CONVERTING A MILLING MACHINE TO A TWO AXIS CONTOURING MACHINE

by

BRUCE P. MIGNANO

B.S., United States Military Academy, 1958
M.S., Stevens Institute of Technology, 1964

A MASTER'S THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering
KANSAS STATE UNIVERSITY
Manhattan, Kansas
1980

Approved by:

Major Professor

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS BOOK
CONTAINS
NUMEROUS PAGES
WITH THE ORIGINAL
PRINTING BEING
SKEWED
DIFFERENTLY FROM
THE TOP OF THE
PAGE TO THE
BOTTOM.

THIS IS AS RECEIVED
FROM THE
CUSTOMER.

ACKNOWLEDGEMENT

The author expresses his gratitude to his family for their patience over the past year. In addition the author is indebted to Professor John E. Biegel for his guidance, suggestions, and good humor during the course of this research.

CHAPTER 1

INTRODUCTION

INTRODUCTION

1.1 HISTORICAL BACKGROUND

> "One can hardly say that the machine tool was a sufficient condition for the Industrial Revolution, but we may be certain that it was a necessary condition for the development of the Industrial society in which we live."*

The above quotation by Robert S. Woodbury underscores the importance of metal removal tools to our society [1]. The machine tool has made and continues to make a significant contribution to our daily lives. The milling machine forms an important class of machine tool and serves as the focal point of this thesis. This introduction will trace the significant developments in the technical history of the milling machine and in so doing explore the current state of the art.

Recorded evidence points to Eli Whitney, Robert Johnson, and James Nasmyth as the builders of the first milling machines [1]. These machines were in operation in the 1820's. In the Whitney milling machine power was delivered to the spindle through a pulley system. Feed power was supplied by a pulley fastened to the same shaft as the spindle pulley. The Whitney machine also provided a mechanism to permit hand, as well as, power feed.

Johnson's milling machine was used in a small arms factory in Middletown, Connecticut around 1818 [1]. This mill used a lathe head as the spindle. The machine did not have power feed. A hand crank was used to feed the workpiece into the stationary cutter.

Between 1829 and 1831, James Nasmyth developed a milling machine in England that was used for milling the sides of small nuts [1]. Although this machine had a specialized purpose, it did incorporate some new ideas, The use of an indexing table and a small tank of water to cool the cutter were two of Nasmyth's creative innovations to the development of milling machines.

* Robert S. Woodbury, Studies in the History of Machine Tools, Cambridge, Mass., The MIT Press, 1960, Preface.

The first significant production milling machine was the Lincoln Miller. The Lincoln Miller was introduced in 1855. It was built by F. A. Pratt and Amos Whitney [1]. The Lincoln Miller was a rugged machine that used a screw and nut power feed mechanism. It was capable of milling heavier work and cutting with greater precision than the earlier machines.

In 1861, Joseph R. Brown while working for the Brown & Sharpe Co. invented the Universal Milling Machine [1]. Impetus for this invention was provided by Frederick W. Howe who wanted a machine to cut the grooves on twist drills. The most significant improvement incorporated in Brown's machine was the use of the column and knee principle to provide accurate adjustment of the cutter relative to the work.

Starting with Brown's Universal Milling Machine, the machine tool industry began to refine feed mechanisms and power drives. In 1900 a geared drive and feed powered by a constant speed electric motor was applied to a Brown and Sharpe milling machine [1]. In addition the feed rate could now be chosen independent of the spindle speed.

In 1910, the automotive industry's demand for machine tools that could achieve the high production rates necessary to satisfy America's ever increasing hunger for automobiles, forced a response from the machine tool industry. The bed-type milling machine became the standard milling machine for the incorporation of the automatic control systems that were developed during the 1920's. E. F. Lathan and E. Stuck used mechanical devices to incorporate a quick return and fully automatic fast feed between cuts. In 1927 J. W. Anderson built a hydraulic, tracer controlled machine [1].

The next development of major significance occurred in 1952, when the United States Air Force contracted with the Massachuetts Institute of Technology to use electronic technology to control a mill [2]. The system developed used vacuum tube technology to process input data from a paper tape.

The control system used the information on the paper tape to control the cutter part relationship along three axes. This was the first use of numerical control techniques on a large machine tool. Although this numerical controlled milling machine offered greater repeatability and a lower error rate than conventional machines, the manufacturing industry had significant problems incorporating the new machine into production operations. The root of these problems was the lack of an efficient part programming system. In 1956, the Air Force again contracted with MIT to provide a programming concept. From this contract the APT language was born. APT meaning Automatically Programmed Tools [2]. The APT language provided the unifying element to eliminate some of industry's problems. During the 1960's APT continued to develop more sophistication in describing part geometry.

Since the development of the first numerically controlled machine in 1952, advances in the control of machine tools have followed the advances in computer technology. In 1952 the control system was a breadboard type made entirely of vacuum tubes. The control unit was larger than the milling machine that it controlled. The rapid advances of solid state technology have produced large reductions in the size of control units. This size reduction was accompanied by significant cost reductions; making the electronic control of machine tools attractive in situations that were unthinkable ten years prior.

In the 1970's two new types of numerical control were introduced. In Direct Numerical Control (DNC) tool movements are directly transmitted from the computer to the NC machine [3]. The same computer may be used to control severa NC machines. Computer Numerical Control (NC) dedicates a single computer to control a single machine [3].

Current solid state technology is packing more memory and operational flexibility into smaller and smaller sapces [4]. From these rapid advances, the microcomputer was born. The versatility and ease of operation of these remarkable tools holds much promise in their application to the control of

machine tools. In 1979 Ebeling, Stanislov and Raley successfully interfaced a microcomputer with a numerically controlled mill [5].

From Eli Whitney's first mill to the current numerically controlled machines the common thread that characterizes the development of the milling machine is the quest for more control and more precision. The microcomputer appears to offer designers another tool to effect this control. Although relatively inexpensive microcomputers are available, the interface between the microcomputer and the milling machine are not available in off-the-shelf black boxes [6]. Instead each interface must be designed step by step to insure successful system operation.

1.2 PROBLEM

To design, build and test a micro-computer system to control a milling machine, creating a two axis contour milling capability.

1.3 METHOD

The process of converting a milling machine to a computer controlled two-axis contouring machine was divided into two phases -1) design and 2) constructio The design phase addressed three specific areas. a) Lead screw power requirements were estimated and resulted in the selection of the appropriate electric motors. b) Motor drive circuits were designed. c) A program was developed that inscribed a cross in a square. The construction phase built, tested, modified and retested the designs developed in the first phase. The workpiece material was acrylic plastic.

1.4 EQUIPMENT

The milling machine is a Sherline Vertical Mill. The spindle is powered by a 1/5 HP motor with a variable speed control. The spindle speeds can be varied from 200 to 2000 rpm. The work table measures 330mm x 70mm with 229mm of movement on the X axis and 76mm on the Y axis. From its uppermost position on the Z axis, the spindle can move 165mm. The movements on all axes are controlled by handwheels to leadscrews. The leadscrews advance loads 1mm for

each revolution of the handwheel.

The microcomputer used is an MMD-1 manufactured by E & L Instruments, Inc.* The MMD-1 features an 8080A Central Processor. The random access memory has a capacity of 512 words. This memory was expanded through the use of an M1 board that provides an additional 1024 words. The expansion board also provides an audio recorder interface. Data is entered by the use of the keyboard. All programs are entered in machine language. Output is provided at special interface sockets or at three latched output ports.

* Manufactured by E & L Instruments, Derby, Ct. 06418.

CHAPTER 2

DESIGN

2.1 GENERAL CONCEPT OF OPERATION

To convert a milling machine to a two-axis contouring machine, the manual control system for the X-and Y-axes must be replaced with a system controlled by electric motors. Circuits to control the drive system must be designed to interface the milling machine with the microcomputer. Fig. 1 depicts, in general terms, the concept of operation after this conversion. A machine language program is loaded into the microcomputer, the microcomputer outputs the necessary data to the circuits that control the motors. The motors position the workpiece relative to the cutter.

2.2 MACHINABILITY DATA

Since the converted machine will be a two-axis (X & Y) contouring machine, the X-and Y-axes will be simultaneously controllable. The Z-axis will be controlled to position the cutter prior to and subsequent to contouring operatio

An initial point of departure in designing the conversion of the milling machine was to examine the recommended machinability data for the material or materials to be machined. In this case acrylic plastic was to be machined. Tables 1 lists the recommended machinability data for acrylic plastics

Table 1. Machinability Data for Acrylic Plastic [7].

Surface Speed	1.5 to 3.0 meter/second
Tool Material	High speed steel or carbide
Feed	.015 to .250 mm/tooth

Using a cutting speed S mps, and a cutter, of D mm diamter, the required spindle N speed in rps is:

$$N = S \times 1000/\Pi D/1000$$

For a 2.4 mm diameter milling cutter with two flutes and a surface speed of 1.5 mps, the calculated spindle speed is

$$N = 1500/\Pi\ (2.4) = 199 \text{ rps}$$

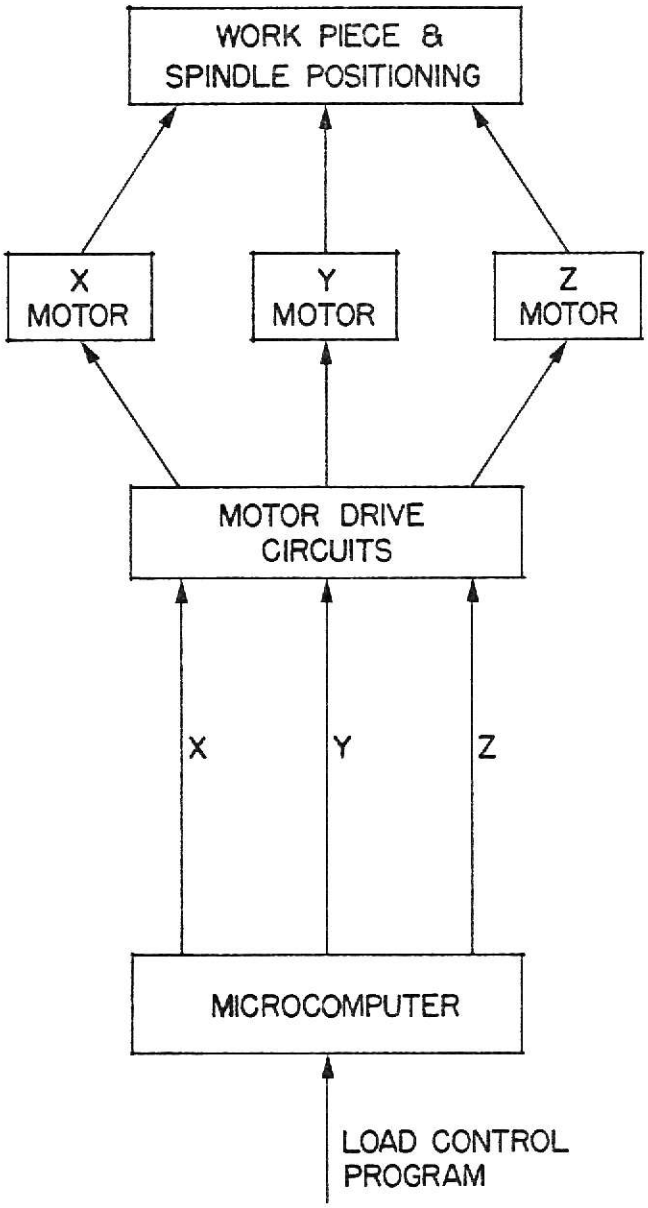


Fig. 1. General Concept of Operation

The feed is given by the following expression

$$F = F_t nN \qquad [2]$$

where F_t = feed per tooth

n = number of teeth

N = spindle speed in rps

For a two fluted cutter and a 0.075 mm/tooth feed, the desired feed is:

$$F = 0.075 \times 2 \times 199 = 29.85 \text{ mm/sec}$$

Based upon the recommended machinability data a spindle speed of 199 rps and a feed of 29.8 mm/second should be used when machining acrylic. However, the limitations imposed by the Sherline milling machine dictate a maximum spindle speed of 33.33 rps. The corresponding feed is 5 mm/second. The mill is unable to operate at the recommended feed and speed and thus a less than optimum cut may result. Since a high quality cut is not of paramount importance in demonstrating microcomputer control of this milling machine, speeds within the operating range of Sherline mill will be used. It should be noted that in applications where the strict adherence to the recommended speed and feeds is important, it may become necessary to redesign the spindle drive system so that recommended speeds and feeds are attainable.

2.3 MOTOR CHARACTERISTICS

Before discussing the leadscrew power requirements, it is necessary to examine the pertinent characteristics of the two types of electric motors to be used in positioning the part. First under consideration is the split phase reversible, AC motor with capacitor start and run. This motor provides for relatively high constant torque when connected to a 110v AC source. The 110v AC motor is not directly compatible with the 5v DC microcomputer [6]. In addition, acceleration circuits, deceleration circuits, and position sensing circuits must be used with an AC motor functioning in precise positioning applications. The second type of motor used is the stepper motor. The stepper motor is a DC permanent magnet motor whose rotor rotates a specific amount each

time a winding is energized. In addition the stepper motor has a holding torque which acts as a brake and serves to hold the load in the desired position. The stepper motor, operating on 5v DC, is compatible with the microcomputer. Based upon these characteristics the stepper motor appears ideally suited to drive the X- and Y- axes, while an AC motor appears to be a suitable candidate to raise and lower the spindle on the Z-axis. Before the specific motors can be selected; however, a detailed look at power requirements is needed.

2.4 LEAD SCREW POWER REQUIREMENTS

In order to estimate the torque that each stepper motor must provide to move a workpiece on the X- and Y- axes, the following relationships were used:

Equation 1. $T = T_L + T_F$ [8]

T = total torque load (millinewton metre (mN·m))

T_L = torque required to accelerate an inertial load (mN·m)

T_F = frictional torque (mN·m)

Equation 2. $T_L = J_T a$ [8]

J_T = total moment of inertia (kilogram metre^2 (kg·m^2)

a = acceleration (radians/sec^2)

Equation 3. $J_T = J_R + J_S + J_{RL}$ [8]

J_R = moment of inertia of the rotor of the motor (kg·m^2)

J_S = moment of inertia of the steel leadscrew (kg·m^2)

J_{RL} = reflected moment of inertia of the load (kg·m^2)

Equation 4 $J_S = D_4\ L\ \rho_S\ \Pi/32$ [8]

Where D = diameter of lead screw in metres

L = lead of screw in metres

ρ_S = density of steel kg/m^3 = 7870 kg/m^3 [9]

Simplifing equation 4 yields the following result

$J_S = D^4 L \times 7.7 \times 10^2$ [7]

The expression for the reflected inertia of a load when connected to a linear screw thread is:

Equation 5 $$J_{RL} = M(\frac{L}{2\pi})^2 \quad (7)$$

M = mass of load - kg

L = lead of screw - m

Using the above relationship to calculate the torque required to drive the X-and Y-leadscrews gives the results in the sections that follow.

X-axis calculations

To calculate the mass of the aluminium work table and X-axis leadscrew, the following expression was evaluated.

$$M_x = M_w + (V_x \cdot \rho\ al)$$

M_x = mass of x axis load (kg)

M_w = mass of workpiece and fixtures (kg)

V_x = estimated volume of table (m^3)

ρ_{al} = density of aluminium (kg/m^3) = 2700 kg/m^3 (9)

Assuming an M_w = 0.5 kg and estimating the volume of the table to be 498 $cm.^3$

$$M_x = 0.5 + (498 \times 10^{-6} \times 2700)$$

$$M_x = 1.84 \text{ kg}$$

Using equation 5 with a .001 m lead, the moment of inertia of the reflected load is:

$$J_{RL} = 1.84\ (\frac{0.001}{2\pi})^2$$

$$J_{RL} = .046 \times 10^{-6}\ kg \cdot m^2$$

The rotor moment of inertia listed in the engineering specifications for the motor under consideration was $9.3 \times 10^{-6}\ kg \cdot m^2$.

Using equation 4 with a lead screw with a diameter of .007 m and a lead of .001 m, the moment of inertia of the steel lead screw is

$$J_S = (.007)^4(.001)\ (770)$$

$$J_s = 1.8 \times 10^{-9} kg \cdot m^2$$

Using equation 3 the total moment of inertia is expressed as

$$J_T = (9.30 + .046 + .00185) \times 10^{-6}$$

$$J_T = 9.35 \times 10^{-6} kg \cdot m^2$$

In order to calculate the torque required by the X-axis stepper motor, it is necessary to assume a stepping rate compatible with the spindle speed of the mill and the machinability data for the acrylic plastic. Earlier in this chapter it was shown that the mill was not capable of developing the recommended feed and speed for milling acrylic plastic. The maximum speed that the mill can develop is 33.33 rps. The corresponding feed for this speed is 5mm/sec. If the stepper motor makes 96 steps/revolution and the lead of the screw is 1mm, then the maximum stepping rate is 480 steps/sec. Thus the stepping rates that are compatible with this mill range from a minimum of 0 steps/sec to 480 steps/sec. Selecting the midpoint 240 steps/sec and calculating the acceleration yields:

$$a = \frac{(240)^2}{2\Pi} \times \frac{2\Pi}{96} \qquad [8]$$

$$a = 1884 \text{ rad/sec}^2$$

Multiplying the acceleration by the total moment of inertia gives the torque required to accelerate the load from 0 rad/sec to 240 rad/sec with a maximum lag of 2 steps.

$$T_L = (8.35 \times 10^{-6}) \times (1884)$$

$$T_L = 17.6 \text{ mN} \cdot \text{m}$$

The calculation above does not consider friction. The frictional torque however, cannot be overlooked.

In order to provide some estimate of the magnitude of this torque, weights were attached to a string and the string was attached to the handwheel. The amount of weight required to cause the handwheel to move was noted. This

value was then multiplied by the distance from the string to the centerline of the leadscrew and yielded an estimate of 42 mN·m.

Using equation 1 the torque required to accelerate a .5 kg load on the X axis is

$T = 17.6 + 42$, or

$T = 59.6$ mN·m

Y axis calculations

The calculations for the required torque to drive a Y axis load is very similar to the X axis calculation; however, the mass of the load is different because the Y axis load includes the X axis load plus the saddle.

$M_Y = 2.17$ kg

$J_{RL} = .055 \times 10^{-6}$ kg·m^2

$J_S = 1.8 \times 10^{-9}$ kg·m^2

$J_R = 9.300 \times 10^{-9}$ kg·m^2

$J_T = 9.36 \times 10^{-6}$ kg·m^2

Using a stepping rate of 240 steps/sec

$T_L = J_T = 17.6$mN·m

The torque due to friction on the Y axis was measured in the same manner as the X axis. There was no measurable difference and $T_F = 42$ mN·m. Thus total torque required to accelerate a .500 kg workpiece on the Y axis is

$T = 17.6 + 42$, or

$T = 59.6$ mN·m

Z axis calculations

The procedure for calculating the torque required from the AC motor driving the Z axis lead screw is developed below (10). Figure 2 is a simplified force diagram that depicts the relationship between forces acting along the Z axis.

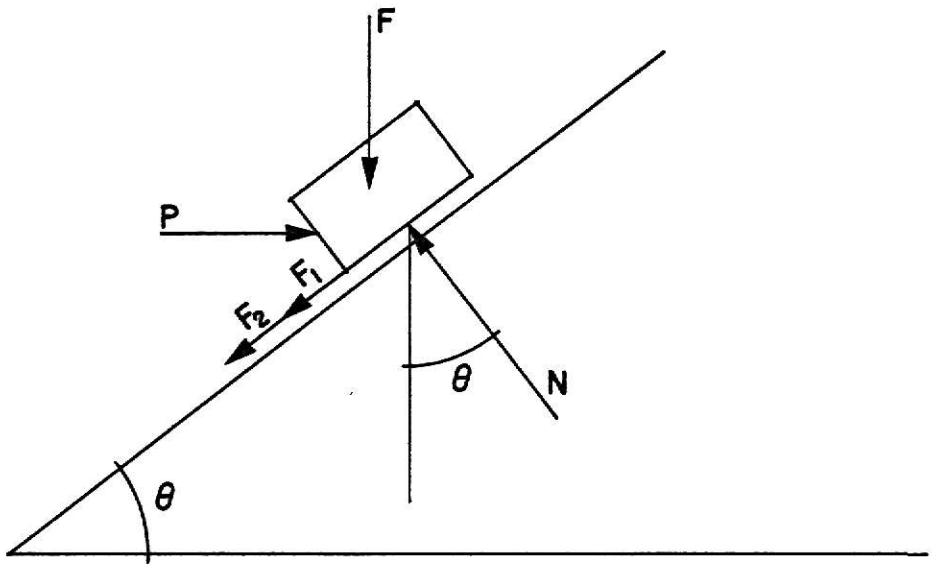


Fig. 2. Simplified Force Diagram Z-Axis

F = force exerted due to the weight of the load (mn)

P = force required to push the load up the incline plane (mN)

N = normal force (mN)

F_1 = force of friction caused by the steel lead screw turning through a brass nut (mN)

F_2 = force of friction caused by the aluminium slide moving against brass ways (mN)

$\tan \theta = \frac{L}{2\ R}$

L = lead of screw, m

R = radius of lead screw, m

$F_1 = N\ \mu_1$

$F_2 = N\ \mu_2$

μ_1 = coefficient of friction between steel and brass assume $\mu_1 = 0.5$ [9]

μ_2 = coefficient of friction between aluminium and brass assume $\mu_2 = 0.6$ [9]

The load on the Z axis consists of the slide, the spindle drive motor and assorted mounts. This load was measured as 2951 g.

To find F the following relationship was used

$F = MA$

$F = (2951) \times 9.8$

$F = 28920$ mN

Summing the forces in the Y direction gives the following

$F_y = N \cos \theta - F - .5N \sin \theta - .6N \sin \theta = 0$

$.9645N = 28920$

$N = 29984nM$

Summing the forces in the X direction gives the following

$F_X = P - N(\sin \theta + .5 \cos \theta + .6 \cos \theta) = 0$

Substituting N = 29984 mN gives the following result

P = 33912 mN

Since P acts at a distance R from the centerline of the lead screw the following relationship was used to give an estimate of the torque required to raise the load.

T = PR

T = torque, mNm

R = radius of the lead screw, m

T = 33917 x .005

T = 169.5 mNm

Table 2 gives the characteristics of the AC motor chosen to drive the Z axis and the characteristics of the DC stepping motors chosen to drive the X and Y axis.

Table 2 Motor Characteristics

				Torque mNm	
Type	Make	HP	RPS	Start	Run
AC	Dayton	1/100	113	2040	1474

Type	Make	Step/Rev	Step Angle	Pull Torque IN/OUT mNm	Max steps/s Pull-in rate
DC stepping motor	Phillips	96	3°45'	63.54	240

[12]

It should be pointed out that the foregoing calculations provide an estimate of the required torque and provide a reasonable basis for preliminary motor selection. The selection of the gear ratio and DC voltage will provide the means for making the final adjustments to insure that the motors are capable

of driving the loads. Although the DC stepper motor is specified as 5v, slightly higher voltage will result in a higher torque without detrimental effect on the motor.

2.5 AC MOTOR DRIVE CIRCUIT

The AC motor dirve circuit serves as the link between the microcomputer and the Z-axis drive motor. The circuit consists of three solid state relays shown on Fig. 3 and five integrated circuit chips shown on Fig. 4. In order to examine the operation of this circuit, it is necessary to begin with the motor and learn just what makes this motor function. The motor is a Dayton 1/100 HP AC motor. The motor has three power input leads, a black common lead, a gray lead to the winding used for counterclockwise rotation of the motor, and a yellow lead to the winding that is used for clockwise rotation. In order to make this motor rotate in a clockwise direction 110v AC must be present between the yellow and black leads. When the 110v AC is applied between the gray winding and the black winding, we get counterclockwise rotation. Knowing this, how do we switch the motor on and off and change the direction of rotation? Fortunately, solid state relays provide the 110v switching function and still maintain their membership in the digital world. Typically, solid state relays consist of a light emitting diode (LED) and a photo transistor. When the LED is energized the light produced triggers the photo transistor which in turn permits current to flow. Thus on the LED side of a solid state relay the +5v DC digital world prevails; while on the opposite side 110v AC dominates. In order for the Z-axis motor to function three solid state relays are needed. One relay acts as an ON and OFF switch of the common side; the second relay controls the ON and OFF functions of the clockwise winding and the third relay controls the ON and OFF functions of the counterclockwise winding. In order to make the motor

DRIVE CIRCUIT Z-AXIS AC MOTOR RELAYS

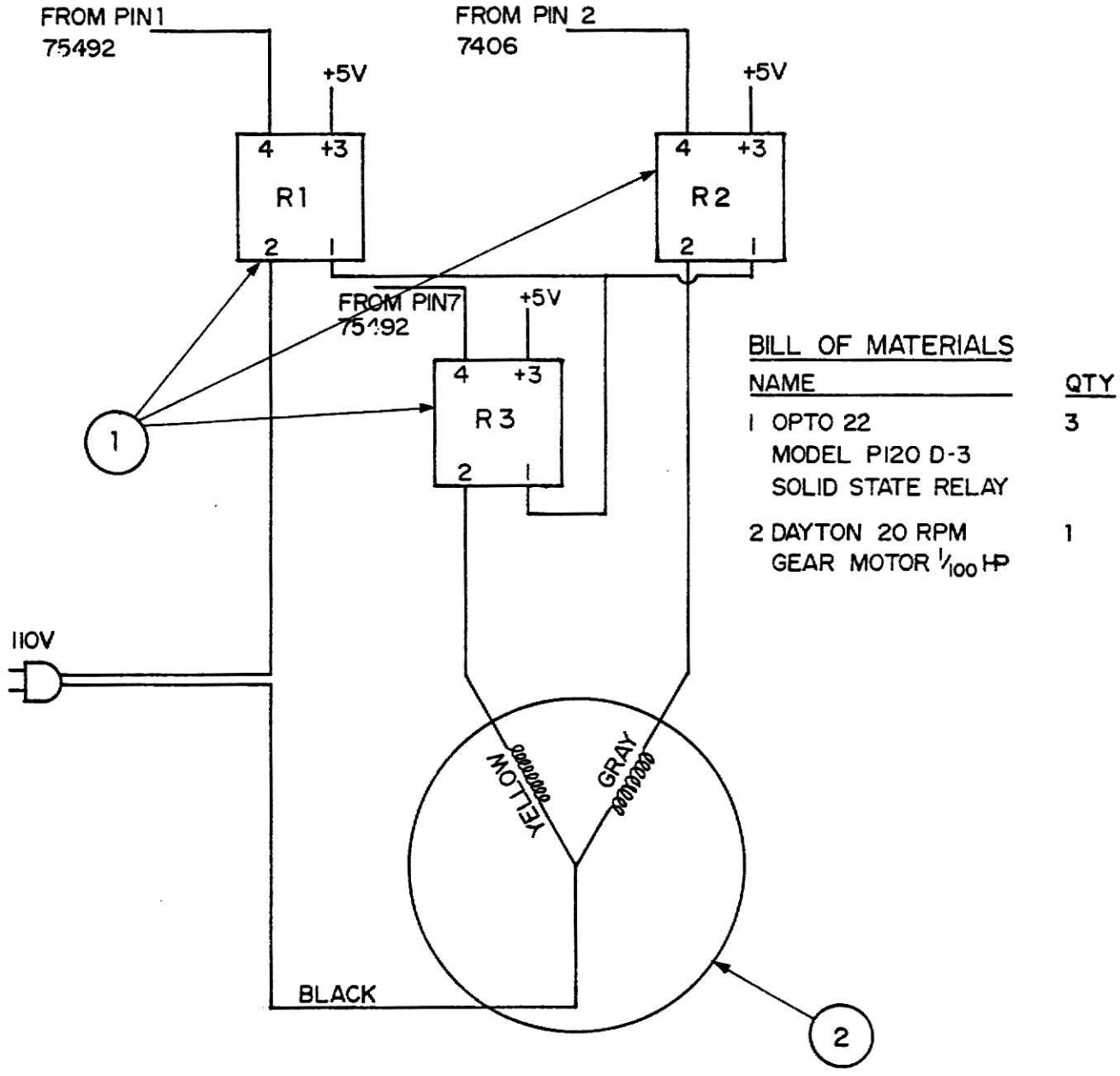
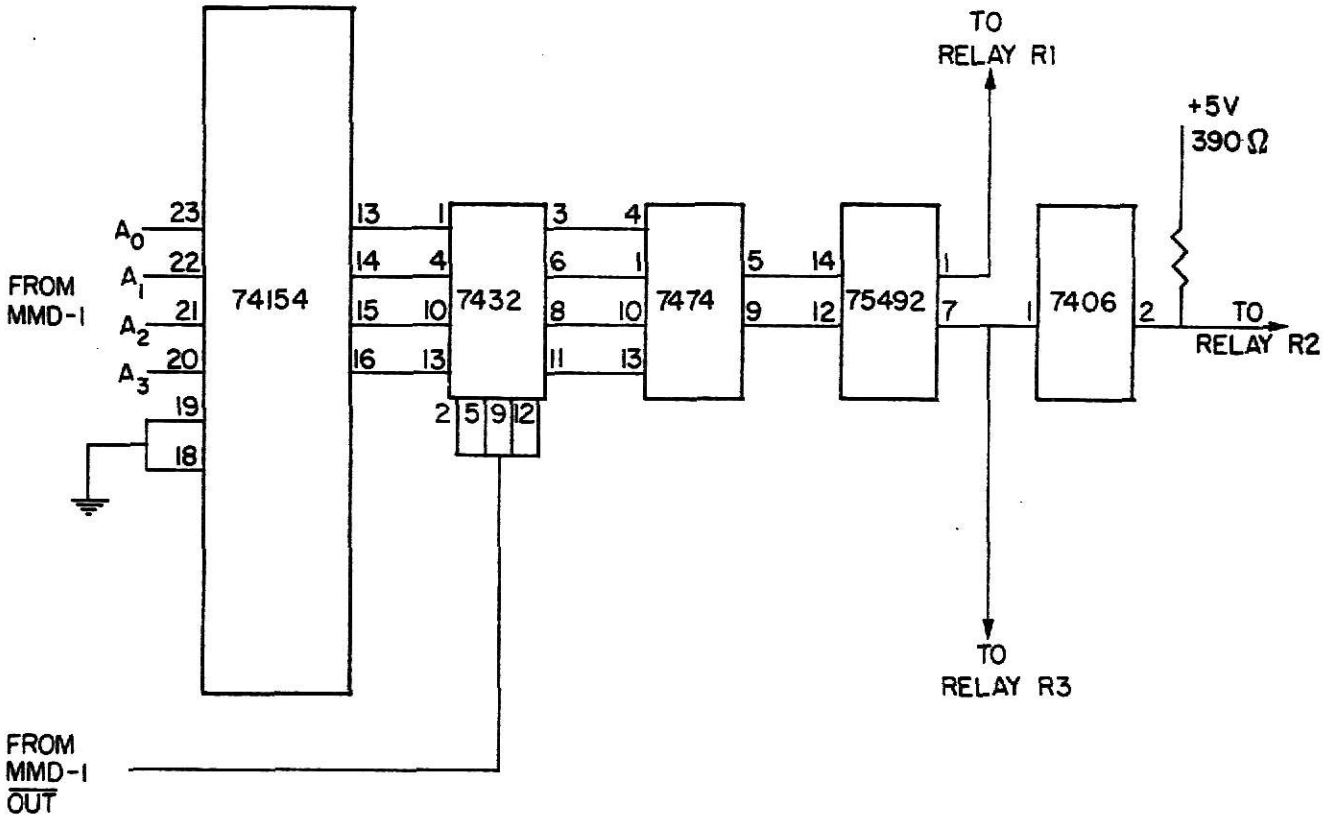


Fig. 3. Z-Axis Relays

DRIVE CIRCUIT Z-AXIS AC MOTOR



PIN HOOK UP

TYPE	GROUND	+5V	MISC PIN CONNECTIONS
74154	12	24	18, 19 - GROUND
7432	7	14	2, 5, 9 & 12 - $\overline{OUT}$ MMD - 1
7474	7	14	2, 12 - GROUND
75492	4	11	1 to 2, 3 to 14, 7 to 9 to 13, 8 to 10 to 12
7406	7	14	2 to +5V thru 390Ω RESISTOR

Fig. 4. AC Motor Drive Circuit

rotate in a counterclockwise direction, a signal must be sent from the microcomputer to energize relay 3, then another signal must be sent to relay 1 to start the motor. Moving back from the +5v side of the relay, the first integrated circuit chip encountered is the SN 7406 IC (13). This chip functions as an inverter and takes the signal going into one of the windings and inverts it. The signal to the SN 7406 comes from an SN 75492 IC which is an inverter buffer [14]. This chip inverts and boosts the power of each input signal sent to one of its input gates. These signals come from an SN 7474 IC, a latch [13]. This chip holds the signal and permits the signals to be viewed downstream as separate and distinct. The input to the SN 7474 comes from an SN 7432. This chip is an OR gate. The logic table for an OR gate is shown in Table 3.

Table 3 Logic Table for OR Gate [13]

A	B	OUTPUT
0	0	0
1	0	1
0	1	1
1	1	1

The B input of all gates on this chip have been connected to outport of the MMD-1, thus this chip will only output a 0 when the A and B inputs are logic 0. The input to the 7432 comes from the 74154, which is a 4-line to 16-line decoder. The logic table for this chip is shown in Table 4.

Table 4 Logic Table for SN 74154 IC [13]

INPUTS				OUTPUTS													
D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

In order to energize relay 1 to turn the motor ON, a logic 0 must be present on the input side of the relay. Since the SN 75492 IC inverts the signal, a logic 1 must be input to the appropriate gate of the SN 75492 IC. This signal comes from the SN 7474 IC latch whose output goes to logic 1 when the input to the SN 7474 IC preset pin is at logic 0. This condition persists until changed by inputting a logic 0 to the clear pin of the SN 7474 IC. The logic 0 at the preset or clear pin is output from the SN 7432 IC when both inputs to the OR gate are logic 0. The input to the SN 7432 IC comes from pin 13 of the SN 74154 IC. Referring to the logic table for the SN 74154 IC chip, to get a logic 0 at pin 13 the input to D, C, B, and A must be 1, 0, 1, 1 respectively. Converting this to octal code 00 001 011 which is 013 decimal. Thus a 013 decimal input to the SN 74154 IC will cause the motor to start,

while a 014 decimal will cause the motor to stop. The microcomputer is used to output these codes to the motor drive circuit by using the two byte OUT instruction. The first byte is the machine language code 323 which means OUT. The second byte is the device code, in this case 013 or 014. In executing the OUT instruction, the microcomputer outputs a signal at the same time the device code 013 or 014 is being output to the address bus. Thus by connecting the input pins of the SN 74154 IC to the address bus of the MMD-1 and the OUT terminal of the MMD-1 to one side of an OR gate, the unique logic 0 may be input to the Preset or Clear pin on the SN 7474 IC. The input code 013 will result in a logic 0 to the Preset of the SN 7474 IC which in turn will turn the motor ON while a code 014 will result in a logic 0 at the Clear of the SN 7474 IC which in turn will cause relay 1 to turn the motor OFF.

The direction of rotation is controlled by activating relay 2 or relay 3 which control clockwise and counterclockwise rotation respectively. The input code 015 and 016 cause the Preset of the SN 7474 IC to be at logic state 0 this in turn causes the relay controlling the particular winding to close and the motor will rotate in the desired direction. In addition the input signal also switches the relay connected to the other winding to the open position. This signal inversion is performed by the SN 7406 IC.

The circuit described above will start and stop and change the direction of rotation of the AC drive motor. When intergrating this circuit with software, incorporating a 10 millisecond delay after outputting the input codes to the circuit will allow the relay sufficient time to react to the signals. The specifics of these delay routines will be given in the chapter on programming.

2.6 DC MOTOR DRIVE CIRCUIT

The DC motor drive circuit serves as the link between the microcomputer and the X and Y axis drive motors. The circuit consists of one integrated circuit chip and four transistors for each axis. The operation of the DC motor is depicted in Fig. 5a and 5b. A permanent magnet with its center attached to a rotor, is centered among four windings. A current flows through the windings, the permanent magnet is attracted or repelled depending upon its polarity. This action produces torque at the rotor. If a logic 1 or ON condition produces an attractive force on the north pole of the permanent magnet, and a logic 0 or OFF produces no current flow in the windings, then a series of logic 1's and logic 0's, in the proper sequence will cause the rotor to rotate. For instance, in Fig. 5a, when a logic 1 is input to winding A and the remaining windings are at logic 0, the north pole of the magnet will be attracted to winding A. In Fig 5b, when logic 1 is maintained at winding A and a logic 1 is input to winding D with winding B and C at logic 0, then the magnet will rotate in a clockwise direction to a position halfway between windings A and D. If counterclockwise rotation is desired, the sequence in Table 5 should be used.

Table 5 Counterclockwise Code for Stepper Motors

OCTAL CODE ABCD	DECIMAL
1000	010
1010	012
0010	002
0110	006
0100	004
0101	005
0001	001
1001	011
1000	010

PRINCIPLES OF OPERATION
DC STEPPER MOTOR

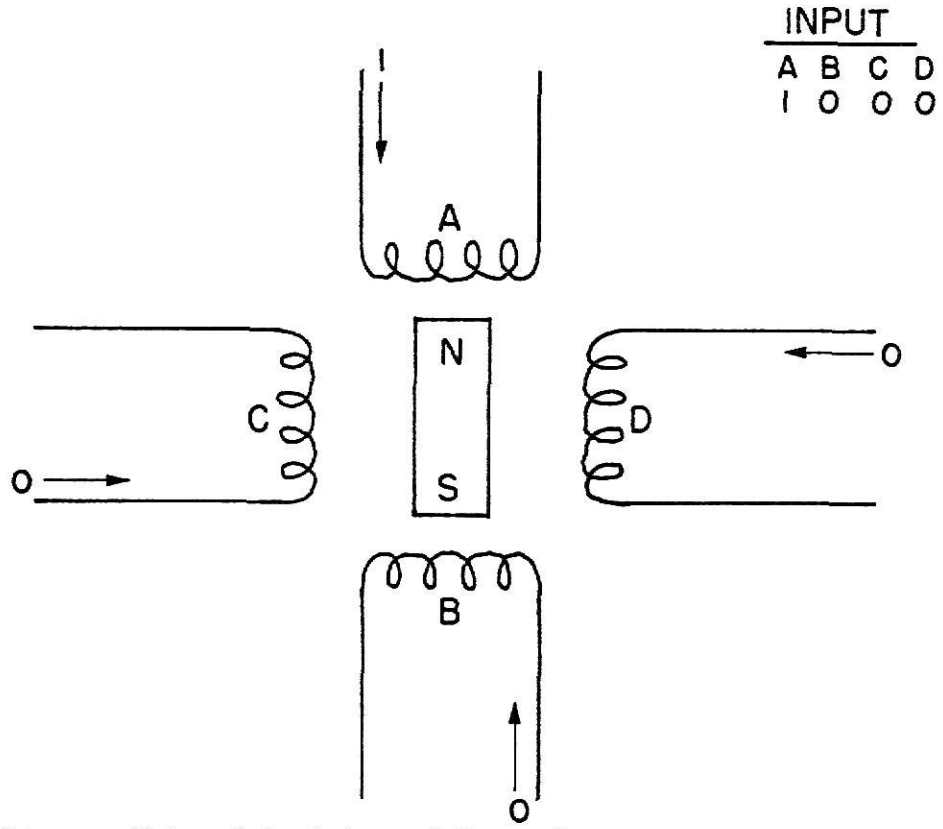


Fig. 5a. DC Stepper Motor Principles of Operation.

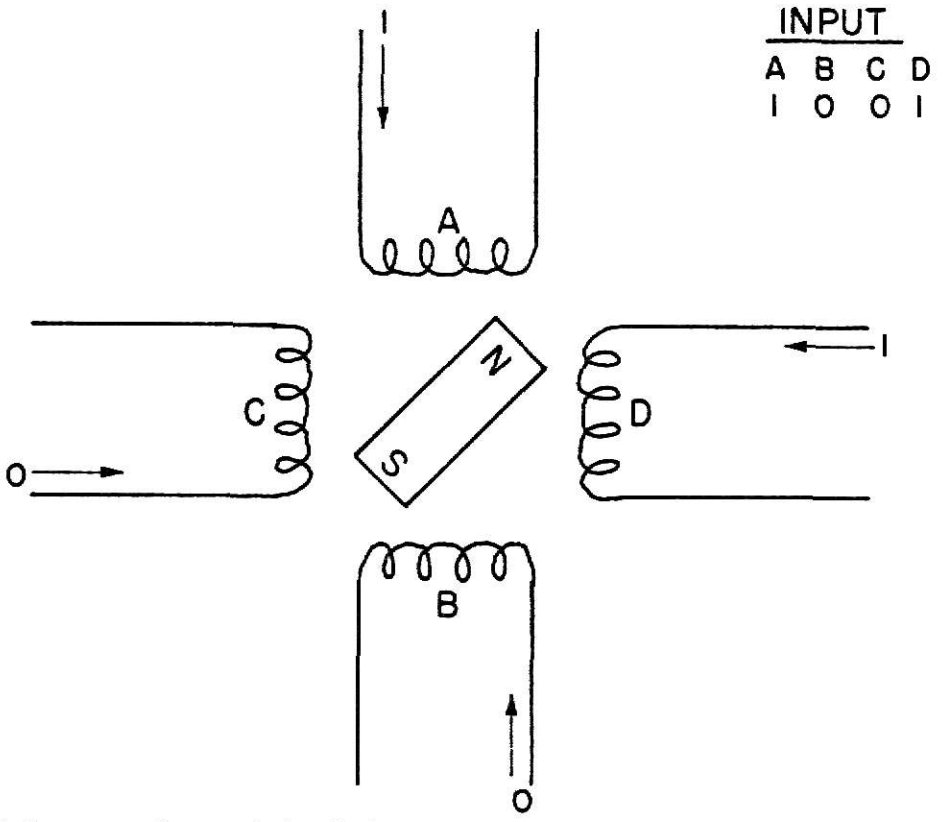


Fig. 5b. DC Stepper Motor Principles of Operation

If clockwise rotation is desired the sequence in Table 6 is used.

Table 6 Clockwise Code for Stepper Motors

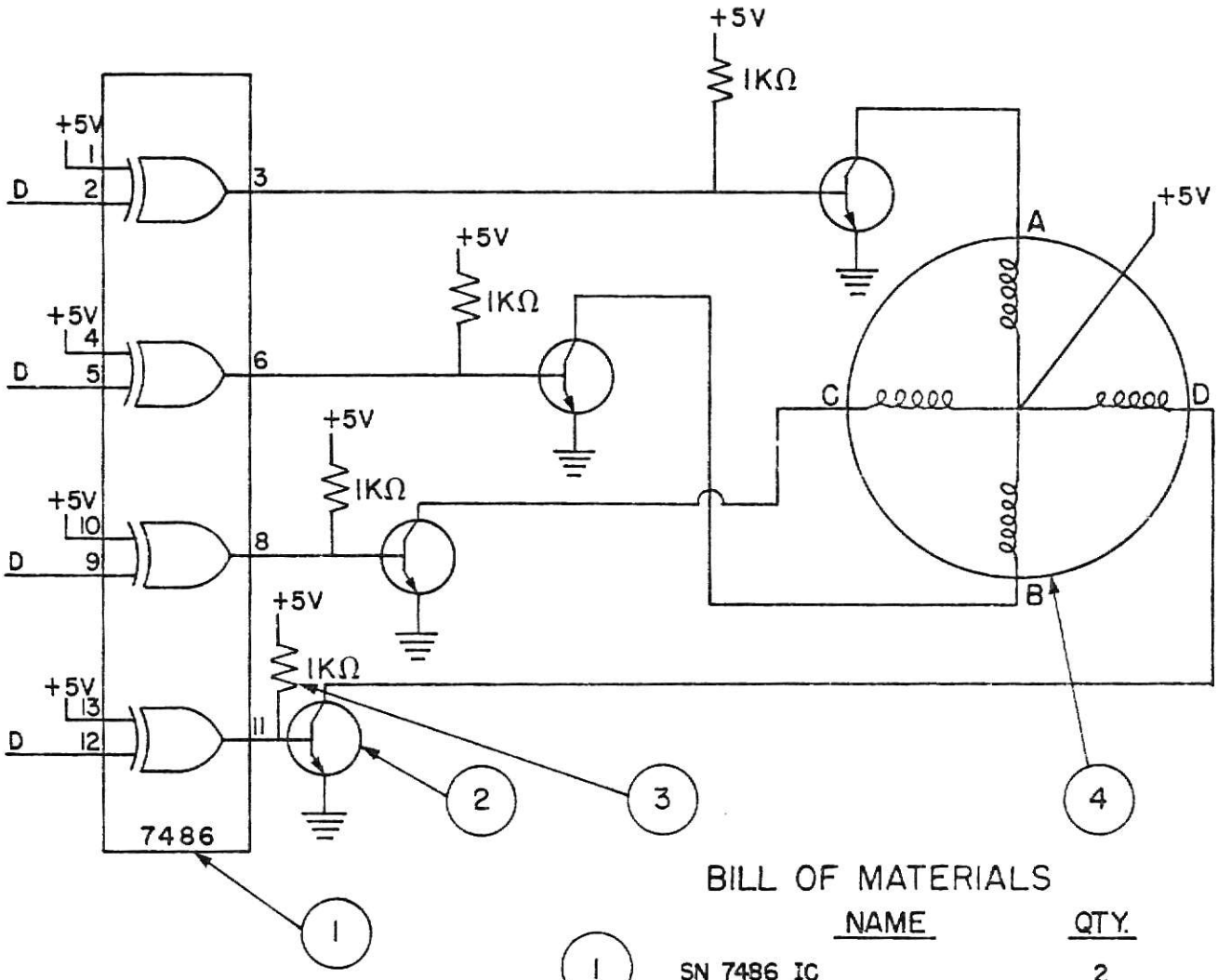
OCTAL CODE ABCD	DECIMAL
1000	010
1001	011
0001	001
0101	005
0100	004
0110	006
0010	002
1010	012
1000	010

Applying these rotation codes to the Phillips DC logic motor, will result in the motor taking steps of 3.75°. Inputting this sequence one time to the windings will cause the rotor to take 8 steps of 3.75° each. Inputting the sequence 12 times will cause the motor to turn through 360°.

The rotational code sequence can be output through the ports of the MMD-1 with port 0 controlling the X axis motor and port 2 controlling the Y axis motor. Since each port is already latched, the use of these ports eliminates the need for latching in the drive circuit external to the MMD-1.

From these ports the signals are sent to one of the input pins of the four exclusive-OR gates on the SN 7486 IC as shown in Figure 6. The other input pin on each gate is connected to +5v. The logic table for the exclusive-OR gate is shown in Table 7.

DRIVE CIRCUIT X&Y AXIS DC STEPPER MOTOR



PIN CONNECTIONS

1, 4, 10, 13, 14	+5V
3, 6, 8, 11	TIP29
X AXIS 2, 5, 9, 12	MMD-1 PORT 0 D13, D14, D15, D16
Y AXIS 2, 5, 9, 12	MMD-1 PORT 2 D21, D22, D23, D24
7	GROUND

BILL OF MATERIALS

	NAME	QTY.
1	SN 7486 IC	2
2	NPN TRANSISTOR TIP29	8
3	RESISTOR 1KΩ	8
4	PHILLIPS DC LOGIC MOTOR	2

Fig. 6. DC Stepper Motor Drive Circuit

Table 7 Exclusive-OR Logic Table [13]

INPUTS	OUTPUTS
A B	
0 0	0
0 1	1
1 0	1
1 1	0

When connected in the manner described earlier, the exclusive-OR gate will cause an inversion of the input signal. An input of logic 0 will result in an output of 1. This output is input to the base of a transistor. The base is pulled up to +5v through a 1K ohm resistor. When a logic 0 is input to the base, there is a difference in potential across the 1K ohm resistor and current flows. This action, in turn, permits current to flow from the +5v winding center tap through the appropriate windings to the collector of the transistor and through the emitter to ground. Since each winding is connected to its own exclusive-OR gate and transistor, the rotational code output from the MMD-1 may be used to control the rotation of the motor.

CHAPTER 3

CONSTRUCTION

CONSTRUCTION

The construction phase was divided into two parts. The first phase includes the mounting of the electric motors on the milling machine. The milling machine was modified along the X and Y axis to permit secure fastening of the motors. In addition, modifications were required to accomodate the extension of the X and Y leadscrews. A 20° pressure angle spur gear with 20 teeth and a pitch diameter of 16 mm was mounted on the X and Y leadscrews. A 20° pressure angle gear with 10 teeth was connected to each motor shaft. The Z-axis lead screw was geared to the AC motor using a gear ratio of 1 to 1.44. Detailed drawings of the modifications and motor mounts are shown in Appendix A.

The second part of the construction phase required the breadboarding and testing of each of the motor drive circuits previously described. After each circuit was successfully tested, the intergrated circuit chips and other components were hardwired to a printed circuit board.

CHAPTER 4

PROGRAMMING

PROGRAMMING

In programming an MMD-1 to control the milling machine two general classes of programs were used. First there are the axis movement programs that produce up or down movement on the Z-axis and + or - movement along the X-and Y-axis. These programs produce motion by causing the MMD-1 to output code on a sequence of codes to the motor drive circuits. The second type of program was the delay subroutine. These subroutines are called by the movement programs to control the time the AC motor is on, the DC motor stepping rate and number of steps taken by the DC motor.

An example of a program to move the spindle down is shown in Appendix B. In this program the code 015 is output to the AC motor drive circuit. A 10 milli-second delay is called to provide time for the relay to react. The program then outputs a 013 code which causes the motor to move the spindle down. The distance that the spindle travels is dependent upon the length of time that the motor is ON. That time is controlled by calling the delay program which, in this case, is located in memory at address Hi 033 Lo 300. This program is shown in Appendix C. By initializing this program with specific values, predetermined delays may be achieved. Table 8 gives the initial values and the resulting delays.

Table 8 Z-Axis Length of Travel Initial Timing Values

Travel mm	Register L
1	1
2	2
3	3
4	4
5	5

To control movement along the X- or Y- axis, the rotation code in the proper sequence must be output to the DC stepper motor circuit. For illustrative purposes the X+ program is shown in Appendix D. This program initializes register L with a value. This value determines the distance traveled in the X+ direction. Table 9 gives the initial values and the resulting distances traveled. The program then moves the first code of the sequence into the accumulator, in this case the code is 010.

Table 9 (X + Y Axis) Length of Travel Initial Timing Values

Travel mm	Register L	Register H
1	30	
2	60	
3	110	
4	140	
5	170	
6	220	
7	250	
8	280	
9	330	
10	350	
20	350	002
30	350	003
50	350	005

The 010 is output from the accumulator to port 0 and input to the DC motor drive circuit for the X-axis. The program then calls the rotation delay subroutine which determines the delay between steps. Table 10 gives the initial values and the resulting stepping rates. Upon returning to the main program the next sequence code is moved to the accumulator, then output thru port 0 to the X-axis motor drive circuit. Again, the rotation delay subroutine is

Table 10 Stepping Rate Timing Values

Register:	B	C	D	E
Stepping Rate Steps/second				
128	4	5	3	4
153.6	4	5	3	2
166.4	4	4	3	4
179.2	3	5	3	4
192	4	5	2	4
217.6	4	3	3	4
256	2	5	3	4
320	4	2	3	4

To obtain desired stepping rate load decimal number above into the appropriate register in rotation delay subroutine.

called to control the stepping rate. Upon return from this subroutine, the main program causes the next sequence code to be moved to the accumulator and then continues as described above. After outputting eight sequence codes the program will decrement the value in register L by one and repeat the sequence codes again. When the value of register L is 0, the program halts.

In order to produce table motion in the negative direction along the X-axis, the counterclockwise codes must be substituted in the program in place of the clockwise codes. To control the motion along the Y-axis, the clockwise or counterclockwise codes must be output to port 2.

To mill a line with a slope of 45°, the X(-) program and the Y(-) program must be output to the motor drive circuits simultaneously. For every step along the X-axis there is a step along the Y-axis. The resulting motion of the table causes the cutter to mill a line with a slope of 45°. An example of a program to mill a 10mm line with a 45° slope is shown in Appendix E.

A program has been prepared that incorporates several of the features described in previous programs. This program will be used to inscribe a

cross in a 30mm square as illustrated in Figure 7. The program is listed in Appendix F. A generalized flow chart for the program is shown in Figure 8. The program first lowers the spindle, directs the mill to cut a 30 mm square. Next, the program directs the cutter along a path that inscribes a cross in the square. Finally, the program directs the Z-axis motor to raise the spindle, then the program comes to a halt. A picture of the finished piece using this program is shown in Figure 9. A detailed 8080 instruction set is contained in (16). An explanation of programming the MMD-1 is contained in (17).

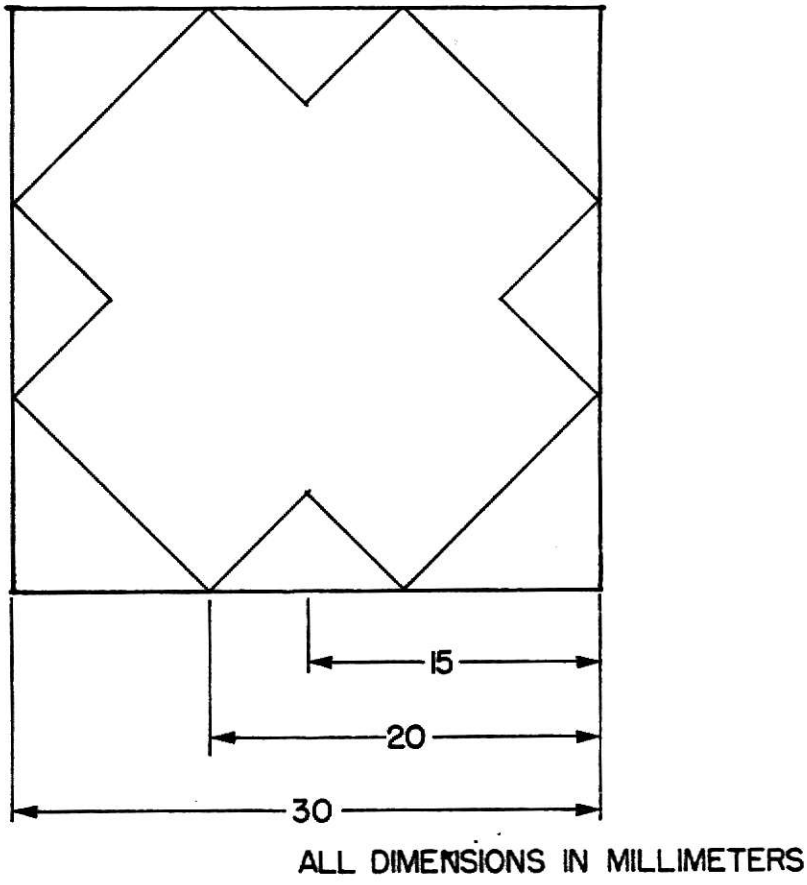


Fig. 7. Cross-in-the-Square Illustration

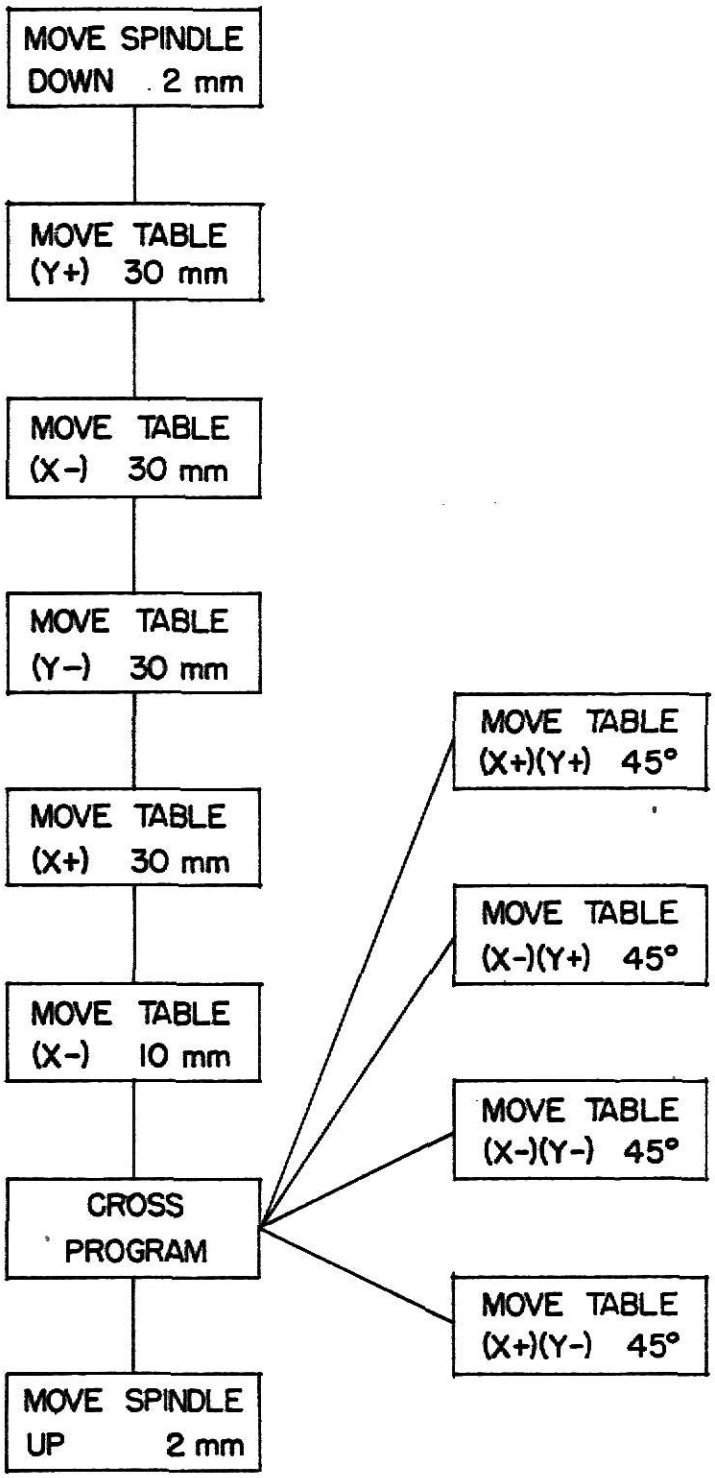


Fig. 8. Cross-in-the-Square Program Flow Chart

ILLEGIBLE DOCUMENT

THE FOLLOWING DOCUMENT(S) IS OF POOR LEGIBILITY IN THE ORIGINAL

THIS IS THE BEST COPY AVAILABLE

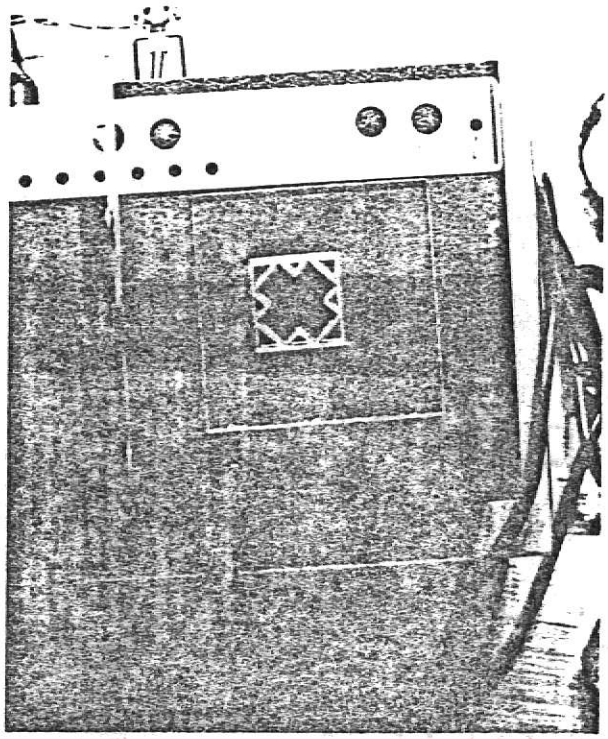


Fig. 9. Cross-in-the-Square Picture

RESULTS

A vertical milling machine with manual control has been successfully converted to a 2 axis contouring machine controlled by a programmable microcomputer. The contouring capability was demonstrated by inscribing a cross in a square piece of acrylic plastic. It took 2 minutes 30 seconds to mill the cross in the square when the mill was under the direction of the microcomputer. If the same figure was to be produced on the mill by manual control, it would take 2 additional set-ups. In addition the microcomputer control introduces a much higher degree of repeatability than one might expect using manual control.

It is significant to note that the techniques applied here on the Sherline Mill can also be applied to standard machine tools, making it possible to turn ordinary machine tools into contouring machines, extending the capabilities of these machines.

FURTHER RESEARCH

In reviewing this effort to determine new directions for further research, several areas emerge immediately. First, the software vs. hardware tradeoff needs to be examined more closely. The magnitude of the program to mill the simple cross in the square figure suggests that additional hardware is needed to provide a more efficient and flexible system of programming. Second, the expansion of the system from two to three or more axes provides a logical extension of the research effort. Finally, the use of adaptive controls to provide positive position control should be considered.

References

1. Woodbury, Robert S., The History of the Milling Machine, The M.I.T. Press Cambridge, Mass., 1960, pp. 16-102.

2. DeGarmo, E. P., Materials and Processes in Manufacturing, 4th Edition, Macmillin Publishing Co., Inc., N.Y., 1974, p. 611.

3. Pressman, R. S., and J. E. Williams, Numerical Control and Computer Aided Manufacturing, John Wiley & Sons, N.Y., 1977, pp. 4 and 289-295.

4. Teschler, Leland, "Coming: New Generations of Microcomputers," Machine Design, March 22, 1979, pp. 108-114.

5. Ebeling, Kenneth, Frank Raley and Joseph Stanislov, "A Low Cost Way to Increase NC Versatility," Manufacturing Engineering, August 1979, pp. 64-65.

6. Biegel, John E., "The Machine Tool/Microprocessor Interface," AIIE 1979 Fall, Industrial Engineering Conference Proceedings, November 13-16, 1979, pp. 404-409.

7. CADCO Engineers, "Machining the Engineering Plastics," Plastics Design and Processing, September 1960.

8. _____, Stepper Motor Handbook, North American Phillips Corp, Cheshire, Conn., pp. 12-13.

9. Avallone, Eugene A., Theodore Baumeister and Theodore Baumeister III, Mark's Standard Handbook for Mechanical Engineers, 8th Edition, McGraw Hill, New York, 1978, pp. 3-29.

10. Higdon, Archie, and William B. Stiles, Engineering Machanics, Prentice-Hall, Inc., Englewood Cliffs, N.J., May 1955, pp. 202-203.

11. _____, Connectors-Specifications 4-Phase DC Logic Drive Motor, Herbach and Rademan, Inc., Phila, Pa., May 1979.

12. _____, Grainger's Motor Book, No. 352, W. W. Grainger, Inc., Topeka, Ks., Winter 1978-1979, p. 110.

13. _____, The TTL Data Book for Design Engineers, 2nd Edition, Texas Instruments, Inc., Dallas, Texas, 1976.

14. _____, Archer Semiconductor Reference Handbook, Radio Shack, Fort Worth, Texas, 1977, p. 38.

15. Kuecken, John A., Solid State Motor Controls, Tab Books, Blue Ridge Summitt, Pa., June 1978, pp. 129-141.

16. Larsen, David G., Peter R. Rony, and Jonathon A. Titus, The Bug Book VI, E & L Instruments, Inc., Danbury, Conn., May 1977, pp. 18-7 thru 18-68.

17. Larsen, David G., Peter R. Rony, and Jonathon A. Titus, The Bug Book V, E & L Instruments, Inc., Danbury, Conn, May 1977, pp. 2-1 thru 6-32.

APPENDIX A

CONSTRUCTION DETAIL

INDEX TO DRAWINGS

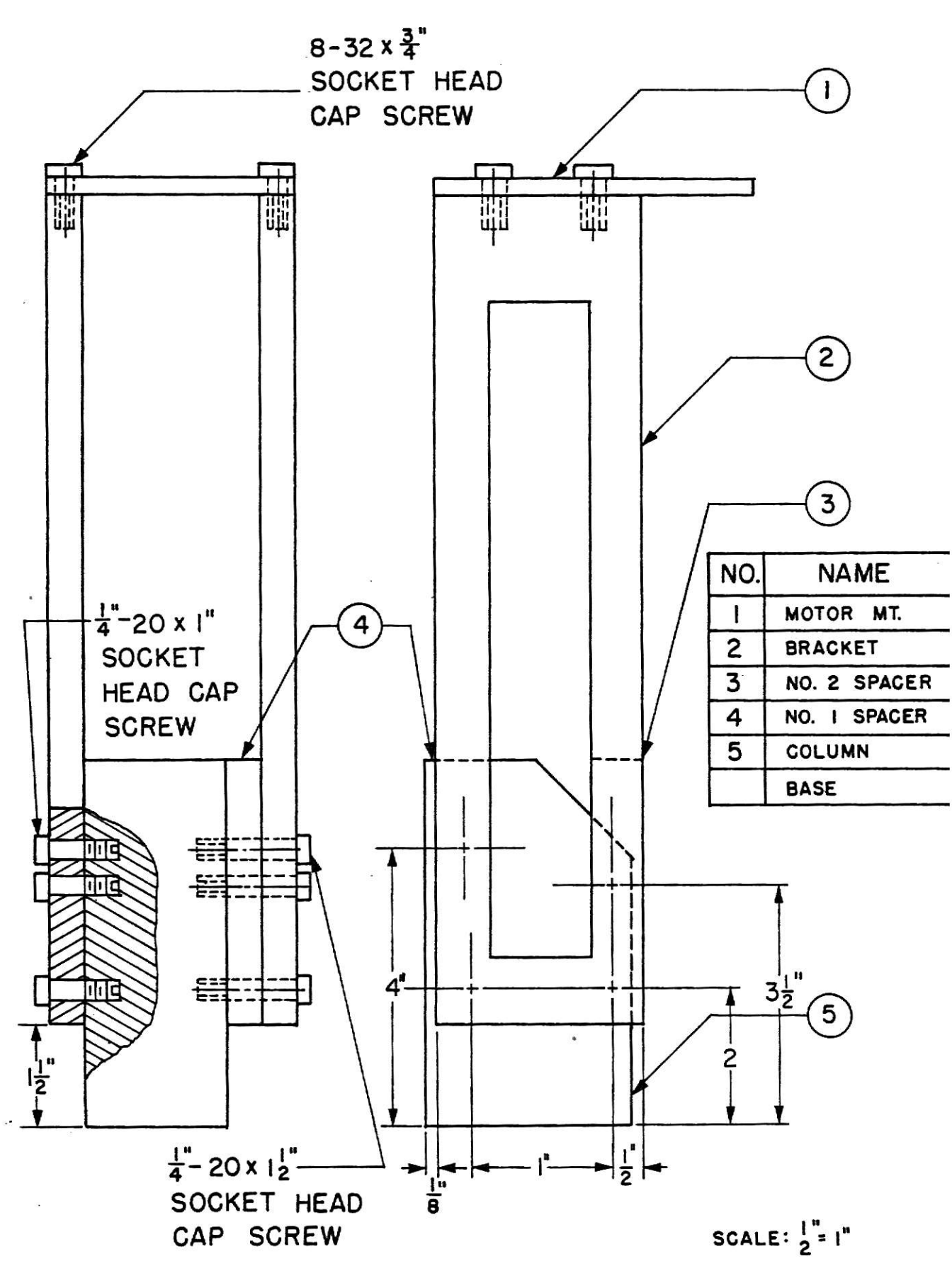


Fig. 10. Z-Axis Mount Assembly

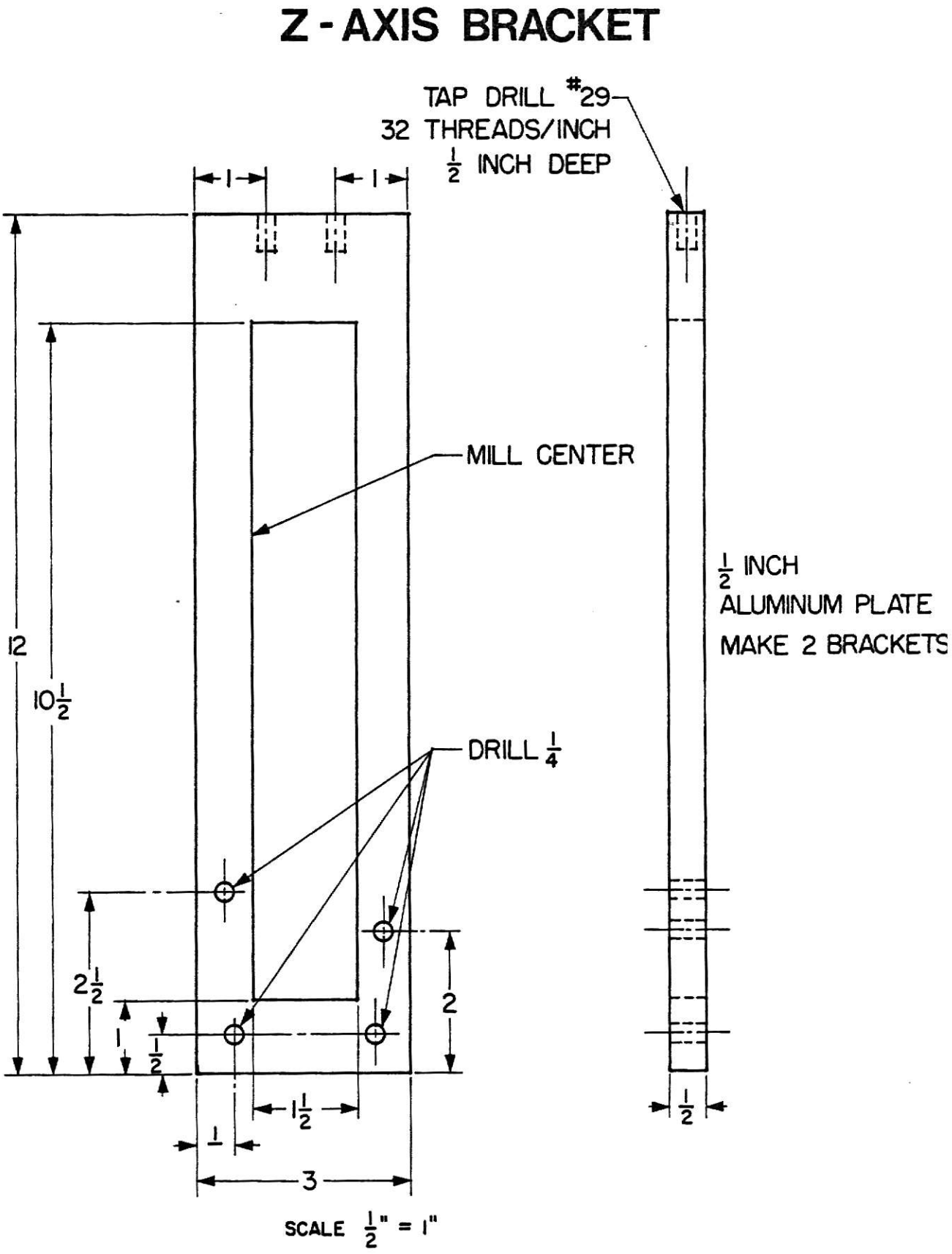


Fig. 11. Z-Axis Bracket

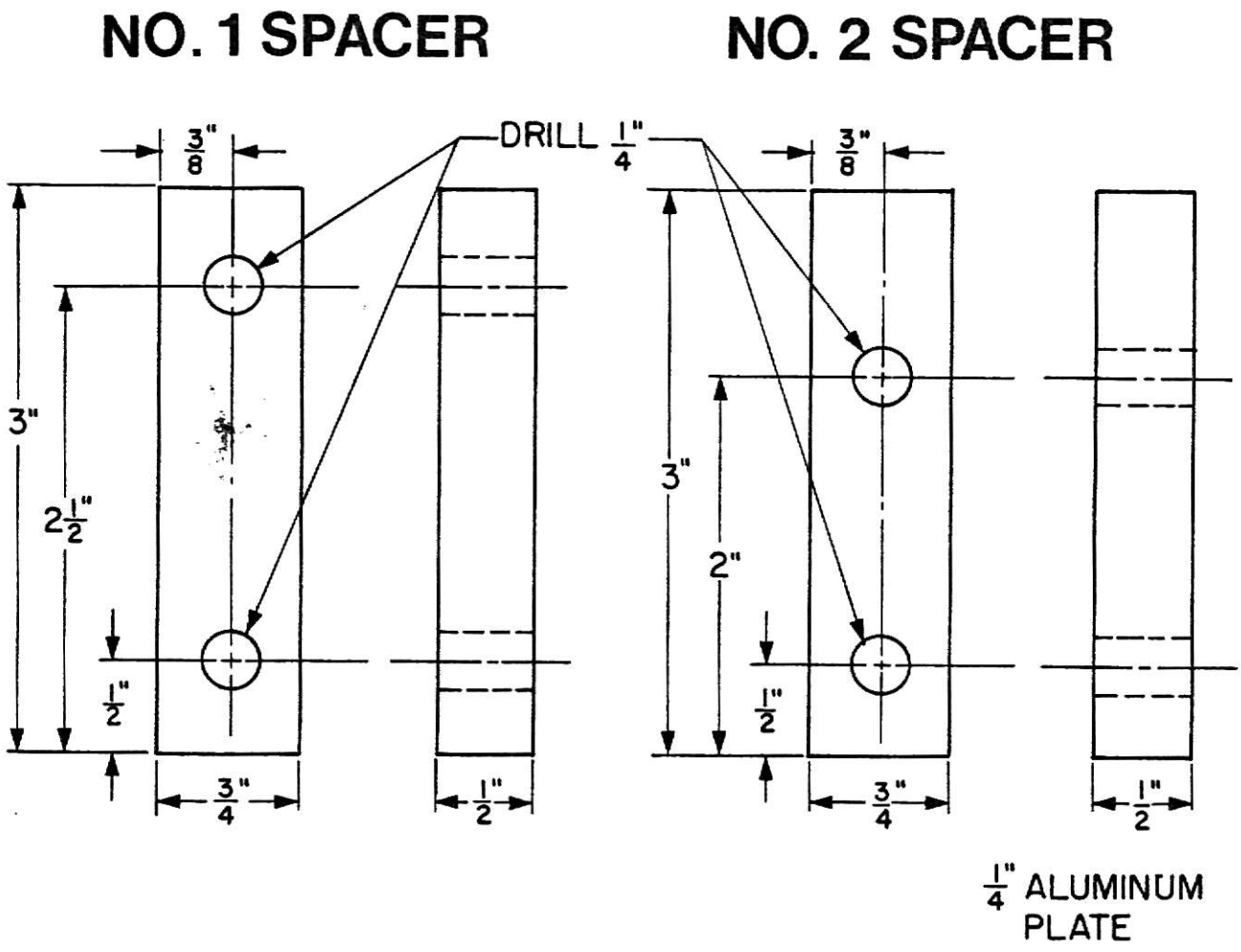
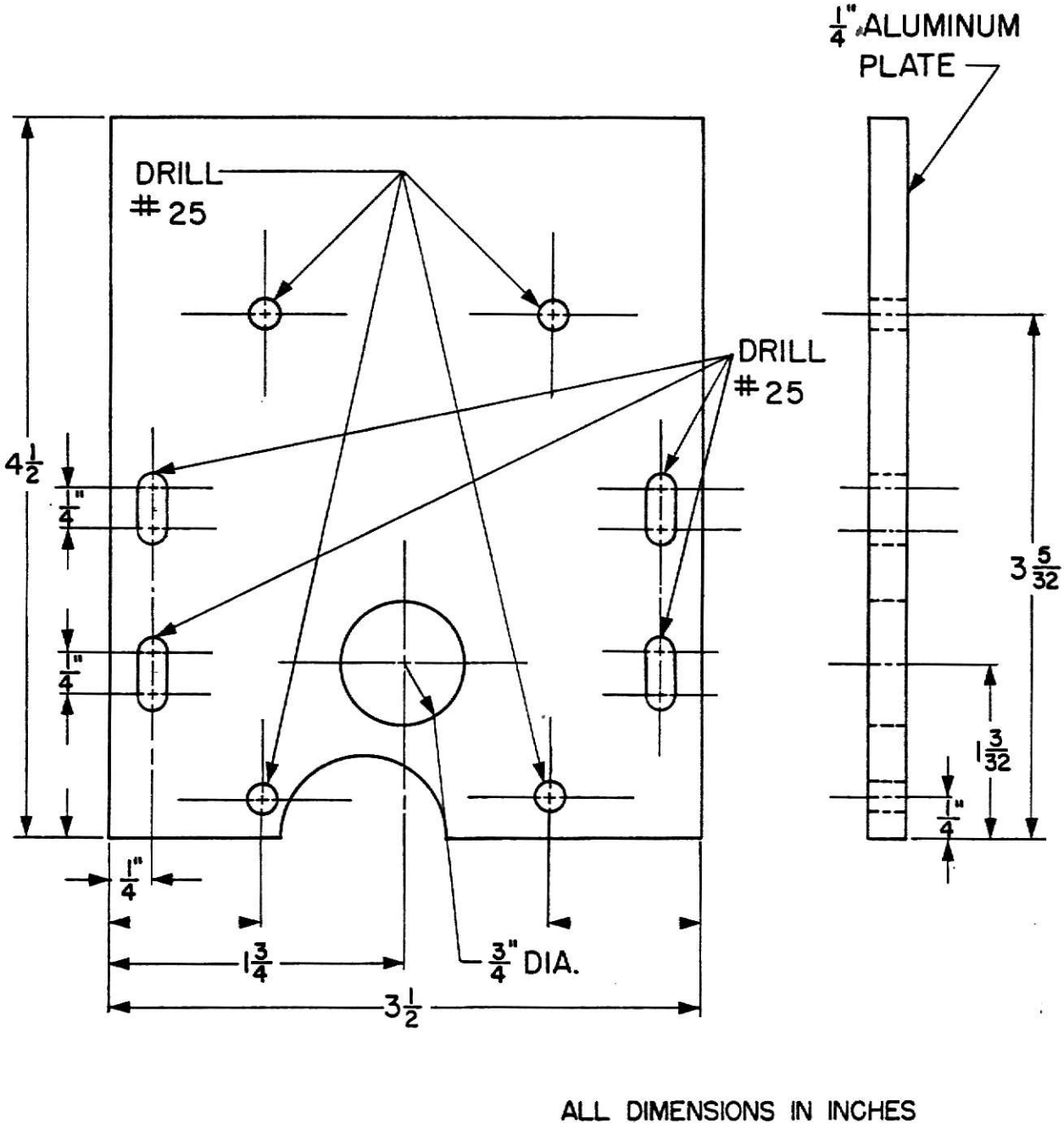


Fig. 12. No. 1 Spacer, No. 2 Spacer

Z-AXIS MOTOR MOUNT



ALL DIMENSIONS IN INCHES

Fig. 13. Z-Axis Motor Mount

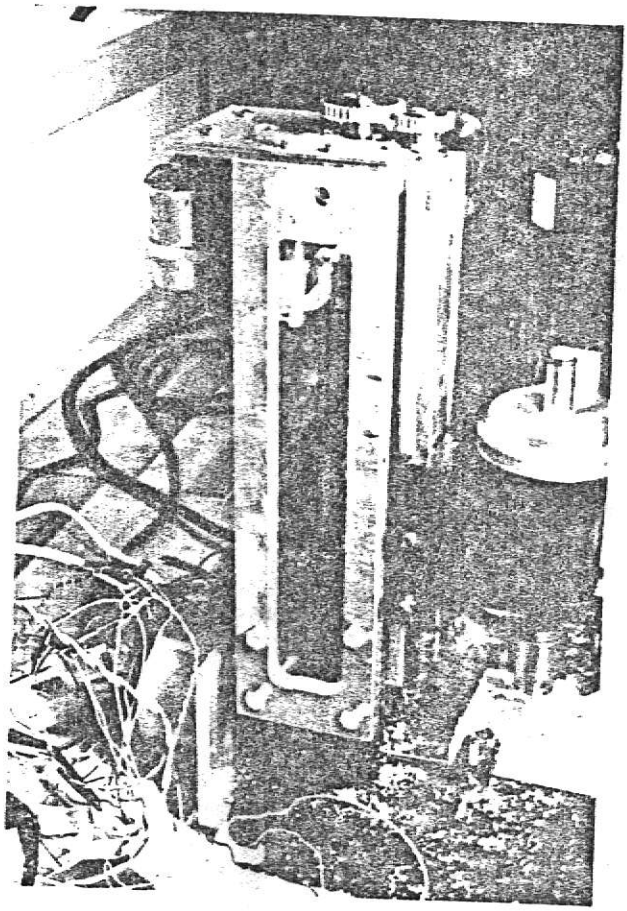


Fig. 14. Z-Axis Mount Assembly Picture

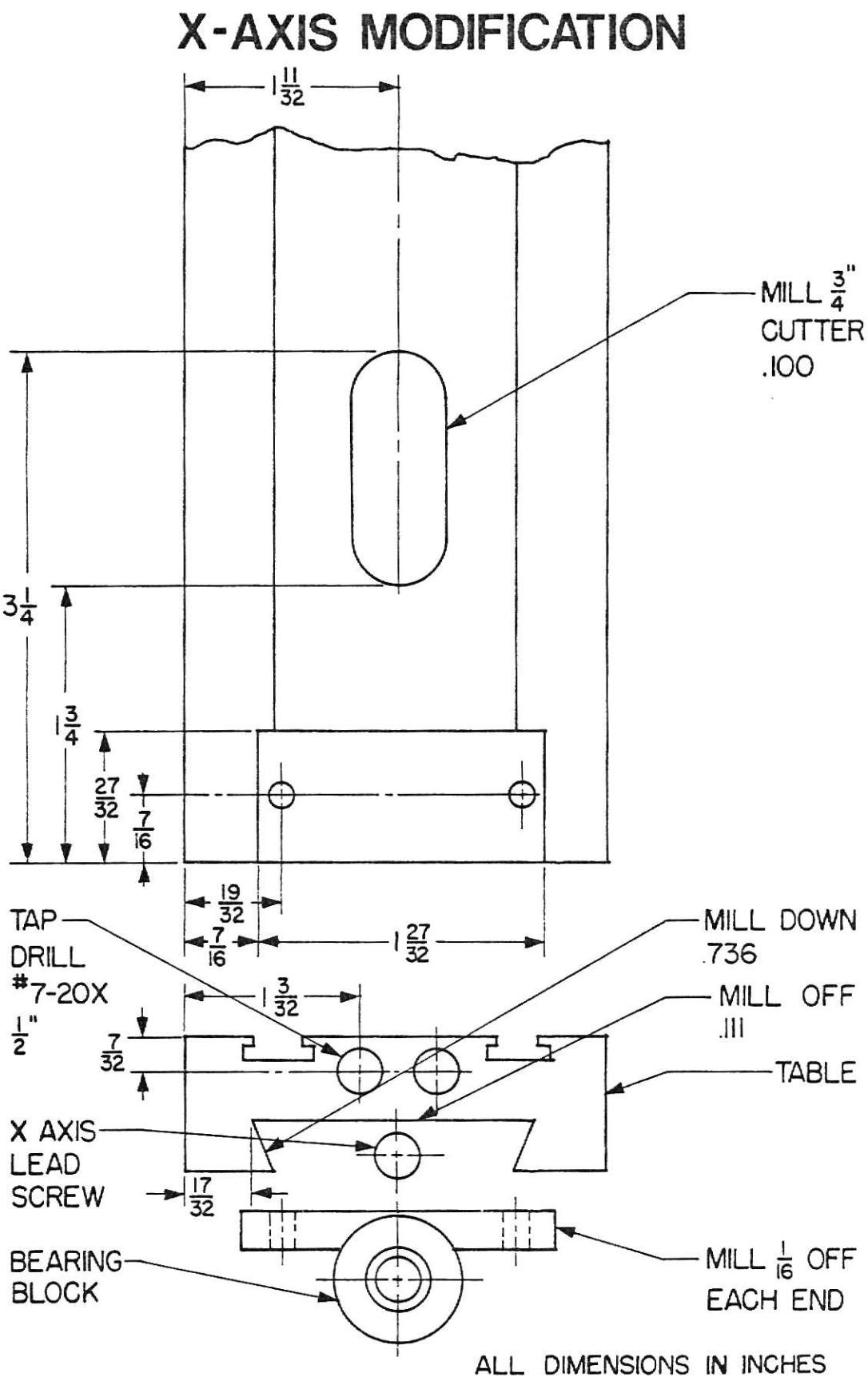


Fig. 15. X-Axis Modification

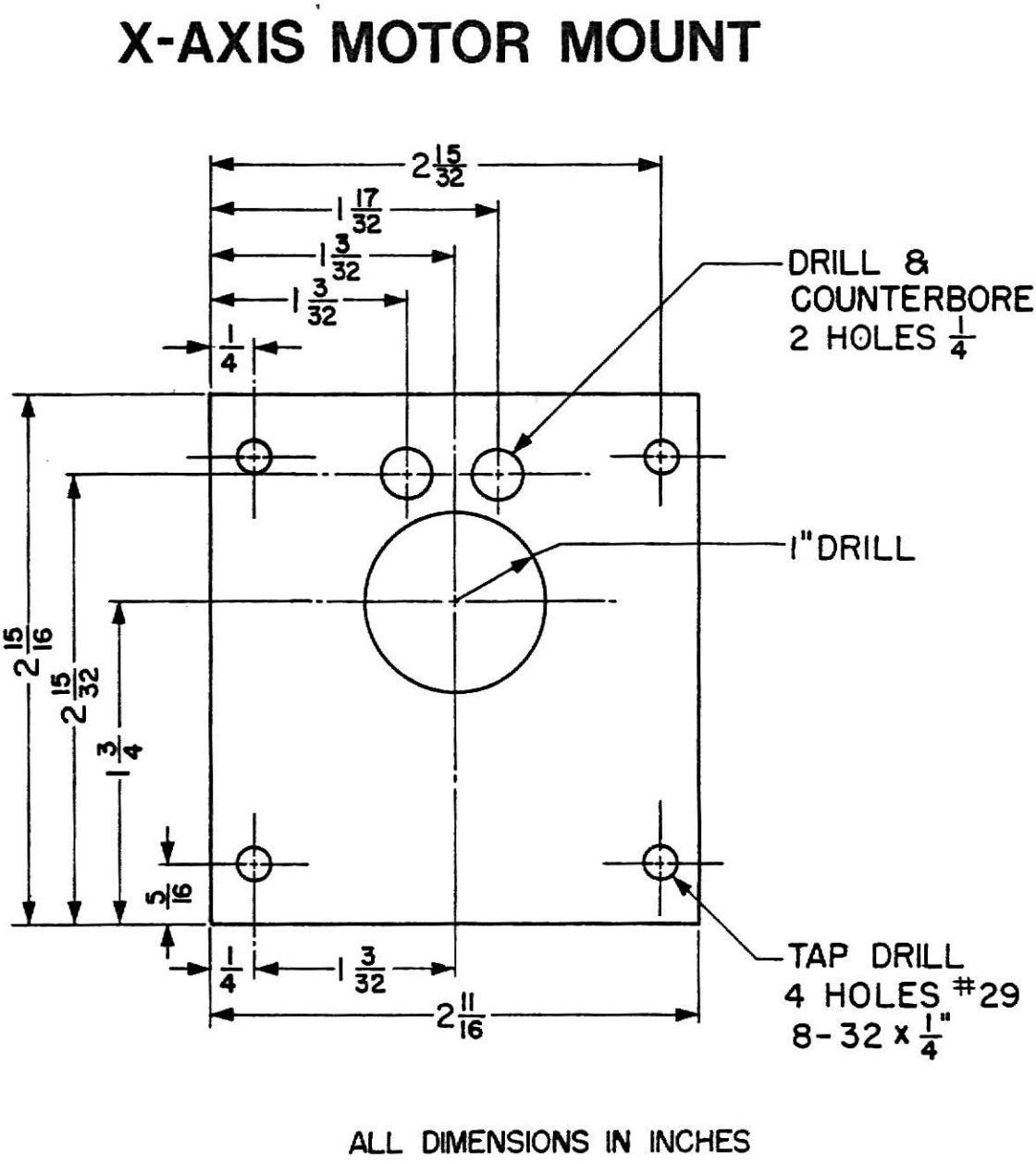


Fig. 16. X-Axis Motor Mount

Fig. 17. X-Axis Mount Assembly Pciture

Y AXIS MODIFICATION

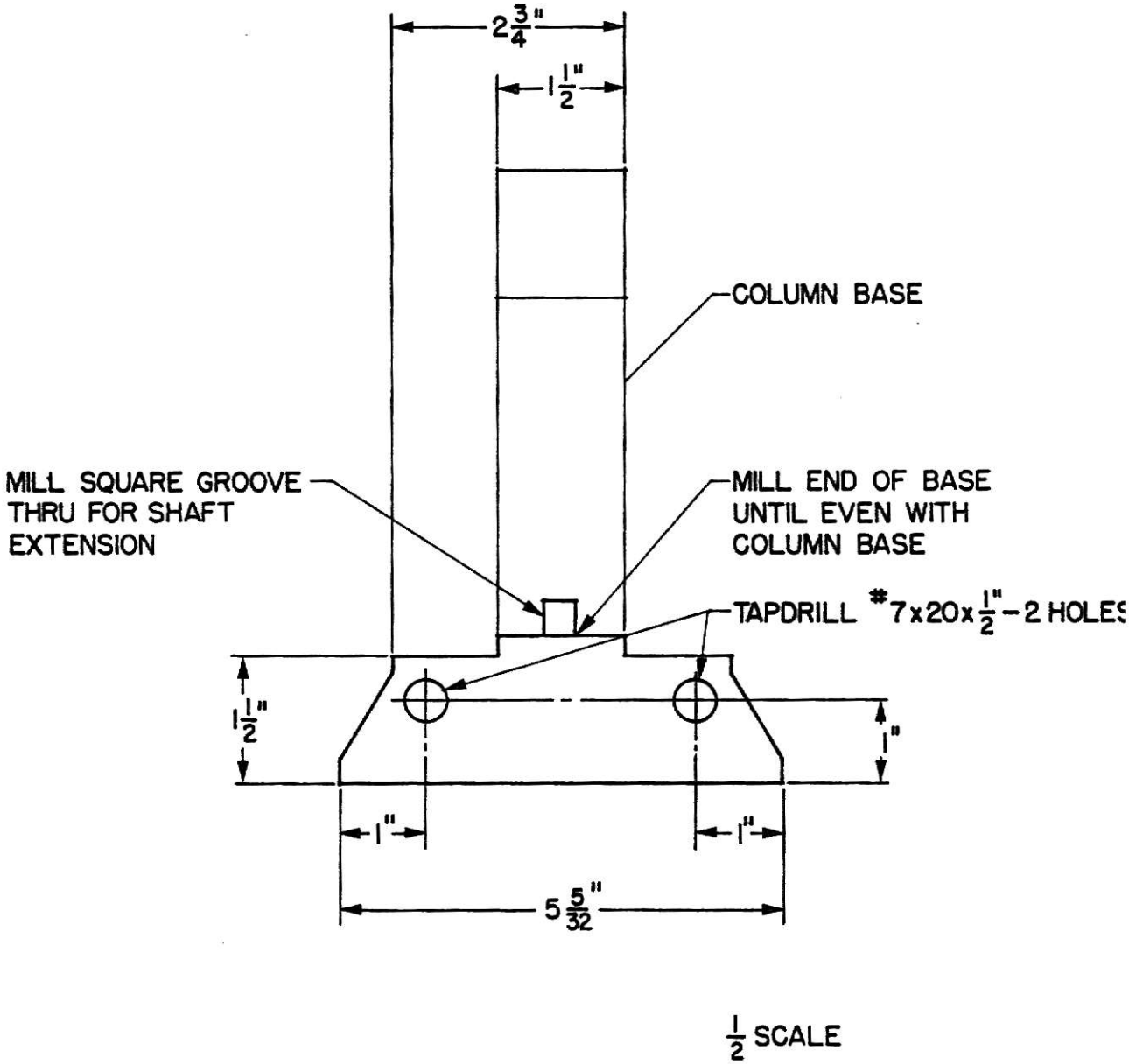
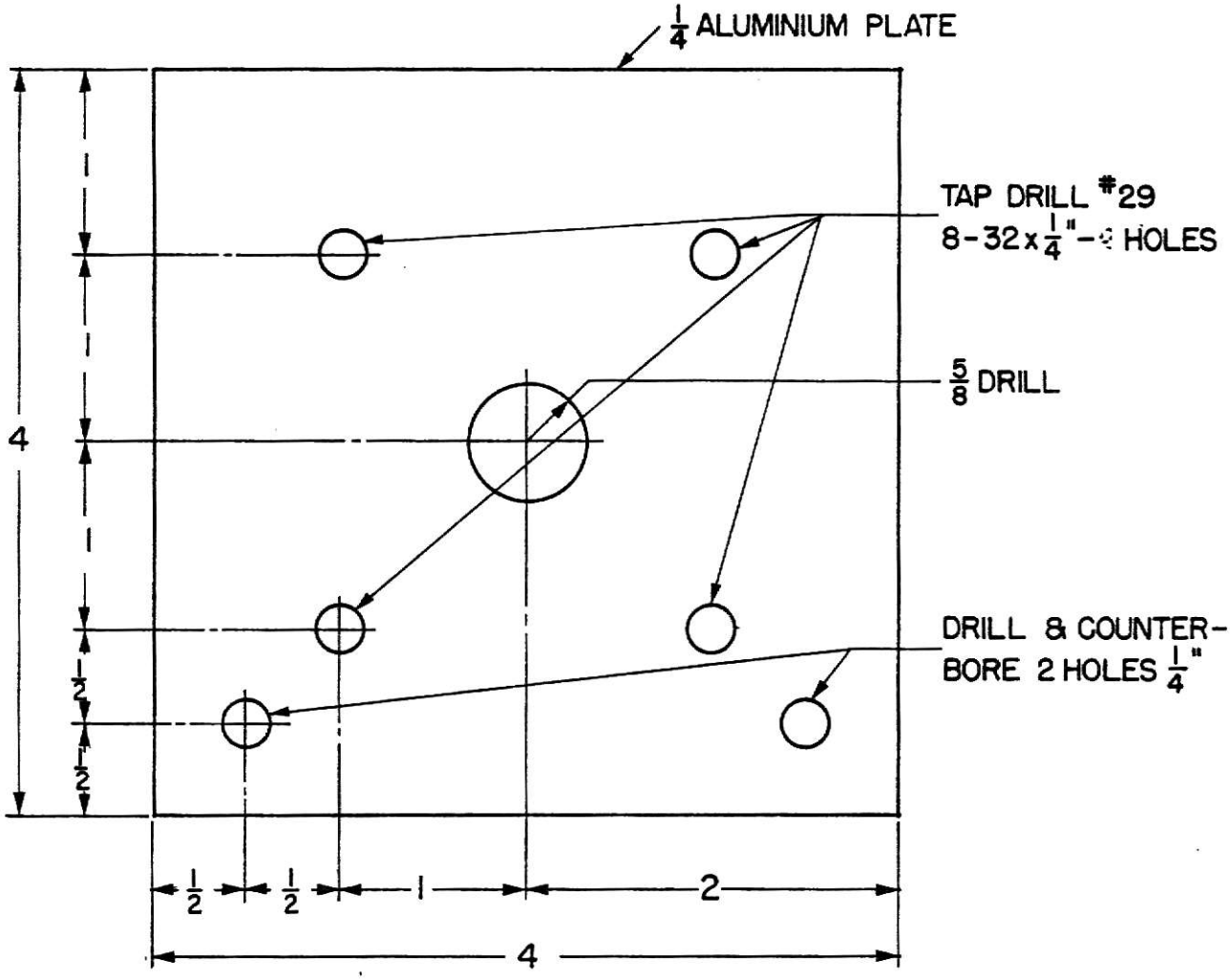


Fig. 18. Y-Axis Modification

Y-AXIS MOTOR MOUNT & SPACER



ALL DIMENSIONS IN INCHES

Y-AXIS SPACER

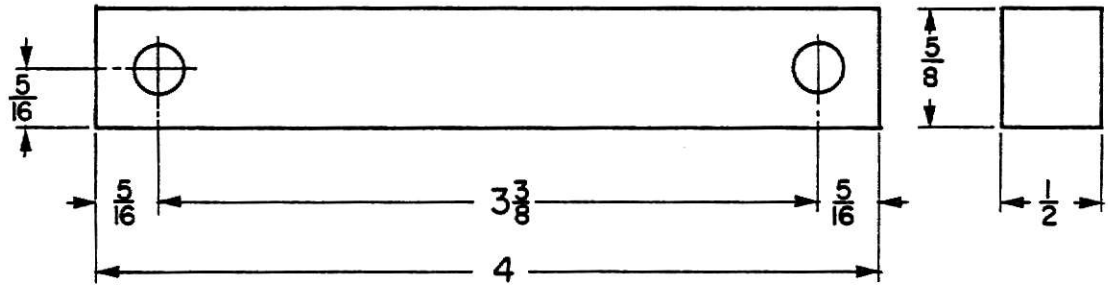


Fig. 19. Y-Axis Motor Mount and Spacer

Fig. 20. Y-Axis Mount Assembly Picture

APPENDIX B

Z AXIS DOWN SUBROUTINE

Appendix B Z Axis Down Subroutine

Location	Code	
031 000	323	Out
1	015	Down
2	315	Call
3	127	10 ms. delay
4	000	
5	323	Out
6	013	Start
7	315	Call
010	300	Variable delay
1	033	
2	323	Out
3	014	Stop
4	315	Call
5	127	10 ms. delay
6	000	
7	166	Halt

APPENDIX C

DELAY SUBROUTINE

APPENDIX C DELAY SUBROUTINE

Location	Octal	Description
033 300	365	Push statusword
1	056	Move immediate byte into Register L
2	xxx	See Table 8
3	006	Move immediate byte into Register B
4	002	Timing value,See Table 10
5	016	Move immediate byte into Register C
6	005	Timing value, see Table 10
7	026	Move immediate byte into Register D
310	003	Timing value, see Table 10
1	036	Move immediate byte into Register E
2	004	Timing value, see Table 10
3	035	Decrement Register E by one
4	302	Jump if not zero
5	311	
6	033	
7	025	Decrement Register D by one
20	302	Jump if not zero
1	307	
2	033	
3	015	Decrement Register C by one
4	302	Jump if not zero
5	305	
6	033	
7	005	Decrement Register B by one
30	302	Jump if not zero
1	303	
2	033	
3	055	
4	302	Jump if not zero
5	303	
6	033	
7	361	Pop statusword
40	311	Return

APPENDIX D

X+ SUBROUTINE

APPENDIX D X+ SUBROUTINE

Location	Octal	Description
030 100	056	MVI L
1	xxx	See Table 9
2	076	MVI A
3	010	
4	323	Out
5	000	Port 0
6	315	Call
7	000	Rotation delay subroutine
110	033	
1	076	MVI A
2	011	
3	323	Out
4	000	Port 0
5	315	Call
6	000	Rotation delay subroutine
7	033	
120	076	MVI A
1	001	
2	323	Out
3	000	Port 0
4	315	Call
5	000	Rotation delay subroutine
6	033	
7	076	MVI A
130	005	
1	323	Out
2	000	
3	315	Call
4	000	Rotation delay subroutine
5	033	
6	076	MVI A
7	004	

150	000	Port 0
1	315	Call
2	000	Rotation delay subroutine
3	033	
4	076	MVI A
5	002	
6	323	Out
7	000	Port 0
030 160	315	Call
1	000	Rotation delay subroutine
2	033	
3	076	MVI A
4	012	
5	323	Out
6	000	Port 0
7	055	DCR L
170	302	JNZ
1	102	
2	030	
3	303	
4	xxx	Location of
5	xxx	Next subroutine

Y-

Location	Octal	Description
031 000	046	Move immediate byte into Register H
1	003	Timing value
2	056	Move immediate byte into Register C
3	350	
4	076	Move immediate byte into accumulator
5	010	
6	323	Out
7	002	Port 2
010	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	012	
5	323	Out
6	002	Port 2
7	315	Call
020	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	002	
4	323	Out
5	002	
6	315	Call
7	000	Delay
030	035	
1	076	Move immediate byte into accumulator
2	006	
3	323	Out
4	002	Port 2
5	315	Call
6	000	Delay
7	035	

Location	Octal	Description
040	076	Move immediate byte into accumulator
1	004	
2	323	Out
3	002	Port 2
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator
050	005	
1	323	Out
2	002	Port 2
031 053	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	001	
060	323	Out
1	002	Port 2
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	011	
7	323	Out
070	002	Port 2
1	315	Call
2	000	Delay
3	035	
4	055	Decrement Register L by one
5	302	Jump if not zero
6	004	
7	031	
100	045	Decrement Register H by one
1	320	Jump if not zero
2	002	
3	031	
4	303	Unconditional jump
5	200	

X+

Location	Octal	Description
031 200	046	Move immediate byte into Register H
1	003	Timing value
2	056	Move immediate byte into Register L
3	350	
4	076	Move immediate byte into accumulator
5	010	
6	323	Out
7	000	Post 0
210	315	Call
1	000	Delay
2	035	
031 213	076	Move immediate byte into accumulator
4	011	
5	323	Out
6	000	Port 0
7	315	Call
220	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	001	
4	323	Out
5	000	Port 0
6	315	Call
7	000	Delay
230	035	
1	076	Move immediate byte into accumulator
2	005	
3	323	Out
4	000	Port 0
5	315	Call
6	000	Delay
7	035	
240	076	Move immediate byte into accumulator
1	004	
2	323	Out
3	000	Port 0

Location	Octal	Description
244	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator
250	006	
1	323	Out
2	000	Port 0
3	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	002	
260	323	Out
1	000	Port 0
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	012	
7	323	Out
270	000	Port 0
1	315	Call
2	000	Delay
3	035	
4	055	Decrement Register L by one
031 275	302	Jump if not zero
6	204	
7	031	
300	045	Decrement Register H by one
1	302	Jump if not zero
2	202	
3	031	
4	303	Unconditional jump
5	000	(X-) lcm
6	032	

10mm X-

Location	Octal	Description
032 000	056	Move immediate byte into Register L
1	350	Timing value
2	076	Move immediate byte into accumulator
3	010	
4	323	Out
5	000	Port 0
6	315	Call
7	000	Delay
010	035	
1	076	Move immediate byte into accumulator
2	012	
3	323	Out
4	000	Port 0
5	315	Call
6	000	Delay
7	035	
020	076	Move immediate byte into accumulator
1	002	
2	323	Out
3	000	Port 0
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator
030	006	
1	323	Out
2	000	Port 0
3	315	Call
4	000	Delay
5	035	
302 6	076	Move immediate byte into accumulator
7	004	

Location	Octal	Description
040	323	Out
1	000	Port 0
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	005	
7	323	Out
050	000	Port 0
1	315	Call
2	000	Delay
3	035	
4	076	Move immediate byte into accumulator
5	001	
6	323	Out
7	000	Port 0
060	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	011	
5	323	Out
6	000	Port 0
7	315	Call
070	000	Delay
1	035	
2	055	Decrement Register L by one
3	302	Jump if not zero
4	002	
5	032	
6	303	Unconditional jump
7	000	(X+) (Y+)
100	033	

Location	Octal	Description
033 000	056	Move immediate byte into Register L
1	350	Timing value
2	315	Call
3	000	(X+) (Y+)
4	034	
5	056	Move immediate byte into Register L
6	170	Timing value
7	315	Call
010	130	(X-) (Y+)
1	034	
2	056	Move immediate byte into Register L
3	170	Timing value
4	315	Call
5	000	(X+) (Y+)
6	034	
7	056	Move immediate byte into Register L
020	350	Timing value
1	315	Call
2	130	(X+) (Y+)
3	034	
4	056	Move immediate byte into Register L
5	170	Timing value
6	315	Call
7	100	(X-) (Y-)
030	035	
1	056	Move immediate byte into Register L
2	170	Timing value
3	315	Call
4	130	(X-) (Y+)
5	034	
6	056	Move immediate byte into Register L
7	350	Timing value

Location	Octal	Description
040	315	Call
1	100	(X-) (Y-)
2	035	
3	056	Move immediate byte into Register L
4	170	Timing value
5	315	Call
6	225	(X+) (Y-)
7	035	
050	056	Move immediate byte into Register L
1	170	Timing value
2	315	Call
3	100	(X-) (Y-)
4	035	
5	056	Move immediate byte into Register L
6	350	Timing value
7	315	Call
033 060	225	(X+) (Y-)
1	035	
2	056	Move immediate byte into Register L
3	170	Timing value
4	315	Call
5	000	(X+) (Y+)
6	034	
7	056	Move immediate byte into Register L
070	170	Timing value
1	315	Call
2	225	(X+) (Y-)
3	035	
4	303	Unconditional jump
5	100	
6	036	

APPENDIX E

PROGRAM TO MILL 10mm LINE WITH SLOPE OF 45°

APPENDIX E 10mm LINE 45° SLOPE

(X-) (Y-)

Location	Octal	Description
035 100	076	Move immediate byte into accumulator
1	010	
2	323	Out
3	000	Port 0
4	323	Out
5	002	Port 2
6	315	Call
7	000	Delay
110	035	
1	076	Move immediate byte into accumulator
2	012	
3	323	Out
4	000	Port 0
5	323	Out
6	002	Port 2
7	315	Call
120	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	002	
4	323	Out
5	000	Port 0
6	323	Out
7	002	Port 2
130	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	006	
5	323	Out
6	000	Port 0
7	323	Out

APPENDIX E (cont)

035 140	002	Port 2
1	315	Call
2	000	Delay
3	035	
4	076	Move immediate byte into accumulator
5	004	
6	323	Out
7	000	Port 0
150	323	Out
1	002	Port 2
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	005	
7	323	Out
035 160	000	Port 0
1	323	Out
2	002	Port 2
3	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	001	
170	323	Out
1	000	Port 0
2	323	Out
3	002	Port 2
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator

APPENDIX E (cont)

	200	011	
	1	323	Out
	2	000	Port 0
	3	323	Out
	4	002	Port 2
	5	315	Call
	6	000	Delay
	7	035	
	210	055	Decrement Register L by one
	1	302	Jump if not zero
	2	100	
	3	035	
035	214	311	Return

APPENDIX F

CROSS IN THE SQUARE PROGRAM

(X+) (Y+)

Location	Octal	Description
034 000	076	Move immediate byte into accumulator
1	010	
2	323	Out
3	000	Port 0
4	323	Out
5	002	Port 2
6	315	Call
7	000	Delay
010	035	
1	076	Move immediate byte into accumulator
2	011	
3	323	Out
4	000	Port 0
5	323	Out
6	002	Port 2
7	315	Call
020	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	001	
4	323	Out
5	000	Port 0
6	323	Out
034 027	002	Port 2
030	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	005	
5	323	Out
6	000	
7	323	Out

Location	Octal	Description
034 100	012	
1	323	Out
2	000	Port 0
3	323	Out
4	002	Port 2
5	315	Call
6	000	Delay
7	035	
110	055	Decrement Register L by one
1	302	Jump if not zero
2	000	
3	034	
4	311	Return

(X-) (Y+)

Location	Octal	Description
034 130	076	Move immediate byte into accumulator
1	010	
2	323	Out
3	000	Port 0
4	323	Out
5	002	Port 2
6	315	Call
7	000	Delay
140	035	
1	076	Move immediate byte into accumulator
2	012	
3	323	Out
4	000	Port 0
5	076	Move immediate byte into accumulator
6	011	
7	323	Out

Location	Octal	Description
150	002	Port 2
1	315	Call
2	000	Delay
3	035	
4	076	Move immediate byte into accumulator
5	002	
6	323	Out
7	000	Port 0
160	076	Move immediate byte into accumulator
1	001	
2	323	Out
3	002	Port 2
4	315	Call
034 165	000	Delay
6	035	
7	076	Move immediate byte into accumulator
170	006	
1	323	Out
2	000	Port 0
3	076	Move immediate byte into accumulator
4	005	
5	323	Out
6	002	Port 2
7	315	Call
200	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	004	
4	323	Out
5	000	Port 0
6	323	Out
7	002	Port 2

Location	Octal	Description
034 250	002	Port 2
1	315	Call
2	000	Delay
3	035	
4	055	Decrement Register L by one
5	302	Jump if not zero
6	130	
7	034	
260	311	Return

Stepper Motor Delay

Location	Octal	Description
035 000	365	Push statusword
1	006	Move immediate byte into Register B
2	002	Timing value
3	016	Move immediate byte into Register C
4	005	Timing value
5	026	Move immediate byte into Register D
6	003	Timing value
7	036	Move immediate byte into Register E
010	004	Timing value
1	035	Decrement Register E by one
2	302	Jump if not zero
3	011	
4	035	
5	025	Decrement Register D by one
6	302	Jump if not zero
7	007	
020	035	
1	015	Decrement Register C by one
2	302	Jump if not zero
3	005	
4	035	
5	005	Decrement Register B by one

Location	Octal	Description
210	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	005	
5	323	Out
6	000	Port 0
7	076	Move immediate byte into accumulator
220	006	
1	323	Out
2	002	Port 2
3	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	001	
230	323	Out
1	000	Port 0
2	076	Move immediate byte into accumulator
3	002	
4	323	Out
5	002	Port 2
6	315	Call
7	000	Delay
034 240	035	
1	076	Move immediate byte into accumulator
2	011	
3	323	Out
4	000	Port 0
5	076	Move immediate byte into accumulator
6	012	
7	323	Out

Location	Octal	Description
040	002	Port 2
1	315	Call
2	000	Delay
3	035	
4	076	Move immediate byte into accumulator
5	004	
6	323	Out
7	000	Port 0
050	323	Out
1	002	Port 2
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	006	
7	323	Out
060	000	Port 0
1	323	Out
2	022	Port 2
3	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	002	
070	323	Out
1	000	Port 0
2	323	Out
3	002	Port 2
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator

Location	Octal	Description
035 270	076	Move immediate byte into accumulator
1	005	
2	323	Out
3	000	Port 0
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator
300	004	
1	323	Out
2	002	Port 2
3	323	Out
4	000	Port 0
5	315	Call
6	000	Delay
7	035	
310	076	Move immediate byte into accumulator
1	005	
2	323	Out
3	002	Port 2
4	076	Move immediate byte into accumulator
5	006	
6	323	Out
7	000	Port 0
320	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	001	
5	323	Out
6	002	Port 2
7	076	Move immediate byte into accumulator

Location	Octal	Description
035 330	002	
1	323	Out
2	000	Port 0
3	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	011	
340	323	Out
1	002	Port 2
2	076	Move immediate byte into accumulator
035 026	302	Jump if not zero
7	003	
030	035	
1	361	Pop statusword
2	311	Return

(X-) (Y-)

Location	Octal	Description
035 100	076	Move immediate byte into accumulator
1	010	
2	323	Out
3	000	Port 0
4	323	Out
5	002	Port 2
6	315	Call
7	000	Delay
110	035	
1	076	Move immediate byte into accumulator
2	012	
3	323	Out
4	000	Port 0
5	323	Out
6	002	Port 2
7	315	Call

Location	Octal	Description
035 120	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	002	
4	323	Out
5	000	Port 0
6	323	Out
7	002	Port 2
130	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	006	
5	323	Out
6	000	Port 0
7	323	Out
140	002	Port 2
1	315	Call
2	000	Delay
3	035	
4	076	Move immediate byte into accumulator
5	004	
6	323	Out
7	000	Port 0
150	323	Out
1	002	Port 2
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	005	
7	323	Out

Location	Octal	Description
035 160	000	Port 0
1	323	Out
2	002	Port 2
3	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	001	
170	323	Out
1	000	Port 0
2	323	Out
3	002	Port 2
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator
200	011	
1	323	Out
2	000	Port 0
3	323	Out
4	002	Port 2
5	315	Call
6	000	Delay
7	035	
210	055	Decrement Register L by one
1	302	Jump is not zero
2	100	
3	035	
4	311	Return

(X+) (Y-)

Location	Octal	Description
035 225	076	Move immediate byte into accumulator
6	010	
7	323	Out
230	002	Port 2
1	323	Out
2	000	Port 0
3	315	Call
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	012	
240	323	Out
1	002	Port 2
2	076	Move immediate byte into accumulator
3	011	
4	323	Out
5	000	Port 0
6	315	Call
7	000	Delay
250	035	
1	076	Move immediate byte into accumulator
2	002	
3	323	Out
4	002	Port 2
5	076	Move immediate byte into accumulator
6	001	
7	323	Out
260	000	Port 0
1	315	Call
2	000	Delay
3	035	
4	076	Move immediate byte into accumulator
5	006	
6	323	Out
7	002	Port 2

Location	Octal	Description
035 343	012	
4	323	Out
5	000	Port 0
6	315	Call
7	000	Delay
350	035	
1	055	Decrement Register L by one
2	302	Jump if not zero
3	225	
4	035	
5	311	Return

Z UP

Location	Octal	Description
036 100	056	Move immediate byte into Register L
1	350	Timing Value
2	323	Out
3	016	Up
4	315	Call
5	127	10 m sec delay
6	000	
7	323	Out
110	013	Start
1	315	Call
2	000	Delay
3	035	
4	055	Decrement Register L by one
5	302	Jump if not zero
6	102	
7	036	
120	323	Out
1	014	Stop
2	166	Halt

Z Down

Location	Octal	Description
036 000	056	Move immediate byte into Register L
1	350	Timing value
2	323	Out
3	015	Down
4	315	Call
5	127	10mx delay
6	000	
7	323	Out
010	013	Start
1	315	Call
2	000	Delay
3	035	
4	055	Decrement Register L by 1
5	302	Jump if not zero
6	002	
7	036	
020	323	Out
1	014	Stop
2	303	Unconditional jump
3	000	
4	030	Y+

Y+

Location	Octal	Description
030 000	046	Move immediate byte into Register H
1	003	3cm
2	056	Move immediate byte into Register L
3	350	Timing value
4	076	Move immediate byte into accumulator
5	010	
6	323	Out
7	002	Port 2
010	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator

Y+ (cont)

Location	Octal	Description
4	011	
5	323	Out
6	002	Port 2
7	315	Call
020	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	001	
4	323	Out
5	002	Port 2
6	315	Call
7	000	Delay
030	035	
1	076	Move immediate byte into accumulator
2	005	
3	323	Out
4	002	Port 2
5	315	Call
6	000	Delay
7	035	
040	076	Move immediate byte into accumulator
1	004	
2	323	Out
3	002	Port 2
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator
050	006	
1	323	Out
2	002	Port 2
3	315	Call
4	000	Delay

Location	Octal	Description
5	035	
6	076	Move immediate byte into accumulator
7	002	
060	323	Out
1	002	Port 2
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	012	
7	323	Out
070	002	Port 2
1	315	Call
2	000	Delay
3	035	
4	055	Decrement Register L by one
5	302	Jump if not zero
6	004	
7	030	
100	045	Decrement Register H by one
1	302	Jump if not zero
2	002	
3	030	
4	303	Unconditional jump
5	200	
6	030	

X-

Location	Octal	Description
030 200	046	Move immediate byte into Register H
1	003	Timing value
2	056	Move immediate byte into Register L
3	350	
4	075	Move immediate byte into accumulator
5	010	
6	323	Out
7	000	Port 0

Location	Octal	Description
210	315	Call
1	000	Delay
2	035	
3	076	Move immediate byte into accumulator
4	012	
5	323	Out
6	000	Port 0
7	315	Call
220	000	Delay
1	035	
2	076	Move immediate byte into accumulator
3	002	
4	323	Out
5	000	Port 0
6	315	Call
7	000	Delay
030 230	035	
1	076	Move immediate byte into accumulator
2	006	
3	323	Out
4	000	Port 0
5	315	Call
6	000	Delay
7	035	
240	076	Move immediate byte into accumulator
1	004	
2	323	Out
3	000	Port 0
4	315	Call
5	000	Delay
6	035	
7	076	Move immediate byte into accumulator
250	005	
1	323	Out
2	000	Port 0
3	315	Call

Location	Octal	Description
4	000	Delay
5	035	
6	076	Move immediate byte into accumulator
7	001	
260	323	Out
1	000	Port 0
2	315	Call
3	000	Delay
4	035	
5	076	Move immediate byte into accumulator
6	011	
7	323	Out
270	000	
1	315	Call
2	000	Delay
3	035	
4	055	Decrement Register L by one
5	302	Jump if not zero
6	204	
7	030	
300	045	Decrement Register H by one
1	302	Jump if not zero
2	202	
3	030	
4	303	Unconditional jump
5	000	
6	031	

CONVERTING A MILLING MACHINE TO A TWO AXIS CONTOURING MACHINE

by

BRUCE P. MIGNANO

B.S., United States Military Academy, 1958
M.S., Stevens Institute of Technology, 1964

AN ABSTRACT OF A MASTER'S THESIS
submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering
KANSAS STATE UNIVERSITY
Manhattan, Kansas
1980

The purpose of this research is to convert a milling machine to a microcomputer controlled two-axis contouring machine. The conversion process is divided into two phases, design and construction. The first phase presents the significant design considerations in two areas: power requirements and circuitry. Lead screw drive power requirements are developed and interfacing circuitry between the microcomputer and the milling machine is designed. The second phase takes the phase one design and constructs and assembles the necessary hardware. Finally, a machine language program is prepared that cause a cross to be inscribed in a square. The system and the program are tested by milling a cross in a square on a piece of acrylic plastic while the milling machine is under the control of the microcomputer.