7-9-2002

Development of an open architecture motion control system

Peng Chen
Florida International University

DOI: 10.25148/etd.FI14060173

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

DEVELOPMENT OF AN OPEN ARCHITECTURE MOTION CONTROL SYSTEM

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

in

MECHANICAL ENGINEERING

by

Peng Chen

2002

To: Dean Vish Prasad
College of Engineering

This thesis, written by Peng Chen, and entitled Development of an Open Architecture Motion Control System, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this thesis and recommend that it be approved.

Sabri Tosunoglu

Kuang-Hsi Wu

Ibrahim Nur Tansel, Major Professor

Date of Defense: July 9, 2002

The thesis of Peng Chen is approved.

Dean Vish Prasad
College of Engineering

Dean Douglas Wartzok
University Graduate School

Florida International University, 2002

DEDICATION

I dedicate this thesis to my dear parents and my lovely younger brother. Without their great personality, sincere understanding and full support, the completion of this work would not have been possible. As well, I dedicate this thesis to the acceptance of multi-culture and to the openness of world education.

ACKNOWLEDGMENTS

I wish to thank the members of my committee, Dr. Tosunoglu, Dr. Wu, and my major professor Dr. Tansel, for their support, patience and encouragement. I especially express my thanks to my major professor for ushering me into the broad kingdom of engineering research.

ABSTRACT OF THE THESIS

DEVELOPMENT OF AN OPEN ARCHITECTURE MOTION CONTROL SYSTEM

by

Peng Chen

Florida International University, 2002

Miami, Florida

Professor Ibrahim Nur Tansel, Major Professor

A multipurpose open architecture motion control system was developed with three platforms for control and monitoring. The Visual Basic user interface communicated with the operator and gave instructions to the electronic components. The first platform had a BASIC Stamp based controller and three stepping motors. The second platform had a controller, amplifiers and two DC servomotors. The third platform had a DSP module. In this study, each platform was used on machine tools either to move the table or to evaluate the incoming signal. The study indicated that by using advanced microcontrollers, which use high-level languages, motor controllers, DSPs (Digital Signal Processor) and microcomputers, the motion control of different systems could be realized in a short time. Although, the proposed systems had some limitations, their jobs were performed effectively.

TABLE OF CONTENTS

CHAPTER PAGE

I. Introduction 1

II. Open Architecture Control System 8
2.1 Open Architecture System Basis 8
2.2 Open Architecture System Development 9
2.2.1 Hardware Consideration 9
2.2.2 Software Consideration 9
2.2.3 Communication Consideration 9
2.3 Motion Control Systems with Openness 10

III. Control of a 2½ Axes Stepping Motor System 12
3.1 Controller 13
3.2 Software 17
3.2.1 Graphical Interface of the Microcomputer 17
3.2.2 Microcontroller Algorithm 18

IV. Control of a DC Servomotors System 22
4.1 Controller 23
4.2 Software 28
4.2.1 Graphical Interface of the Microcomputer 28
4.2.2 Microcomputer Interface Algorithm 33

V. DSP Based Machine Tool Monitoring System. 36
5.1 Analog Signal Generation by Using a Specialized D/A Chip 37
5.2 Signal Monitoring Capability Test of the DSP Evaluation Board 40
5.2.1 Hardware 40
5.2.2 Software 41

VI. Results and Discussion 44
6.1 Control of a 2½ Axes Stepping Motor System 44
6.2 Control of a DC Servomotors System 49
6.3 DSP Based Machine Tool Monitoring System 51
6.3.1 Analog Signal Generation by Using a Specialized D/A Chip 51
6.3.2 Performance of the DSP Evaluation Board 53

VII. System Integration 59

VIII. Conclusion 63
8.1 Future Studies 65

References 66

LIST OF TABLES

TABLE PAGE

Table 1.1 Application into the machine tool area 3

Table 1.2 Application into the robot area 4

Table 4.1 Encoder connections for one servomotor control 25

Table 4.2 Specifications of the servo amplifier 26

Table 4.3 Specifications of the servomotor 26

LIST OF FIGURES

FIGURE PAGE

Figure 3.1 Proposed 2½ axes stepping motor control system 13

Figure 3.2 Developed BASIC-Stamp based controller 14

Figure 3.3 Diagram of the control system 15

Figure 3.4 Diagram of the stepping motor control connection 15

Figure 3.5 Serial communication link between the PC and the BASIC Stamp..... 15

Figure 3.6 Distribution of predetermined regions and the region 3 table motion.....16

Figure 3.7 Graphical user interface (planning stage) 17

Figure 3.8 Graphical user interface (execution stage)..... 18

Figure 3.9 Algorithm of the information exchange module..... 19

Figure 3.10 Compilation window of the control algorithm..... 20

Figure 3.11 Memory consumption window of the control algorithm 21

Figure 4.1 Assembly picture of part of the constructed system 23

Figure 4.2 Sketch map for one servomotor control connection 24

Figure 4.3 Diagram of the system serial communication link 25

Figure 4.4 System parameter installation process 27

Figure 4.5 Reset window after the correct installation..... 28

Figure 4.6 Developed manual navigation software interface..... 29

Figure 4.7 Successful handshaking of the microcomputer and the controller 30

Figure 4.8 Planning stage of the GUI..... 31

Figure 4.9 Running stage of the GUI 32

Figure 4.10 Flow chart of the programmed control interface 35

Figure 5.1 DSP evaluation board for the project 37
Figure 5.2 Setup of the D/A signal generation experiment 38
Figure 5.3 Circuit diagram of the D/A signal generation experiment 39
Figure 5.4 GUI interface of signal generation 40
Figure 5.5 Code Composer window of the project test 41
Figure 5.6 Flow chart of the signal amplitude sampling algorithm 42
Figure 6.1 Speed of the table at different modes 45
Figure 6.2 Repeatability error curves of all axes at lower speed 47
Figure 6.3 Repeatability error curves of all axes at higher speed 47
Figure 6.4 Repeatability error curves of Y axis at different speeds 48
Figure 6.5 Repeatability error curves of X axis at different speeds 48
Figure 6.6 Repeatability error curves of Z axis at different speeds 49
Figure 6.7 Pop-up window for the direct control command test 51
Figure 6.8 Timer delay error of Windows® programming 52
Figure 6.9 Single time plot of the A/D sampling data buffer 54
Figure 6.10 FFT amplitude plot of the A/D sampling data buffer 55
Figure 6.11 Error of the FFT magnitude plot in the frequency domain 56
Figure 6.12 Error of the signal amplitude estimation 57
Figure 6.13 D/A signal generator verification with the DSP module 58
Figure 7.1 Developed open architecture controller 59
Figure 7.2 2½ axes stepping motor system control module 59
Figure 7.3 DC servomotor system control module 60
Figure 7.4 DSP module for machine tool monitoring 60

Figure 7.5 Diagram of the developed open architecture control system 61

Figure 7.6 Simultaneous operation of the user interfaces of the 2-axis DC servomotor controller and DSP 62

Figure 7.5 Diagram of the developed open architecture control system 61

Figure 7.6 Simultaneous operation of the user interfaces of the 2-axis DC servomotor controller and DSP 62

Chapter I

Introduction

Motion control technology has been widely used from manufacturing automation to space exploration. In most of motion control applications, manufacturers develop a customized control system that includes motors, amplifiers, microcontroller and software. This approach makes the widely used consumer products such as printers and fax machines cheap and reliable. On the other hand, customization creates serious problems for the users of large-scale industrial machines. As soon as the microprocessors of the controllers of these machines become obsolete in less than a year, owners start to feel the limitations of them relative to new models. Open architecture control systems have been proposed to design these machines by considering each component as an interchangeable module. Users can update the obsolete parts of these machines and add intelligent monitoring systems using the latest technology. In the machine tool area the need for modular systems has been particularly felt and very important steps have been taken. In this thesis open architecture control and monitoring systems will be mainly developed for machine tools; however, they can be easily implemented for non-destructive inspection systems that move sensors precisely in a given space.

Open architecture control systems [1][2] are very similar to conventional control systems. They include actuators, sensors, feedback devices, amplifiers, controllers, microcontrollers, software and human interfaces. However, all the components in the open architecture control system are carefully documented to let users easily replace, modify or update the components.

In 1987, the work of McMahon *et al.* [3] was published. They developed a generic computer hardware and software system for vacuum coating equipment. The system with open architecture value was structured on a Motorola 68000 based microcomputer. However, in the 1980s, technologies for an open architecture control system building were inadequate whether in hardware components or in software operating systems. There was little research in the 1980s on these systems.

In the 1990s, due to the rapid development of IC (Integrated Circuit) products, software environments and communication means, the development of open architecture control systems were facilitated. Nevertheless, the integrations of interdisciplinary knowledge into the design of such systems took much time; required teamwork, and demanded international standards. The current situation is the research on open architecture motion control systems is still in progress and there is very limited contribution from original equipment manufacturers (OEM).

Most of currently available publications [4-26] regarding the open architecture motion control research are searched from Engineering Index and reviewed. Previous research was mainly focused on the control of machine tools [4][8-11][15][20][23-26] or robots [5-7][12][14][16-19][21]. The searched publications are mostly listed in Table 1.1 and Table 1.2 concerning their specific research on the focused areas in a chronological order. The previous research is categorized and evaluated briefly.

In the previous research, when classified by hardware, the central platform was generally constructed on a standard PC and the external controller was often based on the DSP (Digital Signal Processor). For example, Erol *et al.* [10] proposed an integrated system which supported the combination of DSP boards and PC host computers for

distributed operation. The system also provided a subsystem for sensor-assisted machining. Chang *et al.* [13] presented a system which ran the user interface on a PC and used the DSP as the real-time hardware platform. Xie *et al.* [15] introduced an open CNC system based on ultra-precision machine tools in NUDT. The master was an industrial IPC386 and the slave was a DSP.

Table 1.1 Application into the machine tool area.

Publication Year	Paper Title
2001	New tooling mechanism for CNC lathes [4]
2001	Reconfigurable software for open architecture controllers [8]
2000	Hardware independent architecture for CNC machine tools reconfiguration [9]
2000	Open system architecture modular tool kit for motion and machining process control [10]
2000	Integration of process planning, monitoring and control in a machine tool environment [11]
1998	Computer numerical control (CNC) system for ultra-precision machine tools [15]
1997	Autonomously proficient CNC controller for high-performance machine tools based on an open architecture concept [20]
1995	Modeling and control of CNC machines using a PC-based open architecture controller [23]
1995	Open architecture testbed for real-time monitoring and control of machining processes [24]
1995	Open-architecture controller for die and mold machining [25]
1995	Simulation and implementation of an open architecture controller [26]

Table 1.2 Application into the robot area.

Publication Year	Paper Title
2001	Design and implementation of the robotic platform [5]
2001	Open system real time architecture and software design for robot control [6]
2001	A PC-based open robot control system: PC-ORC [7]
1999	Open structure multiprocessor robot controller [12]
1998	Control of a laboratory robot using an open DSP-based controller [14]
1998	Development and implementation of a NURBS curve motion interpolator [16]
1998	Open architecture for position and force control of robotic manipulators [17]
1998	Intelligent robot system architecture design and implementation [18]
1998	Open sensing architecture to autonomous mobile robots [19]
1997	New robot control system with open architecture [21]

In the previous research, when classified by software, the object-oriented development generally implemented high-level languages programming for software reusability to reconfigure the system for different applications. For example, in the system proposed by Erol *et al.* [10], the C language could be used for programming new functions and integrating them into the system. A simple scripting language could be applied for the setup of communication connections at run-time. The robotic platform

proposed by Loffler *et al.* [5] implemented a single programming language (C++). The robotic axis controller proposed by Baptista *et al.* [17] adopted the C++ language for the development of the driving kernel of software for supervisory, management and control.

The previous work also contributed certain other aspects for the system building. For example, Chang *et al.* [13] implemented the API (Application Programming Interface) to facilitate code development. Lopez-Orozco *et al.* [19] presented a system which allowed the connection of different types of sensors and sensor combinations. It saved users from creating a new structure for each sensorial system. Yamazaki *et al.* [20] proposed a new controller for high performance CNC machine tools which provided intelligent functions to ease less-skilled users. Zhou *et al.* [22] proposed to combine tasks with similar periods into a composite task for the betterment of system performances.

However, the progress of open architecture control system development also demands the work for easy duplication and transplantation acknowledged by more people to propel the acceptance and research growth of the system. In this study, two motion control systems were developed by considering the small-scale and large-scale machine tools, and a DSP based signal monitoring system was tested. The development time, cost, simplicity, reliability and performance of each application was evaluated.

The first stage of the study was to modify a miniature machine tool for the operation under computer control. The system consists of a microcomputer with graphical user interface software and a 2½ axes BASIC Stamp based stepping motor controller. The graphical interface program of the microcomputer communicates with the user and determines the required machining table motions. The microcomputer and the BASIC Stamp based motor controller communicate through the RS-232 port by using a

specially developed protocol. The BASIC Stamp based controller applies voltage to the proper coils of three stepping motors to create requested motions. The performance of the system was tested with a series of experiments.

The second stage of the study was to develop a large-scale motion control system which uses DC (Direct Current) servomotors. The control system was designed for Bridgeport Series 1 milling machines. It consists of microcomputer interfaces and a commercial servomotor controller, which is capable of creating two-axis circular and four-axis linear interpolations. The microcomputer interfaces could change the installation parameters of the numeric controller and reconfigure the numeric controller functions as well as manually navigate the numeric controller for the machining purpose via RS-232 communication. The developed Visual Basic user interface program of the system could be easily customized or updated for almost any motion control application.

The third stage of the study was to evaluate the capabilities of DSPs for the monitoring of sensory data. This information can be used either to control the motion, or to evaluate operating conditions. The system is mainly composed of a DSP module (based on the TMS320LF2407 core) and selected sampling algorithms. The usage of the parallel port with the D/A chip for the generation of analog signals is conducted. The parallel port can be used to move information between the controller and the board in future.

An open architecture motion control system with basic components was developed in the study and the system was tested under real world conditions. The evaluated DSP monitoring system will be used tightly with the control system for the construction of an advanced open architecture control system in the future.

The thesis is organized in the following order:

Chapter 2 discusses open architecture system basis, open architecture system development and the motion control system with openness.

Chapter 3 concentrates on the platform prepared for the 2½ axes stepping motor system control.

Chapter 4 concentrates on the platform prepared for DC servomotors system control.

Chapter 5 concentrates on the platform prepared for machining monitoring using DSPs.

Chapter 6 presents the results and discussion of the performances of three platforms developed.

Chapter 7 presents the integration of three platforms into the open architecture motion control system.

Chapter 8 draws conclusions of the open architecture motion control system development and gives future studies.

Chapter II

Open Architecture Control System

2.1 Open Architecture System Basis

Open architecture control system related studies increased in the 1990s. In Europe, OSACA (Open System Architecture for Controls within Automation Systems) was initiated in 1992 [27]. In Japan, the OSE (Open System Environment for Manufacturing) consortium was established in 1994 [27]. In Canada, the University of British Columbia has developed a user-friendly and modular tool kit for motion and machining process control [10].

Open architecture controllers allow the implementation of applications to run on a variety of platforms from multiple vendors, interoperate with other system applications and present a consistent style of interaction with users [27]. In other words, open architecture systems entitle users to add, replace, reconfigure or expand the standardized components according to the required functions and performances.

There are no strict definitions of the open architecture control systems. Currently, the definitions by Pritschow *et al.* [28] and Wright *et al.* [29] are referential. Based on them, the following four criteria may be used [27]:

(1) Portability of system on different platforms

(2) Interoperability of components with convenience

(3) Scalability of the system with user adaptation

(4) Interchangeability of modules without conflict

2.2 Open Architecture System Development

2.2.1 Hardware Consideration

The open architecture system does not require every controller to be the same. However, different hardware should be able to interact with each other conveniently and could be operated easily. Microcomputers, which use the PC-bus, microcontrollers (or microprocessors), DSPs or PLCs (Programmable Logic Controller) are good choices for the brain of the system. As long as the microcomputers use the PC-bus, they can be obtained from many different manufacturers and can be easily updated when better processors become available. According to the total cost and expected functions the optimal microcomputer should be selected.

2.2.2 Software Consideration

Currently most of the manufacturers use proprietary languages for their controllers. When using the G-language, a specialized postprocessor should be used to be sure about the outcome. To shorten the development time, to reduce the cost and to find low cost program routines, the use of high-level structured languages has initiated. For example, by using the Visual Basic language and the ActiveX control, controllers could be quickly developed. In more demanding applications, the Visual C++ DLL library may also be used for similar purposes. OMAC (Open Modular Architecture Controllers) Users Group defined API specifications recently [27] to be able to expand the program and to add new components.

2.2.3 Communication Consideration

A user-friendly and reconfigurable architecture of the system should be reliable and have the necessary speed to operate the GUI (Graphic User Interface) at acceptable

performance levels. In the background, all the functions of the system and the monitoring of sensors should be performed without creating any noticeable delay. RS-232 and RS-485 protocols have been widely used in the last decade because of their simplicity. For the future, USB (Universal Serial Bus) protocol is expected to find many applications since its speed and the automatic creation of multiple channels will satisfy the needs of future applications. In addition, LAN (Local Area Network) connection may play a role in sophisticated systems [30][31]. The integration of the open architecture control systems with the environment by using wireless communication and fiber optics will be inevitable in the next decade.

2.3 Motion Control Systems with Openness

Generally, speed and position controls are the most important issues in the controller development. Speed variations and the positional accuracy of the system are directly related to the computational algorithm and the hardware, and they determine the quality of the controller. In the machine tool industry, linear and circular interpolation algorithms are widely used. In addition, many manufacturers add implicit and complex curve interpolation routines to their systems. The acceleration and deceleration before or after interpolation also influence machining accuracy [32]. Consequently, it is hard to develop or modify the software of the controllers.

The development of modular motion control software helps developers and manufacturers. Instead of a proprietary system known by only a few people, a modular open architecture system can be easily and quickly modified for different applications. Pressure from customers, short development time, low cost and improved reliability of

open architecture systems are expected to motivate many manufacturers to implement this approach in future.

The brain of a small motion control system with openness can de designed by using the following blocks:

1. A user-friendly HMI (Human Machine Interface) which operates on a microcomputer. The interface should allow the user to input the part program, display the tool path and animate machine tool motions. The controller should also be able to read part programs in various formats.
2. Actuator controllers, which communicate with the HMI, receive the instructions one by one, and operate the actuators. The speed, accuracy and reliability of this controller are very important. Most of the time, controllers are programmed by using the assembly language to reach to desired speeds.
3. The intermediate link between the HMI and actuator controllers. The handling of feedback device signals, the detection of new I/O add-ins or I/O removals, the determination of the component priority, the evaluation of the hardware property correctness, the empowering of data buffer for network, the monitoring of the process sensor signals and the safe operation of the system in case of failure or emergency will be the responsibility of this section. Briefly, the intermediate link is expected to act as a coordinator between actuator controllers and the HMI.

The listed three components of the open architecture control system can be easily modified, upgraded or removed.

Chapter III

Control of a 2½ Axes Stepping Motor System

Stepping motors are widely used in consumer products to create affordable precise motion control systems. Fax machines, printers and electronic typewriters are built by using them. Since stepping motors turn with small steps according to the applied voltage to the coils of them, it is not necessary to use any sensor for feedback. Similar control systems can also be used in industrial motion control applications as long as the necessary torque is within the reach of commercial stepping motors and amplifiers. In this study a controller based on stepping motors was developed and used to automate a miniature milling machine. The hardware of the developed system includes a microcomputer, a BASIC Stamp based microcontroller, three STH-55D246 stepper motors [33] and a miniature vertical milling machine from the CHUNG HSIWH INDUSTRIAL Co., LTD [34]. The microcomputer runs the GUI program. Once the microcomputer determines necessary motions, information is sent to the microcontroller through the RS-232 port. Both units communicate by using a specially prepared protocol for this project. The microcomputer tells the controller how many steps each motor should be turned, the need for synchronization and the required feed rate. The controller is developed with an easy-to-plug socket that could accommodate the BASIC Stamp [35]. The controller applies voltage to the windings of three step motors with proper order to move the machining table. Two of the motors, which create the motions in the horizontal plane, are synchronized. The plane motion is planned through the GUI and the controller manipulates the step motors in both the horizontal plane and the vertical direction.

The controller and the graphic interface of the microcomputer are presented in the following two sections. The whole system, including the GUI, the controller and the milling machine, is shown as Figure 3.1.

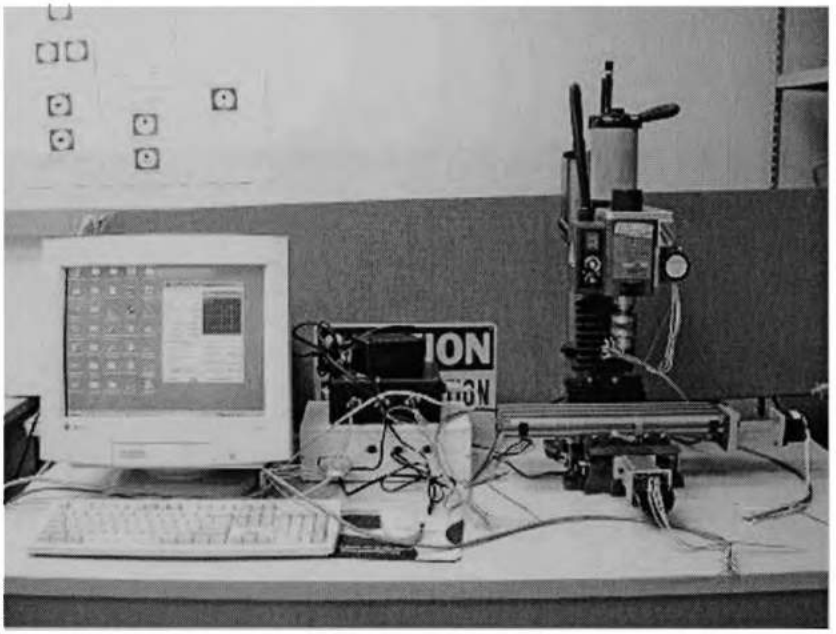

Figure 3.1 Proposed 2½ axes stepping motor control system.

3.1 Controller

The picture of the controller and the schematic diagram of the complete system are presented in Figure 3.2 and Figure 3.3 respectively. The controller allows the convenient downloading of BASIC Stamp programs and meanwhile the controller could be set to communicate with the end-user GUI. Figure 3.4 depicts a brief connection of the coils of one stepping motor to one I/O port of the BASIC Stamp. For every motor totally four TIP120 Darlington transistors are used since the motion driving design adopts full

step control. Supplied current and voltage are regulated to meet the electrical performances of the stepping motors so they could function in normal status. The connection of other motors coils to the machine tool system is similar with the reference of Figure 3.4.

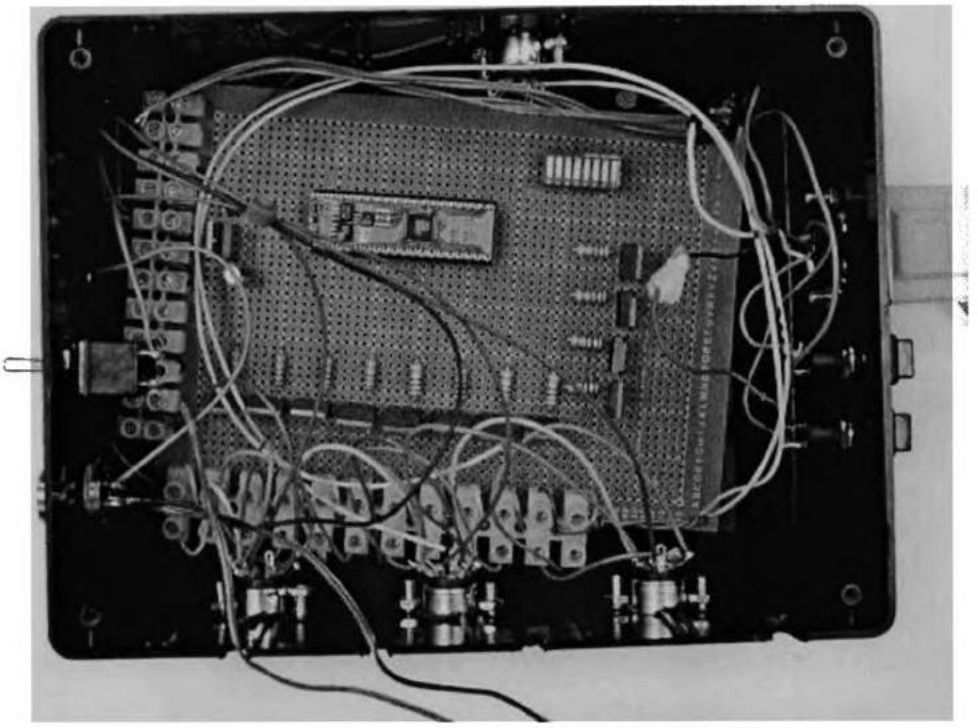

Figure 3.2 Developed BASIC-Stamp based controller.

The communication of the PC and the controller is presented in Figure 3.5. The connection of DSR to RTS and the link of DTR to ATN, which are used for the sense of COM ports and the reset of the microcontroller for motor control algorithm downloading, should be manually disabled to run the system GUI.

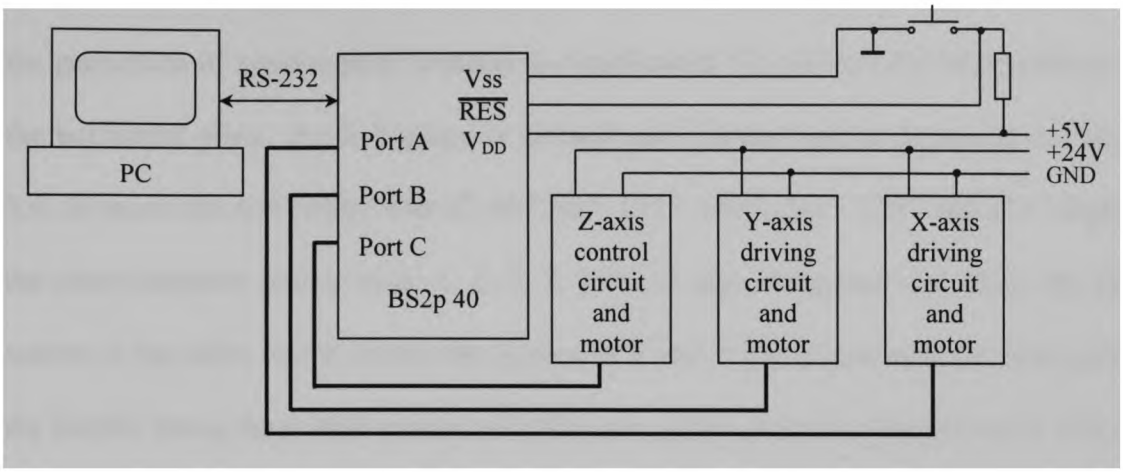

Figure 3.3 Diagram of the control system.

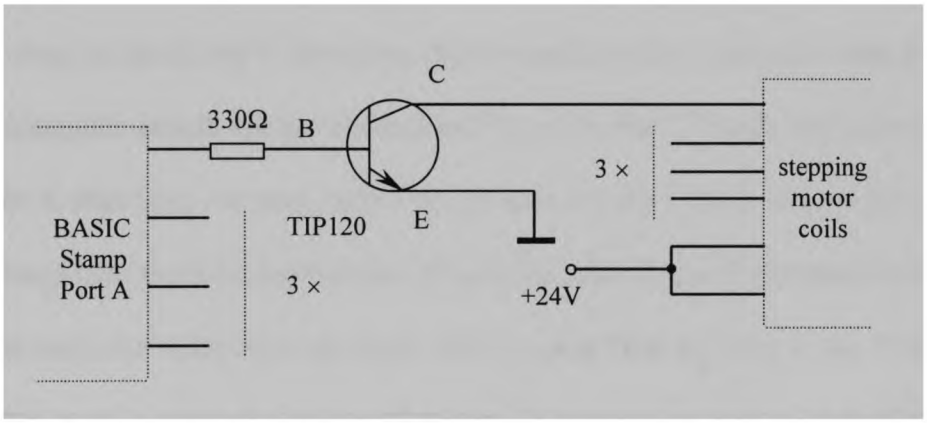

Figure 3.4 Diagram of the stepping motor control connection (totally 4×TIP120).

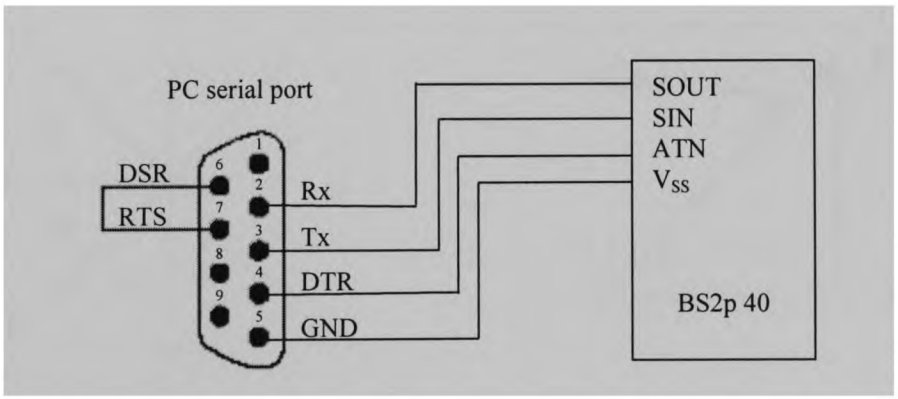

Figure 3.5 Serial communication link between the PC and the BASIC Stamp [36].

Since the BASIC Stamp is presently much slower than microprocessors in speed, the generation of synchronized motions is complicated. To perform the interpolations in the horizontal plane, the X-Y plane is divided into sixteen regions presented in Figure 3.6. To move the tool (table) with 0^{o}, 45^{o}, 90^{o}, 135^{o}, 180^{o}, 225^{o}, 270^{o} and 315^{o} angles, the microcomputer selects mode 1, 3, 5, 7, 9, 11,13 and 15 respectively. Since the lead screws of the table, which creates the motion in X and Y directions, have the same pitch, the BASIC Stamp turns both motors with the same speed. If the tool should move with an arbitrary angle outside of the values as listed above, the microcomputer tells the microcontroller which mode should be selected for interpolations and what the ratio is for the motor steps in the X and Y directions. For example, to move the table with 10^{o} angle, the microcomputer should ask the microcontroller to use the 2^{nd} mode and after every six steps in the X direction, one step should be generated in the Y direction. In this case, the BASIC Stamp uses the same instruction to move the table in the X direction by six steps. After these steps are completed, the table will be moved for one step in the Y direction. On the other hand, to move the table with 80^{o} angle, the strategy for mode 4 will be used.

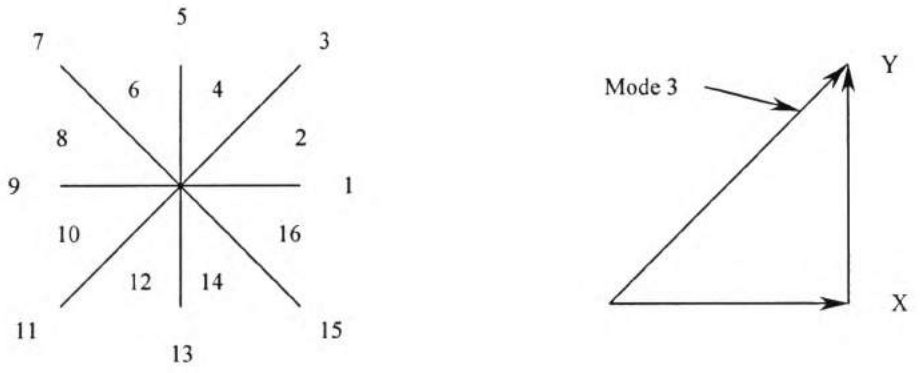

Figure 3.6 Distribution of predetermined regions and the region 3 table motion.

In this case, the first six steps will be created in the Y direction. The table will then be moved by one step in the X direction. The inertia of the table and the workpiece allows the creation of smooth moves when the above strategy is followed. Users do not notice any jerks or vibrations when the table or the head moves.

3.2 Software

3.2.1 Graphic Interface of the Microcomputer

The microcomputer communicates with the user through a user-friendly graphic interface program. This program is prepared by using Visual Basic 6.0 and consists of two modules: The hardware independent graphic interface and the hardware adopted information exchange module.

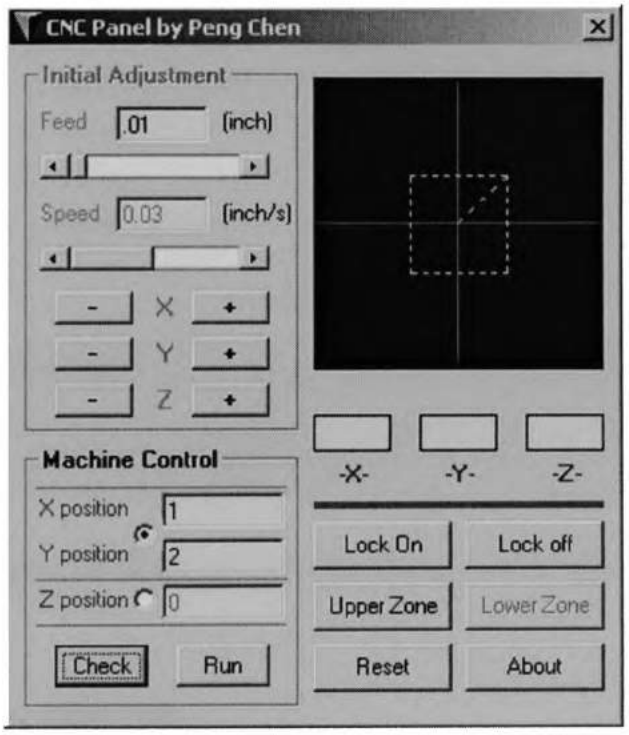

Figure 3.7 Graphical user interface (planning stage).

The screenshots of the graphic interface module are presented in Figure 3.7 and Figure 3.8. This module shows all the possible options, the user's selections and the motions of the table on the screen. Users should select the feed rate and speed. Later the origin of the machining workspace is adjusted. The graphic area of the program represents the workspace. Users are allowed to move the table with direct mouse clicks on buttons. In critical cases, users can first inspect the next position of the tool on the graphical screen before they actually move the tool. The interface inspects users' selections to avoid motion errors.

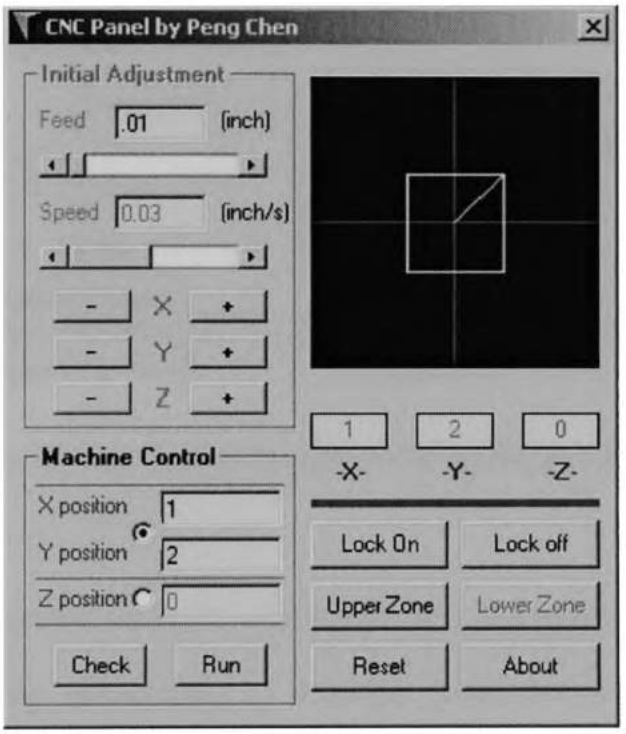

Figure 3.8 Graphical user interface (execution stage).

3.2.2 Microcontroller Algorithm

Once the cutting operation is decided, the information exchange module takes it

over. The algorithm of this module is presented in Figure 3.9. It first determines the region (mode), total steps, the interpolation scale and the cutting speed. The table motion limits and the maximum allowable speed are determined by considering the envelope of the workspace and the speed of the components including the BASIC Stamp and stepping motors. After calculations are completed, information is sent to the BASIC Stamp based stepping motor controller by using the RS-232 port at 9600 b/s.

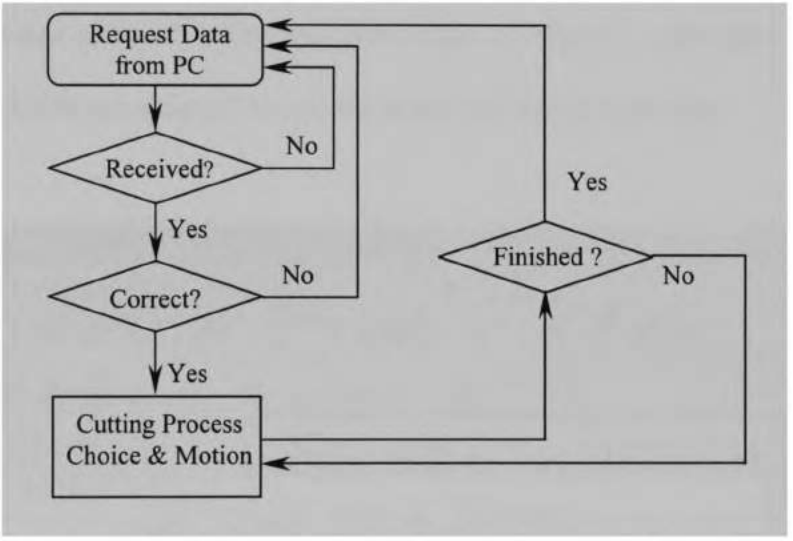

Figure 3.9 Algorithm of the information exchange module.

Since the algorithm for the microcontroller part is programmed in the P-BASIC language [36][37], the downloading of compiled codes into the microcontroller is necessary for the system at the beginning. The following graph (Figure 3.10) shows the compilation interface that is ready to download the user-coded program into the microcontroller through RS-232 communication. To enable the downloading, users are expected to press two physical buttons next to the communication cable as could be seen

in Figure 3.2. This is to connect DTR to RTS and also to enable the ATN connection, which allows the BASIC Stamp to check the existence of the microcontroller downloading connection and to test necessary signals. As long as the compilation process is correct, once the user presses the filled triangle button the algorithm would be downloaded. To check if the algorithm exceeds the memory capacity of the microcontroller before sending down the program, a capability of the compiler could be applied. Figure 3.11 depicts part of the memory occupation of a tested algorithm for this system. This helps users to watch the percentage of memory utilization and how the EEPROM and RAM are arranged to accommodate the above algorithm.

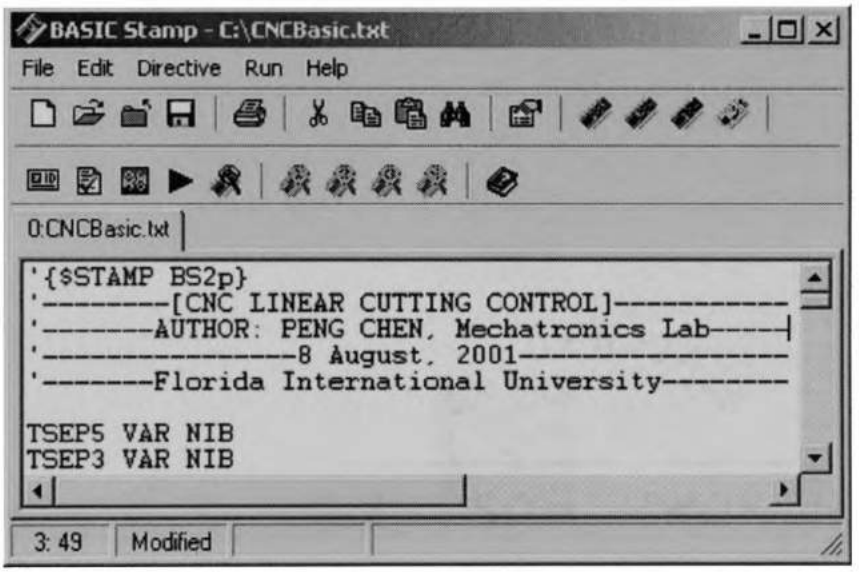

Figure 3.10 Compilation window of the control algorithm.

After the downloading process, users need to release both buttons next to the communication connector (Figure 3.2) so that the ATN signal and the connection of DTR

to RTS are disabled. From this on, the microcontroller system is separated from the compiler system and is able to communicate with the control interface (GUI) without the influence by the compiler system. Obviously, by means of the BASIC Stamp compiler and the developed hardware and software, it is very convenient to change the system algorithm for a new application.

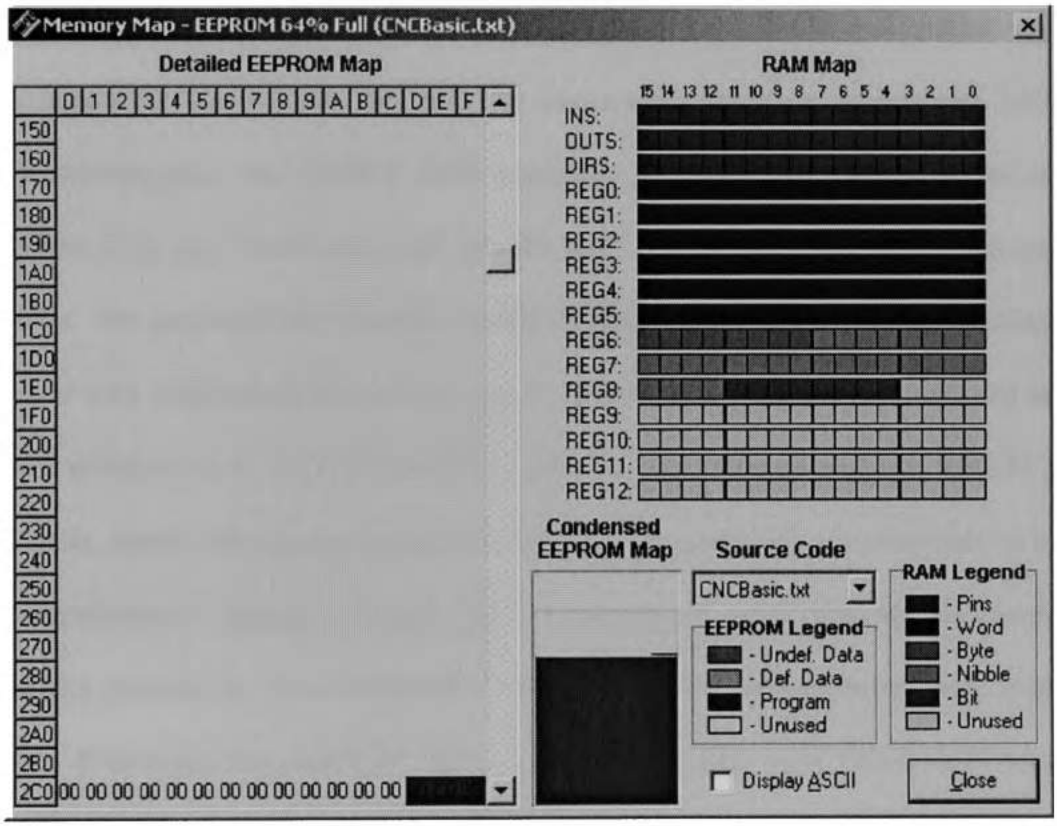

Figure 3.11 Memory consumption window of the control algorithm.

Chapter IV

Control of a DC Servomotors System

Stepping motors cannot create large torques since their rotors are prepared by using permanent magnets. DC servomotors are used in these applications. Since the speed of the DC motor depends on the voltage and the load, its rotations should be carefully monitored by using feedback devices. In this study an open architecture control system was prepared for the DC servomotors. The hardware of the developed system includes one microcomputer, one CYBER 4000 numerical controller [38], two RTS DC servo amplifiers [39], two transformers and two RX330C servomotors [40] with resolvers and encoders. The graphical user interface is run on the microcomputer. The microcomputer GUI provides communication with the user, determines necessary motions and sends relevant information to the CYBER 4000 numerical controller through the RS-232 port. Both units, namely the microcomputer and the numeric controller, communicate by using the manufacturer's special protocol. The microcomputer could read all the necessary machining parameters from standard G-CODE files and download instructions into the CYBER 4000 numerical controller. After resetting the system, the CYBER 4000 numeric controller supplies reference voltage to RTS amplifiers and monitors the positions of the DC motors from the encoder signals. RTS amplifiers provide the necessary voltage to the DC motors to turn them at the desired speed and monitor the resolver signal. The CYBER 4000 numerical controller is capable to work with the conventional G-code and to create linear and circular interpolations [38] at desired feed rates.

Conventional controllers like CYBER 4000 are very convenient for controlling

motor motions if simple control acts are expected from the system; however, they do not have graphical user interfaces, cannot monitor sensory signals and cannot be adapted for specific applications. To make the system user-friendly and easily adaptable, and to add future monitoring capability, a Visual Basic program was developed. The GUI of the program communicates with users and sends instructions to the CYBER 4000 controller.

The numeric controller and the software are presented in the following two sections. The CYBER 4000 numeric controller (upper right), RTS DC servo amplifiers (upper left) and transformers (below) of the control system are presented in Figure 4.1. The diagram of the control system of one DC servomotor is presented in Figure 4.2.

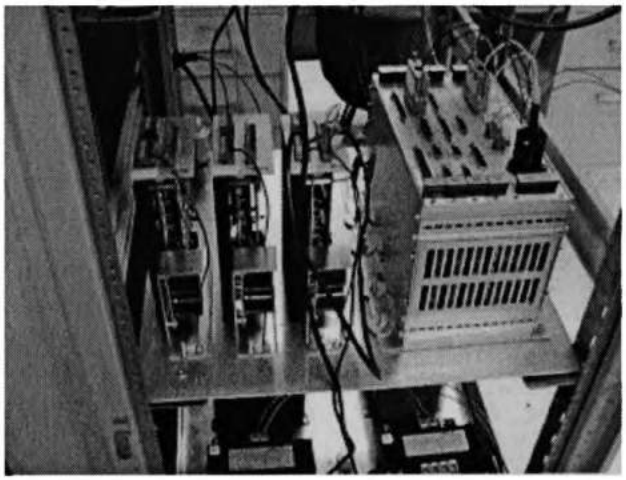

Figure 4.1 Assembly picture of part of the constructed system.

4.1 Controller

The CYBER 4000 numeric controller may control up to four DC servomotors. The serial communication link between the CPU card of the CYBER 4000 controller and the serial port at the microcomputer is presented in Figure 4.3.

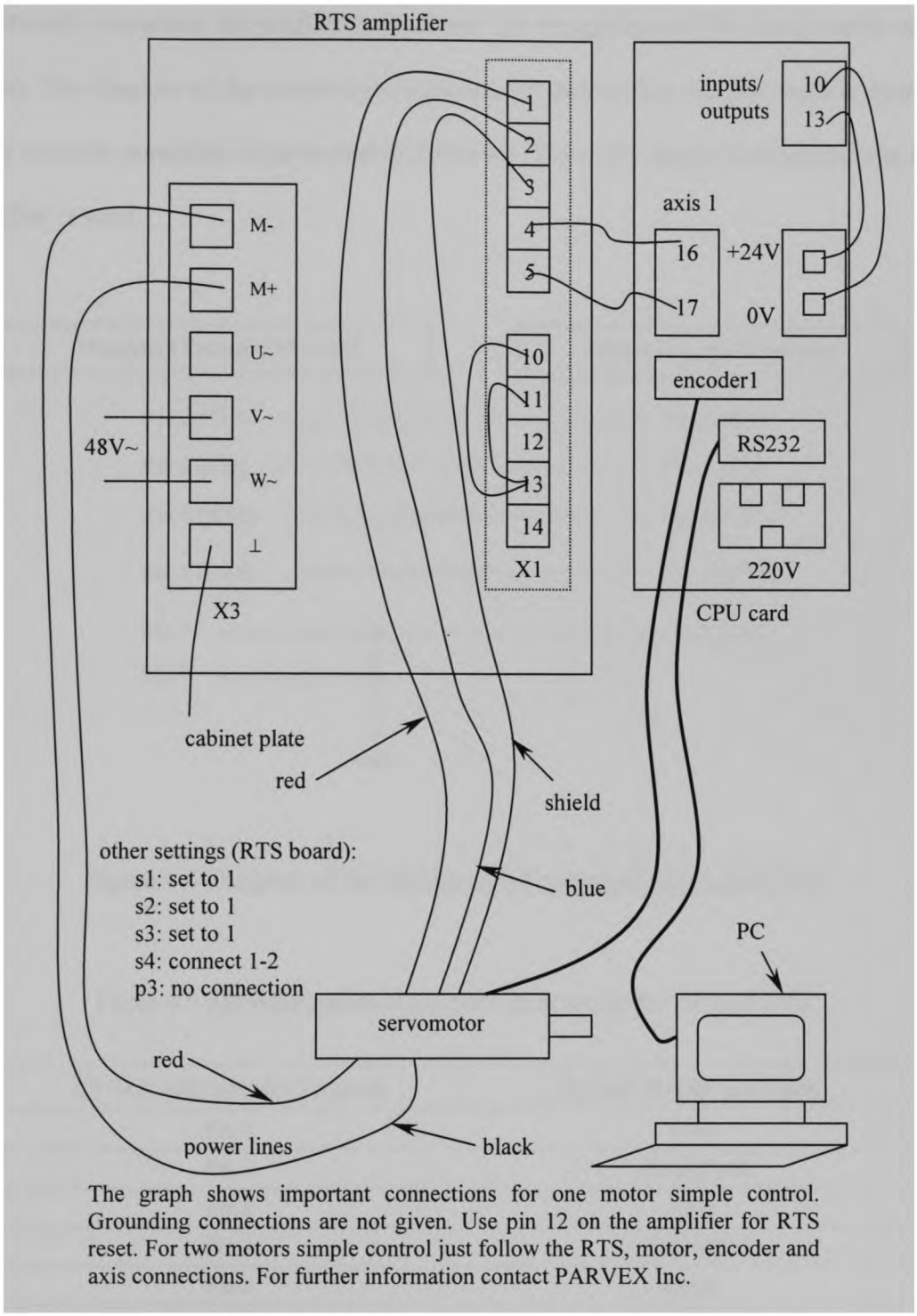

Figure 4.2 Sketch map for one servomotor control connection [38][39].

The proper wiring of the motor encoder with the signal terminal on the CPU card is extremely important for position control and the recognition of the components of the system. The diagram of the connections between the servomotor and the encoder terminal of the numeric controller is presented in Table 4.1. For every single servomotor one RTS amplifier is used.

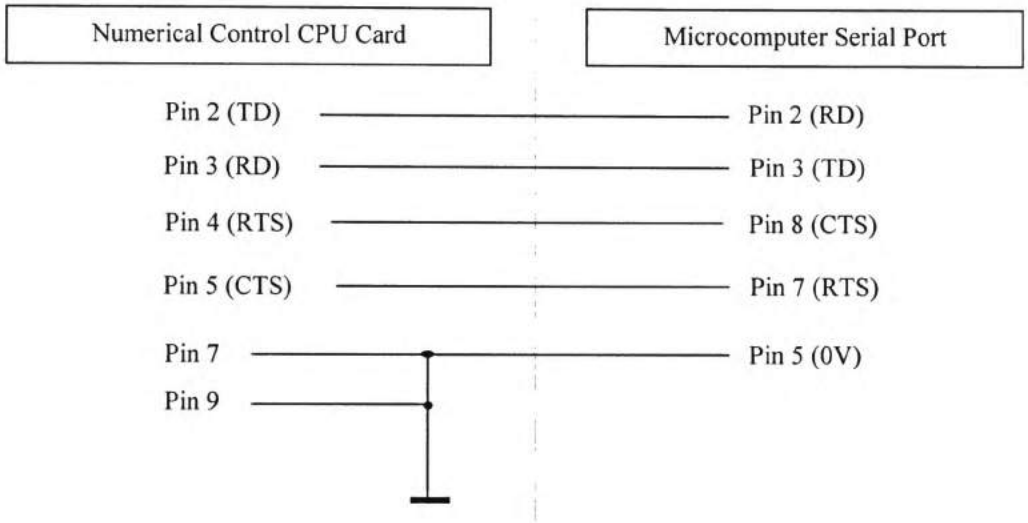

Figure 4.3 Diagram of the system serial communication link [38].

Table 4.1 Encoder connections for one servomotor control [38].

CYBER 4000 Encoder Terminal	Servomotor Encoder Cable
Pin 7	White
Pin 3	White/Black
Pin 8	Blue
Pin 4	Blue/White
Pin 6	Green
Pin 2	Green/White
Pin 5	Red
Pin 9	Black

The specifications of adopted servo amplifiers and servomotors are respectively listed in Table 4.2 and Table 4.3.

Table 4.2 Specifications of the servo amplifier [39].

Type: RTS 10/20-60			
Input voltage range (AC)	30~53 V	Resistive braking power	30W
Nominal input voltage (AC)	48 ± 10% V	Dimensions - Panel mount (1 phase)	180×65×212 (H×L×P)
Nominal DC output voltage	60V	Operating temperature	Normally 0~40°C
Nominal output current	10 A	Chopping frequency	17 kHz current
Peak current limit (2 s clamp)	20 A	Tachogenerator maximum voltage	100 V at input
Instantaneous power (4% of cycle)	800 W	Speed range	1~10000 with tachogenerator

Table 4.3 Specifications of the servomotor [40].

Type: RX330C			
Tacho generator voltage gradient	6 V/1000 rpm	Rated Speed	2900 rpm
Holding brake torque (20°C)	13.3 lb-in	Peak current at low speed	26 A
Encoder standard resolution	500	Torque constant (25°C)	1.5 lb-in/A
Continuous torque at low speed	13.6 lb-in	Thrust load	51.71 lbf
Continuous current at low speed	9.4 A	Radial load (at half shaft length)	112.4 lbf
Rated voltage	59 V	Armature resistance (25°C)	0.45 Ω

Once all the necessary connections are completed and all the components are found operational, the system could be reset and desired motions can be generated. The controller supplies different voltages and currents according to the needs of attached components. Before running the system, proper parameters should be given to the program. Setting up the parameters for the program is presented in Figure 4.4.

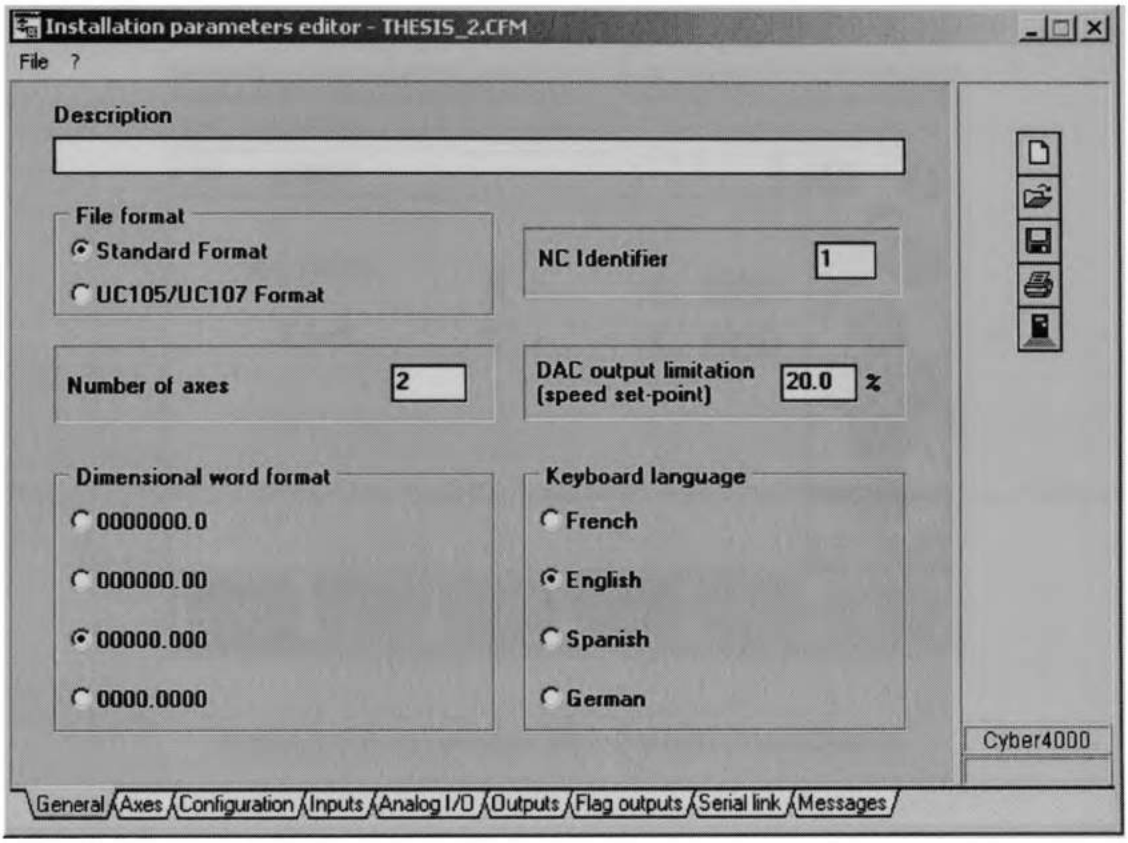

Figure 4.4 System parameter installation process.

PARVEX Motion Explorer [41] allows users to configure each axis separately. It displays the motor position, display format, position display unit, the acknowledgment mode from the encoder, original speed limit, maximum speed, running acceleration and

other necessary steps. The digital and analog components of the controller follow the given instructions precisely and quickly.

To test if the hardware setup is correct or not, the supplied software is applicable (the software is also applicable for motion control). The interface in Figure 4.5 presents a system-resetting communication message after both hardware and software are configured.

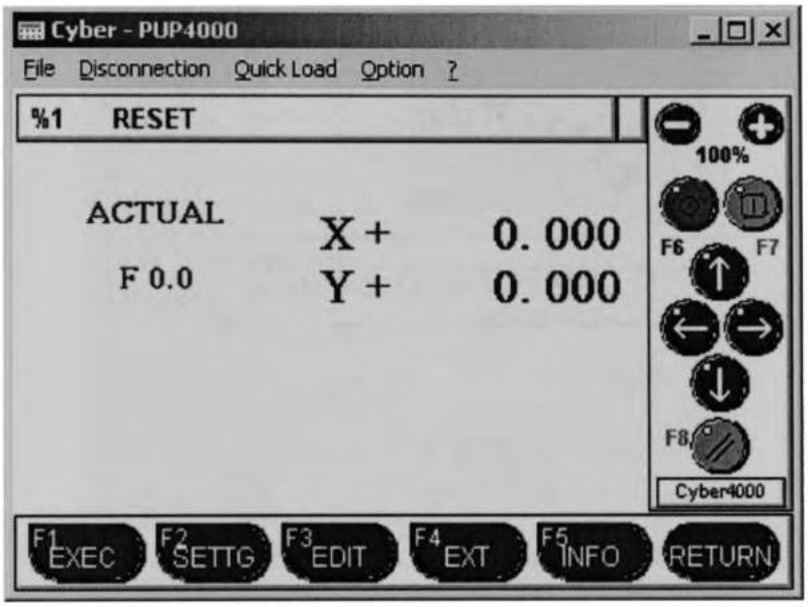

Figure 4.5 Reset window after the correct installation.

4.2 Software

4.2.1 Graphic Interface of the Microcomputer

Due to the design of the controller, the original settings of the system are hard to be completed without the supplied software or it takes a long time to make proper settings with user-developed programs. The definition of the system communication

mode is set via the PARVEX Motion Explorer. The developed software for the microcomputer (Figure 4.6) mainly communicates with the user and gives the instructions to the controller. Although, the developed program is not as powerful as the manufacturer's program, it can be adapted to different applications more easily.

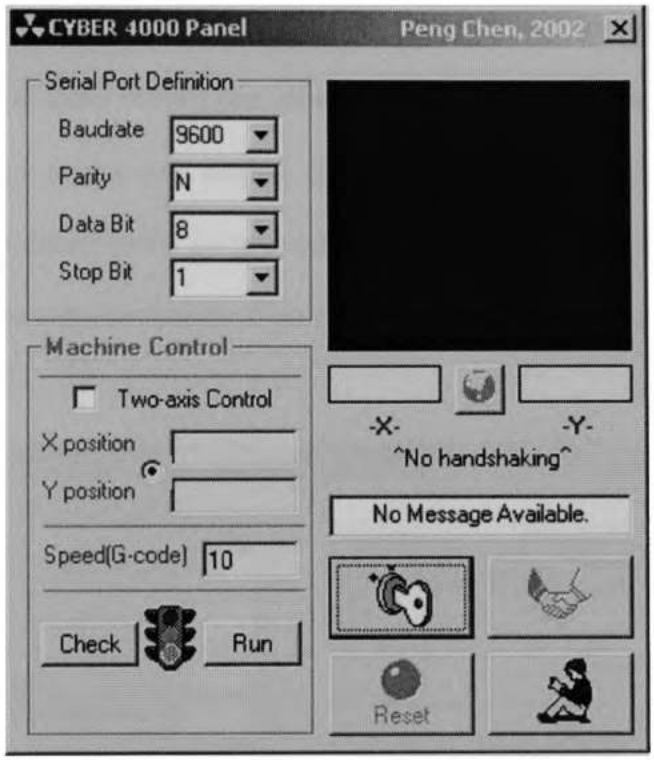

Figure 4.6 Developed manual navigation software interface.

The user interface first shows the RS-232 serial communication configuration will be used. Once started, the GUI interface, by using the sound and a flashing icon, notifies the user to decide whether to establish the link or not. If the RS-232 cable is not connected the GUI can warn the user to check until the cable is well connected. The handshaking between the microcomputer and the controller is the second step. The

purpose is to check if the controller is working properly and if the controller is ready for receiving the control commands from the microcomputer. If the handshaking fails the program could warn the user and also enable the user to repeatedly establish the communication link until the handshaking succeeds. Once the controller confirms that it is ready, software (Figure 4.7) is enabled for users to plan the motion and run the DC motors. The program can sense if there is one motor or two motors connected to the numeric controller. If there is only one motor connected, the interface could change the motion control status to one-axis control (Figure 4.7). However, once both motors are connected all the necessary items for two-axis control could be activated without program restarting. Hence, the software has the capability to identify motor removals or add-ins.

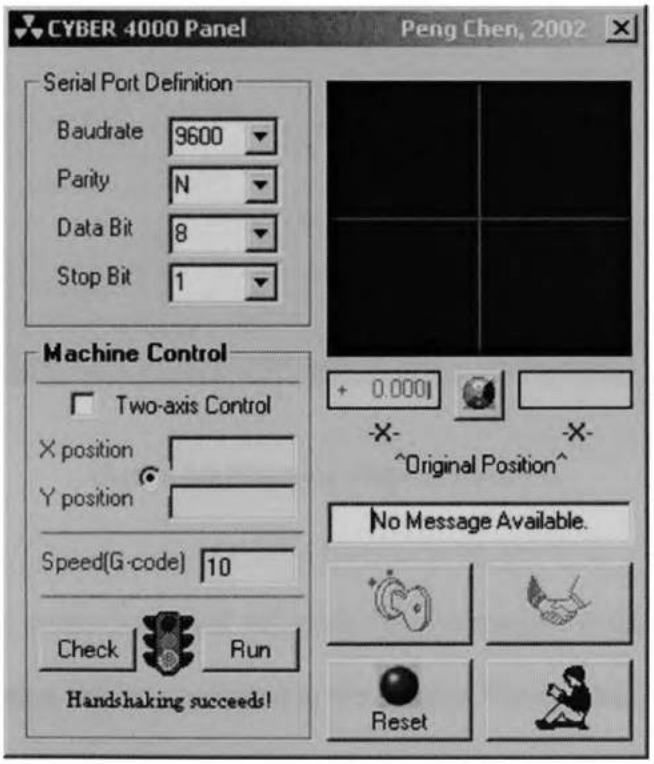

Figure 4.7 Successful handshaking of the microcomputer and the controller.

The control of motors could be planned with the GUI (Figure 4.8). Users should follow the manufacturer's manual [38] for the correct input format. As long as the input format is correct and the running of motors is required, the interface automatically converts the user inputs into a format which could be recognized by the controller and sends the control instruction to the controller.

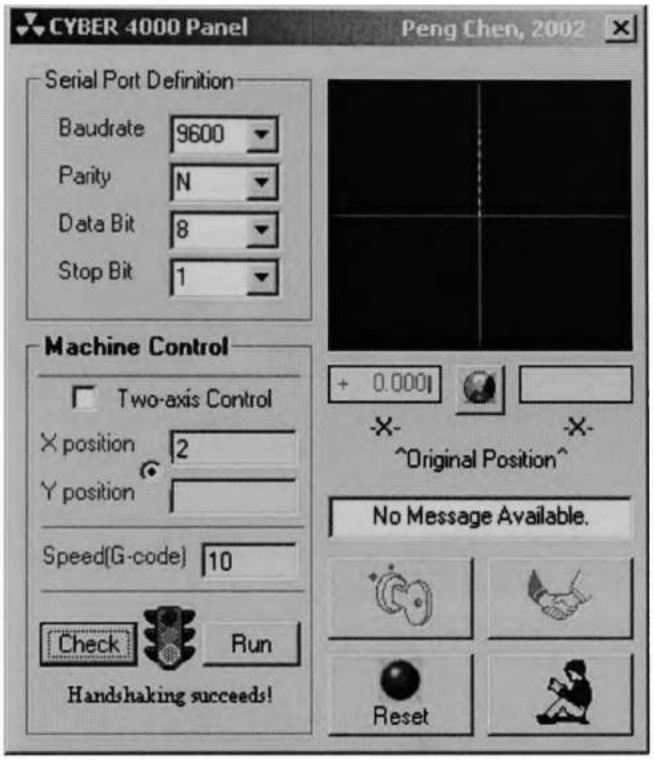

Figure 4.8 Planning stage of the GUI.

The numeric controller could acknowledge the correct motion control command and the relevant motion can be confirmed in the graphic window and the text box (Figure 4.9). To have a visual check of the motion, the starting position of the encoder reading is buffered and displayed on the screen. To inspect the encoder reading after sending out the

motion command, users should press the *Run* button again. The original convex icon with a question mark will change to be concave (Figure 4.9) and the current encoder position (if it is convex then not) is displayed if the controller completes executing the given instruction. Under this condition, the GUI is ready for the next motion control. Otherwise, the previous encoder position will be displayed and the graph window will remain unchanged.

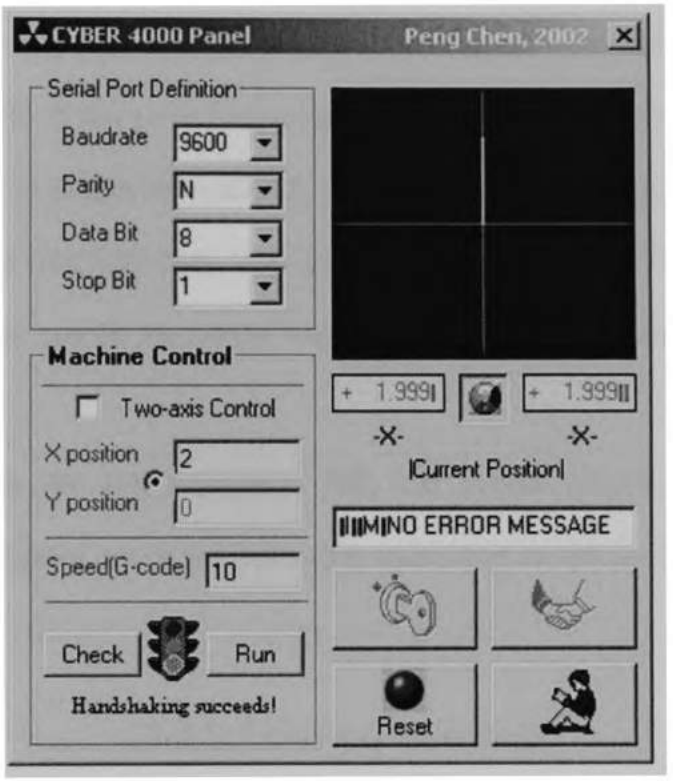

Figure 4.9 Running stage of the GUI.

In order to make the GUI robust, the *Reset* button is considered and the button function enables users to re-establish the link with the controller in case of communication faults at any time. In addition, if the fault relay is open in case of

emergency, the program could respond and the encoder display will be disabled. A warning message will be given right away.

4.2.2 Microcomputer Interface Algorithm

To guarantee the proper operation of the interface program, the algorithm in Figure 4.10 is followed. To minimize the motion error, the interface affords control check before control running. The program automatically transforms the interface control input into acceptable formats to exchange with the numeric controller.

The control command should be given by using the manufacturer's instructions and data formats [38]. If there were any error, a related error message will be sent back to the microcomputer. A proper instruction to turn the motors should be given according to the formats, such as *G36 2000* [38]. This instruction creates continuous movement in the X direction for 2000 mm (if the system was set up for metric operation with the unit as millimeter).

To establish the communication link, the system should also be properly configured. The program sends the following instruction to set up the communication [38]:

Chr$(5) + Chr$(67) + Chr$(13)

The numeric controller sends back a responsive message to acknowledge that an instruction is received. This message indicates the axis name, position sign and position values [38]:

<ACK> C X+12345.999Y- 0.123 <CR> (the value here is arbitrary)

The Windows®-based program tries to detect the character *C* in the microcomputer serial port input buffer. If the character *C* is not found in the acknowledgement message, that means proper link is not established between the microcomputer and the numerical controller. Alike, the connected motor number could be found.

When the link fails at the start of the program, the interface alarms the user about the failure until acknowledged. On the contrary, if the link is handled the interface software activates relevant timers and buttons so that the serial port input and output buffers could be available to store the messages and transfer the instructions. Part of the message is not necessary for the encoder position display and it will be filtered. The algorithm displays the motor position in the correct format after all the unnecessary characters are removed. In addition, some important messages such as a relay is open (means there may be some serious error and the relay is cut off) are displayed within a text box to alert users about the malfunction and let them reset the system manually.

In order to notify the numeric controller that the software is going to send out motion commands manually (or one instruction per operation), the interface software sends the following instruction [38]:

$$Chr\$(5) + Chr\$(73) + Chr\$(13)$$

If the numeric controller sends back *21 73 13* [38], the control command will fail and the software will not be effective for the control. This response indicates that there is no such a command in the queue of the numeric controller and there will not be motion. On the other hand, if the numerical controller is ready, the response message to the

microcomputer will include the *ACK* ASCII characters. The information exchange protocol of the software is presented in Figure 4.10.

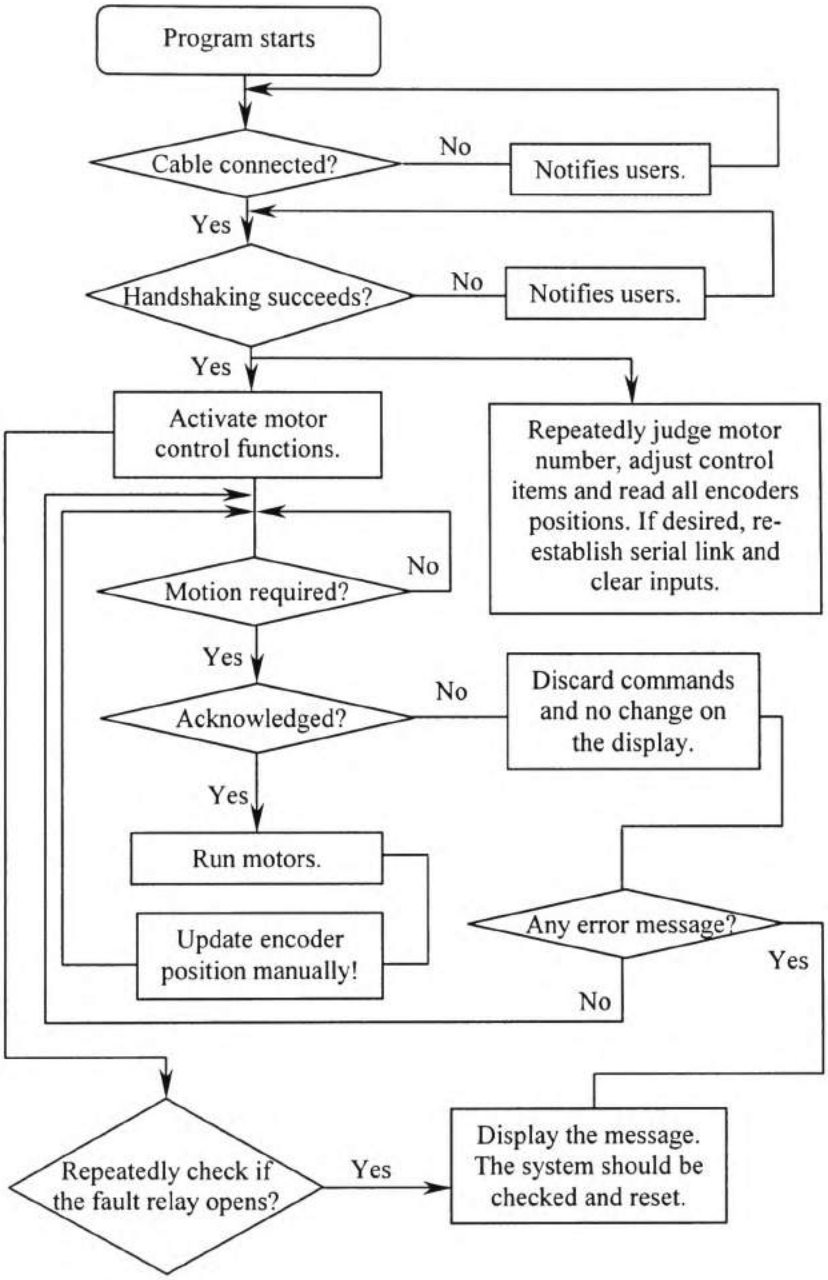

Figure 4.10 Flow chart of the programmed control interface.

Chapter V

DSP Based Machine Tool Monitoring System

The processing of the analog sound and image data is very expensive. Noise increases at each stage and the transmission of the signal requires very large bandwidth. The speed, cost and program languages of conventional microcontrollers are not suitable for these applications. Special purpose digital circuits, with very fast sampling and processing capabilities, are developed to be included in Digital Signal Processors (DSP). Recently, DSPs have found many applications [42-46] including noise elimination, signal analysis and graphic compression. It is noticeable that signal-processing applications, which take only several nanoseconds to run by using the DSPs, would take several milliseconds when the conventional microcontrollers or microprocessors are used. During the analysis and processing of particularly high frequency signals, the usage of DSPs is almost inevitable. Recently, DSPs have found their applications at the health monitoring for machines, displacement control and vibration compensation. Since the DSPs were not convenient for conventional control applications, Microchip Inc. recently introduced the dsPIC^{TM} digital signal controller that combines the control advantages of a high-performance 16-bit microcontroller with the high computation speed of a fully implemented digital signal processor (DSP) [47]. To apply the high speed processing capability of DSPs, a TMS320LF2407PGEA DSP (Texas Instruments Inc.) based evaluation board (eZdsp^{TM} module [48]) manufactured by Spectrum Digital Inc. is used in this application. The picture of the eZdsp^{TM}TMS320LF2407 DSP evaluation board is presented in Figure 5.1.

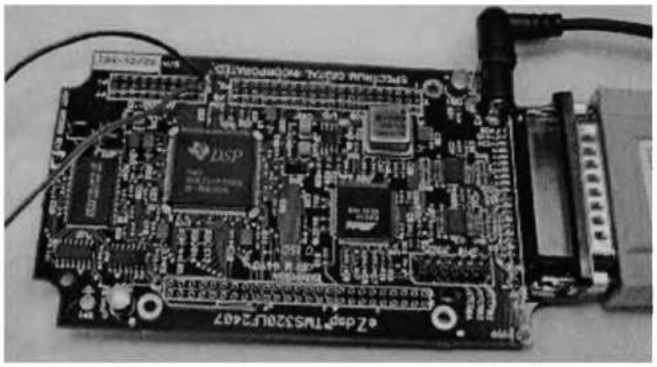

Figure 5.1 DSP evaluation board for the project

The DSP evaluation board was linked to the host PC via the parallel port for programming and data communications. The DSP module has up to16 A/D channels and 40 I/O ports [49] for program execution, signal acquisition (with on-chip timer) and the signal output after the necessary computational process is completed. Through the Code Composer [50], programs can be prepared by using the assembly or/and C language, compiled, debugged, built and downloaded into the Flash memory if the jumpers on the board are properly set. In addition, the DSP board could be set to be a stand-alone product for real applications.

The preparation and programming [51] of the evaluation board and the developed Visual Basic program for the system test are discussed in the following sections.

5.1 Analog Signal Generation by Using a Specialized D/A Chip

Sensors generate analog signals. A flexible signal generator is very useful to simulate complex sensory signals which cannot be generated with conventional

commercial function generators. For example, to take one period of the sensory signal Diehl *et al.* [52] used an electro-optical vibration sensor and repeated it continuously. To evaluate the performance of a signal analyzer a specialized function generator is very beneficial. To generate very complex signals and to be able to operate at high frequencies, the PC parallel port was used to communicate between the microcomputer and the D/A chip. The developed system is presented in Figure 5.2.

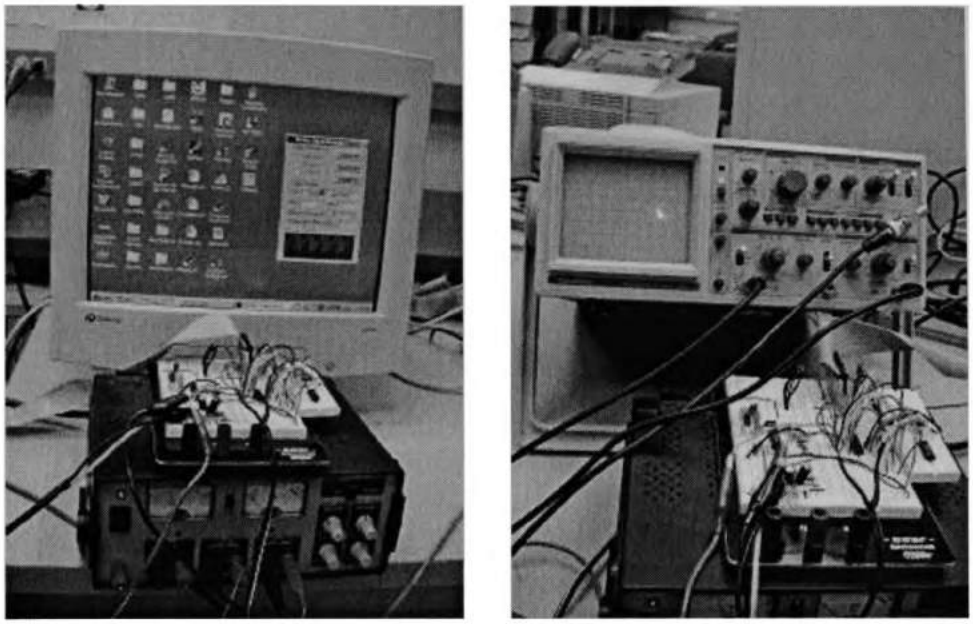

Figure 5.2 Setup of the D/A signal generation experiment.

By using the AD7224 [53] D/A conversion chip, analog signals were generated. This chip is an 8-bit DAC chip with output amplifiers. The standard parallel port on a microcomputer was connected to the DAC chip. The diagram of the electrical connections is presented in Figure 5.3. A BK PRECISION triple output DC power supply 1651 was used as the adjustable referential voltage to change the amplitude scale of the

generated signal. The AD7224 was set to the transparent output with no register latches enabled. The analog output voltage range of the circuit below is approximately 0 V~ (V_{REF}×255/256).

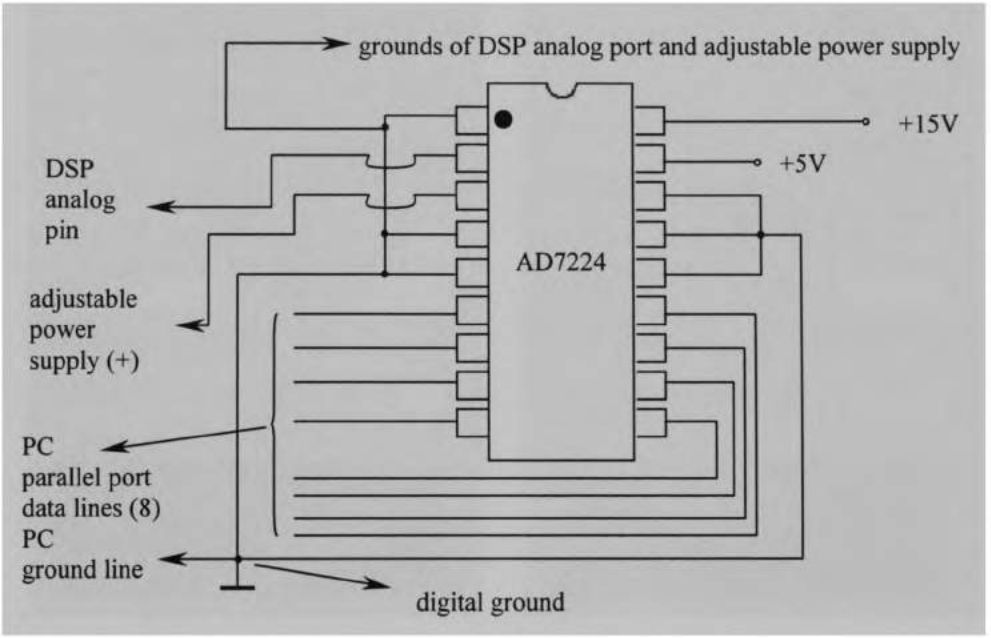

Figure 5.3 Circuit diagram of the D/A signal generation experiment.

The developed Visual Basic signal generation program (Figure 5.4) sends out the scaled digital values through the parallel port. This program has a graphical user interface and a specially prepared support program (DLL file). The program controls the signal output delay, signal scale ratio, peak value recording and referential voltage of the external D/A chip. In addition, the software provides a continuous plot of the data output (single data output is also possible). Since the chip uses only 8 bits, the generated signal is not at the same quality with the typical output of commercial function generators; however, the developed analog signal generation system is very convenient for the

development and testing of control systems by repeating the experimental data in any desired sequence with the programmed interface.

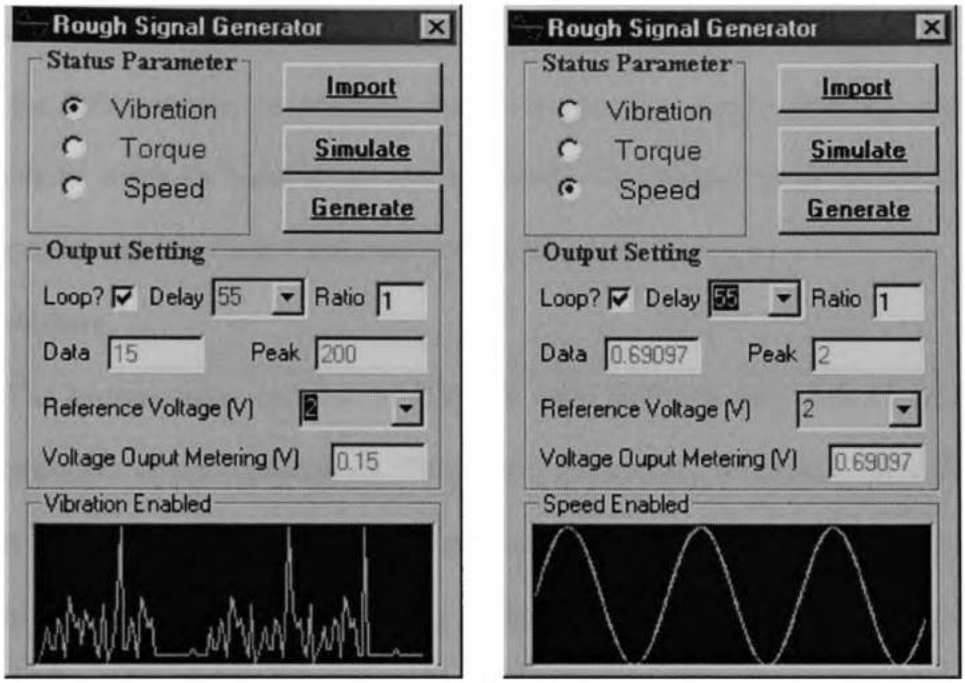

Figure 5.4 GUI interface of signal generation.

5.2 Signal Monitoring Capability Test of the DSP Evaluation Board

5.2.1 Hardware

The DSP board was connected to the parallel port of the microcomputer. An AC adapter provided five-volt supply. The jumpers on the board were set to allow uploading the program to the Flash memory and using the power supply as the reference voltage. The Code Composer was set up and the parallel port was selected for communication.

The input signal (analog) was connected to the desired pin. To minimize interference, all the analogous grounds were connected together. Connection was also

made between the digital grounds and the analog grounds for the D/A chip system (Figure 5.3).

The input sine wave signal was obtained from the function generator of a GoldStar OS-9020G 20 MHz analog oscilloscope. The signal was given to the ADCIN0 pin of the DSP module. In addition, the developed D/A signal generator was tested afterwards to verify its wave shape and amplitude. The frequency of generated signals was inspected with the Code Composer graphic function.

5.2.2 Software

The programming and use of DSPs is quite different from that of commercial microcontrollers or microprocessors. To prepare the system, or even to have the initial test runs of the program, the chip resources should be allocated [54] by considering the available memory. The graphical interface of the evaluation board with an input signal on display is presented in Figure 5.5.

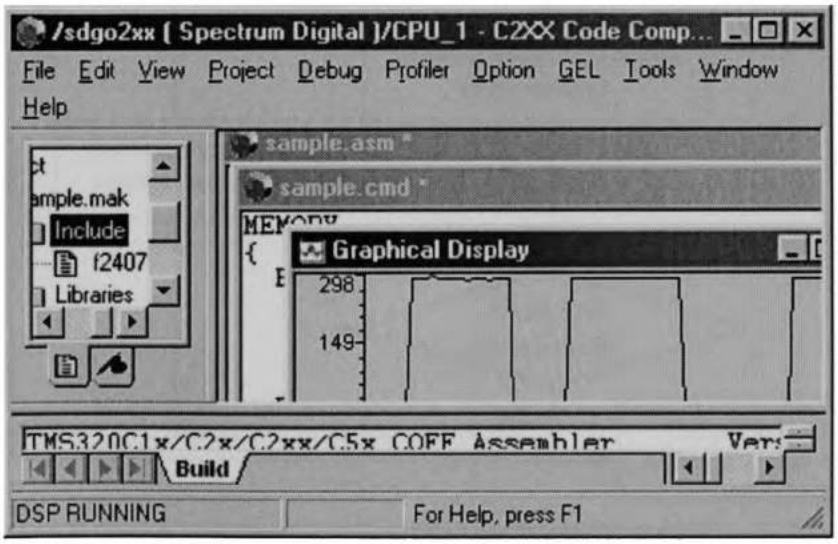

Figure 5.5 Code Composer window of the project test.

To calculate the desired properties of the signal, sampling rate and necessary processes should be carefully prepared. The flow chart of a program to calculate the amplitudes of any given sine waves is presented in Figure 5.6. First the desired sampling rate is selected. The actual sampling rate slightly varies because of the interruption service routine latency and other factors.

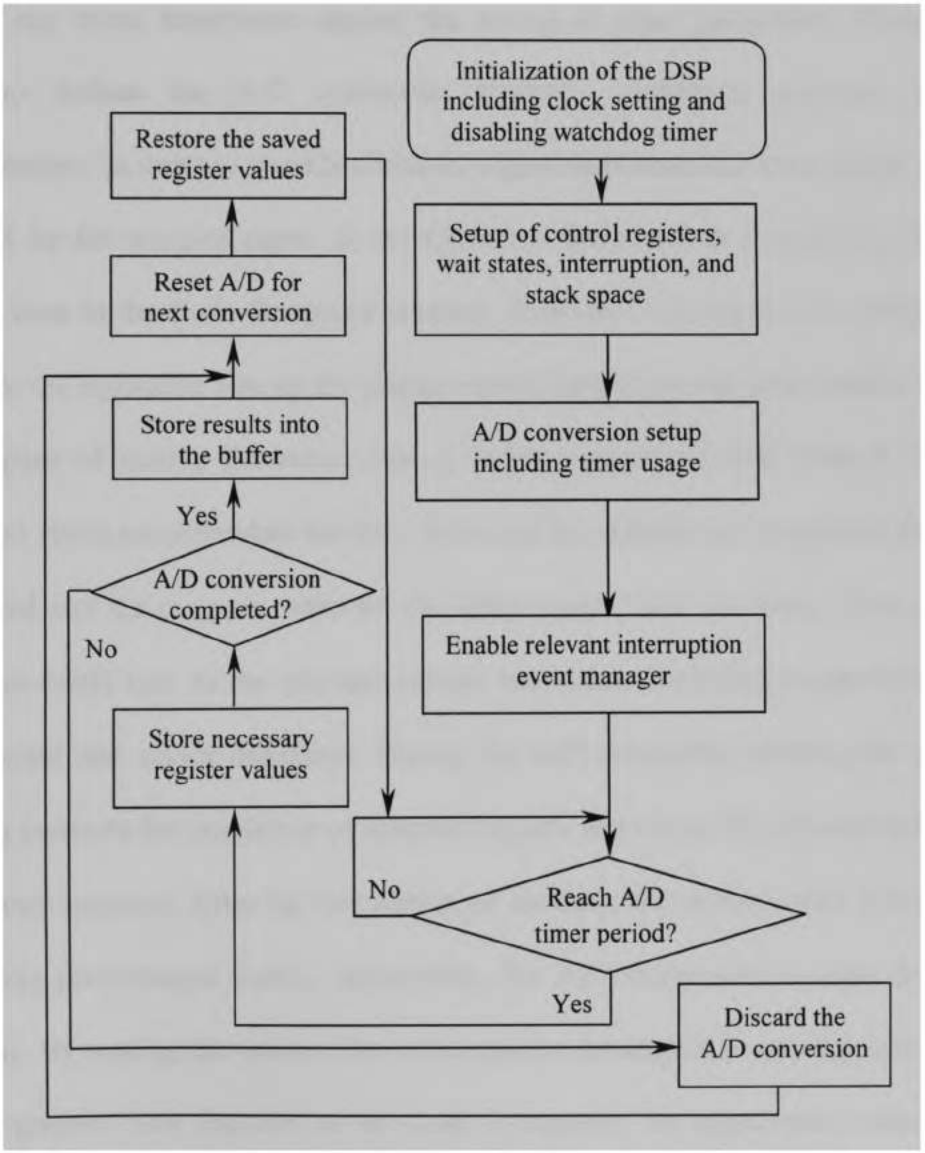

Figure 5.6 Flow chart of the signal amplitude sampling algorithm.

The algorithm first defines some variables for timer events and reserves some stack as well as buffer memory. Subsequently, the watchdog timer is disabled since it is unnecessary for the test of the DSP's capability herein. Following this, the CPU clock is scaled to processor components while some clocks are enabled to allow the usage of desired components on board, such as A/D conversion. Changing values in certain registers carries out the setting. At the same time, interruption is temporarily disabled in case of any event disturbance against the setting of other parameters. Thereafter, the algorithm defines the A/D conversion channel, conversion sequence and other configurations. In order to visually check the signal amplitude and wave shape, a buffer is arranged for the sampled signal so the converted data could be temporarily saved for a graphic view in the Code Composer window. Since the A/D conversion utilizes a timer resource, the algorithm sets up the proper sample frequency and timer-enable mode. For the purpose of storing the saved data, a buffer memory is well defined so that the algorithm could accommodate the data. When all the settings are completed, interruption is enabled and the program waits for the timer event. Once the timer event comes, the algorithm could turn to the relevant service subroutine according to previous memory arrangement and vector definition. During the A/D subroutine service, the conversion function converts the amplitude of sampled signals and stores the relevant numbers into its memory registers. After the completion of the duty, the correct value is read out and put in the pre-arranged buffer. Meanwhile, the A/D conversion is reset for the next sampling. By reading the value in the result register the amplitude could be calculated, or, via the graphic view function of the Code Composer, the signal peak value could be inspected, as the function displays the amplitude values at the selected time interval.

Chapter VI

Results and Discussion

In this section the performance of the three developed platforms will be presented.

6.1 Control of a 2½ Axes Stepping Motor System

A 2½ axes stepping motor system was developed and installed at a miniature milling machine to automate it. The performance of the system was tested while it moved the table of the machine tool. The system was designed to complete the development in a short time and to allow users to update the system easily when necessary. Compared to a conventional assembly language microcontroller, the BASIC Stamp based controller was easy to program and bugs could be detected and fixed quickly.

The P-BASIC program of the controller uses different strategies at each one of the 16 modes to create the linear table motions with any angle by synchronizing the X- and the Y- axes. It also handles the necessary communication with the microcomputer. The program is carefully optimized to take less than 85% of the BASIC Stamp EEPROM space (8×2 kB) and not to miss any steps. The information exchange between the microcomputer and the controller is almost unnoticeable to users since the developed communication protocol allows the transfer of the command and the machine status information in less than 0.5 s with the Intel Pentium® II computer.

The execution of each P-BASIC instruction takes a very long time compared to that of a set of equivalent assembly language statements. In addition, the accurate estimation of necessary clock cycles for the execution of a program line is almost impossible. The feed

rate adjustment, particularly at synchronized motions, required a series of experiments for improved effects.

A speed variable is used in the software to control the table speed. The program uses the speed variable to adjust the duration of the PAUSEs in each loop that fires different stepping motors coils. The time for the table movement of one inch is experimentally observed and presented in Figure 6.1. The time consistency of equivalent table motions is clearly seen in Figure 6.1. To move the table in the mode 1, 5, 9 and 13 only single stepping motor is used. The controller needs the shortest time to execute these instructions. To move the table in the mode 3, 7, 11 and 15 two stepping motors are rotated. In this case, the necessary time for instructions execution is the longest.

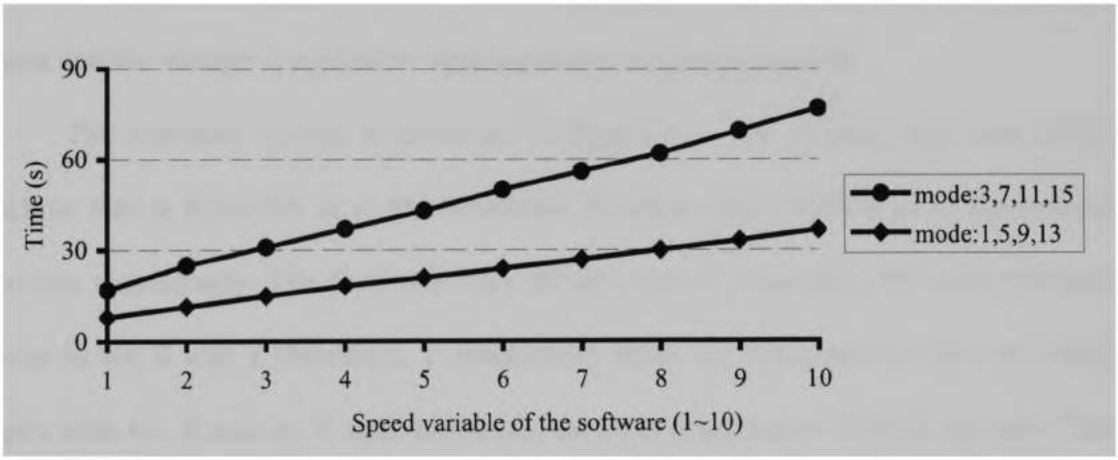

Figure 6.1 Speed of the table at different modes.

The linear characteristics of the execution speed of the instructions with fixed PAUSEs are demonstrated in Figure 6.1. In addition, the execution speed is almost two times faster when only one motor is rotated instead of two by the comparison of mode 1,

5, 9, 13 and mode 3, 7, 11,15. In mode 2, 4, 6, 8, 10, 12, 14 and 16 the execution time curve is between the two curves in Figure 6.1.

The rotation speed of stepping motors is controlled by the PAUSE instruction in the loops, which sequentially fires the different coils of stepping motors. The allowable range of the PAUSE is from 0 to 65535 [36]. Theoretically to reach the maximum speed, the minimum value (zero) should be given to the PAUSE variable (PAUSE 0). The estimated step counts for one-inch motion in mode 1 are 100×(1/0.0311) steps. While the actual time for the motion is 8 s, the theoretical time to move the table by one inch with the parameter PAUSE 1 is around 5.8 s.

The head motion in the Z direction (vertical) is smooth and reliable as long as the power supply provides the maximum current rating of the Darlingtons (10 A) in the circuit and the voltage is applied to stepping motor windings properly.

The complete system is presented in Figure 3.1. The average step feed of the machine tool is 0.000306 in re the horizontal direction and 0.000426 in re the vertical direction respectively. The feed rate error of the system is less than 5% when the tool moves in the X and Y directions. It deteriorates when the tool direction has very small angles with the X-axis or Y-axis. However, the error stays below 20% in any case. The feed rate error is less than 15% in the Z direction.

The repeatability error of the miniature milling machine at the feed rates of 0.03 in/s to 0.06 in/s are presented in Figure 6.2 to Figure 6.6. The absolute repeatability error of the machine while it carried a 3 lb weight was less than 0.001 in re the horizontal plane and less than 0.00005 in re the vertical direction. The absolute repeatability error in the vertical direction (Z axis) is very small since a gearbox is used between the motor and the

motion generating components. The resolution of the system could be improved by using a gearbox; however, in that case the maximum feed rate would be much lower than the present one if stepping motors with faster speeds were not applied.

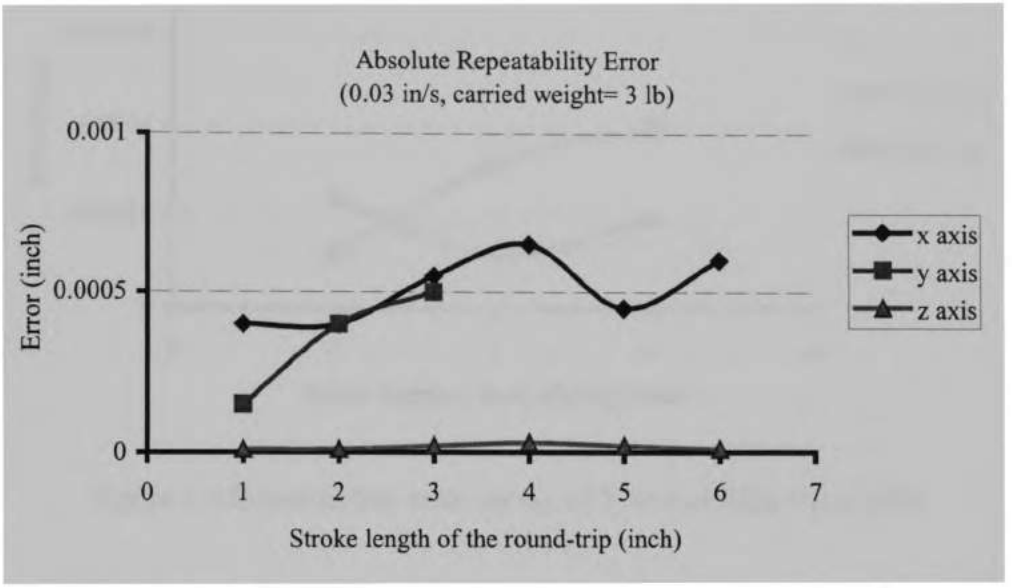

Figure 6.2 Repeatability error curves of all axes at lower speed.

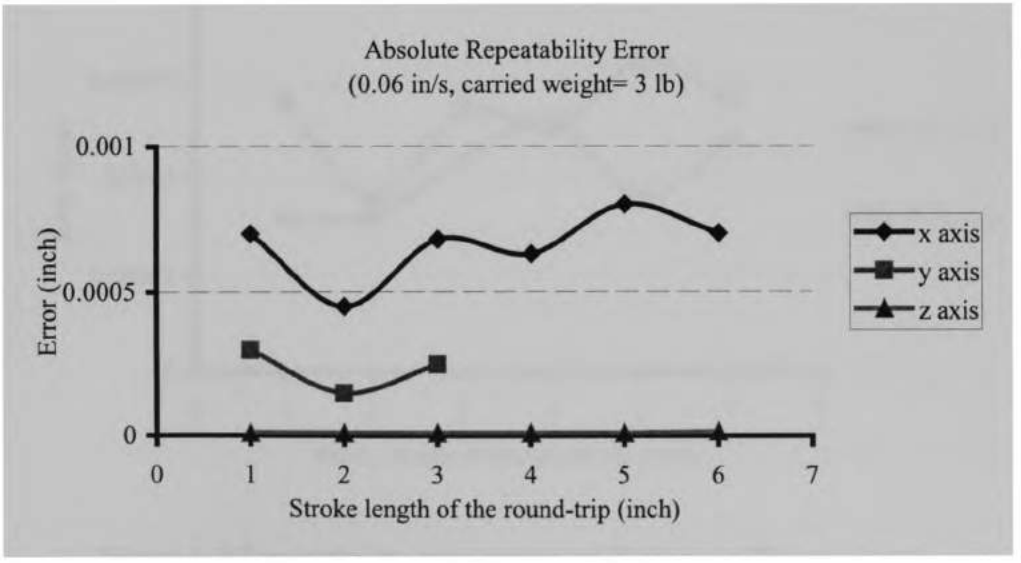

Figure 6.3 Repeatability error curves of all axes at higher speed.

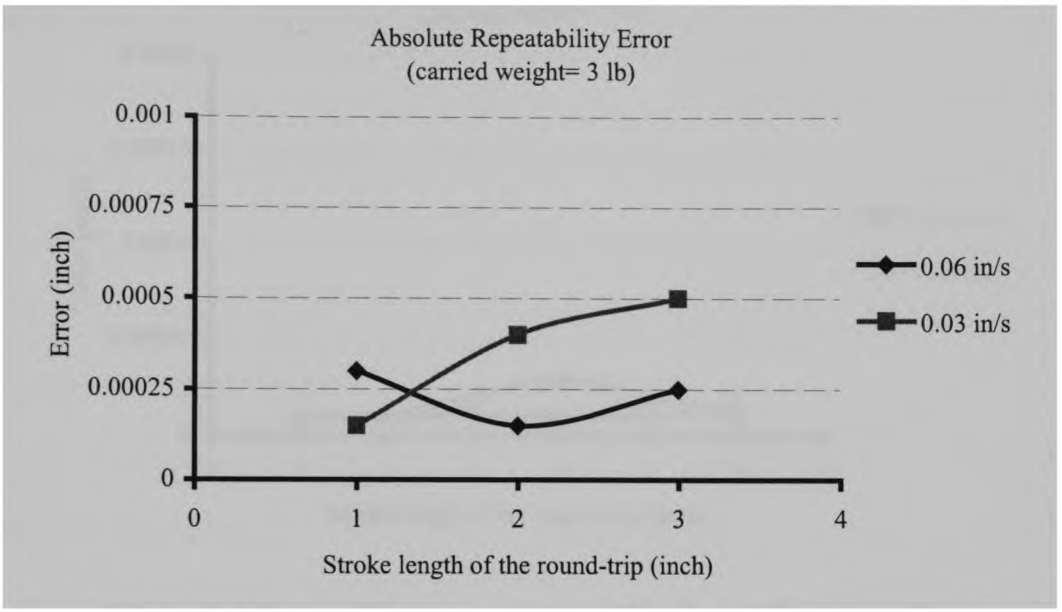

Figure 6.4 Repeatability error curves of Y axis at different speeds.

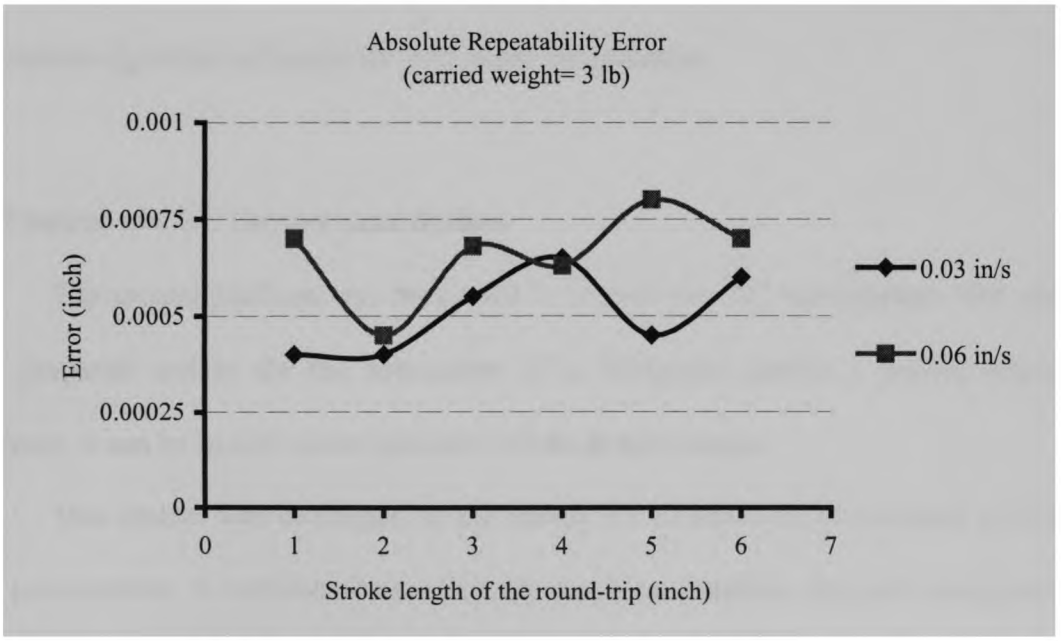

Figure 6.5 Repeatability error curves of X axis at different speeds.

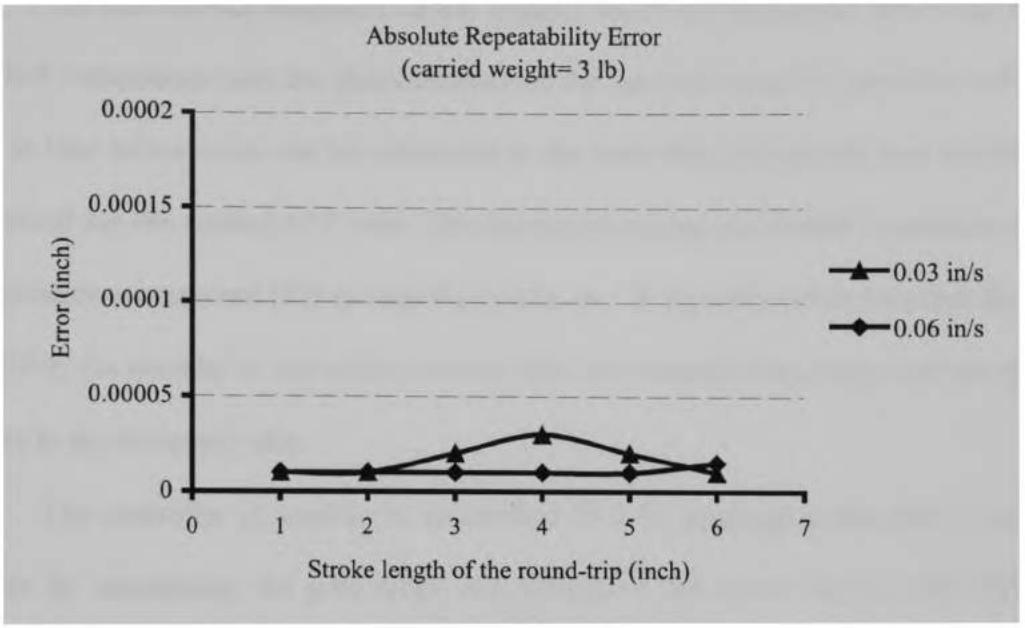

Figure 6.6 Repeatability error curves of Z axis at different speeds.

The memory of the present BASIC Stamp is not big enough for the expansion of the current algorithm to handle the 3-D linear interpolation.

6.2 Control of a DC Servomotors System

The second platform was developed to control two DC servomotors. The system was prepared mainly for the automation of a Bridgeport Series 1 milling machine; however, it can be used to create precise motions in any system.

This system was developed to use mainly the commercial components to control DC servomotors. A customized user-friendly graphical interface program communicates with the operator and gives instructions to the controller. Since the system controller is mass-produced, it has extensive motion capabilities and its price is very affordable.

The commercial controller of the system has to be configured according to the attached components and the characteristics of the load by using the provided software. One to four servomotors can be connected to the controller, though the user interface is optimized for the control of 2 axes. The proper grounding and healthy operation of the components are required [38] to keep the system on. If any component (whether the RTS amplifier, the encoder or any of the motors) fails, the internal relay opens and the system comes to an emergency stop.

The controller is capable to understand ISO G-Language codes and to turn the motors by considering the gear ratios and pitches of the screw. Up to 9999 different programs can be saved in the controller.

The developed manual user interface program checks all the components of the system one by one when the system is turned on. It allows the operator to preview the motions on the screen and to be capable to identify any of the components added into or removed from the system. Operators may use the graphical interface or can send any G-code instructions to the controller. Although the user interface supports only a few instructions, the direct access allows the operator to give any instruction in the controller's instruction list. The messages of the controller are displayed at the screen of the user interface.

The commercial controller can be configured to work with up to four motors. However, it checks each component and reconfigures itself if any of the components of the system malfunctions.

The encoder failures effect the operation of the whole system. If the encoder fails, the commercial and developed software cannot detect the current position of the motor

and thus turns the relevant motor indefinitely. The rotation commands to turn the motor in the opposite direction may abruptly stop the wrong motor.

The user interface currently supports only the linear interpolation. A pop-up window is added to let operators test the system and give any instruction. To send any desired instruction to the controller, the pop-up window (Figure 6.7) can be used.

Figure 6.7 Pop-up window for the direct control command test.

The servomotors of the developed system performed the given tasks satisfactorily.

6.3 DSP Based Machine Tool Monitoring System

The third platform was prepared to investigate the feasibility of the use of DSPs to monitor machining operations and to take corrective actions. DSPs acquire and process the data very quickly. It is necessary to use a separate chip to generate analog signals when the selected DSP is used. In this study an analog signal generator and a DSP module were prepared.

6.3.1 Analog Signal Generation by Using a Specialized D/A Chip

It is necessary to generate the analog signal according to the digital signals from the output ports of the DSP chip. To generate the analog signal, the AD7224 chip was

used [53]. Instead of the output ports of the DSP, the parallel port of a microcomputer was used to give inputs to the AD7224. A Visual Basic program generated the proper sequence. The microcomputer parallel port will be integrated with the DSP module.

To evaluate the accuracy of the Visual Basic timer component and to improve the delay precision, a program was prepared to control the signal generation. Using two approaches to evaluate their advantages generated the timing of the signal. The program used the timer component of the Visual Basic and the SLEEP function of the API. The test was carried out on a microcomputer with the Intel Pentium® III processor (450 MHz). The operating system was Microsoft Windows® 98 (the system RAM is 64 MB). The timing accuracy of both approaches was presented in Figure 6.8. The variable of the controlled delay was managed by either the Visual Basic timer component or the API function. The resolution of the Visual Basic timer component was found to be around 55 ms within the test range. There is no linear relationship between the desired delay and the actual delay for the Visual Basic timer component.

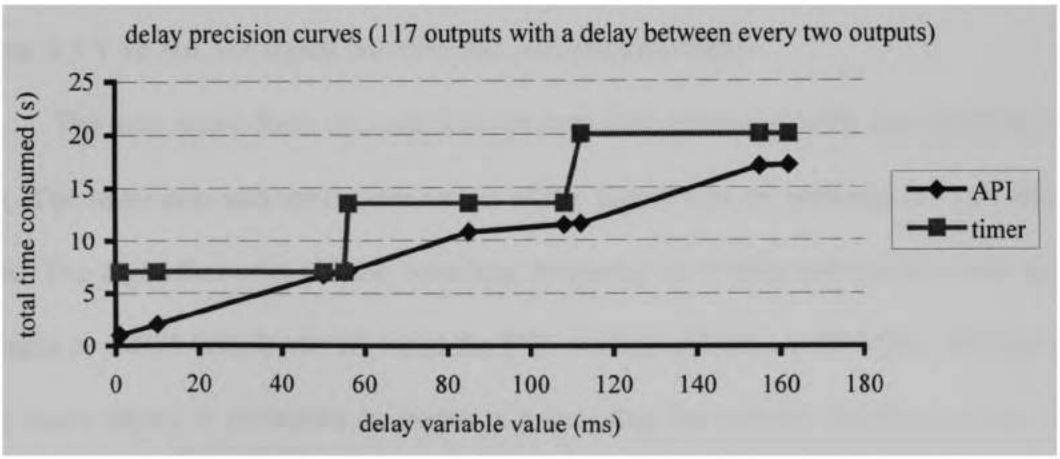

Figure 6.8 Timer delay error of Windows® programming.

The API SLEEP function was found more accurate without any consistent error tendency. The API function is more reliable than the Visual Basic timer component for the generation of high-speed signals. For the simulation of high frequency signals, the API application is recommended.

6.3.2 Performance of the DSP Evaluation Board

It is possible to determine the performances of many manufacturing operations and to evaluate the characteristics of sensory signals by inspecting their frequency spectrum. The DSP is very convenient for performing the real time spectrum analysis if the hardware will be used as a part of the controller. Low cost and extremely high processing speed allowed DSPs to be used in many applications from sound to image processing.

To evaluate the performance of the DSPs, an assembly language program was prepared and modified to select the proper sampling frequency according to the characteristics of the input signal. The peak voltage of the input signal was always kept below 3.3 V so that the signal could be sampled properly [48].

The sine wave from the oscilloscope was first generated with the frequency as 1 kHz. The minimum and maximum values of the signal was set between 0.1 volt and 1.1 volts. The algorithm selected the sampling frequency as 3 kHz and set the timer period variable as 10000 (clock period) since the DSP module operates at 30 MHz. The digitized sine wave signal is presented in Figure 6.9 by using the graphic function of the Code Composer.

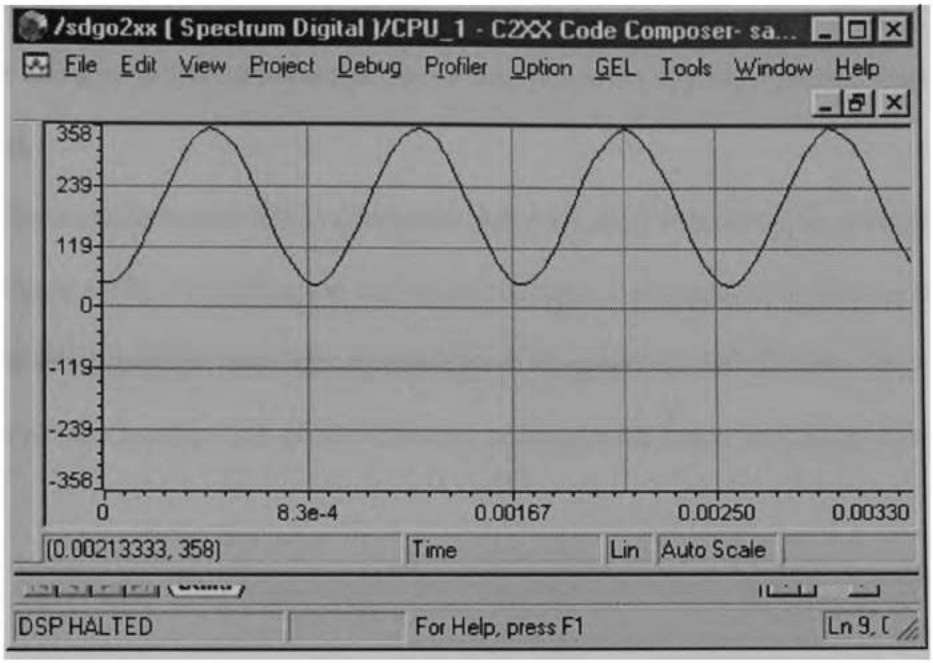

Figure 6.9 Single time plot of the A/D sampling data buffer.

It is possible to calculate the amplitude and frequency of the sine wave by inspecting the plots. An example is presented in Figure 6.9. The coordinate of the cursor is (0.00213333, 358). The first value corresponds to the time and the second value is the voltage level of the signal. Since the A/D reference high and low voltages were set to 3.3 V and 0 V respectively, the signal voltage can be found by using the equation [55]:

$$DigitalValue = 1023 \times \frac{(V_{INPUT} - V_{REFLO})}{V_{REFHI} - V_{REFLO}}$$

As can be seen, the peak voltage of this example was calculated as 1.15 volts (V_{REFHI} as 3.3 volts, V_{REFLO} as 0 volt [48] and the Digital Value as 358). The frequency of

the signal can be easily obtained from the period of the sine wave in the display after the cursor is brought to two consecutive peaks and the time interval between them can be calculated.

The manufacturer's Code Composer software also estimates the spectrum of the signal (Figure 6.10). The values on the horizontal axis correspond to the frequency of the signal. In the presented case the frequency of the peak is 10937.5 Hz. The function generator of the GoldStar OS-9020G 20 MHz analog oscilloscope was set to 10 kHz.

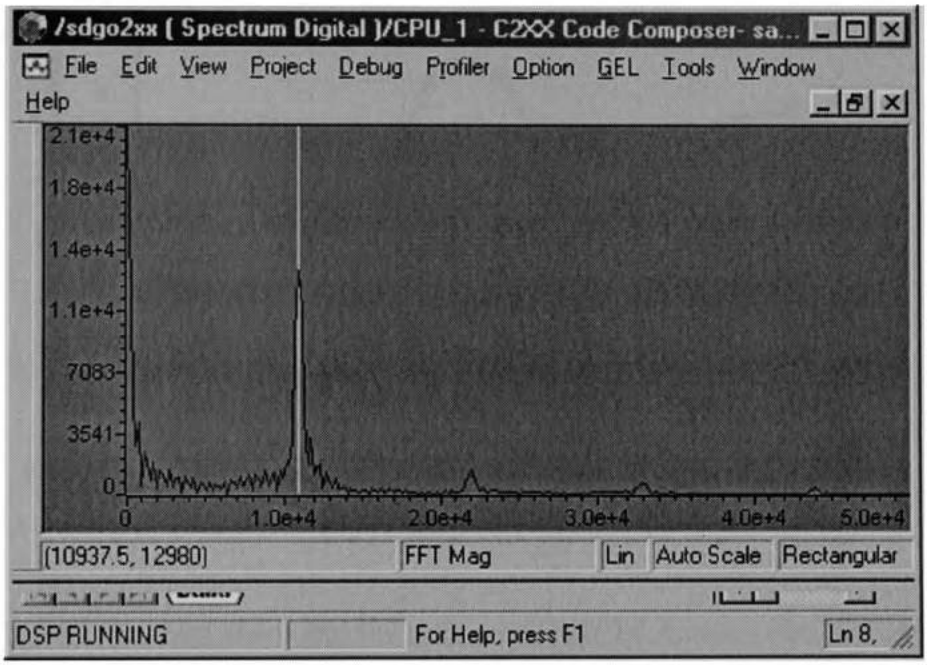

Figure 6.10 FFT amplitude plot of the A/D sampling data buffer.

The spectral response estimation accuracy of the DSP board and the algorithm were tested at different frequencies and the results are presented in Figure 6.11. Although the theoretical upper limit for the signal sampling is around 51.7 kHz (the program took

around 58 cycles for the interruption service), the actual sampled frequency range is carefully observed in concern with interruption service latency. The system had less than 10% frequency estimation error using the Code Composer from 10 kHz to 40 kHz. Based on the results we do not recommend the use of the monitoring system for the analysis of signals with the frequency above 40 kHz. Mechanical vibrations, acoustic signals and the output of force transducers can be analyzed by the system confidently.

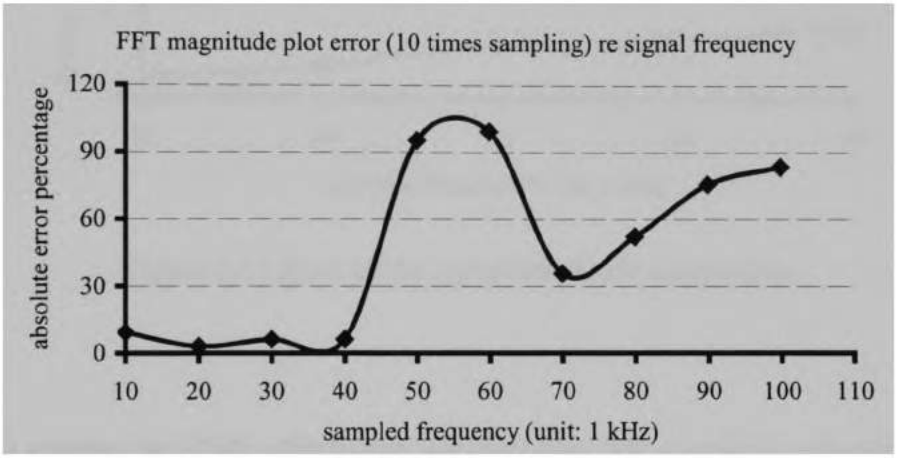

Figure 6.11 Error of the FFT magnitude plot in the frequency domain.

Experiments also show that the A/D amplitude estimation error of incoming sine wave signals at different frequencies (1 kHz ~ 40 kHz) is increased as the frequency increases (Figure6.12). The estimation errors at different frequencies could be reduced, if the sampling rate is optimized (during the test the sampling frequency is ten times of the incoming signal frequency). The programmed D/A signal generator was tested after the evaluation. Figure 6.13 shows the correct analog signal generation by the D/A chip system.

The amplitude and wave shape of the analog signal are verified with the DSP module. The frequency of the programmed certain analog signals is inspected as around 1074 Hz.

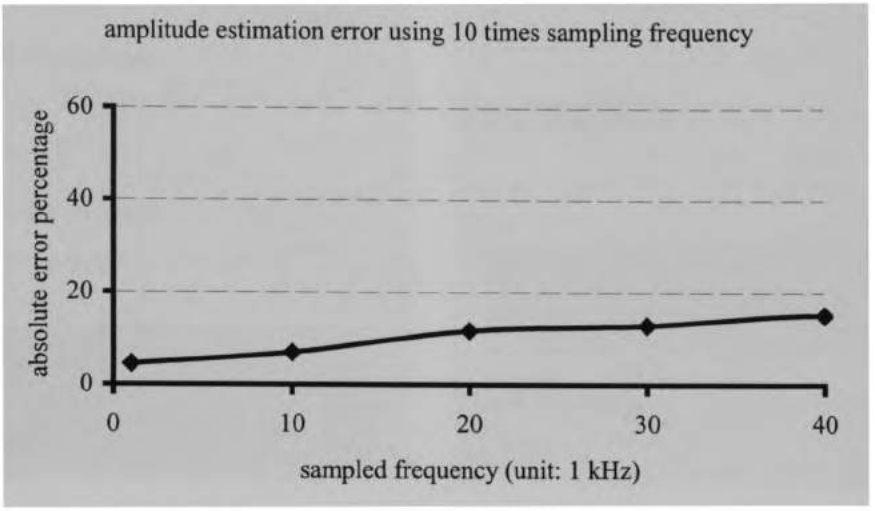

Figure 6.12 Error of the signal amplitude estimation.

To operate the DSPs effectively, the sampling rate should be selected carefully, the computational algorithm should be prepared by considering allowable time interval between the sampling, and a compensation algorithm should be used if necessary. In practice, the sampling rate is selected 10 times higher than the frequency of the monitored activity [56]. Snyder [56] recommended the selection of the sampling frequency from three to fifty times higher than the frequency of the considered activity. The computational algorithm should be prepared by using the floating-point calculations as little as possible. Divisions should be avoided by converting them into multiplications with fractions. The characteristics of the input signal, the hardware of the DSP and computational errors should be considered to compensate spectral estimations.

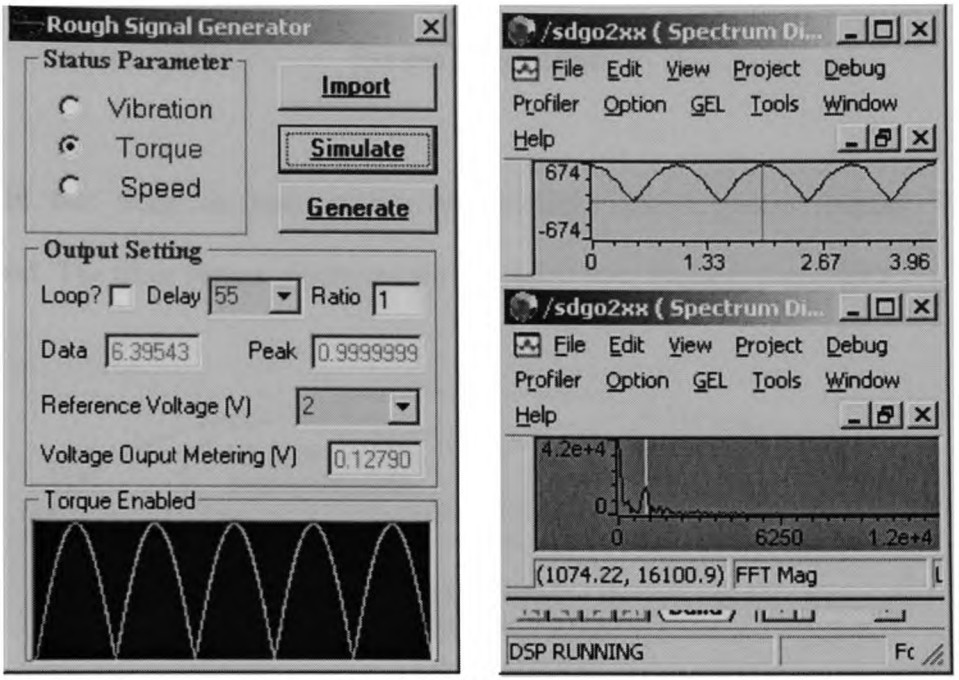

Figure 6.13 D/A signal generator verification with the DSP module (peak output ≈ 2.0 V)

Chapter VII

System Integration

In this study an open architecture motion control system (Figure 7.1) was developed. The three system platforms are presented from Figure7.2 to Figure 7.4.

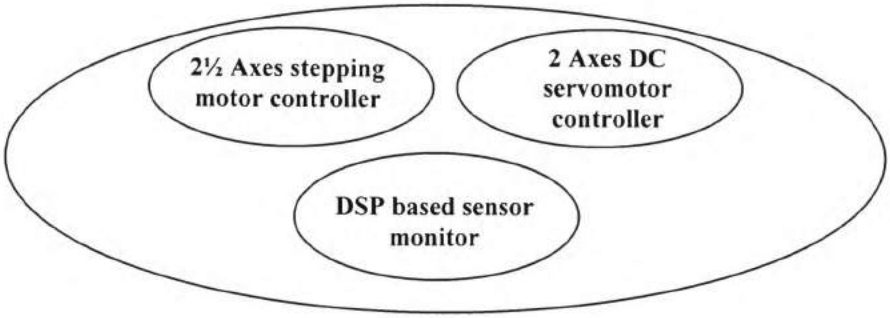

Figure 7.1 Developed open architecture controller.

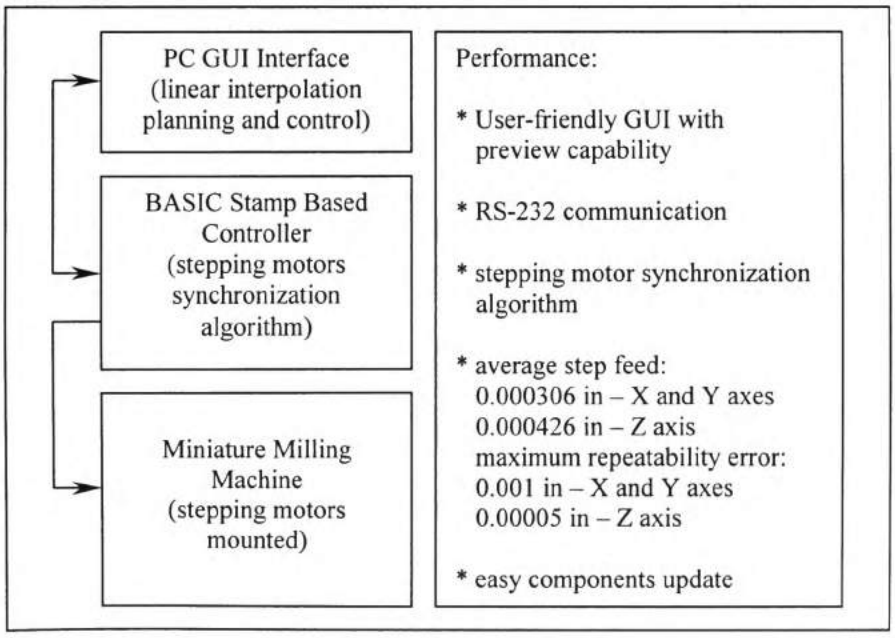

Figure 7.2 2½ axes stepping motor system control module.

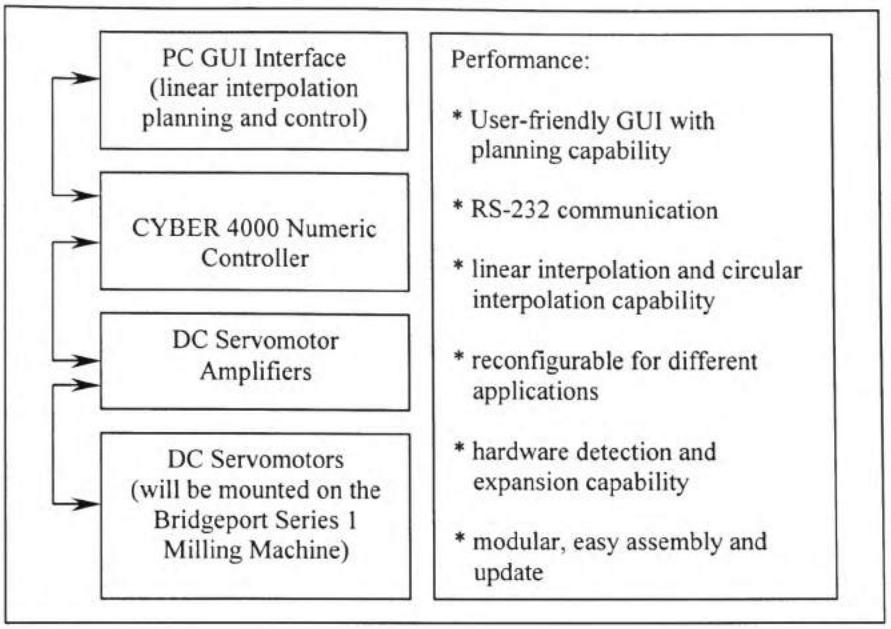

Figure 7.3 DC servomotor system control module.

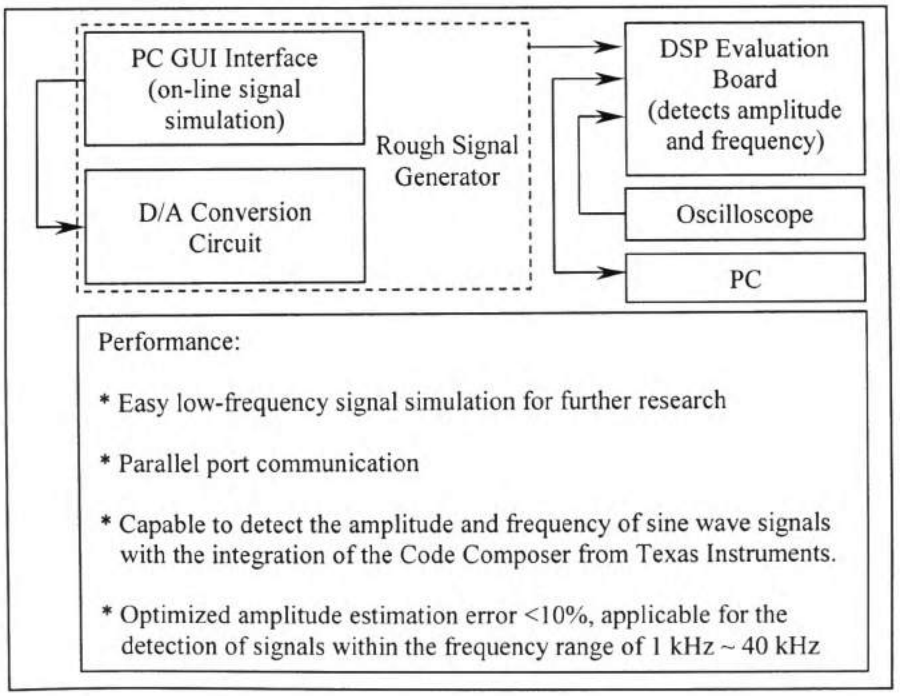

Figure 7.4 DSP module for machine tool monitoring.

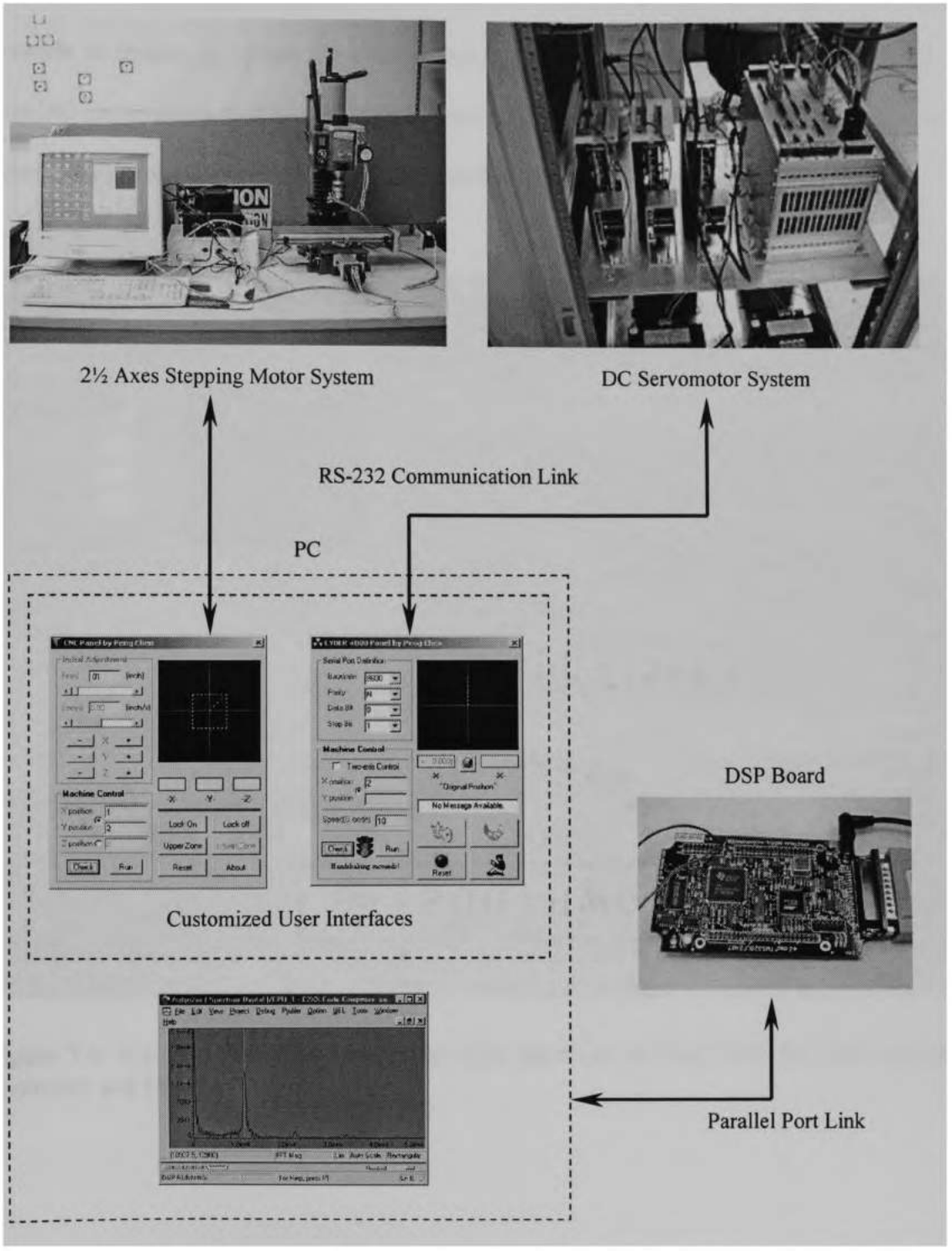

Figure 7.5 Diagram of the developed open architecture control system.

All the modules of the developed system are presented together in Figure 7.5. It is possible to replace or update the components of each platform. The user interface for 2-axis DC servomotors control and the spectrum of the Code Composer (the manufacturer's commercial program) of the DSP are presented in Figure 7.6.

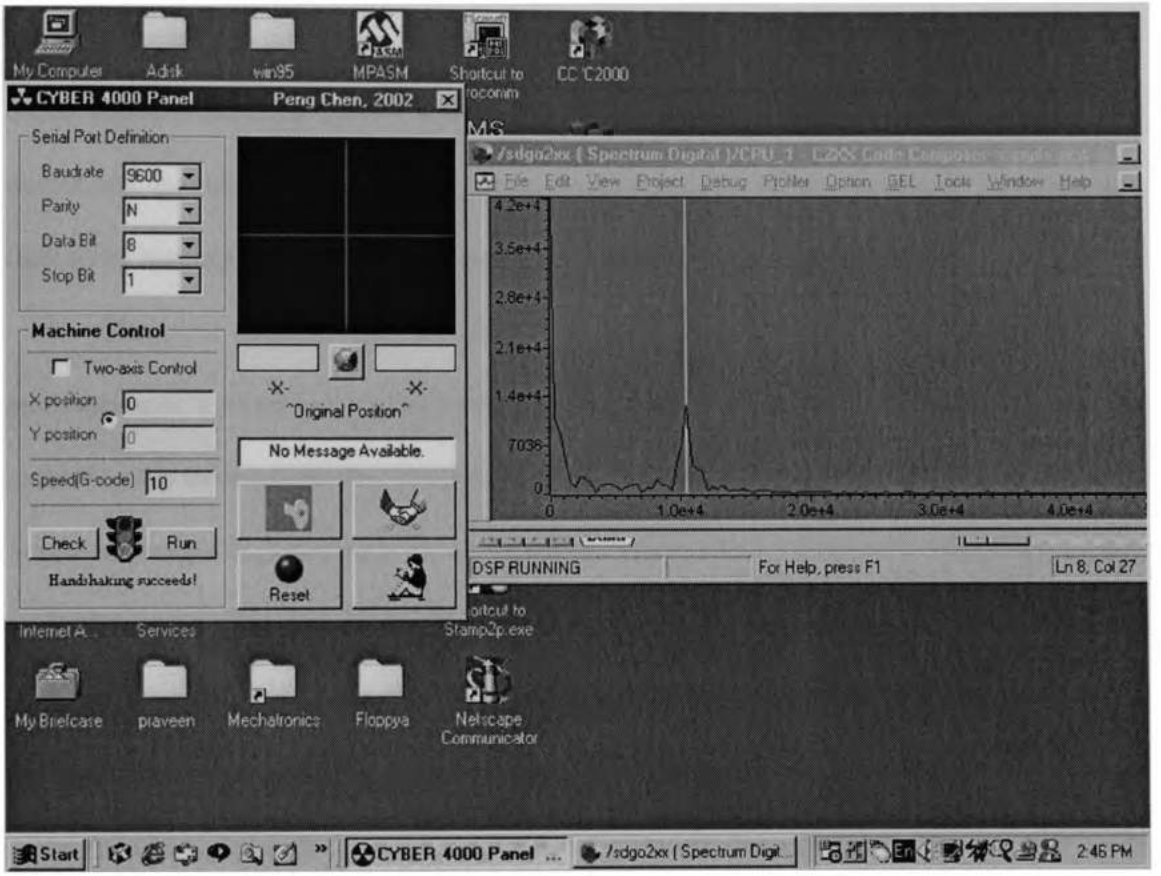

Figure 7.6. Simultaneous operation of the user interfaces of the 2-axis DC servomotor controller and DSP.

Chapter VIII

Conclusion

In this study, a comprehensive multiple-purpose open architecture control system was developed and tested under real world conditions. The system included a 2½-axis stepping motor and a 2-axis DC servomotor based motion control systems. Also, a DSP based signal processing system was developed for the evaluation of sensory signals. The developed modules can be used to move the components of any machine. In this study, they were applied to machine tools and their performances were evaluated.

The first module was developed to move small objects and applied for the automation of a miniature milling machine. Four major component groups of the system including the microcomputer, the graphical user interface software, the 2½ axes stepping motor controller and stepping motors could be individually replaced to adapt the system to different applications. The BASIC Stamp based stepping motor controller was designed to develop a very reliable system in less than ½ man-year. Since the language is not microprocessor specific, the controller can be upgraded easily when faster and more powerful chip packages become available. The result demonstrated that the automated machine tool had less than 0.001-inch (0.001 in re the horizontal plane and 0.00005 in re the vertical axis) repeatability error. The linear interpolation capability of the system allowed the movement of the tool in any direction in the horizontal plane at any supported speed. The user gave instructions by using the graphical interface of the microcomputer. The possibility of giving wrong instructions was minimized by the graphical preview of the motions before the actual tool displacement was created.

The second module was a 2-axis DC servomotor based motion control system. This system is capable to create very precise motions and can be applied to very large machine tools. Four major components of this module included a microcomputer, the developed graphical user interface software, the CYBER 4000 numeric controller and servomotors with amplifiers. Each component of the system could be updated, or replaced according to the application. The user interface of the microcomputer gives the instructions to the numeric controller. However, the numeric controller can be directly programmed with any terminal by using G-language and it is capable to control the rotations of 4 separate motors. The controller monitors the speed and position of the motors with a resolver and an encoder respectively. It is capable to create linear and circular interpolations. The developed Windows®-based manual navigation software was designed to handle two axes since the module will be used to control the table of a Bridgeport Series 1 milling machine. It can be upgraded to handle more axes if necessary with minimal program changes. The software may be expanded to download the full G-code programs into the controller in future. This module was developed in ½ man-year.

The third module was a DSP based very fast signal processing system. Open architecture motion control systems require the processing of sensory signals to create precise motions and to take necessary operational decisions. The tested DSP module digitizes the analog signals and obtains the Fourier Transformations. Fast communication was created between the microcomputer and the DSP without using the Visual Basic timer component.

The modules of the developed open architecture control system can be used individually, or together depending on the application.

8.1 Future Studies

Future studies involve the following:

1. To design a single more sophisticated user interface: Currently customized user interfaces operate the stepping motor and DC servomotor based motion control systems. Although the user interfaces are very similar, they are not identical. In future, a single user interface will be designed with multi-step preview to operate both systems at the same time.
2. To design sophisticated monitoring and quality improvement systems: Operators currently can monitor the spectral characteristics of any sensory signal; however, there is not automatic interpretation of the signal and the system cannot determine the proper adjustments to improve the quality. In the future, the program of the DSP will be improved to evaluate the characteristics of the incoming signal and to select proper operating conditions to operate the system at optimal conditions.
3. To implement into new environments: All the developed platforms were tested by considering machine tools. The open architecture controller will be used in new environments to adjust the orientation of the camera, microscope and sensor of the NDT (Non Destructive Testing) equipment.

References

1. Manufacturing Data System Inc. Internet on-line. <http://www.mdsi2.com>. [28 May 2002].

2. Rockwell Automation. Internet on-line. <http://www.ab.com/cnc>. [28 May 2002].

3. McMahon, Richard M., Hanssmann, M. G., and Shuen, A. K. "Open architecture computer and software systems for the full automation of vacuum coating equipment." *Proceedings, Annual Technical Conference - Society of Vacuum Coaters, Held in Boston, MA, USA 27 April – 1 May 1987,* 145-150. Albuquerque, NM, USA: Society of Vacuum Coaters, 1987.

4. Fang, X. Daniel., and Lee, N. J. "New tooling mechanism for CNC lathes." *International Journal of Machine Tools and Manufacture* 41 (November 2001): 89-101.

5. Loffler, Markus S., Chitrakaran, Vilas K., and Dawson, Darren M. "Design and implementation of the robotic platform." *IEEE Conference on Control Applications - Proceedings 2001, Held in Mexico City, Mexico 5-7 September 2001,* 357-362. IEEE, 2001.

6. Bona, B., Indri, M., and Smaldone, N. "Open system real time architecture and software design for robot control." *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Held in Como, Italy 8-12 July 2001,* 349-354. IEEE, 2001.

7. Hong, K.-S., Choi, K.-H., Kim, J.-G., and Lee, S. "A PC-based open robot control system: PC-ORC." *Robotics and Computer-Integrated Manufacturing* 17 (August 2001): 355-365.

8. Wang, S., and Shin, K. G. "Reconfigurable software for open architecture controllers." *Proceedings - IEEE International Conference on Robotics and Automation, Held in Seoul, South Korea 21-26 May 2001,* 4090-4095. IEEE, 2001.

9. Mao, Junhong., Li, Lichuan., and Wu, Xutang. "Hardware independent architecture for CNC machine tools reconfiguration." *Chinese Journal of Mechanical Engineering* 36 (July 2000): 48-51.

10. Erol, N. Arda., Altintas, Yusuf., and Ito, Mabo Robert. "Open system architecture modular tool kit for motion and machining process control." *IEEE/ASME Transactions on Mechatronics* 5 (September 2000): 281-291.

11. Yellowley, I., Seethaler, R. J. and Yeung, F. W. "Integration of process planning, monitoring and control in a machine tool environment." *Proceedings of SPIE - The International Society for Optical Engineering, Held in Boston, MA, USA 19-20 September 1999,* 38-47. SPIE, 1999.

12. Terbuc, Martin., Hace, Ales., and Jezernik, Karel. "Open structure multiprocessor robot controller." *IEEE International Symposium on Industrial Electronics, Held in Bled, Slovenia 12-16 July 1999,* 899-902. IEEE, 1999.

13. Chang, Timothy., Godbole, Kedar., Eren, Murat., Ji, Zhiming., and Caudill, Reggie. "Development and implementation of an Application Programming Interface for PC/DSP based motion control system." *Proceedings of SPIE - The International Society for Optical Engineering, Held in Boston, MA, USA 4-5 November 1998,* 94-105. Bellingham, WA, USA: SPIE, 1998.

14. Barambones, O., and Etxebarria, V. "Control of a laboratory robot using an open DSP-based controller." *IEE Conference Publication, Held in Swansea, UK 1-4 September 1998,* 1646-1651. Stevenage, England: IEE, 1998.

15. Xie, Xuhui. "Computer numerical control (CNC) system for ultra-precision machine tools." *China Mechanical Engineering* 9 (February 1998): 21-22.

16. Zhang, Qiyi G., and Greenway, R. Bryan. "Development and implementation of a NURBS curve motion interpolator." *Robotics and Computer-Integrated Manufacturing* 14 (February 1998): 27-36.

17. Baptista, Luis., Martins, Jorge., Cardeira, Carlos., and Sa da Costa, Jose. "Open architecture for position and force control of robotic manipulators." *Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems, Held in Lisboa, Portugal 7-10 September 1998,* 471-474. IEEE, 1998.

18. Mitsuya, Eiji., Nanjo, Yoshito., Mitamura, Akio., and Mizukawa, Makoto. "Intelligent robot system architecture design and implementation." *NTT R&D* 47, no.7 (1998): 799-804.

19. Lopez-Orozco, J. A., de la Cruz, J. M., Dominguez, E., Besada, E., and Polo, O. R. "Open sensing architecture to autonomous mobile robots." *IEEE International Symposium on Intelligent Control - Proceedings 1998, Held in Gaithersburg, MD, USA 14-17 September 1998,* 610-615. Piscataway, NJ, USA: IEEE, 1998.

20. Yamazaki, Kazuo., Hanaki, Yoshimaro., Mori, Masahiko., and Tezuka, Kazusaka. "Autonomously proficient CNC controller for high-performance machine tools based on an open architecture concept." *CIRP Annals - Manufacturing Technology* 46, no.1 (1997): 275-278.

21. Zhou, Xuecai., Li, Weiping., and Li, Qiang. "New robot control system with open architecture." *International Conference on Advanced Robotics, Proceedings, ICAR 1997, Held in Monterey, CA, USA 7-9 July 1997,* 813-818. Piscataway, NJ, USA: IEEE, 1997.

22. Zhou, Lei., Washburn, Michael J., Shin, Kang G., and Rundensteiner, Elke A. "Performance evaluation of modular real-time controllers." *Proceedings of the ASME Dynamic Systems and Control Division American Society of Mechanical Engineers, Dynamic Systems and Control Division (Publication) DSC, Held in Atlanta, GA, USA 17-22 November 1996,* 299-306. New York, NY, USA: ASME, 1996.

23. Rober, Stephen J., and Shin, Yung C. "Modeling and control of CNC machines using a PC-based open architecture controller." *Mechatronics* 5 (June 1995): 401-420.

24. Park, Jaehyun., Birla, Sushil., Shin, Kang G., Pasek, Zbigniew J., Ulsoy, Galip., Shan, Yansong., and Koren, Yoram. "Open architecture testbed for real-time monitoring and control of machining processes." *Proceedings of the American Control Conference, Held in Seattle, WA, USA 21-23 June 1995,* 200-204. AACC, 1995.

25. Bailey, T., Ruget, Y., Spence, A., and Elbestawi, M. A. "Open-architecture controller for die and mold machining." *Proceedings of the American Control Conference, Held in Seattle, WA, USA 21-23 June 1995,* 194-199. AACC, 1995.

26. Proctor, Fred., Shackleford, Will., Yang, Charles., Barbera, Tony., Fitzgerald, M. L., Frampton, Nat., Bradford, Keith., Koogle, Dwight., and Bankard, Mark. "Simulation and implementation of an open architecture controller." *Proceedings of SPIE - The International Society for Optical Engineering, Held in Philadelphia, PA, USA 25-26 October 1995,* 196-204. Bellingham, WA, USA: SPIE, 1995.

27. Pritschow, G., Altintas, Y., Jovane, F., Koren, Y., Mitsuishi, M., Takata, S., Van Brussel, H., Weck, M., and Yamazaki, K. "Open controller architecture - Past, present and future." *CIRP Annals – Manufacturing Technology* 50, no. 2 (2001): 463-470.

28. Pritschow, G., Daniel, Ch., Junghans, G., and Sperling, W. "Open system controllers - a challenge for the future of the machine tool industry." *CIRP Annals* 42, no. 1 (1993): 449-452.

29. Wright, P. K., and Greenfeld, I. "Open architecture manufacturing. The impact of open-system computers on self-sustaining machinery and the machine tool industry." *Proceedings of Manufacturing International'90 Part 2: Advances in Manufacturing Systems, Held in Atlanta, GA, USA 25-28 March 1990,* 41-47. New York, NY, USA: ASME, 1990.

30. Hayashi, Tetsuji., and Tokioka, Atsushi. "Experiment for operational use of 7-axis manipulator (improvement of motion controller for portable system)." *Transactions of the Japan Society of Mechanical Engineers, Part C* 59 (September 1993): 2753-2757.

31. Overmars, A. H., and Toncich, D. J. "Application of DSP technology to closed-position-loop servo drive systems." *International Journal of Advanced Manufacturing Technology* 11, no. 1 (1996): 27-33.

32. Kim, Dong-Il., Song, Jin-Il., and Kim, Sungkwun. "Dependence of machining accuracy on acceleration/deceleration and interpolation methods in CNC machine tools." *Proceedings of the 29th IAS Annual Meeting Part 3, Held in Denver, CO, USA 2-5 October 1994,* 1898-1905. Piscataway, NJ, USA: IEEE, 1994.

33. Shinano Kenshi Corporation. Internet on-line. <http://www.shinano.com>. [1 July 2002].

34. CHUNG HSIWH INDUSTRIAL CO., LTD. Internet on-line. <http://machinery.cetra.org.tw/emo/coinfo.asp?ban=55989346>. [28 May 2002].

35. Parallax Inc. 11 June 2002. Internet on-line. <http://www.parallaxinc.com/html_files/products/Basic_Stamps/module_bs2p.asp>. [1 July 2002].

36. Parallax Inc. *BASIC Stamp® Programming Manual Version 2.0c.* 21 June 2002. Internet on-line. <http://www.parallaxinc.com/downloads/Documentation/Basic%20Stamps/BASIC%20Stamp%20Manual%20v2.0.pdf>. [28 May 2002].

37. Kühnel, Claus., and Zahnert, Klaus. *BASIC Stamp.* Boston, USA: Butterworth-Heinemann, 1997.

38. PARVEX Inc. *CYBER 4000 Technical Instructions (PVD3425GB).* September 1996.

39. PARVEX Inc. *RTS DC Servoamplifiers (PVD3416US).* December 1999. Internet on-line. <http://www.parvex.com/pdf/us/com/3416-US.pdf>. [28 May 2002].

40. PARVEX Inc. *RX DC Servomotors (PVD3366US).* December 1999.

41. PARVEX Inc. *Parvex Motion Explorer Software (PVD3495GB).* March 1998.

42. El-Sinawi, A., and Kashani, R. "Active isolation using a Kalman estimator-based controller." *Journal of Vibration and Control* 7 (November 2001): 1163-1173.

43. Smith, D. A., Smith, S., and Tlusty, J. "High performance milling torque sensor." *Journal of Manufacturing Science and Engineering, Transactions of the ASME* 120 (August 1998): 504-514.

44. Wu, Yue., Bobis, James P., and Gehman, Richard. "Design and analysis of an improved high air flow meter with analog/digital filters." *IEEE Transactions on Instrumentation and Measurement* 41 (December 1992): 791-796.

45. Kassim, A. A., Fong, F. K., Chua, K. S., and Rangananth, S. "DSP-based video compression test-bed." *Microprocessors and Microsystems* 20 (May 1997): 541-551.

46. Luo, Fang Lin., and Ye, Hong. "DSP-based tension control and data acquisition for paper machine rewind roll drive." *IEEE Transactions on Industry Applications* 36 (July 2000): 1018-1025.

47. Microchip Technology Inc. *dsPIC™ Digital Signal Controllers.* 28 June 2002. Internet on-line. <http://www.microchip.com/1010/pline/dspic/index.htm>. [1 July 2002].

48. Spectrum Digital Inc. *eZdsp™LF2407 Technical Reference, Rev.C.* August 2001. Internet on-line. <http://www.spectrumdigital.com/technical/pdfs/eZdsp2407.pdf>. [28 May 2002].

49. Texas Instruments Inc. *TMS320LF/LC240xA DSP Controllers Reference Guide: System and Peripherals (SPRU357B).* December 2001.

50. Texas Instruments Inc. *Code Composer Studio Getting Started Guide (SPRU509C).* November 2001. Internet on-line. <http://www-s.ti.com/sc/psheets/spru509c/spru509c.pdf >. [May 28 2002].

51. Texas Instruments Inc. *TMS320F/C24x DSP Controllers Reference Guide: CPU and Instruction Set (SPRU160C).* June 1999.

52. Diehl, P. A., Chen, P., and Tansel, I. N. "Electro-optic vibration sensor for micro - machining," *Proceedings of the 15th Florida Conference on Recent Advances in Robotics, Held in Miami, FL, USA 23-24 May 2002,* edited by Tosunoglu, Sabri. Miami, FL, USA: Department of Mechanical Engineering, Florida International University, 2002.

53. Analog Devices Inc. *LC²MOS 8-Bit DAC with Output Amplifiers.* Internet on-line. <http://www.analog.com/productSelection/pdf/ad7224.pdf>. [25 May 2002].

54. Texas Instruments Inc. *TMS320C1x/C2x/C2xx/C5x Assembly Language Tools User's Guide (SPRU018D).* April 1998.

55. Texas Instruments Inc. *TMS320LF2407A, TMS320LF2406A, TMS320LF2403A, TMS320LF2402A, TMS320LC2406A, TMS320LC2404A, TMS320LC2402A DSP Controllers (SPRS145G)*. February 2002. Internet on-line. <http://www-s.ti.com/sc/psheets/sprs145g/sprs145g.pdf>. [18 June 2002].

56. Snyder, S. D. "Active control - A bigger microprocessor is not always enough." *Noise Control Engineering Journal* 49, no. 1 (2001): 21-29.