A PC-BASED CNC MACHINE CONTROLLER

Dr. Waladin K. Sa'id (prof.)
Control & Computers Eng. Dept.
University of Technology

Asaad A. M. Al-Sudani
(Lecturer)

Abdul-Razak T. Saeed
Electrical Eng. Dept / College of Engineering
University of Baghdad

ABSTRACT
This paper presents the development of a flexible, modular, and cost-effective PC-Based CNC machine controller. Software is designed and developed to perform most of the standard CNC machines control functions. This software is capable of accepting and interpreting the standard CNC (G-Code) programs. It also performs the most important CNC machine controller functions; namely, interpolation and feed rate control. The software has been developed using the Visual C++ programming language operating under WINDOWS environment. It has high flexibility and expandability that makes the task of future development easier. System performance has been evaluated by applying the developed software to control an X-Y recorder, simulating a CNC machine.

KEY WORDS
CNC machines, Controller, G-Code interpretation, Machining, Software Development.

الخلاصة
تقدم هذه الورقة تطوير بناء مسيطر مرن معياري قليل الكلفة لمكائن السيطرة العددية باستخدام الحاسب الشخصي. صمم و طور كيان رياضي يؤدي معظم وظائف مكائن السيطرة العدديـــة القياســـية. يـــوفر البرنامج المصمم إمكانية تقبل و ترجمة البرامج القياسية (شفرة-جي) لمكائن السيطرة العددية. كما يؤدي هذا البرنامج أهم وظائف المسيطر على مكائن السيطرة العددية ألا وهي توليد المسار (Interpolation) و السيطرة على سرعة القطع. صمم البرنامج باستخدام لغة البرمجة (Visual C^{++}) التي تعمل في بيئة نظام التشغيل نوافذ (WINDOWS) . يمتاز البرنامج المصمم بمرونته العالية و قابليته للتوسع مما يجعل مهمة تطويره المستقبلي سهلة. قيم أداء البرنامج عمليا بتطبيقه على منظومة مسجل س _ص (راسم) للـــتحكم بحركة القلم والذي اتخذ نموذج محاكاة لماكنة السيطرة العددية.

INTRODUCTION
The introduction of computerized numerical control (CNC) machining systems in the early 1970s created an important application. Over the years, the requirements imposed on CNC machines have steadily increased in their sophistication, necessitating the use of computers for their control (Hariharan et al. 1978). Because of their obvious advantages of speed, dependability, large storage capacity, and relatively low cost, modern personal computers (PCs) are being used more and more as an integral part of CNC systems.

Today's increasing processor speeds and decreasing costs lure us to program most of the conventional hardware controller functions inside a PC. Such an approach not only overcomes problems like large physical space, time and cost of upgradation present in conventional hardware controllers, but also provides us with a better user interface (Kommaredy et al. 1999).
CNC machines have their own programming language. To incorporate a PC in the manufacturing process, it is necessary first to build powerful software that is capable of performing the related control functions. Therefore, the software to be developed must provide the capability of accepting, interpreting, and executing programs written in that language. In other words, it must make the PC understand what the machine can execute. Hence, the software must contain three interactive modules: input module, command interpreter module, and execution module.
The input module is required to provide a way of inputting and editing CNC part programs. Besides, it has to provide these programs with some kind of storage media (memory, disc files, or possibly both). The command interpreter module is required to interpret the part program commands and extract the necessary instructions from the raw input data available in those commands. The extracted instructions should then be passed to the execution module, which is responsible for executing them.
This paper is concerned with the development of a flexible, modular, easy to use software capable of performing CNC control functions. The Visual C++ programming language operating under WINDOWS environment will be used for this purpose.

SOFTWARE DESIGN

In order to enable the personal computer to accept part programs and execute them, it might be helpful here to devise a specific method. Hereby, each block of the part program can be stored in the memory as a distinct object containing all the information provided by the programmer in that block. For instance, if the programmer has programmed a linear interpolation (G1) block, the X, Y, and Z distances, and the feed rate (F) value have all to be stored in an individual corresponding object.
To meet this requirement a C-language structure has thereby been used to store one block of information of the part program. The definition of this structure in the designed Visual C++ program is as follows:

```
typedef struct
}
int n, g ،m, t؛
long s, f؛
float x, y, z, k, p, l؛
{block؛
```

Where n, g, m, t, s, f, x, y, and z are variables used to store the values of the N, G, M, T, S, F, X, Y, and Z codes, respectively. Each one of the other variables (k, p ،and l) is used to store different codes according to the programmed operation (specified by the G-Code). For example, the k variable is used to store the J-code if a circular interpolation block is programmed, while it is used to store the Q-code in case of a peck drilling cycle. This strategy was used to reduce the storage requirements of the (block) structure.
Part programs, however, normally contain more than one block of information. Therefore an array whose elements are structures of type (block) has been used to store the whole part program. This array is associated with an index to keep track of the last entered block.
The array with its associated index were defined as elements of another structure (partprogram) as follows:

```
typedef struct
}
block blockarray[1000؛[
```

int index;
{partprogram;
Finally a pointer to the partprogram structure was defined as given below:
partprogram* ptr;
This pointer allows access to the index and all of the elements (block structures) of the array.
To provide a method for entering and editing part programs, a dialog box as shown in **Fig. (1)** was developed. This dialog box contains 18 edit boxes, each of which corresponds to one CNC code. Similar dialog boxes were developed to save / load part programs, to start / stop the CNC machine, and to set machine parameters.

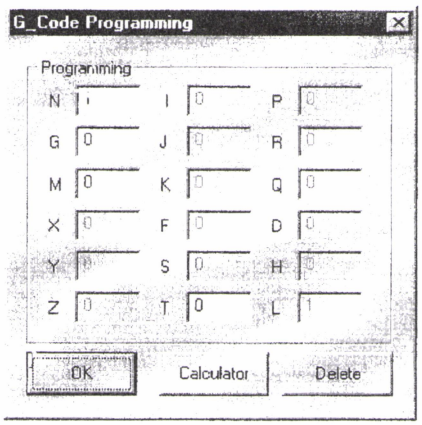


Fig. (1). The G_code programming dialog box.

INTERPOLATION AND FEED RATE CONTROL

The most basic functions of a CNC machine control system are the interpolation and the feed rate control. To perform these functions, two software routines have been developed; the interpolation routine and the feed rate routine.
The interpolation routine performs linear and circular (clockwise and counterclockwise) interpolations and is based on the software Digital Differential Analyzer (DDA) method (Koren 1979), (Koren and Masory 1981), (Koren 1976), (Pfeifer and Derenbach 1973), and (Al-Sudani 1986). **Fig. A.(1)**(Appendix A) shows the flow chart of this subroutine.
The feed rate routine, on the other hand, has been designed to use a WINDOWS timer callback function. For a user specified feed rate of F millimeters per minute, the timer callback frequency is f = F / (60 * BLU) in Hertz. BLU is the minimum length distinguishable by the machine control unit and it is a measurement of the resolution of the system. The callback period is T = 60 * BLU / F in seconds (Spence 2000). During each callback function, a single iteration of the currently active interpolation routine is executed. The flow chart of the feed rate routine is shown in **Fig. A. (2)**.

POSITIONING ROUTINE

In case of idle machining, the path of the cutting tool while travelling from one point to another is without any significance. The only significant point here is to move the tool as rapidly as possible.
As the software DDA method imposes a restriction on the maximum speed of the machine tool, a simple and fast algorithm has been developed to perform positioning. This algorithm simulates the operation of three position counters, one for each axis. The counters are incremented by one for each iteration of this algorithm. Reaching the required position in any axis stops the count of

the corresponding counter in subsequent iterations. The flow chart of the positioning routine is given in Appendix A, **Fig. A. (3)**.

APPLICATION

To prove the validity of the designed software, it has been applied to control an X-Y recorder. This device was chosen to be the test platform since its operation principle is similar to that of the CNC machine. Hereby, the recording area of the X-Y recorder simulates the CNC machine table and its pen simulates the cutting tool of the machine. **Fig. (2)** depicts the block diagram of the experimental set up used for this purpose.

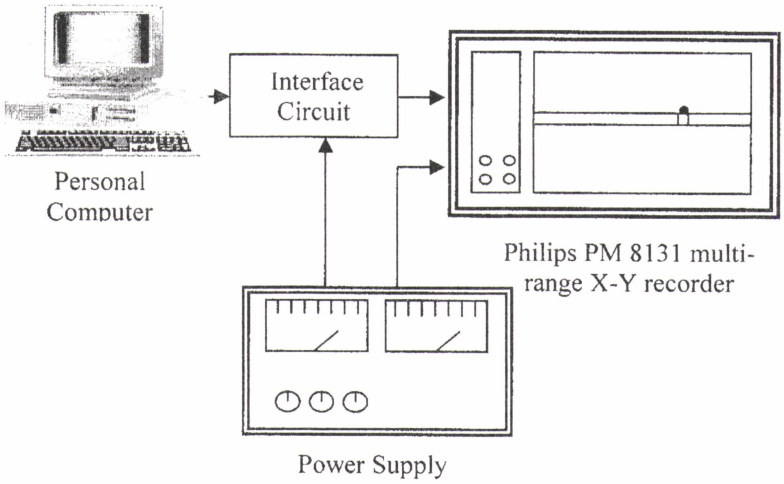


Fig. (2) The developed system block diagram.

RECORDER INTERFACE CIRCUIT

This circuit interfaces the X-Y recorder to the parallel (printer/centronic) port of the PC. The parallel port is an industrial standard interface designed for connecting printers to the PC. However, it can be used for other purposes. The port connector on the PC is a 25-pin D-type female connector. The connector pins are divided into three basic groups; namely the data group, the control group, and the status group (An 1998).

Only four pins of the data group and one pin of the control group are connected to the recorder interface circuit. The four pins of the data group are used to provide the Up-Down counters of the recorder interface with the necessary clock and direction signals. The reset signal of the Up-Down counters is supplied by the first pin of the control group. The used pins together with their corresponding numbers on the parallel port connector and their functions are listed in **Table (1)**

Table (1) The used parallel port connector pins.

Pin Name	Type	Pin No. on the connector	Function
$\overline{INITIALIZE}$	Control	16	Reset
DB0	Data	2	X-Clock
DB1	Data	3	Y-Clock
DB3	Data	5	X-Direction
DB4	Data	6	Y-Direction

The block diagram of the interface circuit is depicted in **Fig. (3)**. The first stage of this circuit is the buffering stage, which provides the required current amplification to the parallel port output signals.The buffer's output signals supply four Up-Down counters (two cascaded 4-bit counters to form an 8-bit counter for each axis) with the necessary count and direction signals. The complete circuit diagram of the interface circuit is given in **Fig. B.(1)** (Appendix B).

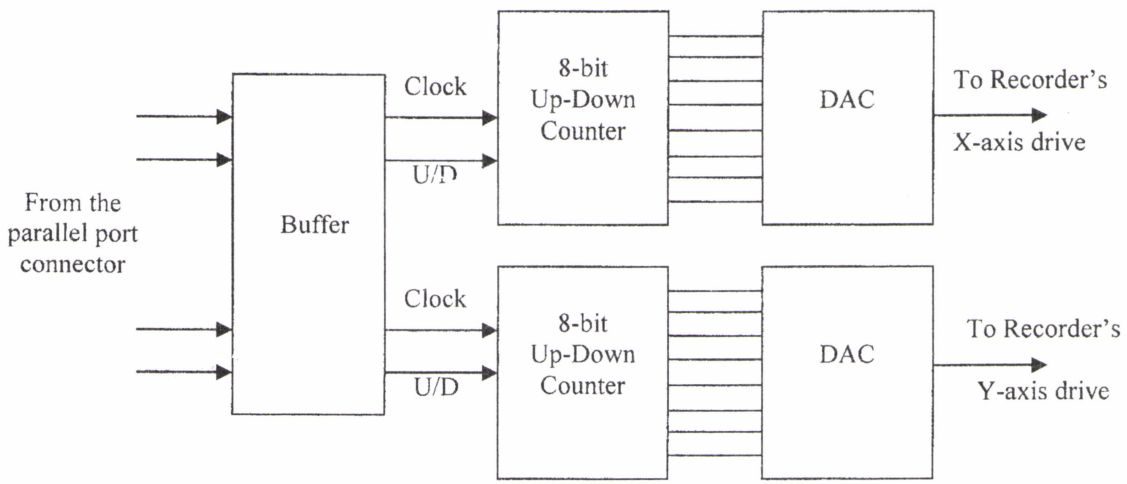


Fig. (3) Block diagram of the interface circuit.

OPERATION PRINCIPLE

To simplify the description of the operation principle of the overall system, the description is divided into two parts; the software operations and the hardware operations.

Software Operations

The control program starts its operation by fetching the part program instruction (one by one) from the memory. The control program then decodes each instruction and accordingly calls the appropriate interpolation routine. Thus passing all the necessary values that comprehend incremental distances and feed rate. In turn, the interpolation routine uses the passed values to perform the necessary computations. The result of these computations is stored in a variable (K) in the memory. The bits functions of this variable are hereby given in **Table (2)**.
The following piece of code illustrates how the X-Direction and Y-Direction bits are set in the interpolation routine:

```
If (X > 0) K=K | 8;
If (Y > 0) K=K | 16;
```

Where X and Y are the required (passed) incremental distances in the X and Y directions, respectively, and (|) is a C-language operator that performs bit-wise OR operation.
The X-Clock and Y-Clock bits are handled similarly. For example, if a pulse needs to be sent to the X-axis, the following operation is performed:

```
K=K | 1;
```

If, on the other hand, a pulse needs to be sent to the Y-axis, the operation given below is performed:

```
K=K | 2;
```

Table (2) Functions of the variable K bits.

Bit Number	Function
1(LSB)	X-Clock
2	Y-Clock
3	Z-Clock
4	X-Direction
5	Y-Direction
6	Z-Direction
7	Unused
8(MSB)	Unused

At the end of each iteration of the interpolation routine, the value of K is sent (through the PC parallel port) to the interface circuit.

Hardware Operations

The parallel port output signals are fed (through the buffer) to the Up-Down counters. Depending on the direction signal, the counter count is incremented or decremented every time a pulse is received by its clock input. The digital output of each one of the two 8-bit counters (X-axis counter and Y-axis counter) is then converted into an analog signal by an 8-bit DAC. Each one of the two resulting analog signals is fed directly to drive the motor of the corresponding axis of the X-Y recorder.

The magnitudes of the recorder input voltages control the position of the drawing pen within the recording area. For instance, if the input range of the recorder is set to 0.1 V/ cm, then a 3V on the X input and a 2V on the Y input yield a position of (3/0.1=30 cm) on the X-axis and (2/0.1=20 cm) on the Y-axis.

TEST PROGRAM

The application of a test G-Code program is given in this section. The dimensions (in mm) of the test part are exhibited in **Fig. (4)**. The G-Code program for this part consists of a sequence of linear and circular interpolation blocks. A complete listing of this program is given below.

N1G0X20Y30Z0;
N2G1X0Y60Z0F60000;
N3G1X52Y0Z0F60000;
N4G3X16Y16I0J16F60000;
N5G3X-16Y16I-16J0F60000;
N6G1X-52Y0Z0F60000;
N7G1X0Y72Z0F60000;
N8G2X40Y40I40J0F60000;
N9G1X52Y0Z0F60000;
N10G1X0Y-68Z0F60000;
N11G3X16Y-16I16J0F60000;
N12G3X16Y16I0J16F60000;
N13G1X0Y68Z0F60000;
N14G1X52Y0Z0F60000;
N15G2X40Y-40I0J-40F60000;
N16G1X0Y-72Z0F60000;
N17G1X-52Y0Z0F60000;
N18G3X-16Y-16I0J-16F60000;

N19G3X16Y-16I16J0F60000;
N20G1X52Y0Z0F60000;
N21G1X0Y-60Z0F60000;
N22G1X-92Y0Z0F60000;
N23G1X0Y68Z0F60000;
N24G3X-16Y16I-16J0F60000;
N25G3X-16Y-16I0J-16F60000;
N26G1X0Y-68Z0F60000;
N27G1X-92Y0Z0F60000 ;

The X-Y recorder output resulting from executing the above program is shown in **Fig. 5**. The figure shows a medium quality output. This is due to the open loop nature of the recorder driver and the inherent error in the 8-bit digital output. The latter is estimated to be no less than 2 mm per bit. However, the prime objective of this experimental work is to demonstrate the overall software and machine applicability performance. In real CNC machines, however, the driver will operate in closed loop with higher digital bit resolution.

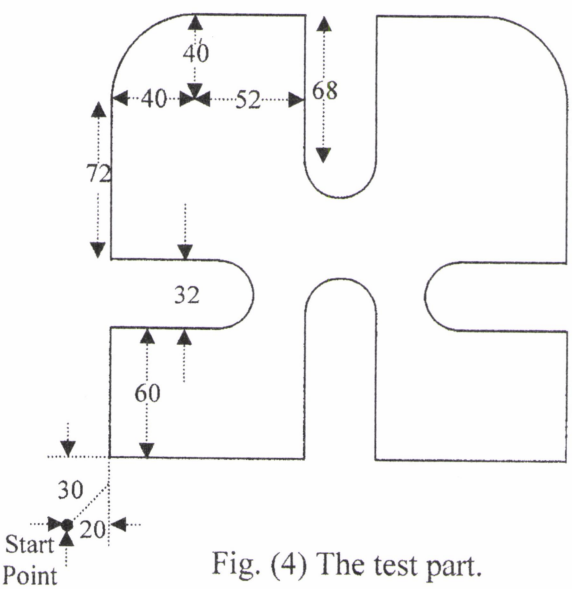


Fig. (4) The test part.

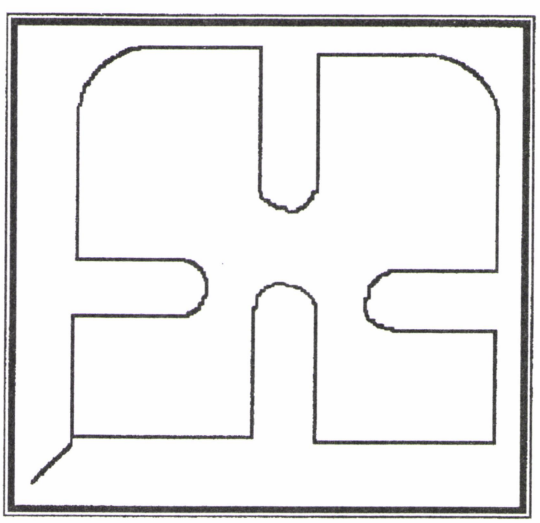


Fig. (5) The recorder output for the test part.

CONCLUSIONS

A modular and a cost- effective CNC machine software has been designed, implemented, and tested. To verify the operation of the control software, it has been applied to control an X-Y recorder, simulating a CNC machine. The results obtained are successful
even with the relatively medium quality of the X-Y recorder output. The open loop nature of the recorder driver is behind this medium quality of the output. Using a closed loop driver should improve this quality.

REFERENCES

Al-Sudani A. (1986), Digitization of Non-Integer Computer Generated Circular Segments by Error Criteria Minimization Algorithms using Storage Matrix Technique, 2nd Int. Baghdad Conference on Computer Technology and Applications, National Computer Centre, Baghdad, 24-26 March.

An P. (1998), PC Interfacing Using Centronic, RS232, and Game Ports. Newnes.

Hariharan Y., Rao K., and Bhattacharjee J. (1978), Design of a Microprocessor-Based Numerical Control System, IEEE Transactions on Industrial Electronics and Control Instrumentation, Vol. 25, No. 4, PP 355-362, November.

Kommaredy S., Kazuo Y., and Yoshihito K., (1999), PC-Based Open Architecture Servo Controller for CNC Machining, Email surya@ ucdavis.edu.

Koren Y., (1976), Interpolator for a Computer Numerical Control System, IEEE Transactions on Computers, Vol. 25, No. 1, PP 32-37, January.

Koren Y., (1979), Design of Computer Control for Manufacturing System, ASME Journal of Engineering for Industry, Vol. 101, PP 326-332, August.

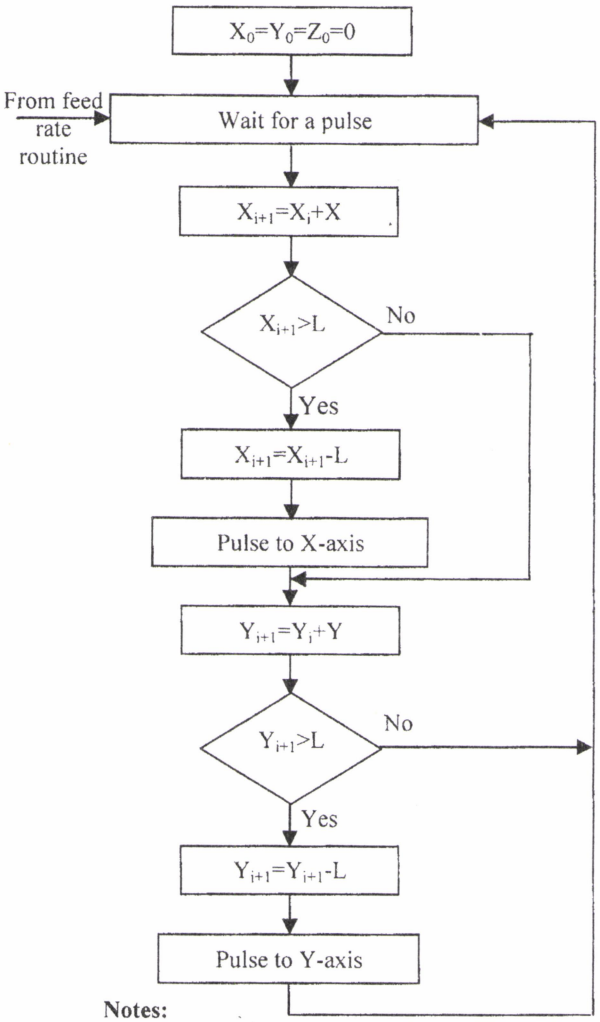
Koren Y., and Masory O. (1981), Reference-Pulse Circular Interpolators for CNC Systems, ASME Journal of Engineering for Industry, Vol. 103, PP 131-136, February.

Pfeifer T. and Derenbach T. (1973), Report on a Special CNC System, Proceedings of the 14th IMTDR Conference, PP 367-372.

Spence A., and Chan, H. (2000), A Table Top CNC / CMM Teaching Apparatus, ASME

APPENDIX A

Flow charts of the Interpolation and Feedrate and Positioning Routines

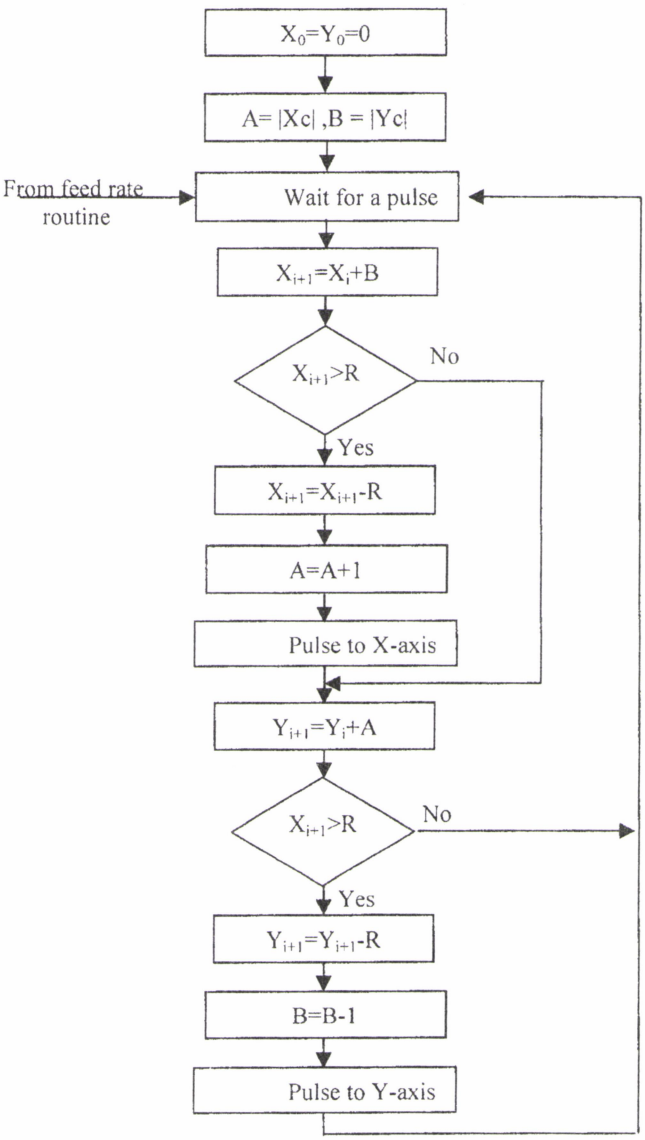


Notes:

*X, Y, and Z are the required incremental distances in the x, y, and z axes, respectively.

*L is the length of the line.

FigA.(1) Flow chart of linear interpolation routine.

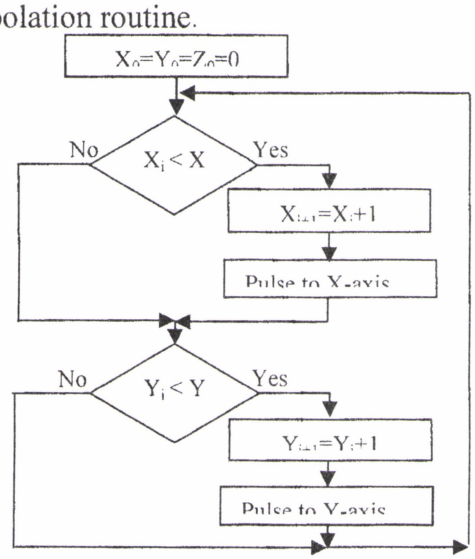


Notes:

*X_2 and Y_2 are the incremental distances to the x and y coordinates of the circular arc end point.

*Xc and Yc are the incremental distances to the x and y coordinates of the center point of the circular arc.

Fig A. (2) Flow chart of circular interpolation routine.



Note:

* X, Y, and Z are the required incremental distances in the x,

FigA. (3) Flow chart of positioning routine.

APPENDIX B

Circuit diagram of the interface circuit used for the experimental tests.

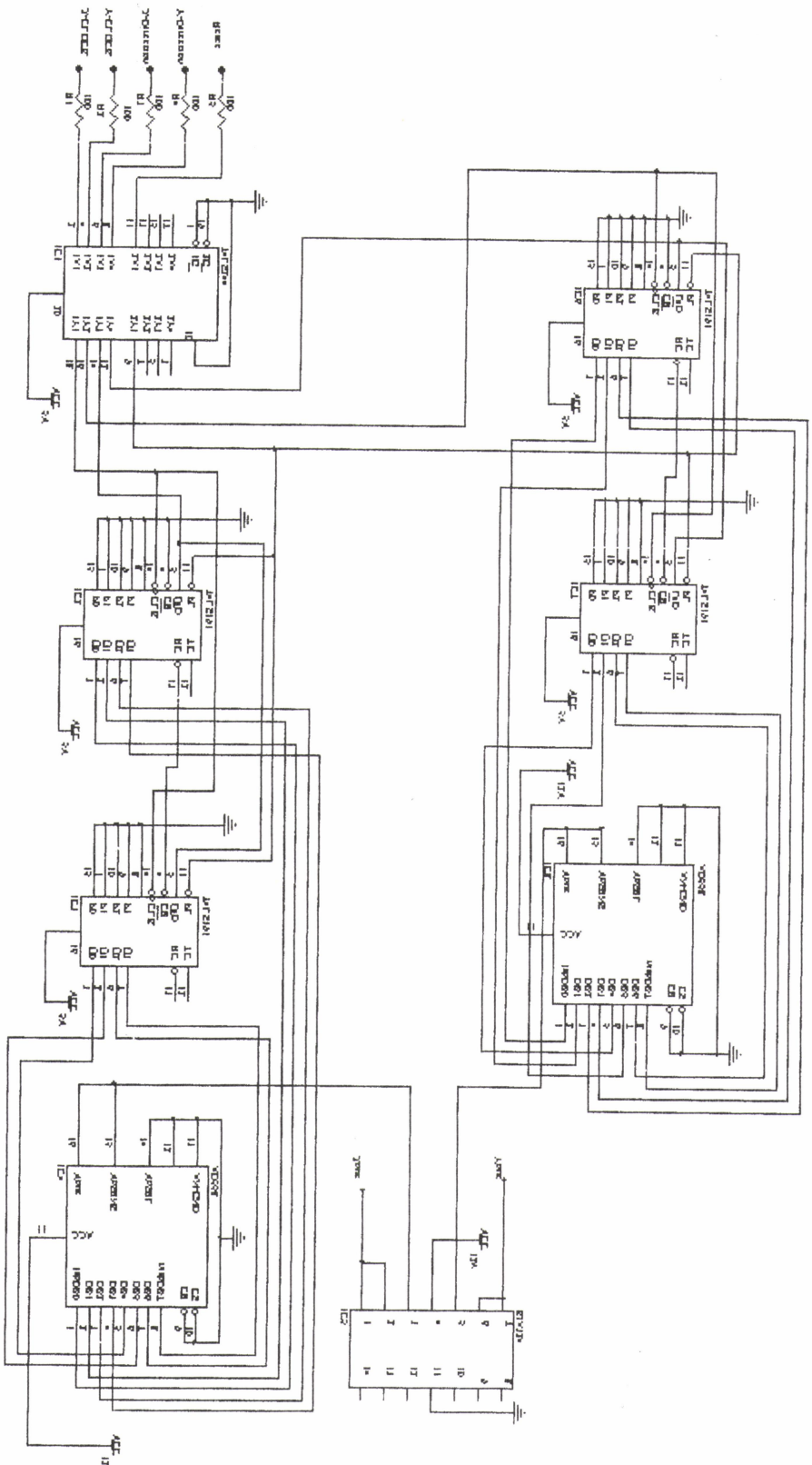


Fig. B.(1) Implemented circuit diagram.