UNIVERSIDAD PANAMERICANA

ESCUELA DE INGENIERIA

CON ESTUDIOS INCORPORADOS A LA
UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

SIMULACION DE PROGRAMAS DE CONTROL NUMERICO PARA LA MAQUINA COMPACT 5 CNC

TESIS QUE PARA OBTENER EL TITULO DE:
INGENIERO MECANICO ELECTRICISTA
AREA: MECANICO ELECTRICA
P R E S E N T A
MAURICIO RENE ROSALES RIVEROLL

DIRECTOR DE TESIS
DR. STANISLAW F. RACZINSKY GAWIN

MEXICO, D.F. 1992

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE GENERAL

ABREVIATURAS COMUNES

ASCII	*American Standard Code for Information Interchange*
CAD	*Computer Aided Design*
CADL	*Cadkey Advanced Design Language*
CAM	*Computer Aided Manufacturing*
CNC	*Computer Numerical Control*
CR	*Carriage Return*
DNC	*Direct Numerical Control*
EIA	*Electronic Industries Association*
IGES	*Initial Graphics Exchange Specification*
ISO	*International Organization of Standarization*
LF	*Line Feed*
MDI	*Manual Data Input*
PC	*Personal Computer*
SI	*Sistema Internacional*

INTRODUCCION

¿Qué se entiende por máquina herramienta? Las personas no familiarizadas con este término podrían pensar que se trata de algún artefacto con martillos, desarmadores o pinzas brotando como tentáculos de su cuerpo en busca de alguna reparación a efectuar. Sin embargo, mucho dista esta definición de lo que en realidad es una máquina herramienta.

Mucho se ha escrito en relación a este tipo de aparatos así como de sus aplicaciones y objetos que producen. Su historia se remonta a siglos de desarrollo continuo y en la actualidad su utilización se extiende a todos los procesos de manufactura convencionales, ya sea de manera directa o indirecta.

Una máquina herramienta es un sistema que transforma la energía de una fuente (generalmente un motor eléctrico) en esfuerzos de deformación, fuerzas de corte o material erosionado con la finalidad de cambiar la forma de un material o desprender del mismo un pedazo de geometría definida. Como ejemplos concretos se pueden mencionar el torno, la fresadora, la rectificadora, el cepillo, la electroerosionadora, las troqueladoras, etcétera, hasta los modernos equipos de *LASER*.

Debido a la influencia que tienen las máquinas herramienta en los procesos de producción, la adecuada asignación de trabajo y control de recursos es esencial para mantener estas máquinas en un punto óptimo de aprovechamiento. Mucha ayuda

proporciona el estudio de tiempos y movimientos de la ingeniería industrial para la optimización de las rutas de producto dentro de los puestos de trabajo o maquinado en el caso de las máquinas herramienta.

Otro apoyo al mejoramiento de la producción es el manejo adecuado de las tablas de corte para los diferentes materiales a maquinar o de las mismas herramientas. Como estos ejemplos, podemos mencionar muchos más, pero el que es de vital importancia para esta tesis es el caso del control numérico dentro del desarrollo de las máquinas herramienta. En algunas ocasiones el diseño de un nuevo tipo de máquina herramienta, como es el sistema *LASER*, depende directamente del control numérico, ya que sin éste no tendría sentido el nuevo equipo.

El control numérico basa su funcionamiento en la modelación de un perfil, contorno o figura, en una serie de coordenadas contenidas en un medio de almacenamiento, y a partir de las cuales la máquina es capaz de reproducir la trayectoria adecuada para obtener la figura deseada.

Dado que la geometría a realizar está determinada en la mayoría de los casos al operador de la máquina (no es el caso de los troqueles), las propiedades que añade el control numérico a las máquinas herramienta hacen que estas últimas aumenten la precisión de sus productos, reduzcan los tiempos muertos de los equipos y, en consecuencia, los costos sean abatidos.

El control numérico no añade características de maquinado a las máquinas herramienta, lo que hace es mejorar el control de los movimientos para optimizar los resultados.

Aún cuando el operador no es el que realiza el maquinado de una pieza directamente, el control numérico no hará la pieza por él. Al igual que las computadoras, una máquina de control numérico necesita un operador, quien le indicará las operaciones a realizar, ya que la máquina no piensa ni juzga los acontecimientos, para eso está el operador.

Dicho lo anterior se concluye que una máquina de control numérico no es la solución absoluta de los problemas de maquinado de una empresa, ya que por sí sola no puede hacer nada. Se necesita personal capacitado para la operación del control numérico a través de la programación correcta de geometrías y condiciones de trabajo.

Un apoyo importante para el operador o programador son las herramientas de programación: editores, postprocesadores, simuladores, etcétera. Los simuladores tienen una importancia relevante dentro de la operación de un sistema de control numérico.

Ya que el operador proporcionará un programa codificado a la máquina, es necesario estar seguro de que el programa está correctamente elaborado, no sólo en sintaxis, sino también en su secuencia de operaciones y en los resultados que se obtendrán.

Por último, es necesario mencionar el gran apoyo proporcionado por los sistemas *CAD* y *CAM*.

En primer lugar, los sistemas *CAD* ayudan al diseñador a optimizar los productos en geometría, materiales y aprovechamiento de los mismos, así como contar con dibujos que fácilmente pueden ser reproducidos o modificados.

Los sistemas *CAM* ayudan al programador al cálculo de trayectorias y a optimizar sus procesos de maquinado; es desde este punto donde se puede notar la gran relación de un sistema *CAM* con un sistema de control numérico. Estos paquetes dan como resultado un programa codificado listo para ser cargado y ejecutado en la máquina de control numérico y, en muchas ocasiones, incluyen un simulador en su sistema. El control numérico se puede utilizar sin un sistema *CAM*, pero no viceversa; el *CAM* no tiene sentido si no se cuenta con control numérico.

El *CAD* es más bien independiente al *CAM* y al control numérico, sin embargo puede relacionarse con los dos anteriores. De este modo se tiene un sistema integral de diseño-manufactura.

Resumiendo el proceso: se diseña la pieza en *CAD* elaborando planos y toda la documentación de la misma. Con la geometría básica se establece relación con el *CAM* a

través de un traductor (*IGES*, *DXF*, *CADL*, etcétera) y se diseñan las trayectorias óptimas para el maquinado de la pieza. El resultado de este proceso es un programa de control numérico, que es modificado por un postprocesador, para que la máquina de control numérico sea capaz de interpretar la información que se le proporciona. Por último, se maquina la pieza.

CAPITULO 1

CONCEPTOS BASICOS DE CONTROL NUMERICO

1.1 DESARROLLO HISTORICO

El desarrollo del control numérico como herramienta de trabajo va muy ligado a la evolución de las computadoras. Si se parte del hecho de que la programación en control numérico se basa en la utilización de datos tabulados según coordenadas, bastará simplemente tener papel, lápiz y una máquina herramienta convencional para decir que se está trabajando con control numérico. Este es, precisamente, el punto de partida.

Hacia el año de 1942, en la Organización Parsons de Detroit, E.U.A., se presentó un contrato para la fabricación de componentes aeronáuticos. Estando los Estados Unidos combatiendo en la Segunda Guerra Mundial, era necesario sacar adelante la producción de componentes en el menor tiempo posible y con la calidad requerida.

A fin de obtener un perfil para las aspas de un helicóptero, se optó por la utilización de coordenadas basándose en los estándares realizados por el Comité Nacional de Apoyo a la Aeronáutica[1] para la tabulación de valores de superficies aerodinámicas.

El maquinado se llevó a cabo en una máquina fresadora tipo *Bridgeport* de dos ejes. Se consideró un factor por el diámetro de la herramienta y se proporcionaron los valores tabulados a dos personas. Una persona manipulaba el eje X y la otra el eje Y a través de sus respectivas manivelas.

Los resultados del maquinado fueron los más exactos jamás alcanzados por alguna máquina herramienta. Mientras que los estándares para la aeronáutica eran ± 0.007" por cada 17 puntos coordenados en la interpolación de la curva de un aspa, en el trabajo con datos tabulados alcanzó ± 0.0015" en 200 puntos.

A partir de este éxito se utilizaron los datos tabulados para la generación de perfiles en la industria aeronáutica, y para el año de 1949, la compañía Parsons pidió al Instituto Tecnológico de Massachusets (*M.I.T.*) el diseño de una máquina controlada por un computador o procesador central, dando como resultado la creación de una herramienta de corte vertical controlada numéricamente en 1952.

El concepto de control numérico y las máquinas mismas fueron dadas a conocer al público en 1954. El nombre de control numérico se debe principalmente a la naturaleza matemática de estas máquinas.

1.2 PRINCIPIO DE OPERACION DE UNA MAQUINA DE CONTROL NUMERICO.

En primer lugar, se necesitan los datos o coordenadas tabuladas a que se hizo referencia anteriormente. Estos datos se pueden obtener a partir de los planos de fabricación o de un sistema *CAM*. En el capítulo 2 se abordará la problemática del proceso de diseño a fin de obtener los datos que necesita la máquina.

En el momento en que se cuenta con los datos tabulados con la secuencia deseada, éstos deben suministrarse a la máquina. La máquina COMPACT 5 CNC, dispone de una memoria para programas de control numérico que da cabida a 210 bloques de programa (0 a 209). Los métodos utilizados para accesar los datos a la máquina pueden ir desde las cintas perforadas[2] hasta lo que se conoce como *DNC*, pasando por las cintas magnéticas, los teclados, etcétera. La modalidad de teclado, con la terminología *MDI*, es la más usada en la actualidad.

Una vez suministrados los datos a la memoria de la máquina y, por supuesto, la pieza de trabajo y las herramientas montadas, se procederá a oprimir el botón de arranque.

Internamente la máquina tomará los bloques uno por uno para analizar su contenido y ejecutarlo. El contenido es el siguiente:

a) Tarea a realizar. Movimiento lineal o en curva, arrancar el husillo, cambiar herramienta, activar líquido de enfriamiento, etcétera.

b) Coordenadas. Valores en los ejes coordenados a alcanzar en el caso de movimientos de la herramienta. Tiempo en segundos en el caso de ciclo de espera, etcétera.

c) Parámetros de trabajo. Velocidad de avance, velocidad del husillo, profundidad de corte o ciclo, etcétera.

Una vez definido lo anterior, el control de la máquina verifica la posición actual de la herramienta y determina el movimiento que se debe realizar para alcanzar las coordenadas. Cabe mencionar que el control revisa la sintaxis y la capacidad de ejecución de cada bloque de programa; si un bloque no puede ser ejecutado, ya sea porque las coordenadas exceden las posibilidades de la máquina o por datos erróneos en una interpolación circular, el control da a conocer la condición de error y el proceso se detiene para evitar algún accidente en el maquinado.

La trayectoria de la herramienta se logra a través de una combinación de movimientos simultáneos de servomotores (cuando se trata de una interpolación), cuya secuencia es controlada por el procesador central o controlador.

Paralelamente al movimiento de la herramienta, sensores de distancia monitorean la posición del cabezal a fin de retroalimentar al controlador y, de esta forma, definir si el movimiento debe continuar o detenerse y, en el caso de una desviación de la trayectoria, que ésta sea corregida antes de que tenga mayor influencia en la pieza de trabajo.

1.3 ESTRUCTURA DE UN PROGRAMA DE CONTROL NUMERICO

La forma en que una máquina de control numérico interpretará las intrucciones dadas es muy específica. En términos de taller, se habla de cilindrado o tal vez algún

careado (*fig 1.1*), para referirse a un torno. En un torno de control numérico, estas operaciones se distinguirán con los códigos *G84* y *G88* respectivamente (ver apéndice C).

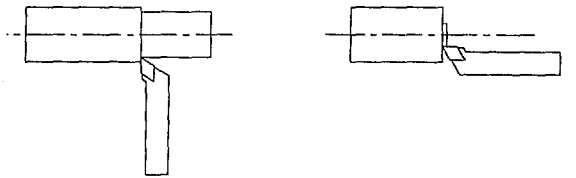


Figura 1.1 CILINDRADO Y CAREADO

Al ver el listado de un programa de control numérico se pueden distinguir los siguientes campos de definición de datos:

a) Número de línea. En la gran mayoría de los casos, esta numeración parte de cero y queda limitada por la capacidad de memoria de la unidad de control. Para superar esta limitación, algunas máquinas utilizan las unidades de disco flexible e incluso una computadora externa (*DNC*) para ampliar la capacidad del número de bloques que se pueden ejecutar como un solo programa. Ciertas curvas aparentemente sencillas en tres dimensiones llegan a utilizar más de 30,000 bloques para su maquinado.

b) Código. Este campo del bloque de programa es el que define la tarea a ejecutar. Existen dos tipos de códigos u operaciones: códigos *G* (*Geometric Word*) o preparatorios y códigos *M* (*Miscelaneous Word*) o misceláneos.

Como lo especifica la definición en inglés, los códigos *G* se refieren a lo que son las operaciones geométricas del maquinado y que, a final de cuentas, dependen del perfil o figura deseada. Los principales tipos de operación contemplados por estos códigos son los siguientes:

- Movimientos (e interpolaciones) lineales o circulares
- Definición de planos de trabajo
- Aplicación de ciclos predefinidos (ciclos enlatados)
- Definición del modo de operación, etcétera

Por lo que respecta a los códigos *M*, estos se refieren a la ejecución de procesos específicos de cada máquina, como pueden ser el encendido del husillo en sentido horario o antihorario, la activación del fluido de enfriamiento para el maquinado, el cambio automático o manual de herramientas, la activación de mecanismos especiales, etcétera.

El resultado de la aplicación de un código *M* puede variar de una máquina a otra, dependiendo del controlador que se utilice, sin embargo, en la actualidad estas operaciones están estandarizadas para la mayoría de los fabricantes de máquinas herramientas.

En el apéndice C, se encuentra una lista de los códigos preparatorios y misceláneos más comunes.

c) Coordenadas en los ejes *X, Y* y *Z*. Como se mencionó en el apartado 1.2, los datos tabulados deben ser ingresados a la máquina de control numérico y específicamente al controlador. Los registros de memoria utilizados para este fin son las letras correspondientes a los ejes coordenados: *X*, *Y* y *Z*. En el caso de las máquinas de cinco ejes, las letras utilizadas son *C* y *D* (o IV y V ejes respectivamente). Los valores contenidos en estas localidades de memoria definen las distancias a recorrer y, en algunos códigos, los parámetros del ciclo de trabajo. El formato de los valores varía de un controlador a otro; mientras que en un controlador *Heidenhein TNC355* el valor 100 significa 100.000 mm, en un controlador *EMCO COMPACT* significará 1.00 mm. Es conveniente revisar siempre los manuales de operación de cada controlador en particular.

Por otro lado, vale la pena mencionar que las coordenadas que manejan los ejes cuarto y quinto, son normalmente de giro, esto es, sus valores estarán dados en grados o en radianes (esta última unidad muy poco usada), al contrario de los tres ejes mencionados con anterioridad, que siempre son lineales.

d) Parámetros de trabajo. Estos parámetros son los que definen las condiciones del maquinado y, el principal de ellos es la velocidad de avance, esto es, la velocidad en la dirección del movimiento, no importando si es lineal o circular. Normalmente se maneja en unidades por minuto y en algunas ocasiones en unidades por revolución, que es mucho más utilizado en los tornos convencionales y sólo a lo largo de los ejes principales.

Otro parámetro a considerar es la velocidad del husillo, sin embargo, no siempre se le encuentra en los códigos de programa, sino que sólo se le puede definir manualmente dependiendo de la máquina que se esté utilizando.

1.4 INTERRELACION CON LAS MAQUINAS HERRAMIENTA

Hasta el momento se ha hablado únicamente de datos en el papel y posteriormente transferidos a un controlador. Los resultados o salidas que dé el controlador serán señales para el movimiento concreto de un eje o más.

1.4.1 Definición de ejes

Al igual que con los códigos de comando y las cintas perforadas, la definición de ejes está estandarizada.[3]

En primer lugar se tiene el trabajo en un solo plano (troqueladoras, electroerosión por hilo, rayo laser, etcétera), en este caso el eje *X* será paralelo al frente de la máquina y el eje *Y* perpendicular. Cuando se habla de más ejes de trabajo, entonces es necesario incluir un eje por lo menos. En este apartado se incluyen los tornos, centros de maquinado, electroerosión por penetración, etcétera. La mayoría de estas máquinas tienen un eje que

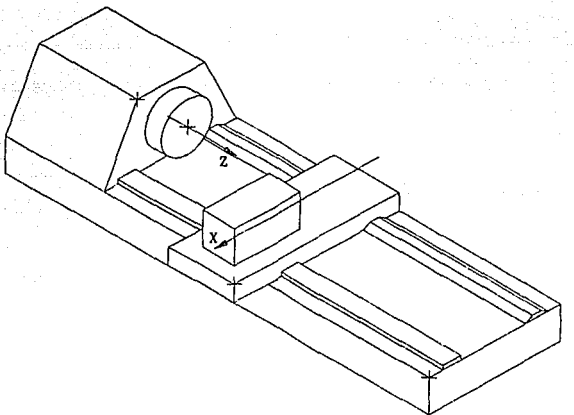


Figura 1.2 DEFINICION DE EJES EN UN TORNO

define el movimiento principal o de corte; si se considera este movimiento como rotatorio, entonces se puede definir un eje para dicho movimiento, este eje es el *Z*, y normalmente su dirección parte de la pieza de trabajo hacia la herramienta, si la herramienta está más alejada de la pieza en este eje, *Z* tendrá un valor mayor y viceversa.

A partir de esta definición, el eje *X* será el que represente un mayor recorrido para la herramienta en dirección perpendicular a *Z* o, en su defecto, en dirección paralela al frente de la máquina herramienta. Por último, el eje *Y* es perpendicular al eje *X* y al eje *Z* según la regla de la mano derecha.

Cuando se utiliza un eje lineal auxiliar y paralelo a los ejes principales[4], se dice que se trata de un eje secundario y se representa con las letras *U*, *V* y *W*, que son paralelos a los ejes *X*, *Y* y *Z* respectivamente.

De la misma manera se habla de los ejes de rotación, o movimientos rotatorios. Estos son los ejes *A*, *B* y *C*, los cuales giran alrededor de *X*, *Y* y *Z* respectivamente.

Eje principal	Eje auxiliar	Eje de rotación
X	*U*	*A*
Y	*V*	*B*
Z	*W*	*C*

1.4.2 Actuadores y retroalimentación

Los actuadores o motores de control son los que dan lugar al movimiento entre la herramienta y la pieza de trabajo. En la gran mayoría de los casos se trata de un movimiento rotatorio que alimenta a un tornillo de avance y este último a la mesa de trabajo o a la herramienta.

Como se mencionó en el apartado 1.2, se dispone de elementos sensores de distancia para determinar la posición de la herramienta respecto a la pieza de trabajo, esto da la retroalimentación al controlador.

Dentro de la definición de los motores, se encuentran básicamente cuatro tipos:

a) Motores hidráulicos. Ofrecen un motor potente y compacto, tienen rápida respuesta a las señales y el defecto de ser ruidosos.

El sistema de suministro hidráulico es relativamente caro y generalmente llega a presentar fugas. Por esta razón, estos sistemas se instalan principalmente en máquinas de control numérico muy grandes.

b) Motores de Paso. Este tipo de motores giran un ángulo definido por cada pulso recibido del controlador y no requieren el uso de sistemas de retroalimentación. Tienen la desventaja de ser relativamente bajos en potencia y velocidad, además de ser caros en relación con otros tipos de motores.

c) Motores de corriente directa con escobillas. Se trata de un motor común de corriente directa y es el más usado en las máquinas de control numérico. Son de precio razonable y tienen un rango muy amplio de potencia y velocidad. Requieren poco mantenimiento.

d) Motores de corriente directa sin escobillas. El funcionamiento de estos motores difiere en la ubicación del campo magnético constante. Mientras que los motores convencionales tienen el campo magnético en el estator, este nuevo tipo de motores lo tienen en el rotor. Como la conmutación de campos para el movimiento se realiza en el estator, no es necesario el uso de escobillas. En sí, el motor es relativamente barato, pero el sistema completo, que incluye el sistema electrónico para la conmutación de campos en el estator, es mucho más caro.

El uso de sistemas de retroalimentación da lugar a los términos de ciclo cerrado y ciclo abierto; este último sólo es utilizado con motores de paso.

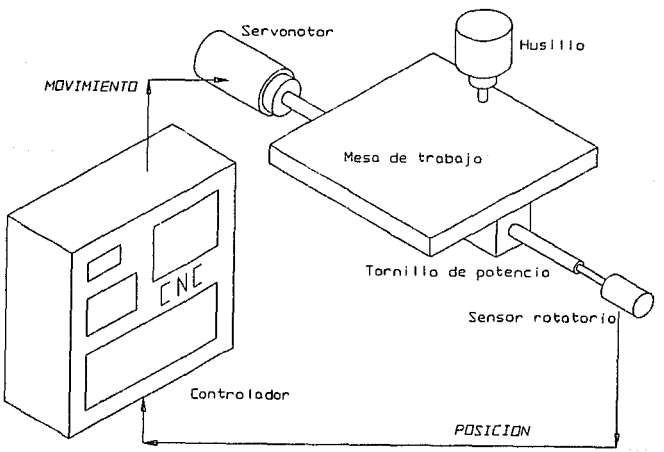


Figura 1.3 RETROALIMENTACION DE UN SISTEMA DE CONTROL NUMERICO

El controlador recibe la señal de retroalimentación y realiza una comparación de valores para determinar si se debe continuar o detenerse.

Existen varios tipos de elementos de retroalimentación:

a) Transformador rotatorio. Son compactos y se colocan directamente en el motor o al final del tornillo de avance.

Básicamente, el cableado del motor es excitado por una corriente alterna de referencia y la amplitud del voltaje inducido en el devanado del estator es proporcional al desplazamiento angular del rotor.

b) Escala lineal. Consiste en dos partes acopladas magnéticamente. La escala se coloca en una parte fija y el cursor en un elemento asociado al movimiento de la mesa o de la herramienta. Durante el movimiento de algún elemento de la máquina, el cursor se mueve a lo largo de la escala. Su funcionamiento es similar al transformador rotatorio, con la diferencia de que éste es lineal.

c) Decodificador óptico. De manera simplificada se trata de un disco con un número determinado de orificios en su periferia conectado al tornillo de avance. Por cada orificio que pasa frente a la fuente de luz y su respectivo sensor, se genera un pulso que se traduce en una distancia recorrida.

1.5 MODALIDADES DE LA PROGRAMACION

Existen varios aspectos a considerar para realizar la programación. Estos dependen básicamente de la geometría a generar y, en algunos casos, del estilo de programación del usuario.

1.5.1 Posibilidades del controlador

Al respecto se pueden mencionar varios puntos que se deben determinar antes de realizar la programación sin incluir las condiciones de trabajo.

a) Programación en sistema absoluto o incremental. Actualmente la mayoría de las máquinas de control numérico trabajan con sistema absoluto.

b) Programación en pulgadas o milímetros

c) Control de diferentes elementos de la máquina. Estos pueden ser: el líquido de enfriamiento, señales de alarma, cambio automático de herramientas, apertura de mordazas, arranque de husillo, etcétera.

1.5.2 Diseño de las trayectorias

Para poder definir una trayectoria, esto es, obtener datos tabulados y ordenarlos en una secuencia de movimientos, se requiere cierto nivel de conocimiento o dominio de las matemáticas.

En general, la programación de coordenadas se realiza para dos tipos de operaciones: posicionando y contorneando.

a) Posicionado. Este tipo de programación es típico de los taladros y las troqueladoras, en los cuales el centro de la herramienta debe ubicarse en una coordenada definida. El movimiento de un punto a otro se realiza sin hacer contacto con la pieza de trabajo y a alta velocidad. Una vez alcanzado el punto, se realiza una operación, y al ser terminada, se puede mover la herramienta al siguiente punto.

Este método es ideal para el uso de ciclos enlatados o subrutinas.

b) Contorneado. La principal característica de este tipo de programación es que la herramienta hace contacto con la pieza mientras se realiza un movimiento. El avance es lento y el husillo debe permanecer en rotación. En el caso de los centros de maquinado, es necesaria la compensación de herramientas, ya que no es lo mismo hacer un contorno por dentro que por fuera. En el proceso de torneado no se utiliza compensación, y en este sentido se podría decir que se trata de simple

posicionamiento; sin embargo, esto no es aplicable, ya que la herramienta se encuentra en contacto con la pieza de trabajo.

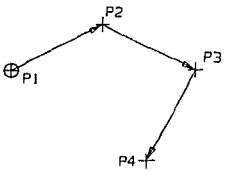


Figura 1.4 POSICIONADO

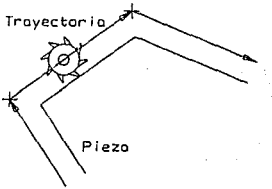


Figura 1.5 CONTORNEADO

Además, mientras que en el posicionado se trabaja sólo con interpolaciones lineales, en el contorneado se trabaja también con interpolaciones circulares, parabólicas, etcétera, dependiendo de la capacidad matemática del controlador.

Partiendo de estos dos tipos de programación se pueden obtener varios grados de complejidad en los programas, desde el posicionado en dos ejes hasta la escultura en cinco ejes. En consecuencia, el nivel de conocimientos necesario será mayor.

El siguiente cuadro da una idea de estos requerimientos.[5]

Requerimientos educacionales	Complejidad de la pieza + 1	2	3	4	5	6	7	8	9
Prácticas básicas	*	*	*	*	*	*	*	*	*
Geometría plana		*	*	*	*	*	*	*	*
Algebra simple		*	*	*	*	*	*	*	*
Geometría analítica simple			*	*	*	*	*	*	*
Geometría de sólidos				*	*	*	*	*	*
Algebra					*	*	*	*	*
Geometría analítica						*	*	*	*
Programación básica de computadoras							*	*	*
Geometría descriptiva								*	*
Trigonometría plana								*	*
Trigonometría esférica									*
Computación digital									*
Análisis numérico									*

+Complejidad de la pieza:
1. Programación punto a punto, cortes rectos en dos y medio ejes[6]
2. Curvas no definidas por fórmulas
3. Cortes repetitivos y familias de partes
4. Curvas y líneas rectas en dos y medio ejes
5. Movimiento rotatorio controlado
6. Planos inclinados y patrones
7. Superficies complejas
8. Superficies regulares y definidas por fórmula
9. Multiejes incluyendo movimientos simultáneos en 5 ejes.

1.6 PROGRAMACION AUTOMATICA

Como ayuda para el programador, a partir de la aparición del control numérico, se han desarrollado lenguajes para la programación y cálculo automático de parámetros y trayectorias. No se trata de un sistema *CAM*, sino de todo un lenguaje de programación como puede ser *BASIC*, *COBOL* o *Pascal*, con la característica de que está orientado al manejo de las máquinas de control numérico.

Los sistemas de programación automática son normalmente desarrollados para usarse en un proceso de producción en particular (torneado, barrenado, fresado, etcétera). Debido a la complejidad de los cálculos que llegan a realizar, estos programas sólo pueden utilizarse en sistemas computacionales de gran tamaño, aunque recientemente han aparecido en el mercado programas para aplicación en computadoras personales.

Al igual que los sistemas *CAM*, la programación automática requiere el uso de un postprocesador para convertir el programa desarrollado en un lenguaje específico a un código de control numérico con el formato particular de la máquina herramienta que se utilice. Normalmente este código es almacenado en cinta perforada (ver *fig 2.5* y *2.6*).

El más universal sistema de programación automática y lenguaje es el *APT* (*Automatically Programmed Tools*). Es de dominio público y ha sido adoptado por la *ISO* y la *ANSI*.

A continuación se muestra una tabla con los principales lenguajes de programación a nivel comercial:

- Todos los sistemas de programación (alrededor de 150).

APLICACION	CAPACIDADES	EJEMPLOS
Barrenado (26)	control punto a punto (22)	IFAPT C CAMPOINT DATAPOINT NUMERICON AUTOSPOT REMAPT SNAP
	control lineal (4)	EXAPT 1 2PL AUTOPROPS SYMAP S
Torneado (28)	control lineal (9)	CAMPTURN SAP GSHAFT CAMSHAFT AMD RESULTS FRED
	control de trayectoria continua en 2D (19)	EXAPT 2 AUTOPIT PROGRAMAT INDEX H100 MITURN AUTOSHAFT IFAPT 2
Barrenado y fresado (22)	control lineal (3)	PROFILE ROMANCE UNIAPT
	control de tayectoria continua en 2 1/2 ejes (9)	EXAPT 1.1 CAMPFIVE NUMERICOMP
Barrenado, fresado y torneado (30)	control de tayectoria continua en 2 ejes (8)	ADAPT COMPACT II DAVID ELAN 30 SPLIT
	control de tayectoria continua en 2 1/2 ejes (7)	MINIAPT EASYPROG AUTO-ACTION TELEAPT BRUSYS Basic-EXAPT

	KUIKRATE
control de trayectoria continua en 3 ejes (15)	APT ADAPT NUFORM FAPT SINAPT SYNAPT

NOTAS DEL CAPITULO 1

[1] Más adelante, Administración Nacional de Aeronáutica y del Espacio, N.A.S.A.
[2] Según la norma EIA 358-B que es compatible con el código ASCII para las computadoras.
[3] EIA RS 267 - B
AIA NAS - 938
ISO/R 841
[4] Debe distinguirse el término *eje principal* al de *movimiento principal*.
[5] Tomado de *Modem Machine Shop NC/CIM 1990 Guidebook*, Gardner Publications Inc., Ohio, E.U.A., 1990.
[6] La programación en dos y medio ejes implica sólo el movimiento simultáneo en dos ejes, quedando el tercero sólo como movimiento alternativo. No es posible el movimiento simultáneo de tres ejes.

CAPITULO 2

MAQUINAS HERRAMIENTA Y CONTROL NUMERICO

2.1 CARACTERISTICAS QUE AÑADE EL CONTROL NUMERICO A LAS MAQUINAS HERRAMIENTA CONVENCIONALES.

A simple vista se puede decir que la diferencia entre una máquina herramienta convencional y una máquina de control numérico es que la segunda tiene botones de control. Este punto de vista dista mucho de lo que en realidad implica el hecho de que una máquina sea controlada numéricamente. Para poder comprender esta diferencia se considerará el conocimiento de las máquinas herramientas convencionales, así como su funcionamiento y se explicará la configuración de una máquina de control numérico.

2.1.1 La estructura o chasis.

En principio, se puede decir que la estructura o esqueleto de los dos tipos de máquinas es la misma. Se trata de una pieza fundida que debe soportar todos los aditamentos que la máquina herramienta convencional o de control numérico pueda tener; respecto de este punto, vale la pena mencionar el hecho de que la máquina de control numérico debe presentar un soporte especial para el módulo de control, y la máquina convencional no. Sin embargo, esto no es lo más importante. El chasis debe ser capaz de soportar las cargas que produce el corte de material en el proceso de maquinado, que en algunos casos llegan a ser sumamente altas, al grado de romper las herramientas de corte.

En general podemos enunciar una serie de características que debe cubrir el chasis de una máquina herramienta, ya sea de control numérico o convencional.

a) Fuerza. Para resistir la presión de corte, el peso de los subensambles (sistemas) montados sobre el mismo, como puede ser el control de los servomotores, las herramientas y el peso de la pieza en que se trabaja.

b) Rigidez. Con la finalidad de que las dimensiones requeridas de la pieza no puedan desviarse bajo la presión de las fuerzas de corte y el peso de la pieza misma.

c) Estabilidad. De forma que no se distorsione la geometría del chasis por la lenta liberación de tensiones internas o por cambios en su estructura cristalina a lo largo de su vida de trabajo.

d) Amortiguamiento. De tal manera que las vibraciones generadas durante el corte se absorban antes de que puedan afectar la precisión y el acabado de la pieza de trabajo.

2.1.2 Fuente de energía.

En las máquinas de control numérico se requiere energía para tres aspectos principales: el husillo (movimiento principal[1]), los controles (movimiento secundario[2]), la computadora o sistema de manejo de datos de control numérico.

Para alimentar el husillo la fuente suele ser, en prácticamente la totalidad de los casos, un motor eléctrico, en particular, de conexión trifásica. La transmisión de la energía suele realizarse por medio de una banda y después es transferida a un sistema de engranes. La potencia nominal del motor es alta y esto es determinante para la adquisición de una máquina herramienta. A este respecto podemos decir que no existe diferencia sustancial entre las máquinas de control numérico y las convencionales.

Algunas máquinas de control numérico tienen motores de corriente directa que permiten hacer variaciones de la velocidad del husillo por medio de un potenciómetro y, de esta manera, eliminar los sistemas de engranes.

En cambio, el aspecto del control de los movimientos secundarios es muy diferente entre los dos tipos de máquinas en cuestión, y es determinante para dar explicación del éxito de las máquinas de control numérico.

Mientras que en una máquina herramienta convencional el movimiento secundario se determina mediante el operador, las máquinas de control numérico lo hacen a través de servomotores, en la mayoría de los casos.

Trabajando paralelamente al control de los movimientos secundarios y como parte de éstos, están los movimientos de avance. Normalmente, este movimiento se realiza automáticamente en las máquinas herramienta convencionales, teniendo la limitante de sólo trabajar a lo largo de los ejes principales. Las máquinas de control numérico realizan su movimiento de avance en prácticamente cualquier dirección.

Por último, por lo que concierne al consumo de energía del controlador de una máquina de control numérico, éste se puede equiparar al de una computadora que requiere ser conectada a una toma de corriente.

2.1.3 Superficie o eje de referencia.

El plano o eje de referencia (fresadora o torno, según sea el caso) da el principio mismo de funcionamiento de cualquier máquina herramienta, de control numérico o convencional. Gracias a esta referencia es como se puede lograr la concentricidad (o excentricidad, si se desea) de una pieza para ser maquinada en el torno.

Una característica clara que debe tener la referencia de una máquina herramienta es la rigidez, ya que alguna variación en la posición de la misma provocaría desde una alteración dimensional de la pieza maquinada, hasta la deformación permanente o ruptura de la pieza o la máquina misma.

2.2 REPERCUSION EN LOS PROCESOS DE MANUFACTURA DESDE EL DISEÑO HASTA LA FABRICACION.

El control numérico ha revolucionado los métodos de trabajo de las personas involucradas con los procesos de maquinado. En principio se puede decir que los tiempos de producción, así como el costo por pieza se han reducido notablemente (*fig 2.1*).

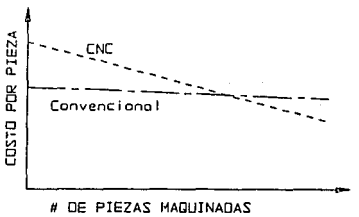


Figura 2.1 COMPARACION DE COSTOS SEGUN LA TECNICA DE MAQUINADO

2.2.1 Condiciones para el uso de control numérico

Antes de tomar la decisión de hacer un maquinado en una máquina de control numérico, se debe pensar en la factibilidad de este método.

Si la pieza sólo requiere maquinados sencillos, no vale la pena fabricarla por control numérico, ya que, tan sólo el tiempo de programación, costaría más que el maquinado mismo. En este caso, el maquinado convencional resulta más conveniente no sólo por el costo, sino también por el tiempo requerido para el trabajo.

Una pieza complicada o repetitiva destinada a alta producción sí justifica la utilización del control numérico.

2.2.2 Proceso de manufactura

Cabe mencionar que el diseño no se hace en función del control numérico[3], sino que el control numérico es una herramienta para llevar a cabo la fabricación de la pieza de

una manera óptima, aunque siempre se deben tomar en cuenta los recursos de los que se dispone (trabajadores, máquinas y herramientas de corte, etcétera). El proceso de manufactura varía en algunos aspectos que se mencionarán en la siguiente secuencia de trabajo (todas las etapas del proceso de manufactura deben retroalimentarse al punto de diseño, ya que todas ellas se llevan a cabo en función de éste).

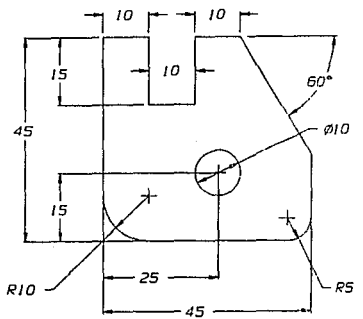
a) Diseño de la pieza y del plano. Básicamente la diferencia en los planos radica en la manera de presentarlos. Cuando se utiliza una máquina herramienta convencional, las dimensiones se ubican principalmente en los lugares en los que es importante que se cumpla un valor definido y en donde se tenga facilidad para realizar las mediciones de verificación de la pieza (*fig 2.2a*). En el caso de que se utilice una máquina de control numérico, se debe tomar en cuenta que trabaja con datos tabulados. Por esto es necesario hacer un plano extra en el que se especifiquen las coordenadas principales referidas a un origen para la fácil elaboración del programa de control numérico (*fig 2.2b*).

b) Plan de trabajo. Aquí se consideran los procesos de maquinado requeridos para obtener la geometría de la pieza a maquinar.

En una máquina convencional, el plan de trabajo queda sujeto a la calidad del maquinado a ejecutar, a las herramientas de corte y a las máquinas herramienta de que se dispone. Por ejemplo, un proceso de desbaste se realizará por lo general en cepillos, sierras o herramentales de los que se sabe no tienen buena precisión y soportan fuerzas de corte bastante altas.

Cuando se utiliza control numérico, la secuencia se elabora en función de las ventajas de movimiento de la herramienta respecto a la pieza buscando su trayectoria óptima y en función de la precisión de la máquina. En este sentido, las herramientas de corte utilizadas deben ser de alta calidad para no afectar la precisión; es recomendable no retirar la pieza de la máquina antes de que haya finalizado todo el proceso de maquinado. En el caso de que se tenga que retirar la

A) MAQUINADO CONVENCIONAL



B) MAQUINADO CON CONTROL NUMERICO

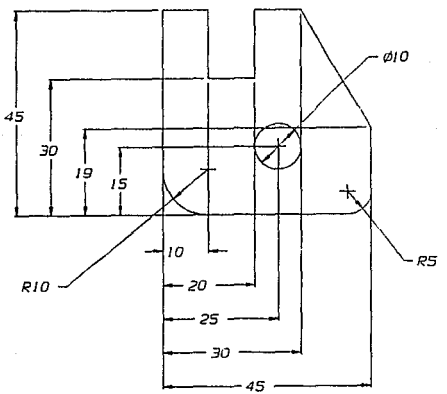


Figura 2.2 ELABORACION DE PLANOS PARA MAQUINADO EN CONTROL NUMERICO

pieza, una opción es maquinar menos material y dejar puntos o planos de referencia a fin de poder localizar el origen y la orientación que se tenía en un principio.

c) Habilitación de material y herramientas. Utilizando máquinas de control numérico, es conveniente habilitar el material no sólo definiendo un plano o eje de referencia (planeado en fresadora o careado en tornos) como se hace con las máquinas convencionales, sino también para disminuir el tiempo de trabajo de la máquina. Lo común es aproximar la forma de la pieza mediante desbaste.

d) Elaboración del programa. Cuando se dispone de los planos en coordenadas, el operador simplemente copia los datos contenidos en el plano de acuerdo a la secuencia de movimientos determinados en el plan de trabajo. La manera en que la máquina reconocerá los comandos que se le suministran se estudió en el capítulo 1.

e) Maquinado de la pieza. En las máquinas convencionales, este punto queda a libre decisión del operador, ya que éste es el que realizará los movimientos que darán forma a la pieza.

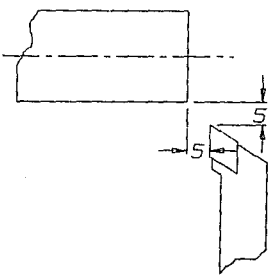


Figura 2.3 POSICIONAMIENTO PARA ARRANQUE DE PROGRAMA

Utilizando una máquina de control numérico, una vez habilitado el material o aproximado a su contorno final, es necesario colocarlo en la mesa de trabajo o en

el husillo (según sea el caso de fresadora o torno) y determinar el origen de donde partirán las coordenadas (*fig 2.3*).El siguiente punto es instalar el programa a ejecutar en la memoria de la máquina. Existen varios métodos para instalar o transmitir los programas a dicha memoria.

Una vez hecho lo anterior, solo queda oprimir el botón de arranque y la máquina hará el resto.

En una máquina herramienta convencional, es usual verificar las dimensiones obtenidas después de cada etapa del maquinado. El primer punto es tomar las medidas iniciales, estas pueden ser en bruto o el resultado de la preparación del material (inciso c), y definir un punto cero de maquinado.

Al trabajar con máquinas de control numérico, la verificación de las dimensiones sólo se realiza al final del maquinado. La fijación del punto cero es parecida a la operación en una máquina convencional, con la diferencia de que en control numérico se deja una distancia de seguridad para el arranque del programa.

Cabe mencionar que en las máquinas de control numérico que cuentan con cabina, se requiere que la puerta esté cerrada para poder trabajar, de lo contrario se activará una alarma y el proceso se detendrá.

2.2.3 Influencia en la producción

Como se mencionó al principio del apartado 2.2, el tiempo del proceso diseño-manufactura se reduce considerablemente utilizando el control numérico. Sin embargo, no es lo mismo hablar del tiempo de maquinado a hablar del tiempo requerido para la elaboración de la pieza, ya que este último implica la programación de la pieza en la máquina y además se corre el riesgo de tener tiempos muertos por falta de trabajo para la máquina.

No así con producciones altas, porque mientras se está maquinando un lote de piezas, se puede programar el siguiente, si no es que ya se cuenta con el programa almacenado.

Con el fin de mantener la o las máquinas con un mínimo de tiempos muertos, es necesario contar con gente capacitada para la supervisión de los trabajos, uno o varios programadores y operadores, y una constante comunicación con el departamento de diseño.

Otro punto de importancia es la adecuada administración de los herramentales, tanto en las condiciones de trabajo como en la vida útil de los mismos.

2.3 UTILIZACION DE LAS MAQUINAS DE CONTROL NUMERICO

Como utilización se entiende la manera en que los datos se proporcionarán a la máquina de control numérico que, a nivel operador, es la principal diferencia entre el manejo de estas máquinas y las máquinas herramienta convencionales, pasando por quién programa y cómo programa.[4]

2.3.1 Los programas

Las estadísticas proporcionan datos acerca de la aplicación industrial del control numérico, y aunque se podría hablar también del nivel educacional, éste no da lugar a puntos de comparación por el hecho de que la finalidad del control numérico es la industria.

En principio se puede decir que todas las personas involucradas con la máquina de control numérico pueden programar. En quien más incide este trabajo es en los operadores; sin embargo, la contratación de programadores especializados y técnicos en *CAD/CAM* ha ganado terreno en el aspecto de la programación (*fig 2.4*).

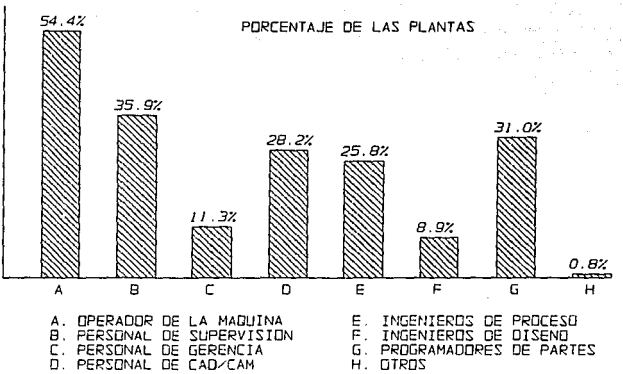
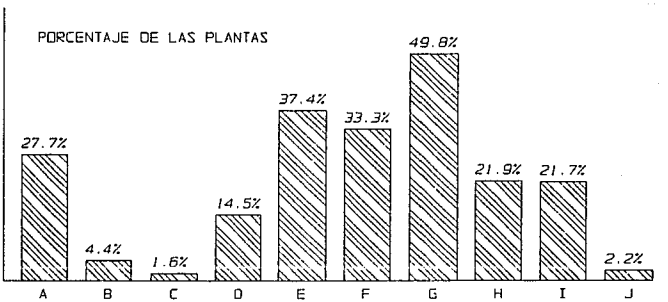


Figura 2.4 GENTE QUE HACE LOS PROGRAMAS DE CONTROL NUMERICO

2.3.2 Métodos utilizados para la programación

Prácticamente el 50% de los usuarios utilizan el *MDI*[5] *(fig 2.5)*. Puede mencionarse el hecho de que el 28% de los talleres siguen programando manualmente sobre cinta perforada. La suma de los porcentajes rebasa el 100%, ya que en un mismo taller se aplican diferentes métodos de programación.

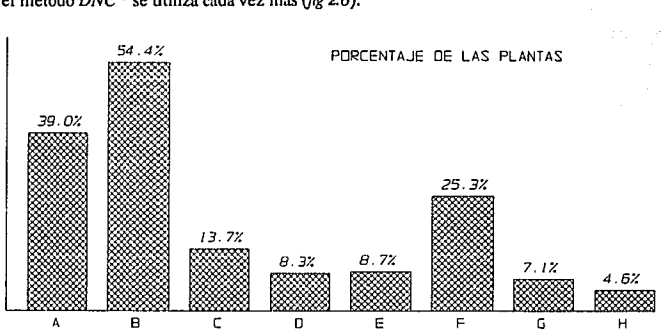


A. GENERACION MANUAL DE PROGRAMAS EN CINTA PERFORADA
B. TIEMPO COMPARTIDO DE COMPUTADORA CON UNA ORGANIZACION DE SERVICIO
C. GENERACION Y ESCRITURA DE PROGRAMAS POR UNA ORGANIZACION EXTERNA DE SERVICIOS
D. TIEMPO COMPARTIDO EN UNA COMPUTADORA PROPIEDAD DE LA EMPRESA
E. PROGRAMAS GENERADOS A PARTIR DE UNA BASE DE DATOS DE CAD
F. PERSONAL Y COMPUTADORAS INTERNOS GENERAN LOS PROGRAMAS
G. (MDI) INGRESO DIRECTO DEL PROGRAMA EN EL CONTROL DE LA MAQUINA
H. GENERACION EN PLANTA UTILIZANDO SOFTWARE INCLUIDO EN EL CONTROLADOR
I. PROGRAMACION EN COMPUTADOR PERSONAL Y SOFTWARE SUMINISTRADOS POR EL PROVEEDOR DE LA MAQUINA COMO UN PAQUETE COMPLETO
J. OTROS

Figura 2.5 METODOS UTILIZADOS PARA LA PROGRAMACION

2.3.3 Métodos utilizados para ingresar el programa a la máquina.

De nuevo sigue siendo la cinta perforada un método muy utilizado, aunque en decadencia. El método más utilizado es el de la cinta magnética, y es interesante notar que el método *DNC* [6] se utiliza cada vez más (*fig 2.6*).



A. ENTRADA Y CORRIDA DEL PROGRAMA DESDE CINTA PERFORADA
B. ENTRADA DEL PROGRAMA DESDE CINTA PERFORADA Y CORRIDA DESDE EL CONTROL DE LA MAQUINA
C. TRANSPORTE DEL PROGRAMA AL CONTROLADOR A TRAVES DE DISCOS DE 5 1/4"
D. TRANSPORTE EN DISCOS DE 3 1/2"
E. TRANSPORTE DEL PROGRAMA POR OTRO MEDIO DE ALMACENAMIENTO DE DATOS
F. TRANSMISION DE PROGRAMAS DIRECTAMENTE DESDE LA COMPUTADORA HASTA EL CONTROLADOR (DNC)
G. TRANSPORTE DEL PROGRAMA EN UNA UNIDAD DE ALMACENAMIENTO Y PROGRAMACION
H. OTROS

Figura 2.6 METODOS DE INGRESO DE LOS PROGRAMAS A LA MAQUINA CNC

NOTAS DEL CAPITULO 2

[1] El movimiento principal es aquél que determina las fuerzas de corte entre la herramienta y el material.
[2] El movimiento secundario es aquél que determina la forma de la pieza a fabricar y en menor grado influye en las fuerzas de corte.
[3] En algunos casos se requiere hacer modificaciones en el diseño para que la máquina sea capaz de efectuar el maquinado.
[4] Tomado de *Modem Machine Shop NC/CIM 1990 Guidebook*, Gardner Publications Inc., Ohio, E.U.A., 1990.
[5] *MDI*. Es el ingreso de datos directamente en el control de la máquina.
[6] *DNC*. Es la comunicación directa de la máquina de control numérico con un sistema de manejo de datos, por ejemplo, una computadora.

CAPITULO 3

EL PROBLEMA DE LA SIMULACION

Hasta el momento se ha tratado el concepto de la máquina herramienta con el apoyo del control numérico.

El control numérico no es una nueva técnica de maquinado, es un nuevo concepto para el control de las máquinas y, como tal, es un nuevo enfoque para el maquinado.

Alrededor del control numérico se ha desarrollado toda una industria para dar apoyo a este tipo de trabajo, cuya diversidad abarca desde firmas productoras de máquinas herramienta, pasando por los fabricantes de controladores, sensores y servomotores, hasta compañías dedicadas a la maquila de programas de partes para control numérico.

Una de estas industrias, que ha tomado gran auge, es la del desarrollo de *software* y, en este sentido, el concepto de simulación es muy importante.

3.1 EL CONCEPTO DE SIMULACION

Es una técnica mediante la cual es posible reproducir el funcionamiento de un sistema o una máquina sin que esta última intervenga.

Dicho en otras palabras, es una herramienta para la verificación de un proceso, sin que éste se lleve a cabo. Concretamente en el control numérico se trata de realizar la secuencia de movimientos tabulados y programados a fin de obtener la pieza deseada sin riesgos para la máquina, la pieza de trabajo, la herramienta o el operador.

Hablando de control numérico existen principalmente tres tipos de simulación:

a) En la máquina. Aunque en este tipo de simulación la máquina herramienta interviene directamente, no se contradice la definición dada, ya que las condiciones de trabajo no se establecen como en el maquinado.

Una técnica común es trabajar la máquina en vacío, esto es, sin la pieza a maquinar y aún sin la herramienta. Otro método es el de sustituir el material a maquinar por un material más blando como puede ser madera o algunas resinas o plásticos, utilizando las herramientas.

Esta técnica tiene la ventaja de que se puede acelerar la velocidad del proceso en comparación a la velocidad real y el resultado de la misma es copia fiel del maquinado. El inconveniente es que invierte tiempo productivo de la máquina y en algunos casos representa un costo extra.

b) En el controlador. Muchas máquinas de control numérico a nivel industrial tienen un simulador integrado al controlador que presenta en una pantalla el resultado de un maquinado; son altamente confiables, puesto que son mucho más rápidos en comparación con la simulación en la máquina.

Una pequeña desventaja es que no siempre abarcan todas las posibilidades del sistema y, además, el controlador consume tiempo productivo de maquinado, aunque éste es relativamente corto. Este problema ha sido solucionado por algunos fabricantes mediante el desarrollo de controladores separados de la máquina.

c) Sistema externo. Este sistema suele ser una computadora, y en consecuencia es necesario el desarrollo del *software*. Los sistemas *CAM* son ampliamente usados en las computadoras ya que permiten la interacción con sistemas *CAD* y son muy versátiles. Por otro lado existen varios programas para la interpretación de códigos de control numérico y la representación de los resultados.

Estos métodos tienen la ventaja de ser sumamente rápidos y la modificación de los programas se realiza de manera relativamente sencilla, además, no consumen

tiempo productivo de la máquina y basta, en algunos casos, con un solo sistema para proporcionar a varias máquinas de control numérico programas verificados (*DNC*).

La principal desventaja es que no siempre se reproducen las características de la máquina de control numérico, y esto obliga a hacer algunas modificaciones al programa antes de ejecutarlo. Los postprocesadores atenúan este problema acoplando los resultados obtenidos del *software* al controlador.

3.2 UTILIZACION DEL SIMULADOR EN LAS COMPUTADORAS COMO HERRAMIENTA DE TRABAJO

De entrada resulta ideal utilizar la computadora para la simulación de programas de control numérico. Se cuenta con una serie de datos tabulados y comandos a ejecutar y la computadora tiene la capacidad de manejar una gran cantidad de datos a grandes velocidades.

A final de cuentas esto resulta obvio, ya que las computadoras y los controladores de las máquinas de control numérico tienen la misma naturaleza.

La tendencia a la baja en los precios de las computadoras ha hecho que éstas sean más accesibles a las personas, e incluso algunos fabricantes de máquinas los incluyen en la venta de los equipos. Por otro lado, la capacitación de gente para el uso de estas máquinas ha encontrado un gran apoyo en las microcomputadoras o computadoras personales (PC).

Partiendo del proceso de manufactura del apartado 2.2.2, la simulación se realizará inmediatamente después de la programación. Algunos operadores acuden a la simulación constantemente mientras se programa.

El objeto de un simulador es la verificación de que una secuencia de órdenes dada dará como resultado la geometría deseada.

Es importante contar con todos los datos necesarios[1] para la simulación cuando el resultado que se busca no es solamente la verificación de la geometría. La simulación puede proporcionar el tiempo aproximado de maquinado y, sabiéndolo aprovechar, los

procesos de maquinado pueden ser optimizados no sólo en tiempo, sino también en la vida del herramental y en el acabado de la pieza.

3.3 CARACTERISTICAS DEL SIMULADOR Y EL CONTROLADOR

Como todo simulador está orientado a un sistema, máquina o mecanismo, es necesario determinar los requisitos del mismo.

3.3.1 El controlador

Se trata de una máquina COMPACT 5 CNC de la firma **EMCO Mayer & Co.** de Austria. El controlador basa su funcionamiento en el *software* A6C 114 004. Tiene capacidad para 210 bloques de comandos, programación de radios, programación en sistema absoluto o incremental, grabador de cintas magnéticas integrado e interface RS-232 para comunicación externa. En el apéndice D se presenta la información técnica de esta máquina.

3.3.2 Códigos de control

El *software* A6C 114 004 incluye el uso de ciclos enlatados en las direcciones *G73*, *G78*, *G81*, *G82*, *G83*, *G84*, *G85*, *G86*, *G88* y *G89*; cambio automático de herramientas (*M06*) y programación de control de radio según la norma DIN 66025 sin limitación de ángulos.

G00 - Movimiento rápido

G01 - Movimiento de corte

G02 - Interpolación circular en el sentido del reloj

G03 - Interpolación circular en el sentido antihorario

G04 - Pausa por tiempo

G21 - Línea en blanco

G25 - Llamada de subrutina

G27 - Salto a línea

G33 - Maquinado de rosca

G70 - Programación en pulgadas

G71 - Programación en milímetros

G73 - Ciclo de barrenado con incrementos de 200 u.

G78 - Ciclo de roscado

G81 - Ciclo de barrenado de un solo paso

G82 - Ciclo de barrenado, un solo paso tiempo de espera

G83 - Ciclo de barrenado con salida al punto de partida

G84 - Ciclo de trabajo longitudinal (cilindrado)

G85 - Ciclo de rimado

G86 - Ciclo de tronzado

G88 - Ciclo de trabajo transversal (careado)

G89 - Ciclo de rimado con tiempo de espera

G90 - Programar en sistema absoluto

G91 - Programar en sistema incremental

G92 - Marca de origen con sistema absoluto

G94 - Avance por minuto

G95 - Avance por revolución

M00 - Pausa de programa

M03 - Enciende el husillo en el sentido del reloj

M05 - Apaga el husillo

M06 - Cambio de herramienta

M17 - Fin de subrutina

M30 - Fin de programa

M98 - Compensación automática de juego mecánico.

M99 - Coordenadas de centro para interpolación circular

3.3.3 Formato del programa

El *software* A6C 114 004 reconoce los bloques de programa con una extensión de 32 caracteres.

En este espacio se encuentran 6 campos, a saber:

Campo No. 1 (*N*) Número de línea.

Inicia en el número 0 y termina en el 209. Soporta hasta 3 dígitos.

Campo No. 2 (*G*) Comandos.

Acepta los códigos G y M. Utiliza un caracter para la letra y dos para el número (*M06*, por ejemplo). Cuando se introduce un código *G*, esta letra no aparece.

Campo No. 3 (*X*) Coordenadas en *X*.

Acepta 4 dígitos y un signo negativo (si se necesita) a la izquierda. En el caso del comando *M99*, el signo se reemplaza por la letra *I*. Define diámetros.

Campo No. 4 (*Z*) Coordenada en *Z*.

Acepta 5 dígitos y un signo negativo a la izquierda. En el caso del comando *M99* el signo se reemplaza por la letra *K*. Define longitudes.

Campo No. 5 (*F*) Velocidad de avance.

Acepta 3 dígitos y una letra del lado izquierdo, dependiendo del comando es la letra que aparece.

Comando	Letra	Significado
M06	*T*	No. de herramienta
G78	*K*	Paso de la rosca
G33	*K*	Paso de la rosca
G25	*L*	No. de línea
G27	*L*	No. de línea
Otras	No aparece letra	Avance

Campo No. 6 (*H*) Profundidad de corte.

Acepta 3 dígitos sin letra ni signo.

Todos los caracteres utilizados obedecen el código *ASCII*, como se comentó en el capítulo primero.

3.3.4 El simulador

Se requiere un programa de tipo didáctico para computadora capaz de realizar la simulación en pantalla en tiempo menor al real y con la mayor resolución posible.

Debe tener definición de las herramientas más usuales en el trabajo rutinario, un editor con capacidad mínima comparable al *software* A6C 114 004 de **EMCO** y un controlador o gestor de archivos para el almacenamiento de programas de control numérico.

Dado que la principal utilización de la máquina COMPACT 5 CNC es la enseñanza, es conveniente orientar el manejo del programa al aprendizaje de los alumnos en el uso de comandos y lenguaje generalizado de control numérico.

Como requisito especial está la comunicación entre la computadora y la máquina de control numérico, para la ejecución inmediata de los programas editados y aprobados.

NOTAS DEL CAPITULO 3

[1] Velocidad de corte, velocidad del husillo, material, potencia de la máquina, etcétera.

CAPITULO 4

DESARROLLO DEL PROGRAMA TurnUP

Como resultado de los requerimientos mencionados en el capítulo anterior, se desarrolló un programa para computadora denominado TurnUP. El nombre del programa se deriva de su aplicación. Es un simulador de programas de un torno de control numérico con fines didácticos para el laboratorio de control numérico de la Universidad Panamericana.

El programa se desarrolló con el lenguaje de programación *Turbo Pascal v. 5.5*, de la compañía Boorland, de los Estados Unidos.

El presente capítulo dará una visión general del programa mediante la revisión de los módulos que lo conforman, a saber:

- Gestión de archivos
- Edición de programas
- Simulación de programas
- Comunicación o transmisión de códigos

4.1 ESTRUCTURA GENERAL DEL PROGRAMA

El lenguaje de programación *Pascal* da la posibilidad de trabajar en forma modular, por lo que el programa TurnUP está compuesto por una gran cantidad de subrutinas o procedimientos.

4.1.1 Variables generales

Para el trabajo conjunto de los procedimientos se han definido constantes y variables generalizadas, esto es, que todos los procedimientos tienen acceso a ellas.

Las constantes y variables se definen de la manera siguiente:

a) Variables generales

```
CONST
  UtilDir = '';                          {Ruta de búsqueda para utilerías del programa}
  GraphDir = '';                         {Ruta de búsqueda para archivos *.bgi}
  ProgDir:Char = 'C';                    {Ruta inicial para programas CNC}
  Version = 'TurnUP v1.2';               {Nombre del programa}
  MaxLineas = 299;                       {Líneas de comando}
  MaxDrv = 110;                          {Parámetros de Driver de gráficos}
  MaxNiveles = 5;                        {Nivel máximo de anidación del programa CNC}
  MaxHerram = 8;                         {Número máximo de herramientas}
  BrocaMin = 1;                          {Tamaño mínimo de la broca}
TYPE
  TipoSistema = Array [0..2] of String;
  TipoHerramienta = Array [1..MaxHerram] of String[20];
  TipoOrdenHerram = Array [1..MaxHerram] Of byte;
CONST
  Sistemas:TipoSistema = (' ','Sistema Internacional','Sistema Inglés');
  Herramientas:TipoHerramienta = ('DERECHA',
                    'IZQUIERDA',
                    'NEUTRA',
                    'TRONZADORA',
                    'ROSCADORA',
                    'BROCA',
                    'CORTADOR INTERNO',
                    'ROSCA INTERNA');
TYPE
  String3 = String[3];
  String4 = String[4];
  String8 = String[8];
  String30 = String[30];
  String80 = String[80];
  TipoLinea = RECORD
        Command:String3;
        CoordX,CoordZ:String8;
        Velocidad:String4;
        ProfCorte:String4;
      END;
  TipoUnidad = 0..2;
  {0. No definido, 1. Sistema Internacional, 2. Sistema Inglés}
```

b) Variables del programa CNC.

```
  Linea:array [0..MaxLineas] of TipoLinea;
  Descripcion:String[25];
  Programador:String30;
  Dimensiones:Record
        Diametro,Largo:Real;
```

```
        End;
Material:String[20];
Fecha:String[10];
OrdHerram:TipoOrdenHerram;
Unidades:TipoUnidad;
```

c) Variables de control de programa

```
Comando:Array [0..199,2..6] of Char;
N:Integer;
SubLinea:Array [1..MaxNiveles] of Integer;
SubNivel:Integer;
```

d) Variables de manejo de archivo

```
Programa:String30;
Lach:Text;
Drive:String[1];
IOStat:Byte;
```

e) Variables internas

```
Renglon:Byte;
I:Integer;
Car:Char;
St:String;
Flag:boolean;
TotalCommands:Byte;
Modulo:String30;
```

f) Variables de control de gráficos

```
DriverName:String;
Driver,Modo:Intoger;
Drv:Array [1..MaxDrv] of Integer;
DiametroBroca:Real;
RelX,RelY:Real;
```

Una variable de importancia relevante es la que define los parámetros generales de colores y graficación denominada *DRV*. Se trata de un arreglo de enteros como se puede ver en la tabla anterior.

Al arrancar el programa, los valores o parámetros de graficación y colores deben ser cargados; de lo contrario, el programa no correrá. Estos valores son constantes y están contenidos en una serie de archivos de texto que a continuación se enlista.

Parámetros para tarjeta de video:

CGATORN.DRV	*CGA (Color Graphics Adapter)*
HERCTORN.DRV	*HERCULES*
EGATORN.DRV	*EGA (Enhaced Graphics Adapter)*
VGATORN.DRV	*VGA (Video Graphics Adapater)*

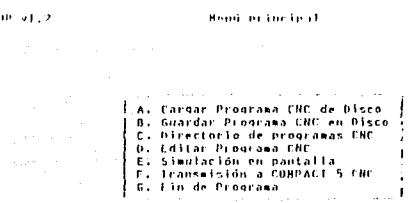
La siguiente tabla presenta el significado de cada parámetro y el valor correspondiente para cada tarjeta de video.

DRV[#]	DEFINICION DEL PARAMETRO	CGA	HERC	EGA	VGA
001	Color de fondo de pantalla	0	0	1	1
002	Color de letras de la línea superior (estatus)	0	0	14	14
003	Color de fondo de la línea superior	7	1	4	4
004	Color letras línea de mensaje	0	0	14	14
005	Color fondo línea de mensaje	7	1	4	4
006	Color líneas de borde	15	1	0	0
007	Color fondo borde	0	0	3	3
008	Color fondo cuadro	0	0	3	3
009	Color texto apagado	7	1	0	0
010	Color texto encendido	15	15	15	15
011	Color fondo encendido	0	0	3	3
012	Color letras de lectura de campo	0	15	15	15
013	Color fondo de lectura de campo	7	1	7	7
014	Color texto en cuadro	7	1	0	0
015	Color texto de editor apagado	7	1	0	0
016	Color texto editor encendido	15	15	15	15
017	Color texto fondo encendido	7	0	4	4
018	Color línea superior de comandos de editor	7	0	15	15
019	Color funciones de editor	15	0	8	8
030	Puerto de comunicaciones (0 = COM1: 1 = COM2:)	0	0	0	0
050	Base de cálculo de herramientas	10	10	10	10
051	Color de las herramientas	15	1	3	3
052	Color del texto para definición de herramientas	7	1	15	15
053	Color del recuadro de selección	15	1	14	14
054	Color del número de herramienta a seleccionar	7	1	2	2
055	Coordenada X de 'HERRAMIENTA #'	220	220	220	220
056	Coordenada Y de 'HERRAMIENTA #'	182	182	300	420
057	Coordenada X Herramienta #1	50	50	50	50
058	Coordenada Y Herramienta #1	20	20	41	60
059	Coordenada X Herramienta #2	210	210	210	210
060	Coordenada Y Herramienta #2	20	20	41	60
061	Coordenada X Herramienta #3	330	330	330	330
062	Coordenada Y Herramienta #3	20	20	41	60
063	Coordenada X Herramienta #4	460	460	460	460
064	Coordenada Y Herramienta #4	20	20	41	60

065	Coordenada X Herramienta #5	570	570	570	570
066	Coordenada Y Herramienta #5	20	20	41	60
067	Coordenada X Herramienta #6	60	60	60	50
068	Coordenada Y Herramienta #6	110	110	215	290
069	Coordenada X Herramienta #7	200	200	260	300
070	Coordenada Y Herramienta #7	110	110	230	300
071	Coordenada X Herramienta #8	380	380	460	480
072	Coordenada Y Herramienta #8	110	110	230	300
073	Coordenada X Herramienta #9 (No existente)	550	550	550	0
074	Coordenada Y Herramienta #9 (No existente)	110	110	230	0
075	Color Recuadro	15	1	14	14
076	Línea superior recuadro	10	10	15	15
077	Color de 'DEFINICION DE HERRAMIENTAS'	7	1	4	4
078	Coordenada Y del renglón superior	11	11	20	27
079	Coordenada Y del renglón inferior	22	22	32	45
080	Coordenada X del letrero #1	6	6	6	6
081	Coordenada X del letrero #2	21	21	21	21
082	Coordenada X del letrero #3	40	40	40	40
083	Coordenada X del letrero #4	54	54	54	54
084	Coordenada X del letrero #5	69	69	69	69
085	Coordenada X del letrero #6	8	8	12	15
086	Coordenada X del letrero #7	22	22	33	38
087	Coordenada X del letrero #8	45	45	59	62
088	Coordenada X del letrero #9 (No existente)	65	65	80	65
089	Escala de las herramientas en su definición	9	0	10	10
090	Color de líneas de separación de áreas de trabajo	7	0	15	15
091	Color Tracking	1	0	15	15
092	Color de nombre del archivo y letras de mensajes	7	0	15	15
093	Color de línea en ejecución	15	0	7	7
094	Color de código numérico	7	0	14	14
095	Color de *chuck*	15	0	2	2
096	Color de ejes	15	0	4	4
097	Color de material	7	0	6	7
098	Color de estatus	15	0	14	14
099	Patrón de material	1	1	1	1
100	Altura de las letras de rastreo	1	1	1	1
101	Distancia del extremo izquierdo a la pieza de trabajo	50	20	50	50
102	Número de renglones de programa mostrados (simulador)	3	3	4	4
103	Tamaño de ejes en Z	2	2	2	2
104	Tamaño de ejes en X	2	2	2	2
105	Tamaño de letras de ejes	3	3	3	3
106	Pixels por avance rápido	50	50	50	50

4.1.2 Estructura del menú principal

Al ejecutarse el programa, aparece en la pantalla el siguiente menú:



Los tres primeros corresponden a la gestión de archivos y los tres siguientes según su denominación

4.2 GESTION DE ARCHIVOS

Para dar mayor facilidad al usuario, se dispuso un formato en archivo de texto. En el apéndice E se muestran varios programas para ejemplificar el formato.

El manejo de archivos realiza tres tareas:

- Cargar los programas desde disco
- Grabar los programas al disco
- Mostrar el directorio de los programas de control numérico en el disco.

Al ser cargado un programa en la memoria, éste se asigna a distintas variables:

VARIABLE	DEFINICION
PROGRAMADOR	Nombre de quien hizo el programa
DESCRIPCION	Explicación de lo que hace el programa
UNIDADES	0 = No definido
	1 = Milímetros (SI)
	2 = Pulgadas (sistema inglés)
DIMENSIONES	Tamaño de la pieza en bruto
LARGO	
DIAMETRO	
HERRAMIENTAS	Secuencia de las herramientas utilizadas
LINEA(N)	Bloques de programa conteniendo cada uno de los campos mencionados en el apartado 3.3.3

Todos los archivos definidos para programas de control numérico tienen la extensión CNC y se ubican en el directorio raíz del controlador de discos que se esté utilizando.

Como extensión del módulo de gestión de archivos se encuentra el procedimiento SAVE, que se utiliza tanto en la opción GRABAR PROGRAMA EN DISCO del menú principal, como en el editor. En este último se activa el procedimiento simplemente con apretar la tecla F2.

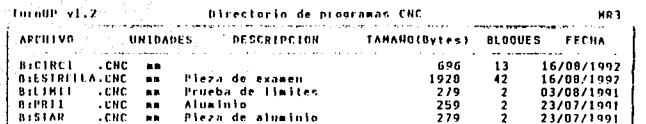
Por último, la opción de DIRECTORIO DE PROGRAMA CNC muestra el contenido de programas en el disco que se esté utilizando. En el desplegado de la pantalla (*fig 4.2*) se incluye la siguiente información:

ARCHIVO - Muestra el nombre del archivo con extensión CNC

UNIDADES - Milímetros o pulgadas

DESCRIPCION - Explicación de lo que hace el programa

TAMAÑO - Espacio que ocupa el archivo en disco (bytes)



```
TurboDP v1.2          Directorio de programas CNC                    MR3

 ARCHIVO        UNIDADES      DESCRIPCION        TAMAÑO(Bytes)  BLOQUES   FECHA
 ---------------------------------------------------------------------------
 B:CIRCL  .CNC   mm                                     696      13    16/08/1992
 B:ESTRELLA.CNC  mm   Pieza de examen                 1920      42    16/08/1992
 B:LIMIT  .CNC   mm   Prueba de limites                279       2    03/08/1991
 B:PRJ1   .CNC   mm   Aluminio                        259       2    23/07/1991
 B:STAR   .CNC   mm   Pieza de aluminio               279       2    23/07/1991
 B:SUB    .CNC   mm   Prueba de subrutinas            978      19    26/08/1991
 B:ROSCA  .CNC   mm                                    533       9    11/02/1992
 B:BORCA  .CNC   in                                    450       7    11/02/1992
 B:BROCA  .CNC   mm                                    411       6    11/02/1992
 B:GROOVING.CNC  mm                                    452       7    11/02/1992
 B:FACING .CNC   mm                                    408       6    16/08/1992
 B:BOQUILLA.CNC  mm   Boquilla                        7162      48    11/02/1992
 B:PPRUEBA .CNC  mm   xxxxxxxxxxxxxxxxxxxxxxxx        267       2    25/09/1991
 B:EMCO   .CNC   mm   Programa del curso EMCO         1421      30    01/01/1980

                        94Kb libres en el Drive B
```

BLOQUES - Número de bloques de programa

FECHA - Fecha de la última actualización

4.3 EDICION DE PROGRAMAS

En términos generales se puede decir que la función del editor es modificar los valores contenidos en las variables mencionadas en el apartado anterior; sin embargo, el editor realiza mucho más.

De entrada, en la opción EDITAR PROGRAMA CNC del menú principal aparecen dos opciones: programa en memoria o programa nuevo. La forma en que se reconoce si algún programa está en memoria es mediante la verificación de la línea cero. Cuando la memoria no contiene programa, este bloque contiene el comando *M30* que, dicho en términos de control numérico, debe entenderse: "la primera instrucción a ejecutar es finalizar el programa" (ver apéndice C en código *M30*). Por esta razón, sólo puede ser programado un código *M30* y éste estará siempre al final del programa.

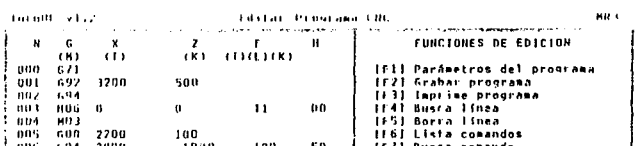
Si se elige la opción de PROGRAMA NUEVO, todas las variables de control de programa serán inicializadas. Como primer paso se piden las unidades, luego los datos del

programador y la descripción del programa, e inmediatamente las dimensiones de la pieza a maquinar en bruto (diámetro y largo).

Una vez dados estos datos se define el nombre del programa y el editor es inicializado con dos líneas.

Bloque #	Comando
0	*G70* o *G71* (Programación en milímetros o pulgadas respectivamente).
1	*M30* Fin de programa

Los bloques de programa se irán insertando entre estos dos comandos, de tal manera que el bloque que contiene al comando *M30* se desplazará hacia adelante en la numeración ocupando siempre el último bloque de programa.

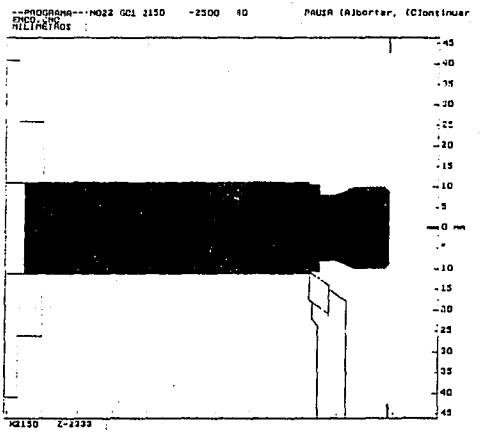


Durante la inserción de datos en los nuevos bloques, los campos son verificados en su contenido, a fin de que los límites de la máquina no sean rebasados. Esto garantiza que la máquina de control numérico reconocerá la instrucción, mas no que el programa esté correcto.

4.4 SIMULACION DE PROGRAMAS

Este módulo se define por las funciones o comandos de control numérico que puede ejecutar y la presentación del proceso. Existen tres modalidades:

- Ejecución continua
- Ejecución por bloques (incluye pausas)
- Simulación rápida (sólo trayectoria de la herramienta)



4.4.1 Preparación de la pantalla

Antes de que el programa sea ejecutado, varios cálculos y áreas de trabajo deben definirse.

Los cálculos más importantes son los siguientes:

a) Definición de escala. La escala del dibujo depende de las dimensiones de la pieza en bruto y será la menor de dos escalas definidas en cada eje:

$$\text{Escala X} = \frac{H * 0.8}{L * RelH}$$

$$\text{Escala Z} = \frac{W * 0.8}{L * RelW}$$

Donde: H = Altura de la pantalla en pixels[1]

W = Ancho de la pantalla en pixels

D = Diámetro de la pieza en bruto

L = Largo de la pieza en bruto

RelH y RelW = Valores de corrección según la tarjeta de video que se utilice.

El factor de 80% se utiliza para que la pieza dibujada no abarque toda la pantalla.

b) Factor de corrección de unidades. Dado que la escala se define a partir de milímetros o pulgadas y el programa de control numérico maneja centésimas de milímetro o milésimas de pulgada[2], es necesario aplicar un factor de corrección a los cálculos que se realicen en la ejecución de las trayectorias de la herramienta. Este factor se define de la siguiente manera:

100 para milímetros

1000 para pulgadas

c) Parámetros de inicio de programa. Se consideran los siguientes valores:

Herramienta en uso: 1

Sistema de coordenadas: incremental

Velocidad del husillo: 2500 RPM (Sólo aplicable en el comando *M03*)

Nivel de anidación de subrutinas: 0

Ubicación de la herramienta: en la esquina inferior derecha del material en bruto mostrado en la pantalla.

4.4.2 Sistemas de coordenadas

Existen tres sistemas de coordenadas funcionando simultáneamente. Son:

a) X, Z. Son las coordenadas de la herramienta definidas de acuerdo al programa de control numérico. Trabaja siempre en sistema absoluto. X define diámetro. Todos los procedimientos trabajan con este sistema (*fig 4.5*).

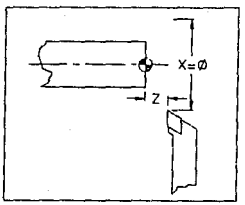


Figura 4.5 PRIMER SISTEMA DE COORDENADAS

b) X_0, Z_0 Origen (en sistema absoluto)

X_1, Z_1 Posición inicial de la herramienta

X_2, Z_2 Posición final de la herramienta

Su valor se define en pixels en relación al origen del sistema de la computadora que se ubica en la esquina superior izquierda de la pantalla (*fig 4.6*).

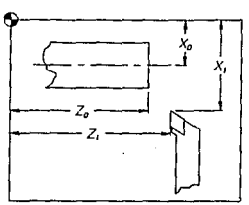


Figura 4.6 SEGUNDO SISTEMA DE COORDENADAS

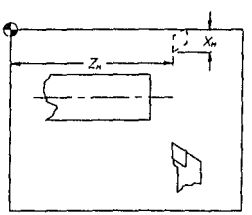


Figura 4.7 TERCER SISTEMA DE COORDENADAS

c) *Xm*, *Zm*. Coordenadas en pixels, también respecto al origen de la pantalla. La diferencia respecto a los anteriores es que éstas definen la posición "espejo" de la herramienta en relación al eje de giro de la pieza (*fig 4.7*).

La conversión de un sistema a otro se realiza de la siguiente manera:

$$X_2 = X_0 + \frac{\text{Escala} * \text{RelH} * X}{2 * \text{UnitFactor}}$$

$$Z_2 = Z_0 + \frac{\text{Escala} * \text{RelW} * Z}{\text{UnitFactor}}$$

$$Xm = X_0 + \frac{\text{Escala} * \text{RelH} * X}{2 * \text{UnitFactor}}$$

$$Zm = Z_2$$

Como el principal sistema de coordenadas (el que define el controlador) trabaja siempre en sistema absoluto dentro del programa, es necesario convertir los valores incrementales a absolutos. Para esto se recurre constantemente a un par de fórmulas que a continuación se presentan:

$$Zf = Zf + Zi * (2 - \text{Incr})$$

$$Xf = Xf * (3 - \text{Incr}) + Xi * (2 - \text{Incr})$$

4.4.3 Movimiento de la herramienta

La mayoría de los comandos basan su funcionamiento en movimientos lineales, es por eso que se desarrolló un procedimiento especial para el movimiento lineal de la herramienta.

En primer lugar se necesitan las coordenadas de inicio y fin de trayectoria: *Xi* y *Zi*; *Xf*, *Zf* (*fig 4.8*). Los valores iniciales corresponden a la coordenada actual de la herramienta y los finales son suministrados al procedimiento dependiendo de los parámetros del comando que se esté ejecutando.

Se determina en qué eje se hará el mayor desplazamiento y en base a éste se hace la segmentación de la línea

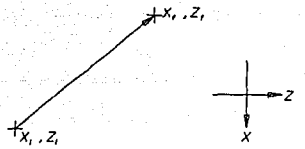


Figura 4.8 INTERPOLACION LINEAL

$$\#\ \text{Segmentos} = \frac{Xf - Xi}{\text{UnitFactor}} * \text{Escala} * \text{RelH} * 2$$

cuando el recorrido mayor es en el eje X

$$\#\ \text{Segmentos} = \frac{Zf - Zi}{\text{UnitFactor}} * \text{Escala} * \text{RelW}$$

cuando el recorrido mayor es en el eje Z

Ya que se tiene la cantidad de segmentos, se determina qué distancia representa cada división:

$$\text{Paso } X = \frac{Xf - Xi}{\#\ \text{segmentos}} \qquad \text{Paso } Z = \frac{Zf - Zi}{\#\text{segmentos}}$$

Por último, estos valores se van incrementando a las coordenadas iniciales simultáneamente hasta que la coordenada final es alcanzada:

$X = X + \text{PasoX}$

$Z = Z + \text{PasoZ}$

Hasta que $Z = Zf$ y $X = Xf$

4.4.4 Comandos

G00, G01 No requiere de cálculos. Las operaciones aritméticas son realizadas por el procedimiento LINEAL.

G02, G03 Estos comandos son de interpolación circular y su ejecución se realiza por segmentos. Trabajan con el comando *M99*.

Primero se consideran los puntos inicial, final y central (coordenadas *I*, *K*) para determinar si los parámetros son válidos. El comando se ejecuta de derecha a izquierda. Se deben cumplir las siguientes condiciones para la ejecución de un comando *G02:*

Asumiendo los valores:

$$Zc = Zi + \text{Sgn}\,(Xf - Xi) * K$$

$$Xc = Xi / 2 + I$$

donde *Zc*, *Xc* son las coordenadas del centro de giro. (*Xc* no representa diámetro, sino radio).

Si *Xf* es menor a *Xi*, esto es, se parte de un diámetro mayor a uno menor, *Zc* debe ser menor o igual a *Zf* (*fig 4.9*).

Utilizando los cuadrantes, el centro sólo puede ubicarse en el tercero.

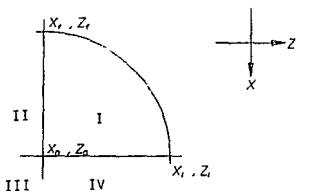


Figura 4.9 INTERPOLACION CIRCULAR
PRIMER CASO

Cuando *Xf* es mayor a *Xi*, la condición cambia: *Zc* debe ser mayor o igual a *Zi*. El centro sólo se puede ubicar en el cuarto cuadrante (*fig 4.10*).

La siguiente condición a cumplir es que los puntos final e inicial sean equidistantes al centro.

Cuando los parámetros han sido aceptados, se determinan los ángulos de inicio y final de la curva.

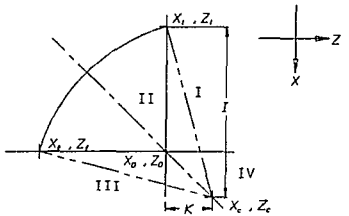


Figura 4.10 INTERPOLACION CIRCULAR SEGUNDO CASO

Se divide la curva en alrededor de 50 segmentos iguales para aproximar la circunferencia a través de líneas abarcando cada segmento.

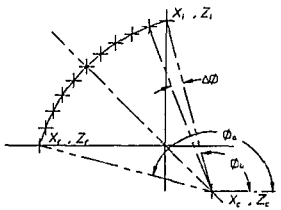


Figura 4.11 ANGULOS QUE DEFINEN LA CURVA

De esta manera, los puntos por los que se trazarán las líneas se definirán en sistema absoluto como sigue:

$Xr = Xc + \text{Radio} * \text{Sen}\Phi * 2$

$Zr = Xc + \text{Radio} * \text{Cos}\Phi$

donde $\Phi = \Phi + \delta\Phi$ en cada iteración y

$$\delta\Phi = \frac{\Phi b - \Phi a}{\#\ \text{segmentos}}$$

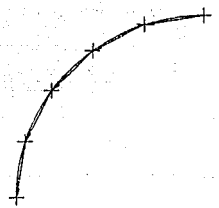


Figura 4.12 SEGMENTACION DE LA CURVA

G04 No requiere cálculos

G25, M17 La parte importante de estos comandos es la forma en que se lleva el registro

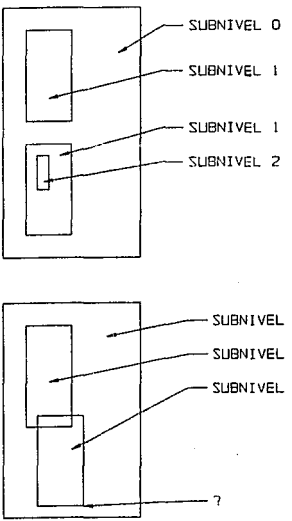


Figura 4.13 ANIDACION DE SUBRUTINAS

de anidación. Para llevar el control, se ha dispuesto un arreglo de enteros de la forma siguiente:

SUBLINEA contiene el número de línea de donde fue llamada la subrutina que origina la anidación SUBNIVEL.

El valor de SUBNIVEL es cero para la ejecución sin subrutinas. Es necesario utilizar el comando *G25* con el *M17* de manera que las subrutinas no se entrelacen.

G27 No requiere cálculos

G33 Este comando añade el manejo de paso de rosca en el movimiento de la herramienta. La única diferencia respecto al procedimiento LINEAL es el desplazamiento de medio paso de rosca hacia la izquierda de la herramienta espejo.

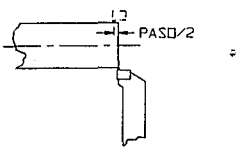


Figura 4.14 ROSCADO

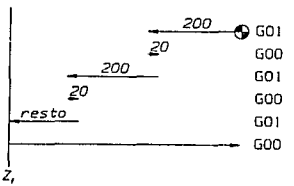


Figura 4.15 CICLO G73

En este caso el movimiento de la herramienta no se hace pixel por pixel sino a distancias definidas que representan el paso de la rosca.

G70, G71 Sólo hace la asignación del valor.

G70 UnitFactor = 1000 milésimas de pulgada

G71 UnitFactor = 100 centésimas de milímetro

No aplicable en la máquina COMPACT 5 CNC

G73 Con este comando se inicia la definición de ciclos enlatados. Concretamente, este comando se utiliza para el uso de brocas; por esto mismo, la coordenada en *X* debe ser cero o, lo que es lo mismo, la broca debe ubicarse sobre el eje de giro de la pieza para poder maquinar.

Dada la profundidad de barreno *Zf* se procede al maquinado con pasos de 2 mm. y retrocesos de 0.2 mm.

G78 Mucho más utilizado que el comando *G33*, sirve también para el maquinado de roscas añadiendo algunos movimientos.

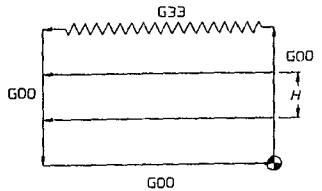


Figura 4.16 CICLO G78

El parámetro H define las profundidades sucesivas en cada pasada de la herramienta hasta llegar al diámetro final.

G81 Igual que *G73* con modificación en su secuencia.

Figura 4.17 CICLO G81

G82 Igual que *G73* con modificación en su secuencia.

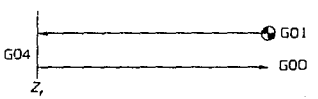


Figura 4.18 CICLO G82

G83 Igual que *G73* con modificación en su secuencia.

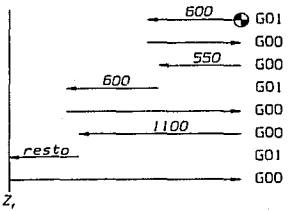


Figura 4.19 CICLO G83

G84 Este comando es muy usado, sobre todo en la etapa de desbaste. Su movimiento es el siguiente:

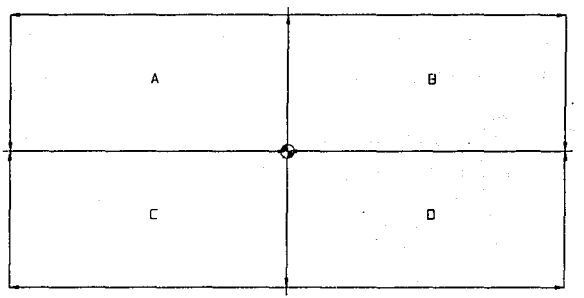


Figura 4.20 CICLO G84

aplicable en un cuadrante a la vez.

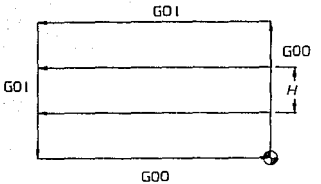


Figura 4.21 CICLO G84. DETALLE DE AVANCE DE PROFUNDIDAD

El parámetro H define las profundidades sucesivas en cada pasada.

G85 Igual que *G73* con modificación en su secuencia.

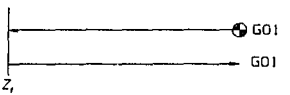


Figura 4.22 CICLO G85

G86 En este comando se utiliza principalmente la herramienta tronzadora. Sus movimientos de corte son en sentido radial

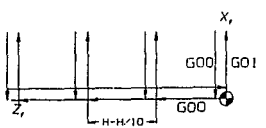


Figura 4.23 CICLO G86

donde H es el ancho de la herramienta.

G88 Este ciclo es igual al *G84* con una variación en la secuencia. Se utiliza principalmente para careado.

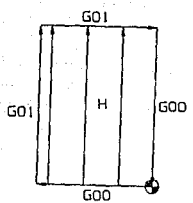


Figura 4.24 CICLO G88

G89 Igual que *G73* con modificación en la secuencia.

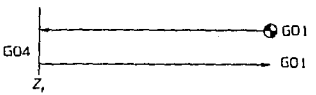


Figura 4.25 CICLO G89

G90 Sin cálculos

G91 Sin cálculos

G92 Sin cálculos. Se da la posición inicial a la herramienta.

G94 Sin cálculos

G95 Sin cálculos

M00 Sin cálculos

M03 Sin cálculos

M04 Sin cálculos

M05 Sin cálculos

M06 Requiere la asignación extra de un diámetro, el cual se aplicará en los comandos *G73, G81, G82, G83, G85* y *G89*, que trabajan con barrenos. Este campo no es aplicable en la máquina COMPACT 5 CNC.

El parámetro *T* funciona para la asignación directa de la herramienta en la simulación, a diferencia de la máquina COMPACT 5 CNC en la que el parámetro *T* es el número de avances de la torreta de herramientas. La compensación de herramienta no tiene efecto en la simulación.

M30 Sin cálculos

M98 No aplicable en la simulación, pero sí en la máquina COMPACT 5 CNC.

4.5 TRANSMISION

Este es el último módulo del programa e interrelaciona la computadora y la máquina de control numérico.

Son varios los requisitos que se deben cumplir para que los datos sean transmitidos en su totalidad correctamente.

a) Formato. Como se mencionó en el apartado 3.4, que trata de los requerimientos del sistema, cada bloque contiene 32 caracteres. Cada comando debe cumplir con un formato propio del renglón. Adicional al programa, se transmiten tres renglones extras para abrir y cerrar la transmisión. La primera línea indica a la máquina COMPACT 5 CNC que la transmisión dará inicio. El caracter de inicio es el signo de porcentaje (%).

La segunda línea conforma el renglón de definición de campos. Contiene el nombre de cada campo (*N, G, X, Z, F, H*) y un apóstrofe entre campos que indica el lugar en que finaliza dicho campo.

La tercera línea en discusión se encuentra al final del programa y es la que indica que la transmisión ha finalizado. Esta línea contiene un sólo caracter, el cual puede tener dos valores: *M* para milímetros o *I* para trabajar en pulgadas.

A continuación se muestra una tabla con el formato general de transmisión

```
%*!
....N^.G^...X.^....Z.^..F^..H.*!
....00.24.....*!
....01.00.-5999..32760.....*!
....02.01....01.....12.499....*!
....03.02.-1000.-.1000..09....*!
....04.03...200....200.199....*!
....05M99.I..00.K..200.....*!
....06.04....10.....*!
....07.21.....*!
....08M06..1222.-10000T.01....*!
....09.78...100.-.2000K120..20*!
....10.73.....-..100.100....*!
....11.81.....02.400....*!
....12.82.....-10200.100....*!
....13.83.....30..09....*!
....14.85.....2000.120....*!
....15.89.....2300.200....*!
....16.86.-.100...3000..50.100*!
....17.88...200....300.200..05*!
....18.84.-1000.-.2000.499..26*!
....19.90.....*!
....20.91.....*!
....21.92...100....200.....*!
....22.94.....*!
....23.95.....*!
....24.33.....-.5000K100....*!
....25M00.....*!
....26.27.....L.02....*!
....27.25.....L.98....*!
....28M03.....*!
....29M05.....*!
....30M08.....*!
....31M09.....*!
....32M17.....*!
....33M22.....*!
....34M23.....*!
....35M98....02.....03.....*!
....36M30.....*!
...M
```

donde:	.	espacio	ASCII 32
	^	apóstrofe	ASCII 96
	*	CR	ASCII 13
	!	LF	ASCII 10

b) Cable. Este depende de la máquina que se esté utilizando. La configuración para la máquina COMPACT 5 CNC con el software A6C 114004 se muestra en el siguiente diagrama[3].

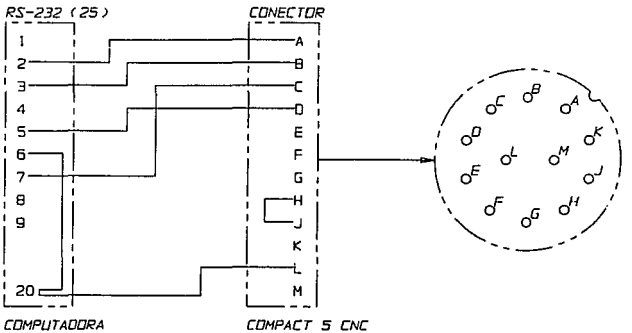


Figura 4.28 CONFIGURACION DEL CABLE DE COMUNICACIONES

La configuración que se presenta corresponde a 300 bps (*baud rate*)[4]. Colocando un puente entre el pin E y tierra, se trabajará a 110 bps.

La terminal corresponde al estándar RS232 con transmisión de señal a 20 mA.

c) Configuración del puerto de comunicaciones. A continuación se presentan los parámetros utilizados.

Puerto: Com1

Baud rate: 300

Paridad: None

Stop bits: 1

Word lenght: 7

NOTAS DEL CAPITULO 4

[1] Unidad de medida de las tarjetas de video. Equivale a un punto sobre la pantalla.
[2] Según el *software* A6C 114 004
[3] Proporcionado por la compañía EMAC (Equipo y Máquinas Computarizadas).
[4] Bits por segundo.

CONCLUSIONES

Resulta conveniente elaborar los programas de control numérico utilizando el simulador, ya que con éste se pueden ver los resultados del comando o programa que se esté elaborando mientras se programa una figura.

Con la ayuda del simulador, los programas no sólo pueden ser elaborados, sino también depurados, con el fin de optimizar el proceso de maquinado.

El programa TurnUP, que incluye gestión, edición, simulación y transmisión de programas, facilita la elaboración de estos últimos, pero sobre todo, reduce considerablemente el tiempo requerido desde el reconocimiento del plano de una pieza a maquinar, hasta el momento en que se cuenta con el programa adecuado para fabricarla.

No contando con el programa TurnUP, el proceso a seguir para ejecutar un programa era el siguiente:

1. Elaborar el programa en papel (*30 min a 1 hr, dependiendo de la complejidad*).
2. Transcribir el programa a la máquina COMPACT 5 CNC (*30 min promedio*).
3. Almacenar el programa en cinta magnética (*5 a 10 min*)
4. Simular el programa en la máquina con una plumilla sobre papel (*30 min promedio*).

TIEMPO APROXIMADO REQUERIDO: 100 min (en el mejor de los casos).

Por seguridad, los programas se graban en cinta magnética varias veces durante la captura de datos; esto aunado al tiempo necesario para ingresar los datos al controlador y para la simulación del programa consume enorme tiempo productivo de la máquina de control numérico.

Utilizando el programa TurnUP, se pasa por el siguiente proceso:

1. Elaboración del programa en computadora (*30 min promedio*).
2. Simulación del programa (*2 a 5 min*).
3. Almecenar el programa en disco flexible (*5 seg*).
4. Transmitir el programa a COMPACT 5 CNC (*3 seg*).
5. Corregir los parámetros de cambio de herramientas (*1 min*).

TIEMPO APROXIMADO REQUERIDO: 36 min (promedio).

Cabe mencionar que los puntos 1, 2 y 3 del proceso de programación con el paquete TurnUP suelen hacerse simultáneamente, es decir, se recurre constantemente a grabar el programa en disco (por seguridad) y a la simulación.

Otra ventaja del programa TurnUP es la capacidad de ser instalado en prácticamente cualquier computadora, siempre y cuando se cuente con tarjeta de gráficos. En consecuencia, la máquina de control numérico se utilizará únicamente para el maquinado y no para la captura y simulación de programas.

La posibilidad de verificar los comandos que se ejecutan de manera evidente y la fácil instalación en más de una computadora, hace del programa un gran apoyo para la enseñanza del control numérico, tanto a nivel escolar como a nivel de capacitación en empresas. Esto cobra importancia con la apertura tecnológica de la industria manufacturera mexicana ya que, para hacer frente a los nuevos retos tecnológicos, es necesario contar con personal capacitado en el área de control numérico, puesto que ésta es una de las ramas que más auge está tomando en nuestro país en los últimos años.

BIBLIOGRAFIA

A.A.V.V.
Machinery Handbook
23 Ed.

A.A.V.V.
Modern machine Shop, NC / CIM 1990 Guidebook
Gardner Publications Inc.
Cincinnati 1990

AMSTEAD, B. H.
Procesos de manufactura
C.E.C.S.A.
México 1988

BOOTHROYD, GEOFFREY
Fundamentals for metal machining and machine tools
Mc Graw Hill Book Company
U.S.A. 1975

DOYLE, LAWRENCE E.
Procesos y materiales de manufactura para ingenieros
Prentice Hall Hispanoamericana
México 1988

GONZALEZ NUÑEZ, JUAN
El control numérico en las máquinas herramienta
C.E.C.S.A.
México 1990

LURIE, G; KOMISSARZHEVSKAYA, V.
Estructura de las rectificadoras
Editorial MIR
Moscú 1975

MANFRED, WECK
Handbook of machine tools volume 3. Automation and controls
John Wiley and Sons
Londres 1980

POLLACK, HERMAN W.
Máquinas herramientas y manejo de materiales
Prentice Hall Hispanoamericana
México 1987

APENDICE A

LISTADO DEL PROGRAMA

A continuación se presenta el listado completo del programa TurnUP que incluye las unidades, utilerías y procedimientos utilizados en su elaboración. El orden en que se encuentran en este apéndice obedece al alfabético.

DEFINIT.PAS

```
UNIT DEFINIT;

INTERFACE

Uses Graph,Utileria;

PROCEDURE Herramienta(Tool,x,y,Color:Integer;Escala:Real);
FUNCTION Limits(Comando:String;Parametro:Char;Valor:LongInt):Boolean;

IMPLEMENTATION

PROCEDURE Herramienta(Tool,x,y,Color:Integer;Escala:Real);
    Var
        Radio,BaseHerramientas:Real;
        HerrColor,OriginColor:Byte;
        Bajo,Izq,Der:Integer;
    Procedure Mueve(dx,dy:Real);
        Begin
            MoveRel(Round(dx*BaseHerramientas*RelX/10),Round(dy*BaseHerramientas*RelY/10))
        End;
    Procedure Linea(dx,dy:Real);
        Begin
            LineRel(Round(dx*BaseHerramientas*RelX/10),Round(dy*BaseHerramientas*RelY/10))
        End;
    Begin
        BaseHerramientas:=Drv[50]*Escala;
        If Unidades = 2 Then BaseHerramientas:=BaseHerramientas/2.54;
        If Not (Abs(Tool) in [1..MaxHerram]) Then Tool:=1;
        OriginColor:=GetColor;
        If Color<>0
            Then
                SetColor(Drv[51])
            Else
                SetColor(0);
        HerrColor:=GetColor;
        MoveTo(x,y);
        Case Tool Of
            -1:Begin
                    Linea(0,-8);
                    Linea(7,-4);
                    Linea(0,8);
                    Linea(-7,4)
                End;
            -2:Begin
                    Linea(-7,-4);
                    Linea(0,-8);
                    Linea(7,4);
                    Linea(0,8);
                End;
            -3:Begin
                    Linea(4,-7);
                    Linea(-4,-7);
                    Linea(-4,7);
                    Linea(4,7);
                End;
            -4:Begin
                    Linea(-3,0);
                    Linea(0,-14);
                    Linea(3,0);
                    Linea(0,14);
                End;
            -5:Begin
                    Bajo:=y-Round(1.5*BaseHerramientas*RelY/10);
                    Izq:=x-Round(0.867*BaseHerramientas*RelX/10);
                    Der:=x+Round(0.867*BaseHerramientas*RelX/10);
                    For i:=y Downto Bajo do
                        Begin
                            Line(Izq,Bajo,x,i);
                            Line(x,i,Der,Bajo)
```

```
            End;
        MoveTo(Izq,Bajo);
        LineTo(x,y);
        LineTo(Der,Bajo);
        Linea(3.2,0);
        Linea(0,-5);
        Linea(-5,0);
        Linea(0,5);
    End;
-7:Begin
        Linea(0,8);
        Linea(7,4);
        Linea(0,-8);
        Linea(-7,-4);
    End;
1:Begin                                 {Derecha}
        Linea(7,4);
        Linea(0,8);
        Linea(-7,-4);
        Linea(0,-8);
        Mueve(1,8*4/7);
        Linea(0,5);
        Linea(2,2);
        Linea(0,30);
        Linea(10,0);
        Linea(0,-37);
        Linea(-6,-3.5);
    End;
2:Begin                                 {Izquierda}
        Linea(-7,4);
        Linea(0,8);
        Linea(7,-4);
        Linea(0,-8);
        Mueve(-1,8*4/7);
        Linea(0,5);
        Linea(-2,2);
        Linea(0,30);
        Linea(-10,0);
        Linea(0,-37);
        Linea(6,-3.5);
    End;
3:Begin                                 {Neutra}
        Linea(4,7);
        Linea(-4,7);
        Linea(-4,-7);
        Linea(4,-7);
        Mueve(-3.5,8);
        Linea(-1.5,2);
        Linea(0,40);
        Linea(10,0);
        Linea(0,-40);
        Linea(-1.5,-2);
    End;
4:Begin                                 {Tronzadora}
        Linea(-3,0);
        Linea(0,14);
        Linea(3,0);
        Linea(0,-14);
        Mueve(-0.5,14);
        Linea(0,30);
        Linea(-2,0);
        Linea(0,-30);
    End;
5:Begin                                 {Rosca externa}
        SetColor(0);
        Bajo:=y+Round(1.5*BaseHerramientas*RelY/10);
        Izq:=x-Round(0.867*BaseHerramientas*RelX/10);
        Der:=x+Round(0.867*BaseHerramientas*RelX/10);
        For i:=y to Bajo do
            Begin
                Line(Izq,Bajo,x,i);
                Line(x,i,Der,Bajo)
```

```
                End;
            SetColor(HerrColor);
            MoveTo(Izq,Bajo);
            LineTo(x,y);
            LineTo(Der,Bajo);
            Linea(3.2,0);
            Linea(0,5);
            Linea(-5,0);
            Linea(0,-5);
            Mueve(0,1);
            Linea(-1,1);
            Linea(0,5);
            Linea(2,2);
            Linea(0,30);
            Linea(10,0);
            Linea(0,-35);
            Linea(-6,-3);
        End;
    6:Begin                                     {Broca}
    Radio:=1.1*DiametroBroca/(2*100);
            If Unidades = 2 then Radio:=Radio*2.54;
            If Radio > 0 Then
                Begin
                    Linea(Radio/2,-Radio);
                    Linea(50*Unidades,0);
                    Linea(0,Radio*2);
                    Linea(-50*Unidades,0);
                    Linea(-Radio/2,-Radio)
                End
            End;
    7:Begin                                     {Cortadora interna}
            Linea(0,-8);
            Linea(7,-4);
            Linea(0,8);
            Linea(-7,4);
            Mueve(1,-8-4/7);
            Linea(0,-3-3/7);
            Linea(59,0);
            Linea(0,10);
            Linea(-56.5,0);
        End;
    8:Begin                                     {Rosca Interna}
            SetColor(0);
            Bajo:=y-Round(1.5*BaseHerramientas*RelY/10);
            Izq:=x-Round(0.867*BaseHerramientas*RelX/10);
            Der:=x+Round(0.867*BaseHerramientas*RelX/10);
            For i:=y Downto Bajo do
                Begin
                    Line(Izq,Bajo,x,i);
                    Line(x,i,Der,Bajo)
                End;
            SetColor(HerrColor);
            MoveTo(Izq,Bajo);
            LineTo(x,y);
            LineTo(Der,Bajo);
            Linea(3.2,0);
            Linea(0,-5);
            Linea(-5,0);
            Linea(0,5);
            Mueve(5,-1);
            Linea(55,0);
            Linea(0,-10);
            Linea(-60,0);
            Linea(0,6);
        End;
    End; {Case Tool}
    MoveTo(x,y);
    SetColor(OriginColor);
End;

FUNCTION Limits(Comando:String;Parametro:Char;Valor:LongInt):Boolean;
    Var
```

```
    Maximo,Minimo:Integer;
Begin                                          {Definición de límites segun COMPAT 5 CNC}
    Limits:=False;
    Parametro:=UpCase(Parametro);
    If Not (Parametro in ['X','Z','F','H','I','K','T','L','D']) Then Exit;
    Case Parametro of
        'X':Begin
                If Unidades = 1
                    Then Maximo:=9999
                    Else Maximo:=3999;
                Minimo:=-Maximo
            End;
        'Z':Begin
                If Unidades = 1
                    Then Maximo:=32760
                    Else Maximo:=12800;
                Minimo:=-Maximo
            End;
        'F':Begin
                If Unidades = 1
                    Then Maximo:=499
                    Else Maximo:=199;
                Minimo:=2
            End;
        'I':Begin
                If Unidades = 1
                    Then Maximo:=5999
                    Else Maximo:=1999;
                Minimo:=0
            End;
        'K':If (Comando = 'G33') or (Comando = 'G78')
                Then
                    Begin
                        If Unidades = 1
                            Then Maximo:=499
                            Else Maximo:=199;
                        Minimo:=2
                    End
                Else
                    Begin
                        If Unidades = 1
                            Then Maximo:=5999
                            Else Maximo:=1999;
                        Minimo:=0
                    End;
        'L':Begin
                Maximo:=MaxLineas;
                Minimo:=0
            End;
        'T':Begin
                    If Unidades = 1
                        Then Maximo:=499
                        Else Maximo:=199;
                    Minimo:=0
                End;
        'H':Begin
                    If Comando = 'G86'
                        Then Minimo:=10
                        Else Minimo:=0;
                    Maximo:=999
                End;
        'D':Begin
            Minimo:=0;
                Maximo:=999
            End;
    End;
    Limits:=(Valor<=Maximo) and (Valor>=Minimo)
End;

END.
```

EDITFUNC.PAS

```
Procedure Abajo;
    Begin
        Loc:=Loc+1;
        If Loc>MaxLineas Then Loc:=MaxLineas;
        If Loc>N+MaxRen-1 Then
            Begin
                N:=N+1;
                If N>MaxLineas Then N:=MaxLineas
            End
    End; {Abajo}

Procedure Arriba;
    Begin
        Loc:=Loc-1;
        If Loc<0 Then Loc:=0;
        If Loc<N Then
            Begin
                N:=N-1;
                If N<0 Then N:=0
            End;
    End; {arriba}

Procedure PaginaArriba;
    Begin
        Loc:=Loc-MaxRen;
        If Loc<0 Then Loc:=0;
        N:=N-MaxRen;
        If N<0 Then N:=0
    End; {PgUp}

Procedure PaginaAbajo;
    Begin
        Loc:=Loc+MaxRen;
        If Loc>MaxLineas Then Loc:=MaxLineas;
        N:=N+MaxRen;
        If N+MaxRen > MaxLineas Then N:=MaxLineas-MaxRen+1
    End; {PgDn}

Procedure PrincipioPagina;
    Begin
        Loc:=N;
        Campo:=1;
        Renglon(PosAnt,0)
    End;

Procedure FinPagina;
    Begin
        Loc:=N+MaxRen-1;
        Campo:=1;
        Renglon(PosAnt,0)
    End;

Procedure PrincipioPrograma;
    Begin
        Loc:=0;
        N:=0;
        Campo:=1
    End;

Procedure FinPrograma;
    Begin
        i:=MaxLineas;
        While Linea[i].Command<>'M30' do i:=i-1;
        Loc:=i;
        N:=((Loc Div MaxRen)+1)*MaxRen-MaxRen;
        Campo:=1
    End;

Procedure BuscaLinea;
```

```
    Begin
        ExpandWindow;
        I:=MaxLineas;
        While Linea[I].Command<>'M30' do I:=I-1;
        Mensaje('Número de la linea a buscar :',0);
        GotoXY(56,25);
        ReadInt(Loc,0,I);
        Mensaje('',0);
        N:=((Loc Div MaxRen)+1)*MaxRen-MaxRen;
        Campo:=1;
        Presenta
    End;

Procedure BorraLinea;
    Begin
        If (Linea[loc].Command = 'M30') or (Linea[loc].Command = 'G71') or (Linea[loc].Command = 'G70')
            Then
                Begin
                    Beep;
                    Mensaje('No se puede borrar esta linea...',1);
                End
            Else
                Begin
                    Mensaje('¿Estás seguro? (S/N)',0);
                    Repeat Car:=UpCase(ReadKey) Until Car in ['N','S'];
                    Mensaje('',0);
                    If Car = 'S' Then
                        For I:=Loc to MaxLineas-1 do
                            Linea[I]:=Linea[I+1];
                    For I:=2 to 6 do Asigna(MaxLineas,I,'');
                    Presenta
                End
    End;

Procedure GrabaPrograma;
    Begin
        Car:='R';
        While Car = 'R' do
            If not Save
                Then
                    Begin
                        Mensaje('El programa no ha sido grabado. [R]eintentar, [C]ancelar',0);
                        Repeat Car:=UpCase(ReadKey) Until Car in ['R','C'];
                        Mensaje('',0);
                    End
                Else
                    Car:='C'
    End;

Procedure Derecha;
    Begin
        Repeat
            Campo:=Campo+1;
            If Campo>6 Then
                Begin
                    Campo:=1;
                    Loc:=Loc+1;
                    If Loc>MaxLineas Then Loc:=MaxLineas;
                    If Loc>N+MaxRen-1 Then
                        Begin
                            N:=N+1;
                            If N>MaxLineas Then N:=MaxLineas
                        End
                End
        Until Not Vacio;
    End; {derecha}

Procedure Izquierda;
    Begin
        Repeat
            Campo:=Campo-1;
            If Campo<1 Then
```

```
            Begin
                Campo:=6;
                Loc:=Loc-1;
                If Loc<0 Then Loc:=0;
                If Loc<N Then
                    Begin
                        N:=N-1;
                        If N<0 Then N:=0
                    End
            End
        Until Not Vacio;
    End; {izquierda}


Procedure Imprime;
    Var
        Pl:Integer;
        CancelPrint:Boolean;
        y,m,d,w:Word;
    Procedure PrintLn(Texto:String);
        Begin
            Mensaje('',0);
            Car:='R';
            While (Car = 'R') and (Not CancelPrint) do
            If PrinterOnLine
                    Then
                        Begin
                            Mensaje('Imprimiendo linea...',0);
                            Writeln(LST,Texto);
                            Exit
                        End
                    Else
                        Begin
                            Mensaje('La impresora no está lista...[R]eintentar, [I]gnorar, [C]ancelar',0);
                            Repeat Car:=UpCase(ReadKey) Until Car in ['R','I','C'];
                            Mensaje('',0);
                            CancelPrint:= Car = 'C'
                        End
        End;
    Procedure Print(Texto:String);
        Begin
            Mensaje('',0);
            Car:='R';
            While (Car = 'R') and (Not CancelPrint) do
            If PrinterOnLine
            Then
                Begin
                    Mensaje('Imprimiendo en linea...',0);
                    Write(LST,Texto);
                    Exit
                End
            Else
                Begin
                    Mensaje('La impresora no está lista...[R]eintentar, [I]gnorar, [C]ancelar',0);
                    Repeat Car:=UpCase(ReadKey) Until Car in ['R','I','C'];
                    Mensaje('',0);
                    CancelPrint:= Car = 'C'
                End
        End;
    Begin
        If PrinterOnLine Then Rewrite(LST) Else Exit;
        CancelPrint:=False;
        Print(#10); {Salto de pagina}
        PrintLn('********************** '+UpString(Programa)+'.CNC **********************');
        PrintLn('');
        GetDate(y,m,d,w);
        PrintLn('       FECHA: '+StrNum(d,0,0)+'/'+StrNum(m,0,0)+'/'+StrNum(y,0,0));
        PrintLn('');
        PrintLn('       NOMBRE DEL PROGRAMADOR: '+Programador);
        PrintLn('       DESCRIPCION DEL PROGRAMA: '+Descripcion);
        If Unidades = 2
            Then
```

```
            PrintLn('       UNIDADES : in')
        Else
            PrintLn('       UNIDADES : mm');
    PrintLn('       LARGO          : '+StrNum(Dimensiones.Largo,0,10));
    PrintLn('       DIAMETRO : '+StrNum(Dimensiones.Diametro,0,10));
    Print('     HERRAMIENTAS:');
    For i:=1 to MaxHerram do Print(StrNum(OrdHerram[i],3,0));PrintLn('');
    Pl:=-1;
    Repeat
        Pl:=Pl+1;
        With Linea[Pl] do
            Begin
                Print('     '+Right(StrNum(Pl+10000,0,0),3));
                Print(' '+Command);
                St:='';For i:=1 to 5-Length(Command) do St:=St+' ';
                Print(St+CoordX);
                St:='';For i:=1 to 10-Length(CoordX) do St:=St+' ';
                Print(St+CoordZ);
                St:='';For i:=1 to 10-Length(CoordZ) do St:=St+' ';
                Print(St+Velocidad);
                St:='';For i:=1 to 10-Length(Velocidad) do St:=St+' ';
                PrintLn(St+ProfCorte)
            End; {With}
    Until Linea[Pl].Command = 'M30';
    Mensaje('',0);
    If PrinterOnLine Then Close(LST)
End; {Imprime}

Procedure AyudaComando(Cmnd:String);
    Const
        LeftChar = 3;
    Var
        j,k:Integer;
        Hach:Text;
    Begin
        If Cmnd = '' Then Cmnd:=Variable(Loc,2);
        Case Cmnd[1] of
            'G':k:=0;
            'M':k:=100
        End;
        Val(Copy(Cmnd,2,2),j,i);
        If i <> 0 Then
            Begin
                Mensaje('Error en comando',1);
                InitEditor;
                Presenta;
                Exit
            End;
        If Comando[j+k,2] = ' ' Then
            Begin
                Mensaje('Comando no reconocido',1);
                InitEditor;
                Presenta;
                Exit
            End;
        Assign(Hach,UtilDir+'CODIGO.HLP');
        {$I-}
        Reset(Hach);
        i:=IOResult;
        {$I+}
        If i<>0 Then
            Begin
                Mensaje(GetIOError(i)+', '+UtilDir+'CODIGO.HLP',1);
                InitEditor;
                Presenta;
                Exit
            End;
        St:='';
        Flag:=False;
        While not (Eof(Hach) or Flag) do
            Begin
                ReadLn(Hach,St);
```

```
                If St[1] = '&' Then Flag:= Copy(St,2,3) = Cmnd
            End;
        If not Flag Then
            Begin
                Mensaje('No hay ayuda disponible para el comando '+Cmnd,1);
                InitEditor;
                Presenta;
                Exit
            End;
        Flag:=False;
        ColorText(Drv[18],Drv[8]);
        ClearWindow(RgtLn+2,3,79,23);
        GotoXY(10,1);Write('AYUDA A COMANDO');
        ColorText(Drv[19],Drv[8]);
        While not (Eof(Hach) or Flag) do
            Begin
                ReadLn(Hach,St);
                Flag:= St[1] = '&';
                If Not Flag Then
                    If St[1] = '#'
                        Then
                            Begin
                                GotoXY(LeftChar,WhereY+2);
                                Writeln('Sintaxis:');
                                ColorText(Drv[18],Drv[8]);
                                Delete(St,1,1);
                                St:=Cmnd+'      '+St;
                                GotoXY(LeftChar,WhereY+1);
                                Writeln(St);
                                ColorText(Drv[19],Drv[8]);
                            End
                        Else
                            Begin
                                GotoXY(LeftChar,WhereY+1);
                                Write(St)
                            End
            End;
        Close(Hach);
        Mensaje('Oprima cualquier tecla...',1);
        InitEditor;
        Presenta
    End;

Procedure ListaComandos;
    Const
        Top=3;
        Columns:Array [1..5] of Byte = (5,10,15,20,25);
    Var
        Ant,MaxR,indice,j,k:Integer;
        Malla:Array [0..99] of String[3];
        Mlx,Mly:Array [0..99] of Integer;
    Procedure Normal(ind:Integer);
        Begin
            GotoXY(Mlx[ind],Mly[ind]);
            Write(Malla[ind])
        End;
    Procedure Inverse(ind:Integer);
        Begin
            ColorText(Drv[16],Drv[17]);
            GotoXY(Mlx[ind],Mly[ind]);
            Write(Malla[ind]);
            ColorText(Drv[15],Drv[8])
        End;
    Begin
        For i:=0 to 99 do Malla[i]:='';
        ColorText(Drv[19],Drv[8]);
        ClearWindow(RgtLn+2,3,79,23);
        ColorText(Drv[18],Drv[8]);
        GotoXY(6,1):Write('COMANDOS RECONOCIDOS');
        GotoXY(1,Top);
        MaxR:=TotalCommands div 5 + 1;
        GotoXY(1,MaxR+Top+3);
```

```
        ColorText(Drv[19],Drv[8]);
        Writeln('       Utiliza las flechas para');
        Writeln('       localizar comando.');
        Writeln('       [Enter] Para seleccionar');
        Writeln('       [Esc] Para cancelar');
        ColorText(Drv[15],Drv[8]);
        j:=1;
        k:=0;
        For i:=0 to 199 do
            If Comando[i,2] in ['G','M'] Then
                Begin
                    k:=k+1;
                    If k > MaxR Then
                        Begin
                            k:=1;
                            j:=j+1
                        End;
                    Indice:=MaxR*(j-1)+k-1;
                    Mlx[Indice]:=Columna[j];
                    Mly[Indice]:=Top+k;
                    Malla[Indice]:=Comando[i,2]+Copy(StrNum(i+100,0,0),2,2);
                    Normal(Indice);
                End;
        j:=1;
        k:=1;
        Ant:=1;
        Repeat
            i:=MaxR*(j-1)+k-1;
            Normal(Ant);
            Inverse(i);
            Ant:=i;
            Car:=UpCase(ReadKey);
            If Car = #0 Then Car:=UpCase(ReadKey);
            Case Car of
                #80:Begin
                            k:=k+1;
                            If k>MaxR Then k:=1
                        End;
                #72:Begin
                            k:=k-1;
                            If k<1 Then k:=MaxR
                        End;
                #77:Begin
                            j:=j+1;
                            If j>5 Then j:=1
                        End;
                #75:Begin
                            j:=j-1;
                            If j<1 Then j:=5
                        End;
            End
        Until Car in [#27,#13];
        If Car = #13
            Then
                AyudaComando(Malla[i])
            Else
                Begin
                    InitEditor;
                    Presenta;
                End
    End;

Procedure BuscaComando;
    Const
        Extra=48;
    Var
        Valor:String;
    Begin
        ExpandWindow;
        Mensaje('Comando a buscar: ',0);
        ColorText(Drv[12],Drv[13]);
        Valor:=LastFindCommand;
```

```
Repeat
    GotoXY(Extra,25);
    Write(Valor);
    Car:=ReadKey;
    If Car <> #13 Then
        Begin
            Valor:='';
            Case Car Of
                '0'..'9':Valor:='G'+Car;
                         '+':Valor:='H';
                         #27:Begin
                                                    Mensaje('',0);
                                                    Presenta;
                                                    Exit
                                                End
            Else Beep;
        End
Until Valor[1] <> ' ';
If Car <> #13 Then
    Begin
        GotoXY(Extra,25);
        Write(' ');
        GotoXY(Extra,25);
        Write(Valor);
        Repeat
            Car:=ReadKey;
            If Not (Car in ['0'..'9',#27]) Then Beep;
            If Car = #27 Then
                Begin
                    Mensaje('',0);
                    Presenta;
                    Exit
                End
        Until Car in ['0'..'9'];
        Valor:=Valor+Car;
        GotoXY(Extra,25);
        Write(Valor);
        If Length(Valor)<3 Then
            Begin
                Repeat
                    Car:=ReadKey;
                    If Not (Car in ['0'..'9',#27]) Then Beep;
                    If Car = #27 Then
                        Begin
                            Mensaje('',0);
                            Presenta;
                            Exit
                        End
                Until Car in ['0'..'9'];
                Valor:=valor+Car
            End
    End;
GotoXY(Extra,25);
Write(Valor);
LastFindCommand:=Valor;
If Not Verifica(Loc,2,Valor)
    Then
        Begin
            LastFindCommand:='  ';
            Beep;
            Error:=2;
        End
    Else
        Begin
            Flag:=False;
            i:=Loc;
            Repeat
                i:=i+1;
                St:=Variable(i,2);
                Flag:=Valor = St;
            Until Flag or (i >= MaxLineas);
```

```
                If Flag
                    Then
                        Begin
                            Loc:=1;
                            N:=((Loc Div MaxRen)+1)*MaxRen-MaxRen;
                            Campo:=2;
                            Presenta;
                        End
                    Else
                        Mensaje('Comando '+Valor+' no localizado',1)
            End;
    Mensaje('',0);
    Presenta;
End;
```

EDITOR.PAS

```
UNIT EDITOR;

INTERFACE

Uses CRT,Graph,Printer,DOS,Utileria,Definit,UtilCNC,UtilCRT,Gestion;

FUNCTION Verifica(Renglon,Columna:Integer;Value:String):Boolean;
PROCEDURE EditProgram;

IMPLEMENTATION


FUNCTION Verifica(Renglon,Columna:Integer;Value:String):Boolean;
    Var
        Valor:LongInt;
        i,j,k:Integer;
    Begin
        Verifica:=True;
        Case Columna of
            2:Begin
                    Case Value[1] of
                        'G':k:=0;
                        'M':k:=100
                    End;
                    Val(Copy(Value,2,2),j,i);
                    If i <> 0 Then
                        Begin
                            Verifica:=false;
                            Mensaje('Error en comando',1);
                            Exit
                        End;
                    If Comando[j+k,2] = ' ' Then
                        Begin
                            Verifica:=False;
                            Mensaje('Comando '+Value+' no reconocido',1);
                            Exit
                        End;
                    If Value = 'M99' Then
                        Begin
                            If Renglon = 0 Then
                                Begin
                                        Mensaje('Falta Comando G02 o G03',1);
                                        Verifica:=False;
                                        Exit
                                End;
                            If (Variable[Renglon-1,2) <> 'G02') and (Variable[Renglon-1,2) <> 'G03') Then
                                Begin
                                        Mensaje('Falta Comando G02 o G03',1);
                                        Verifica:=False;
                                        Exit
                                End
                        End;
                    If (Value = 'G70') Or (Value = 'G71') Then
                        Begin
                            Verifica:= Renglon < 1;
                            Mensaje('No se pueden redefinir las unidades',1)
                        End;
                    If Value = 'M30' Then
                        Begin
                            Verifica:=False;
                            Mensaje('Fin de programa incorrecto',1)
                        End
                End {2}
            Else {Case}
                Begin
                    Valor:=0;
                    St:=Variable[Renglon,2];
                    If St[1] = 'G'
                        Then k:=0
```

```
                    Else k:=100;
                Val(Copy(Variable(Renglon,2),2,2),j,i);
                If i <> 0 Then
                    Begin
                        Verifica:=false;
                        Mensaje('Error en comando',1);
                        Exit
                    End;
                If Comando[j+k,Columna] = '*'
                    Then
                        Case Columna of
                            3:Car:='X';
                            4:Car:='Z';
                            5:Car:='F';
                            6:Car:='H'
                        End
                    Else
                        Begin
                            Car:=Comando[j+k,Columna];
                            Delete(Value,1,1);
                        End;
                Val(Value,Valor,i);
                If i <> 0 Then
                    Begin
                        Mensaje('Error numérico',1);
                        Verifica:=False;
                        Exit
                    End;
                If Not Limits(Variable(Renglon,2),Car,Valor) Then
                    Begin
                        Verifica:=False;
                        Mensaje('Valor fuera de rango',1)
                    End
            End {Else Case}
        End {Case}
    End; {Verifica}

PROCEDURE EditProgram;
    Const
        OffSetRow=5;
        MaxRen=24-OffSetRow;
        Col:Array [1..6] of Byte = (2,7,12,22,32,39);
        LongC:Array [1..6] of Byte = (3,3,5,6,4,4);
        RgtLn=44; {Linea derecha de editor activo}
    Var
        Ch:Char;
        OpEdit,Campo,Loc,NAnt,PosAnt:Integer;
        Error:Byte;
        LastFindCommand:String;
        MuestraFunciones:Boolean;

    {$I EDITPROC.PAS}

    {$I EDITFUNC.PAS}

    Begin {Editor}
        Menu[1]:='Programa en Memoria';
        Menu[2]:='Programa Nuevo';
        Menu[3]:='Menu Principal';
        Menu[4]:=FDM;
        OpEdit:=Entrada(11,5,Menu,Sencillo,Izq,1);
        Case OpEdit of
            1:Begin
                    If Linea[0].Command = 'H30' Then
                        Begin
                            Beep;
                            Mensaje('No hay programa en memoria',1);
                            Exit;
                        End;
                End; {1}
            2:Begin
                    ProgramaNuevo;
```

```
            If Programa = '' Then
            Begin
                  InitProgramCNC;
                  Exit
            End;
      End;
   3:Exit
End; {Case}
NuestraFunciones:=False;
InitEditor;
N:=0;
Loc:=0;
Campo:=1;
NAnt:=1;
PosAnt:=0;
LastFindCommand:='   ';
Repeat
   If N<>Nant Then
      Begin
         Presenta;
         HAnt:=N
      End;
   Renglon(PosAnt,0);
   Renglon(Loc,1);
   PosAnt:=Loc;
   Repeat Ch:=ReadKey Until Ch in [#0,#13,#27];
   If Ch = #0 Then Ch:=ReadKey;
   Case Ch of
      #80:Abajo;
      #72:Arriba;
      #73:PaginaArriba;
      #81:PaginaAbajo;
      #77:Derecha;
      #75:Izquierda;
      #71:PrincipioPagina;
      #79:FinPagina;
      #132:PrincipioPrograma;
      #118:FinPrograma;
      #13:Begin
            Case Campo of {Enter}
               1:If (Variable(Loc,2) = 'G70') or (Variable(Loc,2) = 'G71')
                        Then
                              Beep
                        Else
                              InsertaLinea;
               2:Beep;
               Else EditaCampo
            End; {Case}
            If Loc>MaxLineas Then Loc:=MaxLineas;
            If Loc>N+MaxRen-1 Then
               Begin
                     N:=N+1;
                     If N>MaxLineas Then
                        Begin
                              Beep;
                              N:=MaxLineas
                        End;
               End
         End;
      #59:Begin                                          {F1}
            NuestraFunciones:= (NuestraFunciones = False);
            InitEditor;
            Presenta
         End;
      #60:GrabaPrograma;                                 {F2}
      #61:Imprime;                                       {F3}
      #62:BuscaLinea;                                    {F4}
      #63:BorraLinea;                                    {F5}
      #64:ListaComandos;                                 {F6}
      #65:BuscaComando;                                  {F7}
      #66:AyudaComando('');                              {F8}
      #67:Begin                                          {F9}
```

```
                        ColorText(Drv[14],Drv[0]);
                        ClearWindow(RgtLn+2,3,79,23);
                        EntradaDatos;
                        InitEditor;
                        Presenta
                    End;
            #68:Begin                                                    {F10}
                        InitGraphics;
                        DefineHerramientas;
                        InitText;
                        Pantalla;
                        InitEditor;
                        Presenta
                    End
        End; {case general}
        While Vacio do Campo:=Campo-1
    Until Ch = #27 {Esc}
End; {Editor}

END.
```

EDITPROC.PAS

```
Procedure EntradaDatos;
    Begin
        ColorText(Drv[14],Drv[8]);
        If Unidades = 1
            Then
                St:='mm'
            Else
                St:='in';
        GotoXY(3,1);Write('DATOS DEL PROGRAMA');
        GotoXY(3,3);Write('Programador: ');
        GotoXY(3,6);Write('Descripción: ');
        GotoXY(3,10);Write('DIMENSIONES DE LA PIEZA');
        GotoXY(3,12);Write('Diámetro:          ',St);
        GotoXY(3,14);Write('Longitud:          ',St);
        ColorText(Drv[12],Drv[13]);
        GotoXY(3,4);Write(Programador,'':25-Length(Programador));
        GotoXY(3,7);Write(Descripcion,'':25-Length(Descripcion));
        GotoXY(13,12);Write(Dimensiones.Diametro:7:3);
        GotoXY(13,14);Write(Dimensiones.Largo:7:3);
        Programador:=LeeCampo(3,4,Programador,25);
        Descripcion:=LeeCampo(3,7,Descripcion,25);
        With Dimensiones do
            Begin
                Repeat
                    GotoXY(13,12);ReadReal(Diametro,7);
                    Flag:=(Diametro > 0);
                    If Not Flag Then Mensaje('El diámetro debe ser mayor que cero',1);
                Until Flag;
                Repeat
                    GotoXY(13,14);ReadReal(Largo,7);
                    Flag:=(Largo > 0);
                    If Not Flag Then Mensaje('La longitud debe ser mayor que cero',1);
                Until Flag;
            End; {Dimensiones}
        ColorText(15,Drv[1])
    End;

Procedure ProgramaNuevo;
    Begin
        If Linea[0].Command = 'M30'
            Then
                Ch:='S'
            Else
                Begin
                    Mensaje('Asegúrate de que el programa anterior esté grabado. ¿Continuar? (S/N)',0);
                    Repeat Ch:=UpCase(ReadKey) Until Ch in ['S','N'];
                    Mensaje('',0)
                End;
        If Ch = 'N'
            Then
                Exit
            Else
                Begin
                    Menu[1]:='Milímetros';
                    Menu[2]:='Pulgadas';
                    Menu[3]:=FDN;
                    Unidades:=Entrada(14,12,Menu,Sencillo,Izq,1);
                    InitProgramCNC;
                    If Unidades = 1
                        Then
                            Linea[0].Command:='G71'
                        Else
                            Linea[0].Command:='G70';
                    Linea[1].Command:='M30';
                    Programa:='';
                    Cuadro(41,5,71,21,2);
                    EntradaDatos;
                    Pantalla;
                    SaveProgram;
```

```
                End
    End;

Procedure Renglon(Ubica,Estado:Integer);
    Begin
        ColorText(Drv[15],Drv[8]);
        If not ((Ubica < N) or (Ubica > N+MaxRen-1)) Then
            Begin
                For i:=1 to 6 do
                    Begin
                        GotoXY(Col[i],Ubica-N+1);
                        Write(Variable(Ubica,i))
                    End;
                If Estado = 1
                    Then
                        Begin
                            ColorText(Drv[16],Drv[17]);
                            GotoXY(Col[Campo],Ubica-N+1);
                            Write(Variable(Ubica,Campo));
                            NormVideo
                        End
            End;
    End;

Procedure Presenta;
    Var
        Row:Integer;
    Begin
        ColorText(Drv[15],Drv[8]);
        ClearWindow(2,OffSetRow,RgtLn,23);
        Row:=N-1;
        If Row<-1 Then Row:=-1;
        While (Row<N+MaxRen-1) and (Row<MaxLineas) do
            Begin
                Row:=Row+1;
                Renglon(Row,0)
            End;
    End;

Function Vacio:Boolean;
    Begin
        Vacio:= (Length(Variable(Loc,Campo)) = 0)
    End;

Procedure EditaCampo;
    Var
        Prefijo,Valor,ValorInic,ValorSuperior:String;
        j,k,OffSet,Comm:Byte;
        V:Integer;
    Begin
        If Campo = 1
            Then
                Begin
                    Beep;
                    Exit
                End;
        If Campo = 2
            Then
                Begin
                    ColorText(Drv[12],Drv[13]);
                    Valor:='    ';
                    GotoXY(Col[2],Loc-N+1);
                    Write(Valor);
                    Repeat
                        Car:=UpCase(ReadKey);
                        Case Car Of
                            '0'..'9':Valor:=Valor+Car;
                            '+','N':Valor:='N';
                            #13,#27:Begin
                                        Error:=2;
                                        Exit
                                    End;
```

```
                Else Beep
            End
        Until Valor[1] <> ' ';
        GotoXY(Col[2],Loc-N+1);
        Write(Valor);
        Repeat
            Car:=ReadKey;
            If Car = #27 Then
                Begin
                    Error:=2;
                    Exit
                End;
            If Not (Car in ['0'..'9']) Then Beep;
        Until Car in ['0'..'9'];
        Valor:=Valor+Car;
        GotoXY(Col[2],Loc-N+1);
        Write(Valor);
        If Length(Valor)<3 Then
            Begin
                Repeat
                    Car:=ReadKey;
                    If Car = #27 Then
                        Begin
                            Error:=2;
                            Exit
                        End;
                    If Not (Car in ['0'..'9']) Then Beep;
                Until Car in ['0'..'9'];
                Valor:=valor+Car
            End;
        GotoXY(Col[2],Loc-N+1);
        Write(Valor);
        If Not Verifica(Loc,2,Valor)
            Then Error:=2
            Else Asigna(Loc,2,Valor);
        NormVideo;
    End
Else
    Begin
        If Campo > 6 Then
            Begin
                Error:=1;
                Exit
            End;
        Valor:=Variable(Loc,2);
        If Valor[1] = 'M'
            Then j:=100
            Else j:=0;
        Val(Copy(Valor,2,2),k,i);
        Comm:=j+k;
        Case Comando[Comm,Campo] of
            '*':OffSet:=0;
            ' ' :Exit;
            Else OffSet:=1
        End;
        Prefijo:=Comando[Comm,Campo];
        ColorText(Drv[15],Drv[8]);
        GotoXY(Col[Campo],Loc-N+1);Write(Prefijo);
        Valor:=Variable(Loc,Campo);
        ValorInic:=Copy(Valor,1+OffSet,Length(Valor)-OffSet);
        Repeat
            Valor:=LeeCampo(Col[Campo]+OffSet,Loc-N+1,ValorInic,LongC[Campo]-OffSet);
            If Valor = '' Then
                Begin
                    i:=Loc;
                    ValorSuperior:='';
                    While (i>1) and (ValorSuperior = '') do
                        Begin
                            i:=i-1;
                            ValorSuperior:=Variable(i,Campo);
                        End;
                    Valor:=Copy(ValorSuperior,1+OffSet,Length(ValorSuperior)-OffSet);
```

```
                        End;
                    Val(Valor,V,i);
                    If i<>0 Then Beep;
                    ColorText[Drv[15],Drv[8]);
                    GotoXY(Col[Campo]+Offset,Loc-N+1);Write(''':LongC[Campo]-OffSet);
                Until i = 0;
                If OffSet = 1 Then Valor:=Prefijo+Valor;
                If Verifica(Loc,Campo,Valor)
                    Then Asigna(Loc,Campo,Valor)
                    Else Error:=2;
            End;
    End;

Procedure InsertaLinea;
    Begin
        i:=MaxLineas;
        While Linea[i].Command<>'M30' do i:=i-1;
        if i = MaxLineas Then
            Begin
                Mensaje('No hay espacio en memoria',1);
                Exit
            End;
        If Loc > i Then
            Begin
                Beep;
                Exit
            End;
        For i:=MaxLineas DownTo Loc+1 Do
            Linea[i]:=Linea[i-1];
        For i:=2 to 6 do Asigna(Loc,i,'');
        Presenta;
        Error:=0;
        Campo:=1;
        Repeat
            Campo:=Campo+1;
            EditaCampo;
            Renglon(Loc,0)
        Until Error<>0;
        Case Error of
            1:Begin                                     {Final de entrada normal}
                    Loc:=Loc+1;
                    Campo:=1;
                End;
            2:Begin                                     {Error en linea, borrar}
                    For i:=Loc to MaxLineas-1 do Linea[i]:=Linea[i+1];
                    For i:=2 to 6 do Asigna(MaxLineas,i,'');
                    Campo:=1;
                End
        End; {Case}
        Presenta;
    End;

Procedure DefineHerramientas;
    Var
        Anterior,Opcion:Byte;
        Car:Char;
        OldHerram:TipoOrdenHerram;
        EscTools:Real;
    Procedure Box(Item,Color:Byte);
        Const
            OffX=5;
            OffY=3;
        Var
            Columna,Renglon,OriginColor:Word;
            Altura,Ancho:Byte;
            Hx,Hy:Word;
        Begin
            SetTextStyle(DefaultFont,HorizDir,1);
            Hx:=TextWidth('H');
            Hy:=TextHeight('H');
            Altura:=TextHeight(Herramientas[Item]) + OffY * 2 - 1;
            Ancho:=TextWidth(Herramientas[Item]) + OffX * 2 - 1;
```

```
            OriginColor:=GetColor;
            SetColor(Color);
            If Item<=5
                Then
                    Renglon:=Drv[78]
                Else
                    Renglon:=Drv[79];
            Renglon:=Hy*(Renglon - 1) - OffY;
            Columna:=Hx*(Drv[79+Item] - 1) - OffX;
            Rectangle(Columna,Renglon,Columna + Ancho,Renglon + Altura);
            SetColor(OriginColor);
        End;
    Begin {DefineHerramientas}
        EscTools:=Drv[89]/10;
        If Unidades = 2 Then EscTools:=EscTools*2.54;
        ClearDevice;
        SetColor(Drv[7]);
        DrawText(29,1,'DEFINICION DE HERRAMIENTAS',1);
        SetColor(Drv[75]);
        Rectangle(0,Drv[76],GetMaxX,GetMaxY);
        SetColor(Drv[52]);
        DiametroBroca:=1000/Unidades;
        Herramienta[1,Drv[57],Drv[58],1,EscTools];DrawText(Drv[80],Drv[78],Herramientas[1],1);
        Herramienta[2,Drv[59],Drv[60],1,EscTools];DrawText(Drv[81],Drv[78],Herramientas[2],1);
        Herramienta[3,Drv[61],Drv[62],1,EscTools];DrawText(Drv[82],Drv[78],Herramientas[3],1);
        Herramienta[4,Drv[63],Drv[64],1,EscTools];DrawText(Drv[83],Drv[78],Herramientas[4],1);
        Herramienta[5,Drv[65],Drv[66],1,EscTools];DrawText(Drv[84],Drv[78],Herramientas[5],1);
        Herramienta[6,Drv[67],Drv[68],1,EscTools];DrawText(Drv[85],Drv[79],Herramientas[6],1);
        Herramienta[7,Drv[69],Drv[70],1,EscTools];DrawText(Drv[86],Drv[79],Herramientas[7],1);
        Herramienta[8,Drv[71],Drv[72],1,EscTools];DrawText(Drv[87],Drv[79],Herramientas[8],1);
        DiametroBroca:=0;
        OldHerram:=OrdHerram;
        For i:=1 to MaxHerram do
            Begin
                SetViewPort(1,Drv[56],GetMaxX-1,GetMaxY-1,ClipOn);
                ClearViewPort;
                SetViewPort(0,0,GetMaxX,GetMaxY,ClipOn);
                SetTextStyle(DefaultFont,HorizDir,2);
                SetColor(Drv[54]);
                OutTextXY(Drv[55],Drv[56],'HERRAMIENTA #'+StrNum(i,0,0));
                Opcion:=OldHerram[i];
                Box(Opcion,Drv[53]);
                Repeat
                    Anterior:=Opcion;
                    Repeat
                        Car:=UpCase(ReadKey);
                        If Car = #27 Then Exit;
                        If Car=#0 Then Car:=UpCase(ReadKey)
                    Until Car in [#13,'H','K'];
                    Case Car of
                        'K':{Izquierda}
                                Begin
                                    Opcion:=Opcion-1;
                                    If Opcion<1 Then Opcion:=MaxHerram
                                End;
                        'H':{Derecha}
                                Begin
                                    Opcion:=Opcion+1;
                                    If Opcion>MaxHerram Then Opcion:=1
                                End;
                    End; {Case}
                    Box(Anterior,0);
                    Box(Opcion,Drv[53]);
                Until Car = #13;
                OldHerram[i]:=Opcion;
                Box(Opcion,0);
            End; {For}
        OrdHerram:=OldHerram;
    End;

Procedure TablaFunciones;
    Begin
```

```
        GotoXY(RgtLn+3,WhereY+1);Write('[F2] Grabar programa');
        GotoXY(RgtLn+3,WhereY+1);Write('[F3] Imprime programa');
        GotoXY(RgtLn+3,WhereY+1);Write('[F4] Busca linea');
        GotoXY(RgtLn+3,WhereY+1);Write('[F5] Borra linea');
        GotoXY(RgtLn+3,WhereY+1);Write('[F6] Lista comandos');
        GotoXY(RgtLn+3,WhereY+1);Write('[F7] Busca comando');
        GotoXY(RgtLn+3,WhereY+1);Write('[F8] Ayuda a comando');
        GotoXY(RgtLn+3,WhereY+1);Write('[F9] Datos del programa');
        GotoXY(RgtLn+3,WhereY+1);Write('[F10] Herramientas');
        GotoXY(RgtLn+3,WhereY+2);Write('[Home] Principo pantalla');
        GotoXY(RgtLn+3,WhereY+1);Write('[End] Fin de pantalla');
        GotoXY(RgtLn+3,WhereY+1);Write('[PgUp] Pantalla anterior');
        GotoXY(RgtLn+3,WhereY+1);Write('[PgDn] Pantalla siguiente');
        GotoXY(RgtLn+3,WhereY+1);Write('[Ctrl-PgUp] Principio programa');
        GotoXY(RgtLn+3,WhereY+1);Write('[Enter] Inserta linea / edición');
        GotoXY(RgtLn+3,WhereY+1);Write('[Esc] Salir de editor');
    End;

Procedure TablaEstatus;
    Begin
        GotoXY(RgtLn+3,WhereY+2);Write('Programa: ',Drive,':\',Programa,'.CNC');
        If Unidades = 1
            Then
                St:=' mm'
            Else
                St:=' in';
        GotoXY(RgtLn+3,WhereY+2);Write('Diámetro = ',Dimensiones.Diametro:0:4,St);
        GotoXY(RgtLn+3,WhereY+1);Write('Longitud = ',Dimensiones.Largo:0:4,St);
        GotoXY(RgtLn+4,WhereY+2);Write('Definición de herramientas');
        For I:=1 to MaxHerram do
            Begin
                GotoXY(RgtLn+8,WhereY+1);Write(I);
                GotoXY(RgtLn+13,WhereY);Write(Herramientas[OrdHerram[I]])
            End;
    End;

Procedure InitEditor;
    Begin
        Cuadro(1,2,80,24,2);
        ExpandWindow;
        ColorText(Drv[6],Drv[7]);
        GotoXY(RgtLn+1,2);Write(#203);
        GotoXY(RgtLn+1,24);Write(#202);
        For I:=3 to 23 do
            Begin
                GotoXY(RgtLn+1,I);
                Write(#186)
            End;
        Window(2,3,79,23);
        ColorText(Drv[18],Drv[8]);
        GotoXY(1,1);Write('  N  G    X        Z        F        H');
        GotoXY(1,2);Write('    (M)  (I)      (K) (T)(L)(K)');
        If MuestraFunciones
            Then
                Begin
                    St:='[F1] Parámetros del programa';
                    GotoXY(52,1);Write('FUNCIONES DE EDICION')
                End
            Else
                Begin
                    St:='[F1] Funciones de edición';
                    GotoXY(51,1);Write('PARAMETROS DEL PROGRAMA');
                End;
        ColorText(Drv[19],Drv[8]);
        GotoXY(RgtLn+3,3);Write(St);
        If MuestraFunciones
            Then
                TablaFunciones
            Else
                TablaEstatus;
        ColorText(Drv[14],Drv[8]);
    End;
```

GESTION.PAS

```
UNIT GESTION;

INTERFACE

Uses DOS,CRT,Utileria,UtilCRT,UtilCNC;

FUNCTION Save:Boolean;
PROCEDURE SaveProgram;
PROCEDURE LoadProgram;
PROCEDURE Directorio;

IMPLEMENTATION

FUNCTION Save:Boolean;
    Var
        Reg:Integer;
    Begin
        Mensaje('Grabando programa...',0);
        {$I-}
        Rewrite(Lach);
        IOStat:=IOResult;
        {$I+}
        If IOStat <> 0 Then
            Begin
                Mensaje(GetIOError(IOStat),1);
                Save:=False;
                Exit;
            End;
        Writeln(Lach,'NOMBRE DEL PROGRAMADOR: ',Programador);
        Writeln(Lach,'DESCRIPCION DEL PROGRAMA: ',Descripcion);
        If Unidades = 2
            Then
                Writeln(Lach,'UNIDADES : in')
            Else
                Writeln(Lach,'UNIDADES : mm');
        Writeln(Lach,'LARGO     : ',Dimensiones.Largo:0:10);
        Writeln(Lach,'DIAMETRO  : ',Dimensiones.Diametro:0:10);
        Write(Lach,'HERRAMIENTAS:');
        For i:=1 to MaxHerram do Write(Lach,OrdHerram[i]:3);Writeln(Lach);
        Reg:=-1;
        Repeat
            Reg:=Reg+1;
            With Linea[Reg] do
                Begin
                    Write(Lach,Right(StrNum(Reg+10000,0,0),3):3);
                    Write(Lach,'':1,Command);
                    Write(Lach,'':5-Length(Command),CoordX);
                    Write(Lach,'':10-Length(CoordX),CoordZ);
                    Write(Lach,'':10-Length(CoordZ),Velocidad);
                    WriteLn(Lach,'':10-Length(Velocidad),ProfCorte)
                End; {With}
        Until Linea[Reg].Command = 'M30';
        Close(Lach);
        Mensaje('',0);
        Save:=True;
    End;

FUNCTION EnterName:Boolean;
    VAR
        Nombre:String30;
    Begin
        Cuadro(6,4,50,8,2);
        ColorText(Drv[14],Drv[8]);
        GotoXY(3,2);Writeln('Nombre del archivo : ');
        GotoXY(34,2);Write('.CNC');
        Nombre:=BorraEspacios(UpString(LeeCampo(24,2,Drive+':'+Programa,10)));
        If Copy(Nombre,2,1) = ':'
            Then
                Begin
```

```
                Drive:=Copy(Nombre,1,1);
                Nombre:=Copy(Nombre,3,8);
            End
        Else
            Nombre:=Copy(Nombre,1,8);
    For i:=1 to length(Nombre) do
        If Not(Nombre[i] in ['0'..'9','A'..'Z']) Then
            Begin
            Mensaje('Caracteres Inválidos en el nombre',1);
                EnterName:=False;
                Exit
            End;
    EnterName:=False;
    If Nombre<>'' Then
        Begin
            Programa:=Nombre;
            Assign(Lach,Concat(Drive,':\',Programa,'.CNC'));
            EnterName:=True
        End
End; {EnterName}

PROCEDURE SaveProgram;
    Var
        Salida:Boolean;
    BEGIN
        If Not EnterName Then Exit;
        {$I-}
        Reset(Lach);
        IOStat:=IOResult;
        {$I+}
        If IOStat = 0
            Then
                Begin
                    Beep;
                    Mensaje('El archivo ya existe, escribir sobre él (S/N)?',0);
                    Repeat Car:=UpCase(ReadKey) until Car in ['S','N'];
                    Mensaje('',0);
                    If Car <> 'S' Then Exit
                end
            Else
                If IOStat<>2 Then
                    Begin
                        Mensaje(GetIOError(IOStat),1);
                        Exit
                    End; {IOStat}
        While Not Save do
            Begin
                Mensaje('El programa no ha sido grabado. [R]eintentar, [C]ancelar',0);
                Repeat Car:=UpCase(ReadKey) Until Car in ['R','C'];
                Mensaje('',0);
                If Car = 'C' Then exit
            End;
        ColorText(15,Drv[1]);
    END; {SaveProgram}

PROCEDURE LoadProgram;
    Var
        OldProgram:String30;
        Lect:String80;
        Loc,Posit:Word;
        DirInfo:SearchRec;
        Archivo:String[9];
        Atrib:Word;
        Salida,Inicio:Boolean;
    Begin
        OldProgram:=Programa;
        If not EnterName then
            Begin
                Inicio:=True;
                Repeat
                    If Inicio Then
                        Begin
```

```
                FindFirst(Drive+':\*.CNC',Atrib,DirInfo);
                Inicio:=False
            End;
        Posit:=0;
        While (DosError = 0) and (Posit < 15) do
            Begin
                Posit:=Posit+1;
                Archivo:=DirInfo.Name;
                i:=Pos('.',Archivo);
                Menu[Posit]:=Copy(Archivo,1,i-1);
                FindNext(DirInfo)
            End;
        Inicio:= DosError <> 0;
        Menu[Posit+1]:='-Escape-';
        Menu[Posit+2]:='--Más--';
        Menu[Posit+3]:=FDM;
        If Menu[1] = '-Escape-' Then
            Begin
                Mensaje('No se encuentran archivos *.CNC',1);
                Exit
            End;
        Loc:=Entrada(60,4,Menu,Sencillo,Izq,Posit+2);
        ColorText(15,Drv[1]);
        ClearWindow(60,4,74,22)
    Until Loc<>Posit+2;
    If Loc = Posit+1 Then Exit;
    Programa:=Menu[Loc];
    Assign(Lach,Concat(Drive,':\',Programa,'.CNC'));
End;
{$I-}
Reset(Lach);
IOStat:=IOResult;
{$I+}
If IOStat <> 0 then
    Begin
    Programa:=OldProgram;
        Mensaje(GetIOError(IOStat),1);
        Exit
    End;
Mensaje('Cargando Programa...',0);
InitProgramCNC;
ReadLn(Lach,Lect);
Programador:=Copy(Lect,25,Length(Lect)-24);
ReadLn(Lach,Lect);
Descripcion:=Copy(Lect,27,Length(Lect)-26);
ReadLn(Lach,Lect);
If UpString(Copy(Lect,12,2)) = 'IN'
    Then
        Unidades:=2
    Else
        Unidades:=1;
ReadLn(Lach,Lect);
Val(Copy(Lect,12,20),Dimensiones.Largo,i);
If i <> 0 Then Dimensiones.Largo:=1000;
ReadLn(Lach,Lect);
Val(Copy(Lect,12,20),Dimensiones.Diametro,i);
If i <> 0 Then Dimensiones.Diametro:=22;
ReadLn(Lach,Lect);
For Posit:=1 to MaxHerram do
    Begin
        Val(Copy(Lect,13+Posit*3,1),OrdHerram[Posit],i);
        If i <> 0 Then OrdHerram[Posit]:=Posit
    End;
N:=0;
While Not Eof(Lach) do
    Begin
        ReadLn(Lach,Lect);
        With Lines[N] do
            Begin
                Command:=Copy(Lect,5,3);
                CoordX:=BorraEspacios(Copy(Lect,10,8));
                CoordZ:=BorraEspacios(Copy(Lect,20,8));
```

```
                    Velocidad:=BorraEspacios(Copy(Lect,30,4));
                    ProfCorte:=BorraEspacios(Copy(Lect,40,4));
                End;
            N:=N+1;
        End;
    Close(Lach);
    If Linea[N-1].Command <> 'M30' Then
        Linea[N].Command:='M30'
End; {LoadProgram}


PROCEDURE Directorio;
    VAR
        DirInfo:SearchRec;
        Archivo:String[9];
        Bloques,Atrib:Word;
        Units,Lect:String;
        dt:DateTime;
    Begin
        Cuadro(1,2,80,24,2);
        ColorText(Drv[14],Drv[8]);
        ClrScr;
        Write(' ARCHIVO          UNIDADES          DESCRIPCION                    TAMAÑO(Bytes)    BLOQUES
    FECHA');
        Window(1,2,80,24);
        ColorText(Drv[6],Drv[7]);
        GotoXY(1,3);Write('Ú');
        ColorText(Drv[6],Drv[8]);
        For i:=2 to 79 do Write('-');
        ColorText(Drv[6],Drv[7]);
        Write('¿');
        Window(2,5,79,23);
        FindFirst(Drive+':\*.CNC',Atrib,DirInfo);
        Renglon:=100;
        Flag:=false;
        While DosError = 0 do
            Begin
                Renglon:=Renglon+1;
                If Renglon >= 20 then
                    Begin
                        If Flag Then Mensaje('Oprime cualquier tecla para continuar',1);
                        ColorText(Drv[14],Drv[8]);
                        ClrScr;
                        Renglon:=1;
                        Flag:=True
                    End;
                Descripcion:='';
                Archivo:=DirInfo.Name;
                i:=Pos('.',Archivo);
                Archivo:=Copy(Archivo,1,i-1);
                {$I-}
                Assign(Lach,Drive+':\'+Archivo+'.CNC');
                Reset(Lach);
                ReadLn(Lach,Lect);
                ReadLn(Lach,Lect);
                Descripcion:=Copy(Lect,27,25);
                ReadLn(Lach,Lect);
                Units:=Copy(Lect,12,2);
                ReadLn(Lach,Lect);
                ReadLn(Lach,Lect);
                ReadLn(Lach,Lect);
                Bloques:=0;
                While Not Eof(Lach) do
                    Begin
                        Bloques:=Bloques+1;
                        ReadLn(Lach,Lect)
                    End;
                Close(Lach);
                {$I+}
                ColorText(Drv[14],Drv[8]);
                GotoXY(2,Renglon);Write(Drive,':',Archivo,'.CNC':12-Length(Archivo));
                GotoXY(18,renglon);Write(Units);
```

```
                GotoXY(23,Ranglon);Write(Descripcion);
                GotoXY(50,Renglon);Write(Dirinfo.Size:8);
                GotoXY(60,Renglon);Write(Bloques:4);
                UnPackTime(Dirinfo.Time,Dt);
                With Dt do
                    Begin
                        GotoXY(68,Renglon);
                        Write(Copy(StrNum(Day+100,0,0),2,2),'/',Copy(StrNum(Month+100,0,0),2,2),'/',Year)
                    End;
                FindNext(Dirinfo)
            End;
        Case Drive[1] of
            'A':Atrib:=1;
            'B':Atrib:=2;
            'C':Atrib:=3
        End;
        Mensaje(StrNum(DiskFree(Atrib) div 1024,0,0)+'Kb libres en el Drive '+Drive,1);
        ColorText(15,Drv[1])
    End;

END.
```

PREPARA.PAS

```
PROCEDURE Prepara;
    Var
        CodArch:Text;
        j,k:integer;
        ForceDrive:Char;
    Begin
        If ParamCount < 1
            Then
                DetectGraph(Driver,Mode)
            Else
                Begin
                    DriverName:=ParamStr(1);
                    ForceDrive:=UpCase(DriverName[1]);
                    Case ForceDrive of
                        'C':Driver:=1;  {CGA}
                        'M':Driver:=2;  {MCGA}
                        'E':Driver:=3;  {EGA}
                        'H':Driver:=7;  {HERCULES}
                        'V':Driver:=9;  {VGA}
                        '?':Begin
                                    Writeln(Version,' Parámetros de entrada:');
                                    Writeln('          c - CGA');
                                    Writeln('          m - MCGA');
                                    Writeln('          e - EGA');
                                    Writeln('          h - Hercules');
                                    Writeln('          v - VGA');
                                    Halt
                                End
                        Else Driver:=0
                    End
                End;
        Case Driver Of
            1:Begin
                    DriverName:='CGA';
                    Mode:=CGAHi
                End;
            2:Begin
                    DriverName:='MCGA';
                    Mode:=MCGAHi
                End;
            3:Begin
                    DriverName:='EGA';
                    Mode:=EGAHi
                End;
            7:Begin
                    DriverName:='HERC';
                    Mode:=HercMonoHi
                End;
            9:Begin
                    DriverName:='VGA';
                    Mode:=VGAHi
                End;
            Else Begin
                        Writeln(Version,', No existe soporte gráfico');
                        Halt
                    End;
        End; {case}
        DriverName:=Util1Dir+DriverName+'TORH.DRV';
        Assign(CodArch,DriverName);
        {$I-}
        Reset(CodArch);
        i:=IOResult;
        {$I+}
        If i <> 0 Then
            Begin
                Writeln(Version,', ',GetIOError(i),', ',DriverName);
                Halt
            End;
        j:=0;
```

```
Writeln;
WriteLn('Leyendo archivo ',DriverName);
While (Not Eof(CodArch)) and (j<MaxDrv) do
    Begin
        j:=j+1;
        St:='                    ';
        Readln(CodArch,St);
        Val(St,k,i);
        If i<>0
            Then
                Begin
                    Writeln(Version,', #',j,' Error de inicialización, ',DriverName);
                    Halt
                End
            Else
        Drv[j]:=k
    End;
Assign(CodArch,UtilDir+'CODIGO.CN');
{$I-}
Reset(CodArch);
i:=IOResult;
{$I+}
If i <> 0 Then
    Begin
        Writeln(Version,', ',GetIOError(i),' ',UtilDir,'CODIGO.CN');
        Halt
    End;
For i:=0 to 199 do
For j :=2 to 6 do Comando[i,j]:=' ';
TotalCommands:=0;
WriteLn('Leyendo ',UtilDir,'CODIGO.CN');
While not Eof(CodArch) do
    Begin
        St:='                    ';
        Readln(CodArch,St);
        If UpCase(St[1]) = 'M'
            Then
                j:=100
            Else
                If UpCase(St[1]) = 'G'
                    Then
                        j:=0
                    Else
                        j:=-1;
        If j>=0 Then
            Begin
                Val(Copy(St,2,2),k,i);
                If (i = 0) and (k >= 0) and (k<100) Then
                    Begin
                        TotalCommands:=TotalCommands+1;
                        Comando[j+k,2]:=St[1];
                        Comando[j+k,3]:=St[4];
                        Comando[j+k,4]:=St[5];
                        Comando[j+k,5]:=St[6];
                        Comando[j+k,6]:=St[7];
                    End
            End
    End;
Close(CodArch);
Writeln('Inicializando programa CNC');
Renglon:=0;
Unidades:=0;                {No definido}
If ProgDir in ['A'..'Z']
    Then
        Drive:=ProgDir
    Else
        GetDir(0,Drive);
Programa:='';
InitProgramCNC;     {Inicializa programa CNC}
DiamatroBroca:=0;
Delay(500);
InitGraphics;
```

```
    RestoreCRTMode
End;
```

SIMCOMM.PAS

```
PROCEDURE G00;
    Begin
        If not ValorComando(Xf,Zf,oo3,oo4) Then Exit;
        INCREMENTAL;
        LINEAL(Xf,Zf,0);
    End;

PROCEDURE G01;
    Begin
        If Not ValorComando(Xf,Zf,Feed,oo4) Then Exit;
        INCREMENTAL;
        LINEAL(Xf,Zf,1);
    End;

PROCEDURE G02;
    Const
        Error=10;
    Var
        Xr,Zr,II,JJ,Segmentos:Integer;
        Delta,Da,Db,Xc,Zc,Xa,Za,Xb,Zb,Th,Tha,Thb,Radio:Real;
    Begin
        If Not ValorComando(Xf,Zf,Feed,oo4) Then Exit;
        Incremental;
        If Variable(N+1,2) = 'M99'
            Then
                Begin
                    N:=N+1;
                    If Not ValorComando(II,JJ,oo3,oo4) Then Exit;
                    Zc:=Zi+Sgn(Xf-Xi)*JJ;
                    Xc:=Xi/2+II;
                    If Xf < Xi
                        Then
                            Begin
                                If (Zc > Zf) Then
                                    Begin
                                        RunMessage(Variable(N-1,2) +    ',M99 ERROR DE CUADRANTE',

    'Zc'+StrNum(Zc,0,0)+'>Zf'+StrNum(Zf,0,0),
                                                                        'Xc'+StrNum(Xc,0,0),1);
                                        Exit
                                    End
                            End
                        Else
                            Begin
                                If (Zc < Zi) Then
                                    Begin
                                        RunMessage(Variable(N-1,2) +    ',M99 ERROR DE CUADRANTE',

    'Zc'+StrNum(Zc,0,0)+'<Zi'+StrNum(Zi,0,0),
                                                                        'Xc'+StrNum(Xc,0,0),1);
                                        Exit
                                    End
                            End
                End
            Else
                If Xf < Xi
                    Then
                        Begin
                            Xc:=Xi/2;
                            Zc:=Zf
                        End
                    Else
                        Begin
                            Xc:=Xf/2;
                            Zc:=Zi
                        End;
        Xa:=Xi/2;
```

```
        Za:=Zi;
        Xb:=Xf/2;
        Zb:=Zf;
        Da:=Sqrt(Sqr(Xc-Xa)+Sqr(Zc-Za));
        Db:=Sqrt(Sqr(Xc-Xb)+Sqr(Zc-Zb));
        If Abs(Db-Da)>Error Then
            Begin
                RunMessage(  'G03 ERROR DE CALCULO',
                             'Da = '+StrNum(Da,0,0),
                             'Db = '+StrNum(Db,0,0),1);
                Exit
            End;
        Radio:=Da;
        Tha:=-Abs(angulo(Xa-Xc,Za-Zc));
        Thb:=-Abs(angulo(Xb-Xc,Zb-Zc));
        Segmentos:=Round(Abs(Thb-Tha)/0.0175);
        Delta:=(Thb-Tha)/Segmentos;
        Z:=Zi;
        X:=Xi;
        Th:=Tha;
        While (Abs(Round(Z)-Zf) > Error) or (Abs(Round(X)-Xf) > Error) do
            Begin
                Th:=Th+Delta;
                Xr:=Round((Xc+Radio*Sin(Th))*2);
                Zr:=Round(Zc+Radio*Cos(Th));
                LINEAL(Xr,Zr,1);If OutputFlag then exit
            End;
    End;

PROCEDURE G03;
    Const
        Error=10;   {0.1 mm}
    Var
        Xr,Zr,I1,JJ,Segmentos:Integer;
        Delta,Da,Db,Xc,Zc,Xa,Za,Xb,Zb,Th,Tha,Thb,Radio:Real;
    Begin
        If Not ValorComando(Xf,Zf,Feed,oo4) Then Exit;
        Incremental;
        If Variable[N+1,2] = 'M99'
            Then
                Begin
                    N:=N+1;
                    If Not ValorComando(I1,JJ,oo3,oo4) Then Exit;
                    Zc:=Zi-Sgn(Xf-Xi)*JJ;
                    Xc:=Xi/2-I1;
                    If Xf > Xi
                        Then
                            Begin
                                If (Zc > Zf) Then
                                    Begin
                                        RunMessage(Variable[N-1,2] +    ',M99 ERROR DE CUADRANTE',

'Zc'+StrNum(Zc,0,0)+'>Zf'+StrNum(Zf,0,0),
                                                                        'Xc'+StrNum(Xc,0,0),1);
                                        Exit
                                    End
                            End
                        Else
                            Begin
                                If (Zc < Zi) Then
                                    Begin
                                        RunMessage(Variable[N-1,2] +    ',M99 ERROR DE CUADRANTE',

'Zc'+StrNum(Zc,0,0)+'<Zi'+StrNum(Zi,0,0),
                                                                        'Xc'+StrNum(Xc,0,0),1);
                                        Exit
                                    End
                            End
                End
            Else
                If Xf > Xi
                    Then
```

```
                    Begin
                        Xc:=Xi/2;
                        Zc:=Zf
                    End
                Else
                    Begin
                        Xc:=Xf/2;
                        Zc:=Zi
                    End;
        Xa:=Xi/2;
        Za:=Zi;
        Xb:=Xf/2;
        Zb:=Zf;
        Da:=Sqrt(Sqr(Xc-Xa)+Sqr(Zc-Za));
        Db:=Sqrt(Sqr(Xc-Xb)+Sqr(Zc-Zb));
        If Abs(Db-Da)>Error Then
            Begin
                RunMessage(  'G02 ERROR DE CALCULO',
                             'Da = '+StrNum(Da,0,0),
                             'Db = '+StrNum(Db,0,0),1);
                Exit
            End;
        Radio:=Da;
        Tha:=angulo(Xa-Xc,Za-Zc);
        Thb:=angulo(Xb-Xc,Zb-Zc);
        Segmentos:=Round(Abs(Thb-Tha)/0.0175);
        Delta:=(Thb-Tha)/Segmentos;
        Z:=Zi;
        X:=Xi;
        Th:=Tha;
        While (Abs(Round(Z)-Zf) > Error) or (Abs(Round(X)-Xf) > Error) do
            Begin
                Th:=Th+Delta;
                Xr:=Round((Xc+Radio*Sin(Th))*2);
                Zr:=Round(Zc+Radio*Cos(Th));
                LINEAL(Xr,Zr,1);If OutputFlag Then Exit;
            End;
    End;

PROCEDURE G04;
    Var
        time:integer;
    Begin
        If Not ValorComando(Time,oo2,oo3,oo4) Then Exit;
        Delay(time*10)
    End;

PROCEDURE G25;
    Var
        Jump:Integer;
    Begin
        If Not ValorComando(oo1,oo2,Jump,oo4) Then Exit;
        i:=MaxLineas;
        While Linea[i].Command<>'H30' do i:=i-1;
        If (Jump>0) and (Jump<i)
            Then
                Begin
                    SubNivel:=SubNivel+1;
                    If SubNivel > MaxNiveles
                        Then
                            RunMessage(  'G25 EL NUMERO MAXIMO DE',
                                         'NIVELES HA SIDO REBASADO','','',1)
                        Else
                            Begin
                                SubLinea[SubNivel]:=N;
                                N:=Jump-1
                            End
                End
            Else
                RunMessage('G25 LINEA N'+StrNum(Jump,0,0)+' INEXISTENTE','','',1);
    End;
```

```
PROCEDURE G27;
    Var
        Jump:Integer;
    Begin
        If Not ValorComando(oo1,oo2,Jump,oo4) Then Exit;
        i:=MaxLineas;
        While Linea[i].Command<>'M30' do i:=i-1;
        If (Jump>0) and (Jump<=i)
            Then
                N:=Jump-1
            Else
                RunMessage('G27 LINEA N'+StrNum(Jump,0,0)+' INEXISTENTE','','',1);
    End;

PROCEDURE G33;
    Var
        Hilos,Paso,dZ:Integer;
        Dt:Real;
    Begin
        If Not ValorComando(oo1,Zf,Paso,oo4) Then Exit;
        INCREMENTAL;
        X:=X1;
        Z:=Z1;
        Feed:=Round(Paso*Spindle/100);
        Dt:=Paso/Feed/100;
        Hilos:=Trunc(Abs(Zf-Z1)/Paso);
        i:=0;
        Status;
        If FastSim
            Then
                Begin
                    Dt:=Dt*hilos;
                    Z:=Zf;
                    Trayect;
                    Tiempo:=Tiempo+Dt;
                    OutPutFlag:= Cancel or OutPutFlag;
                    If OutputFlag Then Exit
                End
            Else
                Begin
                    While i < Hilos do
                        Begin
                            i:=i+1;
                            Z:=Z+Paso*Sgn(Zf-Z1);
                            Rosca(Paso);
                            Tiempo:=Tiempo+Dt;
                            OutPutFlag:= Cancel or OutPutFlag;
                            If OutputFlag Then Exit
                        End;
                    Z:=Zf;
                    Tracking;
                End;
        X1:=Round(X);
        Z1:=Round(Z)
    End;

PROCEDURE G70;
    Begin
        UnitFactor:=1000;    {milésimas de pulgada}
    End;

PROCEDURE G71;
    Begin
        UnitFactor:=100;     {centésimas de milimetro}
    End;

PROCEDURE G73;
    Var
        Inicio,Prof,Paso,Retro:Integer;
    Begin
        Paso:=200;
        Retro:=20;
```

```
        If Not ValorComando(oo1,Zf,Feed,oo4) Then Exit;
        If DiametroBroca<BrocaMin Then
            Begin
                RunMessage(  'EL DIAMETRO DE HERRAMIENTA',
                             'INCORRECTO','',1);
                Exit
            End;
        If X1 <> 0 Then
            Begin
                RunMessage('CENTRO DE BROCA ERRONEO','','',1);
                Exit
            End;
        INCREMENTAL;
        If Zf > Z1 Then
            Begin
                RunMessage('DIRECCION DE BARRENO INCORRECTA','','',1);
                Exit
            End;
        Prof:=Z1;
        Inicio:=Z1;
        While Prof-Paso > Zf do
            Begin
                Prof:=Prof-Paso;
                LINEAL(X1,Prof,1);If OutPutFlag Then Exit;
                Prof:=Prof+Retro;
                LINEAL(X1,Prof,0);If OutPutFlag Then Exit;
            End;
        LINEAL(X1,Zf,1);If OutPutFlag Then Exit;
        LINEAL(X1,Inicio,0);
        X1:=Round(X);
        Z1:=Round(Z)
    End;

PROCEDURE G78;
    Var
        Hilos,Xp,H,Xo,Zo,Fs,Paso,Pasos,dZ:Integer;
        Dt:Real;
    Procedure CicloRosca;
        Begin
            X:=X1;
            Z:=Z1;
            Fs:=Abs(Round(Paso*Spindle/100));
            Dt:=Paso/Fs/100;
            Hilos:=Trunc(Abs(Zf-Z1)/Paso);
            i:=0;
            Feed:=Fs;
            Status;
            If FastSim
                Then
                    Begin
                        Dt:=Dt*hilos;
                        Z:=Zf;
                        Trayect;
                        Tiempo:=Tiempo+Dt;
                        OutPutFlag:= Cancel or OutPutFlag;
                        If OutputFlag Then Exit
                    End
                Else
                    Begin
                        While i < Hilos do
                            Begin
                                i:=i+1;
                                Z:=Z+Paso*Sgn(Zf-Z1);
                                Rosca(Paso);
                                Tiempo:=Tiempo+Dt;
                                OutPutFlag:= Cancel or OutPutFlag;
                                If OutputFlag Then Exit
                            End;
                        Z:=Zf;
                        Tracking
                    End;
            X1:=Round(X);
```

```
            Zi:=Round(Z)
        End; {Rosca}
    Begin
        If Not ValorComando(Xf,Zf,Paso,H) Then Exit;
        INCREMENTAL;
        Pasos:=Abs(Trunc((Xf-Xi)/(2*H)));
        H:=H*Sgn(Xf-Xi);
        k:=0;
        Xo:=Xi;
        Zo:=Zi;
        While k<Pasos do
            Begin
                k:=k+1;
                Xp:=Xi+H*k*2;
                LINEAL(Xp,Zo,0);If Outputflag Then Exit;
                CICLOROSCA;      If Outputflag Then Exit;
                LINEAL(Xo,Zf,0);If Outputflag Then Exit;
                LINEAL(Xo,Zo,0);If Outputflag Then Exit;
            End;
        If Xp<>Xf Then
            Begin
                LINEAL(Xf,Zo,0);If Outputflag Then Exit;
                CICLOROSCA;      If Outputflag Then Exit;
                LINEAL(Xo,Zf,0);If Outputflag Then Exit;
                LINEAL(Xo,Zo,0);If Outputflag Then Exit;
            End;
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G81;
    Var
        Inicio:Integer;
    Begin
        If Not ValorComando(oo1,Zf,Feed,oo4) Then Exit;
        If DiametroBroca<BrocaMin Then
            Begin
                RunMessage(  'EL DIAMETRO DE HERRAMIENTA',
                             'INCORRECTO','',1);
                Exit
            End;
        If Xi <> 0 Then
            Begin
                RunMessage('CENTRO DE BROCA ERRONEO','','',1);
                Exit
            End;
        INCREMENTAL;
        If Zf > Zi Then
            Begin
                RunMessage('DIRECCION DE BARRENO INCORRECTA','','',1);
                Exit
            End;
        Inicio:=Zi;
        LINEAL(Xi,Zf,1);If OutPutFlag Then Exit;
        LINEAL(Xi,Inicio,0);If OutputFlag Then exit;
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G82;
    Var
        Inicio:Integer;
    Begin
        If Not ValorComando(oo1,Zf,Feed,oo4) Then Exit;
        If DiametroBroca<BrocaMin Then
            Begin
                RunMessage(  'EL DIAMETRO DE HERRAMIENTA',
                             'INCORRECTO','',1);
                Exit
            End;
        If Xi <> 0 Then
            Begin
```

```
                RunMessage('CENTRO DE BROCA ERRONEO','','',1);
                Exit
            End;
        INCREMENTAL;
        If Zf > Zi Then
            Begin
                RunMessage('DIRECCION DE BARRENO INCORRECTA','','',1);
                Exit
            End;
        Inicio:=Zi;
        LINEAL(Xi,Zf,1);If OutPutFlag Then Exit;
        Feed:=0;Status;
        Delay(500);
        Feed:=FastFeed;
        LINEAL(Xi,Inicio,0);
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G83;
    Var
        Inter,Inicio,Prof,Paso,Retro:Integer;
    Begin
        Paso:=600;
        Retro:=50;
        If Not ValorComando(oo1,Zf,Feed,oo4) Then Exit;
        If DiametroBroca<BrocaMin Then
            Begin
                RunMessage( 'EL DIAMETRO DE HERRAMIENTA',
                            'INCORRECTO','',1);
                Exit
            End;
        If Xi <> 0 Then
            Begin
                RunMessage('CENTRO DE BROCA ERRONEO','','',1);
                Exit
            End;
        INCREMENTAL;
        If Zf > Zi Then
            Begin
                RunMessage('DIRECCION DE BARRENO INCORRECTA','','',1);
                Exit
            End;
        Prof:=Zi;
        Inicio:=Zi;
        While Prof-Paso > Zf do
            Begin
                Prof:=Prof-Paso;
                LINEAL(Xi,Prof,1);If OutPutFlag Then Exit;
                Prof:=Prof+Retro;
                LINEAL(Xi,Inicio,0);If OutPutFlag Then Exit;
                LINEAL(Xi,Prof,0);If OutPutFlag Then Exit;
            End;
        LINEAL(Xi,Zf,1);If OutPutFlag Then Exit;
        LINEAL(Xi,Inicio,0);
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G84;
    Var
        Xp,H,Pasos,Xo,Zo:Integer;
    Begin
        If Not ValorComando(Xf,Zf,Feed,H) Then Exit;
        INCREMENTAL;
        Pasos:=1;
        If H = 0
            Then
                k:=Pasos
            Else
                Begin
                    k:=0;
```

```
                Pasos:=Abs(Trunc((Xf-Xi)/(2*H)));
                H:=H*Sgn(Xf-Xi)
            End;
        If Pasos = 0 Then
            Begin
                RunMessage('VALOR INCORRECTO DE H','','',1);
                Exit
            End;
        Xo:=Xi;
        Zo:=Zi;
        While k<Pasos do
            Begin
                k:=k+1;
                Xp:=Xi+H*k*2;
                LINEAL(Xp,Zo,0);If Outputflag Then Exit;
                LINEAL(Xp,Zf,1);If Outputflag Then Exit;
                LINEAL(Xo,Zf,1);If Outputflag Then Exit;
                LINEAL(Xo,Zo,0);If Outputflag Then Exit;
            End;
        If Xp<>Xf Then
            Begin
                LINEAL(Xf,Zo,0);If Outputflag Then Exit;
                LINEAL(Xf,Zf,1);If Outputflag Then Exit;
                LINEAL(Xo,Zf,1);If Outputflag Then Exit;
                LINEAL(Xo,Zo,0);If Outputflag Then Exit;
            End
    End;

PROCEDURE G85;
    Var
        Inicio:Integer;
    Begin
        If Not ValorComando(ool,Zf,Feed,oo4) Then Exit;
        If DiametroBroca<BrocaMin Then
            Begin
                RunMessage(  'EL DIAMETRO DE HERRAMIENTA',
                             'INCORRECTO','',1);
                Exit
            End;
        If Xi <> 0 Then
            Begin
                RunMessage('CENTRO DE RIMA ERRONEO','','',1);
                Exit
            End;
        INCREMENTAL;
        If Zf > Zi Then
            Begin
                RunMessage('DIRECCION DE RIMADO INCORRECTA','','',1);
                Exit
            End;
        Inicio:=Zi;
        LINEAL(Xi,Zf,1);If OutPutFlag Then Exit;
        LINEAL(Xi,Inicio,1);
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G86;
    Var
        InicioZ,InicioX,Zp,OffSet,Pasos,Avance,H:Integer;
    Begin
        If Not ValorComando(Xf,Zf,Feed,H) Then Exit;
        INCREMENTAL;
        If Abs(Zf-Zi) < H Then
            Begin
                RunMessage('HERRAMIENTA DEMASIADO GRANDE','','',1);
                exit
            End;
        Avance:=Round(Sgn(Zf-Zi)*H*(1-1/10));
        Zp:=Zi;
        InicioX:=Xi;
        InicioZ:=Zi;
```

```
        If Zf < Zi
            Then
                Begin
                    While Zp - H > Zf do
                        Begin
                            LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                            LINEAL(Xf,Zp,1);If OutputFlag Then Exit;
                            LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                            Zp:=Zp+Avance;
                        End;
                    Zp:=Zf+H;
                    LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                    LINEAL(Xf,Zp,1);If OutputFlag Then Exit;
                    LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                    LINEAL(InicioX,InicioZ,0);If OutputFlag Then Exit;
                End
            Else
                Begin
                    Zp:=Zp+H;
                    While Zp < Zf do
                        Begin
                            LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                            LINEAL(Xf,Zp,1);If OutputFlag Then Exit;
                            LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                            Zp:=Zp+Avance
                        End;
                    Zp:=Zf;
                    LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                    LINEAL(Xf,Zp,1);If OutputFlag Then Exit;
                    LINEAL(InicioX,Zp,0);If OutputFlag Then Exit;
                    LINEAL(InicioX,InicioZ,0);If OutputFlag Then Exit;
                End;
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G88;
    Var
        Zp,H,Pasos,Xo,Zo:Integer;
    Begin
        If Not ValorComando(Xf,Zf,Feed,H) Then Exit;
        INCREMENTAL;
        If H = 0
            Then
                k:=Pasos
            Else
                Begin
                    k:=0;
                    Pasos:=Abs(Trunc((Zf-Zi)/H));
                    H:=H*Sgn(Zf-Zi)
                End;
        Xo:=Xi;
        Zo:=Zi;
        While k<Pasos do
            Begin
                k:=k+1;
                Zp:=Zi+H*k;
                LINEAL(Xo,Zp,0);If Outputflag Then Exit;
                LINEAL(Xf,Zp,1);If Outputflag Then Exit;
                LINEAL(Xf,Zo,1);If Outputflag Then Exit;
                LINEAL(Xo,Zo,0);If Outputflag Then Exit;
            End;
        If Zp<>Zf Then
            Begin
                LINEAL(Xo,Zf,0);If Outputflag Then Exit;
                LINEAL(Xf,Zf,1);If Outputflag Then Exit;
                LINEAL(Xf,Zo,1);If Outputflag Then Exit;
                LINEAL(Xo,Zo,0);If Outputflag Then Exit
            End
    End;

PROCEDURE G89;
```

```
    Var
        Inicio,Fo:Integer;
    Begin
        If Not ValorComando(oo1,Zf,Feed,oo4) Then Exit;
        If DiametroBroca<BrocaMin Then
            Begin
                RunMessage(  'EL DIAMETRO DE HERRAMIENTA',
                             'INCORRECTO','',1);
                Exit
            End;
        If Xi <> 0 Then
            Begin
                RunMessage('CENTRO DE RIMA ERRONEO','','',1);
                Exit
            End;
        INCREMENTAL;
        If Zf > Zi Then
            Begin
                RunMessage('DIRECCION DE RIMADO INCORRECTA','','',1);
                Exit
            End;
        Inicio:=Zi;
        LINEAL(Xi,Zf,1);If OutPutFlag Then Exit;
        Fo:=Feed;
        Feed:=0;Status;
        Delay(500);
        Feed:=Fo;
        LINEAL(Xi,Inicio,1);
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G90;
    Begin
        Incr:=2;
        Feed:=0;
        Status;
    End;

PROCEDURE G91;
    Begin
        Incr:=1;
        Feed:=0;
        Status;
    End;

PROCEDURE G92;
    Begin
        If Not ValorComando(Xf,Zf,oo3,oo4) Then Exit;
        Z:=Zf;
        X:=Xf;
        Feed:=0;
        Incr:=2;
        Status;
        Tracking;
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE G94;
    Begin
        AvancePorMin:=True
    End;

PROCEDURE G95;
    Begin
        AvancePorMin:=False
    End;

PROCEDURE M00;
    Begin
        MessagePort;
```

```
        ClearViewPort;
        SetColor(Drv[92]);
        DrawText(1,1,'Paro programado',1);
        DrawText(1,2,'[Espacio] para continuar',1);
        Repeat Until ReadKey = ' ';
        Funciones;
    End;

PROCEDURE M03;
    Begin
        Spindle:=Husillo;
        Status
    End;

PROCEDURE M04;
    Begin
        Spindle:=-Husillo;
        Status
    End;

PROCEDURE M05;
    Begin
        Spindle:=0;
        Status;
    End;

PROCEDURE M06;
    Var
        T1,DiamBr:Integer;
    Begin
        If Not ValorComando(Xf,Zf,T1,DiamBr) Then Exit;
        WorkPort;
        Herramienta(OrdHerram[Tool],Z1,X1,0,Round(Escala*100/UnitFactor));
        Tool:=T1;
        DiametroBroca:=DiamBr;
        Tracking
    End;

PROCEDURE M17;
    Begin
        If SubNivel < 1
            Then
                Begin
                    SubNivel:=0;
                    RunMessage( 'M17 SIN LLAMADA G25',
                                'DE SUBRUTINA','',1);
                End
            Else
                Begin
                    N:=SubLinea[SubNivel];
                    SubNivel:=SubNivel-1
                End
    End;

PROCEDURE M30;
    Begin
        Spindle:=0;
        Feed:=0;
        Status
    End;

PROCEDURE M98;
    Begin
        {Este procedimiento no tiene efecto en la simulacion}
        Tracking
    End;
```

SIMPROC.PAS

```
Function Velocidad:Real;
    Var
        v:Real;
    Begin
        v:=Spindle^X/UnitFactor;
        If Unidades = 1
            Then Velocidad:=v/1000 {metros por minuto}
            Else Velocidad:=v/12;    {pies por minuto}
    End;

Function Sgn(X:Real):Integer;
    Begin
        If X<0
            Then Sgn:=-1
            Else Sgn:=1
    End;

{CONTROL DE PANTALLA}

Procedure WorkPort;
    Begin
        SetViewPort(VPx1+1,VPy1+1,VPx2-1,VPy2-1,ClipOn)
    End;

Procedure LinePort;
    Begin
        SetViewPort(StPg+5,0,StLn-1,VPy1-1,ClipOn)
    End;

Procedure MessagePort;
    Begin
        SetViewPort(StLn+5,0,GetMaxX,VPy1-1,ClipOn)
    End;

Procedure Funciones;
    Begin
        MessagePort;
        ClearViewPort;
        SetColor(Drv[92]);
        DrawText(1,1,'[Esc] PAUSA [F1] TIEMPO',1);
        DrawText(1,2,'[F2] VELOCIDAD DE CORTE',1);
    End;

Procedure RunMessage(Linea1,Linea2,Linea3:String30;Salida:Byte);
    Begin
        MessagePort;
        ClearViewPort;
        Beep;
        DrawText(1,1,Linea1,1);
        DrawText(1,2,Linea2,1);
        DrawText(1,3,Linea3,1);
        Car:=ReadKey;
        Funciones;
        OutPutFlag:= (Salida = 1)
    End;

Procedure Coordenadas;
    Begin
        SetViewPort(0,VPy2+3,St12-1,GetMaxY,ClipOn);
        ClearViewPort;
        SetColor(Drv[91]);
        DrawText(2,1,'X'+StrNum(Round(X),0,0),Drv[100]);
        DrawText(10,1,'Z'+StrNum(Round(Z),0,0),Drv[100]);
        If Z/UnitFactor < -Dimensiones.Largo Then
        Begin
            Beep;
            RunMessages('CONTACTO CON EL HUSILLO','','',1)
        End;
    End;
```

```
Procedure Tracking;
    Begin
        WorkPort;
        X2:=X0+Round(Escala*RelY*X/(2*UnitFactor));
        Z2:=Z0+Round(Escala*RelX*Z/UnitFactor);
        Xm2:=X0-Round(Escala*RelY*X/(2*UnitFactor));
        Zm2:=Z2;
        Herramienta(OrdHerram[Tool],Z1,X1,0,Round(Escala*100/UnitFactor));
        Herramienta(OrdHerram[Tool],Z2,X2,1,Round(Escala*100/UnitFactor));
        Herramienta(-OrdHerram[Tool],Zm2,Xm2,0,Round(Escala*100/UnitFactor));
        X1:=X2;
        Z1:=Z2;
        Coordenadas
    End;

Procedure Trayect;
    Begin
        WorkPort;
        X2:=X0+Round(Escala*RelY*X/(2*UnitFactor));
        Z2:=Z0+Round(Escala*RelX*Z/UnitFactor);
        Xm2:=X0-Round(Escala*RelY*X/(2*UnitFactor));
        Zm2:=Z2;
        Line(Z1,X1,Z2,X2);
        X1:=X2;
        Z1:=Z2;
        Coordenadas
    End;

Procedure Rosca(Paso:Integer);
    Begin
        WorkPort;
        X2:=X0+Round(Escala*RelY*X/(2*UnitFactor));
        Z2:=Z0+Round(Escala*RelX*Z/UnitFactor);
        Xm2:=X0-Round(Escala*RelY*X/(2*UnitFactor));
        Zm2:=Z0+Round(Escala*RelX*(Z-Paso*Sgn(Zf-Zi)/2)/UnitFactor);
        Herramienta(OrdHerram[Tool],Z1,X1,0,Round(Escala*100/UnitFactor));
        Herramienta(OrdHerram[Tool],Z2,X2,1,Round(Escala*100/UnitFactor));
        Herramienta(-OrdHerram[Tool],Zm2,Xm2,0,Round(Escala*100/UnitFactor));
        X1:=X2;
        Z1:=Z2;
        Coordenadas
    End;

Procedure Status;
    Const
        Sys:Array [1..2] of String[11] = ('Incremental','Absoluto');
    Var
        TipoAvance:String;
        Avance:Real;
    Begin
        If Unidades = 1
            Then TipoAvance:=' mm/'
            Else TipoAvance:=' in/';
        If AvancePorMin
            Then TipoAvance:=TipoAvance+'min'
            Else TipoAvance:=TipoAvance+'rev';
        SetViewPort(St12,VPy2+1,GetMaxX,GetMaxY,ClipOn);
        ClearViewPort;
        SetColor(Drv[90]);
        Line(1,0,1,GetMaxY);
        SetViewPort(St12,VPy2+3,GetMaxX,GetMaxY,ClipOn);
        SetColor(Drv[98]);
        DrawText(2,1,'T '+StrNum(Tool,0,0),1);
        DrawText(10,1,'S '+StrNum(Spindle,0,0)+' RPM',1);
        Avance:=100*(Feed/UnitFactor);
        DrawText(25,1,'F '+StrNum(Avance,0,2)+TipoAvance,1);
        DrawText(45,1,Sys[Incr],1);
        FullPort
    End;

Function Opciones:Char;
```

```
    Var
        PausaPrograma,Funcion:Char;
    Begin
        Funcion:='C';
        PausaPrograma:=ReadKey;
        If PausaPrograma in [#27,#13] Then Funcion:=PausaPrograma;
        If PausaPrograma = #0 Then Funcion:=ReadKey;
        Case Funcion of
            #27:Begin
                        SetColor(Drv[92]);
                        MessagePort;
                        ClearViewPort;
                        DrawText(1,1,'PAUSA [A]bortar, [C]ontinuar',1);
                        Repeat Funcion:=UpCase(ReadKey) Until Funcion in ['A','C'];
                        Funciones
                    End;
            #59:Begin
                        SetColor(Drv[92]);
                        MessagePort;
                        ClearViewPort;
                        DrawText(1,1,'CALCULO DE TIEMPO APROXIMADO:',1);
                        DrawText(1,2,StrNum(Tiempo,0,2)+' '+'minutos',1);
                        DrawText(1,3,'Oprimir cualquier tecla',1);
                        Funcion:=ReadKey;
                        Funciones
                    End;
            #60:Begin
                        SetColor(Drv[92]);
                        MessagePort;
                        ClearViewPort;
                        If Unidades = 1
                            Then St:=' m/min'
                            Else St:=' ft/min';
                        DrawText(1,1,'v = '+StrNum(Velocidad,0,2)+St,1);
                        DrawText(1,3,'Oprimir cualquier tecla',1);
                        Funcion:=ReadKey;
                        Funciones
                    End;
        End;
        Opciones:=Funcion
    End;

Function Cancel:Boolean;
    Begin
        Cancel:=False;
        If KeyPressed Then Cancel:= (Opciones = 'A');
    End;

Procedure Ejes;
    Var
        v:Real;
        Indice,MaxX,MaxY,TmLn:Integer;
    Begin
        WorkPort;
        SetColor(Drv[96]);
        MaxY:=VPy2-VPy1-1;
        SetLineStyle(SolidLn,0,3);
        Line(Z0,0,Z0,Round(Drv[103]*4*RelY));
        Line(Z0,MaxY,Z0,Round(MaxY-Drv[103]*4*RelY));
        SetLineStyle(SolidLn,0,1);
        Indice:=0;
        Repeat
            Indice:=Indice+5;
            v:=Indice*Escala*RelX*100/UnitFactor;
            If Indice mod 10 = 0
                Then TmLn:=Drv[103]*2
                Else TmLn:=Drv[103];
            Line(Z0+Round(v),0,Z0+Round(v),Round(TmLn*RelY));
            Line(Z0-Round(v),0,Z0-Round(v),Round(TmLn*RelY));
            Line(Z0+Round(v),MaxY,Z0+Round(v),Round(MaxY-TmLn*RelY));
            Line(Z0-Round(v),MaxY,Z0-Round(v),Round(MaxY-TmLn*RelY));
        Until v > Z0;
```

```
        MaxX:=VPx2-VPx1-1;
        SetLineStyle(SolidLn,0,3);
        Line(MaxX,X0,MaxX-Round(Drv[104]*4*RelX),X0);
        SetLineStyle(SolidLn,0,1);
        Indice:=0;
        Repeat
            Indice:=Indice+5;
            v:=Indice*Escala*RelY*100/UnitFactor;
            If Indice mod 10 = 0
                Then TmLn:=Drv[104]*2
                Else TmLn:=Drv[104];
            Line(MaxX,X0+Round(v),Round(MaxX-TmLn*RelX),X0+Round(v));
            Line(MaxX,X0-Round(v),Round(MaxX-TmLn*RelX),X0-Round(v));
        Until v > X0;
    End; {Ejes}

{CALCULOS DE SIMULACION}

VAR
    oo1,oo2,oo3,oo4:Integer;

Function Angulo(a,b:Real):Real;
    Var
        m:Real;
        s:Integer;
    Begin
        If b = 0 Then
            Begin
                If a<0
                    Then
                        angulo:=-pi/2
                    Else
                        angulo:=pi/2;
                Exit
            End;
        m:=Abs(a/b);
        If (a>=0) and (b>0) Then angulo:=arctan(m);
        If (a>=0) and (b<0) Then angulo:=pi-arctan(m);
        If (a<0) and (b<0) then angulo:=-pi+arctan(m);
        If (a<0) and (b>0) then angulo:=-arctan(m)
    End;

PROCEDURE Lineal(FinalX,FinalZ,Delta:integer); {Coordenadas absolutas}
    Var
        Fo:Integer;
        Segments,Pixels,Distancia,PasoX,PasoZ,DeltaTiempo:Real;
    Begin
        Fo:=Feed;
        If Delta = 0 Then
            Begin
                Delta:=Drv[106];
                Feed:=FastFeed
            End;
        Status;
        If Abs(FinalX-Xi)*RelY/2 >= Abs(FinalZ-Zi)*RelX
            Then
                Begin
                    Pixels:=Unitfactor/(Escala*RelY)*2;
                    Segments:=Abs(FinalX-Xi)/Pixels
                End
            Else
                Begin
                    Pixels:=UnitFactor/(Escala*RelX);
                    Segments:=Abs(FinalZ-Zi)/Pixels
                End;
        Segments:=Int(Segments/Delta)+1;
        If FastSim Then Segments:=1;
        PasoX:=(FinalX-Xi)/Segments;
        PasoZ:=(FinalZ-Zi)/Segments;
        X:=Xi;
        Z:=Zi;
        Distancia:=Sqrt(Sqr(PasoX)+Sqr(PasoZ))/UnitFactor;
```

```
        DeltaTiempo:=0.01*Distancia*UnitFactor/Feed;
        If FastSim
            Then
                Begin
                    X:=FinalX;Z:=FinalZ;
                    Trayect;
                    Tiempo:=Tiempo+DeltaTiempo;
                    OutPutFlag:= Cancel or OutPutFlag;
                    If OutputFlag Then Exit
                End
            Else
                Begin
                    Tracking;
                    While (Round(Z)<>FinalZ) or (Round(X)<>FinalX) do
                        Begin
                            X:=X+PasoX;
                            Z:=Z+PasoZ;
                            Tracking;
                            Tiempo:=Tiempo+DeltaTiempo;
                            OutPutFlag:= Cancel or OutPutFlag;
                            If OutputFlag Then Exit
                        End;
                End;
        Feed:=Fo;
        Xi:=Round(X);
        Zi:=Round(Z)
    End;

PROCEDURE Incremental;
    Begin
        Zf:=Zf+Zi*(2-Incr);
        Xf:=Xf*(3-Incr) + Xi*(2-Incr)
    End;

FUNCTION ValorComando(Var a,b,c,d:Integer):Boolean;
    Var
        j,k,l:Integer;
        VeriCommand:String;
        Valores:Array [3..6] of LongInt;
    Begin
        ValorComando:=True;
        VeriCommand:=Variable(N,2);
        Case VeriCommand[1] of
            'G':k:=0;
            'M':k:=100
        End;
        Val(Copy(VeriCommand,2,2),j,i);
        If i <> 0 Then
            Begin
                RunMessage('ERROR DE FORMATO '+VeriCommand,'','',1);
                ValorComando:=False;
                Exit
            End;
        If Comando[j+k,2] = ' ' Then
            Begin
                ValorComando:=False;
                RunMessage('COMANDO '+VeriCommand+' NO RECONOCIDO','','',1);
                Exit
            End;
        For l:=3 to 6 do
            If Comando[j+k,l] = ' '
                Then
                    Valores[l]:=0
                Else
                    Begin
                        VeriCommand:=Variable(N,l);
                        If Comando[j+k,l] <> '*' Then Delete(VeriCommand,1,1);
                        Val(VeriCommand,Valores[l],i);
                        If i <> 0 Then
                            Begin
                                RunMessage(  'ERROR NUMERICO '+Variable(N,2),
                                             'Param#'+StrNum(l,0,0)+Comando[j+k,l]+
```

```
                                        'Pos#'+StrNum(1,0,0),'',1);
                        ValorComando:=False;
                        Exit
                    End;
                If Comando[j+k,1] = '*'
                    Then
                        Case 1 of
                            3:Car:='X';
                            4:Car:='Z';
                            5:Car:='F';
                            6:Car:='H'
                        End
                    Else
                        Car:=UpCase(Comando[j+k,1]);
                If Not Limits(Variable(N,2),Car,Valores[1]) Then
                    Begin
                        ValorComando:=False;
                        RunMessage( 'Valor Fuera de Rango',
                                    'Param#'+StrNum(1,0,0),'',1);
                        Exit
                    End
            End;
    a:=Valores[3];
    b:=Valores[4];
    c:=Valores[5];
    d:=Valores[6];
End;
```

SIMULAC.PAS

```
PROCEDURE SimPant;
    Const
        Clm:Array [1..6] of Byte = (2,6,10,18,26,31);
    Var
        X1,Z1,Xf,Zf:Integer;
        Tool,Spindle,Husillo,Feed,FastFeed:Integer;
        X,Z,EscalaX,EscalaZ,Escala:Real;
        CentroX,CentroZ:Integer;
        x0,z0,x1,x2,z1,z2:Integer;
        xm1,xm2,zm1,zm2:Integer;
        StPg,StLn,St12,VPx1,VPy1,VPx2,VPy2:Integer;
        UnitFactor,Incr:Integer;
        FastSim,AvanceParMin,ContExec,OutPutFlag:Boolean;
        Comm:String;
        Value:Integer;
        i,j,k,l,Columna,InicioLinea:Integer;
        Tiempo:Real;
        TextInfo:TextSettingsType;

    {$I SIMPROC.PAS}

    {$I SIMCOMM.PAS}

    Begin
        If Linea[0].Command = 'M30' Then
            Begin
                Mensaje('No hay programa en memoria',1);
                Exit
            End;
        If (Dimensiones.Diametro<=0) or (Dimensiones.Largo<=0) Then
            Begin
                Mensaje('Las dimensiones del material no han sido definidas',1);
                Exit
            End;
        Menu[1]:='Ejecución continua';
        Menu[2]:='Ejecución por bloques';
        Menu[3]:='Simulacion rapida';
        Menu[4]:='Menú principal';
        Menu[5]:=FDM;
        i:=Entrada(26,10,Menu,Sencillo,Izq,1);
        If i = 4 Then Exit;
        ContExec:= (i = 1) or (i = 3);
        FastSim:= (i = 3);

{Limitación de lineas visibles}

        If Drv[102] < 3 Then Drv[102]:=3;
        If Drv[102] > 10 Then Drv[102]:=10;
        InitGraphics;
        SetTextStyle(DefaultFont,HorizDir,1);

{Definición de marco}

        VPx1:=0;
        VPy1:=Round(TextHeight('C')*Drv[102]);
        VPx2:=GetMaxX-TextWidth('0.0 in')-5;
        VPy2:=GetMaxY-TextHeight('X')-5;

{Definición de divisiones de las areas de trabajo}

        StPg:=TextWidth(' XXXXXXX.XXX')+2;
        StLn:=TextWidth(StrNum(100,Clm[6]+5,0))+StPg+4;
        SetTextStyle(DefaultFont,HorizDir,Drv[100]);
        St12:=TextWidth(' X-37565 Z-37565 ');

{Determinación de origen y escala del dibujo}

        CentroX:=(VPx2-VPy1) div 2;
        CentroZ:=Drv[101];
```

```
        EscalaX:=((VPy2-VPy1)*0.8/RelY)/(Dimensiones.Diametro);
        EscalaZ:=((VPx2-VPx1)*0.8/RelX)/(Dimensiones.largo);
        If EscaleX<EscalaZ
            Then
                Escala:=EscalaX
            Else
                Escala:=EscalaZ;

{Definición y muestra de estatus inicial}

        FullPort;
        SetColor(Drv[90]);
        Rectangle(VPx1,VPy1,VPx2,VPy2);
        Line(StPg,0,StPg,VPy1);
        Line(StLn,0,StLn,VPy1);
        SetColor(Drv[92]);
        DrawText(2,1,'--PROGRAMA--',1);
        DrawText(2,2,Programa+'.CNC',1);
        If Unidades = 1
            Then
                Begin
                    UnitFactor:=100;
                    FastFeed:=10000;
                    DrawText(2,3,'MILIMETROS',1);
                    St:='0 mm';
                End
            Else
                Begin
                    UnitFactor:=1000;
                    FastFeed:=5000;
                    DrawText(2,3,'PULGADAS',1);
                    St:='0.0 in';
                End;

{Dibujo del material en bruto}

        WorkPort;
        z1:=CentroZ Div 2;
        x1:=Round(CentroX+Dimensiones.Diametro/2*RelY*Escala);
        z2:=Round(CentroZ+Dimensiones.Largo*RelX*Escala);
        x2:=Round(CentroX-Dimensiones.Diametro/2*RelY*Escala);
        If FastSim
        Then
                Begin
            SetColor(Drv[97]);
            Line(z1,x1,z2,x1);
                    Line(z2,x1,z2,x2);
                    Line(z2,x2,z1,x2);
                    Line(z1,x2,z1,x1)
                End
        Else
            Begin
            SetFillStyle(Drv[99],Drv[97]);
            Bar(z1,x1,z2,x2)
            End;
        X0:=CentroX;
        Z0:=z2;

{Dibujo del chuck}

        SetColor(Drv[95]);
        MoveTo(CentroZ,X2);
        LineRel(0,-Round(15*Escala*RelY*(100/UnitFactor)));
        LineRel(-Round(7*Escala*RelX*(100/UnitFactor)),0);
        LineRel(0,-Round(15*Escala*RelY*(100/UnitFactor)));
        LineRel(-Round(5*Escala*RelX*(100/UnitFactor)),0);
        LineRel(0,-Round(15*Escala*RelY*(100/UnitFactor)));
        LineTo(0,GetY);
        Line(0,x2,z1,x2);
        MoveTo(CentroZ,x1);
        LineRel(0,Round(15*Escala*RelY*(100/UnitFactor)));
        LineRel(-Round(7*Escala*RelX*(100/UnitFactor)),0);
```

```
        LineRel(0,Round(15*Escala*RelY*(100/UnitFactor)));
        LineRel(-Round(5*Escala*RelX*(100/UnitFactor)),0);
        LineRel(0,Round(15*Escala*RelY*(100/UnitFactor)));
        LineTo(0,GotY);
        Line(0,x1,z1,x1);

{Dibujo de ejes}

        Ejes;
        i:=0;
        GetTextSettings(TextInfo);
        SetViewPort(VPx2+5,VPy1+1,GetMaxX,VPy2,ClipOn);
        SetTextJustify(LeftText,CenterText);
        OutTextXY(0,XO,St);
        j:=Round(XO/(Escala*RelY*100/UnitFactor));
        Repeat
            i:=i+5;
            x:=i*Escala*RelY*100/UnitFactor;
            OutTextXY(0,XO+Round(x),StrNum(i*100/UnitFactor,0,UnitFactor div 1000));
            OutTextXY(0,XO-Round(x),StrNum(i*100/UnitFactor,0,UnitFactor div 1000));
        Until i + 6 > j;
        WorkPort;
        With TextInfo do
            Begin
                SetTextJustify(Horiz,Vert);
                SetTextStyle(Font,Direction,CharSize)
            End;

{Preparación de parámetros para simulación}

        Tool:=1;
        N:=-1;
        Incr:=1;
        Husillo:=2500;
        Spindle:=0;
        z1:=GetMaxX;
        x1:=x0;
        SubNivel:=0;
        X:=Round(Dimensiones.diametro*UnitFactor);
        Z:=0;

{Inicia programa}

        InicioLinea:=30;
        Car:=#13;
        If ContExec Then RunMessage( 'OPRIME CUALQUIER TECLA',
                                     'PARA INICIAR SIMULACION',
                                     '[Esc] Cancelar',0);
        If Car = #27 Then
            Begin
                InitText;
                Exit
            End;
        OutPutFlag:=False;
        AvancePorHin:=True;
        Tiempo:=0;
        Funciones;
        Repeat
            N:=N+1; {Contador programa CNC}
            If Not ContExec Then
                Repeat
                    Funciones;
                    DrawText(1,3,'[ENTER] EJECUTA BLOQUE No.'+StrNum(N,0,0),1);
                    Car:=Opciones;
                    If Car = 'A' Then
                        Begin
                            InitText;
                            Exit
                        End
                Until Car = #13;
            LinePort;
            ClearViewPort;
```

```
        SetColor(Drv[93]);
        DrawText(1,1,'N',1);
        For j:=1 to Drv[102] do
            Begin
                For l:=1 to 6 do DrawText(Clm[l],j,Variable(N+j-1,l),1);
                SetColor(Drv[94])
            End;
        Comm:=Variable(N,2);
        Val(Copy(Comm,2,2),Value,i);
        if i<>0 Then
            Begin
                Beep;
                Halt
            End;
        If Comm[1] = 'M'
            Then
                Case Value of
                    00:M00;
                    03:M03;
                    04:M04;
                    05:M05;
                    06:M06;
                    17:M17;
                    30:M30;
                    98:M98;
                    Else RunMessage('COMANDO '+Comm+' NO RECONOCIDO','','',0);
                End {Case}
            Else
                Case Value of
                    00:G00;
                    01:G01;
                    02:G02;
                    03:G03;
                    04:G04;
                    25:G25;
                    27:G27;
                    33:G33;
                    70:G70;
                    71:G71;
                    73:G73;
                    78:G78;
                    81:G81;
                    82:G82;
                    83:G83;
                    84:G84;
                    85:G85;
                    86:G86;
                    88:G88;
                    89:G89;
                    90:G90;
                    91:G91;
                    92:G92;
                    94:G94;
                    95:G95;
                    Else RunMessage('COMANDO '+Comm+' NO RECONOCIDO','','',0);
                End; {Case}
        Ejes
    Until OutPutFlag or (Linea[N].Command = 'M30');
    MessagePort;
    ClearViewPort;
    SetColor(Drv[92]);
    DrawText(1,1,'CALCULO DE TIEMPO APROXIMADO:',1);
    DrawText(1,2,StrNum(Tiempo,0,2)+' '+'minutos',1);
    DrawText(1,3,'Oprimir cualquier tecla',1);
    Car:=ReadKey;
    InitText
End; {SimPant}
```

TRANSMIT.PAS

```
VAR
    StatPort:Byte;
    TransError:Boolean;
    Regs:Registers;

FUNCTION Bits(Siete,Seis,Cinco,Cuatro,Tres,Dos,Uno,Cero:Byte):Byte;
    Begin
        Bits:=  Cero + Uno shl 1 + Dos shl Dos + Tres shl Tres + Cuatro shl Cuatro +
                Cinco shl 5 + Seis shl 6 + Siete shl 7;
    End;

PROCEDURE InitCom(NumPort,InitParam:Byte);
    Begin
        With Regs do
            Begin
                AH:=$00;
                AL:=InitParam;
                DX:=Pred(NumPort);
                Intr($14,Regs);
                StatPort:=AH;
            End
    End;

PROCEDURE WriteCarCOM(NumPort:Byte;Car:Char);
    Begin
        With Regs do
            Begin
                AH:=$01;
                AL:=Ord(Car);
                DX:=Pred(NumPort);
                Intr($14,Regs);
                StatPort:=AH
            End
    End;

FUNCTION COMError:Boolean;
    Begin
        COMError:=False;
        If KeyPressed then
            begin
                pausa;
                pausa
            end;
        If (StatPort and $80) = $80
            Then
                Begin
                    COMError:=True;
                    TransError:=True;
                    Beep;
                    St:='';
                    For i:=0 to 7 do
                        Begin
                            If StatPort Mod 2 = 0
                                Then
                                    St:='0 '+St
                                Else
                                    St:='1 '+St;
                            StatPort:= StatPort Div 2
                        End;
                    Mensaje('Error de transmision : '+St,1)
                End
            Else
                Mensaje('Transmitiendo...',0);
    End;

PROCEDURE CompactSCNC;
    Const
        SP=#32;
        AP=#96;
```

```
        CR=#13;
        LF=#10;
        VC=#0;
    Var
        IOPort:Byte;
        Comando,Campo:String;
        Q,Valor:Integer;
        Bloque:String[33];
    Procedure ClearBuffer;
        Begin
            Bloque:='################################';
        End;

    Procedure SERIAL;
        Var
            ChPos:Integer;
        Begin
        IOPort:=Drv[30];
        ChPos:=Pos('#',Bloque);
            Delete(Bloque,ChPos,34-ChPos);
            For i:=1 to Length(Bloque) do
                Begin
                    ColorText(Drv[14],Drv[8]);
                    Write(Bloque[i]);
                    WriteCarCOM(IOPort,Bloque[i]);
                    If COMError Then Exit;
                End;
            ClearBuffer
        End;

    Begin
        If Linea[0].Command = 'M30' Then
            Begin
                Beep;
                Mensaje('No hay programa en memoria...',1);
                Exit;
            End;
        TransError:=False;
        Cuadro(22,7,59,15,2);
        ColorText(Drv[14],Drv[8]);
        Writeln;
        Writeln(' Ejecuta el comando G66 + INP + INP');
        Writeln;
        Writeln(' en la máquina COMPACT 5 CNC antes');
        Writeln;
        Writeln(' de iniciar la transmisión.');
        Mensaje('Oprime cualquier tecla para iniciar transmision...',0);
        Car:=ReadKey;
        If Car=#27 Then Exit;
        InitCOM(IOPort,Bits(0,1,0,1,0,1,1,0));
        ColorText(Drv[14],Drv[8]);
        ClrScr;
        Writeln(StatPort);
        If ComError Then Exit;
{0}     ClearBuffer;
        Bloque[1]:='%';
        Bloque[2]:=CR;
        Bloque[3]:=LF;
        SERIAL;
        If TransError then exit;
        For Q:=1 to 4 do Bloque[Q]:=SP;
        Bloque[5]:='N';
        Bloque[6]:=AP;
        Bloque[7]:=SP;
        Bloque[8]:='G';
        Bloque[9]:=AP;
        For Q:=10 to 12 do Bloque[Q]:=SP;
        Bloque[13]:='X';
        Bloque[14]:=SP;
        Bloque[15]:=AP;
        For Q:=16 to 19 do Bloque[Q]:=SP;
        Bloque[20]:='Z';
```

```
        Bloque[21]:=SP;
        Bloque[22]:=AP;
        Bloque[23]:=SP;
        Bloque[24]:=SP;
        Bloque[25]:='F';
        Bloque[26]:=AP;
        Bloque[27]:=SP;
        Bloque[28]:=SP;
        Bloque[29]:='H';
        Bloque[30]:=SP;
        Bloque[31]:=CR;
        Bloque[32]:=LF;
        SERIAL;
        If TransError then Exit;
        N:=-1;
        Repeat
            N:=N+1;
            Q:=0;
            Bloque[1]:=SP;
            Bloque[2]:=SP;
            Bloque[3]:=SP;
{1}         St:=StrNum(N+1000,0,0);
            Bloque[4]:=St[2];
            Bloque[5]:=St[3];
            Bloque[6]:=St[4];
{2}         Comando:=Variable(N,2);
            If Comando[1] = 'M'
                then
                    Bloque[7]:='M'
                else
                    Bloque[7]:=SP;
            If (Comando = 'G71') or (Comando = 'G70') Then Comando:='G21';
            Bloque[8]:=Comando[2];
            Bloque[9]:=Comando[3];
            Bloque[10]:=SP;
{3}         Campo:=Variable(N,3);
            If Campo = ''
                Then
                    For i:=11 to 15 do Bloque[i]:=SP
                Else
                    Begin
                        If Comando = 'M99'
                            Then
                                Begin
                                    Bloque[11]:=Campo[1];
                                    Delete(Campo,1,1);
                                    Val(Campo,Valor,i)
                                End
                            Else
                                Begin
                                    Val(Campo,Valor,i);
                                    If Valor < 0
                                        Then
                                            Bloque[11]:='-'
                                        Else
                                            Bloque[11]:=SP;
                                End;
                        St:=StrNum(10000+Abs(Valor),0,0);
                        i:=1;
                        Repeat
                            i:=i+1;
                            Bloque[i+10]:=SP
                        Until (St[i] <> '0') or (i > 4);
                        For Q:=1 to 4 do Bloque[Q+10]:=St[Q];
                        Bloque[15]:=St[5]
                    End;
            Bloque[16]:=SP;
{4}         Campo:=Variable(N,4);
            If Campo = ''
                Then
                    For i:=17 to 22 do Bloque[i]:=SP
                Else
```

```
                Begin
                    If Comando = 'M99'
                        Then
                            Begin
                                Bloque[17]:=Campo[1];
                                Delete(Campo,1,1);
                                Val(Campo,Valor,i)
                            End
                        Else
                            Begin
                                Val(Campo,Valor,i);
                                If Valor < 0
                                    Then
                                        Bloque[17]:='-'
                                    Else
                                        Bloque[17]:=SP;
                            End;
                    St:=StrNum(100000+Abs(Valor),0,0);
                    i:=1;
                    Repeat
                        i:=i+1;
                        Bloque[i+16]:=SP
                    Until (St[i] <> '0') or (i > 5);
                    For Q:=i to 5 do Bloque[Q+16]:=St[Q];
                    Bloque[22]:=St[6]
                End;
{5}         Campo:=Variable(N,5);
            If Comando = 'M06' Then Campo:='T1';
            If Campo = ''
                Then
                    For Q:=23 to 26 do Bloque[Q]:=SP
                Else
                    Begin
                        If Campo[1] in ['A'..'Z']
                            Then
                                Begin
                                    Bloque[23]:=Campo[1];
                                    Delete(Campo,1,1)
                                End
                            Else
                                Bloque[23]:=SP;
                        Val(Campo,Valor,i);
                        St:=StrNum(1000+Valor,0,0);
                        i:=1;
                        Repeat
                            i:=i+1;
                            Bloque[i+22]:=SP
                        Until (St[i] <> '0') or (i > 3);
                        For Q:=i to 3 do Bloque[Q+22]:=St[Q];
                        Bloque[26]:=St[4]
                    End;
            Bloque[27]:=SP;
{6}         Campo:=Variable(N,6);
            If Comando = 'M06' Then Campo:='';
            If Campo = ''
                Then
                    For Q:=28 to 30 do Bloque[Q]:=SP
                Else
                    Begin
                        Val(Campo,Valor,i);
                        St:=StrNum(1000+Valor,0,0);
                        i:=1;
                        Repeat
                            i:=i+1;
                            Bloque[i+26]:=SP
                        Until (St[i] <> '0') or (i > 3);
                        For Q:=i to 3 do Bloque[Q+26]:=St[Q];
                        Bloque[30]:=St[4]
                    End;
            Bloque[31]:=CR;
            Bloque[32]:=LF;
            SERIAL;
```

```
        If TransError then exit;
    Until Comando = 'H30';
    For Q:=1 to 3 do Bloque[Q]:=SP;
    If Unidades = 2
        Then Bloque[4]:='"'
        Else Bloque[4]:='H';
    SERIAL;
    Mensaje('Transmisión finalizada',1);
End;
```

TURNUP.PAS

```
PROGRAM TurnUP; {Control y simulación de programas CNC para tornos EMCO}

USES    Graph,Crt,Dos,Printer,
        Utileria,UtilCNC,UtilCRT,
        Definit,Editor,Gestion;

{$I PREPARA.PAS}

{$I SIMULAC.PAS}

{$I TRANSMIT.PAS}

PROCEDURE MenuPrincipal;
    VAR
        Salida:Boolean;
        Op:Byte;
    Begin
        Op:=1;
        Repeat
            Modulo:='Menú principal';
            Pantalla;
            Menu[1]:='Cargar Programa CNC de Disco';
            Menu[2]:='Guardar Programa CNC en Disco';
            Menu[3]:='Directorio de programas CNC';
            Menu[4]:='Editar Programa CNC';
            Menu[5]:='Simulación en pantalla';
            Menu[6]:='Transmisión a COMPACT 5 CNC';
            Menu[7]:='Fin de Programa';
            Menu[8]:=FDM;
            Op:=Entrada(22,7,Menu,Sencillo,Izq,Op);
            Modulo:=Menu[Op];
            Pantalla;
            Case Op of
                1:If Linea[0].Command <> 'M30'
                        Then
                            Begin
                                Mensaje('El programa en memoria será borrado. ¿Continuar? (S/N)',0);
                                Repeat Car:=UpCase(Readkey) Until Car in ['S','N'];
                                Mensaje('',0);
                                If Car = 'S' Then LoadProgram
                            End
                        Else
                            LoadProgram;
                2:SaveProgram;
                3:Directorio;
                4:EditProgram;
                5:SimPant;
                6:Compact5CNC;
                7:Begin
                        If Linea[0].Command <> 'M30' Then
                            Begin
                                Mensaje('¿Grabar programa antes de terminar? (S/N)',0);
                                Repeat Car:=UpCase(Readkey) Until Car in ['S','N'];
                                Mensaje('',0);
                                If Car = 'S' Then SaveProgram;
                            End;
                        Mensaje('¿Salir a DOS? (S/N)',0);
                        Repeat Car:=UpCase(Readkey) Until Car in ['S','N'];
                        If Car = 'N' Then Op:=1
                    End
            End; {case}
            Until Op=7;
    End; {MenuPrincipal}

BEGIN
    Prepara;
    ColorText(15,Drv[1]);
    ClrScr;
    MenuPrincipal;
```

```
    ExpandWindow;
    ClrScr;
    NormVideo;
    Writeln('Fin de programa ',Version);
    Delay(500);
END.
```

UTILCNC.PAS

```
UNIT UTILCNC;
INTERFACE
Uses Utileria;

PROCEDURE InitProgramCNC;
FUNCTION Variable(Renglon,Columna:Integer):String;
PROCEDURE Asigna(Renglon,Columna:Integer;Value:String);

IMPLEMENTATION

PROCEDURE InitProgramCNC;
    Begin
        For N:=0 to MaxLineas do
            With Linea[N] do
                Begin
                    Command:='';
                    CoordX:='';CoordZ:='';
                    Velocidad:='';ProfCorte:='';
                End; {With}
        Linea[0].Command:='M30';
        Programador:='';
        Descripcion:='';
        Dimensiones.Diametro:=0;Dimensiones.Largo:=0;
        For i:=1 to MaxHerram do Ordherram[i]:=i;
        N:=0;
    End;

FUNCTION Variable(Renglon,Columna:Integer):String;
    Begin
        With Linea[Renglon] do
            Case Columna of
                1:Variable:=Copy(StrNum(Renglon+1000,4,0),2,3);          { N }
                2:Variable:=Command;                                          { G }
                3:Variable:=CoordX;                                      { X }
                4:Variable:=CoordZ;                                      { Z }
                5:Variable:=Velocidad;                                   { F }
                6:Variable:=ProfCorte                                    { H }
            End {Case}
    End;

PROCEDURE Asigna(Renglon,Columna:Integer;Value:String);
    Begin
        With Linea[Renglon] do
            Case Columna of
                2:Command:=Value;
                3:CoordX:=Value;
                4:CoordZ:=Value;
                5:Velocidad:=Value;
                6:ProfCorte:=Value
            End {Case}
    End;
END.
```

UTILCRT.PAS

```
UNIT UTILCRT;

INTERFACE

Uses CRT,Utileria;

CONST
    MaxMenu=25; {Numero máximo de opciones en los menús}
    FDM='Fin de Menú';
    Sencillo=1;
    Doble=2;
    Triple=3;

TYPE
    TipoMenu=Array [1..MaxMenu] of String[80];
    Align=(Izq,Der,Centro);

VAR
    Menu:TipoMenu;

PROCEDURE Pantalla; {Presenta la linea de estatus superior}
PROCEDURE Mensaje(s:String;Espera:Byte);
PROCEDURE Cuadro(x1,y1,x2,y2,tipo:integer);
FUNCTION Entrada(x,y:Byte;Letrero:TipoMenu;Espaciado:Byte;Alineacion:Align;Opcion:byte):Byte;
FUNCTION LeeCampo(Px,Py:Byte;Default:String80;Longitud:Byte):String80;
PROCEDURE ReadReal(Var x:Real;LongCampo:Byte);
PROCEDURE ReadInt(Var i:Integer;Min,Max:Integer);

IMPLEMENTATION

PROCEDURE Pantalla; {Presenta la linea de estatus superior}
    Const
    Izquierda = Version;
    Derecha = 'MR3';
    Begin
        Window(1,1,80,1);
        ColorText(Drv[2],Drv[3]);
        ClrScr;
        GotoXY(2,1);Write(Izquierda);
        GotoXY(40-(Length(Modulo) div 2),1);Write(Modulo);
        GotoXY(79-Length(Derecha),1);Write(Derecha);
        NormVideo;
        Mensaje('',0);
        Window(1,2,80,24);
        ColorText(15,Drv[1]);
        ClrScr;
    End;

PROCEDURE Mensaje(s:String;Espera:Byte);
    VAR
        OldX,OldY,x1,y1,x2,y2:Byte;
    Begin
        x1:=Lo(WindMin)+1;
        y1:=Hi(WindMin)+1;
        x2:=Lo(WindMax)+1;
        y2:=Hi(WindMax)+1;
        OldX:=WhereX;OldY:=WhereY;
        Window(1,25,80,25);
        ColorText(Drv[4]+Blink,Drv[5]);
        ClrScr;
        GotoXY(Trunc(40-Length(s)/2),1);
        Write(s);
        If Espera=1 then
            Begin
            Beep;
                Pause;
                ClrScr
            End;
        NormVideo;
```

```
        Window(x1,y1,x2,y2);
        GotoXY(OldX,OldY)
    End; {Mensaje}

PROCEDURE Cuadro(x1,y1,x2,y2,tipo:integer);
    var
        s,Lx,Ly:integer;
        esd,esi,eid,eii,vert,hor:char;
    begin
        if not (tipo in [1..4]) then tipo:=1;
        case tipo of
            1:begin
                    esi:='Ú';esd:='¿';
                    eii:='À';eid:='Ù';
                    vert:='³';hor:='Ä'
                end;
            2:begin
                    esi:='É';esd:='»';
                    eii:='È';eid:='¼';
                    vert:='º';hor:='Í'
                end;
            3:begin
                    esi:='Õ';esd:='¸';
                    eii:='Ô';eid:='¾';
                    vert:='³';hor:='Í';
                end;
            4:begin
                    esi:='Ö';esd:='·';
                    eii:='Ó';eid:='½';
                    vert:='º';hor:='Ä'
                end
        end; {case}
        ExpandWindow;
        Lx:=x2-x1;
        Ly:=y2-y1;
        ColorText(Drv[6],Drv[7]);
        Window(x1,y1,x2,y2+1);
        gotoxy(1,1);write(esi);
        for s:=2 to Lx do write(hor);
        write(esd);
        for s:=2 to Ly do
            begin
                gotoxy(1,s);write(vert);
                gotoxy(Lx+1,s);write(vert)
            end;
        gotoxy(1,Ly+1);write(eii);
        for s:=2 to Lx do write(hor);
        GotoXY(Lx+1,Ly+1);write(eid);
        ColorText(Drv[5],Drv[8]);
        ClearWindow(x1+1,y1+1,x2-1,y2-1);
        NormVideo;
    end;

FUNCTION Entrada(x,y:Byte;Letrero:TipoMenu;Espaciado:Byte;Alineacion:Align;Opcion:byte):Byte;
    VAR
        MaxX,MaxY,n:Byte;
        Anterior,i,j,l:Byte;
        s:String80;
        Sx,Sy:Array [1..MaxMenu] of byte;
        Car1,Car2:Char;
        Caracteres:Set of Char;
    Procedure Enciende(k:Byte);
        Begin
            GotoXY(Sx[k],Sy[k]);
            ColorText(Drv[10],Drv[11]);
            Write(Letrero[k]);
            NormVideo;
        End; {Enciende}
    Procedure Apaga(k:Byte);
        Begin
            ColorText(Drv[9],Drv[8]);
            GotoXY(Sx[k],Sy[k]);
```

```
            Write(Letrero[k]);
            NormVideo
        End;
Begin
    n:=0;
    MaxX:=0;
    Caracteres:=[#13,'H','P','M','K'];
    While Letrero[n+1]<>FDM do
        Begin
            n:=n+1;
            l:=Length(Letrero[n]);
            If l>MaxX then MaxX:=l;
            Caracteres:=Caracteres+[CHR(64+n)]
        End; {FDM}
    MaxY:=(n+1)*Espaciado;
    If Alineacion = Izq
        Then
            Begin
                MaxX:=MaxX+5;
                Cuadro(x,y,x+MaxX+1,y+MaxY,2);
                For i:=1 to n do
                    Begin
                        Sx[i]:=2;
                        Sy[i]:=i*Espaciado;
                        Letrero[i]:=CHR(64+i)+'. '+Letrero[i];
                        Apaga(i)
                    End;
            End {Then}
        Else
            Begin
                MaxX:=MaxX+2;
                Cuadro(x,y,x+MaxX+1,y+MaxY,2);
                For i:=1 to n do
                    Begin
                        Sx[i]:=1+Trunc((MaxX-Length(Letrero[i]))/2);
                        Sy[i]:=i*Espaciado;
                        Apaga(i)
                    End
            End; {Else}
        Enciende(Opcion);
        Repeat
            Anterior:=Opcion;
            Repeat
                Car1:=UpCase(ReadKey);
                If Car1=#0
                    Then
                        Car2:=UpCase(ReadKey)
                    Else
                        Car2:=Car1;
            Until Car2 in Caracteres;
            If Car1=#0
                Then
                    Case Car2 of
                        'H','K':{Arriba}
                                    Begin
                                        Opcion:=Opcion-1;
                                        If Opcion<1 Then Opcion:=n
                                    End;
                        'P','M':{Abajo}
                                    Begin
                                        Opcion:=Opcion+1;
                                        If Opcion>n Then Opcion:=1
                                    End;
                    End {Case}
                Else
                    If Car2<>#13 Then
                        If ORD(Car2)-64 > n
                            Then
                                Beep
                            Else
                                Opcion:=ORD(Car2)-64;
            Apaga(Anterior);
```

```
                Enciende(Opcion);
                If (Ord(Car2) in [65..64+n]) And (Car1<>#0) Then
                    Begin
                        Delay(100);
                        Car2:=#13
                    End
            Until Car2 = #13;
            NormVideo;
            Entrada:=Opcion
    End; {Entrada}

FUNCTION LeeCampo(Px,Py:Byte;Default:String80;Longitud:Byte):String80;
    CONST
        ValidCar=[#0,#8,#32..#127];
    VAR
        C:Char;
        Lectura:String80;
        Loc:Byte;
    Begin
        GotoXY(Px,Py);
        ColorText(Drv[12],Drv[13]);
        Write(Default,'':Longitud-Length(Default));
        Lectura:=Default;
        GotoXY(Px,Py);
        C:=ReadKey;
        If C <> #13 Then
            Begin
                Lectura:='';
                Loc:=1;
                Repeat
                    If C in ValidCar Then
                        Case C of
                            #0:Begin
                                    C:=ReadKey;
                                    Case C of
                                        'K':Begin
                                                Loc:=Loc-1;
                                                If Loc<1 Then Loc:=1
                                            End;
                                        'M':Begin
                                                Loc:=Loc+1;
                                                If Loc>Longitud Then Loc:=Longitud
                                            End;
                                        'S':Delete(Lectura,Loc,1)
                                    End
                                End; {#0}
                            #8:Begin
                                    Loc:=Loc-1;
                                    If Loc>0
                                        Then
                                            Delete(Lectura,Loc,1)
                                        Else
                                            Loc:=1
                                End; {#8}
                            Else
                                Begin
                                    Insert(C,Lectura,Loc);
                                    Loc:=Loc+1;
                                    If Loc>Longitud Then Loc:=Longitud
                                End
                        End; {Case}
                    GotoXY(Px,Py);
                    Lectura:=Copy(Lectura,1,Longitud);
                    Write(Lectura,'':Longitud-Length(Lectura));
                    GotoXY(Px+Loc-1,Py);
                    C:=ReadKey
                Until C = #13
            End;
        LeeCampo:=Lectura;
        NormVideo
    End; {LeeCampo}
```

```
PROCEDURE ReadReal(Var x:Real;LongCampo:Byte);
    VAR
        xx,yy,i:Integer;
        s:String[20];
        anterior:Real;
    Begin
        Anterior:=x;
        s:=Copy(StrNum(anterior,0,20),1,LongCampo);
        xx:=WhereX;
        yy:=WhereY;
        Repeat
            s:=LeeCampo(xx,yy,s,LongCampo);
            Val(s,x,i);
            If i<>0 then Beep
        Until i=0;
        If s='' Then x:=Anterior
    End; {ReadReal}

PROCEDURE ReadInt(Var i:Integer;Min,Max:Integer);
    VAR
        s:String;
        k,xx,yy:Integer;
        b:Boolean;
        LongCampo,L1,L2:Byte;
    begin
        xx:=WhereX;
        yy:=WhereY;
        L1:=Length(StrNum(Min,0,0));
        L2:=Length(StrNum(Max,0,0));
        If L1 > L2
            Then
                LongCampo:=L1
            Else
                LongCampo:=L2;
        b:=False;
        s:=StrNum(i,0,0);
        Repeat
            s:=LeeCampo(xx,yy,s,LongCampo);
            Val(s,i,k);
            b:= (i >= Min) And (i <= Max) And (k = 0);
            If Not b Then Beep
        Until b
    End;

END.
```

UTILERIA.PAS

```
UNIT UTILERIA;

INTERFACE

Uses GRAPH,CRT,DOS;

CONST
    UtilDir='';                                 {Ruta de busqueda para utilerias del programa}
    GraphDir='';                                {Ruta de busqueda para archivos *.bgi}
    ProgDir:Char = 'C';                         {Ruta inicial de busqueda para programas CNC}
    Version='TurnUP v1.2';                      {Nombre del programa}
    MaxLineas=299;                              {Lineas de comando}
    MaxDrv=110;                                 {Parametros de Driver de graficos}
    MaxNiveles=5;                               {Nivel máximo de anidación del programa CNC}
    MaxHerram=8;                                {Número máximo de herramientas}
    BrocaMin=1;

TYPE
    TipoSistema=Array [0..2] of String;
    TipoHerramienta=Array [1..MaxHerram] of String[20];
    TipoOrdenHerram=Array [1..MaxHerram] Of byte;

CONST
    Sistemas:TipoSistema = (' ','Sistema Internacional','Sistema Inglés');
    Herramientas:TipoHerramienta = (  'DERECHA',
                                      'IZQUIERDA',
                                      'NEUTRA',
                                      'TRONZADORA',
                                      'ROSCADORA',
                                      'BROCA',
                                      'CORTADOR INTERNO',
                                      'ROSCA INTERNA');

TYPE
    String3=String[3];
    String4=String[4];
    String8=String[8];
    String30=String[30];
    String80=String[80];
    TipoLinea=RECORD
                Command:String3;
                CoordX,CoordZ:String8;
                Velocidad:String4;
                ProfCorte:String4;
              END;
    TipoUnidad=0..2; {0. No definido, 1. Sistema Internacional, 2. Sistema Ingles}

VAR

{Variables del Programa CNC}

    Linea:array [0..MaxLineas] of TipoLinea;
    Descripcion:String[25];
    Programador:String30;
    Dimensiones:Record
                  Diametro,Largo:Real;
                End;
    Material:String[20];
    Fecha:String[10];
    OrdHerram:TipoOrdenHerram;
    Unidades:TipoUnidad;

{Variables de control de programa}

    Comando:Array [0..199,2..6] of Char;
    N:Integer;
    SubLinea:Array [1..MaxNiveles] of Integer;
    SubNivel:Integer;
```

```
{Variables de manejo de archivo}

    Programa:String30;
    Lach:Text;
    Drive:String[1];
    IOStat:Byte;

{Variables internas}

    Renglon:Byte;
    I:integer;
    Car:Char;
    St:String;
    Flag:boolean;
    TotalCommands:Byte;
    Modulo:String30;

{Variables de control de gráficos}

    DriverName:String;
    Driver,Mode:Integer;
    Drv:Array [1..MaxDrv] of Integer;
    DiametroBroca:Real;
    RelX,RelY:Real;

FUNCTION Log(x:Real):Real;
PROCEDURE Beep;
PROCEDURE Pausa;
FUNCTION StrNum(x:Real;Campo,Decimales:Byte):String;
FUNCTION Right(s:String;i:Byte):String;
FUNCTION UpString(s:String):String;
FUNCTION BorraEspacios(s:String):String;
PROCEDURE Inverse;
PROCEDURE ColorText(Letras,Fondo:Byte);
PROCEDURE ExpandWindow;
PROCEDURE ClearWindow(x1,y1,x2,y2:byte);
FUNCTION GetIOError(Error:Byte):String;
FUNCTION PrinterOnLine:Boolean;
PROCEDURE InitText;
PROCEDURE InitGraphics;
PROCEDURE DrawText(X,Y:Byte;Texto:String;Size:Byte);
PROCEDURE FullPort;

{Utilerias utilizadas por TORNO.PAS}

IMPLEMENTATION

FUNCTION Log(x:Real):Real;
    Begin
        Log:=Ln(x)/Ln(10)
    End;

PROCEDURE Beep;
    Begin
        write(#7)
    End;

PROCEDURE Pausa;
    VAR
        c:Char;
    Begin
        c:=ReadKey
    End;

FUNCTION StrNum(x:Real;Campo,Decimales:Byte):String;
    VAR
        s:string;
    Begin
        Str(x:Campo:Decimales,s);
        StrNum:=s
    End;
```

```
FUNCTION Right(s:String;i:Byte):String;
    Begin
        Right:=Copy(s,Length(s)-(i+1),i);
    End; {Right}

FUNCTION UpString(s:String):String;
    Begin
        For i:=1 to Length(s) do
            s[i]:=UpCase(s[i]);
        UpString:=s
    End; {UpString}

FUNCTION BorraEspacios(s:String):String;
    Begin
        i:=0;
        While i<Length(s) do
            Begin
                i:=i+1;
                If s[i] = ' ' then
                    Begin
                        Delete(s,i,1);
                        i:=i-1;
                    End
            End;
        BorraEspacios:=s
    End; {BorraEspacios}

PROCEDURE Inverse;
    Begin
        TextColor(Black);
        TextBackground(White);
    End;

PROCEDURE ColorText(Letras,Fondo:Byte);
    Begin
        TextColor(Letras);
        TextBackground(Fondo)
    End;


PROCEDURE ExpandWindow;
    Begin
        Window(1,1,80,25)
    End;

PROCEDURE ClearWindow(x1,y1,x2,y2:byte);
    Begin
        Window(x1,y1,x2,y2);
        ClrScr
    End;

FUNCTION GetIOError(Error:Byte):String;
    Begin
        Case Error Of
            0:GetIOError:='Funcionamiento Normal';
            1:GetIOError:='';
            2:GetIOError:='Archivo no encontrado';
            3:GetIOError:='Ruta no encontrada';
            4:GetIOError:='Demasiados archivos abiertos';
            5:GetIOError:='No hay acceso al archivo';
            8:GetIOError:='No hay memoria suficiente';
            100:GetIOError:='Error en lectura de disco';
            101:GetIOError:='Error en escritura de disco';
            102:GetIOError:='Archivo no asignado';
            103:GetIOError:='Archivo no abierto';
            104:GetIOError:='Archivo no abierto para entrada';
            105:GetIOError:='Archivo no abierto para salida';
            150:GetIOError:='Disco protegido contra escritura';
            152:GetIOError:='El controlador de discos no está listo';
            154:GetIOError:='CRC Dañado';
            156:GetIOError:='Error en busqueda de disco';
            158:GetIOError:='Sector no encontrado';
```

```
            159:GetIOError:='La impresora no está lista';
            160:GetIOError:='Falla en escritura de terminal';
            161:GetIOError:='Falla en lectura de terminal';
            162:GetIOError:='Falla generalizada';
            Else GetIOError:='Error de ejecución'
        End {Case}
    End;

FUNCTION PrinterOnLine:Boolean;
    Const
        PrnStatusInt:Byte = $17;
        StatusRequest:Byte = $02;
        PrinterNum:Word = 0; {0 LPT1, 1 LPT2, etc}
    Var
        Regs:Registers;
    Begin
        Regs.AH:=StatusRequest;
        Regs.DX:=PrinterNum;
        Intr(PrnStatusInt,Regs);
        PrinterOnLine:= (Regs.AH and $80) = $80;
    End;

PROCEDURE InitText;
    Begin
        RestoreCRTMode;
        ExpandWindow;
        ColorText(15,Drv[1]);
        ClrScr
    End;

PROCEDURE InitGraphics;
    Var
        Resultado:Integer;
    Begin
        InitGraph(Driver,Mode,GraphDir);
        Resultado:=GraphResult;
        If Resultado <> grOk Then
            Begin
                RestoreCRTMode;
                Writeln;
                Writeln(Resultado,' Error de inicialización de gráficos');
                Halt;
            End;
        RelX:=Round((GetMaxX+1)/320);
        RelY:=Round((GetMaxY+1)/200);
    End;

PROCEDURE DrawText(X,Y:Byte;Texto:String;Size:Byte);
    VAR
        IncrX,IncrY:Byte;
    Begin
        SetTextStyle(DefaultFont,HorizDir,Size);
        IncrX:=TextWidth('X');
        IncrY:=TextHeight('X');
        OutTextXY(IncrX*(X-1),IncrY*(Y-1),Texto)
    End;

PROCEDURE FullPort;
    Begin
        SetViewPort(0,0,GetMaxX,GetMaxY,ClipOn)
    End;

END.
```

APENDICE B

MANUAL DEL USUARIO PROGRAMA TurnUP

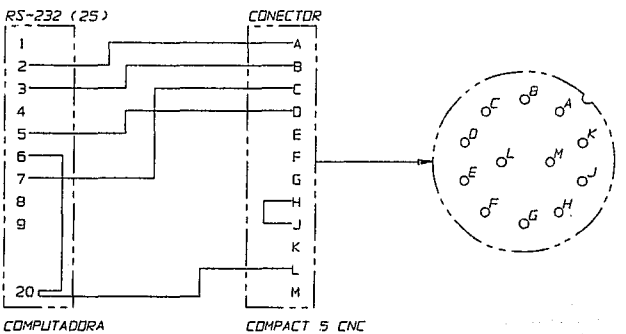
1. COMO EMPEZAR

a) Requisitos del sistema

Es necesario contar con una computadora de tipo AT con procesador 80286 por lo menos. El uso de coprocesador es ideal pero no necesario.

Se pueden utilizar las siguientes tarjetas de video: CGA, HERCULES, EGA y VGA.

Para la transmisión de programas a la máquina COMPACT 5 CNC es necesaria la instalación de un cable con la configuración que se muestra:



Los parámetros del puerto son los siguientes:

Puerto: Com1
Baud rate: 300
Paridad: None
Stop bits: 1
Word lenght: 7

b) El programa

Se necesitan sólo 150 kBytes en disco para almacenar el programa TurnUP y sus archivos. A continuación se enlistan los archivos necesarios para la ejecución del programa:

1. TURNUP.EXE
2. CGATORN.DRV
   EGATORN.DRV
   HERCULES.DRV
   VGATORN.DRV
3. CGA.BGI
   EGAVGA.BGI
   HERC.BGI
   EGAVGA.BGI
4. CODIGO.CN
5. CODIGO.HLP (opcional)

Para cargar el programa, basta con teclear TURNUP en la línea de sistema operativo.
Se puede forzar el funcionamiento de un modo de video en especial:

Entrada		Video
TURNUP	C	CGA
	E	EGA
	V	VGA
	H	HERCULES
	?	Ayuda

Si no se suministra parámetro, el programa hace una detección automática de la tarjeta de video en uso.

2. CONTROL DE ARCHIVOS

Los archivos de programas son guardados en el directorio raíz de la unidad de discos en uso con la extensión CNC. Para cambiar la unidad de discos, basta con especificar la letra de la unidad seguida de dos puntos (a:, b:, etc.) en el campo de entrada del nombre del archivo que se encuentra en las opciones A y B del menú principal. Cuando se ha cargado un programa a la memoria de la computadora, la unidad especificada queda almacenada.
En la opción "Cargar programa CNC del disco", existe la posibilidad de acceso a un menú conformado por los archivos de extensión CNC contenidos en el disco, de tal manera que se puede indicar el archivo a cargar mediante las flechas o la letra que le precede.
Respecto al directorio, éste muestra una lista de los archivos de programas CNC en el disco. Además del nombre, se proporcionan los siguientes datos:

ARCHIVO	Muestra el nombre del archivo con extensión	CNC
UNIDADES	Milímetros o pulgadas	
DESCRIPCION	Explicación de lo que hace el programa	
TAMAÑO	Espacio que ocupa el archivo en disco (bytes)	

BLOQUES	Número de bloques de programa
FECHA	Fecha de la última actualización

3. EDICION DE PROGRAMAS

La edición de los programas sólo se puede llevar a cabo cuando se cuenta con un programa en memoria. En caso contrario, tendrá que declararse un nuevo archivo.

a) Programa nuevo

Al escoger la opción de "Programa nuevo" en el menú "Edición de programas", se irá pasando por una serie de preguntas que definirán el nuevo programa. En orden de aparición, las preguntas son las siguientes:

- Programación en milímetros o pulgadas
- Nombre del programador
- Descripción del programa
- Diámetro de la pieza en bruto
- Largo de la pieza en bruto
- Nombre del archivo

Con todos los datos anteriores suministrados, se realiza la asignación de memoria del programa.

b) Programa en memoria

Al entrar a esta opción, el contenido del programa se despliega en la pantalla del lado izquierdo. El lado derecho de la pantalla presenta los datos generales del programa, mismos de los que se habló en el inciso anterior. Al oprimir la tecla F1, el desplegado de la derecha cambia a una pantalla de ayuda que muestra las funciones que puede efectuar el editor:

F1 - Parámetros de programa
Esta opción presenta la pantalla anterior a la de las funciones. En esta pantalla se muestran los datos de nombre del programador, descripción del programa, largo y diámetro de la pieza en bruto.
F2 - Grabar programa
Almacena el programa en disco sin preguntar el nombre del mismo, como se hace en la opción B del menú principal.
F3 - Imprime programa
Envía un listado del programa a la impresora incluyendo el nombre del archivos y los datos generales del mismo.
F4 - Busca línea.
Localiza la línea cuyo número se especifica. El número no puede ser menor que cero ni mayor al número de línea en la que se encuentra el comando M30
F5 - Borra línea
Elimina la línea de programa en la que se encuentre el cursor.
F6 - Lista comandos
Presenta un menú de los comandos reconocidos por el programa en donde se puede seleccionar alguno para una explicación breve.
F7 - Busca comando
Localiza la línea de programa que contenga el comando suministrado. La búsqueda se inicia en la línea siguiente a donde se encuentra el cursor.

F8 - Ayuda a comando
Presenta una explicación breve del comando correspondiente a la línea en que se encuentra el cursor.
F9 - Datos del programa
Da la posibilidad de cambiar alguno de los datos suministrados en la apertura de programa nuevo (programador, descripción, diámetro, largo).
F10 - Herramientas
Permite el cambio de la secuencia de herramientas para la simulación.

La tecla ENTER ofrece dos modalidades de edición: la primera, estando el cursor ubicado en el campo N, inserta una línea dando posibilidad de ingresar datos a los campos restantes dependiendo del comando. La segunda modalidad es de edición; ubicando el cursor sobre uno de los campos X, Z, F o H, se puede modificar su contenido. El campo G no puede ser modificado.

Al insertar un nuevo comando, si se trata de un código G, sólo basta con dar los dígitos que le siguen y automáticamente insertará la letra G. Para insertar un código M, es necesario oprimir la tecla (+) o (M) antes de dar los números que lo definen.

4. SIMULACION

La opción F del menú principal da entrada al módulo de simulación que tiene tres opciones:

- Ejecución continua
Ejecuta la simulación presentando el material maquinado y la herramienta con movimientos ininterrumpidos.
- Ejecución por bloques
Igual que la ejecución continua con la variación de que antes de cada bloque se da una pausa hasta que el usuario oprima la barra espaciadora.
- Simulación rápida
Presenta el contorno del material en bruto y sólo la trayectoria de la herramienta punto a punto.

Independientemente de la modalidad que se seleccione, se cuenta con dos funciones de ayuda mientras la herramienta está en movimiento:

(Esc) Provoca una pausa durante la ejecución dando las opciones de continuar o abortar el proceso.
(F1) Proporciona un cálculo aproximado del tiempo real transcurrido. Al finalizar la simulación, automáticamente se activa esta opción.

Durante la ejecución del programa, el bloque en acción se muestra en la parte superior de la pantalla. Las coordenadas proporcionadas en la esquina inferior izquierda corresponden al sistema absoluto.

6. TRANSMISION

Para realizar la transmisión de datos a la máquina COMPACT 5 CNC, se debe contar con el cable que conecta a la computadora con la máquina.

Una vez conectado el cable en el puerto 1 de comunicaciones, se puede dar la opción F del menú principal; al hacer esto, aparecerá un letrero dando la indicación de ejecutar el comando G66 + INP + INP en la máquina COMPACT 5 CNC. En este momento el controlador está listo para recibir datos; bastará oprimir cualquier tecla en la

computadora para iniciar la transmisión. Al finalizar, se presenta el mensaje "transmisión finalizada".

APENDICE C

CODIGOS PREPARATORIOS (GEOMETRICOS) Y MISCELANEOS

G00 Interpolación lineal de movimiento rápido.

Sintaxis G00 X.... Z.....

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.

G01 Interpolación lineal.

Sintaxis G01 X.... Z..... F...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: mm/min o in/min
mm/rev o in/rev

G02 Interpolación circular a 90 grados en el sentido del reloj.
De Z+ a Z-

Sintaxis G02 X.... Z..... F...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: m/min o in/min
mm/rev o in/rev

G03 Interpolación circular a 90 grados en el sentido contrario al reloj. De Z+ a Z-

Sintaxis G03 X.... Z..... F...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: m/min o in/min
mm/rev o in/rev

G04 Pausa de tiempo

Sintaxis G04 t...

t = Tiempo en centésimas de segundo (100 = 1 seg).

G21 Linea en blanco. Comando sin efecto.

Sintaxis G21

G25 Llamada de subrutina. El control de programa salta a la línea indicada y regresa con el comando M17.

Sintaxis G25 L...

G27 Salto de línea. El control de programa salta a la línea indicada.

Sintaxis G27 L...

G33 Maquinado de rosca.

Sintaxis G33 X.... Z..... F... P...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: mm/min o in/min
mm/rev o in/rev
P = Paso: hilos/mm o hilos/in

G70 Programación en pulgadas. No se puede redefinir a lo largo del programa.

Sintaxis G70

G71 Programación en milímetros. No se puede redefinir a lo largo del programa.

Sintaxis G71

G73 Ciclo de barrenado con incrementos de 200 unidades.

Sintaxis G72 Z..... F...

Z = Coordenada final del movimiento longitudinal.
F = Avance: mm/min o in/min
mm/rev o in/rev

G78 Ciclo de roscado. Las coordenadas corresponden a la esquina opuesta del rectángulo definido por el punto de inicio y el final en forma ortogonal.

Sintaxis G78 X.... Z..... F... P...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: mm/min o in/min
mm/rev o in/rev
P = Paso: hilos/mm o hilos/in

G81 Ciclo de barrenado de un solo paso.

Sintaxis G81 Z..... F...

Z = Coordenada final del movimiento longitudinal.
F = Avance: mm/min o in/min
mm/rev o in/rev

G82 Ciclo de barrenado de un solo paso con tiempo de espera de 1/2 seg.

Sintaxis G82 Z..... F...

Z = Coordenada final del movimiento longitudinal.
F = Avance: mm/min o in/min
mm/rev o in/rev

G83 Ciclo de barrenado con salida al punto de partida.

Sintaxis G83 Z..... F...

Z = Coordenada final del movimiento longitudinal.
F = Avance: mm/min o in/min
mm/rev o in/rev

G84 Ciclo de trabajo longitudinal. Las coordenadas corresponden a la esquina opuesta del rectángulo definido por el punto de inicio y el final en forma ortogonal.

Sintaxis G84 X.... Z..... F... H...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: mm/min o in/min
mm/rev o in/rev
H = Profundidad de corte.

G85 Ciclo de rimado. Entrada y salida en un solo paso con avance programado.

Sintaxis G85 Z..... F...

Z = Coordenada final del movimiento longitudinal.
F = Avance: mm/min o in/min
mm/rev o in/rev

G86 Ciclo de tronzado. Las coordenadas corresponden a la esquina opuesta del rectángulo definido por el punto de inicio y el final en forma ortogonal.

Sintaxis G88 X.... Z..... F... H...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: mm/min o in/min
mm/rev o in/rev
H = Ancho del tronzador.

G88 Ciclo de trabajo transversal. Las coordenadas corresponden a la esquina opuesta del rectángulo definido por el punto de inicio y el final en forma ortogonal.

Sintaxis G88 X.... Z..... F... H...

X = Coordenada en X del punto final.
Z = Coordenada en Z del punto final.
F = Avance: mm/min o in/min
mm/rev o in/rev
H = Profundidad de corte.

G89 Ciclo de rimado con tiempo de espera de 1/2 seg. Entrada y salida en un solo paso con avance programado.

Sintaxis G89 Z..... F...

Z = Coordenada final del movimiento longitudinal.
F = Avance: mm/min o in/min
mm/rev o in/rev

G90 Programación en sistema absoluto. El origen del sistema se localizará en el extremo derecho de la pieza en el diámetro = 0. Todas las coordenadas parten de este punto y la coordenada.X maneja diámetros.

Sintaxis G90

G91 Programación en sistema incremental. Los valores de coordenadas se miden respecto al lugar en que se encuentra la herramienta.

Sintaxis G91

G92 Programación en sistema absoluto con especificación de la referencia de partida de la herramienta.

Sintaxis G92 X.... Z.....

X = Coordenada en X de la herramienta.
Z = Coordenada en Z de la herramienta.

G94 Definición de avance por minuto. (mm/min)

Sintaxis G94

G95 Definición de avance por revolución o giros del husillo. (mm/rev)

Sintaxis G95

M00 Pausa programada.

Sintaxis M00

M03 Encendido del husillo en el sentido del reloj.

Sintaxis M03

M05 Paro del husillo.

Sintaxis M05

M06 Cambio de herramienta.

Sintaxis M06 X.... Z..... T... D...

X = Corrección de la herramienta en X.
Z = Corrección de la herramienta en Z.
T = Número de herramienta.
D = Diámetro de la broca o rima (sólo aplicable en los ciclos G73, G81, G82, G83, G85 y G89)

M17 Fin de subrutina. Sólo funciona con la llamada de un comando G25.

Sintaxis M17

M30 Fin de programa. Sólo se puede colocar un comando M30 al final de un programa.

Sintaxis M30

M98 Compensación automática de juego mecánico.

Sintaxis M98 X.... Z.....

X = Compensación en X.
Z = Compensación en Z.

M99 Definición de coordenadas de centro para interpolación circular. Sólo aplicable con el uso de G02 y G03.

Sintaxis M99 I.... K....

I = Distancia desde el punto de partida al centro del círculo en X.
K = Distancia desde el punto de partida al centro del círculo en Z.

APENDICE D

INFORMACION TECNICA DE LA MAQUINA COMPACT 5 CNC

DATOS TECNICOS DE LA UNIDAD CNC

1. Memoria de programa:	210 bloques
2. Velocidad de avance:	2 - 499 mm/min 0.002 - 0.499 mm/rev
3. Velocidad de recorrido rápido:	700 mm/min
4. Pasos de roscado:	0.02 - 4.99 (incrementos de 0.01mm)

5. Programación de radios con M99
6. Alarmas en eventos de mal funcionamiento
7. Modificación de programas durante la ejecución
8. Programación absoluta o incremental
9. Cálculo de compensación de herramientas
10. Memoria con cinta magnética
11. Interface RS-232, conexión de video RCA
12. Interface DNC

DIRECCIONES DE ALMACENAMIENTO (SW-A6C 114 004)

	Milímetros		Pulgadas	
	Valor	Dimensión	Valor	Dimensión
N. Número de bloque				
	00-299	1	0-299	1
G. Función de movimiento				
	00-95	1	00-95	1
M. Función miscelánea				
	00-99	1	00-99	1

X. Coordenadas de movimiento				
	0±5999	1/100 mm	0± 1999	1/1000"
Z. Coordenadas de movimiento				
	0±32760	1/100 mm	0± 12900	1/1000"
F. Avance				
	2-499		2-199	
G94		mm/min		110"/min
G95		1/1000 mm/rev		1/10000"/rev
I. Centro de círculo				
	0-5999	1/100 mm	0-1999	1/1000"
K. Centro de círculo				
	0-22700	1/100 mm	0-7500	1/1000"
L. Dirección de salto				
	0-221	1	0-221	1
X. Tiempo de espera				
	0-5999	1/100 seg	0-1999	1/100 seg
T. Dirección de herramienta				
	0-499	1	0-499	1
H. Parámetro de división de corte				
	0-999	1/100 mm	0-999	1/1000"
K. Paso de rosca				
	2-499	1/100 mm	2-199	1/1000"

TABLERO DE CONTROL DE LA MAQUINA COMPACT 5 CNC

1. Switch principal
2. Luz de indicación de encendido
3. Botón de paro de emergencia
4. Pantalla de velocidad del husillo
5. Switch del husillo (0 - 1 - CNC)
6. Switch de selección de trabajo en milímetros o pulgadas
7. Amperímetro para la corriente del motor principal
8. Unidad de cinta magnética
9. Botón para selección de modo manual o programación
10. Luz de indicación de modo de programación
11. Botón de arranque
12. Teclado para entrada de programas
13. Pantalla de valores de los campos de programa
14. Luces de indicación del campo de trabajo

APENDICE E

EJEMPLOS

SUB.CNC

```
NOMBRE DEL PROGRAMADOR: Mauricio Rosales
DESCRIPCION DEL PROGRAMA: Prueba de subrutinas
UNIDADES : mm
LARGO    : 50.0000000000
DIAMETRO : 20.0000000000
HERRAMIENTAS:  1  2  3  4  5  6  7  8  9
000 G71
001 G92  2000       500
002 G91
003 G00  100        0
004 G00  0          -1500
005 G00  -100       0
006 G25                      L12
007 G00  100        0
008 G00  0          -800
009 G00  -100       0
010 G25                      L12
011 G27                      L16
012 G01  -100       -100     100
013 G01  0          -800     100
014 G01  100        -100     100
015 M17
016 G90
017 G00  3000       500
```

EMCO.CNC

```
NOMBRE DEL PROGRAMADOR: Mauricio Rosales
DESCRIPCION DEL PROGRAMA: Programa del curso EMCO, Sep 1990
UNIDADES : mm
LARGO    : 100.0000000000
DIAMETRO : 22.0000000000
HERRAMIENTAS:  1  2  3  4  5  6  7  8
000 G71
001 G92  3200      500
002 G94
003 M06  0         0         T1        D0
004 M03
005 G00  2200      100
006 G84  2000      -1980     100       50
007 G00  2050      -1000
008 G01  1850      -1300     100
009 G01  1850      -1980     100
010 G01  2250      -1980     100
011 G01  1850      -1300     100
012 G01  1650      -1600     100
013 G01  1650      -1980     100
014 G01  2250      -1980     100
015 G00  2250      100
016 G00  1600      100
017 G01  2000      -100      80
018 G01  2000      -1000     80
019 G01  1600      -1600     80
020 G01  1600      -2000     80
021 G01  2150      -2000     80
022 G01  2150      -2500     80
023 G00  3200      500
024 M06  0         0         T5        D0
025 G00  2000      100
026 G78  1784      -1700     K125      5
027 G00  3200      100
028 G00  3200      500
029 M30
```

ROSCA.CNC

```
NOMBRE DEL PROGRAMADOR:
DESCRIPCION DEL PROGRAMA:
UNIDADES : mm
LARGO    : 50.0000000000
DIAMETRO : 20.0000000000
HERRAMIENTAS:  1  2  3  4  5  6  7  8
000 G71
001 G92  3200      500
002 M06  0         0         T5        D0
003 M03
004 G00  2200      100
005 G78  1748      -4000     K150      20
006 G00  3000      100
007 G00  3000      500
008 M30
```

FACING.CNC

```
NOMBRE DEL PROGRAMADOR:
DESCRIPCION DEL PROGRAMA:
UNIDADES : mm
LARGO    : 50.0000000000
DIAMETRO : 22.0000000000
HERRAMIENTAS:  1  2  3  4  5  6  7  8
000 G71
001 G92  2600      200
002 G00  2400      0
003 G88  600       -400      100       60
004 G00  2600      200
005 M47
006 M30
```