov  ax,00A1H;de escritura, write data=0 y write strobe=bajo
        mov  dx,378H ;direccion de puerto en dx
        out  dx,al      ;comienza movimiento de cinta
        mov  ax,30      ;genera retardo de 30 veces 1 milisegundo
DELAY:mov  cx,109H;cuenta para retardo de 1 milisegundo
DLY1: dec  cx           ;decrementa cuenta
        jnz  DLY1       ;cicla si cuenta no igual a cero
        dec  ax         ;decrementa cuenta de milisegundos
        jnz  DELAY      ;otro milisegundo de retardo si no termino
        MOV  CL,0       ;termino delay.total elementos en tabla (0-99)
        MOV  CH,8       ;cuenta para shifts
        MOV  AL,0A1H;prepara valor write data y write enable bajos
        MOV  BH,0       ;poner 00CL en BX [4]
CICLO:MOV  BL,CL;pone en BX el indice de la tabla [2]
        MOV  BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
```

```
ROTA:SHL  BL,1   ;shift en BL para checar si msb es 0 o 1 [2]
     JC   ES_UNO ;si carry msb=1, brinca. [16 o 4]
     JNC  SIGUE  ;si msb=0, entonces sigue sin modificar AL [16]
ES_UNO: MOV  AL,0A5H;[4 ciclos] pone write data one (bit 2) en 1
SIGUE:OUT DX,AL;escribe (salida,con valor write data) a puerto [12]
     OR   AL,8H  ;prepara pulso de write strobe [4]
     OUT  DX,AL  ;pone pulso write strobe alto [12]
     PUSH CX                  ;salva valor de CX [15]
     MOV  CX,3   ;cuenta para retardo de 42 ciclos [4]
DLYBAJ: DEC  CX ;loop de retardo [[2]]
     JNZ  DLYBAJ; [[16 o 4]]. total retardo=42 ciclos [42]
     NOP         ;solo para retardo [3]
     NOP         ;solo para retardo [3]
     POP  CX     ;recupera valor de CX [12]
     MOV  DX,378H;direccion de puerto [4]
     MOV  AL,0A1H;valor para write strobe y data bajos[4]
     OUT  DX,AL  ;Hace bajo a write strobe [12]
```

```
        DEC CH      ;decrementa contador de shift [3]
        JNZ COMPLE;brinca a completar ciclos antes nuevo shift [16/4]
        MOV CH,8H ;reinicializa contador de shifts [4]
        INC CL      ;incrementa cuenta [2]
        CMP CL,100 ;checar si ya termino [4]
        JNZ CICLO  ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP FIN     ;cuenta=100, termino, brinca a detener cinta
COMPLE: JNZ MAS;brinca para seguir completando [16]
MAS:JNZ ROTA   ;completo, brinca a seguir shifts [16]
FIN: MOV AL,0E3H;termino de escribir, ahora detener cinta [4]
        MOV DX,378H; [4]
        OUT DX,AL ;detiene cinta [12]
        pop cx      ;termino. hace pop de registros y regresa
        pop dx
        pop bx
        pop ax
        ret                                    ;
```

```
wrtrk3 endp

code  ends

      end          BEGIN
```

Nuevamente, esta rutina es muy similar a la de wrtrk1.asm y a wrtrk2.asm. El texto cambia para diferenciarse del escrito en los otros tracks, de manera que no haya confusión de un track por otro al leer el texto posteriormente. El valor escrito al puerto 378H es ahora A1 hexadecimal, el cual hace el cambio al track número 3.

4.3.14 La rutina wrtrk4.asm:

```
code  segment      'code'

      assume cs:code

      org          100H

BEGIN:JMP          SHORT WRTRK4

; ----------DEFINICIONES ------ ----------

TEXTODB            30H,30H,30H,30H,30H;tabla datos a escribir comenzando

      DB           30H,30H,30H,30H,30H;por el preamble (15 ceros y un 1)

      DB           30H,30H,30H,30H,30H;

      DB           31H,2EH,2EH,2EH,2EH;1....
```

```
DB      22H,45H,73H,74H,61H;"Esta     TABLA DE TEXTO
DB      20H,65H,73H,20H,75H; es u
DB      6EH,61H,20H,70H,72H;na pr
DB      75H,65H,62H,61H,20H;ueba
DB      70H,61H,72H,61H,20H;para
DB      6CH,61H,20H,67H,72H;la gr
DB      61H,62H,61H,64H,6FH;abado
DB      72H,61H,20H,44H,52H;ra DR
DB      45H,2DH,35H,30H,30H;E-500
DB      30H,2CH,20H,65H,6EH;0, en
DB      20H,65H,6CH,20H,74H; el t
DB      72H,61H,63H,6BH,20H;rack
DB      6EH,75H,6DH,65H,72H;numer
DB      6FH,20H,63H,75H,61H;o cua
DB      74H,72H,6FH,22H,2EH;tro".
DB      2EH,2EH,2EH,2EH,2EH;.....
; ------------------------------------------------
```

```
WRTRK4 proc near ;rutina para escribir a track 4 de cinta
    push ax      ;salva registros
    push bx      ;
    push dx      ;
    push cx      ;valor 0081H pone cinta en track 4, movimiento de
    mov  ax,0081H;de escritura, write data=0 y write strobe=bajo
    mov  dx,378H ;direccion de puerto en dx
    out  dx,al   ;comienza movimiento de cinta
    mov  ax,30   ;genera retardo de 30 veces 1 milisegundo
DELAY:mov cx,109H;cuenta para retardo de 1 milisegundo
DLY1: dec cx     ;decrementa cuenta
    jnz  DLY1    ;cicla si cuenta no igual a cero
    dec  ax      ;decrementa cuenta de milisegundos
    jnz  DELAY   ;otro milisegundo de retardo si no termino
    MOV  CL,0    ;termino delay.total elementos en tabla (0-99)
    MOV  CH,8    ;cuenta para shifts
```

```
        MOV  AL,081H;prepara valor write data y write enable bajos
        MOV  BH,0    ;poner 00CL en BX [4]
CICLO:MOV  BL,CL;pone en BX el indice de la tabla [2]
        MOV  BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
ROTA:SHL  BL,1    ;shift en BL para checar si msb es 0 o 1 [2]
        JC   ES_UNO ;si carry msb=1, brinca. [16 o 4]
        JNC  SIGUE  ;si msb=0, entonces sigue sin modificar AL [16]
ES_UNO: MOV  AL,085H;[4 ciclos] pone write data one (bit 2) en 1
SIGUE:OUT  DX,AL;escribe (salida,con valor write data) a puerto [12]
        OR   AL,8H   ;prepara pulso de write strobe [4]
        OUT  DX,AL ;pone pulso write strobe alto [12]
        PUSH CX      ;salva valor de CX [15]
        MOV  CX,3    ;cuenta para retardo de 42 ciclos [4]
DLYBAJ: DEC  CX ;loop de retardo [[2]]
        JNZ  DLYBAJ; [[16 o 4]]. total retardo=42 ciclos [42]
        NOP          ;solo para retardo [3]
        NOP          ;solo para retardo [3]
```

```
        POP CX      ;recupera valor de CX [12]
        MOV DX,378H;direccion de puerto [4]
        MOV AL,081H;valor para write strobe y data bajos[4]
        OUT DX,AL ;Hace bajo a write strobe [12]
        DEC CH      ;decrementa contador de shift [3]
        JNZ COMPLE;brinca a completar ciclos antes nuevo shift [16/4]
        MOV CH,8H ;reinicializa contador de shifts [4]
        INC CL      ;incrementa cuenta [2]
        CMP CL,100 ;checar si ya termino [4]
        JNZ CICLO  ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP FIN     ;cuenta=100, termino, brinca a detener cinta
COMPLE: JNZ MAS;brinca para seguir completando [16]
MAS:JNZ ROTA   ;completo, brinca a seguir shifts [16]
FIN:  MOV AL,0C3H;termino de escribir, ahora detener cinta [4]
        MOV DX,378H; [4]
        OUT DX,AL ;detiene cinta [12]
        pop cx      ;termino. hace pop de registros y regresa
```

```
        pop dx
        pop bx
        pop ax
        ret                                    ;
WRTRK4endp
code ends
        end         BEGIN
```

Prácticamente la misma rutina que en los casos anteriores, a excepción de los cambios en el valor que se escribe al puerto 378H del puerto paralelo de la computadora personal IBM o compatible (valor 81 hexadecimal), el cual selecciona el track 4. Tambien hay cambio en el texto para diferenciar este de el escrito en otros tracks.

4.3.15 La rutina leetrk1.asm:

```
code segment        'code'
        assume cs:code
        org         100H
BEGIN:JMP           SHORT LEETRK1
```

```
; ----------DEFINICIONES ------ ----------

TEXTODB     30H,30H,30H,30H,30H;tabla de datos para comparar con los
    DB      30H,30H,30H,30H,30H;datos leidos
    DB      30H,30H,30H,30H,30H;
    DB      31H,2EH,2EH,2EH,2EH;1....
    DB      22H,45H,73H,74H,61H;"Esta      TABLA DE TEXTO
    DB      20H,65H,73H,20H,75H; es u
    DB      6EH,61H,20H,70H,72H;na pr
    DB      75H,65H,62H,61H,20H;ueba
    DB      70H,61H,72H,61H,20H;para
    DB      6CH,61H,20H,67H,72H;la gr
    DB      61H,62H,61H,64H,6FH;abado
    DB      72H,61H,20H,44H,52H;ra DR
    DB      45H,2DH,35H,30H,30H;E-500
    DB      30H,2CH,20H,65H,6EH;0, en
    DB      20H,65H,6CH,20H,74H; el t
    DB      72H,61H,63H,6BH,20H;rack
```

```
        DB          6EH,75H,6DH,65H,72H;numer

        DB          6FH,20H,75H,6EH,6FH;o uno

        DB          22H,2EH,2EH,2EH,2EH;"....

        DB          2EH,2EH,2EH,2EH,2EH;.....

; ---------------------------------------------------

LEETRK1proc near ;rutina para leer del track 1 de cinta

        push ax     ;salva registros

        push bx     ;

        push dx     ;

        push cx     ;valor 00F1H pone cinta en track 1, movimiento de

        mov  ax,00F1H;de lectura

        mov  dx,378H ;direccion de puerto en dx

        out  dx,al   ;comienza movimiento de cinta

        mov  ax,25   ;genera retardo de 25 veces 1 milisegundo

DELAY:mov  cx,109H;cuenta para retardo de 1 milisegundo

DLY1: dec  cx       ;decrementa cuenta

        jnz  DLY1   ;cicla si cuenta no igual a cero
```

```
        dec ax          ;decrementa cuenta de milisegundos
        jnz DELAY       ;otro milisegundo de retardo si no termino
        MOV CL,0        ;termino delay.total elementos en tabla (0-99)
        MOV CH,8        ;cuenta para shifts
        MOV BH,0        ;poner 00CL en BX
CICLO:MOV BL,CL;pone en BX el indice de la tabla  [2]
        MOV BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
ROTA:SHL BL,1   ;shift en BL para checar si msb es 0 o 1  [2]
        JC ES_UNO ;si carry msb=1, brinca.  [16 o 4]
STRB:MOV DX,37AH;para checar si read strobe esta bajo  [4]
        IN AX,DX    ;lee status  [12]
        AND AX,08H;enmascara read strobe  [4]
        JZ STRB     ;mientras bit 3 sea 0, strobe esta alto  [16/4]
NOSTRB:IN AX,DX;despues que strobe se hace bajo, checar sea  [16/4]
        AND AX,08H;alto de nuevo. enmascara read strobe  [4]
        JNZ NOSTRB;cuando bit 3 sea 0, entonces read strobe =1 [16/4]
        IN AX,DX    ;lee de nuevo para valor de read data  [12]
```

```
        AND AX,04H;enmascara bit 2  [4]
        JNZ MAL     ;si read data no es 0 entonces hay error [16/4]
        JMP SIGUE   ;si read data fue 0 continua  [15]
ES_UNO: IN  AX,DX;lee de nuevo para valor de read data  [12]
        AND AX,04H;enmascara bit 2  [4]
        JZ  MAL     ;si read data no es 1 entonces hay error [16/4]
SIGUE:DEC CH    ;decrementa contador de shift  [3]
       JNZ ROTA;continua shifts si no han terminado  [16/4]
        MOV CH,8H ;reinicializa contador de shifts  [4]
        INC CL      ;incrementa cuenta  [2]
        CMP CL,100 ;checar si ya termino  [4]
        JNZ CICLO  ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP EXITO     ;cuenta=100, termino, brinca a detener cinta [15]
MAL:MOV AL,0F3H;detiene cinta  [4]
        MOV DX,378H; [4]
        OUT DX,AL ; [12]
        MOV AX,00H;para pasar valor de cero a Pascal (0=error) [4]
```

```
        JMP FIN     ; [15]
EXITO:MOV  AL,0F3H;termino de escribir, ahora detener cinta [4]
        MOV  DX,378H; [4]
        OUT  DX,AL  ;detiene cinta [12]
        MOV  AX,01H;pasa valor de uno a Pascal (1=exito)  [4]
FIN:  pop cx        ;termino. hace pop de registros y regresa
        pop dx
        pop bx
;       pop ax      ;no es necesario ax
        ret                                 ;
LEETRK1endp
code  ends
        end         BEGIN
```

En esta rutina se define el mismo texto que fue escrito a la cinta por la rutina wrtrk1.asm, ya que los bits leídos por leetrk1.asm serán agrupados en bytes que serán comparados uno por uno con los de este texto para verificar si hay errores o no. La cinta es puesta en movimiento de lectura al escribir el valor F1 hexadecimal al puerto 378H del puerto paralelo de la computadora personal. Posteriormente se

espera un tiempo de 25 milisegundos antes de comenzar a leer de la cinta. El proceso de lectura consiste en tomar un byte de la tabla donde fue definido el texto, ponerlo en el registro BX y despues hacer un corrimiento de un bit hacia la izquierda (el bit mas significante pasa al carry) para checar si el MSB es 0 o 1. Despues esperar a que WRITE STROBE se haga verdadero, leer el valor de WRITE DATA y compararlo con el valor obtenido del MSB obtenido anteriormente. En caso de error se brincará al procedimiento 'MAL'. Si no hay error se continuará hasta que se hayan corrido todos los bits del byte, y despues se repetirá con los demás bytes de la tabla del texto. Al terminar se regresará un valor al programa principal dependiendo de si hubo o no error al efectuar la lectura. El valor es puesto por la rutina en el acumulador, y será 0 si hubo error o 1 si hubo éxito.

4.3.16 La rutina leetrk2.asm:

```
code segment    'code'
        assume cs:code
        org     100H
BEGIN:JMP       SHORT leetrk2
; ----------DEFINICIONES ------ ----------
TEXTODB         30H,30H,30H,30H,30H;tabla de datos para comparar con los
        DB      30H,30H,30H,30H,30H;datos leidos
```

```
DB      30H,30H,30H,30H,30H;
DB      31H,2EH,2EH,2EH,2EH;1....
DB      22H,45H,73H,74H,61H;"Esta      TABLA DE TEXTO
DB      20H,65H,73H,20H,75H; es u
DB      6EH,61H,20H,70H,72H;na pr
DB      75H,65H,62H,61H,20H;ueba
DB      70H,61H,72H,61H,20H;para
DB      6CH,61H,20H,67H,72H;la gr
DB      61H,62H,61H,64H,6FH;abado
DB      72H,61H,20H,44H,52H;ra DR
DB      45H,2DH,35H,30H,30H;E-500
DB      30H,2CH,20H,65H,6EH;0, en
DB      20H,65H,6CH,20H,74H; el t
DB      72H,61H,63H,6BH,20H;rack
DB      6EH,75H,6DH,65H,72H;numer
DB      6FH,20H,64H,6FH,73H;o dos
DB      22H,2EH,2EH,2EH,2EH;"....
```

```
        DB          2EH,2EH,2EH,2EH,2EH;.....
;-------------------------------------------------

leetrk2 proc near   ;rutina para leer del track 2 de cinta
        push ax     ;salva registros
        push bx     ;
        push dx     ;
        push cx     ;valor 00D1H pone cinta en track 2, movimiento de
        mov  ax,00D1H;de lectura
        mov  dx,378H ;direccion de puerto en dx
        out  dx,al  ;comienza movimiento de cinta
        mov  ax,25  ;genera retardo de 25 veces 1 milisegundo
DELAY:mov  cx,109H;cuenta para retardo de 1 milisegundo
DLY1: dec  cx       ;decrementa cuenta
        jnz  DLY1   ;cicla si cuenta no igual a cero
        dec  ax     ;decrementa cuenta de milisegundos
        jnz  DELAY  ;otro milisegundo de retardo si no termino
```

```
        MOV CL,0    ;termino delay.total elementos en tabla (0-99)
        MOV CH,8    ;cuenta para shifts
        MOV BH,0    ;poner 00CL en BX
CICLO:MOV BL,CL;pone en BX el indice de la tabla  [2]
        MOV BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
ROTA:SHL BL,1   ;shift en BL para checar si msb es 0 o 1  [2]
        JC ES_UNO ;si carry msb=1, brinca.  [16 o 4]
STRB:MOV DX,37AH;para checar si read strobe esta bajo  [4]
        IN AX,DX   ;lee status  [12]
        AND AX,08H;enmascara read strobe  [4]
        JZ STRB    ;mientras bit 3 sea 0, strobe esta alto  [16/4]
NOSTRB:IN AX,DX;despues que strobe se hace bajo, checar sea  [16/4]
        AND AX,08H;alto de nuevo. enmascara read strobe  [4]
        JNZ NOSTRB;cuando bit 3 sea 0, entonces read strobe =1 [16/4]
        IN AX,DX   ;lee de nuevo para valor de read data  [12]
        AND AX,04H;enmascara bit 2  [4]
        JNZ MAL    ;si read data no es 0 entonces hay error [16/4]
```

```
        JMP SIGUE  ;si read data fue 0 continua  [15]
ES_UNO: IN  AX,DX;lee de nuevo para valor de read data  [12]
        AND AX,04H;enmascara bit 2  [4]
        JZ  MAL    ;si read data no es 1 entonces hay error [16/4]
SIGUE:DEC CH   ;decrementa contador de shift  [3]
        JNZ ROTA;continua shifts si no han terminado  [16/4]
        MOV CH,8H ;reinicializa contador de shifts  [4]
        INC CL     ;incrementa cuenta  [2]
        CMP CL,100 ;checar si ya termino  [4]
        JNZ CICLO  ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP EXITO  ;cuenta=100, termino, brinca a detener cinta [15]
MAL:MOV AL,0D3H;detiene cinta  [4]
        MOV DX,378H; [4]
        OUT DX,AL ; [12]
        MOV AX,00H;para pasar valor de cero a Pascal (0=error) [4]
        JMP FIN    ; [15]
EXITO:MOV AL,0D3H;termino de escribir, ahora detener cinta [4]
```

```
        MOV DX,378H; [4]
        OUT DX,AL ;detiene cinta [12]
        MOV AX,01H;pasa valor de uno a Pascal (1=exito) [4]
FIN:    pop cx          ;termino. hace pop de registros y regresa
        pop dx
        pop bx
        ret                                     ;
leetrk2 endp
code    ends
        end             BEGIN
```

Esta rutina es muy similar a la anterior, con la diferencia de que ahora se efectúa la lectura en el track 2. La definición de la tabla de texto cambia ahora al mencionar el número de track. Tambien cambia el valor para poner en movimiento de lectura a la grabadora de datos DRE-5000, el cual es ahora D1 hexadecimal.

4.3.17 La rutina leetrk3.asm

```
code segment    'code'
        assume cs:code
```

```
        org     100H
BEGIN:JMP       SHORT leetrk3
; ----------DEFINICIONES ------ ----------
TEXTODB         30H,30H,30H,30H,30H;tabla de datos para comparar con los
        DB      30H,30H,30H,30H,30H;datos leidos
        DB      30H,30H,30H,30H,30H;
        DB      31H,2EH,2EH,2EH,2EH;1....
        DB      22H,45H,73H,74H,61H;"Esta       TABLA DE TEXTO
        DB      20H,65H,73H,20H,75H; es u
        DB      6EH,61H,20H,70H,72H;na pr
        DB      75H,65H,62H,61H,20H;ueba
        DB      70H,61H,72H,61H,20H;para
        DB      6CH,61H,20H,67H,72H;la gr
        DB      61H,62H,61H,64H,6FH;abado
        DB      72H,61H,20H,44H,52H;ra DR
        DB      45H,2DH,35H,30H,30H;E-500
        DB      30H,2CH,20H,65H,6EH;0, en
```

```
        DB          20H,65H,6CH,20H,74H; el t
        DB          72H,61H,63H,6BH,20H;rack
        DB          6EH,75H,6DH,65H,72H;numer
        DB          6FH,20H,74H,72H,65H;o tre
        DB          73H,2EH,2EH,2EH,2EH;s"...
        DB          2EH,2EH,2EH,2EH,2EH;.....
; ----------------------------------------------
leetrk3 proc near   ;rutina para leer del track 3 de cinta
        push ax     ;salva registros
        push bx     ;
        push dx     ;
        push cx     ;valor 00E1h pone cinta en track 3, movimiento de
        mov  ax,00E1h;de lectura
        mov  dx,378H ;direccion de puerto en dx
        out  dx,al  ;comienza movimiento de cinta
        mov  ax,25  ;genera retardo de 25 veces 1 milisegundo
DELAY:mov  cx,109H;cuenta para retardo de 1 milisegundo
```

```
DLY1: dec  cx        ;decrementa cuenta
      jnz  DLY1      ;cicla si cuenta no igual a cero
      dec  ax        ;decrementa cuenta de milisegundos
      jnz  DELAY     ;otro milisegundo de retardo si no termino
      MOV  CL,0      ;termino delay.total elementos en tabla (0-99)
      MOV  CH,8      ;cuenta para shifts
      MOV  BH,0      ;poner 00CL en BX
CICLO:MOV  BL,CL;pone en BX el indice de la tabla   [2]
      MOV  BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
ROTA:SHL  BL,1     ;shift en BL para checar si msb es 0 o 1  [2]
      JC   ES_UNO ;si carry msb=1, brinca.   [16 o 4]
STRB:MOV  DX,37AH;para checar si read strobe esta bajo  [4]
      IN   AX,DX   ;lee status  [12]
      AND  AX,08H;enmascara read strobe  [4]
      JZ   STRB    ;mientras bit 3 sea 0, strobe esta alto  [16/4]
NOSTRB:IN  AX,DX;despues que strobe se hace bajo, checar sea  [16/4]
      AND  AX,08H;alto de nuevo. enmascara read strobe  [4]
```

```
        JNZ NOSTRB;cuando bit 3 sea 0, entonces read strobe =1 [16/4]
        IN  AX,DX   ;lee de nuevo para valor de read data  [12]
        AND AX,04H;enmascara bit 2   [4]
        JNZ MAL     ;si read data no es 0 entonces hay error [16/4]
        JMP SIGUE   ;si read data fue 0 continua  [15]
ES_UNO: IN  AX,DX;lee de nuevo para valor de read data   [12]
        AND AX,04H;enmascara bit 2  [4]
        JZ  MAL     ;si read data no es 1 entonces hay error [16/4]
SIGUE:DEC CH    ;decrementa contador de shift  [3]
      JNZ ROTA;continua shifts si no han terminado  [16/4]
        MOV CH,8H ;reinicializa contador de shifts  [4]
        INC CL      ;incrementa cuenta  [2]
        CMP CL,100 ;checar si ya termino  [4]
        JNZ CICLO  ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP EXITO   ;cuenta=100, termino, brinca a detener cinta [15]
MAL:MOV AL,0E3H;detiene cinta  [4]
        MOV DX,378H; [4]
```

```
        OUT DX,AL ; [12]
        MOV AX,00H;para pasar valor de cero a Pascal (0=error) [4]
        JMP FIN     ; [15]
EXITO:MOV AL,0E3H;termino de escribir, ahora detener cinta [4]
        MOV DX,378H; [4]
        OUT DX,AL ;detiene cinta [12]
        MOV AX,01H;pasa valor de uno a Pascal (1=exito) [4]
FIN:  pop cx          ;termino. hace pop de registros y regresa
        pop dx
        pop bx
        ret                                    ;
leetrk3endp
code  ends
        end           BEGIN
```

De nuevo, muy similar a leetrk1.asm, con modificaciones en la definición del texto que ahora menciona que es para el track 3, y el valor para la instrucción para el movimiento de lectura que ahora será E1 hexadecimal.

4.3.18 La rutina leetrk4.asm:

```
code  segment     'code'
      assume cs:code
      org         100H
BEGIN:JMP         SHORT leetrk4
; ---------------DEFINICIONES ------ ---------------
TEXTODB           30H,30H,30H,30H,30H;tabla de datos para comparar con los
      DB          30H,30H,30H,30H,30H;datos leidos
      DB          30H,30H,30H,30H,30H;
      DB          31H,2EH,2EH,2EH,2EH;1....
      DB          22H,45H,73H,74H,61H;"Esta      TABLA DE TEXTO
      DB          20H,65H,73H,20H,75H; es u
      DB          6EH,61H,20H,70H,72H;na pr
      DB          75H,65H,62H,61H,20H;ueba
      DB          70H,61H,72H,61H,20H;para
      DB          6CH,61H,20H,67H,72H;la gr
```

```
        DB      61H,62H,61H,64H,6FH;abado
        DB      72H,61H,20H,44H,52H;ra DR
        DB      45H,2DH,35H,30H,30H;E-500
        DB      30H,2CH,20H,65H,6EH;0, en
        DB      20H,65H,6CH,20H,74H; el t
        DB      72H,61H,63H,6BH,20H;rack
        DB      6EH,75H,6DH,65H,72H;numer
        DB      6FH,20H,63H,75H,61H;o cua
        DB      74H,72H,6FH,2EH,22H;tro."
        DB      2EH,2EH,2EH,2EH,2EH;.....
; ---------------------------------------------------
leetrk4proc near    ;rutina para leer del track 4 de cinta
        push ax     ;salva registros
        push bx     ;
        push dx     ;
        push cx     ;valor 00C1H pone cinta en track 4, movimiento de
        mov  ax,00C1H;de lectura
```

```
        mov  dx,378H ;direccion de puerto en dx
        out  dx,al   ;comienza movimiento de cinta
        mov  ax,25   ;genera retardo de 25 veces 1 milisegundo
DELAY:mov  cx,109H;cuenta para retardo de 1 milisegundo
DLY1: dec  cx      ;decrementa cuenta
        jnz  DLY1    ;cicla si cuenta no igual a cero
        dec  ax      ;decrementa cuenta de milisegundos
        jnz  DELAY   ;otro milisegundo de retardo si no termino
        MOV  CL,0    ;termino delay.total elementos en tabla (0-99)
        MOV  CH,8    ;cuenta para shifts
        MOV  BH,0    ;poner 00CL en BX
CICLO:MOV  BL,CL;pone en BX el indice de la tabla  [2]
        MOV  BL,TEXTO[BX] ;pone en BL valor tabla para indice [12+5=17]
ROTA:SHL  BL,1   ;shift en BL para checar si msb es 0 o 1  [2]
        JC   ES_UNO ;si carry msb=1, brinca.  [16 o 4]
STRB:MOV  DX,37AH;para checar si read strobe esta bajo  [4]
        IN   AX,DX   ;lee status  [12]
```

```
        AND  AX,08H;enmascara read strobe  [4]
        JZ  STRB     ;mientras bit 3 sea 0, strobe esta alto  [16/4]
NOSTRB:IN  AX,DX;despues que strobe se hace bajo, checar sea  [16/4]
        AND  AX,08H;alto de nuevo. enmascara read strobe  [4]
        JNZ  NOSTRB;cuando bit 3 sea 0, entonces read strobe =1  [16/4]
        IN  AX,DX   ;lee de nuevo para valor de read data  [12]
        AND  AX,04H;enmascara bit 2  [4]
        JNZ  MAL     ;si read data no es 0 entonces hay error [16/4]
        JMP  SIGUE  ;si read data fue 0 continua  [15]
ES_UNO: IN  AX,DX;lee de nuevo para valor de read data  [12]
        AND  AX,04H;enmascara bit 2  [4]
        JZ  MAL      ;si read data no es 1 entonces hay error [16/4]
SIGUE:DEC  CH    ;decrementa contador de shift  [3]
      JNZ  ROTA;continua shifts si no han terminado  [16/4]
        MOV  CH,8H ;reinicializa contador de shifts  [4]
        INC  CL      ;incrementa cuenta  [2]
        CMP  CL,100 ;checar si ya termino  [4]
```

```
        JNZ  CICLO   ;lee tabla otra vez si cuenta menor que 100 [16,4]
        JMP  EXITO   ;cuenta=100, termino, brinca a detener cinta [15]
MAL:MOV  AL,0C3H;detiene cinta  [4]
        MOV  DX,378H; [4]
        OUT  DX,AL  ; [12]
        MOV  AX,00H;para pasar valor de cero a Pascal (0=error) [4]
        JMP  FIN     ; [15]
EXITO:MOV  AL,0C3H;termino de escribir, ahora detener cinta [4]
        MOV  DX,378H; [4]
        OUT  DX,AL  ;detiene cinta [12]
        MOV  AX,01H;pasa valor de uno a Pascal (1=exito)  [4]
FIN:  pop cx          ;termino. hace pop de registros y regresa
        pop dx
        pop bx
        ret           ;
leetrk4endp
code  ends
```

```
        end         BEGIN
```

Tambien muy similar a leetrk1.asm, con excepción de los cambios en el texto y en el valor a escribirse a puerto para movimiento de lectura en track 4, el cual es en C1 hexadecimal en este caso.

4.3.19 La rutina leec.asm:

```
code segment    'code'
        assume cs:code
        org 100H
BEGIN:jmp leec
; ---------------------------------------------------
 tempo dw?
; ---------------------------------------------------
leec  proc near
        push bp
        mov bp,sp
        mov         dx,0378H    ; puerto para poner status
        mov         ax,0F1H     ; prepara para lectura
```

```
        out     dx,ax       ; inicia lectura
        mov     cx,00
        mov     dx,00
        mov     ah,01
        int     1AH         ; pone clock en ceros
otro:   mov     ah,00H      ; read time of day, 4 ciclos
        int     1AH         ; 3 ciclos
        cmp     dx,0444H    ; retardo de un minuto
        jnz     otro
        mov     tempo,dx
        mov     dx,0378H    ; para detener cinta antes de salir
        mov     ax,0F2H
        out     dx,al       ; detiene cinta
        mov     ax,tempo
        pop bp
        ret                 ; regresa
leec    endp
```

```
code ends

    end
```

Esta rutina pone el valor F1 hexadecimal en el puerto con dirección 378 hexadecimal del puerto paralelo de la computadora personal IBM o compatible, para poner a la cinta en la grabadora de datos en movimiento de lectura. Despues se genera un retardo de un minuto antes de detener la cinta y regresar. El propósito de esta rutina es el de mantener a la DRE-5000 en movimiento de lectura el tiempo suficiente para que el usuario pueda hacer los ajustes necesarios en la grabadora de datos.

CHAPTER 5 Conclusiones

5.1 Conclusiones sobre el diseño

El diseño del equipo de prueba básicamente se dividio en dos partes principales. Una fue el diseño de la parte física, que consistió en los circuitos electrónicos necesarios para poder interfasar una computadora personal de tipo IBM o compatible a la grabadora de datos DRE-5000. La segunda parte consistió en el desarrollo de los programas que a través de los circuitos mencionados permitan y hagan posible el control y manejo de la grabadora de datos DRE-5000, para poder hacer las pruebas necesarias para asegurar su correcto funcionamiento. El diseño de los circuitos se simplificó grandemente debido a la utilización de la capacidad de procesamiento de la computadora personal. Al mismo tiempo, la utilización de la computadora da gran flexibilidad al equipo de prueba, ya que si llegará a ser

necesario hacer cambios en este, bastará con modificar llos programas, y dado que el diseño de estos se hizo en forma modular, lo más probable es que tuviera que modificarse únicamente alguna de las rutinas de lenguaje ensamblador. En cuanto a los lenguajes utilizados, la combinación de un lenguaje de alto nivel y uno de bajo nivel dió buenos resultados, ya que el lenguaje de alto nivel facilito la creación de la estructura principal y de la interfase interactiva hacia el usuario (por ejemplo, en la elaboración de menús). El lenguaje de bajo nivel permitió a su vez el trabajar más a detalle con los circuitos y con un mayor control donde fue requerido.

La reducción en complejidad de los circuitos permitió tambien una disminución en el tamaño físico del equipo, y un ahorro considerable en cuanto a costo. Dada la complejidad de las funciones que el equipo de prueba desempeña, de no utilizarse la computadora personal se requeriría de una gran cantidad de circuitos que desempeñarán dichas funciones. El utilizar el puerto serie de la computadora personal tambien implicó ventajas, ya que se evitó el tener que desarrollar una tarjeta de interfase para la computadora personal. Tambien esto ayuda en el manejo, ya que es mucho más fácil conecta un cable al puerto paralelo de la computadora que abrirla para instalar una tarjeta de interfase.

Dentro de la familia de las computadoras IBM y compatibles existen diferentes tipos de procesadores tales como el 8088, 8086, 80286, 80386 y 80486, pero tanto el lenguaje ensamblador como Pascal pueden ser utilizados en cualquiera de ellos. Este diseño fue elaborado considerando únicamente al 8088 , pero si se deseará utilizar alguno de los otros procesadores, esto sería posible y sólo necesitarían hacerse pequeñas modificaciones en el código de algunas de las rutinas de lenguaje ensamblador que implican tiempo. Las modificaciones consistirían básicamente en

añadir instrucciones para forzar el tener más ciclos de máquina para que se cumplan los tiempos requeridos. Esto debido a que los procesadores 80X86 son más rápidos que el 8088. Lo anterior aplica tambien aún para el procesador 8088 si se utilizan velocidades de reloj mayores que 4.77 MHz (reloj original de la IBM PC). Por ejemplo, hay algunos sistemas llamados "turbo" que funcionan a velocidades de 5 u 8 MHz, y algunos hasta 12 MHZ (8088).

Los circuitos del equipo de prueba por otra parte, será muy improbable que necesiten ser modificados, ya que estos sólo proveen la interfase física entre el puerto paralelo de la grabadora de datos y la computadora personal IBM o compatible. Este puerto es estandar, y tiene las mismas funciones, y la misma configuración de pins en todos los modelos de computadoras, sin importar el tipo de procesador (microprocesador) que estas utilizen. Si se observan dichos circuitos, se podrá observar que en realidad no son simplemente una interfase para la grabadora de datos DRE-5000, es decir no están limitados a esta función, sino que en realidad proveen una interfase que puedes ser aprovechada con otros propósitos. Más adelante se hablará sobre esto, en la sección de usos y aplicaciones, donde se discutirá sobre posibilidades de uso del equipo de prueba en otras aplicaciones.

Uno de las situaciones más dificiles durante el diseño del equipo de prueba fue el lograr ajustar el tiempo requerido para poder simular la duración del pulso de WRITE STROBE que requiere la grabadora de datos DRE-5000. Para poder cumplir este requerimiento trabajando la grabadora a una velocidad de 63 bits/mm se necesitaba un total de 25 ciclos de máquina, lo que implicaba el uso de un reducico número de instrucciones de lenguaje ensamblador, lo que no permitía

realizar todas las funciones que se deseaba (por ejemplo, en las rutinas wrtrk1, wrtrk2, wrtrk3 y wrtrk4) ya que además de producir el pulso, se requería al mismo tiempo preparar los datos que serían escritos a cinta. Las especificaciones de la grabadora de datos (mostradas en la tabla 1) muestran que la grabadora puede utilizar 2 velocidades diferentes para grabación. La primera es de 63 bits/mm, que era la utilizada inicialmente, y la segunda de 31.5 bits/mm. Estudiando la información de la grabadora de datos DRE-5000 se encontró que la única diferencia entre grabar a una velocidad u otra, consiste únicamente en la duración del pulso de write strobe, el cual dura 10.42 uS en la posición alta para una velocidad de 63 bits/mm (1600 bits/in), mientras que para una velocidad de 31.5 bits/mm (800 bits/in) este pulso alto tiene una duración del doble de tiempo, o sea 20.83 uS. Ya que se vió que ninguna otra característica o función se altera, se decidió hacer las pruebas de grabación en cada uno de los 4 tracks a la velocidad de 31.5 bits/mm. Esto permitió poder utilizar todas las instrucciones deseadas, entre las que se incluye la utilización de una tabla con valores hexadecimales correspondientes a un texto diferente para cada track, que puede asi ser grabado y leido track por track.

La limitación mencionada anteriormente no existiría en el caso de utilizarse procesadores más rápidos, como los 80X86, por lo que si llega a modificarse el programa para utilizarse con una computadora que utilice alguno de estos procesadores, podría tambien reducirse a la mitad la duración del pulso de WRITE STROBE durante las pruebas de escritura y lectura, para que estas fueran efectuadas a 63 bits/mm (1600 bits/in). Sin embargo, esto no es crítico, ya que como se mencionó anteriormente, no se afecta confiabilidad de las pruebas al

trabajar a la velocidad de 31.5 bits/mm (800 bits/in) y además, esta velocidad esta incluida en las especificaciones de la grabadora de datos DRE-5000.

5.2 Usos y Aplicaciones.

5.2.1 Usos.

Entre otros usos que podrían darse al equipo de prueba tendríamos el caso de la utilización de una grabadora de datos para hacer respaldo de los archivos en el disco duro de la computadora personal IBM o compatibles. Los mismos circuitos del equipo de prueba podrían ser utilizados sin necesidad de modificación alguna. En cuanto a los programas, podría conservarse la estructura, y cambiar por ejemplo el texto en los menús, y cambiar las rutinas de lenguaje ensamblador para que en vez de realizar pruebas a la grabadora de datos se realizara el respaldo o restauración de los archivos desde o hacia el disco duro de la computadora personal. Por ejemplo, las rutinas para escritura podrían modificarse para que en vez de leer información de una tabla (como en las pruebas), leyeran la información de disco y la escribieran a cada track de la grabadora de datos en secuencia. Podría tambien estudiarse la posibilidad de utilizar alguna otra técnica para hacer más eficiente la transferencia de datos disco/cinta. Por ejemplo, podría estudiarse la posibilidad de utilizar DMA (Direct Memory Access). Tambien podría considerarse la opción de añadir un circuito que generara los pulsos de WRITE STROBE. Esto haría tambien más eficiente la transferencia de los datos de disco a cinta y viceversa.

El uso de una grabadora de datos para respaldo de disco de la computadora personal podría no limitarse a utilizar el equipo de prueba sólo con la grabadora de

datos DRE-5000. Existen otras marcas de grabadoras de datos que podrían ser utilizadas. Algunas sin cambios, otras con mínimos cambios en la configuración de la interfase. Incluso hay algunas otras grabadoras de datos que requieren un menor número de señales en la interfase (como es el caso de algunas grabadoras Tandberg).

Tambien pudiera ser utilizado el equipo de prueba para probar otras grabadoras que no sean DRE-5000. Posiblemente con mínimas modificaciones en configuración de pins en algunos casos, o bien tal vez hubiera necesidad de cambiar conectores en algunos otros casos. Habría tambien tal vez necesidad de hacer modificaciones en las rutinas de ensamblador. Por ejemplo, alguna otra grabadora de datos podría tener diferentes requerimientos para los pulsos de WRITE STROBE, en este caso necesitarían modificarse tanto las rutinas de lenguaje ensamblador para escritura como las rutinas para lectura, ya que diferencias en los pulsos de WRITE STROBE generarían diferencias tambien en los pulsos de READ STROBE.

Todos los cambios mencionados anteriormente serían mínimos, ya que no se efectuarían cambios drásticos ni en la estructura del programa, ni en los circuitos, y la mayoría de las rutinas de lenguaje ensamblador podrían probablemente seguir siendo utilizadas sin modificaciones.

5.2.2 Aplicaciones

En la sección anterior se mencionó que los circuitos del probador para la grabadorade datos DRE-5000 no están limitados a utilizarse únicamente como interfase entre la computadora personal IBM o compatibles y la grabadora de datos. Como podrá observarse, básicamente estos circuitos proveen interfase para

8 señales de entrada y 8 de salida provenientes de la computadora personal. Los circuitos utilizados son compuertas TTL, y siendo este un estandar ampliamente utilizado, ofrece la ventaja de que estos circuitos podrán ser utilizados prácticamente sin modificación para un sinnúmero de aplicaciones. Por ejemplo, podría construirse un sistema de alarmas contra robo para ser utilizado en el hogar. Construyendo un equipo sencillo que incluyera sensores para puertas y ventanas, podría conectarse este a los circuitos ya existentes del equipo de prueba, y se desarrollarían programas que se encarguen de activa, desactivar y monitorear los sensores. Además, podrían utilizarse otros circuitos que permitieran al ser manejados por la computadora personal a través de los circuitos del equipo de prueba el encendido y apagado de luces a horas determinadas. Podría incluso conectarse un módem a la computadora personal, para que esta hiciera llamadas a la policía, vigilancia, los vecinos, etc. en caso de notar al monitorear los sensores en puertas y ventanas que alguien ha entrado en la casa. Existen en el mercado sensores y circuitos relativamente baratos, que pueden adquirirse fácilmente, y que podrían ser utilizados en la aplicación mencionada anteriormente.

Lo mencionado anteriormente es sólo un ejemplo. Existe un sin número de aplicaciones en que podrían utilizarse los circuitos del equipo de prueba en conjunto con una computadora personal. Dado que es cada día mayor el número de personas que cuenta con una computadora en casa y que en la mayoría de las ocasiones estas computadoras son subutilizadas, el desarrollar aplicaciones caseras podría tener un amplio éxito. Sobre todo considerando la simpleza y el bajo costo de los circuitos del equipo de prueba. Ya que estos circuitos fueron diseñados para utilizarse con el puerto paralelo del equipo de prueba, prácticamente pueden ser

utilizados con cualquier computadora, IBM compatible o no, ya que el puerto paralelo es etandar, por lo que el campo de acción y de aplicaciones se expande aún más si no las limitamos únicamente a este tipo de computadoras. No solamente los circuitos, tambien los programas podrían ser utilizados en otras aplicaciones, con mínimas modificaciones. Por ejemplo, bastaría cambiar el texto de los menús para seguir utilizando estos y podría conservarse la estructura del programa principal. Las rutinas de lenguaje ensamblador podrían tambien ser modificadas y utilizadas, en caso de requerirse, aunque en las aplicaciones que no requirieran tener un control exacto sobre el tiempo (orden de microsegundos), sería posible reemplazar estas por procedimientos escritos en Pascal.

BIBLIOGRAFIA

Interfacing to the IBM Personal Computer
Lewis C. Eggebrecht
First Edition
SAMS

The TTL Data Book for Design Engineers
Second Edition
Texas Instruments

DRE-5000
Service Manual
DRE

IBM PC
Assembler Language and Programming
Peter Abel
Prentice Hall

Programming in Pascal
Peter Grogono
Second Edition
Addison-Wesley

Advanced Turbo Pascal; Programming and Techniques
Peter Abel
Prentice Hall

8086/8088 Assembly Language Programming
Leo J.Scanlon
Robert J.Brady Co.