DEDICATORIA

A mis padres quienes sembraron las bases morales y los conceptos de amor necesarios para construir una vida.

A la memoria del hombre que entregó su vida al servicio de su hogar y su familia.

A mi madre, la mujer abnegada que supo guiar a su familia por el camino de la verdad y honestidad.

A Beatriz, la persona especial que me apoyó todo el tiempo y le ha dado a mi vida un valor diferente.

AGRADECIMIENTO

Un agradecimiento sincero al Ingeniero Fernando Flores, cuya ayuda y conocimientos fueron fundamentales para la realización de este trabajo.

A todos quienes hacen la Facultad de Ingeniería Eléctrica, que sembraron en mi vida bases profundas de dedicación y sacrificio.

Certifico que el presente trabajo ha sido elaborado en su totalidad por el Sr. Francisco Córdova Chehab

Ing. Fernando Flores
Director

INDICE

CAPITULO I.- GENERALIDADES 1

1.1.- Introducción 1

1.1.1.- Importancia 2

1.1.2.- Contenido. 3

1.2.- Descripción General del Sistema. 4

1.3.- Requerimientos del Sistema. 10

1.4.- Flexibilidad. 11

CAPITULO II.- Descripción del Sistema 15

2.1.- Descripción de los Microcontroladores CY-545A 15

2.1.1.- Introducción 15

2.1.1.1.- Funciones de Control de Movimiento. 15

2.1.1.2.- Señales de Movimiento. 16

2.1.1.3.- Funciones Generales y Características Especiales. 16

2.1.2.- Descripción de Pines. 17

2.1.3.- Comandos para Interfases. 22

2.1.3.1.- Interfase Paralelo. 22

2.1.3.2.- Interfase del CY545 al CY233. 23

2.1.3.3.- El Interfase Serial. 25

2.1.4.- Sumario de Comandos. 27

2.1.4.1.- Formato de los Comandos 27

2.1.4.2.- Formato de Comandos ASCII. 27

2.1.4.3.- Formatos de comando binario. 28

2.1.4.4.- Comandos de control del motor. 29

2.1.4.4.1.- Velocidad de los pasos. 29

2.1.4.4.2.- Comandos de dirección de movimientos. 30

2.1.4.4.3.- Comandos para el modo de pasos. 31

2.1.4.4.4.- Comandos para el movimiento del motor. 32

2.1.4.5.- Comandos para Manejo de Bits. 32

2.1.4.5.1.- User Bits. 32

2.1.4.5.2.- Saltos y Esperas dependientes de Bits. 33

2.1.4.5.3.- Comando HOME. 34

2.1.4.6.- Comandos para soporte de memoria externa. 35

2.1.4.6.1.- Puntero de dirección de memoria externa. 35

2.1.4.6.2.- Comandos de escritura en memoria. 35

2.1.4.6.3.- Comandos para ejecutar programas desde memoria externa 36

2.1.4.7.- Comandos para saltos. 37

2.1.4.8.- Comandos de modo de operación. 38
2.1.4.9.- Comandos Generales. 40
2.1.4.9.1.- Comando de inicialización. 40
2.1.4.9.2.- Comandos de retraso de tiempo. 41
2.1.4.9.3.- Comando de solicitud. 41
2.1.4.9.3.1.- Comandos de Display de Status. 41
2.1.4.9.3.2.- Obtención del contenido de la memoria. 42
2.1.4.9.3.3.- Comandos de display para mensajes. 43
2.2.- FORMATOS DE SALIDA DEL PROGRAMA ORCAD. 44
2.2.1.- Características del programa ORCAD/SDT III 44
2.2.1.1.- DRAFT. 44
2.2.1.2.- Librería de Elementos. 45
2.2.1.3.- Creación de elementos para la Librería. 45
2.2.1.4.- Programas Utilitarios. 45
2.2.2.- Características del Programa ORCAD PCB. 46
2.2.3.- Formato de Salida del Programa ORCAD PCB. 46
2.3.- Formatos de Salida del programa TANGO. 48
2.3.1.- TANGO PCB. 49
2.3.1.1.- Formatos de Salida del Programa TANGO PCB. 49
2.3.2.- Programa TANGO ROUTE. 51
2.4.- Formato de Programa diseñado. 52

CAPITULO III.- DISEÑO DEL SISTEMA 55
3.1.- Diseño del Sistema Mecánico. 55
3.1.1.- Diseño de los ejes X y Y. 67
3.1.2.- Diseño del sistema del Eje Z. 66
3.1.3.- Calibración de los sistemas. 67
3.1.4.- Montaje del Sistema Completo. 68
3.1.5.- Panel de Control. 69
3.2.- Diseño del Sistema Electrónico de Control de los motores de pasos y el taladro 72
3.2.1.- Parámetros del CY545. 73
3.2.2.- Circuitos de Reloj y Reset para el CY545. 74
3.2.3.- Interfase con la Memoria. 76
3.2.4.- Diseño del interfase para el Display HP. 79
3.2.5.- Utilización de los User Bits. 82
3.2.5.1.- User Bits para el EJE X y Z. 82
3.2.5.2.- User Bits para el EJE Y. 84
3.2.6.- Diseño del Interfase de control de los motores de paso. 84
3.2.6.1.- Señales que controlan el movimiento de los motores de paso 85
3.2.6.1.1- Señales PULSE y CCW. 85

3.2.6.1.2.- Señal STOPPED. 87
3.2.6.1.3.- Señal SLEW. 88
3.2.6.1.4.- Señal INHIBIT_ABORT. 88
3.2.6.2.- Diseño del Traductor. 91
3.2.6.3.- Motores de Pasos. 95
3.2.6.4.- Diseño del Circuito de Potencia para los motores de paso 96
3.2.6.5.- Operación en el modo JOG. 100
3.2.7.- Diseño del Circuito de Control del Taladro. 101
3.3.- Diseño de Seguridades y Protecciones. 103
3.4.- Diseño del Interfase con el Microcomputador. 104
3.4.1.- Controlador Inteligente de Redes Locales CY-233. 104
3.4.1.1.- Diagrama Lógico. 107
3.4.1.2.-Configuración de Pines del CY-233. 108
3.4.1.3.- Descripción de las señales. 110
3.4.1.4.- Modos de Comunicación. 130
3.4.1.5.- Juegos de Comandos del CY233. 132
3.4.1.5.1.- Comandos Seriales de RED. 134
3.4.1.5.2.- Comandos Paralelos. 141
3.4.1.5.3.- Comando Seriales de LAN. 144
3.4.1.6.- Especificaciones Eléctricas. 148
3.4.1.6.1.- Rangos máximos absolutos. 148
3.4.1.6.2.- Características de operación y DC. 149
3.4.2.- Red de Anillo. 149
3.4.3.- Circuito de Reset. 151
3.4.4.- Circuito de Reloj. 151
3.4.5.- Configuración del CY233. 152
3.4.6.- Conexión al computador. 155
3.4.7.- Circuitos completos. 156
3.5.- Fuentes de Poder. 159
3.5.1.- Fuente de Poder de 12 Voltios. 159
3.5.2.- Fuente de Poder de +5 Voltios. 159

CAPITULO IV.- DESARROLLO DEL SOFTWARE 162
4.1.- Diagrama de Flujo General. 162
4.1.1.- Diagrama de Flujo del Proceso de Perforación Automático. 164
4.2.- Comandos Generales del Microcontrolador CY-545A. 173
4.3.- Programa Principal 174
4.3.1.- Menú Inicial 175
4.3.2.- Modo Automático 176
4.3.2.1.- Calibración del Taladro. 177

4.3.2.2.- Porcentaje Perforado 178
4.3.2.3.- Eje X y Eje Y 178
4.3.2.4.- Parámetros 178
4.3.2.5.- Información, Nombre del Circuito y Archivo Temporal 179
4.3.2.6.- Interrupción 179
4.3.2.7.- Control de Inicio y Menú Inicial 180
4.4.- Programas Auxiliares y Subrutinas. 181
4.4.1.- Modo Manual 181
4.4.1.1.- Parámetros del CY545 182
4.4.1.2.- Display HP 183
4.4.1.3.- Control Manual 184
4.4.1.4.- Programa de CONTROL 184
4.4.1.5.- Control Manual del Taladro. 186
4.4.1.6.- Go Específico 188
4.4.1.7.- Menú Inicial 188
4.4.2.- Manejo de Archivos 188
4.4.2.1.- Tipo de Archivo 189
4.4.2.2.- Ordenamiento X y Ordenamiento Y. 190
4.4.2.3.- Posición 191
4.4.2.4.- Listar Puntos. 191
4.4.2.5.- Conversión 192
4.4.2.6.- Menú Inicial 193
4.4.3.- Reinicio 193
4.4.4.- Subrutinas. 195
4.4.4.1.- Programas y Declaraciones Globales 195
4.4.4.2.- Subrutinas de ARCHIVOS.FRM 198
4.4.4.2.1.- Declaraciones 198
4.4.4.3.- Subrutinas de AUTO.FRM 204
4.4.4.3.1.- Declaraciones 204
4.4.4.3.2.- Subrutinas 205
4.4.4.4.- Subrutinas de INICIAL.FRM 212
4.4.4.5.- Subrutinas de MANUAL.FRM 214
4.4.4.6.- Subrutinas de REINICIO.FRM 226
4.4.4.6.1.- Declaraciones 226
4.4.4.6.2.- Subrutinas 226

Capítulo V.- Resultados Experimentales 230
5.1.- Comprobación de Control de Escalas. 230
5.2.- Comprobación del grado de precisión. 236
5.3.- Ejemplo de Diseño, Construcción y Perforación de un circuito impreso 238

Capítulo VI.- Conclusiones y Recomendaciones **251**
6.1.- Análisis de los resultados experimentales 251
6.2.- Análisis técnico-económico del equipo construido. 252
6.3.- Conclusiones Generales 257
6.4.- Recomendaciones. 259

Bibliografía **261**
Anexos

Capítulo I

Generalidades

CAPITULO I: GENERALIDADES

1.1.- Introducción.

Por muchos años, se han estado desarrollando sistemas para la fabricación de máquinas que podrán convertir el mundo del diseño CAD en algo real. Con el desarrollo de los motores de pasos y de sus controladores, se han logrado metas de diseño importantes en este campo; así, los diseños basados en computadores personales han sido modelados por máquinas que, apoyadas por programas que utilizan los puntos vectorizados de los programas de CAD, son capaces de convertir estas figuras computarizadas en prototipos reales, trabajando en materiales escogidos tales como cera, metales ligeros, madera, etc.

Se ha pretendido construir máquinas que usando programas de diseño CAD, o de circuitos impresos, permitan crear nuevas partes mecánicas, o realizar procesos automatizados de trabajo sobre partes electrónicas específicas, basados en información proporcionada por los computadores.

El diseño, fabricación y construcción de partes, basados en computadores, pretenderá ser la tecnología de punta del Siglo 21, y se le ha dado el nombre de Robotización.

En la actualidad, se han diseñado todos los elementos necesarios para crear este tipo de máquinas, las que estarán basadas fundamentalmente en motores de pasos, y su capacidad para rotar ángulos específicos. Además, los controladores diseñados para hacer más sencillo el control de estos motores, permiten asegurar altos grados de precisión en el trabajo, siendo este factor, junto con el de la automatización, los que han tratado de solucionar este tipo de necesidades. Hoy, es imposible negar la necesidad de automatizar ciertos procesos, que en algunos de los casos, entrañan peligro para el ser humano, y en otros no pueden ser realizados por el hombre.

La importancia de disponer de este tipo de sistemas siguen aumentando en la actualidad, y se va haciendo más necesario cada día el control numérico de procesos .

Basado en estas necesidades, y con las posibilidades actuales, se decidió diseñar y construir un equipo que permita, en base a control numérico, perforar los agujeros correspondientes a un circuito impreso diseñado por computador. En el proceso de diseño se trató de fabricar un sistema abierto, que permita no solo perforar circuitos impresos, sino diseñar herramientas mecánicas. Además, el equipo diseñado debía recibir su información directamente desde el computador, y luego de ello, tener la capacidad de trabajar en forma independiente.

El equipo diseñado utiliza información provista por programas computacionales de diseño de

circuitos impresos. Esta información se analiza y procesa, convirtiéndola en acciones específicas, como son el movimiento a un punto determinado y la perforación de un agujero. Puesto que existen diferentes puntos, el proceso deberá ser repetitivo.

1.1.1.- Importancia

En la actualidad, existen muchos procesos de fabricación que se realizan manualmente, con un grado de precisión bajo, e inversión alta de tiempo.

La necesidad de crear piezas con mayores grados de precisión, con costos económicos bajos y mayor seguridad, ha llevado a los investigadores a diseñar sistemas y componentes que podrían automatizar una gran cantidad de procesos.

En base a estos nuevos diseños, los computadores han comenzado a tomar el control de procesos que antes debían ser realizados por una gran cantidad de personas. Gracias a la automatización, ahora llamada robotización, se han implementado plantas de fabricación de productos en serie, en las que el Gerente es una computadora y los empleados son motores de pasos. Principalmente, se ha reemplazado a la mano de obra humana por robots, en procesos riesgosos de trabajo, que ponían en peligro la vida de los trabajadores. Además, se han enviado a estos modernos elementos electrónicos a realizar tareas que el hombre nunca podría realizar, por falta de elementos vitales de supervivencia; así, han sido equipos, basados en motores de paso, los que han colocado satélites en el espacio, o tomado muestras en otros planetas.

El objetivo de estas investigaciones y nuevos diseños no es reemplazar al hombre, puesto que él seguirá siendo base fundamental para el desarrollo en nuestro planeta. El objetivo es automatizar procesos que requieren de mayor precisión y menor tiempo de ejecución.

Por ello, se ha pretendido diseñar una máquina que permita, en forma experimental, convertir diseños computacionales en prototipos reales, sin la necesidad de utilizar herramientas de trabajo pesado.

1.1.2.- Contenido.

Esta tesis se ha dividido en seis capítulos, cada uno de los cuales se describirá brevemente a continuación.

El Capítulo I trata sobre GENERALIDADES. En él, se especifica el área de trabajo y se justifican los motivos para el diseño. Se describe en forma global el sistema que se ha desarrollado y se establecen sus límites de trabajo.

En el Segundo Capítulo, se describe detalladamente a los microcontroladores de motores de paso CY-545, base fundamental de este diseño. Además, se describen los formatos ASCII de salida de los dos programas utilizados para el diseño de los circuitos impresos: ORCAD y TANGO. Adicionalmente, se describe el formato ASCII propio de trabajo utilizado por este sistema.

En el Tercer Capítulo, se describe detalladamente cada una de las partes mecánicas y electrónicas que forman parte del diseño; esto es, se analiza el diseño mecánico, el sistema de control de los motores de paso y del taladro, las seguridades y protecciones, el interfase con el computador y el diseño de las fuentes de poder.

El Cuarto Capítulo describe todo el proceso de desarrollo del programa utilizado para manejar el equipo.

En el Quinto Capítulo se han comprobado las escalas y el grado de precisión del equipo construido. Además, se describe un proceso completo de construcción de un circuito impreso.

En el Sexto capítulo, se han analizado los resultados experimentales logrados con el equipo diseñado, y se ha realizado un análisis técnico-económico del equipo construido. Además, se incluyen conclusiones y recomendaciones generales.

Por último, se han incluido la Bibliografía y Anexos suplementarios con las especificaciones técnicas de los elementos electrónicos utilizados en el diseño.

1.2.- Descripción General del Sistema.

Basado en los objetivos planteados, se desarrolló un sistema que permita perforar automáticamente un circuito impreso. Pero, se enfocó el diseño a un proceso completo de fabricación, que no solamente se basaba en la construcción de un equipo mecánico que permita mover un taladro a una posición específica, sino que utiliza diferentes recursos de diseño, con el objetivo de automatizar completamente, basado en computadores personales, el diseño y la construcción de circuitos impresos.

El proceso básico de construcción de un circuito impreso se inicia con el diseño esquemático del mismo. Utilizando programas como ORCAD SDT, se pueden generar diseños esquemáticos que incluyen todas las partes que se han utilizado y sus colecciones lógicas.

El programa utilizado para el diseño esquemático, ORCAD SDT, permite obtener información relacionada con los componentes que forman parte del diseño y las colecciones entre ellos, pero no de la ubicación, orientación y forma físicas de los componentes en el circuito impreso. Para ello, es necesario

utilizar programas para diseñar la localización física de los componentes en el circuito, y el enrutamiento de las colecciones.

Por lo tanto, es necesario disponer de las colecciones lógicas entre cada uno de los componentes del circuito. Esta información se obtuvo, en este caso, utilizando el programa NETLIST de ORCAD SDT.

Se han utilizado dos programas para diseñar físicamente el circuito impreso y enrutar las colecciones. Estos programas son ORCAD PCB y TANGO PCB. En base a ellos, se han diseñado los circuitos impresos, y se han enrutado las colecciones, automática (TANGO ROUTE) o manualmente. Para enrutar las colecciones automáticamente, estos programas necesitan de la información sobre las colecciones lógicas entre cada uno de los componentes, para lo que se utilizó NETLIST. Se debe utilizar la opción adecuada de este programa, dependiendo del formato que se vaya a utilizar, ORCAD o TANGO.

Basándose en las características de los programas PCB, se diseñan circuitos impresos con caminos óptimos, los que luego se imprimen. En base a estas impresiones, se construyen los circuitos sobre la baquelita, utilizando diferente métodos. Uno de los más modernos se basa en impresiones del circuito sobre papel plástico especial, utilizando impresoras láser. Este papel impreso, al ser adherido por medio de calor a la baquelita, y luego retirado, sensibiliza el cobre, que al ser tratado con la solución respectiva, no se disolverá. Solo se disolverán las partes no sensibilizadas, que equivalen a los espacios en blanco de la impresión.

Una vez construido el circuito impreso, se deberá perforar los agujeros correspondientes. Para ello, se utilizará el equipo construido. Primero, se deberán utilizar los programas PCB para obtener un archivo ASCII con la información de los agujeros en el sistema. Este archivo ASCII deberá ser convertido, utilizando un programa desarrollado bajo WINDOWS, a un formato ASCII propio. Luego, deberá colocarse la baquelita y calibrarse el origen de coordenadas del sistema. Utilizando el programa desarrollado y el archivo convertido, el computador comandará al equipo para que se perforen todos y cada uno de los puntos en la baquelita. En este punto, se ha terminado el diseño del circuito impreso.

Analizado este proceso, el equipo construido debía permitir perforar los circuitos impresos, basado en la información proporcionada por los programas PCB. Por lo tanto, debieron desarrollarse varias partes dentro de este proceso. Primero, un programa que permita comandar, a través del computador, el equipo construido. Segundo, la circuitería electrónica que controlará los motores de pasos. Y, tercero, la parte mecánica que comandada por los motores de pasos, moverá el taladro a la posición física determinada, y perforará el agujero.

El programa desarrollado permite realizar diferentes funciones, entre ellas: manejar manualmente el equipo electrónico y mecánico, perforar automáticamente un circuito impreso, calibrar el taladro, y convertir los archivos. Este programa maneja directamente el interfase entre el computador y el equipo,

y comanda a los microcontroladores CY545 para que muevan los motores de pasos en secuencias específicas y a coordenadas determinadas. El programa se desarrolló en VISUAL BASIC for WINDOWS, lo que significa que el interfase entre el usuario y el computador es visual y basado en Objetos Orientados, lo que constituye una gran ventaja. De esta manera se aprovechará las opciones multitasking disponibles bajo Windows, siendo posible trabajar en otro tipo de programas, mientras el equipo perfora la baquelita.

La parte electrónica de este diseño se basa fundamentalmente en dos Microcontroladores, el Microcontrolador de Motores de pasos CY545 que permite realizar diferentes acciones sobre los motores de pasos, y el Microcontrolador de comunicaciones CY233, que permite manejar la información entre el computador y los diferentes microcontroladores CY545. La función principal de esta parte electrónica será convertir la información recibida del computador, mediante comunicación serial, a pulsos que serán enviados a los motores de pasos. Además, a través de esta circuitería, se podrán mover manualmente los motores, y por lo tanto se producirán desplazamientos físicos de las diferentes partes del sistema. Para alimentar a los circuitos integrados que forman parte de este sistema, se han utilizado dos fuentes de poder, una de 5 voltios y otra de 12 voltios. La fuente de 5 voltios alimentará a todos los circuitos integrados relacionados con los microcontroladores, y la fuente de 12 voltios, suministrará energía a los motores de paso.

La tercera parte de este sistema, la parte mecánica, permite mover físicamente el taladro en diferentes direcciones. Puesto que cualquier objeto puede ser diseñado en base a tres coordenadas, X, Y y Z, el aparato diseñado debe permitir trabajar sobre estos tres ejes, moviéndose a coordenadas específicas y ejecutando diferentes acciones en cada uno de estos puntos. Esto quiere decir que se necesitarán motores de paso para mover el equipo sobre cada eje de coordenadas. Los ejes X y Y moverán al eje Z a la coordenada específica del punto a perforarse. El eje Z bajará el taladro, se perforará el agujero, y subirá el taladro. Para cada uno de los ejes se han diseñado partes mecánicas similares, las mismas que se acoplarán a los motores de paso.

En las siguiente figuras, se puede observar el equipo construido y sus diferentes partes internas:

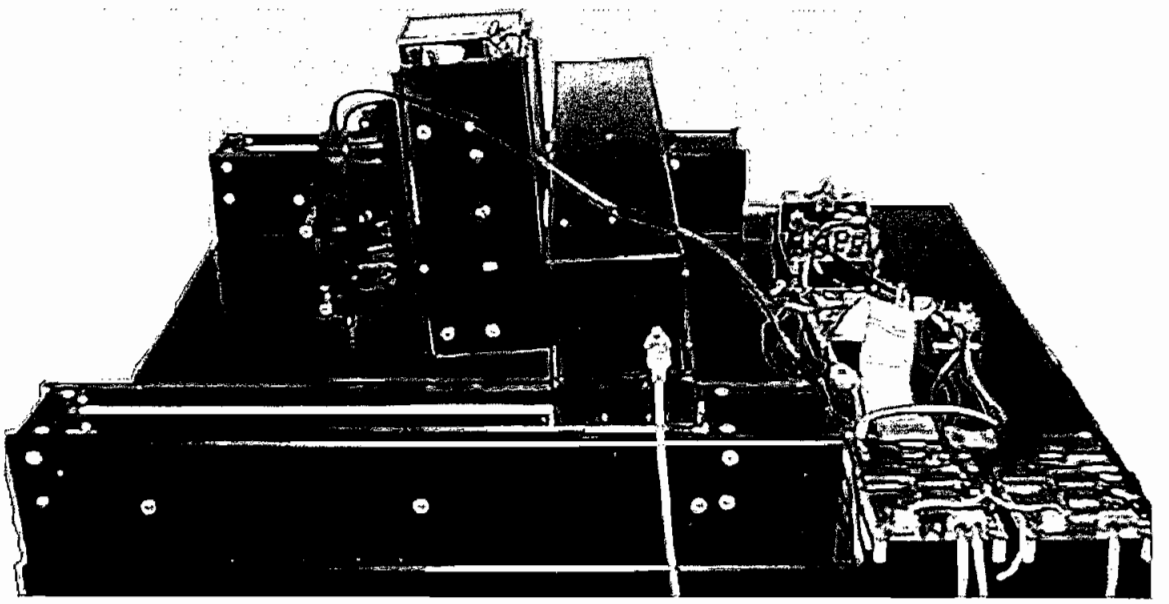


Figura 1.1

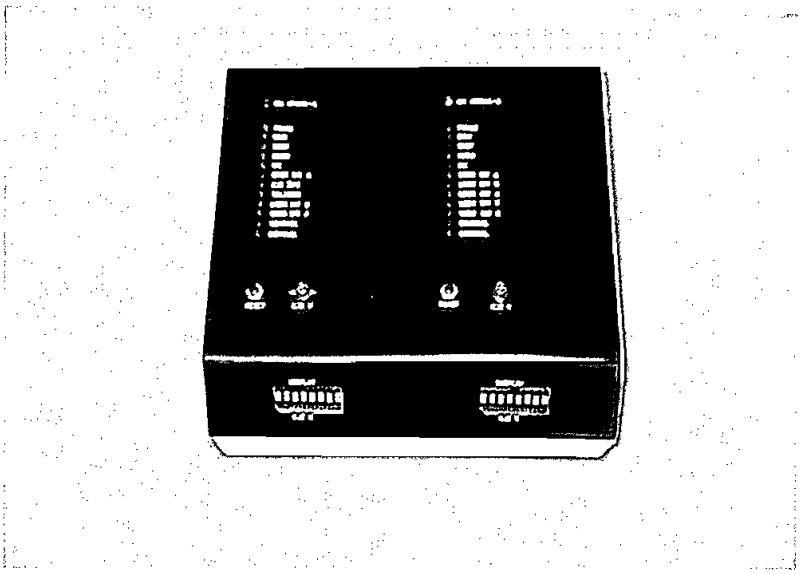


Figura 1.2a

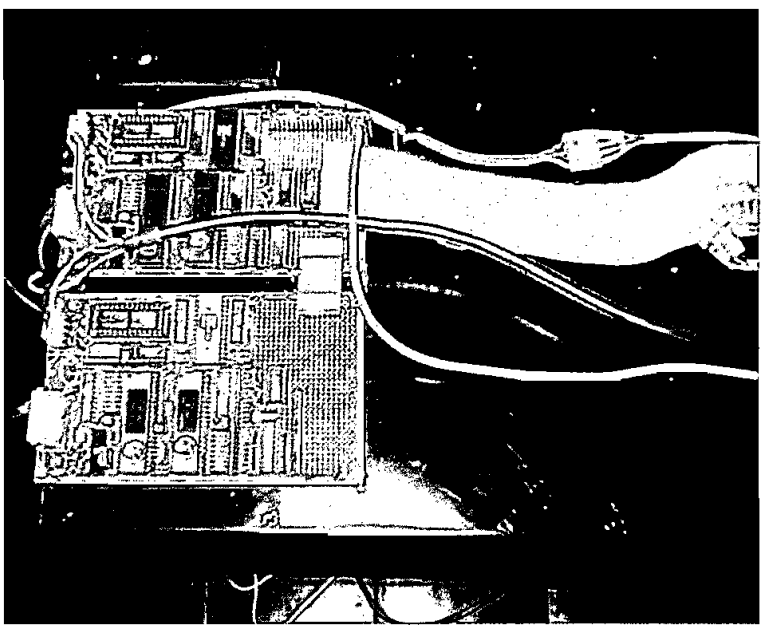


Figura 1.2 b

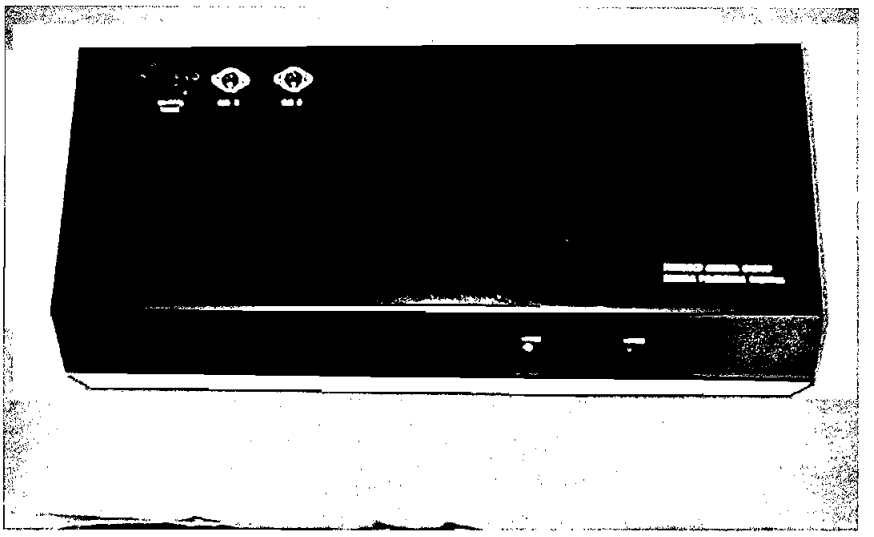


Figura 1.3a

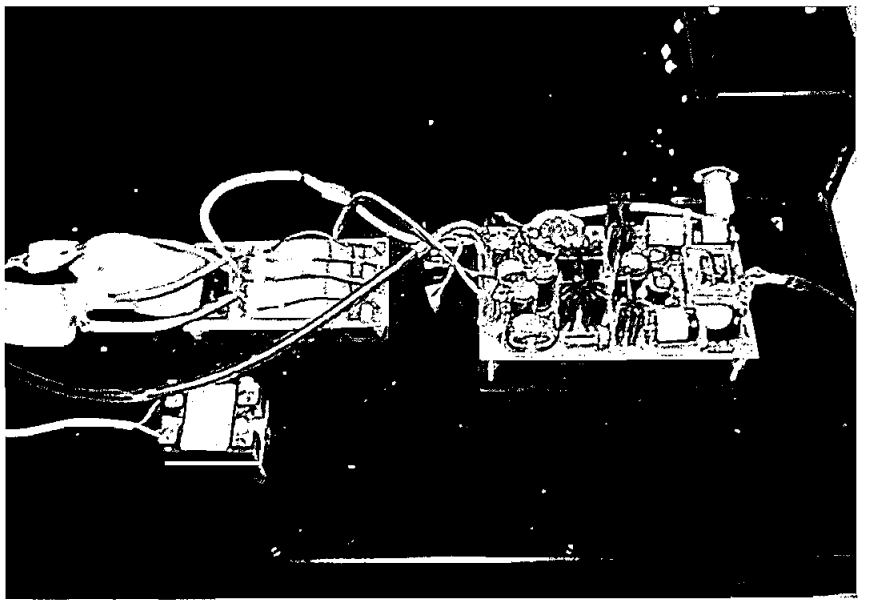


Figura 1.3b

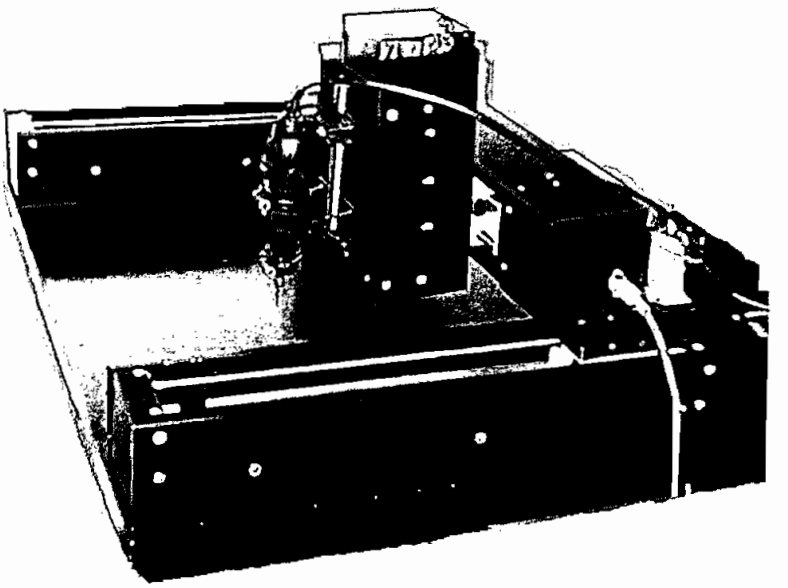


Figura 1.4

Fig. 1.4

En la figura 1.1 se puede observar el equipo completo construido con el panel de control y la parte mecánica diseñada.

En las figuras 1.2a y b se puede observar el panel de control y la distribución interna de la caja de control y comunicaciones.

En las figuras 1.3a y b se observan el frente y la distribución interna de la caja de potencia, que incluye las fuentes de poder con sus transformadores, y los circuitos de control de potencia de los motores de paso.

En la figura 1.4 se puede observar la parte mecánica del equipo construido.

1.3.- Requerimientos del Sistema.

Son varios los requerimientos que necesitó el sistema que se ha desarrollado.

Primero, se necesitó de controladores de motores de pasos que permitan manejarlos y controlarlos con mucha versatilidad. Se debió analizar algunos controladores y decidir cual sería el más

óptimo para el diseño. Se escogió los microcontroladores CY545 de CYBERNETIC MICRO SYSTEM, de gran precisión y funcionalidad, que permiten manejar motores de pasos de diferentes fases y en diferentes condiciones. Poseen un conjunto de órdenes y facilidades de programación que hacen simple y sencillo el manejo de los motores de paso. Además, tienen un juego de señales disponibles para manejo del usuario, las que permiten manejar y controlar diferentes funciones. Adicionalmente, poseen dos interfases, un serial y un paralelo.

Segundo, se necesitó de dispositivos electrónicos inteligentes que permitan comunicarse al equipo con el computador. Estos dispositivos deberían reconocer y analizar la información proveniente del computador central, enrutándola al correspondiente controlador CY545 (al eje de coordenadas correspondiente). Entonces, se decidió utilizar los Microcontroladores de Comunicación de CYBERNETIC MICRO SYSTEMS CY-233. Estos microcontroladores reciben o envían la información al computador en forma serial, y la retransmiten en forma paralela a los controladores CY545. Además, sus direcciones pueden ser programadas mediante dip-switches.

Tercero, se debía desarrollar el programa en un ambiente que disponga de una interfase sencilla y amigable. Entonces, se programó bajo WINDOWS, utilizando el lenguaje VISUAL BASIC desarrollado por MICROSOFT. Puesto que este lenguaje tiene deficiencias en comunicaciones seriales, se debió utilizar el paquete auxiliar PDQComm desarrollado por CRESCENT SOFTWARE, que provee a Visual Basic de las órdenes necesarias para realizar comunicaciones seriales seguras.

Cuarto, la parte mecánica desarrollada debía ser confiable y segura, debiendo permitir movimientos precisos. Entonces, se desarrolló un sistema mecánico general para todos los ejes, que sin ser el más óptimo, pero si de los más económicos, permitió que los motores de pasos desplazarán al taladro a la posición requerida.

Quinto, sabiendo que las condiciones de temperatura y humedad podían producir cambios en la precisión del equipo mecánico, se lo montó en una base metálica de Aluminio, cuyos cambios con el calor no son considerablemente grandes. Se podía haber montado el equipo, para obtener alta precisión, sobre una base de roca de granito de por lo menos 6 pulgadas, pero el costo de este material es excesivamente alto.

Sexto, el traductor y los motores de paso deberán permitir ahorrar espacio en el circuito impreso y hacer el diseño más sencillo. Por ello, se escogieron los motores de paso de US-Cyberlab, los que incluyen en su parte posterior el traductor.

Séptimo, se debía utilizar un sistema que permita que el equipo trabaje en forma independiente del computador, o que al menos permita que el computador pueda trabajar en otros programas simultáneamente; de esta manera se garantizaría la operación independiente del equipo y la utilización

del mismo en otras aplicaciones. Por ello, se realizó el interfase entre el computador y el equipo utilizando el pórtico serial, y los microcontroladores inteligentes CY233. Además, se utilizó los controladores de motores de paso CY545 que tienen la capacidad de trabajar con programas disponibles en memoria. Se podía haber realizado el interfase directamente a través del pórtico paralelo, y no sería necesario utilizar ninguna clase de microcontroladores, puesto que a través de las diferentes señales del pórtico paralelo se enviarían directamente las señales de Pulse y Dir a cada uno de los motores de paso y se monitorearían las señales de control. Pero, en este caso, el equipo trabajaría como esclavo del computador, no se podría utilizar programas de Multitasking o programas de spool para impresoras puesto que este tipo de programas interfieren con el tiempo de trabajo del CPU y por lo tanto harían que los motores trabajen inadecuadamente. Además, disminuiría la posibilidad de utilizar el equipo en otras aplicaciones en forma óptima.

1.4.- Flexibilidad.

Durante el diseño, se ha tenido en cuenta la facilidad de uso del aparato construido, de tal manera que pueda ser utilizado por usuarios con poco o ningún conocimiento sobre Electrónica. Por esta razón se escogió un interfase visual para desarrollar el programa, y se ha tratado de utilizar displays y led's que permitirán al usuario trabajar interactivamente con el taladro.

Además, el interfase físico entre el computador y el equipo es un simple cable telefónico que tiene plugs en sus extremos, de tal manera que se podrá conectar fácilmente los dos dispositivos.

Una de las características más importantes que se ha considerado para este diseño es la posibilidad de poder convertir fácilmente el taladro construido en otro tipo de maquinaria, la que podrá prestar otro tipo de servicios, sin hacer cambios fundamentales en el taladro. Así, si se reemplaza el taladro con una punta de plotter, y se altera el programa, se podrá fácilmente dibujar planos con este equipo, utilizando archivos tipo GERBER. Si se reemplaza el taladro con una cuchilla, se podrá modelar figuras sobre metales ligeros, madera, plástico o cera. Adicionalmente, se podrían ejecutar procesos de renderización sobre metales sólidos de forma y tamaño adecuados. Se puede convertir al aparato en un fabricador de hojas de metal usando una cabeza de Plasma o un láser de Co2. Por otra parte, se podría convertir el equipo en un soplete para hacer dibujos de alta precisión. En el campo de Electrónica, se podría usar el equipo para el ensamblaje automatizado de componentes de montaje en superficie (recogerlos y colocarlos), o se lo podría utilizar como un dispositivo automatizado de pruebas "punto a punto".

Puesto que el aparato diseñado cuenta con 64Kb. de memoria, es posible utilizarlo en aplicaciones que dejen al computador libre para otros propósitos, mientras el aparato trabaja en forma independiente. Es decir, se podría descargar una imagen que deberá ser dibujada, y continuar utilizando el computador como procesador de palabras. Sería recomendable utilizar programas desarrollados bajo entorno

WINDOWS, puesto que de esta manera tan solo se conmutaría entre aplicaciones, dejando residente la que no se utilice.

Como se ha detallado, la capacidad de utilización del equipo en otras áreas es realmente ilimitada. Por ello, se ha tenido cuidado en diseñar cada una de sus partes, de tal manera que no se deban reemplazar cuando se utilice el aparato en otro tipo de aplicación. Inclusive, en el caso de aplicaciones que requieran mantener el computador ocupado todo el tiempo, se podría controlar los motores directamente a través del pórtico paralelo.

A continuación, se detallarán algunas aplicaciones en las que se podría utilizar el equipo construido. En algunos casos sería necesario hacer algunas modificaciones o adaptaciones al equipo original.

APLICACIONES EN TRES EJES:

- Cortado de materiales en base a alambres extracalientes.
- Cortado con flama en donde la altura de la antorcha necesita ser ajustada.
- Cortado con láser en donde la altura del láser necesita ser ajustado.
- Perforado de agujeros con variación de profundidades.
- Aplicación de spray con movimientos en tres dimensiones.
- Aplicación de pegamento con el aplicador moviéndose en tres direcciones.
- Unión de materiales mediante el uso de calor.
- Inserción de partes, inserción de tornillos o clavos.
- Control de movimientos para fotografía en tres ejes.
- Automatización del movimiento de una fotografía en un stand.
- Construcción de un plotter de puntas.
- Corte de artículos de cuero o tejido siguiendo un patrón.
- Estampado de hojas de metal o de otro material.

El eje Z puede ser usado de diferentes maneras. Así, se puede controlar el motor de paso, se puede controlar un Relé o un solenoide. El uso normal del motor de paso en el eje Z es el de levantar o bajar una herramienta, de tal manera que la herramienta no esté en contacto con la pieza de trabajo cuando la herramienta se mueve al siguiente punto de trabajo. Adicionalmente, este motor puede utilizarse para abrir o cerrar una válvula o un reóstato. Un relé en la línea de control en la dirección del eje Z puede ser usado para prender o apagar un láser u otro dispositivo eléctrico . Un solenoide disparado por el eje Z podría ser usado para estampar o golpear una hoja o material parecido.

Adicionalmente a las aplicaciones de movimientos en tres ejes, el equipo puede tener un eje o más moviéndose de manera nolineal. Por ejemplo, el eje X podría controlar la rotación de un tablero.

- Corte con cuchilla en vinyl.
- Cortes de vistas en madera.
- Corte de tejidos o cuero siguiendo un patrón.
- Herramientas automáticas para la construcción de partes diminutas.

El cuarto eje, llamado eje "c" (rotación alrededor del eje Z, de tal manera que el flanco de corte de la herramienta es paralelo al mismo), rotará la punta de corte de un cuchillo de tal manera que esta punta siempre mire en la dirección de movimiento de la herramienta mientras la herramienta se mueva a través de la pieza de trabajo. Es decir, el motor del Eje C puede ser usado con su eje rotacional paralelo al eje Z para apuntar la herramienta de corte en la dirección del movimiento de corte.

Capítulo II

Descripción del Sistema

CAPITULO II

Descripción del Sistema

2.1.- Descripción de los Microcontroladores CY-545A

2.1.1.- Introducción

Hace una década CYBERNETIC MICRO SYSTEMS inventó el primer Chip controlador de motores de pasos, el CY500. Luego, fabricaron el CY512 (1.981) y el CY525(1.983). En 1.988 introdujeron un controlador de cuarta generación, el CY545.

Las principales características de este controlador son las siguientes:

- 27.000 Pasos/Seg.
- 16 millones de movimientos de pasos.
- Interfase serial y paralelo.
- Interfase para Display de Led's
- Interfase para Display LCD.

El CY545 es capaz de trabajar con un CPU Host, o independientemente. De hecho, es posible diseñar un sistema basado solamente en este microprocesador, y operar una máquina independientemente.

Características básicas de este microcontrolador son las siguientes:

- Circuito Integrado de 5 Voltios, 40 pines
- Parámetros y Comandos basados en caracteres ASCII.
- Operación independiente o con Host.
- Interfases internos para displays LCD o LED.

2.1.1.1.- Funciones de Control de Movimiento.

Las principales funciones de control de movimiento son las siguientes:

- Velocidad de pasos programable desde 20 pasos/seg. a 27.000 pasos/seg.
- Curvas de aceleración lineal.
- Aceleraciones parciales para movimientos pequeños en tiempos cortos.
- Parámetros individuales para velocidad inicial, velocidad final, y aceleración. Estos parámetros permiten movimientos totalmente programables.

- Movimientos relativos de +/- 16 Mega pasos desde la posición actual.
- Movimientos absolutos dentro del rango de 16 Millones de pasos.
- Movimientos continuos si número especificado de pasos, los que permiten aceleraciones desde la velocidad inicial a la velocidad final, seguida de movimiento indefinido.
- Control interno de la posición durante todo el movimiento.

2.1.1.2.- Señales de Movimiento.

Las principales señales utilizadas para controlar los movimientos son las siguientes:

- Señales de Dirección y Pulsos.
- Señales de control de Dirección internas y externas.
- Señales de Abort y Rampa forzada de bajada para emergencia o control externo de fin de movimiento.
- Señales para controlar los límites CW (Horario) Y CCW (Antihorario) que inhiben el movimiento cuando se alcanza un límite, pero permiten el movimiento en la dirección opuesta.
- Señales de movimiento y No-Movimiento para llevar a los drivers de potencia de los motores a Alto o Bajo (Cuando funcionan o no).
- Modo JOG(Movimientos manuales) operado externamente, con señales de control de movimiento y señales de inicio y de parada. En este modo, se permiten movimientos manuales con velocidades controladas desde 1 a 1K pasos/seg.
- Censor Automático de Búsqueda de posición inicial.

2.1.1.3.- Funciones Generales y Características Especiales.

Este circuito Integrado presenta las siguientes funciones y características especiales:

- Interfase paralelo con 2 líneas de HandShake.
- Interfase Serial con selección de velocidad de transmisión fija o adaptiva.
- Soporte de Memoria Externa de hasta 64K, implementada como RAM, ROM, EPROM o EEPROM.
- Soporte para seteo externo de parámetros, a través de switches tipo Thumb-wheel.
- Display para Mensajes de salida, que puede recibir datos desde el interfase serial o paralelo.
- Adicionalmente soporta el display paralelo HP HDSP-211x de LEDs,l o un display serial LCD CY325.
- Líneas multipropósito que pueden ser controladas por los usuarios para:

a.- Funciones de fijación y limpieza de bit.

b.- Funciones de salto y prueba de bits.

c.- Funciones de espera por valores de bits determinados.

d.- Funciones de búsqueda automática de Posición Inicial (Home).

2.1.2.- Descripción de Pines.

Pines del CY545

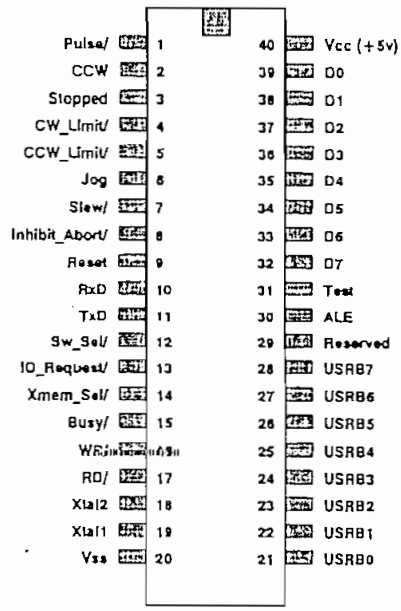


Figura 2.1

La descripción de pines es la siguiente:

Pin 1:(Salida) **PULSE**

Pulsos de Salida, 1 Pulso por Paso.

Pin 2:(Entrada/Salida) **CCW**

Dirección de los Pulsos.

Los pines 1 y 2 deben ir al driver de potencia de los motores de pasos. Un nivel alto en la señal de dirección indica dirección contraria al de las manecillas de reloj, mientras un nivel bajo indica dirección en el sentido de las manecillas de reloj. La señal de pulso es normalmente alta, siendo baja al inicio de cada paso.

Pin 3:(Salida) **STOPPED**

Status del movimiento, en bajo mientras se mueve.

Esta señal puede ser usada para indicar que el movimiento se ha terminado. Puede también controlar la potencia del driver del motor de pasos, llevándola a alto cuando el motor funciona, y a potencia de parque cuando el motor se para.

Pin 4:(Entrada) **CW_LIMIT/**

Indica que se alcanzó el límite en el sentido de las manecillas de reloj.

Pin 5:(Entrada) **CCW_LIMIT/**

Indica que se alcanzó el límite en el sentido contrario de las manecillas de reloj.

Si estas señales están en bajo, indican que un límite se ha alcanzado. El CY545 no continuará moviendo en esa dirección hasta que el límite siga siendo alcanzado. Movimientos normales son posibles en la otra dirección.

Pin 6:(Entrada/Salida) **JOG**

Control Manual de pasos.

Esta señal es la entrada para control manual. Cuando está flotando, el CY545 no moverá manualmente el motor. Sin embargo, si está señal es conectada a tierra, el CY545 moverá el motor en dirección CW, y si es conectado a 5V, el CY545 moverá el motor en dirección CCW. La velocidad de los pasos en el modo JOG es derivada del Parámetro de Velocidad Inicial y la tabla de velocidad normal de pasos, dividiendo la velocidad seleccionada para 20. Esto da un rango de cerca 1 paso/segundo a cerca de 1000 pasos/segundo. La señal de entrada está siempre habilitada, y es chequeada por el CY545 mientras no se está ejecutando otro comando.

Pin 7:(Entrada/Salida) **SLEW/**

Señal indicadora de Slew.

Pin 8:(Entrada) **INHIBIT_ABORT/**

Control de Movimiento externo.

Este grupo de señales indica y controla el status del movimiento. SLEW/ indica cuando la máxima velocidad de pasos seleccionada ha sido alcanzada. Si es llevada a un nivel lógico Bajo externamente, al inicio del movimiento, produce que el CY545 entre al modo continuo de pasos. INHIBIT_ABORT/ puede impedir un movimiento o forzar una rampa de caída, deteniendo el movimiento cuando del CY-545.

Pin 9:(Entrada) **RESET**

Resetea al CY545 con un nivel lógico alto.

Pin 10:(Entrada) **RxD**

Entrada serial de datos al CY545.

Pin 11:(Salida) TxD

Transmisión serial de datos desde el CY-545

Estas son señales TTL que pueden ser usadas para enviar comandos o recibirlos desde el CY545. La adición de un driver/receiver para conversión a niveles de voltaje RS-232, como el Maxim MAX233, permitirá que cualquier dispositivo con RS-232, como el pórtico de comunicaciones de una IBM PC se comunique serialmente con el CY-545.

Pin 12:(Salida)SW_SEL/

Selección de Switch.

Está señal está activa cuando el CY545 está leyendo un parámetro desde los switch Thumb Wheel. La dirección de los switchs es generada por el bus de datos paralelo.

Pin 13:(Entrada) IO_REQUEST/

Señal de entrada de Handshake para pórtico paralelo.

Esta señal va a un nivel lógico bajo cuando un comando paralelo es enviado al CY-545. Esto es usado como parte del handshake paralelo. Si no es usado el pórtico paralelo, esta señal puede ser usada para seleccionar una velocidad fija de transmisión de datos:

0 = 300 baud.
1 = 2400 baud.
F = 9600 baud.

Pin 14:(Entrada/Salida) XMEM_SEL/

Selecciona memoria local externa.

Esta señal va a un nivel lógico bajo cuando el CY545 lee o escribe a la memoria externa. Datos y direcciones son generados por el bus de datos paralelo. El CY545 soporta hasta 64 Kbytes de memoria local externa. Cuando esta señal va abajo el momento del reset de encendido, el CY545 no revisará la secuencia de AUTOINICIO o automáticamente ejecutará comandos desde la memoria externa.

Pin 15:(Entrada/Salida) BUSY/

Esta señal también es utilizada para el Handshake paralelo.

BUSY/ es usada con IO_REQUEST/ para implementar el handshake paralelo de dos líneas. Esta señal indica que el CY545 ha aceptado un caracter paralelo desde el equipo que le envió la señal IO_REQUEST. Si el Handshake paralelo no es usado, este pin puede ser colocado a un nivel lógico bajo para seleccionar una velocidad fija de transmisión serial a través de la señal IO_REQUEST.

Pin 16:(Salida) WR/

Strobe de escritura.

Pin 17:(Salida) RD/

Strobe de Lectura.

Estos Strobe están activos mientras el CY545 está leyendo en los Thumb wheel switches, en el display, en la memoria, o mientras está escribiendo al display externo o la memoria.

Pin 18:(Entrada) XTAL2

Conección del cristal.

Pin 19:(Entrada) XTAL1

Conección del cristal.

Un cristal o una fuente de reloj externo debe ser conectado a estos pines, con valores de frecuencia que van desde los 3.5 MHz hasta los 16 MHz. Para comunicaciones seriales , un cristal de 11 MHz dará velocidades de transmisión estándares.

Pin 20:(Entrada) VSS

Tierra de la fuente de poder.

Pin 21:(Entrada/Salida) USRB0

Bit Programable por el usuario, bit 0

Pin 22:(Entrada/Salida) USRB1

Bit Programable por el usuario, bit 1

Pin 23:(Entrada/Salida) USRB2

Bit Programable por el usuario, bit 2

Pin 24:(Entrada/Salida) USRB3

Bit Programable por el usuario, bit 3

Pin 25:(Entrada/Salida) USRB4

Bit Programable por el usuario, bit 4

Pin 26:(Entrada/Salida) USRB5

Bit Programable por el usuario, bit 5

Pin 27:(Entrada/Salida) USRB6

Bit Programable por el usuario, bit 6 o FPL/ o CTS/

Pin 28:(Entrada/Salida) USRB7

Bit Programable por el usuario, bit 7 o HP_SEL/

Este grupo de señales es usado para múltiples funciones seleccionables por el usuario, incluyendo set y clear, test de bits y saltos, esperas para valores de señal, y funciones automáticas de búsqueda de posición inicial. Cualquiera de estas funciones puede ser ejecutada en cualquiera de estos bits. Los Bits 6 y 7 son también utilizados para funciones específicas como: Control FPL para el CY233, Clear to Send serial, o selección de display HP paralelo local. Si la aplicación no utiliza un CY233, CTS o un Display HP, estos bits pueden ser usados para otras funciones.

Pin 29:(Salida) RESERVED

Señal reservada, no usada en el CY545.

Pin 30:(Salida) ALE

Habilita el Latch de direcciones.

Esta señal es usada para demultiplexar la dirección más baja de memoria desde el bus de datos durante operaciones con memoria externa, display o thumb wheel switch.

Pin 31:(Entrada) TEST

Señal interna de Test, conectada a VCC.

Pin 32:(Entrada/Salida) D7

Bus de datos paralelo, bit 7, MSB

Pin 33:(Entrada/Salida) D6

Bus de datos paralelo, bit 6.

Pin 34:(Entrada/Salida) D5

Bus de datos paralelo, bit 5.

Pin 35:(Entrada/Salida) D4

Bus de datos paralelo, bit 4.

Pin 36:(Entrada/Salida) D3

Bus de datos paralelo, bit 3.

Pin 37:(Entrada/Salida) D2

Bus de datos paralelo, bit 2.

Pin 38:(Entrada/Salida) D1

Bus de datos paralelo, bit 1.

Pin 39:(Entrada/Salida) D0

Bus de datos paralelo, bit 0, LSB.

Este grupo de señales es el bus paralelo de datos, usado para hacer escrituras o lecturas paralelas. Durante el acceso a thumb wheel switches, memoria o display, el byte menos significativo de la dirección es colocado en estas líneas, justamente antes de la transferencia paralela. También son aceptados comandos paralelos en estas líneas, usando las señales de control de Handshake. Estas líneas son de colector abierto, siendo necesarias resistencias de pullup cuando son utilizadas como salidas.

Pin 40:(Entrada) VCC

Entrada de la fuente de poder: +5 Volts.

2.1.3.- Comandos para Interfases.

El CY545 soporta dos tipos de comandos básicos para interfases, interfase serial e interfase paralelo.

2.1.3.1.- Interfase Paralelo.

El interfase paralelo usa dos señales de handshake: IO_REQUEST y BUSY. Cuando se desee enviar un comando al CY545, se debe chequear primero si está ocupado; la señal BUSY debería estar en alto. Entonces, se puede enviar los caracteres por las líneas del bus de datos paralelo.

Después, se debe llevar la señal IO_REQUEST a un nivel lógico bajo, indicando que un caracter está disponible. Luego, se debe esperar hasta que el CY545 vaya a ocupado, con un nivel lógico bajo en la señal de BUSY, indicando que el CY545 ha leído el caracter.

Finalmente, se debe retirar el caracter del bus de datos y llevar IO_REQUEST a un nivel alto.

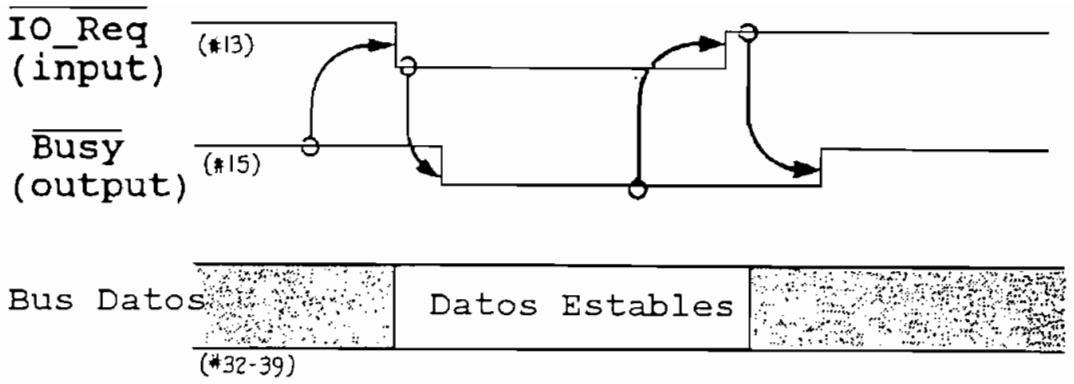


Figura 2.2

Esta función es repetida hasta que todos los caracteres han sido enviados al CY-545. En el modo directo de comandos, el CY545 ejecuta cada comando luego de recibido. Esto significa que el CY545 irá a ocupado por un período de tiempo más largo luego de recibido el último caracter de un comando; de hecho, hasta que el comando sea ejecutado.

El interfase paralelo es compatible con el controlador de redes CY233; por tal razón, se puede comandar varios CY545 desde el pórtico serial de un computador, usando un CY233 como "Front End" de la red, para cada CY545.

2.1.3.2.- Interfase del CY545 al CY233.

Para un interfase basado en el CY233 , la señal DAV del CY233 es conectada a la señal IO_REQUEST del CY545, y la señal ACK del CY233 es conectada a la señal BUSY del CY545. Adicionalmente, la señal FPL del CY233 debe ser conectada a USRB6 del CY545, el cual comparte la función FPL en esta línea.

Un ejemplo de esta conección es el siguiente:

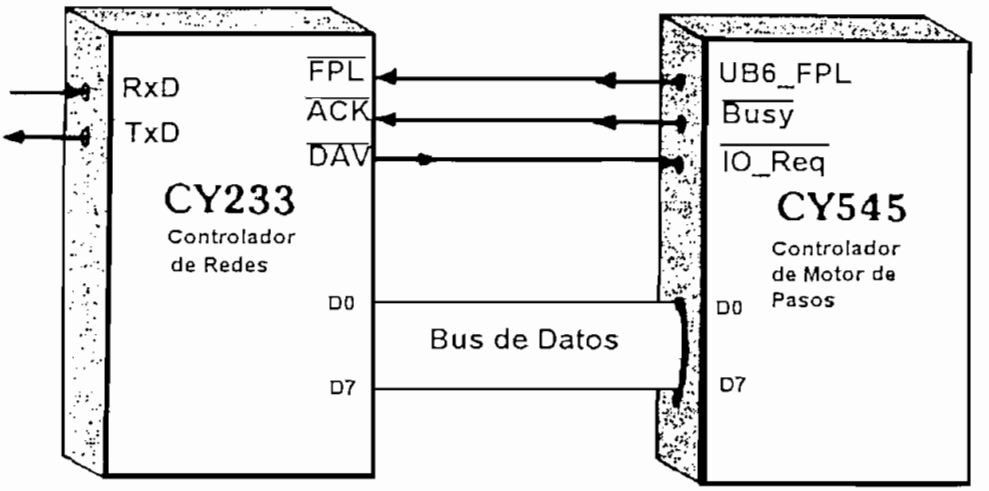


Figura 2.3

El controlador de redes CY233 permite conectar hasta 255 dispositivos a una única línea de comunicación serial RS-232, tales como un pórtico COM1 o COM2 de un IBM-PC, con una única dirección asignada a cada dispositivo. El CY233 es ideal para sistemas distribuidos donde se requiere un control central,pero, no son necesarias velocidades altas.

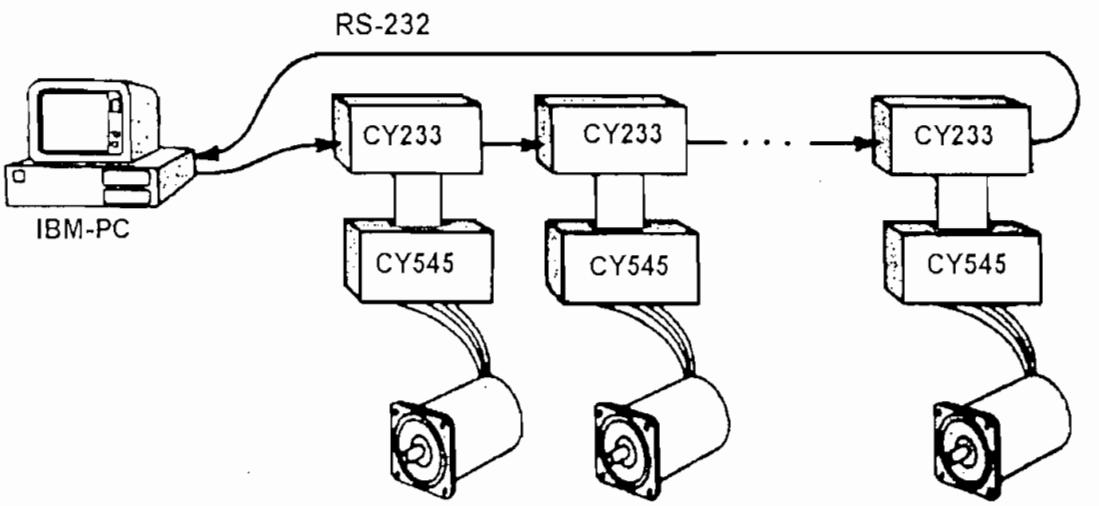


Figura 2.4

2.1.3.3.- El Interfase Serial.

El CY545 también dispone de un interfase serial de comunicaciones, el cual puede ser conectado a un computador o a un terminal. Puesto que los niveles de señales del CY545 son TTL, son necesarios drivers y receivers externos para RS-232, los que convertirán los niveles de voltaje RS-232 a niveles TTL que podrán ser leídos por el CY545. El Chip MAXIM MAX-233 realiza esta función, utilizando un fuente única de 5 voltios.

El interfase serial puede ser operado de dos maneras, una manera con selección fija de la velocidad de transmisión, y otra con una velocidad adaptiva, seleccionada enviando dos caracteres de retorno.

El modo de velocidad fija es seleccionado llevando la señal BUSY a bajo, así el CY545 leerá un 0 Lógico al encendido. El valor de la línea IO_REQUEST determinará la velocidad de transmisión:

F 9600
1 2400
0 300

Cuando la línea IO_REQUEST está flotante, se selecciona 9600 baudios. Cuando está en alto, se seleccionan 2400 baudios, y cuando está en bajo se selecciona 300 baudios. Para que estas velocidades sean correctas, el CY545 debe ser operado con un reloj de 11 MHz; valores diferentes de reloj escalarán linealmente la velocidad de transmisión.

Velocidad Serial Fija
(Sin interfase paralelo)

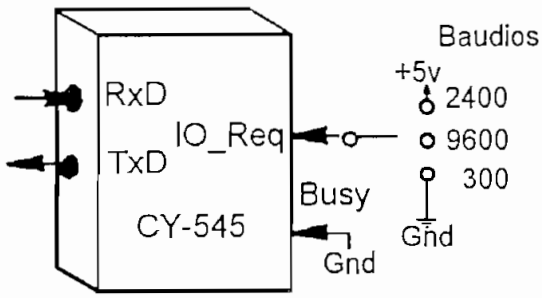


Figura 2.5 a

Velocidad Serial Adaptiva
(Con interfase paralela)

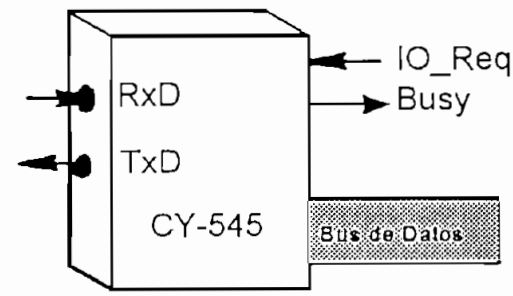


Figura 2.5 b

Un método alternativo de selección de la velocidad de transmisión es a través del Modo de Comando Paralelo. Para este comando, son disponibles 9 velocidades de transmisión, desde 300 a 57600.

Se debe notar que el uso de una velocidad fija de comunicación serial utilizando BUSY y IO_REQUEST , no permitirá la utilización del interfase paralelo, ya que las líneas son usadas para seleccionar la velocidad de transmisión. Esto será útil si solo se va a utilizar el interfase serial. Sin embargo, si se van a utilizar los dos interfases, la velocidad de transmisión deberá ser necesariamente adaptiva, o a través del modo de comandos paralelo.

El default es el modo serial adaptivo, cuando la señal BUSY está flotando. En este modo la velocidad de transmisión no se fija hasta que el CY545 no recibe dos caracteres de retorno. Se debe enviar estos caracteres de retorno, luego del encèndido o luego de un reset (De Hardware o de Software). Una vez que los caracteres de retorno se han recibido, se pueden enviar comandos al CY545.

Cuando se trabaja con el interfase serial, la mejor frecuencia de cristal es la de 11 MHz. Esta frecuencia provee el un margen más amplio de velocidades de transmisión, con el menor error de transmisión. En 11 MHz, se puede seleccionar velocidades de transmisión de hasta 19200 baudios. Sin embargo, algunas otras funciones del CY545, tales como la velocidad de los pasos y los tiempos de retraso, han sido calibradas para frecuencias de reloj de 12 MHz; esto quiere decir que estas funciones se ejecutarán algo más lentas con una frecuencia de 11 Mhz.

Cuando se opera el CY545 a 16 MHz, se limita la operación del interfase serial a 4800 baudios. El error a velocidades de transmisión mayores puede ser muy considerable.

Sin embargo, hay una frecuencia de cristal a la que el CY545 puede trabajar en modo adaptivo: 14.7456 MHz. Con esta frecuencia, la resolución interna del timer del CY545 permite velocidades de transmisión de 300 a 19200 baudios, incluyendo dos más altas: 38400 y 76800 baudios.

Además, el interfase serial del CY545 trabaja con caracteres de formato fijo, los cuales son siempre de 8 bits de datos, sin paridad y un bit de parada. Los 8 bits de datos son usados para interpretar el valor de los caracteres de comando.

Finalmente, el CY545 implementa un señal Clear to Send (CTS/), la cual puede ser usada por el Host para indicar cuando el CY545 está ocupado en un proceso. No se debería enviar ningún caracter adicional al CY545 mientras CTS/ está en un nivel TTL alto. Una vez que el CY545 ha terminado de procesar un comando, CTS/ será habilitado nuevamente, indicando que el CY545 está listo para recibir más caracteres seriales. La polaridad de la señal CTS/ del CY545 le permite ser conectada directamente

a la señal CTS del Host, a través de un driver Standard de RS232. No se requiere de lógica adicional.

La señal CTS/ comparte funciones con el User Bit 6, el cual es también la señal FPL/ para el CY233. Puesto que esta señal tiene múltiples funciones, la función CTS/ es deshabilitada en el momento del arranque o en alguna operación de reset. Se debe enviar un comando Mode con MB5 activado para habilitar la señal CTS/.

2.1.4.- Sumario de Comandos.

Los comandos en el CY545 pueden ser enviados en dos formatos: ASCII o Binario. Cada comando está formado por una letra mayúscula y un argumento numérico. La letra y el argumento están separados por un espacio; argumentos múltiples pueden ser separados por un espacio o una coma. El comando termina con caracter de retorno. Los comandos sin argumentos, simplemente usan la letra, seguidos directamente por el caracter de retorno.

2.1.4.1.- Formato de los Comandos.

El CY545 soporta diferentes formatos de comandos, los cuales pueden ser divididos en dos grandes categorías, formato ASCII y formato BINARIO. El modo más popular de operación es el modo ASCII. Sin embargo, la mayor ventaja del modo binario es el ahorro de caracteres; esto quiere decir que se necesitan menos caracteres para representar los valores de un parámetro. La principal desventaja del modo binario es la dificultad de lectura para las personas; además si el CY545 y el Host se desincronizan en modo binario, debido a la falta de conteo de datos, sería extremadamente difícil resincronizarlos a los dos sin un reset.

2.1.4.2.- Formato de Comandos ASCII.

Todos los formatos de Comando ASCII empiezan con una letra ASCII que selecciona el comando. Si el comando tiene también parámetros, la letra debe ser seguida por un espacio, y luego el valor del parámetro. Múltiples parámetros, tales como los usados para el comando LOOP, pueden ser separados por un espacio o una coma. En los ejemplos se utilizará normalmente una coma. Todos los comandos ASCII terminan con el caracter de retorno ASCII, que se muestra como <cr> en los ejemplos. Comandos sin parámetros, tales como el comando GO, usan solamente una letra de comando, seguido inmediatamente por un caracter de retorno.

Por ejemplo:

G <cr>	Comando Go, sin parámetro.
I <cr>	Comando Initialize, sin parámetro.

Comandos con parámetros pueden usar dos formatos para los valores de los parámetros en el modo ASCII, un formato decimal, que consiste de dígitos del 0 al 9, o un formato hexadecimal, que consiste de dígitos del 0 al 9, letras de A a la F, y el sufijo H. Los parámetros hexadecimales deben comenzar con un dígito.

Por ejemplo:

R 135<cr>	Comando Rate, parámetro decimal simple.
R 87H<cr>	Parámetro similar, pero en forma hex.
R ABH<cr>	Parámetro ilegal, ningún dígito al inicio.
R 0AB<cr>	Parámetro ilegal, sin sufijo H.
R 0ABH<cr>	Parámetro válido.
D 12345<cr>	Parámetro de 16 bits en decimal
D 1A3BH<cr>	Parámetro de 16 bit en Hex.
L 35,27<cr>	Comando Loop, parámetros múltiples
L 23H,27<cr>	Formato mezclado de parámetros.

Los valores de los parámetros pueden ser ingresados con cualquier número de dígitos, pero el comando limitará el valor a un byte si el rango del parámetro es de un solo byte.

Por ejemplo:

R 0ABCH<cr>	El valor es tomado como 0BCH = 188
R 350<cr>	El valor es tomado como 94=350-256.

Los parámetros de 16 bits son limitados a 65535 (0FFFFH) y los parámetros de 8 bits son limitados a 255(0FFH), mientras que los parámetros de 24 bits, usados para conteo de posiciones y de pasos, son limitados a 16777215(0FFFFFFH).

2.1.4.3.- Formatos de comando binario.

El formato binario también empieza con la letra ASCII para cada comando. Si embargo, el formato para los parámetros es diferente en modo binario. La letra del comando siempre va seguida inmediatamente por un contador binario de datos, que representa el número de bytes de datos que el

comando necesita. Para comandos sin parámetros, como el comando GO, el contador de datos es un valor binario cero (No el caracter ASCII 0). Para comandos con parámetros, el contador de datos representa el número de bytes necesitados para especificar los parámetros. Por ejemplo, el comando Number debe tener siempre un contador de datos de tres, para su parámetro de 24 bits, aún si el valor del byte más significante es cero.

El comando QUIT, el cual finaliza la entrada de comandos en la memoria externa, no debería ir con un contador de datos. Este comando va seguido inmediatamente por la letra ASCII del próximo comando. También, el comando STOP debe usar el valor binario 0, no el caracter ASCII "0", seguido por un contador de datos el cual es también 0.

Todos los parámetros múltiples deben ser ingresados con el byte menos significativo primero. Por ejemplo, el comando ZLOOP tiene un contador de 16 bits, seguido por una dirección de 8 bits, por lo tanto tiene un contador un contador de datos de tres, con la siguiente secuencia del byte de datos: LSByte del contador, el MSByte del contador, entonces la dirección del salto. Se va a realizar algunos ejemplos; los bytes de datos son mostrados como valores Hex, con un espacio separando cada uno por claridad, pero son enviados al CY545 como valores de 8 bits, sin los espacios.

G 00	Sin argumento, Contador de bytes 0
R 01 87	Argumento de un solo byte.
D 02 3B 1A	Argumento de dos bytes, valor 1A3BH
N 03 11 22 33	Argumento de tres bytes, valor 332211H
Z 03 21 04 92	Argumentos Múltiples, Z 0421H,92H

La siguiente sección describe cada uno de los comandos del CY545. Estos comandos han sido organizados en tres de funciones relacionadas.

2.1.4.4.- Comandos de control del motor.

Los comandos de control del motor son usados para fijar las condiciones para el movimiento del motor de pasos. Varios comandos fijan parámetros relevantes, y dos comandos causan el movimiento del motor. Adicionalmente, la señal JOG puede ser usada para mover el motor manualmente en cualquier momento.

2.1.4.4.1.- Velocidad de los pasos.

Tres comandos fijan las funciones básicas del movimiento en el CY545. Ellos son:

R Rate	Fija la velocidad máxima de los pasos.
F Rate	Fija la velocidad inicial de los pasos.
S Slope	Fija la aceleración.

Los parámetros para Velocidad Inicial y Velocidad Final son valores de un solo byte que seleccionan la velocidad inicial y la velocidad máxima para el motor. Estos parámetros están indicados en la Tabla de Velocidades que se encuentra en el ANEXO. La velocidad de los pasos (Pasos/seg) viene de una combinación de los valores en esta tabla y la frecuencia de operación del CY545. Los valores de los pasos por segundo de esta tabla han sido calculados para un cristal de 12 MHz. El uso de otras frecuencias causa un escalamiento lineal de las velocidades, y puede ser calculado multiplicando la velocidad dada por el factor Fcy/12, donde Fcy es la frecuencia actual del cristal (en MHz) usada en el CY545.

El parámetro para el comando SLOPE determina cuan rápidamente el CY545 acelera y desacelera cuando se mueve. Este es también un valor de un solo byte, y su rango va desde 1 a 255. Pequeños valores de Slope seleccionan aceleraciones bajas, mientras que valores de slope más altos, seleccionan aceleraciones mayores. En la práctica, se debe experimentar con los valores del parámetro Slope para determinar la aceleración óptima para la aplicación específica.

Las aceleraciones y desaceleraciones son simétricas, y son controladas por este parámetro. Si la distancia del movimiento es demasiado pequeña para permitir que el CY545 alcance la velocidad máxima, lo hará en base a una velocidad parcial, alcanzando la velocidad más alta posible, tomando en cuenta el parámetro Slope, hasta tener que desacelerar para parar en la posición a la que se le ha enviado.

2.1.4.4.2.- Comandos de dirección de movimientos.

Tres comandos determinan la dirección de los movimientos del CY545, en el modo relativo:

N Num24	Fija el número de pasos
+	Selecciona Dirección CW.
-	Selecciona Dirección CCW.

El comando **NUMBER** especifica el número de pasos que el CY545 debe moverse en modo relativo. Usa un parámetro de 24 bits, lo que quiere decir que el CY545 puede moverse + o - 16 millones de pasos desde la posición actual en modo relativo. Cuando el comando es enviado, el CY545 almacena el número de pasos en un registro interno N, hasta que el comando GO sea enviado.

El comando + selecciona el movimiento en dirección de las manecillas del reloj. Con este

comando, la señal CCW va a un nivel lógico bajo.

El comando - selecciona la dirección del movimiento en el sentido contrario al de las manecillas de reloj. Adicionalmente, la dirección de los pasos puede ser controlada externamente, llevando al pin CCW a un nivel de voltaje determinado.

2.1.4.4.3.- Comandos para el modo de pasos.

Dos comandos del CY545 controlan el modo de operación de pasos para el CY545. Ellos son:

A	Pos24 Fija la posición actual.
C	Selecciona el modo continuo de pasos.

El comando A fija la posición actual de pasos del CY545 al valor determinado por su parámetro. Es un valor de 24 bits, así que se puede especificar posiciones desde 0 a 16777215. Normalmente, este comando es usado después de un movimiento a una posición conocida, tales como la posición HOME, y permite calibrar al CY545 para que esta posición tenga siempre el mismo número.

Cuando el CY545 está moviendo el motor, se registra siempre la posición interna. Esta posición es incrementada con movimientos CW, y decrementada con movimientos CCW. Se puede solicitar información al CY545 sobre su posición, mientras no está moviendo el motor.

Este valor de posición no es demasiado importante para movimientos relativos, en los cuales se especifica la dirección y el número de pasos. Sin embargo, es muy importante para movimientos absolutos, en los cuales se especifican posiciones de envío, y el CY545 calcula la dirección y el número de pasos requeridos para alcanzar dicha posición.

El comando CONTINOUS fija un modo especial de pasos CY-545. En este modo el CY545 acelera desde la velocidad inicial hasta la final, y entonces se mueve continuamente. No se detendrá en ninguna posición o número de pasos específico. Para detenerlo, la señal INHIBIT_ABORT debe ser colocado a 0L, forzando al CY545 a un modo de desaceleración. El CY545 parará si la señal INHIBIT_ABORT permanece en bajo.

El comando C fija el modo continuo, pero el movimiento no comienza con este comando. El modo continuo también puede ser seleccionado por medio de una señal externa. SLEW, la que debe estar en bajo cuando el CY545 inicia el movimiento.

2.1.4.4.4.- Comandos para el movimiento del motor.

Finalmente, dos comandos son usados para hacer que el motor se mueva. Ellos son:

G	Pasos en modo relativo.
P pos24	Movimiento a la posición determinada.

El comando GO selecciona el modo relativo de pasos. El CY545 moverá el motor basado en el último número de pasos especificado y en la dirección de pasos especificada. Movimientos en el modo continuo de pasos también debe comenzar con este comando.

Movimientos en modo absoluto son seleccionados con el comando **Position.** Este comando especifica la posición deseada y el inicio del movimiento. Cuando se especifica la posición deseada, el CY545 calculará el número de pasos y la dirección del movimiento, basado en la actual posición.

Si la posición deseada es mayor que la posición actual, el CY545 se moverá en la dirección CW. Si la posición deseada es menos que la posición actual, el CY545 se moverá en la dirección CCW.

2.1.4.5.- Comandos para Manejo de Bits.

Varios comandos permiten especificar cuales Bits seleccionables por el usuario (User Bits). El CY545 tiene por lo menos 6 señales disponibles para usos de propósito general. Se puede usar más señales si no se utiliza el CY233, o el Display de Leds HP, o el interfase paralelo.

Cada uno de estos comandos incluye un parámetro el cual especifica cual bit o combinación de bits usará el comando.

2.1.4.5.1.- User Bits.

Este comando le permite fijar o limpiar un bit específico.

B Bit Fija o Limpia un bit específico

En el modo ASCII, este comando permite un sufijo especial para invertir el valor de bit especificado. Este prefijo el caracter "/", y es usado así:

B 2<cr>	Fija el User Bit 2 a alto.
/B 2<cr>	Limpia el User Bit 2 a bajo.

La misma función del prefijo "/" se la puede realizar de la siguiente manera:

B 02H<cr>	Fija el Bit 2
B 12H<cr>	Limpia el Bit 2.

El caracter "/" es equivalente a realizar una operación OR exclusiva (XOR) del valor del bit y el valor 10H (16 decimal).

Puesto que los User Bits 0 al 5 siempre están disponibles para las aplicaciones, pueden ser tratados como grupo.

2.1.4.5.2.- Saltos y Esperas dependientes de Bits.

Dos funciones del CY545 usan los bits para funciones especiales de test. Estas son:

T bit,addr	Salto a la dirección especificada hasta que el bit alcance el valor especificado.
W bit	Espera hasta que el bit alcance el valor especificado.

Para estos comandos, se considera lo siguiente:

00H a 07H	Verifica si son 1 los user bits 0 al 7
08H a 0FH	Verifica si son 1 los bits del bus de datos 0 al 7
10H a 17H	Verifica si son 0 los user bits 0 al 7
18H a 1FH	Verifica si son 0 los bits del bus de datos 0 al 7
80H a 0BFH	Verifica si los bits 0 al 5 alcanzan un valor en grupo.

Nótese que los USRB0 a USRB5 pueden ser tratados como un grupos de 6 bits.

El comando Til saltará a la dirección especificada por el byte hasta que se alcance el valor especificado por el parámetro en el USRB, que puede ser un valor alto o bajo. Cuando se alcanza este valor, ya no se produce el salto y se continua al siguiente comando.

El comando TIL solo se lo usa en programas en memoria externa. La dirección de salto es de un solo byte, limitando el salto a direcciones en la misma página de 256 bytes. El comando Til puede saltar sobre si mismo.

El comando WAIT espera a que se alcance un valor en los bits, comparándolo con el valor del parámetro. Cuando no se alcanza el valor, el CY545 continua en espera, probando la señal hasta que se

alcance el valor especificado. Una vez que se ha detectado que se alcanzó el valor, el CY545 continua con el próximo comando.

Si se envía un nuevo comando a través del interfase serial o paralelo, cuando el CY545 está esperando en el comando WAIT, el CY545 saldrá del comando WAIT y ejecutará el nuevo comando. Este es un mecanismo para evitar que el CY545 permanezca en ciclos de repetición indefinidamente.

Normalmente se usa el comando WAIT en programas con memoria externa, para sincronizar el CY545 con algún otro control o señal externas.

2.1.4.5.3.- Comando HOME.

Finalmente, el comando HOME usa los bits para controlar movimientos especiales, y causar movimientos durante test de los bits.

H bit Mueve el motor a una posición inicial, usando una señal específica.

Los valores de los parámetros de estos comandos son tratados de la misma manera que en el comando Til o Wait. Esto es, para el comando HOME, se chequea los bits hasta que alcancen un valor específico.

Como el CY545 mueve el motor en este comando, lo hace a una velocidad constante, sin aceleración o desaceleración. La velocidad deseada se escoge con el parámetro F. La velocidad para el comando HOME y para JOG, se determina dividiendo la velocidad escogida para 20. Cuando el reloj es de 12 MHz, esto da un rango desde 1 paso/segundo a cerca de 1000 pasos/segundo para este comando.

Cuando se ejecuta un comando HOME, primero se verifica que se ha alcanzado un bit especificado, utilizando las señales externas. Si se ha alcanzado, el CY545 comenzará a mover el motor en la dirección CCW, y continuará moviéndolo en esta dirección hasta que ya no se haya alcanzado este bit.

Cuando no haya se haya alcanzado este bit, el CY545 moverá el motor en la dirección CW. Continuará moviéndolo hasta que las señales alcancen el valor del parámetro. En este punto el CY545 para, y fija en 0 la posición actual.

Este mecanismo permite encontrar siempre la misma posición mecánica inicial.

2.1.4.6.- Comandos para soporte de memoria externa.

El CY545 dispone de varios comandos que permiten el control de memoria externa. Estos comandos controlan el direccionamiento, la entrada de datos, y la ejecución de los comandos desde la memoria.

2.1.4.6.1.- Puntero de dirección de memoria externa.

Este comando es usado para fijar el puntero de direcciones.

Y Addr16	**Fija el puntero de dirección externa.**

El argumento de 16 bits se coloca en un registro de dirección de memoria del CY545. Otros comandos pueden usar este valor para ejecutar su función.

El comando Y no cambia modos u otras características del CY545, así que cuando se lo envía como un comando directo, simplemente fija el puntero de direcciones. Sin embargo, cuando el comando Y se le ejecuta desde memoria externa, el CY545 ejecutará los comandos desde esta nueva dirección; esto quiere decir que el comando Y permite saltos de direcciones de hasta 16 bits.

2.1.4.6.2.- Comandos de escritura en memoria.

Los dos comandos que controlan la entrada de datos en la memoria son:

E	**Inicia la entrada de comandos seguidos en la memoria.**
Q	**Termina la entrada de comandos en la memoria.**

El comando ENTER se lo debe enviar al CY545 en el modo de comandos directo. Una vez que el comando se ha ejecutado, el CY-545 cambiará al modo de entrada de programas. Todos los comandos siguientes se escribirán en la memoria externa. El CY-545 no ejecutará estos comandos, sólo los escribirá en la memoria.

Los comandos se escriben en la memoria desde la dirección fijada por el Comando Y, y el puntero de dirección se incrementará después de la escritura de cada caracter.

El contenido de la memoria consiste de una cadena de caracteres, los mismos que son idénticos a los enviados al CY545 en el modo directo de comandos. Cada caracter enviado al CY545 será grabado en la memoria exactamente como se lo ha enviado.

El comando QUIT permite interrumpir la escritura de comandos en la memoria externa. Esto quiere decir que el CY545 regresa al modo de comandos directos, en el cual cada comandos es ejecutado una vez recibido. El puntero de dirección de memoria contiene la dirección del próximo caracter que debía se escrito, y la diferencia entre el valor de la dirección inicial y el valor de la dirección final, indicará la cantidad de memoria utilizada por el programa. Antes de correr un programa debería utilizarse el comando Y para direccionar el puntero de direcciones de memoria a la dirección de inicio del programa.

El caracter de retorno es opcional después del comando QUIT en el modo ASCII, y en el modo Binario no se usa el contador de datos después del comando Quit.

2.1.4.6.3.- Comandos para ejecutar programas desde memoria externa.

Dos comandos permiten ejecutar programas desde memoria externa. Ellos son:

X	Ejecuta comandos desde memoria externa.
0	Para la ejecución de los comandos desde memoria externa.

El comando EXECUTE cambia de modo al CY545, desde el modo de comandos directo al modo de ejecución en memoria. El programa comenzará a ejecutarse desde la dirección que indique el puntero de direcciones de memoria. El comando Y debería preceder normalmente al comando X, para fijar la dirección de inicio del programa.

En el modo de ejecución en memoria, el CY545 leerá los caracteres de comando desde la memoria, incrementando el puntero de direcciones después de cada lectura. Cuando se ha leído un comando completo, este es ejecutado desde la misma forma que los comandos directos.

Cuando se termina la ejecución de un comando, el CY545 leerá el próximo comando desde la memoria y lo ejecutará. Este proceso continua hasta que el CY545 encuentra un comando de parada (0).

El comando ZERO (Caracter ASCII 0, no el valor binario 0) es el comando que termina la ejecución de un programa. Cuando este comando se ejecuta, el CY545 cambia del modo de ejecución de memoria al modo de comandos directo. Nótese que en el modo binario, el valor binario CERO debe utilizarse para detener la ejecución de un programa, seguido por el contador de datos CERO.

Si el CY545 es operado en el modo ASCII, se puede detener la ejecución de un programa desde el Host, enviando un caracter de retorno. El CY545 leerá este caracter entre la ejecución de un comando y la lectura del próximo comando desde memoria externa, y detendrá la ejecución del programa.

El modo de comandos binario, no hay un caracter que permita al Host detener la ejecución de un

programa.

2.1.4.7.- Comandos para saltos.

Varios comandos del CY545 permiten controlar los saltos y el flujo de programa, cuando el CY545 está ejecutando programas desde memoria externa. Estos comando normalmente son usados dentro de memoria y no como comandos directos. A pesar de ello, el CY545 puede ejecutarlos en modo directo, pero los resultados pueden ser diferentes a los que se obtendrían si los comandos fueran ejecutados desde el modo de programa. Estos comandos son:

J Addr	Salto a una dirección en la página actual.
L Cnt,Addr	Loop a una dirección, Cnt veces.
Z Cnt16, Addr	Zloop a una dirección, Cnt veces.

En adición a estos comandos, el comando Y, cuando se ejecuta desde memoria externa, puede producir un salto de 16 bits a cualquier dirección de la memoria externa.

Todos los comandos de la lista anterior usan valores de dirección de 8 bits. Esto restringe el rango de direcciones a solo 256 bytes de la página actual de memoria. Cuando se produce un salto, el byte más bajo del puntero de la dirección actual es reemplazado por el parámetro de dirección de 8 bits, y la ejecución del programa continua en esta nueva dirección de 16 bits. Si el salto cae en una nueva página de dirección, el puntero es incrementado antes de que el comando sea ejecutado, y así el salto se producirá a una nueva página de memoria.

El comando JUMP permite alcanzar cualquier dirección en la página actual de memoria.

El comando LOOP provee saltos repetidos, con un número de interacciones específico. Los dos parámetros son de un byte, por lo tanto un loop puede ser repetido hasta 255 veces, y el salto debe producirse dentro de la misma página de memoria.

El comando ZLoop también permite saltos repetitivos. Sin embargo, el contador de saltos es de 16 bits, lo que quiere decir que se pueden hacer 65535 saltos. Los saltos deberán hacerse dentro de la misma página.

Una vez terminado el loop, cuando el contador haya llegado a cero, se ejecuta la siguiente instrucción.

Es lógico suponer que se pueden realizar loops dentro de otros loops. pero evitando cruces.

2.1.4.8.- Comandos de modo de operación.

Para definir el modo de operación del CY545 se utiliza el siguiente comando:

O Mode **Fija el modo de operación del CY545.**

El parámetro del comando MODE es almacenado en un registro interno de 8 bits del CY545. Se puede accesar a estos bits solamente a través del comando Mode, no a través de alguna línea externa de Entrada o Salida, tales como los User Bits. Cada bit del registro del comando Mode controla una característica del CY545, como se indica a continuación:

MB7 Selecciona el modo ASCII o binario.
MB6 Selecciona la velocidad de transmisión serial.
MB5 Habilita la señal CTS/ en USRB6 si se lo fija.
MB4 Envía al Display la posición durante los comandos JOG o HOME.
MB3 Reservado.
MB2 Selecciona el display HP si MB0=1
MB1 Selecciona el display CY325 si MB0=0
MB0 Selecciona un display paralelo o serial.

El comando Mode define los valores de los 8 bits del registro de una sola vez, razón por la cual se debe cuidar de no afectar el valor de una característica cuando se cambia el valor de otra. El valor default para este registro es 80H en el encendido. Esto selecciona el modo de comandos ASCII con un display serial standard.

MB7	MB6	MB5	MB4	MB3	MB2	MB1	MB0	
1	0	0	0	0	0	0	0	= 80h

El bit más significativo selecciona entre el modo de comandos ASCII o el modo de comandos binario. Cuando se fija este bit (1), se selecciona el modo ASCII, y cuando se lo pone en 0, se selecciona el modo binario. Cuando se usa el modo de comandos binarios, se debe enviar primero un comando O en el modo ASCII que ponga en 0 el bit más alto de este registro, de tal manera que el CY545 vaya al modo binario.

Cuando el MB5 del registro de modo en el CY545 es fijado(1), se colocará en el display la posición actual cuando el motor se está moviendo con el comando HOME o con la señal JOG.

Los tres bits más bajos son utilizados para seleccionar que dispositivo se usará como display. Se puede escoger dos tipos de displays: serial o paralelo. El bit menos significativo selecciona entre estas dos

opciones.

Cuando se selecciona un display serial, el bit MB1 permite seleccionar una salida serial standard, usada para conecciones con el Host o como un pórtico serial normal, o un display LCD, controlado por el controlador de display LCD CY-325.

Cuando el bit MB1 es 0, se usará un display serial standard. Cuando se selecciona un display paralelo con el bit MB0, el bit MB2 selecciona una salida paralela standard o un display especial de Leds Hewlett Packard.

MB7	MB6	MB5	MB4	MB3	MB2	MB1	MB0	
				x	x	0	0	Serial Standard
				x	x	1	0	Serial CY325
				x	0	x	1	Paralelo Standard
				x	1	x	1	Paralelo HP Led

Cuando MB2 es cero, la salida paralela standard es seleccionada. Entonces, se usa las señales IO_REQUEST y BUSY para controlar la lectura de información desde el CY545. También el CY-545 llevará a bajo la señal FPL (Compartida con el USRB6) cuando vaya a enviar alguna información. Este protocolo trabajará con un handshake standard de dos líneas o con el pórtico paralelo del controlador de redes CY233.

Cuando MB2 es 1, se selecciona el display HP LED. En este caso, todas las salidas del CY545 van a este display, y se genera una señal especial HP_SEL (Compartida con el USRB7), cuando el CY545 está escribiendo en el display.

Cuando se envia un comando Mode con MB5 en 1, el CY545 habilita la función CTS/ en el USRB6. Después de cualquier reset, esta función es deshabilitada, y USRB6 va a 1l, y la señal CTS/ queda deshabilitada también. Un comando Mode deberá enviarse al CY545 para habilitar nuevamente esta función.

Cuando esta señal está en bajo, el CY545 está listo para recibir más caracteres seriales, y cuando está en alto, el CY-545 está ocupado procesando un comando. La señal CTS/ puede ser usada para controlar automáticamente la velocidad de las comunicaciones seriales desde el host, deteniendo las transmisiones cuando esta señal está en 1 TTL. Esto permitirá al Host enviar una secuencia de comandos, incluyendo comandos de movimiento que pueden tomar tiempo en su ejecución, sin preocuparse en que si el CY545 será o no capaz de aceptar estos comandos. La única excepción se tiene cuando se usa la línea JOG.

Finalmente, si MB6 está en 1, el CY545 usa los cuatro bits más bajos para seleccionar la velocidad de transmisión serial. Esto permite que un comando paralelo fije la velocidad de transmisión serial al valor deseado, sin llevar las líneas I/O Request y Busy a un estado en particular. En esta función, el CY545 no almacena el byte de mode en un registro interno, y no se afectará ningún modo de operación, solamente la velocidad de transmisión. Los bits MB0 a MB3 son tratados como valores binarios, cuando seleccionan la velocidad de transmisión como se indica en la siguiente tabla:

MB3-MB0	a 11MHZ
0	300
1	600
2	1200
3	2400
4	4800
5	9600
6	19200
7	57600
8	VELOCIDAD ADAPTIVA (Hasta 19200 baudios)
9-F	INDEFINIDA

Cuando las frecuencias del cristal son diferentes a 11 MHz, las velocidades se escalan linealmente, mientras que la velocidad adaptiva intentará compensar la velocidad para este nuevo cristal. Para la mayoría de las frecuencias de cristal sobre 11 MHz, el CY545 puede adaptarse exitosamente a velocidades de transmisión de hasta 4800 baudios, con un pequeño error, que asegura una operación serial confiable. Sin embargo, a 14.7456 MHz el CY545 se adaptará a todas las velocidades hasta 19200 baudios, más dos velocidades no standares: 38400 y 76800 baudios.

Cuando la velocidad adaptiva es seleccionada, el CY545 debe recibir dos caracteres de retorno antes de que la velocidad de transmisión sea definida nuevamente, así se usa esta selección solamente si se van a recibir datos seriales en el CY-545. Ninguna transmisión ocurrirá hasta que la velocidad de transmisión sea definida nuevamente.

2.1.4.9.- Comandos Generales.

2.1.4.9.1.- Comando de inicialización.

Este comando permite resetear desde software el CY545:

I Inicializa el CY545.

Cuando el comando Initialize se procesa, actúa como un reset de encendido. El modo de comandos del CY545 se cambia al modo de comandos directo, todas las señales de entrada o salida van a alto, y la velocidad de transmisión serial es reseteada. Cualquier programa que se estaba ejecutando desde memoria externa se para si este comando se ha enviado.

Este comando fija los siguientes parámetros en el CY545:

R 100
F 14
S 220
N 200
y el registro de la posición actual es fijado en 0.

2.1.4.9.2.- Comandos de retraso de tiempo.

El CY545 dispone de una función de retraso de tiempo con el comando:

D Del16 Retraso de los milisegundos especificados.

El parámetro para este comando es de 16 bits, calibrado en milisegundos cuando el reloj corre a 12 MHz. Este comando simplemente causa que el CY545 espere por un tiempo especificado antes de ejecutar el siguiente comando.

Cuando se está trabajando con memoria externa, el comando permite hacer pausas por un tiempo determinado entre funciones del programa.

El valor del parámetro de Delay permite tiempos de retraso entre 1 milisegundo y 65 segundos cuando el CY545 corre con un reloj de 12 MHz. Se pueden producir retrasos más grandes con la repetición de comandos DELAY, o loops con comandos DELAY.

2.1.4.9.3.- Comando de solicitud.

2.1.4.9.3.1.- Comandos de Display de Status.

Junto con los diplays de cadenas alfanuméricas, el CY545 permite solicitar los valores del varios parámetros de trabajo. Esto se lo hace con el comando:

? Cmd

La parte "Cmd" del comando Query es una letra que permite seleccionar el parámetro que se colocará en el display. Las posibilidades son:

N	Número de pasos.
P	Valor actual de la posición.
R	Valor actual de la velocidad final.
F	Valor actual de la velocidad inicial.
S	Valor del parámetro de Slope.
Y	Valor del puntero de la dirección de memoria.
B	Valor actual de los user bits.

Cuando el comando Query se recibe o se lee desde memoria, el CY545 coloca en los displays los parámetros de la siguiente manera:

X=vvvvv<cr> por ejemplo: S=00220<cr>

Donde "X es la letra de parámetro, y "vvvvv" es el valor decimal ASCII de 5 dígitos para los parámetros. Los parámetros de solo 8 bits se colocarán 0s en los dígitos usados. Puesto que la Posición y el Número de pasos requieren de 24 bits, las salidas para estos parámetros serán con números de 8 dígitos, y no los 5 números standard.

En el modo binario, "vvvvv" se reemplaza por dos bytes que representan el valor binario del parámetro, debiendo enviarse primero el byte más significativo. No se debe enviar el caracter de retorno en el modo binario. La posición y el número se envían como valores de tres bytes.

2.1.4.9.3.2.- Obtención del contenido de la memoria.

Si se desea conocer el contenido de la memoria, se debe enviar un comando adicional con el comando "?". Este formato es especial, puesto que se incluye también un segundo parámetro, el cual es un contador.

Por ejemplo:

? M,5 <cr>

Para este comando, M indica que se requiere obtener información de la memoria externa, y el contador indica cuántas líneas se van a observar.

Cuando se conozca la información de la memoria, el CY545 enviará primero el valor del contador (Por Ejemplo: M=00001). Luego envía el contenido de la memoria. Cada caracter de retorno leído desde la memoria cuenta como una línea.

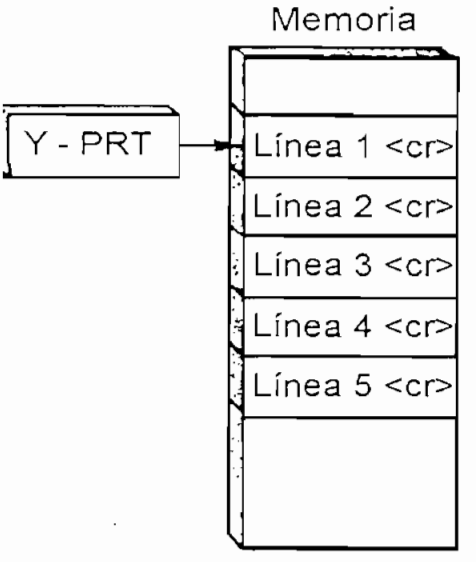


Figura 2.6

En el modo de comandos binario, también se puede leer la memoria, y el comando necesita dos parámetros, la "M" y el contador. Sin embargo, en este caso, el contador es el número de bytes que se van a leer desde la memoria, no el número de líneas.

2.1.4.9.3.3.- Comandos de display para mensajes.

El CY545 también dispone de un mecanismo para enviar mensajes al display que se haya definido. Esto lo hace el siguiente comando:

"String"

Normalmente, estos mensajes son parte de un programa en memoria externa, y pueden indicar algo en el display. Estos mensajes pueden ser manejados por un programador.

Este tipo de formato es bastante general. Puede colocarse cualquier caracter entre las comillas

(excepto otras comillas), incluyendo códigos de control ASCII y caracteres de retorno. El único caracter que determina el fin de un string es el caracter de las comillas (ASCII 22H), que también se usa para iniciar el String.

Los Strings son almacenados en memoria externa si es que se los recibe después de un comando ENTER; se graba cada parte del string incluyendo las comillas iniciales y finales. En el modo binario, el comando de String no incluye el contador de datos.

2.2.- FORMATOS DE SALIDA DEL PROGRAMA ORCAD.

El programa ORCAD es un paquete esquemático de captura. Este paquete permite crear, editar, grabar, imprimir y plotear diseños esquemáticos electrónicos.

Este programa está desarrollado para correr específicamente en computadores compatibles IBM y soporta la mayoría de los impresoras y plotters.

2.2.1.- Características del programa ORCAD/SDT III

El paquete de software OrCAD/SDT III consiste principalmente del programa de dibujo esquemático DRAFT, del Editor gráfico de objetos de librería llamado LIBEDIT, del programa Netlist, del programa de chequeo de diseño, del programa para listar las partes de un diseño, y algunos otros programas utilitarios y de librería.

2.2.1.1.- DRAFT.

Draft es un programa de dibujo de circuitos esquemáticos que permite crear, editar, y grabar hojas de diseño. Las principales características de DRAFT son:

- Dimensiones definibles por el usuario, con resolución de 1 mil.
- Tamaño de texto definible por el usuario.
- Ocho campos.
- Sobre 20.000 elementos de librería.
- Elementos equivalentes con DeMorgan.
- Se puede colocar uniones de alambre, buses, conectores, etiquetas y uniones.
- Cuando se mueven los objetos, las conecciones de alambre relacionados con ellos, se mueven igualmente en tiempo real.
- Rotación de partes y espejo.
- Movimiento, relocalización, y borrado de objetos y bloques de objetos.

- Comando para ejecución por pasos y repetición.
- Grilla visible de puntos.
- Cinco niveles de Zoom.
- Sobre 100 macros asignables por el usuario.
- Listado de parte de librería (Browse).
- Comando para simulación analógica.
- Búsqueda de Cadenas.
- Permite colocar texto verticalmente.
- Salida temporal para ejecución de comandos bajo DOS.
- Soporta hojas de trabajo de tamaños de A a la E, o se pueden se definidas por el usuario.

2.2.1.2.- Librería de Elementos.

Incluida con ORCAD/SDT III viene una librería con los elementos más comúnmente usados en la industria. Estas librerías incluyen elementos TTL, CMOS, memorias, ECL, discretos, análogos, microprocesadores, y dispositivos periféricos.

2.2.1.3.- Creación de elementos para la Librería.

El ORCAD/SDT III permite crear librerías propias de usuario, o modificar las existentes, de dos fáciles maneras.

Primero, se puede invocar al editor gráfico de objetos llamado LIBEDIT. Con este editor, se pueden usar comandos similares a aquellos del ORCAD/SDT III para construir o modificar un elemento en la pantalla y añadirlo a una nueva o ya existente librería. El editor de objetos LIBEDIT también permite crear fácilmente los símbolos standard IEEE/ANSI. LIBEDIT automáticamente convierte los símbolos gráficos a un formato leible por OrCAD/SDT III.

Segundo, se puede usar un editor de texto para crear un archivo de librerías fuente. Un archivo de librerías fuente es un archivo de texto ASCII que contiene instrucciones en el Lenguaje de Descripción de Símbolos de Orcad. Se puede correr el utilitario llamado COMPOSER en el archivo de librerías fuente para producir un archivo de librería de datos leíble por el ORCAD/SDT III.

2.2.1.4.- Programas Utilitarios.

El programa ORCAD incluye algunos programas utilitarios, tales como Annotate, Backanno, CleanUp, Composer, Crossref, Decomp, Erc, Libarch, Libedit, Netlist, Partlist, Plotall, Printall, Treelist, etc.

Se va a analizar ligeramente uno de los programa relacionados con este trabajo. Ellos son:

NETLIST: Este programa genera una lista de conecciones en un número de formatos seleccionables.
Dos de aquellos formatos son: Tango y Orcad. Estos son los formatos que interesan para este trabajo.
Los límites de Netlist los fija la memoria disponible en el sistema. Cuando Netlist corre, procesa cada archivo esquemático separadamente. El límite de 640 Kb de memoria del sistema, permite 8000 segmentos de alambre y alrededor de 2000 a 8000 pines de dispositivos por cada archivo esquemático. Esto incluye solamente aquellos objetos que están en una sola hoja de trabajo.

Información sobre otros comandos y programas se los puede obtener en el manual respectivo.

2.2.2.- Características del Programa ORCAD PCB.

Este programa, lanzado recientemente al mercado, permite realizar el diseño del circuito impreso, basado en el circuito esquemático generado por diferentes programas, entre ellos Orcad SDT III. Este programa tiene una presentación y modo de trabajo muy similar al programa ORCAD SDTIII, con características de manejo de pantalla similares. Permite trabajar con varia capas de cobre y manipular una amplia librería de partes. Adicionalmente, incluye formatos de salida para diferentes clases de photodrill.

Una característica importante de este programa es su capacidad para realizar un enrutamiento automático, basado en archivos Netlist. El programa tiene diferentes algoritmos para el enrutamiento, que producirán resultados diferentes. Por otra parte, permite realizar un enrutamiento manual, siendo posible variar las dimensiones de los caminos y de las vías.

Una parte importante de este programa, utilizado para nuestro trabajo, y que se va a describir detalladamente es el programa NCDRILL, que es el que produce el archivo de salida para el taladro.

2.2.3.- Formato de Salida del Programa ORCAD PCB.

Como se ha dicho, el programa que permite obtener un formato de salida para taladro, es el programa NCDRILL.EXE

La versión que se analizó es la 2.10, del 28 de Marzo de 1.990. Este programa tiene la capacidad para generar, en diferentes formatos, un listado de las coordenadas de cada uno de los agujeros que deberán perforarse. Es decir, genera las direcciones en las que se encuentran cada uno de los agujeros

del circuito impreso. Además, permite obtener información sobre el diámetro del agujero. El formato de este programa es el siguiente:

NCDRILL <ArchivoFuente><ArchivoDest.> <Tamañoherr.> [Switches]

En donde:

ArchivoFuente: es el nombre del archivo que contiene el diseño del circuito impreso.
ArchivoDestino: es el nombre del archivo que contendrá la información de salida.
TamañoHerr.: es el diámetro en pulgadas de la herramienta que se utilizará.
Switches: básicamente permiten escoger el formato de salida. Pueden ser:

/A Formato de salida ASCII
/B Permite extraer información acerca de las vías BLIND. Este comando debe ir seguido por un argumento de dos enteros (1 a 16) indicando las capas que se deben extraer.
/D Formato de Salida EXCELLON. Formato Decimal.
/L Formato de Salida EXCELLON. Formato Leading Zero.
/M Formato de Salida EXCELLON. Formato Métrico.
/O Fija offset X y Y desde el origen. Debe ir seguido de las coordenadas X y Y en pulgadas.
/U Extrae información acerca de las vías enterradas. Debe ir seguido por dos argumentos enteros (1 a 16) que indican las capas que se deben extraer.

En el caso de este trabajo, se deberá utilizar los switches para trabajar con archivos ASCII, que es el formato que reconoce el programa desarrollado, y que luego deberá ser convertido a un formato propio, también escrito en ASCII.

Un ejemplo de este formato, basado en el diseño de un circuito impreso con dos elementos (un circuito integrado de 24 pines y un conector de 4 salidas), es el siguiente:

COMANDO: NCDRILL EJEMPLO EJEMPLO.DRI 0.045 /A

```
ASCII(0.045,0.45,0.5)
ASCII(0.045,0.45,0.6)
ASCII(0.045,0.55,0.6)
ASCII(0.045,0.55,0.5)
ASCII(0.045,0.65,0.5)
ASCII(0.045,0.65,0.6)
ASCII(0.045,0.75,0.6)
ASCII(0.045,0.75,0.5)
ASCII(0.045,0.85,0.5)
ASCII(0.045,0.85,0.6)
ASCII(0.045,1.85,0.65)
ASCII(0.045,1.85,0.75)
```

```
ASCII(0.045,1.95,0.75)
ASCII(0.045,1.95,0.65)
ASCII(0.045,2.05,0.65)
ASCII(0.045,2.05,0.75)
ASCII(0.045,2.15,0.75)
ASCII(0.045,2.15,0.65)
ASCII(0.045,2.25,0.65)
ASCII(0.045,2.25,0.75)
ASCII(0.045,2.35,0.75)
ASCII(0.045,2.35,0.65)
ASCII(0.045,2.45,0.65)
ASCII(0.045,2.45,0.75)
ASCII(0.12,0.1,0.55)
ASCII(0.12,1.2,0.55)
ASCII(0.12,1.5,0.7)
ASCII(0.12,2.8,0.7)
```

Un análisis de este formato nos indica lo siguiente:

ASCII(DiámetroHerram., EjeX, EjeY)

En donde:

DiámetroHerram.	es el diámetro base de la broca que deberá utilizarse para el taladro. En caso de existir orificios de mayor diámetro, se indicarán.
EjeX	es la coordenada del eje X para este punto.
EjeY	es la coordenada del eje Y para este punto.

Como se puede observar, el circuito integrado del ejemplo origina una salida de 24 puntos, y el conector origina una salida de 4 puntos. Nótese que el diámetro de los orificios del conector es mayor que el diámetro de los orificios del circuito integrado. En total, se tiene información sobre 28 puntos, la misma que deberá ser analizada por el programa desarrollado, y convertida a un formato propio.

Deberá tomarse en cuenta que la palabra ASCII tiene un papel fundamental para el reconocimiento del tipo de formato en el programa desarrollado posteriormente. Esta palabra permite reconocer un archivo ASCII generado por el programa NCDRILL de ORCAD.

2.3.- Formatos de Salida del programa TANGO.

Accel, la empresa fabricante de Tango, ofrece dos opciones para el diseño de circuitos impresos: el programa Tango PCB y el programa Tango ROUTE.

2.3.1.- TANGO PCB.

El Programa TANGO PCB es una herramienta de ayuda en diseño esquemático de circuitos impresos, permitiendo el diseño físico de los mismos, incluyendo la colocación de los componentes.

Hay tres tipos de archivos que puede manejar este programa: archivos PCB, archivos de bloques, y archivos de photoplot. Los archivos de PCB son aquellos generados directamente por el programa, luego de haber sido esquematizados por el usuario. Los archivos de Bloques son porciones de una tarjeta, las cuales se definen para uso en el futuro. Los archivos Photoplot son archivos de formato Gerber creados por Tango, que luego son usados para generar el arte final en un fotoplotter.

Al igual que el programa OrCAD/PCB, este programa incluye librerías con elementos comunes, la cual puede ser incrementada por el usuario. Se incluyen elementos de montaje normal y elementos de montaje en superficie.

Este programa puede trabajar con la lista de componentes y uniones generada por ORCAD/SDT III con su programa NETLIST.EXE. Lógicamente, deberá seleccionarse la opción TANGO como formato de salida.

2.3.1.1.- Formatos de Salida del Programa TANGO PCB.

En el programa TANGO, se ha puesto especial atención a la generación de salidas de alta calidad a dispositivos como impresoras matriciales, o FotoPlotter de formato GERBER.

Para obtener un listado de orificios del circuito impreso que se va taladrar, se debe escoger inicialmente las aperturas y el tipo de herramientas que se utilizarán. Para ello, se deberá escoger la Opción OUTPUT en el programa Tango PCB, y seleccionar OUTPUT APERTURES. Dentro de esta opción se deberán asignar las herramientas a cada uno de los diferentes tipos de orificios que se tengan, dependiendo de los elementos que se han escogido utilizando la librería. Si no se ha asignado la apertura a alguno de los elementos que forman parte del circuito impreso, el programa no generará un archivo de salida DRILL.

Una vez asignadas las aperturas, se debe escoger la opción OUTPUT CAM. Dentro de ella, se seleccionará ASCII None como opción para el formato de salida. Es importante escoger este tipo de formato puesto que es el reconocido por el programa desarrollado para controlar el equipo. Luego, se debe escoger NCDRILL como salida y asignar un nombre y extensión al archivo que contendrá la información. Si se han escogido adecuadamente las aperturas, el programa generará un archivo: caso contrario, se visualizará un mensaje.

El archivo generado se analizará brevemente en base al siguiente ejemplo:

```
M48
METRIC,LZ,0000.00
T01F00S00
T01F00S00
T10F00S00
%
T01
X008889Y024764
X011429
T01
X01115Y011429
X012547Y008585
X013919Y011429
X015316Y008585
X016687Y011429
X018084Y008585
X019456Y011429
X020853Y008585
X022224Y011429
T10
X00419Y010007
X029184
M30
```

La segunda línea de este formato indica el tipo de unidades utilizadas. Pueden ser métricas o en pulgadas. La tercera, cuarta y quinta línea especifican el tipo de herramientas que se ha utilizado: esto quiere decir, los diferentes diámetros de los agujeros que se van a perforar.

Luego, se presentan los listados con las coordenadas de cada uno de los elementos que se van a perforar. Cada listado esta encabezado por el tipo de herramienta, y luego se colocan las dimensiones en diez milésimos de centímetro de cada uno de los agujeros. Así:

T01	Herramienta # 01
X01115Y011429	Punto X=0.1115 cm. Y=1.1429 cm.
X012547Y008585	Punto X=0.12547 cm. Y=0.8585 cm.

En el caso de tene puntos con dimensiones X o Y similares, el formato de salida será el siguiente:

T10	
X00419Y010007	Punto X=0.0419 cm. Y=1.0007 cm.
X029184	Punto X=2.9184 cm. Y=1.0007 cm.

T10	
X00419Y010007	Punto X=0.0419 cm. Y=1.0007 cm.
Y009184	Punto X=0.0419 cm. Y=0.9184 cm.

Esta información será analizada por el programa desarrollado, y se convertirá a un formato propio.

2.3.2.- Programa TANGO ROUTE.

Este sofisticado programa permite el ruteo automático en el diseño de circuitos impresos. El ruteo es el diseño de líneas que conectan a los componentes en el circuito PCB. Cuando se ejecuta este proceso manualmente puede tomar demasiado tiempo.

Iniciando con una archivo Net List de conecciones (usualmente la salida de un programa de captura esquemático), se podría usar TANGO PCB para definir el circuito de salida y colocar los componentes. Entonces, TANGO ROUTE usa el archivo PCB (con los componentes y la información de las conecciones de NetList) para rutear segmentos de línea en el circuito impreso.

Tango Route es un ruteador multi capas, lo que quiere decir que cualquier capa habilitada puede ser utilizada para el ruteado. El ruteado automático permite el ahorro de una gran cantidad de tiempo. El colocado regular y organizado de las conecciones de ruteo acelera la ejecución manual del circuito y las técnicas de chequeo del diseño aseguran el correcto ruteado del circuito.

Este programa está provisto de algoritmos de ruteo diseñados para altas complejidades. Se puede ejecutar este programa en cualquier computador IBM PC o compatible, y soporta la mayoría de dispositivos gráficos existentes, incluyendo EGA, VGA y otros adaptadores gráficos de alta resolución. Adicionalmente, soporta el manejo de hasta 32 Mb. de memoria extendida. En este paquete se incluyen las siguientes características:

- Trabaja directamente con los archivos de PCB-TANGO y TANGO-SCHEMATIC.
- Rutea circuitos de hasta 12 capas, conteniendo 10 capas de señales más los planos de tierra y Vcc.
- Incluye ruteos de línea ancho para redes Vcc y Tierra.
- Acomoda circuitos de hasta 32X32 pulgadas.
- Rutea sobre grillas de 25,20, 26.7, 12.5 y 10 mil. El ancho de la línea y el tamaño de la vía están basados en la grilla seleccionada.
- Ejecuta ruteos de 90 y 45 grados para alta densidad.
- Entrega un display gráfico del proceso de ruteado, con un reporte continuo del status actualizado.
- Permite que conecciones críticas sean manualmente preruteadas con el Tango PCB.
- Ejecuta chequeos eléctricos en las secciones preruteadas.
- Incluye algoritmos de ruteo de multipasada diseñados para obtener porcentajes altos de velocidad y

complejidad.

- Ejecuta colocaciones inteligentes de vías en porcentajes altos de complejidad. Soporta la colocación automática de vías para pads de superficie y conectores de flanco.
- Coloca las conecciones no rotueadas en la capa de conecciones y las lista en un file para que sean colocadas manualmente en el circuito.
- Lista estadísticas y mensajes de error desde el trabaja de ruteo.

2.4.- Formato de Programa diseñado.

Puesto que existía la necesidad de permitir procesos posteriores a la perforación de un circuito impreso, tales como la continuación de una perforación que ha sido interrumpida, la perforación a partir de un punto determinado, etc., y sabiendo que los formatos ASCII de los programas TANGO y ORCAD son diferentes (en forma no en fondo) y que contiene alguna información no útil para nuestro caso, se decidió crear un formato propio de trabajo para el programa desarrollado.

Cada archivo de peforación generado por los programas TANGO PCB y ORCAD PCB deberá ser convertido a este formato, utilizando el subprograma **Conversión de Archivos**, del programa principal. Este programa convertirá los archivos obteniendo de ellos información útil, y dejando de lado información que no le servirá.

A continuación, se presenta un ejemplo de este formato:

```
ORCAD
15
.045      1.143     1.27
.045      1.143     1.524
.045      1.397     1.524
.045      1.397     1.27
.045      1.651     1.27
.045      1.651     1.524
.045      2.159     1.524
.045      4.699     1.651
.045      4.699     1.905
.045      4.953     1.905
.045      4.953     1.651
.12       .254      1.397
.12       3.048     1.397
.12       3.81      1.778
.12       7.112     1.778
```

La primera parte del archivo contiene información que indica de donde de obtuvo este archivo. Pueden encontrarse dos palabras: ORCAD o TANGO, puesto que el programa solo reconoce estos dos tipos de formatos. Esta palabra servirá posteriormente para reconocer si este archivo ha sido generado por este programa.

La segunda parte contiene información sobre el número de puntos que contiene el archivo (o circuito impreso a perforarse).

A continuación se listan cada uno de los puntos que contiene el circuito. La información proporcionada es la siguiente:

0.045	1.143	1.27
Ancho (cm)	**Eje X (cm)**	**Eje Y (cm)**

Se listarán todos y cada uno de los puntos que forman parte del circuito impreso y que luego deberán ser peforados.

Capítulo III

Diseño del Sistema

CAPITULO III.- DISEÑO DEL SISTEMA

3.1.- Diseño del Sistema Mecánico.

La parte mecánica de este diseño deberá permitir que el taladro se mueva en tres ejes, un eje X, Y y Z. El diseño se ha basado en una arquitectura de procesamiento distribuida, lo que permite que el sistema se pueda mover independientemente en cada uno de los ejes. En base a los motores de pasos que se han utilizado, y al diseño mecánico del sistema, se ha podido lograr exactitudes de 0.0005".

3.1.1.- Diseño de los ejes X y Y.

El objetivo de este diseño mecánico fue lograr construir un sistema que permita que el taladro pueda ser desplazado en cualquiera de los tres ejes con una exactitud considerable. El diseño para los ejes X y Y es similar, diferenciándose el diseño del eje Z únicamente en su longitud (deberá ser de menor longitud que los otros dos ejes, pero sus partes son idénticas).

Un diagrama de bloque del sistema mecánico es el siguiente:

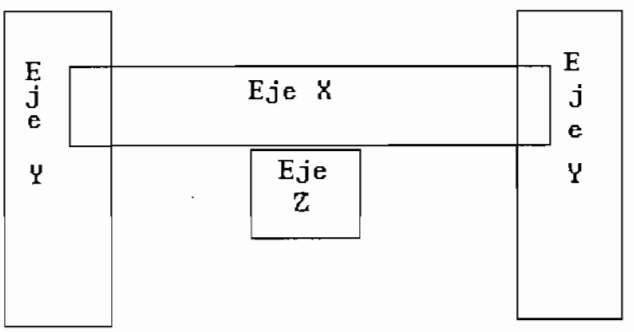


DIAGRAMA DE BLOQUES

Figura 3.1

Se va a detallar la construcción del sistema mecánico como si se lo estuviera montando; de esta manera se podrán detallar dos partes importantes a la vez, la una del diseño y la otra del montaje.

Cada uno de los ejes está montado en un canal C, de 61 cm. de largo, y 10.5 cm. por lado en el canal (Fig. 3.2 a). Este canal constituye el soporte principal para todas las partes del sistema. En la parte inicial del canal se montará el motor de paso, y en su parte final se montarán el bocín, que sostiene la varilla principal del sistema que tiene la forma de un extenso tornillo, y el soporte final que permitirá sostener las varillas lisas que proporcionan alineamiento y soporte al sistema.

En este trabajo se han utilizado motores de pasos US-CYBERLAB modelo 247. Las características eléctricas de estos motores se describirán posteriormente. Las características físicas de este motor se detallan en la figura 3.2 b. Para montar los motores en el canal se ha utilizado un soporte especialmente diseñado para ello (Fig. 3.2c). Este soporte será luego montado con el motor en el canal. Para sujetar el motor a este soporte, se han utilizado tornillos 6-32-1/2" y arandelas planas #6. La forma como se han montado los motores en el soporte se la puede observar en las figuras(3.2d y 3.2e).

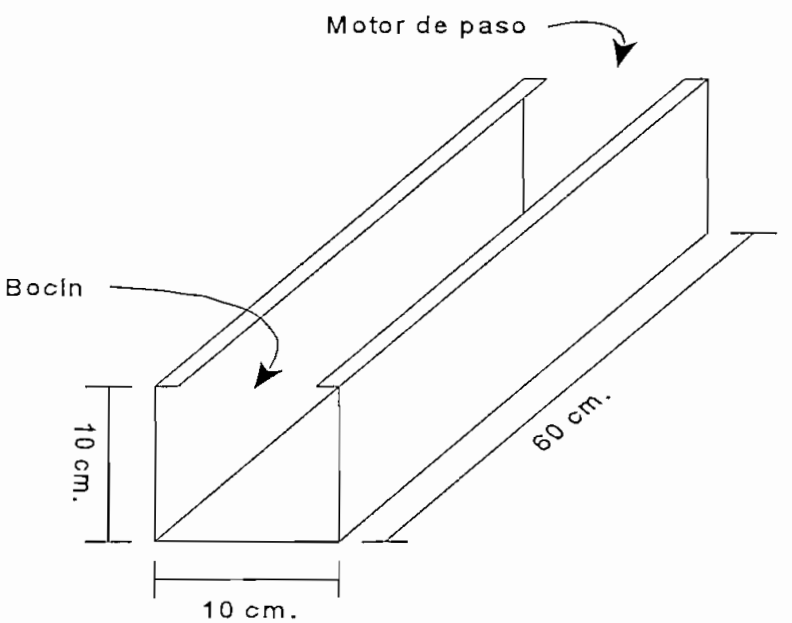
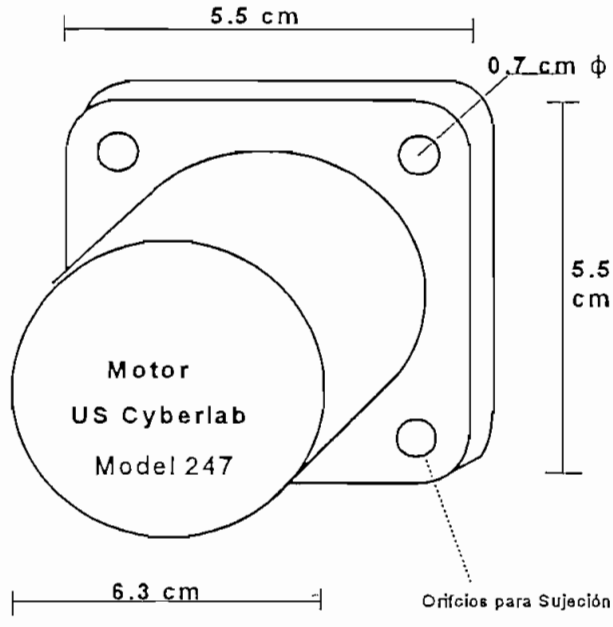


Fig. 3.2 a CANAL C


5.5 cm
0.7 cm ϕ
5.5
cm
Motor
US Cyberlab
Model 247
6.3 cm
Orifcios para Sujeción

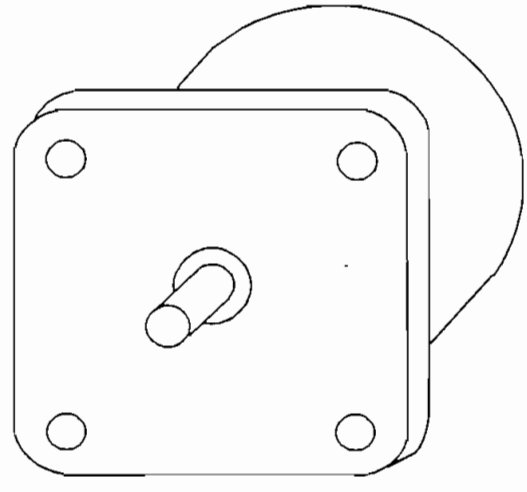


Figura 3.2b

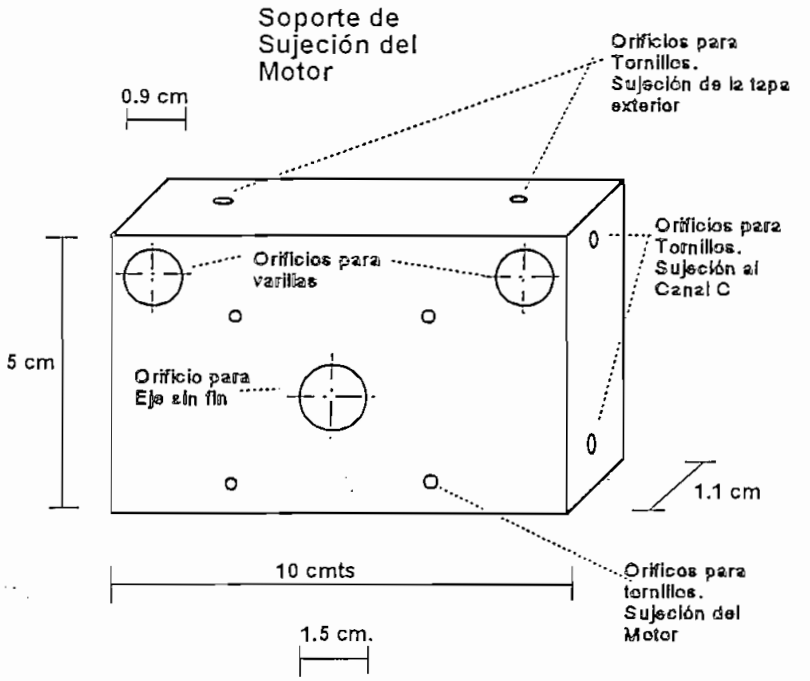


Figura 3.2c

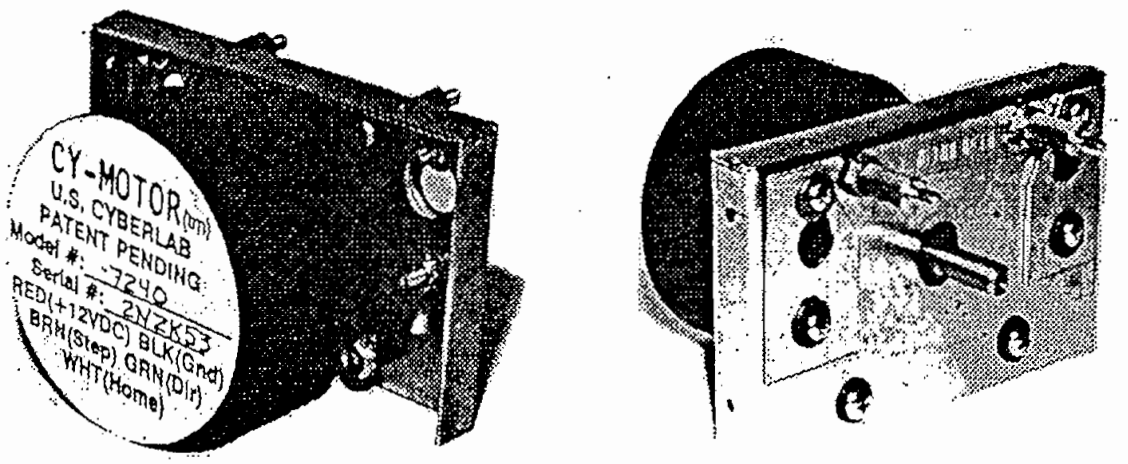


Figura 3.2d

Los tornillos se han montado desde el lado del motor, y se extienden a través de la estructura del mismo, siendo ajustados, por medio de tuercas, en el otro extremo. Inicialmente, no se ajustaron totalmente todos los tornillos en este diseño, con el objetivo de centrar posteriormente los ejes y permitir el libre desplazamiento de cada uno de los sistemas. Adicionalmente, se han montado en este soporte,

del lado de las tuercas, en dos de los tornillos como se puede observar en la figura 3.2d, dos postes que servirán posteriormente como soporte para colocar las tarjetas de efecto Hall, cuya utilidad se detallará más adelante.

Debe notarse que el soporte del motor dispone de dos agujeros sobre cada uno de los cuales se han montado, con tornillos, arandelas y tuercas, dos placas lisas (fig 3.2e) que también disponen de un agujero, a través de los cuales pasarán las varillas lisas. Estos agujeros tienen dos funciones, la una servir de soporte para las varillas lisas que pasarán a través de los mismos, y la otra, gracias a que los agujeros para los tornillos que las sujetan al soporte del motor tienen un mayor diámetro que los tornillos, permitir alinear y calibrar las posiciones de las varillas lisas, y de esta manera lograr que el sistema se deslice libremente.

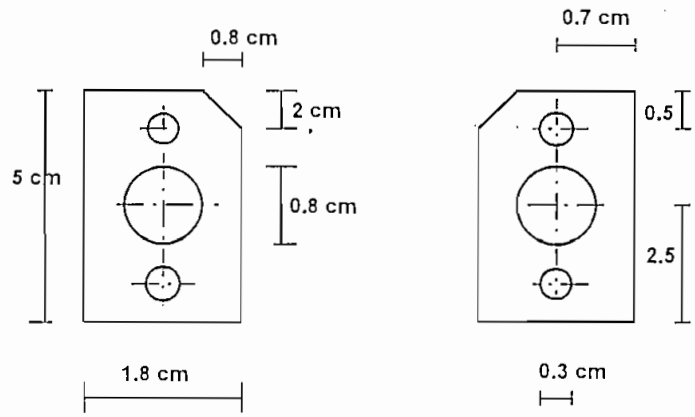


Figura 3.2e

Luego, se montó el motor y su soporte en el canal C. Para ello se usó tornillos #5-1/2". Igualmente, los agujeros para el montaje de este soporte en el canal C, tienen un diámetro mayor que el de los tornillos, para permitir ajustes posteriores en el proceso de alineamiento.

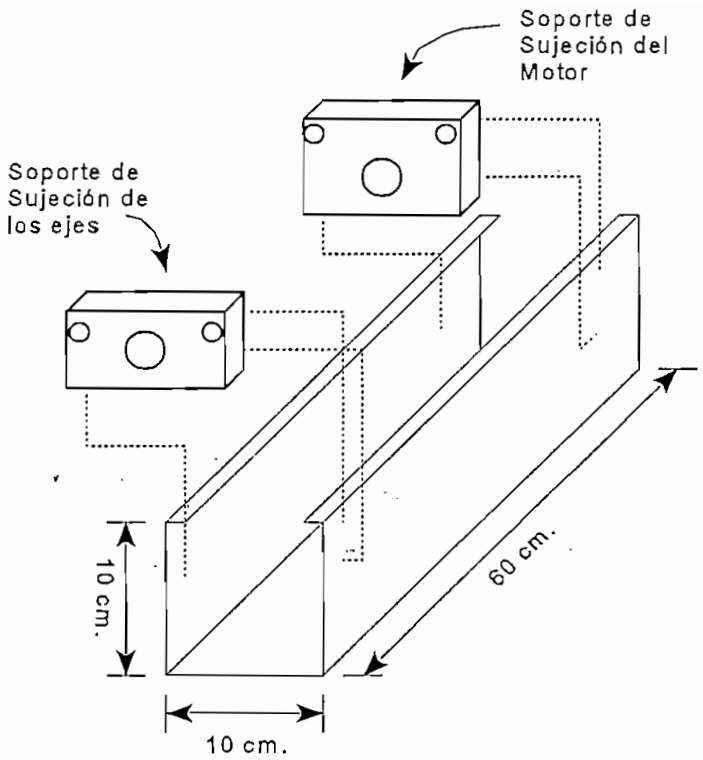


Figura 3.3a

Luego, se montó una lámina en forma de C, de 43 cm. de largo y 10.5 cm de ancho, en el interior del canal C. Esta lámina sirve de base para que el bloque deslizante se mueva a través del canal, dando un soporte adicional al bloque (Figura 3.3b)

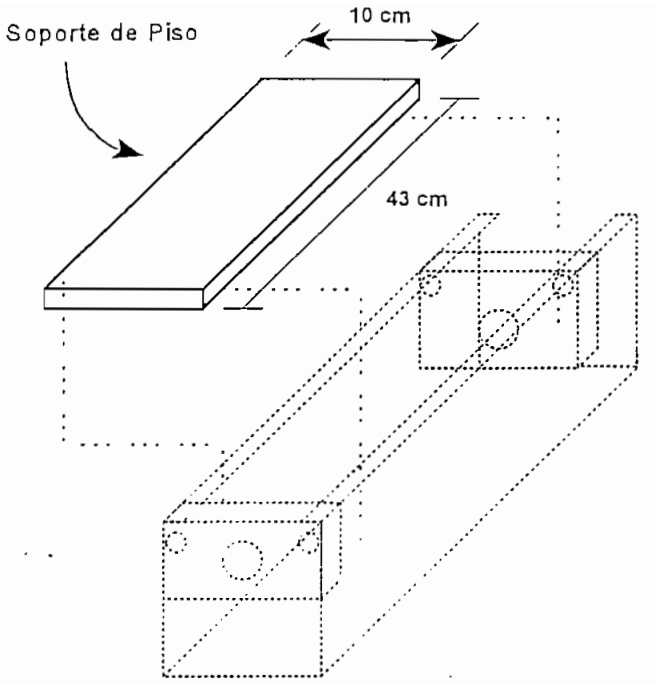


Figura 3.3b

Posteriormente, se colocó la varilla tipo ACME, que tiene la forma de tornillo, en el canal C. Se deslizó esta varilla a través del eje del motor, insertándola de tal manera que el final de la varilla estaba muy cercano al soporte del motor. Esta varilla dispone de un agujero pequeño en este extremo, a través del cual se colocó un tornillo tipo 4-40-1/8". Este pequeño tornillo permitió mantener fijo el eje del motor a la varilla.

Con el eje principal en su sitio, se procedió a colocar el bloque deslizante de color verde (Figura 3.3c), construido de un material de alto peso molecular, lo cual permite que se autolubrique; esta característica permite que el bloque se deslice suavemente a través de las dos varillas lisas. Se ha utilizado una prensa hidráulica para colocar una tuerca tipo ACME en el interior del agujero que tiene el bloque especialmente diseñado para ello. Además, el bloque tiene dos agujeros adicionales a través de los que pasarán las varillas lisas, como se puede observar en la figura 3.3c. A través de la tuerca tipo ACME se enroscó la varilla del mismo tipo, insertando de esta manera el bloque deslizante en el sistema.

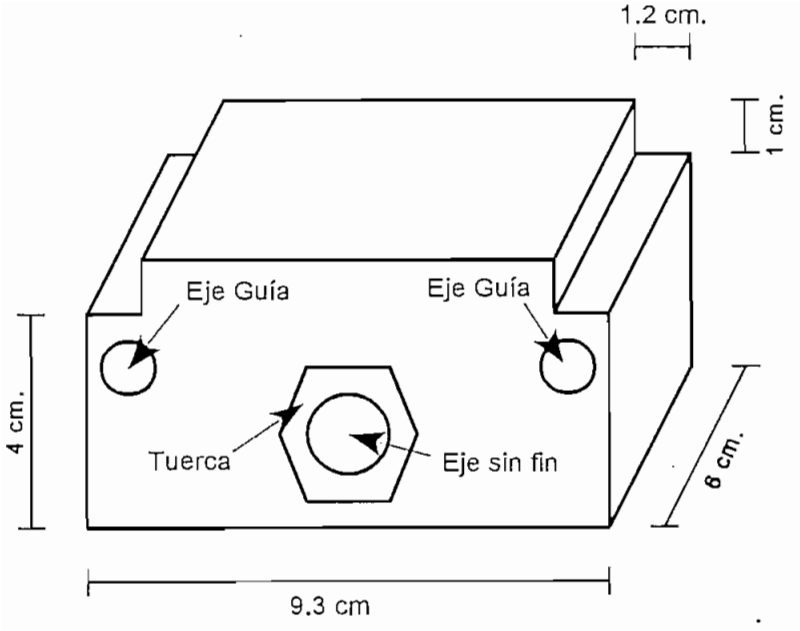


Figura 3.3c

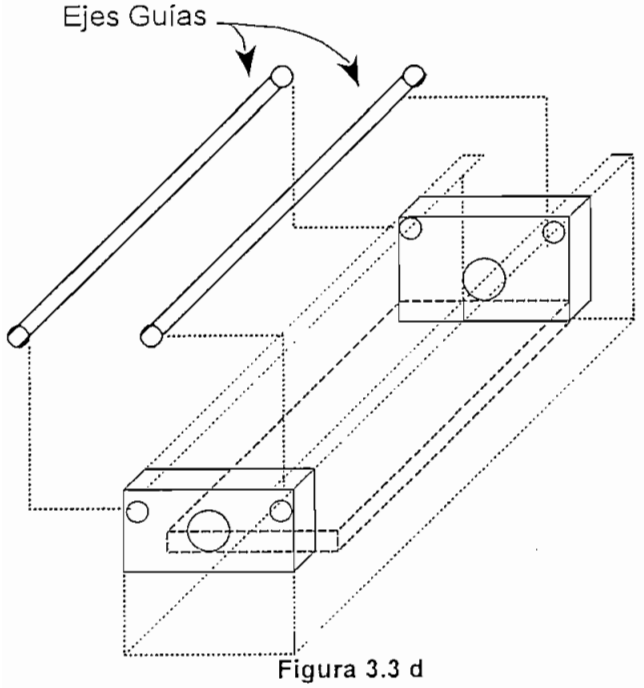


Figura 3.3 d

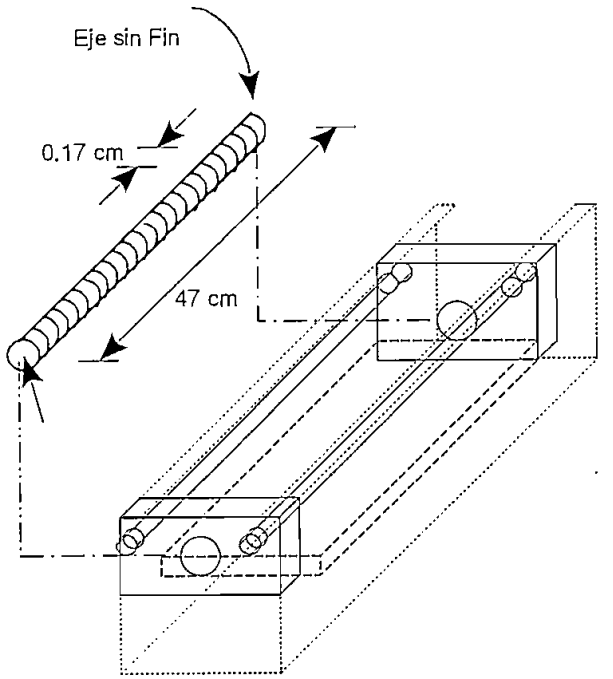


Figura 3.3 e

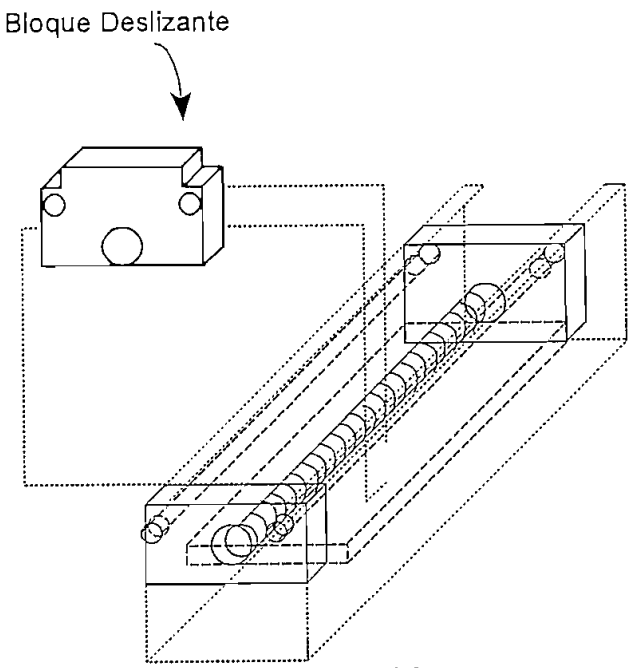


Figura 3.3 f

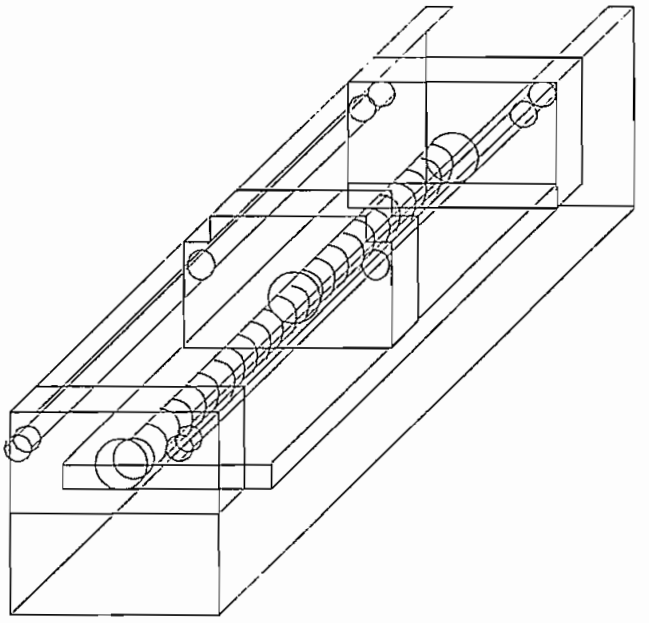


Figura 3.3 g

Se insertó este bloque algunos centímetros en esta varilla. Una vez hecho esto, se deslizó suavemente las varillas lisas a través de los agujeros provistos para ello en el bloque, y a través de los agujeros disponibles en el soporte del motor. El otro lado de las varillas se insertó en un soporte diseñado para ello, muy similar al soporte del motor; esto quiere decir que también dispone de las placas para el alineamiento de las varillas. A más de ello, este soporte dispone de un agujero en el centro del mismo, que sirvió para ajustar el bocín que sirve de soporte para el otro extremo de la varilla ACME. Entonces, se montó este soporte final en el canal C, usando tornillos 4-40-1/2".

Adicionalmente, se han montado dos tapas en cada uno de los extremos del canal. La una, tapará el motor, y la otra el final del canal. La tapa del motor, dispone de los orificios necesarios para colocar un conector tipo DIN, que permitió conectar luego, los cables correspondientes al motor. Véase las figuras 3.4a, 3.4b, en las que se pueden observar el sistema completo, fuera del canal C.

De esta manera, se han armado los sistemas móviles correspondientes a los ejes X y Y. Debe notarse que el eje Y dispone de dos sistemas similares.

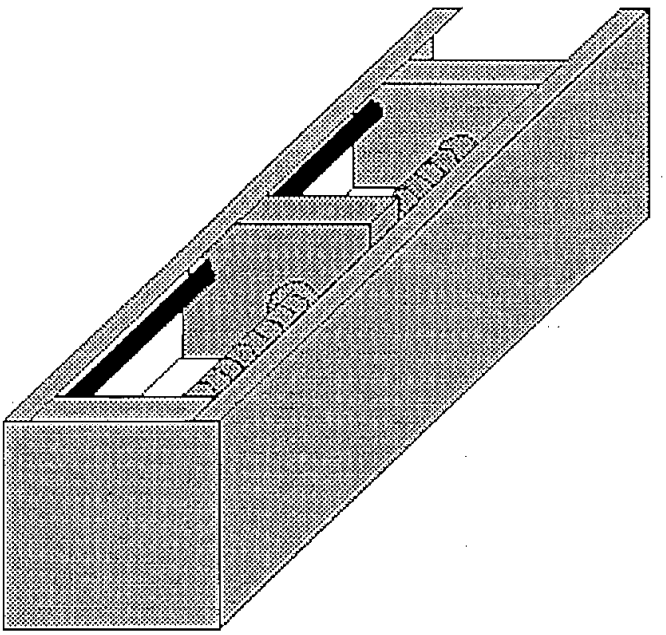


Figura 3.4 a

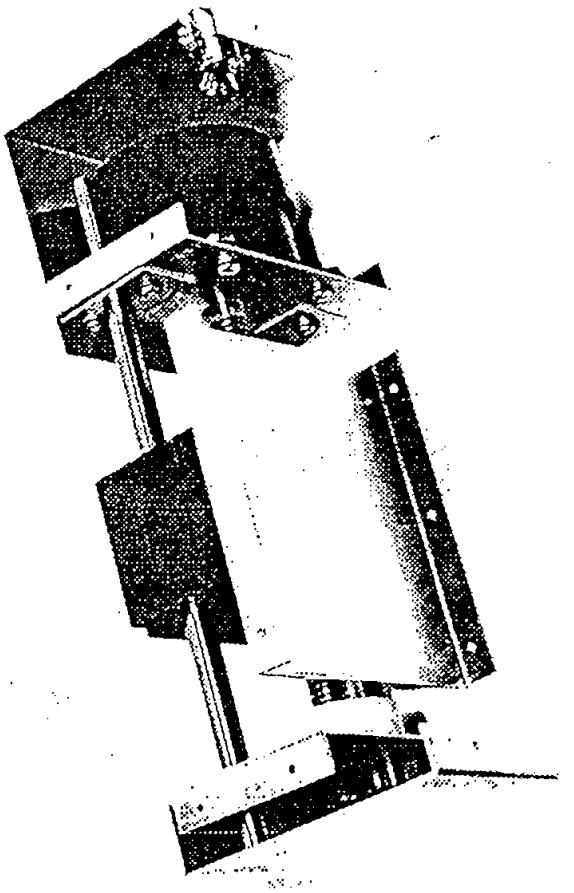


Figura 3.4b

3.1.2.- Diseño del sistema del Eje Z.

El sistema correspondiente al eje Z es similar al de los ejes X y Y. La única diferencia es que el canal C tiene una longitud de 28 cm. Este sistema mecánico se lo montó sobre el bloque deslizante del eje X, mediante el uso de cuatro tornillos número 6. El procedimiento para el montaje de este sistema fue similar al de los otros sistemas.

Este sistema, luego del montaje correspondiente, es el siguiente:

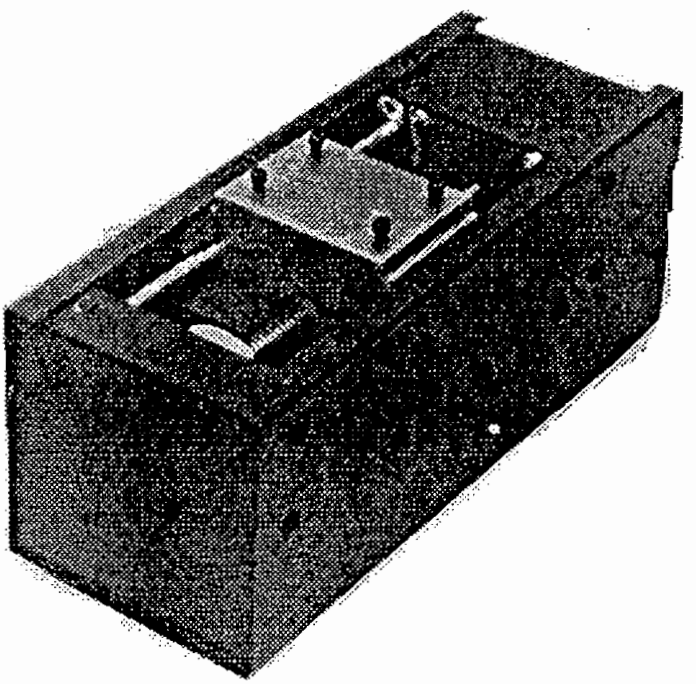


Figura 3.5

El eje Y es el eje que comanda al taladro; es decir, sobre este eje estará montado el taladro. Para ello, se utilizó el bloque deslizante de este sistema, al que se colocó un soporte para el taladro como el que se puede observar en la siguiente figura 3.6. Básicamente este soporte está formado por dos placas, cada una de las cuales dispone de dos bases de forma cilíndrica, las mismas que darán soporte al taladro.

Foto del Taladro

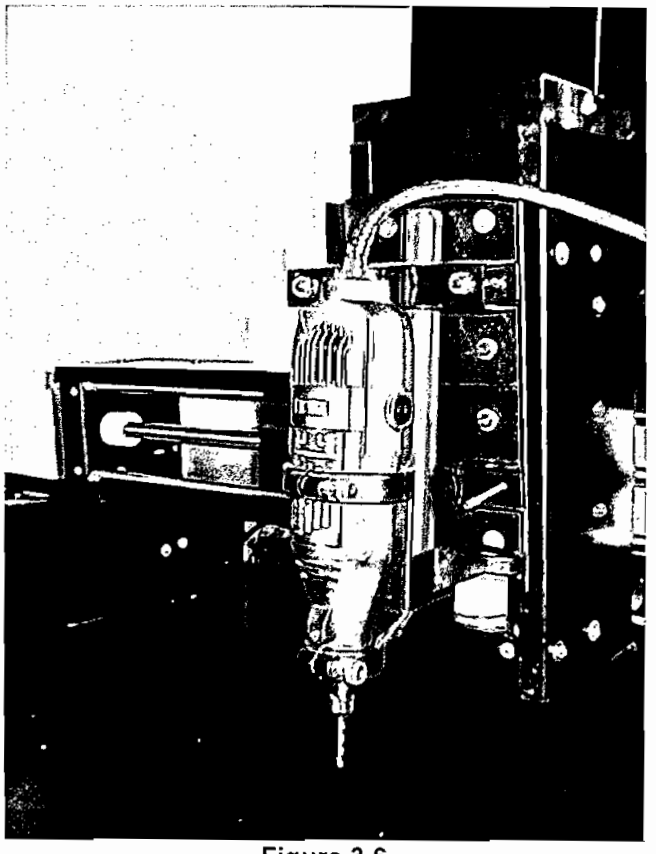


Figura 3.6

Debe notarse que este sistema está provisto de dos varillas colocadas a lo largo de la primera placa; en cada una de estas varillas se ha colocado un resorte, el mismo que evitará que el eje Z (taladro) baje excesivamente.

3.1.3.- Calibración de los sistemas.

Adicionalmente, se realizó la calibración de todos los sistemas (de todos los ejes), con el objetivo de lograr la mayor exactitud en el diseño. Para ello se utilizaron todos los tornillos provistos en cada uno de los mecanismos. De esta manera se logró un buen nivel de perpendicularidad entre cada uno de los ejes. Todos los sistemas se montaron sobre una base de aluminio, especialmente diseñada para este objetivo.

El proceso de alineamiento fue el siguiente:

Usando el programa, se movió el bloque deslizante de cada uno de los ejes, antes de montarlos definitivamente sobre la placa de aluminio, hacia el final del Canal C. Por medio del uso de una regla, se midió la altura de la parte superior del bloque deslizante con respecto a la superficie sobre la que está colocado el canal. Nuevamente, usando el programa, se llevó el bloque deslizante hacia el inicio del canal

C, cerca al motor de paso. Se midió la altura del bloque deslizante con respecto a la superficie sobre la que está asentado el canal C. Cuando se hizo estas dos mediciones, se usó el centro del bloque deslizante como referencia, puesto que como no se han ajustado definitivamente todos los tornillos del mecanismo, el bloque puede girar ligeramente. Teniendo estas dos medidas se debió lograr que las dos sean las mismas, lo que aseguró que el movimiento horizontal del bloque deslizante sea constante. Para ello, se calibraron los tornillos de orientación del motor y los tornillos de montaje del motor.

Posteriormente, se debió chequear la distancia de "lado a lado" del bloque deslizante. Para ello, desde la base del canal C, se trazó sobre la superficie de soporte, una línea paralela separada un centímetro del canal. Luego, se usó esta línea como base de referencia para una regla T. Con esta regla mantenida perpendicularmente, se usó otra regla para medir la distancia desde esta línea al borde del bloque deslizante. Usando el programa, se llevó el bloque al final del Canal C, y nuevamente se midió esta distancia y se la comparó con la obtenida en la otra posición. Estas distancias debían ser las mismas. Para lograrlo, se aflojó los tornillos de orientación del motor, y se ajustó la orientación, o se aflojaron los tornillos de montaje del motor, y se ajustó el motor. Se debió tener cuidado de no permitir que la varilla ACME suba o baje mientras se hace este ajuste, puesto que se perdería el ajuste anterior. Una vez realizado este ajuste, se chequeó nuevamente los dos ajustes realizados.

Luego, con el bloque deslizante fijado, se procedió a alinear las varillas guías. Primero, se ajustó un final de una de las varillas. Usando el programa, se llevó el bloque deslizante a lo largo del canal, mientras se observaba el lado opuesto de la varilla guía. Este lado subía o bajaba mientras el bloque se movía. Entonces, se ajustó cada final de la varilla, hacia arriba o hacia abajo, para producir que el bloque se nivele. Una vez que se logró este objetivo, se ajustó uno de los finales de la segunda varilla guía, y se repitió el proceso, alineando definitivamente las dos varillas.

Finalmente, se alineó la placa de soporte de bloque deslizante. Se debió ajustar la altura de esta placa de final a final, y de lado a lado, de tal manera que el bloque deslizante toque exactamente a la superficie de la placa. Para ello se utilizó los tornillos de soporte de la placa. Una vez que se logró, la presión externa sobre la placa quedó distribuida homogéneamente a lo largo de ella, minimizando la deflección de las varillas guías.

Utilizando similar procedimiento se alineó cada uno de los ejes, incluyendo el eje Z.

3.1.4.- Montaje del Sistema Completo.

Inicialmente, se montó el primer eje Y sobre la placa de aluminio, utilizando tornillos de metal. Asegurados los dos finales del eje, se estableció una base de referencia. Usando una regla T, se trazó dos líneas perpendiculares a este eje, desde el inicio y desde el final del eje. Se colocó el segundo eje Y entre estas líneas, en paralelo con el primer eje. Después, se fijó el eje X sobre los bloques deslizantes

de estos dos ejes. Se lo aseguró en uno de los ejes Y, usando tornillos 6-32, pero sin ajustarlos totalmente. Así, usando una regla T, se aseguró que todo está en escuadra alrededor de la placa de aluminio. Una vez hecho esto, se aseguró los tornillos del otro eje Y.

Luego, usando el programa, se movió los motores asegurando que los dos ejes hayan quedado perpendiculares. Con el eje X y el eje Y calibrados, se usó la regla T para chequear la verticalidad del frente a atrás y de lado a lado. Entonces se procedió a montar el eje Z; se lo calibró usando los tornillos correspondientes, de tal manera que se logre la mayor exactitud posible, manteniendo en escuadra los tres ejes.

Por último, se montó el taladro DREMEL en el bloque deslizante del eje Z. Para ello, se utilizó cuatro tornillos 6-32. Se usó el programa para cuequear y fijar el control de altura del taladro.

3.1.5.- Panel de Control.

Adicionalmente, la placa de aluminio sirvió también como base para la caja que contiene el panel de control, las tarjetas electrónicas y las fuentes. Una vista del diseño de esta placa es la siguiente:

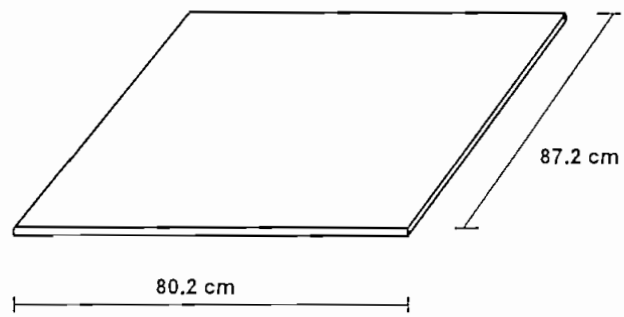


Figura 3.7

Debe notarse que la caja completa está formada en realidad por dos cajas más pequeñas. La primera, contiene en su interior las dos tarjetas electrónicas principales que contienen los microcontroladores usados en este trabajo. La segunda caja contiene las fuentes y los circuitos de potencia para los motores de paso. Los circuitos de las dos cajas están intercomunicados a través de un cable paralelo. Un análisis de las señales que se transmiten a través de este cable se lo puede observar

posteriormente en los diseños electrónicos.

La primera caja tiene la siguiente forma:

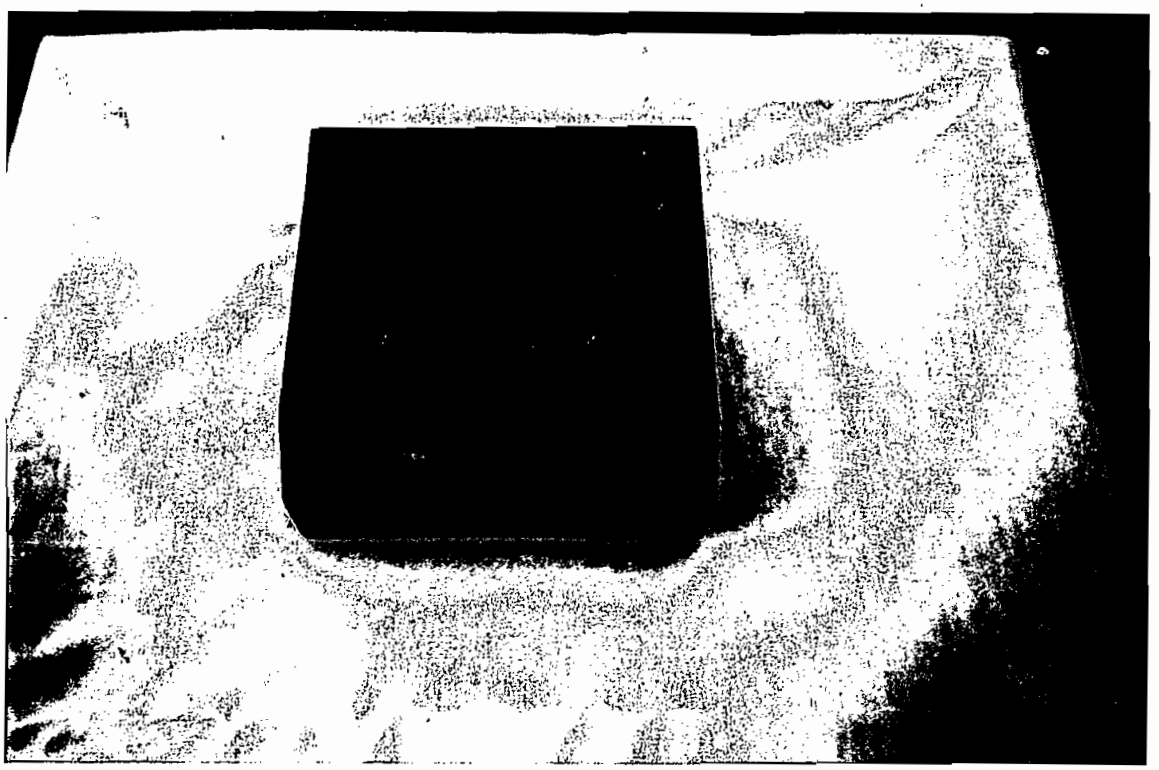


Fig. 3.8

Nótese que esta caja permite observar a través de LED's los niveles de las principales señales que maneja el CY545. Además, en esta caja se encuentran los switches que permitirán mover manualmente los motores de pasos, y los que permitirán resetear los microcontroladores CY545. Adicionalmente, en esta caja se encuentran los conectores que permitirán al equipo conectarse serialmente con el computador.

La segunda caja es de la siguiente forma:

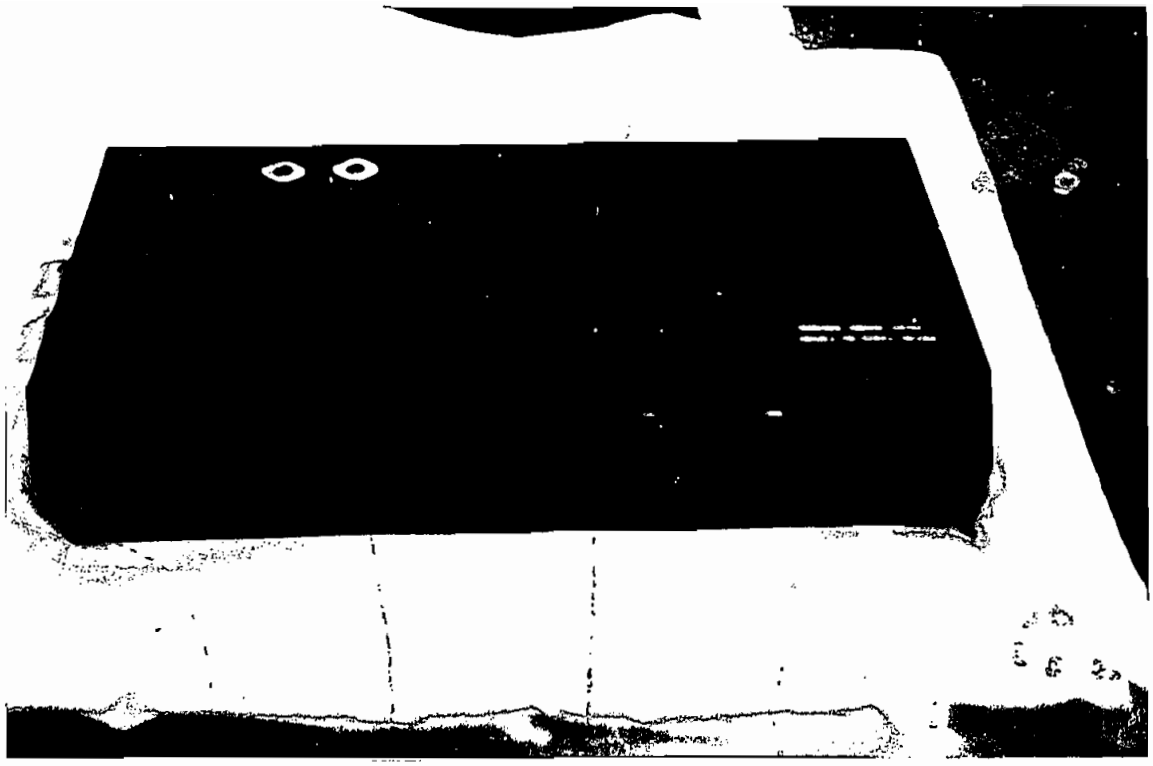


Fig. 3.9

Esta caja contiene las fuentes de poder y el sistema de potencia que controla los motores de paso. El panel de esta caja contiene los LED's de encendido de las fuentes, y los conectores tipo DIN que van a cada uno de los motores. Además contiene el interruptor de encendido y los portafusibles.

3.2.- *Diseño* del Sistema Electrónico de Control de los motores de pasos y el Taladro.

El Diseño del Sistema Electrónico de Control de los motores de pasos y del taladro se basa en el Microprocesador CY-545. Una descripción detallada de este micro-procesador se puede encontrar en el capítulo Dos.

Como en este trabajo se van a controlar, por medio de pasos, básicamente dos movimientos: uno en el eje X y otro en el eje Y, se ha desarrollado dos sistemas de control similares, que al final se diferenciarán por un número de identificación (dirección). Por lo tanto, se van a utilizar dos microcontroladores CY545 trabajando en paralelo, uno para cada eje.

En el siguiente diagrama, se puede observar un detalle de los componentes de este sistema de control:

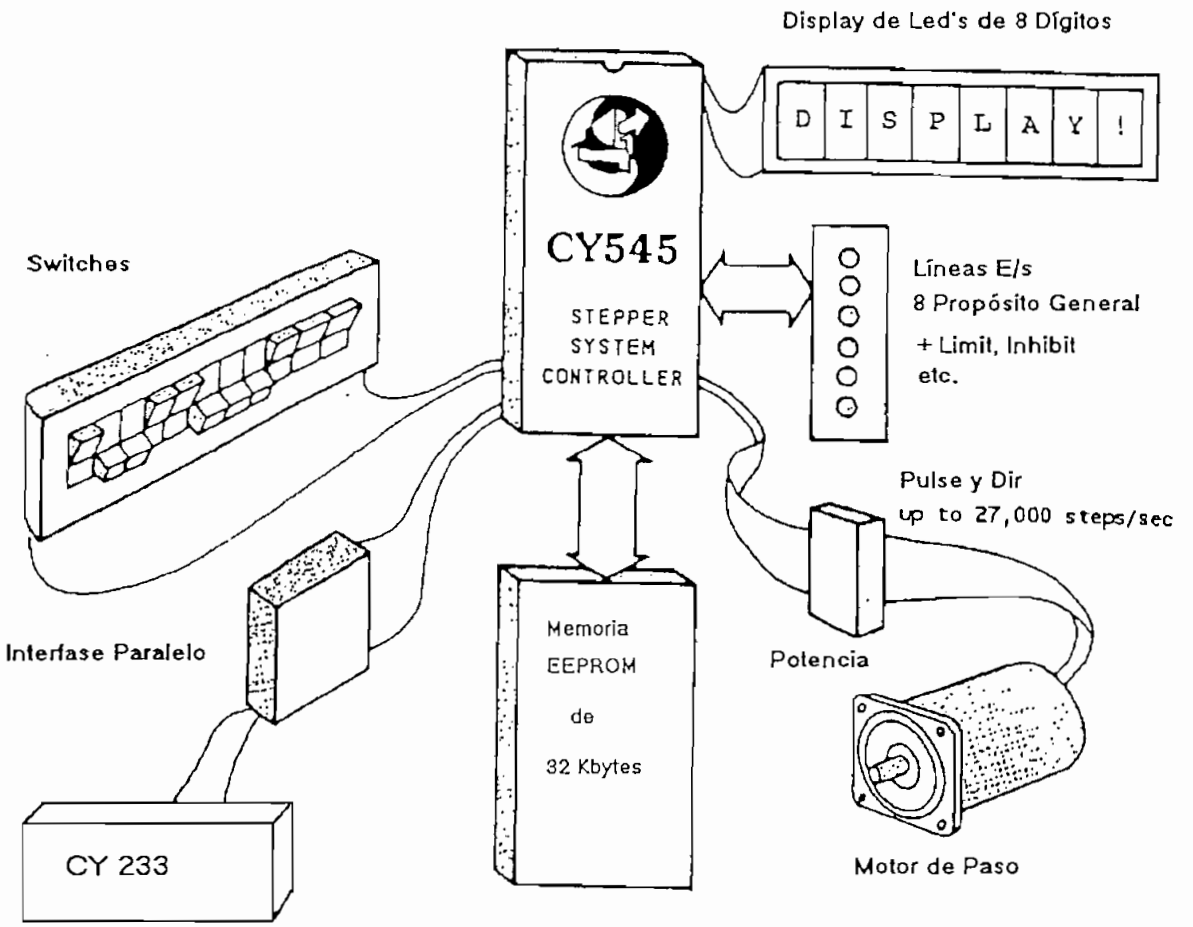


Figura 3.10

A continuación se presentarán los diseños de las diferentes partes del sistema. En caso de existir alguna diferencia entre el diseño del eje X o del eje Y se indicará. En caso de no existir ninguna diferencia, se asumirá que el diseño es similar para los dos ejes.

3.2.1.- Parámetros del CY545.

Cada uno de los CY545 va a comunicarse con el dispositivo que manejará la comunicación serial con el computador, a través del interfase paralelo. Por lo tanto, las señales de control del CY545 para el manejo de este interfase deberán conectarse a este dispositivo. Estas señales son : BUSY/ (Pin 15), IO_REQUEST/ (Pin 13), UB6_FPL/ (Pin 27), y por supuesto el bus de datos (Pin 32 al 39).

Como se va a utilizar un display HP, se utilizará el User Bit 7, UB7_HP_SEL (Pin 28) para seleccionar esta opción cuando sea necesario.

Las señales RxD (Pin 10) y TxD (Pin 11) quedarán sin conección puesto que no se va utilizar el interfase serial del CY-545.

La señal de RESET (Pin 9), se la conectará al circuito diseñado para este objetivo y que se lo analizará posteriormente.

La señal SW_SEL/ (Pin 12) tampoco se utilizará, puesto que no se van a utilizar Thumbwheel Switches para fijar los parámetros de trabajo de los CY545.

La señal XMEM_SEL/ (Pin 14) se utilizará para el manejo de la memoria EEPROM. Para ello, también se utilizará las señales WR/ (Pin 16), RD/ (Pin 17) y ALE (Pin 30). Estas señales también estarán involucradas en el manejo del display HP.

Un análisis del uso de los User Bits se lo hará posteriormente.

La señal XTAL1 (Pin 19) y XTAL2 (Pin 18) se utilizarán para el diseño del circuito de reloj que se lo analizará más tarde.

Las señales de control de los motores de pasos se van a analizar en una sección adicional.

3.2.2.- Circuitos de Reloj y Reset para el CY545.

El CY-545 es la pieza básica de este diseño. Para su funcionamiento requiere de un circuito de cristal, el cual dará el soporte de tiempo para este dispositivo. El valor default para reloj es de 11 MHz, el cual permite al CY545 generar todas las velocidades seriales de transmisión standard; en este caso se ha utilizado un cristal de 16 MHz(x2), puesto que como no se va utilizar el interfase serial, conviene tener la mayor velocidad interna de proceso que se pueda obtener en el CY-545. También forman parte de este circuito de reloj dos condensadores de 33pF (c15 y c16). Estos tres elementos constituyen el sistema básico de reloj para este microprocesador. Es necesario indicar que el CY545 también puede ser operado por un circuito externo de reloj (que no se ha utilizado en el caso de este diseño); en este caso, deberá conectarse la señal externa de reloj al pin 19 (XTAL 1) a través de un buffer CMOS, y el pin 18 (XTAL 2) dejarlo sin conección.

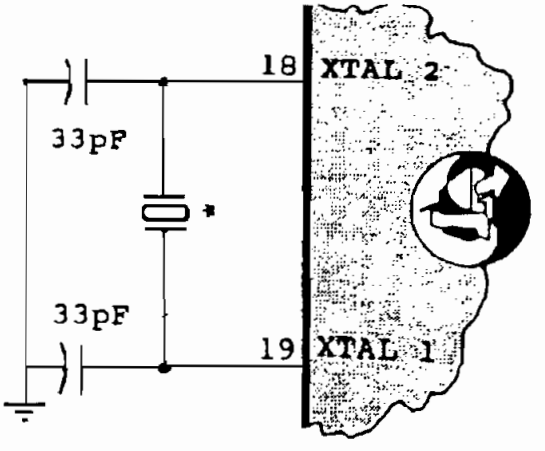


Figura 3.11

Adicionalmente, se ha diseñado un circuito de reset para este microprocesador, tomando en cuenta recomendaciones del fabricante. La línea RESTART (pin #9) debe ser mantenida en alto para que se produzca el reseteo del CY545; esta línea debe ser mantenida en alto por 10mseg desde que se energizó nuevamente al CY545, para que se estabilice. En este circuito se han utilizado un condensador de 4.7uF(c11) y una resistencia de 10K(R1). Además, se ha utilizado un microswitch en paralelo con un switch normal para producir el reseteo. El microswitch se lo utilizará internamente en la tarjeta principal, y el switch normal se lo utilizará en el panel frontal del equipo, para que el usuario u operador pueda resetear el equipo externamente.

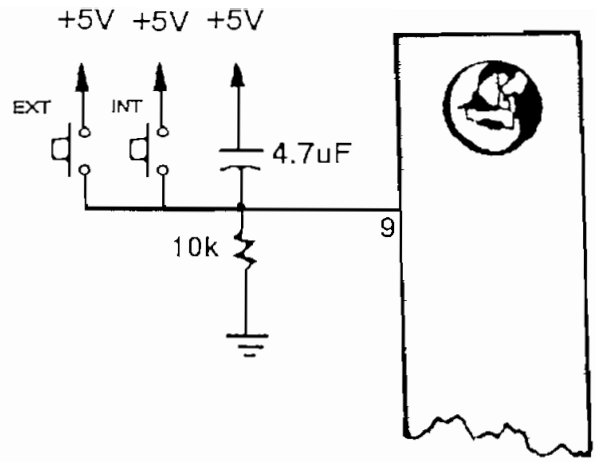


Figura 3.12

Como se pudo observar en el capítulo Dos, este chip dispone de varias señales de control para el motor de pasos, para determinar los límites, y algunas para control adicional; estas señales deben ser enviadas a un buffer de corriente para que puedan ser conectadas a algún otro circuito integrado TTL, puesto que la tecnología de construcción del microcontrolador CY-545 es CMOS.

Por otra parte, se ha implementado un sistema de control por LEDs para los UserBits (bits definibles por el usuario) y para las señales de control Pulse, Dir, Stop y Slew. Estos Leds permitirán observar el estado del microprocesador. Para ello. se ha utilizado los microLeds HLMP-6600 que pueden ser conectados directamente a +5V por un extremo, y por el otro a un Buffer de corriente (74LS244); el circuito integrado 74LS244 no es un inversor. Cuando estos pines tengan como salida un 0L, los leds se encenderán; cuando tengan como salida un 1l, los leds se apagarán.

En el caso de los UserBits (USRB0 a USRB7), se han conectado directamente al Circuito integrado 74LS244 (U11), con el objetivo de poder sensar sus señales visualmente. Adicionalmente, se han conectado estas señales (USRB0 a USRB7) a un juego de dip-switchs cuyo extremo está conectado a tierra. El objetivo de este dip-switch es llevar externamente los User-bits a un nivel lógico alto o bajo, dependiendo de la posición. Es necesario recordar que los User Bits tienen el caracter de entrada y de salida; por lo tanto, con estos dipswitchs se podrá controlar manualmente el nivel lógico a la entrada de los Users Bits. Se debe indicar también que a pesar de que el circuito integrado 74LS244 es de tres estados, en este caso se lo mantiene activado todo el tiempo llevando a tierra los pines 1 y 19 (1G y 2G).

En el caso de las señales Pulse, Dir, Stop y Slew, se han conectado al circuito integrado 74LS244 (U9), y a través de este circuito a los LEDs HLMP-6600. De esta manera se podrá sensar el nivel lógico de estas señales visualmente. Al igual que en el caso de los UserBits, un led encendido indicará un nivel lógico bajo y un led apagado indicará un nivel alto.

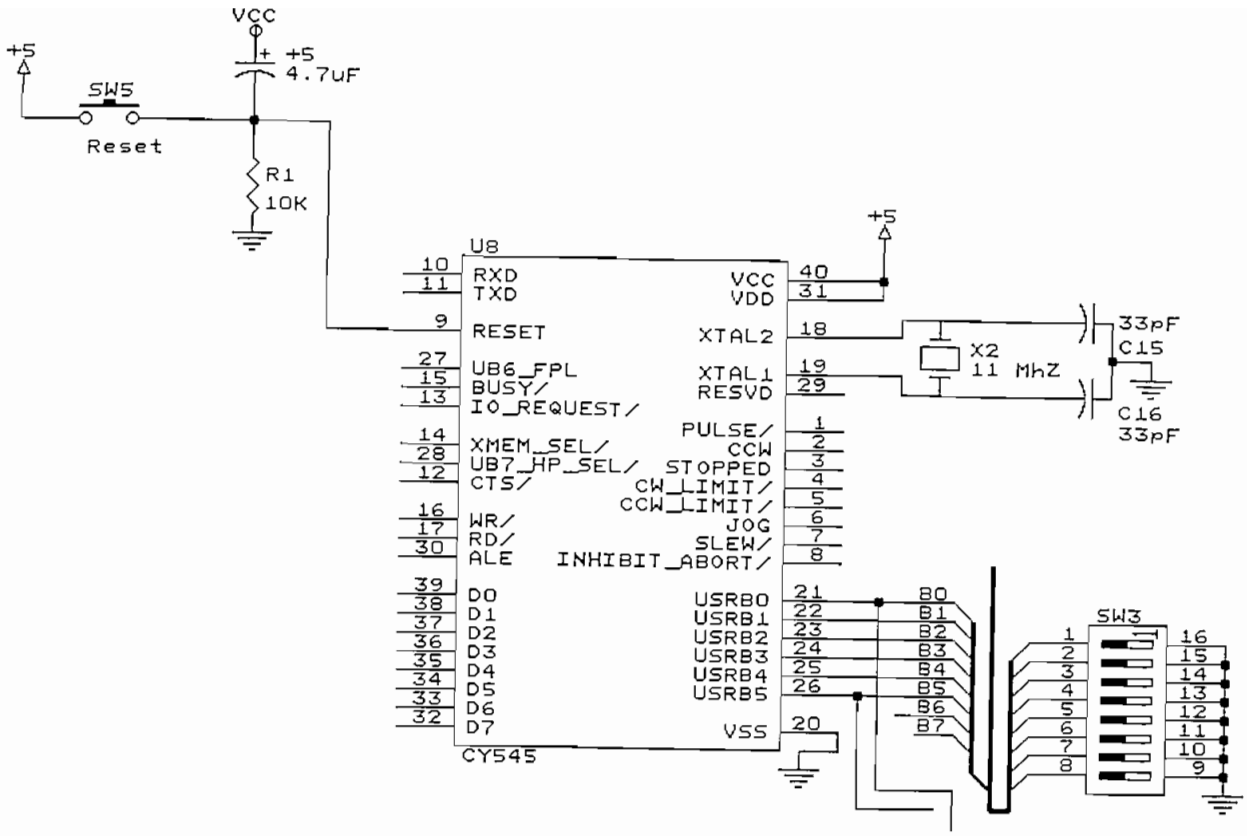


Figura 3.13

Todos estos diseños son válidos tanto para el eje X como para el eje Y.

3.2.3.- Interfase con la Memoria.

El interfase de memoria del CY545 está diseñado para aceptar una variedad de dispositivos de memoria, incluyendo SRAM, EPROM, y EEPROM. En el caso de memoria EPROM, estas deben ser programadas separadamente.

El CY545 puede ser operado con hasta 64Kb de memoria externa, la cual puede mantener varios comandos para que luego sean ejecutados. Como se pudo observar en el capítulo Dos, algunos comandos del CY545 se pueden usar para manejar la memoria, permitiendo direccionar el puntero de memoria, ingresar comandos en la memoria, y ejecutar comandos desde la memoria.

En el caso de esta aplicación se ha utilizado una memoria EEPROM (XICOR X-2864AP 28 pines), la cual puede ser escrita, leída y borrada directamente desde el CY545. El contenido de la memoria EEPROM no desaparece cuando se retira la energía del circuito.

El CY545 dispone de todas las señales de strobe y de direcciones para la memoria. Estas señales son el bus de datos, XMEM_SEL, ALE, WR, y RD. El bus de datos tiene 8 bits, y las direcciones son de hasta 16 bits (64Kb); por lo tanto, el bus de datos de 8 bits deberá manejar estas 16 direcciones de memoria, más los datos que deberán ser entregados por la memoria y conducidos al CY545, o viceversa.

Para mantener estas direcciones, se necesitan dos Latchs externos. Se han utilizado los latchs 74HCT373(U4 y U2) en este caso. Cuando el CY545 lee o escribe a la memoria externa, inicialmente coloca el byte más significativo de direcciones en el bus de datos. Entonces se selecciona la memoria llevando a un nivel lógico bajo la señal XMEM_SLEW. Entonces, el byte direcciones más altas es retenido en el 74LS373 que mantiene las líneas de direcciones A8 a A15. Entonces el CY545 ejecuta una operación de lectura y de escritura (si se desea leer o escribir).

La lectura o escritura se divide en dos pasos. Primero se genera el byte de direcciones menos significativo en el bus de datos. Este valor es mantenido por el otro latch 74LS373, usando la señal ALE del CY545. Entonces, se produce la transferencia de datos. Para una escritura, el CY545 genera los datos a escribirse en el bus de datos, junto con el strobe de escritura, WR. Para una lectura, el bus de datos se mantiene flotante, y un strobe de lectura es generado, usando RD, así la memoria toma control del bus.

Cuando se acaba la transferencia de la memoria, la señal XMEM_SEL va una nivel alto nuevamente, deseleccionando la memoria externa. Esto permite usar el byte menos significativo de direcciones y los strobes RD y WR con otros dispositivos, sin que la memoria interfiera. Unicamente la memoria utiliza el byte más significativo. Una forma típica de onda se muestra a continuación:

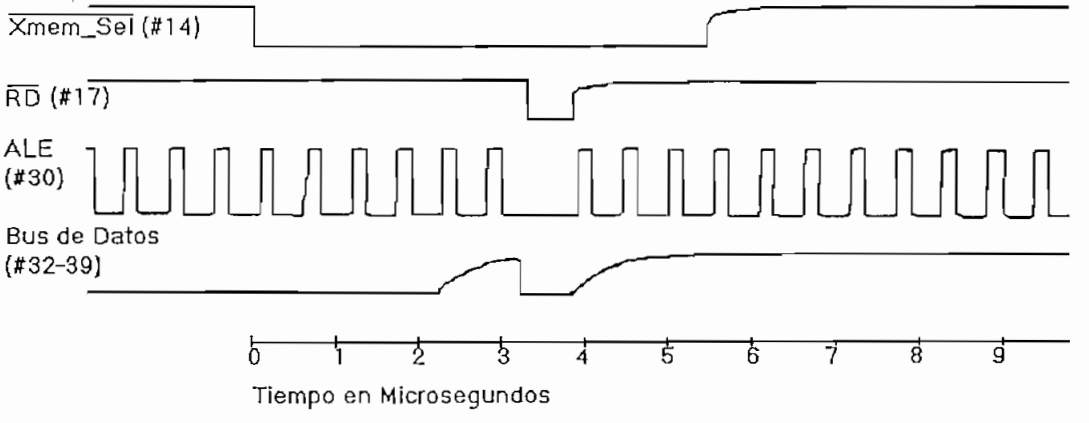


Figura 3.14

Como una característica especial, el CY545 ejecuta una verificación de lectura después de escribir en la memoria. Se ha incorporado esta opción, para dar soporte a las memorias EEPROM que requieren de un tiempo más largo para ejecutar una operación de lectura, y que no pueden ejecutar otra lectura o escritura randómica mientras la primera está en proceso. Como se ha utilizado el interfase paralelo para la comunicación con el Host, se ha evitado este problema, puesto que este interfase controla los tiempos de retraso manteniendo ocupado al CY-545 mientras se está escribiendo en la memoria. Esta característica se muestra en la siguiente forma de onda:

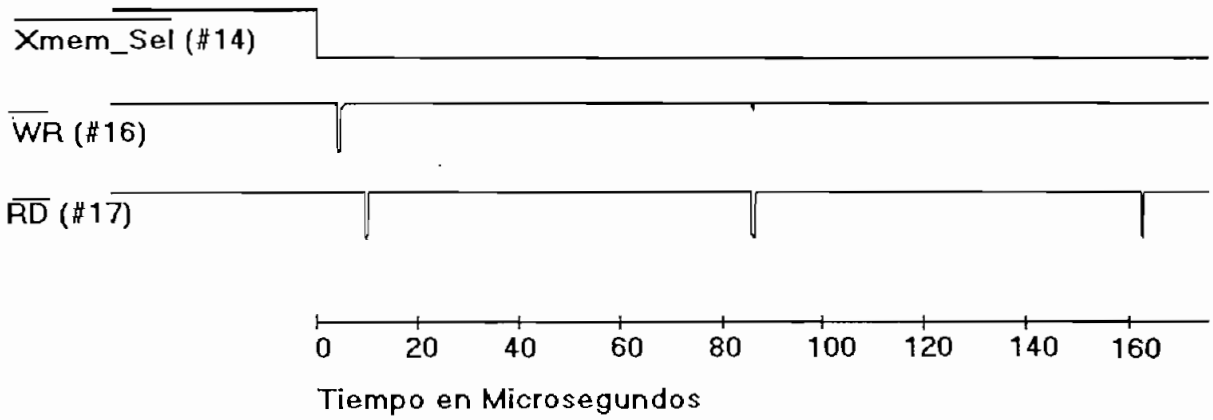


Figura 3.15

A continuación se presenta el diseño del interfase para la memoria, válido para el diseño en los dos ejes:

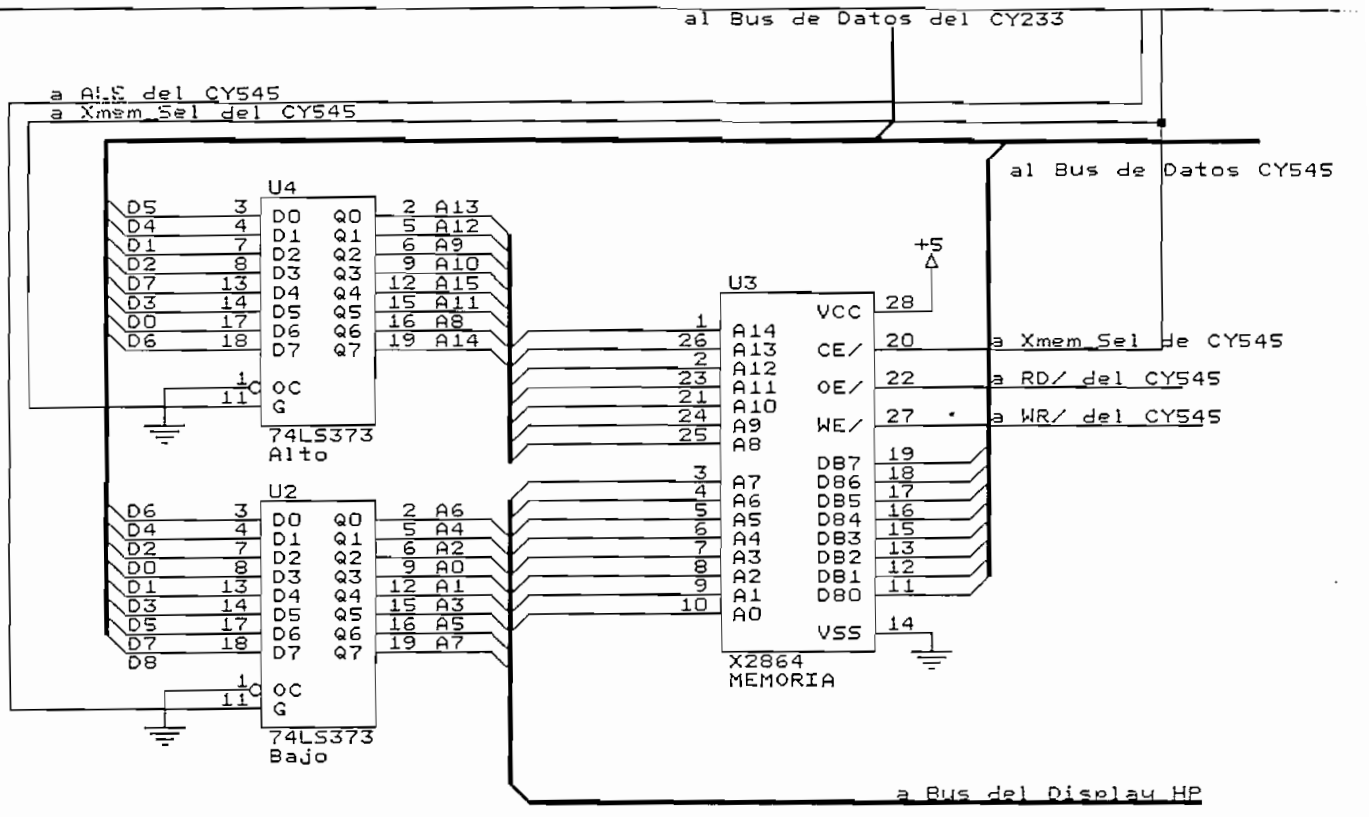


Figura 3.16

3.2.4.- Diseño del interfase para el Display HP.

El CY545 da soporte para los displays HDSP-211x de Hewlett Packard (En este caso se utilizará el display HDSP-2112). Este tipo de display presenta las siguientes características:

- Un interfase paralelo.
- 8 caracteres.
- Matricial.
- Intensidad programable.
- Tipos de letra definibles por el usuario.
- Drivers incluidos.
- Generador de caracteres.
- Refresco lógico.

El CY545 se conecta con el Display a través del bus paralelo de datos, utilizando el byte menos significativo y la señal HP_SEL (USRB7). Para el display HP solo se usan las 6 direcciones más bajas del bus. Estas direcciones deben ser demultiplexadas por un Latch externo, como el 74LS373; como en este caso se está utilizando memoria externa (EEPROM), las salidas del mismo latch de direcciones se pueden conectar al display HP.

Las Señales HP_SEL, RD y WR deben ser combinadas como se muestran (utilizando la compuerta AND(U5A 74LS08) y la compuerta OR(U6A 74LS32)) para generar la señal de habilitación de CHIP CE para este display. Los strobes RD y WR también proveen tiempos de transferencia para datos en estos displays. El byte menos significativo es usado para seleccionar los registros internos y la localización de caracteres en el display, mientras la señales de control generan la selección del display y los strobes de lectura y escritura.

Cuando se selecciona el display HP a través del comando MODE, cualquier mensaje que sea enviado será colocado en el display. Se debe tener cuidado de que estos mensajes no tengan más de 8 caracteres a la vez. Los códigos de caracteres de retorno no son colocados en el display, pero ellos resetean la dirección de caracteres al caracter más a la izquierda del display, y causan que el display se blanquee antes de que el próximo caracter sea escrito.

Los comandos de Query siguen este formato también, por lo que el display HP solo puede mostrar un parámetro a la vez. El display será blanqueado cuando se vaya a mostrar algún otro parámetro solicitado por el comando Query.

El CY545 controla los tiempos para el trabajo con el display HP, y genera las señales de control adecuadas. Adicionalmente, un comando especial ([ ) permite controlar el display para obtener la intensidad deseada de brillo, u otras funciones.

El diseño de este circuito es el siguiente:

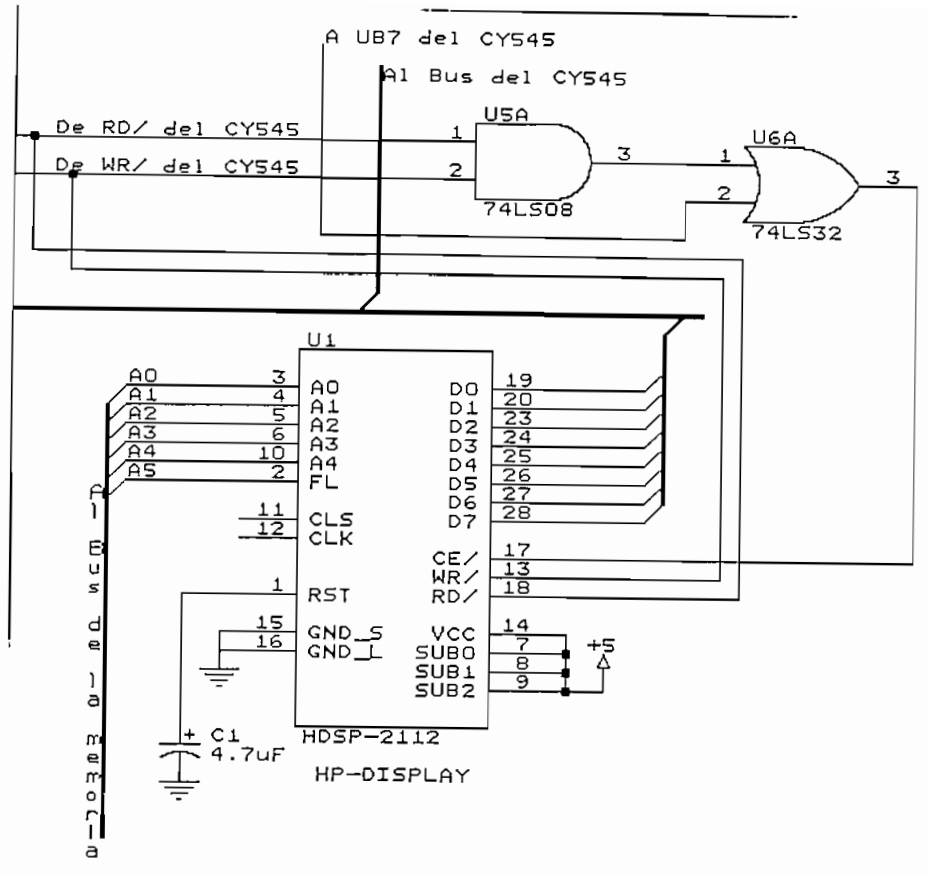


Figura 3.17

Cuando el CY545 escribe o lee en el display, inicialmente envía la señal HP_SEL a un nivel lógico bajo. La escritura o lectura se divide en dos partes. Primero, el byte menos significativo de direcciones se genera en el bus de datos. Este valor se almacena en el LATCH 74LS373, usando la señal ALE. Entonces, se produce la transferencia de datos. Para escritura, el CY545 genera los datos de escritura en el bus de datos, junto con una señal de strobe, WR. Para la lectura, el bus de datos se lo deja flotante, y se genera un strobe de escritura, usando RD, y la memoria toma el control del bus de datos.

Cuando se han transferido los datos, la señal HP_SEL va a alto nuevamente, deshabilitando los strobes RD y WR para el display. Una forma de onda típica de acceso al display es la siguiente:

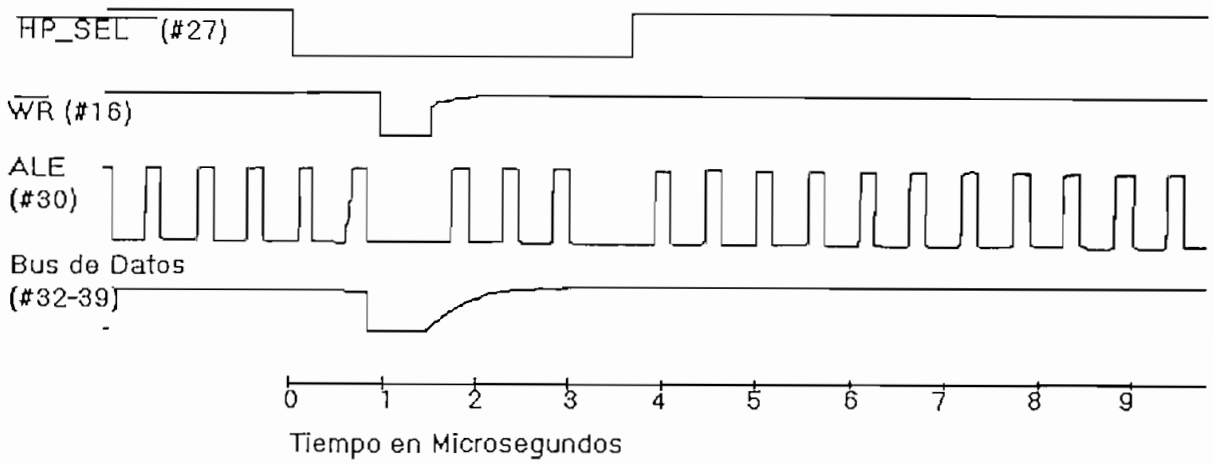


Figura 3.18

Este diseño es válido para los dos ejes.

3.2.5.- Utilización de los User Bits.

Como se ha dicho anteriormente, el CY545 tiene 8 señales que pueden ser útiles para funciones de propósito general. Estas señales son comandadas y censadas a través del comando BIT revisado en el capítulo 2. Se va a hacer el análisis de para qué van a ser utilizados cada uno de estos USERBITS. Como en este trabajo se van controlar cuatro motores, se han desarrollados dos sistemas muy similares para el control de todos ellos. Esto implica que se han utilizado dos Microcontroladores CY545, uno para cada sistema. Por lo tanto, en cada uno de los sistemas se han utilizado los UserBits para diferentes funciones, las mismas que se van a analizar:

3.2.5.1.- User Bits para el EJE X y Z.

En el caso del sistema relacionado con estos ejes, se ha utilizado los User Bits para las siguientes funciones:

User Bit 0: Este User Bit se encuentra conectado a la salida del circuito integrado 74LS244 (U11 Pin 9), correspondiente al USER BIT 0 del CY545 que controla el eje Y. Esto quiere que este USER BIT sensará el nivel lógico del USER BIT 0 en el eje Y. Esto permite controlar que el sistema no baje el taladro hasta que los motores del eje Y hayan terminado de moverse. Inicialmente, este UserBit se encuentra en un nivel lógico alto: una vez que termina el movimiento en el eje Y, el CY545 Y lo lleva a un nivel lógico Bajo (se enciende el Led correspondiente). El CY545 X detecta el 0l en USERBIT 0, y coloca 0l en el Userbit

1 para indicar al CY545Y que se ha leído la información.

User Bit 1: Se encuentra conectado, pero a través del buffer de corriente 74LS244(U11 Pin 12), al USER BIT 1 del CY545 que controla el eje Y. Esto permite avisar al CY545 Y que se ha leído el User Bit 0 (que indicaba que se ha terminado el movimiento en el eje Y). Inicialmente, USRB1 se encuentra en 1l; al leerse USRB0, CY545 X lo lleva a un nivel lógico bajo. Este UserBit será llevado a 1l nuevamente, una vez que se haya terminado de perforar el agujero, y antes de iniciar un nuevo proceso de perforación.

User Bit 2: Controla la posición inicial del Eje Z.

User Bit 3: Controla la posición Home del Eje X Normalemente se encuentra en 1L. Cuando la tarjeta de efecto Hall sensa que el motor que controla el eje X se encuentra cerca, envia un 0L. De esta manera se determina la posición inicial de trabajo.

User Bit 4: Se lo usa para controlar el encendido y apagado del taladro. Cuando se encuentra en 1L el taladro el relé de estado sólido no enciende el taladro. Cuando baja a 0L el taladro perfora el agujero respectivo.

User Bit 5: Puesto que el CY545 X controla tanto el eje X como el eje Z, los pulsos producidos por este microcontrolador deberán ser enviados a uno de los dos motores de paso (X o Z), dependiendo del eje que desee moverse. Para ello, se ha utilizado el demultiplexor 74155N, el cual ruteará las señales PULSE y DIR del CY545X o al motor de paso del eje X o al motor de paso del eje Z. El control de Selección del multiplexor está directamente controlado por el UserBit 5; cuando este bit de usuario se encuentra en un nivel lógico alto, la selección del demultiplexor permite que las señales se enruten hacia el eje X (a través de las salidas 1Y0 y 2Y0). Cuando el USRB5 está en bajo, el demultiplexor enruta las señales hacia el motor de paso que controla el eje Z (a través de las salidas 1Y1 y 2Y1).

User Bit 6: Sin Uso. Se lo podría emplear para generar la señal CTS para transmisión serial, pero como no se utiliza en los CY545 el interfase serial para transmisión de datos, USRB6 puede ser empleado para cualquier función adicional.

User Bit 7: Utilizado para comandar el Display HP. Está conectado al Pin 2 del circuito integrado 74LS32(U6A).

3.2.5.2.- User Bits para el EJE Y.

En el caso de este eje, se ha utilizado los User Bits para las siguientes funciones:

User Bit 0: Este User Bit se encuentra conectado a través del circuito integrado 74LS244(U11 Pin 9) al User Bit 0 (Pin 21) del CY545 que controla el eje X. Esto quiere que el nivel lógico de este USER BIT será censado por el sistema que controla el eje X. Esto permite controlar que no se baje el taladro hasta que se haya terminado el movimiento en los dos ejes X y Y.

User Bit 1: Se encuentra conectado al buffer de corriente 74LS244(U11 Pin 12) del sistema que controla el eje X, y por intermedio de este circuito integrado al User Bit 1 del eje X. Una vez que el CY545 X ha leído el USRB0 del CY545Y (que indica que se ha terminado el movimiento), el CY545X lleva USRB1 a 0l, para indicar al CY545Y que se ha leído la información. Entonces, CY545Y lleva a USRB0 a 1l nuevamente.

User Bit 2: Sin Uso.

User Bit 3: Sin Uso.

User Bit 4: Se lo usa para controlar la posición inicial de trabajo del Eje Y.

User Bit 5: Sin Uso.

User Bit 6: Sin Uso. Se lo podría emplear para generar la señal CTS para transmisión serial, pero como no se utiliza en los CY545 el interfase serial para transmisión de datos, USRB6 puede ser empleado para cualquier función adicional.

User Bit 7: Utilizado para comandar el Display HP. Está conectado al Pin 2 del circuito integrado 74LS32(U6A).

3.2.6.- Diseño del Interfase de control de los motores de paso.

El CY545 implementa ocho señales dedicadas al control de los motores de paso, y adicionalmente dispone de los bit seleccionables por el usuario (User Bits), los cuales pueden generar señales de control para hardware especial, o pueden testear condiciones especiales que guían el CY545 a través de un programa de movimientos (Un análisis de la utilización de los User Bits se lo ha hecho ya anteriormente).

Antes de analizar el diseño de este interfase, se van a analizar las señales que tiene que ver con el control de los motores de pasos.

3.2.6.1.- Señales que controlan el movimiento de los motores de paso.

Estas señales son las siguientes:

3.2.6.1.1- Señales PULSE y CCW.

Las dos señales de control primarias para los motores de paso son PULSE y DIR, las cuales proveen las indicaciones de dirección y pasos para el motor. Estas señales son conectadas, a través de traductores, a drivers de corriente externos (de Potencia) que son los que de hecho proporcionan la corriente para que el motor se mueva. Cada vez que el CY545 direcciona al motor para que se mueva un paso, una señal es generada en la línea PULSE.

El Driver de Potencia hace avanzar el motor un incremento por cada pulso. El incremento de pasos puede ser de un paso completo, un medio paso, un cuarto de paso o un micro paso. Esto debe ser determinado por la aplicación y por el driver de potencia; en este caso se va a utilizar el paso completo. El CY545 no se preocupa del tamaño de los pasos. Toda información de posición es tratada en términos de incremento de pasos, y no en términos de incremento del desplazamiento lineal o grados de rotación del motor.

La señal CCW es una señal que le dice al traductor en qué dirección mover el motor. Cuando está en alto, el driver moverá el motor en contra de las manecillas del reloj, y cuando está en bajo, lo moverá en dirección de las manecillas. Nótese que la rotación física del motor depende de como los bobinados estén conectados al driver de potencia, y no de el nivel de la señal CCW. Cuando esta señal está en alto, el CY545 decrementa la posición actual por cada paso que se mueva, y cuando está en bajo, incrementará la posición actual por cada paso movido.

También debe notarse que la señal CCW puede ser utilizada como una entrada, para forzar pasos en la dirección deseada. En este caso, el comando de dirección de software debería ser seteado a "-", así el CY545 trata de conducir la señal CCW a alto. Después de esto, un circuito externo podrá fácilmente forzar a la señal CCW a bajo, causando que el motor se mueva en la dirección "+". Si esto se intenta mientras el CY545 está llevando la señal CCW a bajo, un circuito externo muy fuerte se necesitaría para lograr que esta señal vaya a alto, ya que tiene mucha más fuerza la corriente externa que la corriente de fuente. En el caso de este trabajo, no se va a utilizar el señal CCW como señal de entrada, pero se ha explicado su utilidad por información.

Varias formas de onda para la señal de PULSE se presentan a continuación, para diferentes velocidades de pasos. Nótese que la señal PULSE está normalmente en alto, y va a un nivel lógico bajo cuando se inicia el movimiento. Permanece en bajo durante una cantidad determinada de tiempo, la cual depende de la frecuencia del cristal del CY545, y va a alto nuevamente durante la duración del tiempo de paso. Cuando llega el tiempo para el próximo paso, va a bajo nuevamente.

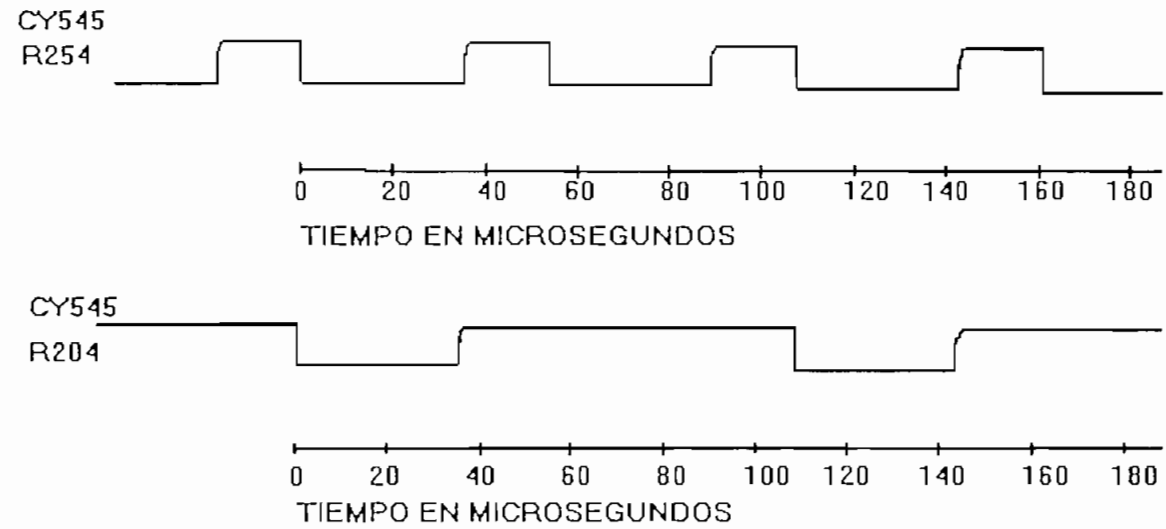


Figura 3.19a

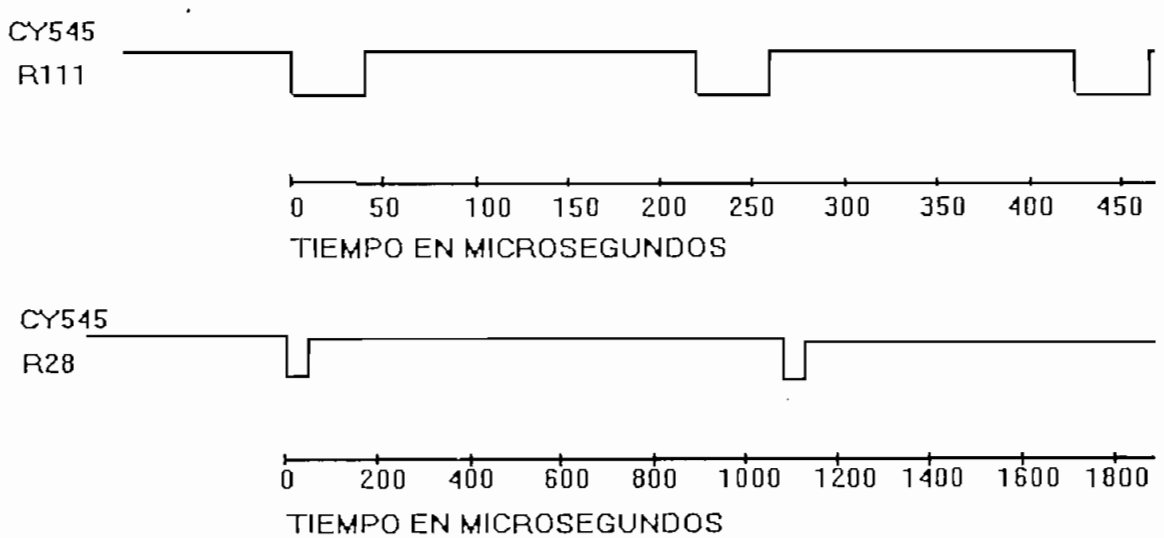


Figura 3.19b

Cuando los parámetros de loa velocidad de los pasos son fijados para habilitar la aceleración, el CY545 variará la velocidad de los pasos desde una velocidad inicial(Parámetro de velocidad inicial) a una velocidad final (Parámetro de Velocidad Final). Esto significa que la forma de onda de PULSE variará desde una velocidad baja a una velocidad más alta, para luego estabilizarse durante el período de slew. El CY545 debe determinar cuando iniciar la desaceleración , de tal manera que el motor vaya moviéndose más lentamente, hasta llegar a la velocidad inicial, y se pare.

3.2.6.1.2.- Señal STOPPED.

La señal STOPPED es una salida del CY-545 que determina el Status del mismo, indicando si el motor se está moviendo o no. Al inicio de un movimiento, desde un comando Go o un comando Position, la señal STOPPED va a un nivel lógico bajo. Permanecerá en bajo mientras dure el movimiento. Al final del movimiento, la señal STOPPED va a un nivel lógico alto nuevamente. La transición de bajo a alto de la señal STOPPED indica que el movimiento ha acabado, y el nivel alto indica que el motor se ha detenido. Una forma de onda típica de esta señal es la siguiente:

Figura 3.20a

Esta señal puede ser usada como un indicador de movimiento completo y puede ser utilizada por un Host. Esta señal también puede controlar la selección del nivel de voltaje para los drivers de potencia de los motores de pasos, intercambiando automáticamente entre una potencia alta mientras se mueven los motores y una potencia de parqueo cuando los motores no se mueven. Esto permite disponer de un torque máximo cuando los motores están en movimiento, y un torque mantenido cuando los motores no se mueven, con el objetivo de evitar el sobrecalentamiento de los motores. El CY545 automáticamente se demora 5 milisegundos después del último paso, antes de llevar la señal STOPPED a un nivel lógico alto. Esto le permite al motor llegar a su posición final antes de que el nivel de potencia alto se retire.

3.2.6.1.3.- Señal SLEW.

Otra salida que permite controlar el STATUS del CY545 es la señal de SLEW, la cual indica cuando se ha alcanzado la velocidad final. Esta señal está normalmente en alto, y va a bajo cuando el CY545 está moviendo el motor sin aceleración o desaceleración. A continuación se muestra una típica forma de onda (Figura 3.20 b). Debe notarse que el tiempo en el que STOPPED y SLEW están en un nivel lógico bajo, representa el tiempo necesario para acelerar y desacelerar el motor.

La señal de SLEW también puede ser usada como una entrada. Si esta señal es llevada a un nivel lógico bajo al inicio de un movimiento, el CY545 entrará en el modo de movimiento continuo. En este modo, acelerará desde la velocidad inicial a la velocidad final, y comenzará a mover el motor indefinidamente a esta velocidad; no habrá ningún número específico de pasos para mover el motor, o alguna posición específica en la cual detenerse. Sin embargo la posición actual será registrada constantemente en su registro respectivo. La señal INHIBIT_ABORT puede usarse para terminar un movimiento continuo. la señal SLEW es chequeada justo antes de que el CY545 comience el primer paso de un movimiento.

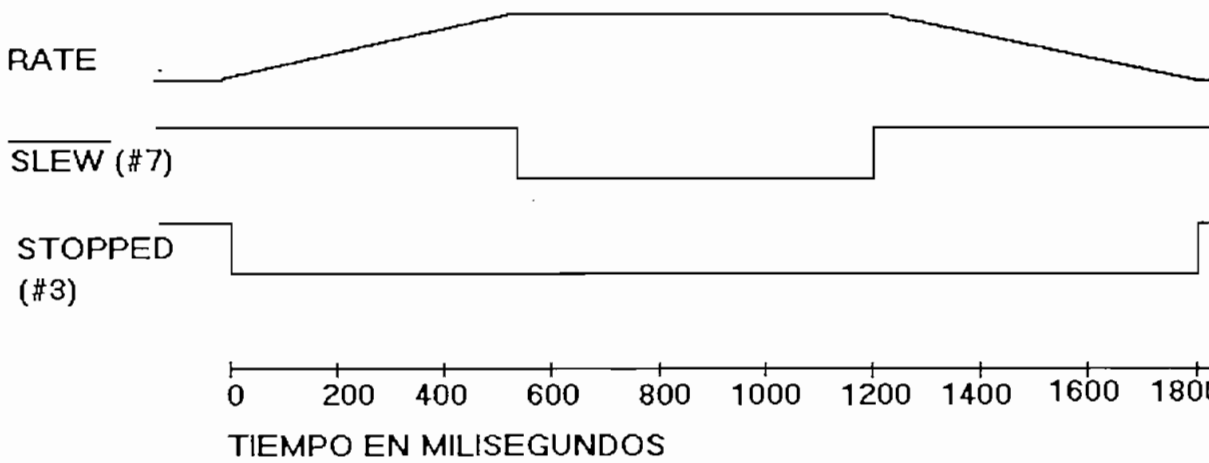


Figura 3.20b

Alternativamente, el modo continuo puede seleccionarse con el comando Continous. Debería enviarse seguidamente un comando GO para iniciar el movimiento.

3.2.6.1.4.- Señal INHIBIT_ABORT.

La señal INHIBIT_ABORT es una entrada de control de pasos para el CY545, con un número de funciones. En general, es usada para cambiar la conducta normal del CY545.

Si esta señal se encuentra en un nivel lógico alto al inicio de un movimiento, comenzado por los comandos Go o Position, este movimiento no comenzará hasta que la señal INHIBIT_ABORT se encuentre en un nivel lógico alto. Esto permitirá sincronizar el movimiento a una señal externa, en lugar de a un comando del CY545. Múltiples ejes de movimiento se podrían comenzar simultáneamente usando esta señal para controlar el inicio de los mismos. Un ejemplo de forma de onda es el mostrado a continuación:

Figura 3.21

Una vez que se ha iniciado el movimiento, esta señal puede ser usada para desacelerar y detener el motor antes de que el CY545 alcance la posición especificada.

Si esta señal se envía a un nivel lógico bajo durante un movimiento, el CY545 cambia de modo e inicia una desaceleración. La desaceleración es simétrica a la aceleración, hasta que se alcanza el valor de velocidad inicial (First Rate). El CY545 revisará entonces la señal INHIBIT_ABORT para cada paso, y si está en bajo, el movimiento parará. De otra manera, el movimiento continuará, a la velocidad de inicio, hasta que se alcance la posición especificada. Para desacelerar el motor y pararlo cuando se alcanza la velocidad inicial, simplemente se debe mantener la señal INHIBIT_ABORT en bajo hasta que el CY-545 lleve a la señal STOPPED a un nivel lógico alto de nuevo. Algunos casos de este tipo son ilustrados a continuación:

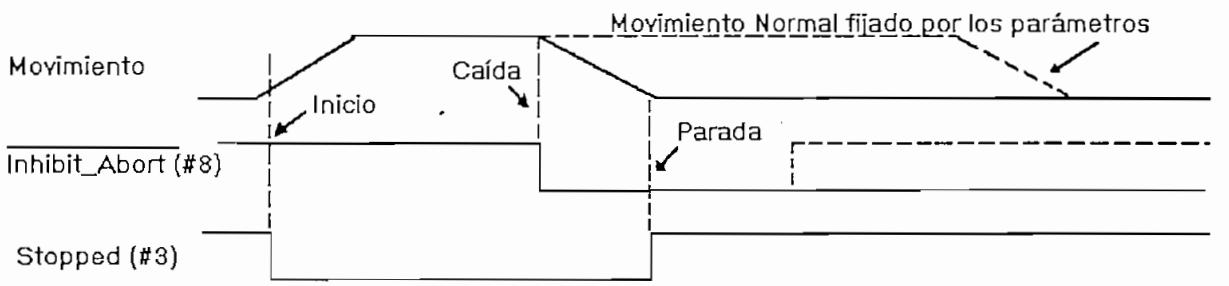


Figura 3.22a

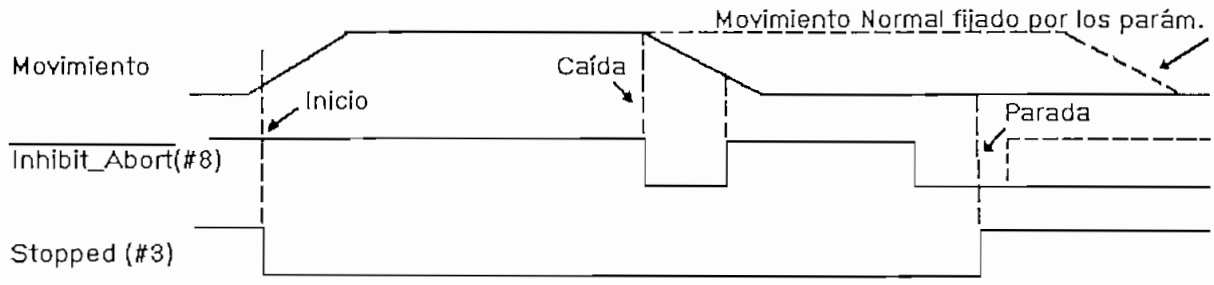


Figura 3.22b

3.2.6.1.5.- Señales CW_LIMIT y CCW_LIMIT.

El CY-545 también provee dos entradas para parar inmediatamente el motor, la señal CW_LIMIT y la señal CCW_LIMIT. Si la señal correspondiente va a un nivel lógico bajo durante un movimiento en esa dirección, el CY545 inmediatamente para el motor. No habrá desaceleración antes de la parada, el CY545 simplemente abandonará el movimiento.

También, si una señal de límite esta abajo cuando se inicia un movimiento, no se realizará este movimiento. El CY545 abandona inmediatamente el movimiento sin haber movido un solo paso. Sin embargo, movimientos normales se permiten en la otra dirección. Ejemplos de estas formas de onda se muestran a continuación:

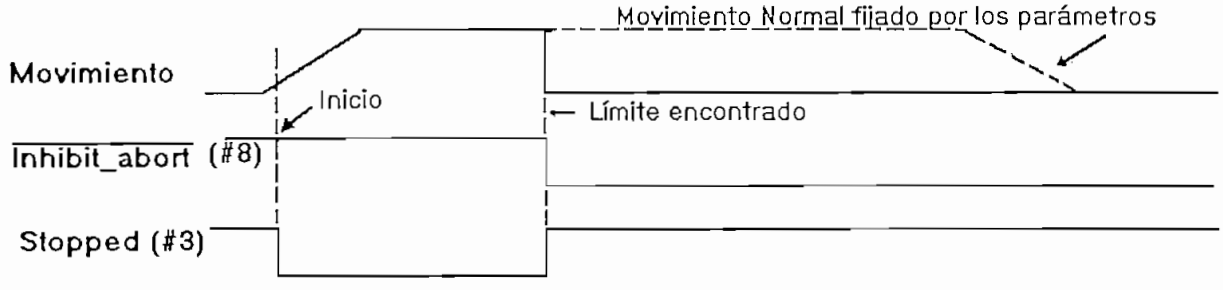


Figura 3.23

Se debe tomar en cuenta que, aunque estas señales proveen al CY545 de alguna protección cuando accidentalmente se alcanza un límite, ellas no deberían ser usadas como las únicas protecciones si al alcanzar un límite se daña equipo importante o se producen daños a algún operador.

El hecho de que se alcance un límite implica que el sistema ha fallado, y no un componente,

incluyendo el computador o el CY545.

La mejor protección en casos en que el sistema falle, es deshabilitar la fuente de poder para los drivers de los motores de pasos. Esto permitirá a los motores pararse, aún si ellos están comandados para moverse continuamente. Entonces, se deberá determinar la causa de la falla y deberá arreglarse antes de energizar nuevamente los motores.

3.2.6.2.- Diseño del Traductor.

Como se ha mencionado anteriormente, se ha utilizado en este diseño motores de paso de la empresa US-CYBERLAB, que incorporan en su parte posterior el traductor. Tan solo requieren de las señales Pulse, Dir, Home, Vcc y Gnd para funcionar.

A pesar de ello, se va a hacer un análisis de las condiciones básicas que debe cumplir un traductor.

Los motores que se han utilizado en este diseño son motores de pasos de 4 fases. Por lo tanto, sabiendo que el CY545 entrega tan solo un tren de pulsos para que el motor se mueva, es necesario convertir este tren de pulsos en información útil para el motor de pasos. Las señales del CY545 que se van a utilizar en este interfase son PULSE y DIR. PULSE entregará el tren de pulsos (uno por cada paso), y DIR entregará la dirección hacia la que debe moverse el motor. Un diagrama de bloques de este circuito es el siguiente:

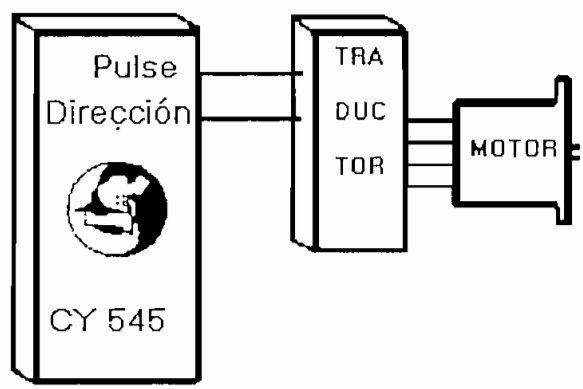


Figura 3.24

Un circuito traductor básico se muestra a continuación:

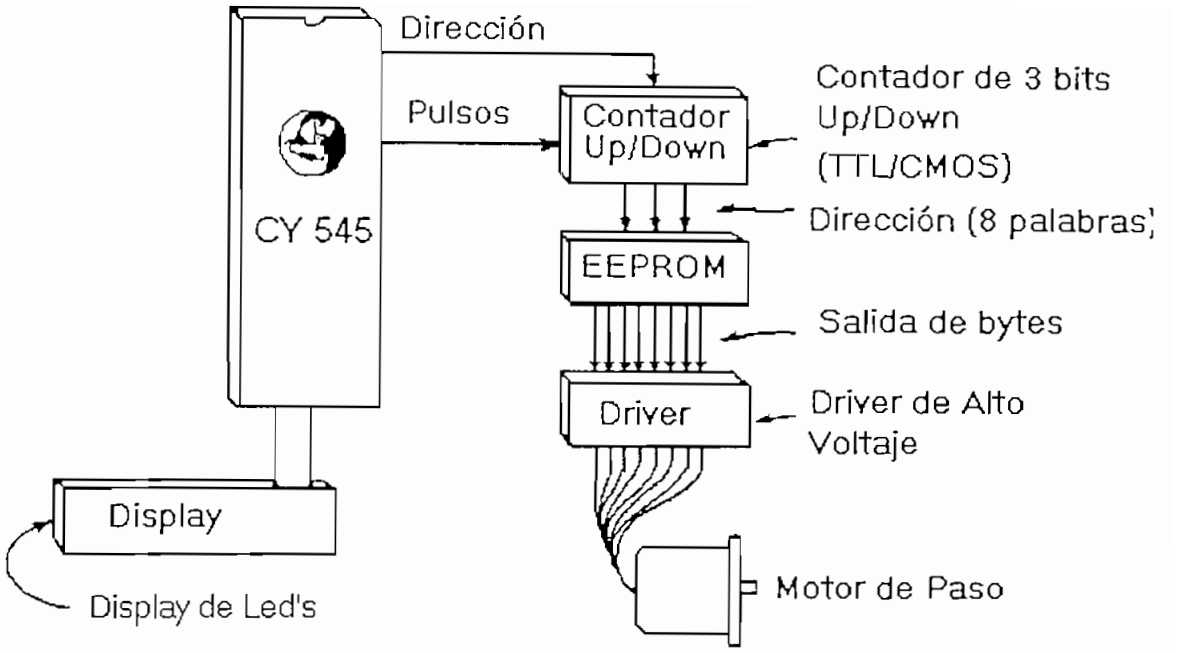


Figura 3.25

El contador que se podra uíilizar es el 74191 al cual se le programaría para que sea un contador módulo 4; se utilizarían las salidas QA (Pin 3) y QB (Pin 2). Estas señales serán conectadas a la EEPROM (2716) a los pines 8 (A) y 7(B) respectivamente.

La EEPROM tiene grabado en sus cuatro primeras direcciones, la secuencia necesaria para que los motores de pasos funcionen. Las salidas de la EEPROM se conectarán al driver de potencia, generando los patrones de fase adecuados para el motor. Nótese que este diseño no solamente maneja motores de varias fases, sino que también permite implementar esquemas de medio paso o cuarto de paso, así como patrones standard de paso completo.

A continuación se muestra un patrón para paso completo:

Contador		EEPROM 2716			
Qb	Qa	Qd	Qc	Qb	Qa
0	0	0	1	0	1
0	1	1	0	0	1
1	0	1	0	1	0
1	1	0	1	1	0

Las salidas de la EEPROM Qa, Qb, Qc y Qd van directamente al driver de potencia, que es el que entregará la potencia necesaria para que los motores se muevan.

El CY-545 provee las señales lógicas y de tiempo necesarias para el control de los motores de pasos. Sin embargo, para hacer un sistema completo, se debe añadir un driver al CY-545. El circuito tomará las señales lógicas generadas por el CY-545 y las traducirá a señales de alta potencia necesarias para correr el motor.

Los diseños unipolares son los más simples, y son generalmente útiles cuando se corre a velocidades menores que 600 pasos/segundo. Estos diseños requieren de motores con 6 u 8 salidas, puesto que la fuente de poder está conectada a la mitad de cada devanado. El fin de cada devanado se lo envía a tierra a través de un transistor controlado por una de las líneas de fase.

El rendimiento de los motores puede ser mejorado añadiendo una resistencia entre la salida de la fuente de poder y el tap central de cada bobinado. Esto decrementa la constante de tiempo de retraro del motor, dando mayor rapidez en la respuesta. El incremento en el rendimiento se paga con el uso de fuentes de poder con volajes más altos y pérdidas por calor en las resistencias. Este tipo de circuitos se conocen como L/xR, donde la x representa el valor de la resistencia relativo a la resistencia del devanado. Un circuito L/R no tendría una resistencia externa, mientras que un circuito L/4R usaría una resistencia de valor tres veces superior a la de la resistencia del bobinado. Nótese que la fuente de poder debería ser cuatro veces el valor nominal del motor con este circuito. También debe notarse que este circuito solamente requiere de un voltaje y un transistor por fase. Los dos principales problemas de esta técnica son las pérdidas de torque a altas velocidades y la pobre eficiencia del motor.

Se pueden utilizar circuitos L/R de alta potencia. Uno de ello, UCN 5804, usa cuatro transistores Darlginton. A pesar de ser sencillos de entender, estos circuitos requieren de buenas técnicas de construcción.

El segundo tipo básico de circuitos es el diseño bipolar. En este caso, el motor es conducido únicamente desde el fin de cada bobinado, con lógica switching usada para controlar la dirección de la corriente a través del bobinado. Estos circuitos pueden ser implementados con motores de 4 entradas, puesto que solamente el se necesita el fin de cada bobinado.

Los diseños bipolares son mas eficientes en conducir el motor, y permiten un rendimiento mayor que los diseños unipolares. También hay una ganancia en torque, puesto que el bobinado entero está conduciendo siempre, no como en el diseño unipolar, en el cual solamente la mitad del bobinado es usado a la vez.

Dos métodos de switcheo de la dirección de la corriente pueden ser usados. Con una fuente de poder con voltaje único, se necesitan 8 transistores, dos por fase. Los transistores son encendidos en pares alternado a través de cada bobinado para controlar la corriente. La segunda alternativa usa solamente cuatro transistores, pero requiere una fuente de poder de doble voltaje. En este caso, un lado

de cada bobinado se conecta a tierra, y el otro es switcheado entre las fuentes de poder positivas y negativas. En los diseños es muy importante asegurar que los dos transistores de cada devanado no operen al mismo tiempo, puesto que se cortocircuitaría la fuente de poder a través de los transistores, y generalmente destruyendo los transistores en este proceso. Usualmente, se debe incluir protección lógica para asegurar que un transistor no opere antes de que el otro sea encendido.

Los diseños más avanzados de drivers son variaciones de los tipos unipolar y bipolar, aunque generalmente son implementados utilizando tecnología bipolar. Estos driverse son capaces de controlar las velocidades mas altas. Trabajan conmutando corriente o voltaje a través del motor a velocidades mucho mayores que las estimadas. Esto se lo puede hacer solamente por un período de tiempo corto , causando que el campo magnético cambie en el motor muy rápidamente, sin exceder la disipación máxima de energía en el motor. Mientras el promedio de disipación no exceda las especificaciones, el motor trabajará adecuadamente. Una vez que se ha alcanzado el límite, el motor puede sobrecalentarse y destruirse.

Una técnica para incrementar el rendimiento del motor sería aplicar un voltaje mas alto al motor en el inicio de cada paso. Esto hace que el motor reaccione muy rápidamente al cambio en las señales de fase. Después de un corto período de tiempo, el voltaje es conmutado a un valor bajo, permitiendo que el motor continue moviéndose sin sobrecalentarse.

Otra técnica de mejoramiento, conocida como diseño de corriente constante, sensa la cantidad de corriente que pase a través de los bobinados, y ajusta el voltaje aplicado al motor para mantener la corriente en el valor máximo especificado. Al inicio del movimiento, el voltaje sería bajo, con ajustes constantes a valores más alto mientras el motor incrementa la velocidad.

Otra técnica, conocida como chopping, también puede ser aplicada a estos diseños. Permite aplicar un voltaje mucho más alto que el valor especificado por un corto período de tiempo. Entonces, se retira este voltaje por otro período de tiempo. Esto ocurre muchas veces por cada paso, con la frecuencia de conmutación conocida como la frecuencia chopping. Esta frecuencia puede ser controlada por tiempo, conmutando a una velocidad dada, o puede ser controlada sensando el flujo de corriente a través del motor, conmutando a una velocidad variable. Esta técnica de conmutación controla la disipación en el motor, permitiendo que trabaje al máximo torque en cualquier velocidad de pasos, ajustando la potencia promedio aplicada. Los drivers de mas alto rendimiento son generalmente diseñados como circuitos bipolares chopping. Un circuito integrado típico usado en estos diseños es el L297 de SGS-Thomson. Este circuito combinado con o el L298 o el L6203, puede proveer hasta cuatro amperios de corriente por bobina.

También se han diseñado drivers de "Micropaso". Esta es una técnica usada para producir muchos pasos intermedios entre los pasos normales del motor. El driver de micropasos lleva el concepto de mitad

de paso en el motor (encender dos bobinas en el motor al mismo tiempo). Este tipo de driver usa dos DACs para producir formas de onda senoidales y cosenoidales en el motor, en lugar de pulsos de pasos de encendido o apagado. El control de corriente usualmente se lo realiza usando técnicas chopping para micropasos.

Como se ha dicho anteriormente, para este trabajo se han utilizado motores de paso que incluyen los traductores completos. Solamente ha sido necesario regular el voltaje provisto por la fuente de 12 Voltios, para cada uno de los motores, utilizando un circuito regulador 7812 de 1 amperio. El circuito diseado para este efecto se analiza posteriormente.

3.2.6.3.- Motores de Pasos.

Haciendo un análisis, en base al trabajo que deben realizar los motores de pasos, y principalmente, en base a la fuerza que debe realizar el motor de pasos que controlará el taladro, se han utilizado escogido los motores de paso fabricados por la Empresa US-Cyberlab. Estos motores de pasos tienen las siguientes características:

CANTIDAD:	4
MARCA:	US-CYBERLAB
VOLTAJE NOMINAL:	12 Voltios.
CORRIENTE POR PASO/BOBINA:	1 Amperio.
PRECISION:	1.8 Grados/Paso (Pasos Completos)
TORQUE:	530 onzas/pulgada.

La rotación del eje del motor se divide en fracciones, y es de 1.8 grados/Paso, o 200 pasos en 360 grados.

El eje principal de cada uno de los sistemas armados (Eje X, ejes Y y Eje Z) permite una resolución de 0.1 pulgadas por revolución. Entonces, puesto que el eje del motor está acoplado directamente al eje mecánico de movimiento, la resolución total del sistema será:

$$200\frac{Pasos}{Revolución} * \frac{Revolución}{0,1pulgadas} = 0.0005\frac{pulgadas}{pasos}$$

Debe tomarse en cuenta que excesivas resoluciones limitarán la velocidad máxima de movimiento del sistema. Al reducir la resolución se incrementará esta velocidad. Si la velocidad de trabajo de los motores (Parámetro R=30) fuese de 1800 pasos/Segundo, la velocidad lineal de movimiento sería:

$$1800\frac{pasos}{segundo}*60\frac{segundos}{minuto}*1\frac{revolución}{200pasos}=540\frac{revoluciones}{minuto}$$

$$540\frac{revoluciones}{minuto}*0.1\frac{pulgadas}{revolución}=54\frac{pulgadas}{minuto}$$

Estos motores poseen suficiente fuerza para mover todo el sistema mecánico y para permitir que el taladro perfore los circuitos impresos.

3.2.6.4.- Diseño del Circuito de Potencia para los motores de pasos.

Para que los motores de paso funcionen adecuadamente, necesitarán de una fuente de corriente, la misma que deberá proporcionar la corriente necesaria para que cada uno de estos motores se mueva.

Se pueden utilizar algunos drivers de corriente disponibles en el mercado. Los factores que inciden en los drivers de potencia son los siguientes:

- Voltaje de los motores de Pasos.
- Voltaje de la señales de control del driver.
- Corriente por paso.
- Fase de los motores de paso.

En este caso se han debido hacer las siguientes consideraciones:

- Los motores de paso trabajan a 12 voltios.
- Las señales de control tiene niveles TTL.
- La corriente por paso deberá ser de 1 Amperio.
- Los motores de pasos son de cuatro fases.

Luego de hacer el estudio de diferentes tipos de drivers, entre ellos, el STK6982B. Este driver para motores de cuatro fases (1.5 Amp/fase), trabaja a 24 Voltios, y por lo tanto no se lo utilizó. Se analizó también el driver STK6722H; este driver maneja también motores de cuatro fases, pero trabaja a 36 voltios. Entonces, se recurrió a los drivers de la empresa que suministró los motores de pasos, los drivers US-Cyberlab. Estos drivers de motores de cuatro fases, trabajan a 12 voltios, 1 amperio por fase y sus señales de control aceptan niveles TTL.

Adicionalmente, estos drivers permiten controlar la posición HOME de los motores de pasos, mediante el uso de la señal proporcionada por la tarjeta que controla del efecto HALL.

Los motores US-CYBERLAB tienen una característica especial, que fue una de las principales razones por la que se decidió utilizarlos: Tienen incluido en la parte posterior, el driver de corriente, el interfase que convierte los pulsos en pulsos para 4 fases y la señal que controla la posición HOME a través de la tarjeta censora del efecto HALL.

Para que los motores funcionen adecuadamente, solamente ha sido necesario regular el voltaje provisto por la fuente de 12 Voltios, para cada uno de los motores, utilizando un circuito regulador 7812 de 1 amperio. Además, por recomendación del fabricante de los motores, se ha incluído una resistencia de 10 Watts, 4 Ohmios, para evitar que se quemen los motores cuando el motor deje de funcionar (Por razones analizadas anteriormente). El circuito construído es el siguiente:

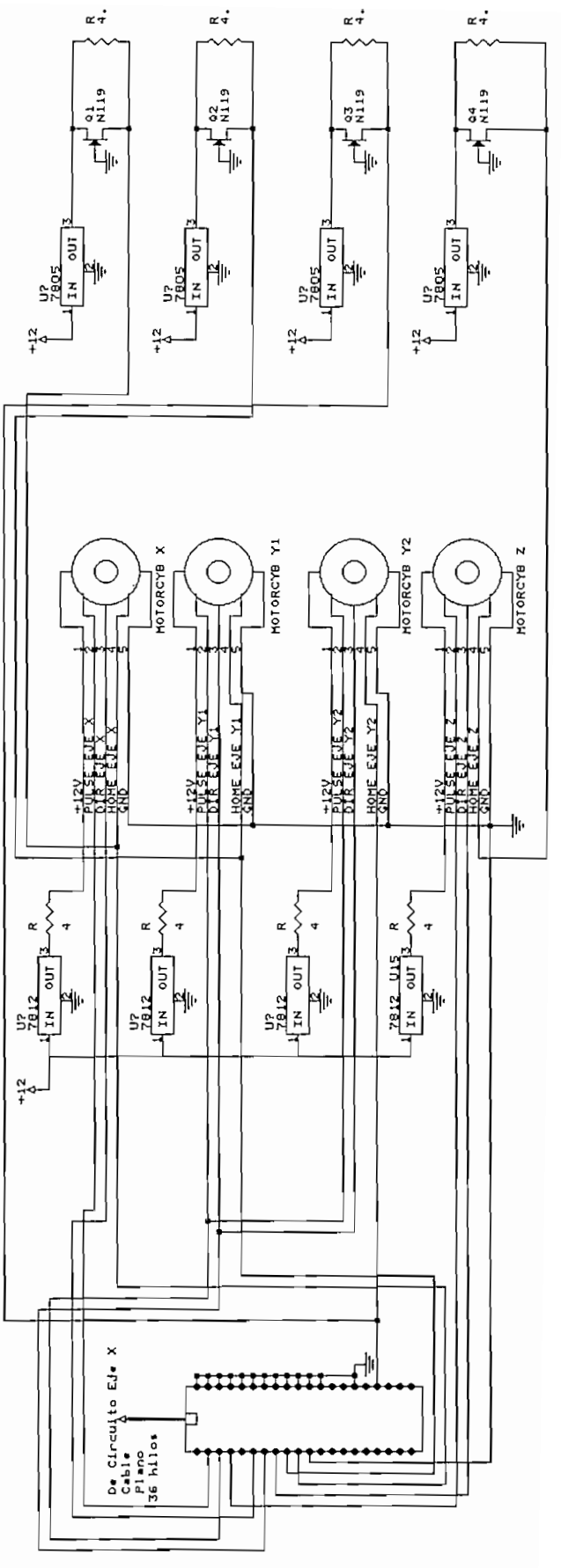

R 4.
Q1 N119
Q2 N119
Q3 N119
Q4 N119
U? 7805
IN
OUT
+12
MOTORCYB X
MOTORCYB Y1
MOTORCYB Y2
MOTORCYB Z
+12V
PULSE EJE X
DIR EJE X
HOME EJE X
GND
PULSE EJE Y1
DIR EJE Y1
HOME EJE Y1
PULSE EJE Y2
DIR EJE Y2
HOME EJE Y2
PULSE EJE Z
DIR EJE Z
HOME EJE Z
R 4
U? 7812
7812 U15
De Circuito EJe X
Cable
Plano
36 hilos

Figura 3.25

Este circuito se ha construído en otra placa, y recibe la información de los microcontroladores CY545 a través de un cable paralelo de 36 hilos. A través de este cable viajan las siguientes señales:

Pulse Eje X
Dir Eje X
Pulse Eje Y
Dir Eje Y
Pulse Eje Z
Dir Eje Z
Gnd

Para cada uno de los motores de pasos salen cinco señales de esta placa, a través de conectores tipo DIN. Estas son todas las señales necesarias para que los motores US-Cyberlab funcionen. Estas señales son:

+12 Voltios
Pulse
Dir
Home
Gnd

Adicionamlemente, como se puede observar en la figura 3.25, se han desarrollado cuatro placas extras que contendrán el sistema de Efecto Hall que sensará la posición HOME de cada uno de los motores.

El diseño de estas placas, cuyo análisis se hará en otro punto, es el siguiente:

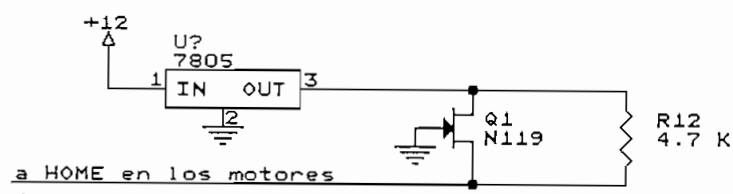


Figura 3.26

Adicionalmente, es necesario indicar que se ha utilizado en cada uno de los CY545, las compuertas 7404 (inversores). Esto permitirá adaptar los niveles de las señales CMOS que salen del CY545, a los niveles TTL provistos por estas compuertas. Las dos señales del CY545 que son enviadas a estas compuertas son: PULSE y DIR, puesto que son las señales utilizadas por los motores de paso.

El circuito utilizado para este diseño es el siguiente:

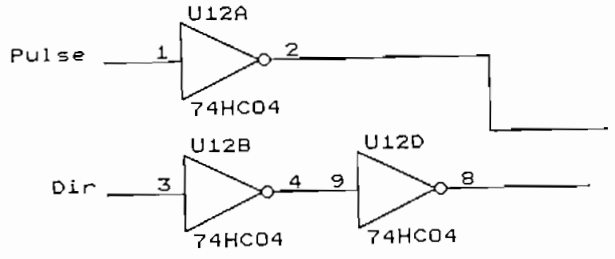


Figura 3.27

3.2.6.5.- Operación en el modo JOG.

El CY545 provee una función de control manual de los pasos a través de la señal JOG. Esta señal es testeada cuando el CY-545 no está ejecutando ningún comando o realizando algún otro movimiento. Esta es una señal especial, puesto que se chequea si está en uno de tres diferentes estados, y es una señal de entrada y de salida.

Cuando la línea se la deja flotante, esta función es deshabilitada, y el CY545 mueve los motores solamente cuando se recibe un comando de movimiento.

Cuando la línea está en un nivel lógico bajo, el CY545 moverá el motor en el sentido de las manecillas del reloj. La señal CCW cambiará de estado, si es necesario, para seleccionar la dirección apropiada.

Cuando la señal está en alto, el CY545 moverá el motor en contra de las manecillas del reloj. Igualmente, la señal CCW cambiará de estado si es necesario, para seleccionar esta dirección.

La señal CCW es reestablecida a su posición original cuando cada paso simple de jog finaliza.

La señal JOG está conectada a un switch de tres posiciones, Off en el centro. Uno de los lados del switch está conectado a tierra, mientras el otro lado está conectado a +5 voltios. El motor se mueve en cualquiera de las dos direcciones llevando el switch desde el centro a cualquiera de los dos lados. Esto se puede observar en las siguientes figuras:

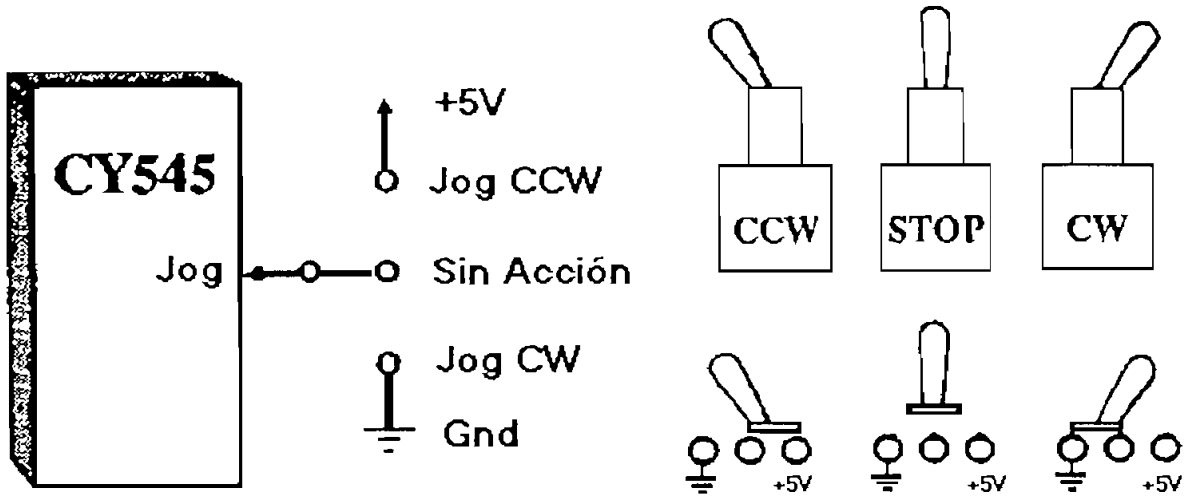


Figura 3.28

Cuando se activa esta señal, el CY545 mueve el motor un paso, luego chequea si existe algún nuevo comando de entrada. Si está disponible un nuevo comando, lo ejecutará; si no, será chequeada nuevamente la señal de JOG. Cuando el CY545 está desocupado, el efecto de la línea de Jog es mover el motor repetidas veces en pasos simples. Debe notarse que el chequeo de nuevos comandos mientras el motor se mueve, permite que se envíen estos comandos al CY545 mientras se encuentra ejecutando la acción apropiada debida a esta línea, proveyendo al mismo tiempo de estos movimientos y de la capacidad de ejecutar comandos.

3.2.7.- Diseño del Circuito de Control del Taladro.

Básicamente, el diseño de este circuito de control funciona apoyado por el sistema que controla el eje X, y se mueve el taladro a través del eje Z. Se han utilizado las mismas señales de este CY545. Como las señales que controlan los motores de pasos son dos: PULSE y DIR, se han demultiplexado utilizando el circuito integrado 74155N (Demultiplexor doble 2 a 4). Este circuito integrado tiene dos señales de control A y B que son las que permiten seleccionar hacia donde se van las señales. Como en este caso, solo se necesita direccionar las dos señales del CY545 hacia uno de los dos motores de pasos (el del eje X o Z), solo se necesitará una señal de control en el demultiplexor; por lo tanto, solo se utilizará la señal A, y la señal B se enviará a 0l. Esto quiere decir que a pesar de que el demultiplexor tiene cuatro

salidas, solo se utilizarán las dos primeras, una de ellas irá al eje X y la otra al eje Z.

Como el demultiplexor es doble, esto permite controlar con la primera parte la señal PULSE y con la segunda la señal DIR. La señal de control A del demultiplexor maneja las dos partes del mismo, a igual tiempo. Esta señal está conectada al User Bit 5 del CY545X, y por medio de software se controlará su estado.

El número de pasos, y principalmente la posición del eje X y del eje Z, será controlada por Software. Debe notarse que los CY545 mantienen en un registro interno su posición. Una vez que se utiliza el demultiplexor para direccionar los pulsos hacia el taladro, el CY545 comanda al motor del eje Z para que baje y suba, regresando a la misma posición inicial. Esto quiere decir que el registro de posición del CY545, al bajar el taladro, se incrementará; pero, al subir, se decrementará regresando al valor original. Esto permitirá que no se altere este registro de posición.

3.3.- Diseño de Seguridades y Protecciones.

Existe un nivel básico de seguridad en este sistema: el control de la posición Home (inicial), disponible en cada uno de los motores de paso, a través de la tarjeta de efecto de Campo.

El control de esta posición se lo realiza a través de una tarjeta diseñada especialmente para ello. Esta tarjeta permite a los motores de pasos conocer cuando el bloque deslizante se encuentra en su posición inicial. Para ello, se han colocado estas tarjetas en los dos postes destinados para el efecto, frente a los motores y en el camino del bloque deslizante. Esta tarjeta está formada por un regulador de voltaje 7805 que convierte el voltaje de 12 voltios entregado por la fuente a 5 voltios, una resistencia de 4.7 k, y un transistor de efecto de campo N119, el cual sensará el campo magnético del imán que se encuentra en el bloque deslizante.

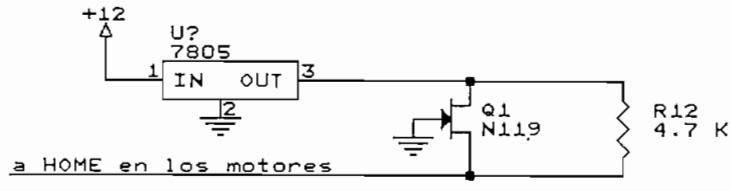


Figura 3.29

Como se puede observar en la figura 3.29, se ha conectado la salida de 5 Voltios del regulador a la entrada Source del transistor de efecto de campo. Gate se ha conectado a tierra, que es la tierra del sistema, y Drain se ha colocado como salida, la misma que irá como entrada a Home en el motor de pasos. Adicionalmente, se ha colocado un resistencia de 4.7 Kohmios entre Source y Drain.

Además, se ha colocado un magneto en el bloque deslizante. La posición de este magneto debe ser la correcta, puesto que uno de los lados produce efecto en el transistor, y el otro no.

De esta forma, mientras el magneto se encuentre lejos, la salida del transistor estará en 1l, con el transistor abierto. Cuando el magneto se encuentra cerca (bloque deslizante), el transistor se cierra y la salida se coloca a 0l, lo que indica que el bloque deslizante a llegado a su posición Home (Inicial). Una

vez que el motor ha detectado este cambio de nivel de voltaje en la señal HOME, detiene el motor. Debe anotarse que el motor no podrá seguir moviéndose en esa dirección, pero si podrá hacerlo en la dirección contraria. Esta constituye una característica importante de los motores de paso US-CYBERLAB.

La función de este sensor es triple, primero evita que el motor mueva el bloque de tal manera, que se produzca una colisión entre el bloque deslizante y el soporte del motor. Es necesario indicar que como el torque de este motor es alto, al producirse una colisión, podría sobrecalentarse y dañarse definitivamente.

La segunda función de este sensor, es permitir que al iniciar el trabajo, todos los motores vayan a la posición inicial, que será la posición Cero en todos los ejes.

La tercera función permite evitar que se pierdan +/- 3 pasos cuando los motores arrancan. La mayoría de los controladores arrancan con el mismo patrón de energización de las bobinas (fase), así que se puede perder acoplamiento con el motor debido a que este patrón no corresponde al estado actual del rotor del motor. Entonces el sensor de Home colocará el motor en la propia orientación de arranque.

En el caso del eje Y, puesto que dispone de dos motores, la salida Home de las tarjetas de efecto de campo se colocan a la entrada de una compuerta OR. De esta manera solo cuando las dos entradas sean 0L, la salida será 1L.

3.4.- Diseño del Interfase con el Microcomputador.

Como se ha dicho anteriormente, el CY545 no se comunica con el computador a través del pórtico serial, sino a través del pórtico paralelo; pero, no lo hace directamente. Para ello, se ha utilizado el Controlador Inteligente de Redes Locales CY-233, también fabricado por Cybernetic Micro Systems.

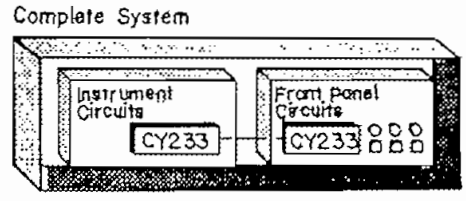
Antes de presentar el circuito diseñado para la comunicación con el microcomputador, se va a presentar un análisis de este circuito integrado.

3.4.1.- Controlador Inteligente de Redes Locales CY-233.

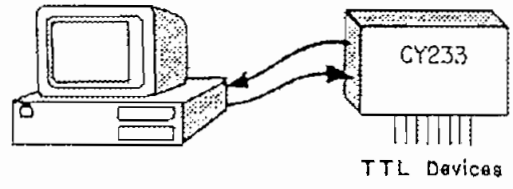
Inicialmente se podría pensar que el CY233 es un UART inteligente. La principal función de un UART es transferir la información entre dos sistemas separados y asincrónicos en tiempos multiplexados o con característica serial. El UART clásico generalmente maneja el flujo de datos entre dos sistemas, uno de los cuales es inteligente. El CY233, un UART inteligente, está diseñado para manejar el flujo de datos entre varios sistemas, uno de los cuales, cualquiera o todos pueden ser inteligentes.

El CY233 será discutido en términos de las siguientes características:

Intra-Sistema
(componente a
componente)



Extra-Sistema
(Sistema a
componentes)



Inter-Sistema
(Sistema a
sistema)

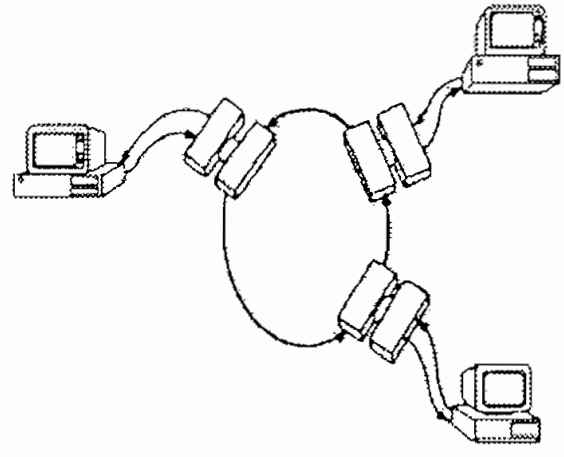


Figura 3.30 y 3.31

El circuito integrado CMOS CY-233 podría mantener un Host inteligente comunicándose con muchos dispositivos paralelos, locales, no inteligentes, tales como switches, LEDs, A-to-Ds, D-to-As, etc.

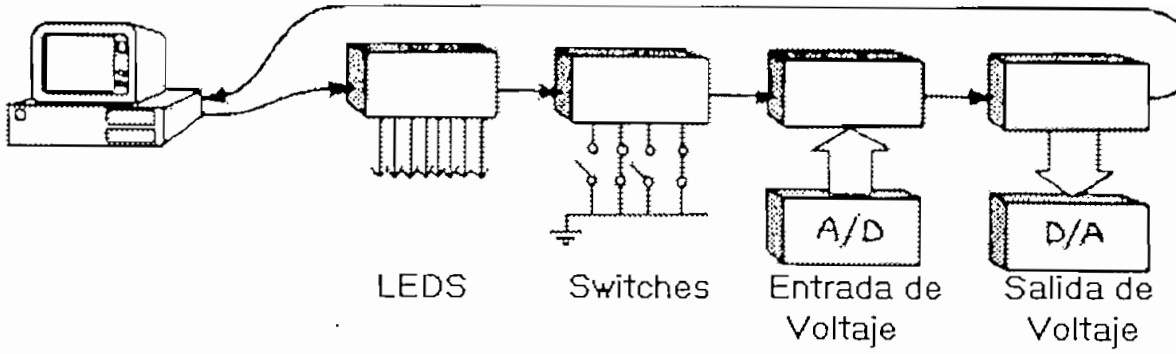


Figura 3.32

Adicionalmente el CY-233 también provee todas las funciones necesarias para soportar aplicaciones InterSistema tales como Redes de Area Locales (LAN), incluyendo aún el soporte para el sistema "Token".

El CY233 puede direccionar hasta a 255 dispositivos locales no inteligentes, y ha sido optimizado para aplicaciones en las cuales algunos o todos de los CY233s, en una red de anillo, son dispositivos inteligentes tales como micro computadores. El soporte primario para tales dispositivos locales inteligentes viene en forma de comandos de sistema y complejas transferencias paralelas adicionales. En particular, los dispositivos locales pueden sensar su propia identificación y la identidad de todos los otros dispositivos del anillo, y de esta manera iniciar una variedad de operaciones.

Finalmente, el CY-233 soporta "Token" por medio del manejo del tráfico en el anillo cuando el anillo está ocupado. Los dos tipos de redes de anillo soportados por el CY-233 se muestran a continuación:

CASO 1: Red de Anillo HOST - El Host está en el anillo.

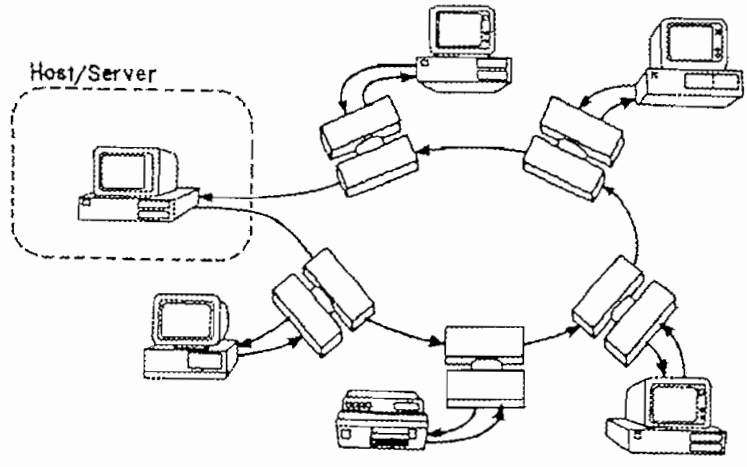


Figura 3.33

CASO 2: Redes anillo PUNTO a PUNTO - Todos los dispositivos están en el anillo

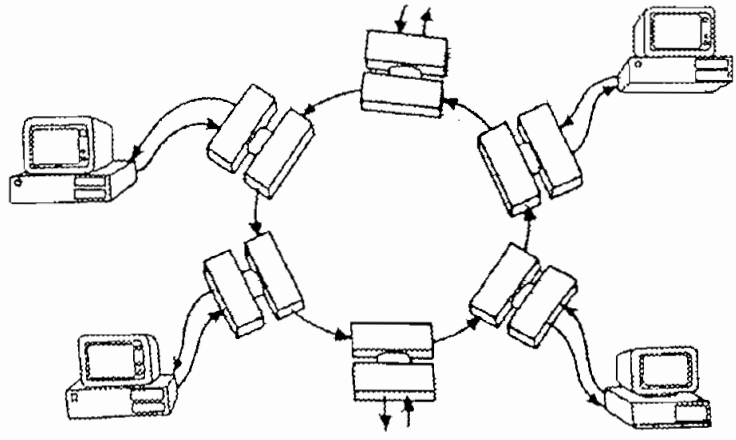


Figura 3.34

El CY-233 simplifica la comunicación de información digital entre sistemas y entre secciones de un sistema. Provee un interfase fácil de usar entre dispositivos seriales RS-232 y dispositivos paralelos compatibles TTL.

3.4.1.1.- Diagrama Lógico.

El Diagrama lógico CY-233 es el siguiente:

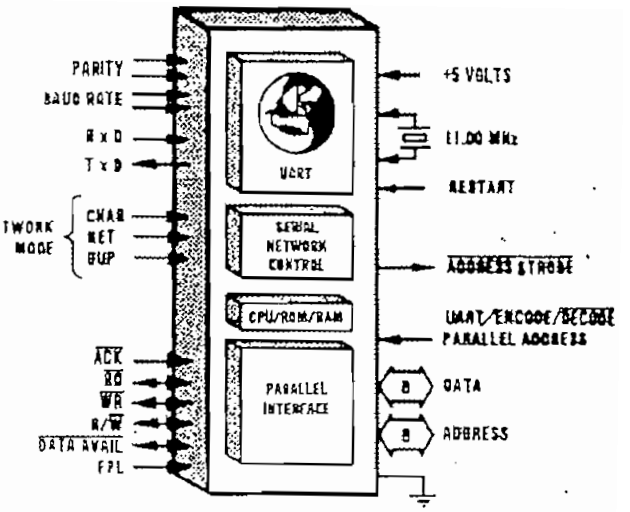


Figura 3.35

3.4.1.2.-Configuración de Pines del CY-233.

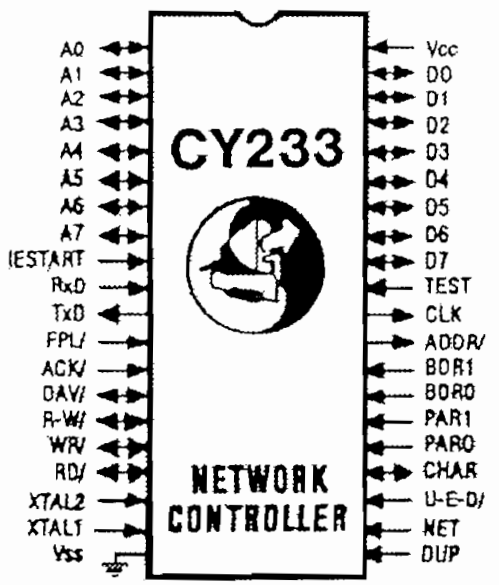


Figura 3.36

La configuración de pines del CY233 es la siguiente:

PIN	Mnemónico	Función
1	A0	Bit de dirección paralela 0 , LSB
2	A1	Bit de dirección paralela 1
3	A2	Bit de dirección paralela 2
4	A3	Bit de dirección paralela 3
5	A4	Bit de dirección paralela 4
6	A5	Bit de dirección paralela 5
7	A6	Bit de dirección paralela 6
8	A7	Bit de dirección paralela 7, MSB
9	RESTART	Reseteo por Hardware del CY233
10	RxD	Recepción de datos seriales en CY233
11	TxD	Transmisión de datos seriales desde el CY233
12	FPL/	Carga Paralela Forzada
13	ACK/	Conocimiento de Transferencia paralela
14	DAV/	Datos disponibles para dispositivo paralelo
15	R-W/	Selección Lectura/Escritura para transferencia
16	WR/	Strobe para escritura paralela o habilitación de bus
17	RD/	Strobe para lectura paralela o alter.
18	XTAL2	Conección del Cristal
19	XTAL1	Conección del Cristal
20	Vss	Tierra de la fuente de poder
21	DUP	Selección del modo Duplex
22	NET	Selección del modo de Red
23	U-E-D/	Selección del modo de codificación de direcciones
24	CHAR	Selección del modo de caracteres
25	PAR0	Selección de paridad y de la longitud de los datos, LSB
26	PAR1	Selección de la paridad y de longitud de los datos, MSB
27	BDR0	Selección de velocidad de transmisión LSB
28	BDR1	Selección de velocidad de transmisión MSB
29	ADDR/	Strobe direcciones para dispositivos paralelos
30	CLK	Salida de reloj, 1/6 de la frecuencia del cristal
31	TEST	Señal de test interno,conectado a VCC
32	D7	Bit de datos paralelo 7, MSB
33	D6	Bit de datos paralelo 6
34	D5	Bit de datos paralelo 5
35	D4	Bit de datos paralelo 4
36	D3	Bit de datos paralelo 3
37	D2	Bit de datos paralelo 2
38	D1	Bit de datos paralelo 1
39	D0	Bit de datos paralelo 0, LSB
40	Vcc	Fuente de poder de +5V.

3.4.1.3.- Descripción de las señales.

A continuación, se van a describir la función de los pines listados en el numeral anterior. En el caso de los niveles de señales, una F indicará que la señal es flotante, y que puede ser enviada a alto o a bajo por el CY-233; un "1" indica un nivel lógico alto TTL que no puede ser enviada a bajo, y un "0" indica un nivel lógico TTL bajo que no puede ser conducido a alto. También, "E" indica que la señal es de entrada, "S" indica que la señal es de salida, y "E/S" indica que el pin es bidireccional. Finalmente, un nombre seguido por "/", como RD/, indica una señal activa en bajo.

- VCC (E) Pin 40: Fuente de poder de +5V.

- VSS (E) Pin 20: Tierra de la fuente de poder.

- XTAL (E) Pin 18,19: Entradas de cristal o del reloj,estas señales no son TTL. El Rango de operación Standard es de 3.5 a 12 MHz, con frecuencias de 11.059 MHz (11 Mhz funciona bien) o 7.3728 MHz para velocidades de transmisión standard.

- CLK (S) Pin 30: Señal de reloj divida para 6. El ancho del pulso es de por lo menos 127 nseg.

- RESTART (E) Pin 9 Activa en alto. El CY233 reseteará todos los modos de operación internos y testeará los seteos de las líneas de Entrada/Salida cuando está señal es pulsada. Esta señal debe estar activa cuando la energía es aplicada.

- TEST (E) Pin 31: Esta es una señal de Test interno que no es usada para operaciones normales. Debe ser conectada a Vcc o a un nivel lógico alto.

- D0-D7 (E/S) Pin 32-39: Bus de datos bidireccional para transferencias paralelas de datos. Estas líneas son de colector abierto y deben ser conectadas a resistencias de Pullup de 10K, la que están conectadas a VCC. Los bytes de datos son transferidos entre el CY233 y el dispositivo paralelo en estas líneas. Las funciones de tiempo y latching difieren con cada modo de operación. Se soportan dos funciones de transferencia básicas, un modo strobe, usando los strobes RD/ y WR/, un modo de handshake,

usando DAV/, ACK/, R-W/, y Bus Enable.

- RD/ (E/S) Pin 17:

Strobe de Lectura/. Strobe para la transferencia de datos desde el dispositivo paralelo hacia el CY233.

F Modo de Strobe habilitado.

Si esta línea se la deja flotante, para que el CY233 pueda conducirla, se habilita el modo de transferencia de datos strobe. En este modo los datos serán leídos en el flanco de subida de RD/. Esta señal puede ser usada para habilitar el driver de salida de tres estados para poner datos en el bus.

0 Modo Strobe deshabilitado, WR/ es BUSEN/

Si RD/ se envía a bajo, el modo de transferencia strobe es deshabilitado, y WR/ asume una función de habilitación del bus. Una versión lógica más baja de RD/ puede ser creada en este modo, combinando DAV/ con R-W/. El CY233 revisa RD/ antes de que se intente alguna transferencia paralela, y así es posible cambiar dinámicamente los modos entre el modo strobe y el modo no-strobe.

1 Modo especial de Red de Area Local.

Si RD/ es enviada a alto, el CY233 entra en el modo de LAN. En este modo, dos CY233 pueden ser conectados a través de sus interfases paralelos. Uno de ellos actúa como un CY233 normal, conectándose a la red de otros CY233, y con todas las operaciones de red disponibles. El otro actúa como un dispositivo especial de red.

La línea RD/ es revisada por el modo LAN solamente en el inicio. Si no se ha enviado la línea a un nivel alto en ese momento, el CY233 entra en el modo UART o en el modo de Red. Estos modos se describirán posteriormente.

- WR/ (E/S) Pin 16:

Strobe de Escritura/. Cuando el modo de transferencia strobe se ha habilitado (RD/ flotando), esta señal indica que la salida de datos por el CY233 es válida. El CY233 conduce el bus datos justo antes generar la señal de strobe WR/. Los datos son válidos en los flancos, y son generalmente retenidos en un latch

hacia un dispositivo de memoria externa en el flanco de subida del strobe. WR/ es siempre una salida desde el CY233 en el modo de transferencia Strobe.

Cuando el modo strobe es deshabilitado, esta señal se convierte en una señal de entrada que habilita el bus BUSEN/. Esta señal permite al dispositivo paralelo indicar cuando el CY233 puede conducir el bus de datos. Si el CY233 tiene un byte de datos para transferirlo, generará DAV/, pero no conducirá el bus de datos a menos que la señal BUSEN/ esté en bajo. Si BUSEN/ está en alto, el CY233 esperará a que vaya a un nivel lógico bajo antes de poner los datos en las líneas de bus. Los datos serán removidos nuevamente cuando la señal BUSEN/ se la envía a alto nuevamente. Si deja BUSEN/ en bajo o se la envía a bajo, los datos permanecerán retenidos (por el Latch) en el bus de datos hasta que otro dato sea escrito, o el CY233 ejecute una función de lectura de datos. Esta acción de retención puede extenderse mientras dure el mensaje serial, y es independiente de la función de handshake DAV/ ACK/. En este modo una versión lógica más baja del strobe WR/ puede ser recreada combinando DAV con R-W/. La función BUSEN/ trata a esta señal como una señal de entrada solamente para el modo no-strobe.

Adicionalmente, si se habilita el comando E (Habilitación de Error), la señal WR/ indica al dispositivo local paralelo que un error, incluyendo paridad, buffer overflow, o timeout, ha sido detectado por el CY233. Cualquier mensaje recibido, removerá el error hasta que se termine la operación pendiente.

Cuando se indica un error, el dispositivo local paralelo puede leer el byte de status de errores desde el CY233, usando un protocolo de Handshake modificado. El byte de status de un error paralelo tiene la siguiente definición. Un bit que está en alto indica que se detectó un error.

DB0 Error de paridad en la recepción.
DB1 Buffer overflow en la recepción.
DB2 Formato de comando Inválida.
DB3 Fuera de tiempo en el handshake de escritura paralelo.
DB4 Fuera de tiempo en el handshake de lectura paralelo.
DB5 Fuera de tiempo en el handshake paralelo en el modo FPL.
DB6 El caracter recibido no fue el caracter de retorno en el seteo de la velocidad de transmisión adaptiva.
DB7 Reservado.

Finalmente, la línea WR/ tiene una función especial en el modo ALN. Cuando la dirección del nodo local se aplicaría al CY233 de red, el CY233 LAN conducirá la línea WR/ a un nivel bajo. Esto habilitaría un buffer de tres estados que conduce las líneas de dirección de los dos CY233. Cuando la línea WR/ esté en alto, el CY233 LAN generará la dirección para el CY233 de red (Network), así el nodo de dirección externamente bufferado sería de tres estados. La otras funciones de WR/ son deshabilitadas en el modo de LAN.

- DAV/(E/S)Pin 14:

Datos Disponibles/. Señal de Handshake para controlar la transferencia de datos entre el CY233 y un dispositivo paralelo. Normalmente se la usa en combinación con ACK/ y R-W/. Si no se usa handshake, DAV/ y ACK/ deberían ser manejadas juntas, dando al CY233 una handshake automático para cada transferencia. Esta señal es generada en los dos modos: modo strobe y modo de transferencia por handshake.

Cuando se habilita el modo Strobe (RD/ flotante), DAV/ tiene las dos funciones juntas: la función de datos disponibles y la función de Bus Enable. En este caso, la señal flotante DAV/ automáticamente llevará los datos cuando el CY233 tenga datos para transferirlos, y retirará los datos del bus cuando DAV/ sea retirado. Una ligera modificación de la función de Bus enable es este modo, permite que el CY233 conduzca el bus de datos antes de generar la señal DAV/ (bajo), y eso permite que los datos sean válidos en los dos flancos de DAV/. De hecho, se genera DAV/ inmediatamente después de que ocurre el Strobe WR/. Si DAV/ es enviado externamente a bajo, el CY233 retendrá y mantendrá los datos en el bus entre transferencias de escritura, similar a mantener Bus enable en bajo cuando el modo de transferencia strobe es deshabilitado. Los datos se mantendrán mientras DAV/ esté en un nivel lógico bajo, o hasta que el CY233 ejecute una transferencia de lectura.

Cuando el CY233 ejecuta una operación de lectura en el modo strobe, DAV/ irá a un nivel lógico bajo para indicar que debería ocurrir una lectura. La señal R-W/ irá a alto, indicando una operación de lectura. El hecho de DAV/ va a un nivel lógico bajo indica al dispositivo paralelo que puede conducir el bus de datos, y que los datos serán leídos por el CY233 cuando ocurra

la señal de strobe RD/.

Para una transferencia de escritura con nostrobe, DAV/ tiene únicamente la función de Datos Disponibles, y WR/ asume la función de Bus Enable. En este modo, DAV/ indica que el CY233 podría transferir datos al dispositivo paralelo, pero el bus de datos no es conducido hasta que BUSEN/ esté en bajo. DAV/ será removido cuando el dispositivo paralelo tenga conocimiento de la transferencia por medio de la señal ACK/.

Para una transferencia de lectura no-strobe, DAV/ también va a bajo para indicar que el CY233 está requiriendo una operación de lectura. El dispositivo paralelo puede conducir el bus de datos mientras DAV/ esté en bajo. El CY233 esperará por ACK/, entonces realizará la lectura justamente después de que DAV/ ha sido enviado a alto nuevamente. En esta operación se ignora la señal BUSEN/, y puede estar en alto o bajo durante la misma. Si el CY233 tiene su bus de datos en el modo de salida retenido (latched), se cambiará la impedancia del bus a alto cuando el CY233 conozca que debería ocurrir una transferencia de lectura, antes de que la señal DAV/ sea generada.

Un ejemplo de tiempo fuera para el handshake de escritura es el siguiente:

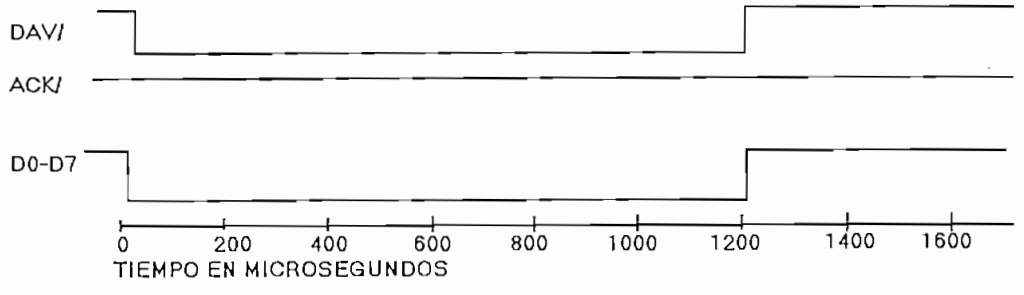


Figura 3.37

A continuación se muestra un ejemplo de tiempos para un mensaje completo para escritura paralela.

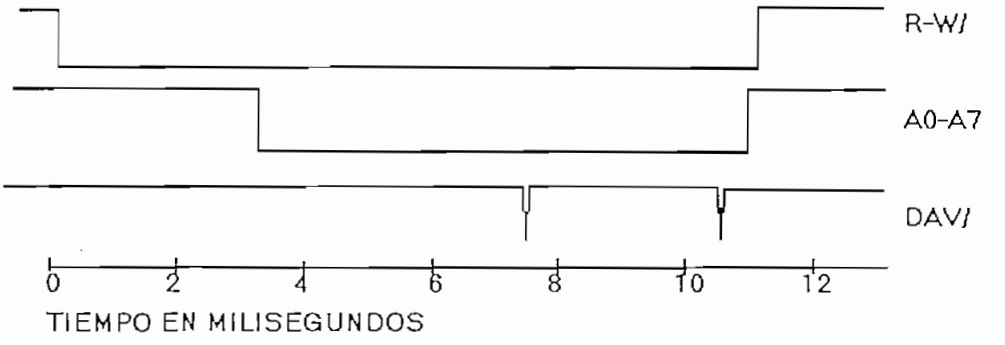


Figura 3.38

A continuación observaremos el handshake de transferencia para una escritura en el modo de strobe:

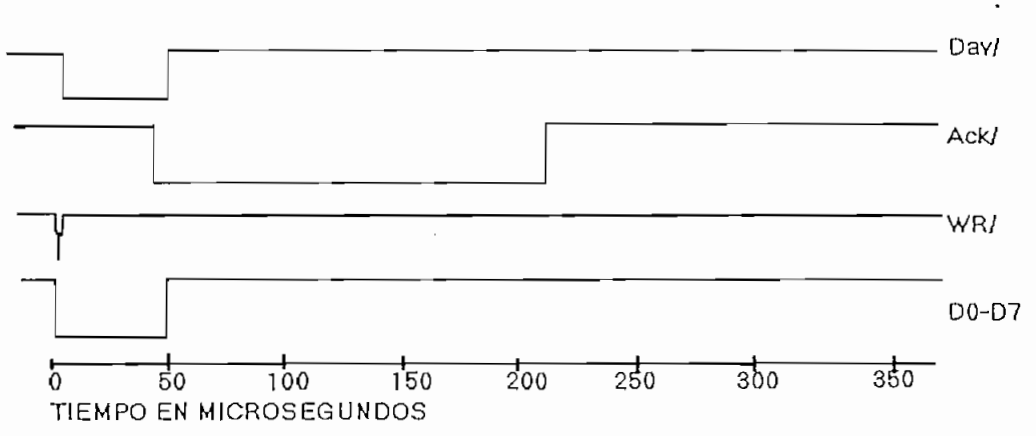


Figura 3.39

Tiempos para la escritura en el modo strobe:

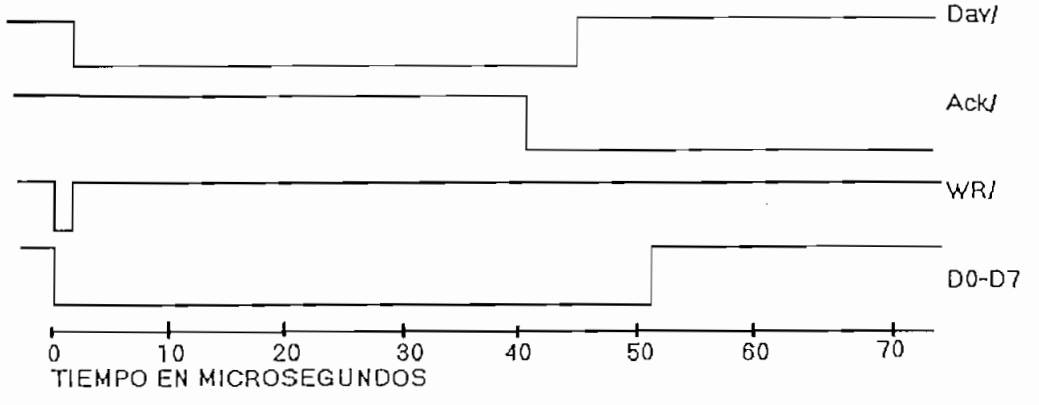


Figura 3.40

Detalles de tiempos para la escritura en el modo strobe:

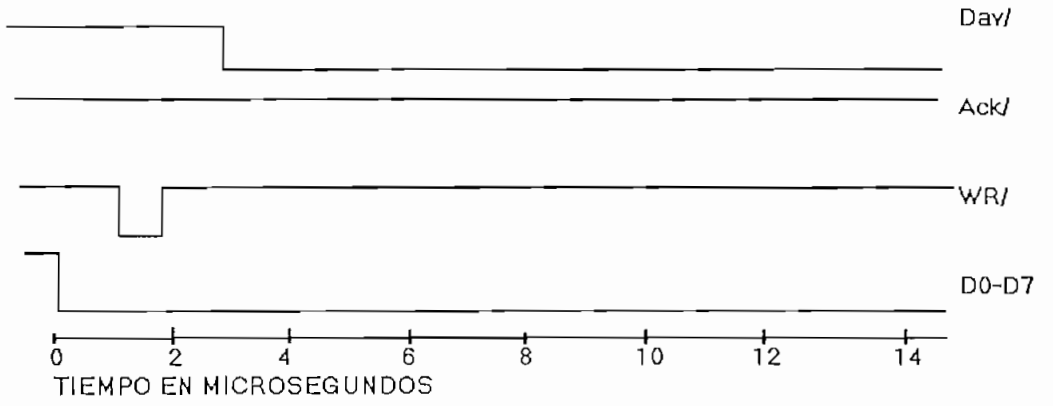


Figura 3.41

Handshake de transferencia para la lectura en el modo strobe.

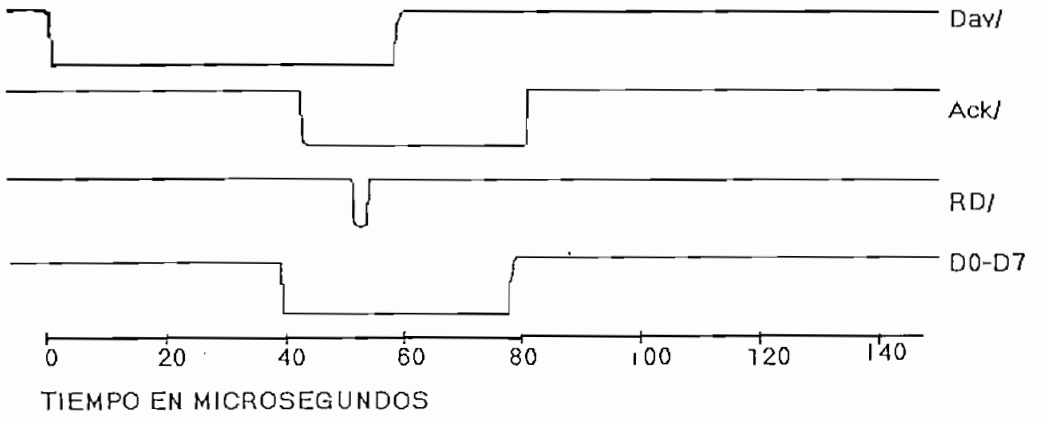


Figura 3.42

Detalles de tiempo para la lectura en el modo strobe:

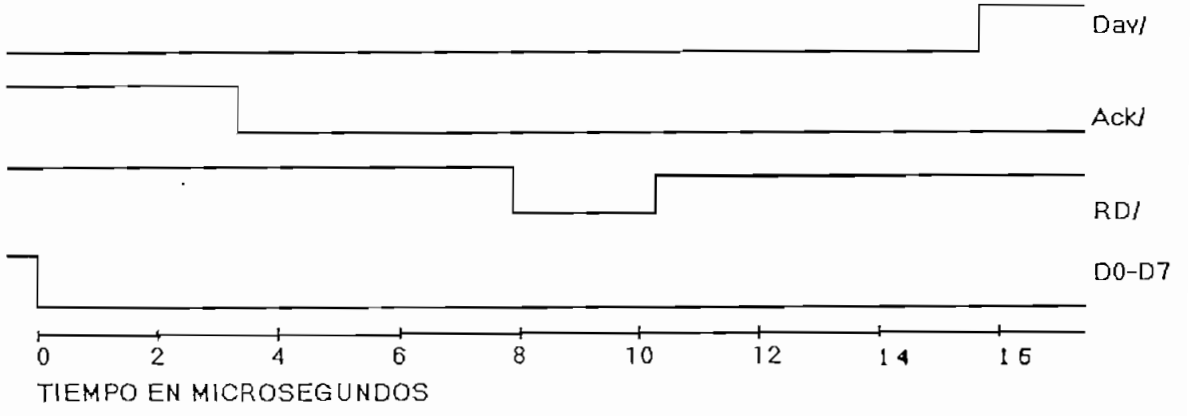


Figura 3.43

Handshake de transferencia para la escritura en el modo no-strobe:

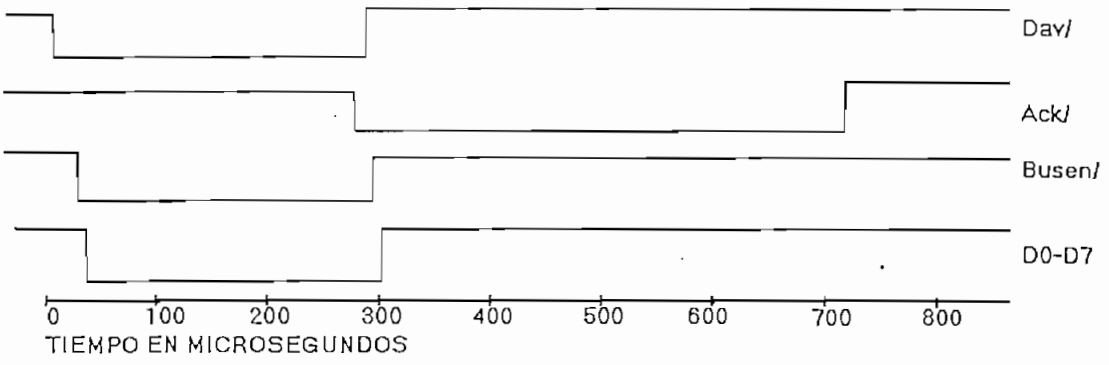


Figura 3.44

Escritura en el modo Strobe con el bus de datos retenido (Latched), BUSEN/ Low:

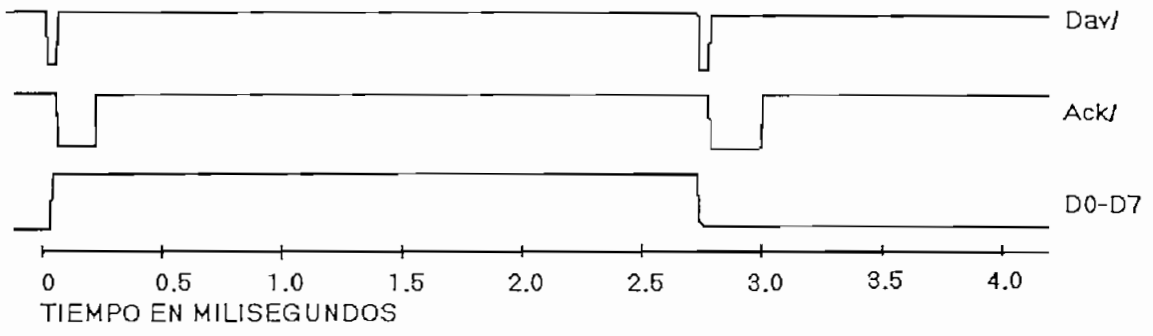


Figura 3.45

Lectura en el modo no-strobe con el bus retenido:

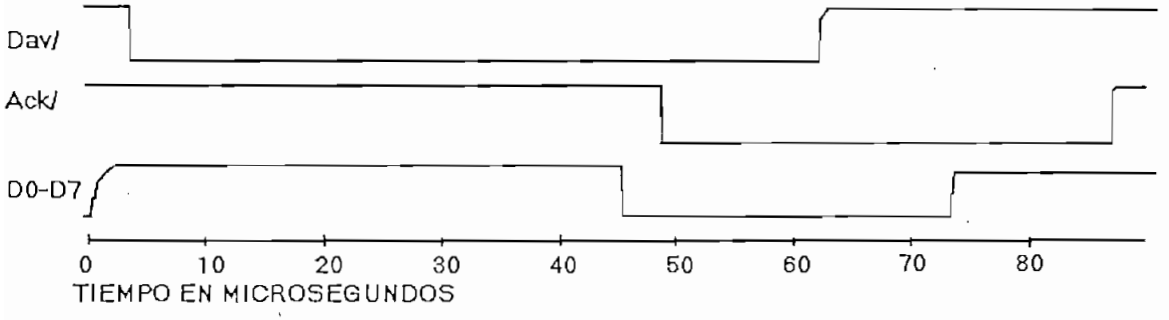


Figura 3.46

- ACK/ (E) Pin 13:

Conocimiento/. Señal de Handshake utilizada para controlar la transferencia de datos entre el CY233 y un dispositivo paralelo. Esta línea es normalmente usada en combinación con DAV/ y R-W/.

Primero se chequea está línea como parte de una función de chequeo de direcciones paralelas. Cuando el CY-233 genera una dirección al comienzo de una transferencia paralela, esta línea debe estar en alto para permitir que la transferencia continúe. Si ACK/ está en bajo, cualquier dirección para el CY233 se tratará como inválida. Este Test ocurre después de que el CY233 ha chequeado las líneas de dirección para validarlas.

Si la dirección es calificada como válida, la transferencia continúa, y el CY233 generará la señal R-W/, conduciendo el bus de datos, y llevando a bajo la señal DAV/. También se generarán transferencias de escritura strobe, con la señal de strobe WR/. El CY233 esperará entonces en este estado hasta que el dispositivo paralelo conduzca la señal ACK/ a bajo, indicando que la transferencia de escritura ha sido aceptado por el dispositivo. Cuando ACK/ va a bajo, el CY233 lleva a DAV/ a alto, y remueve los datos del bus, a menos de que la función de retención (Latching) esté habilitada.

Para transferencias de lectura, el CY233 limpia el bus de datos antes de llevar la señal DAV/ a bajo. Entonces espera a que ACK/ vaya a bajo, indicando que el dispositivo paralelo está conduciendo las líneas de datos para la transferencia. El CY233 lee entonces los datos, generando el strobe RD/ en el modo Strobe, y lleva la señal DAV/ a alto nuevamente, indicando que la transferencia está completa.

Note que el dispositivo paralelo puede controlar la duración de la transferencia de datos controlando el tiempo de la señal ACK/. Sin embargo, el CY233 tiene un límite máximo de tiempo en el handshake y continuará con la transferencia si la señal ACK/ no es recibida dentro del límite de tiempo. El límite de tiempo es fijado a un valor default, relacionado con la velocidad de transmisión del CY-233, cuando el dispositivo es reinicializado. Un comando especial de la red permite deshabilitar esta función de tiempo.

Si el dispositivo paralelo solamente necesita los strobes RD/ y

WR/, ACK/ deberá se enviada a DAV/, la cual permite que el CY233 controle automáticamente el protocolo de handshake. Un ejemplo de Chequeo de direcciones válidas es el siguiente:

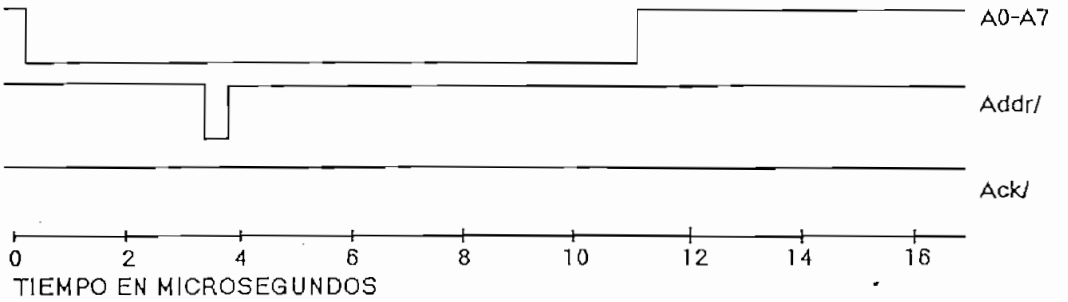


Figura 3.47

En la figura 3.47 se observa un chequeo de la dirección local sin que las líneas de dirección alcancen la dirección solicitada (Dirección no válida).

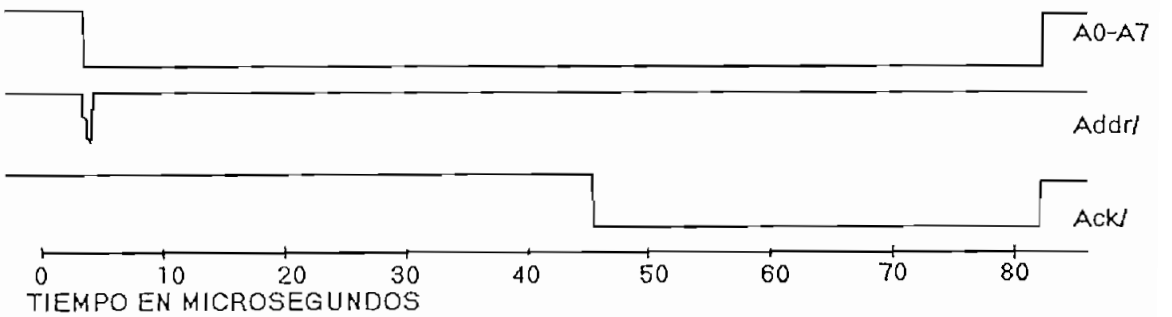


Figura 3.48

En la figura 3.48 se observa el diagrama de tiempo para el chequeo de la dirección local con ACK/ en bajo.
(Dirección inválida).

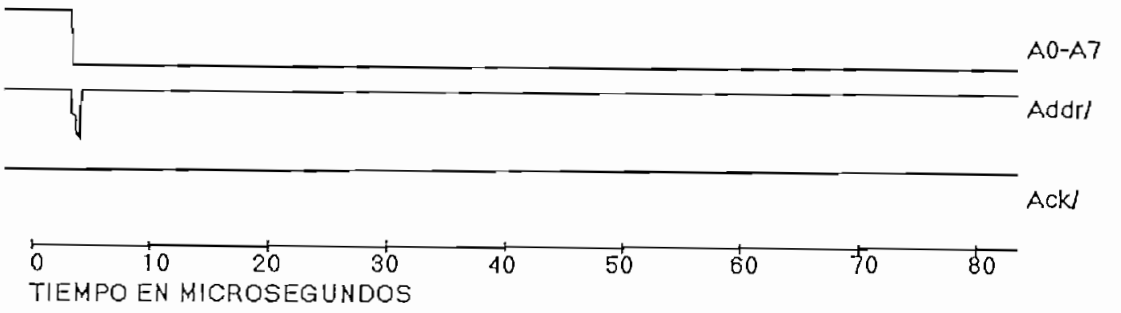


Figura 3.49

En la figura 3.49 se observa el diagrama de tiempo correspondiente al chequeo de la dirección local con resultados positivos y ACK/ en alto. (Dirección Válida).

- R-W/(E/S)Pin 15:

Lectura Escritura/. Esta línea normalmente indica la dirección de la transferencia de datos iniciada por el CY233. Cuando está en alto, el CY-233 está ejecutando una lectura, y cuando está en bajo, el CY233 estará ejecutando una escritura. La combinación lógica de estas señales con DAV/ puede recrear versiones más bajas de RD/ y WR/ cuando se necesitan strobes separados, pero el modo strobe está deshabilitado.
Cuando el dispositivo paralelo inicia una operación de lectura poniendo al CY-233 en el modo Master, o usando la señal de carga forzada paralela (FPL/), el CY233 chequeará el estado de la señal R-W/, la cual puede estar conducida por el dispositivo paralelo. Si R-W/ está en alto, el CY233 genera un mensaje de lectura para la transferencia, y si R-W/ está en bajo, se generará un mensaje de escritura. Esta segunda opción le permite a un dispositivo paralelo escribir datos a otro módulo CY233 en la red. El estado de la línea R-W/ es chequeado durante la primera lectura de datos, cuando DAV/ y ACK/ están en bajo.

- FPL/ (E) Pin 12:

Carga Forzada Paralela/. Esta señal permite al dispositivo paralelo entrar una secuencia de bytes de datos al CY233, sin

cambiarlo al modo master, y sin que el CY233 busque a través de las direcciones buscando una válida. La dirección usada en el mensaje de lectura o escritura resultante es la dirección que estaba siendo utilizada en la línea de direcciones cuando el requerimiento de carga fue detectado.

El ahorro de este chequeo de direcciones hace que esta operación sea más rápida que la lectura en el modo Master, y permite que el dispositivo paralelo ingrese mas de un byte de datos por mensaje.

El modo FPL se inicia por medio de un flanco de caída en la línea FPL/. Este flanco será sensado y grabado en el CY233 cuando se produzca, pero la transferencia de datos ocurrirá únicamente cuando el CY233 esté desocupado, sin ningún mensaje serial transmitido o recibido.

Cuando se detecta un flanco de caída y el CY233 está desocupado, se intentará una lectura de datos. El CY233 no conduce las líneas de direcciones, pero se genera el strobe ADDR/ como una señal para lógica externa que puede ser aplicada a FPL. Entonces, se ejecuta una lectura normal de datos, con la utilización de las líneas normales de handshake. Durante la primera lectura de datos, también se leerán las direcciones manejadas externamente, y se chequeará el estado de la línea R-W/. Estos eventos ocurren cuando las dos líneas DAV/ y ACK/ están en un nivel lógico bajo. Un mensaje de respuesta se inicia después de estos datos, con la lectura o escritura determinada por el estado de la línea R-W/, y utilizando las direcciones del dispositivo generados desde esa lectura por el CY233, a menos que la dirección leída tenga todas las líneas en alto (0FFh). En este caso, el CY233 considera los datos paralelos como un comando paralelo especial, y ejecuta la acción requerida. Los posibles comandos se describen posteriormente.

Si el dispositivo paralelo mantiene la línea FPL/ en bajo, después de la primera lectura de datos, se leerán bytes de datos adicionales, cada uno con el handshake respectivo. Estos bytes llegarán a ser parte del mismo mensaje iniciado desde la lectura del primer dato.

Existen dos maneras de terminar una operación de lectura en el modo FPL. Primero, si se lee el byte terminador del mensaje

desde el dispositivo paralelo; entonces terminará el modo FPL, independientemente del estado de esta línea FPL/. Para continuar leyendo bytes en el modo FPL, el dispositivo paralelo deberá bajar de nuevo la línea FPL/, logrando que nuevamente se detecte un flanco de caída. Esto iniciará un nuevo mensaje en modo FPL. Este mecanismo no trabaja si el CY233 está en el modo de caracteres ASCII HEX. Si el CY233 lee un patrón 0Dh en este modo, los datos son enviados como dos caracteres hex, "0" y "D", y no se acabará el mensaje.

La segunda manera de finalizar la lectura en el modo FPL/ es levantar la señal FPL/ durante la lectura de una transferencia de un byte. Si el último byte leído no fue un byte de terminación de mensaje, el CY233 automáticamente añadirá un terminador, completando el mensaje para la red. En el modo HEX, siempre se añade el terminador al mensaje, ya que todos los datos son enviados como caracteres de dos datos HEX.

Nótese que la simple transferencia de un byte puede ser ejecutada utilizando la línea FPL/, puesto que el flanco de caída es grabado internamente por el CY233. Cuando el CY233 está desocupado, lee el byte, y genera un mensaje alrededor de él. La transferencia de múltiples bytes de datos requiere de que el dispositivo paralelo mantenga la línea FPL/ en bajo hasta que se hayan leído todos los bytes. Este modo se termina cuando se lee un terminador o cuando se lleva a alto la señal FPL/. Note que dispositivos paralelos lentos, o transferencias de gran número de bytes de datos en un mensaje FPL, podrían causar problemas en una red congestionada, llevando al CY233 a un modo de lectura FPL, y manteniéndolo en ese modo mientras se recibe un mensaje serial desde otro nodo. Esto eventualmente sobrecargará los buffers de recepción serial del CY233, causando por lo tanto pérdidas de datos en la red. Los mensajes en el modo FPL deben ser lo más pequeños como sea posible en una red ocupada.

- U-E-D/(E)Pin 23:

UART decodificado y codificado/. Modo de selección para el direccionamiento del dispositivo paralelo. El estado de esta línea indica la forma como el CY-233 deberá tratar los valores de las líneas de direcciones. Así:

1 Modo UART no codificado, no se usan las direcciones.
F Direcciones codificadas, lógica binaria positiva.
0 Direcciones decodificadas, lógica binaria negativa, 1 a 8.

El modo no codificado UART es chequeado en la inicialización, y deshabilita todo comando especial y característica de mensaje del CY233. En este modo, el CY233 trabaja como un simple UART, transfiriendo datos entre los lados seriales y paralelos. Sin embargo, este dispositivo distingue entre los modos de caracteres ASCII, HEX o Binarios, con dos caracteres seriales requeridos para una transferencia HEX, y solamente un caracter serial requerido para una transferencia paralela ASCII o Binaria. También el modo MASTER y el modo FPL trabajarán en el modo UART, en donde las respuestas generadas por el CY233 contendrán datos solamente, sin las porciones del comando normal, direcciones y terminador de un mensaje completo.
Los modos codificados y decodificados son chequeados cada vez que el CY233 intenta generar una nueva dirección; por lo tanto, estos modos pueden ser cambiados durante los mensajes.

- A0-A7(E/S)Pin1-8: Líneas de direcciones. Estas líneas especifican la dirección actual del dispositivo paralelo.
En el modo UART, estas líneas no son usadas, y los datos se transfieren entre el lado serial y el lado paralelo sin ningún protocolo de direcciones o estructura de mensaje. Esto permite usar el CY233 como un simple controlador de aplicaciones donde no se necesitan múltiples dispositivos o mensajes seriales. Internamente, el CY-233 trata a todos los bytes de datos como si tuvieran una dirección válida, pero la dirección nunca es generada en la línea de direcciones, o incluida en un mensaje. También, no se necesitan los terminadores de los mensajes y no serán generados por el CY-233.
En el modo codificado, las líneas de direcciones representan una dirección de lógica binaria positiva. Direcciones válidas son 0 a 254. La dirección 255 no debe usarse, ya que las líneas de direcciones se llevan a un nivel lógico alto entre mensajes.
En el modo decodificado, las líneas de direcciones desempeñan la función de líneas de selección de lógica negativa, con

direcciones válidas del 0 al 7. Si los mensajes recibidos contienen direcciones mas altas que 7, serán interpretadas como módulo 8; así, las direcciones 2 y 10 generarán el mismo patrón de direcciones. Cuando ningún dispositivo está siendo direccionado, todas las líneas de dirección estarán en alto. Cuando se recibe un mensaje, o se inicia una transferencia paralela, una sola línea de direcciones se enviará a un nivel lógico bajo, correspondiente a una de las direcciones 0 a 7, y puede ser usada como una señal de selección de dispositivos.

En los modos codificados y decodificados, se chequea la validez de las direcciones antes de que se habilite la transferencia paralela. Se deben satisfacer dos condiciones para que una dirección sea considerada como válida. Primero, el CY233 debe ser capaz de conducir las líneas de direcciones a un estado especificado por esas direcciones. Esto es chequeado por el CY233 escribiendo la dirección deseada en la línea de direcciones, generando el strobe ADDR/, entonces se lee el estado actual de estas líneas de direcciones. Este estado es comparado con las direcciones deseadas, y se pasará el primer chequeo de validación si las dos son las mismas.

Cualquier línea de dirección enviada a 1 o 0 puede diferir del patrón que el CY233 está intentando generar, restringiendo las posibles direcciones que el CY233 puede generar, mientras que cualquier línea que se ha dejado flotante siempre alcanzará el estado que el CY233 está intentando generar.

El segundo chequeo de direcciones válidas ocurre después de que las líneas de dirección son chequeadas. Cuando se encuentra un patrón de direcciones válido, también se chequeará la línea ACK/, y si esta línea está en un nivel lógico bajo, la dirección se considerará inválida; pero, si la línea ACK/ está en alto, la dirección es válida, y se habilitará la transferencia paralela. La línea ACK/ se chequeará aproximadamente 70useg. después de que se chequearon las líneas de direcciones, permitiendo que lógica externa pueda fijar la línea ACK/ antes que sea chequeada.

Los chequeos de validez se hacen para cada mensaje enviado o recibido por el CY233, lo que permite que lógica externa pueda cambiar dinámicamente el seteo de las direcciones para

cualquier CY233.

- ADDR/ (S) Pin 29:

Strobe de direcciones/. Este strobe es generado después que el CY233 haya escrito una nueva dirección a las líneas A0 a A7, pero antes de que las líneas hayan sido leídas para ver si son comparadas con las direcciones deseadas. El Strobe puede ser utilizado para disparar lógica externa, indicando cuando una dirección generada externamente puede ser cambiada. Este cambio influirá en el chequeo de validez para las direcciones que están siendo generadas. Si no se necesita ADDR/ en alguna aplicación, puede ser dejada flotante.

- RxD (E) Pin 10:

Datos seriales recibidos. Esta señal del CY233 recibe los bytes de datos seriales provenientes de la red.
Esta es una entrada con niveles TTL, con una condición de MARCA, o un valor binario 1 representado como un nivel lógico alto TTL, una condición de ESPACIO, o un valor binario 0 representado como un nivel lógico TTL bajo. Los voltajes Standard RS-232 no son compatibles TTL. Para interfases RS-232, una MARCA es un voltaje negativo, un ESPACIO es un voltaje positivo. Para conectar el CY233 a una red RS-232 se requiere de drivers y receivers de línea externos.

- TxD (S) Pin 11:

Datos seriales transmitidos. Esta señal envía los bytes de datos seriales desde el CY233 a la red.
Esta señal es también de nivel TTL, y requiere de un driver externo para convertir los niveles de voltaje TTL en RS-232. Los tiempos y los niveles lógicos son del mismo tipo que la señal RxD.

- BDR0,BDR1 (E)Pin 27-28

Selección de la velocidad de transmisión. Estas líneas seleccionan la velocidad de transmisión para RxD y TxD. El receptor y el transmisor corren a la misma velocidad de transmisión. La velocidad de transmisión se fija en la reinicialización.
Un juego de velocidades de transmisión se define con la frecuencia standard de operación de 11.059 MHz. La frecuencia más alta, 57600 baudios, no es standard con un reloj de 11.059 MHz, y debería solamente usarse en las redes que no usan un

sistema de Host. Un reloj de 11.00 MHz trabajará bien en lugar del de 11.059 MHz, puesto que el error es de solo 0.6%.

Si se cambia la frecuencia de cristal del CY233 a 7.3728 MHz, se alcanzará una velocidad de 38400 baudios.

Cuando se escoge el modo adaptivo, el CY233 ajustará la velocidad de transmisión para que alcance la que está siendo usada por la red. Esto permite usar una frecuencia de cristal no standard o usar una diferente velocidad de transmisión, así como permitir que el sistema Host determine la velocidad de transmisión en la red. La resolución del contador interno del CY-233 limitará las velocidades de transmisión a los rangos soportados por las velocidades fijas del CY233.

En el modo adaptivo, el sistema Host debe enviar dos caracteres de retorno al CY233 cuando se encienden o cuando se resetean, lo que permite que el CY-233 ajuste la velocidad de transmisión interna para alcanzar la de los caracteres recibidos. Si se han escogido los modos Echo All o Echo invalid, el CY233 repetirá los dos caracteres de retorno después de que los dos han sido recibidos; así, todas las estaciones en un anillo podrán ser inicializadas. Estos caracteres de retorno se deben enviar antes de que comience cualquier operación normal en la red.

Las siguientes velocidades están disponibles en el CY233, usando hardware interno como una fuente de tiempo. La columna próxima a la velocidad de transmisión es el divisor de reloj usado para conseguir esta velocidad.

BDR1,0	Velocidad a 11.059 MHz		Velocidad a 7.3728 Mhz
F F	ADAPTIVA		
F 1	1	57600	38400
F 0	3	19200	12800*
1 F	6	9600	6400*
1 1	12	4800	3200*
1 0	24	2400	1600*
0 F	48	1200	800*
0 1	96	600	400*
0 0	192	300	200*

* Estas son velocidades no standard a 7.3728 Mhz

- *PAR0, PAR1* (E) Pin 25,26

Selección de paridad y longitud de los datos. Estas señales definen la paridad y la longitud de los datos, como se muestra en la tabla siguiente. Estos seteos son válidos para todos los modos de caracteres, pero el modo binario debe usar una de las opciones con 8 bits de datos para conducir todas las líneas de datos y recibir el terminador, 0B3h. La longitud de los datos y la paridad se fijan en la Reinicialización del CY233.

Cuando se especifica la paridad, se debe incluir el bit de paridad en los caracteres recibidos. Mientras 7 bits de datos y paridad de marca son equivalentes a 7 bits de datos, no paridad, y dos bits de parada, no podría ser usada con 7 bits de datos, no paridad, y un bit de parada, ya que el CY233 revisa tanto el bit de paridad (siempre marca) y el bit de parada. Un sistema Host enviando 7 bits de datos, no paridad, y un bit de parada, podría enviar el próximo caracter antes de que el CY233 estuviese listo para ello. Debe notarse que la longitud total de caracteres es: Bit de inicio + bits de datos + paridad, si se usa, + bit de parada.

PAR 1,0	PARIDAD	LONGITUD DE DATOS	LONGITUD TOTAL DE LOS CARACTERES
F F	Marca	7	10
F 1	Par	7	10
F 0	Impar	7	10
1 F	Espacio	7	10
1 1	Ninguna	8	10
1 0	Espacio	8	11
0 F	Marca	8	11
0 1	Par	8	11
0 0	Impar	8	11

- CHAR (E/S)Pin 24:

Caracter. Esta señal selecciona el tipo de caracteres usado por el lado serial para representar los bytes de datos de la transferencia paralela. Este modo de caracteres se determina en la Reinicialización.

Una vez que el CY233 está operando, esta señal se convierte en una salida. El CY233 intentará conducirla a un nivel lógico bajo mientras se está transmitiendo, excepto si la línea está en alto; en este caso el CY233 no podrá conducirla a bajo. Esta señal puede ser usada para habilitar drivers de tres estados, como los usados en las aplicaciones de RS-449, y permite múltiples transmisiones combinadas en una sola línea.

1 Modo de Caracteres ASCII HEX

Dos caracteres ASCII se usan para representar un valor de 8 bits de datos. El primer caracter es la parte alta, y el segundo caracter es la parte baja. Caracteres válidos son del 0 a 9 y de la A a la F o de la a a la f. La transferencia de un byte de datos ocurre por cada dos caracteres seriales de datos. Se ignoran los caracteres inválidos en la porción de datos de los mensajes recibidos, pero serán repetidos, si echo está habilitado. El formato de este mensaje es: <comando, dirección, datos, terminador>, con las direcciones y los datos representados por dos caracteres HEX, pero el comando y el terminador son simples caracteres ASCII. La dirección 0FFh es reservada, y no debe ser usada. Se pueden enviar valores múltiples de datos en un mensaje. El terminador es <cr> (0Dh), y no será escrito cuando se lo reciba.

0 Modo de caracteres Binario

Cada caracter representa el valor de un byte, y la transferencia de datos ocurre por cada byte de datos que se recibe. El formato de este mensaje es: <direcciones, datos, terminador>. Todos los mensajes implican mensajes de escritura, con ningún comando explícito. Las direcciones se representan con un simple byte, y la dirección 0B3h es reservada. Múltiples valores de datos se pueden enviar en un mensaje. El terminador es 0B3h, y no se escribe cuando se lo recibe. Si 0B3h es el primer valor de datos, no se lo considera como terminador, y se lo transfiere al lado paralelo. Cualquier valor 0B3h después del primer byte de datos terminará el mensaje. Este modo deberá usar caracteres seriales de 8 bits de longitud, de lo contrario, no se podrá recibir el terminador.

F Modo de Caracteres ASCII

Cada caracter representa un valor de un byte, y la transferencia de datos ocurre por cada byte de datos recibido. Si se ha escogido una longitud de datos de 8 bits, este modo podrá conducir todas las 8 líneas, como en el modo binario. De lo contrario, el octavo bit será cero cuando se lo escriba e ignorado cuando se lo lea. El formato del mensaje es: <Comando, direcciones, datos, terminador>. Los comandos son simples letras ASCII, las direcciones son representadas por dos caracteres HEX,

y los bytes de datos pueden ser cualquier valor excepto el terminador. El terminador es <cr>, 0Dh, y es escrito cuando se lo recibe, a menos que el comando especifique que no se debe escribir el terminador. La dirección 0FFh es reservada, y no debe ser usada.

Nota: La actual implementación del CY-233 permitirá direcciones 0FFh en mensajes, y chequeará las direcciones 0FFh en el modo Master, cuando se chequee todas las posibles direcciones. Sin embargo, la dirección 0FFh no debe ser válida en general, a causa de que el CY233 conduce las líneas de direcciones a este estado entre los mensajes.

- DUP (E) Pin 21: Duplex. Esta línea selecciona el modo de operación duplex, y junto con la línea NET, determina las características de operación serial del CY233. Este modo es chequeado dinámicamente, mientras el CY233 no está ocupado en alguna otra operación.

- NET (E) Pin 22: Network. Junto con DUP, esta señal determina el protocolo de operación serial del CY233. También se chequea está línea cuando el CY233 está en el modo de Background, sin transferencias pendientes, y puede ser usada para cambiar los modos en el CY233. El cambio más común es entre el Modo Master y el Modo Esclavo de una Selección particular de Echo (Repetición), con los modos asignados de tal manera que permiten un fácil cambio, ya que el Modo Esclavo tiene una línea flotante, y el modo correspondiente Master tiene una línea en cero. Sin embargo, se pueden soportar cambios dinámicos entre cualquiera de estos posibles modos.

La combinación de NET y DUP determina el modo de operación y las características de repetición del CY233. Dos modos de operación y cuatro modos de repetición son posibles, como se muestra abajo:

TIPO	NET	DUP	ECHO	MODO
A	0	0	Inválida	Master
B	0	F	Ninguna	Master
C	0	1	Totla	Master
D	F	0	Inválida	Esclavo
E	F	F	Ninguna	Esclavo
F	F	1	Total	Esclavo
G	1	0	Válida	Master
H	1	F	Válida	Esclavo
I	1	1	Válida	Esclavo

3.4.1.4.- Modos de Comunicación.

Como se mencionó en la descripción de los pines DUP y NET, el CY233 tiene dos modos básicos de comunicación y cuatro opciones de repetición.

En el Modo Esclavo, el CY233 espera por un mensaje serial de la red antes de iniciar la transferencia paralela de datos. La red y la computadora de control determinan el flujo de comunicaciones y las acciones del esclavo CY233. Sin embargo, un dispositivo paralelo puede forzar una operación de lectura usando la señal FPL/.

En el modo Master, el CY233 solicita las lecturas desde el dispositivo paralelo cuando no hay actividad en la red, para este CY233. Las lecturas se producen desde direcciones secuenciales válidas, sabiendo que el CY233 chequea la validez de la dirección antes de realizar la operación de lectura. Nótese que para una dirección válida en el modo codificado, el CY-233 chequeará 255 posibles direcciones válidas antes de llegar nuevamente a la dirección válida. Se envía un byte de datos por cada operación de lectura. Si el dispositivo paralelo conduce la línea R-W/ a un nivel lógico bajo, durante una lectura paralela en el modo Master, el CY233 generará un mensaje de escritura alrededor de los datos leídos. De lo contrario, el CY233 generará un mensaje de lectura.

Un mensaje válido es uno que ocurre en una dirección válida para el CY233 en cuestión. Esto es, el mensaje contiene una dirección para la cual el CY233 respectivo responderá. Un mensaje inválida es un mensaje formado propiamente, pero con una dirección que no es válida para ese CY233. En la mayoría de aplicaciones, a cada CY233 se le asigna una dirección única.

Echo All (Repetición TOTAL) repetirá mensajes válidos e inválidos, añadiendo caracteres de dato cuando se solicita una lectura.

Echo Valid (Repetición Válida) repetirá solamente los mensajes que son válidos para este CY233. Mensajes Inválidos no serán retransmitidos.

Echo Invalid (Repetición Inválida) repetirá únicamente los mensajes que son inválidos para este CY233. Mensajes válidos determinarán que se produzca una transferencia paralela, pero esta no será retransmitida; así, los mensajes válidos serán sacados de la red cuando ellos encuentran a su respectivo CY-233. Sin embargo, un mensaje válido de lectura, que se recibe de la red, retransmitirá las letras de

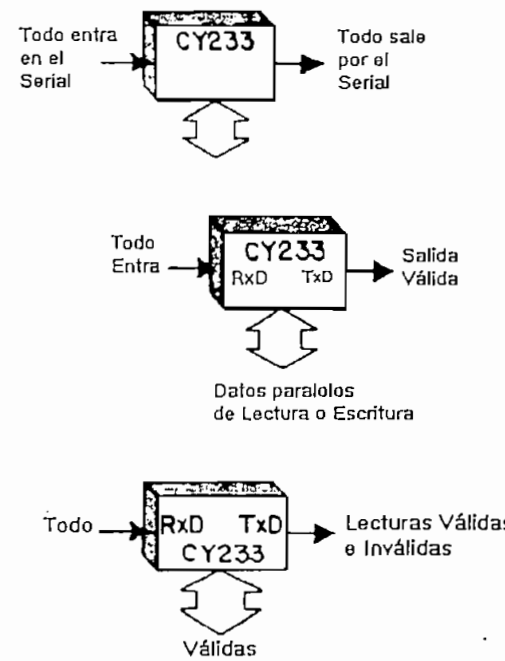


Figura 4.50

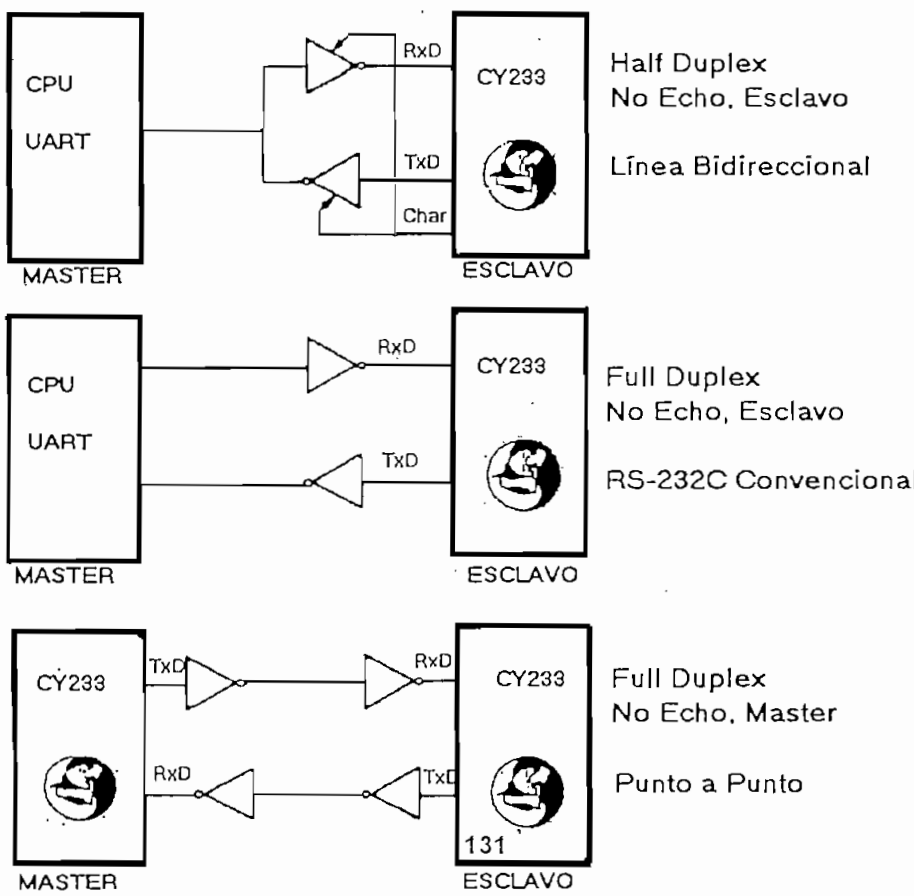
comando y la dirección, junto con los datos leídos y el terminador. Esta característica conserva el formato completo del mensaje de lectura, y permite que los mensajes de la red no se pierdan en la red de anillo.

Esta característica se implementa para todos los comandos que tengan un respuesta local, incluyendo Read, Display y Query.

Echo NONE (Ninguna Repetición) no repetirá ningún mensaje, pero los mensajes válidos causarán transferencias paralelas. Si un mensaje causa una respuesta local, como por ejemplo un mensaje de lectura, el CY233 solamente transmitirá los datos y el terminador, no la letra de comando y la dirección.

En el modo UART, en el cual no hay direcciones, todos los datos de caracteres son considerados válidos, y serán repetidos como en los casos de Echo All y Echo Valid, pero no como en los casos de ECHO Invalid y Echo None.

Nótese que el modo ASCII HEX tiene un filtro para los ca-racteres HEX. Caracteres de datos no HEX serán repetidos como se requiera, pero no se los pasará al dispositivo paralelo. A continuación se presentan algunos ejemplos de configuraciones simples de red:



3.4.1.5.- Juegos de Comandos del CY233.

Esta sección describe los comandos para implementar varias funciones del CY233. Cada comando está designado por una sola letra ASCII, y para comandos seriales, siguen la estructura general de mensajes que se ha descrito en secciones previas. Todos los comandos seriales están disponibles para los modos de caracteres ASCII, y ASCII Hex, pero no para el modo binario. El formato de los mensajes en el modo binario es diferente del modo ASCII, comenzando inmediatamente con la dirección del byte. Todo mensaje serial en el modo binario se considera que es de escritura. Para leer datos en el modo binario, debe ponerse el CY233 en el modo Master, o se debe usar la línea FPL/. Así, ninguno de los otros comandos seriales están disponibles en el modo serial. Sin embargo, los comandos especiales paralelos pueden ser usados en el modo binario.

Los comandos del CY233 usan las letras mayúsculas, y reservan estas letras para encabezonamientos de mensajes cuando se inician los mismos. La porción de datos de un mensaje no está restringida, y cualquier código puede ser usado dentro de la sección de datos, excepto el caracter terminador, el cual finalizará el mensaje.

Se pueden dividir los comandos del CY233 en tres grupos que dependen de la forma como son enviados los mensajes al CY-233. Los comandos más comunes son los comandos seriales de Network, que son enviados y recibidos como mensajes en el lado serial del CY233, funcionando como parte de una red serial.

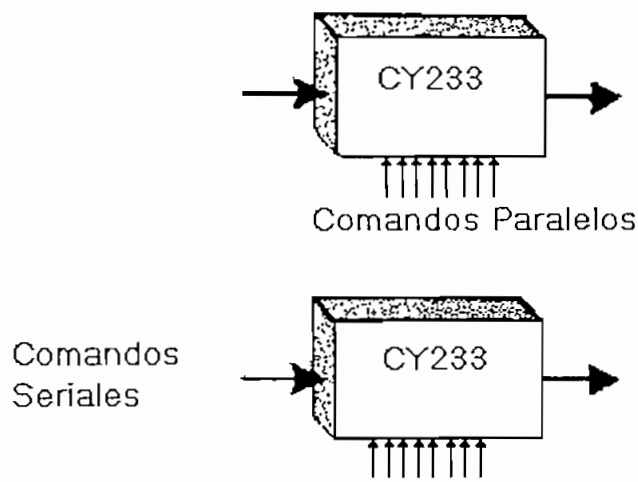


Figura 3.52

Un segundo grupo de comandos soportados por el CY233 son los comandos paralelos. Aquellos comandos son enviados al CY-233 por medio de un dispositivo local paralelo conectado al mismo. Esto

le da a un dispositivo inteligente algún control sobre la operación del CY233 local conectado a este dispositivo. El uso de estos comandos es estrictamente opcional. Estos no son requeridos en un sistema que tiene conectada lógica randómica al lado paralelo del CY233.

El CY233 tratará a los datos localmente generados como un comando paralelo, si fueren leídas por medio de la señal FPL/ con una dirección del tipo 0FFh. Cualquier otra dirección generará un mensaje serial de lectura o escritura, del tipo FPL, a la red. Así, la dirección FF es reservada para los datos generados por FPL/, y señala que un comando está siendo enviado.

La mayoría de los comandos paralelos controlan una función local del CY233, tales como el valor de tiempo fuera del handshake. Aquellos comandos tienen comandos seriales equivalentes del tipo network que ejecutan las mismas funciones. Esto permite que un dispositivo paralelo inteligente pueda configurar el CY233 de la manera más apropiada para este dispositivo, sin tener que cargar el host de la red con los detalles de los requerimientos particulares.

Algunos de los comandos paralelos ejecutan funciones que producen algunos comandos seriales especiales. Estos comandos son usados en circunstancias especiales. Algunos son implementados como soporte al modo LAN del CY233, y probablemente no se los usará en el modo standard network del CY233.

El último grupo de los comandos del CY233 es el de comandos LAN, los cuales son comandos seriales enviados al CY233 LAN desde el sistema serial local conectado al nodo LAN de una red. El CY233 LAN es uno de los dos CY233 que hacen un nodo LAN; el otro es el CY233 network (de red). EL CY233 LAN está conectado al sistema serial local que necesita comunicarse con la red. Cuando el CY233 es operado en el modo de mensaje (no-UART), los comandos seriales de LAN se usan para controlar las comunicaciones entre el sistema local y la red.

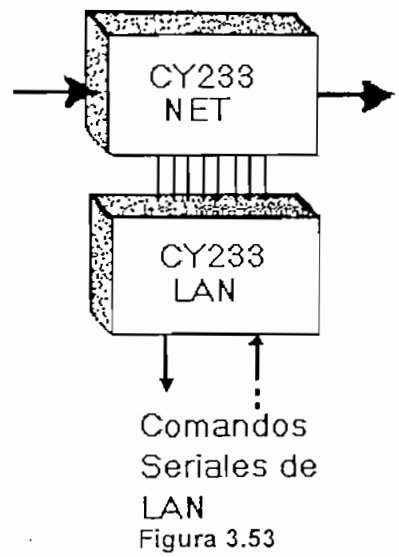


Figura 3.53

Solamente unos pocos comandos seriales de LAN difieren de los comandos seriales de red (Network).

Hay dos clases de comandos no usados en el grupo de red y de LAN. Estos son los comandos No definidos y Reservados por el Usuario. Los comandos no definidos causan un error de formato de comando inválido, si el CY233 recibe algún mensaje que inicia con una de estas letras no definidas. Estos mensajes serán retirados de la red, ya que el CY233 ignorará todos los caracteres de estos mensajes hasta que se reciba el terminador. Esto asegura que ruido o errores de comunicación no se propaguen a través de la red.

Los comandos Reservados por el Usuario son tratados de la misma forma que los comandos No definidos, hasta que el CY233 reciba un comando Echo All (Jaa<cr>). Después de esto, cualquier comando reservado por el usuario se lo trata como un mensaje serial con propósitos de comunicación. Esto permite que los dispositivos que no sean CY233 en la red tengan sus propios mensajes, independientes de aquellos que son del CY-233.

3.4.1.5.1.- Comandos Seriales de RED.

En este tipo de comandos, todos tienen el mismo encabezonamiento, el cual consiste de una letra de comando en particular (mayúscula), seguida por dos caracteres de dirección hex, indicadas por "aa". Esto es seguido por la porción de datos de cada comando, y varía con la función particular involucrada. El comando termina con el caracter terminador, un caracter de retorno en el modo ASCII.

Comando READ(R)

El comando READ designado por R, leerá y transferirá un byte de información, si el comando se lo recibe desde el lado serial, o si es iniciado desde el modo Master. Sin embargo, si se inicia una lectura local desde un dispositivo paralelo, usando la línea FPL/, bytes de múltiples datos pueden retornados en el mismo mensaje. Por lo tanto los mensajes de lectura pueden tener más de un byte. El formato general es el siguiente:

Raa<cr>	Mensaje enviado por el Host Serial, o
Raad...d<cr>	Réplica en ASCII, y
Raahh...hh<cr>	Réplica en Hex.

En el modo ASCII, "d" representa un valor simple de dato, enviado por un simple caracter. En el modo HEX, "hh" representa un valor simple de dato, enviado como dos caracteres Hex, con el más significativo enviado primero.

Mensajes iniciados localmente el modo master, o usando la línea FPL/, siempre contendrán el encabezonamiento "Raa", permitiendo al Host de la red identificar la fuente de los datos.

Comando Write (W)

El comando Write, designado por "W", transferirá cualquier número de bytes al dispositivo paralelo, y termina cuando el terminador se reciba. El formato es:

Waad...d<cr>	en ASCII,
Waahh...hh<cr>	en Hex, y
ad...d<B3>	en Binario.

Nótese que el modo binario usa bytes simples para todos los datos, incluyendo las direcciones del CY233 que inician el mensaje. Un formato idéntico es usado para enviar datos en el modo Master o por la línea FPL/ en modo binario. Si el primer byte de datos del mensaje binario tiene el valor FFh, no se lo considera como terminador, y será escrito al dispositivo paralelo.

En el modo de caracteres ASCII, el caracter de retorno de terminación se lo escribe al dispositivo paralelo, lo que no se hace en el modo binario. En el modo ASCII Hex, no se escribe el terminador, puesto que no es parte del alfabeto Hex. Para escribir el valor 0Dh al dispositivo paralelo, se debe enviar los dos caracteres ASCII "0" y "D".

Cuando el dispositivo paralelo lleva a bajo la línea R-W/ en el modo Máster o en una lectura FPL/, el mensaje resultante será un mensaje escrito en lugar del normal mensaje de lectura. Los formatos son idénticos a los recibidos desde el lado serial, y permiten que un dispositivo paralelo genere mensajes de escritura para otros dispositivos paralelos en el CY233 de red. Esta característica permite que la red no disponga de host, pero límite los tipos de mensajes que pueden ser manejados en la red. Solamente los comandos "R" y "W" estarían disponibles en este caso.

Comando (X) TRANSFER

El comando Transfer, designado por "X", es similar al comando de escritura, pero en el modo de caracteres ASCII no escribirá el caracter de retorno, a menos que sea el primer caracter de datos. Esto permite al usuario operar el CY233 en el modo ASCII, pero se escriben los mensajes al dispositivo paralelo sin escribir el caracter de retorno terminador.

El formato de este comando es el siguiente:

Xaad...d<cr>

El comando escribirá "d...", como un byte por caracter recibido, con un handshake normal realizado por cada transferencia.

Comando (M) FILL

El CY233 tiene un comando especial para la escritura de bytes individuales a direcciones secuenciales, y se lo conoce como el comando FILL, y se usa la letra "M". El formato de este mensaje incluye un parámetro de conteo después de las direcciones, para indicar cuántos bytes deben leerse. El contador es un valor HEX de dos caracteres, similar a las direcciones, y permite que se puedan leer desde 1 hasta 256 bytes de datos. El formato es el siguiente:

Naacc<cr>	si lo envía el Host, y
Naacchh...hh<cr>	que envía el CY233 como réplica.

En el modo de No Repetición, el CY233 no repetirá la parte "Naacc" de la réplica. Los valores son enviados en el formato HEX en los dos modos ASCII y HEX, y habrá cc valores de hh datos en la réplica, con cada valor viniendo de la próxima dirección secuencial, comenzando en la dirección aa. Solamente en la dirección de inicio se chequea la validez. Si el valor cc es cero, se enviarán 256 valores.

El CY233 implementa varios comandos de control de modos que no tienen porción de datos en sus mensajes, Aquellos comandos habilitan una función particular que es especificada por una letra de comando solamente. Ninguna transferencia de datos se produce, puesto que estos comandos controlan algunas funciones internas del CY233. Nótese que se debe recibir una dirección válida antes de que el CY233 actúe en el comando, permitiendo que aquellos comandos sean usados solamente por un dispositivo en particular en una red de varias partes.

Comando ALL ECHO (J)

Este comando habilita la repetición de caracteres entre los mensajes. Las repeticiones continúan hasta que se encuentra un caracter de terminación. Se repetirán también cualquier caracter de comando, código de control, letra minúscula o caracter ASCII. Sin embargo, tan pronto como se encuentra una letra de control válida, el CY233 lo tratará como un mensaje. El formato del mensaje es el siguiente:

Jaba<cr>

Una vez que se ha deshabilitado este modo, solamente se lo puede volver a activar si se reinicializa el CY233.

Comando ERROR ENABLE (K)

Otro comando de control habilita la función del status del error paralelo, y se lo designa con "K". Cuando se recibe este comando, el CY233 indicará el status de error al dispositivo paralelo (cuando ocurre un error), usando la señal WR/, mientras el CY233 está desocupado. El formato de este mensaje es el siguiente:

Kaa<cr>

Comando INITIALIZE (I)

Este comando permite a la red reinicializar el CY233 sin pulsar la línea Restart. El comando INITIALIZE, designado por "I", es equivalente a generar una reinicialización local por hardware. Se deben recibir una dirección válida y un terminador antes de que el CY233 se reinicialice a sí mismo. El formato de este mensaje es el siguiente:

Iaa<cr>

Este comando causa que el CY233 se reconfigure, lo que determinará que se adopten todas las características iniciales de encendido.

Comando QUERY (Q)

El comando QUERY, designado por "Q", enviará el status interno de información del CY233 al lado serial, y no intentará una transferencia paralela. Es utilizado como un mecanismo para reporte de errores, indicando si han ocurrido problemas de paridad, buffer overflow, mensajes mal formados, o fuera de tiempo paralelos. La banderas de status son limpiadas cuando son leídas por este comando. El byte serial del status del error reportado por este comando es separado del status del error paralelo que puede ser leído por un dispositivo local paralelo. Esto permite que tanto la red como el dispositivo paralelo soliciten al CY233 el status del error. Cuando un error ocurre, se fijan los bits de status de los dos bytes, y cuando un byte de status se lee, solamente ese byte de status se lo limpia, permitiendo al otro lado, serial o paralelo, recibir también la información de error.

El formato del comando es el siguiente:

Qaa<cr>	cuando lo envíe el host, y
Qaass<cr>	cuando es enviado por el CY233, o
Qaassss<cr>	cuando lo envía el CY233 con dos valores.

Los bits de error en el byte serial de status son los mismos que aquellos del byte de status paralelo, y son mostrados abajo. Note que el Bit 7 se lo añade como un indicador de status, y no será limpiado cuando se lo lee. Este bit se lo fija cuando el comando paralelo Sense es cargado, y habilita el paso del mensaje Token U al lado paralelo.

Bit 0	Error de paridad recibido.
Bit 1	Buffer overflow en la recepción.
Bit 2	Formato de comando inválido.
Bit 3	Fuera de tiempo en el handshake de escritura paralelo.
Bit 4	Fuera de tiempo en el handshake de lectura paralelo.
Bit 5	Fuera de tiempo en el handshake de escritura en el modo FPL
Bit 6	El caracter recibido no fue un caracter de retorno en el seteo de la velocidad de transmisión adaptiva.
Bit 7	Habilitación del mensaje de escritura TOKEN U.

Comando TIMEOUT (T)

El comando TIMEOUT, designado por "T", permitirá a la red definir el tiempo fuera para el handshake paralelo.

Si un dispositivo local no responde a la transferencia de datos del CY233 dentro de este período, se fija un error, y el Cy233 continúa. Si el CY233 está escribiendo al dispositivo paralelo, la transferencia de datos continúa como si el dispositivo paralelo hubiera generado su propio handshake.

Cuando el CY233 está tratando de leer desde el dispositivo paralelo, un tiempo fuera finalizará la lectura y terminará el comando. No se enviará ningún dato leído en este caso. Sin embargo, se enviará el mensaje terminador, indicando el fin de una transferencia.

El comando T permite a la red redefinir el período de tiempo fuera. El formato de este mensaje es:

Taatttt<cr>

La porción "tttt" representa un valor de tiempo de 16 bits, enviado como 4 dígitos hex. Cada incremento corresponde a cerca de 1,1 useg. en 11 MHz dando un rango de tiempo fuera desde cerca de 1 useg. hasta 72 mseg. para el handshake. Si el valor es cero, se deshabilita la función de tiempo fuera.

Comando PERIODIC DELAY (P)

El CY233 tiene un segundo comando relacionado con el tiempo, designado por "P", para un retraso periódico. Esta función de tiempo puede ser usada cuando el CY233 está en el modo master, y fija un valor de tiempo de retraso para el CY233 entre las lecturas de datos en el modo master.

El default para este retraso de tiempo es la deshabilitación, así el CY233 intenta lecturas en el modo Master tan rápido como sea posible.

Si se especifica el retraso, el CY233 se retrasará el tiempo especificado antes de intentar otra lectura en el modo Master. El formato de este mensaje es:

Paatttt<cr>

Este formato es similar al del comando TIMEOUT. Cada incremento representa 72mseg., dando un rango desde 72 mseg hasta 78 minutos de retraso. Debe notarse que un CY233 que está ocupado procesando otros mensajes de la red no se retrasará exactamente el tiempo especificado por este comando, puesto que otras transferencias de datos efectuarán la función interna de retraso de tiempo.

Comando NETWORK TEST (@)

Cuando el CY233 recibe un mensaje que empieza con "@", es simplemente repetido, sin acción local. Ningún test de direcciones se ejecuta, y la función de repetición ocurre en cualquier modo, incluyendo el caso de No Repetición. La repetición continuará hasta que se recibe el caracter terminador. El propósito de este mensaje es proveer un mecanismo para chequear las redes de anillo. Cuando el host que controla el anillo envía el comando de test, este será repetido de un CY233 a otro, hasta que regrese al host. Si ha llegado de regreso el mensaje, el host habrá detectado que todos los nodos del anillo están trabajando, y que todas las conecciones del anillo están operando. Si el mensaje no regresa, una conección del anillo estará rota, o uno de los nodos no está operando. Sabiendo que cualquier formato para este test se puede utilizar, se recomienda:

@FF<cr>

En un anillo operando correctamente, el mensaje regresará de la misma forma como se lo envió.

Comando ADDRESS SENSE (S)

El comando de censado realiza una función especial en la red de anillo del CY233. La acción tomada por un CY233 particular depende de si la primera dirección del mensaje censado es válida en ese

CY233 o no. Se ha provisto al CY233 de este comando como un mecanismo para que los dispositivos paralelos inteligentes puedan determinar que direcciones están en la red de anillo. Este comando también es soportado por una red LAN para determinar que otros nodos estás residentes y funcionando.

Cuando el mensaje censado es recibido por un CY233 en forma serial, se chequea la dirección, como con cualquier otro mensaje serial. Si la dirección no es válida en este CY233, se repetirá el mensaje censado hasta que se reciba el terminador de mensajes. Antes de que el CY233 repita el mensaje, añadirá la dirección local de este CY233 al final del mensaje.

El mensaje resultante tendrá entonces las direcciones de todos los CY233 del anillo. El mensaje será retransmitido en cualquier modo. El formato general del mensaje es el siguiente:

Saaxx...xx<cr>

Cada "xx" representará la dirección de uno de los CY233s. Si el CY233 recibe el mensaje conteniendo su dirección en la primera parte del mensaje, lo retirará de la red y lo cargará en el lado paralelo.

Comando TOKEN (U)

Cuando el CY233 recibe un mensaje comenzando con la letra de comando Token, "U", ejecutará una de dos funciones. Si el bit 7 del status serial de error está fijado en el CY233, el mensaje token será escrito en el dispositivo paralelo y removido de la red. Ningún chequeo de direcciones se ejecuta en este mensaje.

El comando TOKEN es implementado para permitir que las comunicaciones en la red ocurran en las bases de un token de paso.

El uso de esta función es estrictamente opcional. Las comunicaciones en una red pueden realizarse sin un esquema token total, y se espera que la mayoría de las redes no usen estos mensajes. El esquema token es muy útil en comunicaciones basadas en LAN.

Aunque el mensaje Token soporta un formato general, se recomienda el siguiente:

Uaa<cr>	para mensajes Token de red, con
Uaa<cr>	pasados al dispositivo local.

Si el bit 7 del byte serial del status de error no ha sido fijado para la recepción del mensaje token en el CY233, este no pasará el mensaje al dispositivo paralelo. En este caso, se repite el mensaje Token

al siguiente CY233. Esto permite que un anillo disponga de dispositivos paralelos inteligentes conectados, que pueden operar en una base token, junto con nodos CY233 que no están conectados a dispositivos inteligentes. Aquellos nodos con hardware normal solamente pasan el token al próximo nodo que pueda manejar el mensaje.

Comandos Reservados

En adición a los comandos ya descritos, las letras "G" y "O" están reservadas para propósitos de chequeo. Estas letras no pueden ser usadas para iniciar mensajes normales de comando.

3.4.1.5.2.- Comandos Paralelos.

En adición a los mensajes seriales enviados al CY233 en la red, este también puede aceptar comandos desde el dispositivo local conectado al lado paralelo del mismo. Todo comando paralelo se lo lee usando la línea FPL/ para cargar los bytes del comando, con el dispositivo paralelo suministrando una dirección externa de 0FFh. Cualquier otra dirección tratará los datos cargados como parte de una función de lectura standard FPL/, y generará un mensaje de escritura o lectura en la red, dependiendo del nivel de la línea R-W/.

Cuando el CY233 detecta que el dispositivo paralelo está enviando un comando, se chequea que la letra de comando sea válida, y luego se ejecuta el comando. De otra manera, el CY-233 tratará los datos como una función standard FPL/, y generará un mensaje de escritura o de lectura con dirección FF.

Todo comando paralelo involucra múltiples bytes; por lo tanto, el dispositivo paralelo debe mantener la línea FPL/ en un nivel lógico bajo hasta que el CY233 haya leído el comando entero. Una vez que el comando se ha leído, el CY233 ejecutará la acción deseada, y entonces continúa con su operación normal.

La mayoría de los comandos paralelos duplican las funciones que también están disponibles a través de los mensajes seriales. Sin embargo, algunos ejecutan funciones algo diferentes. Estas serán discutidas en esta sección.

Todos los comandos tienen un formato similar a la estructura de mensajes en el lado serial. Sin embargo, no usan el campo de direcciones. En su lugar, los comandos paralelos usan una letra de comando, un campo de datos (si es necesario), y un terminador. Todos los bytes ingresan usando los valores de los códigos ASCII para estos caracteres. Los parámetros ingresan como caracteres hex cuando sean necesarios, no como valores binarios.

Los comandos son los siguientes:

Comando INITIALIZE (I)

El comando de inicialización paralelo es equivalente a la versión serial, y permite que el dispositivo paralelo reinicialice el CY233 local. El formato del comando es:

I<cr>

Comando ALL ECHO (J)

Es equivalente al comando serial. El formato es el siguiente:

J<cr>

Comando PARALLEL ERROR ENABLE (K)

El comando de habilitación del error, que habilita al CY233 para que presente cualquier error al dispositivo paralelo, también puede ser enviado por el dispositivo paralelo. El formato del comando es el siguiente:

K<cr>

Comando HANDSHAKE TIMEOUT (T)

El dispositivo local paralelo también puede fijar el retraso de tiempo fuera usado por el CY233 en lecturas o escrituras con este dispositivo. La función de este comando es similar a la versión serial. El formato es el siguiente:

Ttttt<cr>

Comando PERIODIC TIME DELAY (P)

El dispositivo local también puede controlar el retraso periódico usado para fijar la velocidad de muestreo para el CY233 en el modo Master. Este comando es equivalente a la versión serial. El formato es:

Ptttt<cr>

Comando LOAD SECOND STATUS BYTE (L)

El dispositivo paralelo tiene la oportunidad de cargar un byte especial en el CY233, usando el comando de carga paralelo, designado por la letra "L". Este comando no tiene un equivalente serial. Es único para el lado paralelo. Si el valor cargado no es cero, el CY233 enviará este valor como parte de un comando serial de solicitud (query). No se ha asignado significado especial a este valor, y puede ser usado como se desee en la aplicación. Una posibilidad podría ser para identificar la naturaleza del dispositivo paralelo conectado al CY_233. Esto permitiría que un host serial, que está controlando la red, determine a través de un comando de solicitud que tipo de dispositivo paralelo está conectado en la red. El formato de este comando es:

Lvv<cr>

La porción vv es un valor hex de dos dígitos que es usado como el valor del segundo byte de status. Cuando el valor no es cero, el CY233 lo enviará después del byte serial del status de error. Se lo fija a cero cuando se resetea el CY233.

Comando SENSE ADDRESS (S)

El comando paralelo para sensar direcciones está relacionado con la versión serial. Sin embargo, la función exacta es diferente que la del lado serial. Cuando el dispositivo paralelo carga el comando en el CY233, se inicia la generación de un mensaje serial censado. La primera dirección del mensaje será la dirección local del CY233, así cuando el mensaje se lo envía alrededor de la red de anillo, será removido del CY233. La respuesta será escrita en el mensaje censado de regreso en el dispositivo local paralelo, con la dirección de todos los CY-233s de la red. El formato del comando es:

S<cr>	Cuando lo envía el pórtico paralelo
Saca<cr>	Cuando se inicia el mensaje serial
Saaxx...xx<cr>	Cuando lo recibe de regreso el CY233.

Debe recordarse que la dirección debe ser 0FFh para que el comando sea cargado; pero, el CY233 tratará de leer la dirección local justo después de que el comando ha sido cargado, y la línea FPL/ está en alto nuevamente. La dirección "aa" usada en este mensaje será la dirección leída por el CY233 de la línea de direcciones.

Un efecto adicional del comando paralelo para censado de direcciones es fijar el bit 7 del byte del status serial de error. Esto habilita al CY233 para que escriba cualquier mensaje token (U) al dispositivo paralelo, en lugar de repetirlo al próximo CY233 del anillo.

Comando TOKEN (U)

El comando paralelo TOKEN está relacionado con la versión serial; pero, cuando se lo carga desde el lado paralelo, su función es diferente. Cuando se recibe el mensaje Token en el lado serial, el CY233 escribe en el dispositivo paralelo, si el bit 7 del byte del status serial de error está fijado. De lo contrario, el mensaje TOKEN solamente es repetido al siguiente CY233. Cuando se carga un mensaje TOKEN desde el lado paralelo, el CY233 generará un mensaje TOKEN en la red.

Este es un mecanismo que crea un mensaje Token y lo envía al próximo CY233 del anillo. El comando del formato es el siguiente:

Uaa<cr>	cuando es cargado por el dispositivo paralelo, y
Uaa<cr>	cuando se lo envía serialmente desde el CY233.

Debe notarse que se puede usar cualquier valor para la porción "aa" del mensaje Token. El CY233 no ejecuta ningún chequeo de direcciones o no espera ningún valor en particular para este campo.

3.4.1.5.3.- Comando Seriales de LAN.

Cuando el Cy233 es operado en este modo, se lo conecta serialmente al sistema serial local, y tiene su lado paralelo conectado a otro CY233 que está operando en el modo normal de red.

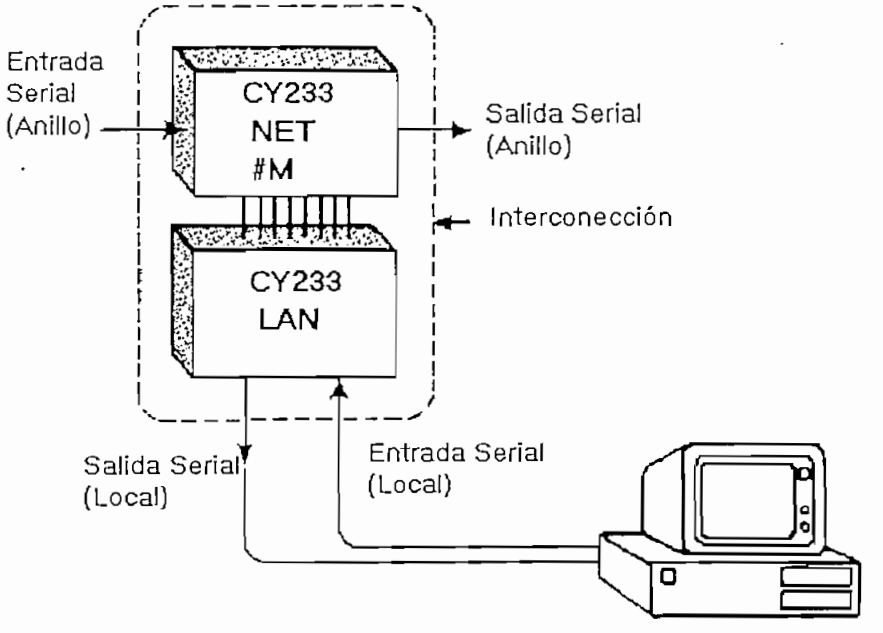


Figura 3.54

Este esquema de conección permite al sistema serial local que sea conectado a la red de otros dispositivos seriales, a través de los nodos del CY233. Una computadora standard serial puede ahora comunicarse con cualquier otro sistema también conectado a la red.

Cuando el CY233 LAN está operado en un modo de mensaje (no UART), el CY233 LAN opera con mensajes seriales similares a la operación de red. Sin embargo, algunos de los comandos de mensaje son un poco diferentes de las versiones de red standard, y otros no deberían ser usados en este esquema. En general, un CY233 tratará un mensaje de la misma manera que el CY-233 de red, a menos que el comando tenga una función especial LAN. Sin embargo, algunos de los comandos, tales como el llenado secuencial y display (M y N), transferencia (X), y la habilitación del error paralelo (K) no tienen el mismo sentido en el sistema LAN.

Los comandos utilizado en este modo son los siguientes:

Comando WRITE (W)

El mensaje de comando más usado en el sistema LAN es el mensaje de escritura. Este es el mensaje que será enviado a otros nodos de la red, implementando así las comunicaciones desde un sistema serial a otro.

Cuando un sistema local genera un mensaje de escritura, el formato es similar al mensaje de escritura normal en red, incluyendo la letra de comando, la dirección del nodo de envío, y el terminador. Debe notarse que la porción de la dirección del mensaje contiene la dirección del nodo de la red a donde el mensaje está siendo enviado, la dirección de envío. Como una función especial en el modo LAN, el CY233 automáticamente añade la dirección fuente del mensaje en la estructura del mensaje. El formato general es:

Waad...d<cr>	Como se lo envía al sistema serial local
Waassd...d<cr>	Como se lo envía a la red, y
Waassd...d<cr>	Como se lo recibe en el sistema de envío.

La porción "aa" del mensaje representa la dirección de envío, mientras que la porción "ss" es la dirección desde donde se envía, añadida por el CY233 LAN. Esto permite al receptor conocer quien envía el mensaje, sin que esa información sea parte del mensaje.

El CY233 también puede ser operado en modo binario o de caracteres HEX, así:

Waahh...hh<cr>	como se lo envía por el sistema local en Hex.
Waasshh...hh<cr>	como se envía al otro usuario.
ad...d<0B3h>	como lo envía el sistema local en binario.
asd...d<0B3h>	como se lo envía al otro usuario en binario.

Comando READ(R)

El comando de lectura y su estructura de mensaje son similares a los usados para la escritura de mensajes en un sistema LAN. Sin embargo, el comando de lectura debe usarse con mucho cuidado. Cuando el comando de lectura llega al nodo de envío, el CY233 de red de ese nodo intentará ejecutar una operación de lectura desde el CY233 LAN. Si el CY233 LAN no tiene algo que leer, la operación fallará. En general, los mensajes de lectura deberían ser usados en una LAN si hay un host conectado a nivel de la red. Estos mensajes nunca deberán ser directos a otro nodo de la red. El formato del comando es el siguiente:

Raad...d<cr>	como se lo envía al sistema serial local
Raassd...d<cr>	como se lo envía a la red.

Comando INITIALIZE (I)

El comando de inicialización LAN es similar al comando de inicialización de RED standard, y tiene el mismo formato:

Iaa<cr>	como lo envía el sistema serial local.

La porción "aa" del mensaje representa la dirección del nodo LAN local. Si se alcanza la dirección, el CY233 LAN del nodo ejecutará la operación de reinicialización, y redeterminará todas las características de encendido. Debe notarse que este comando solo resetea el CY233 LAN del nodo LAN. El CY233 de RED no es afectado. Un comando especial de paso, descrito posteriormente, puede ser usado para reinicializar el CY233 de red.

Comando QUERY (Q)

El CY233 LAN también soporta el comando de solicitud. Este comando enviará el status de error al sistema serial local, puesto que este está conectado al lado serial del CY233 LAN. Los formatos y las funciones son equivalentes a la versión de red standard:

Qaa<cr>	como lo envía el sistema serial local.
Qaass<cr>	como lo envía el CY233 al sistema local.

La porción "Qaa" del CY233 no se enviará si se ha escogido el modo de no repetición para el CY233 LAN. El significado del byte del status del error es el mismo para el CY233 de RED, excepto que el bit 7 del status serial de error LAN indica que la LAN ha detectado un tiempo fuera para la respuesta TOKEN.

Comando HANDSHAKE TIMEOUT (T)

El comando de tiempo fuera de handshake LAN es usado para fijar el valor del tiempo fuera para el handshake entre el CY233 LAN y el CY233 de RED de un nodo LAN. El sistema local es conectado al CY233 LAN a través de un pórtico serial, así que el comando no afectará las comunicaciones entre el LAN CY233 y el sistema local. Este comando puede ser útil si el CY233 de RED y el CY233 LAN no están operando a la misma velocidad de transmisión. El formato es el siguiente:

Taatttt<cr>

Comando LOCAL ADDRESS QUERY (@)

Cuando un LAN CY233 recibe un mensaje de solicitud de dirección, designado por el caracter "@", ejecuta un función diferente a la que ejecuta el CY233 de RED. Debe recordarse que el CY233 de RED solamente repetiría el mensaje en el anillo. El CY233 LAN leerá la dirección que ha sido asignada al nodo local LAN, y retornará el valor de dirección al sistema local serial. Esto permite al sistema local determinar la dirección del nodo al cual está conectado. El formato del mensaje es:

@FF<cr> como lo envía el sistema local serial.
@aa<cr> como lo retorna el CY233 LAN.

Comando PARALLEL PASS (P)

El CY233 LAN soporta un comando especial adicional, el comando de paso paralelo, designado por la letra "P". Este comando no ejecutará ninguna función en el CY233 LAN, pero, pasará el comando en paralelo al CY233 de RED. Esto permite al sistema serial local controlar algunas de las funciones de red del CY233 de RED, aún cuando el sistema local no esté conectado de alguna manera al CY233 de RED de un nodo LAN. La letra de comando actúa como un prefijo para el LAN CY233, y este CY233 pasará cualquier mensaje que siga al CY233 de RED. Los comandos paralelos soportados por el CY233 de RED, como se han descrito en las secciones anteriores, pueden ser enviados usando este comando. El formato general de este comando es el siguiente:

Pcomando<cr>

Donde la letra "P" es un prefijo para el comando paralelo normal soportado por el CY233 de RED. No todos los comandos paralelos son apropiados en un sistema LAN. La mayoría de los comandos útiles se describirán más tarde.

Si el sistema serial local necesita conocer que otros nodos están conectados en la red, puede enviar el comando SENSE ADDRESS (S) al CY233 de RED. Así:

PS<cr>

Una vez que se ha enviado el comando para sensar las direcciones de la red, el CY233 de RED está habilitado para pasar el mensaje TOKEN al CY233 de LAN, si este mensaje llega al nodo. Cuando el CY233 recibe el mensaje TOKEN, también pasará este mensaje al sistema local. Así:

Uaa<cr>

Donde la porción "aa" es el campo de la dirección del mensaje token, no la dirección del nodo local.

En una red basada en token, se le ha permitido al sistema local que recibe el token hacer transmisiones a otros nodos de la red, normalmente usando el mensaje WRITE descrito anteriormente. Una vez que el sistema local ha terminado, libera el token, así el próximo nodo tiene la posibilidad de trasmitirlo. Para liberar el token, debe ser pasado al CY233 de RED como:

Puaa<cr>

el cual causará que el CY233 de RED genere un mensaje token al próximo nodo de la red.

3.4.1.6.- Especificaciones Eléctricas.

Las principales características de operación de este circuito integrado son las siguientes:

3.4.1.6.1.- Rangos máximos absolutos.

Temperatura ambiente	0°C a 70°C
Temperatura de almacenamiento	-65°C a +150°C
Voltaje en cualquier pin con respecto a GND...	-0,5V a Vcc+0,5V
Disipación de Energía	0,2 watts

3.4.1.6.2.- Características de operación y DC.

Temperatura: 0 C a 70 C, Vcc = +5V +/-10%					
Símbolo	PARAMETRO	MIN	MAX	UNI	OBSERVACIONES
Icc	Corriente de Fuente		18	mA	
Vih	Nivel de entrada alto	1.9	Vcc	V	(3.5v para XTAL, RESTART)
Vil	Nivel de entrada alto	-0.5	0.9	V	
Ilo	Leakage bus de datos		10	µA	Estado de alta impedancia
Voh	Nivel de salida alto	2.4		V	Ioh = -60 µA
Vol	Nivel de salida bajo		0.45	V	Iol = 1.6 mA
Fcy	Frecuencia del Cristal	3.5	12	Mhz	Ver circuito de Reloj

3.4.2.- Red de Anillo.

El tipo de red que se utilizó en este trabajo, por sus características, es la red de anillo. El anillo es la red más compleja e interesante. Aún aunque la topología es simple, sale de uno y entra a otro, la variedad de protocolos que se puede escoger es grande. En la red de anillo se pueden realizar la mayoría de las operaciones generales del CY233, y es el único tipo que no requiere de drivers de tres estados.

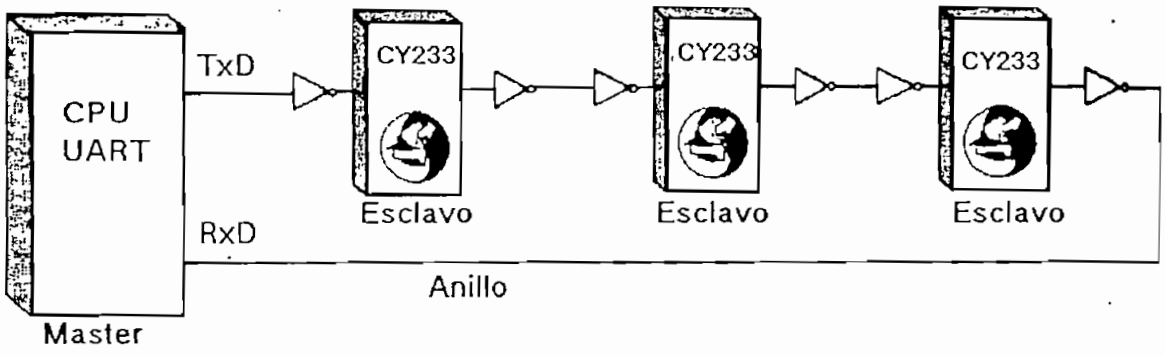


Figura 3.55

La gran ventaja del anillo es que los mensajes pueden entrar y salir del bus cuando se los requiera, y que, muchas cosas pueden estar sucediendo al mismo tiempo. Muchos dispositivos pueden estar transmitiendo y recibiendo al mismo tiempo. Naturalmente, la única desventaja es que el mensaje toma más tiempo en ir alrededor de todo el anillo, lo que significa que los mensajes no llegan a su destino inmediatamente.

Lo más importante de este diseño en anillo es que es un método para remover todos los mensajes después de que ellos han alcanzado la parte necesaria. En otras palabras, no se desea que un mensaje circule indefinidamente.

Una manera de hacer esto es tener un CPU como Master, y tener todos los CY233s como esclavos. El CY233 tiene dos protocolos de repetición, repetición de todo, y no repetición (ninguna repetición), los cuales son particularmente útiles en redes de anillo. Con repetición total, cualquier mensaje enviado por el CPU regresará a él, y el CPU conocerá que fue recibido. Esta es una muy buena característica de verificación.

Es necesario indicar que en el mismo sentido en el que los lados seriales y paralelos son independientes, la entrada serial In, RxD, y la salida serial, TxD son también independientes.

En la red de anillo, cada nodo está conectado en una cadena, con el transmisor del primer nodo conectado al receptor del segundo nodo.

La red de anillo está formada por una cadena de nodos, con los mensaje moviéndose en una sola dirección a través del anillo. En este caso, el Host genera los mensajes, y ellos fluyen al nodo respectivo. Cuando un nodo genera una respuesta para el Host, esta respuesta fluye a través del nodo subsecuente hasta alcanzar el host.

Como se han usado los conectores tipo DB25, un simple esquema de alambrado permite que todos los nodos sean conectados. A continuación veremos un esquema de la conección que se ha realizado:

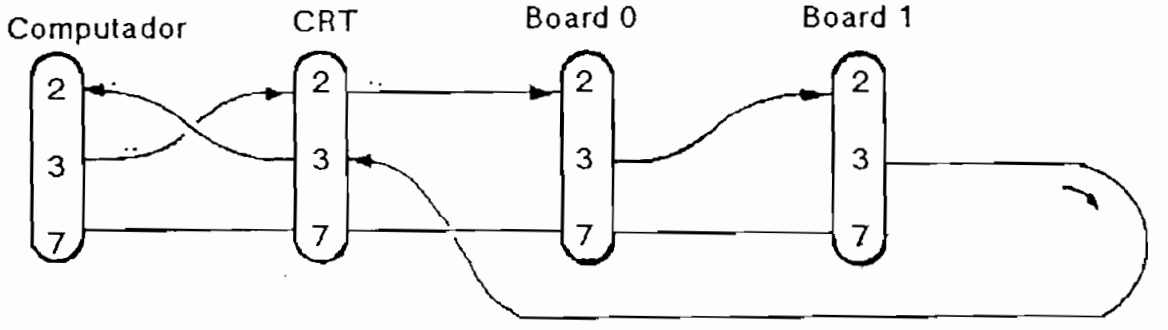


Figura 3.56

El CY233 solamente requiere tres conecciones de un nodo a otro: la señal de tierra, datos recibidos y datos transmitidos. La señal de status de modem, usadas por el protocolo RS-232, no son utilizadas por el CY233.

3.4.3.- Circuito de Reset

El Pin #9 RESTART, debe mantenerse en alto en el encendido, para inicializar propiamente el CY233. Este circuito dispone de un condensador de 4.7 uF, como se lo muestra en la figura 3.57. La línea RESTART debe mantenerse en alto por lo menos por 10 mseg después de que la energía se estabilice en el encendido. Una vez que el CY233 esté corriendo, RESTART debe estar en alto solamente 10useg (con un reloj de 11 MHZ).

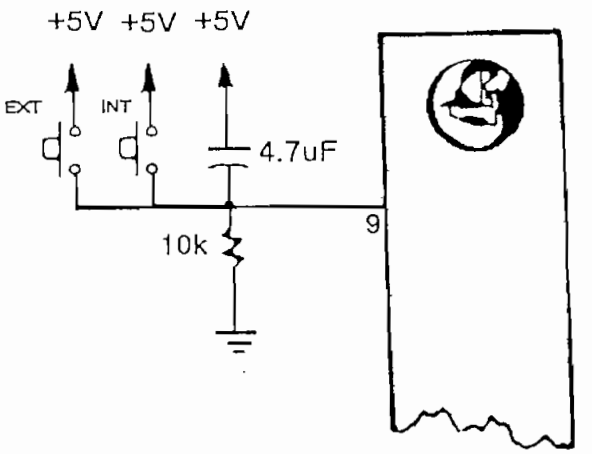


Figura 3.57

Como en el caso del CY545, también se dispone de un switch externo para resetear al CY233. Este, es el mismo switch que resetea al microcontrolador CY545.

3.4.4.- Circuito de Reloj.

El CY233 puede ser operado con un circuito de cristal o con un circuito externo. Estas dos opciones se muestran en la figura 3.58, y en el caso de este trabajo se ha utilizado un circuito de cristal. Como el CY233 requiere de un reloj de 11 MHz para trabajar con las velocidades standard de transmisión, se ha incorporado un cristal que trabaje a esta frecuencia.

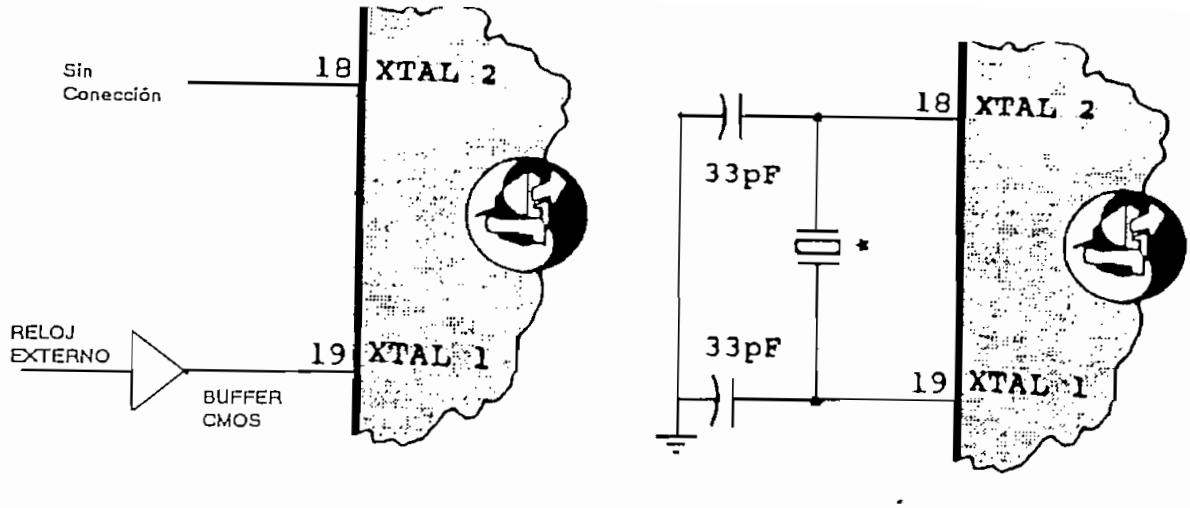


Figura 3.58

3.4.5.- Configuración del CY233.

Como se ha dicho en las secciones anteriores, el CY233, en este trabajo, se lo ha utilizado en una configuración de anillo. Se han utilizado dos controladores seriales CY233 en este trabajo, formado una cadena que incluye al computador central. Los dos controladores seriales CY233 permiten que el Host se comunique con los dos CY545 conectados correspondientemente a cada uno de ellos.

Puesto que solo se han utilizado dos CY233, se ha adoptado el sistema de direcciones codificado 1-8; para ello se ha llevado la señal U-E-D (Pin 23) a Cero lógico. El primer CY-233, que forma parte del manejo del eje X, tiene una dirección hexadecimal 01H, mientras que el segundo, que forma parte del sistema del eje Y, tiene una dirección hexadecimal 02h. Como se puede observar en los circuitos, en los dip-switches que determinan las direcciones de cada uno de los CY233, uno de ellos tiene como dirección 01 y el otro 02. por lo tanto, cualquier mensaje que sea enviado, llevará una de las dos direcciones, dependiendo de que CY545 se desea utilizar.

En cuanto al circuito de reloj, se han utilizado cristales de 11 MHz, para poder disponer de velocidades de transmisión standard. Como no se requiere de excesivas velocidades de transmisión entre el host y el taladro, se ha utilizado una velocidad de 2400 baudios. Para lograrlo, se ha enviado la señal BDR1(Pin 28) a un nivel lógico alto, y la señal BDR0(Pin 27) a un nivel lógico bajo. Esto determina que el CY233 trabaje a una velocidad de transmisión de 2400 baudios. Adicionalmente, se ha escogido NO PARIDAD y 8 bits de DATOS para la transmisión, llevando las líneas PAR0(Pin 25) y PAR1(Pin 26) a un nivel lógico alto.

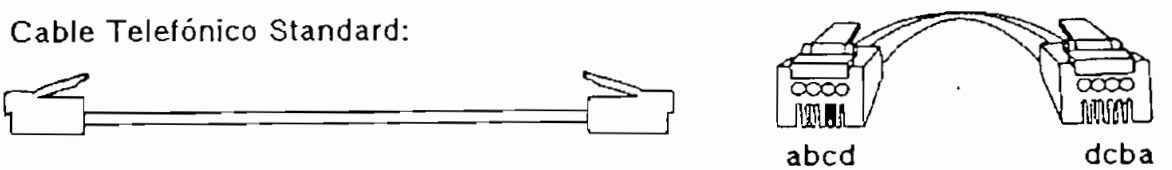
La señales de handshake para la transferencia de datos están conectadas a las señales apropiadas del CY545, implementando el adecuado interfase paralelo entre el CY233 y el CY-545. Las

señales del bus de datos están también conectadas directamente entre el CY233 y el CY545.

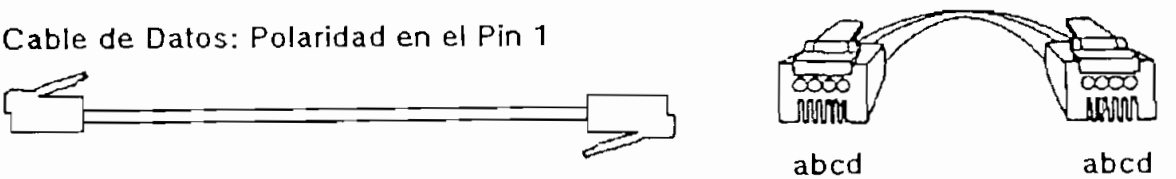
Las líneas para transmisión serial están conectadas al MAX233, y a través de él, a los conectores RJ11 seriales. Por conveniencia, para realizar daisy chain independientemente en el un circuito o en el otro, se ha colocado conectores RJ11 dobles, con un lado de entrada y un lado de salida. La señal de datos recibidos del CY233 está conectada desde el lado de entrada, mientras que la señal de datos transmitidos puede ser conectada al jack de entrada o al jack de salida. En este caso, hemos conectado la señal de datos enviados que viene del computador central a la señal de datos recibidos del CY233 que maneja el eje X, y la señal de datos enviados de este CY233 se conecta a la señal de datos recibidos del CY233 que maneja el eje Y. La señal de datos enviados de este CY233 se ha conectado, a través del jack del otro circuito, a la señal de datos recibidos del computador central, estableciéndose así el anillo.

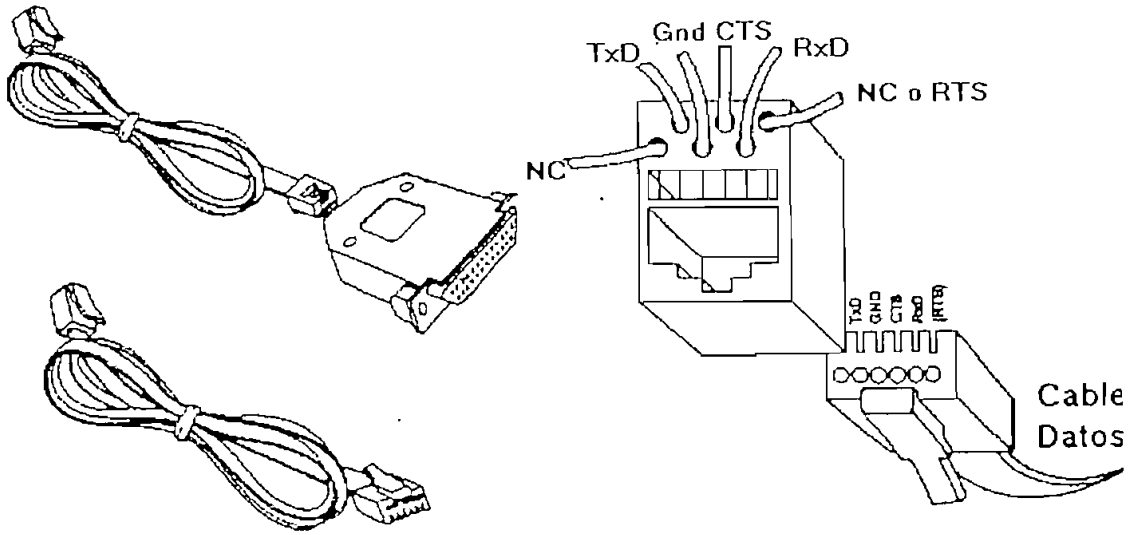
Se ha utilizado para la conección entre el computador y el primer circuito un cable tipo telefónico de cuatro hilos, con un conector DB25 en uno de sus extremos, y un conector RJ-11 en el otro. Este cable tiene la característica de mantener la polaridad en el pin 1, a diferencia de los cables normales.
En la siguiente figura se pueden observar el cable construído y las diferencias con el cable standard:

Cable Telefónico Standard:



Cable de Datos: Polaridad en el Pin 1





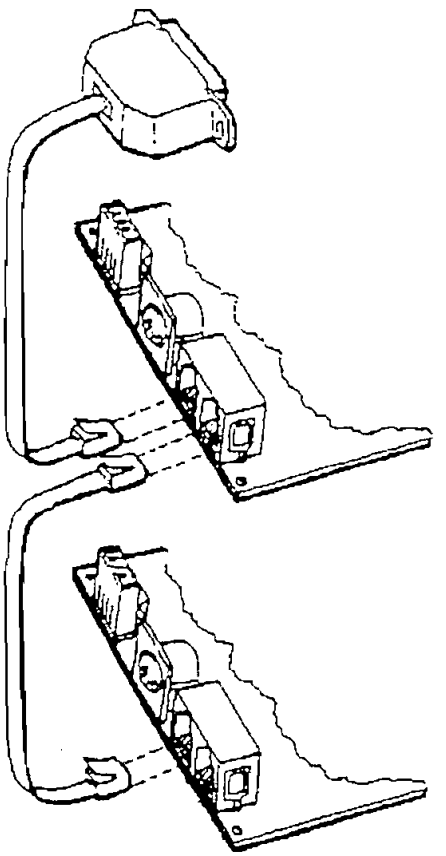


Figura 3.59

Por otra parte, se ha seleccionado el modo de caracteres ASCII, mediante el uso de la señal CHAR (Pin 24). Se ha dejado flotante esta señal.

Como es característica del anillo, se ha determinado que los CY233 funcionen en el modo esclavo SLAVE, con repetición Inválida (Las señales correspondientes a un CY233 no serán retransmitidas). Por lo tanto, se han utilizado las señales NET (Pin 22), dejada flotante, y DUP (Pin 21) colocada a 0 lógico.

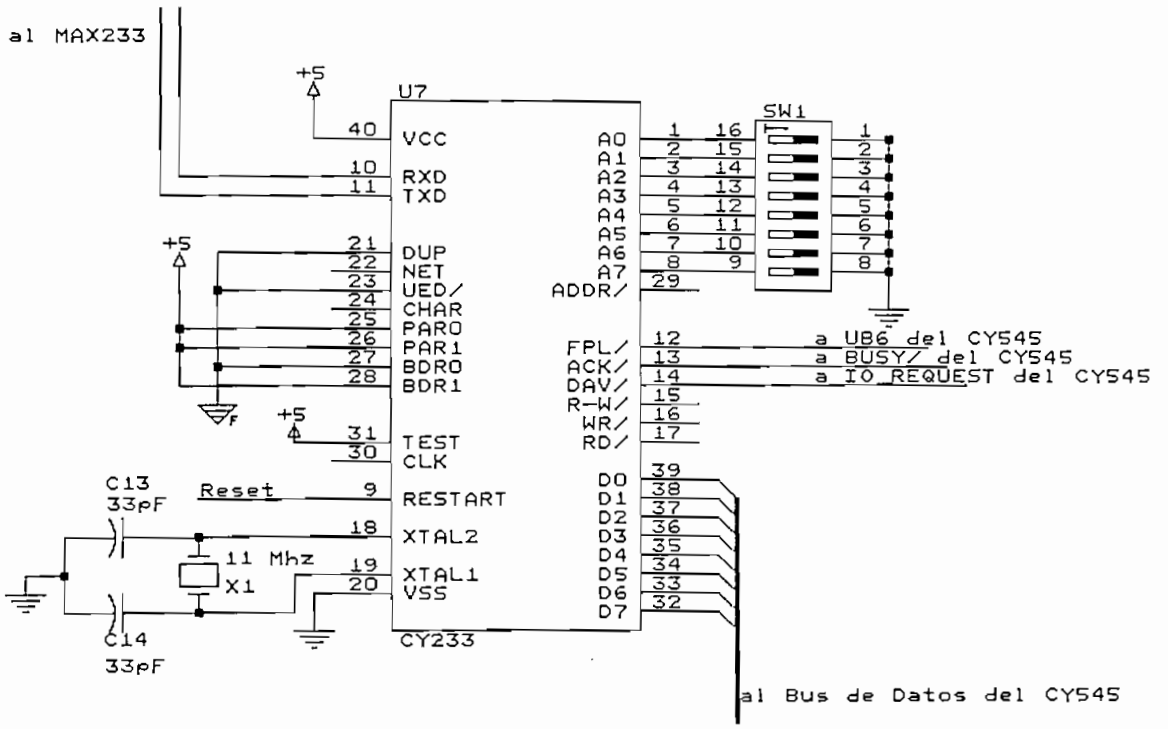


Figura 3.60

3.4.6.- Conección al computador.

Puesto que los niveles de voltaje del CY233 son TTL, es necesario adaptarlos a niveles RS-232. Para ellos se ha utilizado el circuito integrado MAX-233, el cual tiene un sistema interno de generación de voltaje y dos juegos de receivers y drivers. Con el generador interno de voltaje, el MAX 233 solamente requiere de una fuente de +5 voltios, permitiendo de esa manera que se utilice la misma fuente de 5 V. en todo el sistema.

Las señales TxD y RxD del CY233 están conectadas a un driver y un receiver del MAX233, permitiendo de esta manera la comunicación con el HOST o el otro CY233 del sistema.

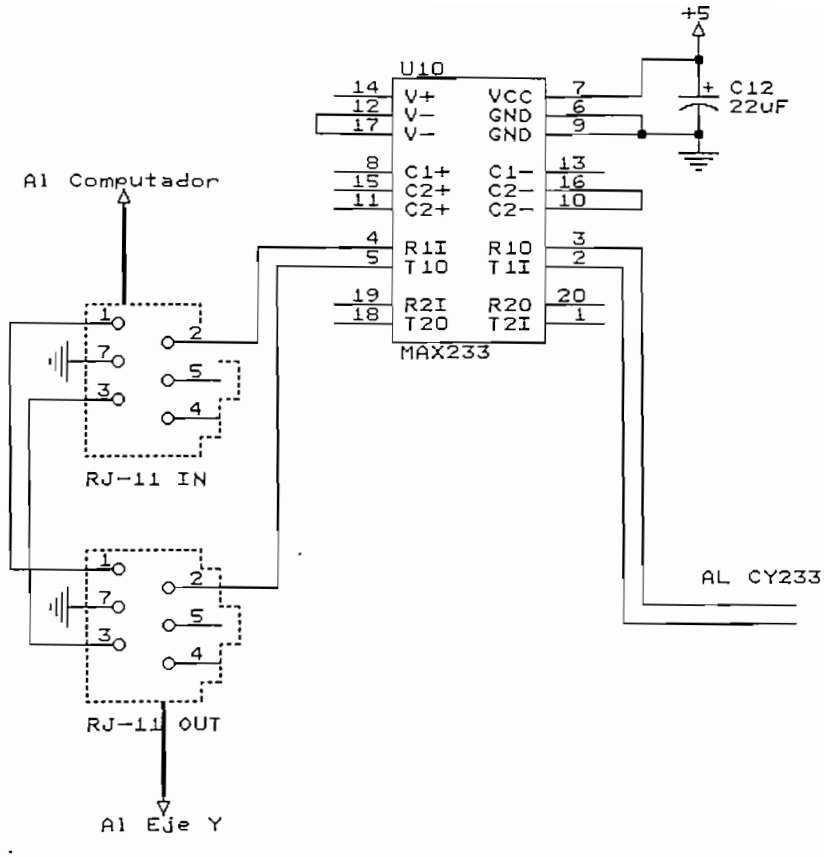


Figura 3.61

3.4.7.- Circuitos completos.

A continuación se muestran los circuitos completos diseñados, en el eje X y en el eje Y.

Obsérvese que prácticamente los circuitos X y Y son parecidos y están unidos a través de un cable plano de 10 hilos. Además, el circuito que controla el eje X y Z está también unido al circuito de potencia de los motores de pasos a través de un cable plano de 36 hilos. Todos estos cables se pueden desprender del equipo, permitiendo el desensamblaje para mantenimiento.

3.5.- Fuentes de Poder.

Para el diseño de las fuentes de poder se ha hecho la siguiente consideración: se necesitan dos voltajes +5 Voltios y +12 Voltios; la fuente de +5 Voltios para todos los circuitos integrados y el Display, y la fuente de +12 voltios para los motores de paso.

3.5.1.- Fuente de Poder de 12 Voltios.

Esta fuente se la ha empleado para suministrar energía a los motores de paso USCYBERLAB. Cada uno de los motores necesita una corriente de 1 Amperio con +12 Voltios por fase. Como son cuatro motores, se necesitarán 4 Amperios a +12 Voltios.

3.5.2.- Fuente de Poder de +5 Voltios.

Esta fuente se la ha empleado para suministrar energía a todos los circuitos integrados, incluyendo dos CY545, dos CY233, dos Display HP, dos memorias XIRCOM X2864AP, dos MAXIM 233, cuatro drivers 74244 y varios LED's entre otros. Cada sistema de control consume aproximadamente 0.950 amperios; como se utilizan dos, se necesitarán aproximadamente dos amperios.

De acuerdo a los análisis previos, se adquirió en la compañía norteamericana JAMECO un fuente de poder Switching de 73 Watts marca Power General. Esta fuente de poder presenta las siguientes características:

Entrada:	115/230 VAC @ 47-450 Hz.
Salida:	+ 5 VDC @ 5A
	+ 12 VCD @ 4A
Tamaño:	6.5" Largo x 4" Ancho x 1.75" Altura
Peso:	1.5 lbs.

Estas características cumplen con los requisitos de corriente contínua necesarios para el funcionamiento adecuado de los circuitos diseñados. A continuación se presenta un diagrama de colecciones de esta fuente de poder:

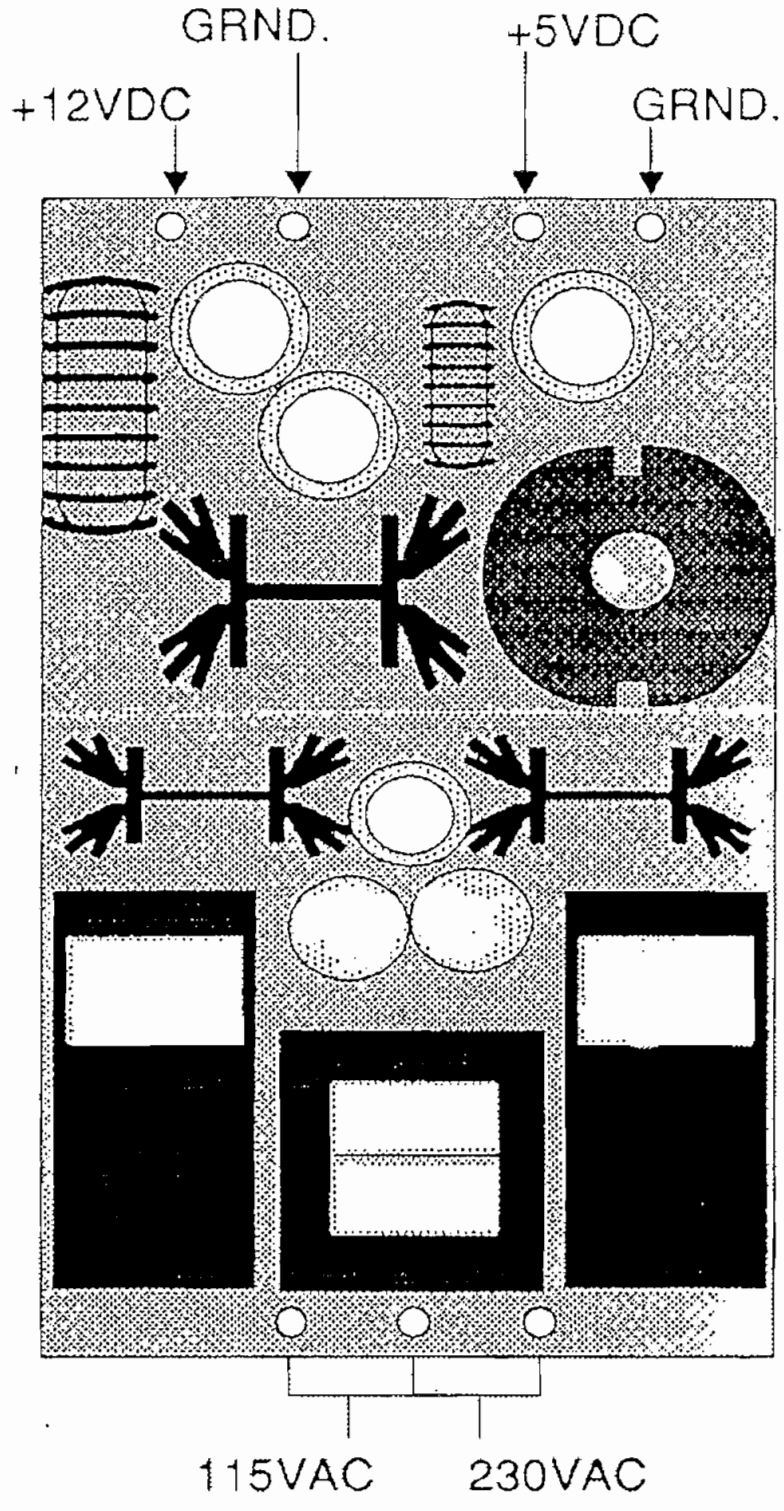


Figura 3.66

Capítulo Cuarto

Desarrollo del Software

CAPITULO CUATRO
DESARROLLO DEL SOFTWARE

Para desarrollar el programa para establecer la comunicación entre el computador y los microcontroladores CY233, y de esta manera manejar los motores de paso, se ha utilizado el lenguaje de alto nivel VISUAL BASIC FOR WINDOWS, de Microsoft Corporation. Puesto que Visual Basic presenta deficiencias en el manejo de comunicaciones seriales, y siendo éstas esenciales para este trabajo, se ha utilizado el programa PDQCOMM for WINDOWS, desarrollado por la empresa CRESCENT SOFTWARE, para dar soporte a Visual Basic en comunicaciones.

El programa diseñado está basado principalmente en dos posibilidades de trabajo: manual y automática. En la forma manual, el usuario podrá manejar cada uno de los motores de pasos en forma independiente, y podrá visualizar los parámetros de trabajo de los microcontroladores CY545. En forma automática, se podrá ejecutar el proceso de perforado de una placa en forma continua, con la posibilidad de calibrar las distancias relativas al movimiento del taladro.

Por otra parte, si se ha interrumpido la perforación en la parte automática, el programa permite reiniciar el proceso, con la posibilidad de poder variar el punto de inicio del trabajo.

Además, el programa tiene la posibilidad de transformar archivos DRILL producidos por los programas ORCAD y TANGO, a archivos que podrán ser leídos directamente por este programa. El programa de conversión permitirá realizar ordenamientos de diferentes tipos, dependiendo del análisis que realice el usuario sobre la optimización del trabajo del equipo.

4.1.- Diagrama de Flujo General.

El diseño del programa para el control del equipo se basó en la flexibilidad que debía tener para permitir ejecutar diferentes funciones, y a la vez permitir controlar los parámetros de trabajo. Una vez analizado el trabajo que debía realizarse, se dividió el programa principal en cinco partes:

- Menú inicial con información general.
- Modo Manual.
- Modo Automático.
- Manejo y Conversión de archivos.
- Reinicio de Perforación.

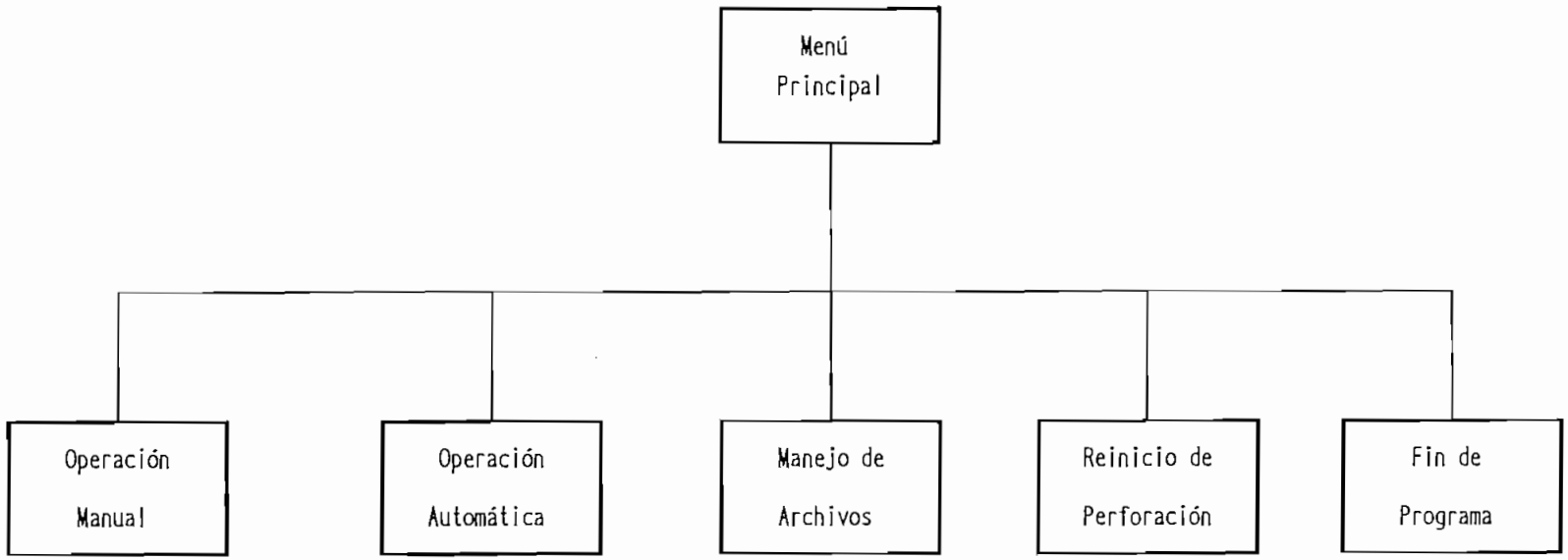


Figura 4.1

Diagrama de Opciones

Programa de Manejo del Equipo

El menú Inicial permite escoger entre las diferentes opciones que tiene el programa, y además presenta información sobre la configuración del sistema.

El modo automático permite ejecutar un proceso continuo de perforación en una plaqueta. Presenta información sobre las posiciones que se están perforando y sobre los parámetros bajo los cuales están trabajando los microcontroladores CY-545 y CY-233.

El modo manual permite realizar movimientos independientes en cada uno de los ejes, así como fijar características de trabajo y parámetros de funcionamiento en todos los microcontroladores. Además, permite calibrar el taladro para el trabajo en el modo automático.

El manejo y conversión de archivos permite convertir y observar archivos provenientes de los programas TANGO PCB y ORCAD PCB. Estos archivos serán convertidos a un formato propio, también ASCII. Durante el proceso de conversión, se puede realizar un reordenamiento de los puntos de perforación en diferentes direcciones, ascendente, descendente o intercalado. Además, es posible realizar un intercambio de ejes.

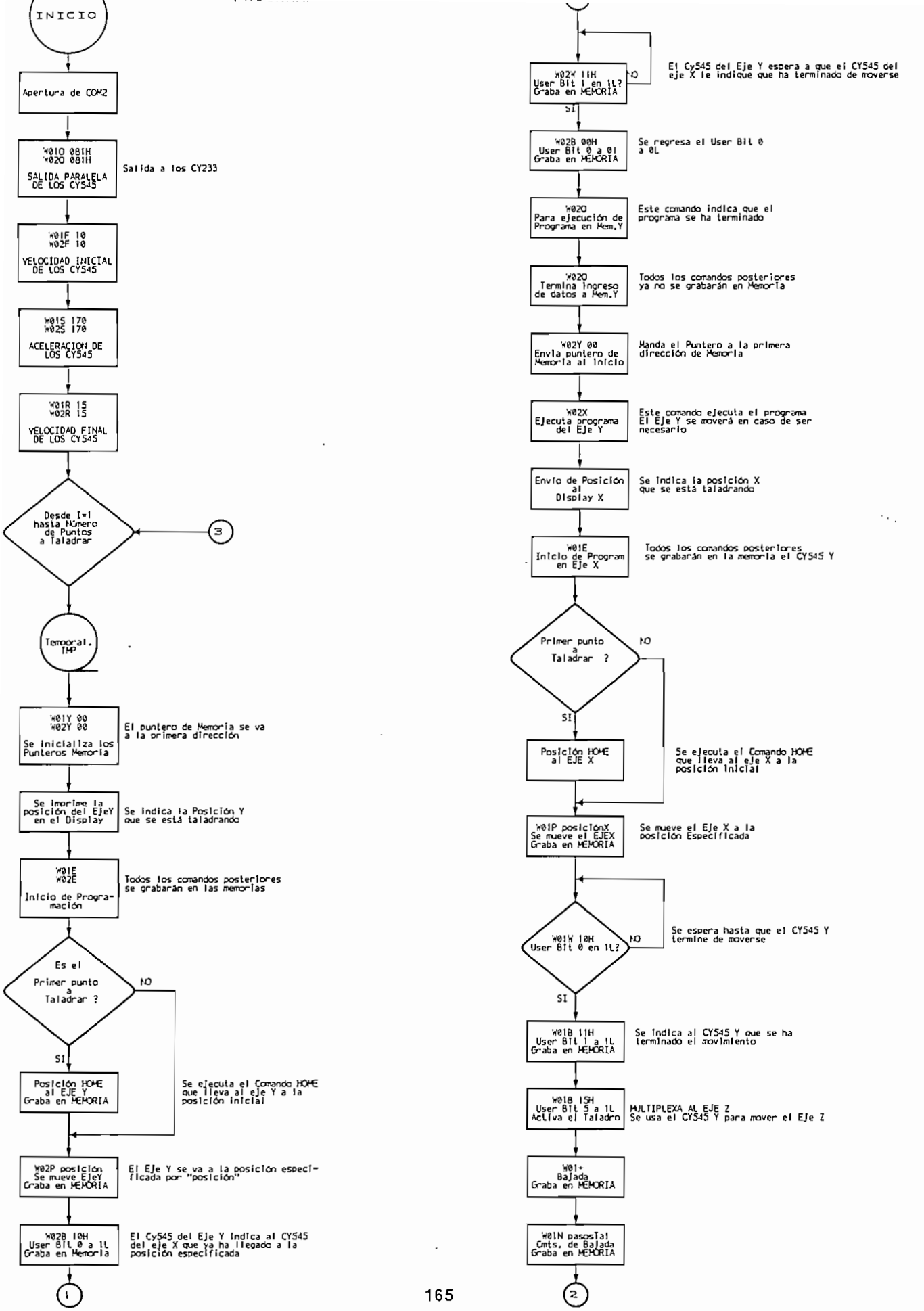
El diseño de cada una de las partes de este programa se explicará más adelante. La parte principal de este trabajo es la que tiene que ver con el proceso de perforación. Como se observa en la figura 4.2, se ha detallado el diagrama de flujo principal de este proceso. Este proceso se basa en un archivo que debe haber sido convertido previamente a un formato reconocido por este programa.

4.1.1.- Diagrama de Flujo del Proceso de Perforación Automático.

Este proceso se ejecuta una vez escogida la opción de modo automático en el menú inicial. Constituye la parte más importante de este programa. El diseño se basó en los comandos del Microcontrolador CY545, que ya se han tratado con anterioridad, y que se revisarán en el siguiente punto.

El objetivo de esta subrutina fue permitir que el equipo perfore automáticamente una serie de orificios en una tarjeta, basado en un archivo que debió haber sido convertido, previamente, al formato propio del programa. Este archivo contiene la información de la posición en el sistema de coordenadas X y Y de cada uno de los puntos, así como el diámetro del agujero.

El diagrama de flujo de este proceso se lo puede observar en la figura 4.2.


INICIO
Apertura de COM2
W01O 081H
W02O 081H
SALIDA PARALELA DE LOS CY545
Salida a los CY233
W01F 10
W02F 10
VELOCIDAD INICIAL DE LOS CY545
W01S 170
W02S 170
ACELERACION DE LOS CY545
W01R 15
W02R 15
VELOCIDAD FINAL DE LOS CY545
Desde I=1 hasta Número de Puntos a Taladrar
3
Temporal. TMP
W01Y 00
W02Y 00
Se inicializa los Punteros Memoria
El puntero de Memoria se va a la primera dirección
Se imprime la posición del EjeY en el Display
Se indica la Posición Y que se está taladrando
W01E
W02E
Inicio de Programación
Todos los comandos posteriores se grabarán en las memorias
Es el Primer punto a Taladrar ?
NO
SI
Posición HOME al EJE Y Graba en MEMORIA
Se ejecuta el Comando HOME que lleva al eje Y a la posición inicial
W02P posición
Se mueve EjeY
Graba en MEMORIA
El EJe Y se va a la posición especificada por "posición"
W02B 10H
User Bit 0 a 1L
Graba en Memoria
El Cy545 del Eje Y indica al CY545 del eje X que ya ha llegado a la posición especificada
1
W02W 11H
User Bit 1 en 1L?
Graba en MEMORIA
NO
SI
El Cy545 del Eje Y espera a que el CY545 del eje X le indique que ha terminado de moverse
W02B 00H
User Bit 0 a 0l
Graba en MEMORIA
Se regresa el User Bit 0 a 0L
W02O
Para ejecución de Programa en Mem.Y
Este comando indica que el programa se ha terminado
W02O
Termina Ingreso de datos a Mem.Y
Todos los comandos posteriores ya no se grabarán en Memoria
W02Y 00
Envia puntero de Memoria al inicio
Manda el Puntero a la primera dirección de Memoria
W02X
EJecuta programa del EJe Y
Este comando ejecuta el programa El EJe Y se moverá en caso de ser necesario
Envio de Posición al Display X
Se indica la posición X que se está taladrando
W01E
Inicio de Program en EJe X
Todos los comandos posteriores se grabarán en la memoria el CY545 Y
Primer punto a Taladrar ?
NO
SI
Posición HOME al EJE X
Se ejecuta el Comando HOME que lleva al eje X a la posición inicial
W01P posiciónX
Se mueve el EJEX
Graba en MEMORIA
Se mueve el Eje X a la posición Especificada
W01W 10H
User Bit 0 en 1L?
NO
SI
Se espera hasta que el CY545 Y termine de moverse
W01B 11H
User Bit 1 a 1L
Graba en MEMORIA
Se indica al CY545 Y que se ha terminado el movimiento
W01B 15H
User Bit 5 a 1L
Activa el Taladro
MULTIPLEXA AL EJE Z
Se usa el CY545 Y para mover el Eje Z
W01+
Bajada
Graba en MEMORIA
W01N pasosTal
Cmts. de Bajada
Graba en MEMORIA
2

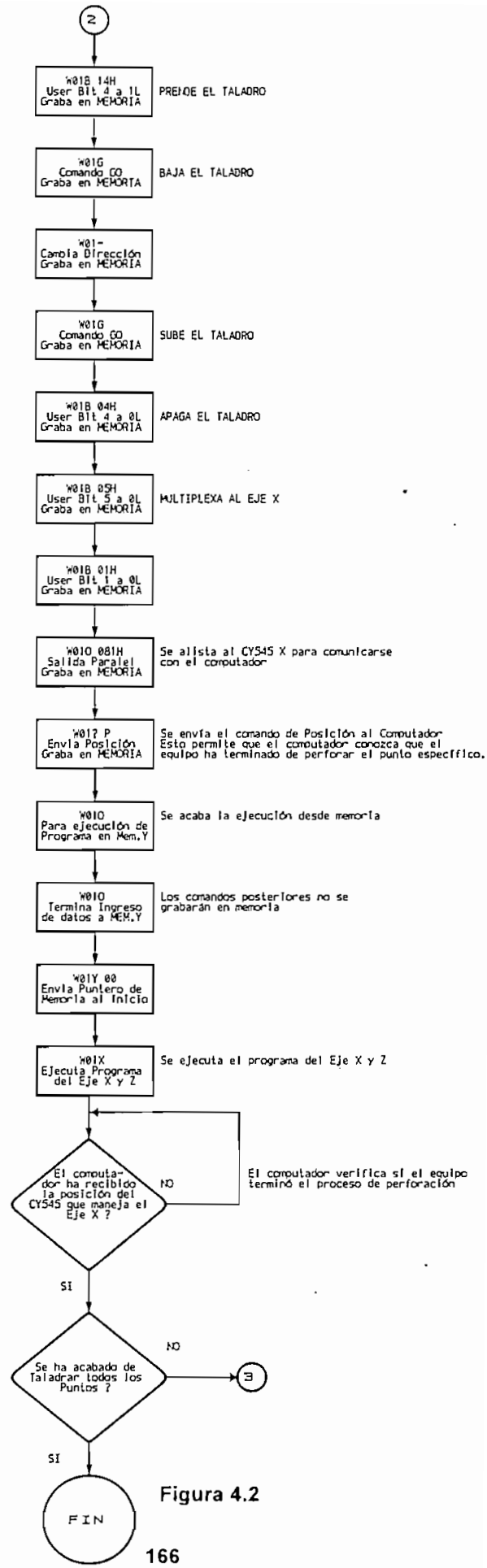

2
W01B 14H
User Bit 4 a 1L
Graba en MEMORIA
PRENDE EL TALADRO
W01G
Comando GO
Graba en MEMORIA
BAJA EL TALADRO
W01-
Cambia Dirección
Graba en MEMORIA
W01G
Comando GO
Graba en MEMORIA
SUBE EL TALADRO
W01B 04H
User Bit 4 a 0L
Graba en MEMORIA
APAGA EL TALADRO
W01B 05H
User Bit 5 a 0L
Graba en MEMORIA
MULTIPLEXA AL EJE X
W01B 01H
User Bit 1 a 0L
Graba en MEMORIA
W010 081H
Salida Paralel
Graba en MEMORIA
Se alista al CY545 X para comunicarse con el computador
W017 P
Envia Posición
Graba en MEMORIA
Se envía el comando de Posición al Computador Esto permite que el computador conozca que el equipo ha terminado de perforar el punto específico.
W01O
Para ejecución de Programa en Mem.Y
Se acaba la ejecución desde memoria
W01O
Termina Ingreso de datos a MEM.Y
Los comandos posteriores no se grabarán en memoria
W01Y 00
Envia Puntero de Memoria al Inicio
W01X
Ejecuta Programa del Eje X y Z
Se ejecuta el programa del Eje X y Z
El computador ha recibido la posición del CY545 que maneja el Eje X ?
NO
El computador verifica si el equipo terminó el proceso de perforación
SI
NO
Se ha acabado de Taladrar todos los Puntos ?
3
SI
FIN

Figura 4.2

El proceso completo de perforación se inicia con la fijación de las variables de dirección para cada uno de los microcontroladores CY233. La una variable, Add1 = "01", es la dirección preestablecida para el primer CY233, mientras que la otra, Add2 = "02", es la dirección preestablecida para el otro microcontrolador CY233. Debe anotarse que estas direcciones han sido seteadas en los dip-switches correspondientes, en las tarjetas de control.

Después, en base al Control de Cliente para selección de archivos que provee Visual Basic, se selecciona el archivo que contiene la información sobre el circuito impreso. Debe anotarse que este archivo debió haber sido convertido previamente, utilizando la opción de Archivos del menú principal, a un formato reconocido por este programa. Por omisión, una vez realizada la conversión, los archivos tendrán la extensión ".TAL". Si se escoge un archivo, el programa tiene la capacidad para reconocer si ha sido generado por este programa; y si no lo es, generará un mensaje de advertencia y retornará al menú inicial.

Escogido el archivo de trabajo y verificado su formato, se lo abre y se obtiene su información, la misma que va a ser almacenada en tres variables:

PosX(i)	contiene la posición en X del punto i.
PosY(i)	contiene la posición en Y del punto i.
Ancho(i)	contiene el diámetro del agujero i.

Luego, se registra la información en un archivo temporal que será el que mantenga los datos del archivo que se está perforando, el subdirectorio en el que se encuentra, y el punto que se está perforando en ese momento.

Con estos datos especificados y con la información en memoria, se inicia el proceso de perforación. Primeramente, se abrirá el pórtico serial, con la siguiente especificación:

2400 baudios
No Paridad
8 bits de datos
1 bit de parada
No señales de handshake

De anotarse que los Controladores de comunicaciones CY233 están seteados para trabajar bajo parámetros similares. Inicialmente se fijan las salidas de los dos CY545 de tal manera que cualquier respuesta de los mismos sea enviada al CY233 y de ahí al computador. El comando que ejecuta esta operación es el siguiente:

W01O 081H<Cr>

W02O 081H<Cr>

Luego, se fijan los parámetros de velocidad inicial, final y pendiente de los CY545. Efectuadas las pruebas correspondientes utilizando la parte manual del programa, se determinó que los parámetros óptimos para el trabajo de los motores de paso son los siguientes:

Velocidad inicial:	10
Velocidad Final:	15.
Aceleración:	170

Por lo tanto, estos son los parámetros que deben ser enviados a los CY545:

W01F 10
W02F 10
W01S 170
W02S 170
W01R 15
W02R 15

Para enviar los datos al pórtico serial se utiliza un función incluida en PDQComm de Crescent Software. Esta función COMPRINT envía los datos y verifica que no exista errores.

Es necesario recalcar que todos los datos enviados anteriormente son ejecutados una sola vez al inicio de la perforación de cada circuito impreso. Los datos que son enviados a continuación, deberán ser enviados a cada uno de los CY545 cada vez que se perfore un agujero.

Debe anotarse también que en la pantalla principal del modo automático existe un control que permite interrumpir el proceso de ejecución del trabajo, y que una vez interrumpido se grabará la información actual de trabajo en el archivo temporal del que se ha hablado anteriormente. En este archivo se grabará el último punto que ha sido perforado. También, cada vez que se inicie la perforación de un nuevo punto, mientras se está perforando normalmente un circuito impreso, se actualizarán los datos del archivo temporal TEMP.TMP.

En cada perforación, primero se fija el contador de la memoria en el CY545 a Cero:

W01Y 00<cr>
W02Y 00<cr>

A continuación se imprime el Display la posición Y que va a ser perforada por el CY545.

Luego, se envían los datos necesarios para que se muevan los motores de pasos que controlan

el eje Y. Estos datos son almacenados en la memoria EEPROM manejada por el CY545, y luego ejecutados. Por lo tanto, primero se envía el comando que hace que todo los comandos recibidos por el CY545 sean almacenados en la memoria:

W02E<cr>

En este punto, se verifica si el punto que se está perforando es el primer punto. En caso de serlo, se deberá enviar el Eje Y a su posición inicial. Esto se lo hace usando el comando HOME y el User Bit 4 del CY545. El momento en que los sensores de efecto Hall han detectado que el bloque de deslizamiento se encuentra cerca a ellos, envian un cero a las dos entradas de una compuerta OR. Con 0l en las entradas, la salida de la compuerta es 1l. Esto es detectado por el CY545, fijando de esta manera la posición inicial del Eje Y.

Luego, se envía el comando que hace que el CY545 mueva el motor a la posición determinada por POSY(i). Esta posición debió ser escalada, como se describió en el capítulo anterior.

Puesto que es necesario que los dos ejes X y Y se hayan detenido para poder bajar el taladro y realizar la perforación, el CY545 que controla el Eje X censa el nivel lógico de la señal USRB0 del CY545 que controla el eje Y. Solamente cuando esta señal se encuentra en un nivel lógico bajo, el CY545 del eje X inicia el proceso de perforación. Por otra parte, el CY545 del eje Y, luego de enviar a un nivel lógico bajo al USRB0, censa el USRB1 del CY545 que controla el eje X. Cuando la señal USRB1 se encuentra en un nivel lógico bajo, el CY545Y envía su señal USRB0 a 1l. De esta manera se controla que los motores de pasos se hayan detenido completamente para iniciar el proceso de perforación. Entonces, se envía al CY545Y el comando que hace que el USRB0 tenga un nivel lógico bajo (Led encendido), y de esta manera se avise al CY545X que este CY545 se ha detenido:

W02B 10H<cr>

Nótese que este User Bit irá a bajo solamente si el CY545Y ha terminado de enviar los pulsos al motor de pasos. Esto garantiza que el User Bit vaya a un nivel lógico bajo solamente cuando el motor se haya detenido.

Luego, el CY545Y deberá sensar el USRB1 del CY545X para verificar si se ha detenido y se puede regresar el USRB0 a su nivel lógico original. Para ello se envía el siguiente comando:

W02W 11H<cr>

Una vez que el CY545Y verifica que la señal USRB1 del CY545X está en bajo, enviará a alto la señal USRB0:

W02B 00H

De esta manera termina el trabajo de este CY545. Como estos datos están siendo grabados en la memoria, es necesario indicar al CY545 que se termina la ejecución de programa desde memoria. Para ello se envía el siguiente comando:

W020<cr>

Y, para que el CY545 termine de escribir los datos que recibe del CY233 en la memoria, se envía el comando:

W02Q<cr>

Este es el momento en el que el CY545Y deberá ejecutar el programa almacenado en la memoria. Puesto que se han grabado datos en ella, el contador de memoria se ha incrementado. Como la primera instrucción que debe ejecutarse está en la dirección 00, se debe enviar este puntero de memoria a 00 con el siguiente comando:

W02Y 00<cr>

Para que el CY545Y ejecute el programa desde memoria, se debe enviar el comando X. Así:

W02X<cr>

Una vez recibido este comando, el CY545Y moverá el motor a la posición indicada por el parámetro de P, fijará un nivel lógico bajo en USRB0 y sensará el nivel lógico del USRB1 del CY545Y. Una vez que este USRB1 tenga un nivel lógico bajo, fijará el nivel lógico de USRB0 a 1l. De esta manera, se habrá movido el CY545Y a la posición especificada.

Una vez que se han ingresado los datos en la memoria EEPROM del sistema que controla el eje Y, se ingresan los datos correspondientes al eje X. Incialmente, se desplegará en el display correspondiente la posición a la que irá el eje X. Luego, se enviará el comando para que los datos enviados al CY545, a través del CY233, sean almacenados en memoria:

W01E<cr>

Entonces se determinará si es el primer punto que va a perforarse. En caso de serlo, se enviará el CY545 X a la posición inicial usando HOME y el User Bit 03.

Luego, el CY545 deberá llevar al motor del eje X a la posición correspondiente, dependiendo del punto que se esté perforando:

W01P PosX(i)<cr>

En donde PosX(i) es la coordenada en X del punto i.

El taladro no puede bajarse hasta que los motores del eje X y Y se hayan detenido completamente. Por lo tanto, este CY545 deberá verificar que el otro CY545 se haya detenido. Para ello se envía el siguiente comando:

W01W 10H<cr>

De acuerdo con esto, se deberá esperar a que el otro CY545 tenga un nivel lógico bajo en su señal USRB0. Esto solo ocurrirá cuando el CY545Y haya detenido el movimiento de los motores en el eje Y.

Entonces, se envía la señal USRB1 a 1l para indicar al CY545Y que se ha detenido el motor X, y el CY545Y llevará a 1l el USRB1.

Como se utilizaron dos controladores CY545, y se manejan tres ejes de coordenadas, se utilizó uno de los dos sistemas para controlar dos ejes, X y Z. Para ello, como ya se describió en capítulos anteriores, se utilizó un demultiplexor de tres estados. Las señales de control de este demultiplexor fueron manejadas por el USRB 5 del CY545 que controla el Eje X. Cuando esta señal tiene un nivel lógico alto, las señales DIR y PULSE del CY545 son direccionadas al motor de pasos que controla el eje X; cuando USRB 5 tiene un nivel lógico bajo, las señales son direccionadas al motor de pasos que controla el eje Z. Una vez movidos los motores del eje X y del eje Y, el taladro estará en la posición adecuada para perforar el punto i. Entonces, CY545X deberá enviar la señal USRB5 a 1l para permitir que el demultiplexor envíe las señales PULSE y DIR del CY545X al motor que controla el movimiento del taladro.

Como el motor debe bajar, se fija una dirección de movimiento semejante al de las manecillas de reloj:

W01+<cr>

Entonces, se deberá bajar el motor el número de pasos especificado por los seteos del Menú

Automático. Luego, se deberá subir el motor el mismo número de pasos que se bajó, para no alterar el registro interno de posición del CY545. Como los CY545 tienen este registro interno de posición, y el CY545X maneja los ejes X y Z, el bajar y subir el taladro el mismo número de pasos asegura que esta posición no haya sido alterada. Por lo tanto, el motor que controla el eje X se moverá desde la posición en que se lo había dejado antes de bajar el taladro. Así, se envían los siguientes comandos:

W01N #Pasos<cr>
W01B 14H <cr>
W01G<cr>
W01B 04H <cr>
W01-<cr>

El User Bit 4 es utilizado para encender el taladro. Esta conectado a un relé de estado sólido el cual conectará 110 voltios al taladro cada vez que reciba la instrucción del CY545. Se debe anotar que el taladro se enciende solamente el momento en que se lo baja y sube.

Debe anotarse también que a pesar de que ha sido alterado el registro del número de pasos, no se lo cambiará. Esto se debe a que los CY545 se mueven a las coordenadas específicas en base a comandos de posición y no de número de pasos. Una vez perforado el agujero, se enviará el comando que haga que el demultiplexor enrute las señales PULSE y DIR al motor del eje X. Esto es, la señal USRB 5 deberá ir a un nivel lógico alto:

W01B 05H<cr>

Como se había llevado USRB1 a 0l para coordinar los movimientos con el USRB1 de CY545Y, se deberá enviarlo a un nivel lógico alto. Para ello, se envía el siguiente comando:

W01B 01H<cr>

Puesto que el computador no debe enviar ningún comando con información sobre nuevas posiciones, hasta que la información actual haya sido procesada por el equipo, se ha desarrollado un sistema para sincronizar el computador con los CY545. Este sistema se basa en el hecho de que una vez perforado el agujero "i", el CY545 que controla el eje X enviará al computador un comando de Query que contendrá la información de su posición actual. El computador analizará esta información, y si reconoce en la tercera posición de la cadena recibida, la letra "P", enviará los comandos para perforar el siguiente punto. En el caso de recibir otro tipo de información, presentará un mensaje en el que se indica que se ha recibido información errónea, y se detendrá el proceso.

Si el CY545 debiera mover el motor desde el inicio del canal al final, se demoraría menos de 20 segundos. Por lo tanto, el computador esperará por la información de Posición, enviado por el CY545X, por un intervalo de 20 segundos. Si después de este tiempo no se han recibido los datos, se detendrá el proceso.

El sistema de sincronización del computador con el equipo se basa en un proceso de chequeo del buffer del puerto serial RS-232. Para ello, se ha utilizado el comando COMLINEINPUT provisto por PDQCOMM. Este comando leerá una línea entera de texto desde el buffer de recepción en una sola operación. Los argumentos de este comando son los siguientes:

ComLineInput(Texto$,Time)

Texto$ recibe información de la próxima línea completa de el buffer de recepción; pero, solamente se chequea el pórtico serial durante Time segundos. Esta función trabaja examinando todos los caracteres en el buffer de recepción del pórtico serial (aquellos que han sido recibidos pero no leídos por el programa), hasta que encuentra un caracter de retorno. Entonces chequea si el próximo caracter es un Line Feed (Alimentación de línea), y lo descarta si no es. Si no encuentra un caracter de retorno, ComLineInput esperará por uno, el tiempo especificado en número de segundos por Time.

De esta manera, si ComLineInput retorna la información enviada por el CY545, y esta información es correcta, se proseguirá con la perforación del siguiente agujero. En caso contrario, se detendrá la perforación.

Por último, una vez perforados todos los orificios, se cerrará el pórtico serial, y se devolverá el control al Menú Automático.

4.2.- Comandos Generales del Microcontrolador CY-545A.

Puesto que ya se analizaron estos comandos en un capítulo anterior, se presenta un resumen de los mismos. Los comandos básicos del CY545 son los siguientes:

A	Fija el registro de posición actual.
B	Fija o limpia un UserBit
C	Modo continuo de pasos
D d	Retraso en Milisegundos
E	Ingreso de comandos a memoria externa
F f	Fija la Velocidad inicial a "f"
G	Go, Movimiento en pasos relativos
H h	Búsqueda de posición de casa (HOME)
I	Inicialización
J a	Salto a la dirección "a"
K	Reservado
L c,a	Salto a una dirección "a", "c" número de veces
M	Reservado
N n	Fija el parámetro del Número de Pasos
O o	Mode
P p	Movimiento a una posición "p"
Q	Salida del ingreso de programa a memoria externa.
R r	Fija la velocidad final a "r"
S s	Fija la pendiente "s" para aceleración y desaceleración
T b,a	Salto a dirección "a" hasta que se produzca "b"
U	Reservado
V	Reservado
W b	Espera a que se alcance el bit "b"
X	Ejecuta comandos en memoria externa
Y a	Puntero de dirección de memoria externa
Z c,a	Salto a una dirección "a" con conteo "c"
+	Dirección de movimiento Sentido de manecillas Reloj
-	Dirección de movimiento contra de manecillas de Reloj
/	Niega o Limpia Valores de Bits
?cmd	Solicitud de parámetro de Comando
0	Para la ejecución de memoria externa
[a,c,d	Comando de cadenas para Display del Leds HP
"chr"	Coloca en el Display la cadena entre comillas

Los comandos descritos anteriormente deberán ir seguidos siempre deun caracter de retorno al final, para que puedan ser interpretados por el CY545 correspondiente.

4.3.- Programa Principal

Como se ha especificado anteriormente, el programa desarrollado en este trabajo tiene cuatro opciones principales: Modo Manual, Modo Automático, Manejo de Archivos y Reinicio. En esta sección se analizará el Menú Inicial y el Modo de Trabajo Automático, que constituyen la parte principal de este programa. En la siguiente sección se analizarán las otras partes del programa, que constituyen partes auxiliares del mismo.

Como se ha desarrollado el programa bajo el entorno de Windows, existe un Icono llamado TALADRO que es el que deberá ser seleccionado para ejecutar el programa.

4.3.1.- Menú Inicial

Una vez cargado Windows, y ejecutado TALADRO, el primer menú que se observa es el menú Inicial. Este menú permite entrar a cada una de las diferentes partes del programa. La forma que contiene este menú inicial se llama INICIAL.FRM. La pantalla de presentación de esta Forma es la siguiente:

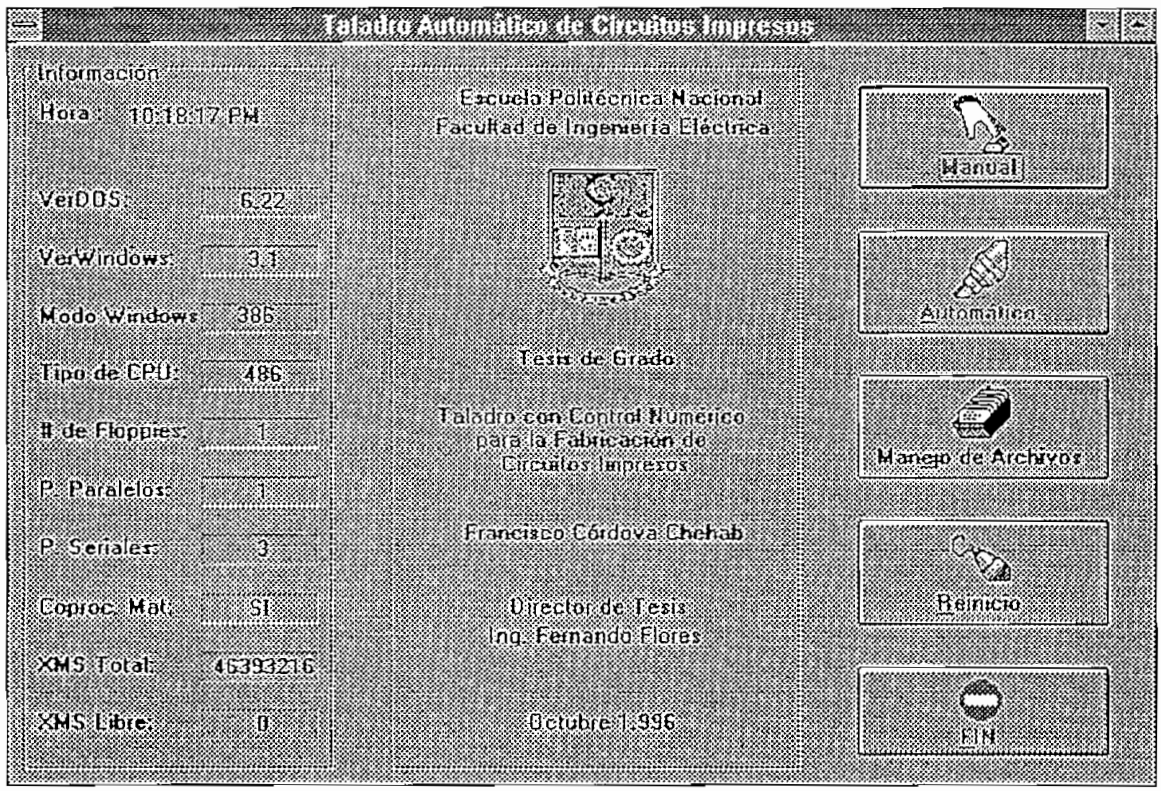


Figura 4.3

Como se puede observar, bajo este menú se podrán escoger las opciones de Modo Manual, Modo Automático, Manejo de Archivos, Reinicio y Fin. La opción de FIN es la que terminará la ejecución del programa.

Adicionalmente, este Menú presenta información sobre el entorno físico y de software sobre el que se está trabajando. Así, se puede observar:

Hora Actual
Versión del Sistema Operativo DOS
Versión de Windows
Modo de trabajo en Windows
Tipo de CPU
Número de Floppies
Número de Pórticos Paralelos
Número de Pórticos Seriales
Coprocesador Matemático
Cantidad de Memoria Total Extendida
Cantidad de Memoria Extendida Libre

Esta información se ha podido obtener en base a algunas subrutinas ejecutadas desde el módulo MODULE1.BAS. Este módulo se ejecuta una vez que se han declarado las subrutinas, funciones y variables globales contenidas en el módulo GLOBAL.BAS, que es el primer módulo que se ejecuta al inicio del programa. Se han listado estos módulos al final de este capítulo.

4.3.2.- Modo Automático

Constituye la parte más importante de este programa. Permite perforar en forma continua un circuito impreso, y controlar la posición de inicio y la distancia de recorrido del taladro.

La forma en la que está almacenado este programa es llamada AUTO.FRM y su diseño gráfico se muestra a continuación:

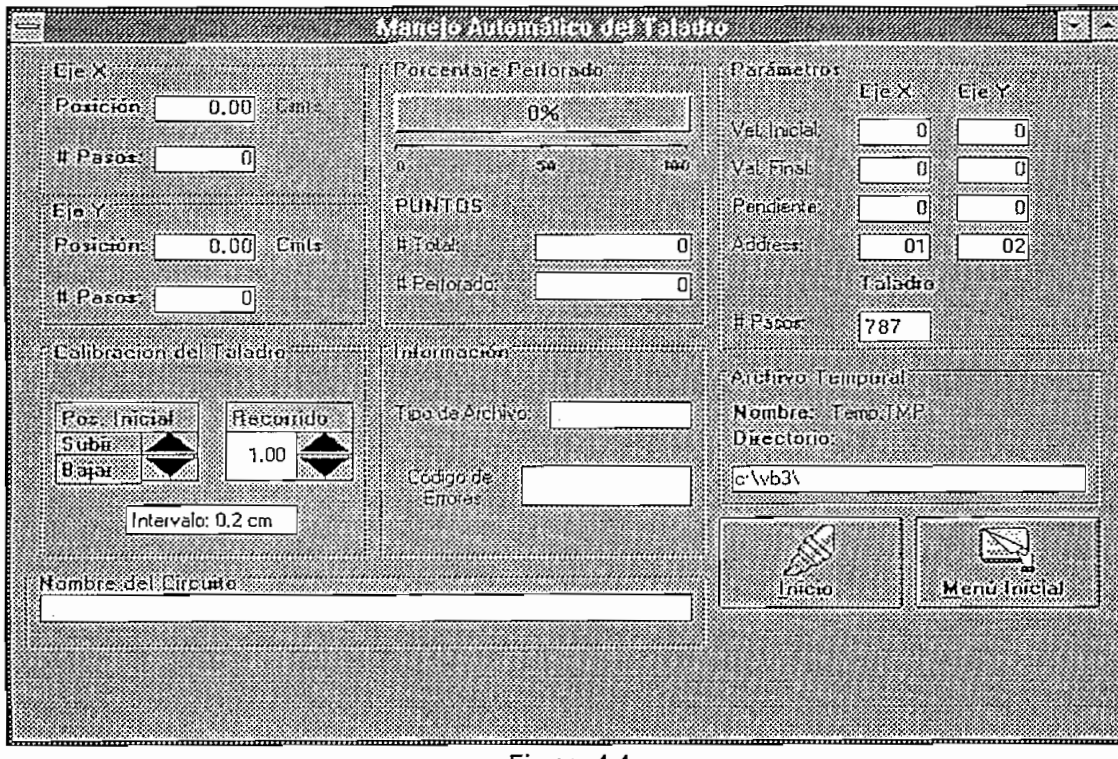


Figura 4.4

Básicamente, esta es una forma de información, puesto que pocos son los parámetros que se pueden controlar a través de ella.

A continuación se analizará cada una de las partes de esta forma:

4.3.2.1.- Calibración del Taladro.

Una sección importante de esta forma es la llamada CALIBRACION DEL TALADRO. Permite fijar, usando el computador, la posición inicial del taladro, y el recorrido. Para ello, se debe fijar el ratón en el control correspondiente para setear cualquiera de los dos parámetros.

Mediante los controles de posición inicial, se puede subir o bajar la posición inicial del taladro. El intervalo prefijado entre cada posición es de 0.2 cm.

Con el control de recorrido, se puede fijar la longitud que debe moverse el taladro para perforar el orificio. El valor mínimo de recorrido es de 1 cm. y el máximo es de 10 cm.; el intervalo entre cada posición es de 0.2 cm. Esta calibración deberá hacerse antes de iniciar el proceso de perforación, puesto que durante el mismo, se deshabilitan estos controles. El número de pasos que deberá moverse el eje Z para bajar o subir el taladro es directamente proporcional a su recorrido; este número se indica en el área de PARAMETROS de esta forma.

Para el diseño de esta sección se usó el control SPIN disponible en Visual Basic. Este control puede ser activado presionando en el botón superior o en el botón inferior, y dependiendo de ello se ejecuta una acción diferente. Las subrutinas asociadas con este proceso son **Spin1_Down, Spin1_Up, Spin2_Down y Spin2_Up.**

4.3.2.2.- Porcentaje Perforado

Esta parte de la forma permite observar visualmente el porcentaje de puntos perforados durante el proceso. Este porcentaje se incrementará cada vez que se perfore un nuevo punto, siendo por supuesto su valor mínimo 0 y su valor máximo 100.

Adicionalmente, se puede observar el número total de puntos y el número de puntos perforados hasta ese momento.

Si se ha utilizado la opción de REINICIO del Menú Inicial, se actualizarán los nuevos valores iniciales de estos controles al inicio del trabajo, dependiendo del punto a partir del cual se inicie nuevamente la perforación.

Iniciado el proceso de perforación, se actualizarán los parámetros de cada uno de los controles cada vez que se haya perforado un agujero.

4.3.2.3.- Eje X y Eje Y

Estos controles permiten observar la posición actual en la que se encontraba el último punto perforado; esto es, las coordenadas del eje X y del eje Y del punto que se perforó. La magnitud de estas coordenadas es Centímetros.

Adicionalmente, se puede observar cuantos pasos se debió mover el eje (X o Y) para llegar a su nueva posición, tomando como referencia la posición anterior.

4.3.2.4.- Parámetros

En esta parte de la forma se indican los parámetros principales de trabajo de cada uno de los CY545. Estos parámetros son: **Velocidad Inicial, Velocidad Final y Pendiente.**

Mediante el uso de la parte Manual de este programa, se estableció los parámetros óptimos para el movimiento de cada uno de los motores. En base a ello se determinó que R=15 es un límite máximo de velocidad para los motores, y que es preferible que la velocidad inicial sea baja. Por otra parte,

velocidades más bajas implican un torque más alto, que es lo que conviene en este tipo de trabajo.

Además, la aceleración deberá ser alta, para lograr que los motores lleguen a la velocidad final en el menor tiempo posible. En base a este análisis se determinaron los siguientes parámetros óptimos de trabajo:

Parámetro de Velocidad Inicial = 10
Parámetro de Velocidad Final = 15
Parámetro de Aceleración = 170

Estos son los parámetros que son mostrados en esta parte de la forma. Además se incluyen las direcciones utilizadas para direccionar a cada uno de los CY233 y el número de pasos que debe moverse el taladro, desde su posición inicial, para perforar el agujero.

4.3.2.5.- Información, Nombre del Circuito y Archivo Temporal

Estas partes de la forma suministran información sobre el archivo escogido.

Información permite conocer el nombre del archivo que contiene los puntos que se están utilizando y un código de errores producidos el momento de seleccionar el archivo. El código de errores no tiene utilidad para el usuario.

Nombre del Circuito contiene el nombre del archivo generado por los programas ORCAD o TANGO, incluyendo la vía de acceso. Este nombre se lo graba el momento de realizar la conversión.

Archivo Temporal contiene el nombre del archivo que se utiliza para mantener temporalmente los datos que permitirán posteriormente poder reiniciar el trabajo con este archivo. El nombre de este archivo es Temp.TMP. El directorio en que se graba depende del directorio de trabajo actual y no puede ser alterado. Es necesario anotar que Temp.TMP contiene información sobre el último archivo que se ha utilizado.

4.3.2.6.- Interrupción

Esta es una parte del programa no visible cuando se muestra la forma. Permite interrumpir el proceso de perforación, cuando el usuario lo desee.

Esta opción aparece cuando se ha iniciado la perforación, y se activa cuando se presiona el botón izquierdo del ratón sobre el cuadrado de control. Tan solo es necesario presionar una vez el botón, para

interrumpir el proceso; pero, en algunos casos, se deberá esperar algunos segundos para tomar nuevamente el control de la forma.

Una vez que se ha interrumpido el proceso, desaparece la opción de interrupción, y se quedarán grabados los datos del circuito y el último punto perforado en el archivo Temp.TMP.

4.3.2.7.- Control de Inicio y Menú Inicial

El control que permite iniciar el proceso de perforado se llama INICIO y está colocado en la parte inferior derecha de la forma. Puesto que este control permite ejecutar el proceso de perforación automático, en su interior se encuentra la subrutina principal de perforación, que se analizó en la sección primera de este capítulo.

El control que permite regresar al menú inicial se llama MENU INICIAL, y se encuentra junto al control INICIO.

Cuando se presiona el control INICIO, se ejecuta la subrutina INICIAR_CLICK. Primero, se muestra la forma que permite seleccionar el archivo que contiene la información sobre el circuito impreso que desea perforarse. Este archivo debió haber sido convertido previamente a un formato reconocido por este programa, utilizando la opción de Manejo de Archivos del Menú Inicial. Si no se ha convertido el archivo, el programa presenta un mensaje con esta información y devuelve el control sobre la forma. Se ha escogido la extensión .TAL para los archivos formateados por el programa de conversión.

Escogido el archivo y verificado su formato, el programa obtiene la información sobre el número total de puntos, las coordenadas X y Y de cada uno de ellos, y el ancho de la herramienta (Broca). Luego, se determinan los parámetros que se guardarán en el Archivo Temporal.

Es entonces, cuando se llama a la subrutina principal de este programa: TALADRAR, con parámetros MinPuntos (Número del punto desde el que deberá comenzar a taladrarse). Esta es la subrutina que se ha analizado en la sección primera de este capítulo, y que comanda al equipo para que se perforen los agujeros.

Una vez perforados los puntos, el programa devuelve el control a la forma de modo Automático, y se podrán ejecutar otros procesos de perforación.

Cada vez que se envía un comando al CY545, puesto que tiene que ser analizado por el CY233, el cual lo repetirá o lo transferirá al CY545 dependiendo de la dirección, es necesario producir una pausa en la ejecución del programa. Esto se logra mediante el uso de la Función PAUSE incorporada en PDQCOmm. Esta función tiene como parámetro un número entero que representa un retraso de

Número/18 segundos.

4.4.- Programas Auxiliares y Subrutinas.

Los programas auxiliares son aquellos que permitirán manejar y calibrar el equipo, así como adecuar los archivos para que puedan ser utilizados en la perforación. Los programas auxiliares son los siguientes:

4.4.1.- Modo Manual

Este Modo es el que permite manejar manualmente el equipo. Esto quiere decir que se podrán controlar cada uno de los motores en forma independiente, pudiéndose realizar diferentes acciones para cada uno de ellos.

Se debe anotar que se diseñó esta forma considerando que se van a manejar dos Microcontroladores CY545, y por lo tanto dos sistemas generales. Por ello, se deberá escoger con cual de los microcontroladores se desea trabajar, y cualquier acción que se ejecute tendrá efecto sobre el Microcontrolador escogido.

La forma en la que está almacenado este programa se llama MANUAL.FRM y su diseño gráfico es el siguiente:

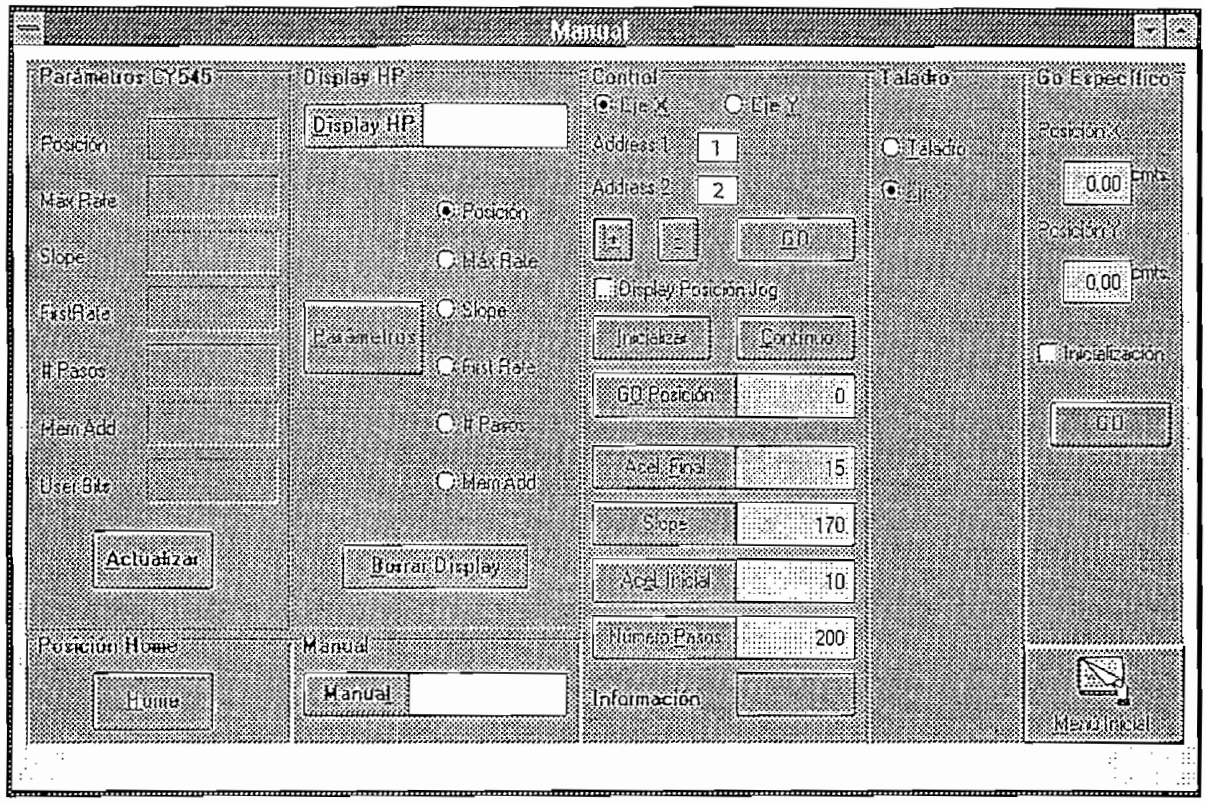


Figura 4.5

Como se puede observar, esta forma tiene varias partes, las que se pueden reconocer por medio del título colocado en la parte superior de las mismas.

Se va analizar cada una de ellas, y cuando sea necesario se presentará el código correspondiente.

4.4.1.1.- Parámetros del CY545

Esta opción, localizada en la parte superior izquierda de la forma, permite observar los parámetros del CY545 seleccionado.

Los parámetros que se presentan son los siguientes:

Posición.....	POSICION
Velocidad Final.....	MAX RATE
Aceleración.....	SLOPE
Velocidad Inicial.....	FIRST RATE
Número de Pasos.....	# PASOS
Púntero de Memoria.....	MEM ADD
Estado de User Bits.....	USER BITS

Es necesario recalcar que cada uno de estos parámetros está almacenado en registros internos del CY545.

Al presionar el control ACTUALIZAR, se actualizarán los parámetros del CY545 escogido. La subrutina que permite obtener estos datos , y que ha sido utilizada en varias partes en esta forma, es llamada ACTUALIZACION. Esta subrutina se basa en el comando Query que permite obtener información sobre cada uno de los parámetros del CY545. Tiene como parámetro, el número de parámetros que se desea leer.

Cada vez que se realiza una operación que tiene que ver con el movimiento de los motores, y por lo tanto con el cambio de alguno de los parámetros, se ejecuta esta subrutina para actualizar los datos. Para optimizar el tiempo de ejecución, solo se leerán los parámetros que hayan sido cambiados, dependiendo de la acción que haya sido ejecutada. Esto se controla con el parámetro que debe recibir la subrutina ACTUALIZACION.

4.4.1.2.- Display HP

Esta parte de la forma permite visualizar, en el display HP, el parámetro escogido. Los parámetros visualizados serán los correspondientes al CY545 escogido. Estos parámetros pueden ser:

- Posición
- Velocidad Final (Max Rate)
- Pendiente (Slope)
- Velocidad Inicial (First Rate)
- Número de Pasos (# de Pasos)
- Puntero de Memoria (Mem Addr)

Una vez seleccionado el parámetro, se deberá presionar el control PARAMETROS para poder visualizarlo en el Display del CY545 correspondiente. Si no se presiona el Control Parámetros, no se visualizará.

Se debe anotar que una vez visualizado el mensaje en el Display, este queda retenido en el mismo, hasta que se envíe un comando que lo deshabilite y borre. Esto se hace seleccionando como salida del CY545 el pórtico paralelo y no el Display; para ello, se debe usar el comando Mode.

Para borrar el dispaly del CY545 seleccionado, se puede usar el Control BORRAR DISPLAY. Una vez presionado, ejecuta una subrutina que selecciona como salida el interfase paralelo del CY545, y por lo tanto, deshabilita el Display, borrándolo. Esta subrutina es llamada BORRADO.

Adicionalmente, se ha provisto al programa de una opción para visualizar en el Display cualquier cadena de caracteres que el usuario envíe. Para ello, se deberá escribir en el control de texto localizado junto al control DISPLAY HP. El control de texto no permite el ingreso de cadena con más de 8 caracteres, puesto que es el número máximo de caracteres que pueden visualizarse en el Display. Una vez escrito el mensaje, se debe presionar el control Display Hp para visualizarlo en el display del CY545 seleccionado. La subrutina que se ha utilizado se llama DISPLAY_CLICK; inicialmente, envía un comando MODE para seleccionar como salida del CY545 el Display HP. Luego, se envía entre comillas el texto contenido en el control de texto que se encuentra junto al control DISPLAY HP.

4.4.1.3.- Control Manual

Se ha diseñado un control que permita enviar, al CY545 seleccionado, los comandos en forma manual, tendiendo en cuenta que no se deberá incluir la dirección puesto que esta será añadida dependiendo del CY545 escogido.

Se podrá enviar cualquier comando válido a los CY545. Los comandos deben ser escritos en el control de texto que se encuentra junto al control MANUAL. Para enviarlo, se deberá presionar el botón en este control.

Este Control es útil puesto que de esta manera se pueden probar y ejecutar acciones que, tal vez, no han sido incluidas en el desarrollo de este programa.

4.4.1.4.- Programa de CONTROL

Esta es la parte más importante del control Manual. Permite seleccionar el CY545, cambiar las direcciones válidas para los CY233, cambiar los parámetros de trabajo y producir movimientos.

En esta parte de la forma se selecciona el CY545 con el que se va a trabajar. Para ello, se deberá presionar el botón derecho del ratón sobre cualquiera de los dos controles de selección **1 CY545** y **2 CY545**. Al seleccionar el microcontrolador con el que se va a trabajar, se añadirá a los comandos la dirección "01" o "02", dependiendo del CY545. Esta dirección se utilizará en todas las partes de esta forma. Las subrutinas que fijan la dirección del CY545 con el que se va a trabajar se llaman **CY5451_CLICK** y **CY5452_CLICK**. Adicionalmente, cada vez que se selecciona un nuevo CY545, se actualizan los parámetros correspondientes.

Además, esta forma muestra las direcciones actuales de los CY233, en los controles de texto **Address 1** y **Address 2**.

Existen dos controles que permiten escoger la dirección de movimiento de los motores, + y -. "+" selecciona el sentido de las manecillas del reloj, y "-" selecciona el sentido en contra de las manecillas del reloj. Las subrutinas relacionadas con estos controles son **Mas** y **Menos.**

Nótese que esta selección solo define el sentido de movimiento de los motores, pero no los mueve. Para ello se deberá seleccionar el comando **GO** localizado junto a los controles de selección de movimiento. Este comando moverá el motor el número de pasos especificado en el parámetro interno del CY545, en la dirección especificada. Nótese que luego de movido el motor, se actualiza el parámetro de posición. La subrutina utilizada es llamada **Go_Click.** Además, si el control **Display Posición JOG** está activado, el comando GO enviará al CY545 la orden de visualizar la posición cada vez que se mueva los motores utilizando el switch de JOG.

Como ya se ha analizado, el Control DISPLAY POSICION JOG permitirá que se visualice la posición del CY545 cuando se lo mueve manualmente.

Existe otro control que permite inicializar el CY545 seleccionado. Este control llamado **INICIALIZAR** envía al CY545 el comando I de inicialización que reseteará por software el microcontrolador. Por lo tanto, se fijarán los parámetros por omisión:

Posición:	00000000
Velocidad Inicial:	14
Aceleración:	220
Velocidad Final:	100
# de Pasos:	200
Puntero de Memoria:	1

La subrutina asociada a este control se llama **INICIALIZAR.** Se incluye en esta subrutina una llamada a la subrutina de Actualización, para refrescar los parámetros visualizados en la forma.

Otro control relacionado con el movimiento se llama **CONTINUO.** Una vez seleccionado, envía al CY545 el comando de movimiento continuo C, que hará que se mueva indefinidamente el motor hasta que se encuentre un censor que lo detenga. La subrutina se llama **CONTINUO_CLICK.**

El último control que hace que los motores se muevan es llamado **GO_POSICION.** Este control mueve el motor a la posición determinada por los datos que se encuentran en el control de texto ubicado junto a él. Para ello, se envía un comando "P" acompañado del parámetro leído en el control de texto. La subrutina asociada se llama **GO_POSICION,** y también actualiza la información sobre los parámetros.

Además, existen cuatro controles que permiten fijar algunos parámetros en los CY545. Estos parámetros son: Velocidad Final (Vel. Final), Aceleración (Slope), Velocidad Inicial (Vel. Inicial), y Número de Pasos (Número de Pasos). Cada uno de estos controles tiene fijado inicialmente en los controles de texto ubicados junto a ellos, el valor por omisión determinado experimentalmente para cada uno de los parámetros. Estos valores son:

Velocidad Inicial:	10
Velocidad Final:	15
Aceleración:	170
Número de Pasos:	200

Cualquiera de estos parámetros puede ser modificado, cambiando el valor del control de texto y presionando el control correspondiente. Entonces se enviará el comando adecuado para modificar el parámetro escogido. Debe anotarse que los controles de texto solo permiten el ingreso de números, y no se podrán escribir valores negativos.

Las subrutinas que realizan estos trabajos son las siguientes:

Velocidad Inicial	Sub FirstRate_Click ()
Velocidad Final	Sub MaxRate_Click ()
Inclinación	Sub Slope_Click ()
Número de Pasos	Sub NumPasos ()

Cada una de estas subrutinas, una vez ejecutada, actualiza los datos de los parámetros de información de los CY545 en la forma. Eso es necesario, puesto que al alterar cualquiera de los parámetros, las condiciones de trabajo de los microcontroladores son diferentes. Además, esta actualización permite verificar el estado del CY545, analizando la información obtenida.

4.4.1.5.- Control Manual del Taladro.

Esta parte de la forma, ubicada en el extremo derecho, permite calibrar el taladro, subiéndolo o bajándolo la distancia determinada en el control de distancia provisto en la forma.

Esta parte de la forma dispone de los siguientes controles:

- Taladro
- Eje X
- Subir
- Bajar

- Control de texto para distancia
- Control de subida o bajada de la distancia
- Control CMTS. para enviar la distancia.

Al iniciar el trabajo en la forma, los únicos controles visibles son **TALADRO** y **EJE X** (que tiene el control de selección); pero, el único control habilitado es TALADRO, puesto que es la única opción que se puede ejecutar con respecto al taladro, en ese momento. Debe notarse, que se puede ejecutar la opción TALADRO, solamente si se ha escogido el CY545 1, puesto que controla el eje X y Z. Si no se ha escogido este CY545, no se podrá escoger ninguna opción de trabajo con el Taladro.

Al escoger la opción TALADRO, esta asume el control de selección y se deshabilita, pudiendo ser presionado solamente el control del EJE X. Pero, se pueden visualizar todos los controles que permiten subir y bajar el taladro: SUBIR, BAJAR, CMTS, SPIN1 y CONTROL DE TEXTO de distancia. Además, se guardan en memoria los datos que tenía el CY545 1, antes de iniciar el trabajo con el Taladro, las que serán actualizadas nuevamente, cuando se termine el trabajo de perforación manual.

En el momento de la visualización, en la parte de MOVIMIENTO, **SUBIR** tiene el control de selección. Esto obliga a que solamente se pueda ejecutar la opción de bajar el taladro.

Al presionar el control **BAJAR**, el motor que controla el eje Z bajará el taladro, la distancia especificada por el Control de Texto de la distancia. Además, asumirá el control de selección. La subrutina asociada con BAJAR se llama **BAJAR_CLICK ()**.

Como BAJAR tiene el control de selección, la única opción que se puede ejecutar es **SUBIR**. Esta opción subirá el taladro, el mismo número de pasos que se lo bajó. La subrutina utilizada se llama **BAJAR_CLICK ()**.

Usando el control **SPIN1**, se puede incrementar o decrementar la distancia de recorrido del taladro. Para ello, se deberá presionar el botón de subida o de baja en el SPIN, y el cambio de valor se observará en el Control de Texto de Distancia. El valor mínimo de distancia es de 0.6 cmts y el máximo es de 10 cmts. El intervalo de cambio es de 0.2 cmts. Escogida la distancia, se deberá presionar el control **Cmts.** para que se envíe el número de pasos proporcionales al CY545. Si no se presiona este control, no se ejecutará ninguna acción y la distancia de recorrido será la anterior.

Las subrutinas que manejan el SPIN 1 son llamadas **SPIN1_UP ()** y **SPIN1_DOWN()**. La subrutina asociada con el control de la distancia de perforación del taladro se llama **DistanciaTaladro_Click ()**.

Cuando se ha terminado el trabajo manual con el taladro, se deberá escoger la opción EJE X. Si la opción BAJAR tiene el control de selección, el programa generará un mensaje en el que se indicará que

el taladro está ABAJO, y si se desea subirlo. Al responder afirmativamente, se regresará al control del Taladro y se podrá subirlo. En caso contrario, se actualizarán los datos anteriores del CY545. Si el taladro no se ha subido, el valor del parámetro de posición del CY545 no será igual al que tenía mientras se trabajaba con el taladro. La diferencia será el número de pasos equivalente a la distancia de recorrido del taladro. Pero, si se escoge la opción EJE X, con SUBIR teniendo el control de selección, la posición anterior del CY545 será la misma que la observada cuando se trabajaba con el taladro.

4.4.1.6.- Go Específico

Esta parte la forma permite ir a una posición específica X y Y al mismo tiempo. Para ello, tan solo basta con indicar en POSICION X y POSICION Y las coordenadas en centímetros. La subrutina Command3D1.Click () se encargará de ejecutar esta operación, una vez que se haya verificado si se debe ir a la posición HOME. Para ello verificará si el usario ha seleccionado el botón de INICIALIZACION.

4.4.1.7.- Menú Inicial

Con el uso de este control, es posible regresar al Menú Inicial, ocultando la forma MANUAL.FRM.

4.4.2.- Manejo de Archivos

Esta forma permite realizar la conversión de los archivos generados por TANGO y ORCAD a un formato reconocido por este programa. Para ello, se podrán escoger diferentes opciones, entre ellas, realizar el proceso de conversión con diferentes tipos de ordenamiento en cada uno de los ejes, convertir las coordenadas X a coordenadas Y y viceversa, y mostrar el archivo convertido.

El nombre de la forma que contiene la información es ARCHIVOS.FRM, y su diseño es el siguiente:

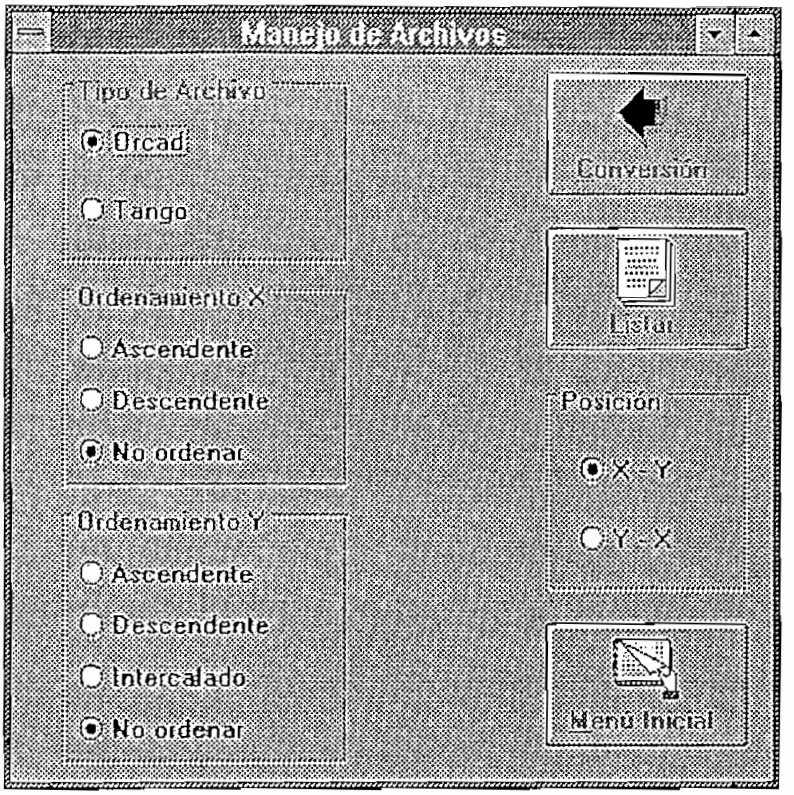


Figura 4.6

Las partes de esta forma se detallan a continuación:

4.4.2.1.- Tipo de Archivo

Permite seleccionar el tipo de archivo que desea convertir. Este programa puede usar archivos ASCII generados por ORCAD y TANGO.

Al inicio del programa, la opción ORCAD tiene el control de Selección. Si se desea convertir un archivo de TANGO, se deberá seleccionar la opción correspondiente.

El momento de realizar la conversión, el programa verifica que el archivo seleccionado haya sido generado por TANGO o por ORCAD. Si se ha seleccionado como tipo de archivo ORCAD y se desea convertir una archivo de TANGO, el programa generará un mensaje en el que se indica que el archivo escogido y el tipo no corresponden. Igual sucede, si se escoge un tipo de Archivo TANGO, y se trata de convertir una archivo de ORCAD.

No es posible convertir archivos generados por otros programas, a menos que se diseñe un traductor que convierta el archivo generado en ASCII a un archivo de formato semejante al utilizado por este programa. La información sobre este formato se encuentra en el capítulo Segundo.

4.4.2.2.- Ordenamiento X y Ordenamiento Y.

Puesto que los motores deben ser movidos a la coordenada específica del punto a perforarse, se puede optimizar las distancias recorridas por los mismos, de tal manera que el tiempo utilizado para perforar los mismos sea el mínimo. Para ello, se deberán ordenar los puntos en la forma más adecuada.

Se pueden escoger diferentes formas de ordenamiento de los puntos, tanto en el eje X como el eje Y. Algunos posibilidades de combinación entre el ordenamiento X y Y no pueden ejecutarse, y el programa las detectará cambiando automáticamente la selección.

En el eje X, se pueden escoger los siguientes tipos:

ASCENDENTE: permite ordenar los puntos en forma ascendente, basado en la coordenada del eje X. En el caso de puntos con igual coordenada X, estos se ordenarán de acuerdo a la selección de ordenamiento del eje Y, la cual puede ser ascendente, descendente, intercalada o sin orden.
DESCENDENTE: En este caso, los puntos se ordenarán en forma descendente, de acuerdo a la coordenada X. De la misma forma que en el caso anterior, si existen puntos con coordenadas X iguales, estos se ordenarán de acuerdo a la selección de ordenamiento del Eje Y, que puede ser Ascendente, Descendente, intercalado o No ordenar.
NO ORDENAR: No se ordenarán los puntos, dejándolos en el orden en el que se encontraban en el archivo ASCII. Al escoger esta opción, la selección de ordenamiento en el eje Y se cambiará automáticamente a NO ORDENAR. Esto quiere decir, que el archivo será convertido en el mismo orden como fue generado por el programa de diseño. La subrutina asociada es llamada **NoordenarX_Claque (Value as Integer).**

En el caso del ordenamiento en el Eje Y, se tienen las siguientes opciones:

ASCENDENTE: Se ordenarán los puntos que tengan la misma coordenada X, en forma ascendente de acuerdo a la coordenada Y. Si en el ordenamiento X se ha escogido NO ORDENAR, el programa cambiará automáticamente la selección a ASCENDENTE en Y, puesto que no se permite esta opción. La subrutina asociada se llama **AscendenteY_Claque (Value as Integer).**
DESCENDENTE: En este caso, se ordenarán los puntos que tengan igual coordenada X, en forma descendente de acuerdo a la coordenada X. De la misma manera, si se detecta que se ha seleccionado NO ORDENAR en X, el programa cambiará esa selección a ASCENDENTE en X. La subrutina es llamada **DescendenteY_Claque (Value as Integer).**
INTERCALADO: con esta opción seleccionada, se ordenarán cada grupo de puntos que tengan la misma coordenada X, en forma intercalada. Esto quiere decir que el primer grupo de puntos con igual coordenada X se ordenará en forma ascendente con respecto a la coordenada Y. El segundo grupo, se ordenará en

forma descendente con respecto a Y. El tercero, en forma ascendente, y así sucesivamente. Tal vez, esta es la forma óptima de ordenamiento. Si al seleccionar INTERCALADO, el programa detecta que se ha escogido NO ORDENAR en X, cambiará la forma de ordenamiento a ASCENDENTE en X. La subrutina que ejecuta esta acción es INTERCALADOY_CLICK (Value as Integer).
NO ORDENAR: En este caso, no se ordenarán los puntos con respecto a la coordenada Y. Al escoger esta opción, el programa selecciona NO ORDENAR en X automáticamente. La subrutina que ejecuta está acción se llama NoordenY_Claque (Value as Integer).

4.4.2.3.- Posición

Esta parte de la forma permite intercambiar las coordenadas X con las coordenadas Y, en el archivo de trabajo. En base a ello, el circuito impreso deberá colocarse en forma invertida en el tablero del equipo para ser perforado.

Existen dos opciones:

X-Y: Esta es la opción seleccionada por omisión. No intercambia las coordenadas del archivo original.
Y-X: Esta opción intercambiará las coordenadas del archivo. Durante el proceso de conversión se detectará que esta opción está activada, y se producirá el intercambio.

4.4.2.4.- Listar Puntos.

Esta opción permite listar el archivo que seleccionado, mostrando en número de puntos, el Diámetro del agujero, la coordenada X y la coordenada Y. Además, indicará el tipo de archivo original, que puede ser ORCAD o TANGO.

La subrutina relacionada se llama LISTAR_CLICK ().

Al seleccionar esta opción, deberá seleccionarse el archivo que se va a observar. Debe anotarse que la extensión de los archivos convertidos por el programa es .TAL, y estos se deberán escoger.

Una vez escogido el archivo, será mostrado en la pantalla. Para desarrollar esta opción, se utilizó el control de GRILLA provisto en Visual Basic. El número de columnas en la grilla es de cuatro, y el número de filas está determinado por el número de puntos listados. Si en la pantalla no se pueden observar todos los puntos del archivo, se puede utilizar el control de scroll provisto en la grilla.

En un control de Texto, se colocará el tipo de archivo que ha sido utilizado.

Dentro de esta forma se ha incluído un botón de IMPRESION el cual permitirá imprimir el archivo que se está listando.

4.4.2.5.- Conversión

Esta es la parte más importante de esta forma. Al presionar el control **CONVERSION**, se convertirá el archivo seleccionado a un formato propio. Existen varias subrutinas asociadas con este control, siendo la principal **CONVERSION_CLICK ()**.

Seleccionado este control, se ejecuta esta subrutina. Primero, se debe escoger el archivo que desea convertirse. La extensión por omisión para estos archivos es **.DRI**. Seleccionado el archivo, el programa verificará que sea un archivo generado en ASCII por el programa TANGO o el programa ORCAD. En caso de no serlo, genera un mensaje de aviso y devuelve el control a la forma Archivos.FRM.

En caso de ser un archivo compatible, comprueba si es el tipo de archivo que se desea convertir. Esto es, comprueba si es un archivo del formato seleccionado en TIPO DE ARCHIVO, que puede ser ORCAD o TANGO. Si no lo es, genera un mensaje de error y regresa al menú de archivos.

Los archivos se reconocen de la siguiente manera:

ARCHIVO DE ORCAD: para reconocer este tipo de archivos, se buscará la palabra ASCII en la primera línea. Si no se la encuentra, el archivo no fue generado por ORCAD.

ARCHIVO DE TANGO: este tipo de archivos contienen en la segunda línea la palabra METRIC. Si no contiene esta palabra, el archivo no es de TANGO, o las unidades no son métricas.

Una vez verificado el tipo de archivo, se analiza el control de posición. Si el control Y-X está activo, se intercambiarán las coordenadas de los puntos. Para ello, se llamará a la función SWAP, disponible en PDQComm. Si no está activo este control, no se intercambiarán las coordenadas de los puntos.

Luego, se analizará el tipo de ordenamiento que debe realizarse, de acuerdo con el control activado en ORDENAMIENTO X y ORDENAMIENTO Y. Primero, se ordenarán los puntos de acuerdo al tipo de orden escogido para el eje X, y luego, se ordenarán los puntos con similar coordenada en X de acuerdo al tipo de orden escogido para el eje Y. Para ello, se utilizarán las siguientes Subrutinas:

SWAPX (): Esta subrutina intercambia las coordenadas X, Y y el diámetro de punto "i", por las coordenadas X, Y y diámetro del punto "i-1".

SWAPXinter (): Esta subrutina realiza el intercalado de los puntos, cuando se ha escogido como opción

en el ordenamiento Y.

SWAPYasc (): permite ordenar en forma ascendente los puntos con coordenadas X iguales y Y diferentes.

SWAPYdesc (): permite ordenar en forma descendente los puntos con coordenadas X iguales y Y diferentes.

Una vez ordenados los puntos de acuerdo al tipo de ordenamiento escogido en los dos ejes, el programa solicitará el nombre con el que se desea grabar el archivo ya convertido. Escogido el nombre, se grabará el archivo y se generará un mensaje indicando que **Se ha convertido y grabado el archivo.** Entonces, se regresará el control a la forma.

4.4.2.6.- Menú Inicial

Este control permite regresar al Menú Principal del programa. Entonces, se esconderá la forma ARCHIVOS.FRM y se mostrará la forma INICIAL.FRM.

4.4.3.- Reinicio

Por medio de esta forma se puede reiniciar un proceso de perforado, que ha sido interrumpido en la forma automática.

La forma que contiene este proceso se llama **REINICIO.FRM**, y su diseño es el siguiente:

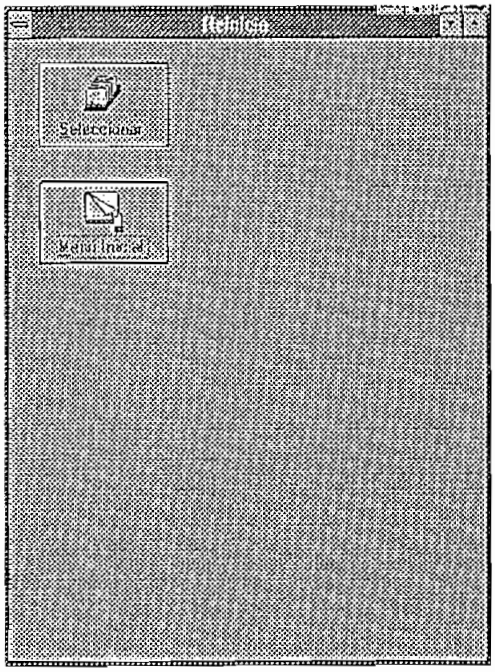


Figura 4.7a

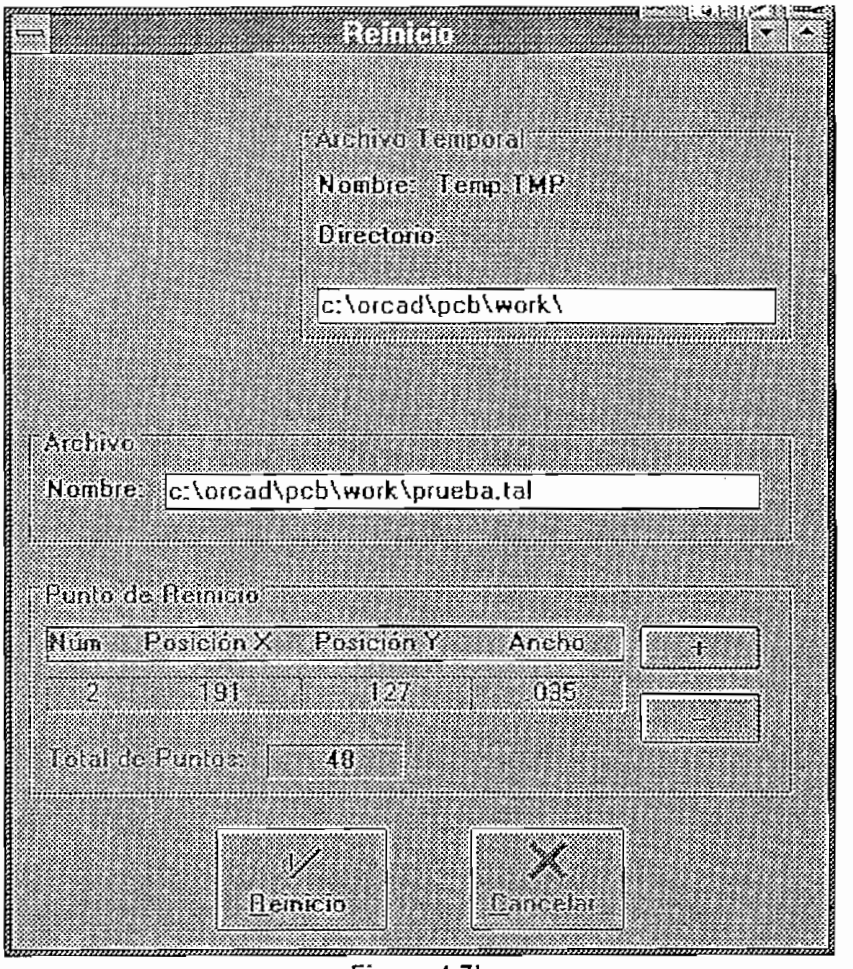


Figura 4.7b

Al iniciar el trabajo de REINICIO solo aparecerán dos controles ESCOGER ARCHIVO y MENU INICIAL. Los demás controles no se visualizarán hasta que se haya seleccionado el archivo temporal. La subrutina asociada con este control se llama ESCOGERAR_CLICK ().

Presionando el Control ESCOGER ARCHIVO, el programa permitirá seleccionar el archivo temporal que se desee utilizar. Como se dijo anteriormente, este archivo contiene información acerca del nombre del archivo convertido, del nombre de archivo original, y del punto último punto perforado antes de iniciar el trabajo.

Seleccionado el archivo, se verifica si la información contenida en él es válida. Para ello, se busca en la primera línea del archivo la palabra "TEMPORAL", grabada como parte del archivo.

Comprobada la validez del archivo, y obtenida su información, se carga el archivo que contiene la información de los puntos que estaban siendo perforados, y se verifica que su formato sea válido; es decir, que sea generado por el programa de conversión.

Entonces, se podrán visualizar las partes escondidas de la forma:

ARCHIVO TEMPORAL que contiene la información sobre el nombre del archivo Temporal, y el subdirectorio donde se encuentra.

ARCHIVO que contiene el nombre del archivo que se está perforando, incluyendo la vía de acceso.

PUNTO DE REINICIO permitirá escoger el punto desde el cual se desea reiniciar el trabajo de perforación. En esta parte de la forma se puede visualizar el número del punto, la posición X, la Posición Y, el Diámetro del agujero y el Número total de puntos. Además existen dos control que permiten visualizar el punto inmediatamente superior o inmediatamente inferior del archivo. Estos son + y -, asociados a las subrutinas Mas_Click () y Menos_Click ().

Una vez determinado el punto desde el cual se desea reiniciar el trabajo, se deberá presionar el control REINICIO, asociado con REINICIO_CLICK (), el cual entregará el control del programa a la forma de modo automático AUTO.FRM, enviando como parámetro la variable MINPUNTOS que contiene el número del punto desde el cual se va a seguir perforando. Al visualizarse la forma AUTO.FRM, se deberá seleccionar el control INICIO, el cual reconocerá automáticamente que se trata de un reinicio de perforación, y comenzará a perforar el circuito impreso desde MINPUNTOS.

4.4.4.- Subrutinas.

A continuación, se listarán las subrutinas y programas globales utilizados en la elaboración de este programa. Cada una de ellos ha sido descrito en las secciones anteriores.

4.4.4.1.- Programas y Declaraciones Globales

Global.BAS

```
DefInt A-Z
Declare Sub StartBreak Lib "PDQCOMM.DLL" ()
Declare Sub EndBreak Lib "PDQCOMM.DLL" ()
Declare Sub ComPutBack Lib "PDQCOMM.DLL" (Char$)
Declare Sub SetComErr Lib "PDQCOMM.DLL" (ByVal ErrNum%)
Declare Sub OpenCom Lib "PDQCOMM.DLL" (Param$)
Declare Sub ComPrint Lib "PDQCOMM.DLL" (Text$)
Declare Sub Closecom Lib "PDQCOMM.DLL" ()
Declare Sub RTS Lib "PDQCOMM.DLL" (ByVal Count%)
Declare Sub FloppyType Lib "QPRO200.DLL" (Drivea%, DriveB%)
Declare Sub SetDrive Lib "QPRO200.DLL" (ByVal Drive$)
Declare Sub SortS Lib "QPRO200.DLL" (Array!, ByVal NumEls%, ByVal Direction%)
Declare Sub GetEquip Lib "QPRO200.DLL" (Floppies%, Parallels%, Serials%)
Declare Sub ProperName Lib "QPRO200.DLL" (Work$)
Declare Sub Swap Lib "QPRO200.DLL" (Data1 As Any, Data2 As Any, ByVal TypeCode%)
```

```
Declare Function ComInput$ Lib "PDQCOMM.DLL" (ByVal Count%)
Declare Function ComErr% Lib "PDQCOMM.DLL" ()
Declare Function ComLoc% Lib "PDQCOMM.DLL" ()
Declare Function ComEof% Lib "PDQCOMM.DLL" ()
Declare Function DiskRoom& Lib "QPRO200.DLL" (ByVal Drive$)
Declare Function DiskSize& Lib "QPRO200.DLL" (ByVal Drive$)
Declare Function DosVer% Lib "QPRO200.DLL" ()
Declare Function GetVol$ Lib "QPRO200.DLL" (ByVal Drive$)
Declare Function Exist% Lib "QPRO200.DLL" (ByVal FileName$)
Declare Function GetCPU% Lib "QPRO200.DLL" ()
Declare Function GetDir$ Lib "QPRO200.DLL" (ByVal Drive$)
Declare Function GetDrive% Lib "QPRO200.DLL" ()
Declare Function MathChip% Lib "QPRO200.DLL" ()
Declare Function ModelInfo$ Lib "QPRO200.DLL" ()
Declare Function WinVer% Lib "QPRO200.DLL" ()
Declare Function Fre& Lib "QPRO200.DLL" (ByVal Flag%)
Declare Function PDQTimer& Lib "PDQCOMM.DLL" ()
Declare Function Compact$ Lib "QPRO200.DLL" (Old$)

'Global Const TRUE = -1
'Global Const FALSE = 0
Global NumParametros As Integer
Global PosicionPasos As String
Global PosicionejeX As String

Global Add1 As String
Global Add2 As String

Global Dato As String
Global Texto As String
Global Address As String
Global Cr  As String

Global Parametro(9) As String
Global ParametroDisplay As String

Global ArchivoTemp As String
Global Compl(500) As String
Global Ancho(500) As Double
Global PosX(500) As Double
Global PosY(500) As Double
Global Posx1(500) As Double
Global Posy1(500) As Double
Global NumPuntos As Integer
Global Numeropuntos As Integer
Global MinPuntos As Integer
Global Nombre As String
Global Control As Integer
Global Escritura As String
Global CentimetX As String
Global CentimetY As String
Global NombreListar As String
```

Module1.BAS

```
Sub Main ()
' VERSION DE DOS
   Form2.VerDos.Caption = Str$(DosVer%() / 100)
' MODO DE WINDOWS
   Form2.Modo.Caption = ModeInfo$()
' VERSION DE WINDOWS
   Form2.VerWindows.Caption = Str$(WinVer%() / 100)
' TIPO DE CPU
   Form2.TipoCPU.Caption = Str$(GetCPU%())
' CONFIGURACION DEL EQUIPOS
   GetEquip Floppies%, Parallels%, Serials%
   Form2.Paralelos.Caption = Str$(Parallels%)
   Form2.Seriales.Caption = Str$(Serials%)
   Form2.Floppies.Caption = Str$(Floppies%)
' COPROCESADOR MATEMATICO
   There% = MathChip%()
   If There% = -1 Then Coproc$ = "SI"
   If There% = 0 Then Coproc$ = "NO"
   Form2.Coprocesador.Caption = Coproc$
' MEMORIA EXTENDIDA
   Form2.XMS.Caption = Str$(Fre&(-2))
   Form2.XMSTOTAL.Caption = Str$(Fre&(-1))
Form2.Show
End Sub

Static Sub ComLineInput (Text$, TimeOutInSeconds%)
   SetComErr 0
   Text$ = ""
   CR$ = Chr$(13)
   LF$ = Chr$(10)
         start& = PDQTimer&()
   Ticks& = TimeOutInSeconds * 18
   Do
    X = DoEvents()
    If ComLoc%() Then
     In$ = ComInput$(1)
     If ComErr%() Then Exit Sub
     If In$ = CR$ Then
      StartLF& = PDQTimer&()
      Do
       X = DoEvents()
       If ComLoc%() Then
        In$ = ComInput$(1)
        If ComErr%() Then Exit Sub
        If In$ <> LF$ Then
         ComPutBack (In$)
         Exit Do
        End If
       End If
       T& = PDQTimer&()
       If T& - StartLF& < 0 Then
        StartLF& = StartLF& - 1573039
       End If
      Loop Until T& - StartLF& > 18
      Exit Sub
     Else
```

```
        Text$ = Text$ + In$
        If Len(Text$) = 256 Then
          SetComErr 62
          Exit Sub
        End If
      End If
    End If
    T& = PDQTimer&()
    If T& - Start& < 0 Then
      Start& = Start& - 1573039
    End If
  Loop Until T& - Start& > Ticks&
  SetComErr 24
End Sub

Static Sub Pause (Ticks%)
  Start& = PDQTimer&()
  Do
    T& = PDQTimer&()
    If T& - Start& < 0 Then
      Start& = Start& - 1573039
    End If
  Loop Until T& - Start& > Ticks%
End Sub
```

4.4.4.2.- Subrutinas de ARCHIVOS.FRM

4.4.4.2.1.- Declaraciones

```
Const OFN_HIDEREADONLY = &H4&
Const OFN_FILEMUSTEXIST = &H1000&
Const OFN_OVERWRITEPROMPT = &O2&

Dim NumComas(5) As Integer
Dim j As Integer
```

4.4.4.2.2.- Subrutinas

```
Sub SwapX ()
        Swap PosX(j), PosX(j - 1), -4
        Swap PosY(j), PosY(j - 1), -4
        Swap Ancho(j), Ancho(j - 1), -4
        j = j - 1
End Sub

Sub SwapXinter ()
' SUBRUTINA PARA REALIZAR EL INTERCALADO
  For i = 2 To NumPuntos - 1
    j = i
    Do While j >= 2
      If PosX(j) <> PosX(j - 1) Then
        If l = 0 Then
          l = 1
          Exit Do
        Else
          l = 0
          Exit Do
        End If
```

```
        ElseIf PosX(j) = PosX(j - 1) And l = 0 Then
          Call SwapYasc
          Exit Do
        ElseIf PosX(j) = PosX(j - 1) And l = 1 Then
          Call SwapYdesc
          Exit Do
        Else Exit Do
        End If
      Loop
  Next i
End Sub

Sub SwapYasc ()
' SUBRUTINA PARA INTERCAMBIO DE X IGUALES, Y DIFERENTES, Y      ASCENDENTES
        k = j
        Do While k >= 2
          If PosY(k) < PosY(k - 1) And PosX(k) = PosX(k - 1) Then
            Swap PosY(k), PosY(k - 1), -4
            Swap PosX(k), PosX(k - 1), -4
            Swap Ancho(k), Ancho(k - 1), -4
            k = k - 1
          Else Exit Do
          End If
        Loop
End Sub

Sub SwapYdesc ()
' SUBRUTINA CON X IGUALES, Y DIFERENTES Y Y DESCENDENTES
        k = j
        Do While k >= 2
          If PosY(k) > PosY(k - 1) And PosX(k) = PosX(k - 1) Then
            Swap PosY(k), PosY(k - 1), -4
            Swap PosX(k), PosX(k - 1), -4
            Swap Ancho(k), Ancho(k - 1), -4
            k = k - 1
          Else Exit Do
          End If
        Loop
End Sub

Sub AscendenteY_Click (Value As Integer)
If NoordenX.Value = True Then
  AscendenteX.Value = True
End If
End Sub

Sub Conversion_Click ()
' BORRADO DE COMPL(I)
For i = 1 To 10
  Compl(i) = ""
Next i
 On Error Resume Next
CMDialog1.DefaultExt = "DRI"
CMDialog1.DialogTitle = "Selección de Archivo"
CMDialog1.Filter = "Drill File (*.DRI)|*.DRI|All Files (*.*)|*.*|"
' Fija el Filtro Default a todos
CMDialog1.FilterIndex = 1
CMDialog1.Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY
```

```
CMDialog1.InitDir = "c:\orcad\pcb\work"
CMDialog1.CancelError = -1
CMDialog1.Action = 1
If Err <> 0 Then
   Close
   Exit Sub
End If
' SUBRUTINA PARA REALIZAR LA CONVERSION de ORCAD
If Orcad.Value = -1 Then
   Nombre$ = CMDialog1.Filename
   Tipo$ = "ORCAD"
   Open Nombre$ For Input As #1
      i = 1
      Do Until EOF(1)
         Line Input #1, Compl(i)
         If Mid$(Compl(i), 1, 5) <> "ASCII" Then
            Tipo$ = ""
            Close
            Beep
            MsgBox "Este no es un archivo generado por ORCAD", 0, "Archivo Incorrecto"
            Exit Sub
         End If
         Compl(i) = RTrim$(Compl(i))
         Compl(i) = Mid$(Compl(i), 7, Len(Compl(i)) - 7)
         k = 1
         For j = 1 To Len(Compl(i))
            Parcial$ = Mid$(Compl(i), j, 1)
            If Parcial$ = "," Then
               NumComas(k) = j
               k = k + 1
            End If
         Next j
         Ancho(i) = Val(Mid$(Compl(i), 1, NumComas(1) - 1))
         PosX(i) = Val(Mid$(Compl(i), NumComas(1) + 1, NumComas(2) - NumComas(1) - 1)) * 2.54
         PosY(i) = Val(Mid$(Compl(i), NumComas(2) + 1, Len(Compl(i)) - NumComas(2))) * 2.54
         i = i + 1
         NumPuntos = i
      Loop
   Close #1
End If
' SUBRUTINA PARA REALIZAR LA CONVERSION DE TANGO
If Tango.Value = -1 Then
   Nombre$ = CMDialog1.Filename
   Open Nombre$ For Input As #1
      Tipo$ = "TANGO"
      Line Input #1, Rev$
      Line Input #1, Rev$
      If Mid$(Rev$, 1, 6) <> "METRIC" Then
         Tipo$ = ""
         Close
         Beep
          MsgBox "Este archivo no es generado por Tango, o sus unidades no son métricas", 0, "Archivo
Incorrecto"
         Exit Sub
      End If
      i = 1
      Do Until EOF(1)
         Line Input #1, Compl(i)
```

```
        If Mid$(Compl(i), 1, 1) = "X" Then
          If Len(Compl(i)) < 8 Then
            PosX(i) = Val(Mid$(Compl(i), 2, 6)) / 10000
            PosY(i) = PosY(i - 1)
            Ancho(i) = 0
            i = i + 1
          Else PosX(i) = Val(Mid$(Compl(i), 2, 6)) / 10000
             PosY(i) = Val(Mid$(Compl(i), 9, 15)) / 10000
             Ancho(i) = 0
             i = i + 1
          End If
        End If
        If Mid$(Compl(i), 1, 1) = "Y" Then
          PosX(i) = PosX(i - 1)
          PosY(i) = Val(Mid$(Compl(i), 2, 6)) / 10000
          Ancho(i) = 0
          i = i + 1
        End If
      Loop
    NumPuntos = i
    Close #1
End If
' SUBRUTINA PARA CONVERSION XY A YX
    If PosicionYX.Value = -1 Then
      For i = 1 To NumPuntos - 1
        Swap PosX(i), PosY(i), -4
      Next i
    End If
'SUBRUTINA DE ORDENAMIENTO
l = 0
If AscendenteX.Value = -1 And (AscendenteY.Value = -1 Or DescendenteY.Value = -1) Then

    For i = 2 To NumPuntos - 1
    j = i
    Do While j >= 2
      If PosX(j) < PosX(j - 1) Then
         Call SwapX
       ElseIf PosX(j) = PosX(j - 1) And AscendenteY.Value = -1 Then
         Call SwapYasc
         Exit Do
       ElseIf PosX(j) = PosX(j - 1) And DescendenteY.Value = -1 Then
         Call SwapYdesc
         Exit Do
       Else Exit Do
       End If
    Loop
    Next i
End If
If AscendenteX.Value = -1 And IntercaladoY.Value = -1 Then
    l = 0
    For i = 2 To NumPuntos - 1
      j = i
      Do While j >= 2
        If PosX(j) < PosX(j - 1) Then
         Call SwapX
        ElseIf PosX(j) = PosX(j - 1) Then
           Call SwapYasc
           Exit Do
```

```
        Else Exit Do
        End If
      Loop
    Next i
    Call SwapXinter
End If
If DescendenteX.Value = -1 And (AscendenteY.Value = -1 Or DescendenteY.Value = -1) Then
    For i = 2 To NumPuntos - 1
    j = i
    Do While j >= 2
      If PosX(j) > PosX(j - 1) Then
         Call SwapX
       ElseIf PosX(j) = PosX(j - 1) And AscendenteY.Value = -1 Then
            Call SwapYasc
            Exit Do
       ElseIf PosX(j) = PosX(j - 1) And DescendenteY.Value = -1 Then
            Call SwapYdesc
            Exit Do
       Else Exit Do
       End If
    Loop
    Next i
End If
If DescendenteX.Value = -1 And IntercaladoY.Value = -1 Then
    I = 0
    For i = 2 To NumPuntos - 1
      j = i
      Do While j >= 2
        If PosX(j) > PosX(j - 1) Then
         Call SwapX
        ElseIf PosX(j) = PosX(j - 1) Then
             Call SwapYasc
             Exit Do
        Else Exit Do
        End If
      Loop
    Next i
    Call SwapXinter
End If
' SUBRUTINA PARA DETERMINAR EL NOMBRE DEL ARCHIVO A GRABARSE
    On Error Resume Next
    CMDialog1.DefaultExt = "TAL"
    CMDialog1.DialogTitle = "Nombre del archivo a grabar"
    CMDialog1.Filter = "Tal File (*.TAL)|*.TAL|All Files (*.*)|*.*|"
' Fija el Filtro Default a todos
    CMDialog1.FilterIndex = 1
    CMDialog1.Flags = OFN_OVERWRITEPROMPT Or OFN_HIDEREADONLY
    CMDialog1.CancelError = -1.
    CMDialog1.Action = 2
    NombreSave$ = CMDialog1.Filename
    If Err = 0 Then
      Open NombreSave$ For Output As #1
      Print #1, Tipo$
      Print #1, NumPuntos
      For i = 1 To NumPuntos - 1
        Print #1, Ancho(i), PosX(i), PosY(i)
      Next i
    Close
```

```
   MsgBox "Se ha convertido y grabado los archivos", 0, "Grabación Confirmada"
   Else Close
      Exit Sub
   End If
End Sub

Sub DescendenteY_Click (Value As Integer)
If NoordenX.Value = True Then
   AscendenteX.Value = True
End If
End Sub

Sub FIN_Click ()
Archivos.Hide
Form2.Show
End Sub

Sub IntercaladoY_Click (Value As Integer)
If NoordenX.Value = True Then
   NoordenX.Value = False
   AscendenteX.Value = True
End If
End Sub

Sub Listar_Click ()
' COLOR DE LA FORMA
Listado.BackColor = RGB(255, 255, 255)
' ALINEAMIENTO DE COLUMNAS EN LA GRILLA
For i = 1 To 3
   Listado.Grid1.FixedAlignment(i) = 1
Next i
' COLOR DE LA GRILLA
Listado.Grid1.ForeColor = RGB(115, 122, 255)
Listado.Grid1.BackColor = RGB(255, 255, 255)
' NO SE UTILIZA HIGHLIGHT EN LAS CELDAS
Listado.Grid1.HighLight = 0
' FIJADO DEL ANCHO DE LAS COLUMNAS
For i = 1 To 3
   Listado.Grid1.ColWidth(i) = 1000
Next i
' LINEAS ENTRE CELDAS
Listado.Grid1.GridLines = -1
' LLENADO DE INFORMACION EN COLUMNAS Y FILAS FIJAS
Listado.Grid1.Row = 0
Listado.Grid1.Col = 0
   Listado.Grid1.Text = "Puntos"
Listado.Grid1.Col = 1
   Listado.Grid1.Text = "Diam Drill"
Listado.Grid1.Col = 2
   Listado.Grid1.Text = "POS X"
Listado.Grid1.Col = 3
   Listado.Grid1.Text = "POS Y"
' LLENADO DE LA GRILLA
On Error Resume Next
' CMDialog1.DefaultExt = "TAL"
CMDialog1.DialogTitle = "Selección de Archivo a Listar"
CMDialog1.Filter = "Drill File (*.TAL)|*.TAL|All Files (*.*)|*.*|"
' Fija el Filtro Default a todos
```

```
CMDialog1.FilterIndex = 1
CMDialog1.Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY
CMDialog1.CancelError = -1
CMDialog1.Action = 1
NombreListar$ = CMDialog1.Filename
If Err = 0 Then
   Open NombreListar$ For Input As #1
      Input #1, Tipo$
      Listado.Tipo.Caption = Tipo$
      Input #1, Numeropuntos
      Listado.Grid1.Rows = Numeropuntos
      Listado.Grid1.Col = 0
         For i = 1 To Numeropuntos - 1
            Listado.Grid1.Row = i
            Listado.Grid1.Text = Str$(i)
         Next i
      For i = 1 To Numeropuntos - 1
         Input #1, Ancho(i), PosX(i), PosY(i)
         Listado.Grid1.Row = i
         Listado.Grid1.Col = 1
            Listado.Grid1.Text = Str$(Ancho(i))
         Listado.Grid1.Col = 2
            Listado.Grid1.Text = Str$(PosX(i))
         Listado.Grid1.Col = 3
            Listado.Grid1.Text = Str$(PosY(i))
      Next i
   Close
ElseIf Err = 32755 Then
    Exit Sub
End If
Listado.Show
End Sub

Sub NoordenX_Click (Value As Integer)
If Control = 0 Then
   Control = Control + 1
   NoordenY.Value = True
End If
Control = 0
End Sub

Sub NoordenY_Click (Value As Integer)
If Control = 0 Then
   Control = Control + 1
   NoordenX.Value = True
End If
Control = 0
End Sub
```

4.4.4.3.- Subrutinas de AUTO.FRM

4.4.4.3.1.- Declaraciones

```
Const OFN_HIDEREADONLY = &H4&
Const OFN_FILEMUSTEXIST = &H1000&
Const OFN_OVERWRITEPROMPT = &O2&
Dim ButtonName$(3)
```

4.4.3.2.- Subrutinas

```
Sub ComPrin ()
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (1)
End Sub

Sub Contador (cuenta As Integer)
For p = 1 To cuenta
Next p
End Sub

Sub Taladrar (MinPuntos As Integer)
Sub Taladrar (MinPuntos As Integer)
On Error GoTo Errorhandler
' VISIBLE E INVISIBLE
   Iniciar.Visible = False
   RegresoInicial.Visible = False
   MsgBox "Por Favor, Encienda el aparato y Verfique el Taladro", 48, "Encienda el Taladro"

'APERTURA DEL PORTICO SERIAL
   OpenCom "COM2:2400,N,8,1,NON"
      If ComErr%() Then
         MsgBox "No se puede abrir COM2"
         CSMeter1.UserValue = 0
         interrupcion.Visible = False
         Iniciar.Visible = True
         RegresoInicial.Visible = True
         Closecom
         Exit Sub
      End If

' SALIDA DEL CY545 AL COMPUTADOR
   dato = "W" + Add1 + "O 081H" + Cr
   ComPrin
   dato = "W" + Add2 + "O 081H" + Cr
   ComPrin

' ENVIO DE PARAMETROS A LOS CY545
   'FIRST RATE
      dato = "W" + Add1 + "F 10" + Cr
      FirstRateX.Caption = "10"
      ComPrin
      dato = "W" + Add2 + "F 10" + Cr
      FirstRateY.Caption = "10"
      ComPrin
   'SLOPE
      dato = "W" + Add1 + "S 170" + Cr
      SlopeX.Caption = "170"
      ComPrin
      dato = "W" + Add2 + "S 170" + Cr
      SlopeY.Caption = "170"
      ComPrin
   'MAX RATE
      dato = "W" + Add1 + "R 15" + Cr
```

```
      MaxRateX.Caption = "15"
      ComPrin
      dato = "W" + Add2 + "R 15" + Cr
      MaxRate.Caption = "15"
      ComPrin
' ENVIO DE DATOS
posy(0) = 0
posx(0) = 0
Homedetectx = 1
Homedetecty = 1
TotalPuntos.Caption = Str$(NumPuntos - 1)
'   Dato = "W" + Add1 + "O 085H" + Cr
'   ComPrin
For i = MinPuntos To NumPuntos - 1
   If Interrumpir.Value = True Then
      MsgBox "Se ha interrumpido el trabajo", 16, "Interrupción"
      CSMeter1.UserValue = 0
      interrupcion.Visible = False
      Iniciar.Visible = True
      RegresoInicial.Visible = True
      dato = "W" + Add1 + "O 081H" + Cr
      ComPrin
      dato = "W" + Add2 + "O 081H" + Cr
      ComPrin
      Closecom
      Exit Sub
   End If
   CSMeter1.PercentValue = CInt(100 * i / (NumPuntos - 1))
   Open ArchivoTemp + "TEMP.TMP" For Output As #2
      Print #2, "TEMPORAL"
      Print #2, Nombre
      Print #2, i
   Close #2
   PuntosPerforados.Caption = Str$(i)
   dato = "W" + Add1 + "Y 00" + Cr
   ComPrin
   dato = "W" + Add2 + "Y 00" + Cr
   ComPrin

' DATOS PARA "Y"
'*****************************************
'Envío de Información al Display Y
   dato = "W" + Add2 + "O 081H" + Cr
   ComPrin
   dato = "W" + Add2 + "O 085H" + Cr
   ComPrin
   Pause (1)
   texto$ = "y=" + LTrim$(Str$(posy1(i)))
   texto$ = Mid$(texto$, 1, 8)
   dato = "W" + Add2 + """" + texto$ + """" + Cr
   ComPrin
   Pause (1)
   Pause (1)
'*****************************************
   dato = "W" + Add2 + "E" + Cr
      ComPrin
      PosicionY.RawData = LTrim$(Str$(posy1(i)))
      PasosY.Caption = LTrim$(Str$((posy(i) - posy(i - 1))))
```

```
   Pause (20)
   If Homedetecty = 1 Then
      dato = "W" + Add2 + "F 120" + Cr
      ComPrint dato
      dato = "W" + Add2 + "R 220" + Cr
      dato = "W" + Add2 + "H 04" + Cr
      ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
      Pause (80)
      dato = "W" + Add2 + "F 10" + Cr
      ComPrin
      dato = "W" + Add2 + "S 170" + Cr
      ComPrin
      dato = "W" + Add2 + "R 15" + Cr
      ComPrin
      Homedetecty = 0
   End If
   dato = "W" + Add2 + "P " + LTrim$(Str$(posy(i))) + Cr
      ComPrin
'   Dato = "W" + Add2 + "O 085H" + Cr
'      ComPrin
'   Escritura = "Y=" + RTrim$(Str$(PosY(i) / 100))
'   Escritura = """" + Compact$(Escritura) + """"
'   Dato = "W" + Add2 + Compact$(Escritura) + Cr
'      ComPrin
'   Dato = "W" + Add2 + "O 081H" + Cr
'      ComPrin
   dato = "W" + Add2 + "B 10H" + Cr
      ComPrin
   dato = "W" + Add2 + "W 11H" + Cr
      ComPrin
   dato = "W" + Add2 + "B 00H" + Cr
      ComPrin
   dato = "W" + Add2 + "0" + Cr
      ComPrin
   dato = "W" + Add2 + "Q" + Cr
      ComPrin
   dato = "W" + Add2 + "Y 00" + Cr
     ComPrin
   dato = "W" + Add2 + "X" + Cr
      ComPrin
' DATOS PARA "X"
'*********************************************
'Envío de Información al Display X
   dato = "W" + Add1 + "O 081H" + Cr
   ComPrin
   dato = "W" + Add1 + "O 085H" + Cr
   ComPrin
   texto$ = "x=" + LTrim$(Str$(posx1(i)))
   texto$ = Mid$(texto$, 1, 8)
   dato = "W" + Add1 + """" + texto$ + """" + Cr
   ComPrin
   Pause (1)
   Pause (1)
'*********************************************
   dato = "W" + Add1 + "E" + Cr
```

```
    ComPrin
    Pause (20)
'  If Homedetectx = 1 Then
'     dato = "W" + Add1 + "F 120" + Cr
'     ComPrint dato
'     dato = "W" + Add1 + "R 220" + Cr
'     comprint dato
'     dato = "W" + Add1 + "H 04" + Cr
'     ComPrint dato
'     If ComErr%() Then
'        MsgBox "No se pueden enviar datos"
'     End If
'     Pause (80)
'     dato = "W" + Add1 + "F 10" + Cr
'     ComPrin
'     dato = "W" + Add1 + "S 170" + Cr
'     ComPrin
'     dato = "W" + Add1 + "R 15" + Cr
'     ComPrin
'     Homedetectx = 0
'  End If

    PosicionX.RawData = LTrim$(Str$(posx1(i)))
    PasosX.Caption = LTrim$(Str$((posx(i) - posx(i - 1))))
  dato = "W" + Add1 + "P " + LTrim$(Str$(posx(i))) + Cr
    ComPrin
  dato = "W" + Add1 + "W 10H" + Cr
    ComPrin
  dato = "W" + Add1 + "B 11H" + Cr
    ComPrin
  dato = "W" + Add1 + "B 15H" + Cr       'Multiplexa al eje Z
    ComPrin
  dato = "W" + Add1 + "+" + Cr           ' Fija dirección de bajada
    ComPrin
  dato = "W" + Add1 + "N " + TaladroPasos.Text + Cr   ' Fija el número de pasos
    ComPrin
  dato = "W" + Add1 + "B 14H" + Cr       ' Enciende el taladro
    ComPrin
  dato = "W" + Add1 + "G" + Cr           ' Baja el taladro
    ComPrin
  dato = "W" + Add1 + "-" + Cr           ' Fija dirección de subida
    ComPrin
  dato = "W" + Add1 + "G" + Cr           ' Sube el taladro
    ComPrin
  dato = "W" + Add1 + "B 04H" + Cr       ' Apaga el Taladro
    ComPrin
  dato = "W" + Add1 + "N " + TaladroPasos.Text + Cr
    ComPrin
  dato = "W" + Add1 + "B 05H" + Cr       ' Multiplexa al eje X
    ComPrin
  dato = "W" + Add1 + "B 01H" + Cr
    ComPrin
  dato = "W" + Add1 + "O 081H" + Cr
    ComPrin
  dato = "W" + Add1 + "? P" + Cr
    ComPrin
  dato = "W" + Add1 + "0" + Cr
    ComPrin
```

```
    dato = "W" + Add1 + "Q" + Cr
      ComPrin
    dato = "W" + Add1 + "Y 00" + Cr
      ComPrin
    dato = "W" + Add1 + "X" + Cr
      ComPrin
Pause (50)

' RECEPCION DE DATOS DEL CHIP
   Do
      If ComEof%() = 0 Then
         Call ComLineInput(texto, 30)
         If ComErr%() = 24 Then
            MsgBox "No se recibe información", 16, "Problemas de Comunicación"
            Closecom
            CSMeter1.UserValue = 0
            interrupcion.Visible = False
            Iniciar.Visible = True
            RegresoInicial.Visible = True
            Exit Sub
         End If
         If Len(texto) > 3 Then
            texto = Mid$(texto, 4, 1)
            If texto = "P" Then
               While ComEof%() = 0
                  Tex$ = ComInput$(ComLoc%())
               Wend
               Exit Do
            Else MsgBox "Se ha recibido información errónea", 64, "Problemas de Comunicación"
               Closecom
               CSMeter1.UserValue = 0
               interrupcion.Visible = False
               Iniciar.Visible = True
               RegresoInicial.Visible = True
               Exit Sub
               End If
         End If
      End If
   Loop
   If ComErr%() Then
      MsgBox "No se pueden recibir datos", 64, "Problemas de Comunicación"
   End If
Homedetecx = 0
Homedetecy = 0
Next i
fin:
dato = "W" + Add1 + "O 081H" + Cr
ComPrin
dato = "W" + Add2 + "O 081H" + Cr
ComPrin
Closecom
CSMeter1.PercentValue = 0
interrupcion.Visible = False
Iniciar.Visible = True
RegresoInicial.Visible = True
GoTo 1000

'Manejo de Errores
```

```
Errorhandler:
      MsgBox "No existe comunicación con el equipo"
      Resume 1000

1000 : End Sub

Sub Iniciar_Click ()
' DIRECCIONES INICIALES
   Add1 = "01"
   Add2 = "02"
   Err = 0
' DETERMINAR SI ES REINICIO O NUEVO
   If MinPuntos <> 0 Then
      GoTo reinicio
   End If
' ARCHIVO QUE VA A TALADRARSE
On Error Resume Next
CMDialog1.DefaultExt = "TAL"
CMDialog1.DialogTitle = "Selección de Archivo"
CMDialog1.Filter = "Drill File (*.TAL)|*.TAL|Todos Files (*.*)|*.*|"
CMDialog1.FilterIndex = 1
CMDialog1.Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY
CMDialog1.CancelError = -1
CMDialog1.Action = 1
Informacion.Caption = Str$(Err)
If Err <> 0 Then
  Exit Sub
End If
Nombre = CMDialog1.Filename
MinPuntos = 1
' OBTENER INFORMACION DEL ARCHIVO
   Open Nombre For Input As #1
   Input #1, Tipo$
   If Tipo$ <> "TANGO" And Tipo$ <> "ORCAD" Then
      Beep
      MsgBox "Este archivo no ha sido generado por este programa", 48, "Archivo Erróneo"
      Close
      Exit Sub
   End If
   TipoArchivo.Caption = Tipo$
   Input #1, NumPuntos
   For i = 1 To NumPuntos - 1
      Input #1, Ancho(i), posx(i), posy(i)
      posx1(i) = posx(i)
      posy1(i) = posy(i)
      posx(i) = CInt(posx(i) * 787.4015748)
      posy(i) = CInt(posy(i) * 787.4015748)
   Next i
   Close #1
reinicio:
'  COLOCAR DRIVE DEFAULT EN ARCHIVO TEMPORAL
   NombreArchivo.Caption = LCase$(Nombre)
   interrupcion.Visible = True
   Drive$ = Chr$(GetDrive%())
   Directory$ = GetDir$(Drive$)
   ArchivoTemp = LCase$(Drive$) + ":" + LCase$(Directory$) + "\"
   Auto.ArchivoTemporal.Caption = ArchivoTemp
Call Taladrar(MinPuntos)
```

```
Interrumpir.Value = False
MinPuntos = 0
End Sub

Sub RegresoInicial_Click ()
Auto.Hide
Form2.Show
End Sub

Sub Spin1_SpinDown ()
If Val(Centimetros.Text) > .5 Then
   Centimetros.Text = Str$(Val(Centimetros.Text) - .2)
   TaladroPasos.Text = Str$(CInt(Val(Centimetros.Text) * 787))
End If
End Sub

Sub Spin1_SpinUp ()
If Val(Centimetros.Text) < 10 Then
   Centimetros.Text = Str$(Val(Centimetros.Text) + .2)
   TaladroPasos.Text = Str$(CInt(Val(Centimetros.Text) * 787))
End If

Sub Spin2_SpinDown ()
'SETEOS INICIALES
   Add1 = "01"
   OpenCom "COM2:2400,N,8,1,NON"
      If ComErr%() Then
         MsgBox "No se puede abrir COM2"
         Closecom
         Exit Sub
      End If
' SALIDA DEL CY545 AL COMPUTADOR
   dato = "W" + Add1 + "O 081H" + Cr
   ComPrin
   Pause (1)
' ENVIO DE PARAMETROS AL CY545
   'FIRST RATE
      dato = "W" + Add1 + "F 10" + Cr
      ComPrin
   'SLOPE
      dato = "W" + Add1 + "S 170" + Cr
      ComPrin
   'MAX RATE
      dato = "W" + Add1 + "R 15" + Cr
      ComPrin
'ENVIO DE DATOS AL COMPUTADOR
   dato = "W" + Add1 + "-" + Cr
      ComPrin
      Pause (1)
   dato = "W" + Add1 + "B 15H" + Cr
      ComPrin
      Pause (1)
   dato = "W" + Add1 + "N 157" + Cr
      ComPrin
      Pause (1)
   dato = "W" + Add1 + "G" + Cr
      ComPrin
      Pause (15)
```

```
   dato = "W" + Add1 + "B 05H" + Cr
      ComPrin
      Pause (1)
   Closecom
End sub

Sub Spin2_SpinUp ()
   Add1 = "01"
   OpenCom "COM2:2400,N,8,1,NON"
      If ComErr%() Then
         MsgBox "No se puede abrir COM2"
         Closecom
         Exit Sub
      End If
' SALIDA DEL CY545 AL COMPUTADOR
   dato = "W" + Add1 + "O 081H" + Cr
   ComPrin
   Pause (1)
' ENVIO DE PARAMETROS AL CY545
   'FIRST RATE
      dato = "W" + Add1 + "F 10" + Cr
      ComPrin
   'SLOPE
      dato = "W" + Add1 + "S 170" + Cr
      ComPrin
   'MAX RATE
      dato = "W" + Add1 + "R 15" + Cr
      ComPrin
'ENVIO DE DATOS AL COMPUTADOR
   dato = "W" + Add1 + "+" + Cr
      ComPrin
      Pause (1)
   dato = "W" + Add1 + "B 15H" + Cr
      ComPrin
      Pause (1)
   dato = "W" + Add1 + "N 157" + Cr
      ComPrin
      Pause (1)
   dato = "W" + Add1 + "G" + Cr
      ComPrin
      Pause (15)
   dato = "W" + Add1 + "B 05H" + Cr
      ComPrin
      Pause (1)
   Closecom
End Sub
```

4.4.4.4.- Subrutinas de INICIAL.FRM

```
Sub ComPrin ()
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (1)
End Sub

Sub Automatico_Click ()
```

```
Form2.Hide
Cr = Chr$(13)
' COLOCAR DRIVE DEFAULT EN ARCHIVO TEMPORAL
   Drive$ = Chr$(GetDrive%())
   Directory$ = GetDir$(Drive$)
   ArchivoTemp = LCase$(Drive$) + ":" + LCase$(Directory$) + "\"
   Auto.ArchivoTemporal.Caption = ArchivoTemp
   Auto.TipoArchivo.Caption = ""
   MinPuntos = 0
'APERTURA DEL PORTICO SERIAL
   OpenCom "COM2:2400,N,8,1,NON"
      If ComErr%() Then
         MsgBox "No se puede abrir COM2"
         CloseCom
         Exit Sub
      End If
   Add1 = "01"
   Add2 = "02"
' SALIDA DEL CY545 AL COMPUTADOR
   Dato = "W" + Add1 + "O 081H" + Cr
   ComPrin
   Dato = "W" + Add2 + "O 081H" + Cr
   ComPrin
' ENVIO DE PARAMETROS A LOS CY545
   'FIRST RATE
      Dato = "W" + Add1 + "F 2" + Cr
      ComPrin
      Dato = "W" + Add2 + "F 2" + Cr
      ComPrin
   'SLOPE
      Dato = "W" + Add1 + "S 50" + Cr
      ComPrin
      Dato = "W" + Add2 + "S 50" + Cr
      ComPrin
   'MAX RATE
      Dato = "W" + Add1 + "R 5" + Cr
      ComPrin
      Dato = "W" + Add2 + "R 5" + Cr
      ComPrin
   CloseCom
Auto.Show
End Sub

Sub FIN_Click ()
End
End Sub

Sub Form_Click ()
Form2.Hide
End
End Sub


Sub Manual_Click ()
Form2.Hide
Cr = Chr$(13)
' COLOR DE LA FORMA
ManualFor.BackColor = &H80000005
```

```
Add1 = ManualFor.Direccion1.Text
Add2 = ManualFor.Direccion2.Text
Address = "0" + LTrim$(Str$(ManualFor.Direccion1.Value))
' APERTURA DE PORTICO SERIAL
   OpenCom "COM2:2400,N,8,1,NON"
      If ComErr%() Then
         MsgBox "No se puede abrir COM2"
         Exit Sub
      End If

' DECLARACION DE PARAMETROS
   parametro(0) = "P"
   parametro(1) = "R"
   parametro(2) = "S"
   parametro(3) = "F"
   parametro(4) = "N"
   parametro(5) = "Y"
   parametro(6) = "B"
  ParametroDisplay = "P"
Control = 2
ManualFor.Actualizar.Value = True
ManualFor.Show
End Sub

Sub Reinicio_Click ()
Reiniciar.Show
End Sub

Sub OK_Click ()
Listado.Hide
Archivos.Show
End Sub
```

4.4.4.5.- Subrutinas de MANUAL.FRM

```
Sub actualizacion (NumParametros As Integer)
' SALIDA DEL CY545 AL COMPUTADOR
   Dato = "W" + Address + "O 081H" + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (2)
' ENVIO DE DATOS PARA LEER POSICION
For j = 0 To NumParametros
   Dato = "W" + Address + "? " + parametro(j) + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (1)
   'RECEPCION DE DATOS DEL CHIP
      Do
         If Not (ComEof%()) Then
            Texto = ComInput$(ComLoc%())
            i = Len(Texto)
            Posicion(j).Caption = Mid$(Texto, 6, i - 5)
```

```
        End If
        If ComErr%() <> 57 Then
            Exit Do
        End If
    Loop
  If ComErr%() Then
    MsgBox "No se pueden Recibir datos"
  End If
Next j
End Sub

Sub Borrado ()
' BORRADO DEL DISPLAY
  Dato = "W" + Address + "O 081H" + Cr
  ComPrint Dato
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If
  Contador (1000)
End Sub

Sub Contador (cuenta As Integer)
For i = 1 To cuenta
Next i
End Sub

Sub Actualizar_Click ()
Call actualizacion(6)
End Sub

Sub Bajar_Click (Value As Integer)
MousePointer = 11
If DisplayPos.Value Then
 Call Borrado
  ' ENVIO DE DATOS AL DISPLAY
  dato = "W" + Address + "O 095H" + Cr
  ComPrint dato
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If
  Pause (1)
End If
'DIRECCION DE MOVIMIENTO HACIA ABAJO
  dato = "W" + Address + "-" + Cr
  ComPrint dato
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If
    Pause (1)
'ORDEN PARA QUE SE MUEVA
  dato = "W" + Address + "G" + Cr
  ComPrint dato
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If

  z = Val(posicion(4).Caption) / 30
  Pause (z)
```

```
    Call actualizacion(0)
    MousePointer = 0
End Sub

Sub BorrarDisplay_Click ()
Call Borrado
End Sub

Sub Continuo_Click ()
    Dato = "W" + Address + "C" + Cr
    ComPrint Dato
        If ComErr%() Then
            MsgBox "No se pueden enviar datos"
        End If
    Contador (1000)
End Sub

Sub CY5451_Click (Value As Integer)
    If cy5452.Value = -1 Then
        cy5452.Value = 0
    End If
    Address = "0" + LTrim$(Direccion1.Value)
    Call actualizacion(6)
    TaladroControl.Enabled = True
End Sub

Sub CY5452_Click (Value As Integer)
    If cy5451.Value = -1 Then
        cy5451.Value = 0
    End If
    Address = "0" + LTrim$(Direccion2.Value)
    Call actualizacion(6)
    TaladroControl.Enabled = False
End Sub

Sub Direccion1_KeyPress (KeyAscii As Integer)
        Add1 = Direccion1.Text
        Address = "0" + LTrim$(Direccion1.Value)
textoinformacion.Text = Address + "1XX"
End Sub

Sub Direccion2_KeyPress (KeyAscii As Integer)
        Add2 = Direccion2.Text
        Address = "0" + LTrim$(RTrim$(Direccion2.Text))
textoinformacion.Text = Add2
End Sub

Sub Display_Click ()
Call Borrado
' ENVIO DE DATOS AL DISPLAY
    Dato = "W" + Address + "O 085H" + Cr
    ComPrint Dato
        If ComErr%() Then
            MsgBox "No se pueden enviar datos"
        End If
    Contador (500)
' ENVIO DE DATOS PARA IMPRIMIR PARAMETROS
    Dato = "W" + Address + """" + TextoDisplay.Text + """" + Cr
```

```
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
End Sub

Sub DistanciaTaladro_Click ()
   Dato = "W" + Address + "N" + RTrim$(Str$(Val(Centimetros.Text) * 10)) + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (2)
Call actualizacion(4)
End Sub

Sub FirstRate_Click ()
   Dato = "W" + Address + "F " + Compact$((FirstRateVal.Text)) + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (2)
Call actualizacion(3)
End Sub

Sub Go_Click ()
If DisplayPos.Value Then
  Call Borrado
' ENVIO DE DATOS AL DISPLAY
   Dato = "W" + Address + "O 081H" + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (1)
End If
   Dato = "W" + Address + "G" + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   z = Int(Val(Posicion(4).Caption) / 22)
   Pause (z)
   Call actualizacion(0)
End Sub

Sub Goposicion_Click ()
   Dato = "W" + Address + "P" + RTrim$(DatosPosicion.Text) + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Contador (2000)
   Posicion(0).Caption = DatosPosicion.Text
End Sub
```

```
Sub Inicializar_Click ()
   Dato = "W" + Address + "I" + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Contador (1000)
Call actualizacion(6)
End Sub

Sub ManejoManual_Click ()
   Dato = "W" + Address + ManejoManualTexto.Text + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Contador (1000)
End Sub

Sub Mas_Click ()
   Dato = "W" + Address + "+" + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Contador (1000)
Call actualizacion(0)
End Sub

Sub MaxRate_Click ()
   Dato = "W" + Address + "R " + Compact$((MaxRateVal.Text)) + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (2)
Call actualizacion(1)
End Sub

Sub Menos_Click ()
   Dato = "W" + Address + "-" + Cr
   ComPrint Dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Contador (1000)
Call actualizacion(0)
End Sub

Sub MenuInicial_Click ()
CloseCom
ManualFor.Hide
Form2.Show
End Sub

Sub NumPasos_Click ()
   Dato = "W" + Address + "N " + Compact$((PasosVal.Text)) + Cr
   ComPrint Dato
```

```
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If
  Pause (2)
Call actualizacion(4)
End Sub

Sub Option3D1_Click (Value As Integer)
ParametroDisplay = "P"
End Sub

Sub Option3D2_Click (Value As Integer)
ParametroDisplay = "R"
End Sub

Sub Option3D3_Click (Value As Integer)
ParametroDisplay = "S"
End Sub

Sub Option3D4_Click (Value As Integer)
ParametroDisplay = "F"
End Sub

Sub Option3D5_Click (Value As Integer)
ParametroDisplay = "N"
End Sub

Sub Option3D6_Click (Value As Integer)
ParametroDisplay = "Y"
End Sub

Sub Parametros_Click ()
Call Borrado
' ENVIO DE DATOS AL DISPLAY
  Dato = "W" + Address + "O 085H" + Cr
  ComPrint Dato
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If
  Contador (500)
' ENVIO DE DATOS PARA IMPRIMIR PARAMETROS
  Dato = "W" + Address + "? " + ParametroDisplay + Cr
  ComPrint Dato
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If
End Sub

Sub Slope_Click ()
  Dato = "W" + Address + "S " + Compact$((SlopeVal.Text)) + Cr
  ComPrint Dato
    If ComErr%() Then
      MsgBox "No se pueden enviar datos"
    End If
  Pause (2)
Call actualizacion(2)
End Sub
```

```
Sub Spin1_SpinDown ()
If Val(Centimetros.Text) > .4 Then
   Centimetros.Text = Str$(Val(Centimetros.Text) - .2)
End If
End Sub

Sub Spin1_SpinUp ()
If Val(Centimetros.Text) < 10 Then
   Centimetros.Text = Str$(Val(Centimetros.Text) + .2)
End If
End Sub

Sub Subir_Click (Value As Integer)
MousePointer = 11
If DisplayPos.Value Then
  Call Borrado
   ' ENVIO DE DATOS AL DISPLAY
   dato = "W" + Address + "O 095H" + Cr
   ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (1)
End If
'DIRECCION DE MOVIMIENTO HACIA ARRIBA
   dato = "W" + Address + "+" + Cr
   ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
      Pause (1)
'ORDEN PARA QUE SE MUEVA
   dato = "W" + Address + "G" + Cr
   ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If

   z = Val(posicion(4).Caption) / 30
   Pause (z)
   Call actualizacion(0)
   MousePointer = 0
End Sub

Sub Taladro_Click (Value As Integer)
MousePointer = 11
If Control <> 0 Then
'ESCRIBIR PARAMETROS DEL NUMERO DE PASOS  Y POSICION
   PosicionejeX = posicion(0).Caption
   PosicionPasos = posicion(4).Caption
   dato = "W" + Address + "N" + RTrim$(Str$(Val(Centimetros.Text) * 10)) + Cr
   ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (2)
   Call actualizacion(4)
'VISIBLE CONTROL DEL TALADRO
```

```
    Subir.Visible = True
    Bajar.Visible = True
    DistanciaTaladro.Visible = True
    Centimetros.Visible = True
    Spin1.Visible = True
    Taladro.Enabled = False
    TaladroejeX.Enabled = True
    TaladroMovimiento.Visible = True
    Controles.Enabled = False
    Goespecífico.Enabled = False
    ParámetrosEst.Enabled = False
    HomeEst.Enabled = False
    ManualEst.Enabled = False
    DisplayHPEst.Enabled = False
    Menuinicial.Enabled = False
'DATO ENVIADO PARA ACTIVAR EL DEMULTIPLEXER
    dato = "W" + Address + "B 15H" + Cr
    ComPrint dato
      If ComErr%() Then
        MsgBox "No se pueden enviar datos"
      End If
    Pause (2)
End If
MousePointer = 0
End Sub

Sub TaladroEjeX_Click (Value As Integer)
'CONTROL DE TALADRO ARRIBA
    If Bajar.Value Then
      Respuesta = MsgBox("PELIGRO: el taladro debe estar arriba. Desea subirlo?", 4, "PELIGRO Taladro
abajo")
      If Respuesta = 6 Then
        Control = 0
        Taladro.Value = True
        Exit Sub
      End If
    End If
'ESCRIBIR PARAMETROS DEL NUMERO DE PASOS Y POSICION
    Dato = "W" + Address + "N " + PosicionPasos + Cr
    ComPrint Dato
      If ComErr%() Then
        MsgBox "No se pueden enviar datos"
      End If
    Pause (2)
    Dato = "W" + Address + "A " + Compact$(RTrim$(PosicionejeX)) + Cr
    ComPrint Dato
      If ComErr%() Then
        MsgBox "No se pueden enviar datos"
      End If
    Pause (5)
    Call actualizacion(4)
'CONTROL DEL TALADRO INVISIBLE
    Subir.Visible = False
    Bajar.Visible = False
    DistanciaTaladro.Visible = False
    Centimetros.Visible = False
    Spin1.Visible = False
    Taladro.Enabled = True
```

```
    TaladroejeX.Enabled = False
    TaladroMovimiento.Visible = False
    Controles.Enabled = True
'DATO ENVIADO PARA DESACTIVAR EL DEMULTIPLEXER
    Dato = "W" + Address + "B 05H" + Cr
    ComPrint Dato
      If ComErr%() Then
        MsgBox "No se pueden enviar datos"
      End If
    Pause (2)
Control = 1
End Sub

Sub Command3D1_Click ()
    Add1 = "01"
    Add2 = "02"

If Val(posicion(1).Caption) < 1600000 Then
' SALIDA DEL CY545 AL COMPUTADOR
    dato = "W" + Add1 + "O 081H" + Cr
    ComPrint dato
    dato = "W" + Add2 + "O 081H" + Cr
    ComPrint dato

' ENVIO DE PARAMETROS A LOS CY545
    'FIRST RATE
      dato = "W" + Add1 + "F 10" + Cr
      ComPrint dato
      dato = "W" + Add2 + "F 10" + Cr
      ComPrint dato
    'SLOPE
      dato = "W" + Add1 + "S 170" + Cr
      ComPrint dato
      dato = "W" + Add2 + "S 170" + Cr
      ComPrint dato
    'MAX RATE
      dato = "W" + Add1 + "R 15" + Cr
      ComPrint dato
      dato = "W" + Add2 + "R 15" + Cr
      ComPrint dato
' ENVIO DE DATOS
    dato = "W" + Add1 + "Y 00" + Cr
    ComPrint dato
    dato = "W" + Add2 + "Y 00" + Cr
    ComPrint dato
   Centimety = Str$(CInt(Val(CentímetrosY.Rawdata) * 787.4015748))
   Centimetx = Str$(CInt(Val(Centimetrosx.Rawdata) * 787.4015748))

' DATOS PARA "Y"
'*****************************************

'Envío de Información al Display Y
    dato = "W" + Add2 + "O 081H" + Cr
    ComPrint dato
    dato = "W" + Add2 + "O 085H" + Cr
    ComPrint dato
    Pause (1)
    texto$ = "y=" + LTrim$(CentímetrosY.Rawdata)
    texto$ = Mid$(texto$, 1, 8)
```

```
    dato = "W" + Add2 + """" + texto$ + """" + Cr
    ComPrint dato
    Pause (1)
    Pause (1)
'*****************************************
    dato = "W" + Add2 + "E" + Cr
    ComPrint dato
    Pause (20)
    If Inicialización.Value = -1 Then
       dato = "W" + Add2 + "F 120" + Cr
       ComPrint dato
       dato = "W" + Add2 + "R 220" + Cr
       ComPrint dato
       dato = "W" + Add2 + "H 04" + Cr
       ComPrint dato
       If ComErr%() Then
          MsgBox "No se pueden enviar datos"
       End If
       Pause (80)
       dato = "W" + Add2 + "F 10" + Cr
       ComPrint dato
       dato = "W" + Add2 + "S 170" + Cr
       ComPrint dato
       dato = "W" + Add2 + "R 15" + Cr
       ComPrint dato
    End If
       dato = "W" + Add2 + "P" + Centimety + Cr
       ComPrint dato
       dato = "W" + Add2 + "B 10H" + Cr
       ComPrint dato
    dato = "W" + Add2 + "W 11H" + Cr
       ComPrint dato
    dato = "W" + Add2 + "B 00H" + Cr
       ComPrint dato
    dato = "W" + Add2 + "0" + Cr
       ComPrint dato
    dato = "W" + Add2 + "Q" + Cr
       ComPrint dato
    dato = "W" + Add2 + "Y 00" + Cr
      ComPrint dato
    dato = "W" + Add2 + "X" + Cr
       ComPrint dato
' DATOS PARA "X"
'*********************************************
'Envío de Información al Display X
    dato = "W" + Add1 + "O 081H" + Cr
    ComPrint dato
    dato = "W" + Add1 + "O 085H" + Cr
    ComPrint dato
    texto$ = "x=" + LTrim$(Centimetrosx.Rawdata)
    texto$ = Mid$(texto$, 1, 8)
    dato = "W" + Add1 + """" + texto$ + """" + Cr
    ComPrint dato
    Pause (1)
    Pause (1)
'*********************************************
    dato = "W" +-Add1 + "E" + Cr
       ComPrint dato
```

```
      Pause (20)
'   If incialización.value=-1 then
'      dato = "W" + Add1 + "F 120" + Cr
'      ComPrint dato
'      dato = "W" + Add1 + "R 220" + Cr
'      comprint dato
'      dato = "W" + Add1 + "H 04" + Cr
'      ComPrint dato
'      If ComErr%() Then
'         MsgBox "No se pueden enviar datos"
'      End If
'      Pause (80)
'      dato = "W" + Add1 + "F 10" + Cr
'      ComPrint dato
'      dato = "W" + Add1 + "S 170" + Cr
'      ComPrint dato
'      dato = "W" + Add1 + "R 15" + Cr
'      ComPrint dato
'   End If

   dato = "W" + Add1 + "P" + Centimetx + Cr
      ComPrint dato
   dato = "W" + Add1 + "W 10H" + Cr
      ComPrint dato
   dato = "W" + Add1 + "B 11H" + Cr
      ComPrint dato
   dato = "W" + Add1 + "B 01H" + Cr
      ComPrint dato
   dato = "W" + Add1 + "O 081H" + Cr
      ComPrint dato
   dato = "W" + Add1 + "0" + Cr
      ComPrint dato
   dato = "W" + Add1 + "Q" + Cr
      ComPrint dato
   dato = "W" + Add1 + "Y 00" + Cr
      ComPrint dato
   dato = "W" + Add1 + "X" + Cr
      ComPrint dato
Pause (50)

GoTo 120
If Inicialización.Value = -1 Then
   MousePointer = 11
   If DisplayPos.Value Then
      Call Borrado
      ' ENVIO DE DATOS AL DISPLAY
      dato = "W" + Address + "O 081H" + Cr
      ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (1)
   End If

' ENVIO DE PARAMETROS de aceleración A los CY-545
   'FIRST RATE
      dato = "W01F 120" + Cr
      ComPrint dato
```

```
      dato = "W02F 120" + Cr
      ComPrint dato
   'MAX RATE
      dato = "W01R 220" + Cr
      ComPrint dato
      dato = "W02R 220" + Cr
      ComPrint dato
   ' Envío a la posición inicial

   dato = "W01H 03" + Cr
   ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (80)
   dato = "W02H 04" + Cr
   ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   Pause (80)

   ' Cambio a aceleraciones anteriores
   'FIRST RATE
      dato = "W01F 10" + Cr
      ComPrint dato
      dato = "W02F 10" + Cr
      ComPrint dato
   'MAX RATE
      dato = "W01R 15" + Cr
      ComPrint dato
      dato = "W02R 15" + Cr
      ComPrint dato
   End If

   Centimety = Str$(CInt(Val(CentímetrosY.Rawdata) * 787.4015748))
   Centimetx = Str$(CInt(Val(Centimetrosx.Rawdata) * 787.4015748))
   dato = "W01P" + Centimetx + Cr
      ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If
   z = Int(Val(posicion(4).Caption) / 13)
   Pause (z)

   dato = "W02P" + Centimety + Cr
      ComPrint dato
      If ComErr%() Then
         MsgBox "No se pueden enviar datos"
      End If

   z = Int(Val(posicion(4).Caption) / 13)
   Pause (z)
   Call actualizacion(0)
End If
120 :
MousePointer = 0
```

```
End Sub
```

4.4.4.6.- Subrutinas de REINICIO.FRM

4.4.4.6.1.- Declaraciones

```
Const OFN_HIDEREADONLY = &H4&
Const OFN_FILEMUSTEXIST = &H1000&
Const OFN_OVERWRITEPROMPT = &O2&
```

4.4.4.6.2.- Subrutinas

```
Sub Informacion (MinPuntos As Integer)
   Numero.Caption = Str$(MinPuntos)
   PosicionX.Caption = Str$(PosX(MinPuntos))
   PosicionY.Caption = Str$(PosY(MinPuntos))
   AnchoPunto.Caption = Str$(Ancho(MinPuntos))
End Sub

Sub Cancelar_Click ()
   EscogerAr.Visible = True
   MenuInicial.Visible = True
   Archivo.Visible = False
   Puntos.Visible = False
   Reinicio.Visible = False
   Cancelar.Visible = False
   Temporal.Visible = False
End Sub

Sub EscogerAr_Click ()
MousePointer = 11
' ESCOGER ARCHIVO
archivar:
On Error Resume Next
CMDialog1.DialogTitle = "Selección de Archivo a Reiniciar"
CMDialog1.Filter = "Temp File (*.TMP)|*.TMP|Todos Files (*.*)|*.*|"
CMDialog1.FilterIndex = 1
CMDialog1.Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY
CMDialog1.CancelError = -1
CMDialog1.Action = 1
Nombre = CMDialog1.Filename
MousePointer = 11
If Err <> 0 Then
   Exit Sub
End If
' VERIFICAR QUE EL ARCHIVO SEA TEMPORAL Y OBTENER ARCHIVO
   Open CMDialog1.Filename For Input As #1
      Input #1, Tipo$
      If Tipo$ <> "TEMPORAL" Then
         MsgBox "Este archivo no es Temporal", 16, "Archivo Incorrecto"
         Close #1
         GoTo archivar
      End If
      Input #1, Nombre
      Input #1, MinPuntos
```

```
    Close #1
' ABRIR ARCHIVO QUE SERA EJECUTADO
    Open Nombre For Input As #1
    Input #1, Tipo$
    If Tipo$ <> "TANGO" And Tipo$ <> "ORCAD" Then
       Beep
       MsgBox "Este archivo no ha sido generado por este programa", 48, "Archivo Erróneo"
       Close
       GoTo archivar
    End If
'   TipoArchivo.Caption = Tipo$
    Input #1, NumPuntos
    For i = 1 To NumPuntos - 1
       Input #1, Ancho(i), posx(i), posy(i)
'      posx1(i) = posx(i)
'      posy1(i) = posy(i)
       posx(i) = CInt(posx(i) * 787.4015748)
       posy(i) = CInt(posy(i) * 787.4015748)
    Next i
    Close #1
' COLOCAR DRIVE DEFAULT EN ARCHIVO TEMPORAL
    Drive$ = Chr$(GetDrive%())
    Directory$ = GetDir$(Drive$)
    ArchivoTemp = LCase$(Drive$) + ":" + LCase$(Directory$) + "\"
    ArchivoTemporal.Caption = ArchivoTemp
    TotalPuntos.Caption = Str$(NumPuntos - 1)
    NombreArchivo.Caption = LCase$(Nombre)
' HABILITAR CONTROLES
    EscogerAr.Visible = False
    MenuInicial.Visible = False
    Archivo.Visible = True
    Puntos.Visible = True
    Reinicio.Visible = True
    Cancelar.Visible = True
    Temporal.Visible = True
    Call Informacion(MinPuntos)
MousePointer = 0
End Sub

Sub Mas_Click ()
If MinPuntos > 1 Then
    MinPuntos = MinPuntos - 1
    Call Informacion(MinPuntos)
End If
End Sub

Sub Menos_Click ()
If MinPuntos < (NumPuntos - 1) Then
    MinPuntos = MinPuntos + 1
    Call Informacion(MinPuntos)
End If
End Sub

Sub MenuInicial_Click ()
Reiniciar.Hide
Form2.Show
End Sub
```

```
Sub Reinicio_Click ()
MsgBox "Presione Inicio al entrar al modo automático", 64, "Reinicio"
   EscogerAr.Visible = True
   MenuInicial.Visible = True
   Archivo.Visible = False
   Puntos.Visible = False
   Reinicio.Visible = False
   Cancelar.Visible = False
   Temporal.Visible = False
Reiniciar.Hide
Form2.Hide
Auto.Show
End Sub
```

Capítulo Cinco

Resultados Experimentales

Capítulo Cinco

Resultados Experimentales

5.1.- Comprobación de Control de Escalas.

El objetivo de comprobar el control de escalas es verificar si el equipo construído está perforando los orificios en los sitios exactos, tomando en consideración que siempre existirán errores debidos al movimiento de la broca el momento de peforar el agujero, y al error propio del equipo.

Para comprobar el escalamiento, se han diseñado varios circuitos impresos formados únicamente por agujeros. Cada cirucito impreso permitirá controlar las escalas de conversión tanto en el eje X como en el eje Y. Se han diseñado tres circuitos impresos. Los dos primeros para controlar el escalamiento de cada uno de los ejes por separado, y el tercero para controlar el escalamiento de los dos ejes al mismo tiempo.

Inicialmente, se ha diseñado un arreglo de treinta agujeros en línea recta, separados cada uno 1 cmt. del otro, y ubicados en forma horizontal. El equipo deberá perforar los agujeros en el sitio exacto. Si existiese error en la determinación de las escalas, mientras más se aleje el equipo del origen, deberá notarse que el taladro no perfora el agujero en el sitio adecuado, existiendo un cierto desplazamiento. El circuito impreso que se diseñó, usando ORCAD PCB es el siguiente:

Figura 5.1

Se ha escogido una distacia Offset en el eje Y de 1 centímetro. El archivo que contiene la información sobre la ubicación de los agujeros en el circuito ha sido nombrado ESCALA1.DRI. El archivo es el siguiente:

```
ASCII(0.03,0.79,0.395)
ASCII(0.03,1.19,0.395)
ASCII(0.03,1.575,0.395)
ASCII(0.03,1.97,0.395)
ASCII(0.03,2.365,0.395)
ASCII(0.03,2.755,0.395)
ASCII(0.03,3.15,0.395)
ASCII(0.03,3.545,0.395)
ASCII(0.03,3.94,0.395)
ASCII(0.03,4.33,0.395)
ASCII(0.03,4.725,0.395)
ASCII(0.03,5.12,0.395)
ASCII(0.03,5.515,0.395)
ASCII(0.03,5.905,0.395)
ASCII(0.03,6.3,0.395)
```

Mediante la utilización del programa diseñado en Visual Basic, se convirtió el archivo a un formato reconocido por el programa, el cual ha sido llamado ESCALA1.TAL, y contiene la siguiente información:

```
ORCAD
16
.03     2.0066    1.0033
.03     3.0226    1.0033
.03     4.0005    1.0033
.03     5.0038    1.0033
.03     6.0071    1.0033
.03     6.9977    1.0033
.03     8.001     1.0033
.03     9.0043    1.0033
.03     10.0076   1.0033
.03     10.9982   1.0033
.03     12.0015   1.0033
.03     13.0048   1.0033
.03     14.0081   1.0033
.03     14.9987   1.0033
.03     16.002    1.0033
```

En base a este archivo, se procedió a perforar los agujeros. Como se trataba de comprobar las escalas, se imprimió en papel, utilizando PRINTPCB de Orcad PCB, una copia del circuito impreso ESCALA1, y utilizando el equipo, se procedió a perforar la hoja de papel. De esta forma se minimizó el error que produciría la broca del taladro el momento de la perforación.

Como resultado de esta prueba, se verificó que el equipo había sido calibrado correctamente. Al observar los agujeros perforados, se pudo comprobar que todos se encontraban en los sitios exactos, lo cual indicaba que el factor de escalamiento en el eje X había sido el adecuado.

Comprobado el factor de escala en el eje X, se procedió a comprobar el escalamiento en el eje Y, para lo cual se diseñó un circuito similar al anterior con puntos verticales dispuestos a una distancia offest de 1 centímetro en el eje X.

El circuito impreso, llamado ESCALA2, que se diseñó es el siguiente:

Figura 5.2

Utilizando OrcadPCB, se obtuvo la información de los puntos de este circuito, bajo el archivo ESCALA2.DRI.

ASCII(0.03,0.395,0.395)
ASCII(0.03,0.395,0.79)
ASCII(0.03,0.395,1.185)
ASCII(0.03,0.395,1.575)
ASCII(0.03,0.395,1.97)
ASCII(0.03,0.395,2.365)
ASCII(0.03,0.395,2.755)
ASCII(0.03,0.395,3.15)
ASCII(0.03,0.395,3.545)

```
ASCII(0.03,0.395,3.94)
ASCII(0.03,0.395,4.33)
ASCII(0.03,0.395,4.725)
ASCII(0.03,0.395,5.12)
ASCII(0.03,0.395,5.51)
ASCII(0.03,0.395,5.905)
ASCII(0.03,0.395,6.3)
ASCII(0.03,0.395,6.695)
ASCII(0.03,0.395,7.09)
```

Usando el programa desarrollado, se procedió a convertir este archivo al formato propio diseñado. El nuevo archivo obtenido, llamado ESCALA2.TAL, es el siguiente:

```
ORCAD
19
.03     1.0033     1.0033
.03     1.0033     2.0066
.03     1.0033     3.0099
.03     1.0033     4.0005
.03     1.0033     5.0038
.03     1.0033     6.0071
.03     1.0033     6.9977
.03     1.0033     8.001
.03     1.0033     9.0043
.03     1.0033     10.0076
.03     1.0033     10.9982
.03     1.0033     12.0015
.03     1.0033     13.0048
.03     1.0033     13.9954
.03     1.0033     14.9987
.03     1.0033     16.002
.03     1.0033     17.0053
.03     1.0033     18.0086
```

En base a este archivo, y utilizando el aparato, se procedió a perforar la hoja de papel impresa usando PRINTPCB de ORCADPCB y correspondiente a este archivo.

Al analizar los resultados, se observó que todos los puntos habían sido perforados en los sitios adecuados, lo cual indicaba que el factor de escalamiento en el eje Y había sido el correcto.

Una vez comprobados los factores de escalamiento en el eje X y Y por separado, se procedió a verificar que el factor de escalas determinado funcionaba adecuadamente cuando se movían los dos ejes al mismo tiempo. Para ello, se diseñó un circuito impreso compuesto de agujeros entrelazados en forma de X.

El circuito impreso diseñado, llamado ESCALA3, es el siguiente:

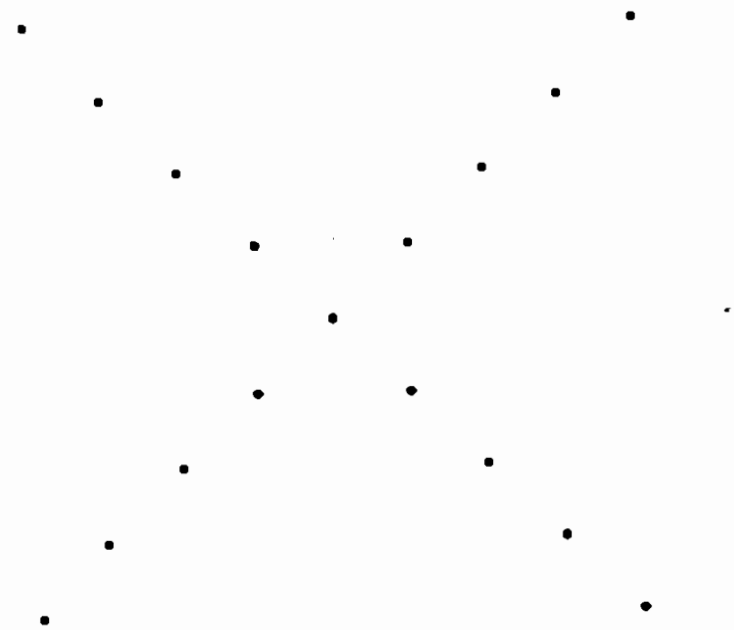


Figura 5.3

El equipo contruído deberá perforar los agujeros en los puntos exactos. El archivo Orcad que contiene la información sobre los puntos se llama ESCALA3.DRI y se lista a continuación:

```
ASCII(0.03,0.395,0.395)
ASCII(0.03,0.79,0.79)
ASCII(0.03,1.185,1.185)
ASCII(0.03,1.575,1.575)
ASCII(0.03,1.97,1.97)
ASCII(0.03,1.575,2.365)
ASCII(0.03,1.185,2.755)
ASCII(0.03,0.79,3.15)
ASCII(0.03,0.435,3.545)
ASCII(0.03,2.365,2.365)
ASCII(0.03,2.755,2.755)
ASCII(0.03,3.15,3.15)
ASCII(0.03,3.545,3.545)
ASCII(0.03,2.365,1.575)
ASCII(0.03,2.755,1.18)
ASCII(0.03,3.15,0.79)
ASCII(0.03,3.545,0.395)
```

El archivo que contiene la información que utilizará el programa diseñado para perforar los puntos se llama ESCALA3.TAL y es el siguiente:

```
ORCAD
18
.03     1.0033     1.0033
.03     2.0066     2.0066
.03     3.0099     3.0099
.03     4.0005     4.0005
.03     5.0038     5.0038
.03     4.0005     6.0071
.03     3.0099     6.9977
.03     2.0066     8.001
.03     1.1049     9.0043
.03     6.0071     6.0071
.03     6.9977     6.9977
.03     8.001      8.001
.03     9.0043     9.0043
.03     6.0071     4.0005
.03     6.9977     2.9972
.03     8.001      2.0066
.03     9.0043     1.0033
```

Una vez perforada la hoja de papel, obtenida a través del programa ORCAD, se pudo comprobar que los factores de escala de los dos ejes trabajaban adecuadamente y permitían obtener las exactitudes necesarias para realizar procesos de perforación considerablemente precisos.

5.2.- Comprobación del grado de precisión.

Para comprobar el grado de precisión se usó un sistema bastante parecido al utilizado en el proceso de comprobación de escalas.

Para comprobar el grado de precisión se utilizó la opción MANUAL del programa diseñado. Para ello, luego de especificar los parámetros de Velocidad Inicial, Velocidad Final y Aceleración, se movió el taladro desde una posición inicial a una posición final, con el parámetro N (Número de Pasos) en 2000. Entonces, se midió la distancia entre los dos puntos, y se dividió para el número de pasos. El resultado de esta división nos dió la distancia que se mueve el sistema por cada paso.

$$\frac{2,5cm}{2000pasos}=0.00125cm/paso=0.000492pulg/paso$$

Haciendo un análisis teórico, para determinar si esta resolución es la que debería tener el equipo, se obtuvo::

Centímetros/Revolución de los ejes: 0.254
Pasos/Revolución de los motores: 200

Entonces:

$$\frac{200pasos/Revol.}{0.254cmt./Revol.}=787,4Pasos/cmt=0.00127cmts./Paso$$

Como se puede observar, el grado de precisión obtenido en forma práctica es prácticamente igual al obtenido en forma analítica. Además una resolución de 0.00127 cmts/pasos permite controlar de manera bastante exacta el lugar de las perforaciones con el taladro.

Además, sabiendo que existe la flexibilidad de que el orificio se perfore con cierto margen de error, hecho que no dificultaría el trabajo posterior con el circuito impreso, esta resolución permite trabajar muy bien con el equipo.

Es necesario analizar también la velocidad a la que trabajará el equipo, en base a los parámetros que se han fijado.

Siendo los parámetros los siguientes:

Velocidad Inicial = 10
Velocidad Final = 15
Aceleración = 170

Cuando el equipo llegue a trabajar a su máxima velocidad (R=15), estará trabajando aproximadamente a 344 pasos/segundo, de acuerdo a las tablas de conversión proporcionadas por el Manual del CY545, para frecuencias de 11 Mhz. Esto quiere decir que:

$$0.00127 Cm/paso * 344 pasos/seg = 0.4368 cm/seg.$$

Lo cual implica que el equipo se moverá 1 centímetro en 2.29 segundos. (Obteniento el inverso de la cantidad anterior). Esta velocidad es adecuada para el tipo de mecanismos que se han usado en este trabajo..

5.3.- Ejemplo de Diseño, Construcción y Perforación de un circuito impreso.

El objetivo de realizar este ejemplo es mostrar en forma práctica el objetivo básico de la realización de este trabajo: diseñar y construir en forma automatizada un circuito impreso. Para ello se usó el circuito diseñado para el panel de leds de las cajas del equipo.

La realización de este trabajo se llevó a cabo en varios pasos, los mismos que se detallan a continuación:

1.- Se diseñó un circuito que permitirá encender 13 leds mediante señales de control provenientes de los circuitos integrados CY545. Además el circuito permitirá resetear los CY545 y CY233 y ejecutar la opción de JOG usando un switch. El circuito incluye una red de resistencias de 330 Ohmios contenida en un chip de 16 pines, 2 circuitos integrados 74HCT244, un zócalo de 34 pines y una resistencia de 330 Ohmios 1/8 Watt.

Mediante el uso del programa ORCAD SDTIII se diseñó el circuito esquemático, el mismo que se muestra a continuación:

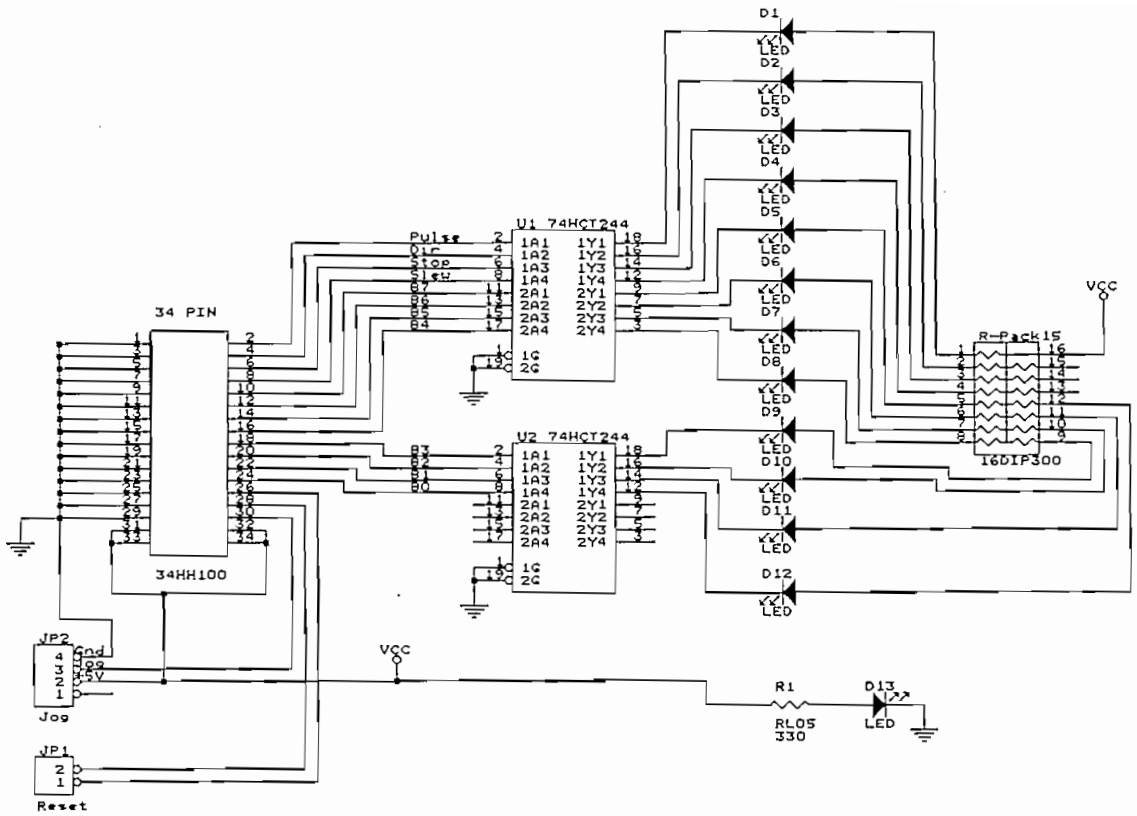


Fig. 5.4

Este circuito esquemático se grabó en el archivo LEDS..

2.- Mediante el uso del programa NETLIST de ORCAD, se generó el listado de todos los elementos del circuito y las conecciones existentes entre ellos. Este listado se almacenó bajo el nombre: LEDS.NET.

El comando que se utilizó para obtener este archivo es el siguiente:
NETLIST LEDS LEDS.NET /S ORCADPCB

El switch /s permite obtener el listado en formato ORCADPCB, lo que permitirá utilizar este listado con el programa PCB. El listado de red generado por este comando es el siguiente:

```
( { OrCAD PCB NetList Format
                    Revised: February 5, 1994
                    Revision:

 Time Stamp - 5-FEB-1994 19:43:49 }
( ECDD4DA6 +5V D13 +5V
 ( ANODE N00035 )
 ( CATHODE GND )
 )
( ECC84B83 10MA D1 10MA
 ( ANODE N00002 )
 ( CATHODE GND )
 )
( ECC84B84 10MA D2 10MA
 ( ANODE N00004 )
 ( CATHODE GND )
 )
( ECC84B85 10MA D3 10MA
 ( ANODE N00006 )
 ( CATHODE GND )
 )
( ECC84B86 10MA D4 10MA
 ( ANODE N00008 )
 ( CATHODE GND )
 )
( ECC84B87 10MA D5 10MA
 ( ANODE N00010 )
 ( CATHODE GND )
 )
( ECC84B88 10MA D6 10MA
 ( ANODE N00012 )
 ( CATHODE GND )
 )
( ECC84B89 10MA D7 10MA
 ( ANODE N00014 )
 ( CATHODE GND )
 )
( ECC84B8B 10MA D8 10MA
 ( ANODE N00016 )
 ( CATHODE GND )
 )
( ECC84B8C 10MA D9 10MA
```

(ANODE N00018)
(CATHODE GND)
)
(ECC84B8D 10MA D10 10MA
(ANODE N00020)
(CATHODE GND)
)
(ECC84B8E 10MA D11 10MA
(ANODE N00022)
(CATHODE GND)
)
(ECC84B8F 10MA D12 10MA
(ANODE N00024)
(CATHODE GND)
)
(ECDD4DA8 330 R1 330
(1 VCC)
(2 N00035)
)
(ECC84B91 74HCT244 U1 74HCT244
(1 GND)
(2 ?1)
(3 N00016)
(4 ?2)
(5 N00014)
(6 ?3)
(7 N00012)
(8 ?4)
(9 N00010)
(10 GND)
(11 ?5)
(12 N00008)
(13 ?6)
(14 N00006)
(15 ?7)
(16 N00004)
(17 ?8)
(18 N00002)
(19 GND)
(20 VCC)
)
(ECC84B92 74HCT244 U2 74HCT244
(1 GND)
(2 ?9)
(3 ?10)
(4 ?11)
(5 ?12)

```
( 6 ?13 )
( 7 ?14 )
( 8 ?15 )
( 9 ?16 )
( 10 GND )
( 11 ?17 )
( 12 N00024 )
( 13 ?18 )
( 14 N00022 )
( 15 ?19 )
( 16 N00020 )
( 17 ?20 )
( 18 N00018 )
( 19 GND )
( 20 VCC )
)
( ECC84B9C JOG JP2 JOG
( 1 ?21 )
( 2 VCC )
( 3 ?22 )
( 4 GND )
)
( ECC84B99 RESET JP1 RESET
( 1 ?23 )
( 2 ?24 )
)
)
```

3.- Una vez creado el archivo de red, y utilizando el programa PCB de ORCAD se generará el circuito impreso. Para ello, se deberá inicializar el trabajo en PCB usando la opción NETLIST. Y, se deberá ingresar el nombre de archivo LEDS.NET. De esta manera, automáticamente se observará en pantalla todos los elementos del circuito impreso. Entonces, se deberá reubicarlos, de acuerdo a las dimensiones de circuito que se requieran; luego, se deberá ejecutar el proceso de autoruteado, el que establecerá los caminos y uniones de cobre necesarios para unir los diferentes elementos.

El archivo PCB generado se llama LEDS.PCB y es el siguiente:

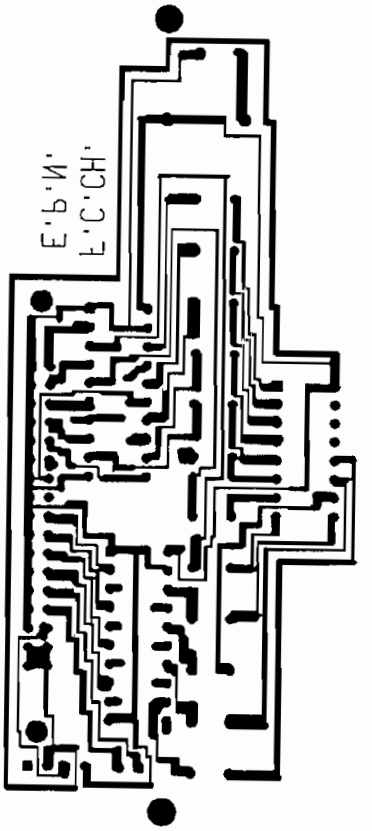


Figura 5.5

4.- En base a este archivo, y usando el comando de ORCAD PCB llamado NCDRILL, se generará el archivo de taladro, de formato ASCII.

El comando es el siguiente:
NCDRILL leds.PCB leds.DRI 0.03 /A

El switch /A permite obtener un archivo en formato ASCII que será leído por el programa diseñado en este trabajo. El parámetro 0.03 indica el tamaño de la herramienta. El listado de este archivo es el siguiente:

ASCII(0.035,1.145,2.18)
ASCII(0.035,1.145,2.28)
ASCII(0.035,1.145,2.38)
ASCII(0.035,1.145,2.48)
ASCII(0.035,1.145,2.58)
ASCII(0.035,1.145,2.68)
ASCII(0.035,1.145,2.78)
ASCII(0.035,1.145,2.88)
ASCII(0.035,1.145,2.98)
ASCII(0.035,1.145,3.08)
ASCII(0.035,1.445,3.08)

ASCII(0.035,1.445,2.98)
ASCII(0.035,1.445,2.88)
ASCII(0.035,1.445,2.78)
ASCII(0.035,1.445,2.68)
ASCII(0.035,1.445,2.58)
ASCII(0.035,1.445,2.48)
ASCII(0.035,1.445,2.38)
ASCII(0.035,1.445,2.28)
ASCII(0.035,1.445,2.18)
ASCII(0.035,1.69,2.14)
ASCII(0.035,1.89,2.14)
ASCII(0.035,1.89,1.865)
ASCII(0.035,1.69,1.865)
ASCII(0.035,1.69,1.59)
ASCII(0.035,1.89,1.59)
ASCII(0.035,1.9,0.8)
ASCII(0.035,1.7,0.8)
ASCII(0.035,1.695,2.415)
ASCII(0.035,1.895,2.415)
ASCII(0.035,1.89,2.69)
ASCII(0.035,1.69,2.69)
ASCII(0.035,1.69,2.97)
ASCII(0.035,1.89,2.97)
ASCII(0.035,2.12,2.985)
ASCII(0.035,2.12,2.885)
ASCII(0.035,2.12,2.785)
ASCII(0.035,2.12,2.685)
ASCII(0.035,2.12,2.585)
ASCII(0.035,2.42,2.585)
ASCII(0.035,2.42,2.685)
ASCII(0.035,2.42,2.785)
ASCII(0.035,2.42,2.885)
ASCII(0.035,2.42,2.985)
ASCII(0.035,2.42,3.085)
ASCII(0.035,2.42,3.185)
ASCII(0.035,2.42,3.285)
ASCII(0.035,2.12,3.285)
ASCII(0.035,2.12,3.185)
ASCII(0.035,2.12,3.085)
ASCII(0.035,1.89,3.285)
ASCII(0.035,1.69,3.285)
ASCII(0.035,1.565,3.48)
ASCII(0.035,1.565,3.58)
ASCII(0.035,1.565,3.68)
ASCII(0.035,1.565,3.78)
ASCII(0.035,1.565,3.88)
ASCII(0.035,1.565,3.98)

ASCII(0.035,1.565,4.08)
ASCII(0.035,1.565,4.18)
ASCII(0.035,1.565,4.28)
ASCII(0.035,1.565,4.38)
ASCII(0.035,1.69,4.385)
ASCII(0.035,1.89,4.385)
ASCII(0.035,1.89,4.66)
ASCII(0.035,1.69,4.66)
ASCII(0.035,1.265,4.38)
ASCII(0.035,1.265,4.28)
ASCII(0.035,1.265,4.18)
ASCII(0.035,1.265,4.08)
ASCII(0.035,1.265,3.98)
ASCII(0.035,1.265,3.88)
ASCII(0.035,1.265,3.78)
ASCII(0.035,1.265,3.68)
ASCII(0.035,1.265,3.58)
ASCII(0.035,1.265,3.48)
ASCII(0.035,1.69,3.56)
ASCII(0.035,1.89,3.56)
ASCII(0.035,1.89,3.835)
ASCII(0.035,1.69,3.835)
ASCII(0.035,1.69,4.11)
ASCII(0.035,1.89,4.11)
ASCII(0.04,0.845,4.63)
ASCII(0.04,0.945,4.63)
ASCII(0.04,1.045,4.63)
ASCII(0.04,1.145,4.63)
ASCII(0.04,1.325,4.635)
ASCII(0.04,1.425,4.635)
ASCII(0.04,1.535,1.16)
ASCII(0.04,1.935,1.16)
ASCII(0.045,0.84,2.495)
ASCII(0.045,0.84,2.595)
ASCII(0.045,0.84,2.695)
ASCII(0.045,0.84,2.795)
ASCII(0.045,0.84,2.895)
ASCII(0.045,0.84,2.995)
ASCII(0.045,0.84,3.095)
ASCII(0.045,0.84,3.195)
ASCII(0.045,0.84,3.295)
ASCII(0.045,0.84,3.395)
ASCII(0.045,0.84,3.495)
ASCII(0.045,0.84,3.595)
ASCII(0.045,0.84,3.695)
ASCII(0.045,0.84,3.795)
ASCII(0.045,0.84,3.895)

ASCII(0.045,0.84,3.995)
ASCII(0.045,0.84,4.095)
ASCII(0.045,0.94,4.095)
ASCII(0.045,0.94,3.995)
ASCII(0.045,0.94,3.895)
ASCII(0.045,0.94,3.795)
ASCII(0.045,0.94,3.695)
ASCII(0.045,0.94,3.595)
ASCII(0.045,0.94,3.495)
ASCII(0.045,0.94,3.395)
ASCII(0.045,0.94,3.295)
ASCII(0.045,0.94,3.195)
ASCII(0.045,0.94,3.095)
ASCII(0.045,0.94,2.995)
ASCII(0.045,0.94,2.895)
ASCII(0.045,0.94,2.795)
ASCII(0.045,0.94,2.695)
ASCII(0.045,0.94,2.595)
ASCII(0.045,0.94,2.495)
ASCII(0.12,0.89,2.145)
ASCII(0.12,0.89,4.445)
ASCII(0.13,1.535,0.625)
ASCII(0.13,1.545,4.87)

5.- Ahora, utilizando Windows, se ejecutará el programa TALADRO que permitirá perforar los agujeros en el circuito impreso. Para ello, deberá escogerse la opción CONVERSION de ARCHIVOS, la cual convertirá el archivo LEDS.DIR de formato ASCII a un archivo LEDS.TAL, también de formato ASCII, pero editado de tal manera que será reconocido por este pograma. Debe anotarse que no se escogió ningún tipo de ordenamiento y la conversión Y-X.

Un listado del archivo LEDS.TAL es el siguiente:

```
ORCAD
129
.035        5.5372      2.9083
.035        5.7912      2.9083
.035        6.0452      2.9083
.035        6.2992      2.9083
.035        6.5532      2.9083
.035        6.8072      2.9083
.035        7.0612      2.9083
.035        7.3152      2.9083
.035        7.5692      2.9083
.035        7.8232      2.9083
.035        7.8232      3.6703
```

.035	7.5692	3.6703
.035	7.3152	3.6703
.035	7.0612	3.6703
.035	6.8072	3.6703
.035	6.5532	3.6703
.035	6.2992	3.6703
.035	6.0452	3.6703
.035	5.7912	3.6703
.035	5.5372	3.6703
.035	5.4356	4.2926
.035	5.4356	4.8006
.035	4.7371	4.8006
.035	4.7371	4.2926
.035	4.0386	4.2926
.035	4.0386	4.8006
.035	2.032	4.826
.035	2.032	4.318
.035	6.1341	4.3053
.035	6.1341	4.8133
.035	6.8326	4.8006
.035	6.8326	4.2926
.035	7.5438	4.2926
.035	7.5438	4.8006
.035	7.5819	5.3848
.035	7.3279	5.3848
.035	7.0739	5.3848
.035	6.8199	5.3848
.035	6.5659	5.3848
.035	6.5659	6.1468
.035	6.8199	6.1468
.035	7.0739	6.1468
.035	7.3279	6.1468
.035	7.5819	6.1468
.035	7.8359	6.1468
.035	8.0899	6.1468
.035	8.3439	6.1468
.035	8.3439	5.3848
.035	8.0899	5.3848
.035	7.8359	5.3848
.035	8.3439	4.8006
.035	8.3439	4.2926
.035	8.8392	3.9751
.035	9.0932	3.9751
.035	9.3472	3.9751
.035	9.6012	3.9751
.035	9.8552	3.9751
.035	10.1092	3.9751

.035	10.3632	3.9751
.035	10.6172	3.9751
.035	10.8712	3.9751
.035	11.1252	3.9751
.035	11.1379	4.2926
.035	11.1379	4.8006
.035	11.8364	4.8006
.035	11.8364	4.2926
.035	11.1252	3.2131
.035	10.8712	3.2131
.035	10.6172	3.2131
.035	10.3632	3.2131
.035	10.1092	3.2131
.035	9.8552	3.2131
.035	9.6012	3.2131
.035	9.3472	3.2131
.035	9.0932	3.2131
.035	8.8392	3.2131
.035	9.0424	4.2926
.035	9.0424	4.8006
.035	9.7409	4.8006
.035	9.7409	4.2926
.035	10.4394	4.2926
.035	10.4394	4.8006
.04	11.7602	2.1463
.04	11.7602	2.4003
.04	11.7602	2.6543
.04	11.7602	2.9083
.04	11.7729	3.3655
.04	11.7729	3.6195
.04	2.9464	3.8989
.04	2.9464	4.9149
.045	6.3373	2.1336
.045	6.5913	2.1336
.045	6.8453	2.1336
.045	7.0993	2.1336
.045	7.3533	2.1336
.045	7.6073	2.1336
.045	7.8613	2.1336
.045	8.1153	2.1336
.045	8.3693	2.1336
.045	8.6233	2.1336
.045	8.8773	2.1336
.045	9.1313	2.1336
.045	9.3853	2.1336
.045	9.6393	2.1336
.045	9.8933	2.1336

.045	10.1473	2.1336
.045	10.4013	2.1336
.045	10.4013	2.3876
.045	10.1473	2.3876
.045	9.8933	2.3876
.045	9.6393	2.3876
.045	9.3853	2.3876
.045	9.1313	2.3876
.045	8.8773	2.3876
.045	8.6233	2.3876
.045	8.3693	2.3876
.045	8.1153	2.3876
.045	7.8613	2.3876
.045	7.6073	2.3876
.045	7.3533	2.3876
.045	7.0993	2.3876
.045	6.8453	2.3876
.045	6.5913	2.3876
.045	6.3373	2.3876
.12	5.4483	2.2606
.12	11.2903	2.2606
.13	1.5875	3.8989
.13	12.3698	3.9243

6.- Una vez convertido el archivo, deberá ejecutarse el proceso de perforación del circuito impreso. Para ello, deberá correrse la opción AUTOMATICO del programa TALADRO. Una vez escogido el archivo, calibrado y sujetado el circuito impreso, se procede a la perforación de cada uno de los agujeros en la baquelita. Como se puede observar en el circuito impreso taladrado, los resultados obtenidos son muy aceptables y se encuentran dentro de los parámetros razonables de un diseño.

De esta manera se ha diseñado un circuito impreso en forma totalmente automatizada, basando el diseño en un computador personal y en el equipo construído durante este trabajo.

Capítulo Seis

Conclusiones y Recomendaciones

Capítulo Seis

Conclusiones y Recomendaciones

6.1.- Análisis de los resultados experimentales

Analizando los resultados experimentales, se puedo observar que el dispositivo construido cumplió con los objetivos propuestos.

El análisis del control de Escalas determinó que existía la correcta relación de factores de escalamiento entre los paquetes de diseño de software y el equipo construido. Esto implica que el equipo puede ser utilizado en forma adecuada para cumplir los objetivos planteados. Este resultado se ha logrado gracias al uso de motores de pasos de alta resolución. Estos motores giran un ángulo muy pequeño con cada paso, y de esta manera la pérdida de uno de estos pasos no sería determinante en el escalamiento y precisión final.

Además, el diseño del eje sin fin permite que con cada paso de motor el sistema se mueva distancias bastante pequeñas en cualquiera de los tres ejes. Esto permite que incluso las distancias pequeñas impliquen un gran número de pasos en la mayoría de movimientos.

Analizando las velocidades, el equipo no permite realizar trabajos en forma muy rápida. Se podría mejorar la velocidad, mejorando el sistema mecánico de movimiento. Como se ha analizado anteriormente, los motores de pasos pueden girar a mayores velocidades, pero al hacerlo la fuerza del motor disminuye atascando el sistema mecánico en diferentes puntos de su recorrido. Resultado de las diferentes pruebas, se determinó que los parámetros de velocidad usados en los experimentos eran los óptimos para este tipo de sistema mecánico.

Se han observado algunos problemas con el taladro en el momento de perforar la baquelita. Dependiendo del largo y el grosor de la broca, ésta fleja y algunas veces perfora el agujero ligeramente a un lado del sitio debido. Este problema no es muy importante, puesto que siendo el desplazamiento mínimo, al insertar los elementos electrónicos en la baquelita perforada, estos pueden ser desviados ligeramente hacia el orificio perforado. Para evitar este problema se pueden usar brocas especiales para este tipo de trabajo.

De acuerdo a los ejemplos realizados, se pudo determinar que el mejor procedimiento para construir un circuito impreso se basa en perforar primero la baquelita, y luego someterla al proceso de eliminación del cobre. De esta manera, se harán coincidir los orificios perforados con el diseño del circuito impreso, y así se logrará mayor exactitud en la elaboración de la placa.

6.2.- Análisis técnico-económico del equipo construido.

El diseño del equipo construido se lo ha realizado con la intención de no solamente usarlo para los fines determinados en esta tesis de grado, sino también con fines de ampliación y futuras diferentes aplicaciones. Es por ello que se han sobredimensionado los aspectos de diseño y la utilización de ciertos componentes. Como ejemplo, se podía haber diseñado el sistema de comunicaciones usando el pórtico paralelo, en lugar de implementar un sistema de comunicaciones seriales en red. Pero, si en el futuro se desea expandir el aparato para incorporar nuevos elementos, tan solo será necesario añadir un elemento más a la red serial de CY233. Por otro parte, no hubiera sido necesario usar memoria para almacenar los programas que contienen las órdenes a ser ejecutadas por los CY545; pero, se podría diseñar una nueva versión de software, con otras funciones, que permitirá que el aparato construido funcione independientemente de la computadora, luego de que la información sea recibida y almacenada en esta memoria.

Todos estos dispositivos adicionales incorporados al equipo determinaron que el costo aumentara considerablemente. Esto, desde el punto de vista del autor de esta tesis, redundó en beneficios puesto que futuras aplicaciones, usando como base el mismo aparato, podrán ser implementadas sin incurrir en gastos fuertes y sin tener que cambiar las partes fundamentales del equipo. La principal modificación que se deberá hacer para cambiar las funciones del equipo serán el taladro y el software. El taladro puesto que se podrán utilizar otro tipo de herramientas para ejecutar otras actividades. El software puesto que dependiendo de la aplicación se deberán ejecutar órdenes y seguir secuencias diferentes.

Para la construcción del equipo se utilizaron tanto elementos traídos de los Estados Unidos como elementos nacionales. Obviamente, todos los elementos electrónicos fueron importados. Los procesadores CY545 y CY233, y el display HP fueron adquiridos a la empresa Intectra en California. El resto de componentes se los compró a Jameco, tales como compuertas, zócalos, cables, conectores, leds, mini-leds, dip-switches, switches, resistencias, condensadores y fuente de poder. Los motores de paso y el taladro DREMEL se adquirieron a la empresa US-Cyberlab.

Luego de realizar el diseño del sistema de red serial usando los CY233 y del sistema de controladores CY545, se decidió comprar a Intectra el prototipo de evaluación CYB-545 fabricado por Cybernetics. Este sistema incluía un circuito impreso para la instalación de los controladores CY545 y CY-233, el mismo que cumplía con todos los requisitos básico del diseño. Esta tarjeta permitió aprovechar los CY545 en su máximo nivel e incluyó la posibilidad de conectar varios CY-233 en red, objetivos de este diseño. Además, estas tarjetas incluían una área libre para diseño de wire wrap, la misma que fue utilizada en un 70%. Por lo tanto, se trajeron dos tarjetas para este equipo. Cabe anotar que se podían haber construido los circuitos impresos en el país, como se lo hizo con el circuito impreso del sistema de potencia para el control de los motores de paso y los circuitos impreso de los leds, pero, la densidad de las tarjetas

y la necesidad de un diseño en dos capas determinaron que sería preferible traerse las tarjetas de los Estados Unidos.

Como ya se ha anotado, las tarjetas de leds, incorporadas a las cajas, fueron diseñadas y construidas localmente. Se han utilizado dos capas de baja densidad en la construcción. La tarjeta que contiene los reguladores y las resistencias para los motores de pasos fue construida también localmente.

En cuanto a la parte mecánica, se debe indicar que luego de haber diseñado un prototipo que no funcionó en forma adecuada, el cual fue construido usando partes de impresoras EPSON, y con la colaboración de una empresa nacional, se determinó que sería preferible usar componentes americanos. De esta manera, basados en contactos con una empresa en los Estados Unidos, se usó un sistema de movimiento para cada uno de los ejes que ya había sido usado en desarrollos de prototipos para este tipo de aplicaciones. Los elementos de este diseño se compraron en la empresa Small Parts en los Estados Unidos. Las cajas, la base y los sistemas de sujeción fueron construídos en el Ecuador.

Para la elaboración del software se usó el programa Visual Basic 3.0 y los utilitarios de Crescent, y su costo no será incluido en este análisis puesto que se considera que su utilización no es exclusiva para el diseño de este tipo de programas.

A continuación se detallará la lista de componentes usados, incluyendo su costo real a la fecha. Se han incluido los costos de importación para los equipos que vinieron de los Estados Unidos.

Tarjetas CY545			
Elemento	Cant.	Precio Unitario	Precio Total
Circuito Impreso CYB550	2	S/. 160.000,00	S/. 320.000,00
Compuerta AND 7408	2	S/. 1.500,00	S/. 3.000,00
Compuerta OR 7432	2	S/. 1.640,00	S/. 3.280,00
Max233 RS-232 driver/receiver	2	S/. 8.500,00	S/. 16.320,00
Buffer Octal de tres estados 74LS244	4	S/. 1.640,00	S/. 3.280,00
Latch Octal 74LS373	4	S/. 1.760,00	S/. 3.520,00
Controlador de Pasos CY545	2	S/. 180.000,00	S/. 360.000,00
Controlador Serial/Paralelo CY233	2	S/. 180.000,00	S/. 360.000,00
Socket de 40 pines	4	S/. 1.440,00	S/. 5.760,00
Socket de 14 pines	4	S/. 400,00	S/. 1.600,00
Socket de 20 pines	6	S/. 960,00	S/. 3.840,00
Sockets de 28 pines	4	S/. 1.120,00	S/. 4.480,00

Tarjetas CY545					
Cristales de 11 Mhz MP110	4	S/.	3.200,00	S/.	12.800,00
Condensador cerámico 0.01uF 50V	12	S/.	250,00	S/.	3.000,00
Condensador cerámico 33pF	8	S/.	500,00	S/.	4.000,00
Condensador electrolítico 4.7uF 63V	6	S/.	362,00	S/.	1.448,00
Condensador electrolítico 22uF 25V	6	S/.	362,00	S/.	1.448,00
Resistencia 1/4Watt 680 ohm	2	S/.	89,00	S/.	178,00
Resistencia 1/4Watt 10k ohm	2	S/.	89,00	S/.	178,00
Paquete de resistencias 10K 10pin SIP	2	S/.	800,00	S/.	1.600,00
Led Rojo con resist. interna HLMP6600	26	S/.	350,00	S/.	9.100,00
Switch de tres posiciones TT11EGPC1	2	S/.	1.800,00	S/.	3.600,00
Reset de botón Alco-FSM	2	S/.	1.545,00	S/.	3.090,00
Switch de deslizamiento 4P3T MMS-43	2	S/.	1.800,00	S/.	3.600,00
DIP-Switch 8 de 2 posiciones 76SB08	4	S/.	2.628,00	S/.	10.512,00
Cabezales de wrip-wrap	2	S/.	1.860,00	S/.	3.720,00
Cabezales para soldar	2	S/.	1.840,00	S/.	3.680,00
Jack doble RJ11 para RS-232	2	S/.	3.200,00	S/.	6.400,00
Socket de poder 180° 5pin DIN	2	S/.	2.490,00	S/.	4.980,00
Conectores de 4 pines con bloqueo	2	S/.	3.990,00	S/.	7.980,00
Espaceadores de Nylon 4-40x¾"	8	S/.	164,00	S/.	1.312,00
Tornillos de Nylon 4-40x⅜"	8	S/.	420,00	S/.	3.360,00
Memoria EEPROM 8Kx8 X2864AP-45	2	S/.	27.440,00	S/.	54.880,00
Display de LED HP 8carac. HDSP2112	2	S/.	55.600,00	S/.	111.200,00
Cable de 4 conduc. con conect. RJ45	2	S/.	2.180,00	S/.	4.360,00
Cable de 6 conduc.	1	S/.	600,00	S/.	600,00
Convertidor RJ-45 to DB-25	1	S/.	2.340,00	S/.	2.340,00
Conectores	9	S/.	5.000,00	S/.	45.000,00
Cables Paralelos	6	S/.	4.000,00	S/.	24.000,00
Cable Wire Wrapping	100	S/.	100,00	S/.	10.000,00
Protector de cables	2 mts	S/.	2.000,00	S/.	4.000,00

Tarjetas CY545					
Varios		S/.	100.000,00	S/.	100.000,00
			Subtotal 1	S/.	1.527.446,00

Tarjeta de Potencia para Motores de pasos					
Elemento	Cant.	Precio Unitario		Precio Total	
Reguladores 78L05 5V	4	S/.	1.320,00	S/.	5.280,00
Resistencia 4 Ohmios 10W	4	S/.	1.800,00	S/.	7.200,00
Espaceadores 4-40-3/8" metal	4	S/.	164,00	S/.	656,00
Tornillo 4-40-1/4"	8	S/.	528,00	S/.	4.224,00
Circuito impreso 4"x6"	1	S/.	18.800,00	S/.	18.800,00
Socket de 40 pines con crimp	1	S/.	3.688,00	S/.	3.688,00
Cable paralelo de 40 conductores /mt.	0.4	S/.	5.600,00	S/.	2.240,00
Varios		S/.	18.000,00	S/.	18.000,00
			Subtotal 2	S/.	60.088,00

Sistema mecánico					
Elemento	Cant.	Precio Unitario		Precio Total	
Motores de Paso CYMotor	4	S/.	135.000,00	S/.	540.000,00
Canales C	3	S/.	58.000,00	S/.	174.000,00
Canale C de 6"	1	S/.	42.000,00	S/.	42.000,00
Bloques Deslizantes	4	S/.	40.000,00	S/.	160.000,00
Imanes	4	S/.	2.500,00	S/.	10.000,00
Tornillo sin fin ACME 1/2"	3	S/.	84.112,00	S/.	252.336,00
Tornillo sin fin ACME 1/2" 6"	1	S/.	34.816,00	S/.	34.816,00
Varillas Guía de 3/8"	6	S/.	18.752,00	S/.	112.512,00
Varillas Guía de 3/8" 6"	2	S/.	7.488,00	S/.	14.976,00
Placas sujetadoras de Varillas guía	16	S/.	4.050,00	S/.	64.800,00
Cojinetes para los tornillos sin fin	4	S/.	2.500,00	S/.	10.000,00
Tapas Finales	8	S/.	12.300,00	S/.	98.400,00
Bandejas sujetadoras de 18"	3	S/.	3.450,00	S/.	10.350,00

Sistema mecánico					
Bandejas sujetadoras de 16"	1	S/.	2.350,00	S/.	2.350,00
Soporte de Sujeción del Eje X	2	S/.	1.430,00	S/.	2.860,00
Arandela #6	100	S/.	282,00	S/.	28.200,00
Tuercas 4-40	68	S/.	304,00	S/.	20.672,00
Tuercas 6-32	33	S/.	304,00	S/.	10.032,00
Tornillos Philips 6-32-1/2"	33	S/.	255,00	S/.	8.415,00
Tornillos #6x1/2"	8	S/.	272,00	S/.	2.176,00
Tornillos 4-40-1/2"	32	S/.	253,00	S/.	8.096,00
Tornillos #5x1/2"	100	S/.	568,00	S/.	56.800,00
Cable de Audio de 5 conductores /mt	6	S/.	1.100,00	S/.	6.600,00
Base (Incluye Pintura)	1	S/.	120.000,00	S/.	120.000,00
Cajas (Incluye Pintura y Letras)	2	S/.	80.000,00	S/.	160.000,00
Tabla 57x75 cm	1	S/.	43.000,00	S/.	43.000,00
Sujetadores de circuitos	2	S/.	30.000,00	S/.	60.000,00
Varios		S/.	150.000,00	S/.	150.000,00
			Subtotal 3	S/.	**2.203.391,00**

Tarjetas de efecto de campo					
Elemento	Cant.		Precio Unitario		Precio Total
Circuito Impreso	4	S/.	10.000,00	S/.	60.000,00
Condensador Electrolítico de 10uF	4	S/.	580,00	S/.	2.320,00
Resistencia de 4.7K 1/8Watt	4	S/.	200,00	S/.	800,00
Transistores de Efecto de Campo Sup.	4	S/.	1.000,00	S/.	4.000,00
Varios		S/.	3.000,00	S/.	3.000,00
			Subtotal 4	S/.	**70.120,00**

Varios					
Elemento	Cant.		Precio Unitario		Precio Total
Relé de Estado Sólido 10 amp.	1	S/.	35.800,00	S/.	35.800,00
Cable de 4 conductores /mt	2	S/.	1.000,00	S/.	2.000,00
Switch con fusible y toma	1	S/.	8.300,00	S/.	8.300,00

Varios					
Toma para taladro IEC	1	S/.	3.320,00	S/.	3.320,00
Leds	2	S/.	300,00	S/.	600,00
Conectores DIN 5 pines Hembra	8	S/.	3.080,00	S/.	24.640,00
Conectores DIN 5 pines Macho	8	S/.	2.280,00	S/.	18.240,00
Taladro Dremel	1	S/.	192.500,00	S/.	192.500,00
Fuente de Poder	1	S/.	95.840,00	S/.	95.840,00
Cable de poder	1	S/.	3.000,00	S/.	3.000,00
Elementos para pruebas		S/.	38.000,00	S/.	38.000,00
Varios		S/.	25.000,00	S/.	25.000,00
			Subtotal 5	S/.	447.240,00

Precio Total = $\sum$ (Subtotal N) ; N = 1,.....,5

Precio Total = $ 4'308.285,oo

En este análisis no se ha incluido el valor del prototipo inicial que se construyó. Aproximadamente su valor fue de $800.000,oo incluyendo los sistemas de impresoras y la elaboración.

6.3.- Conclusiones Generales

Una vez terminado este proyecto y tomando en cuenta los resultados obtenidos se puede concluir los siguiente:

- Este trabajo muestra todos los aspectos correspondientes al diseño y construcción de un taladro prototipo con control numérico para la Fabricación de circuitos impresos, habiéndose cumplido los objetivos planteados al inicio del proyecto.

- El equipo construido ha permitido demostrar que es posible diseñar en nuestro país pequeños prototipos que permiten automatizar la fabricación a baja escala de diferentes tipos de piezas.

- La utilización de motores de pasos da como resultado la obtención de un nivel de alto de precisión. Además el margen de error disminuye puesto que los motores de pasos tienen un error

acumulativo nulo.

- La velocidad del sistema pudo haber sido mayor si los sistemas de desplazamiento construídos en cada uno de los ejes hubieran sido construídos usando rodelas para los desplazamientos.

- Se hubiese podido diseñar el sistema de tal manera que no fuese el taladro el que se moviera, sino el circuito impreso. Esto hubiera permitido aumentar la velocidad de operación del taladro, puesto que se hubieran necesitado motores de menor potencia y sistemas de desplazamientos mas livianos y menos complicados. Pero, puesto que uno de los objetivos del sistema fue permitir posibles utilizaciones posteriores en aplicaciones diferentes al taladro, tales como plotter o cortador, el diseño basado en la movilización de la herramienta de trabajo permite realizar este otro tipo de trabajos.

- En el diseño y construcción del equipo se pudo determinar que el controlador CY545 presenta grandes ventajas en el manejo y control de motores de pasos, así como en la automatización de procesos de control numérico. Este controlador permitió una programación fácil, redujo el hardware notablemente, dispone de un set de instrucciones versátiles específicos para la aplicación, y permite controlar otros dispositivos auxiliares.

- Como se había planteado en los objetivos de este trabajo, el equipo fue diseñado considerando futuras ampliaciones a bajo costo. Además, el sistema puede ser fácilmente modificado para trabajar en otro tipo de funciones, sin necesidad de incluir mas hardware adicional que la herramienta que se utilizará dependiendo de la aplicación. El software deberá ser reprogramado dependiendo del trabajo a realizar.

- El hecho de trabajar con los controladores CY233 permitió crear una red tipo Token ring que permitió controlar el envío y la recepción de información en cada una de las tarjetas. Además, el uso de estos controladores permitirá fácilmente futuras expansiones de hardware, pudiendo llegar a controlarse hasta 255 direcciones diferentes.

- En caso de requerir la ampliación del área de trabajo, tan solo deberán modificarse los sistemas de desplazamiento en los tres ejes. El sistema de control basado en los circuitos CY233 y CY545 permitirá controlar cualquier nuevo motor de pasos sin cambios en estos circuitos. Obviamente, si se dispone de nuevos motores de pasos, con características radicalmente diferentes a los utilizados en este trabajo, el sistema de alimentación de los motores de pasos y tal vez del equipo deberá ser modificado.

- Todos los circuitos integrados se encuentran montados sobre zócalos, lo que permite el reemplazo inmediato en caso de daños.

- El software diseñado permite realizar un proceso de perforación automático de circuitos impresos en forma fácil. Se ha programado con herramientas visuales de computación, facilitando de esta manera el manejo del equipo. Siendo la perforación de circuitos impresos una labor que debe ser realizada por Ingenieros o Tecnólogos Electrónicos, o personas con experiencia en la fabricación de circuitos impresos, el manejo de la parte automática del software puede ser entendida fácilmente. En cuanto a la parte manual, puesto que fue desarrollada para el manejo específico de los controladores CY545 y CY233, podrá ser utilizada por personas con alguna experiencia o conocimiento en el manejo de este tipo de circuitos integrados.

6.4.- Recomendaciones.

- Durante el transporte del equipo se debe tener cuidado con los sistemas de desplazamiento de los ejes X, Y y Z puesto que son bastante susceptibles a descalibraciones. Esto produciría fallas en los desplazamientos y atascamientos de los bloques deslizantes. El continuo uso del equipo con uno de los ejes atascados podría producir daños irremediables en alguno de los motores de pasos.
- Se recomienda usar motores de pasos en sistemas que necesiten posicionamientos precisos y fiables. También se recomienda el uso de microcontroladores puesto que facilitan el diseño y reducen el uso de hardware.

- Es necesario que nuestro país disponga de compañías que permitan fabricar circuitos impresos con tecnologías nuevas, evitándose de esta manera el uso de compañías internacionales que elevan el costo de nuevos proyectos electrónicos.

- Futuras modificaciones a este trabajo deberán incluir mejoras en el software, de tal manera que las perforaciones automáticas de los circuitos se realicen con independencia del computador, posterior a la recepción de datos. Esto se puede hacer puesto que los controladores CY 545 disponen de memoria suficiente, usando la EEPROM, para almacenar hasta 64K de instrucciones programa y datos.

- Es necesaria la actualización constante en el campo de la electrónica, puesto que de esta manera se puede responder en forma eficaz e inmediata a las necesidades nacionales. También es necesaria la actualización constante de los nuevos sitios para adquirir materiales y literatura electrónica, lográndose así una base sólida para el diseño y construcción de cualquier sistema.

- Lastimosamente, cuando nuestro país no disponga de los recursos tecnológicos necesarios para soportar una aplicación, deberá ser necesario recurrir, con un conocimiento adecuado, al soporte

de compañías vendedoras de productos y sistemas en otros países.

- Se recomienda el uso de otras herramientas con el equipo. En este caso se ha usado un taladro, pero podría usarse lásers, cuchillas o algún otro instrumento de trabajo que permitirá ejecutar diversas aplicaciones.

- El uso de un cuarto eje que iría acoplado al eje Z sería recomendable. Este eje rotaría +/- 180 grados, y así se podrían ejecutar trabajos de modelación de figuras en 3 dimensiones.

Bibliografía

1. **ANGULO J.M. & AVILES R.**, "Curso de Robótica", Paraninfo, Madrid, 2da. Edición, 1.985.
2. **ECG SEMICONDUCTORS**, "Master Replacement Guide", Philips ECG, Whilliamsport P.A., 2da. Edición, 1989.
3. **MOMPIN POBLET J.**, "Electrónica y Automática Industriales", Marcombo, Barcelona, 1981.
4. **NATIONAL SEMICONDUCTOR**, "Voltage Regulator Handbook", National Semiconductor, 1982.
5. **TEXAS INSTRUMENTS**, "The TTL Data Book", Segundo Volumen, Texas Instruments Incorporated, Dallas T., 1985.
6. **CONTROL SYSTEMS MAGAZINE**, "Microcomputer Speed Control of Stepper Motors", Chen C.S., II (Número 1, 1982), p.17-19.
7. **SUPERIOR ELECTRIC**, "Slo-Syn DC Stepping Motors", The Superior Electric Company, Connecticut, 1980.
8. **CYBERNETIC MICRO SYSTEMS,** "CY545 High Perfomance Stepper System Controller Manual", Cybernetic Micro Systems Inc, California, 1992.
9. **CYBERNETIC MICRO SYSTEMS,** "CY550 Kit for Stepper Motor Prototyping Applications Manual", Cybernetic Micro Systems Inc, California, 1992.
10. **CYBERNETIC MICRO SYSTEMS,** "CY233 Local Intelligent Network Controller Manual ", Cybernetic Micro Systems Inc, California, 1991
11. **CYBERNETIC MICRO SYSTEMS,** "Cybernetics Micro Systems Control ICs", Cybernetic Micro Systems Inc, California, 1990.
12. **ACCEL TECHNOLOGIES**, "Tango Hands-on Evaluation Package", Accel Technologies Inc, California, 1.989.
13. **OrCAD SYSTEMS CORPORATION,** "Schematic Design Tools", Orcad Systems Corporation, Oregon, 1991.
14. **REVISTA ESPAÑOLA DE ELECTRONICA**, "Teoría y Control de los motores paso a paso", Ricardo Campos, Mayo, Junio y Julio-Agosto, 1.987.
15. **MICROSOFT,** "Visual Basic rorgramming System for Windows, Language

Reference", Versión 3.0, Microsoft, California, 1993

16. **MICROSOFT,** "Visual Basic Programming System for Windows, Programmers Guide", Version 3.0, Microsoft, California, 1993

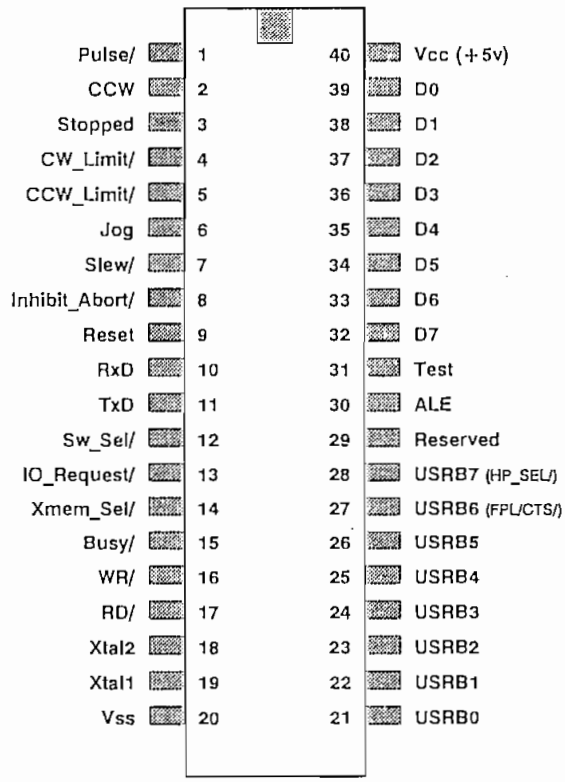
17. **CRESCENT SOFTWARE,** "QuickPak Professional for Widnows", Crescent Software, Volumen 1 y 2, Connecticut, 1993.

18. **CRESCENT SOFTWARE,** "PDQComm for Windows", Crescent Software, Connecticut, 1993.

Anexos

CY545 Summary

CY545 Pins



CY545 Commands

A p	At current step position
B	user Bit set or clear
C	Continuous Step mode
D d	Delay milliseconds
E	Enter External memory prog
F f	Firstrate
G	Go, step relative
H h	Home seek
I	Initialize
J a	Jump to address
K	reserved
L c,a	Loop to addr for count
M	reserved
N n	Number of Steps
O o	Mode
P p	Position for stepping
Q	Quit entering to external mem
R r	Rate max
S s	Slope of accel/decel
T b,a	Til bit matches, loop to addr
U	reserved
V	reserved
W b	Wait for bit match
X	eXecute external commands
Y a	external memory addr pointer
Z c,a	ZillionLoop to addr for count
+	CW direction
-	CCW direction
/	Negate or clear bit values
?cmd	Query command parameter
0	Stop execution of ext memory
[a,c,d	HP-LED command string
"chr"	display string between quotes

CY545 Step Rate Table @ 11 MHz

R 0	18	R 43	1605	R 86	3485	R 128	5238	R 171	6944	R 214	10300
R 1	22	R 44	1649	R 87	3526	R 129	5268	R 172	6997	R 215	10417
R 2	46	R 45	1694	R 88	3567	R 130	5299	R 173	7051	R 216	10536
R 3	69	R 46	1743	R 89	3609	R 131	5329	R 174	7106	R 217	10659
R 4	92	R 47	1787	R 90	3652	R 132	5361	R 175	7161	R 218	10784
R 5	115	R 48	1833	R 91	3696	R 133	5392	R 176	7218	R 219	10913
R 6	137	R 49	1878	R 92	3741	R 134	5424	R 177	7275	R 220	11044
R 7	160	R 50	1926	R 93	3788	R 135	5456	R 178	7333	R 221	11179
R 8	183	R 51	1971	R 94	3835	R 136	5489	R 179	7392	R 222	11317
R 9	206	R 52	2015	R 95	3884	R 137	5522	R 180	7453	R 223	11458
R 10	229	R 53	2065	R 96	3934	R 138	5556	R 181	7514	R 224	11603
R 11	252	R 54	2107	R 97	3986	R 139	5589	R 182	7576	R 225	11752
R 12	275	R 55	2152	R 98	4038	R 140	5624	R 183	7639	R 226	11905
R 13	298	R 56	2198	R 99	4092	R 141	5658	R 184	7703	R 227	12061
R 14	321	R 57	2247	R 100	4148	R 142	5694	R 185	7768	R 228	12222
R 15	344	R 58	2292	R 101	4186	R 143	5729	R 186	7835	R 229	12387
R 16	367	R 59	2338	R 102	4224	R 144	5765	R 187	7902	R 230	12557
R 17	412	R 60	2381	R 103	4264	R 145	5802	R 188	7971	R 231	12731
R 18	459	R 61	2431	R 104	4304	R 146	5839	R 189	8041	R 232	12911
R 19	504	R 62	2477	R 105	4344	R 147	5876	R 190	8112	R 233	13095
R 20	550	R 63	2518	R 106	4386	R 148	5914	R 191	8185	R 234	13285
R 21	596	R 64	2568	R 107	4428	R 149	5952	R 192	8258	R 235	13480
R 22	641	R 65	2612	R 108	4472	R 150	5991	R 193	8333	R 236	13682
R 23	688	R 66	2657	R 109	4516	R 151	6031	R 194	8410	R 237	13889
R 24	733	R 67	2704	R 110	4561	R 152	6071	R 195	8488	R 238	14103
R 25	780	R 68	2753	R 111	4606	R 153	6111	R 196	8567	R 239	14323
R 26	825	R 69	2795	R 112	4653	R 154	6152	R 197	8648	R 240	14550
R 27	871	R 70	2838	R 113	4701	R 155	6194	R 198	8730	R 241	14785
R 28	917	R 71	2892	R 114	4750	R 156	6236	R 199	8814	R 242	15027
R 29	963	R 72	2938	R 115	4799	R 157	6279	R 200	8900	R 243	15278
R 30	1008	R 73	2976	R 116	4850	R 158	6322	R 201	8987	R 244	15537
R 31	1054	R 74	3025	R 117	4902	R 159	6366	R 202	9076	R 245	15805
R 32	1100	R 75	3066	R 118	4955	R 160	6410	R 203	9167	R 246	16082
R 33	1146	R 76	3118	R 119	4982	R 161	6455	R 204	9259	R 247	16369
R 34	1192	R 77	3161	R 120	5009	R 162	6501	R 205	9354	R 248	16667
R 35	1237	R 78	3194	R 121	5037	R 163	6548	R 206	9450	R 249	16975
R 36	1284	R 79	3228	R 122	5064	R 164	6595	R 207	9549	R 250	17296
R 37	1329	R 80	3262	R 123	5093	R 165	6643	R 208	9649	R 251	17628
R 38	1374	R 81	3297	R 124	5121	R 166	6691	R 209	9752	R 252	17974
R 39	1421	R 82	3333	R 125	5150	R 167	6740	R 210	9857	R 253	18333
R 40	1467	R 83	3370	R 126	5179	R 168	6790	R 211	9964	R 254	18707
R 41	1513	R 84	3408	R 127	5208	R 169	6841	R 212	10073	R 255	19097
R 42	1559	R 85	3446			R 170	6892	R 213	10185		

CY545 Step Rate Table @ 12 MHz

R 0	20	R 43	1751	R 86	3802	R 128	5714	R 171	7576	R 214	11236
R 1	25	R 44	1799	R 87	3846	R 129	5747	R 172	7634	R 215	11364
R 2	50	R 45	1848	R 88	3891	R 130	5780	R 173	7692	R 216	11494
R 3	75	R 46	1901	R 89	3937	R 131	5814	R 174	7752	R 217	11628
R 4	100	R 47	1949	R 90	3984	R 132	5848	R 175	7812	R 218	11765
R 5	125	R 48	2000	R 91	4032	R 133	5882	R 176	7874	R 219	11905
R 6	150	R 49	2049	R 92	4082	R 134	5917	R 177	7937	R 220	12048
R 7	175	R 50	2101	R 93	4132	R 135	5952	R 178	8000	R 221	12195
R 8	200	R 51	2151	R 94	4184	R 136	5988	R 179	8065	R 222	12346
R 9	225	R 52	2198	R 95	4237	R 137	6024	R 180	8130	R 223	12500
R 10	250	R 53	2252	R 96	4292	R 138	6061	R 181	8197	R 224	12658
R 11	275	R 54	2299	R 97	4348	R 139	6098	R 182	8264	R 225	12821
R 12	300	R 55	2347	R 98	4405	R 140	6135	R 183	8333	R 226	12987
R 13	325	R 56	2398	R 99	4464	R 141	6173	R 184	8403	R 227	13158
R 14	350	R 57	2451	R 100	4525	R 142	6211	R 185	8475	R 228	13333
R 15	375	R 58	2500	R 101	4566	R 143	6250	R 186	8547	R 229	13514
R 16	400	R 59	2551	R 102	4608	R 144	6289	R 187	8621	R 230	13699
R 17	450	R 60	2597	R 103	4651	R 145	6329	R 188	8696	R 231	13889
R 18	500	R 61	2653	R 104	4695	R 146	6369	R 189	8772	R 232	14085
R 19	550	R 62	2703	R 105	4739	R 147	6410	R 190	8850	R 233	14286
R 20	600	R 63	2747	R 106	4785	R 148	6452	R 191	8929	R 234	14493
R 21	650	R 64	2801	R 107	4831	R 149	6494	R 192	9009	R 235	14706
R 22	700	R 65	2849	R 108	4878	R 150	6536	R 193	9091	R 236	14925
R 23	750	R 66	2899	R 109	4926	R 151	6579	R 194	9174	R 237	15152
R 24	799	R 67	2950	R 110	4975	R 152	6623	R 195	9259	R 238	15385
R 25	851	R 68	3003	R 111	5025	R 153	6667	R 196	9346	R 239	15625
R 26	900	R 69	3049	R 112	5076	R 154	6711	R 197	9434	R 240	15873
R 27	950	R 70	3096	R 113	5128	R 155	6757	R 198	9524	R 241	16129
R 28	1000	R 71	3155	R 114	5181	R 156	6803	R 199	9615	R 242	16393
R 29	1050	R 72	3205	R 115	5236	R 157	6849	R 200	9709	R 243	16667
R 30	1100	R 73	3247	R 116	5291	R 158	6897	R 201	9804	R 244	16949
R 31	1149	R 74	3300	R 117	5348	R 159	6944	R 202	9901	R 245	17241
R 32	1200	R 75	3344	R 118	5405	R 160	6993	R 203	10000	R 246	17544
R 33	1250	R 76	3401	R 119	5435	R 161	7042	R 204	10101	R 247	17857
R 34	1300	R 77	3448	R 120	5464	R 162	7092	R 205	10204	R 248	18182
R 35	1350	R 78	3484	R 121	5495	R 163	7143	R 206	10309	R 249	18519
R 36	1401	R 79	3521	R 122	5525	R 164	7194	R 207	10417	R 250	18868
R 37	1449	R 80	3559	R 123	5556	R 165	7246	R 208	10526	R 251	19231
R 38	1499	R 81	3597	R 124	5587	R 166	7299	R 209	10638	R 252	19608
R 39	1550	R 82	3636	R 125	5618	R 167	7353	R 210	10753	R 253	20000
R 40	1600	R 83	3676	R 126	5650	R 168	7407	R 211	10870	R 254	20408
R 41	1650	R 84	3717	R 127	5682	R 169	7463	R 212	10989	R 255	20833
R 42	1701	R 85	3759			R 170	7519	R 213	11111		

CY545 Step Rate Table @ 14.75 MHz

R 0	24	R 43	2153	R 86	4674	R 128	7024	R 171	9312	R 214	13811
R 1	30	R 44	2211	R 87	4728	R 129	7064	R 172	9383	R 215	13968
R 2	61	R 45	2272	R 88	4783	R 130	7105	R 173	9455	R 216	14128
R 3	92	R 46	2337	R 89	4839	R 131	7146	R 174	9528	R 217	14293
R 4	123	R 47	2396	R 90	4897	R 132	7188	R 175	9603	R 218	14461
R 5	154	R 48	2458	R 91	4956	R 133	7230	R 176	9678	R 219	14633
R 6	184	R 49	2519	R 92	5017	R 134	7273	R 177	9755	R 220	14809
R 7	215	R 50	2582	R 93	5079	R 135	7316	R 178	9833	R 221	14990
R 8	246	R 51	2643	R 94	5143	R 136	7360	R 179	9913	R 222	15175
R 9	277	R 52	2701	R 95	5208	R 137	7405	R 180	9993	R 223	15365
R 10	307	R 53	2768	R 96	5275	R 138	7449	R 181	10075	R 224	15559
R 11	338	R 54	2826	R 97	5344	R 139	7495	R 182	10158	R 225	15759
R 12	369	R 55	2885	R 98	5415	R 140	7541	R 183	10243	R 226	15963
R 13	399	R 56	2948	R 99	5487	R 141	7587	R 184	10329	R 227	16173
R 14	430	R 57	3013	R 100	5562	R 142	7635	R 185	10417	R 228	16389
R 15	461	R 58	3073	R 101	5613	R 143	7682	R 186	10506	R 229	16610
R 16	492	R 59	3136	R 102	5664	R 144	7731	R 187	10596	R 230	16838
R 17	553	R 60	3193	R 103	5717	R 145	7780	R 188	10688	R 231	17072
R 18	615	R 61	3260	R 104	5771	R 146	7829	R 189	10782	R 232	17312
R 19	676	R 62	3322	R 105	5825	R 147	7879	R 190	10878	R 233	17560
R 20	737	R 63	3377	R 106	5881	R 148	7930	R 191	10975	R 234	17814
R 21	799	R 64	3443	R 107	5938	R 149	7982	R 192	11074	R 235	18076
R 22	860	R 65	3502	R 108	5996	R 150	8034	R 193	11174	R 236	18346
R 23	922	R 66	3563	R 109	6055	R 151	8087	R 194	11277	R 237	18624
R 24	983	R 67	3626	R 110	6115	R 152	8140	R 195	11381	R 238	18910
R 25	1046	R 68	3691	R 111	6177	R 153	8194	R 196	11488	R 239	19206
R 26	1106	R 69	3747	R 112	6239	R 154	8249	R 197	11596	R 240	19511
R 27	1167	R 70	3805	R 113	6303	R 155	8305	R 198	11706	R 241	19825
R 28	1229	R 71	3877	R 114	6369	R 156	8362	R 199	11819	R 242	20150
R 29	1291	R 72	3940	R 115	6435	R 157	8419	R 200	11934	R 243	20486
R 30	1352	R 73	3991	R 116	6504	R 158	8477	R 201	12051	R 244	20833
R 31	1413	R 74	4057	R 117	6573	R 159	8536	R 202	12170	R 245	21193
R 32	1476	R 75	4111	R 118	6644	R 160	8596	R 203	12292	R 246	21564
R 33	1536	R 76	4181	R 119	6680	R 161	8656	R 204	12416	R 247	21949
R 34	1598	R 77	4239	R 120	6717	R 162	8717	R 205	12543	R 248	22348
R 35	1659	R 78	4283	R 121	6754	R 163	8780	R 206	12672	R 249	22762
R 36	1722	R 79	4328	R 122	6791	R 164	8843	R 207	12804	R 250	23192
R 37	1781	R 80	4374	R 123	6829	R 165	8907	R 208	12939	R 251	23638
R 38	1843	R 81	4421	R 124	6867	R 166	8972	R 209	13076	R 252	24101
R 39	1906	R 82	4470	R 125	6905	R 167	9038	R 210	13217	R 253	24583
R 40	1967	R 83	4519	R 126	6944	R 168	9105	R 211	13361	R 254	25085
R 41	2028	R 84	4569	R 127	6984	R 169	9173	R 212	13507	R 255	25608
R 42	2090	R 85	4621			R 170	9242	R 213	13657		

CY545 Step Rate Table @ 16 MHz

R 0	26	R 43	2335	R 86	5070	R 128	7619	R 170	10025	R 213	14815
R 1	33	R 44	2398	R 87	5128	R 129	7663	R 171	10101	R 214	14981
R 2	67	R 45	2465	R 88	5188	R 130	7707	R 172	10178	R 215	15152
R 3	100	R 46	2535	R 89	5249	R 131	7752	R 173	10256	R 216	15326
R 4	133	R 47	2599	R 90	5312	R 132	7797	R 174	10336	R 217	15504
R 5	167	R 48	2667	R 91	5376	R 133	7843	R 175	10417	R 218	15686
R 6	200	R 49	2732	R 92	5442	R 134	7890	R 176	10499	R 219	15873
R 7	233	R 50	2801	R 93	5510	R 135	7937	R 177	10582	R 220	16064
R 8	267	R 51	2867	R 94	5579	R 136	7984	R 178	10667	R 221	16260
R 9	300	R 52	2930	R 95	5650	R 137	8032	R 179	10753	R 222	16461
R 10	333	R 53	3003	R 96	5722	R 138	8081	R 180	10840	R 223	16667
R 11	367	R 54	3065	R 97	5797	R 139	8130	R 181	10929	R 224	16878
R 12	400	R 55	3130	R 98	5874	R 140	8180	R 182	11019	R 225	17094
R 13	433	R 56	3197	R 99	5952	R 141	8230	R 183	11111	R 226	17316
R 14	467	R 57	3268	R 100	6033	R 142	8282	R 184	11204	R 227	17544
R 15	500	R 58	3333	R 101	6088	R 143	8333	R 185	11299	R 228	17778
R 16	534	R 59	3401	R 102	6144	R 144	8386	R 186	11396	R 229	18018
R 17	600	R 60	3463	R 103	6202	R 145	8439	R 187	11494	R 230	18265
R 18	667	R 61	3537	R 104	6260	R 146	8493	R 188	11594	R 231	18519
R 19	733	R 62	3604	R 105	6319	R 147	8547	R 189	11696	R 232	18779
R 20	800	R 63	3663	R 106	6380	R 148	8602	R 190	11799	R 233	19048
R 21	866	R 64	3735	R 107	6441	R 149	8658	R 191	11905	R 234	19324
R 22	933	R 65	3799	R 108	6504	R 150	8715	R 192	12012	R 235	19608
R 23	1000	R 66	3865	R 109	6568	R 151	8772	R 193	12121	R 236	19900
R 24	1066	R 67	3933	R 110	6633	R 152	8830	R 194	12232	R 237	20202
R 25	1135	R 68	4004	R 111	6700	R 153	8889	R 195	12346	R 238	20513
R 26	1200	R 69	4065	R 112	6768	R 154	8949	R 196	12461	R 239	20833
R 27	1266	R 70	4128	R 113	6838	R 155	9009	R 197	12579	R 240	21164
R 28	1333	R 71	4206	R 114	6908	R 156	9070	R 198	12698	R 241	21505
R 29	1401	R 72	4274	R 115	6981	R 157	9132	R 199	12821	R 242	21858
R 30	1467	R 73	4329	R 116	7055	R 158	9195	R 200	12945	R 243	22222
R 31	1533	R 74	4400	R 117	7130	R 159	9259	R 201	13072	R 244	22599
R 32	1601	R 75	4459	R 118	7207	R 160	9324	R 202	13201	R 245	22989
R 33	1667	R 76	4535	R 119	7246	R 161	9390	R 203	13333	R 246	23392
R 34	1734	R 77	4598	R 120	7286	R 162	9456	R 204	13468	R 247	23810
R 35	1799	R 78	4646	R 121	7326	R 163	9524	R 205	13605	R 248	24242
R 36	1867	R 79	4695	R 122	7366	R 164	9592	R 206	13746	R 249	24691
R 37	1932	R 80	4745	R 123	7407	R 165	9662	R 207	13889	R 250	25157
R 38	1999	R 81	4796	R 124	7449	R 166	9732	R 208	14035	R 251	25641
R 39	2067	R 82	4848	R 125	7491	R 167	9804	R 209	14184	R 252	26144
R 40	2133	R 83	4902	R 126	7533	R 168	9877	R 210	14337	R 253	26667
R 41	2200	R 84	4957	R 127	7576	R 169	9950	R 211	14493	R 254	27211
R 42	2268	R 85	5013					R 212	14652	R 255	27778

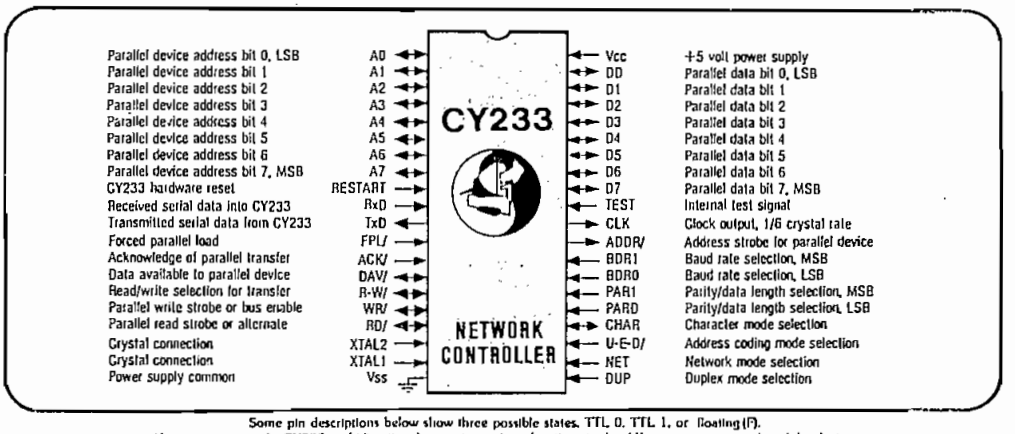
Stepper Motor Controller Selection Guide

CY 500	CY 512	CY 525	CY 545	Function
2K	5K	10K	27K	Max Usable Step/Sec
64K	64K	64K	1.6M	Max Number of Steps
21	25	26	28	Number of Instructions
18	48	60	64K	Program Storage (bytes)
Exp	Exp	Lin	Lin	Accel (Exponent/Linear)
				Motor Support
	•	•	•	Pulse & Direction Output
•	•	•		4-phase Output
•	•	•	o	2-phase Compatible
	o	o	o	3-phase Compatible
•	•	•	o	4 phase Compatible
	o	o	o	5-phase Compatible
•	•	•	•	Full Step
•	•	o	o	Half Step
		o	o	Quad Step
			o	Micro Step
•	o	o	•	Single Step (Jog Mode)
o	o	•	•	Continuous Stepping
•	•	•	•	Constant Rate stepping
•	•	•	•	Ramped Stepping
			•	Low Power Standby
			•	Motor On/Off Output
o	o	o	•	Limit Detection
				Command Interface
			•	Serial Interface
•	•	•	•	Parallel Interface
•	•	•	•	Binary Data Structure
•	•	•	•	ASCII Data Structure
•	•	•		Internal Stored Program
			•	External Stored Program
•	•	•	•	Direct Command Mode
			•	Thumbwheel Input Support
			•	External Display Support
	o		•	Stand Alone Operation

• Yes
o To a Degree or with Additional Work

CY 500	CY 512	CY 525	CY 545	Function
				Program Features
	•	•	•	List Prog Buffer Contents
	•	•	•	Display # of Steps Param
	•	•	•	Display Accel Parameter
	•	•	•	Display Step Rate
	•	•	•	Display Current Position
			•	Display Val of Ext Inputs
•	•	•	•	Stored Prog Execution
•	•	•	•	Conditional Prog Structure
•	•	•	•	Programmable Time Delay
	256	256	64K	Program Repetition Count
	•	•	•	Unconditional Branch
1	1	1	16	Programmable I/O Lines
		•		Program Labels
•	•	•	•	Multi-controller Sync
		•		Live Cmds During Prog Exec
				Motion Features
o	o	•	•	Programmable Start Rate
•	•	•	•	Programmable Slew Rate
o	o	•	•	Program. Accel/decel Slope
•	•	•	•	Software Direction Control
	•	•	•	Automatic Direction Finding
•	•	•	•	Program. Num of Steps
			•	Home Seek Command
•	•	•	•	Absolute Position Stepping
•	•	•	•	Relative Num of Steps
•	•	•	•	Emergency Stop/Abort
	•	•	•	Decelerating Stop/Abort
•	•	•	•	Step Inhibit Input
	•	o		Closed Loop Rate Control
•			•	External Direction Control
•	•	•	•	Motion Complete Indicator
	•	•	•	Slew Indicator
•	•	•	o	Prog Complete Indicator
•			•	External Jog Mode
		•		On-the-Fly Rate Change
		•		On-the-Fly Position Output

Pin Configuration and Description



Some pin descriptions below show three possible states, TTL 0, TTL 1, or floating (F).
Also, many pins on the CY233 are bidirectional, assuming various functions in the different operating modes of the device.

Pin	Mnemonic	Function
1 (I/O)	A0	Parallel device address bit 0, LSB
2 (I/O)	A1	Parallel device address bit 1
3 (I/O)	A2	Parallel device address bit 2
4 (I/O)	A3	Parallel device address bit 3
5 (I/O)	A4	Parallel device address bit 4
6 (I/O)	A5	Parallel device address bit 5
7 (I/O)	A6	Parallel device address bit 6
8 (I/O)	A7	Parallel device address bit 7, MSB

These pins specify the current address to the parallel device. They are used for address generation and testing

9 (I) Restart CY233 hardware reset
The CY233 is hardware reset by this active high signal.

10 (I) RxD Received serial data into CY233
TTL level serial data is input to the CY233 through this pin.

11 (O) TxD Transmitted serial data from CY233
TTL level serial data is output on this pin.

FPL/ Forced parallel load
The local parallel device can force a multi-byte read or parallel command operation using this pin. The read address must also be supplied.

13 (I) ACK/ Acknowledge of parallel transfer
14 (I/O) DAV/ Data available to parallel device
15 (I/O) R-W/ Read/Write selection for transfer
These pins are used for handshake control of parallel data transfers, indicating when data are stable and have been accepted by the CY233 and local parallel device.

16 (I/O) WR/ Parallel write strobe or Bus Enable
17 (I/O) RD/ Parallel read strobe or alternate select
As an alternative to the handshake data transfers, the CY233 also supports strobed transfers. These pins control the timing of the transfers in the strobed mode. Alternate functions are available in the handshake mode.

18 (I) XTAL2 Crystal connection
19 (I) XTAL1 Crystal connection
An external 11 MHz crystal or clock source is connected to these pins. All baud rates and device timing are derived from this clock.

20 (I) VSS Power supply common

21 (I) DUP Duplex mode selection
22 (I) NET Network mode selection
Select the communications mode through these pins.

NET	DUP	ECHO	MODE	USEFUL FOR
0	0	Invalid	Master	Ring Network
0	F	None	Master	Point to Point
0	1	All	Master	Ring Network
F	0	Invalid	Slave	Ring Network
F	F	None	Slave	Half Dup, Line Turnaround
F	1	All	Slave	Console to CY233
1	0	Valid	Master	Console to CY233
1	F	Valid	Slave	Bus
1	1	Valid	Slave	Bus

23 (I) U-E-D/ Address coding mode selection
Select the CY233 addressing format through this pin.

1 Uncoded UART mode, addresses not used
F Encoded Address, positive true binary
0 Decoded Address, negative true, 1 of 8

Pin	Mnemonic	Function
24 (I/O)	CHAR	Character mode selection

This pin selects the data representation for the contents of serial messages. When the CY233 is running, it may also be used to enable tri-state transmit drivers.

F ASCII Character mode
1 ASCII Hexadecimal character mode
0 Binary Character mode

25 (I) PAR0 Parity/Data length selection, LSB
26 (I) PAR1 Parity/Data length selection, MSB
Select parity and character length options with these pins

PAR1, 0	Parity	Data Length	Total Character Length
F F	Mark	7	10
F 1	Even	7	10
F 0	Odd	7	10
1 F	Space	7	10
1 1	None	8	10
1 0	Space	8	11
0 F	Mark	8	11
0 1	Even	8	11
0 0	Odd	8	11

27 (I) BDR0 Baud rate selection, LSB
28 (I) BDR1 Baud rate selection, MSB
These pins select the CY233 baud rate as follows.

BDR1, 0	Rate with 11.059 Mhz clock
F F	Self Adaptive
F 1	57600
F 0	19200
1 F	9600
1 1	4800
1 0	2400
0 F	1200
0 1	600
0 0	300

29 (O) ADDR/ Address strobe for parallel device
A short strobe is generated on this line when the CY233 has written a new address to lines A0-A7, but before the lines are tested. Could be used to trigger external addressing logic.

30 (O)	CLK	Clock output, 1/6 crystal rate
31 (I)	TEST	Internal test signal, connect to Vcc
32 (I/O)	D7	Parallel data bit 7, MSB
33 (I/O)	D6	Parallel data bit 6
34 (I/O)	D5	Parallel data bit 5
35 (I/O)	D4	Parallel data bit 4
36 (I/O)	D3	Parallel data bit 3
37 (I/O)	D2	Parallel data bit 2
38 (I/O)	D1	Parallel data bit 1
39 (I/O)	D0	Parallel data bit 0, LSB

These bi-directional lines are used for all parallel data transfers between the CY233 and local parallel devices.

40 (I) VCC +5 Volt power supply

CY233 Electrical Specifications

ABSOLUTE MAXIMUM RATINGS:

Ambient temperature under bias 0°C to 70°C
Storage temperature -65°C to +150°C
Voltage on any pin with respect to GND -0.5V to Vcc+0.5V
Power Dissipation 0.5 watts

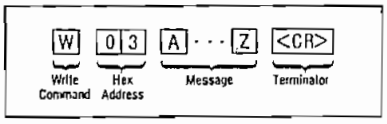
DC & OPERATING CHARACTERISTICS:
(TA=0°C to 70°C, Vcc=+5V±10%)

SYM	PARAMETER	MIN	MAX	UNIT	REMARKS
ICC	Vcc power supply current		18	mA	
VIH	input high level	1.9	Vcc	V	[illegible] (XTAL, RI, START)
VIL	input low level	-.5	0.9	V	
ILO	data bus leakage		10	µA	[illegible] impedance state
VOH	output high level	2.4		V	IOH = [illegible] µA
VOL	output low level		0.45	V	IOL = [illegible] mA
FCY	crystal frequency	3.5	12	MHz	[illegible] circuits

Basic Message Structure

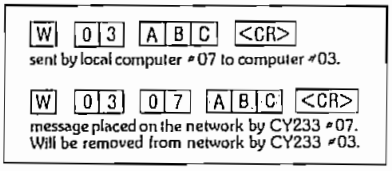
ASCII Message Format

When multiple CY233s are connected in a network, messages cause data to flow between nodes of the network. A message is a simple structure that consists of four elements: a command letter, an address, the data contents, and a message terminator. A typical message is shown below:



- This message will write 'A .. Z<cr> to the device attached to the network as address #03. The above message format is typical of Host Ring and Party Line networks in which a host computer sends messages to devices.

When most devices on the net are computers, it is desirable to identify the source as well as the destination of the message. This is done automatically in LAN networks, both Peer Ring and Host Ring LANs. A local computer creates a write message with data and a target address to which the data is to be sent. The local CY233 appends its address to the target address, as it places the message on the network.



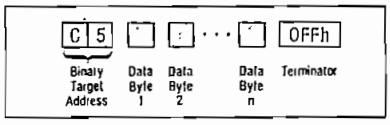
The above structure allows any computer on the network to communicate with any other unambiguously. Network query and status commands allow each local computer to poll the network for automatic configuration.

Token Support

CY233s recognize local intelligent devices and perform special functions in support of token passing modes in which local devices wait for the 'token' before placing their message on the network. Watchdog timeout functions prevent the token from being 'stuck' at any node that does not respond. Each local CY233 takes responsibility for keeping the token passing, if its local device fails to respond in a specified period.

Binary Message Format

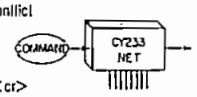
For simplest modes, such as the Wire Saver, a simple binary mode message uses only the destination address, data contents, and the special terminator, 0FFh. All messages in this mode are implied writes. For example:



CY233 Commands

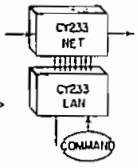
All CY233 commands are listed below. While many are self explanatory, the interpretation of some of them is dependent upon the network topology and operating mode. A complete description is beyond the scope of this data sheet, however a detailed user manual (~150 pages) is available from Cybernetic Micro Systems.

Network Serial Commands



@	Test Network Link, @aa<cr>
A_F	Undefined, Hex Character conflict
G	Reserved command letter
H	Undefined
I	Initialize CY233, Iaa<cr>
J	Enable Echo All mode, Jaa<cr>
K	Enable parallel Error status, Kaa<cr>
L	Undefined
M	Fill consecutive locations, Maahh_hh<cr>
N	Display consecutive locations, Naacchh_hh<cr>
O	Reserved command letter
P	Set Periodic Master mode delay, Paattt<cr>
Q	Query for serial error status, Qaass<cr>
R	Read from parallel device, Raad_d<cr>
S	Sense local address, Saabb_11<cr>
T	Set handshake Timeout delay, Taattt<cr>
U	Token message, Uaa<cr>
V	Undefined
W	Write to parallel device, Waad_d<cr>
X	Transfer to parallel device, Xaad_d<cr>
Y_	User Reserved, ASCII 59h_5Fh

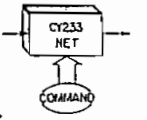
LAN Serial Commands



@	Return LAN Node Address, @aa<cr>
G	Reserved command letter
I	Initialize CY233, Iaa<cr>
O	Reserved command letter
P	Pass Parallel Command to Network CY233, PCmd<cr>
Q	Query for serial error status, Qaass<cr>
R	Generate Read Message to Network, Raad_d<cr>
T	Set LAN handshake Timeout delay, Taattt<cr>
W	Generate Write Mesage to Network, Waad_d<cr>

Commands not listed above should not be used in a Local Area Network.

Parallel Commands



G	Reserved command letter
I	Initialize CY233, I<cr>
J	Enable Echo All mode, J<cr>
K	Enable parallel Error status, K<cr>
L	Load value of Status byte 2, Lvv<cr>
O	Reserved command letter
P	Set Periodic Master mode delay, Ptttt<cr>
S	Sense nodes on Ring, S<cr>
T	Set handshake Timeout delay, Tttt<cr>
U	Generate Token message, Uaa<cr>

Only the commands listed above are available as Parallel commands. Any others will set the parallel status error bit.

4-Phase Stepping Motor Manufacturers

AEROTECH, INC.
101 Zeta Dr.
Pittsburgh PA 15238 USA
(412) 963-7470

AIRPAX
NORTH AMERICAN PHILIPS
604 West Johnson Avenue
Cheshire CT 06410 USA
(203) 271-6000

AMERICAN ELECTRONICS INC.
P.O. Box 3070
Fullerton CA 92634 USA
(714) 871-3020

B&B MOTOR & CONTROL CORP.
B&B Long Island City
Apple Hill Commons
Burlington CT 06013 USA
(203) 673-7151; (800) 638-7807

BERGER-LAHR CORP.
41170 Joy Road
Plymouth MI 48170 USA
(313) 459-8300

BODINE ELECTRIC CO.
2500 W. Bradley Place
Chicago IL 60618 USA
(312) 478-3515

CANON USA
ELECTRONIC. COMPS. DIV.
One Canon Plaza
Lake Success NY 11042 USA
(516) 488-6700

CLIFTON PRECISION
LITTON SYSTEMS, INC.
Maple at Broadway
Clifton Heights PA 19018 USA
(215) 622-1000

CROUZET CONTROLS, INC.
1083 State Pkwy.
Schaumburg IL 60173 USA
(312) 843-2200

DENSITRON CORP. OF AMERICA
2039 Hwy. 1 S., P.O. Box 1318
Camden SC 29020 USA
(803) 432-5008

EASTERN AIR DEVICES
MOTOR DIVISION
1 Progress Drive
Dover NH 03820 USA
(603) 742-3330

ECM MOTOR COMPANY
1301 E. Tower Road
Schaumberg IL 60196 USA
(312) 885-4000

ELECTRIC MOTOR
ENGINEERING
1847 Victory Blvd.
Glendale CA 91201 USA
(818) 502-1866

HAYDON SWITCH &
INSTRUMENT
1500 Meriden Road
Waterbury CT 06705 USA
(203) 756-7441

HURST MANUFACTURING CORP
EMERSON ELECTRIC CO.
P.O. Box 326
Princeton IN 47670 USA
(812) 385-2564

ICON CORPORATION
156 6th Street
Cambridge MA 02142 USA
(617) 868-5400

INLAND MOTOR
4020 East Inland Road
Sierra Vista AZ 85635 USA
(602) 459-1150

ISKRA ELECTRONICS INC.
222 Sherwood Ave.
Farmingdale NY 11735 USA
(516) 753-0400; (800) 862-2101

JAPANESE PRODUCTS CORP.
7 Westchester Plaza
Elmsford NY 10523 USA
(914) 592-8880

MESUR-MATIC ELECTRONICS
CORPORATION
10 Colonial Road Suite 6
Salem MA 01970 USA
(617) 745-7000

MICRO MO ELECTRONICS INC.
742 2nd Avenue S.
St. Petersburg FL 33701 USA
(813) 822-2529

MOLON MOTOR & COIL
CORPORATION
3737 Industrial Avenue
Rolling Meadows IL 60008 USA
(312) 259-8700

MOTION CONTROL GROUP
AMERICAN PRECISION IND.
3229 Roymar Road
Oceanside CA 92054 USA
(619) 439-7500

NOVATRONICS OF CANADA LTD
677 Erie Street, P.O. Box 610
Stratford, Ontario, Canada N5A 6V6
(519) 271-3880

ORIENTAL MOTORS USA
2701 Plaza Del Amo Suite 702
Torrance CA 90503 USA
(213) 515-2264

PORTESCAP U.S. INC.
36 Central Avenue
Hauppauge NY 11788 USA
(516) 234-3900

P M I MOTION TECH.
KOLLMORGEN CORP.
49 Mall Drive
Commack NY 11725 USA
(516) 864-1000

PACIFIC SCIENTIFIC MCD
4301 Kishwaukee Street
Rockford IL 61109 USA
(815) 226-3100

SINGER COMPANY
Route 70
Black Mountain NC 28711 USA
(704) 686-3811

STOCK DRIVE PRODUCTS
DIV. OF DESIGNATRONICS INC.
2101 Jericho Turnpike
New Hyde Park NY 11040 USA
(516) 328-0220; (516) 328-3300

SUPERIOR ELECTRIC CO.
383 Middle Street
Bristol CT 06010 USA
(203) 582-9561

VERNITRON CORP.
VERNITECH DIVISION
300 Marcus Boulevard
Deer Park NY 11729 USA
(516) 586-5100

WARNER ELECTRIC BRAKE &
CLUTCH
449 Gardener Street
South Beloit IL 61080 USA
(815) 389-3771

b_7 b_6 b_5 → / Bits					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Row ↓ / Column →	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS ←	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF ↓	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT ↑	ESC	+	;	K	[	k	{
1	1	0	0	12	FF →	FS	,	<	L	\	l	¦
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL RUB

Figure 5.1 ASCII Character Set