Compumotor

Model 3000 Controller User Guide

Compumotor Division
Parker Hannifin
88-003181-05

TECHNICAL BULLETIN PARKER COMPUMOTOR

No. 100 **August 15, 1986**

PRODUCT: **3000 Indexers**
DESCRIPTION: **Enhancements in 3000 Indexer C–revision software**
SOFTWARE AFFECTED: **3000 controller, p/n 92–5003–01/11C; 1811 axis cards, p/n 92–3657–01/11I**
NEXT SCHEDULED RELEASE: **December 1, 1986**

The following is a summary of the additions and changes to Rev. C of the 3000 software. All items listed here have been incorporated into the revised manual and may be found in the index and table of contents (p/n 88–003181–04). To verify the revision of software in your 3000:

1. turn off the 3000
2. depress the START key
3. turn on the 3000
4. after 1 second, release the START key
5. read the software revision in the display

MAIN MENU –

RESUME function. After the STOP key is pressed, and motor motion has stopped, the RESUME softkey may be pressed to complete any moves that were stopped and resume program execution where it was before being stopped.

EDIT MENU –

MODE command. Allows changing the axis mode at any point in the program. The choices are NULL M_INC M_ABS E_INC E_ABS.

```
100MODE M_INC E_ABS NULL E_INC
```

BOOST command. Allows boost output to activate or deactivate. The choices are NULL, YES, NO, x (x is a 16 bit number). If a number is used, the auto boost will be enabled. The number will be the time in milliseconds of post accel/decel boost.

```
100BOOST NO NULL 25 YES
```

SHTDN command. Allow shutdown output to activate or deactivate. The choices are NULL, YES, NO.

```
100SHTDN NO NULL NO YES
```

ON trigger. When a specified pattern appears on the input lines a subroutine call is executed. Any moves in progress will not be terminated by the ON trigger GOTO. Only the last ONTRIG command encountered will be active, and it will deactivate when its GOTO is executed. The sample rate is every 100 ms, and between commands.

```
100ON xx01 xxxx xxxx xxxx GOTO 5000
110MOVE 50000 NULL NULL NULL
120GOTO 110
5000MOVE STOP NULL NULL NULL
5010DONE
```

STOP parameter. Added to the MOVE and MOVI command. This will stop the move in progress, at the deceleration rate defined prior to that move.

DATA statement as in BASIC. The 3000 will make moves on axis 1 of 50 steps, then 44 steps, then 56 steps, and then it will execute statement 300. Statement 300 will reset the data pointer, and Statement 310 will repeat the process again.

```
100DATA 50 44 56 99
200INPUT Q1 = SRC3
205IF Q1 = 99 THEN 300
210MOVE Q1
240GOTO 200
300INPUT Q1 = SRC4
310GOTO 200
```

Note that if all the data numbers are used up and another is requested prior to a SRC4 input, the program will abort.

Note this includes adding SRC3 and SRC4 to the INPUT command.

INPUT command. Input into variable Qx from other sources. The format is 0100 INPUT Q3 = SRCx where x is a number as defined below.

SRC0 parameter. Front panel input (This is the default if no SRC is specified). For example 100INPUT Q5 will work the same as earlier versions because it will default to SRC0.

```
100INPUT Q7 or 100INPUT Q7 = SRC0
```

SRC1 parameter. Currently active remote port input (RS 232 or IEEE-488). For example, 0100INPUT Q6 = SRC1 will print to the port "INPUT Q6?" and then wait until a number is input. The number must be terminated with a semicolon. — 25,000;

```
100INPUT Q7 = SRC1
```

SRC2 parameter. Trigger input. Inputs 1 through 16 will be in bits 0 through 15.

```
100INPUT Q7 = SRC2
```

SRC3 parameter. DATA input. The next data number.

SRC4 parameter. Restores data pointer to first data statement. Q variable used will not be modified.

SRC5x parameter. Axis status input, where x is the axis number. This is a direct result of the 1811 request status command ("88").

```
100INPUT Q7 = SRC51
```

SRC6x parameter. Axis position input, where x is the axis number.

```
100INPUT Q7 = SRC61
```

SRC7x parameter. Axis velocity input, where x is the axis number.

```
100INPUT Q7 = SRC71
```

MATH command. Allow 31 bit integers in the MATH statement.

```
100MATH Q4 = Q3 / Q1
100MATH Q4 = Q3 * Q1
```

In both cases Q1 can be a 31 bit number.

```
100MATH Q4 = Q3 AND Q1
100MATH Q4 = Q3 OR  Q1
```

```
100MATH Q4 = Q3 XOR Q1
```

The boolean operators AND, OR, XOR are supported.

```
100MATH Q4 = COM
```

The contents of Q4 are complemented.

```
100MATH Q4 = BIN
```

The contents of Q4 are converted to a binary number. It will be assumed that the number in Q4 prior to conversion to binary, was a 4 digit BCD number.

```
100MATH Q4 = BCD
```

The contents of Q4 are converted to a 4 digit BCD number. It will be assumed that the number in Q4 prior to conversion to BCD, was a binary number.

WRITE command. Write to output the Q variable to the currently active remote port (RS-232 or IEEE-488). If the starting column number is higher than 40 and less than 100, the contents of the variable will be sent to the port. Example: 0100WRITE 0 0 50 Q25 Note that the other 2 parameters are still required, even though they are ignored. Note that string variables may be used: 0100WRITE 0 0 50 Q105

GOTO variable. The line number to GOTO can be a variable.

```
100GOTO Q5
```

OUTPUT Q variable. Output to opto-22 port:

```
0100OUTPUT Q5 Q6
```

Q5's first 15 bits will be output to outputs 2 thru 16. Q6's first 15 bits will be the mask. If bit is 1 output will change, if bit is 0 output will not change. If Q6 not specified, defaults to no mask.

SETUP MENU -

SETUP LIMITS function.
a. Software limits for each axis. Upper limit, lower limit, and deceleration rate must be specified for each axis. If the upper and lower limits are equal, no software limit exists. Note that this requires 1811 software rev I or later. If not, setup will be ignored.
b. Abort program on limits YES/NO. If any axis encounters a limit then abort the 3000 program and stop all motion. The sample rate will be every 100 milliseconds.

SETUP GO HOME function.

a. Active level.
b. Active edge.
c. Final approach velocity as percentage of initial search velocity.
d. Final approach direction.

Modified SETUP functions.

SETUP AXES function. Fix conflict between OUTSTR and stall detect output. A new setup menu to allow enabling or disabling the output on stall. In the "SETUP AXES", this setup submenu looks like "YES NO YES NO STALL-OUTPUT?". The remote version must look like "YES NO YES NO OUTPUT;". The default is YES on all axes.

Note that the OUTSTR command will still conflict with STALL if YES is used. Note that all remote interface versions of the SETUP AXES command are of a slightly different format:

```
M_INC M_ABS E_INC E_ABS PO; (positioning)
NO    NO    YES   NO    ABORT;  (stall=abort?)
NO    NO    NO    YES   STOP;   (stall-stop?)
NO    NO    YES   NO    OUTPUT; (stall=output?)
NO    NO    NO    YES   PAUSE;  (not db=pause?)
```

To be compatible with some old software, the STOP parameters can be MOVE or STOP instead of NO or YES.

SETUP position maintenance function. Also added is a setup for the position maintenance gain constant and algorithm.

SETUP UNITS function. Extends the UNITS VELOCITY RANGE to 1 MHZ. Note that this range is called "FAST" in the remote setup language.

REMOTE command language. Some new Remote commands have been added to the 3000, and the manual has been greatly expanded with examples for each Remote command.

Q Variables. A method is provided for reading and writing Q variables asynchronously to the user's program.

Writing: `VARIABLE Q26 = 25000;`

This is unchanged.

Reading: `VARIABLE Q26 = ?;`

This will return the number stored in Q26. The number will consist of 11 characters. Leading zeros will be sent as spaces. A semicolon will end the string.

If Q26 contains the number 25000 and text is off `Q26 25000;_crc_lfc`" will be returned.

If Q26 contains the number 25000 and text is on `Q26 25000;_crc_lfcINPUT?`" will be returned.

String Variables. Maximum of ten variables, of no more 20 than characters. These may only be loaded from the remote interface. These string variables will be allowed with the INPUT command and with the WRITE command. They will be Q100 thru Q109. Spaces may be included in the string but multiple spaces will be truncated to a single space,

`VARIABLE Q105 = This is a test;`

This will put "This is a test" in variable Q105. Via the remote port only.

The command "`0100INPUT Q25 Q105`" will cause the string followed by a space to be displayed on the lcd. When the operator inputs a valid number the program will continue.

Documented commands.

Some Remote commands that have always existed are now documented in the manual. `VARIABLE, LCD1, LCD2`

Other enhancements.

Xon/Xoff incorporated for RS-232 recieve. A 600 character input buffer is used. The RS-232 transmit retains Xon/Xoff recognition from earlier versions.

Cursor rolls around to other side of LCD when end is reached during cursor right or cursor left movement.

Entry errors. Attempts to enter numbers that are too large or contain decimal points in the editor will be rejected and the operator will be informed that an entry error has occurred.

EEPROM contains a CRC check. If a data error is detected, the operator is given the choice of accepting the possibly flawed data.

Change the DELETE command. Allow the user to delete all lines between the specified line numbers, even though the stated line numbers are nor used, i.e., DELETE 0000 thru 9999 will delete every line in the program, even if the first line number is 10 and the last is 250. Note that backward deletes will not be done. DELETE 570 thru 10 will do nothing. Note that DELETE 0100 thru 0000 will delete the single line 100.

Fix DISPLAY so that you can display three things such as velocity on axis 1, position on axis 2, and line number even if you have only a two axis system. Note that you still CANNOT display two items for the same axis at the same time, such as position and status for axis 1.

Prevent **RECALL** and **SAVE** from working during program execution.

Allow **IF/THEN** to work with integers.

TECHNICAL BULLETIN

PARKER COMPUMOTOR

No. 101 **August 15, 1986**

PRODUCT: 3000 Indexers
DESCRIPTION: Known problems in 3000 Indexer software, revision C.
SOFTWARE AFFECTED: 3000 controller, p/n 92-5003-01/11C;
1811 axis cards, p/n 92-3657-01/11I
NEXT SCHEDULED RELEASE: December 1, 1986

To verify the revision of software in your 3000:

1. turn off the 3000
2. depress the START key
3. turn on the 3000
4. after 1 second, release the START key
5. read the software revision in the display

NOTE
Revision C of 3000 software requires revision I software in the 1811 axis cards to work properly.

The problems are:

1. Software limits are not fully implemented. Although they can be setup to decelerate the motor upon encountering a software limit, the status of the axis does not change, causing "Abort on Limits" to hang the 3000. Also, the Go-Home "bounce-off-limits" feature is intermittent with software limits.

   We recommend that software limits not be used in Revision C of the 3000 software.

2. The position display may be off by a few counts during velocity streaming. The actual motor position is correct however. The reported motor position is off whether the report is shown on the display or taken over a remote interface.

The problems described here do not have any known side effects. By avoiding use of these features you may operate all other functions of the 3000 with no adverse effects.

88-006252-01

TECHNICAL BULLETIN

PARKER COMPUMOTOR

No. 103 **September 2, 1986**

PRODUCT: 3000 Indexers
DESCRIPTION: New release of 3000 Motor Controller software, revision C1.
SOFTWARE AFFECTED: 3000 controller, p/n 92–5003–01/11C1;
NEXT SCHEDULED RELEASE: December 1, 1986

To verify the revision of software in your 3000:

1. turn off the 3000
2. depress the STOP key
3. turn on the 3000
4. after 1 second, release the STOP key
5. read the software revision in the display

Technical Bulletin 101 refers to striking the striking the START key to accomplish the above procedure. It should say the STOP key.

Description:

The problems described below were fixed in revision C1 of the 3000 controller software. The problems existed only in revision C of 3000 software.

1. IEEE–488 mode must be selected, saved and the 3000 powered down and back up before a program may be loaded and executed over the remotr IEEE–488 interface.

   If you are using the IEEE–488 option, programs loaded over the remote interface will not run unless a SAVE has been executed with the indexer set for IEEE–488 remote operation <u>and</u> the 3000 has been powered off and repowered. This need only be done once. The unit will operate correctly from then on.

2. The LOOP command will not allow you to use a Q variable to define the number of iterations of the loop. The 3000 responds to the attempt with the message "Invalid data entered". Regular loops with numbers entered work fine.

Revision levels:

The letter suffix of 3000 software indicates a major revision of the software. The letter after the number indicates a minor revision. So –C on the end of the current Model 3000 software part number says a major revision has taken place (see Technical Bulletin #100). The one in C1 indicates that a minor problem was fixed and was not a major revision of the software.

NOTE

Revision C1 of 3000 software requires revision I software, or higher, in the 1811 axis cards to work properly. (I is the highest revision as of this writing.)

88-010400-01

This errata is intended to accompany the 3000 Indexer software part number 92-5003-01D4 and 92-6897-01D4.

MANUAL CHANGES

CHAIN command description

The CHAIN command is used to reset all stack pointers that the 3000 ONTRIG command may have misaligned. When the 3000 program encounters LOOPs and GOSUBs it will save the line number it should return to when executing ENDLOOPs and RETURNs. These return addresses are stored using stack pointers.

If the 3000 gets the ONTRIG input, the program utilizes a GOTO command. The 3000 exits any loops or subroutines it was performing, but the stack pointers ARE NOT reset. This may cause unpredictable errors in the program. In order to reset the stack pointers, a CHAIN command should be executed.
EXAMPLE:

```
  10 ONTRIG 1XXX XXXX XXXX XXXX GOTO 2000
     .
 200 LOOP 5
 210 LOOP 4
     .
     .
     .
 420 GOSUB 1000
     .
 490 ENDLOOP
     .
     .
 550 ENDLOOP
 560 GOTO 200
     .
     .
1000 WRITE
     .
1050 RETURN
     .
2000 CHAIN 2010 (THIS RESETS THE STACK POINTERS AND GOES TO 2010)
2010 OUTPUT 000 0000 0000 0000
2020 MOVI STOP STOP
2030 GOTO 10
```

NOTE: A CHAIN command is not necessary if there are no LOOPs or GOSUBs in a program.

The user could also completely restart the program over with a START command.

ONSTOP command

(F2, Page 9) The ONSTOP command controls the state (0 or 1) of bits 2 through 15 on the 16-bit programmable output connector. It functions identically to the OUTPUT command except that the execution of the ONSTOP command is delayed until the program stops. The bits can be programmed using 1's, 0's, X's, or Q variables. When program execution stops for any reason, the bit pattern will be output as specified. If Q variables were used in the expression, the values of the Q variable at the time execution stops will be used (not the value of the Q variable at the time the ONSTOP command was executed). If an ONSTOP command is not encountered during a program's execution the programmable output bits will remain unchanged when the program stops.

ENTSTR Command

The following is a new description for the ENTSTR command. The new command allows the user to initialize the contouring mode coordinate system to position 0,0 or the current absolute coordinates at the time the ENTSTR command is executed. Figure 4-49 is no longer current. It should now show:

```
0090ENTSTR YES    YES    ZERO
      YES     NO           ZERO   ABSO
F1     F2     F3     F4    F5     F6
```

(f1, Page 6) The ENTSTR command is used to set the absolute velocity streaming mode for Axis 1 and Axis 2 when performing Velocity Profiling and Contouring. It caused the 3000 to set the velocities on all the indicated axes to zero. This command also establishes a new x,y coordinate system for the Contouring mode of operation. This new x,y coordinate system can be initialized to 0,0 (using the ZERO option) or to the current absolute coordinates at the time the ENTSTR command is issued (using the ABSO option). If neither ZERO or ABSO are specified, the default is ZERO. See Appendix B, Section B.2.1, Velocity Streaming, and Section B.2.2, Contouring, for examples of programs using absolute velocity streaming.

NOTE:

For instructions on how to use the Streaming function in your application, refer to the discussion of Velocity Profiling described in Section 4.3.5.2.

Each axis that is to begin absolute velocity streaming must be set to YES.

WRITE command

The manual incorrectly describes the WRITE command. If

```
WRITE     5     30     10     Q15
```

is entered, the manual-described command would clear columns 5 through 30 and put Q15 at column 10. What really happens is 30 columns are cleared starting at column 5. The Q15 response is displayed starting at column 10.

LIMITS command
HOME command

The only exception to the LIMITS=ABORT? YES option is during a HOME command. The HOME command allows the axis or axes going home to reverse direction when a limit is encountered the first time. An axis that encounters a limit for a second time during a HOME command will stop immediately and if more than one axis was specified in the HOME command the program will not abort until all axes have finished (successfully or unsuccessfully) their HOME sequence. If only one axis was specified in the HOME command, the program will abort immediately after encountering a second limit.

OTHER CHANGES

Allow saving of literals. By placing ^'s around a string variable (Q101 and greater), the variable ~~is~~ saved "as is". A variable such as ^A5 V5 D12800 G^ can be saved. The carets (^) are not saved as part of the string; i.e. the variable is reported back as:

```
A5 V5 D12800 G ;
```

If a closed loop move was stopped and then resumed, the move would be completed in motor steps rather than encoder steps. The move will now be resumed in one of two ways depending upon the revision level of the 1811 software. If it is prior to revision level K the move will be resumed open-loop. This may lead to some inaccuracy. The revision level K software introduces a new closed-loop resume command (37H) which is used to complete the move.

Appendices

List of Figures

x

List of Tables

1. INTRODUCTION

1.1. Overview

The Compumotor 3000 Indexing System[1] (Figure 1-1) is a programmable indexing system capable of controlling up to four axes (motors) at one time. It is programmed and controlled from the Front Panel using a keypad and an alphanumeric liquid crystal display, which combine with a menu- driven control program and non-volatile memory to allow you to enter and store motion sequences. It can also be controlled by a remote computer or terminal via an RS-232C or IEEE-488 interface. There are programmable inputs and outputs for each axis, compatible with OPTO-22 signal conditioning products, that allow the 3000 to interact with the rest of your system.

The 3000, which is actually a front end for up to four Compumotor Model 1811 single-board Indexers, will drive any size Compumotor motor/drive set or any standard 200/400-step translator and stepper motor combination. Depending on the particular Compumotor motor/drive used, a single revolution can be subdivided into 1,000, 21,600, 25,000, 25,400, 36,000, or 50,000 individual steps. Using the Indexer, you can rotate the motor shaft to a precise position and stop; rotate at a constant velocity; alternate back and forth between two angular positions; or use a combination of such moves.

The 3000 is normally used for precise position control with no external position feedback. An optional operating mode (Encoder Interface) is available for closed-loop operation to further enhance the accuracy of position control and increase the "stiffness" of the motor. This is accomplished by using an incremental optical Encoder and requires the use of Model 1811 single-board Indexers.

A combination of successive positions, motions, accelerations, decelerations, and rotational velocities can be programmed. The commands are described in detail in Chapter 4.

Note

The Compumotor Driver and the Compumotor Motor, which are a matched pair, are referred to as the motor/drive in this manual. They are not built into the 3000.

---

[1]Referred to as 3000 throughout this manual.

1.2. General Description

The 3000 consists of the Front Panel, a controller, and up to four Compumotor Model 1811 single-board Indexers that communicate with the controller via an IEEE-796 Multibus. Figure 1-2 is a block diagram of the unit.

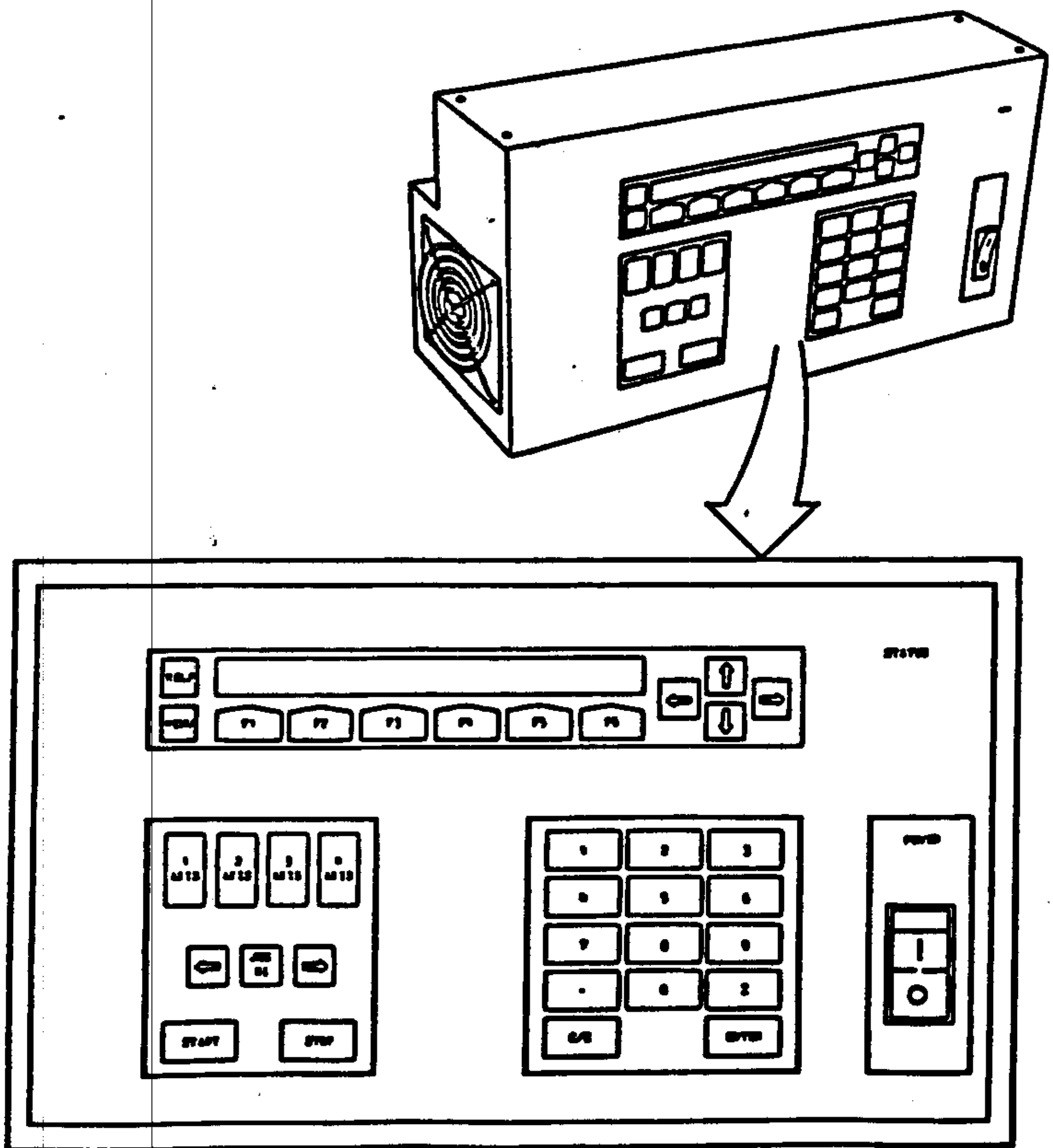


Figure 1-1: Compumotor 3000 Indexing System

Four closed-loop models are available, as listed in Table 1-1. The closed-loop models (-P) have provisions for connecting to a TTL-compatible incremental optical encoder. Table 1-1 also lists four open-loop models that are no longer being manufactured.

Table 1-1: Indexing System Model Numbers

Number of Axes	Open Loop Model No.	Closed Loop Model
1	3000-1	3000-1-P
2	3000-2	3000-2-P
3	3000-3	3000-3-P
4	3000-4	3000-4-P

Programs that are saved using the SAVE command are loaded from Main Memory into the 3000's resident 4 Kbytes of electrically erasable programmable read-only memory (EEPROM), referred to as non-volatile memory (NVM). Programs that have been saved can be recalled from NVM into Main Memory using the RECALL command. This process is illustrated in Figure 1-3. The number of program statements or lines allowed depends on the number of axes controlled by the unit, as shown in Table 1-2. However, RS-232C or IEEE-488 compatible storage devices can be used to store larger programs.

Table 1-2: Maximum Program Steps vs Number of Axes

Number of Axes	Maximum Number of Program Steps
1	305
2	260
3	200
4	165

1.3. Optional Features

The standard 3000 configuration controls the motion of one axis (-1). The -2, -3, and -4 versions control two through four axes, respectively. The Encoder Interface option (-P) specifies that all axes are equipped with Model 1811 single-board Indexers.

Figure 1-2: 3000 Indexer Block Diagram

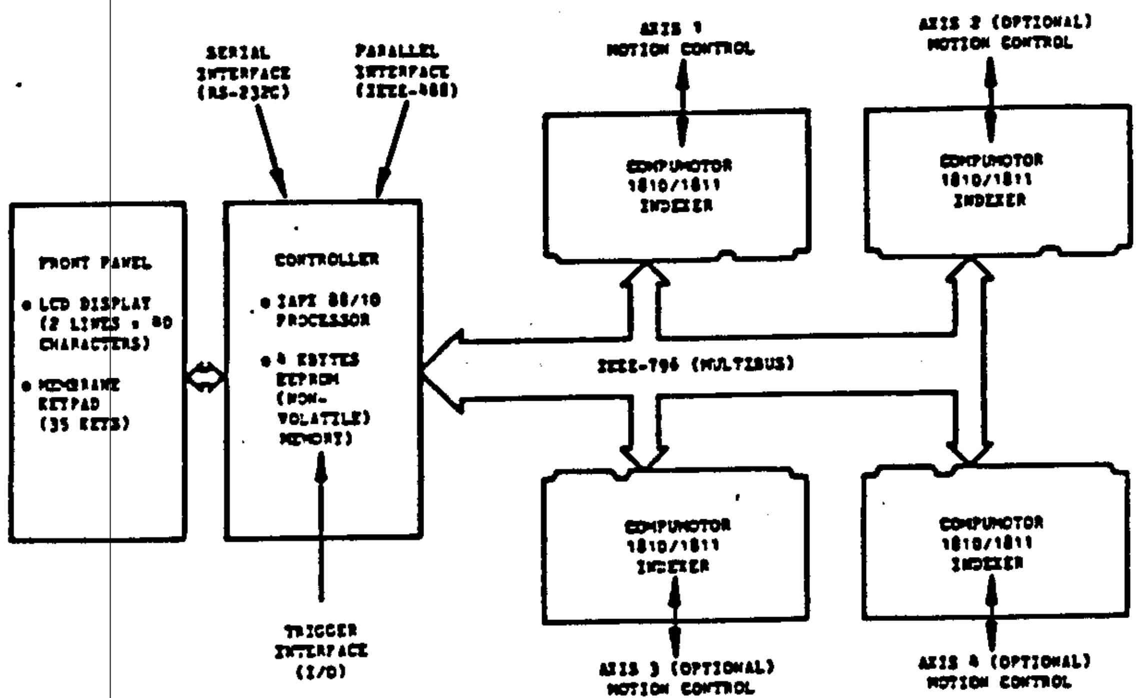
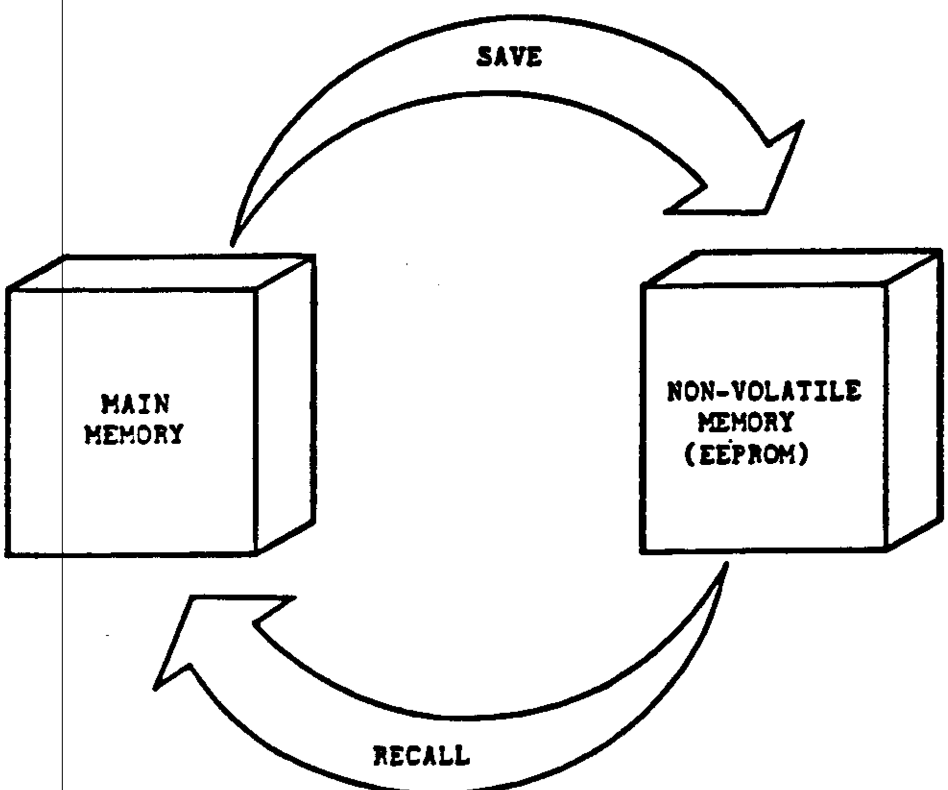


Figure 1-3: Main and Non-Volatile Memories



1.4. Specifications

Performance, electrical, environmental and mechanical specifications are presented in Tables 1-3 and 1-4. Input and output specifications are provided in Chapter 3.

1.5. How to Use this Manual

You do not have to be an expert in any field to use the 3000 Indexing System. This manual contains all information you will need to install and successfully operate the 3000. If a terminal or computer is intended for control, you will need access to the computer or terminal manuals that define setup conditions for RS-232C or IEEE-488 operation of the respective terminal or computer. This manual assumes that you have access to the applicable equipment manuals.

Chapter 2 presents the concepts and terminology used in the manual. Read this chapter completely.

Chapter 3 describes installation and checkout procedures for the 3000. Separate sections are included in Chapter 3 that describe installation of the parallel and serial interfaces, limit switches, and optional features.

Chapter 4 describes programming and operating procedures and commands.

Appendix A lists the commands used to program and operate the 3000.

Appendix B contains programming examples.

Appendix C provides procedures for field installation of the Model 1811 single-board Indexer.

Appendix D provides dimensions for installation.

1.6. Technical Assistance

If you need Technical Assistance, please contact your Compumotor distributor or:

Customer Service Department
Compumotor Corporation
1179 North McDowell Boulevard
Petaluma, CA 94952
(707) 778-1244 (Inside California)
(800) 358-9068 (Outside California)

Do not ship the unit to the factory for service until you have been given an authorization number by the factory.

Table 1-3: 3000 Series Performance and Electrical Specifications

Parameter	Value
Performance:	
Stepping Accuracy	±0 Steps from Preset total
Velocity Accuracy	±0.02% of max rate
Velocity Repeatability	±0.02% of Set rate
Velocity Range	0.017 steps/secto 1,092,000
steps/sec	
Power (50/60 Hertz):	
100-130 VAC input	1.0 Amps maximum

Table 1-4: 3000 Series Mechanical and Environmental Specifications

Parameter	Value
Size:	
Height	8.5 in. (21.6)
Width	16.0 in. (40.6 cm)
Depth	6.3 in. (16.0 cm)
Weight Net	17 (7.7) Pounds (Kg)
Shipping	25 (11.4)
Operating Temperature	32 to 122°F (0 to 50°C) 0 to 95% Humidity, non-condensing
Storage Temperature	-22 to 185°F (-30 to 85°C)

2. BASIC SYSTEM CONCEPTS

2.1. Overview

All Indexer/Indexer System applications involve either rotation of a Compumotor motor shaft to a precise angular position (number of steps) or rotation of the motor shaft at a prescribed angular velocity (revolutions per second). In most models the Indexer, Motor, and motor/drives are separate assemblies. In some of the models the Indexer and Driver are combined into one assembly.

A technique known as microstepping, which is described in detail the Compumotor Product Catalog, is used to obtain high-resolution rotary motion. A standard translator/stepper motor can be stepped in equal increments of 200 or 400 steps per revolution (steps/rev), making each step equivalent to 1.8 or 0.9 degrees. Microstepping permits as many as 50,000 steps/rev, depending on the design of the motor/drive. You can choose from Compumotor motor/drive options ranging from 1,000 to 50,000 steps/rev for your application. Both the size and the resolution of the Compumotor motor/drive must be determined for any given application. Refer to the Compumotor Product Catalog for a discussion of motor characteristics, torque calculations, and selection criteria.

Each time the Compumotor motor/drive receives one step pulse, it rotates the motor one angular increment. You may choose to control either angular position or angular velocity with a high degree of precision without any external feedback.

When commands are structured for the system, it is necessary to think in terms of the number of steps/rev; e.g., for a 25,000-step motor, 180 degrees = 12,500 steps, 1 revolution = 25,000 steps, etc. You must refer the desired load motion for your application to rotation of the Compumotor motor shaft. Shaft motion is specified in terms of acceleration, angular velocity, and angular position.

2.2. Acceleration/Velocity Profiles

The motor moves in incremental steps. In practical applications, load inertia tends to smooth the actual load motion. The small size of the motor steps minimizes end-of-move settling time at moderate acceleration values.

2.2.1. Acceleration/Deceleration

Acceleration/deceleration is defined as the rate of increase/decrease of angular velocity in steps, or revolutions per second per second (rev/sec/sec). When the Indexer drives the motor from rest to a given velocity, a string of pulses is sent to the motor/drive.

Relatively high pulse rates are used to drive the high-resolution Compumotor Motors; therefore, the shaft velocity buildup can be graphically portrayed by a straight line, as shown in Figure 2-1.

The acceleration/deceleration rate must be selected as a function of the inertia of the load to allow operation within the torque limits of the motor. A margin of 50 to 100 percent of normal load torque requirements is recommended. If the selected acceleration rate requires torques that exceed the capability of the Compumotor motor/driver, the motor will stall without any damage to it.

The acceleration/deceleration rate can be varied between broad limits. When very high acceleration rates are used, there may be a tendency for overshoot and longer setting times. Normally the acceleration and deceleration rates are equal.

2.2.2. Motion Profiles

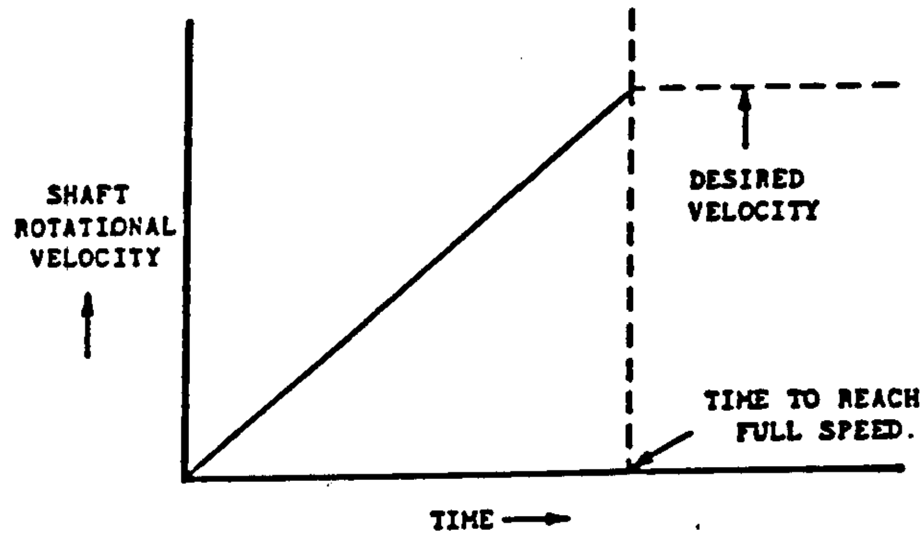
Refer to Figure 2-2. The trapezoidal motion pattern (Figure 2-2A) will occur when the angular distance to be covered is large compared with the angular distance required to reach the specified angular velocity. Part of the motion occurs at a constant velocity. An example is a move in which one-fourth of the move time is acceleration, one-half of the time velocity is constant, and one-fourth of the time is deceleration. The triangular motion pattern (Figure 2-2B) occurs when the angular distance required to reach the specified velocity is large compared to the angular distance to the new position. In this case the motor shaft velocity continually changes during the move, and there is no constant velocity phase. The average velocity would be one-half the peak velocity.

2.3. Motion Characteristics

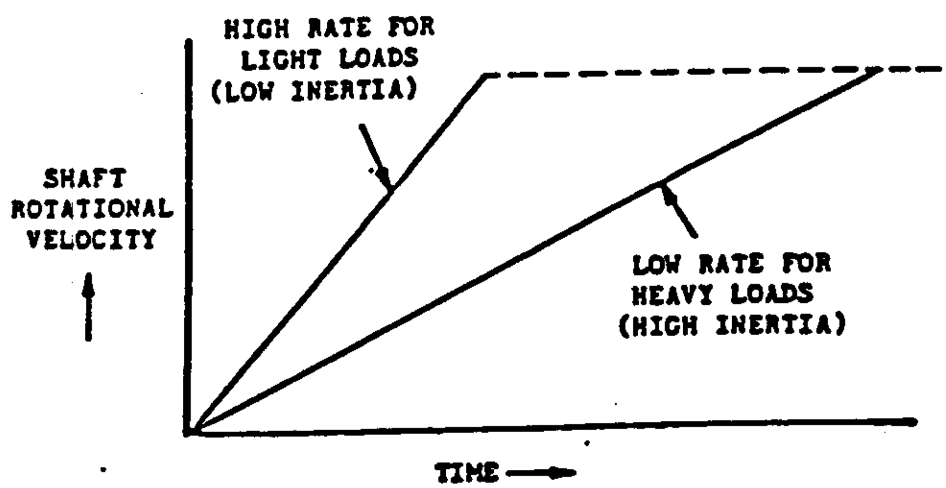
2.3.1. Motor Resolution

Motor resolution is defined by the motor/drive used with the Indexer (Table 2-1).

Figure 2-1: Velocity Buildup

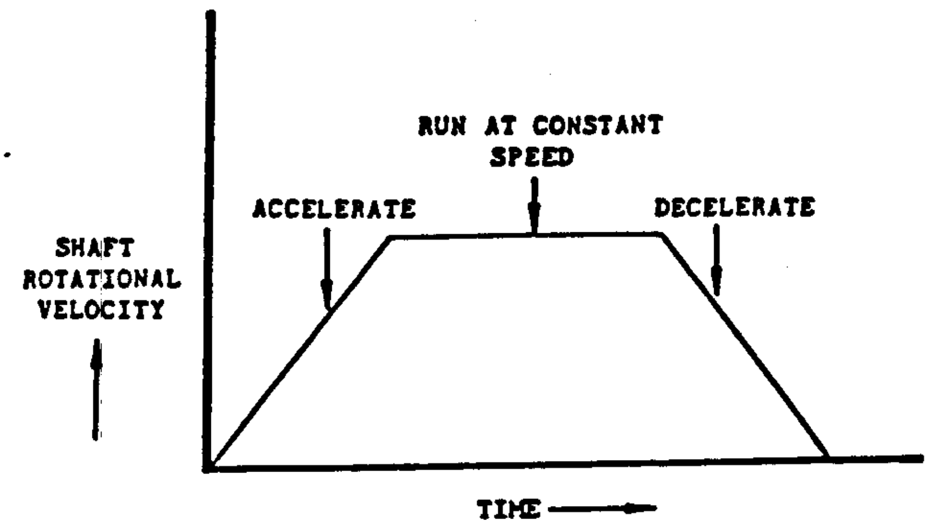


A. ACCELERATION CURVE

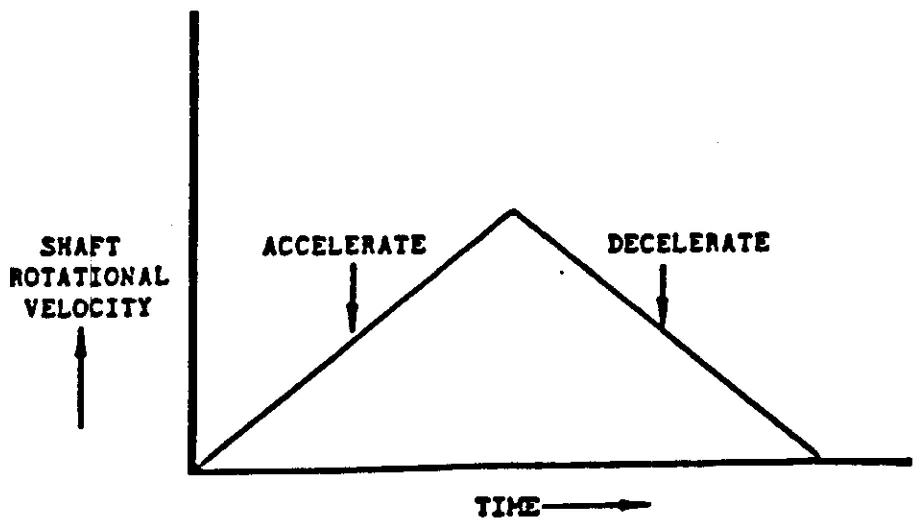


B. INDEXER CAPABILITY

Figure 2-2: Motion Profiles



A. TRAPEZOIDAL MOTION PATTERN



B. TRIANGULAR MOTION PATTERN

Table 2-1: Resolution Per Step

Motor Resolution (steps/rev)	Equivalent Value Per Step part of rev*	min of arc*
50,000	2.0000×10^{-5}	0.4320
36,000	2.7778×10^{-5}	0.6000
25,400	3.9370×10^{-5}	0.8504
25,000	4.0000×10^{-5}	0.8640
21,600	4.6296×10^{-5}	1.0000
4,096	2.4410×10^{-4}	5.2734
1,000	1.0000×10^{-3}	21.6000
400	2.5000×10^{-3}	54.0000
200	5.0000×10^{-3}	108.0000

* Rounded values

2.3.2. Velocity Range

Velocity ranges can be selected via the command interface.

2.3.3. Acceleration/Deceleration Range

The acceleration range can be selected via the command interface. The basic considerations when selecting acceleration range are the torque capability of the motor, risk of load overshoot, and the physical characteristics of the mechanical transmission coupled to the motor/drive.

2.4. Analysis of Your Applications

The basic elements that are present in most applications are shown in Figure 2-3.

2.4.1. General Considerations

2.4.1.1. Limit Switches

In all cases you must determine whether limit switches will be required to protect the load. (See Chapter 3 for limit switch installation instructions.)

2.4.1.2. Encoders

Determine if Encoder feedback is required. When you select the Encoder Interface, you have a number of choices, as follows:

a. You can mount the Encoder at the load and assure the proper positioning of the load despite any gear play or friction effects of the mechanical transmission between the motor and the load.

b. You can use a motor with the Encoder mounted on the motor shaft so that the motor will resist movement induced by externally applied torques.

c. You can use the Encoder as a stall detector to identify whether the motor has stalled.

d. You can use the Encoder to independently verify the load position.

2.4.1.3. Discrete Inputs

Determine if the discrete inputs for START, STOP, COUNTERCLOCKWISE LIMIT, and CLOCKWISE LIMIT inputs are required.

2.4.1.4. Motor Steps

Determine the number of steps required to reach a desired angular position of the Compumotor motor shaft based on the angular resolution of your system.

2.4.1.5. Motor/Drive Sizing

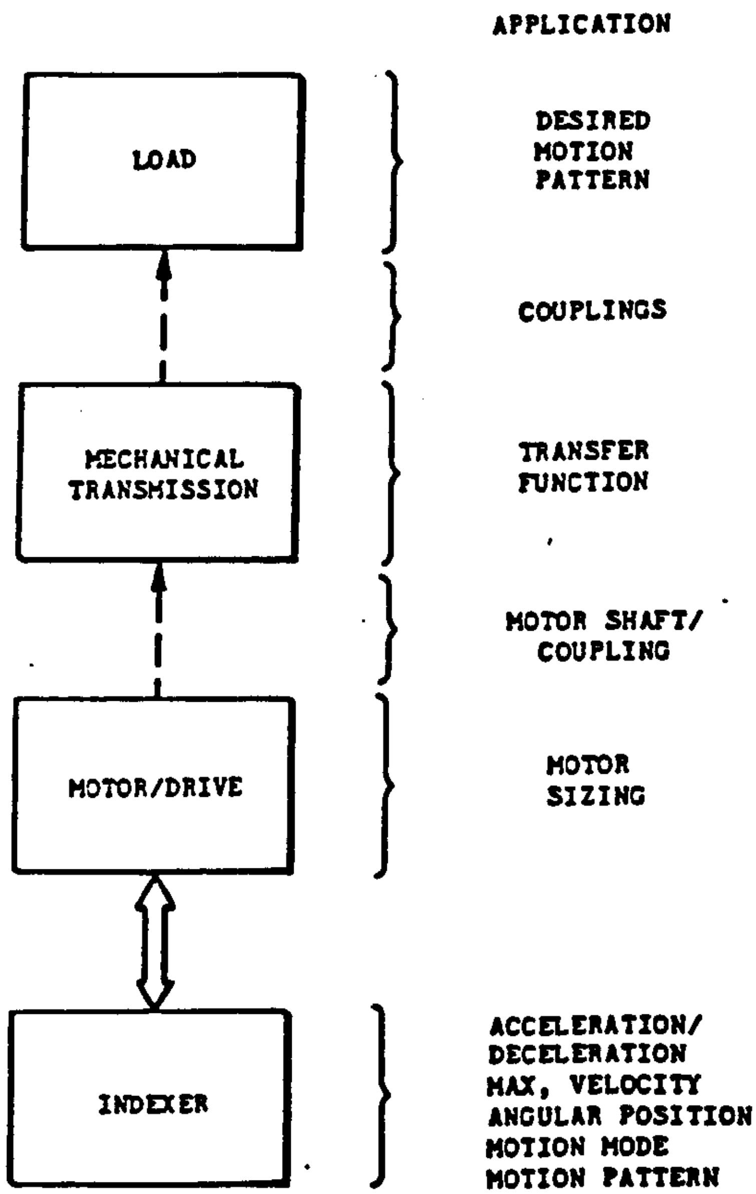
The motor/drive must be sized for the application. Refer to the Compumotor Product Catalog for sizing guidelines.

2.4.2. Definition of Acceleration and Velocity

The desired load motion pattern must then be referenced to the Compumotor motor shaft. The motor/drive acts as a relatively stiff drive source. Theoretically you could use detailed mathematical analyses to determine the proper values of position, acceleration, and velocity. In addition to acceleration, velocity, and position, your analysis must include a definition of any gear play, coupling effects, transmission elasticity, load inertia, load friction, etc. In practical situations use of such detailed analytic techniques is not necessary and is frequently undesirable. Alternatively, you can approximate an initial set of position, acceleration, and velocity values. These approximations can then be tested in the actual application and revised as necessary.

A motor/drive will stall (without any damage) whenever the performance specified by a command exceeds the capability of the motor/drive being used. You can experimentally determine the actual velocity and acceleration values. You can set the Indexer to the set of approximate values and then adjust the values to achieve the desired performance.

Figure 2-3: Basic Elements to Be Analyzed



2.5. Equipment Configuration Alternatives

2.5.1. Simplest Configuration

The simplest configuration (Figure 2-4) consists of a basic Indexer and a Compumotor motor/drive connected to a load.

2.5.2. Limit Switches

Limit switches can be installed on the load (which is recommended) and connected to the Indexer to protect the load from overtravel. This is shown in Figure 2-5.

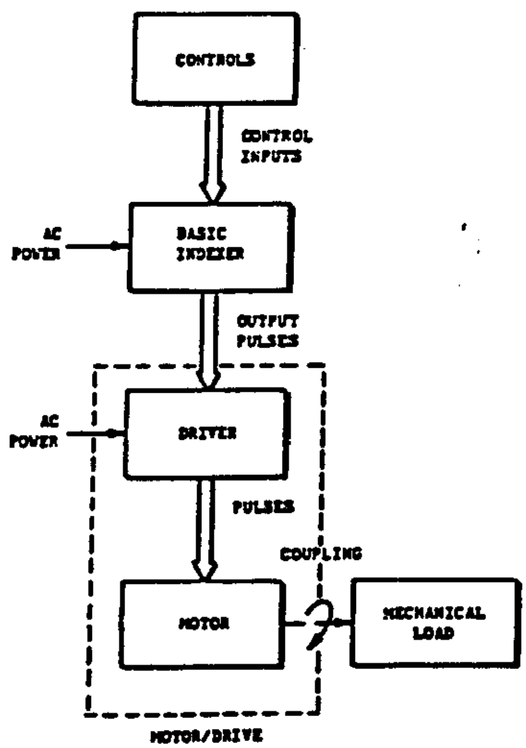


Figure 2-4: Simplest Configuration

Figure 2-5: Limit Switch Input

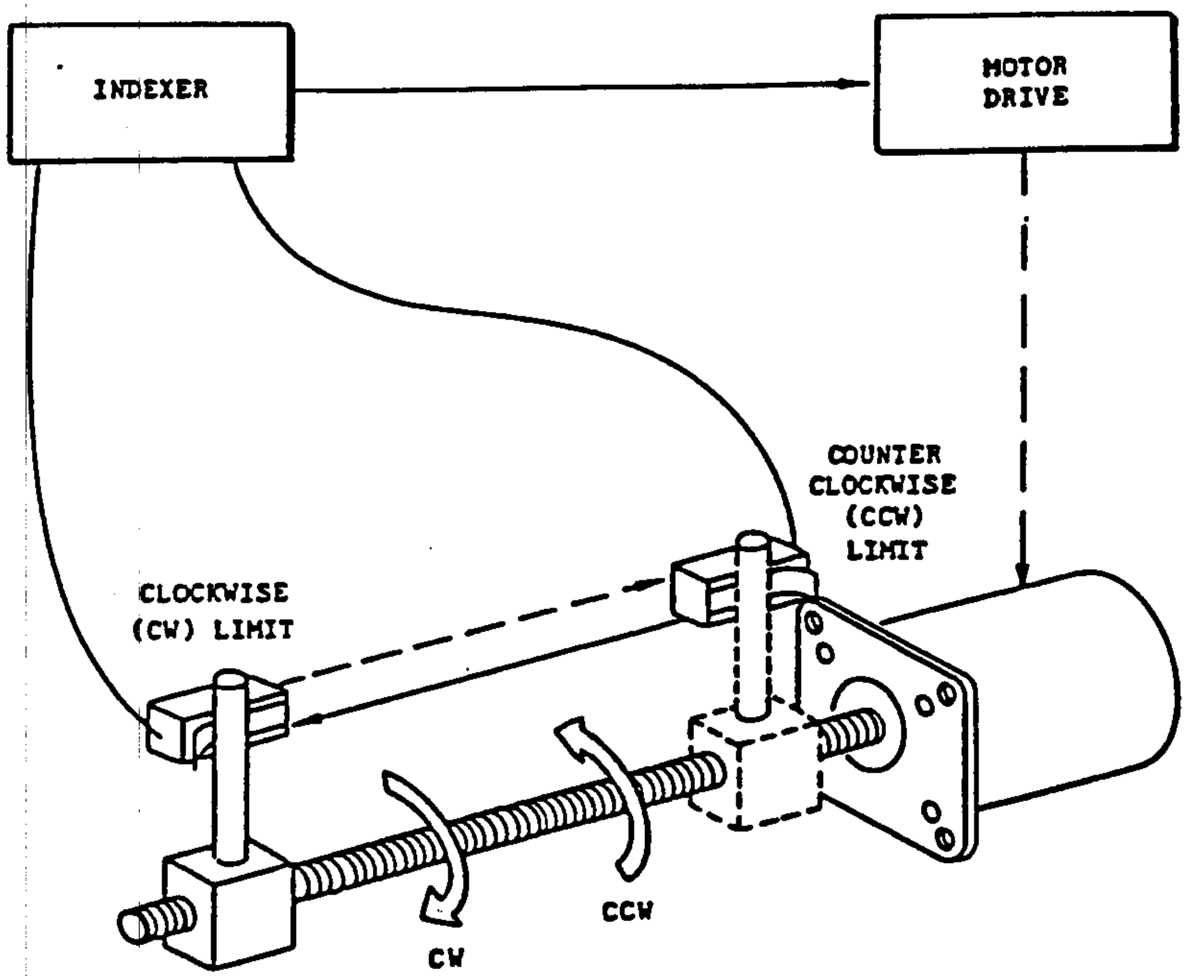
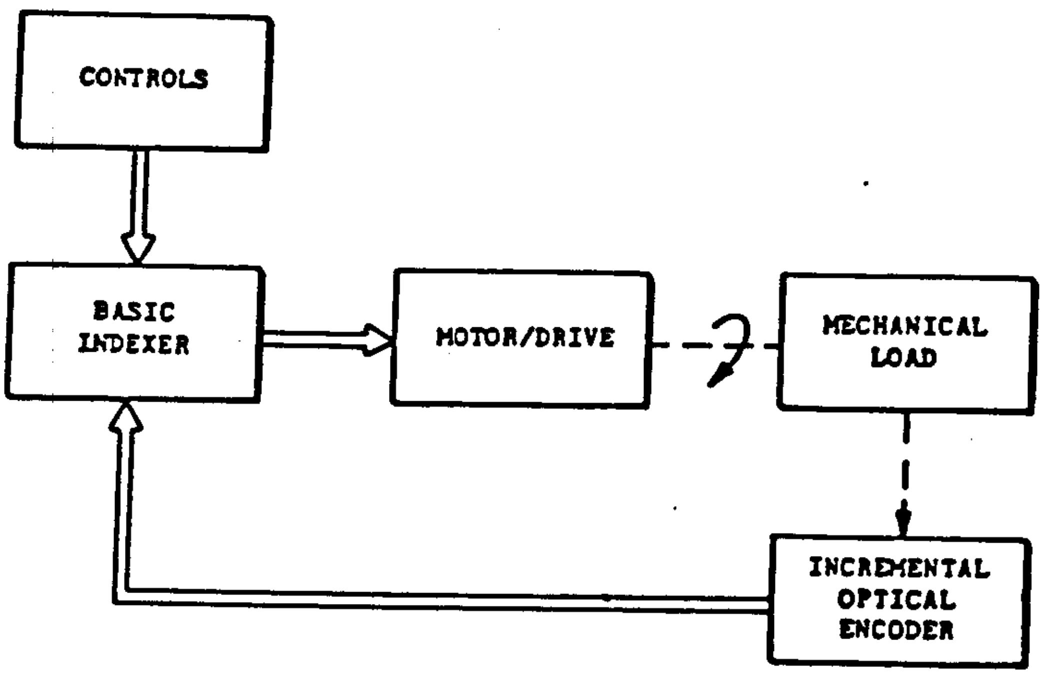


Figure 2-6: Addition of Encoder Interface



2.5.3. Encoder Interface

The Encoder Interface can be used, as shown in Figure 2-6, to

a. Provide closed loop operation for improved load positioning accuracy.

b. Independently verify position.

c. Increase resistance of the motor to perturbing external torques.

d. Detect motor stall.

2.5.4. Indexer Controller

A programmable controller or computer can be used for controlling the Indexer, as shown in Figure 2-7.

One advantage of this approach is that command messages can be stored and used repeatedly at different times. A number of programs can be stored and then recalled in response to load conditions. Operation can be automatic and interactive if the proper load sensors and process controllers are installed.

2.5.5. Multiple Function Assemblies

Multiple function assemblies are also available. Some specific examples are:

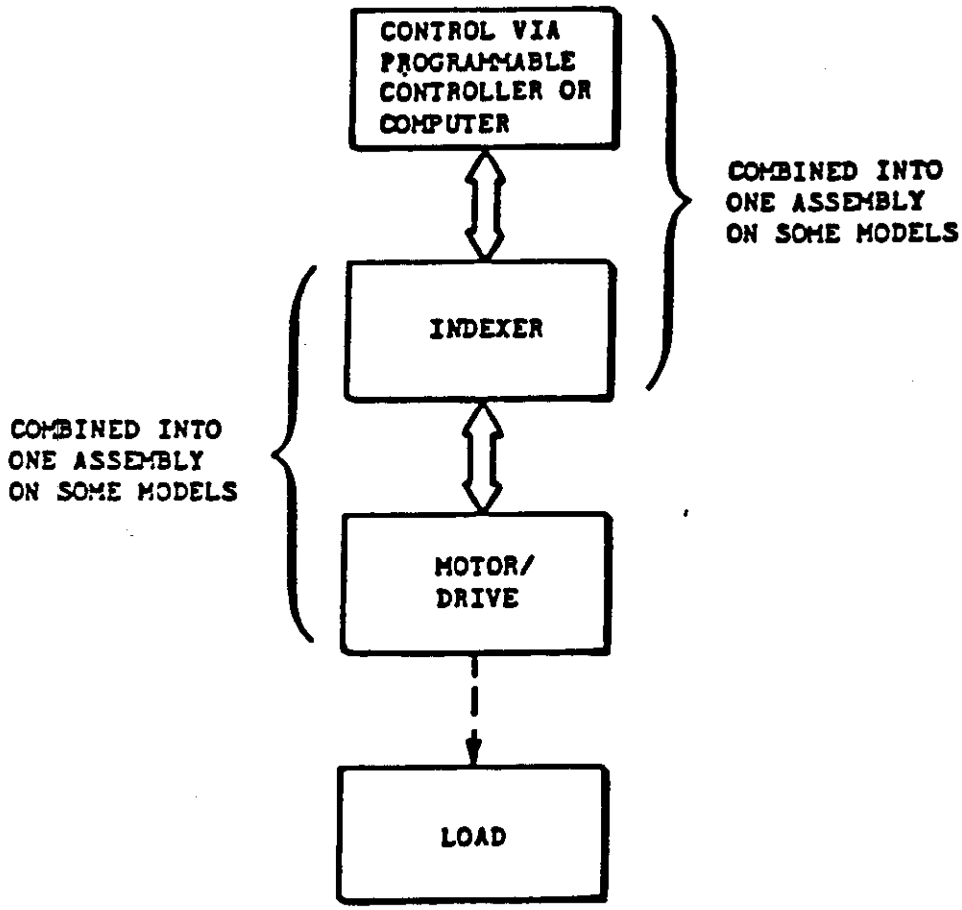
a. Basic Indexer plus power supply plus driver is one package (CX Series).

b. Local control panel plus basic Indexer plus power supply plus Encoder Interface plus IEEE-488 interfaces is one package (2100 Series)

c. Liquid crystal display control panel plus four basic Indexers plus power supply is one package (3000 Series).

Consult with your Compumotor representative to select the proper equipment models and options for your application.

Figure 2-7: Indexer Controlled by Programmable Controlled Computer



3. INSTALLATION

3.1. Overview

This chapter provides installation and test procedures for the 3000 Indexing System. Instructions for installing 1811 axis cards into the 3000 are provided in Appendix C.

3.2. Inspection

Inspect the shipping carton carefully for evidence of physical abuse or damage. Report any such findings immediately to your receiving department and to the carrier. Compumotor Corporation cannot be responsible for in-transit damage.

Unpack the shipping carton and inspect the Indexer for any damage, cracks broken parts, or damaged cables. Save the packing materials until functional checks have been completed.

3.3. Setup

Each 3000 is ready to operate when removed from the packing container. The unit is powered from 120 VAC, 50/60 Hz. The mounting ears on each side of the unit are reversible, so the 3000 may be either rack mounted or bolted to a flat surface (such as the back of a NEMA rack) with the Control Panel facing out. The 3000 is fan-cooled, and the areas around the fan filters on the left and right sides of the unit must be left clear and unobstructed. The cover must be on during operation or the circuit cards can overheat due to poor air flow.

The 3000 features password-protected programming. This means that only a limited number of operations may be performed after turning on the unit without entering a password. These operations are: RECALL (a program from non-volatile memory), RUN (a program), RESUME (a program), and ACCESS (enter a password). The ACCESS command must be performed before any programming is allowed. After turning on the unit, press the function key labeled ACCESS and enter the number 3000, followed by the ENTER key. The factory-set access code is "3000," but you may change it if you desire (see Section 4.3.3.4).

Note

The 3000 requires that limit switches or bypass jumpers be connected to each axis' Auxiliary Connector. The 3000 will not move motors unless this is done (See Figure 3-1).

3.4. AC Power Connector

AC power is applied to the Rear Panel of the 3000 via the standard three-pole AC power cord supplied with the unit.

3.5. Communications Ports and Programmable I/O

The 3000's connectors are all located on the Rear Panel of the unit. There are four connectors that are independent of the number of axes controlled by the unit: two programmable input/output ports and two communications interfaces (RS-232C and IEEE-488).

The RS-232C or IEEE-488 interfaces can be used to communicate with a computer, load programs from another 3000, send them to another 3000, or print them on a printer. They can also be used to allow the 3000 to receive or transmit parameters during operation. This powerful feature permits real-time modification of parameters to accommodate for tool offsets, scaling, process variations, etc.

The programmable input/output connectors can be used to make your 3000 interactive with the rest of your system by accepting triggers caused by external events or by providing a signal to trigger external events.

The four connectors are described in Sections 3.5.1 through 3.5.3.

3.5.1. IEEE-488 Interface

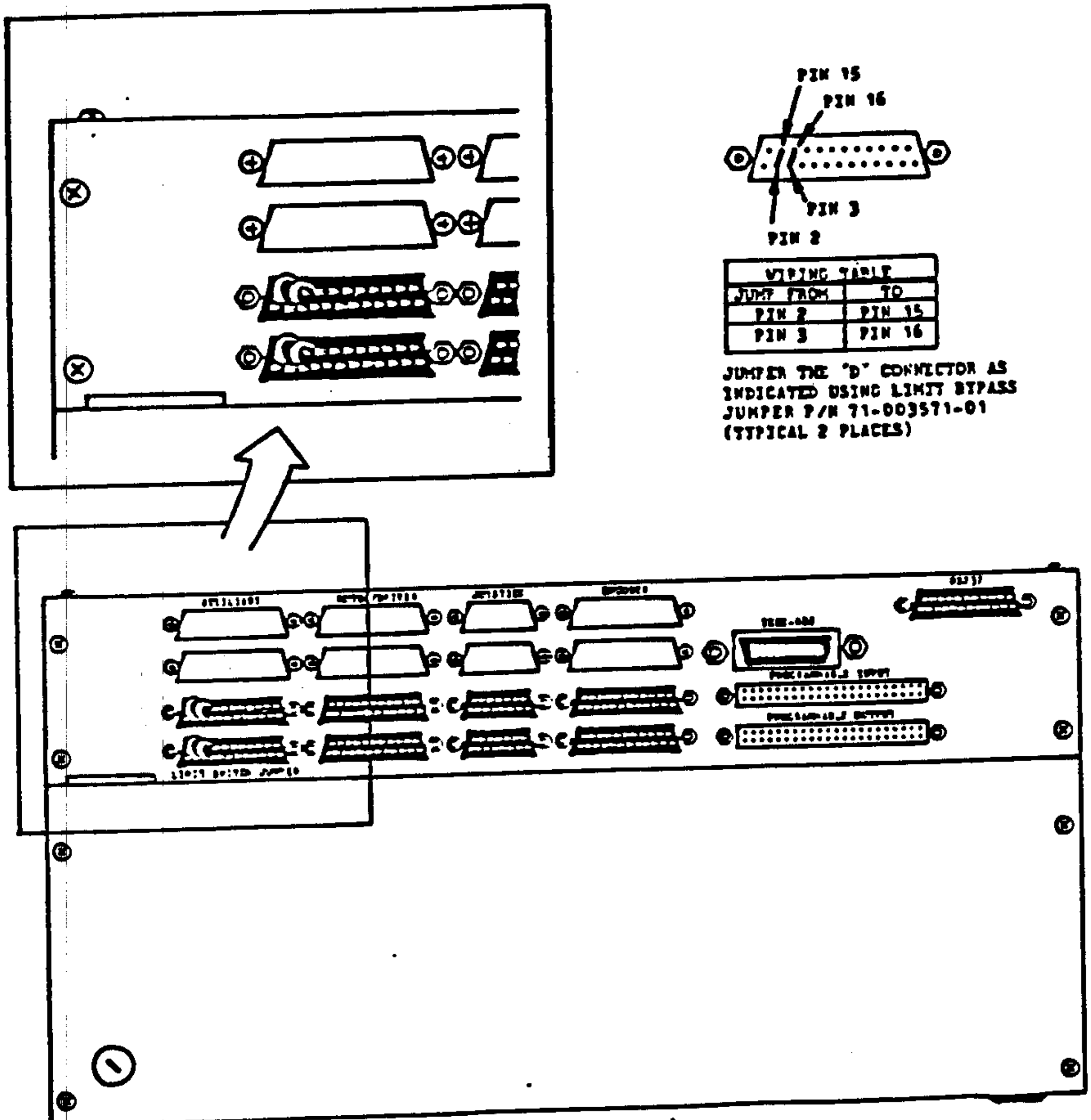
This 25-pin, double-row, leaf connector complies with the IEEE-488 specification and connects to an IEEE-488 bus device. The 3000 functions as a talker/listener; its device address is specified with the SETUP Menu and is stored in the unit's non-volatile memory. Table 3-1 lists the pin assignments for this connector.

3.5.2. RS-232C Connector

This 25-pin, female "D" connector provides a three-wire RS-232C interface. The interface can be operated at all standard baud rates between 110 and 19,200 and can be configured as DTE (Data Terminal Equipment) or DCE (Data Communications Equipment). DTE and DCE operation differ in the pin used to transmit and receive data. DTE or DCE operation is selected with the SETUP function from the Front Panel (Section 4.3.3.1), eliminating special cables. The number of stop bits, number of data bits, parity, and baud rate are also specified with the SETUP function and can be stored in the unit's non-volatile memory.

The pin assignments for this connector are listed in Table 3-2.

Figure 3-1: Limit Switch Jumpers



WIRING TABLE	
JUMP FROM	TO
PIN 2	PIN 15
PIN 3	PIN 16

JUMPER THE "D" CONNECTOR AS INDICATED USING LIMIT BYPASS JUMPER P/N 71-003571-01 (TYPICAL 2 PLACES)

Table 3-1: IEEE-488 Connector Pin Assignments

Number	Signal
1	DIO1
2	DIO2
3	DIO3
4	DIO4
5	EO1
6	DAV
7	NRFD
8	NDAC
9	IFC
10	SRQ
11	ATN
13	SHIELD (CASE GROUND)
14	DIO5
15	DIO6
16	DIO7
17	DIO8
18	REN
19	SIGNAL GROUND (DC COMMON)
20	SIGNAL GROUND (DC COMMON)
21	SIGNAL GROUND (DC COMMON)
22	SIGNAL GROUND (DC COMMON)
23	SIGNAL GROUND (DC COMMON)
24	SIGNAL GROUND (DC COMMON)

Table 3-2: RS-232C Connector Pin Assignments

Pin Number	Signal
1	SHIELD (CASE GROUND)
2	TRANSMIT or RECEIVE
3	RECEIVE or TRANSMIT
7	SIGNAL GROUND (DC COMMON)
All other pins	(N.C.)

3.5.3. Programmable Input and Output Connectors

There are two 50-pin flat cable connectors on the rear of the 3000 that are compatible with OPTO-22 optically isolated signal-conditioning panels and can be used for programmable I/O. The 3000 has 16 programmable inputs and 15 programmable outputs. The sixteenth output is a hard-wired machine status output. This status output is ON (true) when the 3000 is operating properly and OFF (false) when it is not.

For most applications I/O is attached to the 3000 through the OPTO-22 signal-conditioning panels. One flat cable connects the 3000 to a signal-conditioning panel used for input, and another flat cable connects the 3000 to another signal-conditioning panel used for output. These panels, or "module mounting racks," are available in 8 and 16-line versions. Separate racks are required for input and output.

OPTO-22, Crydom, and Potter & Brumfield all manufacture signal-conditioning products suitable for use with the 3000. Each position in the rack can have any OPTO-22 I/O module installed in it. These modules come in different voltage and current ratings. There are versions for interfacing with signals ranging from 5 VDC to 280 VAC. Consult your local OPTO-22, Crydom, or Potter & Brumfield distributor for details.

The following OPTO-22 module mounting racks and flat cables work with the 3000:

a. Module mounting racks

   (1) PB8 - 8 lines

   (2) PB16A - 16 lines

Note

A jumper must be placed between the "+" solder pad and the solder pad next to pin 49 on each module mounting rack (OPTO-22 board) for the unit to operate. This jumper provides +5 volts from the 3000 to power the board. If this jumper is not installed, you must provide +5 Volts to power the board.

b. Cables

   (1) OD-2 - 2 feet

   (2) OD-4 - 4 feet

   (3) OD-8 - 8 feet

   (4) OD-10 - 10 feet

Table 3-3 lists the pin assignments for both connectors. All even numbered pins are connected to logic ground (DC ground).

All 3000 outputs are driven with 74LS240 drivers. All inputs are TTL-compatible.

3.5.3.1. Optically Isolated Outputs

A schematic diagram of a typical optically isolated output circuit is shown in Figure 3-2. These circuits are not part of the 3000, but are typical of circuits available from other manufacturers. These circuits are open-collector, optically coupled transistors capable of sinking 100 milliamps when switched ON. When OFF, they behave like an open circuit unless pulled up through resistors to an external voltage source.

Table 3-3: Programmable Input and Output Connector Pin Assignments

Flat Cable Pin No.	Output Connector	Input Connector
47	Output 1 (dedicated machine status output)	Input 1
45	Output 2	Input 2
43	Output 3	Input 3
41	Output 4	Input 4
39	Output 5	Input 5
37	Output 6	Input 6
35	Output 7	Input 7
33	Output 8	Input 8
31	Output 9	Input 9
29	Output 10	Input 10
27	Output 11	Input 11
25	Output 12	Input 12
23	Output 13	Input 13
21	Output 14	Input 14
19	Output 15	Input 15
17	Output 16	Input 16
49	+5 VDC	+5 VDC

3.5.3.2. Optically Isolated Inputs

A schematic diagram of a typical optically isolated input is shown in Figure 3-3. These circuits are not part of the 3000, but are typical of circuits available from other manufacturers. These inputs are current- sourcing inputs and require sinking outputs or contact-closure type outputs to drive them. TTL output modules in a low state exhibit the proper current-sinking characteristics. Open-collector circuits also exhibit the proper current-sinking characteristics.

Optically isolated inputs can be driven by switching them to ground or sinking a 15-milliamp current to ground. The inputs are pulled up to an internal +5 VDC source. The same source is available on pin 49 (+5 VOLTS) for external use.

Internal blocking diodes allow input voltages of up to 30 VDC to be applied to the optically isolated inputs. The polarity of the input signals is set at the factory for high - false, low - true.

Figure 3-2: Typical Example of Optically Isolated Output

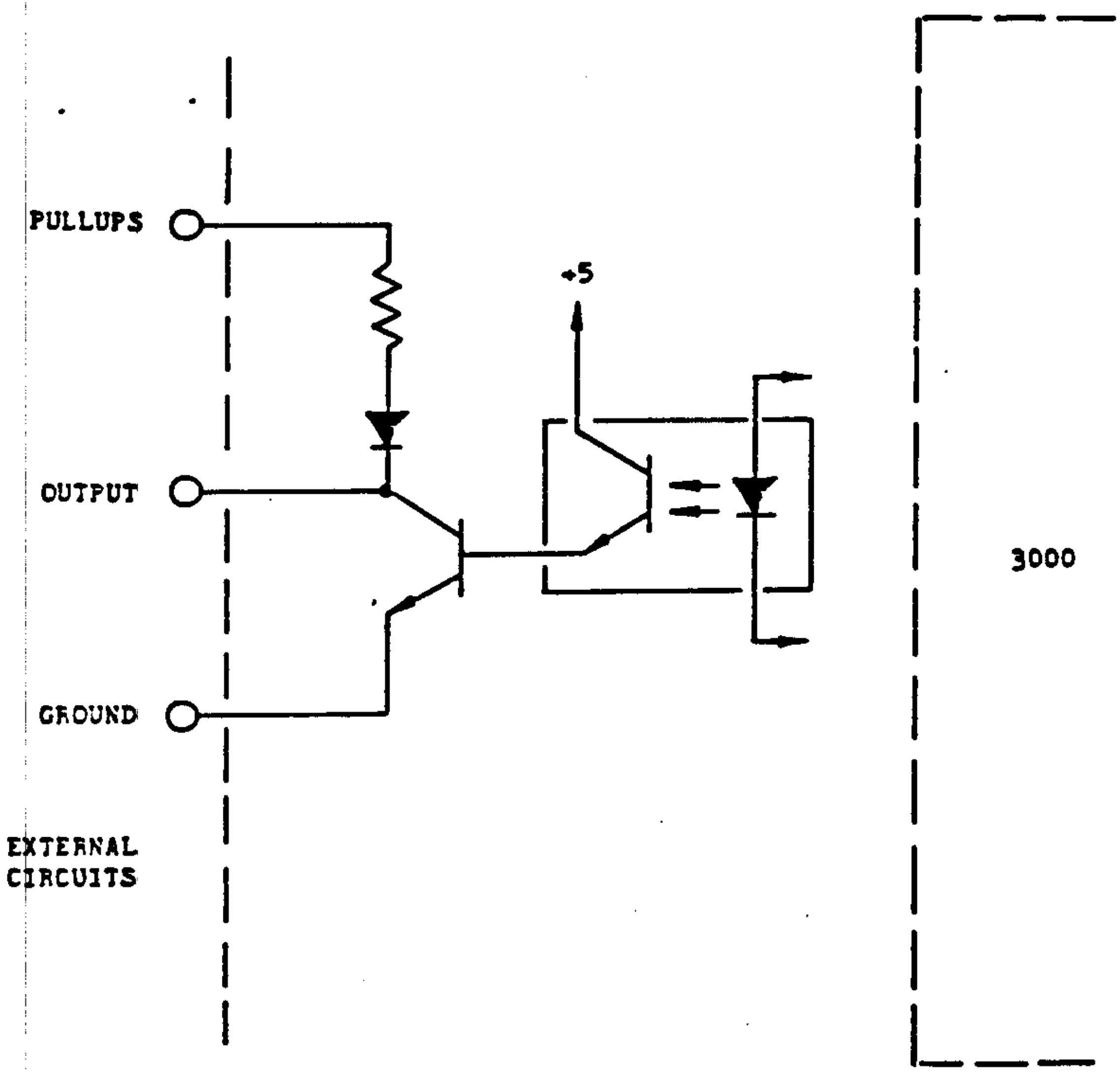
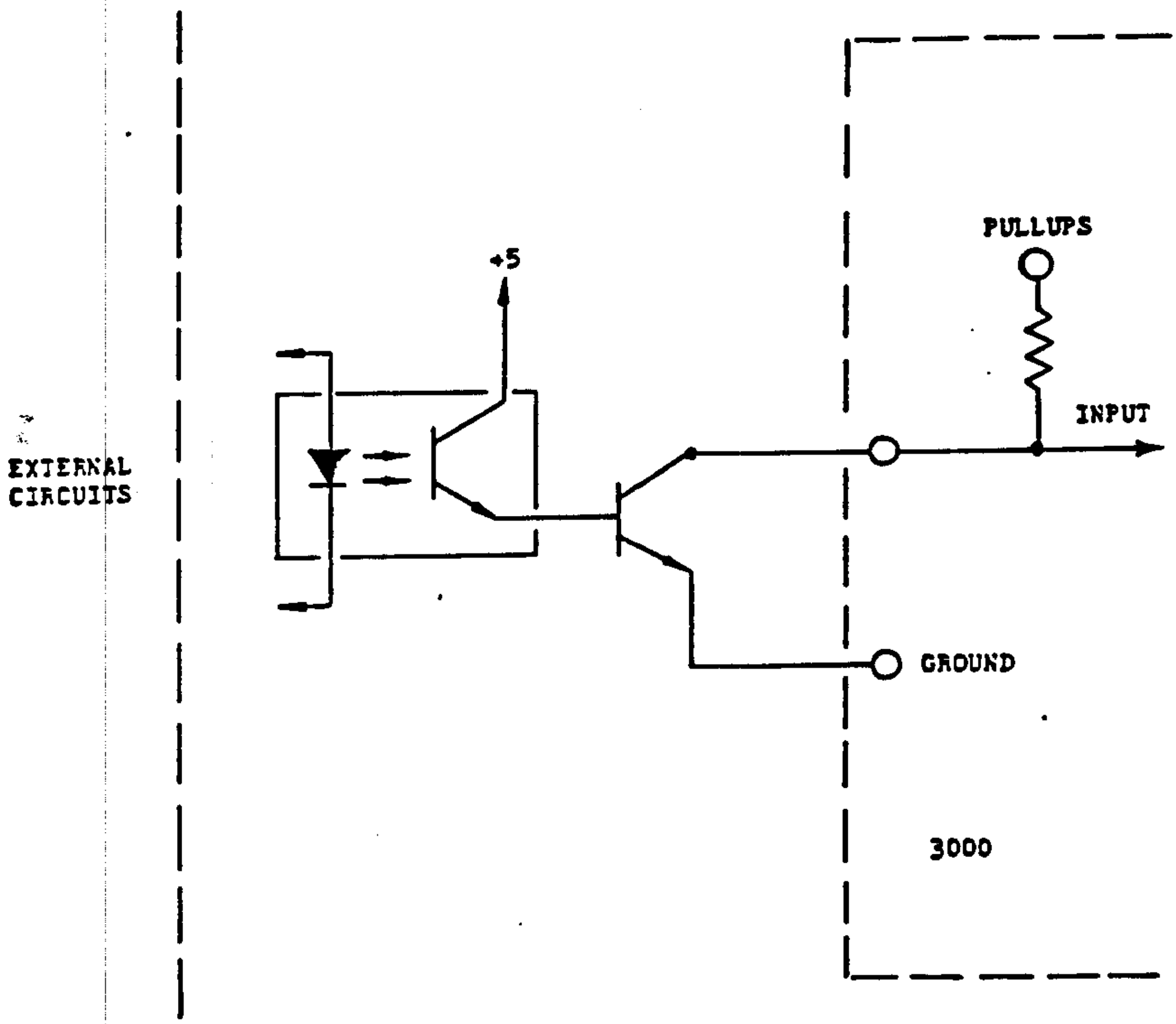


Figure 3-3: Typical Example of Optically Isolated Input



3.6. Individual Axis Connectors

There are four connectors specific to each axis on the rear of the 3000. These connectors are standard, female "D" connectors. There are cutouts on the rear panel for up to four axes. Units with fewer than four axes have cover plates installed over the connector cutouts. These connectors are described in the Sections 3.6.1 through 3.6.4.

The electrical characteristics and functions described use the term LS TTL equivalent load. This means an input at a TTL high level (2.0 volts to 5.5 volts) will source 20 microamps of current, and an input at a TTL low level (0.8 volts to 0 volts) will sink 0.4 milliamps of current. All connections listed as RETURN are connected to DC common.

Note

The Auxiliary, Motor/drive, and Encoder connectors are identical, and can be inserted incorrectly. Be sure you attach each cable to the proper connector.

3.6.1. Auxiliary Connector

The Auxiliary Connector is a 25-pin "D" connector. All outputs are 3.7 volts nominal, 60 milliamp maximum sink or source, and short-circuit protected, unless otherwise noted. Inputs are two LS TTL equivalent loads or less with 3.3 Kohm pullups and 0.1 microfarad noise filter capacitors to ground, unless otherwise noted. Table 3-4 lists the pin assignments for this connector.

3.6.1.1. SHUTDOWN

(Pin 22, Active Low) Activating this input will cause the motor to shut down, which disables the Driver's power amplifier and leaves the motor with only a small amount of holding torque. This input is ignored if the motor is moving.

3.6.1.2. CW LIMIT

(Pin 2, Active High) This input is activated when the axis reaches the extreme mechanical limit stop. Activation of this input causes the Indexer to stop any motor motion in the clockwise direction within 200 microseconds. The signal must remain active for a minimum duration of 75 microseconds.

Table 3-4: Auxiliary Connector Pin Assignments

Pin Number	Signal
1	TRIGGER 6
14	TRIGGER 6 RETURN
2	CW LIMIT
15	CW LIMIT RETURN
3	CCW LIMIT
16	CCW LIMIT RETURN
4	TRIGGER 1
17	TRIGGER 1 RETURN
5	HOME LIMIT
18	TRIGGER 5
6	TRIGGER 2
19	TRIGGER 4
7	PROGRAMMABLE OUTPUT 2
20	PROGRAMMABLE OUTPUT 2 RETURN
8	PROGRAMMABLE OUTPUT 1
21	PROGRAMMABLE OUTPUT 1 RETURN
9	TRIGGER 3
22	SHUTDOWN IN
10	AUXILIARY CONNECTOR +5 VOLTS
23	SHIELD (CASE GROUND)
11	(N.C.)
24	(N.C.)
12	(N.C.)
25	(N.C.)
13	(N.C.)

CW LIMIT may be wired to a normally closed limit switch of suitable contact rating. The polarity (normally closed) is "fail-safe" in that an open connection causes the motor to stop if it is moving clockwise. If no limit switch is connected to this input, bypass jumpers should be installed as shown in Figure 3-1. On-board jumpers on each axis card (JU5 and JU6) are provided and may also be used to bypass this input if no limit switch is connected to the input. These jumpers must be removed when limit switches are used and installed when limit switches are not used.

3.6.1.3. CCW LIMIT

(Pin 3, Active High) This is the same as the CW LIMIT, except that it affects motion in the counterclockwise direction.

Note

The 3000 Indexer uses normally closed limit switches. This means that unless limit switches are connected to the Auxiliary Connector <u>or</u> jumpers are installed in that Connector, your 3000 will not move the motor(s). Two jumpers are required: one should connect pin 2 to pin 15, and the other should connect pin 3 to pin 16 on the Auxiliary Connector (refer to Figure 3-1).

3.6.1.4. PROGRAMMABLE OUTPUT BITS 1 AND 2

(Pin 8 and Pin 7) These two TTL-compatible output bits can be toggled ON and OFF with the OUTSTR instruction in velocity profiling. Refer to Section 4.3.5.16 for details on the operation of these bits.

3.6.1.5. HOME LIMIT

(Pin 5, Active High or Low) This input identifies a HOME position that may be used as a reference point. It is not used to stop motion as are the CW and CCW LIMIT inputs. Exercising the HOME command causes the Indexer to search for the HOME limit switch. In units with the Encoder Interface, this input functions as HOME ENABLE. In this situation the switch should be mounted in such a way that it is activated (closed) only during a single revolution of the Encoder. The Encoder's HOME channel will then be used as the final HOME position.

3.6.1.6. +5 VOLTS

(Pin 10) This 5-volt supply voltage is connected to the 3000's 5-volt power supply through a 1-ohm resistor. It is limited to 250 milliamps. It may be used to power optically isolated limit switches or for any application requiring a power source of this type.

3.6.1.7. GND

(Pins 14, 17, 20, and 21) This is the ground reference for signals on the Auxiliary Connector. It is connected to the 3000's logic (DC) ground.

3.6.1.8. SHIELD

(Pin 23) Connected to the chassis ground. Cables (such as limit switch cables) should be constructed with shielded cable, with the shield connected to this point.

3.6.2. Encoder Connector

The Encoder Connector is a 25-pin "D" connector, which is optional. It is used only in Encoder Interface (-P) versions (utilizing the 1811 card). All outputs are capable of driving twenty LS TTL equivalent loads, unless otherwise noted. Inputs are two LS TTL equivalent loads or less with 3.3 Kohm pullups and a 0.1 microfarad noise filter capacitor to ground, unless otherwise noted. Pin assignments for this connector are summarized in Table 3-5.

Table 3-5: Encoder Connector Pin Assignments

A standard 25-pin male "D" connector (DB25P) is required to connect the feedback encoder to the 2100 "POSITION SENSOR" connector.

Pin	Signal
1	QUADRATURE CHANNEL A+ (+3 V TO +5 V SQUARE WAVE REQUIRED)
2	QUADRATURE CHANNEL A- (OPTIONAL)
3	QUADRATURE CHANNEL B+
4	QUADRATURE CHANNEL B- (OPTIONAL)
5	CHANNEL Z+ (INDEX)
6	CHANNEL Z- (OPTIONAL)
7	(N.C.)
8	SHIELD (CASE GROUND)
9	(N.C.)
10	ENCODER PULSE OUTPUT (X4 PULSE MULTIPLICATION)
11	ENCODER DIRECTION OUTPUT
12	(N.C.)
13	HOME ENABLE
14	SIGNAL GROUND (DC COMMON)
15	SIGNAL GROUND (DC COMMON)
16	SIGNAL GROUND (DC COMMON)
17	SIGNAL GROUND (DC COMMON)
18	SIGNAL GROUND (DC COMMON)
19	SIGNAL GROUND (DC COMMON)
20	SIGNAL GROUND (DC COMMON)
21	(N.C.)
22	(N.C.)
23	+5 VDC (150 MILLIAMPS MAX)
24	+5 VDC
25	+5 VDC

Shell: ITT #DB110963-3

3.6.2.1. CHANNEL A+ and CHANNEL B+

(Pin 1 and Pin 3) These inputs connect to TTL quadrature outputs of an incremental optical encoder. The input impedance is 100 Kohms.

3.6.2.2. CHANNEL A- and CHANNEL B-

(Pin 2 and Pin 4) High impedance (100 Kohm nominal) input optionally connected to a complementary output of a quadrature encoder. Should be left unconnected if this input is not used or not required.

3.6.2.3. CHANNEL Z+

(Pin 5) Identical to CHANNEL A+ (above) except that this input applies to the HOME channel of the encoder signal.

3.6.2.4. CHANNEL Z-

(Pin 6) Connects to complementary output (if present) of encoder's HOME channel.

3.6.2.5. HOME ENABLE

(Pin 13) This signal is used with systems having CHANNEL Z connected to a rotary encoder. This input marks the encoder revolution in which the HOME bit from CHANNEL Z is valid.

This input is shared with the HOME LIMIT input on the Auxiliary Connector and therefore cannot be used if the HOME LIMIT input on the Auxiliary Connector is being used.

3.6.2.6. ENCODER STEP OUT

(Pin 10) This is the STEP output from the quadrature detector (active high). This output can be used to monitor encoder pulses after the quadrature detector has conditioned the quadrature input signals. The encoder circuitry is a times four detector; therefore, four pulses will be seen for every line of the Encoder. Pulses are nominally 229 nanoseconds wide.

3.6.2.7. ENCODER DIRECTION OUT

(Pin 11) DIRECTION output from the quadrature detector. This output can be used to track the direction of the pulses coming out of STEP OUT (above). A jumper on each 1800 board inverts the polarity of this signal.

3.6.2.8. +5 VOLTS

(Pins 23, 24, and 25) This supply voltage is connected to the 3000's +5-volt power supply through a 1-ohm resistor. It is limited to 250 milliamps of current.

3.6.2.9. SHIELD

(Pin 8) Connected to chassis ground. Encoder cable shields should be connected to this point and NOT to D.C. common.

Note

> Many commercial 25-pin "D" connector and cable assemblies are designed to be used for RS-232C data transmission and have their cable shield connected to pin 1. Modify any cables of that type so that the cable shield connects to pin 8.

3.6.3. Motor/Drive Connector

The motor/drive connector is a 25-pin "D" connector. Outputs are 3.7 volts nominal, 60- milliamp maximum sink or source, and short-circuit protected, unless otherwise noted. Inputs are two LS TTL equivalent loads with 3.3 Kohm pullup resistors and 0.1 microfarad noise filter capacitors to ground, unless otherwise noted. The pin assignments for this connector are summarized in Table 3-6. A typical motor/drive-to-Indexing System connection is shown in Figure 3-4.

3.6.3.1. STEP

(Pin 1) This is the step pulse output to the motor/drive (active high). Pulse width is a function of chosen velocity range with minimum width equal to 500 nanoseconds at a maximum repetition rate of 1.09 MHz. Pulse width is always one-half the period of the maximum frequency out for the chosen velocity range. (Velocity ranges are functions of the units selected by the UNITS function. Refer to Section 4.3.3.3.)

Table 3-6: Motor/Drive Connector Pin Assignments

Pin Number	Signal
1	STEP
14	STEP RETURN
2	DIRECTION
15	DIRECTION RETURN
3	CW STEP
16	REMOTE SHUTDOWN
4	CCW STEP
17	REMOTE SHUTDOWN RETURN
5	SHIELD (CASE GROUND)
18	RESOLUTION SELECT
6	RESOLUTION SELECT RETURN
19	FAULT RESET RETURN
7	FAULT RESET
20	DC COMMON
8	MOTOR/DRIVE +5 VOLTS
21	DRIVE FAULT RETURN
9	DRIVE FAULT
22	SLIP FAULT RETURN
10	SLIP FAULT
23	BOOST RETURN
11	BOOST
24	SIGNAL GROUND (DC COMMON)
12	SIGNAL GROUND (DC COMMON)
25	(N.C.)
13	(N.C.)

3.6.3.2. DIRECTION

(Pin 2) This signal is TTL high when motion in the clockwise direction is expected and TTL low when motion in the counterclockwise direction is expected.

3.6.3.3. REMOTE SHUTDOWN

(Pin 16) Compumotor Drivers will turn off current to the Motor's windings when this signal goes active (high). This signal is activated by the SHUTDOWN IN pin on the Auxiliary Connector.

3.6.3.4. CW STEP

(Pin 3) This is the same as the STEP output, except that it is active only for clockwise motion. It is intended for driving non-Compumotor Drivers.

This output is an active-low, open-collector drive. A maximum pullup voltage of 40 volts may be used, and it will sink a maximum of 300 milliamps.

3.6.3.5. CCW STEP

(Pin 4) This signal is the same as the STEP output, except that it is active only for counterclockwise motion. It is intended for driving non-Compumotor Drivers.

This output is an active-low, open-collector drive. A maximum pullup voltage of 40 volts may be used, and it will sink a maximum of 300 milliamps.

3.6.3.6. MOTOR/DRIVE +5 VOLTS

(Pin 8) This supply voltage is connected to the 3000's +5-volt signal through a 1-ohm resistor. It is limited to 250 milliamps of current. This line is reserved for the use of future Compumotor motor/drives and should not be used in normal operation.

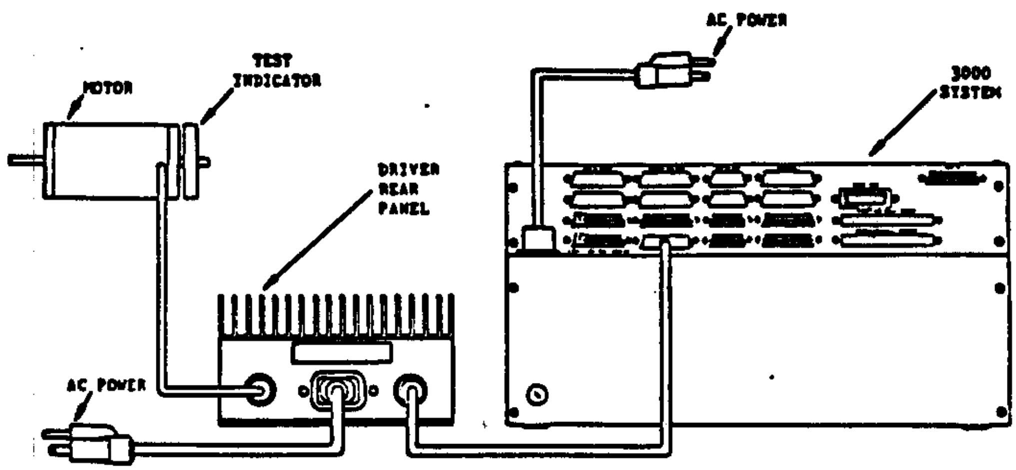
3.6.3.7. BOOST

(Pin 11) Some Compumotor motor/drives provide extra current to the motor when this signal goes high, which results in increasing the torque. This signal is activated by the BOOST command during 3000 program execution.

3.6.3.8. SHIELD

(Pin 5) This pin is connected to the chassis ground. Compumotor 3000-to-motor/drive cables have the cable shield connected to this point.

Figure 3-4: Motor/Drive-to-Indexing System Connection



3.6.4. Joystick Connector

A 15-pin "D" connector is used for convenient connection to the Compumotor two-axis Joystick, Model 852. All outputs are 3.7 volts nominal, 60 milliamp maximum sink or source, and short-circuit protected, unless otherwise noted. All inputs are two LS TTL equivalent loads or less with 3.3 Kohm pullups and 0.2 microfarad bypass capacitors to ground, unless otherwise noted. Table 3-7 lists the pin assignments for this connector.

Table 3-7: Joystick Connector Pin Assignments

Number	Signal
1	INDEXER REMOTE ENABLE IN
2	INDEXER DIRECTION IN
3	INDEXER AT ZERO IN
4	INDEXER STEP PULSE IN
5	(N.C.)
6	INDEXER REMOTE ACKNOWLEDGE OUT
7	INDEXER SHUTDOWN IN
8	SHIELD (CASE GROUND)
9	SIGNAL GROUND (DC COMMON)
10	SIGNAL GROUND (DC COMMON)
11	INDEXER +5VDC
12	(N.C.)
13	(N.C.)
14	CCW LIMIT OUT
15	CW LIMIT OUT

3.6.4.1. REMOTE ENABLE

(Input, Active Low) This signal is used by the Joystick to request control of the motor and is part of a handshake procedure between a Model 852 Joystick and a Model 3000 Indexer.

3.6.4.2. AT ZERO

(Input, Active High) When this signal is active, it tells the 3000 that there are no motor pulses coming in the STEP IN line, so the Joystick is set to zero velocity. This signal is required to warn the 3000 of possible directional changes. This signal must be active for at least 4 milliseconds before a direction change occurs (at DIRECTION IN) and for a least 4 milliseconds after a direction change occurs.

3.6.4.3. SHUTDOWN IN

(Input, Active Low) This input causes the REMOTE SHUTDOWN signal on the Motor/Drive Connector to become active. A command to the 3000 and the SHUTDOWN IN on the Auxiliary Connector are two other ways that cause REMOTE SHUTDOWN to go active. This input is physically shared with the SHUTDOWN IN signal on the Auxiliary Connector. Therefore, if a Joystick is connected to the 3000, the SHUTDOWN IN signal on the Auxiliary Connector cannot be used and, vice-verse, if the SHUTDOWN IN signal on the Auxiliary Connector is being used, a Joystick cannot be connected.

3.6.4.4. STEP IN

(Input, Active High) This is the input pulse train from a Joystick. The pulse width is limited to 250 nanoseconds and a repetition rate of 2 MHz. The pulses are routed directly to the STEP output on the Motor/Driver Connector when the Joystick has been enabled by the appropriate command to the 3000. Therefore, the STEP output on the Motor/Driver Connector shows the status of the input that enabled the Joystick.

There is no 0.1 microfarad filter capacitor to ground on this input.

3.6.4.5. DIRECTION IN

(Input) This is the direction signal from the Joystick. When this input is high, it is interpreted as a request for motion in the CW direction. When this input is low, it is interpreted as a request for motion in the CCW direction. It is assumed that this signal will not change unless it is both preceeded and followed by an active AT ZERO signal. The AT ZERO signal must be active for a least 4 milliseconds before and 4 milliseconds after the DIRECTION IN signal changes.

There is no 0.1 microfarad filter capacitor to ground on this input.

3.6.4.6. JOYSTICK +5 VOLTS

(Input) This supply voltage is connected to the 3000 +5 volt supply via a 1-ohm resistor and is limited to 250 milliamps of current. The Model 852 Joystick uses this line to power opto-isolators.

3.6.4.7. REMOTE ACKNOWLEDGE

(Output, Active High) This signal is the other half of the REMOTE ENABLE signal for the handshake between a Model 852 Joystick and a Model 3000 Indexer. When the 3000 activates this signal, it gives control of the Motor/Drive to the Joystick. The 3000 keeps track of pulses to the Motor/Drive and continues to monitor the CW and CCW limit switch inputs. (Striking a limit switch will not disable the Joystick.) Disabling the Joystick from the 3000 causes this signal to go inactive.

3.6.4.8. SHIELD

It provides a shield ground point for external cables coming into a "D" bulkhead connector without splitting that wire out of the flat ribbon cable.

3.7. Limit Switches

The limit switches are connected to the Auxiliary Connector between pins 2 and 15 (CW LIMIT) or pins 3 and 16 (CCW LIMIT).

If initial systems tests are to be performed with the motor connected to a limited-travel mechanism, it is essential that limit switches be installed to protect the mechanism. The motor itself can be stalled indefinitely without damage.

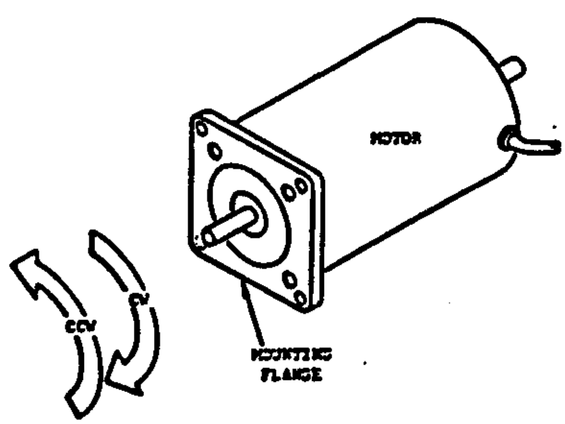
Typical switches include mechanical contact closures, opto-interrupters, or Hall-effect devices. The easiest to mount and wire are typically the mechanical-type switches. A normally closed contact is required for Indexer operation (high true).

Disconnect AC power to the motor/drive and rotate the Motor by hand to determine in which direction the mechanism moves for clockwise rotation of the motor shaft. The sense of clockwise and counterclockwise motor rotation is determined as if the observer were looking at the Motor from the mounting flange end (see Figure 3-5).

Note

Install the limit switches so that there is room to stop. Wire the COMMON (C) terminal to pin 15 or 16 (RTN) and the normally closed (NC) terminal to pin 2 (CW Limit) or pin 3 (CCW LIMIT). Otherwise, the motor will not move.

Figure 3-5: Direction of Rotation Determination



3.8. Encoder Interface

3.8.1. Encoder Interface Functions

When equipped with Model 1811 Indexers, the 3000 includes the circuitry necessary to interface with an incremental optical encoder. Dual-channel incremental encoders with quadrature and with single-ended or differential-TTL outputs may be used. The encoder may be used as part of a closed-loop system or as an independent means to verify position. The encoder may also be used to detect whether a stall has occurred during a move. The functions that are added when the Encoder Interface is used are:

a. Encoder referenced positioning
b. Encoder position servoing
c. Motor stall detection
d. Position loss detection
e. Multiple axis move termination

Implementation of the encoder functions listed above requires connection of the encoder and setup of the appropriate parameters via the programming inputs. The other functions listed may be implemented without an encoder.

3.8.2. Encoder Signals

The Indexer will supply power (up to 1 amp) to drive an encoder. Encoder requirements are summarized in Table 3-8. A typical encoder input circuit is shown in Figure 3-6.

If a reducer is introduced between the motor shaft and the Encoder, the number of encoder steps must be divided by the equivalent gear ratio to define the number of encoder steps received by the Indexer.

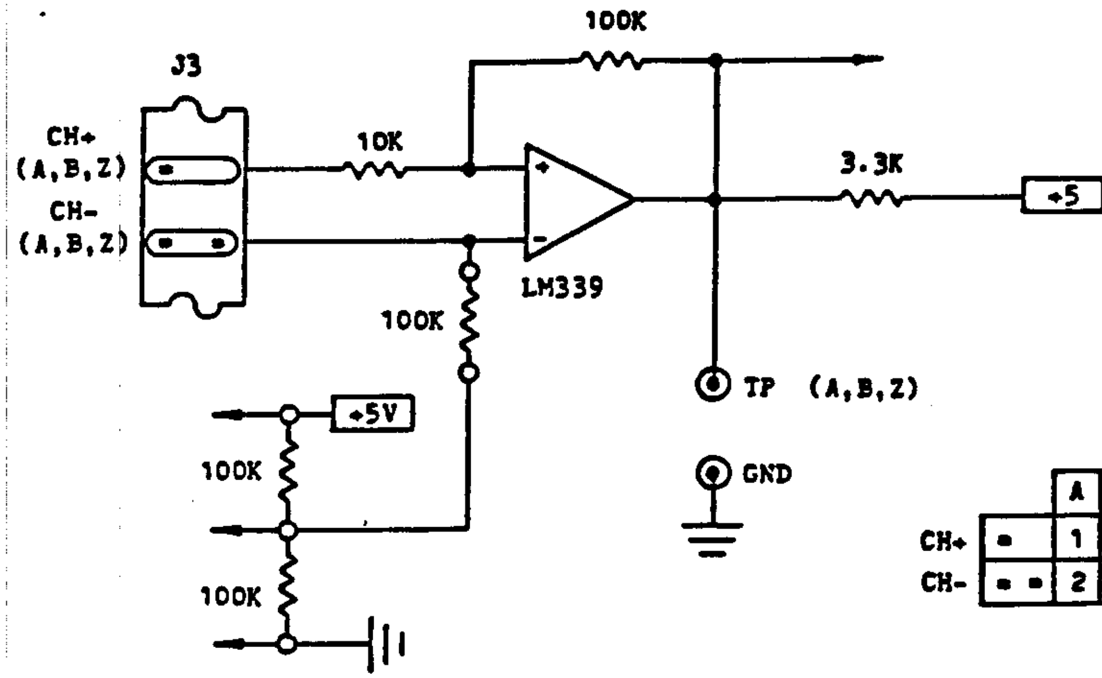
For example, using a 25,000-step motor, a 1000-line Encoder, and a 10:1 reducer like the setup illustrated in Figure 3-7, the ratio of required motor steps to encoder steps would be changed as shown below:

Parameter	Encoder Direct Mounted to Motor	10:1 Reducer Between Motor and Encoder
Number encoder steps seen by Indexer per revolution	4,000	400
Required ratio motor steps to encoder steps	25,000/4000	25,000/400

Table 3-9: Encoder Signal Characteristics

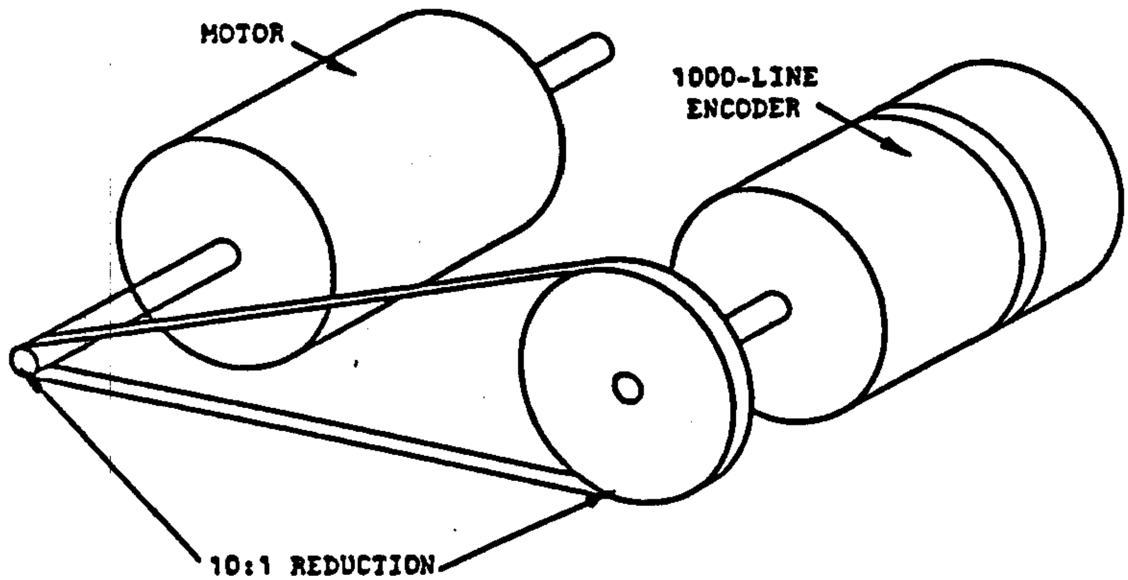
Signal	Characteristic
Turn-On Voltage	2.7 volts with respect to signal ground
Turn-Off Voltage	2.3 volts with respect to signal ground
Input Impedance	113 Kohms
Maximum Frequency	100 KHz
Maximum Input Voltage	5 VDC
Power Drain	150 mA

Figure 3-6: Typical Encoder Input Circuit



		A	B	Z
CH+	▪	1	3	5
CH-	▪ ▪	2	4	6

Figure 3-7: Encoder with Reduction



3.8.3. Motor to Encoder Steps Ratio

The number of encoder steps seen by the Indexer is equal to four times the number of encoder "lines." For example, a 1000-line Encoder mounted directly on the Motor will generate 4000 encoder steps for each revolution of the motor shaft. A minimum of two motor steps per encoder step is required for successful operation of the Position Maintenance function. In the example above, a motor resolution of better than 8000 steps/rev would be required.

The default values are for a 25,000-step Motor and a 1,000-line Encoder. The ratio of motor to encoder steps is controlled with the ENCODER Command, see Section 4.3.3.8.

Note

> For best results, the number of motor steps per revolution, divided by four, should always be greater than the number of encoder steps per revolution (after reduction).

3.8.4. Encoder-Referenced Positioning

The standard 3000 normally operates using position parameters in units of motor steps. When the Encoder Interface is used, the parameters will be in units of encoder steps. In this mode, when making a preset move, the Indexer sends pulses to the motor until the correct number of encoder steps is returned.

The power-on default setting is referenced to motor steps. Encoder referenced positioning can be used via the AXES command, see Section 4.3.3.5.

3.9. Operational Test

If you need to check your indexing system to be sure it is operating properly, perform the operational test described in the sections that follow.

3.9.1. Test Setup

a. Mount an indicator with a distinctive visible mark on the motor shaft of the axis or axes to be tested.

b. Position the motor(s) so that you can see the mark on the indicator(s) and so that the motor(s) can rotate without interference.

c. Connect the motor/drive(s) to the 3000 as described in Section 3.6.3.

d. Connect AC input power to the 3000 as described in Section 3.4.

3.9.2. Test

a. Apply AC power to the 3000 by pressing the Front Panel ON switch. The POWER-UP Menu will be displayed.

b. Press the ACCESS function key in the POWER-UP Menu.

c. Enter the access code for your system ("3000" if your system access code has not been changed from the factory-set value). The MAIN Menu will be displayed.

d. Press the EDIT function key. The first line of the program and the first page of the EDIT Menu will be displayed.

e. Enter the program listed in Appendix B, Problem 3 (or any other proven program).

f. Press the MENU key. This will return you the MAIN Menu.

g. Press the START key. The program should execute as described in Problem 3 (or as previously experienced, if you used your own program). If the motor(s) does not operate as intended, refer to Section 3.10 for troubleshooting procedures.

3.10. Troubleshooting

If your indexing system does not function properly, perform the following checks:

a. Make sure there is a program in Main Memory. The 3000 will not operate unless an operational program is in Main Memory. This can be done by using the EDIT Menu to enter a program, by loading a program from a remote source via the IEEE-488 or RS-232C interface, or by using the RECALL function (Section 4.3.2.4) to recall an existing program from non-volatile memory. If a data error is detected during this operation, an error message will be displayed and you have the choice of recalling the program by pressing ENTER or aborting the recall by pressing STOP.

b. Step through the program to make sure that there are no programming errors.

c. Check all rear-panel connector cables to make sure they are properly connected at both ends.

d. Check to make sure an 1811 Indexer is installed and properly connected for each axis you wish to control. Be sure that each motor/drive cable is connected to the rear-panel connector that corresponds to the 1810 or 1811 that controls that axis.

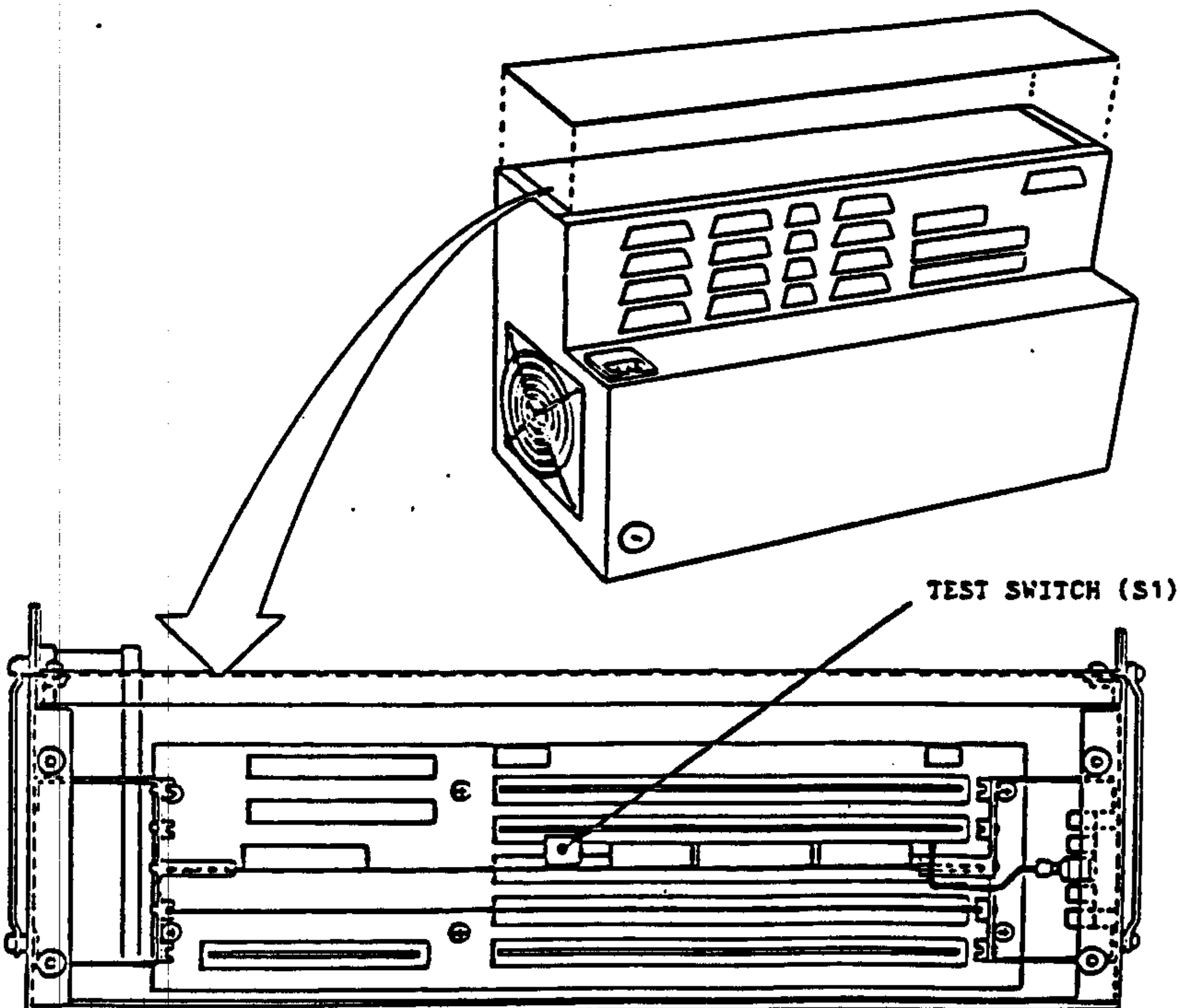
e. Make sure that all limit switches connected to the Auxiliary connectors are closed. If there are no limit switches, be sure jumpers are installed on the limit switch connector pins (Figure 3-1). A motor will not move unless both limit switches (CW and CCW) for the axis are closed or the inputs are shorted to ground by jumpers.

f. Make sure that the motor for the axis you wish to test is free to rotate in both directions, then remove the top cover from the 3000 and press the TEST switch (S1) on the 1811 Indexer that corresponds to that axis (see Figure 3-8). The motor should rotate clockwise one revolution, then counterclockwise one revolution. If this test executes properly, your Indexer and motor/drive combination are working properly, and the problem is in your program or in the 3000 system. If this test does not execute as described, perform step g.

g. Carefully remove the 1811 for the faulty axis and inspect to make sure that all jumpers are correctly installed (refer to Figure C-1).

h. If you have performed all of the above checks and your 3000 still does not function properly, contact your Compumotor distributor or further assistance as described in Section 1.6.

Figure 3-8: Test Switch (S1) on the 1811 Indexer



4. PROGRAMMING AND OPERATION

4.1. Overview

4.1.1. General

The 3000 can be programmed and operated by using the Front Panel controls and indicators to enter commands or via a remote-control interface (RS-232C or IEEE-488) from a terminal or computer.

The 3000 commands are functionally organized into a set of menus. Each menu permits you to select a function or command by pressing the appropriate function key or sending the proper command over the remote- control interface. Menus that have more than six commands have more than one page. The next page of a menu is selected by pressing the ETC key, usually function key F6. There are four primary menus as shown in Figure 4-1. The menus are described in detail in Section 4.3. Appendix A lists each command, both alphabetically and by menu, and briefly describes its function.

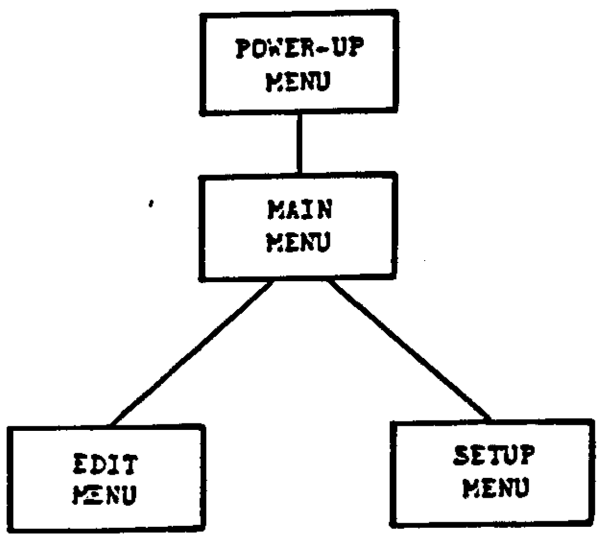


Figure 4-1: Model 3000 Primary Menus

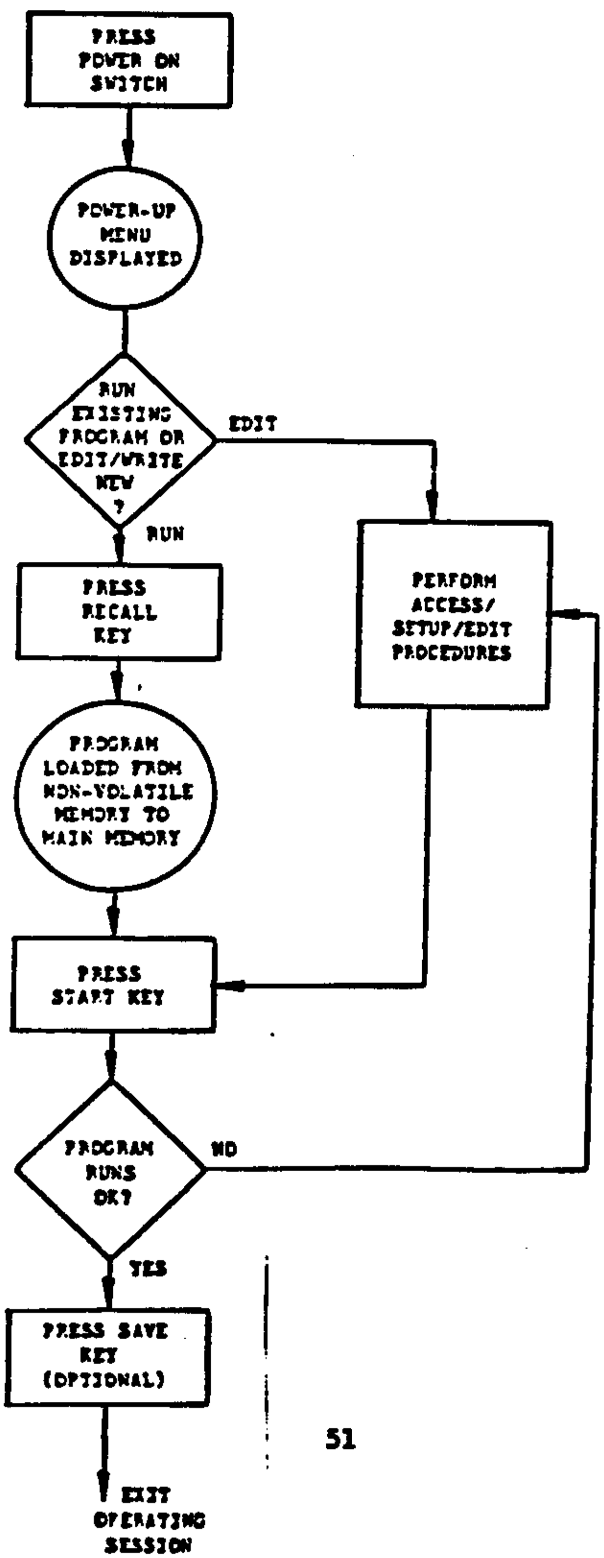
4.1.2. Operational Choices

You can elect to operate the 3000 from the Front Panel or from a remote terminal or computer. Operation from the Front Panel is typically more convenient, but operation from a computer or terminal may allow you to use larger and more complex programs.

You can recall a program that is stored in the 3000's non-volatile, or permanent, memory (NVM) and run that program without using any of the 3000's programming capabilities. A typical operating session using a stored program is shown in Figure 4-2. All operations required to run an existing program can be performed using the commands available in the POWER-UP Menu (no access code required).

When you turn on the 3000, the POWER-UP Menu will be displayed. At this point, you can choose to simply RECALL an existing program and run that program, or perform the ACCESS procedure described in this section to enter into EDIT session. If you are going to run an existing program, you need only RECALL the program into Main Memory, then RUN the program. A more detailed description of the use of the function keys required to perform these operations is in Section 4.3.1.

Figure 4-2: Typical Operation Using Stored Programs



If you elect to modify an existing program, change the setup of the 3000 configuration, or prepare new programs, you need to perform the procedures outlined in Figure 4-3.

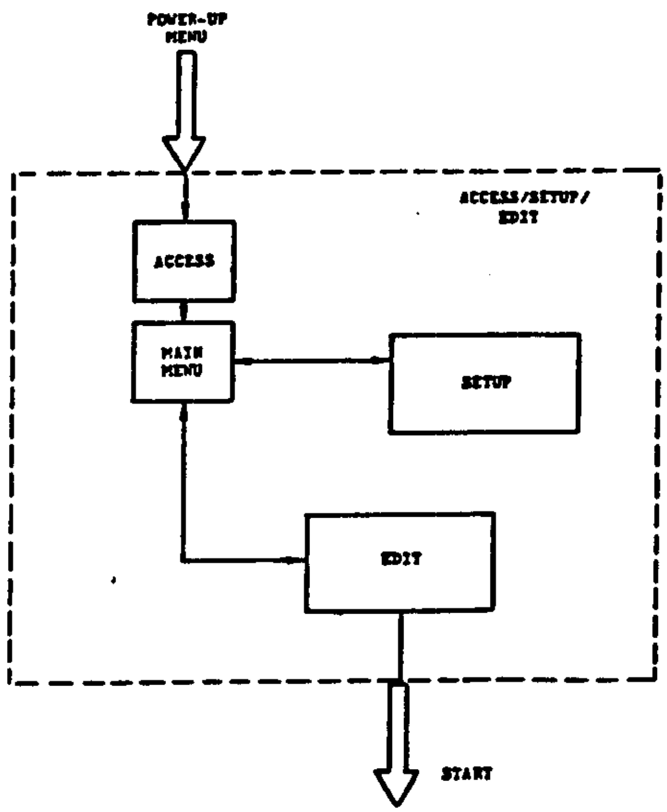


Figure 4-3: Basic Access/Setup/Edit Functions

The ACCESS procedure must be performed before you can edit a program or change the 3000's SETUP parameters. The major elements of the ACCESS procedure are shown in Figure 4-4. When you press the ACCESS key in the POWER-UP Menu (see Section 4.3.1.1), the system will ask for the access code. When you enter the access code (the four-digit number "3000" is the factory-set access code), the MAIN Menu will be displayed. The MAIN Menu (see Section 4.3.2) will allow you, among other things, to perform the EDIT and SETUP procedures. Instructions on how to change the access code are provided in Section 4.3.3.4.

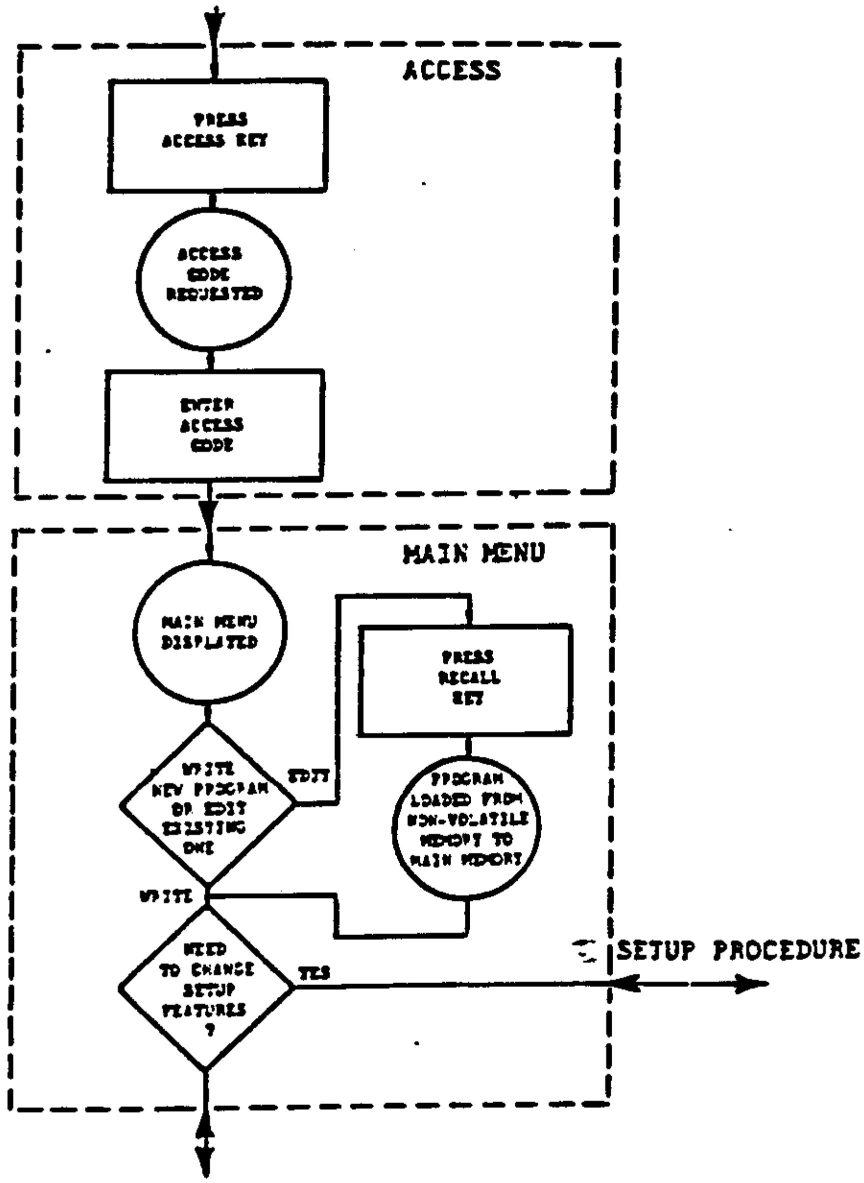


Figure 4-4: ACCESS Procedure

After you have performed the ACCESS procedure, you will be able to enter the EDIT function. The basic components of an editing session are shown in Figure 4-5 and are described in detail in Section 4.1.3. After performing the EDIT procedures, press the MENU key to return you to the MAIN Menu.

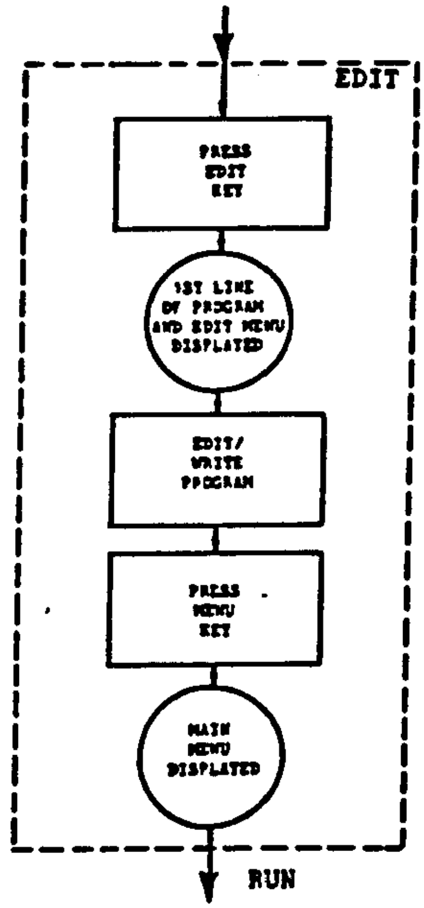


Figure 4-5: EDIT Procedure

Performing the ACCESS procedure also allows you to enter the SETUP function. The SETUP function allows you to define the basic parameters of your system's operation (acceleration, velocity, etc.). The basic components of the SETUP procedure are shown in Figure 4-6 and are described in detail in Section 4.3.3. After performing the SETUP procedures, press the MENU key to return the MAIN Menu.

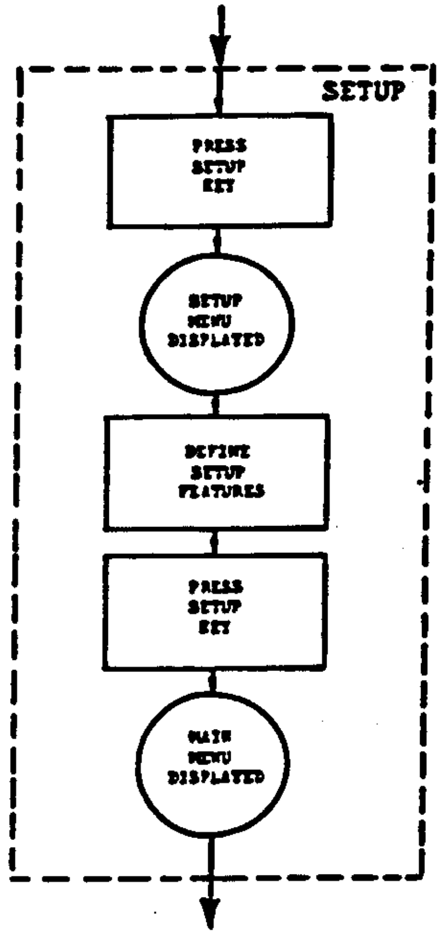


Figure 4-6: SETUP Procedure

4.1.3. Introduction to Programming

A program is a complete set of instructions (commands) that you enter into the 3000 to cause it to move the axes that it controls in the manner in which you want them moved. Instructions are contained in a series, or sequence, of statements, each with its own line number. The 3000 begins program execution at the first line of a program (the line with the lowest numerical value for a line number) and proceeds from one instruction to the next, doing what each one says to do, until it encounters a DONE statement. It then stops and waits for a restart or some kind of new operating session. The liquid-crystal display on the Front Panel, referred to as "the display" in this manual, displays one line of the program at a time on the upper lines of the display, as shown in Figure 4-6. The lower line of the display indicates the function presently assigned to each of the six variable function keys.

In the example shown in Figure 4-7, the upper line of the Front Panel Display shows line 0040 of the program. By using either the LIST command (Section 4.3.12.1) or the cursor keys, you can scroll the program through the display to show any line desired. In this example pressing the down cursor key would display line 0050 of the program. When you enter an instruction on a program line and press the ENTER function key, the next line of the program will be displayed. The next line number in series (in increments of 0010) will be assigned automatically to this next line if it is a new line.

4.1.3.1. Analyzing Your Problem

When analyzing your problem, you must first state its nature. Then you must define the data that you will be able to provide the 3000 (steps, velocity, acceleration, etc.) and define the actions you expect the 3000 to perform (refer also to Chapter 2). Finally, you must convert this information into an algorithm, the computations and logical processes the 3000 must perform and the motion it must initiate to convert the data you provide into the actions you desire.

4.1.3.2. Preparing a Flowchart

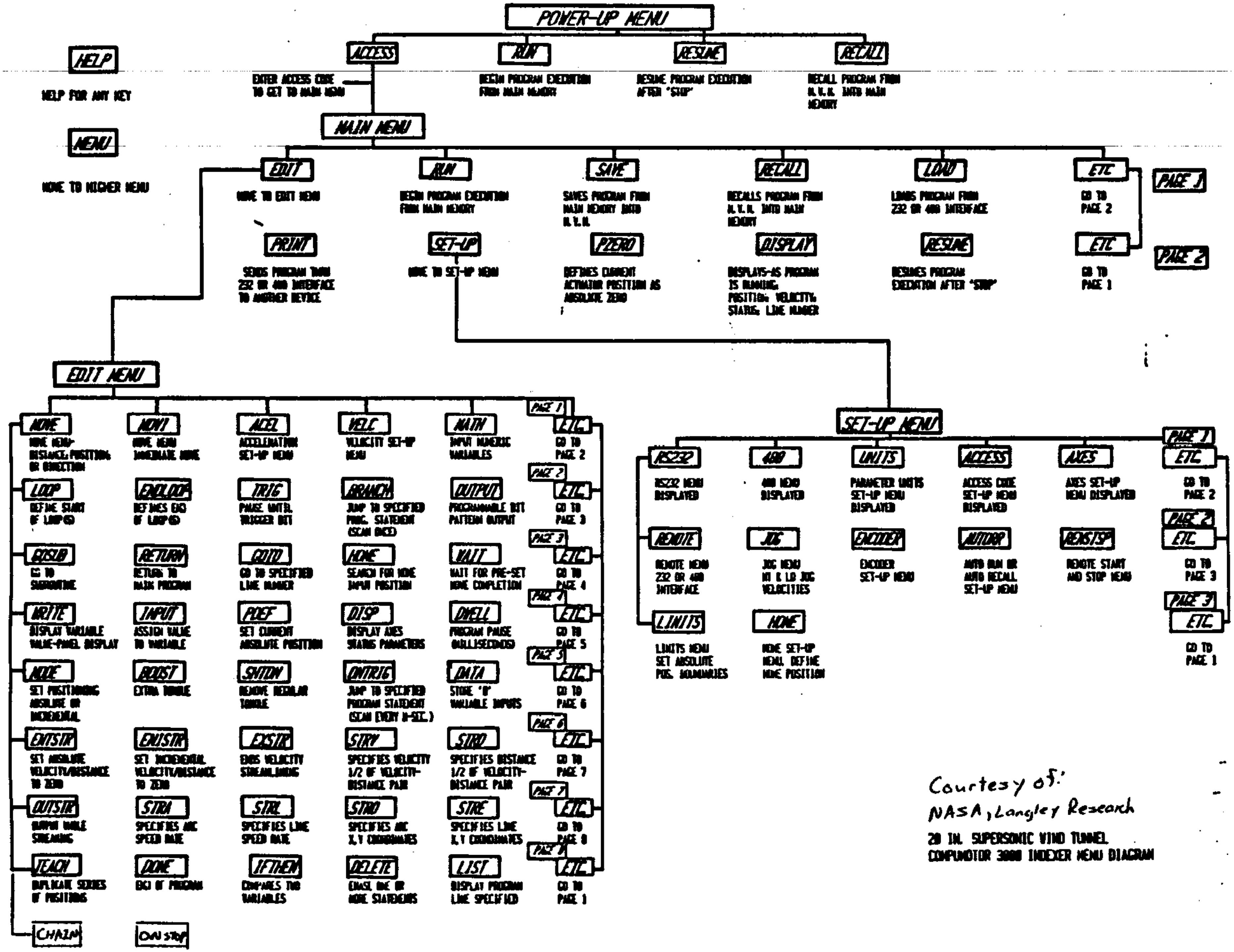
A flowchart (see Figures 4-2 through 4-5) is a drawing that portrays

a. the means for providing data the the 3000.

b. the required actions to be taken by the 3000.

c. the logical and arithmetic steps required to achieve the proper result.

A flowchart has two primary uses: to help you write your program by serving as a "blueprint" and to document the logic of the program for future use.



Courtesy of:
NASA, Langley Research

20 IN. SUPERSONIC WIND TUNNEL
COMPUMOTOR 3000 INDEXER MENU DIAGRAM

Flowcharts use specific symbols to represent different activities, and written messages within each symbol explain the activity associated with that symbol. Table 4-1 shows standard symbols recommended for use when designing programs.

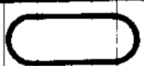
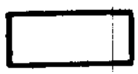
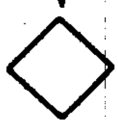
4.1.3.3. Entering the Program

Analyze your problem and prepare a flowchart to resolve it. Use your flowchart as a guide. Refer to the command descriptions in this chapter to determine which command(s) to use and how to use them properly in your program. Write your program on a sheet of paper; then, when you are reasonably sure the program is correct, enter the program into the 3000.

4.1.3.4. Editing the Program

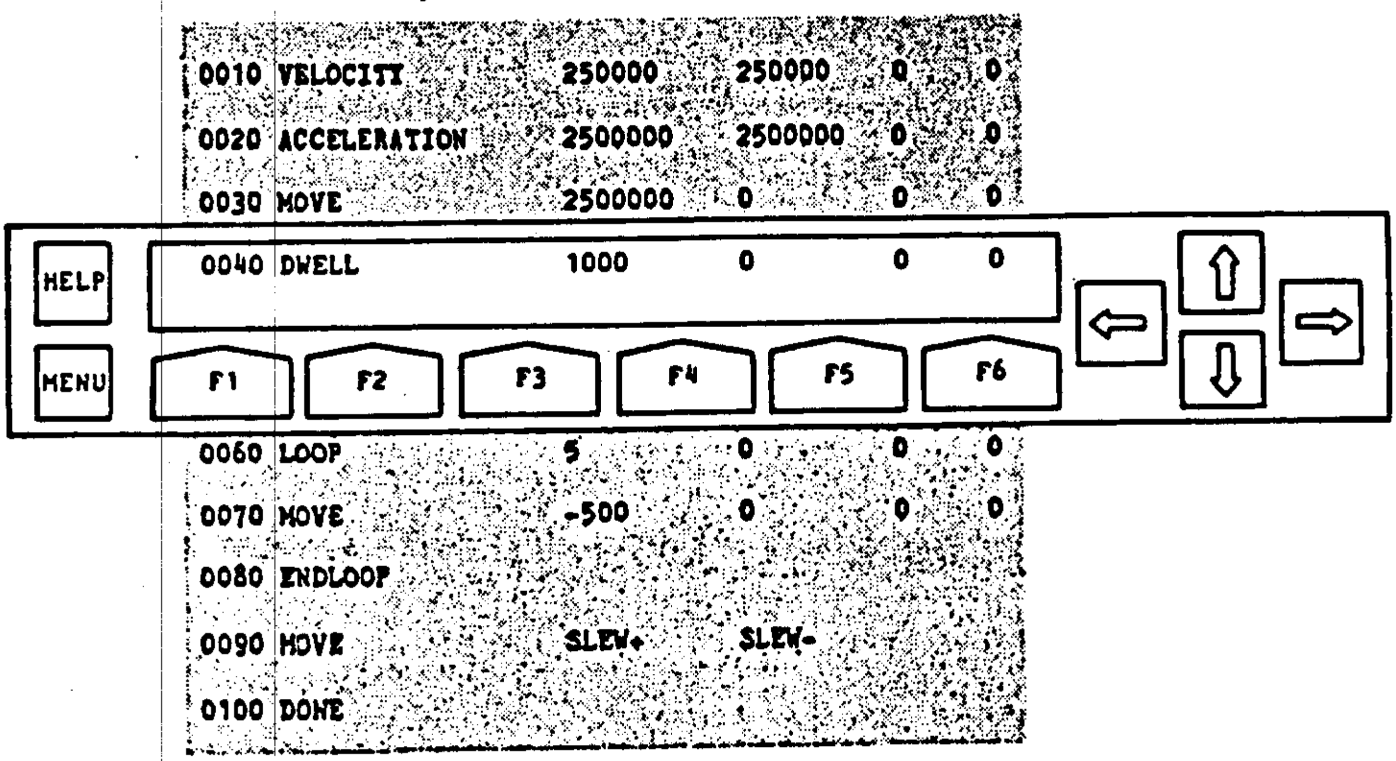
There is a good chance that the program you write will not execute perfectly the first time you try to run it. When this happens, compare the actual execution of the program with the desired execution as presented in your flowchart. Determine which steps or instructions are not working as intended, then use either the cursor keys or the LIST command to step through the program to the "trouble" steps and modify them so that the program will run as intended. Repeat this procedure until all steps are correct and the program executes as intended.

Table 4-1: Flowchart Symbols

Symbol	Name	Meaning
	Terminal	Indicates start/end of program
	Input/Output	Indicates when an input or output operation is to be performed.
	Process	Indicates calculations or data manipulation.
	Flowline	Indicates flow of logic.
	Decision	Represents a decision point or question that requires a choice of which logical path to follow.
	Preparation	Indicates a preparation statement, as in the LOOP command.

To change the line number of any line in your program, step through the program using either the up or down cursor keys or use the LIST to display the desired line, position the cursor over the current line number, enter the new line number on the keypad, then press ENTER. To enter a new line of program (for example, to enter program line 0035 between lines 0030 and 0040), position the cursor at any point in the program, manually enter the program line (including the new line number) on the keypad, enter the appropriate command for that line, then press ENTER. This new line will be positioned in its proper location within the program (between lines 0030 and 0040) and the line you "typed over" when entering the new line will remain unchanged.

Figure 4-7: Program Line Display



4.1.3.4. Editing the Program

There is a good chance that the program you write will not execute perfectly the first time you try to run it. When this happens, compare the actual execution of the program with the desired execution as presented in your flowchart. Determine which steps or instructions are not working as intended, then use either the cursor keys or the LIST command to step through the program to the "trouble" steps and modify them so that the program will run as intended. Repeat this procedure until all steps are correct and the program executes as intended.

Table 4-1: Flowchart Symbols

Symbol	Name	Meaning
	Terminal	Indicates start/end of program
	Input/Output	Indicates when an input or output operation is to be performed.
	Process	Indicates calculations or data manipulation.
	Flowline	Indicates flow of logic.
	Decision	Represents a decision point or question that requires a choice of which logical path to follow.
	Preparation	Indicates a preparation statement, as in the LOOP command.

To change the line number of any line in your program, step through the program using either the up or down cursor keys or use the LIST to display the desired line, position the cursor over the current line number, enter the new line number on the keypad, then press ENTER. To enter a new line of program (for example, to enter program line 0035 between lines 0030 and 0040), position the cursor at any point in the program, manually enter the program line (including the new line number) on the keypad, enter the appropriate command for that line, then press ENTER. This new line will be positioned in its proper location within the program (between lines 0030 and 0040) and the line you "typed over" when entering the new line will remain unchanged.

4.2. Front Panel Operation

All operator controls and indicators for the 3000 are located on the Front Panel. These controls, indicators, and their functions are listed in Table 4-2 and are shown in Figure 4-8.

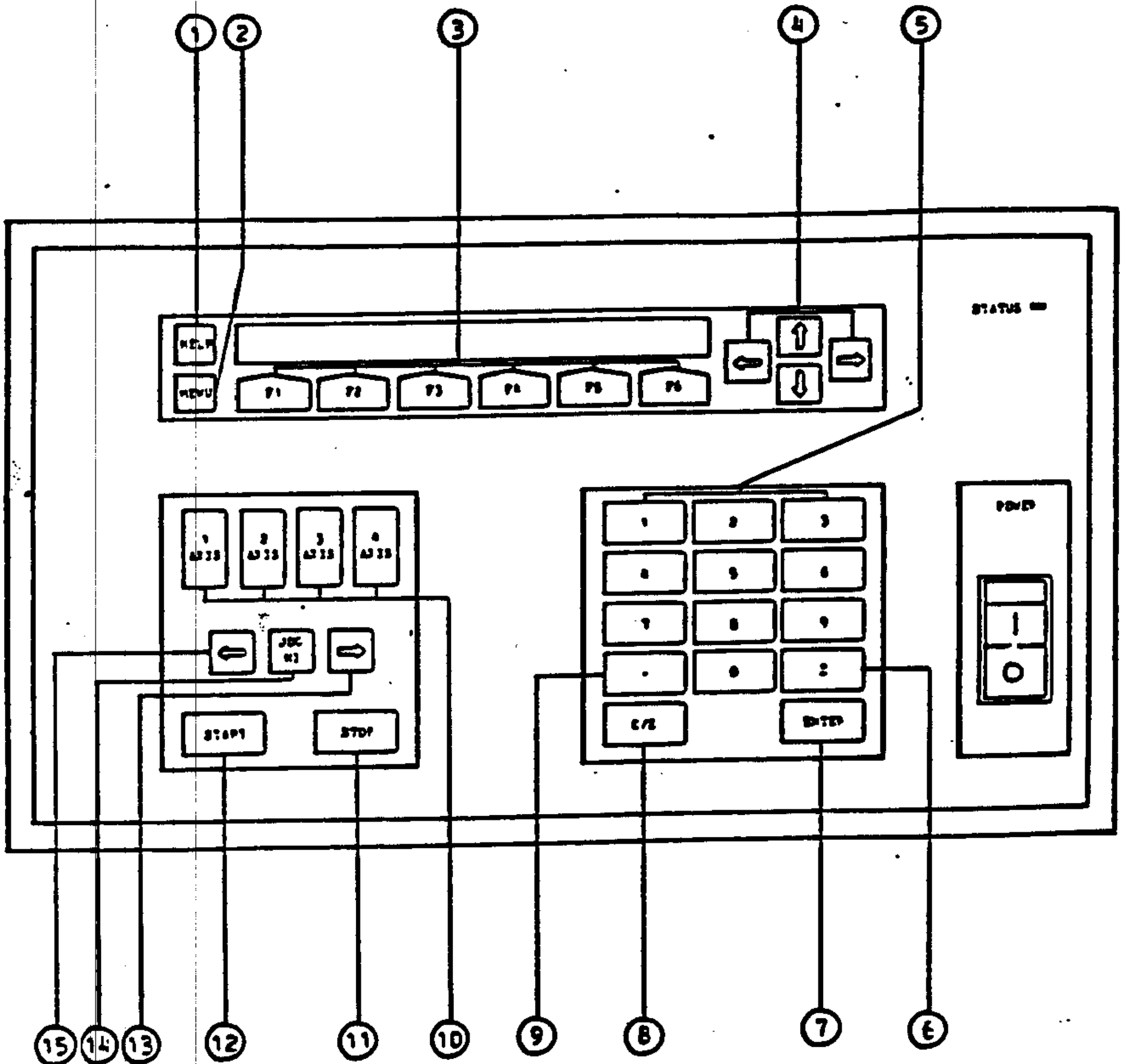
Table 4-2: 3000 Indexer Front Panel Controls and Indicators

No.	Key	Description
1	HELP	This key can be pressed at any time during the programming operation, followed by another key to otain a description of that key. The help message is displayed on the top line of the display one line at a time. Press the down arrow cursor control key to display the next line of the message. Hold the DOWN cursor key down to cause the message to be scrolled through at high speed.
2	MENU	The MENU key returns you to the previous Menu level.
3	F1-F6	Six function keys are located directly beneath the bottom line of the display. The function of each key is labeled directly above the key on the display. As the programming menus change, the labels on the keys change. This feature of the 3000 permits its simple-to-understand Front Control Panel to perform many tasks. Pressing any of the six keys invokes the function labeled by that key. The sixth key (F6) is usually labeled ETC and is used to bring up the next page of the menu onto the screen
4	CURSOR	There are four cursor control keys to the right of the display. These are used during programming and data entry to move the cursor left and right on the current line. The up and down keys are used during programming to scroll through the program. The down cursor key displays the next program statement and the up cursor key causes the previous program line to be displayed.
5	0-9	These keys enter the digits 0-9 on the display at the cursor location.
6	±	This key places a minus sign at the current cursor location. If a minus sign already exists, press this key to change it into a plus.

Table 4-2: 3000 Indexer Front Panel Controls and Indicators (Cont'd)

No.	Key	Description
7	ENTER	The ENTER key is used when you are entering programs into the 3000 in EDIT mode or when entering commands at the MAIN Menu level. Each statement that is put on the display must be terminated with the ENTER key in order for the 3000 to store it in program memory.
8	C/E	This key places a space at the current cursor location.
9	.	This key places a decimal point at the current cursor location.
10	AXIS 1-4	These keys serve several functions. When you want to JOG an axis, you must press the AXIS key corresponding to the axis you wish to JOG. When the operator is entering a program, the AXIS key will tab the cursor to the right column for entering data for the MOVE, ACEL, and VEL commands. These keys are also used with the TEACH command to select an axis. When the 3000 is executing a program, the display on the AXIS key lights whenever the corresponding axis is in motion.
11	STOP	This key causes the motors on all four axes to decelerate to a stop and will stop program execution.
12	START	This key starts executing the program in the 3000's Main Memory. Program execution begins with the lowest numbered program line.
13	-->	This is the JOG CLOCKWISE key. If the 3000 is not running a program, pressing this causes the selected Axis (1-4) to move clockwise.
14	JOG HI	A jog normally takes place at 2500 steps/second (0.1 rev/sec on a 25-Kbyte motor). When JOG HI is held down followed by the activation of one of the other JOG keys, a jog takes place at 25000 steps/sec (1.0 rev/sec on a 25-Kbyte motor). The JOG and JOG HI speed may be changed with the JOG SPEED SETUP.
15	<--	This is the JOG COUNTERCLOCKWISE key. If the 3000 is not running a program, pressing this causes the selected Axis (1-4) to the counterclockwise. The axis to be Jogged is selected with one of the AXIS keys.

Figure 4-8: 3000 Indexer Front Panel Controls and Indicators



4.3. Detailed Description of Model 3000 Menus

The four primary menus were shown in Figure 4-1. There are additional submenus that are accepted via the primary menus. When you require a function that is not available on the menu currently being displayed, pressing the MENU key to the left of the display will move you "up" to the next level. For example, if you are using the MAIN Menu, pressing the MENU key will cause the POWER-UP Menu to be displayed, or pressing the MENU key while you are in the EDIT Menu will take you up to the MAIN Menu. The MENU key, then, is used to move up in menu level. Moving down in menu level is accomplished by selecting the desired menu while you are in the next higher menu. For example, pressing the SETUP function key while in the MAIN Menu moves you down one menu level to the SETUP Menu.

4.3.1. POWER-UP Menu

When the 3000 is turned ON, or if the MENU key is pressed when the MAIN Menu is displayed, the POWER-UP Menu (Figure 4-9) will be displayed. The insert of Figure 4-9 shows the relationship of this menu to the other three menus.

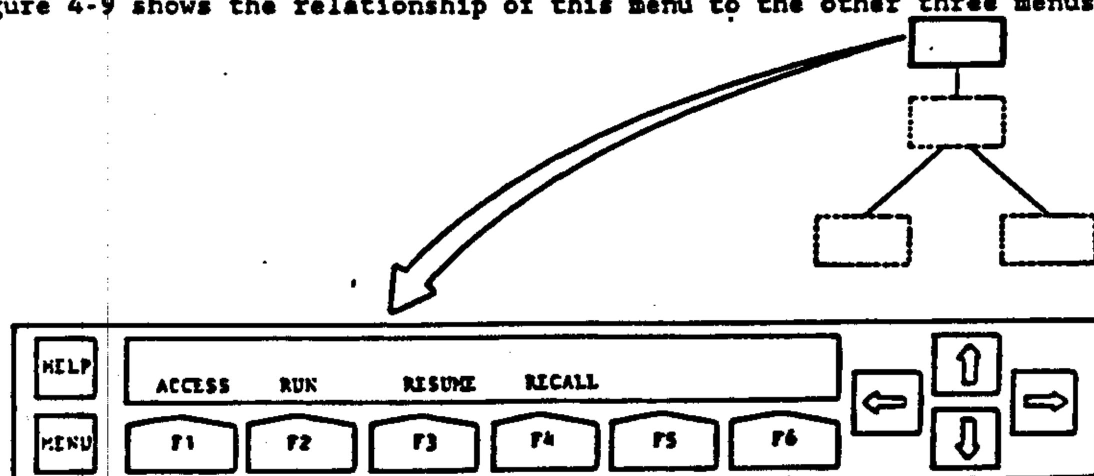


Figure 4-9: POWER-UP Menu

4.3.1.1. ACCESS Command

(F1) Pressing the ACCESS function key causes the 3000 to ask for the system password, or access code. You must enter this password before access to the SETUP or EDIT Menu is permitted. Each unit shipped from the factory has the four-digit number "3000" as its access code. You can change this password to any other number with the ACCESS command in the SETUP Menu.

4.3.1.2. RUN Command

(F2) The RUN command is used to begin running a program in Main Memory. Pressing the RUN key will cause the line RUN FROM 0000 to be displayed. Press the ENTER key to cause program execution to begin with the first line of the program, or begin at another line number by entering that line number and pressing ENTER.

No program exists in Main Memory upon power-on. You must put one there with the RECALL command or by entering a program with the EDIT function. Pressing RUN without a program in Main Memory will have no result.

4.3.1.3. RESUME Command

(F3) Pressing the RESUME function key completes any moves that were stopped when the STOP key was pressed, and resumes program execution at the point where the program was halted.

Note

> The RESUME command will not complete velocity streaming profiles because velocity streaming must be entered using either the ENTSTR or ENISTR commands.

4.3.1.4. RECALL Command

(F4) This command loads a program from the 3000's NVM into Main Memory. The entire contents of the NVM is loaded into Main Memory when this function key is pressed. The programs may then be edited with the EDIT function or executed with the RUN command.

4.3.2. MAIN Menu

The MAIN Menu, which has two pages, is the 3000's primary menu. This menu is shown in Figure 4-10. You can use it to save programs that are in Main Memory by putting them into NVM, recall programs that are stored in NVM, control communications over one of the two computer interfaces (IEEE-488 or RS-232C), define each axis' zero position, and select information to be displayed during program execution. The MAIN Menu is also used to invoke the EDIT and SETUP Menus. Refer to Figures 4-11 and 4-12 while discussing the commands in this Menu.

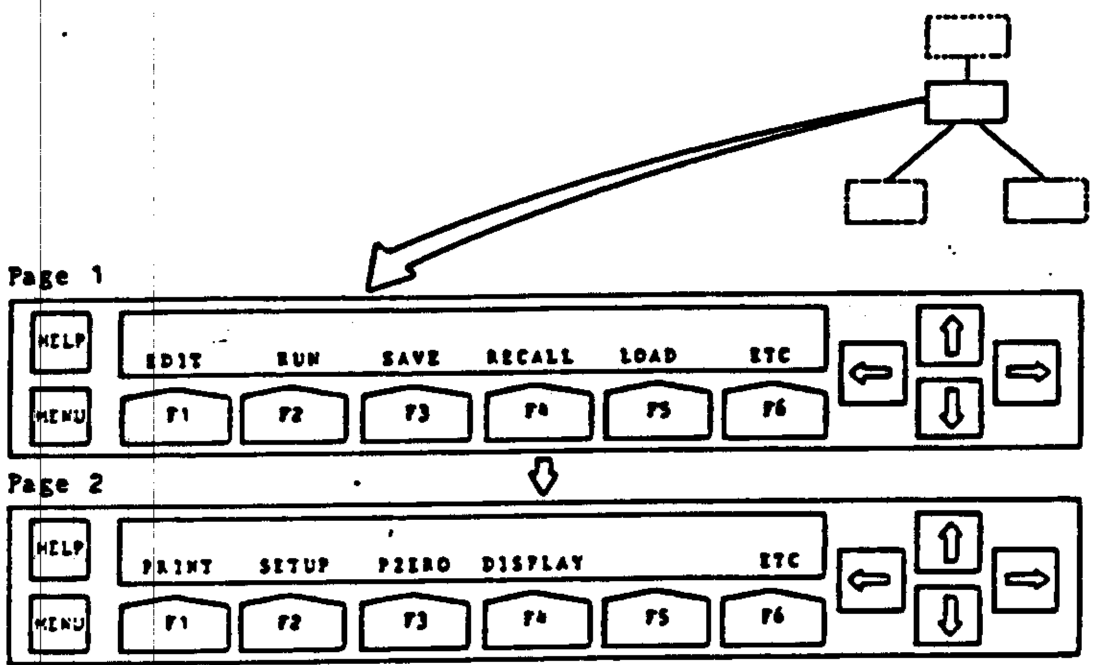


Figure 4-10: MAIN Menu

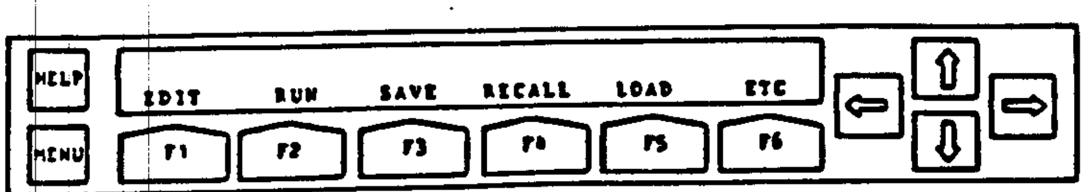


Figure 4-11: Page 1 - MAIN Menu

4.3.2.1. EDIT Command

(F1, Page 1) This function key moves you down one menu level from the MAIN Menu to the EDIT Menu. The EDIT Menu allows you to enter programs into the 3000's Main Memory. Each statement in the program must contain a line number. Enter programs by pressing the key that corresponds to the function you want in your program. Each function is described in detail in Section 4.3.4.

4.3.2.2. RUN Command

(F2, Page 1) When you press this key, the line RUN FROM 0000 appears on the display. You may key in the desired statement number and press ENTER, and the 3000 will execute the program stored in its memory starting at the selected statement number, or press ENTER without entering a line number and the 3000 will begin program execution with the first statement in the program (the first statement is the line with the lowest line number).

4.3.2.3. SAVE Command

(F3, Page 1) The SAVE command causes the 3000 to save the program(s) and SETUP parameters that are currently in the Main Memory by loading them into the NVM. Once the program is saved, it will not be lost if the 3000 loses power or is turned OFF. Note that any previous programs in the NVM will be erased.

4.3.2.4. RECALL Command

(F4, Page 1) Press this function key to RECALL the program(s) and SETUP parameters stored in NVM into Main Memory. The 3000 does not automatically load the program from NVM on power-up. RECALL causes the entire contents of the NVM to be loaded into Main Memory, and erases any programs(s) previously stored in Main Memory. If a data error is detected during this operation, an error message will be displayed and you have the choice of recalling the program by pressing ENTER or aborting the recall by pressing STOP.

4.3.2.5. LOAD Command

(F5, Page 1) The LOAD command instructs the 3000 to accept program statements from the RS-232C or IEEE-488 interface and store them in MAIN Memory. Statements must not exceed 40 characters and must be terminated with a semicolon. The LOAD function is completed when a two-character line consisting of an asterisk followed by a semicolon is sent to the 3000. Each statement must conform to the EDIT description (Section 4.3.4). Refer also to Section 4.3.

4.3.2.6. ETC Command

(F6, Page 1) The ETC key, which is always function key F6 on any page of a multipage menu, is used to step to the next page of the menu. Pressing the ETC key while the last page of a menu is displayed will cause the first page of the menu to be displayed, starting the cycle over again.

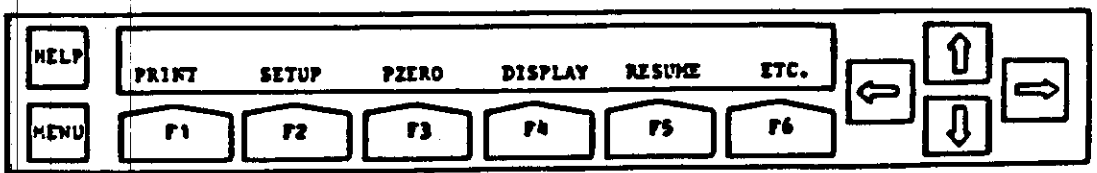


Figure 4-12: Page 2 - MAIN Menu

4.3.2.7. PRINT Command

(F1, Page 2) The PRINT command causes the program stored in the 3000's memory to be sent to the RS-232C or IEEE-488 interface with a semicolon, a carriage return, and a line feed terminating each statement. The entire program is sent followed by a line consisting of an asterisk followed by a semicolon, a carriage return, and a line feed. The PRINT and LOAD functions may be used together to send a program from one 3000 to another.

4.3.2.8. SETUP Command

(F2, Page 1) The SETUP function key takes you down one menu level to the SETUP Menu to set up operating parameters for several 3000 commands, as described in Section 4.3.3.

4.3.2.9. RESUME Command

(F5, Page 2) Pressing the RESUME function key completes any moves that were stopped when the STOP key was pressed, and resumes program execution at the point where the program was halted.

Note

> The RESUME command will not complete velocity streaming profiles because velocity streaming must be entered using either the ENTSTR or ENISTR commands.

4.3.2.10. PZERO Command

(F3, Page 2) This command defines the current position of all axes as the absolute zero position. That position then serves as the zero reference for subsequent absolute positioning within a 3000 program.

4.3.2.11. DISPLAY Command

(F4, Page 2) The DISPLAY function key allows the 3000 to display the status position, velocity, or program line number dynamically for each axis on the upper line of the display whenever a program is running.

The format of the DISPLAY command is shown in Figure 4-13. The top line of the display is divided into four fields. You can select the information to be displayed in each field by choosing one of five different parameters: position, velocity, axis status, blank, or current program line number.

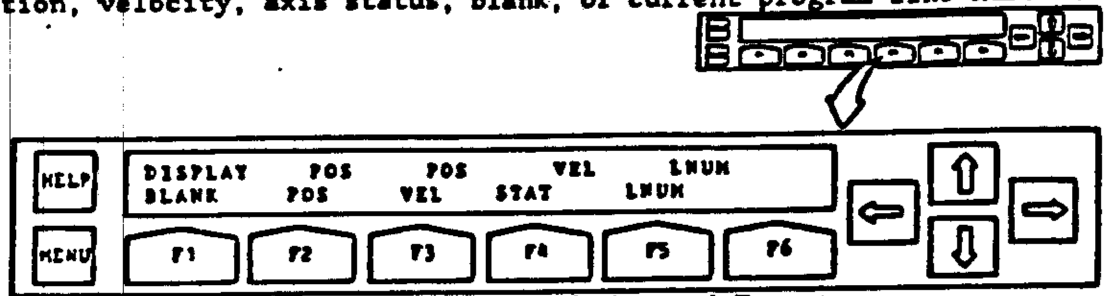


Figure 4-13: DISPLAY Command Format

Select the parameter to be displayed for each axis with function keys F1 through F5. Separate each selection by a space, using the C/E key. Complete the command by pressing ENTER. The selections available are shown in Figure 4-14.

- VEL - current velocity in user units
- POS - current position in user units
- BLANK - nothing for this axis
- LNUM - the program line number currently being executed (IDLE if no program is running)
- STAT - the axis status. The possibilities for this display are:

  - STOPPED - motor is not moving
  - MOVING - motor is moving
  - ERROR - the axis is not responding; possible 1811 failure
  - +LIMIT - the CW limit has been hit
  - -LIMIT - the CCW limit has been hit
  - LIMITS - both limits are active
  - HOME - the axis is in the home position
  - +CONVEL - the axis is moving clockwise at constant velocity
  - -CONVEL - the axis is moving counter clockwise at constant velocity
  - +ACCEL - the axis is accelerating in the clockwise direction
  - -ACCEL - the axis is accelerating in the CCW direction
  - +DECEL - the axis is decelerating in the CW direction
  - -DECEL - the axis is decelerating in the CCW direction

Figure 4-14: DISPLAY Selections

For example, the DISPLAY Menu shown in Figure 4-15 will display the current position for Axes 1 and 2, the current velocity for Axis 3, and the number of the program line being executed for Axis 4. The values are updated ten times per second and are displayed continuously during program execution.

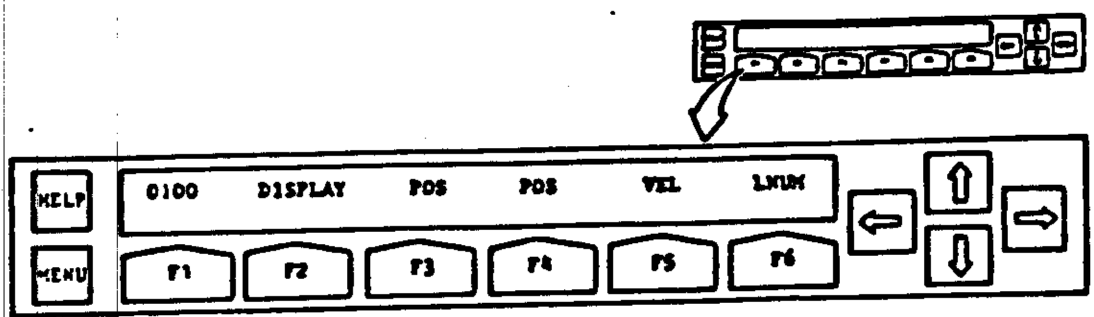


Figure 4-15: DISPLAY Example

4.3.3. SETUP Menu

The SETUP Menu, shown in Figure 4-16, has two pages and is the menu you use to configure the 3000's operating parameters. You can use the ETC key to toggle back and forth between the two menu pages. The setups that are established with this menu can be saved with the SAVE function (Section 4.3.2.3).

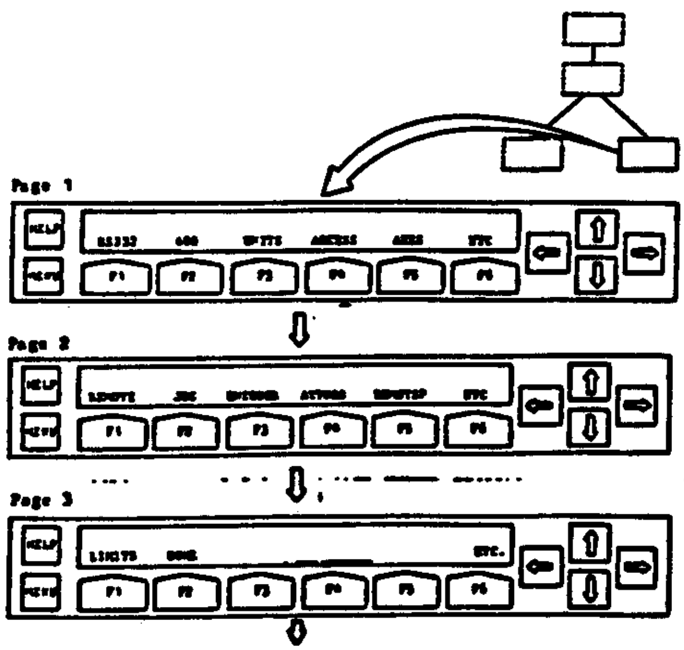


Figure 4-16: SETUP Menu

4.3.3.1. RS-232 Command

(F1, Page 1) When the RS-232 (F1) key is pressed, the RS-232 SETUP Menu is displayed. The RS-232 command allows you to set the baud rate, parity, data bits, and stop bits for the UART in the 3000. It also allows you to select DTE or DCE configuration by selecting which pins of the RS-232C connector are used to transmit and receive data. For use with most terminals and printers, the 3000 should be DCE. When used as a peripheral to a host computer, the 3000 should be DTE. When you are downloading programs from one 3000 to another, the two 3000s should have different configurations (if a 1-to-1 RS-232C cable is used).

Pressing the RS-232 key causes the lower display line to display the selections available for the RS-232C interface, as shown in Figure 4-17. The currently selected values for each of these parameters are displayed directly above each word.

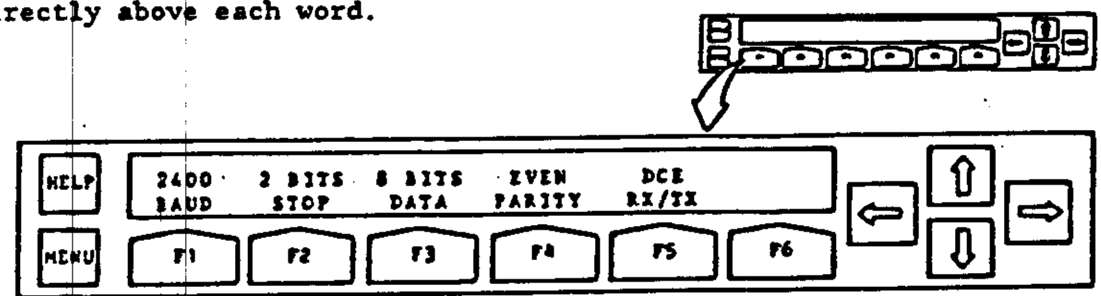


Figure 4-17: RS-232 SETUP Submenu

Pressing the function key for any of the five parameters (listed below) will cause the unit to increment through the list of possible values for that parameter. Pressing ENTER will "freeze" the value displayed. The choices for each parameter are:

Baud Rate:	110, 300, 600, 1200, 2400, 4800, 9600, and 19200.
Stop Bits:	1, 1.5, or 2.
Data Bits:	5, 6, 7, or 8.
Parity:	Odd, even, or none.
Rx/Tx:	DTE receives on pin 3 and transmits on pin 2. DCE mode transmits on pin 3 and receives on pin 2.

Note

Although the RS-232C interface can accept baud rates as high as 19,200, the incoming characters are buffered and processed at an effective rate of 1200 baud. Thus, it is possible that remote command execution may fall behind the incoming commands.

Press ENTER to set up the RS-232C interface according to the displayed parameters. Press the MENU key to return to the SETUP Menu.

4.3.3.2. 488 Command

(F2, Page 1) When the 488 (F2) key is pressed the IEEE-488 SETUP Submenu (Figure 4-18) is displayed. This command selects the 3000's IEEE-488 device address and whether a service request (SRQ) is generated by the 3000 each time it wants to send a message.

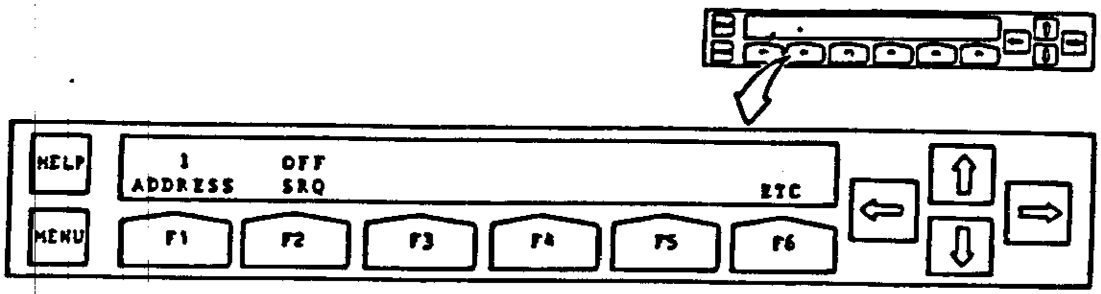


Figure 4-18: IEEE-488 SETUP Submenu

To select the 3000's IEEE-488 device address, press the ADDRESS function key and enter a number from 0 to 30 (followed by pressing ENTER). Pressing the SRQ key toggles the SRQ flag ON and OFF. When the flag is ON, the 3000 generates an SRQ whenever it wants to send data to the host computer. The 3000 generate an SRQ for each line of text it wishes to transmit. SRQs are also generated if the 1811 Command Language is being used and individual 1811 boards have information to transmit (see Section 4.4.2).

4.3.3.3. UNITS Command

(F3, Page 1) When the UNITS (F3) key is pressed, the UNITS SETUP Submenu in (Figure 4-19) is displayed. This submenu allows you to specify distance, velocity, and acceleration parameters in useful units. For example, a machine being controlled by a 3000 might move .001 inch for every 12 motor steps. It may be more convenient to program the move distances in multiples of .001 inch, rather than in motor steps. The 3000 allows you to specify a distance unit of 12, useful for this example, so that programming a distance of "1" will generate 12 motor steps and move the machine .001 inch. Note that the UNITS value may contain a decimal point. The valid range for a unit specification is: 0.0001 to 65,535. The value may be a maximum of six characters including the decimal point.

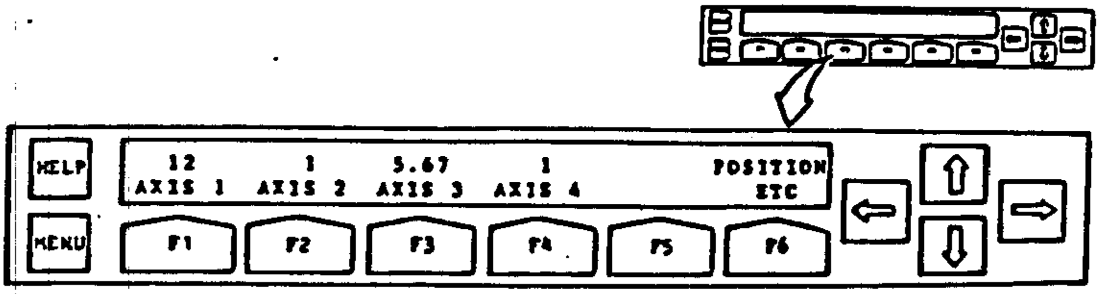


Figure 4-19: UNITS SETUP Submenu

The UNITS SETUP Submenu also allows selection of velocity (output pulse frequency) ranges for each axis. Different velocity ranges determine top speed, output pulse width, and resolution. Each axis can be programmed for a different velocity range.

Note

The UNITS command does not operate on default velocity and acceleration values.

The upper display line shows the values for each axis for the parameter identified by the keyword (the keyword is POSITION in the example shown in Figure 4-19). There are four keywords for this submenu; each time you press the ETC function key, the keyword will change to the next one in series and the display for the four axes will change to display the values for the new keyword.

The keywords are:

POSITION - determines multiplier used for move distances in the MOVE, STRD, or TEACH commands. The default value is 1.

Note

If fractional position units are used with one of the incremental positioning modes, truncation will produce a cumulative positioning error.

VELOCITY - determines multiplier used for velocity entries with the VEL, STRV, and JOG VELOCITY commands. Default value is 1 step/sec. Changing this value to 2500 would result in units of 1/10 rev/sec for a standard 25000-step/rev Compumotor.

ACCEL - selects multiplier for acceleration values entered with the ACEL command. Default value is 1 step/sec/sec. Changing this value to 25000 would result in units of rev/sec/sec for standard 25000-step/rev compumotors.

RANGE VEL - selects the velocity range for each axis. There are five possible values. Each range provides a different output pulse width on the STEP line as follows:

Range (KHz)	Output Pulse Width	Resolution (Hz)
0-1000	0.5 microsec	30.0
0-500	1.0 microsec	15.0
0-50	10.0 microsec	1.5
0-5	100.0 microsec	0.15
0-0.5	1.0 millisec	0.015

To change the keyword, press the ETC key until the desired keyword appears in the upper right on the display. Change the POSITION, VELOCITY, and ACEL units by pressing the appropriate function key to select an axis, entering the number on the numeric keypad, then pressing ENTER. Change the velocity range by first selecting the RANGE VEL keyword with the ETC key, then pressing an individual axis function key several times until the desired velocity range appears on the upper line of the display. Press ENTER to "freeze" the new value. The 0-500 KHz range is the default range. Standard stepper translators require the 0-50 KHz range.

An example of this command was shown in Figure 4-19. The keyword POSITION tells you that the parameter being displayed is the distance multiplier. This example selects distance multipliers of 1 for the second and fourth axes and a multiplier of 12 for the first axis. Note that the third axis has a fixed-point, noninteger value for the multiplier. Fixed-point numbers with decimal points can be used for any of the UNITS multipliers. Multipliers must not exceed the value of 65535 or be longer than six digits including a decimal point. Programmed distances, velocities, and accelerations will be multiplied by the displayed parameter, and the result will be truncated if it is noninteger.

To exit the UNITS function, press the MENU key to return to the SETUP Menu.

4.3.3.4. ACCESS Command

(F4, Page 1) When the ACCESS (F4) function key is pressed, the ACCESS SETUP Submenu (Figure 4-20) is displayed. The ACCESS function key in the SETUP Menu is used to change the 3000's access code, or password.

Note

> Remember that this is different from the ACCESS command in the POWER-UP Menu.

Press the ACCESS key while in the SETUP Menu to display the password currently in use, as shown in Figure 4-20.

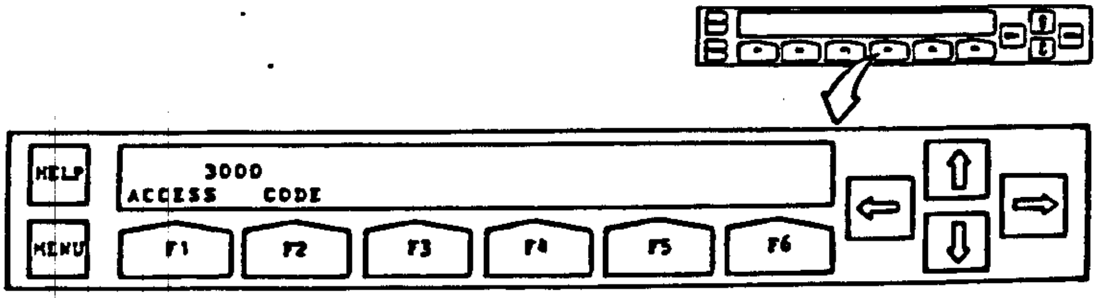


Figure 4-20: ACCESS SETUP Submenu

To change the password (shown as the factory-set "3000" in Figure 4-20), enter a new four-digit number, then press ENTER.

Note

The new password will not be saved unless you press the SAVE function key in the MAIN Menu.

To exit the ACCESS SETUP Menu, press the MENU key to return to the SETUP Menu.

4.3.3.5. AXES Command

(F5, Page 1) When the AXES (F5) function key is pressed, the AXES SETUP Submenu (Figure 4-21) is displayed. This submenu allows you to select absolute or incremental positioning individually for each axis. The choice of absolute or incremental positioning affects the meaning of the distance specified for the MOVE and MOVI commands in the EDIT Menu. For axes where incremental positioning is specified, the distance on the MOVE or MOVI parameter line is the distance and direction to move away from the current position. For axes in which absolute positioning is specified, the distance on the MOVE and MOVI parameter line is the position relative to the zero reference position established on power-up or by the PZERO or PDEF functions.

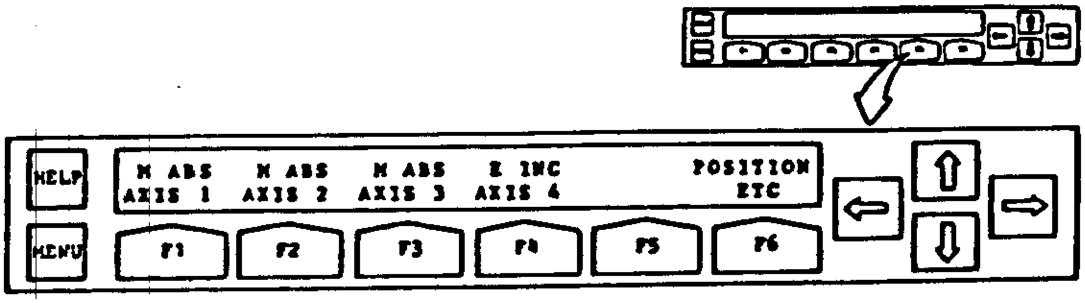


Figure 4-21: AXES SETUP Submenu (Keyword POSITION)

The AXES SETUP Submenu also determines whether distance units (before multiplication by each axis' UNITS multiplier) are to be specified as motor steps or encoder steps. Encoder steps may be specified as the distance unit for 3000s equipped with the Encoder Interface (-P option). If distance units of encoder steps are selected with the AXES command, remember that the 3000's quadrature detector multiplies the Encoder's standard line rate by 4, so a 1000-line encoder produces 4000 steps/rev when used with a 3000.

The keyword (shown as "POSITION" in Figure 4-21) can be POSITION, STALL - STOP?, NOT DB - PAUSE?, STALL - ABORT?, or STALL - OUTPUT?. If the keyword is POSITION, the descriptors above each axis label indicate the following:

Label	Description
M_INC	Motor steps, incremental positioning
M_ABS	Motor steps, absolute positioning
E_INC	Encoder steps (after quadrature), incremental positioning
E_ABS	Encoder steps (after quadrature), absolute positioning

Like the UNITS VEL RANGE Submenu, pressing the function key for any axis causes the upper line to scroll through the possible entries. Pressing F1 will, for example, cause the display above AXIS 1 to change from M_INC to M_ABS. Pressing it again will change the display to E_INC, etc. Entries are "frozen" by pressing ENTER.

Remember that Encoder units for distance entries may be specified only on 3000s equipped with the Encoder Interface. Using Encoder units in open-loop machines or without installing an appropriate incremental optical Encoder will cause all move commands to be ignored.

Figure 4-21 shows a setup that specifies that the first three axes in the machine are configured open loop, with distance units of Motor steps (before multiplication by the UNITS multiplier) and absolute positioning relative to each axis' zero position. The fourth axis, equipped with an Encoder, will interpret all distance entries to be Encoder counts after quadrature detect and will operate with an incremental coordinate system.

When you press the ETC function key so that the keyword in this menu is STALL - STOP?, you can program the axes with the Encoder Interface so the 3000 will issue a STOP to the specified axis when a stall is detected. This will cause the axis to stop the move currently in progress. The 3000 program will continue to execute and this may cause the axis to start moving again. Press the function key below each AXIS label to cause the YES and vice versa). Press ENTER to "freeze" the new parameter. The example in Figure 4-22 specifies move termination upon stall detect for Axis 4 only. Specifying move termination without connecting an optical Encoder to the appropriate axis or axes will generate an immediate stall and move termination.

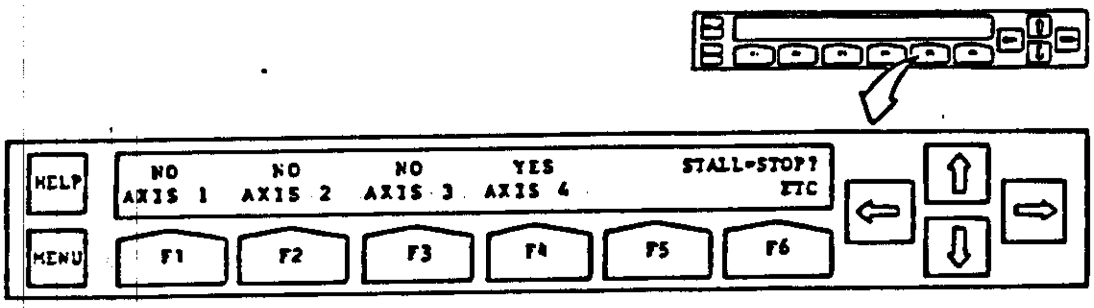


Figure 4-22: AXES SETUP Submenu (Keyword STALL - STOP?)

When you press the ETC function key so that the keyword in the menu is NOT DB - PAUSE?, you can program the axes with the Encoder Interface so the 3000 will cause the program to pause until all axes selected are within the user-definable deadband. Press the function key below each AXIS label to cause the displayed value (NO or YES) to toggle to the other value (NO will change to YES and YES to NO). Press ENTER to "freeze" the new parameter. The example in Figure 4-23 specifies PAUSE for deadband for Axis 4 only. Specifying PAUSE for deadband without connecting an optical encoder to the appropriate axis or axes will generate an immediate stall and the program will pause forever.

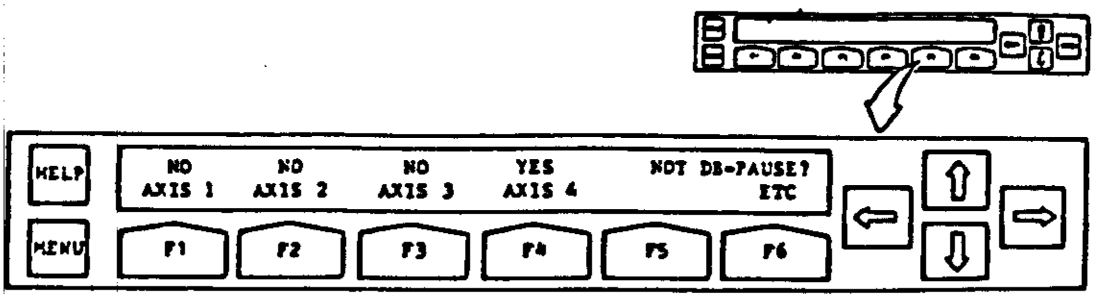


Figure 4-23: AXES SETUP Submenu (Keyword NOT DB - PAUSE?)

When you press the ETC function key so that the keyword in this menu is STALL - ABORT?, you can program the axes with the Encoder Interface so the 3000 will terminate program execution when a stall is detected. Press the function key below each AXIS label to cause the displayed value (NO or YES) to toggle to the other value (NO will change to YES and YES to NO). Press ENTER to "freeze" the new parameter. The example in Figure 4-24 specifies program abort only if Axis 4 stalls.

Note

Specifying program abort without connecting an optical Encoder to the appropriate axis or axes will generate an immediate stall and program abort.

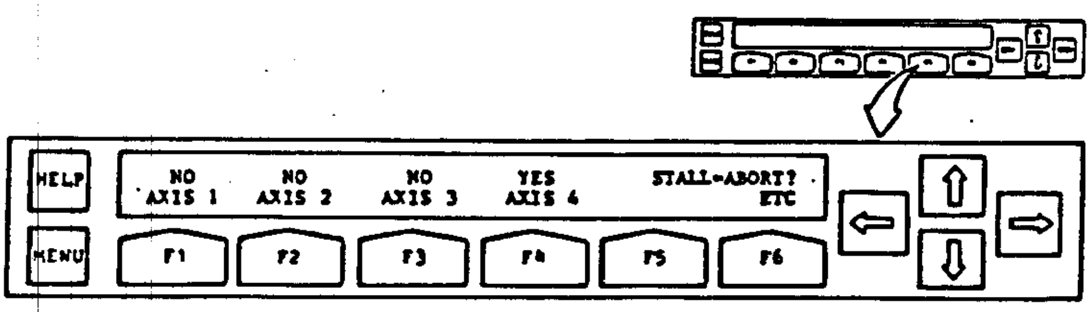


Figure 4-24: AXES SETUP Submenu (Keyword STALL - ABORT?)

When you press the ETC function key so that the keyword in this menu is STALL - OUTPUT?, you can program the axes with an Encoder Interface so the 3000 will enable or disable the output when a stall is detected. The default is NO, disabling the output when a stall is detected. Press the function key below each AXIS label to cause the displayed value (NO or YES) to toggle to the other value (NO will change to YES and YES to NO). Press ENTER to "freeze" the new parameter. The example in Figure 4-25 disables the output for all four axes when a stall is detected:

Note

If YES is specified, this will cause a conflict between OUT 1 on the Auxiliary Connector and the OUTSTR command output during velocity streaming or contouring because the same output is used for both functions.

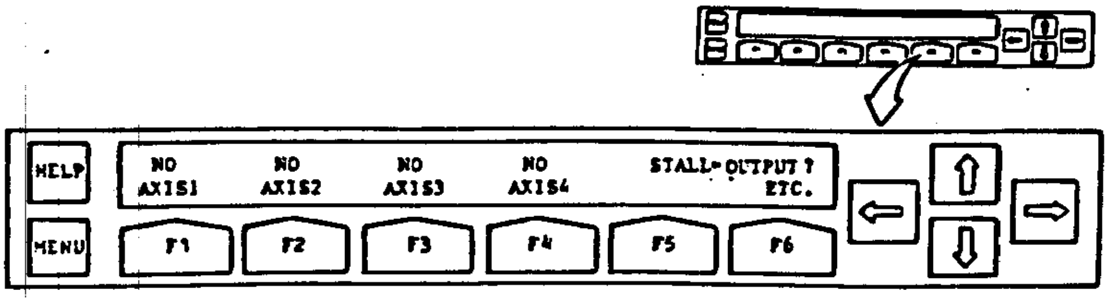


Figure 4-25: AXES SETUP Submenu (Keyword STALL - OUTPUT?)

To exit the AXES SETUP Submenu, press the MENU key to return to the SETUP Menu.

4.3.3.6. REMOTE Command

(F1, Page 2) When the REMOTE (F1) function key is pressed, the REMOTE SETUP Submenu (Figure 4-26) is displayed. The REMOTE command selects which of the two available communications interfaces (RS-232C or IEEE-488) is to be used to control operation of the 3000 and whether a menu display will be sent over the communications interface during remote operations.

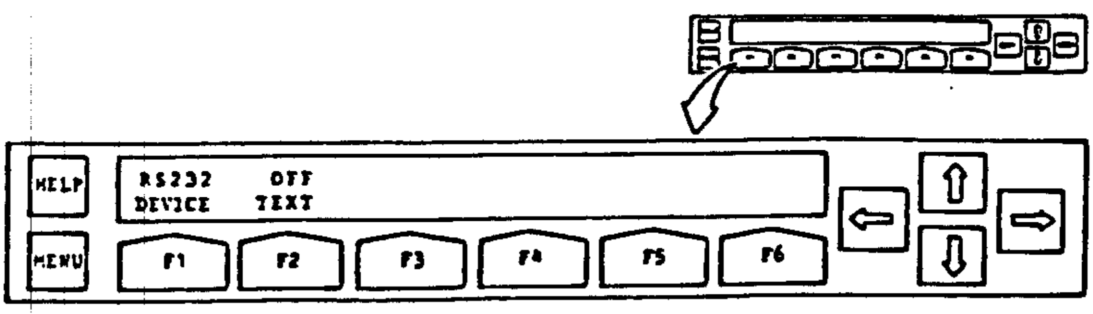


Figure 4-26: REMOTE SETUP Submenu

Above the word DEVICE is the label RS-232 if the RS-232C port is in use (as shown in Figure 4-26) or IEEE-488 if the IEEE-488 port is specified. Above TEXT is the word OFF or ON, depending on whether menus are to be sent to the host device over the link (ON) or not (OFF). These menus are especially useful if the 3000 is being programmed from a dumb terminal or terminal emulator as they walk the programmer through each menu. TEXT ON will also result in character echo when using the RS-232C interface. If the 3000 is being programmed from an external computer, this function is usually left OFF. When transferring a program from one 3000 to another 3000, TEXT must be set to OFF in both units.

4.3.3.7. JOG Command

(F2, Page 2) When the JOG (F2) function key is pressed, the JOG VELOCITY SETUP Submenu (Figure 4-27) is displayed. The JOG VELOCITY SETUP Submenu sets the low and high JOG velocities for each axis. Subsequent use of the JOG keys on the Front Panel will move the axes at the velocities specified by this command.

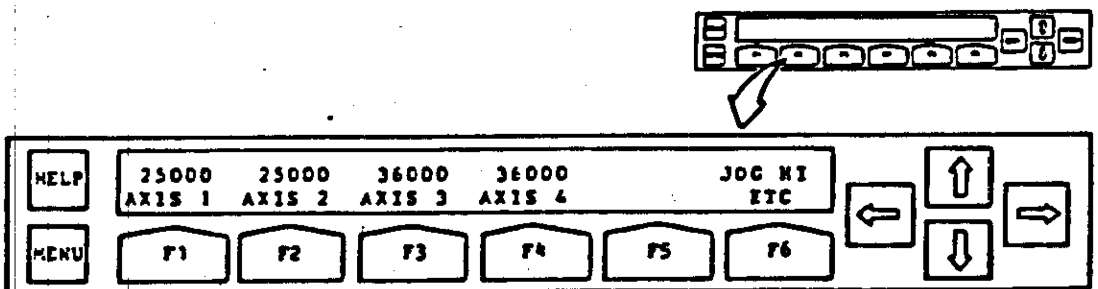


Figure 4-27: JOG VELOCITY SETUP Submenu

Note

This velocity will be scaled by the UNITS function.

The keyword for this menu is either JOG HI or JOG LOW, and the JOG velocities for each of the two speeds are displayed for each axis above the AXIS function keys. The velocities are multiplied by the velocity UNITS function scale factor, and the resulting product must be greater than 0 and less than 500,000. To specify the JOG speed for an axis, press the ETC function key. To select either JOG HI or JOG LOW, press the function key that selects the desired axis, enter the desired JOG speed in velocity units on the numeric keypad, then press ENTER.

In the example shown in Figure 4-27, Axes 1 and 2 have a high JOG velocity of 25,000 velocity units; Axes 3 and 4 are set for 36,000 velocity units.

To exit the JOG VELOCITY SETUP Menu, press the MENU key to return to the SETUP Menu.

4.3.3.8. ENCODER Command

(F3, Page 2) When the ENCODER (F3) function key is pressed, the ENCODER SETUP Submenu (Figure 4-28) is displayed. Indexers equipped with the Encoder Interface (-P option) require specification of the motor resolution (in steps/rev) and encoder resolution (in steps/rev, after the 4x quadrature detect circuitry) for proper closed-loop operation. These values can be different for each axis.

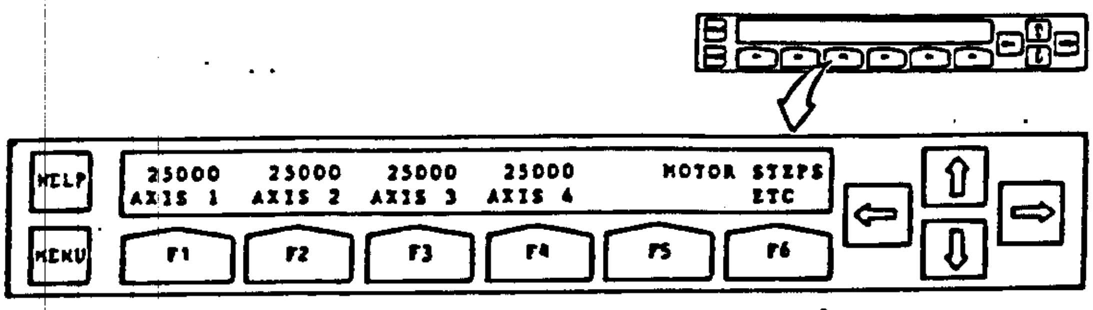


Figure 4-28: ENCODER SETUP Submenu (Keyword - MOTOR STEPS)

The display shows the number of motor steps/rev for each axis. Press the ETC function key to change the display to show the number of encoder steps/rev (Figure 4-29).

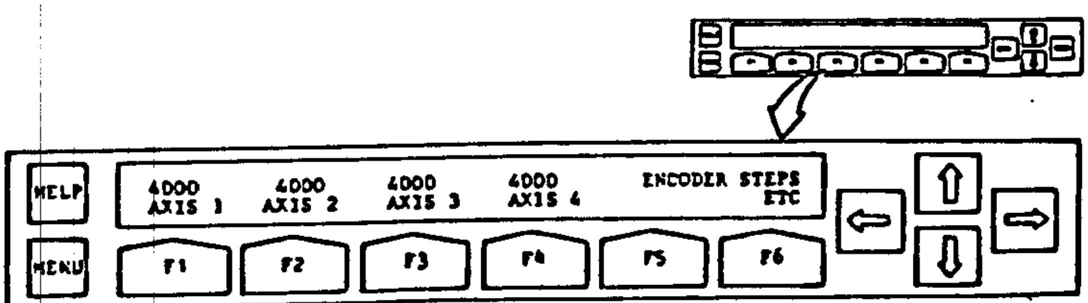


Figure 4-29: ENCODER SETUP Submenu (Keyword - ENCODER STEPS)

The numbers used here are representative of typical values. The number of steps per revolution (motor steps) specified for each axis must be less than 65535. The same restriction applies to the encoder resolution (encoder steps). The value 4000 would be used to specify a 1000-line encoder, which yields 4000 counts per revolution (after quadrature) when connected to a unit with the Encoder Interface. If the encoder is related to the motor in a fashion that does not result in one encoder revolution per motor resolution, that factor must be taken into account so that the motor and encoder resolutions specified result in the correct ratio of motor counts to encoder counts.

To change a value, move the cursor via the cursor arrow keys to select the desired axis. Enter the new value on the numeric keypad, then press ENTER to "freeze" the new value.

Press the ETC function key to change the display to show the number of steps in the DEADBAND (Figure 4-30) to use with NOT DB - PAUSE in the AXES SETUP submenu (Section 4.3.3.8). The number placed here is the width of the "in position" deadband region in encoder steps. The number must range from 0-65535.

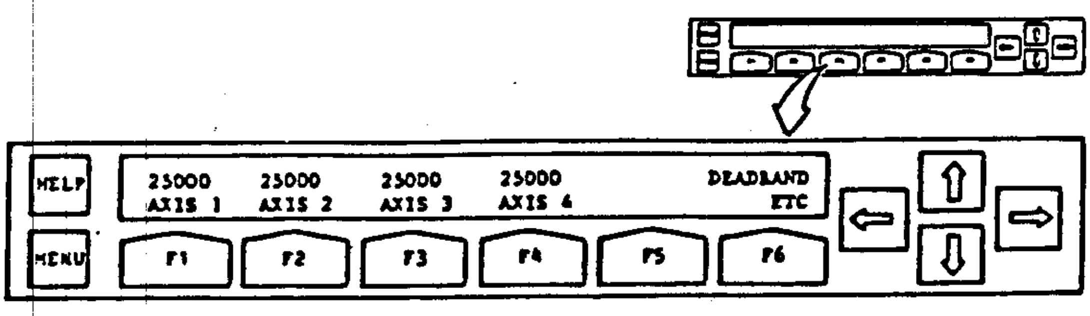


Figure 4-30: ENCODER SETUP Submenu (Keyword - DEADBAND)

Press the ETC function key to change the display to show the number of steps in the BACKLASH (Figure 4-31) region.

This command identifies the number of motor steps allowed without seeing an encoder pulse before the 3000 will determine that a stall condition exists. This is default set to 250 motor steps, which should be adequate when the Encoder is attached to the Motor. The backlash value must be between 0 and 65535.

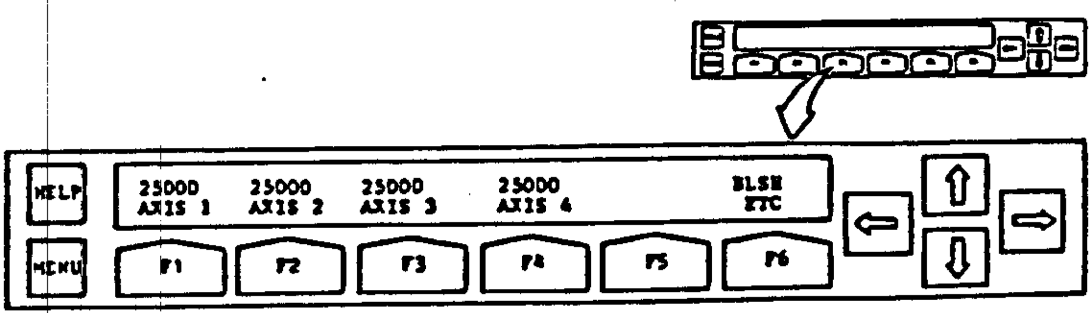


Figure 4-31: ENCODER SETUP Submenu (Keyword - BLSH)

Press the ETC function key to change the display to show the maximum servo correction velocity (Figure 4-32).

This command sets the maximum servo correction velocity for the Position Maintenance function. The units are in scaled velocity units, the same as in program velocity statements. The set velocity should be such that the Motor can start and stop reliably with no acceleration time.

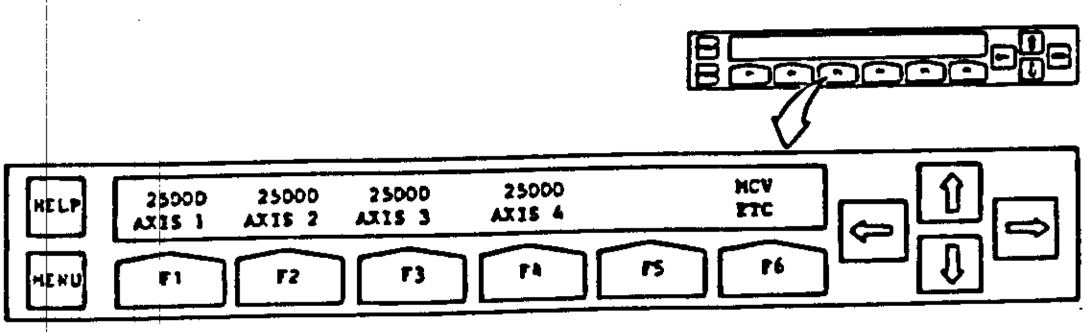


Figure 4-32: ENCODER SETUP Submenu (Keyword - MCV)

Press the ETC function key to change the display to show the position maintenance algorithm (Figure 4-33).

This command creates a correction velocity based on the positional error of the encoder. This error is the difference between the desired absolute encoder position and the actual encoder position. To correct this error, choose one of the three following algorithms by specifying the appropriate algorithm number:

Algorithm Number	Algorithm
1	correction velocity = K * error
2	correction velocity = K * $(\text{error})^{1/2}$
3	correction velocity = K * $(\text{error})^2$

K is the correction gain constant, which is defined by the SETUP ENCODER Submenu (Keyword - GAIN) below.

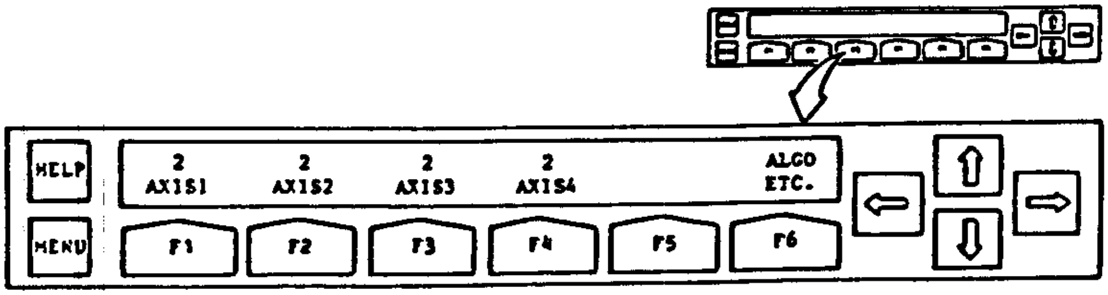


Figure 4-33: ENCODER SETUP Submenu (Keyword - ALGO)

Press the ETC function key to change the display to specify the correction gain constant for the position maintenance algorithm (Figure 4-34). See the SETUP ENCODER Submenu (Keyword - ALGO) above.

This command determines the correction velocity gain for the position maintenance algorithm and may be any number from 0 to 255. If 0 is specified, no correction will occur. Increasing the gain increases the speed with which position maintenance takes place. If the gain is too high, the Motor may oscillate. If the gain is too low, the Motor may take too long to correct its position. The optimum setting is usually determined by experimenting.

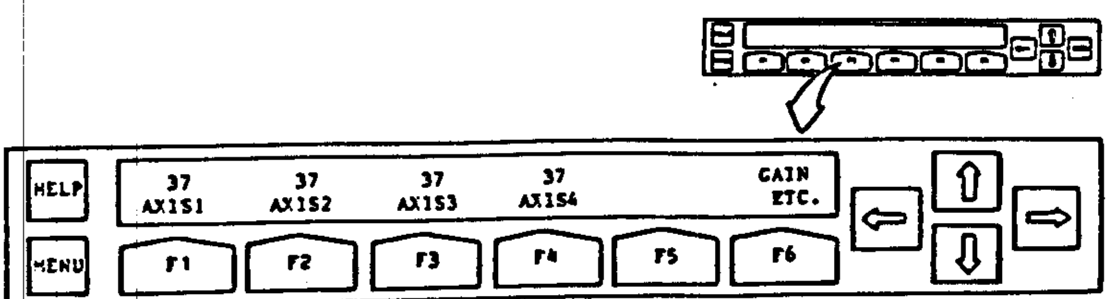


Figure 4-34: ENCODER SETUP Submenu (Keyword - GAIN)

Press the MENU key to return to the SETUP Menu.

4.3.3.9. AUTORR Command

(F4, Page 2) When the AUTORR function key (F4) is pressed, the AUTORR SETUP submenu (Figure 4-35) is displayed. This function allows you to select the RECALL and/or AUTORUN feature. If the AUTORECALL feature (AUTORCL) is selected, the 3000 will automatically copy the EEPROM contents into the program space when power is applied to the unit.

If the AUTORUN feature is selected, the 3000 will automatically run a program from the first line number when power is applied to the unit.

Note

If you select AUTORUN, you MUST select AUTORCL.

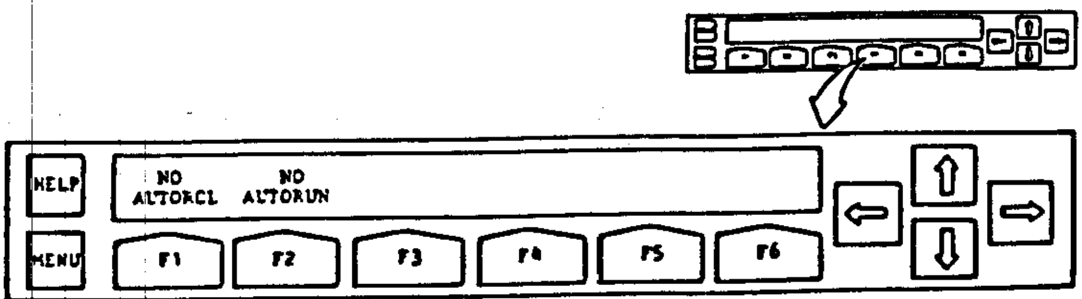


Figure 4-35: AUTORR SETUP Submenu

To exit the AUTORR SETUP Menu, press the MENU key to return to the SETUP Menu.

4.3.3.10. REMSTSP Command

(F5, Page 2) When the REMSTSP function key (F5) is pressed, the REMSTSP SETUP submenu (Figure 4-36) is displayed. These functions allow use of the OPTO-22 inputs for remote start/stop inputs to start or stop the program. INPUT #1 is assigned to STOP (if selected) and INPUT #2 is assigned to START the program.

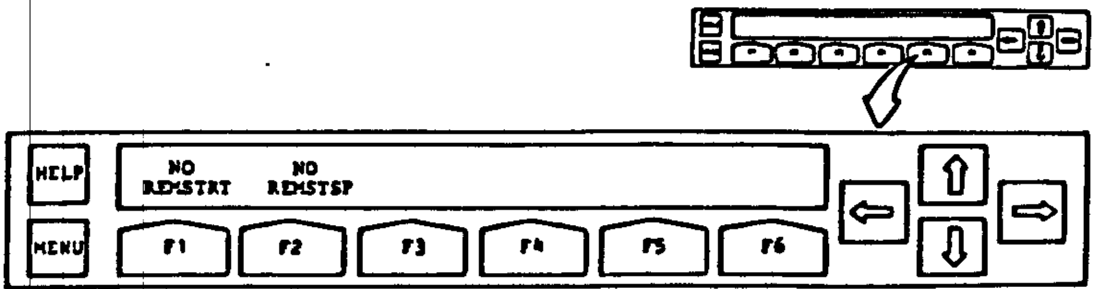


Figure 4-36: REMSTSP SETUP Submenu

To exit the RMSTSP SETUP Menu, press the MENU key to return to the SETUP Menu.

4.3.3.11. LIMITS Command

(F1, Page 3) When the LIMITS (F1) function key is pressed, the LIMITS SETUP Submenu (Figure 4-37) is displayed. This submenu allows you to set the absolute position boundaries for motion for each axis so that overtravel will not occur. A deceleration value is associated with these boundaries that causes the axis to decelerate at a specified rate converging towards the boundaries. An abort option allows you to cancel the program when either a hardware or software limit has been exceeded (see Section 3.7, Limit Switches for information on hardware limits).

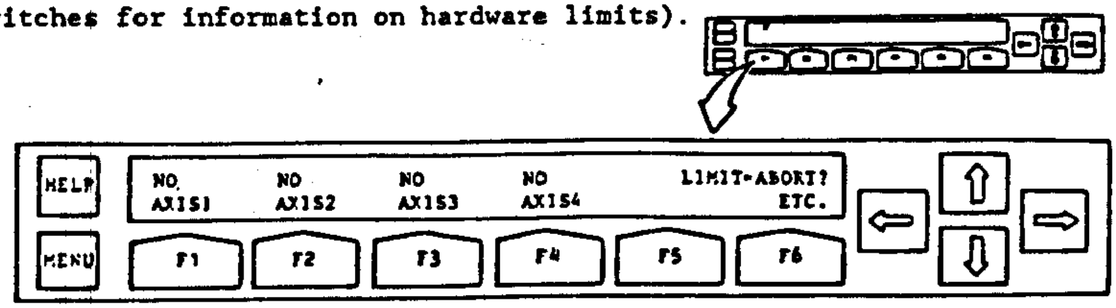


Figure 4-37: LIMITS SETUP Submenu

Various commands affect the upper and lower limits and the deceleration rate. The SETUP UNITS, SETUP AXES and MODE commands cause the value of the upper and lower limits and the deceleration rate to change because the parameters of these commands are multipliers for the SETUP LIMITS parameters. The PZERO and PDEF commands change the absolute reference position of an axis causing the upper and lower limits to use this new reference position. Although an axis can never go outside the limits during a move, it is possible, using the PZERO and PDEF commands before a move has begun, to start a move outside the limits. If this happens, the 3000 will always move in the direction to get back within the limits. Therefore, you should check all of these commands when setting up your limits and deceleration rate to see the effect they will have on the SETUP LIMITS parameters.

Note

The LIMITS - ABORT? YES option of the SETUP LIMITS command and the DISPLAY STAT command can be used as programming tools to tell you when your program has exceeded the limits.

The upper display line shows the values for each axis for the parameter identified by the keyword. The keyword LIMIT - ABORT? in the example shown in Figure 4-37 is the first keyword you will see when you press the UNITS (F3) key. There are four keywords for this submenu; each time you press the ETC function key, the keyword will change to the next one in series and the display for the four axes will change to display the values for the new keyword. The keywords are:

LIMIT - ABORT? - Determines whether the program will be aborted if a hardware or software limit is exceeded. The default is NO.

UPPER - Sets the absolute upper limit of motion an axis can travel. This value can be positive or negative but must always be more positive than the lower limit. The default is 0, or no limit set.

LOWER - Sets the absolute lower limit of motion an axis can travel. This value can be positive or negative but must always be more negative than the upper limit. The default is 0, or no limit set.

DECEL - Sets the rate an axis will decelerate when approaching the upper or lower boundary limits. The default is 2,500,000 steps/sec^2.

To change the keyword, press the ETC key until the desired keyword appears in the upper right on the display. To enter the UPPER, LOWER, and DECEL parameters, press the appropriate AXIS keys, enter the numbers using the keypad, and then press ENTER. To enter the LIMIT - ABORT? parameters of YES or NO, press the function keys under the appropriate axes until the desired option appears, and then press ENTER. When all the parameters in all the keywords have been entered, press the MENU key.

4.3.3.12. HOME Command

(F2, Page 3) When the HOME (F1) function key is pressed, the HOME SETUP Submenu (Figure 4-38) is displayed. This submenu allows you to define the HOME input active position for each axis. When an axis is directed to search for HOME by the HOME command, it will begin moving in the direction specified in the HOME command at the velocity specified in the last VELC command. When the HOME input becomes active, the axis will decelerate and backtrack if necessary to arrive at the HOME position. The HOME position can be either the "+" (CW) edge of the HOME input or the "-" (CCW) edge of the HOME input. The final approach to that edge is at a specified percentage of the initial search velocity, and the edge can be approached in the CW direction or the CCW direction so that the HOME input is active when the axis stops.

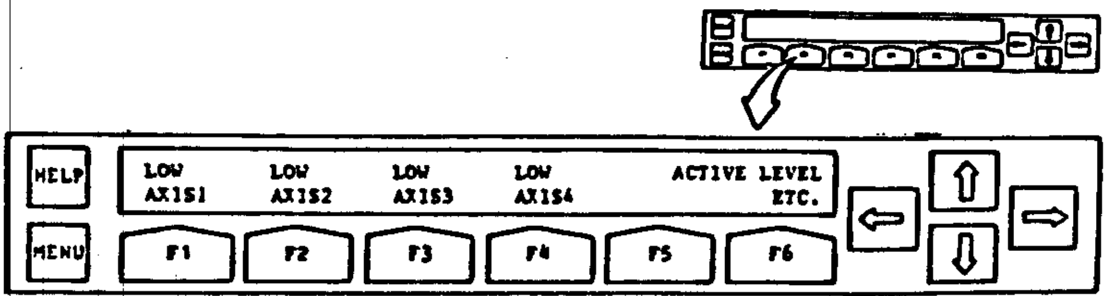


Figure 4-38: HOME SETUP Submenu

The upper display line shows the values for each axis for the parameter identified by the keyword. The keyword ACTIVE LEVEL in the example shown in Figure 4-38 is the first keyword you will see when you press the HOME (F2) key. There are four keywords for this submenu; each time you press the ETC function key, the keyword will change to the next one in series and the display for the four axes will change to display the values for the new keyword. The keywords are:

ACTIVE LEVEL - Sets the active state of the HOME input to either HIGH, which means active high (TTL 1), or LOW, which means active low (TTL 0). Figure 4-39 illustrates an active HOME input. The duration of the active state of the HOME input must be greater than 4 milliseconds.

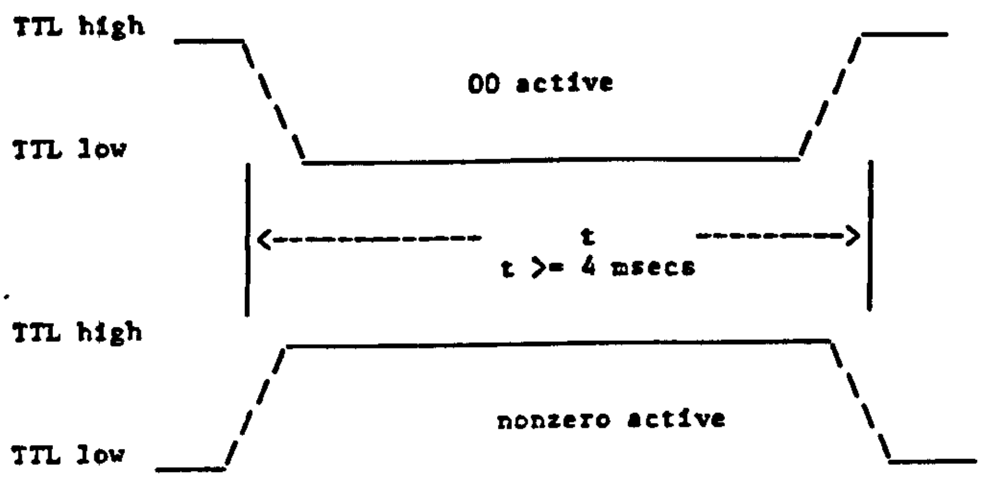


Figure 4-39: Active HOME Input

ACTIVE EDGE - Specifies whether the edge of the active HOME input will be the "+" (CW) edge or the "-" (CCW) edge. Figure 4-40 illustrates these edges.

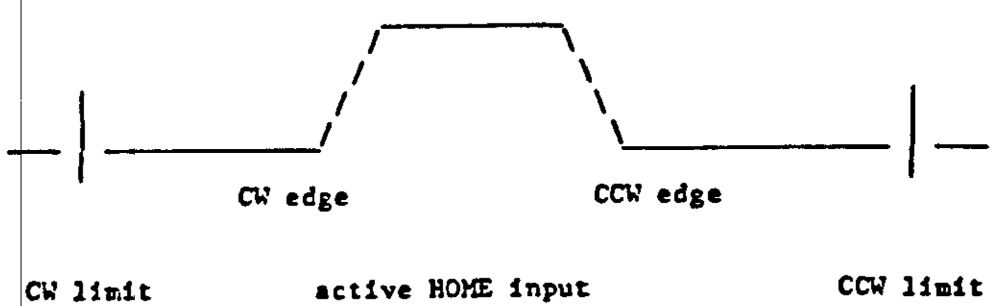


Figure 4-40: Active HOME Input Edges

FINAL VEL - Sets the velocity of the final approach to the edge of the active HOME input. This velocity is a percentage of the velocity specified by the last VELC command and can range from 1% to 100%. The lower the final velocity is, the more accurate the home position will be.

FINAL DIR - Sets the direction of the final approach to the edge of the active HOME input. If this parameter is CW, the final approach is in the clockwise direction towards the CW limit. If this parameter is CCW, the final approach is in the counterclockwise direction towards the CCW limit.

To change the keyword, press the ETC key until the desired keyword appears in the upper right on the display. To enter the ACTIVE LEVEL, ACTIVE EDGE and FINAL DIR parameters, press the function keys under the appropriate axes until the desired option appears, and then press ENTER. To enter the FINAL %VEL, press the appropriate AXIS keys, enter the numbers using the keypad, and then press ENTER. When all the parameters in all the keywords have been entered, press the MENU key.

The following example explains what happens when Axis 1 searches and finds its HOME position. Figure 4-41 is a diagram of this Homing move. The HOME SETUP parameters for Axis 1 are:

ACTIVE LEVEL = LOW
ACTIVE EDGE = CW
FINAL %VEL = 20
FINAL DIR = CCW

When a HOME command is encountered in the program, Axis 2 continues traveling at the velocity specified in the last VELC command searching for the active low input. When the "+" edge (CW) of the input is found, the axis starts to decelerate overshooting the HOME input edge. The axis recovers from this overshoot, then travels at 20% of the beginning velocity in the CCW direction until the HOME input edge is encountered.

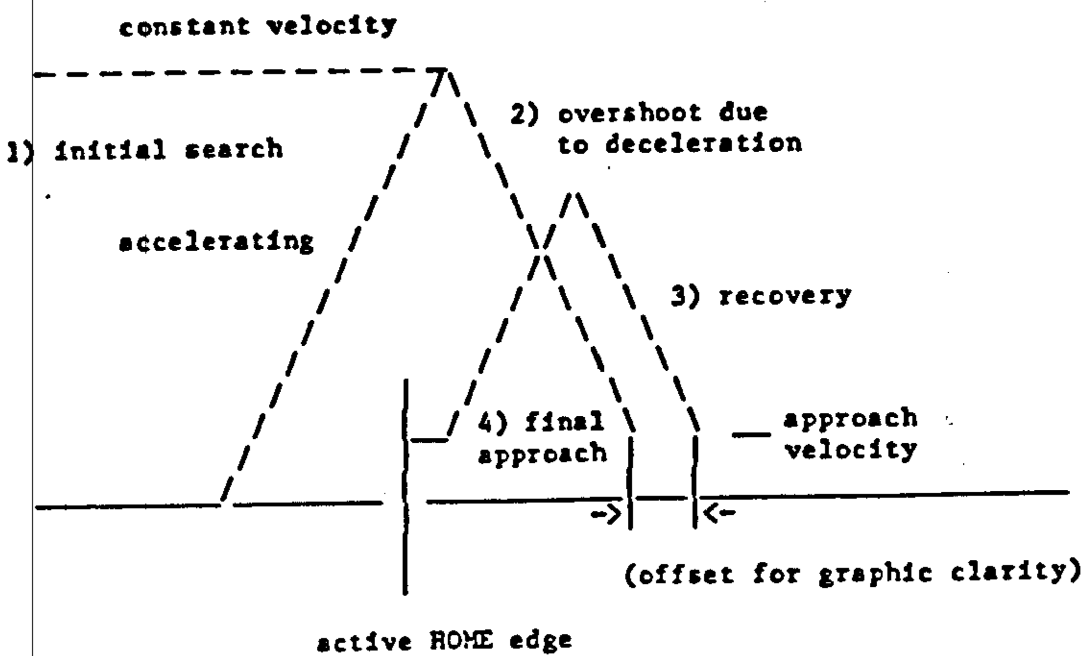


Figure 4-41: Homing Cycle

4.3.4. EDIT Menu -- Overview

The EDIT Menu is used to enter and modify programs within the 3000. It offers 40 different commands, displayed on 8 menu pages. Figures 4-42 and 4-43 show the relationship of the EDIT Menu to the other menus and the commands on each of its pages. As with the other menus, you must press the ETC key to change the display to the next menu page.

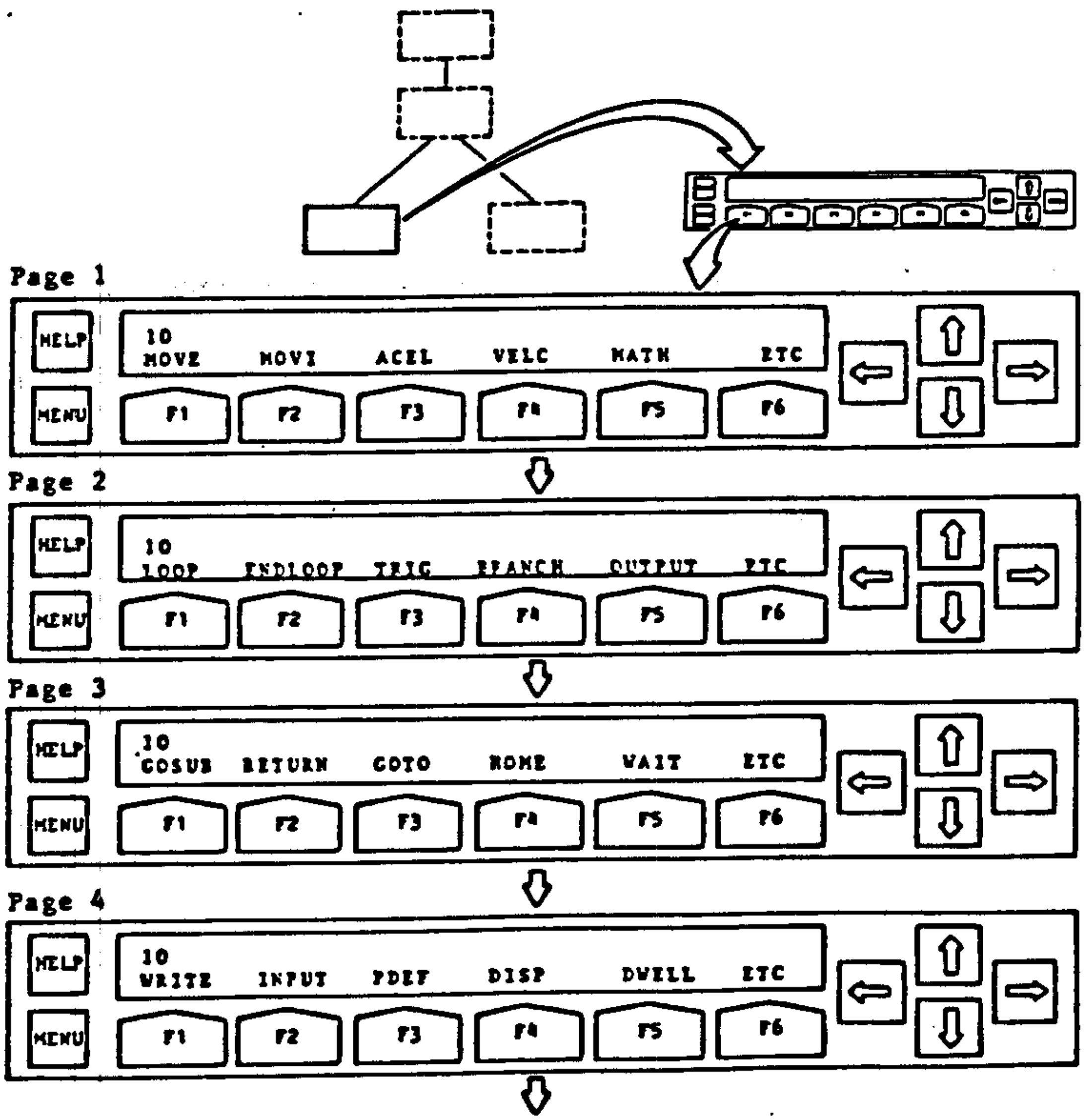


Figure 4-42: Pages 1 through 4 - EDIT Menu

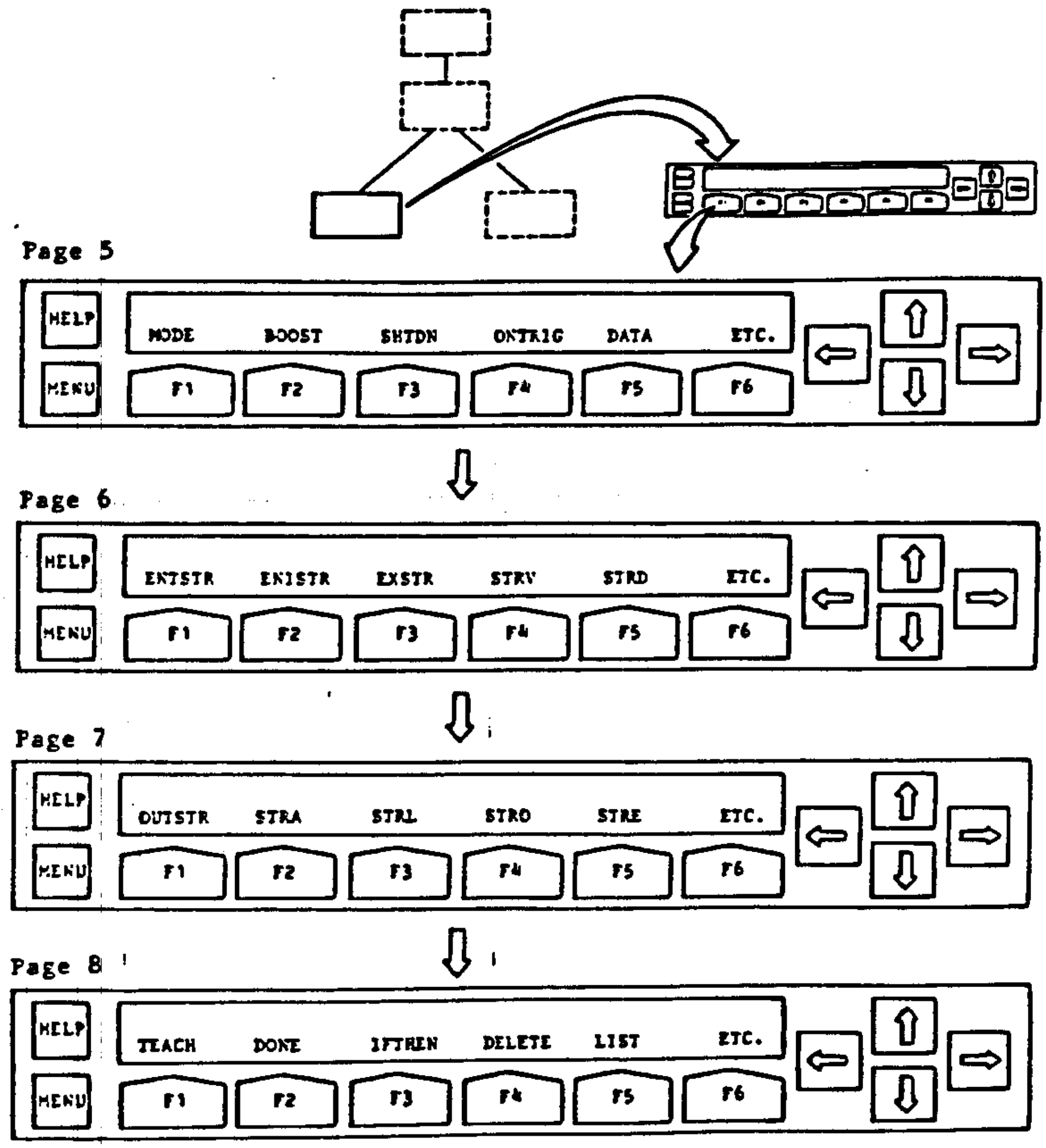


Figure 4-43: Pages 5 through 10 - EDIT Menu

To enter most numeric parameters associated with the EDIT Menu commands, press the desired function key, key in the value on the numeric keypad, and then press ENTER to store the number. Spaces between numbers are created with the C/E key, and the left and right arrows move the cursor left and right rolling around to the other side when the end is reached. Quick entry of parameters for different axes is accomplished by pressing the AXIS #x key below the Display.

Note

Pressing any of the four AXIS # keys moves the cursor to the right place for entering that axis parameter.

Note

Numbers that are too large or contain decimal points are rejected and cause the message "Invalid date entered"to be displayed on the Front Panel Display.

The up arrow and down arrow keys to the right of the Display can be used during editing to scroll forwards and backwards through your program. This allows easy access to any part of the program. Press either key once to cause the Display to show the previous program statement (up arrow) or the next program statement (down arrow), if any. Hold either key down to step rapidly through the program.

Press the MENU key to return to the Main Menu at the end of an editing session.

The EDIT Menu commands are grouped together according to the functions they perform. These functions are:

a. Motion Control

b. Motor/Drive Control

c. Program Control

d. Position/Coordinate Reference

e. Arithmetic/Computational

f. Input/Output

g. Teach

h. Utilities

Each function and the commands associated with it are discussed in detail below. Refer to Appendix A for a summary of these commands.

4.3.5. EDIT Menu -- Motion Control

Fourteen commands are used to specify motion parameters: four commands specify standard moves, which have trapezoidal or triangular velocity profiles; six commands specify Velocity Profiling, which allows complex velocity profiles within a preset move; and four commands specify Contouring, which creates complex profiles by connecting line and arc segments as part of a continuous move.

4.3.5.1. Trapezoidal/Triangular Moves

The standard motion-control velocity profile is a trapezoid or a triangle. Trapezoidal moves accelerate the Motor to a specified velocity, maintain that velocity for a period of time, and decelerate to a stop. Moves of a short distance or with low accelerations may never reach the specified velocity and begin decelerating when one-half of the programmed distance has been traversed. These moves are referred to as triangular moves.

There are two types of standard moves. Preset Moves move the Motor a specific distance and require an acceleration parameter, a velocity, and a distance or position. Slewing Moves require only an acceleration and a velocity. A Slewing Move is one that accelerates to a specific velocity and maintains that velocity until commanded to do something else.

Four commands are used in programming these moves:

a. MOVE - Commands the axis to move a preset distance or slew in a particular direction. All motion initiated by a MOVE command (that is, a preset distance or a new velocity for MOVE slew) must be completed before the program will execute the next statement. The distance moved is a function of the mode (absolute or incremental) specified for each axis and the values are multiplied by the scale factors defined by the UNITS command (Section 4.3.3.3).

b. MOVI - Like MOVE function except that the next statement in the program is executed immediately after beginning the MOVI.

c. ACEL - Sets the acceleration and deceleration ramps for each axis. The units are steps/second/second and are multiplied by a scale factor if one is specified by the UNITS command.

d. VELC - Specifies the velocity for each axis. This velocity is the maximum allowed for subsequent moves until another VELC command is executed. The values are in units of steps/second and the values entered are multiplied by the scale factors specified by the UNITS command.

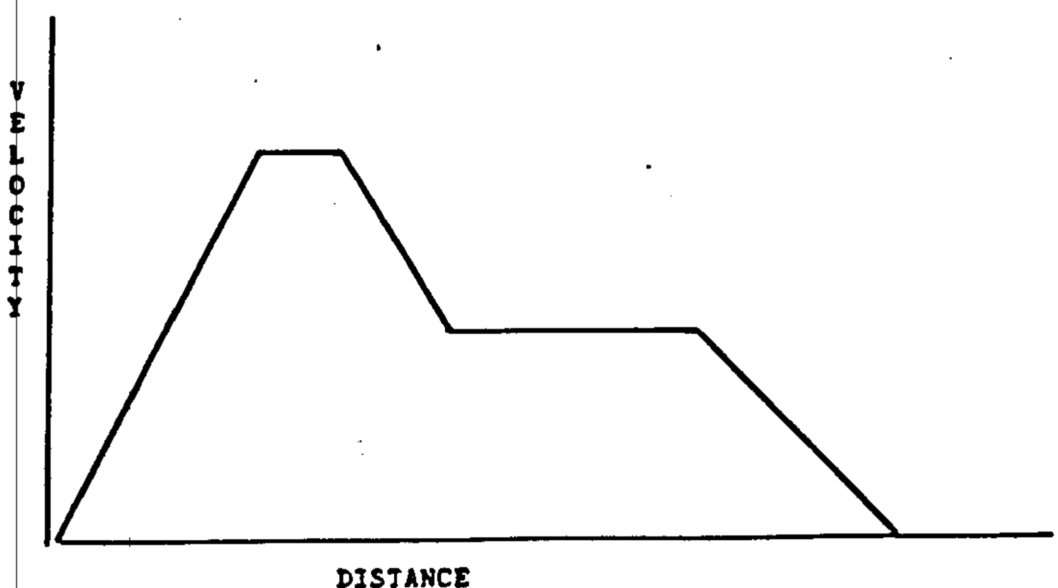
4.3.5.2. Velocity Profiling

The 3000 is capable of producing move profiles more complex than the standard triangle and trapezoid. Velocity Profiling allows you to move a preset distance with several different accelerations and decelerations and at different velocities. You accomplish this by defining the beginning of a profiled move with the ENTSTR command or the ENISTR command, which initializes the "Streaming" function starting at velocity zero and position zero (current position). These commands are followed by pairs of numbers representing distances from the beginning of the move and the desired velocities at those points. By specifying several velocity/distance pairs, you can form complex profiles. The 3000 automatically calculates the required accelerations and decelerations to reach the desired velocities at the desired distances.

Note

> Velocity Profiling works on Axes 1 and 2 only. The remaining axes can perform only trapezoidal and triangular moves.

An example of a profiled move is illustrated in Figure 4-44.



At each "knee", a new velocity and distance pair is used to determine the next breakpoint.

Figure 4-44: Velocity Profiling

Six functions are used to implement Velocity Profiling:

a. ENTSTR - The ENTSTR command tells the 3000 that an absolute velocity streaming move is to begin and sets the velocity and distance to zero.

b. ENISTR - The ENISTR command tells the 3000 that an incremental velocity streaming move is to begin and sets the velocity and distance to zero.

c. STRV - This command specifies the velocity half of the velocity/ distance pair, the velocity the Motor will be running at when the position specified with the STRD command has been reached. This command must always be followed at each occurrence with an STRD command. Any command(s) between STRV and STRD will be ignored.

d. STRD - The STRD command specifies the distance half of the velocity/distance pair. The distances are specified as absolute coordinates relative to the position the axis is at when the ENTSTR command is performed. It is always preceeded by a matching STRV command.

e. EXSTR - The EXSTR (Exit Streaming) command terminates Velocity Profiling.

f. OUTSTR - The OUTSTR (Output While Streaming) command allows one or two TTL output bits on each axis' Auxiliary Connector to be set high or low under program control at each breakpoint or "knee" in Velocity Profiling. This can be used to trigger external events when a particular distance or velocity is reached.

Note

If two axes are in Velocity Profiling mode, each pair of STRV and STRD commands must take the same amount of time. If one axis is slower than the other and does not finish its segment in time, it will receive a discontinuous jump each time the STRV and STRD pair for the other axis is completed.

4.3.5.3. Contouring

In addition to Velocity Profiling, the 3000 can be programmed for complex contouring. In the Contouring mode, Axis 1 and Axis 2 of the 3000 move simultaneously to produce line and arc segments and to connect these segments into a continuous complex contour. The 3000 automatically calculates the required accelerations and decelerations to traverse the contour and to move from segment to segment within a contour at a constant feedrate.

Note

Contouring is possible only when the 3000 is in the Velocity Streaming mode using the ENTSTR command or the ENISTR command.

A Cartesian x,y coordinate system is used by the 3000 to represent two-axis contouring moves (see Figure 4-45). The x coordinate represents Axis 1 and the y coordinate represents Axis 2. Therefore, the two-motor axis position in the x,y plane is always defined with two numbers, x,y, where x is the position of Axis 1 and y is the position of Axis 2. All four axes can be active at the same time. However, while Axis 1 and Axis 2 can perform all types of moves, Axis 3 and Axis 4 can only perform trapezoidal or triangular moves.

Figure 4-45: Cartesian x,y Coordinates

To perform Contouring, the 3000 must be programmed for Velocity Streaming using one of the following commands:

a. ENTSTR - Sets the 3000 in the absolute velocity streaming mode, positioning the Motors' coordinates at 0,0 in the x,y plane. This location becomes the absolute reference point in the x,y coordinate system for all segments of a contour that follow this command.

b. ENISTR - Sets the 3000 in the incremental velocity streaming mode, positioning the Motors' coordinates at 0,0 in the x,y plane. However, unlike the ENTSTR command, the ENISTR command does not maintain this point as a reference point for all segments of a contour. It uses this 0,0 location as the reference point only for the first segment of a contour and then it shifts the reference point to the location where the next segment begins, and so forth, until all the segments of a contour are completed. The final reference point on the x,y coordinates is the starting location of the final segment.

Four commands are used to implement Contouring: two commands are associated with line segment contouring and two commands are associated with arc segment contouring. The feedrate specified for Contouring uses UNITS parameters specified in the SETUP Menu. These functions are as follows:

a. STRL - The STRL command specifies the feedrate required at the end of a line segment. The next segment begins with the same feedrate. If the beginning and ending feedrates are the same, the segment is traversed with a constant feedrate.

b. STRE - The STRE command specifies the x,y coordinates at the end of a line or an arc segment. In the absolute velocity streaming mode, the x,y coordinates represent the location in the x,y plane with reference to the 0,0 point. In the incremental velocity streaming mode, the x,y coordinates represent the location in the x,y plane with reference to the beginning point of the line or arc segment being traversed.

Note

For an arc segment, the end point specified with STRE coordinates x,y must be on the arc. If not, the 3000 will continue on an arc segment to a point and then proceed with a straight line to the specified end point at a high velocity, which may cause the motor to stall.

c. STRA - The STRA command specifies the feedrate at which to traverse an arc segment. The feedrate along an arc segment is always constant and must be equal to the end velocity of a preceeding line segment to prevent a possible motor stall.

d. STRO - The STRO command specifies the x,y coordinates of the center, or origin, of an arc segment. The 3000 uses these origin coordinates to calculate the radius of the arc segment.

4.3.5.4. MOVE Command

(F1, Page 1) The MOVE command (Figure 4-46) lets you enter the distance or position of a Preset Move or the direction of a Continuous Move for each of the four axes. The 3000 will start all axes (up to four) at once. Before proceeding to the next statement, the 3000 will wait until axes without NULL entries finish either Preset Moves or the acceleration or deceleration of Continuous Moves.

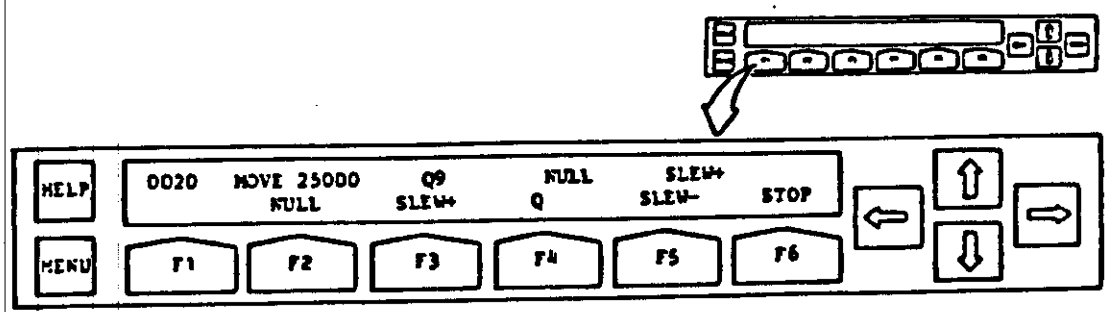


Figure 4-46: MOVE Command

The MOVE parameter for each axis can be a number, a variable, or the words SLEW+, SLEW-, or NULL. If the parameter shown is a number or variable, it has one of two meanings. If the axis is set up for relative positioning, it is the distance to move in motor steps relative to the current position. The distance is a signed number. Positive distances result in clockwise moves, negative distances in counterclockwise moves. If the axis is set up for absolute positioning, that is the absolute position to which to move. The sign is assumed to be positive if not entered. If the parameter is SLEW+ or SLEW-, that axis will begin to move in the specified direction. When the set velocity is reached, the axis will continue moving at constant velocity until directed to change velocity via subsequent VELC, and MOVE or MOVI program statements. The motion may also be stopped using the MOVE STOP command.

The NULL key (F2) can be used to indicate that no action should take place on a given axis. Use this key when you want to leave one or more axes undisturbed in their operation while initiating a move on another.

The Q key (F4) can be pressed, followed by a numeric entry (1-99) and ENTER, to specify that a variable (Q1-Q99) will be used to define the move distance or position.

The STOP key (F6) can be pressed to stop the move in progress at the deceleration rate defined by the ACEL command prior to the move.

In the example shown in Figure 4-46, when the program executes statement 0020, Axis 1 will perform a preset move of 25000 counts (multiplied by any scale factor established by UNITS command). Axis 2 will move a preset distance specified by the variable Q9 (also multiplied by the UNITS scale factor), whose value is established with an INPUT statement or with the MATH function. Axis 3 will remain undisturbed and will continue to do whatever it had been doing (or not doing). Axis 4 will slew in a clockwise direction at the current acceleration specified for the axis (ACEL command) to the velocity currently specified for that axis (VELC command).

All Preset Moves must be completed and all Slewing Moves must reach their programmed velocity before the next statement in the program will be executed.

4.3.5.5. MOVI Command

(F2, Page 1) This command is identical in all respects to the MOVE command except that the next program statement is executed immediately after the command is executed. The MOVI command allows the 3000 to begin a move and then proceed to the next instruction without waiting for Preset Moves to be completed or Slewing Moves to reach their programmed velocity.

4.3.5.6. ACEL Command

(F3, Page 1) The ACEL command specifies the acceleration and deceleration ramps for each axis. The entries may be a number, a variable, or the word NULL, in which case the previously programmed value is used. The format of the command is shown in Figure 4-47.

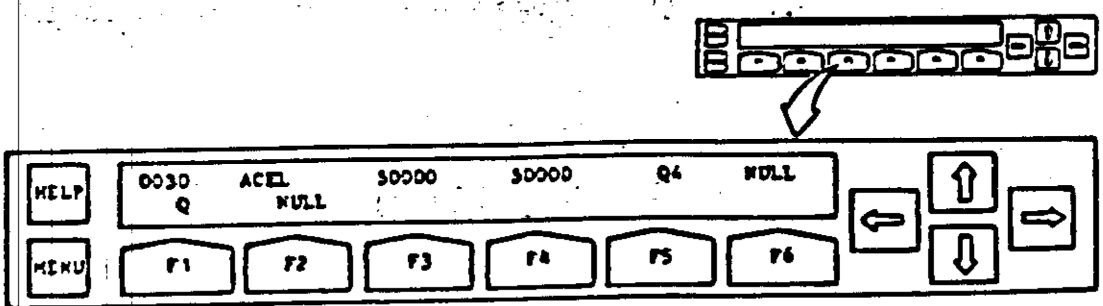


Figure 4-47: ACEL Command

Remember that any value you enter will be multiplied by the scale factor you entered using the UNITS command.

Note

The 3000 will accelerate each axis at a rate of 25000 steps/sec/sec unless you change that value by using this command. This default value corresponds to 1 rev/sec/sec for a standard 25000-step/rev Compumotor. Default values are not multiplied by UNITS command scale factors.

The example shown in Figure 4-47 would set the acceleration for Axes 1 and 2 to 50000. If the UNITS scale factor is 1, this would correspond to an acceleration of 2 rev/sec/sec with a standard 25000- step/rev Compumotor. Axis 3 uses the value of variable Q4 and Axis 4 remains undisturbed from the previous setting. If there was no previous setting, the default value of 25000 will be retained.

4.3.5.7. VELC Command

(F4, Page 1) The VELC command is used to specify the peak velocity for each axis. Entries may be a number, a variable, or the word NULL, in which case the previously programmed value is used. The format of the command is shown in Figure 4-48. Remember that any value you enter will be multiplied by the scale factor you entered using the UNITS command.

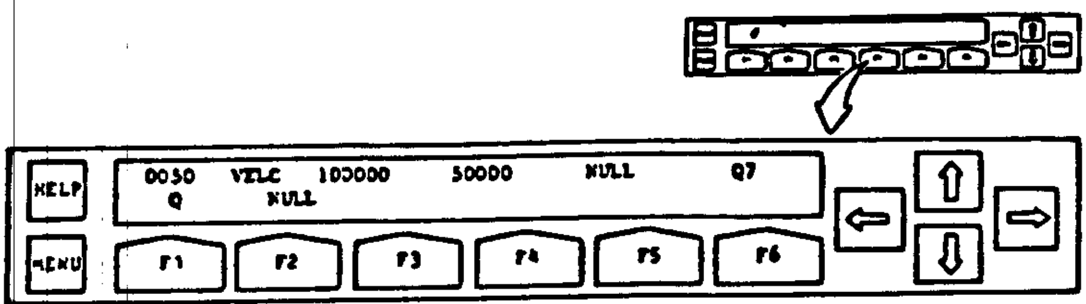


Figure 4-48: VELC Command

Note

The peak velocity for each axis will be 25000 steps/sec unless you change that value with this command. This default value corresponds to 1 rev/sec for a standard 25000-step/rev Compumotor. Default values are not multiplied by UNITS scale factors.

The example shown in Figure 4-48 sets the peak velocity for Axis 1 to 100,000 and for Axis 2 to 50,000. This number will then be multiplied by the velocity scale factor set with the UNITS command. Axis 3 will remain unchanged and Axis 4 will use the value of variable Q7 as its velocity.

4.3.5.8. ENTSTR Command

(F1, Page 6) The ENTSTR command (Figure 4-49) is used to set the absolute velocity streaming mode for Axis 1 and Axis 2 when performing Velocity Profiling and Contouring. It causes the 3000 to set the velocities on all indicated axes to zero. This command also establishes an absolute reference point in the x,y coordinates for the Contouring mode of operation. See Appendix B, Section B.3.1, Velocity Streaming, and Section B.3.2, Contouring, for examples of programs using absolute velocity streaming.

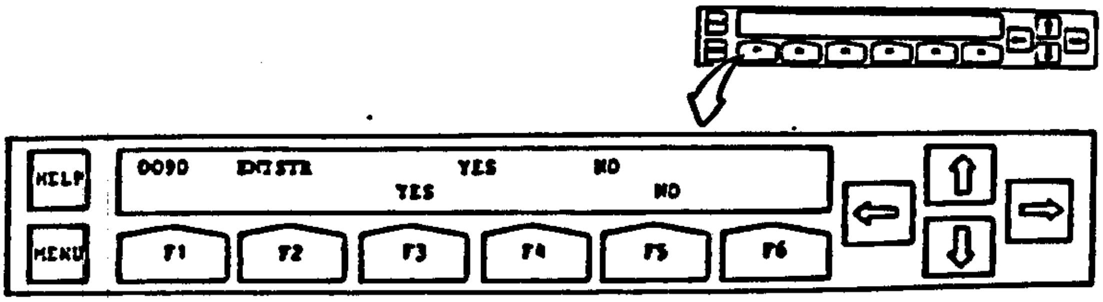


Figure 4-49: ENTSTR Command

Note

For instructions on how to use the Streaming function in your application, refer to the discussion of Velocity Profiling described in Section 4.3.5.2.

The selection of YES/NO is provided by F3 and F5. Each axis that is to begin absolute velocity streaming must be set to YES. In the example shown in Figure 4-49, Axis 1 is set up for absolute velocity streaming and Axis 2 is not.

Note

Only Axes 1 and 2 can be set up for Velocity Profiling or Contouring.

4.3.5.9. ENISTR Command

(F2, Page 6) The ENISTR command (Figure 4-50) sets Axis 1 and Axis 2 of the 3000 to the incremental velocity streaming mode. When this command is executed, the velocities of the two axes are set to zero and a relative reference point is established in the velocity-distance coordinates for Velocity Profiling and in the x,y coordinates for Contouring. Unlike the ENTSTR command, the ENISTR command does not maintain the initial coordinates' reference point as a permanent reference point. The reference coordinates always shift to the beginning point of the distance or the segment being traversed.

Incremental velocity streaming is useful in applications requiring repetitive moves, such as winding wire. The incremental approach requires fewer programming statements than the absolute approach, when programming relatively simple, repetitive operations. See Appendix B, Section B.3.1, Velocity Streaming, and Section B.3.2, Contouring, for examples of programs using these two approaches.

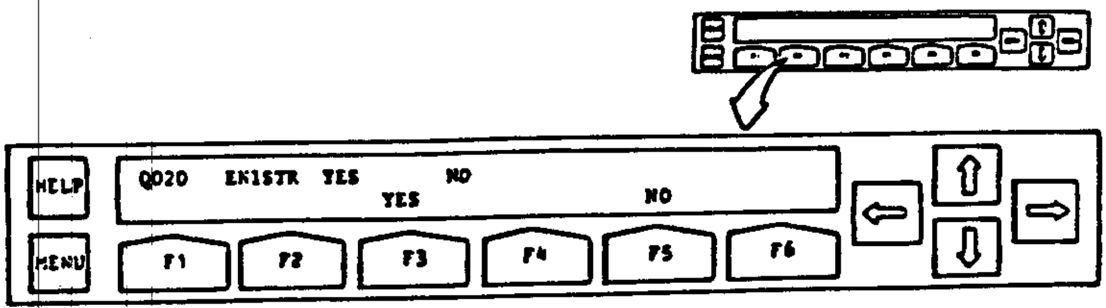


Figure 4-50: ENISTR Command

The selection of YES/NO is provided by F3 and F5. Each axis that is to begin incremental velocity streaming must be set to YES. In the example shown in Figure 4-50, Axis 1 is set up for incremental velocity streaming and Axis 2 is not.

Note

> Programming with the STRD command, and the LINE and ARC commands depends on the command used to enter velocity streaming. Appendix B shows examples of programs using the STRD, the LINE and the ARC commands when entering velocity streaming with the ENTSTR command or the ENISTR command.

4.3.5.10. STRV Command

(F4, Page 6) This command specifies the velocity at which an axis should be running when the corresponding distance specified by the STRD command is reached. STRV and STRD must always occur as a pair for profiling to work properly. Any command(s) between STRV and STRD will be ignored. The velocity values must be positive and less than 8 digits long. The format of the command is shown in Figure 4-51.

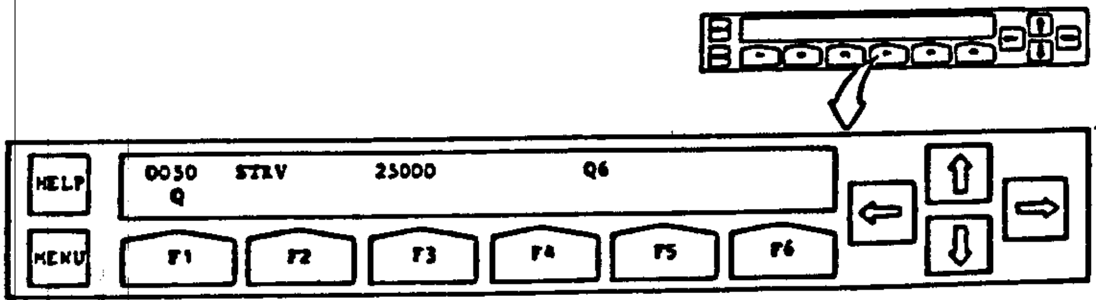


Figure 4-51: STRV Command

The STRV parameter for each axis can be a number (up to eight digits long) or a variable. If a number is desired, enter the number(s) in the appropriate position using the left and right arrow keys or the AXIS keys, then press ENTER to enter the number and increment the line number in preparation for the next command. If a variable is to be used, press F1, enter a number describing the variable then press ENTER.

The sample Display shown in Figure 4-51 calls for a velocity value of 25000 for Axis 1 and a velocity equal to the value of Q6 for Axis 2 (remember that these numbers are multiplied by the UNITS velocity scale factor).

4.3.5.11. STRD Command

(F5, Page 6) The STRD command defines the distance half of each velocity/distance pair required for profiling. The parameter may be entered as either a number or as a variable and it must be no more than eight digits in length, including the sign. Distances in profiling are absolute with respect to the point at which profiling was entered. Positive distances specified in conjunction with a velocity in the corresponding STRV command yield clockwise motions, and negative distances result in counterclockwise motion. You cannot program a profile that changes direction (e.g., from a positive distance to a negative distance) while the axis is in motion. If the previous program statement did not end with a velocity of zero, then a velocity/distance pair with a zero velocity must be entered. The format of the command is shown in Figure 4-52.

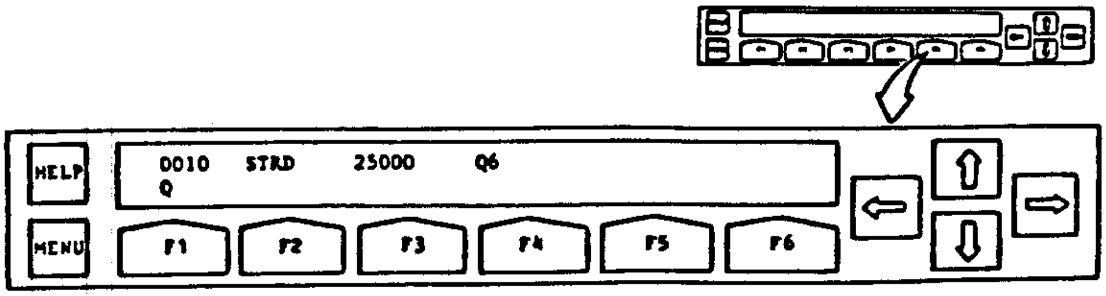


Figure 4-52: STRD Command

The distance entered for each axis can be a number (up to eight digits long) or a variable. If a number is desired, enter the number(s) in the appropriate position (using the left and right arrow keys or the AXIS keys to position the cursor), then press ENTER to enter the number and increment the line number in preparation for the next command. If a variable is to be used, press F1, enter a number describing the variable, then press ENTER.

The example shown in Figure 4-52 specifies a distance of 25000 for Axis 1 and a distance equal to the value of variable Q6 for Axis 2 (both multiplied by the UNITS distance scale factor).

4.3.5.12. STRL Command

(F3, Page 7) The STRL command specifies the feedrate at the end of a line segment. Only the ending feedrate is specified because the beginning feedrate of the next segment must always be the same as the ending feedrate of the preceeding segment. If the beginning feedrate of a segment is different from the ending feedrate of the same segment, the acceleration/deceleration values are automatically computed to adjust the velocity of Axis 1 and Axis 2 to move the load in a straight line. If the feedrate is a negative value, the sign is ignored and its absolute value is used.

If the STRL command is omitted from the program, the feedrate of the last segment traversed will be used and the line segment will be drawn at a constant velocity. The format of the command is shown in Figure 4-53.

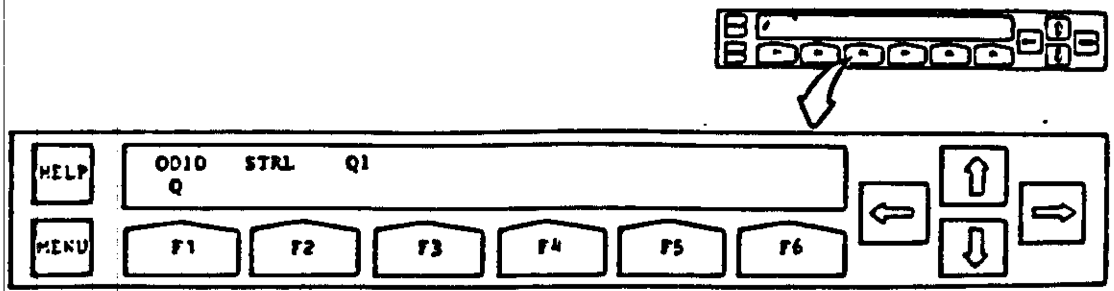


Figure 4-53: STRL Command

The STRL parameter must be either a variable defined by the MATH command or a number from 1 to 8 digits in length. If a variable is used, press F1, enter a number that will be the variable number, and then press ENTER. If a number is used, position the cursor to the appropriate location using the arrow keys, use the keypad to enter the number, and then press ENTER.

The sample Display in Figure 4-53 shows the feedrate equal to the value of the variable Q1. The value assigned to the variable or the number entered directly to indicate the feedrate are multiplied by the UNITS velocity/distance scale factor resulting in the actual feedrate for the two axes.

4.3.5.13. STRE Command

(F5, Page 7) The STRE command specifies the x,y coordinates at the end of a line or an arc segment. If the 3000 entered velocity streaming using the ENTSTR command, the x,y coordinates of the end of a segment are referenced to the origin or 0,0 of the x,y plane. If the 3000 entered velocity streaming using the ENISTR command, the x,y coordinates of the end of a segment are always referenced to the beginning point of the segment being traversed, thus computing the coordinates on a contour incrementally for each segment.

The position unit value used by the STRE command is specified with the SETUP UNITS Menu.

The format and Display example of the STRE command is shown in Figure 4-54.

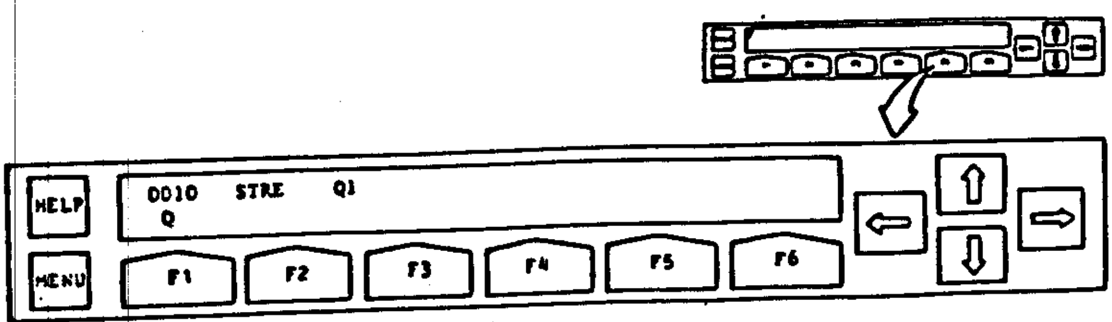


Figure 4-54: STRE Command

The STRE parameter can be specified with a number or a variable. If a variable is specified, press F1, enter the number that will be the variable number, and then press ENTER. If a number is used, position the cursor to the appropriate location using the arrow keys, use the keypad to enter the number, and then press ENTER.

4.3.5.14. STRA Command

(F2, Page 7) The STRA command specifies the feedrate at which to traverse an arc segment. The feedrate along an arc segment is always constant. Therefore, care should be taken to match the ending velocity of a line segment with the specified feedrate of a following arc segment, otherwise the discontinuity between the line velocity and the arc feedrate may cause the motor to stall.

If the feedrate is a negative value, the arc is traversed in a clockwise (CW) direction. If the feedrate is a positive value, the arc is traversed in a counterclockwise (CCW) direction.

The format of the STRA command is shown in Figure 4-55.

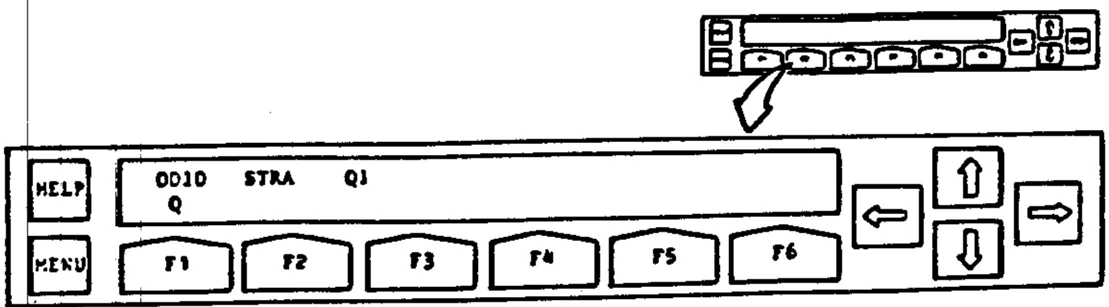


Figure 4-55: STRA Command

The STRA parameter must be either a variable defined by the MATH command or a number from 1 to 8 digits in length. If a variable is used, press F1, enter a number that will be the variable number, and then press ENTER. If a number is used, position the cursor to the appropriate location using the arrow keys, use the keypad to enter the number, and the press ENTER.

The example in Figure 4-55 displays the feedrate equal to the variable Q1, which is multiplied by the UNITS scale factor of the SETUP Menu.

4.3.5.15. STRO Command

(F4, Page 7) The STRO command specifies the x,y coordinates for the center of the circle that overlaps with an arc segment. The x-coordinate indicates the position of Axis 1 and the y-coordinate indicates the position of Axis 2 of the origin, or center, of an arc segment. The 3000 uses the origin's x,y coordinate values to compute the radius of an arc segment.

The x,y coordinates of the origin of an arc segment depend on the command used to enter velocity streaming. If the ENTSTR command is used, the value of the x,y coordinates of the origin of an arc is fixed. This origin is always referenced with respect to the 0,0 position in the x,y plane. If the ENISTR command is used, the x,y coordinate values of the origin of an arc segment depend on the x,y coordinates of the beginning point of that arc segment. For an example of arc contouring using both programming methods, see Appendix B, Section B.3.2, Contouring.

The format of the STRO command is shown in Figure 4-56.

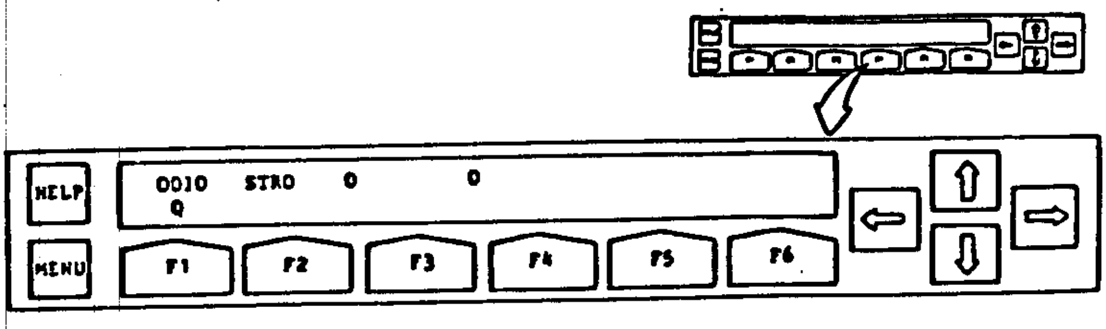


Figure 4-56: STRO Command

The STRO parameter can be represented by a variable or a number. If a variable is used, press F1, enter a number that will be the variable number, and then press ENTER. If a number is used, position the cursor to the appropriate location using the arrow keys, use the keypad to enter the number, and then press ENTER.

The example in Figure 4-56 displays the origin of the arc segment.

4.3.5.16. OUTSTR Command

(F1, Page 7) This command controls the state (0 or 1) of two TTL-compatible output bits on each axis' Auxiliary Connector. Each bit can be programmed to go low, go high, or remain unchanged when the distance specified by the velocity/distance pair is reached. This feature is very useful for synchronizing external machine functions. The following signals appear on following pins on the Auxiliary Connector:

Output	Pin
1	8
2	7

The format of the command is shown in Figure 4-57.

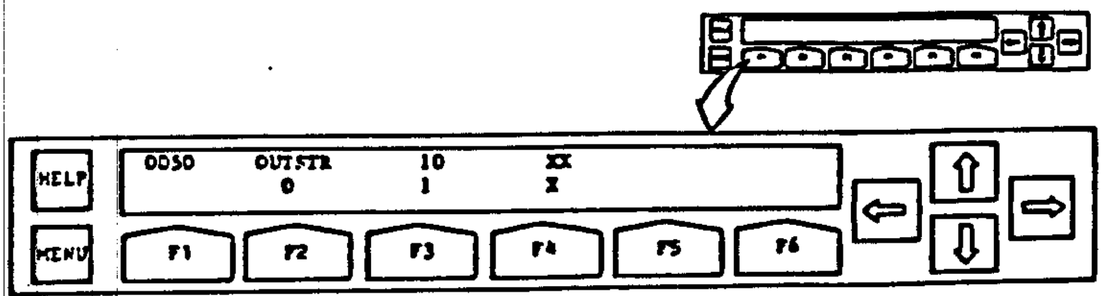


Figure 4-57: OUTSTR Command

The value entered for each axis can be any combination of 0s, 1s, or Xs. The first character specifies the desired condition of output 1; the second character refers to output 2. Entering a 0 tells the 3000 to put a 0 on that output (logical low); entering a 1 outputs a 1 (logical high); entering an X causes that output to remain unchanged.

The example shown in Figure 4-57 sets Output 1 on the Auxiliary Connector for Axis 1 to a "1", sets Output 2 on the same connector to a 0, and leaves the bits associated with Axis 2 unchanged.

Note

The OUTSTR output 2 (OUT 2) located on pin 7 of the Auxiliary Connector and the BOOST output on the Motor Drive Connector share the same 1811 TTL control line. To control OUTSTR OUT 2, install jumper JU3 and remove jumper JU4. To control the BOOST output, install jumper JU4 and remove jumper JU3. These two straps are mutually exclusive. The 3000 default strapping is with jumper JU3 installed, thus selecting OUTSTR OUT 2. Any attempt to invoke the OUTSTR OUT 2 function if the control line is strapped to select the BOOST output, will affect the BOOST function not the OUTSTR function, and vice versa.

4.3.5.17. EXSTR Command

(F3, Page 6) The EXSTR command is used to terminate Velocity Profiling. The final velocity/distance pair preceding this command must call for a zero velocity on both axes. This command has no parameters or variables associated with it. The command is of the form shown in Figure 4-58.

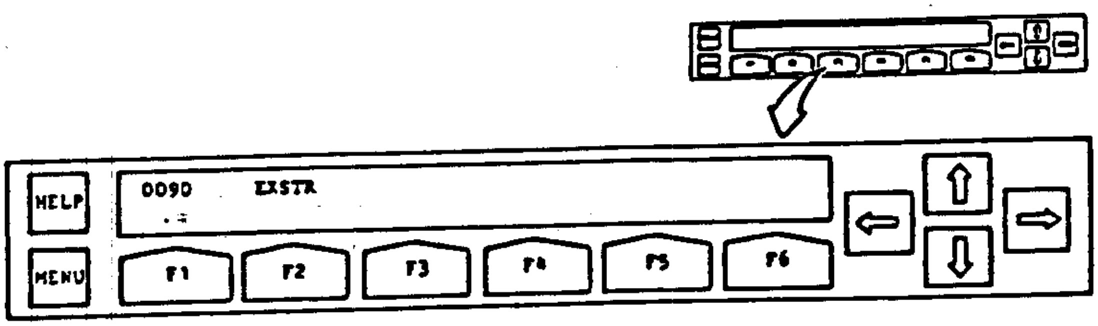


Figure 4-58: EXSTR Command

There must be one EXSTR command for each ENTSTR command. They may not be nested, but placing an entire profiling sequence within a loop (refer to the LOOP and ENDLOOP commands) permits repetitive profiling and complex profiles. .pa

4.3.6. EDIT Menu -- Motor/Drive Control

The amount of torque on a Motor can be controlled by the motor/drive using the following commands:

a. BOOST - Applies extra torque to a Motor and removes extra torque from a Motor.

b. SHTDN - Removes regular torque from a Motor and reapplies regular torque to a Motor.

4.3.6.1. BOOST Command

(F2, Page 5) The BOOST command allows you to apply extra torque to some compumotors (e.g.,P-Drives) by increasing the current to the Motor. This can be used to prevent oscillation at the end of acceleration or deceleration, to overcome slow acceleration and deceleration of a load, or any other appropriate application. The format of this command is shown in Figure 4-59.

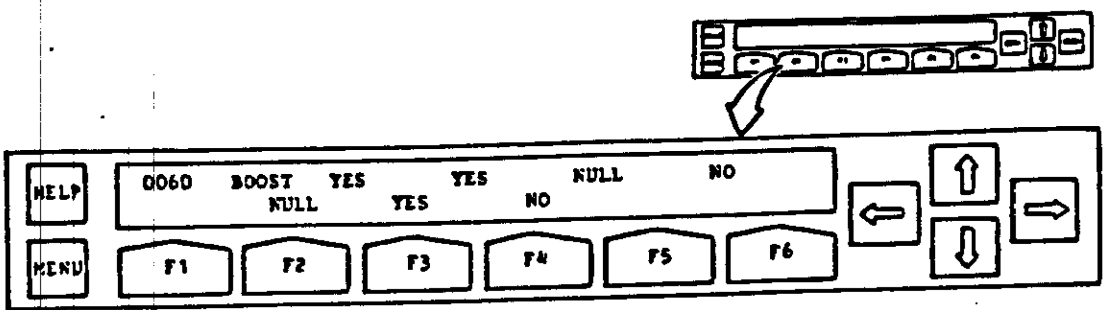


Figure 4-59: BOOST Command

The BOOST parameter must be either NULL, YES, NO, or a number from 1 to 65,535. If NULL is selected, the torque remains unchanged. If YES is selected, extra torque is applied to a Motor. If NO is selected, extra torque applied to a Motor by a previous BOOST YES command is removed from a Motor. If a number is entered, torque is automatically applied to a Motor at the beginning of acceleration or deceleration and remains applied after acceleration or deceleration for the number of milliseconds specified by this number. A BOOST command with a YES or NO disables this automatic torque.

Note

When using the YES parameter, remember that the P-Drive Motor can only have extra torque applied to it for 20 seconds and must go back to normal torque for 40 seconds before extra torque can be reapplied to it. If these limits are exceeded, the Motor will automatically go back to its normal torque.

To enter the BOOST parameter for each axis, position the cursor at the desired axis using the left and right arrow keys or the AXIS keys, press the appropriate function key or enter a number using the keypad, and then press ENTER.

The example shown in Figure 4-59 applies extra torque to Axis 1 and Axis 2, and removes extra torque from Axis 4. The torque on Axis 3 remains unchanged.

Note

The BOOST output on the Motor Drive Connector shares the same 1811 TTL control line as the OUTSTR OUT 2 output on the Auxiliary Connector. To control the BOOST output, install jumper JU4 and remove JU3. These two jumpers are mutually exclusive. The 3000 default strapping is with jumper JU3 installed selecting OUTSTR OUT 2. If the 1811 TTL control line is strapped to jumper JU3, any attempt to invoke the BOOST function will result in affecting the OUTSTR OUT 2 output.

4.3.6.2. SHTDN Command

(F3, Page 5) The SHTDN command removes regular torque from a Motor by activating the SHTDN input on the motor/drive. This can be used to remove holding torque to allow manual positioning of a load. The format of the command is shown in Figure 4-60.

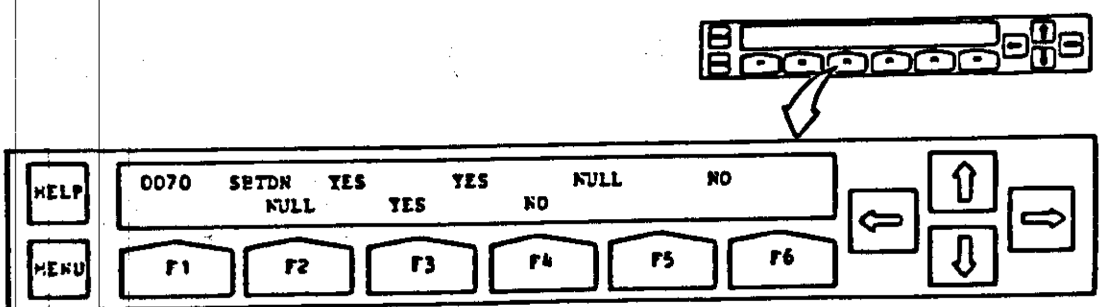


Figure 4-60: SHTDN Command

The SHTDN parameter must be either NULL, YES or NO. If NULL is selected, the torque remains the same. If YES is selected, regular torque is removed from a Motor. If NO is selected, regular torque removed from a Motor by a previous SHTDN YES command is reapplied to the Motor.

To enter the SHTDN parameter for each axis, position the cursor at the desired axis using the left and right arrow keys or the AXIS keys, press the appropriate function key or enter a number using the keypad, and then press ENTER.

The example shown in Figure 4-60 removes regular torque from Axis 1 and Axis 2, and reapplies regular torque to Axis 4. The torque on Axis 3 remains unchanged.

Note

Torque can also be removed from a Motor by a device connected to the Auxiliary Connector or a Joystick connected to the Joystick Connector. This is done via the SHUTDOWN inputs on these connectors. The SHTDN YES command disables these SHUTDOWN inputs preventing torque from being removed by these devices. The SHTDN NO command re-enables these SHUTDOWN inputs allowing torque to be removed by these devices. Torque can never be removed from a Motor from any source while the motor is running, and a Motor cannot be moved if torque has been removed from it by any source.

4.3.7. EDIT Menu -- Program Control

The 3000 executes the statements or commands in the program sequentially. The first program statement (the one with the lowest line number) is normally executed first, followed by the program statement with the next highest line number, etc. Each command is executed in turn until all have finished, then the system awaits further instruction. This simple program structure is adequate for many tasks.

Many applications however, are repetitive, require synchronization with some external event or need to respond to different situations with different actions. These applications require program control functions that allow the sequence of command execution to be more complex than the simple situation described above.

Program control commands alter the flow of command execution in a variety of ways. The loop command instructs the machine to perform a sequence of commands a number of times -- perhaps indefinitely. Subroutines provide a neat and orderly way to "call up" a frequently used set of instructions. Branch commands cause the program to jump to a specific place if a series of external conditions is matched. Trigger functions cause the 3000 to halt program execution until a specific external bit pattern is sensed. The DWELL command causes the 3000 to pause for a predetermined amount of time. These commands allow relatively small yet very powerful programs to be written with considerable ease.

The 3000 offers the following program control commands:

a. DWELL - Causes the 3000 to pause for a programmed number of milliseconds before executing the next command.

b. LOOP - Defines the beginning of a group of commands to be performed a number of times.

c. ENDLOOP - Defines the end of a group of commands in a loop.

d. BRANCH - Jumps to a specific program statement if a specific bit pattern is present on the OPTO-22 compatible inputs. The bit pattern is examined once when this command is executed.

e. ONTRIG - Jumps to a specific program statement if a specific bit pattern is present on the OPTO-22 compatible inputs. The bit pattern is examined every 100 milliseconds and between commands.

f. IF THEN - Compares two variable and, if the result is true, the program branches to a line number specified by the command.

g. TRIG - Causes the 3000 to pause program execution until a specific bit pattern is present on the OPTO-22 compatible input lines.

h. GOSUB - Instructs the 3000 to jump to a subroutine that contains a series of frequently used instructions.

i. RETURN - Returns from a subroutine to the main program.

j. GOTO - Causes the 3000 to jump to a specific line number.

k. WAIT - Causes the 3000 to pause until the specified axes have finished their moves before proceeding to next command.

4.3.7.1. DWELL Command

(F5, Page 4) The DWELL command causes the 3000 to wait for a specified amount of time before executing the next statement in the program. The format for this command is shown in Figure 4-61.

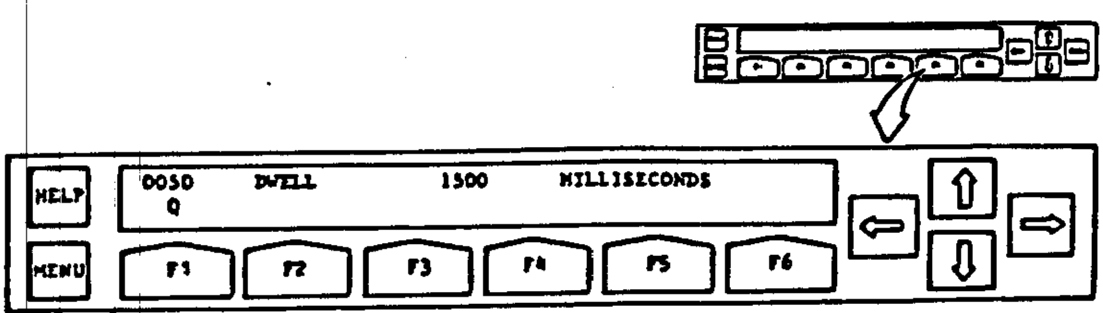


Figure 4-61: DWELL Command

In this example, the 3000 is to wait 1500 milliseconds before continuing program execution. This number can be entered from the keypad or can be a variable. Pressing F1 followed by a number and ENTER causes this command to use the specified variable. If you enter the number from the keypad, it must be less than 1,073,741,824. The time delay will not start until the instructions for any previous line statement are complete. For example, if a Preset Move is being performed in the previous line, the Motor must complete that move before the time delay will begin.

4.3.7.2. LOOP Command

(F1, Page 2) The 3000 will repeat the instructions that occur between the LOOP and ENDLOOP statements a specified number of times. The LOOP and ENDLOOP commands must always be used in pairs. The 3000 allows the loops to be nested. That is, you can have a loop within a loop, nested to eight levels. The format is shown in Figure 4-62, where nnnnn is the number of times to repeat the statements within the loop.

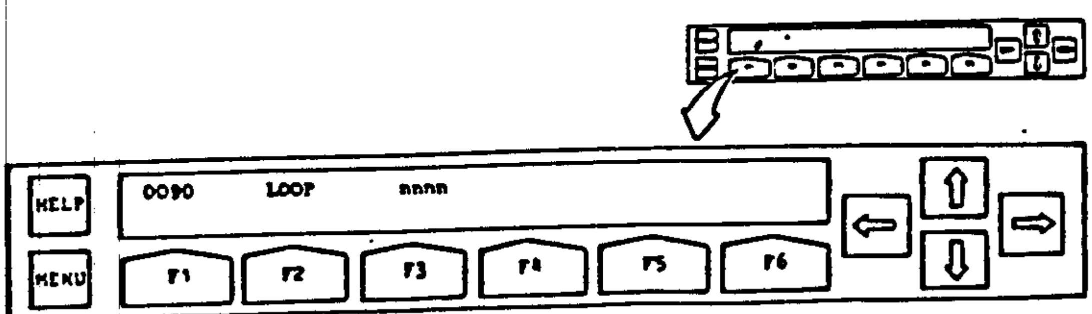


Figure 4-62: LOOP Command

The number nnnnn cannot exceed 65535. If nnnnn equals zero, the 3000 will repeat the group of statements forever. A variable (Q1-Q99) may be used.

------------------------------START EXAMPLE------------------------------

Example:

```
0100LOOP      10
0110MOVE      25000     0    0    0
0120DWELL     1000      MILLISECONDS
0130ENDLOOP
0140OUTPUT    XXX  XXXX XXXX XXX1
0150DONE
```

------------------------------END EXAMPLE------------------------------

In the example shown above, a 25000-step/rev Motor connected to Axis 1 will move one revolution and then will wait for one second. This sequence will be repeated 10 times; then the 3000 will set Output Bit 16, leaving the state of all other output bits unchanged.

4.3.7.3. ENDLOOP Command

(F2, Page 2) The LOOP and ENDLOOP commands are always used in pairs. See Section 4.3.7.2 for an example of how to use this command at the end of a loop.

4.3.7.4. BRANCH Command

(F4, Page 2) The BRANCH command allows you to examine the state of the programmable interface bits and execute one of two different program statements based on the state of those bits. If the bit pattern matches the pattern in the BRANCH statement, the 3000 will then jump to the statement number specified on the BRANCH statement and continue program execution beginning with that statement. If the pattern does not match, the 3000 will execute the next statement in the program. The bit pattern is examined once when this command is executed. An example of the BRANCH command is shown in Figure 4-63.

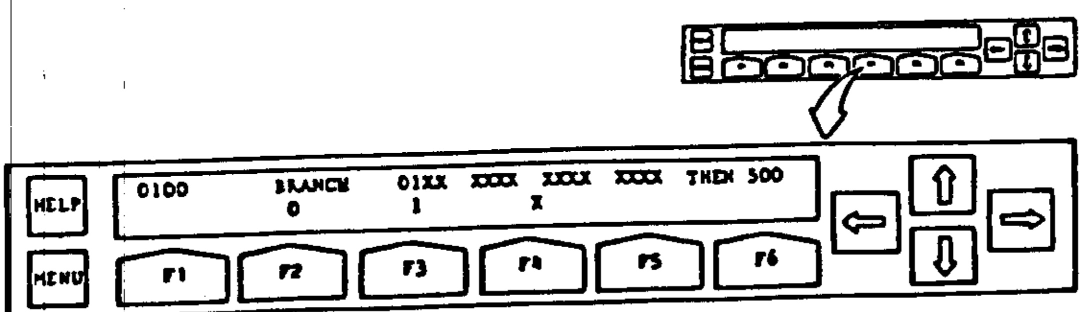


Figure 4-63: BRANCH Command

The four groups of digits following the word BRANCH represent the 16 input bits of the programmable interface, numbered 1-16 from left to right. The number to the right of the THEN statement is the line number the program will jump to if the input state matches the command. Zeros represent logic 1(high) inputs and are specified by pressing function key F2 when the cursor is under the desired bit location. Ones represent logic 0 (low) and are selected with F3. Xs are "don't cares," and no comparison will take place on bits marked with an "X" (selected with F4).

In the example shown in Figure 4-63, if Trigger Bit 1 is OFF (high) and Trigger Bit 2 is ON (low), the 3000 will go to statement number 5000. If those two trigger bits are in any other state, the 3000 will execute the statement following number 0100. The condition of Bits 3-16 are ignored in this example.

4.3.7.5. ONTRIG Command

(F4, Page 5) The ONTRIG command allows you to continuously examine the state of the programmable interface bits and, if the bit pattern matches the pattern in the ONTRIG command, the 3000 will jump to the statement number specified in the ONTRIG command and continue program execution beginning with that statement.

The 3000 begins to check the bit pattern as soon as the ONTRIG command is executed and continues examining the bit pattern every 100 milliseconds and between the execution of commands. The bit pattern is checked until one of the following conditions is met: the bit pattern is found, the program terminates, or another ONTRIG command with a different bit pattern is encountered in the program stream. In the last case, the 3000 starts looking for the new bit pattern.

Note

> When the bit pattern is found, any command in progress is completed before branching occurs, except for the DWELL, TRIG, MOVE, HOME, and WAIT commands. The 3000 will abort the DWELL and TRIG commands and then branch, but will branch while the MOVE, HOME, and WAIT commands are still in progress.

An example of the ONTRIG command is shown in Figure 4-64.

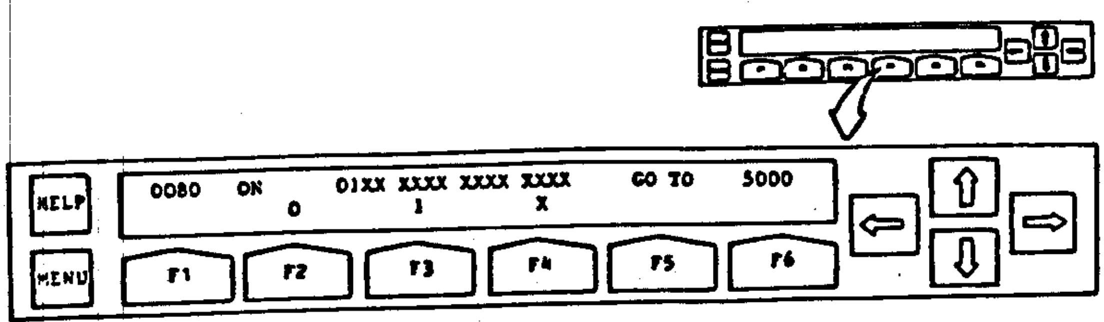


Figure 4-64: ONTRIG Command

The four groups of digits following the word ONTRIG represent the 16 input bits of the programmable interface, numbered 1-16 from left to right. The number to the right of the GOTO statement is the line number the program will jump to if the input state matches the command. Zeros represent logic 1 (high) inputs and are specified by pressing function key F2 when the cursor is under the desired bit location. Ones represent logic 0 (low) and are selected with function key F3. Xs are "don't cares", and no comparison will take place on bits marked with an "X" (selected with function key F4).

In the example shown in Figure 4-64, if Trigger Bit 1 is OFF and Trigger Bit 2 is ON, the 3000 will interrupt whatever it is doing to go to statement number 5000. If those two trigger bits are in any other state, the 3000 will continue the execution of the program. The condition of Bits 3-16 are ignored in this example.

4.3.7.6. IF THEN Command

(F3, Page 8) The IF THEN command (Figure 4-65) compares two variables and, based on the result of this comparison, will either jump to a new line number in the program or continue executing the program sequentially. If the comparison is true, the execution of the program continues with the line number found to the right of the IF THEN command statement. If the comparison is false, the program execution continues sequentially. The format of this command is:

IF Qn operator Qm THEN nbr

Where:

Qn, Qm	Any 2 of the 99 Q variables, or an integer may be used instead of the Qm variable.
nbr	The statement number in the program to be executed if the comparison is true.
operators	
=	If Qn is equal to Qm, go to statement "nbr"; otherwise, execute the next statement.
<>	If Qn is not equal to Qm, go to statement "nbr"; otherwise, execute the next statement.
<	If Qn is less than Qm, go to statement "nbr"; otherwise, execute the next statement.
>	If Qn is greater than Qm, go to statement "nbr"; otherwise, execute the next statement.
<=	If Qn is less than or equal to Qm, go to statement "nbr"; otherwise, execute the next statement.
>=	If Qn is greater than or equal to Qm, go to statement "nbr"; otherwise, execute the next statement.
OR	If the logical OR of Qn and Qm results in a non-zero value, go to statement "nbr". If the result is zero, execute the next statement.
AND	If the logical AND of Qn and Qm results in a non-zero value, go to statement "nbr". If the result is zero, execute the next statement.

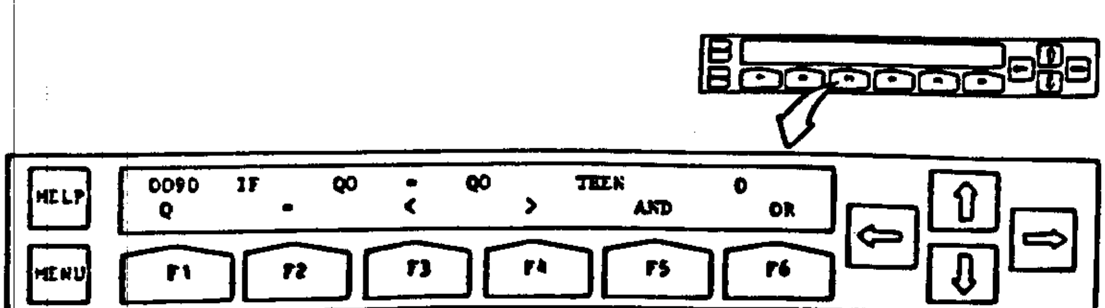


Figure 4-65: IF THEN Command

In the following example, Axis 1 makes a 25000 step move five times and, when the IF THEN statement becomes true, the operation stops.

-----------------------------START EXAMPLE-----------------------------

```
0100MATH  Q1 = 5
0110MATH  Q2 = 0
0120IF    Q2 = Q1 THEN 300
0130MOVE  25000
0140MATH  Q2 = Q2 + 1
0150GOTO  0120
0300DONE
```

-----------------------------END EXAMPLE-----------------------------

4.3.7.7. TRIG Command

(F3, Page 2) The TRIG command (Figure 4-66) instructs the 3000 to wait until the bit pattern on the 16-bit programmable interface matches the pattern set in the TRIG Menu. It will wait forever until this condition is met before executing the next line of program. Refer to Section 4.3.7.4 for details on setting the comparison condition for each bit.

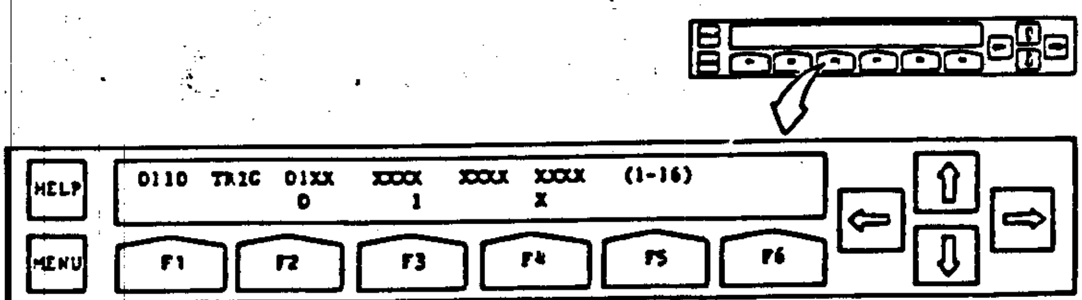


Figure 4-66: TRIG Command

-----------------------------START EXAMPLE-----------------------------

```
0100MOVE  10000    0    0    0
0110TRIG  01XX  XXXX XXXX XXX1
0120MOVE  25000    0    0    0
0130DONE
```

-----------------------------END EXAMPLE-----------------------------

In this example, Axis 1 completes a move of 10000 steps then, waits until Input Bit 1 goes high and Inputs 2 and 16 go low. Then the 3000 will move Axis 1 25000 steps. The state of Input Bits 2-15 will not be examined.

4.3.7.8. GOSUB Command

(F1, Page 3) This command causes the 3000 to jump to a subroutine. When the statement is executed, the 3000 remembers where it is in the program and then jumps to the statement number specified in the GOSUB command. When the 3000 encounters a RETURN statement, it will come back to the next statement after the GOSUB. GOSUB statements may be nested; that is, you can GOSUB to a subroutine, which in turn does a GOSUB to another subroutine, etc., to a depth of eight levels. Once a subroutine is defined, it can be called with the GOSUB command as many times as needed. The format for this command is shown in Figure 4-67, where nnnn is the first statement number in the subroutine.

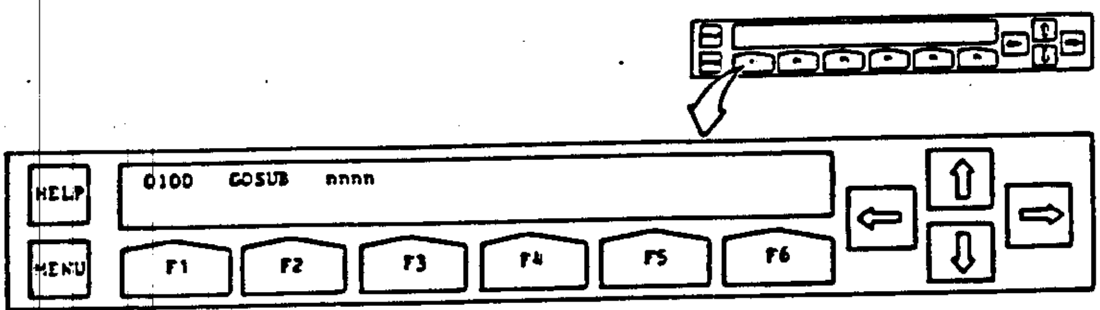


Figure 4-67: GOSUB Command

```
-----------------------------START EXAMPLE-----------------------------

0100GOSUB      5000
0110MOVE       1000      0     0     0
0120DONE
  .
  .
5000MOVE        0       25000  0     0
5010DWELL      1500 MILLISECONDS
5020RETURN

------------------------------END EXAMPLE------------------------------
```

When the 3000 executes the program shown in the example above, it will go from line number 0100 to line number 5000, which performs a 25000-step move on Axis 2, wait 1.5 seconds, and then return to line number 0110 to perform a 1000-step move on Axis 1.

4.3.7.9. RETURN Command

(F2, Page 3) This command is used in conjunction with the GOSUB command. It causes the 3000 to return to the program statement following the GOSUB statement that caused the program to move to the subroutine. Refer to the Section 4.3.7.8 for an example of how this command is used.

4.3.7.10. GOTO Command

(F3, Page 3) The GOTO command causes the 3000 to jump program execution to the program statement specified in the command and begin executing instructions starting with that line number. Either a numeric value or a numeric Q variable may be used to specify the program statement number. The format for this command is shown in Figure 4-68.

If a 0 is specified, the program execution will jump to the beginning of the program. If a non-existent program statement number is specified, the program will abort. If a negative program statement number is specified by the Q variable, the program will also abort. A negative program statement number cannot be entered directly.

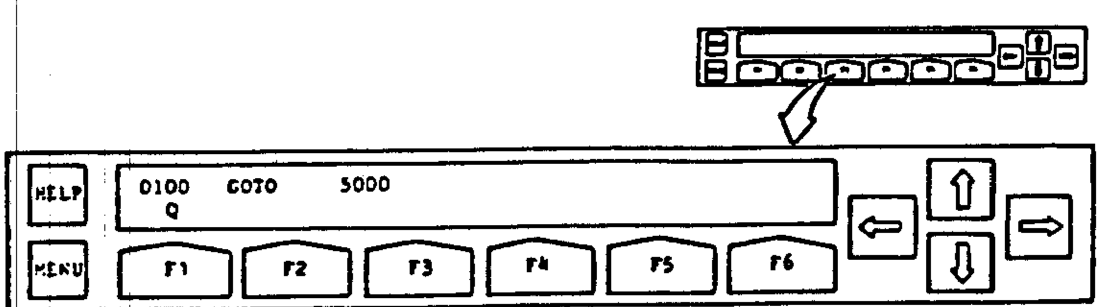


Figure 4-68: GOTO Command

The example shown would cause the program, upon hitting line 0100, to jump directly to line number 5000 and begin executing at that line. Program lines between 0100 and 5000 would not be executed.

4.3.7.11. WAIT Command

(F5, Page 3) The WAIT function instructs the 3000 to wait for specific axes to have completed their Preset Moves or to have completed acceleration or deceleration of Continuous Moves. Each axis may be specified to wait or not wait individually. The next program statement will not be executed until all axes specified to WAIT have finished. This command is intended to be used in conjunction with the MOVI command to allow flexibility in sequences of moves on multiple axes. The WAIT command is shown in Figure 4-69.

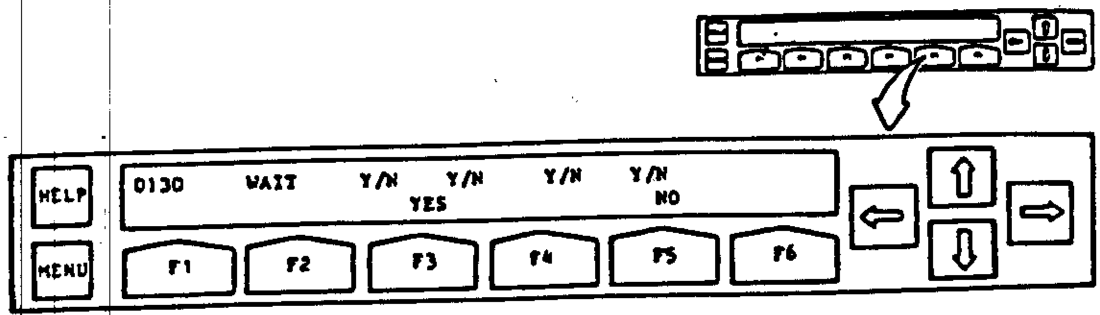


Figure 4-69: WAIT Command

In Figure 4-69, y/n is one of the following:

n -- NO, do not WAIT on this axis
y -- YES, WAIT for this axis to finish

To specify the conditions for each axis, move the cursor with the left and right arrows to the location corresponding to the axis and press F3 or F5 followed by ENTER.

----------------------------START EXAMPLE----------------------------

0010MOVI	25000	50000	0	0
0020WAIT	YES	NO	NO	NO
0030MOVI	25000	NULL	0	0
0040WAIT	YES	YES	NO	NO
0050MOVE	-36000	22525	0	0

----------------------------END EXAMPLE----------------------------

In this example Axes 1 and 2 both begin Preset Moves of different lengths. The 3000 will wait for Axis 1 to finish its first move and will start Axis 1 in its second move before Axis 2 has finished its first move. The 3000 then waits for both axes to finish before starting the final pair of moves.

4.3.8. EDIT Menu -- Position/Coordinate Reference

The 3000 has one command that you can use to select absolute or incremental positioning of each axis and two commands that you can use to establish the zero location for axes that are being used in absolute mode (refer to the UNITS command in the SETUP Menu). Axes being used in absolute mode reference all positions to a zero position. Preset Moves in absolute mode move to the command position, which is relative to zero. This is different from incremental positioning, where the distance command specifies the amount to move.

The three referencing commands are:

a. MODE - Set absolute or incremental positioning.

b. HOME - Search for the HOME input active position.

c. PDEF - Set the current absolute position. This command can be used to declare the current motor position to be zero (position = 0) or it can be used to declare the current position to be some nonzero position. This latter feature is useful for specifying tool offsets, etc.

The HOME command can also be used when programming in incremental mode to establish a mechanical reference. This is usually done once at the beginning of a program.

4.3.8.1. MODE Command

(F2, Page 5) The MODE command allows you to select either absolute or incremental positioning for each axis at any point in the program. The choice of absolute or incremental positioning affects the meaning of the distance specified for the MOVE and MOVI commands in the EDIT Menu. For axes where incremental positioning is specified, the distance on the MOVE or MOVI parameter line is the distance and direction to move away from the current position. For axes in which absolute positioning is specified, the distance on the MOVE and MOVI parameter line is the position relative to the zero reference position established on power-up or by the PZERO or PDEF functions.

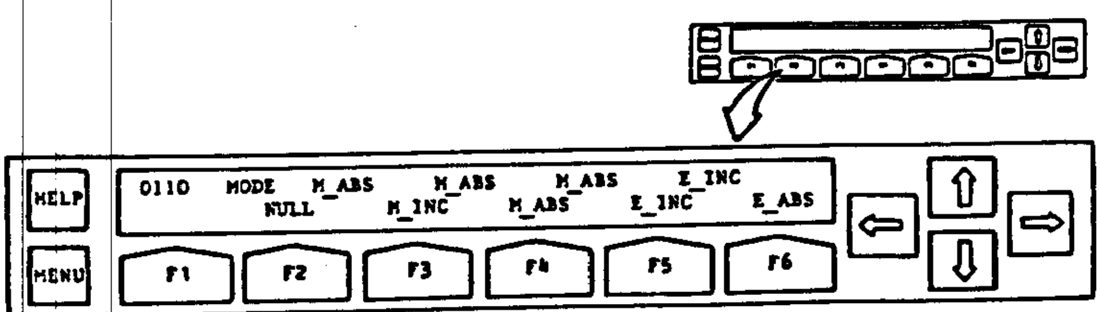


Figure 4-70: MODE Command

The MODE command also determines whether distance units (before multiplication by each axis' UNITS multiplier) are to be specified as motor steps or encoder steps. Encoder steps may be specified as the distance unit for 3000s equipped with the Encoder Interface (-P option). If distance units of encoder steps are selected with the MODE command, remember that the 3000's quadrature detector multiplies the Encoder's standard line rate by 4, so a 1000-line encoder produces 4000 steps/rev when used with a 3000.

The descriptors on the second line of Figure 4-70 indicate the following:

Label	Description
NULL	No change
M_INC	Motor steps, incremental positioning
M_ABS	Motor steps, absolute positioning
E_INC	Encoder steps (after quadrature), incremental positioning
E_ABS	Encoder steps (after quadrature), absolute positioning

To enter the MODE parameters for each axis, position the cursor at the desired axis using the left and right arrow keys or the AXIS keys, press the appropriate function key, and then press ENTER.

Figure 4-70 shows a setup that specifies that Axes 1, 2, and 3 in the machine are configured open loop, with distance units of Motor steps (before multiplication by the UNITS multiplier) and absolute positioning relative to each axis' HOME position. Axis 4, equipped with an Encoder, will interpret all distance entries to be Encoder counts after quadrature detect and will operate with an incremental coordinate system.

4.3.8.2. HOME Command

(F4, Page 3) The HOME command allows you to program the 3000 to search individually for the HOME input active position for each axis. Those axes directed to search for HOME will begin moving in the specified direction at the velocity most recently specified in the VELC command. When the HOME input becomes active, the axis will decelerate and backtrack if necessary to arrive at the HOME position. The HOME position is a function of the active level, active edge, final approach direction, and final approach velocity. All of these parameters can be defined using the SETUP HOME function.
The format for the HOME command is shown in Figure 4-71. Each axis in the HOME command has three possible entries:

a. NO - do not do the HOME move
b. YES- - start HOME in the "-" direction
c. YES+ - start HOME in the "+" direction.

The "+" direction refers to clockwise shaft rotation when the Motor is viewed from the mounting flange end.

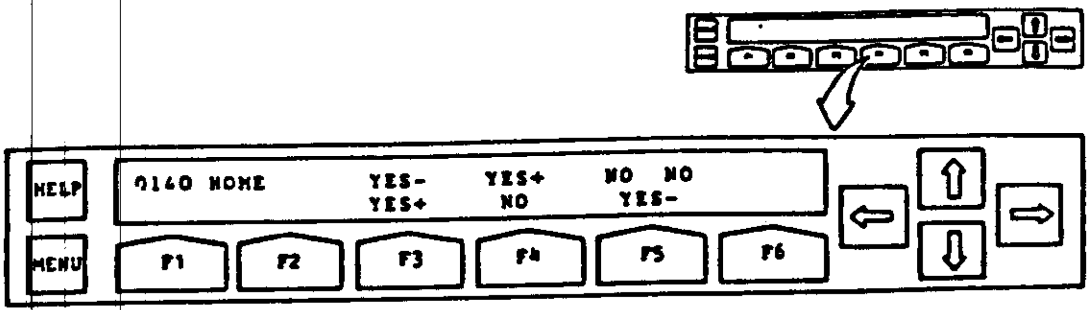
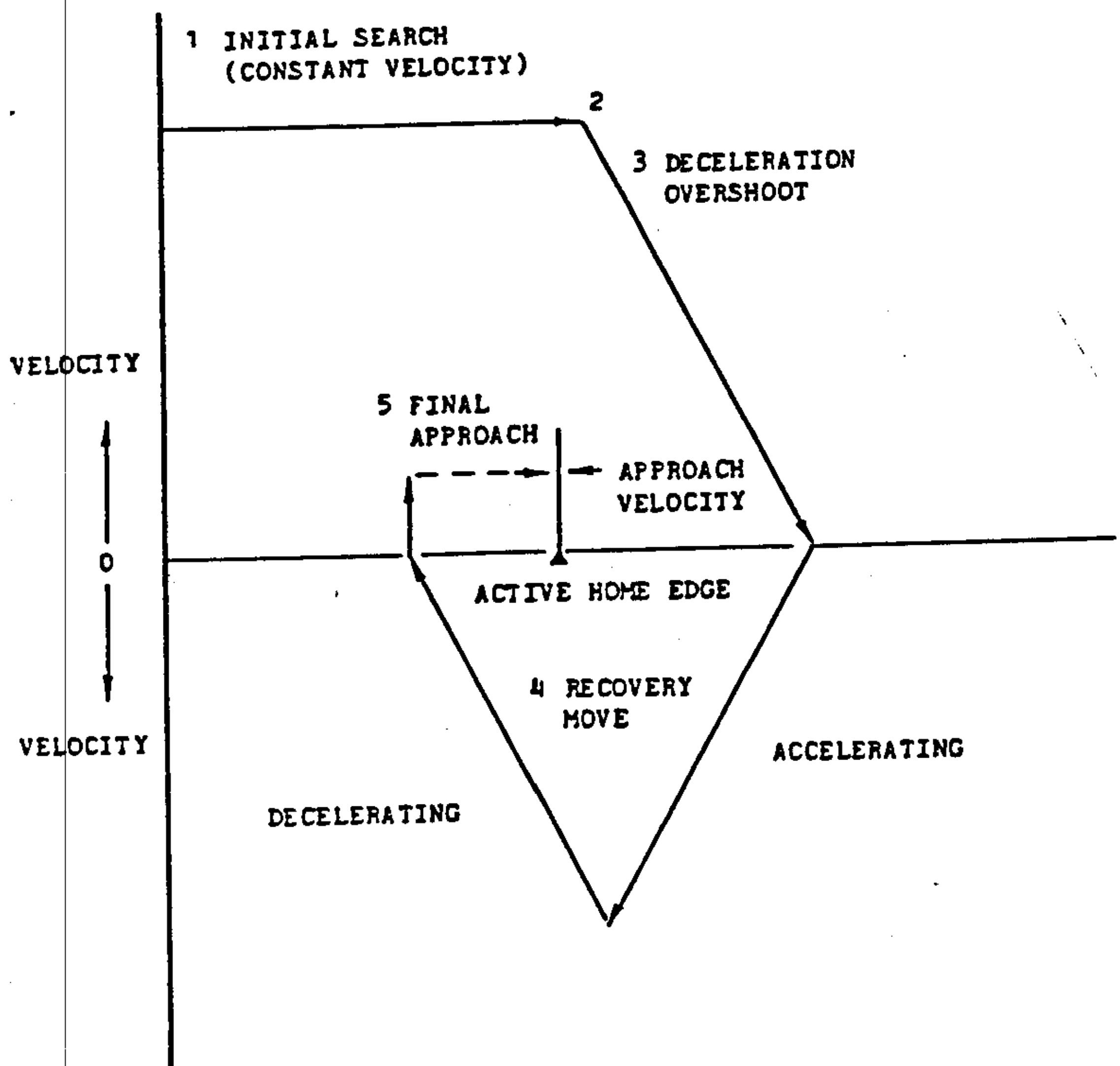


Figure 4-71: HOME Command

In the example shown in Figure 4-71, Axis 1 will begin the HOME move in the counterclockwise direction and Axis 2 will begin motion in the clockwise direction. Both Axes 1 and 2 will start at the same time. Axes 3 and 4 will not move HOME. The next statement in the program will not be executed until Axes 1 and 2 have either successfully found the HOME inputs or encountered both "+" (CW) and "-" (CCW) limits.

Figure 4-72 shows the Homing cycle starting initial search for the HOME input (1). When the active edge of interest is found (2) either at constant velocity or during acceleration, the 3000 will begin to decelerate the Motor. This results in overshoot due to deceleration (3). The 3000 notes the distance it has overshot and will perform a recovery move (4) to get back to the approximate location of the HOME position. Then the final approach (5) to the HOME position will be performed.

Figure 4-72: Homing Cycle



4.3.8.3. PDEF Command

(F3, Page 4) The PDEF command (Figure 4-73) is used to define the current motor position as a given absolute position. PDEF assigns an absolute position value to the present position. Each axis may be given a different value, a number or a variable. Any variable or number will be multiplied by the position scale factor specified with the UNITS command. Any axis may have the NULL entry, which indicates that it is left alone (not redefined). Axes that do not have a NULL entry must be stopped when this command is being performed. If incremental positioning is being used, the PDEF command has no immediate effect. However, if the MODE command is used later, the position reference is that which is established by the last PDEF command.

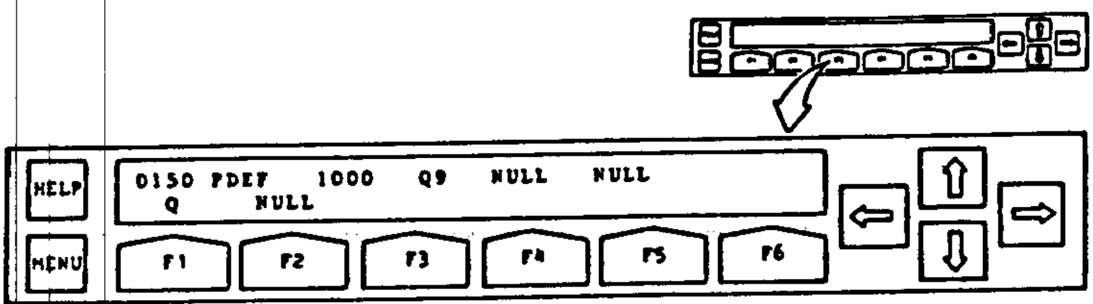


Figure 4-73: PDEF Command

The four sets of parameters define the value for the absolute position for each of the four axes. To enter a numeric value for position, enter the number on the keypad and press ENTER. To choose a variable, press function key F1 followed by the variable number (entered on the keypad).

The example in Figure 4-73 specifies that the present position of Axis 1 is to be declared absolute position 1000. The present position of Axis 2 is declared to be the value of variable Q9. Axes 3 and 4 are left undisturbed.

4.3.9. EDIT Menu -- Arithmetic/Computational

4.3.9.1. MATH Command

(F5, Page 1) The 3000 is capable of processing numbers and variables. The MATH command allows you to specify parameters with numeric variables instead of fixed numbers and perform arithmetic operations on these variables. You can use this command to dynamically alter the distance traversed, the number of times a loop is executed, the length of a dwell, etc. Numeric variables can be defined dynamically during program execution with the INPUT command (see Section 4.3.10.1) and can be scaled, offset, converted, etc. with the MATH command. The MATH command is a powerful computation function that permits the following operations:

Operator	Function
+	Addition
-	Subtraction
/	Division
*	Multiplication
AND	Boolean Multiplication
OR	Boolean Addition
XOR	Boolean Exclusive Addition
COM	Complement of a Binary Number
BIN	Conversion of Binary Coded Decimal to Binary
BCD	Conversion of Binary to Binary Coded Decimal

Up to 99 variables, labeled Q1 through Q99, are stored internally as 32-bit integers, giving each variable the range of -1 billion to +1 billion. Variables can be used by a variety of commands (MOVE, VELC, ACEL, DWELL, LOOP, STRA, STRE, STRL, STRO, STRV, STRD, IF THEN, GOTO and MOVI), but they must be defined with a MATH command prior to their use in one of those commands.

Variables may be read into the 3000 "on-the-fly" while the program is executing in the 3000 Remote Command Language.

The format for the MATH command is shown in Figure 4-74. In this example Qn is the variable Q1 through Q99, operand#1 and operand#2 are numbers or other variables, and operator is one of those listed in the above table. Press the ETC function key to display all the operators that can be used with this command. The operator and operand#2 are not required for all forms of this command.

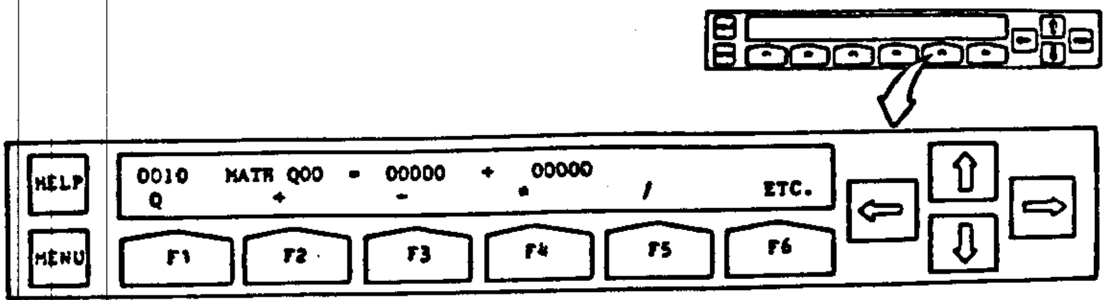


Figure 4-74: MATH Command

The operators AND, OR, and XOR perform Boolean operations on Operand#1 and Operand#2. These operations can be used with the INPUT Qn = SRC2 command to allow greater flexibility in conditional branching.

The operators COM, BIN, and BCD perform different types of conversions on Operand#1 and are used primarily with the INPUT Qn = SRC2 command and the OUTPUT Qn Qm command. Operand#2 is never present when using these operators.

COM complements all 32 bits of Operand#1. BIN converts a Binary Coded Decimal number specified by Operand#1 into binary. The number must not exceed 4 digits or 9999. If any of the 4 four-bit digits exceed 9 or if the high-order 16 bits are not zero, the Q variable is set to 0. BCD converts a binary number specified by Operand#1 into Binary Coded Decimal. If the conversion results in a number greater than 9999, the Q variable will be set to 0.

Note

There must be a space on either side of the operator for the statement to be interpreted correctly.

```
-------------------------------START EXAMPLE------------------------------
0010MATH  Q1  = 250000
0020MATH  Q6  = Q1
0030MATH  Q4  = Q1 * 500
0040MATH  Q5  = Q4 / Q6
0050MATH  Q6  = Q6 + 1000
0060MATH  Q10 = Q4 - Q1
  .
  .
0200LOOP  Q5
0210MOVE  Q1   Q4   0   0
0220DWELL Q10  MILLISECONDS
--------------------------------END EXAMPLE-------------------------------
```

Note

Numeric variables must be defined with the MATH function before they are used by the rest of the program.

4.3.9.2. Indexed Variables

The 3000 allows the use of indexed variables. The format of an indexed variable is:

QQn

Where: Q = Q

Qn = the value of the variable defined by the MATH command, which becomes the variable number of Q.

To illustrate how the 3000 treats variables and indexed variables, let's use the example below for one indexed variable. When the 3000 encounters variable Q10, it replaces this variable with 20. When the 3000 encounters indexed variable QQ10, it does the following: first it replaces Q10 with 20, thus indexed variable QQ10 becomes variable Q20; then it replaces variable Q20 with 50000, causing the axis to move 50000 counts.

-----------------------------------START EXAMPLE-----------------------------

<u>Using one indexed variable:</u>

```
0100MATH  Q10 = 20
0110MATH  Q20 = 50000
0120MOVE  QQ10
```

<u>Using several indexed variables:</u>

```
0100MATH  Q1  = 30
0110LOOP  40
0120MATH  QQ1 = 25000
0130MATH  Q1  = Q1 + 1
0140ENDLOOP
0150DONE
```

-----------------------------------END EXAMPLE-------------------------------

4.3.10. EDIT Menu -- Input/Output

These commands are used to assign values to a specified variable from various sources during program execution and, display the value of a specified variable and the status of each axis on the Front Panel Display.

a. INPUT - Assign a value to a specified variable

b. DATA - Store the values to be used by the Q variable in the INPUT Qn = SRC3 command

c. WRITE - Display the value of a specified variable on the Front Panel Display

d. DISPLAY - Displays parameters associated with the status of each axis

e. OUTPUT - Places a specified bit pattern on the programmable output connector

4.3.10.1. INPUT Command

(F2, Page 4) The INPUT command allows you to assign a value to a variable during program execution from one of several different sources. The format of the INPUT command is shown in Figure 4.75. Qn refers to the variable that will be assigned a value and SRCx refers to the source of that value.

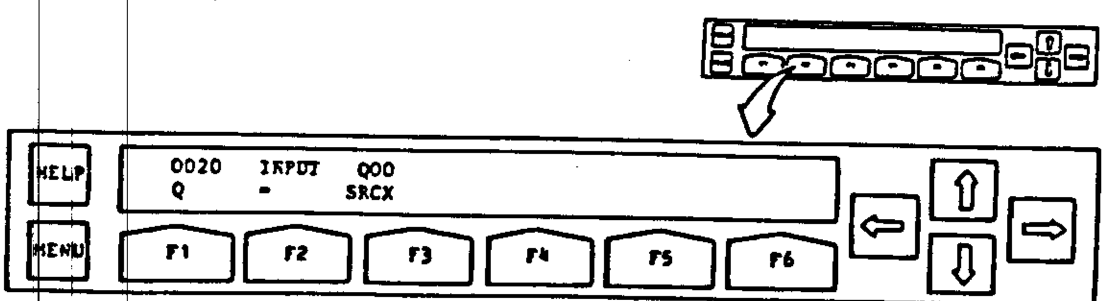


Figure 4-75: INPUT Command

The following lists all the values for SRCx and their functions:

SRC0 - Requests that the value of the variable be entered from the Front Panel.

SRC1 - Requests that the value of the variable be entered from the remote terminal or computer.

SRC2 - Sets the input 16-bits of the programmable interface as the value of the variable.

SRC3 - Sets the next value specified in the current DATA command as the value of the variable.

SRC4 - Resets the DATA command pointer to the beginning of the first DATA command, but does not change the value of the variable.

SRC5x - Sets the last byte of the two-byte response of the 1811 Indexer 88 command of the axis specified by x as the 8 low-order bits of the variable.

SRC6x - Sets the current position of the axis specified by x as the value of the variable.

SRC7x - Sets the current velocity of the axis specified by x as the value of the variable.

SRC0 causes the 3000 to display the message ENTER DATA Qn = 0 on the Front Panel Display, which requests that you enter a value for the variable using the keypad on the Front Panel. SRC1 is the same as SRC0 except it causes the message INPUT Qn? to be displayed on the remote terminal or computer and expects the response to be entered from one of these devices. String variables may be used with both of these INPUT commands to explain what the variable represents. The following statement will cause the message SAMPLE NUMBER? instead of ENTER DATA Q7 = 0 to be displayed on the Front Panel Display:

```
0200INPUT      Q7 = SRC0 Q100
```

Note

String variables can only be defined remotely using a terminal or computer. However, once defined, they can be used directly by the 3000.

The above example assumes Q100 has been defined remotely with the statement "VARIABLE Q100 = SAMPLE NUMBER;".

SRC2 uses the input 16-bits of the programmable interface as the value of the variable. These bits can be set by thumbwheel or other digital equipment and are in Binary Coded Decimal form. If the device sets the bits to 1 by being connected to ground, this is called a negative logic device. If the device sets the bits to 1 by being connected to VCC, this is called a positive logic device.

In order to perform any arithmetic operations with the variable, it must be in negative logic binary form. This conversion is done using the MATH command. The following example converts a negative logic Binary Coded Decimal number to binary:

```
0100INPUT Q7 - SRC2
0110MATH  Q7 - BIN
```

Before converting a positive logic Binary Coded Decimal number to binary, the complement must be taken to convert negative logic and the 16 high-order bits must be masked off. In the following example, statement 0110 complements the 16 input bits producing 32 bits, statement 0120 masks off the high-order 16 bits, and statement 0130 converts these bits to binary:

```
0100INPUT Q7 - SRC2
0110MATH  Q7 - COM
0120MATH  Q7 - Q7 AND 65535
0130MATH  Q7 - BIN
```

SRC3 causes the 3000 to get the value of the variable from the DATA command(s). Therefore, when using SRC3, at least one DATA command must be included in the program. SRC4 resets the value to be used for the variable to the first value in the first DATA command, but does not assign this value to the variable. Refer to Section 4.3.10.2, DATA Command, for a detailed discussion of the relationship between the DATA command and the INPUT command.

SRC5x requires x to be a value from 1 to 4 representing one of the four axes. The second byte of the 1811 Indexer 88 command represents the move status of the axis specified by x. The following defines each bit of this status byte:

bit 0	-> 1	the + direction limit switch is asserted
	-> 0	the + direction limit switch is not asserted
bit 1	-> 1	the - direction limit switch is asserted
	-> 0	the - direction limit switch is not asserted
bit 2	-> 1	the HOME limit switch is asserted
	-> 0	the HOME limit switch is not asserted
bit 3	-> 1	the direction of motion is - direction
	-> 0	the direction of motion is + direction
bit 4	-> 1	the motor is moving
	-> 0	the motor is not moving
bit 5	-> 1	the motor is at constant, nonzero velocity
	-> 0	the motor is not at constant, nonzero velocity
bit 6	-> 1	the motor is accelerating
	-> 0	the motor is not accelerating
bit 7	-> 1	the motor is decelerating
	-> 0	the motor is not decelerating

SRC6x and SRC7x require x to be a value from 1 to 4 representing one of the four axes. The current position and velocity in the units specified in the SETUP UNITS submenu for the specified axis is used as the value of the variable. These are the same values that the DISPLAY command displays on the Front Panel Display.

4.3.10.2. DATA Command

(F5, Page 5) The DATA command stores the values to be used by variable Qn in the INPUT Qn = SRC3 command during program execution (see Section, INPUT Command).

The DATA command may contain as many numeric constants as will fit on a line and any number of DATA commands may be used in one program. The 3000 treats the constants in all the DATA commands in a program as one continuous list of items or one continuous DATA command. Each DATA command is placed in order by its line number. When the 3000 encounters an INPUT Qn = SRC3 command during program execution, it uses the constants in the DATA commands one after another each time it encounters the Qn variable used in this INPUT command. An INPUT Qn = SRC4 command causes the 3000 to return to the first constant in the first DATA command, but does not assign this value to the variable.

The format for the DATA command is shown in Figure 4-76.

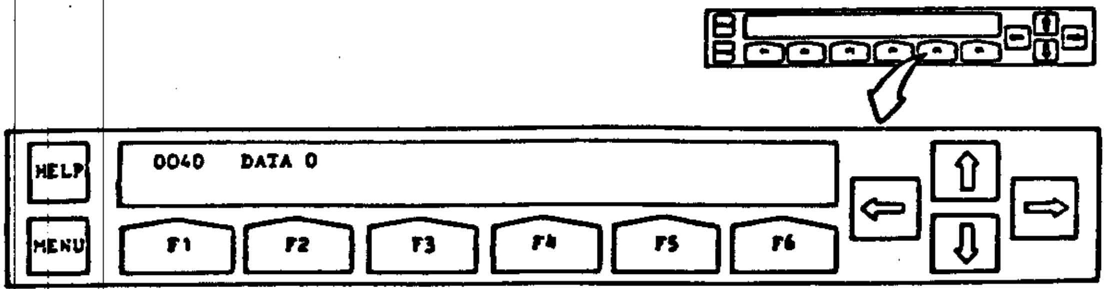


Figure 4-76: DATA Command

Each DATA parameter must be a number from 1 to 31 digits in length separated by a space. Enter each number using the keypad, separate each number with a space by pressing C/E, and then press ENTER.

The programs below show the use of the DATA command to move Axis 1 50, 44, and 56 steps, and then jump to statement 0300. The first program resets the DATA command to the first constant 50, whereas the second program does not.

--------------------------------START EXAMPLE--------------------------------

<u>Example 1:</u>

```
0100DATA 50 44 56 99

0180Q4 = 99
0200INPUT Q1 = SRC3
0205IF Q1 = Q4 THEN 300
0210MOVE Q1
0240GOTO 200
0300INPUT Q1 = SRC4
0310VELC 25000   2500
  .
  .
  .
```

<u>Example 2:</u>

```
0100DATA 50 44 56 99

0180Q4 = 99
0200INPUT Q1 = SRC3
0205IF Q1 = Q4 THEN 300
0210MOVE Q1
0240GOTO 200
0310VELC 25000   2500
  .
  .
  .
```

--------------------------------END EXAMPLE--------------------------------

Note

If the Qn variable in the INPUT Qn = SRC3 command is encountered during program execution after all the DATA command constants have been used up, the program will abort.

4.3.10.3. WRITE Command

(F1, Page 4) The WRITE command displays the value of the variable on the Front Panel Display, or the remote computer or terminal. For the Front Panel Display, you can also specify the location of the number on the Display and a region of the Display to be cleared. The format is shown in Figure 4-77, where the first parameter specifies the beginning column to be cleared and the second parameter specifies the number of columns to be cleared (there are 40 columns on the Display). The beginning column the variable will be displayed into is specified by the third parameter. If this parameter is greater than 40, the variable will be displayed on the remote terminal or computer. The variable to be displayed is specified by Q00. Each parameter must be separated by a space and the command incorporated by pressing ENTER.

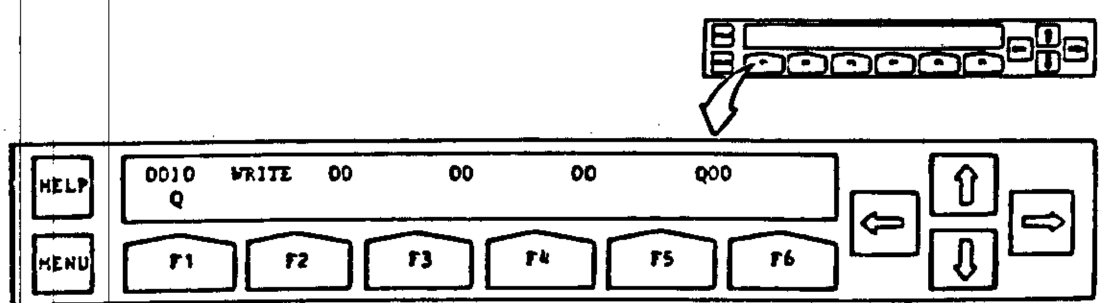


Figure 4-77: WRITE Command

Both numeric and string variables may be displayed. A string variable is left justified, and a numeric variable is right justified converting leading zeroes to blanks. When displaying variables on the Front Panel Display, "Qn =" always preceeds the variable value. When displaying variables on a terminal or computer, only the value of the variable is shown. The following example displays several variables on the Front Panel Display on the same line by executing several WRITE commands, where each command selects a different part of the Display to be written into:

--------------------------------START EXAMPLE--------------------------------

```
0020WRITE        1   20    1   Q1
0030WRITE       20   40   20   Q2
```

---------------------------------END EXAMPLE---------------------------------

The first WRITE command will clear columns 1 through 20 on the Display and display the value of Q1 starting at column 1. The second command will clear columns 20 through 40 and display Q2 starting at column 20. In this manner two variables can be displayed at once.

Note

> When using the Front Panel Display, if your starting column number does not allow for the maximum size of your variable plus the variable name (17 characters), those characters that cannot fit will be written on top of each other in the last column (column 40). Thus, all of the characters that are written over will be lost and only the last digit or character will appear in column 40.

4.3.10.4. DISPLAY Command

(F4, Page 4) This command is identical to the DISPLAY command in the Main Menu, except that in the EDIT Menu the DISPLAY command can be given a line number and placed in a program to trigger changes in the display format during program execution. Refer to Section 4.3.2.10 for a detailed discussion of the use of this command.

4.3.10.5. OUTPUT Command

(F5, Page 2) The OUTPUT command (Figure 4-78) controls the state (0 or 1) of the optically isolated output bits on the 16-bit programmable output connector. Fifteen of the sixteen output bits are programmable, the first bit (output bit 1) is hard-wired as the machine status output; it is OFF (high) when the 3000 is operating correctly and ON (low) when it is not. The status of this bit is indicated by the STATUS lamp on the Front Panel. This command is useful for triggering external events at predetermined times during program execution.

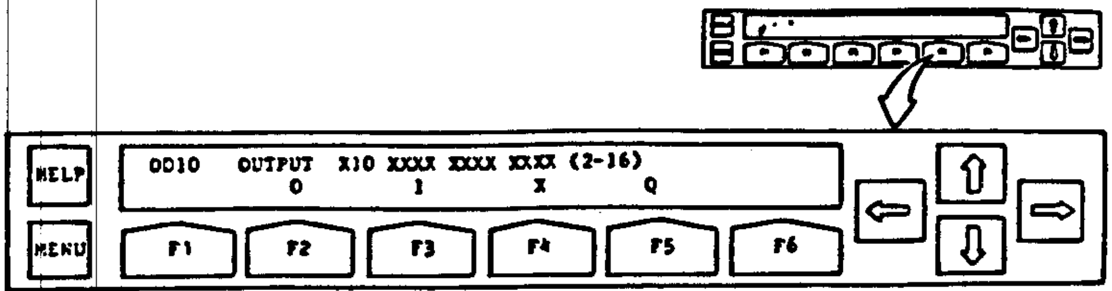


Figure 4-78: OUTPUT Command

The four groups of digits following the word OUTPUT represent the 15 programmable output bits of the programmable interface, numbered 2-16 from left to right. Zeros represent logic 1 (high) and are specified by pressing function key F2 when the cursor is under the desired bit location. Ones represent a logic 0 (low) and are selected with F3. Xs are "don't care," and bits marked with an "X" will remain unchanged from their previous state (selected with F4).

In the example shown in Figure 4-78, output bit 3 is set to 1 (ON, true) and output bit 4 is set to 0 (OFF, false). The status of the other bits will remain unchanged.

Variables can also be used to control the bit pattern of the output bits and the mask for the "don't care" bits. The format of the OUTPUT command using variables is OUTPUT Qn Qm where Qn is the bit pattern of the output bits and Qm is the mask. These variables must be defined as decimal numbers by the MATH command before referencing them in the OUTPUT command. These decimal numbers are converted to binary by the 3000 to produce the desired bit pattern and the mask. The maximum decimal number allowed is 32767.

The output bits are sent to digital equipment that expects this output to be in Binary Coded Decimal form. This equipment can be either negative logic, which sets the bits to 1 by being connected to ground, or positive logic, which sets the bits to 1 by being connected to VCC. The 3000 uses binary negative logic, therefore, these bits must first be converted to Binary Coded Decimal by the MATH command as follows:

```
0100MATH   Q7 - BCD
0110OUTPUT Q7
```

The following example outputs to a positive logic machine. Statement 0100 converts the bits to Binary Coded Decimal, statement 0110 complements the bits producing 32 bits, statement 0120 masks off the high-order 16 bits, and statement 0130 outputs the bits to the positive logic machine:

```
0100MATH   Q7 = BCD
0110MATH   Q7 = COM
0120MATH   Q7 = Q7 AND 65535
0130OUTPUT Q7
```

4.3.11. EDIT Menu -- Teach

(F1, Page 8) The 3000 has a powerful command that you can use to "teach" one or more axes a series of positions. These positions can then be duplicated, at any speed and acceleration, over and over again under program control. This function is useful for robotics, inspection, and general automation tasks. The TEACH command is used in conjunction with the AXIS#1 through AXIS#4 keys and the JOG keys to select an axis, move it to a new position, and store that position. The Joystick may also be used to move the motors during TEACH. Each axis being "taught" must be in absolute mode (refer to Sections 4.3.3.5 and 4.3.8.1) or the results will be unpredictable. The format of the TEACH command is shown in Figure 4-79.

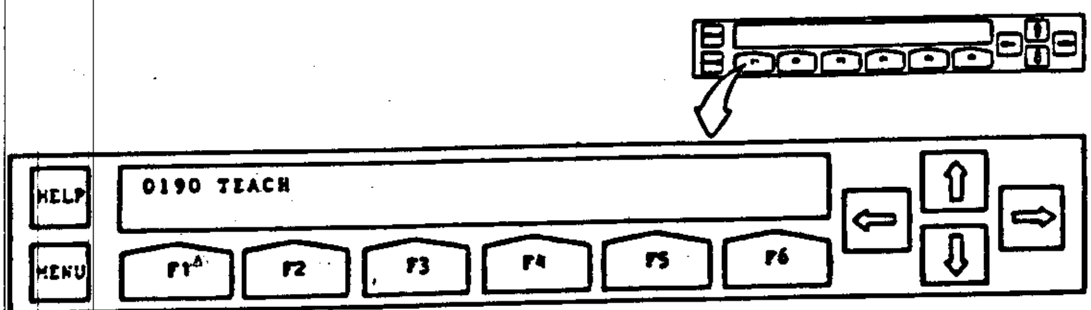


Figure 4-79: TEACH Command

To teach the 3000 a series of position:

a. Make sure each axis is in the absolute positioning mode (Sections 4.3.3.5 and 4.3.8.1).

b. Press the TEACH function key.

c. Press one of the four AXIS keys (directly above the START and STOP keys) to select an axis.

d. Using the JOG keys or the Joystick, move that axis to the desired position.

e. If desired, select another axis and repeat steps c and d to JOG it to a new position.

f. When all axes are in the desired positions, press ENTER. This causes the 3000 to generate a program line with a MOVE instruction followed by the current absolute positions for each axis.

g. Repeat steps 2 through 6 as many times as you wish to teach the 3000 a sequence of moves (limited by the available number of program statements, as shown in Table 1-2).

When the program is run, each axis will make the moves specified using the velocities and accelerations specified by any VEL and ACEL instructions preceding the MOVEs. You may then go back and EDIT the program to fine-tune the positions or to change the accelerations and velocities for each axis.

------------------------------START EXAMPLE------------------------------

Suppose the 3000 commands an X, Y table with Axis 1 controlling the X- (dimension horizontal) and Axis 2 controlling the Y- (motion vertical). A rectangular motion is desired that graphically looks like:

```
        60000
        20000
                  40000           90000
```

Press TEACH, JOG to the first point (lower left), and press ENTER to log in the first point. Press TEACH again, JOG to the upper left point, and press ENTER again to enter the second point.

Continuing around the finally returning to the first position at the lower left will generate a series of MOVEs that look like:

```
0010MOVE  20000    40000    0   0
0020MOVE  60000    40000    0   0
0030MOVE  60000    90000    0   0
0040MOVE  20000    90000    0   0
0050MOVE  20000    40000    0   0
```

------------------------------END EXAMPLE------------------------------

4.3.12. EDIT Menu -- Utilities

Three other useful commands in the EDIT Menu are used to list a line of the program in Main Memory on the Front Panel Display, delete program lines, and indicate the end of a program:

a. LIST - Display a specific line of the program on the Front Panel Display.

b. DELETE - Erase lines of the program.

c. DONE - This command indicates the end of a program. The 3000 will cease program execution when this statement is encountered.

4.3.12.1. LIST Command

(F5, Page 8) This command instructs the 3000 to display the program line specified. This command is different than others in the EDIT Menu in that it is performed immediately; it is not stored in the program as a statement. The LIST command is useful when programming, as it provides a way to refer to a previously entered command for reference. The LIST command does not alter the command listed in any way.

When LIST is invoked, the Display says LIST 0000 and is displayed the cursor is positioned at the beginning of the 0000. Enter the line number of the program line to be listed followed by ENTER. The Display then shows the indicated program line.

4.3.12.2. DELETE Command

(F4, Page 8) The DELETE command is used to erase one or more statements, or program lines, from the program. The format for this command are shown in Figure 4-80, where 0010 is the first statement to be erased and 0500 is the last statement to be erased. All commands between these two will be deleted as well. To delete only a single statement, use the keypad to enter the line number for that statement in the first field (occupied by 0010 in the example shown in Figure 4-80), leave the second field all zeros, and press ENTER.

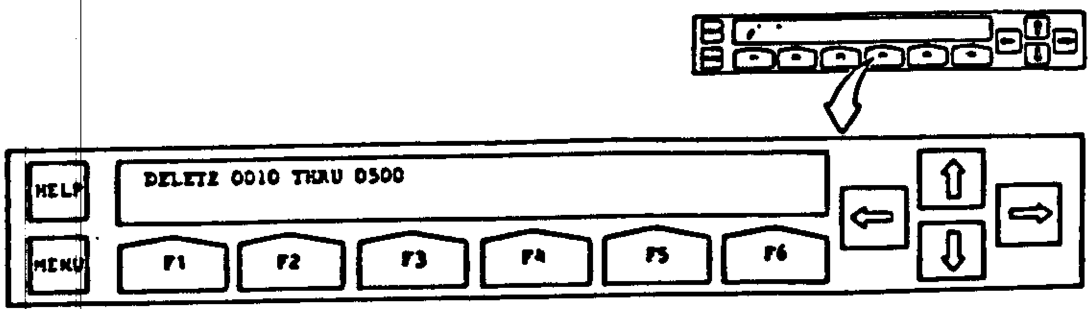


Figure 4-80: DELETE Command

4.3.12.3. DONE Command

(F2, Page 8) This command is placed at the end of a program to finish or terminate a program. More than one DONE statement can be placed in the 3000's memory, which allows different programs to coexist in memory. When the DONE command is encountered, any axes in motion will immediately decelerate to a stop. The RUN function key or the START button must be pressed in order to begin running again.

```
--------------------------START EXAMPLE--------------------------------

0010DWELL 200 MILLISECONDS
0020MOVE  2500 -350      0    0
0030DONE


0500MOVE  355  -27000    0    0
0510DONE

---------------------------END EXAMPLE--------------------------------
```

This example demonstrates the concept of two different programs or two different program endings in memory. If you enter line 0010 after pressing the RUN key, the 3000 will execute statement 10, then statement 20, then the DONE command in line 0030. If the machine is started at a line number before to line 0500 or branches to line 0500, then the DONE statement in line 0510 terminates the program.

4.4. Remote Control Operation

Remote control operation can be divided into two groups: the 3000 Remote Command Language, which allows you to implement the 3000 control functions froma terminal or computer; and the 1811 Remote Command Language, which allows you to independently control each 1811 Indexer from a terminal or computer.

The terminal or computer must be connected to the 3000's RS-232C or IEEE-488 interface. The REMOTE Submenu in the SETUP Menu is used to select the desired interface. The REMOTE Submenu also has a TEXT ON or TEXT OFF feature to select whether the 3000 displays prompts on the terminal or computer.

The interface selected with the REMOTE Submenu is always active and commands may be issued to the 3000 at any time, even when a program is running in the machine. Commands sent to the 3000 will be inserted concurrently with program execution and both program statements and variable values can be updated at any time.

Warning

If a new program is downloaded while another program is running, the 3000 will begin executing the new program.

All commands are terminated with a semicolon (;). When the semicolon (;) is sent to the 3000, it responds with a carriage return <CR> and line feed <LF> to indicate that it is awaiting input.

Backspacing with the control-H character is allowed as long as it occurs before the semicolon (;) command terminator.

4.4.1. 3000 Remote Command Language

The 3000 Remote Command Language was designed to allow you to take full advantage of the 3000 features without using the Front Panel. This includes all the MAIN Menu control functions, program creation, SETUP functions, and the definition of character strings for later display on the 3000 Front Panel Display. Table 4.3 summarizes the 3000 Remote Command Language.

Table 4-3: 3000 Remote Command Language Commands

Command	Description
HELP;	Explains the two types of remote command languages used by the 3000 and how to display the commands used with each language.
MENU;	Lists all 3000 remote language commands on the remote terminal or computer.
LOAD;	Instructs the 3000 that the following statements are program lines to be added to the program in MAIN Memory.
SETUP;	Allows machine configuration. This is similar to the SETUP Menu on the Front Panel.
START;	Tells the 3000 to begin running the program, beginning with the lowest numbered statement.
RUN FROM <num>;	Starts executing at the line number specified by <num>.
STOP;	Commands the 3000 to stop executing the program and decelerate all axes to a stop.
RESUME;	Causes the 3000 to resume executing the program that was halted by the STOP command.
SAVE;	Specifies that the program currently in Main Memory is to be saved in the unit's non-volatile EEPROM memory.
RECALL;	Instructs the 3000 to load main program memory from the EEPROM.
PZERO;	Defines the current position as absolute zero.
ACCESS <num>;	Allows access to the MAIN Menu.
PRINT <num> THRU <num>;	Causes the program statements specified to be sent over the interface.
DISPLAY;	Sets up the real-time display on the Front Panel.
REPORT;	Displays one line of status information on the remote terminal or computer. This is a "snapshot" that is identical to the contents of the Front Panel Display.

Table 4-3: 3000 Remote Command Language Commands (Cont'd)

Command	Description
VARIABLE Q<num> =<<num> or Q<num> or string>>;	Defines, changes and displays values of variables.
LCD<num>;	Displays the top or bottom line of the Front Panel Display on the remote terminal or computer.

The 3000 Remote Command Language is generally the same in function and syntax as the corresponding Front Panel functions. Each command is described below with an example. Only the first 4 letters of each command are required. The SETUP and LOAD commands are multiple lined and require additional lines to be entered below them to complete the necessary information.

4.4.1.1. HELP Command

The HELP command displays the power-up message describing the two remote command languages and shows the commands used to list command summaries for each. The following example will display the text shown below it:

```
HELP;
```

The 3000 has two types of remote command language. The 1811 command language is used to send commands directly to the specified 1811 axis. The 3000 command language is used to send 3000 commands which perform the same functions found on the 3000 Front Panel. Both command languages input one line at a time, with backspacing allowed, and lines terminated with a semicolon. Control characters are ignored. Use the following commands to display the menus.

```
HELP;    - displays this description
/M;      - displays 1811 command language menu
MENU;    - displays 3000 command language menu
```

4.4.1.2. MENU Command

The MENU command lists all the commands of the 3000 Remote Command Language. The following example will display the text shown below it:

```
    MENU;

REMOTE 3000 COMMAND LANGUAGE
    no "/" in front, all capitals, end with ";"
    command         description
    -------         -----------
    HELP;           displays language descriptions
    MENU;           display this menu
    START;          start 3000 program
    RUN FROM <num>; run 3000 from statement
    RESUME;         resume program
    SAVE;           save program in EEPROM
    RECALL;         recall program from EEPROM
    LOAD;           load program from remote
    PRINT <num>     THRU <num>; print program lines
    SETUP <func>;   setup selected function
    PZERO;          set position = 0 all axes
    ACCESS <num>;   enter access code
    STOP;           stop motors, program execution
    VARIABLE Q<num> = <num>; set/get variable value
    DISPLAY<func> <func> <func> <func>; display functions
    REPORT;         report current display values
    LCD <num>;      send current display of LCD<1 or 2>
```

4.4.1.3. START Command

The START command causes the 3000 to begin program execution starting with the lowest program statement number. The following example will start program execution:

```
START;
```

4.4.1.4. RUN Command

The RUN command allows you to begin program execution at a specified statement number. The following example will cause the 3000 to start executing the program at statement number 500 or, if 500 does not exist, at the next higher numbered statement:

```
RUN FROM 500;
```

4.4.1.5. RESUME Command

The RESUME command causes the 3000 to resume program execution that was halted by the STOP command. Any moves that were stopped by the STOP command will be completed before the next statement is executed. The following example will restart program execution:

```
RESUME;
```

4.4.1.6. SAVE Command

The SAVE command causes the 3000 to store the current program and setup parameters in EEPROM. Although the Front Panel SAVE command asks if it is "OK to SAVE?", the remote SAVE command does not. The following example will save the current program:

```
SAVE;
```

4.4.1.7. RECALL Command

The RECALL command causes the 3000 to copy the program and setup parameters stored in EEPROM into working memory. Although the Front Panel RECALL command asks if it is "OK to RECALL?", the remote RECALL command does not. The following example will copy the program and setup parameters from EEPROM into working memory:

```
RECALL;
```

4.4.1.8. PRINT Command

The PRINT command causes the 3000 to list the specified range of statements on the remote terminal or computer. Incoming commands will be buffered but not executed while the PRINT command is executing. The range specification is optional. The following example prints the entire 3000 program:

```
PRINT;
```

The next example prints all program statements from the beginning of the program up to and including program statement number 500:

```
PRINT 0 THRU 500;
```

The next example prints all program statements starting at statement number 1000 and ending with the last statement in the program:

```
PRINT 1000;
```

4.4.1.9. PZERO Command

The PZERO command sets the absolute encoder step positions and the absolute motor step positions of all axes to zero. It does not cause the motors to move. The following example sets all step positions to zero:

```
PZERO;
```

4.4.1.10. ACCESS Command

The ACCESS command asks you to enter the system password, which allows you to change the Front Panel from the POWER-UP Menu to the MAIN Menu, or vice versa. The following example uses the default password to change the Front Panel menus:

```
ACCESS 3000;
```

4.4.1.11. STOP Command

The STOP command causes the 3000 to stop all motors and program execution. The RESUME command will begin program execution where the STOP command left off. The following example stops all motors and program execution:

```
STOP;
```

4.4.1.12. VARIABLE Command

The VARIABLE command allows you to define, change, or request the value of a Q variable at any time, independent of program execution. Variables Q1 through Q99 are numeric variables and cannot exceed 11 digits. Variables Q100 through Q109 are character string variables and cannot exceed 20 characters. Spaces may be included, but multiple spaces will only produce one space. Indexed variables, described in Section 4.3.9.2, are also supported. Variable values are assigned or changed if the parameter to the right of the equals (=) sign is an integer or another variable. A variable value is reported if that parameter is a question mark (?). The 3000 will report a maximum of 11 characters, changing leading zeros to blanks.

The following are examples of assigning values to and reporting values of variables using the VARIABLE command:

Assigns the value "1234" to variable Q5:

```
VARIABLE Q5 = 1234;
```

Assigns the value of variable Q6 to Q5:

```
VARIABLE Q6 = Q5;
```

Assigns the character string "board length" to variable Q105:

```
VARIABLE Q105 = board length;
```

Reports the value of variable Q5:

```
VARIABLE Q5 = ?;

        1234
```

4.4.1.13. DISPLAY Command

The DISPLAY command allows you to display information about any axis on the Front Panel Display. See Section 4.3.2.10 for the parameters that can be displayed using this command. The following example will display the position of Axis 1, the velocity of Axis 2, the status of Axis 3 and the program line number currently being executed for Axis 4:

```
DISPLAY POS VEL STAT LNUM;
```

4.4.1.14. REPORT Command

The REPORT command causes the 3000 to display the DISPLAY command parameter values on the remote terminal or computer being used. These values will be displayed even if the Front Panel is not currently active. The following is an example of this command and what would be displayed if the above DISPLAY parameters had been requested:

```
REPORT;

+100555 25002 +CONVEL 120;
```

4.4.1.15. LCD Command

The LCD command displays the specified line of the Front Panel Display and can be issued at any time. A "1" will display the top LCD line and a "2" will display the bottom LCD line. The following example will display the top LCD line:

```
LCD 1;
```

4.4.1.16. LOAD Command

The LOAD command allows program statements to be loaded into the 3000. They are formatted the same as when programming from the Front Panel. This is a multiple-line command. After the 3000 receives the LOAD command and responds with a <CR> and <LF>, program lines may be sent. Each line must be terminated with a semicolon (;). The LOAD command is terminated by an asterisk (*) followed by a semicolon (;). The following example will load statements 0010 through 0050 into the 3000:

```
LOAD;
0010VELC    100000    25000    25000    75000;
0020ACEL    500000   500000   500000   500000;
0030MOVE       100     -350    25000    12500;
0040MOVE      -100     -350    25000        0;
0050DONE;
*;
```

4.4.1.17. SETUP Command

The SETUP command allows you to configure the 3000 to your specific application. You can use all the 3000 Front Panel SETUP functions except the RS-232 and the 488 functions. (See Section 4.3.3, SETUP Menu for a full description of the SETUP functions, their use, their submenus, and their parameters.)

The SETUP command is a multiple-line command. The format of the SETUP command is:

```
SETUP <function>;
<submenu>;
   .
   .
[ENTER;]
```

where:

function - UNITS
ACCESS
AXES
REMOTE
JOG
ENCODER
AUTORR
REMSTSP
LIMITS
HOME

submenu - the top line of the Front Panel Display

Some SETUP functions have several submenus, which are distinguished by the keyword in the upper right-hand corner of the Front Panel Display. For remote operation, the submenus can be entered in any order and the keyword can be abbreviated by entering the first two letters. The word SETUP and its functions can be abbreviated by entering the first four letters of each word. Some SETUP functions require the word "ENTER;" as the last line. Each SETUP function is described below with examples of all the possible submenus for that function. The keywords are abbreviated and "ENTER;" is used where necessary.

Note

When changing SETUP functions with more than one submenu, change only the appropriate submenu.

UNITS

The UNITS function allows you to set scale factors for position, velocity, and acceleration values, and to set the velocity range for each axis. The following is an example:

```
SETUP UNITS;
25      40.96     12.5      125      PO;
250     409.6     125       1250     VE;
2500    4096      1250      12500    AC;
HIGH    MID       LOW       FAST     RA;
ENTER;
```

In the above example, PO = Position, VE = Velocity, AC = Acceleration, and RA = Range.

Note

In Front Panel operation, the velocity ranges are displayed as 500 KHZ, 50 KHZ,5 KHZ, 500 HZ, and 1 MHZ. The corresponding parameters for the remote SETUP UNITS are: HIGH, MID, LOW, SLOW, and FAST.

ACCESS

The ACCESS function allows you to change the access code. The following example changes the access code to "1234":

```
SETUP ACCESS;
1234;
```

AXES

The AXES function allows you to set the mode of all the axes. The following is an example:

```
SETUP AXES;
M_INC     M_ABS     E_INC     E_ABS     PO;
NO        NO        YES       NO        AB;
NO        NO        NO        YES       ST;
NO        NO        YES       NO        OU;
NO        NO        NO        YES       PA;
ENTER;
```

In the above example, PO = Positioning Mode, AB = Stall = Abort?, ST = Stall = Stop?, OU = Stall = Output?, and PA = Not Deadband = Pause?.

REMOTE

The REMOTE function allows you to change the remote interface set from the Front Panel and to choose whether prompts will be displayed on the terminal or computer being used. Both interfaces can be connected and one will resign control to the other as soon as the semicolon (;) of the second line of the REMOTE function is received. The following example selects the RS-232C interface and requests prompts be displayed:

```
SETUP REMOTE;
SERIAL ON;
```

The following example selects the IEEE-488 interface and requests no prompts be displayed:

```
SETUP REMOTE;
PARALLEL OFF;
```

Note

In Front Panel operation, the interfaces are designated as RS232 and 488. The corresponding terms for the remote command are SERIAL and PARALLEL.

JOG

The JOG function allows you to specify the JOG HIGH and JOG LOW speeds for each axis. Speeds are in the velocity units that were previously set up. The following is an example:

```
SETUP JOG;
10     20     30     40     HI;
2      4      6      8      LO;
ENTER;
```

In the above example, HI = JOG High Speed, and LO = JOG Low Speed.

ENCODER

The ENCODER function allows you to specify the motor and encoder resolutions, the position maintenance function, deadband, and backlash. The following is an example:

```
SETUP ENCODER;
25000     4096     16384     200     MO;
4000      400      1000      50      EN;
40        4        10        1       DE;
2500      409      163       20      BA;
25000     4096     16384     200     MA;
1         2        3         2       AL;
37        50       37        10      GA;
ENTER;
```

In the above example, MO - Motor Steps, EN - Encoder Steps, DE - Deadband, BA - Backlash, MA - Maximum Correction Velocity, AL - Algorithm, and GA - Gain.

AUTORR

The AUTORR function allows you to automatically recall and/or run the program saved in EEPROM on power-up. A YES as the first word of the second line of this function will automatically recall the program; a NO will not. A YES as the second word of the second line of this function will automatically run the program; a NO will not. The following example automatically recalls the program but does not run it:

```
SETUP AUTORR;
YES NO;
```

Note

You cannot automatically run a program without automatically recalling it. Thus, NO YES; as the second line of the AUTORR function is illegal.

REMSS

The REMSS function allows you to select whether the 3000 will use INPUT #1 on the Back Panel for a remote STOP line and INPUT #2 for a remote START line. A YES as the first word of the second line of this function selects INPUT #1 to be used as a remote STOP line; a NO deselects INPUT #1 as a remote STOP line. A YES as the second word of the second line of this function selects INPUT #2 to be used as a remote START line; a NO deselects INPUT #2 as a remote START line. The following example causes INPUT #1 to be used as a remote STOP line and INPUT #2 not to be used as a remote START line:

```
SETUP REMSS;
YES NO;
```

LIMITS

The LIMITS function allows you to set the absolute position boundaries for motion for each axis so that overtravel will not occur. A deceleration value is associated with these boundaries that causes the axis to decelerate at a specified rate converging towards the boundaries. An abort option allows you to cancel the program when either a hardware or software limit has been exceeded. The following is an example:

```
SETUP LIMITS;
250000      40000      100000      409600      UP;
-250000     -40000     -100000     -409600     LO;
250000      40000      100000      409600      DE;
YES         NO         YES         NO          AB;
ENTER;
```

In the above example, UP = Upper Bound, LO = Lower Bound, DE = Deceleration, and AB = Abort on Limit?.

HOME

The HOME function allows you to specify the HOME input active level, the active edge, the final approach direction and the final approach velocity for HOME for each axis. See Section 4.3.3.12, HOME function, for a detailed description of these parameters. The following is an example:

```
SETUP HOME;
HIGH        LOW        HIGH        LOW         LE;
CW          CCW        CW          CCW         ED;
CCW         CW         CCW         CW          DI;
10          20         30          40          VE;
ENTER;
```

In the above example, LE = Active Level, ED = Active Edge, DI = Final Search Direction, and VE = Final Search Velocity %.

4.4.2. 1811 Remote Command Language

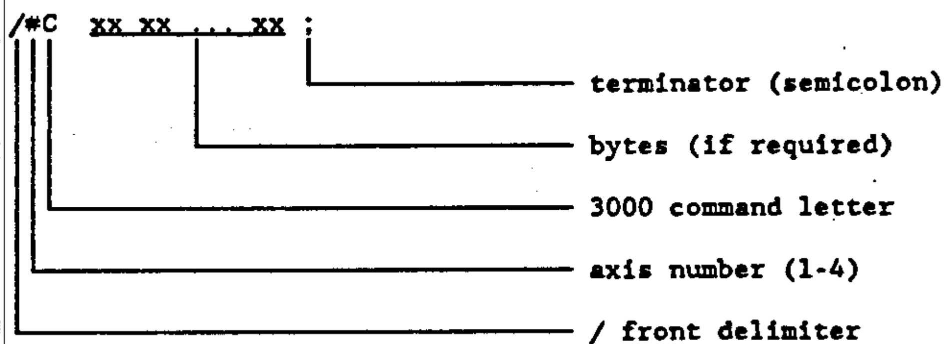
The 1811 Remote Command Language allows you to send commands to, and receive responses directly from, the 1811 Indexer controlling each axis of the 3000 system. This allows control of each 1811 in the 1811 Command Language and use of all the features and commands available with the 1811. Refer to the Model 1811 Operator's Manual for details. In addition, you may reset each axis of the 3000 and examine the status port of each axis. The Remote Command Language is intended to be used with either a remote terminal or computer. If you select operation with a terminal, a command summary will be displayed on power-up, and the prompt

INPUT?

will be displayed when the 3000 is ready to accept input. Each character will be echoed as it is typed, and descriptive messages will accompany any error code that the 3000 sends. If computer operation (TEXT OFF) is selected, the menu and prompt are not displayed, characters are not echoed, and error codes are not accompanied by descriptions. Only 1811 responses and 3000 error codes are sent to a remote computer.

4.4.2.1. Syntax

The following command syntax must be used to allow you to communicate with an individual 1811. The 1811 Command Language rules must be followed, and you should have a copy of the Model 1811 Operator's Manual. In addition, the following format is expected by the 3000:



Only inputs starting with / and ending with ; will be processed by the 3000. Of the characters between the delimiters, the following will be ignored: carriage returns, linefeeds, and other control characters except backspace, which will delete the previous character. All command letters must be preceded by the axis number of the 1811 they are referring to. An exception is the M command, which merely displays the command menu. If the letter command is one that is followed by an 1811 command, each byte in the 1811 command is represented by a pair of hex characters without leading zeros or trailing notation such as "H" or "X." The 3000 command input is terminated by a semicolon, which will cause the 3000 to process the input and execute the commands.

4.4.2.2. Functions

The following list gives the command letters allowed and a description of their function.

Note

Each command must be preceded by the / character and the axis number.

```
C xx xx xx...; send 1811 command described by xx xx xx.
L;             get 1811 pending message
M;             display remote command Menu
R xx xx xx...; send 1811 command and display response
X;             display 1811 status port
Z;             reset 1811
```

These command letters are described in Table 4-4.

4.4.2.3. Responses

The 3000 will send 1811 response from the L and R commands to the remote interface in the following format:

```
*xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;  (CR) (LF)
```

end of line

semicolon

16 pairs hex (bytes)

* preceeds output

4.4.2.4. Error Responses

The 3000 will send error codes in response to input syntax errors or if the 1811 has no message when one is requested via the L or R command. The error response format is shown below. The error text is not displayed when the 3000 is set up to operate with a computer.

```
:xx; <error message text>  (CR)  (LF)
```

end of line

optional text

semicolon

2 char hex error code

: preceeds output

Six error codes may arise during operation. These are listed and described in Table 4-5.

Table 4-4: 1811 Command Letters

Letter	Description
C	This command expects a string of bytes to follow and will send this string to the 1811 as a command. This command should be used to send 1811 commands that will not generate an 1811 response or whose response may not come before another 1811 command needs to be sent.
L	This command will display an 1811 response if one is pending. Otherwise it will display an error code and message. This command should be used to get 1811 responses from request commands sent via the C command.
M	This command displays the remote 1811 menu if the user is operating in terminal mode. If operating in computer mode, the output is suppressed.
R	This command expects a string of bytes to follow and will send it to the 1811 as a command. This command should be used to send 1811 request commands; that is, 1811 commands that will generate an immediate 1811 response.
X	This command is used to display the value of the 1811 status port, which allows the user to determine if the 1811 has a message pending or is ready to accept command. For a full description of the bits in this status port, refer to the 1811 product description.
Z	This command will reset the 1811. It should not generally be required, but it is available for diagnostic purposes.

Table 4-5: 1811 Error Codes

Code	Description
00	Axis not present. This error occurs if an attempt is made to address an axis not present in the machine.
01	Invalid axis specification. This error occurs if the first character following the / is not 1 thru 4.
02	Invalid command. This error occurs if a command letter other than the ones listed above follows the axis specification.
03	Byte string syntax error. This error occurs if the byte string contains nonhexadecimal characters or if an odd number of hexadecimal characters is contained in the byte string.
04	Excess command input. This error occurs on commands with no following byte string if any unignored characters come between the command letter and the semicolon.
05	No 1811 message ready. This error occurs if an L or R command is issued and the 1811 has no immediate response. The 3000 allows several milliseconds before concluding that no 1811 message is ready.

4.4.3. RS-232C and IEEE-488 Interfaces

The Remote Command Languages are the same for both the RS-232C and the IEEE-488 interfaces. To provide visual feedback for RS-232C terminal users, the 3000 provides a prompt or a new line after receiving a line from the remote interface. After the LOAD and SETUP command lines, and after each data line following those command lines, the 3000 responds with a <CR>, <LF>. After the other commands (one-line commands), the 3000 responds with the prompt INPUT? if TEXT is ON.

4.4.3.1. RS-232C Interface

The 3000 Front Panel function keys and Display provide a method for setting up the 3000's RS-232C interface. By selecting the SETUP function key, you can choose to set up either RS-232C or IEEE-488 interfaces. By choosing RS-232C, you can set up the baud rate, number of data bits, stop bits, parity (even, odd, or none), DTE or DCE configuration and terminal (TEXT ON) or computer (TEXT OFF) operation. When your choices are all displayed on the 3000 Display, press the ENTER key to set the 3000 RS-232C interface to the desired configuration (Refer to in Section 4.4.3.).

The RS-232C interface has a 600 character buffer for incoming characters, and both Transmit and Receive support the X-ON, X-OFF protocol. Although the 3000 will not respond to incoming commands while it is printing, the incoming characters will be buffered and the commands will be executed later. When the input buffer is filled to 580 characters, the 300 sends the X-OFF character. When the input buffer drops below 500 characters, the X-ON character is sent. If the input is overfilled, some characters may be lost.

4.4.3.2. IEEE-488 Interface

The IEEE-488 interface has no input character buffer because handshaking is included as part of the IEEE-488 standard. Thus, it is important that the host only attempt to send characters when the 3000 is ready, and the host accept 3000 responses when the 3000 is ready to send them.

The 3000 IEEE-488 interface includes serial poll response. As usual, Bit 6 indicates a service request to the host. You may choose to activate SRQ when the 3000 has a message via the SETUP 488 procedure. In addition, two other bits in the serial poll response always have significance. Bit 5 indicates when the 3000 has a message ready, and Bit 4 indicates when the 3000 is ready to receive text. When using the IEEE-488 interface with a host computer, it is important that the host program communicate with 3000 according to these bits in the serial poll register.

The program shown in Figure 4-81 was written in BASIC on an HP-85 serving as an IEEE-488 terminal. Statements 5 and 7 declare the size of the output and input buffers, respectively. Statements 10 and 20 define keyboard interrupts. Statements 22 and 24 define and enable the SRQ interrupt procedure. Statements 28 thru 40 check the MESSAGE READY bit of the 3000. If the MESSAGE READY bit is active, the 3000 message is read and displayed by statements 50 and 60, after which the program continues to check the MESSAGE READY bit. If the user presses KEY 1, the program goes to statement 100, whereupon the user is prompted for input. Statements 105 and 110 wait for the 3000 RECEIVER READY Bit. When the RECEIVER READY bit goes active, the program waits for the use input at statement 112. When the user has entered a line, the program sends that line to the 3000 at statement 115 and returns. If the user presses KEY 2, the program goes to where the 3000 serial poll register is read. At statement 210, the data read from the serial poll is displayed in binary format. If SRQ goes active, the program goes to statement 300, which reads the status of port 7. Next the contents of the serial poll are read and displayed via the subroutine at statement 200. Finally, the interrupt for port 7 is reenabled. The important thing to notice about this sample program is that the 3000 RECEIVER READY is active before any attempt is made to send the 3000 a message, and the 3000 MESSAGE READY bit is active before the host attempts to read the 3000.

```
  5 DIM A$[40]
  7 DIM B$[40]
 10 ON KEY# 1, "CMD" GOSUB 100
 20 ON KEY# 2, "SPOLL" GOSUB 200
 22 ON INTR 7 GOSUB 300
 24 ENABLED INTR 7;8
 25 CLEAR 2@ KEY LABEL
 28 KEY LABEL
 30 X=SPOLL (701)
 40 IF BIT (X,5) <>1 THEN GOTO 28
 50 ENTER 701 USING "%,#%K" ; B$
 60 DISP B$
 70 GOTO 30
100 CLEAR @ KEY LABEL
102 DISP "COMMAND"
105 X-SPOLL (701)
110 IF BIT (X,4) <> THEN GOTO 105
112 INPUT A$
115 OUTPUT 701 ; A$
120 RETURN
200 X=SPOLL(701)
210 DISP "SERIAL POLL IS", DTB$(X)
220 RETURN
300 STATUS 7, 1 ; A
310 GOSUB 200
320 ENABLE INTR 7;8 @ RETURN
```

Figure 4-81: Sample IEEE-488 Program

The syntax of this program may change slightly if a computer other than the HP-85 is used. Refer to the appropriate programming manual for the equipment used.

Appendix A: SUMMARY OF COMMANDS

A. Command Summary by Menu

Command Name	Menu Page	Key	Description
			POWER-UP MENU
ACCESS	1	1	Asks user to enter password
RUN	1	2	Execute program in Main Memory from 000
RESUME	1	3	Resumes program execution
RECALL	1	4	Loads contents of NVM into Main Memory
			MAIN MENU
EDIT	1	1	Invokes EDIT Menu
RUN	1	2	Executes program in Main Memory from 000
SAVE	1	3	Loads contents of Main Memory into NVM
RECALL	1	4	Loads contents of NVM into Main Memory
LOAD	1	5	Loads from IEEE or RS-232C into Main Memory
ETC	1	6	Steps to next page of menu
PRINT	2	1	Outputs Main Memory to IEEE or RS-232C
SETUP	2	2	Invokes SETUP Menu
RESUME	2	5	Resumes program execution
PZERO	2	3	Defines current position of all axes as zero
DISPLAY	2	4	Sets up data for display during execution
			SETUP MENU
RS-232	1	1	Sets up five parameters for RS-323C port
488	1	2	Sets 3000's address/selects SRQ ON/OFF
UNITS	1	3	Sets parameters for Pos, Vel, Accel & Vel Rn
ACCESS	1	4	Changes access code word
AXES	1	5	Selects Absolute or Incremental Positioning
REMOTE	2	1	Selects IEEE-488 or RS-232/Menus to host ON/OFF
JOG	2	2	Sets LO/HI JOG velocities
ENCODER	2	3	Specifies motor/encoder steps/revolution
AUTORR	2	4	Selects AUTORECALL/AUTORUN feature
REMSTSP	2	5	Uses OPTO-22 inputs to start/stop program
LIMITS	3	1	Sets absolute position boundaries
HOME	3	2	Defines the HOME input axis position parameters

Command Name	Menu Page	Key	Description
			EDIT MENU -- MOTION CONTROL
MOVE	1	1	Moves axis (must complete before next command)
MOVI	1	2	Moves axis (next command accepted immediate)
ACEL	1	3	Specifies acceleration in steps/sec/sec
VELC	1	4	Specifies velocity in steps/sec
ENTSTR	6	1	Sets absolute velocity streaming mode
ENISTR	6	2	Sets incremental velocity streaming mode
STRV	6	4	P/O velocity/distance setup
STRD	6	5	P/O velocity/distance setup
STRL	7	3	Specifies feedrate at end of segment
STRE	7	5	Specifies coordinates at end of segment
STRA	7	2	Specifies feedrate of arc segment
STRO	7	4	Specifies coordinates of center of arc segment
OUTSTR	7	1	Outputs trigger bit at break points
EXSTR	6	3	Exits Streaming function
			EDIT MENU -- MOTOR/DRIVE CONTROL
BOOST	5	2	Applies extra torque to a motor
SHTDN	5	3	Removes regular torque from a motor
			EDIT MENU -- PROGRAM CONTROL
DWELL	4	5	Waits for predefined period of time
LOOP	2	1	Defines beginning of loop
ENDLOOP	2	2	Defines end of loop
BRANCH	2	4	Tests input once/jump if condition exists
ONTRIG	5	4	Tests input repeatedly/jump if condition exists
IFTHEN	8	3	Tests 2 variables/jump if condition false
TRIG	2	3	Pauses/waits for predefined programmable input
GOSUB	3	1	Jumps to subroutine then return to next line
RETURN	3	2	Returns from subroutine to main program
GOTO	3	3	Jumps program to specified line
WAIT	3	5	Pauses until all axes finish moving
			EDIT MENU -- POSITION/COORDINATE REFERENCE
MODE	5	1	Sets absolute or incremental velocity streaming
HOME	3	4	Searches for the HOME input axis position
PDEF	4	3	Defines current position
			EDIT MENU -- ARITHMETIC AND COMPUTATION
MATH	1	5	Defines numeric variables

Command Name	Menu Page	Key	Description
			EDIT MENU -- INPUT/OUTPUT
INPUT	4	2	Assigns variable values from different sources
DATA	5	5	Stores values used by INPUT = SRC3 command
WRITE	4	1	Displays variables
DISPLAY	4	4	Triggers display during program execution
OUTPUT	2	5	Sets programmable output port
			EDIT MENU -- TEACH
TEACH	8	1	Uses JOG function to input program
			EDIT MENU -- UTILITIES
LIST	8	5	Displays specified line(s) of program
DELETE	8	4	Deletes specified line(s) of program
DONE	8	2	Defines end of program
RESUME	1	3	Resumes program execution

B. Command Summary by Alphabet

Command Name	Menu Page	Key	Description
488	1	2	Sets 3000's address/selects SRQ ON/OFF
ACCESS	1	4	Changes access code word
ACCESS	1	1	Asks user to enter password
ACEL	1	3	Specifies acceleration in steps/sec/sec
AUTORR	2	4	Selects AUTORECALL/AUTORUN feature
AXES	1	5	Selects Absolute or Incremental Positioning
BOOST	5	2	Applies extra torque to a motor
BRANCH	2	4	Tests input once/jump if condition exists
DATA	5	5	Stores values used by INPUT - SRC3 command
DELETE	8	4	Deletes specified line(s) of program
DISPLAY	2	4	Sets up data for display during execution
DISPLAY	4	4	Triggers display during program execution
DONE	8	2	Defines end of program
DWELL	4	5	Waits for predefined period of time
EDIT	1	1	Invokes EDIT Menu
ENCODER	2	3	Specifies motor/encoder steps/revolution
ENDLOOP	2	2	Defines end of loop
ENISTR	6	2	Sets incremental velocity streaming mode
ENTSTR	6	1	Enters Streaming function
ETC	1	6	Steps to next page of menu
EXSTR	5	4	Exits Streaming function
GOSUB	3	1	Jumps to subroutine then return to next line
GOTO	3	3	Jumps program to specified line
HOME	3	2	Defines the HOME input axis position parameters
HOME	3	4	Searches for the HOME input axis position
IFTHEN	8	3	Tests 2 variables/jump if condition false
INPUT	4	2	Assigns variable values from different sources
JOG	2	2	Sets LO/HI JOG velocities
LIMITS	3	1	Sets absolute position boundaries
LIST	8	5	Displays specified line(s) of program
LOAD	1	5	Loads from IEEE or RS-232C into Main Memory
LOOP	2	1	Defines beginning of loop
MATH	1	5	Defines numeric variables
MODE	5	1	Sets absolute or incremental velocity streaming
MOVE	1	1	Moves axis (must complete before next command)
MOVI	1	2	Moves axis (next command accepted immediate)
ONTRIG	5	4	Tests input repeatedly/jump if condition exists
OUTPUT	2	5	Sets programmable output port
OUTSTR	7	1	Outputs trigger bit at break points
PDEF	4	3	Defines current position
PRINT	2	1	Outputs Main Memory to IEEE or RS-232C
PZERO	2	3	Defines current position of all axes as zero
RECALL	1	3	Loads contents of NVM into Main Memory
RECALL	1	4	Loads contents of NVM into Main Memory

Command Name	Menu Page	Key	Description
REMOTE	2	1	Selects IEEE-488 or RS-232/Menus to host ON/OFF
REMSTSP	2	5	Uses OPTO-22 inputs to start/stop program
RESUME	1	3	Resumes program execution
RESUME	2	5	Resumes program execution
RETURN	3	2	Returns from subroutine to main program
RS-232	1	1	Sets up five parameters for RS-323C port
RUN	1	2	Executes program in Main Memory from 000
SAVE	1	3	Loads contents of Main Memory into NVM
SETUP	2	2	Invokes SETUP Menu
SHTDN	5	3	Removes regular torque from a motor
STRA	7	2	Specifies feedrate of arc segment
STRD	6	5	P/O velocity/distance setup
STRE	7	5	Specifies coordinates at end of segment
STRL	7	3	Specifies feedrate at end of segment
STRO	7	4	Specifies coordinates of center of arc segment
STRV	6	4	P/O velocity/distance setup
TEACH	8	1	Uses JOG function to input program
TRIG	2	3	Pauses/waits for predefined programmable input
UNITS	1	3	Sets parameters for Pos, Vel, Accel & Vel Rng
VELC	1	4	Specifies velocity in steps/sec
WAIT	3	5	Pauses until all axes finish moving
WRITE	4	1	Displays variables

Appendix B: PROGRAMMING EXAMPLES

This appendix contains examples of programs for the 3000 system, which show you how to use typical 3000 commands. This appendix is divided into programs that exemplify simple moves and complex moves.

> **Note**
>
> Examples of how to use other commands, such as TEACH, MATH, WAIT, TRIG, LOOP, and DONE, are contained in Chapter 4.

A. Examples of Simple Moves

Example 1

You want to inspect the die on a semiconductor mask. You must start at a reference point and visit each point in a matrix. At each point you need to wait for 20 seconds, then return to the reference point.

The solution consists of a two-axis stage and a 3000 system with two-axis capability. A picture of the mask is shown below. The program starts at the point marked x and visits all the points marked with o. When all the dies on the mask have been scanned, you end up at point Q.

```
<----> 1000 motor steps

     x    o    o    o    o    o    o    o    o

5000 motor steps

     o    o    o    o    o    o    o    o    o

     o    o    o    o    o    o    o    o    o

     o    o    o    o    o    o    o    o    o

     Q
```

The program looks like this:

```
0010ACEL      500000    500000
0020VELC      250000    250000
0100LOOP      2
0110GOSUB     4000
0120MOVE      NULL      5000
0125DWELL     20000     MILLISECONDS
0130GOSUB     6000
0140MOVE      NULL      5000
0145DWELL     20000     MILLISECONDS
0150ENDLOOP
0160MOVE      NULL      -20000
0170DONE
```

```
4000LOOP      8
4010MOVE      1000     NULL
4020DWELL     20000    MILLISECONDS
4030ENDLOOP
4040RETURN

6000LOOP      8
6010MOVE      -1000    NULL
6020DWELL     20000    MILLISECONDS
6030ENDLOOP
6040RETURN
```

Example 2

You have a simple punch operation that makes a preset move on one axis, waits for 5 seconds then returns the axis to a reference position. However, if programmable input bit #1 is on (low), the machine is broken and you do not want the motor to move if START is pressed. If you press START, the 3000 should not move the motor and it should turn on output bit #2 to indicate an error.

The program looks like this:

```
0010ACEL      500000
0020VELC      250000
0030BRANCH    1XXX  XXXX  XXXX  XXXX  THEN  4000
0040MOVE      10000
0050DWELL     5000        MILLISECONDS
0060MOVE      -10000
0070DONE

4000OUTPUT     1XX  XXXX  XXXX  XXXX
4010DONE
```

Example 3

You need a simple demonstration program that rotates Motor #1 one revolution clockwise at 1 rev/sec with a 10 rev/sec/sec acceleration rate. The motor should then dwell there for one second. Motor #2 should then duplicate the same move after Motor #1 is done. Then Motor #1 should rotate counter-clockwise 5 times moving 1/5 revolution each time. Finally, after another 1 second delay, both motors should slew to a velocity of 5 revolutions/ second and hold there for 5 seconds before the DONE statement cause both motors to stop.

Here is a program that will do this:

```
0010VELC        25000     25000
00210ACEL       250000    250000
0030MOVE        25000     NULL
0040DWELL       1000      MILLISECONDS
0050MOVE        NULL      25000
0060LOOP        5
0070MOVE        -5000     NULL
0080ENDLOOP
0085DWELL       1000      MILLISECONDS
0086VELC        125000    125000
0090MOVE        SLEW+     SLEW+
0100DWELL       5000      MILLISECONDS
0110DONE
```

B. Examples of Complex Moves

B.1. Velocity Streaming

The programs in this section show how to move the motor 25000 steps in three increments at different velocities. The first program is an example of absolute velocity streaming using the ENTSTR command, which specifies the motor position with reference to an absolute location determined when the ENTSTR command is executed. The second program is an example of incremental velocity streaming using the ENISTR command, which shifts the reference point to the starting point of the segment being traversed.

Example 1

```
0010ENTSTR              ;enter absolute streaming
0020STRV 10000          ;velocity is 10000 steps/sec
0030STRD  5000          ;move 5000 steps
0040STRV 15000          ;velocity is 15000 steps/sec
0050STRD 15000          ;move 10000 steps
0060STRV 0              ;stop
0070STRD 25000          ;move 10000 steps

0080EXSTR               ;exit streaming
0090DONE                ;exit program
```

Example 2

```
0010ENISTR              ;enter incremental streaming

0020STRV 10000          ;velocity is 10000 steps/sec
0030STRD  5000          ;move 5000 steps
0040STRV 15000          ;velocity is 25000 steps/sec
0050STRD 10000          ;move 10000 steps
0060STRV 0              ;stop
0070STRD 10000          ;move 10000 steps
```

```
0080EXSTR          ;exit streaming
0090DONE           ;exit program
```

B.2.2. Contouring

The programs in this section show how to draw a pie using line and arc segments. The first example shows contouring using the absolute velocity streaming command ENTSTR. The second example shows contouring using the incremental velocity streaming command ENISTR. Figure B-1 shows and example of the pie to be drawn.

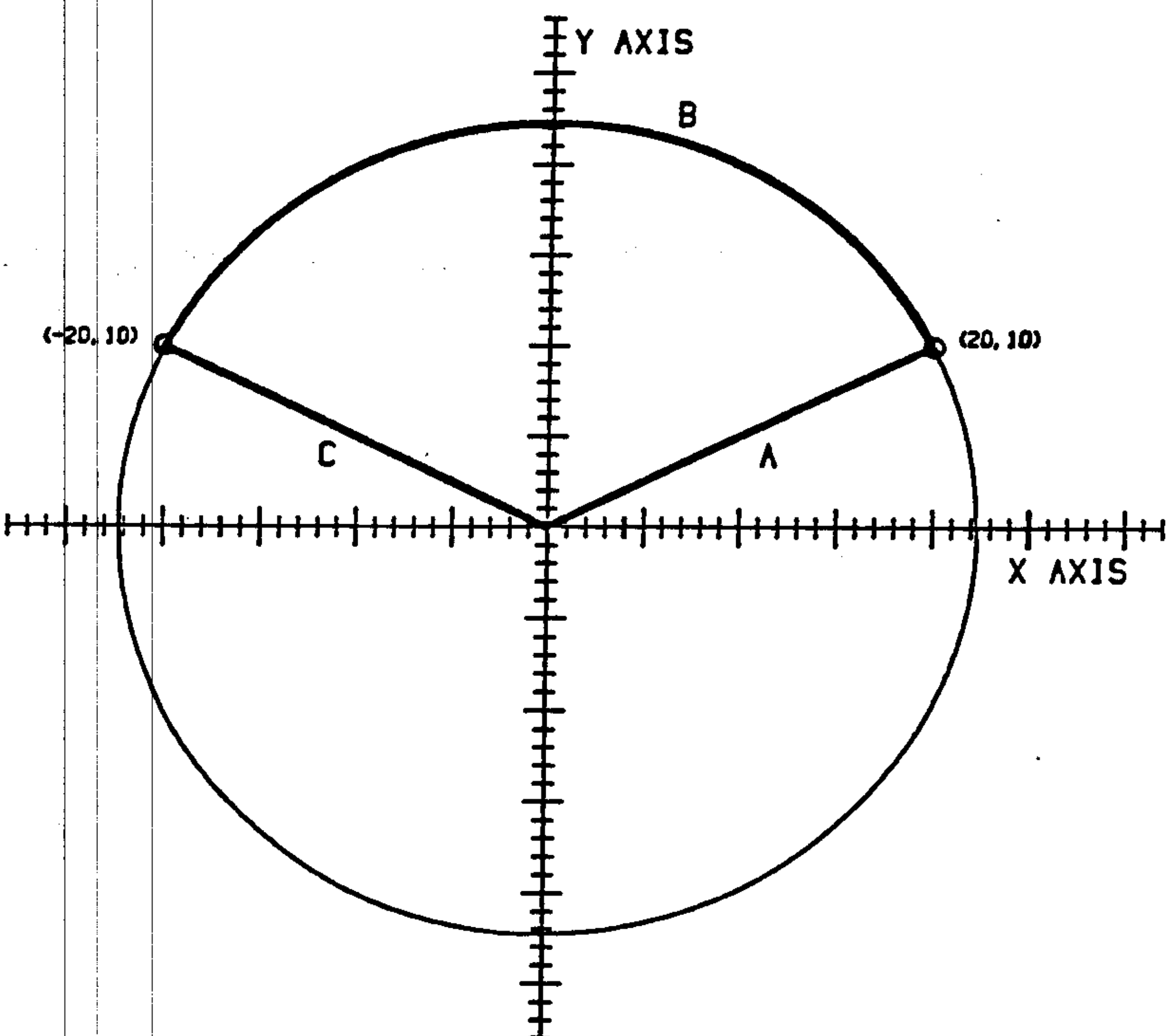


Figure B-1: Contouring a Pie

Example 1

This example draws a pie using the absolute velocity streaming command ENTSTR. Segment A is a line that starts with 0 velocity and accelerates to 25000 steps/sec (lines 0120-0130). Segment A ends at position 20,10 and moves right into a CCW arc (segment B) with a feedrate of 25000 steps/sec, which is the ending feedrate of the line.

The center, or origin, of the arc is at 0,0 and the end of the arc is at 20,10 (lines 0140-0160). Segment C is a line that is used to decelerate to 0 velocity (lines 0170-0180).

Lines 0190 and 0200 take the 3000 out of streaming mode and end the program.

```
0100MATH  Q1 = 25000      ;feedrate
0110ENTSTR     YES   YES  ;feedrate is 0

0120STRL  Q1              ;accelerate to 25000 steps/sec
0130STRE       20    10   ;end of line segment A
0140STRA  Q1              ;CCW arc at 25000 steps/sec
0150STRO       0     0    ;center of arc segment B
0160STRE       -20   10   ;end of arc segment B
0170STRL  0               ;decelerate to zero velocity
0180STRE       0     0    ;end of line segment C

0190EXSTR                 ;exit streaming
0200DONE                  ;exit program
```

Example 2

This example draws the pie in Figure B-1 using the incremental velocity streaming command ENISTR. In this case, the reference coordinates shift to the starting point of the segment being traversed.

```
0100MATH  Q1 = 25000      ;feedrate
0220ENISTR     YES   YES  ;feedrate is 0

0120STRL  Q1              ;accelerate to 25000 steps/sec
0130STRE       20    10   ;end of line segment A
0140STRA  Q1              ;CCW arc at 25000 steps/sec
0150STRO       -20   -10  ;center of arc segment B
0160STRE       -40   0    ;end of arc segment B
0170STRL  0               ;decelerate to 0 velocity
0180STRE       20    -10  ;end of line segment C

0190EXSTR                 ;exit streaming
0200DONE                  ;exit program
```

Appendix C: FIELD INSTALLATION OF 1811 CARDS

It is possible to add 1811 axis cards to a 3000 in the field if the 3000 has less than four axes in it. This procedure is normally done by authorized Compumotor distributors only.

The procedure requires a small flat blade screwdriver, a 1/4" nutdriver, needlenose pliers, and a clean place to work. Disconnect AC power going to the unit.

C.1. Procedure

a. Inspect the 1811 card to be installed. Make sure that the EPROMs are marked 92-3657. Earlier versions of the 1811 and 3000 contained software with the part number 92-2514. This older software will not work in a Revision A Model 3000. If the 1811 you are attempting to install in the 3000 has the older software, obtain an RMA from Compumotor Corporation and return it for update.

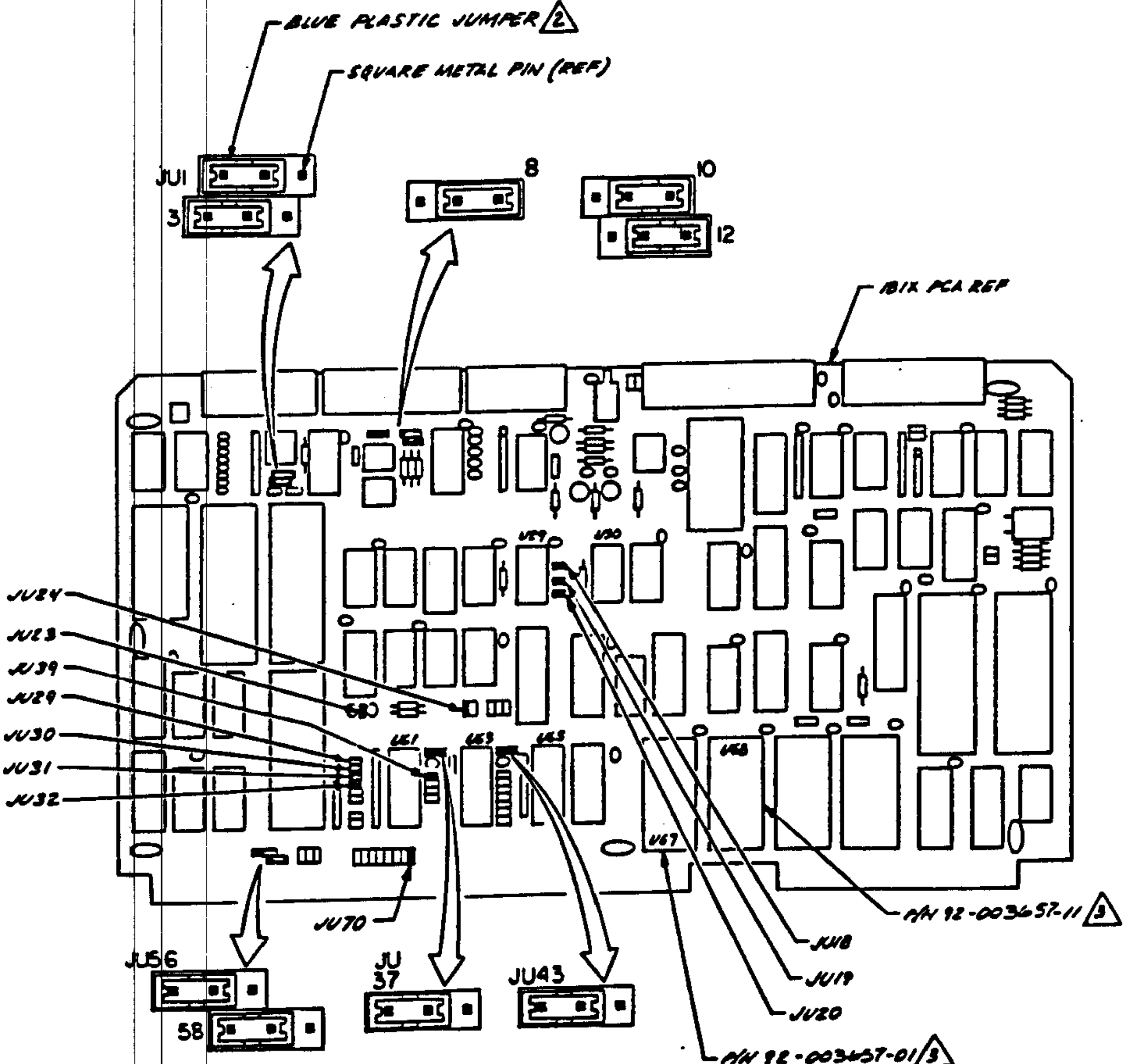
b. Configure the jumpers on the card according to Figure C-1. Each blue jumper can be removed with needlenose pliers and re-inserted with finger pressure. Note 1 on Figure C-1 lists the jumpers that must be in place for the 1811 to operate correctly in the 3000. All axes are configured the same, as the 3000 addresses axes by location.

c. Using a screwdriver, remove the top cover on the 3000. Disconnect the flat cables that go between the rear connector panel and each axis card. Be very careful to ease the connectors out without tugging on the flat cables themselves. Disconnect the RS232C and IEEE-488 cables that go between the rear panel and the master controller card, which occupies the frontmost slot in the unit. Do not disturb the other cables on the master controller. Remove the rear connector panel with the four screws that attach it, and swing it away from the chassis. The programmable input and output cables remain connected, so be careful not to stretch or pull on them.

d. On the connector panel remove all of the black 4-40 Phillips head screws that retain the hole cover(s) for the axis (axes) being installed. Discard the screws but save the hole cover plate, which will be used to attach the new connectors.

e. Install the 15-pin joystick connector and the 25-pin Motor/Drive and Auxiliary Connectors as indicated in Figure C-2. Install the 25-pin encoder connector if an 1811 card with Position Tracking is being incorporated. Sandwich the connectors between the connector panel and the hole cover bracket. Attach and tighten the 4-40 jackscrews supplied with the connector kit. Refer to Figure C-2.

Figure C-1: 1811 Jumper Configuration



f. Insert the additional 1811 card in their card guide. Install each board in the appropriate slot, with Axis #1 being the rearmost slot and Axis #4 directly behind the controller card. The cards should push down with little effort. Push down firmly to seat the cards in the motherboard connectors.

g. Reinstall the rear connector panel and reinsert all flat cables. Remember that Axis #1 is towards the rear of the unit and Axis #4 is directly behind the controller board at the front of the unit.

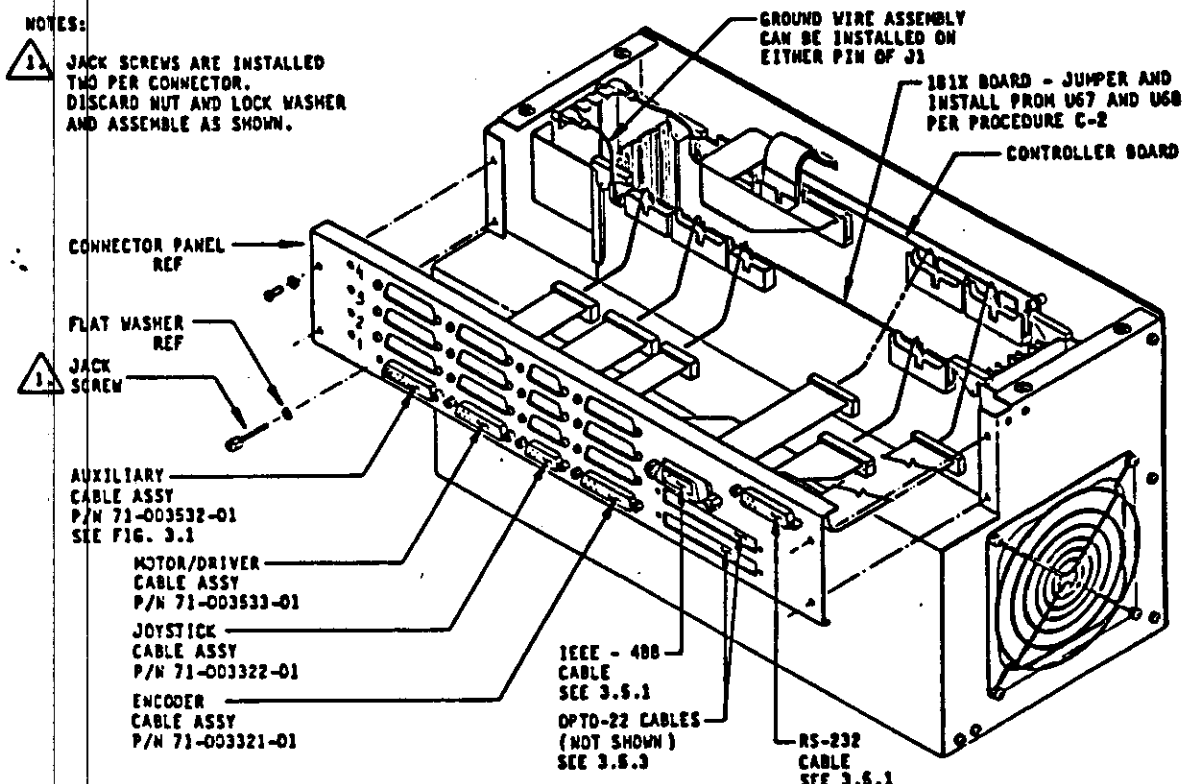
h. Check to make sure that all boards are seated firmly in the motherboard. Ensure that no flat cables are being pinched or stretched.

i. Attach the earth wire supplied with each 1811. This has a spade lug on one end that attaches to the brass ground bar on the right side of the chassis and a push-on connector at the other end that connects to the 2 pin ramp connector on the 1811 card Refer to Figure C-2.

j. Before you apply power, check for screws or debris that might have fallen into the unit. Turning the unit upside down and giving it a shake should dislodge unwanted metal parts.

k. Replace the top cover. The procedure is finished.

Figure C-2: Rear-Panel Connector Installation



Appendix D: DIMENSIONS FOR MOUNTING

D.1. Mounting Dimensions

Figures D-1 and D-2 show the dimensions of the 3000 that you will need to know to mount the unit. Figure D-3 is an illustration of the Rear Panel.

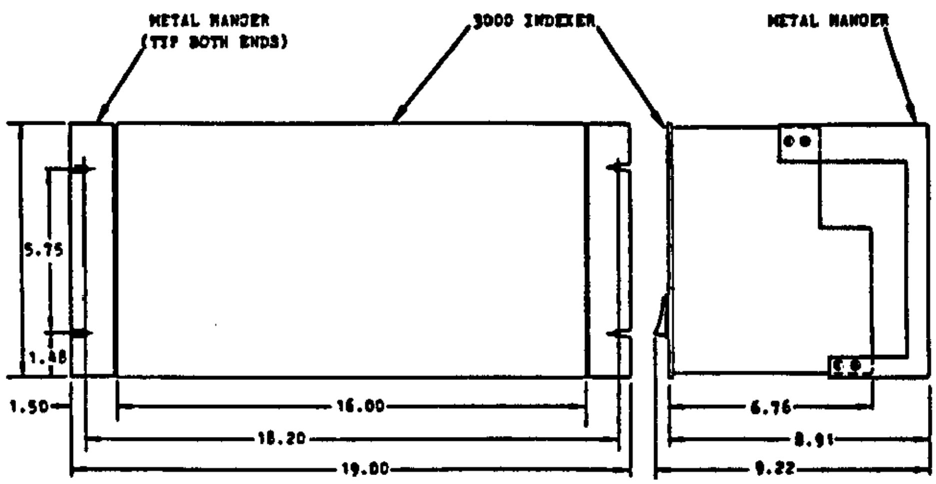


Figure D-1: Overall Dimensions

Figure D-2: Rack Mounting Kit