UNIDEX® 21 SERIES
MACHINE CONTROLLERS

OPERATION & TECHNICAL MANUAL

P/N: EDU125

AEROTECH, INC.
101 Zeta Drive
Pittsburgh, PA 15238-2897

Manual Structure

Aerotech has made every effort to structure each manual to best suit the needs of our customer(s). Therefore, we have consolidated all manuals applicable to the Unidex 21 into the *Unidex 21 Operation & Technical Manual* shipped with your controller.

The *Unidex 21 Operation & Technical Manual* is comprised of the following three manuals:

UNIDEX 21 HARDWARE MANUAL

- Provides the User with information concerning the internal structure.
- Provides details necessary to interface peripheral equipment.
- Describes the controller features.
- Provides solutions to identify and correct failure.

UNIDEX 21 PROGRAMMING MANUAL

- Provides the User with detailed information for each of the program commands.
  For example:
  - Command name and description as well as any parameters required by that command
  - Description of the command structure
  - Feedback to the User following command execution
  - Example of command use
  - Example of command used in conjunction with other commands
  - Comments pertaining to the user of the command
  - Names of related commands whose description may be helpful

UNIDEX 21 USER'S MANUAL

- Provides the User with general operating information.
- Describes how to make interface connections.
- Provides program editing capability via "On Screen" or "Menu Driven" methods.
- Provides file management capabilities. For example:
  - listing
  - verifying
  - transferring
- Describes Machine Mode programming and operation.
- Provides a detailed explanation for each of the parameters.
- Provides a Test Mode for testing EPROM, RAM, and Parameters.
- Capability to change the System Mode, Batch Mode, Console Mode, and Debug Mode.

UNIDEX® 21
MOTION CONTROLLER
HARDWARE MANUAL

PN: EDU 119

AEROTECH, INC., 101 Zeta Drive, Pittsburgh, PA 15238
(412) 963-7470 • TWX 710-795-3125 • FAX(412)963-7459

Aerotech Inc.
Release No. 1.0
July, 1991

DISCLAIMER:

The information contained in this manual is subject to change due to improvements in design. Though this document has been checked for inaccuracies, Aerotech, Inc. does not assume responsibility for any errors contained herein.

TRADEMARKS:

Unidex is a registered trademark of Aerotech, Inc.

IBM PC/XT/AT are registered trademarks of the International Business Machines Corporation.

PAMUX is a registered trademark of the Opto 22 Corporation.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

CHAPTER 2: SYSTEM OVERVIEW

CHAPTER 3: JUMPERS AND SWITCHES

CHAPTER 4: POWER INTERFACE

CHAPTER 5: MOTOR INTERFACE

CHAPTER 6: LIMIT AND HOME SWITCH INTERFACE

TABLE OF CONTENTS(CON'T)

LIST OF FIGURES

LIST OF FIGURES (CON'T)

CHAPTER 1: INTRODUCTION

This manual contains both electrical and mechanical hardware information for the Unidex 21 Motion Controller, Models U21B and U21R. For a thorough understanding of the complete Unidex 21 Control System it is also necessary to be familiar with the following manuals:

Unidex 21 User's Manual

Unidex 21 Programming Manual

Unidex 21 Options Manual

SECTION 1-1: USING THIS MANUAL

The focus of this manual is to provide the User with general information concerning the internal structure of the Unidex 21 as well as providing details necessary to interface peripheral equipment. Following is a brief description of the contents of each of the Chapters:

Chapter 1:	Introduction
Chapter 2:	Illustrations and descriptions of the external characteristics of the Unidex 21, Models U21B and U21R. A block diagram and description of internal system interaction, and simplified system wiring diagram are included for reference.
Chapter 3:	Illustrations and descriptions of Jumpers and Switches on the various Printed Circuit Boards of the Unidex 21.
Chapter 4:	Power requirements and information.
Chapter 5:	Detailed description and illustration of the Motor Interface Connectors.

Chapter 6:	General description of the Limit and Home Switch functions.
Chapter 7:	Detailed descriptions and illustrations of the interface for the various transducer types that may be used in conjunction with the Unidex 21.
Chapter 8:	Detailed description and illustration of the MST and Indexer Board Interface Connector.
Chapter 9:	Detailed description and illustration of the IEEE-488 Interface Connector.
Chapter 10:	Detailed descriptions and illustrations of the RS-232 Ports A and B and the Terminal Port.
Chapter 11:	Detailed description and illustration of the Synchronous Serial Interface Connector.
Chapter 12:	Detailed description and illustration of the interface connector used for a supplementary EGA Monitor.
Chapter 13:	Detailed description and illustration of the Opto 22, PAMUX Bus 24/I/O Channel, I/O Power and Miscellaneous I/O Interface Connectors.
Chapter 14:	Detailed description of the interface connectors used for Joystick Mouse operation of the Unidex 21.
Chapter 15:	Detailed description and illustration of the Trackball Interface Connector.
Chapter 16:	Detailed description and illustration of the PC/AT Keyboard Interface Connector.
Chapter 17:	Basic troubleshooting procedures to identify and correct possible malfunctions that may occuras a result of improper hardware configurations.

CHAPTER 2: SYSTEM OVERVIEW

SECTION 2-1: PACKAGING

The Unidex 21 is available in three major package configurations:

Desktop - Models U21B and U21R

Free Standing Enclosure - Model U21C

Panel or Rack Mounted - Models U21P or U21K

The operating parameters for all models are similar and are detailed in the *Unidex 21 Users Manual* and the *Unidex 21 Programming Manual.* Extensive hardware variations exist between Models however, warranting a separate Hardware Manual for each of the three package configurations.

This manual is intended for use with Unidex 21, Models U21B or U21R. Before proceeding, make certain that this manual is the appropriate manual for the Unidex 21 Model being used.

SECTION 2-2: UNIDEX 21 MODEL U21B/U21R DESCRIPTION

Models U21B and U21R are the most compact and self contained of the available Unidex 21 package configurations. (See Figures 2-1, 2-2 and 2-3.)

Each is capable of serving up to four integral DC Servo or AC Brushless Drivers.

Model U21B is full-featured except for the Operator's Front Panel. Communication with the U21B is accomplished through it's RS-232 terminal port to a remote data terminal. The U21B may be equipped with an optional CRT Driver Card and Front Panel Card permitting the addition of an EGA compatible monitor and AT-style keyboard providing full stand-alone operation.

Model U21R builds upon the U21B with the addition of a sealed- membrane front panel and electroluminescent display.

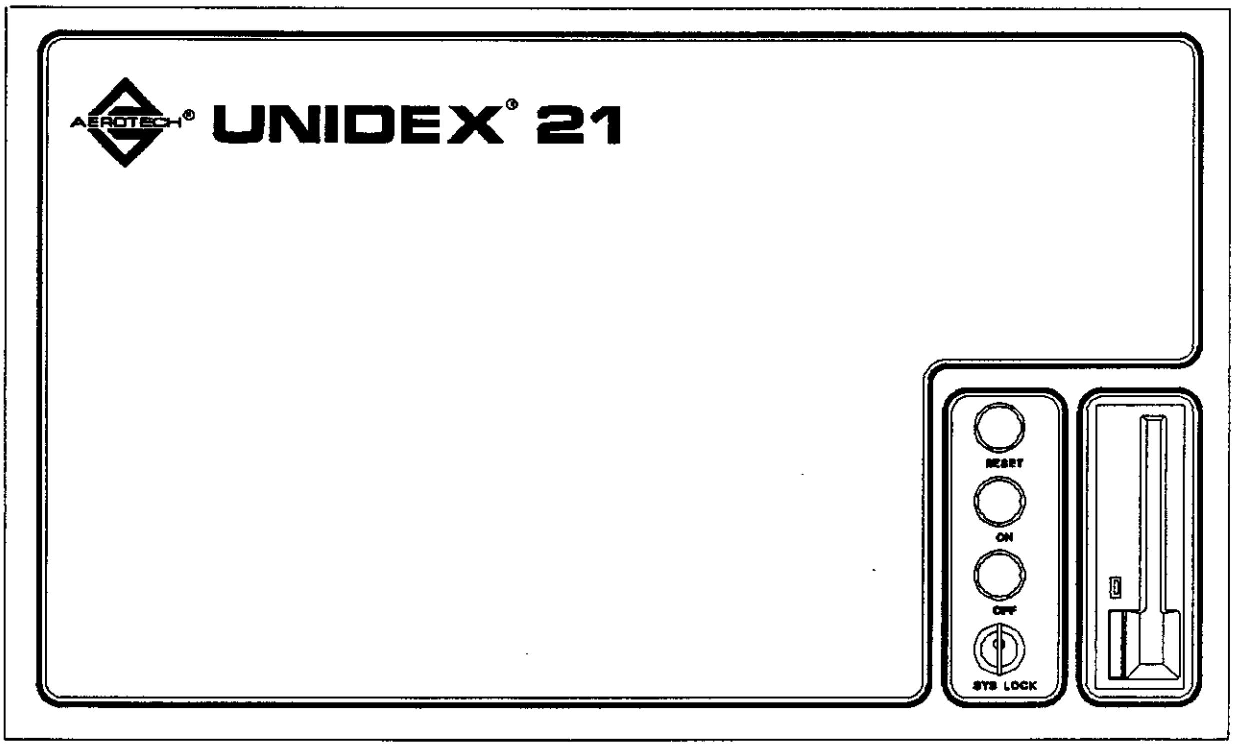


Figure 2-1: Unidex 21, Model U21B (Shown with Optional Disk Drive)

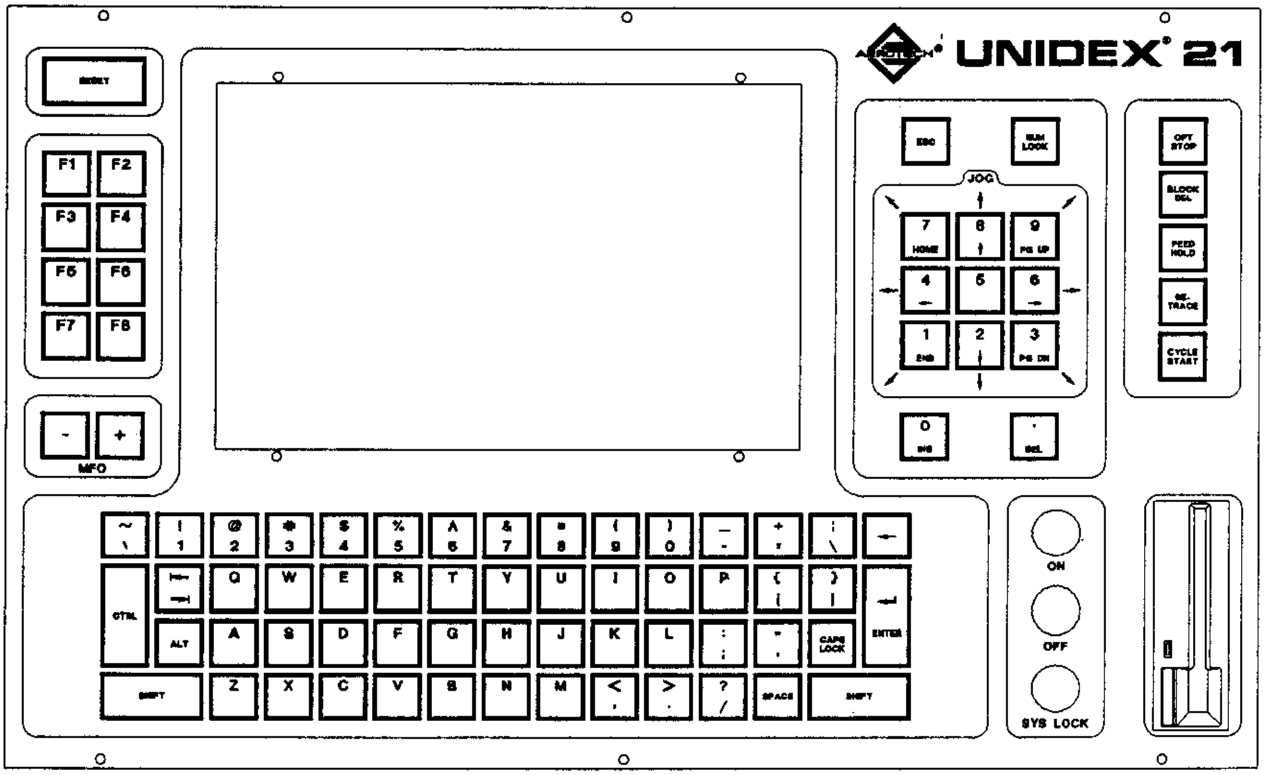


Figure 2-2: Unidex 21, Model U21R (Shown with Optional Disk Drive)

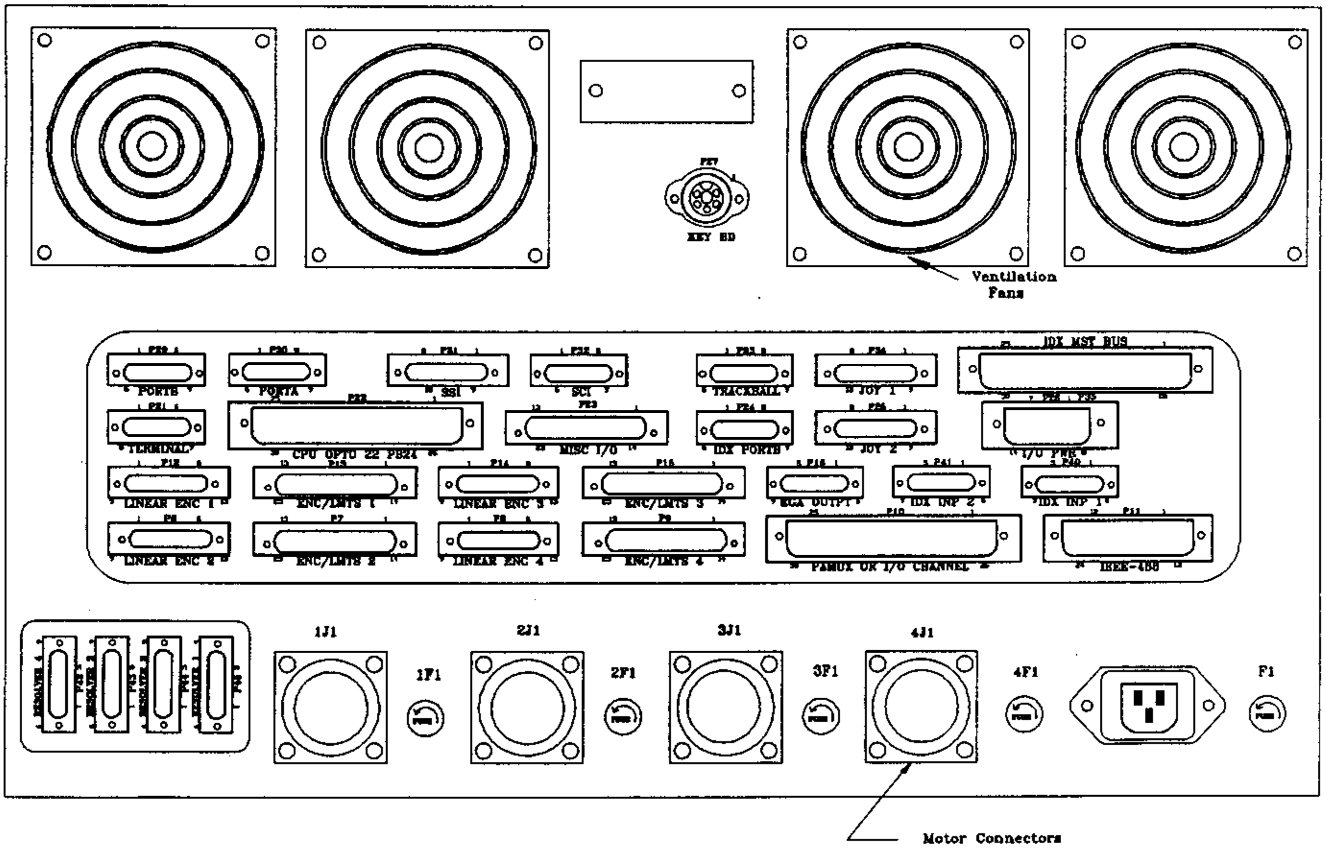


Figure 2-3: Unidex 21 Rear Panel, Models U21R and U21B

SECTION 2-3: INTERNAL STRUCTURE

The Unidex 21 is a microprocessor based multiple axis Motion Controller. A basic Unidex 21 consists of a CPU Board, an Indexer Board, a DSP Board, an optional graphics (CRT) Board and Front Panel Board and the Axis Amplifiers. (See Figures 2-4, 2-5, and 2-6.) A system wiring diagram is provided in Figure 2-7.

2-3-1: CPU BOARD

The CPU Board is the front end processor of the Unidex 21 system. It contains a Motorola 68000 microprocessor to handle File editing, File management, System Parameters, Machine Modes (Mdi, Jog, Debug) and peripheral communications via RS-232. It also contains eight (8) Opto-Isolated Outputs and sixteen (16) Opto-Isolated Inputs.

The CPU Board communicates with the Indexing Board through the VME Bus. The VME Bus allows communication between multiple boards within a system. In the Unidex 21 system the CPU Board is the Bus master with the CRT and the Indexer Boards as slaves.

The Beeper, Keyboard, and Reset signals from the CRT Board are interfaced to the CPU Board through one half of the Extension Bus Board. The Extension Board also provides the link for the CPU's I/O and RS-232 to the Rear Panel Interface Board.

CPU Board communication consists of three RS-232 Ports. Two of these Ports (Port A and Port B) are multi-purpose ports which can be used to interface various peripheral equipment. The third RS-232 Port is dedicated to an external terminal (TeleVideo 905 Display Terminal or equivalent). When a terminal is connected to this port the Unidex 21 is fully operational from the terminal. (Certain functions such as the Function Key assignment, display of Graphics etc. are not available to the Terminal user.)

The I/O interface of the CPU Board is arranged such that an Opto- 22 PB24 board may be directly connected. The PB24 may then be equipped with various Opto 22 I/O modules. The Unidex 21 hardware defines Modules 0-7 as outputs and Modules 8-23 as inputs.

The CPU Board also provides three Interrupt Inputs accessable to the User. Two of these Interrupts are considered "User Interrupts" as they can be defined within a program to perform a User specified function. The third interrupt is the High Speed Interrupt and is dedicated to the purpose of data collection (e.g. PB24 Input Status).

2-3-2 INDEXER BOARD

The Indexing Board passes information to the CPU such as errors and messages to indicate when indexing is complete. The Indexing Board receives motion commands from the CPU and breaks them into data strings that digitally represent the trajectories necessary to make the requested move(s). The Indexer Board then transfers these trajectory commands through the Extension Bus Board to the DSP Board at 1 millisecond intervals.

The Indexer Board also contains the Input connections for a Trackball, two Joysticks and two Handwheels.

The MST Bus of the Indexer Board is a 16 bit output bus which is totally optocoupled. The data that is output on this bus is written with M (Miscellaneous), S (Spindle), or T (Toolchanger) commands which may be toggled while axes are in motion. Strobe signal outputs exist for the M, S, and T functions to latch the data into the receiving devices. There is also an Acknowledge (ACK-N) signal used for handshaking between the Indexer Board and the receiving device(s).

2-3-3 DSP BOARD

The DSP Board contains a Motorola 56001 Digital Signal Processor which controls the servo loops of the axes by the use of software alogrithms (PID loops). The alogrithms read the Encoder Feedback and generate current commands via D/A converters. By the use of an extension bus, the DSP receives operating parameters such as gain settings. Trajectory information is processed to set the commanded axes in motion. When trajectory commands end, the 56001 reverts to the role of maintaining axis position.

The DSP Board accepts Encoder and Limit data input from four axes through DSP connector P3. The Encoder inputs accept differential line driven, square wave Encoders. The DSP Board is configured for a standard quadrature Encoder. Clock and direction Encoders as well as CW and CCW Train Encoders may be used with appropriate jumper configurations.

The Limit inputs consist of a CW, CCW, and Home Limit line for each axis. The Limit inputs may be active high or active low. Polarity is established within the Unidex 21 parameters.

A variety of functions are interfaced to the DSP Board through the P2 connector. Of most significance are the Current Command output signals used to command the Axis Amplifiers. A shutdown signal that may be used for disabling the drives is also included in this signal group.

The P2 connector also contains the SCI (Serial Communication Interface) and SSI (Synchronous Serial Interface) ports for the DSP56001. To date these functions have not been implemented in the Unidex 21 System.

The DSP Board contains two opto-coupled interrupt sources that are interfaced through the P2 connector. One is a Fast Feedhold Input which is capable of stopping a current axis move in 1 milli-second as opposed to the implementation of a Feedhold command through the CPU Board, which may take up to 300 milli-seconds. The second interrupt to the DSP Board is the "Position-Grabbing" interrupt which, when activated, stores the current axis position and reports it to the CPU Board.

When a Brushless Motor System is used, the DSP Board's P2 connector contains the Hall Effect Switch Inputs. These switches (three (3) per axis) are located on the motors and output a six (6) step pattern which is read by the DSP56001 upon power-up to initialize motor position, thereafter the pattern is read on a 0.25 milli-second basis to provide accurate motor commutation.

Finally, the P2 connector provides the Vertical Axis Brake interface. Two lines are provided for this function. The first is a Brake Output line which activates a Brake whenever a fault is encountered that causes the vertical axis motor to lose torque. The second Brake line is an input to the DSP Board that signals the DSP56001 Brake/Axis integrity.

2-3-4 CRT AND FRONT PANEL BOARD S (OPTIONAL)

Both the CRT and Front Panel Boards are necessary if the User desires to operate the Unidex 21 from something other than a TeleVideo 905 Display Terminal. The CRT Board is capable of driving four types of displays; an Electo-Luminescent display, an IBM EGA 640x400 display, a monochrome display and an IBM EGA 640x350 display. The CRT Board provides either an EIA character set or an IBM character set.

The output lines of the CRT Board's P3 connector are the outputs of differential line drivers to provide noise immunity. The J3 connector of the Front Panel Board provides line receivers to return these signals to single ended TTL levels. It also provides a 16 pin locking header for interfacing to the EL Display, a 9 pin D style connector for interfacing directly to an EGA Monitor, and a 10 pin locking header which provides a connection to the Rear Panel Interface Board for connecting a EGA Monitor to the Unidex 21 as an external display.

The P4 connector of the CRT Board provides a line driver/line receiver interface to the Disk Drive via the Front Panel Board. The signals present on this connector are differential versions of the signals on the CPU P3/CRT P5 connection. This connector also provides the Front Panel System Reset and Keyboard Clock Lines in differential format that run through line receivers and return to the CPU Board.

The Front Panel Board provides connections for an IBM PC/AT keyboard and connections and decoding circuitry for the Front Panel Membrane Keyboard. Also contained on the Front Panel Board is the Disk Drive Interface complete with Disk Drive power connections. The Beeper and Reset Switch are routed through the Front Panel Board along with the switch and lamp signals for Power On/Off.

2-3-5 AXIS AMPLIFIERS

The type of Axis Amplifier included in the Unidex 21, corresponds to the type of Servo Motor used for each axis. The Unidex 21 is capable of controlling both DC Brush and AC Brushless motor types.

The DC Brush type motor uses a PWM DC Servo Amplifier. The DC Amplifier may be wired for up to 160 Volts PWM. A 20 Amp Peak, 10 Amp continuous version is available (DS16020) as well as a 30 Amp Peak, 15 Amp continuous version (DS16030). Future plans include a Linear DC Servo Amplifier rated at 30 Volts with 15 Amps Peak, 5 Amp continuous.

The AC Brushless type drive is not presently available.

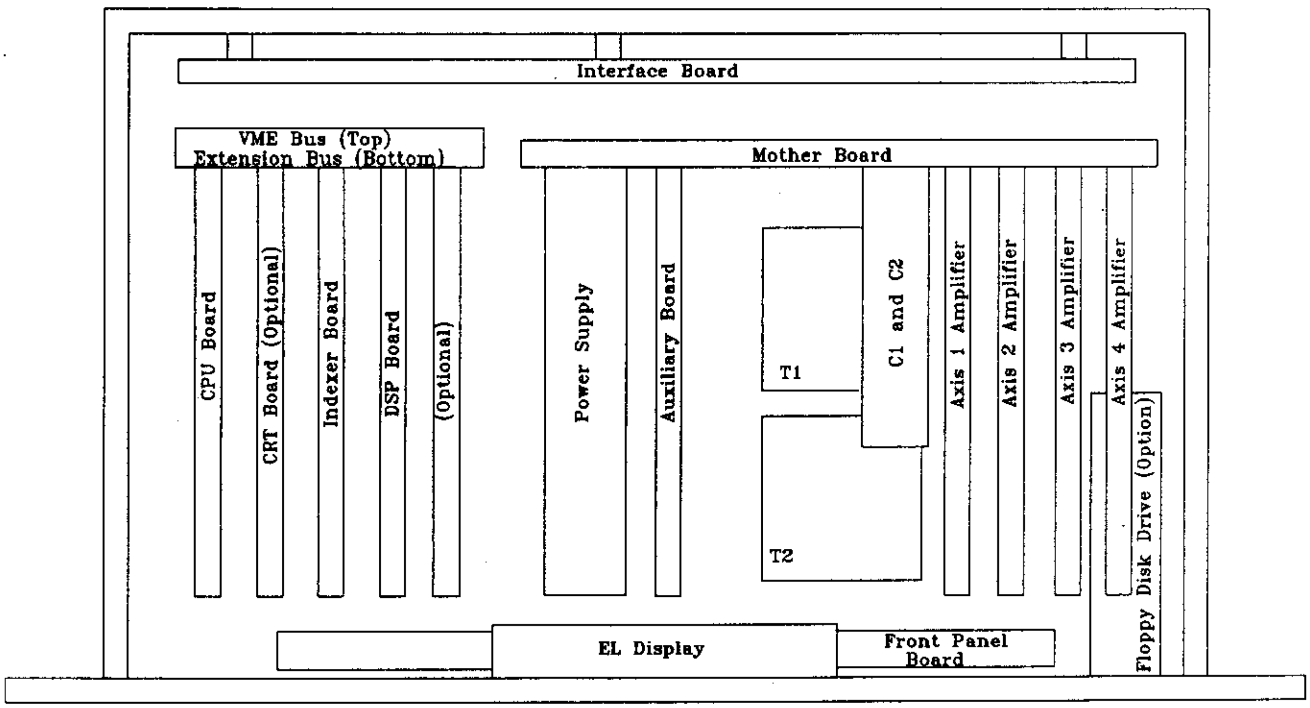


Figure 2-4: Unidex 21 Models U21R and U21B (Top View-Cover Removed)

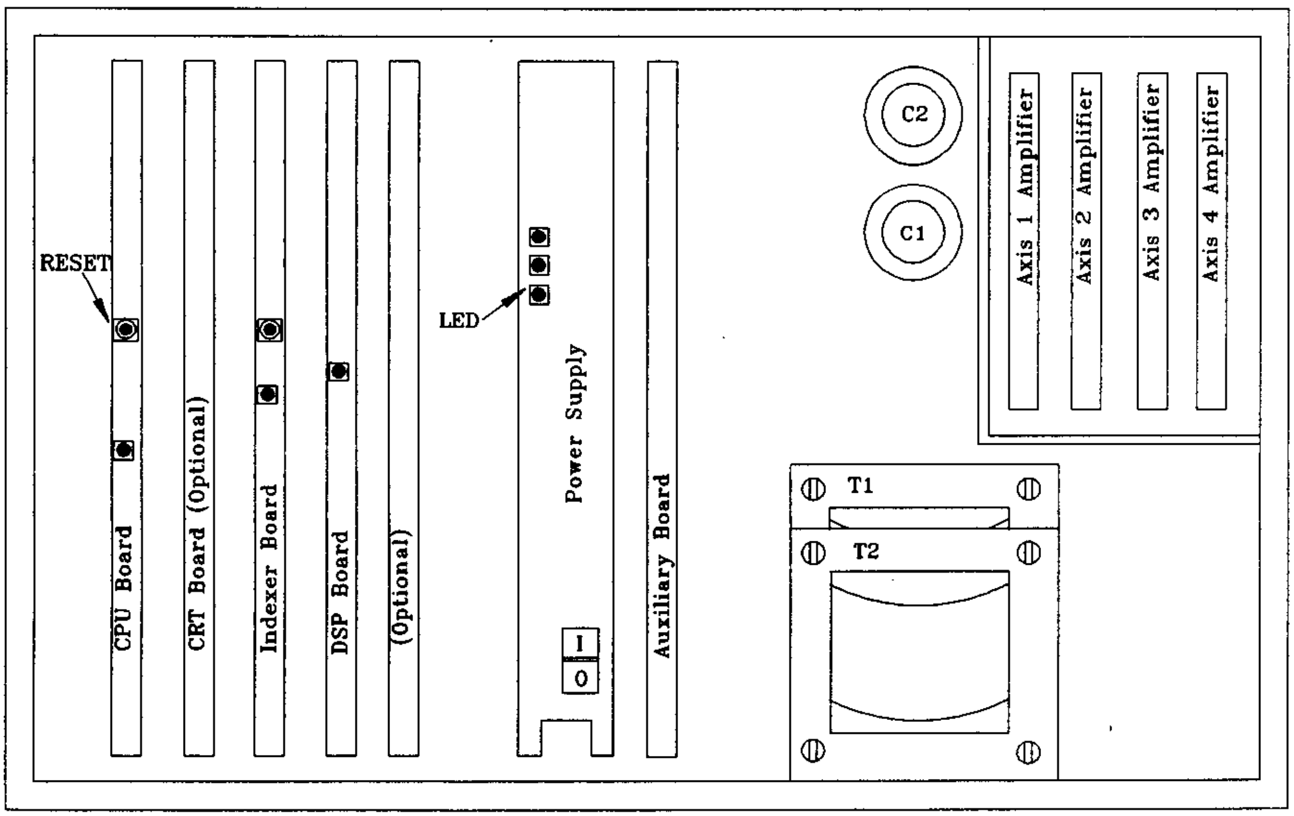


Figure 2-5: Unidex 21 Models U21R and U21B (Front View-Front Panel Removed)

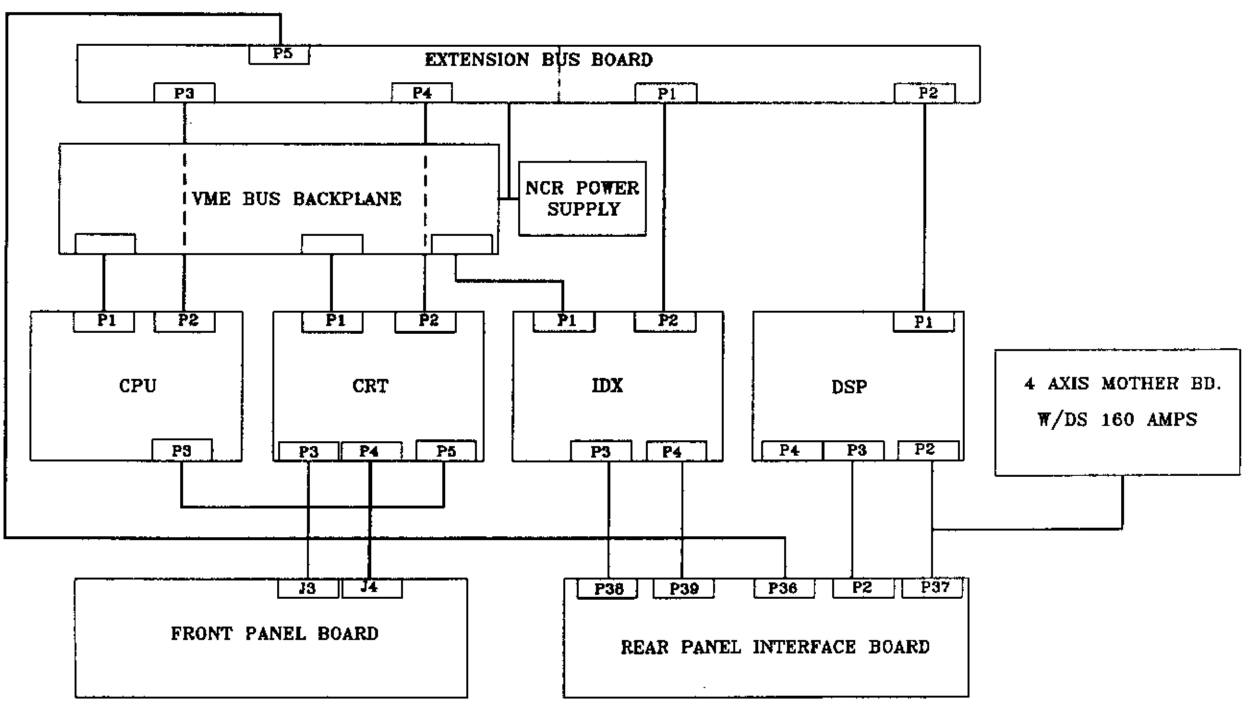


Figure 2-6: Unidex 21 System Block Diagram

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 3: JUMPERS AND SWITCHES

Many of the circuit boards that make up the Unidex 21 contain Jumpers and/or Switches. The following sections provide a board by board listing of the Jumpers and/or Switches, possible configurations and a brief explanation as to their function. Refer to Chapter 2, Figures 2-4 and 2-5 for relative board locations.

3-1: FRONT PANEL BOARD JUMPERS AND SWITCHES

The optional Front Panel Board (Figure 3-1) contains the Line Driver Receivers that provide interface between the CRT Line Drive Differential outputs and the following devices: EL or EGA Display, the Disk Drive, the Front Panel RESET, the Beeper, the Keyboard Clock, and the Keyboard Data signals.

The following Jumpers are located on the Front Panel Board:

JUMPER	CONDITIONS	FUNCTION
JP1	1-2, 3-4	EL Display Internal Clock Synchronization polarity set Negative
	1-4, 2-3	EL Display Internal Clock Synchronization polarity set Positive (Default)
JP2	1-2	Secondary EGA Display color signal not used. Connector J2, Pin1 is Shield Ground.
	2-3	Secondary EGA Display color signal used. Connector J2, Pin1 is Signal Ground. (Configured in conjunction with JP3 being 2-3) (Default)

JUMPER	CONDITIONS	FUNCTION
JP3	1-2	Disables secondary EGA Display color signal. Connector J2, Pin 2 is Signal Ground. (Configured in conjunction with JP2 being 1-2)
	2-3	Enables secondary EGA Display color signal. Connector J2, Pin 2 is Secondary Red.(Configured in conjunction with JP2 being 2-3) (Default)
JP4	1-2, 3-4	EL Display Vertical Sink Polarity set Negative (Default)
	1-4, 2-3	EL Display Vertical Sink Polarity set Positive (Default)

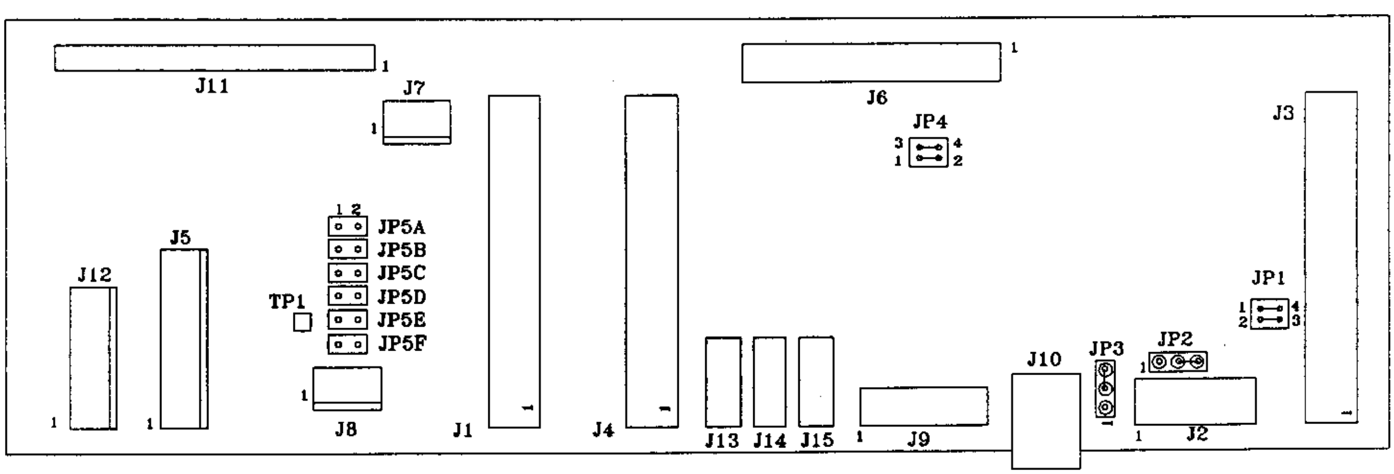


Figure 3-1: Unidex 21 Front Panel Board Jumpers and Switches

SECTION 3-2: CPU BOARD JUMPERS AND SWITCHES

The CPU Board is the Front End of the Unidex 21 System. It is directed by a Motorola 68000 Microprocessor. The CPU provides file editing, file management, floppy disk functions and RS-232 communications, as well as being the System's Bus Master. The CPU Board also contains eight (8) Opto-Isolated Outputs, sixteen (16) Opto-Isolated Inputs, and three (3) User Interrupts.

The following Jumpers are located on the CPU Board.

JUMPER	CONDITIONS	FUNCTION
JP1	1-2	External Battery operation. Configured in conjunction with JP2 being OUT. (An external Battery may be used to provide backup for the CPU and/or the Indexer Boards)
	2-3	Battery #1 is active (Configured in conjunction with JP2 being IN) (Default)
JP2	IN	Battery #1 is active (Configured in conjunction with JP1 being 2-3) (Default)
	OUT	External Battery operation. Configured in conjunction with JP1 being 1-2. (An external Battery may be used to provide backup for the CPU and/or the Indexer Boards)
JP3	1-2	Watchdog Input is active (Default)
	2-3	Watchdog Input not active (Factory Software Development Use Only)
JP4	1-2	High Write Current for Disk Drive
	2-3	Reduced Write Current for Disk Drive (Default)

JUMPER	CONDITIONS	FUNCTION
JP5	IN	All VME Bus Boards are affected by Reset (Default)
	OUT	Only CPU Board is affected by Reset
JP6	1-2	Pre-compensation value of 125 Nano-Seconds for Disk Drive
	2-3	Pre-compensation value of 187 Nano-Seconds for Disk Drive (Default)
JP7	1-2	High Speed Interrupt Disabled
	2-3	High Speed Interrupt Enabled (Default)
JP8	IN	RS-232 Clear to Send Termination to Ground, enables PC/AT Keyboard and Membrane input (Default)
	OUT	Disables PC/AT Keyboard and Membrane input (JP8 must always be installed)
JP9	IN	16 mHz Clock Signal sent out to System (Default)
	OUT	System will not operate (JP9 must be installed unless the unit is one of a multi-unit system and another board is configured to be the Master)
JP10	IN	Enables the Bus Busy Signal (Default)
	OUT	Disables the Bus Busy Signal

JUMPER	CONDITIONS	FUNCTION
JP11	1-2	CPU Board passes Interrupt Acknowledge Signal "In" to Interrupt Ackowledge "Out" (Default)
	2-3	CPU Generates Interrupt Acknowlege Signal "Out" that other VME Bus Boards receive as Interrupt Acknowledge "In"
JP12	1-2	Provides Bus arbitration, activating Bus Request (2) Signal
	2-3	Provides Bus arbitration, activating Bus Request (3) Signal (Default)
JP13	None Installed (Default)	Available to provide Bus arbitration to activate Bus Request (0) or Bus Request (1) Signals
JP14	IN	Enables Bus Grant In Signal (Default)
	OUT	Disables Bus Grant In Signal
JP15	None Installed (Default)	Available to provide Bus arbitration to activate Bus Grant "In" (0) or Bus Grant "In" (1) Signals
JP16	1-2	Provides Bus arbitration, activating Bus Grant "In" (3) Signal (Default)
	2-3	Provides Bus arbitration, activating Bus Gain "In" (2) Signal
JP17	None Installed (Default)	Available to provide Bus arbitration to activate Bus Grant "Out" (0) and Bus Grant "Out" (1) Signals

JUMPER	CONDITIONS	FUNCTION
JP18	1-2	Provides Bus arbitration, activating Bus Grant "Out" (3) Signal (Default)
	2-3	Provides Bus arbitration, activating Bus Grant "Out" (2) Signal
JP19	IN	Enables Bus Interrupt Request (6) Signal (Default)
	OUT	Disables Bus Interrupt Request (6) Signal
JP20	IN	Enables Bus Interrupt Request (5) Signal (Default)
	OUT	Disables Bus Interrupt Request (5) Signal
JP21	IN	Enables Bus Interrupt Request (4) Signal (Default)
	OUT	Disables Bus Interrupt Request (4) Signal
JP22	IN	Enables Bus Interrupt Request (3) Signal (Default)
	OUT	Disables Bus Interrupt Request (3) Signal
JP23	IN	Enables Bus Interrupt Request (2) Signal (Default)
	OUT	Disables Bus Interrupt Request (2) Signal
JP24	IN	Enables Bus Interrupt Request (1) Signal (Default)
	OUT	Disables Bus Interrupt Request (1) Signal

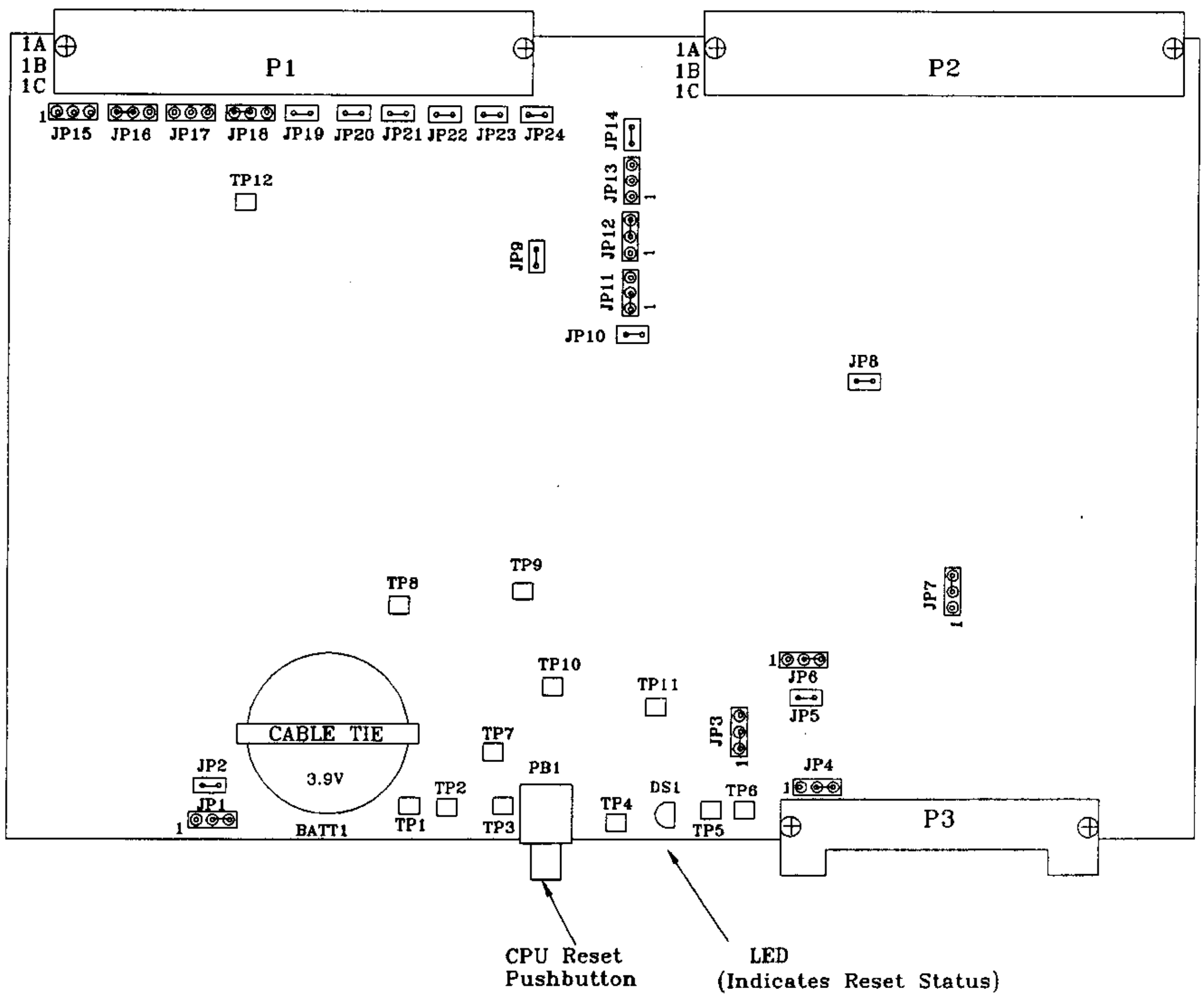


Figure 3-2 : Unidex 21 CPU Board - Jumpers and Switches

SECTION 3-3: INDEXER BOARD JUMPERS AND SWITCHES

The Indexer Board functions as the translator between the CPU and the DSP boards. It accepts motion commands from the CPU Board and converts them to trajectory commands which are a digital representation of the commanded move(s). The data is then transferred to the DSP Board which in turn outputs current commands that correspond to the trajectory commands.

The Indexer Board also provides auxilliary functions such as Joystick, Handwheel, and Trackball Inputs. The MST Bus for Output functions also is contained on the Indexer Board.

JUMPER	CONDITIONS	FUNCTION
JP1	1-2	Enables Watchdog Input Signal (Default)
	2-3	Disables Watchdog Input Signal (Factory Software Development Use Only)
JP2	1-2	Enables MST Bus RESET function (Default)
	2-3	Disables MST Bus RESET function
JP3	1-2	Determines the address range used by the CPU for one Indexer Board. (Configured in conjunction with JP5 being 1-2) (Default)
	2-3	(Not available for Unidex 21 Models U21B or U21R)
JP4	1-2	System RESET resets Indexing Board
	2-3	System RESET does not reset Indexing Board

JUMPER	CONDITIONS	FUNCTION
JP5	1-2	Determines the address range used by the CPU for one Indexer Board. (Configured in conjunction with JP3 being 1-2) (Default)
	2-3	(Not available for Unidex 21 Models U21B or U21R)
JP6	1-2	Battery #1 active (Configured in conjunction with JP7 being 1-2) (Default)
	2-3	Battery #1 is disabled. Configured in conjunction with JP7 being 2-3. (An external Battery may be used to provide backup for the Indexer and/or the CPU Boards)
JP7	1-2	Battery #1 is active (Configured in conjunction with JP6 being 1-2) (Default)
	2-3	Battery #1 is disabled. Configured in conjunction with JP6 being 2-3. (An external Battery may be used to provide backup for the Indexer and/or the CPU Boards)
JP8	1-10	Interrupt Request Signal (5) active (Default)
	2-9	Interrupt Request Signal (4) active (Not available at this time)
	3-8	Interrupt Request Signal (3) active (Not available at this time)
	4-7	Interrupt Request Signal (2) active (Not available at this time)
	5-6	Interrupt Request Signal (1) active (Not available at this time)

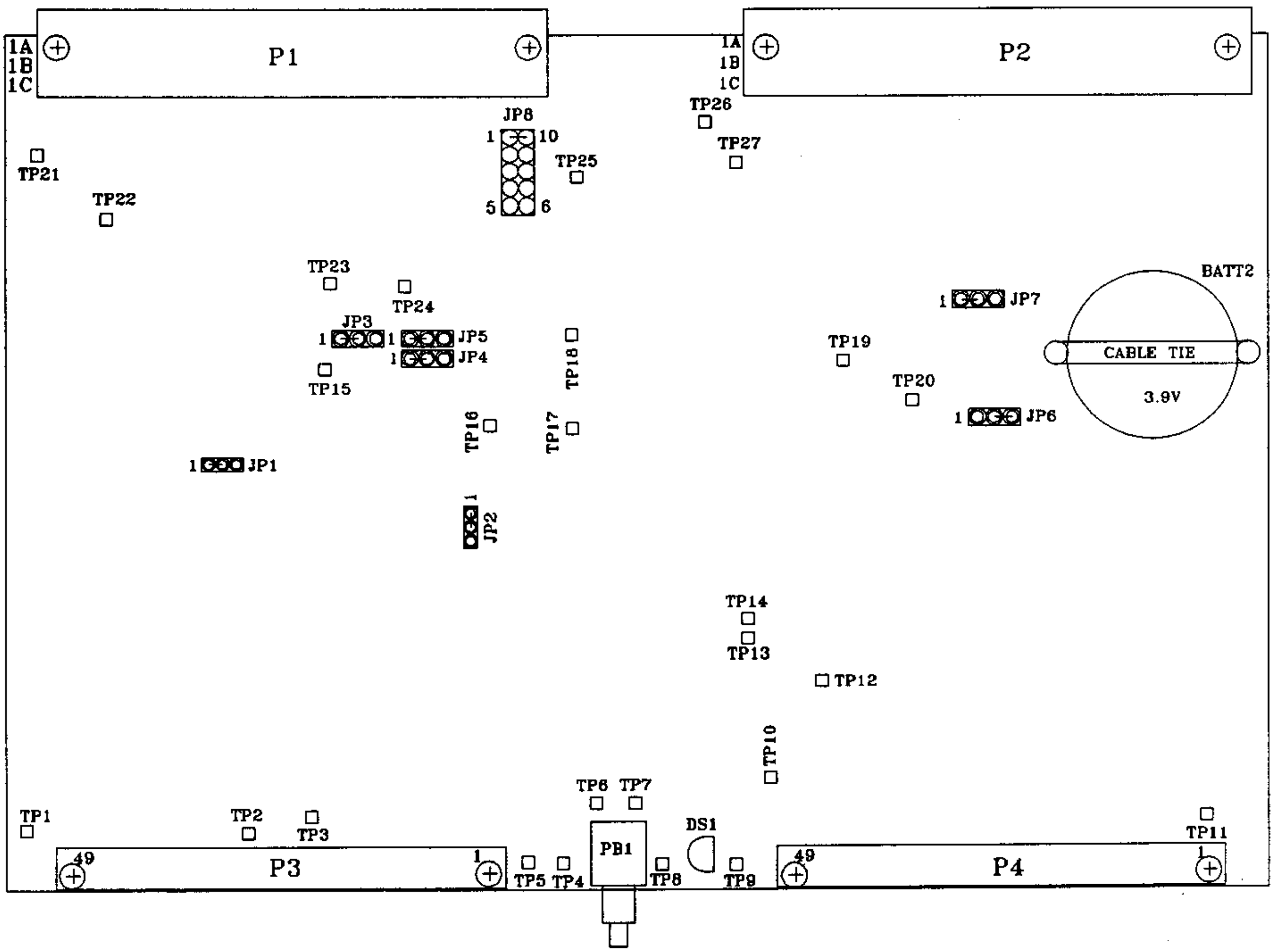


Figure 3-3: Unidex 21 Indexer Board - Jumpers and Switches

SECTION 3-4: DSP BOARD JUMPERS AND SWITCHES

The Unidex 21's DSP Board receives all motion information from the Indexer Board and converts this digital data to analog current commands which interface directly to the Power Amplifiers. The core of the DSP Board is the DSP56001 Digital Signal Processor. The exceptionally fast speed of this processor is due to it's single cycle instruction and parrallel processing abilities which are enhanced by being run on a 20MHz clock.

As well as generating current commands for eight axes of motors the DSP Board accepts Transducer signals for tracking the axes motion and monitors the Limit signals for each axis.

The following jumpers are located on the DSP Board:

JUMPER	CONDITIONS	FUNCTION
JP1	1-2	Enables Strobe Input Signal (Default)
	2-3	Disables Strobe Input Signal (Factory Software Development Use Only)
JP2	1-2	If Opto Isolator M15 is a HCPL2601, Pin 7 is enabled for output
	2-3	If Opto Isolator M15 is a 6N136, Pin 7 requires no connection (Default)
JP3	1-2	If Opto Isolator M13 is a HCPL2601, Pin 7 is enabled for output
	2-3	If Opto Isolator M13 is a 6N136, Pin 7 requires no connection (Default)
JP4 through JP19		Three Encoder types may be used. JP4 through JP19 must be configured in accordance with Figure 3-4

JUMPER	CONDITIONS	FUNCTION
JP21	1-2	If JP21, JP22 and JP23 are "1-2" the 24 bit word Static RAM capacity is 8Kx24
	2-3	If JP21, JP22 and JP23 are "2-3" the 24 bit word Static RAM capacity is either 32Kx24 or 128Kx24 (Default)
JP22	1-2	If JP21, JP22 and JP23 are "1-2" the 24 bit word Static RAM capacity is 8Kx24
	2-3	If JP21, JP22 and JP23 are "2-3" the 24 bit word Static RAM capacity is either 32Kx24 or 128Kx24 (Default)
JP23	1-2	If JP21, JP22 and JP23 are "1-2" the 24 bit word Static RAM capacity is 8Kx24
	2-3	If JP21, JP22 and JP23 are "2-3" the 24 bit word Static RAM capacity is either 32Kx24 or 128Kx24 (Default)
JP24 through JP26		Up to eight (8) DSP Boards may be used. JP24 through JP26 of each Board must be configured in accordance with Figure 3-5.

		ENCODER TYPE/JUMPER CONFIGURATION			
AXIS	JUMPERS	NONE	QUADRATURE SQUARE WAVE	CLOCK & DIRECTION	CW/CCW CLOCK
1	JP14 JP15	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3
2	JP10 JP11	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3
3	JP18 JP19	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3
4	JP6 JP7	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3
5	JP12 JP13	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3
6	JP8 JP9	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3
7	JP16 JP17	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3
8	JP4 JP5	1-2 1-2	2-3 1-2	1-2 2-3	2-3 2-3

Figure 3-4: Configuration of DSP Board Jumpers JP4 through JP19

DSP BOARD ADDRESS	JUMPER CONFIGURATIONS		
	JP24 (LEAST SIGNIFICANT BIT)	JP25	JP26 (MOST SIGNIFICANT BIT)
0	1-2	1-2	1-2
1	1-2	1-2	2-3
2	1-2	2-3	1-2
3	1-2	2-3	2-3
4	2-3	1-2	1-2
5	2-3	1-2	2-3
6	2-3	2-3	1-2
7	2-3	2-3	2-3

Figure 3-5: Configuration of DSP Board Jumpers JP24 through JP26

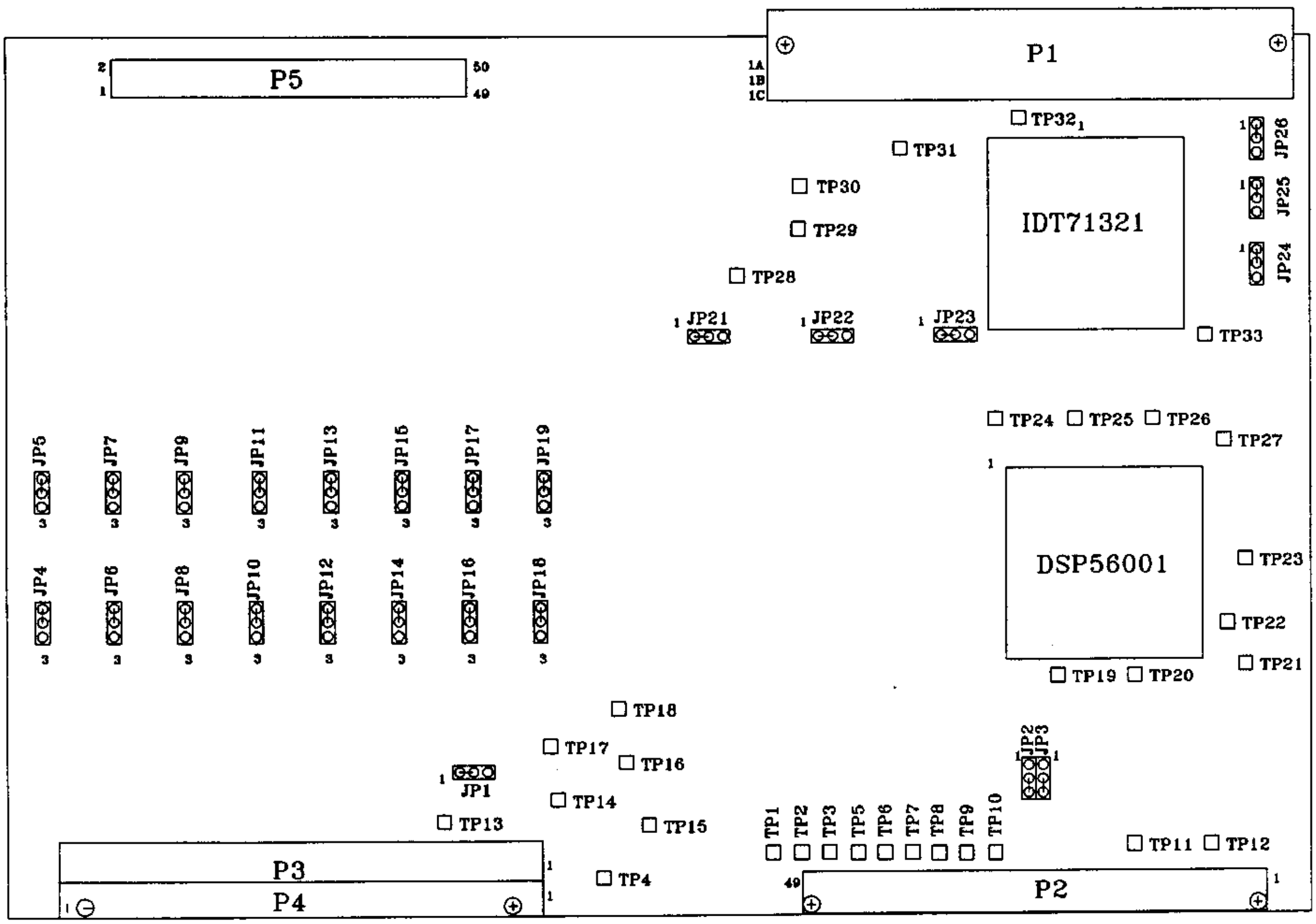


Figure 3-6: Unidex 21 DSP Board - Jumpers and Switches

SECTION 3-5: CRT BOARD JUMPERS AND SWITCHES

The Unidex 21's optional CRT Board provides several display functions. If the Unidex 21 is not equipped with the CRT Board, it is operable only by the use of a Televideo 905 Terminal. Equipped with the CRT Board, the Unidex 21 is capable of outputting a display to an Electro-Luminescent Panel or one of three monitor types: an IBM EGA 640x400, an IBM 640x350, or Monochrome.

The CRT Board also contains Line Drivers and Line Receivers for the optional Floppy Disk Drive and the Front Panel Keyboard lines.

The CRT Board contains a ten (10) position DIP Switch (SWN1) used for selecting the Character set, the Display type and for enabling the Front Panel Reset.

Switch configurations are as follows:

SWITCH NUMBER AND POSITION										
1	2	3	4	5	6	7	8	9	10	MODE
1	X	X	X	X	X	X	X	X	X	EIA CHARACTER SET
0	X	X	X	X	X	X	X	X	X	IBM CHARACTER SET
X	0	0	X	X	X	X	X	X	X	IBM EGA 640x350
X	0	1	X	X	X	X	X	X	X	IBM EGA 640x400
X	1	0	X	X	X	X	X	X	X	MONOCHROME
X	1	1	X	X	X	X	X	X	X	EL DISPLAY
X	X	X	X	X	X	X	X	1	X	FRONT PANEL RESET ENABLED
X	X	X	X	X	X	X	X	0	X	FRONT PANEL RESET DISABLED

1 = ON 0 = OFF X = NOT APPLICABLE

EXAMPLE:

1	2	3	4	5	6	7	8	9	10
1	0	1	X	X	X	X	X	1	X

EIA CHARACTER SET SELECTED
IBM EGA 640x400 MONITOR SELECTED
FRONT PANEL RESET ENABLED

SECTION 3-6: AUXILIARY BOARD JUMPERS AND SWITCHES

The Unidex 21's Auxiliary Board provides several functions. It contains a 24 VDC unregulated Power Supply to provide continuous power which is used to produce the logic for the Unidex 21's Power On/Off.

The Auxiliary Board also contains a Shunt Regulator to protect the Amplifier from damage due to increases in Bus voltage. The Shunt Regulator has been factory set for the appropriate voltage and must not be re-adjusted by the User.

Lastly, the Auxiliary Board monitors the Motor Contactors and the AC line current initiating the AC Fail error message when appropriate.

The following Jumpers are located on the Auxiliary Board:

JUMPER	CONDITIONS	FUNCTION
JP1	IN	If JP1 is "In", JP2 is "Out" and JP3 is "In", the System is configured for 115 VAC operation. (Default)
	OUT	* If JP1 is "Out", JP2 is "In" and JP3 is "Out", the System is configured for 230 VAC operation.
JP2	IN	* If JP2 is "In", JP1 is "Out" and JP3 is "Out", the System is configured for 230 VAC operation.
	OUT	If JP2 is "Out", JP1 is "In" and JP3 is "In", the System is configured for 115 VAC operation. (Default)

JUMPER	CONDITIONS	FUNCTION
JP3	IN	If JP1 is "In", JP2 is "Out" and JP3 is "In", the System is configured for 115 VAC operation. (Default)
	OUT	*If JP1 is "Out", JP2 is "In" and JP3 is "Out", the System is configured for 230 VAC operation.
JP4	1-2	Loss of Power to Motor Contactors will NOT Shut System Off. If JP4 is "1-2" and JP5 is "2-3", loss of Motor Contactor Power will cause an AC FAIL Message to be displayed. (Default)
	2-3	If JP4 is "2-3" and JP5 is "2-3" Motor Contactor Status is monitored. System will Shut Off if Motor Contactors Lose Power.
JP5	1-2	Motor Contactor Status is not monitored (Default)
	2-3	If JP4 is "2-3" and JP5 is "2-3" or if If JP4 is "1-2" and JP5 is "2-3", Motor Contactor Status is monitored and an AC Fail Fault message is reported to the CPU Board which in turn Displays an AC FAIL message on the Screen.

*NOTE: **If the Unidex 21 is changed from115 Volt operation to 230 Volt operation the DPC-20-220 Transformer must be changed to a DPC-24-180.**

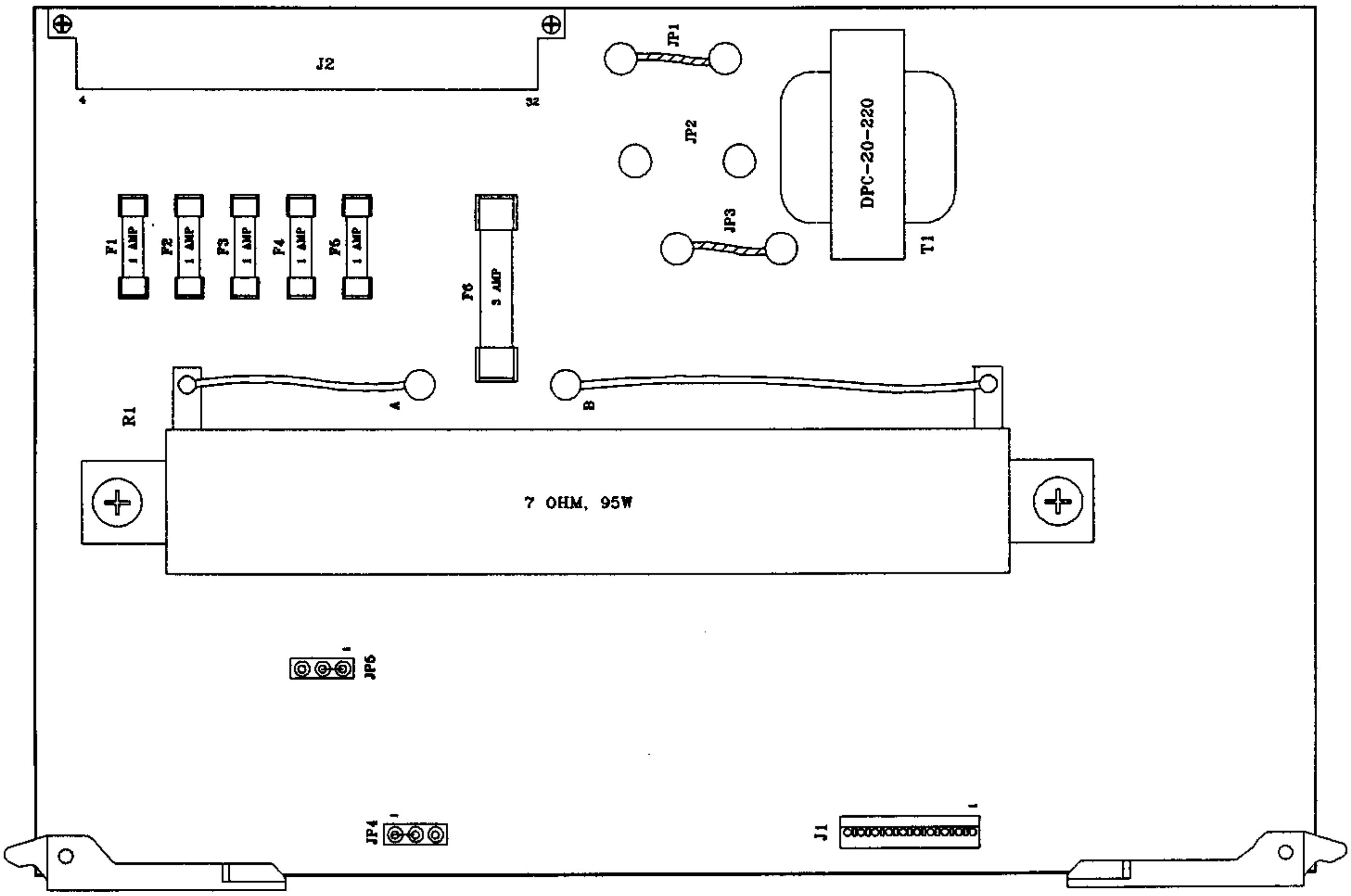


Figure 3-7 : Unidex 21 Auxiliary Board - Jumpers and Switches

CHAPTER 4: POWER INTERFACE

SECTION 4-1: POWER REQUIREMENTS

The Unidex 21, Models U21B or U21R may be factory wired for an input power of either 115 VAC, single phase, 50/60 Hz, 30 amps, 230 VAC, single phase, 50/60 Hz, 15 amps.

WARNING: **Located on the side panel of each Unidex 21 is a Power Requirement label. Deviation from the power specifications on the label could result in extensive damage to the equipment. Consult your Aerotech representative for further information.**

Prior to making any power connections refer to the Power Requirement label for input power specifications.

SECTION 4-2: INPUT POWER CONNECTION

Figure 4-1 shows an outline of the INPUT POWER receptacle. The Input Power cable is supplied with the Unidex 21.

WARNING: **Prior to making any electrical connections, make certain all POWER switches are in the OFF position.**

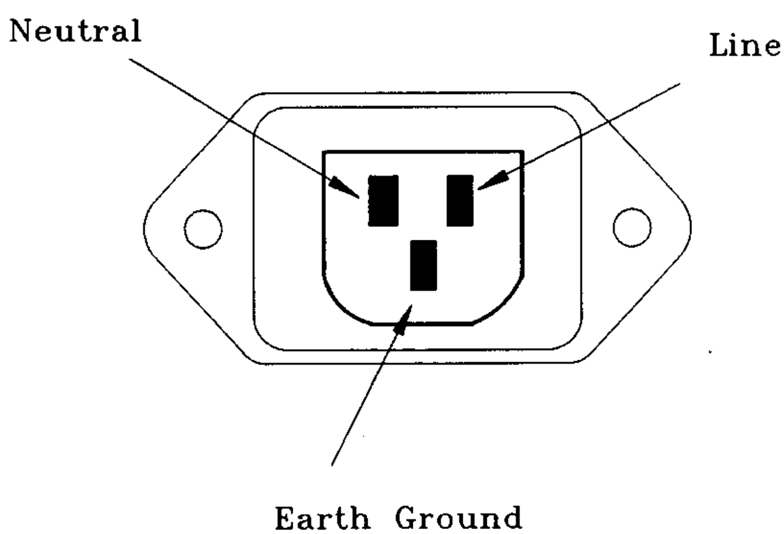


Figure 4-1: AC Input Power Connector

SECTION 4-3: POWER SYSTEM FUSES AND BATTERIES

The following is a list of all of the fuses incorporated in the Unidex 21 power system.

LABEL	PART NUMBER	RATING	LOCATION
F1	313005	10ASB	AC Input/Rear Panel (110V)
	XXXXXX	5ASB	AC Input/Rear Panel (230V)
1F1	Fuse rating determined by Motor type		Axis 1 Rear Panel Motor Connection
2F1	Fuse rating determined by Motor type		Axis 2 Rear Panel Motor Connection
3F1	Fuse rating determined by Motor type		Axis 3 Rear Panel Motor Connection
4F1	Fuse rating determined by Motor type		Axis 4 Rear Panel Motor Connection
F1-F5	218001	1ASB	Auxilliary Board AC Input
F6	313003	3AS 3ASB	Auxilliary Board Shunt
F1	XXXXXX	XXASB	In-Line T1
F2	XXXXXX	XXASB	In-Line T2
F1	XXXXXX	XXASB	NCR Power Supply
F1	XXXXXX	12ASB	Axis 1 Amplifier (DS16020)
		20ASB	Axis 1 Amplifier (DS16030)
F1	XXXXXX	12ASB	Axis 2 Amplifier (DS16020)
		20ASB	Axis 2 Amplifier (DS16030)
F1	XXXXXX	12ASB	Axis 3 Amplifier (DS16020)
		20ASB	Axis 3 Amplifier (DS16030)
F1	XXXXXX	12ASB	Axis 4 Amplifier (DS16020)
		20ASB	Axis 4 Amplifier (DS16030)
BATT1	BCX72 3850	3.9V	CPU Memory Backup
BATT2	BCX723850	3.9V	Indexer Memory Backup

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 5: MOTOR INTERFACE

The Rear Connector Panel of the Unidex 21, Models U21B and U21R contains the Motor Connectors for four axes. Details of the connectors are shown in Figures 5-1, 5-2 and 5-3.

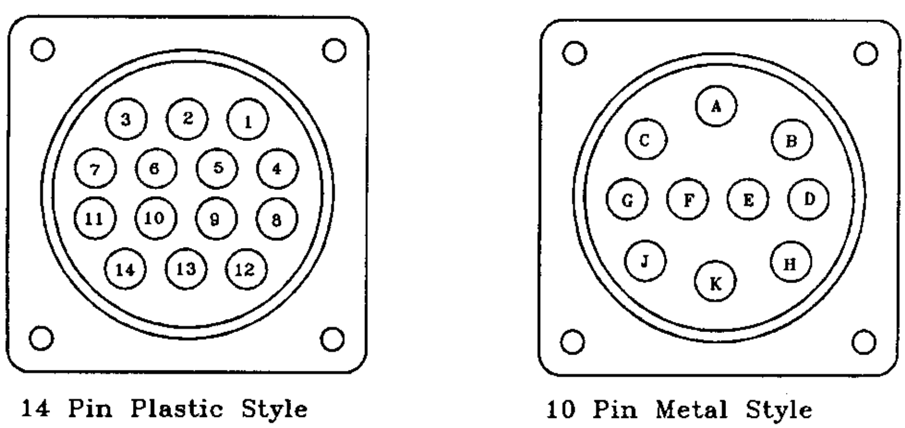


Figure 5-1: Unidex 21 Motor Output Receptacles

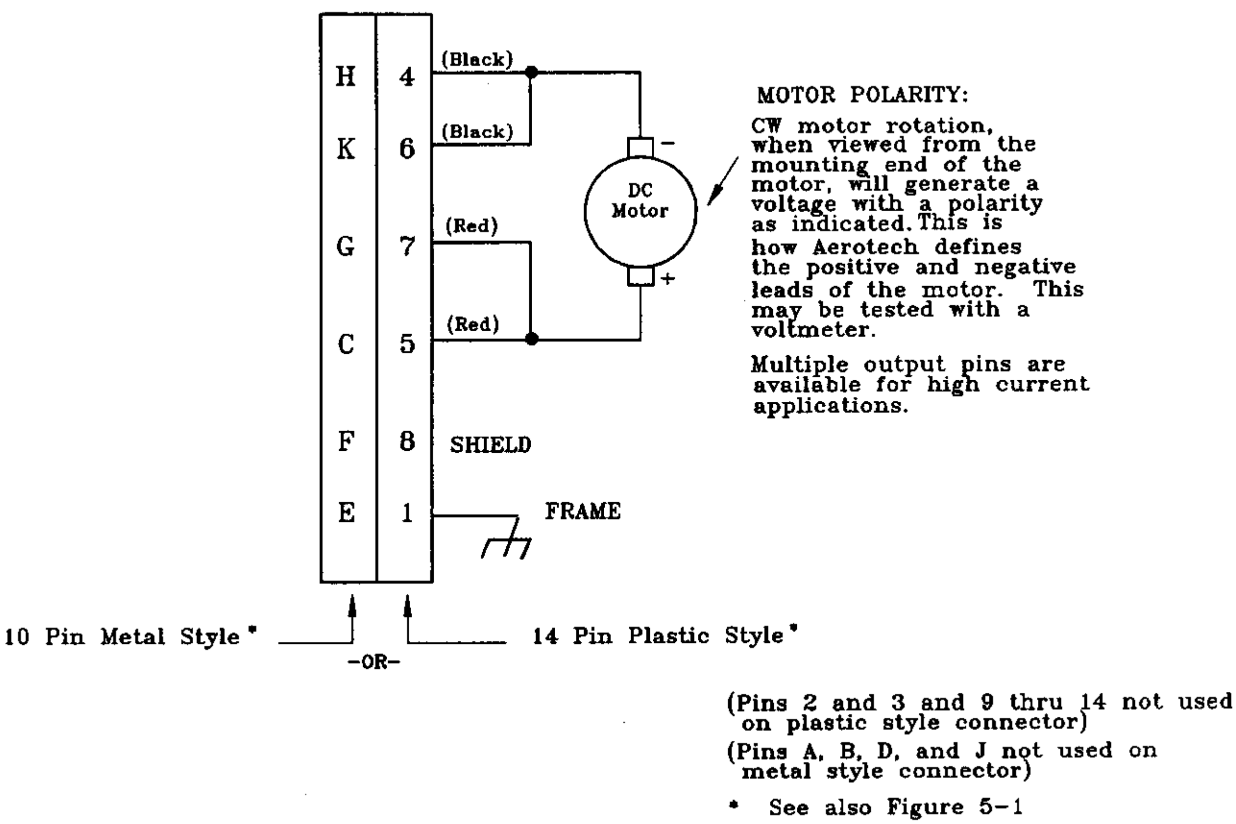
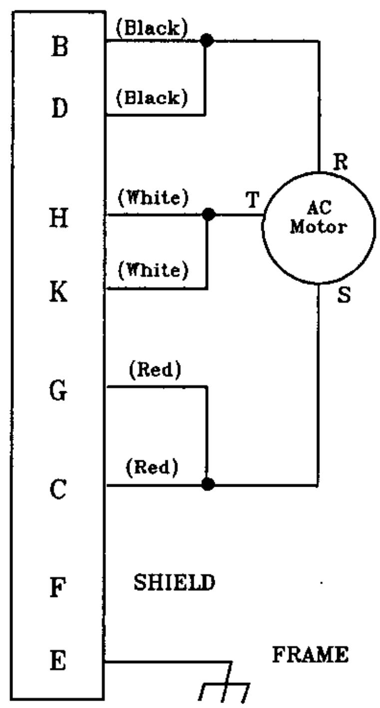


Figure 5-2: Outline of Connections for "Brush" Type DC Servo Motor



MOTOR POLARITY

CW Motor rotation (as viewed from the Motors mounting end) generates a specific Sine Wave pattern. Identification of this pattern defines the Motor's leads. This may be tested with the use of an Oscilloscope. See Figures 5-4 and 5-5.

Pins A and J not used

10 Pin Metal Style
(Standard for AC Brushless Motors)
See also Figure 5-1

Figure 5-3: Outline of Connections for "Brushless" Type AC Motor

SECTION 5-1: MOTOR PHASING

5-1-1 BRUSH TYPE DC SERVO MOTORS

The Motor is commanded to turn by the Servo Amplifier. The turning of the Motor generates a feedback voltage having an amplitude proportional to speed and a polarity representing direction. It is important that correct polarity is observed when the motor is connected to Aerotech equipment.

WARNING: **If the Motor is not properly connected, instability of the Servo Drive system may result in a runaway condition.**

The polarity of the Motor is in reference to Clockwise (CW) rotation of the Motor. This may be determined as follows:

Connect a Voltmeter to the Motor's terminals (or leads). Turn the Motor Shaft in a Clockwise (CW) direction. A positive indication will occur while the Motor Shaft is being turned if the correct polarity has been identified. If no indication or a negative indication is noted, the polarity of the Motor leads has not been identified correctly and the Motor connections must be reversed. Refer to Figure 5-4.

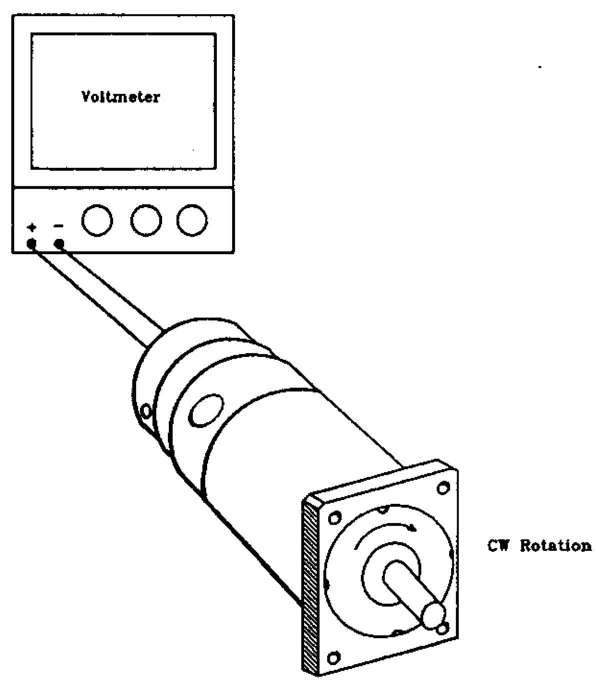


Figure 5-4: DC "Brush" Type Motor Phasing

5-1-1 AC BRUSHLESS TYPE SERVO MOTORS

Motor polarity must also be determined for Brushless type Motors. It is important that correct polarity is observed when the Motor is connected to Aerotech equipment.

WARNING: **If the Motor is not properly connected, instability of the Servo Drive system may result in a runaway condition.**

The polarity of the Motor is in reference to Clockwise (CW) rotation of the Motor. This may be determined as follows:

Connect an Oscilloscope to two of the Motor's leads. Turn the Motor Shaft in a Clockwise (CW) direction. The Oscilloscope indication should be as illustrated in Figure 5-5 (for Motors equipped with Hall Sensors) or Figure 5-6 (for Motors equipped with Resolvers or Encoders). If the Oscilloscope indication is not in accordance with Figures 5-5 or 5-6 or no indication is noted, the polarity of the Motor leads has not been identified correctly and the Motor connections must be changed.

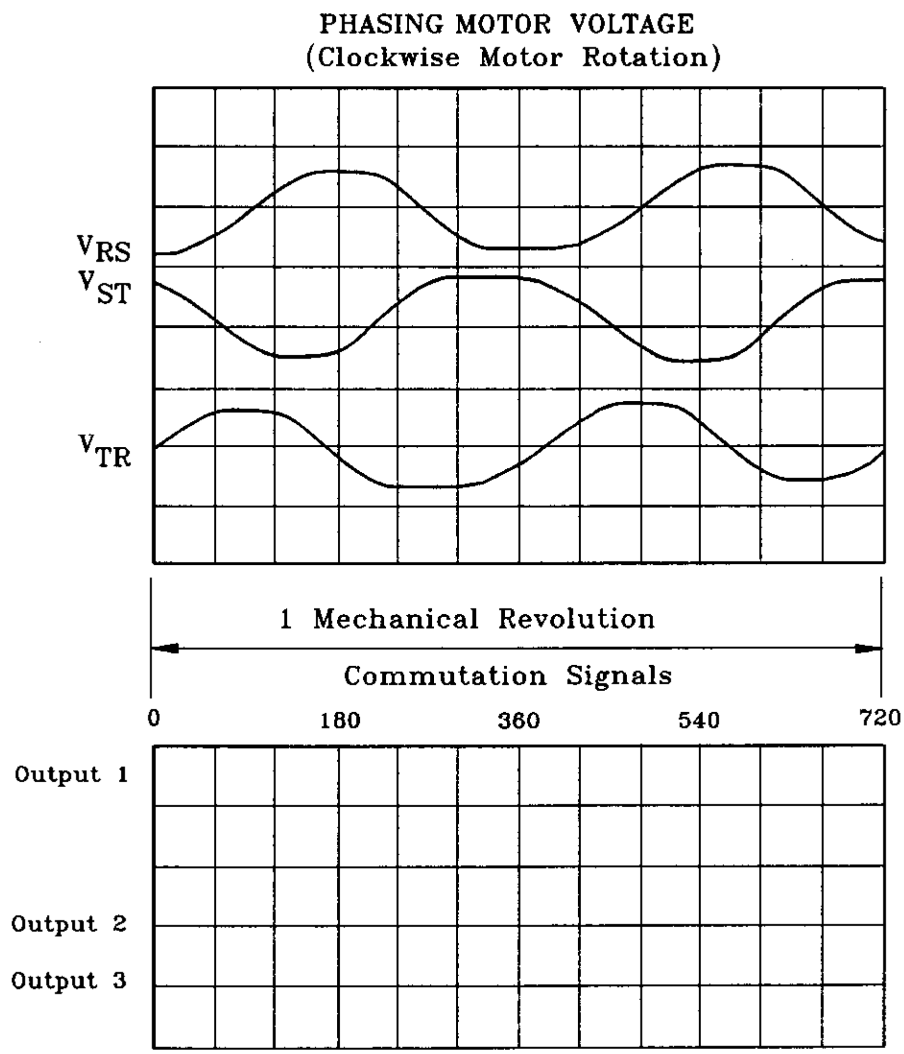


Figure 5-5: "Brushless" Motor Equipped With Hall Sensors

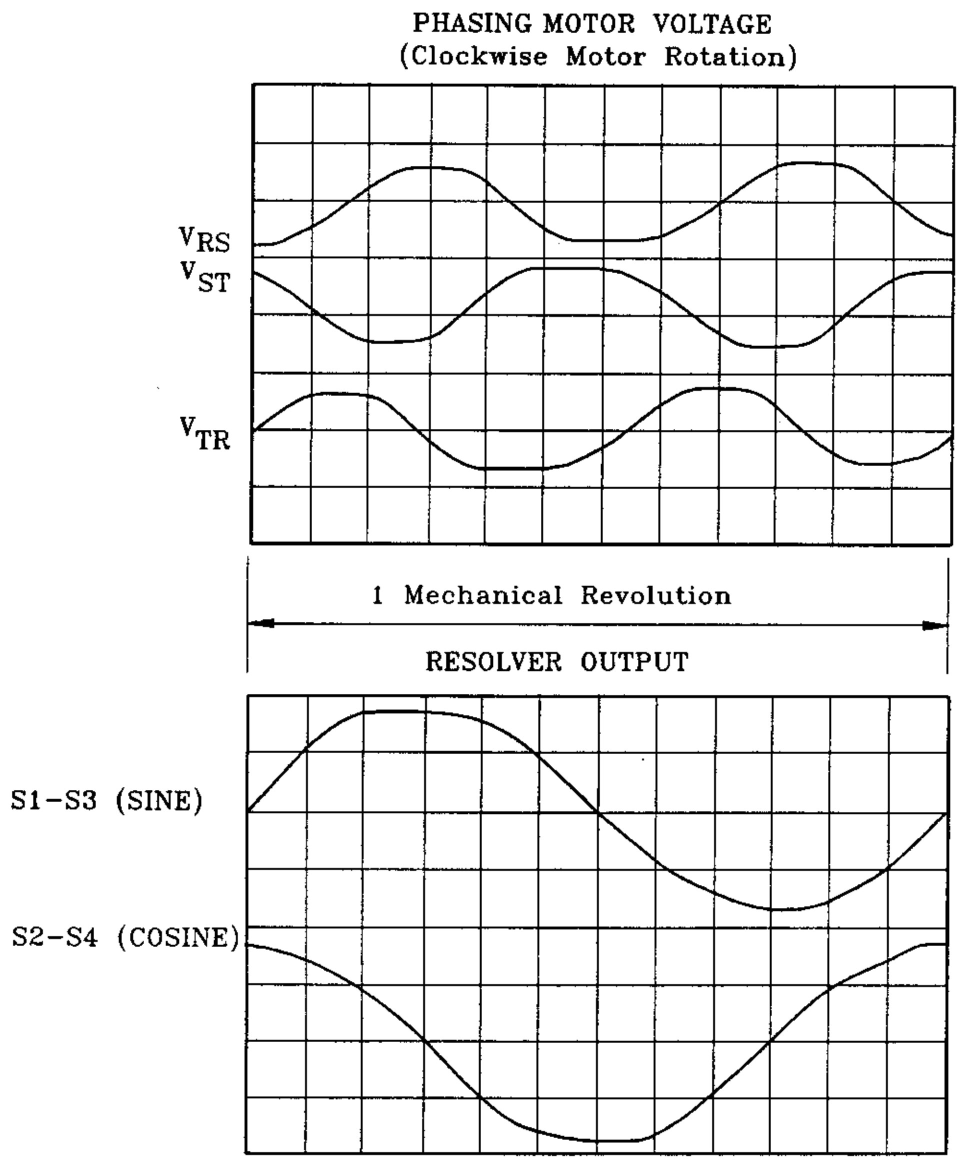


Figure 5-6: "Brushless" Motor Equipped With Resolver or Encoder

SECTION 5-2:MOTOR FUSING AND CURRENT LIMIT

Motor fusing and current limit are provided to protect the Motor from damage due to overheating caused by excessive peak current conditions.

The Motor Fuse (located on the Servo Controller) protects the Motor from overheating. Fuse rating is determined by the continuous current rating or the torque rating of the Motor.

The current limit adjustment on the Servo Controller is determined by the peak torque or peak current rating of the Servo Motor.

5-2-1 "BRUSH" TYPE DC SERVO MOTORS

The following is a list of some standard Aerotech DC Servo Motors and suggested Fuse and Current Limit ratings.

MOTOR	FUSING	CURRENT LIMIT
1017-01	4ASB	16A
1035-01	4ASB	16A
1050-01	5ASB	20A
1075-01	5ASB	20A
1135-01	5ASB	20A
1210-01	5ASB	20A
1410-03	8ASB	30A
1580-02	8ASB	30A
1960-02	12ASB	30A

5-2-2: "BRUSHLESS" TYPE AC SERVO MOTORS

The following is a list of some standard Brushless AC Servo Motors and suggested Fuse and Current Limit ratings.

MOTOR	FUSING	CURRENT LIMIT
(Not currently available)		

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 6: LIMIT AND HOME SWITCH INTERFACE

The optically isolated Limit and Home Switch Inputs are interfaced to the Unidex 21 through the ENC/LMT Connectors (P7, P9, P13, and P15).

An ENC/LMT Connector is provided for each of the four axes. Details of a typical connector are shown in Figure 6-1. Electrical characteristics of a typical Limit or Home Switch Input are shown in Figure 6-2.

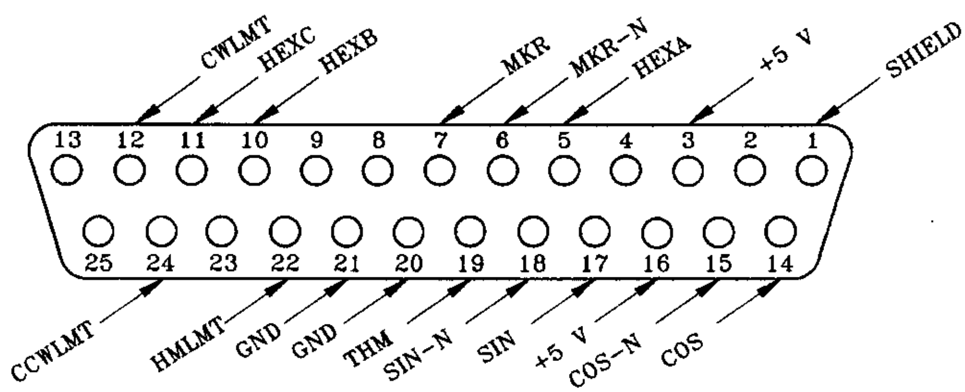


Figure 6-1: ENC/LMTS Connector (P7, P9, P13, P15)

PIN	DESCRIPTION
1	Encoder Shield connection. Sent to Chassis Ground unless otherwise specified.
2,4,8,9,13,23,25	(Not Used)
3	+5 V Input Common
5	Hall Effect Switch signal feedback to the DSP Board for Motor commutation (Active only when using a Brushless type Motor)
6	Inverse Marker pulse feedback from the Encoder to the DSP Board
7	Positive Marker pulse feedback from the Encoder to the DSP Board
10	Hall Effect Switch signal feedback to the DSP Board for Motor commutation (Active only when using a Brushless type Motor)
11	Hall Effect Switch signal feedback to the DSP Board for Motor commutation (Active only when using a Brushless type Motor)
12	CW * Limit Switch Input Signal to the DSP Board
14	Positive Cosine feedback from the Encoder to the DSP Board
15	Inverse Cosine feedback from the Encoder to the DSP Board.
16	+5 V Input Common
17	Positive Sine feedback from the Encoder to the DSP Board
18	Inverse Sine feedback from the Encoder to the DSP Board
19	Motor Thermistor feedback providing motor temperature data to the R/D Board (Active only when using a Brushless type Motor)
20,21	Ground, common to all Encoder connectors
22	Home Limit Switch Input signal to the DSP Board
24	CCW * Limit Switch Input Signal to the DSP Board

* See Section 6-3 for Limit Switch/Direction relationship

SECTION 6-1: LIMIT OR HOME SWITCH INPUTS

Aerotech supplied Limit and Home Switches are of the Normally Open, Logic 0 type (Refers to Contact positions when not in Limit) (see Figure 6-2).

If Normally Closed, Logic 1, Limit and/or Home Switches are used , the Axis Parameter "28: Home Switch is Normal Open?" and /or "30: Limit Switch is Normal Open", must be reconfigured. (See Chapter 6 of the *Unidex 21 User's Manual*)

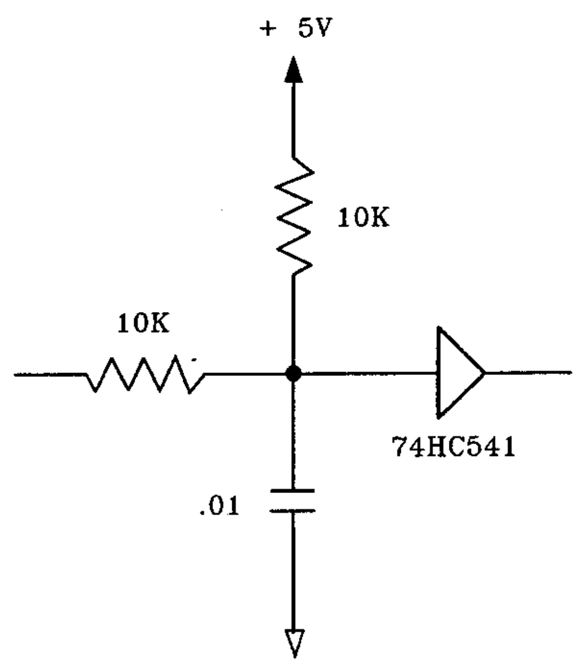


Figure 6-2: Electrical Characteristics of a Typical Limit Switch or Home Input

SECTION 6-2: LIMIT SWITCH DIRECTION

The CW and CCW Limit Switches provide travel limit protection for the Unidex 21 Drive System. Clockwise (CW) and Counterclockwise (CCW) motor rotation are referenced as viewed from the mounting flange of the motor. Aerotech has established this standard to clarify the Limit direction when one controller is interfaced to several mechanical configurations. The CW Limit stops CW motor rotation and the CCW Limit stops CCW rotation. (See Figure 7-6.)

Aerotech tables are manufactured such that the CW and CCW Limit Switches precede the rubber mechanical stops by approximately one revolution of the ballscrew. It is not recommended that the table be allowed to come into contact with the mechanical stops.

WARNING: **Damage to the equipment may occur if the table is allowed to come into contact with the mechanical stops.**

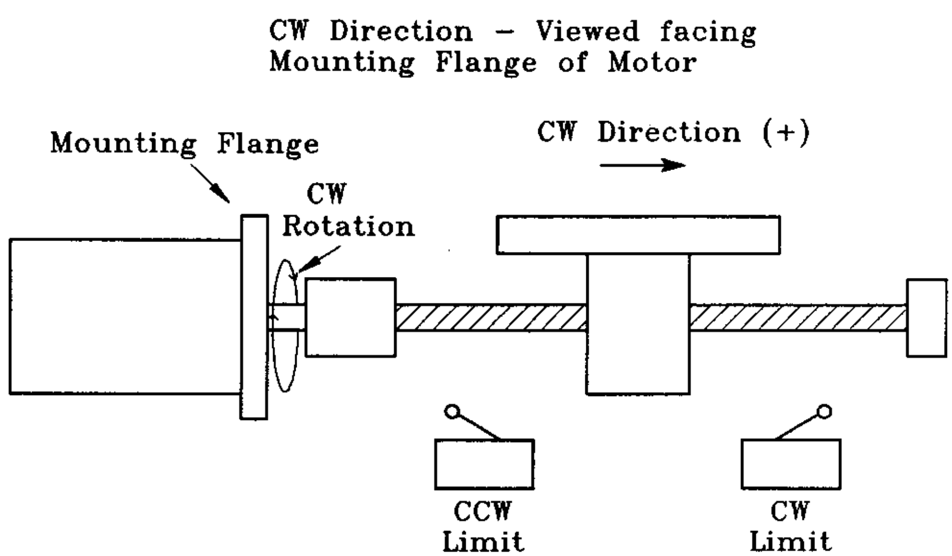


Figure 6-3: Limit Switch Direction

SECTION 6-3: HOME SWITCH POSITION

Most Aerotech Control systems are equipped with a cold-start reference point, which Aerotech refers to as the Home Position. This is the Machine Home and is a constant position depending on the marker location of a rotary or linear encoder.

When a Home command is issued, the motor will turn CCW (default) or CW (optional) until Home Limit Switch activation occurs, establishing the Home reference point. (Refer to the *Unidex 21 Programming Manual* for an explanation of the Home command and the *Unidex 21 User's Manual* for an explanation of Axis Parameters that affect motion to the Home Limit Switch position.) Upon Home Limit switch activation, the motor will reverse and rotate in the opposite direction until the Marker is encountered . (See the *Unidex 21 User's Manual* for an explanation of the Axis Parameters affecting the Home Limit Switch/Marker relationship.)

For most Rotary motion stages the Home Limit Switch is an independent switch and cam, incorporated into the system specifically for the Home cycle. When using a Linear motion stage, the CW or CCW Limit Switches may double as the Home Limit Switch. (See Section 6-4 of this Manual for Motor wiring details.)

Regardless of whether the Home reference point is established by a CW, CCW or Home Limit Switch, it's optimum position should be 1/2 revolution (± 1/4 revolution) from the location that the Switch opens (for a Normally Open type switch) to the leading edge of the marker pulse. (See Figure 6-3.)

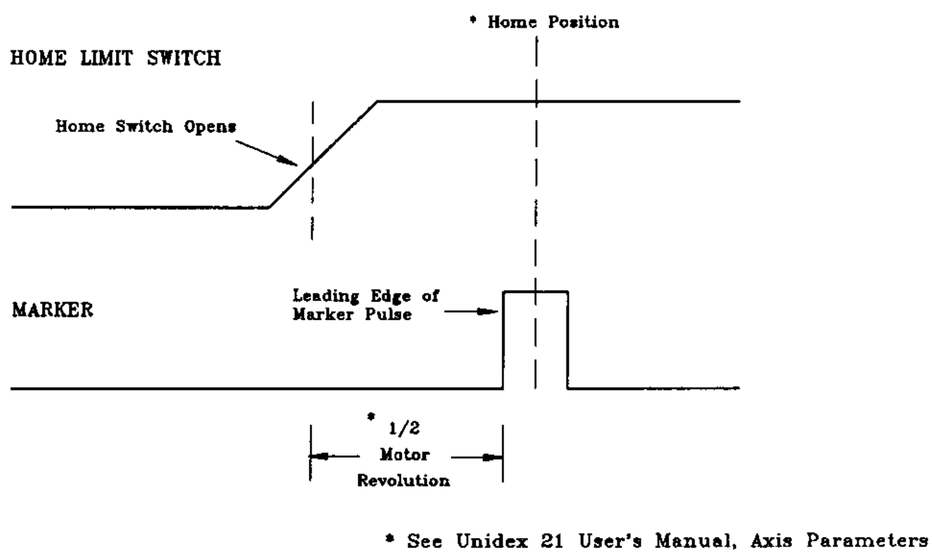


Figure 6-4: Home Switch Position

SECTION 6-4: MOTOR CONNECTIONS FOR HOME AND LIMIT SWITCHES

6-4-1: "BRUSH" TYPE DC SERVO MOTORS

6-4-2: "BRUSHLESS" TYPE AC SERVO MOTORS

CHAPTER 7: TRANSDUCER INTERFACE

Linear Encoders, Rotary Encoders, Resolvers, or Inductosyn ® type transducers may be used with the Unidex 21 Motion Controller to provide position feedback. The following Sections provide interface details for each transducer type.

SECTION 7-1: ENCODER INTERFACE

Linear and Rotary Encoders are interfaced to the Unidex 21 through the LINEAR ENC and the ENC/LMTS connectors. Phasing for the two types of Encoders is the same and is discussed in Section 7-1-1. Details concerning each of these connectors are provided in Sections 7-1-2 and 7-1-3.

7-1-1: ENCODER PHASING

Phasing for Encoders is as follows: as the Motor turns CW, the leading signal is the Cosine signal, the trailing signal is the Sine signal, and the Marker coincides with a positive Cosine signal (see Figure 7-1).

WARNING: **Improper phasing may cause a runaway condition which could result in personal injury or damage to the equipment.**

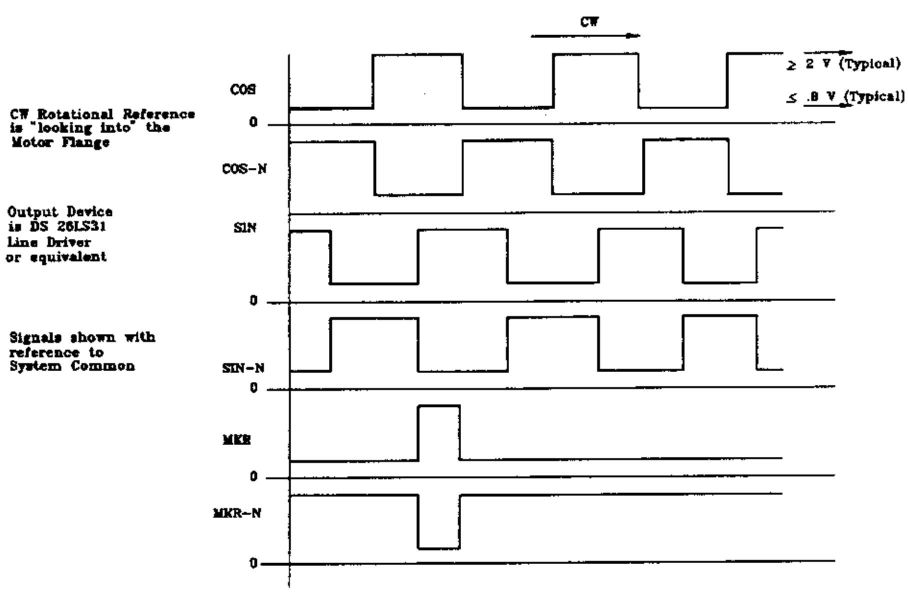
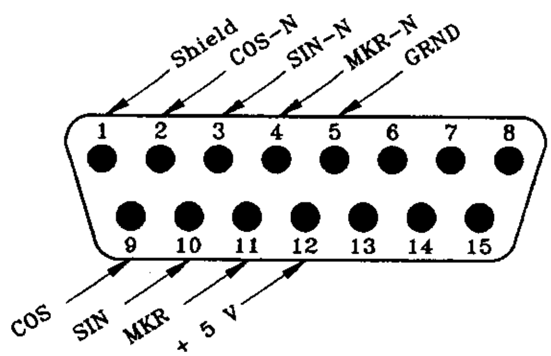


Figure 7-1: Encoder Signal Definition

7-1-2: LINEAR ENCODER INTERFACE (CONNECTORS P6, P8, P12 and P14)

The Linear Encoder Interface connectors provide for termination of basic control signals between the Unidex 21 and a Linear type Encoder. The Linear Encoder inputs are designed to accept complementary line driven Sine, Cosine and Marker signals.

A Linear Encoder connector is provided for each of the four axes. Details of a connector are shown in Figure 7-2. The Electrical Characteristics of a typical Linear Encoder input are shown in Figure 7-3. The four Encoder connectors are electrically the same.



PIN	DESCRIPTION
1	Linear Encoder Shield connection. Sent to Chassis Ground unless otherwise specified.
2	Inverse Cosine feedback from the Encoder to the DSP Board.
3	Inverse Sine feedback from the Encoder to the DSP Board
4	Inverse Marker pulse feedback from the Encoder to the DSP Board
5	Ground, common to all Encoder connectors
6,7,8,13,14,15	(Not Used)
9	Positive Cosine feedback from the Encoder to the DSP Board
10	Positive Sine feedback from the Encoder to the DSP Board
11	Positive Marker pulse feedback from the Encoder to the DSP Board
12	+5 V Input Common

Figure 7-2: LINEAR ENC Connector (P6,P8,P12 and P14)

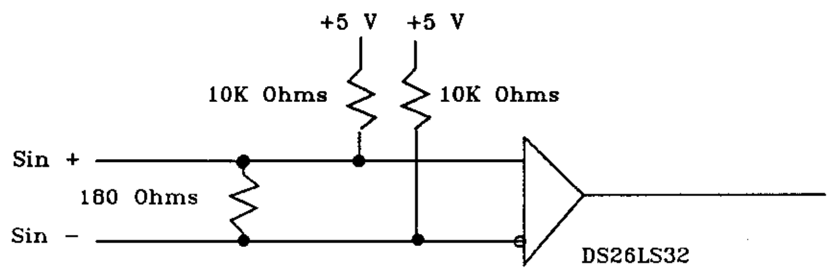


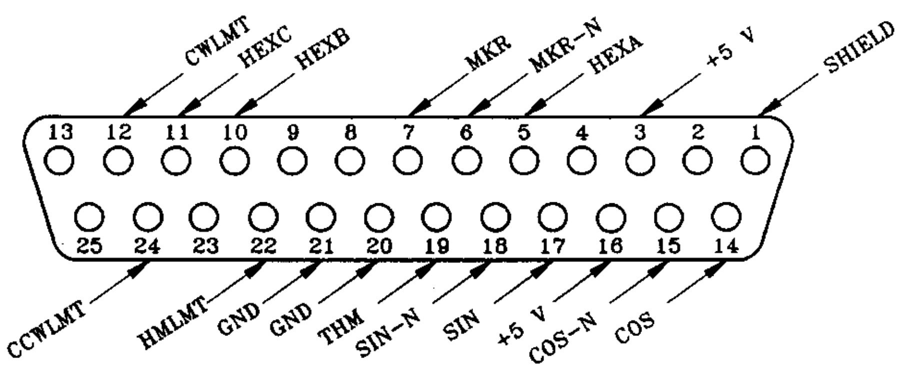
Figure 7-3: Electrical Characteristics of a Typical Linear Encoder Input

7-1-3: ENCODER/LIMIT INTERFACE (CONNECTORS P7, P9, P13 and P15)

The Encoder/Limit Interface connectors provide for termination of basic line driven control signals, optically isolated Motor travel limits, Motor commutation signals and Thermistor feedback.

An Encoder/Limit connector is provided for each of the four axes. Details of a connector are shown in Figure 7-4.

The Electrical Characteristics of a typical Limit Switch Input are shown in Figure 7-5. The four Encoder/Limit connectors are electrically the same.



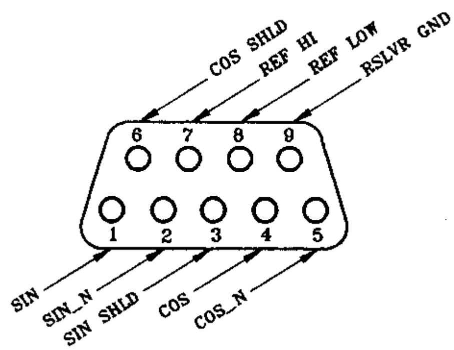
NOTE: **Refer to Chapter 6 of this manual for additional information-concerning these inputs.**

Figure 7-4: ENC/LMTS Connector (P7,P9,P13,P15)

SECTION 7-2: RESOLVER INTERFACE

The Resolver Connectors (P42,P43,P44,P45) provide termination of control signals between the Unidex 21 and a Resolver or Inductosyn.

A Resolver Connector is provided for each of the four axes. Details of the Resolver Interface connector are shown in Figure 7-5. Electrical characteristics of the connector are shown in Figure 7-6. The four Resolver Connectors are electrically the same.



PIN	DESCRIPTION
1	Positive Sine feedback from the Resolver to the R/D Board
2	Inverse Sine feedback from the Resolver to the R/D Board
3	Sent to Ground on the R/D Board unless otherwise specified
4	Positive Cosine feedback from the Resolver to the R/D Board
5	Inverse Cosine feedback from the Resolver to the R/D Board
6	Sent to Ground on the R/D Board unless otherwise specified
7	Output signal from R/D Board to Resolver or Inductosyn
8	Output signal from R/D Board to Resolver or Inductosyn (Currently tied to Ground)
9	Chassis Ground

Figure 7-5: RESOLVER Connector (P42,P43,P44,P45)

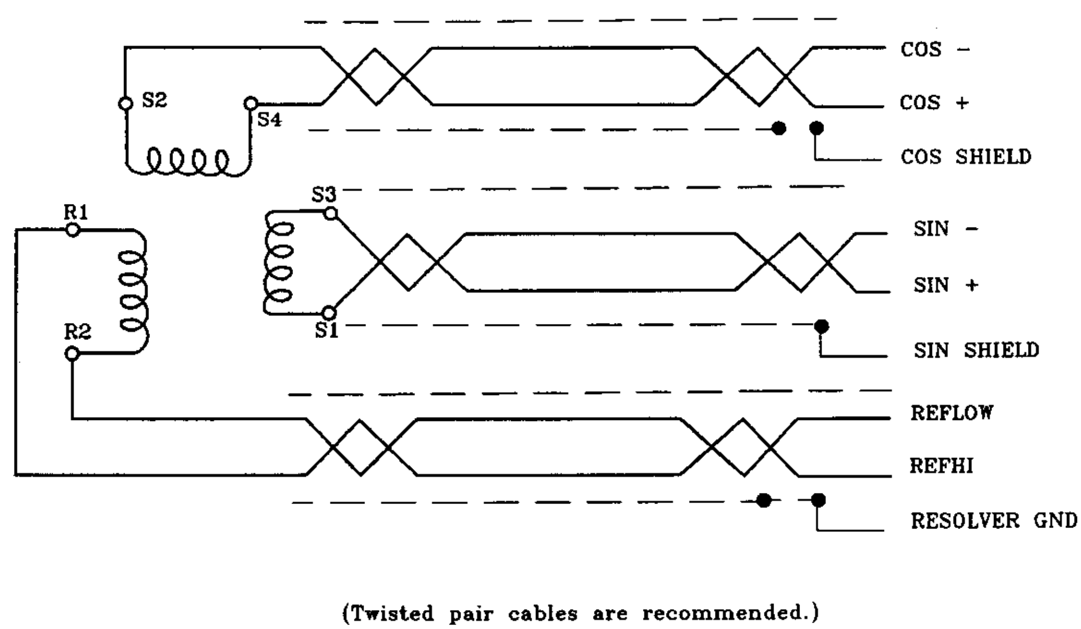


Figure 7-6: Electrical Characteristics of the RESOLVER Connector (P42,P43,P44,P45)

CHAPTER 8: INDEXING BOARD INTERFACE

The Indexer Board may be interfaced through two sets of connectors, the IDX MST Connector and the IDX INPUT 1 and the IDX INPUT 2 Connectors. A third connector IDX PORT B is currently reserved for future Indexer Board interface requirements.

SECTION 8-1: INDEXING MST BUS INTERFACE (P35)

Details of the IDX MST BUS (P35) connector are shown in Figure 8-1. Electrical characteristics of the connector are illustrated in Figure 8-2.

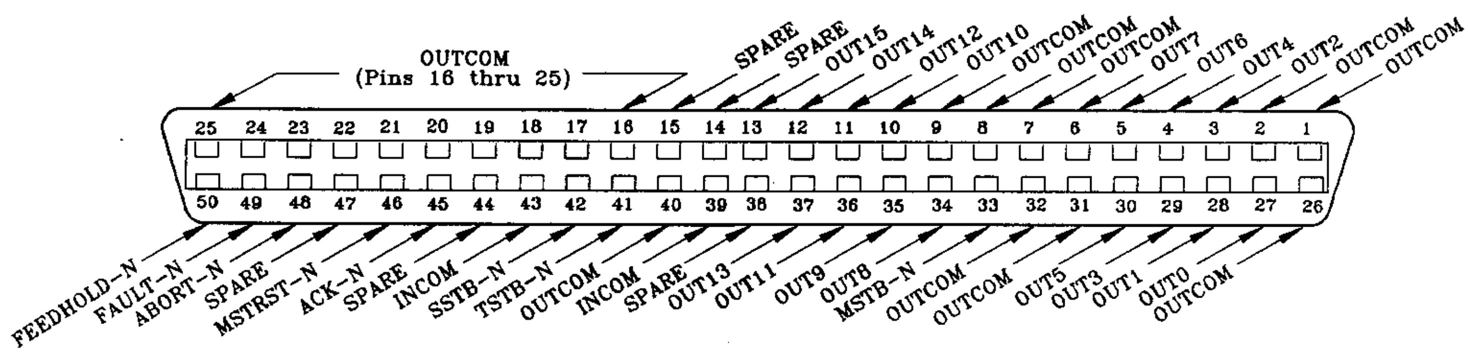


Figure 8-1: IDX MST BUS Connector (P35)

PIN	DESCRIPTION
1	Output Common *
2	Output Common *
3	Data Bit 2 Out
4	Data Bit 4 Out
5	Data Bit 6 Out
6	Data Bit 7 Out
7	Output Common *
8	Output Common *
9	Output Common *
10	Data Bit 10 Out
11	Data Bit 12 Out
12	Data Bit 14 Out
13	Data Bit 15 Out, Most Significant Bit
14,15,38,44	(Not Used)
16-26	Output Common *
27	Data Bit 0 Out, Least Significant Bit
28	Data Bit 1 Out
29	Data Bit 3 Out
30	Data Bit 5 Out
31	Output Common *
32	Output Common *
33	Qualifying output line for the M Strobe command signal. (See Chapter 6 of the *Unidex 21 User's Manual* for setting signal delay time.)
34	Data Bit 8 Out
35	Data Bit 9 Out
36	Data Bit 11 Out
37	Data Bit 13 Out
39	Input Common **
40	Output Common *
41	Qualifying output line for the T Strobe command signal. (See Chapter 6 of the *Unidex 21 User's Manual* for setting signal delay time.)

PIN	DESCRIPTION
42	Qualifying output line for the S Strobe command signal. (See Chapter 6 of the *Unidex 21 User's Manual* for setting signal delay time.)
43	Input Common **
45	Input line for the M, S and T Acknowledge Signal. (See Chapter 6 of the *Unidex 21 User's Manual* for setting Acknowledge delay time.)
46	Output signal derived from a Power Up or Front Panel System Reset to clear peripheral devices to their initial state.
47	(Not used)
48	(Not determined)
49	(Not Determined)
50	(Not Determined)

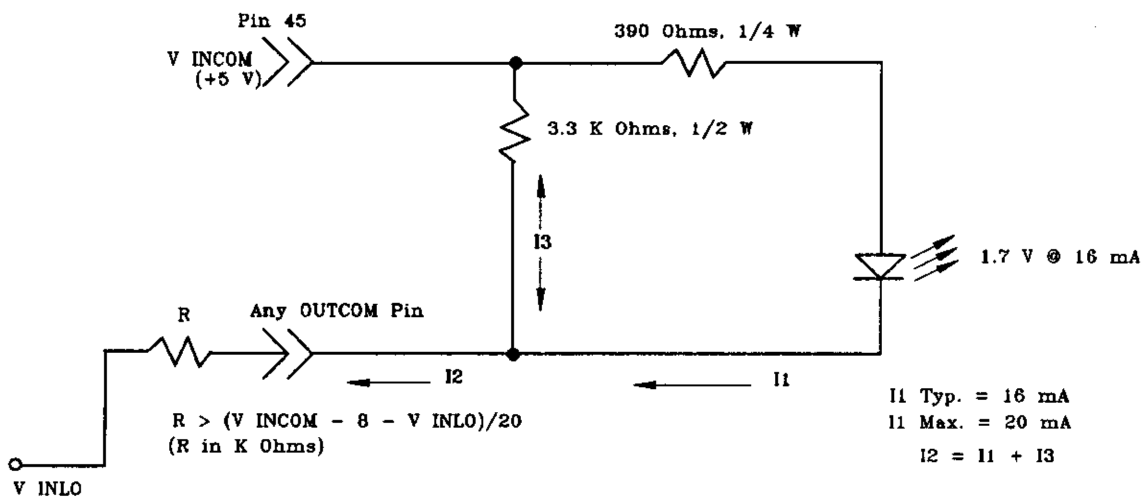
* Output Common is shared by all Indexing Board Outputs

** Input Common is shared by all Indexing Board Inputs.

ACK-N, ABORT-N, FAULT-N, FEEDHOLD-N (TLP621-4 Opto Coupler Input)

Example:

ACK-N (Typical for FAULT-N & FEEDHOLD-N)



EXAMPLE:

TTL Inputs

V INCOM = +5 V
I2 = 9 mA
R = 0

D0 THRU D15, MSTB-N, TSTB-N, SSTB-N & MSTRST-N OUTPUTS

Example:

D0 (Typical for D1 thru D15, MSTB-N, TSTB-N, SSTB-N & MSTRST-N)

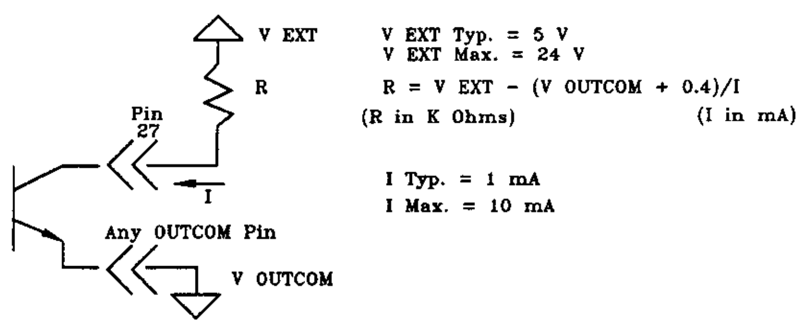
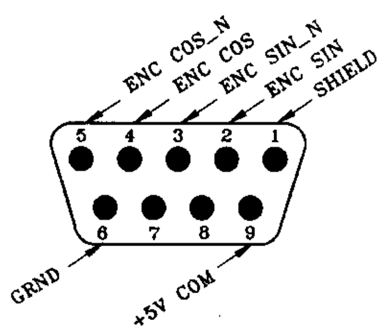


Figure 8-2: Electrical Characteristics of IDX MST Input Connector (P35)

SECTION 8-2: INDEXING INPUT 1 AND INDEXING INPUT 2 INTERFACE (P40,P41)

The IDX INP 1 and IDX INP 2 Connectors (P40 and P41) may be used to interface to Quadrature, CLK/DIR or CW/CCW type Transducers or to provide Handwheel or Auto-Focus Input. Refer to the *Unidex 21 User's Manual* for Parameter settings affecting the use of the IDX INP 1 and IDX INP 2 Connectors. The IDX INP 1 and IDX INP 2 Connectors are electrically identical.

Details of the IDX INP 1 and IDX INP 2 Connectors are shown in Figure 8-3. Electrical characteristics of the connectors are shown in Figure 8-4.



PIN	DESCRIPTION
1	Shield Connection (Sent to Chassis Ground unless otherwise specified.)
2	Encoder Sine Phase Input
3	Inverse Encoder Sine Phase Input
4	Encoder Cosine Phase Input
5	Inverse Encoder Cosine Input
6	Ground
7,8	(Not Used)
9	+5 V Input Common

Figure 8-3: IDX INP 1 and IDX INP 2 Connector (P40 and P41)

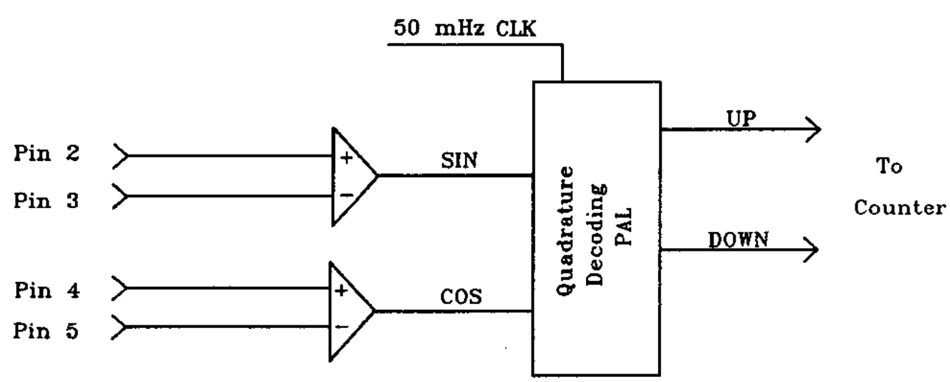


Figure 8-4: Electrical Characteristics of the IDX INP 1 and IDX INP 2 Connectors (P40 and P41)

CHAPTER 9: IEEE-488 INTERFACE

The Unidex 21's IEEE-488 interface connector (P11) permits a host computer to interactively control a Unidex 21. (Refer to the *Unidex 21 User's Manual*, Chapter 6 for Parameter settings affecting the use of the IEEE-488 Interface.)

IEEE-488 contains 8 data lines and 8 control lines. It can accomodate up to 14 devices and provides a Service Request line from all devices to the Bus Controller.

Details of the IEEE-488 connector are shown in Figure 9-1. Configurations for multiple device connections to the IEEE-488 Interface are illustrated in Figure 9-2.

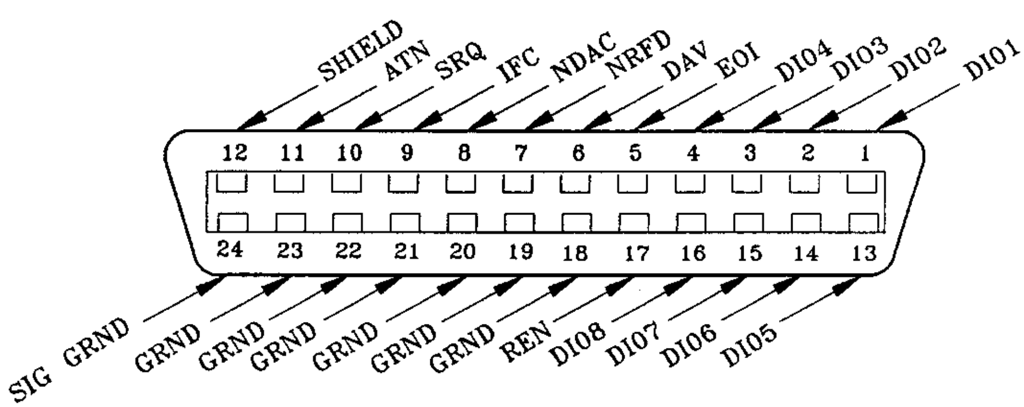


Figure 9-1: IEEE-488 Connector (P11)

PIN	DESCRIPTION
1,2,3,4 13,14,15,16	Used for sending data (ATN lines false) or bus interface messages (ATN line true)
5	End or Identify, once asserted indicates the termination of data flow. Asserted when the last data byte is placed on the Bus.
6	Data Valid, asserted by the Talker to indicate to all Listeners that data on the Bus is valid.
7	Not Ready For Data, when true, indicates to Talker that all Listeners are not ready for data.
8	Not Data Accepted, when true indicates to the Talker that all Listeners have not accepted the data placed on the Bus
9	Interface Clear, asserted only by the System Controller to place all devices in the unaddressed state. Devices go into the Talker idle, Listener state. If control has been passed to another device, the System Controller again becomes active by asserting IFC.
10	Service Request, asserted by a device to indicate a need for interaction with the Controller.
11	Attention, asserted true by active Controller to send Bus interface messages. When ATN is asserted, signals on the data lines are interpreted as messages. ATN asserted with EOI does a Parallel Poll. When ATN is false, data may be sent over the Bus by a designated Talker.

PIN	DESCRIPTION
12	Shield
17	Remote Enable, remotely asserted to program devices on the Bus. Any device addressed to listen while REN is true, is placed in the Remote Mode of operation.
18	Ground (Twisted Pair with DAV)
19	Ground (Twisted Pair with NRFD)
20	Ground (Twisted Pair with NDAC)
21	Ground (Twisted Pair with IFC)
22	Ground (Twisted Pair with SRQ)
23	Ground (Twisted Pair with ATN)
24	Signal Ground

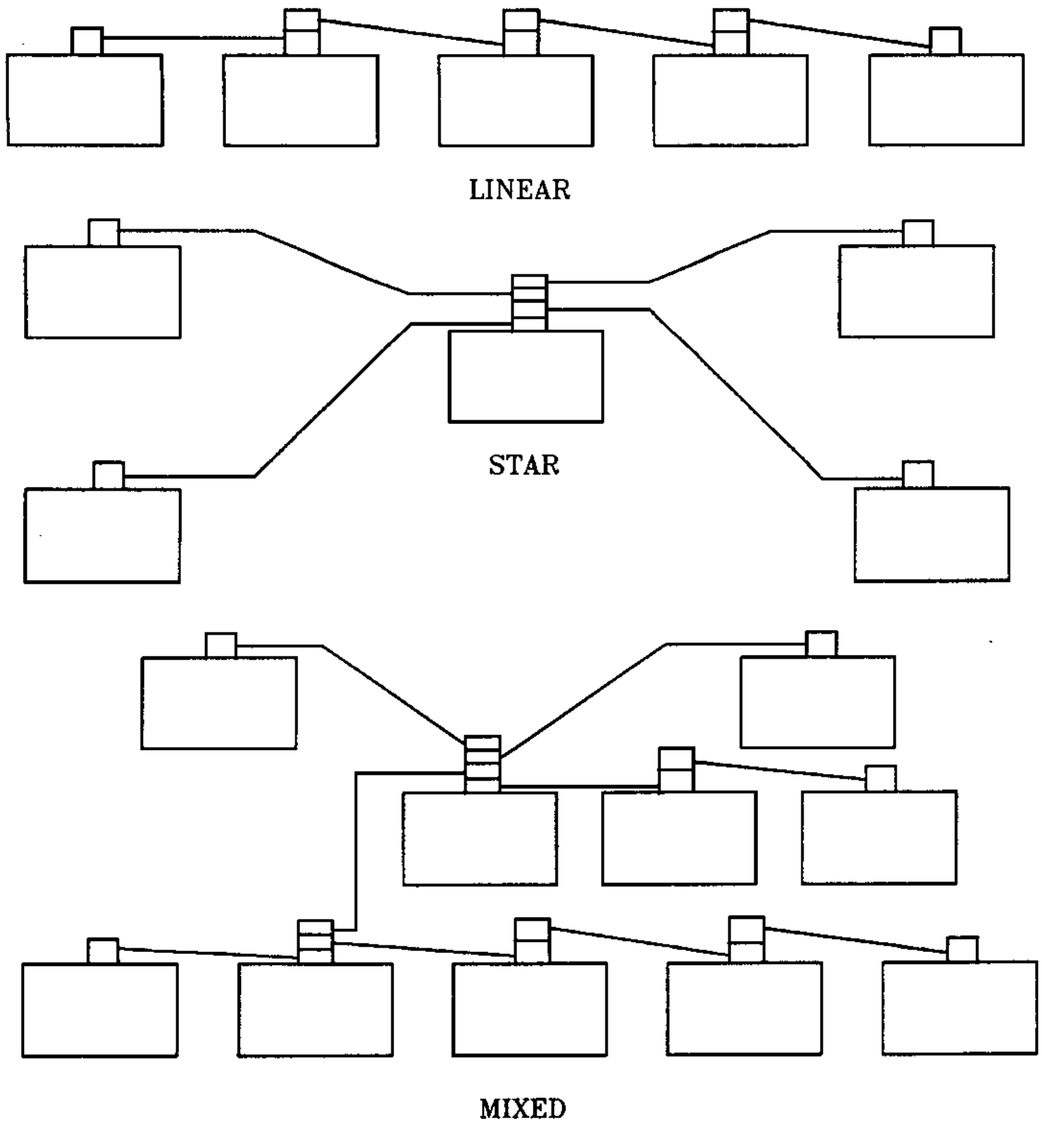


Figure 9-2: IEEE-488 Cabling Configurations

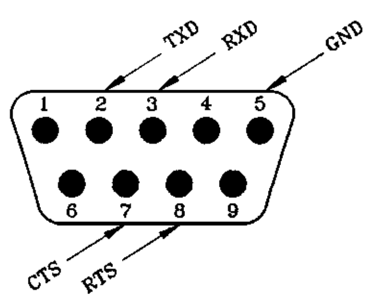
CHAPTER 10: SERIAL ASYNCHRONOUS COMMUNICATION

SECTION 10-1: PORT A, PORT B AND TERMINAL INTERFACE

PORT A and PORT B (P30 and P29) provide a RS-232 interface to accommodate communication between the Unidex 21 and peripheral devices such as a Host Computer, Printer, Slave Controller, etc. (Refer to the *Unidex 21 User's Manual*, Chapter 6, for Parameter settings affecting Port A and Port B communication.)

The TERMINAL Connector (P21) is also an RS-232 interface, however, it is dedicated solely to provide interface to a TeleVideo 905 Video Display Terminal. (Refer to the *Unidex 21 User's Manual* for Parameter settings affecting Terminal use.)

Details of the PORT A, PORT B and TERMINAL Connectors are shown in Figure 10-1. Electrical characteristics of the connectors are illustrated in Figure 10-2.



PIN	DESCRIPTION
1,4,6,9	(Not Used)
2	Transmit Data, Unidex 21 Output, transfers data to an RS-232 device
3	Receive Data, Unidex 21 Input, accepts input data from an RS-232 device
5	Ground
7	Clear to Send, Unidex 21 Input, goes high when RS-232 device is ready to receive data
8	Request to Send Data, Unidex 21 Output, sends a request signal to the RS-232 device to initiate sending

Figure 10-1: PORT A, PORT B and TERMINAL Connector (P30, P29, and P21)

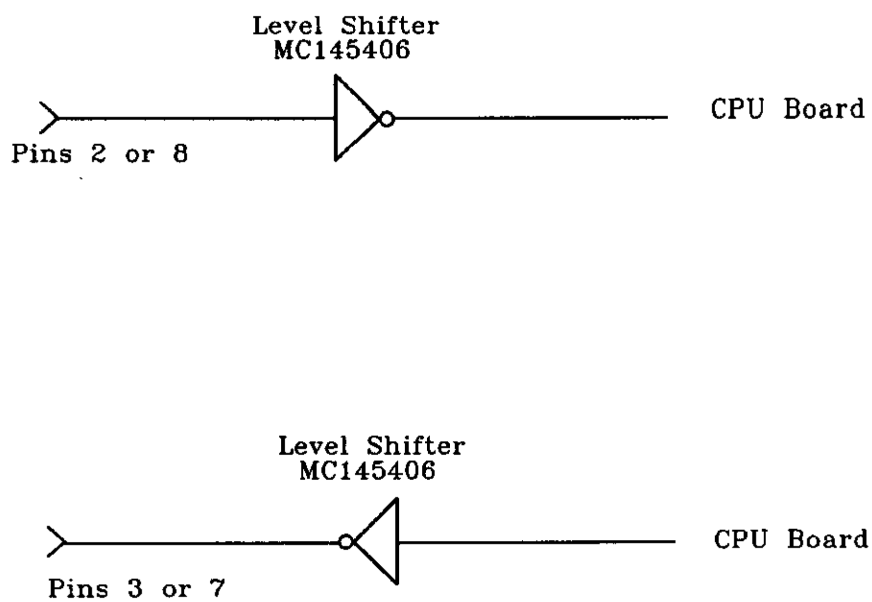
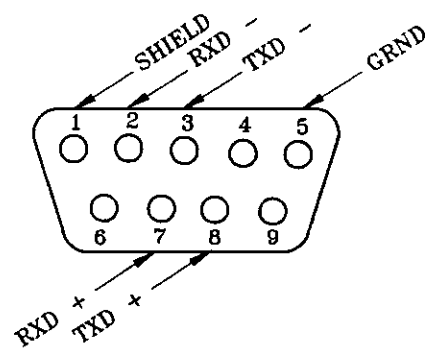


Figure 10-2: Electrical Characteristics of the PORT A, PORT B and TERMINAL Connectors (P30, P29 and P21)

SECTION 10-2: SERIAL COMMUNICATION INTERFACE

The SCI Connector (P32) provides for full Asynchronous Serial capability from the Unidex 21's DSP Boards to other DSP Boards, Microproccessors or periperals. These ports support standard bit rates (9600, 4800, 1200, and 300 bits/second), and protocols (1 start, 8 data, even/odd parity, 1 stop).

Details of the SCI Connector are shown in Figure 10-3. Electrical characteristics of the connector are illustrated in Figure 10-4.



PIN	DESCRIPTION
1	Shield
2	Receive Data (Inverse), accepts input data from a RS-232 device
3	Transmit Data (Inverse), transfers data to a RS-232 device
4,6,9	(Not Used)
5	Ground
7	Receive Data (Positive), accepts input data from a RS-232 device
8	Transmit Data (Positive), transfers data to a RS-232 device

Figure 10-3: SCI Connector (P32)

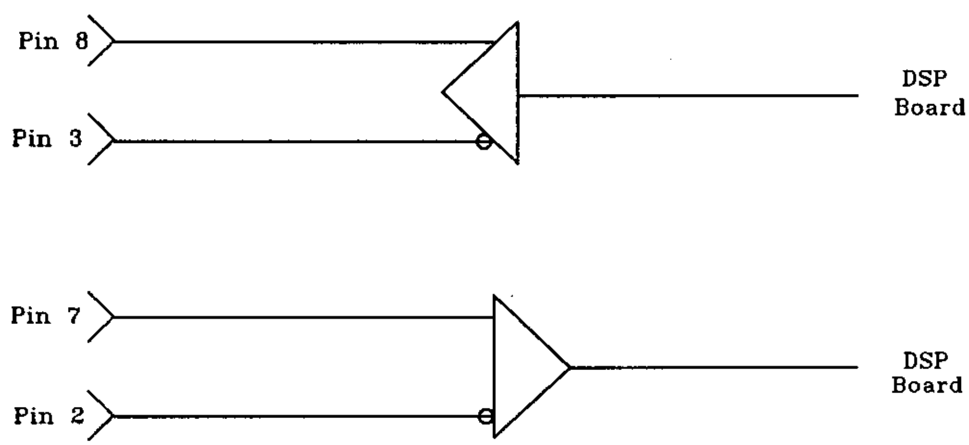


Figure 10-4: Electrical Characteristics of the SCI Connector (P32)

CHAPTER 11: SYNCHRONOUS SERIAL INTERFACE

The SSI Connector (P31) provides for communication with a variety of Synchronous Serial devices including industry-standard Codecs, Serial Peripherals (A/D, D/A) and Shift Registers. Data can be transmitted at a maximum of 5 million bits/second with word lengths of 8, 12, 16, or 24 bits.

NOTE: The SSI Connector does not currently support Customer interface.

Details of the SSI Connector are shown in Figure 11-1. Electrical characteristics are illustrated in Figure 11-2.

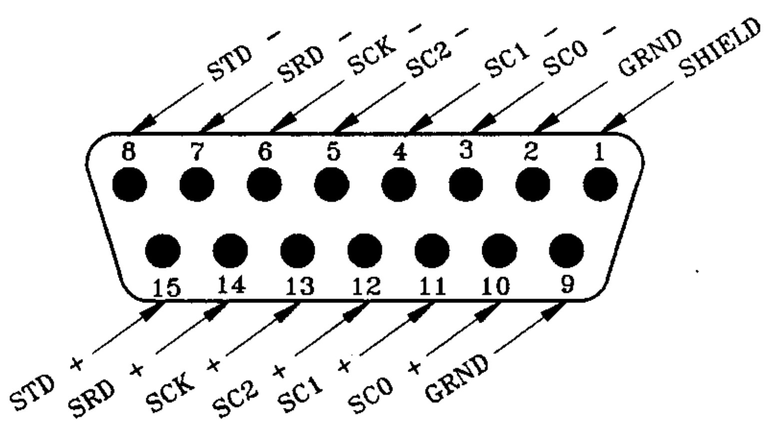


Figure 11-1: SSI Connector (P31)

PIN	DESCRIPTION
1	Shield
2	Ground
3	Serial Control Zero (Inverse), used for Serial Flag I/O such as is required in multiple device addressing
4	Serial Control One (Inverse), used for Serial Flag I/O such as is required in multiple device addressing
5	Serial Control Two (Inverse), generates the Frame Sync I/O Signal
6	Serial Clock (Inverse), clock output to the Receiver
7	Serial Receive Data (Inverse), receives Serial data and transfers data to Serial Shift Register
8	Serial Transmit Data (Inverse), transmits data from the Serial Transmit Shift Register
9	Ground
10	Serial Control Zero (Positive), used for Serial Flag I/O such as is required in multiple device addressing
11	Serial Control One (Positive), used for Serial Flag I/O such as is required in multiple device addressing
12	Serial Control Two (Positive), generates the Frame Sync I/O Signal
13	Serial Clock (Positive), clock output to the Receiver
14	Serial Receive Data (Positive), receives Serial data and transfers data to Serial Shift Register
15	Serial Transmit Data (Positive), transmits data from the Serial Transmit Shift Register
16 through 25	(Not Used)

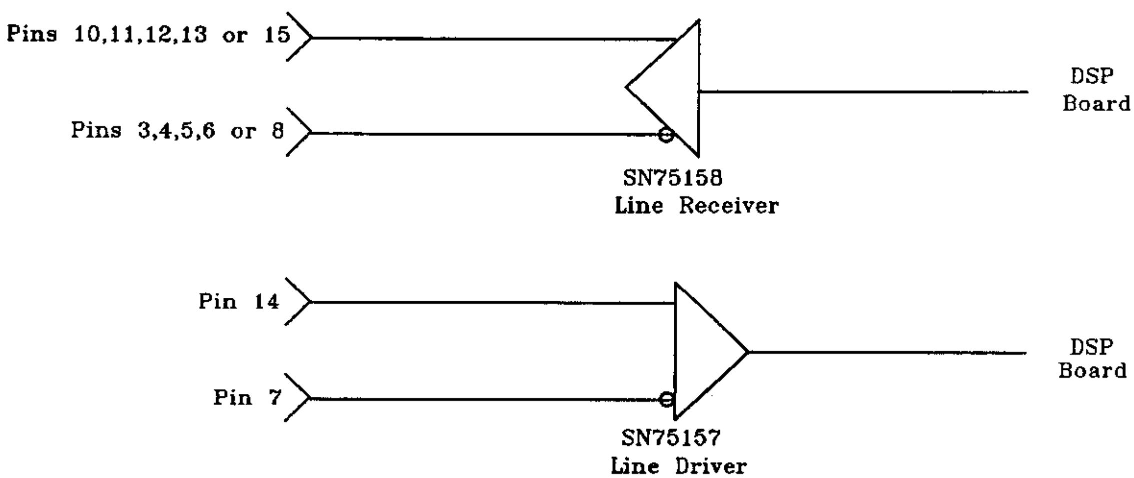


Figure 11-2: Electrical Characteristics of the SSI Connector (P31)

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 12: EGA MONITOR INTERFACE

An EGA Monitor may be interfaced to a Unidex 21 (equipped with an optional CRT Board) through the EGA OUT Connector (P16).

Details of the EGA OUT Connector are shown in Figure 12-1. Electrical characteristics are shown in Figure 12-2.

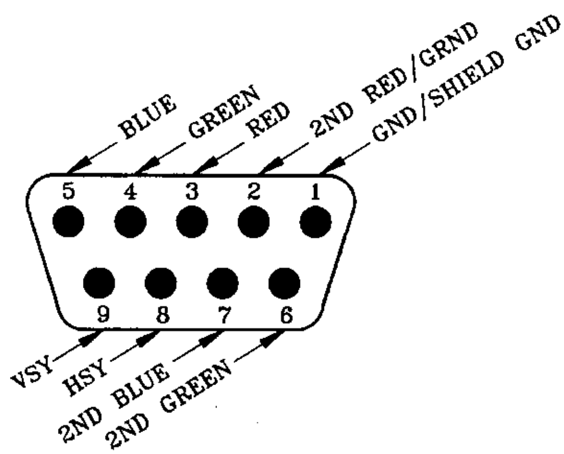


Figure 12-1: EGA OUTPT Connector (P16)

PIN	DESCRIPTION
1	Provides Shield Ground for EGA Monitors with only Red, Green, and Blue Signals or Signal Ground for EGA Monitors also having secondary Red, Green and Blue Signals (Selectable by Jumper JP2 of the Front Panel Board - See Chapter 3 of this Manual)
2	Provides Signal Ground for EGA Monitors with only Red, Green and Blue Signals or carries the Secondary Red line of an EGA Monitor equipped with secondary Red, Green and Blue Signals. (Selectable by Jumper JP3 of the Front Panel Board - See Chapter 3 of this Manual)
3	Primary Red Signal
4	Primary Green Signal
5	Primary Blue Signal
6	Secondary Green Signal
7	Secondary Blue Signal
8	Horizontal Signal Adjustment
9	Vertical Signal Adjustment

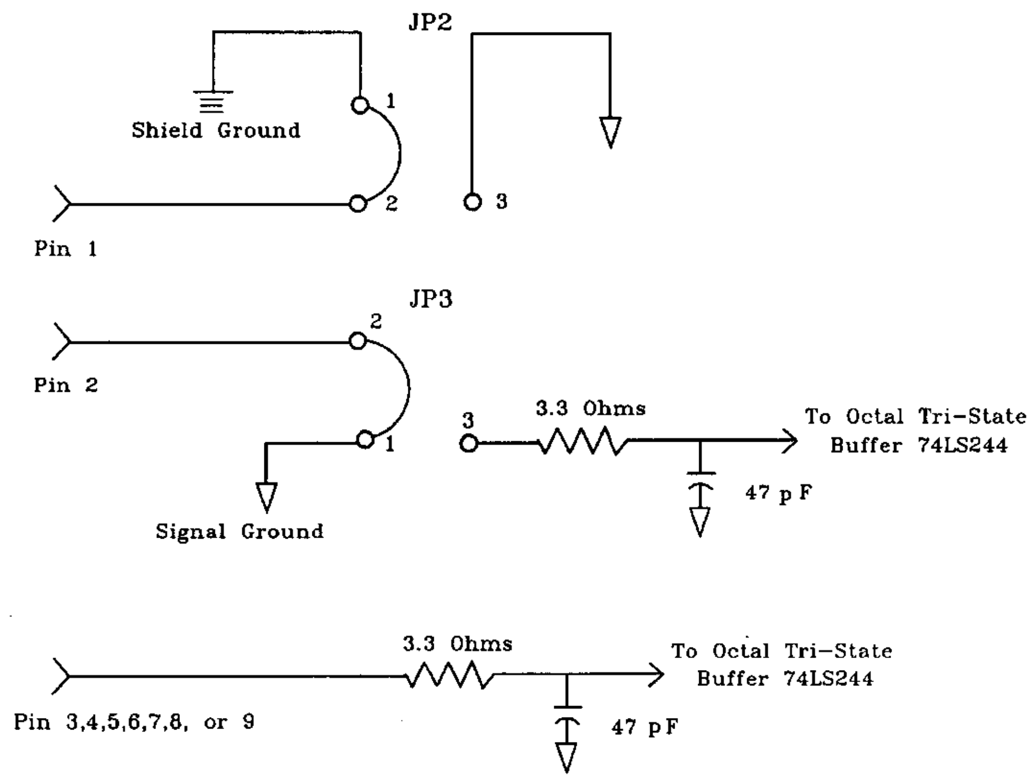


Figure 12-2: Electrical Characteristics of the EGA OUTPT Connector (P16)

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 13: I/O INTERFACE

SECTION 13-1: CPU I/O CONNECTOR(P22)

Details of the general purpose CPU I/O connector are shown in Figure 13-1. This connector provides up to 8 outputs and 16 inputs, all of which are opto-coupled. When used with the OPTO 22 I/O mounting rack assemblies (or equivalent) the Model OPC interconnection cable (standard 5 ft. length) provides a direct mating between the 50-pin Champ style connector and the edge connector on the OPTO board. Opto 22 Models PB8, PB16 and PB24 (or equivalent) may be utilized to provide either 8 outputs, 8 outputs/8 inputs, or 8 outputs/16 inputs with OPC interconnection. Other combinations are available with special cabling.

Electrical characteristics of the connector are illustrated in Figure 13-2. In order to provide optimum noise immunity, a seperate Power Supply must be used.

Table 13-1 lists the standard I/O interconnection between the Unidex 21 and the I/O mounting rack assemblies. Figure 13-3 illustrates OPTO 22 I/O mounting rack assembly wiring and jumper placement. These jumpers are essential so that an external +5V may be routed back to the Unidex 21's opto couplers.

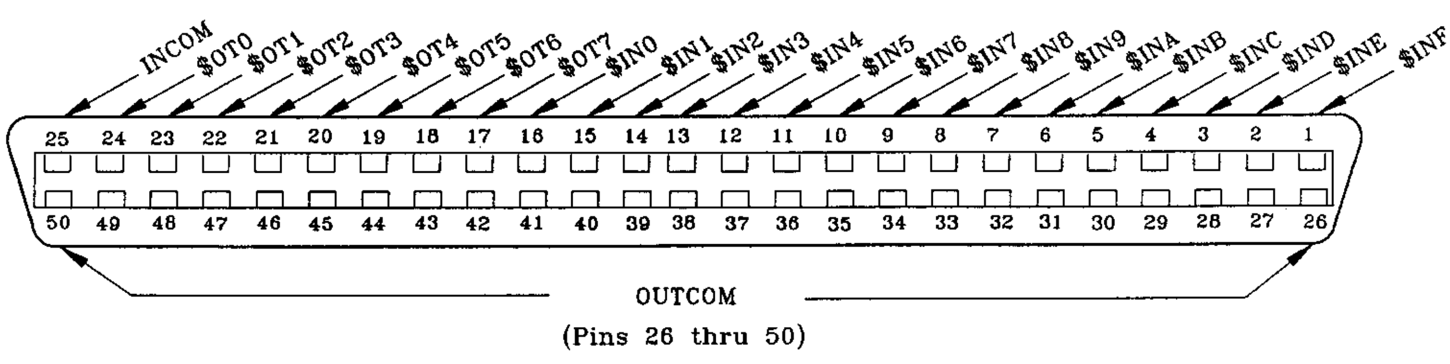


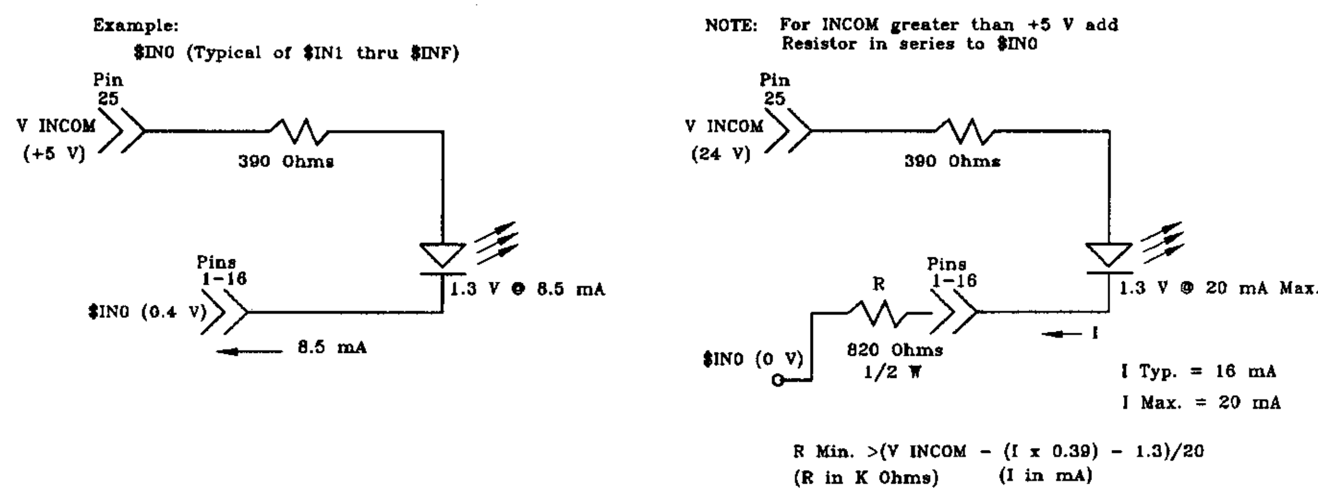
Figure 13-1: General Purpose CPU I/O Connector (P22)

PIN	DESCRIPTION
1	Data Input Bit 15, Address $INF
2	Data Input Bit 14, Address $INE
3	Data Input Bit 13, Address $IND
4	Data Input Bit 12, Address $INC
5	Data Input Bit 11, Address $INB
6	Data Input Bit 10, Address $INA
7	Data Input Bit 9, Address $IN9
8	Data Input Bit 8, Address $IN8
9	Data Input Bit 7, Address $IN7
10	Data Input Bit 6, Address $IN6
11	Data Input Bit 5, Address $IN5
12	Data Input Bit 4, Address $IN4
13	Data Input Bit 3, Address $IN3
14	Data Input Bit 2, Address $IN2
15	Data Input Bit 1, Address $IN1
16	Data Input Bit 0, Address $IN0
17	Data Output Bit 7, Address $OT7
18	Data Output Bit 6, Address $OT6
19	Data Output Bit 5, Address $OT5
20	Data Output Bit 4, Address $OT4
21	Data Output Bit 3, Address $OT3
22	Data Output Bit 2, Address $OT2
23	Data Output Bit 1, Address $OT1
24	Data Output Bit 0, Address $OT0
25	Input Common (5V Typical)
26 thru 50	Output Common (Grnd. Typical)

Sample Output Command: $OT0 = H,01 or $OT0 = H,00

Sample Input Command: (JUMP, ENT1, $IN0.EQ.H, 01)

$INO thru $INF INPUTS



$OTO thru $OT7

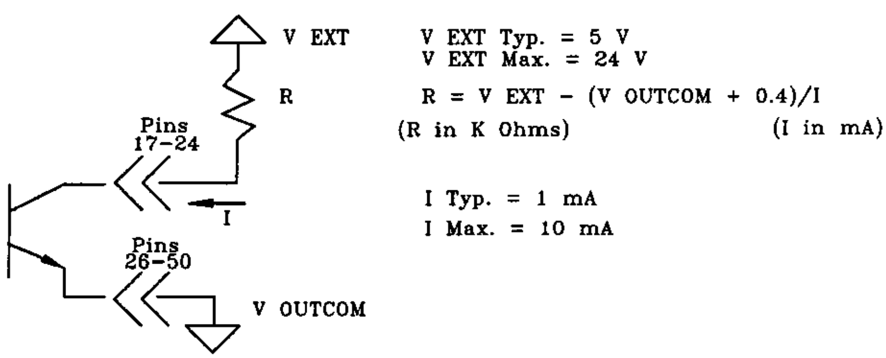


Figure 13-2: Electrical Characteristics of CPU I/O (P22)

INTERFACE CABLE ASSEMBLY (Model OPC) | PB8, PB16A, PB16C AND PB24 BOARD

OPTO-INTERFACE (P22)	CONTROL CONNECTION (EDGE CONNECTOR ON OPTO BOARD)		MODULE POSITION	CONNECTION DESCRIPTION	TYPE OF MODULE	FIELD CONNECTION (BARRIER STRIP)
25	49			+5V from external power supply		
24	47		0	$OT0	Output	1 and 2
23	45		1	$OT1	Output	3 and 4
22	43		2	$OT2	Output	5 and 6
21	41	PB8	3	$OT3	Output	7 and 8
20	39		4	$OT4	Output	9 and 10
19	37		5	$OT5	Output	11 and 12
18	35		6	$OT6	Output	13 and 14
17	33		7	$OT7	Output	15 and 16
16	31		8	$IN0	Input	17 and 18
15	29		9	$IN1	Input	19 and 20
14	27	PB16A	10	$IN2	Input	21 and 22
13	25	AND	11	$IN3	Input	23 and 24
12	23	PB16C	12	$IN4	Input	25 and 26
11	21		13	$IN5	Input	27 and 28
10	19		14	$IN6	Input	29 and 30
9	17		15	$IN7	Input	31 and 32
8	15		16	$IN8	Input	33 and 34
7	13		17	$IN9	Input	35 and 36
6	11		18	$INA	Input	37 and 38
5	9	PB24	19	$INB	Input	39 and 40
4	7		20	$INC	Input	41 and 42
3	5		21	$IND	Input	43 and 44
2	3		22	$INE	Input	45 and 46
1	1		23	$INF	Input	47 and 48

Pins 26 through 50 are signal common.

All even pins (2 to 50) are signal common.

WARNING: Type of module (input or output) cannot be interchanged. To do so may damage Unidex 21.

NOTE: +5 volt connection on Pin 25 (P22) or Pin 49 of edge connector is required when using PB8, PB16 and PB24.

This should be supplied from an external power supply and requires Opto rack jumper insertion as per Figure 13-3.

Typical Modules: IDC5, IDC5B, IAC5, IAC5A
ODC5, ODC5A, OAC5, OAC5A

Table 13-1: Interconnection Diagram of PB8, PB16A or C, or PB24 I/O Board

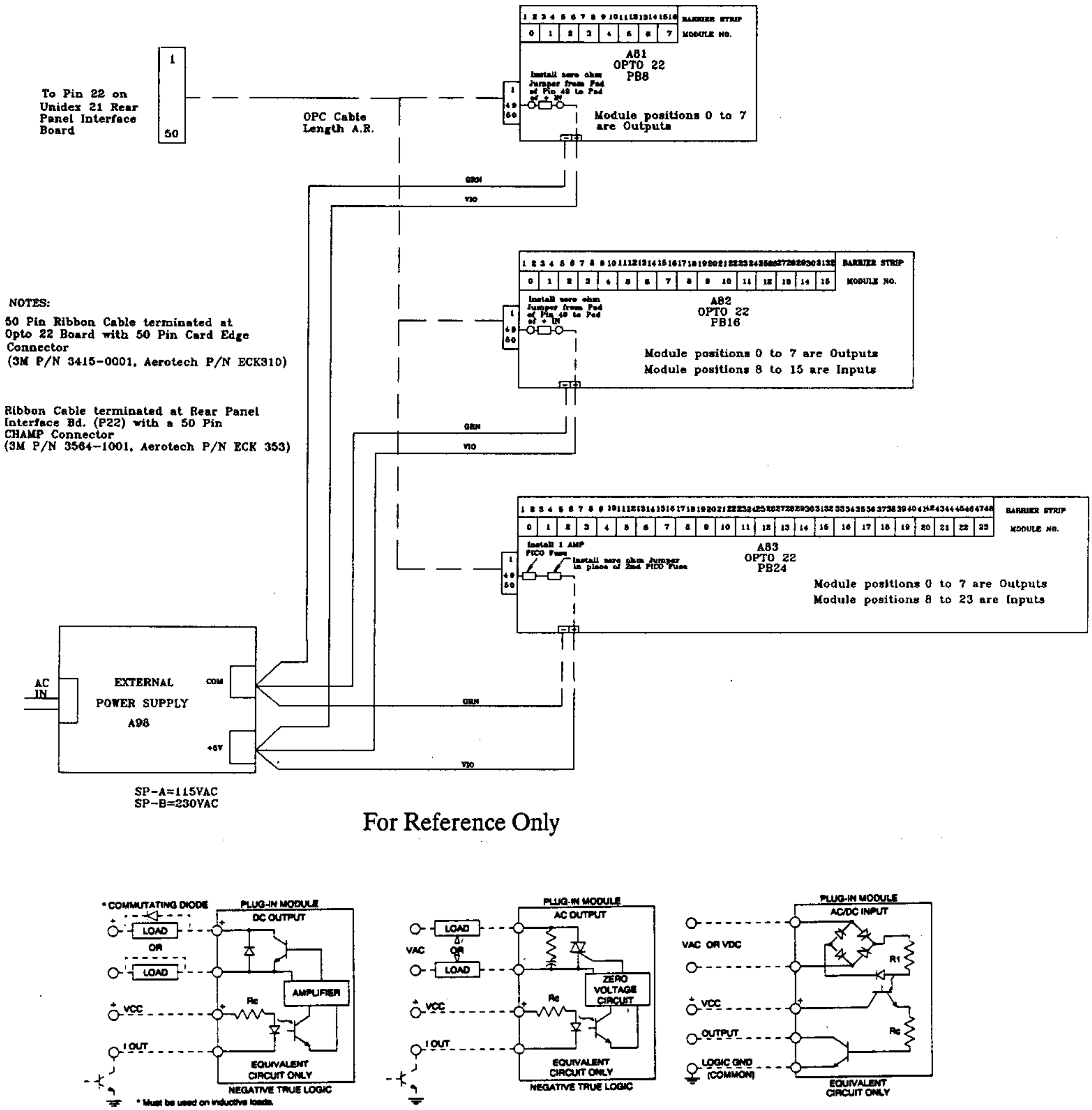


Figure 13-3: Unidex 21 CPU OPTO 22 I/O

SECTION 13-2: MISCELLANEOUS I/O

Details of the MISC I/O connector are shown in Figure 13-4. Electrical characteristics of the connector are shown in Figure 13-5.

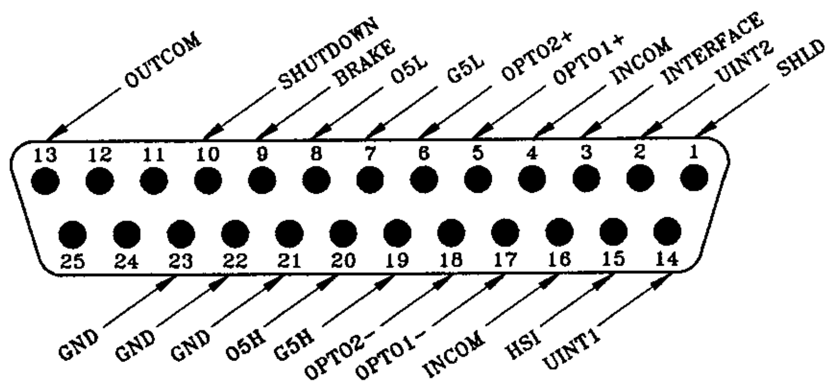


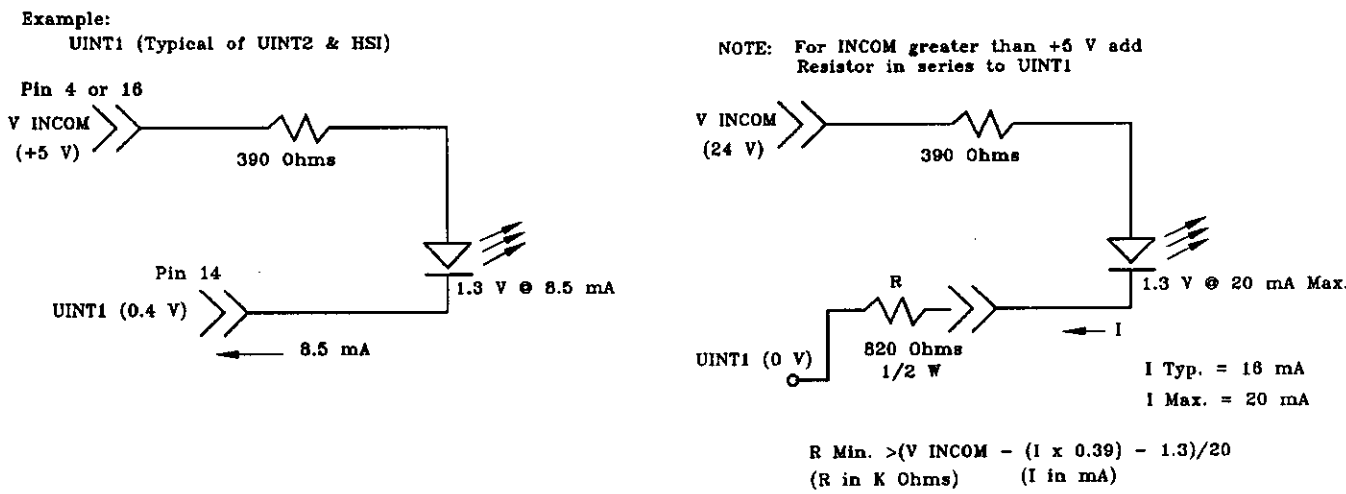
Figure 13-4: MISC I/O Connector (P23)

PIN	DESCRIPTION
1	Shield
2	User Interrupt 2 Input to CPU Board. Activates an established Interrupt routine.
3	Interface, used in conjunction with High Speed Interrupt Memory allocation.
4	Input Common, used in conjunction with High Speed Interrupt and User Interrupts 1 and 2*.
5	Fast Feedhold Positive Opto 1 Interrupt Input to DSP
6	Position Grabbing, Positive Opto 2 Interrupt Input to DSP
7	Not Determined
8	Not Determined
9	DSP Output activates Brake In Fault condition.
10	Monitors Brake Interlock, provides DSP Input to activate shutdown if Brake not connected.
11,12,24,25	(Not Used)
13	Output Common, used in conjunction with "Interface" (Pin3) * *
14	User Interrupt 2 Input to CPU Board. Activates an established Interrupt routine.
15	High Speed Interrupt for data collection
16	Input Common, used in conjunction with High Speed Interrupt and User Interrupts 1 and 2 *.
17	Fast Feedhold Negative Opto 1 Interrupt Input to DSP
18	Position Grabbing, Negative Opto 2 Interrupt Input to DSP
19	Not Determined
20	Not Determined
21,22,23	Ground

* IN COM is shared by all CPU Inputs and Interrupts

** OUT COMM is shared by all CPU Outputs and Interfaces

UINT1, UINT2, HSI INPUTS (PS2401A4 Opto Coupler)



INTERFACE OUTPUT (PS2401A Opto Coupler)

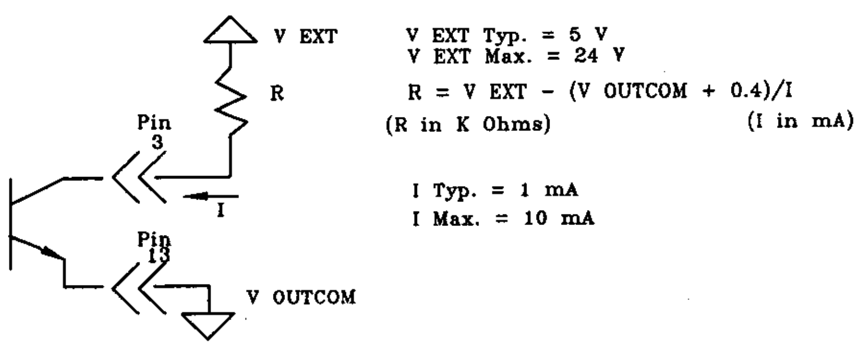


Figure 13-5: Electrical Characteristics of MISC I/O Connector (P35)

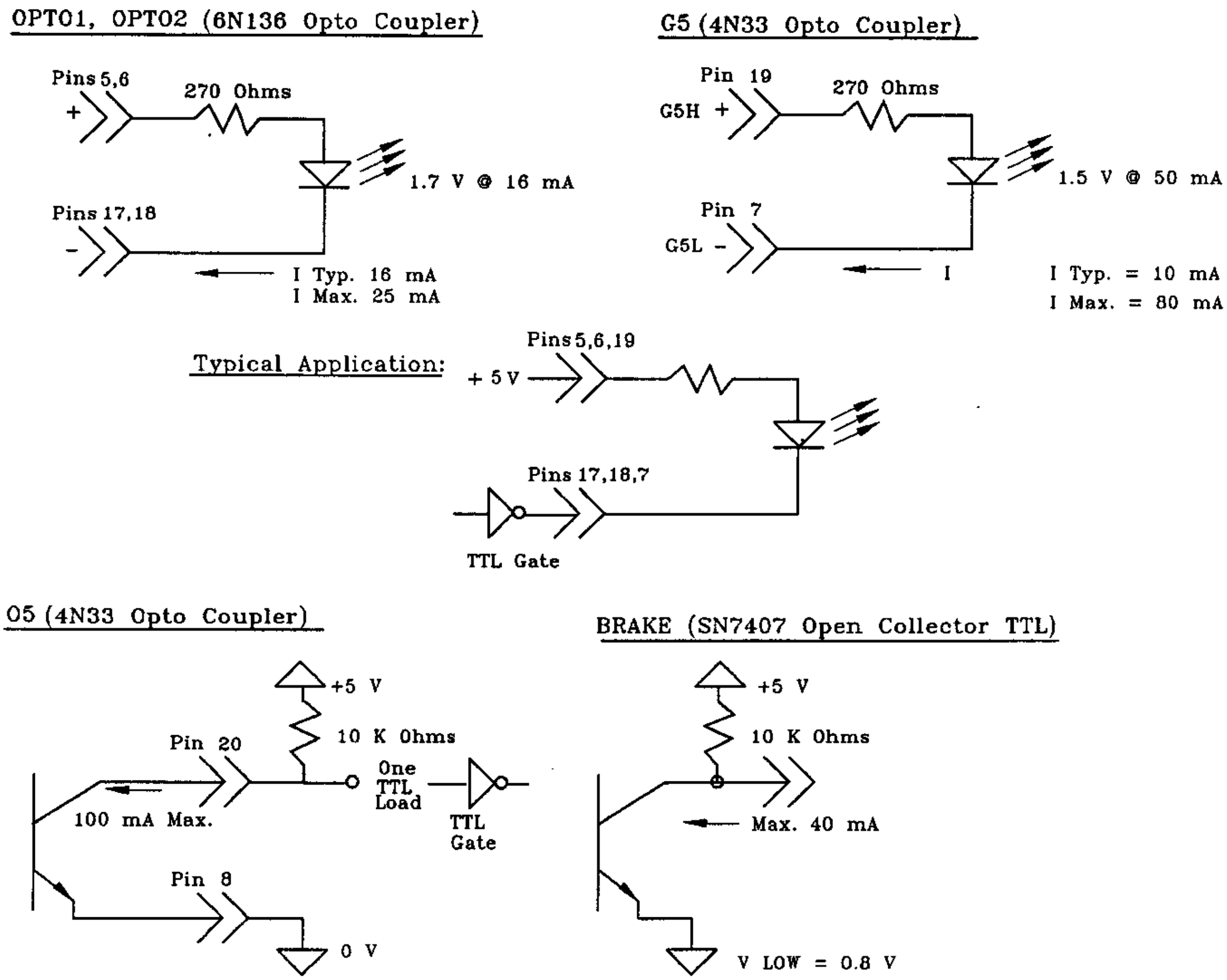
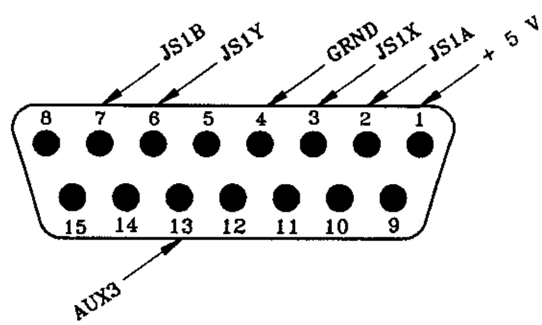


Figure 13-5Con't .: Electrical Characteristics of MISC I/O Connector (P35)

CHAPTER 14: JOYSTICK INTERFACE

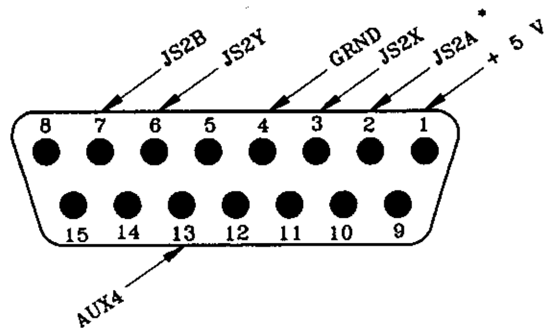
The Unidex 21 is configured to accommodate two optional JBV Joysticks.

Details of the Joystick 1 and Joystick 2 connectors (P34 and P25) are shown in Figure 14-1 and 14-2. Electrical characteristics of the connector are shown in Figure 14-3.



PIN	DESCRIPTION
1	+ 5V Input Common
2	Joystick Pushbutton "A" signal for axis grouping
3	Velocity control line for Axis 1 (as designated by Pushbutton "A")
4	Ground
5,8,9,10,11,12,14,15	(Not Used)
6	Velocity control line for Axis 2 (as designated by Pushbutton "A")
7	Joystick Pushbuton "B" signal for Joystic/Axis speed ratio
13	Receives signal upon Power Up from XY Potentiometers to provide Joystick "zero" position

Figure 14-1: JOYSTICK 1 Connector (P34)

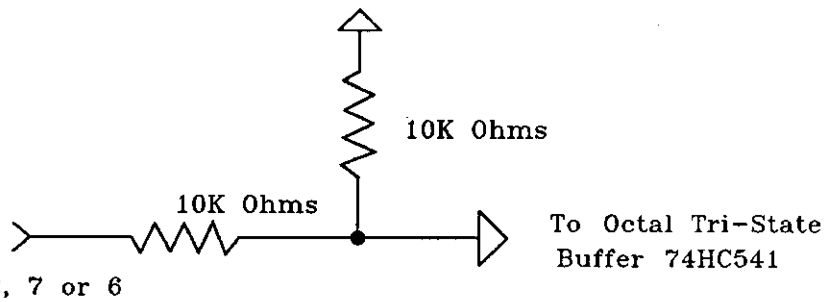


PIN	DESCRIPTION
1	+ 5V Input Common
2	Joystick Pushbutton "A" signal for axis grouping
3	Velocity control line for Axis 3 (as designated by Pushbutton "A")
4	Ground
5,8,9,10,11,12,14,15	(Not Used)
6	Velocity control line for Axis 4 (as designated by Pushbutton "A")
7	Joystick Pushbuton "B" signal for Joystic/Axis speed ratio
13	Receives signal upon Power Up from XY Potentiometers providing Joystick "zero" position

*** NOTE: When the Unidex 21 is used in the Teach Mode (Refer to the RECO command in the Unidex 21 Programming Manual) Pin 2 of the Joystick 2 Connector may be used to supply a hardware input signal from a triggering device to enable or disable position recording.**

Figure 14-2: JOYSTICK 2 Connector (P25)

Input Example:
JOYSTICK 1 Connector
(Typical of JOYSTICK 2 Connector)



Output Example:
JOYSTICK 1 Connector
(Typical of JOYSTICK 2 Connector)

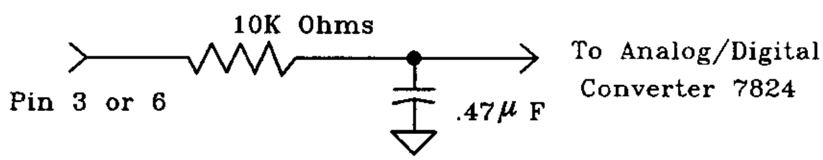


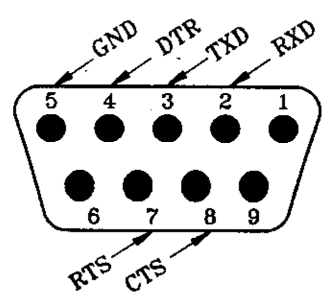
Figure 14-3: Electrical Characteristics of JOYSTICK 1 and 2 Connectors (P34 and P25)

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 15: TRACKBALL INTERFACE

A Trackball may be interfaced to the Unidex 21 to be used for manual positioning. The Trackball Connector (P33) is in RS-232 format.

Details of the Trackball Connector are shown in Figure 15-1. Electrical characteristics of the connector are shown in Figure 15-2.



PIN	DESCRIPTION
1,6,9	(Not Used)
2	Receive Data
3	Transmit Data
4	Data Terminal Ready
5	Ground
7	Request to Send
8	Clear to Send

Figure 15-1: TRACKBALL Connector (P33)

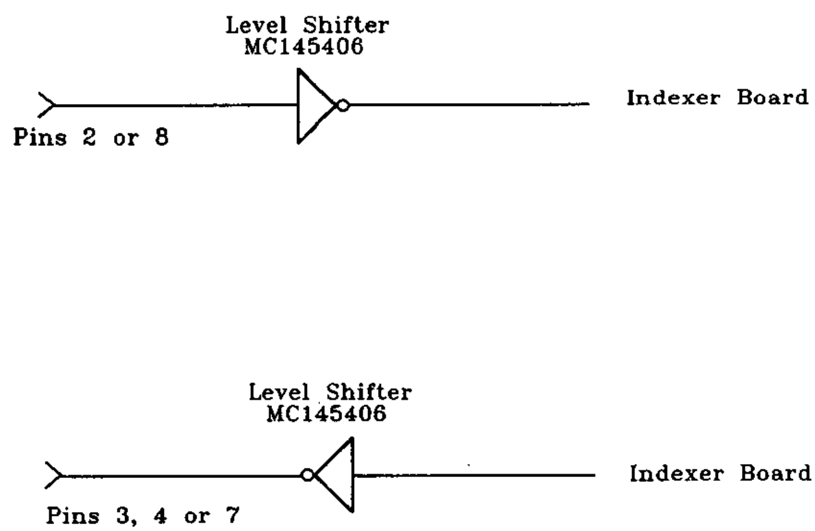
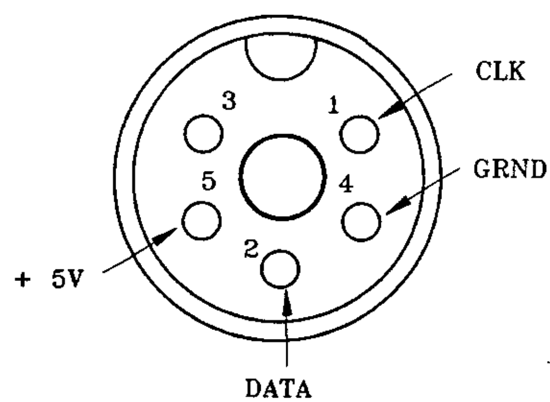


Figure 15-2: Electrical Characteristics of the TRACKBALL Connector (P33)

CHAPTER 16: KEYBOARD INTERFACE

The Keyboard Connector provides an interface between the Unidex 21 and an external Keyboard. The Keyboard must be a 101-key, "Qwerty" IBM PC/AT Keyboard operated by IBM Scan Code 2 (such as Model G81-1000 HBU/04 available from Cherry Electrical Products, 3600 Sunset Ave., Waukegan, Il. 60687).

Details of the KEY BD Connector (P27) are shown in Figure 16-1. Electrical characteristics of the connector are shown in Figure 16-2.



PIN	DESCRIPTION
1	Clock
2	Data Output
3	(Not Used)
4	Ground
5	+5V Input Common

Figure 16-1: KEY BD Connector (P27)

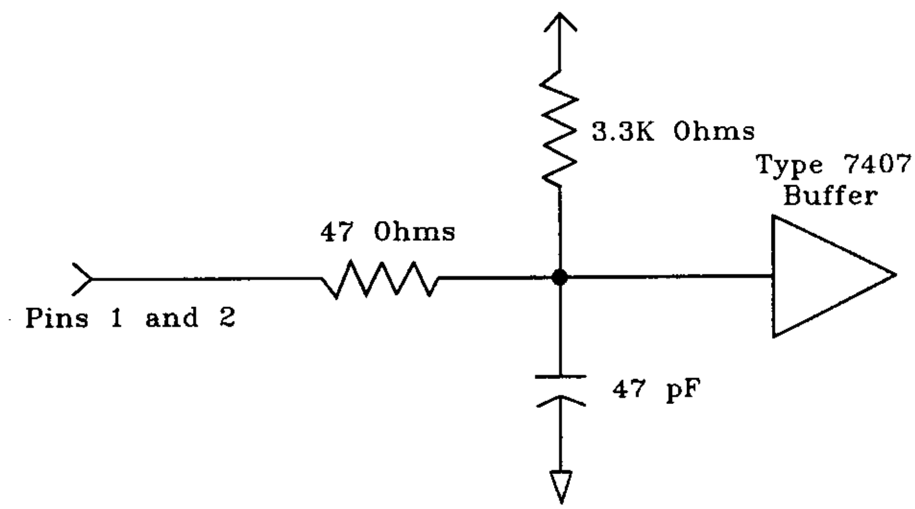


Figure 16-2: Electrical Characteristics of the KEY BD Connector (P27)

CHAPTER 17: TROUBLESHOOTING

This Chapter is provided to be used as an aid in the diagnosis and solution of problems due to improper hardware configurations.

SYMPTOM	POSSIBLE CAUSE
Power Up - Unidex 21 does not respond. (Cooling Fans do NOT Come On.)	Power Source Not Active
	Power input cable damaged or improperly connected (See Chapter 4 of this manual)
	Power Input Fuse Blown
Power Up - Fans Come On - No Display	Internal AC Power Cord to Power Supply damaged or incorrectly connected
	Power Supply's ON/OFF Switch in the OFF position
	Power Supply Fuse Blown
	Front Panel and/or CRT Board not included in system configuration
Power Up - Fans Come On - No Display	Front Panel and the CRT Board are not properly seated

SYMPTOM	POSSIBLE CAUSE
Power Up - Fans Come On - No Display - Con't	Cables to the Front Panel and CRT Boards are loose or improperly connected
Unidex 21 active but does not respond to Keyboard entry	System Key Switch is OFF
	Error condition exists, locking up system
	Keyboard Connection Cable is damaged or incorrectly connected
	Keyboard not functional
Unidex 21 does not respond appropriately to keyboard entries	Parameter Settings do not correspond to system configuration. (See Unidex 21's User Manual, Chapter 6.)
	A circuit board is not properly seated.
	Poor Ground and/or Shield connection.
Unidex 21 does not respond appropriately to keyboard entries	Parameter Settings do not correspond to system configuration. (See Unidex 21's User Manual, Chapter 6.)
	A circuit board is not properly seated.
	Poor Ground and/or Shield connection.

SYMPTOM	POSSIBLE CAUSE
Unidex 21 does not respond appropriately to keyboard entries - Con't	Motor and/or Encoder cables not properly routed or connected.
Unidex 21 responds to Keyboard entry but improper communication takes place between Unidex 21 and Transducer.	Blown Rear Panel Motor Fuse
	Motor and/or Transducer cables not properly routed or connected.
	Power Supply's 5V output not functioning (Any Power Suppy LED Off)
Axis Movement Unstable	System needs tuned. (See Auto-Tune procedure as described in the Unidex 21 User's Manual, Chapter 6)
Axis Movement Unstable	Motor and/or Transducer cables not properly routed or connected.
	Amplifier and/or Motor sized improperly
	Line Noise from improper Ground or Shield connections.
Motor does not move when requested	Parameter Settings do not correspond to system configuration (See Unidex 21's User Manual, Chapter 6.)
	Wrong Axis designation
	Motor and/or Transducer cables not properly routed or connected.

SYMPTOM	POSSIBLE CAUSE
Motor does not move when requested- Con't	Blown Rear Panel Motor Fuse
	System needs tuned. (See Auto-Tune procedure as described in the Unidex 21 User's Manual, Chapter 6)
	Faulty Motor
Motor does not move when requested	Line Noise from improper Ground or Shield connections.
Motor does not stop when requested	Wrong Axis designation
	Incorrect Amplifier Offset
	Line Noise from improper Ground or Shield connections.
Motor moves in the opposite direction from what is requested	Reversed Motor Polarity (DC Servo Brush Type only) (See Chapter 5 of this Manual)
Eratic Motor movement	Inproper Motor Phasing (AC Servo Brushless Type only) (See Chapter 5 of this Manual)
Motor Speed does not correspond to requested speed	Amplifier and/or Motor sized improperly
	Parameter Settings do not correspond to system configuration (See Unidex 21's User Manual, Chapter 6.)

SYMPTOM	POSSIBLE CAUSE
System does not respond to Limit Switch	Limit Switch wiring faulty
System does not respond to Limit Switch	Parameter Settings do not correspond to system configuration (See Unidex 21's User Manual, Chapter 6.)
	Line Noise from improper Ground or Shield connections.

UNIDEX® 21
MOTION CONTROLLER
USER'S MANUAL

PN: EDU 116

REFERENCE
DOCUMENT

AEROTECH

AEROTECH, INC., 101 Zeta Drive, Pittsburgh, PA 15238
(412) 963-7470 • TWX 710-795-3125 • FAX(412)963-7459

Aerotech, Inc.
Release No. 4.0
February, 1993

DISCLAIMER:
The information contained in this manual is subject to change due to improvements in design. Though this document has been checked for inaccuracies, Aerotech does not assume responsibility for any errors contained herein.

TRADEMARKS:
Unidex is a registered trademark of Aerotech, Inc.

IBM PC/XT/AT are registered trademarks of the International Business Machines Corporation.

PAMUX is a registered trademark of the Opto 22 Corporation.

Manual Update (3.0 to 4.0)

Aerotech's Unidex 21 User's Manual has been updated as of **February 10, 1993** to include the following changes:

Reorganization of Chapter 6 "The Parameter Mode" which includes a break up of the following sections, as well as each change for that section:

INTRODUCTION

GETTING STARTED

GENERAL PARAMETERS

General Parameter #3 – Added additional Segment Calculation Rates
"4" for a 8ms rate
"5" for a 16ms rate

General Parameter #11 – Added reference to the maximum setting for the parameter (1999 bytes).

General Parameter #30 – Deleted sentence re: The appropriateness of the Axes Ramp time setting may be verified by application of certain parameters derived from application of the Auto-tune function.

General Parameter #45 – Added note re: Perpendicularity error compensation will not be enabled until the axes are sent home.

General Parameter #52 – Added sentence re: Although there is no operational difference between the master and slave controller, typically the master is set to Yes, and the Slave is set to No.

General Parameter #57 – "NEW" - Master Trajectory Linear?

General Parameter #58 – "NEW" - Fast FeedHold Ramp Time (ms)

General Parameter #59 – "NEW" - Master Parabolic Coefficient

General Parameter #60 – "NEW" - Exp Filter Level (0-7)

General Parameter #61 – Ramping During G23 Operation?

AXIS PARAMETERS

Axis Parameter #21 – Added note re: Setting this parameter to zero for all axes will disable this feature.

Axis Parameter #23 – Added sentence re: The minimum setting for the parameter is 1000.

Axis Parameter #33 – Added sentence re: A value of zero will disable max velocity error checking.

AXIS PARAMETERS (CON'T)

Axis Parameter #34	– Added sentence re: A value of zero will disable max velocity error checking.
Axis Parameter #35	– Added sentence re: A value of zero will disable max velocity error checking.
Axis Parameter #38	– Added sentence re: A coefficient of zero represents a linear ramp. Higher values result in a steeper curve.
Axis Parameter #45	– Added note re: Backlash compensation will not be enabled unless the axis is sent home.

AXES AUTO-TUNE

FRONT PANEL FUNCTION KEYS

LOAD/SAVE PARAMETER

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

SECTION 1-1: OVERVIEW OF THE UNIDEX 21

The Unidex 21 Motion Controller is a multi-microprocessor based motion controller capable of producing linear, circular, helical, spherical and elliptical interpolation as well as velocity profiling and cubic spline contouring.

Some of the available package configurations are shown below:

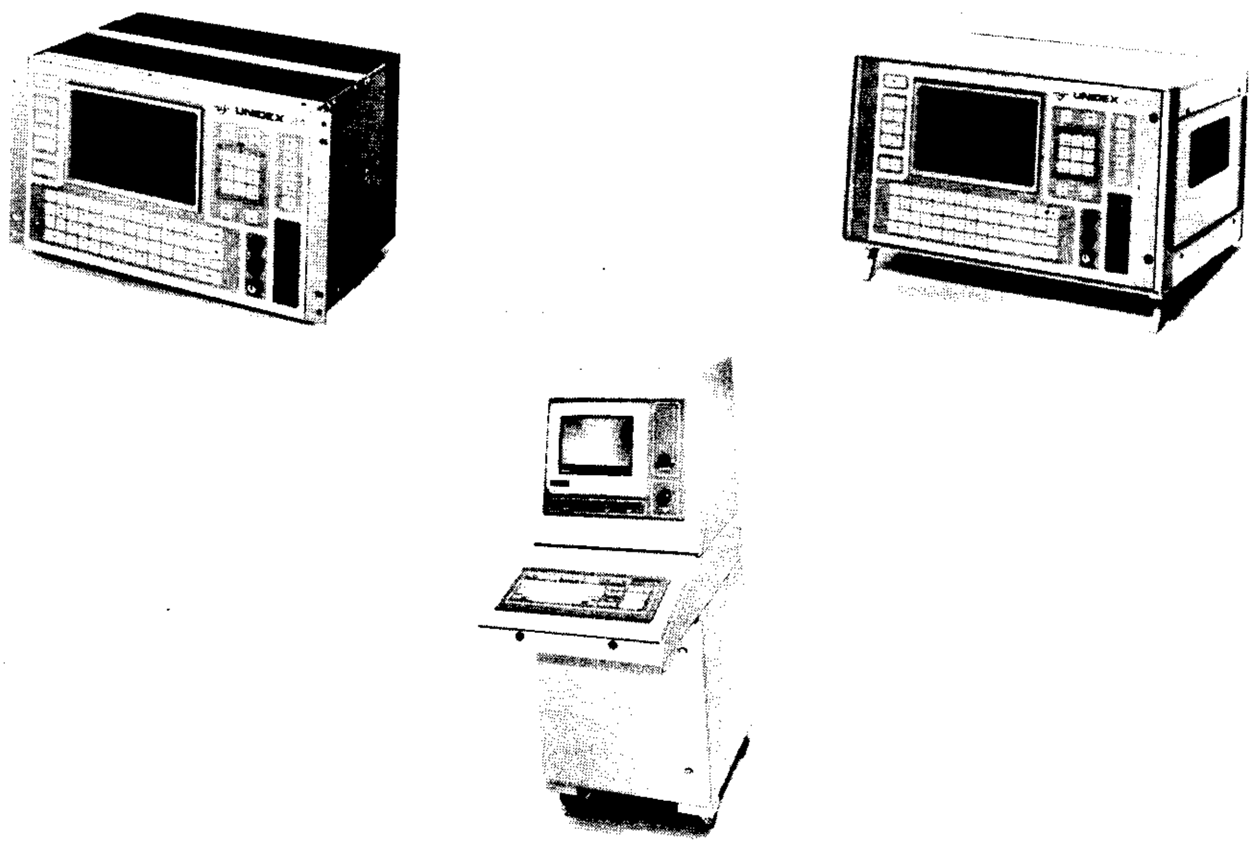


Figure 1-1: The: Unidex 21 Motion Controller Family

SECTION 1-2: UNIDEX 21 COMPONENTS

The basic components of Unidex 21 are illustrated below:

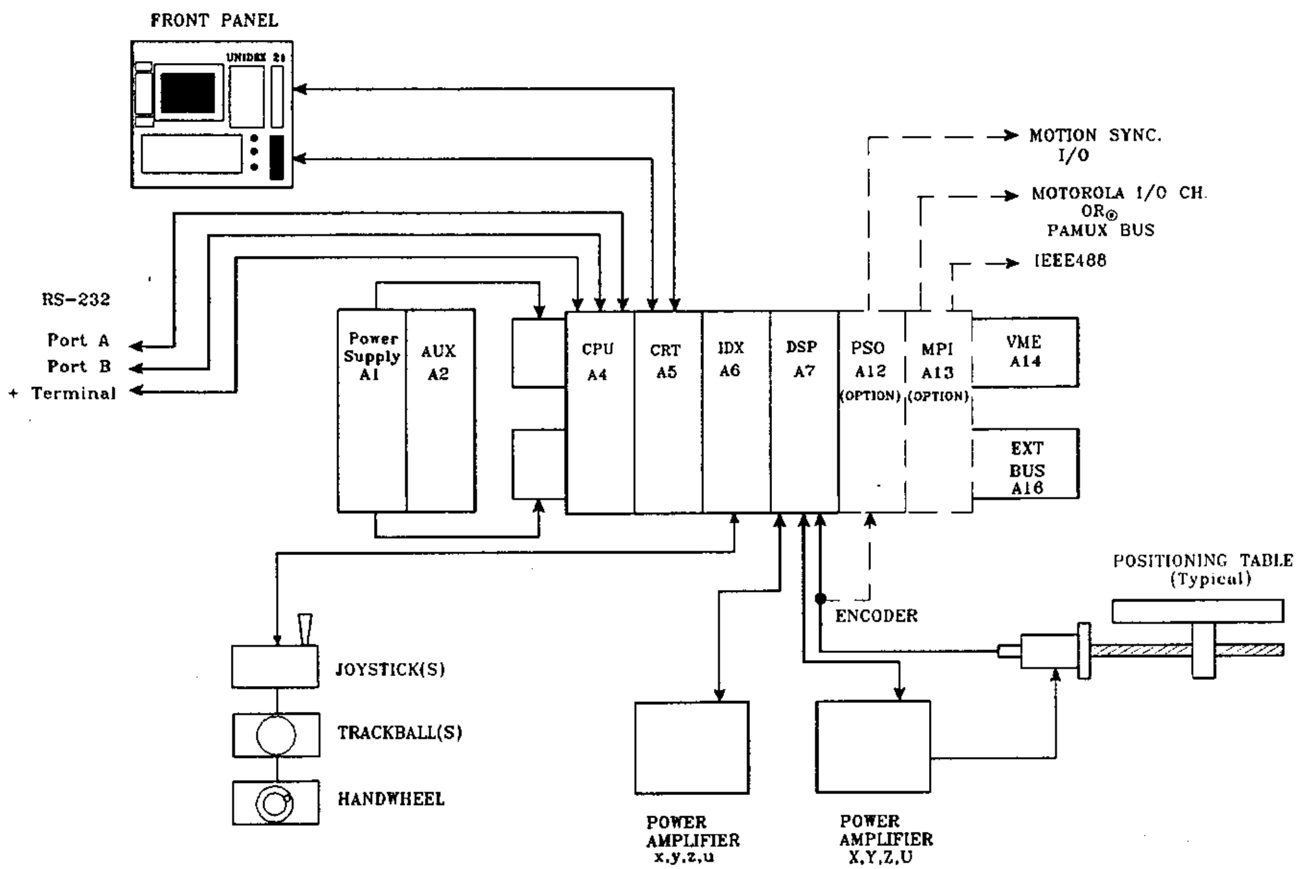


Figure 1-2: Unidex 21 Component Block Diagram

CHAPTER 2: GETTING ACQUAINTED WITH UNIDEX 21

This Chapter provides the User with general Unidex 21 operating information. (See Section 2-5 for Model variations.)

SECTION 2-1: INTERFACE CONNECTIONS

Prior to Unidex 21 operation, make certain all applicable interface connections have been correctly made. The following is a list of the connectors available on the Unidex 21 Rear Interface Panel (refer to Figure 2-1 for connector locations):

NOTE: **Throughout this manual, Axes 1, 2, 3, 4, 5, 6, 7, and 8 will be referred to Axes X, Y, Z, U, x, y, z, and u respectively.**

Reference	Name	Description
P6/P6A*	Y LINEAR ENC	Y Axis Linear Encoder (15 pin, Male "D" type)
P6B *	y LINEAR ENC	y Axis Linear Encoder (15 pin, Male "D" type)
P7/P7A*	Y ENC/LMTS	Y Axis Encoder Limit Switch, Hall Effect (25 pin, Female "D" type)
P7B *	y ENC/LMTS	y Axis Encoder Limit Switch, Hall Effect (25 pin, Female "D" type)
P8/P8A*	U LINEAR ENC	U Axis Linear Encoder (15 pin, Male "D" type)
P8B *	u LINEAR ENC	u Axis Linear Encoder (15 pin, Male "D" type)
P9/P9A*	U ENC/LMTS	U Axis Encoder Limit Switch,Hall Effect (25 pin, Female "D" type)
P9B *	u ENC/LMTS	u Axis Encoder Limit Switch,Hall Effect (25 pin, Female "D" type)
P10	PAMUX or I/O CHANNEL	(50 pin, Champ)
P11	IEEE-488	(24 pin, Champ)
P12/P12A*	X LINEAR ENC	X Axis Linear Encoder (15 pin, Male "D" type)

Reference	Name	Description
P12B *	x LINEAR ENC	X Axis Linear Encoder (15 pin, Male "D" type)
P13/P13A*	X ENC/LMTS	X Axis Encoder Limit Switch, Hall effect (25 pin, Female "D" type)
P13B *	x ENC/LMTS	x Axis Encoder Limit Switch, Hall effect (25 pin, Female "D" type)
P14/P14A*	Z LINEAR ENC	Z Axis Linear Encoder (15 pin, Male "D" type)
P14B *	z LINEAR ENC	z Axis Linear Encoder (15 pin, Male "D" type)
P15/P15A*	Z ENC/LMTS	Z Axis Encoder Limit Switch (25 pin, Female "D" type)
P15B*	z ENC/LMTS	z Axis Encoder Limit Switch (25 pin, Female "D" type)
P16	EGA OUTPUT	EGA Monitor (9 pin, Female "D" type)
P21	TERMINAL	TeleVideo 905 Video Display Terminal (9 pin, Male "D" type)
P22	CPU OPTO 22 PB24	Eight OPTO Isolated Outputs, Sixteen Opto Isolated Inputs, Interfaces directly to CPU, PB24 (50 pin, Champ)
P23	MISC I/O	(25 pin, Female "D" type)
P23A *	MISC I/O1	(25 pin, Female "D" type)
P23B *	MISC I/O2	(25 pin, Female "D" type)
P24	IDX PORT B	Indexing Board , RS-232 (9 pin, Male "D" type)
P25	JOY 2	Joystick # 2 Input (15 pin, Female "D" type)
P26	I/O PWR	Power Output to Motorola I/O Channel Card (14 pin, Champ)
P27	KEY BD	AT-Style Keyboard Interface (5 pin, Female DIN , Type Mab55V)
P29	PORT B	RS-232 Port B (9 pin, Male "D" type)

Reference	Name	Description
P30	PORT A	RS-232 Port A (9 pin, Male "D" type)
P31	SSI	Serial Synchronous (DSP) Interface (15 pin, Female "D" type)
P31A *	SSI1	Serial Synchronous (DSP) Interface (15 pin, Female "D" type)
P31B *	SSI2	Serial Synchronous (DSP) Interface (15 pin, Female "D" type)
P32	SCI	Serial Communication (DSP) Interface (9 pin, Male "D" type)
P32A *	SCI1	Serial Communication (DSP) Interface (9 pin, Male "D" type)
P32B *	SCI2	Serial Communication (DSP) Interface (9 pin, Male "D" type)
P33	TRACKBALL	Trackball Input Interface (9 pin, Male "D" type)
P34	JOY 1	Joystick # 1 Input (15 pin, Female "D" type)
P35	IDX MST BUS	Misc., Spindle Tool Changer Interface to Indexing Board (50 pin, Champ)
P40	IDX INP1	Handwheel /Auto-Focus Interface (9 pin, Female "D" type)
P41	IDX INP2	Handwheel /Auto-Focus Interface (9 pin, Female "D" type)
P42/P42A *	U RESOLVER	U Axis Resolver (9 pin, Female "D" type)
P42B *	u RESOLVER	u Axis Resolver (9 pin, Female "D" type)
P43/P43A *	Z RESOLVER	Z Axis Resolver (9 pin, Female "D" type)
P43B *	z RESOLVER	z Axis Resolver (9 pin, Female "D" type)
P44/P44A *	Y RESOLVER	Y Axis Resolver (9 pin, Female "D" type)
P44B *	y RESOLVER	y Axis Resolver (9 pin, Female "D" type)

Reference	Name	Description
P45/P45A *	X RESOLVER	X Axis Resolver (9 pin, Female "D" type)
P45B *	x RESOLVER	x Axis Resolver (9 pin, Female "D" type)
P48A *	DRIVE INT 1	Current Command and Shutdown Interface to Power Rack (9 pin, Female "D" type)
P48B *	DRIVE INT 2	Current Command and Shutdown Interface to Power Rack (9 pin, Female "D" type)
P51	(Optional)	(50 pin, Champ)
P52	(Optional)	(50 pin, Champ)

"*" denotes connectors available on Model U21C only.

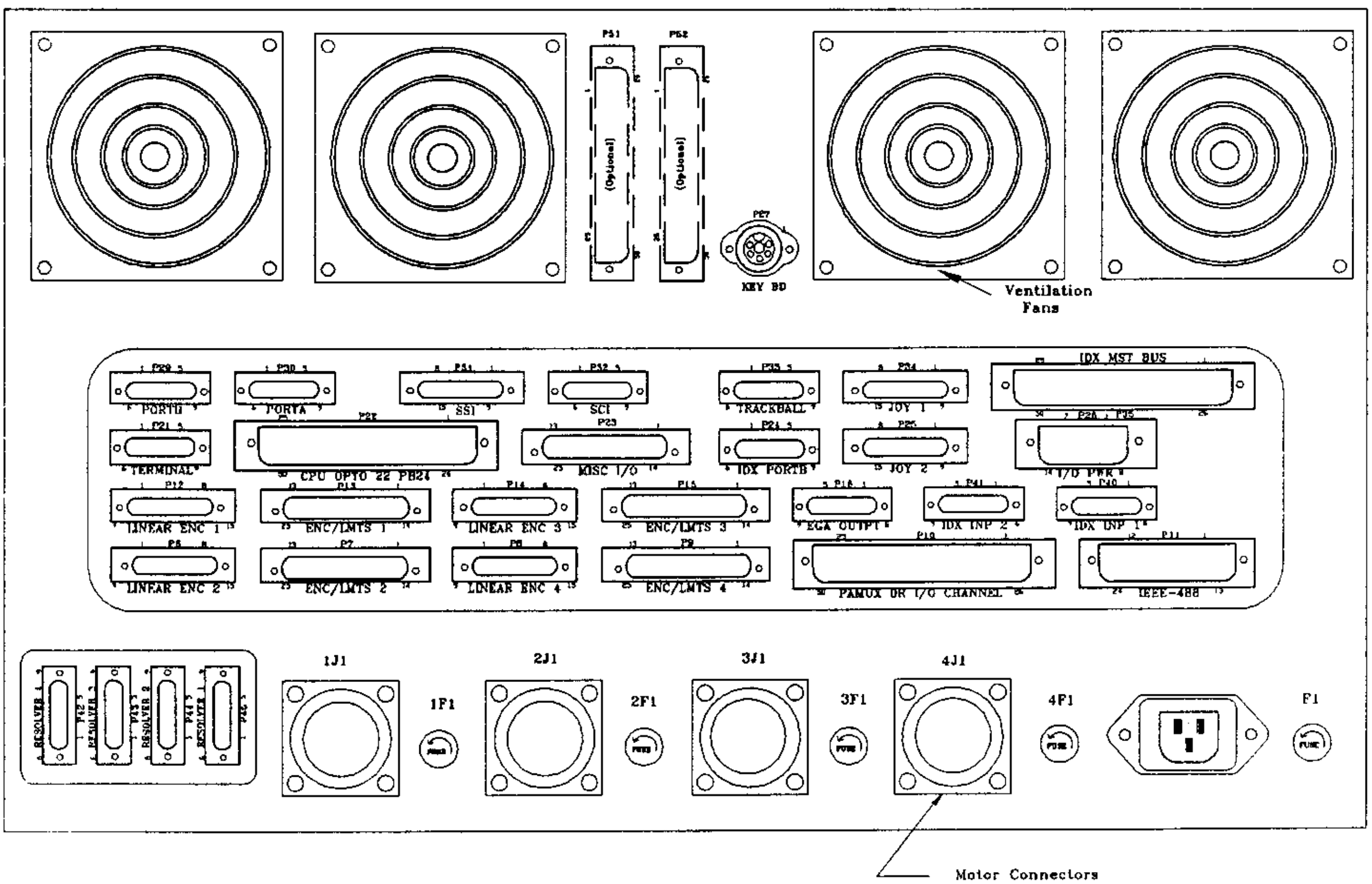
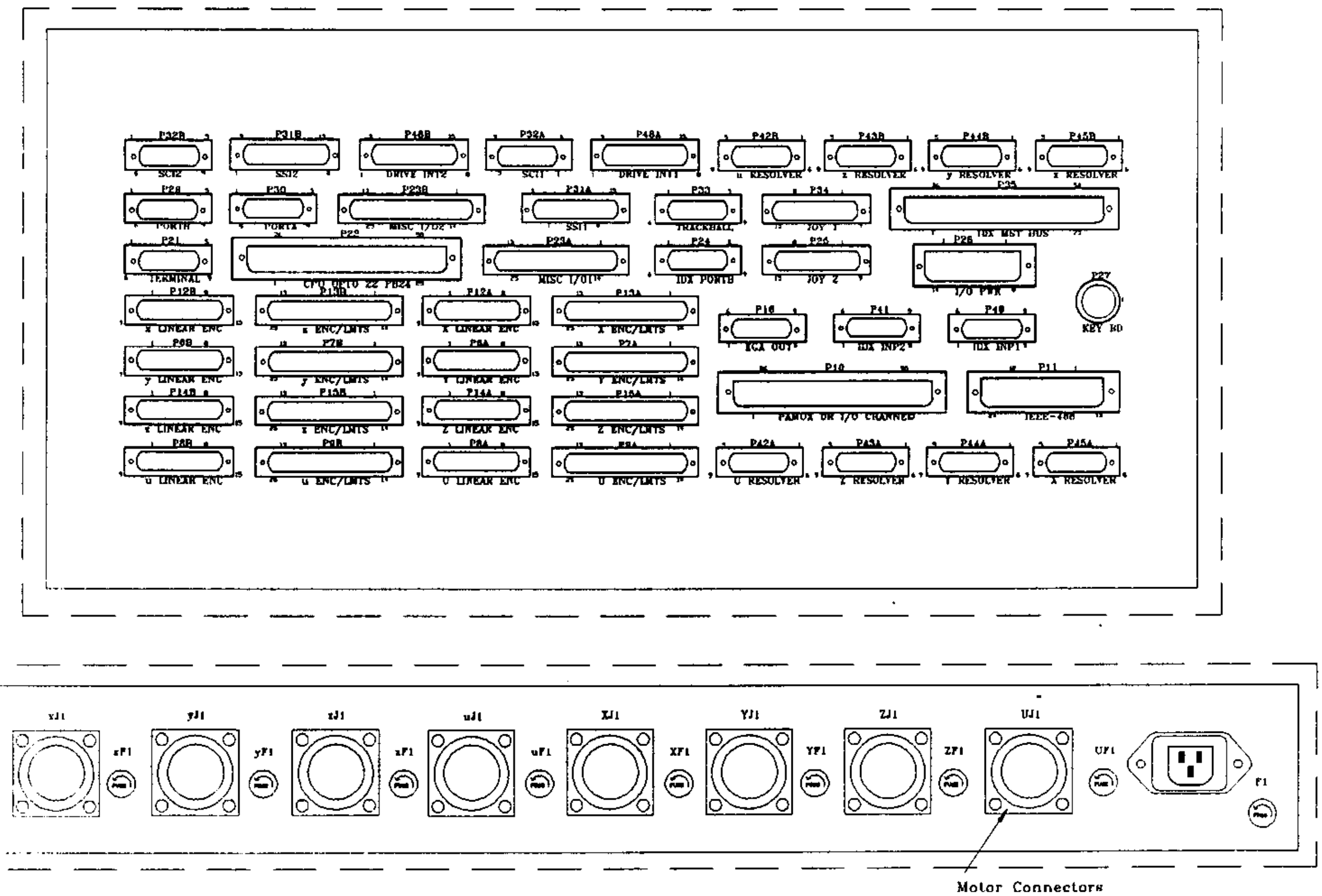


Figure 2-1: Unidex 21, Model U21B and U21R Rear Panel



NOTE: The relative location of the 8 Axis Rear Connector Panel and Motor Connector Panel is dependent upon packaging configurations.

Figure 2-2: Unidex 21, Model U21C Rear Panel

SECTION 2-2: POWER UP

WARNING: Prior to Power Up of the Unidex 21, make certain that all Interface connections are correct and securely fastened. See the Unidex 21 Hardware Manual for connection details.

Move the POWER Switch of the Unidex 21 to the ON position.
The following Power Up Initial Selection menu screen will be displayed:

```
UNIDEX 21       Version xx


EPROM OK       PARAMETER OK       RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug

Active axes: ......................Alt-Z    clear message
```

The version number displayed in this Initial Selection screen, refers to the Unidex 21 Software version currently installed.

Upon Power Up the Unidex 21 tests three data storage areas.

2-2-1: EPROM VERIFICATION

A checksum is done on all data contained in the EPROM. "OK" is displayed if the integrity of the data is verified. If an error is detected, the following message is displayed:

```
EPROM checksum error ; Select function:
```

If an error exists in the System EPROM the User must contact the Customer Service Department of Aerotech, Inc. for appropriate action. Make certain the Unidex 21's serial number is available at the time of the call.

2-2-2: PARAMETER VERIFICATION

A checksum is done on all parameter settings. If an error exists between the last saved set of parameters and the currently active set of parameters the following message is displayed:

Parameter error ; Press space bar to use default data, any other key to ignore it.

To return to default parameters press the space bar/key. The User may then enter the Parameter Mode and load an error-free copy of the file containing the desired parameter settings. (See Chapter 6: Parameter Mode.)

2-2-3: RAM VERIFICATION

RAM verification is internally divided into two sections:

1) System RAM
2) User RAM

The Read and Write capability is checked in both sections of RAM.

If a Read/Write error is detected, one of the following messages is displayed:

RAM (0) fail @ (address location) = (fail data)
RAM (F) fail @ (address location) = (fail data)
RAM (5) fail @ (address location) = (fail data)
RAM (A) fail @ (address location) = (fail data)

A checksum is also done on the User's RAM to verify file integrity. Detection of a checksum error prompts the display of the following message:

RAM checksum error (filename.type)

The "filename.type" identified, is the file containing the error. To eliminate the file error, the User must enter the File Mode, delete the file containing the error, then reload a previous error-free copy of the file.

2-2-4: USER'S RAM

The number of bytes of available User's RAM is displayed.

2-2-5: MESSAGE LINE

Upon Power Up the axes that are active and clear for movement will be listed in the message line near the botttom of the screen. If the User does not desire this display, pressing the "Alt" and "Z" keys will clear the message from the display.

One of the following Error messages may appear within this display line:

Active axes: .... Doesn't Match This error message is displayed upon a System Power Up or Reset if an inconsistancy is noted between the Axis Parameter, Existance Checking setting, and the Hardware configuration.

Axis in Limit: (axis name) (direction) Alt-Z to move out

The indicated axis is positioned at a directional limit. Press the "Alt" and "Z" keys to move the axis from the limit. This message may occur at any time during operation.

Axis in Trap (axis name) (position/velocity/integral)

The indicated axis cannot be activated because the Integral position exceeds the specified limits.

Typical reasons are:

Position feedback is interrupted
Gain settings are too low

SECTION 2-3: ON-SCREEN HELP MENU

The Unidex 21 contains an extensive HELP file that is accessible from any function mode or screen. To enter the HELP file press the "Alt" and "H" keys. The HELP file displayed will be that of the HELP file last used.

An example of a typical HELP menu is as follows:

Press the "Alt" and "H" keys to enter the HELP menu.

Press the "1" key. The display will be as follows:

```
.................. (HELP message)


Abort      Block#    Copy        Down       Erase
Find       Get File  H-home      I - CI     Joystick digitizing
K          LI        M           Next page  Optional-edit
Prev page  Quit      Replace     Set & run  T-reTrieve
Up         V         Write & end

0-Main 1-Edit 2-File 3-Machine 4-P(general) 5-P(axes) 6-Test 7-System 8-Others
```

NOTE: **The Help Menu currently active will be highlighted at the bottom of the display.**

Press the capitalized letter of the sub-function for which HELP information is requested. A brief explanation of the sub-function will be displayed.

Press the number of the desired main function. The HELP menu of the selected main function will be displayed.

NOTE: **The HELP menu displayed for both Parameter functions (4P (general) and 5P (axes)) provides only the Parameter number as identification. Refer to Chapter 6: Parameter Mode, for a listing of the Parameter numbers and functions.**

SECTION 2-4: PRINT SCREEN

If the Unidex 21 is equipped with a front panel display, the User has the ability to print the current display. The printer must be connected to the Unidex 21 through the RS-232 , Port A or Port B as selected in the Parameter Mode (refer to Chapter 6 of this manual for Parameter Mode details).

To print the current display press the "Alt" and "P" keys The display screen will be sent to the printer

NOTE: **When printing a display screen containing a graphic, such as Step Response and Velocity curves, a bit mapped pattern of the screen, will be sent to the printer.**

SECTION 2-5: FRONT PANEL OPERATION

2-5-1 FUNCTION

Unidex 21 Models U21R and U21C have several system functions available directly through use of specific Front Panel keys. A brief explanation of some of these functions follows:

2-5-1-1 RESET

The RESET key is used to re-initialize the system without a complete Power Down.

2-5-1-2 OPTIONAL STOP

The OPTIONAL STOP key is used in conjunction with the M1 command. When the M1 command is decoded, if the OPTIONAL STOP key has been toggled to the ON position, program execution will stop. If the OPTIONAL STOP key is deactivated, the M1 command will be ignored. The OPTIONAL STOP status is visible at the Status Line at the bottom of the display. If the OPTIONAL STOP function is active, "Opt-Stop" will be highlighted.

Refer to the *Unidex 21 Programming Manual* for additional information.

2-5-1-3 BLOCK DELETE

The BLOCK DELETE key is used in conjunction with the / command. When the / command is decoded, if the BLOCK DELETE key has been toggled to the ON position, the program block following the / will not be processed. If the BLOCK DELETE key is deactivated, the / will be ignored. The BLOCK DELETE status is visible at the Status Line at the bottom of the display. If the BLOCK DELETE is active, "Block-delete" will be highlighted.

Refer to the *Unidex 21 Programming Manual* for additional information concerning BLOCK DELETE.

2-5-1-4 FEEDHOLD

The FEEDHOLD key is used interchangably with the "Feedhold" function that is initiated through the Machine Mode.

Refer to Chapter 5 of this Manual for detailed information concerning Feedhold.

2-5-1-5 RETRACE

The RETRACE key is used to stop a program run and reverse program processing, one block at a time.

Refer to the *Unidex 21 Programming Manual* for a detailed explanation of the RETRACE function.

2-5-1-6 CYCLE START

The CYCLE START key is used interchangably with the "C-start" function that is initiated through the Machine Mode.

Refer to Chapter 5 of this Manual for detailed information concerning Cycle Start.

2-5-1-7 MFO

The Manual Feedrate Override keys provide the User the capability to change the programmed feedrate by a designated percentage. The increment of change is determined by Parameter22 (see Chapter 6 of this Manual). Pressing the MFO + key will increase the feedrate, the MFO - key will decrease the feedrate.

In Unidex 21 Controllers containing Software Version 2.10 or greater, MFO may also be input to the Unidex 21 through a potentiometer connected to the Rear Panel's JOY 2 (P25) connector. Refer to the *Unidex 21 Hardware Manual* for connection details.

2-5-2 FRONT PANEL CONFIGURATIONS

The Unidex 21 Controller's front panel is available in three basic configurations:

Model U21B - blank

Model U21R - sealed membrane front panel keyboard with EL (electroluminescent) display

Model U21C - sealed Qwerty keyboard and color EGA monitor

Operating procedures, though similar, vary somewhat between models. The following paragraphs provide a summary of the differences and the effects of these differences on the general operating procedures as provided in the remaining sections of this manual.

2-5-2-1: Model U21B

The Unidex 21 Model U21B is a full featured version of the Unidex 21 with the exception of the Operator's front panel. (See Figure 2-1.)

The User has the option of using one of several devices to communicate with the Unidex 21, Model U21B.

TeleVideo 905 Video Display Terminal

A TeleVideo 905 Video Display terminal may be used to communicate with the Unidex 21, Model U21B. (See the *Unidex 21 Hardware Manual* for interface details.) The following functional and operational exceptions apply :

<u>Functional</u>

1) The SYSTEM LOCK switch located on the blank front panel is not active.

2) The RESET pushbutton located on the blank front panel is not active. System reset is accomplished by depressing the "CTRL" key and then pressing the "DEL" key two times.

   NOTE: **At times it may be necessary to power down the Unidex 21 to accomplish a complete reset.**

3) Power ON/OFF is controlled by the ON/OFF pushbutton on the blank front panel.

Operational

1) MFO control is accomplished by pressing the "PAGE ERASE" key to incrementally decrease the MFO value, or by pressing the "BACK TAB" key to incrementally increase the MFO value. (The MFO increment size is established in the Parameter Mode, refer to Chapter 6 of this manual for Parameter Mode details .) The MFO value may be changed in one step increments by pressing the "SHIFT" key in conjunction with the "PAGE ERASE" or "BACK TAB" key. (Refer to the *Unidex 21 Hardware Manual* for external MFO potentiometer connection details.)

2) The RETRACE function is enabled from the VT905 terminal by pressing the "CTRL" and "R" keys.

3) The FEEDHOLD and CYCLE START functions are not available directly from the VT905 terminal. They are available to the terminal User through the Machine Mode display only.

4) The OPTIONAL STOP and BLOCK DELETE enable/disable functions are not available to the VT905 terminal User.

IBM AT Keyboard and EGA Monitor

An IBM AT keyboard (84 or 101 key) and EGA monitor may be used to communicate with a U21B that is equipped with the optional front panel and CRT display function cards. (See the *Unidex 21 Hardware Manual* for interface details.) The following functional and operational exceptions apply:

Functional

1) The SYSTEM LOCK switch located on the blank front panel is active. The SYS LOCK key must be "in" for the keyboard to be active.

2) The RESET pushbutton located on the blank front panel is active. System reset may be accomplished by pressing either the RESET pushbutton on the blank front panel or by depressing the "CTRL", "ALT" and "DEL" keys on the keyboard.

3) Power ON/OFF is controlled by the ON/OFF pushbutton on the blank front panel.

Operational

1) MFO control is accomplished by pressing the "F9" key to incrementally decrease the MFO value, or by pressing the "F10" key to incrementally increase the MFO value. (The MFO increment size is established in the Parameter Mode, refer to Chapter 6 of this manual for Parameter Mode details.) The MFO value is changed in one step increments by pressing the "SHIFT" key in conjunction with the "F9" or "F10" key. (Refer to the *Unidex 21 Hardware Manual* for external MFO potentiometer connection details.)

2) The RETRACE function is enabled from the IBM keyboard by pressing the "-" key, located on the Calculator keypad.

3) The FEEDHOLD function is enabled from the IBM keyboard by pressing the "*/Prt Sc" key , of the IBM 84 key keyboard, or by pressing the "*" key on the Calculator keypad of the IBM 101 key keyboard.

4) The CYCLE START function is enabled from the either IBM keyboard by pressing the " + " key, located on the Calculator keypad.

5) The OPTIONAL STOP function is enabled from the IBM keyboard by pressing the "Scroll-Lock/Break" key of the IBM 84 key keyboard, or by pressing the "Scroll-Lock" key of the IBM 101 key keyboard.

6) The BLOCK DELETE function is enabled from the IBM keyboard by pressing the "Sys Req" key of the IBM 84 key keyboard, or by pressing the "Scroll-Lock" key of the IBM 101 key keyboard.

Remote Front Panel

The Unidex 21 Model U21B equipped with the optional CRT display function card, may be controlled by a Aerotech supplied Remote Front Panel. (See the *Unidex 21 Hardware Manual* for interface details.) The Remote Front Panel provides operation as described throughout this manual with the following functional exceptions:

1) The SYSTEM LOCK switch located on the blank front panel is active. The SYS LOCK key must be "in" for the sealed membrane keyboard to be active.

2) The RESET pushbutton located on the blank front panel is not active. System reset is accomplished by depressing the RESET pushbutton on the Remote Front Panel

3) Power ON/OFF is controlled by the ON/OFF pushbutton on the blank front panel.

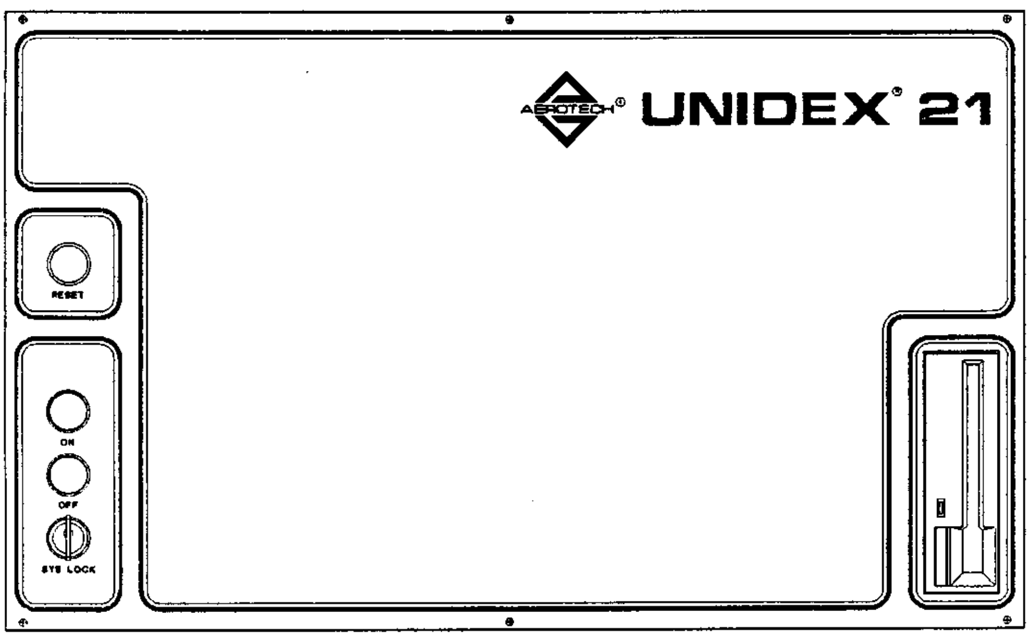


Figure 2-3: Unidex 21, Model U21B Front Panel (Shown with optional Disc Drive)

2-5-2-2: Model U21R

The Unidex 21 Model U21R is a rugged full featured rack or panel mounted version of the Unidex 21 containing a sealed membrane front panel and EL (electroluminescent) display. (See Figure 2-4.)

Operation of the Unidex 21, Model U21R is as described in the following sections of this manual.

(See the *Unidex 21 Hardware Manual* for interface details.)

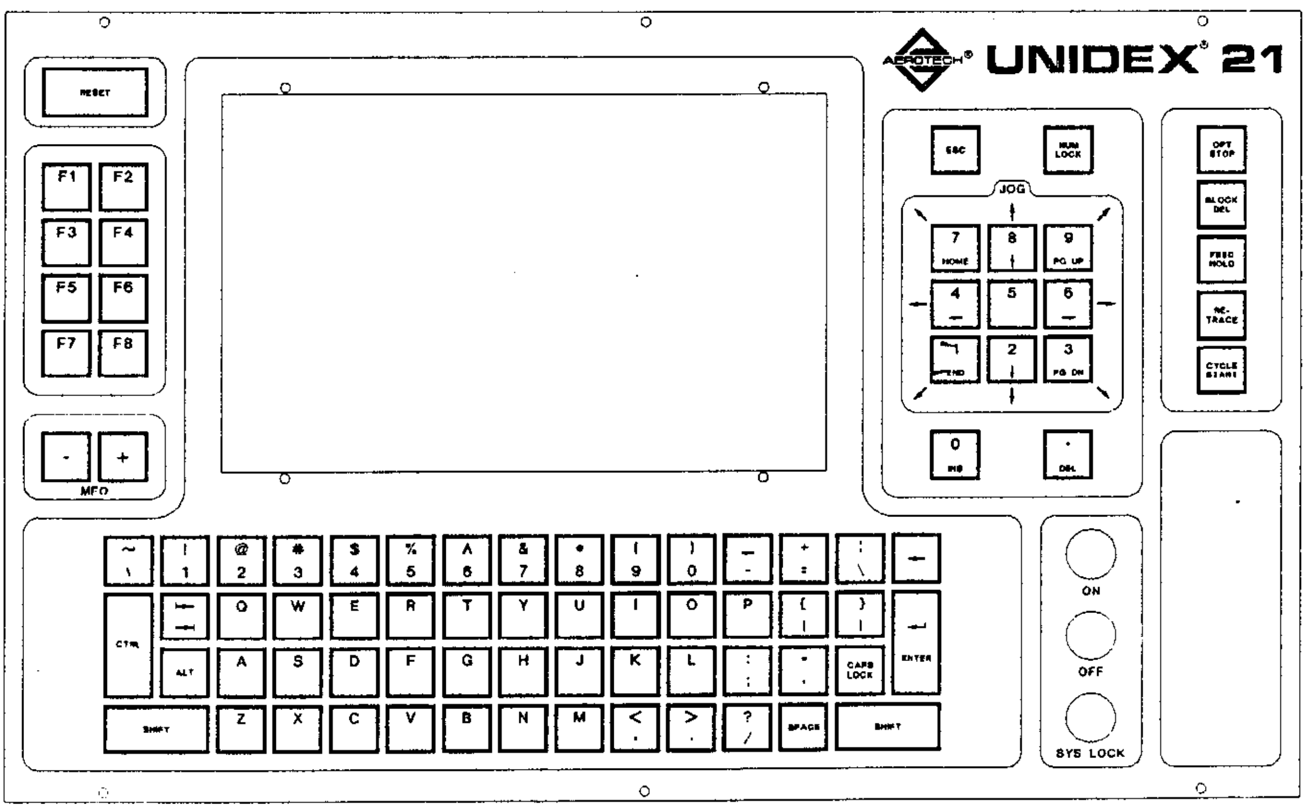


Figure 2-4: Unidex 21, Model U21R Front Panel (Shown with optional Disc Drive)

2-5-2-3: Model U21C

The Unidex 21 Model U21C is a complete multi-axis turnkey system enclosed in a NEMA 12 rated free-standing floor console. A qwerty keyboard and EGA color monitor are standard (see Figure 2-5). (See the *Unidex 21 Hardware Manual* for interface details.)

Though the keyboard has a slightly different configuration than the Unidex 21, Model U21R, operation of the Unidex 21, Model U21C is as described in this manual with the following functional exceptions.

1) The EMERGENCY STOP pushbutton located on the Model U21C console, is used to de-activate the motor drivers, stopping all axes motion. Depressing the EMERGENCY STOP pushbutton does not de-activate the Unidex 21 controller.

2) A Remote Keyboard connector is included on the front of the Model U21C Console for increased accessability. Functionally it is the same as the Rear panel connector of the other two models. (See the *Unidex 21 Hardware Manual* for interface details.)

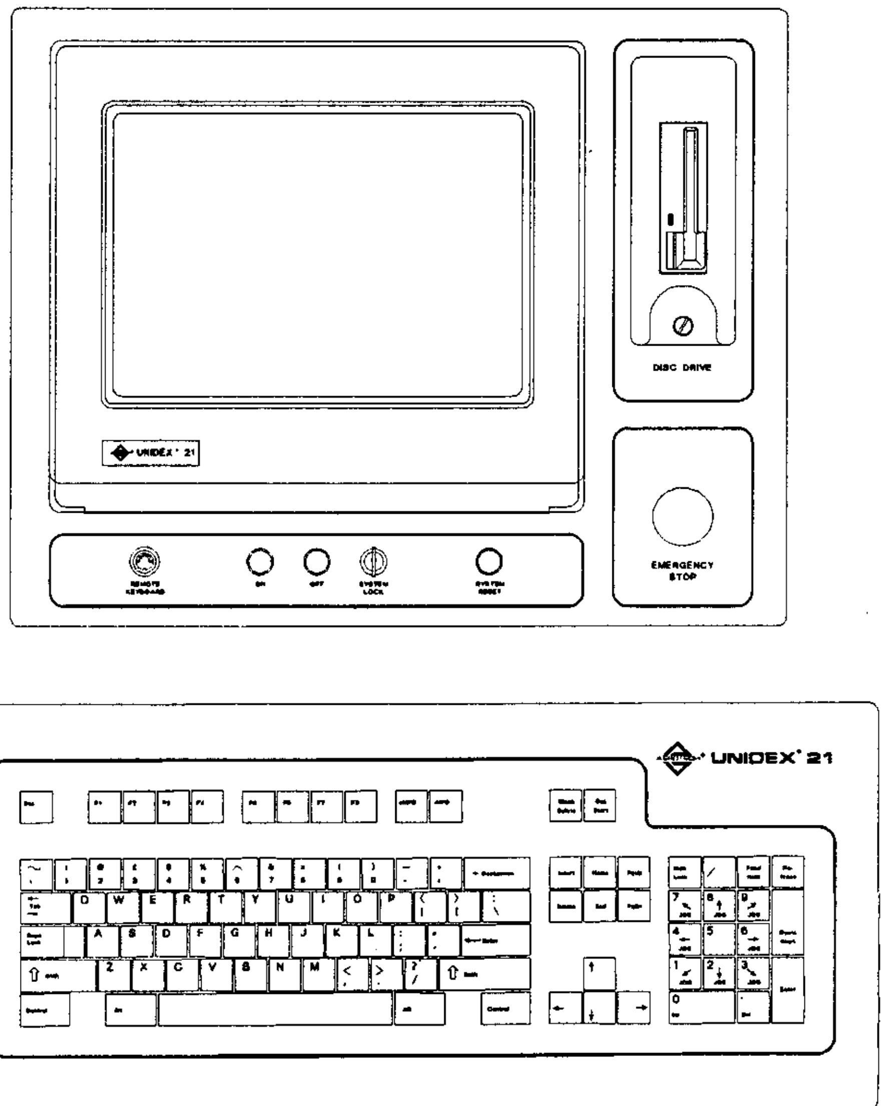


Figure 2-5: Unidex 21, Model U21C Front Panel (Shown with optional Disc Drive)

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3 : EDIT MODE

SECTION 3-1: INTRODUCTION

Unidex 21 contains a versatile editor enabling the User to edit programs by two methods; "On Screen Editing" and "Menu Driven Editing".

"On Screen Editing" is done directly by entering the desired machine code into the program block. This editing function provides a "what-you-see-is-what you-get" type of editing . The entire program is visible and is altered character-by-character.

"Menu Driven Editing" is an edit function available through an Edit sub-screen. It provides menu selections from which the Unidex 21 writes the program in machine code. Menu selections affecting the entire program as well as those creating or editing a specific program block are available.

This Chapter provides a detailed description of each of the editing methods.

SECTION 3-2: GETTING STARTED

Following Power-up the Initial Selection screen shown below will be displayed:

```
UNIDEX 21      Version  xx


EPROM  OK      PARAMETER OK      RAM  checksum
User's RAM  (bytes)  = xxxxxxx

Edit, File, Machine, Parameter. Test, System, Batch, Console, Debug
```

Enter the Edit Mode by pressing "E" from the Initial Selection screen.

The following message will be displayed:

```
Editing mode
> Ctrl (Quit, Abs, Inc digitizing) Filename.type
```

Ctrl Quit -	Press "Control Q" to return to the Initial selection screen.
Filename.type -	A filename may consist of up to 20 characters (numbers or letters) if it is to be stored in RAM or 18 characters if disk storage is used. It is suggested that the file extension be used to identify a file type, however no restrictions exist except that it be no more than 3 characters in length. Pressing the "Control" and "Q" keys will abandon the Edit mode and return to the Initial selection screen.
Ctrl A	Pressing the "Control" and "A" keys enables editing with joystick digitizing in the Absolute mode. (Refer to the *Unidex 21 Option Manual* for Joystick information)
Ctrl I	Pressing the "Control" and "I" keys enables editing with joystick digitizing in the Incremental mode. (Refer to the *Unidex 21 Options Manual* for Joystick information.)

SECTION 3-3: MAIN EDIT SCREEN

Enter the filename to edit or the name of a file to be created.

If the attributes for an existing file have been established as "Read Only" (see Chapter 4; File Mode) an applicable message will appear.

NOTE: **If a File attribute is "Read Only", it is not possible to save any on-screen editing that is performed on the file.**

The following Edit screen will be displayed:

```
      (Program Block)
      (Program Block)

CI: ON/OFF  Ctrl: (Prev. page)   (roll Down ) (Block #)   (Erase)   (Optional-edit)
LI: ON/OFF        (Next page)    (roll Up )   (reTrieve)  (Quit)    (Copy)  (Find)
                  (Set & run)    (Get file)   (Replace)   (Abort)   (Write & end)
```

The following is a description of the "On Screen" Edit mode functions:

(Program Blocks) -	Program Blocks may be entered or edited from this listing. Use of the "arrow" keys provides cursor placement.
CI: ON/OFF -	The "Insert" key is used to toggle this function (Character Insert) ON or OFF. When ON, characters are inserted without overwriting existing text.
LI: ON/OFF -	The "Control" and "Insert" keys are used to toggle the Line Insert Function ON or OFF. When ON, a new line is inserted with each carriage return.
Ctrl Prev page -	Press the "Control" and "P" keys to move the cursor to the previous page. The PG UP key performs the same function.
Ctrl Next page -	Press the "Control" and "N" keys to move the cursor to the next page. The PG DN key performs the same function.
Ctrl Set & run -	The Set & Run function provides the ability to start the run of a program at a selected block. This selection is used in conjunction with the Machine Mode. Use the "arrow" keys to move the cursor to the program block where the run is to start. Press the "Control" and the "S" keys to initiate the function. The Machine Mode main screen will be displayed and the program segment may be run. (See Chapter 5: Machine Mode)
Ctrl roll Down -	Press the "Control" and "D" keys to scroll the program display towards the display's bottom.
Ctrl roll Up -	Press the "Control" and "U" keys to scroll the program display towards the display's top.

Ctrl Get file - The **Get file** function provides the ability to retrieve and display another existing file without leaving the Edit mode.

Press the "Control" and "G" keys. The following message will appear: **Get Filename.type from, to.** Enter the filename to be retrieved. Perform a < cr > if the file is to be inserted in it's entirety. If only certain blocks of a file are to be inserted, enter the block numbers in place of **from, to.** The new file will be merged with the existing file at the cursor location.

Ctrl Block # - The Block # function provides the option of displaying block numbers before each program block. Press the "Control" and "B" keys to see: **Number Program Blocks (Yes/No).** Depress "Y" if numbers are desired.

Ctrl reTrieve - The reTreive function is used to establish the location within a program to which previously erased or copied block(s) are to be inserted. Move the cursor to the block number which will precede the inserted block(s). Press the "Control" and "T" keys, the block(s) will be inserted within the program.

Ctrl Replace - The Replace function enables the user to change character strings throughout the file without locating and changing each string individually.

Press the "Control" and "R" keys to see: **FROM string (1 to 30 char.).** Enter the character string to be replaced. The next message will be: **TO string (1 to 30 characters).** Enter the replacement character string. The last message will be: **ALL files/# of blocks.** If the replacement is to occur in all blocks, enter "A". If the replacement is to occur in a specified number of blocks only, enter this number.

Ctrl Erase - To erase blocks from a file, position the cursor at the first block to be erased. Press the "Control" and "E" keys. The display will indicate: **How many blocks to buffer?** Enter the number of blocks to be erased (start with the block containing the cursor).

NOTE: **Erased blocks remain in the system buffer and may be recalled by the "reTrieve" function.**

Ctrl Quit - Press the "Control" and "Q" keys to abandon the Edit function and return to the Initial selection screen. All changes made to the file are discarded.

Ctrl Abort - Press the "Control" and "A" keys to abort a requested function while in the Get file, Find, Replace, Erase and Copy editing functions. You will then be returned to the program being edited.

Ctrl Optional-edit - **Optional Menu Driven** editing is an alternative method of creating or editing program blocks. It provides an easy method of program editing, allowing the user to select a desired menu function which in turn inserts the proper Machine code into a program block. **Menu Driven Editing** may be used to program many common machine moves.

Press the "Control" and "O" keys to enter the **Menu Driven** Edit mode. Refer to Section 3-4 for a detailed description of the **Optional Menu Driven** Edit mode.

Ctrl Copy - Press the "Control" and "C" keys to copy program blocks. The display will indicate: **How many blocks to buffer?** Enter the number of blocks to copy (start with the block containing the cursor.)

Ctrl Write & end - Press the "Control" and "W"" keys to save any changes and return to the Main Selection screen.

Ctrl Find - The **Find** function searches a program, from the cursor block to the end, for a specified string of characters. Press the "Control" and "F" keys to activate the **Find** function. The display will indicate: **FIND string (1 to 30 char.) <CR>**. Enter the character string for the search. The cursor will move to the first occurrence of the specified string. Any editing function may then be performed. Press the "Control" and "F" keys to resume the **Find** function.

SECTION 3-4: OPTIONAL MENU DRIVEN EDITING

Optional Menu Editing may be used to edit an existing program or to create a new program.

When in the Main Edit screen, press the "Control" and the "O" keys to enter the Optional Menu Driven Editing Mode.

The following screen will appear:

```
                ***MENU DRIVEN EDITING***
M > program flow                1 > increment/absolute
2 > english/metric              3 > corner rounding
4 > ac/deceleration             5 > cutter compensation
6 > contour/point               7 > 2nd plane contour
^ > 3rd plane contour           & > 4th plane contour
8 > 1-2 plane select            * > 3-4 plane select
9 > dwell in 100 ms  =          F > contour feedrate =
] > Axes hardware home  =
[ > load axes position =

---- 1st & 2nd plane axes ----  ---- 3rd & 4th plane axes ----
X =        I =                  x =             i =
Y =        J =                  y =             j =
Z =        K =                  z =             k =
U =        P =                  u =             p =

0 > advance  =
; > comment  =
CTRL - Quit, @erase, TAB (new), SPACE (next), BACKSPACE (previous) ?
```

The Optional Edit screen indicates the status of the program block at the cursor location where the Optional Edit mode was initiated. If the block contains data not available through any of the Menu Driven Editing functions it will be displayed at **0 > advance =** .

Menu Driven Editing functions are selected by entering the symbol preceding the desired function. Pressing the symbol key repeatedly will display the selections available as a part of that function.

A description of the elements of the Optional Edit Mode follows:

M > program flow Press the "M" key to establish **program flow** code. The selections are; **program stop, optional stop, end of program, end of data** and **return to start.**

Continue to press the "M" key until the desired **program flow** condition is displayed. Press the "Enter" key, the machine code (M0, M1, M2, M32, or M47) for the desired **program flow** condition is entered into the program block at the cursor location where the Optional Edit mode was initiated.

1 > increment/absolute Press the "1" key to establish **incremental** or **absolute** program mode.

Continue to Press the "1" key until the desired positioning mode is displayed. Press the "Enter" key, the machine code (G90/G91) for the desired positioning mode condition is entered into the program block at the cursor location where the Optional Edit mode was initiated.

If no selection of position mode is made, all moves will be done incrementally.

2 > english/metric Press the "2" key to establish units in either **English** or **Metric.** When the desired unit is displayed, press the "Enter" key. The machine code (G70/G71) for the desired unit condition is entered into the program block at the cursor location where the Optional Edit mode was initiated.

If no unit is selected the program will be written in the units established in the Parameter mode.

3 > corner rounding Press the "3" key to activate the **corner rounding** mode. Continue to press the "3" key until **yes** or **no** is displayed. When the desired condition is displayed, press the "Enter" key. The machine code (G23, G24) is entered into the program block at the cursor location where the Optional Edit mode was initiated.

If no condition is selected the non-corner rounding mode will be in effect.

4 > ac/deceleration Press the "4" key to change **acceleration/deceleration** conditions. Selections are **ac only** and **de only**. When the desired condition is displayed, press the "Enter" key. The machine code (G8, or G9) is entered into the program block at the cursor location where the Optional Edit mode was initiated.

If no condition is selected both acceleration and deceleration will be active.

5 > cutter compensation Press the "5" key to establish **cutter compensation** position. Selections are **cancel, left** or **right**. When the desired condition is displayed, press the "Enter" key. The machine code (G40, G41, or G42) is entered into the program block at the cursor location where the Optional Edit mode was initiated.

If no condition is selected, **cancel** (no cutter compensation) will be in effect.

6 > contour/point Press the "6" key to specify the move type. Selections are **point-to-point, linear, CW circular,** and **CCW circular**. When the desired move type is displayed, press the "Enter" key. The machine code (G0, G1, G2, or G3) is entered into the program block at the cursor position where the Optional Edit mode was initiated.

If no condition is specified, **linear** movement will be in effect.

7 > 2nd plane contour Press the "7" key to specify the move type of the 2nd plane. The selections are **linear, CW circular** and **CCW circular.**

When the desired move type is displayed, press the "Enter" key. The machine code (G11, G12, or G13) is entered into the program block at the cursor position where the Optional Edit mode was initiated.

If no condition is selected, **linear** movement will be in effect.

^ >3rd plane contour Press the "^" key to specify the move type of the 3rd plane. The selections are **linear, CW circular** and **CCW circular.**

When the desired move type is displayed, press the "Enter" key. The machine code (H1, H2, or H3) is entered into the program block at the cursor position where the Optional Edit mode was initiated.

If no condition is selected, **linear** movement will be in effect.

& >4th plane contour Press the "&" key to specify the move type of the 3rd plane. The selections are **linear, CW circular** and **CCW circular.**

When the desired move type is displayed, Press the "Enter" key. The machine code (H11, H12, or H13) is entered into the program block at the cursor position where the Optional Edit mode was initiated.

If no condition is selected, **linear** movement will be in effect.

8 > 1-2 plane select Press the "8" key to specify the axis plane designations for the 1st and 2nd plane. The selections are **XY/ZU, ZX/YU**, and **YZ/XU**.

When the desired axes designations are displayed, press the "Enter" key. The machine code (G17, G18, or G19) is entered into the program block at the cursor position where the Optional Edit mode was initiated.

If no condition is selected, **XY/ZU** will be the designated as the 1st and 2nd axes planes.

*** >3-4 plane select** Press the "*" key to specify the axis plane designations for the 3rd and 4th plane. The selections are **xy/zu, zx/yu**, and **yz/xu**. When the desired axes designations are displayed, press the "Enter" key. The machine code (H17, H18, or H19) is entered into the program block at the cursor position where the Optional Edit mode was initiated.

If no condition is selected, **xy/zu** will be the designated as the 3rd and 4th axes planes.

9 > dwell in 100ms = Press the "9" key to specify a dwell within the program. Key in the desired dwell time in 100ms blocks (7 digits maximum). The machine code (G4) will be entered into the program block at the cursor position where the Optional Edit mode was initiated.

F > contour feedrate = Press the "F" key to specify the contour feedrate (inches/minute). Key in the desired feedrate (9 digits maximum). The feedrate will be incorporated into the program at the cursor position where the Optional Edit mode was initiated.

] > Axis hardware home = Press the "]" key to specify an axis home within the program. Key in the home reference position. The home reference position will be entered into the program block at the cursor position where the Optional Edit mode was initiated.

[> load axis position = Press the "[" key to load in new axes references for the present axis position. The machine code (G92) is entered into the program at the cursor position where the Optional Edit mode was initiated.

0 > advance = A block of machine program to be edited that is not recognized by the Menu Driven Edit Menu is displayed at the **0 > advance =**. Press the "0" key to edit the displayed 0 program block. When editing is complete press the "Enter" key.

; > comment = Press the ";" key to enter comments or notes to follow a program block. Comments or notes are visible from within the program but have no effect on the intent of the program.

CTRL - Quit Press the "Control" and "Q" keys to return to the Main Edit screen.

@ erase Press the "Shift" and "@" keys to erase the currently displayed program block and associated comments.

TAB (new) Press the "TAB" key to insert a new program block into an existing program.

SPACE Press the "SPACE" bar/key to bring the next program block into the Optional Edit screen.

BACKSPACE (previous) Depress the "BACKSPACE" key to bring the previous program block into the Optional Edit screen.

When Optional Menu Driven Editing is complete, press the "Control" and "Q" keys to return to the Main Edit Menu. To save program changes that were made in the Optional Menu Driven Edit Mode, depress the "Control" and "W" keys.

OPTIONAL MENU EDITING PROGRAM EXAMPLE

Optional Edit Menu Entry	Resulting Program
] X,Y,Z	(REF, X,Y,Z)
2 Enter English	G70
1 Enter Absolute	G90
F 1000	F1000
X 10.	
Y5.	X10.,Y5.
9 2	G4 F0.5.
X 20.	
Y10.	X20.,Y10.
M Enter program stop	M0

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 4: FILE MODE

SECTION 4-1: INTRODUCTION

The Unidex 21 provides a full range of file management capabilities. Through the keyboard, files which are retained in the Unidex 21's internal memory as well as files accessable through the optional floppy disk drive, may be listed, verified, transferred, etc.

This chapter provides a detailed description of each of the Unidex 21 file management functions.

SECTION 4-2: GETTING STARTED

Following Power-Up the initial selection screen below is displayed:

```
UNIDEX 21        Version xx


EPROM OK        PARAMETER OK        RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug
```

Press the "F" key to enter the File Mode. The following message will be displayed:

```
Attrib, Backup, Copy, Dir, Erase, Input, Output, PLC, Load-from-pointer,
Rename, Sumcheck, Transfer, Verify, aXiscal, Ctrl Quit


> Select function
```

SECTION 4-3: FILE MANAGEMENT FUNCTIONS

4-3-1: ATTRIBUTES

The File Attribute function permits the user to establish conditions to limit the accessibility of a particular file.

Press the "A" key to establish the access attributes of a file.

The Display will be:

```
Memory file attribute function
> Ctrl - Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the name of the file to set attributes.

The display will be:

```
> Read only?    Write - read?    No change?
```

Read Only - Press the "R" key to prevent any alterations from being made to the file. While Attribute is set to **Read only**, an attempt to alter the file, will result in the following message:

```
File Attribute Read Only. Hit "Enter" to continue
```

Write-Read - Press the "W" key to allow editing of the designated file.

No change - Press the "N" key to retain a previously established attribute. No change is most often used when the Attribute function has been entered inadvertedly and no status change is desired.

4-3-2: BACKUP

The File Backup function provides file management utilities for the optional Floppy Disk Backup system. Press the "B" key to see the following Backup sub-menu:

```
Floppy disk utility

Attrib, Dir, Erase, Format, Output, Rename, Verify, Ctrl Quit,
Load-to-Memory, Save-to-disk

> Select function:
```

4-3-2-1: BACKUP ATTRIBUTES

Press the "A" key to establish the access attributes of a floppy disk file.
The Display will be:

```
Disk file attribute function
> Ctrl - Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main Backup File menu.

Enter the name of the file to set the attributes.
The Display will be:

```
> Read only?    Write - read?    No change?
```

Read Only - Press the "R" key to prevent alterations from being made to the file. While Attribute is set to **Read only**, any attempt to alter the file, will result in the following message:

```
File Attribute Read Only. Hit "Enter" to continue
```

Write-Read - Press the "W" key to allow editing of the designated file.

No change - Press the "N" key to retain a previously established attribute. No change is most often used when the Attribute function has been entered inadvertently and no status change is desired.

4-3-2-2: BACKUP DIRECTORY

Press the "D" key for a directory of the files available from the floppy disk drive. The display will be:

Disk file name	Type	Status	Length	Last edit date
....	....	....	....	....

More, others Quit
> press any key to Quit, Mem.left (bytes) = xxxxxx

Disk file name - The name of all files contained on the floppy disk will be listed below.

Type - The file name extension is listed below.

Status - The file Status column indicates the Attribute setting of a file. A file having a "R" in this column has been set for "Read only" through the Attribute function. This file cannot be altered in any way. A file having a "B" in this column indicates the file consists of binary data. A file with no entry in this column is set for "Write - Read" and may be altered by the User.

Length - The file Length column indicates the number of bytes a file occupies on the disk.

Last Edit Date - The Last Edit Date column indicates the date a file was last copied.

Memory.left (bytes) - "Memory left" indicates the number of bytes still available on the floppy disk.

Press any key to return to the Backup file menu.

4-3-2-3: BACKUP ERASE

Press the "E" key, the display will be:

```
> Ctrl Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the Backup file menu.

Enter the filename.type to be erased. The file will be erased from the disk.

4-3-2-4: BACKUP FORMAT

The Format function is used to format a floppy disk within the Unidex 21's disk drive. Prior to formatting a disk, the disk configuration, single or double sided, must be selected in the Parameter Mode (Chapter 6).

From the Backup file menu, press the "F" key. The display will be:

```
WARNING!! Format will erase all data on disk.

> Press <CR> to continue, any other key to quit.
```

When format of the disk is complete the following message will be displayed:

```
***Format complete, Format another? (Yes/No)
```

4-3-2-5: BACKUP OUTPUT

The Output function permits the user to send files from the floppy disk to the RS-232 Output.

Press the "O" key to initiate the RS-232 output function.

The display will be:

```
Output files from disk to RS-232/IEEE488
> Ctrl - Quit, Ctrl - All <CR>, filename.type =
```

Press the "Control" and "Q" keys to return to the main Backup File menu.

Press the "Control" and "A" keys to send all files on the floppy disk through one of the RS-232 ports or the IEEE-488 Port.

Enter the filename.type to be sent through the RS-232 or IEEE-488 Port.

The display will be:

```
> port-A, port-B, Ieee488, Ctrl-Quit
```

Press the "A" key to send the file through RS-232 Port-A. Press the "B" key to send the file through RS-232 Port-B.

Press the "I" key to send the file through the IEEE-488 Port.

The file output format will be:

(header)
%
file text
(end-of-file)

Press the "Control" and "Q" keys to return to the main Backup File menu.

4-3-2-6: BACKUP RENAME

Press the "R" key to change the name of a file on the floppy disk.

The display will be:

```
Rename file in disk
> Ctrl-Quit, filename.type=
```

Press the "Control" and "Q" keys to return to the main Backup File menu.

Enter the filename.type to be renamed.

The display will be:

```
> Ctrl-Quit, filename.type
```

Press the "Control" and "Q" keys to return to the main Backup File menu.

Enter the new filename.type.

4-3-2-7: BACKUP VERIFY

The Verify function provides the user with a method of checking files and or recorded data that has been copied onto the floppy disk, against the original file that remains in system memory.

Press the "V" key to initiate file or data verification. The display will be:

```
Verify file or recorded data from disk to memory
>Ctrl-Quit, Ctrl-All <CR> filename.type =
```

Press the "Control" and "Q" keys to return to the main Backup File menu.

Press the "Control", "A" and "Enter" keys to Verify all the files on the floppy disk against the corresponding files in the system memory.

NOTE: Recorded (Binary) data files must be verified individually.

Enter the filename.type from the disk, which is to be verified.
The display will be:

```
>Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main Backup file menu.

Enter the memory filename.type to be used to verify the disk file.

Following file verification, one of two messages will be displayed:

```
Verify OK, press any key to continue
```

```
Verify fail, press any key to continue.
```

Press any key to return to the main Backup file menu.

4-3-2-8: BACKUP QUIT

Press the "Control" and "Q" keys to return to the main File menu.

4-3-2-9: BACKUP LOAD-TO-MEMORY

The Load-to-Memory function is used to transfer files from the disk drive to the Unidex 21 memory. Depress the "L" key to initiate the Load-to-memory function.

The display will be:

```
Load file(s) or recorded data from disk to memory
>Ctrl-Quit, Ctrl-All <cr>, filename.type =
```

Press the "Control" and "Q" keys to return to the main Backup File menu.

Press the "Control" and "A" keys to load all of the files contained on the floppy disk into the Unidex 21 memory.

NOTE: **Recorded (Binary) data files must be loaded individually**

Enter the disk filename.type which is to be loaded into the Unit's memory.

The display will be:

```
>Ctrl-Quit, filename.type =
```

Enter the filename.type of the file which will reside in the Unidex 21's memory.

NOTE: **The Load-to-memory function will not over-write an existing file. To update a file in memory from the floppy disk, it is necessary to first erase/rename the file from the Unidex 21 memory, then load the file from the disk to memory.**

4-3-2-10: BACKUP SAVE-TO-DISK

The Save-to-disk function allows the User to copy files from the Unidex 21's memory onto the Floppy disk. Depress the "S" key to initiate the Save-to-disk function.

The display will be:

```
Save file(s) or recorded data from memory to disk
>Ctrl-Quit, All, Recorded <cr>, file =
```

Press the "Control" and "Q" keys to return to the main Backup File menu.

Press the "Control", "A" and "Enter" keys to copy all files from the Unidex 21's memory to the Floppy disk.

Press the "Control" and "R" keys to copy Recorded (Binary) files from the Unidex 21's memory to the Floppy disk.

NOTE: Recorded (Binary) data files must be backed up individually.

Enter the filename.type of the file to be saved.

The display will be:

```
>Ctrl-Quit, filename.type =
```

Enter the disk filename.type which will receive the file being saved.

NOTE: The Save-to-disk function will not over-write an existing file. To update a file onto floppy disk from memory, it is necessary to first erase/rename the file from the disk, then save the file from memory to disk.

4-3-3: COPY

The File Copy function permits the user to duplicate files within the Unidex 21's memory. Press the "C" key from the main File menu to see:

```
Copy file from memory to memory
>Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the File main menu.

Enter the filename.type of the origin file.

The display will be:

```
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the new filename.type. The file will be copied to this name.

4-3-4: DIRECTORY

To initiate a Directory listing of the Unidex 21's memory, press the "D" key. The display will be:

```
memory file name   type   status   length   sumchk   last edit date



> More, others  Quit
> Press any key to quit, memory left (bytes) = xxxxxxx     allocated memory = xxxxxxxx
```

Memory File name - The names of all of the files contained in memory will be listed below.

Type - The file name extension is listed below.

Status - The file status column indicates a file's Attribute setting. A file having an "R" in this column has been set for "Read only" through the Attribute function. This file cannot be altered in any way. A file with no entry in this column is set for "Write-read" and may be edited by the User.

A file with an "A" in this column is set "Active" to run and cannot be altered. This is established in the Machine mode.

Length - The file Length column indicates the number of bytes a file occupies in memory.

Sumchk - The alpha-numeric number listed in the Sumchk column is derived from an internal summation of the bytes in a file. It is used for file integrity verification.

Last edit date - The last edit date column indicates the most recent date a file was created or copied.

Memory left - "Memory left" indicates the number of bytes still available in the Unidex 21's memory.

Allocated memory - The number of memory bytes, allocated by the **MALC** command to run backround functions.

Press any key to return to the File menu.

4-3-5: ERASE

The Erase function permits the user to delete entire files from the Unit's memory. Press the "E" key to see:

```
> Ctrl Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the filename.type to be deleted. The file will be deleted from the Unidex 12's memory.

4-3-6: INPUT

The Input function is used to import files through one of the RS- 232 ports or the IEEE-488 port into the Unidex 21's memory.

Press the "I" key to initiate the Input function.
The display will be:

```
> Input file from RS-232 /IEEE488 to memory
> port-A, port-B, IEEE488, Ctrl-Quit
```

Press the "A" key to receive the file through the RS-232, Port A.
Press the "B" key to receive the file through the RS-232, Port B.
Press the "I" key to receive the file through the IEEE-488 Port.
The display will be:

```
> Port-A/B, IEEE488 to
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the filename.type to be received through the RS-232 or IEEE488 Ports. The file will be input into the Unidex 21's memory. The source file must contain a "%" at the beginning of valid data and an "end-of-file" code at the end.

4-3-7: OUTPUT

The Output function is used to send files from the Unidex 21's memory through one of the RS-232 ports or the IEEE488 port.

Press the "O" key to initiate the Output function.
The display will be:

```
> Output files from memory to RS-232/IEEE488
> Ctrl-Quit, Ctrl-All <cr>, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Press the "Control", "A" and Enter keys to output all files from the Unidex 21's memory, through the RS-232 or IEEE488 ports.

Enter the filename.type to be output through the RS-232 orIEEE488 ports.
The display will be:

```
> port-A, port-B, IEEE488 .Ctrl-Quit
```

Press the "A" key to send the file through the RS-232, Port A.
Press the "B" key to send the file through the RS-232, Port B.
Press the "I" key to send the file through the IEEE-488 Port.

The file output format for a single file is:

(header)
%
file text
(end-of-file)

The file output format for multi-files is:

(header)
%
file text
(end-of-file)
(header)
%
file text
(end-of-file)
....
....
(end-of-all-file)

NOTE: **The Unidex 21 is equipped with an RS-232/IEEE-488 Time-out feature. When sending or receiving data, if the external device does not respond to input/output in a predetermined length of time (see Chapter 6: Parameter Mode) an error message will be displayed.**

Press the "Control" and "Q" keys to return to the main File menu.

4-3-8: RENAME

The Rename function permits the renaming of any file in the Unidex 21's memory. Press the "R" key, the display will be:

```
Rename file in memory
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the filename.type of the file to be renamed.
The display will be:

```
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.
Enter the new filename.type. The File will be renamed.

4-3-9: PLC

The Programmable Logic Controller Option. (Refer to the *Unidex 21 Option Manual* for a description of the PLC option.)

4-3-10: LOAD-FROM-POINTER

The Load-from-pointer function provides access to files stored in a User's auxiliary memory board. Press the "L" key, the display will be:

```
Load file from address pointer to user's memory
> Ctrl-Quit, Address pointer (Hexadecimal) = xxxx
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the Hexadecimal file location. The display will be:

```
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the filename.type for the file being loaded. The source files being loaded must contain a "%" at the beginning of valid data and an "end-of-file" code at the end. (See Chapter 6: Parameter Mode.) The file will now appear in the system directory.

4-3-11: SUMCHECK

The Sumcheck function is used to compare a file contained in the systems memory against a file which has been output through an RS- 232 or IEEE-488 port. Press the "S" key to initiate the Sumcheck function.

The display will be:

```
Output memory file sum-check to RS-232/IEEE488
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main file menu.

Enter the filename.type. The display will be:

```
> port-A, port-B, IEEE488, Ctrl-Quit
```

Press the "A" key to send the sumcheck to RS-232 Port A.
Press the "B" key to send the sumcheck to RS-232 Port B.
Press the "I" key to send the sumcheck to the IEEE-488 Port.
Press the "Control" and "Q" keys to return to the main File menu.

4-3-12: TRANSFER

The Transfer function establishes communication with another AEROTECH device.

NOTE: **A file must have been previously developed to initialize the peripheral AEROTECH device.**

Press the "T" key to initiate the Transfer function. The following series of displays are applicable when establishing communication with another AEROTECH device.

Transfer data to another AEROTECH system **> Ctrl-Quit, filename.type =**

Press the "Control" and "Q" keys to return to the main File menu.

Enter the filename.type of the initialization file to activate the Transfer function. The display will be:

Ctrl-Quit, Input-file, Output-file, Transfer-only:

Press the "Control" and "Q" keys to return to the main File menu.

Press the "I" key to Input a file from another AEROTECH device. The display will be:

> Input file after transfer complete **> port-A, port-B, IEEE488, Ctrl-Quit**

Press the "A" key to input the file through RS-232 Port A.
Press the "B" key to input the file through RS-232 Port B.
Press the "I"key to input thr file through the IEEE-488 Port
Press the "Control" and "Q" keys to return to the main File menu.

The display will be:

> port-A/B , IEEE488,to **> Ctrl-Quit, filename.type =**

Press the "Control" and "Q" keys to return to the main File menu.
Enter the filename.type to which the the file being input will be written.

To Output a file to another AEROTECH device following initialization of the Transfer mode, press the "O" key. The display will be:

> **> Output file after transfer complete**
> **> Ctrl-Quit, Ctrl All, filename.type**

Press the "Control" and "Q" keys to return to the main File menu.

Press the "Control" and "A" keys to Output all files from the User's Unidex 21 memory to another AEROTECH device.

Enter the filename.type of the file which is to be Output from the User's Unidex 21.

The display will be:

> **> port-A, port-B, IEEE488, Ctrl-Quit**

Press the "A" key to Output the file through RS-232 Port A.
Press the "B" key to Output the file through RS-232 Port B.
Press the "I" key to Output thr file through the IEEE-488 Port.
Press the "Control" and "Q" keys to return to the main File menu.

"Transfer-only" sends only the initialization file from the User's Unidex 21 to the peripheral AEROTECH device. When the "Transfer- only" function is complete, communication is possible between the AEROTECH device and the User's Unidex 21.

Press the "T" key, the display will be:

> **Transfer data to another AEROTECH system**
> **> Ctrl-Quit, filename.type =**

Press the "Control" and "Q" keys to return to the main File menu.

Enter the filename.type of the initialization file.
The display will be:

```
> Transfer data only
> port-A, port-B, IEEE488, Ctrl-Quit
```

Press the "A" key to transfer data through RS-232, Port A.
Press the "B" key to transfer data through RS-232, Port B.
Press the "I" key to transfer the data through the IEEE-488 Port.
Press the "Control" and "Q" keys to return to the main File menu.

4-3-13: VERIFY

The Verify function provides the User with a method of assuring that file manipulations have not affected the integrity of a file.

Press the "V" key from the main File menu . The display will be:

```
Verify file from memory to memory:
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.
Enter the filename.type of the the file to be verified.

The display will be:

```
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the filename.type of the file to which the file to be verified should be compared.

4-3-14: AXISCAL

The aXiscal function provides the User with alternatives for storing Axis Calibration data. Calibration data may either be saved from the Indexing Board to RAM in the form of a file, or loaded as a file from RAM to the Indexing Board for processing.

Press the "X" key from the main File menu. The display will be:

```
Axis calibration data, Save to RAM or Load to INDEXING

>ctrl - Quit,  S1 to S8,  L1 to L8
```

Press the "Control" and "Q" keys to return to the main File menu.

Press S1 through S8 to transfer Axis Calibration data from the Indexing Board to a file to be stored in RAM. The display will be:

```
>ctrl - Quit, filename.type
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the Filename.type of the file to which the Axis Calibration data will be saved in RAM.

Press L1 through L8 to load an Axis Calibration file to the Indexing Board for processing. The display will be:

```
>ctrl - Quit, filename.type
```

Press the "Control" and "Q" keys to return to the main File menu.

Enter the Filename.type of the Axis Calibration file that is to be loaded to the Indexing Board for processing.

NOTE: **Comments may be added to a File containing Axis Calibration data by preceeding the text with a ";". (See the** *Unidex 21 Programming Manual* **for details on the use of the ";".) All data located between the ";" and the block's end is ignored by the Unidex 21.**

4-3-15: CTRL-QUIT

Press the "Control" and "Q" keys to return to the initial selection menu.

CHAPTER 5: THE MACHINE MODE

5-1: INTRODUCTION

The Machine Mode is the mode in which programs are run and axis motion will occur. Prior to entering the Machine Mode, make certain that the appropriate Parameters have been established. (See Chapter 6: The Parameter Mode)

WARNING: **Prior to entering the Machine Mode, make certain all hardware is configured such that no injury to personnel or damage to equipment will occur.**

The following sections provide detailed information concerning Machine Mode programming and operation.

SECTION 5-2: GETTING STARTED

Following Power-Up, the initial selection screen is displayed:

```
UNIDEX 21      Version xx

EPROM OK       PARAMETER OK       RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter. Test, System, Batch, Console, Debug
```

Press the "M" key to enter the Machine Mode. The following screen will be displayed:

```
                X      0.000        x      0.000
                Y      0.000        y      0.000
                Z      0.000        z      0.000
                U      0.000        u      0.000


H1 H11 H17 G1 G11 ...... (system codes)        Feedrate  =  0.00        0.00

........................................(program)


Ctrl (Auto/Single [S]) (C-start) (Error ack) (Feedhold) (Track) (D-Scope)(Quit)

> Home, Jog, Mdi, Run, Slew, Wheel, Ctrl-^Abort    select :
```

SECTION 5-3: MACHINE MODE SCREEN DESCRIPTION

A description of the display follows:

5-3-1: X 0.000 x 0.000
Y 0.000 y 0.000
....

Relative tracking position is displayed for each axis as the program is run. Position is displayed either in machine or program steps (See Chapter 6: Parameter Mode).

5-3-2: (system codes)

These codes indicate the status of the following parameters:

Contour Type (G1,G2,G3,G5,H1,H2,H3,H5)
Contour Plane ((G17,G18,G19,H17,H18,H19)
Corner/Non-Corner Rounding (G23,G24)
ICRC (G40,G41,G42)
Metric/English (G71/G70)
Absolute/Incremental (G90/G91)

5-3-3: Feedrate

Feedrate is the speed at which the axes execute a programmed move. The left display indicates the Feedrate as programmed. The right display indicates the programmed Feedrate multiplied by the MFO% value.

5-3-4: (Program display)

Each block of the program is displayed as it is being executed. If Block numbers have been established in the Edit Mode they will be displayed before each block.

5-3-5: Ctrl (Auto/Single [S/A]

A program may be run one block at a time or continuously from start to finish. The default setting runs the program one block at a time. Press the "Control" and "C" keys, or the Front Panel's CYCLE START button to step through the program.

Press the "Control" and "A" keys to run the program automatically.

5-3-6: Cycle Start

Press the "Control" and "C" keys or the Front Panel's CYCLE START key to step through a program that is to be run, one block at a time.

The Cycle Start function may be factory set to respond to continuous key depression in one of two ways; either continuous key depression will have no effect on the single program block run, and each key depression will run only one block, or continuous key depression will result in program blocks being run one after another as long as the key continues to be depressed.

This function may also be used to continue a program following a Program Stop (M0) or Optional Program Stop (M1).

5-3-7: Error ack

At times, during the run of a program, an error message may appear. If instructions are not provided to alleviate the error condition, the message may be cleared to permit further action, by depressing the "Control" and "E" keys, acknowledging the error message and continuing with the requested function..

5-3-8: Feedhold

Press the "Control" and "F" keys to halt all axis motion. The Front Panel FEEDHOLD key may also be used for this function. Press the "Control" and "F" keys a second time to resume axis motion.

Several conditions exist concerning the Feedhold operation :

1) Use of the Abort function while in the Machine Mode will cancel a previously established Feedhold.

2) If a Feedhold is initiated while using the Mdi function of the Machine Mode, in order to quit the function, either the Feedhold must be released to allow completion of the move or the move must be aborted ("Control" " ^ ").

3) If a Feedhold is initiated while in the Machine Mode's Jog function, in order to quit the function, either the Feedhold must be released to allow completion of the move, or the system must be Reset. Aborting the move ("Control"" ^ ") will not permit exit from the Jog function.

4) If a Feedhold is initiated when a program is in the Machine Run Mode the following message will be displayed:

```
> <release feedhold>, Jog, Wheel, Ctrl- ^ abort     select:
```

Press the "Control" and "F" keys or the Front Panel FEEDHOLD key to cancel the Feedhold and continue running the program.

Press the "J" key to enter the Jog mode. (See Section 5-3-13 for a description of the Jog function.)

Press the "W" key to enter the Handwheel mode. (See Section5-3-17 for a description of the Handwheel function.)

Press the "Control" and " ^ " keys to abort this function and return to the Machine Mode menu.

5-3-9: Track

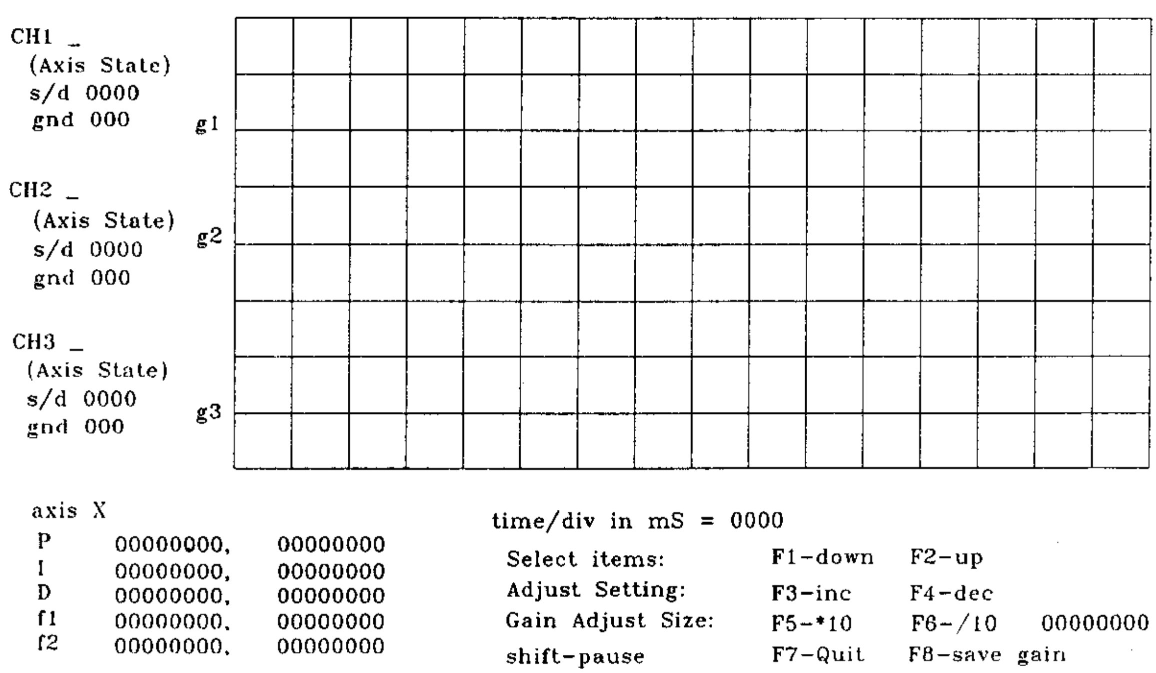
The Unidex 21 automatically updates the Machine Mode tracking display unless instructed otherwise from within the program (TRAK,0) or through an MDI entry. When the automatic tracking has been disabled in one of these ways, current information may be displayed by depressing the "Control" and "T" keys.

5-3-10: D-Scope

The Digital Scope function permits the User to view positioning functions while a program is being run.

Press the "Control" and "D" keys to enter the D-Scope Mode.

The display will be:



The following is a description of the Digital Scope functions:

CH1_/CH2_/CH3_ The axis to be profiled is entered at this location. Three axes profiles may be viewed simultaneously.

Axis State The User selects the Axis State to be profiled. The Axis States available for profiling are;

mS Command - displays the motion profile that is currently executing.

Position Error - displays an indication of the difference between the requested position and the actual position.

Integral Error - displays the accumulation of the position error.

Velocity Error - displays an indication of the difference between the requested velocity and the actual velocity.

s/d The desired number of machine steps per vertical division is entered at this location.

gnd Indicates the relative location of the ground to the baseline.

Axis_ Designates the name of the axis for which the gain and feedforward settings are displayed.

The following group of settings are the PID and Feedforward Gains for a designated axis. The first number is the default setting or the setting that has been established within the Auto-Tune procedure. (Refer to Chapter 6 : Parameter Mode, of the *Unidex 21 User's Manual.*) The second number may be established by the User as a proposed PID or Feedforward setting. These settings may be modified by the User and the effect upon the motion profile can be monitored on the display. When the desired system response is obtained, these settings may then be saved and will become the new default parameters.

P	00000000,	00000000	Proportional Gain Setting.
I	00000000,	00000000	Integral Gain Setting
D	00000000,	00000000	Derivative Gain Setting
f1	00000000,	00000000	Velocity Feedforward Setting
f2	00000000,	00000000	Acceleration Feedforward Setting

time/div in mS — Provides the User the ability to establish the number of milli-Seconds each division will represent.

Select Items — The F1 key moves the cursor down through the list of options. The F2 key moves the cursor up through the list of options. The cursor must be located at the setting to be altered.

Adjust Settings — When the cursor is located at the setting to be changed, the F3 key may be used to increment through the setting variables. The F4 key may be used to decrement through the setting variables.

Gain Adjust Size — Increments (F5) or decrements (F6) by a power of 10 the constant that is located to the far right of F6. This constant is used to modify PID and Feedforward Gains (i.e. when the constant is set at 1000, pressing F4 or F3 will cause the selected Gain value to increment or decrement by 1000).

Shift — Causes the display to pause.

Quit Press the F7 key to quit the D-Scope Mode and return to the Machine Mode.

Save Gain Press the F8 key to save the Gain and Feedforward settings that were established while in the D-Scope Mode.

5-3-11: Quit

Press the "Control" and "Q" keys to return to the previous selection screen.

5-3-12: Home

The Home function of the Machine Mode is used to send any axis to it's Home position. If it is requested within a program, all axis movement is stopped and the selected axis is returned to it's Home position.

The Feedrate and direction of the Home move is established within the Parameter Mode (Chapter 6 of this manual).

Following a Home function the Auto-Single run mode returns to the default of Single block movement.

Press the "H" key from the main Machine menu to select the axis to send Home. The display will be:

```
home >              X (X axis)  Y (Y axis)  Z (Z axis)  U (U axis)
(selected axis)     x (x axis)  y (y axis)  z (z axis)  u (u axis)
```

Press the key(s) of the axes to be sent Home. If the Tracking display is ON, the axis position display for the appropriate axis should go to zero. When axes selection is complete, press the "Control" and "Q" keys to return to the main Machine Mode menu.

5-3-13: Jog

The Jog function enables the User to move one axis at a time with the use of the Keyboard's arrow keys.

The assignment of arrow key/axis and direction is done in the Parameter Mode's Axis settings (Chapter 6). The default setting for positive direction is CW. The default settings for arrow/axis assignment are as follows:

Press the "Control" and "G" keys to toggle between arrow/axis keys 1-4 and 5-8.

Axis 1-4	Axis 5-8
X axis = 6-4 keys	x axis = 6-4 keys
Y axis = 9-1 keys	y axis = 9-1 keys
Z axis = 8-2 keys	z axis = 8-2 keys
U axis = 3-7 keys	u axis = 3-7 keys

Press the "J" key to enter the Jog function. The following menu will be displayed:

```
jog> Top, High, Medium, Low, Step, Velocity, Ctrl-(Return), (Preset) (Group 1)
<speed = (T/H/M/L/S)    increment = (10000/1000/100/10/1) steps
```

NOTE: **If a Feedhold is initiated while in the Machine Mode's Jog function, in order to quit the function, either the Feedhold must be released to allow completion of the move, or the system must be Reset. Aborting the move ("Control"" ^ ") will not permit exit from the Jog function.**

5-3-13-1: Speed/Increment

Press the "T", "H" "M" "L" or "S" keys to set the axis speed or incremental distance of a jog move.

The Jog/Velocity function permits the User to jog an axis at a set speed but with no set incremental distance. The axis will continue to move as long as the arrow key is depressed. (The Jog/Velocity function is not valid forUnidex 21, Model U21B used in conjunction with a TeleVideo 905 Video Display Terminal.)

5-3-13-2: Return

The Return function permits the User to send the axis to the position established by the Preset function. (See next Section)

5-3-13-3: Preset

The Preset function is used to establish a temporary point to which the User desires to return. Use the Preset function as follows:

Jog the axis to a desired location. Press the "Control" and "P" keys to temporarily store this point.

The axis may now be jogged to another location, pressing the "Control" and "R" keys will return the axis to the Preset location.

The Preset location remains until overwritten.

5-3-13-4: Group 1/2

Group 1 consists of axes 1-4 (X, Y, Z,U). Group 2 consists of axes 5-8 (x, y, z,u). Press the "Control" and "G" keys to toggle between Group 1 or Group 2.

5-3-14: Mdi (Manual Data Input)

Mdi (Manual Data Input) allows the user to stop a program run, enter a move or change a status, then return to the program run. A status change by the Mdi function overrides those previously established in the program. The "up arrow" and "down arrow" keys may be used to recall previously entered Mdi data. (See Chapter 6 of this manual for Parameter Mode information, and the *Unidex 21 Programming Manual* for programming information.)

NOTE: **If a Feedhold is initiated while using the Mdi function of the Machine Mode, in order to quit the function, either the Feedhold must be released to allow completion of the move or the move must be aborted ("Control" " ^ ").**

5-3-15: Run

The Run function provides the User with the ability to execute a complete program. Press the "R" key to begin a program run, the following message will be displayed:

```
run > Filename.type =
```

The Filename.type of the program to be run may be entered or the "up arrow" and/or "down arrow" keys may be used to recall previously run filenames. Press Enter when the desired filename is displayed. If the "Single/Auto" function is set to "A" the program will soon begin to run. If the "Single/Auto" function is set to "S" only the first block of the program will run. Subsequent blocks must be run by pressing the "Control" and "C" keys or the CYCLE START key of the Front Panel. If the program was initially set to run Single and is switched to Auto the "Control" and "C" keys and or the Front Panel CYCLE START key must be pressed to start program run.

NOTE: **Some Unidex 21 systems may be factory configured such that the program will not begin to run until the "Control" and "C" keys or the Front Panel CYCLE START key is depressed.**

The "Control" and "Quit" keys may be pressed to exit the Machine Mode during the running of the program. The program will continue to run although other Unidex 21 functions are available to the User.

NOTE: **If a Feedhold is initiated when a program is in the Machine Run Mode the User is given the option of entering the Jog or Handwheel Mode then returning to the program run. (See Item 5-3-8 for details concerning Machine Mode Feedhold.)**

5-3-16: Slew

The Slew function is used in conjunction with the Joystick or Trackball option. It provides a function similar to Jog in that it permits manual axis movement. In addition to being accessable from the main Machine Mode it may also be activated from the "Mdi" mode or from the User's program.

When activated from the Machine Mode **all** axes are enabled and under Joystick/Trackball control.

The following Joystick/Trackball settings are available from the Parameter Mode (Chapter 6):

Joystick/Trackball Axis pairings for Slew operation.
Assignment of positive direction. The default setting for positive direction is CW.
An absolute window may be established governing axis movement while under Joystick/Trackball control.
The Joystick/Trackball Hi/Lo speed (steps/sec).

Press the "S" key from the Machine Mode main menu to activate the Slew function. The display will be:

```
Slew > Joystick, Trackball
```

Press the "J" key if a Joystick is being used, Press the "T" key if a Trackball is being used.

Refer to the *Unidex 21 Options Manual* for detailed instructions concerning Joystick/Trackball use.

5-3-17: Wheel

Press the "W" key from the Machine Mode main menu to activate the Unidex 21 for Handwheel use. The display will be:

```
Handwheel         X (X   ) Y(Y   ) Z(Z   ) U(U   )
(selected axis)   x (x   ) y(y   ) z(z   ) u(u   )
                  Disarm (D/A)  Input (1/2)  High (XXX)  Low
```

Press the key of the axis to be active for Handwheel use. The selected axes name will be displayed at the lower left corner of the display.

Upon axis selection the Disarm status will change to Armed (A).

NOTE: **During Handwheel operation the Handwheel may be disarmed at any time by pressing the "D" key.**

Input is available from one of two ports. Press the "I" key to toggle between Input ports 1 and 2.

The ratio of Handwheel increments to machine steps is established by the High/Low settings of this display.

Press the "H" key to increase the number of machine steps equal to each Handwheel increment. (Pressing the "H" key increases the machine steps in increments as established in the Parameter Mode (see Chapter 6 of this Manual for Parameter information). Press the "L" key to decrease the number of machine steps equal to each Handwheel increment. (Pressing the "L" key decreases the machine steps in increments as established in the Parameter Mode (see Chapter 6 of this Manual for Parameter information). (Refer to the *Unidex 21 Options Manual* for detailed Handwheel operating instructions.)

NOTE: **If the Handwheel has been previously enabled by an Mdi entry or program command, entering the Handwheel function through the Machine Mode allows the User to establish different parameters and use the Handwheel. At the conclusion of Machine Mode Handwheel use the Machine Mode Handwheel function may be "quit" and the Handwheel settings will return to those established through the Mdi or Program.**

5-3-18: Ctrl-^ Abort

A program being run or Mdi data being entered, may be aborted by pressing the "Control" and "^" keys. The program run will stop and the files set to an inactive status. Mdi data will be abandoned. If a Home or Slew move is in progress, the current move will be completed before the files are set to an inactive status.

This is the only function that will abort a program run once it has been initiated.

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 6: THE PARAMETER MODE

SECTION 6-1: INTRODUCTION

Program parameters are established within this mode. Default parameters are factory set and applicable for most program applications. Before entering the Parameter mode, the User must have established a thorough understanding of the Unidex 21 as well as a high degree of familiarity with this manual and the Unidex 21 Hardware and Unidex 21 Programming manuals.

WARNING: PRIOR TO CHANGING ANY DEFAULT PARAMETER SETTINGS, SAVE THE DEFAULT SETTINGS TO A FILE.

Prior to establishing any User parameters, save the default parameters to a file to facilitate a return to default settings. (See Section 6-7 for "Load/Save Parameter" details.) After User parameters are established, it is suggested that these parameters are also saved to a file.

Initialize User established parameters to the Unidex 21 by pressing RESET or the Power OFF push-button. Parameter settings remain in memory until changed or another parameter file is loaded.

Once User parameters have been established, save these Parameters to a file for future reference.

Current default parameters appear in brackets [ ] after entering a particular parameter. To return to the factory default setting, press the "Control", "D", and <cr> keys.

The following sections provide an explanation of each of the parameters as well as suggestions for use.

SECTION 6-2: GETTING STARTED

Following Power-Up, the initial selection screen is displayed:

```
UNIDEX 21       Version xx

EPROM OK        PARAMETER OK        RAM checksum

User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug
```

Press the "P" key to enter the Parameter mode.

SECTION 6-3: GENERAL PARAMETERS

If the instructions in "Getting Started" have been completed, the following General Parameters screen (PAGE 1) will be displayed:

```
0 : System Password                          1 : Skip Auto-Boot Function ?
2 : IDX Buffer 1 block only ?                3 : IDX seg. calculate base (1/2/3)
4 : COMM input feedback ?                    5 : System default at metric ?
6 : RS232 protocol Port-A                    7 : Additional RAM in 1024 bytes
8 : RS232 protocol Port-B                    9 : Debug display is at front panel ?
10 : RS232/IEEE488 time out (sec)            11 : Parts Program stack size in bytes
12 : Edit block buffer (1 to 40)             13 : Edit default Char-insert ?
14 : Edit default Line-insert ?              15 : Edit TAB space
16 : End of All File Code CHR$(n)            17 : End of File Code CHR$(n)
18 : Beeper Duration (1 to 280) ms           19 : Double sided floppy disk ?
20 : Beeper Frequency  (2 to 20K)            21 : Display Blank-Out (minutes)
22 : MFO inc./step (-100 to 100)             23 : Tracking Display program step ?
24 : Y pixel size reduce to (%)              25 : Print screen to port-A ?
26 : Joystick axis pair                      27 : Digitize with Joystick ?

200 : 2ND PAGE            201 : Axes Auto-Tune            202: 3RD PAGE

300 : Load/save parameter                    301 : Front panel function keys

401 : 1st axis      402 : 2nd axis      403 : 3rd axis      404 : 4th axis
405 : 5th axis      406 : 6th axis      407 : 7th axis      408 : 8th axis

ctrl-Quit, number <cr>to each parameter =
```

Enter "200", the screen will display the following General Parameters (PAGE 2).

28 : Input 1 is 0-CW/CCW, 1-CLK/DIR, 2-QUAD x 1, 3-QUAD x 2
29 : Input 2 is 0-CW/CCW, 1-CLK/DIR, 2-QUAD x 1, 3 -QUAD x 2

30 : Axis Ramp Time (1-32767) ms	31 : Power on remote control 0/1/2/3/4
32 : M Strobe Delay (0-65535) ms	33 : M Ack Delay to 65535 ms, 0 no
34 : S Strobe Delay (0-65535) ms	35 : S Ack Delay to 65535 ms, 0 no
36 : T Strobe Delay to (0-65535)	37 : T Ack Delay to 65535 ms, 0 no
38 : Quick Stop Hi-Lo trigger ?	39 : Quick Stop at trigger point ?
40 : IDX does checksum ?	41 : GANTRY (msmsmsms) m,s = 1,8
42 : Input 1 Handwheel Scale 0-254	43 : Input 2 Handwheel Scale 0-254
44 : Roll over max # 99999999	45 : H-V pairs (hvhvhvhv)h,v=1,8
46 : 1 perpendicular error arc sec	47 : 2 perpendicular error arc sec
48 : 3 perpendicular error arc sec	49 : 4 perpendicular error arc sec
50 : Reset MALC memory 0/1/2	51 : Default at Front Panel Interface?
52 : SYNC code 0?	53 : IEEE488 Setup
54 : Keep position during reset?	55 : MFO adjust Handwheel Scale?
56 : Axis Trap negate Output (0-8)	57 : Master Trajectory Linear
58 : Fast Feedhold Ramp Time (ms)	59 : Master Parabolic Coefficient

200 : PREVIOUS PAGE
>Ctrl-Quit, number <cr>to each parameter =

Enter "202", the screen will display the following General Parameters (PAGE3).

60 : G23 Exp Filter Level (1-7)	61 : Ramping During G23 Operation
62 :	63 :
64 :	65 :
66 :	67 :
68 :	69 :
70 :	71 :
72 :	73 :
74 :	75 :
76 :	45 :
46 :	77 :
78 :	79 :
80 :	81 :
82 :	83 :
84 :	85 :
86 :	87 :

A description of each of the General Parameters follows:

6-3-1 0 : SYSTEM PASSWORD

Enter "0" to establish password parameters. The display will be:

```
# password    EFMPT
1. --------------------------
2. --------------------------

Ctrl-Quit, Ctrl-Erase <nn> <cr>; Ctrl-Add <1 to 8 char>/option <cr>
```

Press the "Control" and "Q" keys to return to the Main Parameter Menu.

To erase a password, press the "Control" and "E" keys. Enter the password number which is to be erased.

To add a password, press the "Control" and "A" keys. Enter the password name (1 to 8 characters) an "/" and the letter(s) of the option(s) that the password User is to have access. The option letters are: **E** - Edit mode, **F** - File mode, **M** - Machine mode, **P** - Parameter mode, and **T** - Test mode.

NOTE: **To avoid password "lock-out" it is suggested that the primary User of this system establish a password which is active in all options.**

6-3-2 1 : SKIP AUTO BOOT

An Auto-Boot program is a User defined program which may be configured to automatically run upon a system Power-Up or Reset. The Unidex 21 treats this file as a batch file, although the program data is variable, the program name must be "AUTOBOOT.$$$". This parameter is used to turn the automatic execution of this program On/Off. The default setting skips the Auto-Boot program.

Enter "1" to change the status of the Auto-Boot program. Press the "Control" and "N" keys to toggle between Yes (the Auto-Boot program is not executed) or No (the Auto-Boot program is executed).

6-3-3 2 : IDX BUFFER 1 BLOCK ONLY

Program blocks may be sent to the Indexing Buffer in two ways. Blocks may be sent one at a time or in a continuous flow, replacing blocks in all four indexing buffers as soon as they are completed.

The default setting is to send blocks one block at a time.

Enter "2" to change the status of the Index Buffer. Press the "Control" and "N" keys to toggle between Yes (one block at a time) or No (four blocks at a time).

6-3-4 3 : IDX SEGMENT CALCULATE BASE (1/2/3/4/5)

The Segment Calculation Base Rate for the Indexing Board may be changed from the standard 1 msec rate to a rate of up to 16 msec. The default setting is 1msec.

Enter "3" to change the Segment Calculation Rate. Enter one of the following:

"1" for a 1ms rate
"2" for a 2ms rate
"3" for a 4ms rate
"4" for a 8ms rate
"5" for a 16ms rate

6-3-5 4 : COMM INPUT FEEDBACK

The Communication Input Feedback parameter determines whether the Unidex 21 requires "echo" characters for RS232 and IEEE488 input.

The default setting requires "echo" characters.

Refer to the *Unidex 21 Programming Manual* for further definition of the "COMM" command.

Refer to the *Unidex 21 Options Manual* for IEEE-488 information.

Enter "4" to change the feedback status. Press the "Control" and "N" keys to toggle between Yes (feedback echoes are required) or No (no echo characters required).

6-3-6 5 : SYSTEM DEFAULT AT METRIC

The System may be configured to use either English or Metric units as a default.

The default setting is Metric.

Enter "5" to change the units from the default of Metric (yes) to English (no).

6-3-7 6 : RS232 PROTOCOL PORT-A

The characteristics of RS232 Port-A, may be established within this parameter setting.

The default settings are; **Char** - 8, **Stop** - 1, **Parity** - No and **Baud** - 9600.

Enter "6" the following display will appear:

```
(Ctrl-Default) Char [8] Stop [1] Parity [N] Baud [9600]
```

Press the "C" key to toggle between 5, 6, 7, or 8 character bytes.

Press the "S" key to set the Stop Bit to 1 or 2.

Press the "P" key to set the Parity as None, Odd, or Even.

Press the "B" key to set the Baud Rate. Continue to press the "B" key until the desired rate appears, then press Enter.

6-3-8 7 : ADDITIONAL RAM IN 1024 BYTES

The amount of on board memory may be varied. The speed of file management is directly proportional to the amount of RAM being used.

The default setting is 320.

Enter "7" to change the amount of RAM. Enter the desired number of bytes of RAM to be used (0 = the minimum amount, 147,456 bytes) (320 = the maximum amount, 475,136 bytes). If more than 475,136 bytes of memory is required, an additional memory board must be installed.

6-3-9 8 : RS232 PROTOCOL PORT-B

The characteristics of RS-232 Port-B may be established within this parameter setting.

The default settings are; **Char** - 8, **Stop** - 1, **Parity** - No and **Baud** - 9600.

Enter "6", the following display will appear:

(Ctrl-Default) Char [8] Stop [1] Parity [N] Baud [9600]

Press the "C" key to toggle between 5, 6, 7, or 8 character bytes.

Press the "S" key to set the Stop Bit to 1 or 2.

Press the "P" key to set the Parity as None, Odd, or Even.

Press the "B" key to set the Baud Rate. Continue to press the "B" key until the desired rate appears, then press Enter.

6-3-10 9 : DEBUG DISPLAY IS AT FRONT PANEL

If the Unidex 21 being used is not equipped with a display screen, this parameter configures the system to send the Debug Display to an IBM PC/AT (equipped with the AEROTECH Display Software package).

The default setting is for the Debug Display located on the Unidex 21's front panel.

Enter "9" to change the Debug Display. Press the "Control" and "N" keys to toggle between Yes (Unidex 21's front panel will be used for the Debug Display) or No (Debug Display sent to a remote screen).

6-3-11 10 : RS232/IEEE488 TIME-OUT (SECONDS)

The Unidex 21 contains a Time-Out feature for file Input/Output through the RS232 or the IEEE488 ports. When RS232 or IEEE488 mode file transmission is initiated, the Unidex 21 will "look" for the data for a predetermined amount of time before producing an error message.

The default setting is 600 seconds.

Enter "10" to set the length of time the Unidex 21 waits for a return signal following an RS-232 or IEEE-488 transmission. Enter the new time (in seconds).

Refer to the *Unidex 21 Options Manual* for IEEE-488 information.

6-3-12 11 : PARTS PROGRAM STACK SIZE IN BYTES

This parameter is used to change the amount of RAM reserved for program stacks. (Refer to the *Unidex 21 Programming Manual* for additional information.) The maximum setting for this parameter is 1999 bytes.

The default setting is 300 bytes.

Enter "11" to change the amount of memory reserved for stacking. Enter the new number of bytes.

6-3-13 12 : EDIT BLOCK BUFFER (1 TO 40)

The amount of memory reserved for an Edit function buffer may be changed within this parameter. The Edit function buffer is used to store program blocks which are copied or erased while in the Edit mode.

The default setting is 8 blocks. Each block of program equals 256 bytes of memory.

Enter "12" to change the Edit buffer size. Enter the new buffer size (1 to 40 blocks).

6-3-14 13 : EDIT DEFAULT CHARACTER INSERT

While in the Edit mode, the User may insert characters in two ways. Characters may be inserted at the cursor, displacing existing text, or characters may be inserted such that existing text is over-written.

The default setting is that text will be inserted at the cursor, displacing existing text.

Enter "13" to change the Character Insert status. Press the "Control" and "N" keys to toggle between Yes (text inserted at cursor) or No (inserted text over-writes existing text).

6-3-15 14 : EDIT DEFAULT LINE INSERT

While in the Edit mode, a carriage return may be set to insert a new line in a program or to move to the next line of text.

The default setting is to insert a new line at each carriage return.

Enter "14" to change the Line Insert status. Press the "Control" and "N" keys to toggle between Yes (carriage return inserts new line) or No (carriage return moves to next line of text).

6-3-16 15 : EDIT TAB SPACE

While in the Edit mode, the number of spaces advanced by a TAB is set by this parameter.

The default setting is 10.

Enter "15" to change the TAB spacing. Enter the desired TAB spacing.

6-3-17 16 : END OF ALL FILE CODE CHR$(n)

When sending multiple files through an RS-232 or IEEE-488 port, a character is necessary to signal when data transmission has been completed. This parameter permits the User to change the End of All File character(s) for a particular system.

The default setting is 17.

Enter "16" to change the End of All File character(s). Enter the new End of File character(s).

Refer to the *Unidex 21 Option Manual* for IEEE-488 information.

6-3-18 17 : END OF FILE CODE CHR$(n)

When sending a file through an RS-232 or IEEE-488 port, it is necessary to indicate when file data transmission is complete. This parameter permits the User to change the End of File character(s) for a particular system.

The default setting is 9.

Enter "17" to change the End of File character. Enter the new End of File character(s).

Refer to the *Unidex 21 Options Manual* for IEEE-488 information.

6-3-19 18 : BEEPER DURATION (1 TO 280) ms

The length of time (ms) the Unidex 21's beeper stays On may be changed through this parameter.

The default setting is 50 ms.

Enter "18" to change the Beeper Duration. Enter the new time (in ms).

6-3-20 19 : DOUBLE SIDED FLOPPY DISK

Prior to formatting disks through the Unidex 21's Floppy Disk Drive, this parameter must be configured (single or double sided) the same as the floppy disk being formatted.

The default setting is Double Sided.

Enter "19" to change the Floppy Disk Configuration. Press the "Control" and "N" keys to toggle between Yes (format double-sided disks) or No (format single-sided disks).

6-3-21 20 : BEEPER FREQUENCY (2 TO 20K)

The sound frequency of the beeper is varied by this parameter. The frequency can be changed to between 2 and 20,000 hertz.

The default setting is 2000 hertz.

Enter "20" to change the Beeper Frequency. Enter the desired Beeper Frequency.

6-3-22 21 : DISPLAY BLANK-OUT (MINUTES)

The Unidex 21 is equipped with a "screen saving" technique which blanks-out the display screen if no entry has been made within a certain number of minutes. The time elapsing between the last key entry and screen Blank-Out is set by this parameter.

The default setting is 60 minutes.

Enter "21" to set the time to elapse (minutes) before screen Blank-Out. Enter the new time.

6-3-23 22 : MFO INC./STEP (-100 TO 100)

This parameter is used to change the increments/decrements of the MFO keys (+,- /F9, F10) or initialize and provide the zero offset for an external MFO adjustment.

Enter "22" to set the increments available through the MFO keys. Enter the desired MFO increment (-100 to 100).

If an external MFO adjustment is being used, this parameter must be set with a negative number. The negative sign indicates to the Unidex 21 that an external MFO adjustment is being used, the number entered provides the amount of zero offset. The amount of zero offset required is determined as follows:

Enter a negative number (-100 to 100) into this parameter.
Power down or reset the system to load the parameter.
Turn the MFO potentiometer fully CCW.
The MFO indication should go to approximately zero.

If the MFO% indication does not go to approximately zero, re-enter the Parameter mode and adjust this parameter setting as necessary. The default setting is 10.

6-3-24 23 : TRACKING DISPLAY PROGRAM STEP

While in the Machine mode, tracking may be displayed in Machine or Program steps.

The default setting is Program steps.

In the Debug mode, tracking is displayed in Program steps only.

Enter "23" to change the Tracking Display. Press the "Control" and "N" keys to toggle between Yes (tracking displayed in Program steps) or No (tracking displayed in Machine steps).

6-3-25 24 : Y PIXEL SIZE REDUCE TO (%)

Variance in monitors require that the pixel size of the Y axis be reduced for optimum viewing during the Debug Display mode.

The default setting is 100 %.

Enter "24" to change the Y pixel size. Enter the percentage of the desired reduction.

6-3-26 25 : PRINT SCREEN TO PORT-A

The User may choose to do a "Print Screen" through either RS232 Port-A or Port-B.

The default setting is Port-A.

> NOTE: **The "Print Screen" function is valid only in conjunction with the Unidex 21's Front Panel. (Key in "Alt P")**

Enter "25" to change the "Print Screen" RS232 port. Press the "Control" and "N" keys to toggle between Yes ("Print Screen" goes through RS232 Port-A) or No ("Print Screen" goes through RS232 Port-B).

6-3-27 26 : JOYSTICK AXIS PAIR

This parameter is used to set the axes configurations for Joystick or Trackball use.

The default settings are as shown in the display below.

Enter "26" to change the Joystick axes configurations.

The display will be:

```
                -horizontal-            -vertical-              - 3rd move-
                code XYZU/xyzu          code XYZU/xyzu          code XYZU/xyzu
1st
                00   10001000           01   01000100           02   00100010
                03   00100010           04   00010001           05   00000000
                06   00000000           07   00000000           08   00000000
                .........................  .........................  .........................
                09   00000000           10   00000000           11   00000000
                12   00000000           13   00000000           14   00000000
                15   00000000           16   00000000           17   00000000
                18   00000000           19   00000000           20   00000000
                21   00000000           22   00000000           23   00000000
                24   10001000           25   01000100           26   00100010
                27   00100010           28   00010001           29   00000000
                30   00000000           31   00000000           32   00000000
                .........................  .........................  .........................
                33   00000000           34   00000000           35   00000000
                36   00000000           37   00000000           38   00000000
                39   00000000           40   00000000           41   00000000
                42   00000000           43   00000000           44   00000000
                45   00000000           46   00000000           47   00000000

        >Ctrl-Quit, Ctrl-Defaut <cr>, code/nnnnnnnn <cr> =
```

Press the "Control" and "Q" keys to return to the Main Parameter Menu.

Press the "Control" and "D" keys to load default axes configurations.

To change the status of an axis, enter the code number, then enter a "0" to turn that axis OFF or a "1" to turn that axis ON.

6-3-28 27 : DIGITIZE WITH JOYSTICK

Prior to use, Joystick or Trackball digitizing must be delineated from this parameter.

The default setting is for Joystick digitizing.

Enter "27" to change the digitizing status. Press the "Control" and "N" keys to toggle between Yes (digitizing done by Joystick) or No (digitizing done by Trackball).

Refer to the *Unidex 21 Options Manual* for Joystick and Trackball information.

6-3-29 28 : INPUT 1 IS 0-CW/CCW, 1-CLK/DIR, 2-QUAD x 1, 3-QUAD x 2

The first auxiliary input must be configured so as to be in conjunction with the signal requirements of the input device.

Enter "28" to change the configuration of Input 1. Press the number associated with the desired configuration.

"0" for a CW/CCW input signal type
"1" for a CLK/DIR input signal type
"2" for a QUAD x 1 input signal type
"3" for a QUAD x 2 input signal type

The default setting is 0-CW/CCW input signal type.

6-3-30 29 : INPUT 2 IS 0-CW/CCW, 1-CLK/DIR, 2-QUAD x 1, 3-QUAD x 2

The second auxiliary input must be configured so as to be in conjunction with the signal requirements of the input device.

Enter "29" to change the configuration of Input 2. Press the number associated with the desired configuration.

"0" for a CW/CCW input signal type
"1" for a CLK/DIR input signal type
"2" for a QUAD x 1 input signal type
"3" for a QUAD x 2 input signal type

The default setting is 0-CW/CCW input signal type.

6-3-31 30 : AXES RAMP TIME (1-32767) ms

While contouring, acceleration and deceleration times may be optimized by adapting the time to reflect equipment limitations.

The default setting is 100 msec.

Enter "30" to change the Axes Ramping Time. Enter the new time (in msec).

6-3-32 31 : POWER ON REMOTE CONTROL 0/1/2/3/4

This parameter must be configured prior to putting the Unidex 21 under remote control. The default setting is for no remote control.

Enter "31" to change the remote power status.

Enter a "0" for no remote control.

Enter a "1" for Unidex 21 to be under RS-232 remote control following a Power-Up or Reset. There will be no Main Display on the Unidex 21 screen.

Enter a "2" for Unidex 21 to be under RS-232 remote control following a Power-Up or Reset. There will be a Main Display on the Unidex 21 screen.

Enter a "3" for Unidex 21 to be under IEEE-488 remote control following a Power-Up or Reset. There will be no Main Display on the Unidex 21 screen.

Enter a "4" for Unidex 21 to be under IEEE-488 remote control following a Power-Up or Reset. There will be a Main Display on the Unidex 21 screen.

6-3-33 32 : M STROBE DELAY (0-65535) ms

The M function Strobe Delay parameter provides the User the ability to change the length of time that the system output signal stays ON, allowing a slow device time to respond.

The default setting is for zero msec (no Strobe Delay).

Enter "32" to change the M function Strobe Delay time. Enter the new M Strobe Delay time (in msec).

6-3-34 33 : M ACK DELAY TO 65535 ms, 0 no

The **M** function Acknowledge Delay parameter provides the User the ability to establish a time that the Unidex 21 will wait for an acknowledgment signal from a slow device.

The default setting is zero (0) for no acknowledgment needed. Setting a **M** function Acknowledge Delay time of 65535 msec causes the Unidex 21 to scan indefinitely and check every 1msec for an acknowledge signal.

Enter "33" to change the **M** function Acknowledge Delay time. Enter the new **M** function Acknowledge Delay time (in msec).

6-3-35 34 : S STROBE DELAY (0-65535) ms

The **S** function Strobe Delay parameter provides the User the ability to change the length of time that the system output signal stays ON, allowing a slow device time to respond.

The default setting is zero (0) msec **S** function Strobe Delay time.

Enter "34" to change the **S** function Strobe Delay time. Enter the new **S** function Strobe Delay time (in msec).

6-3-36 35 : S ACK DELAY TO 65535 ms, 0 no

The **S** function Acknowledge Delay parameter provides the User the ability to establish a time that the Unidex 21 will wait for an acknowledgment signal from a slow device. The default setting is zero (0) for no acknowledgment needed. Setting a **S** function Acknowledge Delay time of 65535 msec causes the Unidex 21 to scan indefinitely and check every 1msec for an acknowledge signal.

Enter "35" to change the **S** function Acknowledge Delay time. Enter the new **S** function Acknowledge Delay time.

6-3-37 36 : T STROBE DELAY (0-65535) ms

The **T** function Strobe Delay parameter provides the User the ability to change the length of time that the system output signal stays ON, allowing a slow device time to respond. The default setting is for zero (0) msec **T** function Strobe Delay time.

Enter "36" to change the **T** function Strobe Delay time. Enter the new **T** function Strobe Delay time (in msec).

6-3-38 37 : T ACK DELAY TO 65535 ms, 0 no

The T function Acknowledge Delay parameter provides the User the ability to establish a time that the Unidex 21 will wait for an acknowledgment signal from a slow device. This time must include the time of the strobe signal.

The default setting is zero (0) for no acknowledgment needed. Setting a T function Acknowledge Delay time of 65535 msec causes the Unidex 21 to scan indefinitely and check every 1msec for an acknowledge signal.

Enter "37" to change the T function Acknowledge Delay time. Enter the new T function Acknowledge Delay time (in msec).

6-3-39 38 : QUICK STOP HI-LO TRIGGER

A Quick Stop is an interrupt signal to feedhold the Unidex 21 as quickly as possible. Quick Stop may be initiated by either a High/Low or a Low/High trigger signal. The interrupt signal is externally supplied, by the User, through an I/O connector on the rear panel of the Unidex 21.

The default setting is for a Quick Stop to occur at an input signal with a High to Low edge trigger.

NOTE: All User supplied Interrupt signals must be debounced.

Enter "38" to change the Quick Stop trigger signal. Press the "Control" and "N" keys to toggle between Yes (High/Low signal trigger) or No (Low/High signal trigger).

6-3-40 39 : QUICK STOP AT TRIGGER POINT

When a standard Quick Stop is initiated, deceleration occurs and axis movement halts at the conclusion of the deceleration ramp. If this parameter is activated (Yes) then the Unidex 21 moves the axis back to the point at which the trigger occurred, thus eliminating the ramp time distance.

The default setting is Yes.

Enter "39" to change the Quick Stop Trigger Point status. Press the "Control" and "N" keys to toggle between Yes (the axes will return to the point of the trigger) or No (the axes remain at the position following a deceleration ramp).

6-3-43 40 : IDX DOES CHECKSUM ?

This parameter configures the Indexing Board either to do a Checksum on each program block or to skip the Checksum feature. Skipping the Checksum operation provides faster processing time (3.6 +3.4N msec, N = no. of bytes in a block).

The default setting is for the Indexing Board to do a Checksum on each block.

Enter "40" to change the Indexing Board Checksum status. Press the "Control" and "N" keys to toggle between Yes (Checksum performed) and No (Checksum not performed).

6-3-44 41 : GANTRY (msmsmsms) m,s = 1,8

The Gantry parameter establishes the Master/Slave relationship of multiple axes motors being used in pairs. A maximum of four Master/Slave groups (8 axes) may exist.

Enter "41" to activate, deactivate, or change the Gantry Master/Slave status of one or more axes groups.

Enter the motor numbers to be paired as Master/Slave. For example, entering "18" will pair up the axis motors that have been designated 1 and 8. Motor 1 is designated as the Master and motor 8 is designated as the Slave. All programming for these axes must be input through axis motor 1.

NOTE: If the system is not being used in a Gantry configuration, all axis motor designations must be 0 (ms = 00).

6-3-43 42 : INPUT 1 HANDWHEEL SCALE (0-254)

If Input 1 is being used for the Handwheel option, the relationship between Handwheel increments and the number of Machine steps to be moved is established with this parameter.

The default is 10 Machine steps moved for each Handwheel increment.

Enter "42" to change the Handwheel increment/Machine step ratio of Input 1.

Enter the desired Scaling Factor (0-254). For example, If "10" is entered, the axes will move 10 Machine steps for each increment of the Handwheel.

Refer to the *Unidex 21 Options Manual* for Handwheel information.

6-3-44 43 : INPUT 2 HANDWHEEL SCALE (0-254)

If Input 2 is being used for the Handwheel option, the relationship between Handwheel increments and the number of Machine steps to be moved is established with this parameter.

The default is 10 Machine steps moved for each Handwheel increment.

Enter "43" to change the Handwheel increment/Machine step ratio of Input 2.

Enter the desired Scaling Factor (0-254). For example, If "10" is entered, the axes will move 10 Machine steps for each increment of the Handwheel.

Refer to the *Unidex 21 Options Manual* for Handwheel information.

6-3-45 44 : ROLL OVER MAX # 99999999

The Tracking Display is capable of indicating up to 99999999 Machine steps. If further Tracking Display is necessary, this parameter is used to instruct the system to automatically roll the display over to 00000000 and continue counting.

The default is "Yes" providing display roll over.

Enter "44" to activate/deactivate the Tracking Display roll over feature. Press the "Control" and "N" keys to toggle between Yes (Tracking Display will roll over) and No (Tracking Display will not roll over).

6-3-46 45 : H-V PAIRS (hvhvhvhv)h,v = 1,8

The H-V Pairs parameter identifies, to the system, the horizontal and vertical relationship of axes pairs. The pair identification is used for Perpendicularity Error Compensation. Eight axes may be identified for a total of four pairs.

The default setting is for no axis pairing, i.e., no Perpendicularity Error Compensation.

Enter "45" to identify horizontal and vertical axes pairs. Enter a horizontal axis number and a vertical axis number to be paired for Perpendicularity Error Compensation. For example, entering "13" will pair the horizontal axis number "1" with the vertical axis number "3". A total of four pairs may be entered.

NOTE: Perpendicularity Error Compensation will not be enabled until the axes are sent Home.

6-3-47 46 : 1 PERPENDICULAR ERROR ARC SEC

Positioning Accuracy is directly related to the perpendicularity of the paired axes. This parameter permits the User to enter into the system the amount of the error, in arc seconds, that has been calculated for the first axis pair (as established in Parameter 45: H-V Pairs). Perpendicularity Error is defined as the positive or negative deviation of the two paired stages from 90 degrees. An error factor in the CW direction is entered as a negative value. An error in the CCW direction is entered as a positive value.

The default setting is for zero (0) error.

Enter "46" to permit entry of the Perpendicularity Error for axis pair number 1. Enter the amount of error (in arc seconds).

6-3-48 47 : 2 PERPENDICULAR ERROR ARC SEC

Positioning Accuracy is directly related to the perpendicularity of the paired axes. This parameter permits the User to enter into the system the amount of the error, in arc seconds, that has been calculated for the second axis pair (as established in Parameter 45: H-V Pairs). Perpendicularity Error is defined as the positive or negative deviation of the two paired stages from 90 degrees. An error factor in the CW direction is entered as a negative value. An error in the CCW direction is entered as a positive value.

The default setting is for zero (0) error.

Enter "47" to permit entry of the Perpendicularity Error for axis pair number 2. Enter the amount of error (in arc seconds).

6-3-49 48 : 3 PERPENDICULAR ERROR ARC SEC

Positioning Accuracy is directly related to the perpendicularity of the paired axes. This parameter permits the User to enter into the system the amount of the error, in arc seconds, that has been calculated for the third axis pair (as established in Parameter 45: H-V Pairs). Perpendicularity Error is defined as the positive or negative deviation of the two paired stages from 90 degrees. An error factor in the CW direction is entered as a negative value. An error in the CCW direction is entered as a positive value.

The default setting is for zero (0) error.

Enter "48" to permit entry of the Perpendicularity Error for axis pair number 3. Enter the amount of error (in arc seconds).

6-3-50 49 : 4 PERPENDICULAR ERROR ARC SEC

Positioning Accuracy is directly related to the perpendicularity of the paired axes. This parameter permits the User to enter into the system the amount of the error, in arc seconds, that has been calculated for the fourth axis pair (as established in Parameter 45: H-V Pairs). Perpendicularity Error is defined as the positive or negative deviation of the two paired stages from 90 degrees. An error factor in the CW direction is entered as a negative value. An error in the CCW direction is entered as a positive value.

The default setting is for zero (0) error.

Enter "49" to permit entry of the Perpendicularity Error for axis pair number 4. Enter the amount of error (in arc seconds).

6-3-51 50 : RESET MALC MEMORY 0/1/2

The **MALC** command is used to allocate memory to be used for background functions. This parameter provides the User the ability to select the status of the Memory Allocation, following a Power-Down or Reset.

Enter "50" to change the Memory Allocation status.

The following options are available to set Memory Allocation status following a Power-up or Reset. Press the number of the desired status.

"0" Configures the system such that it will wait for the User to determine Memory Allocation status.

"1" Configures the system such that all previously allocated memory is cleared and returned to memory pool.

"2" Configures the system to retain the previously established amount of allocated memory.

6-3-52 51 : DEFAULT AT FRONT PANEL INTERFACE?

If the Unidex 21 Front Panel and a TeleVideo 905 terminal are being used in conjunction, this parameter may be used to designate which is to be considered the Main Interface.

The default setting is for the Unidex 21's Front Panel to be the Main Interface.

Enter "51" to change the Quick Stop Trigger Point status. Press the "Control" and "N" keys to toggle between Yes (the Unidex 21's Front Panel will be the Main Interface) or No (the TeleVideo 905 terminal will be the Main Interface).

6-3-53 52 : SYNC CODE 0?

If two Unidex 21's are being used together, each unit must be given a distinct code. This parameter permits the User to establish this code for Handshaking.

Although there is no operational difference between the Master and Salve controller, typically the Master is set to "Yes", and the Slave is set to "No".

Enter "52" to establish the SYNC code for this unit. Press the"Control" and "N" keys to toggle between Yes (the SYNC code for this unit is 0) or No (the SYNC code for this unit will be 1).

6-3-54 53 : IEEE-488 SETUP

The Unidex 21 may be controlled by a host computer through an IEEE-488 bus. Prior to IEEE-488 use, the settings of this parameter must be appropriately configured.

Enter "53" to establish IEEE-488 parameters. The display will be:

```
IEEE-488 SET UP

0: address mode (0 talk only) (1 listen only) (2 major only) (3 major/minor)
   (4 primary/secondary) (5 primary/primary) = 2
1: 1st address (0 to 31) = 2
2: 2nd address (0 to 31) = 3
3: PPR (0 no) (1 to 8 - in phase) (9 to 16 - reverse phase) = 1
4: EOS data (0 to FF) = 0A
5: EOS bits (0-7) (1-8) = 1
6: set EOI with last byte of write? (0 - yes) (1 - no) = 0
7: terminate read on EOS? (0 - yes) (1 - no) = 0
8: set EOI with last byte of write? (0 - yes) (1 - no) = 0

NOTE: EOS will not affect EOI during File mode Input/Output case
      Input  - Unidex 21 will wait for EOI or end-of-file code
      Output - Unidex 21 will set EOI with end-of-file code

Qtrl-Quit, Ctrl Default , code/nnnnnnnn =
```

Refer to the *Unidex 21 Options Manual* for a detailed explanation of the IEEE-488 Parameter settings.

6-3-55 54 : KEEP POSITION DURING RESET?

This parameter allows the User to configure the Unidex 21 such that following a System Reset, the axes position is retained on the Tracking Display.

The default setting is for the position information to be retained on the Tracking Display following a System Reset.

Enter "54" to change the Reset Tracking Display status. Press the "Control" and "N" keys to toggle between Yes (the Tracking Display will retain position information following a System Reset) or No (the Tracking Display will clear to zero following a System Reset).

6-3-56 55 : MFO ADJUST HANDWHEEL SCALE?

This parameter is used to link the MFO increment/decrement setting (Parameter 22) to the Handwheel scale factor (Parameters 42 and/or 43). If the link is established, the Handwheel scale factor will be adjusted by the MFO increment/decrement setting. This provides the User with the ability to change the Handwheel increment/Machine step relationship without returning to the Parameter mode.

The default setting is for the MFO setting to adjust the Handwheel scale factor.

Enter "55" to change the MFO/Handwheel scale status. Press the "Control" and "N" keys to toggle between Yes (Handwheel scale factor will be adjusted by the MFO setting) or No (Handwheel scale factor will not be adjusted by the MFO setting).

Refer to the *Unidex 21 Options Manual* for Handwheel information.

6-3-57 56 : AXES TRAP NEGATE OUTPUT (0-8)

When an Axis Trap occurs, the User may select an output line (0-8) in which the signal will go low. This signal may then be used by an auxiliary device to initiate some function.

The default setting is for no change in the output signal when an axis is in a trap condition.

Enter "56" to assign an output line to signal an Axis Trap condition.

Enter a "0" for no function to occur when an axis encounters a trap condition.

Enter a "1 thru 8" to assign an output line to go low when an axis encounters a fault (programming error, limit, trap, etc.) condition.

Enter a "1 thru 8" to assign an output line to go Low when an axis encounters a trap condition only.

6-3-58 57 : MASTER TRAJECTORY LINEAR?

For all contoured moves, each individual axis is synchronized with a "virtual" master axis. Therefore, the Acceleration/Deceleration Ramping Trajectory for each axis follows the Acceleration/Deceleration Ramping Trajectory for the Master Axis. This ramping trajectory may be defined as either Linear or Parabolic. A Parabolic Ramping Trajectory is derived by applying a "Parabolic Coefficient" (see Parameter #59) to the maximum slope. (See Chapter 5: Machine Mode for additional information.)

The default setting is for a Linear Type Trajectory.

Enter "57" to change the trajectory type. Press the "Control" and "N" keys to toggle between Yes (Linear Trajectory) and No (Parabolic Trajectory).

6-3-59 58 : FAST FEEDHOLD RAMP TIME (ms)

This parameter is used to specify the amount of time in which to decelerate all axes to a stop when the Quick Stop interrupt signal is detected. (Refer to parameters 38 and 39.)

Enter "58" to change deceleration time during a Quick Stop. Enter the new time (in msec).

6-3-60 59 : MASTER PARABOLIC COEFFICIENT

The Parabolic Coefficient is the factor used to convert the linear velocity/time slope into a Parabolic Trajectory. It is suggested that coefficient selections be made in steps of 10,000. The following formula is applied to the coefficient value selected.

$$\frac{\text{Parabolic Coefficient}}{65535} + 1 = \text{Maximum Slope}$$

A coefficient of zero represents a Linear Ramp. Higher coefficient values result in a steeper curve.

The default setting is 24576.

Enter "59" to change the Master Parabolic Coefficient. Enter the new number (0-65535).

6-3-61 60 : EXP FILTER LEVEL (0-7)

During G23 (Corner Rounding) operation, the Unidex 21 applies an Exponential Filter on the millisecond Position/Velocity Trajectory command being sent to the DSP Servo Control Card. This filter permits the Unidex 21's PIDF Servo Loop to digitally replicate the traditional error based type of Servo Loop Closure.

The digital filter works under the following relationships:

Filter Total = Filter Total + Command In

$$\text{Command Out} = \frac{\text{Filter Total}}{2^n} \quad \text{where n is the Filter Level}$$

Filter Total = Filter Total - Command Out

Therefore, higher filter values result in more Corner Rounding.

Enter "60" to change the G23 Exponential Filter Level. Enter the new number (0-7).

6-3-62 61 : RAMPING DURING G23 OPERATION?

The G23 (Corner Rounding) mode is used to permit the User to program moves which are not necessarily tangent, and have the controller provide relatively continuous Velocity Contouring. Some applications utilize this feature to smooth out step velocities resulting from non-tangent moves while contouring. However, in other applications such as Raster Scanning, it is desirable to permit the User to program a set of linear moves at right angles to each other, and have the controller round the corners. That is, as soon as the first axis begins its deceleration ramp, the second axis is to begin its acceleration ramp. This could be achieved by setting the Ramp Time (RAMP) command to one millisecond each time the User enters Corner Rounding (G23) mode, and restoring it when in Non-Corner Rounding (G24) mode. This implementation was deemed unnecessarily cumbersome.

If this parameter is set to "Yes", the normal Ramp Time will be in effect for both Corner Rounding (G23) mode and Non-Corner Rounding (G24) mode. If this parameter is set to "No", a Ramp Time of one (1) millisecond will be used while operating in Corner Rounding mode, and the normal Ramp Time parameter will be used in Non-Corner Rounding mode.

Enter "61" to change the ramping during G23 operation parameter. Press "Control" and "N" keys to toggle between Yes (G23 normal ramping) and No (G23 1ms ramping).

SECTION 6-4: AXIS PARAMETERS

A description of each of the Axis Parameters follows:

6-4-1 401 : 1ST AXIS

NOTE: Since the parameters are the same for 401 : 1st Axis through 408 : 8th Axis, one set will be described as being representative of the rest.

Enter "401" to configure the parameters for the 1st Axis. The display will be:

1st axis

0: Metric Constant	**1: English Constant**
2: Digits after "." (1 to 6)	**3: Digits after "." (1 to 6)**
4: Home direction is CCW ?	**5: Home Limit to Marker Steps** 0
6: Home Feedrate steps/sec	**7: Home Offset steps** 0
8: Top Feedrate steps/sec	**9: ac/de steps/sec/sec**
10: Symbol (1 to 4 char)	**11: Kp (0-16777215) =**
12: Ki (0-16777215) =	**13: Kd (0-16777215) =**
14: Kf1 (0-16777215) =	**15: Kf2 (0-16777215) =**
16: Jog control key (3,6,8,9)	**17: + move is CW ?**
18: Jog/Slew + direction is CW ?	**19: Feedhold enabled in Free Run ?**
20: Axis Calibration on ?	**21: Joystick abs. window**
22: Joystick high (steps/sec)	**23: Joystick low (steps/sec)**
24: Start/Stop speed (steps/sec)	**25: Modulo Machine step**
26: Power-on Home Feedrate (s/s)	**27: Home Switch is at end ?**
28: Home Switch is normal open ?	**29: Position Lag factor (0-65535)**
30: Limit Switch is normal open ?	**31: Switch-Mechanical limit steps**
32: Max Overshoot % (n/128)	**33: Max Velocity Error (0-65535)**
34: Max Position Error (0-65535)	**35: Max Integral Error n*100 0-65535**
36: MFO enabled in Free Run ?	**37: Trajectory is Linear type ?**
38: Parabolic Coefficient (0-65535)	**39: Machine Origin steps**

200: NEXT PAGE
>Ctrl-Quit, number <cr> to each parameter =

Enter "200" to go to the Next Page for more Axis Parameters.

The display will be:

```
1st axis

  40: Input 1 Auto-Focus convert factor (n/65535) 0 to 8388607
  41: Input 2 auto-focus convert factor (n/65535) 0 to 8388607
  42: Feedback Device?                    43: Driver Type?
  44: Brushless Commutation factor        45: Backlash Machine steps (0-65535)
  46: Existance Checking  0/1/2/3         47: CCW Software Limit, mach-steps
  48: CW Software Limit, mach-steps       49: Step Size for Step Response
  50:                                     51:
  52:                                     53:
  54:                                     55:
  56:                                     57:
  58:                                     59:
  60:                                     61:

200: PREVIOUS PAGE
  <Ctrl-Quit,number<cr> to each parameter =
```

A description of each of the Axis Parameters follows:

6-4-1-1 0 : METRIC CONSTANT

The Metric Constant parameter defines the Metric program (G71) mode's Machine step and Program step conversion factor. The number selected is incorporated into the following calculation:

$$\text{Machine steps} = \frac{\text{(this number) * (Program steps)}}{1000000}$$

The default setting is 1000000.

Press the "0" key to change the Metric value constant for this axis. Enter the new Metric value constant.

6-4-1-2 1 : ENGLISH CONSTANT

The English Constant parameter defines the English program (G70) mode's Machine step and Program step conversion factor. The number selected is incorporated into the following calculation:

$$\text{Machine steps} = \frac{\text{(this number) * (program steps)}}{1000000}$$

The default setting is 2540000.

Press the "1" key to change the English value constant for this axis. Enter the new English value constant.

6-4-1-3 2 : DIGITS AFTER "." (1 TO 6)

This parameter delineates the number of digits to be programmed after a decimal point in the Metric program (G71) mode.

The default setting is 3.

Press the "2" key to set the number of digits that will follow a decimal point. Enter the desired number of digits.

6-4-1-4 3 : DIGITS AFTER "." (1 TO 6)

This parameter delineates the number of digits to be programmed after a decimal point in the English program (G70) mode.

The default setting is 4.

Press the "3" key to set the number of digits that will follow a decimal point. Enter the desired number of digits.

6-4-1-5 4 : HOME DIRECTION IS CCW ?

The Hardware Home position may be established from either a CW or CCW motor direction.

The default setting is CCW.

Press the "4" key to establish motor direction for a Hardware Home. Press the "Control" and "N" keys to toggle between Yes (motor direction is counterclockwise) or No (motor direction is clockwise).

6-4-1-6 5 : HOME LIMIT TO MARKER STEPS

A distance (in Machine steps) that the axes may move at maximum speed before slowing to search for a Marker is established in this parameter. (See the *Unidex 21 Programming Manual* for a detailed description of the HOME command.) This distance is dependent upon system configuration and hardware variances.

The default setting is 1000 steps.

Press the "5" key to establish a distance from the Home Limit to the Marker. Enter the approximate distance in Machine steps.

6-4-1-7 6 : HOME FEEDRATE STEPS/SEC

The feedrate of a normal Home cycle is determined by the setting of this parameter. (See the *Unidex 21 Programming Manual* for a detailed description of the HOME command.)

The default setting is 2000 steps per second.

Press the "6" key to establish the feedrate of a Home cycle. Enter the new feedrate.

6-4-1-8 7 : HOME OFFSET STEPS

This parameter is used to establish offsets to provide for a Home position which is not at the Home Marker. (See the *Unidex 21 Programming Manual* for a detailed description of the HOME command.) The distance from the Home Marker to the desired Home Offset must be measured and then converted to steps.

The default setting is zero (the Home position is at the Home Marker).

Press the "7" key to establish the offset required for the desired Home position. Enter the offset (in steps).

6-4-1-9 8 : TOP FEEDRATE STEPS/SEC

The Top Feedrate is the highest speed for which this axis is mechanically configured. It will be used for all "G0" commands and as a basis for the Jog Feedrate.

The default setting is 128000 steps per second.

Press the "8" key to set the Top Feedrate. Enter the desired Top Feedrate (steps per second).

6-4-1-10 9 : AC/DE STEPS/SEC/SEC

This parameter is used to set the maximum acceleration/deceleration rate. The system uses this setting to verify all acceleration rates which are input for "G0" ramping.

The default setting is 1,000,000 steps per second per second.

Press the "9" key to establish the axis' maximum acceleration/deceleration rate. Enter the new rate (in steps/sec/sec).

6-4-1-11 10 : SYMBOL (1 TO 4 CHAR.)

This parameter permits the User to designate a new name for an axis. All system references will change to the new name, however, the system will continue to recognize the default name.

The default name of this axis is X.

Press the "10" key to change the axis name. Enter the new axis name (1 to 4 characters). Make certain name conflicts do not exist.

6-4-1-12 11 : Kp

The "P" (Proportional) gain parameter sets the "P" gain value within the PIDF loop. The "P" gain produces an output directly proportional to the Position Error and thus produces a constant counteracting force to the error.

The default setting is 102400.

Prior to changing the "Kp" value, see Section 6.5 "Axis Auto-Tune"for further information.

Press the "11" key to change the system's Proportional gain. Enter the new amount of "P" gain.

6-4-1-13 12 : Ki

The "I" (Integral) gain parameter sets the "I" gain value within the PIDF loop. The "I" gain produces an output which is a summation of the position errors, producing an increasing counter-acting force for a constant or increasing Position Error.

The default setting is 20.

Prior to changing the "Ki" value, see Section 6-5 (Axis Auto-Tune) for further information.

Press the "12" key to change the amount of Integral gain.

Enter the new amount of "I" gain.

6-4-1-14 13 : Kd

The "D" (Derivative) gain parameter sets the "D" gain in the PIDF loop. The "D" gain serves to dampen system response by producing a restraint, as long as the system is progressing toward error reduction.

The default setting is 896000.

Prior to changing the "Kd" value, see Section 6-5 (Axis Auto Tune) for further information.

Press the "13" key to change the amount of Derivative gain. Enter the new amount of "D" gain.

6-4-1-15 14 : Kf1

The "Kf1" (Velocity Feedforward) parameter optimizes the gain settings of the PIDF loop. The Velocity Feedforward value is a function of the System inertia and torque constant. Since it is not strictly error driven, it complements the PID gain by reducing the PID compensation effort required and by reducing system lag.

Prior to changing the "Kf1" value, see Sections 6-5 (Axis Auto-Tune) for further information.

Press the "14" key to change the "Kf1" parameter. Enter the new "Kf1" value.

6-4-1-16 15 : Kf2

The "Kf2" (Acceleration Feedforward) parameter optimizes the gain settings of the PIDF loop. The Acceleration Feedforward value is a function of the System inertia and torque constant. Since it is not strictly error driven, it complements the PID gain by reducing the PID compensation effort required and by reducing system lag.

Prior to changing the "Kf2" value, see Sections 6-5 (Axis Auto-Tune) for further information.

Press the "15" key to change the "Kf2" parameter. Enter the new "Kf2" value.

6-4-1-17 16 : JOG CONTROL KEY (3, 6, 8, 9)

The user may designate the arrow keys (3-7, 6-4, 8-2, 9-1) to be used to control this axis while in the Jog mode.

The default setting for this axis is 6.

Press the "16" key to change the arrow keys controlling this axis. Enter the key number of the desired arrow key. Press the "Control" and "G" keys to set axes 5-8.

6-4-1-18 17 : + MOVE IS CW ?

For a programmed + (positive) move, the motor may be set for CW or CCW direction.

The default setting is for CW motor direction for any programmed + move.

Press the "17" key to change the motor direction for a programmed + move. Press the "Control" and "N" keys to toggle between Yes (CW motor direction) or No (CCW motor direction).

6-4-1-19 18 : JOG/SLEW + DIRECTION IS CW ?

Motor direction may be set for CW or CCW movement when a + (positive) move is requested while in the Jog/Slew mode.

The default setting is for CW direction motor movement when a + move is requested in the Jog/Slew mode.

Press the "18" key to change the motor direction for a Jog/Slew + move. Press the "Control" and "N" keys to toggle between Yes (CW motor direction) or No (CCW motor direction).

6-4-1-20 19 : FEEDHOLD ENABLED IN FREE RUN

This parameter determines whether the Feedhold key may be used to control Free Run motion for this axis.

The default setting enables the Feedhold key for Free Run control.

Press the "19" key to change the status of the Feedhold key during a Free Run of this axis. Press the "Control" and "N" keys to toggle between Yes (Feedhold key active for a Free Run of this axis) or No (Feedhold key not active for a Free Run of this axis).

6-4-1-21 20 : AXIS CALIBRATION ON ?

If the Unidex 21 is equipped with the Axis Calibration option, this parameter must be configured appropriately.

The default setting is Yes, axis calibration ON for this axis.

Press the "20" key to change the status of Axis Calibration. Press the "Control" and "N" keys to toggle between Yes (Axis Calibration - ON) or No (Axis Calibration - OFF).

Refer to the *Unidex 21 Options Manual* for Axis Calibration information.

6-4-1-22 21 : JOYSTICK ABS. WINDOW

While in the Joystick Mode an "Absolute Window" may be created to provide an area of movement where extremely fine positioning capabilities exist. The number selected in this parameter represents the increment factor that will be used for the Joystick handle deflection/Position movement relationship.

NOTE: Setting this parameter to zero (0) for all axes will disable this feature.

The default setting is for 10 steps.

Press the "21" key to change the Joystick window size. Enter the new window size.

Refer to the *Unidex 21 Options Manual* for Joystick information.

6-4-1-23 22 : JOYSTICK HIGH (STEPS/SEC)

Resolution ratio's vary between axes. When a Joystick move is requested, the speed/distance ratio for each affected axis must be compatible. This parameter establishes the speed at which the axis will move when a Joystick move is requested at the "High" speed setting. To derive the Joystick "High" speed setting for this axis, proceed as follows:

Determine the distance (mm/in) that this axis will move per second at maximum speed.

Multiply this value by the axis resolution ratio (number of Machine steps that equal 1 (mm/in)).

The resulting number (steps/sec) is the Joystick-High value to be entered for this parameter.

The default setting is 50000 steps per second.

Press the "22" key to change the Joystick's "High" speed setting. Enter the new speed setting (in steps/sec).

Refer to the *Unidex 21 Options Manual* for additional Joystick information.

6-4-1-24 23 : JOYSTICK LOW (STEPS/SEC)

Resolution ratio's vary between axes. When a Joystick move is requested, the speed/distance ratio for each affected axis must be compatible. This parameter establishes the speed at which the axis will move when a Joystick move is requested at the "Low" speed setting. To derive the Joystick "Low" speed setting for this axis, proceed as follows:

Determine the distance (mm/in) that this axis will move per second at the low speed.

Multiply this value by the axis resolution ratio (number of Machine steps that equal 1 (mm/in)).

The resulting number (steps/sec) is the Joystick-Low value to be entered for this parameter.

The minimum setting for this parameter is 1000.

The default setting is 10000 steps per second.

Press the "23" key to change the Joystick's "Low" speed setting.

Enter the new speed setting (in steps/sec).

Refer to the *Unidex 21 Options Manual* for additional Joystick information.

6-4-1-25 24 : START/STOP SPEED (STEPS/SEC)

The Start/Stop Speed parameter establishes the maximum speed this axis may be started or stopped without initiating an error condition.

The default setting is 2000 steps/sec.

Press the "24" key to change the Start/Stop speed. Enter the new speed.

6-4-1-26 25 : MODULO MACHINE STEP

This parameter is used to establish the step number which will signal the Unidex 21 to return its axis position counters to zero and begin another count of steps, starting with 1. The default setting is 0 (no counter reset).

Press the "25" key to change the number of steps after which the Unidex 21's counters return to zero. Enter the new step number.

6-4-1-27 26 : POWER-ON HOME FEEDRATE (S/S)

This parameter provides the User the ability to set a Home Feedrate specifically for a Home move following a Power-Up. To avoid possible equipment damage, it is suggested that this Home Feedrate be considerably slower than the programmed Home Feedrate since axis disorientation is likely.

The default setting is 2000 steps per second.

Press the "26" key to change the Home feedrate following a Power-Up. Enter the new feedrate.

6-4-1-28 27 : HOME SWITCH IS AT END ?

This parameter setting delineates for the Unidex 21 whether the Home Limit Switch is located at the end of axis travel or is a selected position as that of a Home Limit Switch on a Rotary Table.

The default setting specifies the Home Limit Switch to be at the end of axis travel.

Press the "27" key to change the Home Limit Switch status. Press the "Control" and "N" keys to toggle between Yes (the Home Limit Switch is located at the end of axis travel) or No (the Home Limit Switch is a selected point).

6-4-1-29 28 : HOME SWITCH IS NORMAL OPEN ?

This parameter must be configured to reflect the type of Home Limit Switch being used.

The default is that the Home Limit Switch is a normally open type switch.

Press the "28" key to change the Home Limit Switch status. Press the "Control" and "N" keys to toggle between Yes (the Home Limit Switch is a normally open switch) or No (the Home Limit Switch is a normally closed switch).

6-4-1-30 29 : POSITION LAG FACTOR (0-65535)

(Description not available at this time.)

6-4-1-31 30 : LIMIT SWITCH IS NORMAL OPEN ?

This parameter must be configured to reflect the type of CW and CCW Limit Switches being used.

The default is that the Limit Switches are a normally open type switch.

Press the "30" key to change the CW and CCW Limit Switch status. Press the "Control" and "N" keys to toggle between Yes (the Limit Switches are a normally open switch) or No (the Limit Switches are a normally closed switch).

6-4-1-32 31 : SWITCH-MECHANICAL LIMIT STEPS

The User must enter the number of Machine steps separating the Limit Switch from the Mechanical Stop.

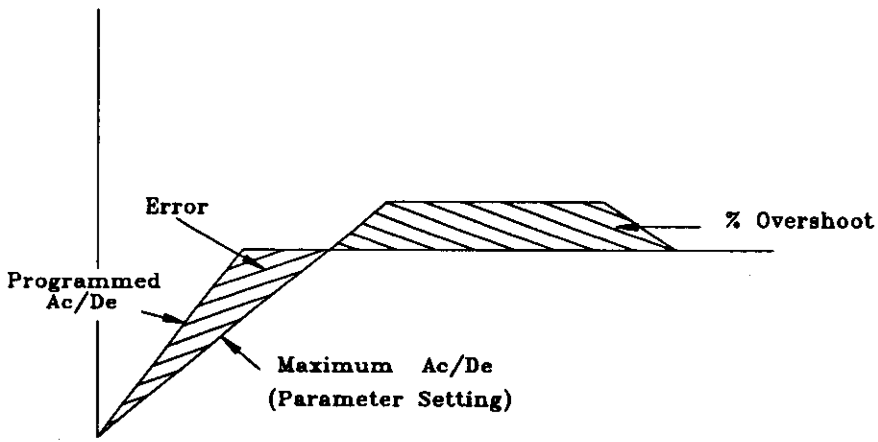
The default setting is 1000 Machine steps.

Press the "31" key to change the number of Machine steps designated as separating the Limit switches from the Mechanical stops. Enter the number of Machine steps.

6-4-1-33 32 : MAX OVERSHOOT % (n/128)

The maximum percentage of allowable Axis Overshoot must be entered for this parameter, 128 = 100%. The Unidex 21 uses this parameter to determine the amount of overshoot permissible when dissipating errors resulting from acceleration at a rate greater than the parameter setting of the Ac/De rate (see Item 6-4-1-10).

The default setting is 1.



Press the "32" key to change the maximum percentage of Axis Overshoot. Enter the percentage of the allowable overshoot.

6-4-1-34 33 : MAX VELOCITY ERROR (0-65535)

The maximum amount of allowable Velocity Error must be input at this parameter. The Unidex 21 presents an Error message if this parameter is exceeded during operation. A value of zero (0) will disable Maximum Velocity Error checking.

The default setting is 1000.

Press the "33" key to change the maximum allowable amount of Velocity Error. Enter the new amount (0-65535).

6-4-1-35 34 : MAXIMUM POSITION ERROR (0-65535)

The maximum amount of allowable Position Error must be input at this parameter. The Unidex 21 presents an Error message if this parameter is exceeded during operation. A value of zero (0) will disable Maximum Position Error checking.

The default setting is 1000.

Press the "34" key to change the maximum allowable amount of Position Error. Enter the new amount (0-65535).

6-4-1-36 35 : MAXIMUM INTEGRAL ERROR (0-65535)

The maximum amount of allowable Integral Error must be input at this parameter. The Unidex 21 presents an Error message if this parameter is exceeded during operation. The desired Integral Error value must be divided by 100 before it is entered. Example: If an Integral Error value of 20,000 is desired, enter 200 into this parameter. A value of zero (0) will disable maximum Integral Error checking.

The default setting is 1000.

Press the "35" key to change the maximum allowable amount of Integral Error. Enter the new amount (0-65535) ÷100.

6-4-1-37 36 : MFO ENABLED IN FREE RUN

This parameter permits the Manual Feedrate Override (MFO) function to be active in the Free Run mode.

The default setting is for MFO to be active during Free Run.

Press the "36" key to change the status of the MFO during Free Run. Press the "Control" and "N" keys to toggle between Yes (MFO enabled during Free Run) or No (MFO disabled during Free Run).

6-4-1-38 37 : TRAJECTORY IS LINEAR TYPE ?

Acceleration/ Deceleration (G0) Ramping Trajectory may be defined as either Linear or Parabolic. A Parabolic Ramping Trajectory is derived by applying a "Parabolic Coefficient" (see Parameter 38) to the maximum slope. (See Chapter 5: Machine Mode for more information.)

The default setting is for a Linear type trajectory.

Press the "37" key to change the trajectory type. Press "Control" and "N" keys to toggle between Yes (Linear Trajectory) or No (Parabolic Trajectory).

6-4-1-39 38 : PARABOLIC COEFFICIENT (0 - 65535)

The Parabolic Coefficient is the factor used to convert the linear velocity/time slope into a Parabolic Trajectory. It is suggested that coefficient selections may be made in steps of 10,000. The following formula is applied to the coefficient value selected:

$$\frac{\text{Parabolic Coefficient}}{65535}$$

The default setting is 24576.

A coefficient of zero (0) represents a Linear Ramp. Higher values result in a steeper curve.

Press the "38" key to change the Parabolic Coefficient. Enter the new number (0-65535).

6-4-1-40 39 : MACHINE ORIGIN STEPS

The Machine Origin parameter permits the User to establish a location, in Machine steps, to which the designated axes will return when the **MORG** command is used.

The default setting is zero.

Press the "39" key to enter the Machine Origin (in Machine steps) for this axis.

6-4-1-41 40 : INPUT 1 AUTO-FOCUS CONVERT FACTOR (n/65535) 0 TO 8388607

This parameter provides the User with the ability to establish the default value of the Conversion Factor used in conjunction with the **AFCO** command. This Conversion Factor is used to convert the Input Clock to Machine steps. The following formula applies:

$$\text{Machine steps} = \text{Input Clock} * \frac{n}{65535}$$

The default conversion factor is zero (0), indicating Input 1 is not used for Auto-Focus.

6-4-1-42 41 : INPUT 2 AUTO-FOCUS CONVERT FACTOR (n/65535) 0 TO 8388607

This parameter provides the User with the ability to establish the default value of the Conversion Factor used in conjunction with the **AFCO** command. This Conversion Factor is used to convert the Input Clock to Machine steps. The following formula applies:

$$\text{Machine steps} = \text{Input Clock} * \frac{n}{65535}$$

The default Conversion Factor is zero (0), indicating Input 2 is not used for Auto-Focus.

6-4-1-43 42 : FEEDBACK DEVICE?

This parameter configures the System for the type of position Feedback Device being used. The Unidex 21 may be configured to recognize any of the following Feedback Devices:

Enter a "0" if a Square Wave Encoder is providing position feedback.

Enter a "1" if a Resolver is providing position feedback.

Enter a "2" if the optional Resolution Multiplier Board with Sine Wave Encoder is providing position feedback.

Enter a "3" if an Open Loop Stepper Motor is being used.

The default setting is for a Square Wave Encoder as the Feedback Device.

6-4-1-44 43 : DRIVER TYPE?

This parameter configures the System for the type of Drive system being used. The Unidex 21 may be configured to recognize any of the following Driver types:

Enter a "0" if a DC Servo Brush Drive system is being used.

Enter a "1" if a AC Brushless Drive system is being used.

Enter a "2" if a Stepping Drive system is being used.

The default is for the use of a DC Servo Brush type Drive.

NOTE: **If a Brushless Drive System is established by this parameter, the next parameter, "Brushless Commutation Factor", must be configured appropriately.**

6-4-1-45 44 : BRUSHLESS COMMUTATION FACTOR

This parameter is valid only when a Brushless type motor is being used. The Brushless Commutation Factor is established as follows:

If the axis position feedback device is an Encoder, the Commutation Factor is the number of Machine steps per electric cycle (1 to 65535).

If the axis position feedback device is a Resolver, the Commutation Factor is the number of electric cycles per revolution (2, 3, 4).

The default setting is 2 (indicating a Brushless motor with Resolver position feedback).

Press the "44" key to change the Brushless motor Commutation Factor. Enter the new Commutation Factor.

6-4-1-46 45 : BACKLASH MACHINE STEPS, 0 TO 65535

To provide greater Positioning Accuracy, this parameter allows the User to enter a number of Machine steps (0 to 65535) to compensate for any backlash present in the system.

NOTE: **Backlash compensation will not be enabled unless the axis is sent Home.**

The default value is zero.

Press the "45" key to change the number of Machine steps that is necessary to compensate for System Backlash. Enter the compensation number (in Machine steps).

6-4-1-47 46 : EXISTENCE CHECKING 0/1/2/3

This parameter provides the User the ability to establish active axis status conditions that will be monitored by the Unidex 21 upon Power-Up or Reset. The use of this parameter gives added awareness to the User of the current configuration of the hardware.

The default setting is zero.

Press the "46" key to change the active status for this axis. The following options are available to set axis status:

0 - the status of this axis is irrelevant
1 - this axis **must** exist
2 - this axis **must not** exist
3 - this axis **always** exists regardless of status - No Auto-Tune procedure possible for this axis

Results of system monitoring are displayed in the Active Axis line of each display.

6-4-1-48 47 : CCW SOFTWARE LIMIT, MACHINE STEPS

This parameter permits the User to establish a CCW Software Travel Limit (in Machine steps) that is in reference to the Hardware Home.

The default setting is zero, indicating that a CCW Software Limit has not been established.

Press the "47" key to change the CCW Software Limit. Enter the new limit.

6-4-1-49 48 : CW SOFTWARE LIMIT, MACHINE STEPS

This parameter permits the User to establish a CW Software Travel Limit (in Machine steps) that is in reference to the Hardware Home.

The default setting is zero, indicating that a CW Software Limit has not been established.

Press the "48" key to change the CW Software Limit. Enter the new limit.

6-4-1-50 49 : STEP SIZE FOR STEP RESPONSE

This parameter permits the User to establish an optimum Step Size to produce a desired Step Response. Typically, the Step Size is 1/4 of the motor's resolution, i.e., a Step Size of 500 steps per revolution would be used for a 2000 line Encoder.

If AEROTECH, Inc. motors are being used in the system, the appropriate Step Size has been established as a default parameter setting (500 steps).

Press the "49" key to change the Step Size. Enter the new Step Size.

SECTION 6-5: AXES AUTO-TUNE

The Unidex 21 Auto-Tune function provides two tuning methods to optimize the Step Response output. The PIDF gains and the Step Size may either be individually adjusted for specific Step Response outputs, or the Unidex 21 may be requested to automatically adjust the PIDF gains. If the Unidex 21 is equipped with a Front Panel or EGA monitor, a graphic display of the Step Response and Velocity Curves is available to illustrate new or existing settings.

Enter "201" to change the PIDF gains or Step Size settings.

The display will be:

Axes	Parameter Setting					Auto/Manual Tune					Step
# Name	Kp	Ki	Kd	f1	f2	Kp	Ki	Kd	f1	f2	Size
1 X	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500
2 Y	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500
3 Z	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500
4 U	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500
5 x	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500
6 y	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500
7 z	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500
8 u	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00500

A1 to A8 : Auto tune, Ad/cm, cm is current command % (1-100) default 30
S1 to S8 : Save to Parameter
P1 to P8 : Positive step response with auto/manual tune gain
N1 to N8 : Negative step response
M1 to M8 : Manual input gain or step size, format --Md/item/data
d = 1 - 8, item = P,I,D,F1,F2,S data = 0 - 16777215

ctrl-Quit, select =

A description of each of the axes for changing the PIDF gains or Step Size settings follows:

6-5-1: AXES # NAME

Each axes is assigned an **Axes #** (number), this number is used to specify the axis for all display functions.

The axes **Name** is the name assigned in the Axis Parameter, Symbol Setting. The default settings are X,Y,Z,U,x,y,z, and u.

6-5-2: PARAMETER SETTINGS

The Parameter Setting values indicate settings established at the factory, by the User from the Axis Parameter mode, through previously saved Manual settings or through previously saved Auto-Tune settings. An explanation of each of the Parameter settings follows:

6-5-2-1: Kp

The Kp parameter sets the Proportional gain within the PIDF loop. The "P" gain produces an output directly proportional to the position error, thus producing a constant counteracting force to the error.

6-5-2-2: Ki

The Ki parameter sets the Integral gain in the PIDF loop. The "I" gain produces an output which is a summation of the position errors, producing an increasing counteracting force for a constant or increasing position error.

6-5-2-3: Kd

The Kd parameter sets the Derivative gain in the PIDF loop. The "D" gain serves to dampen system response by producing a restraint as the system progresses toward error reduction.

6-5-2-4: Kf1

The Kf1 parameter sets Velocity Feedforward in the PIDF loop. The Velocity Feedforward value is a function of the system friction and torque constant. Since it is not strictly error driven, it complements the PID gain by reducing the compensation effort required, thus reducing system lag.

6-5-2-5: Kf2

The Kf2 parameter sets the Acceleration Feedforward in the PIDF loop. The Acceleration Feedforward value is a function of the system inertia and torque constant. Since it is not strictly error driven, it complements the PID gain by reducing the compensation effort required, thus reducing system lag.

6-5-2-6: Step Size

The optimum Step Size setting is dependent upon system configurations. Typically, the Step Size is 1/4 of the motor revolution, i.e., the Step Size of 500 steps per revolution would be used for a 2000 line encoder.

6-5-3: AUTO/MANUAL TUNE

The Auto/Manual Tune function permits the User to establish new settings for the gain values listed in the Parameter Setting section of this display. These values may be set manually or automatically by using the following options:

WARNING: USE OF THE AUTO-TUNE PROCEDURE MAY CAUSE SUDDEN MECHANICAL MOVEMENT.

6-5-3-1: A1 TO A8: AUTO TUNE, Ad/cm IS CURRENT COMMAND % (1- 100) DEFAULT 30

Selecting **A1** through **A8** initiates the Auto-Tune procedure for the designated axis (1-8). The generated values appear on the display when Auto-Tune is complete.

NOTE: **The Auto-Tune procedure cannot take place on axes under gantry control (see the GANTRY Parameter setting). If an axis is selected that is under gantry control, an error message (Can't tune gantry axes! Ctrl-Quit, select =) will be displayed.**

NOTE: **The Auto-Tune procedure should not be done on the vertical (Z) axis.**

A graphic display of the Step Response and Velocity Curves produced from these values is available by depressing **P1** through **P8** for a Positive Step Response display or **N1** through **N8** for a Negative Step Response display. A typical positive Step Response and Velocity Curve graphic is shown in Figure 6-1. The Step Response procedure may be performed while in the graphic mode by depressing "**P (1-8)**" or "**N (1-8)**". Press the "**Q**" key to return to the main Auto-Tune display.

NOTE: **The graphic display retains all previous data. Newly generated curves are superimposed upon existing curves. The display may be cleared by depressing the "R" key.**

NOTE: **For optimum system performance, the Step Response display should exhibit some overshoot (approx. 25%). Systems that are overdamped (almost no indication of overshoot during Step Response) will have long settling times during Velocity Profiling.**

It is recommended that the Step Response procedure be repeated a minimum of four times for each axis prior to saving these values to the Parameter Settings. Following each performance of the Step Response, the graphic should be examined for consistency. If inconsistency in "Overshoot" and/or "Settling Time" is evident, the current command may be increased or decreased, in steps of 5%, to achieve stabilization. Following each current command change, the Step Response should again be run a minimum of four times, and checked for consistency.

To increase or decrease the current command, proceed as follows:

Press the "A" key, the axis number **(1-8)** "/" and the new current command percentage **(1-100)**. The default current command percentage is 30.

If the output displayed by the Step Response curve is satisfactory, press "**S (1-8)**" to save the Auto-Tune gain values. The saved values will replace the previously established gain values in all Parameter Settings.

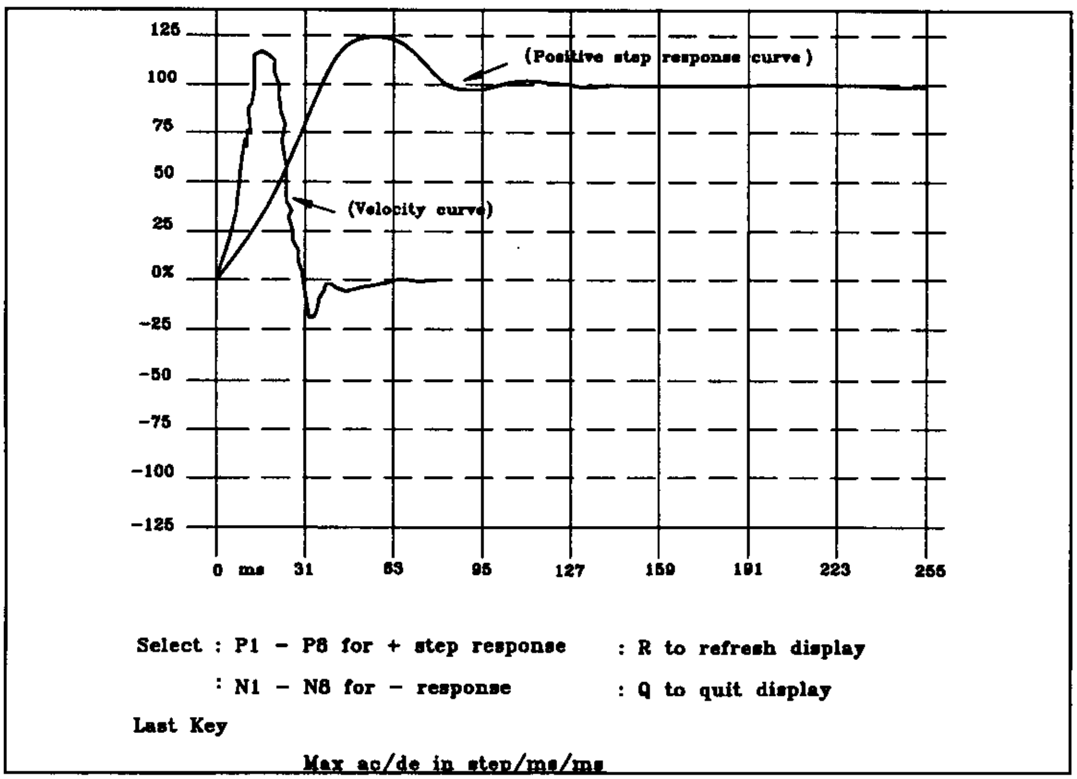


Figure 6-1: Typical Step Response and Velocity Curve Graphic

6-5-3-2: S1 TO S8 : SAVE TO PARAMETER

Selecting **S1** through **S8** saves the settings that have been established manually or by the Auto-Tune. Previous values in the Parameter Settings are replaced by the new values.

6-5-3-3: P1 TO P8 : POSITIVE STEP RESPONSE, WITH AUTO/MANUAL TUNE GAIN

Selecting **P1** through **P8** displays the graphic of the Positive Step Response for the Step Size and gain values that have been either automatically or manually derived.

6-5-3-4: N1 TO N8 : NEGATIVE STEP RESPONSE

Selecting **N1** through **N8** displays the graphic of the Negative Step/Response for the Step Size and gain values either automatically or manually derived.

6-5-3-5: M1 TO M8 : MANUAL INPUT GAIN OR STEP SIZE, FORMAT-- Md/item/data d = 1-8, item = P,I,D,F1,F2,S data = 0-16777215

Selections through this function provide the User with the ability to manually configure the gain values and/or Step Size to achieve specific Step Response reaction. See Figures 6-2 and 6-3 for a summary of the affects of gain variation.

NOTE: **The Velocity Feedforward Parameter Setting, should not, under most circumstances be manually set. Use the Unidex 21's Auto-Tuning function to select new Velocity Feedforward values.**

NOTE: **The Acceleration Feedforward Parameter Setting, should not, under most circumstances be manually set. Use the Unidex 21's Auto- Tuning function to select new Acceleration Feedforward values.**

6-5-3-5-1: MANUAL GAIN SETTING

To manually alter a gain value and/or Step Size, use the following format:

Md/item/data
d = Axes numbers 1-8
item = Parameter Settings, P,I,D,F1,F2, or Step Size, S
data = 0 -167772155

Press **P1** through **P8** for the Graphic display of the Positive Step/Response output.
Press **N1** through **N8** for the Graphic display of the Negative Step/Response output.

In addition to the Step Response curve, a Velocity Curve is shown. From this curve, the maximum acceleration value is derived by calculating the time required to reach the maximum velocity from zero. Use this value to establish "Acceleration/Deceleration steps/sec/sec" setting otherwise referred to as Axis Parameter #9.

> NOTE: **Prior to entering the "Max ac/de/in step/ms/ms" value into the Axis Parameter setting, msec must be changed to sec. (Multiply the Max ac/de/in step/ms/ms value by 1,000,000.)**

6-5-3-5-2: MANUAL STEP SIZE SETTING

Typically, the Step Size is 1/4 of the motor's resolution, i.e., a Step Size of 500 steps per revolution would be used for a 2000 line Encoder. The Step Size is not automatically generated by the Auto-Tune procedure. If AEROTECH, Inc. motors are being used in the system, the appropriate Step Size has been established as a default parameter setting (500 steps). If AEROTECH, Inc. motors are not being used , the Step Size may be established as follows:

Press "M1" through "M8", "/", "S", "/", then the desired Step Size.

Press **P1** through **P8** for the Graphic display of the Positive Step/Response output.

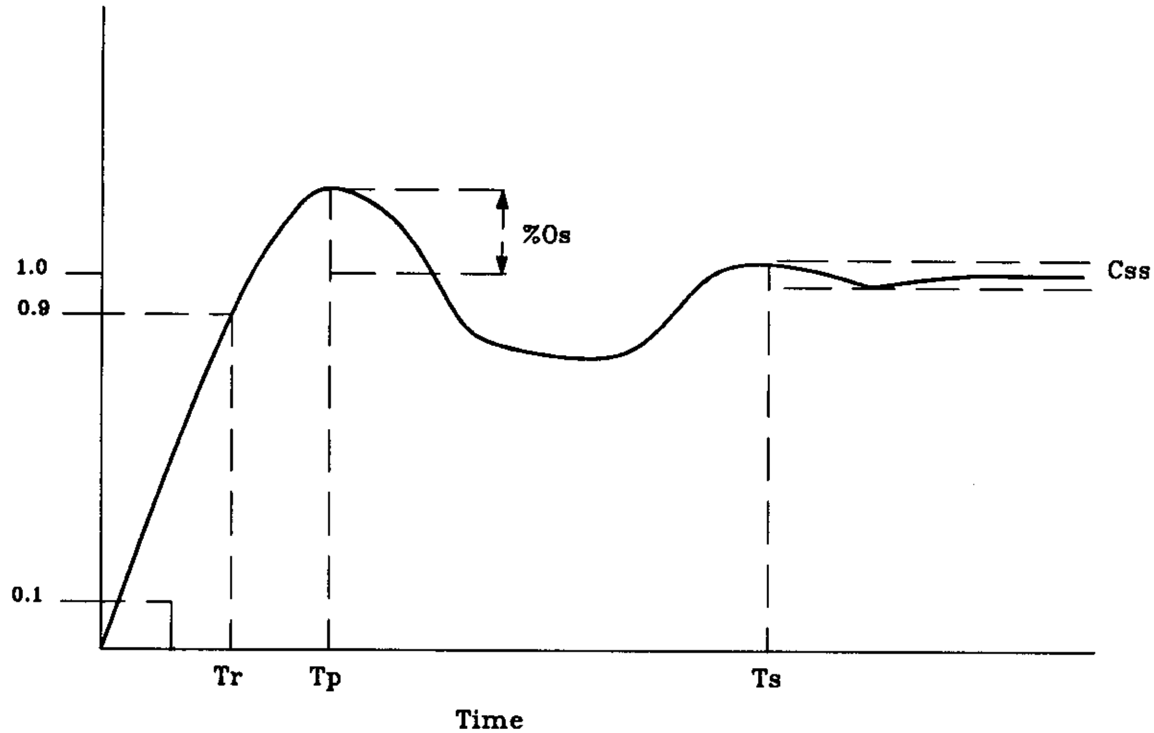
Press **N1** through **N8** for the Graphic display of the Negative Step/Response output.

> NOTE: **If a Step Size value has been selected that is too large, Response Overshoot will consistently occur regardless of gain variations.**

> NOTE **Selecting S1 through S8 saves the new Step Size as well as gain value settings. Upon Power Up or Reset, the previous values in the Parameter Settings are replaced by the new values.**

When the desired Step/Response output has been established, press **S (1-8)** to save these values.

Press the "Control" and "Q" keys to return to the main Parameter menu.



Tr	Rise Time of the Step Response is defined as the time that is required for the response to rise from 10 percent of the final value to 90 percent of the final value.
Tp	Peak Time is the time that the response takes to reach its peak value.
Ts	Settling Time is the time required for the output to stabilze to within 5 percent of its final value.
Css	Steady State is the final value of the Step Response.
%Os	Percent Overshoot is derived by the following formula:

$$\text{Percent Overshoot} = \frac{\text{Peak Value} - \text{Steady State Value}}{\text{Steady State Value}} \quad \text{X} \quad 100$$

Figure 6-2: Typical Step Response

PARAMETER	Kp Increase	Ki Increase	Kd Increase
Rise Time	Decrease	Decrease	Increase
Overshoot	Increase	Increase	Decrease
Settling Time	Increase	Increase	Decrease
Steady State Error	Decrease	Decrease	—
Damping Factor	Decrease	Decrease	Increase

Figure 6-3: Gain Adjustment Affects on System Response

6-6: FRONT PANEL FUNCTION KEYS

Function keys F1 through F8 may be configured to initiate various program functions.

Enter "301", the display will be:

Key	Mode	Title	bytes
F1			
F2			
F3			
F4			
F5			
F6			
F7			
F8			

Key definition buffer left (bytes) = xxx (total of 512 bytes)

Press the "Control" and "Q" keys to return to the Main Parameter Menu.

Press the number of the Function key to be configured.

The following display will appear:

Select	:	**Alt-Abort, Alt-Backspace, mode code/title/key-data <Alt-End>**
mode code	:	**all = 0, general =1, edit = 2, file = 3, machine = 4,**
		p-meter = 5, test = 6, system = 7, INT 1/2 = 8
title	:	**1 to 15 characters**
Key-data	:	**ASCII, Ctrl Keys are represented by πA to πZ.**
Status F-	:	**(mode code)/(title)/Key-data**
New data	:	

A description of the "on screen" Function key menu follows:

Select : Select provides a menu of options available in the Function key mode. The options are:

Alt-Abort	Press the "ALT" and "A" keys to cancel any changes that have been made and return to the main Function key display.
Alt-Backpace	Press the "ALT" and "B" keys to backspace and delete entered characters.
Alt-End	Press the "Alt" and "E" keys at the completion of data entry to save the new Function key configuration and return to the main Function key menu.

mode code : The mode code delineates the mode from which the Function key is to be active.

all =	0 - the Function key is active from any mode at any display.
general =	1 - the Function key is active from the Main Display Menu only.
edit =	2 - the Function key is active in the Edit mode only.
file =	3 - the Function key is active in the File mode only.
machine =	4 - the Function key is active in the Machine mode only.
p-meter =	5 - the Function key is active in the Parameter mode only.
system =	7 - the Function key is active in the System mode only.
INT 1/2 =	8 - the Function key is activated to initiate Interrupt 1 or 2.

Key in the appropriate mode code number followed by "/".

title : The title is the Function key label which will be displayed at all selection screens. The title (1 to 15 characters in length) may be the same as the program name.

Key in the desired title immediately following the "mode code/".

Key-data : The Key-data line contains an explanation of the ASCII code displayed during entry of the Key data. Key-data entry consists of the actual keystrokes necessary to initiate a function.

For example, DRFILENAME.TYP π A π M π D is the Key-data entry for Debug, Run, Filename.typ, Ctrl-Automatic, Enter, Ctrl-Display.

Status F - The Status line provides the User with the data to which the Function key is currently configured.

New data : The New data line is the location where the User reconfigures the Function key. Data must be entered as described in "Select".

When the data is entered in this manner, use of the Function key will initiate all entered data.

SECTION 6-7: LOAD/SAVE PARAMETER

Parameter settings may be saved to a file and reloaded as desired.

Enter "300" to Load or Save parameter settings.

The following screen will be displayed:

```
Ctrl-Quit; Load-from (filename.typ) ; Save-to (filename.typ)

Select =
```

Press the "Control" and "Q" keys to return to the Main Parameter Menu.

Key in "S", the Filename.type and then "carriage return" to save the current parameter display to a file. Any file saved through the Parameter mode receives an attribute setting of "Read only".

Key in "L," the Filename.type and then "carriage return" to load an existing parameter file into the current parameter display.

NOTE: Loading a Parameter file over- writes all existing Parameter settings.

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 7: TEST MODE

SECTION 7-1: INTRODUCTION

The Unidex 21 contains a self-test mode initiated through the main selection menu. It is not necessary to do a system Power Down or Reset prior to performing the test.

The Test mode provides the ability to test three storage locations, EPROM, RAM and the Parameters. In addition to the testing function this mode provides the ability to clear all User files from memory.

SECTION 7-2: GETTING STARTED

Following Power-Up the Initial Selection screen below will be displayed:

```
UNIDEX 21      Version xx


EPROM OK       PARAMETER OK       RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug
```

Enter the Test Mode by pressing the "T" key from the Initial Selection screen.

The following message will be displayed:

```
Ram, Eprom, Parameter, Ctrl-Clear memory, Ctrl-Quit
```

SECTION 7-3: RAM CHECK

Depress the "R" key to perform a RAM check. The RAM check is internally divided into two subsections:

1) System RAM
2) User RAM

The Read and Write capability is checked in both subsections of RAM.

The User RAM, in addition checking Read and Write, also checks the User's file checksum to verify file integrity.

If the both sections of the RAM check are verified, the following message will appear:

RAM OK ; Select function :

If a Read/Write error is detected one of the following is displayed:

RAM (0) fail @ (address location) = (fail data)
RAM (F) fail @ (address location) = (fail data)
RAM (5) fail @ (address location) = (fail data)
RAM (A)fail @ (address location) = (fail data)

If a checksum error is detected the following will be displayed:

RAM checksum error (filename.type)

SECTION 7-4: EPROM

Depress the "E" key to perform an EPROM test. The Unidex 21 performs a checksum on all data contained on EPROM.

If the check is verified the following screen will be displayed:

```
EPROM OK ; Select function :
```

If the EPROM check is not verified the following will be displayed:

```
EPROM checksum error ; Select function :
```

If an EPROM error is detected, contact a AEROTECH, Inc. Service Representative for assistance.

SECTION 7-5: PARAMETER

Press the "P" key to perform a checksum on the Unidex 21's parameters. Verification is performed between the last previously saved parameters and the currently active parameters.

If the Parameter check is verified the following screen will be displayed:

```
Parameter OK ; Select function :
```

If the Parameter check is not verified the following will be displayed:

Parameter error ; Select function :

SECTION 7-6: CTRL- CLEAR MEMORY

Press the "Control" and "C" keys to clear RAM memory.

SECTION 7-7: CTRL-QUIT

Press the "Contol" and "Quit" keys to return to the main selection menu.

CHAPTER 8: SYSTEM MODE

SECTION 8-1: INTRODUCTION

The Unidex 21 contains a System Mode initiated through the main selection menu.

SECTION 8-2: GETTING STARTED

Following Power-Up the Initial Selection screen will be displayed:

```
UNIDEX 21       Version xx


EPROM OK        PARAMETER OK        RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug
```

Enter the System Mode by pressing the "S" key from the Initial Selection screen.

The following message will be displayed:

```
Date, Time, Version, Standard, Password, Ctrl-Quit
> Select function:
```

SECTION 8-3 : DATE

Depress the "D" key to verify or change the System's date. The following will be displayed:

"Current Date" (dd-mmm-yyyy) Ctrl-Quit < = >

If the "Current Date" is to be changed, enter the desired date in the format illustrated below:

dd;	must be entered as digits 01 to 31	
mmm;	JAN	JUL
	FEB	AUG
	MAR	SEP
	APR	OCT
	MAY	NOV
	JUN	DEC
yyyy;	1990	

If the "Current Date" is not to be changed, press the "Control" and "Q" keys to return to the main System menu.

SECTION 8-4: TIME

Depress the "T" key to verify or change the System's Time setting, The following will be displayed:

"Current Time" (hh-mm-ss) Ctrl-Quit < = >

If the "Current Time" is to be changed, enter the desired time in the format illustrated below:

hh;	00 to 23
mm;	00 to 59
ss;	00 to 59

If the "Current Time" is not to be changed, press the "Control" and "Q" keys to return to the main System menu.

SECTION 8-5: VERSION

Depress the "V" key to display the Version number of the software installed in the Unidex 21.

The following will be displayed:

```
Version X.X ; Select function :
```

SECTION 8-6: STANDARD

Press the "S" key to display the Standard syntax being used by the system. The display below shows the system default standard:

```
>XYZU IJKP LCD OAB xyzu ijkp lcd oab NGHFMST ; Select function
```

XYZU -	represents axes 1-4
IJKP -	represents circular interpolation parameters of axes 1-4
LCD/OAB -	represent polar cordinate commands for circular interpolation of axes 1-4
xyzu -	represents axes 5-8
ijkp -	represents circular interpolation parameters of axes 5-8
lcd/oab -	represents polar coordinate commands for circular interpolation of axes 5-8
NGHFMST -	RS274-D Commands

SECTION 8-7: PASSWORD

Applicable passwords and their active functions may be established within the Parameter Mode (see Chapter 6 of this manual). Thereafter, upon Power-Up a Password will be required to operate the System. Any established password may be used, however, it may be that all modes are not available for all passwords.

The System Password function permits the User to initialize the system to another previously established password without enacting a complete Power Down.

Press the "P" key to initialize a different System password..
The following will be displayed:

```
> Select function
  Password:
```

Enter the Password and Reset the System. The System will now be active to the functions established in the Parameter Mode for this password.

CHAPTER 9: BATCH MODE

SECTION 9-1: INTRODUCTION

Batch files may be created in the Edit Mode to simplify and accelerate a frequently used series of events. Batch files typically contain one or more command functions that are to be executed one at a time (see Figure 9-1 for an example of Batch File creation).

The Unidex 21's Batch Mode function provides the User with the ability to process Batch files that have been previously created and reside in User memory.

SECTION 9-2: GETTING STARTED

Following Power-Up the Initial Selection screen will be displayed:

```
UNIDEX 21       Version xx


EPROM OK        PARAMETER OK        RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter. Test, System, Batch, Console, Debug
```

Enter the Batch Mode by pressing the "B" key from the Initial Selection screen. The following message will be displayed:

```
Batch function mode
> Ctrl-Quit, filename.type =
```

Press the "Control" and "Q" keys to return to the Initial Selection menu.

Enter the filename.type of the Batch file to be executed.

UNIDEX 21 BATCH FILE EXAMPLE

A Batch File is created in the Edit Mode in the same way as a standard program file, with the following exceptions:

1) The "Control" character is represented by the " ^ " key.

2) Any function requiring an "Enter" or <CR> must be followed by a " ^ M".

EXAMPLE FILE:

M	; from the Power-Up Initial selection screen, entry is made into the Machine Mode. (No <CR> required)
M	; from the Machine Mode entry is made into Mdi. (No <CR> required)
(REF,U) ^M	; U axis is sent Home (<CR> required to activate)
G11 F100. U1. ^M	; U axis moves 1 (in./mm) with Linear contouring at a Feedrate of 100 steps/sec. (<CR> required to activate)
^Q	; Quits Mdi Mode (No <CR> required)
^Q	; Quits Machine Mode (No <CR> required)
F	; from the Power-Up Initial selection screen, entry is made into the File Mode. (No <CR> required)

Figure 9-1: Batch File Example

CHAPTER 10: CONSOLE MODE

SECTION 10-1: INTRODUCTION

The Console mode provides the User the ability to establish monitor characteristics for the Unidex 21's console or front panel display.

SECTION 10-2: GETTING STARTED

Following Power-Up the Initial Selection screen will be displayed:

```
UNIDEX 21        Version xx


EPROM OK       PARAMETER OK        RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug
```

Enter the Console Mode by pressing the "C" key from the Initial Selection screen.

The following will be displayed:

```
Last key in =

Alt - Activate console #1  Alt - Backround color  Alt - Quit console mode
Alt - Interface PLC  Unidex 21  Terminal  Monitor  Alt - Through Port A  B
```

(THIS PAGE INTENTIONALLY LEFT BLANK)

CHAPTER 11: THE DEBUG MODE

SECTION 11-1: INTRODUCTION

The Debug Mode provides the User with the ability to simulate program runs, to aide in the elimination of programming errors.

All applicable Machine Mode functions are duplicated in the Debug mode and are accessed in the same manner asdetailed in the Machine Mode. A Display function is added to provide the User with a visual verification of the program prior to entering the Machine Mode. Make certain that all appropriate Parameters have been properly established. (See Chapter 6: The Parameter Mode)

The following sections provide information concerning the Setup andDisplay functions of the Debug Mode. For information regarding any other Debug Mode functions, refer to Chapter 5: Machine Mode.

SECTION 11-2: GETTING STARTED

Following Power-Up the initial selection screen is displayed:

```
UNIDEX 21      Version  xx


EPROM  OK      PARAMETER OK      RAM  checksum
User's RAM  (bytes)  = xxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug
```

Depress the "D" key to enter the Debug Mode. The following screen will be displayed:

```
                 X      0.000        x      0.000
                 Y      0.000        y      0.000
                 Z      0.000        z      0.000
                 U      0.000        u      0.000


   H1 H11 H17 G1 G11 ...... (system codes)        Feedrate =  0.00     0.00

   ..........................................(program)


  Ctrl (Auto/Single [S]) (C-start) (Error ack) (Feedhold) (Track) (Display) (Quit)

   > Home, Jog, Mdi, Run, Setup,  Ctrl- ^ Abort        select :
```

SECTION 11-3: DEBUG MODE SCREEN DESCRIPTION

See Chapter 5: Machine Mode for an explanation of all of the Debug functions except "Display" and "Setup".

> NOTE: **The Tracking display in the Debug Mode indicates Program steps only.**

11-3-1: DISPLAY

The Display function provides the User the ability to change the screen from the main Debug display, to a simulated result of the program run.

To use the Display screen to debug a program, proceed as follows:

Enter the Filename.type of the program to be run. If the "Single/Auto" function is set to "A", the program will soon begin to run. If the "Single/Auto" function is set to "S", only the first block of the program will run. Subsequent blocks must be run by pressing the "Control" and "C" keys, the Numeric "+" or the CYCLE START key of the Front Panel. If the program was initially set to run Single and is switched to Auto the "Control" and "C" keys and or the Front Panel CYCLE START key must be pressed to start program run.

Press the "Control" and "D" keys to change the screen to the program display screen. The following will be displayed:

```
                                       OUTPUT :  7 6 5 4 3 2 1 0
                                                 x x x x x x x x
                                      AXIS POSITION :

                                      program        machine
                                      +00000.000     +00000.000
                                      +00000.000     +00000.000
 Y         (Program display)          +00000.000     +00000.000
                                      +00000.000     +00000.000
                                      +00000.000     +00000.000
                                      +00000.000     +00000.000
                                      +00000.000     +00000.000
                                      +00000.000     +00000.000

                                      Select functions:

                                      Ctrl-Q : Quit
                                      Ctrl-W : Update position
               X 25                   Ctrl-X : Disable pen
                                      Ctrl-Y : Dash line
                                      Ctrl-Z : Refresh display
```

11-3-1-1: (Program Display)

This is the area used to provide a program simulation. The axes names, as defined in Debug Setup (See next Section) are labeled. In addition to the axis name, the "Display Window" size is also indicated (established in Debug Setup).

11-3-1-2: OUTPUT

The Unidex 21 contains eight Output pins (addressed by system variables; $OTP0 through $OTP7). This display is included to indicate to the User which Output Pins are active.

An "X" indicates a non active output, a "1" indicates an active high output, a 0 indicates an active low output.

11-3-1-3: AXIS POSITION

The Axis Position indication of the Debug Display supply the User with a numerical axis position in conjunction with the simulated program run. The axes position is displayed in both program and machine steps.

11-3-1-4: CTRL-Q

Press the "Control" and "Q" keys to quit the Display mode and return to the main Debug screen.

11-3-1-5: CTRL-W

Press the "Control" and "W" keys to activate or deactivate the Axis position display.

11-3-1-6: CTRL-X

Press the "Control" and "X" key to Disable or Enable the pen. If a simulation of the actual workpiece is desired the pen should be disabled at times when the tool would not be touching the workpiece. If axis movement is to be simulated, the pen should be enabled for entire program run.

11-3-1-7: CTRL-Y

Press the "Control" and "Y" keys to toggle between solid and dash lines. It may be desirable to specify different line types when a program requires overlapping lines such as in program comparisons.

11-3-1-8: CTRL-Z:

Press the "Control" and "Z" keys to provide a clean program display screen.

11-3-2: SETUP

From the main Debug screen press the "S" key to establish the Debug Display parameters. The following will be displayed:

```
Ctrl - (First ) (Second) (Reference) (Window) (Down) (up) (Abort) (Quit)

Setup >
   Plane = ( X,Y) Reference = (1/2/3/4) Window (XX)
   Down = (XXXX = 1/0)   Up = (XXXX = 1/0)
```

11-3-2-1: CTRL-FIRST

Press the "Control" and "F" keys to change the first axis designation in the Debug Display's Plane setting. (Axis symbols are established in the Axis Parameter Mode, Chapter 6).

11-3-2-2: CTRL-SECOND

Press the "Control" and "S" keys to change the second axis designation in the Debug Display's Plane setting. (Axis symbols are established in the Axis Parameter Mode, Chapter 6).

11-3-2-3: CTRL-REFERENCE

Press the "Control" and "R" keys to change the Home Reference location. The following selections are possible:

1 = display of the program will be referenced from the lower left corner of the display

2 = display of the program will be referenced from the lower right corner of the display

3 = display of the program will be referenced from the upper right corner of the display

4 = display of the program will be referenced from the upper left corner of the display

11-3-2-4: CTRL-WINDOW

Press the "Control" and "W" keys to establish the window size in the Debug Display. The window size may be in either Metric or English and is referenced to workpiece size. For example, if a window size of 25 inches is entered, the Debug Display will show the segment of the workpiece which is encompassed within a 25 inch window. Additional segments may be viewed by changing the reference point as explained above.

11-3-2-5: CTRL-DOWN

Press the "Control" and "D" keys to specify a function for "pen down" activate the Ouputs (1-8).

For example if; DOWN = ($OT7 = 1); each time ($OT7 = 1) is decoded the Debug Display will respond with a pen down.

11-3-2-6: CTRL-UP

Press the "Control" and "U" keys to specify the output function which will activate a Debug Display pen up.

For example if; UP = (M90 = 0); each time (M90 = 0) is decoded the Debug Display will respond with a pen up.

NOTE: **Both the pen down and pen up functions may be activated or deactivated by the Ctrl-X keys**

11-3-2-7: CTRL- QUIT

Press the "Control" and "Q" keys to return to the previous selection screen.

SERVICE AND REPAIR

Customer repair of the equipment is limited. Control Board(s) may be removed and replaced if necessary, however, component level repair must not be attempted.

On-site service should be performed by an experienced electronic technician, preferably one trained by Aerotech.

SHIPMENT

The procedure for shipping equipment to Aerotech, described below, pertains to warranty as well as non-warranty repairs.

1. Before returning any equipment a "Return Authorization Number" must be obtained from Aerotech. (Be prepared to give the serial number of the equipment being returned.)

2. The equipment being returned must be encased in a proper cushioning material and enclosed in a cardboard box.

Call for a "Return Authorization Number" if it is necessary to ship any equipment to the factory.

WARNING: DAMAGE TO THE EQUIPMENT DUE TO IMPROPER PACKAGING MAY VOID WARRANTY!

Aerotech Sales and Service offices are listed below. For service and information, contact the office servicing your area.

AEROTECH, INC. SALES OFFICES

WORLD HEADQUARTERS
AEROTECH ,INC.
101 Zeta Drive
Pittsburgh, Pa. 15238

Phone (412) 963-7470
FAX (412) 963-7459
TWX (710) 795-3125

AEROTECH, LTD.
Aldermaston
Berkshire RG7 4QW, England

Phone (07356) 77274
TLX 847228
FAX (07356) 5022

AEROTECH GMBH
Neumeyerstrasse 90
8500 Nuernberg 10
West Germany

Phone (0911) 521031
TLX 622474
FAX (0911) 521235

Warranty and Field Service Policy

Aerotech, Inc. warrants its products to be free from defects caused by faulty materials or poor workmanship for a minimum period of one year from date of shipment from Aerotech. Aerotech's liability is limited to replacing, repairing or issuing credit, at its option, for any products which are returned by the original purchaser during the warranty period. Aerotech makes no warranty that its products are fit for the use or purpose to which they may be put by the buyer, whether or not such use or purpose has been disclosed to Aerotech in specifications or drawings previously or subsequently provided, or whether or not Aerotech's products are specifically designed and/or manufactured for buyer's use or purpose. Aerotech's liability on any claim for loss or damage arising out of the sale, resale or use of any of its products shall in no event exceed the selling price of the unit.

Laser Product Warranty

Aerotech, Inc. warrants its laser products to the original purchaser for a minimum period of one year from date of shipment. This warranty covers defects in workmanship and material and is voided for all laser power supplies, plasma tubes and laser systems subject to electrical or physical abuse, tampering (such as opening the housing or removal of the serial tag) or improper operation as determined by Aerotech. This warranty is also voided for failure to comply with Aerotech's return procedures.

Return Products Procedure

Claims for shipment damage (evident or concealed) must be filed with the carrier by the buyer. Aerotech must be notified within (30) days of shipment of incorrect materials. No product may be returned, whether in warranty or out of warranty, without first obtaining approval from Aerotech. No credit will be given nor repairs made for products returned without such approval. Any returned product(s) must be accompanied by a return authorization number. The return authorization number may be obtained by calling an Aerotech service center. Products must be returned, prepaid, to an Aerotech service center (no C.O.D. or Collect Freight accepted). The status of any product returned later than (30) days after the issuance of a return authorization number will be subject to review.

Returned Product Warranty Determination

After Aerotech's examination, warranty or out-of-warranty status will be determined. If upon Aerotech's examination a warrantied defect exists, then the product(s) will be repaired at no charge and shipped, prepaid, back to the buyer. If the buyer desires an air freight return, the product(s) will be shipped collect. Warranty repairs do not extend the original warranty period.

Returned Product Non-Warranty Determination

After Aerotech's examination, the buyer shall be notified of the repair cost. At such time the buyer must issue a valid purchase order to cover the cost of the repair and freight, or authorize the product(s) to be shipped back as is, at the buyer's expense. Failure to obtain a purchase order number or approval within (30) days of notification will result in the product(s) being returned as is, at the buyer's expense. Repair work is warranted for (90) days from date of shipment. Replacement components are warranted for one year from date of shipment.

Rush Service

At times, the buyer may desire to expedite a repair. Regardless of warranty or out-of-warranty status, the buyer must issue a valid purchase order to cover the added rush service cost. Rush service is subject to Aerotech's approval.

On-Site Warranty Repair

If an Aerotech product cannot be made functional by telephone assistance or by sending and having the customer install replacement parts, and cannot be returned to the Aerotech service center for repair, and if Aerotech determines the problem could be warranty- related, then the following policy applies.

Aerotech will provide an on-site field service representative in a reasonable amount of time, provided that the customer issues a valid purchase order to Aerotech covering all transportation and subsistence costs. For warranty field repairs, the customer will not be charged for the cost of labor and material. If service is rendered at times other than normal work periods, then special service rates apply.

If during the on-site repair it is determined the problem is not warranty related, then the terms and conditions stated in the following "On-Site Non-Warranty Repair" section apply.

On-Site Non-Warranty Repair

If an Aerotech product cannot be made functional by telephone assistance or purchased replacement parts, and cannot be returned to the Aerotech service center for repair, then the following field service policy applies.

Aerotech will provide an on-site field service representative in a reasonable amount of time, provided that the customer issues a valid purchase order to Aerotech covering all transportation and subsistence costs and the prevailing labor cost, including travel time, necessary to complete the repair.

AEROTECH, Inc., 101 Zeta Drive, Pittsburgh, Pennsylvania 15238

Phone (412) 963-7470 • TWX 710-795-3125 • FAX (412) 963-7459

ADDENDUM 1: REMOTE OPERATION OF THE UNIDEX 21 VIA A HOST CONTROLLER

All Models of the Unidex 21 Controller may be remotely operated by a Host Controller. The communication interface between the Host Controller and the Unidex 21 is the Unidex 21's RS-232 Port A, hence, any Controller capable of RS- 232 interface may be used.

SECTION 1A-1: INITIALIZATION

Prior to the initial operation of a Unidex 21 through a Host Controller, the Unidex 21 must be configured for Remote Control. This may be accomplished in three ways:

1A-1-1: PARAMETER SETTING

Following Power-Up, the initial selection screen is displayed:

```
UNIDEX 21      Version xx

EPROM OK       PARAMETER OK       RAM checksum
User's RAM (bytes) = xxxxxxx

Edit, File, Machine, Parameter,Test, System, Batch, Console, Debug
```

Press the "P" key to enter the Parameter mode.

The following screen will be displayed:

```
0 : System password                    1 : Skip auto-boot function ?
2 : IDX buffer 1 block only ?          3 : IDX seg. calculate base (1/2/3)
4 : COMM input feedback ?              5 : System default at metric ?
6 : RS232 protocol port-A              7 : Additional RAM in 1024 bytes
8 : RS232 protocol port-B              9 : Debug display is at front panel ?
10 : RS232/IEEE488 time out (sec)      11 : Parts program stack size in bytes
12 : Edit block buffer (1 to 40)       13 : Edit default Char-insert ?
14 : Edit default Line-insert ?        15 : Edit TAB space
16 : End of all file code CHR$(n)      17 : End of file code CHR$(n)
18 : Beeper duration (1 to 280) ms     19 : Double side floppy disk ?
20 : Beeper frequency  (2 to 20K)      21 : Display blank-out (minutes)
22 : MFO inc./step (-100 to 100)       23 : Tracking display program step ?
24 : Y pixel size reduce to (%)        25 : Print screen to port-A ?
26 : Joystick axis pair                27 : Digitize with joystick ?

200 : NEXT PAGE                        201 : Axes auto-tune

300 : Load/save parameter              301 : Front panel function keys
401 : 1st axis       402 : 2nd axis    403 : 3rd axis       404 : 4th axis
405 : 5th axis       406 : 6th axis    407 : 7th axis       408 : 8th axis

ctrl-Quit, number <cr> to each parameter =
```

Enter "200" to go to the Next Page.

The display will be:

```
28 : Input 1 is 0-CW/CCW, 1-CLK/DIR, 2-QUAD x 1, 3-QUAD x 2
29 : Input 2 is 0-CW/CCW, 1-CLK/DIR, 2-QUAD x 1, 3 -QUAD x 2
30 : Axis ramp time (1-32767) ms         31 : Power on remote control 0/1/2/3/4
32 : M strobe delay (0-65535) ms        33 : M ack delay to 65535 ms, 0 no
34 : S strobe delay (0-65535) ms        35 : S ack delay to 65535 ms, 0 no
36 : T strobe delay to (0-65535)        37 : T ack delay to 65535 ms, 0 no
38 : Quick stop Hi-Lo trigger ?         39 : Quick stop at trigger point ?
40 : IDX does checksum ?                41 : GANTRY (msmsmsms) m,s = 1,8
42 : Input 1 handwheel scale 0-254      43 : Input 2 handwheel scale 0-254
44 : Roll over max # 99999999           45 : H-V pairs (hvhvhvhv)h,v=1,8
46 : 1 perpendicular error arc sec      47 : 2 perpendicular error arc sec
48 : 3 perpendicular error arc sec      49 : 4 perpendicular error arc sec
50 : Reset MALC memory 0/1/2            51 : Default at Front Panel Interface?
52 : SYNC code 0?                       53 : IEEE488 Setup
54 : Keep position during reset?        55 : MFO adjust Handwheel scale?
56 : Axis trap negate Output (0-8)      57 :
58 :                                    59 :


200 : PREVIOUS PAGE
> Ctrl-Quit, number <cr> to each parameter =
```

Enter "31" to change the Unidex 21's Remote Control status.

Enter a "0" for no Remote Control.

Enter a "1" for the Unidex 21 to be under RS-232 Remote Control following a Power Up or Reset. The Unidex 21 display is not active and will not be updated during Remote operation.

Enter a "2" for the Unidex 21 to be underRS-232 Remote Control following a Power Up or Reset. The Unidex 21 display will be active and will be updated during Remote operation.

Enter a "3" for Unidex 21 to be under IEEE-488 Remote Control following a Power Up or Reset. The Unidex 21 display is not active and will not be updated during Remote operation.

Enter a "4" for Unidex 21 to be under IEEE-488 Remote Control following a Power Up or Reset. The Unidex 21 display is active and will be updated during Remote operation.

A Parameter setting of "1", "2", "3" or "4" will cause the Unidex 21 to automatically go into Remote Control at the next Power Up or Reset.

1A-1-2: INITIALIZATION THROUGH HOST CONTROLLER

Remote Operation may also be initiated through the Host Controller as follows:

Press the "Control", "\", and "0" keys for no Remote Control.

Press the "Control", "\", and "1" keys to put the Unidex 21 under Remote Control. The Unidex 21 display is not active and will not be updated during Remote operation.

Press the "Control", "\", and "2" keys to put the Unidex 21 under Remote Control. The Unidex 21 display will be active and will be updated during Remote operation.

1A-1-3: INITIALIZATION THROUGH UNIDEX 21

The Unidex 21's may be initialized through it's keyboard (sealed membrane Front Panel, TeleVideo 905 Terminal or IBM AT) as follows:

Press the "Control", "\", and "0" keys for no Remote Control.

Press the "Control", "\" and "1" keys to put the Unidex 21 under RS-232 Remote Control. The Unidex 21 display is not active and will not be updated during Remote operation.

Press the "Control", "\", and "2" keys to put the Unidex 21 under RS-232 Remote Control. The Unidex 21 display will be active and will be updated during Remote operation.

Press the "Control", "\" and "3" keys to put the Unidex 21 under IEEE-488 Remote Control. The Unidex 21 display is not active and will not be updated during Remote operation.

Press the "Control", "\", and "2" keys to put the Unidex 21 under IEEE-488 Remote Control. The Unidex 21 display will be active and will be updated during Remote operation.

SECTION 1A-2: OPERATION

Following initialization, the Unidex 21 is controlled by the Host Controller.

Communication from the Host Controller to the Unidex 21 is accomplished in the same manner as communication from the TeleVideo 905 Terminal. (See Chapter 2 of the *Unidex 21 User's Manual.*)

NOTE: **Regardless of the keyboard configuration of the Host Controller, communication to the Unidex 21 must follow TeleVideo 905 Terminal input conventions**

As data is keyed into the Unidex 21 from the Host Controller, the requested function is performed. When the Unidex 21 is ready for anotherr command, the Unidex 21 feeds back the same key sequence to the Host Controller's display. If an error is detected by the Unidex 21, the error Code identification number will be sent to the Host Controllers display instead of the input key sequence.

The next Section provides a complete list of possible error codes and their corresponding messages.

SECTION 1A-3: ERROR CODES AND MESSAGES

During RS-232 data transmission and/or performance of a function, if an error is detected, the Unidex 21 will feed back an error code in the following format:

Master Error Code (087H) followed by the Secondary Error Code (1 or 2 bytes)

During IEEE-488 data transmission and/or performance of a function, if an error is detected following a Serial Poll, the Unidex 21 will feed back an error code in the following format:

Master Error Code (C0H) followed by the Secondary Error Code (1 or 2 bytes)

The following is a list of the Secondary Error Codes and Messages as well as the function from which they may occur.

1A-3-1: EDIT MODE

The following Secondary Error Codes/Messages may appear while in the Edit Mode:

10H - Input key undefined
11H - Not enough User's RAM space
12H - File format error
13H - File not found
14H - File read only
15H - Block functions got range error
16H - Input key not ctrl-Q or ctrl-W
17H - Input key not Y or N

1A-3-2: FILE MODE

The following Secondary Error Codes/Messages may appear while in the File Mode:

20H - Input key undefined
21H - Undefined I/O port
22H - File format error
23H - File not found
24H - File read only
25H - File currently active
26H - No disk
27H - Not enough User's RAM space
28H - File verify error
29H - RS232/IEEE-488 time out, or transfer interface fail

2AH - Target file exists already
2BH - Not enough disk space
2CH - Disk write protected
2DH - Disk access fail
2EH - Disk up load fail

1A-3-3: MACHINE MODE

The following Secondary Error Codes/Messages may appear while in the Machine Mode:

30H - Input key undefined
31H - File not found
32H - Illegal filename.type
33H - Sub-program not found
34H - Can't open read file
35H - Can't open write file
36H - Write file not closed

40H - Undefined symbol
41H - Format error
42H - Undefined Type 2 command
43H - Undefined G code
44H - Undefined M code
45H - Illegal BCD format
46H - Illegal system variable
47H - Undefined variable
48H - Illegal I/O format
49H - Illegal mathematics format
4AH - Undefined array
4BH - Miss CLS command
4CH - Undefined subroutine
4DH - Undefined entry
4EH - Undefined condition
4FH - Stack overflow
50H - Miss return address
51H - Undefined safe zone
52H - Illegal function in MDI
53H - Not enough memory space
54H - Circle miss center point
55H - No feed rate

56H - Move into safe zone
57H - Undefined data in read file
58H - In ICRC look ahead
59H - <no>
5AH - MALC format error
5BH - CPAG format error, or need (MALC, < 1, option
5CH - Undefined H code
5DH - Undefined axis plane
5EH - Axis can't be both master & slave, or more than 1 master
5FH - PLC Option not foun, or ladder program not exist
60H - Need (MALC to allocate memory
61H - No recorded position to play back, need (RECO
62H - PSO Option not found

1A-3-4: PARAMETER MODE

The following Secondary Error Codes/Messages may appear while in the Parameter Mode:

70H - Input key undefined
71H - Input data error
72H - Not enough memory for p-meter save
73H - File exist already for p-meter save
74H - File not found for p-meter load

1A-3-5: TEST MODE

The following Secondary Error Codes/Messages may appear while in the Test Mode:

80H - Input key undefined
81H - RAM fail at (0) case
82H - RAM fail at (F) case
83H - RAM fail at (5) case
84H - RAM fail at (A) case
85H - RAM checksum error
86H - EPROM checksum error
87H - PARAMETER checksum error

1A-3-6: SYSTEM MODE

The following Secondary Error Codes/Messages may appear while in the System Mode:

90H - Input key undefined
91H - TIME input error
92H - DATE input error

1A-3-7: MAIN ERRORS

The following Secondary Error Codes/Messages may appear during Remote operation:

A0H - Input key undefined
A1H - No password privlege
A2H - Batch file not found or format error
A3H - RAM error during power on test
A4H - Indexing board error during power on test
A5H - Real time clock fail, set at default data

1A-3-8: SPECIAL REMOTE SYSTEM-FAIL ERROR

During RS-232 data transmission and/or performance of a function, a Special Remote System error having two bytes of Secondary Error may be detected, it will be displayed in the following format:

Master Error Code (087H) followed by 0E0H and the Secondary Error Code

During IEEE-488 data transmission and/or performance of a function, a Special Remote System error having two bytes of Secondary Error may be detected, it will be displayed in the following format:

Master Error Code (C0H) followed by 0E0H and the Secondary Error Code

The following Secondary Error Codes/Messages may appear during data transmission and/or performance of a function:

80H - Indexer 68000 CPU Bus Error
81H - Indexer 68000 Address Error
82H - Indexer 68000 Illegal Instruction
83H - Indexer 68000 Zero Divide
84H - Indexer 68000 Line 1010 Emulation
85H - Indexer 68000 Line 1111 Emulation

86H - Indexer 68000 Uninitialized Interrupt Vector
87H - Indexer 68000 Spurious Interrupt
88H - Indexer Dual-Port Ram Group B Checksum
89H - Indexer Dual-Port Ram Group B Data Out of Boundary
8AH - Feedrate is 0 or Negative Value
8BH - Invalid Sin/Cos Combination
8CH - Invalid Contouring Plane

A0H - Axis in Limit (Software or Hardware)
A1H - Axis Trap (Velocity or Position or Integral)
A2H - M Function Output Fail to Detect the Acknowledge Signal
A3H - S Function Output Fail to Detect the Acknowledge Signal
A4H - T Function Output Fail to Detect the Acknowledge Signal
A5H - DSP Feedback Illegal Code

B0H - MFO = 0 or Feedhold is On
B1H - AC Fail
B2H - Joy-Stick/ Trackball/Handwheel Motion Hit Software or Hardware Limit

SECTION 1A-4: SAMPLE PROGRAM

The following program is representative of a Basic Program that may be sent to the Unidex 21 from a Host Controller.

```
10    CLS
20    CLOSE
30    PRINT "SAMPLE PROGRAM FOR UNIDEX21 HOST REMOTE  CONTROL OPERATION"
40    PRINT "SET RS232 AS 9600,N,8,1, CONNECT TO UNIDEX21 PORT-A"
50    ON ERROR GOTO 1000
60    OPEN "COM1:9600,N,8,1" FOR RANDOM AS 1
70    RT$ = CHR$ (28)                         'Remote enable code is ctrl\2
80    GOSUB 500                               'Sent to Unidex21
90    RT$ = "2"                               'Remote case  2
100   GOSUB 500
110   RT$ = INKEY$                            'Read data from keyboard
120   If RT$ = " " THEN 110                   'No data
130   GOTO 100                                'Sent out

500   PRINT #1, RT$;                          'Sent to Unidex21
510   RTB$ = INPUT$(1, #1)                    'Feedback from Unidex21
515   IF RT$ < >CHR$(10) THEN 540             'If LF skip everything
520   IF RT$ <  > RTB$ THEN 550               'The same?
530   PRINT RT$;                              'Yes
540   RETURN                                  'No error, back to caller
550   IF RTB$  <  >CHR$(135) THEN 900         'Remote error code
560   RTB$ = INPUT$(1,#1)                     'Yes, what is the error code?
570   IF RTB$ = CHR$(224) THEN 600            '2 bytes error code?
580   PRINT "REMOTE ERROR CODE ="; ASC(RTB$)
590   RETURN                                  'Let caller decide what to do

600   RTB1$ = INPUT$(1, #1)                   '2nd error byte
610   PRINT "REMOTE ERROR CODE ="; ASC(RTB$); "+"; ASC(RTB1$)
620   RETURN                                  'Let caller decide what to do

900   PRINT  "HOST REMOTE CONTROL FAIL"
910   GOTO 1010

1000  PRINT " ERROR CODE = "; ERR; "try again"
1010  STOP
1020  END
```

INDEX

A

B

C

D

E

F

G

H

I

J

UNIDEX® 21
PROGRAMMING MANUAL

PN: EDU118

REFERENCE
DOCUMENT

AEROTECH, Inc., 101 Zeta Drive, Pittsburgh, PA 15238
(412) 963-7470 • TWX 710-795-3125 • FAX(412) 963-7459

DISCLAIMER:

The information contained in this manual is subject to change due to improvements in design. Though this document has been checked for inaccuracies, Aerotech does not assume responsibility for any errors contained herein.

TRADEMARKS:

Unidex is a registered trademark of Aerotech, Inc.

IBM PC/XT/AT are registered trademarks of the International Business Machines Corporation.

PAMUX is a registered trademark of the Opto 22 Corporation.

REVISION HISTORY:

Revision 1 - February, 1992
Revision 2 - January, 1993

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

This manual provides detailed information for each of the program commands utilized by the Unidex 21 Motion Controller.

SECTION 1-1: COMMAND STRUCTURE

Commands recognized by the Unidex 21 consist of three general types:

1. A subset of RS-274-D commands (G & M codes)
2. A subset of RS-447 commands (i.e., DENT, DFS, etc.)
3. Aerotech developed extensions (i.e., $INP, $OTP, etc.)

The programming language syntax for these commands are of two types:

TYPE 1 : Machine program data in RS-274-D like format.

TYPE 2 : Machine setup, initialization, or operation parameters in RS-447 like format. Type 2 commands are designated by either ( ) or < >.

NOTE: The Type 1 (RS-274-D) format requires a space between each command (for instance G2 G17 I1).

Both Type 1 and Type 2 commands conform to the Electronic Industries Association (EIA) standards.

By conforming to these EIA standards wherever possible, the Unidex 21 program commands may be readily converted to/from other CNC controllers. Also, CAD to NC postprocessors may be easily configured.

SECTION 1-2: ENTERING PROGRAM COMMANDS

The Unidex 21 recognizes program commands created in the following ways:

The Unidex 21 Edit Mode

The Unidex 21 Motion Controller contains an Edit mode in which programs may be created or amended. The Front Panel or integral keyboard is used to enter or revise the commands. See the *Unidex 21 User's Manual* for a detailed description of the Edit Mode.

The Unidex 21 Machine Mode

The Machine Mode of the Unidex 21 may be used to amend program commands of a previously loaded program. The MDI function of the Machine Mode provides the User with the ability to stop the program, enter a move or change a status, and then return to the program run. See the *Unidex 21 User's Manual* for a detailed description of the Machine Mode.

IBM/AT Computer - DOS Operating System

Programs may be created remotely if an IBM/AT computer equipped with the IBM-DOS Operating System is used. Programs created in this manner may be copied onto a 3 1/2" floppy disk and then directly copied into the Unidex 21's memory. See the *Unidex 21 User's Manual* for a detailed description of the File Mode.

Non-IBM Computer

A non-IBM computer may be used to create programs if it is interfaced to the Unidex 21 through one of the RS-232 ports. See the *Unidex 21 Hardware Manual* for information concerning RS-232 interface and the *Unidex 21 User's Manual* for a detailed description the File Mode and the procedure of loading a file through the RS-232 ports.

CHAPTER 2: USING THE UNIDEX 21 PROGRAMMING MANUAL

This manual is to be used as an aid when programming the Unidex 21 Motion Controller. Prior to any actual programming of the Unidex 21, it is strongly suggested that the User be thoroughly familiar with the *Unidex 21 User's Manual* and have studied the Sample Programs provided throughout Chapter 3 and the appendices of this manual.

The Unidex 21's internal HELP menu is available to the User at all times by pressing the "Alt" and "H" keys.

The commands presented in this manual are in alphabetical order, symbol commands are listed first. Each command is described on an individual page with an example of it's usage. Names of related commands or commands whose description may be helpful are also included.

It should be noted that one of the most important programming capabilities which separates the Unidex 21 from other similar controllers is its extensive variable programming capability.

The following is a sample page with an explanation of the various subsections.

XX - Command

NAME:

The command name.

FUNCTION:

A description of the command and any parameters that are required by the command.

FORMAT:

A description of the command structure.

RETURNS:

Feedback to the User following execution of the command.

EXAMPLE:

```
An example of command use          ; Explanation of the example (not intended for
                                     system entry).
```

PROGRAMMING EXAMPLE:

```
An example of command used in conjunction with other commands  ; Explanation of the com-
                                                               ; mand(s) in the example
                                                               ; (not intended for system
                                                               ; entry).
```

NOTES:

Additional comments pertaining to the use of the command.

RELATED COMMANDS:

The names of additional commands that are related or whose description may be helpful.

CHAPTER 3: UNIDEX 21 PROGRAM COMMANDS

The following pages contain the commands utilized by the Unidex 21.

The Unidex 21 utilizes a combination of standard EIA and RS274-D type commands (G & M code) and an extended subset of RS447 commands.

Each command is presented alphabetically with system level variables and operands first.

%

NAME:

% - Program Title

FUNCTION:

The % command may be used as an option for beginning a program. Any text located between the % and the first <ENTER> is ignored by the Unidex 21. A program name or a comment may also be inserted into this block. Programs which are transferred via the RS-232 and IEEE488 interface may utilize the "%" as a beginning of file transmission character. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode settings.)

FORMAT:

% Program Title <ENTER>

RETURNS:

EXAMPLE:

```
% Widget Cutting <ENTER>     ; The program name is Widget Cutting.
G1                           ; All moves will be linear.
G70                          ; Initiate English programming (inches).
F100.                        ; A Feedrate of 100 inches per minute is established.
X10. Y10.                    ; Move X axis 10 inches in the positive direction,
                             ; and Y axis 10 inches in the positive direction.
```

NOTES:

The applicable Main System Parameters for RS-232 and IEEE488 communication are #6, 8, 10, 16, 17, and 53 (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode).

RELATED COMMANDS:

;, /

/

NAME:

/ - Block Delete Operator

FUNCTION:

The Block Delete operator permits the User to designate program blocks to be skipped during a program run. During program editing, a "/" may be inserted at the block's beginning to designate an optional block. Pressing the front panel **BLOCK DELETE** key toggles between activating the Block Delete function (designated blocks are skipped) or deactivating the Block Delete (all blocks are executed regardless of designation). From an AT type keyboard the Prt Sc (print screen function key) will also toggle the Block Delete.

FORMAT:

/(command name)

RETURNS:

EXAMPLE:

```
(DFS,TEST
 .
 .
 (JUMP,ENT1,VAR1.EQ.2)        ; Jump to Entry Point ENT1 if VAR1 equals 2. The system
 /(ABTS,ENT2)                 ; would not execute the ABTS command when BLOCK
 (DENT,ENT1)                  ; DELETE is active.
 .
)
(DENT,ENT2)
```

This example controls a jump to the Entry Point ENT2 before completion of the "TEST" Subroutine. If **BLOCK DELETE** is OFF and VAR1 does not equal 2 then the Abort Subroutine **ABTS** command will be executed.

NOTES:

RELATED COMMANDS:

;

;

NAME:

; - Comment Operator

FUNCTION:

Comments may be added to any command by preceding them with the ;. Any text following the ; is ignored by the Unidex 21.

FORMAT:

(program command) ; comment

RETURNS:

EXAMPLE:

```
(REF,Y,y,Z,z)      ; Axes Y, y, Z, and z are at the Software Home position.

G1                 ; All moves will be linear.
G70                ; Initiate English programming (inches).
F100               ; A Feedrate of 100 inches per minute is established.
X10. Y5.           ; Move X axis 10 inches in the positive direction,
                   ; and Y axis 5 inches in the positive direction.
```

NOTES:

Each comment line must be preceded with the ";" Comment Operator.

RELATED COMMANDS:

%, /

$HSI

NAME:

$HSI - High Speed Interrupt Buffer

FUNCTION:

The **$HSI** command is a system variable that reads one element of data from the High Speed Interrupt Buffer. Data elements are floating point format.

FORMAT:

$HSI<0>	; Get number of data elements which have been stored.
$HSI<N>	; Get Nth data element which has been stored. Note that N may range from 1 to the number returned by **$HSI<0>**.

RETURNS:

EXAMPLE:

Assume High Speed Interrupt Buffer is active and contains the following:
0.0, 16.3, 22.4, 64.75, remainder not used

```
VAR1=$HSI<0>                ; VAR1= 4, the number of data elements stored.
VAR2=0                      ; Set Variable VAR2 equal to 0.
VAR3=1                      ; Initialize the data pointer.
(RPT, VAR1                  ; Repeat the Loop 4 times.
 VAR2=VAR2+ $HSI<VAR3>      ; VAR2 = 0.0, then 16.3, then 38.7, and finally 103.45.
 VAR3=VAR3 + 1              ; Increment the data pointer to 2, 3, 4, and 5,
                            ; note, however, that 5 is not used to read data.
)

VAR2 =VAR2/VAR1             ; Average data, VAR2 = 25.8625
```

NOTES:

For a more complete example utilizing the **$HSI** and related commands, see the example under the **HSIE** command

RELATED COMMANDS:

HSIE, MALC

$INn

NAME:

$INn - Input System Variable

FUNCTION:

The Unidex 21 has 16 optoisolated inputs available on the CPU card. The **$INn** represents the logic state of one of the input lines when n=0 to F. Or, a **$INP** command may be utilized to look at the logic state of all 16 input lines at once. These inputs typically interface to an I/O mounting rack. The values are recorded in hexadecimal and may be shifted, summed, or converted utilizing Unidex 21's Math Package. When programming the outputs, the User must recognize that input values are reversed relative to their module location. Therefore, if outputs 0 and 1 are set to zero active (ON) and all others on the I/O mounting rack are inactive (OFF), then the associated **$INP** hexadecimal status value is FFFC **not** CFFF.

FORMAT:

$IN0-$INF	; Represents individual input variable logic states.
or	
$INP	; Represents all 16 input variable logic states. Used when you wish to control program flow based upon entire word.

RETURNS:

EXAMPLES:

VAR1=**$IN0**	; If Input 0 is active (ON) then VAR1=H,00.
(JUMP,CRY1,**$INP**.EQ.H,0007)	; Jump to Entry Point CRY1 if inputs 0, 1, and 2 are inactive (OFF) and all other outputs are active (ON).
(JUMP,CRY2,**$IN1**.EQ.H,00)	; Jump if the value of Input equals H,00 active (ON).
(MSG,#H,**$INP**)	; Display value of inputs in hexadecimal; values would range from 0 to FFFF. This is useful for debug purposes in order to check the value of inputs.

$INn

NOTES:

1) Valid individual input characters are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F.

2) On a standard I/O mounting rack assembly, PB16 or PB24 Input 0 (**$IN0**) actually resides at module position 8. (Modue positions 1 to 7 are reserved as outputs, see **$OTn.**)

3) Additional interrupt inputs, high speed position grabbing, and fast feedhold inputs are also available.

4) Reference the *Unidex 21 Hardware Manual* for further details.

RELATED COMMANDS:

$OTn, DVAR, HSIE, INT1/INT2, MSG, SIOC

$MFO

NAME:

$MFO - Manual Feed Override System Variable

FUNCTION:

Represents the current percentage of Manual Feedrate Override. Manual Feedrate Override is the multiplier used to raise or lower the Feedrate at which the system is operating. Or, a percentage multiplier of a programmed base. As an example, if a User's Parts Program specifies a Feedrate of 100.0 inches per minute a Manual Feedrate Override setting of 125% would cause the system to operate at 125.0 inches per minute.

FORMAT:

$MFO

$MFO may be equated to any other variable.

RETURNS:

Returned value is the percentage value of the current **MFO** divided by 100.

For example: **$MFO** = $\frac{\text{MFO\%}}{100}$

EXAMPLES:

VAR1=**$MFO**	; If MFO is set to 150%, VAR1 will equal 1.5.
(MSG,#**$MFO**)	; The current MFO setting will be displayed as 1.25 if the MFO is set at 125%.

NOTES:

1) While **$MFO** is the **MFO** percentage divided by 100, the **UMFO** program command is the actual % value. i.e., 125% equals 125.

2) **G0** moves are truncated at 100% maximum **MFO**.

RELATED COMMANDS:

CPAG, DVAR, MSG, UMFO

$nAP/$nRP

NAME:

$nAP - Absolute Commanded Position Register Variables

$nRP - Relative Commanded Position Register Variables

FUNCTION:

The Unidex 21 utilizes two position registers per axis referred to as Absolute and Relative Axis Commanded Position Registers. Each may contain the same or different value depending upon the interaction of other system commands like **FXOF**, **G92**, **HOME**, and **MORG**. The Tracking Display typically shows the **$nRP** values.

The **$nAP** refers to the axes Absolute Commanded Position Registers, where "n" is the primary axis letter (i.e., **X**, **Y**, **Z**, **U**, **x**, **y**, **z**, **u**). Upon homing the system or initial power up, these registers are initialized at zero. These registers are unaffected by the **G92** command, yet may be utilized to reinitialize the **$nRP** registers to **$nAP** values. This is especially useful if switching between **G90** and **G91** modes in which case the User is unsure of his exact position. They may also be utilized for verification of offset values placed in the User's Parts Program.

The **$nRP** refers to the axes Relative Commanded Position Registers, where "n" is the primary axis letter (i.e., **X**, **Y**, **Z**, **U**, **x**, **y**, **z**, **u**). Upon homing the system or initial power up, these registers are also initialized at zero. These registers may be changed by utilizing the **G92** command. When the control makes a move in **G90** or **G91** mode, it utilizes the **$nRP** registers. See Notes for special cases and related commands.

FORMAT:

$XAP $YAP $ZAP $UAP $xAP $yAP $zAP $uAP

$XRP $YRP $ZRP $URP $xRP $yRP $zRP $uRP

These expressions may be equated to any other variable.

RETURNS:

$nAP/$nRP

EXAMPLES:

VAR1=**$xRP**	; VAR1 is set equal to the Relative Position of the x axis.
VAR1=**$xAP**+**$XAP**	; VAR1 is set equal to the sum of the Absolute Position of the x axis and the X axis.
(MSG, X position is #**$XAP**)	; Message display will show: "X position is 25.000" if the current Absolute Position is 25.000.
G92 X=**$XAP**	; The system X axis Relative Machine Register Position is reinitialized to the Absolute Position Register value. The machine's position Tracking Display will also be changed.

PROGRAMMING EXAMPLE:

The following is an example of how the system variables relating to Absolute and Relative Positions are related to each other as well as how they are affected by the various commands.

NOTE: The Machine Origin steps parameter for the X axis corresponds with 10.0 inches. The Y axis parameter corresponds with 25.0 inches.

```
G1              ; All moves will be linear.
G17             ; The XY plane is the 1st plane.
G40             ; Ensure Cutter Compensation is OFF.
G91             ; Initiate Incremental Positioning.
G70             ; Initiate English Programming (inches).
F1000.          ; Feedrate is 1000 inches per minute.
(REF, X, Y)     ; Find the Hardware Home for the X and Y axes.
                ;     $XRP=0.0     $YRP=0.0
                ;     $XAP=0.0     $YAP=0.0
X10. Y100.      ; Move X axis 10 inches in the positive direction,
                ; and Y axis 100 inches in the positive direction.
                ;     $XRP=10.0    $YRP=100.0
                ;     $XAP=10.0    $YAP=100.0
X10. Y-10.      ; Move X axis 10 inches in the positive direction,
                ; and Y axis 10 inches in the negative direction.
                ;     $XRP=20.0    $YRP=90.0
                ;     $XAP=20.0    $YAP=90.0
```

$nAP/$nRP

PROGRAMMING EXAMPLE (CON'T):

```
G92 X0. Y0.            ; Establish a Software Home at the current position.
                       ;    $XRP=0.0      $YRP=0.0
                       ;    $XAP=20.0     $YAP=90.0
X-20. Y35.             ; Move X axis 20 inches in the negative direction,
                       ; and Y axis 35 inches in the positive direction.
                       ;    $XRP=-20.0    $YRP=35.0
                       ;    $XAP=0.0      $YAP=125.0
(FXOF, X100., Y100.)   ; Move axes to a new coordinate frame such that $nRP
                       ; registers retain their value while $nAP values are
                       ; changed.
                       ;    $XRP=-20.0    $YRP=35.0
                       ;    $XAP=100.0    $YAP=100.0
X10. Y15.              ; Move X axis 10 inches in the positive direction,
                       ; and Y axis 15 inches in the positive direction.
                       ;    $XRP=-10.0    $YRP=50.0
                       ;    $XAP=110.0    $YAP=115.0
(MORG, X,Y)            ; Move to the Machine Origin for these axes, which is
                       ; the Hardware Home position including Machine Origin
                       ; Offset while not disturbing the $nRP registers.
                       ; (See Note above)
                       ;    $XRP=-10.0    $YRP=50.0
                       ;    $XAP=10.0     $YAP=25.0
X20. Y50.              ; Move X axis 20 inches in the positive direction,
                       ; and Y axis 50 inches in the positive direction.
                       ;    $XRP=10.0     $YRP=100.0
                       ;    $XAP=30.0     $YAP=75.0
G90                    ; Initiate Absolute Positioning.
X100. Y100.            ; Move to Absolute Position 100., 100.  In this case,
                       ; the Y axis does not move.
                       ;    $XRP=100.0    $YRP=100.0
                       ;    $XAP=120.0    $YAP=75.0
M2                     ; End of The Program.
```

$nAP/$nRP

NOTES:

1) Normally upon power up or after homing, the **$nAP** and **$nRP** registers will equal the same value. The exception is when an Absolute Positioning system (i.e., resolver/inductosyn combination) is utilized. Here, the **$nAP** and **$nRP** registers will equal the machines Absolute Position.

2) The related commands, listed below, also affect **$nAP** and **$nRP** commands.

3) Usage of Machine Origin and Home Offset parameters also affect **$nAP** and **$nRP** commands.

4) The applicable Individual Axis Parameter is #7 as established under the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

5) **$nAP** and **$nRP** reflect the commanded position of the axis and not necessarily the actual axis position.

RELATED COMMANDS:

FXOF, G92, HOME, MORG, REF

$OTn

NAME:

$OTn - Address Logic Output System Variable

FUNCTION:

The Unidex 21 has 8 optoisolated outputs available on the CPU card. The **$OTn** represents these outputs, where "n" is the output number ranging from 0 - 7. Alternatively, **$OTP** may be used to write all 8 output lines at once. The values are set up as hexadecimal, and may be shifted, summed, or converted utilizing Unidex 21's Math Package (see Chapter 4). When programming the outputs, the User must recognize that outputs are programmed in reverse to their module location. As an example, to turn ON outputs 0, 1, and 2 (set to zero) and all other outputs OFF (set to one) program **$OTP**=H, F8 **not** 8F.

These outputs typically interface to an I/O mounting rack. As opposed to MST bus outputs, if an output is commanded between motion blocks, motion will be momentarily interrupted during processing. If output on-the-fly is desired, utilize the MST bus instead.

FORMAT:

$OTn=x	; In this case, "n" is the Output number (0-7) and "x" equals 0 (ON) or 1 (OFF). An alternate way is to specify the hex equivalent such that "x" equals H,00 (ON) or H,01 (OFF).
$OTP=H,xx	; This addresses all 8 outputs simultaneously and must utilize hexadecimal format only. "xx" ranges from 00 (all ON) to FF (all OFF).

RETURNS:

EXAMPLE:

$OT5=0	; Output 5 is active ON.
$OT6=H,01	; Output 6 is inactive OFF.
$OTP=H,6D	; This is equivalent to binary representation 01101101 which means that Output 0=1 (OFF) etc.
$OTP=VAR1	; The output state is equal to the data value of VAR1.

$OTn

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #1 and 2 located in Appendix 3 of this manual.

NOTES:

1) On a standard I/O mounting rack assembly PB8, 16, or 24 **$OT0** resides at position 0.

2) These outputs are not latched, and should be considered write only. Once enabled, the opto rack assembly will remain in the set state until <reset>, power down, or readdressing occurs.

3) Additional I/O is available via the MST bus and PAMUX I/O channel options. (See the *Unidex 21's Options Manual* for further detail.)

4) An output may also be activated by utilizing output on axis trap parameter which may be set under Unidex 21's Main System Parameter #56. (See the *Unidex 21's User's Manual* for a detailed description of the Parameter Mode.) This output will also be negated for program faults.

RELATED COMMANDS:

$INn, DVAR, SIOC

$POT

NAME:

$POT - RS-232 Port Receive Buffer

FUNCTION:

The **$POT** command is a system variable that reads one element of data from the background RS-232 Port Buffer (refer to the **PORT** and **MALC** commands). Data elements are four bytes in length, with the top three bytes set to zero and the least significant byte containing the data byte received from the RS-232 port (in hexadecimal format).

FORMAT:

$POT<0>	; Get number of data elements which have been stored.
$POT<N>	; Get Nth data element which has been stored. N may range from 1 to the number returned by **$POT<0>**.

RETURNS:

EXAMPLE:

Assume the Port Buffer is active and contains the following data:
00000031 00000032 00000033 00000034, remainder not used

VAR1=**$POT<0>**	; VAR1 = 4, the number of data elements stored
VAR1=**$POT<1>**	; VAR1 = H, 00000031
VAR1=**$POT<2>**	; VAR1 = H, 00000032
VAR1=**$POT<3>**	; VAR1 = H, 00000033
VAR1=**$POT<4>**	; VAR1 = H, 00000034

NOTES:

For a more complete example of the use of **$POT** and related commands, see the example under the **PORT** command.

RELATED COMMANDS:

MALC, PORT

$R

NAME:

$R - Read Command

FUNCTION:

The **$R** command is a system variable that reads one element of data from a Read file, then increments the pointer to the next available data. If retrace is being used, the **$R** command will move back one data element and read. Data elements are separated by a space or a comma.

FORMAT:

$R

$R may be used as any other variable.

RETURNS:

EXAMPLE:

Assume data file is open and contains the following:

0, 16, 22, 64.75

VAR1=**$R**	; VAR1 = 0
VAR2=**$R**	; VAR2 = 16
VAR1=**$R**	; VAR1 = 22
VAR1=**$R**	; VAR1 = 64.75
(MEND, R)	; Reset the read pointer
VAR2=**$R**	; VAR2 = 0

NOTES:

For a more complete example utilizing the **$R** and related commands, see the example under the **OPEN** command.

RELATED COMMANDS:

END, MEND, OPEN, WRIT

$RTP

NAME:

$RTP - Real Time Position Buffer

FUNCTION:

The **$RTP** command is a system variable that reads one element of data from the Real Time Position Buffer (refer to the **MALC** and **RETP** commands). Data elements are hexadecimal format. The values represent axis position in machine steps.

FORMAT:

$RTP<0>	; Get number of data elements which have been stored. This number is the product of the number of times data has been stored times the number of axes for which position information is being recorded.
$RTP<N>	; Get Nth data element which has been stored. N may range from 1 to the number returned by **$RTP**<0>.

RETURNS:

EXAMPLE:

Assume Real Time Position Buffer is active, enabled to fetch position for the X and Y axes, and contains the following data:

00000000 00000163 80000224 000FA175, remainder not used

```
VAR1=$RTP<0>            ; VAR1 = 4, the number of data elements stored
VAR1=BTF($RTP<1>)       ; VAR1 = 0.0, first X axis position recorded

VAR2=$RTP <2>           ; VAR2 = H, 00000163, first Y axis position recorded
VAR2=BTF(VAR2)          ; VAR2 = 355.0

VAR1=$RTP<3>            ; VAR1 = H, 80000224, second X axis position recorded
VAR1=BTF(VAR1)          ; VAR1 = 2147483100.0

VAR2=BTF($RTP<4>)       ; VAR2 = 1024373.0
```

$RTP

NOTES:

For a more complete example utilizing the **$RTP** and related commands, see the example under the **RETP** command.

RELATED COMMANDS:

MALC, RETP

$TOD

NAME:

$TOD - Time of Day System Variable

FUNCTION:

This command outputs the current time/date for the **COMM**, **CPAG**, **MSG**, and **WRIT** commands. Individual components may also be accessed as floating point numbers.

These may be used for data logging or Statistical Process Control (SPC).

FORMAT:

$TOD	; Full time/date information for the **COMM**, **CPAG**, **MSG**, and **WRIT** commands only.
or	
$TOD<option>	; Individual component of time/date information
$TOD<Y>	; Number of current year
$TOD<M>	; Number of current month
$TOD<D>	; Number of current day
$TOD<H>	; Number of current hour
$TOD<m>	; Number of current minute
$TOD<S>	; Number of current second

RETURNS:

EXAMPLES:

```
(MSG, #$TOD)            ; The message display shows:  24-SEP-2004 05:35:22

(MSG, #$TOD<Y>)         ; The message display shows:  2004

(MSG, #$TOD<M>)         ; The message display shows:  09

VAR1=#$TOD<H>+10        ; Variable VAR1 equals 15 if the hour is 05.
```

$TOD

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #2 located in Appendix 3 of this manual.

NOTES:

RELATED COMMANDS:

COMM, CPAG, MSG, WRIT

$000-$FAF

NAME:

$000-$FAF - Extended I/O Cability

FUNCTION:

Unidex 21 provides for extended Input/Output capabilities beyond the standard 16 inputs, 8 outputs, MST bus, and 3 User interrupts.

The extended I/O is accessed using the system I/O variable. This permits the User to utilize the Motorola I/O channel, integral PLC option, and Opto 22 PAMUX channel.

$000-$7FF is used to address the Motorola I/O channel.

$800-$F6F is used to address the PLC Dual-Port Memory locations.

$F70-$FAF is used to address the PAMUX I/O channels for reading or writing data.

FORMAT:

$000-$FAF	; Address entire byte of I/O
or	
$000**<0-7> - **$FAF<0-7>	; Address individual bit of I/O

RETURNS:

EXAMPLES:

VAR1=**$000**	; Lower byte of VAR1 is set to the status of the first eight I/O lines of the Motorola I/O channel at **$000**, upper 3 bytes of VAR1 are set to 0.
$VAR1=**$000**<3>	; Variable VAR1 is set to 1 if the status of the fourth I/O lines (line 3 of 0-7) of the Motorola I/O channel at address **$000** is active, or to 0 if it is not.
$700<7>=0	; Set I/O line 7 at **$700** to inactivate.

$000-$FAF

EXAMPLES (CON'T):

$701=H,72	; Set I/O lines 0-7 at **$701** to the following pattern: (01110010)
	; Line 0 OFF
	; Line 1 ON
	; Line 2 OFF
	; Line 3 OFF
	; Line 4 ON
	; Line 5 ON
	; Line 6 ON
	; Line 7 OFF

NOTES:

1) Refer to the *Unidex 21 Hardware Manual* and individual *Options Manual* for a complete description of each I/O group.

2) The Unidex 21's Math Package (see Chapter 4) may be utilized on I/O system variables.

RELATED COMMANDS:

DVAR, SIOC

ABTS

NAME:

ABTS - Abort Subroutine

FUNCTION:

Upon completion of a Subroutine, the Unidex 21 normally returns to the next block of the "Caller" Program (the program that called the Subroutine). The **ABTS** command allows the Subroutine to be aborted and program execution to continue from an entry point defined by the User.

FORMAT:

(**ABTS**,entry point)

RETURNS:

EXAMPLE:

```
(DFS,TEST
 .
 .
 (JUMP,ENT1,VAR1.EQ.2)          ; Jump to Entry Point ENT1 if variable
 (ABTS,ENT2)                    ; VAR1 equals 2. If it does not, then
 (DENT,ENT1)                    ; ABTS command will be executed.
 .
)
(DENT,ENT2)
```

This example controls a jump to the Entry Point ENT2 before completion of the Subroutine rather than returning to the Main Program.

ABTS

NOTES:

1) The **ABTS** may be used in one of four ways; directly, indirectly, conditionally direct, or conditionally indirect. An example of each follows:

(**ABTS**,ENT1)	; Aborts current Subroutine, jumps to Entry Point ENT1.
(**ABTS**,#VAR1)	; Variable (VAR1) contains the ASCII name of the entry point.
(**ABTS**,ENT1, VARA.EQ.1)	; Program jumps to Entry Point ENT1 if VARA equals 1.
(**ABTS**,#VAR1,VARA.EQ.1)	; Program jumps to entry point defined by VAR1 if VARA equals 1.

2) **ABTS** is not a modal command.

3) The former return address is removed from the program stack.

RELATED COMMANDS:

CLS, DENT, DFLS, DFS, INT1/INT2

ACDE

NAME:

ACDE - Maximum Acceleration/Deceleration

FUNCTION:

The **ACDE** command provides the User with the ability to set the Acceleration/Deceleration rate for each axis, overriding the Parameter Mode setting of the maximum Acceleration and Deceleration rates. The Acceleration/Deceleration Parameter Mode settings are not changed by this command. The rates set with this command remain in effect until updated by a subsequent **ACDE** command, or the system is reset.

The **ACDE** rate effects only the jog functions (keyboard arrow keys) in both velocity and step modes, **FREE** (cases 2+3), **FXOF**, **G0**, **HOME** (REF), and **MORG** commands.

FORMAT:

(**ACDE**, axis name and value, axis name and value, ...)

RETURNS:

EXAMPLES:

```
(ACDE,X1000000,Y=VAR1,Z=ARY<3>)      ; This sets the Acceleration/Deceleration rate
                                     ; of the X, Y, and Z axes utilizing numeric
                                     ; value, variables value, and an array element
                                     ; respectively.

(ACDE,D)                             ; This returns the Acceleration/Deceleration
                                     ;  rate of all axes to the values established in
                                     ;  the Parameter Mode.
```

ACDE

NOTES:

All Acceleration/Deceleration rates established by the **ACDE** command are set in machine steps/sec^2, no decimals, for each axis. This means the User must take into account the machines step resolution, before parameter scaling.

RELATED COMMANDS:

DVAR, FILT, FREE, FXOF, G0, HOME, INT1/INT2, MORG, RAMP, REF

AFCO

NAME:

AFCO - Auto-Focus

FUNCTION:

The **AFCO** command provides the User with Auto-Focus control, i.e., each axis has the ability to lock into any one of two inputs on the Unidex 21's Indexer board. The input may be quadrature, CW/CCW clock, or CL/DIR signals. The number of counts present in the Auto-Focus register determines the velocity which the axes will move. This is different from the **HWEL** command where each count causes a particular distance to be moved. The Auto-Focus conversion factor as established in the Individual Axis Parameter Mode may be programmed to a different factor with this command. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.) The parameter values are not changed by this command. The Auto-Focus mode set by this command remains in effect until a subsequent **AFCO** command is used, or the system is reset.

FORMAT:

(**AFCO**, axis name and input number, axis name and input number, ...)

or

(**AFCO**, axis name and input number, <conversion factor 1 to 255>)

RETURNS:

EXAMPLES:

(**AFCO**,X1,Y2)	; The X axis is locked to Input 1. ; The Y axis is locked to Input 2 such that a CW input signal will cause CCW motion. ; The current conversion factor will be used.
(**AFCO**,X1,Y1)	; Both the X and Y axes are locked to Input 1. ; The current conversion factor will be used.
(**AFCO**,X=VAR1)	; The X axis is locked to data of variable VAR1, where variable VAR1= 0,1,2. ; The current conversion factor will be used.

AFCO

EXAMPLES (CON'T):

```
(AFCO,X1,<3.5>)          ; The X axis is locked to Input 1.
                         ; The conversion factor is set to 3.5.

(AFCO,X0)                ; The X axis is unlocked from any input.
```

NOTES:

1) The **AFCO** command can reverse the clock input direction if the input number is preceded by a "-".

2) A single input may be locked to multiple axes.

3) The applicable Main System Parameters are #28 and #29. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

4) The applicable Individual Axis Parameters are #40 and #41 as established under the Individual Axis Parameters 401-408 grouping. (See also the *Unidex 21 User's Manual* for further detail regarding these Parameters.)

5) The conversion factor scales the **AFCO** input, where the parameter values may be multiplied by up to 255.

RELATED COMMANDS:

HWEL, SLEW

CCP

NAME:

CCP - Cutter Compensation

FUNCTION:

The Cutter Compensation command provides two axis setting and tool diameter information for **ICRC**. This information is used when **ICRC** is activated. The tool diameter value of **CCP** should be positive only. For more complete examples of the use of **CCP** with **G40**, **G41**, and **G42**, refer to the Intersectional Cutter Radius Compensation (**ICRC**) Appendix 1.

FORMAT:

(**CCP**,1st,2nd,nnn)	; 1st is the name of the first axis, 2nd is the name of the second axis, nnn is the tool diameter (programmed as a decimal number). The tool diameter may also be expressed as a variable.

A special format may be used in order to match the format recognized by the Unidex 16. The format is:

(**CCP**,T1=VAR) is the same as (**CCP**,X,Y,VAR)

(**CCP**,T1,nn) is the same as (**CCP**,X,Y,nn)

RETURNS:

EXAMPLES:

G70	; Initiates English Programming (inches).
(**CCP**,X,Y,1.5)	; ICRC is on the X/Y plane with a tool diameter of 1.5 inches.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #5 located in Appendix 3 of this manual.

CCP

NOTES:

1) **ICRC** is valid for Linear and Circular Interpolation only.

2) The **CCP** command is a modal command, i.e., any subsequent **CCP** command for a particular tool updates the tools **CCP** to the new value.

RELATED COMMANDS:
DVAR, G40, G41, G42

CLS

NAME:

CLS - Call Subroutine

FUNCTION:

The **CLS** command calls a User Defined Subroutine or a User Defined Library Subroutine.

FORMAT:

(**CLS**, subroutine name)

The Subroutine name may range from 2 to 4 characters in length. It should be defined uniquely by a **DFS** Define Subroutine or **DFLS** Define Library Subroutine command. The first two characters are letters from A-Z, the remaining two are alphanumeric characters.

A Subroutine may be called directly, such as:

(**CLS**,AB12)	; Subroutine AB12 is called.

or indirectly:

(DVAR,VAR1)	; A variable name defined as ASCII string, must be contained within quotes.
(**CLS**,#VAR1)	; The variable VAR1 contains the ASCII name of the Subroutine. In this example, it would call the Subroutine named ABCD.

RETURNS:

The return address is kept in a User's stack. If the Subroutine is defined as a Library Subroutine, the modal commands are also kept in the stack.

EXAMPLE:

```
VAR1="SUB1"
(CLS,#VAR1)
M2
```

CLS

EXAMPLE (CON'T):

```
(DFS,SUB1
  .
  .
)

        This is the same as;

(CLS,SUB1)
M2
(DFS,SUB1
  .
  .
)
```

In the first example, #VAR1 tells the Unidex 21 that the Subroutine is defined by a variable. This method of defining subroutines allows flexibility in programming since the Subroutine being called can vary with the value of VAR1.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #4 located in Appendix 3 of this manual.

NOTES:

1) All User defined variables are global. The Unidex 21 does not provide local variables; however, if a User Defined Variable is to be used as a local variable in a Subroutine, the value of the variable may be stored on the User stack with the **PUSH** instruction, and later restored with the **POP** instruction.

2) Calling a normal Subroutine requires 8 bytes of User memory. Calling a Library Subroutine requires 52 bytes of User memory.

RELATED COMMANDS:

ABTS, DFLS, DFS, POP, PUSH, STKP

COEF

NAME:

COEF - Coefficient for Parabolic Trajectory Ramping

FUNCTION:

The **COEF** command is used to select a Parabolic Trajectory Coefficient for Ramping. The coefficient may range from 0 to 65,535. The value established by this command will override the coefficient established in the Parameter Mode, but does not change the Parameter Mode setting. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.) The value of the Parabolic Trajectory Coefficient set with this command will remain in effect until a subsequent coefficient command is issued, or the system is reset.

FORMAT:

(**COEF**,axis and coefficient, axis and coefficient,...)

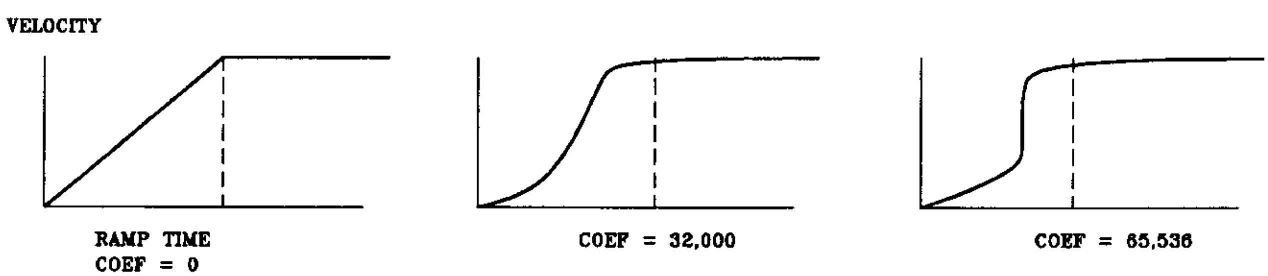
RETURNS:

EXAMPLE:

```
(COEF,X20., Y1000., Z=VAR1)
```

NOTES:

1) The value of the coefficient affects the shape of the Acceleration/Deceleration velocity profile. A coefficient of 0 becomes a linear ramp. Higher values result in a steeper curve as illustrated below.



2) The relevant Main System Parameters are #30, 57, and 58. The applicable Individual Axis Parameters are #9 and #38 as established under the 401-408 grouping.

RELATED COMMANDS:

ACDE, FILT, RAMP, TRAJ

COMM

NAME:

COMM - Communication through RS-232 Port A/B

FUNCTION:

The **COMM** command provides routing for data input or output through the RS-232 port within a program. This is particularly useful for controlling remote devices, and passing messages between external instruments such as Lasers and PLC's (Programmable Logic Controllers).

FORMAT:

(**COMM**,port,<optional input>, text)

The Unidex 21 recognizes the following format for output through RS-232:

#VAR	- as the decimal format of the variable
#H:VAR	- as the hexadecimal format of a variable
#C:VAR	- as the character format
#C:VAR1,VAR2,VAR3	- as a variable string

RETURNS:

Comm, I,

EXAMPLES:

(**COMM**,A,...text)	; Data is sent through the RS-232, Port A.
(**COMM**,B,<VAR1,ARY<2>>, ...text..)	; Data is sent and input data returned through the RS-232, Port B.
(**COMM**,A,#C:H,0D0A)	; Transmit the carriage return and line feed through the RS-232, Port A.

Input data may be floating numbers (123.4), hex (H,1F30), or characters (ABCD).

If an input string consists of more than four characters, "12345678", the string will be divided into groups of four characters; VAR1="1234" ARY<2>="5678". See the **MSG** command for more examples of output text formatting and use of input variables.

COMM

PROGRAMING EXAMPLE:

```
(DVAR,VAR1,VAR2)                     ; Define variables VAR1 and VAR2.
(COMM,A,Aerotech)                    ; This sends "Aerotech" through
                                     ; the RS-232, Port A.
(COMM,B,Aerotech)                    ; This sends "Aerotech" through
                                     ; the RS-232, Port B.
(COMM,B,#C:H,0D0A)                   ; This transmits carriage returns
                                     ; and line feed on RS-232, Port B.
(COMM,A,<VAR1>, "NEXT")              ; Character data is sent and input data
                                     ; returned on the RS-232, Port A.
(MSG,#C:VAR1)                        ; Print the character format variable
                                     ; VAR1 to the screen.
(COMM,A,<VAR2>, input 4 numbers)     ; Data is sent and data returned
                                     ; on the RS-232, Port A.
(MSG,#VAR2)                          ; Print the decimal format variable
                                     ; to the screen.
M0                                   ; Program Stop
```

NOTES:

1) Prior to inputting data, adequate variable names must be used to accommodate an input string.

2) Echo status for RS-232 input data is established within the Parameter Mode. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

3) Variables may contain 4 alphabetic characters maximum and may be strung together to larger input strings.

4) The Unidex 21 will wait until data is received before proceeding with the next program block. Data validity is not checked. However, if necessary, this may be done using a separate command(s).

RELATED COMMANDS:

DVAR, MSG

CPAG

NAME:

CPAG - Customer Display Page

FUNCTION:

The **CPAG** command provides the User the ability to replace the Machine/Debug display with a Customer Designed Display Page. This can be useful to provide menu driven operation with highly visible and customized prompts. When a page is active you can toggle it on/off by typing <alt H> to return to the normal machine mode display. Note that memory must be allocated via the **MALC** command before trying to create **CPAG**.

FORMAT:

(**CPAG**,option 1,option 2,....)

Options may be either a variable or an array.

(**CPAG**,0) ; Deactivates the Customer Display Page and returns to the Machine/Debug Display.

(**CPAG**,1) ; Activates the Customer Display Page.

(**CPAG**,2,x1,y1,x2,y2,color) ; Draws a box of the specified color with <x1,y1> as the upper left corner and <x2,y2> as the lower right corner.

x : 0 to 79 (80 columns)
y: 0 to 23 (24 rows)

color code	background	text
0	black	white
1	blue	yellow
2	green	magenta
3	cyan	red
4	red	cyan
5	magenta	green
6	yellow	blue
7	white	black

CPAG

FORMAT (CON'T):

(**CPAG**,3,x1,y1,x2,y2,char) ; This draws a box filled with the specified character with <x1, y1> as the upper left corner, and <x2, y2> as the lower right corner. The character is fixed (i.e., cannot be a variable or array element).

(**CPAG**,4,x1,y1,string) ; Starts from x1, y1. Display string is in 1x1 format. The string is fixed and can't be a variable or array element.

(**CPAG**,5,x1,y1,string) ; Starts from x1, y1. Display string is in 2x2 format. The string is fixed and can't be a variable or array element.

(**CPAG**,6,x1,y1,Variable/Array) ; Starts from x1,y1 to display the contents of variable or array element in 1x1 format.

(**CPAG**,7,x1,y1,Variable/Array) ; Starts from x1,y1 to display the contents of variable or array element in 2x2 format.

(**CPAG**,8,set,case,format,size,adjust,x1,y1,system variable) ; Enables a 200 msec periodic update of information.

; Set: 1 to mmm (see MALC)
; Case: 1 1x1 character size
; 2 2x2 character size

0 disables this set of functions, no code follows, so (**CPAG**,8,set,0) is a complete code

Format: 1 – unsigned Hex number

0 – unsigned Binary number

1 – BCD, 1 digit after decimal point

2 – BCD, 2 digits after decimal point

3 – BCD, 3 digits after decimal point

N – BCD, N digits after decimal point

CPAG

FORMAT (CON'T):

	; Size: space needed for character display (<127)
	; Adjust: 0 starts from left, with leading zero
	; <> 0 start from right no leading zero
	; x1, y1: start from this point
	; System variable: axis position or 1/0
(**CPAG**,9,set,case,y1,nn,mm)	; For 200 msec display of User's Parts Program
	; Set: 1 to mmm (see **MALC**)
	; Case: 0-disable, <> 0 enable
	; y1: current block display
	; Line # on which to display current block
	; nn: number of blocks above current block
	; mm: number of blocks below current block

0 disables this set of functions, so (**CPAG**,9,set,0) is a complete code.

RETURNS:

EXAMPLE:

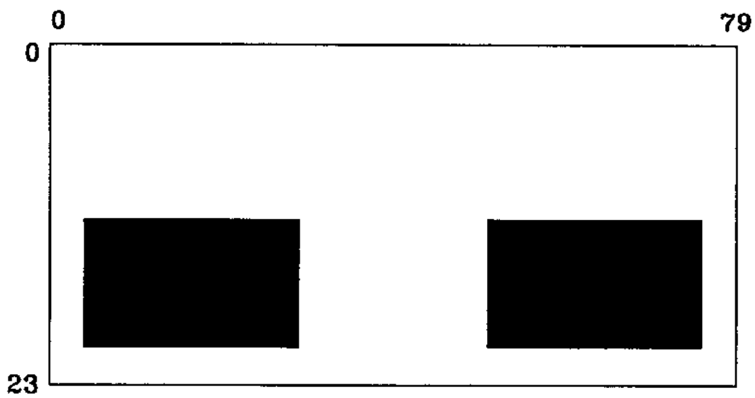
PROGRAMMING EXAMPLE #1:

The following is an example illustrating the functionality of the **CPAG** command showing the effect on a shadow of the screen. The actual image will not be displayed until the Customer Defined Page is activated.

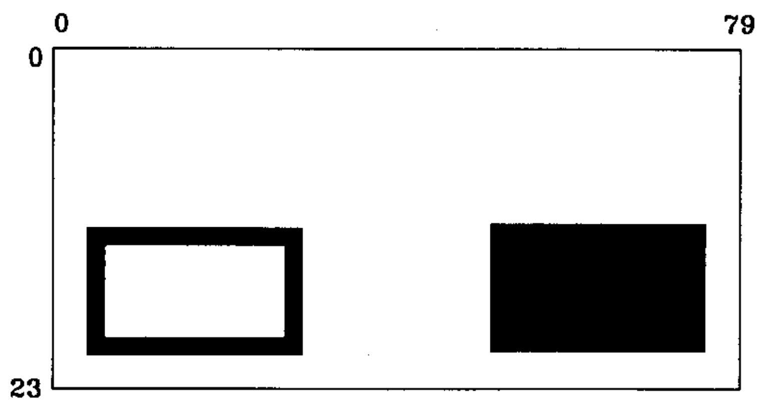
(DVAR,VAR1)	; The variable VAR1 holds the interrupt count.
(**CPAG**,2,4,12,26,20,7)	; A box will be drawn in reverse video with corners at
	; 4,12, and 24,20.
(**CPAG**,2,55,12,75,20,7)	; An additional box will be drawn with corners at 55,12
	; and 75,20.
	; Refer to the figure on the following page.

CPAG

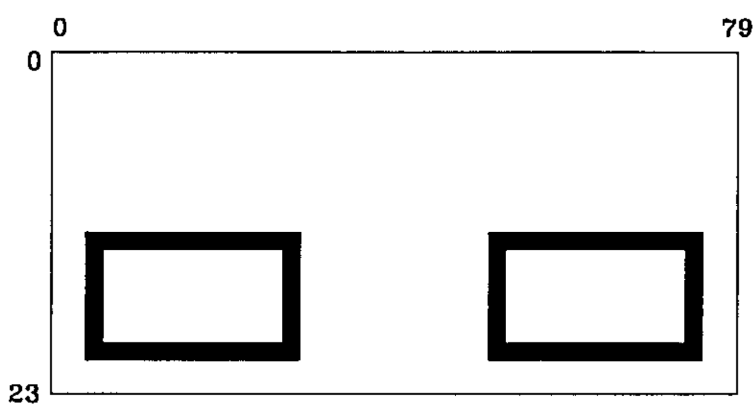
PROGRAMMING EXAMPLE #1 (CON'T):



(**CPAG**,2,6,14,24,18,0) ; A normal colored box is drawn within the 1st box,
; (see the figure below)



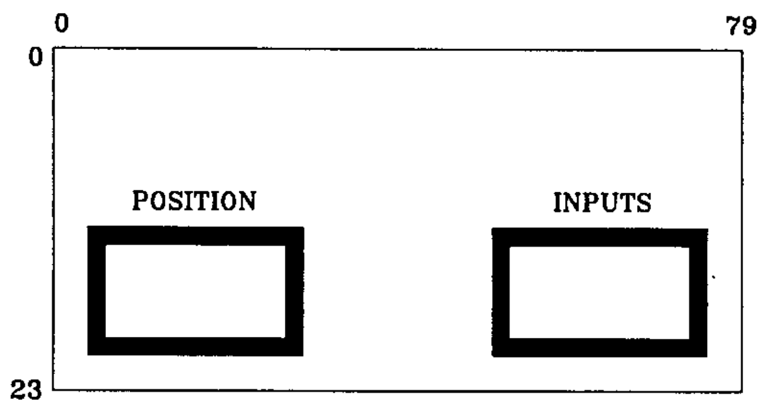
(**CPAG**,2,59,15,71,17,0) ; and another within the 2nd box.
; (See the Figure below.)



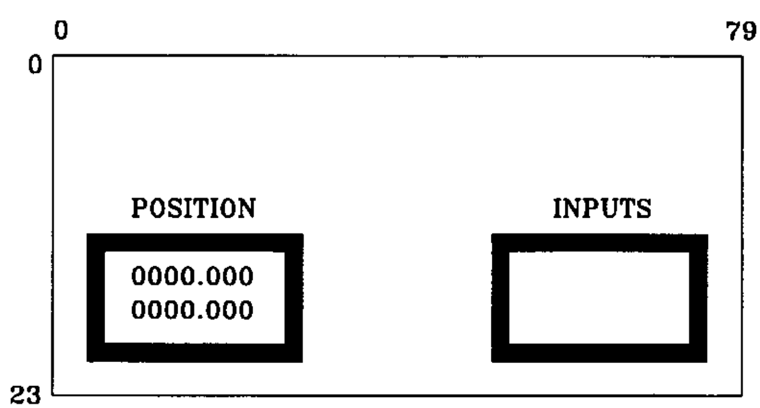
CPAG

PROGRAMMING EXAMPLE #1 (CON'T):

```
(CPAG,5,8,10,POSITION)              ; Display the word position (2x2 format) above
                                    ; the 1st box, and I/O above the second box
(CPAG,5,60,10,INPUTS)               ; (same format).
                                    ; See the Figure below.
```



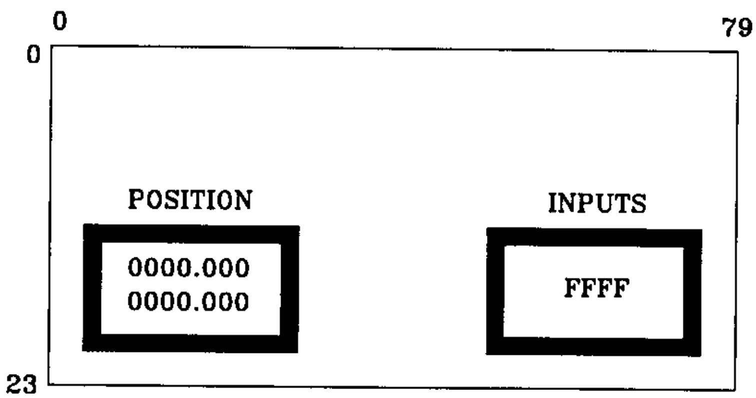

```
G92 X0. Y0.                         ; Reset all positions to 000.000.
(MALC,<1,4>)                        ; Allocate memory for 4 CPAG items.
(CPAG,8,1,2,3,14,0,6,15,$XRP)       ; Display X axis Relative Position at 6,15.  Use 2x2
                                    ; format, BCD with 3 digits after decimal point, and
                                    ; left justified (leading 0's).
(CPAG,8,2,3,14,0,6,17,$YRP)         ; Display Y axis Relative Position at 6,17.  Format is
                                    ; the same.  Both are updated every 200 msec.
                                    ; See the Figure below.
```



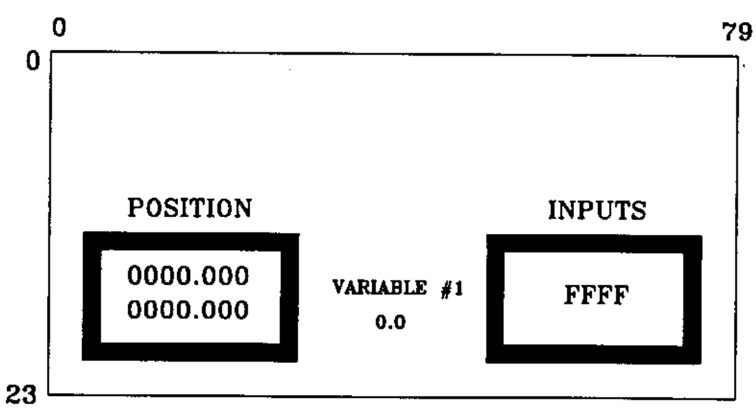
CPAG

PROGRAMMING EXAMPLE #1 (CON'T):

```
(CPAG,8,3,2,-1,4,1,62,16,$INP)    ; Display current inputs in hexadecimal at location
                                  ; 62,16. Uses 2x2 format. Right justified, maximum
                                  ; of 4 digits. Also updated every 200 msec.
                                  ; See the Figure below.
```




```
(CPAG,4,35,15,VAR1)               ; Display VAR1 at location 35,15 and the current
(CPAG,6,37,17,VAR1)               ; of VAR1 in 1x1 format at 37,17.
                                  ; (BCD with N digits after decimal)
                                  ; See the Figure below.
```

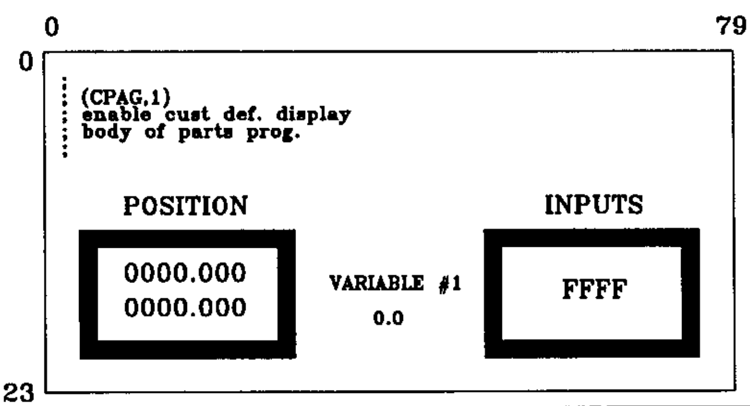



```
(CPAG,9,4,1,5,2,3)                ; Display the Parts Program with current line on line 5. Two
                                  ; lines will be displayed above, and three lines below.
                                  ; See the Figure below.
```

CPAG

PROGRAMMING EXAMPLE #1 (CON'T):

```
(CPAG,1)        ; Enable the Customer Defined Display body of the Parts
                ; Program. X and Y Relative Position will be
                ; continuously updated on display as will the inputs and
                ; outputs. The value of variable VAR1 will be updated
                ; only if the program issues another CPAG command.
(CPAG,0)        ; Disable the Customer Defined Display. Display normal
                ; Machine Mode screen again.
M2              ; End of The Program.
```



PROGRAMMING EXAMPLE #2:

This example shows the various methods of using the **CPAG** command to display text, system variables, User variables, and the currently executing Parts Program. It also demonstrates how the program may receive input from the User while using the Customer Defined Page screen.

```
(MALC,<1,9>)                    ;
(CPAG,2,0,0,79,27,7)            ;
(CPAG,2,0,0,79,8,1)             ;
(CPAG,3,0,0,79,27, )            ;
(CPAG,5,1,1,HORZ)               ;
(CPAG,8,1,2,4,10,1,9,1,$XRP)    ;
(CPAG,5,1,3,VERT)               ;
(CPAG,8,2,2,4,10,1,9,3,$YRP)    ;
```

CPAG

PROGRAMMING EXAMPLE #2 (CON'T):

```
(CPAG,5,1,5,ROTY)                              ;
(CPAG,8,3,2,4,10,1,9,5,$ZRP)                   ;
(CPAG,5,1,7,PROB)                              ;
(CPAG,8,4,2,4,10,1,9,7,$URP)                   ;
(CPAG,5,41,1,AZIM)                             ;
(CPAG,8,5,2,4,10,1,49,1,$xRP)                  ;
(CPAG,5,41,3,PLC1)                             ;
(CPAG,8,6,2,4,10,1,49,3,$INP)                  ;
(CPAG,5,41,5,PLC2)                             ;
(CPAG,8,7,2,4,10,1,49,5,$INP)                  ;
(CPAG,5,41,7,PLC3)                             ;
(CPAG,8,8,2,4,10,1,49,7,$ZAP)                  ;
(CPAG,2,0,11,79,22,4)                          ;
(CPAG,2,0,16,79,16,2)                          ;
(CPAG,9,9,1,16,5,6)                            ;
(CPAG,5,3,24,LASER POWER=2.35Kw)               ;
(CPAG,5,3,26,LASER GAS :ON ;SHUTTER :OPEN)     ;
(CPAG,1)                                       ;
M0                                             ;
```

HORIZ	.5298	AZIM	.0787
VERT	.5271	PLC1	.9769
ROTY	.3314	PLC2	.9042
PROB	.6254	PLC3	.3792

```
(CPAG,2,0,16,79,16,2)                      ;
(CPAG,9,9,1,16,5,6)                        ;
(CPAG,5,3,24,LASER  POWER=2.35Kw)          ;
(CPAG,5,3,26,LASER  GAS  :ON  :SHUTTER  :OPEN)  ;
(CPAG,1)                                   ;
M0                                         ;
```

CPAG

NOTES:

1) A 200 msec Customer Display requires a Memory Allocation **MALC** prior to initiation.

2) Each data item is to be updated every 200 msec and must have a unique "set" number.

3) The applicable Main System Parameter is #50 (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode).

4) The set parameter in **CPAG** options 8 and 9 associates the chosen system variable with a memory set allocated with (MALC,1,n).

RELATED COMMANDS:

DARY, DVAR, MALC

DARY

NAME:

DARY - Define Array

FUNCTION:

The **DARY** command is used to Define an Array. The array may have been created to save information into a data file for ease of element retrieval. Also, a table of either on/off points or individual fine points may be created for usage by the PSO option card.

FORMAT:

(**DARY**,array name 1 and size,array name 2 and size,....)

RETURNS:

EXAMPLE:

```
(DARY,AR1<n>,AR2<m>,..)        ; Defines n+1 elements of array named AR1
                                 and m+1 elements of array named AR2.
                                 ("n" and "m" non zero, positive are fixed
                                 integers. The array consists of "0" to "n/m"
                                 elements.)

(DARY,AR1<10>)                 ; Defines 11 elements of array named "AR1".
                                 The 11 programmed elements of the array
                                 may be referenced as AR1<0> through
                                 AR1<10>,

(DARY,AR1<3>)
AR1<0>=1
AR1<1>=AR1<0>+1
AR1<2>=AR1<1>*2
AR1<3>=10
(MSG, elements 0-3 of AR1 are: #AR1<0>, #AR1<1>, #AR1<2>, #AR1<3>)
```

In this example, the message display on the Unidex 21 would present:

"elements 0-3 of AR1 are: 1,2,4,10"

DARY

NOTES:

1) Array size is enclosed in < > not ( ).

2) By allocating n + 1 elements for an array, this command allows the programmer to number the first element of the array as "0" (0-(n-1) or as "1" (1-n) according to preference.

3) All elements of the array are initialized to zero.

4) The minimum array size has two elements: (DARY,ARY<1>) and contains elements ARY<0> and ARY<1>.

RELATED COMMANDS:

$R, DVAR, END, MEND, OPEN, WRIT

DENT

NAME:

DENT - Define Entry Point

FUNCTION:

The **DENT** command is used to Define an Entry Point for a **JUMP** or **INT1/INT2** command.

FORMAT:

(**DENT**,entry point name)

The entry point name may be 2 to 4 characters in length. The first two characters must be A-Z, the second two may be any alphanumeric characters although they cannot conflict with system commands.

RETURNS:

EXAMPLE:

```
(DENT,ENT1)
```

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples # 1, 2, 3, and 7 located in Appendix 3 of this manual.

NOTES:

1) The name used to Define an Entry Point must be a unique identification that is not used anywhere else in the program in a definition, and it cannot be a User variable.

2) The **DENT** command should occupy its own block within a program.

RELATED COMMANDS:

ABTS, INT1/INT2, JUMP

DFLS

NAME:

DFLS - Define Library Subroutine

FUNCTION:

The **DFLS** command is used to define a Library Subroutine. Defining a Library Subroutine is done in the same manner as defining a Subroutine. A Library Subroutine however, restores previous modal machine states upon it's completion (see Notes).

A Library Subroutine may be called by a **CLS** or **INT1/INT2** command.

FORMAT:

```
(DFLS,subroutine name
 .
 .
 .
)
```

The Library Subroutine name may be 2 to 4 characters in length. The first two characters of the Library Subroutine name must be letters, A-Z, the second two may be any alphanumeric characters.

RETURNS:

The program will continue with the next program block following a **CLS** command or **INT1/INT2** call.

EXAMPLE:

```
M2
(DFLS,SB12
 S300 M3
 Z-400. F100.
 M5
 G4 F0.3
 M4
 Z500.
)
```

DFLS

NOTES:

1) The **DFLS** command should occupy its own block within a program and be placed at the end of the Main Program following an **M2** command.

2) A program calling a Library Subroutine will not have its modal information altered after the Subroutine is executed. The modal commands which are reinstated after a Subroutine are:

   All modal **G** and **H** codes
   DRUN
   F
   MIR
   RTRS
   SCF
   SIOC
   UMFO

3) A **DFLS** command requires 8 bytes of User stack memory.

4) The name used to define a Subroutine must be a unique identification that is not used anywhere else in the program in a definition, and it cannot be a User variable. However, a **CLS** Call Subroutine command can be set up as a variable.

RELATED COMMANDS:

ABTS, CLS, DFS, INT1/INT2

DFS

NAME:

DFS - Define Subroutine

FUNCTION:

The **DFS** command is used to Define a Subroutine. Every Subroutine must be explicitly defined by a **DFS** command.

FORMAT:

```
(DFS,subroutine name
 .
 .
 .
)
```

The Subroutine name may be 2 to 4 characters in length. The first two characters of the Subroutine name must be letters, A-Z, the second two must be alphanumeric characters.

RETURNS:

The program will continue with the next program block following a **CLS** command or **INT1/INT2** call.

EXAMPLE:

```
M2
(DFS,TAP1
 S500 M3
 Z-1000. F200.
 M5
 G4 F0.2
 M4
 Z1000.
)
```

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #4 located in Appendix 3 of this manual.

DFS

NOTES:

1) The **DFS** command should occupy its own block within a program and be placed at the end of the Main Program following an **M2** command.

2) The modal information of the program calling the Subroutine may be altered after execution of the Subroutine.

3) Subroutines may call other subroutines.

4) Each **DFS** command requires 8 bytes of User stack memory.

5) The name used to define a Subroutine must be a unique identification that is not used anywhere else in the program in a definition and it cannot be a User variable. However, a **CLS** Call Subroutine command can be set up as a variable.

RELATED COMMANDS:

ABTS, CLS, DFLS, INT1/INT2

DRUN

NAME:

DRUN - Dry Run

FUNCTION:

The **DRUN** command is used to enable or disable a Dry Run on an individual axis. Enabling an axis for Dry Run prevents that axis from performing program moves. This is useful for debugging a program i.e., disabling Z axis while permitting X and Y.

FORMAT:

(**DRUN**,axis name and enable/disable, axis name and enable/disable,...)

The Dry Run function is enabled by placing a 1 (non-zero) next to the axis name.

The Dry Run function is disabled by placing a 0 next to the axis name.

RETURNS:

EXAMPLE:

(**DRUN**,X0,Y1)	; Disables a Dry Run on the X axis, and enables a Dry Run on the Y axis.
Then, upon entering the program:	
G2 G17 X10. Y10. I10. F100.	; Only the X axis will move.
(**DRUN**,variable)	; If the variable is 0, Dry Run is disabled; if the variable is non-zero, Dry Run is enabled.

NOTES:

1) All other commands are processed as usual.

2) Similar results can be obtained by using the **MTOR** command.

RELATED COMMANDS:

MTOR

DVAR

NAME:

DVAR - Define Variable

FUNCTION:

The **DVAR** command is used to Define a Variable. The ability to utilize variables is probably the single most powerful programming feature. Almost every command and function may be set and changed through the use of variables. This enhances Parts Program versatility and saves program memory space. There are two types of variables, System Variables and Program Variables. System Variables are denoted by the **$nnn** system and are utilized for I/O, Position Registers, and general machine functions. The System Variables name may not be changed although the User can equate their value to another program variable. Program Variables are defined and utilized within a Parts Program, and must be defined prior to usage.

FORMAT:

(**DVAR**,variable name,variable name,...)

RETURNS:

EXAMPLE:

```
(DVAR,VAR1,VAR2,XYZ,BI34,...)
```

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #7 located in Appendix 3 of this manual.

NOTES:

1) The name used to Define a Variable must be a unique identification that is not used anywhere.

2) The variable name may range from 2 to 4 characters in length.

3) The first two characters of the variable name must be letters, A-Z, the second two may be any alphanumeric characters.

DVAR

NOTES (CON'T):

4) All variables defined occupy 4 bytes of User memory. These 4 bytes are initialized to zero, and can be used to hold either numeric or character data.

RELATED COMMANDS:

$ System Variables, CLS, COMM, CPAG, DARY, JUMP, MALC, TERM

DZON

NAME:

DZON - Define Safe Zone

FUNCTION:

The **DZON** command is used to Define the number of Safe Zones to be included within a program. Safe Zones define areas where axes can operate. An attempt to operate outside a Safe Zone will generate an error message and stop axis motion (see **ZONE** command).

FORMAT:

(**DZON**,n)	; The number of Safe Zones entered must be an integer, (not a variable or an array) ranging from 1 to 65,535, and should appear only one time at the beginning of the program. Internally, this number becomes n+1 (0 to n) sets of Safe Zones.

RETURNS:

EXAMPLE:

(DZON,15)	; "16" sets of Safe Zones are defined in the program following this command.

NOTES:

1) If the number of Safe Zones entered by this command is less than the actual number of Safe Zones contained in the program, the following error message will appear when the program begins to run:

STATUS: error, undefined Safe Zone

2) The number of Safe Zones entered by this command must include any Safe Zones defined in programs "joined" with this one.

3) Safe Zone limits are not in effect when motion is initiated using the **HOME**, **FREE**, and **JOG/SLEW** commands. The User may want to utilize **LIMT** instead.

RELATED COMMANDS:

LIMT, ZONE

ELPS

NAME:

ELPS - Ellipse Look-Up Table

FUNCTION:

The Unidex 21 provides Elliptical Trajectory programming which is useful for laser hole cutting in a tubular structure or x-y flat sheet processing. The ellipse is executed at constant surface speed as opposed to other controllers which scale circles such that the resultant velocity varies widely. The **ELPS** command is used to generate a Look-Up Table to be used when performing an Elliptical Trajectory. At the present time, two Look-Up Tables may be stored in memory on the Unidex 21's Indexer board. The Look-Up Table is necessary to maintain constant surface speed while performing elliptical motion. The **ELPS** command may take several seconds for the initial completion of the Look-Up Table, after which execution as called by the **G2**, and **G3** commands occurs immediately.

FORMAT:

(**ELPS**,g,Xa,Yb)	g=	the ellipse identification number. This number must be used to reference this ellipse.
	X=	represents the axis name from which conversion factors will be referenced
	a=	length of the horizontal axis in program units
	Y=	represents the axis name from which conversion factors will be referenced
	b=	length of the vertical axis in program units

RETURNS:

EXAMPLE:

(ELPS,1,X10.,Y20.)	; A Look-Up Table will be generated for Ellipse #1 with a horizontal axis length of 10 program units, and a vertical axis length of 20 program units.

Refer to the **G2** command for a programming example of drawing an ellipse.

ELPS

PROGRAMMING EXAMPLE:

```
; ********************************************ELLIP
(REF,X,Y)                    ; Establishes an X and Y Home position.
G70                          ; Initiate English Programming (inches).
G91                          ; Initiate Incremental Positioning.
G1 X10. Y35. F200.           ; Linear Offset Move.
(ELPS,1,X10.,Y20.)           ; Generate an Ellipse Look-Up Table.
G2,1,45,90,90,X0. Y0. F100.  ; Initiate a G2 Ellipse command.
M0                           ; Program Stop
; ********************************************SMALL ELLIPSE
(REF,X,Y)                    ; Establishes an X and Y Home position.
G70                          ; Initiate English programming (inches)
G91                          ; Initiate Incremental Positioning.
G1 X10. Y35. F200.           ; Linear Offset Move.
(ELPS,1,X2.,Y4.)             ; Generate an Ellipse Look-Up Table.
G2,1,45,90,90,X0. Y0. F100.  ; Initiate a G2 Ellipse command.
M0                           ; Program Stop
```

NOTES:

1) The **ELPS** command needs to be executed only once for a given ellipse. If the major and/or minor axes lengths change, the command must be reentered.

2) The Look-Up Table established for a given ellipse is retained in memory on the Unidex 21 Indexer board , even through a power down, until changed.

3) Actual elliptical motion is executed via the **G2** and **G3** commands.

4) Minor block execution delays are possible when partial ellipses are executed or if parts rotation is utilized.

RELATED COMMANDS:

G2/G12/H2/H12, G3/G13/H3/H13

F

NAME:

F - Define Axis Feedrate

FUNCTION:

The **F** command is used to Define the Axis Feedrate in inches (**G70**) or millimeters (**G71**) per minute. Rotary axes will be specified in units per minute. The **F** command indicates the vectorial Feedrate of a move and applies to **G0**, **G1**, **G2**, **G3**, **G5**, and **FREE** axis commands.

FORMAT:

Feedrate values may be designated in two ways:

without a "."	**F1234**	; The system always places a decimal point two digits from right. The Feedrate will be 12.34 inches or millimeters per minute.
with a "."	**F1234.**	; The system interprets the value as written. The Feedrate will be 1,234 inches or millimeters per minute.

RETURNS:

EXAMPLE:

```
G71 X10. F1000          ; The X axis will move 10 millimeters at a Feedrate of 10.00
                          millimeters per minute.
G70 X10. F1000.         ; The X axis will move 10 inches at a Feedrate of 1,000 inches
                          per minute.
(DVAR,FEED)
FEED=250.0
F=FEED                  ; The Feedrate is equal to 250 program units per minute.
```

NOTES:

1) The **F** command is modal. User Manual Feedrate Override (**MFO**) adjusts this modal Feedrate. The only exceptions are **G0** moves, where **MFO** is limited to a maximum 100% override.

NOTES (CON'T):

2) The programmed Feedrate is in units per minute and is dependent upon the setting of parameters #0, 1, 2, and 3 established under the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

RELATED COMMANDS:

$MFO, DVAR, FREE, G70, G71, UMFO

FILT

NAME:

FILT - Digital Filter

FUNCTION:

The **FILT** command is used to enable or override digital filtering which can occur on the millisecond position/velocity trajectory command being sent to the Unidex 21's DSP Servo Control card. This command is modal, and remains until changed or the system is reset.

The Unidex 21 provides a high performance PIDF Proportional, Integral, Derivative Servo Loop that also implements two feedforward gains: velocity and acceleration. As a result, the system follows (tracks) very closely to the actual command with minimal servo following error or lag. When moves are programmed with tangency, i.e., corners are programmed with small arcs, the Unidex 21 will smoothly follow the path under **G8** Velocity Profiling.

However, there exists a large base of CNC's which utilize an error based type of servo loop closure. In this case, programmed moves must not necessarily have tangency and corners but can be programmed as two linear moves. When executed under **G23** Corner Rounding Mode, these systems will inherently round the corners and provide relatively continuous velocity contouring even with non-tangent moves. Please refer to **G8**, **G9**, **G23**, and **G24** program commands for further information.

By utilizing the **FILT** command and associated Main System Parameters #60 and #61, the Unidex 21 is able to digitally replicate more traditional CNC type control functionality. This is of particular benefit for Smart 1 and Unidex 16 users, who may more easily utilize their programs on the Unidex 21. Also, CAD/CAM systems with post processor converters are more readily compatible with the Unidex 21.

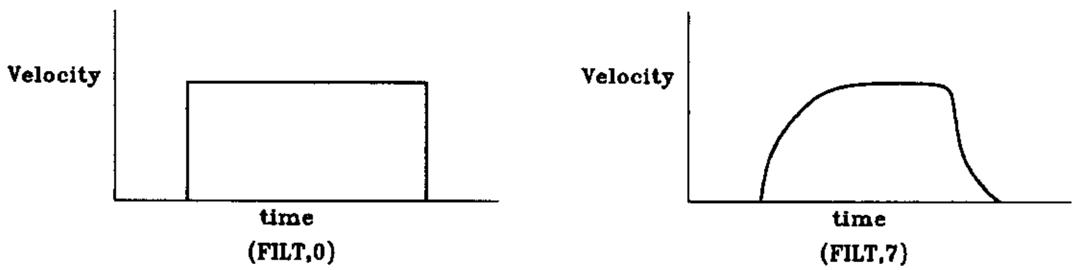
The digital filter works under the following relationships:

Filter Total	= Filter Total + Command In
Command Out	= Filter Total/2^n; where "n" equals FILT#
Filter Total	= Filter Total - Command Out

This provides an exponential ramp to the motion, eliminating the need for **RAMP**. However, the User may select to program these as a supplement.

FILT

The values for **FILT** range from 0 (no filter) to 7 (maximum filter). The extreme spectrum would be:



point-to-point move assuming no Acceleration/Deceleration commands.

For normal systems a filter value of 4, 5, or 6 usually provides the best performance.

FORMAT:

(**FILT**,n)	; Where "n" ranges from 0-7

RETURNS:

EXAMPLES:

(FILT,0)	; Turn OFF the Digital Filter.
(FILT,VAR1)	; Set the Digital Filter to the value of VAR1. (range 0-7)

NOTES:

1) The Digital Filter cannot be turned OFF while the axes are moving or a sharp velocity transition may result. It should be turned OFF only after **G9** or **G24** type motion.

2) To disable axis ramping when operating in the **FILT** mode, axis parameter #61 must be set to "NO" (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode).

RELATED COMMANDS:

FREE

NAME:

FREE - Axis Free Run (enabled or disabled)

FUNCTION:

The **FREE** command is used to enable or disable Axis Free Run. This command is sent directly to the Unidex 21's DSP Servo Control card for execution. Therefore, no synchronization between axis contouring and the Free Run axis is possible. Once started, the Free Run command can cause motion to occur asynchronously from the User Parts Program flow. The Free Run command block execution time is 1ms and may be inserted between normal motion blocks to speed up program flow.

FORMAT:

(**FREE**,axis name and case, F ± Feedrate, D distance)

The code is selected as follows:

code 0 – stop Free Run

1 – one direction Free Run, with F + (CW), F - (CCW) Feedrate

2 – one shot Free Run, with F + (CW), F - (CCW) Feedrate and D distance

3+ n – back and forth Free Run, with F + (CW), F - (CCW) Feedrate and D distance
if n=0, Free Run back and forth an infinite number of times
if n=1 to 16777215, Free Run back and forth n times

(**FREE**,D,axis,MST data)	; When Axis Free Run is complete, output **M,S,T** data, at most four **M,S,T** commands.
(**FREE**,D,axis)	; Disable the **M,S,T** output function.
(**FREE**,W,axis1,axis2,...)	; Wait for completion of the axes Free Run before proceeding with the next program block.

RETURNS:

FREE

EXAMPLES:

(FREE,X1,F100.)	; Free Run is enabled for the X axis at a Feedrate of 100 inches or millimeters per minute in the CW (+) direction.
(FREE,X1,F-100.)	; Free Run is enabled for the X axis at a Feedrate of 100 inches or millimeters per minute in the CCW (-) direction.
(FREE,X0)	; Free Run is stopped for the X axis.
(FREE,Y=VAR1,F=VAR2)	; Free Run status and Feedrate are contained in variables VAR1 and VAR2.
(FREE,X2,F100.,D20.)	; A one shot Free Run is enabled for the X axis at a Feedrate of 100 inches or millimeters per minute in the CW (+) direction for a distance of 20 inches or millimeters.
(FREE,X3,F100.,D30.)	; A back and forth type Free Run is enabled for the X axis at a Feedrate of 100 inches or millimeters per minute starting in the CW (+) direction for a distance of 30 inches or millimeters.
(FREE,X3,F-100.,D30.)	; Same as above except moves CCW (-) direction first.
(FREE,X3+5,F-100.,D30.)	; Same as above except moves back and forth five times.
(FREE,D,X,M80,S2048,T101,M91)	; Upon completion of the X axis Free Run output M80, S2048, T101, and M91.

FREE

NOTES:

1) The **FREE** command is not modal. The CW (+) and CCW (-) direction are as specified in the axis setup of the Parameter Mode.

2) Case: 1 Free Run makes use of ramp time.

3) Case: 2 and 3 Free Run use **ACDE** rate or the Individual Axis Parameter #9 value as established under the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

4) The Feedrate specified is only used during the Free Run and has no effect on the modal Feedrate parameter.

5) Manual Feedrate Override and Feedhold may be set up to also affect the Free Run axes. Refer to parameters #19 and 36 as established under the Individual Axis Parameters 401-408 grouping. (See also the *Unidex 21 User's Manual* for further detail on these parameters.)

RELATED COMMANDS:

ACDE, COEF, DVAR, F, RAMP, TRAJ

FXOF

NAME:

FXOF - Fixture Offset

FUNCTION:

The **FXOF** command provides the User the ability to move the axes to a new coordinate frame to accommodate a Fixture Offset. This position is an absolute distance from the Hardware Home (including any offsets) and automatically updates the **$nAP** Position Registers, while keeping the **$nRP** Position Registers at current values.

FORMAT:

(**FXOF**,axis name and new position,axis name and new position,....)

RETURNS:

$nAP Position Register value changes to reflect the new position as referenced to the Hardware Home Position; however, the **$nRP** Position Register value remains unchanged.

EXAMPLE:

```
(FXOF,X10.,Y=VAR1,Z=ARY<3>)          ; Moves the X, Y, and Z axes to a new coordi-
                                       nate frame located at (10.0,VAR1,ARY<3>).
```

NOTES:

1) All positions are referenced to the Hardware Home.

2) The axis speed is established in the "Top Feedrate steps/sec" Parameter #8 of the Individual Axis Parameters 401-408 grouping (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode).

3) The **FXOF** command executes point-to-point motion only, no contouring.

4) See the **$nAP/$nRP** command description and program example for further explanation.

RELATED COMMANDS:

$nAP, $nRP, G0, MORG, ROTA

G0

NAME:

G0 - Point-to-Point Positioning at rapid traverse rate

FUNCTION:

The **G0** command sets up axis movement for point-to-point positioning at a rapid traverse rate. The rate is determined by the "Top Feedrate counts/sec" Parameter #8 as established under the Individual Axis Parameters 401-408 grouping (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode). The Acceleration/Deceleration Ramping Trajectory for **G0** is always parabolic.

FORMAT:

G0 axis name and move distance

RETURNS:

EXAMPLE:

```
G0 X10.          ; Moves to X10. using top system Feedrate.

G0 X5. Y10.      ;
```

NOTES:

1) The **G0** command is modal.

2) The **G8** command cannot be used in conjunction with the **G0** command.

3) The **G0** feedrate is limited to 100% MFO maximum.

4) The applicable Individual Axis Parameters are #8, 9, and 38 as established under the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

RELATED COMMANDS:

COEF, FXOF, MORG

G1/G11/H1/H11

NAME:

G1/G11/H1/H11 - Linear Contouring

FUNCTION:

The **G1/G11/H1/H11** commands initiate Linear Contouring. As an example, if a move of X1. Y1. were programmed under **G0** motion, the axes would perform that move at their respective top feedrates uncoordinated. **G1** ensures that the axes will move synchronization at the programmed vectorial feedrate.

FORMAT:

G1 or **G11** or **H1** or **H11**

RETURNS:

EXAMPLE:

G17 G90 G70 G1 X4.0 Y3.0 F50.	; A straight line will be produced from the initial position to the X=4.0, Y=3.0 coordinate position, at a Feedrate of 50 inches per minute.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples # 1, 2, 5, and 6 located in Appendix 3 of this manual.

NOTES:

1) Prior to selecting a code for Linear Contouring, the appropriate axes/plane relationship must have been established **(G17/H17, G18/H18, G19/H19)**.

2) The following code/plane relationships exist:

G1 - 1st plane axes	**H1** - 3rd plane axes
G11 - 2nd plane axes	**H11** - 4th plane axes

G1/G11/H1/H11

NOTES (CON'T):

3) The Contour Feedrate command (**F**) must be used in conjunction with the **G1/G11/H1/H11** commands.

4) The **G1/G11/H1/H11** commands are modal.

5) When transferring from lines to arcs and back again the User must reestablish **G1** mode of operation

   or

   a "circle missed center point" error will result.

RELATED COMMANDS:

F, FILT, G17/H17, G18/H18, G19/H19, G70, G71, G90, G91, RAMP

G2/G12/H2/H12

NAME:

G2/G12/H2/H12 - CW Circular Contouring

FUNCTION:

The **G2/G12/H2/H12** commands initiate Circular Contouring in the CW direction.

FORMAT:

G2 or **G12** or **H2** or **H12**

RETURNS:

EXAMPLE:

G17 G90 G70 **G2** X4.0 Y3.0 I2.0 J1.5 F50.	; A CW arc will be produced from the initial position to the X=4.0, Y=3.0 coordinate position, the center points being 2.0 and 1.5 (offset from starting point) at a Feedrate of 50 inches per minute.
G2 L10.0 C35.0 D70.0 F50.	; A CW arc with a 10 inch radius will be produced from the initial position of 35° and ending at 70°.

For more detailed examples, see **I,J,K,P/i,j,k,p** and **LCD,OAB/lcd,oab** commands.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #1 located in Appendix 3 of this manual.

NOTES:

1) Prior to selecting a code for Circular Contouring, the appropriate axes/plane relationship must have been established (**G17/H17, G18/H18, G19/H19**).

2) The following code/plane relationships exist:

G2 - 1st plane axes	**H2** - 3rd plane axes
G12 - 2nd plane axes	**H12** - 4th plane axes

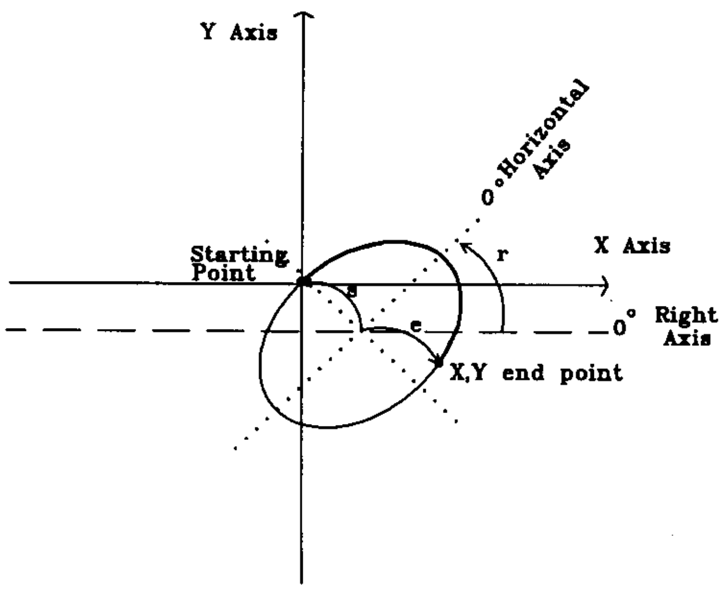
G2/G12/H2/H12

NOTES (CON'T):

3) Helical Interpolation is accomplished by programming two axes to do Circular Interpolation and a third axis to do Linear Interpolation.

4) Spherical Interpolation is accomplished by programming two axes to do Circular Interpolation and then programming the third and a hidden fourth axis to also do Circular Interpolation.

5) An Elliptical Contour is accomplished using the Circular Contouring command in conjunction with the following special code:
G,g,r,s,e,Zn Xa Yb Zd F



G– specifies axis plane and direction of rotation **(G2/G3/G12/G13/H2/H3/H12/H13)**

g – ellipse identification number (See **ELPS** command)

r – rotation angle of ellipse (in degrees) with respect to the right axis 0°

s – starting angle of ellipse (in degrees) with respect to the horizontal axis 0°

G2/G12/H2/H12

NOTES (CON'T):

e – ending angle of ellipse (in degrees) with respect to the horizontal axis 0°

Z – third axis name (omit if not used)

n – the options for n are as follows:

0 - third axis motion is not synchronized with axes performing the ellipse

1 - third axis moves with constant speed and is synchronized with axes performing the ellipse

2 - third axis moves proportionally to the angular velocity of the axes performing the ellipse

3 - third axes follows a tangent to the axes performing the ellipse

Xa Yb – axis name and calculated endpoint of the move (calculated by User). Note that if both end points (a,b) are zero, a full ellipse that begins and ends with the starting point will be generated.

d – distance of the third axis move

F – surface velocity of the ellipse

An example of Elliptical Contouring follows:

```
G2,1,45,90,90,Z3 X0. Y0. Z10. F100.    ; A CW full ellipse will be produced using
                                         Look-Up Table 1, with a 45° angle of rotation,
                                         starting and ending at 90° with a tangentially
                                         synchronized third axis.
```

6) It is not necessary for both axes involved in a Circular Contour to have the same resolution.

G2/G12/H2/H12

NOTES (CON'T):

7) The Contour Feedrate command (**F**) must be used in conjunction with the **G2/G12/H2/H12** commands.

8) The **G2/G12/H2/H12** commands are modal.

9) When transferring from lines to arcs and back again the User must reestablish **G1** mode of operation

   or

   a "circle missed center point" error will result.

RELATED COMMANDS:

ELPS, **F**, **G3/G13/H3/H13**, **G17/H17**, **G18/H18**, **G19/H19**
I/J/K/P and **i/j/k/p**,
L/C/D, **O/A/B** and **l/c/d**, **o/a/b**,
PLNE

G3/G13/H3/H13

NAME:

G3/G13/H3/H13 - CCW Circular Contouring

FUNCTION:

The **G3/G13/H3/H13** commands initiate Circular Contouring in the CCW direction.

FORMAT:

G3 or **G13** or **H3** or **H13**

RETURNS:

EXAMPLE:

G17 G90 G70 **G3** X4.0 Y3.0 I2.0 J1.5 F50.	; A CCW arc will be produced from the initial position to the X=4.0, Y=3.0 coordinate position, the center points being 2.0 and 1.5 (offset from starting point) at a Feedrate of 50 inches per minute.
G3 L12.0 C90.0 D180.0 F50.	; A CCW arc will be produced starting at 90°, ending at 180° at a Feedrate of 50 inches per minute.

For more detailed examples, see **I/J/K/P**, **i/j/k/p** and **LCD**, **OAB**, **lcd,oab** commands.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #1 located in Appendix 3 of this manual.

NOTES:

1) Prior to selecting a code for Circular Contouring, the appropriate axes/plane relationship must have been established (**G17/H17, G18/H18, G19/H19**).

2) The following code/plane relationships exist:

G3 - 1st plane axes	**H3** - 3rd plane axes
G13 - 2nd plane axes	**H13** - 4th plane axes

G3/G13/H3/H13

NOTES (CON'T):

3) Helical Interpolation is accomplished by programming two axes to do Circular Interpolation and a third axis to do Linear Interpolation.

4) Spherical Interpolation is accomplished by programming two axes to do Circular Interpolation and then programming the third and a hidden fourth axis to also do Circular Interpolation.

5) An Elliptical Contour is accomplished using the Circular Contouring command in conjunction with a special code. (Refer to Note 5 of the **G2** command for a description and example of the Elliptical Contour code.)

6) It is not necessary for both axes involved in a Circular Contour to have the same resolution.

7) The Contour Feedrate command (**F**) must be used in conjunction with the **G3/G13/H3/H13** commands.

8) The **G3/G13/H3/H13** commands are modal.

9) When transferring from lines to arcs and back again the User must reestablish **G1** mode of operation

   or

   a "circle missed center point" error will result.

RELATED COMMANDS:

ELPS, **F**, **G17/H17**, **G18/H18**, **G19/H19**,
I/J/K/P and **i/j/k/p**,
L/C/D, **O/A/B** and **l/c/d**, **o/a/b**,
PLNE

G4

NAME:

G4 - Dwell

FUNCTION:

The Dwell command provides a time delay of a programmed or established duration. The **G4** command must be immediately followed by an **F** command to establish the Dwell duration. The resolution of the **G4** command is 0.1 seconds.

FORMAT:

G4 Fnn

The Dwell duration may be entered in either decimal or integer form. If the Dwell duration value is in decimal form (followed by a ".") or in the form of a variable, the Dwell will be in seconds. If the Dwell duration value is in integer form (not followed by a ".") the Dwell will be in tenths of seconds.

RETURNS:

EXAMPLE:

G4 F5.	; A Dwell of 5 seconds is established.
G4 F.5 or **G4** F5.	; A Dwell of 5/10 (1/2) second is established.
G4 F=VAR1	; The Dwell time is set to the value of variable VAR1.
G4 F=ARY<n>	; The Dwell time is located in ARY<n>.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #2, 3, and 4 located in Appendix 3 of this manual.

G4

NOTES:

1) The Dwell command must occupy its own block within a program.

2) System interrupts or aborts are not active during dwells but will be executed immediately following the Dwell completion.

RELATED COMMANDS:

G5/G15/H5/H15

NAME:

G5/G15/H5/H15 - Three Point Interpolation of two dimensional arc for contouring

FUNCTION:

The **G5/G15/H5/H15** commands initiate Three Point Interpolation of a two dimensional arc for contouring. Utilizing these commands will alert the Unidex 21 that the values of **I/J/K/P** and **i/j/k/p** represent points on the arc, not center points. The arc direction is internally derived. The **G5/G15/H5/H15** command is utilized when digitized data is acquired from the Trackball/Joystick/Teachpendant options.

FORMAT:

G5 or **G15** or **H5** or **H15**

RETURNS:

System automatically resets to **G2/G3**, **H2/H3**, **G12/G13**, or **H12/H13** mode.

EXAMPLE:

G17 G91 G70 **G5** X4.0 Y0. I2.0 J2.0 F100.	; A CW arc will be produced from the initial position to the X=4.0, Y=0 coordinate position crossing points I=2.0, J=2.0 at a Feedrate of 100 inches per minute.

PROGRAMMING EXAMPLE:

(REF,X,Y)	; Establish an X and Y Home position.
G70	; Initiate English Programming (inches).
G91	; Initiate Incremental Positioning.
G1 X2. Y2. F200.	; Initiate a Linear Offset Move.
G5 X4. Y0. I2. J2. F100.	; Perform Three Point Interpolation.
M0	; Program Stop.

G5/G15/H5/H15

NOTES:

1) The following code/plane relationships exist:

G5 - 1st plane axes	**H5** - 3rd plane axes
G15 - 2nd plane axes	**H15** - 4th plane axes

2) Helical Interpolation is accomplished by programming two axes to do Circular Interpolation and a third axis to do Linear Interpolation.

3) Spherical Interpolation is accomplished by programming two axes to do Circular Interpolation and then programming the third and a hidden fourth axis to also do Circular Interpolation.

4) To indicate the axes involved in the Circular Interpolation, enter a **G17/H17**, **G18/H18**, **G19/H19**, or **PLNE** command prior to the **G5/G15/H5/H15** command.

5) It is not necessary for both axes involved in a Circular Contour to have the same resolution.

6) **G5/G15/H5/H15** are not modal commands.

7) When transferring from lines to arcs and back again the User must reestablish **G1** mode of operation

   or

   a "circle missed center point" error will result.

RELATED COMMANDS:

G17/H17, **G18/H18**, **G19/H19**,
I/J/K/P and **i/j/k/p**,
PLNE

G8

NAME:

G8 - Velocity Profiling (acceleration)

FUNCTION:

The **G8** command is used for Velocity Profiling. During Velocity Profiling, priority is given to attaining and maintaining the programmed Feedrate of the current motion command. This command also permits M-function output on-the-fly, and provides an accuracy of within 1ms of the commanded output.

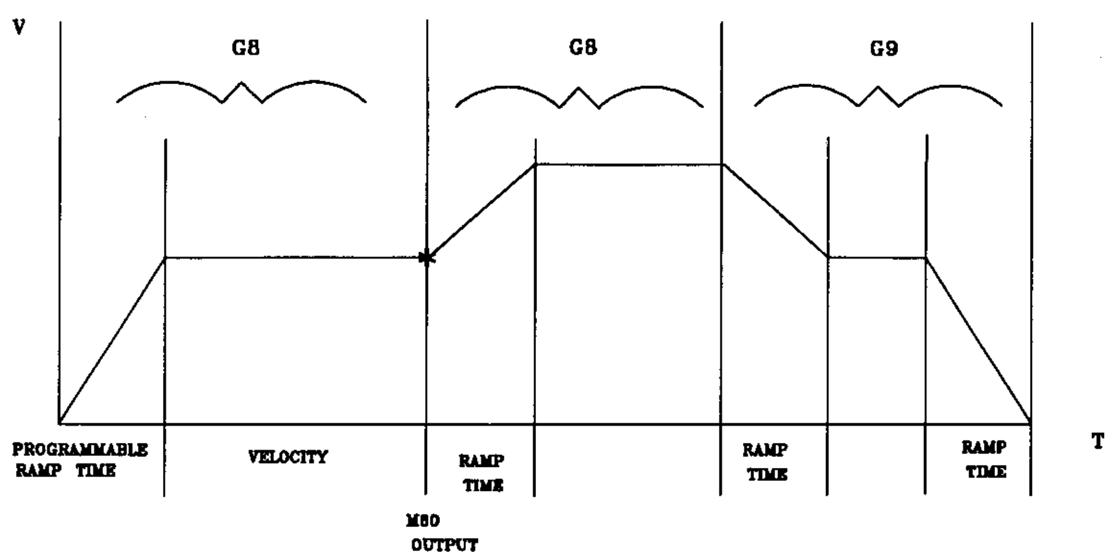
FORMAT:

G8

RETURNS:

EXAMPLE:

```
G24                        ; Use Non-Corner Rounding mode of operation.
G1 G8 X1. F100. M80        ; At the end of X travel, the axis maintains contouring velocity
                             with no deceleration and activates M80 at the end of the 1
                             inch move.
G1 G8 X1. F200.
G1 G9 X1. F100.            ; At the conclusion of G9, move deceleration to zero velocity.
```



G8

NOTES:

1) If the **G8** command is not used, the default provides for both Acceleration and Deceleration to zero velocity.

2) The **G8** command is used for contouring only. If **G0** has been previously programmed the **G8** command will be ignored.

3) The first **G8** command (i.e, after a previously executed **G9** command) will have a beginning vector velocity of zero. Subsequent **G8** commands will be executed beginning at the vector velocity reached by the previous **G8** command.

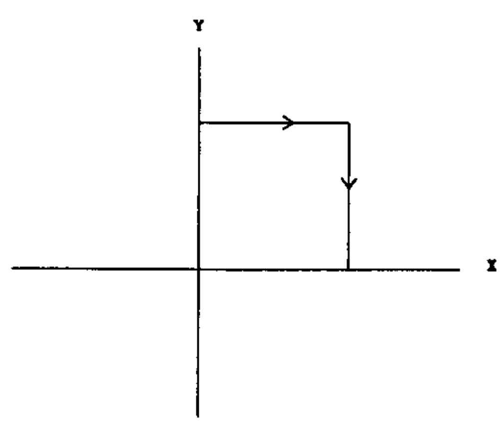
4) The Acceleration/Deceleration mechanism used is dependent upon the current setting of the **G23/G24** modal parameter. (Refer to Appendix 2 "Corner Rounding/Velocity Profiling" for further detail.)

5) To maintain smooth transitions from block-to-block motion, the **G8** command requires that consecutive moves be tangent to each other.

6) Velocity Profiling may be used to produce a non-tangential shape by using arcs to initiate transitional moves. See the following example.

Velocity Profiling **cannot** be maintained smoothly.

G1 **G8** X4.
G1 G9 Y-4.

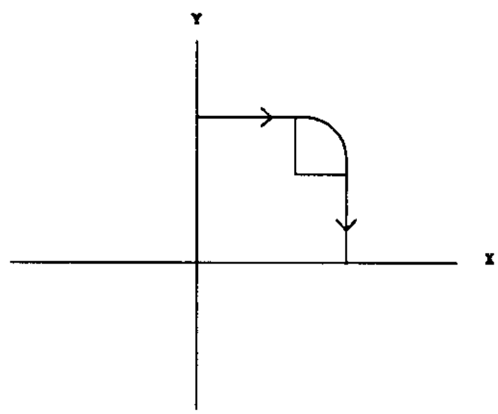


G8

NOTES (CON'T):

Velocity Profiling will be maintained smoothly.

G1 **G8** X4.
G2 **G8** X1. Y-1. J-1
G1 G9 Y-4.



RELATED COMMANDS:

COEF, FILT, G9, G23, G24, MST, RAMP, TRAJ

G9

NAME:

G9 - Velocity Profiling (deceleration)

FUNCTION:

The **G9** command is used in conjunction with the **G8** command to provide controlled deceleration to zero velocity after Velocity Profiling. The motion executed under this mode is guaranteed to feed the servo control loop(s) the final position of the specified command.

FORMAT:

G9

RETURNS:

EXAMPLE:

See the example under the **G8** command.

NOTES:

1) A **G9** command not preceded by a **G8** command will have a beginning and ending vector velocity of zero.

2) A **G9** command preceded by a **G8** command has a beginning vector velocity equal to the ending velocity of the previous **G8** command. The ending velocity of the **G9** movement will be zero.

3) The Acceleration/Deceleration mechanism used is dependent upon the current setting of the **G23/G24** modal parameter. (Refer to Appendix 2 "Corner Rounding/Velocity Profiling" for further detail.)

RELATED COMMANDS:

COEF, FILT, G8, G23, G24, RAMP, TRAJ

G17/H17

NAME:

G17/H17 - Axis Plane Designation for execution of Circular Interpolation

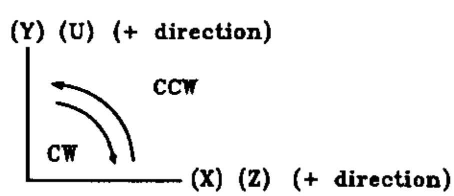
FUNCTION:

The **G17/H17** command specifies the axis plane for the execution of Circular Interpolation where:

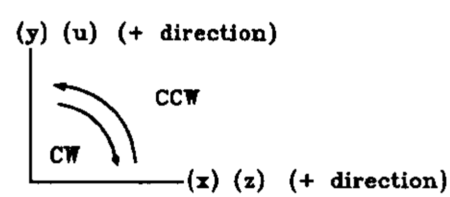
X/Y axes is the 1st contouring plane (**G1/G2/G3/G5**)
Z/U axes is the 2nd contouring plane (**G11/G12/G13/G15**)
x/y axes is the 3rd contouring plane (**H1/H2/H3/H5**)
z/u axes is the 4th contouring plane (**H11/H12/H13/H15**)

This permits the User to contour between different groups of axes. Refer to the illustration below for **G17/H17** rotational direction.

G17



H17



FORMAT:

G17 or **H17**

RETURNS:

EXAMPLE:

G17 G2 X4.0 Y0. I2.0 J0. F100.	; A CW arc is performed on the XY plane.

G17/H17

NOTES:

1) The **G17/H17** command is modal and may be specified once at the beginning of the program or on the same line as the contour. The exception is to utilize Cutter Compensation, where the contour plane must be specified.

2) The **G17/H17** command is a default.

RELATED COMMANDS:

G1/G11/H1/H11, G2 /G12/H2/H12, G3/G13/H3/H13, G5/G15/H5/H15, G18/H18, G19/H19, PLNE

G18/H18

NAME:

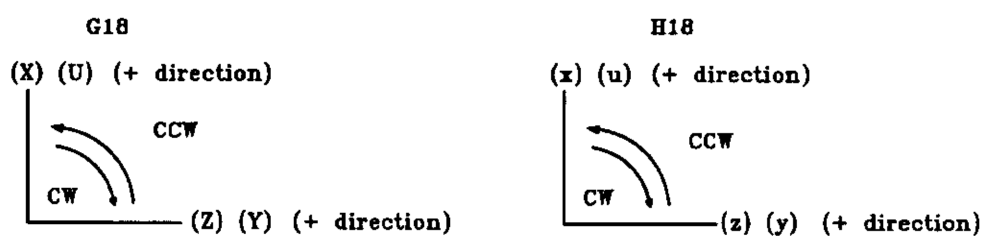
G18/H18 - Axis Plane Designation for execution of Circular Interpolation

FUNCTION:

The **G18/H18** command specifies the axis plane for the execution of Circular Interpolation where:

Z/X axes is the 1st contouring plane **(G1/G2/G3/G5)**
Y/U axes is the 2nd contouring plane **(G11/G12/G13/G15)**
z/x axes is the 3rd contouring plane **(H1/H2/H3/H5)**
y/u axes is the 4th contouring plane **(H11/H12/H13/H15)**

This permits the User to contour between different groups of axes. Refer to the illustration below for **G18/H18** rotational direction.



FORMAT:

G18 or **H18**

RETURNS:

EXAMPLE:

G18 G2 Z4.0 X0. K2.0 I0. F100.	; A CW arc is performed on the ZX plane.

G18/H18

NOTES:

1) The **G18/H18** command is modal and may be specified once at the beginning of the program or on the same line as the contour. This exception is to utilize Cutter Compensation, where the contour plane must be specified.

2) The **G17/H17** command is the default.

RELATED COMMANDS:

G1/G11/H1/H11, G2 /G12/H2/H12, G3/G13/H3/H13, G5/G15/H5/H15, G17/H17, G19/H19, PLNE

G19/H19

NAME:

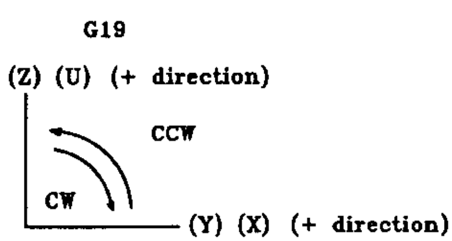
G19/H19 - Axis Plane Designation for execution of Circular Interpolation

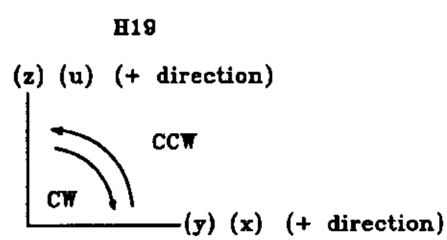
FUNCTION:

The **G19/H19** command specifies the axis plane for the execution of Circular Interpolation where:

Y/Z axes is the 1st contouring plane **(G1/G2/G3/G5)**
X/U axes is the 2nd contouring plane **(G11/G12/G13/G15)**
y/z axes is the 3rd contouring plane **(H1/H2/H3/H5)**
x/u axes is the 4th contouring plane **(H11/H12/H13/H15)**

This permits the User to contour between different groups of axes. Refer to the illustration below for **G19/H19** rotational direction.





FORMAT:

G19 or **H19**

RETURNS:

EXAMPLE:

```
G19 G2 Y4.0 Z0. J2.0 K0. F100.          ; A CW arc is performed on the YZ plane.
```

G19/H19

NOTES:

1) The **G19/H19** command is modal and may be specified once at the beginning of the program or on the same line as the contour. The exception is to utilize Cutter Compensation, where the contour plane must be specified.

2) The **G17/H17** command is the default.

RELATED COMMANDS:

G1/G11/H1/H11, G2 /G12/H2/H12, G3/G13/H3/H13, G5/G15/H5/H15, G17/H17, G18/H18, PLNE

G23

NAME:

G23 - Corner Rounding

FUNCTION:

There are two modes of operation for which a motion program may execute. The first ensures that the axes reach their final position before beginning the next motion block, accelerating and decelerating to the end point. This is known as the Non-Corner Rounding Mode (**G24**) and utilizes a Unidex 21 Indexer board "block complete" and "in position" handshake.

The second mode is **G23**, where the system does not wait for the "in position" handshake for the Unidex 21 Indexer board before executing the next block of motion. This is utilized for more traditional machining operations where relatively constant surface speed must be maintained. **G23** is referred to as Corner Rounding due to the effect that is seen when motion commands are executed in sequence prior to the full execution of the previous command.

In both cases, the **RAMP** (Defines Axis Ramping time) and **FILT** (Digital Filter) commands can influence performance and may be utilized in conjunction with the **G23/G24** modes.

The amount of Corner Rounding is influenced by the **RAMP**, **FILT**, and servo loop settings, and, of course, their related parameter settings. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.) **G23** provides the ability to program "non-tangential" moves possibly resulting in smooth motion when used with the proper **FILT/RAMP** settings. This is different from **G24** motion where move tangency considerations should be observed. See the individual above referenced commands for further discussion and explanation.

FORMAT:

G23

RETURNS:

EXAMPLE:

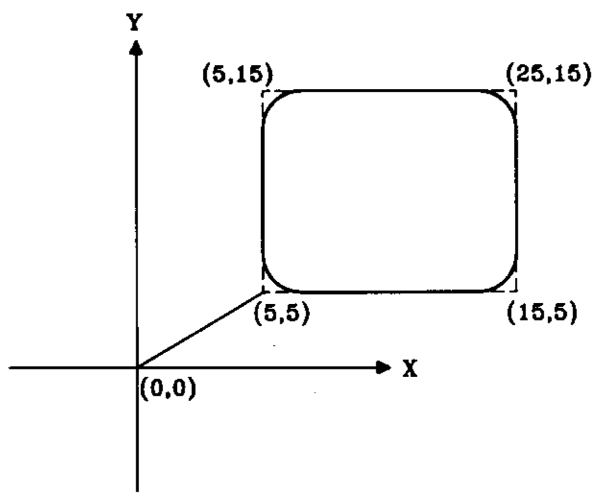
```
G70     ; Initiate English Programming (inches).
G91     ; Initiate Incremental Positioning.
G23     ; Use Corner Rounding mode of operation.
```

G23

EXAMPLE (CON'T):

F100.	; A Feedrate of 100 inches per minute is established.
X10.	; Move the X axis 10 inches in the positive direction.
Y10.	; Move the Y axis 10 inches in the positive direction.
X-10.	; Move the X axis 10 inches in the negative direction.
Y-10.	; Move the Y axis 10 inches in the negative direction.

The above example is illustrated below.



PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #6 located in Appendix 3 of this manual.

NOTES:

1) The **G23** command is modal.

2) The **G24** command is the default.

3) Refer to the Corner Rounding/Velocity Profiling Appendix for further discussion.

RELATED COMMANDS:

FILT, G8, G9, G24, RAMP, SIOC

G24

NAME:

G24 - Non-Corner Rounding

FUNCTION:

There are two modes of operation for which a motion program may execute. The first ensures that the axes reach their final position before beginning the next motion block, accelerating and decelerating to the end point. This is known as the Non-Corner Rounding Mode (**G24**) and utilizes a Unidex 21 Indexer board "block complete" and "in position" handshake.

The second mode is **G23**, where the system does not wait for the "in position" handshake for the Unidex 21 Indexer board before executing the next block of motion. This is utilized for more traditional machining operations where relatively constant surface speed must be maintained. **G23** is referred to as Corner Rounding due to the effect that is seen when motion commands are executed in sequence prior to the full execution of the previous command.

In both cases, the **RAMP** (Defines Axis Ramping time) and **FILT** (Digital Filter) commands can influence performance and may be utilized in conjunction with the **G23/G24** modes.

NOTE: The **FILT** command has no effect on **G24** motion.

The execution of **G24** is influenced by the **RAMP** and servo loop gain settings, and, of course, their related parameter settings. (See the Unidex *21 User's Manual* for a detailed description of the Parameter Mode.) Refer to the **G8** and **G9** commands in addition to the individual above referenced commands for further discussion and explanation.

FORMAT:

G24

RETURNS:

EXAMPLE:

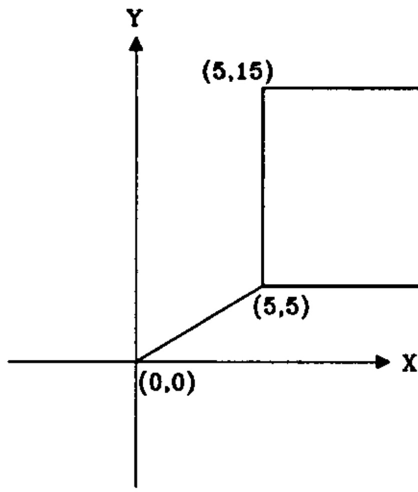
G70	; Initiate English Programming (inches).
G91	; Initiate Incremental Positioning.
G24	; Use Non-Corner Rounding mode of operation.

G24

EXAMPLE (CON'T):

```
F100.      ; A Feedrate of 100 inches per minute is established.
X10.       ; Move the X axis 10 inches in the positive direction.
Y10.       ; Move the Y axis 10 inches in the positive direction.
X-10.      ; Move the X axis 10 inches in the negative direction.
Y-10.      ; Move the Y axis 10 inches in the negative direction.
```

The above example is illustrated below.



PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #2 and 6 located in Appendix 3 of this manual.

NOTES:

1) The **G24** command is modal.

2) The **G24** command is the default.

3) Refer to the Corner Rounding/Velocity Profiling Appendix for further discussion.

RELATED COMMANDS:

FILT, G8, G9, G23, RAMP, SIOC

G40

NAME:

G40 - Deactivate Cutter Radius Compensation

FUNCTION:

The **G40** command deactivates Cutter Radius Compensation.

FORMAT:

G40 and end move

RETURNS:

EXAMPLE:

G40 X1. Y1. F100.	; Deactivate the Cutter Radius Compensation and perform the end move.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #5 located in Appendix 3 of this manual.

NOTES:

1) The **G40** command is modal.

2) The **G40** command is the default.

3) The **G40** command must be followed by an end move.

4) For more complete examples regarding the use of **G40** with **CCP**, **G41**, and **G42** refer to the Cutter Compensation Appendix.

RELATED COMMANDS:

CCP, G41, G42

G41

NAME:

G41 - Activate Cutter Radius Compensation - Left

FUNCTION:

The **G41** command initiates Cutter Radius Compensation left. The cutter is offset to the left side of the work piece. Left/right is relative to the direction in which the cutter is moving.

FORMAT:

G41

RETURNS:

EXAMPLE:

G41 X2. Y2. F100.	; Activate the Cutter Radius Compensation left.

PROGRAMMING EXAMPLE:

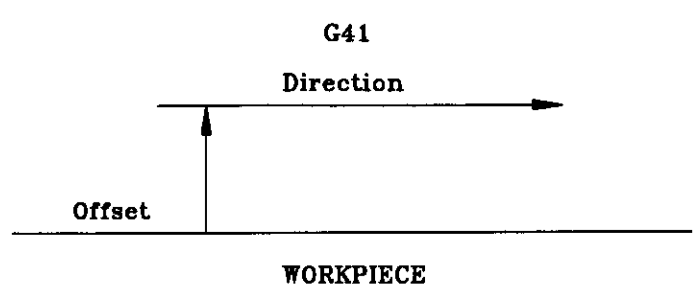
For an example of how this command may be used, refer to Programming Example #5 located in Appendix 3 of this manual.

NOTES:

1) The **G41** command is modal.
2) The **G40** command is the default.
3) For more complete examples regarding the use of **G41** with **CCP**, **G40**, and **G42** refer to the Cutter Compensation Appendix.

RELATED COMMANDS:

CCP, **G40**, **G42**



G42

NAME:

G42 - Activate Cutter Radius Compensation - Right

FUNCTION:

The **G42** command initiates Cutter Radius Compensation right. The cutter is offset to the right side of the work piece. Left/right is relative to the direction in which the cutter is moving.

FORMAT:

G42

RETURNS:

EXAMPLE:

G42 X2. Y2. F100.	; Activate the Cutter Radius Compensation right.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #5 located in Appendix 3 of this manual.

NOTES:

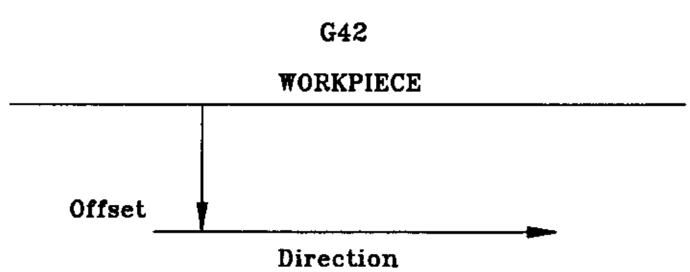
1) The **G42** command is modal.

2) The **G40** command is the default.

3) For more complete examples regarding the use of **G42** with **CCP**, **G40**, and **G41** refer to the Cutter Compensation Appendix.

RELATED COMMANDS:

CCP, G40, G41



G70

NAME:

G70 - English Programming

FUNCTION:

The **G70** command initiates English Programming. All dimensional and offset commands will be represented in inches.

FORMAT:

G70

RETURNS:

EXAMPLE:

G70	; Initiate English Programming (inches).
X10.	; Move the X axis 10 inches in the positive direction.
Y5.	; Move the Y axis 5 inches in the positive direction.
F100.	; A Feedrate of 100 inches per minute for the X and Y axes is established.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #1, 2, 5, and 6 located in Appendix 3 of this manual.

NOTES:

1) The **G70** command is modal.

2) The default command is established in the Parameter Mode and is dependent upon the system's mechanical resolution. For a detailed description of the parameters which set up units and default settings, refer to the General Machine Parameter #5, and the Individual Axis Parameters #0, 1, 2, and 3 as established under the 401-408 grouping of the *Unidex 21 User's Manual.*

RELATED COMMANDS:

G71

G71

NAME:

G71 - Metric Programming

FUNCTION:

The **G71** command initiates Metric Programming. All dimensional and offset commands will be represented in millimeters.

FORMAT:

G71

RETURNS:

EXAMPLE:

G71	; Initiate Metric Programming (millimeters).
X4.	; Move the X axis 4 millimeters in the positive direction.
Y2.	; Move the Y axis 2 millimeters in the positive direction.
F100.	; A Feedrate of 100 millimeters per minute for the X and Y axes is established.

NOTES:

1) The **G71** command is modal.

2) The default command is established in the Parameter Mode and is dependent upon the system's mechanical resolution. For a detailed description of the parameters which set up units and default settings, refer to the General Machine Parameter #5, and the Individual Axis Parameters #0, 1, 2, and 3 as established under the 401-408 grouping of the *Unidex 21 Users Manual.*

RELATED COMMANDS:

G70

G90

NAME:

G90 - Absolute Position Programming

FUNCTION:

The **G90** command initiates programming in Absolute Position values.

FORMAT:

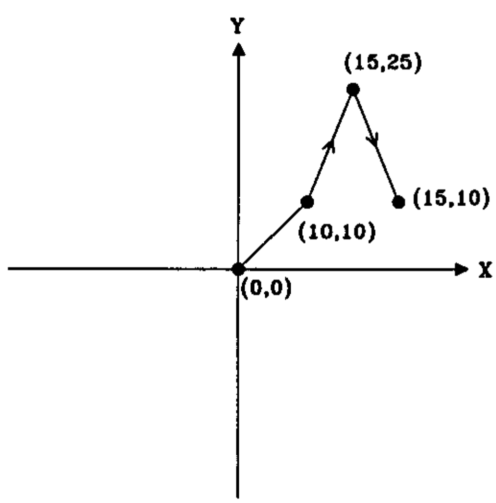
G90

RETURNS:

EXAMPLE:

G92 X0. Y0.	; A Software Register value of X=0, Y=0 is established.
G90	; Initiate Absolute Positioning.
X10.0 Y10.0	; Move the X and Y axes in increments of inches or millimeters from the current position to X=10.0, Y=10.0.
X15.0 Y25.0	; Move the X and Y axes in increments of inches or millimeters such that the current position is X=15.0, Y=25.0.
X15.0 Y10.0	; The X axis will not move, the Y axis will move in increments of inches or millimeters to Y=10.0.

The above example is illustrated below.



G90

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #1, 2, and 6 located in Appendix 3 of this manual.

NOTES:

1) The **G90** command is modal.

2) While in the **G90** (Absolute mode), if any of the following features are activated, a **G92** (Software Home) position must be reestablished prior to requesting any other moves:

AFCO, FREE, HWEL, MIR, SCF, SCO, SLEW

3) The **G91** Incremental Positioning mode is the default setting.

RELATED COMMANDS:

$nAP, $nRP, G91, G92

G91

NAME:

G91 - Incremental Position Programming

FUNCTION:

The **G91** command initiates programming in Incremental Position values.

FORMAT:

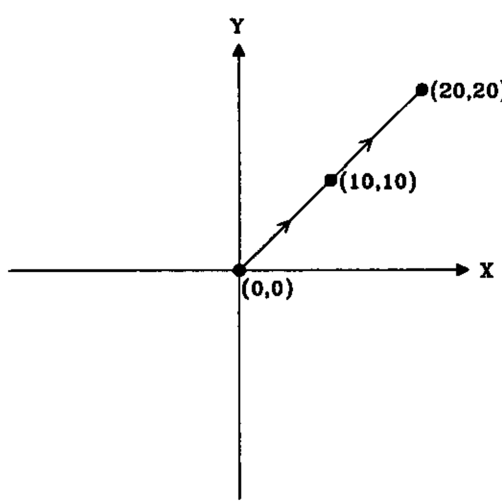
G91

RETURNS:

EXAMPLE:

G92 X0. Y0.	; A Software Register value of X=0, Y=0 is established.
G91	; Initiate Incremental Positioning.
X10. Y10.	; Move the X and Y axes 10 inches or millimeters from the current position to X=10.0, Y=10.0.
X10. Y10.	; Move the X and Y axes 10 inches or millimeters such that the current position is X=20.0, Y=20.0.

The above example is illustrated below.



G91

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #1, 2, and 5 located in Appendix 3 of this manual.

NOTES:

1) The **G91** command is modal.

2) The **G91** Incremental Positioning mode is the default command.

RELATED COMMANDS:

$nAP, $nRP, G90, G92

G92

NAME:

G92 - Software Home

FUNCTION:

The **G92** command establishes a Software Register Value for the **$nRP** Machine Position Registers at the current position of either all or the specified axes. The absolute and incremental **G90/G91** programming modes utilize the **$nRP** Position Registers for motion. The **G92** command is useful for setting position offsets, or returning the **$nRP** registers to **$nAP** values. The Machine Position Tracking Display is also affected by the **G92** command.

FORMAT:

G92

or

G92 specified axis name

RETURNS:

EXAMPLES:

G92	; Set all axes to the Software Register value of zero.
G92 X10. Y20.	; Set the Software Registers for the X and Y axes such that the current position for the X axis is 10 inches or millimeters and the current position for the Y axis is 20 inches or millimeters, the remaining axes Software Register values remain unchanged.
G92 X=VAR1, Y=YAP	; Set the X axis $nRP register to the value of variable VAR1, and the Y axis $nRP register to the $YAP value.

NOTES:

1) The **G92** command is not modal.

G92

NOTES (CON'T):

2) The **G92** command changes the value of the **$nRP** Machine Incremental Commanded Position Registers but not the value of the **$nAP** Machine Absolute Commanded Position Registers.

RELATED COMMANDS:

$nAP, $nRP, DVAR, G90, G91

HOME

NAME:

HOME - Hardware Home

FUNCTION:

The **HOME** and **REF** commands are the same. These commands are used to send any or all axes to the Hardware Home position. The axis first moves to the Home Limit Switch, and then moves out until the Home Marker is located. If a Home position is required that is different from the Home Marker position, a Home Offset position may be established within the Individual Axis Parameter settings (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode). The system utilizes two different Home feedrates for homing: Initial power on Home Feedrate or Normal Time Feedrate. This provides for rapid system homing after the Unidex 21's DSP Servo Control card has established the initial Home position. See figures 3-1 and 3-2 for an illustration of axis movement following a Home command.

FORMAT:

(**HOME**, axis name ,axis name,...)

RETURNS:

$nRP and **$nAP** cleared to zero

EXAMPLE:

```
(HOME,X,Y)                    ; The X and Y axes are sent Home simultaneously.
```

ADDITIONAL EXPLANATION:

Figure 3-1 illustrates the use of the Home command following a Power Up in which case the current axis position is unknown. The axis moves toward the Home limit at a constant Feedrate established in the "Power On Home Feedrate, steps/sec" Parameter #28 of the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

Upon reaching the Home Limit Switch, axis direction is reversed and acceleration occurs until the Home Feedrate is reached. This Feedrate is maintained until deceleration (after coming back out of the limit) occurs, completing the Marker Offset distance established in the "Home Limit to Marker, steps" of the Individual Axis Parameters 401-408 grouping. (See also the *Unidex 21 User's Manual* for further detail.)

HOME

ADDITIONAL EXPLANATION (CON'T):

When the Marker Offset distance has been completed, motion will continue at a rate of one-half machine count per millisecond until the Home Marker is reached. Acceleration occurs repeatedly until the Home Feedrate is reached. This Feedrate is maintained until deceleration occurs, completing the Home Offset distance established in Parameter #7 of the Individual Axis Parameters 401-408 grouping, and the Home position is reached. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

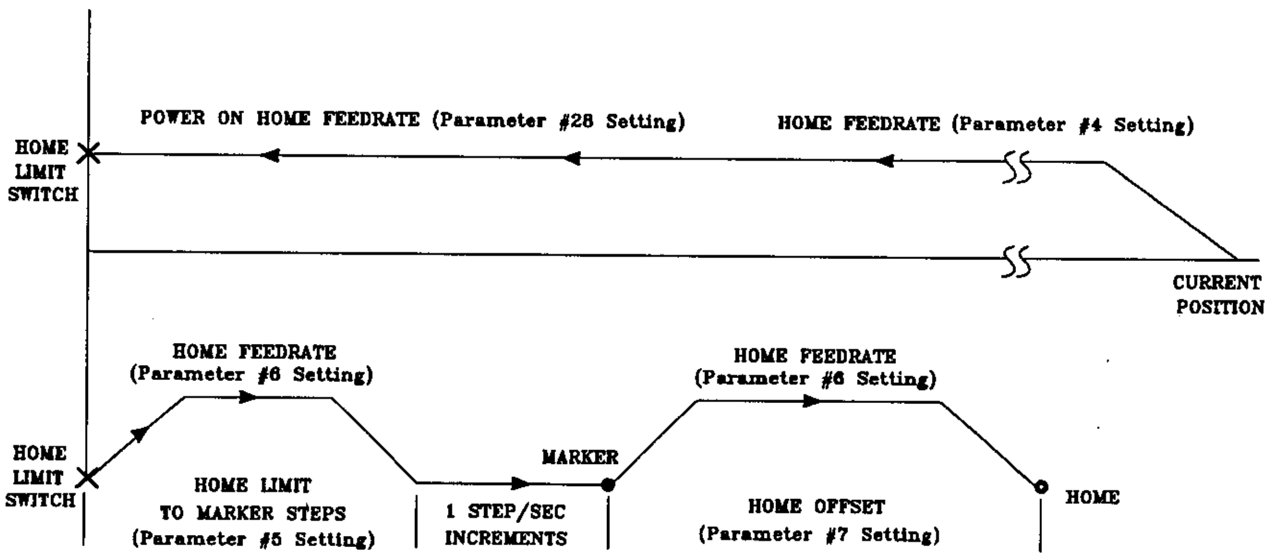


Figure 3-1: Home Command Following a Power Up

Figure 3-2 illustrates the use of a Home command from a known axis position after a Power Up Home command has been executed. The axis approaches the Home position at the Feedrate established in the "Home Feedrate, steps/sec" Parameter #6 of the Individual Axis Parameters 401-408 grouping. (See also the *Unidex 21 User's Manual* for further information.) Deceleration occurs as the axis approaches the Home Limit Switch. (The deceleration rate is 25% of the Individual Axis Parameter Setting-Ac/De,steps/sec^2.) Following deceleration, motion may continue in one step increments until the Home Limit Switch is reached.

Upon reaching the Home Limit Switch, axis direction is reversed and acceleration occurs until the Home Feedrate is reached. This Feedrate is maintained until deceleration occurs, completing the Marker Offset distance established in "Home Limit Marker, steps" parameter of the Individual Axis Parameters 401-408 grouping.

HOME

ADDITIONAL EXPLANATION (CON'T):

When the Marker Offset distance has been completed, motion will continue in one step increments until the Home Marker is reached. Acceleration occurs repeatedly until the Home Feedrate is reached. This Feedrate is maintained until deceleration occurs, completing the Home Offset distance established in the "Home Offset, steps" Parameter #7 of the Individual Axis Parameters 401-408 grouping.

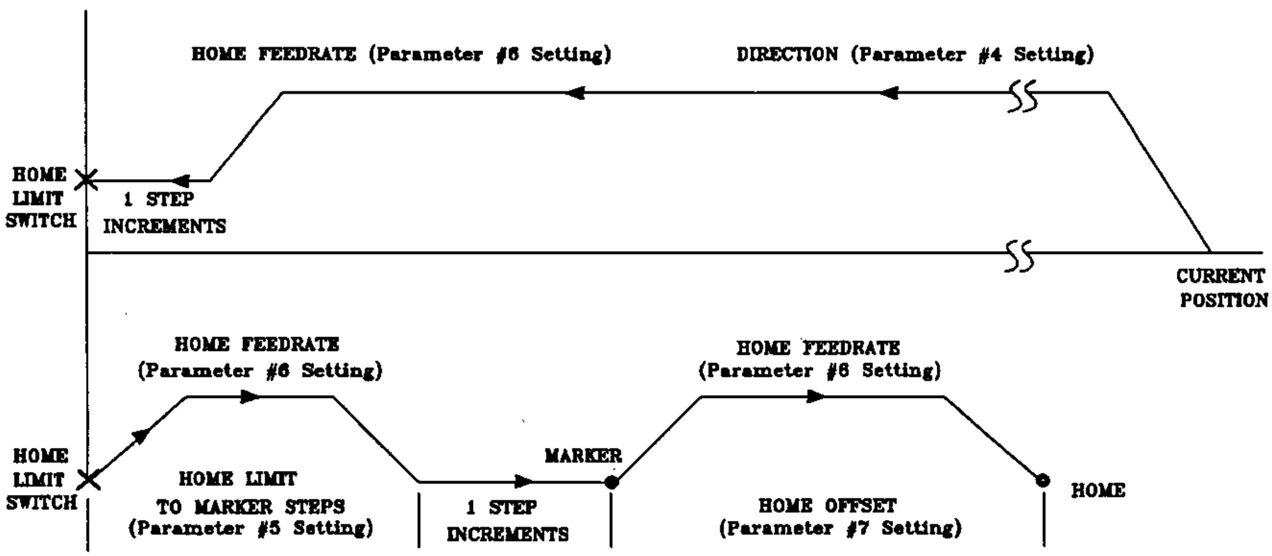


Figure 3-2: Home Command From a Known Position

NOTES:

1) The Home Feedrate may be adjusted without entering the Parameter Mode by using the **MFO** Manual Feed Override.

2) On systems with linear encoders or very fine resolution, it may be necessary to insert a Home Offset or Home Limit to marker steps via Parameter Mode in order to decrease the time for execution of a Home cycle.

3) The applicable parameters are #4, 5, 6, 7, 9, 26, 27, and 28 established in the Individual Axis Parameters 401-408 grouping located in the *Unidex 21 User's Manual.*

HOME

NOTES (CON'T):

4) On rotary systems which have a cam type switch (normally open on one side, and closed on the other) the Unidex 21 will automatically move in the correct direction to the Home position.

RELATED COMMANDS:

$nAP, **$nRP**, **ACDE**, **FXOF**, **G92**, **MORG**, **REF**

HSIE

NAME:

HSIE - High Speed Interrupt Control

FUNCTION:

The **HSIE** command is used for data collection from the CPU card. The data acquisition event is triggered externally, by the User, through the (P23) I/O connector on the Interface panel of the Unidex 21. Data that can be collected consists of system variables, **$nAP**, **$nRP**, **$INP**, **$IN0-$INF**, **$000-$7FF**, or **$800-$F6F**. Typical timing latency for an input device (excluding the system variable) is 45 micro seconds for the first data and 25 micro seconds thereafter unless further latency is encountered due to the read cycle of the data card. The accuracy of the **$nAP** and **$nRP** Machine Commanded Position Register acquisition is dependent upon Unidex 21's ability to access the Indexer board.

FORMAT:

(**HSIE**,option)

RETURNS:

EXAMPLE:

(**HSIE**,1)

PROGRAMMING EXAMPLE:

The following High Speed Interrupt example shows how the **HSIE** command is used to collect data in the background and store that data to a file. This example saves both Absolute and Relative Position to the TEST.DAT file.

```
(DVAR,CNT)                     ; CNT is the variable used to write the data to a file.
G17                            ; The XY plane is the 1st plane.
G40                            ; Ensure Cutter Compensation is OFF.
G70                            ; Initiate English Programming (inches).
G91                            ; Use Incremental Positioning.
F1000.                         ; A Feedrate of 1000 inches per minute is established.
(MALC,<0,200,$XRP,$XAP>)       ; Allocate memory for High Speed Interrupt for a total
                               ; of 200 data items.
```

HSIE

PROGRAMMING EXAMPLE (CON'T):

```
                        ; Record 100 X Axis Incremental Positions,
                        ; and 100 X Axis Absolute Positions.
(HSIE,1)                ; Enable the High Speed Interrupt.
                        ; Axis position will be captured upon
                        ; each falling edge of the HSI Misc I/O pin.
G1 X100.                ; Linear Contouring at Feedrate of 100 inches per minute.
X-50.                   ; Move the X axis 50 inches in the negative direction.
X30.                    ; Move the X axis 30 inches in the positive direction.
X-80.                   ; Move the X axis 80 inches in the negative direction.
(HSIE,0)                ; Disable the High Speed Interrupt and store the
                        ; accumulated data to a file.
(OPEN,W,TEST.DAT)       ; Open the Output file.
CNT=1.                  ; Start with the 1st new data item.
(RPT,$HSI<0>            ; Repeat the number of data items.
 (WRIT,#$HSI<CNT>)      ; Write the data item to the file.
 CNT=CNT+1.             ; Move to the next data item.
)                       ; End of Repeat Loop
(END,S)                 ; Close the File and save all data written.
M2                      ; End of The Program
```

NOTES:

1) Prior to enabling the High Speed Interrupt, the MALC <.....> command must be used to allocate appropriate memory space and specify the data to be captured.

2) The following options are available when using the **HSIE** command:

option 0 – Disable the High Speed Interrupt.

1 – Input starts at the same memory location. New data will overwrite previous data.

2 – New data will start at the next memory location. Data collection will stop when the end of memory is reached.

HSIE

NOTES (CON'T):

option 3 – New data will start at the next memory location. Data collection will stop when the end of memory is reached. When the **HSIE** is enabled, the interface pin on the User I/O bus will go HIGH. When the end of memory is reached the pin goes LOW. (The User may use this function to control an external device or to connect to **INT1/INT2**.)

3) Options 1, 2, and 3 enable the High Speed Interrupt. The function remains in effect and continues to run in the background until the system is reset or the command (**HSIE**,0) is entered, to disable it.

4) All User Interrupt Signals must be debounced. Reference the *Unidex 21's Hardware Manual* for a description of the miscellaneous I/O connector P23.

RELATED COMMANDS:

$HSI, END, INT1/INT2, MALC, MEND, OPEN, RETP, WRIT

HWEL

NAME:

HWEL - Handwheel Control

FUNCTION:

The **HWEL** command initiates Handwheel Control of a designated axis or axes. Two Handwheel/Axis combinations may be used simultaneously, if both inputs are utilized. A Scaling Factor establishes the ratio between Handwheel Steps and Machine Steps. Setting the Scaling Factor to zero disables the Handwheel Control. One Handwheel input may be assigned to only 1 axis at a time.

FORMAT:

(**HWEL**,input source, scaling factor,axis name)

RETURNS:

EXAMPLE:

(HWEL,1,50,X)	; The X axis is activated for Handwheel Control through Input Port 1. Each Handwheel Step equals 50 Machine Steps.
(HWEL,VAR1,0,X)	; Handwheel Control of the X axis is disabled for Input 1 or 2, dependent upon the value of variable VAR1.

NOTES:

1) The input source may be 1 or 2, and may be specified as a variable.

2) The Scaling Factor may be 0 to 255, and may be specified as a variable.

3) The Handwheel inputs may be configured to accept Clock/Direction, CCW/CW clock, Quadrature (X1) or Quadrature (X2). Refer to the General Machine Parameters #28, 29, 42, 43, and 45 located in the *Unidex 21 User's Manual.*

4) Manual Feedrate Override **MFO** may also be utilized to adjust the Handwheel scale. Refer to General Machine Parameter #55, also located in the *Unidex 21 User's Manual.*

RELATED COMMANDS:

AFCO, DVAR

I,J,K,P and i,j,k,p

NAME:

I, **J**, **K**, **P** and **i**, **j**, **k**, **p** - Circular Interpolation parameters

FUNCTION:

The **I**, **J**, **K**, **P** and **i**, **j**, **k**, **p** commands are Circular Interpolation parameters that are parallel to the X, Y, Z, U or x, y, z, u axes respectively. The values may be positive or negative.

When performing Circular Interpolation using the **G2/G12/H2/H12** or **G3/G13/H3/H13** commands, the **I**, **J**, **K**, **P** and **i**, **j**, **k**, **p** commands indicate the center of the arc (radius). These codes program the offset vector, which is the signed incremental distance from the beginning of the arc to the arc center. The offset vector is determined incrementally regardless of whether the Unidex 21 is in the Absolute (**G90**) or Incremental (**G91**) mode.

The offsets for each axis are as follows:

AXIS	OFFSET
X	I
Y	J
Z	K
U	P
x	i
y	j
z	k
u	p

When performing Circular Interpolation using the **G5/G15/H5/H15** commands, the **I**, **J**, **K**, **P** and **i**, **j**, **k**, **p** commands indicate points along the arc. (Any point may be used except for the starting and ending point.)

The Unidex 21 can be programmed to move in a complete circle by one command block that specifies the direction, centerpoint, and axes plane.

I,J,K,P and i,j,k,p

FORMAT:

Inn

or

I = variable

RETURNS:

EXAMPLE:

G17 G2 X2.0 Y-0.5 I1.0 J-0.1	; A CW arc will be produced from the initial position to the X=2.0, Y=-0.5 coordinate position with the center point I=1.0, J=-0.1. See Figure 3-3 below.

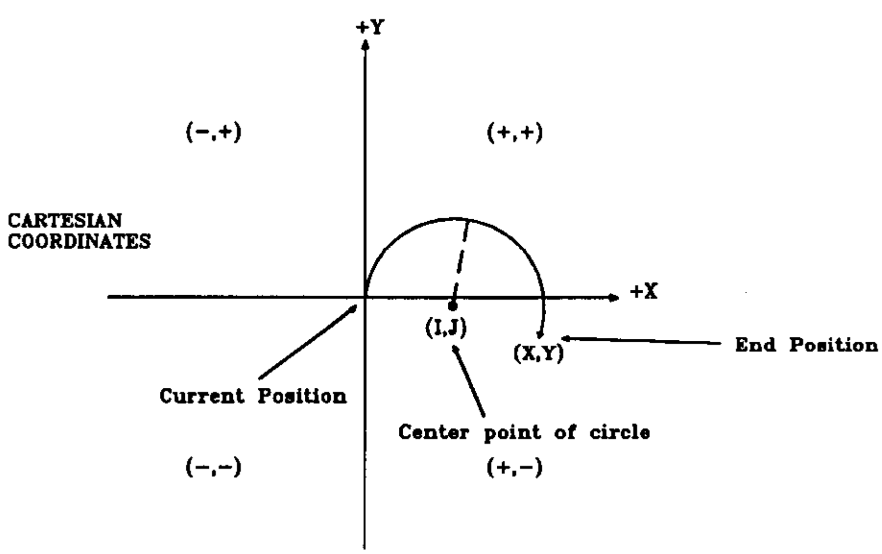


Figure 3-3: Circular Interpolation Illustrating the Use of "I" and "J" as Center Points of an ARC.

I,J,K,P and i,j,k,p

EXAMPLE (CON'T):

G2 G17 I1.0 J1.0	; A CW circle on the X/Y plane will be produced. The ending point is assumed to be 0 (the same as the starting point). The center point is established by the offsets of I=1.0, J=1.0 from the starting point (0,0). See Figure 3-4 below.

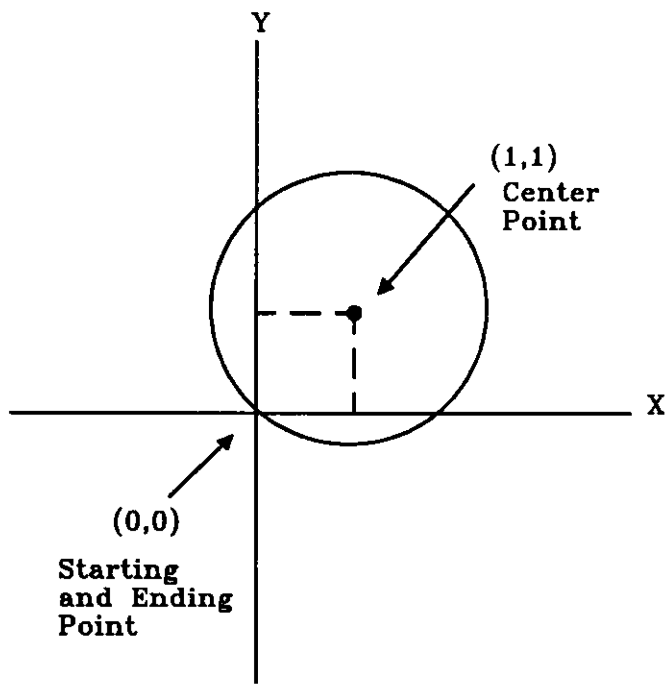


Figure 3-4: Circular Interpolation Illustrating the Use of "I" and "J" for Center Points of a Circle.

I,J,K,P and i,j,k,p

EXAMPLE (CON'T):

G17 G5 X4.0 Y0. I2.0 J2.0	; A CW arc on the X/Y plane will be produced from the initial position to the X=4.0, Y=0. coordinate position crossing points I=2.0, J=2.0. See Figure 3-5 below.

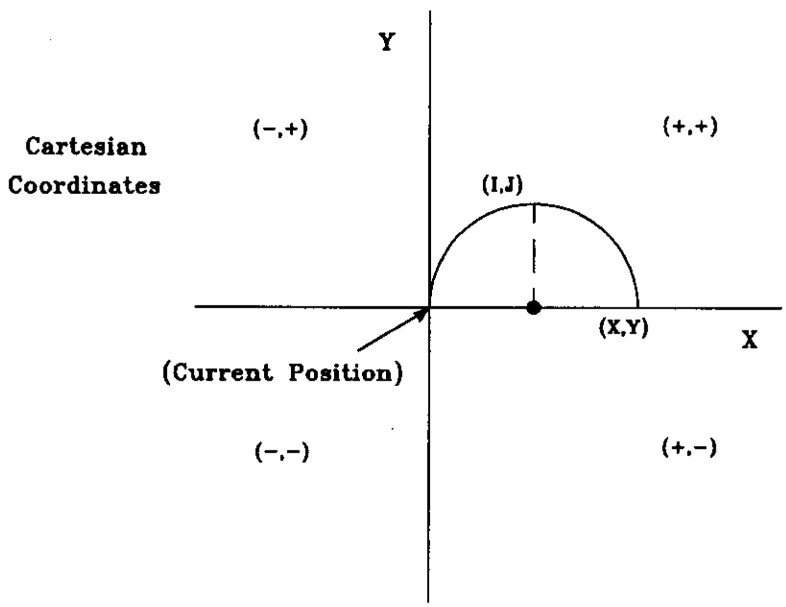


Figure 3-5: Circular Interpolation Illustrating the Use of "I" and "J" as Crossing Points on an Arc.

NOTES:

English/Metric units and decimal point placement used for the **I,J,K,P** and **i,j,k,p** commands will be the same as that established for the corresponding X, Y, Z, U and x, y, z,u axes.

RELATED COMMANDS:

G2/G12/H2/H12, G3/G13/H3/H13, G5/G15/H5/H15, G17/H17, G18/H18, G19/H19, PLNE

INT1/INT2

NAME:

INT1/INT2 - User Interrupt Control

FUNCTION:

The **INT1/INT2** commands are a very powerful feature of Unidex 21 used for permitting instantaneous (7millisecond max latency) response to an input signal in order to redirect program flow. This may be utilized for interlocks or PLC interface.

INT1 and **INT2** are negative edge triggered interrupts, i.e., the interrupt can only be generated when the signal goes from HIGH to LOW. The User may program any one of these interrupts to perform specific interrupt functions. These signals are available via the connectors located on Unidex 21's rear panel Interface board. Similarly, the User may program the system function key in the Parameter Mode in order to emulate this function for **INT1** and **INT2**. Additional Interrupts (**INT3-7**) are also available, see notes below for further explanation.

FORMAT:

(**INT1**,option,xxxx)

The following options are available when using the **INT1/INT2** command (The "xxxx" provides the Interrupt service entry point or Subroutine):

option 0 – Disable Interrupt

1 – Enable Interrupt. Unidex 21 will not stop or abort the current move when jumping to Subroutine xxxx . It will perform defined functions (I/O, mathematical, system variable Input/Output, or MST functions) as long as these are not axis moves. Upon completion of the xxxx Subroutine, the Unidex 21 continues with the next block of functions.

2 – Enable Interrupt. Unidex 21 will stop the current move and abort any functions remaining in this block and store the current machine position. Upon completion of the xxxx Subroutine, the Unidex 21 will return to the next block of program.

INT1/INT2

FORMAT (CON'T):

option 3 – Enable Interrupt. Unidex 21 will stop the current move and abort any functions remaining in this block storing the current machine position. It will then jump to the Defined Entry Point xxxx, but will not precede to the next block when finished. When this option is used in conjunction with the **MSTD** command the User may dedicate **MST** data to be output at the same moment that motion stops. The Unidex 21 does look ahead while interpreting the Parts Program.

4 – Enable Interrupt. Unidex 21 will finish all of the functions in the current block before going to Subroutine xxxx. Upon completion of the Subroutine, the Unidex 21 will return to the next block of program.

5 – Enable Interrupt. Unidex 21 will finish all of the functions in the current block, then jump to the Defined Entry Point xxxx. Upon completion, the Unidex 21 will not return to the next block of program.

6 – Same as option three, but the Unidex 21 does not look ahead while interpreting the Parts Program.

RETURNS:

EXAMPLE:

```
(INT1,1,OFF)          ; Enable Interrupt 1, option 1. ( Subroutine OFF)
X10. Y10. F100.       ; An X and Y axis motion at a Feedrate of 100 inches or
                        millimeters per minute is established.
M2                    ; End of The Program
(DFS,OFF              ; Start of the interrupt service routine.
  (INT1,0)            ; Disable Interrupt 1 to prevent multiple triggers.
  M80                 ;
  (INT1,1,OFF)        ; Re-enable Interrupt 1, option 1. ( Subroutine OFF)
)                     ; End of interrupt service routine.
```

INT1/INT2

NOTES:

1) Once serviced, interrupts **(INT1/INT2)** remain enabled until disabled by the program. This is also true for additional interrupts **(INT3-7)**.

2) When Options 2, 4, 5, or 6 are enabled the Unidex 21 does not look ahead while interpreting the Parts Program, thus making program execution slower than Options 1 and 3.

3) You may enable a "dummy" interrupt function with n= 2, 4, or 5 in order to stop Unidex 21 from looking ahead. This is useful when you want to enable the **SKEY** Softkey function but do not want Unidex 21 to look ahead, thus ensuring no blocks of the program are skipped over.

4) All User Interrupt signals must be debounced.

5) The additional interrupts **INT3-6** require usage of the optional Unidex 21 MPI Hardware card. These are functionally similar to **INT1/INT2** except option 3 is unavailable.

6) **INT7** provides a special **MST** output from a hardware signal input to the Unidex 21 either from a special Indexer board pin marked "feedhold" or from the **T** strobe (requires hardware modification).

   The first case format is:

   (INT7,1,Mxx,Sxx,Txx,Mxx) ; Interrupt source is from MST bus connector pin marked "feedhold". Upon each HIGH to LOW trigger the Unidex 21 will output MST data, at most 4 sets. If subsequent interrupts occur before the MST data is executed, they will be ignored.

   The second case format is:

   (INT7,n,Mxx,Sxx,Mxx,Mxx); After MST bus "T" strobe line receives "n" (n=2 to 1,677,215) times trigger MS output will occur, at most 4 sets, with no T data permitted. This requires Indexer board hardware modification and factory initialization.

INT1/INT2

PROGRAMMING EXAMPLE:

This Interrupt example shows how to grab the value found at Input address $700 (exact time an interrupt occurs). These values are placed in an array.

```
G17                   ; The XY plane is the 1st plane.
G40                   ; Ensure Cutter Compensation is OFF.
G70                   ; Initiate English Programming (inches).
G91                   ; Initiate Incremental Positioning.
F1000.                ; A Feedrate of 1000 inches per minute is established.
(DARY,ARY<100>)       ; Array to hold the values read.
(DVAR,INDX)           ; Array Index is Initialized to 0.
(INT1,1,ISR1)         ; Enable Interrupt INT1.
                      ;
program block         ; These blocks may contain whatever commands
program block         ; are necessary. Upon receiving an interrupt,
program block         ; the Subroutine ISR1 will be called. Motion will
program block         ; not be stopped. When this routine finishes,
program block         ; program execution will continue.
                      ;
(INT1,0)              ; Disable Interrupt INT1.
program block         ; Analyze data placed into the array.
program block         ;
                      ;
M2                    ; End of The Program
(DFS,ISR1             ; Start of the interrupt service routine.
 ARY<INDX>=$700       ; Read the current input value into the array.
 INDX=INDX+1          ; Move to the next array location.
)                     ; End of the interrupt service routine.
```

Also, see Programming Example #4 located in Appendix 3 of this manual for further information regarding use of the **INT1/INT2** commands.

RELATED COMMANDS:

HSIE, MSTD, SKEY

JOIN

NAME:

JOIN - Join Subprogram to Main Program

FUNCTION:

The **JOIN** command is useful when the User wants to separate files which relate to common machine functions, and recall them easily without having to rewrite each within the main Parts Program. Using this command will conserve User RAM. There are two cases under which the **JOIN** command works.

case 1 – applies to a single program which is joined with a Main Program.

2 – applies to the joining of multiple or all programs in memory.

In both cases, the only way for the Main Program to interface with each other is via the **JUMP/CLS** commands. See notes below regarding case 1 and case 2 special considerations.

FORMAT:

(JOIN,filename.type)	; Join 1 Subprogram only
(JOIN,+,file1.type,file2.type,file3.type, etc...)	; Join multiple subprograms together
(JOIN,+)	; Join all programs in memory together

RETURNS:

EXAMPLE:

(JOIN,LASER.PP)	; The Subprogram LASER.PP will be joined to the current Main Program.
(JOIN,+,LASER.PP,GAS,SHUTTER)	; The selected programs LASER.PP, GAS, and SHUTTER will be joined with the Main Program.
(CLS,GAS)	; Call the GAS Parts Program as a Subroutine.

JOIN

NOTES:

1) The following two cases should be considered when joining a program(s):

case 1 – Single program join

1) The only way for two programs to interface with each other is via **JUMP/CLS**.

2) The following commands need memory management, so they may only be utilized in the Main Program.

END, JOIN, MALC, MEND, JOIN, WRIT

3) The following commands are permitted within all programs and can be accessed by either the Main Program or the Subprogram.

CLS, DARY, DENT, DFLS, DFS, DVAR, DZON, JUMP

case 2 – Multiple program join

1) When multiple files are joined, the Unidex 21 system creates a special $$$.$$$ program for run mode operation only. In this case, each Subprogram will be assigned as a Subroutine, with the first 4 characters of the file name assigned as the Subroutine name. Therefore, these joined programs must follow the rules for variables. i.e., first two characters must be A-Z.

2) Since each Subprogram is called via its first 4 characters it is not possible to utilize **DFLS** or **DFS** as the first lines within the program. Also, each one of the subprograms cannot contain the following:

M2, M30, M47

RELATED COMMANDS:

CLS, DARY, DENT, DFLS, DFS, DVAR, DZON, JUMP

JUMP

NAME:

JUMP - Jump to User Defined Entry Block

FUNCTION:

The **JUMP** command causes program execution to jump to and continue from a specific block. The block entry point is identified by a name which is 2 to 4 characters in length with the restriction that the first two must be letters, A-Z, while the remaining can be any alphanumeric characters.

The **JUMP** command may be one of four types; direct, indirect, conditional direct, or conditional indirect. Examples of each type are provided below.

If a variable is utilized it must be identified with the **DVAR** Define Variable command.

FORMAT:

(**JUMP**,entry point)

RETURNS:

EXAMPLE:

(**JUMP**,AA11)	; Program execution goes directly to Entry Point AA11.
(**JUMP**,#VAR1)	; Program execution goes to a Defined Entry Point. Variable VAR1 contains the ASCII name of the entry point.
(**JUMP**,ENT1,VARA.EQ.VARB)	; Program execution goes to Entry Point ENT1 if variable VARA =VARB.
(**JUMP**,#VAR1,VARA.NE.VARB)	; Program execution goes to the entry point contained in variable VAR1 if VARA does not equal VARB.
(**JUMP**,END,$IN0.EQ.H,00)	; Program execution goes to Entry Point END if input 0 is zero (ON).

JUMP

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #1, 2, 3, and 7 located in Appendix 3 of this manual.

NOTES:

1) Do **NOT** duplicate names when defining Jumps and Subroutines.

2) The (**JUMP**,nnnn) command should occupy its own program block.

RELATED COMMANDS:

CLS, DENT, DVAR, INT1/INT2

LCD, OAB and lcd, oab

NAME:

LCD, **OAB**, and **lcd**, **oab** - Polar coordinates for Circular Interpolation

FUNCTION:

The **LCD**, **OAB**, and **lcd**, **oab** commands are the Polar commands for Circular Interpolation. They are defined as follows:

LCD	the 1st axes plane
OAB	the 2nd axes plane
lcd	the 3rd axes plane
oab	the 4th axes plane

where;

L,O,l,o	the radius of the circle
C,A,c,a	the starting angle as measured from the Horizontal axes, 0° direction, CCW is positive
D,B,d,b	the ending angle

See Figure 3-6 for an example of Polar Coordinate Programming.

FORMAT:

Lnn Cd1 Dd2
L = variable **C**=variable **D**=variable

Both the starting and ending commands are in degrees. If minutes and seconds of an arc are required, they must be entered as fractions of degrees. Example:

C112+2/60+16/3600

LCD, OAB and lcd, oab

FORMAT (CON'T):

The angle may also be entered as a decimal. Example:

D112.03777

RETURNS:

EXAMPLE:

G2 L1. C140. D45. F50.	; Initiate CW Circular Contouring such that an arc of 95° will be produced where: L1. specifies the radius C140. specifies the angle of the starting point D45 specifies the angle of the ending point See Figure 3-6 below.

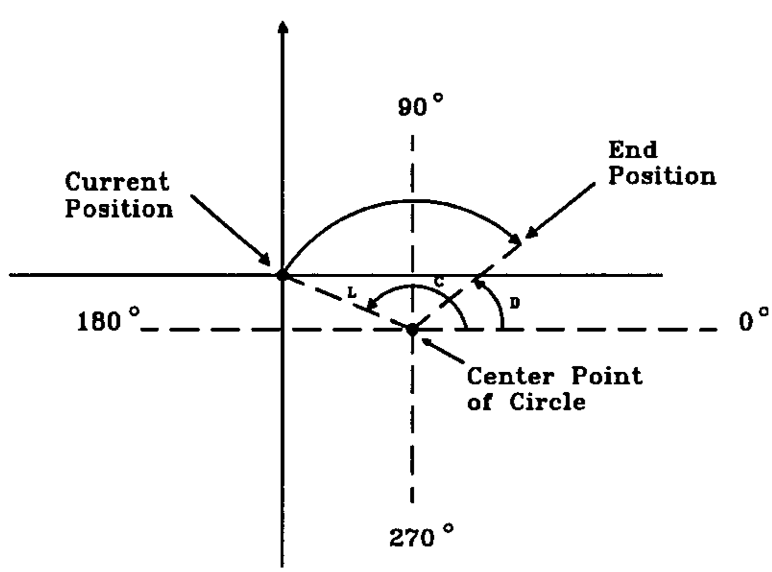


Figure 3-6: Polar Coordinate Programming

LCD, OAB and lcd, oab

PROGRAMMING EXAMPLE:

```
G70                     ; Initiate English Programming (inches).
G91                     ; Initiate Relative Positioning.
(REF,X,Y)               ; A Software Home for the X and Y axes is established.
G1 X2. Y2. F200.        ; Initiate a Linear Move for the X and Y axes,
                        ; at a Feedrate of 200 inches per minute.
G2 L1. C140. D45. F50.  ; Initiate CW Circular Contouring such that an arc of 95°
                        ; will be produced at a Feedrate of 50 inches per minute
                        ; where:
                        ;       L1. specifies the radius
                        ;       C140. specifies the angle of the starting point
                        ;       D45 specifies the angle of the ending point
M0                      ; Program Stop
```

NOTES:

The English/Metric and decimal placement parameters established in the Individual Axis Parameters 401-408 grouping apply to all **L,O** and **l,o** dimensions. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

RELATED COMMANDS:

G2/G12/H2/H12, G3/G13/H3/H13, G17/H17, G18/H18, G19/G19, PLNE

LIMT

NAME:

LIMT - Set Software Limit

FUNCTION:

The **LIMT** command is used to establish a CW and CCW Software Limit as referenced to the Home position. The values established by this command will override the values established in the Individual Axis Parameter Mode, but does not change the Parameter Mode settings. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.) The values established by this command will be in effect until a subsequent **LIMT** command, or the system is reset.

FORMAT:

(**LIMT**, case, axis 1 name CW limit, axis 1 name CCW limit, axis 2 name....)

The cases are:

case 0 – removes any Software Limits of the specified axes.

1 – limits are referenced from the Hardware Home position using Absolute coordinates.

2 – limits are referenced from the current position Incrementally.

RETURNS:

EXAMPLE:

(LIMT,0,X0.,Y0.)	; Disable the X and Y axes Software Limit checking.
(LIMT,1,X20.,X100.,Y20.,Y100.)	; CW and CCW limits of 20 and 100 program units from the Hardware Home position are established for the X and Y axes.
(LIMT,2,X-10.,X10.)	; Software Limit is set from current X position, -10.(in/mm) and +10. (in/mm).

LIMT

NOTES:

The applicable parameters for setting the CW and CCW Software Limits are #47 and 48 as established under the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of these parameters.)

RELATED COMMANDS:

FXOF, MORG

LINK

NAME:

LINK - Link Multiple Axes to a Group

FUNCTION:

The **LINK** command connects "n" number of axes together in "m" number of groups. Axes that are linked together will perform the same point-to-point or contouring moves.

To retain Tracking integrity for each axis position, all axes linked into the same group should be contained under the same conditions (mirror, scaling, hardware resolution, decimal point placement, dry run, positive move direction, etc.,). However, if the conditions are different use the **G92** (reset software position registers) command after an unlink to reinitialize.

FORMAT:

```
(LINK,<axis name,axis name,...>)
        or
(LINK,0)                          ; Un-link all axes.
```

RETURNS:

EXAMPLE:

```
(LINK,<X,Y,Z>,<U,x>,<y,z>)        ; Put X,Y,Z into the 1st group,
                                  ; U,x into the 2nd group, and
                                  ; y,z into the 3rd group.

X10. U35. y18. F10.               ; Use linked axis motion.
```

In the above example,

each time the X axis moves, Y and Z will perform the same move.
each time the Y axis moves, X and Z will perform the same move.
each time the Z axis moves, X and Y will perform the same move.

The same is true for groups 2 and 3.

```
(LINK,0)                          ; Un-link all axes.
```

LINK

NOTES:

1) Only one axis should be programmed to move within each group, failure in doing so could result in unpredictable motion.

2) The **LINK** command is modal and remains in effect until changed, or the system is reset.

3) The **LINK** command does not provide a Gantry system, it performs only contouring or point-to-point motion (No Home, Free Run, etc.). To implement Gantry type control, refer to the Main System Parameter #41 located in the *Unidex 21's User's Manual*.

RELATED COMMANDS:

MALC

NAME:

MALC - User's Memory Allocate

FUNCTION:

The **MALC** command is used to Allocate Memory that is to be used for background functions. All functions associated with the **MALC** command run in the background, as a result, the **MALC** command may be input during program initialization. If the **MALC** command is used during a program run, those blocks of memory will become unavailable to the User, and may only be retrieved by overwriting the memory with another **MALC** command or by performing a system reset.

Available background functions are;

<0,n,source1,source2,...>	**HSIE** Memory Allocation. Memory is allocated for n sets of input, where n must be an integer. Each input is 4 bytes in length, where n sets need 4 * n bytes.
<1,n>	**CPAG** Memory Allocation. Memory is allocated for n sets. Information is updated every 200 msec. Each n set requires 20 bytes of memory.
<2,n>	Allocates memory for the **RETP** command to fetch Real-Time position, or for the **RECO** command that is used with the Teach Mode. n specifies the number of 256 byte blocks that are to be allocated for use by this function. Each block is capable of holding 64 position samples.
<3,n>	Allocates memory for the **PORT** command which enables a RS-232 port to collect data. Each n is equal to 1 byte.

MALC

FUNCTION (CON'T):

During the **HSIE** function only, the following system variables are permitted to be used as input sources. Each time an interrupt is received, the data is stored sequentially in memory.

$XRP, $YRP, $ZRP, $URP **$XAP, $YAP, $ZAP, $UAP**
$xRP, $yRP, $zRP, $uRP **$xAP, $yAP, $zAP, $uAP**
$INP (16 bits only, so high bits = 0)
$000-7FF (Motorola I/O bus)
$800 -$F6F (PLC DUAL-PORT RAM)
$070 -$FAF PAMUX I/O bus)

FORMAT:

(**MALC**,<background function1>,<background function2>,...)

RETURNS:

EXAMPLE:

(MALC,<0,200,$XRP,$INP,$000,$800>)	; Each time an **HSI** interrupt occurs, capture X axis Relative Position, all inputs, Motorola I/O channel 0, and PLC dual port ram location 800.

NOTES:

1) Background functions must always be contained within < >.

2) For complete examples on specific uses in each mode, refer to the command activating that mode.

3) The applicable parameter for **MALC** memory options is Parameter #50 established under the Individual Axis Parameters 401-408 grouping (see the Unidex 21 User's Manual for a detailed description of the Parameter Mode).

RELATED COMMANDS:

CPAG, HSIE, PLAY, PORT, RECO, RETP

MIR

NAME:

MIR - Mirror Image

FUNCTION:

The **MIR** command activates the Mirror Image function. The Unidex 21 Mirror Image is available for all eight axes. The **MIR** command operates in both the Incremental and Absolute mode. While in the Absolute Mode however, it must be remembered that functions are always in reference to the Software Home established by the **G92** command which means that you must return to that location before enabling the Mirror Image function.

FORMAT:

(**MIR**,axis name and enable/disable,axis name and enable/disable....)

The Mirror function is enabled by placing a non-zero number next to the axis name.

The Mirror function is disabled by placing a zero next to the axis name.

RETURNS:

EXAMPLE:

(**MIR**,X1.,Y1.)	; Activate the Mirror Image for the X and Y axes.
(**MIR**,X0.,Y0.)	; Deactivate the Mirror Image for the X and Y axes.

PROGRAMMING EXAMPLE:

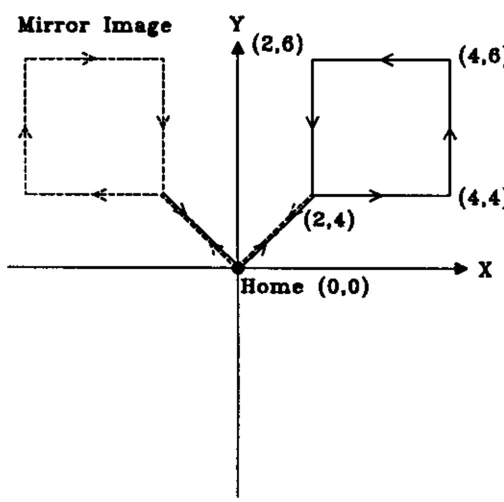
G70	; Initiate English Programming (inches).
(REF,X,Y)	; A Software Home for the X and Y axes is established.
(**MIR**,X1.,Y1.)	; Activate the Mirror Image function for the X and Y axes,
	; changing the positive X and Y values to negative values.
(CLS,BOX1)	; Call the Subroutine BOX1.
(**MIR**,X0.,Y0.)	; Turn OFF the Mirror Image function.
M2	; Stop the Program

MIR

PROGRAMMING EXAMPLE (CON'T):

```
(DFS,BOX1        ; Define the Subroutine BOX1.
  X2.            ; Initiate a positive Linear Move for the X axis.
  Y4.            ; Initiate a positive Linear Move for the Y axis.
  F100.          ; A Feedrate of 100 inches per minute is established.
  X2.            ; Initiate a positive Linear Move for the X axis.
  Y2.            ; Initiate a positive Linear Move for the Y axis.
  X-2.           ; Initiate a negative Linear Move for the X axis.
  Y-2.           ; Initiate a negative Linear Move for the Y axis.
  X-2.           ; Initiate a Linear Move for the X axis to the Home position.
  Y-4.           ; Initiate a Linear Move for the Y axis to the Home position.
)                ;
M0               ; Program Stop
```

The above example is illustrated below.



NOTES:

The **MIR** command is modal.

RELATED COMMANDS:

G92

MORG

NAME:

MORG - Machine Origin

FUNCTION:

The **MORG** command is used to return the axes to a Home Position including the "Machine Origin Offset Steps" Parameter #39 of the Individual Axis Parameters (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode). After performing motion, the User can quickly return to the Home Offset position without going through a complete Home Cycle. This command provides a rapid way to reposition to a known location without rehoming the system and is executed at a **G0** Feedrate.

FORMAT:

(**MORG**,axis name,axis name,...)

RETURNS:

The position register value of the **$nAP** is adjusted to zero, the Machine Origin Offset upon completion.

The **$nRP** Position Register remains the same value as was in effect prior to issuing the **MORG** command.

EXAMPLE:

```
(MORG,X,Y)        ; The X and Y axes are sent to the location established in the
                    Individual Axis Parameter Mode for the X and Y axes.
```

PROGRAMMING EXAMPLE:

```
G70               ; Initiate English Programming (inches).
G91               ; Initiate Incremental Positioning.
(REF,X,Y)         ; A Hardware Home for the X and Y axes is established.
(MORG,X,Y)        ; Machine Origin command for the X and Y axes. The X axis
                  ; and the Y axis are offset by the distance specified by their
                  ; Machine Origin Offset (Parameter #39).
M0                ; Program Stop
```

MORG

NOTES:

1) The applicable parameter is "Machine Origin steps" #39 as established under the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

2) The axes speed is determined by the "Top Feedrate steps/sec" Parameter #8 established under the Individual Axis Parameters 401-408 grouping (see also the *Unidex 21 User's Manual*). The Acceleration/Deceleration Ramping Trajectory is always parabolic.

3) The **MORG** distance is a positive or negative number referenced from the Hardware Home exclusive of any Offsets.

4) Review the **$nAP/$nRP** command description and program example for further explanation.

5) The **MORG** command executes point-to-point motion only.

RELATED COMMANDS:

$nAP, $nRP, FXOF, G0, G92

MSG

NAME:

MSG - Display Screen Messages

FUNCTION:

The **MSG** command permits a message to be entered into a program and displayed on the screen at the time of execution. The message displayed may consist of text plus the value of User variables and system variables. The message may also prompt User input in order to control program flow and set the value of variables. The system defaults to only permitting messages to contain four lines of text. During program execution, message program lines may sometimes be confused with the Parts Program scrolling on the screen. If the (**TRAK**,2) command is utilized, the message area is more readable.

FORMAT:

(**MSG**,...message)	; To display message only
(**MSG**,<VAR1,VAR2>,...message...)	; To display message and accept User input into variables VAR1 and VAR2

The Unidex 21 recognizes the following format for display:

#VAR	as the decimal representation of the variable
#H:VAR	as the Hexadecimal format of a variable
#C:VAR	as the character format
#C:VAR1,VAR2,VAR3	as a variable string

To provide the variable input function the Unidex 21 must be in the Machine Mode.

To output a "#" to the display use "##".

The Message can NOT contain parenthesis (except for the opening and closing parenthesis). Use the "<" and ">" instead.

RETURNS:

MSG

EXAMPLE:

```
(MSG<VAR1,VAR2,VAR3,VAR4>...text...)

if the input data is:

        12,H,10FA,"Unidex21"<ENTER>

then,

        VAR1 = 12 as a real number
        VAR2 = 10FA, as a hexadecimal number
        VAR3 = "Unid"
        VAR4 = "ex21"
```

If a longer character string was requested, more **VAR** commands would have been necessary, since a variable holds only four characters. (A space between characters is counted as a character.)

PROGRAMMING EXAMPLE #1:

The following example is designed to show the operation of the **MSG** command when outputting both text and variables.

```
(DVAR,STR1,STR2,FILT,BIN)          ; STR1 and STR2 are to be used for
                                   ; ASCII data.
                                   ; FILT is to hold floating point data.
                                   ; BIN is to hold binary data.
STR1="Test"                        ; Initialize all variables.
STR2="ing"                         ;
FILT=1234.5678                     ;
BIN=H,486121                       ;
                                   ; The message displayed will be:
(MSG,Beginning Test)               ; Beginning test
(MSG,#C:STR1)                      ;    Test
(MSG,#C:STR2)                      ;     ing
```

MSG

PROGRAMMING EXAMPLE #1 (CON'T):

```
                                         ; The message displayed will be:
(MSG,#C:STR1,STR2)                       ;     Testing
(MSG,Beginning #C:STR1,STR2)             ;     Beginning Testing
                                         ;
(MSG,#H:STR1)                            ;     54657374
                                         ; Hex representation of ASCII codes.
                                         ;
(MSG,#FLT)                               ;     1234.5678
                                         ; The Decimal representation of floating point
                                         ; format.
(MSG,#H:FLT)                             ;     9A522B4B
                                         ; The Hex representation of a floating point
                                         ; number.
(MSG,#H,BIN)                             ;     486121
(MSG,#C:BIN)                             ; Ha!
                                         ; Character representation of hex data.
M2                                       ; End of The Program
```

PROGRAMMING EXAMPLE #2:

The following example shows the operation of the **MSG** command when input is being received form the User.

```
(DVAR,STR1,STR2,FLT,BIN)                 ; Variables STR1 and STR2 are to be
                                         ; used for ASCII data.
                                         ; Variable FLT is to hold floating point data.
                                         ; Variable BIN is to hold binary data.
                                         ; Prompt the User.
(MSG,<STR1,STR2,FLT,BIN>,
  Enter 5 - 8 Character String,
  a floating, and a binary number
  separated by commas)                   ;
M2                                       ; End of The Program
```

MSG

PROGRAMMING EXAMPLE #2 (CON'T):

The screen will be displayed as:

"Enter 5-8 Character String, a floating, and a binary number separated by comma's"

USER INPUT	VARIABLE VALUES
"ABCDEFGH",1234.5678,H,55AA	STR1 = ABCD STR2 = EFGH FLT = 1234.5678 BIN = H,55AA
"AB","CD",25.78,H,BE2F	STR1 = AB STR2 = CD FLT = 25.78 BIN = H,BE2F
"Unidex",1965.0419,H,FACO	STR1 = Unid STR2 = ex FLT = 1965.0419 BIN = H,FACO
"This is a test"	STR1 = This STR2 = is FLT = a te BIN = st

As can be seen from this example, any input received from the User should be checked for validity prior to using it.

Also, for additional information of how this command may be used, refer to Programming Examples #1, 2, 3, 4, and 7 located in Appendix 3 of this manual.

MSG

NOTES:

1) The Unidex 21 will display a real constant greater than 99999999 as:

   1.000000E8

   and a real constant less than 0,0000001 as:

   1.0E-8.

2) The "up arrow" and/or "down arrow" keys may be used to recall previously entered data.

3) Input is terminated when the User presses the ENTER key.

4) There is no type checking of the data being entered by the User. This must be checked in a separate routine which may be part of a User Parts Program.

5) If the User enters less data than expected, unset variables are set to 0.

6) If the User enters more data than can be placed into the variables supplied, unpredictable results may occur.

7) Hex numbers which require more than 4 bytes of storage are truncated to 32 bits. (Least Significant)

RELATED COMMANDS:

$nAP, $nRP, COMM, CPAG, DVAR, TERM

M, S, T

NAME:

M,S,T - Output Capability (16 bit outputs)

FUNCTION:

The Unidex 21 provides additional output capability through the MST bus. Historically, the MST bus has controlled Machine Spindle and Tool Functions. Today, the Unidex 21's MST bus provides output on-the-fly capability when used in conjunction with the **G8** and **G9** commands and can control an analog output through the use of an (SDA) Spindle D to A option card. The **M**, **S**, or **T** commands are used exclusively as outputs. Each command contains 16 bits, **M**0 to **M**0FFFF, **S**0 to **S**0FFFF, and **T**0 to **T**0FFFF respectively. The duration of the output and the delay time to receive an acknowledgement for **M**, **S**, and **T** commands is established in the Individual Axis Parameter Mode. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

M commands may be interpreted in two ways by the Unidex 21.

The following **M** commands provide special functions:

M0	Program Stop. Upon completion of all other words in the block, the program will stop. To continue the program, press the CYCLE START pushbutton located on the front panel of the Unidex 21.
M1	Optional (planned) Stop. When the **M1** command is decoded (the OPTIONAL STOP command is active) program execution will stop. To activate the OPTIONAL STOP function, depress the toggle type switch located on the front panel of the Unidex 21. However, if the OPTIONAL STOP function is deactivated, the **M1** command will be ignored by the Unidex 21. To continue running the program, press the CYCLE START pushbutton located on the front panel of the Unidex 21.
M2	End of The Program. After completion of all other commands, program execution will stop. Press the CYCLE START pushbutton on the Unidex 21's front panel to return to the program's beginning and restart the program run.
M30	End of The Program. The program will not run again if CYCLE START is pressed.

M, S, T

FUNCTION (CON'T):

M47 Return to Program Start. When the M47 command is decoded, program execution will continue from the program's beginning.

Aerotech offers a Latched M-function (LM16) option card (see the *Unidex 21 Options Manual* for further details) as well as provide jumper selectability of addresses. However, when utilizing the LM16 card option, if one of the above system "M" codes is needed to serve as an output, it is necessary to program them as a three digit "M" function.

For Example:

M2	; Normally signifies the End of The Program, but due to request from the User.
M102	; Permits **M2** to be configured as an output. (Aerotech's LM16 option card decodes only the last two digits.)

FORMAT:

Mxxxx, Sxxxx, Txxxx

M=variable, **S**=variable, **T**=variable

The value of the **M, S, T** commands may be either positive or negative.

If the data begins with an A, B, C, D, E, or F, make certain that it is preceded with a "0".

RETURNS:

EXAMPLE:

```
M0011          ; OK
M1AB           ; OK
SABCD          ; Will not be correctly decoded
S0ABCD         ; OK
S-2000         ; OK
M80            ; OK
```

M, S, T

PROGRAMMING EXAMPLE:

For an example of how the **M0**, **M1**, **M30**, and **M47** commands may be used, refer to Programming Examples #1, 2, 3, 4, 6, and 7 located in Appendix 3 of this manual.

NOTES:

The applicable Main System Parameters are #32, 33, 34, 35, 36, and 37 (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode).

RELATED COMMANDS:

G8, G9, MSTD

MSTD

NAME:

MSTD - MST Strobe/Ack Delay, also Output during INT1/INT2, Option 3

FUNCTION:

The **MSTD** command is used to override the **M**, **S**, and **T** commands Strobe/Ack Delays as established in the Individual Axis Parameter Mode. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.) The Parameter values established by this command will be in effect until a subsequent **MSTD** command, or the system is reset.

When this command is used in conjunction with the **INT1/INT2** Option 3, the User may dedicate **M, S, T** Data to be output at the same moment that motion stops.

As an option, the User may utilize these strobe and acknowledge hardware lines as a handshaking to an external device.

FORMAT:

(**MSTD**,n,mmmm)

The following codes are used for the n and mmmm entries:

n = 0	**M** Strobe Delay	8	**INT1** - S
1	**M** Ack Delay	9	**INT1** - T
2	**S** Strobe Delay	10	**INT2** - No MST output
3	**S** Ack Delay	11	**INT2** - M
4	**T** Strobe Delay	12	**INT2** - S
5	**T** Ack Delay	13	**INT2** - T
6	**INT1** - No MST output		
7	**INT1** - M		

Note that (6/7/8/9) and (10/11/12/13) are mutually exclusive groups i.e., (that is, n = 6 will override a previous n = 7, 8, or 9).

mmmm = 0 to 65535 delay in msec

or

mmmm = **M, S, T** data

MSTD

RETURNS:

EXAMPLE:

(MSTD,0,100)	; The **M** Strobe Delay is set to 100 msec.
(MSTD,7,80)	; At the moment that **INT1** Option 3 Interrupt stops motion, **M80** will be output.
(MSTD,7,=VAR1)	; The Output data is a variable.

NOTES:

1) If an Ack Delay is equal to zero, the acknowledgment is not checked.

2) If an Ack Delay is set to 65,535 msec, the Unidex 21 will scan indefinitely checking every 1 msec for an acknowledge signal.

3) The applicable Main System Parameters are #32, 33, 34, 35, 36, and 37 (see the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode).

RELATED COMMANDS:

INT1/INT2, M,S,T, MSTD

MTOR

NAME:

MTOR - Motor Current Command Control

FUNCTION:

The **MTOR** command is used to Activate or Deactivate the current command to the servo amplifier and thus the motor torque of an individual axis or all axes. This is useful when the User wishes to disable axes torque and freely move the axes around. Also, during program debug the User may run his actual Parts Program (less motion) in the Machine Mode and utilize the Digital Scope of the Unidex 21 in order to view each axes trajectory commands.

FORMAT:

(**MTOR**,on/off,axis name,axis name,...)

The axis motor is Activated by placing a non-zero number next to the command.

The axis motor is Deactivated by placing a zero next to the command.

RETURNS:

EXAMPLES:

(**MTOR**,0, X,Y,VERT)	; The motor torque to the X, Y, and VERT axes is turned OFF.
(**MTOR**,1,U,x)	; The motor torque to the U and x axes is turned ON.
(**MTOR**,0)	; All axes motor torque is turned OFF.
(**MTOR**,1)	; All axes motor torque is turned ON.
(**MTOR**,VAR1,X,Y)	; The motor torque of X and Y axes is either ON or OFF dependent upon the value of variable VAR1.

MTOR

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #2 located in Appendix 3 of this manual.

NOTES:

1) The **MTOR** command will not activate a motor that has been deactivated by the use of the Emergency Stop pushbutton (Unidex 21 Emergency Stop circuitry option).

2) The **MTOR** command is NOT a substitute for Emergency Stop circuitry. Only the current command to the amplifier is disabled, not the amplifier itself.

3) While the system position tracking display is updated, position of variables **$nAP** and **$nRP** are not updated until the motor torque is restored.

4) Usage on a vertical axes with/without brake control circuitry may cause the axis to fall as the brake is not automatically engaged.

5) The axis traps are defeated when motion torque is turned OFF. The applicable parameters are #29, 32, 33, 34, and 35 as established under the Individual Axis Parameters 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

RELATED COMMANDS:

$nAP, $nRP, DRUN, DVAR

OPEN,MEND,END,WRIT

NAME:

OPEN,MEND,END,WRIT - Reads/Writes data to memory

FUNCTION:

These commands are used to Read and/or Write data to memory. (See examples below.)

The **$R** (Read) command is a system variable that initiates the read of one element of data from a Read file, then increments the pointer to the next available data. However, if retrace is being used the **$R** command will move back one element and read.

FORMAT:

(**OPEN**,R,Filename.type)	; Open a designated file for Read only.
(**OPEN**,W,filename.type)	; Open a designated file for Write only. An error format will result if the file currently exists.
(**OPEN**,w,Filename)	; Open the designated file for Write only. If the file currently exists, the old data will be deleted.
(**MEND**,R)	; Restore the Read pointer to the beginning of the file.
(**MEND**,W)	; Restore the Write pointer to the beginning of the file.
(**END**,S)	; Close the Write file and save the data.
(**END**,A)	; Close the Write file and abort the data.
(**WRIT**,text and data)	; Write the data to the Write file. (format same as **MSG**)

RETURNS:

OPEN,MEND,END,WRIT

EXAMPLE:

```
(DVAR,HEX,FLT,STR1,STR2)        ; Define all variables.
HEX=H,AA55                      ; Initialize each variable to a known state.
FLT=77.69                       ;
STR1="Aero"                     ;
STR2="tech"                     ;
(OPEN,W,TEST.OUT)               ; Open the file "TEST.OUT".
                                ; File Output
(WRIT,#C:STR1,STR2)             ; Aerotech
(WRIT,Data Acquired on #$TOD)   ; Data acquired on 09-JAN-2004  11:31:28
(WRIT,Variable HEX=#H:HEX)      ; HEX = AA55
(WRIT,Variable FLT=#FLT)        ; FLT = 77.69
(END,S)                         ; Close the file "TEST.OUT".
M2                              ; End of The Program
```

Assume the file TEST.IN contains the following data items:

```
1234.5678
H,7F3B
"Read"
"Me"
```

```
(DVAR,HEX,FLT,STR1,STR2)        ; Define all variables.
(OPEN,R,TEST.OUT)               ; Open the file "TEST.OUT".
FLT=$R                          ; FLT = 1234.5678
HEX=$R                          ; HEX=H,7F3B
STR1=$R                         ; STR1 = Read
STR2=$R                         ; STR2 = Me
(MEND,R)                        ; Move the Read pointer to the top of the file.
STR2= $R                        ; STR2 = 1234.5678 (Access as Float)
STR1= $R                        ; STR1 = H,7F3B (Access as Hex)
HEX= $R                         ; HEX = Read (Access as Char)
FLT= $R                         ; FLT = Me (Access as Char)
(END,A)                         ; Close the Read file.
```

OPEN,MEND,END,WRIT

NOTES:

1) If a Write file is opened from the Machine Mode, the system will remain in the Machine Mode until the file is closed (using **END**,S, or **END**,A).

2) The output of the **WRIT** command is in ASCII characters (not binary). Therefore, text can be embedded into the file for readability. It can then be printed on a normal text printer, or edited using the editor of preference.

3) The data items read using the **$R** system variable can be of any type supported by the system (i.e., Float, Hex, or Character). However, the type of the data being read depends upon the type of data found in the file, not the type of the variable it is being read into.

   The following rules apply to the data in the Read file:

   a) Hexadecimal data must be preceded by capital "H," (i.e., H,AA55). Failure to use a capital "H" will result in a undefined variable error.
   b) Character data must be no longer than 4 characters in length and enclosed in quotes (i.e., "HELP").
   c) Floating point data requires no special formatting.

4) When all data has been read from a file, subsequent reads will return zero as the data.

RELATED COMMANDS:

$R, DVAR, MSG

PID

NAME:

PID - **Kp, Ki, Kd, Kf1,** and **Kf2** Value

FUNCTION:

When peak performance must be maintained, under changing functional, inertial, or velocity conditions, it may be necessary to change the servo gain parameters. The **PID** command was designed to provide the User with the ability to set the **Kp, Ki, Kd, Kf1,** and **Kf2** values for each axis, overriding the Individual Axis Parameter Mode settings. The Parameter values are not changed by this command. The values established by this command will be in effect until a subsequent **PID** command, or the system is reset.

FORMAT:

(**PID**,axis name,gain,and value)

RETURNS:

EXAMPLE:

(PID,X,P10.,I=VAR1,D=ARY<3>,F1=20.,F2=30.)	; The **PID** F1 and F2 values will be set only for the X axis.
(PID,Y,D)	; Return the **PID** values to the default values (Parameter setting) for only the Y axis.

NOTES:

1) Changing the **Kp, Ki, Kd, Kf1,** and/or the **Kf2** values for an axis will affect the motion control response of that axis. Therefore, the values should not be changed during motion.

2) All values for an axis must be specified, even if only changing one item.

3) Only one axis may be changed at a time.

RELATED COMMANDS:

DVAR

PLAY

NAME:

PLAY - Play Back the Recorded Axes Motion

FUNCTION:

The **PLAY** command initiates a repeat of motion that has been recorded using the **RECO** command.

FORMAT:

(**PLAY**,n)

The following options are used for **PLAY** command entries:

If n = 1 – motion will commence to the first recorded position at **G0** then continue the recorded motion. Tracking Display ON

-1 – motion is the same as 1 above. Tracking Display OFF

2 – motion will commence from the axes current position. Tracking Display ON

-2 – motion is the same as 2 above. Tracking Display OFF

RETURNS:

EXAMPLE:

(PLAY,1)	; Axes will move at G0 to first recorded position, and continue the recorded motion.

NOTES:

1) Prior to using the **PLAY** command, make certain the **RECO** command is disabled.

2) To load a recorded file, see "File Mode" in the *Unidex 21 User's Manual.*

RELATED COMMANDS:

RECO

PLC

NAME:

PLC - Programmable Logic Control Interface

FUNCTION:

The Unidex 21 offers an optional PLC (Programmable Logic Controller) which resides on an AT bus Dual-Port RAM interface. The **PLC** command initiates an Interface with the optional Programmable Logic Controller.

FORMAT:

The following codes are used for **PLC** command entries:

(**PLC**,0)	; Reset **PLC**, clear **PLC** memory
(**PLC**,R)	; Set the **PLC** to the Run Mode
(**PLC**,P)	; Stop Run Mode, set to Program Mode
(**PLC**,L,filename.type)	; Load a Ladder Program to **PLC**

RETURNS:

EXAMPLE:

(**PLC**,L,LASER.LAD)	; Load a **PLC** Ladder Program named LASER.LAD from the User's memory to the **PLC**.

NOTES:

1) A **$800** to **$F6F** memory map connects the Unidex 21 to the **PLC** Dual-Port RAM at the 20000H to 2076FH location.

2) Refer to the PLC in the *Unidex 21 Options Manual* for further detail.

RELATED COMMANDS:

$800 to **$F6F**

PLNE

NAME:

PLNE - Define Circular Contour Plane

FUNCTION:

The Unidex 21 provides full contouring capability for any combination of axes. The **PLNE** command establishes Circular Contour Planes among the eight axes. It is normally used to define planes which contain axis pairs that cannot be generated using the **G17/H17**, **G18/H18**, and **G19/H19** commands. It also provides for redefinition of primary axis plane designation such that the User may create a **G2 G17** contouring plane grouping for any arbitrary pair of axes.

FORMAT:

(**PLNE**,p1a1,p1a2,p2a1,p2a2,p3a1,p3a2,p4a1,p4a2)

where

p1a1	; refers to the 1st plane, 1st axis
p1a2	; refers to the 1st plane, 2nd axis
p2a1	; refers to the 2nd plane, 1st axis
p2a2	; refers to the 2nd plane, 2nd axis
p3a1	; refers to the 3rd plane, 1st axis
p3a2	; refers to the 3rd plane, 2nd axis
p4a1	; refers to the 4th plane, 1st axis
p4a2	; refers to the 4th plane, 2nd axis

The sequence in which the axes names appear defines a three dimensional group.

RETURNS

PLNE

EXAMPLE:

PLANES DEFINED	1st	2nd	3rd	4th
G17/H17	X/Y	Z/U	x/y	z/u
(PLNE,X,Y,Z,U,x,y,z,u)	X/Y	Z/U	x/y	z/u
G18/H18	X/Z	Y/U	x/z	y/u
(PLNE,X,Z,Y,U,x,z,y,u)	X/Z	Y/U	x/z	y/u
G19/H19	Y/Z	X/U	y/z	x/u
(PLNE,Y,Z,X,U,y,z,x,u)	Y/Z	X/U	y/z	x/u
(PLNE,X,x,Y,y,Z,z,U,u)	X/x	Y/y	Z/z	U/u
(PLNE,u,z,Z,x,Y,U,X,y)	u/z	Z/x	Y/U	X/y
(PLNE,z,u,x,y,Z,U,X,Y)	z/u	x/y	Z/U	X/Y

NOTES:

1) All axes must be set up even if less than eight are being used.

2) The three dimensional group defined by the sequence in which the axes names appear is used to determine CW/CCW direction for a particular plane. CW/CCW direction for a plane is determined from the perspective of the positive direction of the remaining axis in the three dimensional group. For Example:

   (PLNE,X,Y,Z,U,x,y,z,u)

   defines the X and Y axis to be in the first plane, and X, Y, and Z to be within the same three dimensional group. CW direction for this plane will be defined with respect to positive positions on the Z axis. That is, if the XY plane were a tabletop, and Z positive was in the upward direction, CW direction would be determined looking down onto the table. However, if Z positive was in the downward direction, CW direction would be determined from under the table, looking up.

PLNE

NOTES (CON'T):

Other three dimensional groups defined in this example are:

Z/U/x, x/y/z, and z/u/X

3) The planes set up using this command are modal. That is, they stay in effect until a subsequent **G17/H17**, **G18/H18**, **G19/H19**, or **PLNE** command is executed.

RELATED COMMANDS:

G2, G3, G17/H17, G18/H18, G19/H19

POP

NAME:

POP - Pull Data Out of User's Stack

FUNCTION:

The **POP** command is used to retrieve data from the program stack that was previously stored by the **PUSH** command.

FORMAT:

(**POP**,data2,data1)

Data is retrieved from the data stack in a "last in - first out" basis.

RETURNS:

EXAMPLE:

```
(PUSH,VAR1,VAR2)          ; Data will be stacked as follows:
                                    VAR2
                                    VAR1

(POP,VAR2,VAR1)           ; Data will be retrieved in the correct order.

(POP,VAR1,VAR2)           ; Data will be swapped between VAR1 and VAR2.
```

For a more complete example see the **PUSH** instruction.

NOTES:

Each data string stored by the **PUSH** command requires 4 bytes of User stack memory. Program stack size (bytes) is established within the Parameter Mode. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

RELATED COMMANDS:

PUSH, STKP

PORT

NAME:

PORT - RS-232 Port Background Data Collection

FUNCTION:

The **PORT** command is used to Activate or Deactivate an RS-232 port for background data collection. Data is collected and stored in the User's memory (see Notes).

FORMAT:

(**PORT**,option)

RETURNS:

EXAMPLE:

(PORT,A)	; Port A will be activated to receive data, but all previously collected data will be erased.
(PORT,B)	; Port B will be activated to receive data, but all previously collected data will be erased.
(PORT,0)	; Any previously activated port will be de-activated.

PROGRAMMING EXAMPLE:

This RS-232 **PORT** program example shows how the **PORT** command can be used to permit serial communications. For generality, assume that the program is to wait until a "1" is received, and move the X axis 10 inches in the positive direction. If a "2" is received, the program is to move the X axis 10 inches in the negative direction. If a "3" is received, the program should be terminated.

```
(DVAR,CNT,TMP)          ; CNT determines the number of characters
                        ; processed.
                        ; TMP is used for temporary storage.
G1                      ; All moves will be linear.
G17                     ; The XY plane is the 1st plane.
```

PORT

PROGRAMMING EXAMPLE (CON'T):

```
G40                                  ; Ensure Cutter Compensation is OFF.
G70                                  ; Initiate English Programming (inches).
G91                                  ; Initiate Incremental Positioning.
F1000.                               ; A Feed Rate of 1000 inches per minute
                                     ; is established.
CNT=0.                               ; No Characters are processed at this time.
(MALC,<3,100>)                       ; Allocate the receive buffer for 100 bytes.
(PORT,A)                             ; Open RS-232, Port A.
(DENT,GETC)                          ; Poll for characters to be received.
(JUMP,CHK,CNT.NE.$POT<0>)            ; If character is received, jump to CHK
(JUMP,GETC)                          ; otherwise, continue to look for it.
(DENT,CHK)                           ; Characters received, check if good.
CNT=CNT+ 1.                          ; Adjust characters processed count.
TMP=CNT                              ; Use TMP to hold the array index.
(JUMP,CHK2,CNT.LT.100)               ; Jump to CHK2 if not the last entry of array.
                                     ; Allocated memory is full.
(PORT,A)                             ; Turn ON again to reset the Pointer.
CNT=0.                               ; Reset the count.
(DENT,CHK2)                          ; Check for "1","2","3".
(JUMP,RX1,$POT<TMP>.EQ.H,31)         ; Jump to RX1 if byte received is a "1",
(JUMP,RX2,$POT<TMP>.EQ.H,32)         ; Jump to RX2 if byte received is a "2",
(JUMP,RX3,$POT<TMP>.EQ.H,33)         ; Jump to RX3 if byte received is a "3",
(JUMP,GETC)                          ; otherwise, continue to look for them.
(DENT,RX1)                           ; A "1" is received.
X10.                                 ; Move the X axis 10 inches in the positive
                                     ; direction.
(JUMP,GETC)                          ; Look for more characters.
(DENT,RX2)                           ; A "2" is received.
X-10.                                ; Move the X axis 10 inches in the negative
                                     ; direction.
(JUMP,GETC)                          ; Look for more characters.
(DENT,RX3)                           ; A "3" is received.
M2                                   ; End of The Program
```

PORT

NOTES:

Prior to using the **PORT** command, an appropriate amount of memory must have been allocated by the **MALC** <3n>command (Each n = 1 byte).

RELATED COMMANDS:

$POT, **MALC**

PUSH

NAME:

PUSH - Push Data Into User's Stack

FUNCTION:

The **PUSH** command is used to store data in the program stack.

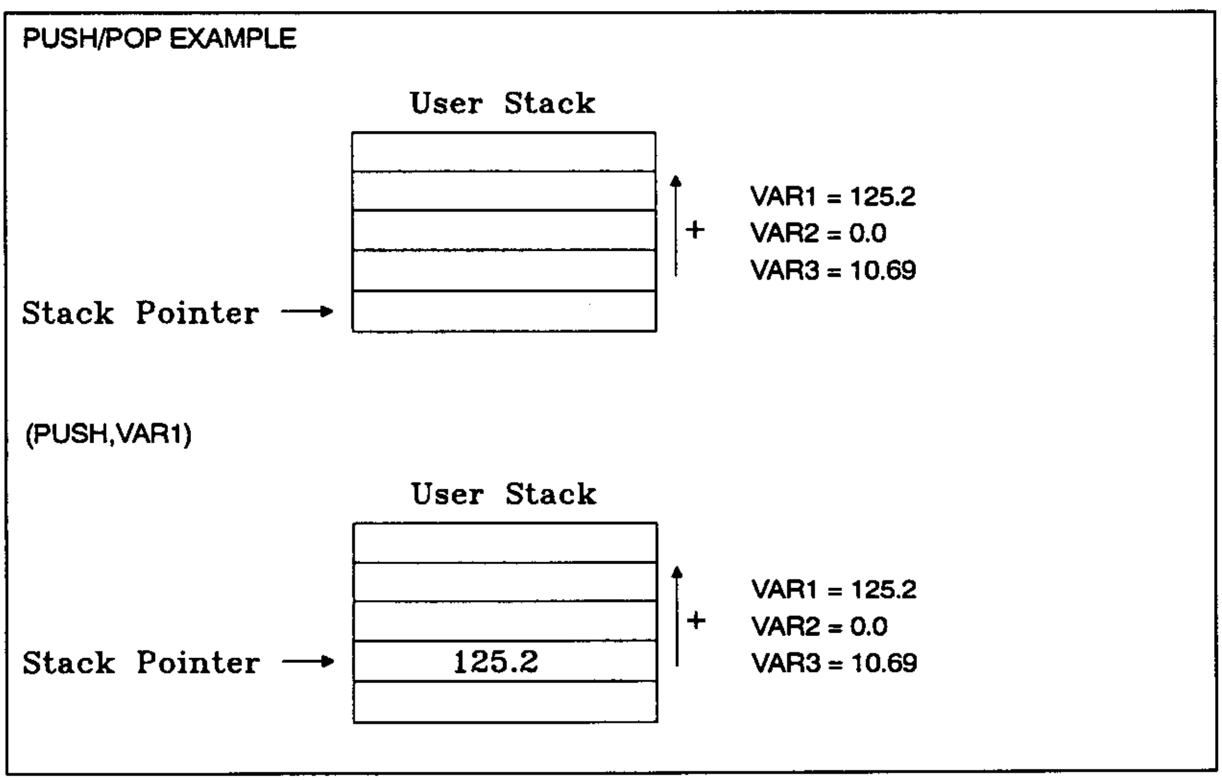
FORMAT:

(**PUSH**,data1,data2)

Data is stored in the data stack in a "last in - first out" basis.

RETURNS:

EXAMPLE:



PUSH

EXAMPLE (CON'T):

(PUSH,VAR2,VAR3)

User Stack

Stack Pointer →	10.69
	0.0
	125.2

+

VAR1 = 125.2
VAR2 = 0.0
VAR3 = 10.69 ; Stack pointer increments are positive.

(PUSH,2.75)

User Stack

Stack Pointer →	2.75
	10.69
	0.0
	125.2

+

VAR1 = 125.2
VAR2 = 0.0
VAR3 = 10.69 ; Stack pointer increments are positive.

(POP,VAR1)

User Stack

Stack Pointer →	10.69
	0.0
	125.2

+

VAR1 = 2.75
VAR2 = 0.0
VAR3 = 10.69 ; Stack pointer decrements after pop command.

(STKP,-2)

User Stack

	10.69
	0.0
Stack Pointer →	125.2

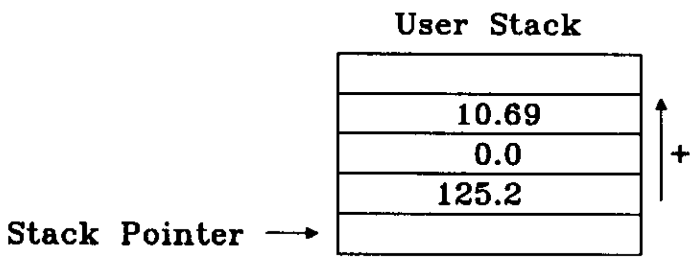
+

VAR1 = 2.75
VAR2 = 0.0
VAR3 = 10.69 ; Stack pointer decrements 2 positions.

PUSH

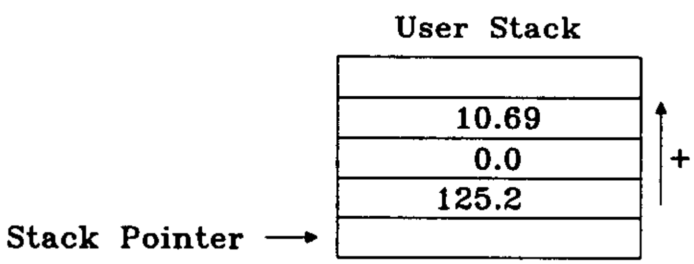
EXAMPLE (CON'T):

(POP,VAR3)



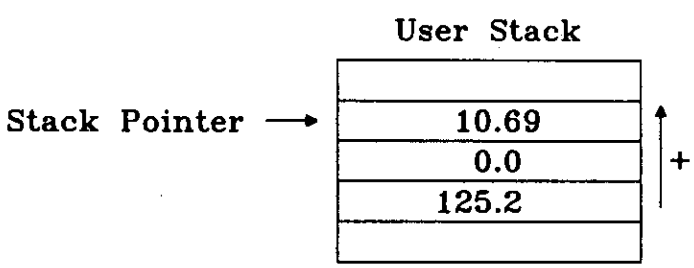
VAR1 = 2.75
VAR2 = 0.0
VAR3 = 125.2 ; Stack pointer decrements one position.

VAR2=H,FFFF



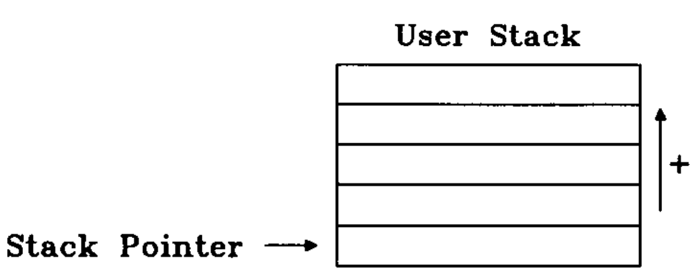
VAR1 = 2.75
VAR2 = H,FFFF
VAR3 = 125.2

(STKP,+3)



VAR1 = 2.75
VAR2 = H,FFFF
VAR3 = 125.2

(POP,VAR3,VAR2,VAR1)



VAR1 = 125.2
VAR2 = 0.0
VAR3 = 10.69

PUSH

NOTES:

Each data string stored by the **PUSH** command requires 4 bytes of User programmable memory. Program stack size (bytes) is established within the Parameter Mode. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

RELATED COMMANDS:

POP, STKP

RAMP

NAME:

RAMP - Define Axis Ramping time

FUNCTION:

The **RAMP**command is used to globally set the Ramping time of all axes, overriding the default Main System Parameter #30 setting established under the Individual Axis Parameter Mode (see the *Unidex 21 User's Manual* for a detailed description). This command is modal, and remains in effect until a subsequent **RAMP** command is issued, or the system is reset. Ramp time, as implemented in the Unidex 21 is the Acceleration/Deceleration time utilized during contouring or coordinated axes motions such as **G1**, **G2**, **G3**, or **G5**. The ramped trajectory may be either modified parabolic or linear and can also be affected by utilizing the Digital Filter Main System Parameter #60 (see also the *Unidex 21 User's Manual*) or through the **FILT** command. **G8** commands with feedrate charges utilize the Ramp time to accelerate or decelerate to the next velocity.

FORMAT:

(**RAMP**,ramping time)

Ramp values may be designated in two ways:

without a "."	; The Ramping time value will be in milliseconds.
with a "."	; The Ramping time value will be in seconds.

RETURNS:

EXAMPLES:

(**RAMP**,1000)	; The Ramping time of all axes will be 1000 msec. (no decimal point)
(**RAMP**,1.000)	; The Ramping time of all axes will be 1.0 sec. (decimal point used)
(DVAR,RMP)	; Define the variable RMP.
RMP=2.5	; Set the value of the variable RMP.
(**RAMP**,RMP)	; Utilize the variable to set the Ramp time of all axes to 2.5 seconds.

RAMP

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #2 located in Appendix 3 of this manual.

NOTES:

1) Ramping time must be 1 to 32,767 msec.

2) The Ramp time is used for coordinated axis motion. Uncoordinated motion commands (**FXOF**, **G0**, **HOME**, **MORG**) use the **ACDE** rate instead.

RELATED COMMANDS:

ACDE, COEF, DVAR, FILT, FREE, G8, G9, G23, G24, TRAJ

RECO

NAME:

RECO - Record Axis Motion

FUNCTION:

The **RECO** command is used to initiate a Teach Mode during manual axes manipulation to sample axes positions. The positions are recorded in the User's memory. Typically, a User would disable the axes torque via the **MTOR** command, enable the recording function, and then move the axes. A good application for the usage of the **RECO** command is for recording the motions of a human machine operator, and then replaying the motion which has been recorded.

The hardware input signal from the triggering device is input at Joystick Port J2A. The signal at this port is monitored during each sample and may be configured to record only upon receipt of either a HIGH or LOW signal. The default is for recording to occur at a HIGH signal from Port J2A.

FORMAT:

(**RECO**<axis name,axis name,....>,rate,deadband,1/0)

or

(**RECO**,0)

axis name	The names of the axes whose position is to be recorded. Must be enclosed in < >.
rate	The frequency at which position sampling occurs. (1 to 255 msec)
deadband	The minimum number of Machine Steps of position change that is to be considered a position change.
1/0	Establishes the record signal from J2A as HIGH or LOW 1 - recording will occur only when J2A goes HIGH 0 - recording will occur only when J2A goes LOW

RETURNS:

RECO

EXAMPLE:

(RECO,<X,Y>,50,100,1)	; When the signal of J2A goes HIGH, the positions of the X and Y axis will be recorded every 50 msec. The deadband is 100 Machine Steps.
(RECO,0)	; Disable the position recording.

NOTES:

1) Prior to using the **RECO** command an appropriate amount of memory must have been allocated by the **MALC,<2,n>** command. (Each n = 256 bytes)

2) Position recording must be disabled before attempting to use the **PLAY** command.

3) To store the recorded memory to a file see the *Unidex 21 User's Manual* for a more complete description of the "File Mode".

RELATED COMMANDS:

MALC, MTOR, PLAY, REPT

REF

NAME:

REF - Send Axis to Hardware Home

FUNCTION:

The **REF** command is the same as the **HOME** command. Refer to the description of the **HOME** command contained in this manual.

FORMAT:

RETURNS:

EXAMPLE:

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Examples #5, and 6 located in Appendix 3 of this manual.

NOTES:

RELATED COMMANDS:

HOME

RETP

NAME:

RETP - Real Time Position Fetch

FUNCTION:

Sometimes it is necessary to record the axes position in synchronization with an external event. An example would be either flow detection in an ultrasonic test or pixel data acquisition from a camera system. The **RETP** command is used to initiate a Real Time Position Fetch. The Position Fetch event is triggered externally, by the User, through an I/O connector located on Unidex 21's rear panel (see Note 1). Data is latched in the Unidex 21's DSP Servo Control card with a latency of 3 to 8 microseconds for all 8 axes.

The position of the designated axis or axes is recorded in the User's memory (see Note 2), and may be transferred via RS-232 or disk for further analysis.

FORMAT:

(**RETP**,axis name,axis name,...)

RETURNS:

EXAMPLE:

```
(RETP,X,Y)          ; Enable a Real Time Position Fetch for the X and Y axes.

(RETP,0)            ; Disable a Real Time Position Fetch for all axes.
```

PROGRAMMING EXAMPLE:

The following program example shows Real Time Position Fetching with output to a data file.

```
(DVAR,ITER)         ; Define the local variable ITER.
(MALC,<2,100>)      ; Allocate memory for a Real Time Position function.
(RETP,X)            ; Enable a Real Time Position Fetch on the X axis.
G1 F100. X10.       ; Initiate a Linear Move.
(RETP,0)            ; Disable the Real Time Position Fetch.
```

RETP

PROGRAMMING EXAMPLE (CON'T):

```
(OPEN,W,TST.TMP)            ; Open the external file to write data to.
(MEND,W)                    ; Set the pointer to the top of the file.
ITER=1                      ; Set the iteration counter to 1.
(RPT,$RTP<0>                ; Repeat Loop
                            ; Note that the $RTP is a pointer to the Real Time Position
                            ; buffer. The first element in this buffer ($RTP<0>) indicates
                            ; the size of the buffer, subsequent buffer elements are the
                            ; stored axis position .If multiple axis positions are desired
                            ; i.e.,(RETP,X,Y) then the data is stored as follows for N
                            ; position fetches:
                            ;   $RTP<0> = Number of elements in buffer
                            ;   $RTP<1> = First position fetch X axis
                            ;   $RTP<2> = First position fetch Y axis
                            ;   $RTP<3> = Second position fetch X axis
                            ;   $RTP<4> = Second position fetch Yaxis
                            ;   || || || || || || ||
                            ;   $RTP<2N-1> = Nth position fetch X axis
                            ;   $RTP<2N> = Nth position fetch Y axis
(WRIT,#BTF<$RTP<ITER>>)     ; Write data out to the file. Note that the Position Fetch data
                            ; is stored internally in a Binary or Hex format.
ITER=ITER+1                 ; Point to the next element in the buffer.
)                           ;
(END,S)                     ; Close the file and save the data.
```

NOTES:

1) All User supplied interrupt signals must be debounced. This input is directly transferred into Unidex 21's DSP Servo Control card. Refer to a description of the miscellaneous I/O connector (P23) located on the Unidex 21 Interface panel.

2) Prior to using the **RETP** command an appropriate amount of memory must have been allocated by the **MALC**,<2,n> command. (Each n = 256 bytes)

RETP

NOTES (CON'T):

3) Real Time Position Fetch events may occur at 1 msec intervals maximum, even though the latency of acquisition time is only 3 to 8 microseconds. This is due to the data transfer rate limitations between the CPU and DSP card.

RELATED COMMANDS:

$HSI, DVAR, END, HSIE, MALC, MEND, OPEN, WRIT

ROTA

NAME:

ROTA - Parts Rotation

FUNCTION:

The **ROTA** command is used to define the plane and degree of angle for Part Rotation. The Parts Rotation programming feature permits the User to utilize a one shape Subroutine or move sequence and change the orientation without reprogramming or changing coordinate reference frames.

FORMAT:

(**ROTA**,axis plane for rotation,angle)

The angle required for entry must be converted to degrees, or may be a User variable.

A positive angle entry will produce a CCW angle and is referenced from the axes plane. The entry is modal such that rotation will be in effect until deactivated.

An angle entry of zero will deactivate Parts Rotation.

RETURNS:

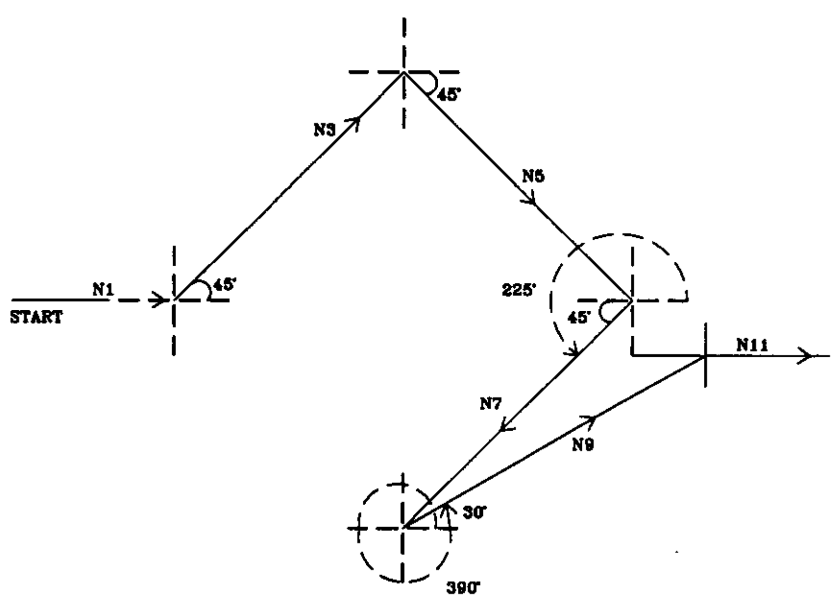
EXAMPLE:

(ROTA,X,Y,45)	; The Part Rotation angle will be set to 45° in the XY plane.
(ROTA,X,Y,VAR1)	; The Part Rotation angle will be set to the value of VAR1.
(ROTA,X,Y,0)	; Deactivate the Parts Rotation.

ROTA

EXAMPLE (CON'T):

LINE	COMMAND	DESCRIPTION
N1	G1 X5. F1000.	Move the X axis five units (in/mm) in the positive (CCW) direction.
N2	(ROTA,X,Y,45)	Set the X,Y Part Rotation angle to 45°.
N3	X5.	Move the X axis five units (in/mm) in the positive (CCW) direction.
N4	(ROTA,X,Y-45)	Set the X,Y Part Rotation angle to -45°.
N5	X5.	Move the X axis five units (in/mm) in the positive (CCW) direction.
N6	(ROTA,X,Y,225)	Set the X,Y Part Rotation angle to 225°.
N7	X5.	Move the X axis five units (in/mm) in the positive (CCW) direction.
N8	(ROTA,X,Y,390)	Set the Part Rotation angle to 390° (30°).
N9	X5.	Move the X axis five units (in/mm) in the positive (CCW) direction.
N10	(ROTA,X,Y,0)	Disable the X,Y Part Rotation.
N11	X5.	Move the X axis five units (in/mm) in the positive (CCW) direction.
N12	M2	End of The Program



ROTA

PROGRAMMING EXAMPLE:

```
G70                                    ; Initiate English Programming (inches).
(DVAR,ANG1,ANG2,VAR1,VAR2,
 VAR3,VAR5,INCA,INCB,INCX,INCY)        ; Define the listed variables.
G1 X1. Y3. F200.                       ; Linear Move for the X and Y axes.
VAR2= 2. INCX= 0. INCY= 0.             ; Initialize variables, VAR2 is radius (2).
ANG1=180                               ;
(RPT,2                                 ; Repeat the Loop once.
  INCA=INCX INCB=INCY                  ; X and Y Offset
  ANG2=ANG1-(360./20.)                 ; Angle 2 is the angle Offset.
  G91 G1 X=INCA Y=INCB F300.           ; Initiate Relative Positioning
                                       ; for X and Y = variables.
  (RPT,20                              ; Start the Repeat Loop.
    G1 X=INCX                          ; Initiate a Linear Move for X variables.
    (ROTA,X,Y,VAR5)                    ; The Part Rotation angle will be set to the
                                       ; value of VAR5.
    G1 G91 X.4                         ; Move the X axis in the positive direction.
    X-.2                               ; Move the X axis in the negative direction.
    Y-.2                               ; Move the Y axis in the negative direction.
    Y.4                                ; Move the Y axis in the positive direction.
    Y-.2                               ; Move the Y axis in the negative direction.
    X-.2                               ; Move the X axis in the negative direction.
    (ROTA,X,Y,0)                       ; Reset the Part Rotation angle.
    G2 L=VAR2 C=ANG1 D=ANG2            ; Position the angle such that:
                                       ;   L=Radius
                                       ;   C=Beg Angle
                                       ;   D=End Angle
    VAR1=360./20.                      ; Define the angle increment (18°).
    VAR5=VAR5-VAR1                     ; Establish a new angle for ROTA command.
    ANG2=ANG2-VAR1                     ; Increment the ending angle.
    ANG1=ANG1-VAR1                     ; Increment the beginning angle.
  )                                    ; Close the nested Repeat Loop.
```

ROTA

PROGRAMMING EXAMPLE (CON'T):

```
  VAR2=VAR2-.5                    ; Decrease the radius of VAR2.
  INCX=0 INCY=0                   ; Reinitialize the variables.
  G1 X-1. Y-3.                    ; Linear Move of the X axis in the negative
                                  ; direction, and the Y axis in the negative
                                  ; direction.
  G1 X1.5 Y3.25                   ; Linear Move of the X axis in the positive
                                  ; direction, and Y axis in the positive direction.
  VAR5=VAR5-9                     ; Decrement the variables.
  ANG1=ANG1-9                     ;
)                                 ; Close the outer Repeat Loop.
M0                                ; Program Stop
```

NOTES:

RELATED COMMANDS:

DVAR, MIR

RPT

NAME:

RPT - Repeat Loop

FUNCTION:

The **RPT** command is used to initiate a Repeat Loop. The repetition of the loop will be determined by the repeat count number via direct numeric data or a User variable. A Repeat Loop may also be nested within a Repeat Loop.

FORMAT:

```
(RPT,repeat count number or User variable
    data
    data
)
```

The **RPT** command and loop closure parentheses must be placed on separate lines, no other code can be inserted. The last line should contain the ending parentheses only.

RETURNS:

EXAMPLES:

```
(RPT,10                 ; The entire loop will be repeated 10 times.
  X1000.
)                       ; Close the Repeat Loop.
```

Repeat loops may also be nested together as follows:

```
(RPT,8                  ; The outer loop will be repeated 8 times.
  G1 X1000. Y1000. F100.
  (RPT,VAR1             ; The inner (nested) loop will be repeated based upon the value
                          of User variable VAR1.
    Z1500.
    X-1000. Y-1000.
  )                     ; Close the nested Repeat Loop.
)                       ; Close the outer Repeat Loop.
```

RPT

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #2 located in Appendix 3 of this manual.

NOTES:

The Repeat function requires 16 bytes of stack memory which is taken from the User RAM during program execution.

RELATED COMMANDS:

DVAR

RTRS

NAME:

RTRS - Retrace

FUNCTION:

The **RTRS** command provides the User with the ability to Retrace a program flow directly along the originally programmed path. The **RTRS** command must only be used with programs and/or program blocks that are repeatable in reverse. Refer to the notes on this command for specific programming considerations when using the Retrace function. This function may also be activated from the membrane panel retrace key or <cntrl> R from an AT style keyboard. When enabled in this manner, the Unidex 21 automatically forces the system into a single block mode of operation.

FORMAT:

(**RTRS**,enable/disable)

The Retrace function is enabled by placing a non-zero number next to the command.

The Retrace function is disabled by placing a zero next to the command.

RETURNS:

EXAMPLE:

```
N1  (RTRS,0)              ; The program will run in a normal forward motion, N2 - N5.
N2  program block
N3  program block
N4  program block
N5  program block
N6  (RTRS,1)              ; Retrace will begin, program blocks N5 - N2 will be performed.
                            When N1 is reached, program will proceed forward, N2 - N5.
                            Back and forth motion will result.
```

RTRS

EXAMPLE (CON'T):

```
N1  (DVAR,VARB)          ; Defines the variable VARB.
N2  (RTRS,0)             ; Ensure can't retrace past this point.
N3  program block        ; Program will run in a normal forward motion
N5  program block        ; for blocks N3 - N8.
N6  program block        ;
N7  program block        ;
N8  program block        ;
N9  VARB= BTF($IN0)      ; $IN0 will determine if the program will retrace.
N10 (RTRS,VARB)          ; If $IN0 was HIGH, N9 - N2 will be retraced
N11 M2                   ; else, End of The Program.
```

NOTES:

Most commands may be used with the Retrace function and produce the desired results. The following items provide programming constraints that must be acknowledged prior to using the Retrace function for certain special commands..

1) The following commands provide the same functions during Retrace as they do during a standard program run.

HOME	See Note 7
REF	
RTRS	
SLEW	

2) During a Retrace the following commands are adjusted to their complementary function:

CLS	Following the Subroutine the return point will be one block above CLS command.
DFLS	Library Subroutine flow will be in reverse.
DFS	Subroutine flow will be in reverse.

RTRS

NOTES (CON'T):

POP	The PUSH function will be performed.
PUSH	The POP function will be performed.
RPT	Repeat Loop flow will be in reverse.
STKP	The stack pointer index sign will be reversed.

3) When the **$R** system command is encountered during a Retrace, the stack pointer is internally returned to the upper data set before the command is read. To maintain the correct data/name relationship each **$R** command should occupy its own program block. For Example:

VAR1=$R VAR2=$R	During Retrace VAR1 and VAR2 data will be correctly assigned.
VAR1=$R VAR2=$R	During Retrace VAR1 and VAR2 data will be assigned incorrectly.

4) To perform ICRC commands during a Retrace, the appropriate Retrace condition must be provided within the command. For example:

G40	The previous mode was G40.
G41 X1.	During initial program run the G41 command establishes a left cutter Offset of X1. During Retrace the Unidex 21 will read G40 in the previous motion block to deactivate ICRC with an end move of X1.
G41 X1.Y1.	G41 has been previously established for Retrace.
G40 X2.	During initial program run the G40 command will deactivate ICRC with an end move of X2. During Retrace the Unidex 21 will read the G41 command in the previous motion block to establish a right cutter Offset of X2.
G42 X2.	During initial program run the G42 command establishes a right cutter Offset of X2. During Retrace the Unidex 21 will read G40 in the previous motion block to deactivate ICRC with an end move of X2.

RTRS

NOTES (CON'T):

G42 X1. Y2.	G42 has been previously established for Retrace.
G40 X2.	During initial program run the G40 command deactivates ICRC with an end move of X2. During Retrace the Unidex 21 will read the G42 command in the previous motion block and establish a left cutter Offset of X2. (NOTE: While in Retrace the Unidex 21 treats G42 commands as G41 commands.)

5) The following commands provide the same function during Retrace as they do in a standard program run. Appropriate Retrace motion, however, will occur only if the command's complement is contained in the preceding program block. For example:

(SCO,X0,Y0) (SCO,X1,Y1)	Appropriate Retrace motion is provided without affecting the standard program run.

ACDE	**MTOR**
AFCO	**PID**
CCP	**PLC**
COEF	**PLNE**
COMM	**RAMP**
CPAG	**ROTA**
DRUN	**SCF**
FREE	**SCO**
HWEL	**SIOC**
INT1/INT2	**TERM**
LINK	**TRAJ**
MIR	**TRAK**
MSG	**UMFO**
MSTD	**ZONE**

RTRS

NOTES (CON'T):

6) The following commands cannot be contained in a program or program segment for Retrace since they are valid in one direction only:

ABTS	**MALC**
CFIT	**MEND**
ELPS	**OPEN**
END	**PLAY**
HSIE	**RECO**
JOIN	**RETP**
JUMP	**WRIT**

7) The following commands cannot be retraced because of the reasons provided:

FXOF	The UCO4 (A4) flags are confused in the Absolute Mode.
G92	When Position Registers are established, the previous program position is erased. Any Incremental (G91) moves that follows will be performed correctly but any Absolute (G90) move will not.
HOME, REF	When a HOME move is requested, the previous program position is erased. Any Incremental (G91) move that follows will be performed correctly but any Absolute (G90) move will not.
MORG	The UCO4 (A4) flags are confused in the Absolute Mode.
SCRB	Character scribing is a special function that cannot be performed in reverse.

RELATED COMMANDS:

SCF

NAME:

SCF - Scaling Factor

FUNCTION:

The **SCF** command provides a User with the ability to Scale Up or Scale Down an individual axis. Errors will not accumulate from move to move when the Scaling Factor is being utilized. The **SCF** command does not effect an inch to metric conversion. The Unidex 21 executes the Scaling Factor prior to making the conversion. Once **SCF** is set it must be enabled through the use of **SCO** command.

FORMAT:

(**SCF**,axis name and scaling factor)

The Scaling Factor must be a positive number ranging in value from .00001 to 99.99999.

RETURNS:

EXAMPLE:

(**SCF**,X0.33333,Y1.5)	; The Scaling Factor of the X axis is 0.33333, and the Scaling Factor of the Y axis is 1.5.
(**SCF**,X=VAR1,Y=VAR2)	; The Scaling Factor will be set to the values of variables VAR1 and VAR2.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #1 located in Appendix 3 of this manual.

NOTES:

1) The **SCF** command is modal.

2) The default Scaling Factor is 1.

SCF

NOTES (CON'T):

3) Circles cannot be scaled into the form of an ellipse through use of the **SCF** command. Only the individual points are affected. For example:

   G2 X2. Y4. I6. J8.

   becomes

   G2 X1. Y1. I3. J2.

   after

   (**SCF**, X.5, Y.25)

4) **G4** Dwell commands are not affected by scaling.

RELATED COMMANDS:
DVAR, ELPS, SCO

SCO

NAME:

SCO - Scaling ON/OFF Control

FUNCTION:

The **SCO** command provides a User the ability to enable or disable the Scaling Factor. Scaling is operative in both the Incremental and Absolute mode. When in the Absolute Mode, keep in mind that this function is referenced to the "Home" as established by **G92.** Prior to enabling **SCO**, the individual axes Scale Factor must be enabled by the **SCF** command.

FORMAT:

(**SCO**,axis name and enable/disable scaling function)

The Scaling function is enabled by placing a non-zero number next to the axis name.

The Scaling function is disabled by placing a zero next to the axis name.

RETURNS:

EXAMPLE:

(**SCO**,1)	; Enable all Scaling Factors.
(**SCO**,0)	; Disable all Scaling Factors.
(**SCO**,X0,Y1)	; Disable the X Scaling Factor, and enable Y Scaling Factor.
(**SCO**,X=VARA,Y=VARB)	; Enable or disable scaling on the XY axes based upon the value of variables VARA and VARB.

PROGRAMMING EXAMPLE:

For an example of how this command may be used, refer to Programming Example #1 located in Appendix 3 of this manual.

SCO

NOTES:

1) When an interrupt is enabled within a program, depending on the interrupt type, the program may skip the remainder of the program block. If this occurs, remember the current Machine Position and go to a Subroutine or entry point. It is then necessary that the User reestablish the Software Home before Scaling is reenabled. Reestablishing the Software Home is necessary since when the Unidex 21 returns from the Subroutine or entry point, it remembers the current Machine Position and the Scaling is no longer referenced to the prior Software Home.

2) The **SCO** command is modal.

3) The default is a disabled Scaling function.

RELATED COMMANDS:

DVAR, SCF

SCRB

NAME:

SCRB - Character Scribing

FUNCTION:

The Unidex 21 provides a Character Scribing facility as a standard feature. Block letters, numbers, and some ASCII characters are included. The **SCRB** command initiates Character Scribing. Prior to using this feature, the CHARACTER.$$$ file must be loaded into the User's memory (see Note 1). The following names are used within the CHARACTER.$$$ file and must NOT be redefined anywhere in the Main Program:

SCRB< >	Array name
SCRB	Entry name
S$RB	Subroutine name

FORMAT:

(**SCRB**,option,variable/text)

where

option 0 – outputs text
1 – converts variable to (±) 8 digit integer with leading zero
2 – converts variable to (±) 8 digit integer without leading zero

Character Scribing is capable of twenty characters per call.
The character ")" is scribed by using \)\.

RETURNS:

EXAMPLE:

(SCRB,AEROTECH)	; Each character of AEROTECH will be scribed.
(SCRB,1,VAR)	; Variable is converted to a ± 8 digit integer with a leading zero.
(SCRB,2,VAR)	; Each variable is converted to a ± 8 digit integer with no leading zero.

SCRB

NOTES:

1) The CHARACTER.$$$ file may be loaded into the User's memory from the Edit Mode's "Get File" function. The program will always exist in the Unidex 21 firmware, but is not accessed unless requested in order to save User RAM. The CHARACTER.$$$ file may be edited in the same manner as a program.

2) When using options 1 and 2, the variable is always interpreted as a floating point number. This number will be rounded to the nearest integer value.

RELATED COMMANDS:

DVAR, SCF, SCO

SIOC

NAME:

SIOC - System I/O Control

FUNCTION:

The **SIOC** command controls the system I/O by indicating either to wait or not to wait before processing I/O code. The Unidex 21 typically operates in a block lookahead mode which means that a non-motion command may be processed before an in-process move command is completed.

The I/O functions that are under **SIOC** control are:

$INn
$OTn
$000 to **$7FF**
$800 to **$F6F**
$F70 to **$FAF**

FORMAT:

(SIOC,n)

If n equals zero, the program will wait until the last block of instructions are complete.

If n does not equal zero I/O code is processed immediately after it is decoded.

RETURNS:

EXAMPLES:

```
(SIOC,0)
G1 X10. Y10.        ; The command block will be completed prior to
$OT7= H,1             processing the I/O function.

(SIOC,VAR1)
G1 X10. Y10.        ; The I/O function $OT7=H,1 will be processed before the
$OT7=H,1              command block completion if the value of variable VAR1
                      is non-zero.
```

SIOC

NOTES:

The default status of **SIOC** is zero, the last block of the program will be completed prior to processing the I/O code.

RELATED COMMANDS:

$INn, $OTn, $000-$FAF, DVAR, INT1/INT2

SKEY

NAME:

SKEY - Define Custom Softkey

FUNCTION:

The Unidex 21 provides two methods in which to program the Function Keys F1 - F8 for custom definition. The methods are as follows:

method 1– utilizes the Function Keys to perform batch mode programming, system level key strobe emulation, or User interrupt 1/2 activation. The settings of these keys are established under the Main System Parameter #301. These System Level Function Keys can be extremely useful, and if desired, multiple levels may be initiated by loading different parameter files. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

method 2– is implemented utilizing the **SKEY**, **SKLV**, and **SKYD** commands. This method provides a more enhanced way of programming the F1 - F8 Function Keys within a User's Parts Program. Up to 100 levels of each Function Key may be utilized to create totally menu driven operation. The Software Programmable Function Keys can be interactive with the System Level Function Keys (method 1 above). Also, a program containing function key commands may be joined into a Main Program via the **JOIN** command. Of course, variables may also be utilized within the Softkey **SKEY** commands.

FORMAT:

The explanation will be divided into two parts for easier reference, a Detailed Explanation and a Summary.

SKEY

DETAILED EXPLANATION:

(SKEY,intn#,label,title,key#,level#,function#)

where

intn#/label

intn# 0 – Delete this Softkey. Note that neither the entry field nor the title field is necessary when using this number. However, if these fields are not specified, spaces must be included within the command as placeholders. (See Example)

1 – Enable Softkey. Unidex 21 will not stop or abort the current move when jumping to the entry point specified by label. It will perform defined functions (I/O, mathematical, system variable input/output, or **MST**). Note that there is a great flexibility in the usage of the entry field with this option. The entry point format specified determines the option being used. Refer to the following examples.

a) **M,S,T** output is to be performed

entry point format: **MFxxxx** or **SFxxxx** or **TFxxxx**

where:

M,S,T specifies the appropriate output bus

and

xxxx specifies the value to be output on that bus. (Hexadecimal number)

b) Level is to be changed only;

By simply omitting the entry point (i.e., label) all together, detection of a Softkey will initiate a Softkey level change only (refer to function below). Note that a space is required as a placeholder for the entry point.

SKEY

DETAILED EXPLANATION (CON'T):

2 – Enable Softkey as **INTN2** user interrupt option. The label refers to the Subroutine name. Unidex 21 will stop the current move and abort any functions remaining in this block. The current Machine Position will be stored. Upon completion of the specified Subroutine, the Unidex 21 will return to the next block of the program.

The Subroutine specified must be defined elsewhere in the program using either the **DFLS** or **DFS** commands.

3 – Enable Softkey as **INTN3** user interrupt option. The label refers to the entry point location. Unidex 21 will stop the current move and abort any functions remaining in this block. It will then jump to the specified entry point.

The entry point specified must be defined elsewhere in the program using the **DENT** command.

4 – Enable Softkey as **INTN4** user interrupt option. The label refers to the Subroutine name. Unidex 21 will finish all of the functions in the current block before going to the specified Subroutine. Upon completion of the routine, the Unidex 21 will return to the next block of the program.

The Subroutine specified must be defined elsewhere in the program using either the **DFLS** or **DFS** commands.

5 – Enable Softkey as **INTN5** user interrupt option. The label refers to the entry point location. Unidex 21 will finish all of the functions in the current block before jumping to the specified entry point.

The entry point specified must be defined elsewhere in the program using the **DENT** command.

title – This specifies a character string which will be displayed at the F1 - F8 Function Keys. The title can be 1 to 15 characters in length.

SKEY

DETAILED EXPLANATION (CON'T):

key – Key number within the specified level. This number must be less than or equal to the number of keys per level (defined with the **SKYD** command).

Note that if there are more than 8 softkeys per level, the keys are displayed in groups of seven. Softkey number 8 (F8) is used to move between the different groups. The Unidex 21 will display

F8 -- ETC --

In this case, the softkeys are referenced 1-7 (F1-F7), 8-13 (F1-F7), etc., (i.e., F8 is not accessible). However, if there are exactly eight softkeys per level, F8 is available for general purpose use.

level – This is the level for which the specified key is to be active. This number must be less than or equal to the number of levels defined with the **SKYD** command.

function – This feature permits the User to change levels and/or change groups within the current level. The possible options are as follows:

option 0 – Execute only. No level change

1 – Execute and disable this Softkey. The key will be re-enabled any time this level is entered from a higher level, (Level 0 is the highest). It cannot be re-enabled from a lower level.

2 – Execute and move to next key group if available (i.e., if -- ETC -- is displayed).

3 – Execute and increment level number to lower level.

4 – Execute and decrement level number to higher level.

5 – Execute and move to level found on Softkey stack. Stack pointer is adjusted accordingly.

SKEY

DETAILED EXPLANATION (CON'T):

6 – Execute and reinitialize the Softkey stack pointer.

7 – Execute and blank out all softkeys. All machine sub-modes are disabled (refer to **SKLV**,N).

8 – Execute and re-enable system Function Keys. All machine sub-modes are re-enabled (Refer to **SKLV**,S).

9 – Reserved

10-109 – Execute and move to level 0-99 (i.e., function-10)

Expanded Functions

The expanded functions listed below have been included to give the User some control over the Softkey stack. Enhance the function # below with 3, 4, and 10 to 109 cases such that:

256 + function#	- Current level number is placed onto the Softkey stack. The stack pointer is adjusted appropriately.
512 + function#	- Current level number is placed onto the Softkey stack. The stack pointer is not adjusted.
768 + function#	- One Softkey stack entry is removed from the Softkey stack. The stack pointer is adjusted appropriately.
1024+ function#	- The Softkey stack is cleared, and the stack pointer is reinitialized.

For Example:

259	- Execute and increment level number to lower level. The previous level is stored on the Softkey stack. The stack pointer is adjusted.

SKEY

DETAILED EXPLANATION (CON'T):

516	- Execute and decrement level number to higher level. The previous level is stored on the Softkey stack. The stack pointer is not adjusted.
778	- Execute and move to level 0. Discard 1 Softkey stack entry. The stack pointer is adjusted.
1044	- Execute and move to level 10. The Softkey stack will be cleared, and the stack pointer reset.

To make this feature even more useful, the function number may be expressed as a variable. The value of this variable will be evaluated when the key is pressed. To utilize this feature, the variable name must be preceded by a #.

SUMMARY:

(SKEY, intn#,label,title,key#,level#,function#)

intn# 0 – Delete this key.

1 – For **MST** output, entry =MF/SF/TFxxx or level change only, entry =,, or same as INTN interrupt option, entry = subroutine 2/3/4/5 same as INTN interrupt option.

label – Subroutine label (INTN# 2/4)
or entry point (INTN# 3/5)
or don't care (INTN# 0/1 or , ,)
or (INTN# 1)

where

MFxxxx is the output M function Mxxxx
SFxxxx is the output S function Sxxxx
TFxxxx is the output T function Txxxx

SKEY

SUMMARY (CON'T):

title – 1 to 15 characters for F1 - F8.

key# – 1 to 100, if > 8, display in groups of 7 with F8 - ETC-.

level# – 0 to 99, 0 is highest level.

function – This feature permits the User to change levels and/or change groups within the current level. The possible options are as follows:

option 0 – Execute only. No level change

1 – Execute, then clear this key, re-enable when higher level is activated.

2 – Execute, then change to ETC-, if more than 8 keys.

3 – Execute, then move to lower level.

4 – Execute, then move to upper level.

5 – Execute, then move to level in stack.

6 – Execute, then do (SKLV,I).

7 – Execute, then do (SKLV,N).

8 – Execute, then do (SKLV,S).

9 – Reserved

10 to 109 – Move to level 0 to 99.

SKEY

SUMMARY (CON'T):

Expanded Functions

The expanded functions listed below have been included to give the User some control over the Softkey stack. Enhance the function # below with 3, 4, and 10 to 109 cases such that:

256*1 + function#	- Save the level to the stack, adjust the pointer.
256*2 + function#	- Save the level to the stack, do NOT adjust the pointer.
256*3 + function#	- Discard 1 stack, adjust the pointer.
256 * 4 + function#	- Clear the stack, initialize the pointer.

(SKLV,nn)	; Change to level nn, 0 - 99.
(SKLV,N)	; Black out Softkeys and machine system Function Keys, do not permit jog/mdi/auto/single/slew/run/feedhold/quit/retrace/abort/handwheel. Only permit cycle start/tracking display/error acknowledge.
(SKLV,S)	; Re-enable machine sub-mode and system Function Keys.

EXAMPLE:

This Softkey example illustrates the use of the Unidex 21 Softkey option. It uses three levels of softkeys with 14 keys per level (F1-F7, F1-F7). The User moves between the levels by pressing the desired Softkey.

```
(SKYD,14,3,128)     ; Define Softkey table to permit 3 Softkey levels with 14
                      softkeys per level. 128 bytes are allocated for the Softkey
                      stack.
(SKLV,I)            ; Ensure that the Softkey stack is initialized.
(SKLV,0)            ; Ensure that the current Softkey level is 0.
```

SKEY

EXAMPLE (CON'T):

```
LEVEL #0 GROUP #1
(SKEY,1, ,Laser Control,1,0,268)        ; F1 - Push the current Softkey level on
                                        ;      stack. Change to Softkey level #2.
                                        ;      Do NOT abort the current move.
(SKEY,1, ,Motor Control,2,0,3)          ; F2 - Increment to Softkey level #1.
                                        ;      Do NOT abort the current move.
                                        ; F3 - Not used
(SKEY,3,ENDP,Exit Program,4,0,7)        ; F4 - Jump to ENDP immediately. Don't wait
                                        ;      for the current block. Keep the Softkey
                                        ;      level the same. But delete all softkeys.
                                        ; F5 - Not used
                                        ; F6 - Not used
                                        ; F7 - Not used
                                        ; F8 - --ECT--

LEVEL #0 GROUP #2                       ; This level is not used

LEVEL #1 GROUP #1
(SKEY,4,GOX,Free Run X,1,1,0)           ; F1 - Call the Subroutine GOX after the
                                        ;      current block completes.
                                        ;      Do NOT change the Softkey level.
(SKEY,2,STPX,Stop Free Run X,2,1,0)     ; F2 - Call the Subroutine STPX now.
                                        ;      Do NOT wait til the block finishes.
                                        ;      Do NOT change the Softkey level.
                                        ; F3 - Not used
(SKEY,1, ,Exit Motor Control,4,1,4)     ; F4 - Exit laser control.
                                        ;      Change the Softkey level to that next
                                        ;      higher level (0). The current block is
                                        ;      not aborted
```

SKEY

EXAMPLE (CON'T):

```
LEVEL #1 GROUP #1 (CON'T)
(SKEY,1, ,Laser Control,5,1,259)         ; F5 - Push the current Softkey level on
                                         ;        stack. Increment to Softkey level
                                         ;        #2. Do NOT abort the current move.
                                         ; F6 - Not used
                                         ; F7 - Not used
                                         ; F8 - --ECCT--

LEVEL #1 GROUP #2
(SKEY,4,GOY,Free Run Y,8,1,0)            ; F1 - Call the Subroutine GOY after the
                                         ;        current block is completed.
                                         ;        Do NOT change the Softkey level.
(SKEY,2,STPY,Stop Free Run Y,9,1,0)      ; F2 - Call the Subroutine STPY now.
                                         ;        Do NOT wait till the block finishes.
                                         ;        Do NOT change the Softkey level.
                                         ; F3 - Not used
(SKEY,1, ,Exit Motor Control,11,1,4)     ; F4 - Exit the Laser Control. Change the
                                         ;        Softkey level to that next higher
                                         ;        level (0). The current block is not
                                         ;        aborted.
(SKEY,1, ,Laser Control,12,1,259)        ; F5 - Push the current Softkey level on stack.
                                         ;       Increment to Softkey level #2.
                                         ;       Do NOT abort the current move.
                                         ; F6 - Not used
                                         ; F7 - Not used
                                         ; F8 - --ECT--
LEVEL #2 GROUP #1
```

Note that this level may become active from either level 0 or from level 1. The previous level is found on the Softkey stack.

SKEY

EXAMPLE (CON'T):

```
LEVEL #2 GROUP #1 (CON'T)
(SKEY,1,MF80,Laser On,1,2,2)             ; F1 - Place H,80 on M-bus
                                         ;      then move to next Softkey
                                         ;      group. The level will not change.
                                         ;      Do NOT abort the current move.
                                         ; F2 - Not used
                                         ; F3 - Not used
(SKEY,1, ,Exit Laser Control,4,2,5)      ; F4 - Exit the Laser Control. Change the
                                         ;      Softkey level to that when pressed
                                         ;      Laser Control. The current block is
                                         ;      not aborted.
                                         ; F5 - Not used
                                         ; F6 - Not used
                                         ; F7 - Not used
                                         ; F8 - --ECT--

LEVEL #2 GROUP #2
                                         ; F1 - Not used
(SKEY,1,MF90,Laser Off,9,2,2)            ; F2 - Placed H,90 on M-bus then move
                                         ;      to the previous Softkey group.
                                         ;      Do NOT change the level.
                                         ;      Do NOT abort the current move.
                                         ; F3 - Not used
(SKEY,1, ,Exit Laser Control,11,2,5)     ; F4 - Exit the Laser Control and change
                                         ;      the Softkey level to that when
                                         ;      pressed Laser Control. The current
                                         ;      block is not aborted.
                                         ; F5 - Not used
                                         ; F6 - Not used
                                         ; F7 - Not used
                                         ; F8 - --ECT--
```

SKEY

EXAMPLE (CON'T):

```
BODY OF PARTS PROGRAM

(DVAR,VAR1)                          ; The Parts Program may contain any
(DENT,LOOP)                          ; commands necessary.
(MSG,<VAR1>,Enter 9 to exit now)     ;
G4 F.1                               ;
(JUMP,ENDP,VAR1.NE.9)                ;
(JUMP,LOOP)                          ;
(DENT,ENDP)                          ; Label for Program End
(FREE,X0,Y0)                         ; Ensure that both axes are OFF.
M90                                  ; Ensure that the laser is OFF.
(SKLV,S)                             ; Re-enable the machine sub-modes.
M2                                   ; End of The Program
                                     ;
(DFS,GOX                             ; Subroutine to start the X Free Run.
  (SKEY,0, , , 1,1,0)                ; Disable the Free Run Function Key "X".
  (FREE,X1,F10.)                     ; Start the X axis in continuous Free Run.
)                                    ;
(DFS,STPX                            ; Subroutine to stop the X Free Run.
  (FREE,X0)                          ; Stop free running on the X axis.
  (SKEY,4,GOX,Free Run X,1,1,0)      ; Reactivate the Free Run Function Key
                                     ; "X" again.
)                                    ;
(DFS,GOY                             ; Subroutine to start the Y Free Run.
  (SKEY,0, , ,8,1,0)                 ; Disable the Free Run Function Key "Y".
  (FREE,Y1,F10.)                     ; Start the Y axis in continuous Free Run.
)                                    ;
(DFS,STPY                            ; Subroutine to stop the Y Free Run.
  (FREE,Y0)                          ; Stop free running on the Y axis.
  (SKEY,4,GOY,Free Run Y,8,1,0)      ; Reactivate the Free Run Function Key
                                     ; "Y" again.
)                                    ; Label for Program End
```

SKEY

NOTES:

1) Prior to using the **SKEY** command a Softkey dimension must be defined utilizing the **SKYD** command.

2) The intn#, key#, level#, and function# can all be either BCD or variables where the value will be assigned when the **SKEY** command is decoded. A special case exists if the function # is programmed as "#VAR", which will assign the value when the key is actually pressed.

RELATED COMMANDS:

CPAG, DVAR, INT1/INT2, JOIN, SKLV, SKYD

SKLV

NAME:

SKLV - Change Custom Softkey level

FUNCTION:

This command permits the User to dynamically change the Softkey level, return the Function Keys to the system level, as well as clear the Softkey stack.

FORMAT:

(**SKLV**,option)

where

option

0-99 – Change to level option 0-99.

N – Disable all softkeys and machine mode sub-modes.

NOTE: No; jog/mdi/auto-single/slew/run/feedhold/home/quit/retrace/abort/handwheel is permitted. Only cycle-start/track/error ack are permitted.

S – Re-enable the machine mode sub-modes and system Function Keys.

I – Clear the Softkey stack, initialize the stack pointer.

RETURNS:

EXAMPLES:

(SKLV,0)	; Set Softkey level to 0. (highest)
(SKLV,10)	; Set Softkey level to 10. (assuming level defined)
(SKLV,N)	; Blank out all softkeys and disable machine sub-modes.

SKLV

EXAMPLES (CON'T):

(SKLV,S)	; Re-enable the softkeys and machine sub-modes.
(SKLV,I)	; Clear the Softkey stack and reinitialize the stack pointer.

NOTES:

Refer to the **SKEY** command for a more comprehensive example, and further explanation.

RELATED COMMANDS:

DVAR, SKEY, SKYD

SKYD

NAME:

SKYD - Define Softkey table

FUNCTION:

The **SKYD** command, in conjunction with the related commands, permits the User to modify the actions taken when a Function Key is pressed from within a User Parts Program. The **SKYD** command allocates the memory needed to hold the Sotfkey table, as well as the Softkey stack.

FORMAT:

(SKYD,K,L,S)

where

K - specifies the number of softkeys per level (range: 1 - 100)

L - specifies the number of Softkey levels (range: 1 - 100)

S - specifies the number of bytes to reserve for the Softkey stack.

EXAMPLE:

```
(SKYD,10,10,100)          ; Defines the Softkey table with 10 keys per level and 10 levels.
```

In this example, 100 bytes were reserved for the Softkey stack.

NOTES:

1) The amount of User memory allocated is (K * L * 12) + bytes (i.e., each table entry requires 12 bytes).

2) The levels defined by this command are referenced 0 - (L - 1).

3) Refer to the **SKEY** command for a more comprehensive example, and further explanation.

RELATED COMMANDS:

DVAR, SKEY, SKLV

SLEW

NAME:

SLEW - Joystick/Teach Pendant/Trackball/Mouse Slew

FUNCTION:

The **SLEW** command enables the Joystick, Teach Pendant, Trackball, or Mouse to be called from within either a User's Parts Program or MDI Mode. Alternatively, in the Machine Mode the User can manually select the **SLEW** function. Or, when running a Parts Program in Single Mode, the User may select **SLEW** (or any other Machine Mode sub function) and then continue with the Parts Program execution.

In many instances, a User needs to perform an initial part alignment or acquire data points within a User Parts Program. The **$nAP** and **$nRP** system variables can be utilized to set the value of User defined variables for Part Program usage. This is different from digitizing, where a User may create a Parts Program by moving the axes to different points and recording the positional data. Please refer to digitizing in the "Edit Mode" section of the *Unidex 21 User's Manual.*

FORMAT:

(**SLEW**,Joystick/Trackball,active axes name)

RETURNS:

The Unidex 21 remains in the **SLEW** mode until the User quits by pressing either the button located on the Joystick handle or the middle button on a Trackball.

EXAMPLE:

```
(SLEW,X,Y,VERT,U)
      or
(SLEW,J,X,Y,Z,U)           ; Program run stops and axes X, Y, VERT, and U are active to
                             the Joystick control.

(SLEW,T,X,Y)               ; Program run stops and axes X and Y are active to the
                             Trackball/Mouse control.
```

SLEW

NOTES:

1) If neither Joystick/Teach Pendant nor Trackball/Mouse is specified, the axes will be active to Joystick/Teach Pendant control. Both Joystick/Teach Pendant and Trackball/Mouse cannot be active at the same time. The handwheels may be utilized in conjunction with either Joystick/Teach Pendant or Trackball/Mouse.

2) An axis that is not included in the **SLEW** command is disabled from Joystick or Trackball control.

3) The applicable Joystick/Trackball Main System Parameters are #26 and 27. (See the *Unidex 21 User's Manual* for further explanation.)

4) The applicable Joystick/Trackball Individual Axis Parameters are #18, 21, 22, and 23 established under the 401-408 grouping. (See also the *Unidex 21 User's Manual.*)

RELATED COMMANDS:
$nAP, **$nRP**, **AFCO**, **HWEL**

SYNC

NAME:

SYNC - Synchronize Two Unidex 21's

FUNCTION:

The **SYNC** command is used to permit handshaking between two Unidex 21 controllers. Utilizing this command will provide beyond 8 axes of control as well as control program flow of a Master/Slave arrangement. The communications link used is the RS-232 Terminal Port. Therefore, both systems must have front panels. Typically, a User might also tie together Port B of the Master with Port A of the Slave such that remote communication control and file transfer facilities would also be available.

FORMAT:

(**SYNC**,option,timeout)

where

option 0 – Cancels Synchronization.

1 – Sends Synchronized code to the Slave.

2 – Waits for Synchronize code from the Master.

3 – Permits automatic Synchronization after each program block. That is, the Master controller will send **SYNC** code after executing each program block, and the Slave waits for a Synchronized code before executing another block. Both must be operating in this mode.

and

timeout – Specifies the number of seconds the Unidex 21 should wait for the Synchronization code from the remote system. If timeout value is equal to zero, the timeout feature is disabled (i.e., waits forever).

SYNC

EXAMPLE:

```
(SYNC,1,0)      ; Send SYNC code to the remote system.
(SYNC,2,10)     ; Wait for a SYNC code from the remote system.  If
                ; one is not received within ten seconds, notify the
                ; User of the error.
(SYNC,2,0)      ; Wait indefinitely for a SYNC code from the remote
                ; system.
(SYNC,3,5)      ; Synchronize with remote system after each program
                ; block.  If feedback is not received within 5 seconds
                ; of completing the current block, notify the User of
                ; the error.
(SYNC,0,0)      ; Cancel the Synchronization.
```

NOTES:

1) The two units being Synchronized must have the **SYNC** code Main System Parameter set to a unique value (0/1). (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

2) When using options 0 and 1, the timeout period must be specified, but has no effect on command operation.

3) The wait for the **SYNC** code command, option 2, ignores all characters received previously. Therefore, if the Master sends the **SYNC** code before the Slave begins waiting for it, Synchronization will be lost.

RELATED COMMANDS:

STKP

NAME:

STKP - User's Stack Pointer Adjust

FUNCTION:

The **STKP** command provides the User with the ability to change the pointer location within the program stack.

When data is stored within the program stack by the **PUSH** command, the pointer is always located at the last stack entry. If a **POP** command is initiated, the data at the pointer is retrieved.

If the User is knowledgeable as to the location of the data in the stack, the **STKP** command in conjunction with the **POP** command may be used to retrieve data that is located at some point within the stack.

FORMAT:

(**STKP**,+/-nn)

The value selected for nn will be internally multiplied by 4.

RETURNS:

New program stack pointer.

EXAMPLE:

```
(STKP,+2)        ; Move the pointer down the stack 2*4 bytes.

(STKP,-3)        ; Move the pointer up the stack 3*4 bytes.
```

A more complete example is found with the **PUSH** command instruction.

NOTES:

RELATED COMMANDS:

POP, PUSH

TERM

NAME:

TERM - Terminal Display for MSG Function

FUNCTION:

The **TERM** command is used in conjunction with the **MSG** command to utilize the lower portion of the machine mode's display. The **MSG** command normally displays four lines only.

Enabling the **TERM** command disables the Tracking Display. The User may then use the lower part of the display below that which is used for the machine mode's function display.

FORMAT:

(**TERM**,nn)

If nn is zero the **MSG** display is limited to four lines only.

If nn is a non-zero number **MSG** may use the lower portion of the machine mode display.

RETURNS:

EXAMPLE:

(TERM,0)	; The MSG display is limited to four lines.
(TERM,1)	; The MSG may use the entire display.

NOTES:

The default setting is (**TERM**,0).

RELATED COMMANDS:

CPAG, MSG

TRAJ

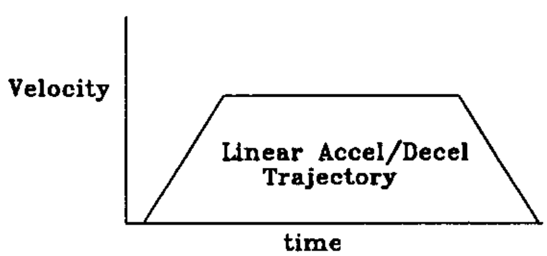
NAME:

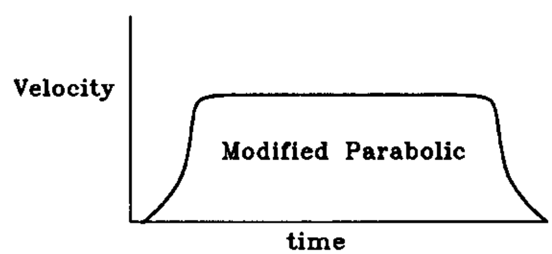
TRAJ - Axis Acceleration/Deceleration Trajectory, Type Selection

FUNCTION:

The **TRAJ** command permits the selection of either Linear Trajectory or Modified Parabolic Trajectory, overriding the Axis Parameter Mode setting. The parameter values are not changed by this command. The values established by this command will be in effect until a subsequent **TRAJ** command, or the system is reset, at which time the parameter values are re-instated.

Normally, the User will want to have all axes set to the same trajectory type to provide fully coordinated motion. The Modified Parabolic type of Acceleration/Deceleration is often referred to as "anti-jerk". The Parabolic Trajectory Coefficient may also be changed through the **COEF** command which changes the linear velocity/time slope of the Parabolic Acceleration/Deceleration.





FORMAT:

(**TRAJ**,axis name and linear/parabolic,axis name and linear/parabolic....)

The trajectory will be Linear if a zero is placed next to the axis name.

The trajectory will be Parabolic if a non-zero number is placed next to the axis name.

RETURNS:

TRAJ

EXAMPLE:

(TRAJ,X0,Y1,Z=VAR1,U=$INn)	; X axis has Linear Trajectory, Y axis has a Parabolic Trajectory, Z axis trajectory is dependent on value of VAR1, and U axis trajectory is dependent on Input bit "n" (1 or 0).
(TRAJ,0)	; All axes have Linear Trajectory.
(TRAJ,1)	; All axes have Parabolic Trajectory.

NOTES:

1) This command only affects coordinated motion commands such as **G1**, **G2**, **G3**, **G5**, **G8**, and **G9**, uncoordinated axis motion commands always use Parabolic Trajectory.

2) The applicable Main System Parameters are #57 and 59. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

   The applicable Individual Axis Parameter is #38 established under the 401-408 grouping. (See also the *Unidex 21 User's Manual* for further explanation.)

3) Usage of the **FILT** command or values in parameters #60 (exponential filter) may influence the actual trajectory under **G23** mode.

RELATED COMMANDS:

COEF, DVAR, FILT

TRAK

NAME:

TRAK - Position Tracking Display Control

FUNCTION:

The **TRAK** command permits the User to enable or disable the Position Tracking Displays for all the axes. This will speed up the system Parts Program execution by eliminating the update of the Position Tracking Displays which occurs approximately every 200ms. Under normal circumstances, it is not necessary to utilize **TRAK** for this purpose.

The **TRAK** command is also useful when the User wishes to display messages or gather input within a Parts Program, but does not want to have the operation confused by the scrolling of the Parts Program blocks.

FORMAT:

(**TRAK**,n)

If n= 0 – the Auto-Tracking Display will be OFF.

n= 1 – the Auto-Tracking Display will be ON.

n= 2 – the Auto-Tracking Display will be ON, however the Parts Program will not be displayed as it is run.

RETURNS:

EXAMPLE:

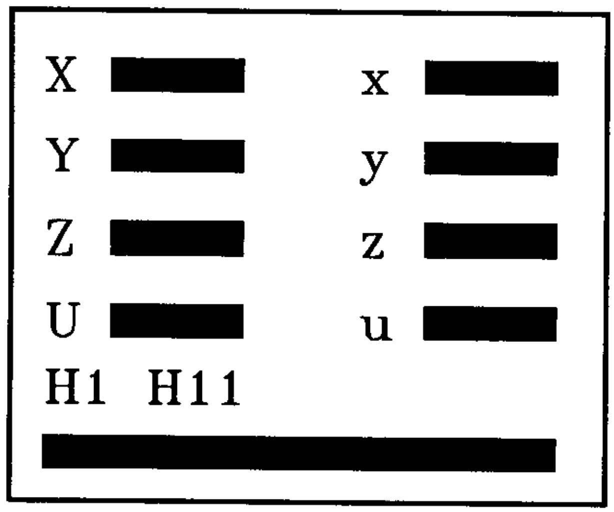
```
(TRAK,1)          ; The Auto-Tracking Display will be ON as the program is run.

(TRAK,VAR1)       ; Tracking Display option is set as per value of VAR1.
```

TRAK

NOTES:

Disabling the Tracking Display can free up the CPU for time which may be needed for processing motion commands.



NOTE: Shaded areas affected by TRAK command.

RELATED COMMANDS:

DVAR, MSG

UMFO

NAME:

UMFO - User Defined MFO Setting

The **UMFO** or Manual Feedrate Override command allows the User to vary the programmed feedrate up or down based upon a percentage, where 100% equals the programmed feedrate. Sometimes during a critical production process it is desired to prevent an operation from inadvertently changing the feedrate.

FUNCTION:

The **UMFO** command permits the User to set the percentage of Feedrate Override, and at the same time disable the Unidex 21's Front Panel **MFO** (Manual Feedrate Override) key. If the Front Panel **MFO** key is disabled, the desired percentage of the maximum Feedrate (0 - 200) must be included with the **UMFO** command.

FORMAT:

(**UMFO**,enable/disable,feedrate)

The **MFO** key is enabled by placing a zero next to the command.

The **MFO** key is disabled by placing a non-zero number next to the command.

RETURNS:

EXAMPLE:

(**UMFO**,1,100) VAR1=75.5	; The MFO key is disabled and the Feedrate is set at 100%.
(**UMFO**,1,VAR1)	; The MFO key is disabled and the Feedrate is established in VAR1 (75.5%).
(**UMFO**,0)	; The MFO key is enabled.

UMFO

NOTES:

1) The value of **UMFO** is set as a percentage value between 0 and 199 as a direct numeric entry. For example:

    75.5 equals 75.5%

    while the **$MFO** System Variable expresses this differently as

    0.755 equals 75.5%.

2) **MFO** affects all axes at the same time, and may be set to not affect the Free Run axes. The applicable Individual Axis Parameter is #36 established under the 401-408 grouping. (See the *Unidex 21 User's Manual* for a detailed description of the Parameter Mode.)

3) **MFO** may also be set up to affect the handwheel scaling. The applicable Main System Parameter is #55 (see also the *Unidex 21 User's Manual* for further detail).

RELATED COMMANDS:
$MFO, FREE, HWEL

ZONE

NAME:

ZONE - Safe Zone

FUNCTION:

The **ZONE** command is used to establish areas that an axis (axes) is not to travel. Safe Zones are used to prevent the tool from coming into contact with fixtures and to ensure that limits beyond the workspace are not exceeded.

The number of Safe Zones to be included in a program must be fixed at the beginning of the program with the **DZON** command. Zone parameters are referenced from the Software Home.

FORMAT:

(**ZONE**,zone name,enable/disable,zone parameters)

A Zone is enabled by placing a non-zero number next to the zone name.

A Zone is disabled by placing a zero next to the zone name.

RETURNS:

EXAMPLE:

```
(DZON,3)                                ; Enables 4 sets of Safe Zones.

(ZONE,0,1,X-10.,X10.,Y-3.,Y3.)          ; Zone 0 is enabled and zone parameters are
                                          established.

(ZONE,2,0)                              ; Zone 2 is disabled.

(ZONE,3,VAR1,X=VARA,X=VARB,Y=ARY<2>,Y=ARY<3>)   ; Zone 3 is enabled and
                                                  zone parameters are es-
                                                  tablished by variables and
                                                  arrays.
```

ZONE

NOTES:

1) The Safe Zones are only active under **G** code commands.

2) Safe Zone limits are not in effect when motion is initiated with the Home, Free Run, and Jog/Slew commands. The User may want to utilize **LIMT** instead.

3) The **ZONE** command is modal and remains until the system is reset or powered down.

RELATED COMMANDS:

DZON

CHAPTER 4: UNIDEX 21 MATH PACKAGE

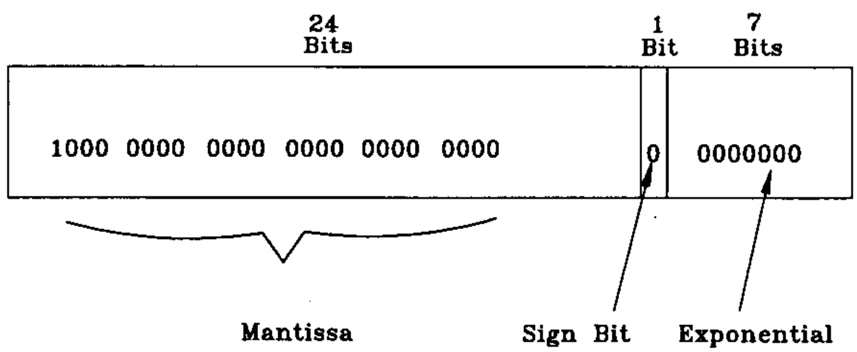
The Unidex 21 contains a powerful integral math package providing the User with full algebraic and logic capabilities which include addition, subtraction, multiplication, division, square root, trigonometric functions, and a variety of conversions.

SECTION 4-1: FLOATING POINT FORMAT

The Floating Point Format supports zero and non-zero values within the following range:

$$9.22337177 \times 10^{18} > \text{positive number} > 5.42101070 \times 10^{-20}$$

$$-9.22337177 \times 10^{18} > \text{negative number} > -2.71050535 \times 10^{-20}$$



Mantissa	The Mantissa consists of 24 bits normalized (Most Significant Bit is always 1) Binary data.
Sign Bit	The Sign Bit designates the positive (0) or negative (1) value.
Exponential	The Exponent consists of 7 Bits of Binary data. 41H exponent = 1 40H exponent = 0 3FH exponent = -1 41H to 7FH are positive exponents 01H to 3FH are negative exponents

SECTION 4-2: BINARY NUMBER FORMAT

The Unidex 21 processes Binary Numbers in the following formats:

Hex Numbers

H,dddddddd ; d = 0 to 9
A to F

Character Numbers

"ABCD" ; Each character is processed through ASCII code

Hex numbers may contain 8 digits maximum.

Refer to Appendix 4 (HEX NUMBERS AND EQUIVALENTS) for a decimal to Binary to Hex conversion chart.

SECTION 4-3: LOGIC FUNCTIONS FOR BINARY NUMBERS

The Unidex 21 provides specific Logic functions for use with Binary format (Hexadecimal and Character) constants.

These functions are followed by a number (n). This number indicates the number of bytes (each byte contains 8 bits) that are effected by the function. (The Unidex 21 assigns four bytes of memory for each variable.) If no number follows a function, the default is 1, only Byte 1 will be effected by that function.

Bytes are arranged as follows:

MOST SIGNIFICANT BYTE | LEAST SIGNIFICANT BYTE

BYTE 4	BYTE 3	BYTE 2	BYTE 1

SECTION 4-4: FUNCTIONS AND CONDITIONS

The following pages contain operations and conditions that may be utilized within the Unidex 21 Math package. All of these operations and conditions may also utilize User variables in addition to direct numeric entry.

+

NAME:

+ - Addition of Floating Point Numbers

FUNCTION:

The + function is used for addition of two floating point numbers.

FORMAT:

X = n+p

RETURNS:

The result is a floating point number.

EXAMPLE:

VAR1=20.2+40	; Value of variable VAR1 will be 60.2.
VAR1=VAR2+VAR2	; Value of variable VAR1 will be the sum of VAR2 and VAR3 or 60.2.
VAR2=H,28	; Value of variable VAR2 will equal 40.
VAR1= BTF(VAR2)-20.2	; Value of variable VAR1 will equal 60.2.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

NAME:

- - Subtraction of Floating Point Numbers

FUNCTION:

The - function is used to do subtraction of two floating point numbers.

FORMAT:

X=n-p

RETURNS:

The result is a floating point number.

EXAMPLE:

VAR1=40-20.2	; Value of variable VAR1 will be 19.8.
VAR1=VAR2-VAR3	; Value of variable VAR1will be the difference of VAR2 and VAR3 or 19.8.
VAR2=H,28	; Value of variable VAR2 will equal 40.
VAR1=BTF(VAR2)-20.2	; Value of variable VAR1 will equal 19.8.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

*

NAME:

* - Multiplication of Floating Point Numbers

FUNCTION:

The * function is used to do multiplication of two floating point numbers.

FORMAT:

X=n*p

RETURNS:

The result is a floating point number.

EXAMPLE:

```
VAR1=2.2*4              ; Value of variable VAR1 will be 8.8.
VAR1=VAR2*VAR3          ; Value of variable VAR1 will be the product of VAR2 and VAR3
                          or 8.8.

VAR2=H,4                ; Value of variable VAR2 will be 4.

VAR1=BTF(VAR2)*2.2      ; Value of variable VAR1 will be 8.8.
```

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.LT.

NAME:

.LT. - Condition compares two Floating Point Numbers

FUNCTION:

The **.LT.** function compares two floating point numbers. If the value of the first number **is less than** the value of the second number, the result is "True" (1). If the value of the first number **is not less than** the value of the second number, the result will be "False" (0).

FORMAT:

floating point number1**.LT.**floating point number2

RETURNS:

The result is a floating point number "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1.LT.SIN<30>)	; Program flow will go to Entry Point ENT1 if VAR1 **is less than** SIN <30>.
VAR2=VAR1.LT.SIN<30>	; VAR2 equals H,1 if VAR1 **is less than** SIN<30>, otherwise VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.LE.

NAME:

.LE. - Condition compares two Floating Point Numbers

FUNCTION:

The **.LE.** function compares two floating point numbers. If the value of the first number **is less than or equal to** the value of the second number, the result is "True" (1). If the value of the first number **is not less than or equal to** the value of the second number, the result will be "False" (0).

FORMAT:

floating point number1**.LE.** floating point number2

RETURNS:

The result is a floating point number, "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1.LE.SIN<30>)	; Program flow will go to Entry Point ENT1 if VAR1 **is less than or equal to** SIN <30>.
VAR2=VAR1.LE.SIN<30>	; VAR2 equals H,1 if VAR1 **is less than or equal to** SIN<30>, otherwise VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.NOTn.

NAME:

.NOTn. - Converts Binary Numbers

FUNCTION:

The **.NOTn.** function converts each bit from one to zero or zero to one.

FORMAT:

.NOTn. Binary number ; Where n is the number of bytes to invert, starting with the least significant byte.

RETURNS:

The result is a binary number.

EXAMPLE:

```
VAR1=.NOT2.H,F32        ; Bytes 1 and 2 are "inverted".
                        ; Hex F32 converted to Binary is:
                        ;     B2            B1
                        ;  00001111      00110010
```

once complemented it becomes Hex F0CD which is:

```
                        ;     B2            B1
                        ;  11110000      11001101

VAR1=.NOT1.H,F32        ; Byte 1 is "inverted".
                        ; Hex F32 converted to Binary is:
                        ;     B2            B1
                        ;  00001111      00110010
```

once byte 1 is complemented it becomes Hex FCD which is:

```
                        ;     B2            B1
                        ;  00001111      11001101
```

Note that byte 2 was not converted in this case.

.NOTn.

NOTES:

1) Variables may also be utilized.

2) This function requires only one operand.

RELATED COMMANDS:

DVAR

.ANDn.

NAME:

.ANDn. - AND's two Binary Numbers

FUNCTION:

The **.ANDn.** function is used to "AND" two binary numbers. The table below illustrates the **.ANDn.** function:

1 **.AND.** 1 = 1
1 **.AND.** 0 = 0
0 **.AND.** 1 = 0
0 **.AND.** 0 = 0

The "AND" operation is performed on a bit-by-bit basis, as illustrated above.

FORMAT:

Binary number**.ANDn.**Binary number ; Where "n" is the number of bytes to "AND", starting with the least significant byte.

RETURNS:

The result is a binary number.

EXAMPLE:

```
VAR1=VAR1.AND2.VAR2        ; Value of variable VAR1 is equal to the value of VAR1 .AND.
                             VAR2.

                           ; If   VAR1 is H,8A or 10001010
                                  VAR2 is H,9C or 10011100
                             then VAR1 equals H,88 or 10001000.
```

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.ORn.

NAME:

.ORn. - OR's two Binary Numbers

FUNCTION:

The **.ORn.** function is used to "OR" two binary numbers. The table below illustrates the **.ORn.** function:

1 **.ORn.** 1 = 1
1 **.ORn.** 0 = 1
0 **.ORn.** 1 = 1
0 **.ORn.** 0 = 0

The "OR" operation is performed on a bit-by-bit basis, as illustrated above.

FORMAT:

Binary number.**ORn.**Binary number ; Where "n" is the number of bytes to "OR", starting with the least significant byte.

RETURNS:

The result is a binary number.

EXAMPLE:

```
VAR1=VAR1.OR.VAR2          ; Value of variable VAR1 is equal to the value of VAR1 .OR.
                             VAR2.

                           ; If    VAR1 is H,8A or 10001010
                                   VAR2 is H,9C or 10011100
                             then VAR1 equals H,9E or 10011110.
```

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.XORn.

NAME:

.XORn. - EXCLUSIVE OR'S two Binary Numbers

FUNCTION:

The **.XORn.** function is used to "EXCLUSIVE OR" two binary numbers. The table below illustrates the **.XORn.** function:

1 **.XORn.** 1 = 0
1 **.XORn.** 0 = 1
0 **.XORn.** 1 = 1
0 **.XORn.** 0 = 0

The "XOR" operation is performed on a bit-by-bit basis, as illustrated above.

FORMAT:

Binary number1**.XORn.**Binary number2 ; Where "n" is the number of bytes to "EXCLUSIVE OR", starting with the least significant byte.

RETURNS:

The result is a binary number.

EXAMPLE:

VAR1=VAR1**.XOR.**VAR2	; Value of variable VAR1 **is equal to** the value of VAR1 .XOR. VAR2.
	; If VAR1 is H,8A or 10001010 VAR2 is H,9C or 10011100 then VAR1 equals H,9G or 00010110.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.LSLn.

NAME:

.LSLn. - Logical Shift Left of Binary Numbers

FUNCTION:

The**.LSLn.** (Logical Shift Left) shifts a specified number of bits left. It requires two operands in a specified order.

FORMAT:

Binary number**.LSLn.**Binary number

The first operand provides the number being shifted. The second operand specifies the number of bits of the shift, and "n" indicates the number of bytes to shift across.

RETURNS:

The result is a binary number.

EXAMPLE:

VAR1=VAR1.LSL.H,3	; The Bits in Byte 1 are shifted to the left 3 spaces.
	; If Byte 1 is 10101010 the first 3 bits on the left (101) will be replaced by the next 3 (010). The empty spaces on the right result will be replaced with zeros. The result will be 01010000.
VAR1=VAR1.LSL2.H,3	; The 3 bits to be shifted left from Byte 1 will shift over to Byte 2. The 3 bits shifted out of Byte 2 will "fall off". The 3 empty spaces on the far right of Byte 1 will be replaced with zeros.

B4	B3	B2	B1
00000110	00100011	10010111	01101111

Following VAR1=VAR1.LSL2.H,3 VAR1 will be:

B4	B3	B2	B1
00000110	00100011	10111011	01111000

.LSLn.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.LSRn.

NAME:

.LSRn. - Logical Shift Right of Binary Numbers

FUNCTION:

The**.LSRn.** (Logical Shift Right) function shifts a specified number of bits right. In all other respects it is the same as the **.LSLn.** function.

FORMAT:

RETURNS:

EXAMPLE:

NOTES:

RELATED COMMANDS:

.ROLn.

NAME:

.ROLn. - Rotate Binary Numbers Left

FUNCTION:

The**.ROLn.** (Rotate Left) function shifts a specified number of bits left and moves the displaced bits to the far right. It requires two operands in a specified order.

FORMAT:

Binary number**.ROLn.**Binary number

The first operand provides the number being rotated. The second operand specifies the number of bits for the rotation, and "n" indicates the number of bytes to rotate across.

RETURNS:

The result is a binary number.

EXAMPLE:

VAR1=VAR1**.ROL.**H,3	; The Bits in Byte 1 are shifted to the left 3 spaces and the displaced Bits are "wrapped around" to the right. If Byte 1 is 10101010 the first 3 Bits (101) will be replaced by the next three (010). The empty spaces on the right are filled with the displaced Bits resulting in 01010101.
VAR1=VAR1**.ROL2.**H,3	; The 3 Bits to be shifted left from Byte 1 will shift over to Byte 2. The displaced Bits from Byte 2 will "wrap around" to the right, replacing the empty spaces in Byte 1.

B4	B3	B2	B1
00000110	00110111	01110010	11001010

Following VAR1=VAR1**.ROL2.**H,3 VAR1 will be:

B4	B3	B2	B1
00000110	00110111	10010110	01010011

.ROLn.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.RORn.

NAME:

.RORn. - Rotate Binary Numbers Right

FUNCTION:

The **.RORn.** (Rotate Right) function shifts a specified number of bits right. The displaced bits replace the empty spaces on the far left. In all other respects it is the same as the **.ROLn.** function.

FORMAT:

RETURNS:

EXAMPLE:

NOTES:

RELATED COMMANDS:

.ADDn.

NAME:

.ADDn. - Addition of Binary Numbers

FUNCTION:

The **.ADDn.** function is used to do addition of two binary numbers.

FORMAT:

Binary number1**.ADDn.**Binary number2 ; Where "n" indicates the number of bytes to add.

RETURNS:

The result is a binary number.

EXAMPLE:

```
VAR1=VAR1.ADD.VAR2        ; Variable VAR1 is added to variable VAR2.


 10111011                 ; Binary numbers are added in the conventional way
+01001011                   with the "carrying" of numbers. When ADDing binary
1|00000110                  numbers in an 8 Bit register, the last carried number
                            is dropped.
```

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.SUBn.

NAME:

.SUBn. - Subtraction of Binary Numbers

FUNCTION:

The **.SUBn.** function is used to do subtraction of two binary numbers.

FORMAT:

Binary number1**.SUBn.**Binary number2 ; Where "n" indicates the number of bytes to subtract.

RETURNS:

The result is a binary number.

EXAMPLE:

VAR1=VAR1.**SUB**.VAR2	; Variable VAR2 is subtracted from variable VAR1.
10111011 - 01001011 01110000	; Binary numbers are subtracted in the conventional way with the "borrowing" of numbers.
VAR1=VAR2.**SUB**.VAR1	; Value of variable VAR1 will be the difference between VAR2 and VAR1 or 10010000.
01001011 - 10111011 10010000	;

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.TSTA.

NAME:

.TSTA. - Test And of Binary Numbers

FUNCTION:

The **.TSTA.** (Test And) function tests several bits and produces a "True " (1) result only if all bits are tested true. This function requires two operands in a specified sequence.

FORMAT:

Binary number1**.TSTA.**Binary number2

The first operand provides the number being tested. The second operand specifies the bits that are to be tested.

RETURNS:

The result is a binary number, "1" for True, "0" for False.

EXAMPLE:

```
VAR1=H,3F.TSTA.H,11                  ; H,3F = 0011 1111
                                     ; H,11 = 0001 0001
                                     ; VAR1 equals H,1 - both Bits tested true (1)

(JUMP,ENT1,$INP.TSTA.H,12.EQ.H,1)    ; $INP = 0011 1010
                                     ; H,12 = 0001 0010
```

Both Bits tested true, therefore, the jump to ENT1 will be accomplished.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.TSTO.

NAME:

.TSTO. - Test Or of Binary Numbers

FUNCTION:

The **.TSTO.** (Test Or) function tests several bits and produces a "True" (1) result if any of the bits test true. This function requires two operands in a specified sequence.

FORMAT:

Binary number1**.TSTO.n.**Binary number2

The first operand provides the number being tested. The second operand specifies the number of bits to be tested.

RETURNS:

The result is a binary number, "1" for True, "0" for False.

EXAMPLE:

```
VAR1=H,35.TSTO.H,3                    ; H,35 = 0011 0101
                                      ; H,3 = 0000 0011
                                      ; VAR1 equals H1, one Bit tested true (1).

(JUMP,ENT1,$001.TSTO.H,6.EQ.H,1)      ; $001 = 0010 0101
                                      ; H,6 = 0000 0110
```

One Bit tested true, therefore, the jump to ENT1 will be accomplished.

```
$023=H,3F.TSTO.H,C0                   ; H,3F = 0011 1111
                                      ; H,C0 = 1100 0000
```

Data on I/O port $023 will be set to zero, the result is false and no Bits tested are true.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.HI.

NAME:

.HI. - Compare unsigned Binary Numbers

FUNCTION:

The **.HI.** function compares two unsigned binary numbers. The "Most Significant Bit" (MSB) is compared first. If both numbers have the same value (either a 0 or a 1) the next bit of both the numbers is compared. If the first number has a bit value of 1 and the second number has a bit value of 0, the result would be "True" (1), otherwise, the result is "False" (0).

FORMAT:

Binary number1**.HI.**Binary number2

RETURNS:

The result is a binary number, "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1.HI.H,30)	; Program flow will go to Entry Point ENT1 if variable VAR1 is greater than H,30.
VAR2=VAR1.HI.H,30	; VAR2 equals H,1 if VAR1 is greater than H,30 otherwise, VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.LS.

NAME:

.LS. - Compares unsigned Binary Numbers

FUNCTION:

The **.LS.** function compares two unsigned binary numbers. The "Most Significant Bit" (MSB) is compared first. If both numbers are of equal value (either a 0 or a 1) or the first number has a value of 0 and the second number has a value of 1 the result is "True" (1), otherwise, the result is "False" (0).

FORMAT:

Binary number1**.LS.**Binary number2

RETURNS:

The result is a binary number, "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1**.LS.**H,30)	; Program flow will go to Entry Point ENT1 if the Bits of VAR1 are equal to or less than H,30.
VAR2=VAR1**.LS.**H,30	; VAR2 equals H,1 if the Bits of VAR1 are equal to or less than H,30, otherwise VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

CHAPTER 5: SUMMARY

COMMAND SUMMARY:

The following list of commands may be used as a quick reference. Detailed information for each of the commands is available in Chapter 3 of this manual.

%	Program Title
/	Block Delete Operator
;	Comment Operator
$HSI	High Speed Interrupt Buffer
$INn	Input System Variable
$MFO	Manual Feed Override System Variable
$nAP	Absolute Position Register Variables
$nRP	Relative Position Register Variables
$OTn	Address Logic Output System Variable
$POT	RS-232 Port Receive Buffer
$R	Read command
$RTP	Real Time Position Buffer
$TOD	Time of Day System Variable
$000-$FAF	Extended Input/Output Capabilities
ABTS	Abort Subroutine
ACDE	Maximum Acceleration/Deceleration
AFCO	Auto-Focus
CCP	Cutter Compensation
CLS	Call Subroutine
COEF	Coefficient for Parabolic Trajectory Ramping
COMM	Communication through RS-232 Port A/B
CPAG	Customer Display Page
DARY	Define Array
DENT	Define Entry Point
DFLS	Define Library Subroutine
DFS	Define Subroutine
DRUN	Dry Run
DVAR	Define Variable

COMMAND SUMMARY (CON'T):

DZON	Define Safe Zone
ELPS	Ellipse Look-Up Table
F	Define Axis Feedrate
FILT	Digital Filter
FREE	Axis Free Run (enabled or disabled)
FXOF	Fixture Offset
G0	Point-to-Point Positioning at rapid traverse rate
G1/G11/H1/H11	Linear Contouring
G2/G12/H2/H12	CW Circular Contouring
G3/G13/H3/H13	CCW Circular Contouring
G4	Dwell
G5/G15/H5/H15	Three Point Interpolation of two dimensional arc for contouring
G8	Velocity Profiling (acceleration)
G9	Velocity Profiling (deceleration)
G17/H17	Axis Plane Designation for execution of Circular Interpolation
G18/H18	Axis Plane Designation for execution of Circular Interpolation
G19/H19	Axis Plane Designation for execution of Circular Interpolation
G23	Corner Rounding
G24	Non-Corner Rounding
G40	Deactivates Cutter Radius Compensation
G41	Activates Cutter Radius Compensation-Left
G42	Activates Cutter Radius Compensation-Right
G70	English Programming
G71	Metric Programming
G90	Absolute Position Programming
G91	Relative Position Programming
G92	Software Home
HOME	Hardware Home
HSIE	High Speed Interrupt Control

COMMAND SUMMARY (CON'T):

HWEL	Handwheel Control
I,J,K,P and i,j,k,p	Circular Interpolation parameters
INT1/INT2	User Interrupt Control
JOIN	Join Subprogram to Main Program
JUMP	Jump to User Defined Entry Block
LCD, OAB and lcd, oab	Polar coordinates for Circular Interpolation
LIMT	Set Software Limit
LINK	Link Multiple Axes to a Group
MALC	User's Memory Allocate
MIR	Mirror Image
MORG	Machine Origin
MSG	Display Screen Messages
M, S, T	Output Capability (16 bit outputs)
MSTD	MST Strobe/Ack Delay, also Output during INT1/INT2, Option 3
MTOR	Motor Current Command Control
OPEN, MEND, END, WRIT	Reads/Writes data to memory
PID	Sets Kp, Ki, Kd, Kf1, and Kf2 Values
PLAY	Play Back the Recorded Axes Motion
PLC	Programmable Logic Control Interface
PLNE	Define Circular Contour Plane
POP	Pull Data Out of User's Stack
PORT	RS-232 Port Background Data Collection
PUSH	Push Data Into User's Stack
RAMP	Define Axis Ramping time
RECO	Record Axis Motion
REF	Send Axis to Hardware Home
RETP	Real Time Position Fetch
ROTA	Parts Rotation
RPT	Repeat Loop
RTRS	Retrace
SCF	Scaling Factor
SCO	Scaling ON/OFF Control

COMMAND SUMMARY (CON'T):

SCRB	Character Scribing
SIOC	System I/O Control
SKEY	Define Custom Softkey
SKLV	Change Custom Softkey level
SKYD	Define Softkey table
SLEW	Joystick/Teach Pendant/Trackball/Mouse Slew
SYNC	Synchronize Two Unidex 21's
STKP	User's Stack Pointer Adjust
TERM	Terminal Display for MSG function
TRAJ	Axis Acceleration/Deceleration Trajectory, Type Selection
TRAK	Position Tracking Display Control
UMFO	User Defined MFO Setting
ZONE	Define Safe Zone

COMPARISON OPERATORS SUMMARY:

.EQ.	Equal
.NE.	Not Equal
.GT.	Greater Than
.GE.	Greater Than or Equal To
.LT.	Less Than
.LE.	Less Than or Equal To

FLOATING POINT OPERATORS SUMMARY:

+	Addition Operator
-	Subtraction Operator
*	Multiplication Operator
/	Division Operator
()	Operation Precedence
!	Exponentials
SIN	Sine Value
COS	Cosine Value

FLOATING POINT OPERATORS SUMMARY (CON'T):

TAN	Tangent
ATN	Arctangent
DEG	Radian to Degree Conversion
RAD	Degree to Radian Conversion
ABS	Absolute Value
SQR	Square Root
INT	Float to Integer Conversion (Rounding)
FTB	Float to Binary

BINARY OPERATORS SUMMARY:

.NOTn.	1's Complement
.ANDn.	And
.ORn.	Or
.XOR.	Excluive or
.LSLn.	Logical Shift Left
.LSRn.	Logical shift Right
.ROLn.	Rotate Left
.RORn.	Rotate Right
.ADDn.	Addition
.SUBn.	Subtraction
.TSTA.nnnn.	Test And
.TSTO.nnnn.	Test or
.HI.	Unsigned Comparison Higher
.LO.	Unsigned Comparison Lower
BTF	Binary to Float Conversion

SYSTEM VARIABLES SUMMARY:

$URP, $XRP, $YRP, $ZRP	Relative Position Variables
$uRP, $xRP, $yRP, $zRP,	
$UAP, $XAP, $YAP, $ZAP	Absolute Position Variables
$uAP, $xAP, $yAP, $zRP	

SYSTEM VARIABLES SUMMARY (CON'T):

$TOD	Time Of Day
$INn	Current Input(s)
$OTn	Output(s) (Write only)
$MFO	Manual Feed Override
$R	Read Command
$HSI	High Speed Interrupt Buffer
$RTP	Real Time Position Buffer
$POT	RS-232 Port Receive Buffer
$000-$7FF	Motorola I/O Channels
$800-$F6F	PLC Dual Port Memory Locations
$F70-$FAF	PAMUX I/O Channels

APPENDIX 1: CUTTER COMPENSATION (ICRC)

ICRC stands for Intersectional Cutter Radius Compensation. In cutting a workpiece, sometimes the radius of the cutter must be taken into consideration. For example, when an endmill is used to cut the sides of a workpiece, the center of the endmill follows the programmed path. The outside edge of the endmill cuts around the actual workpiece offset from the programmed path by the radius of the tool.

Cutter radius compensation is an option which allows the operator to program the center of the cutter in such applications, so that the outside edge of the endmill cuts along the programmed path. Without this option the operator would have to offset the actual piece dimensions with the radius of the tool. When it becomes necessary to program angles other than 90°, it is no longer just a radius offset. This option dramatically decreases the programming effort by handling all the axis offsets. Also, the **ICRC** option allows the same program to be used with tools of different diameters just by changing the tool diameter information with the **CCP** command.

COMMANDS:

There are 4 commands to use:

G40	- Cutter compensation/Offset, cancel
G41	- Cutter compensation-Left, turn ON
G42	- Cutter compensation-Right, turn ON
(CCP,AX1,AX2,DIAM)	- Axis pair and tool diameter

The commands **G41** and **G42** are oriented in the direction of cutter motion. Example:

G41	Causes the cutter to make a path to the left of the nominal path by the amount of the radius determined from the tool diameter. Left is relative to the direction in which the cutter is moving.
G42	Causes the cutter to make a path to the right of the nominal path by the amount of the radius determined from the tool diameter. Right is relative to the direction in which the cutter is moving.

COMMANDS (CON'T):

The following diagram illustrates when to choose the **G41** or **G42** command.

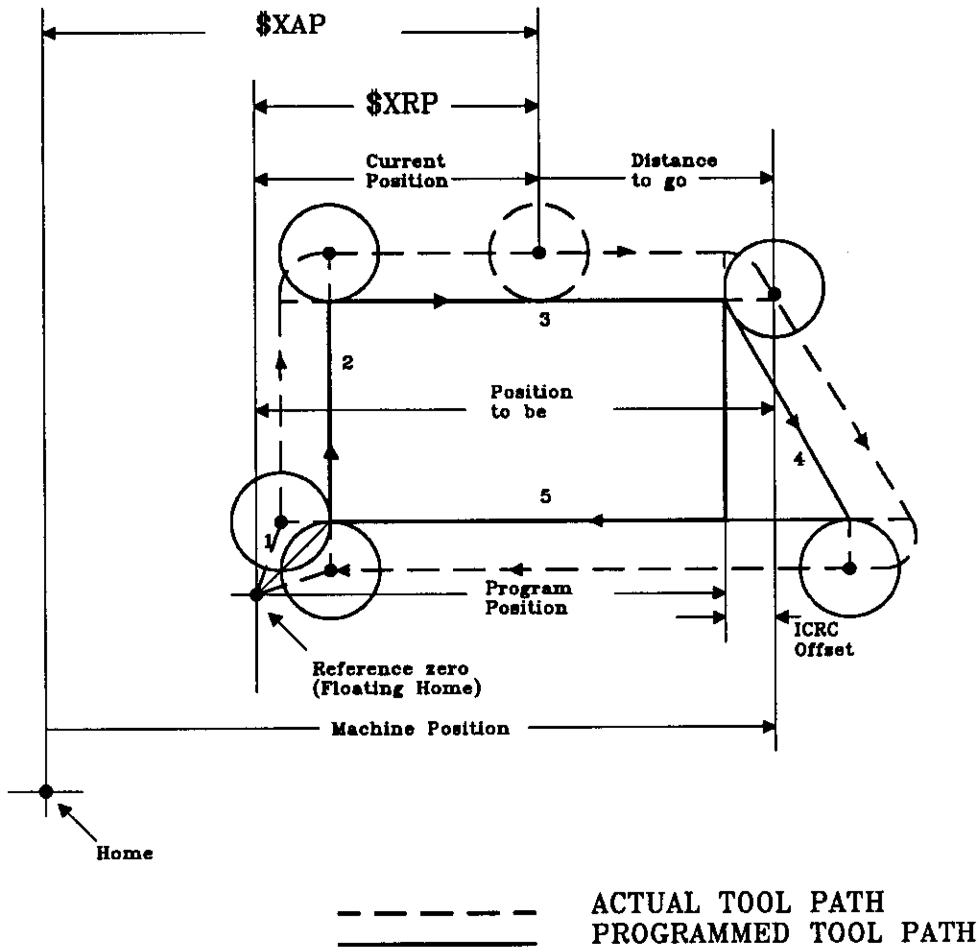
START-UP:

To start **ICRC**, first define the axis pair and tool diameter using the **CCP** command. Then enter a **G41** or **G42** followed by a start-up move before starting a cut. At this point, the operator can program the center of the tool to follow the workpiece contour.

As you can see, the solid line is the shape to be cut. The cutter center traces the dotted line, which is offset from the true shape of the workpiece by the amount of the radius.

The following example shows all X position registers, assuming the Unidex 21 is in the middle of move #3 at the dashed circle position. It also demonstrates the tool adjustments necessary to accommodate **ICRC**.

START-UP (CON'T):



When making a linear to linear move using **ICRC**, you may switch from **G41** to **G42** and vice versa with no problem.

For the sake of accuracy however, when making a linear to circular or circular to linear move and switching from **G41** to **G42**, a transitional move containing a **G40** (cancel **ICRC**) must be placed between the two.

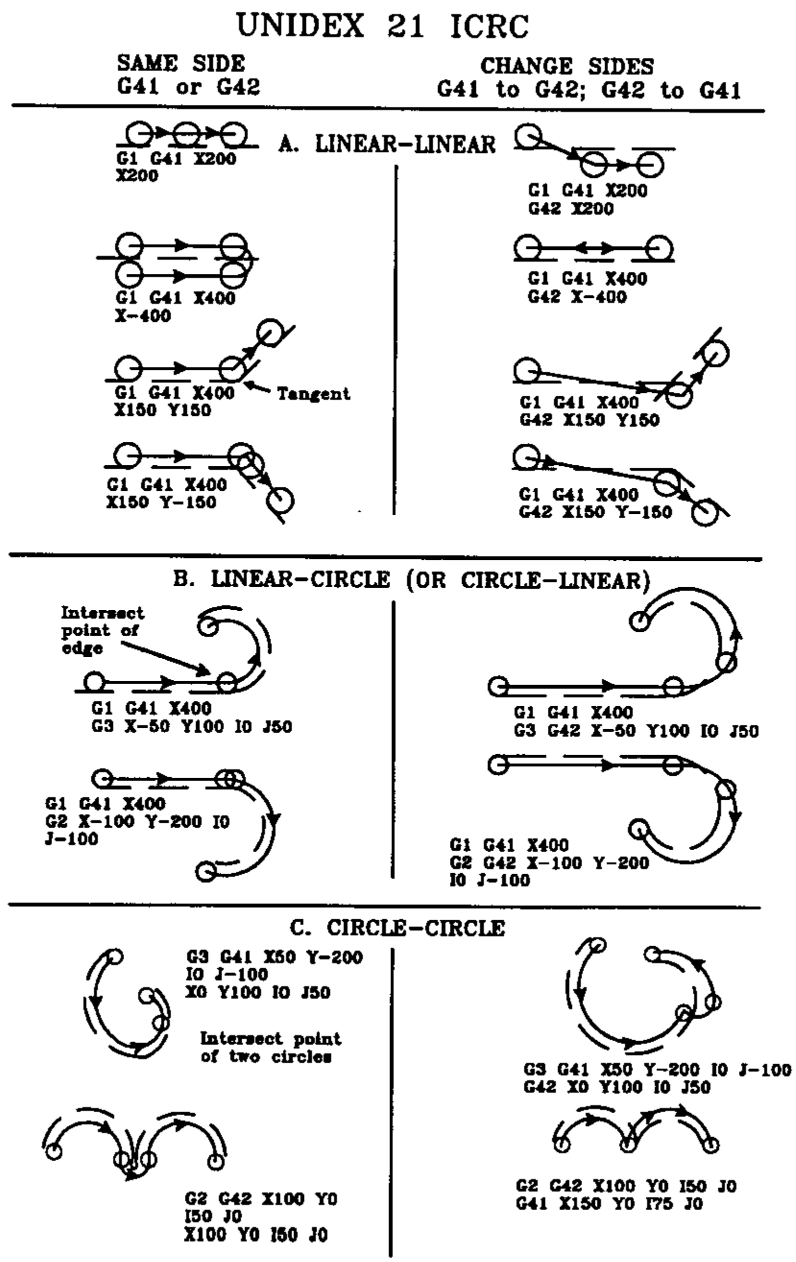
Another method of switching sides with **ICRC** is to break the transitional move (the one between linear and circular) into two moves. Execute one in **G41** and one in **G42**.

CANCELING ICRC:

To end **ICRC**, make an ending move to direct the cutter away from the workpiece after cutting, using the **G40** command to turn off the **ICRC.**

ICRC must be canceled before an **M2** command.

ICRC CONFIGURATIONS:



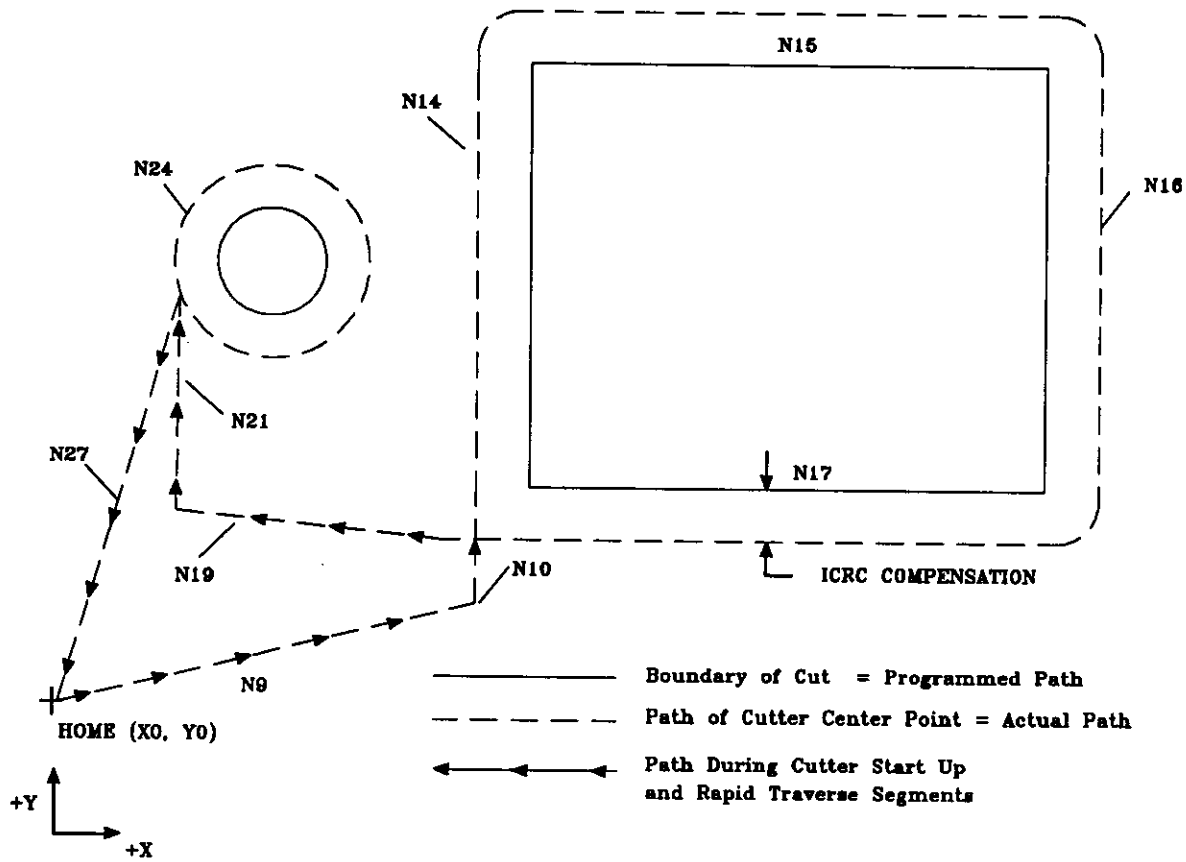
PROGRAMMING EXAMPLE:

The following Unidex 21 program cuts a rectangle and a circle from a material using a laser. Radius compensation **ICRC** is used to compensate for laser beam "kerf" (the width of the cut made by a tool such as a saw, torch, laser beam, or water jet). The **ICRC** compensation offsets are exaggerated for clarity.

An X-Y positioning stage combination is used to move the part under the cutting laser. Distances are in inches.

LINE	COMMAND	DESCRIPTION
N2	M80	Laser is OFF.
N3	(REF,X,Y)	Find Hardware Home for the X and Y axes.
N6	G91 G70	Initiate Incremental Positioning and English Programming (inches).
N7	(CCP, X, Y, 0.5)	Set ICRC to X/Y plane with a tool diameter of 0.5 inches.
N8	G41	Turn ON tool radius compensation (ICRC). Compensate to the left of the workpiece.
N9	G1 X4.25 Y0.6 F100.0	Linear Offset Move for the X and Y axes.
N10	Y0.9	Y offset to start of rectangle pattern.
N11	M90	Laser is ON.
N12	(MSG, Cutting Rectangle)	A message will be displayed.
N13	G4 F0.5	A Dwell of 0.5 seconds is established.
N14	Y3.1	Side of rectangle
N15	X3.4	Side of rectangle
N16	Y-3.1	Side of rectangle
N17	X-3.4	Side of rectangle
N18	M80	Laser is OFF.
N19	G40 X-2.75	Deactivate the Cutter Radius Compensation and perform the end move.
N20	G41	Activate the Cutter Radius Compensation left.
N21	X0.5 Y1.5	Offset to the starting point of the circle.
N22	M90	Laser is ON.
N23	(MSG,0, Cutting Circle)	A message will be displayed.
N24	G2 I0.6 J0	A 0.6 inch radius circle will be produced.
N25	M80	Laser is OFF.
N26	G4 F0.5	A Dwell of 0.5 seconds is established.
N27	G40 G90 G0 X0.0 Y0.0	Go home at a rapid traverse rate.
N29	M2	End of The Program

PROGRAMMING EXAMPLE (CON'T):



APPENDIX 2: CORNER ROUNDING/VELOCITY PROFILING

CORNER ROUNDING (G23) AND NON-CORNER ROUNDING (G24) MODES:

An insertion of the code **G23** within a motion command statement (e.g., G8 G1 G23 X100.0 Y200.0) or on a separate line, sets the Unidex 21 for the "corner rounding" mode. The definition of "corner rounding" as applied to the Unidex 21 is as follows:

- Set the trajectory generator of the Unidex 21 to execute all subsequent motion commands (either linear or circular commands, in a single or multi-axis configuration) at the commanded feedrate. Depending upon the setting of Main System Parameter #60 or 61, the Unidex 21 may either utilize the ramptime parameter setting (**RAMP**) or digital filter (**FILT**) for acceleration/deceleration.

- Disable the "motion complete flag" in the servo control loops of each of the enabled axes. This effectively tells the trajectory generator to "not wait" for the given axes to stabilize on the final commanded position of the just completed motion command before sending the next motion command(s) to the servo loops.

An insertion of the code **G24** instead of the code **G23** "inverts" the meaning of the two statements for the **G23** mode. In otherwords with **G24** inserted, the "motion complete flag" is enabled insuring that the commanded position of the currently executing motion command is completely executed by the servo loops before the next command(s) is sent down by the trajectory generator.

Also, it is possible to utilize both the axes **RAMP** or **FILT** commands (or their related parameter settings) in order to optimize the trajectory command execution and provide similar performance to "following error" based motion controllers. Please refer to related commands and parameters for further explanation.

Note that commands **G23** and **G24** are system modal. In other words, once the command **G23** is encountered either in the execution of a program or the system "MDI" (manual) mode it stays in effect until changed by **G24** or a system reset.

PROFILING (G8) AND NON-PROFILING (G9) MODES:

An insertion of the **G8** command enables the velocity profiling mode. Unlike the **G23/G24** commands which are modal, this command must be inserted within each motion command in order for it to be recognized for the given motion command (e.g., G8 G1 X100.0 Y100.0). The definition of "profiling" as applied to the Unidex 21 is as follows:

- Each motion command processed by the trajectory generator is setup to link the moves together such that constant vector velocity is maintained between commanded motion blocks or, if the Feedrate is changed, the system will accelerate/decelerate to the next velocity utilizing either **RAMP** time, **FILT** value, or the combination of both. Note that the two aforementioned commands have user program setting capability or Main System Parameter settings. Refer to related commands and parameters for further explanation.

- If the motion command containing this code is the first motion command within a string of other **G8** motion commands (e.g., if it is the first command after a previously executed **G9** command), the vector velocity of this command will begin at zero speed. Subsequent **G8** commands encountered after this first **G8** command will be executed beginning at the vector velocity reached by the previous **G8** command, if the Feedrate is not changed in this program block.

- If the **G24** mode is enabled, each new vector velocity specified by **G8** commands will be "ramped" to that specified velocity starting at the "attained" velocity of the previous **G8** command (the ramp time being specified by the **RAMP** command, e.g., (RAMP,100)). Note that for this particular mode of operation, it is important that the vector velocity "direction" be maintained between ending of one **G8** command to the beginning of the next **G8** or ending **G9** command in order for the desired position trajectory to be maintained. If vector velocity direction needs to be changed, it MUST be accomplished through the programming of an "arc" or circle segment. This is the only way to ensure tangency.

- If the **G23** mode is enabled and Main System Parameter #61 is set to NO, each new vector velocity specified by **G8** commands will be instantaneously executed at the specified velocity. In this mode, the **RAMP** command is ignored by the trajectory generator. This means that when it is desired to change vector velocity directions between **G8** programmed segments, the programmer need only specify new "linear" command segments (e.g., G8 G1 X100.0 Y100.0) in order to change direction. The acceleration/deceleration can only be provided by the digital filter setting (either via **FILT** command or Main System Parameter #60).

PROFILING (G8) AND NON-PROFILING (G9) MODES (CON'T):

An insertion of the **G9** command disables the velocity profiling mode and essentially puts the trajectory generator in the "point to point" position mode. The motion command executed under this mode is "guaranteed" to feed the servo control loop(s) the final or ending position of the specified command. Note that this command is "inferred" (e.g., considered the default command) in the **G8/G9** mode of operation. The motion command "G1 X100.0 Y100.0" is automatically recognized by the trajectory generator software as being the command "G9 G1 X100.0 Y100.0". The definition of "non-profiling" as applied to the Unidex 21 is as follows:

- Each **G9** motion command processed by the trajectory generator is set up to execute based on the priority of attaining the final commanded position. The velocity profile will be calculated in such a fashion as to insure that the position generated by the trajectory generator is guaranteed to be obtained at a final axis velocity of zero.

- A **G9** command NOT preceded by a **G8** command has a beginning and ending vector velocity of zero.

- A **G9** command preceded by a **G8** command has a beginning vector velocity equal to the ending velocity of the previous **G8**.

In the "following error" mode, an inherent "exponential" ramp function, is added to each of the servo axis, effectively "smoothing" out sudden changes in velocity commands. The effect is illustrated in Figure A-1.

PROFILING (G8) AND NON-PROFILING (G9) MODES (CON'T):

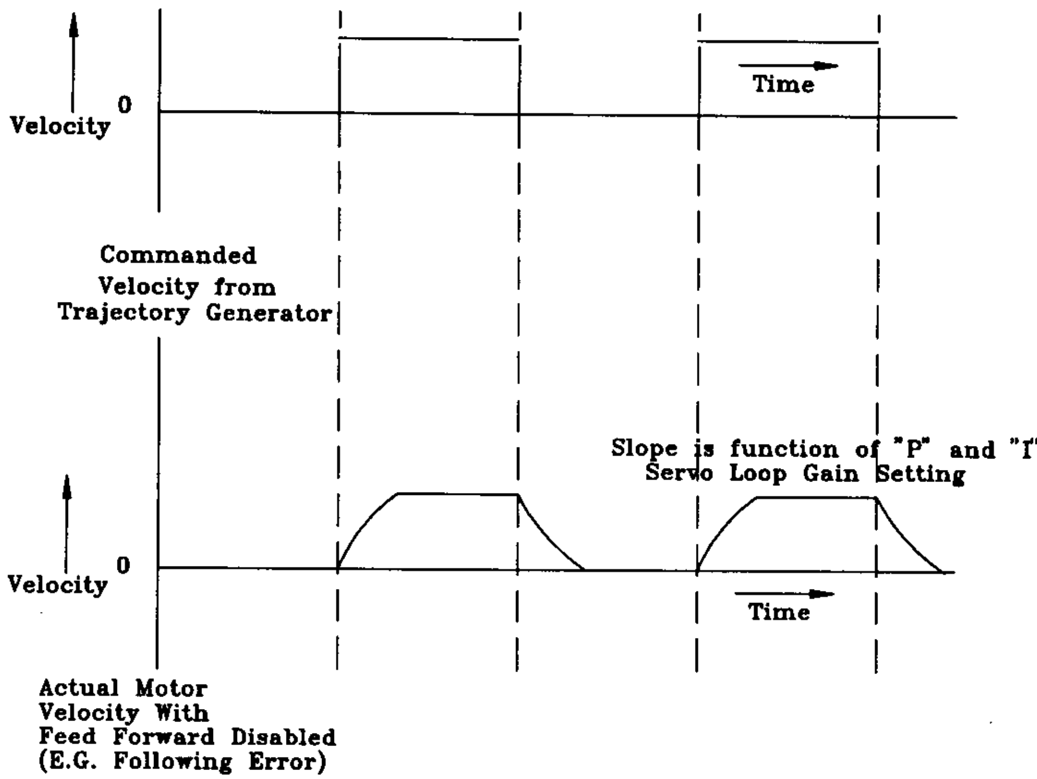


Figure A-1: Effects of "G23" Mode on Trajectory Generators with Following Error Enabled (KF1=0)

This effect may also be obtained using the **FILT** command or Main System Parameter #60. In this mode of operation, however, the exponential ramp is applied to the velocity command. Note that following error does not need to be enabled in this mode.

PROFILING (G8) AND NON-PROFILING (G9) MODES (CON'T):

Combining the **G8** function with **G23**, and enabling "following error" or using the **FILT** command for the servo loops allows the position profiles such as the one depicted in Figure A-2, to be executed.

Referring to Figure A-2, take note that the joining points of the **G8** command segments (which are strings of arc and linear commands) are not tangent to each other. When operating in the **G23** mode, segment tangency is not required. In addition, velocities for each segment can be changed at will (changing of velocities can not be made apparent in this diagram) without significantly changing the intended profile.

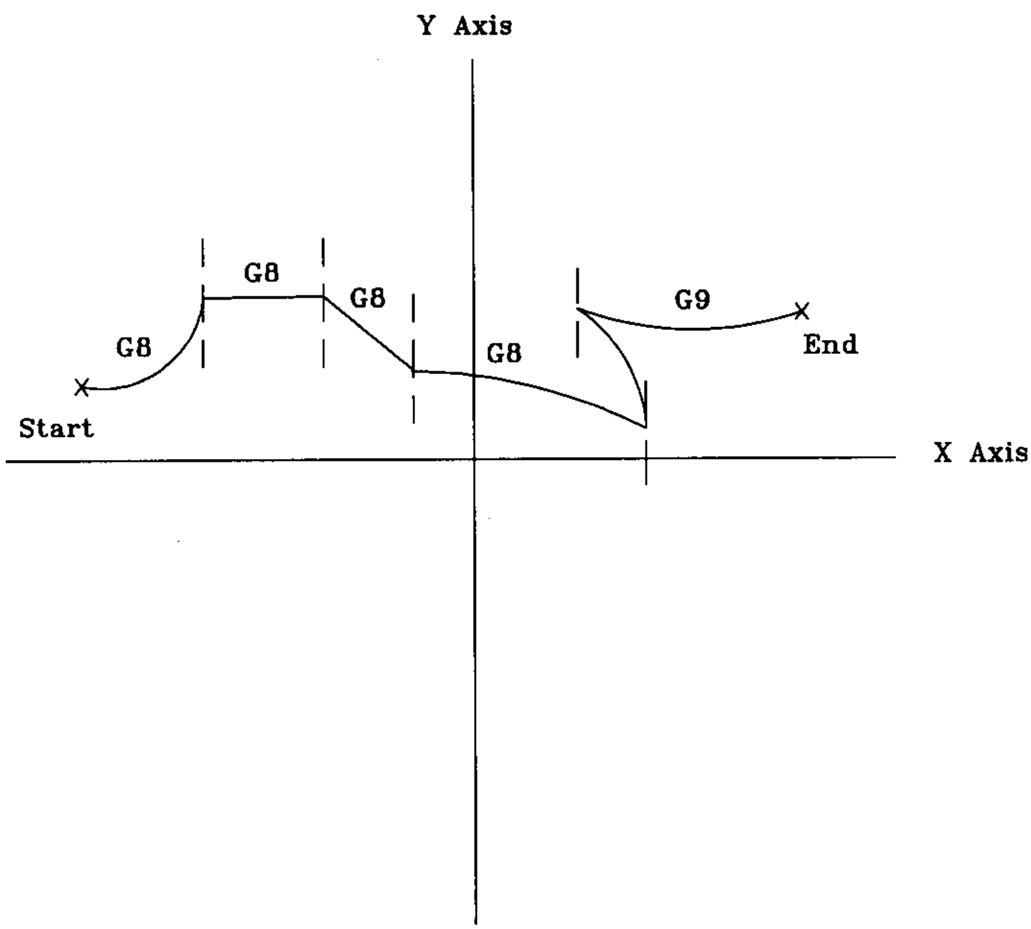


*Figure A-2: Typical Two Axis Position Profile Using **G8**, **G23** with Following Error or Filt Enabled*

PROFILING (G8) AND NON-PROFILING (G9) MODES (CON'T):

Combining the **G8** function with **G24** allows position profiles such as the one depicted in Figure A-3 to be executed. Note that for this particular mode of operation, allowing following error to be enabled or disabled produces negligible effects on the motion profile unless high vector velocities are commanded.

Referring to Figure A-3, notice that the joining points of each segment are tangent. Tangency between all **G8** and **G9** segments is a requirement for **G24** operation. As in the case of the **G23** operation described above, segment velocities can be changed at will. In addition, ramptime can be changed from segment to segment.

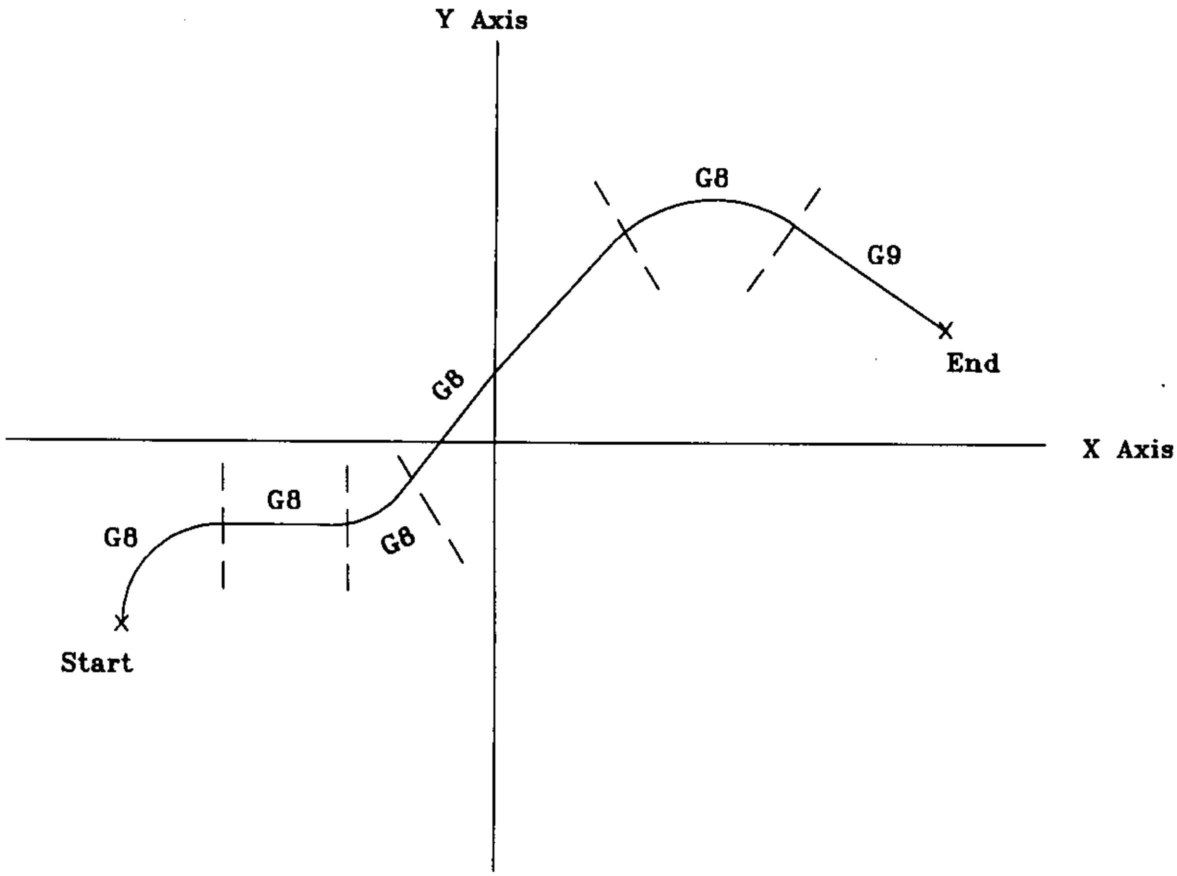


Figure A-3: Typical Two Axis Position Profile Using "G8" and "G24" Modes

APPENDIX 3: PROGRAMMING EXAMPLES

PROGRAMMING EXAMPLE #1:

This example utilizes the following commands:

$OT7	**DENT**	**G1**	**G2**
G3	**G70**	**G90**	**G91**
JUMP	**M0**	**M2**	**MSG**
SCF	**SCO**		

```
(DENT,ENT1)                                      ; Define the Entry Point ENT1.
(SCF,X.4,Y.4)                                    ; Scaling factor, X.4 Y.4.
(SCO,1)                                          ; Scaling is ON.
G91 G70                                          ; Initiate Incremental Positioning and
                                                 ; English Programming (inches).
(DENT,HFCL)                                      ; Define the Entry Point HFCL.
(MSG,---press cycle start to run 4 half circles---)  ; Display a message.
M0                                               ; Program Stop
$OT7=H,0                                         ; Pen down, Output 7 is ON.
G2 X2. Y0. I1. J0.                               ; 1st 1/2 circle CW
G3 X2. Y0. I1. J0.                               ; 2nd 1/2 circle CCW
G3 X-2. Y0. I-1. J0.                             ; 3rd 1/2 circle CCW
G2 X-2. Y0. I-1. J0.                             ; 4th 1/2 circle CW
$OT7=H,1                                         ; Pen up, Output 7 is OFF.
(DENT,QRCL)                                      ; Define the Entry Point QRCL.
G90 G1 X.5 Y3.5 F300.                            ; Absolute Linear Offset move.
(MSG,---press cycle start to run 8 qtr circles---)   ; Display a message.
M0                                               ; Program Stop
(SCO,0)                                          ; Scaling is turned OFF.
$OT7=H,0                                         ; Pen down, Output 7 is ON.
G2 X1. Y1. I1. J0.                               ; 1st 1/4 circle CW
G2 X1. Y-1. I0. J-1.                             ; 2nd 1/4 circle CW
G3 X1. Y-1. I1. J0.                              ; 3rd 1/4 circle CCW
G3 X1. Y1. I0. J1.                               ; 4th 1/4 circle CCW
G3 X-1. Y1. I-1. J0.                             ; 5th 1/4 circle CCW
G3 X-1. Y-1. I0. J-1.                            ; 6th 1/4 circle CCW
```

PROGRAMMING EXAMPLE #1 (CON'T):

```
G2 X-1. Y-1. I-1. J0.          ; 7th 1/4 circle CW
G2 X-1. Y1. I0. J1.            ; 8th 1/4 circle CW
$OT7=H,1                       ; Pen up, Output 7 is OFF.
(JUMP,ENT1)                    ; Jump to Entry Point ENT1.
M2                             ; End of The Program
```

PROGRAMMING EXAMPLE #2:

This example utilizes the following commands:

$OT7	**$TOD**	**DENT**	**G1**
G4	**G24**	**G70**	**G90**
G91	**JUMP**	**M0**	**M2**
MTOR	**MSG**	**RAMP**	**RPT**

```
(MSG,#$TOD)                                                  ; Display the Time and
                                                             ; Date.
(DENT,STRT)                                                  ; Define Entry Point STRT.
G91 G70                                                      ; Initiate Incremental Posi-
                                                             ; tioning and English
                                                             ; Programming (inches).
G24                                                          ; Non-Corner Rounding
                                                             ; mode.
(DENT,SCA1)                                                  ; Define Entry Point SCA1.
G90 G1 X0. Y0. F300.                                         ; ABS Linear Offset move.
G91                                                          ; Initiate Incremental Posi-
                                                             ; tioning.
(MSG,---press cycle start to continue with scan pattern- #1---) ; Display a message.
M0                                                           ; Program Stop
(MSG,)                                                       ; Clear the message.
$OT7=H,0                                                     ; Pen down, Output 7 is ON.
(RAMP,250)                                                   ; Ramp time is .250 sec-
                                                             ; onds.
```

PROGRAMMING EXAMPLE #2 (CON'T):

```
(RPT,10                                                      ; Scan will move Y in/out 5",
                                                             ; inc. of .25" on X.
 G1 Y5. F300.                                                ; Linear Contouring move.
 G1 X.25 F100.                                               ; Linear Contouring move.
 G1 Y-5. F300.                                               ; Linear Contouring move.
 G1 X.25 F100.                                               ; Linear Contouring move.
)                                                            ; Close the Repeat Loop.
Y5. F300.                                                    ; Linear Contouring move.
$OT7=H,1                                                     ; Pen up, Output 7 is OFF.
(DENT,SCA2)                                                  ; Define Entry Point SCA2.
G90 G1 X0. Y0. F300.                                         ; ABS Linear Offset move.
G91                                                          ; Initiate Incremental Posi-
                                                             ; tioning.
(MSG,---press cycle start to con't with scan pattern #2---)  ; Display a message.
M0                                                           ; Program Stop
(MSG,)                                                       ; Clear the message.
$OT7=H,0                                                     ; Pen down, Output 7 is ON.
(RAMP,100)                                                   ; Ramp for 100 milliseconds
(RPT,10                                                      ; Repeat Loop
 G1 X5. F300.                                                ; Linear Contouring move.
 G1 Y.25 F100.                                               ; Linear Contouring move.
 G1 X-5. F300.                                               ; Linear Contouring move.
 G1 Y.25                                                     ; Linear Contouring move.
)                                                            ; Close the Repeat Loop.
G1 X5. F300.                                                 ; Linear Offset move.
$OT7=H,1                                                     ; Pen up, Output 7 is OFF.
(MSG,---press cycle start to return absolute X0/Y0---)       ; Display a message.
M0                                                           ; Program Stop
G90 G70                                                      ; Initiate Absolute Position-
                                                             ; ing and English Program-
                                                             ; ming (inches).
G1 X0. Y0. F300.                                             ; Linear Offset move.
G91                                                          ; Initiate Incremental Posi-
                                                             ; tioning.
(MTOR,0,X,Y)                                                 ; Motors OFF for X, Y axis.
```

PROGRAMMING EXAMPLE #2 (CON'T):

```
(MSG, motors OFF, move X and Y, press cycle start to con't)     ; Display a message.
M0                                                              ; Program Stop
(MSG,---load new parts in fixture-press cycle start to con't---); Display a message.
M0                                                              ; Program Stop
(MSG, clear all personnel-press cycle start to power motors)    ; Display a message.
M0                                                              ; Program Stop
(MSG, ---press cycle start to repeat program---)                ; Display a message.
(MTOR,1,X,Y)                                                    ; Motors ON for X, Y axis.
G4 F1.                                                          ; A Dwell of 1 second is es-
                                                                ; tablished.
M0                                                              ; Program Stop
(MSG,)                                                          ; Clear the message.
(JUMP,STRT)                                                     ; Jump to Entry Point STRT.
M2                                                              ; End of The Program
```

PROGRAMMING EXAMPLE #3:

This example utilizes the following commands:

DENT	**G4**	**INP**	**JUMP**
M0	**M2**	**MSG**	**OTP**

```
N1 (DENT,STRT)                                  ; Define the Entry Point STRT.
N3 (MSG,#H:$INP)                                ; Display input status of 16 inputs in hex.
N4 (MSG,#H:$IN0,#H:$IN1,#H:$IN2,#H:$IN3)        ; Display value of inputs 0,1,2,and 3.
N5 (MSG,               )                        ; Display a message.
N6 $OTP=H,00                                    ; All outputs ON (reverse logic) (powers up
                                                ; with outputs OFF).
N7 $OTP=H,FF                                    ; All outputs are OFF.
N8 $OTP=H,FE                                    ; Output 1 is ON.
N9 $OTP=H,FC                                    ; Outputs 1 and 2 are ON.
N10 $OTP=H,F8                                   ; Outputs 1,2, and 3 are ON.
N11 $OTP=H,F0                                   ; Outputs 1,2,3, and 4 are ON.
N12 $OTP=H,E0                                   ; Outputs 1,2,3,4, and 5 are ON.
N13 $OTP=H,C0                                   ; Outputs 1,2,3,4,5, and 6 are ON.
N14 $OTP=H,80                                   ; Outputs 1,2,3,4,5,6, and 7 are ON.
N15 $OTP=H,00                                   ; Outputs 1,2,3,4,5,6,7, and 8 are ON.
```

PROGRAMMING EXAMPLE #3 (CON'T):

```
N16 $OTP=H,FF                                  ; All outputs are OFF.
N17 G4 F1.                                     ; A Dwell of 1 second is established.
N18 (MSG,waiting on INT=0)                     ; Display a message.
N19 (DENT,LOOP)                                ; Define the Entry Point LOOP.
N20 (JUMP,CNT1,$IN0.EQ.H,0)                    ; Perform the specified jump.
N21 (JUMP,CNT2,$IN1.EQ.H,0)                    ; Perform the specified jump.
N22 (JUMP,CNT3,$IN2.EQ.H,0)                    ; Perform the specified jump.
N23 (JUMP,CNT4,$IN3.EQ.H,0)                    ; Perform the specified jump.
N24 (JUMP,CNT5,$IN4.EQ.H,0)                    ; Perform the specified jump.
N25 (JUMP,CNT6,$IN5.EQ.H,0)                    ; Perform the specified jump.
N26 (JUMP,CNT7,$IN6.EQ.H,0)                    ; Perform the specified jump.
N27 (JUMP,CNT8,$IN7.EQ.H,0)                    ; Perform the specified jump.
N28 (JUMP,LOOP)                                ; Jump to the Entry Point LOOP.
N29 G4 F1.                                     ; A Dwell of 1 second is established.
N30 (MSG, the inputs are negative logic also)  ; Display a message.
N31 G4 F1.                                     ; A Dwell of 1 second is established.
N32 (MSG, all outputs are off)                 ; Display a message.
N33 G4 F1.                                     ; A Dwell of 1 second is established.
N34 (DENT,CNT1)                                ; Define the Entry Point CNT1.
N35 (MSG, DENT CNT1 "H,0" INPUT 1 is on)       ; Display a message.
N36 M0                                         ; Program Stop
N37 (JUMP,STRT)                                ; Jump to Entry Point STRT.
N38 (DENT,CNT2)                                ; Define the Entry Point CNT2.
N39 (MSG, DENT CNT2 "H,0" INPUT 2 is on)       ; Display a message.
N40 M0                                         ; Program Stop
N41 (JUMP,STRT)                                ; Jump to Entry Point STRT.
N42 (DENT,CNT3)                                ; Define the Entry Point CNT3.
N43 (MSG, DENT CNT3 "H,0" INPUT3 is on)        ; Display a message.
N44 M0                                         ; Program Stop.
N45 (JUMP,STRT)                                ; Jump to Entry Point STRT.
N46 (DENT,CNT4)                                ; Define the Entry Point CNT4.
N47 (MSG, DENT CNT4 "H,0" INPUT 4 is on)       ; Display a message.
N48 M0                                         ; Program Stop
N49 (JUMP,STRT)                                ; Jump to Entry Point STRT.
N50 (DENT,CNT5)                                ; Define the Entry Point CNT5.
```

PROGRAMMING EXAMPLE #4:

This example utilizes the following commands:

CLS	**DFS**	**G4**	**INT1**
INT2	**M0**	**M2**	**M30**
M47	**MSG**		

```
(CLS,MSG1)                                                  ; Call Subroutine MSG1.
(INT1,4,IN01)                                               ; Jump to DFS, IN01 upon
                                                            ; INT1 going low.
(INT2,4,IN02)                                               ; Jump to DFS, IN02 upon
                                                            ; INT2 going low.
M47                                                         ; Program Repeat
M2                                                          ; End of The Program
(DFS,MSG1                                                   ; Define Subroutine MSG1.
 (MSG, testing interrupts 1 and 2)                          ; Display a message.
 G4 F.5                                                     ; Dwell 1/2 sec or 500 msec
 (MSG,)                                                     ; Clear the message.
 G4 F.5                                                     ; Dwell 1/2 sec or 500 msec
)                                                           ; End of Subroutine.
(DFS,IN01                                                   ; Define Subroutine IN01.
 (MSG, you have activated INT1, press cycle start to con't.) ; Display a message.
 M0                                                         ; Program Stop
 (MSG,)                                                     ; Clear the message.
)                                                           ; End of Subroutine IN01.
(DFS,IN02                                                   ; Define Subroutine IN02.
 (MSG, you have activated INT2, press cycle start to con't.) ; Display a message.
 M0                                                         ; Program Stop
 (MSG,)                                                     ; Clear the message.
)                                                           ; End of Subroutine IN02.
M30                                                         ; Physical End of Program
```

PROGRAMMING EXAMPLE #5:

This example utilizes the following commands:

CCP	**G1**	**G40**	**G41**
G42	**G70**	**G91**	**REF**

```
G70                 ; Initiate English Programming (inches).
G91                 ; Initiate Incremental Positioning.
(REF,X,Y)           ; Find the Hardware Home for the X and Y axes.
G1 F200.            ; Linear moves with a Feedrate of 200.
(CCP,X0.,Y.2)       ; Cutter Compensation for a tool Dia of .25".
; ********************************************Without Cutter Compensation left
X2. Y2.             ; Linear Offset move for the X and Y axes.
Y2.                 ; Linear move of the Y axis.
X2.                 ; Linear move of the X axis.
Y-2.                ; Linear move of the Y axis.
X-2.                ; Linear move of the X axis.
X-2. Y-2.           ; Move back to Home or the starting point.
; ******************************************Cutter Compensation left
G41 X2. Y2.         ; Initiate Cutter Compensation left with index.
Y2.                 ; Linear move of the Y axis.
X2.                 ; Linear move of the X axis.
Y-2.                ; Linear move of the Y axis.
X-2.                ; Linear move of the X axis.
G40                 ; Deactivate the Cutter Compensation.
X-2. Y-2.           ; Move back to Home or the starting point.
; ********************************************Without Cutter Compensation right
X6. Y2.             ; Linear Offset move of the X and Y axes.
X2.                 ; Linear move of the X axis.
Y2.                 ; Linear move of the Y axis.
X-2.                ; Linear move of the X axis.
Y-2.                ; Linear move of the Y axis.
X-6. Y-2.           ; Move back to Home or the starting point.
; ******************************************Cutter Compensation right
G42 X6. Y2.         ; Initiate Cutter Compensation right with index.
X2.                 ; Linear move of the X axis.
```

PROGRAMMING EXAMPLE #5 (CON'T):

```
Y2.                 ; Linear move of the Y axis.
X-2.                ; Linear move of the X axis.
Y-2.                ; Linear move of the Y axis.
G40                 ; Deactivate Cutter Compensation.
X-6. Y-2.           ; Move back to Home or the starting point.
M47                 ; Restart The Program
```

PROGRAMMING EXAMPLE #6:

This example utilizes the following commands:

G1	**G23**	**G24**	**G70**
G90	**M0**	**REF**	

```
G70                 ; Initiate English Programming (inches).
G91                 ; Initiate Incremental Positioning.
(REF,X,Y)           ; Find Hardware Home for the X and Y axes.
G1 X2. Y2. F200.    ; Linear move with Feedrate of 200.
; *******************************************Corner Rounding Mode
G23                 ; Corner Rounding mode
G1 X2.              ; Linear move of the X axis.
Y2.                 ; Linear move of the Y axis.
X-2.                ; Linear move of the X axis.
Y-2.                ; Linear move of the Y axis.
M0                  ; Program Stop
; *******************************************Non-Corner Rounding mode
G24                 ; Non-Corner Rounding mode
G1 X2.              ; Linear move of the X axis.
Y2.                 ; Linear move of the Y axis.
X-2.                ; Linear move of the X axis.
Y-2.                ; Linear move of the Y axis.
M0                  ; Program Stop
M2                  ; End of The Program
```

PROGRAMMING EXAMPLE #7:

This example utilizes the following commands:

DENT	**DVAR**	**EQ**	**JUMP**
M0	**MSG**		

```
(DVAR,VAR1,VAR2,VAR3,VAR4,VAR5,
 VAR6,FDRT,XDST,YDST)                                   ; Define indicated variables.
(MSG,<VAR1,VAR2,VAR3,VAR4>,
 Please enter your name in "quotes")                    ; Input messages.
(MSG, Welcome, #C:VAR1,VAR2,VAR3,VAR4)                  ; Display a message.
(MSG,<VAR5>, Please enter your
 employee number at this time)                          ; Input messages.
(MSG,#C:VAR1,VAR2,VAR3,VAR4, Please
 enter your employee #, #VAR5, press cycle start to con't) ; Display a message.
M0                                                      ; Program Stop
(DENT,ENT1)                                             ; Define Entry Point ENT1.
(MSG,<VAR6>,Enter 1-Home, 2-HF circle)                  ; Display a message.
(MSG,)                                                  ; Clear the message.
(JUMP,HOME,VAR6.EQ.1)                                   ; Jump if equal to...
(JUMP,HFCL,VAR6.EQ.2)                                   ; Jump if equal to...
(JUMP,ENT1)                                             ; Jump to Entry Point ENT1.
(DENT,CONT)                                             ; Define Entry Point CONT.
(DENT,HOME)                                             ; Define Entry Point HOME.
(REF,X,Y)                                               ; Find Hardware Home for
                                                        ; the X and Y axes.
(MSG,<FDRT>,Enter X and Y axis Feedrate)                ; Input messages.
(MSG,<XDST>, Enter X Offset)                            ; Input messages.
(MSG,<YDST>, Enter Y Offset)                            ; Input messages.
G1 X=XDST Y=YDST F=FDRT                                 ; Linear X/Y Variable move.
(JUMP,ENT1)                                             ; Jump to Entry Point ENT1.
(DENT,HFCL)                                             ; Define Entry Point HFCL.
(MSG, Running HFCL, press cycle start to return)        ; Display a message.
M0                                                      ; Program Stop
(MSG,)                                                  ; Clear the message.
(JUMP,ENT1)                                             ; Jump to Entry Point ENT1.
M2                                                      ; End of The Program
```

DECIMAL	BINARY	HEX	DECIMAL	BINARY	HEX	DECIMAL	BINARY	HEX	DECIMAL	BINARY	HEX	DECIMAL	BINARY	HEX
0	00000000	00	52	00110100	34	104	01101000	68	156	10011100	9C	208	11010000	D0
1	00000001	01	53	00110101	35	105	01101001	69	157	10011101	9D	209	11010001	D1
2	00000010	02	54	00110110	36	106	01101010	6A	158	10011110	9E	210	11010010	D2
3	00000011	03	55	00110111	37	107	01101011	6B	159	10011111	9F	211	11010011	D3
4	00000100	04	56	00111000	38	108	01101100	6C	160	10100000	A0	212	11010100	D4
5	00000101	05	57	00111001	39	109	01101101	6D	161	10100001	A1	213	11010101	D5
6	00000110	06	58	00111010	3A	110	01101110	6E	162	10100010	A2	214	11010110	D6
7	00000111	07	59	00111011	3B	111	01101111	6F	163	10100011	A3	215	11010111	D7
8	00001000	08	60	00111100	3C	112	01110000	70	164	10100100	A4	216	11011000	D8
9	00001001	09	61	00111101	3D	113	01110001	71	165	10100101	A5	217	11011001	D9
10	00001010	0A	62	00111110	3E	114	01110010	72	166	10100110	A6	218	11011010	DA
11	00001011	0B	63	00111111	3F	115	01110011	73	167	10100111	A7	219	11011011	DB
12	00001100	0C	64	01000000	40	116	01110100	74	168	10101000	A8	220	11011100	DC
13	00001101	0D	65	01000001	41	117	01110101	75	169	10101001	A9	221	11011101	DD
14	00001110	0E	66	01000010	42	118	01110110	76	170	10101010	AA	222	11011110	DE
15	00001111	0F	67	01000011	43	119	01110111	77	171	10101011	AB	223	11011111	DF
16	00010000	10	68	01000100	44	120	01111000	78	172	10101100	AC	224	11100000	E0
17	00010001	11	69	01000101	45	121	01111001	79	173	10101101	AD	225	11100001	E1
18	00010010	12	70	01000110	46	122	01111010	7A	174	10101110	AE	226	11100010	E2
19	00010011	13	71	01000111	47	123	01111011	7B	175	10101111	AF	227	11100011	E3
20	00010100	14	72	01001000	48	124	01111100	7C	176	10110000	B0	228	11100100	E4
21	00010101	15	73	01001001	49	125	01111101	7D	177	10110001	B1	229	11100101	E5
22	00010110	16	74	01001010	4A	126	01111110	7E	178	10110010	B2	230	11100110	E6
23	00010111	17	75	01001011	4B	127	01111111	7F	179	10110011	B3	231	11100111	E7
24	00011000	18	76	01001100	4C	128	10000000	80	180	10110100	B4	232	11101000	E8
25	00011001	19	77	01001101	4D	129	10000001	81	181	10110101	B5	233	11101001	E9
26	00011010	1A	78	01001110	4E	130	10000010	82	182	10110110	B6	234	11101010	EA
27	00011011	1B	79	01001111	4F	131	10000011	83	183	10110111	B7	235	11101011	EB
28	00011100	1C	80	01010000	50	132	10000100	84	184	10111000	B8	236	11101100	EC
29	00011101	1D	81	01010001	51	133	10000101	85	185	10111001	B9	237	11101101	ED
30	00011110	1E	82	01010010	52	134	10000110	86	186	10111010	BA	238	11101110	EE
31	00011111	1F	83	01010011	53	135	10000111	87	187	10111011	BB	239	11101111	EF
32	00100000	20	84	01010100	54	136	10001000	88	188	10111100	BC	240	11110000	F0
33	00100001	21	85	01010101	55	137	10001001	89	189	10111101	BD	241	11110001	F1
34	00100010	22	86	01010110	56	138	10001010	8A	190	10111110	BE	242	11110010	F2
35	00100011	23	87	01010111	57	139	10001011	8B	191	10111111	BF	243	11110011	F3
36	00100100	24	88	01011000	58	140	10001100	8C	192	11000000	C0	244	11110100	F4
37	00100101	25	89	01011001	59	141	10001101	8D	193	11000001	C1	245	11110101	F5
38	00100110	26	90	01011010	5A	142	10001110	8E	194	11000010	C2	246	11110110	F6
39	00100111	27	91	01011011	5B	143	10001111	8F	195	11000011	C3	247	11110111	F7
40	00101000	28	92	01011100	5C	144	10010000	90	196	11000100	C4	248	11111000	F8
41	00101001	29	93	01011101	5D	145	10010001	91	197	11000101	C5	249	11111001	F9
42	00101010	2A	94	01011110	5E	146	10010010	92	198	11000110	C6	250	11111010	FA
43	00101011	2B	95	01011111	5F	147	10010011	93	199	11000111	C7	251	11111011	FB
44	00101100	2C	96	01100000	60	148	10010100	94	200	11001000	C8	252	11111100	FC
45	00101101	2D	97	01100001	61	149	10010101	95	201	11001001	C9	253	11111101	FD
46	00101110	2E	98	01100010	62	150	10010110	96	202	11001010	CA	254	11111110	FE
47	00101111	2F	99	01100011	63	151	10010111	97	203	11001011	CB	255	11111111	FF
48	00110000	30	100	01100100	64	152	10011000	98	204	11001100	CC			
49	00110001	31	101	01100101	65	153	10011001	99	205	11001101	CD			
50	00110010	32	102	01100110	66	154	10011010	9A	206	11001110	CE			
51	00110011	33	103	01100111	67	155	10011011	9B	207	11001111	CE			

SERVICE AND REPAIR

Customer repair of the equipment is limited. Control Board(s) may be removed and replaced if necessary, however, component level repair must not be attempted.

On-site service should be performed by an experienced electronic technician, preferably one trained by Aerotech.

SHIPMENT

The procedure for shipping equipment to Aerotech, described below, pertains to warranty as well as non-warranty repairs.

1. Before returning any equipment a "Retern Authorization Number" must be obtained from Aerotech. (Be prepared to give the serial number of the equipment being returned.)

2. The equipment being returned must be encased in a proper cushioning material and enclosed in a cardboard box.

Call for a "Return Authorization Number" if it is necessary to ship any equipment to the factory.

WARNING: **DAMAGE TO THE EQUIPMENT DUE TO IMPROPER PACKAGING MAY BOID WARRANTY!**

AEROTECH, INC. SALES OFFICES

Aerotech Sales and Serviece Offices are listed below. For service and information, contact the office servicing your area.

WORLD HEADQUARTERS **AEROTECH, INC.** 101 Zeta Drive Pittsburgh, PA 15238	Phone (412) 963-7470 FAX (412) 963-7459 TWX (710) 795-3125

AEROTECH, LTD. Aldermaston Berkshire RG7 4QW, England	Phone (07356) 77274 FAX (07356) 5022 TLX 847228 Country Code (44)

AEROTECH GMBH Neumeyerstrasse 90 8500 Nuernberg 10 West Germany	Phone (0911) 521031 FAX (0911) 521235 TLX 622474 Country Code (49)

AEROTECH

Warranty and Field Service Policy

Aerotech, Inc. warrants its products to be free from defects caused by faulty materials or poor workmanship for a minimum period of one year from date of shipment from Aerotech. Aerotech's liability is limited to replacing, repairing or issuing credit, at its option, for any products which are returned by the original purchaser during the warranty period. Aerotech makes no warranty that its products are fit for the use or purpose to which they may be put by the buyer, where or not such use or purpose has been disclosed to Aerotech in specifications or drawings previously or subsequently provided, or whether or not Aerotech's products are specifically designed and/or manufactured for buyer's use or purpose. Aerotech's liability on any claim for loss or damage arising out of the sale, resale or use of any of its products shall in no event exceed the selling price of the unit.

Laser Product Warranty

Aerotech, Inc. warrants its laser products to the original purchaser for a minimum period of one year from date of shipment. This warranty covers defects in workmanship and material and is voided for all laser power supplies, plasma tubes and laser systems subject to electrical or physical abuse, tampering (such as opening the housing or removal of the serial tag) or improper operation as determined by Aerotech. This warranty is also voided for failure to comply with Aerotech's return procedures.

Return Products Procedure

Claims for shipment damage (evident or concealed) must be filed with the carrier by the buyer. Aerotech must be notified within (30) days of shipment of incorrect materials. No product may be returned, whether in warranty or out of warranty, without first obtaining approval from Aerotech. No credit will be given nor repairs made for products returned without such approval. Any returned product(s) must be accompanied by a return authorization number. The return authorization number may be obtained by calling an Aerotech service center. Products must be returned, prepaid, to an Aerotech service center (no C.O.D. or Collect Freight accepted). The status of any product returned later than (30) days after the issuance of a return authorization number will be subject to review.

Returned Product Warranty Determination

After Aerotech's examination, warranty or out-of-warranty status will be determined. If upon Aerotech's examination a warrantied defect exists, then the product(s) will be repaired at no charge and shipped, prepaid, back to the buyer. If the buyer desires an air freight return, the product(s) will be shipped collect. Warranty repairs do no extend the original warranty period.

Returned Product Non-Warranty Determination

After Aerotech's examination, the buyer shall be notified of the repair cost. At such time the buyer must issue a valid purchase order to cover the cost of the repair and freight, or authorize the product(s) to be shipped back as is, at the buyer's expense. Failure to obtain a purchase order number or approval within (30) days of notification will result in the product(s) being returned as is, at the buyer's expense. Repair work is warranted for (90) days from date of shipment. Replacement components are warranted for one year from date of shipment.

Rush Service

At times, the buyer may desire to expedite a repair. Regardless of warranty or out-of-warranty status, the buyer must issue a valid purchase order to cover the added rush service cost. Rush service is subject to Aerotech's approval.

On-Site Warranty Repair

If an Aerotech product cannot be made functional by telephone assistance or by sending and having the customer install replacement parts, and cannot be returned to the Aerotech service center for repair, and if Aerotech determines the problem could be warranty-related, then the following policy applies:

Aerotech will provide an on-site field service representative in a reasonable amount of time, provided that the customer issues a valid purchase order to Aerotech covering all transportation and subsistence costs. For warranty field repairs, the customer will not be charged for the cost of labor and material. If service is rendered at times other than normal work periods, then special service rates apply.

If during the on-site repair it is determined the problem is not warranty related, then the terms and conditions stated in the following "On-Site Non-Warranty Repair" section apply.

On-Site Non-Warranty Repair

If any Aerotech product cannot be made functional by telephone assistance or purchased replacement parts, and cannot be returned to the Aerotech service center for repair, then the following field service policy applies:

Aerotech will provide an on-site field service representative in a reasonable amount of time, provided that the customer issues a valid purchase order to Aerotech covering all transportation and subsistence costs and the prevailing labor cost, including travel time, necessary to complete the repair.

AEROTECH, Inc., 101 Zeta Drive, Pittsburgh, Pennsylvania 15238

Phone (412) 963-7470 • TWX (710) 795-3125 • FAX (412) 963-7459

INDEX

/

NAME:

/ - Division of Floating Point Numbers

FUNCTION:

The / function is used for the division of two floating point numbers.

FORMAT:

X=n/p

RETURNS:

The result is a floating point number.

EXAMPLE:

VAR1=2.0/4	; Value of variable VAR1 will be 0.5.
VAR1=VAR2/VAR3	; Value of variable VAR1 will be the quotient of VAR2 (dividend) divided by VAR3 (divisor) or 0.5.
VAR2=H,4	; Value of variable VAR2 will be 4.
VAR1=2.0/BTF(VAR2)	; Value of variable VAR1 will be 0.5.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

( )

NAME:

( ) - Mathematical Phrases of Floating Point Numbers

FUNCTION:

The ( ) function is used to establish mathematical phrases of floating point numbers that are to be treated as a single term.

FORMAT:

RETURNS:

The result is a floating point number.

EXAMPLE:

VAR1=2*(2+4)	; Value of variable VAR1 will be 12.
VAR1=VAR2*(VAR2+VAR3)	; Value of variable VAR1 will the product of the quantity VAR2 plus VAR3 multiplied by VAR2 or 12.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

NAME:

! - Exponents of Floating Point Numbers

FUNCTION:

The ! function is used to express exponents of floating point numbers.

FORMAT:

X = n!p

RETURNS:

The result is a floating point number.

EXAMPLE:

VAR1=2!4	; Value of variable VAR1 will be 2x2x2x2 or 16.
VAR1=VAR2!VAR3	; Value of variable VAR1, in this case, will be evaluated by VAR2xVAR2xVAR2xVAR2 or 16.

Exponents can be some fixed number as well as the value of some other variable as shown in the example above.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

SIN

NAME:

SIN - Sine Value of a Floating Point Angle

FUNCTION:

The **SIN** function is used to derive the Sine value of a floating point angle.

FORMAT:

SIN(X)

RETURNS:

The result is a floating point number.

EXAMPLE:

```
VAR1=SIN(30.0)     ; Value of variable VAR1 will be the Sine of 30° or 0.5.
VAR1=SIN(VAR2)     ; Value of variable VAR1 will be the Sine of VAR2 or 0.5.
```

NOTES:

1) Variables may also be utilized.

2) All angles are expressed in decimal degrees.

RELATED COMMANDS:

DVAR

COS

NAME:

COS - Cosine Value of a Floating Point Angle

FUNCTION:

The **COS** function is used to derive the Cosine value of a floating point angle.

FORMAT:

COS(X)

RETURNS:

The result is a floating point number.

EXAMPLE:

```
VAR1=COS(60.0)        ; Value of variable VAR1 will be the Cosine of 60° or 0.5.
VAR1=COS(VAR2)        ; Value of variable VAR1 will be the Cosine of VAR2 or 0.5.
```

NOTES:

1) Variables may also be utilized.

2) All angles are expressed in decimal degrees.

RELATED COMMANDS:

DVAR

TAN

NAME:

TAN - Tangent Value of a Floating Point Angle

FUNCTION:

The **TAN** function is used to derive the Tangent value of a floating point angle.

FORMAT:

TAN(X)

RETURNS:

The result is a floating point number.

EXAMPLE:

```
VAR1=TAN(30.0)        ; Value of variable VAR1 will be the Tangent of 30° or 0.5773.
VAR1=TAN(VAR2)        ; Value of variable VAR1 will be the Tangent of VAR2 or 0.5773.
```

NOTES:

1) All angles are expressed in decimal degrees.

2) Variables may also be utilized.

RELATED COMMANDS:

DVAR

ATN

NAME:

ATN - Arctangent value of a Floating Point Number

FUNCTION:

The **ATN** function is used to derive the Arctangent (Inverse function of the Tangent) value of a floating point number.

FORMAT:

ATN(X)

RETURNS:

The result is a floating point number in decimal degrees.

EXAMPLE:

```
VAR1=ATN(1.732)     ; Value of variable VAR1 will be the Arctangent of 1.732 or 60°.
VAR1=ATN(VAR2)      ; Value of variable VAR1 will be the Arctangent of VAR2 or 60°.
```

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

DEG

NAME:

DEG - Radian to Decimal Degree Conversion of Floating Point Numbers

FUNCTION:

The **DEG** function is used to convert radians to decimal degrees (both are floating point numbers).

FORMAT:

DEG(X)

RETURNS:

The result is a floating point number in decimal degrees.

EXAMPLE:

VAR1=**DEG**(0.5236)	; Value of variable VAR1 will be the decimal degree equivalent of 0.5236 radians or 30°.
VAR1=**DEG**(VAR2)	; Value of variable VAR1 will be the decimal degree equivalent of VAR2 or 30°.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

RAD

NAME:

RAD - Decimal Degree to Radian Conversion of Floating Point Numbers

FUNCTION:

The **RAD** function is used to convert decimal degrees to radians (both are floating point numbers).

FORMAT:

RAD(X)

RETURNS:

The result is a floating point number in radians.

EXAMPLE:

VAR1=**RAD**(45.0)	; Value of variable VAR1 will be the radian equivalent of 45° or 0.7854 radians.
VAR1=**RAD**(VAR2)	; Value of variable VAR1 will be the radian equivalent of VAR2 or 0.7854 radians.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

ABS

NAME:

ABS - Absolute Value of a Floating Point Number

FUNCTION:

The **ABS** function is used to express the Absolute value of a floating point number.

FORMAT:

ABS(X)

RETURNS:

The result is a positive floating point number.

EXAMPLE:

VAR1=**ABS**(-30)	; Value of variable VAR1 will be 30.0.
VAR1=**ABS**(VAR2)	; Value of variable VAR1 will be the Absolute value of VAR2 or 30.0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

SQR

NAME:

SQR - Square Root of a Floating Point Number

FUNCTION:

The **SQR** function is used to derive the Square Root value of a floating point number.

FORMAT:

SQR(X)

RETURNS:

The result is a positive floating point number.

EXAMPLE:

VAR1=**SQR**(25)	; Value of variable VAR1 will be the Square Root of 25 or 5.0.
VAR1=**SQR**(VAR2)	; Value of variable VAR1 will be the Square Root of VAR2 or 5.0.

NOTES:

1) Variables may also be utilized.

2) The Unidex 21 recognizes only positive floating point numbers.

RELATED COMMANDS:

DVAR

INT

NAME:

INT - Rounding of Floating Point Numbers

FUNCTION:

The **INT** function is used to round-off the fractional part of any floating point number.

FORMAT:

INT(X)

RETURNS:

The result is a floating point integer.

EXAMPLE:

VAR1=**INT**(123.05)	; Value of variable VAR1 will be 123.0.
VAR1=**INT**(VAR2)	; Value of variable VAR1 will round-off the fractional part of VAR2 resulting in a value of VAR1 equals 123.0.
VAR3=**INT**(123.5)	; Value of variable VAR1 will be 124.0.
VAR3=**INT**(VAR4)	; Value of variable VAR3 will round-off the fractional part of VAR4 resulting in a value of VAR4 equals 124.0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

BTF

NAME:

BTF - Convert a Binary Number to a Floating Point Number

FUNCTION:

The **BTF** function is used to convert a binary number to a floating point number. Floating point numbers and binary numbers cannot be used within the same equation, unless one or the other is converted.

FORMAT:

BTF(X)

RETURNS:

The result is a floating point number.

EXAMPLE:

```
(DVAR,VAR1,VAR2)          ; Define variables VAR1 and VAR2.

VAR1=H,1E                 ; Value of variable VAR1 will be 1E Hex (30.0).
VAR2=BTF(VAR1)            ; Value of variable VAR2 will be 30.0.
```

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

FTB

NAME:

FTB - Convert a Floating Point Number to a Binary Number

FUNCTION:

The **FTB** function is used to convert a floating point number to a binary number. Floating point numbers and binary numbers cannot be used within the same equation, unless one or the other is converted.

FORMAT:

FTB(X)

RETURNS:

The result is a binary number.

EXAMPLE:

```
VAR1=40.0              ; Value of variable VAR1 will be H,28.
VAR2=FTB(VAR1)         ; Value of variable VAR2 will be converted to H,28.
```

NOTES:

1) Variables may also be utilized.

2) The fractional portion of the floating point number is rounded to the nearest integer.

RELATED COMMANDS:

DVAR

.EQ.

NAME:

.EQ. - Condition compares two Floating Point Numbers

FUNCTION:

The **.EQ.** function is used to compare two floating point numbers. If the numbers **are equal,** the result is "True" (1). If the numbers **are not equal,** the result is "False" (0).

FORMAT:

floating point number1**.EQ.**floating point number2

RETURNS:

The result is a floating point number, "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1.EQ.SIN<30>)	; Program flow will go to Entry Point ENT1 if VAR1 **is equal** to SIN <30>.
VAR2=VAR1.EQ.SIN<30>	; VAR2 equals H,1 if VAR1 **is equal** to SIN<30>, otherwise VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.NE.

NAME:

.NE. - Condition compares two Floating Point Numbers

FUNCTION:

The **.NE.** function compares two floating point numbers. If the numbers **are not equal** the result is "True" (1). If the numbers **are equal**, the result is "False" (0).

FORMAT:

floating point number1**.NE.**floating point number2

RETURNS:

The result is a floating point number, "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1**.NE.**SIN<30>)	; Program flow will go to Entry Point ENT1 if VAR1 **is not equal** to SIN <30>.
VAR2=VAR1**.NE.**SIN<30>	; VAR2 equals H,1 if VAR1 **is not equal** to SIN<30>, otherwise VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.GT.

NAME:

.GT. - Condition compares two Floating Point Numbers

FUNCTION:

The **.GT.** function compares two floating point numbers. If the value of the first number **is greater** than the value of the second number, the result is "True" (1). If the value of the first number **is not greater** than the value of the second number, the result will be "False" (0).

FORMAT:

floating point number1**.GT.**floating point number2

RETURNS:

The result is a floating point number, "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1**.GT.**SIN<30>)	; Program flow will go to Entry Point ENT1 if VAR1 **is greater** than SIN <30>.
VAR2=VAR1**.GT.**SIN<30>	; VAR2 equals H,1 if VAR1 **is greater** than SIN<30>, otherwise VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

.GE.

NAME:

.GE. - Condition compares two Floating Point Numbers

FUNCTION:

The **.GT.** function compares two floating point numbers. If the value of the first number is **greater than or equal to** the value of the second number, the result is "True" (1). If the value of the first number **is not greater than or equal to** the value of the second number, the result will be "False" (0).

FORMAT:

floating point number1**.GE.**floating point number2

RETURNS:

The result is a floating point number, "1" for True, "0" for False.

EXAMPLE:

(JUMP,ENT1,VAR1.GE.SIN<30>)	; Program flow will go to Entry Point ENT1 if VAR1 **is greater than or equal to** SIN <30>.
VAR2=VAR1.GE.SIN<30>	; VAR2 equals H,1 if VAR1 **is greater than or equal to** SIN <30>, otherwise VAR2 equals H,0.

NOTES:

Variables may also be utilized.

RELATED COMMANDS:

DVAR

CHAPTER 5: POSITION SYNCHRONIZED OUTPUT OPTION (PSO)

The PSO option is a programmable, DSP based, Position Synchronized Output control card that precisely.coordinates the functioning of a peripheral device with the motion initiated by the Unidex 21 Controller.

The PSO provides both digital and analog outputs. Variable output spacing, multiple pulse firing, pulse width, analog range, ramping functions, safe zone, and power level adjustment are all coordinated by transducer feedback.

SECTION 5-1: HARDWARE CONFIGURATION

5-1-1 JUMPERS AND SWITCHES

The PSO Card contains several jumpers and one eight position DIP Switch that must be appropriately configured prior to operation. (See Figure 5-1 for relative locations of the Switch and Jumpers.)

The PSO Card DIP Switch provides the VME Bus address and should be configured as follows:

SWITCH	POSITION	CONDITION
1	0	OFF
2	1	ON
3	0	OFF
4	1	ON
5	1	ON
6	0	OFF
7	1	ON
8	1	ON

The following is a list of the Jumpers on the PSO Card and an explanation as to their function:

JP1	1-2	High Speed Interrupt using a HCPL2601 Opto Coupler in the M9 location.
	2-3	High Speed Interrupt using a HP6N136 Opto Coupler in the M9 location. (Default)
JP2	1-2	Opto Isolated User Input using a HCPL2601 Opto Coupler in the M10 location.
	2-3	Opto Isolated User Input using a HP6N136 Opto Coupler in the M10 location. (Default)
JP3	1-2	Opto Isolated User Input using a HCPL2601 Opto Coupler in the M11 location.
	2-3	Opto Isolated User Input using a HP6N136 Opto Coupler in the M11 location. (Default)
JP4	1-2	Opto Isolated User Input using a HCPL2601 Opto Coupler in the M12 location.
	2-3	Opto Isolated User Input using a HP6N136 Opto Coupler in the M12 location. (Default)
JP5	1-2	Opto Isolated User Input using a HCPL2601 Opto Coupler in the M13 location.
	2-3	Opto Isolated User Input using a HP6N136 Opto Coupler in the M13 location. (Default)
JP6	1-2	Opto Isolated Output using either a 4N33 or HP6N136 Opto Coupler in the M14 location. (Default)
	2-3	Opto Isolated Output using a HCPL2601 Opto Coupler in the M14 location.

JP7	1-2	Opto Isolated Output using either a 4N33 or HP6N136 Opto Coupler in the M15 location. (Default)
	2-3	Opto Isolated Output using a HCPL2601 Opto Coupler in the M15 location.
JP8	1-2	Opto Isolated Output using either a 4N33 or HP6N136 Opto Coupler in the M16 location. (Default)
	2-3	Opto Isolated Output using a HCPL2601 Opto Coupler in the M16 location.
JP9	1-2	Opto Isolated Output using either a 4N33 or HP6N136 Opto Coupler in the M17 location. (Default)
	2-3	Opto Isolated Output using a HCPL2601 Opto Coupler in the M17 location.
JP10		Factory use only.
JP11		Factory use only.
JP12		Factory use only.
JP16	1-2	Internal or external Battery Back-Up.
	2-3	No Battery Back-Up. (Default)
JP17	1-2	Internal or external Battery Back-Up.
	2-3	No Battery Back-Up. (Default)
JP18	1-2	External Battery Back-Up.
	2-3	No Battery Back-Up. (Default)

JP19	1-2	External Battery Back-Up.
	2-3	No Battery Back-Up. (Default)
JP20	1-2	Internal or external Battery Back-Up.
	2-3	No Battery Back-Up. (Default)
JP21	1-2	Internal or external Battery Back-Up.
	2-3	No Battery Back-Up. (Default)
JP22		Factory Set
JP23		Factory Set
JP24		Factory Set
JP25		Factory Set
JP26		Factory Set
JP27	1-2	Active Low Polarity for either Opto or TTL Main Output. (Default)
	2-3	Active High Polarity for either Opto or TTL Main Output.
JP28	1-2	Active Low Polarity for either Opto or TTL AUX2 Output. (Default)
	2-3	Active High Polarity for either Opto or TTL AUX2 Output.
JP29	1-2	Active Low Polarity for either Opto or TTL AUX3 Output. (Default)
	2-3	Active High Polarity for either Opto or TTL AUX3 Output.

JP30	1-2	Active Low Polarity for either Opto or TTL AUX4 Output. (Default)
	2-3	Active High Polarity for either Opto or TTL AUX4 Output.
JP31	1-2	Provides TTL Output for the AUX4 signal.
	2-3	Provides Opto Output for the AUX4 signal. (Default)
JP32	1-2	Provides TTL Output for the AUX3 signal.
	2-3	Provides Opto Output for the AUX3 signal. (Default)
JP33	1-2	Provides TTL Output for the AUX2 signal.
	2-3	Provides Opto Output for the AUX2 signal. (Default)
JP34	1-2	Provides TTL Output for the MAIN signal.
	2-3	Provides Opto Output for the MAIN signal. (Default)

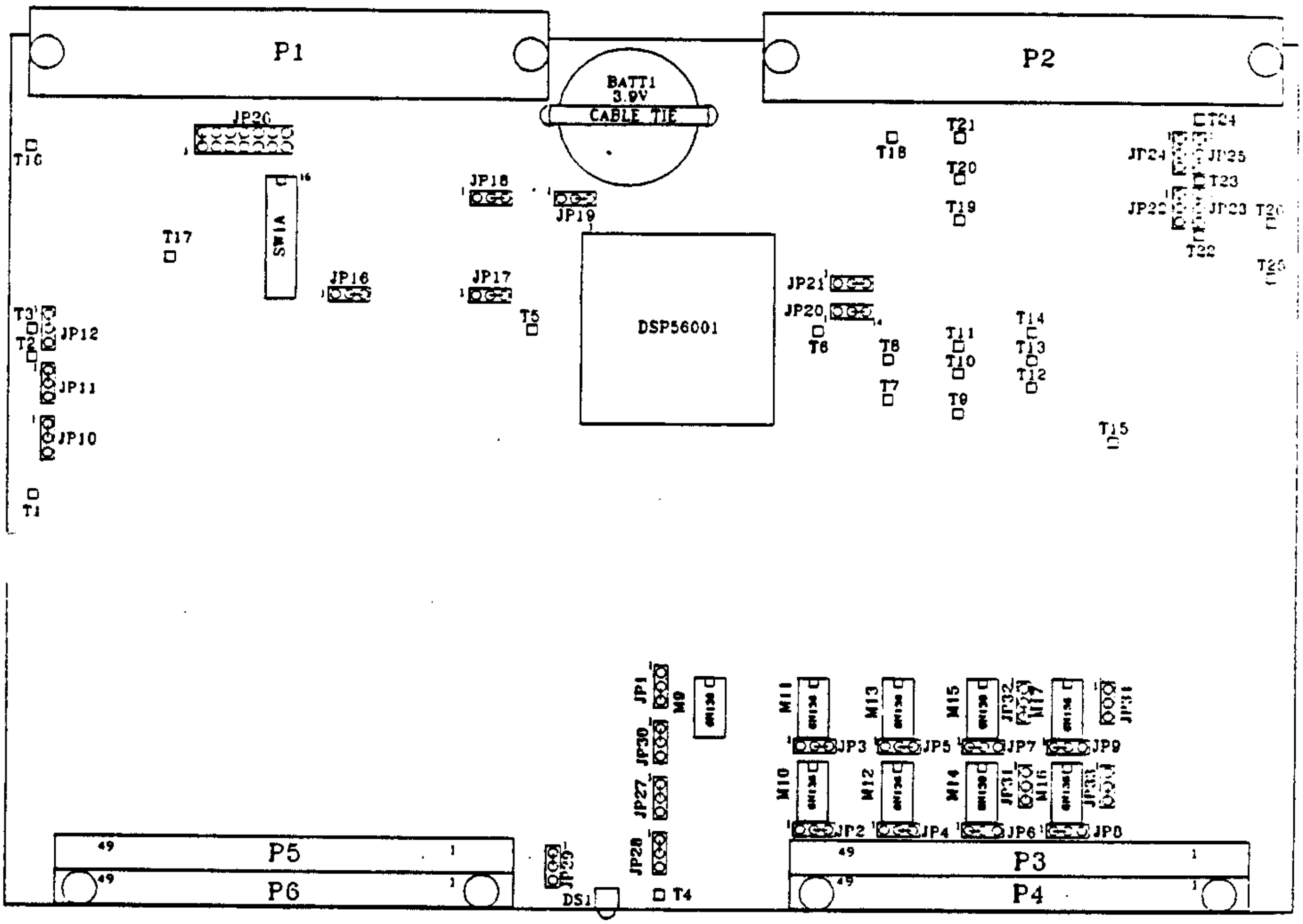


Figure 5-1: PSO Card - Jumpers and Switches

5-1-2 PSO INTERFACE

The PSO Card is interfaced at Connectors P51 and P52 on the Rear Panel of the Unidex 21 Controller. Details of the connectors are shown in Figures 5-2 and 5-3. Electrical characteristics of the connectors are illustrated in Figure 5-4.

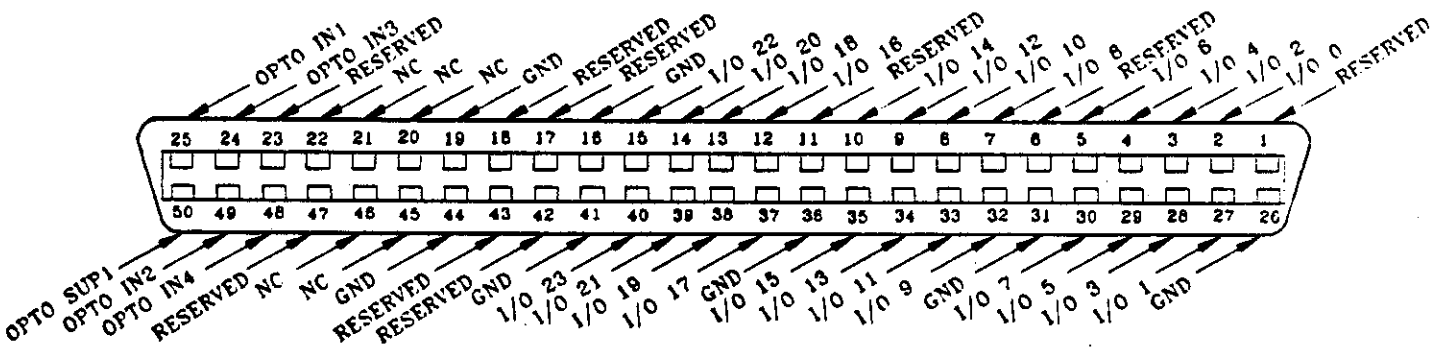


Figure 5-2: PSO Connector J51

PIN	DESCRIPTION
18	Reserved
19	Common
20,21,22	Not Connected
23	Positive Opto Input
24	Opto Input 3
25	Opto Input 1
26	Common
27	I/O Bit 1 (TTL)
28	I/O Bit 3 (TTL)
29	I/O Bit 5 (TTL)
30	I/O Bit 7 (TTL)
31	Common
32	I/O Bit 9 (TTL)
33	I/O Bit 11 (TTL)
34	I/O Bit 13 (TTL)
35	I/O Bit 15 (TTL)
36	Common
37	I/O Bit 17 (TTL)

5-1-2-1 J51 - CONNECTOR

PIN	DESCRIPTION
1	Reserved
2	I/O Bit 0 (TTL)
3	I/O Bit 2 (TTL)
4	I/O Bit 4 (TTL)
5	I/O Bit 6 (TTL)
6	Reserved
7	I/O Bit 8 (TTL)
8	I/O Bit 11 (TTL)
9	I/O Bit 12 (TTL)
10	I/O Bit 14 (TTL)
11	Reserved
12	I/O Bit 16 (TTL)
13	I/O Bit 18 (TTL)
14	I/O Bit 20 (TTL)
15	I/O Bit 22 (TTL)
16	Common
17	Reserved

PIN	DESCRIPTION
38	I/O Bit 19 (TTL)
39	I/O Bit 21 (TTL)
40	I/O Bit 23 (TTL)
41	Common
42,43	Reserved
44	Common
45,46	Not Connected
47	Reserved
48	Opto Input 4
49	Opto Input 2
50	User supplied Input Opto Voltage

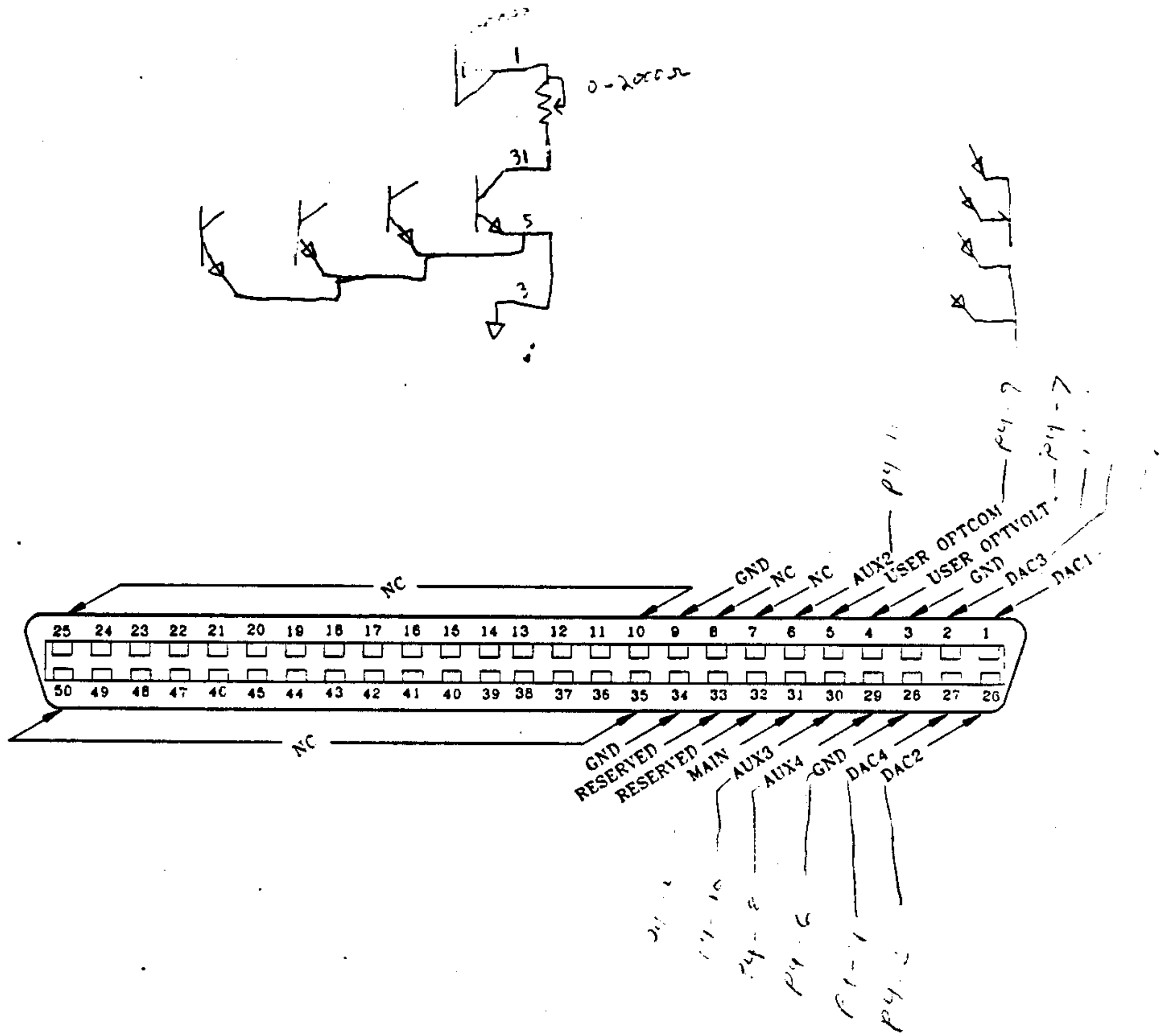


Figure 5-3: PSO Connector J52

PIN	DESCRIPTION
1	Digital to Analog Converter Output - 1 (+/- 10V @ 100 mA)
2	Digital to Analog Converter Output - 3 (+/- 10V @ 100 mA)
3	Common
4	User supplied Output Opto Voltage, +5V (Required with 6N or HCP only)
5	User supplied Output Opto Voltage Common
6	AUX2 Output
7,8	Not Connected
9	Common
10 thru 25	Not Connected
26	Digital to Analog Converter Output - 2 (+/- 10V @ 100 mA)
27	Digital to Analog Converter Output - 4 (+/- 10V @ 100 mA)
28	Common
29	AUX4 Output
30	AUX3 Output

PIN	DESCRIPTION
31	MAIN Output
32	Reserved
33	Reserved
34	Common
35 thru 50	Not Connected

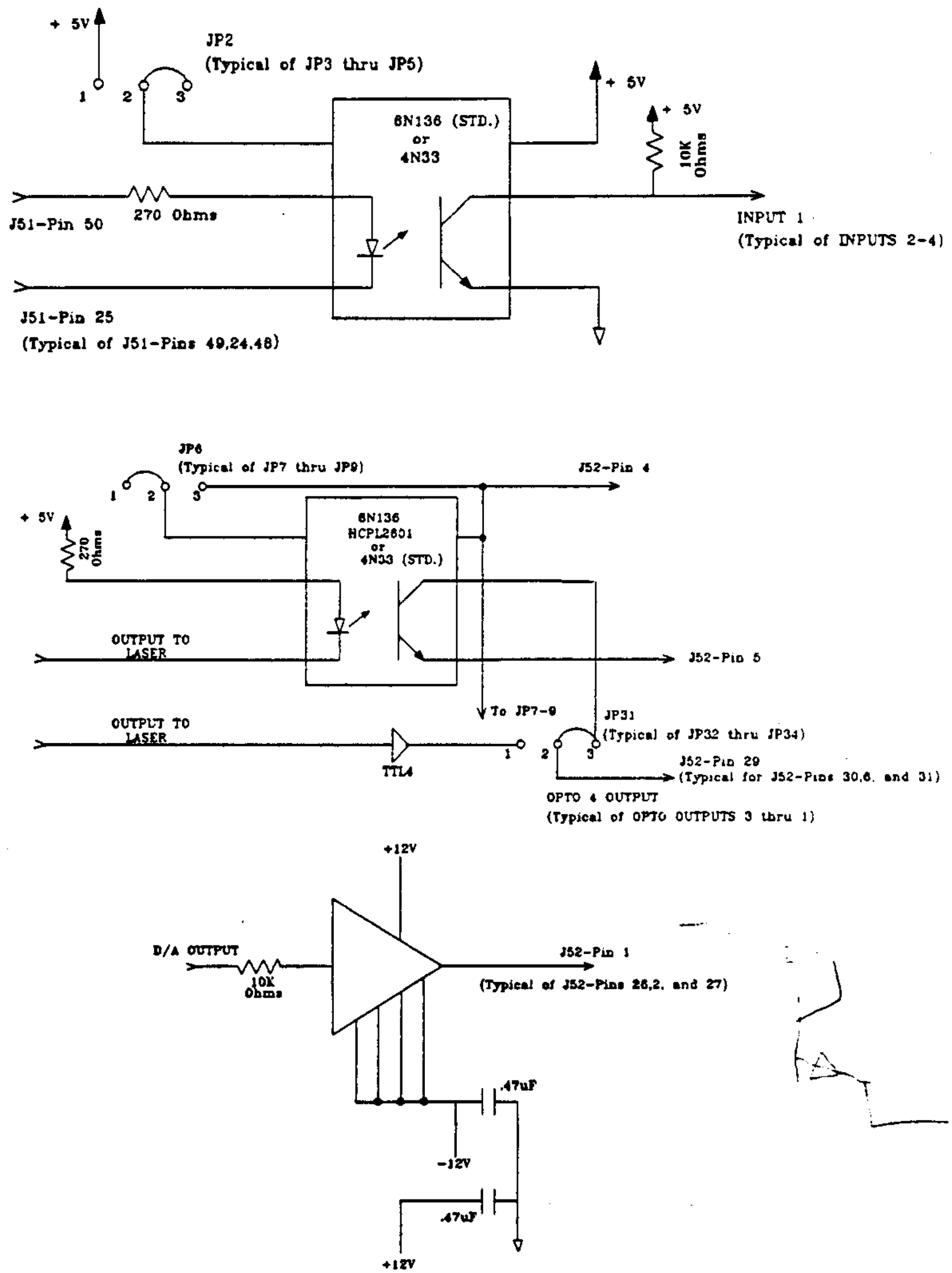


Figure 5-4: Electrical Characteristics of the PSO Interface Connectors (J51 and J52)

SECTION 5-2: USING THE PSO BOARD

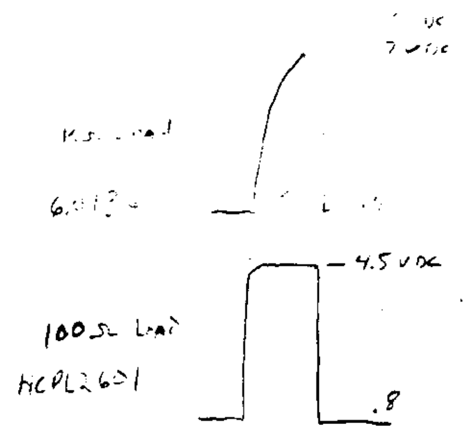
The Position Synchronized Output Board provides a variety of Outputs that may be used to synchronize control with motion, it is most commonly used for the control Laser firing. With the use of a variety of commands the PSO Board may be instructed to activate up to four firing outputs with analog level controls and various types of single-shot or pulse train outputs.

The Position Synchronized Output Board is activated through either a parts program or from the MDI Mode. The following sections provide the commands related to PSO function:

CP7 Sheet 8
6N136 1-2
HCPL2601 2-3

M14 AUX 4
M15 CP7 AUX 3. Sheet 1
M16 AUX 2
M17 NO OUTPUT

6N136 8 mA 15 VDC 200 Ω
5 VDC



Amphenol #57-305 $28.00

PSOC

EXAMPLES (CON'T:

For the same example, the Output Bits will be:

Bits 4, 5, 10 and 11 are driven high.
Bits 6 through 9 are driven low.
Bits 0 through 3 and 12 through 15 are not changed.

NOTE: **This example assumes that a (PSOC,4,1,2) command has been previously issued.**

(PSOC,4,i,o) ; Configuration of the 24 Bit I/O Bus to determine the number of Input and Output lines assigned. (Inputs are justified to the Least Significant Bit.) Following are the available configurations of the I/O Bus:

If

i = 0, and o = 3 Bits 0-23 are configured as Outputs
i = 1, and o = 2 Bits 0-7 are configured as Inputs, Bits 8-23 are configured as Outputs
i = 2, and o = 1 Bits 0-15 are configured as Inputs, Bits 16-23 are configured as Outputs
i = 3, and o = 0 Bits 0-23 are configured as Inputs

For instance (PSOC,3,xx1x0101,xxxx11000011xxxx) would be (PSOC,4,1,2).

NOTE: **The sum of the value of "i" and "o" must be equal to 3.**

NOTES:
The **PSOC** commands are modal.

RELATED COMMANDS:
(PSOF,3) (PSOF,4) (PSOF,5)

PSOC

NAME:
Position Synchronized Conditional Output

FUNCTION:
Tracking is enabled, based on the Input Bit or Word.

FORMAT:
(**PSOC**,condition)

EXAMPLE:

(PSOC,0) ; Input signal conditions are ignored (Default).

(PSOC,1,i,n) ; Position Tracking is enabled only when an Input "i" (0 through 23) is "High" (n does not equal zero) or "Low" (n equals zero). (Refer to the (**PSOC,4**) command.) When the Position Counter is disabled, Counter data is retained.

(PSOC,2,i,n) ; Position Tracking is enabled only when an Input "i" (0 through 23) is "High" (n does not equal zero) or "Low" (n equals zero). (Refer to the (**PSOC,4**) command.) When the Position Counter is disabled, Counter data is reset to zero.

PSOC,3,xx1x0101,xxxx11000011xxxx) ; Position Tracking enabled only when Input Bits 0-7 are configured as shown (0 = Low, 1 = High, x = Not Checked).

For the above example Tracking is enabled when:
Bits 0, 2, and 5 are High.
Bits 1 and 3 are Low.
Bits 4, 6, and 7 are not checked.

If the Input Bits are not configured as indicated above for this example, Tracking is disabled and the Output Bits will be configured as established.

PSOD

NAME:
Position Synchronized Output Distance

FUNCTION:
Establishes the number of machine steps traveled before synchronized output occurs. This command is used in conjunction with the (**PSOF**,3.,.) command only.

FORMAT:
(**PSOD**,case,distance)

EXAMPLE:

(**PSOD**,0,n)	; The Pulse Output will occur at a fixed incremental distance "n".
(**PSOD**,1,ary <n>,+m)	; The Pulse Output will occur at incremental distances as defined in "ary<n>" (array name) starting at array set number "n" and continuing "+/- m" number of array sets.
(**PSOD**,2,ary<n> ,+m)	; The Pulse Output will occur at absolute distances as defined in "ar1" (array name) starting at array set number "n" and continuing "+/- m" number of array sets.

RELATED COMMANDS:
(**PSOF**,3)

PSOF

NAME:
Position Synchronized Output Firing

FUNCTION:
Activates or Deactivates the Pulse Train Output and Tracking.

FORMAT:
(PSOF,case,condition)

EXAMPLE:

(PSOF,0)	; Output Firing Pulse Train and Tracking Disabled. (Default) May also be used to abort a previously activated Pulse Train.
(PSOF,1)	; Activates the Output Firing Pulse Train as established by the **(PSOP)** Command. The Pulse Train will continue until disabled by the **(PSOF,0)** command. No position Tracking.
(PSOF,2,n)	; Activates the Output Firing Pulse Train (established by the **(PSOP)** command) "n" number of times. If n = zero the Output Firing Pulse Train will not be activated until the Previous Output Firng Pulse Train is complete. No position Tracking.
(PSOF,3,X,Y)	; The Output Firing Pulse Train (established by the **(PSOP)** command) is activated. Position Counters will "lock on" the motion of the X and Y Axes (See NOTES). Output firing will occur at distances as established by the **(PSOD)** command. A maximum of three (3) axes may be "locked on" simultaneously.
(PSOF,4,n,X,Y,Z)	; Activates the Output Firing Pulse Train and locks the Position Counters onto Axes X, Y, and Z (See NOTES). A maximum of three (3) axes may be "locked on" simultaneously. The Firing Pattern is determined by "Bit Mapping" as established by the **(PSOM)** command. If Bit = 1, the Output will go/remain high. If Bit = 0, the Output will go/remain low.

PSOF

EXAMPLES (CON'T)

The Pulse Output will occur at a fixed incremental distance "n". If "n" is positive, the Bit Pattern will be run in a forward direction. If "n" is negative, the Bit Pattern will be run in reverse.

(PSOF,5,n,X) ; Activates the Output Firing Pulse Train and locks the Position Counter on the X Axis (See NOTES). A maximum of three (3) axes may be "locked on" simultaneously. The Firing Pattern is determined by "Bit Mapping", established by the (PSOM) command.

If Bit = 1, the Output will be 1 Pulse Train.
If Bit = 0, there will be no Output.

The Pulse Output will occur at a fixed incremental distance "n". If "n" is positive, the Bit Pattern will be run in a forward direction. If "n" is negative, the Bit Pattern will be run in reverse.

NOTES:

The Position Counters translate the value of the Firing Distance "n", as follows:

For a single axis: $n = \text{Counter } 1$

For two axes: $$n = \sqrt{\text{Counter1}^2 + \text{Counter2}^2}$$

For three axes: $$n = \sqrt{\text{Counter1}^2 + \text{Counter2}^2 + \text{Counter3}^2}$$

RELATED COMMANDS:

(PSOC), (PSOD), (PSOM), (PSOP), (PSOR), (PSOT), (PSOT)

PSOM

NAME:

Position Synchronized Pulse Output using Bit Mapping.

FUNCTION:

Establishes a condition such that the Pulse Train Output will occur in accordance with a bit pattern located in a previously established array .

FORMAT:

(PSOM,case,array name <starting byte number>, +/- number of bits to process)

EXAMPLE:

(PSOM,0,ary<n>,m) ; The Pulse Output will occur in accordance with Bit Mapping defined in "ary<n>" (array name) starting at array set number "n" and continuing "+/- m" number of bytes.

(PSOM,1,**$POT**<n>,m) ; The Pulse Output will occur in accordance with Bit Mapping sent from the RS-232 buffer. Data will be retrieved from the "n"th set and continue "m" number of sets.

RELATED COMMANDS:

$POT, (PSOC), (PSOD), (PSOF), (PSOF), (PSOP), (PSOR), (PSOS), (PSOT)

PSOP

NAME:

Position Synchronized Output Pulse Train

FUNCTION:

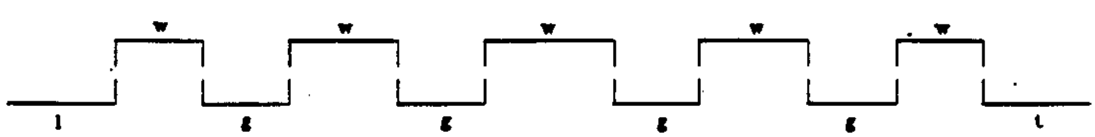
Configuration of the Pulse Output Train.

FORMAT:

(PSOP,case,condition)

EXAMPLES:

(PSOP,0,w)	; Establishes the width, "w" of a single pulse output in milliseconds (See illustration below)
(PSOP,1,l,w,t)	; Establishes a Pulse Train with the following characteristics: l = pulse lead in milliseconds w = pulse width in milliseconds t = pulse trail in milliseconds (See illustration below)
(PSOP,2,l,w,t,r,g)	; Establishes a Pulse Train with the following characteristics: l = pulse lead in milliseconds w = pulse width in milliseconds t = pulse trail in milliseconds r = ramp up and down time in milliseconds g = interval between ramps in milliseconds If the interval is set at "0" the interval will be the same as the Pulse Train width (w). (See illustration below)



w increases with r
g does not equal zero

PSOP

EXAMPLES (CON'T):

(PSOP,3,ary < n >,m) Establishes a Pulse Train with the following characteristics:
off ary < n > milliseconds, + on ary < n + 1 > milliseconds
+ off ary < n + 2 > milliseconds, + on ary < n + 3 > milliseconds
+ .... milliseconds, + off ary < n + -m > milliseconds

(PSOP,4,w) ; Establishes the width, "w" of a single pulse output in micro seconds

(PSOP,5,) ; Toggles between:
An odd event (1,3,5,......) enables a pulse output
An even event (2,4,6,.....) disables a pulse output

RELATED COMMANDS:
(PSOC), (PSOD), (PSOF), (PSOM), (PSOR), (PSOS), (PSOT)

PSOR

NAME:

Position Synchronized Output with Real-Time Control

FUNCTION:

Provides various configurations of the PSO Board's Position Counter.

EXAMPLES:

(PSOR,0) ; Clears all previous real-time control data from counter

(PSOR,1) ; Stops Position Counter from recording new data, retains current data under Operator command.

(PSOR,2,i,n) ; Stops Position Counter from recording new data, retains current data. Activated by User selected Interrupt.

If i = 1 INT1
i = 2 INT2
If n is not zero, this function is enabled
If n = 0, this function is disabled

NOTE: The (PSOR,2,i,n) command is used in conjunction with INT1/2, option 3 only.

(PSOR,3) ; Stops Position Counter from recording new data, returns Counter to zero

(PSOR,4,i,n) ; Stops Position Counter from recording new data, returns Counter to zero. Activated by User selected Interrupt.

If i = 1 INT1
i = 2 INT2
If n is not zero, this function is enabled
If n = 0, this function is disabled

NOTE: (PSOR,4,i,n) is used in conjunction with INT1/2, option 3 only.

RELATED COMMANDS:

INT1/INT2,3,xxxx

PSOT

NAME:

PSOT - Position Synchronized Output, Digital or Analog

FUNCTION:

Provides the User the ability to configure the four D/A outputs or Binary output bits.

FORMAT:

(PSOT,case, condition)

EXAMPLES:

(PSOT,0,b,n,b,n,........) ; Sets output bit "b" either high or low "n".
If "n is not 0, the output will be high.
If "n" = 0, the output will be low.

(PSOT,1,n) ; Sets number of outputs in accordance with Hex Data "n"

(PSOT,2,d,n,d,n,........) ; Establishes output configuration for a Bipolar (11bit + 1sign) DAC. (Default)
d = DAC output line (0-3)
n = -10V to +10 V output voltage
The output voltage has a minimum step size of 4.88mV.

(PSOT,3,d,n,d,n,........) ; Establishes output configuration for a Unipolar 12 Bit DAC.
d = DAC output line (0-3)
n = 0 to +10V output voltage
The output voltage has a minimum, step size of 2.44mV.

RELATED COMMANDS:

(PSOC)

5-3: BIT MAPPING - APPLICATION AND EXAMPLE

Bit Mapping is the process in which Bit Patterns are input to the Unidex 21 Controller, and then used to establish a Pulse Train Output pattern. The **(PSOM)** command is used in conjunction with Bit Mapping. Prior to using a **(PSOM)** command the following set up procedures are necessary:

> NOTE: **The procedures provided below need not occur in the order presented.**

1) Use the **(MALC**,3,n) command to allocate memory for RS-232 data collection.

2) Use the **(PORT**,A) or **(PORT**,B) command to designate an RS-232 Port to be used for data collection if a Remote Controller is being used.

3) Define the Setup and Feedback Arrays. The Array command format must be as follows:
   (DARY,SET < dimension > ,FBK < dimension >)

SETUP ARRAY

The Setup array provides the parameters necessary for the Unidex 21 to receive Bit Mapping data. The Setup Array must be in the following format:

> NOTE: **All data is in Hex format unless otherwise specified**

<0> = H,1
<1> = Initiates a 1 byte start code, a 0 initiates no preference
<2> = Number of bytes of Line Header, STX + Type
<3> = 1 byte Acknowledge code
<4> = 1 byte Line Header code
<5> = 1 byte Line Description code
<6> = 1 byte End Code (must be the same number of bytes as the Header code)
<7> = X Axis conversion factor, machine step/pixel
<8> = Y Axis conversion factor, machine step/pixel
<9> = Left pixel limit of horizontal scan
<10> = Right pixel limit of horizontal scan
<11> = Bottom pixel limit of vertical scan
<12> = Upper pixel limit of vertical scan
<13> = Feedrate (Floating Point Format)

<14> = X Axis Machine Step/Programmed Unit (Floating Point Format)
<15> = Y Axis Machine Step/Programmed Unit (Floating Point Format)

<20> = 1 Bit "not acknowledged" code + 1 Bit Time Out Error
<21> = 1 Bit "not acknowledged" code + 1 Bit Block Size Error
<22> = 1 Bit "not acknowledged" code + 1 Bit Invalid Block Type
<23> = 1 Bit "not acknowledged" code + 1 Bit XOR Check Fail
<24> = 1 Bit "not acknowledged" code + 1 Bit SUM Check Fail
<25> = 1 Bit "not acknowledged" code + 1 Bit Not Enough Memory
<26> = 1 Bit "not acknowledged" code + 1 Bit Line Contains Too Many Characters
<27> to <29> Reserved for additional General Errors

<30> = 1 Bit "not acknowledged" code + 1 Bit Header Block Unspecified Error
<31> = 1 Bit "not acknowledged" code + 1 Bit Header Block Position Outside of Boundary
<32> = 1 Bit "not acknowledged" code + 1 Bit Header Block Unspecified Error
<33> to <39> Reserved for additional Header Block Errors

<40> = 1 Bit "not acknowledged" code + 1 Bit Line Description Block Unspecified Error
<41> = 1 Bit "not acknowledged" code + 1 Bit Line Description Block's Header Block Not Received
<42> = 1 Bit "not acknowledged" code + 1 Bit Line Description Block Length does not Match Number Specified in Header Block
<43> to <49> Reserved for Additional Line Description Errors

<50> to <59> Reserved for Unidex 21

FEEDBACK ARRAY

The Feedback Array provides Feedback of the Parameters requested by the Unidex 21. Two sets of Feedback Arrays are created to provide continuous information processing. The Feedback Array will be returned in the following format:

NOTE: **All data in the two sets of Feedback Arrays must be in Hex format unless otherwise specified**

Data Set 1

<0> = "xxx0" No data available at this time
"xxx1" END
"xxx2" Data is ready , waiting for output of last byte, Indexing Mode
"xxx3" Data is ready, waiting for output of last byte, Auto Positioning Mode
<1> = 1 byte feedback to remote controller
<2> = Horizontal axis position (floating point format)
<3> = Vertical axis scan distance (floating point format)
<4> = Horizontal axis scan distance (floating point format)
<5> = Number of bytes of Bit Mapping data (floating point format)
<6> = Scan firing distance and direction (floating point format)
<7> = Horizontal axis position freerun, one Output Pulse (floating point format)
<8> = Horizontal axis position freerun, one Output Pulse (floating point format)
<9> = Vertical axis position freerun, one Output Pulse (floating point format)
<10> = Vertical axis feedrate freerun, one Output Pulse (floating point format)
<11> = Horizontal axis scan freerun, one Output Pulse (floating point format)
<12> = Horiz. axis scan feedrate freerun, one Output Pulse (floating point format)
<13> to <19> Spare
<20> = Bit mapping data starts here
<n> = Bit mapping data ends here

Data Set 2

<n+1> = "xxx0" No data available at this time
"xxx1" END
"xxx2" Data is ready , waiting for output of last byte, Indexing Mode
"xxx3" Data is ready, waiting for output of last byte, Auto Positioning Mode
<n+2> = 1 byte feedback to remote controller
<m+3> = Horizontal axis position (floating point format)
<n+4> = Vertical axis scan distance (floating point format)
<n+5> = Horizontal axis scan distance (floating point format)
<n+6> = Number of bytes of Bit Mapping data (floating point format)
<n+7> = Scan firing distance and direction (floating point format)
<n+8> = Horizontal axis position freerun, one Output Pulse (floating point format)
<n+9> = Horizontal axis position freerun, one Output Pulse (floating point format)
<n+10> = Vertical axis position freerun, one Output Pulse (floating point format)
<n+11> = Vertical axis feedrate freerun, one Output Pulse (floating point format)
<n+12> = Horizontal axis scan freerun, one Output Pulse (floating point format)
<n+13> = Horizontal axis scan feedrate, free one shot (floating point format)
<n+14> to <n+20> Spare
<n+21> = Bit mapping data starts here
<2n-1> = Bit mapping data ends here

SAMPLE PROGRAM

```
(MALC,<3,512>)                  ;Allocate 512 Bytes for RS232
(DARY,SET<60>,FBK<100>)         ;Setup Case 2 & Feedback Array
SET<0>=H,1                      ;Set to Case 1, Bit Map
SET<1>=H,2                      ;STX Code
SET<2>=H,8                      ;Line Header STX+TYPE+2X+2Y+2SIZE
SET<3>=H,06                     ;Acknowledge
SET<4>=H,10                     ;Line Header Type
SET<5>=H,20                     ;Line Description Type
SET<6>=H,30                     ;End Operation
SET<7>=H,22                     ;1 Pixel = X Axis Machine Steps
SET<8>=H,22                     ;1 Pixel = Y Axis Machine Steps
SET<9>=H,0                      ;X Left Pixel Limit
SET<10>=H,6280                  ;X Right Pixel Limit
SET<11>=H,0                     ;Y Bottom Pixel Limit
SET<12>=H,6280                  ;Y Upper Pixel Limit
SET<13>=300.                    ;Feedrate
SET<14>=10000.                  ;X Machine Step/Program Unit Factor
SET<15>=10000                   ;Y Machine Step/Program Unit Factor
SET<20>=H,1501                  ;Time Out Error
SET<21>=H,1502                  ;Block Size Error
SET<22>=H,1503                  ;Invalid Block Type
SET<23>=H,1504                  ;XOR Check Error
SET<24>=H,1505                  ;Sum Check Error
SET<25>=H,1506                  ;Insufficient Memory
SET<26>=H,1507                  ;Line Too Long for Memory Allocated in Buffer

SET<30>=H1510                   ;Unspecified Header Error
SET<31>=H,1511                  ;Position Out of Bounds
SET<32>=H,1512                  ;Out of Memory

SET<40>=H,1520                  ;Unspecified Header Error
SET<41>=H,1521                  ;Header Not Received
SET<42>=H,1522                  ;Header Length Does Not Match Specified Size
```

```
SET<50>=0                    ;No Line Description
SET<51>=0                    ;No Line Description Buffer
SET<52>=0                    ;Currently X and Y at Zero

G4 F1.
(UMFO,1,100)

(PORT,A,SET,FBK)
(COMM,A,#C:<SET<3>.LSL4.H,10>.OR2.$INP) ; We Are Ready, Show Input Line
G1 G23 F=SET<13>                        ; Set Feedrate
(MSG, PO RUNNING)
(SIOC,1)
(DVAR,INDX,OFST,TIM1,TIM2,TIM3,TIM4)
INDX=0 OFST=50                          ; Start From First Set
FBK<0>="ENT0" FBK<OFST>="ENT0"
TIM1=$TOD<m> TIM2=$TOD<S>    ; Start Time
(DENT,ENT0)
(JUMP,#FBK<INDX>)
(DENT,ENT1)
FBK <INDX>=0
(PORT,0)
TIM3=$TOD<m> TIM4=$TOD<S>
(MSG, ALL DONE, #TIM1 #TIM2 #TIM3 #TIM4)
M2
(DENT,ENT2)
(COMM,A,#C:FBK<INDX+1>)
FBK<INDX>="ENT0"
X=FBK<INDX+2> Y=FBK<INDX+3>
(PSOM,0,FBK<INDX+20>,FBK<INDX+5>)
(PSOF,4,FBK<INDX+6>,X)
X=FBK<INDX+4>
INDX=INDX+OFST OFST=-OFST
(JUMP,ENT0)

(DENT,ENT3)
(COMM,A,#C:FBK<INDX+1>)
```

```
FBK<INDX>="ENT0"
X=FBK<INDX+3>
(PSOM,0,FBK<INDX+20>,FBK<INDX+5>)
(PSOF,4,FBK<INDX+6>,X)
(FREE,X=2,F=FBK<INDX+8>,D=FBK<INDX+7>,Y=2,F=FBK<INDX+
10>,D=FBK<INDX+9>)
X=FBK<INDX+4> Y=FBK<INDX+3>
INDX=INDX+OFST OFST=-OFST
(JUMP,ENT0)
```

PSO PROGRAM FLOW

(PSOP,....)
SETUP FIRING OUTPUT MODE

(PSOS, . .) Scaling

(PSOD,...)
SETUP FIRING DISTANCE

(*PSO OPTIONAL FIRING CONTROL AND/OR OUTPUT CONTROL*

(PSOF,...)
ENABLE FIRING OUTPUT AND POSITION TRACKING

MOTION PROGRAM

.
.
.

(PSOF,0)
DISABLE FIRING

TO:	UNIDEX 21 PSO USERS
FROM:	Ron Rekowski
DATE:	December 9, 1991
SUBJECT:	**New PSO Functions**

Four new laser firing functions have been added to the UNIDEX 21 Position Synchronised Ouput Card (PSO). These new functions provide an output voltage proportional to the vector displacment and/or velocity of the user specified axis. Following is a listing of the command syntax for each of the functions along with a brief example of their operation.

Analog Output - Velocity Ramping - Bipolar DAC
(PSOT,4,n,d,m,v)

n D/A channel, $0 \leq n \leq 3$
d Analog output voltage at zero velocity, $-10 \leq d \leq 10$
m Maximum analog output voltage at target velocity, $-10 \leq m \leq 10$
v Target velocity, $-2^{23} \leq v \leq 2^{23} - 1$ (in $\frac{machine\ steps}{msec}$)

Analog Output - Velocity Ramping - Unipolar DAC
(PSOT,5,n,d,m,v)

n D/A channel, $0 \leq n \leq 3$
d Analog output voltage at zero velocity, $0 \leq d \leq 10$
m Maximum analog output voltage at target velocity, $0 \leq m \leq 10$
v Target velocity, $-2^{23} \leq v \leq 2^{23} - 1$ (in $\frac{machine\ steps}{msec}$)

NOTES:

- The user must specify a firing distance and pulse output mode along with the analog ramping output to enable tracking of the desired axis encoder feedback. The firing and pulse modes specified in the example program can be replaced by any valid PSO firing or pulse mode commands.

- The velocity ramping mode and the position ramping mode may both be active at the same time. Setting the position and velocity mode outputs for the same D/A channel will result in the summation of the two signals at the specified output.

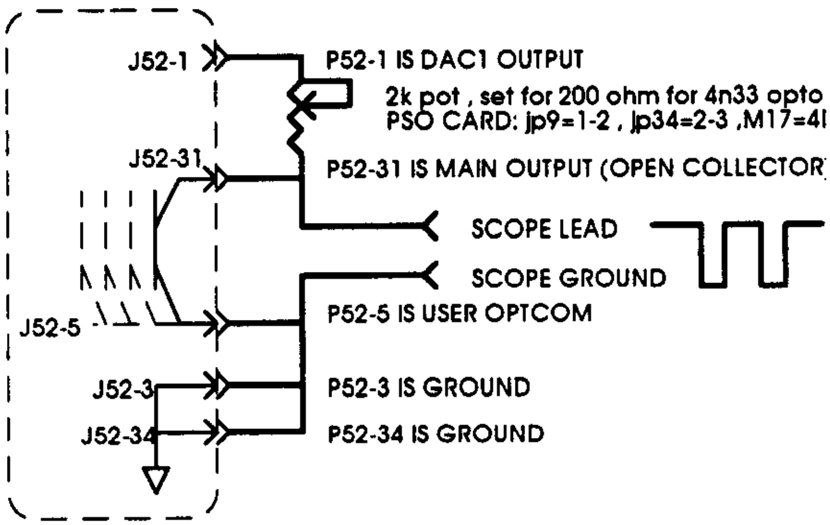
- The analog ramping modes can be disabled by two methods:

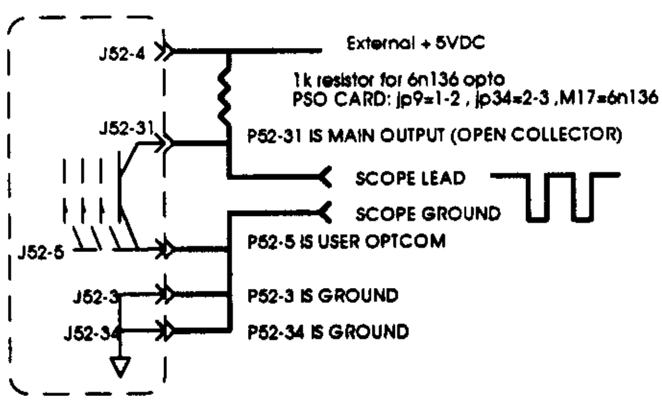
  1. By executing a (PSOF,0) command which disables the tracking of the encoder feedback and results in zero volts output for the position and/or velocity ramping outputs.
  2. By executing a (PSOT,1,*n*,*d*) command , where *n* is the channel of the position/velocity output to be disabled, and *d* is a user specified voltage.

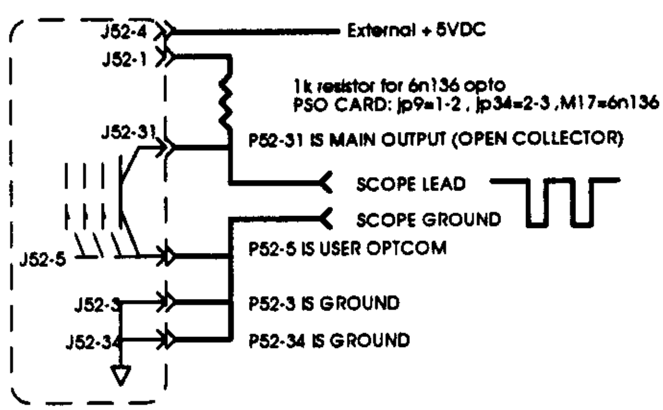
- Once a position and/or velocity ramping mode has been disabled, the ramping mode can only be re-enabled by issuing a new (PSOT,4/5/6/7,...) command followed by a new (PSOF,*n*) command.

PSOF2.DOC

```
;****************************************************************************
;PSO OUTPUT POLARITY MAY BE CHANGED BY CHANGING JP27
;USING PSO CONNECTOR ON BACK OF U21 LABELED P52
;MAIN OUTPUT IC ON PSO CARD IS M17,AUX2=M16,AUX3=M15,AUX4=M14 4N33 IS STANDARD
;INPUT 1 IC ON PSO CARD IS M10,IN2=M11,IN3=M12,IN4=M13        6N136 IS STANDARD
;WHEN USING 6N136 / HCPL2601 FOR OUTPUTS YOU MUST SUPPLY +5VDC TO J52 PIN4
;RECOMMENDED LOAD FOR 6N136 IS 1000 OHM, FOR HCPL2601 IS 100 OHM AT 5 VDC MAX
;RECOMMENDED LOAD FOR 4N33 IS 200 OHM ,WHEN USING 5VDC.
;J52 CONNECTOR WIRING INFO: CONNECT J52-3 TO J52-5,J52-31 TO RESISTOR LOAD
;J52-31 TO RESISTOR LOAD , +5VDC OR DAC OUTPUT TO OTHER END OF RESISTOR LOAD
;DAC2 OUTPUT IS PIN 26 ON J52 CONNECTOR
;ecs100=4n33,ecs101=6n136,ecs119=hcpl2601,eck572=50pin mate
;****************************************************************************
;THIS IS SAMPLE PROGRAM FOR U21 PSO CARD
G1 G90 F1.0
(PSOT,2,0,5)          ;DAC0 MAIN OUTPUT,DAC3 IS FOR 6N136/HCPL2601 POWER
(PSOP,1,50,50,0)      ;PULSE WIDTH IS 5MSEC
(PSOF,2,1000)         ;FIRE 1000 PULSES
M47
.
```







CHAPTER 5: IEEE-488 OPTION

The IEEE-488 option for the Unidex 21 provides control of the Unidex 21 from a host computer through the IEEE-488 Bus. Once communication is established, the command sequence and operating instructions are as described in the *Unidex 21 Programming Manual* and the *Unidex 21 User's Manual.*

SECTION 5-1 HARDWARE REQUIREMENTS

5-1-1 IEEE-488 INTERFACE

IEEE-488 has 8 data lines and 8 control lines. (Refer to the *Unidex 21 Hardware Manual* for connector details.) It can accommodate up to 14 devices and provides a Service Request line from all devices to the Bus Controller. These properties lead to a more rapid form of communication between Unidex 21 and the controller. Bus disciplines are not necessary if the controller has IEEE-488 interface and device driver software that interfaces with the language to be used.

5-1-2 SIGNAL LINES OF THE IEEE-488 BUS

The IEEE-488 transfers data and commands between devices through 16 signal wires.

Eight of the lines are for the transfer of data (DI01 to DI08). Data and message transfers are asynchronous and are coordinated by the three handshake lines.

The remaining five lines, for example "ATN" (Attention) and "SRQ" (Service Request), are used for bus management. Each line, when asserted Low (ground), represents a single line message sent on the bus. (Refer to the *Unidex 21 Hardware Manual* for a description of each of these lines.)

5-1-3 CABLE RESTRICTIONS OF THE IEEE-488 BUS

The devices in a system are connected by a 24-wire cable using 24-pin connectors as specified in the IEEE-488 standard.

Certain limitations exist concerning the length of the cables and the number of devices allowable on the bus. The maximum number of devices on the bus is 14. The total length of the cable is limited to 20 meters (65.6 feet) or 2 meters multiplied by the number of devices (whichever is shorter in length). A list of cable suppliers follows:

CABLE MANUFACTURERS

HEWLETT-PACKARD
Palo Alto, California 94304

PN	Length
HP 10833D	.5 Meter
HP 10833A	1 Meter
HP 10833B	2 Meters
HP 10833C	4 Meters
HP 10834A	Adapter

BELDEN CORPORATION
Richmond, Indiana 47374

PN	Length
9642	1 Meter
9643	2 Meters
9644	4 Meters
9645	8 Meters
9646	16 Meter

SECTION 5-2: SET UP

Connect the Controller to the IEEE-488 Connector (P11) on the Rear Panel of the Unidex 21.

NOTE: **In order for the Unidex 21 to recognize the IEEE-488 interface, connection must be made to the Unidex 21 before System power up or Reset.**

Power Up the Unidex 21.

5-2-1 INITIAL CONFIGURATION

Remote operation of the Unidex 21 may be initiated with several configurations. The following is a list of the possible configurations and the key combinations that must be entered to initiate them.

NOTE: The following keyboard entries may be made either from the Remote Controller or the Unidex 21's Front Panel.

Condition	Keyboard Entry
Remote Control OFF	Ctrl\0
Remote Control ON, Display OFF	
RS-232 Port A	Ctrl\1
IEEE-488	Ctrl\3
Remote ON, Display ON	
RS-232 Port A	Ctrl\2
IEEE-488	Ctrl\4

5-2-2 PARAMETER SETTINGS

Before using an IEEE-488 device to control the Unidex 21, certain Parameter settings must be established. The paragraphs that follow provide instructions for setting the applicable parameters. For a detailed description of all parameter settings refer to the *Unidex 21 User's Manual.*

The Initial Selection Screen shown below is displayed upon power-up of the Unidex 21:

```
UNIDEX 21     Version XX

EPROM OK    PARAMETER OK     RAM checksum


User's RAM (bytes) = xxxxxxxx

Edit, File, Machine, Parameter, Test, System, Batch, Console, Debug
```

Press the "P" key to enter the Parameter Mode.

The following screen will be displayed:

```
0 : System password                    1 : Skip auto-boot function ?
2 : IDX buffer 1 block only ?          3 : IDX seg. calculate base (1/2/3)
4 : COMM input feedback ?              5 : System default at metric ?
6 : RS232 protocol port-A              7 : Additional RAM in 1024 bytes
8 : RS232 protocol port-B              9 : Debug display is at front panel ?
10 : RS232 time out (seconds)          11 : Parts program stack size in bytes
12 : Edit block buffer (1 to 40)       13 : Edit default Char-insert ?
14 : Edit default Line-insert ?        15 : Edit TAB space
16 : End of all file code CHR$(n)      17 : End of file code CHR$(n)
18 : Beeper duration (1 to 280) ms     19 : Double side floppy disk ?
20 : Beeper frequency (2 to 20K)       21 : Display blank-out (minutes)
22 : MFO inc./step (0 to 100)          23 : Tracking display program step ?
24 : Y pixel size reduce to (%)        25 : Print screen to port-A ?
26 : Joystick axis pair                27 : Digitize with joystick ?

200 : NEXT PAGE                        201 : Axes auto-tune

300 : Load/save parameter              301 : Front panel function keys
401 : 1st axis      402 : 2nd axis     403 : 3rd axis      404 : 4th axis
405 : 5th axis      406 : 6th axis     407 : 7th axis      408 : 8th axis

ctrl-Quit, number <cr> to each parameter =
```

5-2-2-1 COMM INPUT FEEDBACK

General Parameter 4, Communication Input Feedback, determines whether "echo" characters are required for RS-232 or IEEE-488 input. The default setting requires "echo" characters. Refer to the *Unidex 21 Programming Manual* for details concerning the (COMM command.

Enter "4" to change the Feedback status. Press the "N" key to toggle between Yes (feedback echoes are required) or No (no echo characters required).

5-2-2-2 TIME-OUT

The Unidex 21 contains a time-out feature when files are input or output through the RS-232 or IEEE-488 ports. When the IEEE-488 mode of file transmission is initiated, the Unidex 21 will "look" for the data for a predetermined amount of time before displaying an error message. The default time is 600 seconds.

Enter "10" to set the length of time the Unidex 21 will wait for a return signal following an IEEE-488 transmission. Enter the new time in seconds.

5-2-2-3 END OF ALL FILE CODE

General Parameter 16 provides the User the ability to establish a character that will signal to the Unidex 21 that data transmission is complete. The default character is 17. (See also Parameter 53, IEEE-488 Set Up)

Enter "16" to change the End of All File Character. Enter the new End of All File character(s).

5-2-2-4 END OF FILE CODE

A character may be established to signal the Unidex 21 that a file data transmission is complete. General Parameter 17 provides the User the ability to establish an End of File Code for each system. The default character is "9". (See also Parameter 53, IEEE-488 Set Up)

Enter "17" to change the End of File character. Enter the new End of File character.

5-2-2-5 POWER ON REMOTE CONTROL

General Parameter 31 may be configured such that the Unidex 21 will be in the desired Remote state following a power-up or reset. The default is for no Remote Control.

Enter "31"to change the Remote Power status.

Enter "0" for no Remote Control upon power-up or reset.

Enter a "1" to establish Remote Control of the Unidex 21 through RS-232 Port A following a power-up or reset. The Unidex 21 Front Panel display will not be active.

Enter a "2" to establish Remote Control of the Unidex 21 through RS-232 Port A following a power-up or reset. The Unidex 21 Front Panel display is active.

Enter a "3" to establish Remote Control of the Unidex 21 through the IEEE-488 Port following a power-up or reset. The Unidex 21 Front Panel display will not be active.

Enter a "4" to establish Remote Control of the Unidex 21 through the IEEE-488 Port following a power-up or reset. The Unidex 21 Front Panel Display will be active.

5-2-2-6 IEEE-488 SET-UP

General Parameter 53 provides the User with a variety of set-up parameters for use specifically with a IEEE-488 interface.

Enter "53" to establish IEEE-488 parameters. The display will be:

IEEE488 SET UP

0: address mode (0 talk only) (1 listen only) (2 major only) (3 major/minor)
(4 primary/secondary) (5 primary/primary) = 2
1: 1st address (0 to 31) = 2
2: 2nd address (0 to 31) = 3
3: PPR (0 no) (1 to 8 - in phase) (9 to 16 - reverse phase) = 1
4: EOS data (0 to FF) = 0A
5: EOS bits (0-7) (1-8) = 1 EOS on [illegible]
6: set EOI with ~~last byte of write?~~ (0 - yes) (1 - no) = 0
7: terminate read on EOS? (0 - yes) (1 - no) = 0
8: set EOI with last byte of write? (0 - yes) (1 - no) = 0

NOTE: EOS will not affect EOI during File mode Input/Output case
Input - Unidex 21 will wait for EOI or end-of-file code
Output - Unidex 21 will set EOI with end-of-file code

Ctrl-Quit, Ctrl Default , code/nnnnnnnn =

A description of each of the IEEE-488 parameters follows:

Code 0 - Establishes the mode to which the Unidex 21 is to be addressed. The default is setting "2", the Unidex 21 is addressed as Major (1 Address only)

Enter "0/0" to configure the Unidex 21 as a device that only sends data to receivers (Talker).

Enter "0/1" to configure the Unidex 21 as a device that only receives data messages from a Talker (Listener).

Enter "0/2" to configure the Unidex 21 for 1 Address Bit.

Enter "0/3" to configure the Unidex 21 for 2 Address Bits, one being Major and one being Minor. (Either may be assigned to be a Talker or a Listener.)

Enter "0/4" to configure the Unidex 21 such that the 1st address is Primary (Talker) and the 2nd address is Secondary (Listener).

Enter "0/5" to configure the Unidex 21 such that both the 1st address and the 2nd address are Primary (Talkers)

Code 1 - Establishes the address byte for the 1st address. Enter the desired address byte for the 1st address (1/1 thru 31). The default address byte is 2.

Code 2 - Establishes the address byte for the 2nd address. Enter the desired address byte for the 2nd address (2/1 thru 31). The default address byte is 3.

Code 3 - Configures Parallel Poll Response. The default configuration is In-Phase Parallel Polling at address "1".

Enter "3/0" for no Parallel Polling.

Enter"3/1 thru 8" to establish an address for In-Phase Parallel Polling of the Unidex 21. (Selected Bit will go "High" with Parallel Poll.)

Enter"3/9 thru 16" to establish an address for Reverse Phase Parallel Polling of the Unidex 21. (Selected bit will go "Low" with Parallel Poll.)

Code 4 - Establishes the character(s) (0 thru FF) used to terminate a read or an output operation. The default End Of String data is 0A.

Code 5 - The number of bits available for EOS data must be delineated by this parameter. ("0" establishes 7 bits, "1" establishes 8 bits) The default is "1" for an 8 bit system.

Code 6 - The End Of Identify signal may be sent with the End Of String signal to indicate to the Unidex 21 the last byte of the data string to be transmitted. The default is for the EOI signal to accompany the EOS signal.

Enter "6/0" for the EOI signal occur in conjunction with the EOS signal.

Enter "6/1" for no EOI signal.

Code 7 - The EOS signal may be used to indicate read data termination. This parameter may be used to configure the Unidex 21 such that the EOS signal will or will not terminate the reading of data. The default is "0", the reading of data will be terminated by the EOS signal.

Enter "7/0" if the EOS signal is to terminate the reading of data.

Enter "7/1" if the EOS signal is not to terminate the reading of data.

Code 8 - The EOI signal may be set to be used in conjunction with the last byte of the write signal to eliminate the need for an EOS character at the end of every data string.

Enter "8/0" to set the EOI signal with the last byte of the write signal.

Enter "8/1" if the EOI signal is not to be sent with the last byte of the write signal.

SECTION 5-3: SERVICE REQUEST AND POLLING

5-3-1 SERVICE REQUEST

A Service Request signal is necessary in remote operation where a host controller is a master and a controlled device is a slave. The purpose of the Service Request signal is for the slave device to catch the attention of the master controller.

The slave device has the capacity to send a request signal to the master controller whenever it requires the attention of the master. The reason for the request may be an error condition or the completion of a task.

NOTE: **If the Service Request was initiated by an error condition, the Unidex 21 will not respond to any further system commands until it is serial polled by the master controller.**

The Unidex 21 implements a Service Request by asserting the SRQ line on the IEEE-488 bus. The master controller may be programmed to be interupted by a SRQ and to take the necessary action.

5-3-1 PARALLEL POLLING

Parallel Polling is done to identify configured devices and indicate to the host controller when a device on the IEEE-488 bus is requesting service (SRQ). A composite poll response is sent to the host controller. The host controller then may Serial Poll the devices to determine the device number and the nature of the service request.

The Parallel Poll bit assigned to each Unidex 21 is selected in the Parameter Mode (See Section 5-2.)

5-3-2 SERIAL POLLING

Serial Polling is done on one device at a time to determine which device has made a Service Request and the reason for the request. Any device may be polled at any time, regardless of the number of devices on the line.

A Unidex 21 will Request Service (set SRQ) at specific times, such as when a program is completely executed. At such a time, further operations will be suspended until Unidex 21 is Serial Polled by the Controller. Upon being polled, the Unidex 21 will transmit its status.

The Unidex 21 sends following status codes as a result of a Serial Poll:

CODE	DESCRIPTION
40H	Remote status good, awaits instruction
C0H	Remote Error, read 1 or 2 byte error code. Unidex 21 awaits Serial Poll.
41H	Remote status good, awaits Input
42H	Remote status good, awaits Output

SECTION 5-4: OPERATION

Following initialization, the Unidex 21 is controlled by the Host Controller.

Communication from the Host Controller to the Unidex 21 is accomplished in the same manner as communication from the TeleVideo 905 Terminal. (See Chapter 2 of the *Unidex 21 User's Manual.*)

NOTE: **Regardless of the keyboard configuration of the Host Controller, communication to the Unidex 21 must follow TeleVideo 905 Terminal input conventions**

The next Section provides a complete list of possible error codes and their corresponding messages.

5-4-1: ERROR CODES AND MESSAGES

During data transmission and/or performance of a function, if an error is detected the Unidex 21 will feed back an error code in the following format:

Master error Code (C0H) followed by the Secondary Error Code (1 or 2 bytes)

The following is a list of the Secondary Error Codes and Messages as well as the function from which they may occur.

5-4-1-1: EDIT MODE

The following Secondary Error Codes/Messages may appear while in the Edit Mode:

10H - Input key undefined
11H - Not enough User's RAM space
12H - File format error
13H - File not found
14H - File read only
15H - Block functions got range error
16H - Input key not ctrl-Q or ctrl-W
17H - Input key not Y or N

5-4-1-2: FILE MODE

The following Secondary Error Codes/Messages may appear while in the File Mode:

20H - Input key undefined
21H - Undefined I/O port
22H - File format error
23H - File not found
24H - File read only
25H - File currently active
26H - No disk
27H - Not enough User's RAM space
28H - File verify error
29H - RS-232/IEEE-488 time out, or transfer interface fail

2AH - Target file already exists
2BH - Not enough disk space
2CH - Disk write protected
2DH - Disk access fail
2EH - Disk up load fail

5-4-1-3: MACHINE MODE

The following Secondary Error Codes/Messages may appear while in the Machine Mode:

30H - Input key undefined
31H - File not found
32H - Illegal filename.type
33H - Sub-program not found
34H - Can't open read file
35H - Can't open write file
36H - Write file not closed

40H - Undefined symbol
41H - Format error
42H - Undefined Type 2 command
43H - Undefined G code
44H - Undefined M code
45H - Illegal BCD format
46H - Illegal system variable
47H - Undefined variable
48H - Illegal I/O format
49H - Illegal mathematics format
4AH - Undefined array
4BH - Miss CLS command
4CH - Undefined subroutine
4DH - Undefined entry
4EH - Undefined condition
4FH - Stack overflow

50H - Miss return address
51H - Undefined safe zone
52H - Illegal function in MDI
53H - Not enough memory space
54H - Circle miss center point
55H - No feed rate

56H - Move into safe zone
57H - Undefined data in read file
58H - In ICRC look ahead
59H - <no>
5AH - MALC format error
5BH - CPAG format error, or need (MALC, < 1, option
5CH - Undefined H code
5DH - Undefined axis plane
5EH - Axis can't be both master & slave, or more than 1 master
5FH - PLC Option not foun, or ladder program not exist
60H - Need (MALC to allocate memory
61H - No recorded position to play back, need (RECO
62H - PSO Option not found

5-4-1-4: PARAMETER MODE

The following Secondary Error Codes/Messages may appear while in the Parameter Mode:

70H - Input key undefined
71H - Input data error
72H - Not enough memory for p-meter save
73H - File exist already for p-meter save
74H - File not found for p-meter load

5-4-1-5: TEST MODE

The following Secondary Error Codes/Messages may appear while in the Test Mode:

80H - Input key undefined
81H - RAM fail at (0) case
82H - RAM fail at (F) case
83H - RAM fail at (5) case
84H - RAM fail at (A) case
85H - RAM checksum error
86H - EPROM checksum error
87H - PARAMETER checksum error

5-4-1-6: SYSTEM MODE

The following Secondary Error Codes/Messages may appear while in the System Mode:

90H - Input key undefined
91H - TIME input error
92H - DATE input error

5-4-1-7: MISC. ERRORS

The following Secondary Error Codes/Messages may also appear during Remote operation:

A0H - Input key undefined
A1H - No password privlege
A2H - Batch file not found or format error
A3H - RAM error during power on test
A4H - Indexing board error during power on test
A5H - Real time clock fail, set at default data

5-4-1-8: SPECIAL REMOTE SYSTEM-FAIL ERROR

During data transmission and/or performance of a function, a Special Remote System error having two bytes of Secondary Error may be detected, it will be displayed in the following format:

Master Error Code (C0H) followed by 0E0H and the Secondary Error Code

The following Secondary Error Codes/Messages may appear during data transmission and/or performance of a function:

80H - Indexer 68000 CPU Bus Error
81H - Indexer 68000 Address Error
82H - Indexer 68000 Illegal Instruction
83H - Indexer 68000 Zero Divide
84H - Indexer 68000 Line 1010 Emulation
85H - Indexer 68000 Line 1111 Emulation

86H - Indexer 68000 Uninitialized Interrupt Vector
87H - Indexer 68000 Spurious Interrupt
88H - Indexer Dual-Port Ram Group B Checksum
89H - Indexer Dual-Port Ram Group B Data Out of Boundary
8AH - Feedrate is 0 or Negative Value
8BH - Invalid Sin/Cos Combination
8CH - Invalid Contouring Plane

A0H - Axis in Limit (Software or Hardware)
A1H - Axis Trap (Velocity or Position or Integral)
A2H - M Function Output Fail to Detect the Acknowledge Signal
A3H - S Function Output Fail to Detect the Acknowledge Signal
A4H - T Function Output Fail to Detect the Acknowledge Signal
A5H - DSP Feedback Illegal Code

B0H - MFO = 0 or Feedhold is On
B1H - AC Fail
B2H - Joy-Stick/ Trackball/Handwheel Motion Hit Software or Hardware Limit

SECTION 5-5: SAMPLE PROGRAM

The following program is representative of a QuickBasic Program (Version 4.0 +) that may be sent to the Unidex 21 from a Host Controller.

> NOTE: **This program is applicable only to controllers equipped with the GPIB Interface Board.(National Instruments, Austin, Texas)**

```
' Common GPIB staus variables:
COMMON SHARED /NISTABLK/ IBSTA%, IBERR%, IBCNT%
' GPIB Subroutine Declarations:

    DECLARE SUB IBBNA (BD%, BDNAME$)
    DECLARE SUB IBCAC (BD%, V%)
    DECLARE SUB IBCLR (BD%)
    DECLARE SUB IBCMD (BD%, CMD$)
    DECLARE SUB IBCMDA (BD%, CMD$)
    DECLARE SUB IBDMA (BD%, V%)
    DECLARE SUB IBEOS (BD%, V%)
    DECLARE SUB IBEOT (BD%, V%)
    DECLARE SUB IBFIND (BDNAME$, BD%)
    DECLARE SUB IBGTS (BD%, V%)
    DECLARE SUB IBIST (BD%, V%)
    DECLARE SUB IBLOC (BD%)
    DECLARE SUB IBONL (BD%, V%)
    DECLARE SUB IBPAD (BD%, V%)
    DECLARE SUB IBPCT (BD%)
    DECLARE SUB IBPPC (BD%, V%)
    DECLARE SUB IBRD (BD%, RD$)
    DECLARE SUB IBRDA (BD%, RD$)
    DECLARE SUB IBRDF (BD%, FLNAME$)
    DECLARE SUB IBRDI (BD%, IARR%( ), CNT%)
    DECLARE SUB IBRDIA (BD%, IARR% ( ), CNT%)
    DECLARE SUB IBRPP (BD%, PPR%)
    DECLARE SUB IBRSC (BD%, V%)
```

```
DECLARE SUB IBRSP (BD%, SPR%)
DECLARE SUB IBRSV (BD%, V%)
DECLARE SUB IBSAD (BD%, V%)
DECLARE SUB IBSIC (BD%)
DECLARE SUB IBSRE (BD%, V%)
DECLARE SUB IBSTOP (BD%)
DECLARE SUB IBTMO (BD%, V%)
DECLARE SUB IBTRAP (MASK%, MODE%)
DECLARE SUB IBTRG (BD%)
DECLARE SUB IBWAIT (BD%, MASK%)
DECLARE SUB IBWRT (BD%, WRT$)
DECLARE SUB IBWRTA (BD%,WRT$)
DECLARE SUB IBWRTF (BD%, FLNAME$)
DECLARE SUB IBWRTI (BD%, IARR%( ), CNT%)
DECLARE SUB IBWRTIA (BD%, IARR%( ), CNT%)

' GPIB Function Declarations

DECLARE FUNCTION ILBNA% (BD%, BDNAME$)
DECLARE FUNCTION ILCAC% (BD%, V%)
DECLARE FUNCTION ILCLR% (BD%)
DECLARE FUNCTION ILCMD% (BD%, CMD$, CNT%)
DECLARE FUNCTION ILCMDA% (BD%, CMD$, CNT%)
DECLARE FUNCTION ILDMA% (BD%, V%)
DECLARE FUNCTION ILEOS% (BD%, V%)
DECLARE FUNCTION ILEOT% (BD%, V%)
DECLARE FUNCTION ILFIND% (BDNAME$)
DECLARE FUNCTION ILGTS% (BD%, V%)
DECLARE FUNCTION ILIST% (BD%, V%)
DECLARE FUNCTION ILLOC% (BD%)
DECLARE FUNCTION ILONL% (BD%, V%)
DECLARE FUNCTION ILPAD% (BD% V%)
DECLARE FUNCTION ILPCT (BD%)
DECLARE FUNCTION ILPPC% (BD%, V%)
DECLARE FUNCTION ILRD% (BD%, RD$, CNT%)
DECLARE FUNCTION ILRDA% (BD%, RD$, CNT%)
DECLARE FUNCTION ILRDF% (BD%, FLNAME$)
```

```
DECLARE FUNCTION ILRDI% (BD%, IARR%,( ), CNT%)
DECLARE FUNCTION ILRDIA% (BD%, IARR%( ), CNT%)
DECLARE FUNCTION ILRPP% (BD%, PPR%)
DECLARE FUNCTION ILRSC% (BD%, V%)
DECLARE FUNCTION ILRSP% (BD%, SPR%)
DECLARE FUNCTION ILRSV% (BD%, V%)
DECLARE FUNCTION ILSAD% (BD%, V%)
DECLARE FUNCTION ILSIC% (BD%)
DECLARE FUNCTION ILSRE% (BD%, V%)
DECLARE FUNCTION ILSTOP% (BD%)
DECLARE FUNCTION ILTMO% (BD%, V%)
DECLARE FUNCTION ILTRAP% (MASK%, MODE%)
DECLARE FUNCTION ILTRG% (BD%)
DECLARE FUNCTION ILWAIT% (BD%, MASK%)
DECLARE FUNCTION ILWRT% (BD%, WRT$, CNT%)
DECLARE FUNCTION ILWRTA% (BD%, WRT$, CNT%)
DECLARE FUNCTION ILWRTF% (BD%, FLNAME$)
DECLARE FUNCTION ILWRTI% (BD%, IARR%( ), CNT%)
DECLARE FUNCTION ILWRTIA% (BD%, IARR%( ), CNT%)

CLS

DIM B$(255)


BOARD$ = "GPIB0"
CALL IBFIND(BOARD$, BOD%)
IF BOD% < 0 THEN 1000

PRINT "BOARD DESCRIPTOR = ", BOD%

BOARD$ = "U21"
CALL IBFIND(BOARD$, EQP%)
IF EQP% < 0 THEN 2000

PRINT "DEVICE DESCRIPTOR = ", EQP%
```

```
      CALL IBCLR (EQP%)

5     PRINT " "
      PRINT " 0 - REMOTE CONTROL  1 - INPUT FILES"
      PRINT " 2 - OUTPUT FILES    3 - INPUT THEN OUTPUT STRING"
6     PRINT " "
      INPUT " SELECT 0 TO 3 ONLY = ", SE

      IF SE = 0 THEN 10
      IF SE = 1 THEN 20
      IF SE = 2 THEN 30
      IF SE = 3 THEN 40
      GOTO 6

10    PRINT "KEY INPUT = ";
11    A$ = INKEY$
      IF A$ = " " THEN 11
      PRINT ASC(A$); A$
      GOSUB 200
      GOTO 10

20    NST% = 64 +2
      GOSUB 400

      ' PRINT "ST% = "; ST%

      MASK% = &H6000

      B$ = SPACE$(250)
      CALL IBRD(EQP%, B$)

21    CALL IBWAIT(EQP%, MASK%)
      IBS% = IBSTA% \ 256

      IF IBS% < 64 THEN 22
      IBS% = IBS% - 64
22    IF IBS% < 32 THEN 21
```

```
        PRINT B$

25      INPUT "FILE INPUT DONE, 0-QUIT, 1-SENT BACK "; SE
        IF SE = 0 THEN 5
        GOTO 31
30      INPUT "STRING TO OUTPUT = ", B$

31      NST% = 64 + 1
        GOSUB 400

        ' PRINT "ST% ="; ST%

        CALL IBWRT(EQP%, B$)
        INPUT "OUTPUT DONE, 0-QUIT, 1-MORE"; SE
        IF SE = 0 THEN 5
        GOTO 30

40      CAS = 0
41      NST% = 64 + 2
        GOSUB 400

        MASK% = &H6000
        B$ = SPACE$(30)
        CALL IBRD(EQP%, B$)
42      CALL IBWAIT(EQP%, MASK%)

        IBS% = IBSTA% \ 256
        IF IBS% < 64 THEN 43
        IBS% = IBS% - 64
43      IF IBS% < 32 THEN 42

        IF CAS < > 0 THEN 350
        PRINT B$

        INPUT "DATA TO OUTPUT = ", DA$
        DA$ = DA$ + CHR$(13)
```

```
     NST% = 64 + 1
     GOSUB 400

     CALL IBWRT(EQP%, DA$)

     GOTO 40

200  CALL IBWRT(EQP%, A$)
     MASK% = &HE800
210  CALL IBWAIT(EQP%, MASK%)

     IF IBSTA% / 256 > = 128 THEN 300
     IF IBSTA% / 256 > = 64 THEN 310
     IF IBSTA% / 256 > = 32 THEN 320
     IF IBSTA% / 256 > = 8 THEN 330

     GOTO 210

300  PRINT "GPIB ERROR"
     RETURN
310  PRINT "TIME OUT"
     RETURN
320  PRINT "DETECT END"
     RETURN
330  PRINT "SERIAL POLL = ";
     CALL IBRSP(EQP%, ST%)
     IF ST% = 192 THEN 340
     PRINT ST%
335  RETURN
340  PRINT ST%;
     CAS = 1
     GOTO 41
350  PRINT ASC(MID$(B$, 1, 1)); ASC(MID$(B$, 2, 1))
     RETURN
```

```
400  MASK% = &H4800
410  CALL IBWAIT(EQP%, MASK%)
     IBS% = (IBSTA% \ 256) AND 72

     IF IBS% < 72 THEN 420
     IBS% = IBS% - 64
420  IF IBS% < 8 THEN 410

     CALL IBRSP(EQP%, ST%)
     IF ST% = NST% THEN 335
     GOTO 410

1000 PRINT "BOARD NOT FOUND"
     STOP
2000 PRINT "DEVICE NOT FOUND"
3000 STOP

     END
```