UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DESARROLLO DE UNA INTERFAZ COMPUTACIONAL PARA EL CONTROL DE MICROMÁQUINAS-HERRAMIENTA

T E S I S

PARA OBTENER EL GRADO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

ALFREDO YÁÑEZ SOTO

Dirigida por:

Dr. Alberto Caballero Ruiz
Dr. Leopoldo Ruiz Huerta

CIUDAD UNIVERSITARIA MARZO, 2007.

Agradecimientos

A mis padres por el apoyo, la educación, el ejemplo y el amor que me han brindado.

A mis hermanas, familiares y amigos, por su ánimo y consejos.

A mis tutores Alberto y Leopoldo, por sus recomendaciones, consejos y aportaciones a este trabajo.

A todos los compañeros y amigos del Lab.: Oscar, Bogar, Memo, Héctor, Gengis, etc.; por su apoyo y amistad.

A mis sinodales por sus correcciones, comentarios y aportaciones.

ÍNDICE TEMÁTICO

ÍNDICE DE FIGURAS Y TABLAS

Figura Págs.

Tabla Págs.

INTRODUCCIÓN

En las últimas décadas la microtecnología ha tomado un papel muy importante con aplicaciones en diversas áreas como: la aeroespacial, la automotriz, la médica, las telecomunicaciones, las tecnologías de la información, entre otras; donde se desarrollan productos cada vez más pequeños que ofrecen muchas ventajas frente a sus antecesores de mayor tamaño. En estas aplicaciones se conjugan diversas tecnologías como son: la microóptica, micromecánica, microelectrónica, etc. Dentro de la micromecánica existen diversas tecnologías para la fabricación de piezas micromecánicas, como es la tecnología MEMS (*MicroElectroMechanical Systems*), MST (*MicroSystem Tecnology*) y MMT (*MicroMachine Technology*). Dentro de esta última tecnología, en el Laboratorio de Micromecánica y Mecatrónica (LMM) del Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la Universidad Nacional Autónoma de México (UNAM), se construyó un prototipo de micromáquina herramienta que realiza funciones de torneado, fresado y taladrado. Este prototipo presenta un control que hace uso de las capacidades de una computadora personal (PC), la cual se comunica con la micromáquina herramienta a través del puerto paralelo. Para cada pieza que se desea manufacturar se realiza un programa diferente, en lenguaje C++. Sin embargo, este procedimiento resulta tardado y poco conveniente si lo que se busca es diversidad en la producción de piezas micromecánicas. Por tal razón, en el presente trabajo se diseña y desarrolla una aplicación que interpreta instrucciones de código numérico estándar, las cuales pueden ser generadas de forma automática a partir de un software de manufactura asistida por computadora (CAM). Estas instrucciones son procesadas por la aplicación y a partir de ellas se generan señales de control que son enviadas a través del puerto paralelo de la PC hacia los actuadores de la micromáquina herramienta para lograr la manufactura de piezas.

En el primer capítulo, de antecedentes, se habla de la importancia que tiene la automatización dentro de los procesos industriales y las formas que existen de

automatización en la industria moderna. Dentro de estas formas de automatización se habla del control numérico computarizado, que constituye una forma de automatización programable. Se da un panorama general del control numérico en máquinas herramientas, del desarrollo de la micromecánica en el mundo y se mencionan aplicaciones del microequipo en diversos sectores como el automotriz, el aeroespacial, la medicina, las telecomunicaciones, las tecnologías de la información, entre otros. Para finalizar este capítulo, se habla de la micromecánica en el CCADET y del prototipo de micromáquina herramienta construido en el LMM.

En el segundo capítulo se plantea la problemática que se tiene en la manufactura de piezas con la micromáquina herramienta. Se establece el objetivo y la metodología a seguir para el presente trabajo.

En el tercer capítulo se habla del análisis, diseño e implementación de la aplicación desarrollada. Donde se analizan varios elementos, a partir de los cuales se establecen los requerimientos de la aplicación. Se evalúan alternativas para el desarrollo del proyecto. Se describe el funcionamiento de la aplicación y de los elementos que conforman el sistema de control, así como la configuración de algunos de estos elementos. Y se detalla la aplicación a nivel de usuario y a nivel de programador.

En el cuarto capítulo se describen las pruebas realizadas de maquinados de piezas más representativas con el fin de probar la eficiencia del nuevo sistema de control. Se analizan los resultados obtenidos y se presentan las conclusiones de este proyecto.

Capítulo 1

ANTECEDENTES

1.1. La automatización como respuesta a la situación actual de los procesos de producción.

En los últimos años, la tendencia a la globalización y la segmentación internacional de los mercados ha crecido. Todo esto habla de una libre competencia y surge la necesidad de adecuar las industrias a fin de que puedan satisfacer el reto de los próximos años. Una opción frente a esto es la reconversión de las industrias mediante la automatización. Sin embargo, debe hacerse en la forma adecuada, de modo que se pueda absorber gradualmente la nueva tecnología en un tiempo razonable; todo esto sin olvidar los factores del rendimiento de la inversión y de la capacidad de producción.

Entre las formas de automatización que se aplican en la industria moderna, se encuentran las siguientes:

- El control automático de procesos.
- El procesamiento electrónico de datos.
- La automatización fija.
- El control numérico computarizado.
- La automatización flexible.

El **control automático de procesos,** se refiere al manejo de procesos caracterizados por diversos tipos de cambios (generalmente químicos y físicos); un ejemplo de esto podría ser el proceso de refinación del petróleo.

El control automático [6] es el mantenimiento del valor de una cantidad o condición respecto a un valor deseado. Esto se consigue midiendo el valor de la variable a controlar y comparándola con el valor deseado para proceder a reducir la diferencia entre estos dos valores. En consecuencia, el control automático exige un lazo cerrado de acción y reacción que funciona sin intervención humana. En la figura 1.1 se muestra el diagrama de bloques del control en lazo cerrado.

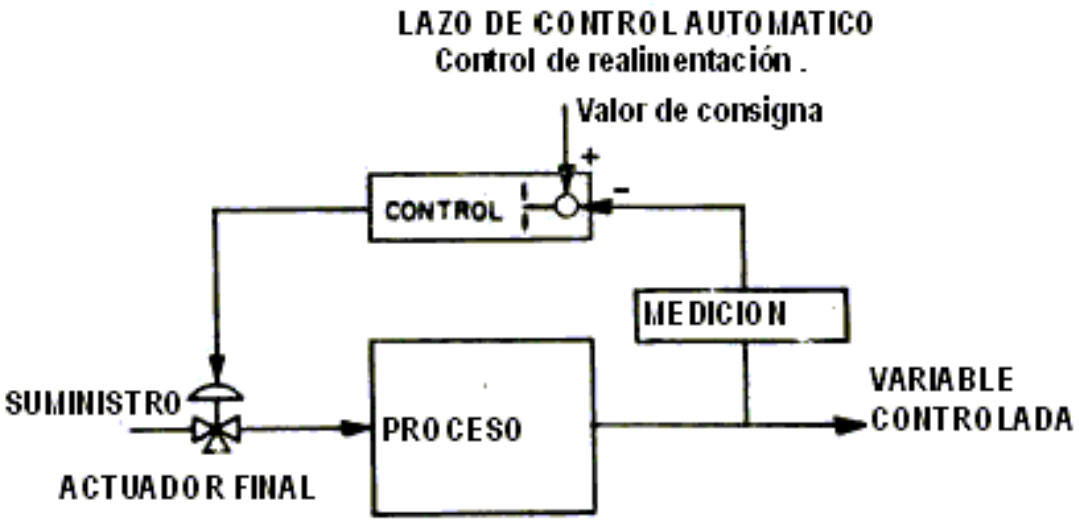

Figura 1.1. Diagrama de bloques para el control de lazo cerrado.

El elemento más importante de cualquier sistema de control automático es el control en lazo cerrado (ver figura 1.1). Un ejemplo en el que podemos encontrarlo, es el mecanismo de piloto automático y el avión que controla. Su objetivo es mantener una dirección específica del avión, a pesar de los cambios atmosféricos. El sistema ejecutará su tarea midiendo continuamente la dirección instantánea del avión y la comparará con el valor de consigna (set-point en Inglés), con el fin de ajustar automáticamente la diferencia; para ajustar dicha diferencia, el sistema de control del avión moverá algunos mecanismos que modifican la dirección del avión (timón, aletas, etc.), de modo que la posición instantánea coincida con la especificada. El piloto u operador, quien fija con anterioridad el piloto automático, no forma parte del sistema de control.

El **procesamiento electrónico de datos** [7] frecuentemente está relacionado con los sistemas de información, centros de cómputo, etc. Sin embargo, en la actualidad también se considera dentro de estos la obtención, análisis y registros de datos a través de interfases y computadoras.

Figura 1.2. Elementos que conforman un sistema de información.

Un sistema de información (SI), como se aprecia en la figura 1.2, es un conjunto de elementos o componentes interrelacionados para recolectar (entrada), manipular

(proceso) y diseminar (salida) datos e información, y en todo momento se cuenta con un mecanismo de realimentación. Todos estos elementos trabajan en conjunto con el fin de cumplir una tarea.

Un ejemplo de un sistema de información, es el proceso del pago de la nómina de los empleados de una empresa. Las tarjetas de registro de entrada y salida de cada empleado representan la entrada al sistema. El procesamiento de esa información consiste en realizar los cálculos pertinentes para obtener el pago neto de cada empleado. Esto implica, multiplicar el número de horas trabajadas por el índice salarial por hora del empleado para obtener el pago bruto y considerar las horas extras trabajadas, realizando las deducciones correspondientes a impuestos y prestaciones, para obtener el pago neto. La salida consiste en la impresión de los cheques para los empleados. En cuanto a la realimentación, ésta consiste en detectar errores o problemas durante las actividades de entrada y de procesamiento. Por ejemplo, si al momento de capturar la cantidad de horas trabajadas por un empleado se escribió 400 horas en vez de 40, el sistema de información determinará que la cifra de 400 horas rebasa la escala de horas que puede trabajar un empleado durante una semana, proporcionando la realimentación al sistema en forma de un mensaje de error. De no detectarse este error, se imprimiría en el cheque una cifra muy elevada.

Más específicamente, un sistema de información basado en computadoras (SIBC) está compuesto por hardware, software, bases de datos, telecomunicaciones, personas y procedimientos específicamente configurados para recolectar, manipular, almacenar y procesar datos para ser convertidos en información.

En el nivel corporativo, los tipos de sistemas de información de uso más común en las organizaciones comerciales son los sistemas de procesamiento de transacciones (TPS, *Transaction Processing System*) y comercio electrónico, los sistemas de información administrativa (MIS, *Management Information System*), los sistemas de apoyo para la toma de decisiones (DSS, *Decision Support System*) y los sistemas expertos; en conjunto, auxilian a los empleados de las organizaciones en la ejecución de tareas tanto rutinarias como especiales, desde el registro de las ventas hasta el procesamiento de la nómina, el apoyo para la toma de decisiones de varios departamentos y la propuesta de alternativas a proyectos y oportunidades de gran escala.

La **automatización fija,** es aquella asociada al empleo de sistemas lógicos tales como: los sistemas de relevadores y compuertas lógicas; sin embargo, estos sistemas se han ido flexibilizando con la introducción de algunos elementos de programación como el caso de los Controladores Lógicos Programables (PLC'S).

La automatización fija [8] se utiliza cuando el volumen de producción es muy alto y por tanto es adecuada para diseñar equipos especializados para procesar el producto (o un componente de un producto) con alto rendimiento y con elevadas tasas de producción. Un buen ejemplo de la automatización fija, como se muestra en la figura 1.3, es el de embotelladoras de bebida. Uno de los problemas que presenta la automatización fija es que el equipo está especialmente diseñado para obtener un sólo producto, y una vez que se halla acabado el ciclo de vida del mismo es probable que el equipo quede obsoleto.

Figura 1.3. Máquina embotelladora de bebida (ejemplo de automatización fija).

La **automatización programable** se emplea cuando el volumen de producción es relativamente bajo y hay una diversidad de producción a obtener. En este caso el equipo de producción está diseñado para ser adaptable a variaciones en la configuración del producto. Esta característica de adaptabilidad se realiza haciendo funcionar el equipo bajo el control de un "programa" de instrucciones que se preparó especialmente para el producto dado. El programa se introduce en el equipo de producción y este último realiza una secuencia particular de operaciones de procesamiento (o montaje) para obtener el producto. En términos de economía el costo del equipo programable puede repartirse entre un gran número de productos aún cuando sean diferentes. Gracias a la característica de programación y a la adaptabilidad resultante del equipo, muchos productos diferentes y únicos en su género pueden obtenerse económicamente en pequeños lotes. Como ejemplo de esto se encuentra el **control numérico computarizado.** Este tipo de control se ha aplicado con éxito a Máquinas Herramientas de Control Numérico (MHCN). Entre las que podemos encontrar: fresadoras CNC, tornos CNC, máquinas de electro-erosión, máquinas de corte por hilo, etc.

La **automatización flexible** o también conocido como sistema de fabricación integrado por computadora o celda de manufactura flexible, es adecuada para un rango de producción de volumen medio, debido a que toma características tanto de la automatización fija como de la automatización programable. Los sistemas automatizados flexibles suelen estar constituidos por una serie de estaciones de trabajo que están interconectadas por un sistema de almacenamiento y manipulación de materiales. Una computadora central se encarga de controlar las diversas actividades que se producen en el sistema, encaminando las diversas piezas a las distintas etapas de producción y controlando las operaciones programadas en las distintas etapas. En la figura 1.4 se muestra un ejemplo de este tipo de automatización flexible.

Figura 1.4. Celda de manufactura flexible que realiza operaciones de punzonado EDM y corte de electrodos.

1.2. El control numérico en máquinas herramienta.

Los cambios en la demanda de productos de las últimas décadas y la evolución de la microelectrónica han llevado a replantear completamente el concepto de producción y a incorporar la automatización de procesos. En este marco, el control numérico es un ejemplo de automatización programable que se diseñó para adaptar las variaciones en la configuración de los productos. Su principal aplicación se centra en volúmenes de producción bajos y medios. La *Electronic Industries Association* (EIA) define el control numérico (CN) como "un sistema en el cual las acciones son controladas por la inserción directa de datos numéricos en algún punto y el sistema debe interpretar, al menos, alguna porción de estos datos". En el CN, los números forman un programa de instrucciones diseñado para realizar una tarea o manufacturar una pieza en particular y cuando cambia la operación a realizar, cambia el programa. Esta capacidad para cambiar el programa para cada nueva tarea es lo que da al CN su flexibilidad [4].

La aplicación del CN abarca gran variedad de procesos. Estas aplicaciones pueden dividirse en dos categorías:

1ª. Aplicaciones con máquinas herramienta, tales como el taladro, laminado, torneado, etc.
2ª. Aplicaciones sin máquinas herramienta, tales como el ensamble, trazado e inspección.

El principio de operación común de todas las aplicaciones del control numérico es el control de la posición relativa de una herramienta o elemento para procesar con respecto al objeto a ser procesado.

Hoy en día las Máquinas Herramienta de Control Numérico por Computadora (MHCNC) son un elemento básico en todo sistema de fabricación y han llegado a reemplazar a las máquinas herramienta convencionales debido a las ventajas que presentan sobre estas

últimas [3], como son por ejemplo:

1. Reducción de los tiempos de fabricación.
2. Reducción del tamaño del lote económico y, por lo tanto, del nivel de almacenes.
3. Aumento de la flexibilidad de producción expresada en términos de fácil adaptabilidad a la realización de distintos tipos de productos manufacturados, respondiendo ágilmente a las necesidades del mercado.
4. Disminución de rechazos de piezas, debido a una mayor precisión de las máquinas.
5. Mayor duración de las herramientas, debido a su mejor aprovechamiento.
6. Supresión del trazado de piezas antes del mecanizado.
7. Ahorro de utillaje, al realizar en una máquina mayor número de operaciones.
8. Posibilidad para realizar de manera económica piezas de geometría complicada.
9. Mejora de la gestión de la fabricación (tiempos más uniformes).
10. Mejora de la seguridad al reducirse el grado de interactividad máquina-operario durante los procesos de maquinado.
11. Menor número de operarios para el manejo de las máquinas.

Algunas desventajas [3] que presentan las MHCNC son:

1. Costo horario elevado por la importante inversión de adquisición de una MHCNC, debido no sólo al precio de la misma sino también a los elementos auxiliares. Ello obliga a asegurar un alto nivel de ocupación de la máquina y la puesta de varios turnos del equipo para conseguir una amortización razonable.

2. Necesidad de personal más calificado en programación y mantenimiento, lo que se traduce en mayores costos en la formación de recursos humanos y en salarios.

3. Alto costo del servicio postventa y de mantenimiento de los equipos en razón de su mayor complejidad. Se estima que el costo de mantenimiento de una MHCNC es un 50% más elevado que en las convencionales.

4. Necesidad de un tiempo de adaptación y de un cambio en la estructura organizacional y de gestión de la fabricación. No es fácil adaptar a los empleados a las nuevas técnicas exigidas por el CNC.

1.2.1. Componentes básicos de un sistema CN.

Los componentes básicos de un sistema CN [4] son:

1. Programa de instrucciones.
2. Unidad de control.
3. Máquina herramienta.

1.2.1.a. Programa de instrucciones.

El programa de instrucciones [4] es un conjunto de directrices paso a paso que indican a la máquina qué hacer. Es codificado en alguna forma simbólica o numérica y almacenado dentro de algún tipo de medio (cinta magnética, disco duro, disco flexible, etc.), para ser interpretado posteriormente por la unidad de control. Existen dos formas de ingresar este programa de instrucciones a la unidad de control: manualmente y asistido por computadora.

a) Forma manual [4].

Si la geometría de la pieza es simple, es razonable que un programa CN sea escrito manualmente. Por ejemplo, si la pieza requiere solamente cortes rectangulares que involucren el mecanizado en los ejes X y Y, la programación es eficiente. Éste también es el caso para los procesos “punto-a-punto”, como el taladrado. Los programas escritos manualmente pueden crearse de muchas maneras, siendo la más frecuente el uso del teclado alfanumérico perteneciente a la MHCN.

Un segundo método es desarrollar programas CN en una computadora, usando un editor de texto y almacenando dicho programa en algún tipo de medio. Posteriormente el programa puede trasladarse al controlador de la máquina, mediante una unidad de lectura de disco flexible o alguna otra vía de comunicación que soporte la MHCN, para que se ejecute.

Y una última forma es usar un programa interactivo. Este programa solicita al usuario la información geométrica de la pieza, las especificaciones de las herramientas de corte y la ubicación de la pieza sobre la máquina y genera un código CN “limpio” para fabricar la pieza. Este tipo de programas son muy adecuados para reducir los tiempos de programación y para tratar piezas con geometrías complejas.

b) Forma Automática [4].

La programación automática o asistida por computadora es hecha independientemente del equipo de manufactura; en ella se codifica directamente la tarea, que ha sido definida por el usuario, en un programa-pieza [4].

En un software de Manufactura Asistida por Computadora (CAM), es posible obtener a partir de un Diseño Asistido por Computadora (CAD), programas-pieza en código numérico.

1.2.1.a.1. Código G y M.

El código numérico estándar que utilizan las máquinas CNC es el código G y M que forman parte del estándar ANSI/EIA-RS-274-D. En este caso, el programa-pieza es estructurado; contiene datos de cabecera, definiciones técnicas y geométricas, instrucciones de ejecución e instrucciones de fin de programa.

El código G corresponde a un conjunto de instrucciones denominadas funciones preparatorias, las cuales se relacionan con el modo de posicionamiento de la herramienta. Algunas de estas funciones preparatorias [1] son:

- G00 Posicionamiento punto a punto.
- G01 Interpolación lineal para dimensiones medias.
- G10 Interpolación lineal para dimensiones grandes.
- G11 Interpolación lineal para dimensiones pequeñas.
- G02 Interpolación circular para dimensiones medias en sentido horario.
- G03 Interpolación circular en sentido antihorario.
- G20 Interpolación circular para dimensiones grandes y sentido horario.
- G21 Interpolación circular para dimensiones pequeñas y sentido horario.
- G30 Interpolación circular para dimensiones grandes y sentido antihorario.
- G31 Interpolación circular para dimensiones pequeñas y sentido antihorario.
- G33 Fileteado de paso constante.
- G34 Fileteado de paso creciente.

- G35 Fileteado de paso decreciente.
- G81, G82, G83, G84, G85, G86, G87, G88, G89 Ciclos fijos.

El código M corresponde a un conjunto de instrucciones también conocidas como funciones auxiliares [1]. Estas instrucciones corresponden a acciones complementarias que deben tomar tanto la máquina herramienta como el control. Algunas de estas instrucciones son:

- M00 Parada programada.
- M01 Parada facultativa.
- M02 Fin de programa.
- M03 Rotación del husillo en sentido horario.
- M04 Rotación del husillo en sentido antihorario.
- M05 Parada de husillo.
- M06 Cambio de herramienta.
- M07 Refrigeración 1 en marcha.
- M08 Refrigeración 2 en marcha.
- M09 Parada de la refrigeración.
- M13 Rotación del husillo en sentido horario y refrigeración.
- M14 Rotación del husillo en sentido antihorario y refrigeración.
- M19 Parada del husillo con orientación determinada.
- M30 Fin de programa y vuelta a la secuencia número 1.
- M38 Gama de velocidades de rotación 1.
- M39 Gama de velocidades de rotación 2.
- M50 Refrigeración 3 en marcha.
- M51 Refrigeración 4 en marcha.
- M60 Cambio de pieza.
- M68 Tomar la pieza.
- M69 Soltar la pieza.
- M79 Desplazamiento de la mesa.

Un ejemplo de sintaxis para un programa de instrucciones en código numérico es el siguiente [1]:

N4, G2, X ± 4.3, Y ± 4.3, Z ± 4.3, I ± 4.3, J ± 4.3, K ± 4.3, F4, S4, T2, M2, LF.

Donde:

N4: Indica el número de bloque y está representado por cuatro dígitos que pueden ir desde el 1 hasta el 9999.

G2: Representa una función preparatoria (instrucción en código G), la cual se expresa con la letra G seguida por dos dígitos. Por ejemplo G00, G01, etc.

X ± 4.3
Y ± 4.3
Z ± 4.3

Para las coordenadas de un punto, se escribe la letra del eje seguida por el valor de su coordenada. Este valor está formado por 7 dígitos (4 para la parte entera y tres para la parte decimal).

I ± 4.3
J ± 4.3
K ± 4.3

Esta parte se emplea en la interpolación circular. Los valores programados bajo las direcciones I, J, K están relacionados con las direcciones de los ejes X, Y, Z, respectivamente. Cada uno de estos valores está formado por 7 dígitos (4 para la parte entera y tres para la parte decimal).

F4 Representa a la función de avance, la cual se expresa con la letra F seguida por un número de 4 dígitos que indica la velocidad en mm/min.

S4 Representa a la función de velocidad de giro del eje principal, la cual se expresa con la letra S seguida por un número de 4 dígitos.

T2 Representa a la función de selección de herramienta, la cual se expresa con la letra T seguida por un número de 2 dígitos. Este número sirve para especificar la herramienta seleccionada.

LF Indica el fin del bloque.

1.2.1.b. Unidad de control.

El segundo componente básico en un sistema CN es la unidad de control [4], la cual contiene componentes electrónicos que leen e interpretan el programa de instrucciones y lo convierten en acciones mecánicas para la máquina herramienta. Los elementos que la conforman son:

- Unidad de entrada o lector de datos.
- Unidad de memoria fija (ROM) y volátil (RAM).
- Uno o varios procesadores.
- Unidad de enlace con la máquina.

En la actualidad también se utilizan PC's como reemplazo de la unidad de control, obteniendo así el mismo funcionamiento.

1.2.1.c. Máquina herramienta.

Por último, la máquina herramienta [4] es la parte del sistema CN que realiza el trabajo u operaciones de mecanizado. Una máquina herramienta consiste en una mesa de trabajo, cabezales porta herramientas, motores y los controles necesarios para mover los elementos de la máquina en forma automática. También incluye herramientas de corte, de sujeción y un almacén de herramientas. Algunas máquinas poseen, además, elementos automáticos de entrada y salida de piezas y de cambio de herramientas.

1.2.2. Aspectos generares de la tecnología de fabricación de las MHCN.

Para que una máquina herramienta realice los desplazamientos correspondientes a los cálculos efectuados por el control de la MHCN es necesario contemplar varios puntos en la fabricación de MHCN. Entre los puntos más importantes están:

- El tipo de control para el posicionamiento de la máquina herramienta.
- La forma de medir los desplazamientos de la máquina herramienta.
- Las características de diseño de la máquina herramienta.
- El cambio automático de herramientas.

1.2.2.a. Sistemas de control de posicionamiento.

El control numérico tiene por objetivo conducir automáticamente la máquina herramienta a una posición determinada. Cuando una orden ha sido emitida, es preciso cerciorarse de que la misma adopta la posición exacta. Para ello existen dos tipos de sistemas:

a) Sistema de lazo abierto.

b) Sistema de lazo cerrado.

En el sistema de lazo abierto se suprime el retorno de la información del sensor de posición, el cual sí existe en un sistema de bucle cerrado, y en este caso se emplean motores de pasos. El principio de funcionamiento de estos motores se presta para que trabajen de forma adecuada máquinas que poseen un sistema para el conteo de los pulsos enviados a los motores.

El sistema de lazo cerrado consiste en comparar en todo momento la posición de un elemento móvil con la orden dada. La señal que se envía al motor es función de la relación entre la posición y la orden (ver figura 1.5).

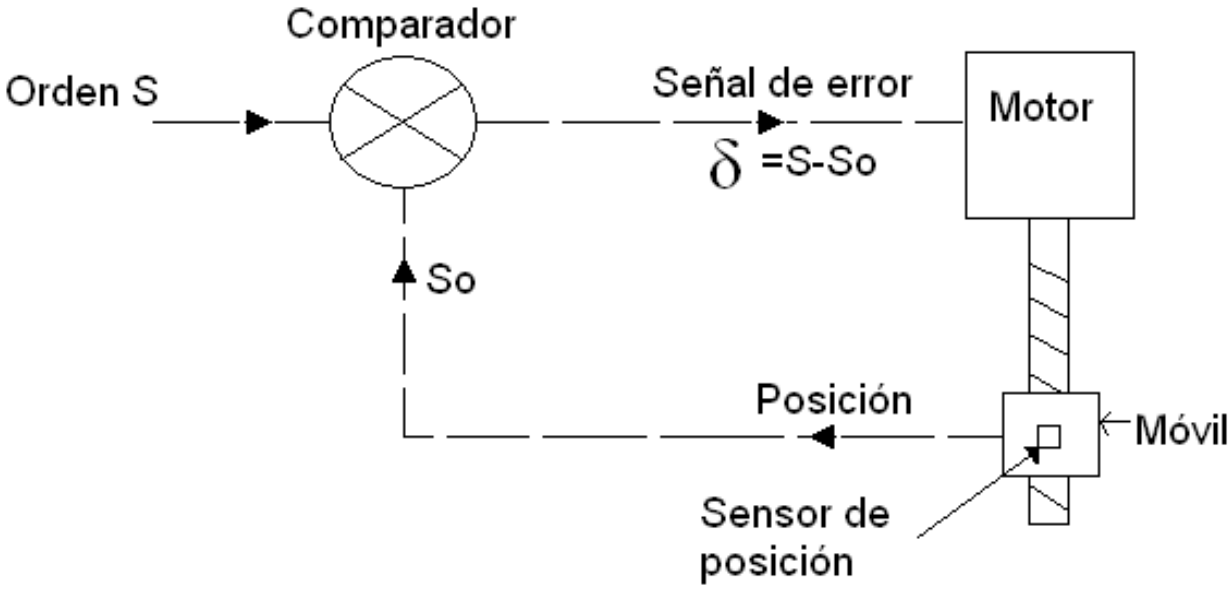

Figura 1.5. Sistema de bucle cerrado.

Las máquinas que utilizan este sistema normalmente tienen dos vías de retorno de información, una para el control de la posición y otra para el control de la velocidad de desplazamiento del móvil, ya que momentos antes de alcanzar el desplazamiento deseado disminuye la velocidad del móvil para conseguir un posicionado correcto. Para este tipo de sistema de control se utilizan motores de corriente continua, corriente alterna o hidráulicos.

1.2.2.b. Medición de los desplazamientos.

La medición de los desplazamientos o de las posiciones de los elementos controlados es la base de los sistemas de control numérico que funcionan en lazo cerrado. Esta labor se realiza mediante un sensor de posición.

El papel que juega el sensor de posición es el de transformar el desplazamiento, de un elemento móvil, de una magnitud mecánica a una magnitud eléctrica, para ser analizado por el equipo de control y proceder a su tratamiento correspondiente.

Existe una gran variedad de sensores de posición, los cuales se pueden clasificar en función de los siguientes conceptos [1]:

- Por la naturaleza de la información que manejan: analógica o digital.
- Por la relación entre la magnitud mecánica y la magnitud eléctrica: absoluta o incremental.
- Por el emplazamiento del sensor en la cadena de control: medida directa o indirecta.
- Por la forma física del sensor: lineal o rotativo.

Aunque existe una gran variedad de sensores de posición en el mercado, en la actualidad dos de los más empleados en MHCN son: el *resolver* y el *inductosyn*. Estos son dos sensores analógicos: el primero trabaja frecuentemente como sensor indirecto y el segundo, como sensor directo.

1.2.2.c. Características de diseño.

Las principales características funcionales de las MHCN son la precisión y la capacidad de arranque de material, y por ello requieren unas características de diseño y construcción mejores que las máquinas convencionales.
Los factores más importantes que se han de tener en cuenta para conseguir esta precisión y capacidad de arranque de material son [1]: holguras, rozamientos, deformaciones, vibraciones, desalineaciones, rigidez, etc. Y con el fin de cuidar este tipo de factores se emplean diversos elementos en la construcción de MHCN, como por ejemplo: husillos de bolas, guías de rodadura (patines), guías hidrostáticas, etc.

1.2.2.d. Cambio automático de herramientas.

El cambio automático de herramientas es otra de las particularidades de las MHCN, ya que se reduce los tiempos necesarios para cambiarlas manualmente.

Las herramientas necesarias para el maquinado de la pieza que se está trabajando están codificadas, de tal forma que, cuando se precisa cambiar de herramienta para realizar otra operación, este cambio lo hace automáticamente la máquina.

Entre los distintos sistemas de cambio de herramienta están: el cambio por giro de torreta y el cambio por desplazamiento de un brazo giratorio.

1.3. Desarrollo de micromecánica en el mundo.

El avance tecnológico de las últimas décadas ha llevado a realizar componentes cada vez de menor tamaño. El desarrollo de estos productos ha sido impulsado principalmente por las ventajas que ofrecen respecto a sus antecesores de mayor tamaño, como son: un aumento en su velocidad de funcionamiento, reducción de costos de operación, mayor precisión, etc.

Las técnicas desarrolladas desde principios de los 80's para la industria microelectrónica fueron extendiéndose progresivamente a otros campos. Así surgieron diversas áreas microtecnológicas, como la micromecánica, microacústica, microóptica, etc., las cuales, a su vez, se combinan entre sí para crear microsistemas integrados con distintas tecnologías.

En diferentes partes del mundo se desarrollan tecnologías basadas en la microelectrónica para la miniaturización de sistemas, por ejemplo, en los Estados Unidos de Norteamérica se desarrollan los *MicroElectroMechanical Systems* (MEMS), en Europa los *MicroSystem Tecnology* (MST), y finalmente los *MicroMachine Technology* (MMT) que se desarrollan principalmente en Japón [16].

La tecnología MEMS tiene una gran aplicación en diversos sectores como: el automotriz, donde encontramos acelerómetros; en la industria biotecnológica, en la que se han desarrollado biochips para detección de agentes biológicos; en la industria de la computación, donde encontramos las cabezas para impresoras de inyección de tinta; etc[17]. Sin embargo, esta tecnología no permite producir partes cónicas, tornillos u otros componentes con formas tridimensionales complejas; así también, presenta problemas de ensamblaje y frecuentemente se requiere de un estudio específico para producir un nuevo microdispositivo; además, se deben emplear sólo materiales compatibles con la tecnología de silicio, y se requiere el uso de tecnología avanzada y muy costosa.

Como alternativa para generar micromecanismos se encuentra el uso de tecnologías mecánicas convencionales utilizando, ya sea, máquinas herramienta superprecisas o micromáquinas herramienta y micromanipuladores. El uso de máquinas herramienta superprecisas representa un mayor costo en el desarrollo de productos comerciales, mientras que el uso de micromáquinas herramientas y micromanipuladores nos ofrece mayores ventajas en cuanto a espacio, consumo de energía, entre otras.

Dentro de esta última alternativa de fabricación utilizando micromáquinas herramienta y micromanipuladores, en 1996, en el *Mechanical Engineering Laboratory* (MEL) del *Nacional Institute of Advanced Industrial Science and Technology* (NIAIST) en Japón, se desarrolló un microtorno con dimensiones de 30x30x30 mm [18], mostrado en la figura 1.6. Mientras que un año más tarde, en *The Internacional Research and Training Center of* UNESCO/IIP *of Information Technologies and Systems, Nacional Academy of Science of Ukraine* se desarrolló un prototipo de microcentro de maquinado con dimensiones de 100x100x130 mm, ver figura 1.7 [19].

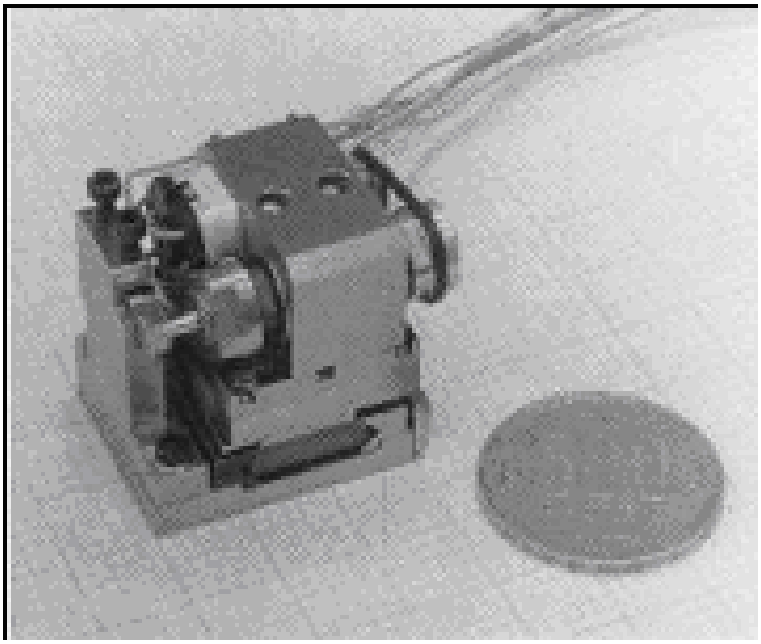

Figura 1.6. Microtorno japonés.

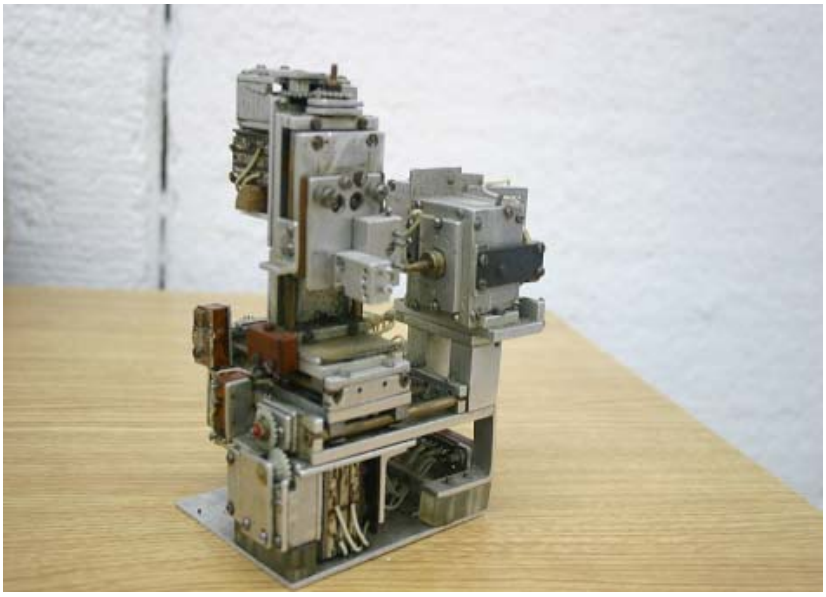

Figura 1.7. Microcentro de maquinado Ucraniano.

Posteriormente, en el MEL en Japón, se continuaron desarrollando prototipos de microequipo como: micromanipuladores, microfresadoras, microtornos, etc. (ver figura 1.8); los cuales forman parte del primer prototipo de microfábrica, el cual fue desarrollado por el MEL en 1999 y presentado en el *2nd International Workshop on Microfactories* en Suiza, en el 2000. Esta microfábrica, que se muestra en la figura 1.9, tiene dimensiones de 625x490x380 mm.

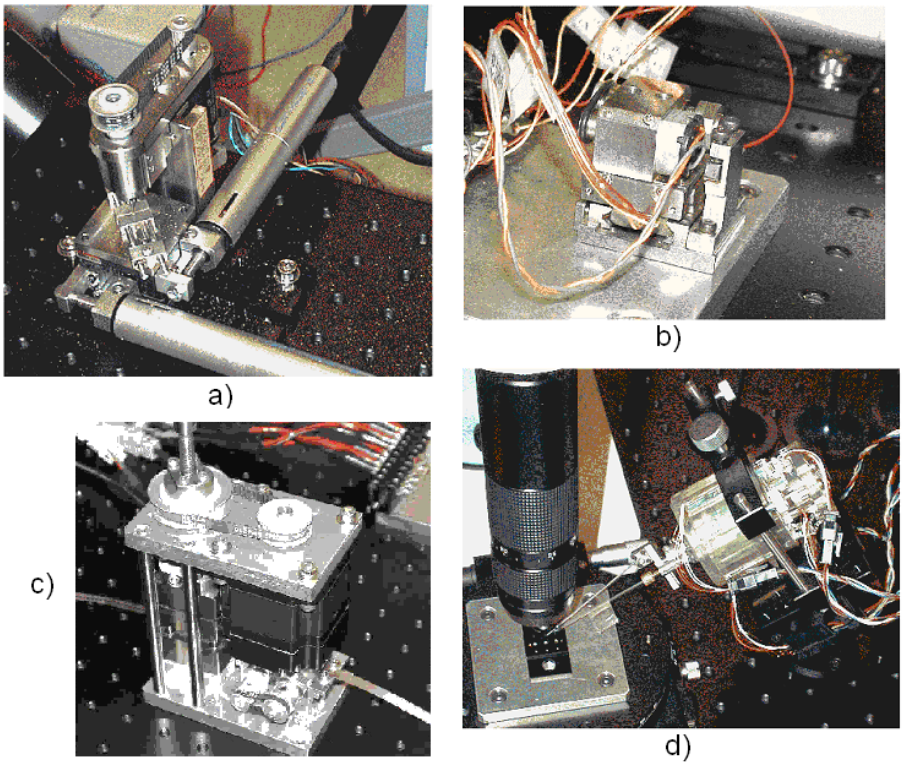

Figura 1.8. a) Microfresadora; b) Microtorno; c) Microprensa; d)Micromanipulador de dos dedos. Prototipos desarrollados por el MEL en Japón para formar parte de una microfábrica [20].

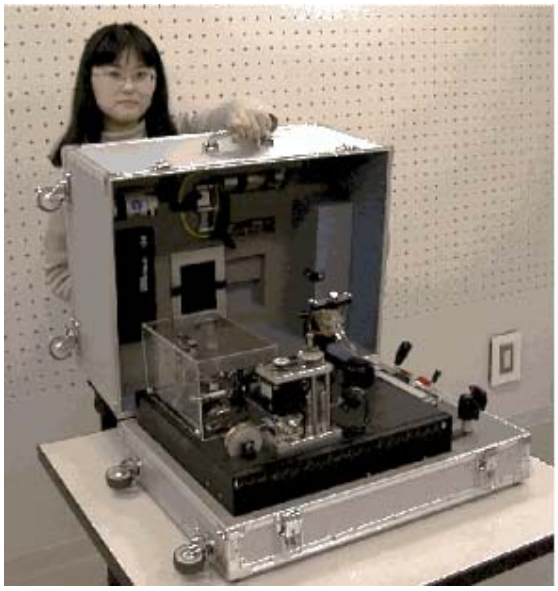

Figura 1.9. Microfábrica sobre mesa desarrollada por el MEL [20].

En el año 2004, la compañía NanoWave (Nano Corporation) presentó, durante el *Workshop on Microfactories* celebrado en Shanghai China, un torno ultra compacto de CNC comercial, con dimensiones, en la base, de 100x150 mm, una capacidad de 10 mm de longitud para torneado y piezas con diámetro hasta de 5 mm. Su eje rotacional opera a 10,000 rpm y tiene una resolución de 200 nm. El costo aproximado de este producto es de $35,000.00 usd. [21] . En la figura 1.10 se muestra este micro torno.

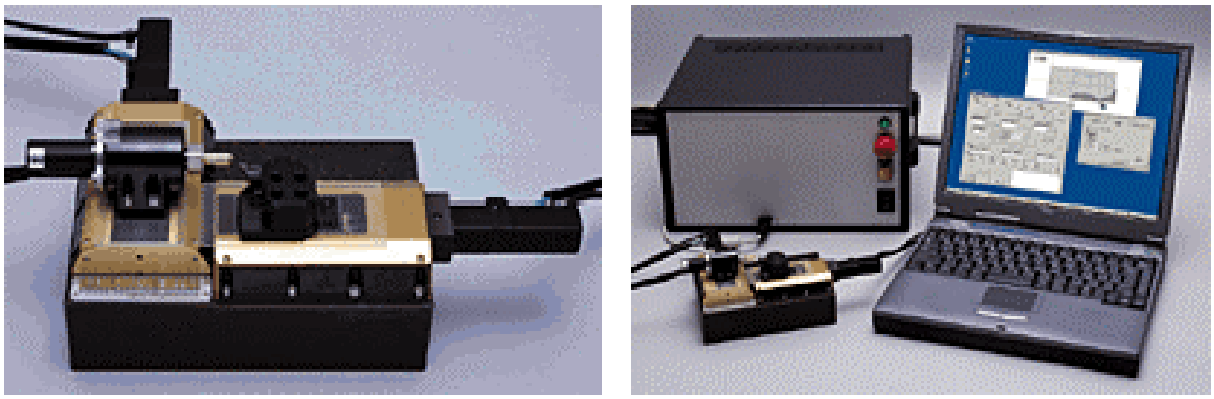

Figura 1.10. Microtorno CNC comercial modelo NANOWAVE MTS2.

1.4. Áreas de aplicación del microequipo.

El microequipo participa en un gran número de aplicaciones y mercados debido a las ventajas que presenta frente a sistemas de mayor tamaño, entre las cuales se encuentran: su tamaño y peso pequeño, consumo de energía reducido, alta precisión, posibilidad de fabricación masiva a bajo costo, entre otras. La variedad de productos existentes en la actualidad es muy grande, desde sensores de presión en un motor de automóvil hasta micropinzas para cirugía. Los principales sectores en los que se encuentran estos microsistemas son: el sector automotriz, aeroespacial y defensa, medioambiente y agricultura, medicina, telecomunicaciones, domótica, electrodomésticos, tecnologías de la información, entre otros. A continuación se mencionan algunas aplicaciones en estas áreas.

1.4.1. Sector automotriz.

Los primeros microsistemas en el sector automotriz fueron introducidos en los años 80. Desde entonces, han ido surgiendo constantemente nuevas aplicaciones potenciales en materia de gestión del motor, control de calidad del aire y gases de escape, cajas de cambio, frenos antibloqueo (ABS), control de la dinámica del vehículo y antideslizante (ASC), control de navegación adaptativo (ACC), “airbag”, detección de obstáculos y mejora de la visibilidad, etc.

Dentro de los microsistema frecuentemente utilizado en aplicaciones de seguridad para automóvil, se encuentran: el acelerómetro, el cual es un componente clave de los sistemas de “airbag”, siendo el responsable de su activación al detectar un choque; sensores magnéticos que registran la velocidad; sensores de presión, con el fin de medir la presión en los neumáticos; etc. Mientras que para mejorar el confort de los ocupantes del vehículo, podemos encontrar: sensores de humedad, sistemas de control de temperatura, microsistemas para el filtrado del aire, entre otros.

1.4.2. Sector aeroespacial y defensa.

El sector aeronáutico se caracteriza por su necesidad de instrumentación y sistemas de medida precisos. Por ello, encontramos una gran variedad de microsistemas, por ejemplo: acelerómetros, diversos sensores (de presión, humedad, temperatura), etc. Mientras que en el sector aeroespacial, la microtecnología es la clave en la fabricación de sistemas robustos, efectivos y miniaturizados para la exploración del espacio. Los problemas más comunes que se presentan son debidos a las duras condiciones del entorno de trabajo, por lo que se requieren microsistemas capaces de soportar estas condiciones, como son por ejemplo los microrobots. Otros microsistemas que encontramos en esta área son: microinstrumentos y microelectrónica para misiones espaciales, microhigrómetros, micromotores de propulsión, fotodetectores de infrarrojo, microsismómetros, etc.

El microsismómetro que se muestra en la figura 1.11 [17], fue diseñado y fabricado para utilizarse en Marte. Este sismómetro forma parte de un conjunto de sismómetros y acelerómetros para estudios planetarios y de microgravedad desarrollados por CSMT (*Center For Science, Mathematics and Technology*) para la NASA.

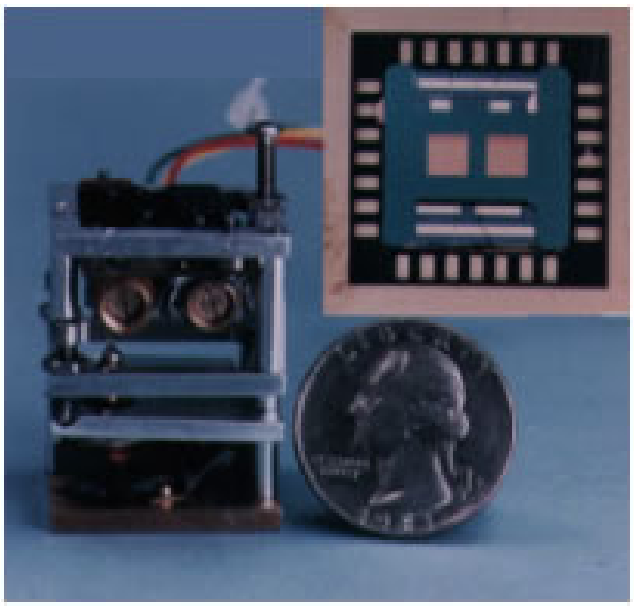

Figura 1.11. Microsismómetro desarrollado para su utilización en Marte.

1.4.3. Medioambiente y agricultura.

Entre los sistemas basados en microtecnologías aplicables a este sector, se pueden mencionar equipos de metrología, sensores químicos para medir los niveles de contaminación, monitorización medioambiental, o microsistemas para mediciones metereológicas.

En la figura 1.12 [17] puede verse un higrómetro desarrollado por el CSMT en Estados Unidos, el cual fue diseñado para formar parte de una microestación meteorológica. Dicha estación sirve para realizar mediciones en las capas superiores de la atmósfera (este tipo de estaciones son llamadas globosondas). En esta microestación también encontramos sensores de silicio micromecanizado para medición de viento, presión y temperatura, un sensor de densidad de radiación y un microanemómetro de efecto Doppler láser, capaz de medir la velocidad del viento de forma remota.

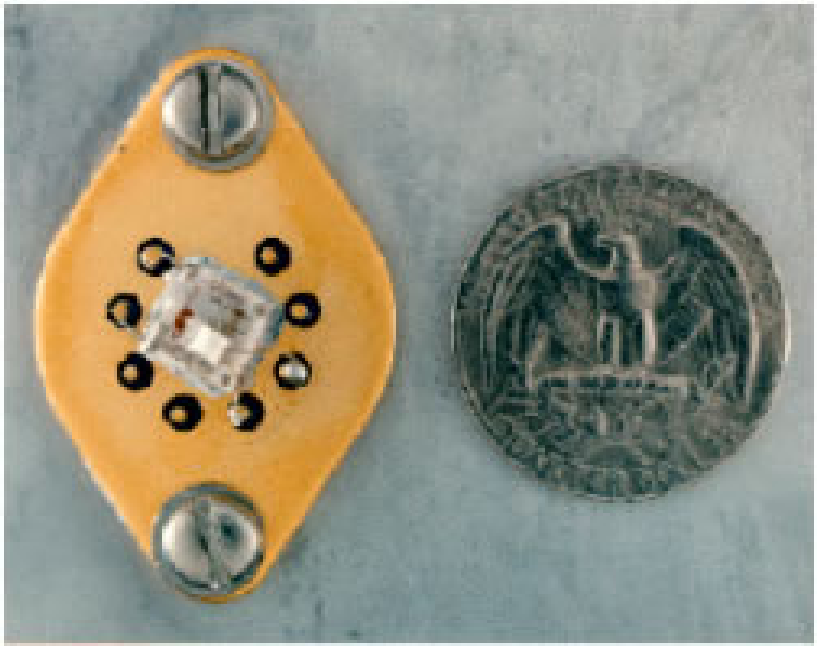

Figura 1.12. Higrómetro para una microestación meteorológica

Para el control de pureza de alimentos podemos encontrar: microsensores, intrumentación que integra espectrómetros de infrarrojo, dispositivos de análisis de color, etc.

1.4.4. Medicina.

Gracias a los microsistemas, las nuevas técnicas en este sector tienden a ser menos invasivas, más baratas y con una mayor rapidez de diagnóstico. En esta área encontramos microsistemas que permiten obtener información del cuerpo humano, entre los que están: microsensores de visión, medidores de glucosa, de presión sanguínea, de oxígeno en sangre, endoscopios, sistemas de diagnosis genética, etc. Los sistemas de medición de glucosa en la sangre, por ejemplo, son capaces de trabajar con muestras de tan sólo 3 microlitros, integrando microcapilares y microelectrodos (en el caso de sensores electroquímicos) [17].

Gracias a la microtecnología se logran importantes mejoras en la cirugía mínimamente invasiva, en la que podemos encontrar dispositivos e instrumentos miniaturizados que

incorporan microestructuras mecánicas, componentes microópticos y de fibra óptica, microsensores táctiles y de presión, etc. Por ejemplo, los motores lineales utilizados en microcirugía controlan la posición de instrumentos quirúrgicos a muy pequeña escala (del orden de la micra) mejorando los resultados de la mayoría de las técnicas quirúrgicas. Así también, los sensores de tejidos activos con microsensores de fuerza incorporados, permiten identificar en tiempo real las propiedades de los tejidos de forma que el cirujano sólo manipule los tejidos insanos. Otros sistemas similares, aunque menos frecuentes, son los microsistemas de estimulación de nervios y músculos, los cuales se integran en órganos artificiales [17].

1.4.5. Telecomunicaciones.

Dentro de las telecomunicaciones, la microtecnología desempeña un papel fundamental. El intercambio de información aumenta de forma exponencial exigiendo la aparición de nuevas ofertas y el aumento de capacidad de las autopistas de la información. La rápida expansión de los sistemas de telecomunicación basados en fibra óptica han provocado un espectacular crecimiento tanto en el número como en el tipo de fabricantes de dispositivos y componentes ópticos. Inicialmente, tales fabricantes contaban con una ingeniería muy costosa para la producción de conectores de fibra óptica. Sin embargo, el uso de microcomponentes permite disminuir el costo de estas redes.

1.4.6. Domótica.

Los microsistemas incorporados en aplicaciones de domótica pueden ser tan simples como detectores de fugas de gas, o sensores de presión con función de autoencendido similares a los utilizados en balanzas. Sin embargo, los sistemas multifuncionales que se desarrollan hoy en día están enfocados a aplicaciones más complejas como: control de olores en cocina, análisis de contaminación, medición y ajuste de temperatura, luminosidad y humedad, cámaras de seguridad, llaves inteligentes, etc.

1.4.7. Tecnologías de la información.

El mercado de tecnologías de la información es el que mayor volumen de microsistemas integra. Numerosas microestructuras con funciones mecánicas, magnéticas y ópticas, entre otras, forman parte de los dispositivos de almacenamiento de datos (discos duros, lectoras de CD-ROM…) y de los periféricos de entrada y salida de datos (monitores, escáners, lectoras de código de barras, “displays” de pantalla plana, etc.). Los cabezales magnéticos de discos duros y ópticos para CD-ROMs, junto con los cabezales de inyección de tinta para impresoras y los “displays”, constituyen las principales aplicaciones de los microsistemas dentro de este sector.

1.5. Micromecánica en el CCADET.

Debido a las numerosas aplicaciones y al importante papel que desempeñan los microsistemas; en el Laboratorio de Micromecánica y Mecatrónica (LMM) del Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la UNAM, desde 1999 se sigue una línea de investigación en tecnología para la fabricación de piezas micromecánicas, la cual se basa en el uso de micromáquinas herramienta y dispositivos de microensamble. El objetivo que se persigue dentro de esta línea de investigación es desarrollar micromáquinas herramienta y dispositivos de microensamble a través de varias generaciones con el fin de conseguir un menor tamaño. De esta manera, se propone desarrollar la primera generación de microequipo, con dimensiones aproximadas de unas decenas de centímetros, mediante equipo de escala convencional. Mientras que las generaciones subsecuentes, con dimensiones cada vez menores, se fabricarán con

equipo de las generaciones previas, lo anterior con el fin de abatir el costo que implica llegar de un sólo paso al desarrollo de micromáquinas herramienta con dimensiones milimétricas [2].

En el año 2000, el LMM construyó un prototipo de micromáquina herramienta, muy similar al prototipo desarrollado en Ucrania en 1997 por el Dr. Ernst Kussul (quién cambió su residencia a México y labora actualmente en el LMM). Este prototipo, desarrollado por el LMM, presenta mejoras en cuanto a rigidez y en el uso del modo dinámico para el control de motores de pasos [2]. Este prototipo de micromáquina herramienta presenta un control que hace uso de las capacidades de una computadora personal (PC), la cual se comunica con la micromáquina herramienta a través del puerto paralelo. A continuación se describe este prototipo.

1.5.1. Descripción del prototipo de microcentro de maquinado.

Este prototipo de micromáquina herramienta también es denominado como microcentro de maquinado debido a que realiza funciones de torneado, fresado y taladrado. El tipo de trabajo depende de la herramienta empleada y de la configuración en la que trabaje el equipo. Sus dimensiones totales son de 130 x 160 x 85 mm [9].

Como se puede observar en las figuras (1.13, 1.14 y 1.15), sobre el bastidor (1) están montadas tres guías (2), (4), (6), mismas que soportan tres carros (3), (5), (7) mediante un esquema serial, es decir, cada guía es instalada sobre el carro previo para obtener con esto movimientos de translación en los ejes “X”, “Y” y “Z”. Cuenta también con un sistema de sujeción de piezas (10) que está acoplado a una caja de reducción de engranes y que también son instalados sobre la base. Los actuadores empleados para proporcionar movimiento, tanto en los carros como en el sistema de sujeción son motores de pasos (8) auxiliados mediante cajas de engranes con una relación de 84.7:1. En el último carro se encuentra montado un sistema porta herramientas para realizar trabajos de torneado; para el caso de fresado y taladrado se coloca un sistema de sujeción especial en lugar del portaherramientas (12).

El volumen de trabajo obtenido con este microcentro de maquinado es: 20 mm de desplazamiento en los ejes X y Z, mientras que el eje Y tiene 35 mm de desplazamiento.

La resolución manejada con el conjunto de transmisión y motor de cuatro pasos (similar para los tres ejes translacionales), es de 1.88 μm.

El diseño detallado de este prototipo es mostrado en la tesis “*Diseño y construcción de un microcentro de bajo costo*” [9].

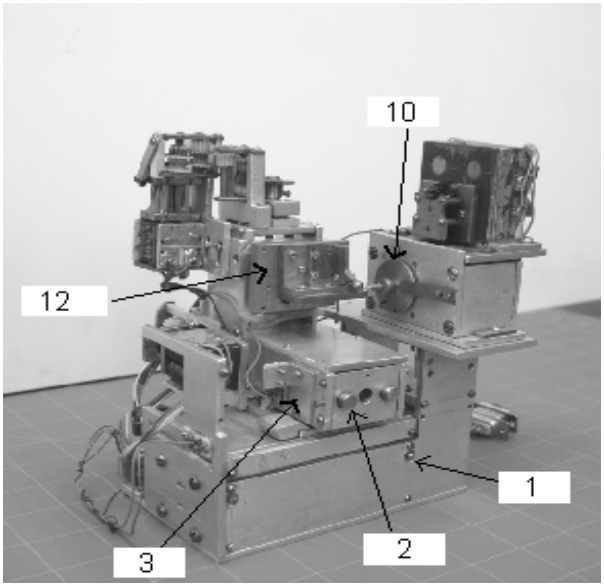

Figura 1.13. Vista frontal del microcentro de maquinado desarrollado en el LMM del CCADET, UNAM.

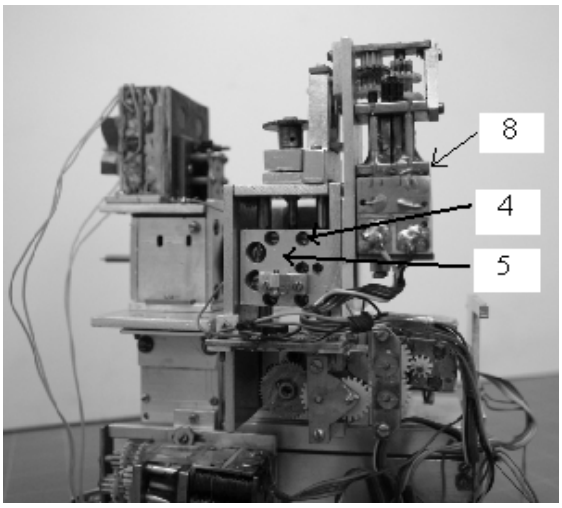

Figura 1.14. Vista posterior del microcentro de maquinado desarrollado en el LMM del CCADET, UNAM.

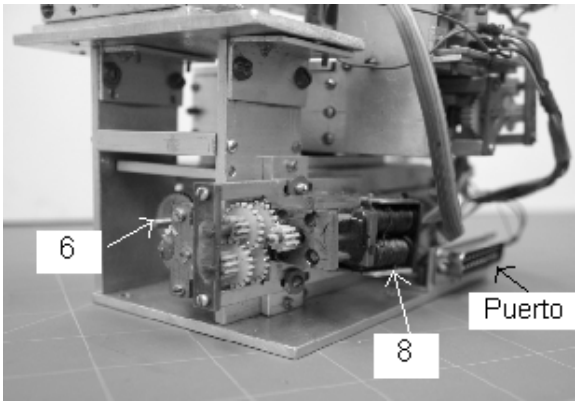

Figura 1.15. Vista posterior del eje Y del microcentro de maquinado desarrollado en el LMM del CCADET, UNAM.

Algunas micropiezas producidas con este microcentro de maquinado se muestran en la figura 1.16.

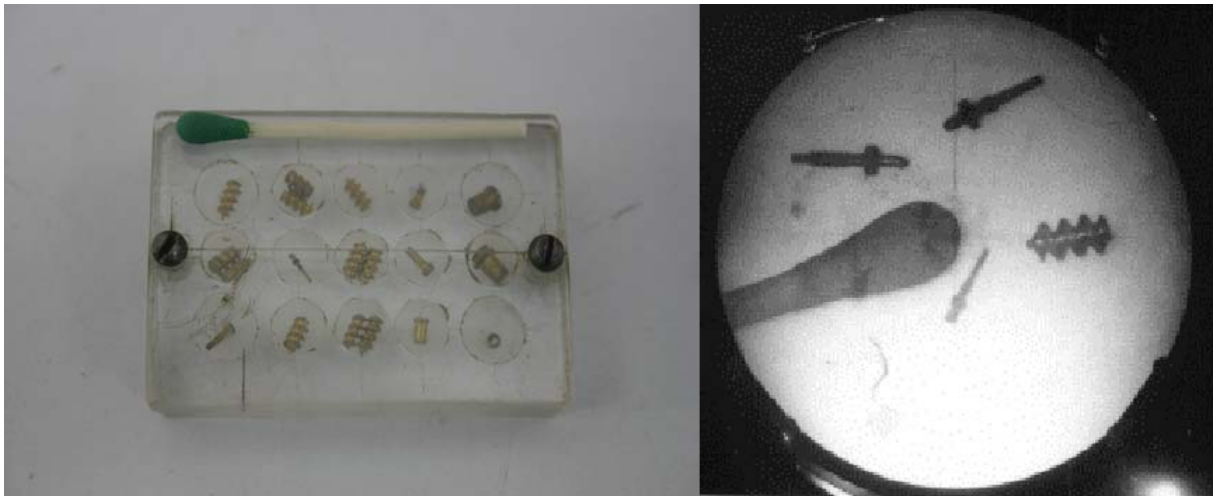

Figura 1.16. Micropiezas producidas con el microcentro de maquinado del LMM.

Para producir cada una de esta piezas es necesario cambiar las herramientas de corte de forma manual y se tiene que programar un conjunto de instrucciones en lenguaje C++. Cada pieza requiere de un programa diferente, el cual se ejecuta en una PC, y como resultado se obtienen señales de control que son enviadas, a través de dos puertos paralelos de la PC, hacia la MMH y poder así manufacturar una micropieza. Estas señales de control pasan primero por una etapa de amplificación de la señal donde se consigue una ganancia en corriente. Las salidas de esta etapa son conectadas a cuatro motores de pasos. Tres de estos motores son los encargados de posicionar la herramienta de corte en los tres ejes (X, Y, Z); y el cuarto motor se utiliza para dar movimiento al husillo de la micromáquina herramienta [2].

El presente trabajo tiene la intención de apoyar esta línea de investigación que se sigue en el LMM, buscando mejorar el proceso de manufactura de micropiezas mediante una interfaz capaz de interpretar un código numérico estándar y acercarse así a lo que sería un microcentro de maquinado CNC.

En el siguiente capítulo se plantea la problemática a resolver, el objetivo del trabajo, y la metodología a seguir para conseguir dicho objetivo.

Capítulo 2

OBJETIVO Y METODOLOGÍA

2.1. Definición del problema.

Actualmente, para llevar a cabo la manufactura de una cierta pieza empleando una micromáquina herramienta (MMH) se tiene que desarrollar un programa diferente por cada una de estas piezas. Este programa se realiza empleando rutinas que efectúan movimientos básicos de la MMH, desarrolladas en lenguaje C++. Sin embargo, este procedimiento resulta tardado y poco conveniente si lo que se busca es diversidad y facilidad en la producción de piezas. Así también, el tamaño y la complejidad del programa aumenta en relación a la complejidad de la geometría de la pieza que se desea manufacturar. Por tal razón, se requiere del diseño y desarrollo de una aplicación capaz de interpretar instrucciones de código G y M , convirtiéndolas en señales de control para micromáquinas herramienta que permitan la manufactura de piezas. Este tipo de instrucciones de código G y M forman parte de un estándar en el control numérico por computadora y pueden ser generadas de forma automática a partir de un software de manufactura asistida por computadora (CAM), con lo que se facilitará la manufactura de geometrías complejas y permitirá maximizar las capacidades de producción de estos equipos.

2.2. Objetivo.

Desarrollar una aplicación que sea capaz de interpretar un conjunto de instrucciones de código G y M, y generar las señales de control necesarias para la manufactura de piezas en una micromáquina herramienta.

2.3. Metodología.

Los siguientes pasos conforman el desarrollo del proyecto:

Análisis

1. Recopilar información sobre sistemas de control numérico aplicado a máquinas herramienta y entender el funcionamiento de sus partes.
2. Estudiar la sintaxis del código G y M.
3. Estudiar los antecedentes y parte del funcionamiento de las micromáquinas herramienta desarrolladas en el LMM.
4. Observar la forma en cómo se realiza el diseño de una pieza en dos dimensiones en un software CAD para ser exportada a un software CAM.
5. Observar el proceso de manufactura en dos máquinas CNC importando el archivo generado en el software CAD para familiarizarse con el software y conocer las características que presentan, así como también, conocer la estructura del código G y M que generan.
6. Identificar las instrucciones en código G y M que serán soportadas por la aplicación con base en los procesos de maquinado que podrán realizarse y a las características de la micromáquina herramienta.
7. Escoger un lenguaje de programación.
8. Definir el puerto de comunicación de la PC a ocupar.

Diseño.

9. Desarrollar una propuesta de la aplicación con base en los requerimientos básicos.

Implementación.

10. Desarrollar el procedimiento para leer un archivo de texto que contenga un conjunto de instrucciones en código G y M.
11. Desarrollar las rutinas que decodifiquen cada una de las sentencias del archivo leído.
12. Simular gráficamente dentro de la interfaz el proceso de manufactura, implementando el funcionamiento de cada instrucción.
13. Ir añadiendo mayor funcionalidad a la interfaz considerando las características de los software's CNC que se tienen en el LMM.
14. Hacer pruebas de comunicación entre la PC y los motores a pasos de la micromáquina herramienta y lograr el control necesario en función de los parámetros de posición y velocidad de avance, mediante la implementación de rutinas para los movimientos básicos.
15. Implementar funciones correspondientes a instrucciones en código G y M con base en las rutinas realizadas en el punto anterior.
16. Corregir errores en el maquinado de las instrucciones y en la ejecución de los procedimientos.
17. Hacer que trabajen de forma conjunta las rutinas para el maquinado y la simulación gráfica de los procesos maquinados.
18. Añadir otras características de funcionalidad a la interfaz.

Pruebas.

19. Realizar pruebas de procesos de manufactura, empezando con piezas cuyos maquinados se efectúen mediante el movimiento de un eje a la vez; y pasar después, a maquinados que realicen interpolación en dos ejes.

20. Realizar pruebas de procesos de manufactura que involucren un mayor número de instrucciones de código numérico, las cuales hayan sido generadas a partir de un software CAD y CAM, e interpretadas por la interfaz que controla a la MMH, para obtener finalmente la pieza deseada.

Evaluación de los resultados.

21. Tomar medidas a los maquinados realizados y verificar las piezas obtenidas.
22. Evaluar los resultados obtenidos.

Trabajo futuro.

23. Proponer mejoras al sistema.

Capítulo 3

ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN

El ciclo de vida de los sistemas (SLC, *system life cycle*) es el proceso evolutivo que se sigue al implementar un sistema o subsistema de información basado en computadora. Este SLC consta de cinco fases. Las primeras cuatro fases son las de planificación, análisis, diseño e implementación; que en conjunto constituyen el ciclo de vida del desarrollo de sistemas (SDLC, *system development life cycle*). La quinta fase es la fase de uso que dura hasta que llega el momento de rediseñar el sistema. Para el rediseño del sistema se requiere de la repetición de todo el ciclo [15].

En este capítulo se describen tres de las fases del ciclo de vida del sistema a desarrollar, que son: análisis, diseño e implementación.

Los elementos que se toman en cuenta para la fase de análisis son: el sistema de control de la MMH desarrollada en el LMM, el proceso de manufactura de una pieza con dicho sistema, y las características de funcionalidad de un software CAM dentro de un sistema CNC. Con base en estos elementos se establecen los requerimientos de la aplicación a implementar.

En la fase de diseño se presentan las alternativas que se tienen para el desarrollo del proyecto, por ejemplo: la herramienta para el desarrollo del sistema y el puerto de comunicación a emplear.

Finalmente, la última etapa consiste en la implementación donde se describe la estructura y funcionamiento del sistema.

3.1. Análisis.

3.1.1. Sistema de control de la MMH desarrollada en el LMM.

El sistema de control para la MMH, desarrollado en el LMM, se realiza empleando una computadora personal que envía señales de control a través de dos puertos

paralelos. Estas señales de control pasan por un amplificador de DC de 16 canales antes de llegar finalmente a los actuadores de la MMH. Así también, la MMH cuenta con unos sensores de contacto en cada eje de movimiento (3 sensores) con el fin de detectar el momento en que se llega a la posición de inicio. Un cuarto sensor, de contacto eléctrico, es utilizado con el fin de determinar las posiciones relativas de diferentes instrumentos empleados en los procesos de manufactura. Las señales de dichos sensores son realimentadas a la PC, pasando primero por unos acondicionadores de voltaje TTL para poder ser leídas por el puerto paralelo. Estos elementos en conjunto constituyen el sistema de control para la MMH, el cual se muestra en la figura 3.1.

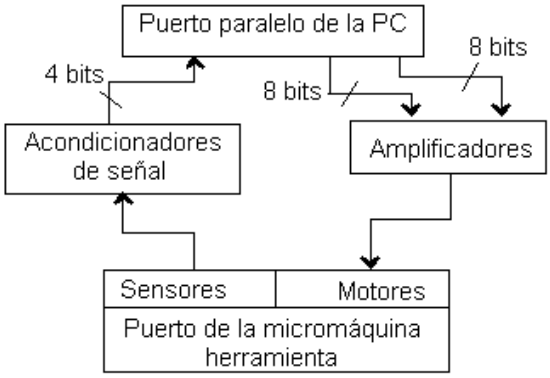

Figura 3.1. Sistema de control para la MMH desarrollada en el LMM.

En este sistema de control se hace uso de dos puertos paralelos de una PC. El primer puerto tiene asignado el control de los movimientos de translación del eje X y del eje Z. Mientras que el segundo puerto tiene asignado el movimiento de translación del eje Y, así como el movimiento de rotación del husillo y las señales de retroalimentación de los sensores eléctricos.

En cuanto al software de control, este fue desarrollado con la herramienta *Borland C++ Builder.* Esta aplicación tiene implementadas rutinas de control para movimientos predefinidos, como son: movimientos con desplazamientos fijos hacia delante y hacia atrás para cada uno de los ejes, regreso de los carros de cada eje a la posición de origen, así como rutinas para realizar mediciones. En la figura 3.2 se puede observar la pantalla principal de este programa de control. En la parte superior izquierda se aprecian los botones que ejecutan las rutinas para los movimientos de los tres ejes. Para cada eje de translación se tienen cuatro botones. El primero de ellos corresponde a una rutina que hace mover el carro, del eje en cuestión, 100 pasos hacia delante; el segundo botón, realiza el mismo procedimiento pero avanzando 1000 pasos; finalmente, el tercer y cuarto botón, tienen la misma función que los dos anteriores, pero realizan el movimiento en sentido contrario. Para el eje de rotación se cuenta también con cuatro botones que ejecutan movimientos de rotación horario o antihorario de 100 o 1000 pasos. Existe también, un botón llamado START que regresa los tres carros, de cada uno de los ejes, a la posición de origen. Se pueden observar también otros botones

que sirven para manufacturar alguna pieza o para realizar mediciones con el fin de caracterizar la MMH.

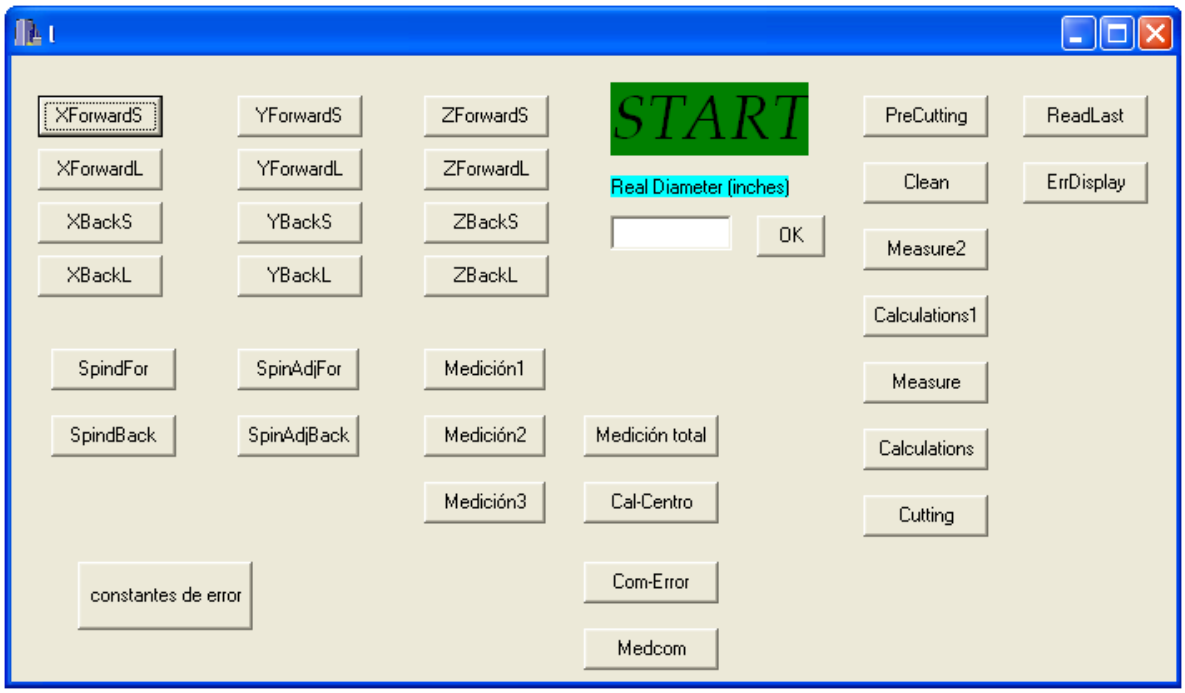

Figura 3.2. Pantalla principal del programa de control actual.

3.1.2. Manufactura de una pieza en la MMH.

Antes del desarrollo de este proyecto, la manufactura de piezas era realizada por medio de un programa en el que se toman como base las rutinas de movimientos lineales de los tres ejes y del movimiento de rotación del husillo, desarrolladas con anterioridad. Empleando estas rutinas es posible implementar nuevos procedimientos para obtener movimientos de dos ejes simultáneamente, y así generar piezas de acuerdo a la geometría deseada.

Un ejemplo de una pieza manufacturada mediante este sistema es un microtornillo[10], el cual presenta las siguientes características:

- *Diámetro*: ϕ = 0.8 mm.
- *Longitud*: l = 2.22 mm.
- *Paso* = 0.2 mm.
- *Material*: barras de latón de ϕ = 3.2 mm o 1/8 de pulgada.

Para generar el programa para la manufactura de esta pieza, se deben de tomar en cuenta las características de funcionamiento de la MMH. Dichas características son las siguientes [9]:

- *Desplazamiento* lineal de 1.88 μm por paso de motor en cualquiera de sus tres ejes.

- *Desplazamiento angular* del husillo: 10° por paso del motor.
- *Backlash* de 345.27 μm.

Con base en estas características de funcionamiento de la MMH y de las características de la pieza a manufacturar, se programan las rutinas para el maquinado. Estas rutinas se ejecutan como parte de todo un proceso de manufactura, el cual se resume en los siguientes pasos:

1. Colocar el portaherramientas y el material de trabajo en la micromáquina herramienta.
2. Determinar el punto de origen del maquinado, situando la herramienta de corte.
3. Maquinar el primer diámetro del microtornillo (ϕ = 1.2 mm).
4. Maquinar el segundo diámetro del microtornillo (ϕ = 0.8 mm).
5. Maquinar la cuerda del microtornillo.

En el tercero y cuarto paso se ejecutan algunas de las rutinas programadas para el maquinado. Para estas rutinas fue necesario calcular el número de pasos del motor que se ejecutan dentro de un ciclo y el número de veces que debe ejecutarse este ciclo para alcanzar el diámetro requerido en cada caso. La velocidad con que se mueve la herramienta de corte, para estos dos casos, es constante. En la figura 3.3 se muestra una imagen del proceso de manufactura de este microtornillo.

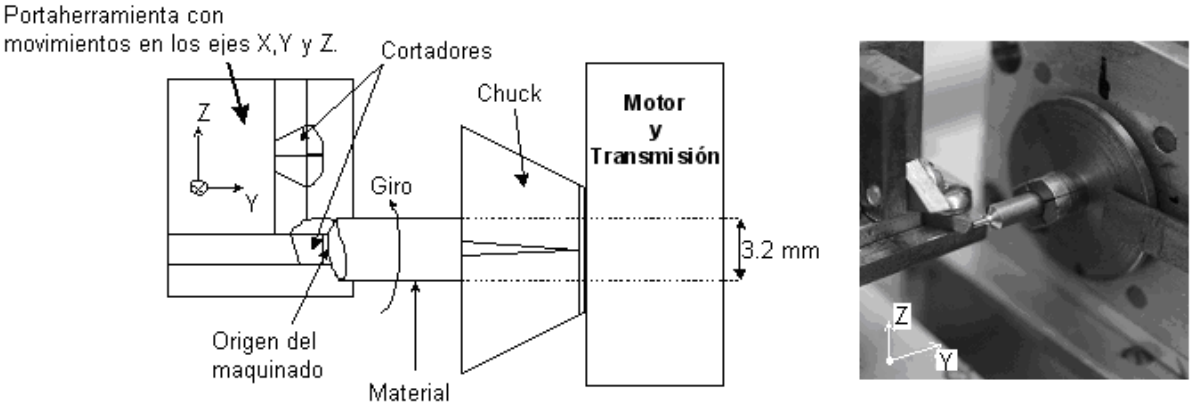

Figura 3.3. Manufactura de un microtornillo en la MMH.

Para el maquinado de la cuerda de este microtornillo, es necesario calcular la relación de velocidad entre el eje Y y el eje de rotación. Resultando así la relación de velocidad del rotor 3:1 respecto a la velocidad lineal, para conseguir el paso del tornillo deseado. En la figura 3.4 se muestra la cuerda obtenida.

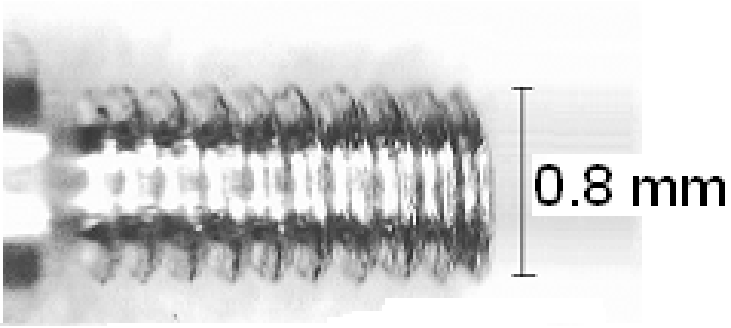

Figura 3.4. Cuerda del microtornillo.

De esta forma es como se realiza la manufactura de piezas. El tiempo empleado en la programación de este microtornillo fue aproximadamente de dos semanas, considerando el tiempo de aprendizaje del proceso de manufactura con este sistema de control; mientras que el tiempo requerido para manufacturar este microtornillo fue de 25 minutos [10]. Como puede verse el tiempo empleado para la creación de los programas es largo. Ahora bien, si se quisiera manufacturar un microtornillo con diferentes características, se tendría que hacer un nuevo programa y realizar los cálculos pertinentes. En este sentido, con la aplicación a desarrollar se busca abrir una brecha en el desarrollo de aplicaciones que en un futuro alcancen una funcionalidad semejante a la que presentan los softwares CAD/CAM. En el siguiente apartado se describe uno de estos softwares.

3.1.3. Características de un software CAM.

En este apartado se describe la aplicación *FLASHCUT™ CNC versión 1.4*, ya que se cuenta con ella dentro del LMM. Esta aplicación, para Windows®, funciona como un sistema de control CNC que se puede ocupar con diferentes máquinas herramienta, pero que particularmente se ocupa con la máquina fresadora marca *Sherline CNC*.

Las principales características de *FLASHCUT™ CNC* son[11]:

- Compatibilidad del código G y M tanto con fresadoras como con tornos.
- Capacidad para importar archivos DXF y generar el código G y M correspondiente a dicho archivo.
- Generación de gráficos en tiempo real del proceso de manufactura que se esté realizando.
- Diferentes modos de control (mediante código G y M, joystick, y posicionamiento en algún punto definido).
- Capacidad de controlar hasta 4 ejes de movimiento.
- Envío de señales vía puerto serie.

La pantalla principal de esta aplicación se puede ver en la figura 3.5.

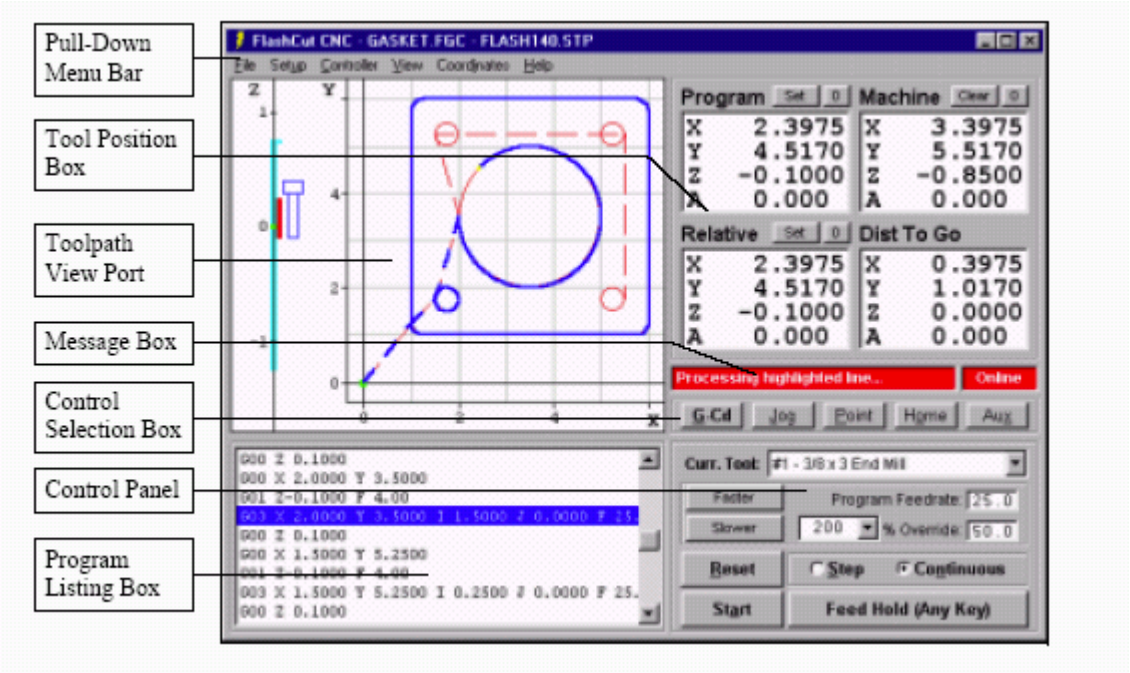

Figura 3.5. Pantalla principal de FLASHCUT™ CNC.

En cuanto a los modos de control que presenta *FLASHCUT™ CNC* están:

a) Modo de control mediante código G y M. En este modo el software manipula la máquina herramienta con base en las instrucciones dadas. Este código G y M puede ser programado de forma manual por el usuario o puede ser obtenido de forma automática al importar un archivo DXF. La ejecución de este código puede realizarse paso por paso (instrucción por instrucción) o el código completo (todas las instrucciones).

b) Modo joystick. Este modo de control permite posicionar la herramienta en cualquiera de sus ejes con sólo mantener presionado el botón o tecla en la dirección que se desea mover la herramienta. Este modo de control ofrece diferentes niveles de velocidad para realizar estos movimientos.

c) Posicionamiento hacia algún punto definido. Este modo de control sirve para posicionar la herramienta en algún punto como puede ser: el cero del programa, el cero máquina, el cero relativo, entre otros.

Cabe mencionar que el tiempo para producir una pieza en una MHCNC es considerablemente menor en comparación con el tiempo que se requiere en fabricar esa misma pieza en la MMH. Por ejemplo, un microtornillo de dimensiones muy cercanas al que se mencionó anteriormente, en el apartado 3.1.2, requiere de un tiempo aproximado de 30 minutos para la programación de esa micropieza y un minuto catorce segundos para la manufactura de la misma, acupando un torno CNC marca Boxford que se tiene dentro del LMM [10].

3.1.4. Requerimientos básicos para la aplicación a desarrollar.

Con base en los elementos analizados se puede ver que para la manufactura de piezas utilizando el sistema de control actual, se deben de tomar las rutinas ya programadas de movimientos básicos y crear los procedimientos adecuados para la manufactura de esas piezas. Una alternativa para la solución de este problema es crear una aplicación que interprete instrucciones de código G y M y genere las señales de control para la MMH, consiguiendo así una mayor versatilidad en la manufactura de piezas.

Los requerimientos básicos para la aplicación a desarrollar son los siguientes:

- Ambiente gráfico.
- Compatibilidad de código G y M con fresadora.
- Interpretar de forma gráfica instrucciones de un programa pieza en código G y M.
- Modo de control mediante código G y M.
- Modo de control por joystick.
- Control de 3 ejes de movimiento.
- Envío de señales vía puerto serie o paralelo.

3.2. Diseño.

3.2.1. Sistema de control propuesto y alternativas para los elementos que conforman el sistema.

El diagrama de bloques del sistema propuesto para el control de la MMH, es el que se muestra en la figura 3.6. Este sistema de control esta formado por tres elementos esenciales que son: una PC que ejecuta un software de control, un puerto de comunicación y hardware adicional. Para estos elementos se hace una valoración de qué herramienta de desarrollo ocupar para el software de control, qué vía de comunicación emplear y qué elementos de hardware son requeridos.

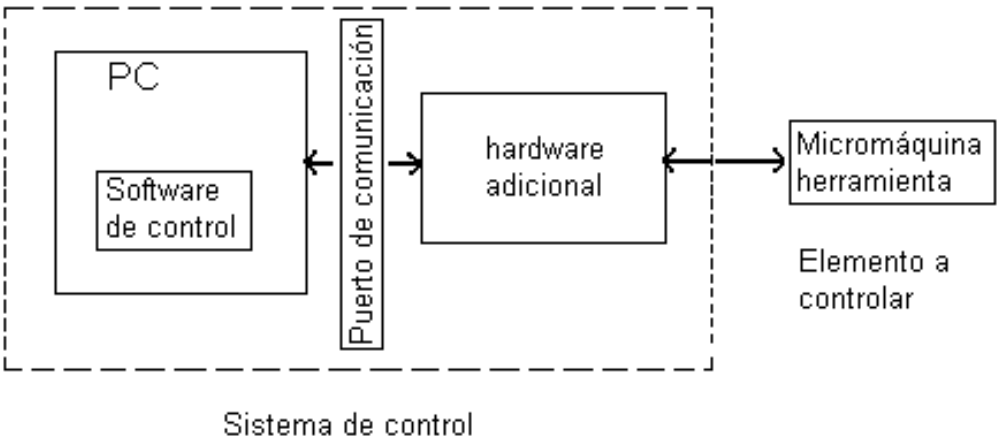

Figura 3.6. Diagrama de bloques del sistema de control propuesto.

En la tabla 3.1 se presentan alternativas para la configuración de este sistema.

Elementos del sistema	Alternativas
Software de control	Herramientas de desarrollo: ▪ *Borland C++* ▪ Java ▪ *Visual Basic 6.0*
Puerto de comunicación	▪ Puerto serie ▪ Puerto paralelo ▪ Puerto USB
Hardware adicional	1.- Etapa de potencia y de acoplamiento de señales. 2.- Dispositivos de control (junto con una etapa de potencia): ▪ Procesador digital de señales (DSP) ▪ Arreglo de compuertas programables de campo (FPGA) ▪ Controlador de interfaz periférico (PIC)

Tabla 3.1. Opciones posibles para la configuración del sistema.

3.2.2. Evaluación de las alternativas para los elementos del sistema.

SOFTWARE DE CONTROL

Para el desarrollo del software de control, las tres alternativas propuestas cubren con los requerimientos establecidos. A cada característica se le asignó un valor conforme a su importancia para cubrir los objetivos de este proyecto. En la tabla 3.2 se muestran los valores asignados a estas características; donde el valor de 1 significa “no importante”, 2 “importante” y 3 “muy importante”. Para evaluar cada una de las herramientas de desarrollo de software se califica en escala de 1 a 10 cada una de las características y se multiplica la calificación otorgada por el valor que tiene asignada dicha característica. La suma de estos productos da una cifra final. La cifra más alta determina la opción a escoger. La evaluación de estas alternativas, tomando en cuenta sus características, se muestra en la tabla 3.3.

Característica a evaluar	Valor asignado
Curva de aprendizaje rápido de la herramienta.	2
Facilidad para generar interfases gráficas.	2
Eficiencia en el manejo de puertos de comunicación.	3
Menor tiempo requerido para el desarrollo de la aplicación.	3
Conocimiento de esta herramienta.	3

Tabla 3.2. Valores asignados a las características para evaluar la herramienta de desarrollo del software.

Herramienta	Curva de aprendizaje rápido de la herramienta (2)	Facilidad para generar interfases gráficas (2)	Eficiencia en el manejo de puertos de comunicación (3)	Menor tiempo requerido para el desarrollo de la aplicación (3)	Conocimiento de esta herramienta (3)	Total
Borland C++	8	9	9	9	4	100
Java	7	5	8	7	8	93
Visual Basic 6.0	9	9	8	9	9	114

Tabla 3.3. Evaluación de las herramientas para el desarrollo de la aplicación.

PUERTO DE COMUNICACIÓN

Para las opciones que se tienen en cuanto al puerto de comunicación a emplear, se asignaron valores a algunas de las características que presentan estos puertos. Los valores asignados a estas características, de acuerdo a su importancia para cubrir los objetivos de este proyecto, son: 1 significa “no importante”, 2 “poco importante”, 3 “importante” y 4 “muy importante”. En la tabla 3.4 se muestran estas características con sus valores asignados. La evaluación de dichas características, para cada uno de los puertos de comunicación, se muestra en la tabla 3.5. Dicha evaluación consiste en otorgar una calificación en escala de 1 a 10 para cada característica. Esta calificación otorgada se multiplica por el valor que tiene asignado cada característica. La suma de estos productos da una cifra final. La cifra más alta determina la opción a escoger.

Característica a evaluar	Valor asignado
Velocidad de transmisión.	3
Longitud máxima del cable, soportada para la comunicación.	1
Soportado por las herramientas de desarrollo de aplicaciones.	4
Facilidad para controlarlo.	3
Menos hardware adicional requerido.	4

Tabla 3.4. Valores asignados a las características para evaluar el puerto de comunicación.

Puerto de comunicación	Velocidad de transmisión (3)	Longitud máxima del cable, soportada para la comunicación (1)	Soportado por las herramientas de desarrollo de aplicaciones (4)	Facilidad para controlarlo (3)	Menos hardware adicional requerido (4)	Total
Puerto serie	7	7	9	8	7	116
Puerto paralelo	8	6	7	9	9	121
Puerto USB	9	8	5	6	7	101

Tabla 3.5. Evaluación de los puertos de comunicación.

Los valores que se tomaron en cuenta para evaluar las características de velocidad de transmisión y de longitud máxima de cable que soportan estos puertos, se muestra en la tabla 3.6.

Puerto de comunicación	Versión	Velocidad máxima de transmisión	Longitud máxima de cable
Serie	UART 8250	0.1152 Mbps	1 metro
Paralelo	En modo estándar "Centronics"	0.92 Mbps	0.45 metros
USB	1.1	12 Mbps	5 metros
USB	2.0	480 Mbps	5 metros

Tabla 3.6. Características de los puertos de comunicación.

HARDWARE ADICIONAL.

Alternativa 1: Utilizar etapa de potencia y de acoplamiento de señales.

Para el hardware adicional se presentan dos alternativas. Una de ellas es utilizar un hardware similar al que utiliza el sistema de control actual, el cual se mencionó anteriormente -en la parte de análisis-. Esta opción presenta las siguientes ventajas y desventajas:

- Ventajas:

  1.- La electrónica se simplifica al emplear la menor cantidad de hardware.

  2.- El control se centraliza en el software y la complejidad de los movimientos que pueda realizar la MMH dependen sólo de él.

- Desventajas:

  1.- La complejidad del software de control aumenta, debido a que el software debe generar todas las señales para el control de los actuadores.

  2.- Para no acrecentar la complejidad del software, el control sobre los actuadores se limita a ser en lazo abierto.

  3.- El control de la MMH se ve afectado por retrasos que provoca el sistema operativo durante la gestión de procesos.

  4.- La ejecución de otros procesos en la PC, que no estén destinados al control de la MMH debe evitarse; con el fin de reducir los retrasos en las señales de control.

  5.- Se requieren 2 puertos paralelos para el control de una MMH; por lo que, si se desea controlar más de una MMH, el número de puertos paralelos aumenta.

Alternativa 2: Utilizar algún dispositivo de control.

La segunda opción es utilizar un dispositivo electrónico que realice las tareas de control. Dentro del LMM se han desarrollado trabajos empleando las tres alternativas mencionadas en la tabla 3.1. Esta opción presenta las siguientes ventajas y desventajas:

- Ventajas:

  1.- El control de los movimientos básicos de la MMH se delega a este dispositivo, quedando en un nivel superior el software de control para gestionar dichos movimientos y conseguir una mayor funcionalidad.

  2.- Los movimientos de la MMH no se ven afectados por los retrasos provocados por el sistema operativo.

  3.- Se puede implementar, en este dispositivo, un control en lazo cerrado sobre los actuadores.

  4.- Se puede realizar en la PC, alguna otra tarea en paralelo con el software de control.

  5.- Se utiliza un sólo puerto de comunicación entre este dispositivo y la PC, con la opción de poder controlar más de una MMH (lo anterior, dependiendo de las características del dispositivo).

- Desventajas:

  1.- Para utilizar este tipo de dispositivos de control y que trabajen de forma conjunta con una PC, es necesario que las rutinas de control implementadas

en estos dispositivos sean capaces de efectuar movimientos correspondientes a una interpolación lineal; es decir que contengan una instrucción a la que se le dé como parámetro las coordenadas (X,Y) del punto inicial y final para que realice el maquinado correspondiente.

2.- La electrónica se complica y el tiempo de desarrollo aumenta.

3.2.3. Configuración seleccionada.

La herramienta de desarrollo de aplicaciones escogida es *Visual Basic 6.0*, ya que esta herramienta resultó ser la mejor opción, de acuerdo a la evaluación presentada anteriormente. La principal ventaja que presenta esta herramienta consiste en la disminución del tiempo que se requiere para el desarrollo de aplicaciones.

Los puertos de comunicación cuya transmisión se realiza de forma serial (puerto serie y puerto USB) presentan algunas desventajas frente al puerto paralelo, ya que estos puertos requieren de dispositivos electrónicos con el fin de manejar las señales enviadas desde la PC. Por otro lado, la complejidad para controlarlos es relativamente mayor que en el puerto paralelo. Por estas razones, el puerto de comunicación escogido es el puerto paralelo.

Debido a que el desarrollo hasta ahora alcanzado en la programación con los dispositivos de control se limita en conseguir movimientos independientes en los ejes X, Y y Z de la MMH, el desarrollo de un programa en la PC que manipule al dispositivo de control sería el mismo o hasta más elaborado que si se realiza sin este dispositivo. Por esta razón se ha decidido utilizar la etapa de potencia y acoplamiento de señales que se utiliza en el sistema de control actual.

3.2.4. Diagrama de flujo de datos, de la aplicación a desarrollar.

En la figura 3.7 se muestra cómo interactúan los módulos de la aplicación a desarrollar. Los elementos “joystick” y “programa pieza” constituyen la entrada al sistema para cada uno de los modos de control soportados por la aplicación (por joystick y mediante código G y M, respectivamente). El elemento “programa pieza” representa al archivo que contiene el conjunto de instrucciones de código G y M que serán procesadas para obtener una cierta pieza. Mientras que el elemento “joystick” está representado por 6 botones (2 botones -en direcciones opuestas- para cada uno de los ejes: X, Y, Z). Para el control mediante código G y M, el flujo de datos pasa por los módulos (1), (2) y (3). Mientras que el flujo de datos, para el modo de control por “joystick”, pasa por el módulo (2) y (3). En ambos modos de control, se puede deshabilitar el envío de señales a la MMH, por lo que en este caso el flujo de datos no pasaría por el modulo (3).

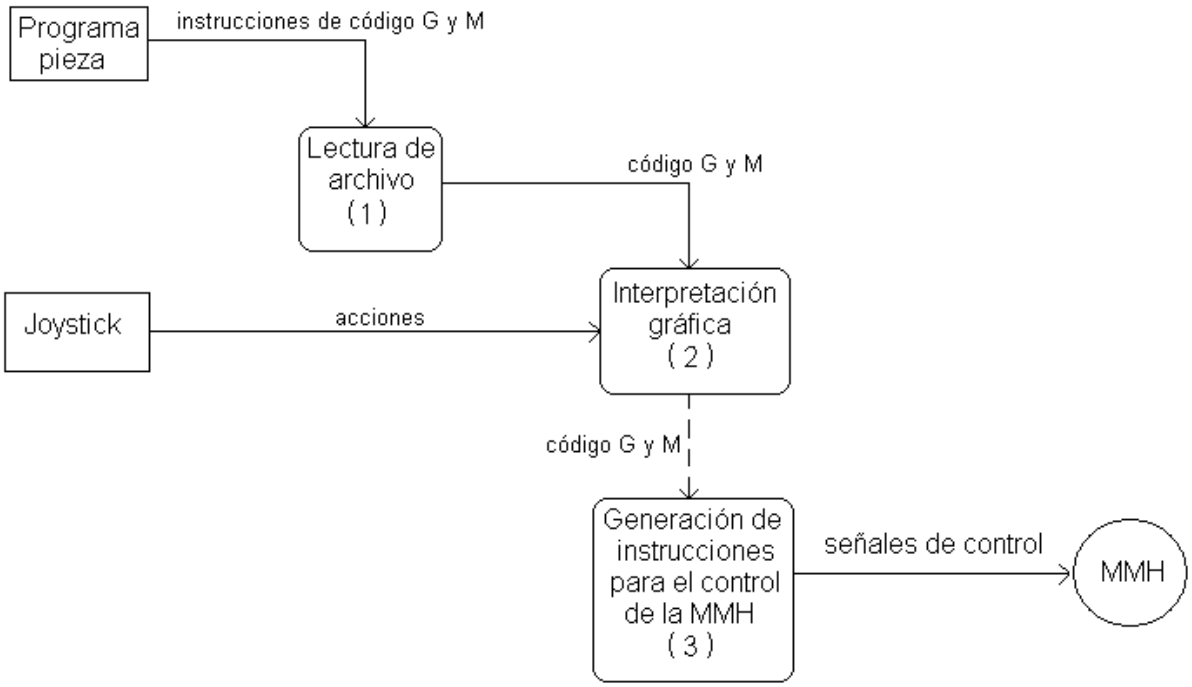

Figura 3.7. Diagrama de flujo de datos de la aplicación a desarrollar.

3.3. Implementación.

Esta parte consiste en la implementación de los módulos (1), (2) y (3), presentados en la figura 3.7. A continuación se detalla cada uno de ellos.

3.3.1. Descripción de los módulos de la aplicación.

3.3.1.a. Lectura de archivo.

Consiste en abrir el archivo para la lectura de las instrucciones en código G y M. Se lee instrucción por instrucción y durante esta lectura se detectan las cotas (superior e inferior) de cada eje. Después se copian todas las instrucciones a una lista dentro de la aplicación y por último se cierra el archivo.

DETECCIÓN DE COTAS.

El proceso de detección de cotas tiene la finalidad de obtener los valores para determinar el área en que se moverá la herramienta de corte y así, maximizar el gráfico que se mostrará como resultado de la interpretación gráfica. En este proceso, para el caso de instrucciones relacionadas con movimientos lineales (G00 y G01), resulta fácil identificar estos valores, pero para el caso de instrucciones que involucran movimientos circulares (G02 y G03), es necesario conocer varios puntos de la trayectoria. Para este último caso se utiliza una función que calcula un determinado número de puntos, tomando como parámetros de entrada el último punto a ser graficado y/o maquinado (tomado de la última instrucción leída), y los parámetros de la instrucción actual (G02 ó G03).

3.3.1.b. Interpretación gráfica.

Este proceso consiste en generar un gráfico que muestra la ruta que sigue la herramienta de corte durante un proceso de maquinado. Este módulo se encarga de atender dos tipos de flujo de datos; por un lado, atiende las acciones indicadas mediante el control por joystick, y por otro lado, atiende las instrucciones de código G y M contenidas dentro de un programa pieza. Para el control por joystick, se dibuja una línea dentro del cuadro de imagen de la aplicación en la dirección indicada por los botones del joystick (lo anterior, para los ejes X y Y). Para el control mediante código G y M, el gráfico obtenido es el resultado de interpretar una serie de instrucciones en código G.

INSTRUCCIONES DE CÓDIGO G INTERPRETADAS POR LA APLICACIÓN.

Las instrucciones de código G son las que realmente influyen en la generación del gráfico cuando la aplicación trabaja en el modo de control mediante código G y M.

Del conjunto de instrucciones de código G mencionadas en los antecedentes se escogieron las instrucciones más representativas conforme al tipo de movimiento que realizan. Estas instrucciones, que se implementan dentro de la aplicación para el control de micromáquinas herramienta, son:

- G00 Posicionamiento punto a punto.
- G01 Interpolación lineal para dimensiones medias.
- G02 Interpolación circular para dimensiones medias en sentido horario.
- G03 Interpolación circular en sentido antihorario.

Estas cuatro instrucciones son las básicas y con ellas se pueden efectuar todo tipo de movimiento para la manufactura de piezas en la MMH. La sintaxis de estas instrucciones, se describen a continuación.

- **Sintaxis para la instrucción G00 (posicionamiento punto a punto).**

G00 X*valor_x* Y*valor_y*
G00 Z*valor_z*

Donde *valor_x*, *valor_y* y *valor_z*, es la coordenada en el eje X, en el Y y en el Z, respectivamente, a donde se desea mover la herramienta de corte. En el primer caso, si una de las dos coordenadas (X o Y) no cambia de valor, aún así es necesario ponerla.

- **Sintaxis para la instrucción G01 (interpolación lineal para dimensiones medias).**

G01 X*valor_x* Y*valor_y* [F]
G01 Z*valor_z*

La sintaxis para esta instrucción es similar a la sintaxis de la instrucción G00. La diferencia que se refleja en el grafico obtenido, entre estas dos instrucciones, es el color que se emplea para graficar las trayectorias. Una instrucción de posicionamiento punto a punto (G00) se grafica en color naranja, mientras que para la interpolación lineal (G01) se emplea el color rojo. En el maquinado, la diferencia entre estas dos instrucciones radica en que la velocidad de avance es la máxima posible con G00; mientras que en G01 la velocidad de avance se establece mediante el parámetro F. El parámetro F sólo repercute al momento del maquinado, por lo tanto, al momento de generar el gráfico no se considera este parámetro.

- **Sintaxis para la instrucción G02 (interpolación circular en sentido horario).**

G02 X*valor_x* Y*valor_y* I*valor_i* J*valor_j*

Donde *valor_x* y *valor_y*, son las coordenadas del punto final del arco. Y *valor_i* y *valor_j*, son las coordenadas del centro del arco, relativas al último punto.

Ejemplo:

G01 X1.0 Y1.0 F3.0	Mueve la herramienta, en línea recta, a la posición (1,1).
G02 X3.0 Y3.0 I1.0 J1.0	Mueve la herramienta, realizando una interpolación circular en sentido horario, dando como resultado un arco que inicia en (1,1), con centro en (2,2) y finalizando en (3,3). A la misma velocidad que se ha establecido anteriormente (en este caso se estableció en la instrucción anterior.)

El resultado de estas dos instrucciones se puede apreciar en la figura 3.8.

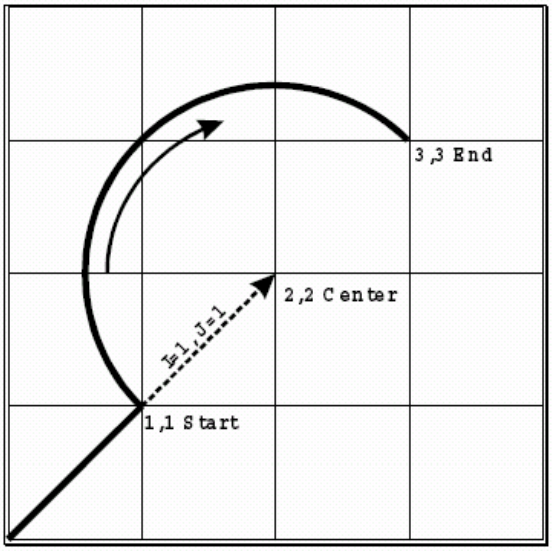

Figura 3.8. Ejemplo de la instrucción G02.

- **Sintaxis para la instrucción G03 (interpolación circular en sentido antihorario).**

G03 X*valor_x* Y*valor_y* I*valor_i* J*valor_j*

Donde *valor_x* y *valor_y*, son las coordenadas del punto final del arco. Y *valor_i* y *valor_j*, son las coordenadas del centro del arco, relativas al último punto.

Ejemplo:

G01 X2.0 Y1.0 F8.0	Mueve la herramienta, en línea recta, a la posición (2,1).
G03 X0.0 Y3.0 I-1.0 J1.0	Mueve la herramienta, realizando una interpolación circular en sentido antihorario, dando como resultado un arco que inicia en (2,1), con centro en (1,2) y finalizando en (0,3). A la misma velocidad que se ha establecido anteriormente (en este caso se estableció en la instrucción anterior.)

El resultado de estas dos instrucciones se puede apreciar en la figura 3.9.

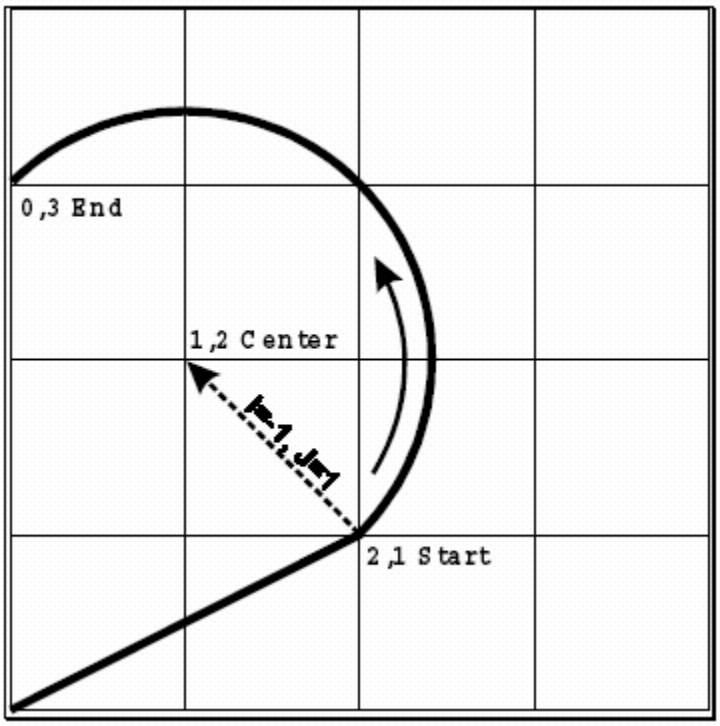

Figura 3.9. Ejemplo de la instrucción G03.

La sintaxis para estas instrucciones se tomó con base en el código numérico que genera el software *FLASHCUT™* al importar un archivo DXF. Aunque estas instrucciones son en su mayoría estándar, en el caso de la interpolación circular existe también otro tipo de sintaxis. Sin embargo, la aplicación desarrollada se apego a la sintaxis manejada por *FLASHCUT™* .

3.3.1.c. Generación de instrucciones para el control de la MMH.

Este proceso consiste en enviar las señales de control a cada uno de los actuadores de la MMH, como respuesta a las ordenes recibidas por los modos de control que ofrece la aplicación. Para abundar sobre estos modos de control, es necesario mencionar las características de los actuadores, la configuración de los puertos paralelos y el hardware adicional empleado. A continuación se detallan estos elementos.

3.3.1.c.1. Características de los actuadores empleados en la MMH.

La MMH emplea cuatro motores a pasos, de imán permanente y bipolares, para conseguir el movimiento en el husillo y en cada eje. Tres de estos motores están acoplados a tres cajas de reducción para lograr una resolución, en el desplazamiento de cada eje, de 1.88 [μm] por paso del motor. El cuarto motor ofrece una resolución de 10° de giro por paso del motor. La velocidad de estos actuadores dependerá de la frecuencia de los pulsos que les envié el software de control.

También, la MMH, emplea tres sensores de contacto al inicio del desplazamiento de cada eje, para determinar la posición de inicio con mayor facilidad. Un cuarto sensor de contacto eléctrico se utiliza para determinar el momento en el que el material de trabajo hace contacto con la herramienta de corte.

3.3.1.c.2. Configuración de los puertos paralelos.

Para el control de la MMH se necesitan 4 señales para cada uno de los motores, por lo que se requiere un total de 16 salidas de la PC. Y para cada sensor de la MMH, se necesita una entrada, por lo que son 5 entradas. La distribución de estas entradas y salidas en cada uno de los puertos paralelos se muestra en la tabla 3.7.

Dir. del puerto paralelo	Subpuerto	Bits	Dirección	Destinado a:
0x378	Datos	D0-D3	Salida	Controlar motor en el eje X.
0x378	Datos	D4-D7	Salida	Controlar motor en el eje Y.
0xBC00	Datos	D4-D7	Salida	Controlar motor en el eje Z.
0xBC00	Datos	D0-D3	Salida	Controlar motor en el eje de rotación (husillo).
0xBC01	Estado	S3-S7	Entrada	Entrada de los sensores.

Tabla 3.7. Distribución de las señales de entrada y salida en los puertos paralelos.

3.3.1.c.3. Hardware adicional.

Para el hardware adicional, se ocupó el arreglo ya existente que se utiliza con el sistema de control actual. Este hardware está formado, principalmente, por un amplificador de DC de 16 canales y unos acondicionadores de voltaje TTL. El primero, se ocupa para amplificar las señales provenientes de los puertos paralelos de la PC, con el fin de que los actuadores reciban las señales con la corriente necesaria para su funcionamiento; mientras que los acondicionadores de voltaje TTL, adecuan las señales provenientes de los sensores de contacto para que puedan ser leídas por el puerto paralelo de la PC.

3.3.1.c.4. Modos de control.

La aplicación ofrece dos modos de control, que son: control por joystick y control mediante código G y M. Existe también, una rutina que regresa los carros de la MMH a una posición inicial. Estos modos de control se detallan a continuación.

3.3.1.c.4.1. Modo de control mediante código G y M.

Este modo de control consiste en procesar un conjunto de instrucciones en código G y M (programa pieza), generando así las señales correspondientes para el movimiento de la MMH. Estas instrucciones se pueden ejecutar una por una o en forma secuencial hasta alcanzar el final del programa pieza.

Para este modo de control, además de las instrucciones de código G (mostradas en la parte de la interpretación gráfica), también se procesa un conjunto de funciones auxiliares o instrucciones en código M que hacen referencia al modo de funcionamiento de una MHCNC y del control numérico.

De las instrucciones de código M mencionadas en los antecedentes, no todas están presentes en una MHCNC sino que cada máquina especial adopta las funciones necesarias para realizar su trabajo. Por ejemplo, una fresadora, un torno y una máquina de electroerosión requieren un diferente conjunto de instrucciones. También, el número de instrucciones que soporta una MHCNC depende de sus características; por ejemplo, si una MHCNC no tiene la funcionalidad de realizar un cambio automático de herramienta, entonces las instrucciones asociadas con tal efecto no estarán soportadas por su control numérico. Por tal motivo, para el funcionamiento de la aplicación para el control de micromáquinas herramienta se han seleccionados las siguientes instrucciones:

- M00 Parada programada.- Esta instrucción detiene el funcionamiento de la máquina herramienta hasta que el operador dé la orden de reanudar.
- M02 Fin de programa.
- M03 Rotación del husillo en sentido horario.
- M04 Rotación del husillo en sentido antihorario.
- M05 Parada del husillo.

- M30 Fin de programa y vuelta a la instrucción número 1.

Para el procesamiento de estas instrucciones en código G y M, se utiliza una rutina que divide cada instrucción en partes. Primero se identifica el tipo de instrucción, ya sea G o M, y dependiendo de la instrucción que se trate, se obtienen los parámetros asociados a dicha instrucción. Una vez que se conoce el tipo de instrucción y sus parámetros, se mandan a llamar las funciones adecuadas para efectuar el maquinado correspondiente; y después del maquinado, se dibuja dicho trazo en el cuadro de imagen de la aplicación. La figura 3.10 muestra un esquema de la forma en que se procesan estas instrucciones.

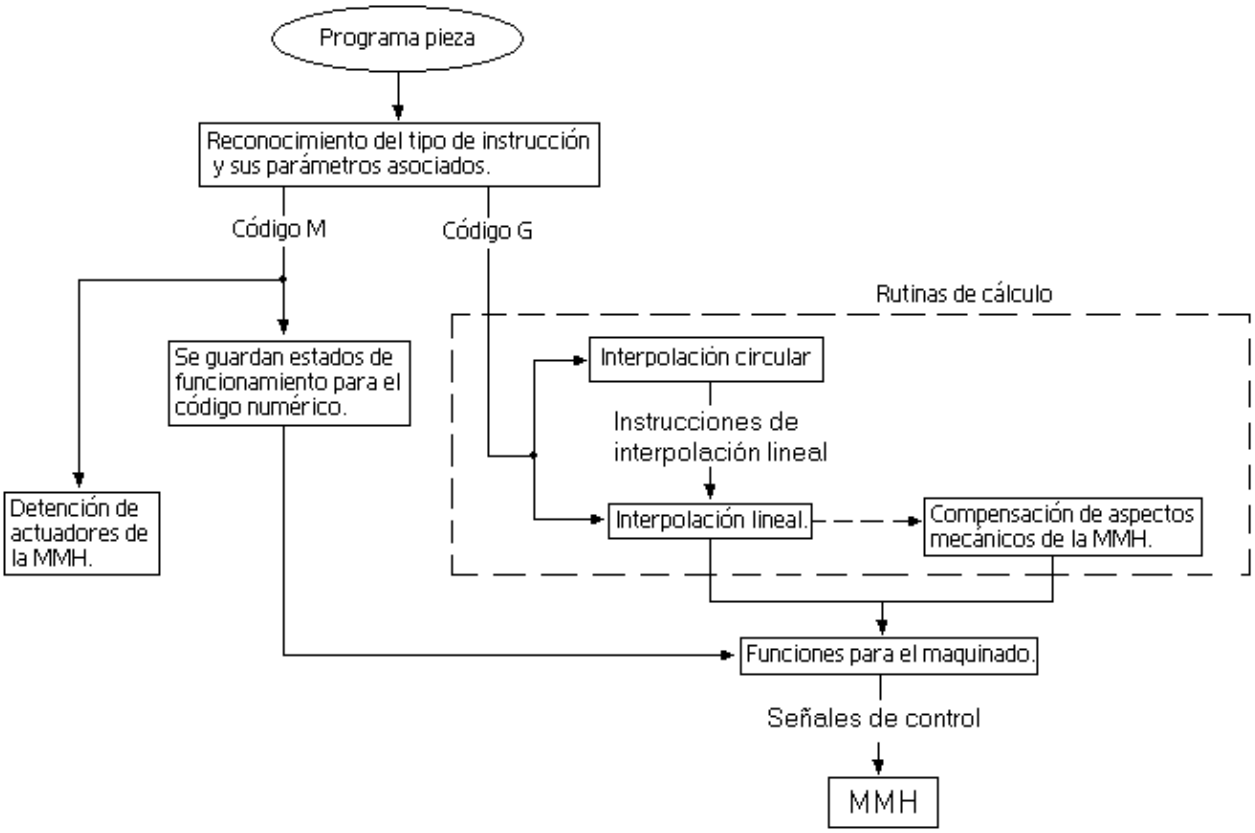

Figura 3.10. Diagrama del modo de control mediante código G y M.

Para este modo de control, las instrucciones en código M tomarán diferentes acciones como es: detener el funcionamiento de los actuadores de la MMH, o guardar diferentes estados para el funcionamiento del control numérico.

Las instrucciones en código G se procesan de acuerdo al tipo de instrucción que se trate (posicionamiento punto a punto, interpolación lineal o interpolación circular). Las instrucciones de posicionamiento punto a punto y las de interpolación lineal son procesadas por la rutina “interpolación lineal”, como se muestra en la figura 3.10. Esta rutina utiliza a las “funciones para el maquinado”, a las cuales les envía los parámetros necesarios para su funcionamiento. Finalmente, las “funciones para el maquinado” generan las señales de control que son enviadas a la MMH. En caso de que se requiera compensar el backlash, la rutina de

“interpolación lineal” ejecuta una rutina adicional, la cual a su vez, utiliza una de las “funciones para el maquinado” que envía las señales de control necesarias para realizar dicha compensación.

Por otro lado, las instrucciones de interpolación circular pasan por una rutina que calcula una serie de puntos correspondientes a un arco de circunferencia y genera, con dichos puntos, una serie de instrucciones de interpolación lineal que son enviadas a la rutina “interpolación lineal” (antes mencionada) para ser procesadas.

3.3.1.c.4.1.1. Funciones para el maquinado.

Para realizar el maquinado de cualquier pieza con la MMH, utilizándola como fresadora, se han implementado dos funciones esenciales y una función adicional que sirve para compensar el backlash de la MMH. Estas funciones se describen a continuación.

Función XYH.

Esta función envía las señales adecuadas, a través de los puertos paralelos, a los actuadores correspondientes de los ejes X, Y y husillo. Este envío de señales es multiplexado en el tiempo de forma tal que pueden estar los tres actuadores trabajando simultáneamente. Sólo en el caso de que el desplazamiento en el eje X o Y sea igual a cero, entonces, estarán trabajando sólo dos actuadores (el husillo siempre estará en funcionamiento).

En seudocódigo, la declaración para esta función es de la siguiente forma:

Void ***funcionXYH*** *(double valorX , double valorY)*

Como se aprecia en la declaración anterior, realmente se trata de un procedimiento y no de una función, el cual recibe como parámetros dos valores numéricos (tipo double), que corresponden al desplazamiento (X, Y), relativo al punto en el que estén situados los ejes de la MMH. El diagrama de flujo para esta función es el que se muestra en la figura 3.11. El envío de señales a los motores de los ejes X, Y y husillo siguen una misma lógica, la cual se muestra en el diagrama de flujo de la figura 3.12.

En la rutina de envío de señales para el control del husillo, la variable booleana *horarioH* (análoga a la variable *horarioX* de la rutina de envío de señales para controlar al eje X; figura 3.12) obtiene su valor del conjunto de instrucciones de código G y M, más específicamente de las funciones auxiliares M03 y M04, las cuales indican que el sentido de rotación del husillo será horario y antihorario, respectivamente.

La función *tiempoActual( )*, que se utiliza dentro de la *función XYH*, corresponde a la función *timeGetTime( )* de *Visual Basic 6*. Con esta función se obtiene el tiempo actual del sistema.

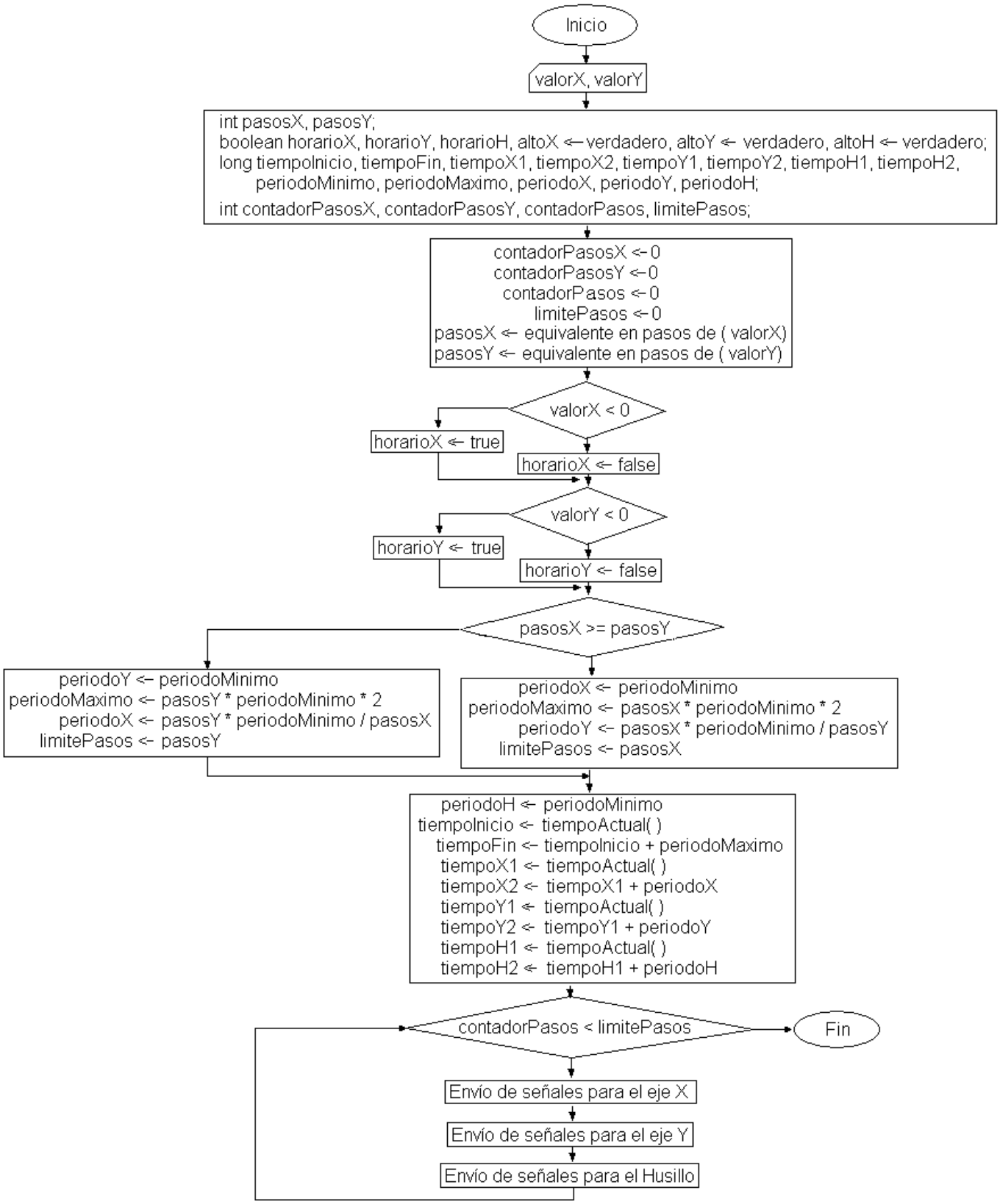

Figura 3.11. Diagrama de flujo para la función XYH

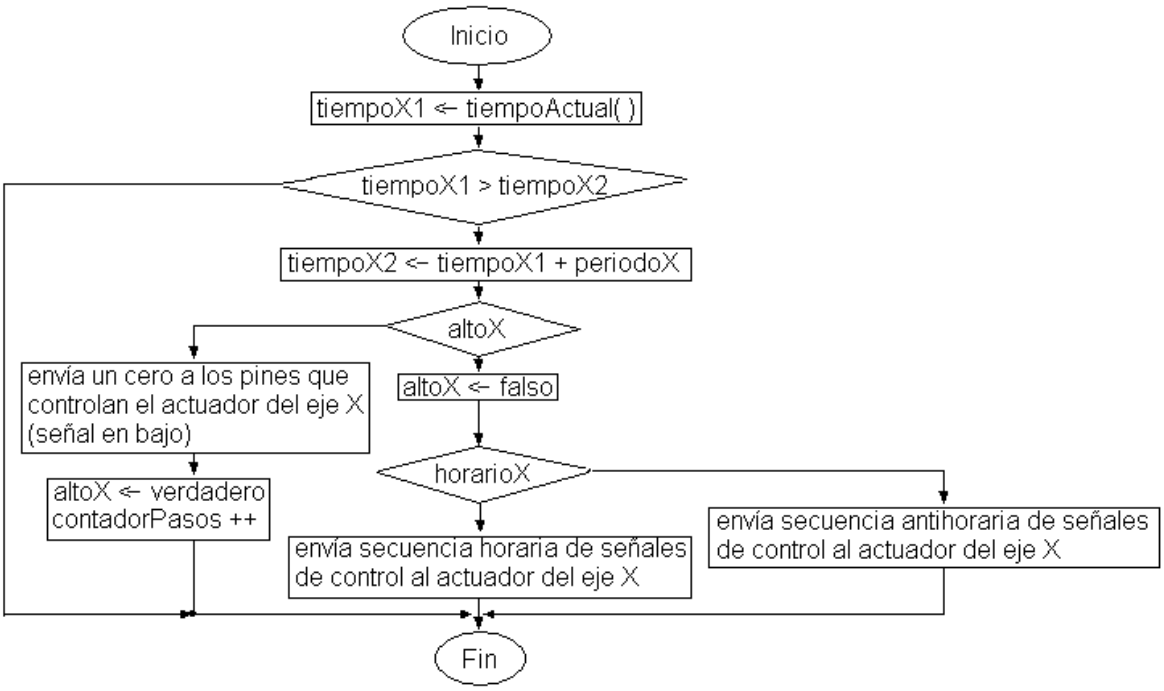

Figura 3.12. Diagrama de flujo para la rutina de envío de señales para controlar al eje X, contenida dentro de la función XYH.

La secuencia de datos que se envía a cada motor para que gire en un sentido, es la que se muestra en la figura 3.13. El sentido de giro contrario se logra invirtiendo el orden en que se envía dicha secuencia de datos. Para este tipo de motores, se mantienen energizadas sus dos bobinas con el fin de obtener el mayor par posible. Los bits que se envían por A y B corresponden a los extremos de una bobina, mientras que B y C corresponden a los extremos de la otra bobina. Para el envío de estos bits se debe tomar en cuenta la posición que le corresponde a cada actuador dentro del puerto de datos del puerto paralelo. Estos bits se envían mediante una función cuyos parámetros son: una dirección de un subpuerto del puerto paralelo y el valor de los bits enviados en su equivalente decimal.

A	D	B	C
1	0	0	1
1	1	0	0
0	1	1	0
0	0	1	1

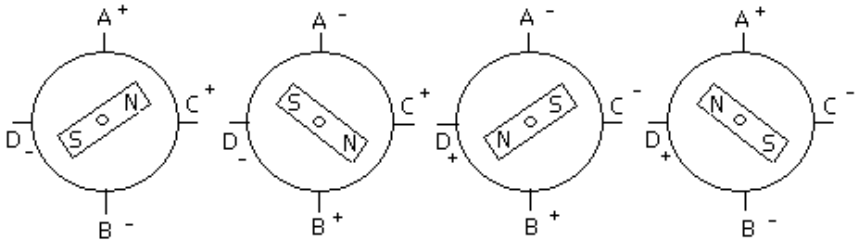

Figura 3.13. Secuencia de datos para el control de los actuadores.

La variable *periodoMinimo*, de la *función XYH*, corresponde al tiempo mínimo en el que un pulso permanecerá en alto o en bajo; es decir, es igual a la mitad del período de la señal enviada a una de las terminales (A, B, C, o D) de un motor, y donde dicha señal tiene un ciclo de trabajo del 50%.

La rutina de envío de todas las señales de control necesarias para efectuar el traslado de los ejes de la MMH al punto deseado (pasado como argumento a esta función), se completa cuando la variable *contadorPasos* alcanza el valor de la variable *limitePasos*, cuyo valor es igual al mayor número de pasos que deba moverse la MMH en sus ejes X o Y.

Función ZH.

Esta función es la encargada de enviar las señales de control a los actuadores del eje Z y husillo, manteniendo una lógica similar a la que presenta la *función XYH*. La declaración, en seudocódigo, para esta función queda de la siguiente forma:

Void ***funcionZH*** *(double valorZ)*

Como se aprecia en la declaración anterior, este procedimiento recibe como argumentos un valor numérico (tipo double) que corresponde al desplazamiento en el eje Z, relativo al punto en el que se encuentran los ejes de la MMH. La lógica que sigue esta función es muy similar a la que presenta la *función XYH*, nada más que en este caso el envío de señales se realiza para el eje Z y husillo. Debido a que la trayectoria de corte para esta función es en un sólo eje, el período de las señales enviadas al actuador del eje Z siempre es el más corto, lo que sucede de igual forma con las señales enviadas al actuador del husillo.

Función backlashXY.

La declaración, en seudocódigo, para esta función es de la siguiente forma:

Void ***backlashXY*** *(String eje, boolean horario)*

Este procedimiento es llamado por la *función XYH* en el caso de que se detecte un cambio de sentido en el movimiento del eje X o Y; por ejemplo, si la MMH se movió en sentido positivo del eje X en la instrucción anteriormente procesada de código G, y la instrucción actual que va a procesar la *función XYH* requiere el movimiento en sentido negativo del eje X, entonces, se ejecuta la rutina backlashXY, pasándole como parámetros el eje en el que se va a hacer la compensación (mediante la variable *eje*) y el sentido en el que se va a realizar dicha compensación (mediante la variable *horario*).

Este procedimiento sigue una lógica similar a la de la *función XYH*, sin embargo, la variable que limita el envío de señales al eje en el que se está realizando la compensación tiene un valor fijo, en número de pasos, que corresponde al backlash que se tiene en el eje X y Y. Este valor teórico es de 345.27[μm], que corresponde a 182 pasos aproximadamente[9]. Sin embargo, en las pruebas realizadas con estas funciones se detectó un valor diferente debido, en gran parte, al desgaste que ha tenido esta MMH. A continuación se presentan las pruebas realizadas con estas funciones.

3.3.1.c.4.1.2. Pruebas realizadas con la aplicación, la cual utiliza las funciones para el maquinado.

Durante el proceso de depuración de estas funciones, se realizaron diferentes maquinados de figuras geométricas básicas como: rectas, cuadrados, rectángulos y círculos; principalmente. Estas pruebas se realizaron con el sistema completo (una PC y el software de control, una etapa de potencia y la MMH).

Como material de trabajo, para estas pruebas, se utilizaron láminas de latón delgadas bañadas con tinta; y como herramienta de corte se utilizó una punta de latón. Durante el maquinado la punta de latón hace contacto con la lámina, de tal forma que sólo marca la trayectoria seguida por la herramienta al quitar la tinta que cubre a la lámina. En la figura 3.14 se muestra el sistema justo antes de realizar un maquinado.

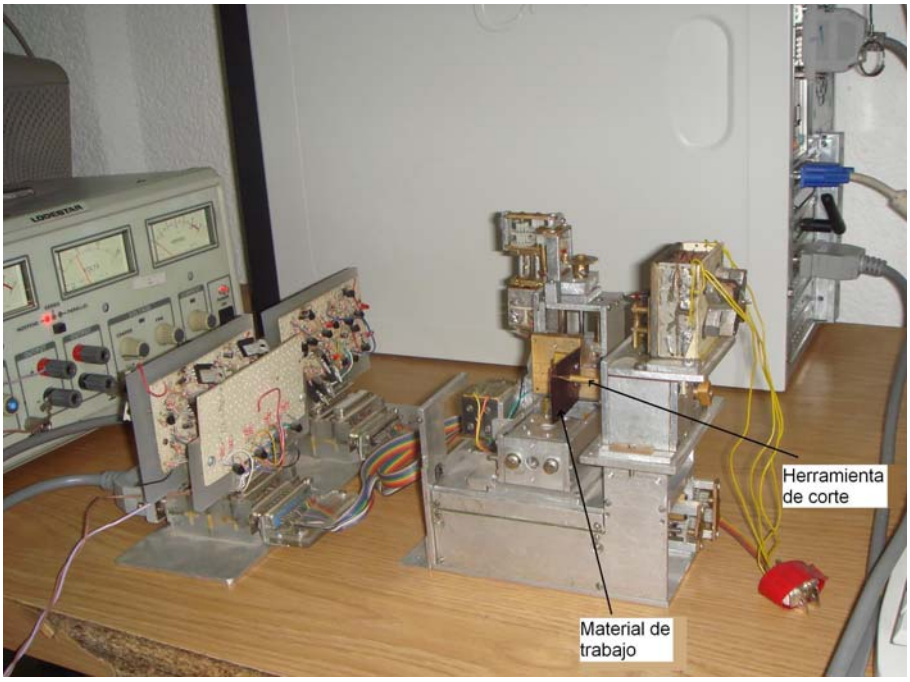

Figura 3.14. Imagen del sistema antes de realizar un maquinado.

Una prueba que se realizó consistió en maquinar un rectángulo sin tomar en cuenta la rutina de compensación de backlash. En este maquinado, que se muestra en la figura 3.15, se aprecia que el rectángulo no alcanza a cerrarse. Además, se puede observar que para llegar al punto en el que se inició el maquinado (para completar el rectángulo), la distancia que faltó recorrer en el eje X es mayor a la distancia que faltó recorrer en el eje Y, lo cual nos indica que el backlash en el eje X es mayor que en el eje Y. Estas distancias fueron medidas en el laboratorio de Metrología del CCADET, UNAM; obteniéndose los siguientes valores:

Para completar el rectángulo, la distancia que faltó recorrer en el eje X fue de 0.675 [mm], y en el eje Y fue de 0.321 [mm].

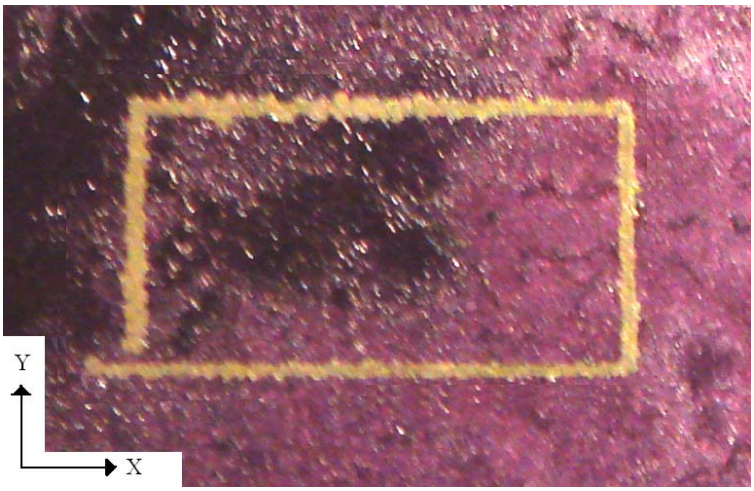

Figura 3.15. Maquinado de un rectángulo sin utilizar la rutina de compensación de backlash.

Para tener una mejor medición del backlash se realizó una prueba que consistió en maquinar dos rectas paralelas al eje X unidas en uno de sus extremos, sin tomar en cuenta la rutina de compensación de backlash. En este maquinado, para llegar al punto de inicio se mueve la herramienta en el mismo sentido que se va a maquinar la primera recta en el eje X, esto con el fin de que al inicio del maquinado no exista pérdida de pasos provocada por el bascklash (no existe backlash). Para la segunda recta, como existe un cambio de sentido en el movimiento del eje X (lo cual da lugar a la existencia de backlash), existe entonces una pérdida de pasos, y por consiguiente, esta segunda recta es de longitud menor que la primera. Al medir la diferencia de longitud entre las dos rectas, obtenemos el backlash en el eje X. Este mismo procedimiento se realizó también para el eje Y. En la figura 3.16 se muestran estos dos maquinados.

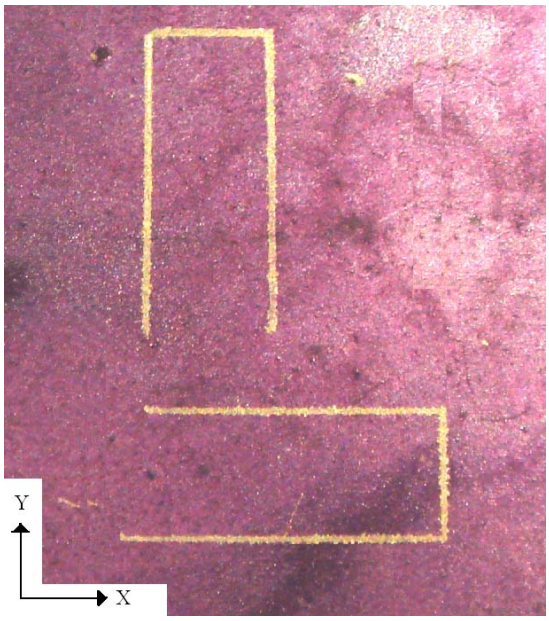

Figura 3.16. Maquinados para determinar el backlash en el eje X y Y.

Las mediciones de las rectas que conforman el maquinado para determinar el backlash en los ejes X y Y, se realizó en el Laboratorio de Metrología del CCADET. Las mediciones que se obtuvieron fueron las siguientes:

Backlash en el eje X de 0.654 [mm], que corresponde a 348 pasos aproximadamente. Y backlash en el eje Y de 0.302 [mm], que corresponden a 161 pasos aproximadamente.

Otra de las pruebas que se realizaron fue la de observar, en el osciloscopio, las señales de salida de los puertos paralelos correspondientes al maquinado de una recta inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y. En esta prueba se varió la frecuencia de la señal, cambiando el valor de uno de los campos que aparecen dentro de la aplicación ("*periodo medio de la señal*", en la ventana de configuración de la aplicación). En las figuras 3.17, 3.18, 3.19 y 3.20 se muestran las señales registradas con el osciloscopio para un periodo medio de la señal igual a 3 [ms].

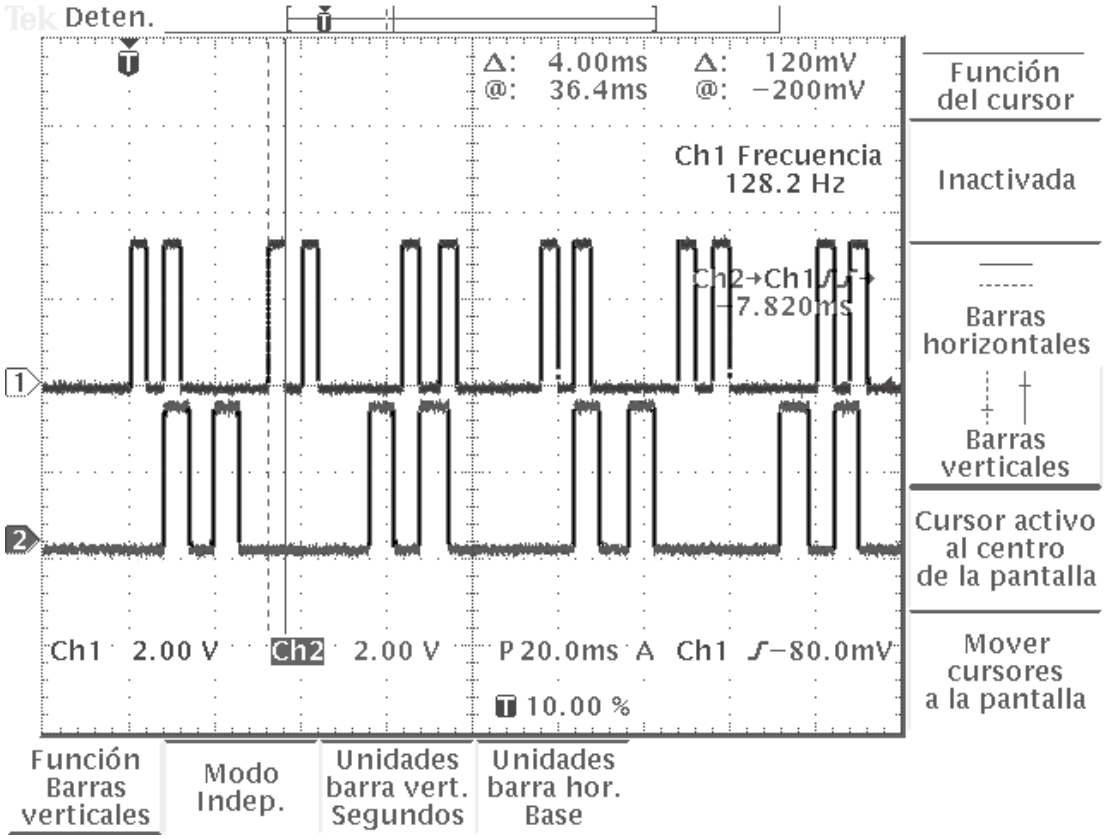

Figura 3.17. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 3 [ms]. Medición del tiempo en el que permanece en alto la señal en el eje X.

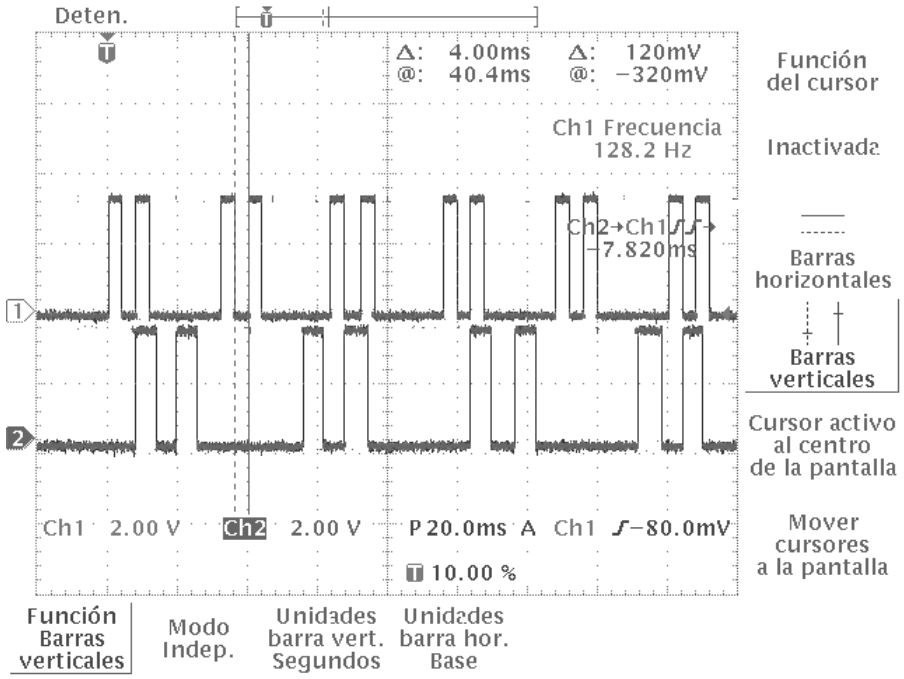

Figura 3.18. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 3 [ms]. Medición del tiempo en el que permanece en bajo la señal en el eje X.

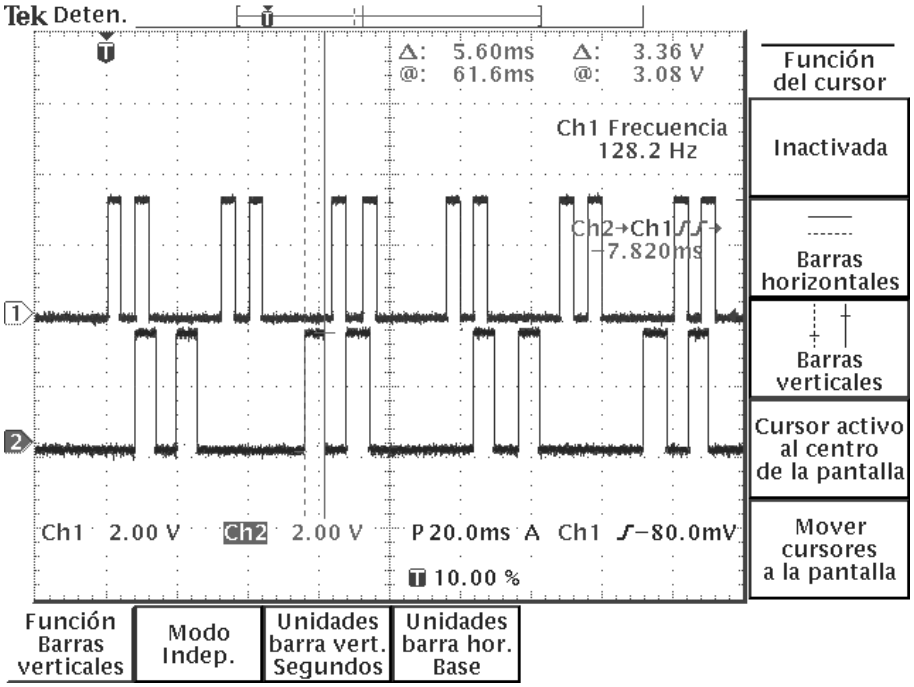

Figura 3.19. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 3 [ms]. Medición del tiempo en el que permanece en alto la señal en el eje Y.

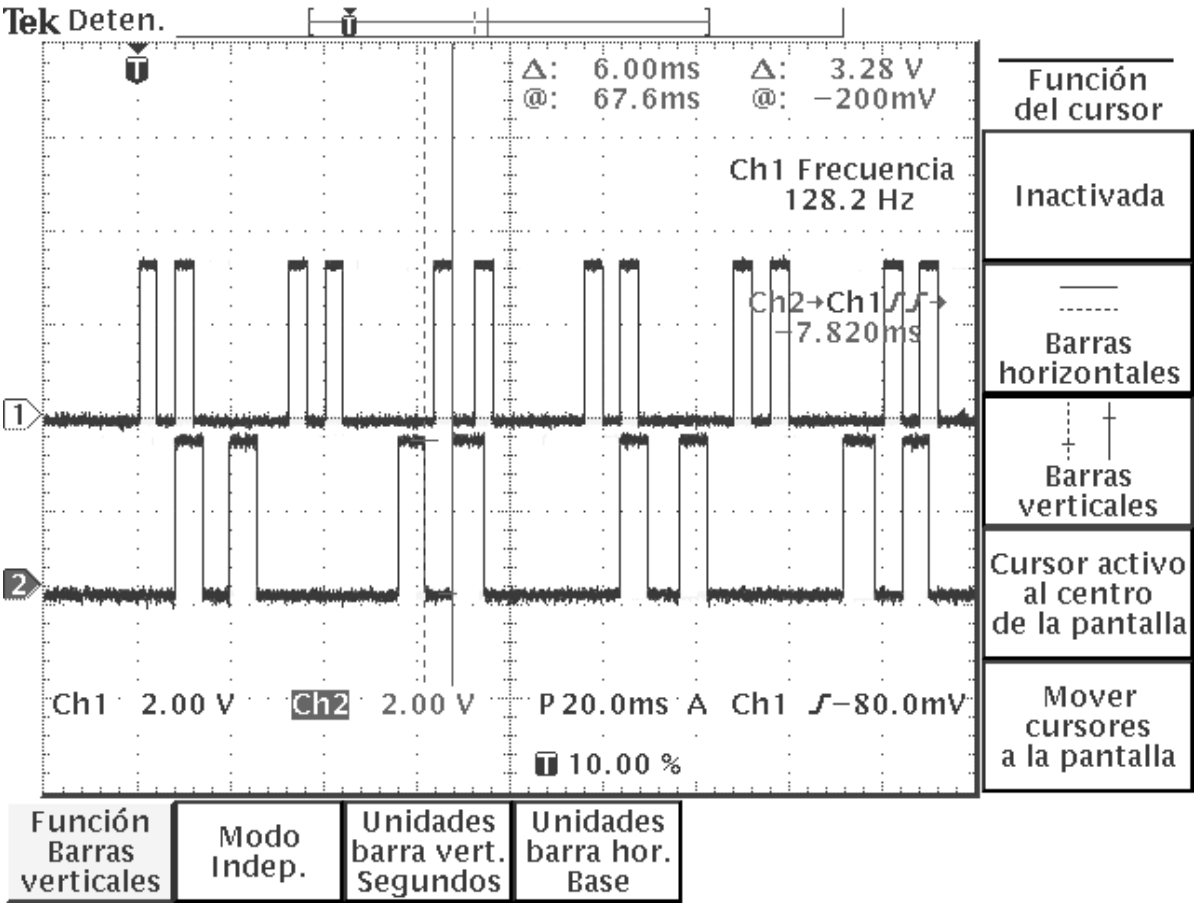

Figura 3.20. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 3 [ms]. Medición del tiempo en el que permanece en bajo la señal en el eje Y.

Como se puede observar en estas figuras (3.17, 3.18, 3.19 y 3.20), el tiempo en que permanece en alto la señal para el eje X difiere en 1 [ms], respecto al valor establecido dentro de la aplicación. Lo mismo pasa cuando la señal, en el eje X, tiene un nivel bajo de voltaje. Mientras que para el eje Y, el tiempo en que permanece en alto y en bajo la señal se aproxima más al tiempo esperado (a 6 [ms], el doble que en el eje X debido a que la distancia recorrida en el eje Y es la mitad de la recorrida en el eje X). Este mismo maquinado se realizó ahora con un periodo medio de la señal igual a 20 [ms]. En las figuras 3.21, 3.22, 3.23 y 3.24, se muestran las señales registradas en el osciloscopio, para este último maquinado.

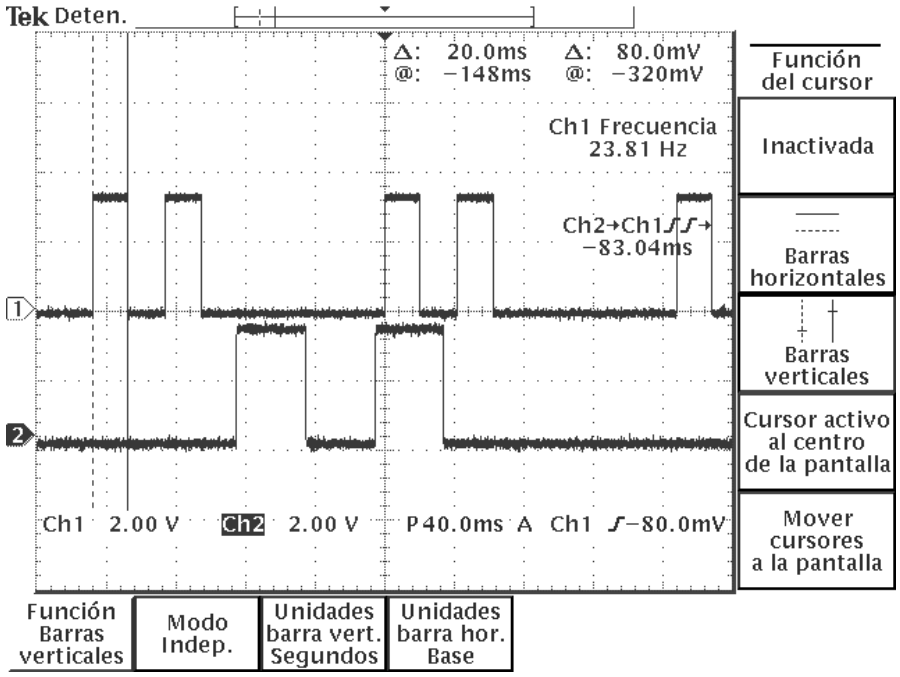

Figura 3.21. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 20 [ms]. Medición del tiempo en el que permanece en alto la señal en el eje X.

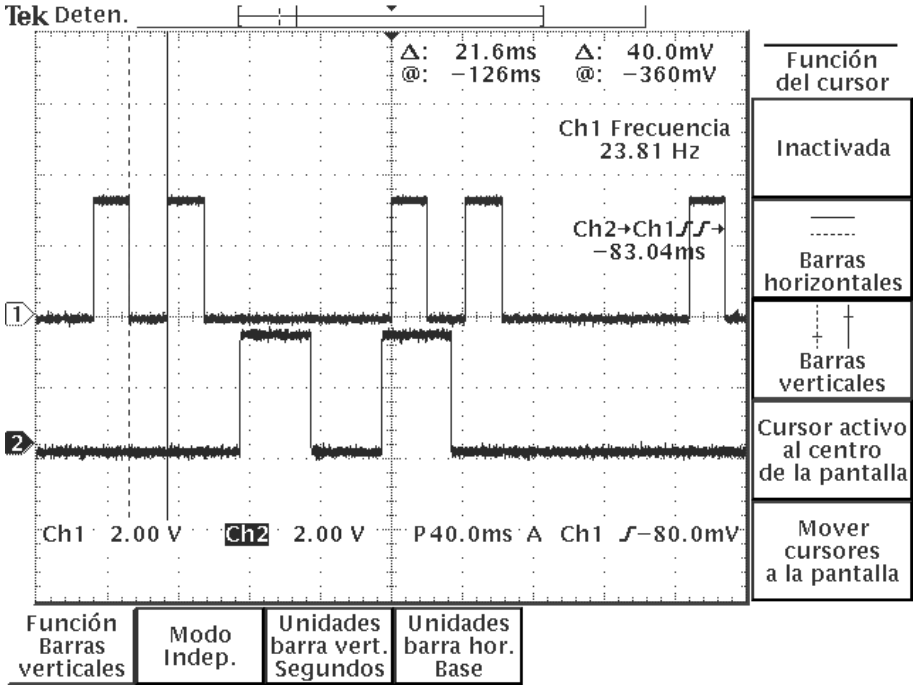

Figura 3.22. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 20 [ms]. Medición del tiempo en el que permanece en bajo la señal en el eje X.

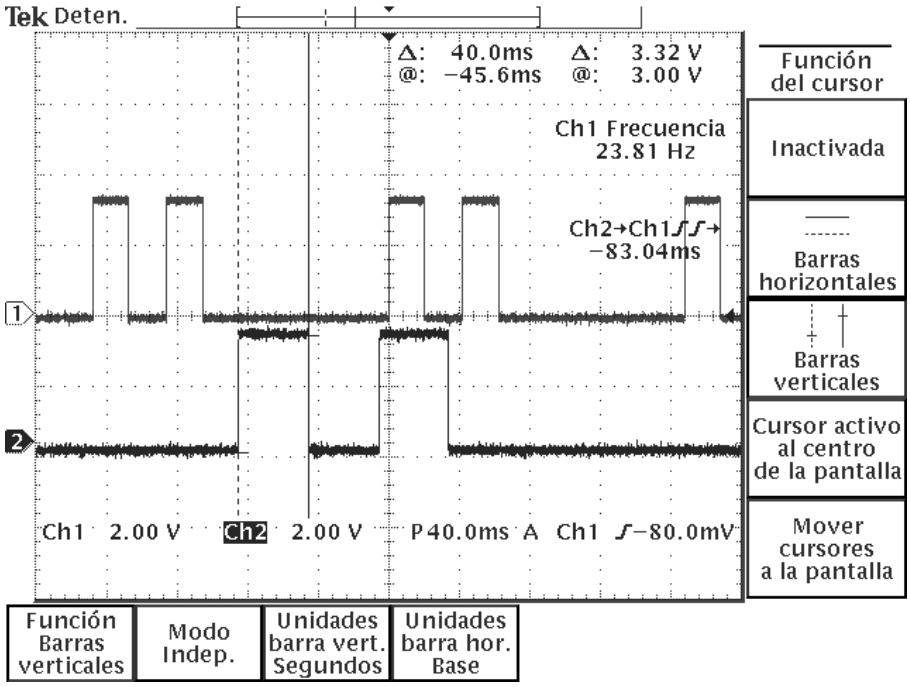

Figura 3.23. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 20 [ms]. Medición del tiempo en el que permanece en alto la señal en el eje Y.

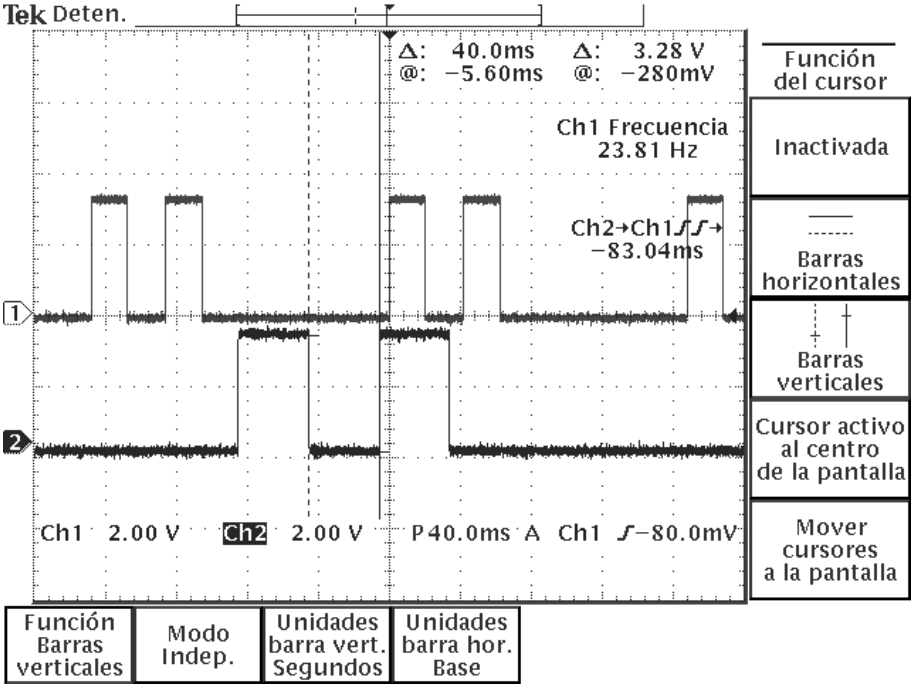

Figura 3.24. Señales de salida del maquinado de una línea inclinada cuyo recorrido en el eje X es el doble del recorrido en el eje Y, con un medio periodo igual a 20 [ms]. Medición del tiempo en el que permanece en bajo la señal en el eje Y.

Como puede verse, el valor de la frecuencia obtenido para estas señales mejora notablemente al aumentar el valor del periodo medio de la señal, el cual se establece dentro del apartado de configuración de la aplicación. Por lo que para conseguir un mejor maquinado, se aconseja establecer un mayor periodo medio de la señal en maquinados que realicen interpolación en dos ejes (X y Y). Mientras que para maquinados que se realicen en un sólo eje, se puede establecer el mínimo valor para el periodo medio de la señal, sin afectar el resultado en el maquinado y reduciendo considerablemente el tiempo de este maquinado. Lo anterior se logra gracias a que la *función XYH* está limitada por el mayor número de pasos que debe recorrer la herramienta de corte en el eje X o Y, con lo cual la variación en el periodo de la señal no repercute en maquinados que se realicen en un solo eje.

Para este mismo maquinado se contabilizó el número de pasos que se recorren en cada eje al variar el periodo medio de la señal. En la tabla 3.8, se muestran los resultados obtenidos.

Los valores teóricos del número de pasos que se deben efectuar para este maquinado son los siguientes:

Valores teóricos: número de pasos en el eje X = 5263, número de pasos en el eje Y = 2632.

Número de pasos recorridos en el eje X	Número de pasos recorridos en el eje Y	Periodo medio de la señal [ms]
5263	2763	20
5263	2719	30
5263	2697	40

Tabla 3.8. Número de pasos recorridos al variar el periodo medio de la señal para el maquinado de una recta inclinada cuyo desplazamiento en el eje X es de 10 [mm] y en el eje Y es de 5 [mm].

Como se observa en la tabla anterior, con un periodo medio de la señal de 40 [ms] se consigue una aproximación muy buena para el eje que no está limitado en número de pasos. En caso de que se desee una mayor precisión, el usuario debe entonces establecer un mayor periodo medio de la señal.

3.3.1.c.4.2. Modo de control por joystick y regreso de los ejes a la posición de inicio.

El modo de control por joystick tiene la finalidad de mover cualquiera de los ejes X, Y o Z en ambos sentidos, mediante la activación de unos botones asociados a dichos movimientos. La lógica que se sigue para darle funcionalidad a los eventos de dichos botones, es similar a la que presentan las funciones de maquinado descritas anteriormente. Este tipo de control es usado principalmente para llegar al punto que será establecido como el cero del programa, donde a partir de este

punto se inicia el maquinado de un programa pieza. El diagrama de la figura 3.25 muestra el funcionamiento para este modo de control.

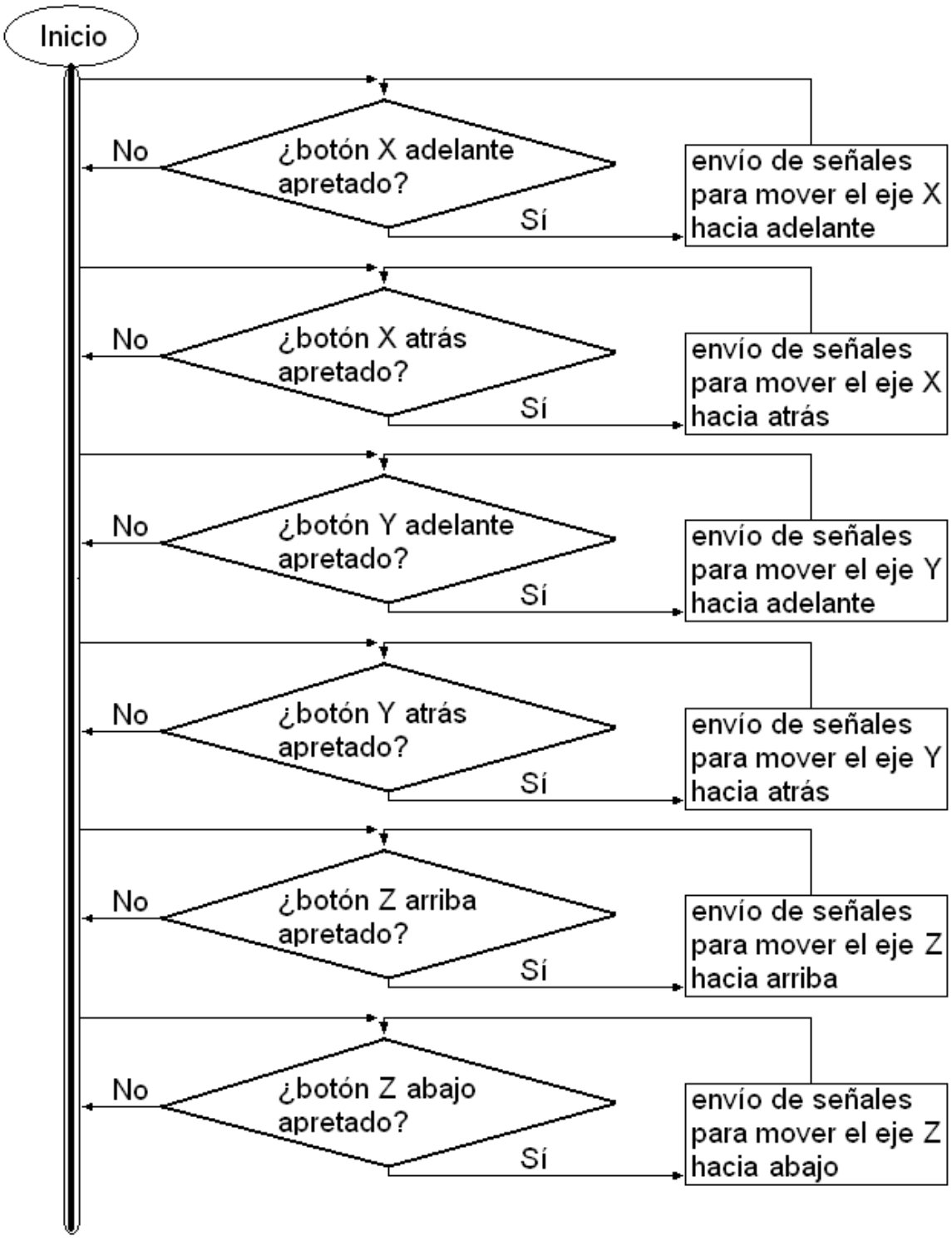

Figura 3.25. Diagrama del funcionamiento para el modo de control por joystick.

Existe también un procedimiento que regresa los tres ejes (X, Y, Z) a su posición de origen (ver figura 3.26). En este procedimiento el actuador del eje Z es puesto en funcionamiento hasta alcanzar la posición de origen que está determinada por la activación de un sensor de contacto en este eje. De igual forma, el siguiente actuador que se pone a funcionar, es el correspondiente al eje Y; mientras que el regreso a la posición de origen en el eje X, se realiza en última instancia. Este orden en el movimiento de los ejes responde a cuestiones de seguridad. Las rutinas de movimiento de los ejes se ejecutan bajo un ciclo en el que constantemente se lee el puerto de estado para determinar si es necesario regresar algún eje a su posición de inicio o ya se ha alcanzado dicha posición.

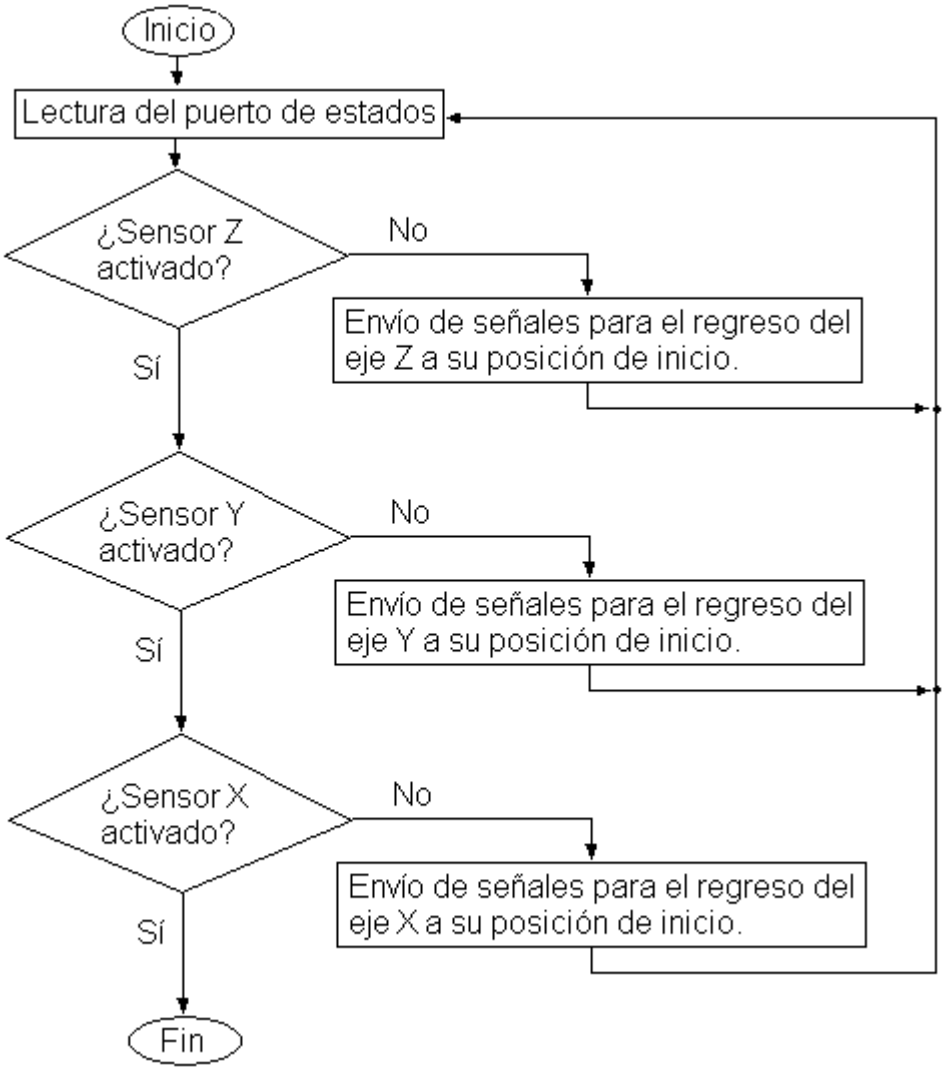

Figura 3.26. Diagrama de flujo del procedimiento que regresa los tres ejes a su posición de inicio.

3.3.2. Detalle de la aplicación a nivel de usuario.

3.3.2.a. Descripción de las pantallas que conforman la aplicación.

La pantalla principal de la aplicación es la que se muestra en la figura 3.27. En esta pantalla se pueden distinguir cinco secciones: un área de menús, un cuadro de imagen, un cuadro de lista, un área de despliegue de coordenadas y un área de control de la MMH.

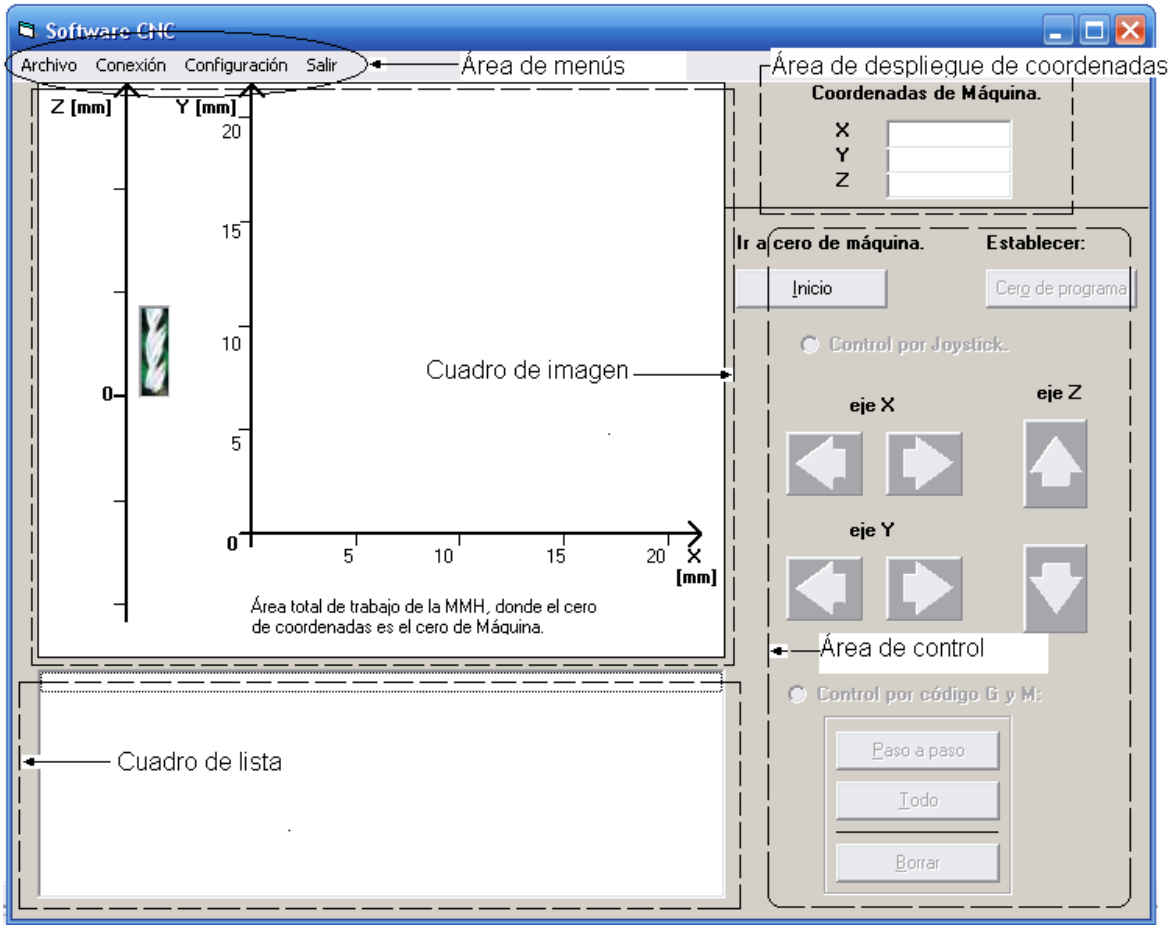

Figura 3.27. Pantalla principal de la aplicación para el control de MMHs.

El *cuadro de imagen* es la sección en la cual se muestra gráficamente el resultado de procesar un programa pieza y el resultado de efectuar las acciones de control mediante joystick, ya sea que se envíen o no las señales de control a la MMH. En este cuadro de imagen se encuentra el plano XY y el eje Z; en este último, el movimiento de la herramienta de corte está representado por una imagen que se mueve a lo largo de este eje, conforme las instrucciones de código numérico son procesadas. De igual forma, sobre el plano XY, aparece un grafico de las trayectorias de movimiento que sigue la MMH. La escala de graduación para los tres ejes (X,Y,Z) se determina en base al proceso de detección de cotas, antes mencionado. Las cotas para el eje X y Y son iguales (se escoge el mayor entre el

valor máximo del eje X y el valor máximo del eje Y), mientras que para el eje Z, la cota superior e inferior tienen el mismo valor absoluto (se escoge el mayor, sin considerar el signo, entre el valor mínimo y máximo detectado en este eje).

El *cuadro de lista* es la sección en la cual se escriben las instrucciones leídas del archivo que contiene al programa pieza. La instrucción que se encuentre seleccionada en esta lista será la que se procesará mediante los botones situados en el área de control.

El *área de despliegue de coordenadas* muestra en primera instancia las coordenadas relativas al cero de máquina (posición de inicio de los tres ejes de la MMH). Este tipo de coordenadas aparece durante el tiempo en el que se está buscando establecer el cero del programa (punto a partir del cual se empezará con el maquinado mediante el control impuesto por el programa pieza). Una vez establecido el cero del programa se mostrarán ahora las coordenadas relativas respecto al cero del programa, llamadas coordenadas del programa.

En el *área de control de la MMH* se encuentran los dos modos de control soportados por la aplicación, así como una rutina para el regreso de los ejes a su posición de inicio. Para el control por joystick, existen dos botones por cada eje con el fin de trasladar al eje en cuestión en ambos sentidos. En el control por código G y M, el botón “Paso a paso” ejecuta la instrucción seleccionada en el cuadro de lista, mientras que el botón “Todo” ejecuta todas las instrucciones del cuadro de lista en forma secuencial. El botón “Borrar”, empleado únicamente en este último tipo de control, borra el gráfico contenido en el cuadro de imagen y selecciona la primera instrucción del programa pieza.

Los menús que presenta esta pantalla principal son los siguientes:

El menú “archivo” sólo contiene al submenú “Abrir Código-G…”, el cual se habilita sólo después de haber establecido el cero del programa. Al hacer clic sobre el submenú “Abrir Código-G…”, se borra todo gráfico existente en el cuadro de imagen y aparece un explorador de archivos con el fin de que el usuario especifique el archivo que contiene al programa pieza.

En el menú conexión se presenta el submenú “En línea”, que al hacer clic sobre él se habilita o deshabilita el envío de señales hacia la MMH. Cuando el submenú “En línea” tiene a lado una marca, significa que el envío de señales está habilitado.

El menú “salir” sirve para terminar la ejecución de la aplicación, preguntando al usuario si realmente desea salir.

Al hacer clic sobre el menú “Configuración” aparece una nueva ventana (ver figura 3.28) donde se presentan varias opciones para configurar la aplicación. En esta ventana se pueden especificar las direcciones base de los puertos paralelos a los cuales se enviaran las señales de control, el medio período de las señales enviadas, así como una velocidad para el control mediante joystick. La velocidad uno corresponde a 3 [ms] de período medio (que es el máximo posible), la velocidad dos corresponde a 6 [ms] y la velocidad tres corresponde a 10 [ms].

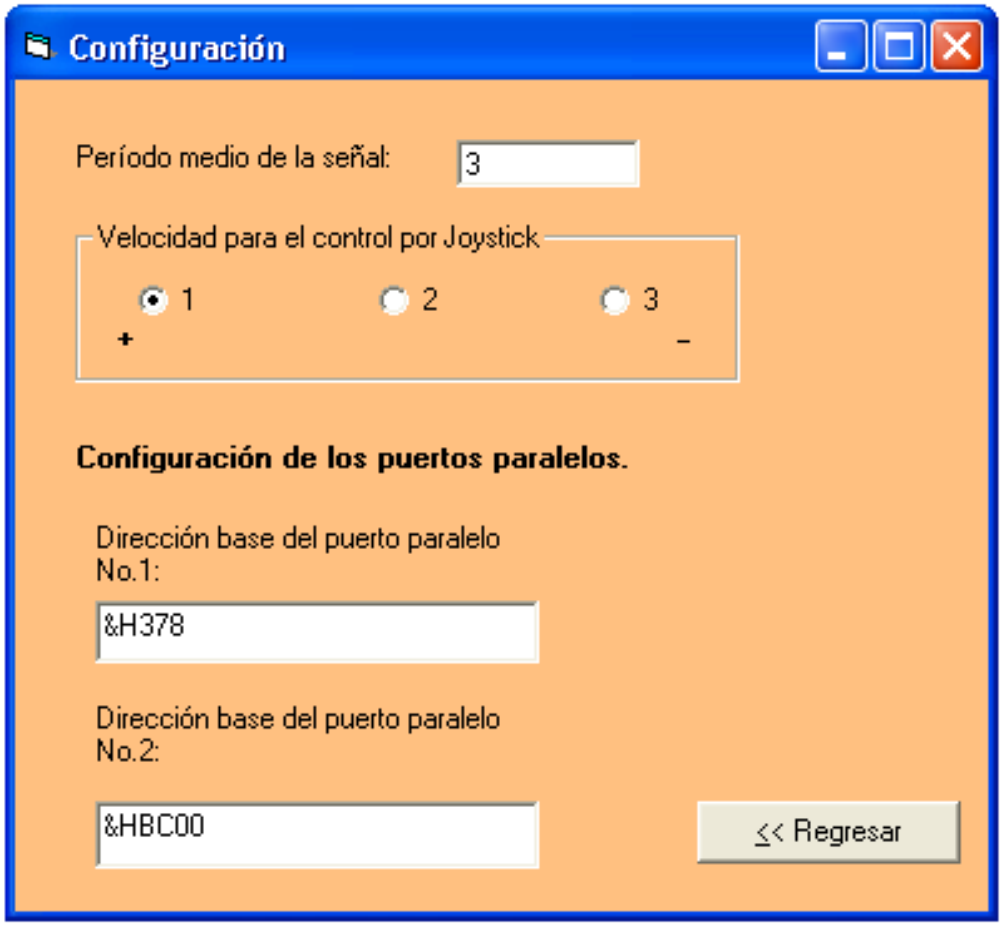

Figura 3.28. Pantalla de configuración.

3.3.2.b. Pasos para realizar un maquinado.

1.- Para realizar algún cambio en los parámetros de configuración, se debe dar clic en el menú “Configuración” y realizar dichos cambios dentro de la ventana “Configuración”, que inmediatamente aparece. Para efectuar los cambios y regresar a la “Pantalla Principal” se debe dar clic en el botón “Regresar”. En caso de que exista algún error en los datos proporcionados, aparecerá alguno de los mensajes de error que se muestran en la figura 3.29.

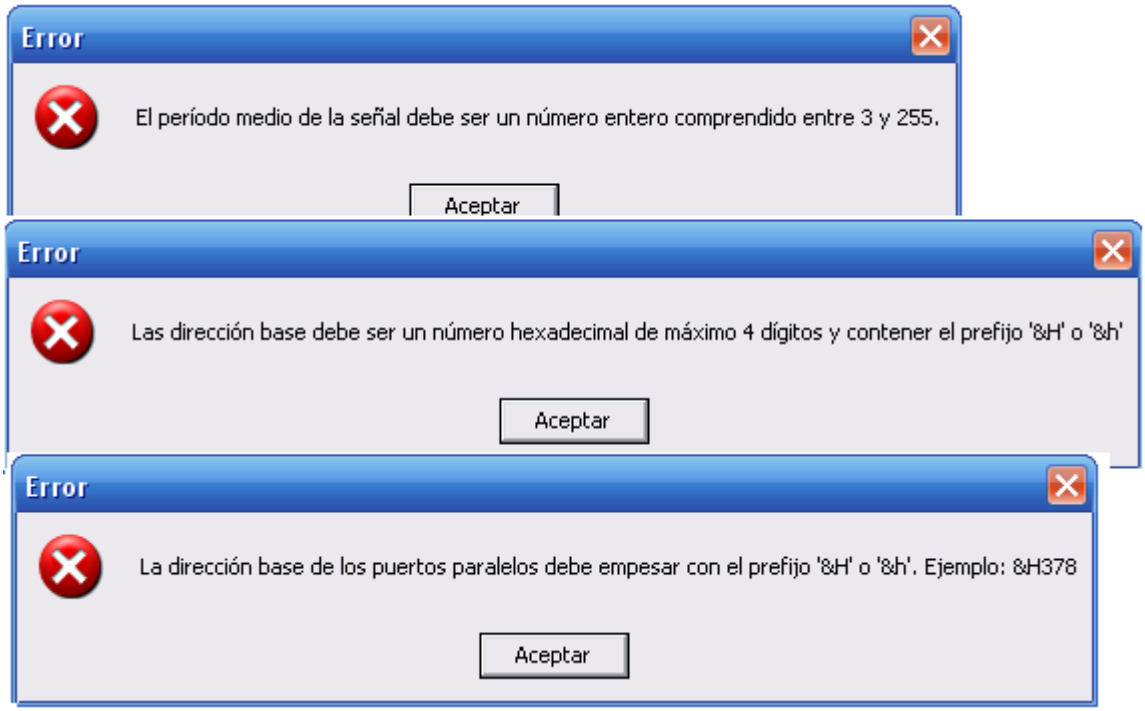

Figura 3.29. Diferentes errores en los datos de configuración proporcionados por el usuario.

2.- Después de haber colocado el material de la pieza a maquinar y de haber realizado las consideraciones previas al maquinado, el primer paso es habilitar el envío de señales, a través del puerto paralelo, mediante la opción “EnLínea” contenida en el menú “Conexión” y dar clic en “Aceptar” cuando aparezca el mensaje de seguridad (ver figura 3.30).

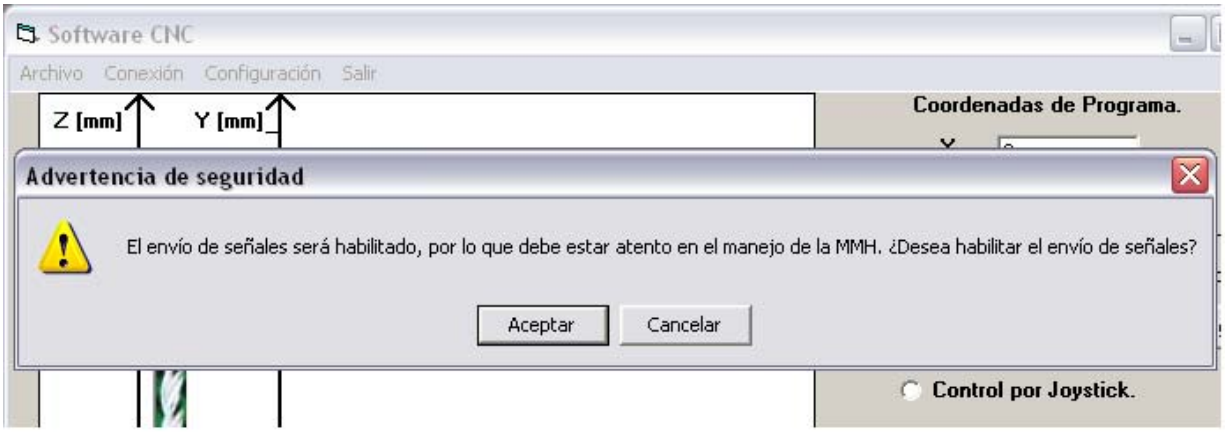

Figura 3.30. Mensaje de seguridad para habilitar el envío de señales hacia la MMH.

3.- Después de haber habilitado el envío de señales se debe ejecutar el procedimiento que regresa a la posición de inicio los tres ejes de la MMH, mediante el botón “Inicio”.

4.- Una vez que ya se ha ejecutado el procedimiento de inicio, automáticamente se habilitan los mandos para el control por joystick y es necesario alcanzar, con dichos mandos, el punto que será el “cero de programa”. Para tener un mejor control y exactitud a la hora de definir el “cero de programa” puede ser pertinente modificar la velocidad de control por joystick en la ventana de “Configuración” de la aplicación.

5.- Se debe establecer el cero del programa mediante el botón “cero de programa”, el cual además de cambiar el tipo de coordenadas mostradas en la pantalla, habilita el submenú “Abrir Código-G…”.

6.- Al dar clic sobre el submenú “Abrir Código-G…”, es necesario especificar la ruta del archivo que contiene al programa pieza (ver figura 3.31). El archivo abierto aparecerá en el cuadro de lista de la “Pantalla Principal”.

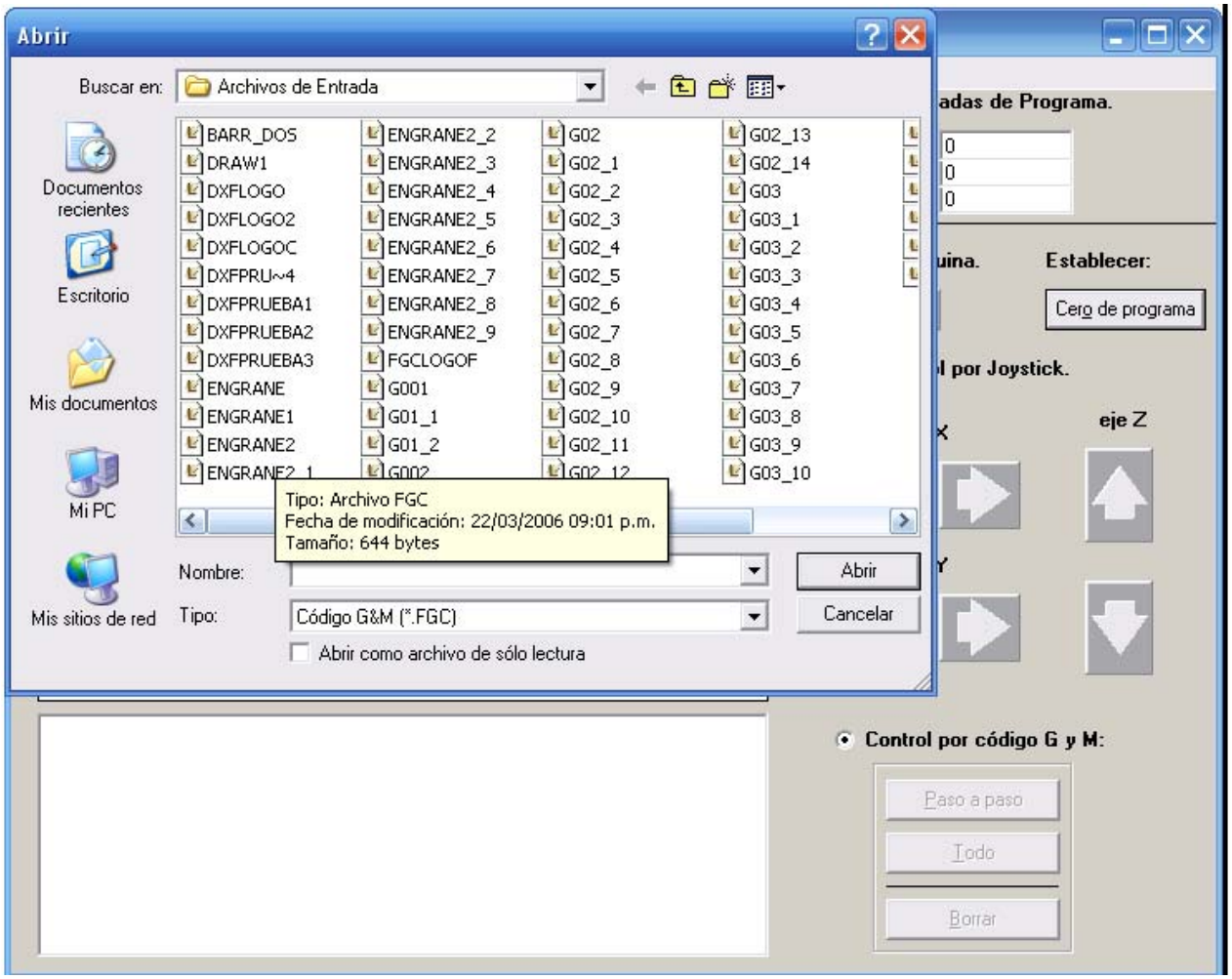

Figura 3.31. Explorador de archivos para seleccionar al programa pieza.

7.- Una vez que ya se tiene el programa pieza contenido en la lista de la pantalla principal, se debe seleccionar la opción de “control mediante código G y M”, para habilitar estos mandos.

8.- En este momento se puede seleccionar de la lista la instrucción a partir de la cual se desea efectuar el maquinado y ejecutar dicha instrucción mediante el botón “Paso a paso”. Si se desean ejecutar todas las instrucciones de forma secuencial hasta alcanzar el final del programa pieza, es necesario oprimir el botón “Todo”.

9.- Finalmente, para terminar la ejecución de la aplicación se debe dar clic en el menú “Salir” y confirmar dicha decisión en la ventana de mensaje “Salir” (ver figura 3.32).

Figura 3.32. Ventana para confirmar que se desea salir de la aplicación.

3.3.3. Detalle de la aplicación a nivel programador.

La figura 3.33 muestra la estructura de formularios y módulos de la aplicación bajo el entorno de la herramienta de desarrollo “Visual Basic 6.0”. Dentro de los formularios se encuentran las ventanas que se despliegan para el funcionamiento de la aplicación. Mientras que en los módulos de la aplicación se encuentran funciones propias de la aplicación, controladores y utilerías.

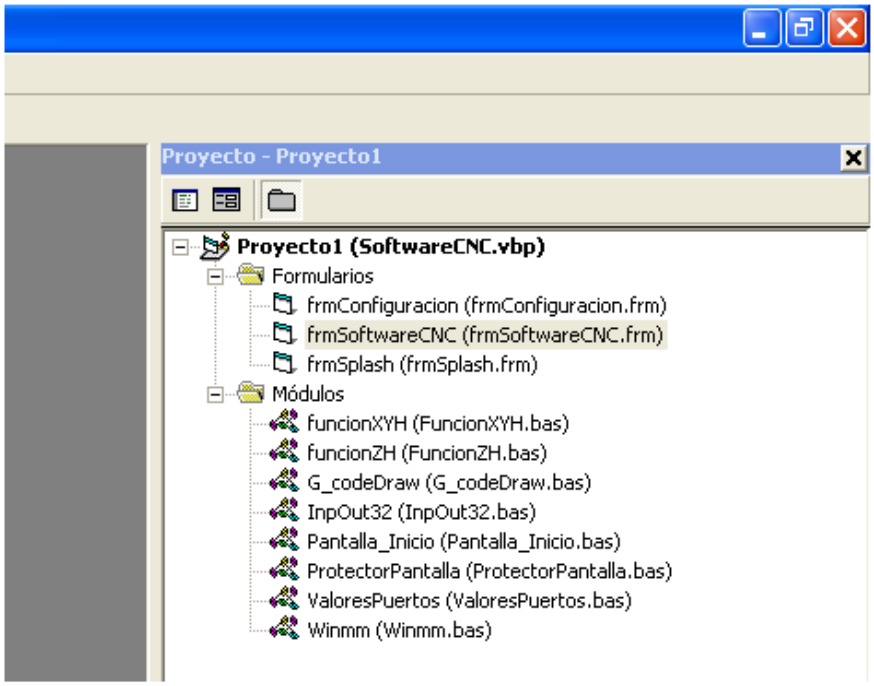

Figura 3.33. Estructura de formularios y módulos de la aplicación.

3.3.3.a. Controladores y utilerías.

En el módulo **"InpOut32"** se encuentran declaradas dos funciones para lectura y escritura del puerto paralelo. Dicha declaración es la siguiente:

*Public Declare Function **Inp** Lib "inpout32.dll" Alias "Inp32" _*
(ByVal PortAddress As Integer) _
As Integer

*Public Declare Sub **Out** Lib "inpout32.dll" Alias "Out32" _*
(ByVal PortAddress As Integer, _
ByVal Value As Integer)

La primera de estas funciones (*Inp*), nos regresa un valor entero que corresponde al valor leído de uno de los subpuertos del puerto paralelo. A esta función se le debe suministrar como parámetro la dirección del subpuerto del cual se desea obtener el dato.

La segunda función (*Out*), nos permite enviar datos a la dirección del subpuerto especificado como parámetro. Este parámetro es un número de tipo entero, donde su correspondiente en binario es el que realmente se envía por cada uno de los pines del puerto paralelo.

Estas dos funciones utilizan una DLL (Dynamic Link Library, Biblioteca de vínculos dinámicos) llamada **"*inpout32.dll*"**[14]. Esta DLL es la que realiza el trabajo de lectura y escritura del puerto paralelo. Dicha librería puede trabajar a nivel kernel

de Windows, con lo cual no existirá ninguna restricción de seguridad, para la lectura y escritura del puerto paralelo, en sistemas operativos Windows NT, 2000 y XP. Sin embargo, esta DLL puede ocuparse también en sistemas operativos Win 9X. El diagrama de su funcionamiento es el mostrado en la figura 3.34.

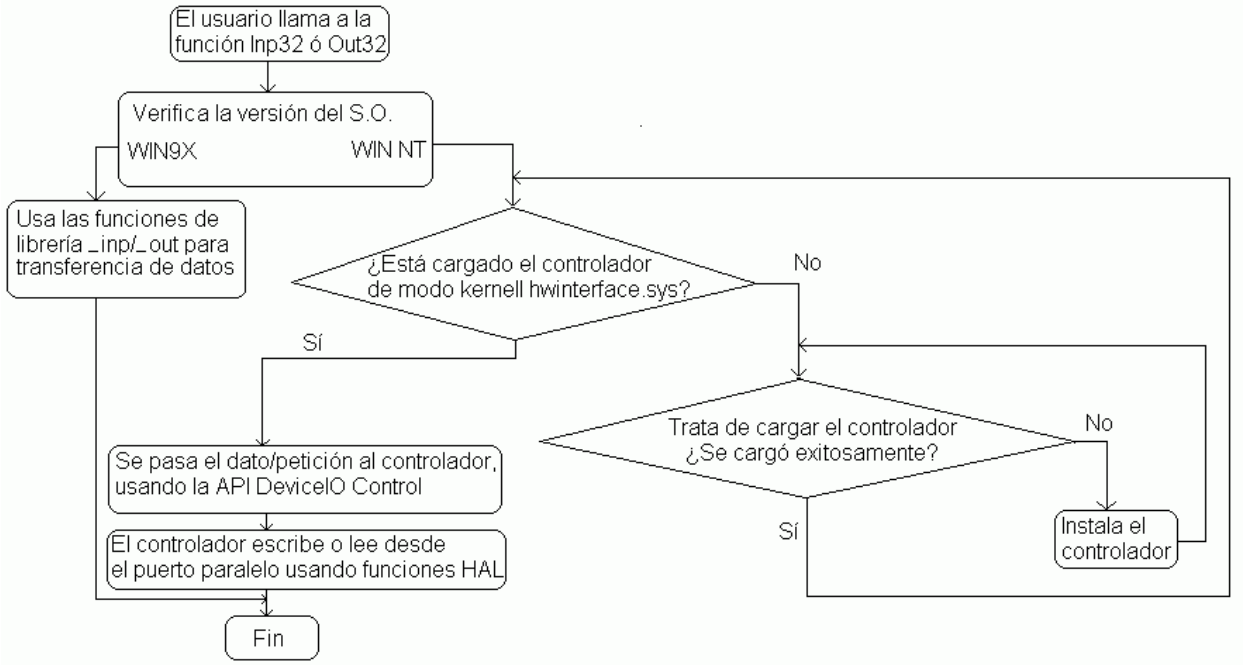

Figura 3.34. Diagrama del funcionamiento de la DLL "inpout32.dll".

Otro de los módulos es **"Winmm"**, en él se encuentran declaradas las funciones que obtienen el tiempo del sistema. Su declaración es la siguiente:

Public Declare Function ***timeGetTime*** *Lib "winmm.dll" () As Long*
Public Declare Function ***timeBeginPeriod*** *Lib "winmm.dll" _*
(ByVal uPeriod As Long) As Long

La función "*timeGetTime( )*" no requiere que se le pase ningún valor como parámetro. Esta función regresa un número (tipo Long) que corresponde al número de milisegundos a partir de que se inició el sistema. La precisión de esta función está en el rango de 1 a 5 milisegundos, dependiendo de las características de la máquina en donde se ejecute. La precisión se debe establecer mediante la función "*timeBeginPeriod*".

"*timeGetTime( )*" es una función que está contenida dentro de la DLL **"*winmm.dll*"**, la cual es una librería multimedia que forma parte del API de Windows. Dentro de esta librería se pueden encontrar, además, funciones para manipular video, CD de audio, midi, archivos de sonido, etc.

En el módulo **"ProtectorPantalla"**, se declara la función *"SystemParametersInfo"*, la cual utiliza la librería ***"user32"*** de Windows con el fin de habilitar y deshabilitar el protector de pantalla del sistema. Deshabilitar el protector de pantalla cuando la

aplicación se está ejecutando es por motivos de seguridad. Con esto el usuario tiene el control en todo momento (al estar visualizando siempre la interfaz) sobre el maquinado. La declaración para esta función es de la siguiente forma:

```
Public Declare Function SystemParametersInfo Lib "user32"_
   Alias "SystemParametersInfoA" (ByVal uAction As Long, ByVal uParam As _
   Long, ByVal lpvParam As Any, ByVal fuWinIni As Long) As Long
```

El protector de pantalla queda deshabilitado al cargarse el formulario **"frmSoftwareCNC"**. Mientras que nuevamente se habilita cuando el usuario sale de la aplicación. La llamada a la función *"SystemParametersInfo" para* des habilitar y habilitar el protector de pantalla, queda de la siguiente forma:

Llamada a la función *"SystemParametersInfo"* para des habilitar el protector de pantalla:

```
SystemParametersInfo SPI_SETSCREENSAVEACTIVE, False, _
   0&, SPIF_SENDWININICHANGE
```

Llamada a la función *"SystemParametersInfo"* para habilitar el protector de pantalla:

```
SystemParametersInfo SPI_SETSCREENSAVEACTIVE, True, _
   0&, SPIF_SENDWININICHANGE
```

donde *SPI_SETSCREENSAVEACTIVE* y *SPIF_SENDWININICHANGE*, son variables establecidas dentro del módulo **"ProtectorPantalla"**.

En el módulo **"G_code_Draw"** se encuentran contenidos los procedimientos que realizan la interpretación gráfica, así como también, las rutinas de cálculo que sirven para el maquinado. Una vez realizados los cálculos necesarios, es en este módulo donde se manda a llamar a las "funciones para el maquinado". Las funciones y procedimientos que están contenidos en este módulo son:

"ponderacionDraw( )".- En este procedimiento se establecen los valores de los ejes X, Y, Z; con base en las cotas obtenidas previamente. Además se obtiene el valor de la variable "ponderacionXY", el cual será un número por el que se multiplicará todo valor X o Y antes de ser graficado, con el fin de adecuar el gráfico a la escala establecida por las cotas.

"angulo(a, b, tipo)".- Esta función es utilizada para calcular un ángulo en radianes de un punto A = (a, b) = (r cos(θ), r sen(θ)). Principalmente, esta función se ocupa para calcular el ángulo de los puntos inicial y final de un arco de circunferencia respecto al centro de dicha circunferencia. Cuando la variable "tipo" es diferente del valor "inicio", el resultado será θ. Y cuando la variable tipo es igual al valor "inicio", el valor obtenido corresponderá al ángulo, en coordenadas polares, del punto –A.

"G00(instrucción)".- Este procedimiento implementa las instrucciones G00 y G01. El parámetro "instrucción" corresponde a una instrucción de código G (G00 o G01). Con dicha instrucción se realizan las rutinas de cálculo y posteriormente se mandan a llamar a las "funciones para el maquinado". Una vez finalizado el maquinado, se refleja este movimiento en el cuadro de imagen de la aplicación.

"G02(instrucción)".- Este procedimiento implementa las instrucciones G02 y G03. En base al parámetro "instrucción" se determina que tipo de instrucción es (G02 o G03) y sus parámetros asociadas, para después efectuar los cálculos necesarios y llamar a las "funciones para el maquinado". Una vez finalizado el maquinado, se refleja este movimiento en el cuadro de imagen de la aplicación.

Los módulos **"funcionXYH"** y **"funcionZH"** contienen la implementación de las "funciones para el maquinado", antes mencionadas. Dentro del módulo **"funcionXYH"** se encuentra además una rutina de compensación del backlash de la MMH, cuya declaración queda de la siguiente forma:

*Public Sub **backlashXY**(eje As String, horario As Boolean)*

Al detectarse un cambio de sentido en cualquiera de los ejes X o Y, el procedimiento *"backlashXY"* es llamado desde el procedimiento *"motorXYH"*, ambos contenidos dentro del módulo **"funcionXYH"**. El procedimiento *"motorXYH"*, el cual implementa el funcionamiento de maquinado en los ejes X y Y, llama a la rutina *"backlashXY"* y le envía como parámetros el sentido en el que se compensará el backlash (segundo parámetro) y el eje en el que se realizará dicha compensación (primer parámetro).

De la misma forma que en el módulo **"funcionXYH"**, existe para el módulo **"funcionZH"** una rutina para compensar el backlash en el eje Z, llamada *"backlashZ"*. Esta rutina sólo recibe como parámetro el sentido en el cual se compensará el backlash en este eje. También, dentro de este módulo, se encuentra el procedimiento *"motorZH"* que es el encargado de realizar el maquinado en el eje Z y de llamar a la rutina *"backlashZ"*, cuando sea necesario.

En el módulo **"ValoresPuertos"**, se encuentra la declaración de las variables de los subpuertos asignados a cada eje de la MMH. También se encuentra el procedimiento *"inicializaPuertos( )"*, el cual, como su nombre lo dice, inicializa los valores de dichas variables y manda una señal en bajo a todos los subpuertos de salida. Estos valores son consultados desde la ventana de "Configuración" y pueden ser modificados por el usuario durante la ejecución de la aplicación.

El módulo **"Pantalla_Inicio"** contiene al procedimiento *"main( )"* el cual muestra por un instante la pantalla de presentación de la aplicación y posteriormente pasa a la "Pantalla Principal". Con este procedimiento se inicia la ejecución de la aplicación.

3.3.3.b. Formularios.

3.3.3.b.1. Formulario “frmConfiguracion”.

El formulario **“frmConfiguracion”** representa a la ventana “Configuración”, que se muestra en la figura 3.35. El botón “Regresar”, además de cambiar de la ventana “Configuración” a la “Pantalla Principal”, manda a llamar al procedimiento *“inicializaPuertos( )”* con el fin de actualizar los valores de los puertos paralelos. Durante la llamada a este procedimiento, se validan los campos para las direcciones base de los puertos paralelos y el campo en el que se establece el período medio de las señales de salida. Las demás opciones de configuración, presentes en este formulario, son consultadas en tiempo de ejecución. El único método dentro de este formulario es *“cmdRegresar_Click( )”*, el cual responde al evento en el que el usuario oprime el botón “Regresar”. Los valores por defecto para las direcciones base de los puertos paralelos, período medio de la señal y velocidad para el control por joystick, son establecidos mediante las propiedades de los objetos (cajas de texto y botones de opción), durante el tiempo de diseño.

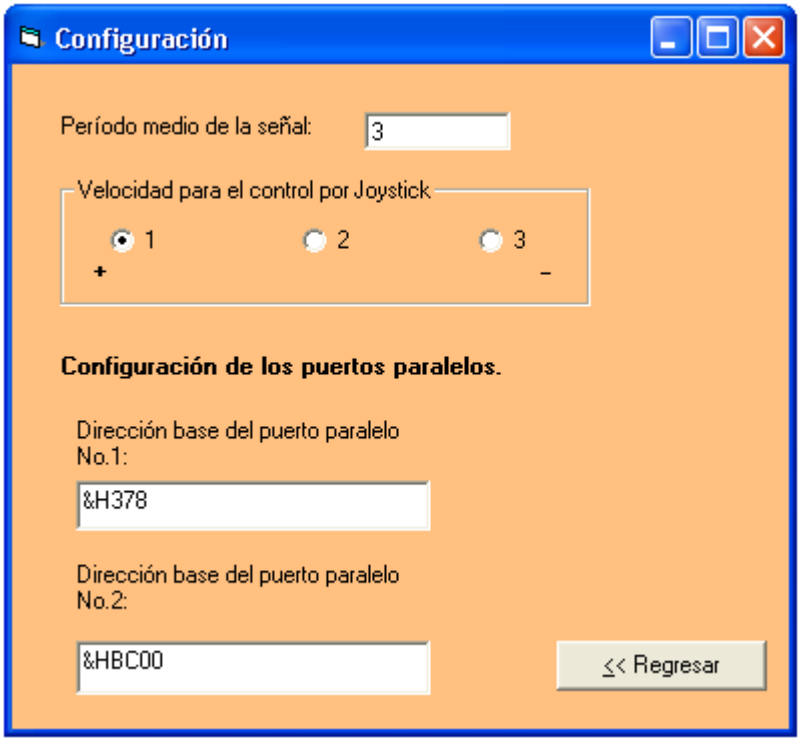

Figura 3.35. Pantalla de configuración.

3.3.3.b.2. Formulario “frmSoftwareCNC”.

El formulario **“frmSoftwareCNC”** representa a la “Pantalla Principal”, que se muestra en la figura 3.36. Este formulario contiene los siguientes métodos:

Form_Load( ).- Este método se ejecuta al cargarse este formulario. Aquí se inicializan algunas variables, se deshabilita el menú “Abrir Código-G…”, se manda

a llamar al procedimiento *"inicializaPuertos"* y se deshabilita el protector de pantalla.

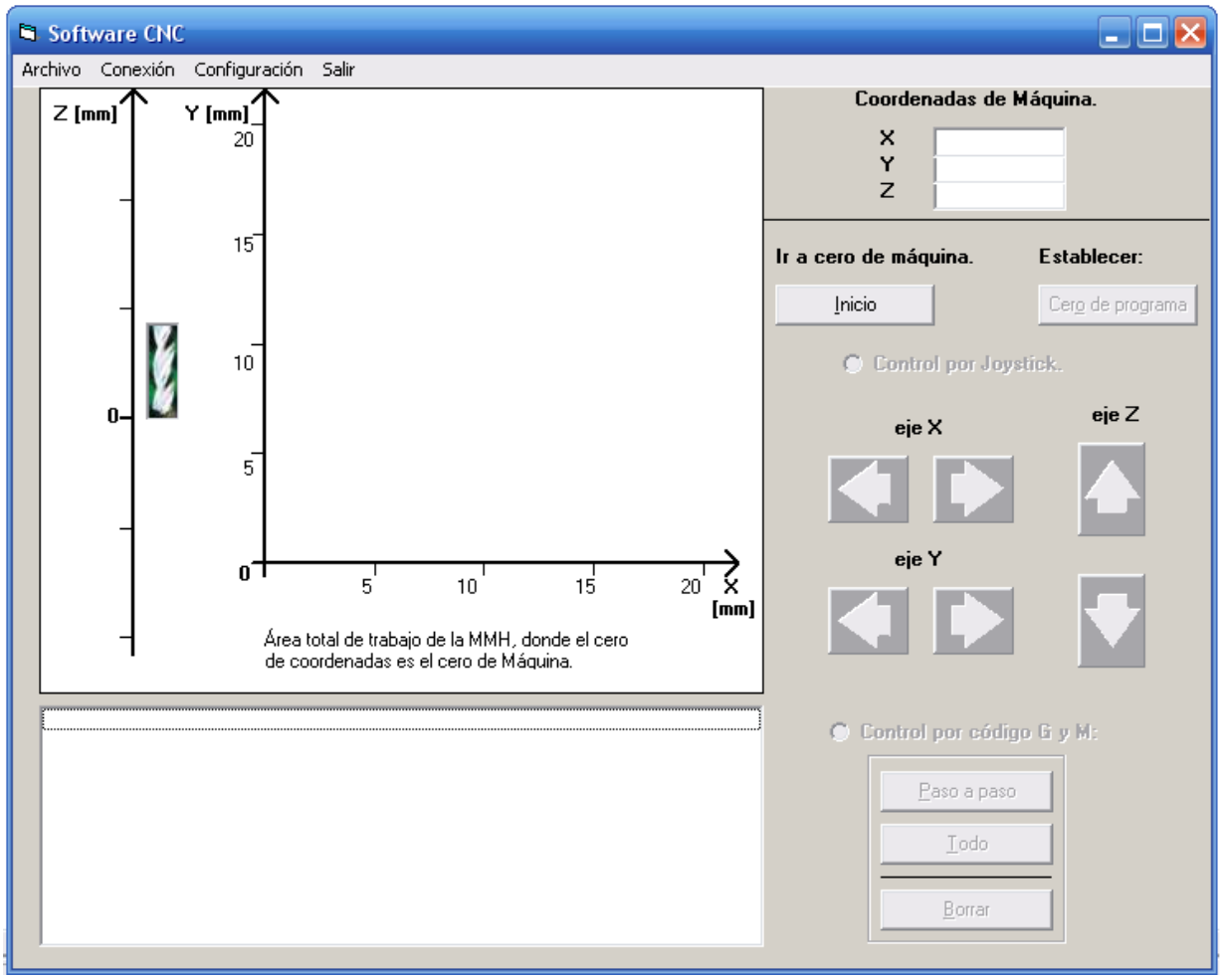

Figura 3.36. Pantalla principal de la aplicación para el control de MMHs.

Métodos asociados al menú:

mnuAbrir_Click( ).- Contiene el código que se ejecuta al momento de dar clic sobre el submenú "Abrir Código-G…", ver figura 3.37. En este método se implementa la lectura del archivo que contiene al programa pieza. Durante la lectura del programa pieza se determinan los valores de las cotas para cada eje; al finalizar la lectura de este archivo, su contenido es copiado al cuadro de lista. Además, se inicializan las principales variables para el maquinado y graficado, los valores de las cajas de texto para las coordenadas de los ejes, y se habilitan o deshabilitan algunos botones.

Figura 3.37. Submenú "Abrir Código-G…".

mnuConfiguracion_Click( ).- Este método oculta el formulario actual y muestra el formulario que contiene a la ventana "Configuración". Este método se ejecuta en el momento de dar clic sobre el menú "Configuración".

mnuEnLinea_Click( ).- Este método se ejecuta al dar clic en el submenú "En línea". Si este submenú no presenta alguna marca ("paloma") significa que el envío de señales no está habilitado, en este caso se mostrará un mensaje de advertencia de seguridad y se le preguntará al usuario si realmente desea habilitar el envío de señales. En caso afirmativo aparecerá una marca ("paloma") en el submenú, ver figura 3.38. Por otro lado, si el envío de señales está habilitado, sólo se despliega un mensaje para informarle al usuario de la acción tomada y se quita la marca ("paloma") del submenú. Los procedimientos y funciones consultan la propiedad "Checked" del submenú "En línea", en tiempo de ejecución, para determinar si el envío de señales está habilitado.

Figura 3.38. Submenú "En línea".

mnuSalir_Click( ).- Este método se ejecuta al dar clic sobre el menú "Salir". En este método se muestra un mensaje en el que se le pregunta al usuario si desea terminar la ejecución de la aplicación, en caso afirmativo se termina la aplicación y queda nuevamente habilitado el protector de pantalla.

Métodos para seleccionar el modo de control:

optCodigo_Click( ).- Este método se ejecuta al dar clic sobre el botón de opción "Control por código G y M". En este método se habilita el submenú "Abrir Código-G…" y el objeto "fraCodigo", el cual contiene los botones para el control mediante código G y M. Además se deshabilitan los objetos que forman parte del control por joystick.

optJoystick_Click( ).- Este método se ejecuta al dar clic sobre el botón de opción "Control por Joystick". En este método se habilitan los mandos para el control por Joystick y se deshabilita "fraCodigo".

Métodos para situar los ejes de la MMH en su posición de inicio y establecer el cero de programa:

cmdHome_Click( ).- Este método responde al evento en el que el usuario oprime el botón "Inicio". Este método contiene la implementación del regreso de los tres ejes a su posición de inicio. Una vez alcanzada la posición de inicio, se habilita el botón "Cero de programa" y el botón de opción para el control por joystick. Se inicializan en cero las coordenadas de los tres ejes y se establece la precisión para la función que devuelve el tiempo del sistema, mediante la sentencia "*timeBeginPeriod* 1". Además se dibujan algunos pixeles en el cuadro de imagen, que serán el punto de partida de los trazos a dibujar.

cmdCeroPrograma_Click( ).- Este método responde al evento clic del botón "Cero de programa". Este método cambia la leyenda de "Coordenadas de máquina" a "Coordenadas de programa", pone en blanco los campos de las coordenadas de los tres ejes y cambia la leyenda de descripción del área gráfica. Además, habilita el botón de opción "Control por código G y M" y ejecuta el evento clic de este botón de opción, el cual a su vez habilita el submenú "Abrir Código-G…".

Métodos asociados con el modo de control mediante código G y M:

Paso_a_paso( ).- Este procedimiento contiene un sentencia "CASE" la cual evalúa una variable que corresponde al código de una instrucción leída desde el cuadro de lista de la "Pantalla Principal". En esta sentencia "CASE" se manda a llamar a la función que corresponda según el código G de la instrucción en curso. A esta función llamada se le pasa como parámetro la instrucción completa que está actualmente siendo procesada. Una vez concluida la ejecución de la función llamada (de haber dibujado y/o maquinado la instrucción en curso), se selecciona la siguiente instrucción de la lista (sin que se procese hasta que el usuario lo indique). Además se modifica la variable "primerTrazo", la cual es consultada por las rutinas de cálculo para tomar algunas consideraciones en caso de que sea el primer trazo realizado. También se modifica la propiedad "Interval" del objeto "timer" (llamado "tmr1"), de este formulario, para el caso en el que se estén procesando varias instrucciones de forma secuencial (mediante la acción del botón "Todo").

cmdstep_Click( ).- Este método responde al evento en el que el usuario oprime el botón “Paso a paso”, el cual procesa la instrucción seleccionada para que sea graficada y/o maquinada. En este método se inicializan algunas variables, se manda a llamar al procedimiento *“ponderacionDraw”* y al procedimiento *“Paso_a_paso”,* además de habilitarse y deshabilitarse algunos botones.

cmdTodo_Click( ).- Este método responde al evento en el que el usuario oprime el botón “Todo”, el cual ejecuta varias instrucciones de código G de forma secuencial hasta alcanzar el final del programa pieza. En este método se inicializan algunas variables y se manda a llamar al procedimiento *“ponderacionDraw”*. Se establece el valor de la variable booleana “todoEnLinea”, con la cual se determina que valor tomará la propiedad “Interval” del control “timer” de este formulario. Cuando “todoEnLinea” toma el valor “true”, indica que sí se están enviando señales de control a la MMH; en este caso la propiedad “Interval” del control “timer” toma un valor mayor que cero, sólo para que se ejecute el método del control “timer”, el cual invocará al procedimiento *“Paso_a_paso( )”* y pondrá el valor de “Interval” a cero. En caso de que “todoEnLinea” tenga el valor “false” el método del control “timer” se ejecutará periódicamente cada 25 milisegundos, sólo para graficar las instrucciones procesadas.

cmdBorrar_Click( ).- Este método responde al evento en el que el usuario oprime el botón “Borrar”. Este método borra el gráfico actual, la graduación de los ejes y coloca la imagen de la herramienta de corte en el cero del eje Z. Inicializa algunas variables y selecciona la primera instrucción de la lista.

tmr1_Timer( ).- Este método manda a llamar al procedimiento *“Paso_a_paso”*. Se puede ejecutar cada vez que finalice la ejecución del procedimiento *“Paso_a_paso”*, en el caso de que la variable “todoEnLinea” sea igual a “true”; o bien cada 25 milisegundos, cuando “todoEnLinea” sea igual a “false”. El control “timer”, quien es el que ejecuta este método, se utiliza para implementar el graficado y/o maquinado de varias instrucciones de código numérico.

Estos métodos asociados con el modo de control mediante código G y M son, en su mayoría, activados por los objetos mostrados en la figura 3.39.

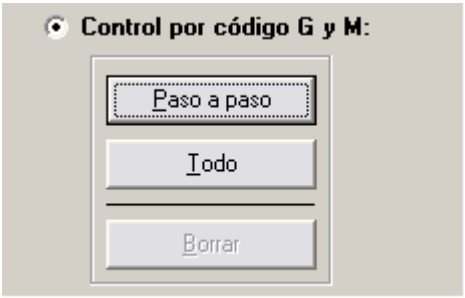

Figura 3.39. Botones que activan los métodos asociados con el modo de control mediante código G y M.

Métodos asociados con el modo de control por joystick:

cmdXadelante_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método responde al evento en el que el usuario mantiene oprimido el botón del ratón cuando el puntero del mismo, se encuentra posicionado sobre el botón que tiene una flecha hacia la derecha (ubicado dentro de los mandos de control por joystick para el eje X). Durante este evento se envían señales por el puerto paralelo para mover el eje X hacia adelante. El envío de señales se interrumpe en el momento en el que el usuario deja de oprimir el botón del ratón. También, aparece un mensaje dentro de una etiqueta que le indica al usuario que se están enviando señales por el puerto paralelo.

cmdXadelante_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método se ejecuta en el momento en el que el usuario deja de oprimir el botón del ratón, después de que se mantuvo oprimido cuando estaba posicionado sobre el botón que tiene una flecha hacia la derecha (ubicado dentro de los mandos de control por joystick para el eje X). Este método borra el mensaje que aparece dentro de una etiqueta, de que se están enviando señales al puerto paralelo.

cmdXatras_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método responde al evento en el que el usuario mantiene oprimido el botón del ratón cuando el puntero del mismo, se encuentra posicionado sobre el botón que tiene una flecha hacia la izquierda (ubicado dentro de los mandos de control por joystick para el eje X). Durante este evento se envían señales por el puerto paralelo para mover el eje X hacia atrás. El envío de señales se interrumpe en el momento en el que el usuario deja de oprimir el botón del ratón. También, aparece un mensaje dentro de una etiqueta que le indica al usuario que se están enviando señales por el puerto paralelo.

cmdXatras_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método se ejecuta en el momento en el que el usuario deja de oprimir el botón del ratón, después de que se mantuvo oprimido cuando estaba posicionado sobre el botón que tiene una flecha hacia la izquierda (ubicado dentro de los mandos de control por joystick para el eje X). Este método borra el mensaje que aparece dentro de una etiqueta, de que se están enviando señales al puerto paralelo.

cmdYadelante_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método responde al evento en el que el usuario mantiene oprimido el botón del ratón cuando el puntero del mismo, se encuentra posicionado sobre el botón que tiene una flecha hacia la derecha (ubicado dentro de los mandos de control por joystick para el eje Y). Durante este evento se envían señales por el puerto paralelo para mover el eje Y hacia adelante. El envío de señales se interrumpe en el momento en el que el usuario deja de oprimir el botón del ratón. También, aparece un mensaje dentro de una etiqueta que le indica al usuario que se están enviando señales por el puerto paralelo.

cmdYadelante_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método se ejecuta en el momento en el que el usuario deja de oprimir el botón del ratón, después de que se mantuvo oprimido cuando estaba posicionado sobre el botón que tiene una flecha hacia la derecha (ubicado dentro de los mandos de control por joystick para el eje Y). Este método borra el mensaje que aparece dentro de una etiqueta, de que se están enviando señales al puerto paralelo.

cmdYatras_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método responde al evento en el que el usuario mantiene oprimido el botón del ratón cuando el puntero del mismo, se encuentra posicionado sobre el botón que tiene una flecha hacia la izquierda (ubicado dentro de los mandos de control por joystick para el eje Y). Durante este evento se envían señales por el puerto paralelo para mover el eje Y hacia atrás. El envío de señales se interrumpe en el momento en el que el usuario deja de oprimir el botón del ratón. También, aparece un mensaje dentro de una etiqueta que le indica al usuario que se están enviando señales por el puerto paralelo.

cmdYatras_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método se ejecuta en el momento en el que el usuario deja de oprimir el botón del ratón, después de que se mantuvo oprimido cuando estaba posicionado sobre el botón que tiene una flecha hacia la izquierda (ubicado dentro de los mandos de control por joystick para el eje Y). Este método borra el mensaje que aparece dentro de una etiqueta, de que se están enviando señales al puerto paralelo.

cmdZabajo_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método responde al evento en el que el usuario mantiene oprimido el botón del ratón cuando el puntero del mismo, se encuentra posicionado sobre el botón que tiene una flecha hacia abajo (ubicado dentro de los mandos de control por joystick para el eje Z). Durante este evento se envían señales por el puerto paralelo para mover el eje Z hacia la pieza o material que se desea manufacturar. El envío de señales se interrumpe en el momento en el que el usuario deja de oprimir el botón del ratón. También, aparece un mensaje dentro de una etiqueta que le indica al usuario que se están enviando señales por el puerto paralelo.

cmdZabajo_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método se ejecuta en el momento en el que el usuario deja de oprimir el botón del ratón, después de que se mantuvo oprimido cuando estaba posicionado sobre el botón que tiene una flecha hacia abajo (ubicado dentro de los mandos de control por joystick para el eje Z). Este método borra el mensaje que aparece dentro de una etiqueta, de que se están enviando señales al puerto paralelo.

cmdZarriba_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método responde al evento en el que el usuario mantiene oprimido el botón del ratón cuando el puntero del mismo, se encuentra posicionado sobre el

botón que tiene una flecha hacia arriba (ubicado dentro de los mandos de control por joystick para el eje Z). Durante este evento se envían señales por el puerto paralelo para alejar el eje Z de la pieza o material que se desea manufacturar. El envío de señales se interrumpe en el momento en el que el usuario deja de oprimir el botón del ratón. También, aparece un mensaje dentro de una etiqueta que le indica al usuario que se están enviando señales por el puerto paralelo.

cmdZarriba_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single).- Este método se ejecuta en el momento en el que el usuario deja de oprimir el botón del ratón, después de que se mantuvo oprimido cuando estaba posicionado sobre el botón que tiene una flecha hacia arriba (ubicado dentro de los mandos de control por joystick para el eje Z). Este método borra el mensaje que aparece dentro de una etiqueta, de que se están enviando señales al puerto paralelo.

Estos métodos asociados con el modo de control por joystick son activados por los objetos mostrados en la figura 3.40.

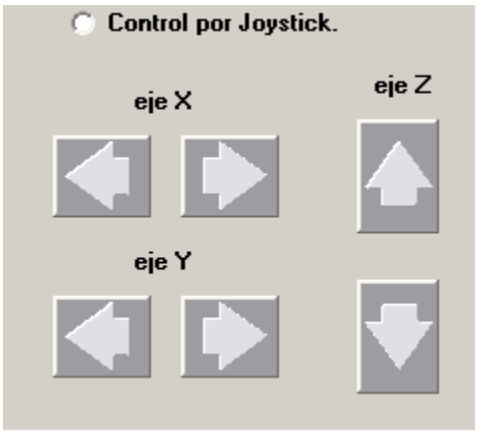

Figura 3.40. Botones que activan los métodos asociados con el modo de control por joystick.

3.3.3.b.3. Formulario “frmSplash”.

El formulario **“frmSplash”** representa a la “pantalla de presentación” de la aplicación, que se muestra en la figura 3.41. Este formulario contiene los siguientes métodos:

Form_KeyPress(KeyAscii As Integer).- Al oprimir cualquier tecla del teclado, este método se ejecuta, obteniéndose como resultado que la “pantalla de presentación” desaparezca y que se muestre la “Pantalla Principal”.

Form_Load( ).- Este método se ejecuta al cargarse el formulario, y en él se inicializa la variable “contador” y se establece la propiedad “Interval” del objeto “timer”.

Frame1_Click( ).- Al dar clic sobre el objeto “frame” de la “pantalla de presentación” (este frame ocupa casi la totalidad de la “pantalla de presentación”), este método se ejecuta, obteniéndose como resultado que la “pantalla de presentación” desaparezca y que se muestre la “Pantalla Principal”.

tmrUnload_Timer( ).- Este es un método de un objeto “timer” que es ejecutado de forma periódica. La frecuencia con que se ejecuta este método está dada por la propiedad “Interval”. El método *Form_Load( )* establece esta propiedad con el valor 1 (un milisegundo). La primera vez que se ejecuta este método se vuelve a establecer la propiedad “Interval” pero ahora para que se ejecute después de cuatro segundos (4000 milisegundos) y se incrementa la variable “contador”. Después de 4 segundos se vuelve a ejecutar este método y mediante la variable contador se determina que el código a ejecutarse será el que establezca a cero la propiedad “Interval”, descargue la “pantalla de presentación” y muestre la “Pantalla Principal”.

Figura 3.41. Pantalla de presentación de la aplicación.

En el siguiente capítulo se presentan diversas pruebas de maquinados utilizando este sistema de control.

Capítulo 4

PRUEBAS Y RESULTADOS

En el capítulo tres se describió el sistema propuesto para el control de una micromáquina herramienta construida en el Laboratorio de Micromecánica y Mecatrónica del CCADET, UNAM; y particularmente, del software de control. También en dicho capítulo, se mencionaron algunas pruebas que se realizaron a este software.

En este capítulo cuatro, se presentan las pruebas que se realizaron de algunos maquinados. Primero se presentan los maquinados de figuras simples, donde el movimiento de la herramienta de corte se realiza en un sólo eje. Posteriormente se presentan maquinados que realizan una interpolación en dos ejes. Y finalmente, los maquinados cuyo número de instrucciones de código numérico aumenta. Así también, se analizan los resultados de dichas pruebas.

4.1. Pruebas realizadas.

Las pruebas que a continuación se describen, se realizaron con el sistema completo (una PC y el software de control, una etapa de potencia y la MMH). En al figura 4.1 se muestra el sistema completo justo antes de efectuar un maquinado.

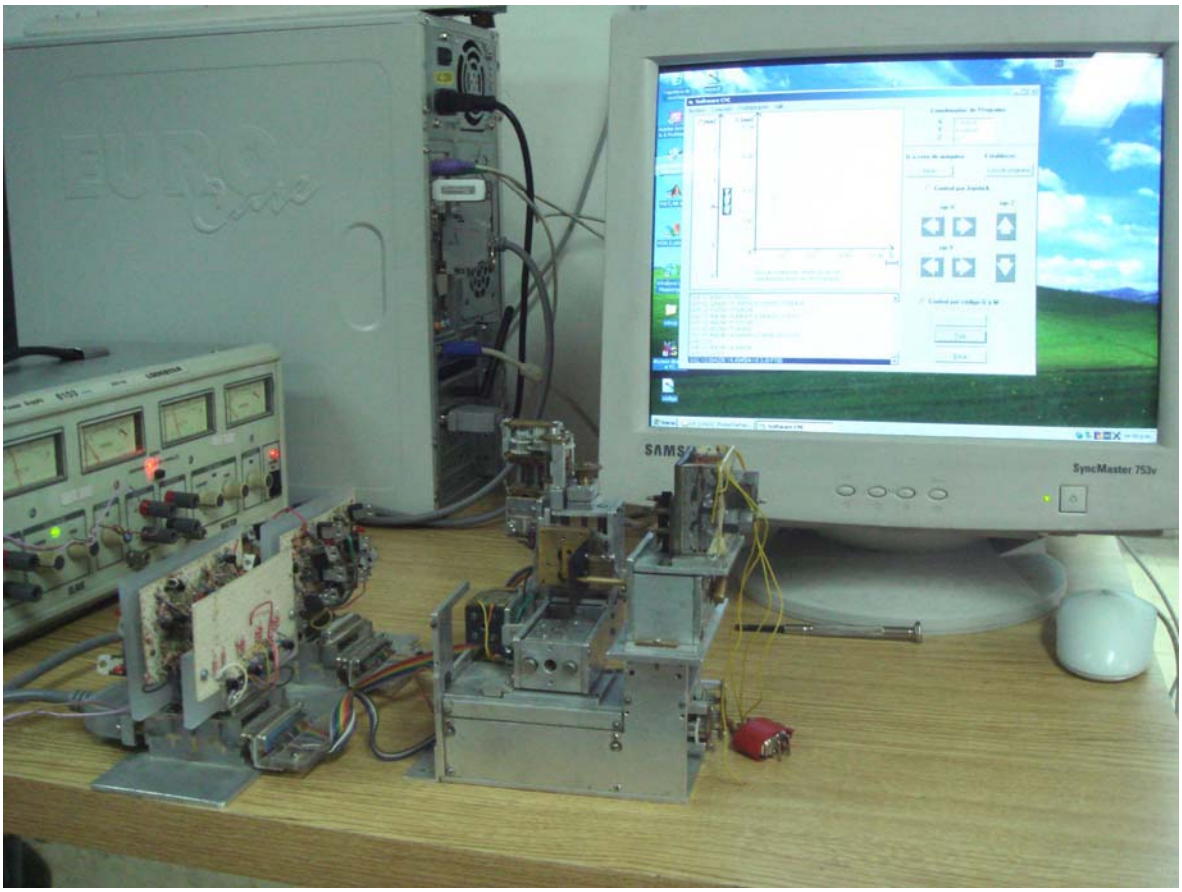

Figura 4.1. Imagen del sistema justo antes de realizar un maquinado.

Las características de la PC en la cual se ejecutó el software de control son las siguientes:

- Sistema operativo: Windows XP.
- Procesador: Pentium 4 a 2.8 GHz.
- Memoria RAM: 512 MB.
- Dos puertos paralelos.

Al igual que en las pruebas presentadas en el apartado 3.3.1.c.4.1.2, para estos maquinados se utilizó como material de trabajo unas láminas de latón delgadas bañadas con tinta; y como herramienta de corte, se utilizó una punta de latón. Durante los maquinados la punta de latón hace contacto con la lámina, de tal forma que sólo marca la trayectoria seguida por la herramienta al quitar la tinta que cubre a la lámina. Para estas pruebas, la MMH opera como fresadora.

Las mediciones de estos maquinados se realizaron en el Laboratorio de Metrología del CCADET, UNAM. El instrumento con el que se midieron estos maquinado fue un comparador óptico marca “Nikon Profile Projector” modelo V-16D, el cual se muestra en la figura 4.2. Este comparador óptico tiene una resolución de 1 [μm].

Figura 4.2. Comparador óptico marca "Nikon Profile Projector" modelo V-16D perteneciente al Laboratorio de Metrología del CCADET, UNAM.

4.1.1. Maquinado simple (figuras conformadas por líneas que son maquinadas un eje a la vez).

Las figuras que se maquinaron bajo este esquema de funcionamiento fueron: un cuadrado de 10 [mm] x 10 [mm], un rectángulo de 10 [mm] en el eje X por 5 [mm] en el eje Y, y un rectángulo de 3 [mm] en el eje X por 10 [mm] en el eje Y; las figuras 4.3, 4.4 y 4.5, corresponden a estos maquinados, respectivamente.

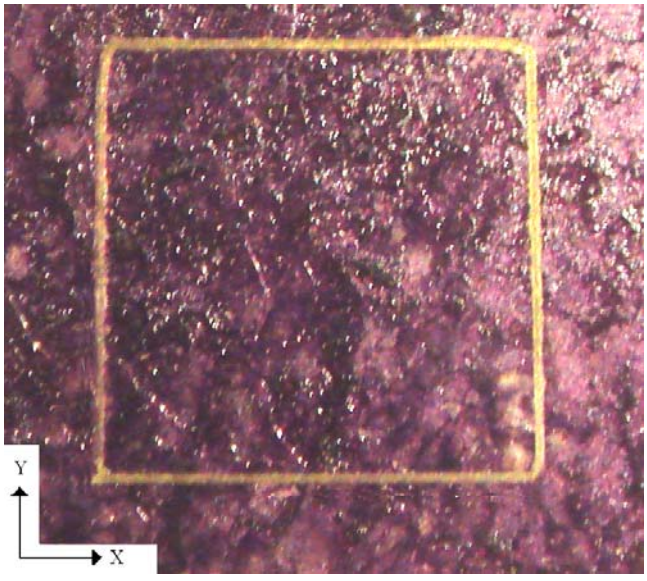

Figura 4.3. Maquinado de un cuadrado de 10 [mm] por 10 [mm].

Las mediciones obtenidas para este cuadrado fueron:

Longitud en el eje X = 9.7 [mm]
Longitud en el eje Y = 9.862 [mm]

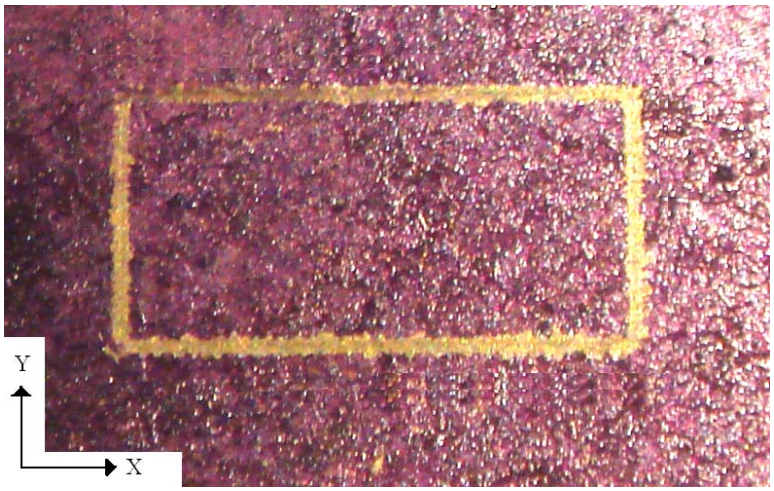

Figura 4.4. Maquinado de un rectángulo de 10 [mm] en el eje X por 5 [mm] en el eje Y.

Las mediciones obtenidas para este rectángulo fueron:

Longitud en el eje X = 9.897 [mm]
Longitud en el eje Y = 4.906 [mm]

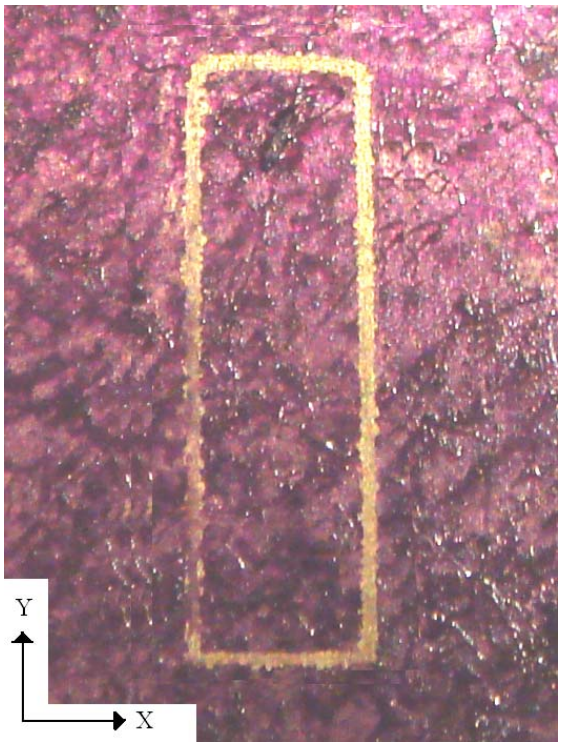

Figura 4.5. Maquinado de un rectángulo de 3 [mm] en el eje X por 10 [mm] en el eje Y.

Las mediciones obtenidas para este rectángulo fueron:

Longitud en el eje X = 2.902 [mm]
Longitud en el eje Y = 9.775 [mm]

El tiempo de duración de los maquinados de estas piezas se muestra en la tabla 4.1. Para estos tres maquinados se estableció el mismo periodo medio de la señal.

Tipo de pieza	Duración del maquinado	Periodo medio de la señal
Cuadrado de 10[mm] x 10[mm]	2 minutos con 53 segundos	3 [ms]
Rectángulo de 10[mm] x 5[mm]	2 minutos con 10 segundos	3 [ms]
Rectángulo de 3[mm] x 10[mm]	1 minuto con 53 segundos	3 [ms]

Tabla 4.1. Tiempo de duración de maquinados simples.

4.1.2. Maquinado compuesto (realiza interpolación en dos ejes).

Para este tipo de maquinados, en los cuales se realiza una interpolación en dos ejes, se maquinó un rectángulo de 10 [mm] en el eje X por 5 [mm] en el eje Y, cuyos vértices son unidos mediante dos líneas inclinadas; ver figura 4.6. También, se maquinó una circunferencia con diámetro igual a 10 [mm], la cual se muestra en la figura 4.7.

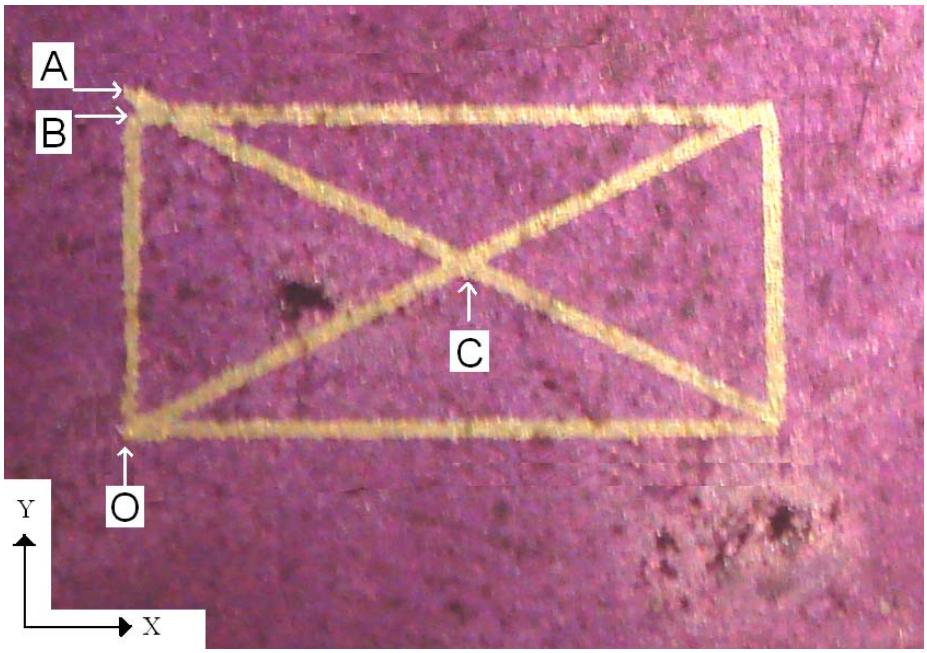

Figura 4.6. Maquinado de un rectángulo de 10 [mm] en el eje X por 5 [mm] en el eje Y, cuyos vértices son unidos mediante dos líneas inclinadas.

La distancia obtenida entre el punto A y B fue de 0.233 [mm]. Mientras que el punto C, está situado en las coordenadas (5.118, 2.562) [mm], respecto al punto O (0, 0).

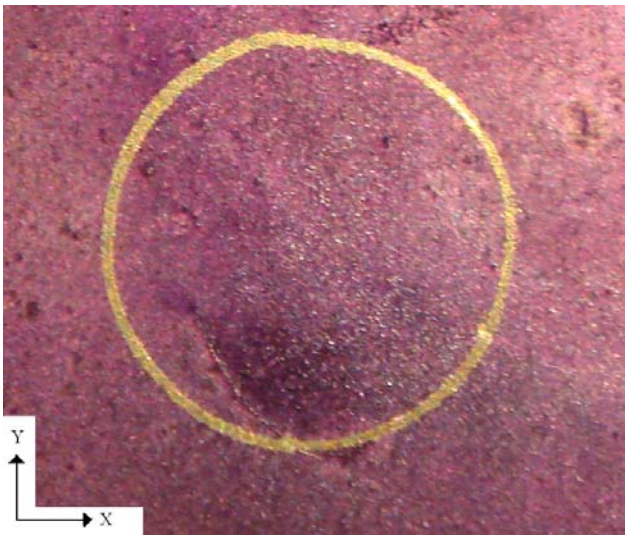

Figura 4.7. Maquinado de una circunferencia con diámetro igual a 10 [mm], maquinadas con un periodo medio de la señal de 40 [ms].

El diámetro horizontal obtenido para esta circunferencia fue de 9.083 [mm]. Mientras que el diámetro vertical fue de 9.178 [mm].

Para el caso del rectángulo cuyos vértices son unidos mediante dos líneas inclinadas, se estableció el periodo medio de la señal con un valor de 3 [ms] para las líneas que son maquinadas en un solo eje; mientras que para las líneas inclinadas, dicho periodo se estableció con un valor de 40 [ms]. Por otro lado, el periodo medio de la señal para el maquinado de la circunferencia, se estableció en 40 [ms].

Se realizó un segundo maquinado para esta circunferencia, pero estableciendo ahora un periodo medio de la señal de 20 [ms]. El resultado obtenido se muestra en la figura 4.8.

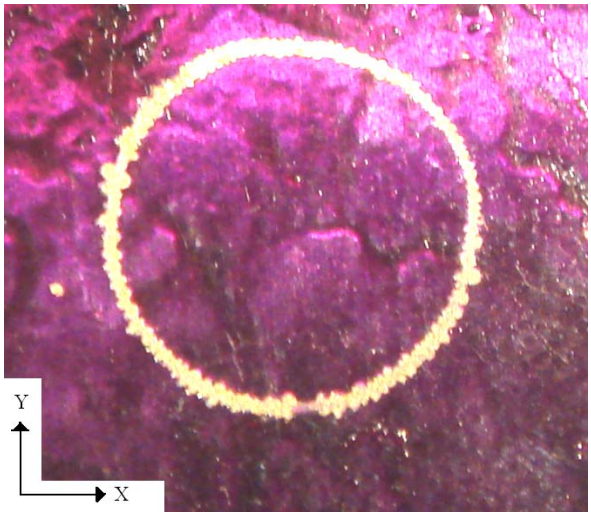

Figura 4.8. Maquinado de una circunferencia con diámetro igual a 10 [mm], maquinadas con un periodo medio de la señal de 20 [ms].

El diámetro horizontal obtenido para esta segunda circunferencia fue de 9.004[mm]. Mientras que el diámetro vertical fue de 9.123 [mm].

El tiempo de duración de los maquinados de estas piezas se muestra en la tabla 4.2.

Tipo de pieza	Duración del maquinado	Periodo medio de la señal
Rectángulo con vértices unidos	16 minutos con 56 segundos	40 [ms] (para interpolación en dos ejes) 3 [ms] (para maquinados en un solo eje)
Circunferencia	10 minutos con 58 segundos	20 [ms]
Circunferencia	21 minutos con 40 segundos	40 [ms]

Tabla 4.2. Tiempo de duración de maquinados compuestos.

Con los maquinados de estas circunferencias se puede observar que al establecer un valor mayor para el periodo medio de la señal, se obtiene una circunferencia más cercana a la deseada. Sin embargo, el tiempo del maquinado aumenta.

4.1.3. Maquinado de figuras más elaboradas (se interpreta un conjunto de instrucciones de código numérico creado mediante software CAD y CAM).

Los maquinados que se realizaron con un mayor número de instrucciones de código numérico, interpretadas por el software de control, fueron: el logotipo del Laboratorio de Micromecánica y Mecatrónica del CCADET, UNAM; y el escudo de los PUMAS de la Universidad Nacional Autónoma de México. Para este tipo de maquinados, el proceso que se siguió para la generación del código numérico, consistió en dibujar la figura correspondiente al maquinado en un software de diseño asistido por computadora (CAD), en este caso se utilizó “SolidWorks”. Este dibujo se guarda en formato DXF (Drawing Interchange Format), que puede ser leído por un software de manufactura asistida por computadora (CAM). El segundo paso consistió en abrir dicho archivo con el software “FlashCut”, el cual es un software CAM, y por medio de este software generar el conjunto de instrucciones de código numérico. A continuación se presentan dichos maquinados.

4.1.3.a. Logotipo del Laboratorio de Micromecánica y Mecatrónica del CCADET, UNAM.

El primer paso, para el maquinado del logotipo del LMM, fue dibujar dicho logotipo dentro de la aplicación “SolidWorks” y guardarlo en un archivo con formato DXF. En la figura 4.9 se muestra el resultado obtenido.

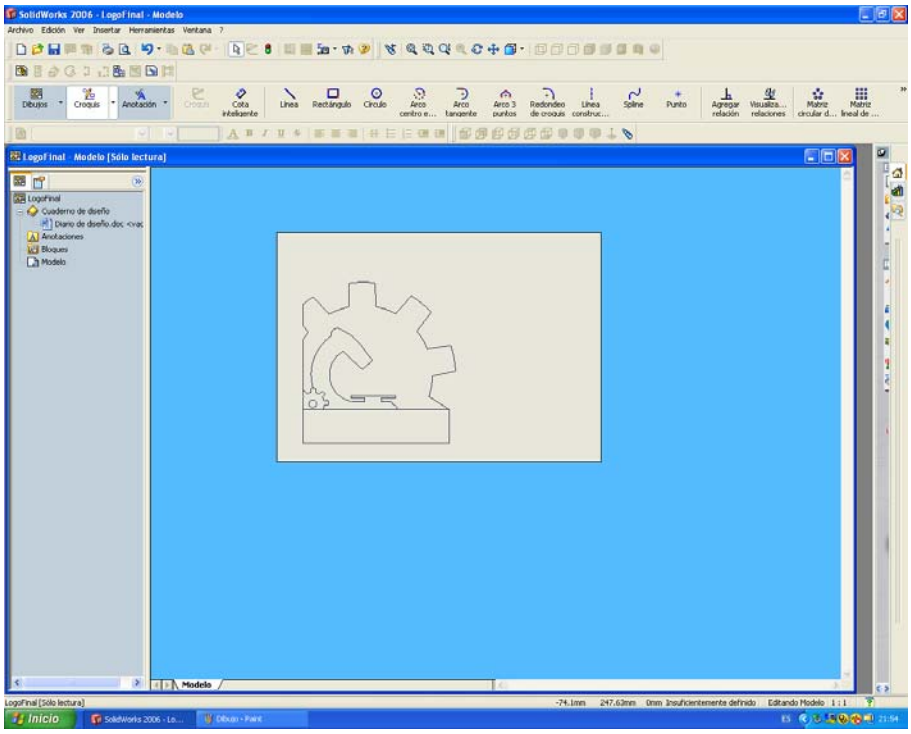

Figura 4.9. Logotipo del LMM, en formato DXF, creado mediante la aplicación "SolidWorks".

El segundo paso consistió en importar el archivo DXF, creado en el paso anterior, al software "FlashCut". Al importar este archivo DXF, "FlashCut" genera automáticamente el código numérico asociado a dicho archivo (ver figura 4.10).

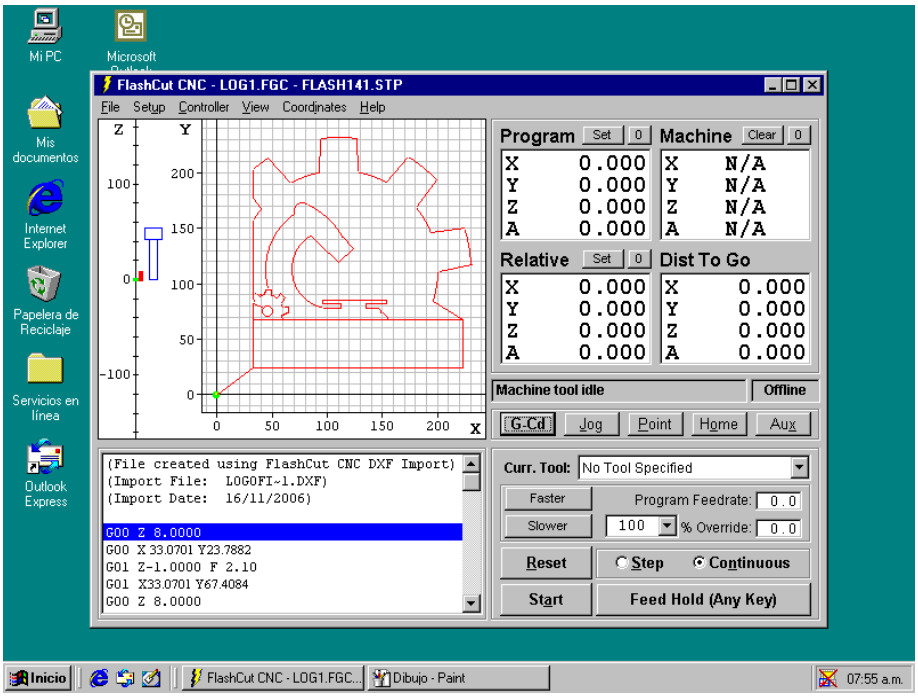

Figura 4.10. Código numérico, generado por "FlashCut", para maquinar el logotipo del LMM.

El tercer y último paso consistió en maquinar el logotipo utilizando la interfaz desarrollada (ver figura 4.11), la cual interpreta el código numérico obtenido en el paso anterior y genera las señales de control necesarias que requiere la MMH para efectuar el maquinado. El maquinado de este logotipo es el que se muestra en la figura 4.12.

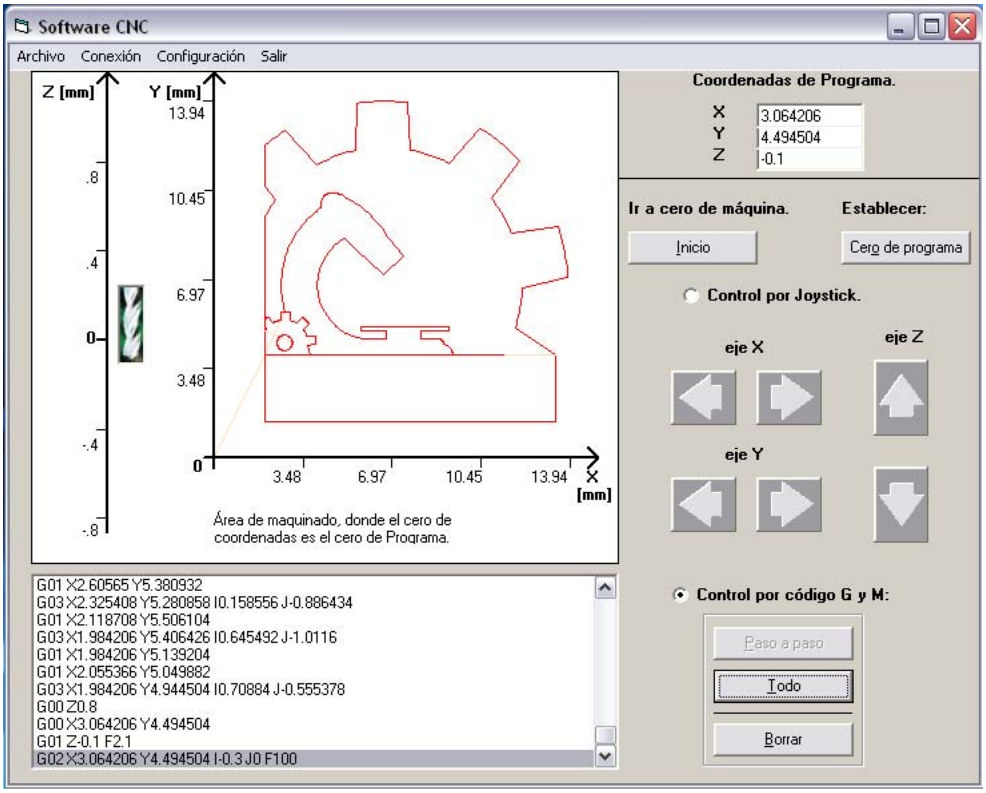

Figura 4.11. Ejecución de las instrucciones para el maquinado del logotipo del LMM, con la interfaz desarrollada.

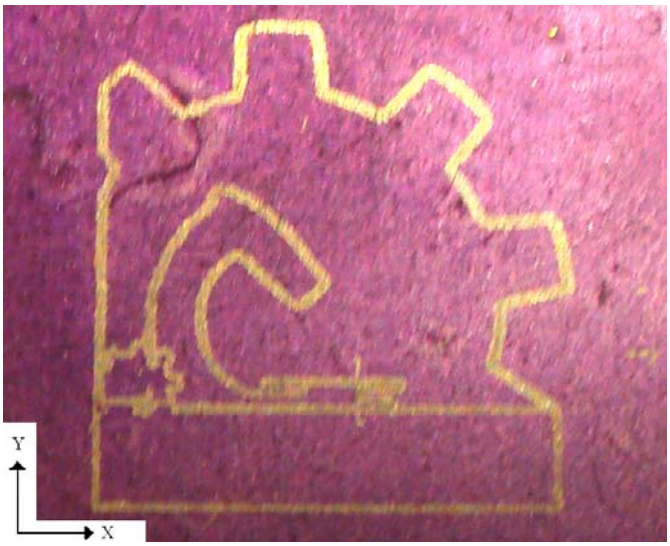

Figura 4.12. Maquinado del logotipo del LMM, utilizando como elemento de control la interfaz desarrollada.

El tiempo requerido para manufacturar esta pieza fue de 45 minutos, aproximadamente; estableciendo el periodo medio de la señal con un valor de 20 [ms] en todo el proceso.

4.1.3.b. Escudo de los PUMAS de la Universidad Nacional Autónoma de México.

Para el maquinado del escudo de los PUMAS se siguieron los mismos pasos que en el maquinado del logotipo del LMM. El archivo DXF generado con la aplicación “SolidWorks” para este escudo, es el que se muestra en la figura 4.13. Mientras que el código numérico generado a partir de este archivo DXF, utilizando “FlashCut”, es el que se muestra en la figura 4.14.

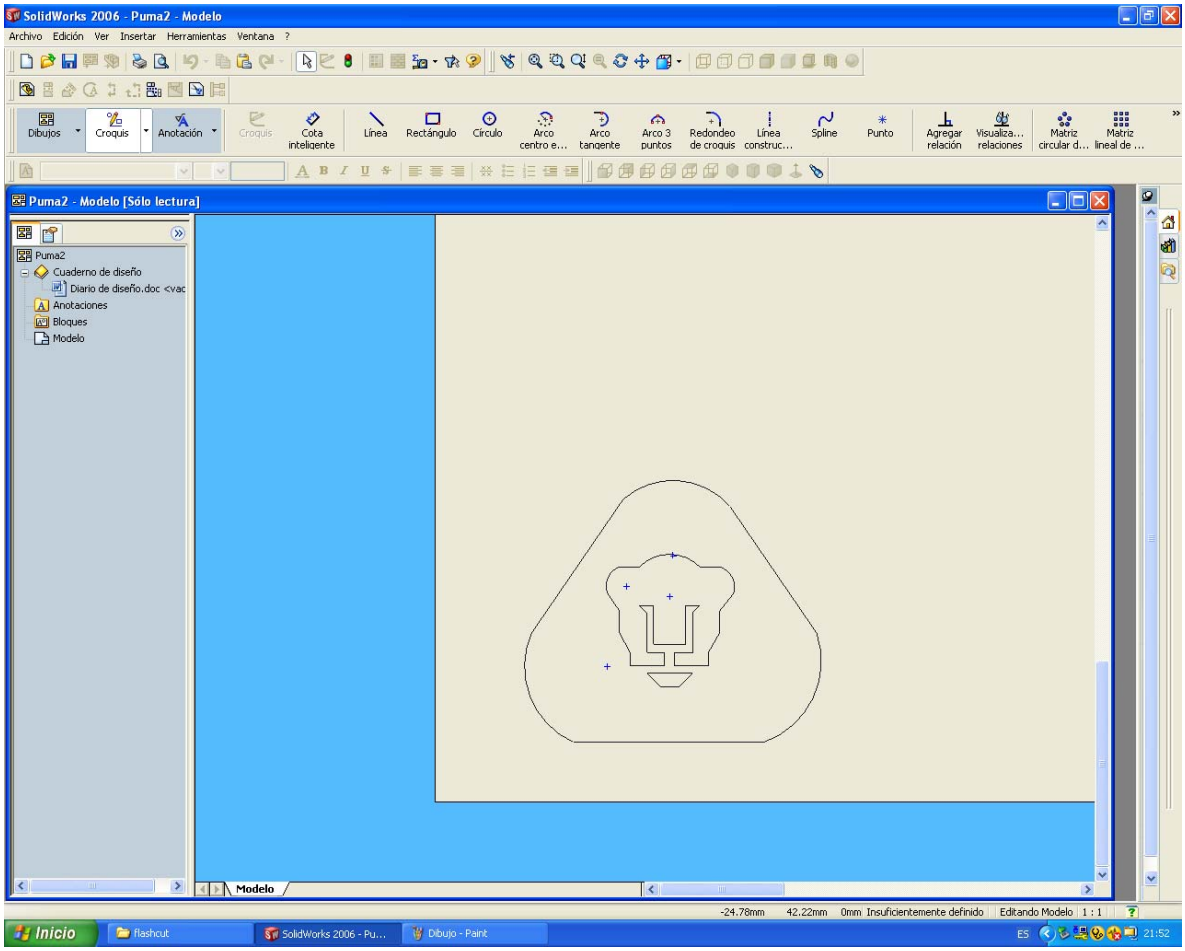

Figura 4.13. Escudo de los PUMAS de la UNAM, en formato DXF, creado mediante la aplicación “SolidWorks”.

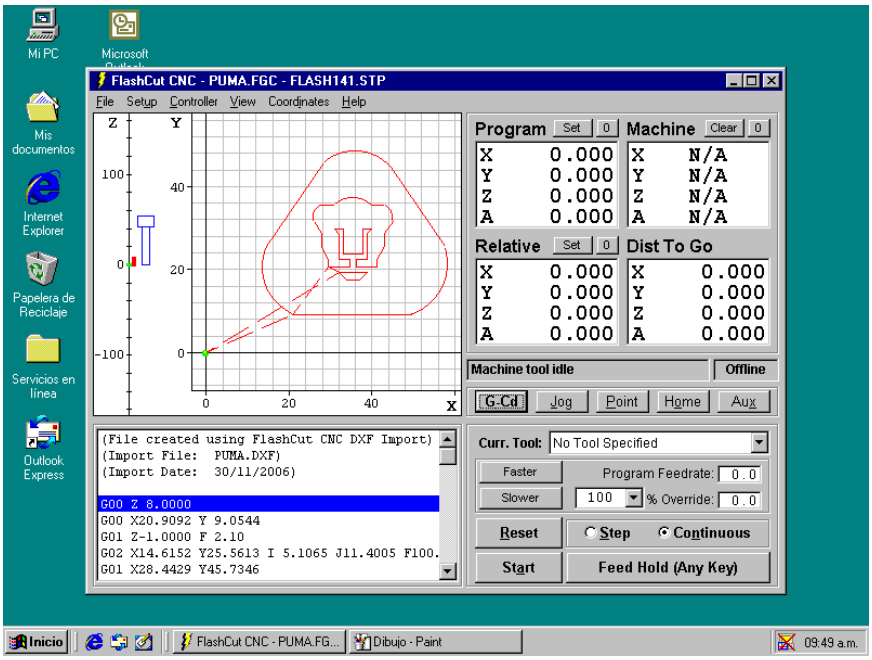

Figura 4.14. Código numérico, generado por "FlashCut", para maquinar el escudo de los PUMAS de la UNAM.

Como último paso, para obtener el maquinado del escudo de los PUMAS de la UNAM, se interpretó el conjunto de instrucciones de código numérico, generadas en el paso anterior, mediante el software de control desarrollado (ver figura 4.15); obteniéndose el resultado que se muestra en la figura 4.16.

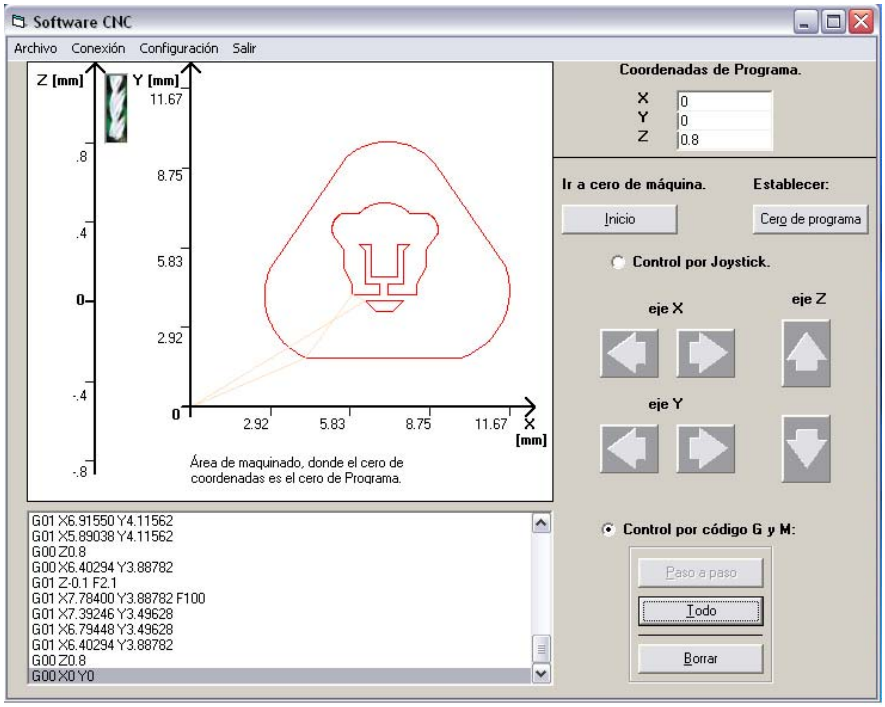

Figura 4.15. Ejecución de las instrucciones para el maquinado del escudo de los PUMAS de la UNAM, con la interfaz desarrollada.

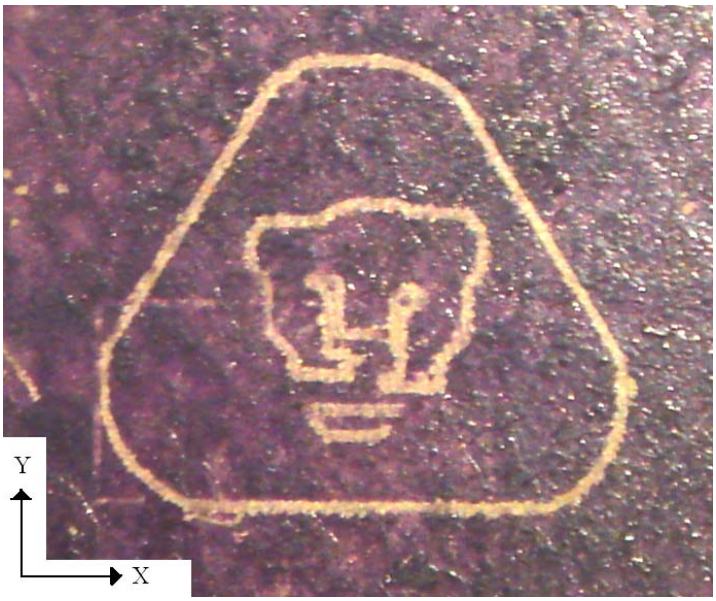

Figura 4.16. Maquinado del escudo de los PUMAS de la UNAM, utilizando como elemento de control la interfaz desarrollada.

El tiempo requerido para manufacturar esta pieza fue de 23 minutos con 30 segundos, aproximadamente; estableciendo el periodo medio de la señal con un valor de 20 [ms] en todo el proceso.

4.2. Resultados obtenidos.

Para el caso de maquinados simples, las piezas obtenidas tienen dimensiones muy cercanas al tamaño indicado por medio de las instrucciones de código numérico. Esta falta de precisión se debe, por una parte, al valor que se establece dentro de la aplicación, correspondiente al desplazamiento que se obtiene en cada eje de la MMH al dar un paso el motor que mueve a cada eje. Así también, otro de los factores que repercuten en esta falta de precisión es el desgaste que va teniendo la MMH a lo largo de su ciclo de vida útil. No obstante, los resultados obtenidos son sumamente buenos.

Para el caso de maquinados compuestos, la precisión de las líneas maquinadas utilizando interpolación en dos ejes es aceptable si se establece un periodo medio de la señal igual o mayor a 20 [ms]. Con un periodo medio de la señal mayor a 20[ms] se obtiene una mayor precisión, sin embargo, el tiempo para el maquinado aumenta.

En los maquinados con mayor número de instrucciones por interpretar, se aprecia, en el caso del logotipo del LMM, una excelente precisión. Por ejemplo, el punto final del maquinado para el engrane más grande de este logotipo (se maquina en sentido horario, por lo que el punto final de este engrane se encuentra a la derecha) coincide con el extremo superior derecho del rectángulo que conforma al logotipo. Por otro lado, se puede apreciar que en maquinados de figuras con

menores dimensiones, como es el caso del engrane pequeño, las figuras se logran ver bien detalladas. Para el caso del escudo de los PUMAS de la UNAM, el dado derecho de la cara del PUMA está un poco más abajo que el lado izquierdo, lo cual puede ser causa de alguna falla mecánica en la MMH, debido al desgaste que ésta ha tenido, y a errores de backlash que se van acumulando durante el maquinado.

Con esta aplicación desarrollada, el proceso de manufactura de piezas en la MMH, desarrollada en el LMM, es mucho más rápido y flexible en comparación con el anterior sistema de control; sobre todo, en el tiempo requerido para la programación de una pieza. Con el anterior sistema de control, el tiempo invertido en crear un programa en lenguaje C++, específico para una pieza; era de una a dos semanas [10]. Mientras que con esta aplicación se logra aprovechar el hecho de poder obtener un conjunto de instrucciones de código numérico mediante software CAD y CAM, con lo cual el tiempo total del proceso de maquinado se reduce considerablemente.

CONCLUSIONES

Con este trabajo se cumple satisfactoriamente el objetivo de contar con un software de control capaz de interpretar instrucciones de código numérico estándar y generar las señales de control hacia una micromáquina herramienta, desarrollada en el LMM; obteniendo así el maquinado correspondiente. Con lo anterior se logra reducir considerablemente el tiempo de manufactura de piezas (utilizando la MMH como fresadora), en comparación con el proceso de manufactura y el sistema de control utilizados antes de realizar este trabajo. Además se deja sentada la base para añadir mayores características a esta interfaz desarrollada, poder adaptarla para procesos de manufactura en los cuales se utilice la MMH en su configuración como torno, etc.

Dentro del funcionamiento de esta interfaz se contemplaron muchas de las características que presenta un software CAM, como es “FlashCut”. El comportamiento de dichas características fue el adecuado, como es el caso por ejemplo, de la rutina que regresa los tres ejes de la MMH a su posición de origen; el dibujo que se pinta dentro del cuadro de imagen de la interfaz, correspondiente a la ruta que sigue la herramienta de corte; etc. Además se añadió como elemento de seguridad la función de deshabilitar el protector de pantalla del sistema al momento de iniciarse la ejecución de la aplicación.

Las pruebas realizadas al sistema de control fueron satisfactorias, tomando en cuenta el desgaste de la MMH, lo cual repercute un poco en la precisión de los maquinados. También, se lograron detectar algunos problemas en este esquema en donde la mayor parte del control se realiza por medio de software.

TRABAJO A FUTURO

Para mejorar un poco la precisión de los maquinados, se está construyendo actualmente una nueva micromáquina herramienta, la cual presenta algunas mejoras, además de no presentar el desgaste que presenta la actual MMH, debido a su tiempo de uso.

Por otro lado, se puede adaptar dicha interfaz con el fin de realizar maquinados con la MMH en su configuración como torno, o incorporar mayores características a esta interfaz.

También, se puede implementar todo el funcionamiento que presenta esta interfaz mediante algún otro lenguaje de programación que permita realizar procesos en paralelo (manejo de hilos) y tener un mejor control del envío de las señales cuando estas son enviadas a dos actuadores a la vez; y medir, si es el caso, las mejoras que se tienen en cuanto a la velocidad de los maquinados que se realizan mediante una interpolación de dos ejes.

Otro de los desarrollos que se contemplan a futuro, es el de utilizar un dispositivo de control adicional en conjunto con esta interfaz, con el fin de que la PC pueda realizar otras tareas en paralelo a la manufactura de una pieza y sin afectar dicho proceso. Pretendiendo, además, conseguir una mayor velocidad en los maquinados.

REFERENCIAS.

1. González Núñez Juan. “El control numérico en las máquinas herramienta”. CECSA, 1990.
   ISBN 968-26-1101-6.
2. E. Kussul, T. Baidyk, L. Ruiz Huerta, A. Caballero Ruiz, G. Velasco y L. Kasatkina. “Development of micromachine tool prototypes for microfactories”. 3 de Octubre de 2002.
   PII: S0960-1317(02)34346-8.
3. “Control numérico”. Tecnología de fabricación y tecnología de máquinas. Área de Ing. de Sistemas y Automática. Escuela Politécnica Superior de Elche.
   http://lorca.umh.es/isa/es/asignaturas/tftm/texto%28cn1%29.pdf
4. Mario Ramos M. “Introducción a las máquinas de control numérico”.
   http://www.cimubb.ubiobio.cl/data/IntroCNC.PDF
5. Diseño y Manufactura asistidos por Computadora. Introducción al CNC (Ingeniería Industrial – UPIICSA).
   http://www.monografias.com/trabajos14/manufaccomput/manufaccomput.shtml.
6. Fundamentos del control automático industrial.
   http://www.sapiensman.com/control_automatico/
7. Sistemas de información. Bachillerato Tecnológico en Computación Grupo CCEA.
   http://bachilleratoccea.org/documentos2/TI_SI%20Intro.html
8. Robótica y automatización.
   http://www.ilustrados.com/publicaciones/EpZplVyuVyyCEmNbMu.php
9. Leopoldo Ruiz Huerta. Diseño y construcción de un microcentro de bajo costo. Tesis de maestría. 2000.
10. José Resendiz Sánchez. Investigación del sistema de control de una micromáquina herramienta y propuesta de optimización para el sistema. Tesis de licenciatura. 2004.
11. Manual de usuario de FlashCut CNC versión 1.4.
12. Héctor H. Silva López. Automatización de una micromáquina herramienta de primera generación.
    Tesis de licenciatura 2005.
13. Robótica. Janina, Rosa María.
    http://www4.uji.es/~al051829/trans22.html
14. Parallel port interfacing Inpout32.dll source code and theory of operation.
    http://www.logix4u.net/inpout_theory.htm
15. Raymond Mc Leod, Jr. “Sistemas de información gerencial”. Prentice Hall Hispanoamérica S.A., 2000.
    ISBN 970-17-0255-7
16. E. Kussul, D. Rachkovskij, T. Baidyk and S. Talayev. 1996.
    Micromechanical engineering: a basis for the low-cost manufacturing of mechanical microdevices using microequipment J. Micromech. Microeng. 6 410-425.
17. Microsistemas-Microtecnología. Revista robotiker – tecnalia.
    http://revista.robotiker.com/revista_estudios/microsistemas.html

18. Okazaki, Kitahara: Miciro-machine tool to machine microparts, Proc. ASPE Annual meeting 2000.
19. E. M. Kussul, L.M. Kasatkina. “Los problemas de desarrollo de máquinas herramienta controladas con computadora para fábricas micro mecánicas sobre mesa.// Sistemas de control y computadoras“. 1998, N5, V.28, pp. 32-39.
20. Nozomu Mishima, Microfactory Project, M4 Workshop on Micro/Meso Mechanical Manufacturing, May 16-17, 2000, NORTHWESTERN UNIVERSITY Evanston, IL, USA. http://www.mech.northwestern.edu/MFG/AML/M4/M4–Files/Mishima/index.htm
21. NANOWAVE. Nano Corporation. http://www.nanowave.co.jp