UNIDEX™ 3
MOTION CONTROLLER
USER'S MANUAL

PN: EDU107

Rev. B-3Q Software

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

February, 1989

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

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.

MANUAL CONVENTIONS

Keyboard entries will be designated by upper case letters enclosed in brackets, such as [PRT].

Unidex IIIa display responses will be underlined

Remote entries will be designated by upper case letters, such as PX <CR> <LF>.

A long string of remote commands will be enclosed in quotes, such as "<ESC> 10, 13, 15" <CR><LF>.

NOTE

Remote entries <ESC> and <CR> <LF> are usually output by the controller via character string commands CHR$(nn). (See appendix 8 for a list of ASCII character codes.) For example:

```
IBM:      PRINT #1, CHR$(27)+"nn"
HP-85     OUTPUT 10, CHR$(27)&"nn"
```

These commands cause Unidex IIIa to enter remote operation using the RS-232C interface.

CHAPTER 1: INTRODUCTION

Unidex IIIa is a complete, extremely fast numerical motion controller, designed to control two axes of DC servo or stepping motor drives. Unidex IIIa's powerful set of motion control commands, battery-backed memory, and versatile interface capability gives it the competence to handle virtually all point-to-point motion control applications.

Unidex IIIa's motion control capabilities include vectorial as well as independent axis feedrates, joystick operation, programmable inputs, conditional skips, and interrupts during moves. Its internal position registers allow ±2,000,000,000 increments of system resolution for each axis. As a stand-alone motion controller, programs or individual motions can be entered, edited, and executed from the front panel controls. Its expandable memory (4 kilobytes standard, 28 kilobytes maximum) stores up to 99 programs which can be randomly accessed either locally (through the Unidex IIIa's keyboard) or remotely (by a controller).

A controller (host computer) can be used to develop and edit a program, download it into the Unidex IIIa and then be disconnected. Unidex IIIa can execute a program from memory and communicate via interrupts with the host, or the host can download individual motions into the Unidex IIIa. If used with a host, the communication bus can be any of the following three:

1. IEEE-488
2. RS-232C
3. RS-422A

Connections for all three interfaces, the 8 buffered outputs, 4 buffered inputs, and the joystick are found on the rear panel of the Unidex IIIa. The RS-232C and RS-422A interfaces have multi-axis daisy-chain capabilities. These connections are also included on the rear panel. A complete listing of all Unidex IIIa features is included at the end of this section.

SECTION 1-1 UNIDEX IIIA OPERATOR'S MANUAL

It is recommended that this manual be read before you attempt the installation or programming of Unidex IIIa.

This chapter (Introduction) and chapter 3 (Getting Acquainted with Unidex IIIa) will familiarize you with Unidex IIIa and some basic programming techniques.

Chapter 2, Installation, gives unpacking instructions and a description of switches, connections, components and configurations. It also describes the methods of interfacing Unidex IIIa(s) with a controller.

Chapter 4, System Programming, describes both local and remote programming as well as interfacing Unidex IIIa with a controller.

Chapter 5, Commands For Motion Programming, is a description of the language required for programming Unidex IIIa.

Chapter 6, Advanced Programming Operations, covers the more intricate motion programming commands, and includes sample programming as well.

Chapter 7 covers Service And Repair.

SECTION 1-2 FEATURES OF UNIDEX IIIA

The features and capabilities of Unidex IIIa are as follows:

- 2 axis control - ±2,000,000,000 steps maximum
- Remote interface - RS-232C, RS-422A and IEEE-488
- Multi-axis daisy chaining capability for the remote interfaces

- Single front panel may control up to 8 axis pairs (optional)
- 8 buffered outputs with strobe
- 4 buffered inputs
- Joystick operation
- Alphanumeric display - 8 characters long
- X and Y tracking displays
- Stored program capability (multiple programs randomly accessible)
- 4K storage - battery back-up (up to 28K optional)
- Downloadable from remote (controller)

PROGRAMMABLE FUNCTIONS

- Incremental mode - absolute mode
- Corner rounding/non-corner rounding
- Repeat cycles
- Dwell
- Unconditional jumps
- Subroutines (stack and jump with return)
- Store (remember) positions on command and recall
- Send X, Y and M statuses to remote controller on command
- Programs printed or sent to controller from Unidex IIIa
- Remote programming

- Trigger mode - allows multi-axis synchronization
- Independent feedrates for both axes - up to 150 KHz (actual value programmed will depend on the capability of the drives)
- Vector feedrate mode (linear interpolation between two points)
- Register operations and register based moves
- Programs executable as subroutines
- Abort on Interrupt
- Reset system on Interrupt
- Feedhold on Interrupt
- Multiple inputs concurrently programmable as interrupt inputs
- Reset system by program statement
- Programmable message display
- Programmable message buffer
- Internal register and I/O status monitoring and display
- Interactive data input from keyboard
- Acceleration/deceleration programming

EDIT FUNCTIONS

- Step
- Backstep
- Search
- Clear entry
- Clear command
- Start

CHAPTER 2: INSTALLATION

SECTION 2-1 UNPACKING UNIDEX IIIA

Remove the system from its shipping container and, referring to the sales order, verify that all items are present. Save the packing material for storing or reshipping the system.

IMPORTANT: **If the shipping container is damaged upon receipt, request that the carrier's agent be present while the system is being unpacked and inspected.**

The system should be inspected upon receipt for broken, damaged or loosened parts. Retighten any loosened connectors.

SECTION 2-2 COMPONENTS

The five subsections of Unidex IIIa are:

1. Power supply
2. MPU board
3. Front panel board
4. Rear panel board (Interface board)
5. MPX board (optional)

In most applications, these components are shipped fully connected and packaged in a 19" rack mount chassis which includes one or two drives (stepping motor translators or DC servo system). You will find all inputs and outputs on the rear panel.

The power supply generates all of the DC power required by the MPU board and the panel board, as well as a typical complement of serial load boards, ramper boards and encoders.

The MPU board executes the program for axis control and stores up to 28K of user programs in battery-backed memory. In addition, the MPU board has the following features:

- Indexing logic and buffered indexer outputs
- Eight buffered "$\overline{M}$" outputs with strobe: TTL compatible high current drivers (figure 2-3)
- Four buffered "C" (control) inputs: one CMOS load each (figure 2-3)
- Joystick interface (figure 2-3)

The front panel board has a display control and keyboard interface, as well as a keyboard, mode select switches, reset switch, alphanumeric display and two numeric tracking displays. In the multi-axis option, MPX, there is also an axis select switch. All front panel board controls are accessible from the front panel.

SECTION 2-3 FRONT PANEL CONTROLS AND DISPLAY

The front panel (figure 2-1) has a 16 key alphanumeric keyboard and five mode select keys. The mode select keys are provided with LED indicators and comprise the following:

1. EDIT
2. AUTO
3. SINGLE
4. IMMEDIATE
5. REMOTE

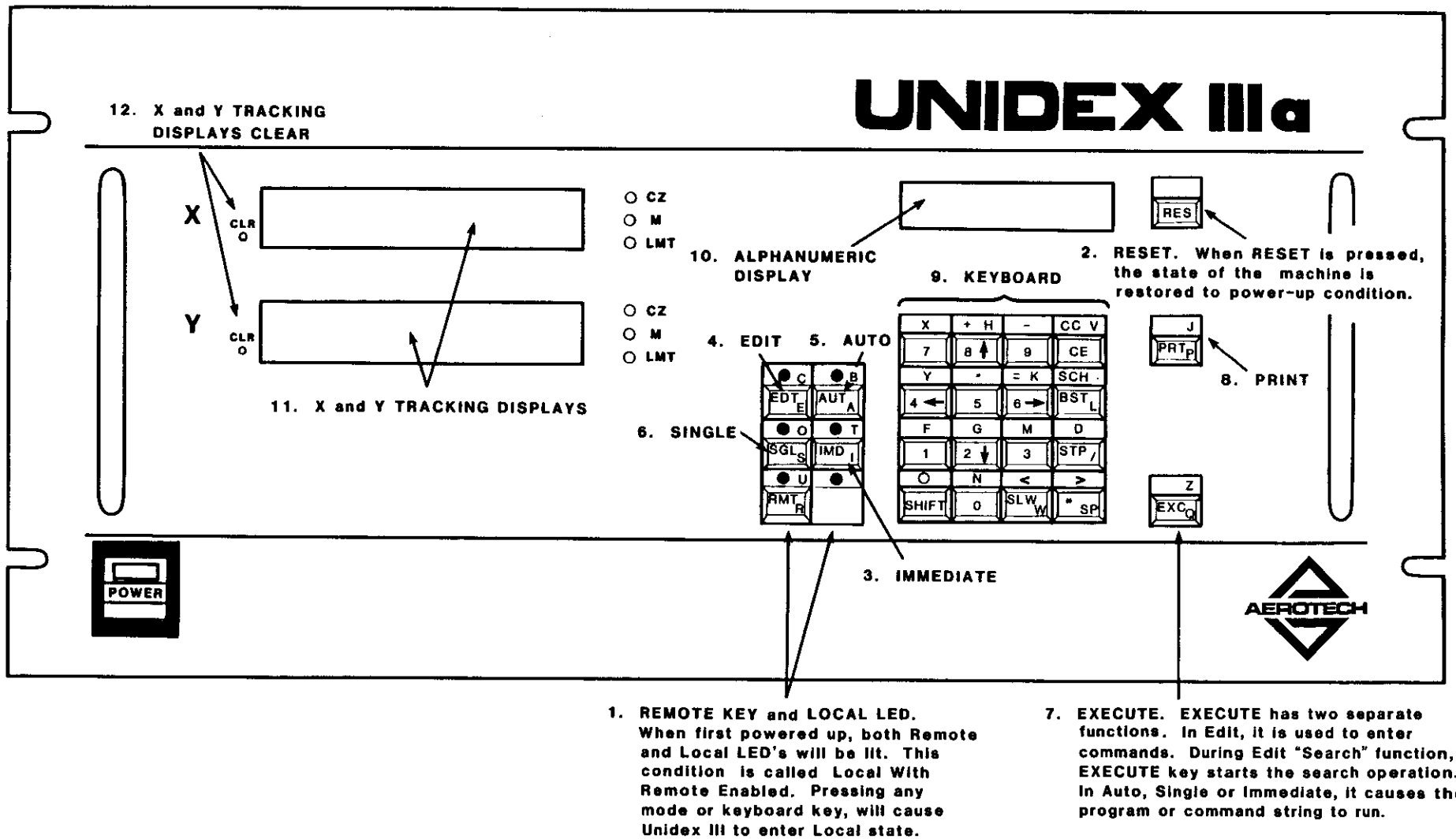


FIGURE 2-1: FRONT PANEL CONTROLS AND DISPLAY

The SHIFT key toggles and is also LED indicated. The remaining front panel LEDs are:

1. X axis (CZ,M,LMT)
2. Y axis (CZ,M,LMT)
3. Local

To explain CZ, M and LMT:

	DC SYSTEM	LO SPEED STEPPER
CZ	Axis in position	Low/high current mode
M	Marker	Marker
LMT	In limit	In limit

Also located on the front panel are:

1. EXECUTE key
2. PRINT key
3. RESET key
4. Eight character alphanumeric display
5. X and Y tracking displays with independent clear buttons

The front panel keys' functions are discussed here and on the front panel illustration (figure 2-1).

A. REMOTE SWITCH AND LOCAL LED

When first powered up, both remote and local LEDs will be lit; this condition is called "local with remote enabled". If you press any keyboard key, the remote LED will go out. Unidex IIIa will be in local mode. You may switch to "local with remote enabled" state again by pressing the REMOTE key.

B. RESET KEY

When you press RES, the state of the machine is restored to power-up conditions.

C. IMMEDIATE KEY

Pressing IMD causes Unidex IIIa to enter the Immediate mode.

D. EDIT KEY

Pressing EDT causes Unidex IIIa to enter the Edit mode.

E. AUTO KEY

Pressing AUT causes Unidex IIIa to enter the Auto mode.

F. SINGLE KEY

Pressing SGL causes Unidex IIIa to enter the Single mode.

G. EXECUTE KEY

Pressing EXC causes the program to execute. In the Edit mode this means the program's motion commands will be displayed. In the Auto, Single or Immediate mode, this means the motion program or string of motion commands will run, causing the drives to move.

H. PRINT KEY

When properly connected to a printer (via RS-232C/RS-422A DTE connector), this key is used to initiate a print-out. If connected to a controller instead, pressing PNT along with a program number will cause the program to be sent over the lines to the controller.

I. KEYBOARD

The keyboard enables you to enter commands and also enter a message when in the message entry mode.

J. ALPHANUMERIC DISPLAY

As you type in your commands or call up a program stored in memory, the commands will appear on the alphanumeric display.

K. X AND Y TRACKING DISPLAYS

The tracking displays show actual positions of the axes. They are driven by the encoder feedback signals in a DC motor system.

L. X AND Y DISPLAYS CLEAR

These pushbutton switches clear the X and Y displays respectively.

SECTION 2-4 REAR PANEL (INTERFACE BOARD) CONNECTIONS

The rear panel of Unidex IIIa (figure 2-2) has connectors for the following:

- IEEE-488 communications bus connector
- RS-232 interface connector and daisy chain connector
- RS-422 interface connector and daisy chain connector
- Joystick connector - see figure 2-3
- Input/output ($\overline{M}$-outputs, C-inputs and Feedhold Inputs) - see figure 2-3
- Motors/encoders (when ordered complete with drives)

A. REAR PANEL SWITCHES

The rear panel switches include:

1. Device address select switch - allows you to set the address for each Unidex IIIa when multiple devices are daisy-chained together. See figure 2-2. The addresses are defined in table 2-1.

2. Last device switch - when multiple devices are daisy-chained together, this switch is set to indicate which Unidex IIIa is the last device in the chain. See figure 2-4.

3. Serial communication format select switches - the switches which select baud rate, character length, parity and stop bits when using RS-232 or RS-422. See figure 2-4.

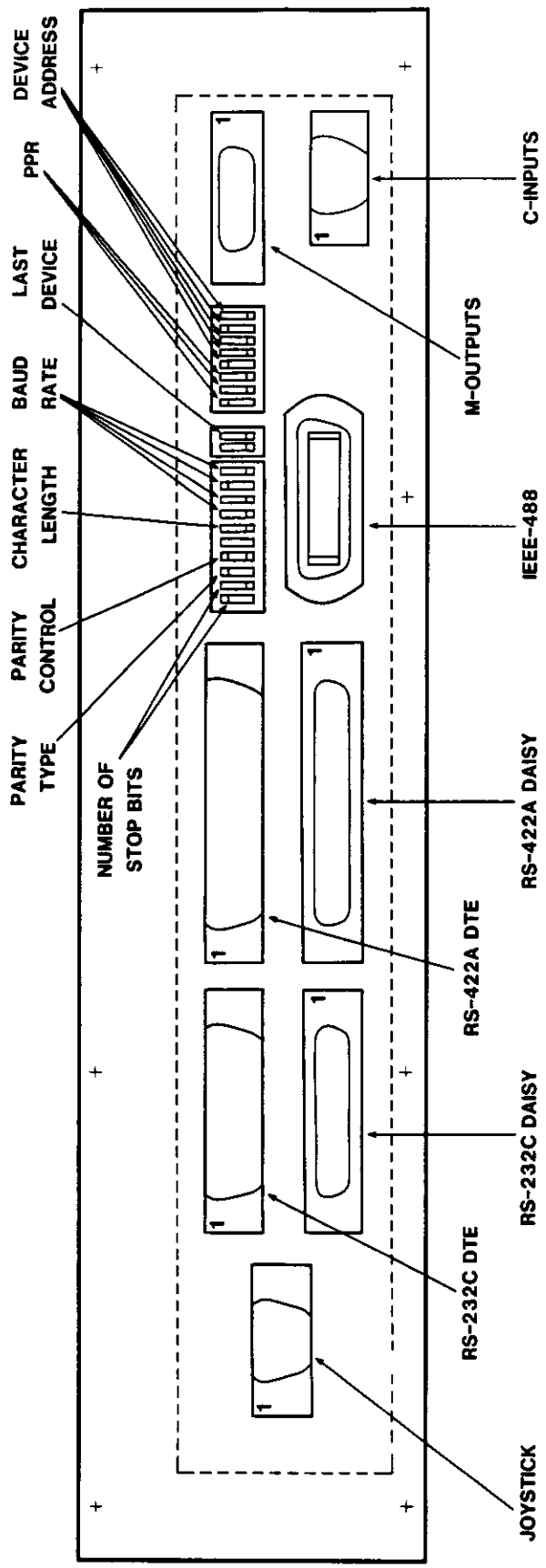


FIGURE 2-2: REAR PANEL

4. Parallel poll response select switches - the switches which select the order of device response for a parallel poll when multiple devices are being interfaced through IEEE-488. See figure 2-5.

NOTE: The methods of connecting multiple devices being interfaced through IEEE-488 are shown in figure 2-6.

NOTE: The method of connecting multiple devices being interfaced through RS-232C/422A is shown in figure 2-7.

SWITCH SETTINGS ON REAR PANEL

When implementing RS-232C/422A, the serial communication format of your Unidex IIIa must match those of your controller.

Baud rate is switch selectable from 45.5 to 38400. Character length is variable from 7 to 8 bits with 1, 2 or 1 1/2 stop bits.

Parity may be selected to be ON or OFF. If on, odd or even parity may be selected. All of these switches are located on the rear panel of the chassis.

Multiple Unidex IIIa devices may be daisy chained, using the serial port. The units connected in a daisy chain fashion may be individually addressed by the controller.

Two RS-232 connectors are provided on the rear panel. You simply connect the controller to the connector marked DTE on the first Unidex IIIa. Then connect the "daisy" connector on this first Unidex IIIa to the DTE connector on the second device of the chain using a one to one cable as in the top illustration of figure 2-7. The "last device" switch on the last device of the chain should be "ON".

If only one Unidex IIIa is on the RS-232/422 line, set it as the last device.

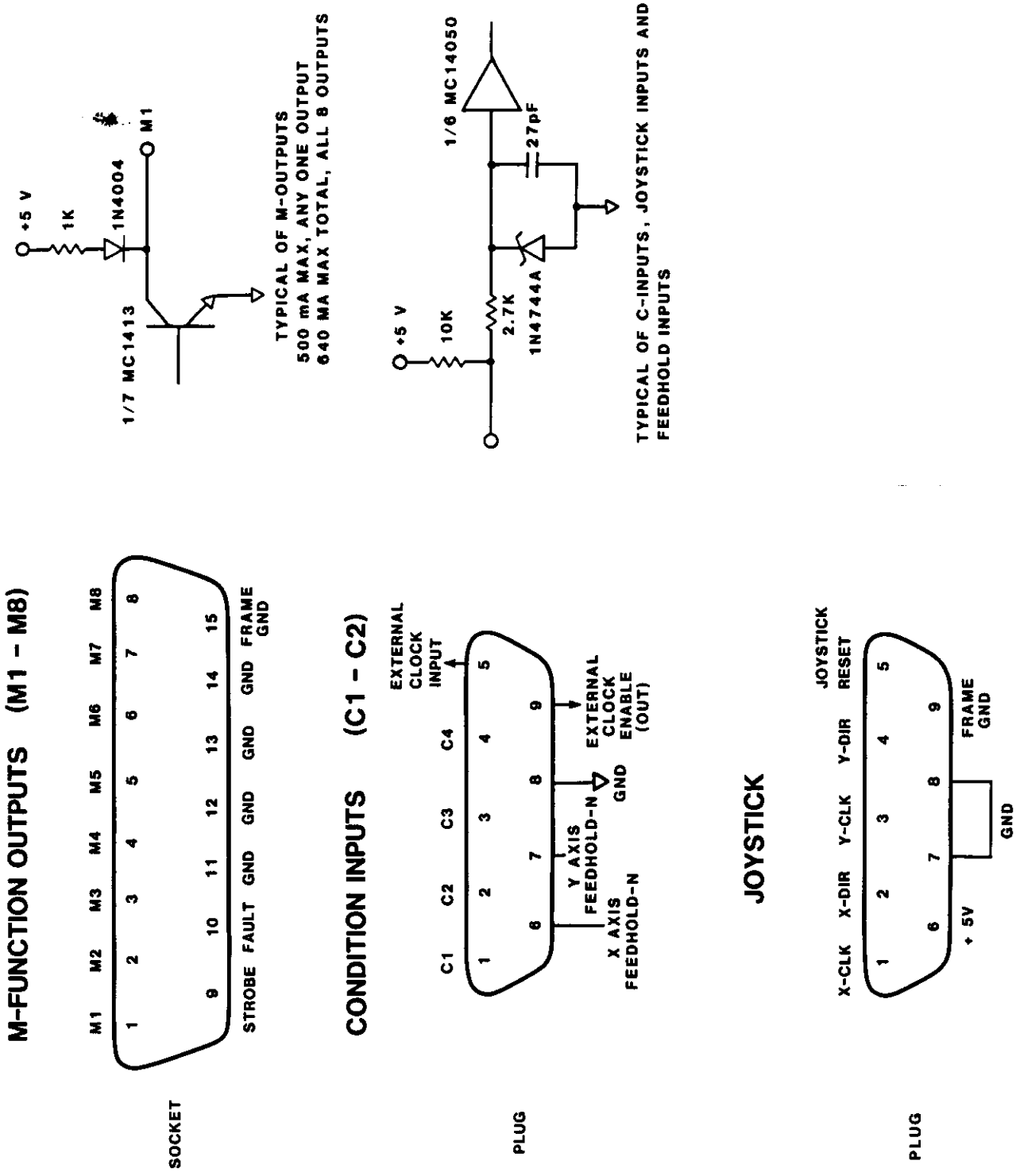


FIGURE 2-3: PIN DEFINITIONS OF M-OUTPUTS, C-INPUTS AND JOYSTICK

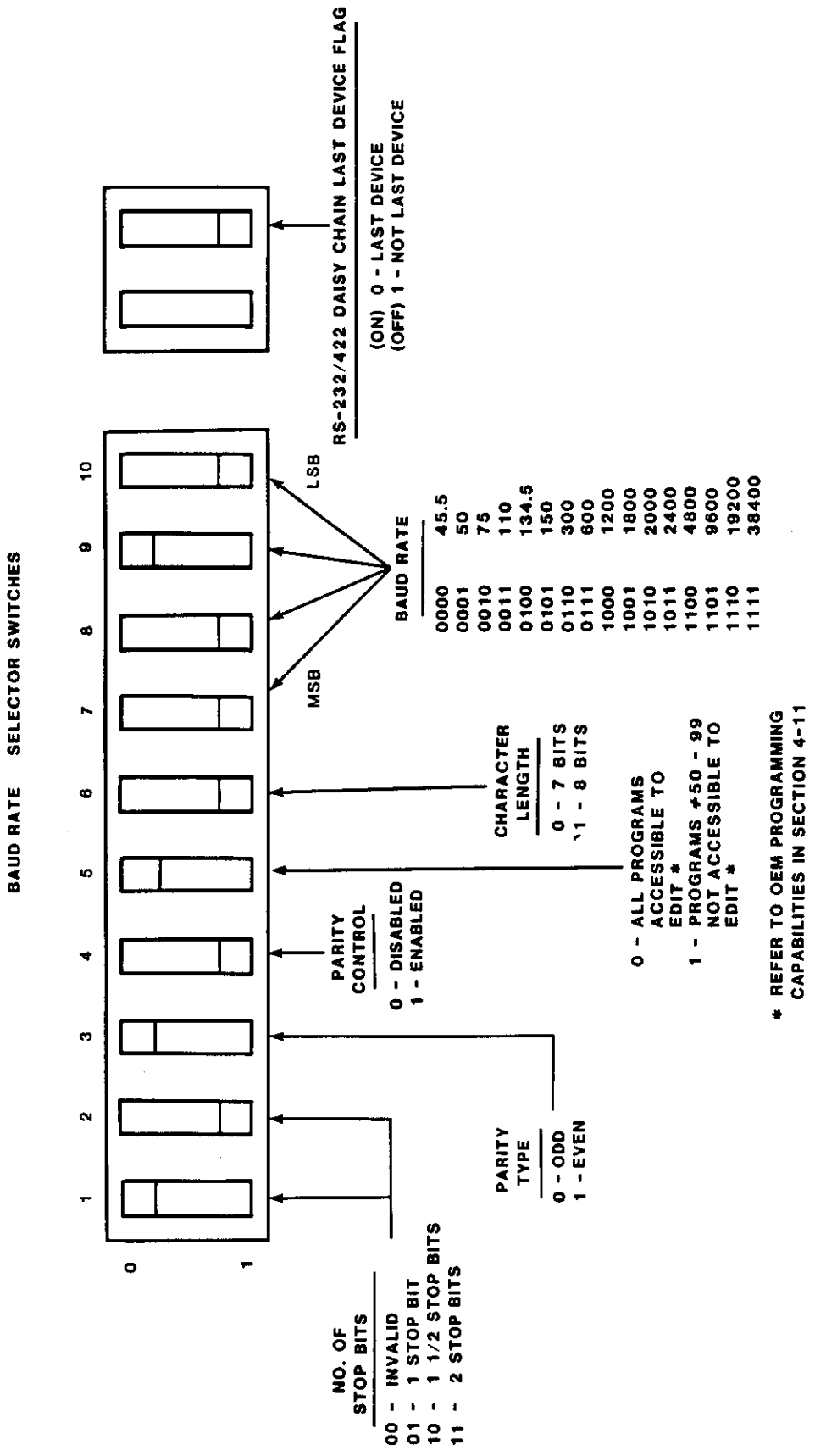


FIGURE 2-4: SELECTOR SWITCHES LOCATED ON REAR PANEL

B. SYSTEM CONFIGURATION

The jumper connections available to you in order to configure your system are illustrated in table 2-2.

SECTION 2-5 JOYSTICK OPTION

Unidex IIIa can interface with any joystick with the following signal specifications. These signals are connected to the Unidex IIIa inputs as shown in figure 2-3.

X - Clock

X - Direction (0 is negative direction, and 1 is positive direction). Standard:
0 = CCW motor rotation
1 = CW motor rotation

Y - Clock

Y - Direction (0 is negative direction, and 1 is positive direction). Standard:
0 = CCW motor rotation
1 = CW motor rotation

Each of these inputs in Unidex IIIa is configured as shown in figure 2-3.

The signal levels on the outputs may be TTL or CMOS compatible.

The joystick button is a normally open switch with one terminal connected to ground (common).

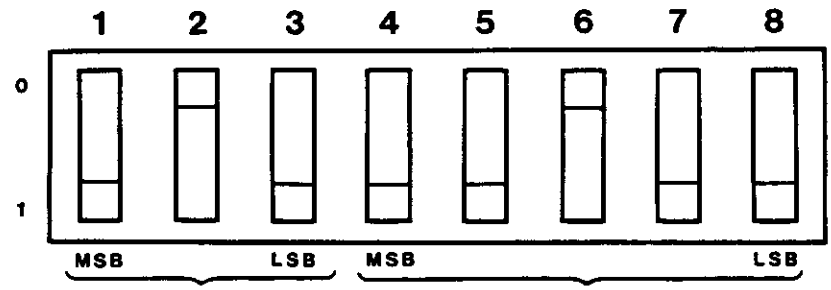
SWITCH 4	5	6	7	8	DECIMAL ADDRESS
OFF	OFF	OFF	OFF	OFF	31*
OFF	OFF	OFF	OFF	ON	30
OFF	OFF	OFF	ON	OFF	29
OFF	OFF	OFF	ON	ON	28
OFF	OFF	ON	OFF	OFF	27
OFF	OFF	ON	OFF	ON	26
OFF	OFF	ON	ON	OFF	25
OFF	OFF	ON	ON	ON	24
OFF	ON	OFF	OFF	OFF	23
OFF	ON	OFF	OFF	ON	22
OFF	ON	OFF	ON	OFF	21
OFF	ON	OFF	ON	ON	20
OFF	ON	ON	OFF	OFF	19
OFF	ON	ON	OFF	ON	18
OFF	ON	ON	ON	OFF	17
OFF	ON	ON	ON	ON	16

SWITCH 4	5	6	7	8	DECIMAL ADDRESS
ON	OFF	OFF	OFF	OFF	15
ON	OFF	OFF	OFF	ON	14
ON	OFF	OFF	ON	OFF	13
ON	OFF	OFF	ON	ON	12
ON	OFF	ON	OFF	OFF	11
ON	OFF	ON	OFF	ON	10
ON	OFF	ON	ON	OFF	9
ON	OFF	ON	ON	ON	8
ON	ON	OFF	OFF	OFF	7
ON	ON	OFF	OFF	ON	6
ON	ON	OFF	ON	OFF	5
ON	ON	OFF	ON	ON	4
ON	ON	ON	OFF	OFF	3
ON	ON	ON	OFF	ON	2
ON	ON	ON	ON	OFF	1
ON	ON	ON	ON	ON	0

* NOT USED

TABLE 2-1: DEVICE ADDRESS SELECT SWITCHES

IEEE - 488 ADDRESS SELECT SWITCHES



PARALLEL POLL CONFIGURE DEFINES PARALLEL POLL RESPONSE

000 - PPR 1
001 - PPR 2
010 - PPR 3
011 - PPR 4
100 - PPR 5
101 - PPR 6
110 - PPR 7
111 - PPR 8

IEEE-488 BUS ADDRESS VALID ADDRESSES

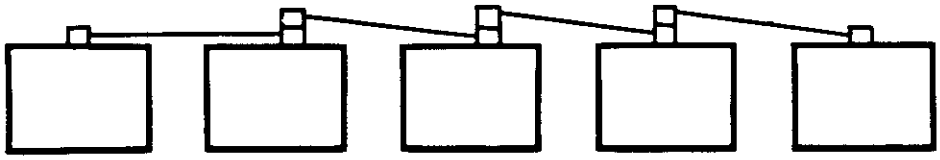
0 DECIMAL TO 30 DECIMAL
00000 TO 11110

IEEE - 488 PARALLEL POLL RESPONSE

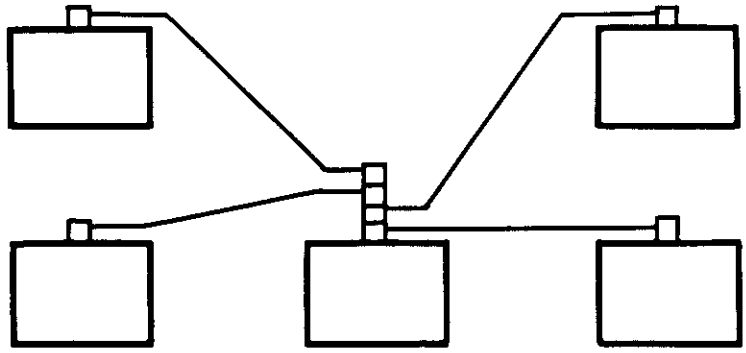
(SIGNAL ON THE DATA LINES)

	8	7	6	5	4	3	2	1
PPR 1	0	0	0	0	0	0	0	1
PPR 2	0	0	0	0	0	0	1	0
PPR 3	0	0	0	0	0	1	0	0
PPR 4	0	0	0	0	1	0	0	0
PPR 5	0	0	0	1	0	0	0	0
PPR 6	0	0	1	0	0	0	0	0
PPR 7	0	1	0	0	0	0	0	0
PPR 8	1	0	0	0	0	0	0	0

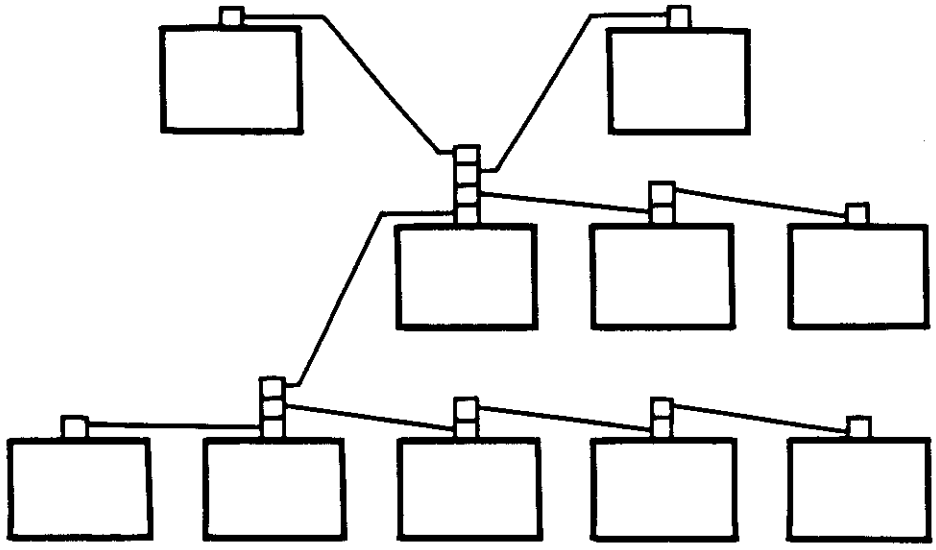
FIGURE 2-5: IEEE-488 ADDRESS SELECT SWITCHES LOCATED ON REAR PANEL



LINEAR



STAR



MIXED

FIGURE 2-6: IEEE-488 CABLING CONFIGURATIONS

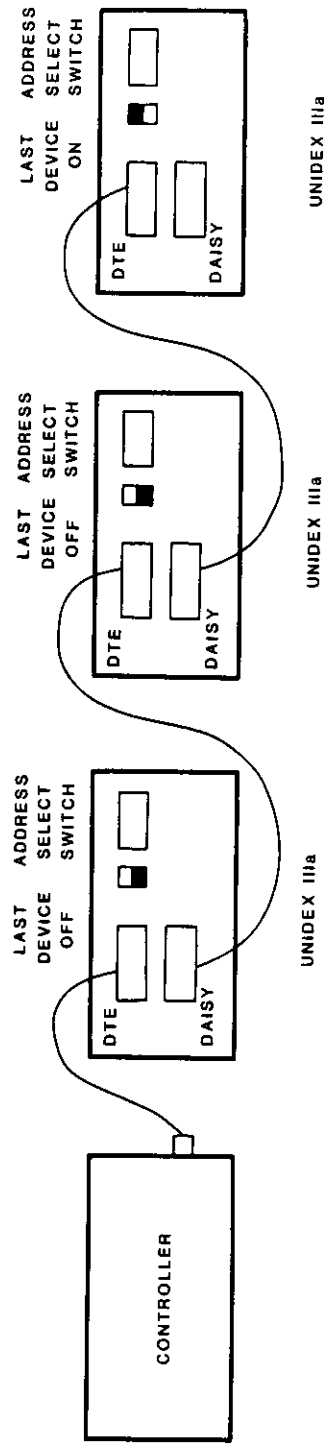


FIGURE 2-7: RS-232C/422A DAISY CHAIN CONNECTIONS

	JP 9	JP10	JP11	JP12	JP13	JP14	JP15	JP16	JP24	JP25	JP26	JP27
DC HARDWARE HOME	2-3	IN	OUT	2-3	2-3	IN	OUT	2-3	OUT	IN	IN	IN
SM. LO SPD. HARDWARE HOME	1-2	IN	OUT	2-3	1-2	IN	OUT	2-3	IN	OUT	IN	IN
SM. HI SPD. HARDWARE HOME	2-3	IN	OUT	2-3	2-3	IN	OUT	2-3	OUT	OUT	IN	IN
SM. LO SPD. SOFTHOME X-CW;Y-CW	1-2	OUT	IN	1-2	1-2	OUT	IN	1-2	IN	IN	IN	OUT
" " " X-CW;Y-CCW	1-2	OUT	IN	1-2	1-2	OUT	IN	1-2	OUT	IN	IN	OUT
" " " X-CCW;Y-CW	1-2	OUT	IN	1-2	1-2	OUT	IN	1-2	IN	OUT	IN	OUT
" " " X-CCW;Y-CCW	1-2	OUT	IN	1-2	1-2	OUT	IN	1-2	OUT	OUT	IN	OUT

TABLE 2-2: SYSTEM CONFIGURATION

DIRECTION JUMPERS
STANDARD DIRECTION
DIRECTION -N

(X AXIS) JP5	(Y AXIS) JP7
IN	IN
OUT	OUT

NOTE: Direction N reverses axis direction as well as home direction. (Home direction reversed on systems with software home only.)

VCC TO DRIVE CONNECTORS
(STANDARD) VCC TO PIN 1
NO CONNECTION

JP6	JP8
J4(X)	J5(Y)
IN	IN
OUT	OUT

"C" INPUTS DEBOUNCE/BYPASS
ASSOCIATED INPUTS
(STANDARD) DEBOUNCE ACTIVE
DEBOUNCE BYPASS

JP20	JP21	JP22	JP23
C1	C2	C3	C4
1-2	1-2	1-2	1-2
2-3	2-3	2-3	2-3

LIMIT
(STANDARD)

JP37
IN
OUT

GO TO LOCAL MODE
REMAIN IN PRESENT MODE

KEYBOARD ADDRESS
ADD JP45, REMOVE JP38 - 44

CLOCK PULSE WIDTH
CONTROL JUMPERS

(STD) DC
SM

WIDTH	JP47 X AXIS	JP46 Y AXIS
1 uS	0-1	0-1
2 uS	0-2	0-2
3 uS	0-3	0-3
4 uS	0-4	0-4

TABLE 2-3: MISCELLANEOUS JUMPERS

SECTION 2-6 POWERING UP

Install all motor connectors and cables before applying power to the system. If the system includes an Aerotech stage, refer to the stage manual. If the system does not include a stage, the drives will have to be adjusted for proper response to the load inertia and friction.

A. A.C. POWER REQUIREMENTS

The standard system requires 115/230 VAC ±10% at 50 to 60 Hz. Unidex IIIa itself requires 1.8 amps at 115 VAC in a typical system. The chassis will be fused at 5 amps.

B. SYSTEM POWER-UP

On power-up, Unidex IIIa will reset both drives, initialize itself, and perform a self-test.

The system initialization at power-up also checks the user memory for any alterations since the previous Edit session. A validity test is first carried out by verifying a prestored word (16 bytes long). Next, a checksum verification is done on the entire available user memory. The checksum is stored in the first two bytes of the memory and the prestored word in addresses 3 through 18.

The self testing during power-up is done in the following order:

1. EPROMs

Checksum verification. A failure will cause the message "ROM ERR" to be displayed by the alphanumeric display. The operating system does not proceed further.

2. System RAM

Write and read testing. The information in the RAM is not destroyed during this test. Failure causes message "SRAM ERR" to be displayed.

3. User Memory

a. Verification of prestored word. Failure causes message "URAM ALT". (Means user RAM altered.) This message is expected when user memory RAM chips have been powered down for any reason such as installation of new RAM chips or a battery failure.

b. Checksum verification. Failure may be due to errors written into the user memory due to electrical noise.

c. Power down in edit mode.

If checksum verification fails, Unidex IIIa will check to see if it was previously powered down during the edit mode. If it was, and a program number had been stored, then Unidex IIIa will display PWR FAIL for 2 seconds, then go into the edit mode and display the program to be edited.

If the user memory was cleared prior to powering down Unidex IIIa, the message "URAM CLR" is displayed at power-up if no failure occurs in the above three tests.

If however, one of the programs in the user memory has been selected as a boot-strap program to run at power-up, Unidex IIIa will execute the appropriate program if the self diagnostic tests are completed without failure.

If alternately, a program (say 47) has been selected as a set-up program, the display will show "RUN - 47" and pressing the [EXC] key will start the execution of the program.

If there are no failures in the self testing, the user memory is not clear and if there are no boot-strap or set-up programs selected, the message "SYS RDY" is displayed.

C. FRONT PANEL RESET

Front Panel Reset performs self-diagnostic tests as above. However, the front panel reset does not cause the boot-strap program to run.

The resetting of Unidex IIIa is disabled when editing a program in the user memory. If the [RES] key is pressed when editing, the message "EDITING!" is displayed. (Pressing the [EXC] key will bring back the prior display.)

In order to reset, you must exit the edit mode.

D. DEFAULT STATES AT POWER-UP

The default states that Unidex IIIa powers - up in are as listed below:

- Local with remote enabled
- Incremental mode (G91 - alternates with G90)
- Non-corner rounding mode (G24 - alternates with G23)
- Independent feedrate mode (G00 - alternates with G01)
- Normal move mode (G49 - alternates with G47 and G48)
- No accel/decel (G36 - alternates with G37, G38 and G39)

E. CLEARING MEMORY

When powering up for the first time or if the user memory has lost power for any reason, it is essential to clear the user memory and revalidate memory start and end pointers that are stored in the user memory itself. The following sequence accomplishes this:

[EDT]	PGM <1>
[CC]	CLR <1>
[0]	CLR 0
[0]	CLR 00
[EXC]	CLEARED

F. SELECTING A BOOT-STRAP PROGRAM

A program to be run as a boot-strap program at power-up is selected using the following key sequence.

Power-up		
[EDT]	PGM <1>	
[SHIFT] [F]	BOOT 37	Program #37 is currently the boot-strap program
[2]	BOOT 72	
[5]	BOOT 25	
[EXC]	BOOT 25	EDIT LED is turned off. Program #25 is now the boot-strap program
Power-down		
Power-up	Executes program #25	

G. SELECTING A SET-UP PROGRAM

The key sequence is described below:

End of program execution	READY__	
[EDT]	EDIT 41	Program #41 has just been executed
[SHIFT] [G]	SETUP 79	Current set-up program
[1] [9]	SETUP 19	
[EXC]	SETUP 19	EDIT LED turns off. Program #19 is now the Set-up program. Boot-strap program is cancelled.
[RES]	RUN 19	
[EXC]	Executes program #19	

Boot-strap and set-up programs are mutually exclusive. Unidex IIIa may be programmed to either run a boot-strap program or a set-up program but not both simultaneously.

H. DESELECTING BOOT-STRAP AND SET-UP PROGRAMS

The boot-strap and set-up programs may be deselected with the following key sequence.

[EDT]	EDIT 76
[SHIFT] [M]	NOBOOT__

[EXC]	NOBOOT__	EDIT LED turns off. Programs #25 and #19 are deselected
[RES]	SYS__RDY	
Power-down		
Power-up	SYS__RDY	

I. DISPLAYING AMOUNT OF FREE MEMORY

It is possible to display the number of free bytes available in the user memory. Use the following key sequence:

Key Sequence	Alphanumeric Display
[EDT] [SHIFT] [D]	FREE MEM
[EXC]	02756 BT

J. USER RAM READ/WRITE TEST

In order to test the User RAM for READ/WRITE operations, the following key sequence is used:

Key Sequence	Alphanumeric Display
[EDT] [SHIFT] [X]	TST URAM
[EXC]	TESTING_
After testing is done....	MEM= 4K

The amount of memory as configured by the PAL (IC no. M3) is displayed for verification.

If any error is detected, the testing will be aborted and the display will read URAM ERR.

SECTION 2-7 COMMUNICATING WITH UNIDEX IIIA

There are four ways communicate with Unidex IIIa (see figure 2-8). They are:

1. Locally, via the front panel keyboard and display
2. Via IEEE-488 communication bus
3. Via RS-232C serial line interface
4. Via RS-422A serial line interface

Any of the three interfaces (IEEE-488, RS-232C and RS-422A) transfer control signals and data signals between the controller and Unidex IIIa(s).

A. LOCAL MODE

In some cases, it may be more efficient and economical to use Unidex IIIa without a controller. Unidex IIIa possesses edit capabilities that allow you to enter and edit short programs efficiently. There is, additionally, a PRINT key that allows you, after connecting Unidex IIIa to a printer, to print out programs from the user memory.

B. IEEE-488 INTERFACE

IEEE-488 has 8 data lines and 8 control lines. It can accommodate up to 14 devices and provides a service request line from all devices to the controller. All of these properties lead to a more rapid form of communication between Unidex IIIa and the controller. You need not concern yourself with bus disciplines if your controller has IEEE-488 interface and device driver software that "hooks" into the Basic, Fortran, Pascal or whatever language you intend to use.

Different ways of connecting multiple devices to the IEEE-488 interface is shown in figure 2-6.

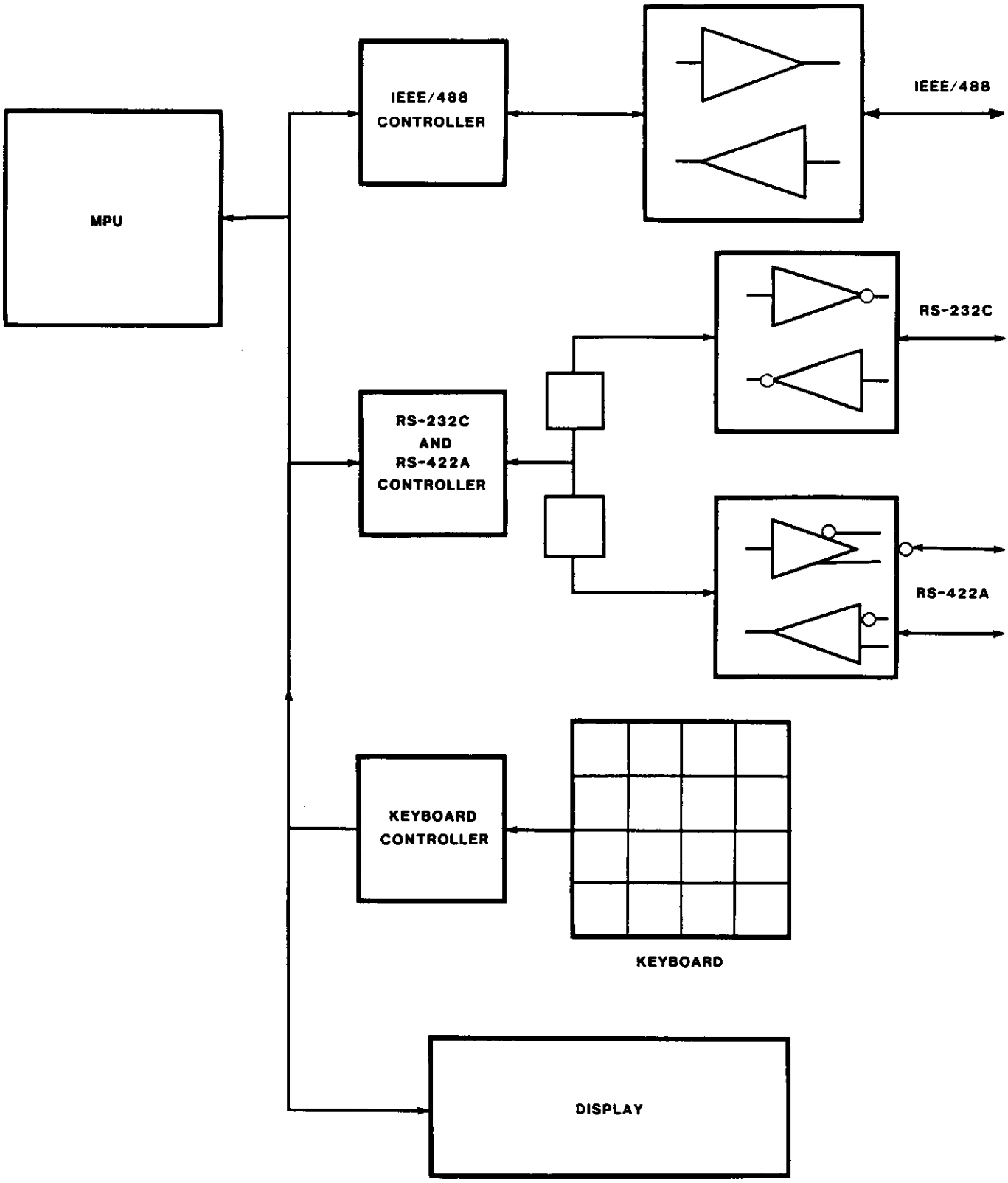


FIGURE 2-8: UNIDEX IIIA LOCAL AND REMOTE MODES OF COMMUNICATION

The devices connected to the bus have certain roles assigned to them. The roles represent the three basic functional elements necessary for effective communication. These three roles are:

1. Listener
2. Talker
3. Controller

1. Device As Listener

A "listener" is a device that has the capability of receiving data from the bus. It can be addressed by an interface message to listen. When addressed to listen, the listener will receive data placed on the bus.

2. Device As Talker

A "talker" is a device with the capability of sending data via the bus, when addressed by an interface message to talk.

3. Device As Controller

A "controller" is a device with the capability of controlling and directing the activity on the bus. A controller can address other devices to listen or to talk. It can also send interface messages to command specific actions from the other devices connected to the bus. You will need a device to act as a controller when implementing the IEEE-488 interface.

Listener, talker and controller capabilities can occur individually or in combinations. For instance, devices such as

the Unidex IIIa or a terminal can be implemented to talk or to listen, but not to control. Many computers, however, are capable of talking, listening and controlling.

Refer again to section 2-4 for all of the possible address combinations for Unidex IIIa.

4. Signal Lines of the IEEE-488 Bus

The IEEE-488 transfers data and commands between devices on 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 for bus management. Each line, when asserted low (ground), represents a single line message sent on the bus. A description of these lines is given in table 2-4.

5. Cable Restrictions of the IEEE-488 Bus

The devices in a system are connected together by a 24-wire cable using 24-pin connectors as specified in the IEEE-488 standard.

There are certain limitations on the length of the cables and the number of devices on the bus.

The maximum number of devices on the bus is limited to 14. The total length of the cable is limited to 20 meters or 2 meters multiplied by the number of devices (whichever is shorter in length). For a complete cable listing, refer to table 2-5.

6. Parallel And Serial Polling

Parallel polling is done to identify which device on the IEEE-488 bus is requesting service (SRQ). Serial polling is then done on the device requesting service in order to determine why.

a. Parallel Polling

In the parallel poll, you may assign each of the eight data lines of the bus to as many as eight different devices.

The parallel poll bit assigned to each Unidex IIIa may be selected via the dipswitch labeled PPR, located on the back panel. Address selection may be between 0 and 7 in binary numbers (figure 2-5).

b. Serial Polling

In the serial poll, each of the devices requesting service is polled one at a time. You may serial poll any device at any time, regardless of the number of devices on the line.

A Unidex IIIa 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 IIIa is serial polled by the controller. Upon being polled, the Unidex IIIa will transmit its status.

MNEMONIC	NAME	FUNCTION
		Bus Management Lines
IFC	INTERFACE CLEAR	System controller alone can assert this line, to place all devices in the unaddressed state. Devices go into talker idle/listener idle state. If control has been passed to another device, system controller again becomes active by asserting IFC.
ATN	ATTENTION	Asserted true by active controller to send bus interface messages on the bus. When ATN is asserted, signals on the data lines are interpreted as messages. ATN asserted with EOI to do a parallel poll. When ATN is false, data may be sent over the bus by a designated talker.
REN	REMOTE ENABLE	Asserted to program devices on the bus remotely. Any device addressed to listen while REN is true, is placed in remote mode of operation.
SRQ	SERVICE REQUEST	Asserted by a device to indicate REQUEST its need for interaction with the controller.
EOI	END OR IDENTIFY	When asserted, indicates the termination of flow of data. Asserted when the last data byte is placed on the bus.
		DATA HANDSHAKE LINES
DAV	DATA VALID	Asserted by the talker to indicate to all listeners that data on the bus is valid.
NRFD	NOT READY FOR DATA	When true, indicates to talker that all listeners are not ready for data.
NDAC	NOT DATA ACCEPTED	When true, indicates to the talker that all listeners have not accepted the data placed on the bus
		DATA LINES
DI01-DI08	DATA LINES	Used for sending data (ATN lines false) or bus interface messages (ATN line true).

TABLE 2-4: IEEE-488 STANDARD INTERFACE BUS SIGNAL LINE

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 Meters

TABLE 2-5: IEEE-488 CABLE MANUFACTURERS

C. RS-232C INTERFACE

RS-232C has 2 data lines and 20 additional lines for control. The control lines implememted in Unidex IIIa are:

1. DSR (data set ready, CC)
2. DTR (data terminal ready, CD)

The RTS signal line is used to implement service request by Unidex IIIa and is not used as a part of communication protocol. Therefore, it is essential that the controller be able to recognize a change in RTS as a service request by Unidex IIIa. If the host cannot recognize this, the service request can be configured as a data byte which Unidex IIIa sends over the data line (software service request).

The interconnection needed to implement the controller configured as a DCE or a DTE with and without RTS as a service request is shown in figure 2-9.

D. RS-422A INTERFACE

Unidex IIIa has connectors on the rear panel of the chassis for any of the communication interfaces, ie., the IEEE-488, RS-232C and RS-422A. In addition, RS-232C and RS-422A have a second connector for daisy chaining another Unidex IIIa, as shown in figure 2-7.

Unidex IIIa is factory-set to use RS-232C and IEEE-488. To use RS-422A instead of RS-232C, jumper plug on socket M78 on the MPU board must be removed and inserted into socket M77, before plugging into the RS-422A connector.

RS-422A implements the same serial line communication protocol as RS-232C with differential line drivers. See table 2-6.

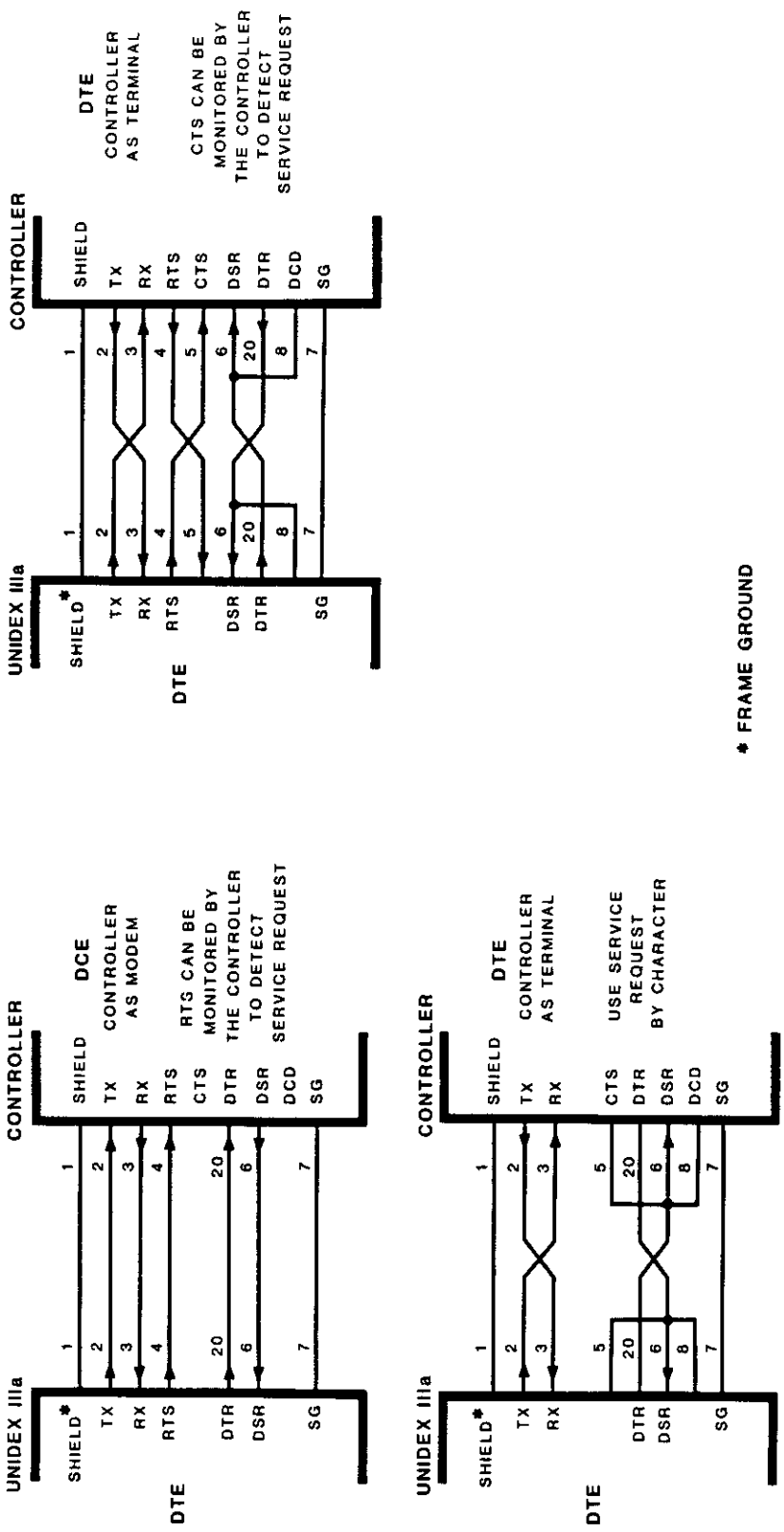


FIGURE 2-9: RS-232C CONNECTIONS

FIRST SEGMENT ASSIGNMENT			SECOND SEGMENT ASSIGNMENT			CIRCUIT CATEGORY	DIRECTION		RS-232C EQUIVALENT
CONTACT NUMBER	CIRCUIT	INTERCHANGE POINTS	CONTACT NUMBER	CIRCUIT	INTERCHANGE POINTS		TO DCE	FROM DCE	
1	SHIELD *	-							PROTECTIVE GROUND (AA)
2			20						
3			21						
4	SD	A-A'	22	SD	B-B'	I	X		TX (BA)
5			23						
6	RD	A-A'	24	RD	B-B'	I		X	RX (BB)
7	RS	A-A'	25	RS	B-B'	I	X		RTS (CA)
8			26			I			
9	CS	A-A'	27	CS	B-B'	I		X	CTS (CB)
10			28						
11	DM	A-A'	29	DM	B-B'	I		X	DSR (CC)
12	TR	A-A'	30	TR	B-B'	I	X		DTR (CD)
13	RR	A-A'	31	RR	B-B'	I		X	DCD (CF)
14			32						
15			33						
16			34						
17			35			I	X		
18			36						
19	SG	C-C'	37	SC	C-B'	I	X		SG (AB)

* FRAME GROUND

NOTE: Blank entries in table above are not used

TABLE 2-6: RS-422A CONNECTIONS

E. END-OF-FILE (EOF) CHARACTER, PROGRAMMABLE

When printing programs, directory or answering a "Q", Unidex 3 sends a terminating character. This terminator is the ASCII code (03) by default.

Version 3Q software allows the user to program a custom character to be used as the terminator. Programming the EOF character may be done via the keyboard or the remote interface.

KEYBOARD MODE:
Type in key sequence [EDT][SHFT][N].
Display shows: MOD. EOF
Type in: [EXC]
Display shows: ENTER >>
Type in the decimal ASCII for the new EOF: [1][2][6]
Followed by [EXC]
Display shows: MODIFIED

The terminator [End of File] character is now "~"

REMOTE MODE:
Send the string "E N"
Display shows: MOD. EOF
Send the string: <CR><LF>
Display shows: ENTER >>
Send the string "1 2 6 <CR><LF>"
Display shows: REMOTE

The modified EOF character is stored in the battery backed user memory and need not be reprogrammed unless altered due to memory fault, as indicated by the displayed message "URAM ALT".

NOTE: When installing version 3Q in systems with 3P or earlier versions, the EOF MUST BE PROGRAMMED TO <03>.

Whenever Unidex 3 displays "URAM ALT", the default value (03) has been automatically programmed.

NOTE: A Communication Software Package with Circular Interpolation Application Software for Unidex IIIa on an IBM PC is available. For more information on the availability, delivery and cost of this software, please contact:

McMAHAN ELECTRO-OPTICS, INC.
2160 Park Avenue, North
Winter Park, Florida 32789

Telephone: (305) 645-0463

CHAPTER 3: GETTING ACQUAINTED WITH UNIDEX IIIA

The purpose of this chapter is to give you a general view of how to operate Unidex IIIa.

SECTION 3-1 TYPES OF COMMANDS

There are two types of commands involved in programming the Unidex IIIa: System commands and Motion commands.

System commands are those which allow you to enter your program into Unidex IIIa's memory and execute it in a specific manner. The system commands are covered in chapter 4, System Programming.

Motion commands are those which control the action of the axes and make up your actual motion program. These commands include the X, Y, F, M, N, D, and G codewords. The motion commands are covered in chapter 5, Motion Programming.

SECTION 3-2 TYPES OF COMMUNICATION

There are two ways to communicate with your Unidex IIIa: Local programming and remote programming.

Entering commands by the front panel keyboard is local programming.

Entering commands by a controller (computer) via the RS-232C, RS-422A or IEEE-488 connectors is remote programming. See appendix 1 for command characters and the ASCII code table. Appendix 6 contains a sample program which demonstrates how to remotely program Unidex IIIa via an IBM® PC.

SECTION 3-3 MODES OF OPERATION

The various modes in which Unidex IIIa can operate are accessible to you in either the local or remote mode. The modes or operation are:

1. Edit mode
2. Auto mode
3. Single mode
4. Immediate mode
5. Remote enable mode

A. EDIT MODE

To enter or modify (edit) a motion program, you must be in the edit mode. The editing of your program is made possible by the following functions:

1. EDIT - Allows you to enter a new program or call up an existing one.

2. STEP/BACKSTEP - Allows you to step through a program, in either a forward or a backward direction.

3. SEARCH - Allows you to locate a certain command or string of characters within your program.

4. CLEAR ENTRY - Allows you to erase an entry within your program.

5. CLEAR COMMAND - Allows you to erase a command within your program.

6. SLEW - Brings you to the start of the program.

B. AUTO MODE

When in the auto mode, you may run a program from beginning to end.

C. SINGLE MODE

When in the single mode, you may execute a program one block at a time. A block is an entire string of commands that ends with the EOB (*) symbol in memory. Just press EXECUTE to run each block of commands. Unidex IIIa will display the message "PGM END" (program end) when the program is finished.

D. IMMEDIATE MODE

The immediate mode allows you to enter a command block and run it immediately without storing it in the user memory. (An immediate mode command block must be less than or equal to 256 bytes.)

E. REMOTE ENABLE MODE

In this mode, either the remote interface or the keyboard may communicate with Unidex IIIa.

Motion statuses (X/Y values, register values, M functions, etc.) remain unchanged when going from remote to local or from local to remote.

SECTION 3-4 LOCAL PROGRAMMING

Upon powering up, the front panel keyboard display will show SYS RDY (system ready), and both the local and remote LED'S will be lit. This is the "local with remote enabled" state. If at this point any key on the front panel is pressed, other than REMOTE, the remote LED will be extinguished and Unidex IIIa will be in the local programming mode.

CAUTION: MAKE CERTAIN THE DRIVES ARE IN A SAFE POSITION BEFORE BEGINNING A PROGRAM.

A. EDITING

1. Entering a New Program

To enter a new program, program 25 for instance, the following sequence of mode commands are entered.

[EDT]	PGM <1> (program 1 is default program)
[2]	EDIT 02
[5]	EDIT 25
[EXC]	>>

Program 1, the default program, appears upon power-up. Program 25 is requested, EXECUTE is pressed, and Unidex IIIa is now ready to receive its new commands.

Your motion commands may now be entered. For example:

[SHIFT]	[G]	>>G
	[7]	>>G7 (axes go home)
[SHIFT]	[X]	>>G7X
	[1]	>>G7X1
	[0]	>>G7X10
	[0]	>>G7X100 (X axis, 100 steps)
[SHIFT]	[F]	>G7X100F
	[1]	G7X100F1
	[0]	7X100F10

```
                .                .  (program
                                    continues)
                .                .

[SHIFT]   [M]        Y100F10M

          [2]        100F10M2    (end of
                                  program)
```

2. Changing an Existing Program

The edit functions help speed up the editing process. How to use these functions when altering a program is explained in the following paragraphs.

a. STEP and BACKSTEP

When you need to examine your program one command at a time, you will need the STEP/BACKSTEP function. This allows you to step through the program either in a forward or backward direction.

Since these functions wrap around, pressing STEP or BACKSTEP will eventually bring the program back to the starting point. Example:

```
[EDT]        EDIT  25

[EXC]        ____>>G7

[STP]        >>G7X100

[STP]        7X100F10
```

Each time STEP is pressed, the next command is displayed.

Each time BACKSTEP is pressed, the previous command is displayed. Example:

[BST]	>>G7X100
[BST]	>>G7
[BST]	>>
[BST]	100F10M2 (loops around)

b. **SEARCH**

When you must search for a particular command, you will need the SEARCH function. Example:

[EDT]		EDIT 25
[EXC]		>>G7
[SHIFT]	[SCH]	(display goes blank)
[SHIFT]	[Y200]	Y200
[EXC]		LOOKING
		0F10Y200

If Y200 is found, it will be shown on the display. If not, "WHAT?" will be seen on the display instead. For example:

[SHIFT]	[SCH]	
[SHIFT]	[Y250]	Y250
[EXC]		WHAT?

c. **SLEW**

During an edit session, pressing the SLEW key brings you back to the first command from wherever you are in the program. Example:

[SLW]	____>>G7

d. **CLEAR ENTRY and CLEAR COMMAND**

These two edit functions are needed when erasing an entry or a command. A new entry or command may be substituted by simply typing it in after the old one has been erased. Example:

[SHIFT] [SCH]	_____
[SHIFT] [Y250]	____Y250
[EXC]	_LOOKING
	0F10Y250
[CE]	00F10Y25
[CE]	100F10Y2
[0]	00F10Y20
[0]	0F10Y200 (Y250 is changed to Y200)

An example using CLEAR COMMAND would be:

[EXC]		_LOOKING
		0F10Y250
[SHIFT]	[CC]	1X100F10
[SHIFT]	[Y]	X100F10Y
	[2]	100F10Y2
	[0]	00F10Y20
	[0]	0F10Y200

To complete the editing session, press any of the mode keys, ie., EDIT, AUTO, IMMEDIATE, SINGLE or REMOTE. Example:

[AUT]	RUN 25

Unidex IIIa is now in the Auto mode instead of the Edit mode. The same program number is still displayed, however. To enter another program, simply type in the desired number. Example:

[AUT]	RUN 25
[0]	RUN 50
[2]	RUN 02
[EXC]	_RUNNING
	READY_

3. <RESET> During Edit

If <RESET> is pressed when a program is being edited, Unidex IIIa will end the program in the appropriate fashion, and will then reset. It will power up again and display the message "SYS RDY".

If there is a compiler error, the system will display "C-ERR nn". You have two choices at this point. You may edit program "nn" to correct the error, or reset the system by pressing <RESET>.

B. EXECUTING A PROGRAM (AUTO MODE)

To enter the auto mode, where the entire program runs at once, press AUTO. Enter the desired program number and then press EXECUTE. Example:

Power up	SYS RDY
[AUT]	PGM <1>
[2]	RUN 02
[EXC]	RUNNING
	READY

To run program 2 again, just press the EXECUTE key. In order to run another program instead, press the AUTO key. You will then be able to type in the new number.

C. EXECUTING A PROGRAM (SINGLE MODE)

Press SINGLE to enter the Single mode, where the program will be executed one block at a time. Example:

[SGL]	WALK 02 (from previous example)
[3]	WALK 23
[0]	WALK 30
[EXC]	RUNNING
	READY

Unidex IIIa is now ready to run the next block. If the end of program has been reached, "PGM END" will be displayed.

D. IMMEDIATE MODE

In order to enter the Immediate mode, where short programs are entered and executed immediately, simply press IMMEDIATE. Example:

[IMD]		ENTER >>
[SHIFT]	[G]	>>G
	[7]	>>G7
[SHIFT]	[X]	>>G7X
.		.
.		.
.		.
[EXC]		RUNNING
		READY (program complete)

Everytime [EXC] is pressed, these commands will run again. To enter the next string of commands, you don't have to press [IMD] again. You simply type in the commands and press [EXC]. The new set of commands will run. It is not retained in memory, however. Once any of the mode keys is pressed, including [IMD], the program is lost.

The program you enter in the immediate mode is stored in a special 256 byte buffer. This buffer is not to be exceeded.

E. PRINTING A PROGRAM

Programs in the user memory may be printed out via the RS232/422 serial port, using the PRINT key. Example:

Power up	SYS RDY
[PRT]	PGM <1>
[2]	PRNT 02
[0]	PRNT 20
[EXC]	PRINTING
	READY

The above sequence makes the assumption that the printer has been correctly interfaced to Unidex IIIa. If program 20 cannot be located in memory, "WHAT?" will be displayed.

```
          ;UNIDEX-III PROGRAM LISTING

3
E25
          ;PROGRAM # 25   LENGTH: 00080 BYTES

          G47 X1=1000 Y1=2000 G49 *
          "<X1/8>" D2000 "<Y1/8>" D2000 *
          G47 F1= X1 F2= Y1 G49 X1000 Y2000 *

%

L

          ;UNIDEX-III PROGRAM LISTING

3
E20
          ;PROGRAM # 20   LENGTH: 00054 BYTES

          G7 G91 G23 *
          X1000 F1000 Y2536 F1500 *
          M=201 D2000 *
          G90 X0 Y0 *

%

L
```

When printing a program, Unidex IIIa inserts certain leading and trailing character strings with each program being printed. They provide a means of printing (uploading) a program via the RS232/RS422 (DTE) port to an off-line storage device such as a cassette tape drive or a paper tape punch and reading it back (downloading) to Unidex IIIa without any additional software on the part of the device (see section 4-7 A for details).

F. PRINTING THE DIRECTORY

Printing the directory (a listing of all programs in memory) is accomplished as follows:

Power up	SYS RDY
[PRT]	PGM <1>
[D]	PRNT DIR
[EXC]	PRINTING
	READY

```
;UNIDEX-III DIRECTORY LISTING

;PROGRAM # 01    LENGTH: 00164 BYTES
;PROGRAM # 05    LENGTH: 00051 BYTES
;FREE: 03849 BYTES
```

G. PRINTING POSITIONS (X/Y)

To print the positions of X or Y, enter [PRT] [X] or [PRT] [Y]. Example:

Power up	SYS RDY
[PRT]	PGM <1>
[X]	PRNT X
[EXC]	PRINTING
	READY

```
X:AXIS =  0000000000
Y:AXIS = -0000052146
```

H. PRINTING MEMORY

To print the entire contents of the user memory, enter [PRT] [0] [0]. Example:

Power up	SYS RDY
[PRT]	PGM <1>
[0]	PRNT 00
[EXC]	PRINTING
	READY

```
          ;UNIDEX-III PROGRAM LISTING

3
E20
          ;PROGRAM # 20   LENGTH: 00054 BYTES

          G7 G91 G23 *
          X1000 F1000 Y2536 F1500 *
          M=201 D2000 *
          G90 X0 Y0 *

&

E25
          ;PROGRAM # 25   LENGTH: 00080 BYTES

          G47 X1=1000 Y1=2000 G49 *
          "<X1/8>" D2000 "<Y1/8>" D2000 *
          G47 F1= X1 F2= Y1 G49 X1000 Y2000 *

%

L

          ;FREE: 03930 BYTES
```

When printing the entire memory, leading and trailing strings are inserted for each program. Each program is followed by a "&", except the last one, which is followed by a "%".

It is possible to print either one program or the entire Unidex IIIa user memory to an off-line storage device and read it back to Unidex IIIa by simply powering up both devices and sending the stored data to Unidex IIIa over the RS232/RS422 link (see section 4-7 G for details).

I. PRINTING REGISTERS

To print the value of a register (X1-X4 or Y1-Y4), enter [PRT] [X] [n] or [PRT] [Y] [n]. Example:

Power up SYS RDY

[PRT]	PGM <1>
[X]	PRNT X
[2]	PRNT X2
[EXC]	PRINTING
	READY

```
X1:REG =  0000000000
Y4:REG = -0000052146
```

J. PRINTING THE MESSAGE BUFFER

The characters in the message buffer may be printed. In the local mode, the key sequence [PRT] [M] causes the alphanumeric display to show "PRNT MES". Pressing the [EXC] key now will print out the contents (32 characters) of the message buffer. Example:

Power up	SYS RDY
[PRT]	PGM <1>
[M]	PRNT MES
[EXC]	PRINTING
	READY

```
**-MESSAGE-REGISTER-CONTENTS-**


**-MESSAGE-REGISTER-CONTENTS-**
```

NOTE: For 3Q software and later, when printing in the LOCAL mode, Unidex 3 will not respond to any received characters other than <XON> and <XOFF>.

This change once again accommodates printers and other devices connected to the RS-232 interface that transmit control codes other than <XON> and <XOFF> back to Unidex 3.

When in REMOTE mode, Unidex 3 will respond as before to received characters.

K. CLEARING PROGRAMS

To erase a program, enter the edit mode, press CLEAR COMMAND and enter the program number. It will clear when executed. Example:

[EDT]	PGM <1>
[SHIFT] [CC]	CLR ?<1>
[5]	CLR 05
[EXC]	CLEARED

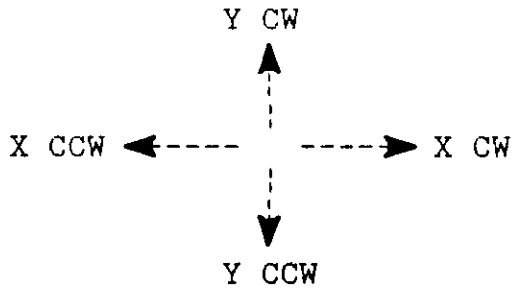
For clearing the entire memory, enter:

[EDT]	PGM <1>
[SHIFT] [CC]	CLR ?<1>
[0]	CLR 00
[EXC]	CLEARED

WARNING: If you execute the above sequence, ALL PROGRAMS WILL BE CLEARED FROM UNIDEX IIIA'S MEMORY!

L. MANUAL CONTROL

The manual control operations are STEP and SLEW. They both operate in the local mode, using the arrow keys. These arrow keys are used to effect motion as shown below:



1. Step Function

The Step function is useful for precise positioning of the drives. After depressing the STEP key, the drives will move one step each time one of the arrow keys is pressed. Example:

[IMD] [STP]

or

SYS RDY [STP]

or

READY [STP]

You may now use the arrow keys to manually position the drives.

2. Slew Function

The Slew function is useful for rapid manual positioning of the drives. After entering a feedrate and depressing the SLEW key, pressing and holding one of the arrow keys will cause one of the drives to move in a given direction until the key is released.

The feedrate can be entered as either frequency (Fnnnn) or pulse period (F=nnnn). Example:

[IMD] [Fnnnn] or [F=nnnn] [SLW]

If no feedrate is entered, the default feedrate is 250 steps per second after power-up. Any feedrate entered is stored

and is valid until changed or until the system is reset.

The X and Y feedrates can be programmed individually. Example:

SYS RDY [IMD] [X] [Fnn] [Y] [Fnn] [SLW]

3. Slew Function Using Joystick

Unidex IIIa is always ready to accept joystick input when idle in the slew mode. In the slew mode, the joystick rather than the arrow keys may be used to control direction. The feedrate controlling the speed of travel is specified to be a fraction of the joystick frequency. This is done by entering command F-nnnnn (2-65535 range). This lets Unidex IIIa know that the joystick will be used for positioning, and that the feedrate will be the joystick frequency divided by a given factor (keyboard slew overrides joystick, however). Example:

[IMD] [F-20] [SLW]

If no joystick feedfactor is entered, the default feedfactor is 2 (F-2).

To exit the step or slew mode, press CE or CC.

These manual functions are also usable whenever the system is ready after power up or after running a program or a block of program. Example:

Power-up [SLW]

or

READY [SLW]

SECTION 3-5 REMOTE PROGRAMMING

Upon powering up, the front panel keyboard will show that both local and remote LEDs are lit. This is the "local with remote enabled" state. If the next command source is the controller, the local LED will be extinguished and Unidex IIIa will go into remote programming mode. The word "remote" will be displayed.

Before sending any commands, Unidex IIIa must be addressed to listen. How it is addressed depends on what type of host is being used (see chapters 2 and 4).

Unidex IIIa implements X-on/X-off protocol during any RS-232C/RS-422A operation.

CAUTION: MAKE SURE DRIVES ARE SAFE BEFORE BEGINNING A PROGRAM.

NOTE: The Command Code and ASCII Code required for the remote programming of Unidex IIIa is in appendix 1.

A sample program which demonstrates remote programming on an IBM PC is given in appendix 6.

A. IMMEDIATE MODE

To enter the Immediate mode, send the ASCII character "I". To begin executing a program, follow the character "I" by a command string, followed by a <CR><LF>. Example:

"I G7 X300 F1 Y250 F1" <CR><LF>

NOTE: Host computers generally send <CR><LF> as an end-of-line sequence when they output any data. If such is the case, the <CR><LF> need not be included in the above string.

When Unidex IIIa completes the command block, it will send a request for service and wait for a

serial poll (see chapter 4). Once the serial poll is done by the controller, you may execute the same command block again by sending <CR><LF>. If a new command block is to be sent, enter the string of commands and send it with the <CR><LF>.

B. DOWNLOADING A PROGRAM (EDIT MODE)

After addressing Unidex IIIa to listen, you must send the ASCII character for Edit, which is E, along with the appropriate number. Example:

E99 <CR><LF> (create program 99 header)

"G90 X100 F10....M30" (motion program entered into user memory one character at a time)

% <CR><LF> (quit edit mode)

E75 <CR><LF> (create program 75 header and wait for motion program characters)

Edit functions STEP, BACKSTEP, SEARCH, CLEAR ENTRY and CLEAR COMMAND are not available in remote. You cannot edit an existing program when in remote. You must instead download the program into Unidex IIIa and edit it from the front panel.

If a program already exists in memory and you send its number over the lines, it will be erased. New data may now be entered.

There are three methods of preparing and downloading a program that can be employed. They are:

1. The program is prepared on the host and then loaded into the Unidex IIIa memory for execution either immediately or at a later

time. The host and Unidex IIIa need not be connected during operation.

2. The program is prepared on the host and is then loaded into the Unidex IIIa memory and executed. Unidex IIIa and the host communicate throughout the operation by means of service requests from Unidex IIIa and serial polls, as well as other commands, from the host.

3. The program is prepared on the host and is then loaded into Unidex IIIa one block at a time. Unidex IIIa executes on a block-by-block basis, requesting service each time a block is completed. After polling Unidex IIIa, the host sends the next command. This is the immediate mode.

C. EXECUTING A PROGRAM, AUTO MODE

In order to run a program in the Auto mode, you simply send the ASCII character "A" along with the program number and execute by sending <CR><LF>. Example:

"A 30" <CR><LF>

In this example, program 30 runs from beginning to end. When this program is complete, Unidex IIIa requests service and waits for a serial poll. To execute the same program again, program the controller to do a serial poll of of Unidex IIIa and send <CR><LF>.

NOTE: For a detailed description of how Unidex IIIa handles service requests, see chapter 4.

D. SINGLE MODE

To enter the Single mode while in remote, send an "S" followed by the program number. Example:

"S 03" <CR><LF>

Program 3 runs to first EOB (*). At this point, Unidex IIIa requests service and waits for a serial poll. The next block may be executed (after the controller does a serial poll) by sending a <CR><LF>.

CHAPTER 4: SYSTEM PROGRAMMING

System programming allows information to be sent to and retrieved from Unidex IIIa. It is the means by which the motion program (chapter 5) is downloaded, edited, run, deleted, and printed.

SECTION 4-1 LOCAL PROGRAMMING

For local system programming functions, refer to chapter 3, Getting Acquainted With Unidex IIIa. There you will find information on Local operating modes (Edit, Auto, Single, Immediate and Manual) and their functions (Step/Backstep, Search, Clear Entry, Clear Command, Step and Slew).

SECTION 4-2 ADDRESSING UNIDEX IIIA(S)

Before any remote system programming can get underway, the Unidex IIIa(s) must be addressed by the controller.

NOTE: Setting device address select switches is explained in chapter 2, Installation.

A. DEVICE ADDRESSING WITH IEEE-488 INTERFACE

Addressing of devices connected to the IEEE-488 interface bus is a primary function of the bus and is listed in the bus specifications.

Basically, for data transfer to occur, the active controller must:

1. Get the attention of all the devices (assert ATN line)
2. Unlisten all devices
3. Designate a talker (address device to talk)
4. Designate listeners (address device(s) to listen)

5. Indicate that data transfer may begin (unassert ATN line)

Different controllers implement the IEEE-488 interface in various ways, using different statements to perform the bus functions. You are advised to refer to your controller manual for proper programming sequences.

B. DEVICE ADDRESSING WITH RS-232C/422A INTERFACE

The serial line daisy chain protocol is unique to Unidex IIIa and a specific character sequence is required to address Unidex IIIa(s) on the line.

To make a device active, enter <ESC> (ASCII code 27) and the device address. All others will become inactive. Messages are received by those that are active only.

The device address is set as shown in table 2-1. Once set, then a sequence such as:

"<ESC> 8, 17, 2" <CR><LF>

addresses devices (8, 17 and 29 in the above example) to be active. The ESCAPE code "<ESC>....." unlistens all Unidex IIIa(s) on the line and restarts the addressing operation. A sequence of two digit numbers separated by a space, comma or semicolon, represents the addresses of the devices to be made active. <CR><LF> ends the addressing sequence.

NOTE: If your computer does not output <CR><LF> at the end of the output statement, then the ASCII code equivalent must be sent. The above example would be sent in BASIC as:

CHR$(27)+"18, 17, 29"+CHR$(10)+CHR$(13)

It is also important that the last device switch be set correctly to ensure proper communication.

SECTION 4-3 REMOTE PROGRAMMING

When Unidex IIIa powers up or whenever the REMOTE key on the front panel keyboard is pressed, Unidex IIIa goes into the remote enable mode. The source of the next command (keyboard or controller) determines whether Unidex IIIa goes into the local or remote mode.

With the exception of the edit functions and manual Step and Slew, any of the modes available in local are available in remote.

Once in the remote mode, only one of the interfaces (IEEE-488, RS-232C or RS-422A) may be active at any time.

Unidex IIIa implements X-on/X-off protocol when either the RS-232C or RS-422A interface is active.

In order to understand remote programming, it is necessary to have an understanding of the service request and serial poll as well. This will be explained in the next section.

A. REMOTE SELF DIAGNOSTIC TEST

When operating Unidex IIIa in the remote mode of operation, either via the RS-232C/422A interface or the IEEE-488 interface, the self diagnostic tests described in section 2-6 may be performed and any failure information retrieved by the remote controller.

The character string shown below, when sent to Unidex IIIa, will initiate the self-test:

"V" <CR><LF>

NOTE: Unidex IIIa must be addressed prior to sending this character string.

As described in chapter 2, the test may be prematurely terminated due to a failure. At the end of the testing sequence, Unidex IIIa will issue a service request to the controller. The controller is required to do a serial poll of Unidex IIIa to retrieve any failure information and enable further communication with Unidex IIIa.

The status byte obtained as a result of serial polling after self-testing is described as follows:

BIT	SET	CLEAR
Bit 7	Indicates failure	No failure
Bit 6	Always set. Setting initiates SRQ	
Bit 5	Not Used	
Bit 4	Not Used	
Bit 3	User memory altered	User memory unchanged
Bit 2	User memory write/ read failure	User memory write/read no failure
Bit 1	System RAM write/ read failure	System RAM write/read no failure
Bit 0	EPROM checksum failure	EPROM checksum verified

Unidex IIIa will issue a service request when the self-test terminates due to a failure or completes without any failure. The controller must serial poll Unidex IIIa to enable further communication.

With a user memory of 28K bytes, the self-test may take up to 4 seconds to complete, making the SRQ necessary to indicate the termination of the test.

It is also important to note that an EPROM error may cause a bus hang-up if the error in the EPROM is in the remote interface software. A bus hang-up situation is detectable by the remote controller.

SECTION 4-4 SERVICE REQUEST AND SERIAL POLL

The service request (SRQ) and serial poll are essential to remote programming.

When Unidex IIIa has completed a task or when some condition arises requiring the controller's attention, Unidex IIIa sends a service request and waits. The controller does a serial poll to find out why the service request was sent. Unidex IIIa sends the controller a status byte which explains the service request. This "handshake" is how the devices communicate.

NOTE: See appendix 2 for Status Bytes.

A. IEEE-488 SERVICE REQUEST

The IEEE-488 interface handles the service request internally. It is a bus function.

B. RS-232C/422A SERVICE REQUEST

The RS-232C/422A serial interface handles the service request one of two ways:

1. CHARACTER AS SERVICE REQUEST

A string, programmable by the controller, is sent by Unidex IIIa as a

service request. A W code (section 4-9) configures the character SRQ. Example:

```
OUTPUT "W%"
```

The % is output by J3 to indicate an SRQ. The controller must periodically read the port to input this character.

2. RTS LINE AS SERVICE REQUEST

The RTS line is turned "ON" as a service request (see figure 2-9).

When using RS-232C/422A for multiple Unidex IIIa devices, the RTS line may be configured as a Series or a Parallel service request:

a. SERIES RTS

ALL Unidex IIIa devices must request service before the RTS line to the controller is raised. A J code (section 4-9) arranges this.

b. PARALLEL RTS

ANY Unidex IIIa requesting service causes the RTS line to the controller to be raised. A K code (section 4-9) arranges this.

SECTION 4-5 ENTERING COMMANDS

A. IMMEDIATE MODE

In the immediate mode, execution begins as soon as Unidex IIIa receives <CR><LF>. Example:

"I G7 X5000 F1000 Y2500 F1000" <CR><LF>

After execution of the block is complete, Unidex IIIa sends a service request to the controller, and waits for a serial poll. Once serial polled by the controller, Unidex IIIa is ready to execute another block of commands. Just enter the commands and send <CR><LF>.

Immediate commands are not part of a prewritten program and are not stored in memory.

NOTE: Commands are sent to Unidex IIIa as 7 or 8 bit ASCII characters. See appendix 1 for ASCII character set.

B. EDIT MODE

In the remote mode of programming, Unidex IIIa's edit functions, ie., step/backstep, search, clear entry and clear command, are not accessible to you. You are limited to entering, storing, printing and deleting a program.

To enter or download a program into Unidex IIIa through a controller, transmit the mode commands:

"E nn" <CR><LF>

The above entry will create a header for program "nn". You may now transmit program "nn" motion commands (see chapter 5 for motion commands). Refer to section 3-5 B for more information.

SECTION 4-6 EXECUTING A PROGRAM

A. SINGLE MODE

Once a program is entered, it can be executed one block at a time.

In the single mode, Unidex IIIa will set a service request (SRQ) after each block. Therefore, the controller serial polls Unidex IIIa after the execution of each block.

To execute a program in the single mode, send "S", program number and <CR><LF>. Example:

"S 22" <CR><LF>

After the execution of the first command block and the serial poll, send a <CR><LF> to execute each of the subsequent command blocks.

B. AUTO MODE

Executing a program in the auto mode enables the program to run automatically, executing the motion commands in the program sequentially.

To run a program in the auto mode, send "A", the program number and <CR><LF>. Example:

"A 25" <CR><LF>

When the program is complete, Unidex IIIa will issue a service request (SRQ) and wait for a serial poll. After the serial poll, you may execute the program again by sending another <CR><LF>. To run another program, send another "A nn <CR><LF>"

NOTE: For the sequence of events in the remote operation of Unidex IIIa, see figure 4-1 (IEEE-488) and figure 4-2 (RS-232C/422A).

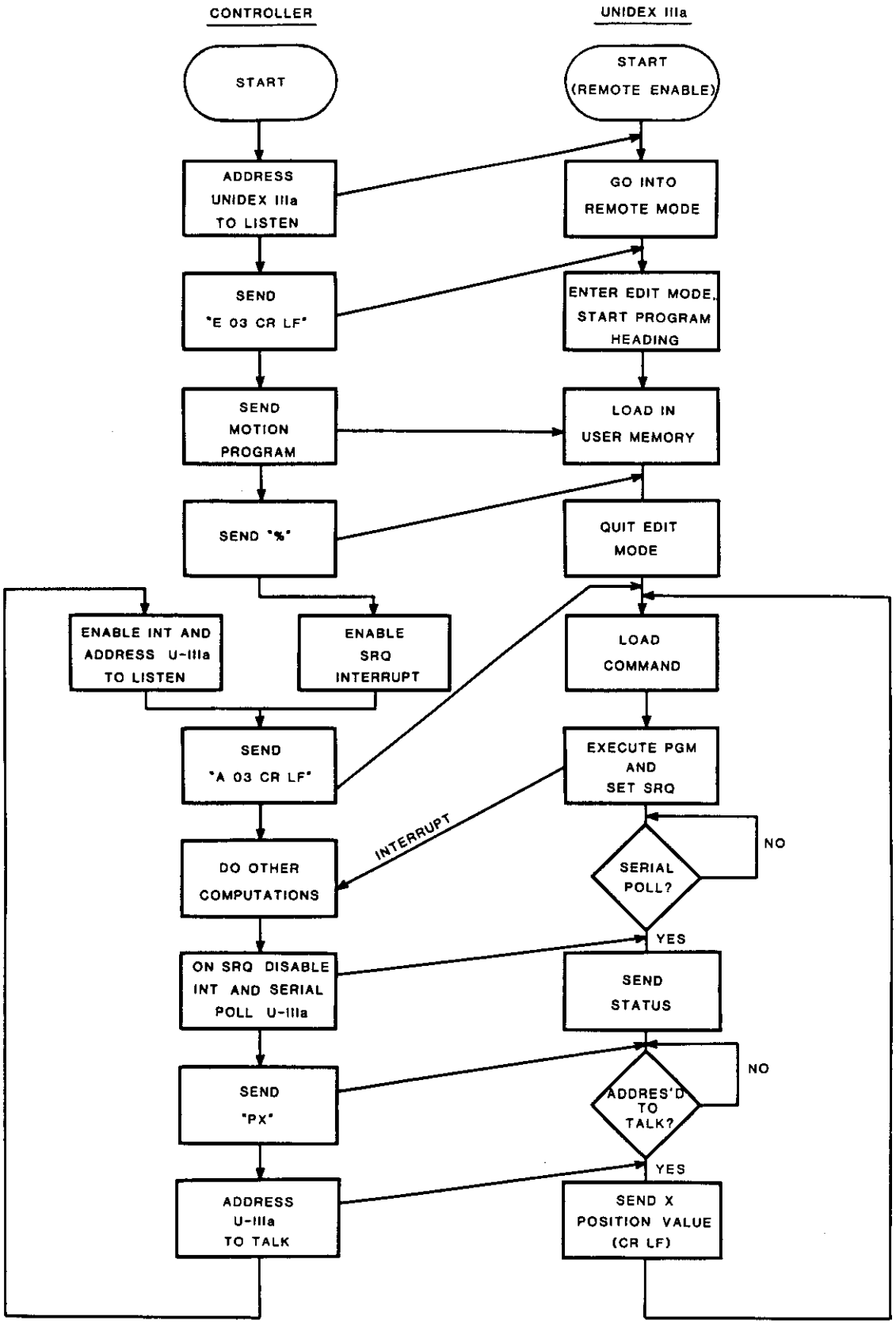


FIGURE 4-1: REMOTE OPERATION, IEEE-488 INTERFACE

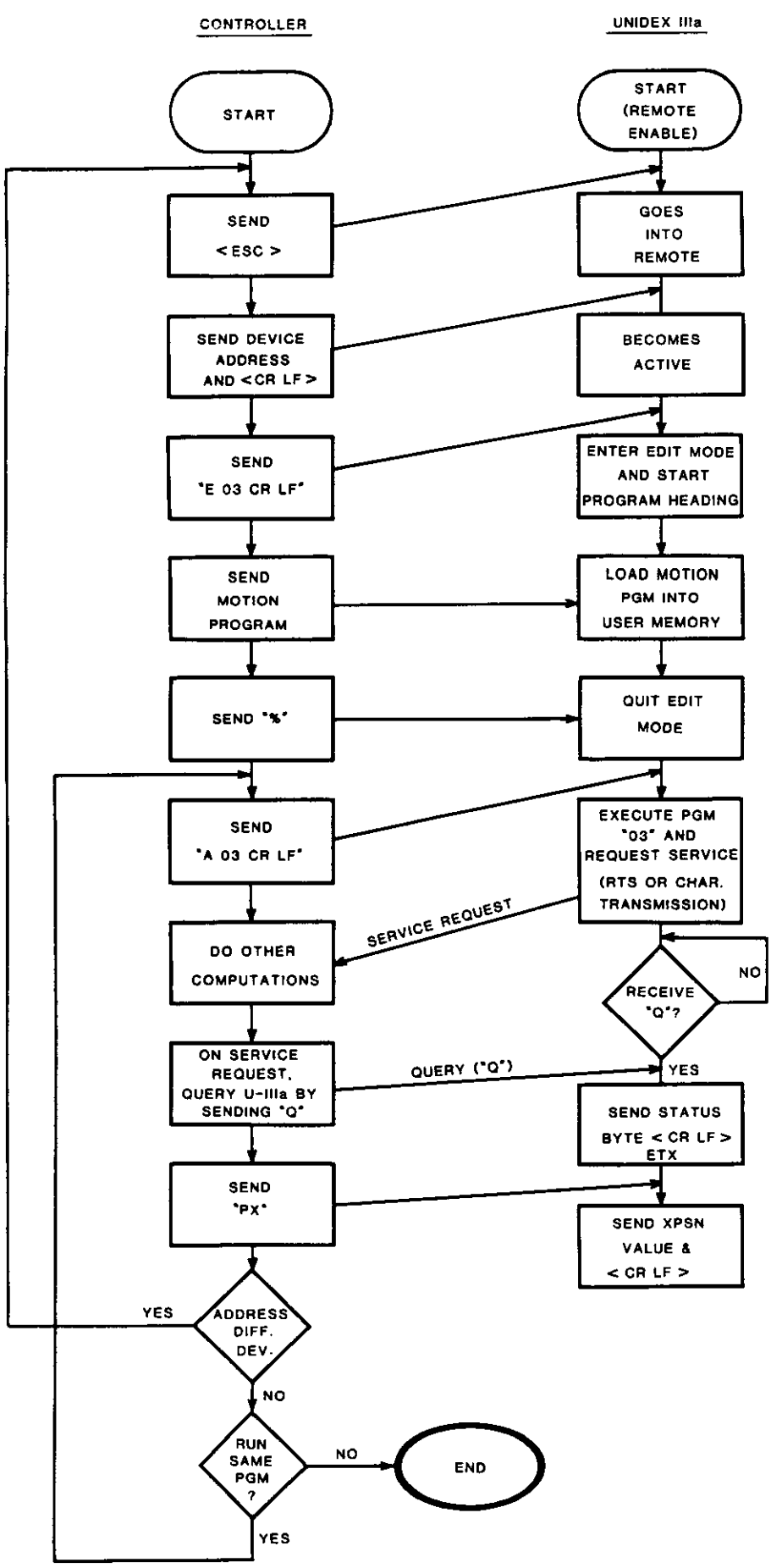


FIGURE 4-2: REMOTE OPERATION, RS-232C/422A INTERFACE

SECTION 4-7 PRINT

When Unidex IIIa is addressed and the ASCII character "P" (for print) is sent over the lines, Unidex IIIa responds by sending the requested program(s), positions, statuses or register contents to the controller.

NOTE: Before sending a print command, when interfacing with IEEE-488, it is necessary to address Unidex IIIa(s) to listen. After issuing the print command, it is necessary to address Unidex IIIa(s) to talk. Unidex IIIa then transmits the program from the user memory.

End of transmission is signaled by <CR><LF> (ETX). When Unidex IIIa completes printing, the end of transmission sequence is as follows:

If the data has a fixed format, such as X axis position or status, the transmission ends with a <CR><LF>. (This is the same for IEEE-488 and RS-232C/422A.)

If the data printed contains a format which has not been predetermined (such as the directory or a program), then the end of transmission ends with a <CR><LF><ETX> in RS-232C/422A and a <CR><LF><ETX><EOI> for IEEE-488.

NOTE: When interfacing with RS-232C/422A, only one Unidex IIIa can be addressed before issuing a Print Directory, Print Program, Print Message or Print Register command. Example:

"<ESC> 2" <CR><LF>
"PD" <CR><LF>

The directory of Unidex IIIa at address 2 will be sent to the controller.

Multiple devices may be addressed for printing X and Y positions and printing status.

When multiple devices are addressed, information will be sent back to the controller from the lowest addressed device to the highest each time the command is sent on the line.

A. PRINTING A PROGRAM

The command Pnn, where "nn" is a program number from 1 to 99, will cause Unidex IIIa to print program "nn". Example:

"P22" <CR><LF>

As mentioned in section 3-4 E, when printing a program, Unidex IIIa inserts certain leading and trailing character strings with each program being printed. They provide a means of printing (uploading) a program via the RS232/RS422 (DTE) port to an off-line storage device such as a cassette tape drive or paper tape punch and reading it back (downloading) to Unidex IIIa without any additional software on the part of the device.

The additional leading string consists of the addressing sequence and the program download command. Unidex IIIa with a device address 3 will add the following leader when printing program #24.

"<ESC> 03" <CR><LF> **"E24" <CR><LF>**

address device edit program #24

When reading back to Unidex IIIa, this string puts Unidex IIIa in remote mode, addresses it and sets up the Unidex IIIa to enter the subsequent text into program #24.

The trailing string consists of "%" <CR><LF> "L" <CR><LF> before the end-of-text transmission.

When downloading to Unidex IIIa the "%" takes Unidex IIIa out of the edit mode and the "L" <CR><LF> puts Unidex IIIa in Remote Enable Mode.

B. PRINTING A STATUS

When "PS <CR><LF>" is sent, the Unidex IIIa(s) at the location(s) addressed sends its 10 statuses to the controller followed by <CR><LF>.

C. PRINTING A POSITION

The position of an axis can be sent with a PX or a PY command. Example:

"PX" <CR><LF>

The above command will send the X position of the addressed Unidex IIIa(s) to the controller.

NOTE: Printing of X, Y and S is done in the order of address sequence, as in serial polling. Example:

"<ESC> 10, 11, 12" <CR><LF>

SYSTEM QUERY	DEVICE RESPONDING
"PX" <CR><LF>	device 10
"PX" <CR><LF>	device 11
"PY" <CR><LF>	device 10
"PS" <CR><LF>	device 10
"PX" <CR><LF>	device 12
"PY" <CR><LF>	device 11
"PX" <CR><LF>	device 10

D. PRINTING REGISTERS (X1-X4 AND Y1-Y4)

The contents of a register such as X1 may be accessed by the controller by the command:

"PX1" <CR><LF>

The contents of register X1 of the Unidex IIIa(s) addressed will be sent to the controller.

E. PRINTING THE MESSAGE BUFFER

The characters in the message buffer may also be printed or accessed by the controller. Unidex IIIa puts out the 32 characters in the message buffer followed by <CR><LF>. The characters in the message buffer may be any of the ASCII coded character set. Example:

"PM" <CR><LF>

The 32 characters of the message buffer will be sent to the controller from the Unidex IIIa addressed.

F. PRINTING THE G25 NUMBER

The "G25=ddd" command (chapter 5) assigns a number (up to 255) to the programmed halt.

When in the remote mode, the remote host may access the G25 number by sending to Unidex IIIa the string:

"PN" <CR><LF>

Unidex IIIa returns the number in binary form as a single byte followed by <CR><LF>.

G. PRINTING ENTIRE MEMORY

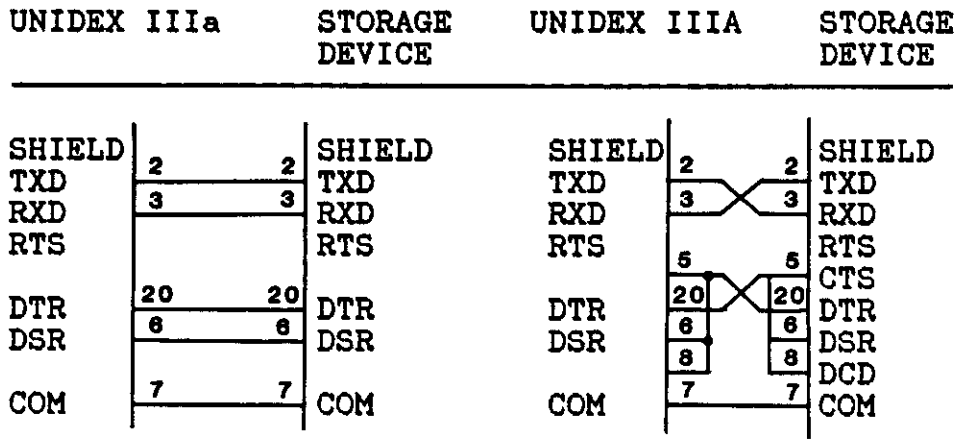
The entire memory of an addressed device may be printed by a P00 <CR><LF> command.

When printing the entire memory, leading and trailing strings are inserted for each program. The character "&" is placed after each program printed, with the character "%" following the last program printed.

It is possible to print either one program or the entire Unidex IIIa user memory to an off-line storage device and read it back to Unidex IIIa by simply powering up both devices and sending the stored data to Unidex IIIa over the RS232/RS422 link.

Unidex IIIa implements the X-on/X-off protocol during data transmission (sending as well as receiving) and therefore the storage device must respond to and implement the same protocol to assure that data is not lost during transfer.

One of the following cabling diagrams is applicable, depending on the configuration of the serial port on the storage device.



SECTION 4-8 ERASE

When it is necessary to erase a program, send an "E" and a dollar sign, followed by the program number and <CR><LF>. Example:

"E $ 25" <CR><LF>

In the above example, program 25 will be erased in all of the addressed Unidex IIIa(s).

The command "E $ 00" <CR><LF> clears the entire memory.

SECTION 4-9 RS-232C/422A INTERFACE CODES

Some ASCII codes have been assigned for control when using RS-232C/422A interface. (These functions are handled by the bus in IEEE-488.) The following commands are also listed in appendix 1.

A. CLEAR DEVICE (C)

"<ESC> 13, 15" <CR><LF>
"C" <CR><LF>

Unidex IIIa devices at addresses 13 and 15 are cleared. If you wish to clear all devices, program:

"<ESC> C" <CR><LF>

All devices will be cleared, since "C" is a universal command. A universal command is one that effects all devices if no addresses are specified.

B. HOLD (H)

"<ESC> 10, 13, 15" <CR><LF>
"H" <CR><LF>

Unidex IIIas at addresses 10, 13, and 15 are held until the trigger (T) command tells them to execute simultaneously. If you wish to hold all devices:

"<ESC> H" <CR><LF>

All devices will hold until the trigger command is sent, since "H" is a univeral command.

C. TRIGGER (T)

"<ESC> 10, 13, 15" <CR><LF>
"T" <CR><LF>

The trigger command tells all addressed devices to start execution simultaneously. "T" is not a univeral command.

D. CANCEL HOLD (O)

Command O cancels Hold. Entering:

"<ESC> 10, 13, 15" <CR><LF>
"O" <CR><LF>

cancels Hold on all addressed devices. Since "O" is a universal command, entering:

"<EXC> O" <CR><LF>

cancels Hold on all devices.

E. GO TO LOCAL (L)

"<ESC> 10, 12, 14" <CR><LF>
"L" <CR><LF>

The L command above causes all addressed devices to go to the local mode of operation. If you program:

"<ESC> L" <CR><LF>

all devices will go to local, since L is a universal command.

F. CONFIGURE SERVICE REQUEST (W)

The W command configures a character (user-selected) as service request. Example:

"<ESC> 12" <CR><LF>
"W%" <CR><LF>

The Unidex IIIa at address 12 will now send a percent sign (%) as a service request.

The W command may also be used to configure a string of 5 characters (user selected) as a service request. Example:

"<ESC> 12" <CR><LF>
"W % 0D 0A 0D 0A" <CR><LF>

The Unidex IIIa located at address 12 will now send a percent sign (%) followed by <CR><LF> <CR><LF> as a service request.

The first character after W is any ASCII coded character to be sent as SRQ, in this case "%". The next 8 characters are hexadecimal digits (0 through F) and are primarily used to configure the end-of-transmission for the service request. Unidex IIIa takes two hex characters at a time and converts them to an 8 bit binary value and stores it. Up to 4 end-of-transmission bytes may be stored in Unidex IIIa. The hexadecimal digits may be selected to configure a specific sequence which a host computer can recognize as end of transmission or a field terminator. For example, the W command:

"WD 4F 4E 45 0A" <CR><LF>

will configure the SRQ string "DONE"<LF>". In this example the spaces after "D" and between the hexadecimal digit pairs are ignored. If the number of hexadecimal digits is odd, Unidex IIIa adds a "0" to the last digit before converting to its binary value. For example, the W command:

"W$ 41 42 4" <CR><LF>

will configure a SRQ string "$AB@". The last digit, "4", will be treated as a "40".

The W command "W %" <CR><LF> will configure the single character SRQ string "%" as before.

G. CANCEL CONFIGURED SERVICE REQUEST (Z)

In order to cancel the W command, and return to the default status of using the RTS line as a service request, send a Z command. Example:

"<ESC> 12" <CR><LF>
"Z" <CR><LF>

Since Z is a universal command, to cancel W on all devices, just send:

"<ESC> Z" <CR><LF>

H. PRINTING THE STATUS BYTES IN HEXADECIMAL FORMAT

Unidex IIIa may be configured to send in Hexadecimal Digit Format, the serial poll status byte via RS-232, and the ten system status bytes as a reply to the command "PS" followed by <CR><LF>.

The command to configure Unidex IIIa is "U" followed by <CR><LF>. To cancel this and resume the binary format the command "Z" <CR><LF> may be sent. It is to be remembered that "Z" <CR><LF> also cancels the soft SRQ by character. This character may be reconfigured to "#" by the "W #" <CR><LF> command.

The following example illustrates this:

CHARACTER SENT TO UNIDEX IIIA	UNIDEX IIIA RESPONSE
<ESC> 0 1 <CR><LF>	Address device 01
U <CR><LF>	
Q <CR><LF>	00 (Status in hex)
P S <CR><LF>	A3 00 01 38 20 00 00 00 00 F0 <CR><LF> (10 status bytes in hex)
Z <CR><LF>	
Q <CR><LF>	(Null character)
P S <CR><LF>	(Non-displayable characters)

I. CONFIGURE SERIES RTS (J)

"<ESC> J" <CR><LF>

Automatically applies to all devices. With Series RTS, all Unidex IIIa(s) must raise their RTS lines before the RTS line to the controller is raised.

J. CONFIGURE PARALLEL RTS (K)

"<ESC> K" <CR><LF>

Automatically applies to all devices. With Parallel RTS, any Unidex IIIa raising the RTS line causes the RTS line to the controller to be raised.

NOTE: J and K must be universal since all of the Unidex IIIa(s) on the line must be configured as either series or parallel.

K. QUERY (SERIAL POLL) (Q)

"<ESC> 13, 14, 18" <CR><LF>
"Q" <CR><LF>
"Q" <CR><LF>
"Q" <CR><LF>

Devices 13, 14, and 18 will be serial polled, in that order.

L. CONTINUE AFTER PROGRAM HALT (B)

After a program halt (G25 command, covered in chapter 5), the "B" command following a serial poll causes the program to continue. Example:

```
"<ESC> 12" <CR><LF>
   "Q" <CR><LF>
   "B" <CR><LF>
```

When the program being executed by the Unidex IIIa at address 12 halts due to a G25 command, the "B" command tells it to continue.

SECTION 4-10 HANDLING LIMITS

It is conceivable that Unidex IIIa may get into a Limit situation by running into one of the axis limits. In such a case there are a number of protection features built into the system.

The hardware that drives the axes motors (the stepper translators, the servo amplifiers) prevents the motors from running further into a limit by inhibiting the clock pulses put out by Unidex IIIa. Clock pulses in the opposite direction are allowed to pass through permitting a retreat from the limit.

Unidex IIIa itself inhibits the indexers from putting out any more clock pulses when a limit is detected in either axis. Unidex IIIa also updates the position registers with the number of clock pulses actually put out by the indexers. The alphanumeric display indicates which axis is in limit with either "-X-LIMIT" or "-Y-LIMIT".

A. LIMIT AT POWER-UP

If one of the limits is active at power-up, Unidex IIIa detects it after initialization and self diagnostic tests are completed. Unidex IIIa will then default to the LOCAL mode and indicate the axis in limit. The procedure to exit from a limit is described in the following paragraphs.

B. LIMIT IN LOCAL MODE

When operating in the local mode, the front panel keys may be used to back out of a limit. The [SLW] key may be pressed to enter the manual slew mode and the appropriate "arrow" key used to slew out of a limit. Alternately, you can execute an immediate command or execute a program from the user memory in the auto mode or single mode to come out of the limit.

It has to be noted that any move commanded by Unidex IIIa to go further into the limit will be executed by Unidex IIIa but will be inhibited by the subsequent hardware as described above. This will result in the axis position registers having incorrect information and therefore not matching the tracking displays.

It is also important to expect the limit switches to exhibit contact bounce and create a second limit situation when backing out of limit. The limit switches are not debounced because ignoring a noisy or a faulty limit switch compromises the protection features built into the system.

C. LIMIT IN REMOTE MODE OF OPERATION

When operating in the REMOTE mode, Unidex IIIa may be set up to do one of the following when a limit is encountered.

1. Jumper 37 Removed

REMAIN IN REMOTE STATE and issue a service request to the controller. The controller has to serial poll Unidex IIIa to continue communication with it.

The controller may then proceed to send appropriate commands to back out of the limit.

2. Jumper 37 Installed

GO TO "LOCAL WITH REMOTE ENABLED" state and issue a service request. The controller has to serial poll Unidex IIIa. The status byte returned will reflect the remote enabled state. Unidex IIIa may now be accessed from the keyboard or via one of the remote interfaces.

If a limit is encountered while in the remote enabled state, Unidex IIIa goes to the local state. Thus if the limit switch has contact bounce and generates a -VE transition more than once, Unidex IIIa will be in the local state and will not be accessible via the remote interfaces.

D. LIMIT DURING MANUAL SLEW OPERATION

If Unidex IIIa detects a limit during a manual slew from the keyboard, the indexers are inhibited and a message indicating the axis in limit is displayed. The [SLW] key has to be pressed to re-enter the manual slew mode and back out of limit.

If the joystick is used for slewing, backing out of a limit may be accomplished without going to the front panel. Detecting a limit disables the joystick, but a change in the direction of any one of the axes (from the joystick) re-enables it and allows the use of the joystick to back out of a limit. That is, if a limit is encountered when slewing X-axis in a CW direction, moving the joystick in the opposite direction automatically re-enables it and slews the X-axis out of the limit in the CCW direction.

E. LIMIT DURING PROGRAMMED SLEW (G78,G79)

If the slew operation is initiated by a "G78" or "G79" command (chapter 5) in a motion program, and a limit is encountered while slewing, it is possible to back out of the limit and resume program execution.

The message "-X-LIMIT" or "-Y-LIMIT" will be displayed when the limit is detected. The [SLW] key must be pressed to re-enter the slew mode and the appropriate arrow key pressed to back out of limit and slew the axis to the desired position. The key sequence [CE] [EXC] will then exit the slew mode and resume execution of the program.

If the joystick is being used, changing the direction re-enables it and after backing out of the limit, program execution may be resumed by pressing the joystick button.

If Unidex IIIa is being operated in the REMOTE mode, a service request is issued when the limit is detected. The controller MUST do a serial poll to release control to the keyboard and joystick so that one of the arrow keys or the joystick may be used to back out of limit. The program execution may be resumed as described above after the serial poll.

SECTION 4-11 OEM PROGRAMMING CAPABILITIES OF UNIDEX IIIA

For OEM customers, the RS-232 serial communication board (690D1268) is modified to include a jumper (the OEM jumper). The function of the jumper is described in the following paragraphs.

For standard customers, it is required that the Switch No. 5 of SW1 on the Interface Board (rear panel) be permanently ON.

The character length for serial communication for both the OEM customer as well as the standard customer is now:

7 bits - SW1: Switch 6 - ON
8 bits - SW1: Switch 6 - OFF

A. FUNCTIONS OF THE OEM JUMPER

The status of the OEM jumper (IN or OUT) determines whether programs 50 through 99 in the user memory may be accessed for editing and printing. The M-function outputs M:5 through M:8 are not accessible to programs 1 through 49 when the jumper is out.

When the OEM jumper is IN, Unidex IIIa allows normal operation. All programs are accessible for editing or printing. Any M-function output may be controlled by any program.

When the OEM jumper is OUT, (or SW1: switch 5 is in the "OFF" position), Unidex IIIa will do the following.

1. Programs 50 through 99 will not be accessible for editing or printing. They may be run in the AUT or SGL modes or as a subroutine of another program. Unidex IIIa will display the message "WHAT?" and turn off all modes if an attempt is made to access these programs. In the remote mode, Unidex IIIa will issue a service request to the controller instead of the message. Local example:

[EDT] [6] [5] [EXC] __WHAT?_

[PRT] [0] [0] [EXC] __WHAT?_

Remote example:

"P 87" <CR><LF>
Unidex IIIa sends service request

The directory may be printed out, listing all programs.

2. Clear memory operation will not be carried out. Local example:

[EDT] [CC] [0] [0] [EXC] __WHAT?_

Remote example:

"E $ 00" <CR><LF>
Unidex IIIa requests service.

3. Programs 1 through 49 or the Immediate mode may only modify M-function-outputs M:1 through M:4. The outputs M:5 through M:8 are accessible only to programs 50 through 99. These outputs (M:5 through M:8) will remain unchanged unless a program numbered 50 through 99 changes them.

   For an M-function command in a program numbered 1 through 49, or in an immediate mode operation, the four outputs M:5 through M:8 will not be affected.

4. The program selected as the boot-strap program with the OEM jumper IN (OEM boot-strap) will be executed before the boot-strap program selected with the jumper OUT (user boot-strap).

5. The OEM boot-strap program or the OEM set-up program can be deselected only with the OEM jumper IN.

6. The user boot-strap or set-up program is deselected when the OEM boot-strap or set-up program is deselected. They have to be re-selected after removing the jumper.

CHAPTER 5: COMMANDS FOR MOTION PROGRAMMING

The commands needed to move the axes are the motion commands. These RS-274-D codewords are explained in this chapter. They include:

1. Feedrate (F)
2. Axis codes (X and Y)
3. Dwell (D)
4. Miscellaneous codes (M)
5. Sequence numbers (N)
6. Preparatory codes (G)
7. EOB (end-of-block) command (*)

The format of the RS-274-D codewords is letter followed by number. Example:

G7 X1000 F200 Y5000 F1000 M-58 D100 M47

The first code, G7, sends all axes home. The second code, X1000, states that the X axis is to move 1000 steps, while the third code, F200, states that the feedrate for the X axis is to be 200 steps per second. The fourth and fifth codes state that the Y axis is to move 5000 steps at a feedrate of 1000 steps per second. The sixth code, M-58, calls for a data output of the number 58 at the M-function output port. The seventh code, D100, calls for a dwell of 100 milliseconds and the last code, M47, calls for a re-execution of the program.

The example shows spaces between commands for the sake of clarity. Actually, spaces are neither needed nor provided within the RS-274-D language.

SECTION 5-1 FEEDRATE (F CODES)

The feedrate functions (F codes) determine the rate of speed at which an axis will move.

Movement can be measured in steps per second or in periods of time between steps.

Feedrate functions also include the joystick feedrate.

A. FEEDRATE FREQUENCY

Feedrate frequency is measured in steps per second. It ranges from 1 to 150,000 steps per second. Example: F25000

B. PULSE PERIOD

Pulse period is measured in microseconds per step. The time can range from 6 microseconds to 4000 seconds (4,000,000,000 microseconds). Example: F=40

C. JOYSTICK FEEDFACTOR

The joystick feedfactor (F-nnnn), is the divisor for the joystick frequency when Unidex IIIa is in the slew mode. The dividing factor can range from 2 to 65535.

If you do not enter the F-nnnn feedfactor when in the immediate/slew mode, a default feedfactor of 2 will apply. This would mean that the joystick input frequency would be divided by 2. (For a high degree of feedrate resolution, choose a high feedfactor.)

Unidex IIIa powers up in the independent feedrate mode (G00). In this mode, the feedrates for the X and Y axes are entered separately following the axis move commands. An "F" command has to be preceded by an axis move command. Example:

G00 * X3000 F600 Y4000 F800 *

In the vectorial feedrate mode (G01), the feedrate command is an individual command and stands alone. It must be entered before the axis codeword(s). Example:

G01 * F1000 X3000 Y4000 *

The feedrate of 1000 steps/second will be vectored into an X axis feedrate of 600 steps/second and a Y-axis feedrate of 800 steps/second to insure that the move between the start and end points is in a straight line.

When changing from "G00" mode to "G01" mode, Unidex IIIa expects a vector feedrate to be available either from an earlier vector move or from a new feed command. But when changing from "G01" to "G00", the X and Y components of the feedrate from the most recent vector move will be effective unless new independent feedrates are entered. Example:

G00 * X100 F100 Y100 F100 * G01 * X200 Y300 *

Unidex IIIa will display "-F-ERROR" after making the first move of X100 Y100. When changing from G01 to G00, however, the vectorial feedrate is still valid. Example:

G01 * F1000 X3000 Y4000 * G00 * X100 Y100 *

The first vector move of X3000 Y4000 will be made with an X feedrate of 600 steps per second and a Y feedrate of 800 steps per second. These feedrates will also be effective during the second move of X100 Y100.

The accuracy of the component feedrates in a vector move depends on the length of the vector. A longer move makes for better accuracy. The best resolution of the clock output is 1 micro-second. The best results are achieved for feedrates of 250 steps/second and higher and when the axis move values are of the same order of magnitude and constitute a vector move of greater than 1000 steps.

SECTION 5-2 X AND Y CODES

The axis functions (X and Y codes) are those which command motion from the X and Y axes or, in connection with certain G codes, specify position related quantities. Code Xnnnnnnnnnn specifies an X axis move (where nnnnnnnnnn <= 2,000,000,000) and code Ynnnnnnnnnn specifies a Y axis move.

In the absolute mode (G90), the number following X or Y represents the destination, as recognized in the absolute position register.

In the incremental mode (G91), the number is an increment to be added to the current value of the absolute position register. (G91 is the default code.)

Following is a list of other codes and conditions which effect the X and Y axes mode of travel.

1. Each axis may move from 1 to 2,000,000,000 steps (±1 to ±2,000,000,000).

2. Under independent feedrate (G00) conditions, the axes may be moved individually or simultaneously.

The following codes are for simultaneous moves:

a. Xnnn Ynnn
b. Xnnn Fnnn Ynnn
c. Xnnn Ynnn Fnnn
d. Snnn Fnnn Ynnn Fnnn

In the above examples, if an X or Y is followed by a feedrate, the new feedrate applies to that axis only. If no feedrate follows an X or Y, the old feedrate is retained.

3. Under G00 conditions, the axes may be programmed to move consecutively. If anything other than a feedrate command is programmed between an otherwise adjacent X and Y pair, the moves will take place individually. Example:

```
X1000 F100 N0 Y1000 F100
X1000 F100 *  Y1000 F100
```

When using an EOB (*) for this purpose, as in the second example, practice caution. It could interfere with a "skip to EOB" command, causing the program to skip to the wrong place.

4. Corner rounding (G23) as opposed to non-corner rounding (G24) is an option available in axis motion. (G24 is the default option, asserted at power-up.)

The distinction between these two codes is nonexistent for a typical non-ramped stepping motor, but is important for a DC servo system where a lag may be substantial. It is also important in ramped stepping motor drives where, again, there is a lag in system response.

In the corner rounding mode, Unidex IIIa considers a move to be complete when it has output all of the pulses called for in the move command.

In the non-corner rounding mode, Unidex IIIa waits for an "in-position" signal from the servo drive or ramped stepper system. It does not consider the move complete until this signal is received.

5. Xnnn and Ynnn codes may be used together with certain G codes which require axes codes to be complete. For example, G92 (preload position registers) must be followed by an X code, a Y code or both. Example:

```
G92 X1000
G92 Y2000
G92 X1000 Y2000
```

The first example tells Unidex IIIa to load X axis position register.

The second example tells Unidex IIIa to load Y axis position register.

The third example tells Unidex IIIa to load both the X and Y position registers.

SECTION 5-3 DWELL (D CODE)

The dwell commands (D codes) are inserted when a pause or a count is needed within the program. The pause may last only a millisecond or can endure until an event has occurred a certain number of times. This depends on which dwell command has been used.

1. Dnnnnnnn indicates a dwell. The digits "nnnnnnn" represent the required time in milliseconds (nnnnnnn < 4,000,000)

2. D=nnnnn indicates the external event counter. Unidex IIIa's C-input connector (rear panel) has an external clock input. This input can be any given event, and can be used as a dwell or a counter.

   There is also an external clock enable output - C input connector. It goes high when D=nnnnn is executed, marking the beginning of the count.

   Unidex IIIa counts each negative going edge (high-to-low transition) at this input, until D=nnnnn is satisfied. The only restriction is that each negative-going edge must be at least two microseconds apart. External clock frequency < 500 KHz.

SECTION 5-4 MISCELLANEOUS CODES (M CODES)

The miscellaneous functions (M codes) have a variety of purposes. Such commands as "program stop", "return to program start" and "return from subroutine" are M functions.

Eight buffered outputs, plus a strobe, are M functions as well. An M output may cause a motion to occur or may signal that a certain function has been completed.

The strobe can be set in order to apprise you of the fact that an M output function has occurred. No other M output should be programmed to arrive before the strobe has gone low again. To do so would mean that you would not receive a signal from the strobe that the second M output function has been activated. Figure 5-1 shows this relationship.

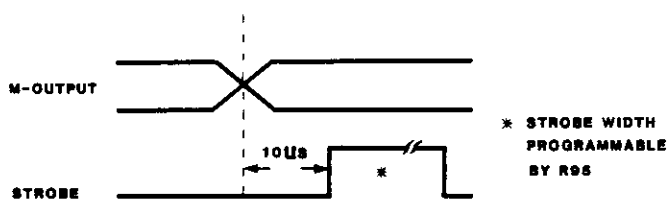


FIGURE 5-1: TIMING DIAGRAM - M-FUNCTION STROBE

The MPU board has the resistor, R95, which controls the duration of the strobe. It is located on the lower right-hand side of the MPU board on stand-offs for easy removal. To substitute another resistor value for the present one, unsolder R95 and remove it. Solder the new resistor in place. Your Unidex IIIa is shipped from the factory with an R95 value of 1.5Kohms. This value gives a strobe pulse duration of 0.15 millisecond. R95 can range from 10 ohms to 10Mohms, giving a pulse duration of from 1 microsecond to 1 second.

A complete listing of the M codes employed by Unidex IIIa is as follows:

1. M0 is the code for a stop, executed by the program. The stop is subject to an interrupt or abort command. The result of aborting a stop is to continue the program. You may use a G301 code which will "arm" C1 with an interrupt when it goes from high-to-low. When this occurs, the program will skip to the EOB and continue, ie., the stop (M0) is interrupted by G301.

2. M2 is the code which indicates the operating end of the current program, where execution comes to a halt.

   It is a general programming practice to place subroutines after the M2.

3. M30 is identical to M2.

4. M47 is the code for the "return to current program start" command. Upon receiving the M47 command, execution restarts from the beginning of the current program.

5. M89 is used to decrement the stack pointer so that the next unstacking will result in the unstacking of the address prior to the most recently stacked address (details in chapter 6).

6. M99 is the code which indicates the "return from subroutine" command.

7. M-nn indicates that the eight M-outputs form two BCD digits corresponding to "nn" in negative (active low) logic (nn = 00 through 99).

8. M=nnn indicates that the eight outputs form the binary code for "nnn" (nnn = 000 through 255), in active low logic.

SECTION 5-5 SEQUENCE NUMBERS (N CODES)

The sequence numbers (N codes) are commands associated with "line numbering" and program flow. For instance, an N code may be used to number a location at random, or to define a particular point in the program to which program flow may be directed, by the use of jumps and subroutine calls.

1. Nnnnnnn is a passive location number used to label a specific location. Use of these is not essential, unless you want to refer to these locations from elsewhere in the program. The "nnnnnn" numbers are decimal digits, as in N1200.

   When Nnnnnnn is encountered in the course of program flow, it is treated as a "no-op" or label.

2. N-nnnnnn is a call to a subroutine starting at line nnnnnn. For example, the command N-1200 would direct program flow to a subroutine that starts on line 1200.

   The end of the subroutine is designated by M99, which effects the return to the main program. After returning to the main program, the statuses of the subroutine, and not those existing within the main program before the subroutine execution, are now in effect. (There are a total of 16 levels of nesting, including all three types of subroutine calls.)

3. N=nnnnnn is a subroutine call that preserves the following statuses:

   Feedrate
   Absolute/incremental mode
   Corner rounding/non-corner rounding mode

   In other words, N=nnnnnn calls to a subroutine as N-nnnnnn does, but the statuses existing before subroutine execution, ie., those of the main program, are maintained after the return from the subroutine. For example, N=1200 calls to subroutine beginning at line N1200, but preserves the statuses of the main program.

4. N>nnnnnn is the command given to "jump to line nnnnnn". For example N>1200 means an unconditional jump to N1200.

5. N<nn is the command given to call program "nn" as a subroutine. As in N=nnnn, the statuses of the calling program are retained upon return. The subroutine program "nn" may in turn call other subroutines, and so on up to a maximum of 16 levels of nesting.

SECTION 5-6 PREPARATORY CODES (G CODES)

The G code categories are:

- System modal codes
- Home codes
- Axis reset codes
- Register preload codes
- Condition test codes
- Interrupt set-up codes
- Flag test codes
- Halt and issue service request code
- Repeat loop codes

NOTE: Advanced G codes are in chapter 6, "Advanced Programming Operation".

NOTE: A summary of all motion commands can be found in appendix 4, listed alphabetically.

A. SYSTEM MODAL CODES

The system modal codes are:

1. G90/G91 - Absolute/incremental
2. G23/G24 - Corner rounding/non-corner rounding
3. G00/G01 - Independent/vectorial feedrate
4. G47/G48/G49 - Register operations
5. G36/G37/G38/G39 - Acceleration/deceleration

1. G90/G91 - Absolute/Incremental Mode

The absolute mode, G90, commands the axes to travel to absolute positions. For instance, in the absolute mode, X100 means "go to position 100". Once this position is attained, further assertions of X100 will have no effect, since the X axis is already at position 100.

The incremental mode, G91, commands the axes to travel a programmed number of steps. For example, X100 in the incremental mode means to "move 100 steps". Every time the X100 command is executed, the X axis will move another 100 steps.

2. G23/G24 - Corner rounding/Non-corner rounding

Corner rounding may be used in a DC drive system to "smooth out" a trajectory consisting of many point-to-point moves. The smoothing is achieved because a DC system has a positional lag during indexing that is made up after the indexer completes the clock pulses. Therefore, if the next move is begun before the lag is made up, the two moves would appear to be a single continuous move. If the second move is at an angle to the first one, the resulting trajectory would have a rounded corner.

When Unidex IIIa is used to drive a non-ramped stepper motor, no positional lag exists. Therefore, corner rounding is not needed. Refer to section 5-2 on X-Y codes for more details.

3. G00/G01 - Independent/Vectorial Feedrate

As explained in section 5-1 on feedrate, G00 is the independent feedrate mode, where the X axis feedrate and the Y axis feedrate are entered separately.

G01 is the vectorial feedrate mode, where only one feedrate is entered and is vectored between both axes.

4. G47/G48/G49 - Register Operations

Unidex IIIa is in the register mode when handling registers. Normal mode (the default mode at power up) is resumed by issuing a G49 command.

Any of the registers may be assigned a value not greater than +/- 2,000,000,000, with a G47 command. This value may be a number or the value of an arithmetic expression including register numbers and the "+" and "-" signs.

An X axis move or a Y axis move or a combined two axis move may be performed based on register values, with a G48 command. In the incremental mode, the number of steps for the move is realized by evaluating an expression containing registers and numbers. In the absolute mode, the value of the expression is used as the destination position for the axis.

Feedrates may be equated to a register or an arithmetic expression as well. In non-vectorial moves with independent feedrates, F1 denotes the X axis feedrate and F2 denotes the Y axis feedrate (in steps/sec).

For vectorial moves (G01 mode), F denotes the vectorial feedrate.

Feedrate variables F, F1 and F2 are valid only in the G47 mode. For all feedrate variables, a zero or a negative value will generate a feedrate error.

5. G36/G37/G38/G39 - Acceleration/deceleration

The G36 command cancels accel/decel. It is the default mode at power-up.

Linear accel/decel mode is effected with the modal command "G37". All subsequent moves will be made with the accel/decel parameters in effect.

The acceleration time is programmed with the command "G37=nnnn" where "nnnn" specifies the time to attain 100% feedrate in milliseconds.

A command such as "G37=500" specifying the acceleration time, does not activate the linear accel/decel mode. To do that the G37 codeword is required.

Vector accel/decel is a name given to the linear accel/decel mode where the specified parameter is the acceleration rate instead of the acceleration time. The rate is programmed with a "G38=nnnnn" command. A "G38" causes subsequent moves to be made with accel/decel. Both of these commands are effective only in the vector feedrate mode (G01).

G39 programs Unidex IIIa to execute moves with an exponentially ramped accel/decel. With G39 the syntax is identical to the linear accel/decel commands. The range of values are also the same. The only difference is that the intermediate feedrate values for the acceleration and deceleration phase of the move are computed using a set of coefficients different from that for the case of linear accel/decel.

B. HOME CODES

Home codes send X, Y or both axes to the mechanical home, which is a physical reference point.

The home codes are:

1. G7 - Both axes go home
2. G60 - X axis go home
3. G60=nnnnnn - Load X axis go home feedrate
4. G61 - Y axis go home
5. G61=nnnnnn - Load Y axis go home feedrate

1. G7 - Both Axes Go Home

The G7 command sends both the X and Y axes home.

2. G60 - X Axis Go Home

The G60 command sends only the X axis home.

3. G60=nnnnnn - X Axis Go Home Feedrate

In order to load the X axis go home feedrate, program G60=nnnnnn. This is valid for software home only.

4. G61 - Y Axis Go Home

Command G61 sends only the Y axis home.

5. G61=nnnnnn - Y Axis Go Home Feedrate

Command G61=nnnnnn loads the Y axis go home feedrate. This is valid for software home only.

C. AXIS RESET CODES

The axis reset codes send a negative going reset pulse to the amplifiers and drives. These codes do not clear the position registers, however. (That function is handled by G92 X0 Y0.)

1. **G10 - Reset Both Drives**

   The G10 command resets both drives.

2. **G11 - Reset X Drive**

   The G11 command resets the X drive.

3. **G12 - Reset Y Drive**

   The G12 command resets the Y drive.

D. PRELOAD REGISTERS

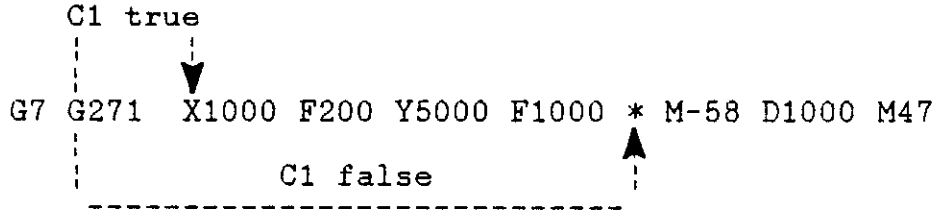
The register preset code is G92. An example of preloading registers is:

```
G92 X0 Y100
```

The X register would now contain 0, and the Y register would contain 100.

E. CONDITION TEST CODES (G271 TO G284)

The G codes that enable conditional skips to be made, make use of the end-of-block (*). The EOB character is a marker which can be used to break up a program into segments, or "blocks". Typically, you will want to segment a portion of a program that is to be executed conditionally. For example:



The code G271 in the above example calls for a test of the C1 input. If, on test, C1 is logic true (positive voltage at C1 input), the code "X100 F200 Y5000 F1000" will be executed. If, on test, C1 is logic false (ground at C1 input) execution will skip to the EOB and proceed from there. In either case, "M-58 D100 M47" will be executed.

1. G271 - Test Input C1

If input C1 is high, continue. If C1 is low, skip to the end-of-block (*). Example:

```
G7 X100 F20 G271 N-800 N>900 *
```

If C1 is high, the program will execute the subroutine, starting on line 800, and after returning from the subroutine, will jump to line 900.

If C1 is low, the above routine will be skipped over and the program will go directly to EOB (*).

2. G272 - G274 - Test Input C2 - C4

Same as above, but for input C2 through C4.

3. G281 - Test Input C1

If input C1 is low, continue. If C1 is high, skip to the end-of-block.

4. G282 - G284 - Test Input C2 - C4

Same as above, but for C2 through C4.

F. FLAG TEST CODES

1. G501 - Clear Flag 1

Clear flag 1 which has been set either by a "G511" command or an abort interrupt input C1, following a G301 or G311 command.

2. G502 - G504 - Clear Flag 2 - 4

Clear flag 2 - 4 which has been set by G512 - G514 or abort interrupt input C2 - C4.

3. G505 - G508 - Clear Flag 5 - 8

Clear flag 5 - 8 (user programmable but not associated with input conditions).

4. G511 - G518

Sets flag 1 through flag 8.

5. G521 - Test Flag 1

Test flag 1 and continue if set. If clear, skip to EOB.

6. G522 - G528 - Test Flag 2 - Flag 8

Same as above, but for flag 2 through 8.

7. G531 - Test Flag 1

Test flag 1 and continue if clear. If set, skip to the EOB.

8. G532 - G538 - Test Flag 2 - Flag 8

Same as above, but for flag 2 through flag 8.

G. REPEAT LOOP CODES

1. G661=nnnnn - Load Repeat Counter 1

Load repeat counter 1. This command loads the repeat counter "nnnnn" (nnnnn <= 65535).

2. G662=nnnnn - G668=nnnnn - Load Repeat Counter 2 - Repeat Counter 8.

Same as above, but for counter 2 through 8.

3. G671 - Decrement Repeat Counter 1

Decrement repeat counter 1. If zero, skip to EOB (*).

4. G672 - G678 - Decrement Repeat Counter 2 - Repeat Counter 8

Same as above, but for counter 2 through 8.

SECTION 5-7 END OF BLOCK

The end-of-block (EOB) is represented by the asterisk (*). In the Unidex IIIa implementation of RS-274-D, blocks are of arbitrary length. You may place an EOB essentially anywhere within the program.

The EOB can be used to aid in documentation, to single-step through an operation or to control program flow. There is no required command order with respect to end-of-block.

When an EOB is executed by Unidex IIIa, it is treated as a no-op, which is there to:

1. Act as a label

2. Define "blocks" for single/step mode program execution.

3. Define the extent of a conditional or abortive skip.

4. Provide a new line when printing programs from Unidex IIIa.

As previously discussed, the code "X1000 F100 Y5000" will cause Unidex IIIa to make the X and Y moves simultaneously. To make them execute consecutively, place a no-op such as EOB in between the two axes. Example:

```
X1000 F100 * Y5000 F500
           or
X1000 F100 N0 Y5000 F500
```

If you use EOB (first example) for the purpose of separating the X move from the Y move, EOB may delimit a skip prematurely. To be safe, use N0 instead.

NOTE: Programmable functions of Unidex IIIa not mentioned or fully explained in this chapter, are covered in chapter 6, "Advanced Programmable Operations".

CHAPTER 6: ADVANCED PROGRAMMABLE OPERATIONS

Programmable functions of Unidex IIIa not mentioned or fully explained in chapter 5 are covered here.

SECTION 6-1 UNIDEX IIIA RESET (G99)

The command G99 resets Unidex IIIa. All the initialization procedures are performed except the self diagnostic tests and execution of the boot-strap program.

SECTION 6-2 EXECUTION OF A PROGRAM AS SUBROUTINE (N<nn)

A program may be called to run as a subroutine of the currently running program. When the subroutine program ends, control is automatically returned to the main program. The subroutine programs may also be executed independently like any program. Subroutine programs are executed as independent subroutines. The following statuses of the main program are preserved:

1. Absolute mode/ Incremental mode
2. Corner rounding/ Non-corner rounding
3. Independent feedrate/ Vectorial feedrate
4. Normal move mode/ Register operation mode
5. The values of axis feedrates

The command that initiates a subroutine program call is N<nn, where "nn" is the two digit program number to be executed as a subroutine (01 to 99).

Subroutines may be nested 16 levels deep. Unidex IIIa maintains a stack space that can store up to 16 return addresses. All types of subroutine calls are included:

Program subroutines	-	N<nn
Independent subroutines	-	N=nnnnn
Ordinary subroutines	-	N-nnnnn

If a subroutine call is made when the stack is full, or if a return from a subroutine is executed when the stack is empty, Unidex IIIa flags a stack overflow error and displays the message "STK OVFL"

A program that is called as a subroutine must be entered into the user memory before the main program that is the calling program. If down loading programs from a remote controller, the subroutine program must be downloaded before the calling program. Unidex IIIa performs a precompilation when quitting the edit mode whereby the jump addresses for the "N>nnnn" command and the subroutine addresses for all three types of subroutines are computed and stored within each program. If a subroutine corresponding to a call is not available, a compile error is generated. Unidex IIIa then displays the message "C-ERR nn" where "nn" is the number of the program which generated the error.

SECTION 6-3 PUSHING LINE ADDRESS ONTO STACK (N+nnnn)

The address of a line (number "nnnn") may be put on the same stack used for storing subroutine return addresses. Example:

N+2500 *

The above command says to stack the address of line 2500. If this command is the most recent stacking operation, then executing a "M99" command will unstack the address of line number 2500 and the program will execute that line.

SECTION 6-4 POP STACK AND CONTINUE (M89)

This command is used to decrement the stack pointer so that the next unstacking will result NOT in the most recently stacked address being unstacked but the address prior to that.

Use of this command may be convenient if say, we want to jump out of a subroutine without going through the normal procedure of a return (M99). Jumping out of a program subroutine is not possible.

Pushing and popping the stack are very sensitive and tricky operations and great caution has to be exercised when incorporating these commands.

The following sample programs illustrate the use of the above commands.

PROGRAM #25

```
1  G90 G01 * F1000 X100 Y200 * ; Do a move
2  G273 N+3000  N>1000 *       ; If C-3 high, stack line 3000
3  N+2000 *                    ; Else stack 2000
4  N1000 N<37 *                ; Pgm #37 as subroutine
5  M99                         ; Unstack and jump to line
6  N3000 *                     ; address 2000 or 3000
7  M=167 X2000 Y5674 *         ; Do a move
8  N-300 *                     ; Subroutine call
9  More moves                  ; Sub. 300 return to these
                               ; moves
10 N2000 X4563 Y1342 M=34 *    ; More moves
11 More moves                  ;
12 M2 *                        ; End of main program #25

13 N300                        ; Subroutine 300
14 M=1 D200 M=0 D200           ; M functions
15 G271 M89 N>2000 *           ; Test input C1 and if low
                               ; skip to EOB.  If high,
                               ; pop stack and jump to
                               ; line address 2000
16 G272 N>300                  ; If input C2 is high go
                               ; back to line 300 (start
                               ; of sub).
17 M99 *                       ; If low return from sub.
```

PROGRAM #37

```
G91 G00 *                       ; Initialize modes
G661=10 * N10                   ; Repeat loop counter
X10 F10 M=8 Y10 F10 *           ; These moves trace a square
X-10 * Y-10 M=0                 ; of side equal to ten steps
G671 N>10 *                     ; End of repeat loop
M2 *                            ; End of program #37
```

In line 2 of the main program, either line address 2000 or line address 3000 is stacked, based on the state of the input C-3. In line 5, after executing program #37, "M99" causes a jump to either line address 3000 or 2000.

So why not test C-3 input at line 5 and jump to wherever required. We do not care whether the state of input C-3 has been changed by program #37 or not. The destination of the jump has been decided prior to executing program #37.

SECTION 6-5 REGISTER OPERATIONS (G45/G46/G47/G48/G49)

Unidex IIIa has eight registers (X1, X2, X3, X4 and Y1, Y2, Y3, Y4) that may be used by a motion program in addition to the X and Y position registers. These registers, as well as the X and Y registers, are used in register operations described in the following paragraphs.

Unidex IIIa is in the register mode when handling registers. Normal mode is the default at power up and return to normal mode from register mode is effected by the modal command "G49".

Simple register arithmetic may be performed using the plus, minus and equal-to signs. A register may be added to or subtracted from a number or a register to arrive at a value required for a specific operation. There are certain restrictions on the syntax when using these signs as described in the following subsections.

A. ASSIGNING A VALUE TO A REGISTER (G47)

Any of the registers may be assigned a value not greater than ± 2,000,000,000. This value may be a number or the value of an arithmetic expression including registers, numbers and the "+" and "-" signs. Example:

```
G47 * X1=2500 * X=5000 Y=3000 * G49 *
```

The above sample assigns the value 2500 to register X1 and value 5000 to X position register and 3000 to Y position register.

```
G47 * Y3 = X1 + X - Y - 350 *
```

This line assigns the value of the expression to the right of the "=" sign (4150) to register Y3.

When using "G47", it is illegal to use a "+" or "-" sign on the left side of the "=" sign. Unidex IIIa will flag an error and if in local mode, will display "ILL.CHAR"

Unidex IIIa also permits programming axis feedrates via the registers XI through X4 and Y1 through Y4 in the G47 mode.

Feedrate may be equated to a register or an arithmetic expression including registers, numbers and the "+" and "-" signs. In non-vectorial moves with independent feedrates for the X and the Y axes, **F1** denotes the X axis feedrate and **F2** denotes the Y axis feedrate in steps per second. For example:

```
G47 F1 = X1 + Y2 - 500 F2 = Y3 G49 *
X50000 Y100000 *
```

In the above example, the X axis will move 50000 steps with a feedrate of F1 steps/second, and the Y axis will move 100000 steps with a feedrate of F2 steps/second.

If the value of F1 or F2 is zero or negative, a feedrate error will result.

For vectorial moves in the G01 mode, **F** denotes the vectorial feedrate. For example:

```
G01 G47 F = X4 + 5000 G48 X = X2 Y = X3 - Y4 G49 *
```

In the above example, the vectorial feedrate of F steps/second will be resolved to make the vectorial move. The distances for the move are also transferred from registers. "F" is valid only in the G01 mode; if used in the non-vectorial mode, Unidex IIIa will flag an **Illegal Character** error. A zero or a negative value will generate a feedrate error.

Feedrate variables F, F1 and F2 are valid for data transfer only in the G47 mode. If included after G45, G46 or G48, an Illegal Character error will result. In the G47 block, feedrate should be on the left side of the "=" sign. (The registers being global, feedrate may be passed as a parameter to subroutines, or entered from the keyboard in the Data Input Mode (G62).)

B. REGISTER BASED MOVE (G48)

An X axis move or a Y axis move or a combined two axis move may be performed based on register values. In the incremental mode, the number of steps for the move is realized by evaluating an expression containing registers and numbers. In the absolute mode , the value of the expression is used as the destination position for the axis. Example:

```
G91 G01 * F10000 * G48 * X = 5000 Y = X + 1000 *
```

The above line makes a two axis move at a vectorial feedrate of 10000 steps/sec. X axis moves 5000 steps and Y axis moves 6000 steps.

```
G90 G00 * G47 X3 = 100 * G48 * X = X3+10 F100 Y=200 F50 *
```

Register X3 is assigned a value of 100. X axis moves to position 110 at 100 steps/sec. and Y axis moves to position 200 at 50 steps/sec., in a two axis move.

C. REGISTER COMPARISON AND CONDITIONAL SKIP (G45/G46)

Just as conditional skip to end-of-block is made based on the level of a C-input, or a repeat counter decrementing to zero, skip to end-of-block can be performed based on comparisons between registers and/or numbers.

Command "G45" allows a skip to end-of-block if the condition specified is TRUE.

Command "G46" allows a skip to end-of-block if the condition specified is FALSE.

The end-of-block automatically cancels the register comparison mode. Example:

```
G90 G00 * G45 X>=3000 X3000 F1000 *
```

If the value of X position register is greater than or equal to 3000, skip to end-of-block, or else move X axis to position 3000 at 1000 steps/second. Example:

```
G90 G00 * G46 X<3000 G48 X=3000 F1000 *
```

This line performs exactly the same function:

```
G45 X1 + 200 <> Y - X3 + 500 M=1 N>20 * M=2 *
```

Here if (X1+200) is equal to (Y-X3+500), M output #1 is turned on and program execution moves to line number 20. If the two expressions are not equal, output #2 is turned on.

As can be seen in the examples above, multiple symbols may be linked together to test for the following conditions:

1. Less than <
2. Greater than >
3. Equal to =
4. Less than or equal to <=
5. Greater than or equal to >=
6. Not equal <> or ><
7. Unconditional <=> or <>= or =<>

For example, the following program calls a subroutine to do a rectangle 10 times. The size of the rectangle is passed to the subroutine in X1 and Y1 registers and the number of repeat loops to be performed is passed in register X2. This enables different rectangles to be described and the repeat loop to be executed many times using the same subroutine. Example:

```
G90 G01 G24 *                    ; Initialize
G47 X1=100 Y1=50 *               ; Rectangle size
X2 = 10                          ; Loop count
N=200 *                          ; Subroutine call
...
...
M2 *
```

```
N200 * G00 *                        ; Subroutine begins
N40 G48                             ; Reg. based moves. Loop
                                    ; begins
X=X1 F100 * Y=Y1 F50 *              ; Two sides of rectangle
X=-X1 * Y=-Y1 *                     ; Two more sides
G47 X2 = X2-1 *                     ; Decrement counter
G45 X2 = 0 N>40 *                   ; If counter zero  quit
                                    ; loop
M99 *                               ; Return
```

It can be seen that the repeat loop in the subroutine was realized using a register operation rather than the usual G661=10 and G671 combination. The count of 10 may be passed to the subroutine via repeat counter by statement G661=10 prior to calling the subroutine. The difference is that G671 always decrements by ONE and quits the loop only when ZERO. The register operation provides for more flexibility.

SECTION 6-6 PROGRAMMABLE FEEDHOLD INTERRUPT (G201 TO G204/G211 TO G214)

The C inputs may be programmed to interrupt and halt program execution and stop clock output from the axis indexers on a high to low or a low to high transition on the input programmed. A transition in the opposite direction releases the hold, allows clock output and resumes program execution.

G201 arms input C1 to interrupt on a -VE (high to low) transition. Similarly, G202, G203 and G204 arm C2, C3 and C4.

G211, G212, G213 and G214 arm the respective inputs to interrupt on the +VE (low to high) transition. Example:

```
G201 * X1000 F1000 M-27 D1000 Y2000 F2000 *
```

Input C1 is armed to interrupt on the high to low (-VE) transition which may occur at any time. If the interrupt occurs during the dwell, the duration of the dwell will be extended by the feedhold. The amount of extension will be the time C1 input remains low.

A command arming an input to interrupt invalidates all previous interrupt-arming commands relating to that input.

More than one input may be concurrently programmed to interrupt. In such a case, the inputs behave in a manner similar to switches connected in series. Any one input may halt the program. All inputs must release hold before program can resume. Example:

```
G211 G212 * X100000 F10000 Y200000 F10000 *
```

The following sequence will hold true:

```
C1 goes high - Indexer clock outputs stopped
C2 goes high - No change
C1 goes low  - No change
C2 goes low  - Clock outputs resume
```

SECTION 6-7 PROGRAMMABLE RESET INTERRUPT (G401 TO G404/G411 TO G414)

The same C1, C2, C3 and C4 inputs may be programmed to interrupt and reset Unidex IIIa. The reset effected is the same as the "Clear Device" operation from a remote controller or the reset executed by the program statement "G99". System initialization is performed without the self-test function. The boot-strap program does not execute.

G401 through G404 arm inputs C1 through C4 to interrupt on the -VE transition. G411 to G414 arm inputs C1 through C4 to interrupt on the +VE transition.

SECTION 6-8 PROGRAMMABLE ABORT INTERRUPT (G301 TO G304/G311 TO G314)

The idea behind the G301 interrupt is to allow you to abort a stop, a move or a dwell. These are commands which take "real time" to execute and which you may want to be able to "abort" on an immediate basis. For example, consider M0, the "halt" or "stop" command. If we program "G301 M0 *", when G301 is executed, an interrupt on C1 is armed; after G301 is executed, M0 is executed. M0 brings further execution to a halt. Nothing happens until C1 input makes a transition from true to false; M0 is then aborted and execution proceeds from EOB.

1. G301 - Interrupt Move And Set Flag 1

Interrupt move, set flag 1 and skip to the EOB for a negative edge (1 to 0 transition) on C1. The position at the time of interrupt is not lost. The XPSN and YPSN are updated upon interrupt, based on the number of pulses output on each axis. This command cancels any prior G311 command.

2. G302 - G304 - Interrupt Move And Set Flag 2 - Flag 4

Same as above, but for C2 - C4. Flag 2 - Flag 4 is set for C2 - C4 negative edge transition.

3. G311 - Interrupt Move And Set Flag 1

Interrupt move, set flag 1 and skip to the EOB for a positive edge (0 to 1 transition) on C1. The position at the time of interrupt is not lost. The XPSN and YPSN are updated upon interrupt, based on the number of pulses output on each axis. This command cancels any prior G301 to G314 command.

4. G312 - G314 - Interrupt Move And Set Flag 2 - Flag 4

Same as above, but for C2 - C4 transition. Sets flag 2 - flag 4.

The following actions are taken when a programmed input causes an interrupt:

1. Halt program execution
2. Stop clock outputs from indexer
3. Update position registers based on clock pulses actually put out by the indexer
4. Set a user flag to indicate the occurrence of the interrupt
5. Skip to end-of-block and continue program execution

SAMPLE PROGRAM

```
1    G90 G01 G24 G49 * F1000 *      ; Initialize & feedrate
2    G7 *                           ; Go home
3    G313                           ; Arm input C3 to inter-
                                    ; rupt on +VE transition
4    N200 X15000 Y25000 *           ; Move command
5    G533 N>300                     ; If flag-3 is CLEAR,
                                    ; end program
6    N=100 *                        ; If SET, do subroutine
7    N>200                          ; Go back and complete
                                    ; interrupted move
8    N300 M2 *                      ; End
```

```
9     N100 * G91 G00 *                ; Subroutine begins
10    G503 *                          ; Clear flag
11    G47 X1=X Y1=Y *                 ; Store position of
                                      ; interrupt
12    G49 * X10 F10 Y10 F10           ; Moves
13    Moves
14    G90 G48 X=X1 Y=Y1               ; Get back to position of
                                      ; interrupt
15    G49 * M99 *                     ; Return
```

It is now possible to remember the X and Y positions when the interrupt occurs (Line 11) and, after executing a subroutine, go back to the position of interrupt (Line 14) and complete the move (Line 7).

SECTION 6-9 DISABLING INTERRUPTS (G321 TO G325)

All of the interrupt arming commands described in the preceding paragraphs give the programmed input the capability to interrupt at any time during program execution. To disable an input from interrupting, the following commands may be used:

```
G321             - Disable C1 input from interrupting
G322,G323,G324   - Disable C2, C3, C4 inputs
G325             - Disable all four inputs
```

SECTION 6-10 PROGRAMMABLE HALT AND SERVICE REQUEST (G25)

The command "G25" causes the motion program to halt and display a message "G25= " if in local mode. To resume the motion program, [EXC] key may be pressed once.

In the remote mode, SRQ bit 6 and G25 stop bit 1 will be set.

In the remote mode, Unidex IIIa issues a service request to the controller. The controller is required to do a serial poll. The status byte returned indicates that the SRQ was generated by a "G25" by virtue of bit 1 being set. The remote controller may read the position registers and status bytes and resume the motion program by sending to Unidex IIIa the character string "B <CR><LF>".

The command "G25=nnn" assigns a number (up to 255) to the programmed HALT. In local mode the message display is "G25=nnn". The number is to identify the HALT within a motion program. The program may be halted at different times in order to perform different operations external to the program.

When in the remote mode, the remote controller may access the G25 number by sending to Unidex IIIa the string "P N <CR><LF>". Unidex IIIa returns the number in binary form as a single byte followed by <CR><LF>.

If a plain "G25" comes after a "G25=nnn" in a motion program, the number "nnn" remains assigned to G25. The number changes only when a new number is assigned.

When in the G25 state, X and Y position and the status bytes may also be accessed by the controller.

Both in local as well as remote mode, a command to print the directory ("PD") or to print a program ("Pnn") will cancel the G25 state and inhibit continuation of the program. Pressing the mode keys [AUT], [SGL], [EDT] or [IMD] will also cancel the G25 state.

A manual slew operation may be performed while in the G25 state. Quitting the SLEW mode by the key sequence [CE] [EXC] will continue program execution with the position registers updated during the slew. However, if any of the axes encounters a limit, the G25 state is canceled.

In the remote mode, after doing a serial poll to exit the SRQ state a GO-TO-LOCAL command may be executed by the controller to put Unidex IIIa in the local with remote enabled state and give control to the keyboard. After a manual slew operation, the [RMT] key may be pressed to put Unidex IIIa back into the remote enabled state. The controller may now send a second GO-TO-LOCAL command and then address Unidex IIIa to resume the remote mode of operation.

SECTION 6-11 PROGRAMMABLE HALT AND ENTRY INTO SLEW MODE (G78/G79)

The command "G78*" halts the motion program and puts Unidex IIIa in the slew mode. The keyboard or the joystick may be used to slew the X and Y axes. When using the keyboard to slew, the key sequence [CE] [EXC] may be pressed to exit the slew mode and resume the motion program. If instead, the joystick is used to slew, and the front panel is inaccessible, the joystick button may be pressed to exit slew mode and resume program execution. Pressing the joystick button will not clear the position registers in this case (as in the normal manual slew which is initiated from the front panel). If a position register is to be cleared, that may be done by a "G92" command in the program immediately following the "G78*" or "G79" command.

Assigning a number to G78 causes a message display "SLEW=ddd". The number "ddd" identifies the SLEW operation and may be any value up to "255".

The axis slew feedrate or the joystick slew rate division factor may be programmed along with the "G78" or the "G78=ddd" command. If these values are not programmed, Unidex IIIa assumes the previously programmed values (the default values if they were never programmed). Example:

```
G78=57 X0 F1000 Y0 F2000 *
```

Unidex IIIa halts and displays "SLEW=57". Keyboard slew will move X axis at 1000 steps/second and Y axis at 2000 steps/second. Joystick feedfactor (division factor) is the previous value. Example:

G78 F-10

Unidex IIIa halts and displays "SLEW=57". Joystick clock rates will now be divided by a factor of 10. Keyboard slew rates will remain at 1000 and 2000 steps/second.

The "*" is essential after a "G78" in order to indicate to the program interpreter the extent of the command. Without it the interpreter looks for feedrates or a joystick division factor following the "G78". In cases where the "*" is inconvenient, "G79" or "G79=ddd" may be used. No slew feedrates or joystick division factor may be entered in the case of "G79" or "G79=ddd".

The X and Y position registers will be updated as a result of the slewing operation. When the motion program resumes, these new values will be effective.

SECTION 6-12 PROGRAMMABLE MESSAGE DISPLAY

Arbitrary messages of up to eight characters in length or the contents of a register such as an axis position register or one of the eight general purpose registers X1 - Y4 may be displayed on the alphanumeric display as a part of a motion program. Also displayable are the state of the eight M-OUTPUTS, the eight bits of the FLAG REGISTER and the four C-INPUTS. These binary values are represented by a string of "1"s and "0"s.

The legends on the keyboard have been modified to accommodate this new feature in the local mode of operation. All the possible ASCII characters are not available due to the small number of keys. However, all the numerals, the entire alphabet and a useful set of symbols are provided.

When editing a program in the local mode, pressing the keys [SHIFT] ["] does the following:

1. Enters the quotation mark (") on the display.

2. Starts the EDIT LED flashing on and off to indicate that you are now in the Message Entry Mode.

3. The keys on the front panel have new character assignments. These characters are printed on the right hand bottom corner of the upper and lower sections of the keycap. Where there are no such characters, the key retains the normal character assignment.

4. The [CE] and [SHIFT] keys do not change their functions.

The key sequence [SHIFT] ["] will exit the Message Entry Mode and turn the EDIT LED continuously ON.

When editing a program from the keyboard, [STP] and [BST] will align the closing quote with the right most position of the alphanumeric display. The entire message including the quotes is treated as a single command for the edit functions: STEP, BACKSTEP and CLEAR-COMMAND. In the SEARCH mode, if the character string searched for is part of a message, the cursor (right most position of display) is aligned with the closing quote.

The [CE] key will put Unidex IIIa in the message entry mode if the character cleared is the closing quote. The EDIT LED will start to blink on and off. If the entire message is cleared including the opening quote, Unidex IIIa exits the message entry mode.

If for some reason, Unidex IIIa is powered down while re-editing a message (in the message entry mode), the program being edited will neither compile nor execute correctly because the closing quote is missing from the program. The program should be cleared from the user memory and re-entered.

A. DISPLAYING ALPHANUMERIC CHARACTERS

Any of the valid ASCII codes is permissible as a character. Some of the codes may not be displayable in the alphanumeric display provided in Unidex IIIa.

The message to be displayed is enclosed within quotation marks. Unidex IIIa completes execution of all moves up to the message and then displays the message. If there are more than eight characters within the quotes, only the first eight are diplayed. If there are less than eight, they are displayed from the left end of the display. A maximum of 32 characters may be entered within the quotes. All characters from the 33rd merely replace the previous one and will eventually be replaced by the closing quote. Example:

```
G91 G00 "X-AXIS"X100F100 "Y-AXIS MOVE" Y100F100 *
```

Unidex IIIa will display "X-AXIS " and make the X move of 100 steps. Then Unidex IIIa will display "Y-AXIS M" and make the Y move.

Display updates by Unidex IIIa such as "READY" or "G25 =126" will override any programmed messages. In order to keep a message on the display for a length of time, a dwell may be required. Example:

```
"MESSAGE" D10000 *
```

The message stays on for 10 seconds before Unidex IIIa displays "READY".

B. DISPLAYING REGISTER VALUES

The value contained in the axis position registers or any of the general purpose registers may also be displayed on the alphanumeric display. Since the display is limited to only 8 characters, the value to be displayed may be formatted to any number of digits from 1 to 10. These formatted numbers may form whole or part of a message to be displayed along with other register values and alphanumeric characters.

The "<" and ">" signs are used to enclose the name of the register, the content of which is to form part of the message display. A "/" and a numeral following the register name, formats the value to a length equal to the value of the numeral. A numeral "0" or the absence of a numeral after the "/" causes the message length to default to 10 digits. For a negative register value, the minus sign takes up the first digit space.

If the X1 register has a value of 4567 and Y axis position register has a value of -5500, the following examples demonstrate the effect of different message display commands.

MESSAGE COMMAND	DISPLAY
"<X1>"	00000045
"<X1/4>"	4567
"<X1/8>"	00004567
"X1=<X1/5>"	X1=04567
"<Y>"	-0000055
"Y = <Y/4>"	Y = -500
"Y =<Y/5>"	Y =-5500
"<Y/5><X1/3>"	-5500567

C. DISPLAYING M-OUTPUT, C-INPUT AND FLAG REGISTER STATUS

The status of the conditional inputs, the M-function outputs or the flag register may be displayed as a message. These statuses are represented by a string of "1"s and "0"s. The C-input status is 4 digits long and the M-output as well as the flag register status has 8 digits.

The M-function output displayed here is the status of the peripheral data register as read by the microprocessor. This register will contain the most recently programmed M output function in BINARY or BCD format. The actual voltage levels at the connector at the rear interface board depends on the loading of the M-output buffer/drivers.

MESSAGE COMMAND	DISPLAY
"<C>"	1010
"INP=<C>"	INP=1010
"<M>"	10011001
"<F>"	00110001

SECTION 6-13 MESSAGE BUFFER

Unidex IIIa features a 32 character buffer that may be written into, printed out or read by a controller. All ASCII coded characters are permitted to be written into the message buffer.

A. PROGRAMMED INPUT TO MESSAGE BUFFER (G63 "mmm")

The message buffer may be written into from a motion program. When a message command is preceded by G63, the message is written into the message buffer instead of being displayed. The first 32 characters of the message generated by the message command are written into the message buffer. A message of less than 32 characters is written leaving the remaining bytes of the message buffer as spaces.

The "G63" remains active only for the first message following it. "G63" turns on the message buffer mode and only a message can cancel it. Other motion commands are allowed between the "G63" and the opening quotation.

COMMAND	MESSAGE BUFFER
G63 "X AXIS NOW AT <X>"	X AXIS NOW AT 0000004567
G63 "<X1>,<Y>,<X1/8>,1"	0000004567,-000005500,00004567,1
G63 "M-OUTPUTS = <M>"	M-OUTPUTS = 10011001
G63 X10F10 "" "MESSAGE"	Move X axis 10 steps. Write all spaces into message buffer. Show "MESSAGE " on display

B. PROGRAMMED PRINTING OF THE MESSAGE BUFFER (G65)

The data in the message buffer may be printed via one of the serial communication ports (RS-232C or RS-422A) when Unidex IIIa is operating in the local mode.

The command "G65" or "G65=nnn" puts out the 32 characters in the message buffer in the format shown below.

<9 spaces><32 chars. of message buffer><CR><LF>

The string of 9 spaces at the beginning is provided as a left margin for printing.

When operating in the remote mode, "G65" is identical to "G25". Unidex IIIa issues a service request to the controller. The controller is required to perform a serial poll. The controller may then read an identification number (the three digits following "G65=") assigned to the service request, by sending to Unidex IIIa the command "PN <CR><LF>".

After decoding the identification number, the controller may decide to read the message buffer.

Reading the message buffer is accomplished by sending the command "PM <CR><LF>" to Unidex IIIa. Unidex IIIa responds by sending to the controller the 32 characters in the message buffer followed by <CR><LF>.

FROM CONTROLLER	TO CONTROLLER
PM <CR><LF>	ABCDEFGHIJKLMNOPQRSTUVWXYZ123456<CR><LF>
PM <CR><LF>	0004567 -005500 10011001 1101 <CR><LF>

SECTION 6-14 DATA INPUT TO UNIDEX IIIA

It is possible to execute a programmed halt and enter data either into the message buffer or any one of the registers before resuming program execution. The keyboard is used to enter the data and restart the program.

The data may be alphanumeric characters such as a part number or a value resulting from a measurement in an automated inspection set up. These characters can be entered into the message buffer for later access by a controller or for printing in the local mode.

Alternately, the data entered may be a numeric value for one of the registers in Unidex IIIa. This value may be used later in the program as a position coordinate, feedrate, loop counter or for a register comparison operation.

Data input is initiated by the "G62..." command. The characters following "G62" specify the destination of the data entered. They also determine the range of characters that may be entered and also whether Unidex IIIa displays a prompt to the operator. The following paragraphs describe the command syntax in detail.

A. INPUT FROM KEYBOARD TO MESSAGE BUFFER (G62D)

To enter alphanumeric data into the message buffer, the motion command "G62D" is used. Unidex IIIa will halt the program and the display will prompt "ENTR MSG". The EDIT LED will flash on and off to indicate message entry mode. The keyboard character assignment changes as described previously. As characters are entered, the display will scroll from right to left. The [CE] key may be used to edit the message. A maximum of 32 characters may be entered from the keyboard. From the 33rd character position, only the last character will be replaced and the display will not scroll.

Pressing [SHIFT] ["] keys will terminate the message entry mode. To resume program execution, the [EXC] key may now be pressed.

The message may be entered again before resuming the program by pressing [SHIFT] ["]. Unidex IIIa is now back in the message entry mode.

If it is desirable to display a customized prompt instead of the default prompt of "ENTR MSG", the program may execute a message display command and then "G62D<". The "<" character prevents Unidex IIIa from updating the display and the display retains the programmed message display.

The following example, with Unidex IIIa in the local mode, illustrates the above:

```
G91 G00 X10 F10 G62D G65 Y10F10 "VOLTAGE" G62D<*
```

1. G91 G00 - Initial set up
2. X10F10 - X axis move
3. G62D - Unidex IIIa prompts "ENTR MSG" and waits for keyboard entry. The EDIT LED flashes on and off.

KEYBOARD ENTRY	DISPLAY	COMMENTS
	ENTR MSG	EDIT LED flashing
[A]	A	
[SHIFT][B]	AB	
[CE]	A	
[CE]		Display blank
[SHIFT][C]	C	
[R]	CR	
[SHIFT]["]	CR	EDIT LED turned off
[SHIFT]["]		Display blank and EDIT LED flashing
[L]	L	
[1]	L1	
[SHIFT]["]	L1	EDIT LED turned off
[EXC]	RUNNING	Program resumes

4. G65 - Unidex IIIa prints out the message buffer

5. Y10 F10 - Y axis move

6. "VOLTAGE" - Message display. Unidex IIIa shows "VOLTAGE".

7. G62D<* - The display does not change but Unidex IIIa goes into message entry mode. The message may be entered as shown in #3

B. INPUT FROM KEYBOARD TO A REGISTER (G62...)

Numeric data input to a Unidex IIIa register is similar to data entry into the message buffer. However, the range of characters accepted by Unidex IIIa is limited to the numerals and the "-" sign. The [CE] key deletes the most recent entry. The [EXC] key resumes program execution. The EDIT LED does NOT flash on and off.

The different commands and corresponding Unidex IIIa prompts are listed below. As in the case of data input to the message buffer, the prompt can be suppressed by a "<" sign after the register name.

COMMAND	PROMPT	FUNCTION
G62X	ENTER X	X position register gets the numeric value entered
G62Y	ENTER Y	Y position register gets value
G62X1	ENTER X1	X1 register gets value
....	....	....
....	....	....
G62Y4	ENTER Y4	Y4 register gets value
G62M	ENTER M	M-output register gets value
G62F	ENTER F	Flag register gets value

The last 10 digits entered will be used to compute the value to be entered into the specified register if the register is X,Y or X1 through Y4. In the case of M-output and the flag register, the 8 least significant bits of the binary value of the entered number will be loaded into the register. The real maximum value that can be entered into these registers is 255. A negative number will be converted to its 2's complement binary equivalent. When entering a negative number the "-" sign should be the first character entered. Unidex IIIa will not accept the minus sign after a numeral unless the numeral is cleared using the [CE] key.

SECTION 6-15 LIMIT DURING PROGRAMMED SLEW (G78/G79)

If the slew operation is initiated by a "G78" or "G79" in a motion program, and a limit is encountered while slewing, it is possible to back out of the limit and resume program execution.

For all information concerning Limits, refer to section 4-10, "Handling Limits".

SECTION 6-16 ACCELERATION/DECELERATION PROGRAMMING

Unidex IIIa implements a form of programmable acceleration and deceleration. With this feature it is possible to ramp up the axis feedrates at the start of a move and ramp down at the end of the move, increasing the performance limit of the amplifiers/translators, motors and stages.

Acceleration and deceleration are always simultaneously programmed. The feedrate ramp may be programmed to be linear or exponential. Also programmable is the TIME it takes to attain 100% programmed feedrate and the start/stop feedrate. In the vector feedrate mode, the acceleration rate may also be programmed in STEPS/SEC/SEC.

After programming the ACCEL/DECEL parameters, the program may switch between linear, exponential or no accel/decel, as desired. The acceleration time and start/stop feedrate will default to specific values at power up.

It should be noted that accel/decel is NOT IN EFFECT during HOME cycle or during SLEW operation.

A. ACCELERATION/DECELERATION IN OPERATION

Unidex IIIa achieves acceleration and deceleration profiles by discreetly incrementing at the start and decrementing at the end of the move, the frequency of the indexer clock outputs.

The feedrate ramp (whether linear or exponential) is effected in a maximum of 24 discrete steps. The intermediate values of feedrates for a programmed feedrate value are computed using prestored set(s) of coefficients (one set for linear and a second set for exponential accel/decel). From the programmed Acceleration Time or the Acceleration Rate, the time period between updating the feedrate from one value to the next (according to the required ramp) is derived. We will refer to this time as the sampling time.

Sampling time (Tsam) = Accel time/23 (G00)
or = Vector feedrate/accel rate/23 (G01 G38)

The divisor is 23 because 24 samples have only 23 time periods between them. The default value for sampling time is 8 milliseconds making the acceleration time 184 mS.

If the duration of the move as computed assuming no accel/decel, is longer than the acceleration time, a trapezoidal feedrate profile is obtained. If the duration is shorter, we do not attain 100% feedrate and the feedrate profile is triangular. In the latter case, to keep computations to a minimum, Unidex IIIa defaults to linear accel/decel.

The detailed computations are as shown below. The definitions of different values are first given.

Tmov: Duration of move without accel/decel

Nmov: Number of samples in move computing from Tmove

Nacl: Number of samples in acceleration or deceleration phase of move (max. 24)

Lmov: Length of move in steps as programmed

Tclk: Time period between clock pulses computed for the programmed feedrate (= 1/feedrate)

Fmax: Maximum feedrate achieved

Fdrt: Programmed feedrate

Tmov = (Lmov-1) * Tclk
Nmov = Tmov/Tsam

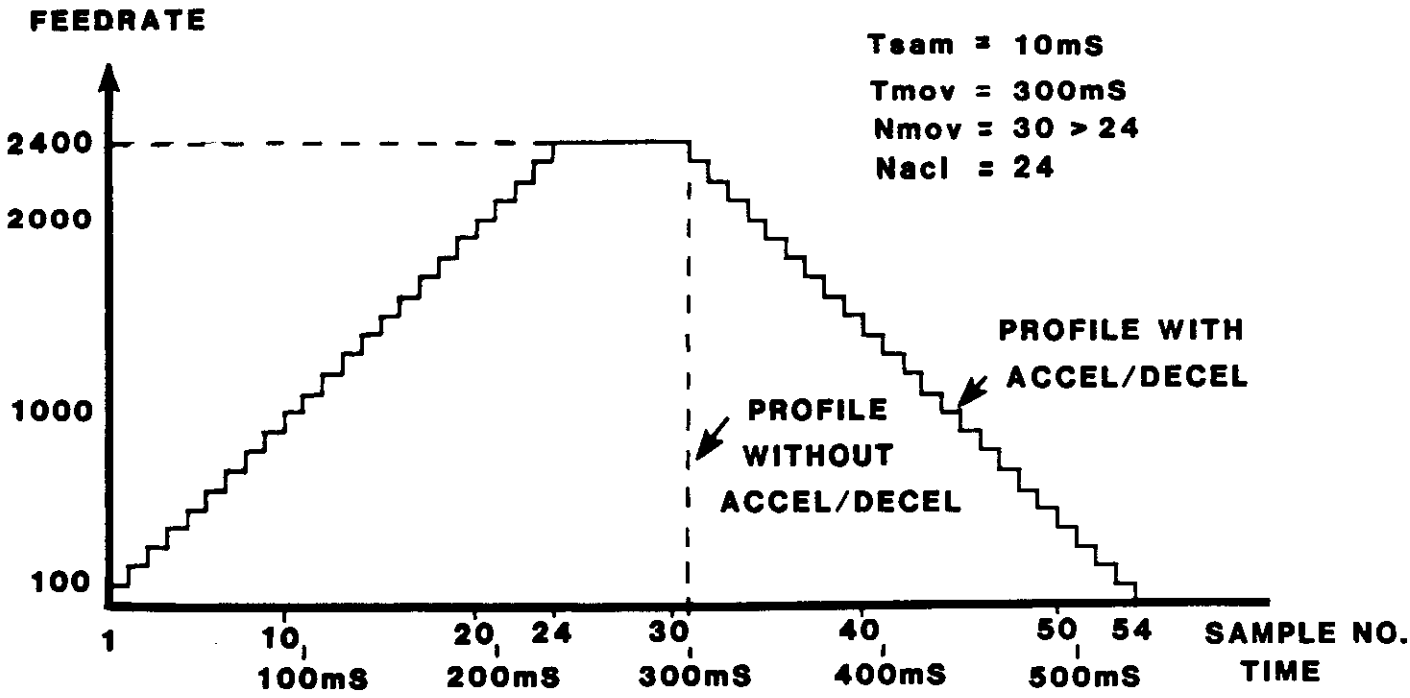
Nacl = 24 if Nmov>24
= Nmov * SQRT(24/Nmov) if Nmov<24
Approximated in Unidex IIIa by interger [(Nmov+24)/2]
(This approximation results in a positive error that causes the deceleration phase to be delayed)

Fmax = Fdrt * Nacl/24 (Unidex IIIa defaults to linear accel/decel when Nacl<24)

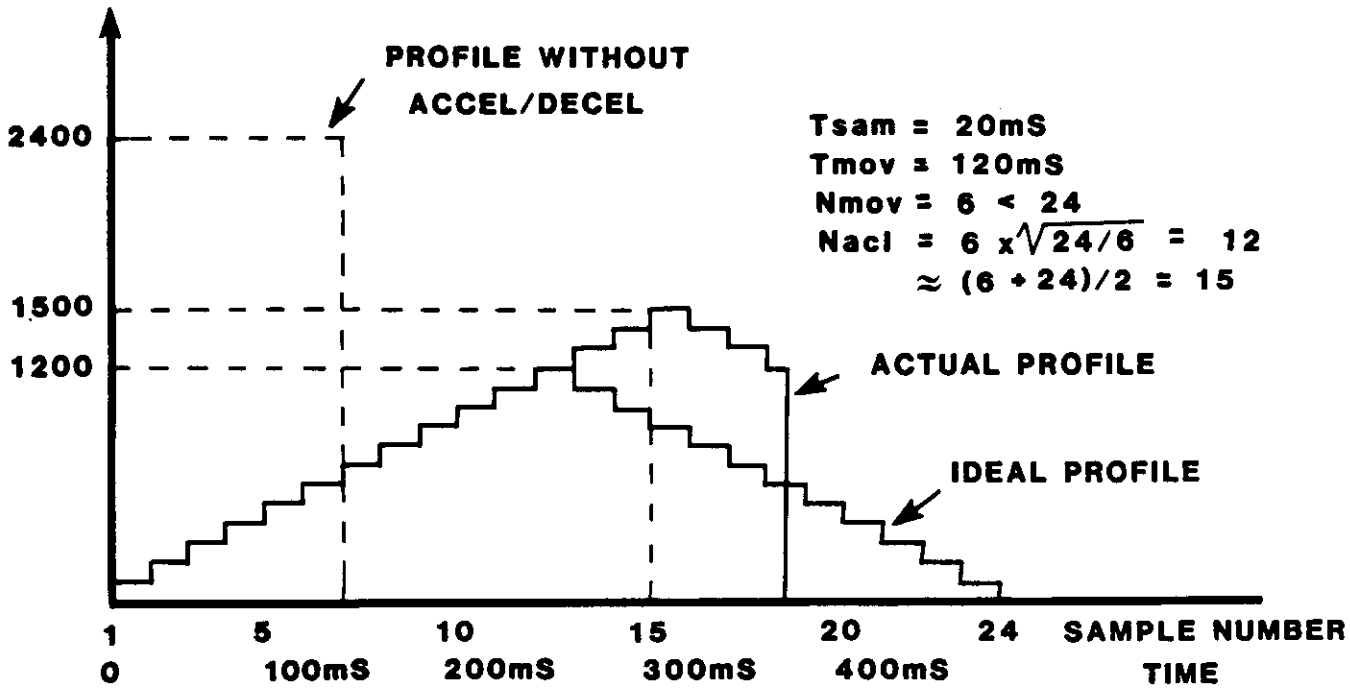
The above computations are illustrated on the next page.

B. PARAMETERS OF ACCEL/DECEL

The relevant parameters that may be programmed in the Unidex IIIa to obtain a desired accel/decel profile are described below. Once programmed, these values remain effective until reprogrammed or Unidex IIIa is reset or powered down. The programmed values remain unchanged at the end of program execution and carry over to the next program. Therefore, it is a good practice to initialize all parameters to required values at the start of a program.



MOVE LONG ENOUGH TO ACCELERATE TO 100% FEEDRATE



MOVE NOT LONG ENOUGH TO ATTAIN 100% FEEDRATE

ACCELERATION/DECELERATION IN OPERATION

1. Acceleration Time

This is the time taken by the axis to accelerate to 100% of the programmed feedrate and also the time taken to decelerate at the end of the move. The acceleration time is specified in milliseconds. The range of the value is between 23 and 4,000,000. Unidex IIIa defaults to a value of 184 milliseconds (corresponding to a sampling time of 8 milliseconds). The minimum sampling time is 1 millisecond. If the value computed from the programmed parameters is less, Unidex IIIa substitutes the value of 1 millisecond.

The acceleration time may be separately specified for linear and exponential accel/decel.

2. Acceleration Rate

In the vector feedrate mode of operation, the rate of acceleration in steps/sec/sec may be specified. This rate applies to the vector feedrate and therefore the individual components of the acceleration rate applied to each axis will be a lesser value.

The acceleration profile in this case is necessarily linear. After computing the component axis feedrates and the sampling time, the implementation of accel/decel is identical to the description above. The criterion for the lower limit of sampling time must be met when specifying an acceleration rate and programming a vector feedrate. A computed sampling time of less than 1 millisecond will be replaced by 1 millisecond.

The default value at power up is 250 steps/sec/sec. The range of the value is from 50 to 5,000,000.

3. Start/Stop Frequency

The start/stop frequency is a value of axis feedrate, below which the indexer clock pulses may directly be applied to the servo system without exceeding the limits of performance of the system. This value may be programmed within a range of 31 to 166,500 steps/seccond. Unidex IIIa powers up with a default value of 30.52 steps/second.

When accel/decel is active, if the feedrate programmed in the independent feedrate mode is less than the start/stop frequency, the related axis move is made without accel/decel. In the vector feedrate mode, the default value of 30.52 is always effective. This is done so that both the axes will move in a synchronized fashion regardless of their component feedrate values as long as the component values are greater than the default value.

C. LINEAR ACCEL/DECEL (G37)

Linear accel/decel mode is effected with the modal command "G37". All subsequent moves will be made with the accel/decel parameters in effect.

The acceleration time is programmed with the command "G37=nnnn" where "nnnn" specifies the time to attain 100% feedrate in milliseconds. The range of values and other constraints are as described in subsection B-1 above.

A command such as "G37= 500" specifying the acceleration time does not activate the linear accel/decel mode. To do that the command "G37" is required.

Linear accel/decel may be activated either in the "G00" or the "G01" mode.

D. EXPONENTIAL ACCEL/DECEL (G39)

The syntax is identical to the linear accel/decel commands. The range of values are also the same. The only difference is that the intermediate feedrate values for acceleration and deceleration phase of the move are computed using a set of coefficients different from that for the case of linear accel/decel.

E. VECTOR ACCEL/DECEL (G38)

Vector accel/decel is a name given to the linear accel/decel mode where the specified parameter is the acceleration rate instead of the acceleration time. The rate is programmed with a "G38=nnnnn" command. A "G38" causes subsequent moves to be made with accel/decel. Both of these commands are effective only in the vector feedrate mode (G01).

A vector feedrate should be effective before specifying the acceleration rate so that Unidex IIIa may compute the sampling time. If a "G38" command is executed in the "G00" mode, Unidex IIIa defaults to linear accel/decel with the currently effective value of linear acceleration time. If a "G38=nnnn" command is executed in the "G00" mode, Unidex IIIa retains the value to be used in a vector move until reprogrammed.

The programmed start/stop frequency has no effect on vector accel/decel. In order that the two axes maintain their relative trajectories during the move, the computed feedrate components are applied to the axes. The minimum component feedrate in this case is the default start/stop frequency value.

F. START/STOP FREQUENCY (G76=nnnn)

The feedrate below which accel/decel is not applied. Also the start and end feedrate values in accel/decel may be programmed from a value of 32 to 166,500 steps/sec.

G. NO ACCEL/DECEL (G36)

This is the power up default mode. A "G36" in the motion program will cancel any active accel/decel modes. One of the commands "G37", "G38" or "G39" is required to initiate some form of accel/decel.

H. ACCEL/DECEL AND CORNER ROUNDING

Unidex IIIa treats the corner rounding feature independently from the accel/decel feature. When moves are programmed without accel/decel, corner rounding may be used in a DC drive system to smooth out a trajectory consisting of many point-to-point moves. The smoothing is achieved because a DC system has a positional lag during indexing that is made up after the indexer completes the clock pulses. Therefore, if the next move is begun before the lag is made up, the two moves would appear to be a single continuous move. If the second move is at an angle to the first one, the resulting trajectory would have a rounded corner.

When accel/decel is active, the lag described above will be small due to the deceleration at the end of the move. Therefore, the corner rounding effect will be minimal.

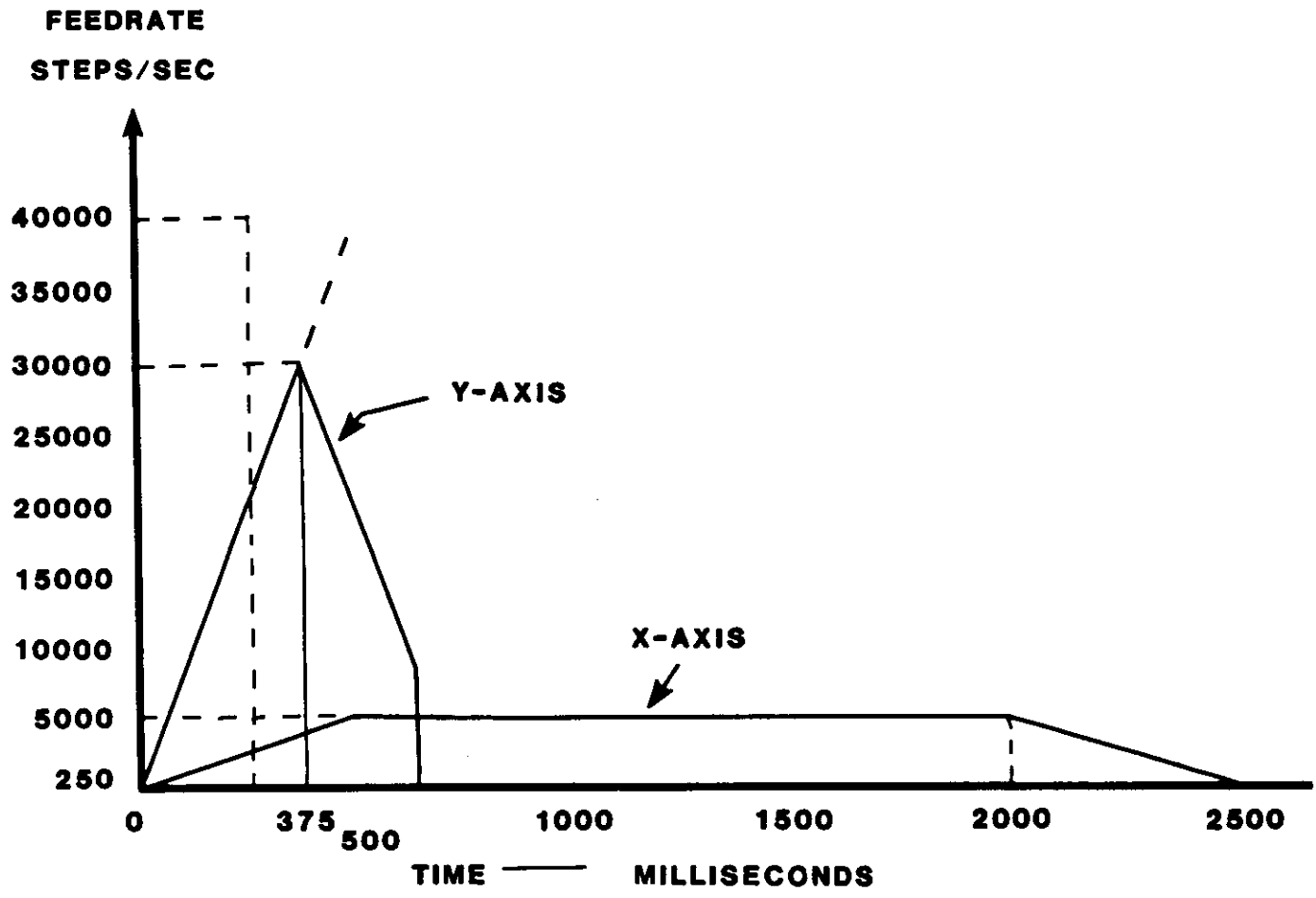
I. SAMPLE PROGRAMS

SAMPLE PROGRAM 1

```
G91 G00 G76=250 G37=500 G37 X10000 F5000 Y10000 F40000 *
```

G91 - Incremental mode
G00 - Independent feedrate mode
G76=250 - Start/stop feedrate of 250 steps/sec.
G37=500 - Linear acceleration time of 500 milliseconds
G37 - Initiate linear acceleration
X10000 F5000 - X axis move of 10000 steps at 5000 steps/sec.
Y10000 F40000 - Y axis move or 10000 steps at 40000 steps/sec.

The feedrate profiles for the two axes are as shown in the diagram below:



SAMPLE PROGRAM 2

```
G91 G01 F10000 G38=20000 G38 X5000 Y10000 G36 X1000 *
```

G01 - Vectorial feedrate
F10000 - Vectorial feedrate of 10000 steps/sec.
G38=20000 - Acceleration rate of 20000 steps/sec/sec
G38 - Initiate accel/decel (linear)
X5000 Y10000 - X and Y vectorial move with accel/decel
G36 - Cancel accel/decel
X1000 - X axis only move

SAMPLE PROGRAM 3

```
G91 G00 G76=500 G39 X10000 F10000 G37 Y5000 F8000
```

G39 X10000 F10000 - X axis move with exponential accel/decel. Acceleration time is default value or has been previously programmed.

G37 Y5000 F8000 - Y axis move with linear accel/decel. Acceleration time as above.

CHAPTER 7: SERVICE AND REPAIR

In the event of a problem, *do not attempt to replace any components.* Since there are no customer servicable parts, *to do so will void your warranty.*

NOTE: Located on the rear panel of Unidex 3, is the serial number. Whenever you contact Aerotech in reference to your Unidex 3, please have this number on hand. All information concerning your system is referenced through this number.

SHIPMENT

The procedure for shipping equipment back to Aerotech, which is described below, pertains to warranty as well as non-warranty repairs.

1. Before shipping any equipment back to Aerotech, the person making the return must call ahead for a "*Return Authorization Number*".

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 part to the factory.

Warning: Damage due to improper packaging voids warranty!

Aerotech Sales and Service offices are listed on this and the following pages. 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, CENTRAL-EAST
856 Cottonwood Drive
Monroeville, PA 15146

Phone (412) 373-4160
FAX (412) 373-4163
WV, western PA, western NY, eastern OH

AEROTECH WEST

Suite 217
7002 Moody Street
La Palma, CA 90623
Phone (213) 860-7470
FAX (213) 860-4639
AZ, southern CA

AEROTECH NORTHEAST
Executive Suite 120
270 Farmington Avenue
Farmington, CT 06032
Phone (203) 673-3330
or (203) 673-2503
FAX (203) 674-1536

MA, CT, VT, ME, RI, NH, eastern NY

AEROTECH SOUTHWEST
6001 Village Glen Drive
#3101
Dallas, TX 75206
Phone (214) 987-4556
FAX (214) 987-4706
TX, OK, LA, AR, CO, UT,MT, WY, ID, NM

AEROTECH MID-ATLANTIC
521 Kingwood Road
King of Prussia, PA 19406

Phone (215) 265-6446
FAX (215) 265-3566
MD, DC, DE, NJ, northern VA, eastern PA

AEROTECH CENTRAL-WEST

26791 Lake Vue Drive #8
Perrysburg, OH 43551
Phone (419) 874-3990
FAX (419) 874-4280
MI, IN, KY, western OH

AEROTECH NORTHWEST

444 Castro Street
Suite 400
Mountain View, CA 94041
Phone (415) 967-4996
FAX (415) 967-4998
northern CA, OR, WA, NV

AEROTECH SOUTH ATLANTIC
8804 Lomas Court
Raleigh, NC 27615
Phone (919) 848-1965
FAX (919) 848-3393
NC, TN, southern VA, AL, FL, GA, SC, MS

AEROTECH MIDWEST
PO Box 625
Dundee, IL 60118
Phone (312) 428-5440
FAX (312) 428-5471
IL, MO, KS, WI, MN, ND, SD, IA, NE

INTERNATIONAL SALES OFFICES

AEROTECH LTD.
3 Jupiter House, Calleva Park
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

AEROTECH AUSTRALASIA
224 Carr Street
Suite 7
Leederville 6007
Western Australia
Phone (619) 328-2540
FAX (619) 227-6670

INTERNATIONAL REPRESENTATIVES

BRASITEC
Rue Americo Brasiliense, 2069
Chacara Santo Antonio
Cep 04715 - Sao Paulo - SP
Brazil
Phone (5511) 523 4044
TLX 1130691 BRTC

Y. BEN MOSHE
PO Box 18125
Tel Aviv 61181
Israel
Phone (9723) 7515007
or (9723) 7513268
TLX 342436 BMS IL
FAX (9723) 727319

SIMCO
208, First Floor
Ajmeri Gate
New Delhi 110 006 India
Phone 652986
TLX 031-62176 HARS IN
FAX (9111) 510697

OPTIKON CORPORATION LTD.

410 Conestogo Road
Waterloo, Ontario
Canada, N2L 4E2
Phone 519-885-2551
FAX 519-885-4712

DONG DO TRADING CO. LTD.
Rm 903, Kwang Sung Bldg.
831-47 Youksamdong
Kangnam-Ku, Seoul, Korea
Phone (822) 556-2292
FAX (822) 556-2902
TLX 29734 DONG DO

TOKYO INSTRUMENTS INC.

Asahi-Seimei Bldg.
6-8-10 Nishikasai
Edogawa-Ku
Tokyo 134 Japan
Phone (813) 686-4711
FAX (813) 686 0831

HISCO (MALAYSIA) SDN.BHD.
1 Lorong SS13/6A
Subang Jaya Indust. Estate
47500 Petaling Jaya
Selanger, Malaysia
Phone (603) 733-4236
FAX (603) 733-6281
TLX 36226 HISCO MA

APPENDIX 1

UNIDEX IIIA
COMMAND CHARACTERS
AND
ASCII CHARACTER SET

UNIDEX IIIA COMMAND CHARACTERS

SYSTEM PROGRAMMING

MODE COMMANDS

E - EDIT

A - AUTO

S - SINGLE

I - IMMEDIATE

R - REMOTE

V - Remote self-diagnostic test

P - PRINT
- PX - PRINT X - axis value
- PY - PRINT Y - axis value
- PS - PRINT Status
- PD - PRINT Directory
- Pnn - PRINT program "nn"
- P00 - PRINT memory
- PN - PRINT identification Number
- PM - PRINT Message buffer
- PX1-PX4 - PRINT X1-X4 register values
- PY1-PY4 - PRINT Y1-Y4 register values

CR LF - Terminates command sequence

INTERFACE FUNCTION COMMANDS (RS-232C)

7F (hex)-<DELETE> - Reset Unidex IIIa

1B (hex)-<ESC> - Addressing

0D 0A (hex)-<CR><LF> - Terminates command sequence

Q - Query

C - Clear

L - Go to local

T - Trigger

H - Hold for trigger

O - Cancel HOLD

J - Serial SRQ

K - Parallel SRQ

W - Configure SRQ

B - Continue after programmed HALT

0-9 - Device addresses

U - Configure status printing in hexadecimal ASCII characters.

% - Quit Edit Mode (recommended at the end of a downloaded file)

Space, Comma or Semicolon - Any of these can separate addresses in addressing sequence.

IEEE-488

H - Hold
O - Cancel HOLD
B - Continue after programmed HALT

NOTE: The equivalent of RS-232C's Q, C, L and T in IEEE-488 are primary bus functions.

NOTE: RS-232C's J, K, U and W are not applicable in IEEE-488.

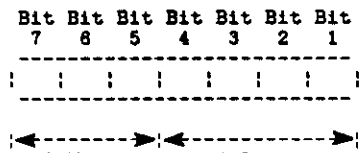
MOTION PROGRAMMING

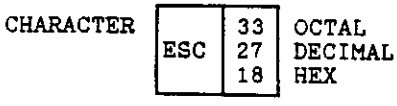
X	1	<
Y	2	>
F	3	=
G	4	+
N	5	-
F	6	"
M	7	; (comment)
D	8	* (EOB)
	9	
	0	

7-BIT ASCII CODE TABLE

ROW	COLUMN / BITS 7 6 5 / 4 3 2 1	0 (0 0 0)	1 (0 0 1)	2 (0 1 0)	3 (0 1 1)	4 (1 0 0)	5 (1 0 1)	6 (1 1 0)	7 (1 1 1)
0	0 0 0 0	NUL 0 0 0	DLE 20 16 10	SP 40 32 20	0 60 48 30	@ 100 64 40	P 120 80 50	` 140 96 60	p 160 112 70
1	0 0 0 1	SOH 1 1 1	DC1 (XON) 21 17 11	! 41 33 21	1 61 49 31	A 101 65 41	Q 121 81 51	a 141 97 61	q 161 113 71
2	0 0 1 0	STX 2 2 2	DC2 22 18 12	" 42 34 22	2 62 50 32	B 102 66 42	R 122 82 52	b 142 98 62	r 162 114 72
3	0 0 1 1	ETX 3 3 3	DC3 (XOFF) 23 19 13	# 43 35 23	3 63 51 33	C 103 67 43	S 123 83 53	c 143 99 63	s 163 115 73
4	0 1 0 0	EOT 4 4 4	DC4 24 20 14	$ 44 36 24	4 64 52 34	D 104 68 44	T 124 84 54	d 144 100 64	t 164 116 74
5	0 1 0 1	ENQ 5 5 5	NAK 25 21 15	% 45 37 25	5 65 53 35	E 105 69 45	U 125 85 55	e 145 101 65	u 165 117 75
6	0 1 1 0	ACK 6 6 6	SYN 26 22 16	& 46 38 26	6 66 54 36	F 106 70 46	V 126 86 56	f 146 102 66	v 166 118 76
7	0 1 1 1	BEL 7 7 7	ETB 27 23 17	' 47 39 27	7 67 55 37	G 107 71 47	W 127 87 57	g 147 103 67	w 167 119 77
8	1 0 0 0	BS 10 8 8	CAN 30 24 18	(50 40 28	8 70 56 38	H 110 72 48	X 130 88 58	h 150 104 68	x 170 120 78
9	1 0 0 1	HT 11 9 9	EM 31 25 19	) 51 41 29	9 71 57 39	I 111 73 49	Y 131 89 59	i 151 105 69	y 171 121 79
10	1 0 1 0	LF 12 10 A	SUB 32 26 1A	* 52 42 2A	: 72 58 3A	J 112 74 4A	Z 132 90 5A	j 152 106 6A	z 172 122 7A
11	1 0 1 1	VT 13 11 B	ESC 33 27 1B	+ 53 43 2B	; 73 59 3B	K 113 75 4B	[133 91 5B	k 153 107 6B	{ 173 123 7B
12	1 1 0 0	FF 14 12 C	FS 34 28 1C	, 54 44 2C	< 74 60 3C	L 114 76 4C	\ 134 92 5C	l 154 108 6C	¦ 174 124 7C
13	1 1 0 1	CR 15 13 D	GS 35 29 1D	- 55 45 2D	= 75 61 3D	M 115 77 4D	] 135 93 5D	m 155 109 6D	} 175 125 7D
14	1 1 1 0	SO 16 14 E	RS 36 30 1E	. 56 46 2E	> 76 62 3E	N 116 78 4E	^ 136 94 5E	n 156 110 6E	~ 176 126 7E
15	1 1 1 1	SI 17 15 F	US 37 31 1F	/ 57 47 2F	? 77 63 3F	O 117 79 4F	_ 137 95 5F	o 157 111 6F	DEL 177 127 7F

7-Bit Code





KEY

CHARACTER ENCODING

APPENDIX 2

UNIDEX IIIA STATUS BYTES

UNIDEX IIIA STATUS BYTES

1. SERIAL POLL STATUS BYTES

When serial polled, Unidex IIIa sends one byte of data to the host. When using the IEEE-488 bus, the status byte is sent as a part of the normal serial polling sequence. When using the RS-232C/422A serial line, the status byte is followed by <CR><LF><ETX> characters.

The 8 bits of the status byte are explained below:

7	6	5	4	3	2	1	0	
No Error	SRQ not active	REMOTE state	REMOTE enabled	Not busy	Not in HOLD mode	Not G25	Not M0	0
Error state	SRQ active	LOCAL state	REMOTE dis-abled	Busy running program	In HOLD mode	In G25 stop	In M0 stop	1

2. DETAILED STATUS BYTES (Sent on command "PS")

In the remote mode, when the command "PS" is sent to Unidex IIIa, 10 bytes of data are returned, each in an 8-bit binary format. When using the IEEE-488 bus, the 10 bytes are followed with <CR><LF>. When using the RS-232C/422A serial line, they are followed with <CR><LF>. Please note that an 8-bit character format is required to read the status bytes completely.

a. SYSTEM CONFIGURATION STATUS

7	6	5	4	3	2	1	0	
DC system	LO.Spd. stepper system	Hard-ware Home	X-axis CW Home	Y-axis CW Home	LOCAL On Limit	No key-board	Not Used	0
Stepper system	HI.Spd. stepper system	Soft-ware Home	X-axis CCW Home	Y-axis CCW Home	REMOTE On Limit	Key-board		1

b. SYSTEM STATUS 1 (Same as serial poll status)

c. SYSTEM STATUS 2

7	6	5	4	3	2	1	0	
Normal Mode (G49)	Register mode 00 - G45 01 - G46 10 - G47 11 - G48		G78 not active	Normal feed-rate (G00)	G79 not active	Non-Corner Roundng	Absolute Mode	0
Registr Mode			G78 active	Vector feed-rate (G01)	G79 active	Corner Roundng	Incrmntl Mode	1

d. COMMUNICATION STATUS

7	6	5	4	3	2	1	0	
Keybd. Not Active	IEEE Not Active	RS-232/ 422 not Active	Not Listenr Addresd	Not Talker Addresd	Not Used	SRQ Via RTS	Not Used	0
Keybd. Active	IEEE Active	RS-232/ 422 Active	Listenr Addresd	Talker Addresd		SRQ via Char.		1

e. COMMUNICATION ENABLED STATUS

7	6	5	4	3	2	1	0	
Keybd. dis-abled	IEEE dis-abled	RS-232/ 422 disabld	Trnsmtr enabld by DC1	DC1 sent on line	Query mode not act	Buffer print not act	Power Up Reset	0
Keybd. enabld	IEEE enabld	RS-232/ 422 enabld	Trnsmtr disabld by DC3	DC3 sent on line	Query mode active	Buffer print acitve	Front Panel Reset	1

f. SYNTAX ERROR STATUS

7	6	5	4	3	2	1	0	
NO SYNTAX ERRORS								0
Error	Illegal Char.	N error	G error	F error	M error	D error	No command	1

g. RUN ERROR STATUS

7	6	5	4	3	2	1	0	
NO RUN ERRORS								0
Error	X Limit	Y Limit	Not Used	Stack Ovflw	Not Used	EOB Srch	Missing program	1

h. EDITOR ERROR STATUS

7	6	5	4	3	2	1	0	
NO EDITOR ERRORS								0
Error	Not Used	Not Used	Not Used	Not Used	Not Used	Edit Buffer Full	Compile Error	1

i. M-FUNCTION OUTPUT STATUS

7	6	5	4	3	2	1	0	
M8 OFF	M7 OFF	M6 OFF	M5 OFF	M4 OFF	M3 OFF	M2 OFF	M1 OFF	0
M8 ON	M7 ON	M6 ON	M5 ON	M4 ON	M3 ON	M2 ON	M1 ON	1

j. CONDITION INPUT STATUS

7	6	5	4	3	2	1	0	
C4	C3	C2	C1	No Fault	Intrnl Clock	Last Device	RTS "ANDed" in daisy chain	0
				Fault light ON	Joystck Clock	Not Last Device	RTS "ORed" in daisy chain	1

APPENDIX 3

UNIDEX IIIA
SYSTEM ERROR CODES

SYSTEM ERROR CODES

NOTE: These error codes are generated and displayed by Unidex IIIa in the event of an error generated by the system and **NOT** by the user. When any of these errors is generated, Unidex IIIa software goes into a "No-op" loop and requires a **RESET** operation to exit.

Every effort has been made to trace customer errors and print out an error message that has a meaning, but some combinations will cause error nn to appear - this chart may be helpful. If not, call the factory.

ERROR 01 - Unpacking routine error. Digits beyond BCD codes.

ERROR 02 - Insert routine (EDITOR). Program cursor greater than memory end. Requires clearing memory.

ERROR 03 - Program header has no program number (EDITOR). Requires clearing memory.

ERROR 04 - Signs (>,<,-,=) routine being done when not in EDIT mode. Signs are recognized only in EDIT mode.

ERROR 05 - Signs command not active. Requires clearing memory.

ERROR 06 - CC (Clear Command) not in EDIT mode.

ERROR 07 - * (End-of-block) not in EDIT mode.

ERROR 08 - N command missing before jump address during pre-compile routine. Requires clearing of memory.

ERROR 09 - Pre-compile routine; jump address beyond memory end. Requires clearing memory.

ERROR 10 - Search routine. Cursor beyond program end.

ERROR 11 - Same as Error 03, during pre-compiling.

ERROR 12 - Same as Error 06, for CE (Clear Entry).

ERROR 13 - Does not exist.

ERROR 14 - No valid mode (EDIT, AUTO, SNGL, IMMD or PRINT) during program number entry.

ERROR 15 - Error handler routine being done without any error bits set.

ERROR 16 - Error bit set, but error type not shown.

ERROR 17 - Alphabet character (command) entered when not in EDIT mode.

ERROR 18 - Program end pointer out of bounds during STEP operation. Requires clearing memory.

ERROR 19 - "Edstat" status bits incorrect when entering EDIT, AUTO, SNGL or IMMD mode.

ERROR 20 - ERROR 29 - Do not exist

ERROR 30 - Numerals counter incorrect during G-code interpreting.

ERROR 31 - Command type number (system generated) out of bounds.

ERROR 32 - N-command got into the loader routine.

ERROR 33 - M syntax check did not do its job.

ERROR 34 - Move buffer loading routine called, but no X or Y commands in command type status (system generated).

ERROR 35 - Move process routine has no control byte.

ERROR 36 - Positional interrupt to be set up for input # 0 (There is no such input: that means syntax check failed).

ERROR 37 - G92 command without X or Y following is being considered for preloading position registers.

ERROR 38 - Error 39 - Do not exist.

ERROR 40 - X-indexer timer times out before counter is zero.

ERROR 41 - X-indexer interrupt not due to X-HI-Counter.

ERROR 42 - Y-indexer timer times out before counter is zero.

ERROR 43 - Y-indexer interrupt not due to Y-HI-Counter.

ERROR 44 - X-axis AT-HOME interrupt without a G60/G7 command.

ERROR 45 - Y-axis AT-HOME interrupt without a G61/G7 command.

ERROR 46 - Dwell timer interrupt not due to assigned counter.

ERROR 47 - X-limit interrupt not legitimate.

ERROR 48 - Y-limit interrupt not legitimate.

ERROR 49 - C-input interrupt not legitimate.

ERROR 50 - C-input interrupt in unauthorized move.

ERROR 51 - C1 interrupt not legitimate.

ERROR 52 - C2 interrupt not legitimate.

ERROR 53 - C3 interrupt not legitimate.

ERROR 54 - C4 interrupt not legitimate.

ERROR 55 - Sign not a sign when interpreting.

ERROR 56 - Error 59 - Do not exist.

ERROR 60 - Error byte MSB not set in error handler routine.

ERROR 61 - Error byte MSB set, but no error bits set.

ERROR 62 - No error bits set in RUNERRSTAT.

ERROR 63 - No error bits set in ERRSTAT.

ERROR 64 - No error bits set in EDITERRSTAT.

ERROR 90 - No accel/decel bits set in accel/decel routine.

ERROR 95 - Sampling interrupt without any axis move.

ERROR 99 - Divisor = 0 in division routine.

APPENDIX 4

UNIDEX IIIA
MOTION COMMAND SUMMARY

MOTION COMMAND SUMMARY

FEEDRATE COMMANDS

Fnnnnnn	Feedrate frequency (steps per second)
F=nnnnn	Pulse period (microseconds per step)
F-nnnn	Joystick feedfactor

AXIS COMMANDS

Xnnnnn	X axis move
Ynnnnn	Y axis move

DWELL COMMANDS

Dnnnnnnn	Dwell in milliseconds
D=nnnnn	External event counter

M COMMANDS

M0	Stop execution by program. Subject to interrupt or abort command.
M2	Operating end of current program
M30	Same as M2
M47	Return to current program start
M89	Decrement stack pointer
M99	Return from subroutine
M-nn	Indicates that eight M-outputs form two BCD digits corresponding to "nn" in negative logic (nn = 00 through 99)
M=nnn	Indicates that eight M-outputs form the binary code for "nnn" (nnn = 000 through 999)
M89	Decrement stack pointer

N COMMANDS

Nnnnnnn	Passive location number used to label a specific location.
N-nnnnnn	Call to subroutine starting at line "nnnnnn"

N=nnnnnn	Call to subroutine starting at line "nnnnnn" but preserve statuses (feedrate, absolute/incremental, corner rounding/non-corner rounding.
N>nnnnnn	Unconditional jump to line "nnnnnn"
N<nn	Call program "nn" as subroutine. Statuses of calling program are retained upon return
N+nnnn	Stack address "nnnn" (unstack and execute with an M99)

G COMMANDS

G00	Independent feedrate
G01	Vectorial feedrate
G7	Both axes go home
G10	Reset both drives
G11	Reset X drive
G12	Reset Y drive
G23	Corner rounding
G24	Non-corner rounding
G25	Program halt and issue service request
G25=nnn	Same as above, but assign identification number
G36	Cancel accel/decel
G37	Linear accel/decel
G37=nnnn	Acceleration time
G38	Vectorial accel/decel
G38=nnnn	Acceleration rate
G39	Exponential accel/decel
G45	Register comparison (skip to EOB if condition is true)
G46	Register comparison (skip to EOB if condition is false)
G47	Assign value to register
G48	Register based move
G49	Cancel register mode

REGISTER COMMANDS:
Rm = Rm<OP>Rm<OP>nnnn where:
R = X or Y
m = 1, 2, 3 or 4
<OP> = + or -

G60	X axis go home
G61	Y axis go home
G60=nnnnnn	X axis go home feedrate
G61=nnnnn	Y axis go home feedrate (Both X and Y go home feedrate apply only to low-speed stepper systems, software home).
G62	Data input
G62D	Data input from keyboard to message buffer
G62...	Data input from keyboard to register
G63"mmm"	Programmed input to message buffer
G65	Programmed printing of message buffer
G65=nnn	Programmed printing of message buffer
G76=nnn	Start/Stop feedrate value for accel/decel
G78*	Halt motion program and enter Slew mode
G78Fnnn*	Halt motion program and slew at a rate of "nnn".
G79	Halt motion program and enter Slew. Cannot enter feedrate or division factor
G78=nnn*	Halt motion program, enter Slew mode and assign an identification number
G79=nnn	Halt motion program, enter Slew mode and assign an identification number
G90	Absolute mode
G91	Incremental mode
G92	Preload position registers
G99	Reset Unidex IIIa
G201	Arms input C1 for feedhold interrupt on -VE transition
G202-G204	Arms inputs C2 to C4
G211	Arms input C1 for feedhold interrupt on +VE transition
G212-G214	Arms inputs C2 to C4
G271	Test input C1 (skip to EOB if low)
G272-G274	Test inputs C2 to C4
G281	Test input C1 (skip to EOB if high)
G282-G284	Test inputs C2 to C4
G301	Arms input C1 for abort interrupt on -VE transition
C302-G304	Arms inputs C2 to C4

G311 Arms input C1 for abort interrupt on +VE transition

G312-G314 Arms inputs C2 to C4

G321 Disable C1 input from interrupting
G322-G324 Disable C2 to C4 inputs from interrupting

G325 Disable all 4 inputs from interrupting

G401 Arms input C1 for Reset interrupt on a -VE transition
G402-G404 Arm inputs C2 to C4

G411 Arms input C1 for Reset interrupt on a +VE transition
G412-G414 Arm inputs C2 to C4

G501 Clears flag 1
G502-G508 Clear flags 2 to 8

G511-G518 Set flags 1 to 8

G521 Test flag 1 skip to EOB if clear
G522-G528 Test flags 2 to 8

G531 Test flag 1 skip to EOB if set
G532-G538 Test flags 2 to 8

G661=nnnnn Load repeat counter 1
G662=nnnnn - G668=nnnnn Load repeat counters 2 to 8

G671 Decrement repeat counter 1 and if zero skip to EOB
G672-G678 Decrement repeat counters 2 to 8

APPENDIX 5

UNIDEX IIIA
SYSTEM DEFAULT VALUES

SYSTEM DEFAULT VALUES

Unidex IIIa powers up in certain default states (refer to section 2-6) and with certain default values. The different default values are listed below.

X Axis Slew Rate - 250 steps per second
Y Axis Slew Rate - 250 steps per second

X Axis Home Feedrate - 250 steps per second
Y Axis Home Feedrate - 250 steps per second

Joystick Feedfactor - 2

Linear Acceleration - Time to attain 100% feedrate - 184 mS
Exponential Accel - Time to attain 100% feedrate - 184 mS

Sampling Time (Tsam) - 8 mS

Start/Stop Frequency - 30.52 steps per second

APPENDIX 6

SAMPLE BASIC PROGRAM
FOR
UNIDEX IIIA

SAMPLE BASIC PROGRAM

```
10  REM SAMPLE PROGRAM - UNIDEX IIIA COMMUNICATION
20  REM USING IBM  PC (AT) - DOS V 3.10 - BASICA
30  OPEN "COM2:9600,N,8,1,RS,LF" AS #1   ' SET UP RS232 PORT
40  A$= "02"                             ' DEVICE ADDRESS
50  PRINT #1, CHR$(27);A$                ' <ESC> 02 <CR><LF>
60  C$="W%"                              ' SRQ SET UP STRING
70  D$="U"
80  PRINT #1, C$                         ' CONFIGURE SRQ AS %
90  PRINT #1, D$                         ' CONFIGURE STATUS IN
                                           HEX ASCII
100 B$="I D1000"                         ' IMMEDIATE MODE COMMAND
                                           (DWELL)
110 PRINT #1, B$                         ' SEND COMMAND
120 GOSUB 230                            ' WAIT FOR SRQ (%)
130 E$="Q"
140 PRINT #1, E$                         ' SEND "Q" FOR SERIAL
                                           POLL
150 INPUT #1, A$ : PRINT A$              ' RECEIVE STATUS (CR LF
                                           TERMINATES)
160 F$ = STRING$(1,A$)                   ' GET FIRST HEX CHARACTER
                                           OF STATUS
170 IF F$ <> "4" GOTO 140                ' FIRST CHARACTER MUST
                                           BE "4"
180 PRINT "DONE"
190 END                                  ' END OF PROGRAM
200 REM
210 REM SUBROUTINE TO WAIT FOR "%" AFTER IMMEDIATE COMMAND
220 REM
230 A$= INPUT$(1,#1)                     ' READ FROM PORT, ONE
                                           CHARACTER AT A TIME
240 IF A$<>"%" GOTO 230                  ' WAIT UNTIL "%"
250 PRINT A$                             ' PRINT ON SCREEN
260 RETURN                               ' RETURN FROM LOOP

OK
RUN
%
40
DONE
OK
```

APPENDIX 7

REMOTE TEST PROGRAMS (IEEE AND RS-232) USING A HEWLETT-PACKARD 85

IEEE TEST PROGRAM

```
 10 ! UNIDEX IIIA/ HP-85 TEST
                PROGRAM
 20 ! JUNE 1985
 30 OPTION BASE 1
 40 DIM S$[4104],J$[4096],L$[60]
    ,P$[3],W$[80]
 42 S$="" @ J$="" @ P$="" @ R=0
    @ A=0 @ G=0 @ D=0 @ M=0
 45 INTEGER T
 50 RESET 7
 60 SET TIMEOUT 7;5000
 70 ON TIMEOUT 7 GOTO 8910
 80 ENABLE KBD 5 ! KBD INPUTS ON
    LY
 85 L$="      UNIDEX IIIa ADDRES
    S"
 90 GOSUB 8800
100 DISP @ DISP "    ENTER DEVIC
    E ADDRESS (XX)";
105 ON ERROR GOTO 90
110 INPUT N$
111 OFF ERROR
115 IF N$="" THEN 90
120 IF LEN(N$)>2 THEN 90
121 FOR I=1 TO LEN(N$)
122 IF NUM(N$[I,I])<48 OR NUM(N$
    [I,I])>57 THEN 90
123 NEXT I
125 N=VAL(N$)
130 IF N<1 OR N>31 THEN 90
150 N=N+700
160 REMOTE N
170 S1=SPOLL(N) @ STATUS 7,1 ; C
    0
180 IF R#1 THEN 3500
200 D=0 @ L$="          MODE SELE
    CTION"
205 GOSUB 8800
210 DISP @ DISP @ DISP "     (1)
     EDIT MODE"
220 DISP "      (2) SINGLE MODE"
230 DISP "      (3) AUTO MODE"
240 DISP "      (4) IMMEDIATE MOD
    E"
250 DISP "      (5) MESSAGE TEST"
260 DISP "      (6) HARDWARE TEST
    "
262 DISP "      (7) END PROGRAM"
265 GOSUB 8700
269 ON ERROR GOTO 200
270 INPUT K$
280 IF LEN(K$)>1 THEN 200
290 K=NUM(K$)-48
300 IF K<1 OR K>7 THEN 200
305 OFF ERROR
310 ON K GOTO 800,2700,3000,360,
    550,3500,9900
360 S1=SPOLL(N) @ STATUS 7,1 ; C
    0
370 DISP @ DISP "SPOLL= ";DTH$(S
    1)
```

```
380 OUTPUT N ;"PS"
390 S1=SPOLL(N) @ STATUS 7,1 ; C
    0
400 ENTER N USING "10(B)" ; S,S1
    ,S2,S3,S4,S5,S6,S7,S8,S9
410 S1=SPOLL(N) @ STATUS 7,1 ; C
    0
420 IF BIT(S,7)=0 THEN F$="F1000
    0"
430 IF BIT(S,7)=1 AND BIT(S,6)=1
     THEN F$="F3000"
440 IF BIT(S,7)=1 AND BIT(S,6)=0
     THEN F$="F400"
450 J$="G7*D1000X1000"&F$&"Y1000
    "&F$&"X-1000Y-1000*"
451 D=0
452 L$="          IMMEDIATE MODE"
453 GOSUB 8800
454 DISP @ DISP "ENTER COMMAND S
    TRING OF LESS    THAN 64 BYT
    ES"
455 DISP @ DISP " USE-  * FOR EO
    B(END OF BLOCK)"
456 DISP "       PRESS END LINE
    TO RUN"
469 DISP @ DISP J$;
470 INPUT S$
480 IF S$="" THEN S$=J$ @ GOTO 4
    90
481 IF LEN(S$)>64 THEN DISP "COM
    MAND STRING IS LONGER THAN 6
    4 BYTES" @ WAIT 500 @ GOTO 4
    51
490 A$="I"
500 GOSUB 8000
510 GOSUB 8600
520 IF T=1 THEN 200
530 IF T=2 THEN 500
540 GOTO 451
550 D=0
555 P$="P10"
560 L$="          MESSAGE TEST"
570 GOSUB 8800
580 DISP @ DISP "PLEASE ENTER YO
    UR NAME";
585 S$=""
590 ON ERROR GOTO 550
600 INPUT N$
610 OFF ERROR
615 IF N$="" THEN N$="YOU"
620 J$="        AEROTECH WELCOME
    S "&N$&" TO THE WORLD OF UNI
    DEX IIIA!!!!"
630 FOR I=1 TO LEN(J$)-8
640 A$=CHR$(34)&J$[I,I+7]&CHR$(3
    4)&"D100*"
650 S$=S$&A$
660 NEXT I
675 OUTPUT N ;"E$10"
680 OUTPUT N ;"E10"
```

```
682 OUTPUT N ;S$
688 OUTPUT N ;"A10"
690 GOSUB 8070
700 GOSUB 8600
710 IF T<1 OR T>3 THEN 700
720 IF T=1 THEN 200
730 IF T=2 THEN 688
740 D=0
750 GOSUB 8800
760 DISP @ DISP "ENTER MESSAGE"
761 DISP "(DO NOT USE QUOTATION
    MARKS)"
765 DISP "(DO NOT USE COMMAS)"
770 INPUT S$
780 J$="          "&S$&"
    "
785 S$=""
790 GOTO 630
800 D=0
810 L$="              EDIT MODE"
820 GOSUB 8800
830 DISP @ DISP "     (1) LOAD P
    ROGRAM FROM TAPE"
840 DISP "     (2) STORE PROGRAM
     TO TAPE"
850 DISP "     (3) LOAD PROGRAM
    FROM                UNIDEX II
    I"
860 DISP "     (4) STORE PROGRAM
     TO                 UNIDEX II
    I"
870 DISP "     (5) CREATE PROGRA
    M"
880 DISP "     (6) DELETE PROGRA
    M"
890 DISP "     (7) CHANGE PROGRA
    M"
895 DISP "     (8) GO TO MENU"
900 GOSUB 8700
905 ON ERROR GOTO 800
910 INPUT T$
920 IF LEN(T$)>1 THEN 800
930 IF NUM(T$)<49 OR NUM(T$)>56
    THEN 800
935 OFF ERROR
940 T=VAL(T$)
950 ON T GOTO 1550,1500,1800,170
    0,960,2300,1200,200
960 ! CREATE OR ADD PROGRAM
965 D=0
970 GOSUB 8800
975 DISP @ DISP "PRESS END LINE
    KEY IF ADDING TO EXISTING PR
    OGRAM" @ DISP
980 DISP @ DISP "ENTER PROGRAM N
    UMBER";
990 INPUT T$
991 IF T$#"" THEN P$=T$
992 IF T$="" THEN 1057
995 IF LEN(P$)>2 THEN 960
```

```
996 FOR I=1 TO LEN(P$)
997 IF NUM(P$[I,I])<48 OR NUM(P$
    [I,I])>57 THEN 970
998 NEXT I
1020 P$="P"&P$
1022 DISP @ DISP "PROGRAM";P$[2]
1025 S$=""
1030 CLEAR @ DISP "ENTER PROGRAM
     USING * FOR EOB   (END OF
     BLOCK)"
1040 DISP @ DISP "PROGRAM CAN BE
     ENTERED 2 LINES  AT ONE TI
     ME"
1050 DISP @ DISP "PRESS END LINE
     KEY TO ENTER     PROGRAM A
     ND EXIT TO MENU"
1055 WAIT 2000 @ GOTO 1060
1057 IF LEN(P$)<1 THEN DISP "NO
     PROGRAM EXISTS" @ WAIT 1000
     @ GOTO 800
1058 DISP @ DISP "PROGRAM ADDITI
     ON"
1059 WAIT 500
1060 CLEAR @ DISP "ENTER PROGRAM
     ";P$[2];
1070 INPUT J$
1075 IF LEN(J$)>1 THEN 1085
1080 IF J$="" THEN 1190
1085 S$=S$&J$
1090 CLEAR @ WAIT 1000 @ DISP "P
     ROGRAM AS ENTERED"
1100 DISP @ DISP J$
1165 DISP @ DISP "PRESS END LINE
     KEY TO CONTINUE";
1170 INPUT T$@ IF T$#"" THEN 116
     5
1180 GOTO 1060
1190 CLEAR @ DISP S$ @ WAIT 2000
1195 IF LEN(S$)<1 THEN P$=""
1196 GOTO 800
1200 ! CHANGE PROGRAM
1210 D=0
1220 L$="         EDIT MODE-CHANG
     E"
1230 GOSUB 8800
1240 IF S$="" THEN 1410
1243 DISP
1244 K=1 @ J$=""
1245 FOR I=K TO LEN(S$)
1250 IF S$[I,I]="*" THEN 1260
1255 NEXT I
1260 DISP S$[K,I] @ DISP
1265 INPUT W$
1270 IF W$="" THEN 1320
1275 FOR J=1 TO LEN(W$)
1280 IF W$[J,J]=S$[K,K] THEN 129
     5
1285 IF J=LEN(W$) THEN 1320
1290 NEXT J
1295 FOR M=LEN(W$) TO J STEP -1
```

```
1300 IF W$[M,M]="*" THEN 1310
1305 NEXT M
1310 J$=J$&W$[J,M]
1315 GOTO 1325
1320 J$=J$&S$[K,I]
1325 IF I=LEN(S$) THEN 1340
1330 K=I+1
1335 GOTO 1245
1340 S$=J$
1345 GOTO 1430
1410 DISP @ DISP "NO PROGRAM EXI
     STS"
1420 WAIT 2000 @ GOTO 800
1430 CLEAR @ DISP S$
1431 DISP "PRESS END LINE KEY TO
      CONTINUE"
1432 INPUT T$
1433 IF T$#"" THEN 1430
1440 GOTO 800
1500 ! STORE PROGRAM ON TAPE
1501 D=0 @ L$="          OUTPUT T
     O TAPE" @ GOSUB 8800
1502 IF LEN(S$)<1 THEN 1542
1503 IF LEN(P$)<3 THEN GOSUB 790
     0
1504 IF P$="00" THEN 800
1505 IF LEN(P$)=2 THEN P$="P"&P$
1507 T=LEN(S$)/100+.5
1508 ON ERROR GOSUB 7800
1509 DISP P$;S$;LEN(S$);T @ WAIT
      500
1510 CREATE P$,1,T*100
1515 ASSIGN# 1 TO P$
1524 PRINT# 1 ; S$
1530 ASSIGN# 1 TO *
1535 OFF ERROR
1537 GOTO 800
1542 DISP @ DISP "NO PROGRAM EXI
     STS"
1543 P$="" @ WAIT 1000
1544 GOTO 800
1550 D=0
1551 L$="      TAPE PROGRAM LIST
     ING"
1552 GOSUB 8800
1553 ON ERROR GOSUB 7800
1555 CAT
1560 DISP @ DISP "     INPUT PRO
     GRAM FROM TAPE"
1570 DISP @ DISP "ENTER PROGRAM
     NUMBER";
1575 ON ERROR GOTO 1550
1580 INPUT P$
1590 IF LEN(P$)>2 THEN 1550
1591 IF LEN(P$)<1 THEN 1550
1595 I=1
1600 FOR I=1 TO LEN(P$)
1610 IF NUM(P$[I,I])<48 THEN 155
     0
1615 IF NUM(P$[I,I])>57 THEN 155
     0
```

```
1620 NEXT I
1625 OFF ERROR
1626 IF P$="00" THEN 800
1630 P$="P"&P$
1640 D=0 @ L$="READING TAPE"
1650 GOSUB 8800
1655 WAIT 500
1656 ON ERROR GOSUB 7800
1660 ASSIGN# 1 TO P$
1670 READ# 1 ; J$
1671 S$=J$
1673 DISP LEN(J$) @ WAIT 100
1680 ASSIGN# 1 TO *
1685 OFF ERROR
1687 P$=""
1690 GOTO 800
1700 D=0 @ L$="" @ GOSUB 8800
1701 IF LEN(S$)<2 THEN DISP "NO
     PROGRAM EXISTS."
1702 WAIT 500
1703 IF LEN(S$)<2 THEN 800
1705 IF LEN(P$)<3 THEN GOSUB 790
     0
1706 IF P$="00" THEN 800
1707 IF LEN(P$)=2 THEN P$="P"&P$
1710 DISP @ DISP "LOADING PROGRA
     M TO UNIDEX III"
1715 J$=S$
1717 OUTPUT N ;"E$"&P$[2,3]
1720 OUTPUT N ;"E"&P$[2,3]
1730 OUTPUT N ;J$
1735 OUTPUT N ;"%"
1760 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
1770 GOTO 800
1800 D=0 @ L$="LOADING DIRECTORY
      FROM           UNIDEX III
     "
1810 GOSUB 8800
1820 OUTPUT N ;"PD"
1830 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
1835 IOBUFFER J$
1840 TRANSFER N TO J$ INTR
1846 ON EOT 7 GOTO 1847
1847 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
1850 OFF EOT 7
1860 DISP J$[11,38]
1865 DISP
1866 K=0
1870 J$=J$[52]
1875 IF LEN(J$)<30 THEN 1950
1880 FOR I=1 TO 100
1890 DISP J$[1,12]
1900 DISP J$[16,34]
1905 DISP
1906 K=1
1910 J$=J$[47]
1920 IF LEN(J$)<30 THEN 1950
```

```
1940 NEXT I
1950 DISP J$[1,18]
1960 DISP @ DISP "PRESS END LINE
      TO CONTINUE";
1970 ON TIMER# 1,10000 GOTO 2000
1980 INPUT T$
1990 IF T$#"" THEN 1980
2000 OFF TIMER# 1
2005 IF K=0 THEN 2275
2006 J$=""
2007 D=0
2010 GOSUB 8800
2020 DISP @ DISP " INPUT PROGRAM
      FROM UNIDEX III"
2030 DISP @ DISP "ENTER PROGRAM
     NUMBER";
2040 INPUT P$
2050 IF LEN(P$)>2 THEN 2010
2060 I=1
2070 FOR I=1 TO LEN(P$)
2080 IF NUM(P$[I,I])<48 THEN 201
     0
2090 IF NUM(P$[I,I])>57 THEN 201
     0
2100 NEXT I
2105 IF P$="00" THEN 800
2110 P$="P"&P$
2120 GOSUB 8800
2130 DISP @ DISP "INPUTTING PROG
     RAM INTO MEMORY"
2140 OUTPUT N ;P$
2150 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
2152 IOBUFFER J$
2154 TRANSFER N TO J$ INTR
2156 ON EOT 7 GOTO 2160
2158 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
2160 OFF EOT 7
2165 J$=J$[97,LEN(J$)-7] @ T=1
2167 FOR I=T TO LEN(J$)
2168 IF J$[I,I]="↑" THEN 2171
2169 IF I=LEN(J$) THEN 2173
2170 NEXT I
2171 J$=J$[1,I-2]&J$[I+10]
2172 T=I-1 @ GOTO 2167
2173 T=1
2174 FOR I=T TO LEN(J$)
2175 IF J$[I,I]=CHR$(34) THEN 21
     79
2176 IF J$[I,I]=CHR$(32) THEN 21
     85
2177 IF I=LEN(J$) THEN 2187
2178 NEXT I
2179 I=I+1
2180 FOR K=I TO LEN(J$)
2181 IF J$[K,K]=CHR$(34) THEN 21
     83
2182 NEXT K
2183 I=K
```

```
2184 GOTO 2178
2185 J$=J$[1,I-1]&J$[I+1]
2186 T=I @ GOTO 2174
2187 J$=J$[1,LEN(J$)-1]
2189 DISP "RECEIVED PROGRAM" @ W
     AIT 1000
2195 IF LEN(S$)#LEN(J$) THEN 227
     0
2200 FOR I=1 TO LEN(J$)
2210 IF J$[I,I]#S$[I,I] THEN 226
     0
2220 NEXT I
2230 DISP @ DISP "UNIDEX PROGRAM
      VERIFIES WITH    MEMORY"
2250 GOTO 2270
2260 DISP @ DISP "UNIDEX PROGRAM
      DOES NOT MATCH  MEMORY"
2270 S$=J$ @ WAIT 1000 @ GOTO 80
     0
2275 DISP @ DISP "NO PROGRAMS IN
      UNIDEX III" @ WAIT 2000 @
     K=0
2278 P$=""
2280 GOTO 800
2300 D=0 @ L$="" @ GOSUB 8800
2310 DISP @ DISP "          DELET
     E PROGRAM"
2320 DISP @ DISP "     (1) FROM
     TAPE"
2330 DISP "     (2) FROM MEMORY"
2340 DISP "     (3) FROM UNIDEX
     III"
2345 DISP "     (4) GO TO MENU"
2350 ON ERROR GOTO 2300
2360 INPUT T$
2370 IF LEN(T$)>1 THEN 2300
2380 IF NUM(T$)<49 OR NUM(T$)>52
      THEN 2300
2390 OFF ERROR
2400 T=VAL(T$)
2410 ON T GOTO 2600,2500,2420,26
     70
2420 GOSUB 7900
2450 OUTPUT N ;"E$"&P$
2460 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
2470 DISP "PROGRAM CLEARED"
2480 WAIT 1000
2485 P$=""
2490 GOTO 2300
2500 T$=P$
2502 IF T$="" THEN 2595
2505 DISP "PROGRAM ";T$
2510 GOSUB 7900
2520 IF P$#"00" THEN 2540
2530 GOTO 2300
2540 IF T$[2]=P$ THEN 2560
2550 DISP "NOT SAME PROGRAM IS I
     N MEMORY" @ WAIT 1000 @ GOT
     O 2300
```

```
2560 S$="" @ J$="" @ P$="" @ T$=
     ""
2570 DISP "MEMORY CLEARED"
2580 WAIT 1000
2590 GOTO 2300
2595 DISP @ DISP "NO PROGRAM EXI
     STS" @ WAIT 1000 @ GOTO 230
     0
2600 GOSUB 7900
2610 IF P$="00" THEN 2300
2620 P$="P"&P$
2625 ON ERROR GOSUB 7800
2630 PURGE P$
2635 OFF ERROR
2640 DISP "PROGRAM ERASED FROM T
     APE"
2650 WAIT 1000
2655 P$=""
2660 GOTO 2300
2670 GOTO 200
2700 ! SINGLE MODE
2705 A=0
2710 D=0
2720 L$="          SINGLE MODE"
2730 GOSUB 8800
2740 GOSUB 7900
2750 IF P$="00" THEN 2700
2755 GOSUB 5000
2760 OUTPUT N ;"S"&P$
2770 GOSUB 8070
2775 IF BIT(S1,7)=1 THEN 200
2780 D=0
2790 GOSUB 8800
2800 DISP @ DISP "     (1) NEXT
     BLOCK"
2810 DISP "     (2) GO TO MENU"
2820 ON ERROR GOTO 2780
2830 INPUT T$
2840 IF LEN(T$)>1 THEN 2780
2850 IF NUM(T$)<49 OR NUM(T$)>51
      THEN 2780
2860 OFF ERROR
2870 T=VAL(T$)
2880 ON T GOTO 2890,2920
2890 OUTPUT N
2895 D=0
2900 GOSUB 8070
2910 GOTO 2780
2920 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
2925 IF BIT(S1,7)=1 THEN GOSUB 8
     120
2930 GOTO 200
3000 ! AUTO MODE
3010 A=1 @ D=0
3020 L$="              AUTO MODE"
3030 GOSUB 8800
3040 GOSUB 7900
3041 IF P$="00" THEN 200
3045 GOSUB 5000
```

```
3051 DISP @ DISP "DO YOU WISH TO
      TRIGGER          UNIDEX II
     I"
3052 DISP @ DISP "(Y) YES OR (N)
      NO";
3053 ON ERROR GOTO 3000
3054 INPUT T$
3055 IF LEN(T$)>1 THEN 3000
3056 IF T$="N" THEN 3060
3057 IF T$="Y" THEN 3059
3058 OFF ERROR @ GOTO 3000
3059 OFF ERROR @ P$=P$&"H"
3060 OUTPUT N ;"A"&P$
3065 D=0
3070 GOSUB 8070
3072 IF BIT(S1,7)=1 THEN 3410
3300 D=0
3310 GOSUB 8800
3320 DISP @ DISP "     (1) GO TO
      MENU"
3330 DISP "     (2) GO AGAIN"
3340 GOSUB 8700
3350 ON ERROR GOTO 3300
3360 INPUT T$
3370 IF LEN(T$)>1 THEN 3300
3380 IF NUM(T$)=49 THEN 3410
3390 IF NUM(T$)=50 THEN G=0
3395 IF NUM(T$)=50 THEN 3060
3400 GOTO 3300
3410 IF BIT(S1,2)=1 THEN OUTPUT
     N ;"O"
3415 A=0
3420 GOTO 200
3500 D=0
3510 L$="          HARDWARE TEST
     "
3520 GOSUB 8800
3530 OUTPUT N ;"V"
3540 ON INTR 7 GOTO 3580
3550 ENABLE INTR 7;8
3560 ON TIMER# 1,10000 GOTO 8900
3570 GOTO 3570
3580 OFF TIMER# 1
3590 S1=SPOLL(N) @ STATUS 7,1 ;
     C0@ OFF INTR 7
3600 IF S1<=192 THEN 3700
3610 DISP @ DISP "UNIDEX HAS A H
     ARDWARE FAILURE!"
3620 DISP "FAILURE(S) IS(ARE) AS
      FOLLOWS:"
3630 IF BIT(S1,3)=1 THEN DISP "U
     SER MEMORY ALTERED."
3640 IF BIT(S1,2)=1 THEN DISP "U
     SER MEMORY READ/WRITE FAILU
     RE."
3650 IF BIT(S1,1)=1 THEN DISP "S
     YSTEM RAM READ/WRITE FAILUR
     E."
3660 IF BIT(S1,0)=1 THEN DISP "U
     NIDEX III SOFTWARE FAILURE.
     "
```

```
3670 DISP @ DISP "CALL OR WRITE
     FACTORY FOR ASSISTANCE."
3680 S6=0 @ GOSUB 8555
3690 GOTO 3710
3700 DISP @ DISP "UNIDEX III HAR
     DWARE TEST PASSES."
3710 WAIT 5000 @ R=1
3720 GOTO 200
4980 D=0
4990 GOSUB 8800
5000 DISP @ DISP "DISPLAY M-FUNC
     TION OUTPUT AND   C-INPUT C
     ONDITION STATUS."
5010 DISP @ DISP "(Y) YES OR (N)
      NO";
5020 INPUT T$
5030 IF T$="Y" THEN 5070
5040 IF T$="N" THEN 5060
5050 GOTO 4980
5060 M=0 @ GOTO 5080
5070 M=1
5080 D=0
5090 GOSUB 8800
5095 RETURN
5100 D=0
5110 GOSUB 8800
5120 DISP "    M-FUNCTION OUTPUT
      STATUS"
5130 DISP @ DISP "      M     CO
     NDITION"
5140 FOR I=8 TO 1 STEP -1
5150 DISP "     ";I;"        ";
5160 IF BIT(S8,I-1)=1 THEN DISP
     "ON"
5170 IF BIT(S8,I-1)=0 THEN DISP
     "OFF"
5180 WAIT 100
5190 NEXT I
5200 WAIT 2000
5210 D=0
5220 GOSUB 8800
5230 DISP "          C-INPUT STAT
     US"
5240 DISP @ DISP "      C-INPUT
          STATUS"
5250 DISP
5260 FOR I=4 TO 1 STEP -1
5270 DISP "         ";I;"
       ";
5280 IF BIT(S9,I+3)=1 THEN DISP
     "HI"
5290 IF BIT(S9,I+3)=0 THEN DISP
     "LO"
5300 WAIT 100
5310 NEXT I
5320 WAIT 2000
5330 RETURN
7000 IF A=0 AND G=1 THEN 7020
7001 IF A=0 THEN 7020
7002 IF D=1 THEN 7020
```

```
7005 GOSUB 8800
7020 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
7025 G=0
7030 IF BIT(S1,0)=1 THEN DISP @
     DISP "IN 'M00' STOP (PROGRA
     M USES C-  INPUT TO CONTINU
     E.)"
7040 IF BIT(S1,0)=0 THEN 7075
7050 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
7055 D=0
7060 IF BIT(S1,0)=0 THEN 8070
7070 GOTO 7060
7075 IF S1=0 THEN 8095
7080 IF S1#4 THEN 7020
7090 DISP @ DISP "IN 'HOLD' MODE
     ! SEND TRIGGER."
7100 DISP @ DISP "PRESS ENDLINE
     KEY TO CONTINUE."
7110 INPUT T$
7120 IF T$="" THEN 7140
7130 GOTO 7000
7140 TRIGGER N
7145 D=0
7150 GOTO 8070
7800 DISP "HP-85 TAPE ERROR IS "
     ;ERRN
7810 WAIT 2000
7830 RETURN
7900 ON ERROR GOTO 7900
7905 DISP @ DISP "ENTER PROGRAM
     NUMBER";
7910 INPUT P$
7920 IF LEN(P$)>2 THEN 7900
7930 I=1
7940 FOR I=1 TO LEN(P$)
7950 IF NUM(P$[I,I])<48 THEN 790
     0
7960 IF NUM(P$[I,I])>57 THEN 790
     0
7970 NEXT I
7980 OFF ERROR
7990 RETURN
8000 ! OUTPUT TO UNIDEX III
8010 D=0
8020 GOSUB 8800
8030 DISP @ DISP "OUTPUT IS"
8040 DISP @ DISP S$
8050 J$=A$&S$
8060 OUTPUT N ;J$
8065 J$=S$
8070 GOSUB 8800
8079 ON INTR 7 GOTO 8100
8080 ENABLE INTR 7;8
8085 ON TIMER# 1,900000 GOTO 890
     0
8090 GOTO 7000
8095 OFF TIMER# 1
8100 S1=SPOLL(N) @ STATUS 7,1 ;
     C0@ OFF INTR 7
```

```
8101 IF BIT(S1,1)=0 THEN 8109
8102 IF D=1 THEN 8104
8103 GOSUB 8800
8104 DISP @ DISP "          IN 'G
     25' STOP"
8105 DISP @ DISP "PRESS ENDLINE
     KEY TO CONTINUE."
8106 INPUT T$
8107 IF T$="" THEN 8109
8108 GOTO 8105
8109 DISP @ DISP "SPOLL= ";DTH$(
     S1)
8110 OUTPUT N ;"PS"
8115 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
8120 ENTER N USING "10(B)" ; S,S
     1,S2,S3,S4,S5,S6,S7,S8,S9
8130 OUTPUT N ;"PX"
8135 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
8140 ENTER N USING "X,K" ; X
8150 OUTPUT N ;"PY"
8155 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
8160 ENTER N USING "X,K" ; Y
8170 DISP "SPOLL= ";DTH$(S1)
8180 DISP "X POSITION IS    ";X
8190 DISP "Y POSITION IS    ";Y
8195 WAIT 1500
8196 IF M=1 AND BIT(S1,1)=1 THEN
      GOSUB 5100
8200 IF BIT(S1,7)=1 THEN 8300
8205 IF BIT(S1,1)=1 THEN 8220
8210 RETURN
8220 OUTPUT N
8225 S1=SPOLL(N) @ STATUS 7,1 ;
     C0
8230 IF BIT(S1,1)=1 THEN OUTPUT
     N ;"B"
8235 D=0
8240 GOTO 8070
8300 D=0 @ GOSUB 8800
8310 DISP @ DISP "UNIDEX III HAS
      DETECTED AN ERROR IN PROGR
     AM"
8320 DISP @ DISP "THE ERROR(S) I
     S(ARE) AS FOLLOWS:"
8330 IF S5=0 THEN DISP "-NO SYNT
     AX ERRORS" @ GOTO 8400
8340 IF S5>0 THEN DISP "-SYNTAX
     ERROR-"
8350 IF BIT(S5,6)=1 THEN DISP "
            ILLEGAL CHARACTER"
8360 IF BIT(S5,5)=1 THEN DISP "
          'N' COMMAND"
8370 IF BIT(S5,4)=1 THEN DISP "
          'G' COMMAND"
8375 IF BIT(S5,3)=1 THEN DISP "
          'F' COMMAND"
8380 IF BIT(S5,2)=1 THEN DISP "
          'M' COMMAND"
```

```
8385 IF BIT(S5,1)=1 THEN DISP "
          'D' COMMAND"
8390 IF BIT(S5,0)=1 THEN DISP "
          NO COMMAND BEFORE INP
     UTT-        ING NUMERALS"
8400 IF S6=0 THEN DISP "-NO RUN
     ERRORS" @ GOTO 8470
8410 IF S6>0 THEN DISP "-RUN ERR
     OR-"
8420 IF BIT(S6,6)=1 THEN DISP "
          X AXIS LIMIT"
8430 IF BIT(S6,5)=1 THEN DISP "
          Y AXIS LIMIT"
8440 IF BIT(S6,3)=1 THEN DISP "
          STACK OVERFLOW"
8450 IF BIT(S6,1)=1 THEN DISP "
          EOB(END OF BLOCK) SEA
     RCH"
8460 IF BIT(S6,0)=1 THEN DISP "
          PROGRAM MISSING"
8470 IF BIT(S7,7)=0 THEN DISP "-
     NO EDITOR ERRORS"
8480 IF BIT(S7,7)=1 THEN DISP "-
     EDITOR ERROR-"
8490 IF BIT(S7,0)=1 THEN DISP "
          COMPILE"
8500 DISP @ DISP "PRESS END LINE
      KEY TO CONTINUE" @ ON ERRO
     R GOTO 8300 @ INPUT T$@ OFF
      ERROR
8501 IF T$#"" THEN 8300
8510 IF BIT(S6,6)=1 OR BIT(S6,5)
     =1 THEN 8530
8520 RETURN
8530 S$="G7*@"
8540 GOSUB 8000
8550 S$=CHR$(34)&"  HOME  "&CHR$
     (34)&"D5000*M2*"
8555 GOSUB 8000
8560 RETURN
8600 WAIT 100 @ D=0
8610 GOSUB 8800
8615 DISP @ DISP
8620 DISP "      (1) GO TO MENU"
8630 DISP "      (2) GO AGAIN"
8640 DISP "      (3) CHANGE"
8645 GOSUB 8700
8646 ON TIMER# 1,900000 GOTO 869
     1
8650 INPUT T$
8655 OFF TIMER# 1
8660 IF LEN(T$)>1 THEN 8600
8670 IF NUM(T$)<49 OR NUM(T$)>51
      THEN 8600
8680 T=VAL(T$)
8690 RETURN
8691 T=1 @ RETURN
8700 DISP @ DISP "  PRESS KEY TO
      ENTER";
8710 RETURN
```

```
8800 IF D=1 THEN RETURN
8801 CLEAR
8805 DISP "    UNIDEX IIIa TEST
     PROGRAM"
8810 DISP @ DISP L$
8815 D=1
8820 RETURN
8900 ! CONTINUE TIMEOUT
8905 OFF TIMER# 1
8910 ABORTIO 7
8915 D=0 @ GOSUB 8800
8920 DISP @ DISP "SET UNIDEX III
      TO REMOTE, THEN  PRESS END
      LINE KEY"
8930 INPUT N$
8940 IF N$#"" THEN 8915
8950 RESET 7
8960 REMOTE N
8970 SEND 7 ; LISTEN N-700
8980 GOTO 100
9900 CLEAR @ DISP @ DISP @ DISP
     @ DISP @ DISP "
      END OF"
9910 DISP @ DISP @ DISP "    UNI
     DEX IIIa TEST PROGRAM" @ EN
     D
```

RS-232 TEST PROGRAM

```
 10 ! UNIDEX III/RS232 PROGRAM
 20 ! FEB 1986
 30 OPTION BASE 1
 40 DIM S$[4104],J$[4096],L$[40]
    ,P$[3],W$[80]
 42 S$="" @ J$="" @ P$="" @ R=0
    @ A=0
 45 INTEGER T
 50 RESET 10
 55 RESUME 10
 60 SET TIMEOUT 10;5000
 70 ON TIMEOUT 10 GOTO 8910
 80 ENABLE KBD 5 ! KBD INPUTS ON
    LY
100 GOSUB 8800
102 DISP @ DISP "    ENTER DEVIC
    E ADDRESS (XX)";
105 ON ERROR GOTO 100
110 INPUT Q$
111 OFF ERROR
115 IF Q$="" THEN 100
120 IF LEN(Q$)>2 THEN 100
121 FOR I=1 TO LEN(Q$)
122 IF NUM(Q$[I,I])<48 OR NUM(Q$
    [I,I])>57 THEN 100
123 NEXT I
130 IF VAL(Q$)<1 OR VAL(Q$)>31 T
    HEN 100
140 Q$="E"&Q$
150 GOTO 8955
170 IF R#1 THEN 3500
200 GOSUB 8800
210 DISP @ DISP @ DISP "     (1)
     EDIT MODE"
220 DISP "      (2) SINGLE MODE"
230 DISP "      (3) AUTO MODE"
240 DISP "      (4) IMMEDIATE MOD
    E"
250 DISP "      (5) MESSAGE TEST"
260 DISP "      (6) HARDWARE TEST
    "
262 DISP "      (7) END PROGRAM"
265 GOSUB 8700
269 ON ERROR GOTO 200
270 INPUT K$
280 IF LEN(K$)>1 THEN 200
290 K=NUM(K$)-48
300 IF K<1 OR K>7 THEN 200
305 OFF ERROR
310 ON K GOTO 800,2700,3000,360,
    550,3500,9900
360 OUTPUT 10 ;"PS"
370 ENTER 10 USING "#10A" ; J$
380 S=NUM(J$[1,1])
420 IF BIT(S,7)=0 THEN F$="F1000
    0"
430 IF BIT(S,7)=1 AND BIT(S,6)=1
     THEN F$="F3000"
440 IF BIT(S,7)=1 AND BIT(S,6)=0
     THEN F$="F400"
```

```
450 J$="G7*D1000X1000"&F$&"Y1000
    "&F$&"X-1000Y-1000*@"
451 GOSUB 8800
452 L$="          IMMEDIATE MODE"
453 DISP @ DISP L$
454 DISP @ DISP "ENTER COMMAND S
    TRING OF LESS    THAN 64 BYT
    ES"
455 DISP @ DISP " USE-  * FOR EO
    B(END OF BLOCK)"
456 DISP "       PRESS END LINE
    TO RUN"
457 IF LEN(J$)<13 THEN J$=S$
460 DISP @ DISP J$;
470 INPUT S$
480 IF S$="" THEN S$=J$ @ GOTO 4
    90
481 IF LEN(S$)>64 THEN DISP "COM
    MAND STRING IS LONGER THAN 6
    4 BYTES" @ WAIT 500 @ GOTO 4
    51
490 A$="I"
500 GOSUB 8000
510 GOSUB 8600
520 IF T=1 THEN 200
530 IF T=2 THEN 500
540 GOTO 451
550 GOSUB 8800
555 P$="P10"
560 L$="           MESSAGE TEST"
570 DISP @ DISP L$
580 DISP @ DISP "PLEASE ENTER YO
    UR NAME";
585 S$=""
590 ON ERROR GOTO 550
600 INPUT N$
610 OFF ERROR
615 IF N$="" THEN N$="YOU"
620 J$="        AEROTECH WELCOME
    S "&N$&" TO THE WORLD OF UNI
    DEX IIIA!!!!              "
630 FOR I=1 TO LEN(J$)-8
640 A$=CHR$(34)&J$[I,I+7]&CHR$(3
    4)&"D200*"
650 S$=S$&A$
660 NEXT I
675 OUTPUT 10 ;"E$10"
680 OUTPUT 10 ;"E10"
682 OUTPUT 10 ;S$
688 OUTPUT 10 ;"A10"
690 GOSUB 8070
700 GOSUB 8600
710 IF T<1 OR T>3 THEN 700
720 IF T=1 THEN 200
730 IF T=2 THEN 688
740 GOSUB 8800
750 DISP @ DISP L$
760 DISP @ DISP "ENTER MESSAGE"
761 DISP "(DO NOT USE QUOTATION
    MARKS)"
```

```
765 DISP "(DO NOT USE COMMAS)"
770 INPUT S$
780 J$="          "&S$&"
    "
785 S$=""
790 GOTO 630
800 GOSUB 8800
810 L$="              EDIT MODE"
820 DISP @ DISP L$
830 DISP @ DISP "      (1) LOAD P
    ROGRAM FROM TAPE"
840 DISP "      (2) STORE PROGRAM
     TO TAPE"
850 DISP "      (3) LOAD PROGRAM
    FROM                  UNIDEX II
    I"
860 DISP "      (4) STORE PROGRAM
     TO                   UNIDEX II
    I"
870 DISP "      (5) CREATE PROGRA
    M"
880 DISP "      (6) DELETE PROGRA
    M"
890 DISP "      (7) CHANGE PROGRA
    M"
895 DISP "      (8) GO TO MENU"
900 GOSUB 8700
905 ON ERROR GOTO 800
910 INPUT T$
920 IF LEN(T$)>1 THEN 800
930 IF NUM(T$)<49 OR NUM(T$)>56
    THEN 800
935 OFF ERROR
940 T=VAL(T$)
950 ON T GOTO 1550,1500,1800,170
    0,960,2300,1200,200
960 ! CREATE OR ADD PROGRAM
965 GOSUB 8800
970 DISP @ DISP L$
975 DISP @ DISP "PRESS END LINE
    KEY IF ADDING TO EXISTING PR
    OGRAM" @ DISP
980 DISP @ DISP "ENTER PROGRAM N
    UMBER";
990 INPUT T$
991 IF T$#"" THEN P$=T$
992 IF T$="" THEN 1057
995 IF LEN(P$)>2 THEN 960
996 FOR I=1 TO LEN(P$)
997 IF NUM(P$[I,I])<48 OR NUM(P$
    [I,I])>57 THEN 970
998 NEXT I
1020 P$="P"&P$
1022 DISP @ DISP "PROGRAM";P$[2]
1025 S$=""
1030 CLEAR @ DISP "ENTER PROGRAM
      USING * FOR EOB   (END OF
     BLOCK)"
1040 DISP @ DISP "PROGRAM CAN BE
      ENTERED 2 LINES  AT ONE TI
     ME"
```

```
1050 DISP @ DISP "PRESS END LINE
      KEY TO ENTER     PROGRAM A
     ND EXIT TO MENU"
1055 WAIT 2000 @ GOTO 1060
1057 IF LEN(P$)<1 THEN DISP "NO
     PROGRAM EXISTS" @ WAIT 1000
      @ GOTO 800
1058 DISP @ DISP "PROGRAM ADDITI
     ON"
1059 WAIT 500
1060 CLEAR @ DISP "ENTER PROGRAM
     ";P$[2];
1070 INPUT J$
1075 IF LEN(J$)>1 THEN 1085
1080 IF J$="" THEN 1190
1085 S$=S$&J$
1090 CLEAR @ WAIT 1000 @ DISP "P
     ROGRAM AS ENTERED"
1100 DISP @ DISP J$
1165 DISP @ DISP "PRESS END LINE
      KEY TO CONTINUE";
1170 INPUT T$@ IF T$#"" THEN 116
     5
1180 GOTO 1060
1190 CLEAR @ DISP S$ @ WAIT 2000
1195 IF LEN(S$)<1 THEN P$=""
1196 GOTO 800
1200 ! CHANGE PROGRAM
1210 GOSUB S800
1220 L$="         EDIT MODE-CHANG
     E"
1230 DISP @ DISP L$
1240 IF S$="" THEN 1410
1243 DISP
1244 K=1 @ J$=""
1245 FOR I=K TO LEN(S$)
1250 IF S$[I,I]="*" THEN 1260
1255 NEXT I
1260 DISP S$[K,I] @ DISP
1265 INPUT W$
1270 IF W$="" THEN 1320
1275 FOR J=1 TO LEN(W$)
1280 IF W$[J,J]=S$[K,K] THEN 129
     5
1285 IF J=LEN(W$) THEN 1320
1290 NEXT J
1295 FOR M=LEN(W$) TO J STEP -1
1300 IF W$[M,M]="*" THEN 1310
1305 NEXT M
1310 J$=J$&W$[J,M]
1315 GOTO 1325
1320 J$=J$&S$[K,I]
1325 IF I=LEN(S$) THEN 1340
1330 K=I+1
1335 GOTO 1245
1340 S$=J$
1350 GOTO 1430
1410 DISP @ DISP "NO PROGRAM EXI
     STS"
1420 WAIT 2000 @ GOTO 800
```

```
1430 CLEAR @ DISP S$ @ DISP
1431 DISP "PRESS END LINE KEY TO
      CONTINUE"
1432 INPUT T$
1433 IF T$#"" THEN 1430
1440 GOTO 800
1500 ! STORE PROGRAM ON TAPE
1501 GOSUB 8800 @ DISP @ DISP "
           OUTPUT TO TAPE" @ D
     ISP
1502 IF LEN(S$)<1 THEN 1542
1503 IF LEN(P$)<3 THEN GOSUB 790
     0
1504 IF P$="00" THEN 800
1505 IF LEN(P$)=2 THEN P$="P"&P$
1507 T=LEN(S$)/100+.5
1508 ON ERROR GOSUB 7800
1509 DISP P$;S$;LEN(S$);T @ WAIT
      1
1510 CREATE P$,1,T*100
1515 ASSIGN# 1 TO P$
1524 PRINT# 1 ; S$
1530 ASSIGN# 1 TO *
1535 OFF ERROR
1537 GOTO 800
1542 DISP @ DISP "NO PROGRAM EXI
     STS"
1543 P$="" @ WAIT 1000
1544 GOTO 800
1550 GOSUB 8800
1551 DISP @ DISP "      TAPE PRO
     GRAM LISTING:"
1552 DISP
1553 ON ERROR GOSUB 7800
1555 CAT
1560 DISP @ DISP "     INPUT PRO
     GRAM FROM TAPE"
1570 DISP @ DISP "ENTER PROGRAM
     NUMBER";
1575 ON ERROR GOTO 1550
1580 INPUT P$
1590 IF LEN(P$)>2 THEN 1550
1591 IF LEN(P$)<1 THEN 1550
1595 I=1
1600 FOR I=1 TO LEN(P$)
1610 IF NUM(P$[I,I])<48 THEN 155
     0
1615 IF NUM(P$[I,I])>57 THEN 155
     0
1620 NEXT I
1625 OFF ERROR
1626 IF P$="00" THEN 800
1630 P$="P"&P$
1640 CLEAR @ GOSUB 8800
1650 DISP @ DISP "READING TAPE"
1655 WAIT 500
1656 ON ERROR GOSUB 7800
1660 ASSIGN# 1 TO P$
1670 READ# 1 ; J$
1671 S$=J$
```

```
1673 DISP LEN(J$) @ WAIT 100
1680 ASSIGN# 1 TO *
1685 OFF ERROR
1690 GOTO 800
1700 GOSUB 8800
1701 IF LEN(S$)<2 THEN DISP "NO
     PROGRAM EXISTS."
1702 WAIT 500
1703 IF LEN(S$)<2 THEN 800
1705 IF LEN(P$)<3 THEN GOSUB 790
     0
1706 IF P$="00" THEN 800
1707 IF LEN(P$)=2 THEN P$="P"&P$
1710 DISP @ DISP "LOADING PROGRA
     M TO UNIDEX III"
1715 J$=S$
1717 OUTPUT 10 ;"E$"&P$[2,3]
1720 OUTPUT 10 ;"E"&P$[2,3]
1730 OUTPUT 10 ;J$
1735 OUTPUT 10 ;"%"
1740 GOSUB 9000
1770 GOTO 800
1800 GOSUB 8800
1810 DISP @ DISP "LOADING DIRECT
     ORY FROM           UNIDEX II
     I"
1815 WAIT 500 @ CLEAR
1820 OUTPUT 10 ;"PD"
1840 ENTER 10 USING "%,#%K" ; J$
1860 DISP J$[11,38]
1865 DISP
1866 K=0
1870 J$=J$[52]
1875 IF LEN(J$)<30 THEN 1950
1880 FOR I=1 TO 100
1890 DISP J$[1,12]
1900 DISP J$[16,34]
1905 DISP
1906 K=1
1910 J$=J$[47]
1920 IF LEN(J$)<30 THEN 1950
1940 NEXT I
1950 DISP J$[1,18]
1960 DISP @ DISP "PRESS END LINE
      TO CONTINUE";
1970 ON TIMER# 1,10000 GOTO 2000
1980 INPUT T$
1990 IF T$#"" THEN 1980
2000 OFF TIMER# 1
2005 IF K=0 THEN 2275
2006 J$=""
2010 GOSUB 8800
2020 DISP @ DISP " INPUT PROGRAM
      FROM UNIDEX III"
2030 DISP @ DISP "ENTER PROGRAM
     NUMBER";
2040 INPUT P$
2050 IF LEN(P$)>2 THEN 2010
2060 I=1
2070 FOR I=1 TO LEN(P$)
```

```
2080 IF NUM(P$[I,I])<48 THEN 201
     0
2090 IF NUM(P$[I,I])>57 THEN 201
     0
2100 NEXT I
2105 IF P$="00" THEN 800
2110 P$="P"&P$
2120 GOSUB 8800
2130 DISP @ DISP "INPUTTING PROG
     RAM INTO MEMORY"
2140 OUTPUT 10 ;P$
2150 ENTER 10 USING "%,#%K" ; J$
2165 J$=J$[97,LEN(J$)-7] @ T=1
2167 FOR I=T TO LEN(J$)
2168 IF J$[I,I]="↑" THEN 2171
2169 IF I=LEN(J$) THEN 2173
2170 NEXT I
2171 J$=J$[1,I-2]&J$[I+10]
2172 T=I-1 @ GOTO 2167
2173 T=1
2174 FOR I=T TO LEN(J$)
2175 IF J$[I,I]=CHR$(34) THEN 21
     79
2176 IF J$[I,I]=CHR$(32) THEN 21
     85
2177 IF I=LEN(J$) THEN 2187
2178 NEXT I
2179 I=I+1
2180 FOR K=I TO LEN(J$)
2181 IF J$[K,K]=CHR$(34) THEN 21
     83
2182 NEXT K
2183 I=K
2184 GOTO 2178
2185 J$=J$[1,I-1]&J$[I+1]
2186 T=I @ GOTO 2174
2187 J$=J$[1,LEN(J$)-1]
2189 DISP "RECEIVED PROGRAM" @ W
     AIT 1000
2195 IF LEN(S$)#LEN(J$) THEN 227
     0
2200 FOR I=1 TO LEN(J$)
2210 IF J$[I,I]#S$[I,I] THEN 226
     0
2220 NEXT I
2230 DISP @ DISP "UNIDEX PROGRAM
      VERIFIES WITH    MEMORY"
2250 GOTO 2270
2260 DISP @ DISP "UNIDEX PROGRAM
      DOES NOT MATCH  MEMORY"
2270 S$=J$ @ WAIT 1000 @ GOTO 80
     0
2275 DISP @ DISP "NO PROGRAMS IN
      UNIDEX III" @ WAIT 2000 @
     K=0
2278 P$=""
2280 GOTO 900
2300 GOSUB 8800
2310 DISP @ DISP "          DELET
     E PROGRAM"
```

```
2320 DISP @ DISP "     (1) FROM
     TAPE"
2330 DISP "      (2) FROM MEMORY"
2340 DISP "      (3) FROM UNIDEX
     III"
2345 DISP "      (4) GO TO MENU"
2350 ON ERROR GOTO 2300
2360 INPUT T$
2370 IF LEN(T$)>1 THEN 2300
2380 IF NUM(T$)<49 OR NUM(T$)>52
      THEN 2300
2390 OFF ERROR
2400 T=VAL(T$)
2410 ON T GOTO 2600,2500,2420,26
     70
2420 GOSUB 7900
2450 OUTPUT 10 ;"E$"&P$
2470 DISP "PROGRAM CLEARED"
2480 WAIT 1000
2485 P$=""
2490 GOTO 2300
2500 T$=P$
2502 IF T$="" THEN 2595
2505 DISP "PROGRAM ";T$
2510 GOSUB 7900
2520 IF P$#"00" THEN 2540
2530 GOTO 2300
2540 IF T$[2]=P$ THEN 2560
2550 DISP "NOT SAME PROGRAM IS I
     N MEMORY" @ WAIT 1000 @ GOT
     O 2300
2560 S$="" @ J$="" @ P$="" @ T$=
     ""
2570 DISP "MEMORY CLEARED"
2580 WAIT 1000
2590 GOTO 2300
2595 DISP @ DISP "NO PROGRAM EXI
     STS" @ WAIT 1000 @ GOTO 230
     0
2600 GOSUB 7900
2610 IF P$="00" THEN 2300
2620 P$="P"&P$
2625 ON ERROR GOSUB 7800
2630 PURGE P$
2635 OFF ERROR
2640 DISP "PROGRAM ERASED FROM T
     APE"
2650 WAIT 1000
2655 P$=""
2660 GOTO 2300
2670 GOTO 200
2700 ! SINGLE MODE
2704 GOSUB 9000
2705 A=0
2710 GOSUB 8800
2720 L$="          SINGLE MODE"
2730 DISP @ DISP L$
2740 GOSUB 7900
2750 IF P$="00" THEN 2700
2755 GOSUB 5000
```

```
2760 OUTPUT 10 ;"S"&P$
2765 ON TIMER# 1,900000 GOTO 890
     0
2770 GOSUB 9100
2772 OFF TIMER# 1
2773 GOSUB 8390
2775 IF BIT(S1,7)=1 THEN 200
2780 GOSUB 8800
2790 DISP @ DISP L$
2800 DISP @ DISP "      (1) NEXT
     BLOCK"
2810 DISP "      (2) GO TO MENU"
2820 ON ERROR GOTO 2780
2830 INPUT T$
2840 IF LEN(T$)>1 THEN 2780
2850 IF NUM(T$)<49 OR NUM(T$)>51
      THEN 2780
2860 OFF ERROR
2870 T=VAL(T$)
2880 ON T GOTO 2890,2920
2890 OUTPUT 10
2900 GOSUB 9100
2905 OFF TIMER# 1
2906 GOSUB 8390
2910 GOTO 2780
2920 IF BIT(S1,2)=1 THEN OUTPUT
     10 ;"O"
2930 GOTO 200
3000 ! AUTO MODE
3004 GOSUB 9000
3005 A=1
3010 GOSUB 8800
3020 L$="              AUTO MODE"
3030 DISP @ DISP L$
3040 GOSUB 7900
3045 IF P$="00" THEN 3000
3046 GOSUB 5000
3049 GOSUB 8800
3050 DISP @ DISP L$ @ DISP
3051 DISP @ DISP "DO YOU WISH TO
      TRIGGER          UNIDEX II
     I"
3052 DISP @ DISP "(Y) YES OR (N)
      NO";
3053 ON ERROR GOTO 3000
3054 INPUT T$
3056 IF T$[1,1]="N" THEN 3060
3057 IF T$[1,1]="Y" THEN 3059
3058 OFF ERROR @ GOTO 3000
3059 OFF ERROR @ P$=P$&"H"
3060 OUTPUT 10 ;"A"&P$
3090 GOSUB 8070
3300 GOSUB 8800
3310 DISP @ DISP L$
3320 DISP @ DISP "      (1) GO TO
      MENU"
3330 DISP "      (2) GO AGAIN"
3340 GOSUB 8700
3350 ON ERROR GOTO 3300
3360 INPUT T$
```

```
3370 IF LEN(T$)>1 THEN 3300
3380 IF NUM(T$)=49 THEN 3410
3390 IF NUM(T$)=50 THEN 3060
3400 GOTO 3300
3410 OUTPUT 10 ;"O" ! CNX HOLD
3415 A=0
3420 GOTO 200
3500 GOSUB 8800
3510 L$="           HARDWARE TEST
     "
3520 DISP @ DISP L$
3530 OUTPUT 10 ;"V"
3540 GOSUB 9100
3590 IF S1<=192 THEN 3690
3600 DISP @ DISP "UNIDEX HAS A H
     ARDWARE FAILURE!"
3610 DISP "FAILURE(S) IS(ARE) AS
      FOLLOWS:"
3620 IF BIT(S1,3)=1 THEN DISP "U
     SER MEMORY ALTERED."
3630 IF BIT(S1,2)=1 THEN DISP "U
     SER MEMORY READ/WRITE FAILU
     RE."
3640 IF BIT(S1,1)=1 THEN DISP "S
     YSTEM RAM READ/WRITE FAILUR
     E."
3650 IF BIT(S1,0)=1 THEN DISP "U
     NIDEX III SOFTWARE FAILURE.
     "
3660 DISP @ DISP "CALL OR WRITE
     FACTORY FOR ASSISTANCE."
3670 S6=0 @ GOSUB 8555
3680 GOTO 3710
3690 DISP @ DISP "UNIDEX III HAR
     DWARE TEST PASSES."
3700 WAIT 5000 @ R=1
3710 GOTO 200
4980 GOSUB 8800
4990 DISP @ DISP L$ @ DISP
5000 DISP @ DISP "DISPLAY M-FUNC
     TION OUTPUT AND   C-INPUT C
     ONDITION STATUS."
5010 DISP @ DISP "(Y) YES OR (NO
     )";
5020 INPUT T$
5030 IF T$="Y" THEN 5070
5040 IF T$="N" THEN 5060
5050 GOTO 4980
5060 M=0 @ GOTO 5080
5070 M=1
5080 RETURN
5090 DISP @ DISP L$ @ DISP
5095 RETURN
5100 GOSUB 8800
5102 OUTPUT 10 ;"PS"
5103 ENTER 10 USING "#10A" ; J$
5106 S8=NUM(J$[9,9])
5107 S9=NUM(J$[10,10])
5110 DISP @ DISP L$ @ DISP
5120 DISP "    M-FUNCTION OUTPUT
      STATUS"
```

```
5130 DISP @ DISP "        M      C
     ONDITION"
5140 FOR I=8 TO 1 STEP -1
5150 DISP "      ";I;"          ";
5160 IF BIT(S8,I-1)=1 THEN DISP
     "ON"
5170 IF BIT(S8,I-1)=0 THEN DISP
     "OFF"
5180 WAIT 100
5190 NEXT I
5200 WAIT 2000
5210 GOSUB 8800
5220 DISP @ DISP L$ @ DISP
5230 DISP "          C-INPUT STAT
     US"
5240 DISP @ DISP "       C-INPUT
          STATUS"
5250 DISP
5260 FOR I=4 TO 1 STEP -1
5270 DISP "         ";I;"
       ";
5280 IF BIT(S9,I+3)=1 THEN DISP
     "HI"
5290 IF BIT(S9,I+3)=0 THEN DISP
     "LO"
5300 WAIT 100
5310 NEXT I
5320 WAIT 2000
5330 RETURN
7800 DISP "HP-85 TAPE ERROR IS "
     ;ERRN
7810 WAIT 2000
7830 RETURN
7900 ON ERROR GOTO 7900
7905 DISP @ DISP "ENTER PROGRAM
     NUMBER";
7910 INPUT P$
7920 IF LEN(P$)>2 THEN 7900
7930 I=1
7940 FOR I=1 TO LEN(P$)
7950 IF NUM(P$[I,I])<48 THEN 790
     0
7960 IF NUM(P$[I,I])>57 THEN 790
     0
7970 NEXT I
7980 OFF ERROR
7990 RETURN
8000 ! OUTPUT TO UNIDEX III
8010 GOSUB 8800
8020 DISP @ DISP L$ @ DISP
8030 DISP @ DISP "OUTPUT IS"
8040 DISP @ DISP S$
8050 J$=A$&S$
8060 OUTPUT 10 ;J$
8070 ON TIMER# 1,900000 GOTO 890
     0
8080 GOSUB 9100
8085 OFF TIMER# 1
8091 GOSUB 8800
8092 DISP @ DISP L$
```

```
8093 DISP @ DISP
8096 GOSUB 8330
8097 IF BIT(S1,7)=1 THEN GOSUB 8
     440
8098 IF S1>0 AND S1<=7 THEN GOSU
     B 8105
8104 RETURN
8105 GOSUB 8800
8110 DISP @ DISP L$
8115 DISP
8120 IF BIT(S1,0)=1 THEN DISP "I
     N 'M00' STOP (PROGRAM USES
     C-   INPUT TO CONTINUE.)" @
      DISP
8121 IF BIT(S1,0)=0 THEN 8130
8122 GOSUB 9000
8124 IF BIT(S1,0)=0 THEN RETURN
8125 GOTO 8122
8130 IF BIT(S1,1)=1 THEN DISP "
               IN 'G25' STOP"
8135 IF BIT(S1,2)=1 AND BIT(S1,1
     )=0 THEN DISP "IN HOLD MODE
     ! SEND TRIGGER"
8145 DISP @ DISP "PRESS END LINE
      KEY TO CONTINUE"
8150 ON ERROR GOTO 8105
8155 INPUT T$
8160 IF T$="" THEN 8185
8165 OFF ERROR
8170 GOTO 8105
8185 IF BIT(S1,1)=1 THEN 8210
8200 OUTPUT 10 ;"T"
8205 RETURN
8210 GOSUB 8390
8211 IF M=1 THEN GOSUB 5100
8213 IF A=1 THEN 8222
8215 GOSUB 8800
8220 DISP @ DISP L$ @ DISP
8222 OUTPUT 10
8223 GOSUB 9000
8224 IF BIT(S1,1)=1 THEN OUTPUT
     10 ;"B"
8225 RETURN
8330 OUTPUT 10 ;"PS"
8335 ENTER 10 USING "#10A" ; J$
8340 S=NUM(J$[1,1])
8345 S1=NUM(J$[2,2])
8350 S2=NUM(J$[3,3])
8355 S3=NUM(J$[4,4])
8360 S4=NUM(J$[5,5])
8365 S5=NUM(J$[6,6])
8370 S6=NUM(J$[7,7])
8375 S7=NUM(J$[8,8])
8380 S8=NUM(J$[9,9])
8385 S9=NUM(J$[10,10])
8387 IF BIT(S1,7)=1 THEN RETURN
8390 OUTPUT 10 ;"PX"
8395 ENTER 10 USING "%,%K" ; X
8400 OUTPUT 10 ;"PY"
8405 ENTER 10 USING "%,%K" ; Y
```

```
8410 DISP "SPOLL=";DTH$(S1)
8415 DISP "X POSITION IS    ";X
8420 DISP "Y POSITION IS    ";Y
8425 WAIT 1500
8430 IF S1<192 THEN S1=0
8435 RETURN
8440 GOSUB 8800
8450 DISP @ DISP "UNIDEX III HAS
      DETECTED AN ERROR IN PROGR
     AM"
8455 DISP @ DISP "THE ERROR(S) I
     S(ARE) AS FOLLOWS:"
8460 IF S5=0 THEN DISP "-NO SYNT
     AX ERRORS" @ GOTO 8505
8465 IF S5>0 THEN DISP "-SYNTAX
     ERROR-"
8470 IF BIT(S5,6)=1 THEN DISP "
            ILLEGAL CHARACTER"
8475 IF BIT(S5,5)=1 THEN DISP "
          'N' COMMAND"
8480 IF BIT(S5,4)=1 THEN DISP "
          'G' COMMAND"
8485 IF BIT(S5,3)=1 THEN DISP "
          'F' COMMAND"
8490 IF BIT(S5,2)=1 THEN DISP "
          'M' COMMAND"
8495 IF BIT(S5,1)=1 THEN DISP "
          'D' COMMAND"
8500 IF BIT(S5,0)=1 THEN DISP "
          NO COMMAND BEFORE INP
     UTT-        ING NUMERALS"
8505 IF S6=0 THEN DISP "-NO RUN
     ERRORS" @ GOTO 8540
8510 IF S6>0 THEN DISP "-RUN ERR
     OR-"
8515 IF BIT(S6,6)=1 THEN DISP "
          X AXIS LIMIT"
8520 IF BIT(S6,5)=1 THEN DISP "
          Y AXIS LIMIT"
8525 IF BIT(S6,3)=1 THEN DISP "
          STACK OVERFLOW"
8530 IF BIT(S6,1)=1 THEN DISP "
          EOB(END OF BLOCK) SEA
     RCH"
8535 IF BIT(S6,0)=1 THEN DISP "
          PROGRAM MISSING"
8540 IF BIT(S7,7)=0 THEN DISP "-
     NO EDITOR ERRORS"
8545 IF BIT(S7,7)=1 THEN DISP "-
     EDITOR ERROR-"
8550 IF BIT(S7,0)=1 THEN DISP "
          COMPILE"
8555 DISP @ DISP "PRESS END LINE
      KEY TO CONTINUE" @ ON ERRO
     R GOTO 8440 @ INPUT T$@ OFF
      ERROR
8560 IF T$#"" THEN 8440
8565 IF BIT(S6,6)=1 OR BIT(S6,5)
     =1 THEN 8575
8570 RETURN
```

```
8575 S$="G7*" @ A$="I"
8580 GOSUB 8000
8585 S$=CHR$(34)&"  HOME  "&CHR$
     (34)&"D5000*M2*"
8590 GOSUB 8000
8595 RETURN
8600 WAIT 100 @ GOSUB 8800
8610 DISP @ DISP L$ @ DISP
8620 DISP "      (1) GO TO MENU"
8630 DISP "      (2) GO AGAIN"
8640 DISP "      (3) CHANGE"
8645 GOSUB 8700
8646 ON TIMER# 1,10000 GOTO 8691
8650 INPUT T$
8655 OFF TIMER# 1
8660 IF LEN(T$)>1 THEN 8600
8670 IF NUM(T$)<49 OR NUM(T$)>51
      THEN 8600
8680 T=VAL(T$)
8690 RETURN
8691 T=1
8692 OFF TIMER# 1
8693 RETURN
8700 DISP @ DISP "  PRESS KEY TO
      ENTER";
8710 RETURN
8800 CLEAR @ DISP
8805 DISP "    UNIDEX IIIa TEST
     PROGRAM"
8810 RETURN
8900 ! CONTINUE TIMEOUT
8905 OFF TIMER# 1
8910 ABORTIO 10
8915 GOSUB 8800
8920 DISP @ DISP "SET UNIDEX III
      TO REMOTE, THEN  PRESS END
      LINE KEY"
8930 INPUT N$
8940 IF N$#"" THEN 8915
8945 RESET 10
8950 RESUME 10
8955 CONTROL 10,2 ; 7
8960 CONTROL 10,3 ; 15
8962 CONTROL 10,5 ; 0
8963 CONTROL 10,4 ; 3
8964 CONTROL 10,9 ; 129
8965 CONTROL 10,11 ; 2
8970 CONTROL 10,12 ; 3
8975 IOBUFFER J$
8979 OUTPUT 10 ;Q$
8980 GOSUB 9000
8981 IF BIT(S1,7)=0 THEN 8985
8983 GOSUB 8330
8985 IF S1>192 THEN S1=S1-192
8986 IF S1>0 AND S1<=7 THEN OUTP
     UT 10 ;"C"
8987 OUTPUT 10 ;Q$
8988 GOSUB 9000
8989 IF BIT(S1,2)=1 THEN OUTPUT
     10 ;"O"
```

```
8990 GOTO 170
9000 OUTPUT 10 ;"Q"
9010 ENTER 10 USING "%,#%K" ; J$
9015 S1=NUM(J$[1,1])
9020 J$=""
9030 RETURN
9100 GOSUB 9000
9111 IF BIT(S1,0)=1 THEN GOSUB 8
     105
9112 IF BIT(S1,1)=1 AND BIT(S1,6
     )=1 THEN GOSUB 8105
9113 IF BIT(S1,2)=1 AND BIT(S1,3
     )=0 AND BIT(S1,6)=0 THEN GO
     SUB 8105
9115 IF BIT(S1,6)=1 AND BIT(S1,1
     )=1 THEN RETURN
9117 IF BIT(S1,6)=1 THEN RETURN
9120 GOTO 9100
9900 CLEAR @ DISP @ DISP @ DISP
     @ DISP @ DISP "
      END OF"
9910 DISP @ DISP @ GOSUB 8805 @
     END
```

NOTE: A tape for the HP 85 containing these two programs is available from Aerotech, Inc. Contact Customer Service Department.

INDEX

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